AURIX™ TC4Dx user manual

About this document

Scope and purpose

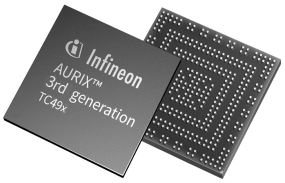
This document describes the Infineon AURIX™ TC4xx microcontrollers based on the Infineon TriCore™ architecture.

For a complete overview of the TriCore™ architecture, this document should be read in conjunction with the 'TriCore™ architecture manual'. Different derivatives of the AURIX™ may contain a different mix of functional blocks and features, but the TriCore™ architecture remains constant to preserve compatibility across all designs, and so it is described separately.

Intended audience

The primary intended audience are potential and actual users of a AURIX™ TC4xx platform product who need a detailed description and reference of the available functional blocks.

TC4xx overview



The AURIX™ TC4xx family of products is the 3rd generation of the Infineon multi-core microcontroller platform for automotive real-time applications in powertrain, chassis applications, safety applications, and advanced driver assistance systems.

This next generation of the AURIX™ platform delivers increased compute performance and a host of additional features to build on the success of the earlier platforms:

- Safe, real-time compute performance through the TriCore™ 1.8 multi-core architecture, up to 500 MHz and fully lock-stepped for ASIL-D safety level
- CPU virtualization and system virtualization support
- Up to 24 MB of on-chip non-volatile memory (NVM) and low-latency NVM access
- Support for software-over-the-air (SOTA) updates with extended A/B swap functionality
- Cyber security features and enhanced cyber security performance
- Low-latency, high-speed communication interfaces, 5 Gbps Ethernet and 8 Gbps PCIe, to address electrical and electronic (E/E) automotive architectures for domain control, gateway, and data fusion companion applications
- Enhanced signal processing performance for powertrain and sensor data fusion applications
- Low-latency real-time signal processing for powertrain and xEV (electric vehicle) applications
- Support for artificial intelligence algorithms, especially neural network inference algorithms
- Architecture enhancements to support the next generation of electrified powertrain applications
- All features designed to meet the functional safety standard ISO 26262
- Developed in 28 nm silicon technology to enable higher integration and lower power consumption

Table of contents

About this document . . . 1
TC4xx overview . . . 1
Table of contents . . . 2
1 Using this document . . . 4
2 System address map . . . 14
3 Distributed functionality . . . 55
4 Central Processing Unit (CPU) . . . 103
5 Non-Volatile Memory (NVM) . . . 642
6 Shared Resource Interconnect (SRI) . . . 994
7 Flexible Peripheral Interconnect (FPI) . . . 1086
8 LLI - NDA only . . . 1204
9 System Direct Memory Access (SDMA) . . . 1205
10 Local Memory Unit (LMU) . . . 1348
11 CSRM - NDA only . . . 1389
12 CSS - NDA only . . . 1389
13 PPU - NDA only . . . 1389
14 Gigabit Ethernet (GETH) . . . 1390
15 LETH - NDA only . . . 2429
16 HSPHY - NDA only . . . 2429
17 High Speed Serial Link (HSSL) . . . 2430
18 eXpanded Serial Peripheral Interface (xSPI) . . . 2598
19 Peripheral Component Interconnect Express (PCIe) . . . 2700
20 Data Routing Engine (DRE) . . . 3721
21 Controller Area Network interface (CAN) . . . 3918
22 Controller Area Network XL interface (CANXL) . . . 4165
23 FlexRay™ controller (ERAY) . . . 4427
24 Asynchronous/Synchronous Interface (ASCLIN) . . . 4628
25 Single Edge Nibble Transmission (SENT) . . . 4787
26 Peripheral Sensor Interface (PSI5) . . . 4855
27 Peripheral Sensor Interface with Serial Interface to PHY (PSI5-S) . . . 4966
28 Micro Second Channel (MSC) . . . 5078

Table of contents

29 Secure Digital Multi-Media Card (SDMMC) 5193
30 Queued Synchronous Peripheral Interface (QSPI) 5295
31 Inter-Integrated Circuit (I2C) 5432
32 CLOCK 5525
33 Logic-Test (TRI) 5680
34 Volatile Memory Test (VMT) 5733
35 System Control Unit (SCU) 5872
36 Power Management System (PMS) 5946
37 System Mode Management (SMM) 6315
38 Voltage and Temperature Monitors (VTMON) 6449
39 Stand-by Controller (SCR) 6479
40 Interrupt Router (IR) 6857
41 Watchdog Timer Unit (WTU) 6990
42 Safety and security alarm Management Unit (SMU) 7056
43 Flexible CRC Engine (FCE) 7270
44 Enhanced Generic Timer Module (eGTM) 7307
45 Analog-to-Digital Converter (ADC) 8309
46 Debug and Trace 8577
47 Ports 8599
48 Firmware 8773
A Product related documents 9013
Revision history 9014
Disclaimer 9015

1 Using this document

Topics:

- Understanding the platform
- Understanding the document structure
- Functional block (FB) chapter structure
- Document conventions
- Registers: How are they documented?
- Register image linking
- Register bit-field access types

These topics provide important information on how to work with and understand this documentation collection.

***Note**:* *In this documentation there might be references to functional blocks which are not part of this device. All available functional blocks are shown in the table of contents.*

There could also be references to other devices than TC4Dx. This information is required to get the necessary overview and context.

1.1 Understanding the AURIX™ platform

The following graphic helps visualize the AURIX™ platform and explains some common terminology. Each platform generation consists of a product family of devices. Each device is made up of functional clusters. Each cluster is made up of functional blocks, and those blocks may contain separate functions (functional sub-blocks).

***Note**: In this documentation, TC4xx and TC4x are both used as family short name for AURIX™ 3rd generation.*

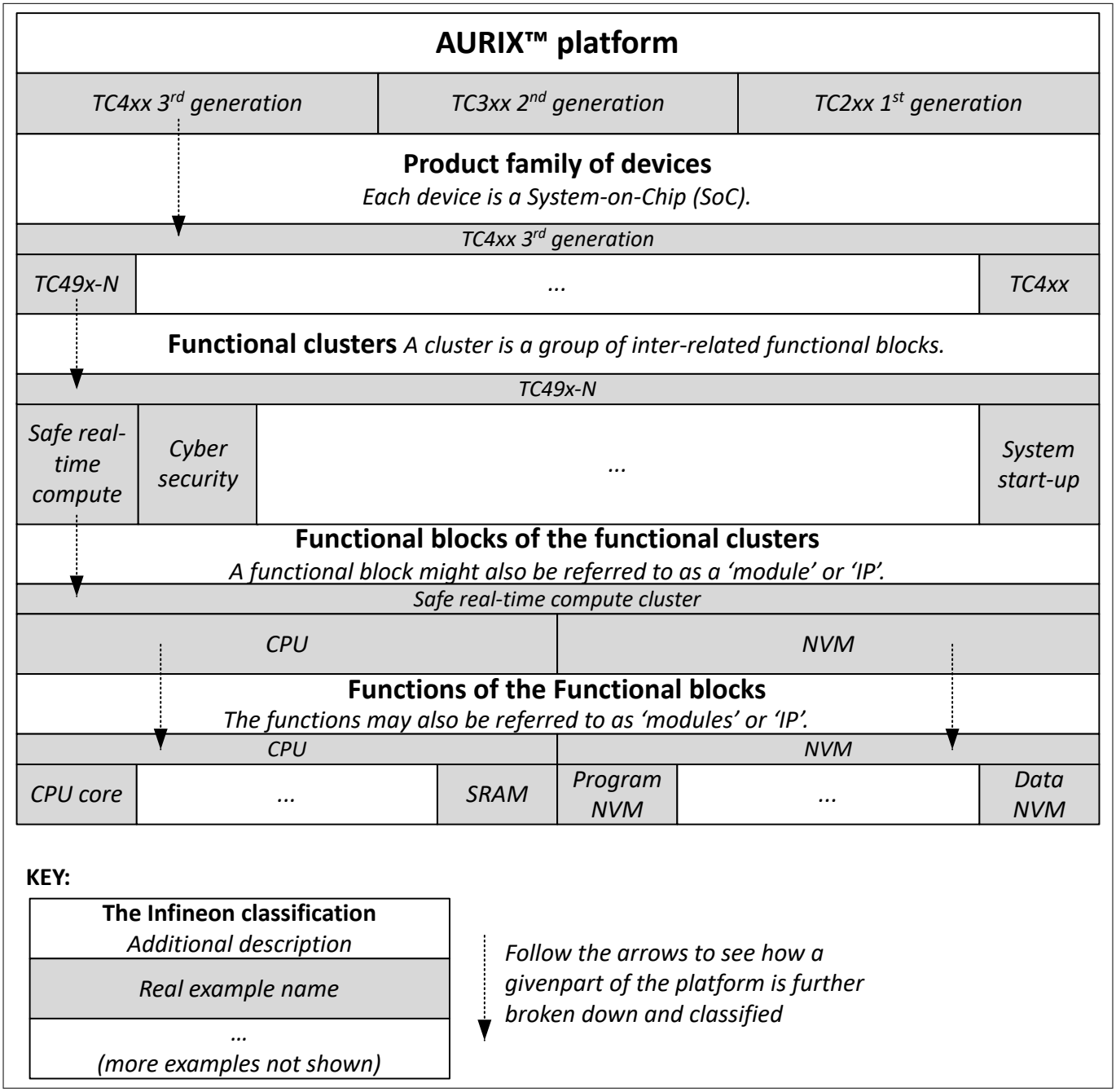


Figure 1 Visualizing the AURIX™ platform

1.2 Understanding the document structure

This document is divided up into the following major sub-sections:

- Preface
  - About this document
  - TC4xx overview
- Using this document
  - Understanding the AURIX™ platform
  - Understanding the document structure (this section)
  - Functional block (FB) chapter structure
  - Document conventions
  - Registers: How are they documented?
  - Register image linking
  - Register bit-field access types
- Features of the functional blocks
  - Up to 5 significant features of each functional block
- System address map
  - The 32-bit address ranges assigned to resources and reserved areas
  - The access capabilities of hardware masters
- Distributed functionality
  - Covers functionality that is shared by multiple functional blocks:
    - System interconnect connects the various master agents and slave agents in AURIX™ and allows them to communicate with each other
    - Software over the air (SOTA) enables software executing out of the application PFLASH (not PFLASHcs) to be updated without impacting the currently executing application
    - Resource access latencies describe CPU access times to various resources in SRI clock cycles
    - Module reset describes the kernel reset function
    - Debug and Trace shows an OTGB overview
- Functional blocks
  - Each functional block (module) has the same top-level topic structure. so for example an introduction paragraph is always followed by a feature list, then a functional overview that contains a block diagram
    - Exceptions to the standard Functional blocks
      - CSRM (only Feature list and overview are provided)
      - CSS (only feature list and overview are provided)
      - Debug and Trace (only feature list, overview, and description are provided)
- Product related documents
  - A list of related documentation that forms the complete document set for a given product
- Revision history
  - A list of major changes for the current document

***Note**:* *Each Functional block has its own revision history, once for the generic part and once for the device specific information*

1.3 Functional block (FB) chapter structure

The majority of functional block (also known as a module or IP) follow the same high-level chapter structure:

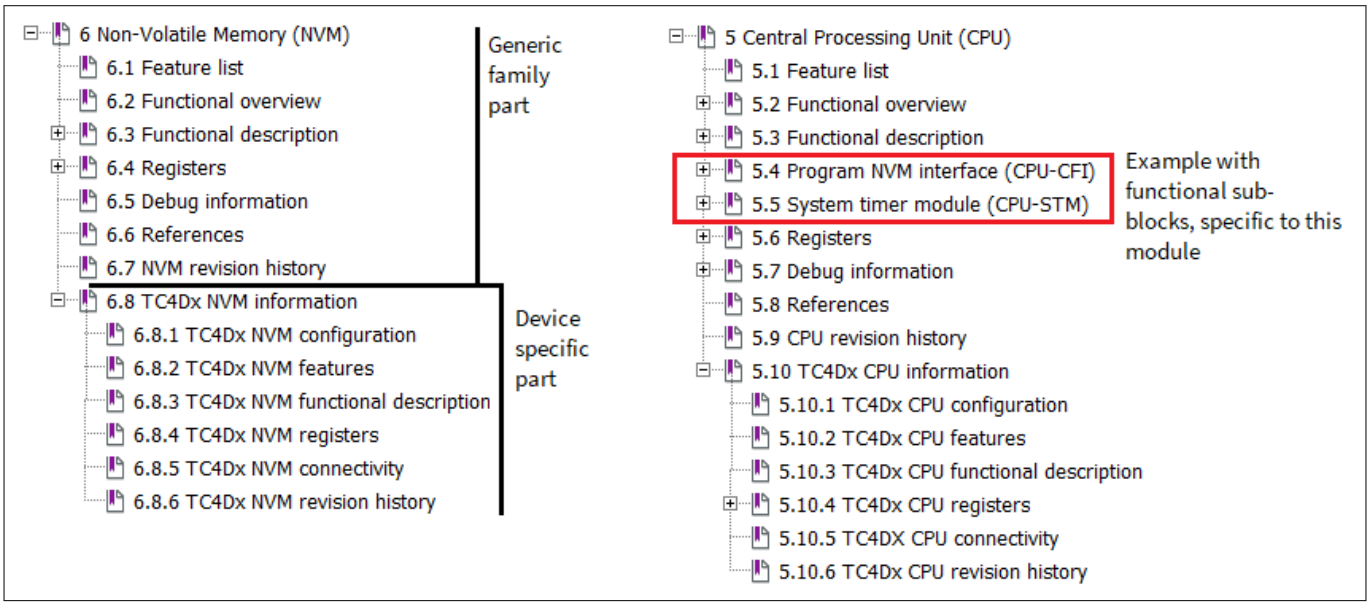


Figure 2 Standard functional block chapter structure

- Introduction paragraph under the functional block name written out in full. For example Central Processing Unit (CPU)
  - Feature list
    - A bullet list of significant features or functions of that block
  - Functional overview
    - A block diagram of the functional block
    - A brief summary intended to give a general understanding of the blocks capabilities and the capabilities of any of its functional sub-blocks (FSBs)
    - The information is generic and not device specific
  - Functional description
    - How to use the functional block. The description may include:
      - Data flow or transfer diagrams, functional timing diagrams, state transition and mode management, address modes, sequence or activity diagrams, alarms, interrupts, error management, triggers, bus interfaces
      - The information is generic and not device specific
  - Functional sub-blocks (FSBs)
    - Feature list
      - A list of the major technical features
    - Functional overview
      - A block diagram
      - A summary of the FSB capabilities
    - Functional description
      - Description of the FSB and how to use it, including information on bus interfaces, elements and signals
  - Registers
    - Register access mode glossary table
    - Register overview table with hyperlinks to each register
    - Description and definition of each register

- Debug information
- References
  - All referenced documents in the FB or FSB description
- Revision history for the sections above, which describe the generic functional block
- Device specific information for this functional block. Each heading is preceded by the device name TC4xx
  - Configuration
    - A list of major technical configuration changes for the current functional block in the context of the given device, as compared to the generic description (above), if applicable
  - Feature list
    - A list of device specific features, if applicable
  - Functional description
    - A description of any features that are listed in the feature list
  - Registers
    - Registers overview table with hyperlinks to the device specific register (if applicable) or to the generic register description
    - Device specific registers
      - This is a delta to the generic register part
  - Connectivity
    - Input and output signals
  - Device specific revision history for this sub-section

1.4 Document conventions

An overview of the common units, data formats, and other text conventions used in this document.

- Unit abbreviations:
  - MHz = Megahertz
    - 1 MHz is 1,000,000 Hz
  - µs = Microseconds
    - The 'µ' prefix scales by 0.000001
  - kBaud, kbit/s = 1,000 characters or bits per second
  - MBaud, Mbit/s = 1,000,000 characters or bits per second
  - Gbps = 1,000,000,000 characters or bits per second
  - Kbyte, KB = 1,024 bytes of memory
    - Synonymous term: Kibibyte, KiB
  - Mbyte, MB = 1,048,576 bytes of memory
    - Synonymous term: Mebibyte, MiB
  - Scaling summary
    - The 'k' prefix (lower case) scales a unit by 1,000. So the kBaud unit scales the preceding expression by 1,000
    - The 'K' prefix (upper case) scales a unit by 1,024. So the Kbyte unit scales the preceding expression by 1,024
    - The 'M' prefix scales by 1,000,000 or 1,048,576
      - 1 Mbyte is 1,024 x 1,024 bytes
      - 1 MBaud, Mbit/s are 1,000,000 characters/bits per second
- Data format quantities
  - Byte = 8-bit quantity
  - Half-word = 16-bit quantity
  - Word = 32-bit quantity
  - Double-word = 64-bit quantity
- Radix
  - The default radix is decimal
  - Hexadecimal constants are suffixed with a subscript letter 'H'. For example:
    - $AA80\ 0000_H$
  - Binary constants are suffixed with a subscript letter B. For example:
    - 1010_B
- Pins that use negative logic are indicated by an over-line. For example:
  - “The external reset pin $\overline{\text{ESR0}}$ has a dual function”
- Functional blocks (modules) are usually referenced by their abbreviation in plain upper case. For example:
  - “The QSPI supports full-duplex and half-duplex synchronous communication"”
- Registers
  - Bit-fields and bits in registers are in general referenced as either:
    - FunctionalBlock_register-name.bit-field

    - FunctionalBlock_register-name.bit
    - For example: “The current CPU priority number bit-field CPU_ICR.CCPN is cleared”
  - Most register names contain a functional block (module) abbreviation prefix which is separated from the actual register name by an underscore '_' character. For example:
    - “QSPI_GLOBALCON”
      - QSPI is the functional block abbreviation prefix
      - GLOBALCON is the kernel register name
        - In topics describing the kernels of peripherals, the registers are usually referenced by the kernel register name
  - Sets of registers may have a lower-case letter appended to indicate a value
    - “MOFCRn” for example, refers to multiple MOFCR registers of number 'n'
      - The bounds of the value 'n' are always defined where the register expression is first used, although the bounds may also be repeated throughout the text. For example:
        - “n = 0 - 255”
- Text conventions for register bit ranges, signals, and pins
  - Register fields, groups of register bits, and groups of pins are collectively referenced in the text in the format 'NAME[A:B]'. This indicates a range for the named group from A to B. For example:
    - CFG[2:0]
  - Individual bits, signals, or pins may be referenced as 'NAME[C]'. For example:
    - SRNP[0]

1.5 Registers: How are they documented?

In each functional block chapter, registers are grouped into both 'family' and 'device specific' sections. The family grouping provides a concise overview of all the registers available for that functional block in the product family. The device specific section gives detailed information on the applicable registers for the specific device.

***Note**: The exceptions to the description given here, are for CSRM, CSS, and Debug and Trace. CSRM and CSS are fully documented in separate documentation that is available on request. The Debug and Trace chapter follows conventions introduced in previous AURIX™ family documentation, and does not follow the standard format of other functional blocks.*

1.6 Register image linking

The register images contain hyperlinks from each bit-field, to the bit-field definition in the register description table that follows the image.

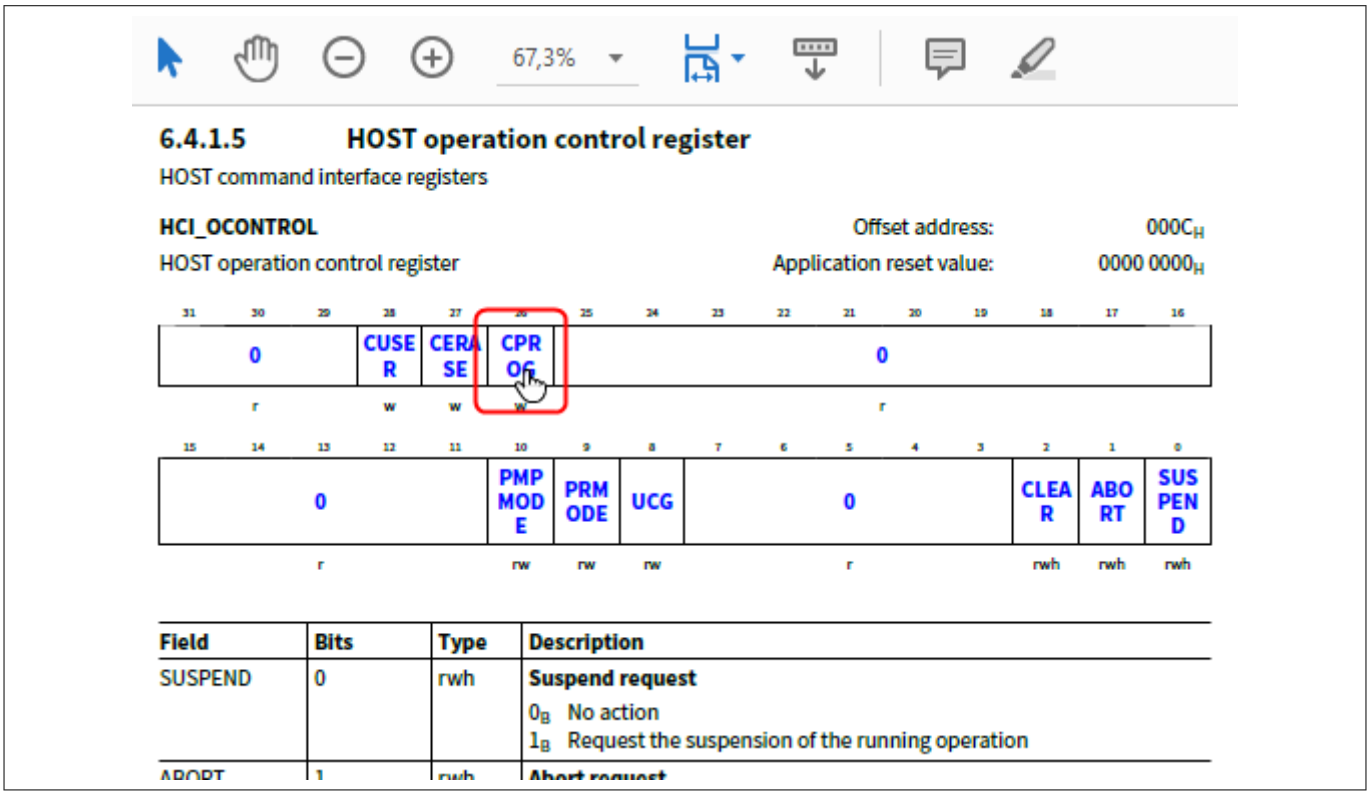


Figure 3 Bit-field hyperlink

If you have the PDF set to 'scroll' (the options to do this depend on the PDF viewer) then selecting the hyperlink will jump you to the corresponding row in the register description table. If the PDF is set to view 'per page' then the link will take you to the top of the page on which that link exists.

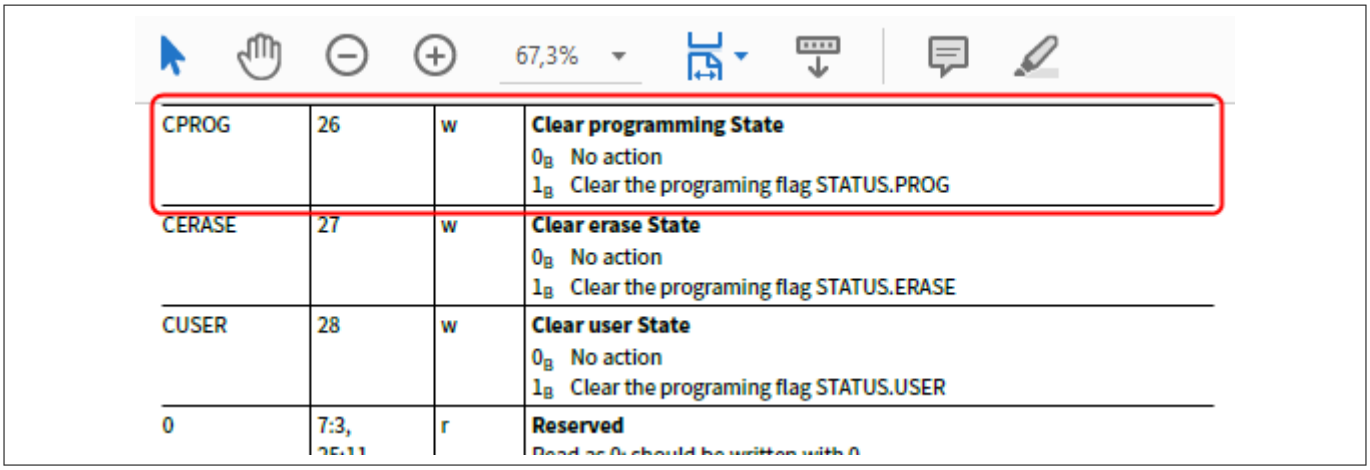


Figure 4 Hyperlink moves you to the correct bit-field

1.7 Register bit-field access types

Table 1 Access types

Access type	Description
r	Software can only read this bit-field. A software write will not change the value of the bit-field.
rh	Software can only read this bit-field. A software write will not change the value of the bit-field. Hardware can change the value of the bit-field.
rc	Reading returns the current value and clears the bit-field to '0'. A software write will not change the value of the bit-field.
w	Software can write this bit-field. Read returns 0 unless explicitly described differently.
wh[1)]	Software can only write this bit-field. Hardware can change the value of the bit-field. Read returns 0 unless explicitly described differently.
rw	Software can both read and write this bit-field.
rwh[1)]	Software can both read and write this bit-field. Hardware can change the value of the bit-field.
rw1s[2)]	Writing '1' to a bit inside this bit-field sets that bit to '1'. Writing '0' has no effect. Read returns the current value.
rw1sh[1)]	If the bit-field written value is '1', the corresponding bit-field is set to '1'. Otherwise, writing '0' has no effect. Hardware can change the value of the bit-field. Read returns the current value.
rw0c[2)]	Writing '0' to a bit inside this bit-field clears that bit to '0'. Otherwise, writing '1' has no effect. Read returns the current value.
rw0ch[1)]	If the bit-field written value is '0', the corresponding bit-field is cleared to ‘0’. Otherwise, writing '1' has no effect. Hardware can change the value of the bit-field. Read returns the current value.
rw1c[2)]	Writing '1' to a bit inside this bit-field clears that bit to '0'. Otherwise, writing '0' has no effect. Read returns the current value.
rw1ch[1)]	If the bit-field written value is '1', the corresponding bit-field is cleared to ‘0’. Otherwise, writing '0' has no effect. Hardware can change the value of the bit-field. Read returns the current value.

1) Performing a read-modify-write (RMW) operation on this bit-field should be avoided as hardware may change the values of the bit-field between READ and WRITE sub-operations of an RMW operation.

2) This access type is sometimes referred to as "sticky" or "write-once", because the bit-field value can be changed only once after each reset.

2 System address map

The system address map defines:

- The 32-bit address ranges assigned to resources and reserved areas
- The access capabilities of hardware masters

2.1 Address segments and accesses

In general, all hardware masters can address peripherals and memories at global addresses.

2.1.1 Acronyms and terms

Key terms used in the system address map.

Table 2 Address map related acronyms and terms

Acronym or term	Definition
Access	A bus access is allowed and is executed.
COMPBBE	A bus access is terminated with a bus error on the COMPB.
CPBBE	A bus access is terminated with a bus error on the CPB.
CSPBBE	A bus access is terminated with a bus error on the CSPB.
LLIBE	A bus access is terminated with a bus error on the LLI.
RCBBE	A bus access is terminated with a bus error on the RCB.
SPBBE	A bus access is terminated with a bus error on the SPB.
SRIBE	A bus access is terminated with a bus error on the SRI.
TPBBE	A bus access is terminated with a bus error on the TPB.

2.1.2 Overview of segments

The address space is divided into sixteen segments. Each segment has a defined purpose.

Segment 0

This segment is reserved.

Segments 1-7

These segments provide access to the following CPU memories:

- Program Scratch-Pad RAM (PSPR)
- Program Cache (PCACHE)
  - If the PCACHE is disabled and mapped into the address space then the PSPR (extension) is available
- Program Tag (PTAG) related to Program Cache (PCACHE)
  - If the PCACHE is disabled then the PTAG can be mapped into the address space
- Data Scratch-Pad RAM (DSPR)
- Data Cache (DCACHE)
  - If the DCACHE is disabled and mapped into the address space then the DSPR (extension) is available
- Data Tag (DTAG) related to Data Cache (DCACHE)
  - If the DCACHE is disabled then the DTAG can be mapped into the address space

Segment 8

This segment provides an alias for the cached access to the non-volatile memory in Segment 10.

Segment 9

This segment provides an alias for the cached access to the volatile memory in Segment 11.

Segment 10

This segment provides non-cached access to non-volatile memory.

Segment 11

This segment provides non-cached access to volatile memory.

Segment 12

This segment is not globally addressable and is reserved for certain compute agents.
Each CPUx has an internal access to CPUx.PSPR through segment C_H . Refer to the CPU chapter for further details.
The PPU has an internal access to PPU.VMEM through segment C_H . Refer to the PPU chapter for further details.

Segment 13

This segment is not globally addressable and is reserved for certain compute agents.
Each CPUx has an internal access to CPUx.DSPR through segment D_H . Refer to the CPU chapter for further details.

Segment 14

This segment provides access to peripheral space.

Segment 15

This segment provides access to peripheral space.

2.1.3 Memory accesses

Topics:

- Standard read-write memories
- Standard read-only memories
- Non-standard memories
- Dedicated memories

This section lists the memory access sizes to the following types of memory: standard read-write memories, standard read-only memories, non-standard memories, and dedicated memories.

Table 3 Key to memory access tables

Notation	Explanation
r	read
rmw	read modify write
w	write
y	'yes', access is supported
-	access is not supported

2.1.3.1 Standard read-write memories

Table 4 Standard read-write memories

Memory	1 Byte		2 Bytes		4 Bytes			8 Bytes		16 Bytes		32 Bytes		64 Bytes	
	r	w	r	w	r	w	rmw	r	w	r	w	r	w	r	w
SCR.XRAM [1]	y	y	y	y	y	y	y	-	-	-	-	-	-	-	-
CSS	y	y	y	y	y	y	-	y	y	y	y	y	y	y	y
CPUx.PSPR	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y
CPUx.DSPR	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y
CPUx.DLMU	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y
CANx.SRAM	-	-	-	-	y	y	y	y	y	y	y	-	-	-	-
GETH0.RAM	-	-	-	-	-	-	-	y	y	y	y	y	y	y	y
PPU.CSM															
PPU.VMEM															
LMU	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y

1) Accesses supported by software executed on a CPU.
2) Accesses are atomic for PPU internal accesses. CPU rmw accesses to PPU internal memories should only be executed when the PPU is not accessing the corresponding memory.

2.1.3.2 Standard read-only memories

Table 5 Standard read-only memories

Memory	1 Byte	2 Bytes	4 Bytes	8 Bytes	16 Bytes	32 Bytes	64 Bytes
	r	r	r	r	r	r	r
NVM.PFLASH	y	y	y	y	y	y	y[1]
NVM.EEPROM	y	y	y	y	-	-	-
NVM.UCB	y	y	y	y	-	-	-
NVM.BROM	y	y	y	y	y	y	-
xSPI	y	y	y	y	y	y	-

1) Only external accesses from the SRI interconnect.

2.1.3.3 Non-standard memories

Table 6 Non-standard memories

- The TAG memories (CPUx.PTAG and CPUx.DTAG) must not be used as general SRAMs
- To support memory testing, the TAG memories may be mapped into the system address map
- Test software must perform read and write accesses to a TAG memory using 64-bit aligned addresses and 32-bit data size
- The upper data bits of a TAG memory are not implemented

Memory	1 Byte		2 Bytes		4 Bytes			8 Bytes		16 Bytes		32 Bytes		64 Bytes	
	r	w	r	w	r	w	rmw	r	w	r	w	r	w	r	w
CPUx.PTAG	-	-	-	-	y	y	-	-	-	-	-	-	-	-	-
CPUx.DTAG	-	-	-	-	y	y	-	-	-	-	-	-	-	-	-

2.1.3.4 Dedicated memories

The following memories are dedicated resources:

- SCR.ROM is only accessible by the SCR
- CSROM is only accessible by the CPUcs

2.2 Address map

The address maps define which address ranges are valid or invalid for the respective segment. The access type column describes the error status location if the access is not permitted.

For the detailed address space of a specific functional block (module), please to the specific user manual chapter describing that block.

2.2.1 Segment 0

Table 7 Address map of segment 0

Address range	Size	Description	Access type	
			Read	Write
0000 0000$_H$ - 0FFF FFFF$_H$	-	Reserved	SRIBE [1]	SRIBE [1]

1) Any CPU load or store operation accessing 0000 0000$_H$ will trap

2.2.2 Segment 1

Table 8 Address map of segment 1

Address range	Size	Description	Access type	
			Read	Write
1000 0000$_H$ - 1003 BFFF$_H$	240 Kbyte	CPUcs.DSPR	Access	Access
1003 C000$_H$ - 1003 FFFF$_H$	16 Kbyte	CPUcs.DSPR (extension) or CPUcs.DCACHE (RAM)	Access [1] / SRIBE	Access [1] / SRIBE
1004 0000$_H$ - 100F FFFF$_H$	-	Reserved	SRIBE	SRIBE
1010 0000$_H$ - 1010 FFFF$_H$	64 Kbyte	CPUcs.PSPR	Access	Access
1011 0000$_H$ - 1011 7FFF$_H$	32 Kbyte	CPUcs.PSPR (extension) or CPUcs.PCACHE (RAM)	Access [1] / SRIBE	Access [1] / SRIBE
1011 8000$_H$ - 1011 FFFF$_H$	-	Reserved	SRIBE	SRIBE
1012 0000$_H$ - 1012 FFFF$_H$	64 Kbyte	Alias for CPUcs.PSPR	Access	Access
1013 0000$_H$ - 1013 7FFF$_H$	32 Kbyte	Alias for CPUcs.PSPR (extension)	Access	Access
1013 8000$_H$ - 1013 FFFF$_H$	-	Reserved	SRIBE	SRIBE
1014 0000$_H$ - 1014 FFFF$_H$	64 Kbyte	Alias for CPUcs.PSPR	Access	Access
1015 0000$_H$ - 1015 7FFF$_H$	32 Kbyte	Alias for CPUcs.PSPR (extension)	Access	Access
1015 8000$_H$ - 1015 FFFF$_H$	-	Reserved	SRIBE	SRIBE
1016 0000$_H$ - 1016 FFFF$_H$	64 Kbyte	Alias for CPUcs.PSPR	Access	Access
1017 0000$_H$ - 1017 7FFF$_H$	32 Kbyte	Alias for CPUcs.PSPR (extension)	Access	Access
1017 8000$_H$ - 1017 FFFF$_H$	-	Reserved	SRIBE	SRIBE
1018 0000$_H$ - 1018 FFFF$_H$	64 Kbyte	Alias for CPUcs.PSPR	Access	Access
1019 0000$_H$ - 1019 7FFF$_H$	32 Kbyte	Alias for CPUcs.PSPR (extension)	Access	Access

(table continues...)

Table 8 (continued) Address map of segment 1

Address range	Size	Description	Access type	
			Read	Write
$1019\ 8000_H$ - $1019\ FFFF_H$	-	Reserved	SRIBE	SRIBE
$101A\ 0000_H$ - $101A\ FFFF_H$	64 Kbyte	Alias for CPUcs.PSPR	Access	Access
$101B\ 0000_H$ - $101B\ 7FFF_H$	32 Kbyte	Alias for CPUcs.PSPR (extension)	Access	Access
$101B\ 8000_H$ - $101B\ FFFF_H$	-	Reserved	SRIBE	SRIBE
$101C\ 0000_H$ - $101C\ FFFF_H$	64 Kbyte	Alias for CPUcs.PSPR	Access	Access
$101D\ 0000_H$ - $101D\ 7FFF_H$	32 Kbyte	Alias for CPUcs.PSPR (extension)	Access	Access
$101D\ 8000_H$ - $101D\ FFFF_H$	-	Reserved	SRIBE	SRIBE
$101E\ 0000_H$ - $101E\ FFFF_H$	64 Kbyte	Alias for CPUcs.PSPR	Access	Access
$101F\ 0000_H$ - $101F\ 7FFF_H$	32 Kbyte	Alias for CPUcs.PSPR (extension)	Access	Access
$101F\ 8000_H$ - $101F\ FFFF_H$	-	Reserved	SRIBE	SRIBE
$1020\ 0000_H$ - $1020\ 17FF_H$	-	CPUcs.DTAG	Access [1] / SRIBE	Access [1] / SRIBE
$1020\ 1800_H$ - $102F\ FFFF_H$	-	Reserved	SRIBE	SRIBE
$1030\ 0000_H$ - $1030\ 2FFF_H$	-	CPUcs.PTAG	Access [1] / SRIBE	Access [1] / SRIBE
$1030\ 3000_H$ - $1FFF\ FFFF_H$	-	Reserved	SRIBE	SRIBE

1) PCACHE/PTAG and DCACHE/DTAG can be only accessed when mapped into the address space

2.2.3 Segment 2

Table 9 Address map of segment 2

Address range	Size	Description	Access type	
			Read	Write
$2000\ 0000_H$ - $2003\ BFFF_H$	240 Kbyte	CPU5.DSPR	Access	Access
$2003\ C000_H$ - $2003\ FFFF_H$	16 Kbyte	CPU5.DSPR (extension) or CPU5.DCACHE (RAM)	Access [1] / SRIBE	Access [1] / SRIBE
$2004\ 0000_H$ - $200F\ FFFF_H$	-	Reserved	SRIBE	SRIBE
$2010\ 0000_H$ - $2010\ FFFF_H$	64 Kbyte	CPU5.PSPR	Access	Access
$2011\ 0000_H$ - $2011\ 7FFF_H$	32 Kbyte	CPU5.PSPR (extension) or CPU5.PCACHE (RAM)	Access [1] / SRIBE	Access [1] / SRIBE
$2011\ 8000_H$ - $2011\ FFFF_H$	-	Reserved	SRIBE	SRIBE
$2012\ 0000_H$ - $2012\ FFFF_H$	64 Kbyte	Alias for CPU5.PSPR	Access	Access
$2013\ 0000_H$ - $2013\ 7FFF_H$	32 Kbyte	Alias for CPU5.PSPR (extension)	Access	Access
$2013\ 8000_H$ - $2013\ FFFF_H$	-	Reserved	SRIBE	SRIBE
$2014\ 0000_H$ - $2014\ FFFF_H$	64 Kbyte	Alias for CPU5.PSPR	Access	Access
$2015\ 0000_H$ - $2015\ 7FFF_H$	32 Kbyte	Alias for CPU5.PSPR (extension)	Access	Access
$2015\ 8000_H$ - $2015\ FFFF_H$	-	Reserved	SRIBE	SRIBE
$2016\ 0000_H$ - $2016\ FFFF_H$	64 Kbyte	Alias for CPU5.PSPR	Access	Access
$2017\ 0000_H$ - $2017\ 7FFF_H$	32 Kbyte	Alias for CPU5.PSPR (extension)	Access	Access
$2017\ 8000_H$ - $2017\ FFFF_H$	-	Reserved	SRIBE	SRIBE
$2018\ 0000_H$ - $2018\ FFFF_H$	64 Kbyte	Alias for CPU5.PSPR	Access	Access
$2019\ 0000_H$ - $2019\ 7FFF_H$	32 Kbyte	Alias for CPU5.PSPR (extension)	Access	Access
$2019\ 8000_H$ - $2019\ FFFF_H$	-	Reserved	SRIBE	SRIBE
$201A\ 0000_H$ - $201A\ FFFF_H$	64 Kbyte	Alias for CPU5.PSPR	Access	Access
$201B\ 0000_H$ - $201B\ 7FFF_H$	32 Kbyte	Alias for CPU5.PSPR (extension)	Access	Access
$201B\ 8000_H$ - $201B\ FFFF_H$	-	Reserved	SRIBE	SRIBE
$201C\ 0000_H$ - $201C\ FFFF_H$	64 Kbyte	Alias for CPU5.PSPR	Access	Access
$201D\ 0000_H$ - $201D\ 7FFF_H$	32 Kbyte	Alias for CPU5.PSPR (extension)	Access	Access
$201D\ 8000_H$ - $201D\ FFFF_H$	-	Reserved	SRIBE	SRIBE
$201E\ 0000_H$ - $201E\ FFFF_H$	64 Kbyte	Alias for CPU5.PSPR	Access	Access
$201F\ 0000_H$ - $201F\ 7FFF_H$	32 Kbyte	Alias for CPU5.PSPR (extension)	Access	Access
$201F\ 8000_H$ - $201F\ FFFF_H$	-	Reserved	SRIBE	SRIBE
$2020\ 0000_H$ - $2020\ 17FF_H$	-	CPU5.DTAG	Access [1] / SRIBE	Access [1] / SRIBE
$2020\ 1800_H$ - $202F\ FFFF_H$	-	Reserved	SRIBE	SRIBE

(table continues...)

Table 9 (continued) Address map of segment 2

Address range	Size	Description	Access type	
			Read	Write
$2030\ 0000_H$ - $2030\ 2FFF_H$	-	CPU5.PTAG	Access [1] / SRIBE	Access [1] / SRIBE
$2030\ 3000_H$ - $2FFF\ FFFF_H$	-	Reserved	SRIBE	SRIBE

1) PCACHE/PTAG and DCACHE/DTAG can be only accessed when mapped into the address space

2.2.4 Segment 3

Table 10 Address map of segment 3

Address range	Size	Description	Access type	
			Read	Write
$3000\ 0000_H$ - $3003\ BFFF_H$	240 Kbyte	CPU4.DSPR	Access	Access
$3003\ C000_H$ - $3003\ FFFF_H$	16 Kbyte	CPU4.DSPR (extension) or CPU4.DCACHE (RAM)	Access [1] / SRIBE	Access [1] / SRIBE
$3004\ 0000_H$ - $300F\ FFFF_H$	-	Reserved	SRIBE	SRIBE
$3010\ 0000_H$ - $3010\ FFFF_H$	64 Kbyte	CPU4.PSPR	Access	Access
$3011\ 0000_H$ - $3011\ 7FFF_H$	32 Kbyte	CPU4.PSPR (extension) or CPU4.PCACHE (RAM)	Access [1] / SRIBE	Access [1] / SRIBE
$3011\ 8000_H$ - $3011\ FFFF_H$	-	Reserved	SRIBE	SRIBE
$3012\ 0000_H$ - $3012\ FFFF_H$	64 Kbyte	Alias for CPU4.PSPR	Access	Access
$3013\ 0000_H$ - $3013\ 7FFF_H$	32 Kbyte	Alias for CPU4.PSPR (extension)	Access	Access
$3013\ 8000_H$ - $3013\ FFFF_H$	-	Reserved	SRIBE	SRIBE
$3014\ 0000_H$ - $3014\ FFFF_H$	64 Kbyte	Alias for CPU4.PSPR	Access	Access
$3015\ 0000_H$ - $3015\ 7FFF_H$	32 Kbyte	Alias for CPU4.PSPR (extension)	Access	Access
$3015\ 8000_H$ - $3015\ FFFF_H$	-	Reserved	SRIBE	SRIBE
$3016\ 0000_H$ - $3016\ FFFF_H$	64 Kbyte	Alias for CPU4.PSPR	Access	Access
$3017\ 0000_H$ - $3017\ 7FFF_H$	32 Kbyte	Alias for CPU4.PSPR (extension)	Access	Access
$3017\ 8000_H$ - $3017\ FFFF_H$	-	Reserved	SRIBE	SRIBE
$3018\ 0000_H$ - $3018\ FFFF_H$	64 Kbyte	Alias for CPU4.PSPR	Access	Access
$3019\ 0000_H$ - $3019\ 7FFF_H$	32 Kbyte	Alias for CPU4.PSPR (extension)	Access	Access
$3019\ 8000_H$ - $3019\ FFFF_H$	-	Reserved	SRIBE	SRIBE
$301A\ 0000_H$ - $301A\ FFFF_H$	64 Kbyte	Alias for CPU4.PSPR	Access	Access
$301B\ 0000_H$ - $301B\ 7FFF_H$	32 Kbyte	Alias for CPU4.PSPR (extension)	Access	Access
$301B\ 8000_H$ - $301B\ FFFF_H$	-	Reserved	SRIBE	SRIBE
$301C\ 0000_H$ - $301C\ FFFF_H$	64 Kbyte	Alias for CPU4.PSPR	Access	Access
$301D\ 0000_H$ - $301D\ 7FFF_H$	32 Kbyte	Alias for CPU4.PSPR (extension)	Access	Access
$301D\ 8000_H$ - $301D\ FFFF_H$	-	Reserved	SRIBE	SRIBE
$301E\ 0000_H$ - $301E\ FFFF_H$	64 Kbyte	Alias for CPU4.PSPR	Access	Access
$301F\ 0000_H$ - $301F\ 7FFF_H$	32 Kbyte	Alias for CPU4.PSPR (extension)	Access	Access
$301F\ 8000_H$ - $301F\ FFFF_H$	-	Reserved	SRIBE	SRIBE
$3020\ 0000_H$ - $3020\ 17FF_H$	-	CPU4.DTAG	Access [1] / SRIBE	Access [1] / SRIBE
$3020\ 1800_H$ - $302F\ FFFF_H$	-	Reserved	SRIBE	SRIBE

(table continues...)

Table 10 (continued) Address map of segment 3

Address range	Size	Description	Access type	
			Read	Write
$3030\ 0000_H$ - $3030\ 2FFF_H$	-	CPU4.PTAG	Access [1] / SRIBE	Access [1] / SRIBE
$3030\ 3000_H$ - $3FFF\ FFFF_H$	-	Reserved	SRIBE	SRIBE

1) PCACHE/PTAG and DCACHE/DTAG can be only accessed when mapped into the address space

2.2.5 Segment 4

Table 11 Address map of segment 4

Address range	Size	Description	Access type	
			Read	Write
$4000\ 0000_H$ - $4003\ BFFF_H$	240 Kbyte	CPU3.DSPR	Access	Access
$4003\ C000_H$ - $4003\ FFFF_H$	16 Kbyte	CPU3.DSPR (extension) or CPU3.DCACHE (RAM)	Access [1] / SRIBE	Access [1] / SRIBE
$4004\ 0000_H$ - $400F\ FFFF_H$	-	Reserved	SRIBE	SRIBE
$4010\ 0000_H$ - $4010\ FFFF_H$	64 Kbyte	CPU3.PSPR	Access	Access
$4011\ 0000_H$ - $4011\ 7FFF_H$	32 Kbyte	CPU3.PSPR (extension) or CPU3.PCACHE (RAM)	Access [1] / SRIBE	Access [1] / SRIBE
$4011\ 8000_H$ - $4011\ FFFF_H$	-	Reserved	SRIBE	SRIBE
$4012\ 0000_H$ - $4012\ FFFF_H$	64 Kbyte	Alias for CPU3.PSPR	Access	Access
$4013\ 0000_H$ - $4013\ 7FFF_H$	32 Kbyte	Alias for CPU3.PSPR (extension)	Access	Access
$4013\ 8000_H$ - $4013\ FFFF_H$	-	Reserved	SRIBE	SRIBE
$4014\ 0000_H$ - $4014\ FFFF_H$	64 Kbyte	Alias for CPU3.PSPR	Access	Access
$4015\ 0000_H$ - $4015\ 7FFF_H$	32 Kbyte	Alias for CPU3.PSPR (extension)	Access	Access
$4015\ 8000_H$ - $4015\ FFFF_H$	-	Reserved	SRIBE	SRIBE
$4016\ 0000_H$ - $4016\ FFFF_H$	64 Kbyte	Alias for CPU3.PSPR	Access	Access
$4017\ 0000_H$ - $4017\ 7FFF_H$	32 Kbyte	Alias for CPU3.PSPR (extension)	Access	Access
$4017\ 8000_H$ - $4017\ FFFF_H$	-	Reserved	SRIBE	SRIBE
$4018\ 0000_H$ - $4018\ FFFF_H$	64 Kbyte	Alias for CPU3.PSPR	Access	Access
$4019\ 0000_H$ - $4019\ 7FFF_H$	32 Kbyte	Alias for CPU3.PSPR (extension)	Access	Access
$4019\ 8000_H$ - $4019\ FFFF_H$	-	Reserved	SRIBE	SRIBE
$401A\ 0000_H$ - $401A\ FFFF_H$	64 Kbyte	Alias for CPU3.PSPR	Access	Access
$401B\ 0000_H$ - $401B\ 7FFF_H$	32 Kbyte	Alias for CPU3.PSPR (extension)	Access	Access
$401B\ 8000_H$ - $401B\ FFFF_H$	-	Reserved	SRIBE	SRIBE
$401C\ 0000_H$ - $401C\ FFFF_H$	64 Kbyte	Alias for CPU3.PSPR	Access	Access
$401D\ 0000_H$ - $401D\ 7FFF_H$	32 Kbyte	Alias for CPU3.PSPR (extension)	Access	Access
$401D\ 8000_H$ - $401D\ FFFF_H$	-	Reserved	SRIBE	SRIBE
$401E\ 0000_H$ - $401E\ FFFF_H$	64 Kbyte	Alias for CPU3.PSPR	Access	Access
$401F\ 0000_H$ - $401F\ 7FFF_H$	32 Kbyte	Alias for CPU3.PSPR (extension)	Access	Access
$401F\ 8000_H$ - $401F\ FFFF_H$	-	Reserved	SRIBE	SRIBE
$4020\ 0000_H$ - $4020\ 17FF_H$	-	CPU3.DTAG	Access [1] / SRIBE	Access [1] / SRIBE
$4020\ 1800_H$ - $402F\ FFFF_H$	-	Reserved	SRIBE	SRIBE

(table continues...)

Table 11 (continued) Address map of segment 4

Address range	Size	Description	Access type	
			Read	Write
$4030\ 0000_H$ - $4030\ 2FFF_H$	-	CPU3.PTAG	Access [1] / SRIBE	Access [1] / SRIBE
$4030\ 3000_H$ - $4FFF\ FFFF_H$	-	Reserved	SRIBE	SRIBE

1) PCACHE/PTAG and DCACHE/DTAG can be only accessed when mapped into the address space

2.2.6 Segment 5

Table 12 Address map of segment 5

Address range	Size	Description	Access type	
			Read	Write
$5000\ 0000_H$ - $5003\ BFFF_H$	240 Kbyte	CPU2.DSPR	Access	Access
$5003\ C000_H$ - $5003\ FFFF_H$	16 Kbyte	CPU2.DSPR (extension) or CPU2.DCACHE (RAM)	Access [1] / SRIBE	Access [1] / SRIBE
$5004\ 0000_H$ - $500F\ FFFF_H$	-	Reserved	SRIBE	SRIBE
$5010\ 0000_H$ - $5010\ FFFF_H$	64 Kbyte	CPU2.PSPR	Access	Access
$5011\ 0000_H$ - $5011\ 7FFF_H$	32 Kbyte	CPU2.PSPR (extension) or CPU2.PCACHE (RAM)	Access [1] / SRIBE	Access [1] / SRIBE
$5011\ 8000_H$ - $5011\ FFFF_H$	-	Reserved	SRIBE	SRIBE
$5012\ 0000_H$ - $5012\ FFFF_H$	64 Kbyte	Alias for CPU2.PSPR	Access	Access
$5013\ 0000_H$ - $5013\ 7FFF_H$	32 Kbyte	Alias for CPU2.PSPR (extension)	Access	Access
$5013\ 8000_H$ - $5013\ FFFF_H$	-	Reserved	SRIBE	SRIBE
$5014\ 0000_H$ - $5014\ FFFF_H$	64 Kbyte	Alias for CPU2.PSPR	Access	Access
$5015\ 0000_H$ - $5015\ 7FFF_H$	32 Kbyte	Alias for CPU2.PSPR (extension)	Access	Access
$5015\ 8000_H$ - $5015\ FFFF_H$	-	Reserved	SRIBE	SRIBE
$5016\ 0000_H$ - $5016\ FFFF_H$	64 Kbyte	Alias for CPU2.PSPR	Access	Access
$5017\ 0000_H$ - $5017\ 7FFF_H$	32 Kbyte	Alias for CPU2.PSPR (extension)	Access	Access
$5017\ 8000_H$ - $5017\ FFFF_H$	-	Reserved	SRIBE	SRIBE
$5018\ 0000_H$ - $5018\ FFFF_H$	64 Kbyte	Alias for CPU2.PSPR	Access	Access
$5019\ 0000_H$ - $5019\ 7FFF_H$	32 Kbyte	Alias for CPU2.PSPR (extension)	Access	Access
$5019\ 8000_H$ - $5019\ FFFF_H$	-	Reserved	SRIBE	SRIBE
$501A\ 0000_H$ - $501A\ FFFF_H$	64 Kbyte	Alias for CPU2.PSPR	Access	Access
$501B\ 0000_H$ - $501B\ 7FFF_H$	32 Kbyte	Alias for CPU2.PSPR (extension)	Access	Access
$501B\ 8000_H$ - $501B\ FFFF_H$	-	Reserved	SRIBE	SRIBE
$501C\ 0000_H$ - $501C\ FFFF_H$	64 Kbyte	Alias for CPU2.PSPR	Access	Access
$501D\ 0000_H$ - $501D\ 7FFF_H$	32 Kbyte	Alias for CPU2.PSPR (extension)	Access	Access
$501D\ 8000_H$ - $501D\ FFFF_H$	-	Reserved	SRIBE	SRIBE
$501E\ 0000_H$ - $501E\ FFFF_H$	64 Kbyte	Alias for CPU2.PSPR	Access	Access
$501F\ 0000_H$ - $501F\ 7FFF_H$	32 Kbyte	Alias for CPU2.PSPR (extension)	Access	Access
$501F\ 8000_H$ - $501F\ FFFF_H$	-	Reserved	SRIBE	SRIBE
$5020\ 0000_H$ - $5020\ 17FF_H$	-	CPU2.DTAG	Access [1] / SRIBE	Access [1] / SRIBE
$5020\ 1800_H$ - $502F\ FFFF_H$	-	Reserved	SRIBE	SRIBE

(table continues...)

Table 12 (continued) Address map of segment 5

Address range	Size	Description	Access type	
			Read	Write
$5030\ 0000_H$ - $5030\ 2FFF_H$	-	CPU2.PTAG	Access [1] / SRIBE	Access [1] / SRIBE
$5030\ 3000_H$ - $5FFF\ FFFF_H$	-	Reserved	SRIBE	SRIBE

1) PCACHE/PTAG and DCACHE/DTAG can be only accessed when mapped into the address space

2.2.7 Segment 6

Table 13 Address map of segment 6

Address range	Size	Description	Access type	
			Read	Write
$6000\ 0000_H$ - $6003\ BFFF_H$	240 Kbyte	CPU1.DSPR	Access	Access
$6003\ C000_H$ - $6003\ FFFF_H$	16 Kbyte	CPU1.DSPR (extension) or CPU1.DCACHE (RAM)	Access [1] / SRIBE	Access [1] / SRIBE
$6004\ 0000_H$ - $600F\ FFFF_H$	-	Reserved	SRIBE	SRIBE
$6010\ 0000_H$ - $6010\ FFFF_H$	64 Kbyte	CPU1.PSPR	Access	Access
$6011\ 0000_H$ - $6011\ 7FFF_H$	32 Kbyte	CPU1.PSPR (extension) or CPU1.PCACHE (RAM)	Access [1] / SRIBE	Access [1] / SRIBE
$6011\ 8000_H$ - $6011\ FFFF_H$	-	Reserved	SRIBE	SRIBE
$6012\ 0000_H$ - $6012\ FFFF_H$	64 Kbyte	Alias for CPU1.PSPR	Access	Access
$6013\ 0000_H$ - $6013\ 7FFF_H$	32 Kbyte	Alias for CPU1.PSPR (extension)	Access	Access
$6013\ 8000_H$ - $6013\ FFFF_H$	-	Reserved	SRIBE	SRIBE
$6014\ 0000_H$ - $6014\ FFFF_H$	64 Kbyte	Alias for CPU1.PSPR	Access	Access
$6015\ 0000_H$ - $6015\ 7FFF_H$	32 Kbyte	Alias for CPU1.PSPR (extension)	Access	Access
$6015\ 8000_H$ - $6015\ FFFF_H$	-	Reserved	SRIBE	SRIBE
$6016\ 0000_H$ - $6016\ FFFF_H$	64 Kbyte	Alias for CPU1.PSPR	Access	Access
$6017\ 0000_H$ - $6017\ 7FFF_H$	32 Kbyte	Alias for CPU1.PSPR (extension)	Access	Access
$6017\ 8000_H$ - $6017\ FFFF_H$	-	Reserved	SRIBE	SRIBE
$6018\ 0000_H$ - $6018\ FFFF_H$	64 Kbyte	Alias for CPU1.PSPR	Access	Access
$6019\ 0000_H$ - $6019\ 7FFF_H$	32 Kbyte	Alias for CPU1.PSPR (extension)	Access	Access
$6019\ 8000_H$ - $6019\ FFFF_H$	-	Reserved	SRIBE	SRIBE
$601A\ 0000_H$ - $601A\ FFFF_H$	64 Kbyte	Alias for CPU1.PSPR	Access	Access
$601B\ 0000_H$ - $601B\ 7FFF_H$	32 Kbyte	Alias for CPU1.PSPR (extension)	Access	Access
$601B\ 8000_H$ - $601B\ FFFF_H$	-	Reserved	SRIBE	SRIBE
$601C\ 0000_H$ - $601C\ FFFF_H$	64 Kbyte	Alias for CPU1.PSPR	Access	Access
$601D\ 0000_H$ - $601D\ 7FFF_H$	32 Kbyte	Alias for CPU1.PSPR (extension)	Access	Access
$601D\ 8000_H$ - $601D\ FFFF_H$	-	Reserved	SRIBE	SRIBE
$601E\ 0000_H$ - $601E\ FFFF_H$	64 Kbyte	Alias for CPU1.PSPR	Access	Access
$601F\ 0000_H$ - $601F\ 7FFF_H$	32 Kbyte	Alias for CPU1.PSPR (extension)	Access	Access
$601F\ 8000_H$ - $601F\ FFFF_H$	-	Reserved	SRIBE	SRIBE
$6020\ 0000_H$ - $6020\ 17FF_H$	-	CPU1.DTAG	Access [1] / SRIBE	Access [1] / SRIBE
$6020\ 1800_H$ - $602F\ FFFF_H$	-	Reserved	SRIBE	SRIBE

(table continues...)

Table 13 (continued) Address map of segment 6

Address range	Size	Description	Access type	
			Read	Write
$6030\ 0000_H$ - $6030\ 2FFF_H$	-	CPU1.PTAG	Access [1] / SRIBE	Access [1] / SRIBE
$6030\ 3000_H$ - $6FFF\ FFFF_H$	-	Reserved	SRIBE	SRIBE

1) PCACHE/PTAG and DCACHE/DTAG can be only accessed when mapped into the address space

2.2.8 Segment 7

Table 14 Address map of segment 7

Address range	Size	Description	Access type	
			Read	Write
$7000\ 0000_H$ - $7003\ BFFF_H$	240 Kbyte	CPU0.DSPR	Access	Access
$7003\ C000_H$ - $7003\ FFFF_H$	16 Kbyte	CPU0.DSPR (extension) or CPU0.DCACHE (RAM)	Access [1] / SRIBE	Access [1] / SRIBE
$7004\ 0000_H$ - $700F\ FFFF_H$	-	Reserved	SRIBE	SRIBE
$7010\ 0000_H$ - $7010\ FFFF_H$	64 Kbyte	CPU0.PSPR	Access	Access
$7011\ 0000_H$ - $7011\ 7FFF_H$	32 Kbyte	CPU0.PSPR (extension) or CPU0.PCACHE (RAM)	Access [1] / SRIBE	Access [1] / SRIBE
$7011\ 8000_H$ - $7011\ FFFF_H$	-	Reserved	SRIBE	SRIBE
$7012\ 0000_H$ - $7012\ FFFF_H$	64 Kbyte	Alias for CPU0.PSPR	Access	Access
$7013\ 0000_H$ - $7013\ 7FFF_H$	32 Kbyte	Alias for CPU0.PSPR (extension)	Access	Access
$7013\ 8000_H$ - $7013\ FFFF_H$	-	Reserved	SRIBE	SRIBE
$7014\ 0000_H$ - $7014\ FFFF_H$	64 Kbyte	Alias for CPU0.PSPR	Access	Access
$7015\ 0000_H$ - $7015\ 7FFF_H$	32 Kbyte	Alias for CPU0.PSPR (extension)	Access	Access
$7015\ 8000_H$ - $7015\ FFFF_H$	-	Reserved	SRIBE	SRIBE
$7016\ 0000_H$ - $7016\ FFFF_H$	64 Kbyte	Alias for CPU0.PSPR	Access	Access
$7017\ 0000_H$ - $7017\ 7FFF_H$	32 Kbyte	Alias for CPU0.PSPR (extension)	Access	Access
$7017\ 8000_H$ - $7017\ FFFF_H$	-	Reserved	SRIBE	SRIBE
$7018\ 0000_H$ - $7018\ FFFF_H$	64 Kbyte	Alias for CPU0.PSPR	Access	Access
$7019\ 0000_H$ - $7019\ 7FFF_H$	32 Kbyte	Alias for CPU0.PSPR (extension)	Access	Access
$7019\ 8000_H$ - $7019\ FFFF_H$	-	Reserved	SRIBE	SRIBE
$701A\ 0000_H$ - $701A\ FFFF_H$	64 Kbyte	Alias for CPU0.PSPR	Access	Access
$701B\ 0000_H$ - $701B\ 7FFF_H$	32 Kbyte	Alias for CPU0.PSPR (extension)	Access	Access
$701B\ 8000_H$ - $701B\ FFFF_H$	-	Reserved	SRIBE	SRIBE
$701C\ 0000_H$ - $701C\ FFFF_H$	64 Kbyte	Alias for CPU0.PSPR	Access	Access
$701D\ 0000_H$ - $701D\ 7FFF_H$	32 Kbyte	Alias for CPU0.PSPR (extension)	Access	Access
$701D\ 8000_H$ - $701D\ FFFF_H$	-	Reserved	SRIBE	SRIBE
$701E\ 0000_H$ - $701E\ FFFF_H$	64 Kbyte	Alias for CPU0.PSPR	Access	Access
$701F\ 0000_H$ - $701F\ 7FFF_H$	32 Kbyte	Alias for CPU0.PSPR (extension)	Access	Access
$701F\ 8000_H$ - $701F\ FFFF_H$	-	Reserved	SRIBE	SRIBE
$7020\ 0000_H$ - $7020\ 17FF_H$	-	CPU0.DTAG	Access [1] / SRIBE	Access [1] / SRIBE
$7020\ 1800_H$ - $702F\ FFFF_H$	-	Reserved	SRIBE	SRIBE

(table continues...)

Table 14 (continued) Address map of segment 7

Address range	Size	Description	Access type	
			Read	Write
$7030\ 0000_H$ - $7030\ 2FFF_H$	-	CPU0.PTAG	Access [1] / SRIBE	Access [1] / SRIBE
$7030\ 3000_H$ - $7FFF\ FFFF_H$	-	Reserved	SRIBE	SRIBE

1) PCACHE/PTAG and DCACHE/DTAG can be only accessed when mapped into the address space

2.2.9 Segments 8 and 10

The read address of the program flash and erase counters varies for the following modes:

- Linear address map
- SOTA address map A
- SOTA address map B

2.2.9.1 Segment 8 linear address map

Table 15 Linear address map of segment 8

Address range	Size	Description	Access type	
			Read	Write
$8000\ 0000_H$ - $801F\ FFFF_H$	2 Mbyte	NVM.PFLASH00 (Program Flash)	Access	SRIBE
$8020\ 0000_H$ - $803F\ FFFF_H$	2 Mbyte	NVM.PFLASH01 (Program Flash)	Access	SRIBE
$8040\ 0000_H$ - $805F\ FFFF_H$	2 Mbyte	NVM.PFLASH10 (Program Flash)	Access	SRIBE
$8060\ 0000_H$ - $807F\ FFFF_H$	2 Mbyte	NVM.PFLASH11 (Program Flash)	Access	SRIBE
$8080\ 0000_H$ - $808F\ FFFF_H$	1 Mbyte	NVM.PFLASH20 (Program Flash)	Access	SRIBE
$8090\ 0000_H$ - $809F\ FFFF_H$	1 Mbyte	NVM.PFLASH21 (Program Flash)	Access	SRIBE
$80A0\ 0000_H$ - $80BF\ FFFF_H$	2 Mbyte	NVM.PFLASH30 (Program Flash)	Access	SRIBE
$80C0\ 0000_H$ - $80DF\ FFFF_H$	2 Mbyte	NVM.PFLASH31 (Program Flash)	Access	SRIBE
$80E0\ 0000_H$ - $80FF\ FFFF_H$	2 Mbyte	NVM.PFLASH40 (Program Flash)	Access	SRIBE
$8100\ 0000_H$ - $811F\ FFFF_H$	2 Mbyte	NVM.PFLASH41 (Program Flash)	Access	SRIBE
$8120\ 0000_H$ - $812F\ FFFF_H$	1 Mbyte	NVM.PFLASH50 (Program Flash)	Access	SRIBE
$8130\ 0000_H$ - $813F\ FFFF_H$	1 Mbyte	NVM.PFLASH51 (Program Flash)	Access	SRIBE
$8140\ 0000_H$ - $83FF\ FFFF_H$	-	Reserved	SRIBE	SRIBE
$8400\ 0000_H$ - $840F\ FFFF_H$	1 Mbyte	NVM.PFLASHcs (Program Flash)	Access	SRIBE
$8410\ 0000_H$ - $8FDF\ FFFF_H$	-	Reserved	SRIBE	SRIBE
$8FE0\ 0000_H$ - $8FE7\ FFFF_H$	512 Kbyte	OLDA	SRIBE	Access / SRIBE
$8FE8\ 0000_H$ - $8FFD\ FFFF_H$	-	Reserved	SRIBE	SRIBE
$8FFE\ 0000_H$ - $8FFE\ FFFF_H$	64 Kbyte	CSROM	Access	SRIBE
$8FFF\ 0000_H$ - $8FFF\ FFFF_H$	64 Kbyte	NVM.BROM	Access	SRIBE

2.2.9.2 Segment 10 linear address map

Table 16 Linear address map of segment 10

Address range	Size	Description	Access type	
			Read	Write
A000 0000$_H$ - A01F FFFF$_H$	2 Mbyte	NVM.PFLASH00 (Program Flash)	Access	SRIBE
A020 0000$_H$ - A03F FFFF$_H$	2 Mbyte	NVM.PFLASH01 (Program Flash)	Access	SRIBE
A040 0000$_H$ - A05F FFFF$_H$	2 Mbyte	NVM.PFLASH10 (Program Flash)	Access	SRIBE
A060 0000$_H$ - A07F FFFF$_H$	2 Mbyte	NVM.PFLASH11 (Program Flash)	Access	SRIBE
A080 0000$_H$ - A08F FFFF$_H$	1 Mbyte	NVM.PFLASH20 (Program Flash)	Access	SRIBE
A090 0000$_H$ - A09F FFFF$_H$	1 Mbyte	NVM.PFLASH21 (Program Flash)	Access	SRIBE
A0A0 0000$_H$ - A0BF FFFF$_H$	2 Mbyte	NVM.PFLASH30 (Program Flash)	Access	SRIBE
A0C0 0000$_H$ - A0DF FFFF$_H$	2 Mbyte	NVM.PFLASH31 (Program Flash)	Access	SRIBE
A0E0 0000$_H$ - A0FF FFFF$_H$	2 Mbyte	NVM.PFLASH40 (Program Flash)	Access	SRIBE
A100 0000$_H$ - A11F FFFF$_H$	2 Mbyte	NVM.PFLASH41 (Program Flash)	Access	SRIBE
A120 0000$_H$ - A12F FFFF$_H$	1 Mbyte	NVM.PFLASH50 (Program Flash)	Access	SRIBE
A130 0000$_H$ - A13F FFFF$_H$	1 Mbyte	NVM.PFLASH51 (Program Flash)	Access	SRIBE
A140 0000$_H$ - A3FF FFFF$_H$	-	Reserved	SRIBE	SRIBE
A400 0000$_H$ - A40F FFFF$_H$	1 Mbyte	NVM.PFLASHcs (Program Flash)	Access	SRIBE
A410 0000$_H$ - A7FF FFFF$_H$	-	Reserved	SRIBE	SRIBE
A800 0000$_H$ - A800 3FFF$_H$	16 Kbyte	NVM.PFLASH00 (Erase Counter)	Access	SRIBE
A800 4000$_H$ - A81F FFFF$_H$	-	Reserved	SRIBE	SRIBE
A820 0000$_H$ - A820 3FFF$_H$	16 Kbyte	NVM.PFLASH01 (Erase Counter)	Access	SRIBE
A820 4000$_H$ - A83F FFFF$_H$	-	Reserved	SRIBE	SRIBE
A840 0000$_H$ - A840 3FFF$_H$	16 Kbyte	NVM.PFLASH10 (Erase Counter)	Access	SRIBE
A840 4000$_H$ - A85F FFFF$_H$	-	Reserved	SRIBE	SRIBE
A860 0000$_H$ - A860 3FFF$_H$	16 Kbyte	NVM.PFLASH11 (Erase Counter)	Access	SRIBE
A860 4000$_H$ - A87F FFFF$_H$	-	Reserved	SRIBE	SRIBE
A880 0000$_H$ - A880 3FFF$_H$	16 Kbyte	NVM.PFLASH20 (Erase Counter)	Access	SRIBE
A880 4000$_H$ - A88F FFFF$_H$	-	Reserved	SRIBE	SRIBE
A890 0000$_H$ - A890 3FFF$_H$	16 Kbyte	NVM.PFLASH21 (Erase Counter)	Access	SRIBE
A890 4000$_H$ - A89F FFFF$_H$	-	Reserved	SRIBE	SRIBE
A8A0 0000$_H$ - A8A0 3FFF$_H$	16 Kbyte	NVM.PFLASH30 (Erase Counter)	Access	SRIBE
A8A0 4000$_H$ - A8BF FFFF$_H$	-	Reserved	SRIBE	SRIBE
A8C0 0000$_H$ - A8C0 3FFF$_H$	16 Kbyte	NVM.PFLASH31 (Erase Counter)	Access	SRIBE
A8C0 4000$_H$ - A8DF FFFF$_H$	-	Reserved	SRIBE	SRIBE

(table continues...)

Table 16 (continued) Linear address map of segment 10

Address range	Size	Description	Access type	
			Read	Write
A8E0 0000$_H$ - A8E0 3FFF$_H$	16 Kbyte	NVM.PFLASH40 (Erase Counter)	Access	SRIBE
A8E0 4000$_H$ - A8FF FFFF$_H$	-	Reserved	SRIBE	SRIBE
A900 0000$_H$ - A900 3FFF$_H$	16 Kbyte	NVM.PFLASH41 (Erase Counter)	Access	SRIBE
A900 4000$_H$ - A91F FFFF$_H$	-	Reserved	SRIBE	SRIBE
A920 0000$_H$ - A920 3FFF$_H$	16 Kbyte	NVM.PFLASH50 (Erase Counter)	Access	SRIBE
A920 4000$_H$ - A92F FFFF$_H$	-	Reserved	SRIBE	SRIBE
A930 0000$_H$ - A930 3FFF$_H$	16 Kbyte	NVM.PFLASH51 (Erase Counter)	Access	SRIBE
A930 4000$_H$ - ABFF FFFF$_H$	-	Reserved	SRIBE	SRIBE
AC00 0000$_H$ - AC00 3FFF$_H$	16 Kbyte	NVM.PFLASHcs (Erase Counter)	Access	SRIBE
AC00 4000$_H$ - ADFF FFFF$_H$	-	Reserved	SRIBE	SRIBE
AE00 0000$_H$ - AE0F FFFF$_H$	1024 Kbyte	NVM.EEPROM0	Access	SRIBE
AE10 0000$_H$ - AE3F FFFF$_H$	-	Reserved	SRIBE	SRIBE
AE40 0000$_H$ - AE41 3FFF$_H$	80 Kbyte	NVM.UCB0	Access	SRIBE
AE41 4000$_H$ - AE7F FFFF$_H$	-	Reserved	SRIBE	SRIBE
AE80 0000$_H$ - AE81 FFFF$_H$	128 Kbyte	NVM.EEPROM1	Access	SRIBE
AE82 0000$_H$ - AEBF FFFF$_H$	-	Reserved	SRIBE	SRIBE
AEC0 0000$_H$ - AEC0 CFFF$_H$	52 Kbyte	NVM.UCB1	Access	SRIBE
AEC0 D000$_H$ - AFDF FFFF$_H$	-	Reserved	SRIBE	SRIBE
AFE0 0000$_H$ - AFE7 FFFF$_H$	512 Kbyte	OLDA	SRIBE	Access / SRIBE
AFE8 0000$_H$ - AFFD FFFF$_H$	-	Reserved	SRIBE	SRIBE
AFFE 0000$_H$ - AFFE FFFF$_H$	64 Kbyte	CSROM	Access	SRIBE
AFFF 0000$_H$ - AFFF FFFF$_H$	64 Kbyte	NVM.BROM	Access	SRIBE

2.2.9.3 Segment 8 SOTA address map A

Table 17 SOTA address map A of segment 8

Address range	Size	Description	Access type	
			Read	Write
$8000\ 0000_H$ - $801F\ FFFF_H$	2 Mbyte	NVM.PFLASH00 (Program Flash)	Access	SRIBE
$8020\ 0000_H$ - $803F\ FFFF_H$	2 Mbyte	NVM.PFLASH10 (Program Flash)	Access	SRIBE
$8040\ 0000_H$ - $804F\ FFFF_H$	1 Mbyte	NVM.PFLASH20 (Program Flash)	Access	SRIBE
$8050\ 0000_H$ - $806F\ FFFF_H$	2 Mbyte	NVM.PFLASH30 (Program Flash)	Access	SRIBE
$8070\ 0000_H$ - $808F\ FFFF_H$	2 Mbyte	NVM.PFLASH40 (Program Flash)	Access	SRIBE
$8090\ 0000_H$ - $809F\ FFFF_H$	1 Mbyte	NVM.PFLASH50 (Program Flash)	Access	SRIBE
$80A0\ 0000_H$ - $81FF\ FFFF_H$	-	Reserved	SRIBE	SRIBE
$8200\ 0000_H$ - $821F\ FFFF_H$	2 Mbyte	NVM.PFLASH01 (Program Flash)	Access	SRIBE
$8220\ 0000_H$ - $823F\ FFFF_H$	2 Mbyte	NVM.PFLASH11 (Program Flash)	Access	SRIBE
$8240\ 0000_H$ - $824F\ FFFF_H$	1 Mbyte	NVM.PFLASH21 (Program Flash)	Access	SRIBE
$8250\ 0000_H$ - $826F\ FFFF_H$	2 Mbyte	NVM.PFLASH31 (Program Flash)	Access	SRIBE
$8270\ 0000_H$ - $828F\ FFFF_H$	2 Mbyte	NVM.PFLASH41 (Program Flash)	Access	SRIBE
$8290\ 0000_H$ - $829F\ FFFF_H$	1 Mbyte	NVM.PFLASH51 (Program Flash)	Access	SRIBE
$82A0\ 0000_H$ - $83FF\ FFFF_H$	-	Reserved	SRIBE	SRIBE
$8400\ 0000_H$ - $840F\ FFFF_H$	1 Mbyte	NVM.PFLASHcs (Program Flash)	Access	SRIBE
$8410\ 0000_H$ - $8FDF\ FFFF_H$	-	Reserved	SRIBE	SRIBE
$8FE0\ 0000_H$ - $8FE7\ FFFF_H$	512 Kbyte	OLDA	SRIBE	Access / SRIBE
$8FE8\ 0000_H$ - $8FFD\ FFFF_H$	-	Reserved	SRIBE	SRIBE
$8FFE\ 0000_H$ - $8FFE\ FFFF_H$	64 Kbyte	CSROM	Access	SRIBE
$8FFF\ 0000_H$ - $8FFF\ FFFF_H$	64 Kbyte	NVM.BROM	Access	SRIBE

2.2.9.4 Segment 10 SOTA address map A

Table 18 SOTA address map A of segment 10

Address range	Size	Description	Access type	
			Read	Write
A000 0000$_H$ - A01F FFFF$_H$	2 Mbyte	NVM.PFLASH00 (Program Flash)	Access	SRIBE
A020 0000$_H$ - A03F FFFF$_H$	2 Mbyte	NVM.PFLASH10 (Program Flash)	Access	SRIBE
A040 0000$_H$ - A04F FFFF$_H$	1 Mbyte	NVM.PFLASH20 (Program Flash)	Access	SRIBE
A050 0000$_H$ - A06F FFFF$_H$	2 Mbyte	NVM.PFLASH30 (Program Flash)	Access	SRIBE
A070 0000$_H$ - A08F FFFF$_H$	2 Mbyte	NVM.PFLASH40 (Program Flash)	Access	SRIBE
A090 0000$_H$ - A09F FFFF$_H$	1 Mbyte	NVM.PFLASH50 (Program Flash)	Access	SRIBE
A0A0 0000$_H$ - A1FF FFFF$_H$	-	Reserved	SRIBE	SRIBE
A200 0000$_H$ - A21F FFFF$_H$	2 Mbyte	NVM.PFLASH01 (Program Flash)	Access	SRIBE
A220 0000$_H$ - A23F FFFF$_H$	2 Mbyte	NVM.PFLASH11 (Program Flash)	Access	SRIBE
A240 0000$_H$ - A24F FFFF$_H$	1 Mbyte	NVM.PFLASH21 (Program Flash)	Access	SRIBE
A250 0000$_H$ - A26F FFFF$_H$	2 Mbyte	NVM.PFLASH31 (Program Flash)	Access	SRIBE
A270 0000$_H$ - A28F FFFF$_H$	2 Mbyte	NVM.PFLASH41 (Program Flash)	Access	SRIBE
A290 0000$_H$ - A29F FFFF$_H$	1 Mbyte	NVM.PFLASH51 (Program Flash)	Access	SRIBE
A2A0 0000$_H$ - A3FF FFFF$_H$	-	Reserved	SRIBE	SRIBE
A400 0000$_H$ - A40F FFFF$_H$	1 Mbyte	NVM.PFLASHcs (Program Flash)	Access	SRIBE
A410 0000$_H$ - A7FF FFFF$_H$	-	Reserved	SRIBE	SRIBE
A800 0000$_H$ - A800 3FFF$_H$	16 Kbyte	NVM.PFLASH00 (Erase Counter)	Access	SRIBE
A800 4000$_H$ - A81F FFFF$_H$	-	Reserved	SRIBE	SRIBE
A820 0000$_H$ - A820 3FFF$_H$	16 Kbyte	NVM.PFLASH10 (Erase Counter)	Access	SRIBE
A820 4000$_H$ - A83F FFFF$_H$	-	Reserved	SRIBE	SRIBE
A840 0000$_H$ - A840 3FFF$_H$	16 Kbyte	NVM.PFLASH20 (Erase Counter)	Access	SRIBE
A840 4000$_H$ - A84F FFFF$_H$	-	Reserved	SRIBE	SRIBE
A850 0000$_H$ - A850 3FFF$_H$	16 Kbyte	NVM.PFLASH30 (Erase Counter)	Access	SRIBE
A850 4000$_H$ - A86F FFFF$_H$	-	Reserved	SRIBE	SRIBE
A870 0000$_H$ - A870 3FFF$_H$	16 Kbyte	NVM.PFLASH40 (Erase Counter)	Access	SRIBE
A870 4000$_H$ - A88F FFFF$_H$	-	Reserved	SRIBE	SRIBE
A890 0000$_H$ - A890 3FFF$_H$	16 Kbyte	NVM.PFLASH50 (Erase Counter)	Access	SRIBE
A890 4000$_H$ - A9FF FFFF$_H$	-	Reserved	SRIBE	SRIBE
AA00 0000$_H$ - AA00 3FFF$_H$	16 Kbyte	NVM.PFLASH01 (Erase Counter)	Access	SRIBE
AA00 4000$_H$ - AA1F FFFF$_H$	-	Reserved	SRIBE	SRIBE
AA20 0000$_H$ - AA20 3FFF$_H$	16 Kbyte	NVM.PFLASH11 (Erase Counter)	Access	SRIBE

(table continues...)

Table 18 (continued) SOTA address map A of segment 10

Address range	Size	Description	Access type	
			Read	Write
AA20 4000$_H$ - AA3F FFFF$_H$	-	Reserved	SRIBE	SRIBE
AA40 0000$_H$ - AA40 3FFF$_H$	16 Kbyte	NVM.PFLASH21 (Erase Counter)	Access	SRIBE
AA40 4000$_H$ - AA4F FFFF$_H$	-	Reserved	SRIBE	SRIBE
AA50 0000$_H$ - AA50 3FFF$_H$	16 Kbyte	NVM.PFLASH31 (Erase Counter)	Access	SRIBE
AA50 4000$_H$ - AA6F FFFF$_H$	-	Reserved	SRIBE	SRIBE
AA70 0000$_H$ - AA70 3FFF$_H$	16 Kbyte	NVM.PFLASH41 (Erase Counter)	Access	SRIBE
AA70 4000$_H$ - AA8F FFFF$_H$	-	Reserved	SRIBE	SRIBE
AA90 0000$_H$ - AA90 3FFF$_H$	16 Kbyte	NVM.PFLASH51 (Erase Counter)	Access	SRIBE
AA90 4000$_H$ - ABFF FFFF$_H$	-	Reserved	SRIBE	SRIBE
AC00 0000$_H$ - AC00 3FFF$_H$	16 Kbyte	NVM.PFLASHcs (Erase Counter)	Access	SRIBE
AC00 4000$_H$ - ADFF FFFF$_H$	-	Reserved	SRIBE	SRIBE
AE00 0000$_H$ - AE0F FFFF$_H$	1024 Kbyte	NVM.EEPROM0	Access	SRIBE
AE10 0000$_H$ - AE3F FFFF$_H$	-	Reserved	SRIBE	SRIBE
AE40 0000$_H$ - AE41 3FFF$_H$	80 Kbyte	NVM.UCB0	Access	SRIBE
AE41 4000$_H$ - AE7F FFFF$_H$	-	Reserved	SRIBE	SRIBE
AE80 0000$_H$ - AE81 FFFF$_H$	128 Kbyte	NVM.EEPROM1	Access	SRIBE
AE82 0000$_H$ - AEBF FFFF$_H$	-	Reserved	SRIBE	SRIBE
AEC0 0000$_H$ - AEC0 CFFF$_H$	52 Kbyte	NVM.UCB1	Access	SRIBE
AEC0 D000$_H$ - AFDF FFFF$_H$	-	Reserved	SRIBE	SRIBE
AFE0 0000$_H$ - AFE7 FFFF$_H$	512 Kbyte	OLDA	SRIBE	Access / SRIBE
AFE8 0000$_H$ - AFFD FFFF$_H$	-	Reserved	SRIBE	SRIBE
AFFE 0000$_H$ - AFFE FFFF$_H$	64 Kbyte	CSROM	Access	SRIBE
AFFF 0000$_H$ - AFFF FFFF$_H$	64 Kbyte	NVM.BROM	Access	SRIBE

2.2.9.5 Segment 8 SOTA address map B

Table 19 SOTA address map B of segment 8

Address range	Size	Description	Access type	
			Read	Write
$8000\ 0000_H$ - $801F\ FFFF_H$	2 Mbyte	NVM.PFLASH01 (Program Flash)	Access	SRIBE
$8020\ 0000_H$ - $803F\ FFFF_H$	2 Mbyte	NVM.PFLASH11 (Program Flash)	Access	SRIBE
$8040\ 0000_H$ - $804F\ FFFF_H$	1 Mbyte	NVM.PFLASH21 (Program Flash)	Access	SRIBE
$8050\ 0000_H$ - $806F\ FFFF_H$	2 Mbyte	NVM.PFLASH31 (Program Flash)	Access	SRIBE
$8070\ 0000_H$ - $808F\ FFFF_H$	2 Mbyte	NVM.PFLASH41 (Program Flash)	Access	SRIBE
$8090\ 0000_H$ - $809F\ FFFF_H$	1 Mbyte	NVM.PFLASH51 (Program Flash)	Access	SRIBE
$80A0\ 0000_H$ - $81FF\ FFFF_H$	-	Reserved	SRIBE	SRIBE
$8200\ 0000_H$ - $821F\ FFFF_H$	2 Mbyte	NVM.PFLASH00 (Program Flash)	Access	SRIBE
$8220\ 0000_H$ - $823F\ FFFF_H$	2 Mbyte	NVM.PFLASH10 (Program Flash)	Access	SRIBE
$8240\ 0000_H$ - $824F\ FFFF_H$	1 Mbyte	NVM.PFLASH20 (Program Flash)	Access	SRIBE
$8250\ 0000_H$ - $826F\ FFFF_H$	2 Mbyte	NVM.PFLASH30 (Program Flash)	Access	SRIBE
$8270\ 0000_H$ - $828F\ FFFF_H$	2 Mbyte	NVM.PFLASH40 (Program Flash)	Access	SRIBE
$8290\ 0000_H$ - $829F\ FFFF_H$	1 Mbyte	NVM.PFLASH50 (Program Flash)	Access	SRIBE
$82A0\ 0000_H$ - $83FF\ FFFF_H$	-	Reserved	SRIBE	SRIBE
$8400\ 0000_H$ - $840F\ FFFF_H$	1 Mbyte	NVM.PFLASHcs (Program Flash)	Access	SRIBE
$8410\ 0000_H$ - $8FDF\ FFFF_H$	-	Reserved	SRIBE	SRIBE
$8FE0\ 0000_H$ - $8FE7\ FFFF_H$	512 Kbyte	OLDA	SRIBE	Access / SRIBE
$8FE8\ 0000_H$ - $8FFD\ FFFF_H$	-	Reserved	SRIBE	SRIBE
$8FFE\ 0000_H$ - $8FFE\ FFFF_H$	64 Kbyte	CSROM	Access	SRIBE
$8FFF\ 0000_H$ - $8FFF\ FFFF_H$	64 Kbyte	NVM.BROM	Access	SRIBE

2.2.9.6 Segment 10 SOTA address map B

Table 20 SOTA address map B of segment 10

Address range	Size	Description	Access type	
			Read	Write
A000 0000$_H$ - A01F FFFF$_H$	2 Mbyte	NVM.PFLASH01 (Program Flash)	Access	SRIBE
A020 0000$_H$ - A03F FFFF$_H$	2 Mbyte	NVM.PFLASH11 (Program Flash)	Access	SRIBE
A040 0000$_H$ - A04F FFFF$_H$	1 Mbyte	NVM.PFLASH21 (Program Flash)	Access	SRIBE
A050 0000$_H$ - A06F FFFF$_H$	2 Mbyte	NVM.PFLASH31 (Program Flash)	Access	SRIBE
A070 0000$_H$ - A08F FFFF$_H$	2 Mbyte	NVM.PFLASH41 (Program Flash)	Access	SRIBE
A090 0000$_H$ - A09F FFFF$_H$	1 Mbyte	NVM.PFLASH51 (Program Flash)	Access	SRIBE
A0A0 0000$_H$ - A1FF FFFF$_H$	-	Reserved	SRIBE	SRIBE
A200 0000$_H$ - A21F FFFF$_H$	2 Mbyte	NVM.PFLASH00 (Program Flash)	Access	SRIBE
A220 0000$_H$ - A23F FFFF$_H$	2 Mbyte	NVM.PFLASH10 (Program Flash)	Access	SRIBE
A240 0000$_H$ - A24F FFFF$_H$	1 Mbyte	NVM.PFLASH20 (Program Flash)	Access	SRIBE
A250 0000$_H$ - A26F FFFF$_H$	2 Mbyte	NVM.PFLASH30 (Program Flash)	Access	SRIBE
A270 0000$_H$ - A28F FFFF$_H$	2 Mbyte	NVM.PFLASH40 (Program Flash)	Access	SRIBE
A290 0000$_H$ - A29F FFFF$_H$	1 Mbyte	NVM.PFLASH50 (Program Flash)	Access	SRIBE
A2A0 0000$_H$ - A3FF FFFF$_H$	-	Reserved	SRIBE	SRIBE
A400 0000$_H$ - A40F FFFF$_H$	1 Mbyte	NVM.PFLASHcs (Program Flash)	Access	SRIBE
A410 0000$_H$ - A7FF FFFF$_H$	-	Reserved	SRIBE	SRIBE
A800 0000$_H$ - A800 3FFF$_H$	16 Kbyte	NVM.PFLASH01 (Erase Counter)	Access	SRIBE
A800 4000$_H$ - A81F FFFF$_H$	-	Reserved	SRIBE	SRIBE
A820 0000$_H$ - A820 3FFF$_H$	16 Kbyte	NVM.PFLASH11 (Erase Counter)	Access	SRIBE
A820 4000$_H$ - A83F FFFF$_H$	-	Reserved	SRIBE	SRIBE
A840 0000$_H$ - A840 3FFF$_H$	16 Kbyte	NVM.PFLASH21 (Erase Counter)	Access	SRIBE
A840 4000$_H$ - A84F FFFF$_H$	-	Reserved	SRIBE	SRIBE
A850 0000$_H$ - A850 3FFF$_H$	16 Kbyte	NVM.PFLASH31 (Erase Counter)	Access	SRIBE
A850 4000$_H$ - A86F FFFF$_H$	-	Reserved	SRIBE	SRIBE
A870 0000$_H$ - A870 3FFF$_H$	16 Kbyte	NVM.PFLASH41 (Erase Counter)	Access	SRIBE
A870 4000$_H$ - A88F FFFF$_H$	-	Reserved	SRIBE	SRIBE
A890 0000$_H$ - A890 3FFF$_H$	16 Kbyte	NVM.PFLASH51 (Erase Counter)	Access	SRIBE
A890 4000$_H$ - A9FF FFFF$_H$	-	Reserved	SRIBE	SRIBE
AA00 0000$_H$ - AA00 3FFF$_H$	16 Kbyte	NVM.PFLASH00 (Erase Counter)	Access	SRIBE
AA00 4000$_H$ - AA1F FFFF$_H$	-	Reserved	SRIBE	SRIBE
AA20 0000$_H$ - AA20 3FFF$_H$	16 Kbyte	NVM.PFLASH10 (Erase Counter)	Access	SRIBE

(table continues...)

Table 20 (continued) SOTA address map B of segment 10

Address range	Size	Description	Access type	
			Read	Write
AA20 4000$_H$ - AA3F FFFF$_H$	-	Reserved	SRIBE	SRIBE
AA40 0000$_H$ - AA40 3FFF$_H$	16 Kbyte	NVM.PFLASH20 (Erase Counter)	Access	SRIBE
AA40 4000$_H$ - AA4F FFFF$_H$	-	Reserved	SRIBE	SRIBE
AA50 0000$_H$ - AA50 3FFF$_H$	16 Kbyte	NVM.PFLASH30 (Erase Counter)	Access	SRIBE
AA50 4000$_H$ - AA6F FFFF$_H$	-	Reserved	SRIBE	SRIBE
AA70 0000$_H$ - AA70 3FFF$_H$	16 Kbyte	NVM.PFLASH40 (Erase Counter)	Access	SRIBE
AA70 4000$_H$ - AA8F FFFF$_H$	-	Reserved	SRIBE	SRIBE
AA90 0000$_H$ - AA90 3FFF$_H$	16 Kbyte	NVM.PFLASH50 (Erase Counter)	Access	SRIBE
AA90 4000$_H$ - ABFF FFFF$_H$	-	Reserved	SRIBE	SRIBE
AC00 0000$_H$ - AC00 3FFF$_H$	16 Kbyte	NVM.PFLASHcs (Erase Counter)	Access	SRIBE
AC00 4000$_H$ - ADFF FFFF$_H$	-	Reserved	SRIBE	SRIBE
AE00 0000$_H$ - AE0F FFFF$_H$	1024 Kbyte	NVM.EEPROM0	Access	SRIBE
AE10 0000$_H$ - AE3F FFFF$_H$	-	Reserved	SRIBE	SRIBE
AE40 0000$_H$ - AE41 3FFF$_H$	80 Kbyte	NVM.UCB0	Access	SRIBE
AE41 4000$_H$ - AE7F FFFF$_H$	-	Reserved	SRIBE	SRIBE
AE80 0000$_H$ - AE81 FFFF$_H$	128 Kbyte	NVM.EEPROM1	Access	SRIBE
AE82 0000$_H$ - AEBF FFFF$_H$	-	Reserved	SRIBE	SRIBE
AEC0 0000$_H$ - AEC0 CFFF$_H$	52 Kbyte	NVM.UCB1	Access	SRIBE
AEC0 D000$_H$ - AFDF FFFF$_H$	-	Reserved	SRIBE	SRIBE
AFE0 0000$_H$ - AFE7 FFFF$_H$	512 Kbyte	OLDA	SRIBE	Access / SRIBE
AFE8 0000$_H$ - AFFD FFFF$_H$	-	Reserved	SRIBE	SRIBE
AFFE 0000$_H$ - AFFE FFFF$_H$	64 Kbyte	CSROM	Access	SRIBE
AFFF 0000$_H$ - AFFF FFFF$_H$	64 Kbyte	NVM.BROM	Access	SRIBE

2.2.10 Segments 9 and 11

2.2.10.1 Segment 9

Table 21 Address map of segment 9

Address range	Size	Description	Access type	
			Read	Write
$9000\ 0000_H$ - $9007\ FFFF_H$	512 Kbyte	CPU0.DLMU	Access	Access
$9008\ 0000_H$ - $900F\ FFFF_H$	512 Kbyte	CPU1.DLMU	Access	Access
$9010\ 0000_H$ - $9017\ FFFF_H$	512 Kbyte	CPU2.DLMU	Access	Access
$9018\ 0000_H$ - $901F\ FFFF_H$	512 Kbyte	CPU3.DLMU	Access	Access
$9020\ 0000_H$ - $9027\ FFFF_H$	512 Kbyte	CPU4.DLMU	Access	Access
$9028\ 0000_H$ - $902F\ FFFF_H$	512 Kbyte	CPU5.DLMU	Access	Access
$9030\ 0000_H$ - $903F\ FFFF_H$	-	Reserved	SRIBE	SRIBE
$9040\ 0000_H$ - $9047\ FFFF_H$	512 Kbyte	LMU0	Access	Access
$9048\ 0000_H$ - $904F\ FFFF_H$	512 Kbyte	LMU1	Access	Access
$9050\ 0000_H$ - $9057\ FFFF_H$	512 Kbyte	LMU2	Access	Access
$9058\ 0000_H$ - $905F\ FFFF_H$	512 Kbyte	LMU3	Access	Access
$9060\ 0000_H$ - $9067\ FFFF_H$	512 Kbyte	LMU4	Access	Access
$9068\ 0000_H$ - $906F\ FFFF_H$	512 Kbyte	LMU5	Access	Access
$9070\ 0000_H$ - $9077\ FFFF_H$	512 Kbyte	LMU6	Access	Access
$9078\ 0000_H$ - $907F\ FFFF_H$	512 Kbyte	LMU7	Access	Access
$9080\ 0000_H$ - $9087\ FFFF_H$	512 Kbyte	LMU8	Access	Access
$9088\ 0000_H$ - $908F\ FFFF_H$	512 Kbyte	LMU9	Access	Access
$9090\ 0000_H$ - $9207\ FFFF_H$	-	Reserved	SRIBE	SRIBE
$9208\ 0000_H$ - $920F\ FFFF_H$	512 Kbyte	PPU.CSM	Access	Access
$9210\ 0000_H$ - $97FF\ FFFF_H$	-	Reserved	SRIBE	SRIBE
$9800\ 0000_H$ - $9FFF\ FFFF_H$	128 Mbyte	xSPI	Access	Access

2.2.10.2 Segment 11

Table 22 Address map of segment 11

Address range	Size	Description	Access type	
			Read	Write
$B000\ 0000_H$ - $B007\ FFFF_H$	512 Kbyte	CPU0.DLMU	Access	Access
$B008\ 0000_H$ - $B00F\ FFFF_H$	512 Kbyte	CPU1.DLMU	Access	Access
$B010\ 0000_H$ - $B017\ FFFF_H$	512 Kbyte	CPU2.DLMU	Access	Access
$B018\ 0000_H$ - $B01F\ FFFF_H$	512 Kbyte	CPU3.DLMU	Access	Access
$B020\ 0000_H$ - $B027\ FFFF_H$	512 Kbyte	CPU4.DLMU	Access	Access
$B028\ 0000_H$ - $B02F\ FFFF_H$	512 Kbyte	CPU5.DLMU	Access	Access
$B030\ 0000_H$ - $B03F\ FFFF_H$	-	Reserved	SRIBE	SRIBE
$B040\ 0000_H$ - $B047\ FFFF_H$	512 Kbyte	LMU0	Access	Access
$B048\ 0000_H$ - $B04F\ FFFF_H$	512 Kbyte	LMU1	Access	Access
$B050\ 0000_H$ - $B057\ FFFF_H$	512 Kbyte	LMU2	Access	Access
$B058\ 0000_H$ - $B05F\ FFFF_H$	512 Kbyte	LMU3	Access	Access
$B060\ 0000_H$ - $B067\ FFFF_H$	512 Kbyte	LMU4	Access	Access
$B068\ 0000_H$ - $B06F\ FFFF_H$	512 Kbyte	LMU5	Access	Access
$B070\ 0000_H$ - $B077\ FFFF_H$	512 Kbyte	LMU6	Access	Access
$B078\ 0000_H$ - $B07F\ FFFF_H$	512 Kbyte	LMU7	Access	Access
$B080\ 0000_H$ - $B087\ FFFF_H$	512 Kbyte	LMU8	Access	Access
$B088\ 0000_H$ - $B08F\ FFFF_H$	512 Kbyte	LMU9	Access	Access
$B090\ 0000_H$ - $B203\ FFFF_H$	-	Reserved	SRIBE	SRIBE
$B204\ 0000_H$ - $B205\ FFFF_H$	128 Kbyte	PPU.VMEM	Access	Access
$B206\ 0000_H$ - $B207\ FFFF_H$	-	Reserved	SRIBE	SRIBE
$B208\ 0000_H$ - $B20F\ FFFF_H$	512 Kbyte	PPU.CSM	Access	Access
$B210\ 0000_H$ - $B7FF\ FFFF_H$	-	Reserved	SRIBE	SRIBE
$B800\ 0000_H$ - $BFFF\ FFFF_H$	128 Mbyte	xSPI	Access	Access

2.2.11 Segment 12

Table 23 Address map of segment 12

Address range	Size	Description	Access type	
			Read	Write
$C000\ 0000_H$ - $CFFF\ FFFF_H$	-	Reserved [1) 2)]	SRIBE	SRIBE

1) See the CPU chapter Local and global addressing sub-chapter for details
2) See the PPU chapter for details

2.2.12 Segment 13

Table 24 Address map of segment 13

Address range	Size	Description	Access type	
			Read	Write
$D000\ 0000_H$ - $DFFF\ FFFF_H$	-	Reserved [1)]	SRIBE	SRIBE

1) See the CPU chapter Local and global addressing sub-chapter for details

2.2.13 Segment 14

Table 25 Address map of segment 14

Address range	Size	Unit	Access type	
			Read	Write
$E000\ 0000_H$ - $E97F\ FFFF_H$	–	Reserved	LLIBE	LLIBE
$E980\ 0000_H$ - $E980\ FFFF_H$	64 Kbyte	PPU.SFR	Access	Access
$E981\ 0000_H$ - $EBFF\ FFFF_H$	–	Reserved	LLIBE	LLIBE
$EC00\ 0000_H$ - $EFFF\ FFFF_H$	64 Mbyte	PCIe1.DATA	Access	Access

2.2.14 Segment 15

The address map of segment 15 includes the following module address ranges:

- Absolute Addressing Range
  - If a module is addressed in the first 16 Kbyte of segment 15, the CPU can access the module with absolute addressing mode
- Others
  - If a module is addressed above the first 16 Kbyte of segment 15, the CPU can access the module with base + offset

Table 26 Address map of segment 15

Address range	Size	Unit	Access type	
			Read	Write
$F000\ 0000_H$ - $F000\ 07FF_H$	2 Kbyte	WTU	Access	Access
$F000\ 0800_H$ - $F002\ 3FFF_H$	–	Reserved	SPBBE	SPBBE
$F002\ 4000_H$ - $F002\ 43FF_H$	1 Kbyte	SCU	Access	Access
$F002\ 4400_H$ - $F002\ 7FFF_H$	-	Reserved	SPBBE	SPBBE
$F002\ 8000_H$ - $F002\ 9FFF_H$	8 Kbyte	SMU	Access	Access
$F002\ A000_H$ - $F003\ 9FFF_H$	-	Reserved	SPBBE	SPBBE
$F003\ A000_H$ - $F003\ A3FF_H$	1 Kbyte	P00	Access	Access
$F003\ A400_H$ - $F003\ A7FF_H$	1 Kbyte	P01	Access	Access
$F003\ A800_H$ - $F003\ ABFF_H$	1 Kbyte	P02	Access	Access
$F003\ AC00_H$ - $F003\ AFFF_H$	1 Kbyte	P03	Access	Access
$F003\ B000_H$ - $F003\ B3FF_H$	1 Kbyte	P04	Access	Access
$F003\ B400_H$ - $F003\ C7FF_H$	–	Reserved	SPBBE	SPBBE
$F003\ C800_H$ - $F003\ CBFF_H$	1 Kbyte	P10	Access	Access
$F003\ CC00_H$ - $F003\ D3FF_H$	–	Reserved	SPBBE	SPBBE
$F003\ D400_H$ - $F003\ D7FF_H$	1 Kbyte	P13	Access	Access
$F003\ D800_H$ - $F003\ DBFF_H$	1 Kbyte	P14	Access	Access
$F003\ DC00_H$ - $F003\ DFFF_H$	1 Kbyte	P15	Access	Access
$F003\ E000_H$ - $F003\ E3FF_H$	1 Kbyte	P16	Access	Access
$F003\ E400_H$ - $F003\ EFFF_H$	–	Reserved	SPBBE	SPBBE
$F003\ F000_H$ - $F003\ F3FF_H$	1 Kbyte	P20	Access	Access
$F003\ F400_H$ - $F003\ F7FF_H$	1 Kbyte	P21	Access	Access
$F003\ F800_H$ - $F003\ FBFF_H$	1 Kbyte	P22	Access	Access
$F003\ FC00_H$ - $F003\ FFFF_H$	1 Kbyte	P23	Access	Access
$F004\ 0000_H$ - $F004\ 03FF_H$	–	Reserved	SPBBE	SPBBE
$F004\ 0400_H$ - $F004\ 07FF_H$	1 Kbyte	P25	Access	Access
$F004\ 0800_H$ - $F004\ 17FF_H$	–	Reserved	SPBBE	SPBBE

(table continues...)

Table 26 (continued) Address map of segment 15

Address range	Size	Unit	Access type	
			Read	Write
F004 1800$_H$ - F004 1BFF$_H$	1 Kbyte	P30	Access	Access
F004 1C00$_H$ - F004 1FFF$_H$	1 Kbyte	P31	Access	Access
F004 2000$_H$ - F004 23FF$_H$	1 Kbyte	P32	Access	Access
F004 2400$_H$ - F004 27FF$_H$	1 Kbyte	P33	Access	Access
F004 2800$_H$ - F004 2BFF$_H$	1 Kbyte	P34	Access	Access
F004 2C00$_H$ - F004 2FFF$_H$	1 Kbyte	P35	Access	Access
F004 3000$_H$ - F004 3FFF$_H$	–	Reserved	SPBBE	SPBBE
F004 4000$_H$ - F004 43FF$_H$	1 Kbyte	P40	Access	Access
F004 4400$_H$ - F005 FFFF$_H$	–	Reserved	SPBBE	SPBBE
F006 0000$_H$ - F006 3FFF$_H$	16 Kbyte	SMM	Access	Access
F006 4000$_H$ - F006 7FFF$_H$	16 Kbyte	CCU	Access	Access
F006 8000$_H$ - F006 80FF$_H$	256 Byte	TRI	Access	Access
F006 8100$_H$ - F006 BFFF$_H$	–	Reserved	SPBBE	SPBBE
F006 C000$_H$ - F006 FFFF$_H$	16 Kbyte	VTMON	Access	Access
F007 0000$_H$ - F023 FFFF$_H$	–	Reserved	SPBBE	SPBBE
F024 0000$_H$ - F024 7FFF$_H$	32 Kbyte	SCR.XRAM	Access	Access
F024 8000$_H$ - F024 8FFF$_H$	–	Reserved	SPBBE	SPBBE
F024 9000$_H$ - F024 9FFF$_H$	4 Kbyte	PMS	Access	Access
F024 A000$_H$ - F03F FFFF$_H$	–	Reserved	SPBBE	SPBBE
F040 0000$_H$ - F041 FFFF$_H$	128 Kbyte	VMT0	Access	Access
F042 0000$_H$ - F043 FFFF$_H$	128 Kbyte	VMT1	Access	Access
F044 0000$_H$ - F045 FFFF$_H$	128 Kbyte	VMT2	Access	Access
F046 0000$_H$ - F047 FFFF$_H$	128 Kbyte	VMT3	Access	Access
F048 0000$_H$ - F049 FFFF$_H$	128 Kbyte	VMT4	Access	Access
F04A 0000$_H$ - F04B FFFF$_H$	128 Kbyte	VMT5	Access	Access
F04C 0000$_H$ - F04D FFFF$_H$	128 Kbyte	VMT6	Access	Access
F04E 0000$_H$ - F1FF FBFF$_H$	–	Reserved	SPBBE	SPBBE
F1FF FC00$_H$ - F1FF FFFF$_H$	1 Kbyte	SBCU	Access	Access
F200 0000$_H$ - F3FF FFFF$_H$	32 Mbyte	HSPHY	Access	Access
F400 0000$_H$ - F440 07FF$_H$	–	Reserved	COMPBBE	COMPBBE
F440 0800$_H$ - F440 09FF$_H$	512 Byte	MSC0	Access	Access
F440 0A00$_H$ - F440 0FFF$_H$	–	Reserved	COMPBBE	COMPBBE

(table continues...)

Table 26 (continued) Address map of segment 15

Address range	Size	Unit	Access type	
			Read	Write
F440 1000$_H$ - F440 11FF$_H$	512 Byte	QSPI0	Access	Access
F440 1200$_H$ - F440 13FF$_H$	512 Byte	QSPI1	Access	Access
F440 1400$_H$ - F440 15FF$_H$	512 Byte	QSPI2	Access	Access
F440 1600$_H$ - F440 17FF$_H$	512 Byte	QSPI3	Access	Access
F440 1800$_H$ - F440 19FF$_H$	512 Byte	QSPI4	Access	Access
F440 1A00$_H$ - F440 1BFF$_H$	512 Byte	QSPI5	Access	Access
F440 1C00$_H$ - F440 1DFF$_H$	512 Byte	QSPI6	Access	Access
F440 1E00$_H$ - F440 1FFF$_H$	512 Byte	QSPI7	Access	Access
F440 2000$_H$ - F440 2DFF$_H$	–	Reserved	COMPBBE	COMPBBE
F440 2E00$_H$ - F440 2FFF$_H$	512 Byte	FCE0	Access	Access
F440 3000$_H$ - F440 3BFF$_H$	6 x 512 Byte	SENT0	Access	Access
F440 3C00$_H$ - F440 3FFF$_H$	–	Reserved	COMPBBE	COMPBBE
F440 4000$_H$ - F440 4BFF$_H$	6 x 512 Byte	SENT1	Access	Access
F440 4C00$_H$ - F440 4FFF$_H$	–	Reserved	COMPBBE	COMPBBE
F440 5000$_H$ - F440 5BFF$_H$	6 x 512 Byte	PSI5	Access	Access
F440 5C00$_H$ - F440 6FFF$_H$	–	Reserved	COMPBBE	COMPBBE
F440 7000$_H$ - F440 7FFF$_H$	4 Kbyte	PSI5S0	Access	Access
F440 8000$_H$ - F440 FFFF$_H$	–	Reserved	COMPBBE	COMPBBE
F441 0000$_H$ - F441 3FFF$_H$	16 Kbyte	DMA0	Access	Access
F441 4000$_H$ - F441 7FFF$_H$	16 Kbyte	DMA1	Access	Access
F441 8000$_H$ - F441 BFFF$_H$	–	Reserved	COMPBBE	COMPBBE
F441 C000$_H$ - F441 CFFF$_H$	4 Kbyte	ERAY0	Access	Access
F441 D000$_H$ - F441 DFFF$_H$	4 Kbyte	ERAY1	Access	Access
F441 E000$_H$ - F442 FFFF$_H$	-	Reserved	COMPBBE	COMPBBE
F443 0000$_H$ - F443 1FFF$_H$	8 Kbyte	IR.SFR	Access	Access
F443 2000$_H$ - F443 5FFF$_H$	16 Kbyte	IR.SRC	Access	Access
F443 6000$_H$ - F447 FFFF$_H$	-	Reserved	COMPBBE	COMPBBE
F448 0000$_H$ - F448 03FF$_H$	4 x 256 Byte	HSSL0	Access	Access
F448 0400$_H$ - F448 FFFF$_H$	–	Reserved	COMPBBE	COMPBBE
F449 0000$_H$ - F449 FFFF$_H$	64 Kbyte	HSCT0	Access	Access
F44A 0000$_H$ - F44A 03FF$_H$	4 x 256 Byte	HSSL1	Access	Access
F44A 0400$_H$ - F44A FFFF$_H$	–	Reserved	COMPBBE	COMPBBE

(table continues...)

Table 26 (continued) Address map of segment 15

Address range	Size	Unit	Access type	
			Read	Write
F44B 0000$_H$ - F44B FFFF$_H$	64 Kbyte	HSCT1	Access	Access
F44C 0000$_H$ - F44D FFFF$_H$	128 Kbyte	I2C0	Access	Access
F44E 0000$_H$ - F44F FFFF$_H$	128 Kbyte	I2C1	Access	Access
F450 0000$_H$ - F451 FFFF$_H$	128 Kbyte	I2C2	Access	Access
F452 0000$_H$ - F469 FFFF$_H$	–	Reserved	COMPBBE	COMPBBE
F46A 0000$_H$ - F46A 7FFF$_H$	32 Kbyte	PCIe1.SFR	Access	Access
F46A 8000$_H$ - F46A FFFF$_H$	32 Kbyte	PCIe0.SFR	Access	Access
F46B 0000$_H$ - F46B 0FFF$_H$	4 Kbyte	SDMMC0	Access	Access
F46B 1000$_H$ - F46B FFFF$_H$	–	Reserved	COMPBBE	COMPBBE
F46C 0000$_H$ - F46C 01FF$_H$	512 Byte	ASCLIN0	Access	Access
F46C 0200$_H$ - F46C 03FF$_H$	512 Byte	ASCLIN1	Access	Access
F46C 0400$_H$ - F46C 05FF$_H$	512 Byte	ASCLIN2	Access	Access
F46C 0600$_H$ - F46C 07FF$_H$	512 Byte	ASCLIN3	Access	Access
F46C 0800$_H$ - F46C 09FF$_H$	512 Byte	ASCLIN4	Access	Access
F46C 0A00$_H$ - F46C 0BFF$_H$	512 Byte	ASCLIN5	Access	Access
F46C 0C00$_H$ - F46C 0DFF$_H$	512 Byte	ASCLIN6	Access	Access
F46C 0E00$_H$ - F46C 0FFF$_H$	512 Byte	ASCLIN7	Access	Access
F46C 1000$_H$ - F46C 11FF$_H$	512 Byte	ASCLIN8	Access	Access
F46C 1200$_H$ - F46C 13FF$_H$	512 Byte	ASCLIN9	Access	Access
F46C 1400$_H$ - F46C 15FF$_H$	512 Byte	ASCLIN10	Access	Access
F46C 1600$_H$ - F46C 17FF$_H$	512 Byte	ASCLIN11	Access	Access
F46C 1800$_H$ - F46C 19FF$_H$	512 Byte	ASCLIN12	Access	Access
F46C 1A00$_H$ - F46C 1BFF$_H$	512 Byte	ASCLIN13	Access	Access
F46C 1C00$_H$ - F46C 1DFF$_H$	512 Byte	ASCLIN14	Access	Access
F46C 1E00$_H$ - F46C 1FFF$_H$	512 Byte	ASCLIN15	Access	Access
F46C 2000$_H$ - F46C 21FF$_H$	512 Byte	ASCLIN16	Access	Access
F46C 2200$_H$ - F46C 23FF$_H$	512 Byte	ASCLIN17	Access	Access
F46C 2400$_H$ - F46C 25FF$_H$	512 Byte	ASCLIN18	Access	Access
F46C 2600$_H$ - F46C 27FF$_H$	512 Byte	ASCLIN19	Access	Access
F46C 2800$_H$ - F46C 29FF$_H$	512 Byte	ASCLIN20	Access	Access
F46C 2A00$_H$ - F46C 2BFF$_H$	512 Byte	ASCLIN21	Access	Access
F46C 2C00$_H$ - F46C 2DFF$_H$	512 Byte	ASCLIN22	Access	Access

(table continues...)

Table 26 (continued) Address map of segment 15

Address range	Size	Unit	Access type	
			Read	Write
F46C 2E00$_H$ - F46C 2FFF$_H$	512 Byte	ASCLIN23	Access	Access
F46C 3000$_H$ - F46C 31FF$_H$	512 Byte	ASCLIN24	Access	Access
F46C 3200$_H$ - F46C 33FF$_H$	512 Byte	ASCLIN25	Access	Access
F46C 3400$_H$ - F46C 35FF$_H$	512 Byte	ASCLIN26	Access	Access
F46C 3600$_H$ - F46C 37FF$_H$	512 Byte	ASCLIN27	Access	Access
F46C 3800$_H$ - F46F FFFF$_H$	–	Reserved	COMPBBE	COMPBBE
F470 0000$_H$ - F470 8FFF$_H$	36 Kbyte	CAN0.SRAM	Access	Access
F470 9000$_H$ - F470 FFFF$_H$	–	Reserved	COMPBBE	COMPBBE
F471 0000$_H$ - F471 10FF$_H$	4352 Byte	CAN0.SFR	Access	Access
F471 1100$_H$ - F471 FFFF$_H$	–	Reserved	COMPBBE	COMPBBE
F472 0000$_H$ - F472 4FFF$_H$	20 Kbyte	CAN1.SRAM	Access	Access
F472 5000$_H$ - F472 FFFF$_H$	–	Reserved	COMPBBE	COMPBBE
F473 0000$_H$ - F473 10FF$_H$	4352 Byte	CAN1.SFR	Access	Access
F473 1100$_H$ - F473 FFFF$_H$	–	Reserved	COMPBBE	COMPBBE
F474 0000$_H$ - F474 4FFF$_H$	20 Kbyte	CAN2.SRAM	Access	Access
F474 5000$_H$ - F474 FFFF$_H$	–	Reserved	COMPBBE	COMPBBE
F475 0000$_H$ - F475 10FF$_H$	4352 Byte	CAN2.SFR	Access	Access
F475 1100$_H$ - F475 FFFF$_H$	–	Reserved	COMPBBE	COMPBBE
F476 0000$_H$ - F476 4FFF$_H$	20 Kbyte	CAN3.SRAM	Access	Access
F476 5000$_H$ - F476 FFFF$_H$	–	Reserved	COMPBBE	COMPBBE
F477 0000$_H$ - F477 10FF$_H$	4352 Byte	CAN3.SFR	Access	Access
F477 1100$_H$ - F477 FFFF$_H$	–	Reserved	COMPBBE	COMPBBE
F478 0000$_H$ - F478 4FFF$_H$	20 Kbyte	CAN4.SRAM	Access	Access
F478 5000$_H$ - F478 FFFF$_H$	–	Reserved	COMPBBE	COMPBBE
F479 0000$_H$ - F479 10FF$_H$	4352 Byte	CAN4.SFR	Access	Access
F479 1100$_H$ - F47B FFFF$_H$	–	Reserved	COMPBBE	COMPBBE
F47C 0000$_H$ - F47C 3FFF$_H$	16 Kbyte	CANXL0.SRAM	Access	Access
F47C 4000$_H$ - F47C FFFF$_H$	–	Reserved	COMPBBE	COMPBBE
F47D 0000$_H$ - F47F FFFF$_H$	192 Kbyte	CANXL0.SFR	Access	Access
F480 0000$_H$ - F4FF FFFF$_H$	–	Reserved	COMPBBE	COMPBBE
F500 0000$_H$ - F500 1FFF$_H$	8 Kbyte	ADC.SFR	Access	Access
F500 2000$_H$ - F501 F7FF$_H$	–	Reserved	COMPBBE	COMPBBE

(table continues...)

Table 26 (continued) Address map of segment 15

Address range	Size	Unit	Access type	
			Read	Write
F501 F800$_H$ - F501 FFFF$_H$	2 Kbyte	TMADC.SFR	Access	Access
F502 0000$_H$ - F502 07FF$_H$	2 Kbyte	TMADC0 (Read Write)	Access	Access
F502 0800$_H$ - F502 0FFF$_H$	2 Kbyte	TMADC1 (Read Write)	Access	Access
F502 1000$_H$ - F502 17FF$_H$	2 Kbyte	TMADC2 (Read Write)	Access	Access
F502 1800$_H$ - F502 1FFF$_H$	2 Kbyte	TMADC3 (Read Write)	Access	Access
F502 2000$_H$ - F50B F7FF$_H$	–	Reserved	COMPBBE	COMPBBE
F50B F800$_H$ - F50B FFFF$_H$	2 Kbyte	CDSP.SFR	Access	Access
F50C 0000$_H$ - F50C 1FFF$_H$	8 Kbyte	CDSP0	Access	Access
F50C 2000$_H$ - F50C 3FFF$_H$	8 Kbyte	CDSP1	Access	Access
F50C 4000$_H$ - F5FF FBFF$_H$	–	Reserved	COMPBBE	COMPBBE
F5FF FC00$_H$ - F5FF FFFF$_H$	1 Kbyte	COMBCU	Access	Access
F600 0000$_H$ - F600 01FF$_H$	512 Byte	CSCU	Access	Access
F600 0200$_H$ - F600 03FF$_H$	512 Byte	TRNG	Access	Access
F600 0400$_H$ - F600 FFFF$_H$	–	Reserved	CSPBBE	CSPBBE
F601 0000$_H$ - F601 7FFF$_H$	32 Kbyte	PKC	Access	Access
F601 8000$_H$ - F6FF FBFF$_H$	–	Reserved	CSPBBE	CSPBBE
F6FF FC00$_H$ - F6FF FFFF$_H$	1 Kbyte	CSBCU	Access	Access
F700 0000$_H$ - F7FF FFFF$_H$	–	Reserved	SRIBE	SRIBE
F800 0000$_H$ - F800 7FFF$_H$	–	Reserved	SRIBE	SRIBE
F800 8000$_H$ - F800 8002$_H$	3 Byte	NVM.FSI_HOST (COMMx Registers)	Access	Access
F800 8003$_H$ - F802 7FFF$_H$	–	Reserved	SRIBE	SRIBE
F802 8000$_H$ - F802 8002$_H$	3 Byte	NVM.FSI_CSRM (COMMx Registers)	Access	Access
F802 8003$_H$ - F803 FFFF$_H$	–	Reserved	SRIBE	SRIBE
F804 0000$_H$ - F807 FFFF$_H$	256 Kbyte	NVM.DMU (incl. SFR)	Access	Access
F808 0000$_H$ - F808 FFFF$_H$	64 Kbyte	NVM.DMU (Host Cmd. Seq. Interpreter)	SRIBE	Access
F809 0000$_H$ - F80B FFFF$_H$	–	Reserved	SRIBE	SRIBE
F80C 0000$_H$ - F80C FFFF$_H$	64 Kbyte	NVM.DMU (CS Cmd. Seq. Interpreter)	SRIBE	Access
F80D 0000$_H$ - F83F FFFF$_H$	–	Reserved	SRIBE	SRIBE
F840 0000$_H$ - F840 0FFF$_H$	4 Kbyte	CPU0.FSFR	Access	Access
F840 1000$_H$ - F840 FFFF$_H$	–	Reserved	SRIBE	SRIBE
F841 0000$_H$ - F841 3FFF$_H$	16 Kbyte	NVM.PFRWB00.UR	Access	Access
F841 4000$_H$ - F841 7FFF$_H$	–	Reserved	SRIBE	SRIBE

(table continues...)

Table 26 (continued) Address map of segment 15

Address range	Size	Unit	Access type	
			Read	Write
F841 8000$_H$ - F841 BFFF$_H$	16 Kbyte	NVM.PFRWB01.UR	Access	Access
F841 C000$_H$ - F843 FFFF$_H$	–	Reserved	SRIBE	SRIBE
F844 0000$_H$ - F844 0FFF$_H$	4 Kbyte	CPU1.FSFR	Access	Access
F844 1000$_H$ - F844 FFFF$_H$	–	Reserved	SRIBE	SRIBE
F845 0000$_H$ - F845 3FFF$_H$	16 Kbyte	NVM.PFRWB10.UR	Access	Access
F845 4000$_H$ - F845 7FFF$_H$	–	Reserved	SRIBE	SRIBE
F845 8000$_H$ - F845 BFFF$_H$	16 Kbyte	NVM.PFRWB11.UR	Access	Access
F845 C000$_H$ - F847 FFFF$_H$	–	Reserved	SRIBE	SRIBE
F848 0000$_H$ - F848 0FFF$_H$	4 Kbyte	CPU2.FSFR	Access	Access
F848 1000$_H$ - F848 FFFF$_H$	–	Reserved	SRIBE	SRIBE
F849 0000$_H$ - F849 3FFF$_H$	16 Kbyte	NVM.PFRWB20.UR	Access	Access
F849 4000$_H$ - F849 7FFF$_H$	–	Reserved	SRIBE	SRIBE
F849 8000$_H$ - F849 BFFF$_H$	16 Kbyte	NVM.PFRWB21.UR	Access	Access
F849 C000$_H$ - F84B FFFF$_H$	–	Reserved	SRIBE	SRIBE
F84C 0000$_H$ - F84C 0FFF$_H$	4 Kbyte	CPU3.FSFR	Access	Access
F84C 1000$_H$ - F84C FFFF$_H$	–	Reserved	SRIBE	SRIBE
F84D 0000$_H$ - F84D 3FFF$_H$	16 Kbyte	NVM.PFRWB30.UR	Access	Access
F84D 4000$_H$ - F84D 7FFF$_H$	–	Reserved	SRIBE	SRIBE
F84D 8000$_H$ - F84D BFFF$_H$	16 Kbyte	NVM.PFRWB31.UR	Access	Access
F84D C000$_H$ - F84F FFFF$_H$	–	Reserved	SRIBE	SRIBE
F850 0000$_H$ - F850 0FFF$_H$	4 Kbyte	CPU4.FSFR	Access	Access
F850 1000$_H$ - F850 FFFF$_H$	–	Reserved	SRIBE	SRIBE
F851 0000$_H$ - F851 3FFF$_H$	16 Kbyte	NVM.PFRWB40.UR	Access	Access
F851 4000$_H$ - F851 7FFF$_H$	–	Reserved	SRIBE	SRIBE
F851 8000$_H$ - F851 BFFF$_H$	16 Kbyte	NVM.PFRWB41.UR	Access	Access
F851 C000$_H$ - F853 FFFF$_H$	–	Reserved	SRIBE	SRIBE
F854 0000$_H$ - F854 0FFF$_H$	4 Kbyte	CPU5.FSFR	Access	Access
F854 1000$_H$ - F854 FFFF$_H$	–	Reserved	SRIBE	SRIBE
F855 0000$_H$ - F855 3FFF$_H$	16 Kbyte	NVM.PFRWB50.UR	Access	Access
F855 4000$_H$ - F855 7FFF$_H$	–	Reserved	SRIBE	SRIBE
F855 8000$_H$ - F855 BFFF$_H$	16 Kbyte	NVM.PFRWB51.UR	Access	Access
F855 C000$_H$ - F857 FFFF$_H$	–	Reserved	SRIBE	SRIBE

(table continues...)

Table 26 (continued) Address map of segment 15

Address range	Size	Unit	Access type	
			Read	Write
F858 0000$_H$ - F858 0FFF$_H$	4 Kbyte	CPUcs.FSFR	Access	Access
F858 1000$_H$ - F858 FFFF$_H$	–	Reserved	SRIBE	SRIBE
F859 0000$_H$ - F859 3FFF$_H$	16 Kbyte	NVM.PFRWBcs.UR	Access	Access
F859 4000$_H$ - F87F FFFF$_H$	–	Reserved	SRIBE	SRIBE
F880 0000$_H$ - F880 FFFF$_H$	64 Kbyte	CPU0.SFR (incl. STM0 and OVL)	Access	Access
F881 0000$_H$ - F881 FFFF$_H$	64 Kbyte	CPU0.HR1.CSFR	Access	Access
F882 0000$_H$ - F882 FFFF$_H$	64 Kbyte	CPU0.HR2.CSFR	Access	Access
F883 0000$_H$ - F883 FFFF$_H$	64 Kbyte	CPU0.HR0.CSFR	Access	Access
F884 0000$_H$ - F884 FFFF$_H$	64 Kbyte	CPU1.SFR (incl. STM1 and OVL)	Access	Access
F885 0000$_H$ - F885 FFFF$_H$	64 Kbyte	CPU1.HR1.CSFR	Access	Access
F886 0000$_H$ - F886 FFFF$_H$	64 Kbyte	CPU1.HR2.CSFR	Access	Access
F887 0000$_H$ - F887 FFFF$_H$	64 Kbyte	CPU1.HR0.CSFR	Access	Access
F888 0000$_H$ - F888 FFFF$_H$	64 Kbyte	CPU2.SFR (incl. STM2 and OVL)	Access	Access
F889 0000$_H$ - F889 FFFF$_H$	64 Kbyte	CPU2.HR1.CSFR	Access	Access
F88A 0000$_H$ - F88A FFFF$_H$	64 Kbyte	CPU2.HR2.CSFR	Access	Access
F88B 0000$_H$ - F88B FFFF$_H$	64 Kbyte	CPU2.HR0.CSFR	Access	Access
F88C 0000$_H$ - F88C FFFF$_H$	64 Kbyte	CPU3.SFR (incl. STM3 and OVL)	Access	Access
F88D 0000$_H$ - F88D FFFF$_H$	64 Kbyte	CPU3.HR1.CSFR	Access	Access
F88E 0000$_H$ - F88E FFFF$_H$	64 Kbyte	CPU3.HR2.CSFR	Access	Access
F88F 0000$_H$ - F88F FFFF$_H$	64 Kbyte	CPU3.HR0.CSFR	Access	Access
F890 0000$_H$ - F890 FFFF$_H$	64 Kbyte	CPU4.SFR (incl. STM4 and OVL)	Access	Access
F891 0000$_H$ - F891 FFFF$_H$	64 Kbyte	CPU4.HR1.CSFR	Access	Access
F892 0000$_H$ - F892 FFFF$_H$	64 Kbyte	CPU4.HR2.CSFR	Access	Access
F893 0000$_H$ - F893 FFFF$_H$	64 Kbyte	CPU4.HR0.CSFR	Access	Access
F894 0000$_H$ - F894 FFFF$_H$	64 Kbyte	CPU5.SFR (incl. STM5 and OVL)	Access	Access
F895 0000$_H$ - F895 FFFF$_H$	64 Kbyte	CPU5.HR1.CSFR	Access	Access
F896 0000$_H$ - F896 FFFF$_H$	64 Kbyte	CPU5.HR2.CSFR	Access	Access
F897 0000$_H$ - F897 FFFF$_H$	64 Kbyte	CPU5.HR0.CSFR	Access	Access
F898 0000$_H$ - F898 FFFF$_H$	64 Kbyte	CPUcs.SFR (incl. STMcs)	Access	Access
F899 0000$_H$ - F899 FFFF$_H$	64 Kbyte	CPUcs.HR1.CSFR	Access	Access
F89A 0000$_H$ - F8CF FFFF$_H$	–	Reserved	SRIBE	SRIBE
F8D0 0000$_H$ - F8D4 FFFF$_H$	320 Kbyte	CSS	Access	Access

(table continues...)

Table 26 (continued) Address map of segment 15

Address range	Size	Unit	Access type	
			Read	Write
F8D5 0000$_H$ - F8EF FFFF$_H$	–	Reserved	SRIBE	SRIBE
F8F0 0000$_H$ - F8F0 FFFF$_H$	64 Kbyte	SRI0.SFR	Access	Access
F8F1 0000$_H$ - F8F1 FFFF$_H$	64 Kbyte	SRI1.SFR	Access	Access
F8F2 0000$_H$ - F8F2 FFFF$_H$	64 Kbyte	SRI2.SFR	Access	Access
F8F3 0000$_H$ - F8F3 FFFF$_H$	64 Kbyte	SRI3.SFR	Access	Access
F8F4 0000$_H$ - F8F4 FFFF$_H$	64 Kbyte	SRI4.SFR	Access	Access
F8F5 0000$_H$ - F8F5 FFFF$_H$	64 Kbyte	SRI5.SFR	Access	Access
F8F6 0000$_H$ - F8F6 FFFF$_H$	64 Kbyte	SRI6.SFR	Access	Access
F8F7 0000$_H$ - F8F7 FFFF$_H$	64 Kbyte	SRI7.SFR	Access	Access
F8F8 0000$_H$ - F8FF FFFF$_H$	–	Reserved	SRIBE	SRIBE
F900 0000$_H$ - F901 FFFF$_H$	128 Kbyte	GETH0.SFR	Access	Access
F902 0000$_H$ - F902 FFFF$_H$	64 Kbyte	GETH0.RAM	Access	Access
F903 0000$_H$ - F903 7FFF$_H$	32 Kbyte	DRE.SFR	Access	Access
F903 8000$_H$ - F903 FFFF$_H$	32 Kbyte	DRE.RAM	Access	Access
F904 0000$_H$ - F904 FFFF$_H$	64 Kbyte	xSPI.SFR	Access	Access
F905 0000$_H$ - F907 FFFF$_H$	–	Reserved	SRIBE	SRIBE
F908 0000$_H$ - F909 FFFF$_H$	128 Kbyte	eGTM.CLUSTER0	Access	Access
F90A 0000$_H$ - F90B FFFF$_H$	128 Kbyte	eGTM.CLUSTER1	Access	Access
F90C 0000$_H$ - F90D FFFF$_H$	128 Kbyte	eGTM.CLUSTER2	Access	Access
F90E 0000$_H$ - F90E FFFF$_H$	64 Kbyte	eGTM.SFR	Access	Access
F90F 0000$_H$ - F93F FFFF$_H$	–	Reserved	SRIBE	SRIBE
F940 0000$_H$ - F943 FFFF$_H$	256 Kbyte	LETH0	Access	Access
F944 0000$_H$ - F980 FFFF$_H$	–	Reserved	SRIBE	SRIBE
F981 0000$_H$ - F981 FFFF$_H$	64 Kbyte	PPU.STUDMI	Access	Access
F982 0000$_H$ - F982 FFFF$_H$	64 Kbyte	PPU.DEBUG	Access	Access
F983 0000$_H$ - F983 FFFF$_H$	64 Kbyte	PPU.SAFETY	Access	Access
F984 0000$_H$ - F984 FFFF$_H$	64 Kbyte	PPU.AP	Access	Access
F985 0000$_H$ - F985 FFFF$_H$	64 Kbyte	PPU.CSM.AP	Access	Access
F986 0000$_H$ - F986 FFFF$_H$	64 Kbyte	PPU.VMEM.AP	Access	Access
F987 0000$_H$ - F98F FFFF$_H$	–	Reserved	SRIBE	SRIBE
F990 0000$_H$ - F990 FFFF$_H$	64 Kbyte	LLI.SFR	Access	Access
F991 0000$_H$ - F9FF FFFF$_H$	–	Reserved	SRIBE	SRIBE

(table continues...)

Table 26 (continued) Address map of segment 15

Address range	Size	Unit	Access type	
			Read	Write
FA00 0000$_H$ - FA01 FFFF$_H$	128 Kbyte	MCDS4P (incl. 32 Kbyte TBUF)	Access	Access
FA02 0000$_H$ - FA03 FFFF$_H$	–	Reserved	TPBBE	TPBBE
FA04 0000$_H$ - FA05 FFFF$_H$	128 Kbyte	MCDS2P (incl. 16 Kbyte TBUF)	Access	Access
FA06 0000$_H$ - FA0F FFFF$_H$	–	Reserved	TPBBE	TPBBE
FA10 0000$_H$ - FA10 1FFF$_H$	8 Kbyte	TRIF	Access	Access
FA10 2000$_H$ - FA17 FFFF$_H$	–	Reserved	TPBBE	TPBBE
FA18 0000$_H$ - FA18 03FF$_H$	1 Kbyte	CBS	Access	Access
FA18 0400$_H$ - FAFF FBFF$_H$	–	Reserved	TPBBE	TPBBE
FAFF FC00$_H$ - FAFF FFFF$_H$	1 Kbyte	TBCU	Access	Access
FB00 0000$_H$ - FB00 FFFF$_H$	64 Kbyte	LMU0.SFR	Access	Access
FB01 0000$_H$ - FB01 FFFF$_H$	64 Kbyte	LMU1.SFR	Access	Access
FB02 0000$_H$ - FB02 FFFF$_H$	64 Kbyte	LMU2.SFR	Access	Access
FB03 0000$_H$ - FB03 FFFF$_H$	64 Kbyte	LMU3.SFR	Access	Access
FB04 0000$_H$ - FB04 FFFF$_H$	64 Kbyte	LMU4.SFR	Access	Access
FB05 0000$_H$ - FB05 FFFF$_H$	64 Kbyte	LMU5.SFR	Access	Access
FB06 0000$_H$ - FB06 FFFF$_H$	64 Kbyte	LMU6.SFR	Access	Access
FB07 0000$_H$ - FB07 FFFF$_H$	64 Kbyte	LMU7.SFR	Access	Access
FB08 0000$_H$ - FB08 FFFF$_H$	64 Kbyte	LMU8.SFR	Access	Access
FB09 0000$_H$ - FB09 FFFF$_H$	64 Kbyte	LMU9.SFR	Access	Access
FB0A 0000$_H$ - FBFF FFFF$_H$	–	Reserved	SRIBE	SRIBE
FC00 0000$_H$ - FFFF FFFF$_H$	64 Mbyte	PCIe0.DATA	Access	Access

2.3 System address map revision history

Reference	Description of change(s)
Date range: 2024-08-17 to 2025-04-08	
Segment 15	• SCR.PER.SFR removed from Segment 15

3 Distributed functionality

Topics:

- System interconnect
- Software Over The Air (SOTA)
- Resource access latencies
- Module reset

3.1 System interconnect

The system interconnect connects the various master agents and slave agents in AURIX™ and allows them to communicate with each other. It is tailored to meet the bandwidth and latency requirements of the various data flows between master and slave agents.

3.1.1 Functional overview

The system interconnect is composed of a number of interconnect segments and interconnect protocols. There are bridges implemented between the different interconnect segments such that any master agent in the system can access any slave agent in the system, except where explicitly disabled due to security or other reasons. The result is a uniform system address map, every part of which can be accessed by every master agent in the system[2]. These bridges and their capabilities are described further in the SRI chapter.

[2] There are a few exceptions to this uniform access as noted in the system address map chapter.

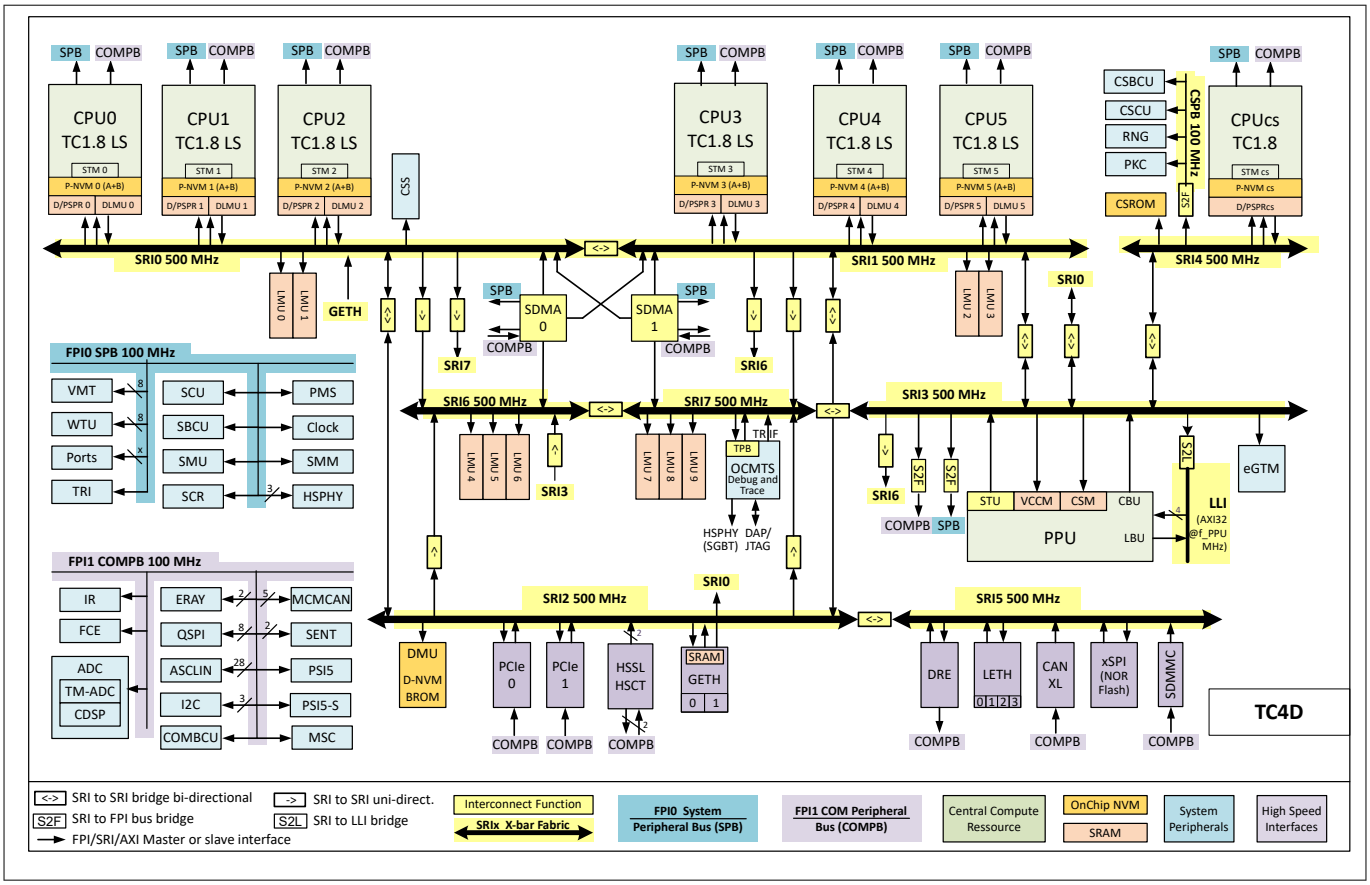


Figure 5 TC4Dx block diagram

3.1.2 Functional description

The system interconnect is implemented using the following interconnect segments and interconnect protocols.

- System Resource Interconnect (SRI) protocol
  - The SRI is a 64-bit interconnect used to connect the TriCore™ CPUs, the system DMA and other high bandwidth requestors to high bandwidth memories and other resources for instruction fetches and data accesses
  - The SRI fabric provides safety through end-to-end EDC and other safety mechanisms
  - The SRI fabric implements the SRI protocol and consists of multiple crossbars labeled SRIx (x = 0-7)
- Flexible Peripheral Interconnect (FPI) protocol
  - The FPI is a 32-bit multi-master interconnect
  - The FPI implementations provide safety through end-to-end EDC and other safety mechanisms
  - The following (device-specific) interconnect segments implement the FPI protocol
    - System peripheral bus (SPB): The SPB connects the TriCore™ CPUs, the system DMA and other SPB masters to medium and low bandwidth functional blocks
    - Converter peripheral bus (CPB): The CPB connects the TriCore™ CPUs and the system DMA to the GTM and related functional blocks
    - Communications peripheral bus (ComPB): The ComPB connects masters to the off-chip communications functional blocks

  - Cyber-security peripheral bus (CSPB): The CSPB connects the CPUcs to dedicated CSRM functional blocks
  - Tool peripheral bus (TPB): The TPB connects masters to the TRACE functional block and its associated functional blocks
  - Radar control bus (RCB): The RCB connects masters to the Radar functional blocks
- AMBA AXI protocol with user extensions
  - The Low Latency Interconnect (LLI) implements a 32-bit AXI interconnect which consists of a single crossbar with a limited number of master and slave agents
  - The LLI implementation provides safety through end-to-end EDC and other safety mechanisms, implemented as user extensions to the AMBA AXI protocol
  - The LLI provides a low latency path from the PPU and other SRI masters to the PPU configuration and debug registers, the eGTM, HRPWM (when present on the device) and the TMADC

3.1.2.1 Master agent function identifier

Each master agent function on the system interconnect is assigned a unique master agent function identifier (ID). The master agent function identifier is transmitted in the address phase of any system interconnect transaction. The master agent function identifier can be used at the slave function interface through the access protection mechanism to restrict the functions allowed read and write access to its internal resources. The master agent function ID in combination with the access protection in the slave agents is used for several system functions such as access protection, software encapsulation, hardware virtualization, software debug and trace.

The master agent function ID consists of:

- TAG-ID, master agent function TAG identifier (TAG ID), 6-bit
- VM, Virtual machine information (VM), 3-bit, optional information
- VM_VALID, signal which indicates whether the transmitted VM is to be used (VM validity indicator)
- PRS, Protection register information (PRS), 3-bit, optional information
- PRS_VALID, signal which indicates whether the transmitted PRS is to be used (PRS validity indicator)

Master agents which are not using the VM information for their master agent function ID drive the VM validity indicator to 0_B. Similarly, master agents which are not using the PRS information for their master agent function ID drive the PRS validity indicator to 0_B.

3.1.2.1.1 Master agent function TAG-ID assignments

The following table lists the master agent function TAG identifier for each master agent function in the microcontroller.

***Note**: Where TAG-IDs are assigned to multiple modules it is ensured that only one of the multiple modules is available in a dedicated device. Example: TAG-IDs assigned to SDMA1 and RIF0/1, RMEM, RDMA .*

Table 27 Master agent function TAG-ID assignments

TAG-ID		Module	Location	Visible	Description	Short name
000000_B	0_H	CPU0	SRI/SPB/CPB	All	DMI/DMBI - D TAG ID	CPU0.D
000001_B	1_H	CPU0	SRI/SPB/CPB	All	DMI/DMBI - DS TAG ID	CPU0.DS
000010_B	2_H	CPU1	SRI/SPB/CPB	All	DMI/DMBI - D TAG ID	CPU1.D
000011_B	3_H	CPU1	SRI/SPB/CPB	All	DMI/DMBI - DS TAG ID	CPU1.DS
000100_B	4_H	CPU2	SRI/SPB	All	DMI/DMBI - D TAG ID	CPU2.D
000101_B	5_H	CPU2	SRI/SPB	All	DMI/DMBI - DS TAG ID	CPU2.DS
000110_B	6_H	CPU3	SRI/SPB	All	DMI/DMBI - D TAG ID	CPU3.D
000111_B	7_H	CPU3	SRI/SPB	All	DMI/DMBI - DS TAG ID	CPU3.DS
001000_B	8_H	CPU4	SRI/SPB	All	DMI/DMBI - D TAG ID	CPU4.D
001001_B	9_H	CPU4	SRI/SPB	All	DMI/DMBI - DS TAG ID	CPU4.DS
001010_B	A_H	CPU5	SRI/SPB	All	DMI/DMBI - D TAG ID	CPU5.D
001011_B	B_H	CPU5	SRI/SPB	All	DMI/DMBI - DS TAG ID	CPU5.DS
001100_B	C_H	CPUCS	SRI/SPB	All	DMI/DMBI - D TAG ID	CPUCS.D
001101_B	D_H	CPUCS	SRI/SPB	All	DMI/DMBI - DS TAG ID	CPUCS.DS
001110_B	E_H	PCIE1	SRI	All	PCI Express 1	PCIE
001111_B	F_H	PPU	LLI	LLI	PPU LBU TAG-ID	PPU.LBU
010000_B	10_H	DMA0	SRI/SPB/CPB	All	DMA0 Resource Partition TAG-ID Pool Unique IDs for the individual DMA0 PRS groups can be configured through the additional use of VMID and PRS	DMA0.T0
010001_B	11_H	DMA0	SRI/SPB/CPB	All	DMA0 Resource Partition TAG-ID Pool	DMA0.T1
010010_B	12_H	DMA0	SRI/SPB/CPB	All	DMA0 Resource Partition TAG-ID Pool	DMA0.T2
010011_B	13_H	DMA0	SRI/SPB/CPB	All	DMA0 Resource Partition TAG-ID Pool	DMA0.T3
010100_B	14_H	DMA0	SRI/SPB/CPB	All	DMA0 Resource Partition TAG-ID Pool	DMA0.T4

(table continues...)

Table 27 (continued) Master agent function TAG-ID assignments

TAG-ID		Module	Location	Visible	Description	Short name
010101_B	15_H	DMA0	SRI/SPB/CPB	All	DMA0 Resource Partition TAG-ID Pool	DMA0.T5
010110_B	16_H	DMA0	SRI/SPB/CPB	All	DMA0 Resource Partition TAG-ID Pool	DMA0.T6
010111_B	17_H	DMA0	SRI/SPB/CPB	All	DMA0 Resource Partition TAG-ID Pool	DMA0.T7
011000_B	18_H	HSSL0	SRI/SPB	All	High Speed Serial Link 0	HSSL0
011001_B	19_H	HSSL1	SRI/SPB	All	High Speed Serial Link 1	HSSL1
011010_B	$1A_H$	PPU.CBU	SRI	All	PPU, Combined Instruction and Data Interface	PPU.CBU
011011_B	$1B_H$	GTM	SRI	All	GTM	GTM
011100_B	$1C_H$	Cerberus	SRI	All	Cerberus (DAP/JTAG) and IOCE (DAPE)	CBS/IOCE
011101_B	$1D_H$	PCIE0	SRI	All	PCI Express 0	PCIE
011110_B	$1E_H$	DRE	SRI/SPB	All	Data Routing Engine TAG-ID Pool	DRE.T0
011111_B	$1F_H$	DRE	SRI/SPB	All	Data Routing Engine TAG-ID Pool	DRE.T1
100000_B	20_H	DMA1	SRI/SPB	All	DMA1 Resource Partition TAG-ID Pool in devices with DMA1	DMA1.T0
100001_B	21_H	DMA1	SRI/SPB	All	DMA1 Resource Partition TAG-ID Pool in devices with DMA1	DMA1.T1
100010_B	22_H	DMA1/ RMEM	SRI/SPB	All	DMA1 Resource Partition TAG-ID Pool in devices with DMA1 RMEM in devices with RMEM	DMA1.T2 RMEM
100011_B	23_H	DMA1/ RDMA	SRI/SPB	All	DMA1 Resource Partition TAG-ID Pool in devices with DMA1 RDMA in devices with RDMA	DMA1.T3 RDMA
100100_B	24_H	DMA1/ RIF0	SRI/SPB	All	DMA1 Resource Partition TAG-ID Pool in devices with DMA1 RIF0 in devices with RIF0	DMA1.T4 RIF0
100101_B	25_H	DMA1/ RIF1	SRI/SPB	All	DMA1 Resource Partition TAG-ID Pool in devices with DMA1 RIF1 in devices with RIF1	DMA1.T5 RIF1
100110_B	26_H	DMA1/ RIF2	SRI/SPB	All	DMA1 Resource Partition TAG-ID Pool in devices with DMA1 RIF2 in devices with RIF2	DMA1.T6 RIF2
100111_B	27_H	DMA1	SRI/SPB	All	DMA1 Resource Partition TAG-ID Pool in devices with DMA1	DMA1.T7
101000_B	28_H	CPU0	SRI	SRIx	PMI/PMBI	CPU0.P

(table continues...)

Table 27 (continued) Master agent function TAG-ID assignments

TAG-ID		Module	Location	Visible	Description	Short name
101001_B	29_H	CPU1	SRI	SRIx	PMI/PMBI	CPU1.P
101010_B	$2A_H$	CPU2	SRI	SRIx	PMI/PMBI	CPU2.P
101011_B	$2B_H$	CPU3	SRI	SRIx	PMI/PMBI	CPU3.P
101100_B	$2C_H$	CPU4	SRI	SRIx	PMI/PMBI	CPU4.P
101101_B	$2D_H$	CPU5	SRI	SRIx	PMI/PMBI	CPU5.P
101110_B	$2E_H$	CPUCS	SRI	SRIx	PMI/PMBI	CPUCS.P
101111_B	$2F_H$	-	-	-	Reserved	-
110000_B	30_H	GETH	SRI	All	GETH DMA Channel 0	GETH.C0
110001_B	31_H	GETH	SRI	All	GETH DMA Channel 1	GETH.C1
110010_B	32_H	GETH	SRI	All	GETH DMA Channel 2	GETH.C2
110011_B	33_H	GETH	SRI	All	GETH DMA Channel 3	GETH.C3
110100_B	34_H	GETH	SRI	All	GETH DMA Channel 4	GETH.C4
110101_B	35_H	GETH	SRI	All	GETH DMA Channel 5	GETH.C5
110110_B	36_H	GETH	SRI	All	GETH DMA Channel 6	GETH.C7
110111_B	37_H	GETH	SRI	All	GETH DMA Channel 7	GETH.C7
111000_B	38_H	PPU.STU	SRI	All	PPU, Streaming Data Interface	PPU.STU
111001_B	39_H	SDMMC	SPB	All	SDMMC DMA	SDMMC
111010_B	$3A_H$	XSPI	SRI	All	xSPI DMA	XSPI
111011_B	$3B_H$	CANXL	SRI	All	CANXL (XCAN0 - XCAN3). The ID is used for all nodes DMA in the CANXL instance. Unique IDs for the individual DMA channels can be configured by additionally using the VMID and PRS information.	CANXL
111100_B	$3C_H$	LETH0/ LETH1	SRI	All	LETH TAG-ID Pool. The ID can be used in both LETH instances. Unique IDs for the individual DMA channels can be configured by additionally using the VMID and PRS information.	LETH
111101_B	$3D_H$	LETH0/ LETH1	SRI	All	LETH TAG-ID Pool. The ID can be used in both LETH instances. Unique IDs for the individual DMA channels can be configured by additionally using the VMID and PRS information.	LETH
111110_B	$3E_H$	OCMTS TRIF/TME	SRI	All	OCMTS Trace Move Engine for MCDS data	TRIF

(table continues...)

Table 27 (continued) Master agent function TAG-ID assignments

TAG-ID		Module	Location	Visible	Description	Short name
111111_B	$3F_H$	-	-	-	Reserved Not used by any master agent function for interconnect access.	-

3.1.2.2 Access protection

The system interconnect provides access protection mechanisms that enable application software to restrict read and write accesses to each slave function from a desired set of master agent functions.

These access protection mechanisms can be used to assign on chip resources like system functions, CPUs, volatile and non-volatile memories, communication functions, DMA resource partitions and Interrupt Router service request nodes to an application software instance. This can be used to partition the MCU resources between multiple applications and provide logical freedom from interference between these applications. In addition it supports CPU hypervisor functions with access protection control on VM ID and PRS granularity.

The system interconnect access protection mechanisms are an essential feature of the TC4xx safety and security architecture.

3.1.2.2.1 Feature list

- The PROT mechanism provides the ability to assign ownership of a set of registers to a single master agent function, known as the PROT owner
  - Once assigned, only the PROT owner is permitted to lock or unlock the protected registers and modify them
- The Access Protection Unit (APU) provides the ability to protect and restrict read and write access to a slave function
  - An APU can be configured to allow no access, read-only access, write-only access or read and write access to a slave function for each master agent function
  - The master agent function is identified by a master agent function identifier, which consists of a 6-bit TAG-ID, an optional 3-bit virtual machine (VM) ID and an optional 3-bit protection register set (PRS) ID
  - For system memories, the APU allows a smaller region of the memory region to be configured for protection by specifying the start and end addresses of the protection region

3.1.2.2.2 Functional overview

Access protection is implemented by gating accesses at the slave interface based on the master agent function that initiated the access. Each on chip resource with interconnect mastering capability has a unique master agent function identifier consisting of a TAG ID, VM ID and PRS ID that can be used to identify the origin of an on chip transaction. Each slave function implements register and memory access protection as necessary in order to provide freedom from interference between different application software instances attempting to access that slave function.

3.1.2.2.3 Functional description

Access protection in the slave interfaces is implemented using two functions which interact with each other.

1. The PROT mechanism allows a coarse-grained access protection to groups of registers
   a. It is usually used to protect registers that are not protected by an APU (for example, the APU registers themselves)
   b. It allows for a safe and secure assignment of resources to master agent functions. Each group of registers may be assigned to a single PROT owner master agent function
   c. It provides write protection to the group of registers
   d. It also allows software to lock control and configuration registers from further modification and then later unlock the registers in a safe and secure manner when required to do so
2. The Access Protection Unit (APU) provides fine-grained access protection to groups of registers related to a specific function or to volatile and non-volatile memories
   a. Each group of registers may be enabled for access from a configurable set of master agent functions
   b. It provides separate controls to allow sets of master agent functions read-only access, write-only access, read and write access or to disallow access from them
   c. It can also be used to provide access protection to configurable regions of a memory, allowing safe and secure partitioning of the memory between the different groups of master functions

The access protection logic (including PROT and APU) is reset on application reset and is not affected by either module or kernel reset.

Each functional block connected to the system interconnect implements access protection to its slave functions through one or more PROT and APU instances. The access protection details for each register in the functional block is provided in the functional block specific register overview table. This table contains columns specifying the read and write access protection mechanism for each register, including details of which PROT, APU or other mechanism provides the protection.

PROT mechanism

The "PROT" resource protection concept uses a PROT state machine, a PROT register and access gates to control access to the protected registers and hence to the protected resource itself.

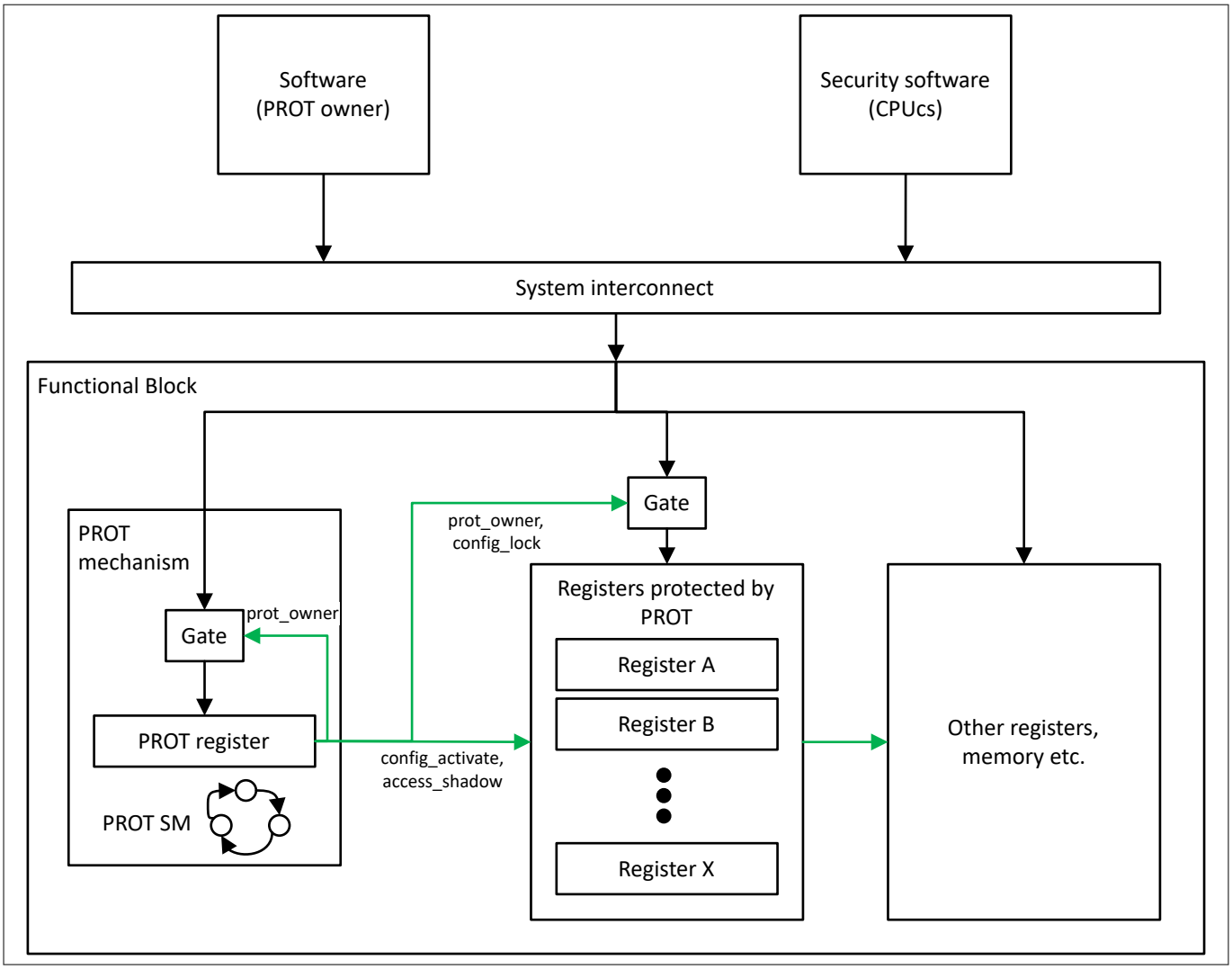


Figure 6 PROT concept

The software actors involved in the PROT concept are:

- The PROT owner: A configurable master agent function which is permitted to configure the PROT as well as the protected resource registers
- The secure master: A master agent function that is capable of generating CS transactions on the system interconnect. In particular, the CPUcs and a secure SDMA resource partition may generate CS transactions on the system interconnect which are marked by an asserted cs_valid bit on FPI buses and an asserted sri_sm bit on SRI interconnects.

The hardware blocks that interact to implement the PROT concept are:

- The PROT mechanism
  - The PROT state machine: A state machine that has a defined set of transitions supported in each state. The current state is used to control the access gates
  - The PROT register: Allows configuration and visibility of the PROT owner and the PROT state
  - The access gate to the PROT register: Controls access to the PROT register based on the PROT owner
- The protected registers: The set of registers which configure or control the protected function and which have to be protected against unsafe or non-secure modification
- The protected function: The module or logic which is configured or controlled by the protected registers
- The access gate to the protected registers: Uses information about the current PROT state and PROT owner to gate accesses to the protected registers to implement the protection

The "protection" aspect of the PROT concept is realized by ensuring that critical operations like configurations can only be done by the PROT owner in specified states which are not the normal operation states. These critical states can only be reached by specific PROT state transitions, triggered again by specified masters.

Resources that have their control shared between application software and security software are handled by having the (safe) PROT owner do the configuration and then having the security software check the configuration before it becomes active.

PROT state machine

The PROT state machine has a set of protection states and the possible transitions out of those states. Each transition is also annotated with the master agent function that is allowed to trigger that transition.

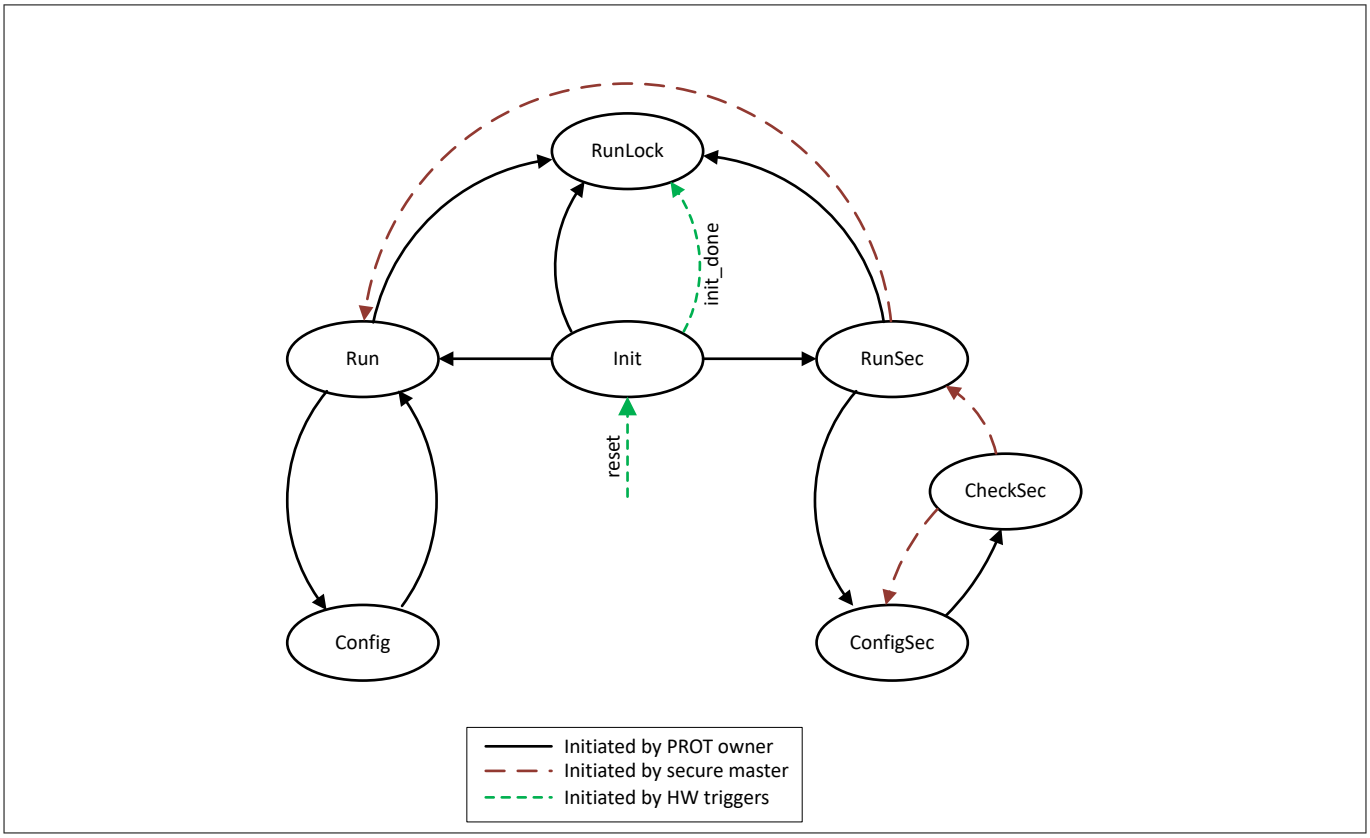


Figure 7 PROT state machine

State transitions are triggered by explicit software writes to the STATE field of the PROT register, except in the following cases:

1. PROT state machines will transition to the Init state on assertion of the application reset
2. PROT state machines which are in the Init state will transition to the RunLock state in response to the assertion of the init_done input signal
   - **a.** The assertion of the init_done input signal for cyber-security relevant PROT instances is controlled by the CSCU_INIT.DONE bit-field. See chapter CSRM for the list of PROT instances that are cyber-security relevant
   - **b.** The assertion of the init_done input signal for all other PROT instances is controlled by the SMM_PROTINIT.PINIT_DONE bit-field

Init state

The PROT state machine resets to the Init state. The protected function configuration may be modified in this state. The expected usage is that after reset, software does initial configuration of the protected function in the Init state, then sets the PROT owner and transitions to one of the Run, RunLock or RunSec states.

RunLock state

The RunLock state is used for functions which are configured once and do not need to be reconfigured until the next application reset.

The protected registers cannot be modified in the RunLock state and there are no transitions out of the RunLock state that permit later modification of the protected registers. Therefore the safety and security of the protected resource is ensured if the initial configuration is done safely and securely.

***Note**: There are two valid encodings of the RunLock state - 110_B and 111_B. Both these encodings may be used by the hardware or software and are indistinguishable from each other in terms of behavior.*

Run, config states

The Run state is the normal operation state of a PROT mechanism that is owned by a single master agent function. The protected registers cannot be modified in the Run state.

When the PROT owner wishes to reconfigure the protected registers, it transitions the PROT state to Config. The protected registers may be modified by the PROT owner in the Config state. Once the configuration is complete, the PROT owner transitions the PROT state back to Run, re-locking the protected registers.

RunSec, ConfigSec, CheckSec states

The RunSec state is similar to the Run state, except that it is used for functions that need shared control between application software and security software. The protected registers cannot be modified in the RunSec state.

When using the RunSec state, it is expected that software has configured the PROT owner to be a safe master agent function (for example, a lock-stepped CPU). A safe master agent function is one that is related to the safety task or safety application. The state transitions out of RunSec enforce a safe and secure interaction between the safe (though possibly malicious) PROT owner and the secure master. Note that if the PROT owner is malicious no safety can be guaranteed.

A configuration is always initiated by the PROT owner which writes the PROT state to transition to ConfigSec. This allows the PROT owner to then modify the protected registers. To ensure the security of the system, the modifications to the protected configuration registers should not immediately become active - they should only become active once security software has checked them for security issues and certified them. This is done by either:

- Disabling the protected function (access to the protected function, interconnect mastering by the protected function and so on) during reconfiguration such that any configuration that has not yet been checked by the security software is never used by the protected function
- Shadowing the protected registers so that the protected function can continue using the previous configuration while software is reconfiguring the shadow registers. The shadow registers are copied, moved or swapped to the active registers after the security checks have passed

***Note**: Shadowing of the protected registers is only implemented in a few functions like system memories where shared control between application software and security software is necessary. Details of the shadowing can be found in the respective functional block (module) chapters.*

Once the configuration is done, the PROT owner then writes the PROT state register to transition it to CheckSec. In this state, modification of the protected registers is no longer possible. This allows security software to check the new configuration before it is made active. If the security checks pass, the security software running on the secure master writes the PROT state register to transition back to the RunSec state. This transition from CheckSec to RunSec also triggers the activation of the new configuration of the protected registers. If the security checks fail, the security software running on the secure master writes the PROT state back to ConfigSec, indicating to the PROT owner that the security checks have failed and that it should fix the configuration. The PROT owner then takes further action to fix the configuration and retry the security checks or raises an alarm if the failure is unexpected.

The transition from RunSec back to Run state is only needed in special cases, for instance when a function is released from the secure domain back to a single-ownership usage model.

PROT access checks

PROT owner identification

The conditions for matching an incoming interconnect transaction to the PROT owner are as follows.

1. If PROT.ODEF is 0_B, the match condition is true regardless of the master agent function initiating the interconnect transaction. This means that any master agent function can behave as the PROT owner
2. If PROT.ODEF is 1_B
   a. The master agent function TAG ID on the interconnect transaction has to exactly match the value in the PROT.TAGID bit-field of the PROT register
   b. If PROT.VMEN is 0_B, the VM validity indicator and the VM ID in the interconnect transaction are ignored and not used in the match condition
   c. If PROT.VMEN is 1_B, the interconnect transaction has to have the VM validity indicator set and the VM ID in the interconnect transaction has to match the value in the PROT.VM bit-field of the PROT register
   d. If PROT.PRSEN is 0_B, the PRS validity indicator and the PRS in the interconnect transaction are ignored and not used in the match condition
   e. If PROT.PRSEN is 1_B, the interconnect transaction has to have the PRS validity indicator set and the PRS in the interconnect transaction has to match the value in the PROT.PRS bit-field of the PROT register

Secure master identification

Accesses from the secure master are identified through the cs_valid bit on FPI interconnects and the sri_sm bit on SRI interconnects.

PROT state transition checks

The state transitions in the PROT state machine can only be initiated by either the PROT owner, the secure master or by hardware. The following table describes the valid PROT state transitions, the master agent functions that can trigger each transition and the response to each type of trigger. Writes to the PROT.STATE bit-field with PROT.SWEN=1_B from a master agent function that is not allowed to initiate that state transition or that cause a state transition not listed in the table below (including self-state transitions) will not be executed and will terminate with an alarm or bus error.

***Note**: Where the secure master is also the PROT owner, writes from the secure master can effect the PROT state transitions allowed to either the PROT owner or the secure master.*

Table 28 PROT state transitions

PROT state transition	HW trigger	Response to PROT owner writes of STATE bit-field with SWEN=1_B	Response to secure master writes of STATE bit-field with SWEN=1_B	Response to other master agent function writes of STATE bit-field with SWEN=1_B
Any PROT state to Init	State transition on assertion of Application Reset	Alarm or Bus Error	Alarm or Bus Error	Alarm or Bus Error
Init to RunLock	State transition on assertion of init_done (through SCU/CSCU register)	State transition	Alarm or Bus Error	Alarm or Bus Error
Init to Run	N/A	State transition	Alarm or Bus Error	Alarm or Bus Error
Init to RunSec	N/A	State transition	Alarm or Bus Error	Alarm or Bus Error
Run to Config	N/A	State transition	Alarm or Bus Error	Alarm or Bus Error
Config to Run	N/A	State transition	Alarm or Bus Error	Alarm or Bus Error
Run to RunLock	N/A	State transition	Alarm or Bus Error	Alarm or Bus Error
RunSec to RunLock	N/A	State transition	Alarm or Bus Error	Alarm or Bus Error
RunSec to ConfigSec	N/A	State transition	Alarm or Bus Error	Alarm or Bus Error
ConfigSec to CheckSec	N/A	State transition	Alarm or Bus Error	Alarm or Bus Error
CheckSec to RunSec	N/A	Alarm or Bus Error	State transition	Alarm or Bus Error
CheckSec to ConfigSec	N/A	Alarm or Bus Error	State transition	Alarm or Bus Error
RunSec to Run	N/A	Alarm or Bus Error	State transition	Alarm or Bus Error

PROT register access rules

There are various conditions which have to be satisfied for a write access to a PROT register. A write which fails the required conditions below to update a PROT register will not update any of the bit-fields of the PROT register and will be terminated with an alarm or a bus error.

The following rules apply for accesses to the PROT register when ODEF is 0_B.

1. The PROT register can be read or written by any master agent function
2. A locked read-write sequence to the PROT register is not executed and terminates with an alarm or bus error
3. A write which represents an invalid PROT state transition is not executed and terminates with an alarm or bus error

The following rules apply for accesses to the PROT register when ODEF is 1_B.

1. The PROT register can be read by any master agent function
2. A locked read-write sequence to the PROT register is not executed and terminates with an alarm or bus error
3. A write to the PROT register from any master agent function other than the PROT owner or secure master is not executed and terminates with an alarm or bus error
4. Only the current PROT owner is allowed to modify the PROT owner fields. An attempt by any master agent function other than the PROT owner to update the PROT owner fields of the PROT register by

writing PROT.OWEN=1_B along with the other PROT owner fields is not executed and terminates with an alarm or bus error

5. A write by either the PROT owner or the secure master which represents an invalid PROT state transition is not executed and terminates with an alarm or bus error

***Note**: Certain functions only allow 32-bit accesses to their registers as indicated by the '32' in the access mode columns of the register overview tables in the respective functional block. Software shall follow this 32-bit access rule where a PROT register is marked as such.*

Protected registers access check

The following rules apply to accesses to the registers protected by a PROT mechanism.

- Registers protected by the PROT mechanism can be read by any master agent function, unless prevented by other access protection mechanisms like an APU
- The following PROT states are locked states: Run, RunLock, RunSec, CheckSec. When the PROT state machine is in a locked state, all write accesses to the registers protected by the PROT mechanism are not executed and terminate with an alarm or bus error
- The following PROT states are unlocked states: Init, Config, ConfigSec.
  - When the PROT state machine is in a unlocked state, only the PROT owner has write access to the registers protected by the PROT mechanism. Writes from disallowed master agent functions are not executed and terminate with an alarm or bus error
  - When the PROT state machine is in a unlocked state and no PROT owner is defined, all master agent functions have write access to the registers protected by the PROT mechanism, unless prevented by other access protection mechanisms like an APU

Related information

PROT state machine on page 66

PROT register

The PROT register allows software to define the PROT owner and provides the defined owner access to modify the PROT owner and state machine.

Each instance of the PROT mechanism may have implementation specific fields in the lower half of the PROT register. When such implementation specific fields are defined, they can only be written by the PROT owner when the PROT state is already unlocked or in the same write transaction that transitions the PROT state to unlocked (when the PROT state is Init, Config or ConfigSec or in the same write that changes the PROT state from Run to Config, RunSec to ConfigSec or CheckSec to ConfigSec). Writes from the PROT owner to the implementation specific fields when the PROT state is locked will be ignored and the implementation specific fields will not be updated. The PROT mechanism outputs the value of any implementation specific fields without any internal processing. See section "APU arrays to protect system memories" for an example of such an implementation specific field. Unused bits in this range are marked reserved, read-only zero.

After application reset, the PROT mechanism is in the Init state and ODEF is set to 0_B. This means that any master agent function can configure the PROT, as well as the protected registers immediately out of reset. Software will have to configure the PROT owner and switch to one of the Run states in order to secure the PROT mechanism.

***Note**: For functions related to cyber-security, the reset value of the PROT owner is instead be set to the CPUcs D TAG with ODEF=1_B.*

Related information

APU arrays to protect system memories on page 77

Resource Protection Register

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner and in some PROT states, the secure master to update the PROT state as described in the PROT mechanism.

PROT Offset address: 000_H

Resource Protection Register value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OWEN	ODEF	TAGID						PRSEN	PRS			VMEN	VM		
w	rw	rw						rw	rw			rw	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ImplementationSpecific								0				SWEN	STATE		
rw								r				w	rwh		

Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
ImplementationSpecific	15:8	rw	**Implementation Specific fields, reserved if none**
VM	18:16	rw	**Virtual Machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual Machine definition Enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection Set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.

(table continues...)

(continued)

Field	Bits	Type	Description
PRSEN	23	rw	**Protection Set definition Enable for PROT owner** 0_B **Disable**: PRS is not part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner
OWEN	31	w	**Owner write enable** 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	7:4	r	**Reserved** Read as 0; should be written with 0.

Table 29 Access mode restrictions of PROT sorted by descending priority

Mode name	Access mode		Description
write 1 to .**SWEN**	rwh	STATE	
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

Interfaces

Table 30 PROT interfaces

Signal	Direction	Description
clock	in	Input clock
application_reset	in	Reset input to PROT mechanism.
init_done	in	On the rising edge of init_done, all PROT state machines which are in the Init state are transitioned to the RunLock state.
Interconnect interface	in-out	Interconnect slave interface (for example SRI, FPI) which is used to access registers in the functional block.
owner	out	PROT owner information for use in external access gates.
config_lock	out	Indicates when active that the protected registers cannot be modified by application software.

(table continues...)

Table 30 (continued) PROT interfaces

Signal	Direction	Description
config_activate	out	Indicates when active that the modified (shadow) configuration should now be made active.
access_shadow	out	Indicates when active that read and write accesses should be targeted to the shadow state rather than the active state.

The PROT mechanism does not generate any interrupts. It generates either an alarm or a bus error for invalid writes to the PROT register or the protected resource registers.

Lock signals

The PROT states are decoded to generate the config_lock, and access_shadow output signals.

The config_lock signal is used to ensure that registers protected by the PROT mechanism can only be modified when the PROT is in an unlocked state (Init, Config or ConfigSec).

Table 31 PROT outputs

PROT state	config_lock	access_shadow
Init	0	1
Run	1	0
RunLock	1	0
RunSec	1	0
Config	0	1
ConfigSec	0	1
CheckSec	1	1

Shadow access signals

For protected registers which are shadowed, the shadow state of the register is copied into the active state whenever the config_activate signal is set. The config_activate signal is kept asserted when the PROT state is Init. Any transition in the PROT state from a configuration or check state to a run state (Config to Run, CheckSec to RunSec) also generates a single cycle pulse in the config_activate signal.

The copy of state from shadow to active has to be done such that no transaction on the interconnect ever sees a partially updated active state. This can be ensured by either copying all shadow state into the active state in a single cycle or by holding off new interconnect transactions while the shadow state is being copied into the active state.

For protected registers that are shadowed, the access_shadow signal determines whether read and write accesses are directed to the active or shadow state. The desired behavior is described in the following table.

Table 32 Access destination for shadowed protected registers

PROT state	Read accesses	Write accesses
Init	Active state	Active state
Run/RunLock/RunSec	Active state	Disabled
Config/ConfigSec	Shadow state	Shadow state
CheckSec	Shadow state	Disabled

This behavior is implemented using the following rules for protected registers that are shadowed:

- When access_shadow is 0_B, read and write accesses to the protected registers are steered to the active copy. Write accesses are disabled when config_lock is 1_B
- When access_shadow is 1_B, read and write accesses to the protected registers are steered to the shadow copy. Write accesses are disabled when config_lock is 1_B
  - ***Note**:* *Even though access_shadow is set in the Init state and this means that read and write accesses are steered to the shadow state in Init, the config_activate signal is set in Init and causes the shadow state to be copied over into the active state on every write. This means that the shadow and active states are kept consistent throughout Init and the net effect to software is the same as if the active state were being directly accessed.*

Related information

Protecting APU registers on page 74

Use cases

A few high level examples of PROT usage in the system are presented below.

Protecting shadowed registers

When configuration or control registers affect both the safety and security of the module, any modifications to the registers has to be done in a safe and secure manner. This is achieved by applying the following measures:

- The affected registers are shadowed so that they may be modified without the changes taking immediate effect
- They are protected by a PROT mechanism that is operated by software, see chapter "RunSec, ConfigSec and CheckSec states"

When software accesses shadowed registers such as those described above, the accesses are steered to either the active state or the shadow state as described in the table below.

Table 33 Access destination for shadowed protected registers

PROT state	Read accesses	Write accesses
Init	Active state	Active state
Run/RunLock/RunSec	Active state	Disabled
Config/ConfigSec	Shadow state	Shadow state
CheckSec	Shadow state	Disabled

Related information

RunSec, ConfigSec, CheckSec states on page 67

Protecting APU registers

A common example of a set of protected registers is the APU register set. These are usually protected by a PROT mechanism and can only be configured by the PROT owner (when the PROT owner is defined).

If the APU configuration registers are shadowed, the previous APU configuration continues to be active until any re-configuration of the APU is complete.

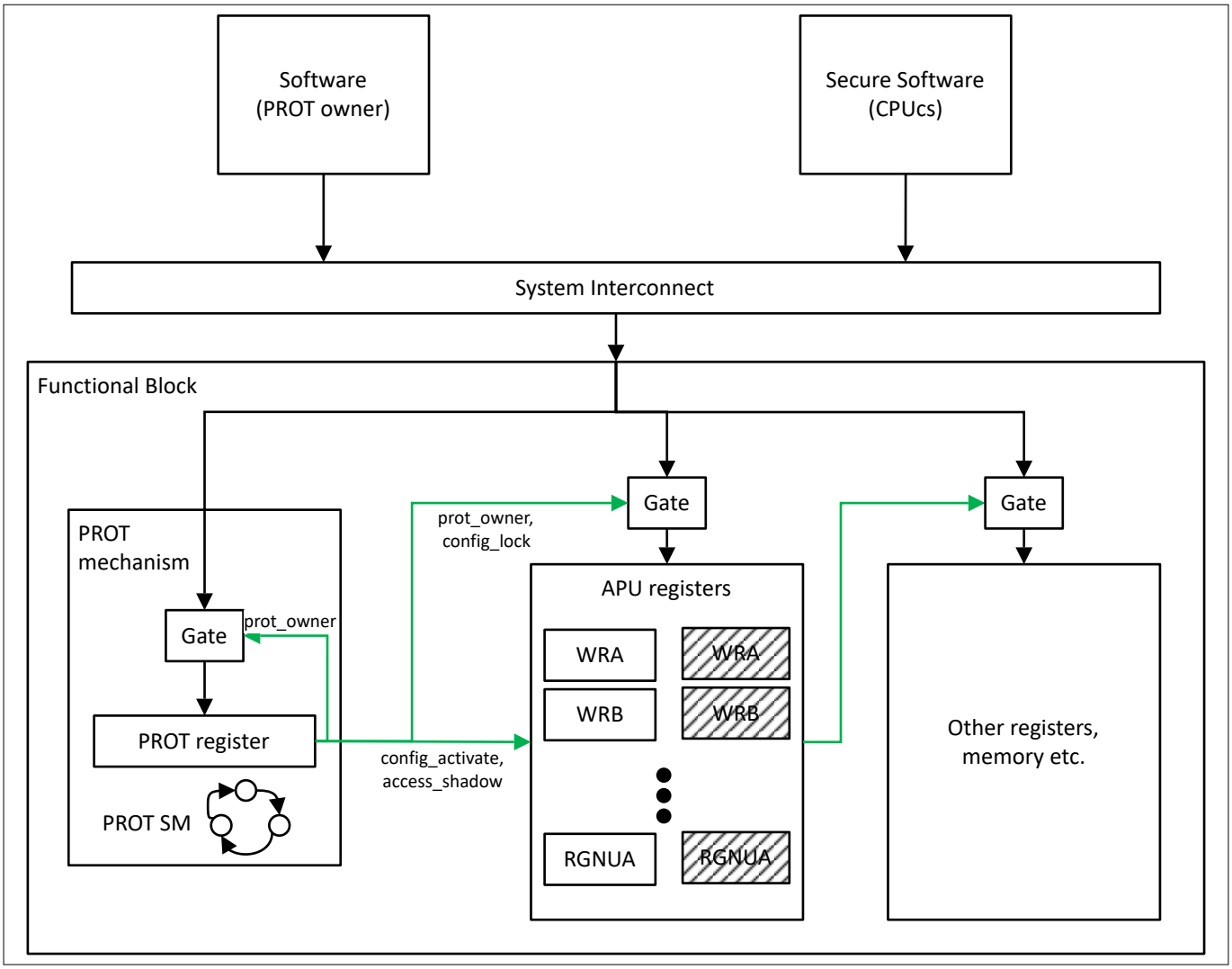


Figure 8 APU protection with shadowed state

Note that when a number of APUs together partition and protect a single function as is common in the system memories, it is not necessary that the state of all the APUs is shadowed. See chapter "APU arrays to protect system memories" for details.

Related information

APU arrays to protect system memories on page 77

Secure control of CSRM functions

There are several functions which are critical to the security of the MCU. The approach used to maximize security for these functions is that the reset state of related PROT and APU instances is set such that only the CPUcs has configuration access to these functions. After boot, the secure software can choose to open up access to these functions to other master agent functions as appropriate to the security of the application use case.

Reset state of security related PROT instances

- STATE = Init
- ODEF = 1
- TAGID = CPUCS.D

- VMEN = 0
- PRSEN = 0

Reset state of security related APU instances

- CPUCS TAG-IDs are enabled for read and write access
- All other TAG-IDs are disabled for read and write access
- All VM and PRS encodings are enabled for read and write access

Access protection unit (APU)

The access protection unit (APU) provides fine-grained control of the protection settings for a group of registers. Each APU can be set up to protect a set of registers or a memory region. The APU consists of a set of configurable registers (see "APU registers") and access checking logic which either allows or disallows each access from the bus interface to the intended target. Disallowed accesses are terminated with an alarm or bus error.

Related information

APU registers on page 78

Access protection check

The APU allows configuration software to enable or disable access to each master agent function in the MCU through the ACCEN_WRA/ACCEN_WRB and ACCEN_RDA/ACCEN_RDB registers. Each master agent function can be configured to have read-only access, write-only access, read and write access or can be disallowed access altogether

In addition, the APU supports virtualization by supporting access checks on the VM and PRS fields presented on the interconnect interface.

- When the interconnect interface presents a transaction with the VM validity indicator set, the access is only granted if the bit in ACCEN_VM corresponding to the VM number on the interconnect transaction is set. When the VM validity indicator on the transaction is not set (for example, when virtualization is disabled), the ACCEN_VM register is ignored
- When the interconnect interface presents a transaction with the PRS validity indicator set, the access is only granted if the bit in ACCEN_PRS corresponding to the PRS number on the interconnect transaction is set. When the PRS validity indicator on the transaction is not set, the ACCEN_PRS register is ignored

When the region registers (ACCEN_RGNLA, ACCEN_RGNUA) are implemented, they are used to restrict the APU instance to protect a subset of the memory region decoded by the slave.

An access operation (read or write) is executed if all of the following conditions are true:

- The bit corresponding to the master agent function TAG ID in the interconnect transaction is set in either the ACCEN_RDA/ACCEN_RDB or ACCEN_WRA/ACCEN_WRB register for read or write accesses respectively
- Either the VM validity indicator is not set in the interconnect transaction or the bit corresponding to the VM ID in the interconnect transaction is set in the ACCEN_VM register for read or write accesses respectively

- Either the PRS validity indicator is not set in the interconnect transaction or the bit corresponding to the PRS in the interconnect transaction is set in the ACCEN_PRS register for read or write accesses respectively
- Either the APU does not implement the ACCEN_RGNLA and ACCEN_RGNUA registers or the masked address in the interconnect transaction is such that $\text{ACCEN_RGNLA} \leq \text{masked_address} < \text{ACCEN_RGNUA}$. The masked address is defined as follows:
  - For addresses in segments 0-7 and 12-15, the masked address is the same as the address in the interconnect transaction
  - For addresses in segments 8-11, the masked address is the address in the interconnect transaction bitwise-ANDed with DFFFFFFF_H

***Note**: The ACCEN_RGNLA and ACCEN_RGNUA registers are usually only implemented in APUs protecting system memories to allow application software to partition the memory into smaller regions with different access protection rules.*

APU arrays to protect system memories

System memories such as DSPRs, PSPRs, DLMUs, LMUs, and PFLASHs are used to store data for many different software tasks and may therefore have to be partitioned into regions that can each have a different access protection configurations. This is implemented using an array of APU instances for each memory, with the ability to specify the region that each APU instance protects within that memory.

The region for each APU is configured by writing the lower and upper addresses of the desired region into the ACCEN_RGNLA and ACCEN_RGNUA registers respectively.

The APU configuration therefore is critical in partitioning the memory between different software tasks of various safety and security levels. The safety and security of the APU configuration is ensured through the following scheme:

- The array of APU registers is not directly writable by software; they are instead read-only
- There is a single additional shadow APU register set which is writable by software and protected by a PROT mechanism. The PROT register contains an implementation specific SEL field which allows the PROT owner to specify the target APU instance to be configured
- When the PROT owner wishes to modify the configuration of a certain target APU instance:
  - If the PROT state is currently Init:
    - The PROT owner writes the target APU instance number to the implementation specific SEL field of the PROT register
    - The PROT owner then writes the shadow APU's registers with the complete new configuration data
    - Since the PROT state is Init, this immediately updates the active state of the target APU instance
  - When the PROT state is not Init:
    - The PROT owner writes to the PROT register to unlock the configuration and writes the target APU instance number into the implementation specific SEL field in the PROT register (either in the same write or after the PROT state has been unlocked)
    - The PROT owner then writes the shadow APU's registers with the complete new configuration data
    - If some portion of the memory contains security related information, this configuration can then be checked by security software for security issues as described in the PROT mechanism chapter
    - When the PROT mechanism next transitions to a locked state, the new configuration is copied from the shadow registers into the target APU instances registers and becomes active

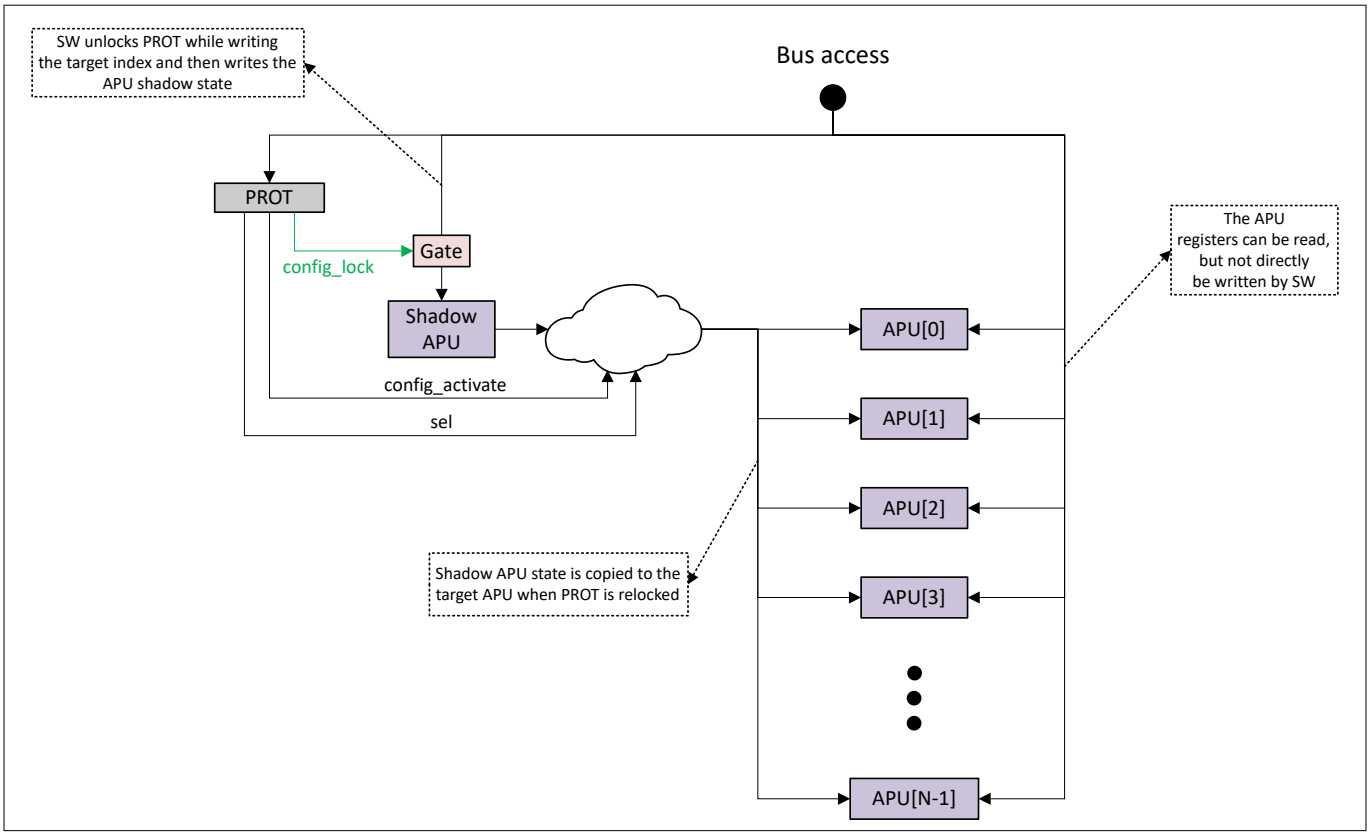


Figure 9 APU array configuration

Related information

PROT mechanism on page 64

APU registers

The APU registers are laid out as follows in a contiguous range of addresses.

The ACCEN_WRA, ACCEN_WRB, ACCEN_RDA, ACCEN_RDB, ACCEN_VM and ACCEN_PRS registers are mandatory for every instantiation of the APU. The ACCEN_RGNLA and ACCEN_RGNUA registers are optionally present in APUs protecting system memories and allow the APU instance to protect a configurable subset of the memory region. When the ACCEN_RGNLA and ACCEN_RGNUA registers are not present, they shall be marked reserved and no other registers shall be overlaid on them.

The reset state of the APU is set to enable all master agent functions for read access and enable only CPU0 and Cerberus for write access. For security related functions like the CPUcs APU the reset value is instead set to allow only the CPUcs with read and write access.

Table 34 Registers overview - ACCEN (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ACCEN_WRA	Write access enable register A	000_H	U	SV, SE		79
ACCEN_WRB	Write access enable register B	004_H	U	SV, SE		79

(table continues...)

Table 34 (continued) Registers overview - ACCEN (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ACCEN_RDA	Read access enable register A	008_H	U	SV, SE		80
ACCEN_RDB	Read access enable register B	$00C_H$	U	SV, SE		80
ACCEN_VM	VM access enable register	010_H	U	SV, SE		81
ACCEN_PRS	PRS access enable register	014_H	U	SV, SE		82
ACCEN_RGNLA	Region lower address register	018_H	U	SV, SE		82
ACCEN_RGNUA	Region upper address register	$01C_H$	U	SV, SE		83

Write access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

ACCEN_WRA Offset address: 000_H

Write access enable register A value: $1000\ 0003_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access

Write access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

ACCEN_WRB Offset address: 004_H

Write access enable register B value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN63	EN62	EN61	EN60	EN59	EN58	EN57	EN56	EN55	EN54	EN53	EN52	EN51	EN50	EN49	EN48
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN47	EN46	EN45	EN44	EN43	EN42	EN41	EN40	EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-63)	q-32	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access

Read access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

ACCEN_RDA Offset address: 008_H

Read access enable register A value: FFFF $FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access

Read access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

ACCEN_RDB Offset address: $00C_H$

Read access enable register B value: FFFF $FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN63	EN62	EN61	EN60	EN59	EN58	EN57	EN56	EN55	EN54	EN53	EN52	EN51	EN50	EN49	EN48
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN47	EN46	EN45	EN44	EN43	EN42	EN41	EN40	EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-63)	q-32	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access

VM access enable register

This register defines which virtual machine encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each VMx.

ACCEN_VM Offset address: 010_H

VM access enable register value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for VM ID q** This bit enables read access to the access protected region for transactions with the VM ID q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for VM ID q** This bit enables write access to the access protected region for transactions with the VM ID q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

PRS access enable register

This register defines which protection register set encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each PRSx.

ACCEN_PRS Offset address: 014_H

PRS access enable register value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for PRS q** This bit enables read access to the access protected region for transactions with the PRS q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for PRS q** This bit enables write access to the access protected region for transactions with the PRS q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

Region lower address register

This register sets the lower bound of the access protected region. Addresses greater than or equal to the lower bound are considered part of the access protected region.

ACCEN_RGNLA Offset address: 018_H

Region lower address register value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADDR															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR										0					
rw										r					

Field	Bits	Type	Description
ADDR	31:6	rw	**Bits 31:6 of the lower bound of the access protected region**
0	5:0	r	**Reserved** Read as 0; should be written with 0.

Region upper address register

This register sets the upper bound of the access protected region. Addresses stricly less than the upper bound are considered part of the access protected region.

ACCEN_RGNUA Offset address: $01C_H$

Region upper address register value: $FFFF\ FFC0_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADDR															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR										0					
rw										r					

Field	Bits	Type	Description
ADDR	31:6	rw	**Bits 31:6 of the upper bound of the access protected region**
0	5:0	r	**Reserved** Read as 0; should be written with 0.

Access protection of unused resources

When software does not intend to use a resource, it can choose not to configure the resource at all and leave any PROT and APU mechanisms in the resource in their reset states.

After all configuration by application startup software (ASSW) is complete, the ASSW writes to the SMM_PROTINIT.PINIT_DONE or CSCU_INIT.DONE register fields which causes all PROT mechanisms in that domain which are still in their Init states to transition to the RunLock state. No further modification of protected configuration is possible in the RunLock state until the next application reset. For unused modules which default to having their clocks turned off at reset, this disables the module entirely.

This mechanism thereby removes all access to any slave functions that are left unconfigured by software, effectively putting them in a safe and secure state.

Register access modes

Read and write access to registers and memory locations are sometimes . The terms defined in thetable bel ow may be used in memory and register access tables to indicate such restrictions on access. Eachfunction al block also defines additional terms in its access mode glossary table to indicate module specificrestrictio ns.

In general, if an access type is not permitted under these rules (for example, an attempted write to a read-only register or an attempted user mode access to a SV protected register) then a Bus Error will result, unless the access is also marked as nBE or otherwise stated in the specific module chapter. Where there are multiple access terms specified for a single register in a comma separated list, the access is only permitted if all of the conditions are satisfied when the access is executed.

Bit-fields access restrictions, where for example a write access is permitted or not depending on value of another bit-field in the same register, are described in separate "Access mode restrictions" tables for each register and its bit-fields. When violated, these bit-field restrictions do not generate a Bus Error unless specified differently.

Table 35 Access terms

Symbol	Description
BE	Always returns a Bus Error. This is used for example on a write access to indicate a read-only register.
nBE	Indicates that no Bus Error is generated when accessing this address range, even though it is either an access to an undefined address or the access does not follow the given rules.
nE	Indicates that no Error is generated when accessing this address or address range, even though the access is to an undefined address or address range. This is used for example on CPU accesses (MTCR or MFCR) to undefined addresses in the CSFR range.
TM	Access only when in test mode.
M	Indicates a module specific access condition. Refer to the register description for details of the specific access condition.
OEN	Access only when OCDS is enabled.
PW	Access only when the correct password is given.
STP	Access only during start-up, before start-up protection is set (SMM_STCON.ST_PROT=1).
CSSTP	Access only during secure start-up, before secure start-up protection is set (CSCU_STCON.CSSTP=1).
SV	Access only when supervisor mode is active on the interconnect.
PROT	Access restrictions as defined in the PROT register access rules.
16	Access when using 16-bit width or higher.
32	Access only when using 32-bit width.
64	Access only when using 64-bit width.
U	No access restrictions.

3.2 Software Over The Air (SOTA)

Software updates Over The Air (SOTA) is supported by providing the ability to split the PFLASH into two regions. The SOTA functionality enables software executing out of the application PFLASH (PFLASH other than the PFLASHcs) to be updated without impacting the currently executing application.

The application could be executing out of the active PFLASH region, while an update to the application code is downloaded and programmed into the inactive PFLASH region.

The SOTA functionality on the PFLASHcs is more in that a programming operation to the inactiveregion of the PFLASHcs cannot happen while the active region of the PFLASHcs is being accessed.

3.2.1 Feature list

- Ability to program the inactive application PFLASH region while retaining full read access to code and data in the active application PFLASH region
- Ability to swap the active PFLASH region by updating the SOTA swap configuration UCB. The updated configuration becomes active on the next system reset
- Ability to disable the SOTA mechanism and access all of the PFLASH in the MCU with a linear address map

3.2.2 Functional overview

The SOTA functionality can be enabled and configured independently for the application PFLASH and PFLASHcs through configuration in the SOTA UCBs. When SOTA is disabled, the MCU addresses the respective application PFLASH or PFLASHcs in the linear addressing mode. When SOTA is enabled, the MCU splits the respective application PFLASH or PFLASHcs into two address regions, one of which can be set as the active region as described below. See the system address map chapter for details of the PFLASH memory address ranges for each SOTA configuration.

Application PFLASH linear addressing mode

When SOTA is disabled for the application PFLASH, all the application PFLASH memory is available for instruction fetch and to read in a single contiguous memory region.

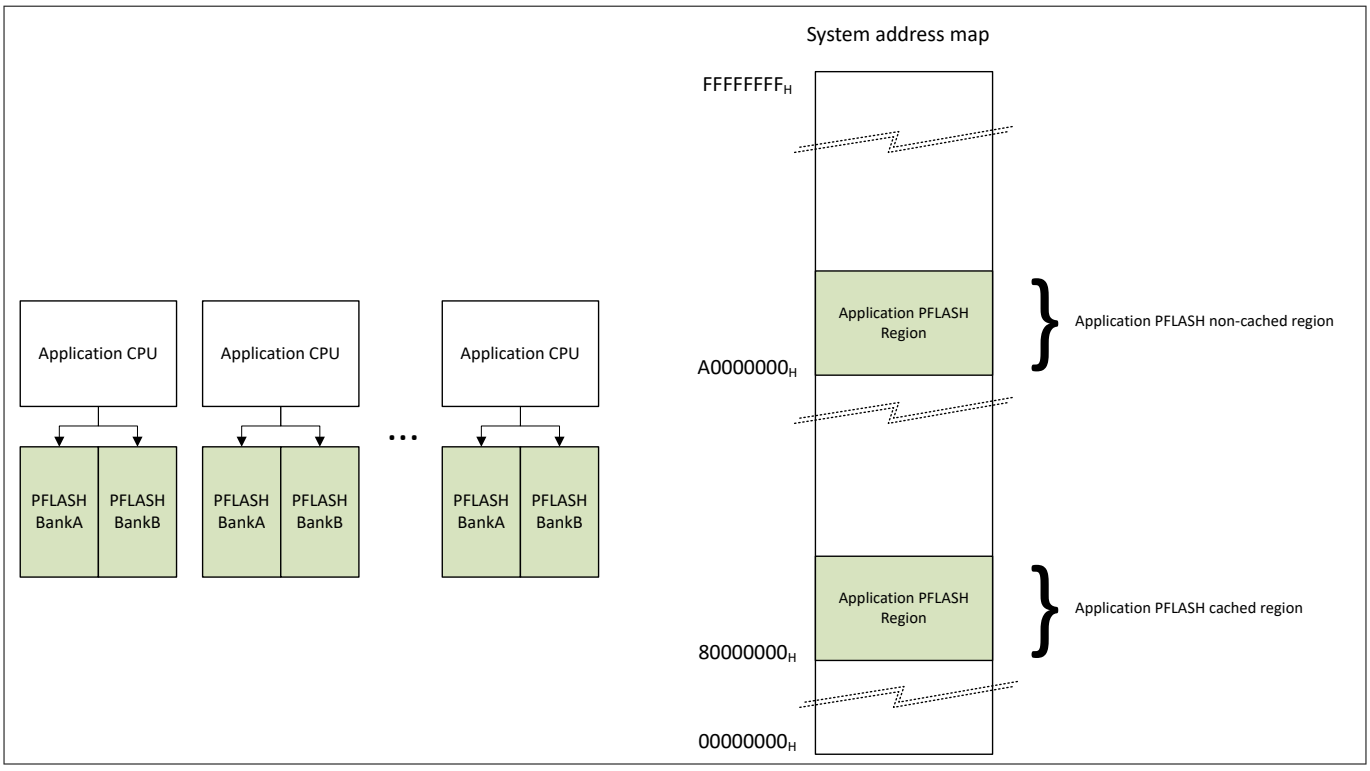


Figure 10 System address map with application PFLASH in linear addressing mode

Application PFLASH SOTA addressing mode

When SOTA is enabled for the application PFLASH, the application PFLASH is split into two equal regions.

- Application PFLASH region A: The memory available in one PFLASH bank of each of the application CPUs is available for instruction fetch and to read in a single contiguous memory region
- Application PFLASH region B: The memory available in the other PFLASH bank of each of the application CPUs is available for instruction fetch and to read in a single contiguous memory region

One of the application PFLASH regions A or B can be marked as an active region and this region is available for application software to read or execute from. The other region is marked as an inactive region and is available for programming operations such as to write a new software image. The programming operations to the inactive region can happen without any effect on instruction fetch or read accesses to the active region.

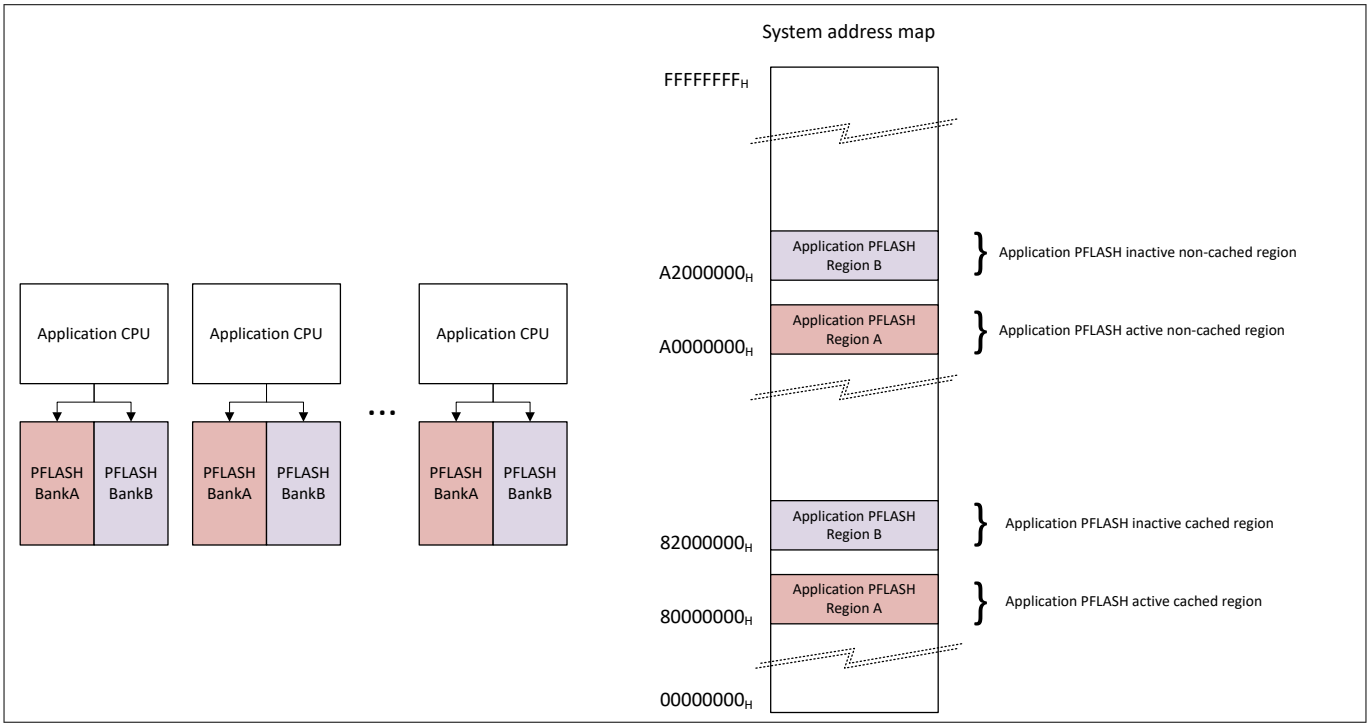


Figure 11 System address map with application PFLASH in SOTA-A mode

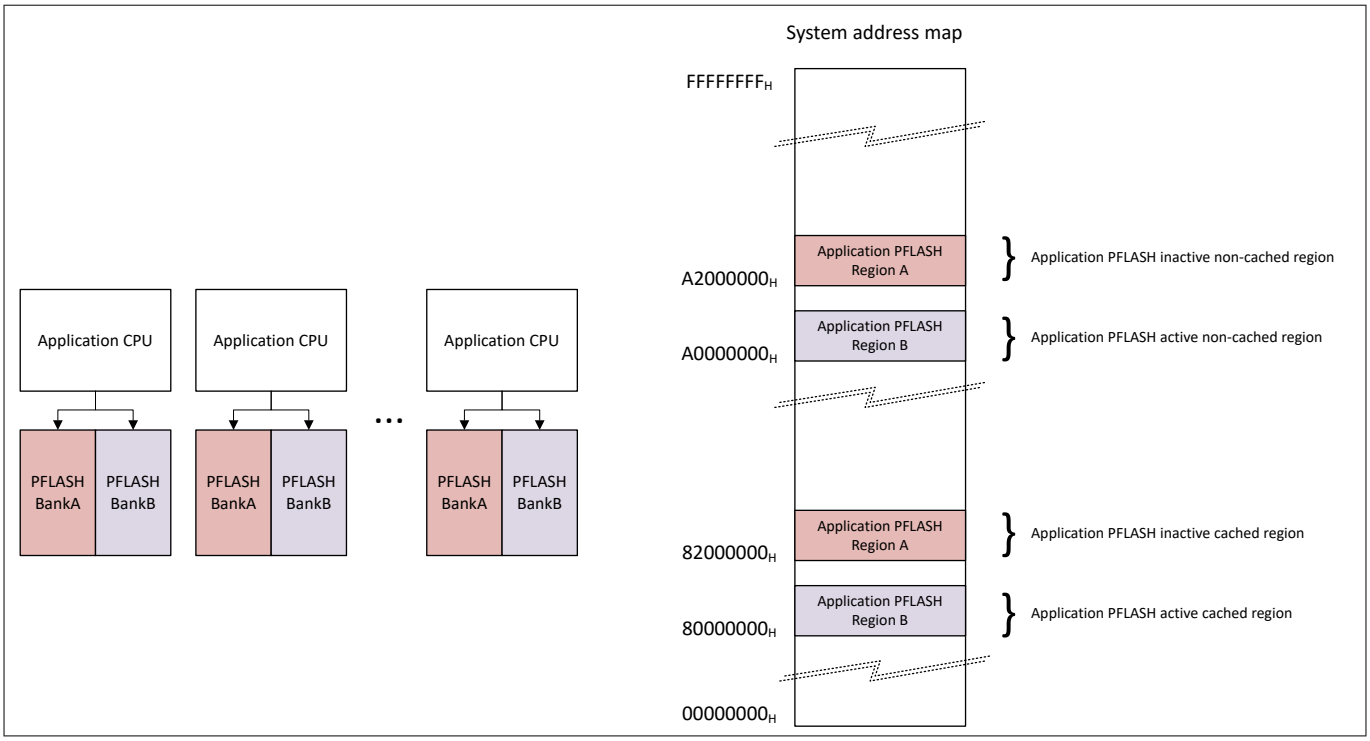


Figure 12 System address map with application PFLASH in SOTA-B mode

When SOTA is enabled for the application PFLASH, the active and inactive regions can be swapped by having application software update the SOTA swap configuration in the UCB. When a system reset is next applied, the MCU will boot up with the new SOTA swap configuration.

PFLASHcs linear addressing mode

When SOTA is disabled for the PFLASHcs, the PFLASHcs memory is available for instruction fetch and to read in a single contiguous memory region.

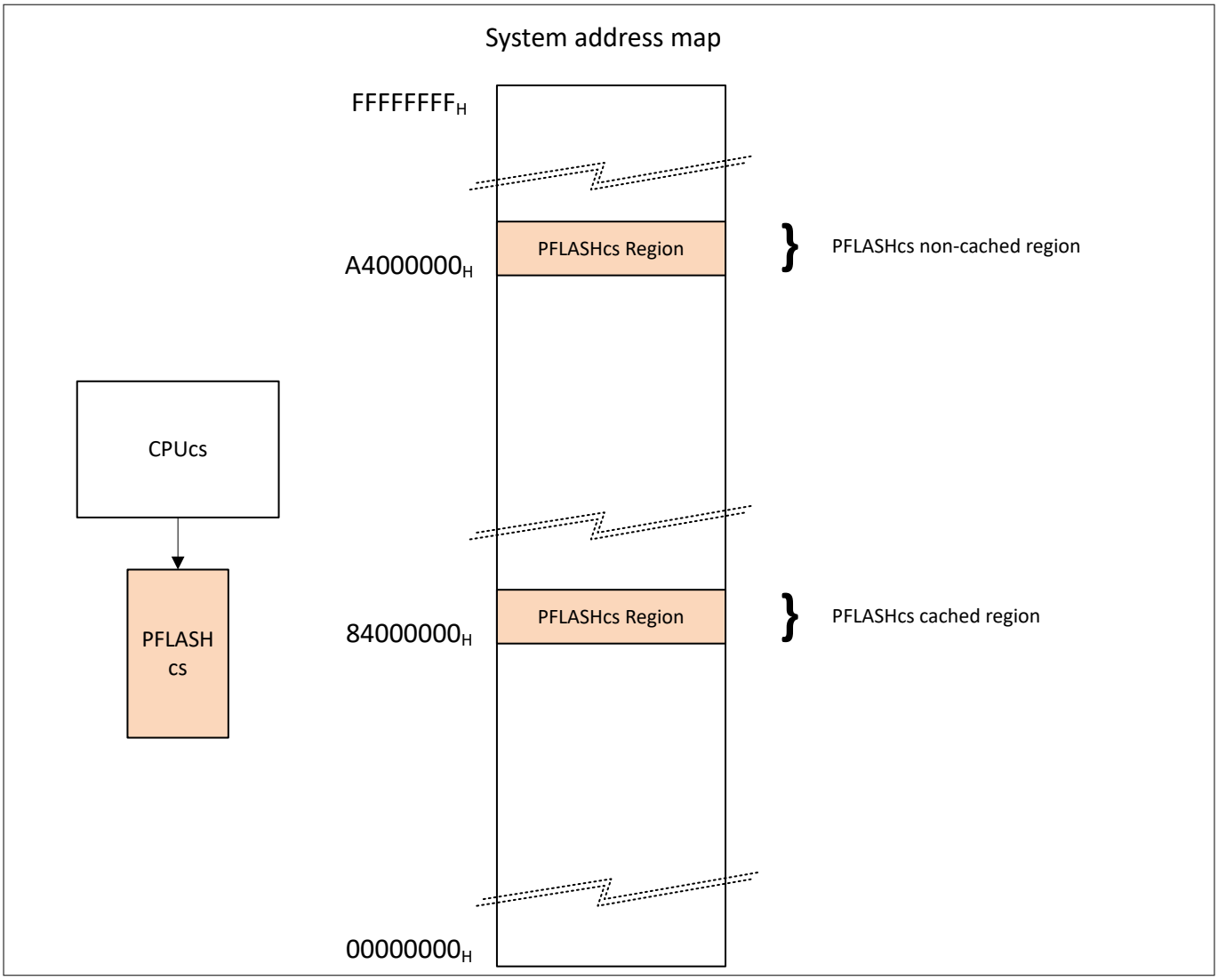


Figure 13 System address map with PFLASHcs in linear addressing mode

PFLASHcs SOTA addressing mode

When SOTA is enabled for the PFLASHcs, the PFLASHcs is split into two equal regions:

- PFLASHcs region A: One half of the PFLASHcs is denoted region A and is available for instruction fetch and to read in a single contiguous memory region
- PFLASHcs region B: One half of the PFLASHcs is denoted region B and is available for instruction fetch and to read in a single contiguous memory region

One of the PFLASHcs regions A or B can be marked as an active region and this region is available for application software to read or execute from. The other region is marked as an inactive region and is available for programming operations such as to write a new software image. Since both PFLASHcs regions physically map to a single PFLASH bank, programming accesses to the inactive region cannot happen at the same time as instruction fetch or read accesses to the active region.

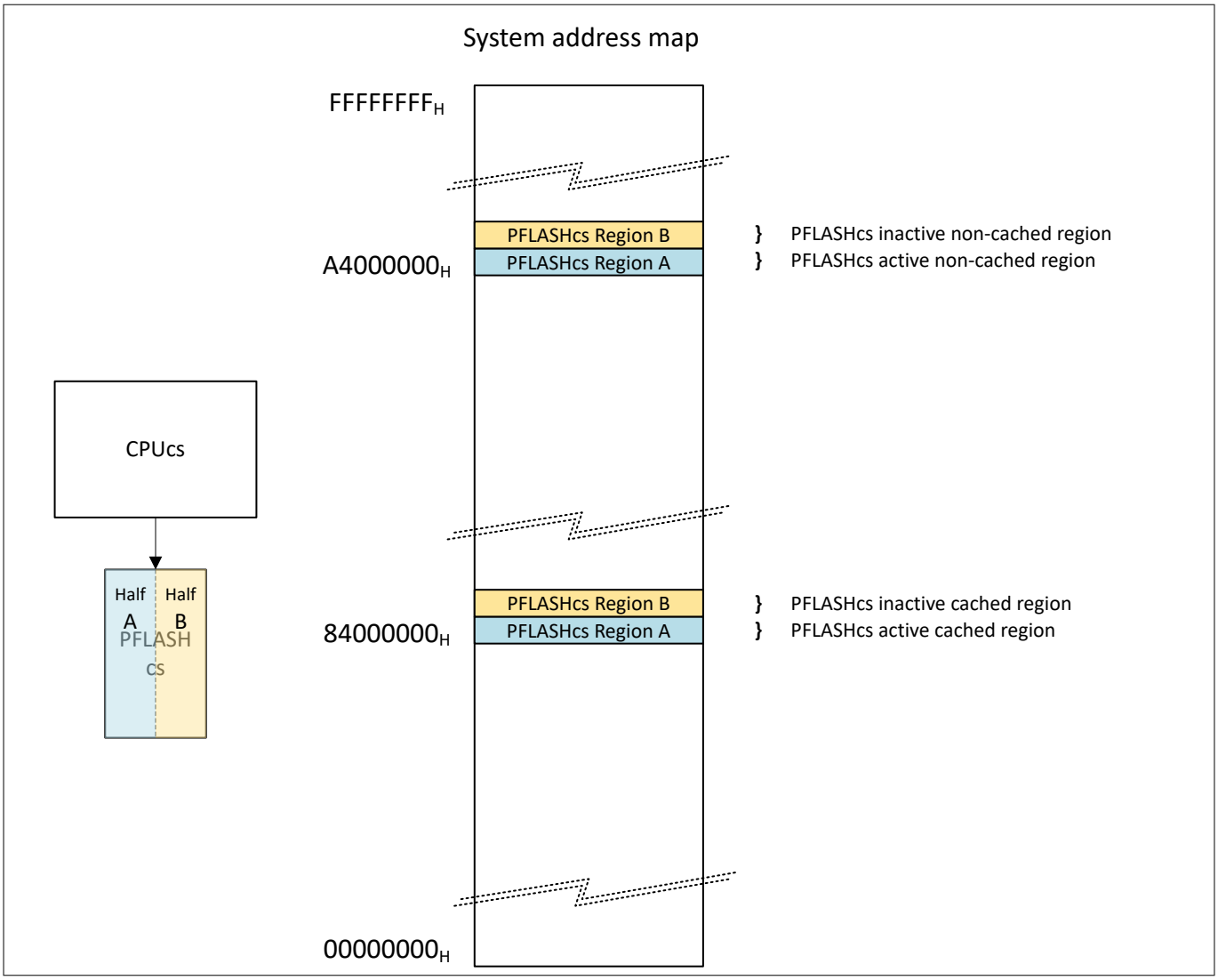


Figure 14 System address map with PFLASHcs in SOTA-A mode

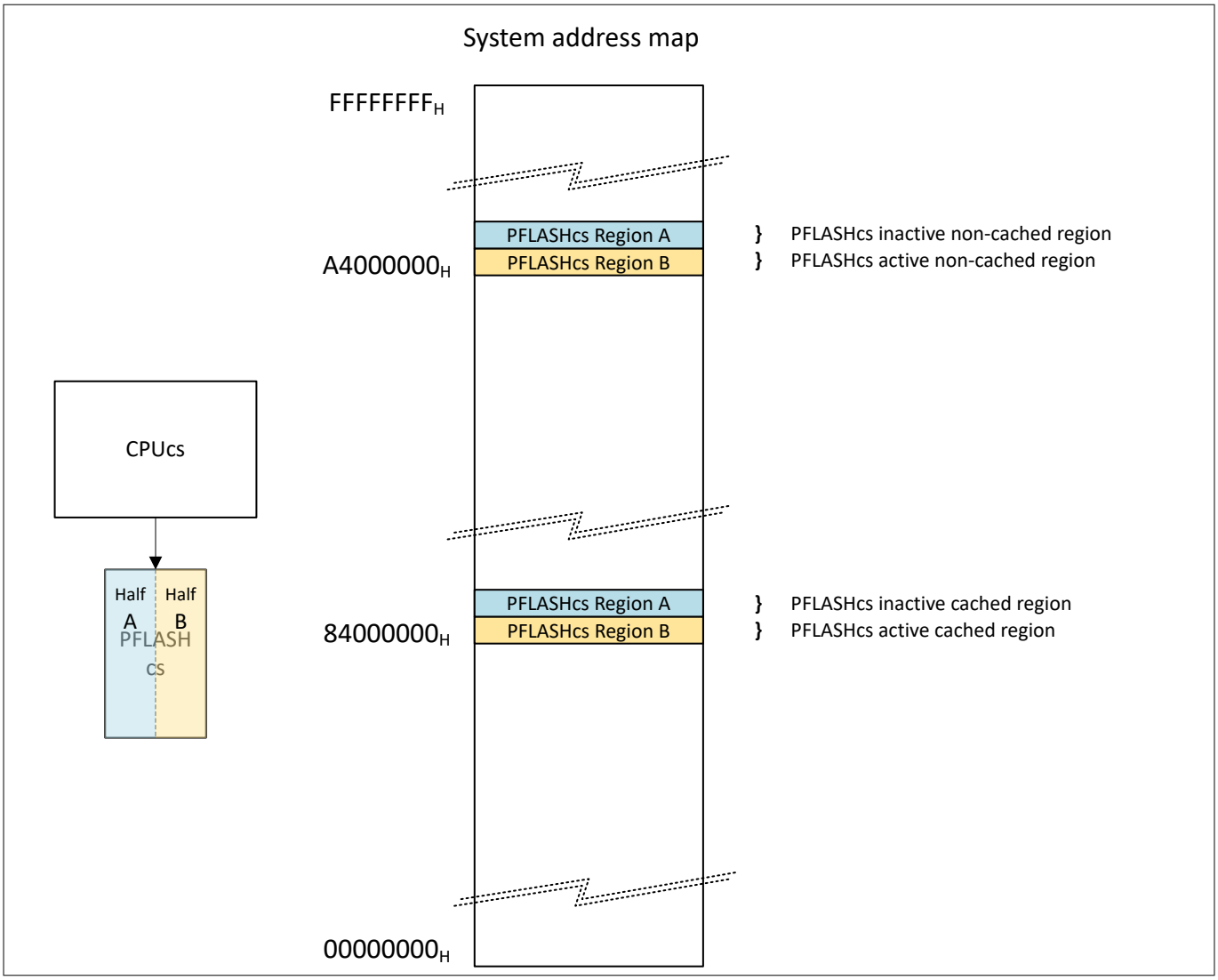


Figure 15 System address map with PFLASHcs in SOTA-B mode

When SOTA is enabled for the PFLASHcs, the active and inactive regions can be swapped by having secure software update the SOTA swap configuration in the UCB. When a system reset is next applied, the MCU will boot up with the new SOTA swap configuration.

3.2.3 Functional description

When SOTA is enabled, the active PFLASH region can be read or executed from, while programming or verification operations can be performed on the inactive PFLASH region. When a SOTA update has completed and the banks are swapped around, only the address mapping between the physical PFLASH modules and the system address map will change. This means that no data needs to be copied and the system address ranges being executed from are always the same.

***Note**: All NVM operations are performed using the physical system address of the PFLASH, so an NVM operation always uses the linear system address map regardless of SOTA or swap settings. 'NVM operation' is a term used for any command sequence such as a program or erase targeting a PFLASH and does not include reads.*

The parameters that control SOTA address map switching and related functions are configured in UCB and the hardware configuration is only updated during the subsequent system reset. This prevents unintentional changes during application execution.

Enabling SOTA on AURIX™ TC4xx devices takes effect with no loss of usable PFLASH memory apart from the partitioning of the memory into active and inactive PFLASH regions.

3.2.3.1 Performance considerations

The CPU access to its local Program Flash bank is optimized for maximum performance. In the AURIX™ TC4xx SOTA can be enabled with no impact on the performance of the application CPU access to the local PFLASH bank. This is achieved by providing each application CPU with two local PFLASH banks and simply switching between the two when a SOTA swap is requested.

3.2.3.2 Configuring SOTA

This section describes the configuration required to enable SOTA and swap between the active and inactive regions.

3.2.3.2.1 Configuration parameters

The SOTA functionality can be enabled and configured independently for the application PFLASH and PFLASHcs. The table below shows the UCB parameters which configure whether SOTA is enabled and which region is set to the active region.

Table 36 SOTA configuration parameters

Affected PFLASH	Parameter	Description	Configuration UCB
Application PFLASH	SOTA mode enable	If SWAPEN is set to enable, SOTA mode will be entered after the next system reset. Linear mode will be enabled otherwise.	UCB_RTC_USERCFG_ORIG.SWAP_ena, UCB_RTC_USERCFG_COPY.SWAP_ena
	Bank swap configuration	If SOTA mode is valid and enabled and the SWAP information configured in the UCB is valid, then after the next system reset the configured region A or B is made active.	UCB_RTC_SWAP_ORIG, UCB_RTC_SWAP_COPY
PFLASHcs	SOTA mode enable	If SWAPEN is set to enable, SOTA mode will be entered after the next system reset. Linear mode will be enabled otherwise.	UCB_CS_USERCFG_ORIG.SWAPCS_ena, UCB_CS_USERCFG_COPY.SWAPCS_ena
	Bank swap configuration	If SOTA mode is valid and enabled and the SWAP information configured in the UCB is valid, then after the next system reset the configured region A or B is made active.	UCB_CS_SWAP_ORIG, UCB_CS_SWAP_COPY

3.2.3.2.2 Initial device configuration for SOTA

There are some recommendations for installing the initial device configuration for a device which has SOTA enabled.

Starting from the delivery state, the initial execution image is programmed onto the active PFLASH region. If SOTA mode on PFLASHcs is required, the initial execution image should include the CSRM program code in the active PFLASHcs region. It is recommended that the sectors used are then protected by installing sector specific write protection in the UCB using the ORIG and COPY programming sequence as described in the 'UCB handling' section of the NVM chapter.

The start address of the initial image is programmed in UCB_RTC_BMHD and UCB_CS_BMHD and the standard programming of the Boot Mode Header UCBs is performed.

In order to select SOTA-A mode on the application PFLASH (PFLASH region A is active, PFLASH region B is inactive), the following values should be programmed into the UCB entry:

- Write the value 55_H into UCB_RTC_SWAP.SWAP_ORIG_MARKER0.MARKERx
- Write the system address of UCB_RTC_SWAP.SWAP_ORIG_SAL0 into UCB_RTC_SWAP.SWAP_ORIG_SAL0.SALx
- Write the confirmation code 43211234_H into UCB_RTC_SWAP.SWAP_ORIG_STATUS0.STATUSx
- Write the combined CRC32 value of SAL0, STATUS0, MARKER0 into UCB_RTC_SWAP.SWAP_ORIG_CRCSE0.CRCSEx
- Finally, UCB_RTC_USERCFG.SWAP_ena.SWAPEN is set to enabled, which enables SOTA mode for the application PFLASH with the next system reset

In order to select SOTA-A mode on the PFLASHcs (PFLASH region A is active, PFLASH region B is inactive), the following values should be programmed into the UCB entry:

- Write the value 44_H into UCB_CS_SWAP.SWAP_ORIG_MARKER0.MARKERx
- Write the system address of UCB_CS_SWAP.SWAP_ORIG_SAL0 into UCB_CS_SWAP.SWAP_ORIG_SAL0.SALx
- Write the confirmation code 43211234_H into UCB_CS_SWAP.SWAP_ORIG_STATUS0.STATUSx
- Write the combined CRC32 value of SAL0, STATUS0, MARKER0 into UCB_CS_SWAP.SWAP_ORIG_CRCSE0.CRCSEx
- Finally, UCB_CS_USERCFG.SWAP_ena.SWAPEN is set to enabled, which enables SOTA mode for the PFLASHcs with the next system reset

UCB_RTC_FLASH and UCB_RTC_PFLASH_OTPx are then programed with the necessary values to setup the required OTP and WOP settings on the application PFLASH. Similarly, UCB_CS_FLASH and UCB_CS_PFLASH_OTPx are programmed with the necessary values to setup the required OTP and WOP settings on the PFLASHcs bank.

***Note**: Any OTP or WOP protected sectors cannot be re-programmed with a new image.*

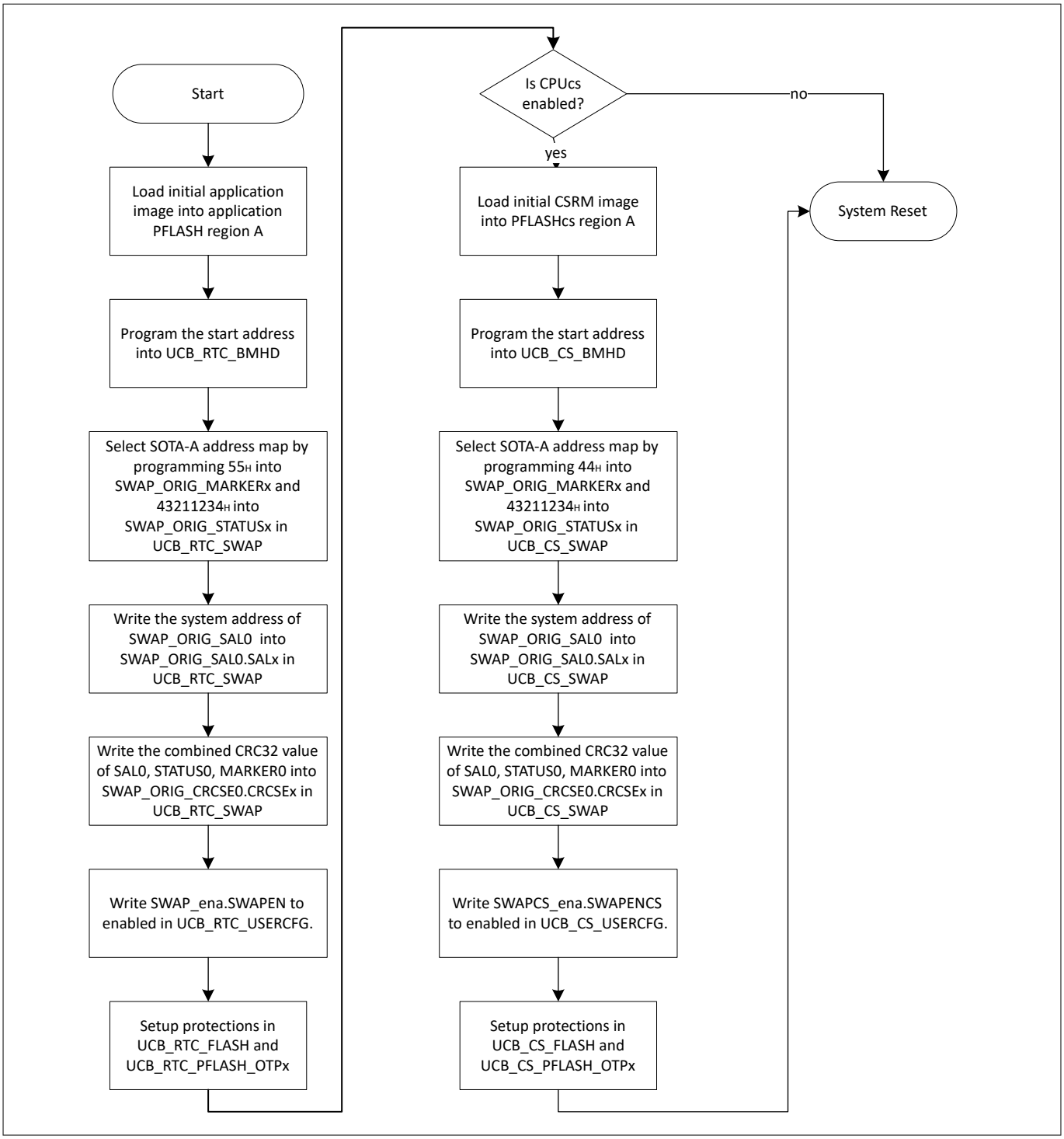


Figure 16 Initial SOTA configuration

3.2.3.2.3 Runtime SOTA configuration

Here we detail the recommendations for installing the new image during a running application and configuring the device to swap to the new image.

In order to swap to the new image, the new program image needs to be loaded into the inactive PFLASH region. In order to do this, the sector specific write protection must be disabled for those banks by presenting the UCB_RTC_PFLASH or UCB_CS_PFLASH password for the ‘Disable Protection’ DMU command sequence in the

NVM. The master performing the update must also be enabled with write access to the inactive PFLASH region in the respective APU in the NVM.

Since concurrent NVM operations (like a program or erase) to PFLASH and DFLASH are not supported, PFLASH operations must either be scheduled for times when no DFLASH operation is taking place or any ongoing DFLASH operation must be suspended to allow the PFLASH operation to occur. Therefore there needs to be some synchronization between the EEPROM driver running in the application and the Secure Flash Bootloader which is performing the update. An alternative is to allocate all PFLASH banks to the CSRM command interface in the CSRM_ALL_PFLASH mode and have the CSRM perform all PFLASH programming while the host can continue accessing the host DFLASH through the host command interface.

The newly written image must have any errors identified and corrected before write protection is re-enabled as described in the safety manual.

In the case of a hard failure during SOTA re-programming or erasing of PFLASH, the Replace Logical Sector feature can be used (See NVM chapter for more details). This feature allows you to map a failing logical sector to a redundant sector using the ‘Replace Logical Sector’ DMU command sequence.

The next step is to configure the SWAP information by configuring the address map to be selected in the UCB_RTC_SWAP or UCB_CS_SWAP. After disabling the sector specific write protection as described above, UCB_RTC_SWAP.SWAP_ORIG_MARKERx is changed to either:

- 55_H, selecting the SOTA-A address map, if application PFLASH region A contains the new image
- AA_H, selecting the SOTA-B address map, if application PFLASH region B contains the new image

Similarly, after disabling the CS sector specific write protection as described above, UCB_CS_SWAP.SWAP_ORIG_MARKERx is changed to either:

- 44_H, selecting the SOTA-A address map, if PFLASHcs region A contains the new image
- Either 11_H or 22_H, selecting the SOTA-B address map, if PFLASHcs region B contains the new image. The value programmed depends on the size of the PFLASHcs bank in the device being programmed

The other fields in the UCB_RTC_SWAP or UCB_CS_SWAP entry - SWAP_ORIG_SALx.SALx, SWAP_ORIG_STATUSx.STATUSx and SWAP_ORIG_CRCSEx.CRCSEx are then programmed as with the initial configuration. The previous (x-1) UCB_RTC_SWAP or UCB_CS_SWAP entry is invalidated by over-programming all-1 into SWAP_ORIG_STATUS(x-1) and SWAP_ORIG_SAL(x-1) (over-programming with all-1 delivers an ECC correct result for UCB and DFLASH). For all of these, 'x' should be increased by one each time images are swapped, starting from 1 the first time after the initial configuration.

If UCB_RTC_SWAP or UCB_CS_SWAP is full (that is 'x' has reached 31), the whole UCB may be erased and 'x' set back to 0 before a new entry is added. The write protection is then re-enabled by using the ‘Resume Protection’ DMU command sequence and updating the master access protection in the relevant APU to detect and prevent accidental writes.

***Note**: The last valid entry of the SWAP information (that is SWAP information stored at the highest value of ‘x’) is used by the Startup Software for configuring SOTA in the system.*

It is possible to configure 32 SWAPs in UCB_RTC_SWAP or UCB_CS_SWAP before an erase is required. The maximum number of SWAPs possible during the lifetime of the device is dependent on the datasheet parameter for PFLASH erase or program cycles. To perform 1000 application PFLASH SWAP configurations, a total of at least 62 UCB erase or program cycles are required during the lifetime (two UCB erases are required per 32 SWAP updates as both UCB_RTC_SWAP_ORIG and UCB_RTC_SWAP_COPY need to be erased and updated). This needs to be taken into account while updating the other UCBs in order to adhere to the datasheet parameter for UCB program or erase cycles.

To begin running with the new image, a system reset needs to be triggered (application resets have no effect).

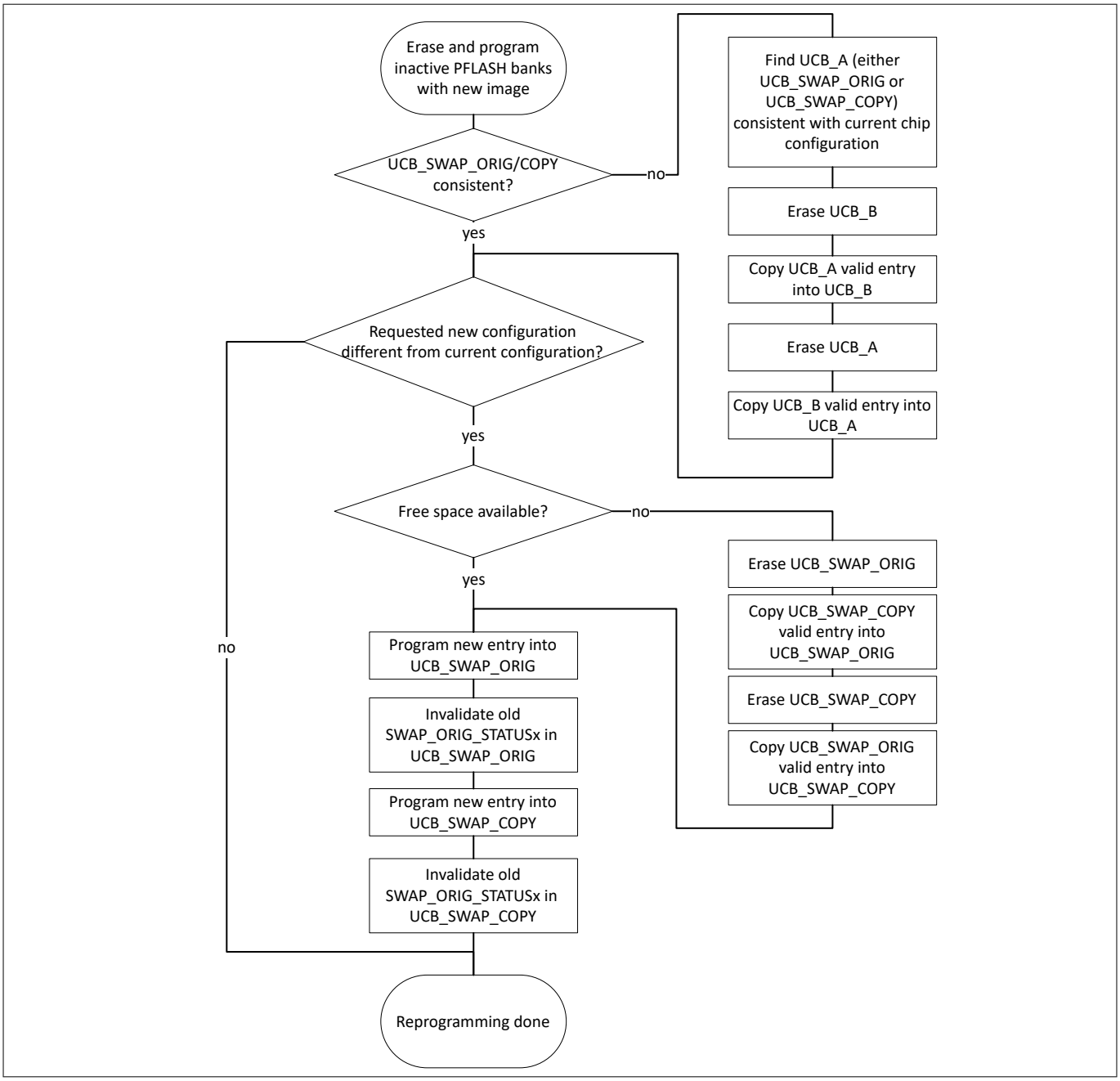


Figure 17 Runtime SWAP configuration

3.2.3.3 Safety

If SOTA is disabled, the entire PFLASH region should be protected by appropriate APU configuration in the DMU which prevents unintentional changes to the contents. If SOTA is enabled, APU protection for the inactive PFLASH region should be configured to allow them to be updated by enabled software masters.

However, even for the active PFLASH region, the requirement for safety application software to be checked before execution remains. More details on the APU protection features in the DMU are available in the ‘Protection’ section of the NVM chapter.

3.2.3.4 Security

The security protection offered to an NVM operation on the PFLASH remains the same as defined in the ‘Protection’ section of the NVM chapter irrespective of the active or inactive nature of the PFLASH region.

3.3 Resource access latencies

The following tables describe the CPU access times to various resources in SRI clock cycles.

- For the values of data access or instruction fetch accesses, the access times are the minimum number of CPU stall cycles to complete the access
- If there is a conflict for the resource accessed, there may be additional stall cycles till the conflicting access completes
- For write access, the access times are the maximum for a sequence of such access (non-conflicting). In many cases for a single access or a short sequence, write buffering reduces the stall effect seen by a CPU, sometimes to 0. However, as with loads and fetches, if there is a conflict for the resource accessed, there may be additional stall cycles till the conflicting access completes
- For data accesses, the access times listed are for 32-bit accesses. The access times for 64-bit write accesses to DLMU or LMU memories can be lower than the numbers listed below due to the SRAM organization

Access latency for FPI bus peripherals

Access to critical module registers is implemented with 1 wait-state (for example QSPI, ASCLIN, EVADC, DSADC result registers and DMA). Some modules may implement additional wait-states as for example in the following instances:

- Access to module internal SRAMs (for example MCAN, DMA, ERAY, GTM, SCR, GETH)
- System infrastructure modules (for example VMT, DMA)
- Access to control registers in large and complex modules that require internal registers stages to meet the timing (for example GTM)

Access latency for PFlash and DFlash accesses

The PFlash and DFlash wait-states are described in the NVM section of the documentation. The wait-states are configured through the following registers:

- PFLASH(HOST): DMU_GP_HOST_PFWAIT
- PFLASH(CSRM): DMU_GP_CSRM_PFWAIT
- DFLASH(HOST): DMU_GP_HOST_DFWAIT
- DFLASH(CSRM): DMU_GP_CSRM_DFWAIT

In all cases the values programmed in the registers are 1 less than the actual value. For example, DMU_GP_HOST_PFWAIT.RFLASH=7 corresponds to 8 read cycles.

For example, the number of stall cycles for a data read from a PFlash instance with DMU_GP_HOST_PFWAIT.RFLASH=7 accessed on the local SRI is as follows.

```
PWS           = (PFLASH READ CYCLES)
              = (DMU_GP_HOST_PFWAIT.RFLASH + 1)
              = 8

Stall cycles  = 5 + PWS
              = 13
```

3.3.1 TC4Dx resource access latencies

Access latency for FPI bus peripherals

The following table lists the stall cycles for CPU accesses to FPI peripherals.

Table 37 Access latency for FPI peripherals (TC4Dx)

CPU access type	Stall cycles for SPB peripheral access[1]	Stall cycles for ComPB peripheral access[1]
CPU0-5 data read[2]	$n \times (4 + \text{Module wait–states})$, where $\left(n = \frac{f_{CPU}}{f_{SPB}}\right)$	$n \times (4 + \text{Module wait–states})$, where $\left(n = \frac{f_{CPU}}{f_{SPB}}\right)$
CPU0-5 data write[2]	$n \times (4 + \text{Module wait–states})$, where $\left(n = \frac{f_{CPU}}{f_{SPB}}\right)$	$n \times (4 + \text{Module wait–states})$, where $\left(n = \frac{f_{CPU}}{f_{SPB}}\right)$
PPU data read[2]	50	50
PPU data write	17	17

1) The number of wait-states for read and for write accesses is >= 1 and depends on the accessed module and its configuration.
2) The final number of stall cycles will depend on the real number of wait-states generated by the target resource.

***Note**: Due to the various bridges between the PPU and the peripherals no generic formula can be given. The numbers documented are for the following frequency ratio PPU/Peripheral bus: 450 MHz/100 MHz*

Access latency for CPU local resources

The following table lists the stall cycles for CPU instruction fetch, read, and write accesses to internal memories.

Table 38 CPU accesses: Stall cycles for local resources

	Local CPU
Data read from DSPR	0
Data write to DSPR	0
Instruction fetch from DSPR	See SRI resources tables[1]
Data read from DLMU	1
Data write to DLMU	4 (2 for 64-bit write accesses)
Instruction fetch from DLMU	See SRI resources tables[1]
Data read from PSPR	See SRI resources tables[1]
Data write to PSPR	See SRI resources tables[1]
Instruction fetch from PSPR	0
Data read from PFlash	5 + PWS[2]
Instruction fetch from PFlash (buffer miss)	2 + PWS[2]
Instruction fetch from PFlash (buffer hit)	3

1) Data access to code side memories or code accesses to data side memories are made through the SRI interconnect.
2) PWS: Configured PFlash wait-states (Includes cycles for PFlash access cycles only). ECC correction latency is only incurred when the incoming data requires ECC correction.

Access latency for SRI resources

The following tables list the stall cycles for CPU read and write accesses over the SRI interconnect to various target memories.

Table 39 CPU accesses: Stall cycles for reads to SRI resources (TC4Dx)

	DSPR	DSPR	DLMU	DLMU	LMU	LMU	LMU	PFlash	PFlash	PPU	PPU
	0-2	3-5	0-2	3-5	0-1	2-3	4-9	0-2	3-5	CSM	VMEM
CPU0-2	7	14	7	14	8	15	15	$PWS + 9$	$PWS + 16$	36	38
CPU3-5	14	7	14	7	15	8	15	$PWS + 16$	$PWS + 9$	36	38
PPU	31	31	31	31	32	32	30	$\lceil PWS \times 0.9 \rceil + 33$	$\lceil PWS \times 0.9 \rceil + 33$	N.A.	N.A.

***Note**: The PPU is running asynchronous to the rest of the system. The documented numbers represent the average access latency in PPU clocks for accesses originating from the PPU. Individual accesses may see different access latencies. They are representative for SRI frequency of 500 and PPU frequency of 450, represented by a scaling factor of 0.9. PFlash access latency could change between random and sequential access.*

***Note**: The CPU stall cycles do not apply to the local PFlash path, for example CPU0 to PFlash0. Those stall cycles are listed in the local resources table.*

Table 40 CPU accesses: Stall cycles for non-pipelined writes to SRI resources (TC4Dx)

	DSPR	DSPR	DLMU	DLMU	LMU	LMU	LMU	PPU	PPU
	0-2	3-5	0-2	3-5	0-1	2-3	4-9	CSM	VMEM
CPU0-2	4	7	4	7	5	7	7	13	13
CPU3-5	7	4	7	4	7	5	7	13	13
PPU	7	7	7	7	7	7	6	N.A.	N.A.

Table 41 CPU accesses: Stall cycles for pipelined writes to SRI resources (TC4Dx)

	DSPR	DSPR	DLMU	DLMU	LMU	LMU	LMU	PPU	PPU
	0-2	3-5	0-2	3-5	0-1	2-3	4-9	CSM	VMEM
CPU0-2	3	5	4	5	4	5	5	13	13
CPU3-5	5	3	5	4	5	4	5	13	13
PPU	7	7	7	7	7	7	6	N.A.	N.A.

***Note**: The PPU is running asynchronous to the rest of the system. The documented numbers represent the average access latency, individual accesses may see different access latencies. They are representative for SRI frequency of 500 and PPU frequency of 450*

***Note**: Although the table lists DSPR access latencies, the PSPR access latencies will be equivalent to the listed DSPR latencies for data access through SRI*

A CPU's fetch latency from another CPU's PFlash either on the local or remote crossbar is influenced by the executing CPU's overall pipeline state. Additionally, it interacts with the pre-fetch mechanism when pre-fetch buffers are enabled on the PFlash path. The front end of the CPU pipeline operates on fetches which are equivalent to cache line size, even when the cache is disabled. The summary is indicative of the access latency assuming one pre-fetch buffer of the other CPU is assigned to the fetch CPU's P side SRI TAG-ID. Even though the instruction fetch and execution are decoupled, the actual stall cycles will depend on execution of fetched instructions. The following stall cycles apply when executing all instruction from the previously fetched line assuming they are all 32_B, can be dual issued and is devoid of exceptions, dependencies and hazards:

Table 42 CPU stall cycles due to line fetch(256b) over SRI from PFlash of another CPU

Fetch conditions	Pflash on local crossbar	Pflash on other crossbar
Instruction fetch (buffer miss)	$PWS + 9$	$PWS + 16$
Instruction fetch (buffer hit)[1)]	$\max(PWS + 2, 11) - 4$	$\max(PWS + 2, 18) - 4$

1) The reduction by 4 accounts for the 4 cycles to execute 8 instructions delivered in the fetch packet

3.3.2 TC4Dx SDMA throughput

The following tables describe the achieved SDMA throughput moving data from and to various system memories.

- The throughput is measured at the following clock frequencies: f_{SRI} = 500 MHz, f_{SPB} = 100 MHz
- The throughput is measured as the Gbyte/s (rounded-down) achieved by the SDMA instance while moving 32 KB of data from the source memory to a destination memory using a 512-bit move size
- When there are multiple DMA channels accessing a common memory location, the DMA channels are configured with addresses greater than 64 KB apart such that the DMA move requests to the memory are not pipelined
  - If the two DMA channels happen to target addresses within the same 64 KB region on source or destination memories, the throughput achieved will be better than the results below
- If there is a conflict for the resource accessed, there may be a drop in the throughput achieved till the conflicting access completes

Table 43 TC4Dx SDMA0 single channel throughput (Gbyte/s)

	Read location							
Write location	**DSPR0-2**	**DSPR3-5**	**DLMU0-2**	**DLMU3-5**	**LMU0-1**	**LMU2-3**	**LMU4-6**	**LMU7-9**
DSPR0-2	1	1	1	1	1	0.9	0.9	0.8
DSPR3-5	1	0.9	1	0.9	0.9	0.9	0.9	0.8
DLMU0-2	1	1	1	1	1	0.9	0.9	0.8
DLMU3-5	1	0.9	1	0.9	0.9	0.9	0.9	0.8
LMU0-1	1	1	1	1	1	0.9	0.9	0.8
LMU2-3	1	0.9	1	0.9	0.9	0.9	0.9	0.8

(table continues...)

Table 43 (continued) TC4Dx SDMA0 single channel throughput (Gbyte/s)

	Read location							
Write location	**DSPR0-2**	**DSPR3-5**	**DLMU0-2**	**DLMU3-5**	**LMU0-1**	**LMU2-3**	**LMU4-6**	**LMU7-9**
LMU4-6	1	0.9	1	0.9	0.9	0.9	0.9	0.8
LMU7-9	1	0.9	1	0.9	1	0.9	0.9	0.8

Table 44 TC4Dx SDMA0 dual channel throughput (Gbyte/s)

	Read location							
Write location	**DSPR0-2**	**DSPR3-5**	**DLMU0-2**	**DLMU3-5**	**LMU0-1**	**LMU2-3**	**LMU4-6**	**LMU7-9**
DSPR0-2	1.8	1.8	1.8	1.8	1.8	1.7	1.7	1.4
DSPR3-5	1.9	1.5	1.9	1.5	1.9	1.4	1.7	1.4
DLMU0-2	1.8	1.8	1.8	1.8	1.8	1.7	1.7	1.4
DLMU3-5	1.9	1.5	1.9	1.5	1.9	1.4	1.7	1.4
LMU0-1	1.8	1.8	1.8	1.8	1.8	1.7	1.7	1.4
LMU2-3	1.9	1.5	1.9	1.5	1.9	1.4	1.7	1.4
LMU4-6	1.9	1.8	1.9	1.8	1.9	1.7	1.4	1.4
LMU7-9	2	1.8	2	1.8	1.9	1.7	1.4	1.4

Table 45 TC4Dx SDMA1 single channel throughput (Gbyte/s)

	Read location							
Write location	**DSPR0-2**	**DSPR3-5**	**DLMU0-2**	**DLMU3-5**	**LMU0-1**	**LMU2-3**	**LMU4-6**	**LMU7-9**
DSPR0-2	0.9	1	0.9	1	0.9	0.9	0.8	0.9
DSPR3-5	1	1	1	1	0.9	1	0.8	0.9
DLMU0-2	0.9	1	0.9	1	0.9	0.9	0.8	0.9
DLMU3-5	1	1	1	1	0.9	1	0.8	0.9
LMU0-1	0.9	1	0.9	1	0.9	0.9	0.8	0.9
LMU2-3	1	1	1	1	0.9	1	0.8	0.9
LMU4-6	0.9	1	0.9	1	0.9	1	0.8	0.9
LMU7-9	0.9	1	0.9	1	0.9	0.9	0.8	0.9

Table 46 TC4Dx SDMA1 dual channel throughput (Gbyte/s)

	Read location							
Write location	**DSPR0-2**	**DSPR3-5**	**DLMU0-2**	**DLMU3-5**	**LMU0-1**	**LMU2-3**	**LMU4-6**	**LMU7-9**
DSPR0-2	1.5	1.9	1.5	1.9	1.4	1.9	1.4	1.7

(table continues...)

Table 46 (continued) TC4Dx SDMA1 dual channel throughput (Gbyte/s)

	Read location							
Write location	**DSPR0-2**	**DSPR3-5**	**DLMU0-2**	**DLMU3-5**	**LMU0-1**	**LMU2-3**	**LMU4-6**	**LMU7-9**
DSPR3-5	1.8	1.8	1.8	1.8	1.7	1.8	1.4	1.7
DLMU0-2	1.5	1.9	1.5	1.9	1.4	1.9	1.4	1.7
DLMU3-5	1.8	1.8	1.8	1.8	1.7	1.8	1.4	1.7
LMU0-1	1.5	1.9	1.5	1.9	1.4	1.9	1.4	1.7
LMU2-3	1.8	1.8	1.8	1.8	1.7	1.8	1.4	1.7
LMU4-6	1.8	2	1.8	2	1.7	1.9	1.4	1.4
LMU7-9	1.8	1.9	1.8	1.9	1.7	1.9	1.4	1.4

3.4 Module reset

It is proposed to stop all participants that have read or write access to the functional block (FB) kernel registers during the normal application, before a functional block module reset is started by application software, through the functional block KRST registers.

This means that for example DMA channels and software tasks that access the kernel are to be stopped, and the last write accesses by bridges must be waited for.

To detect unexpected write accesses during or after functional block kernel access, the kernel can also be switched off through the CLC register before a kernel reset.

Accesses to the kernel that is switched-off are normally terminated with an SRI or FPI error and signaled as an alarm to the SMU.

Many functional blocks support a reset of their kernel function.

If available, the kernel reset can be started and controlled with the registers:

- RST_CTRLA
- RST_CTRLB
- RST_STAT

There are also functional blocks with several kernel functions that have their own kernel reset function, such as ADC for example.

Which functional block registers are reset by a kernel reset function is described in the respective functional block register overview table.

Registers that are not normally reset by a kernel reset are PROTs, APUs, OCS, and CLC.

4 Central Processing Unit (CPU)

The TriCore™ 1.8 central processing unit (TC1.8 CPU) is a superscalar implementation of the version 1.8 of the TriCore™ architecture. TriCore™ is a 32-bit microcontroller-DSP architecture optimized for real-time embedded systems.

The TriCore™ instruction set architecture (ISA) combines the real-time capability of a microcontroller, the computational power of a Digital Signal Processor (DSP), and the high performance and features of a Reduced instruction set computer (RISC) load-store architecture, in a compact re-programmable core.

The ISA supports a uniform, 32-bit address space and memory-mapped Input/Output (IO). The architecture supports both 16-bit (reduced code space and low memory requirement) and 32-bit instruction formats. Real-time responsiveness is supported by low interrupt latency and fast-context-switching. The high performance architecture minimizes interrupt latency by avoiding long multi-cycle instructions and by providing a flexible hardware-supported interrupt scheme.

Compared to version 1.6, version 1.8 of the TriCore™ architecture includes virtualization support and a double precision IEEE-754-2019 Floating-point unit.

Please note that TriCore™ architectural information is defined in the TriCore™ 1.8 architecture manual. Topics covered by that manual include:

- Architecture overview
- Programming model
- CPU registers
- Tasks and functions
- Interrupt handling
- Traps
- Memory protection system
- Temporal protection system
- Floating-point operations
- Debug
- Instruction set
- Virtualization

***Note**: The optional MMU extension is not implemented in this product family.*

4.1 Feature list

The real-time core implementations have the following features:

- 32-bit load store little-endian architecture
- 16-bit and 32-bit instructions for reduced code size
- Single and double-precision floating-point unit (IEEE-754-2019)
- Virtualization
  - Dual virtual machine support per core with three hardware resource partitions replicating most state
  - Support for one hypervisor, one real-time virtual machine, and up to 6 other virtual machines
- Temporal protection system allowing time-bounded real-time operation
- 4 Gbyte address range (2^{32})
- Multiple data types
  - Boolean, integer with saturation, bit array, signed fraction, character, double-word integers, quad-word integers, signed integers and unsigned integers
- Multiple data formats
  - Bit, byte (8-bit), half-word (16-bit), word (32-bit), double-word (64-bit) and quad-word (128-bit)

- Byte and bit addressing
- Multiply and accumulate (MAC) instructions
  - Single 32 x 32 and dual 16 × 16, 16 × 32 architecture
- Integer arithmetic with saturation
- Packed data
- Addressing modes
  - Absolute, circular, bit reverse, long + short, base + offset with pre-update and post-update
- Instruction types
  - Arithmetic, address arithmetic, comparison, address comparison, logical, MAC, shift, coprocessor, bit logical, branch, bit-field, load and store, packed data and system
- General Purpose Register Set (GPRS):
  - Sixteen 32-bit data registers
  - Sixteen 32-bit address registers
  - Three 32-bit status and program counter registers (PSW, PC, PCXI)
- Debug support
- Flexible memory protection system providing multiple protection sets with multiple protection ranges per set
- Dedicated system timer with extensions for virtualization

4.2 Functional overview

The central processing unit implements version 1.8 of the TriCore™ architecture. A device specific number of instances can be used to run the application software. The CPU operation is supported by local SRAMs, a local NVM connection and configuration registers. The TC1.8 CPU block diagram is shown in the figure below:

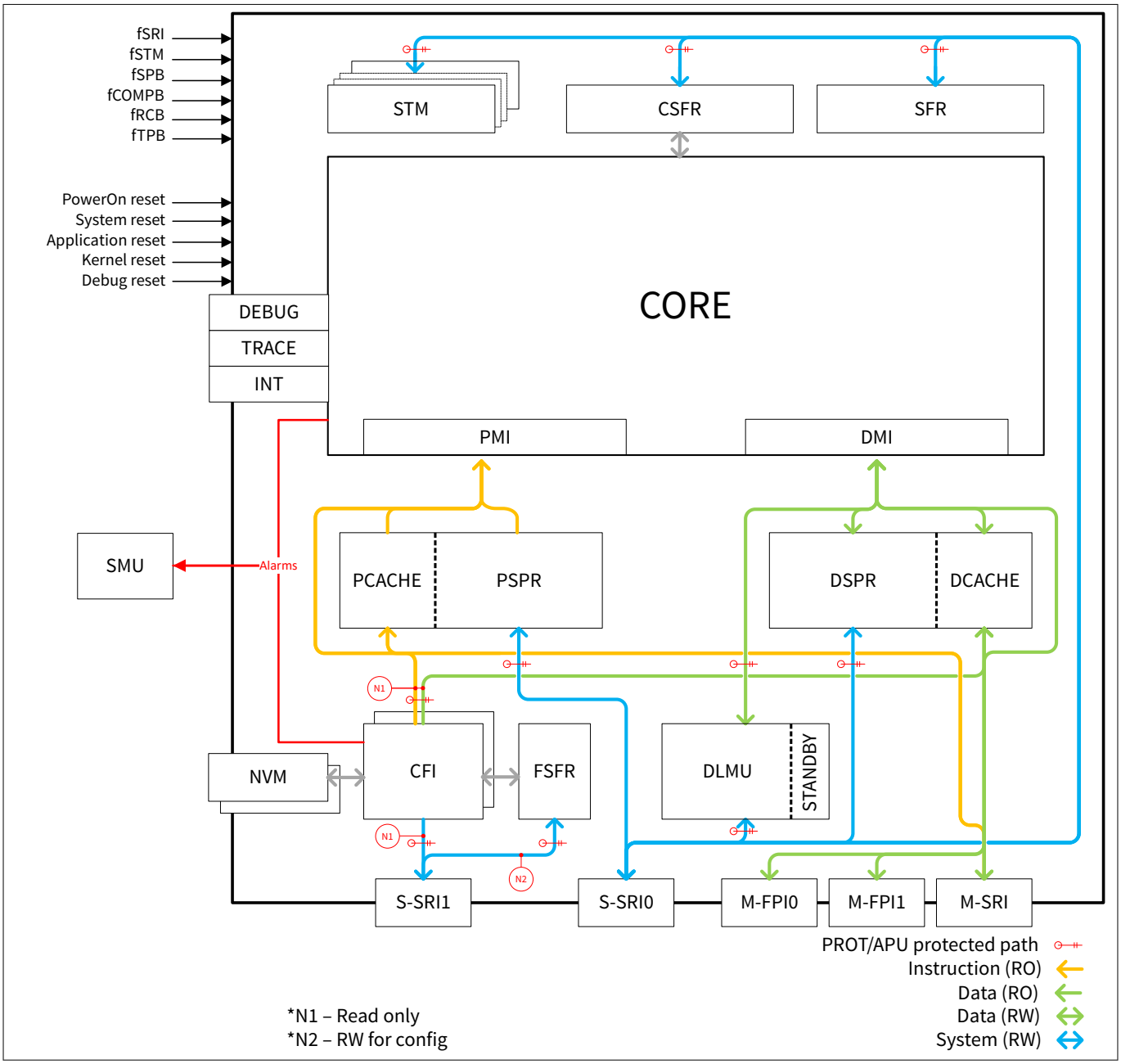


Figure 18 CPU block diagram

For the alarm description please refer to the alarm mapping tables in the SMU functional block and the Safety Manual.

TC1.8 CPU connections

The TC1.8 CPU contains interfaces to the following memories, interconnect, and bus interfaces:

- Data scratchpad SRAM (DSPR)
- Program scratchpad SRAM (PSPR)
- Data cache (DCache)

- Program cache (PCache)
- Distributed LMU memory (DLMU)
- Local NVM bank (x2 some instances) through the NVM interface (CFI)
- SRI slave interface for other resources (S-SRI0)
- SRI slave interface for Local NVM (S-SRI1)
- SRI master interface (M-SRI)
- FPI master interface (M-FPI0)
  - FPI0 is always connected to the SPB
- FPI master interface (M-FPI1)
  - FPI1 can connect to other FPI buses depending on the device and the core instance

The protected paths are:

- S-SRI0 accesses to PSPR, DSPR, DLMU, SFR, CSFR, or STM
- S-SRI1 (if present) accesses to CFI, or FSFR
- PMI or PCACHE accesses to CFI
- DMI or DCACHE accesses to CFI
- DMI accesses to local DLMU

TC1.8 CPU units

The TC1.8 CPU comprises of the following units:

- TriCore™ based processing unit (CORE)
  - Program memory interface (PMI)
  - Data memory interface (DMI)
- Special function registers (SFR)
- Core special function registers (CSFR)
- NVM special function registers (FSFR)
- System timer (STM)

Related information

TC4Dx SMU alarm mapping tables on page 7233

4.2.1 Acronyms and terms

A table of common acronyms or terms that are referenced in this functional block, with a corresponding definition or explanation.

Table 47 CPU acronyms and terms

Acronym or term	Definition or explanation
CFI	Program NVM and FSFR interface
CPUcs	CPU within the cyber security real-time module
DCache	Data cache
DMBI	Data memory and bus interface unit
DSP	Digital signal processing
DSPR	Data scratchpad RAM
DTAG	Data cache tag RAM

(table continues...)

Table 47 (continued) CPU acronyms and terms

Acronym or term	Definition or explanation
FIFO	First in first out
FPU	Floating-point unit
FSFR	NVM special function registers
IEEE	Institute of Electrical and Electronics Engineers
IEEE-754	IEEE Standard for Floating-point Arithmetic
IEEE-754-2019	IEEE-754 revision 2019
MMU	Memory management unit
NVM	Non-volatile memory
OCDS	On-chip debug system
OTGB	OCDS trigger bus
PCache	Program cache
PMBI	Program memory and bus interface unit
PNVM	Program NVM
PSPR	Program scratchpad RAM
PTAG	Program cache tag RAM
RISC	Reduced instruction set computer
SIST	Software in system test
TC1.8	TriCore™ 1.8

4.2.2 CPU block diagram

The processor core comprises of:

- An Instruction Fetch Unit
- An Execution Unit
- A General Purpose Register File (GPR)
- A CPU Slave interface (CPS)
- Optional Floating-Point Unit (FPU)
  - Single and double-precision floating-point units,
  - or single precision floating-point unit only
  - or no FPU

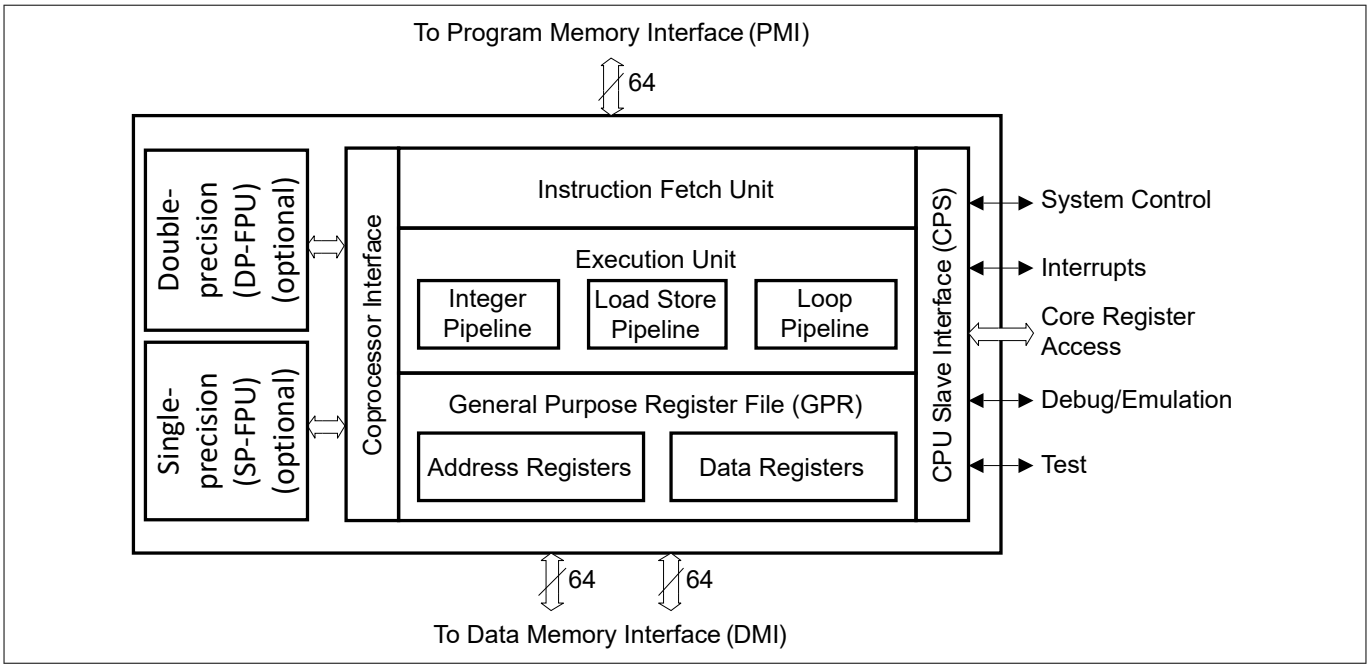


Figure 19 CPU block diagram

4.2.3 Instruction fetch unit

The Instruction Fetch Unit pre-fetches and aligns incoming instructions from the 64-bit wide Program Memory Interface (PMI). Instructions are placed in predicted program order in the Issue unit (FIFO).

The Instruction Protection Unit checks the validity of accesses to the PMI and the integrity of incoming instructions fetched from the PMI.

The branch unit (Prefetch) examines the fetched instructions for branch conditions and predicts the most likely execution path based on previous branch behavior. The Program Counter Unit (PC) is responsible for updating the program counter.

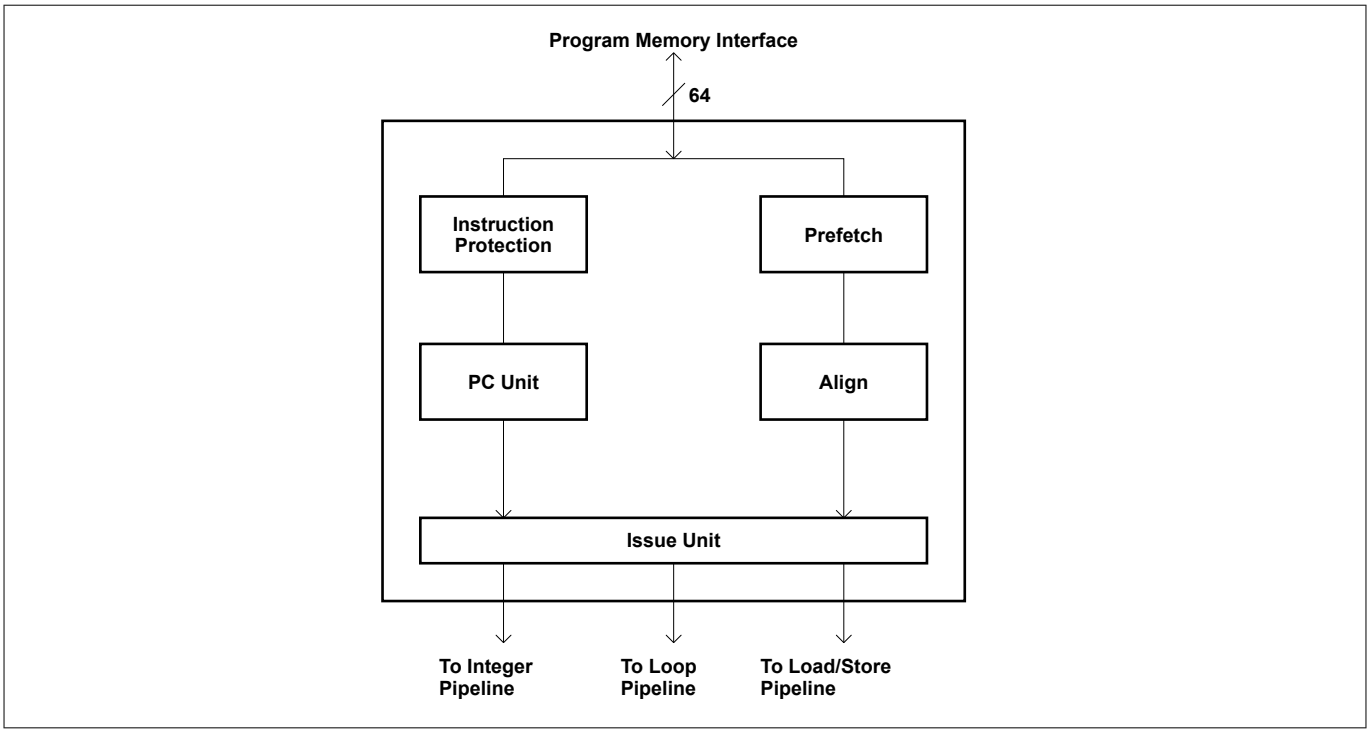


Figure 20 Instruction fetch unit

4.2.4 Execution unit

The Execution Unit contains the Integer Pipeline (IP), the Load/Store (LS) Pipeline and the Loop Pipeline (LP). All three pipelines operate in parallel, permitting up to three instructions to be executed in one clock cycle. In the execution unit all instructions pass through a decode stage followed by two execute stages (three for multiply-accumulate (MAC)). Pipeline hazards (stalls) are minimized by the use of forwarding paths between pipeline stages allowing the results of one instruction to be used by a following instruction as soon as the result becomes available.

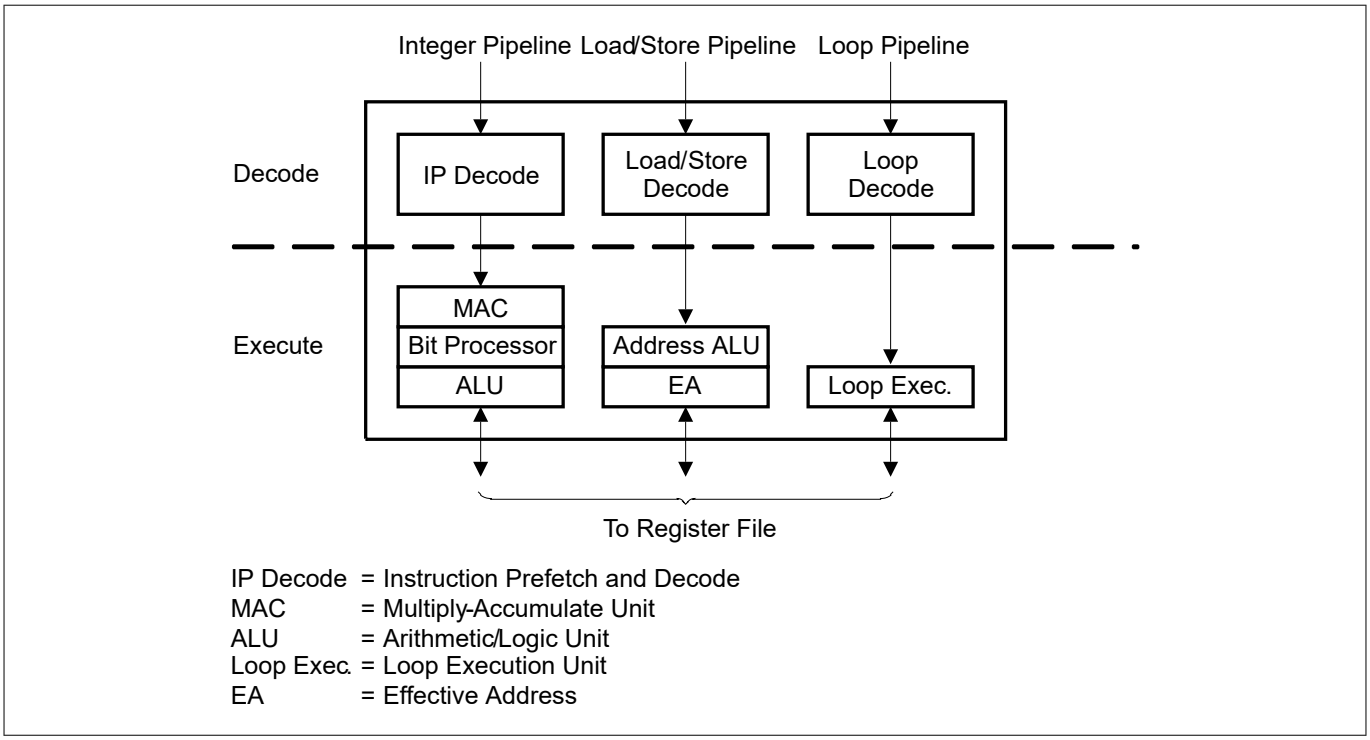


Figure 21 Execution unit

4.2.5 General purpose register file

The CPU has a General Purpose Register (GPR) file, divided into an Address Register File (registers A0 through A15) and a Data Register File (registers D0 through D15).

The data flow for instructions issued to the Load/Store Pipeline is steered through the Address Register File.

The data flow for instructions issued to the Integer Pipeline and for data load/store instructions issued to the Load/Store Pipeline is steered through the Data Register File.

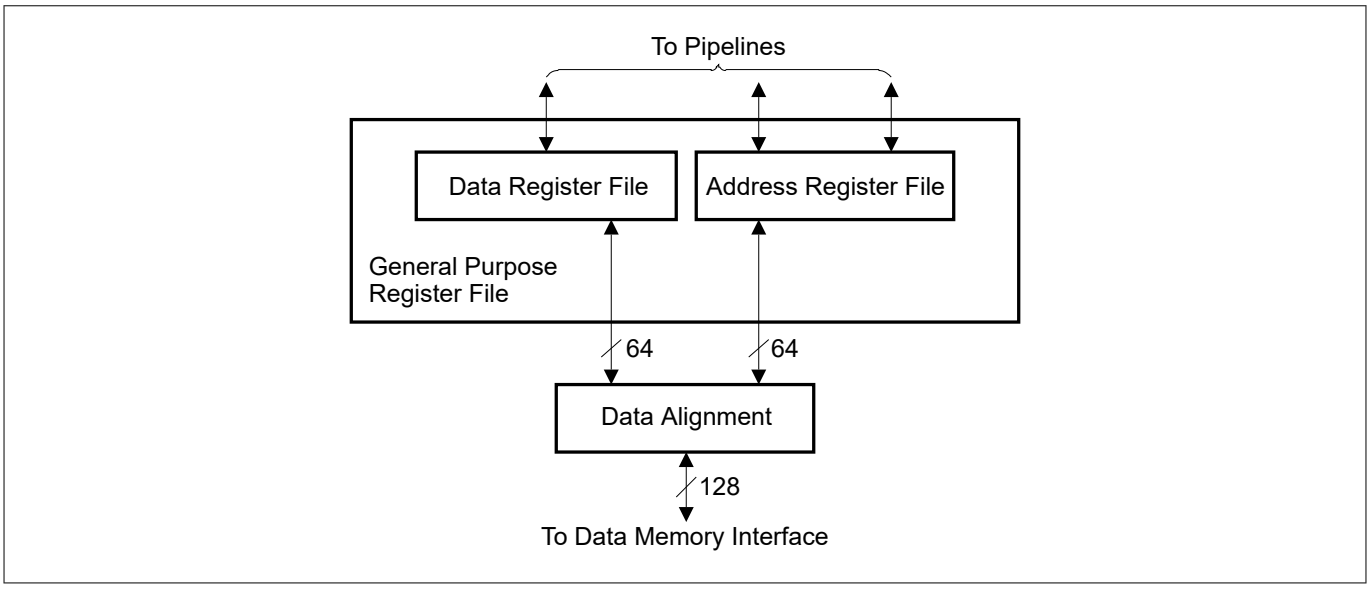


Figure 22 General purpose register file

4.3 Functional description

This sections describes the architectural choices made where there are allowed options, the implementation features such as instruction timing and memory hierarchy and system features implemented in the TC1.8 CPU like overlay support for calibration.

4.3.1 CPU implementation specific features

A description of those properties of the TC1.8 CPU which are allowed by version 1.8 of the TriCore™ architecture for an implementation to determine.

4.3.1.1 Configurability

Hardware support for several optional extensions is provided on specific TriCore™ instances. Refer to the TCCON register for a runtime accessible definition of the instance hardware features. Cores in some members of the product family or even some cores within a product might not have the same hardware features.

4.3.1.2 Context save areas and context operations

The CPU uses a uniform context-switching method for function calls, interrupts, and traps. In all cases the Upper Context of the task is automatically saved and restored by hardware. Saving and restoring of the Lower Context may be optionally performed by software.

Please refer to the TriCore™ architecture manual volume 1 for more details. The Context Save Areas (CSA) and addresses targeted by explicit context load or store instructions, such as LDLCX for example, may be placed in DSPR, DLMU, or non-peripheral SRI memory (cached or uncached).

CSA placement in DSPR or DLMU

The actual timing of context operations is dependent upon the placement of the Context Save Areas. Maximum performance is achieved when the Context Save Area is placed in DSPR.

In this case all context save and restore operations take four cycles (using DSPR assuming no external interference).

If the CSA is placed in the DLMU, context save and restore operations take eight cycles.

CSA placement in cached external memory

In this situation the timing is also dependent on the state of the data cache. The best case for CSA operations using the data cache occurs when context saves do not incur a cache line writeback, and context restores hit in the data cache (best case for a cache hit is four cycles).

CSA placement in non-cached external memory

CSA placement in non cached external memory is allowed, but not recommended for performance reasons. The execution time will be dependent on the system memory used.

4.3.1.3 Program counter (PC) register

The Program Counter (PC) holds the address of the instruction that is currently fetched and forwarded to the CPU pipelines. The CPU handles updates of the PC automatically.

Software can use the current value of the PC for various tasks, such as performing code address calculations.

Reading the PC through software executed by the CPU must only be done with an MFCR instruction. Such a read will return the PC of the MFCR instruction itself.

Explicit writes to the PC through an MTCR instruction must not be done due to possible unexpected behavior of the CPU. The PC may be written by an external agent only when the CPU is halted.

The CPU will not perform Load or Store instructions to the mapped address of any of the CSFR's in segment 15 (including the PC). A MEM trap will be generated if such an access is attempted.

Bit 0 of the PC register is read-only and is hard-wired to 0.

4.3.1.4 Store buffers

To increase performance, the TC1.8 CPUs implement store buffering to decouple memory write operations from CPU instruction execution. All stores from the CPU are placed in the store buffer prior to being written to local memory or transferred through the system interconnect.

Write data is taken from the store buffers and written when the target memory or interface becomes available.

In normal operation, with bit SMACON.IODT=0_B (default after reset), non-dependent loads may bypass store operations. When both a store and a load operation are present in the pipeline, the CPU will prioritize load operations over store operations in order to improve performance unless:

- The store buffer is full
- The load is to peripheral space and a store to peripheral space exists in the store buffer (in order access to peripheral space is guaranteed)
- The access belongs to an atomic operation
  - Atomic operations require an empty store buffer, this ensures that any buffered state is written to memory prior to the atomic operation
- The in-order data transfer is enabled (SMACON.IODT)
  - Memory operations are performed in program order when the SMACON.IODT bit is set. This should only be performed by test routines at start-up or shut-down as it will severely limit performance

To improve the performance of load operations, the CPU will snoop the content of the store buffer for a matching address. If a match is found, the load data is retrieved from the store buffer before the store is committed to memory.

To improve performance the store buffer will merge consecutively addressed[3)] data elements to reduce the number of memory accesses required. The following operations are merged in the store buffer:

- Consecutive byte writes to the same half-word memory location
- Consecutive half-word writes to the same word memory location
- Consecutive word writes to the same double-word memory location

For snooping and merging operations the properties of the load or store operation are checked against properties of the store contained in the store buffer. In such case the store buffer entry is flushed before the load completes. Snooping and merging do not happen when:

- The load or store belongs to a different virtual machine
- The load or store belongs to a task executed with a different protection set
- The load and store operations do not use the same master agent function identifier

The TC1.8 CPU store buffer can hold the data for up to 6 operations.

For TC1.8, the store buffers are implemented as a FIFO. An item at the head of the FIFO can be commenced on a distinct target as long as the previous head item has commenced even if it has not completed. The distinct targets are:

- Local DSPR
- Local DLMU
- SRI
- FPI (FPI0/FPI1 where present)

[3] Since loads can bypass stores, two store operations to be merged may not be consecutive in the instruction flow

This allows an item to be removed from the head of the FIFO, every clock cycle, to commence access to each free target without stalling the entire FIFO.

If there is a requirement that data is written to local memory prior to execution of a subsequent instruction then a LSYNC or a DSYNC instruction may be used to flush the store buffers. A DSYNC instruction must be used to guarantee that a write is completed at the target destination.

4.3.1.5 Interrupt system

An interrupt request can be generated by on-chip peripheral units, or it can be generated by external sources. Requests can be targeted to any CPU (please refer to the interrupt system described in the Interrupt Router (IR) chapter).

The interrupt system evaluates service requests for priority and to identify whether the CPU should receive the request. The highest-priority service request is then presented to the CPU.

On taking an interrupt the CPU will vector to a unique PC generated from the interrupt priority number and the Base Interrupt Vector (BIV). The spacing between the vector PCs in the interrupt vector table may be selected to be either 32 Bytes or 8 Bytes using BIV.VSS.

The TC1.8 CPU implements a fast interrupt mechanism for consecutive interrupts which avoids some context save and restore operations and therefore speeds up interrupt routine entry. The fast interrupt mechanism operates when:

- An Interrupt is pending
- A Return From exception (RFE) instruction is being executed
- The priority of the pending interrupt is greater than the priority level that would be returned to should the RFE be executed (ICR.PIPN > PCXI.PCPN)
- Interrupts will be enabled after RFE is executed (PCXI.PIE == 1)

The fast Interrupt mechanism will operate unless a MTCR, or MTDCR, instruction targeting PCXI, PSW, PC, CORECON, CPU_ID, CORE_ID, BIV, BTV, ISP, ICR, PPRS, FCX, LCX, BOOTCON, CCON, DMS, DCX, DBGTCR, HRHV_VMn_ICR and DBGCFG executes before the RFE.

The fast interrupt mechanism will operate unless context instruction executes before the RFE. These instructions are: call (CALL, CALLA, CALLA), return (RET), begin interrupt service routine (BISR), save lower context (SVLCX), restore lower context (RSLCX), load upper or lower context (LDUCX, LDLCX), and store upper or lower context (STUCX, STLCX).

In the absence of the fast interrupt mechanism, an interrupt immediately following an RFE would cause a restore of the upper context immediately followed by a save of the same upper context back to exactly the same memory location.

The fast interrupt mechanism replaces this redundant restore and save sequence with a load of PCXI, or PSW, or A10, or A11 from the saved context. System state at the end of the fast interrupt is the same as if the standard RFE or interrupt sequence had been performed.

Related information

Interrupt Router (IR) on page 6857

4.3.1.6 Trap system

The following traps have implementation-specific properties.

- MPX Memory Protection Execute (Class 1 TIN 4)
  - The MPX trap is raised whenever a program attempts to execute an instruction from a memory area for which it does not have execute permission and memory protection is enabled

- UOPC - Unimplemented Opcode (Class 2 TIN 2)
  - The UOPC trap is raised on optional half-precision and single-precision FPU instructions (coprocessor 1) when hardware support is not provided
  - The UOPC trap is raised on optional double-precision FPU instructions (coprocessor 2) when hardware support is not provided
  - The UOPC trap is raised on optional MMU instructions and coprocessor three instructions, neither are implemented in TC1.8
- OPD - Invalid Operand (Class 2 TIN 3)
  - The OPD trap is raised for instructions that take even-odd register pairs as an operand where the operand specifier is odd
  - The OPD trap is also raised for instructions that take a quad data register as an operand where the operand specifier is not [0, 4, 8, 12]
  - The OPD trap is also raised when a MTCR, MFCR, MTDCR, or MFDCR targets a GPR, a non-existing CSFR (virtualization disabled), or uses a misaligned offset
  - When virtualization is not present or disabled, the OPD trap is also raised for MTCR or MTDCR targeting CSFRs protected with PROTSFRE (virtualization disabled)
- DSE - Data Access Synchronous Error (Class 4 TIN 2)
  - The DSE trap is generated by the DMI module for an out-of-range access to DSPR
  - The DSE trap is also generated when a load access from the CPU encounters certain error conditions, such as an interconnect error
  - When a DSE trap is generated, the exact cause of the error can be determined by reading the Data Synchronous Trap Register, DSTR. Details of possible error conditions and the corresponding flag bits are found in that register
- DAE - Data Access Asynchronous Error (Class 4 TIN 3)
  - The DAE is generated by the DMI module when a store or cache management access from the CPU encounters certain error conditions, such as an interconnect error
  - When a DAE trap is generated, the exact cause of the error can be determined by reading the Data Asynchronous Trap Register, DATR. Details of possible error conditions and the corresponding flag bits are found in that register
- PIE Program Memory Integrity Error (Class 4 TIN 5)
  - The PIE trap is raised whenever an uncorrectable memory integrity error is detected in an instruction fetch from a local memory or the SRI
  - Program memories are protected from memory integrity errors on a 64-bit basis. A PIE trap is raised when an attempt is made to execute an instruction from any fetch group containing a memory integrity error
  - The PIEAR and PIETR registers may be interrogated to determine the source of any error more precisely. PIEAR and PIETR are only updated if PIETR.IED is not set
- DIE Data Memory Integrity Error (Class 4 TIN 6)
  - The DIE trap is raised whenever an uncorrectable memory integrity error is detected in a data access to a local memory or the SRI
  - DIE traps are always asynchronous independent of the operation which encountered the error
  - DSPR memories are protected from memory integrity errors on a 16-bit basis. DLMU memories are protected from memory integrity errors on a 64-bit basis
  - A DIE trap is raised if any memory half-word (local DSPR) or double-word (local DLMU or SRI) accessed by a load or store operation contains an uncorrectable error and no other DIE trap have been raised since the clearing of the DIETR.IED bit

- The DIEAR and DIETR registers may be interrogated to determine the source of any error more precisely
- Subsequent DIE traps are inhibited if DIETR.IED is set

4.3.1.7 WAIT instruction

The WAIT instruction will suspend the execution of instructions until the occurrence of one of the following events:

- Enabled Interrupt
- Non-Maskable Interrupt
- Asynchronous Trap
- Idle Request
- Suspend Request
- Asynchronous Debug Halt or Trap Request
- Hypervisor and VM switch

4.3.1.8 Speculation extent

The TC1.8 CPU may perform both necessary and speculative accesses:

- Necessary accesses
  - Those accesses that the CPU is required to make to correctly compute the program
- Speculative accesses
  - Those accesses that the CPU may make in order to improve performance either in correct or incorrect anticipation of a necessary access

The following speculative accesses are performed by the TC1.8 CPU:

- Speculative fetch accesses
  - The processor may speculatively fetch up to 64 bytes of instruction on the predicted execution path
- Speculative data read accesses
  - For cached data locations the processor will read an entire cache line containing data read for a required access

The processor does not perform:

- Speculative instruction execution
- Speculative write accesses
- Speculative accesses to peripheral space

4.3.1.9 Instruction memory range limitations

To ensure the processor cores are provided with a constant stream of instructions the Instruction Fetch Units will speculatively fetch instructions from up to 64 bytes ahead of the current PC.

If the current PC is within 64 bytes of the top of an instruction memory the Instruction Fetch Unit may attempt to speculatively fetch instructions from beyond the physical memory range. This may then lead to error conditions and alarms being triggered by the bus and memory systems.

The processor will not attempt a prefetch if memory protection is enabled, and the region is not declared as executable by the MPU. Thus in systems that are not using memory protection the upper 64 bytes of any memory should therefore not be used for instruction storage. This memory area should be initialized with NOP instructions.

4.3.1.10 Atomicity of data accesses

The data alignment rules along with the number of bus transactions for each access type are detailed in the following tables.

Table 48 Alignment rules for non-peripheral space

Access type	Access size	Alignment of address in memory	Min/Max number of SRI transactions
Load, Store Data Register	Byte	Byte (1_H)	1/1
	Half-Word	2 bytes (2_H)	1/1
	Word	2 bytes (2_H)	1/2*
	Double-Word	2 bytes (2_H)	1/3*
	Quad-Word	4 bytes (4_H)	1/3*
Load, Store Address Register	Word	4 bytes (4_H)	1/1
	Double-Word	4 bytes (4_H)	1/2*
SWAP, LDMST, CMPSWAP.W, SWAPMSK.W	Word	4 bytes (4_H)	1/1
ST.T	Word[1)]	4 bytes (4_H)[2)]	1/1
Context operations	16 x 32-bit registers	64 bytes (40_H)	2/2*

1) The architecture is supported through the use of a word access.
2) The implementation takes the byte address and uses the word enclosing address to perform the access.

Table 49 Alignment rules for peripheral space

Access type	Access size	Alignment of address in memory	Number of SRI/FPI transactions
Load, Store Data Register	Byte	Byte (1_H)	1
	Half-Word	2 bytes (2_H)	1
	Word	4 bytes (4_H)	1
	Double-Word	8 bytes (8_H)	1
	Quad-Word	16 bytes (10_H)	1/2*
Load, Store Address Register	Word	4 bytes (4_H)	1
	Double-Word	8 bytes (4_H)	1
SWAP.W, LDMST, CMPSWAP.W, SWAPMSK.W	Word	4 bytes (4_H)	1
ST.T	Word[1)]	4 bytes (4_H)[2)]	1/1
Context operations	16 x 32 bit registers	Not permitted	-

Please note the following cases:

- Where a single access results in a single transaction, atomicity of the data is preserved when viewed from any external master

- Where a single access leads to multiple transactions (marked as "*" in the tables above) then atomicity is not guaranteed. In these accesses it is possible for another master to read or write some of the target memory locations between the transactions required to complete the access
- In the case of word and double word access to the SRI, the number of transactions will be 1 for naturally aligned data values and therefore atomicity is preserved

The atomicity of data accesses to CPU CSFR registers is not guaranteed if the local processor is performing a MTCR or MTDCR instruction to the same CSFR register.

The instructions SWAP.W, LDMST, CMPSWAP.W, SWAPMSK.W, and ST.T are always atomic in operation. They perform a word read followed by a word write to a word aligned memory location. Both read and write access permissions must exist for the word aligned address.

4.3.1.11 A11 usage

In normal usage A11 will always contain the target of the next RET, RFE, or RFH instruction. The processor uses this fact to speculatively load the return target ahead of the execution of the RET, RFE, or RFH instruction. Code that modifies the A11 (for example test code) should be aware that any value stored in A11 may be used as the target of such speculation. If the value in A11 is not a valid address the speculation may lead to error conditions and alarms being triggered by the bus and memory systems.

It is therefore recommended that A11 should only ever contain a valid address value. This can be achieved by ensuring that the modification of A11 is either followed by an ISYNC or by using another address general purpose register, when the computation leads to partial results, before transferring the complete result back to A11.

4.3.1.12 Independent core kernel reset

The CPU core may be reset while the rest of the system remains operational. A module reset can be triggered by the safety management unit (SMU) or by writing '1' into both of the module reset registers CPU_KRST1.RST and register CPU_KRST0.RST.

The cause of the last reset to occur is indicated by the CPU_KRST0.RSTSTAT[0] or CPU_KRST0.RSTSTAT[1] bits. The status bits may be cleared by writing KRSTCLR.CLR to one.

The KRSTCLR, KRST1 and KRST0 registers are protected by PROTRSTE.

Registers related to the Debug Reset (Class 1), and STM registers are not influenced by the kernel reset.

Related information

Safety and security alarm Management Unit (SMU) on page 7056

4.3.1.13 CPU clock control

The effective CPU execution frequency may be reduced by programming the associated CCON.DIV. The effective execution frequency seen by the processor is given by the following equation:

$$f_{CPU} = f_{SRI}\frac{(64 - \text{CCON.DIV})}{64}$$

Where:

- f_{CPU} is the effective frequency seen by the processor
- f_{SRI} is the base SRI frequency

A CCON.DIV value of 0 results in the core being clocked at the SRI frequency (no frequency reduction).

To avoid synchronization issues typically associated with clock division, the clock control mechanism stalls the issue of instructions into the processor pipeline rather than modifying the actual applied clock. An incoming instruction fetch packet is stalled for the number of cycles required to approximate the required execution

frequency. The stall is seen by the processor as a stall in the instruction stream in the same way a stalling instruction memory would be seen.

In most scenarios, this mechanism provides a good approximation to clock division based control. The actual reduction in effective frequency will be dependent on the executed code. TPS_TIMERn (n=0-2) and CCNT count rate is not affected by CCON.DIV.

4.3.1.14 CPU memory protection registers

The TC1.8 CPU implements eight Memory Protection Register Sets. The sets specify memory protection ranges and permissions for code and data. The PSW.PRS bit field determines which of these sets is currently in use by the CPU. Note that the three bit PRS field is composed as PRS[2:0] === {PSW[15], PSW[13], PSW[12]}.

The CPUs each implement 24 data and 16 code range comparators. These may be flexibly shared amongst the protection sets to provide a maximum of 24 data ranges and 16 code ranges per set.

The protection registers protect the whole address space. The granularity of the protection ranges is 8 bytes for data and 32 bytes for code.

Code protection ranges define from which memory areas the CPU is allowed to fetch instructions. Instruction fetches from an area outside a valid code protection range will lead to a trap condition.

Data protection ranges define which memory areas the CPU may access for read and/or write operations. Each range may be separately enabled for read and/or write access. Data read accesses from an area outside a valid data protection range with read permissions will lead to a trap condition. Data write accesses to an area outside a valid data protection range with write permissions will lead to a trap condition.

There are no implementation specific features of the Memory Protection Registers. They are described in detail in the TriCore™ architecture manual.

Crossing protection boundaries

When protection is enabled the TC1.8 CPU checks the base address of instruction fetch and data access operations against the enabled MPU protection ranges. If this base address is outside of the enabled protection ranges a trap will be raised.

The TC1.8 CPU does check the full extent of instruction fetch. An instruction fetch that starts at the very top of an enabled region that extends into a non-enabled region triggers a trap condition.

The TC1.8 CPU does not check the full extent of data access. A data access that starts at the very top of an enabled region may therefore extend into a non-enabled region without triggering a trap condition.

In the following example a 4-byte instruction “DCBA” is placed at a valid location at the end of “A” in protection Region-B but extends into invalid Region-A. Execution of this instruction will lead to an MPX trap condition because a portion of the instruction resides in an invalid region.

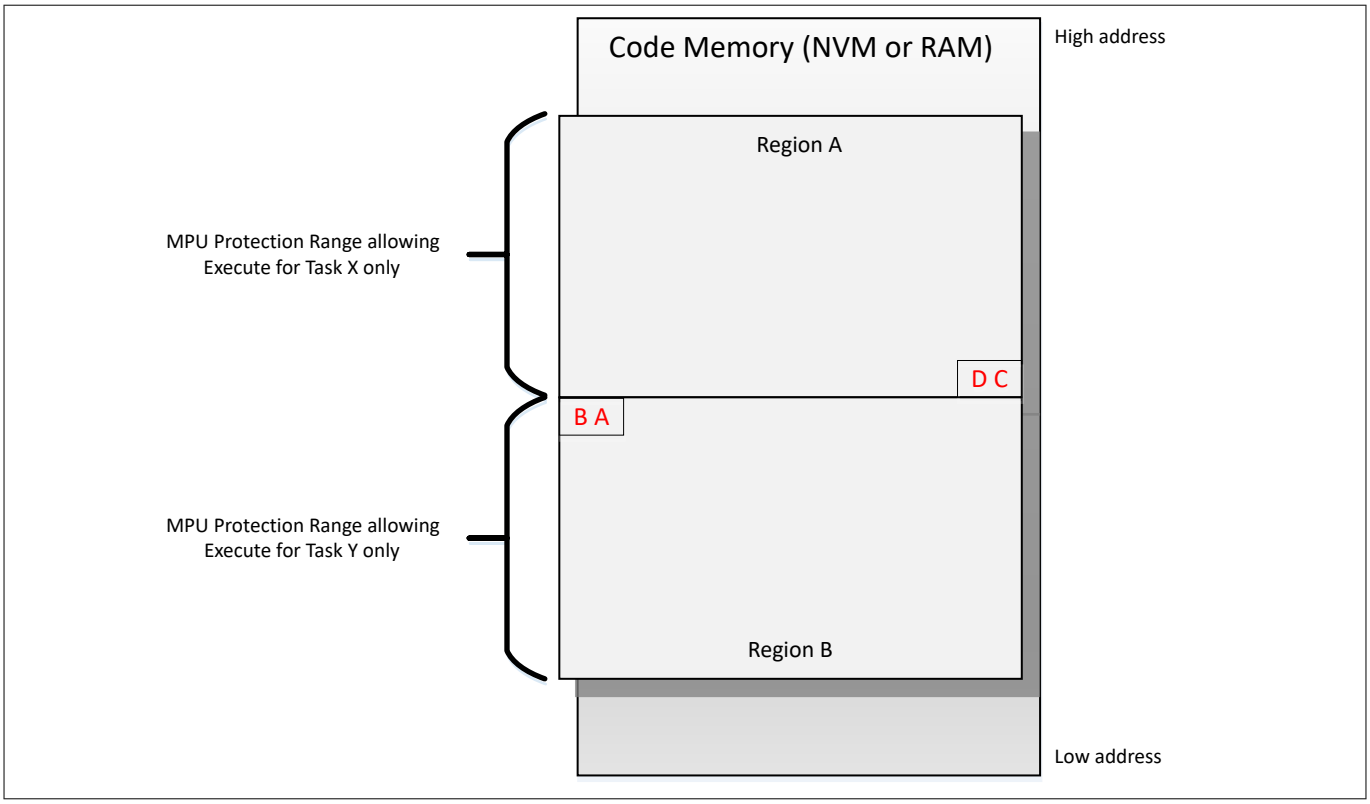


Figure 23 Crossing protection boundaries

To maintain separation between protection regions the following buffers regions should be placed between each memory protection region.

Code protection regions

- No buffer is required

Level 1 data protection

General case:

- Align protection boundaries to 64 bytes
- Minimum 12-byte[4)] buffer between "allowed" and "not allowed" data areas

Special case for regions not containing STLCX or STUCX context data:

- Minimum 12-byte[4)] buffer between "allowed" and "not allowed" data areas

***Note**: "Data areas" are not the same as “protection ranges”*

Virtualization - Level 2 data protection regions (for devices with CPU hardware support for virtualization)

The hypervisor must always allow for context operation in guest VMs.

- Align protection boundaries to 64 bytes
- Minimum 12-byte[4)] buffer between "allowed" and "not allowed" data areas

4 The gap is derived from the alignment rules for load/store instructions. The worst case is a word aligned load or store quad (LD.DD or ST.DD) when the first word falls in the first range, while the remaining 12 bytes cross into the next region

4.3.1.15 Memory integrity registers

Registers are provided which allow monitoring and debug of memory integrity errors experienced by code on the CPU and for capturing APU access violations by external masters.

Program integrity error and APU access violation information registers

Two registers allow software to localize the source of the last detected program memory integrity error or the last APU access violation:

- Program Integrity Error Trap Register (PIETR)
- Program Integrity Error Address Register (PIEAR)

These registers are updated when a program integrity error condition, or an APU access violation, is detected and the PIETR.IED bit is zero. On update the PIETR.IED bit is set to one and remains set until cleared by software. Whilst PIETR.IED is set further hardware updates of PIETR and PIEAR are inhibited. The various error scenarios are defined below.

Program integrity errors occurring on a program instruction fetch from local program memory

When a memory integrity error is detected during a local program fetch from a program memory, the IE_S, IE_C, IE_T, IE_LPB bits of PIETR are updated to denote in which memory structure the error was detected. If the IE_C bit is set the E_INFO field will be updated with the cache way. The PIEAR register is updated with the address of the access. If the error detected is an uncorrectable bit error the IE_UNC bit is set. The IED bit is set and all other PIETR register bits are set to zero. Since instruction fetches are speculative, the PIETR and PIEAR registers may be updated without a corresponding PIE trap.

Program integrity errors occurring on an access to program memory by an external master

When a memory integrity error is detected during an external read from program scratchpad memory, the IE_S, IE_C bits are updated to denote in which memory structure the error was detected. The IE_BS is set and E_INFO updated. The PIEAR register is updated with the address of the access. If the error detected is an uncorrectable error the IE_UNC bit is set. The IED bit is set and all other PIETR register bits are set to zero. Note that a memory read can be generated by a sub-word write operation. The IE_C bit can only be set if the cache is mapped into memory in SIST mode. In this case the E_INFO field indicates the tag of the requesting master.

PIExR register updates due to APU access violations

When an APU access violation is detected during an access to the CPU registers, the IE_SP, IE_BS bits are set and the E_INFO field of PIETR is updated. The E_VMN and E_VMN_EN fields are updated if VMN_EN is set or cleared otherwise. The E_PRS and E_PRS_EN fields are updated if PRS_EN is set or cleared otherwise. The PIEAR register is updated with the address of the access. The IED bit is set and all other PIETR register bits are set to zero. If virtualization is present and enabled only HRHV_PIETR and HRHV_PIEAR will be updated.

Program integrity errors caused by interconnect EDC errors

For a program fetch which accesses the interconnect and experiences an EDC error the IE_BI bit is set and the IE_BS bit is clear. The PIEAR register is updated with the address of the access.

For read by an external master which is received with an EDC error the IE_BI bit is set and the IE_BS bit is set and the E_INFO field is updated. The PIEAR register is updated with the address of the access.

For uncorrectable errors, IE_UNC bit is also set.

For an access by an external master which is received with an address phase EDC error, the IE_ADDR and IED bits are set, all other bits are cleared.

Program integrity error - memory test mode violation

The PIETR.IE_MTMV bit is set if the cache controller attempt to access the PSPR and PCache or PTAG memories while they are in a test mode.

Data integrity error and APU access violation information registers

Two architecturally visible registers allow software to localize the source of the last detected uncorrectable data memory integrity error or the last APU access violation:

- Data Integrity Error Trap Register (DIETR)
- Data Integrity Error Address Register (DIEAR)

These registers are updated when an uncorrectable data integrity error condition, or an APU access violation, is detected and the DIETR.IED bit is zero. On update the DIETR.IED bit is set to one and remains set until cleared by software. While DIETR.IED is set, further hardware updates of DIETR and DIEAR are inhibited. The various error scenarios are defined below.

Data integrity errors occurring on a load from local data memory

When a memory integrity error is detected during a load from a local data memory, the IE_S, IE_C, IE_T and IE_LPB, IE_DLMU bits of DIETR are updated to denote in which memory structure the error was detected. If the IE_C bit is set the E_INFO field will be updated with the cache way. The DIEAR register is updated with the address of the access. If the error detected is an uncorrectable error, the IE_UNC bit is set. The IED bit is set and all other DIETR register bits are set to zero. Note that a memory read may also be generated by a sub-word write operation or a cache line eviction.

Data integrity errors occurring on an access to data memory by an external access

When a memory integrity error is detected during an external bus read from data memory, the IE_S, IE_C, IE_DLMU bits of DIETR are updated to denote in which memory structure the error was detected. The IE_BS is set and E_INFO updated. The DIEAR register is updated with the address of the access. If the error detected is an uncorrectable error, the IE_UNC bit is set. The IED bit is set and all other DIETR register bits are set to zero. Note that a memory read may also be generated by a sub-word write operation or a cache line eviction. The IE_C bit can only be set if the cache is mapped into memory in SIST mode. In this case the E_INFO field indicates the tag of the requesting master.

DIExR register updates due to APU access violations

When an APU access violation is detected during a data memory access, the IE_SP, IE_BS bits are set and the E_INFO field updated. The E_VMN and E_VMN_EN fields are updated if VMN_EN is set or cleared otherwise. The E_PRS and E_PRS_EN fields are updated if PRS_EN is set or cleared otherwise. The DIEAR register is updated with the address of the access. The IED bit is set and all other DIETR register bits are set to zero.

Data integrity errors caused by interconnect EDC errors

For a local data access which accesses the interconnect and experiences an EDC error the IE_BI bit is set and the IE_BS bit is clear. The DIEAR register is updated with the address of the access.

For read by an external master which is received with an EDC error the IE_BI bit is set and the IE_BS bit is set and the E_INFO field is updated. The DIEAR register is updated with the address of the access.

For uncorrectable errors, IE_UNC bit is also set.

When an ECC error is detected at the data bus interface (either master or slave), the IE_BI bit is set. If the error is during an external access the IE_BS bit is set and the E_INFO field is updated. The DIEAR register is updated with the address of the access. If the error detected is an uncorrectable error the IE_UNC bit is set. The IED bit is set and all other DIETR register bits are set to zero.

Data integrity error - memory test mode violation

The DIETR.IE_MTMV bit is set if the cache controller attempt to access the DSPR and DCache, or DTAG, or DLMU memories while they are in a test mode.

4.3.2 Virtualization

Some instances of the TC1.8 CPU support the TriCore™ real-time virtualization extensions as described in the TriCore™ architecture manual. The presence of virtualization is core specific. The TCCON register can be used to identify cores with virtualization hardware support. Implementation specific details are described in the following sections.

4.3.2.1 Number of supported virtual machines

This section is described in the TriCore™ architecture manual volume 1.

4.3.2.2 Replicated state

Each hardware resource[5)] has a dedicated set of fully replicated CSFR (this does not include SFRs such as Overlay and FSFRs) registers, with the following exceptions:

- The Interrupt Control Register is replicated per virtual machine not per hardware resource. These CSFR registers will be identified by the prefix VM0_, VM1_, VM2_, VM3_, VM4_, VM5_, VM6_, and VM7_
- Read only state is not replicated (that is ID registers)
- CSA managed state is not replicated
- The debug and trace registers are only accessible by the Hypervisor
- The overlay registers are not replicated (SFR)
- The FLASHCON registers are not replicated (FSFR)

Each CSFR register is identified by the prefix HRHV_, HRA_ or HRB_. So FCX for example, is replicated and renamed as HRHV_FCX, HRA_FCX, HRB_FCX.

- Software running in HRHV has access to all registers
- Software running in HRA has access to HRA registers only
- Software running in HRB has access to HRB registers only

Table 50 Replicated state per HR (z = A/B/HV)

Name	Name	Name	Name
HRz_A0	HRz_A1	HRz_A8	HRz_A9
HRz_BIV	HRz_BTV	HRz_CCNT	HRz_CCTRL
HRz_CORECON	HRz_CPRy_U (y=0-15)	HRz_CPRy_L (y=0-15)	HRz_CPXE_y (y=0-8)
HRz_DATR	HRz_DEADD	HRz_DIEAR	HRz_DIETR
HRz_DSTR	HRz_DPRE_y (y=0-8)	HRz_DPRy_U (y-0-23)	HRz_DPRy_L (y=0-23)
HRz_DPWE_y (y=0-7)	HRz_FCX	HRz_FPU_TRAP_CON	HRz_FPU_TRAP_OPC
HRz_FPU_TRAP_PC	HRz_FPU_TRAP_SRCy_U (y=1-3)	HRz_FPU_TRAP_SRCy_L (y=1-3)	HRz_ICNT
HRz_ISP	HRz_LCX	HRz_MnCNT (n=1-3)	HRz_PIEAR
HRz_PIETR	HRz_PSTR	HRz_FPU_SYNC_TRAP_CON	HRz_FPU_SYNC_TRAP_OPC
HRz_TPS_CON	HRz_TPS_TIMERn (n=0-2)	-	-

5 A hardware resource represents a collection of architectural and software state dedicated to a given software partition

Table 51 Non-replicated state accessible to guest VM

Name	Name	Name	Name
A2-A7, A10-A15 (AGPRs)	BOOTCON	CCON	COMPAT
CORE_ID	CPU_ID	CUS_ID	D0-D15 (DGPRs)
DCON0	DCON1	DCON2	ICR
LCLCON	LCLTEST	PC	PCON0
PCON1	PCON2	PCXI	PMA0
PMA1	PMA2	PPRS	PSW
SMACON	SEGEN	SWID	TCCON

Table 52 Non-replicated debug and trace

Name	Name	Name	Name
DBGACT	DBGCFG	DBGSR	DBGTCR
CREVT	DCX	DMS	EXEVT
SWEVT	TRCCFG	TRCFILT	TRCLIM
TRIG_ACC	TRnADR (n=0-7)	TRnEVT (n=0-7)	TS16PTCCTRL

Table 53 HV only state

Name	Name	Name	Name
BHV	VCON0	VCON1	VCON2
VMn_ICR[1)]	VMn_PETHRESH[1)]	-	-

1) Replicated per VM (n=0..7)

Non-replicated registers are aliased in each of the hardware resources.

4.3.2.3 CSFR access from the local core

Access to CSFR state is performed from the local core through the MTCR or MFCR instructions, and the MTDCR or MFDCR instructions. These instructions use a 16-bit constant for their access address.

For MTDCR and MFDCR operations const[2] should always be “0” otherwise the operation is accessing a misaligned location and will lead to an OPD trap.

For version 1.8 of the TriCore™ architecture, offset bits [1:0] are used by VM0 to access CSFRs in its own or other hardware resources. See the TriCore™ architecture manual for more details.

A MTCR or MFCR operation in VM1-VM7 that targets an undefined CSFR location or a MTDCR or MFDCR operation in VM1-VM7 that targets an undefined double CSFR register location will result in a HV trap (Class 4 HVCSFR) if virtualization is configured and enabled. In all other cases an OPD trap is raised.

Table 54 CSFR access hardware resource selection rules from the local core

Currently active hardware resource	Const[1:0] == 00_B	Const[1:0] == 01_B	Const[1:0] == 10_B	Const[1:0] == 11_B
HRHV	Access to own CSFR register set	Access to HRA CSFR register set[1]	Access to HRB CSFR register set[2]	Misaligned CSFR access, OPD trap
HRA		Misaligned CSFR access, OPD Trap		
HRB				

1) The define CPU_HVHRA_<regname> short address for CSFR domain CSFR is provided for that purpose
2) The define CPU_HVHRB_<regname> short address for CSFR domain CSFR is provided for that purpose

Table 55 CSFR access exceptions for local core

	Priority (0:low, 5: high)	0	1	2	3	4
Virtualization enable	**Executing hardware resource**	**Target CSFR is undefined**	**Target is other HR**	**Target is misaligned[1]**	**PROTSFRE Protected[2]**	**Target is GPR**
1	VM0	OPD Trap	OK	OPD Trap	OPD Trap	OPD Trap[3]
1	Guest VM	HV Trap	OPD Trap	OPD Trap	HV Trap	OPD Trap
0	N/A	OPD Trap	N/A	OPD Trap	OPD Trap	OPD Trap

1) With regards to the operand granularity
2) Protection applies to writes to own space
3) Excludes HRA/B global registers

Related information

Non Replicated CSFRs supported through HV traps on page 126

4.3.2.3.1 Global address registers and GPRs

The global address registers (A0, A1, A8, and A9) are not saved and restored as part of the context save area. They therefore are replicated per hardware resource.

Accessing GPRs through the MTCR, MFCR, MTDCR, or MFDCR instructions is an illegal operation in the TriCore™ architecture. This remains the case for accesses to the GPRs of the currently executing VM. The hypervisor may access the Global Address GPRs (A0, A1, A8, or A9) of HRA, HRB using MTCR, MFCR, MTDCR, or MFDCR.

Related information

Instructions that are new for version 1.8 of the TriCore architecture on page 136
Boot operation on page 131

4.3.2.3.2 Hypervisor register offsets

SFR	HRA	HRB	HRHV	Offset	Description
			X	$3B000_H$	VCON0
			X	$3B004_H$	VCON1

			X	$3B008_H$	VCON2
			X	$3B010_H$	BHV
			X	$3B100_H + x*4_H$	VMn_ICR
			X	$3B200_H + x*4_H$	VMn_PETHRESH

4.3.2.3.3 Non Replicated CSFRs supported through HV traps

Some VM state needs to be under control of the hypervisor. To allow this, Guest VM accesses (read or write) to the following CSFRs will cause a trap to the hypervisor (Class 4 - HVCSFR).

Table 56 CSFRs requiring HV support

Register name	Function	Address
SEGEN	SRI Error Generation Control	1030_H
LCLTEST	Lockstep Test Register	1040_H
COMPAT	Compatibility Register	9400_H
All debug registers	All debug registers	Ranges $F000_H$ and $FD00_H$ (performance counters are replicated per HR and are decoupled from debug and therefore do not fall in this category)
PMA0/ PMA1/ PMA2	Physical memory attributes	8100_H / 8104_H / 8108_H
DCON0	Data Control Register 0[1)]	9040_H
DCON2	Data Control Register 2	9000_H
PCON0	Program Control Register 0[1)]	$920C_H$
PCON2	Program Control Register 2	9208_H
LCLCON	Lockstep Control Register	$FE64_H$
CCON	Clock Control Register	$FE68_H$
All undefined register locations[2)]		

1) Guest VMs requiring different cache configurations must be handled in the hypervisor software
2) Details of operation when virtualization is not configured or is configured but not enabled can be found in the section titled "CSFR access from the local core".

Related information

CSFR access from the local core on page 124

4.3.2.3.4 MTCR, MTDCR targeting HRA/HRB CSFR's

To support the Hypervisor in restoring the complete state of a non-resident VM, the operation of MTCR and MTDCR executed in the Hypervisor context will override hardware writeable fields (normally read only for the application) when the target CSFR belongs to either HRA or HRB.

A Hypervisor targeting the guest VM hardware resources of a different TC1.8 CPU (HRA or HRB CSFR's) can also override the hardware writeable fields of the target CPU if the access passes the target CPU APU checks.

***Note**: A guest VM operation targeting another CPU would be to only update the softwarewriteable fields of the target CPU.*

Related information

Instruction operation updates on page 136

4.3.2.3.5 MTCR/MFCR, MTDCR/MFDCR to non-existent or protected locations

A MTCR or MFCR operation by a guest VM that targets a non-existent CSFR location, or a MTDCR or MFDCR operation by a guest VM that targets a non-existent double CSFR register location will result in a HV trap (Class 4 HVCSFR).

A MTDCR to a non-existent register in either (not both) locations will be ignored.

A MFDCR from a non-existent register in either (not both) location will return 00000000_H for that location.

In all other cases (VM0 execution or virtualization not configured or enabled) an OPD trap is raised.

A MTCR operation by a guest VM that targets a CSFR location protected by PROTSFRE or PROTSFRSE will result in a HV trap (Class 4 HVCSFR).

A MTDCR operation by a guest VM that targets a double CSFR location where either of the CSFRs is protected by PROTSFRE or PROTSFRSE will result in a HV trap (Class 4 HVCSFR).

In all other cases (VM0 execution or virtualization not configured or enabled) an OPD trap is raised.

Related information

Instruction operation updates on page 136

4.3.2.3.6 PROT usage with virtualization present and enabled

The HV needs to be able to change the state of a guest VM's protected register during a VM swap without the time overhead of an unlock or lock operation. The HV is therefore allowed write access to all HRA, and HRB CSFRs without requiring a PROTSFRE unlock or lock sequence.

- Accesses to HRHV CSFRs and SFRs are protected by PROTSFRE and PROTSFRSE. A move to a HRHV CSFR that is protected will generate an OPD trap
- In the case where virtualization is not configured or not enabled, endinit like protection is maintained for the relevant registers. In this case a move to a CSFR that is protected will generate an OPD trap
- MTCR/MTDCR to protected CSFRs in a guest VM will generate a HV trap (Class 4 HVCSFR)

Table 57 PROT for configurations with virtualization present and enabled

Instance	Registers covered	Notes
PROTRSTE	KRST0, KRST1 (No APU)	Previously system ENDINIT. Separated from other PROT as the owner is likely to be a remote master
PROTSFRE	PMA0, PMA1, DCON0, PCON0, SEGEN	Previously CPU ENDINIT
	STM_CLC (No APU)	Previously system ENDINIT STM is now part of CPU
	HRHV_BIV, HRHV_BTV, HRHV_ISP, HRHV_BHV	Provides protection for the HV
PROTSFRSE	COMPAT, SMACON, HRn_CORECON, ACCENSFRCFG (APU)	Previously safety ENDINIT HV required
PROTSTMSE	ACCENSTMCFG (APU)	Previously safety ENDINIT HV required to configure APU
PROTDLMUSE	ACCENDLMUCFG (APU)	Previously safety ENDINIT

(table continues...)

Table 57 (continued) PROT for configurations with virtualization present and enabled

Instance	Registers covered	Notes
PROTSPRSE	ACCENSPRCFG (APU)	Previously safety ENDINIT
PROTPFSFRE	PFIy_FLASHCON0, PFIy_FLASHCON1 (No APU)	Previously system ENDINIT
PROTPFSFRSE	ACCENPFSFRCFG (APU)	Previously safety ENDINIT
PROTPFSE	ACCENPFCFG (APU)	Previously safety ENDINIT

4.3.2.4 Hypervisor traps

This section is described in the TriCore™ architecture manual volume 1.

4.3.2.5 Interrupts for the hypervisor

This section is described in the TriCore™ architecture manual volume 1.

4.3.2.5.1 Hypervisor context examples

This section is described in the TriCore™ architecture manual volume 1.

4.3.2.6 Interrupts

4.3.2.6.1 Interrupt interface

Each Interrupt SRN is extended to store a Virtual Machine Number (VMN) 0-7 alongside the TOS and SRPN. The SRN requests an interrupt at priority SRPN on the Service provider identified by the TOS and on the virtual machine identified by the VMN.

The VM value is presented to the processor alongside the interrupt request and PIPN. This allows incoming interrupts to be directed to the correct machine.

Each virtual machine has a separate ICR register and so a separate PIPN, IE, and CCPN. The interrupt state of each VM is independent of the other VMs.

The Interrupt router provides an indication of the presence or not of a valid interrupt request for a given VM. This information will be used by the CPU to update the PIPN for the VM presented. When the request is not valid, VMx_ICR.PIPN will be set to 0.

4.3.2.6.2 Interrupt pre-emption by other VMs

This section is described in the TriCore™ architecture manual volume 1.

4.3.2.6.3 Interrupt flow diagrams

The PIPN and VMN are presented to the CPU and compared against the CCPN and IE value for that machine in the standard way. In addition a pre-emption check will be performed if the requested VM is not currently running.

If the incoming interrupt is not for the currently running VM then a pre-emption check is performed to determine whether a VM switch should be performed.

If the pre-emption check passes then the Hypervisor will be called to perform a VM switch.

In the case where hardware has decided to “keep pending” the interrupt will simply be left pending. This situation will continue until the hardware state changes to a state in which the interrupt can be taken. This

significantly simplifies the hardware in that there is no need for mechanisms to allow the interrupt to be deferred to a later point in time.

For a taken interrupt, if the currently running machine is the same as the machine requested by the interrupt then the interrupt will be passed directly to the running machine.

In all cases a software swap of a virtual machine to or from memory is only required if the VM state is not currently held in one of the hardware resources sets. If the state is already held in one of the HR sets then a hardware switch is all that is required.

Use cases

In the case where there is a single executing VM (VM1) interrupts will be taken in VM1 unless targeted at the hypervisor (VM0). In this case a switch to the hypervisor will only be initiated if the incoming interrupt is above the preemption threshold of the hypervisor and if the hypervisor is in a state to receive interrupts.

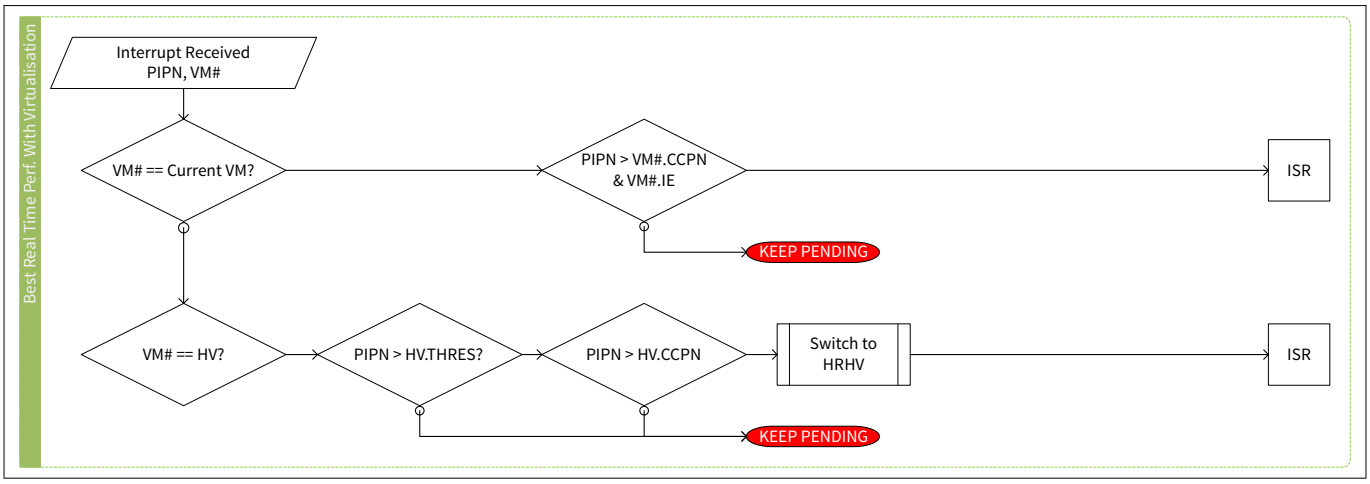


Figure 24 Interrupt flow diagram for 1 hypervisor and 1 real-time virtual machine

In the case where there are two active VMs (VM1, 2) interrupts targeted at the currently executing VM will be taken in that VM. Interrupts targeted at the other VM or the hypervisor (VM0) will initiate a switch to the target machine only if the incoming interrupt is above the preemption threshold of the target VM or hypervisor, and if that VM or hypervisor is in a state to receive interrupts.

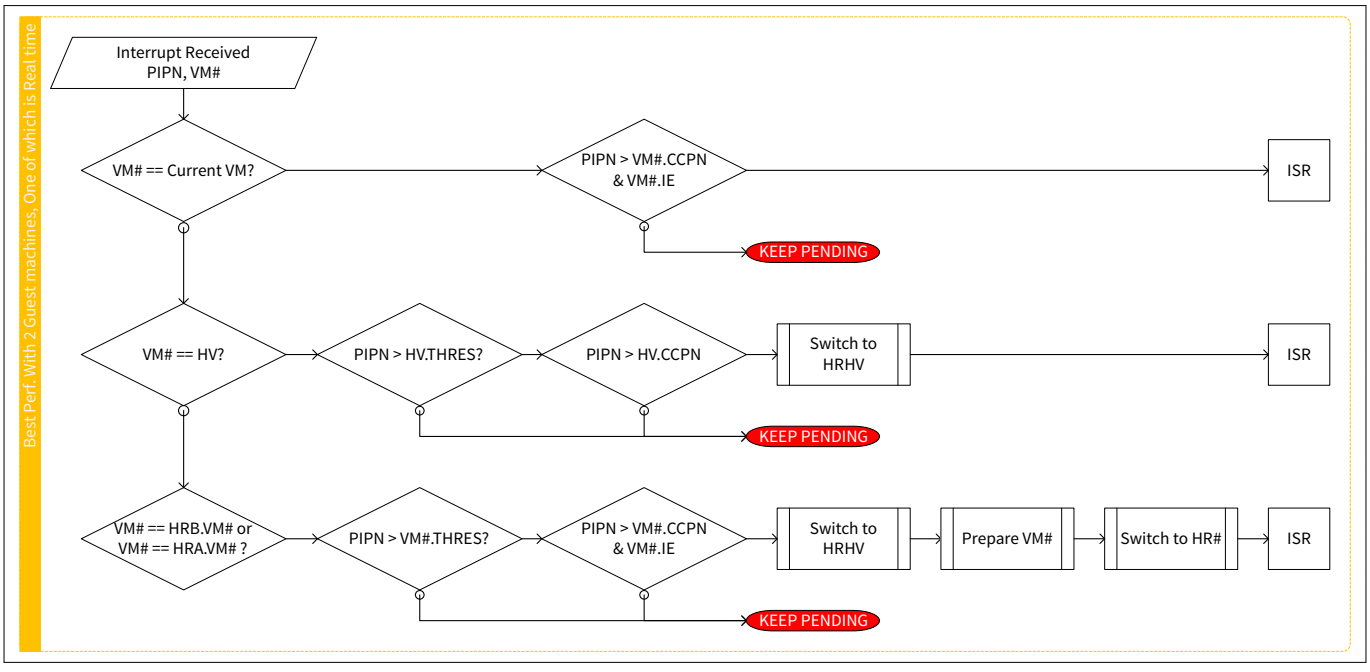


Figure 25 Interrupt flow diagram for 1 hypervisor and 2 resident virtual machines

When more than 2 VMs are required, the hypervisor software will be required to swap the content of hardware resource 2 to/from memory in order to place the targeted virtual machine in the hardware resource prior to the execution of the RFH instruction.

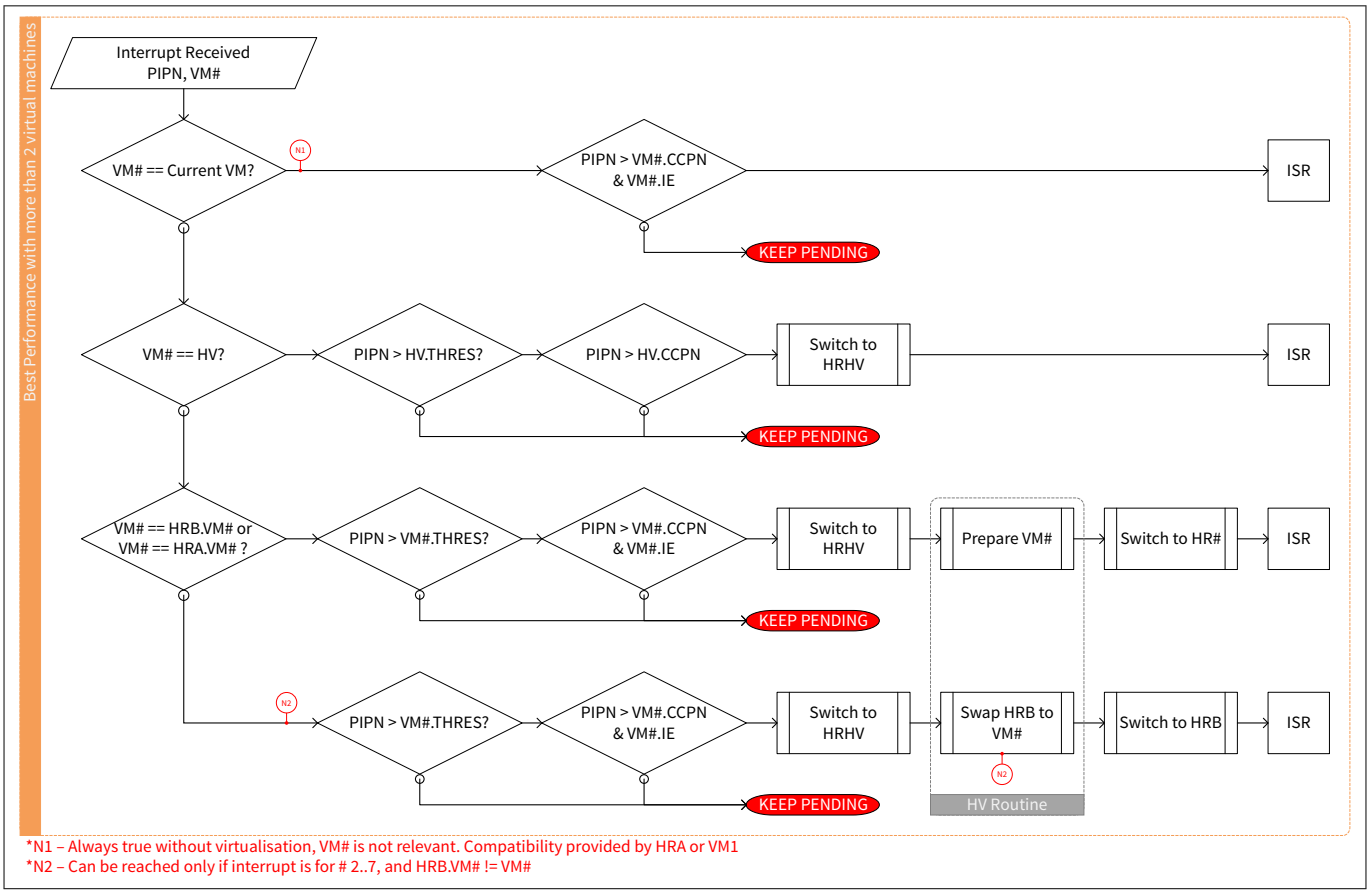


Figure 26 Interrupt flow diagram – 3 to 7 virtual machines

4.3.2.6.4 Interrupt corner cases

Configuration of the IR for each specific CPU should match the managed number of VMs. However, if there is a mismatch between the IR configuration and the use of virtualization or the number of supported guest VMs then this can be dealt with by SW.

Virtualization enabled and presentation of an interrupt to a non-enabled VM. There are two logical options that a system must choose:

- All unused VMs should have interrupts disabled. An interrupt to such a VM will be ignored by the HW
- All unused VMs should have interrupts enabled and their priority and threshold set to 0. An interrupt to such a VM will result in a HV switch. The HV would then flag this as a detected error

Virtualization disabled and presentation of an interrupt to a VM:

- The CPU will ignore the VMN field of the incoming interrupt and take the interrupt due to the standard ICR.IE and PIPN > CCPN tests

4.3.2.6.5 Hypervisor interrupt entry sequence

This section is described in the TriCore™ architecture manual volume 1.

4.3.2.6.6 Interrupt ack and interrupt enter

Enter

An interrupt enter should be sent to the IR once the conditions are met for that interrupt to be taken. For interrupts to the current VM this means passing the PIPN > CCPN check. For interrupts to the hypervisor or to other virtual machines, enter will be raised before switching to HRHV.

Acknowledge

Interrupt acknowledge should be sent before executing the first instruction of the ISR. This corresponds to the “Take Interrupt” box in the interrupt diagrams.

4.3.2.7 Virtualization instructions

Hypervisor call (HVCALL) and return (RFH) instructions are required.

Cache instructions which are not constrained by the virtual machine number are also required.

4.3.2.8 Boot operation

In version 1.8 of the TriCore™ architecture, the BHALT bit is available in a dedicated CSFR BOOTCON. BHALT. (Bit0)

The required boot sequence varies depending on whether the virtualization extension is present or not (see TCCON).

CPU instance with hardware virtualization

At boot the TC1.8 CPU starts in VM0/Hypervisor with virtualization enabled. All registers in all hardware resources, are visible and are accessible. This allows boot code to initialize the HV state along with other processor state as required.

A write to VCON0.VEN is required to set the virtualization operation of the core. This should be done prior to executing the first RFH instruction. This is not enforced in hardware but execution of a RFH prior to the first VCON0.VEN write is an illegal operation from an architectural viewpoint leading to undefined results.

CPU instance without hardware virtualization

A CPU that is configured without hardware virtualization will boot directly into VM1 with all virtualization extensions unavailable.

4.3.2.9 Disabling hypervisor

This section is described in the TriCore™ architecture manual volume 1.

4.3.2.10 Memory protection

HRHV, HRA, and HRB each have replicated memory protection sets with the following configurations.

Table 58 Number of memory protection ranges and sets

Hardware resource	Data protection ranges	Code protection ranges	Protection sets
HRHV	24	16	8
HRA	24	16	8
HRB	24	16	8

4.3.2.11 Protection state on entering hypervisor

On any entry to the hypervisor the PSW.PRS is set to 0.

Protection is enabled if HRHV_CORECON.PROTEN ==1.

4.3.2.12 Watchdogs

If the run-time monitoring function of the watchdog (WDT) functionality is required by one VM (VM0/HV or VM1) then it can be directly allocated. If WDT functionality is required by more than one guest VM (that is VM1..VM7) then the HV must provide the Program Flow Monitoring features through para-virtualization.

4.3.2.13 NMI virtualization

This section is described in the TriCore™ architecture manual volume 1.

4.3.2.14 Traps

This section is described in the TriCore™ architecture manual volume 1.

4.3.2.15 Asynchronous traps

For all asynchronous traps taken in HV, to allow software to determine whether an RFE or RFH should be used to terminate the exception handler, bit D15[31] is set whenever the trap entry requires a switch of VM. A simple conditional jump can be used to select the operation required to complete the trap handler.

```
        // return from handler
        JZ.T D15, 31, rfe_target
        RFH;
rfe_target:
        RFE;
```

DAE - data access asynchronous error (TIN 3)

The Data Access Asynchronous Error Trap (DAE) is generated by the DMI module when a store or cache management access from the CPU encounters certain error conditions, such as a Bus error. Since the source of the DAE trap is not directly linked to the current instruction, for example cache eviction, or even to the current VM, a DAE will lead to a switch to the Hypervisor.

When a DAE trap is generated, the exact cause of the error can be determined by reading the Data Asynchronous Trap Register, HRA_DATR. For the TC1.8 CPU HRA_DATR is extended to capture VMN, VMN_EN, PRS, PRS_EN and the value of the S bit used for the transaction.

DIE - data memory integrity error (TIN 6)

The DIE trap is raised whenever an uncorrectable memory integrity error is detected in a data access to a local memory or the SRI. The trap is of Class-4 and has a TIN of 6.

DIE traps are always asynchronous independent of the operation which encountered the error.

A DIE trap is raised if any memory half-word (local memory) or double-word (SRI) accessed by a load or store operation contains an uncorrectable error and no other DIE trap have been raised since the clearing of the DIETR.IED bit. The DIEAR and DIETR registers may be interrogated to determine the source of any error more precisely. Subsequent DIE traps are inhibited if DIETR.IED is set.

DIE will lead to a switch to the Hypervisor. For the TC1.8 CPU DIETR is extended to capture VMN, VMN_EN, PRS, and PRS_EN used for the transaction..

CAE - coprocessor asynchronous error (TIN 4)

Coprocessor errors relate to the software execution and are taken in the VM that triggered them. If the exception arises while the VM is being switched, the exception will remain pending until the hardware returns to that VM.

TAE - temporal asynchronous error (TIN 7)

Temporal traps guard against task run-time overrun and are taken in the VM that triggered them. If the exception arises while the VM is being switched, the exception will remain pending until the hardware returns to that VM.

4.3.2.16 Context save area (CSA) protection

In both version 1.8 of the TriCore™ architecture and earlier TriCore™ architectures, the memory accesses generated by CSA save and restore operations do not undergo level 1 protection checks. In version 1.8 of the TriCore™ architecture all Guest VM accesses, including CSA save and restore operations, are checked by the level 2 protection checks (L2 MPU).

When a guest VM is running the L2 MPU will check the base address accesses generated by CSA save and restore operations. If this address is outside of the enabled protection ranges a hypervisor trap will be raised. Since only the base address is being checked it is possible for an access to start at the top of an enabled region and to extend into a non-enabled region without triggering a trap condition. To maintain separation between protection regions of the L2 MPU, each data protection region should be aligned to 64-bytes and a buffer register of 12-bytes should be placed between data regions with different level of protection or different guest VMs. Due to the above CSA handling, it is not necessary for the HV to check the specific values of the CSA pointers within a VM.

CSA save and restore operations performed by the hypervisor do not undergo protection checks.

L2 MPU traps for CSA are synchronous to the event that triggered the error. When an L2 memory protection violation due to a CSA access, HRHV_DSTR will be updated.

4.3.2.17 Virtualization and caches

Cache bypass

Caches are bypassed after reset. Enabling of the caches is done through the DCON0 and PCON0 CSFRs.

Guest VM access (read or write) to DCON0 and PCON0 will cause a trap to the hypervisor to allow consistent policy control of these shared resources.

Cache write-back

When a cacheline write-back is required, the line is presented to the interconnect with the same VM information that allocated the line in the cache.

Cache invalidation

Cache invalidation of clean cache lines can be achieved in an efficient way using the DCON1 and PCON1 CSFRs. Since caches are shared and eviction of clean data related to a non-running VM, or to a different PRS, will naturally happen while executing another VM, no restriction is added for access, read and write, to DCON1 and PCON1 registers. These can be accessed by any VM.

CACHEA.*, CACHEI.* instructions

When operating with virtualization enabled these instructions are to invalidating cache entries forthe virtual machine defined in VCON2.VMN. This allows the HV to selectively operate on a particular VM's cachelines if re quired. This also ensures that a VM can only operate on its own entries.

CACHEA.*.VM, CACHEI.*.VM instructions

New versions of the CACHEA.*.VM and CACHEI.*.VM instructions are introduced to provide a mechanism for the hypervisor to operate on lines from all VMs.

***Note**: These instructions operate on any valid data tag entry regardless of PSW.PRS, and PSW.S. bit.*

Related information

Instruction operation updates on page 136
Cache architecture on page 151

4.3.2.18 Virtualization and overlay

Virtualization provides two levels of MPU within the CPU. Overlay address modification only happens after the address has passed both MPU checks. The APU on all overlay memories is a final check which redirected accesses must pass to succeed.

The overlay mechanism does not restrict itself to a VM. If the NVM region is overlayed and allowed by MPU checks the overlay address substitution occurs. The destination APU check must still be passed by the access. Overlay is implemented using SFRs which are not replicated. Overlay is expected to be used for a whole system (or not). If finer granularity is required then it must be provided through some software layer.

4.3.2.19 System timer extensions for virtualization

To support virtualization it is required that interrupts can be generated per VM based on either Absolute Time or Relative time. To support the following changes are made to the STM:

- 64-bit relative time is implemented per VM, clocked from the same source as the existing timer (STM_RELTIM[63:0]) to hold the relative time value of the machine when it is not currently active:
  - This timer is loaded/unloaded on a VM switch/swap
  - This timer should be readable in the same way as the existing timer
- The compare registers (STM_CMP0, STM_CMP1) are replicated per VM:
  - STM_CMPc
- The compare match control register is duplicated per VM:
  - STM_CMCON
- Each comparator may be selected to compare against either the absolute or relative time:
  - New RELCOMP bits added to STM_CMCON [31,15]
- The interrupt control registers are replicated per VM:
  - STM_ICR
- The interrupt ISCR registers are replicated per VM:
  - STM_ISCR
- The interrupt ISR registers are replicated per VM:
  - STM_ISR
- The support of per VM system timers means an increase in the number of timer interrupts per core to 16
- A guest VM's access to the STM register is independent of the VM's number:
  - This is managed by the CPU redirecting guest VM STM accesses to the relevant STM registers
  - Hypervisor software can address any VM specific STM register

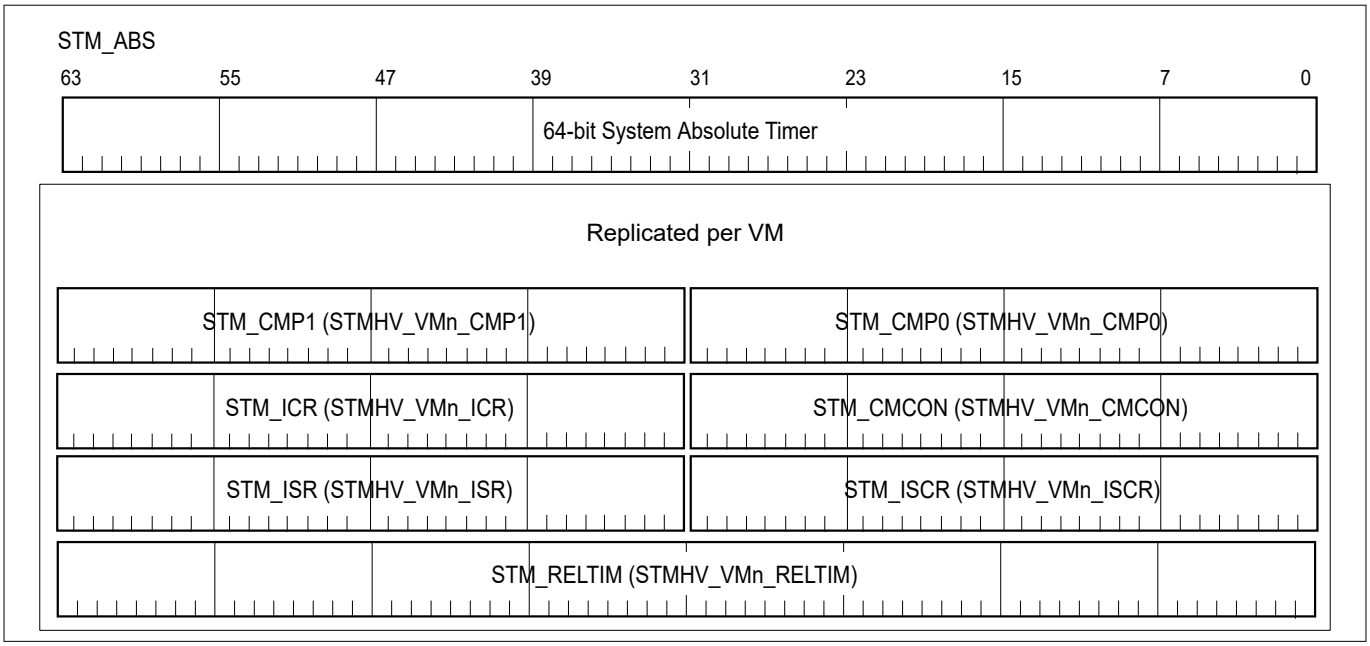


Figure 27 STM registers

4.3.2.20 STM register access (access virtualization)

The STM registers are located in the CPU SFR address space. To minimize the need for para-virtualization of the STM function, all guest virtual machines access their STM instance at the same offset. The same offset is also used if virtualization is disabled or not present. The hardware will then compute the offset corresponding to the requesting guest VM prior to performing the access. The access virtualization only applies to guest virtual machines and not to the hypervisor which will be able to access any of the virtual STM instances at their actual offset.

```
STMHV_VMx: 0xF8800000 + (CORE_ID << 18) + STM register offset
```

Guest VM example: CPU0 VM2 STM_CMP0 will be accessed in software at address 0xF8800120 (using the CPU0_STM_CMP0 address) which will result in an access at 0xF8800140 on the system interconnect (CPU0_STMHV_VM2_CMP0 address).

Hypervisor example: CPU0 VM0 STM_CMP0 must be accessed by the hypervisor software using CPU0_STMHV_VM0_CMP0 address.

Other masters attempting to access a CPU STM register must use the specific address corresponding to the targeted VM.

To provide isolation one additional PROT and 8 APUs are provided for the STM functionality (see section STM protection).

4.3.2.21 Temporal protection system timers (TPS) extensions for virtualization

The temporal protection system is replicated per hardware resource, and HRz_TPS_TIMERn only decrement while that hardware resource is active.

4.3.2.22 Para-virtualization and integrators tasks

This section details the modifications required to software to allow it to run as a virtual machine on the TC1.8 CPU.

4.3.3 Instructions that are new for version 1.8 of the TriCore™ architecture

The following instructions are new in version 1.8 of the TriCore™ architecture compared to version 1.6, and will not be available on CPUs implementing earlier architectures.

New instructions (see TriCore™ architecture manual for details):

- Hypervisor call and return from hypervisor (HVCALL, RFH)
- Load/Store quad data word (LD.DD, ST.DD)
- Move to or from core register pair (MTDCR, MFDCR)
- Carry-less multiplication (MULP.B)
- Call indirect variant (CALLI 16-bit)
- Jump relative indirect (JRI)
- 64-bit integer divide and remainder (DIV64, DIV64.U, REM64, REM64.U)
- Local data synchronization (LSYNC)
- New single-precision floating-point instructions (ABS.F, MAX.F, MIN.F, NEG.F)
- New double-precision floating-point coprocessor (ABS.DF, ADD.DF, CMP.DF, DFTOF, DFTOI, DFTOIN, DFTOIZ, DFTOU, DFTOUZ, DFTOL, DFTOLZ, DFTOUL, DFTOULZ, FTODF, ITODF, LTODF, MADD.DF, MAX.DF, MIN.DF, MSUB.DF, MUL.DF, NEG.DF, QSEED.DF, SUB.DF, ULTODF, UTODF)

4.3.4 Instruction operation updates

The operation of some instructions has been updated compared to previous implementations.

Related information

Virtualization and caches on page 133
MTCR, MTDCR targeting HRA/HRB CSFR's on page 126
MTCR/MFCR, MTDCR/MFDCR to non-existent or protected locations on page 127

4.3.4.1 MSUB.Q

AURIX™ TC3xx implementations of the MSUB.Q instruction favored a slightly more accurate result during internal rounding. For better bit compatibility, the MSUB instruction in AURIX™ TC4xx returns the same result as a MUL.Q followed by a SUB.

4.3.4.2 CACHEI.* instruction input operands

The CACHEI.* instructions require a 'way' and an index value to be supplied in a valid address, where a valid address is an address that will pass memory protection and null pointer checks. The CACHEI.* instructions also require an address to be cacheable. The local portion of DLMU (even when accessed through segment 9_H) is not cacheable. The local DSPR's global address (segment $[7\text{-CORE_ID}]_H$, that is 0x700xxxxx, 0x600xxxxx and so on) is not cacheable either.

The location of these bits in the 32-bit address is as follows.

Table 59 Way, and index location

Function	Address bits
Way	[0]
Index	[12:5]

4.3.5 CPU instruction timing

This section gives information on CPU instruction timing by execution units.

Common timing assumptions:

- All targets yield a full instruction in one access. That is, not 16-bit of a 32-bit instruction
- All code fetches take a single cycle

Definition of terms:

Repeat rate: Assuming the same instruction is being issued sequentially, repeat is the minimum number of clock cycles between two consecutive issues. There may be additional delays described elsewhere due to internal pipeline effects when issuing a different subsequent instruction.

Result latency: The number of clock cycles from the cycle when the instruction is issued to the cycle when the result value is available to be used as an operand to a subsequently issued instruction or written into a GPR. Result latency is not meaningful for instructions that do not write a value into a GPR.

Address latency: The number of clock cycles from the cycle when the instruction is issued to the cycle when the addressing mode updated value is available as an operand to a subsequently issued instruction or written into an Address Register.

Flow latency: The number of clock cycles from the cycle when the instruction is issued to the cycle when the next instruction (located at the target location or the next sequential instruction if the control change is conditional) is issued.

4.3.5.1 Integer pipeline instructions

These tables define the TC1.8 CPU integer pipeline instruction timings.

Simple arithmetic instruction timings

Table 60 Simple arithmetic instruction timing: Arithmetic instructions

Instruction	Result latency	Repeat rate	Instruction	Result latency	Repeat rate
ABS	1	1	MAX.H	1	1
ABS.B	1	1	MAX.HU	1	1
ABS.H	1	1	MAX.U	1	1
ABSDIF	1	1	MIN	1	1
ABSDIF.B	1	1	MIN.B	1	1
ABSDIF.H	1	1	MIN.BU	1	1
ABSDIFS	2	1	MIN.H	1	1
ABSDIFS.H	2	1	MIN.HU	1	1
ABSS	2	1	MIN.U	1	1
ABSS.H	2	1	RSUB	1	1

(table continues...)

Table 60 (continued) Simple arithmetic instruction timing: Arithmetic instructions

Instruction	Result latency	Repeat rate	Instruction	Result latency	Repeat rate
ADD	1	1	RSUBS	2	1
ADD.B	1	1	RSUBS.U	2	1
ADD.H	1	1	SAT.B	1	1
ADDC	1	1	SAT.BU	1	1
ADDI	1	1	SAT.H	1	1
ADDIH	1	1	SAT.HU	1	1
ADDS	2	1	SEL	1	1
ADDS.H	2	1	SELN	1	1
ADDS.HU	2	1	SUB	1	1
ADDS.U	2	1	SUB.B	1	1
ADDX	1	1	SUB.H	1	1
CADD	1	1	SUBC	1	1
CADDN	1	1	SUBS	2	1
CSUB	1	1	SUBS.H	2	1
CSUBN	1	1	SUBS.HU	2	1
MAX	1	1	SUBS.U	2	1
MAX.B	1	1	SUBX	1	1
MAX.BU	1	1	SHUFFLE	2	1
POPCNT.W	2	1			

Table 61 Simple arithmetic instruction timing: Compare instructions

Instruction	Result latency	Repeat rate	Instruction	Result latency	Repeat rate
EQ	1	1	LT.B	1	1
EQ.B	1	1	LT.BU	1	1
EQ.H	1	1	LT.H	1	1
EQ.W	1	1	LT.HU	1	1
EQANY.B	1	1	LT.U	1	1
EQANY.H	1	1	LT.W	1	1
GE	1	1	LT.WU	1	1
GE.U	1	1	NE	1	1
LT	1	1			

Table 62 Simple arithmetic instruction timing: Count instructions

Instruction	Result latency	Repeat rate	Instruction	Result latency	Repeat rate
CLO	1	1	CLS.H	1	1
CLO.H	1	1	CLZ	1	1
CLS	1	1	CLZ.H	1	1

Table 63 Simple arithmetic instruction timing: Extract instructions

Instruction	Result latency	Repeat rate	Instruction	Result latency	Repeat rate
DEXTR	2	1	INS.T	1	1
EXTR	2[1)]	1	INSN.T	1	1
EXTR.U	2[1)]	1	INSERT	2	1
IMASK	2	1			

1) When bit 0 is not included. When bit 0 is part of the extract operation the latency is reduced to 1 cycle

Table 64 Simple arithmetic instruction timing: Logical instructions

Instruction	Result latency	Repeat rate	Instruction	Result latency	Repeat rate
AND	1	1	OR.EQ	1	1
AND.AND.T	1	1	OR.GE	1	1
AND.ANDN.T	1	1	OR.GE.U	1	1
AND.EQ	1	1	OR.LT	1	1
AND.GE	1	1	OR.LT.U	1	1
AND.GE.U	1	1	OR.NE	1	1
AND.LT	1	1	OR.NOR.T	1	1
AND.LT.U	1	1	OR.OR.T	1	1
AND.NE	1	1	OR.T	1	1
AND.NOR.T	1	1	ORN	1	1
AND.OR.T	1	1	ORN.T	1	1
AND.T	1	1	XNOR	1	1
ANDN	1	1	XNOR.T	1	1
ANDN.T	1	1	XOR	1	1
NAND	1	1	XOR.EQ	1	1
NAND.T	1	1	XOR.GE	1	1
NOR	1	1	XOR.GE.U	1	1
NOR.T	1	1	XOR.LT	1	1
OR	1	1	XOR.LT.U	1	1
OR.AND.T	1	1	XOR.NE	1	1
OR.ANDN.T	1	1	XOR.T	1	1

Table 65 Simple arithmetic instruction timing: Move instructions

Instruction	Result latency	Repeat rate	Instruction	Result latency	Repeat rate
CMOV	1	1	MOV.U	1	1
CMOVN	1	1	MOVH	1	1
MOV (32-bit)	1	1	MOV (64-bit)	2	1
MTDCR	-	1	MFDCR	2	1

Table 66 Simple arithmetic instruction timing: Shift instructions

Instruction	Result latency	Repeat rate	Instruction	Result latency	Repeat rate
SH	1	1	SH.NE	1	1
SH.AND.T	1	1	SH.NOR.T	1	1
SH.ANDN.T	1	1	SH.OR.T	1	1
SH.EQ	1	1	SH.ORN.T	1	1
SH.GE	1	1	SH.XNOR.T	1	1
SH.GE.U	1	1	SH.XOR.T	1	1
SH.H	1	1	SHA	1	1
SH.LT	1	1	SHA.H	1	1
SH.LT.U	1	1	SHAS	2	1
SH.NAND.T	1	1			

Table 67 Simple arithmetic instruction timing: Coprocessor 0 instructions

Instruction	Result latency	Repeat rate	Instruction	Result latency	Repeat rate
BMERGE	2	1	IXMIN	2	1
BSPLIT	2	1	UNPACK	2	1
PARITY	2	1	IXMAX	2	1
PACK	2	1	IXMAX.U	2	1
IXMIN.U	2	1	CRC32B.W	2	1
CRC32L.W	2	1	CRC32.B	2	1
CRCN	2	1	MULP.B	2	1

Table 68 Simple arithmetic instruction timing: Integer divide instructions

Instruction	Result latency	Repeat rate	Instruction	Result latency	Repeat rate
DVADJ	2	1			
DIV[1)]	4-11	3-9	DIV.U[1)]	4-11	3-9
DVINIT	2	1	DVINIT.U	2	1
DVINIT.B	2	1	DVSTEP	2	1
DVINIT.H	2	1	DVINIT.HU	2	1
DVSTEP	6	4	DVSTEP.U	6	4
DIV64[1)]	4-19	3-17	DIV64.U[1)]	4-19	3-17
REM64[1)]	4-19	3-17	REM64.U[1)]	4-19	3-17

1) The latency and repeat rate values listed for the DIV, DIV.U, DIV64, DIV64.U, REM64, and REM64.U instructions are the minimum and maximum values. The pipeline provides early termination of the instruction once the full result is available.

Multiply instruction timings

Table 69 Multiply instruction timing

Instruction	Result latency	Repeat rate	Instruction	Result latency	Repeat rate
MUL	2	1	MUL.Q	3	1
MUL.U	2	1	MULM.H	3	1
MULS	3	1	MULR.H	3	1
MULS.U	3	1	MULR.Q	3	1
MUL.H	3	1			

Multiply accumulate (MAC) instruction timings

Table 70 Multiply accumulate instruction timing

Instruction	Result latency	Repeat rate	Instruction	Result latency	Repeat rate
MADD	3	1	MSUB	3	1
MADD.U	3	1	MSUB.U	3	1
MADDS	3	1	MSUBS	3	1
MADDS.U	3	1	MSUBS.U	3	1
MADD.H	3	1	MSUB.H	3	1
MADD.Q	3	1	MSUB.Q	3	1
MADDM.H	3	1	MSUBM.H	3	1
MADDMS.H	3	1	MSUBMS.H	3	1
MADDR.H	3	1	MSUBR.H	3	1
MADDR.Q	3	1	MSUBR.Q	3	1
MADDRS.H	3	1	MSUBRS.H	3	1
MADDRS.Q	3	1	MSUBRS.Q	3	1
MADDS.H	3	1	MSUBS.H	3	1
MADDS.Q	3	1	MSUBS.Q	3	1
MADDSU.H	3	1	MSUBAD.H	3	1
MADDSUM.H	3	1	MSUBADM.H	3	1
MADDSUMS.H	3	1	MSUBADMS.H	3	1
MADDSUR.H	3	1	MSUBADR.H	3	1
MADDSURS.H	3	1	MSUBADRS.H	3	1
MADDSUS.H	3	1	MSUBADS.H	3	1

For All MADD, MSUB, and MUL type instructions the result latency is reduced to 1 for accumulator forwarding between similar instructions. Instructions with 32-bit result forwarding to 32-bit accumulate and 64-bit results forwarding to 64-bit accumulate are considered to be similar.

4.3.5.2 Load-store pipeline instructions

These tables define the TC1.8 CPU load-store pipeline instruction timings.

Table 71 Load store arithmetic instructions

Instruction	Result latency	Repeat rate	Instruction	Result latency	Repeat rate
ADD.A	1	1	GE.A	1	1
ADDIH.A	1	1	LT.A	1	1
ADDSC.A	2	1	NE.A	1	1
ADDSC.AT	2	1	NEZ.A	1	1
EQ.A	1	1	SUB.A	1	1
EQZ.A	1	1	NOP	1	1

Table 72 Trap and interrupt instructions

Instruction	Result latency	Repeat rate	Instruction	Result latency	Repeat rate
DEBUG	–	1	TRAPSV[1)]	–	1
DISABLE	–	1	TRAPV[1)]	–	1
ENABLE	–	1	RSTV	–	1
RESTORE	–	1	WAIT	-	1
TRAPINV[2)]	-	-			

1) Execution cycles when no TRAP is taken. The execution timing in the case of raising these TRAPs is the same as other TRAPs such as SYSCALL.
2) The execution timing is the same as an IOPC TRAP

Table 73 Address move instructions

Instruction	Result latency	Repeat rate	Instruction	Result latency	Repeat rate
MFCR	2	1	MOV.A	1	1
MTCR	-	1	MOV.AA	1	1
MOVH.A	1	1	MOV.D	1	1

Table 74 Sync instructions

Instruction	Result latency	Repeat rate	Instruction	Result latency	Repeat rate
DSYNC	-	1[1)]	ISYNC[2)]	-	1
LSYNC	-	1[1)]			

1) Repeat rate assumes the store buffer is empty and all stores completed.
2) Repeat rate assumes that code fetch takes a single cycle.

Load instruction timing

Table 75 Load instruction timing

Instruction	Address latency	Result latency	Repeat rate	Instruction	Address latency	Result latency	Repeat rate
LD.A	1	3	1	LD.Q	1	2	1
LD.B	1	2	1	LD.W	1	2	1
LD.BU	1	2	1	LDLCX	-	5-9	5-9
LD.D	1	2	1	LDUCX	-	5-9	5-9
LD.DA	1	3	1	SWAP.W	2	3	6
LD.DD	2	2	2	LEA[1)]	–	1	1
LD.H	1	2	1	CMPSWAP.W	2	3	6
LD.HU	1	2	1	LHA	-	1	1
SWAPMSK.W	2	3	6				

1) The addressing mode returning an updated address is not relevant for this instruction.

Store instruction timing

Cache and Store instructions as with Load instructions will have a result for the pre-increment, post-increment, circular, or bit-reverse addressing modes, but do not produce a ‘register’ result.

- The memory references is naturally aligned
- The memory accessed takes a single cycle to accept a data item, or is snooped from the store buffer
- Timing is best case; no cache misses, no pending stores

Table 76 Cache instructions

Instruction	Result latency	Repeat rate	Instruction	Result latency	Repeat rate
CACHEA.I	1	1	CACHEA.WI[1)]	1	1
CACHEA.W[1)]	1	1	CACHEI.W[1)]	1	1
CACHEI.WI[1)]	1	1	CACHEI.I[1)]	1	1
CACHEA.W.VM[1)]	1	1	CACHEI.W.VM[1)]	1	1
CACHEI.WI.VM[1)]	1	1	CACHEI.I.VM[1)]	1	1
CACHEA.I.VM[1)]	1	1	CACHEA.WI.VM[1)]	1	1

1) Repeat rate assumes that no memory writeback operation occurs. Otherwise the repeat rate will depend upon the time for the castout buffers to clear.

Table 77 Store instructions

Instruction	Address latency	Repeat rate	Instruction	Address latency	Repeat rate
ST.A	1	1	ST.T	-	6
ST.B	1	1	ST.W	1	1
ST.D	1	1	STLCX	-	5-9
ST.DD	2	2	STUCX	-	5-9

(table continues...)

Table 77 (continued) Store instructions

Instruction	Address latency	Repeat rate	Instruction	Address latency	Repeat rate
ST.DA	1	1	LDMST	-	6
ST.H	1	1	-	-	-
ST.Q	1	1			

4.3.5.3 Load-store and integer conditional control flow instructions

Conditional control flow instructions generally do not produce any result value and only affect the next program counter value.

However, there are three exceptions. These are the LOOP instruction which produces an address register result, and the integer pipeline instructions JNED and JNEI, which produce a data register result.

Table 78 LOOP, JNED, and JNEI result latency

Instruction	Result latency
LOOP	4[1)]
JNED/JNEI	1

1) For LOOP result forwarding to non-LOOP instructions. For LOOP to LOOP the result latency is 1 clock

The remaining conditional control flow instructions are the integer pipeline conditional instructions (IP columns in table below)– JEQ, JGE, JGE.U, JGEZ, JGTZ, JLEZ, JLT, JLT.U, JLTZ, JNE, JNZ, JNZ.T, JZ and the load-store pipeline conditional instructions (LS columns in table below) - JEQ.A, JNE.A, JNZ.A, JZ.A.

Control flow instruction timing for the TC1.8 CPU can vary due to the state of branch target buffers and fetch FIFOs. The LOOP instruction (LP columns in table below) has the same penalties as a load-store conditional control flow instruction:

- Correctly predicted not taken branches incur no penalty
- Correctly predicted taken branches incur a penalty of up to two cycles depending on the state of the fetch FIFOs and the branch target buffer
- Incorrectly predicted LS instructions incur a three-cycle branch recovery penalty
- Incorrectly predicted IP instructions incur a four-cycle branch recovery penalty

Timing assumptions:

- All targets yield a full instruction in one access. That is, not 16-bit of a 32-bit instruction
- All code fetches take a single cycle

Table 79 Control flow timing

Prediction-result	Flow latency (LS, LP)	Repeat rate (LS, LP)	Flow latency (IP)	Repeat rate (IP)
Correct not-taken	1	1	1	1
Correct taken	1-3	1-3	1-3	1-3
Incorrect not-taken	3	4	4	4
Incorrect taken	3	4	4	4

Branch prediction

The TC1.8 CPU branch prediction system contains a dedicated loop buffer for nested loops, along with a branch target buffer and a branch history buffer.

4.3.5.4 Unconditional non CSA control flow instruction timing

Unconditional control flow changing instructions incur a penalty of up to two cycles depending on the state of branch target buffers and fetch FIFOs.

With the exception of the LOOPU instruction executed on the loop pipeline, the remainder - FCALL*, FRET, J, JA, JI, JL*I, JRI are all load-store instructions.

Timing Assumptions:

- All targets yield a full instruction in one access. That is, not 16-bit of a 32-bit instruction
- All code fetches take a single cycle

The following FCALL*/FRET control flow instructions all provide results to the implicit address register pair P[10] (A[10],A[11]).

Table 80 Unconditional FCALL/FRET control flow timing

Instruction	Flow latency	P[10] result latency	Repeat rate
FCALL	1-3	1	1-3
FCALLA	1-3	1	1-3
FCALLI	1-3	1	1-3
FRET	1-3	1	1-3

Some of following unconditional control flow instructions provide results to the implicit address register A[11].

Table 81 Unconditional jump control flow timing

Instruction	Flow latency	A[11] result latency	Repeat rate
J	1-3	-	1-3
JA	1-3	-	1-3
JI	1-3	-	1-3
JL	1-3	1	1-3
JLA	1-3	1	1-3
JLI	1-3	1	1-3
JRI	1-3	-	1-3

4.3.5.5 Load-store CSA instruction timing

This section summarizes the timing of the TC1.8 CPU CSA Control Flow instructions. Most change control flow, although some (BISR, SVLCX, and RSLCX) do not, and only update the CSA structures.

Timing assumptions:

- All targets yield a full instruction in one access. That is, not 16-bit of a 32-bit instruction
- All code fetches take a single cycle
- Timing is best case; no data cache misses for context operations, no pending stores

Some of the following instructions also provide a result to the implicit address register A[11]

Table 82 Load-store CSA control flow instruction timing

Instruction	Flow latency	A[11] result latency	Repeat rate
BISR	-	-	4/8[1)]
CALL	4/8[1)]	1	4/8[1)]
CALLA	4/8[1)]	1	4/8[1)]
CALLI	4/8[1)]	1	4/8[1)]
RET	4/8[1)]	-	4/8[1)]
RFE	4/8[1)]	-	4/8[1)]
RFM	4/8[1)]	-	4/8[1)]
RSLCX	-	-	4/8[1)]
SVLCX	-	-	4/8[1)]
SYSCALL	4/8[1)]	-	4/8[1)]
HVCALL	5/9[1)]	-	5/9[1)]
RFH	5/9[1)]	-	5/9[1)]

1) DSPR or cache hit/DLMU

4.3.5.6 Floating-point pipeline timing

This table defines the TC1.8 CPU floating point pipeline instruction timings.

Table 83 Floating-point instructions

Instruction	Result latency	Repeat rate	Instruction	Result latency	Repeat rate
ABS.F	1	1	ABS.DF	2	1
ADD.F	2	1	ADD.DF	2	1
CMP.F	1	1	CMP.DF	1	1
DIV.F	8	6	DIV.DF	16	14
FTOI	2	1	DFTOI	2	1
FTOIN	2	1	DFTOIN	2	1
FTOIZ	2	1	DFTOIZ	2	1
-	-	-	DFTOL	2	1
-	-	-	DFTOLZ	2	1
FTOQ31	2	1	-	-	-
FTOQ31Z	2	1	-	-	-
FTOU	2	1	DFTOU	2	1
FTOUZ	2	1	DFTOUZ	2	1
-	-	-	DFTOUL	2	1
-	-	-	DFTOULZ	2	1

(table continues...)

Table 83 (continued) Floating-point instructions

Instruction	Result latency	Repeat rate	Instruction	Result latency	Repeat rate
FTOHP	2	1	DFTOF	2	1
HPTOF	2	1	FTODF	2	1
ITOF	2	1	ITODF	2	1
-	-	-	LTODF	2	1
MADD.F	3	1	MADD.DF	4	2
MAX.F	1	1	MAX.DF	2	1
MIN.F	1	1	MIN.DF	2	1
MSUB.F	3	1	MSUB.DF	4	2
MUL.F	2	1	MUL.DF[1)]	2-3	1-2
NEG.F	1	1	NEG.DF	2	1
Q31TOF	2	1	-	-	-
QSEED.F	1	1	QSEED.DF	2	1
SUB.F	2	1	SUB.DF	2	1
UPDFL	–	1	-	-	-
UTOF	2	1	UTODF	2	1
-	-	-	ULTODF	2	1

1) Double-Precision MUL.DF takes 3 cycles if any operand or the result is subnormal.

4.3.6 Local memory details

This section describes the features of the local memories of the TC1.8 CPU.

The processor implementation includes the following local memories:

- Data Scratch Pad Ram (DSPR)
  - Single cycle memory for local data operations, with code access through SRI
- Data Cache (DCache)
  - Single cycle memory for data operations
  - DCache can optionally be used as additional DSPR
- Distributed LMU (DLMU)
  - Two-cycle memory for local data operations, with code access through SRI
  - Contiguous with other DLMU memory in the system
  - The DLMU may be implemented as a standard SRAM or as a Standby SRAM. Where standby SRAM is available the DLMU contents will be maintained during standby power saving mode
- Program Scratch Pad Ram (PSPR)
  - Single cycle memory for local code fetch, with data access through SRI

- Program Cache (PCache)
  - Single cycle memory for code fetch
- Local PNVM Bank (LPB)
  - None, one, or two PNVM modules with multi-cycle NVM memory, local code and data read operations are direct access

The sizes of each of these memories is detailed in the CPU configurations section in the device specific manual.

Related information

NVM prefetch buffer (FPB) on page 173

TC4Dx CPU configuration on page 396

4.3.6.1 Data scratchpad RAM (DSPR)

The DSPR provides fast, deterministic data access to the CPU for use by performance critical code sequences.

The DSPR is organized as multiple memory “towers”. This organization allows the CPU to access 64-bit of data from any 16-bit aligned address

The DSPR may also be accessed from the SRI Slave interface by another bus master, with both read and write transactions supported. The DSPR may be accessed by the SRI Slave interface using any SRI transaction type, including burst transfers. In accordance with the SRI protocol, accesses to the SRI Slave interface must be naturally aligned.

The data scratchpad RAM is ECC protected across 16-bit data. Byte writes to the DSPR will be performed as read-modify-write operations at the memory interface. Any read-modify-write operation that detects an error on the read phase will not perform the write phase of operation. Byte writes to uninitialized memory always result in an ECC error being detected on the read phase and therefore no data will be written to the memory.

4.3.6.2 Distributed LMU (DLMU)

The DLMU provides fast, deterministic data access to a segment of the global LMU, ideally for use by the local CPU for global data.

The DLMU is organized as a 64-bit wide memory with ECC protection implemented on a 64-bit granularity. Within a double-word, sub-double-word write accesses are performed as internal read-modify-write operations.

The DLMU may also be accessed by an external master, with read, write, and atomic read-write sequence. The DLMU may be accessed using any SRI transaction type, including burst transfers.

The DLMU is not cached for data accesses by the local CPU regardless of the address segment. This is the same as for the DSPR.

Errors detected at the ECC decoders are signaled to the SMU through the SSH logic of the associated SRAM.

Any read-modify-write operation that detects an error on the read phase will not perform the write phase of operation. Sub 64-bit writes to un-initialized memory will always result in an ECC error being detected on the read phase and so no data will be written to the memory.

The TC1.8 CPU DLMU memory:

- Runs at half the frequency of the TC1.8 CPU and will introduce stalls into the CPU pipeline for load operations and store operations when the store buffers are full
- Contains a section of standby RAM (refer to the device specific CPU configuration)
- Does not implement data pacing to support burst read BTR transfers in cases when no ECC error is encountered. When a burst read encounters an ECC error in the memory, each beat of the burst that has an error will be delayed by 1 cycle using the data pacing feature of the SRI

4.3.6.3 Program Scratchpad RAM (PSPR)

The Program scratchpad RAM provides a fast, deterministic program fetch access from the CPU for use by performance critical code sequences.

***Note**: CPU program fetch accesses to scratchpad RAM are never cached in the program cache and are always directly targeted to the scratchpad RAM*

The CPU fetch interface will generate aligned accesses (64-bit), which will result in 64-bits of instruction being returned to the CPU.

Note that the CPU Fetch Unit can only read from the scratchpad RAM and can never write to it.

The scratchpad RAM may also be accessed from the SRI Slave interface by another bus master, such as the Data Memory Interface (DMI). The scratchpad RAM may be both read and written from the SRI. The SRI Slave interface supports all SRI transaction types.

The Program scratchpad RAM is ECC protected across 64-bit data. Writes to the PSPR that are less than 64 bits, or which are 64 bits but not naturally aligned will be performed as read-modify-write operations at the memory interface. Any read-modify-write operation that detects an error on the read phase will not perform the write phase of operation. Sub-64 bit writes to uninitialized memory always result in an ECC error being detected on the read phase and therefore no data will be written to the memory.

4.3.6.4 Local and global addressing

The TriCore™ architecture supports closely coupled program and data SRAM memories known as Program Scratch Pad RAM (PSPR) and Data Scratch Pad RAM (DSPR). The local addressing of the PSPR memory is always located at $C0000000_H$. The local addressing of the DSPR is always located at $D0000000_H$.

In a multiprocessor system the scratch pad memories appear in the global address map in the following locations:

Table 84 Global address locations

CORE_ID value	PSPR base	DSPR base
0	70100000_H	70000000_H
1	60100000_H	60000000_H
2	50100000_H	50000000_H
3	40100000_H	40000000_H
4	30100000_H	30000000_H
5	20100000_H	20000000_H
6 (CPUcs)	10100000_H	10000000_H

The CPUs always use the global addresses for SRI transactions. Therefore a data load from $C0000000_H$ (a CPUs local PSPR) will result in a SRI transaction with an address in the range 10100000_H - $701FFFFF_H$ dependent on the CORE_ID value of the processor.

Similarly a code fetch form $D0000000_H$ (a CPUs local DSPR) will result in a bus transaction with an address in the range 10000000_H - $700FFFFF_H$ dependent on the CORE_ID value of the processor.

4.3.6.5 PSPR address aliasing

In some situations the software in the PSPR is not static but dynamically loaded. For debugging this is hard to handle in ELF files and for breakpoints and trace. To assist debugging, knowledge of which code image is resident would ideally be available to the debug tooling.

When the option to enable the PSPR aliasing is enabled in COMPAT.PSPRALIAS==0, the TC1.8 CPU implements 8 mirrors of the physical PSPR regions per core. This means that 8 different code images can be easily distinguished by address for local code fetches and for accesses via the SRI slave interface. The run-time software still needs to ensure that the right code is loaded, but for debug and trace, for the MPU and within the ELF file, the different code can be easily distinguished.

The TC1.8 CPU supports 8 aliased 128K PSPR regions between 1 Mb and 2 Mb in each CPU segment. When the PCACHE is not required, the PSPR extension also supports the aliasing. Mapped TAGs are relocated to a different offset.

Table 85 PSPR address locations

Memory region alias	Start offset	End offset
0	100000_H	$11FFFF_H$
1	120000_H	$13FFFF_H$
2	140000_H	$15FFFF_H$
3	160000_H	$17FFFF_H$
4	180000_H	$19FFFF_H$
5	$1A0000_H$	$1BFFFF_H$
6	$1C0000_H$	$1DFFFF_H$
7	$1E0000_H$	$1FFFFF_H$

The 8 alias regions are supported for both local and global addressing.

Concurrent use of two aliases for the same physical locations is not supported. Multiple ELF images can be dynamically managed in non-overlapping subsets of the PSPR. When different images are loaded into overlapping portions of the PSPR, then the software should ensure that the store buffer is flushed in case the PSPR is used as a shared data location.

4.3.6.6 Cache and tag storage addressing

For the purposes of testing the storage arrays, or if the caches are not functionally required, the cache and tag memories may be mapped into the system address space. When mapped the cache memories are contiguous with the DSPR/PSPR memories respectively as detailed in the following table.

When mapped into the address space the cache memories behave as extensions to the PSPR/DSPR memories and may be used as such.

The mapping of cache and tag memories to the TriCore™ address space is controlled by the MEMMAP register in the VMT module.

The base address of the cache storage when mapped as SPR extension is detailed in the following table:

Table 86 Cache storage addressing when mapped

Memory	Base local address	Base global address
Program cache as PSPR extension	$C0000000_H$+PSPR memory size	PSPR base + PSPR memory size
Data cache as DSPR extension	$D0000000_H$+DSPR memory size	DSPR base + DSPR memory size

When mapped as PSPR extension, it will also be accessible in each aliased PSPR region.

The mapping of the Tag memories is provided for test purposes only. They may not be used as standard memory.

The base address of the tag storage when mapped into the address space is detailed in the following table:

Table 87 TAG storage addressing when mapped

Memory	Base local address	Base global address
Program TAG	$C0200000_H$	PSPR base + 200000_H
Data TAG	$D0200000_H$	DSPR base + 200000_H

4.3.6.7 Cache architecture

The TC1.8 CPU implements both a 2-way set-associative instruction cache and a 2-way set-associative data cache.

The data cache is a writeback cache with allocation on read and write.

Caches and PSW.S

Only one copy of an address will exist in the cache at any one time. An attempt to access an existing address with a different master TAG ID (based on PSW.S) will result in the eviction (writeback of dirty data) or invalidation of the current entry and a refetch using the new access information.

Caches and VMN

Only one copy of an address will exist in the cache at any one time. An attempt to access an existing address with a new VM number will result in the eviction or invalidation of the current entry and a refetch using the new access information.

Caches and PRS

When the option to capture the PRS in data cache is enabled COMPAT.TAGPRS, an attempt to load an existing address with a different PRS will result in the eviction or invalidation of the existing entry. When COMPAT.TAGPRS is disabled, all dirty cache data will be written back with no PRS value (PRS=0 and PRS_EN=0).

***Note**: This is independent of the value of* CORECON.PROTEN

Data cache invalidate

To enable the fast removal of clean data from the data cache, the 'data cache invalidate' control bit is provided in the DCON1 CSFR. The register bit DCON1.DCINV operates in a similar way to the bit PCON1.PCINV. When the bit is set to 1 all clean data lines within the data cache are invalidated.

Related information

Virtualization and caches on page 133

4.3.6.8 Data cache (DCACHE)

The DCache is a two-way set-associative cache with a Least-Recently-Used (LRU) replacement algorithm. Each line contains 256 bits of data along with ECC bits. A single valid bit and a single dirty bit are associated with each line.

CPU data accesses to a cacheable memory segment target the DCache. If the requested address and its associated data are found in the cache (Cache Hit), the data is passed to/from the CPU Load-Store Unit without

incurring any wait states. If the address is not found in the cache (Cache Miss), the DMI cache controller issues a cache refill sequence and wait states are incurred whilst the cache line is refilled. The CPU load-store interface will generate unaligned accesses (16-bit aligned), which will result in up to 64-bits of data being transferred to or from the CPU (for non-context operations). If the data access is made within a DCache line, no matter the alignment, and a cache hit is detected then the requested data is returned to the CPU in a single cycle. If the data access is made to the end of a DCache line, such that the requested data would span two DCache lines, a single wait cycle is incurred (if both cache lines are present in the cache, otherwise a refill sequence is required for the missing cache line(s)).

The data cache is of the writeback type. When the CPU writes to a cacheable location the data is merged with the corresponding cache line and not written to main memory immediately. Associated with each cache line is a single ‘dirty’ bit, to denote that the data in the cache line has been modified. Whenever a CPU load-store access results in a cache miss, and each of the potential cache ways that could hold the requested cache line are valid, one of the cache lines is chosen for eviction based upon the LRU replacement algorithm. The line selected for eviction is then checked to determine if it has been modified using its dirty bit. If the line has not been modified the line is discarded and the refill sequence started immediately. If the line has been modified then the dirty data is first written back to main memory before the refill is initiated. Due to the single dirty bit per cache line, 256 bits of data will always be written back, resulting in a SRI Burst-4 Transfer (BTR4) transactions.

Data Cache refills always result in the full cache line being refilled, with the critical double-word of the DCache line being fetched first. A refill sequence will always affect only one cache line. There is no prefetching of the next cache line. Due to the uniform size of DCache refill sequences, such refills are always implemented using SRI Burst Transfer 4 (BTR4) transactions.

All Cache SRAMs are ECC protected.

All TAG SRAMs are ECC protected

Data cache bypass

The Data Cache may be bypassed, under control of DCON0.DCBYP, to provide a direct data access path to memory. The default value of DCON0.DCBYP is such that the data cache is bypassed after reset.

Whilst the data cache bypass is enabled, a data access request by the CPU to a cacheable address will result in a forced cache miss. Any valid cache lines within the data cache will remain valid and unchanged whilst the data cache is bypassed. As such data accesses to cacheable addresses with the data cache bypass enabled behave identically to data accesses to non-cacheable addresses.

4.3.6.9 Program cache (PCACHE)

The program Cache is a two-way set-associative cache with a Least-Recently-Used (LRU) replacement algorithm. Each PCACHE line contains 256 bits of instruction and associated ECC bits.

CPU program fetch accesses which target a cacheable memory segment (and where the PCACHE is not bypassed) target the PCACHE. If the requested address and its associated instruction are found in the cache (Cache Hit), the instruction is passed to the CPU Fetch Unit without incurring any wait states. If the address is not found in the cache (Cache Miss), the PMI cache controller issues a cache refill sequence and wait states are incurred whilst the cache line is refilled. The fetch request always returns an aligned packet to the CPU.

The program cache is ECC protected on 64-bit data.

The program TAG Ram is ECC protected on 22-bit data.

Program Cache Refill Sequence

Program Cache refills are performed using a critical double-word first strategy with cache line wrapping such that the refill size is always 4 double-words. PCACHE refills are always performed in 64-bit quantities. A refill sequence will always affect only one cache line. There is no prefetching of the next cache line.

PCACHE refills are therefore implemented using an SRI Burst Transfer 4 (BTR4) transfers. The program cache supports instruction streaming, meaning that it can deliver available instruction half-words to the CPU Fetch Unit whilst the refill operation is ongoing.

Program Cache Bypass

The Program Cache may be bypassed, under control of PCON0.PCBYP, to provide a direct instruction fetch path for the CPU Fetch Unit. The default value of PCON0.PCBYP is such that the PCACHE is bypassed after reset.

Whilst PCACHE bypass is enabled, a fetch request by the CPU to a cacheable address will result in a forced cache miss, such that the cache controller issues a standard refill sequence and supplies instruction half-words to the CPU using instruction streaming, without updating the cache contents. Any valid cache lines within the PCACHE will remain valid and unchanged whilst the PCACHE is bypassed. As such, instruction fetch requests to cacheable addresses with PCACHE bypass enabled behave identically to instruction fetch requests to non-cacheable addresses.

Program Cache Invalidation

The PMI does not have automatic cache coherency support. Changes to the contents of memory areas external to the PMI that may have already been cached in the PCACHE are not detected. Software must provide the cache coherency in such a case. The PMI supports this via the cache invalidation function. The PCACHE contents may be globally invalidated by writing a ‘1’ to PCON1.PCINV. The PCACHE invalidation is performed over 64 cycles by a hardware state machine which cycles through the PCACHE entries marking each as invalid. During an invalidate sequence the CPU may continue to fetch instructions from non-cacheable memory. Any attempt to fetch instructions from a cacheable memory location during an invalidation sequence will result in the CPU stalling until the sequence completes. The status of the PCACHE invalidation sequence may be determined by reading the PCON1.PCINV bit.

4.3.6.10 DSPR access timing

The TC1.8 CPU implements a DSPR such that memory access are optimized for data aligned to the EABI rules. Data aligned to these rules is returned in a single cycle. Data not aligned to these rules may be returned in two cycles.

To maximize throughput for LD.DD and ST.DD instructions targeting the DSPR memory, they should use addresses with 16-byte alignment (the architecturally supported alignment may be no faster than a pair of LD.D & ST.D accesses).

Non-EABI aligned accesses to the DSPR can be counted by the performance counters.

4.3.7 CSFR, SFR and FSFR state

The internal processor state in the CPUs is split into two distinct groups. The majority of the registers are core special function registers (CSFRs). These registers are architecturally defined and must be accessed using MTCR/MFCR/MTDCR/MFDCR instructions (for access by the local processor). The remainder are SFRs and are implementation specific.

These SFR registers are accessed using load and store instructions through the system interconnect.

The TC1.8 CPU's FLASHCON (NVM control) registers are located in the FSFR address region to ensure separation between CPU and NVM access path.

4.3.7.1 CSFR, SFR, and FSFR address map and external access

Each CPU has a dedicated set of control and status registers accessed at the addresses detailed in this section. These registers are distinguished as Core Special Function Registers (CSFRs), Special Function Registers (SFRs), and NVM Special Function Registers (FSFRs).

CSFRs are usually registers defined in the architecture to support features required by an operating system (OS), real time operating system (RTOS), or hypervisor. A CPU must access its own CSFR registers using MTCR, MFCR, MTDCR, and MFDCR instructions. The CSFR registers of other CPUs may be accessed using load and store

instructions to their system addresses. The update of CSFRs of other CPUs is only well defined if that CPU is halted.

SFRs are registers that implement system features that are not usually part of the TriCore™ architecture. SFR registers of any CPU may only be accessed using load and store instructions. The system timer, kernel reset, overlay control, PROT, and APU registers of a CPU are mapped into its SFR address range.

PNVM control and status registers for the local NVM banks are located in the FSFR address range.

The CSFR space for each Hardware resource (HRHV,A,B) and the SFR space each occupy a separate 64 KB access region in Segment 15 for each processor.

The FSFR space is non-contiguous with the SFR/CSFR space and occupies another 64 KB region in segment 15 for each processor.

TC18 - Physical address of any CPU CSFR or SFR or FSFR

```
FSFR       : 0xF8400000 + (CORE_ID << 18) + FSFR offset
SFR        : 0xF8800000 + (CORE_ID << 18) + SFR offset
CSFR HRA   : 0xF8810000 + (CORE_ID << 18) + CSFR offset
CSFR HRB6) : 0xF8820000 + (CORE_ID << 18) + CSFR offset
CSFR HRHV6) : 0xF8830000 + (CORE_ID << 18) + CSFR offset
```

SFR, CSFR and FSFR bus access width

Accesses to SFR, CSFR, and FSFR registers from the system will support accesses widths of 32 and 64-bit.

Related information

FSFR range on page 172

4.3.7.2 SFR and CSFR partitioning

All new hypervisor registers will be added at offset $B000_H$ (Only in HRHV).

SFR	HRA	HRB	HRHV	Offset	Type
X				0000_H	STM
X				1000_H	Test registers
X				$D000_H$	Kernel reset SFRs
X				$E000_H$	APU SFRs
X				$FB00_H$	Overlay SFRs
	X	X	X	8100_H	PMA CSFRs
	X	X	X	9000_H	DMI CSFRs (including integrity error)
	X	X	X	9200_H	PMI CSFRs (including integrity error)
	X	x	X	$A000_H$	FPU CSFRs
			X	$B000_H$	Hypervisor registers
	X	X	X	$C000_H$	MPU Data protection ranges CSFRs
	X	X	X	$D000_H$	MPU Code protection ranges CSFRs
	X	X	X	$E000_H$	MPU (Code, Data) control CSFRs

6 only accessible when virtualization is present and enabled

SFR	HRA	HRB	HRHV	Offset	Type
	X	X	X	$E400_H$	Temporal protection CSFRs
				$F000_H$ & $FD00_H$	Debug registers
	X				Virtualization disabled
	X	X	X		Virtualization enabled (registers not replicated, trap to hypervisor when in HRA/B)
	X	X	X	$FC00_H$	Performance counters
	X	X	X	$FE00_H$	Main CSFRs 00 PCXI, PSW, PC, ----, 10 ----, CORECON, CPU_ID, CORE_ID, 20 BIV, BTV, ISP, ICR, 30 ----, PPRS, FCX, LCX, 40 SWID 50 CUS_ID 60 BOOTCON, LCLCON, CCON, TCCON Not replicated => PCXI, PSW, PPRS, PC, xxxID BOOTCON, LCLCON, CCON, TCCON
	X	X	X	$FF00_H$	GPRs - Only partially replicated (A0, A1, A8, A9).

***Note**: With the exception of hypervisor registers, non-replicated CSFRs are aliased in the memory map. For example, PCXI can be accessed at $1FE00_H$, $2FE00_H$ and $3FE00_H$.*

4.3.7.3 CORECON (SYSCON replacement)

In the AURIX™ TC4xx family the SYSCON register is renamed CORECON. This removes name clashes with the SYSCON register in the SCU which does control the system.

In order to make CORECON fully virtualized, both ESDIS and BHALT bits are relocated to COMPAT.ESEN and BOOTCON.BHALT.

The usage of (PROTSFRSE protection of the CPU core control register (CORECON) is configurable by the application software. From system reset, as well as after any kernel reset, the register is not subject to PROTSFRSE protection. Clearing the compatibility control register safety protection field (COMPAT.SP=$0B_B$) enables the PROTSFRSE protection of the register.

Related information

System Control Unit (SCU) on page 5872

4.3.7.4 Emulator space disable

The emulator space control bit (ESDIS) was previously held in the CORECON register (SYSCON.ESDIS/ CORECON.ESDIS). For version 1.8 of the TriCore™ architecture this control is moved to the COMPAT register bit[8]. To provide a compatible behavior from reset, the polarity of the bit is inverted and the control becomes the emulator space enable (COMPAT.ESEN).

4.3.7.5 Boot control: Boot halt

A new register BOOTCON ($FE60_H$) is introduced to hold the BHALT bit.

Following a system reset, both CPU0 and CPUcs will start their boot sequences. All other cores will be placed in the boot-halt state. In the boot-halt state the core is halted, and the instruction fetch system stalled. All traps and interrupts are ignored by a core in this state, but all memories and registers associated with the core are available for access through the system interconnect.

A core in boot-halt has BOOTCON.BHALT set. The core will remain halted until the BOOTCON.BHALT bit is written to “0”, when it will start to fetch from the PC defined in the PC CSFR. This will be the boot PC unless otherwise updated. A write to “1” of the BOOTCON.BHALT will be ignored.

4.3.7.6 Floating-point error registers

The following floating point error registers are updated in respect to previous TriCore™ implementations:

```
FPU_TRAP_OPC: Bit[9] – Double/single precision identifier
```

The following FPU trap registers are renamed and contain the lower 32-bit of input operand SRC registers for single and double precision floating-point operations. In order to improve software handling, the registers offsets are not compatible with previous versions of the architecture to allow reading of a complete operand in one instruction using the new MFDCR instruction.

```
FPU_TRAP_SRC1 -> FPU_TRAP_SRC1_L (0xa010)
FPU_TRAP_SRC2 -> FPU_TRAP_SRC2_L (0xa018 was previously 0xa014)
FPU_TRAP_SRC3 -> FPU_TRAP_SRC3_L (0xa020 was previously 0xa018)
```

The following FPU trap registers are new for version 1.8 of the TriCore™ architecture, and contain the upper 32-bit of input operand SRC registers for double precision floating-point operations.

```
FPU_TRAP_SRC1_U (0xa014)
FPU_TRAP_SRC2_U (0xa01C)
FPU_TRAP_SRC3_U (0xa024)
```

4.3.7.7 BTV, BHV reset values

BTV and BHV reset values are computed using fixed offsets from the base address of the ROM associated with the CPU instance.

For host CPUs the base address of the system BOOTROM will be used, while for the Secure CPU instance (CPUcs) the base address of the CSROM will be used.

BTV reset value = ROMADDR_base[31:16], 16’hFE00

BHV reset value = ROMADDR_base[31:16], 16’hFD00

4.3.7.8 Performance counters

The performance counters are decoupled from the debug system and can now be enabled on any device without requiring the use of the debug system. This allows performance to be monitored by/within applications.

Table 88 Counter source details

Counter name	Function
IP_DISPATCH_STALL	The counter is incremented on every cycle in which the Integer dispatch unit is stalled for whatever reason
LS_DISPATCH_STALL	The counter is incremented on every cycle in which the Load-Store dispatch unit is stalled for whatever reason
LP_DISPATCH_STALL	The counter is incremented on every cycle in which the Loop dispatch unit is stalled for whatever reason
PCACHE_HIT	The counter is incremented whenever the target of a cached fetch request from the fetch unit is found in the program cache
PCACHE_MISS	The counter is incremented whenever the target of a cached fetch request from the fetch unit is not found in the program cache and hence a bus fetch is initiated
MULTI_ISSUE	The counter is incremented in any cycle where more than one instruction is issued
DCACHE_HIT	The counter is incremented whenever the target of a cached request from the Load-Store unit is found in the data cache
DCACHE_MISS_CLEAN	The counter is incremented whenever the target of a cached request from the Load-Store unit is not found in the data cache and hence a bus fetch is initiated with no dirty cache line eviction
DCACHE_MISS_DIRTY	The counter is incremented whenever the target of a cached request from the Load-Store unit is not found in the data cache and hence a bus fetch is initiated with the writeback of a dirty cache line
TOTAL_BRANCH	The counter is incremented in any cycle in which a branch instruction is in a branch resolution stage of the pipeline (IP_EX1, LS_DEC, LP_DEC). This counter is also incremented for ISYNC, CALL, RET, RFE, RFM, FCALL/FRET and RFH instructions
PMEM_STALL	The counter is incremented whenever the fetch unit is requesting an instruction and the Instruction memory is stalled for whatever reason
DMEM_STALL	The counter is incremented whenever the Load-Store unit is requesting a data operation and the data memory is stalled for whatever reason

Performance counter registers

The various counter sources listed above may be selected for counting by programming the M1, M2 and M3 fields in the CCTRL register. (See the TriCore™ architecture manual for register details). The mappings are listed below.

Table 89 Performance counter registers

Name	Description	Offset address
CCTRL	Counter Control Register	$FC00_H$
CCNT	CPU Clock Count Register	$FC04_H$
ICNT	Instruction Count Register	$FC08_H$
M1CNT	Multi Count Register 1	$FC0C_H$
M2CNT	Multi Count Register 2	$FC10_H$
M3CNT	Multi Count Register 3	$FC14_H$

Table 90 Multi count configuration

CCTRL M1/M2/M3	M1CNT count function	M2CNT count function	M3CNT count function
000	IP_DISPATCH_STALL	LS_DISPATCH_STALL	LP_DISPATCH_STALL
001	PCACHE_HIT	PCACHE_MISS	MULTI_ISSUE
010	DCACHE_HIT	DCACHE_MISS_CLEAN	DCACHE_MISS_DIRTY
011	TOTAL_BRANCH	PMEM_STALL	DMEM_STALL

4.3.8 Data access overlay (OVC)

The data overlay provides the capability to redirect selected data accesses to the overlay memory. Data accesses made by the TriCore™ to Program NVM or Online Data Acquisition space can be redirected.

The presence of overlay is core specific. The TCCON register can be used to identify cores with overlay hardware support.

***Note**: The CPUcs instance does not have overlay support.*

Overlay memory may be located in the Local Memory (if present), in the DPSR/PSPR memory, or in another Program NVM bank[7].

Overlay functionality makes it possible for example, to modify the application's test and calibration parameters, which are typically stored in NVM memory, during the run-time of a program.

Only read and write data accesses are redirected. The access redirection is executed without performance penalty.

***Attention**: As the address translation is implemented in the DMI it is only effective for data accesses. Instruction fetches by the PMI or accesses by any other master, including the debug interface, are not effected*

***Warning**: Overlay address modification only happens after the address has passed all the MPU checks. The APU on all overlay memories is a final check which redirected accesses must pass to succeed*

Summary of features and functions

- Redirecting data accesses addressed to Program NVM and OLDA
- Support redirection to overlay memory located in:
  - Program NVM[7]
  - Local Memory Unit (LMU), if present, including DLMU
  - DSPR or PSPR memory
- Individual overlay system per processor core
- Up to 32 overlay ranges ("blocks") available in each TriCore™ instance
- Two levels of priority:
  - 2 low priority ranges with block sizes of 32 bytes up to 4 MB
  - 30 high priority ranges with block sizes of 32 bytes up to 512 KB
- Redirected address range span within an overlay memory of up to 16 MB/4 MB (low priority/high priority blocks)
  - Low priority blocks have a span of up to 16 MB for emulation devices, and are limited to a span of up to 4 MB in production devices
- Overlay memory location and block size selected individually for every overlay block
- Multiple overlay blocks can be enabled or disabled with one register write access
- Programmable flush (invalidate) control for data cache in DMI
- Overlay start/stop synchronized against data load

7 emulation devices only

4.3.8.1 Data access redirection

The principle of redirecting data access from the original target memory (“Target Address”) to overlay memory (“Redirected Address”) is shown here.

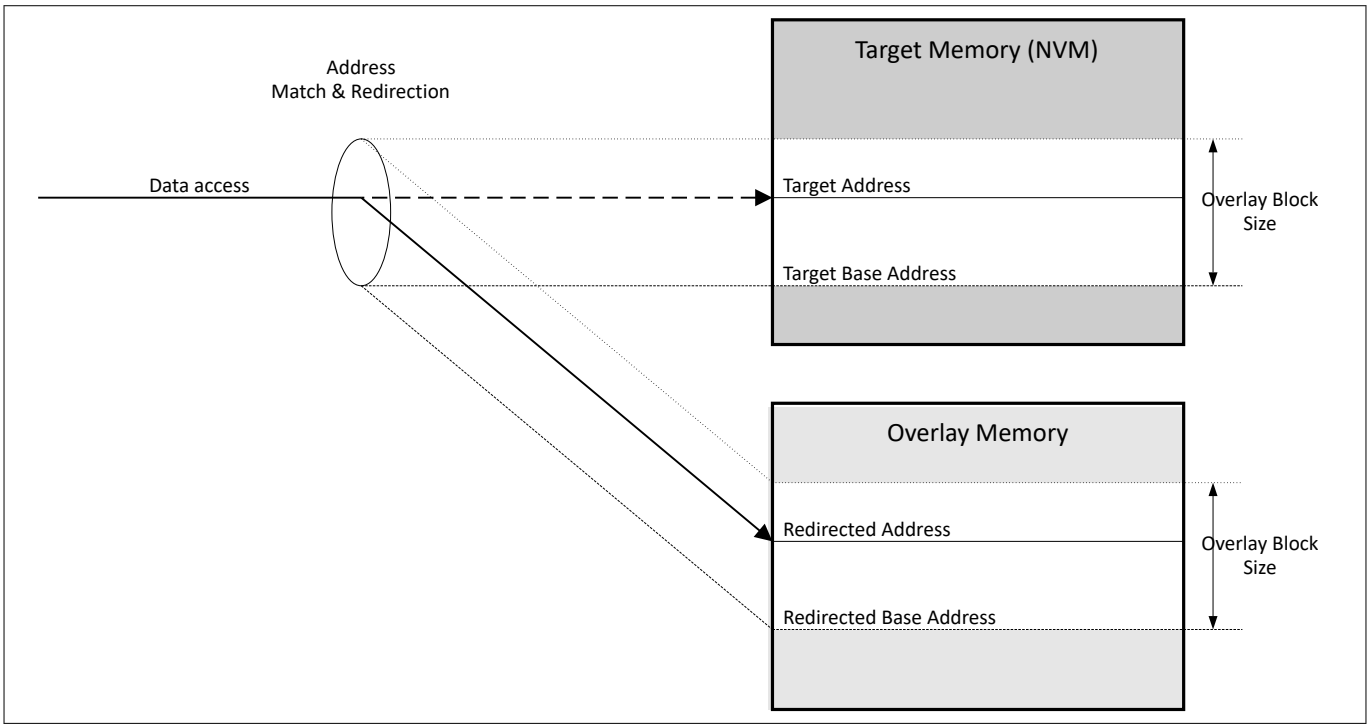


Figure 28 Data access redirection

Data access overlay is defined using overlay ranges (“overlay blocks”).

Each overlay block defines one continuous range of address space for which the accesses are redirected.

Each overlay block is configured with the following parameters:

- Overlay Block Target Base Address
  - The start address for the range of the target addresses to be redirected
- Overlay Block Size
  - The size of the range of the addresses to be redirected
- Overlay Block Redirection Base Address
  - The start address for redirection

In AURIX™ up to 32 overlay ranges can be used in each TriCore™ instance.

Each overlay block has 3 associated registers for independent configuration of these parameters.

The overlay parameters are configured as follows:

- Target Base Address is configured with OTARi register
- Overlay Block Size is configured with OMASKi register
- Redirection Base Address is configured with RABRi register

The size of the overlay memory blocks depends on the block index.

The size for the low priority set (block 0 and 1) can be 2^n x 32 bytes, with n = 0 to 17. This gives the block size range from 32 bytes to 4 Mbytes.

For the high priority set (blocks 2 to 31) n = 0 to 14. This gives the block size range from 32 bytes to 512 Kbytes.

The start address of the block can only be an integer multiple of the programmed block size (natural alignment boundary). If the OTARi register value, or RABRi register value is not aligned with the block size, the least significant bit values are ignored and treated as zero.

The Redirection Base Address is determined by two fields in the RABRi register:

- RABRi.OMEM selecting the Overlay Memory
- RABRi.OBASE selecting the base address within this memory

Each overlay block can be activated or deactivated with its RABRi.OVEN bit. Overlay blocks can be activated and deactivated independently, by directly accessing the RABRi register, or in groups, where multiple configured blocks are activated of deactivated concurrently. For information on concurrent block activation see the chapter "Global Overlay Control Synchronization".

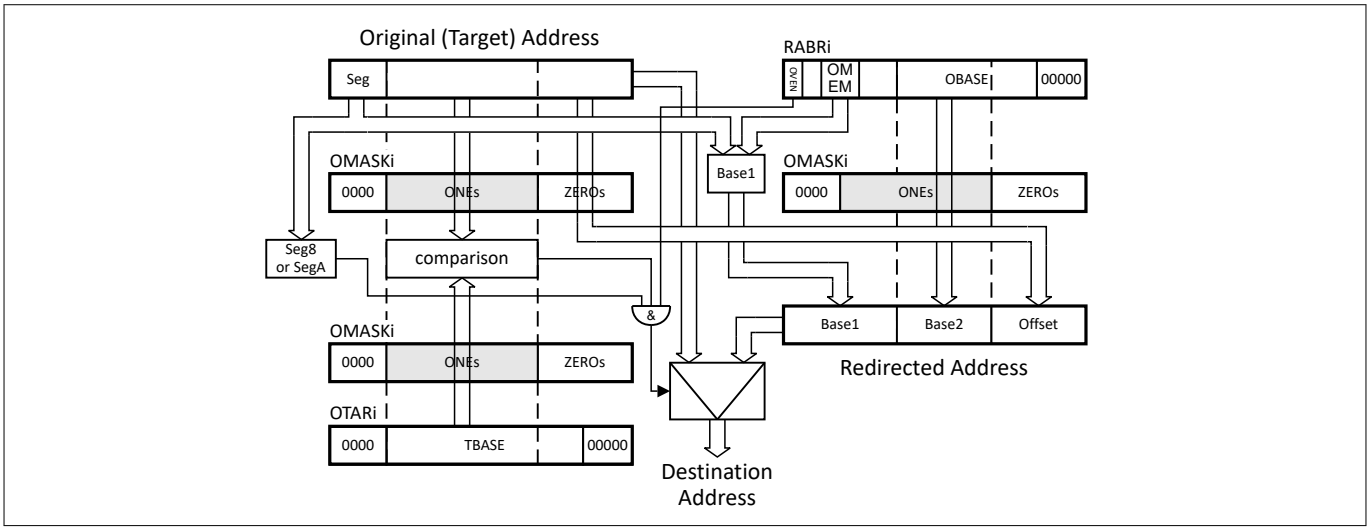


Figure 29 Address redirection process

Any data access to segment 8_H or segment A_H is checked against all the activated overlay blocks.

For each activated overlay block, address bits 27..5 are compared with the target base address (OTARi), and this bit-wise comparison is qualified by the content of OMASKi register.

Address bits participate in the comparison if the corresponding OMASKi bits are set to one. The access is redirected, if all the address bits selected by OMASKi equal to the corresponding bits in OTARi register.

The address for redirection is constructed as follows:

- Low priority ranges
  - Address bits 31..24 are set according to the Overlay Memory Selection (RABRi.OMEM) and the cache-ability of the original address
  - For address bits 23..5:
    - If the corresponding OMASKi bit is set, the address bit value is taken from RABRi.OBASE field
    - If the corresponding OMASKi bit is cleared, the address bit value is taken from the original address
  - Address bits 4..0 are always taken directly from the original address
- High priority ranges
  - Address bits 31..22 are set according to the Overlay Memory Selection (RABRi.OMEM) and the cache-ability of the original address
  - For address bits 21..5:
    - If the corresponding OMASKi bit is set, the address bit value is taken from RABRi.OBASE field;
    - If the corresponding OMASKi bit is cleared, the address bit value is taken from the original address
  - Address bits 4..0 are always taken directly from the original address

If there is no redirection, the original address is used to perform the access.

Overlapping redirections

Overlay ranges are split into two sets:

- Ranges 0 and 1 form the low priority set
- Ranges 2 to 31 form the high priority set

In order to minimize the number of descriptors used, the overlay supports the configuration of a low priority range as a wide overlay window, while using one of the high priority ranges to refine the redirection. An active high priority range takes precedence during block comparison.

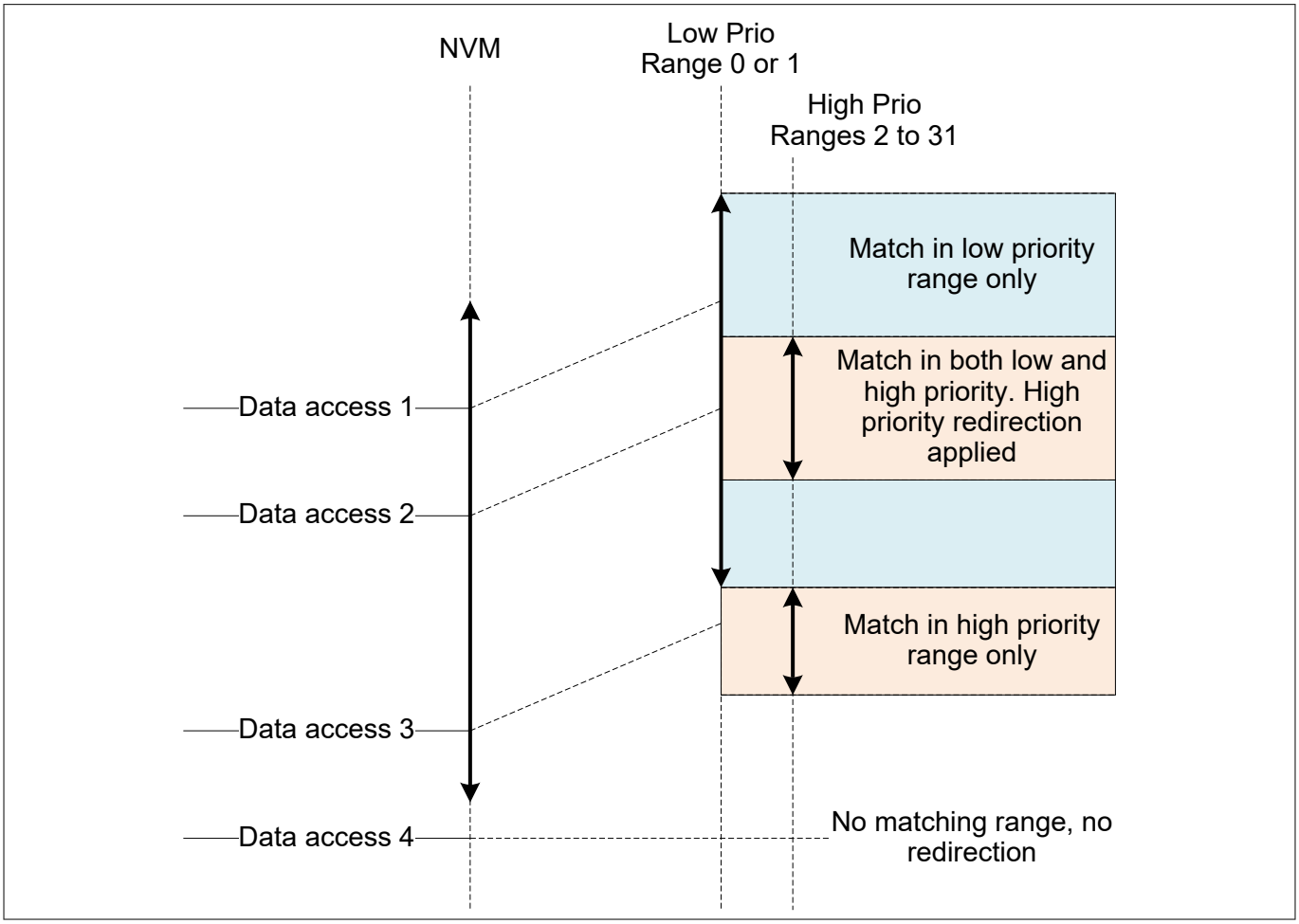


Figure 30 Two levels of priority

Target address ranges for activated overlay blocks in the same priority level should not overlap or an exception may occur. See the chapter "Access Protection, Attributes, Concurrent Matches".

Related information

Global overlay control synchronization on page 165

Access protection, attributes, concurrent matches on page 166

4.3.8.2 Target memories

Data access to any memory within the segment 8_H or segment A_H may be redirected to overlay memory. In particular, access to the following memories may be redirected:

- Program NVM
- Data NVM
- OLDA space

4.3.8.2.1 Online data acquisition (OLDA) space

Calibration is additionally supported by a virtual OLDA memory range. The base address of the virtual OLDA memory range is defined in the system address map.

4.3.8.3 Overlay memories

Here all the supported overlay memories are described. Depending on the device type, only a subset of the listed overlay memories are available. Overlay memory is selected independently for each block through the RABRi.OMEM field value.

4.3.8.3.1 Local memory

If present, the local memory (LMU) can be selected for overlay. The local memory is selected for overlay block 'i' redirection if the RABRi.OMEM value is either 8_H, 9_H or A_H. Values 9_H and A_H are reserved in production devices and only usable in emulation devices

LMU low - range 0 to 4 MB

The base address of the LMU is B/9000 0000_H.

During address translation, the upper 10 address bits are set to:

- $B0_H00_B$ (for target segment A_H)

or to

- 90_H00_B (for target segment 8_H)

LMU mid - range 4 MB to 8 MB

The base address of the LMU is B/9040 0000_H.

During address translation, the upper 10 address bits are set to:

- $B0_H01_B$ (for target segment A_H)

or to

- 90_H01_B (for target segment 8_H)

LMU high - range 8 MB to 12 MB

The base address of the LMU is B/9080 0000_H.

During address translation, the upper 10 address bits are set to:

- $B0_H10_B$ (for target segment A_H)

or to

- 90_H10_B (for target segment 8_H)

4.3.8.3.2 OWN

NVM to NVM redirection can be selected for overlay. The overlay to NVM (OWN) space is selected for overlay block 'i' redirection if the RABRi.OMEM value is C_H, D_H or E_H. These values are reserved in production devices and only usable in emulation devices.

PNVM low - range 0 to 16 MB

The base address of this space is A/8000 0000_H.

During address translation:

- For ranges 0 and 1, the upper 8 address bits are set to$A0_H$ (for target segment A_H) or to 80_H (for target segment 8_H)
- For ranges 2 to 31, the upper 10 address bits are set to$A0_H00_B$ (for target segment A_H) or to 80_H00_B (for target segment 8_H)

PNVM high - range 16 MB to 32 MB

The base address of this space is A/8100 0000_H.

During address translation:

- For ranges 0 and 1, the upper 8 address bits are set to$A1_H$ (for target segment A_H) or to 81_H (for target segment 8_H)
- For ranges 2 to 31, the upper 10 address bits are set to$A1_H00_B$ (for target segment A_H) or to 81_H00_B (for target segment 8_H)

PNVM SOTA alternate bank

The base address of this space is A/8200 0000_H.

During address translation:

- For ranges 0 and 1, the upper 8 address bits are set to$A2_H$ (for target segment A_H) or to 82_H (for target segment 8_H)
- For ranges 2 to 31, the upper 10 address bits are set to$A2_H00_B$ (for target segment A_H) or to 82_H00_B (for target segment 8_H)

4.3.8.3.3 DSPR and PSPR memory

Data scratch memory (DSPR) or program scratch memory (PSPR), from any application core, can be selected for overlay by configuring the block RABRi.OMEM value. During address translation, the upper 10 address bits are set to the target PSPR or DSPR target address.

Depending on the value set in RABRi.OBASE either DSPR memory (starting at offset 0_H) or PSPR memory (starting at offset 10 0000_H) can be used.

4.3.8.4 Global overlay control

Overlay can be disabled or enabled individually for each core with the SCU OVCENABLE register. If OVCENABLE.OVENx bit is cleared no address redirection is permitted on Core-x regardless of the remaining register settings. A write to the OVCENABLE register does not change any of the remaining register values.

While each overlay block can be activated and deactivated individually by writing its RABRi.OVEN bit, dedicated functionality is provided for concurrently activating and deactivating multiple blocks. This can be useful in maintaining data consistency across several memory regions.

For the purpose of concurrent activation and deactivation overlay blocks are selected in two stages:

- The individual blocks for activation and deactivation are selected with CPUx_OSEL registers for each core 'x' independently
- The set of cores is selected with OVCCON.CSEL field

Multiple overlay blocks can be simultaneously activated or deactivated with OVCCON.OVSTRT bit. When OVCCON.OVSTRT bit is written with one:

- If the OVCCON.CSELx bit is written with one, and the CPUx_OSEL.SHOVENi bit value is one, overlay block 'i' in core 'x' is activated, and CPUx_RABRi.OVEN bit is set
- If OVCCON.CSELx bit is written with one, and CPUx_OSEL.SHOVENi bit value is zero, overlay block 'i' in core 'x' is deactivated, and CPUx_RABRi.OVEN bit is cleared
- If OVCCON.CSELx bit is written with zero, the overlay configuration in core 'x' is not effected

The actions listed above are executed concurrently. The overlay configuration is not changed otherwise. With this function it is possible to switch directly from one set of overlay blocks to another set of overlay blocks.

Multiple overlay blocks can be simultaneously deactivated with OVCCON.OVSTP bit. When OVCCON.OVSTP is written with one:

- If OVCCON.CSELx bit is one, all the overlay blocks in core 'x' are deactivated, and all CPUx_RABRi.OVEN bits are cleared
- If OVCCON.CSELx bit is zero, the overlay configuration in core 'x' is not effected

The actions listed above are executed concurrently. The overlay configuration is not changed otherwise.

***Note**: Overlay should not be enabled or disabled using global OVCENABLE register if any of the blocks are enabled with RABRi.OVEN. Instead, OVSTRT or OVSTP should be used if concurrent block enabling or disabling is required.*

When OVCCON.DCINVAL is written with one, all the unmodified (clean) data cache lines in the selected cores are invalidated. The data cache lines containing modified (dirty) data are not effected. The cores not selected with OVCCON.CSEL field are not effected. Data cache invalidation can be combined with OVSTRT or OVSTP action. This function helps to assure that the CPU can access the new data after the overlay blocks have been activated or deactivated.

***Note**: OVCCON.CSEL field is written together with OVSTRT, OVSTP and DCINVAL bits in the same register, and only impacts any action triggered by the same write. CSEL, OVSTRT, OVSTP and DCINVAL do not retain the written value and always read as zero.*

The OVCCON.OVCONF user control flag is provided, together with its protection bit OVCCON.OVCONF_P. These bits do not impact the overlay functionality.

When the OVCCON register is written with CSELx set and either OVSTRT or OVSTP set, and at the same time CPUx_RABRi register is written, the resulting value of CPUx_RABRi.OVEN bit is not defined. Possibility of such simultaneous access should be avoided.

OVSTRT, OVSTP, and DCINVAL actions are not performed when TriCore™ is in idle state.

4.3.8.4.1 Global overlay control synchronization

When OVSTRT, OVSTP, or DCINVAL action is requested, its execution may be delayed to prevent changing overlay configuration during an ongoing data load.

Sufficient time should be allowed after an action request, before another action is requested. If a new action is requested while a previous action is still pending, due to synchronization with CPU loads, some actions may be lost.

4.3.8.5 Overlay configuration change

The overlay block should be disabled by clearing its RABRi.OVEN bit, before any changes are made to the OTARi, OMASKi, or RABRi registers. If not, unintended access redirection may occur.

The overlay block should only be enabled if the target address, the overlay memory selection, the redirection address, and the mask have all been configured with intended values.

***Note**: The overlay control does not prevent configuring the translation logic incorrectly. In particular, redirection to not implemented or forbidden address range is not prevented.*

Special care needs to be taken to synchronize overlay redirection change to the executed instruction stream if data consistency is required.

External accesses may be buffered in the CPU. External accesses that have not been completed may still be affected by overlay configuration changes. Therefore it is advised to ensure completion of all pending accesses, for example by executing a DSYNC instruction, before any overlay range is activated or deactivated.

When the overlay block is enabled and the same memory location is written through the target address space and read through the redirected address space, or vice-versa, the access synchronization needs to be enforced with a DSYNC and data cache writeback, if applicable.

4.3.8.6 Access protection, attributes, concurrent matches

When data access is redirected by overlay, access protection is applied as follows:

- Target address is subject to CPU Memory Protection (MPU) validation
- Redirected address is subject to APU validation

Physical memory attributes for the redirected access are determined based on the target address segment (see the CPU Physical Memory Attributes chapter).

Concurrent matches are only permitted between one of the overlay ranges 0,1 and one of the overlay ranges 2..31.

When an address matches two of the enabled ranges 0,1 or two or more of enabled overlay ranges 2..31, an exception is raised and the memory access is not performed.

A load operation with multiple matches on overlay ranges raises a data access synchronous error (DSE) trap, and a store operation raises a data access asynchronous error (DAE) trap. In such cases the relevant trap information registers are updated. The relevant registers are: data synchronous trap register (DSTR), data asynchronous trap register (DATR), and data error address register (DEADD). Please see the DMI Registers chapter for more information.

4.3.8.7 Overlay control registers

Overlay block control registers are located in each CPU that supports data access overlay. Overlay global control registers are located in the SCU. Overlay register access can be by APU register protection.

Per core overlay register

- CPUx_RABRi
- CPUx_OTARi
- CPUx_OMASKi
- CPUx_OSEL

Where 'x' is the core number and 'i' is the block number.

The overlay blocks are organized into two groups:

- Low priority group: Block numbers 0 and 1
- High priority group: Block numbers 2 to 31

Global overlay registers

- OVCENABLE
- OVCCON

These registers are described in the SCU chapter.

4.3.8.7.1 Block control registers

For each of the 32 overlay memory blocks, indicated by index i, three registers control the overlay operation and the memory selection:

- Redirected address base register RABRi, which selects the overlay memory, holds the block base address within this memory and contains block enable bit
- Overlay target address register OTARi, which holds the base address of the memory block being overlaid
- Overlay mask register OMASKi, which determines which bits from RABRi are used for the base address of the overlay memory and block, and which bits of the original data address are directly used as offset within the block

Additionally, the overlay range select register OSEL determines which blocks are to be enabled and which blocks are to be disabled when the OVCCON.OVSTRT bit is set.

All overlay block control registers are reset to their default values with the application reset. A special debug reset is not considered.

4.3.8.8 Global overlay control registers

Two registers globally control the overlay operation for all the cores:

- Overlay enable register OVCENABLE can be used to disable or enable data access overlay individually for each core
- Overlay control register OVCCON can be used to perform the following actions on a selected set of cores:
  - Concurrently enable or disable selected overlay blocks
  - Concurrently disable overlay blocks
  - Invalidate data cache

4.3.9 CPU timers

This section describes the TC1.8 CPU specific details of the temporal protection system.

4.3.9.1 Temporal protection system timers (TPS)

To assist in task run-time monitoring, the TC1.8 CPU implements a temporal protection system. This consists of three independent decrementing counters (TPS_TIMERn) arranged to generate a temporal asynchronous error trap (TAE - Class-4, Tin-7) on decrement to zero. A control register (TPS_CON) contains status and control bits for the temporal protection system.

The temporal protection system is enabled by the CORECON.TPROTEN register bit. The temporal protection registers are core special function registers (CSFRs), and are described in detail in the TriCore™ architecture manual.

The TC1.8 CPU decrements the TPS_TIMERn on each CPU clock. CPU clock control until either the count value reaches zero, or the timer is de-activated by writing zero to the TPS_TIMERn register. (CCON.DIV) has no effect on TPS_TIMERs decrement frequency. While the CPU clock is stopped (as happens in the CPU suspend state) then no decrementing occurs for this period.

4.3.10 Access protection and ENDINIT replacement

The TC1.8 CPU implements several access protection units (APUs), also known as safety protection, to provide protection against illegal or unintended bus access to the local memory system and control registers.

In order to provide access to the resources while reconfiguration is in progress, all APU instances in the TC1.8 CPU are configured through shadow registers. Each APU has an associated PROT. The shadow APU registers and their associated PROT are the only writeable registers in the APU.

Please refer to the Access Protection chapter for details on APU register configuration.

Table 91 Summary of APUs

Instance	Resource protected	AURIX™ TC3xx equivalent
ACCENDLMU	Ranged based APU protection with shadow configuration APU. 16 ranges can be configured to restrict accesses to the local DLMU to specific TAG, VMN and PRS	DLMU_SPROT_*
ACCENSPR	Ranged based APU protection with shadow configuration APU. 16 ranges can be configured to restrict accesses to the local DSPR/PSPR to specific TAG, VMN and PRS	SPR_SPROT_*
ACCENSFR	Range-less based APU protection with shadow configuration APU	SFR_SPROT_*
ACCENSTM	Virtualization not present or disabled only. Range-less based APU protection with shadow configuration APU	STM_ACCEN*
ACCENSTMVMx	Virtualization enabled only. Range-less based APU protection with shadow configuration APU	STM_ACCEN*
ACCENPFSFR	Range-less APU protection with shadow configuration APU	SFR_SPROT_*
ACCENPF	Ranged based APU protection with shadow configuration APU. 8 ranges can be configured to restrict accesses to the local PNVM to specific TAG, VMN and PRS. This APU will only provide read protection. Write to PNVM is not possible from this interface	LPB_SPROT_* (Previously range-less)

4.3.10.1 ENDINIT replacement

TC1.8 CPU PROT implementation

In AURIX™ TC4xx the control of the ENDINIT functionality is provided locally to the CPU by PROT instances (Please see the Access Protection chapter for background on the PROT functionality). With local control, the concepts of system wide ENDINIT and CPU ENDINIT, are no longer applicable.

PROT usage with no virtualization (or virtualization disabled)

To allow distinct resource control, resources are split across the following PROT instances:

Table 92 PROT for configurations without virtualization or with virtualization disabled

Instance	Registers covered	
PROTRSTE	KRST0, KRST1	Previously system ENDINIT. Separated from other PROT as the owner is likely to be a remote master
PROTSFRE	PMA0, PMA1, DCON0, PCON0, SEGEN, BIV, BTV, ISP	Previously CPU ENDINIT
PROTSFRE	STM_CLC	Previously system ENDINIT STM is now part of the CPU
PROTSFRSE	COMPAT, SMACON, CORECON, ACCENSFRCFG (APU)	Previously safety ENDINIT
PROTSTMSE	ACCENSTMCFG (APU)	Previously safety ENDINIT
PROTDLMUSE	ACCENDLMUCFG (APU)	Previously safety ENDINIT
PROTSPRSE	ACCENSPRCFG (APU)	Previously safety ENDINIT
PROTPFSFRE	PFIy_FLASHCON0, PFIy_FLASHCON1	Previously system ENDINIT
PROTPFSFRSE	ACCENPFSFRCFG (APU)	Previously safety ENDINIT
PROTPFSE	ACCENPFCFG (APU)	Previously safety ENDINIT

4.3.10.2 SPR protection

The TC1.8 CPU implements one PROTSPRSE, an ACCENSPRCFG APU, and 16 ACCENSPR range based APUs. To configure the APU, the implementation field of the PROTSPRSE is populated with a SEL index field to allow the software to select the range to be configured.

PROTSPRSE[11:8] = SEL.

All ACCENSPR registers are therefore read only.

4.3.10.3 DLMU protection

The TC1.8 CPU implements one PROTDLMUSE, an ACCENDLMUCFG APU, and 16 ACCENDLMU range based APUs. To configure the APU, the implementation field of the PROTDLMUSE is populated with a SEL index field to allow the software to select the range to be configured.

PROTDLMUSE[11:8] = SEL

All ACCENDLMU registers are therefore read only.

The DLMU is accessible both in segment 9_H and B_H.

In order to give the same protection to both addresses, the ACCENDLMUy_RGNUA and ACCENDLMUy_RGNLA registers will capture the segment 9_H address (bit 29 of the range registers is read-only).

4.3.10.4 SFR protection

The TC1.8 CPU implements one PROTSFRE, a PROTSFRSE, an ACCENSFRCFG APU, and one ACCENSFR range-less APU.

All ACCENSFR registers are therefore read only.

4.3.10.5 RESET protection

The TC1.8 CPU implements one PROTRSTE to protect from involuntary writes to the kernel reset registers. The registers external access is also protected by the ACCENSFR. A master must have access to the resource and disable the PROTRSTE lock in order to trigger a kernel reset.

4.3.10.6 STM protection

The TC1.8 CPU STM_ABS is protected by PROTSFRSE. The rest of the STM configuration is protected by its own PROT and APU.

Virtualization not present

In this configuration there is only one STM instance. The TC1.8 CPU provides a PROTSTMSE, an ACCENSTMCFG APU and one ACCENSTM range-less APU. The PROTSTMSE does not have a SEL field when virtualization is not present.

All ACCENSTM registers are therefore read only.

Virtualization present and enabled (for devices with CPU hardware support for virtualization)

In this configuration there is one STM instance per virtual machine. To provide a level of flexibility a separate APU is provided for each VM STM register set. The configuration of these VM specific APUs is achieved through a unique PROTSTMSE. The implementation specific field of the PROTSTMSE is populated with a SEL index field to allow the hypervisor software to select the VM specific ACCENSTM to be configured.

PROTSTMSE[10:8] = SEL.

All ACCENSTM registers are read only.

Virtualization present and disabled (for devices with CPU hardware support for virtualization)

In this configuration there is only one STM instance. The TC1.8 CPU provides a PROTSTMSE, an ACCENSTMCFG APU and one ACCENSTM range-less APU. The PROTSTMSE.SEL index must be set to 1 by the application software for the ACCENSTM to be configured.

PROTSTMSE[10:8] = 1.

All ACCENSTM registers are read only.

4.3.10.7 NVM protection

All external accesses to any local NVM resources are directed to the CPU S-SRI1 slave port. This includes NVM and registers. The TC1.8 CPU provides separate protections for these.

Register accesses

The TC1.8 CPU implements PROTPFSFRE , PROTPFSFRSE , ACCENPFSFRCFG and ACCENPFSFR to protect the configuration of the FLASHCON SFRs and accesses to the user and test registers associated with the local NVM banks.

NVM and erase counters

The TC1.8 CPU implements PROTPFSE, ACCENPFCFG and 8 ACCENPF range based APU to protect from read access to the local portions of the NVM, shared between both banks, as well as the erase counter sections. To

configure the APU, the implementation field of the PROTPFSE is populated with a SEL index field to allow the software to select the range to be configured.

PROTPFSE[10:8] = SEL

The NVM is accessible both in segment 8_H and A_H. In order to give the same protection to both addresses, the ACCENPFy_RGNLA and ACCENPFy_RGNUA registers will capture the segment 8_H address (bit 29 of the range registers is read-only).

All ACCENSFRPF and ACCENPF registers are therefore read only.

4.3.10.8 VM number

To enable a slave device to differentiate between different master VMs the bus fabric is extended to send the current VM number with all accesses. The VM number will be stored in the data tag to provide the correct VM number in the case of dirty data being written back.

4.3.10.9 PRS value

To enable a slave device to differentiate between different masters executing with different PRS values the bus fabric is extended to send the current PRS value with all accesses. All dirty cache data will be written back with a default PRS value (PRS=0 and PRS_EN=0) when COMPAT.TAGPRS is set or with the captured PRS and PRS_EN=1 when the bit is cleared.

4.3.10.10 Secure master value

To enable the correct programming of the PROT state machine, the bus fabric is extended with a secure_master indicator. This is additional to the master TAG ID. The application instances of the TriCore™ CPU will drive the secure master to a value of 0.

The CPUcs instance will be configurable and will drive secure_master to the value configured by the CSSCU.

4.3.11 FPI interfaces

The TC1.8 CPU implements a maximum of two FPI interfaces. It is a device specific feature for each CPU. The allocation of addresses to the FPI interfaces is fixed for a device.

If the second FPI is not implemented, accesses to the address region will be sent to the SRI interconnect.

4.4 Program NVM interface (CPU-CFI)

The TC1.8 CPU processor provides high bandwidth interfaces to the banks of NVM allocated to the processor instance.

4.4.1 Feature list

- Double program NVM interface for increased performance
- Homogenous SOTA A/B swap performance

4.4.2 Functional overview

The following sections give an overview of the TC1.8 CPU program NVM interface Implementation.

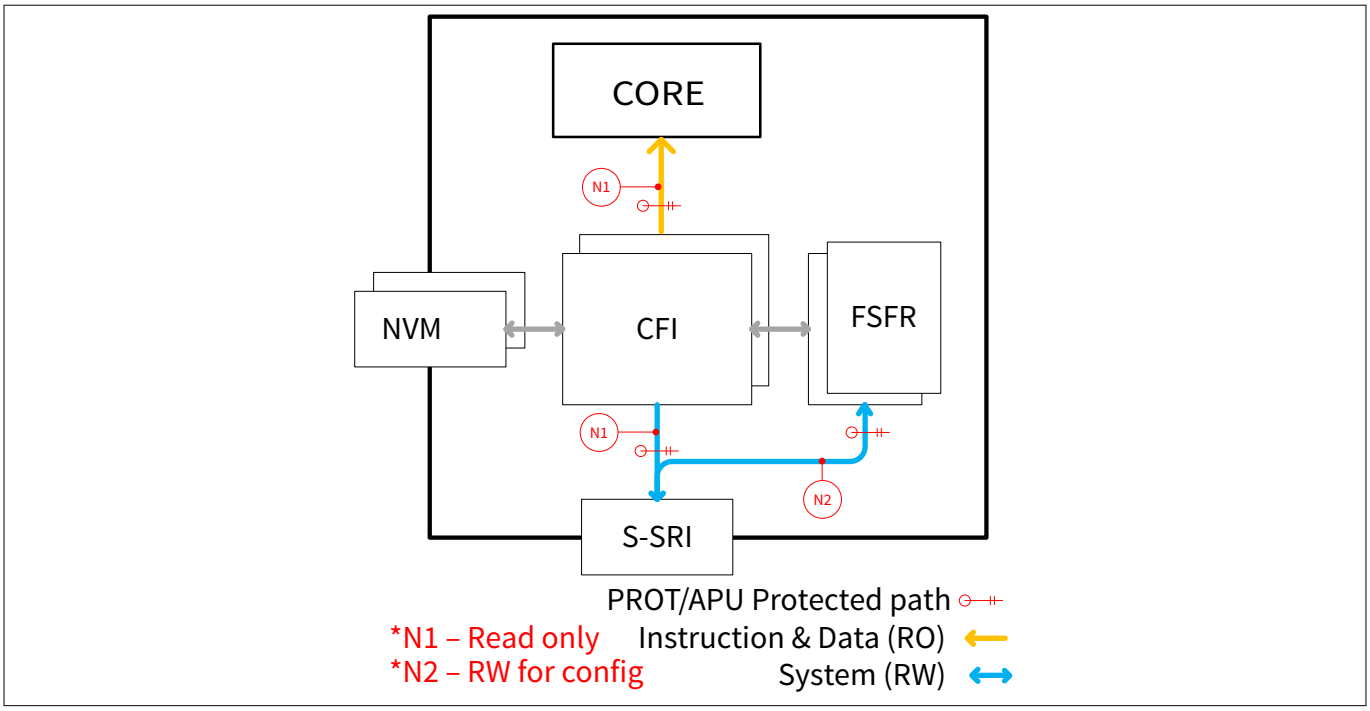


Figure 31 CPU-PFI block diagram

The program NVM interface connects the core and interconnect to the local portion of NVM and FSRS's.

4.4.3 Functional description

This section details the implementation specific features of the program NVM interface.

4.4.3.1 Program NVM interface

***Note**: For naming compatibility with earlier devices the historical names FLASHCON, PFI and FPB are preserved, despite being NVM technology agnostic*

4.4.3.1.1 FSFR range

All registers related to the function of the NVM attached to a CPU are located in the FSFR range. All the registers are SFR's and can be configured using load store instructions.

Table 93 FSFR address ranges

Address range	Size	Function
$F8400000_H$ – $F8400FFF_H$	4096 bytes	CPU0.FSFR (NVM Interface SFR)

Related information

CSFR, SFR, and FSFR address map and external access on page 153

4.4.3.1.2 FLASHCON renaming and relocation

Some CPU instances have more than one NVM interface therefore FLASHCON0 and FLASHCON1 are provided by NVM interface (PFI) and are renamed accordingly PFIy_FLASHCON0 and PFIy_FLASHCON1.

Table 94 ADDRCONFIG

Short name	Offset	Description
PFIy_FLASHCON0	200_H+y*8	CPU NVM Interface y Configuration register 0
PFIy_FLASHCON1	204_H+y*8	CPU NVM Interface y Configuration register 1

4.4.3.1.3 NVM prefetch buffer (FPB)

The prefetch path instantiates prefetch buffer(s) to speculatively fetch data. If the address of an access matches any of the FPBs then the data is returned from this FPB.

FPB configuration

The CPU contains a set of FLASHCON registers for configuring its local PFI. Each PFI instantiates five FPBs. An FPB may be assigned to an on chip master (see table below) by software programming the 6-bit master TAG identification number to the appropriate fields of the PFIy_FLASHCON0 register.

Table 95 FPB assignment

FPB	Description
0	Permanently assigned to the local CPU PMBI master TAG identification number
1-4	Assigned to an on chip master by software programming the PFIy_FLASHCON0 register with the appropriate 6-bit master TAG identification number. ***Note**: To reduce power consumption prefetching from NVM, an individual FPB may be disabled by software assigning the FPB to the reserved 6-bit master TAG identification number* ***Note**: The role of the FPB is to attempt to request the most likely NVM entry that could be requested by a certain master. The PFIy_FLASHCON0 is used to associate the FPB with the master TAG IDs that need to make use of the FPB. This means that each buffer can only be updated/filled by a matching TAG request, but the data is usable by any master asking for a particular address.*

Performance may be optimized by assigning a maximum of two FPBs to one on chip master. If two FPBs are assigned then adjacent FPBs must be assigned, for example:

- Local CPU PMBI on chip master: FPB0 assigned by default and software may additionally assign FPB1 for optimal local CPU performance
- Other on chip masters: software may assign FPB1 and FPB2, FPB2 and FPB3, or FPB3 and FPB4

Related information

Local memory details on page 147
TC4Dx CPU configuration on page 396

4.4.3.1.4 Erase counter and register accesses

The PFI to CPU point-to-point connection may be used to read the erase counter (NVM Flash devices), PFRWB registers (NVM Flash devices), or PMUR registers (NVM RRAM devices).

The access size must be an 32-bit word else the access will error.

4.4.3.1.5 NVM interface error capture registers

To monitor and debug violations of PROT and or APU protection during accesses to the CPU NVM interface, two registers (CFIERR and CFIERRAR) are provided. These registers allow the software to localize the source of the last error detected by the CFI. The registers are updated when an APU violation is detected and the CFIERR.ED bit is zero. On update the CFIERR.ED bit is set to one and remains set until cleared by software.

Table 96 Error capture registers

Short name	Offset	Description
CFIERR	250_H	CFI Error register
CFIERRAR	254_H	CFI Error address register

Error due to PROT and APU violation.

A PROT or APU error detected during an access to the local NVM range or to the FSFR range will result in the update of the CFIERR register. The register will capture the properties of the erroring transaction. E_TAG field is updated with the transaction master TAG ID. The E_VMN and E_VMN_EN fields are updated if VMN_EN is set or cleared otherwise. The E_PRS and E_PRS_EN fields are updated if PRS_EN is set or cleared otherwise. The CFIERR.ED bit is set. The CFIERRAR register is updated with the address of the access.

Integrity error due to ECC error during request address phase

When an ECC error is detected during a request address phase then the CFIERR.ED bit is set. The CFIERRAR register is updated with address 00000000_H.

Integrity error due to ECC error during a write access to the FSFR

When an ECC error is detected during a write data phase then the CFIERR.ED bit is set. The CFIERRAR register is updated with address of the access.

4.4.3.1.6 Power domain switch

Apart from CPU0, some CPUs may be capable of being powered down in order to reduce power consumption in parts of the system. The local banks of NVM associated with such CPUs will be available regardless of the CPU being powered up or down.

Related information

Power supply rails and distribution on page 5948

4.5 System timer module (CPU-STM)

Each TC1.8 CPU includes a System Timer (STM). The STM is designed for system timing applications requiring both high precision and a long period.

4.5.1 Feature list

The STM has the following features:

- Free-running 64-bit counter
- All 64-bits can be read synchronously
- Flexible service request generation based on compare match with partial ABS_TIMER or REL_TIMER fields
- Counting starts automatically after an system reset
- Support for virtualization with eight relative timers to support eight virtual machines

4.5.2 Functional overview

The following sections give an overview of the TC1.8 CPU STM implementation.

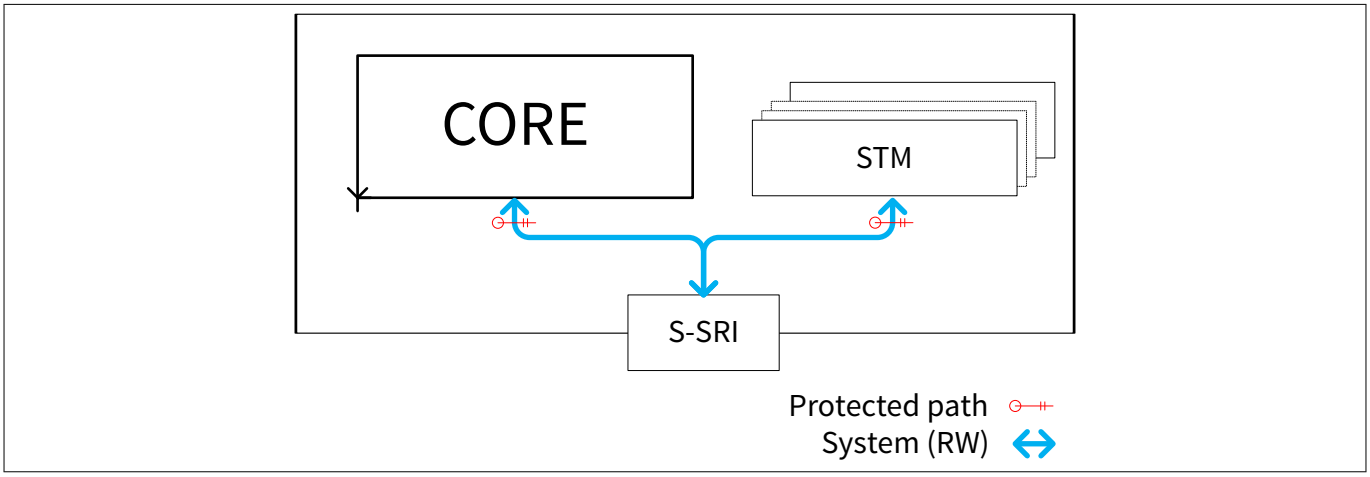


Figure 32 CPU-STM block diagram

The STM is accessed via the SRI slave interface.

4.5.3 Functional description

In AURIX™ TC4xx, the STM is part of the CPU and its addresses are contained within the CPU SFR space.

When migrating from previous generations the following aspects should be considered:

- STM is connected to system reset (therefore it is not affected by CPU kernel reset functionality)
- The STM ID register is removed
- The existing 64-bit timer is renamed STM_ABS[63:0]
- The interrupt control and status register is split into two registers
  - STM_ISCR – Interrupt set/clear register
  - STM_ISR – Interrupt status register
- The STM is clocked at f_{stm} to allow the timer to be consistent independently of the CPU frequency variation

4.5.3.1 Reading the timers

The entire 64-bit timer can only be read coherently using a 64-bit access (for example a LD.D instruction, or DMA single data transfer double word access). This avoids the need to have non-interruptible code if reading two component fields, which provides a performance benefit for real-time systems.

The STM can be optionally disabled for power-saving purposes, or suspended for debugging purposes. In suspend mode, the STM clock is stopped but all registers are still readable.

4.5.3.2 Compare register operation

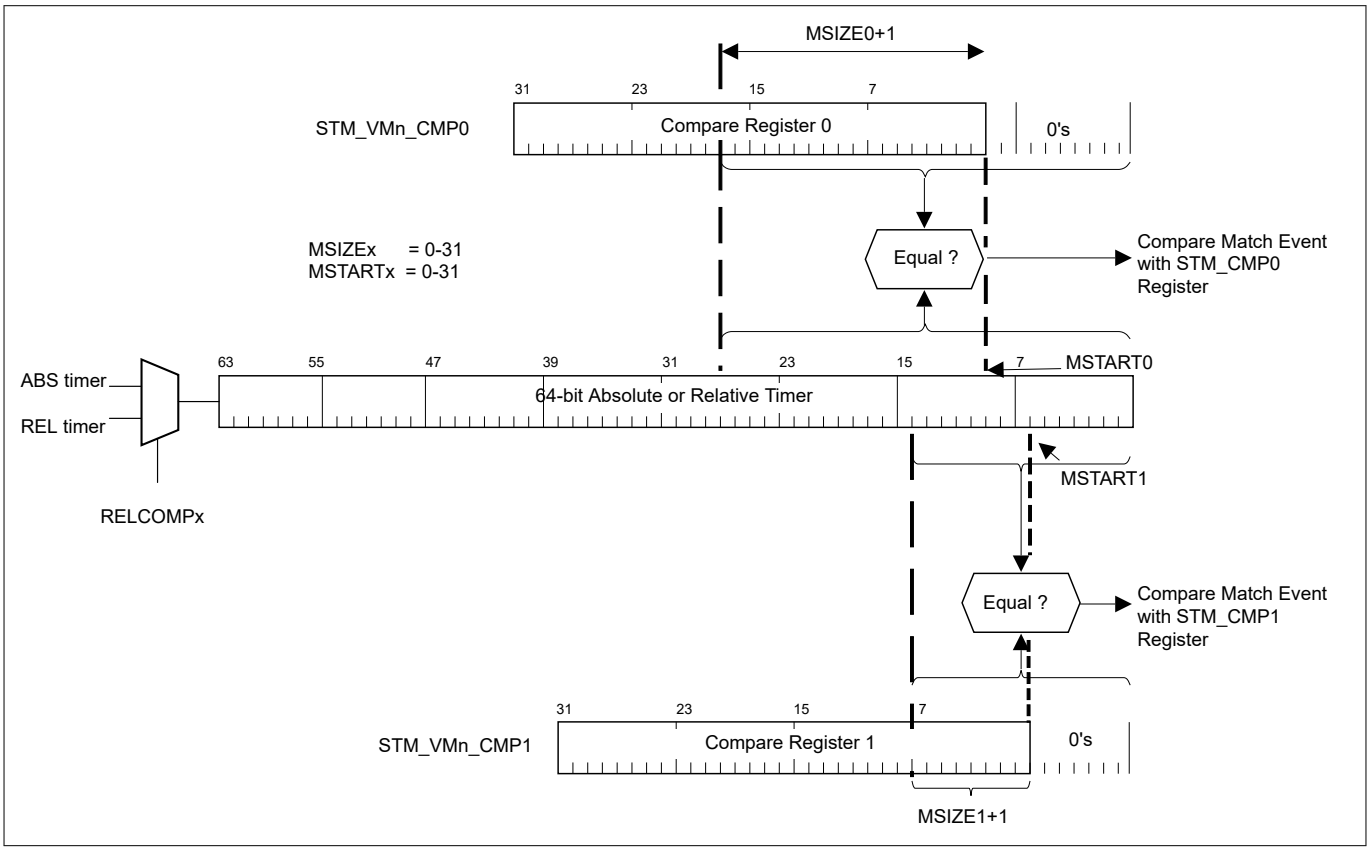


Figure 33 Compare mode operation

A compare operation with MSIZE not equal 11111_B always implies that the compared value as stored in the CMP register is right-extended with zeros. The timer value STM is taken either from STM_ABS or STM_RELTIM depending on the value of RELCOMP. This means that in the example of the figure above, the compare register content CMP0[17:0] plus ten zero bits right-extended is compared with STM[27:0] with STM[9:0] compared against 000_H. In case of register CMP1, STM[14:0] with STM[6:0] compared against 00_H are compared with CMP1[7:0] plus seven zero bits right-extended.

4.5.3.3 Compare match interrupt control

The compare match interrupt control logic is shown in the figure below. Each STM_CMPc (c=0-1) register has its compare match interrupt request flag (ICR.CMPxIR) that is set by hardware on a compare match event. The interrupt request flags can be set (ISCR.CMPxIRS) or cleared (ISCR.CMPxIRR) by software. Note that setting ICR.CMPxIR by writing a 1 into ISCR.CMPxIRS does not generate an interrupt at STMIRx. The compare match interrupts from CMP0 and CMP1 can be further directed by ICR.CMPxOS to either output signal STMIR0 or STMIR1.

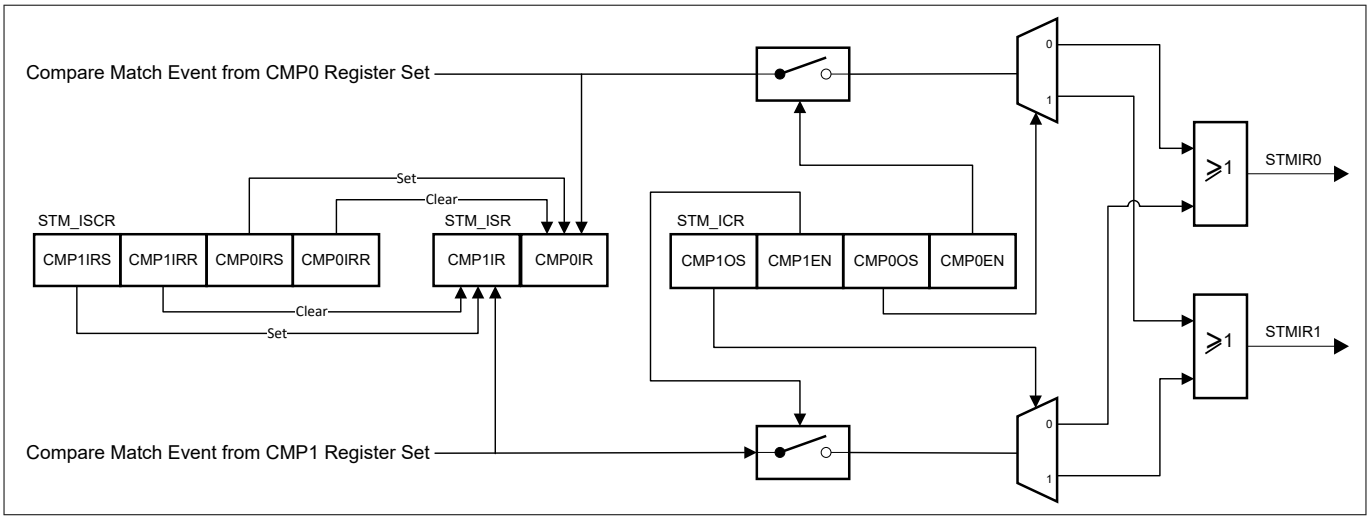


Figure 34 Interrupt Control

The compare match interrupt flags ICR.CMPxIR are immediately set after an STM reset operation, caused by a compare match event with the reset values of the STM_ABS and the compare registers STM_CMPc. This does not directly generate compare match interrupts because the compare match interrupts are automatically disabled after an STM reset operation (ICR.CMPxEN = 0). Therefore, before enabling a compare match interrupt after an STM system reset, the software should configure the STM and modify the reset values of the compare registers. Otherwise, undesired compare match interrupt events are triggered. The ISR.CMPxIR flags which are set after an STM reset can be cleared by writing register ISCR with CMPxIRR set.

4.5.3.4 STM as reset trigger

A compare match triggered by an CMP0 or CMP1 event can generate a reset in the system. The reset has to be enabled for each STM module individually in register SMM_RSTTRIGCTRLB. In case the STM compare match is used to trigger an application reset, the STM comparator time duration should be chosen adequately large enough that the application can service the STM after reset release (for example 5 ms). This is to avoid consecutive application resets being triggered by STM as it was not serviced after reset.

4.5.3.5 STM service request mapping

The STM supports two timer interrupts per VM. When virtualization is present and enabled, this results in 16 timer interrupts. When virtualization is not present, or disabled, only 2 timer interrupts are used. The following table maps the timer interrupts to their respective service request register

Table 97 Service request register mapping

Virtual machine number	Interrupt number	Corresponding IR SRC register
VM0	Service request 0 (STMIR0) Service request 1 (STMIR1)	IR_SRC_STMCPUwSR0 (w=0-5) IR_SRC_STMCPUwSR1 (w=0-5)
VM1[1)]	Service request 0 (STMIR0) Service request 1 (STMIR1)	IR_SRC_STMCPUwSR2 (w=0-5) IR_SRC_STMCPUwSR3 (w=0-5)
VM2	Service request 0 (STMIR0) Service request 1 (STMIR1)	IR_SRC_STMCPUwSR4 (w=0-5) IR_SRC_STMCPUwSR5 (w=0-5)

(table continues...)

Table 97 (continued) Service request register mapping

Virtual machine number	Interrupt number	Corresponding IR SRC register
VM3	Service request 0 (STMIR0) Service request 1 (STMIR1)	IR_SRC_STMCPUwSR6 (w=0-5) IR_SRC_STMCPUwSR7 (w=0-5)
VM4	Service request 0 (STMIR0) Service request 1 (STMIR1)	IR_SRC_STMCPUwSR8 (w=0-5) IR_SRC_STMCPUwSR9 (w=0-5)
VM5	Service request 0 (STMIR0) Service request 1 (STMIR1)	IR_SRC_STMCPUwSR10 (w=0-5) IR_SRC_STMCPUwSR11 (w=0-5)
VM6	Service request 0 (STMIR0) Service request 1 (STMIR1)	IR_SRC_STMCPUwSR12 (w=0-5) IR_SRC_STMCPUwSR13 (w=0-5)
VM7	Service request 0 (STMIR0) Service request 1 (STMIR1)	IR_SRC_STMCPUwSR14 (w=0-5) IR_SRC_STMCPUwSR15 (w=0-5)

1) When virtualization is not present or disabled, only SR2 and SR3 are used

4.5.3.6 STM register offsets and virtualization support

This section provides an overview of the location of the STM register offsets.

Table 98 Virtualization not present or disabled (1 STM instance), or guest VM software

SFR	HRA	HRB	HRHV	Offset	Description
X				00000_H	STM_CLC
X				00004_H	STM_OCS
X				00020_H	STM_ABS
X				00120_H	STM_CMP0
X				00124_H	STM_CMP1
X				00128_H	STM_CMCON
X				$0012C_H$	STM_ICR
X				00130_H	STM_ISCR
X				00134_H	STM_ISR
X				00138_H	STM_RELTIM[1)]

1) Provided for software compatibility. When virtualization is disabled, the STM_RELTIM and STM_ABS are aliased.

When virtualization is enabled there is one STM instance per guest virtual machine plus another instance for the hypervisor (8 instances in total). Only the hypervisor requires knowledge of the bus addresses of the different instances. Guest VM software must use the fact that the hardware provides address morphing.

Table 99 Virtualization enabled (8 VMn STM instances), hypervisor software or other master

SFR	HRA	HRB	HRHV	Offset	Description
X				00000_H	STMHV_CLC
X				00004_H	STMHV_OCS
X				00020_H	STMHV_ABS
X				00100_H+n*20_H	STMHV_VMn_CMP0 (n=0-7)
X				00104_H+n*20_H	STMHV_VMn_CMP1 (n=0-7)
X				00108_H+n*20_H	STMHV_VMn_CMCON (n=0-7)
X				$0010C_H$+n*20_H	STMHV_VMn_ICR (n=0-7)
X				00110_H+n*20_H	STMHV_VMn_ISCR (n=0-7)
X				00114_H+n*20_H	STMHV_VMn_ISR (n=0-7)
X				00118_H+n*20_H	STMHV_VMn_RELTIM (n=0-7)

***Note**: When virtualization is not present of disabled, only VM1 register set is accessible.*

4.6 Registers

4.6.1 Registers of CPU

4.6.1.1 Register overview - access mode glossary

Table 100 Register overview - access mode glossary

Keyword	Description
DLMUSE	Access protection using PROT register PROTDLMUSE.
RSTE	Access protection using PROT register PROTRSTE.
SFRE	Access protection using PROT register PROTSFRE.
SFRSE	Access protection using PROT register PROTSFRSE.
SPRSE	Access protection using PROT register PROTSPRSE.
STMSE	Access protection using PROT register PROTSTMSE.
APU-P0	Protection group consisting of registers ACCENSFRCFG_WRA, ACCENSFRCFG_WRB, ACCENSFRCFG_RDA, ACCENSFRCFG_RDB, ACCENSFRCFG_VM, ACCENSFRCFG_PRS, ACCENSFR_WRA, ACCENSFR_WRB, ACCENSFR_RDA, ACCENSFR_RDB, ACCENSFR_VM, ACCENSFR_PRS.
P0	Access protection using *APU-P0* registers.
APU-P1	Protection group consisting of registers ACCENSTMCFG_WRA, ACCENSTMCFG_WRB, ACCENSTMCFG_RDA, ACCENSTMCFG_RDB, ACCENSTMCFG_VM, ACCENSTMCFG_PRS, ACCENSTMVMy_WRA, ACCENSTM_WRA, ACCENSTMVMy_WRB, ACCENSTM_WRB, ACCENSTMVMy_RDA, ACCENSTM_RDA, ACCENSTMVMy_RDB, ACCENSTM_RDB, ACCENSTMVMy_VM, ACCENSTM_VM, ACCENSTMVMy_PRS, ACCENSTM_PRS.

(table continues...)

Table 100 (continued) Register overview - access mode glossary

Keyword	Description
P1	Access protection using *APU-P1* registers.
APU-P2	Protection group consisting of registers ACCENSPRCFG_WRA, ACCENSPRCFG_WRB, ACCENSPRCFG_RDA, ACCENSPRCFG_RDB, ACCENSPRCFG_VM, ACCENSPRCFG_PRS, ACCENSPRCFG_RGNLA, ACCENSPRCFG_RGNUA, ACCENSPRy_WRA, ACCENSPRy_WRB, ACCENSPRy_RDA, ACCENSPRy_RDB, ACCENSPRy_VM, ACCENSPRy_PRS, ACCENSPRy_RGNLA, ACCENSPRy_RGNUA.
P2	Access protection using *APU-P2* registers.
APU-P3	Protection group consisting of registers ACCENDLMUCFG_WRA, ACCENDLMUCFG_WRB, ACCENDLMUCFG_RDA, ACCENDLMUCFG_RDB, ACCENDLMUCFG_VM, ACCENDLMUCFG_PRS, ACCENDLMUCFG_RGNLA, ACCENDLMUCFG_RGNUA, ACCENDLMUy_WRA, ACCENDLMUy_WRB, ACCENDLMUy_RDA, ACCENDLMUy_RDB, ACCENDLMUy_VM, ACCENDLMUy_PRS, ACCENDLMUy_RGNLA, ACCENDLMUy_RGNUA.
P3	Access protection using *APU-P3* registers.
32	Access only when using 32-bit width.
STP	Access only during start-up, before start-up protection is set (SMM_STCON.ST_PROT=1).
PROT	Access restrictions as defined in the PROT register access rules.
DEN	Description can be found in global access mode definition.
64	Access only when using 64-bit width.
nBE	Indicates that no Bus Error is generated when accessing this address range, even though it is either an access to an undefined address or the access does not follow the given rules.

4.6.1.2 Register overview - CPU domain SRI (ascending offset address)

Table 101 Register overview - CPU domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
STM_CLC	CPUx system timer clock control register	00000_H	P0, 32	P0, SV, SFRE, 32	System Reset	351
STMHV_CLC	CPUx system timer clock control register, hypervisor view	00000_H	P0, 32	P0, SV, SFRE, 32	System Reset	351
STM_OCS	CPUx OCDS control and status register	00004_H	P0, 32	P0, SV, 32, DEN	Debug Reset	352
STMHV_OCS	CPUx OCDS control and status register, hypervisor view	00004_H	P0, 32	P0, SV, 32, DEN	Debug Reset	352
STM_ABS	CPUx system timer absolute timer	00020_H	(P0, 32)| (P0, 64)	P0, SV, 64, SFRSE	System Reset	353
STMHV_ABS	CPUx system timer absolute timer, hypervisor view	00020_H	(P0, 32)| (P0, 64)	P0, SV, 64, SFRSE	System Reset	353

(table continues...)

Table 101 (continued) Register overview - CPU domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
STMHV_VMn_CMPc	CPUx VMn system timer compare register c, hypervisor view	00100_H+n*20_H+c*4	P1, 32	P1, SV, 32	System Reset	354
STMHV_VMn_CMCON	CPUx VMn system timer compare match control register, hypervisor view	00108_H+n*20_H	P1, 32	P1, SV, 32	System Reset	355
STMHV_VMn_ICR	CPUx VMn system timer interrupt control register, hypervisor view	$0010C_H$+n*20_H	P1, 32	P1, SV, 32	System Reset	356
STMHV_VMn_ISCR	CPUx VMn system timer interrupt set/clear register, hypervisor view	00110_H+n*20_H	P1, 32	P1, SV, 32	System Reset	357
STMHV_VMn_ISR	CPUx VMn system timer interrupt status register, hypervisor view	00114_H+n*20_H	P1, 32	P1, SV, 32	System Reset	358
STMHV_VMn_RELTIM	CPUx VMn system timer relative timer, hypervisor view	00118_H+n*20_H	(P1, 32)|(P1, 64)	P1, SV, 64	System Reset	359
STM_CMPc	CPUx system timer compare register c	00120_H+c*4	P1, 32	P1, SV, 32	System Reset	354
STM_CMCON	CPUx system timer compare match control register	00128_H	P1, 32	P1, SV, 32	System Reset	355
STM_ICR	CPUx system timer interrupt control register	$0012C_H$	P1, 32	P1, SV, 32	System Reset	356
STM_ISCR	CPUx system timer interrupt set/clear register	00130_H	P1, 32	P1, SV, 32	System Reset	357
STM_ISR	CPUx system timer interrupt status register	00134_H	P1, 32	P1, SV, 32	System Reset	358
STM_RELTIM	CPUx system timer relative timer	00138_H	(P1, 32)|(P1, 64)	P1, SV, 64	System Reset	359
KRST0	CPUx reset register 0	$0D000_H$	P0, 32	RSTE, P0, SV, 32	Kernel Reset	297
KRST1	CPUx reset register 1	$0D004_H$	P0, 32	RSTE, P0, SV, 32	Kernel Reset	298
KRSTCLR	CPUx reset clear register	$0D008_H$	P0, 32	RSTE, P0, SV, 32	Kernel Reset	299
PROTSPRSE	CPUx PROTSPR register safe endinit	$0E008_H$	32	SV, 32, PROT	Application Reset	305
ACCENSPRCFG_WRA	CPUx shadow SPR, write access enable register A	$0E020_H$	32	SPRSE, SV, 32	See 307	307
ACCENSPRCFG_WRB	CPUx shadow SPR, write access enable register B	$0E024_H$	32	SPRSE, SV, 32	See 308	308

(table continues...)

Table 101 (continued) Register overview - CPU domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ACCENSPRCFG_RDA	CPUx shadow SPR, read access enable register A	$0E028_H$	32	SPRSE, SV, 32	See 308	308
ACCENSPRCFG_RDB	CPUx shadow SPR, read access enable register B	$0E02C_H$	32	SPRSE, SV, 32	Application Reset	309
ACCENSPRCFG_VM	CPUx shadow SPR, VM access enable register	$0E030_H$	32	SPRSE, SV, 32	Application Reset	310
ACCENSPRCFG_PRS	CPUx shadow SPR, PRS access enable register	$0E034_H$	32	SPRSE, SV, 32	Application Reset	310
ACCENSPRCFG_RGNLA	CPUx shadow SPR, region lower address register	$0E038_H$	32	SPRSE, SV, 32	Application Reset	311
ACCENSPRCFG_RGNUA	CPUx shadow SPR, region upper address register	$0E03C_H$	32	SPRSE, SV, 32	Application Reset	312
PROTDLMUSE	CPUx PROTDLMU register safe endinit	$0E048_H$	32	SV, 32, PROT	Application Reset	312
ACCENDLMUCFG_WRA	CPUx shadow DLMU, write access enable register A	$0E060_H$	32	DLMUSE, SV, 32	Application Reset	314
ACCENDLMUCFG_WRB	CPUx shadow DLMU, write access enable register B	$0E064_H$	32	DLMUSE, SV, 32	Application Reset	315
ACCENDLMUCFG_RDA	CPUx shadow DLMU, read access enable register A	$0E068_H$	32	DLMUSE, SV, 32	Application Reset	315
ACCENDLMUCFG_RDB	CPUx shadow DLMU, read access enable register B	$0E06C_H$	32	DLMUSE, SV, 32	Application Reset	316
ACCENDLMUCFG_VM	CPUx shadow DLMU, VM access enable register	$0E070_H$	32	DLMUSE, SV, 32	Application Reset	316
ACCENDLMUCFG_PRS	CPUx shadow DLMU, PRS access enable register	$0E074_H$	32	DLMUSE, SV, 32	Application Reset	317
ACCENDLMUCFG_RGNLA	CPUx shadow DLMU, region lower address register	$0E078_H$	32	DLMUSE, SV, 32	Application Reset	317
ACCENDLMUCFG_RGNUA	CPUx shadow DLMU, region upper address register	$0E07C_H$	32	DLMUSE, SV, 32	Application Reset	318
PROTSFRE	CPUx PROTSFR register endinit	$0E080_H$	32	SV, 32, PROT	Application Reset	318
PROTSFRSE	CPUx PROTSFR register safe endinit	$0E088_H$	32	SV, 32, PROT	Application Reset	320
ACCENSFRCFG_WRA	CPUx shadow SFR, write access enable register A	$0E0A0_H$	32	SFRSE, SV, 32	See 322	322
ACCENSFRCFG_WRB	CPUx shadow SFR, write access enable register B	$0E0A4_H$	32	SFRSE, SV, 32	See 323	323
ACCENSFRCFG_RDA	CPUx shadow SFR, read access enable register A	$0E0A8_H$	32	SFRSE, SV, 32	See 323	323
ACCENSFRCFG_RDB	CPUx shadow SFR, read access enable register B	$0E0AC_H$	32	SFRSE, SV, 32	See 324	324

(table continues...)

Table 101 (continued) Register overview - CPU domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ACCENSFRCFG_VM	CPUx shadow SFR, VM access enable register	$0E0B0_H$	32	SFRSE, SV, 32	Application Reset	325
ACCENSFRCFG_PRS	CPUx shadow SFR, PRS access enable register	$0E0B4_H$	32	SFRSE, SV, 32	Application Reset	325
PROTRSTE	CPUx PROTRST register endinit	$0E0C0_H$	32	SV, 32, PROT	See 326	326
PROTSTMSE	CPUx PROTSTM Register Safe Endinit	$0E0D8_H$	32	SV, 32, PROT	Application Reset	328
ACCENSTMCFG_WRA	CPUx shadow STM, write access enable register A	$0E0E0_H$	32	STMSE, SV, 32	Application Reset	330
ACCENSTMCFG_WRB	CPUx shadow STM, write access enable register B	$0E0E4_H$	32	STMSE, SV, 32	Application Reset	330
ACCENSTMCFG_RDA	CPUx shadow STM, read access enable register A	$0E0E8_H$	32	STMSE, SV, 32	Application Reset	331
ACCENSTMCFG_RDB	CPUx shadow STM, read access enable register B	$0E0EC_H$	32	STMSE, SV, 32	Application Reset	331
ACCENSTMCFG_VM	CPUx shadow STM, VM access enable register	$0E0F0_H$	32	STMSE, SV, 32	Application Reset	332
ACCENSTMCFG_PRS	CPUx shadow STM, PRS access enable register	$0E0F4_H$	32	STMSE, SV, 32	Application Reset	333
ACCENSPRy_WRA	CPUx SPR range y, write access enable register A	$0E200_H$+ y*20_H	32	SPRSE, SV, 32	See 333	333
ACCENSPRy_WRB	CPUx SPR range y, write access enable register B	$0E204_H$+ y*20_H	32	SPRSE, SV, 32	See 334	334
ACCENSPRy_RDA	CPUx SPR range y, read access enable register A	$0E208_H$+ y*20_H	32	SPRSE, SV, 32	See 335	335
ACCENSPRy_RDB	CPUx SPR range y, read access enable register B	$0E20C_H$+ y*20_H	32	SPRSE, SV, 32	Application Reset	335
ACCENSPRy_VM	CPUx SPR range y, VM access enable register	$0E210_H$+ y*20_H	32	SPRSE, SV, 32	Application Reset	336
ACCENSPRy_PRS	CPUx SPR range y, PRS access enable register	$0E214_H$+ y*20_H	32	SPRSE, SV, 32	Application Reset	337
ACCENSPRy_RGNLA	CPUx SPR range y, region lower address register	$0E218_H$+ y*20_H	32	SPRSE, SV, 32	Application Reset	337
ACCENSPRy_RGNUA	CPUx SPR range y, region upper address register	$0E21C_H$+ y*20_H	32	SPRSE, SV, 32	Application Reset	338
ACCENDLMUy_WRA	CPUx DLMU range y, write access enable register A	$0E400_H$+ y*20_H	32	DLMUSE, SV, 32	Application Reset	338
ACCENDLMUy_WRB	CPUx DLMU range y, write access enable register B	$0E404_H$+ y*20_H	32	DLMUSE, SV, 32	Application Reset	339
ACCENDLMUy_RDA	CPUx DLMU range y, read access enable register A	$0E408_H$+ y*20_H	32	DLMUSE, SV, 32	Application Reset	339

(table continues...)

Table 101 (continued) Register overview - CPU domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ACCENDLMUy_RDB	CPUx DLMU range y, read access enable register B	$0E40C_H$+ y*20_H	32	DLMUSE, SV, 32	Application Reset	340
ACCENDLMUy_VM	CPUx DLMU range y, VM access enable register	$0E410_H$+ y*20_H	32	DLMUSE, SV, 32	Application Reset	340
ACCENDLMUy_PRS	CPUx DLMU range y, PRS access enable register	$0E414_H$+ y*20_H	32	DLMUSE, SV, 32	Application Reset	341
ACCENDLMUy_RGNLA	CPUx DLMU range y, region lower address register	$0E418_H$+ y*20_H	32	DLMUSE, SV, 32	Application Reset	342
ACCENDLMUy_RGNUA	CPUx DLMU range y, region upper address register	$0E41C_H$+ y*20_H	32	DLMUSE, SV, 32	Application Reset	342
ACCENSFR_WRA	CPUx SFR, write access enable register A	$0E600_H$	32	SFRSE, SV, 32	See 343	343
ACCENSFR_WRB	CPUx SFR, write access enable register B	$0E604_H$	32	SFRSE, SV, 32	See 343	343
ACCENSFR_RDA	CPUx SFR, read access enable register A	$0E608_H$	32	SFRSE, SV, 32	See 344	344
ACCENSFR_RDB	CPUx SFR, read access enable register B	$0E60C_H$	32	SFRSE, SV, 32	See 345	345
ACCENSFR_VM	CPUx SFR, VM access enable register	$0E610_H$	32	SFRSE, SV, 32	Application Reset	345
ACCENSFR_PRS	CPUx SFR, PRS access enable register	$0E614_H$	32	SFRSE, SV, 32	Application Reset	346
ACCENSTMVMy_WRA	CPUx VMy STM, write access enable register A, virtualization view	$0E700_H$+ y*20_H	32	STMSE, SV, 32	Application Reset	347
ACCENSTMVMy_WRB	CPUx VMy STM, write access enable register B, virtualization view	$0E704_H$+ y*20_H	32	STMSE, SV, 32	Application Reset	347
ACCENSTMVMy_RDA	CPUx VMy STM, read access enable register A, virtualization view	$0E708_H$+ y*20_H	32	STMSE, SV, 32	Application Reset	348
ACCENSTMVMy_RDB	CPUx VMy STM, read access enable register B, virtualization view	$0E70C_H$+ y*20_H	32	STMSE, SV, 32	Application Reset	349
ACCENSTMVMy_VM	CPUx VMy STM, VM access enable register, virtualization view	$0E710_H$+ y*20_H	32	STMSE, SV, 32	Application Reset	350
ACCENSTMVMy_PRS	CPUx VMy STM, PRS access enable register, virtualization view	$0E714_H$+ y*20_H	32	STMSE, SV, 32	Application Reset	350
ACCENSTM_WRA	CPUx STM, write access enable register A	$0E720_H$	32	STMSE, SV, 32	Application Reset	347

(table continues...)

Table 101 (continued) Register overview - CPU domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ACCENSTM_WRB	CPUx STM, write access enable register B	$0E724_H$	32	STMSE, SV, 32	Application Reset	347
ACCENSTM_RDA	CPUx STM, read access enable register A	$0E728_H$	32	STMSE, SV, 32	Application Reset	348
ACCENSTM_RDB	CPUx STM, read access enable register B	$0E72C_H$	32	STMSE, SV, 32	Application Reset	349
ACCENSTM_VM	CPUx STM, VM access enable register	$0E730_H$	32	STMSE, SV, 32	Application Reset	350
ACCENSTM_PRS	CPUx STM, PRS access enable register	$0E734_H$	32	STMSE, SV, 32	Application Reset	350
OSEL	CPUx overlay range select register	$0FB00_H$	P0, 32	P0, SV, 32	Kernel Reset	299
RABRi	CPUx redirected address base register i	$0FB10_H$+ i*12	P0, 32	P0, SV, 32	Kernel Reset	300
OTARi	CPUx overlay target address register i	$0FB14_H$+ i*12	P0, 32	P0, SV, 32	Kernel Reset	302
OMASKi	CPUx overlay mask register i	$0FB18_H$+ i*12	P0, 32	P0, SV, 32	Kernel Reset	303
HRA_SEGEN	CPUx SRI error generation register	11030_H	P0, 32	SFRE, P0, SV, 32	Application Reset	255
HRA_LCLTEST	CPUx lockstep test register	11040_H	P0, 32	P0, SV, 32	Cold PowerOn Reset	256
HRA_PMA0	CPUx data access cacheability register	18100_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	232
HRA_PMA1	CPUx code access cacheability register	18104_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	233
HRA_PMA2	CPUx peripheral space identifier register	18108_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	234
HRA_DCON2	CPUx data control register 2	19000_H	P0, 32	P0, SV, 32	Kernel Reset	257
HRA_DCON1	CPUx data control register 1	19008_H	P0, 32	P0, SV, 32	Kernel Reset	258
HRA_SMACON	CPUx SIST mode access control register	$1900C_H$	P0, 32	SFRSE, P0, SV, 32	Kernel Reset	234
HRA_DSTR	CPUx HR specific data synchronous trap register	19010_H	P0, 32	P0, SV, 32	Kernel Reset	259
HRA_DATR	CPUx HR specific data asynchronous trap register	19018_H	P0, 32	P0, SV, 32	Kernel Reset	261
HRA_DEADD	CPUx HR specific data error address register	$1901C_H$	P0, 32	P0, SV, 32	Kernel Reset	262
HRA_DIEAR	CPUx data integrity error address register	19020_H	P0, 32	P0, SV, 32	Kernel Reset	279

(table continues...)

Table 101 (continued) Register overview - CPU domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
HRA_DIETR	CPUx data integrity error trap register	19024_H	P0, 32	P0, SV, 32	Kernel Reset	280
HRA_DCON0	CPUx data memory control register	19040_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	258
HRA_PSTR	CPUx HR specific program synchronous trap register	19200_H	P0, 32	P0, SV, 32	Kernel Reset	295
HRA_PCON1	CPUx program control 1	19204_H	P0, 32	P0, SV, 32	Kernel Reset	292
HRA_PCON2	CPUx program control 2	19208_H	P0, 32	P0, SV, 32	Kernel Reset	293
HRA_PCON0	CPUx program control 0	$1920C_H$	P0, 32	SFRE, P0, SV, 32	Kernel Reset	294
HRA_PIEAR	CPUx HR specific program integrity error address register	19210_H	P0, 32	P0, SV, 32	Kernel Reset	281
HRA_PIETR	CPUx HR specific program integrity error trap register	19214_H	P0, 32	P0, SV, 32	Kernel Reset	282
HRA_COMPAT	CPUx compatibility control register	19400_H	P0, 32	SFRSE, P0, SV, 32	Kernel Reset	235
HRA_FPU_TRAP_CON	CPUx HR specific trap control register	$1A000_H$	P0, 32	P0, SV, 32	Kernel Reset	263
HRA_FPU_TRAP_PC	CPUx HR specific trapping instruction program counter register	$1A004_H$	P0, 32	P0, SV, 32	Kernel Reset	265
HRA_FPU_TRAP_OPC	CPUx HR specific trapping instruction opcode register	$1A008_H$	P0, 32	P0, SV, 32	Kernel Reset	266
HRA_FPU_TRAP_SRC1_L	CPUx HR specific trapping instruction operand register	$1A010_H$	P0, 32	P0, SV, 32	Kernel Reset	267
HRA_FPU_TRAP_SRC1_U	CPUx HR specific trapping instruction operand register	$1A014_H$	P0, 32	P0, SV, 32	Kernel Reset	268
HRA_FPU_TRAP_SRC2_L	CPUx HR specific trapping instruction operand register	$1A018_H$	P0, 32	P0, SV, 32	Kernel Reset	268
HRA_FPU_TRAP_SRC2_U	CPUx HR specific trapping instruction operand register	$1A01C_H$	P0, 32	P0, SV, 32	Kernel Reset	269
HRA_FPU_TRAP_SRC3_L	CPUx HR specific trapping instruction operand register	$1A020_H$	P0, 32	P0, SV, 32	Kernel Reset	270
HRA_FPU_TRAP_SRC3_U	CPUx HR specific trapping instruction operand register	$1A024_H$	P0, 32	P0, SV, 32	Kernel Reset	270

(table continues...)

Table 101 (continued) Register overview - CPU domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
HRA_FPU_SYNC_TRAP_CON	CPUx HR specific synchronous trap control register	$1A030_H$	P0, 32	P0, SV, 32	Kernel Reset	271
HRA_FPU_SYNC_TRAP_OPC	CPUx HR specific trapping instruction opcode register	$1A034_H$	P0, 32	P0, SV, 32	Kernel Reset	273
HRA_DPRy_L	CPUx HR specific data protection range y, lower bound register	$1C000_H$+ y*8	P0, 32	P0, SV, 32	Kernel Reset	283
HRA_DPRy_U	CPUx HR specific data protection range y, upper bound register	$1C004_H$+ y*8	P0, 32	P0, SV, 32	Kernel Reset	284
HRA_CPRy_L	CPUx HR specific code protection range y, lower bound register	$1D000_H$+ y*8	P0, 32	P0, SV, 32	Kernel Reset	285
HRA_CPRy_U	CPUx HR specific code protection range y, upper bound register	$1D004_H$+ y*8	P0, 32	P0, SV, 32	Kernel Reset	286
HRA_CPXE_y	CPUx HR specific code protection execute enable register set y	$1E000_H$+ y*4	P0, 32	P0, SV, 32	Kernel Reset	287
HRA_DPRE_y	CPUx HR specific data protection read enable register set y	$1E010_H$+ y*4	P0, 32	P0, SV, 32	Kernel Reset	288
HRA_DPWE_y	CPUx HR specific data protection write enable register set y	$1E020_H$+ y*4	P0, 32	P0, SV, 32	Kernel Reset	289
HRA_CPXE_y	CPUx HR specific code protection execute enable register set y	$1E040_H$+ (y-4)*4	P0, 32	P0, SV, 32	Kernel Reset	290
HRA_DPRE_y	CPUx HR specific data protection read enable register set y	$1E050_H$+ (y-4)*4	P0, 32	P0, SV, 32	Kernel Reset	291
HRA_DPWE_y	CPUx HR specific data protection write enable register set y	$1E060_H$+ (y-4)*4	P0, 32	P0, SV, 32	Kernel Reset	292
HRA_TPS_CON	CPUx HR specific temporal protection system control register	$1E400_H$	P0, 32	P0, SV, 32	Kernel Reset	296
HRA_TPS_TIMERn	CPUx HR specific temporal protection system timer register n	$1E404_H$+ n*4	P0, 32	P0, SV, 32	Kernel Reset	297
HRA_TRnEVT	CPUx trigger event n	$1F000_H$+ n*8	P0, 32	P0, SV, 32	Debug Reset	209

(table continues...)

Table 101 (continued) Register overview - CPU domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
HRA_TRnADR	CPUx trigger address n	$1F004_H$+ n*8	P0, 32	P0, SV, 32	Debug Reset	210
HRA_CCTRL	CPUx HR specific counter control	$1FC00_H$	P0, 32	P0, SV, 32	Kernel Reset	226
HRA_CCNT	CPUx HR specific CPU clock cycle count	$1FC04_H$	P0, 32	P0, SV, 32	Kernel Reset	227
HRA_ICNT	CPUx HR specific instruction count	$1FC08_H$	P0, 32	P0, SV, 32	Kernel Reset	228
HRA_M1CNT	CPUx HR specific multi-count register 1	$1FC0C_H$	P0, 32	P0, SV, 32	Kernel Reset	229
HRA_M2CNT	CPUx HR specific multi-count register 2	$1FC10_H$	P0, 32	P0, SV, 32	Kernel Reset	230
HRA_M3CNT	CPUx HR specific multi-count register 3	$1FC14_H$	P0, 32	P0, SV, 32	Kernel Reset	231
HRA_DBGSR	CPUx debug status register	$1FD00_H$	P0, 32	P0, SV, 32	Debug Reset	211
HRA_EXEVT	CPUx external event register	$1FD08_H$	P0, 32	P0, SV, 32	Debug Reset	212
HRA_CREVT	CPUx core register access event	$1FD0C_H$	P0, 32	P0, SV, 32	Debug Reset	213
HRA_SWEVT	CPUx software debug event	$1FD10_H$	P0, 32	P0, SV, 32	Debug Reset	214
HRA_DBGACT	CPUx debug action configuration	$1FD14_H$	P0, 32	P0, SV, 32	Debug Reset	215
HRA_TRIG_ACC	CPUx trigger event accumulator	$1FD30_H$	P0, 32	P0, SV, 32	Debug Reset	217
HRA_DMS	CPUx debug monitor start address	$1FD40_H$	P0, 32	P0, SV, 32	Debug Reset	218
HRA_DCX	CPUx debug context save area pointer	$1FD44_H$	P0, 32	P0, SV, 32	Debug Reset	218
HRA_DBGTCR	CPUx debug trap control	$1FD48_H$	P0, 32	P0, SV, 32	Application Reset	219
HRA_DBGCFG	CPUx debug configuration	$1FD4C_H$	P0, 32	P0, SV, 32	Debug Reset	220
HRA_TRCCFG	CPUx trace configuration	$1FD50_H$	P0, 32	P0, SV, 32	Debug Reset	221
HRA_TRCFILT	CPUx trace filter	$1FD54_H$	P0, 32	P0, SV, 32	Debug Reset	222
HRA_TRCLIM	CPUx trace bandwidth limiter	$1FD58_H$	P0, 32	P0, SV, 32	Debug Reset	224
HRA_TS16PTCCTRL	CPUx MCDS performance counter control register	$1FD60_H$	P0, 32	P0, SV, 32	Debug Reset	225
HRA_PCXI	CPUx previous context information register	$1FE00_H$	P0, 32	P0, SV, 32	Kernel Reset	236
HRA_PSW	CPUx HR specific program status word	$1FE04_H$	P0, 32	P0, SV, 32	Kernel Reset	237

(table continues...)

Table 101 (continued) Register overview - CPU domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
HRA_PC	CPUx program counter	$1FE08_H$	P0, 32	P0, SV, 32	Kernel Reset	240
HRA_CORECON	CPUx HR specific core configuration register	$1FE14_H$	P0, 32	SFRSE, P0, SV, 32	Kernel Reset	250
HRA_CPU_ID	CPUx identification register	$1FE18_H$	P0, 32	P0, SV, 32	Kernel Reset	241
HRA_CORE_ID	CPUx core identification register	$1FE1C_H$	P0, 32	P0, SV, 32	Kernel Reset	242
HRA_BIV	CPUx HR specific base interrupt vector table pointer	$1FE20_H$	P0, 32	SFRE, P0, SV, 32	Kernel Reset	251
HRA_BTV	CPUx HR specific base trap vector table pointer	$1FE24_H$	P0, 32	SFRE, P0, SV, 32	Kernel Reset	252
HRA_ISP	CPUx HR specific interrupt stack pointer	$1FE28_H$	P0, 32	SFRE, P0, SV, 32	Kernel Reset	253
HRA_ICR	CPUx interrupt control register	$1FE2C_H$	P0, 32	P0, SV, 32	Kernel Reset	242
HRA_PPRS	CPUx previous PRS	$1FE34_H$	P0, 32	P0, SV, 32	Kernel Reset	244
HRA_FCX	CPUx HR specific free CSA list head pointer	$1FE38_H$	P0, 32	P0, SV, 32	Kernel Reset	254
HRA_LCX	CPUx HR specific free CSA list limit pointer	$1FE3C_H$	P0, 32	P0, SV, 32	Kernel Reset	255
HRA_SWID	CPUx software identification	$1FE40_H$	P0, 32	P0, SV, 32	Kernel Reset	244
HRA_CUS_ID	CPUx customer ID register	$1FE50_H$	P0, 32	P0, SV, 32	Kernel Reset	245
HRA_BOOTCON	CPUx core boot configuration register	$1FE60_H$	P0, 32	P0, SV, 32	Kernel Reset	246
HRA_LCLCON	CPUx lockstep control register	$1FE64_H$	P0, 32	STP, SFRSE, P0, SV, 32	Cold PowerOn Reset	247
HRA_CCON	CPUx clock control register	$1FE68_H$	P0, 32	P0, SV, 32	Kernel Reset	248
HRA_TCCON	CPUx hardware configuration	$1FE6C_H$	P0, 32	P0, SV, 32	Kernel Reset	248
Dy	CPUx data general purpose register y	$1FF00_H$+ y*4	P0, 32	P0, SV, 32	Kernel Reset	274
Ay	CPUx address general purpose register y	$1FF80_H$+ y*4	P0, 32	P0, SV, 32	Kernel Reset	274
HRB_SEGEN	CPUx SRI error generation register	21030_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	255
HRB_LCLTEST	CPUx lockstep test register	21040_H	P0, 32	P0, SV, 32	Cold PowerOn Reset	256
HRB_PMA0	CPUx data access cacheability register	28100_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	232

(table continues...)

Table 101 (continued) Register overview - CPU domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
HRB_PMA1	CPUx code access cacheability register	28104_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	233
HRB_PMA2	CPUx peripheral space identifier register	28108_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	234
HRB_DCON2	CPUx data control register 2	29000_H	P0, 32	P0, SV, 32	Kernel Reset	257
HRB_DCON1	CPUx data control register 1	29008_H	P0, 32	P0, SV, 32	Kernel Reset	258
HRB_SMACON	CPUx SIST mode access control register	$2900C_H$	P0, 32	SFRSE, P0, SV, 32	Kernel Reset	234
HRB_DSTR	CPUx HR specific data synchronous trap register	29010_H	P0, 32	P0, SV, 32	Kernel Reset	259
HRB_DATR	CPUx HR specific data asynchronous trap register	29018_H	P0, 32	P0, SV, 32	Kernel Reset	261
HRB_DEADD	CPUx HR specific data error address register	$2901C_H$	P0, 32	P0, SV, 32	Kernel Reset	262
HRB_DIEAR	CPUx data integrity error address register	29020_H	P0, 32	P0, SV, 32	Kernel Reset	279
HRB_DIETR	CPUx data integrity error trap register	29024_H	P0, 32	P0, SV, 32	Kernel Reset	280
HRB_DCON0	CPUx data memory control register	29040_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	258
HRB_PSTR	CPUx HR specific program synchronous trap register	29200_H	P0, 32	P0, SV, 32	Kernel Reset	295
HRB_PCON1	CPUx program control 1	29204_H	P0, 32	P0, SV, 32	Kernel Reset	292
HRB_PCON2	CPUx program control 2	29208_H	P0, 32	P0, SV, 32	Kernel Reset	293
HRB_PCON0	CPUx program control 0	$2920C_H$	P0, 32	SFRE, P0, SV, 32	Kernel Reset	294
HRB_PIEAR	CPUx HR specific program integrity error address register	29210_H	P0, 32	P0, SV, 32	Kernel Reset	281
HRB_PIETR	CPUx HR specific program integrity error trap register	29214_H	P0, 32	P0, SV, 32	Kernel Reset	282
HRB_COMPAT	CPUx compatibility control register	29400_H	P0, 32	SFRSE, P0, SV, 32	Kernel Reset	235
HRB_FPU_TRAP_CON	CPUx HR specific trap control register	$2A000_H$	P0, 32	P0, SV, 32	Kernel Reset	263
HRB_FPU_TRAP_PC	CPUx HR specific trapping instruction program counter register	$2A004_H$	P0, 32	P0, SV, 32	Kernel Reset	265
HRB_FPU_TRAP_OPC	CPUx HR specific trapping instruction opcode register	$2A008_H$	P0, 32	P0, SV, 32	Kernel Reset	266

(table continues...)

Table 101 (continued) Register overview - CPU domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
HRB_FPU_TRAP_SRC1_L	CPUx HR specific trapping instruction operand register	$2A010_H$	P0, 32	P0, SV, 32	Kernel Reset	267
HRB_FPU_TRAP_SRC1_U	CPUx HR specific trapping instruction operand register	$2A014_H$	P0, 32	P0, SV, 32	Kernel Reset	268
HRB_FPU_TRAP_SRC2_L	CPUx HR specific trapping instruction operand register	$2A018_H$	P0, 32	P0, SV, 32	Kernel Reset	268
HRB_FPU_TRAP_SRC2_U	CPUx HR specific trapping instruction operand register	$2A01C_H$	P0, 32	P0, SV, 32	Kernel Reset	269
HRB_FPU_TRAP_SRC3_L	CPUx HR specific trapping instruction operand register	$2A020_H$	P0, 32	P0, SV, 32	Kernel Reset	270
HRB_FPU_TRAP_SRC3_U	CPUx HR specific trapping instruction operand register	$2A024_H$	P0, 32	P0, SV, 32	Kernel Reset	270
HRB_FPU_SYNC_TRAP_CON	CPUx HR specific synchronous trap control register	$2A030_H$	P0, 32	P0, SV, 32	Kernel Reset	271
HRB_FPU_SYNC_TRAP_OPC	CPUx HR specific trapping instruction opcode register	$2A034_H$	P0, 32	P0, SV, 32	Kernel Reset	273
HRB_DPRy_L	CPUx HR specific data protection range y, lower bound register	$2C000_H$+ y*8	P0, 32	P0, SV, 32	Kernel Reset	283
HRB_DPRy_U	CPUx HR specific data protection range y, upper bound register	$2C004_H$+ y*8	P0, 32	P0, SV, 32	Kernel Reset	284
HRB_CPRy_L	CPUx HR specific code protection range y, lower bound register	$2D000_H$+ y*8	P0, 32	P0, SV, 32	Kernel Reset	285
HRB_CPRy_U	CPUx HR specific code protection range y, upper bound register	$2D004_H$+ y*8	P0, 32	P0, SV, 32	Kernel Reset	286
HRB_CPXE_y	CPUx HR specific code protection execute enable register set y	$2E000_H$+ y*4	P0, 32	P0, SV, 32	Kernel Reset	287
HRB_DPRE_y	CPUx HR specific data protection read enable register set y	$2E010_H$+ y*4	P0, 32	P0, SV, 32	Kernel Reset	288
HRB_DPWE_y	CPUx HR specific data protection write enable register set y	$2E020_H$+ y*4	P0, 32	P0, SV, 32	Kernel Reset	289

(table continues...)

Table 101 (continued) Register overview - CPU domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
HRB_CPXE_y	CPUx HR specific code protection execute enable register set y	$2E040_H$+(y-4)*4	P0, 32	P0, SV, 32	Kernel Reset	290
HRB_DPRE_y	CPUx HR specific data protection read enable register set y	$2E050_H$+(y-4)*4	P0, 32	P0, SV, 32	Kernel Reset	291
HRB_DPWE_y	CPUx HR specific data protection write enable register set y	$2E060_H$+(y-4)*4	P0, 32	P0, SV, 32	Kernel Reset	292
HRB_TPS_CON	CPUx HR specific temporal protection system control register	$2E400_H$	P0, 32	P0, SV, 32	Kernel Reset	296
HRB_TPS_TIMERn	CPUx HR specific temporal protection system timer register n	$2E404_H$+n*4	P0, 32	P0, SV, 32	Kernel Reset	297
HRB_TRnEVT	CPUx trigger event n	$2F000_H$+n*8	P0, 32	P0, SV, 32	Debug Reset	209
HRB_TRnADR	CPUx trigger address n	$2F004_H$+n*8	P0, 32	P0, SV, 32	Debug Reset	210
HRB_CCTRL	CPUx HR specific counter control	$2FC00_H$	P0, 32	P0, SV, 32	Kernel Reset	226
HRB_CCNT	CPUx HR specific CPU clock cycle count	$2FC04_H$	P0, 32	P0, SV, 32	Kernel Reset	227
HRB_ICNT	CPUx HR specific instruction count	$2FC08_H$	P0, 32	P0, SV, 32	Kernel Reset	228
HRB_M1CNT	CPUx HR specific multi-count register 1	$2FC0C_H$	P0, 32	P0, SV, 32	Kernel Reset	229
HRB_M2CNT	CPUx HR specific multi-count register 2	$2FC10_H$	P0, 32	P0, SV, 32	Kernel Reset	230
HRB_M3CNT	CPUx HR specific multi-count register 3	$2FC14_H$	P0, 32	P0, SV, 32	Kernel Reset	231
HRB_DBGSR	CPUx debug status register	$2FD00_H$	P0, 32	P0, SV, 32	Debug Reset	211
HRB_EXEVT	CPUx external event register	$2FD08_H$	P0, 32	P0, SV, 32	Debug Reset	212
HRB_CREVT	CPUx core register access event	$2FD0C_H$	P0, 32	P0, SV, 32	Debug Reset	213
HRB_SWEVT	CPUx software debug event	$2FD10_H$	P0, 32	P0, SV, 32	Debug Reset	214
HRB_DBGACT	CPUx debug action configuration	$2FD14_H$	P0, 32	P0, SV, 32	Debug Reset	215
HRB_TRIG_ACC	CPUx trigger event accumulator	$2FD30_H$	P0, 32	P0, SV, 32	Debug Reset	217

(table continues...)

Table 101 (continued) Register overview - CPU domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
HRB_DMS	CPUx debug monitor start address	$2FD40_H$	P0, 32	P0, SV, 32	Debug Reset	218
HRB_DCX	CPUx debug context save area pointer	$2FD44_H$	P0, 32	P0, SV, 32	Debug Reset	218
HRB_DBGTCR	CPUx debug trap control	$2FD48_H$	P0, 32	P0, SV, 32	Application Reset	219
HRB_DBGCFG	CPUx debug configuration	$2FD4C_H$	P0, 32	P0, SV, 32	Debug Reset	220
HRB_TRCCFG	CPUx trace configuration	$2FD50_H$	P0, 32	P0, SV, 32	Debug Reset	221
HRB_TRCFILT	CPUx trace filter	$2FD54_H$	P0, 32	P0, SV, 32	Debug Reset	222
HRB_TRCLIM	CPUx trace bandwidth limiter	$2FD58_H$	P0, 32	P0, SV, 32	Debug Reset	224
HRB_TS16PTCCTRL	CPUx MCDS performance counter control register	$2FD60_H$	P0, 32	P0, SV, 32	Debug Reset	225
HRB_PCXI	CPUx previous context information register	$2FE00_H$	P0, 32	P0, SV, 32	Kernel Reset	236
HRB_PSW	CPUx HR specific program status word	$2FE04_H$	P0, 32	P0, SV, 32	Kernel Reset	237
HRB_PC	CPUx program counter	$2FE08_H$	P0, 32	P0, SV, 32	Kernel Reset	240
HRB_CORECON	CPUx HR specific core configuration register	$2FE14_H$	P0, 32	SFRSE, P0, SV, 32	Kernel Reset	250
HRB_CPU_ID	CPUx identification register	$2FE18_H$	P0, 32	P0, SV, 32	Kernel Reset	241
HRB_CORE_ID	CPUx core identification register	$2FE1C_H$	P0, 32	P0, SV, 32	Kernel Reset	242
HRB_BIV	CPUx HR specific base interrupt vector table pointer	$2FE20_H$	P0, 32	SFRE, P0, SV, 32	Kernel Reset	251
HRB_BTV	CPUx HR specific base trap vector table pointer	$2FE24_H$	P0, 32	SFRE, P0, SV, 32	Kernel Reset	252
HRB_ISP	CPUx HR specific interrupt stack pointer	$2FE28_H$	P0, 32	SFRE, P0, SV, 32	Kernel Reset	253
HRB_ICR	CPUx interrupt control register	$2FE2C_H$	P0, 32	P0, SV, 32	Kernel Reset	242
HRB_PPRS	CPUx previous PRS	$2FE34_H$	P0, 32	P0, SV, 32	Kernel Reset	244
HRB_FCX	CPUx HR specific free CSA list head pointer	$2FE38_H$	P0, 32	P0, SV, 32	Kernel Reset	254
HRB_LCX	CPUx HR specific free CSA list limit pointer	$2FE3C_H$	P0, 32	P0, SV, 32	Kernel Reset	255
HRB_SWID	CPUx software identification	$2FE40_H$	P0, 32	P0, SV, 32	Kernel Reset	244
HRB_CUS_ID	CPUx customer ID register	$2FE50_H$	P0, 32	P0, SV, 32	Kernel Reset	245

(table continues...)

Table 101 (continued) Register overview - CPU domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
HRB_BOOTCON	CPUx core boot configuration register	$2FE60_H$	P0, 32	P0, SV, 32	Kernel Reset	246
HRB_LCLCON	CPUx lockstep control register	$2FE64_H$	P0, 32	STP, SFRSE, P0, SV, 32	Cold PowerOn Reset	247
HRB_CCON	CPUx clock control register	$2FE68_H$	P0, 32	P0, SV, 32	Kernel Reset	248
HRB_TCCON	CPUx hardware configuration	$2FE6C_H$	P0, 32	P0, SV, 32	Kernel Reset	248
HRB_Ay	CPUx HR specific address general purpose register y	$2FF80_H$+ y*4	P0, 32	P0, SV, 32	Kernel Reset	274
HRB_Ay	CPUx HR specific address general purpose register y	$2FFA0_H$+ (y-8)*4	P0, 32	P0, SV, 32	Kernel Reset	274
HRHV_SEGEN	CPUx SRI error generation register	31030_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	255
HRHV_LCLTEST	CPUx lockstep test register	31040_H	P0, 32	P0, SV, 32	Cold PowerOn Reset	256
HRHV_PMA0	CPUx data access cacheability register	38100_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	232
HRHV_PMA1	CPUx code access cacheability register	38104_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	233
HRHV_PMA2	CPUx peripheral space identifier register	38108_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	234
HRHV_DCON2	CPUx data control register 2	39000_H	P0, 32	P0, SV, 32	Kernel Reset	257
HRHV_DCON1	CPUx data control register 1	39008_H	P0, 32	P0, SV, 32	Kernel Reset	258
HRHV_SMACON	CPUx SIST mode access control register	$3900C_H$	P0, 32	SFRSE, P0, SV, 32	Kernel Reset	234
HRHV_DSTR	CPUx HR specific data synchronous trap register	39010_H	P0, 32	P0, SV, 32	Kernel Reset	259
HRHV_DATR	CPUx HR specific data asynchronous trap register	39018_H	P0, 32	P0, SV, 32	Kernel Reset	261
HRHV_DEADD	CPUx HR specific data error address register	$3901C_H$	P0, 32	P0, SV, 32	Kernel Reset	262
HRHV_DIEAR	CPUx data integrity error address register	39020_H	P0, 32	P0, SV, 32	Kernel Reset	279
HRHV_DIETR	CPUx data integrity error trap register	39024_H	P0, 32	P0, SV, 32	Kernel Reset	280
HRHV_DCON0	CPUx data memory control register	39040_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	258
HRHV_PSTR	CPUx HR specific program synchronous trap register	39200_H	P0, 32	P0, SV, 32	Kernel Reset	295
HRHV_PCON1	CPUx program control 1	39204_H	P0, 32	P0, SV, 32	Kernel Reset	292
HRHV_PCON2	CPUx program control 2	39208_H	P0, 32	P0, SV, 32	Kernel Reset	293

(table continues...)

Table 101 (continued) Register overview - CPU domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
HRHV_PCON0	CPUx program control 0	$3920C_H$	P0, 32	SFRE, P0, SV, 32	Kernel Reset	294
HRHV_PIEAR	CPUx HR specific program integrity error address register	39210_H	P0, 32	P0, SV, 32	Kernel Reset	281
HRHV_PIETR	CPUx HR specific program integrity error trap register	39214_H	P0, 32	P0, SV, 32	Kernel Reset	282
HRHV_COMPAT	CPUx compatibility control register	39400_H	P0, 32	SFRSE, P0, SV, 32	Kernel Reset	235
HRHV_FPU_TRAP_CON	CPUx HR specific trap control register	$3A000_H$	P0, 32	P0, SV, 32	Kernel Reset	263
HRHV_FPU_TRAP_PC	CPUx HR specific trapping instruction program counter register	$3A004_H$	P0, 32	P0, SV, 32	Kernel Reset	265
HRHV_FPU_TRAP_OPC	CPUx HR specific trapping instruction opcode register	$3A008_H$	P0, 32	P0, SV, 32	Kernel Reset	266
HRHV_FPU_TRAP_SRC1_L	CPUx HR specific trapping instruction operand register	$3A010_H$	P0, 32	P0, SV, 32	Kernel Reset	267
HRHV_FPU_TRAP_SRC1_U	CPUx HR specific trapping instruction operand register	$3A014_H$	P0, 32	P0, SV, 32	Kernel Reset	268
HRHV_FPU_TRAP_SRC2_L	CPUx HR specific trapping instruction operand register	$3A018_H$	P0, 32	P0, SV, 32	Kernel Reset	268
HRHV_FPU_TRAP_SRC2_U	CPUx HR specific trapping instruction operand register	$3A01C_H$	P0, 32	P0, SV, 32	Kernel Reset	269
HRHV_FPU_TRAP_SRC3_L	CPUx HR specific trapping instruction operand register	$3A020_H$	P0, 32	P0, SV, 32	Kernel Reset	270
HRHV_FPU_TRAP_SRC3_U	CPUx HR specific trapping instruction operand register	$3A024_H$	P0, 32	P0, SV, 32	Kernel Reset	270
HRHV_FPU_SYNC_TRAP_CON	CPUx HR specific synchronous trap control register	$3A030_H$	P0, 32	P0, SV, 32	Kernel Reset	271
HRHV_FPU_SYNC_TRAP_OPC	CPUx HR specific trapping instruction opcode register	$3A034_H$	P0, 32	P0, SV, 32	Kernel Reset	273
HRHV_VCON0	CPUx virtualization control register 0	$3B000_H$	P0, 32	P0, SV, 32	Kernel Reset	275
HRHV_VCON1	CPUx virtualization control register 1	$3B004_H$	P0, 32	P0, SV, 32	Kernel Reset	276

(table continues...)

Table 101 (continued) Register overview - CPU domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
HRHV_VCON2	CPUx virtualization control register 2	$3B008_H$	P0, 32	P0, SV, 32	Kernel Reset	276
HRHV_BHV	CPUx base hypervisor vector table pointer	$3B010_H$	P0, 32	P0, SV, 32, SFRE	Kernel Reset	277
HRHV_VMn_ICR	CPUx Virtual machine n interrupt state	$3B100_H$+ n*4	P0, 32	P0, SV, 32	Kernel Reset	278
HRHV_VMn_PETHRESH	CPUx Virtual machine n pre-emption threshold	$3B200_H$+ n*4	P0, 32	P0, SV, 32	Kernel Reset	279
HRHV_DPRy_L	CPUx HR specific data protection range y, lower bound register	$3C000_H$+ y*8	P0, 32	P0, SV, 32	Kernel Reset	283
HRHV_DPRy_U	CPUx HR specific data protection range y, upper bound register	$3C004_H$+ y*8	P0, 32	P0, SV, 32	Kernel Reset	284
HRHV_CPRy_L	CPUx HR specific code protection range y, lower bound register	$3D000_H$+ y*8	P0, 32	P0, SV, 32	Kernel Reset	285
HRHV_CPRy_U	CPUx HR specific code protection range y, upper bound register	$3D004_H$+ y*8	P0, 32	P0, SV, 32	Kernel Reset	286
HRHV_CPXE_y	CPUx HR specific code protection execute enable register set y	$3E000_H$+ y*4	P0, 32	P0, SV, 32	Kernel Reset	287
HRHV_DPRE_y	CPUx HR specific data protection read enable register set y	$3E010_H$+ y*4	P0, 32	P0, SV, 32	Kernel Reset	288
HRHV_DPWE_y	CPUx HR specific data protection write enable register set y	$3E020_H$+ y*4	P0, 32	P0, SV, 32	Kernel Reset	289
HRHV_CPXE_y	CPUx HR specific code protection execute enable register set y	$3E040_H$+ (y-4)*4	P0, 32	P0, SV, 32	Kernel Reset	290
HRHV_DPRE_y	CPUx HR specific data protection read enable register set y	$3E050_H$+ (y-4)*4	P0, 32	P0, SV, 32	Kernel Reset	291
HRHV_DPWE_y	CPUx HR specific data protection write enable register set y	$3E060_H$+ (y-4)*4	P0, 32	P0, SV, 32	Kernel Reset	292
HRHV_TPS_CON	CPUx HR specific temporal protection system control register	$3E400_H$	P0, 32	P0, SV, 32	Kernel Reset	296
HRHV_TPS_TIMERn	CPUx HR specific temporal protection system timer register n	$3E404_H$+ n*4	P0, 32	P0, SV, 32	Kernel Reset	297

(table continues...)

Table 101 (continued) Register overview - CPU domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
HRHV_TRnEVT	CPUx trigger event n	$3F000_H$+ n*8	P0, 32	P0, SV, 32	Debug Reset	209
HRHV_TRnADR	CPUx trigger address n	$3F004_H$+ n*8	P0, 32	P0, SV, 32	Debug Reset	210
HRHV_CCTRL	CPUx HR specific counter control	$3FC00_H$	P0, 32	P0, SV, 32	Kernel Reset	226
HRHV_CCNT	CPUx HR specific CPU clock cycle count	$3FC04_H$	P0, 32	P0, SV, 32	Kernel Reset	227
HRHV_ICNT	CPUx HR specific instruction count	$3FC08_H$	P0, 32	P0, SV, 32	Kernel Reset	228
HRHV_M1CNT	CPUx HR specific multi-count register 1	$3FC0C_H$	P0, 32	P0, SV, 32	Kernel Reset	229
HRHV_M2CNT	CPUx HR specific multi-count register 2	$3FC10_H$	P0, 32	P0, SV, 32	Kernel Reset	230
HRHV_M3CNT	CPUx HR specific multi-count register 3	$3FC14_H$	P0, 32	P0, SV, 32	Kernel Reset	231
HRHV_DBGSR	CPUx debug status register	$3FD00_H$	P0, 32	P0, SV, 32	Debug Reset	211
HRHV_EXEVT	CPUx external event register	$3FD08_H$	P0, 32	P0, SV, 32	Debug Reset	212
HRHV_CREVT	CPUx core register access event	$3FD0C_H$	P0, 32	P0, SV, 32	Debug Reset	213
HRHV_SWEVT	CPUx software debug event	$3FD10_H$	P0, 32	P0, SV, 32	Debug Reset	214
HRHV_DBGACT	CPUx debug action configuration	$3FD14_H$	P0, 32	P0, SV, 32	Debug Reset	215
HRHV_TRIG_ACC	CPUx trigger event accumulator	$3FD30_H$	P0, 32	P0, SV, 32	Debug Reset	217
HRHV_DMS	CPUx debug monitor start address	$3FD40_H$	P0, 32	P0, SV, 32	Debug Reset	218
HRHV_DCX	CPUx debug context save area pointer	$3FD44_H$	P0, 32	P0, SV, 32	Debug Reset	218
HRHV_DBGTCR	CPUx debug trap control	$3FD48_H$	P0, 32	P0, SV, 32	Application Reset	219
HRHV_DBGCFG	CPUx debug configuration	$3FD4C_H$	P0, 32	P0, SV, 32	Debug Reset	220
HRHV_TRCCFG	CPUx trace configuration	$3FD50_H$	P0, 32	P0, SV, 32	Debug Reset	221
HRHV_TRCFILT	CPUx trace filter	$3FD54_H$	P0, 32	P0, SV, 32	Debug Reset	222
HRHV_TRCLIM	CPUx trace bandwidth limiter	$3FD58_H$	P0, 32	P0, SV, 32	Debug Reset	224
HRHV_TS16PTCCTRL	CPUx MCDS performance counter control register	$3FD60_H$	P0, 32	P0, SV, 32	Debug Reset	225
HRHV_PCXI	CPUx previous context information register	$3FE00_H$	P0, 32	P0, SV, 32	Kernel Reset	236

(table continues...)

Table 101 (continued) Register overview - CPU domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
HRHV_PSW	CPUx HR specific program status word	$3FE04_H$	P0, 32	P0, SV, 32	Kernel Reset	237
HRHV_PC	CPUx program counter	$3FE08_H$	P0, 32	P0, SV, 32	Kernel Reset	240
HRHV_CORECON	CPUx HR specific core configuration register	$3FE14_H$	P0, 32	SFRSE, P0, SV, 32	Kernel Reset	250
HRHV_CPU_ID	CPUx identification register	$3FE18_H$	P0, 32	P0, SV, 32	Kernel Reset	241
HRHV_CORE_ID	CPUx core identification register	$3FE1C_H$	P0, 32	P0, SV, 32	Kernel Reset	242
HRHV_BIV	CPUx HR specific base interrupt vector table pointer	$3FE20_H$	P0, 32	SFRE, P0, SV, 32	Kernel Reset	251
HRHV_BTV	CPUx HR specific base trap vector table pointer	$3FE24_H$	P0, 32	SFRE, P0, SV, 32	Kernel Reset	252
HRHV_ISP	CPUx HR specific interrupt stack pointer	$3FE28_H$	P0, 32	SFRE, P0, SV, 32	Kernel Reset	253
HRHV_ICR	CPUx interrupt control register	$3FE2C_H$	P0, 32	P0, SV, 32	Kernel Reset	242
HRHV_PPRS	CPUx previous PRS	$3FE34_H$	P0, 32	P0, SV, 32	Kernel Reset	244
HRHV_FCX	CPUx HR specific free CSA list head pointer	$3FE38_H$	P0, 32	P0, SV, 32	Kernel Reset	254
HRHV_LCX	CPUx HR specific free CSA list limit pointer	$3FE3C_H$	P0, 32	P0, SV, 32	Kernel Reset	255
HRHV_SWID	CPUx software identification	$3FE40_H$	P0, 32	P0, SV, 32	Kernel Reset	244
HRHV_CUS_ID	CPUx customer ID register	$3FE50_H$	P0, 32	P0, SV, 32	Kernel Reset	245
HRHV_BOOTCON	CPUx core boot configuration register	$3FE60_H$	P0, 32	P0, SV, 32	Kernel Reset	246
HRHV_LCLCON	CPUx lockstep control register	$3FE64_H$	P0, 32	STP, SFRSE, P0, SV, 32	Cold PowerOn Reset	247
HRHV_CCON	CPUx clock control register	$3FE68_H$	P0, 32	P0, SV, 32	Kernel Reset	248
HRHV_TCCON	CPUx hardware configuration	$3FE6C_H$	P0, 32	P0, SV, 32	Kernel Reset	248
HRHV_Ay	CPUx HR specific address general purpose register y	$3FF80_H$+ y*4	P0, 32	P0, SV, 32	Kernel Reset	274
HRHV_Ay	CPUx HR specific address general purpose register y	$3FFA0_H$+ (y-8)*4	P0, 32	P0, SV, 32	Kernel Reset	274

4.6.1.3 Register overview - CPU domain CSFR (ascending offset address)

Table 102 Register overview - CPU domain CSFR (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU_SEGEN	CPUx SRI error generation register	01030_H	P0, 32	SFRE, P0, SV, 32	Application Reset	255
CPU_LCLTEST	CPUx lockstep test register	01040_H	P0, 32	P0, SV, 32	Cold PowerOn Reset	256
CPU_PMA0	CPUx data access cacheability register	08100_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	232
CPU_PMA1	CPUx code access cacheability register	08104_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	233
CPU_PMA2	CPUx peripheral space identifier register	08108_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	234
CPU_DCON2	CPUx data control register 2	09000_H	P0, 32	P0, SV, 32	Kernel Reset	257
CPU_DCON1	CPUx data control register 1	09008_H	P0, 32	P0, SV, 32	Kernel Reset	258
CPU_SMACON	CPUx SIST mode access control register	$0900C_H$	P0, 32	SFRSE, P0, SV, 32	Kernel Reset	234
CPU_DSTR	CPUx HR specific data synchronous trap register	09010_H	P0, 32	P0, SV, 32	Kernel Reset	259
CPU_HVHRA_DSTR	CPUx HR specific data synchronous trap register	09011_H	P0, 32	P0, SV, 32	Kernel Reset	259
CPU_HVHRB_DSTR	CPUx HR specific data synchronous trap register	09012_H	P0, 32	P0, SV, 32	Kernel Reset	259
CPU_DATR	CPUx HR specific data asynchronous trap register	09018_H	P0, 32	P0, SV, 32	Kernel Reset	261
CPU_HVHRA_DATR	CPUx HR specific data asynchronous trap register	09019_H	P0, 32	P0, SV, 32	Kernel Reset	261
CPU_HVHRB_DATR	CPUx HR specific data asynchronous trap register	$0901A_H$	P0, 32	P0, SV, 32	Kernel Reset	261
CPU_DEADD	CPUx HR specific data error address register	$0901C_H$	P0, 32	P0, SV, 32	Kernel Reset	262
CPU_HVHRA_DEADD	CPUx HR specific data error address register	$0901D_H$	P0, 32	P0, SV, 32	Kernel Reset	262
CPU_HVHRB_DEADD	CPUx HR specific data error address register	$0901E_H$	P0, 32	P0, SV, 32	Kernel Reset	262
CPU_DIEAR	CPUx data integrity error address register	09020_H	P0, 32	P0, SV, 32	Kernel Reset	279
CPU_HVHRA_DIEAR	CPUx data integrity error address register	09021_H	P0, 32	P0, SV, 32	Kernel Reset	279
CPU_HVHRB_DIEAR	CPUx data integrity error address register	09022_H	P0, 32	P0, SV, 32	Kernel Reset	279
CPU_DIETR	CPUx data integrity error trap register	09024_H	P0, 32	P0, SV, 32	Kernel Reset	280

(table continues...)

Table 102 (continued) Register overview - CPU domain CSFR (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU_HVHRA_DIETR	CPUx data integrity error trap register	09025_H	P0, 32	P0, SV, 32	Kernel Reset	280
CPU_HVHRB_DIETR	CPUx data integrity error trap register	09026_H	P0, 32	P0, SV, 32	Kernel Reset	280
CPU_DCON0	CPUx data memory control register	09040_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	258
CPU_PSTR	CPUx HR specific program synchronous trap register	09200_H	P0, 32	P0, SV, 32	Kernel Reset	295
CPU_HVHRA_PSTR	CPUx HR specific program synchronous trap register	09201_H	P0, 32	P0, SV, 32	Kernel Reset	295
CPU_HVHRB_PSTR	CPUx HR specific program synchronous trap register	09202_H	P0, 32	P0, SV, 32	Kernel Reset	295
CPU_PCON1	CPUx program control 1	09204_H	P0, 32	P0, SV, 32	Kernel Reset	292
CPU_PCON2	CPUx program control 2	09208_H	P0, 32	P0, SV, 32	Kernel Reset	293
CPU_PCON0	CPUx program control 0	$0920C_H$	P0, 32	SFRE, P0, SV, 32	Kernel Reset	294
CPU_PIEAR	CPUx HR specific program integrity error address register	09210_H	P0, 32	P0, SV, 32	Kernel Reset	281
CPU_HVHRA_PIEAR	CPUx HR specific program integrity error address register	09211_H	P0, 32	P0, SV, 32	Kernel Reset	281
CPU_HVHRB_PIEAR	CPUx HR specific program integrity error address register	09212_H	P0, 32	P0, SV, 32	Kernel Reset	281
CPU_PIETR	CPUx HR specific program integrity error trap register	09214_H	P0, 32	P0, SV, 32	Kernel Reset	282
CPU_HVHRA_PIETR	CPUx HR specific program integrity error trap register	09215_H	P0, 32	P0, SV, 32	Kernel Reset	282
CPU_HVHRB_PIETR	CPUx HR specific program integrity error trap register	09216_H	P0, 32	P0, SV, 32	Kernel Reset	282
CPU_COMPAT	CPUx compatibility control register	09400_H	P0, 32	SFRSE, P0, SV, 32	Kernel Reset	235
CPU_FPU_TRAP_CON	CPUx HR specific trap control register	$0A000_H$	P0, 32	P0, SV, 32	Kernel Reset	263
CPU_HVHRA_FPU_TRAP_CON	CPUx HR specific trap control register	$0A001_H$	P0, 32	P0, SV, 32	Kernel Reset	263
CPU_HVHRB_FPU_TRAP_CON	CPUx HR specific trap control register	$0A002_H$	P0, 32	P0, SV, 32	Kernel Reset	263
CPU_FPU_TRAP_PC	CPUx HR specific trapping instruction program counter register	$0A004_H$	P0, 32	P0, SV, 32	Kernel Reset	265

(table continues...)

Table 102 (continued) Register overview - CPU domain CSFR (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU_HVHRA_FPU_TRAP_PC	CPUx HR specific trapping instruction program counter register	$0A005_H$	P0, 32	P0, SV, 32	Kernel Reset	265
CPU_HVHRB_FPU_TRAP_PC	CPUx HR specific trapping instruction program counter register	$0A006_H$	P0, 32	P0, SV, 32	Kernel Reset	265
CPU_FPU_TRAP_OPC	CPUx HR specific trapping instruction opcode register	$0A008_H$	P0, 32	P0, SV, 32	Kernel Reset	266
CPU_HVHRA_FPU_TRAP_OPC	CPUx HR specific trapping instruction opcode register	$0A009_H$	P0, 32	P0, SV, 32	Kernel Reset	266
CPU_HVHRB_FPU_TRAP_OPC	CPUx HR specific trapping instruction opcode register	$0A00A_H$	P0, 32	P0, SV, 32	Kernel Reset	266
CPU_FPU_TRAP_SRC1_L	CPUx HR specific trapping instruction operand register	$0A010_H$	P0, 32	P0, SV, 32	Kernel Reset	267
CPU_HVHRA_FPU_TRAP_SRC1_L	CPUx HR specific trapping instruction operand register	$0A011_H$	P0, 32	P0, SV, 32	Kernel Reset	267
CPU_HVHRB_FPU_TRAP_SRC1_L	CPUx HR specific trapping instruction operand register	$0A012_H$	P0, 32	P0, SV, 32	Kernel Reset	267
CPU_FPU_TRAP_SRC1_U	CPUx HR specific trapping instruction operand register	$0A014_H$	P0, 32	P0, SV, 32	Kernel Reset	268
CPU_HVHRA_FPU_TRAP_SRC1_U	CPUx HR specific trapping instruction operand register	$0A015_H$	P0, 32	P0, SV, 32	Kernel Reset	268
CPU_HVHRB_FPU_TRAP_SRC1_U	CPUx HR specific trapping instruction operand register	$0A016_H$	P0, 32	P0, SV, 32	Kernel Reset	268
CPU_FPU_TRAP_SRC2_L	CPUx HR specific trapping instruction operand register	$0A018_H$	P0, 32	P0, SV, 32	Kernel Reset	268
CPU_HVHRA_FPU_TRAP_SRC2_L	CPUx HR specific trapping instruction operand register	$0A019_H$	P0, 32	P0, SV, 32	Kernel Reset	268
CPU_HVHRB_FPU_TRAP_SRC2_L	CPUx HR specific trapping instruction operand register	$0A01A_H$	P0, 32	P0, SV, 32	Kernel Reset	268
CPU_FPU_TRAP_SRC2_U	CPUx HR specific trapping instruction operand register	$0A01C_H$	P0, 32	P0, SV, 32	Kernel Reset	269

(table continues...)

Table 102 (continued) Register overview - CPU domain CSFR (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU_HVHRA_FPU_TRAP_SRC2_U	CPUx HR specific trapping instruction operand register	$0A01D_H$	P0, 32	P0, SV, 32	Kernel Reset	269
CPU_HVHRB_FPU_TRAP_SRC2_U	CPUx HR specific trapping instruction operand register	$0A01E_H$	P0, 32	P0, SV, 32	Kernel Reset	269
CPU_FPU_TRAP_SRC3_L	CPUx HR specific trapping instruction operand register	$0A020_H$	P0, 32	P0, SV, 32	Kernel Reset	270
CPU_HVHRA_FPU_TRAP_SRC3_L	CPUx HR specific trapping instruction operand register	$0A021_H$	P0, 32	P0, SV, 32	Kernel Reset	270
CPU_HVHRB_FPU_TRAP_SRC3_L	CPUx HR specific trapping instruction operand register	$0A022_H$	P0, 32	P0, SV, 32	Kernel Reset	270
CPU_FPU_TRAP_SRC3_U	CPUx HR specific trapping instruction operand register	$0A024_H$	P0, 32	P0, SV, 32	Kernel Reset	270
CPU_HVHRA_FPU_TRAP_SRC3_U	CPUx HR specific trapping instruction operand register	$0A025_H$	P0, 32	P0, SV, 32	Kernel Reset	270
CPU_HVHRB_FPU_TRAP_SRC3_U	CPUx HR specific trapping instruction operand register	$0A026_H$	P0, 32	P0, SV, 32	Kernel Reset	270
CPU_FPU_SYNC_TRAP_CON	CPUx HR specific synchronous trap control register	$0A030_H$	P0, 32	P0, SV, 32	Kernel Reset	271
CPU_HVHRA_FPU_SYNC_TRAP_CON	CPUx HR specific synchronous trap control register	$0A031_H$	P0, 32	P0, SV, 32	Kernel Reset	271
CPU_HVHRB_FPU_SYNC_TRAP_CON	CPUx HR specific synchronous trap control register	$0A032_H$	P0, 32	P0, SV, 32	Kernel Reset	271
CPU_FPU_SYNC_TRAP_OPC	CPUx HR specific trapping instruction opcode register	$0A034_H$	P0, 32	P0, SV, 32	Kernel Reset	273
CPU_HVHRA_FPU_SYNC_TRAP_OPC	CPUx HR specific trapping instruction opcode register	$0A035_H$	P0, 32	P0, SV, 32	Kernel Reset	273
CPU_HVHRB_FPU_SYNC_TRAP_OPC	CPUx HR specific trapping instruction opcode register	$0A036_H$	P0, 32	P0, SV, 32	Kernel Reset	273
CPU_VCON0	CPUx virtualization control register 0	$0B000_H$	P0, 32	P0, SV, 32	Kernel Reset	275

(table continues...)

Table 102 (continued) Register overview - CPU domain CSFR (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU_VCON1	CPUx virtualization control register 1	$0B004_H$	P0, 32	P0, SV, 32	Kernel Reset	276
CPU_VCON2	CPUx virtualization control register 2	$0B008_H$	P0, 32	P0, SV, 32	Kernel Reset	276
CPU_BHV	CPUx base hypervisor vector table pointer	$0B010_H$	P0, 32	P0, SV, 32, SFRE	Kernel Reset	277
CPU_VMn_ICR	CPUx Virtual machine n interrupt state	$0B100_H$+ n*4	P0, 32	P0, SV, 32	Kernel Reset	278
CPU_VMn_PETHRESH	CPUx Virtual machine n pre-emption threshold	$0B200_H$+ n*4	P0, 32	P0, SV, 32	Kernel Reset	279
CPU_DPRy_L	CPUx HR specific data protection range y, lower bound register	$0C000_H$+ y*8	P0, 32	P0, SV, 32	Kernel Reset	283
CPU_HVHRA_DPRy_L	CPUx HR specific data protection range y, lower bound register	$0C001_H$+ y*8	P0, 32	P0, SV, 32	Kernel Reset	283
CPU_HVHRB_DPRy_L	CPUx HR specific data protection range y, lower bound register	$0C002_H$+ y*8	P0, 32	P0, SV, 32	Kernel Reset	283
CPU_DPRy_U	CPUx HR specific data protection range y, upper bound register	$0C004_H$+ y*8	P0, 32	P0, SV, 32	Kernel Reset	284
CPU_HVHRA_DPRy_U	CPUx HR specific data protection range y, upper bound register	$0C005_H$+ y*8	P0, 32	P0, SV, 32	Kernel Reset	284
CPU_HVHRB_DPRy_U	CPUx HR specific data protection range y, upper bound register	$0C006_H$+ y*8	P0, 32	P0, SV, 32	Kernel Reset	284
CPU_CPRy_L	CPUx HR specific code protection range y, lower bound register	$0D000_H$+ y*8	P0, 32	P0, SV, 32	Kernel Reset	285
CPU_HVHRA_CPRy_L	CPUx HR specific code protection range y, lower bound register	$0D001_H$+ y*8	P0, 32	P0, SV, 32	Kernel Reset	285
CPU_HVHRB_CPRy_L	CPUx HR specific code protection range y, lower bound register	$0D002_H$+ y*8	P0, 32	P0, SV, 32	Kernel Reset	285
CPU_CPRy_U	CPUx HR specific code protection range y, upper bound register	$0D004_H$+ y*8	P0, 32	P0, SV, 32	Kernel Reset	286
CPU_HVHRA_CPRy_U	CPUx HR specific code protection range y, upper bound register	$0D005_H$+ y*8	P0, 32	P0, SV, 32	Kernel Reset	286

(table continues...)

Table 102 (continued) Register overview - CPU domain CSFR (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU_HVHRB_CPRy_U	CPUx HR specific code protection range y, upper bound register	$0D006_H$+y*8	P0, 32	P0, SV, 32	Kernel Reset	286
CPU_CPXE_y	CPUx HR specific code protection execute enable register set y	$0E000_H$+y*4	P0, 32	P0, SV, 32	Kernel Reset	287
CPU_HVHRA_CPXE_y	CPUx HR specific code protection execute enable register set y	$0E001_H$+y*4	P0, 32	P0, SV, 32	Kernel Reset	287
CPU_HVHRB_CPXE_y	CPUx HR specific code protection execute enable register set y	$0E002_H$+y*4	P0, 32	P0, SV, 32	Kernel Reset	287
CPU_DPRE_y	CPUx HR specific data protection read enable register set y	$0E010_H$+y*4	P0, 32	P0, SV, 32	Kernel Reset	288
CPU_HVHRA_DPRE_y	CPUx HR specific data protection read enable register set y	$0E011_H$+y*4	P0, 32	P0, SV, 32	Kernel Reset	288
CPU_HVHRB_DPRE_y	CPUx HR specific data protection read enable register set y	$0E012_H$+y*4	P0, 32	P0, SV, 32	Kernel Reset	288
CPU_DPWE_y	CPUx HR specific data protection write enable register set y	$0E020_H$+y*4	P0, 32	P0, SV, 32	Kernel Reset	289
CPU_HVHRA_DPWE_y	CPUx HR specific data protection write enable register set y	$0E021_H$+y*4	P0, 32	P0, SV, 32	Kernel Reset	289
CPU_HVHRB_DPWE_y	CPUx HR specific data protection write enable register set y	$0E022_H$+y*4	P0, 32	P0, SV, 32	Kernel Reset	289
CPU_CPXE_y	CPUx HR specific code protection execute enable register set y	$0E040_H$+(y-4)*4	P0, 32	P0, SV, 32	Kernel Reset	290
CPU_HVHRA_CPXE_y	CPUx HR specific code protection execute enable register set y	$0E041_H$+(y-4)*4	P0, 32	P0, SV, 32	Kernel Reset	290
CPU_HVHRB_CPXE_y	CPUx HR specific code protection execute enable register set y	$0E042_H$+(y-4)*4	P0, 32	P0, SV, 32	Kernel Reset	290
CPU_DPRE_y	CPUx HR specific data protection read enable register set y	$0E050_H$+(y-4)*4	P0, 32	P0, SV, 32	Kernel Reset	291
CPU_HVHRA_DPRE_y	CPUx HR specific data protection read enable register set y	$0E051_H$+(y-4)*4	P0, 32	P0, SV, 32	Kernel Reset	291

(table continues...)

Table 102 (continued) Register overview - CPU domain CSFR (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU_HVHRB_DPRE_y	CPUx HR specific data protection read enable register set y	$0E052_H$+(y-4)*4	P0, 32	P0, SV, 32	Kernel Reset	291
CPU_DPWE_y	CPUx HR specific data protection write enable register set y	$0E060_H$+(y-4)*4	P0, 32	P0, SV, 32	Kernel Reset	292
CPU_HVHRA_DPWE_y	CPUx HR specific data protection write enable register set y	$0E061_H$+(y-4)*4	P0, 32	P0, SV, 32	Kernel Reset	292
CPU_HVHRB_DPWE_y	CPUx HR specific data protection write enable register set y	$0E062_H$+(y-4)*4	P0, 32	P0, SV, 32	Kernel Reset	292
CPU_TPS_CON	CPUx HR specific temporal protection system control register	$0E400_H$	P0, 32	P0, SV, 32	Kernel Reset	296
CPU_HVHRA_TPS_CON	CPUx HR specific temporal protection system control register	$0E401_H$	P0, 32	P0, SV, 32	Kernel Reset	296
CPU_HVHRB_TPS_CON	CPUx HR specific temporal protection system control register	$0E402_H$	P0, 32	P0, SV, 32	Kernel Reset	296
CPU_TPS_TIMERn	CPUx HR specific temporal protection system timer register n	$0E404_H$+n*4	P0, 32	P0, SV, 32	Kernel Reset	297
CPU_HVHRA_TPS_TIMERn	CPUx HR specific temporal protection system timer register n	$0E405_H$+n*4	P0, 32	P0, SV, 32	Kernel Reset	297
CPU_HVHRB_TPS_TIMERn	CPUx HR specific temporal protection system timer register n	$0E406_H$+n*4	P0, 32	P0, SV, 32	Kernel Reset	297
CPU_TRnEVT	CPUx trigger event n	$0F000_H$+n*8	P0, 32	P0, SV, 32	Debug Reset	209
CPU_TRnADR	CPUx trigger address n	$0F004_H$+n*8	P0, 32	P0, SV, 32	Debug Reset	210
CPU_CCTRL	CPUx HR specific counter control	$0FC00_H$	P0, 32	P0, SV, 32	Kernel Reset	226
CPU_HVHRA_CCTRL	CPUx HR specific counter control	$0FC01_H$	P0, 32	P0, SV, 32	Kernel Reset	226
CPU_HVHRB_CCTRL	CPUx HR specific counter control	$0FC02_H$	P0, 32	P0, SV, 32	Kernel Reset	226
CPU_CCNT	CPUx HR specific CPU clock cycle count	$0FC04_H$	P0, 32	P0, SV, 32	Kernel Reset	227
CPU_HVHRA_CCNT	CPUx HR specific CPU clock cycle count	$0FC05_H$	P0, 32	P0, SV, 32	Kernel Reset	227

(table continues...)

Table 102 (continued) Register overview - CPU domain CSFR (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU_HVHRB_CCNT	CPUx HR specific CPU clock cycle count	$0FC06_H$	P0, 32	P0, SV, 32	Kernel Reset	227
CPU_ICNT	CPUx HR specific instruction count	$0FC08_H$	P0, 32	P0, SV, 32	Kernel Reset	228
CPU_HVHRA_ICNT	CPUx HR specific instruction count	$0FC09_H$	P0, 32	P0, SV, 32	Kernel Reset	228
CPU_HVHRB_ICNT	CPUx HR specific instruction count	$0FC0A_H$	P0, 32	P0, SV, 32	Kernel Reset	228
CPU_M1CNT	CPUx HR specific multi-count register 1	$0FC0C_H$	P0, 32	P0, SV, 32	Kernel Reset	229
CPU_HVHRA_M1CNT	CPUx HR specific multi-count register 1	$0FC0D_H$	P0, 32	P0, SV, 32	Kernel Reset	229
CPU_HVHRB_M1CNT	CPUx HR specific multi-count register 1	$0FC0E_H$	P0, 32	P0, SV, 32	Kernel Reset	229
CPU_M2CNT	CPUx HR specific multi-count register 2	$0FC10_H$	P0, 32	P0, SV, 32	Kernel Reset	230
CPU_HVHRA_M2CNT	CPUx HR specific multi-count register 2	$0FC11_H$	P0, 32	P0, SV, 32	Kernel Reset	230
CPU_HVHRB_M2CNT	CPUx HR specific multi-count register 2	$0FC12_H$	P0, 32	P0, SV, 32	Kernel Reset	230
CPU_M3CNT	CPUx HR specific multi-count register 3	$0FC14_H$	P0, 32	P0, SV, 32	Kernel Reset	231
CPU_HVHRA_M3CNT	CPUx HR specific multi-count register 3	$0FC15_H$	P0, 32	P0, SV, 32	Kernel Reset	231
CPU_HVHRB_M3CNT	CPUx HR specific multi-count register 3	$0FC16_H$	P0, 32	P0, SV, 32	Kernel Reset	231
CPU_DBGSR	CPUx debug status register	$0FD00_H$	P0, 32	P0, SV, 32	Debug Reset	211
CPU_EXEVT	CPUx external event register	$0FD08_H$	P0, 32	P0, SV, 32	Debug Reset	212
CPU_CREVT	CPUx core register access event	$0FD0C_H$	P0, 32	P0, SV, 32	Debug Reset	213
CPU_SWEVT	CPUx software debug event	$0FD10_H$	P0, 32	P0, SV, 32	Debug Reset	214
CPU_DBGACT	CPUx debug action configuration	$0FD14_H$	P0, 32	P0, SV, 32	Debug Reset	215
CPU_TRIG_ACC	CPUx trigger event accumulator	$0FD30_H$	P0, 32	P0, SV, 32	Debug Reset	217
CPU_DMS	CPUx debug monitor start address	$0FD40_H$	P0, 32	P0, SV, 32	Debug Reset	218
CPU_DCX	CPUx debug context save area pointer	$0FD44_H$	P0, 32	P0, SV, 32	Debug Reset	218

(table continues...)

Table 102 (continued) Register overview - CPU domain CSFR (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU_DBGTCR	CPUx debug trap control	$0FD48_H$	P0, 32	P0, SV, 32	Application Reset	219
CPU_DBGCFG	CPUx debug configuration	$0FD4C_H$	P0, 32	P0, SV, 32	Debug Reset	220
CPU_TRCCFG	CPUx trace configuration	$0FD50_H$	P0, 32	P0, SV, 32	Debug Reset	221
CPU_TRCFILT	CPUx trace filter	$0FD54_H$	P0, 32	P0, SV, 32	Debug Reset	222
CPU_TRCLIM	CPUx trace bandwidth limiter	$0FD58_H$	P0, 32	P0, SV, 32	Debug Reset	224
CPU_TS16PTCCTRL	CPUx MCDS performance counter control register	$0FD60_H$	P0, 32	P0, SV, 32	Debug Reset	225
CPU_PCXI	CPUx previous context information register	$0FE00_H$	P0, 32	P0, SV, 32	Kernel Reset	236
CPU_PSW	CPUx HR specific program status word	$0FE04_H$	P0, 32	P0, SV, 32	Kernel Reset	237
CPU_PC	CPUx program counter	$0FE08_H$	P0, 32	P0, SV, 32	Kernel Reset	240
CPU_CORECON	CPUx HR specific core configuration register	$0FE14_H$	P0, 32	SFRSE, P0, SV, 32	Kernel Reset	250
CPU_HVHRA_CORECON	CPUx HR specific core configuration register	$0FE15_H$	P0, 32	SFRSE, P0, SV, 32	Kernel Reset	250
CPU_HVHRB_CORECON	CPUx HR specific core configuration register	$0FE16_H$	P0, 32	SFRSE, P0, SV, 32	Kernel Reset	250
CPU_CPU_ID	CPUx identification register	$0FE18_H$	P0, 32	P0, SV, 32	Kernel Reset	241
CPU_CORE_ID	CPUx core identification register	$0FE1C_H$	P0, 32	P0, SV, 32	Kernel Reset	242
CPU_BIV	CPUx HR specific base interrupt vector table pointer	$0FE20_H$	P0, 32	SFRE, P0, SV, 32	Kernel Reset	251
CPU_HVHRA_BIV	CPUx HR specific base interrupt vector table pointer	$0FE21_H$	P0, 32	SFRE, P0, SV, 32	Kernel Reset	251
CPU_HVHRB_BIV	CPUx HR specific base interrupt vector table pointer	$0FE22_H$	P0, 32	SFRE, P0, SV, 32	Kernel Reset	251
CPU_BTV	CPUx HR specific base trap vector table pointer	$0FE24_H$	P0, 32	SFRE, P0, SV, 32	Kernel Reset	252
CPU_HVHRA_BTV	CPUx HR specific base trap vector table pointer	$0FE25_H$	P0, 32	SFRE, P0, SV, 32	Kernel Reset	252
CPU_HVHRB_BTV	CPUx HR specific base trap vector table pointer	$0FE26_H$	P0, 32	SFRE, P0, SV, 32	Kernel Reset	252
CPU_ISP	CPUx HR specific interrupt stack pointer	$0FE28_H$	P0, 32	SFRE, P0, SV, 32	Kernel Reset	253

(table continues...)

Table 102 (continued) Register overview - CPU domain CSFR (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU_HVHRA_ISP	CPUx HR specific interrupt stack pointer	$0FE29_H$	P0, 32	SFRE, P0, SV, 32	Kernel Reset	253
CPU_HVHRB_ISP	CPUx HR specific interrupt stack pointer	$0FE2A_H$	P0, 32	SFRE, P0, SV, 32	Kernel Reset	253
CPU_ICR	CPUx interrupt control register	$0FE2C_H$	P0, 32	P0, SV, 32	Kernel Reset	242
CPU_PPRS	CPUx previous PRS	$0FE34_H$	P0, 32	P0, SV, 32	Kernel Reset	244
CPU_FCX	CPUx HR specific free CSA list head pointer	$0FE38_H$	P0, 32	P0, SV, 32	Kernel Reset	254
CPU_HVHRA_FCX	CPUx HR specific free CSA list head pointer	$0FE39_H$	P0, 32	P0, SV, 32	Kernel Reset	254
CPU_HVHRB_FCX	CPUx HR specific free CSA list head pointer	$0FE3A_H$	P0, 32	P0, SV, 32	Kernel Reset	254
CPU_LCX	CPUx HR specific free CSA list limit pointer	$0FE3C_H$	P0, 32	P0, SV, 32	Kernel Reset	255
CPU_HVHRA_LCX	CPUx HR specific free CSA list limit pointer	$0FE3D_H$	P0, 32	P0, SV, 32	Kernel Reset	255
CPU_HVHRB_LCX	CPUx HR specific free CSA list limit pointer	$0FE3E_H$	P0, 32	P0, SV, 32	Kernel Reset	255
CPU_SWID	CPUx software identification	$0FE40_H$	P0, 32	P0, SV, 32	Kernel Reset	244
CPU_CUS_ID	CPUx customer ID register	$0FE50_H$	P0, 32	P0, SV, 32	Kernel Reset	245
CPU_BOOTCON	CPUx core boot configuration register	$0FE60_H$	P0, 32	P0, SV, 32	Kernel Reset	246
CPU_LCLCON	CPUx lockstep control register	$0FE64_H$	P0, 32	STP, SFRSE, P0, SV, 32	Cold PowerOn Reset	247
CPU_CCON	CPUx clock control register	$0FE68_H$	P0, 32	P0, SV, 32	Kernel Reset	248
CPU_TCCON	CPUx hardware configuration	$0FE6C_H$	P0, 32	P0, SV, 32	Kernel Reset	248
CPU_HVHRA_Ay	CPUx HR specific address general purpose register y	$0FF81_H$+ y*4	P0, 32	P0, SV, 32	Kernel Reset	274
CPU_HVHRB_Ay	CPUx HR specific address general purpose register y	$0FF82_H$+ y*4	P0, 32	P0, SV, 32	Kernel Reset	274
CPU_HVHRA_Ay	CPUx HR specific address general purpose register y	$0FFA1_H$+ (y-8)*4	P0, 32	P0, SV, 32	Kernel Reset	274
CPU_HVHRB_Ay	CPUx HR specific address general purpose register y	$0FFA2_H$+ (y-8)*4	P0, 32	P0, SV, 32	Kernel Reset	274

4.6.1.4 CPUx trigger event n

HRA_TRnEVT (n=0-7) Offset address: $1F000_H$+n*8

CPUx trigger event n Debug Reset value: $0000\ 0000_H$

CPU_TRnEVT (n=0-7) Offset address: $0F000_H$+n*8

Short address for domain CSFR Debug Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0			ALD	AST	0										
r			rw	rw	r										

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		RNG	TYP	0				CNT		0		BBM	0		EN
r		rw	rw	r				rw		r		rw	r		rw

Field	Bits	Type	Description
EN	0	rw	**Trigger enable** Trigger enable. If enabled, the settings defined in debug action configuration register (DGBACT) will determine if the trigger is active or not. Once an even numbered comparator has been set to range, the EN settings of its associated upper neighbour will be ignored. 0_B Disabled 1_B Event enabled according to the settings defined in DBGACT
BBM	3	rw	**Break before make (BBM) or break after make (BAM) selection** Code triggers BBM or BAM selection. ***Note**: Data access and data/code combination access triggers can only create BAM Debug Events. When these triggers occur, TRnEVT.BBM is ignored.* 0_B Triggers break after make (BAM) 1_B Triggers break before make (BBM). Default to BAM for data or combined triggers regardless of this setting.
CNT	7:6	rw	**Counter** When this event occurs adjust the control of the performance counters in task mode as follows: 00_B No change 01_B Start the performance counters 10_B Stop the performance counters 11_B Toggle the performance counter control (i.e. start it if it is currently stopped, stop it if it is currently running)
TYP	12	rw	**Input selection** 0_B Address 1_B Program counter

(table continues...)

(continued)

Field	Bits	Type	Description
RNG	13	rw	**Compare type** Once an even numbered comparator has been set to range, the EN settings of its associated upper neighbour will be ignored. 0_B Equality 1_B Range
AST	27	rw	**Address store** Used in conjunction with TYP=0
ALD	28	rw	**Address load** Used in conjunction with TYP=0
0	2:1, 5:4, 11:8, 26:14, 31:29	r	**Reserved - RES** Read as 0; should be written with 0.

Table 103 Mirrors of HRA_TRnEVT (n=0-7)

Short name	Long name	Offset address
HRB_TRnEVT (n=0-7)	CPUx trigger event n is an alias with matching index variables.	$2F000_H$+n*8
HRHV_TRnEVT (n=0-7)	CPUx trigger event n is an alias with matching index variables.	$3F000_H$+n*8

4.6.1.5 CPUx trigger address n

HRA_TRnADR (n=0-7) Offset address: $1F004_H$+n*8

CPUx trigger address n Debug Reset value: $0000\ 0000_H$

CPU_TRnADR (n=0-7) Offset address: $0F004_H$+n*8

Short address for domain CSFR Debug Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADDR															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR															
rw															

Field	Bits	Type	Description
ADDR	31:0	rw	**Comparison address** Code or data address, or address range. ***Note**: For PC comparison, bit[0] is always zero.*

Table 104 Mirrors of HRA_TRnADR (n=0-7)

Short name	Long name	Offset address
HRB_TRnADR (n=0-7)	CPUx trigger address n is an alias with matching index variables.	$2F004_H$+n*8
HRHV_TRnADR (n=0-7)	CPUx trigger address n is an alias with matching index variables.	$3F004_H$+n*8

4.6.1.6 CPUx debug status register

For configurations without Virtualization or if Virtualization is disabled, the field EVTVMN is reserved and will read as 0

HRA_DBGSR Offset address: $1FD00_H$

CPUx debug status register Debug Reset value: $0000\ 0000_H$

CPU_DBGSR Offset address: $0FD00_H$

Short address for domain CSFR Debug Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0													EVTVMN		
r													rh		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			EVTSRC					PEVT	0		SUSOUT	SUSIN	HALT		DE
r			rh					rwh	r		rwh	rh	rwh		rh

Field	Bits	Type	Description
DE	0	rh	**Debug enable** Determines whether the core debug controller is enabled or not. 0_B The core debug controller is disabled 1_B The core debug controller is enabled
HALT	2:1	rwh	**CPU halt request / status field** HALT can be set or cleared by software. HALT[0] is the actual Halt bit. HALT[1] is a mask bit to specify whether or not HALT[0] is to be updated on a software write. HALT[1] is always read as 0. HALT[1] must be set to 1 in order to update HALT[0] by software (R: read; W: write). 00_B R: CPU running. W: HALT[0] unchanged. 01_B R: CPU halted. W: HALT[0] unchanged. 10_B R: Not Applicable. W: reset HALT[0]. 11_B R: Not Applicable. W: If DBGSR.DE == 1 (The core debug controller is enabled), set HALT[0]. If DBGSR.DE == 0 (The core debug controller is not enabled), HALT[0] is left unchanged.

(table continues...)

(continued)

Field	Bits	Type	Description
SUSIN	3	rh	**Suspend-in halt** State of the Suspend-In signal. 0_B The Suspend-In signal is negated. The CPU is not in Halt Mode, (except when the Halt mechanism is set following a Debug Event or a write to DBGSR.HALT). 1_B The Suspend-In signal is asserted. The CPU is in Halt Mode.
SUSOUT	4	rwh	**Current State of the Core Suspend-Out Signal** 0_B Core suspend-out inactive 1_B Core suspend-out active
PEVT	7	rwh	**Posted event** 0_B No posted event 1_B Posted event
EVTSRC	12:8	rh	**Event source** 00_H External event - EXEVT 01_H Core register access event - CREVT 02_H Software debug event - SWEVT 10_H 16 + n - Trigger event - TRnEVT (n = 0,7) **others,** Reserved
EVTVMN	18:16	rh	**Event virtual machine number** The number of the VM running when the event occurred unless it was an external event, in which case forced to zero.
0	6:5, 15:13, 31:19	r	**Reserved - RES** Read as 0; should be written with 0.

Table 105 Mirrors of HRA_DBGSR

Short name	Long name	Offset address
HRB_DBGSR	CPUx debug status register is an alias.	$2FD00_H$
HRHV_DBGSR	CPUx debug status register is an alias.	$3FD00_H$

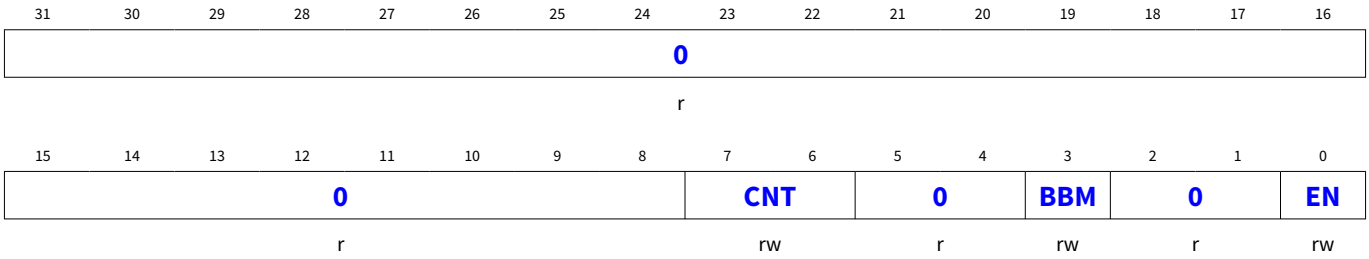
4.6.1.7 CPUx external event register

HRA_EXEVT Offset address: $1FD08_H$

CPUx external event register Debug Reset value: $0000\ 0000_H$

CPU_EXEVT Offset address: $0FD08_H$

Short address for domain CSFR Debug Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
EN	0	rw	**Event enable** Event enable. If enabled, the settings defined in DGBACT will determine if the event is active or not. 0_B Disabled 1_B Event enabled according to the settings defined in DBGACT
BBM	3	rw	**Break before make (BBM) or break after make (BAM) selection** 0_B Break after make (BAM) 1_B Break before make (BBM)
CNT	7:6	rw	**Counter** When this event occurs adjust the control of the performance counters in task mode as follows: 00_B No change 01_B Start the performance counters 10_B Stop the performance counters 11_B Toggle the performance counter control (i.e. start it if it is currently stopped, stop it if it is currently running)
0	2:1, 5:4, 31:8	r	**Reserved - RES** Read as 0; should be written with 0.

Table 106 Mirrors of HRA_EXEVT

Short name	Long name	Offset address
HRB_EXEVT	CPUx external event register is an alias.	$2FD08_H$
HRHV_EXEVT	CPUx external event register is an alias.	$3FD08_H$

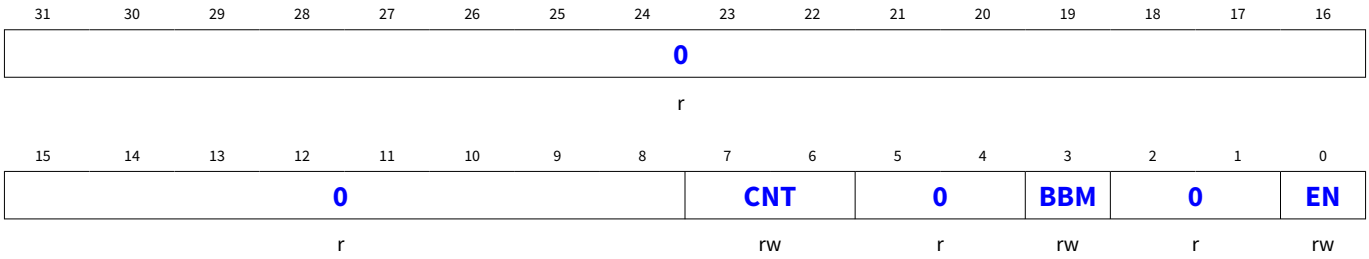
4.6.1.8 CPUx core register access event

HRA_CREVT Offset address: $1FD0C_H$

CPUx core register access event Debug Reset value: $0000\ 0000_H$

CPU_CREVT Offset address: $0FD0C_H$

Short address for domain CSFR Debug Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
EN	0	rw	**Event enable** Event enable. If enabled, the settings defined in DGBACT will determine if the event is active or not. 0_B Disabled 1_B Event enabled according to the settings defined in DBGACT
BBM	3	rw	**Break before make (BBM) or break after make (BAM) selection** 0_B Break after make (BAM) 1_B Break before make (BBM)
CNT	7:6	rw	**Counter** When this event occurs adjust the control of the performance counters in task mode as follows: 00_B No change 01_B Start the performance counters 10_B Stop the performance counters 11_B Toggle the performance counter control (i.e. start it if it is currently stopped, stop it if it is currently running)
0	2:1, 5:4, 31:8	r	**Reserved - RES** Read as 0; should be written with 0.

Table 107 Mirrors of HRA_CREVT

Short name	Long name	Offset address
HRB_CREVT	CPUx core register access event is an alias.	$2FD0C_H$
HRHV_CREVT	CPUx core register access event is an alias.	$3FD0C_H$

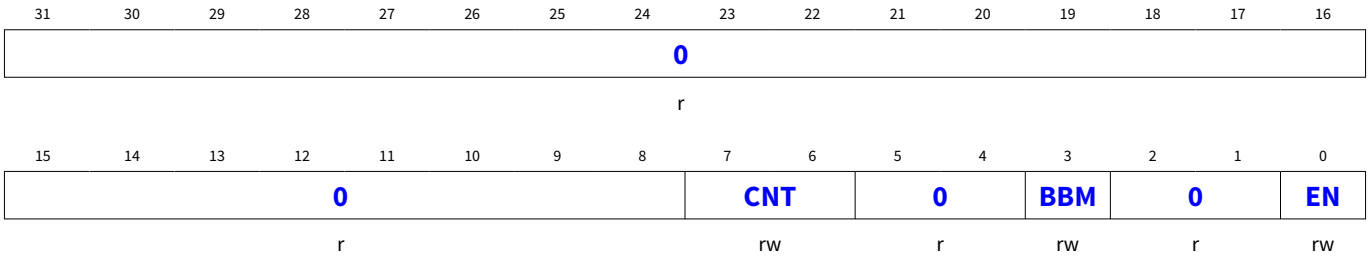
4.6.1.9 CPUx software debug event

HRA_SWEVT Offset address: $1FD10_H$

CPUx software debug event Debug Reset value: $0000\ 0000_H$

CPU_SWEVT Offset address: $0FD10_H$

Short address for domain CSFR Debug Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
EN	0	rw	**Event enable** Event enable. If enabled, the settings defined in DGBACT will determine if the event is active or not. 0_B Disabled 1_B Event enabled according to the settings defined in DBGACT
BBM	3	rw	**Break before make (BBM) or break after make (BAM) selection** 0_B Break after make (BAM) 1_B Break before make (BBM)
CNT	7:6	rw	**Counter** When this event occurs adjust the control of the performance counters in task mode as follows: 00_B No change 01_B Start the performance counters 10_B Stop the performance counters 11_B Toggle the performance counter control (i.e. start it if it is currently stopped, stop it if it is currently running)
0	2:1, 5:4, 31:8	r	**Reserved - RES** Read as 0; should be written with 0.

Table 108 Mirrors of HRA_SWEVT

Short name	Long name	Offset address
HRB_SWEVT	CPUx software debug event is an alias.	$2FD10_H$
HRHV_SWEVT	CPUx software debug event is an alias.	$3FD10_H$

4.6.1.10 CPUx debug action configuration

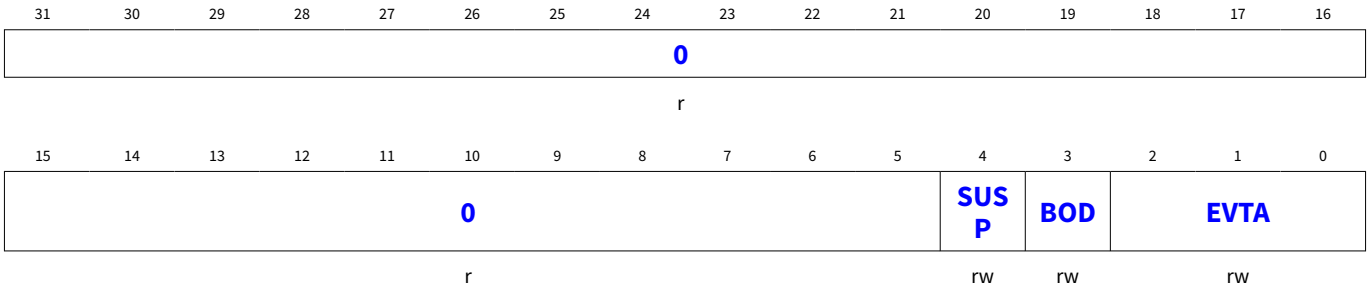
Debug action associated with the enabled debug event (CREVT, SWEVT, TRnEVT and EXEVT)

HRA_DBGACT Offset address: $1FD14_H$

CPUx debug action configuration Debug Reset value: $0000\ 0000_H$

CPU_DBGACT Offset address: $0FD14_H$

Short address for domain CSFR Debug Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
EVTA	2:0	rw	**Event associated** Debug Action associated with the Debug Event: 000_B BOD=0: Disabled. BOD=1: Disabled. 001_B BOD=0: Pulse BRKOUT Signal. BOD=1: None. 010_B BOD=0: Halt and pulse BRKOUT Signal. BOD=1: Halt. 011_B BOD=0: Breakpoint trap and pulse BRKOUT Signal. BOD=1: Breakpoint trap. 100_B BOD=0: Breakpoint interrupt 0 and pulse BRKOUT Signal. BOD=1: Breakpoint interrupt 0. 101_B BOD=0: If implemented, breakpoint interrupt 1 and pulse BRKOUT Signal. BOD=1: If implemented, breakpoint interrupt 1. If not implemented, None. 110_B BOD=0: If implemented, breakpoint interrupt 2 and pulse BRKOUT Signal. BOD=1: If implemented, breakpoint interrupt 2. If not implemented, None. 111_B BOD=0: If implemented, breakpoint interrupt 3 and pulse BRKOUT Signal. BOD=1: If implemented, breakpoint interrupt 3. If not implemented, None.
BOD	3	rw	**Breakout disable** 0_B BRKOUT signal asserted according to the action specified in the EVTA field 1_B BRKOUT signal not asserted. This takes priority over any assertion generated by the EVTA field.
SUSP	4	rw	**CDC Suspend-Out Signal State** Value to be assigend to the DBGSR.SUSOUT when the event is raised. 0_B Do nothing 1_B Set DBGSR.SUSOUT
0	31:5	r	**Reserved - RES** Read as 0; should be written with 0.

Table 109 Mirrors of HRA_DBGACT

Short name	Long name	Offset address
HRB_DBGACT	CPUx debug action configuration is an alias.	$2FD14_H$
HRHV_DBGACT	CPUx debug action configuration is an alias.	$3FD14_H$

4.6.1.11 CPUx trigger event accumulator

This register is cleared by any read operation, write operations are ignored.

HRA_TRIG_ACC — Offset address: $1FD30_H$

CPUx trigger event accumulator — Debug Reset value: $0000\ 0000_H$

CPU_TRIG_ACC — Offset address: $0FD30_H$

Short address for domain CSFR — Debug Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								T7	T6	T5	T4	T3	T2	T1	T0
r								rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
T0	0	rh	**Trigger-0** active since last cleared
T1	1	rh	**Trigger-1** active since last cleared
T2	2	rh	**Trigger-2** active since last cleared
T3	3	rh	**Trigger-3** active since last cleared
T4	4	rh	**Trigger-4** active since last cleared
T5	5	rh	**Trigger-5** active since last cleared
T6	6	rh	**Trigger-6** active since last cleared
T7	7	rh	**Trigger-7** active since last cleared
0	31:8	r	**Reserved - RES** Read as 0; should be written with 0.

Table 110 Mirrors of HRA_TRIG_ACC

Short name	Long name	Offset address
HRB_TRIG_ACC	CPUx trigger event accumulator is an alias.	$2FD30_H$
HRHV_TRIG_ACC	CPUx trigger event accumulator is an alias.	$3FD30_H$

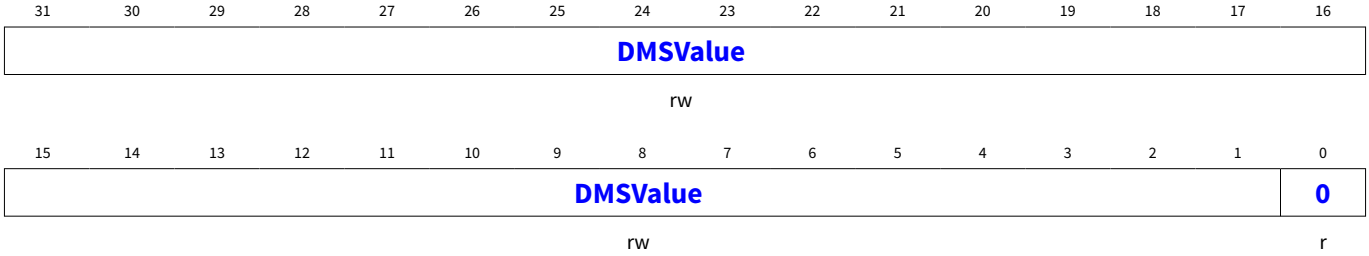
4.6.1.12 CPUx debug monitor start address

HRA_DMS Offset address: $1FD40_H$

CPUx debug monitor start address Reset values see: Table 111

CPU_DMS Offset address: $0FD40_H$

Short address for domain CSFR Reset values see: Table 111



Field	Bits	Type	Description
DMSValue	31:1	rw	**Debug monitor start address** The address at which monitor code execution begins when a breakpoint trap is taken.
0	0	r	**Reserved - RES** Read as 0; should be written with 0.

Table 111 Reset values

Applies to CPU_DMS

Applies to HRA_DMS

Reset type	Reset value	Note
Debug Reset	1010 0000 0000 0000 0000 001X XX00 0000_B	XXX = CORE_ID

Table 112 Mirrors of HRA_DMS

Short name	Long name	Offset address
HRB_DMS	CPUx debug monitor start address is an alias.	$2FD40_H$
HRHV_DMS	CPUx debug monitor start address is an alias.	$3FD40_H$

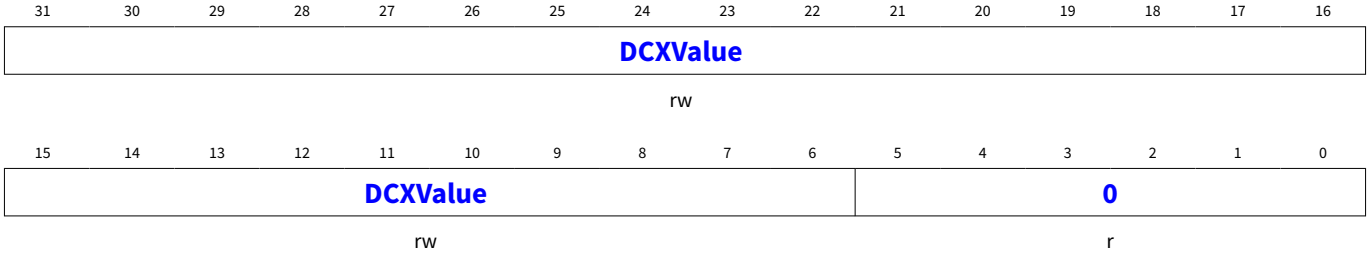
4.6.1.13 CPUx debug context save area pointer

HRA_DCX Offset address: $1FD44_H$

CPUx debug context save area pointer Reset values see: Table 113

CPU_DCX Offset address: $0FD44_H$

Short address for domain CSFR Reset values see: Table 113



Field	Bits	Type	Description
DCXValue	31:6	rw	**Debug context save area pointer** Address where the debug context is stored following a breakpoint trap.
0	5:0	r	**Reserved - RES** Read as 0; should be written with 0.

Table 113 Reset values

Applies to CPU_DCX

Applies to HRA_DCX

Reset type	Reset value	Note
Debug Reset	1010 0000 0000 0000 0000 010X XX00 0000_B	XXX = CORE_ID

Table 114 Mirrors of HRA_DCX

Short name	Long name	Offset address
HRB_DCX	CPUx debug context save area pointer is an alias.	$2FD44_H$
HRHV_DCX	CPUx debug context save area pointer is an alias.	$3FD44_H$

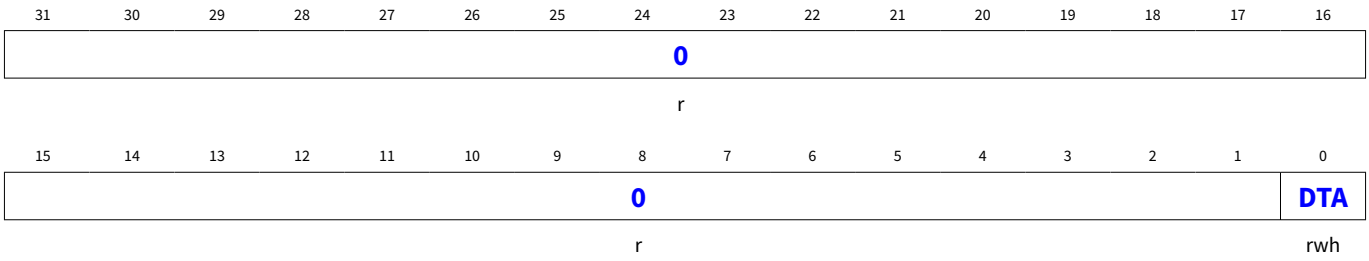
4.6.1.14 CPUx debug trap control

HRA_DBGTCR Offset address: $1FD48_H$

CPUx debug trap control Application Reset value: $0000\ 0001_H$

CPU_DBGTCR Offset address: $0FD48_H$

Short address for domain CSFR Application Reset value: $0000\ 0001_H$



Field	Bits	Type	Description
DTA	0	rwh	**Debug trap active bit** A breakpoint trap may only be taken in the condition DTA == 0. Taking a breakpoint trap sets the DTA bit to one. Further breakpoint traps are therefore disabled until such time as the breakpoint trap handler clears the DTA bit or until the breakpoint trap handler terminates with a RFM. 0_B No breakpoint trap is active 1_B A breakpoint trap is active
0	31:1	r	**Reserved - RES** Read as 0; should be written with 0.

Table 115 Mirrors of HRA_DBGTCR

Short name	Long name	Offset address
HRB_DBGTCR	CPUx debug trap control is an alias.	$2FD48_H$
HRHV_DBGTCR	CPUx debug trap control is an alias.	$3FD48_H$

4.6.1.15 CPUx debug configuration

The TC and TCP bits enable that a tool can reliably configure this register before or after it is configured by SW. The use case is that SW configures it for Trace Based Measurement (TBM) but a debug tool must still have all options. The simple rule is: SW will write the register with TC and TCP set to 0, a tool will always write with both bits set to 1.

For configurations without Virtualization or if Virtualization is disabled, the fields VMn are reserved and will read as 0.

HRA_DBGCFG Offset address: $1FD4C_H$

CPUx debug configuration Debug Reset value: $0000\ 0000_H$

CPU_DBGCFG Offset address: $0FD4C_H$

Short address for domain CSFR Debug Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								VM7	VM6	VM5	VM4	VM3	VM2	VM1	VM0
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN	0													TCP	TC
rw	r													w	rw

Field	Bits	Type	Description
TC	0	rw	**Tool control** 0_B Control by SW The register is updated with the written value. Shall be used by SW 1_B Control by tool Only if TCP is 1 the register is updated with the written value. Shall be used by a tool

(table continues...)

(continued)

Field	Bits	Type	Description
TCP	1	w	**Tool control protection** 0_B TC is not changed. Shall be used by SW 1_B TC is set to the written value The new TC value is already effective for this write. Shall be used by a tool
EN	15	rw	**Debug enable** DBGSR.DE must be 1 to use the core debug functionality. 0_B Disabled 1_B Enabled
VMn (n=0-7)	n+16	rw	**Debug enable for Virtual Machine n** For CPUs with virtualization present and enabled, both EN and VMn bits must be set to enable debugging. 0_B Debug for VMn is blocked 1_B Debug for VMn is enabled if debugging is enabled (EN == 1).
0	14:2, 31:24	r	**Reserved - RES** Read as 0; should be written with 0.

Table 116 Mirrors of HRA_DBGCFG

Short name	Long name	Offset address
HRB_DBGCFG	CPUx debug configuration is an alias.	$2FD4C_H$
HRHV_DBGCFG	CPUx debug configuration is an alias.	$3FD4C_H$

4.6.1.16 CPUx trace configuration

The TC and TCP bits enable that a tool can reliably configure this register before or after it is configured by SW. The use case is that SW configures it for Trace Based Measurement (TBM) but a debug tool must still have all options. The simple rule is: SW will write the register with TC and TCP set to 0, a tool will always write with both bits set to 1.

For configurations without Virtualization or if Virtualization is disabled, the fields VMn are reserved and will read as 0.

HRA_TRCCFG Offset address: $1FD50_H$

CPUx trace configuration Debug Reset value: $0000\ 0000_H$

CPU_TRCCFG Offset address: $0FD50_H$

Short address for domain CSFR Debug Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								VM7	VM6	VM5	VM4	VM3	VM2	VM1	VM0
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN	0													TCP	TC
rw	r													w	rw

Field	Bits	Type	Description
TC	0	rw	**Tool control** 0_B Control by SW The register is updated with the written value. Shall be used by SW 1_B Control by Tool Only if TCP is 1 the register is updated with the written value. Shall be used by a tool
TCP	1	w	**Tool control protection** 0_B TC is not changed. Shall be used by SW 1_B TC is set to the written value The new TC value is already effective for this write. Shall be used by a tool
EN	15	rw	**Trace enable** For all trace sources of the CPU subsystem. For CPUs with Virtualization and Virtualization is enabled, both EN and VM{n} bits must be set to enable tracing. 0_B Disabled 1_B Enabled
VMn (n=0-7)	n+16	rw	**Trace enable for virtual machine n** These bits only exist in CPUs with Virtualization and Virtualization is enabled. 0_B Trace for VMn is blocked 1_B Trace program, data for VMn is output if tracing is enabled (EN == 1)
0	14:2, 31:24	r	**Reserved - RES** Read as 0; should be written with 0.

Table 117 Mirrors of HRA_TRCCFG

Short name	Long name	Offset address
HRB_TRCCFG	CPUx trace configuration is an alias.	$2FD50_H$
HRHV_TRCCFG	CPUx trace configuration is an alias.	$3FD50_H$

4.6.1.17 CPUx trace filter

The TC and TCP bits enable that a tool can reliably configure this register before or after it is configured by SW. The use case is that SW configures it for Trace Based Measurement (TBM) but a debug tool must still have all options. The simple rule is: SW will write the register with TC and TCP set to 0, a tool will always write with both bits set to 1.

SLF, AA and CSA filters are applied independently.

HRA_TRCFILT	Offset address:	$1FD54_H$
CPUx trace filter	Debug Reset value:	$0000\ 0000_H$
CPU_TRCFILT	Offset address:	$0FD54_H$
Short address for domain CSFR	Debug Reset value:	$0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
A15	A14	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0							CSA	AA	0	SLF		0		TCP	TC
r							rw	rw	r	rw		r		w	rw

Field	Bits	Type	Description
TC	0	rw	**Tool control** 0_B Control by SW The register is updated with the written value. Shall be used by SW 1_B Control by Tool Only if TCP is 1 the register is updated with the written value. Shall be used by a tool
TCP	1	w	**Tool control protection** 0_B TC is not changed. Shall be used by SW 1_B TC is set to the written value The new TC value is already effective for this write. Shall be used by a tool
SLF	5:4	rw	**Store/Load filter for relative addressing ST/LD instructions** SLF, AA and CSA filters are applied independently. 00_B All ST/LD instructions pass this filter 01_B Only ST/LD instructions with short offset off10 pass this filter 10_B Only ST/LD instructions which use A[n] when TRCFILT.An==0 pass this filter 11_B Only ST/LD instructions with either off10 or A[n] when TRCFILT.An==0 pass this filter
AA	7	rw	**Filter for absolute addressing ST/LD instructions** Only taken into account if SLF is 10_B or 11_B 0_B ST/LD instructions with absolute addressing pass this filter. 1_B ST/LD instructions with absolute addressing are blocked
CSA	8	rw	**Filter for context save area** 0_B CSA context save and restore pass this filter 1_B CSA context save and restore trace is blocked
An (n=0-15)	n+16	rw	**SLF trace filter for CPU address register An** Only taken into account if SLF is 10_B or 11_B. 0_B ST/LD instructions using A[n] passes this filter 1_B ST/LD instructions using A[n] are blocked
0	3:2, 6, 15:9	r	**Reserved - RES** Read as 0; should be written with 0.

Table 118 Mirrors of HRA_TRCFILT

Short name	Long name	Offset address
HRB_TRCFILT	CPUx trace filter is an alias.	$2FD54_H$
HRHV_TRCFILT	CPUx trace filter is an alias.	$3FD54_H$

4.6.1.18 CPUx trace bandwidth limiter

The TC and TCP bits enable that a tool can reliably configure this register before or after it is configured by SW. The use case is that SW configures it for Trace Based Measurement (TBM) but a debug tool must still have all options. The simple rule is: SW will write the register with TC and TCP set to 0, a tool will always write with both bits set to 1.

HRA_TRCLIM Offset address: $1FD58_H$

CPUx trace bandwidth limiter Debug Reset value: $0000\ 0000_H$

CPU_TRCLIM Offset address: $0FD58_H$

Short address for domain CSFR Debug Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0					DEPTH			STALL				0		TCP	TC
r					rw			rw				r		w	rw

Field	Bits	Type	Description
TC	0	rw	**Tool control** 0_B Control by SW The register is updated with the written value. Shall be used by SW 1_B Control by Tool Only if TCP is 1 the register is updated with the written value. Shall be used by a tool
TCP	1	w	**Tool control protection** 0_B TC is not changed. Shall be used by SW 1_B TC is set to the written value The new TC value is already effective for this write. Shall be used by a tool
STALL	7:4	rw	**Number of CPU write back stall cycles** 0_H The bandwith limiter is disabled 1_H **Const2**: Stall the write buffer by 2 cycles … F_H **Const16**: Stall the write buffer by 16 cycles

(table continues...)

(continued)

Field	Bits	Type	Description
DEPTH	10:8	rw	**Depth of the simulated FIFO** The simulated FIFO depth is DEPTH+1. Once the simulated FIFO depth is reached, the CPU write buffer will be stalled and thus no new trace generated for the STALL number of CPU clock cycles. Note: The minimum value for DEPTH is 1 resulting in a simulated FIFO depth of 2. 000_B Reserved 001_B **Const2**: Simulated FIFO depth of 2 … 111_B **Const8**: Simulated FIFO depth of 8
0	3:2, 31:11	r	**Reserved - RES** Read as 0; should be written with 0.

Table 119 Mirrors of HRA_TRCLIM

Short name	Long name	Offset address
HRB_TRCLIM	CPUx trace bandwidth limiter is an alias.	$2FD58_H$
HRHV_TRCLIM	CPUx trace bandwidth limiter is an alias.	$3FD58_H$

4.6.1.19 CPUx MCDS performance counter control register

HRA_TS16PTCCTRL Offset address: $1FD60_H$

CPUx MCDS performance counter control register Debug Reset value: $0000\ 0000_H$

CPU_TS16PTCCTRL Offset address: $0FD60_H$

Short address for domain CSFR Debug Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0												VMEN0	VM0		
r												rw	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0													TGB	TGS	
r													rw	rw	

Field	Bits	Type	Description
TGS	1:0	rw	**Trigger set for OTGB0/1** 00_B No trigger set output 01_B Trigger set 'TS16P_TC' selected **others,** Reserved
TGB	2	rw	**OTGB0/1 bus select** 0_B Trigger set output on OTGB0 1_B Trigger set output on OTGB1

(table continues...)

(continued)

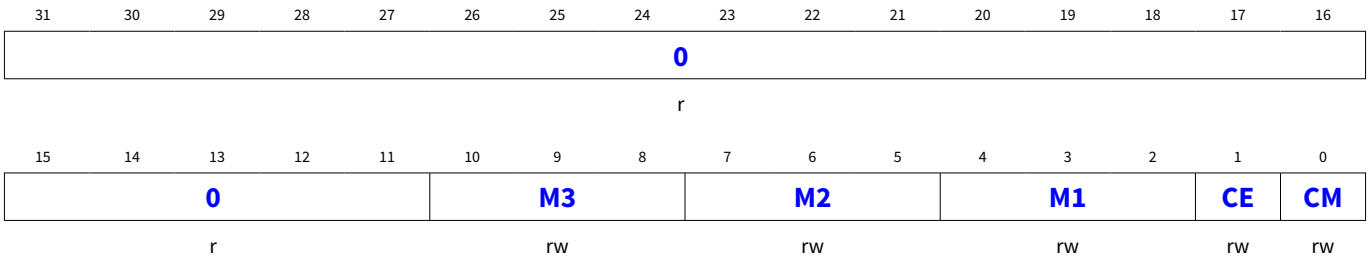
Field	Bits	Type	Description
VM0	18:16	rw	**Virtual machine definition** Restricts the VM for the current CPU. This field is ignored if VMEN0=0_B
VMEN0	19	rw	**Enable for virtual machine definition** 0_B Any VM 1_B Only the virtual machine defined by VM0
0	15:3, 31:20	r	**Reserved** Read as 0; should be written with 0.

Table 120 Mirrors of HRA_TS16PTCCTRL

Short name	Long name	Offset address
HRB_TS16PTCCTRL	CPUx MCDS performance counter control register is an alias.	$2FD60_H$
HRHV_TS16PTCCTRL	CPUx MCDS performance counter control register is an alias.	$3FD60_H$

4.6.1.20 CPUx HR specific counter control

HRA_CCTRL Offset address: $1FC00_H$
CPUx HR specific counter control Kernel Reset value: $0000\ 0000_H$
HRB_CCTRL Offset address: $2FC00_H$
CPUx HR specific counter control Kernel Reset value: $0000\ 0000_H$
HRHV_CCTRL Offset address: $3FC00_H$
CPUx HR specific counter control Kernel Reset value: $0000\ 0000_H$
CPU_CCTRL Offset address: $0FC00_H$
Short address for domain CSFR Kernel Reset value: $0000\ 0000_H$
CPU_HVHRA_CCTRL Offset address: $0FC01_H$
Short address for domain CSFR Kernel Reset value: $0000\ 0000_H$
CPU_HVHRB_CCTRL Offset address: $0FC02_H$
Short address for domain CSFR Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
CM	0	rw	**Counter mode** 0_B Normal mode 1_B Task mode
CE	1	rw	**Count enable** 0_B **Disable**: The counters CCNT, ICNT, M1CNT, M2CNT, M3CNT are disabled 1_B **Enable**: The counters CCNT, ICNT, M1CNT, M2CNT, M3CNT are enabled
M1	4:2	rw	**M1CNT configuration**
M2	7:5	rw	**M2CNT configuration**
M3	10:8	rw	**M3CNT configuration**
0	31:11	r	**Reserved - RES** Read as 0; should be written with 0.

4.6.1.21 CPUx HR specific CPU clock cycle count

HRA_CCNT Offset address: $1FC04_H$

CPUx HR specific CPU clock cycle count Kernel Reset value: $0000\ 0000_H$

HRB_CCNT Offset address: $2FC04_H$

CPUx HR specific CPU clock cycle count Kernel Reset value: $0000\ 0000_H$

HRHV_CCNT Offset address: $3FC04_H$

CPUx HR specific CPU clock cycle count Kernel Reset value: $0000\ 0000_H$

CPU_CCNT Offset address: $0FC04_H$

Short address for domain CSFR Kernel Reset value: $0000\ 0000_H$

CPU_HVHRA_CCNT Offset address: $0FC05_H$

Short address for domain CSFR Kernel Reset value: $0000\ 0000_H$

CPU_HVHRB_CCNT Offset address: $0FC06_H$

Short address for domain CSFR Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SOvf	**CountValue**														
rwh	rwh														

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CountValue															
rwh															

Field	Bits	Type	Description
CountValue	30:0	rwh	**Count value** Current Count of the CPU Clock Cycles.

(table continues...)

(continued)

Field	Bits	Type	Description
SOvf	31	rwh	**Sticky overflow bit** This bit is set by hardware when count value [30:0] = 31'h7FFF_FFFF. It can only be cleared by software.

4.6.1.22 CPUx HR specific instruction count

HRA_ICNT Offset address: $1FC08_H$

CPUx HR specific instruction count Kernel Reset value: $0000\ 0000_H$

HRB_ICNT Offset address: $2FC08_H$

CPUx HR specific instruction count Kernel Reset value: $0000\ 0000_H$

HRHV_ICNT Offset address: $3FC08_H$

CPUx HR specific instruction count Kernel Reset value: $0000\ 0000_H$

CPU_ICNT Offset address: $0FC08_H$

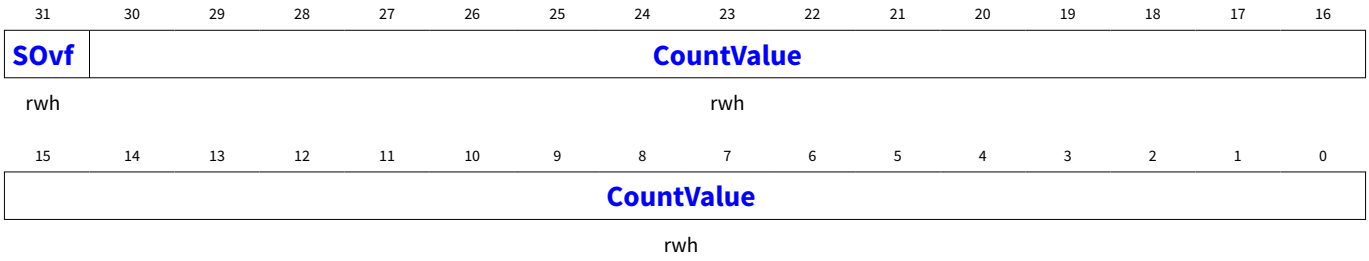
Short address for domain CSFR Kernel Reset value: $0000\ 0000_H$

CPU_HVHRA_ICNT Offset address: $0FC09_H$

Short address for domain CSFR Kernel Reset value: $0000\ 0000_H$

CPU_HVHRB_ICNT Offset address: $0FC0A_H$

Short address for domain CSFR Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
CountValue	30:0	rwh	**Count value** Count of the Instructions Executed.
SOvf	31	rwh	**Sticky overflow bit** This bit is set by hardware when count value [30:0] = 31'h7FFF_FFFF. It can only be cleared by software.

4.6.1.23 CPUx HR specific multi-count register 1

HRA_M1CNT	Offset address:	$1FC0C_H$
CPUx HR specific multi-count register 1	Kernel Reset value:	$0000\ 0000_H$
HRB_M1CNT	Offset address:	$2FC0C_H$
CPUx HR specific multi-count register 1	Kernel Reset value:	$0000\ 0000_H$
HRHV_M1CNT	Offset address:	$3FC0C_H$
CPUx HR specific multi-count register 1	Kernel Reset value:	$0000\ 0000_H$
CPU_M1CNT	Offset address:	$0FC0C_H$
Short address for domain CSFR	Kernel Reset value:	$0000\ 0000_H$
CPU_HVHRA_M1CNT	Offset address:	$0FC0D_H$
Short address for domain CSFR	Kernel Reset value:	$0000\ 0000_H$
CPU_HVHRB_M1CNT	Offset address:	$0FC0E_H$
Short address for domain CSFR	Kernel Reset value:	$0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SOvf	**CountValue**														
rwh	rwh														

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CountValue															
rwh															

Field	Bits	Type	Description
CountValue	30:0	rwh	**Count value** Count of the Selected Event.
SOvf	31	rwh	**Sticky overflow bit** This bit is set by hardware when count value [30:0] = 31'h7FFF_FFFF. It can only be cleared by software.

4.6.1.24 CPUx HR specific multi-count register 2

HRA_M2CNT	Offset address:	$1FC10_H$
CPUx HR specific multi-count register 2	Kernel Reset value:	$0000\ 0000_H$
HRB_M2CNT	Offset address:	$2FC10_H$
CPUx HR specific multi-count register 2	Kernel Reset value:	$0000\ 0000_H$
HRHV_M2CNT	Offset address:	$3FC10_H$
CPUx HR specific multi-count register 2	Kernel Reset value:	$0000\ 0000_H$
CPU_M2CNT	Offset address:	$0FC10_H$
Short address for domain CSFR	Kernel Reset value:	$0000\ 0000_H$
CPU_HVHRA_M2CNT	Offset address:	$0FC11_H$
Short address for domain CSFR	Kernel Reset value:	$0000\ 0000_H$
CPU_HVHRB_M2CNT	Offset address:	$0FC12_H$
Short address for domain CSFR	Kernel Reset value:	$0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SOvf	**CountValue**														
rwh	rwh														

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CountValue															
rwh															

Field	Bits	Type	Description
CountValue	30:0	rwh	**Count value** Count of the Selected Event.
SOvf	31	rwh	**Sticky overflow bit** This bit is set by hardware when count value [30:0] = 31'h7FFF_FFFF. It can only be cleared by software.

4.6.1.25 CPUx HR specific multi-count register 3

HRA_M3CNT	Offset address:	$1FC14_H$
CPUx HR specific multi-count register 3	Kernel Reset value:	$0000\ 0000_H$
HRB_M3CNT	Offset address:	$2FC14_H$
CPUx HR specific multi-count register 3	Kernel Reset value:	$0000\ 0000_H$
HRHV_M3CNT	Offset address:	$3FC14_H$
CPUx HR specific multi-count register 3	Kernel Reset value:	$0000\ 0000_H$
CPU_M3CNT	Offset address:	$0FC14_H$
Short address for domain CSFR	Kernel Reset value:	$0000\ 0000_H$
CPU_HVHRA_M3CNT	Offset address:	$0FC15_H$
Short address for domain CSFR	Kernel Reset value:	$0000\ 0000_H$
CPU_HVHRB_M3CNT	Offset address:	$0FC16_H$
Short address for domain CSFR	Kernel Reset value:	$0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SOvf	**CountValue**														
rwh	rwh														

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CountValue															
rwh															

Field	Bits	Type	Description
CountValue	30:0	rwh	**Count value** Count of the Selected Event.
SOvf	31	rwh	**Sticky overflow bit** This bit is set by hardware when count value [30:0] = 31'h7FFF_FFFF. It can only be cleared by software.

4.6.1.26 CPUx data access cacheability register

The Physical Memory Attributes registers (PMA0, PMA1, PMA2) define the physical memory attribute for each segment in the physical address space. The register is protected by CPUx_PROTSFRE and can be read with the MFCR instruction and written by the MTCR instruction. Note that when changing the value of the registers both the instruction and data caches should be invalidated, a DSYNC instruction should be executed immediately prior to the MTCR with an ISYNC instruction executed immediately following. This is required to maintain coherency of the processors view of memory.

The register PMA0 defines the data access cacheability of the segment in the physical address space. If bit n in the register is set then segment-n will be seen as cacheable for data accesses.

PMA0 is freely programmable with the following restrictions:

- In PMA0 Segment-C and its global address equivalent (Segment[7-CORE_ID], that is 0x701xxxxx, 0x601xxxxx and so on) must have the same value

Irrespective of the setting of the PMA registers the following constraints are always enforced:

- Segments-F and segment-E are constrained to be Peripheral space and hence non-cacheable
- Segment-A is constrained to be non-cacheable memory
- Segment-D (local data memory) and its global address equivalent (Segment[7-CORE_ID], that is 0x700xxxxx, 0x600xxxxx and so on)) are constrained to be non-cacheable for data accesses

Note: CPUx data accesses to CPUy DSPR (x!=y) can be made cacheable by enabling DCn bit (n=0-7) corresponding CPUy DSPR global address.

HRA_PMA0 — Offset address: 18100_H

CPUx data access cacheability register — Kernel Reset value: $0000\ 0300_H$

CPU_PMA0 — Offset address: 08100_H

Short address for domain CSFR — Kernel Reset value: $0000\ 0300_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DC15	DC14	DC13	DC12	DC11	DC10	DC9	DC8	DC7	DC6	DC5	DC4	DC3	DC2	DC1	DC0
r	r	r	rw	rw	r	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
DCn (n=0-9,11-12)	n	rw	**Data access cacheability for segment-n** 0_B Data access to segment-n is non-cacheable 1_B Data access to segment-n is cacheable
DCn (n=10,13-15)	n	r	**Data access cacheability for segment-n** 0_B Data access to segment-n is constrained to be non-cacheable
0	31:16	r	**Reserved - RES** Read as 0; should be written as 0.

Table 121 Mirrors of HRA_PMA0

Short name	Long name	Offset address
HRB_PMA0	CPUx data access cacheability register is an alias.	28100_H
HRHV_PMA0	CPUx data access cacheability register is an alias.	38100_H

4.6.1.27 CPUx code access cacheability register

The Physical Memory Attributes registers (PMA0,PMA1,PMA2) define the physical memory attribute for each segment in the physical address space. The register is protected by CPUx_PROTSFRE and can be read with the MFCR instruction and written by the MTCR instruction. Note that when changing the value of the registers both the instruction and data caches should be invalidated, a DSYNC instruction should be executed immediately prior to the MTCR with an ISYNC instruction executed immediately following. This is required to maintain coherency of the processors view of memory.

The register PMA1 defines the code access cacheability of the segment in the physical address space. If bit n in the register is set then segment-n will be seen as cacheable for code accesses.

PMA1 is freely programmable with the following restrictions:

- In PMA1 Segment-D and its global address equivalent (Segment[7-CORE_ID], that is 0x700xxxxx, 0x600xxxxx and so on) must have the same value

Irrespective of the setting of the PMA registers the following constraints are always enforced:

- Segments-F and segment-E are constrained to be Peripheral space and hence non-cacheable
- Segment-A is constrained to be non-cacheable memory
- Segment-C (local program memory) and its global address equivalent (Segment[7-CORE_ID], that is 0x701xxxxx, 0x601xxxxx and so on) are constrained to be non-cacheable for code accesses

Note: CPUx code accesses to CPUy PSPR (x!=y) can be made cacheable by enabling CCn bit (n=0-7) corresponding CPUy PSPR global address.

HRA_PMA1 Offset address: 18104_H

CPUx code access cacheability register Kernel Reset value: $0000\ 0300_H$

CPU_PMA1 Offset address: 08104_H

Short address for domain CSFR Kernel Reset value: $0000\ 0300_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CC15	CC14	CC13	CC12	CC11	CC10	CC9	CC8	CC7	CC6	CC5	CC4	CC3	CC2	CC1	CC0
r	r	rw	r	rw	r	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
CCn (n=0-9,11,13)	n	rw	**Code access cacheability for segment-n** 0_B Code access to segment-n is non-cacheable 1_B Code access to segment-n is cacheable
CCn (n=10,12,14-15)	n	r	**Code access cacheability for segment-n** 0_B Code access to segment-n is constrained to be non-cacheable
0	31:16	r	**Reserved - RES** Read as 0; should be written as 0.

Table 122 Mirrors of HRA_PMA1

Short name	Long name	Offset address
HRB_PMA1	CPUx code access cacheability register is an alias.	28104_H
HRHV_PMA1	CPUx code access cacheability register is an alias.	38104_H

4.6.1.28 CPUx peripheral space identifier register

The Physical Memory Attributes registers (PMA0,PMA1,PMA2) define the physical memory attribute for each segment in the physical address space. The register is protected by CPUx_PROTSFRE and can be read with the MFCR instruction and written by the MTCR instruction. Note that when changing the value of the registers both the instruction and data caches should be invalidated, a DSYNC instruction should be executed immediately prior to the MTCR with an ISYNC instruction executed immediately following. This is required to maintain coherency of the processors view of memory.

The register PMA2 defines the peripheral space identifier of the segment in the physical address space. If bit n in the register is set then segment-n will be seen as a peripheral segment.PMA2 is a read-only register.

HRA_PMA2 Offset address: 18108_H

CPUx peripheral space identifier register Kernel Reset value: $0000\ C000_H$

CPU_PMA2 Offset address: 08108_H

Short address for domain CSFR Kernel Reset value: $0000\ C000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PSI															
r															

Field	Bits	Type	Description
PSI	15:0	r	**Peripheral space identifier segments F to 0**
0	31:16	r	**Reserved - RES** Read as 0; should be written as 0.

Table 123 Mirrors of HRA_PMA2

Short name	Long name	Offset address
HRB_PMA2	CPUx peripheral space identifier register is an alias.	28108_H
HRHV_PMA2	CPUx peripheral space identifier register is an alias.	38108_H

4.6.1.29 CPUx SIST mode access control register

HRA_SMACON Offset address: $1900C_H$

CPUx SIST mode access control register Kernel Reset value: $0000\ 0000_H$

CPU_SMACON Offset address: $0900C_H$

Short address for domain CSFR Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0							IODT	0							
r							rw	r							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															
r															

Field	Bits	Type	Description
IODT	24	rw	**In-order data transactions** 0_B Normal operation Non-dependent loads bypass stores 1_B In-order operation All memory operations are performed in program order. Loads always flush preceding stores. This should only be performed by test routines at start-up or shut-down as it will severely limit performance
0	23:0, 31:25	r	**Reserved - RES** Read as 0; should be written with 0.

Table 124 Mirrors of HRA_SMACON

Short name	Long name	Offset address
HRB_SMACON	CPUx SIST mode access control register is an alias.	$2900C_H$
HRHV_SMACON	CPUx SIST mode access control register is an alias.	$3900C_H$

4.6.1.30 CPUx compatibility control register

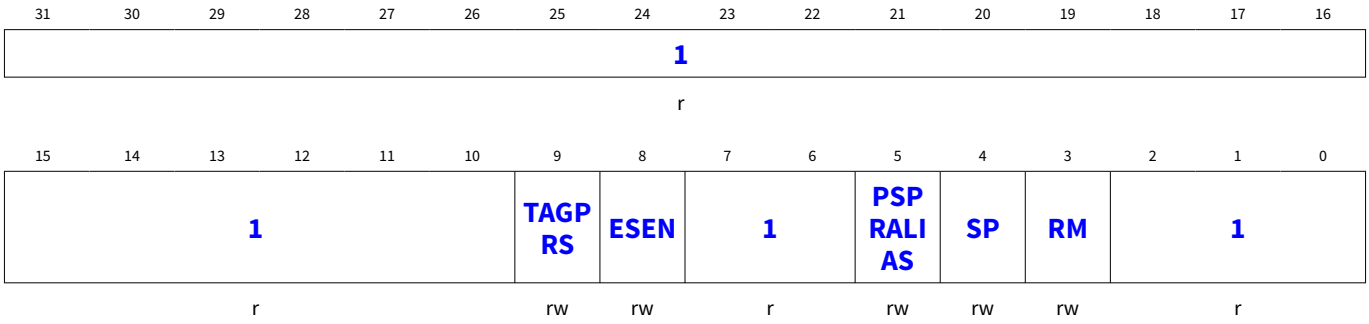
The Compatibility Control Register (COMPAT) is an implementation-specific CSFR which allows certain elements of backwards compatibility with earlier TriCore behavior to be forced. The reset value of the COMPAT register ensures that backwards compatibility with earlier TriCore is enabled by default. This register is protected by CPUx_PROTSFRSE.

HRA_COMPAT Offset address: 19400_H

CPUx compatibility control register Kernel Reset value: FFFF FFFF$_H$

CPU_COMPAT Offset address: 09400_H

Short address for domain CSFR Kernel Reset value: FFFF FFFF$_H$



Field	Bits	Type	Description
RM	3	rw	**Rounding mode compatibility** 0_B PSW.RM not restored by RET 1_B PSW.RM restored by RET (Compatibility mode)
SP	4	rw	**CORECON safety protection mode compatibility** 0_B CORECON[31:1] protected by PROTSFRSE 1_B CORECON[31:1] not protected (Compatibility mode)

(table continues...)

(continued)

Field	Bits	Type	Description
PSPRALIAS	5	rw	**PSPR aliasing compatibility** 0_B PSPR aliasing enabled 1_B PSPR aliasing disabled
ESEN	8	rw	**Emulator space compatibility** 0_B Emulator space disabled 1_B Emulator space Enabled (Compatibility mode)
TAGPRS	9	rw	**Capture PRS in DTAG compatibility** 0_B PRS captured in data TAG 1_B PRS not captured in data TAG (Compatibility mode)
1	2:0, 7:6, 31:10	r	**Reserved - RES** Read as 1; should be written with 1.

Table 125 Mirrors of HRA_COMPAT

Short name	Long name	Offset address
HRB_COMPAT	CPUx compatibility control register is an alias.	29400_H
HRHV_COMPAT	CPUx compatibility control register is an alias.	39400_H

4.6.1.31 CPUx previous context information register

The Previous Context Information Register (PCXI) contains linkage information to the previous execution context, supporting interrupts and automatic context switching. The PCXI is part of a task's state information. The Previous Context Pointer (PCX) holds the address of the CSA of the previous task.

HRA_PCXI Offset address: $1FE00_H$

CPUx previous context information register Kernel Reset value: $0000\ 0000_H$

CPU_PCXI Offset address: $0FE00_H$

Short address for domain CSFR Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		PCPN								PIE	UL	PCXS			
r		rwh								rwh	rwh	rwh			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PCXO															
rwh															

Field	Bits	Type	Description
PCXO	15:0	rwh	**Previous context pointer offset field** The PCXO and PCXS fields form the pointer PCX, which points to the CSA of the previous context.

(table continues...)

(continued)

Field	Bits	Type	Description
PCXS	19:16	rwh	**Previous context pointer segment address** Contains the segment address portion of the PCX. This field is used in conjunction with the PCXO field.
UL	20	rwh	**Upper or lower context tag** Identifies the type of context saved. If the type does not match the type expected when a context restore operation is performed, a trap is generated. 0_B Lower Context 1_B Upper Context
PIE	21	rwh	**Previous interrupt enable** Indicates the state of the interrupt enable bit (ICR.IE) for the interrupted task.
PCPN	29:22	rwh	**Previous CPU priority number** Contains the priority level number of the interrupted task.
0	31:30	r	**Reserved - RES** Read as 0; should be written as 0.

Table 126 Mirrors of HRA_PCXI

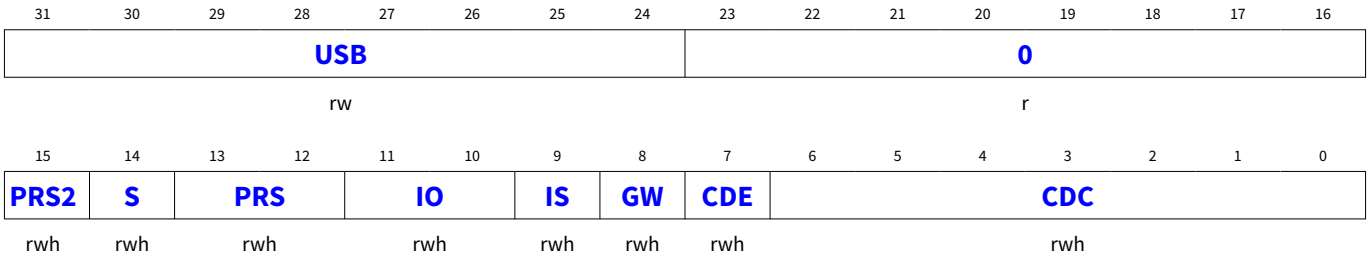
Short name	Long name	Offset address
HRB_PCXI	CPUx previous context information register is an alias.	$2FE00_H$
HRHV_PCXI	CPUx previous context information register is an alias.	$3FE00_H$

4.6.1.32 CPUx HR specific program status word

The Program Status Word register (PSW) is a 32-bit register that contains task-specific architectural state not captured in the General Purpose Register values. The lower half holds control values and parameters related to the protection system, including:

- The Protection Register Set (PRS)
- The I/O privilege level (IO)
- The Interrupt Stack flag (IS)
- The Global register Write permission flag (GW)
- The Call Depth Counter (CDC)
- The Call Depth Count Enable field (CDE)

HRA_PSW	Offset address:	$1FE04_H$
CPUx HR specific program status word	Kernel Reset value:	$0000\ 0B80_H$
CPU_PSW	Offset address:	$0FE04_H$
Short address for domain CSFR	Kernel Reset value:	$0000\ 0B80_H$



Field	Bits	Type	Description
CDC	6:0	rwh	**Call depth counter** Consists of two variable width subfields. The first subfield consists of a string of zero or more initial 1 bits, terminated by the first 0 bit. The remaining bits form the second subfield (CDC.COUNT) which constitutes the Call Depth Count value. The count value is incremented on each Call and is decremented on a Return. $0ccccc_B$: 6-bit counter; trap on overflow $10ccccc_B$: 5-bit counter; trap on overflow $110cccc_B$: 4-bit counter; trap on overflow $1110ccc_B$: 3-bit counter; trap on overflow $11110cc_B$: 2-bit counter; trap on overflow $111110c_B$: 1-bit counter; trap on overflow 1111110_B : Trap every call (Call Trace mode) 1111111_B : Disable Call Depth Counting When the call depth count (CDC.COUNT) overflows a trap (CDO) is generated. Setting the CDC to 1111110_B allows no bits for the counter and causes every call to be trapped. This is used for Call Depth Tracing. Setting the CDC to 1111111_B disables Call Depth Counting.
CDE	7	rwh	**Call depth count enable** Enables call-depth counting, provided that the PSW.CDC mask field is not all set to 1. If PSW.CDC = 1111111_B, call depth counting is disabled regardless of the setting on the PSW.CDE bit. 0_B Call depth counting is temporarily disabled It is automatically re-enabled after execution of the next Call instruction. 1_B Call depth counting is enabled

(table continues...)

(continued)

Field	Bits	Type	Description
GW	8	rwh	**Global address register write permission** Determines whether the current execution thread has permission to modify the global address registers.Most tasks and ISRs use the global address registers as ‘read only’ registers, pointing to the global literal pool and key data structures. However a task or ISR can be designated as the ‘owner’ of a particular global address register, and is allowed to modify it. The system designer must determine which global address variables are used with sufficient frequency and/or in sufficiently time-critical code to justify allocation to a global address register. By compiler convention, global address register A[0] is reserved as the base register for short form loads and stores. Register A[1] is also reserved for compiler use.Registers A[8] and A[9] are not used by the compiler, and are available for holding critical system address variables. 0_B Write permission to global registers A[0], A[1], A[8], A[9] is disabled 1_B Write permission to global registers A[0], A[1], A[8], A[9] is enabled
IS	9	rwh	**Interrupt Stack Control** Determines if the current execution thread is using the shared global (interrupt) stack or a user stack. 0_B USer stack If an interrupt is taken when the IS bit is 0, then the stack pointer register is loaded from the ISP register before execution starts at the first instruction of the Interrupt Service Routine (ISR). 1_B Shared Global Stack If an interrupt is taken when the PSW.IS bit is 1, then the current value of the stack pointer is used by the Interrupt Service Routine (ISR).
IO	11:10	rwh	**Access privilege level control (I/O privilege)** Determines the access level to special function registers and peripheral devices. 00_B User-0 mode No peripheral access. Access to memory regions with the peripheral space attribute are prohibited and results in a PSE or MPP trap. This access level is given to tasks that need not directly access peripheral devices. Tasks at this level do not have permission to enable or disable interrupts. 01_B User-1 mode Regular peripheral access. Enables access to common peripheral devices that are not specially protected, including read/write access to serial I/O ports, read access to timers, and access to most I/O status registers. Tasks at this level may disable interrupts. 10_B Supervisor mode Enables access to all peripheral devices. It enables read/write access to core registers and protected peripheral devices. Tasks at this level may disable interrupts. 11_B Reserved

(table continues...)

(continued)

Field	Bits	Type	Description
PRS	13:12	rwh	**Protection register set** Selects the active Data and Code Memory Protection Register Set. The memory protection register values control load, store and instruction fetches within the current process.
S	14	rwh	**Alternate task identifier selector** Selects which of the master agent function TAG-ID to use for the load/ store accesses (CPUx.D or CPUx.DS)
PRS2	15	rwh	**Protection register set MSB** Selects the active Data and Code Memory Protection Register Set. The memory protection register values control load, store and instruction fetches within the current process.
USB	31:24	rw	**User status bits** The eight most significant bits of the PSW are designated as User Status Bits. These bits may be set or cleared as side effects of instruction execution. Refer to the TriCore Architecture manual for details.
0	23:16	r	**Reserved - RES** Read as 0; should be written with 0.

Table 127 Mirrors of HRA_PSW

Short name	Long name	Offset address
HRB_PSW	CPUx HR specific program status word is an alias.	$2FE04_H$
HRHV_PSW	CPUx HR specific program status word is an alias.	$3FE04_H$

4.6.1.33 CPUx program counter

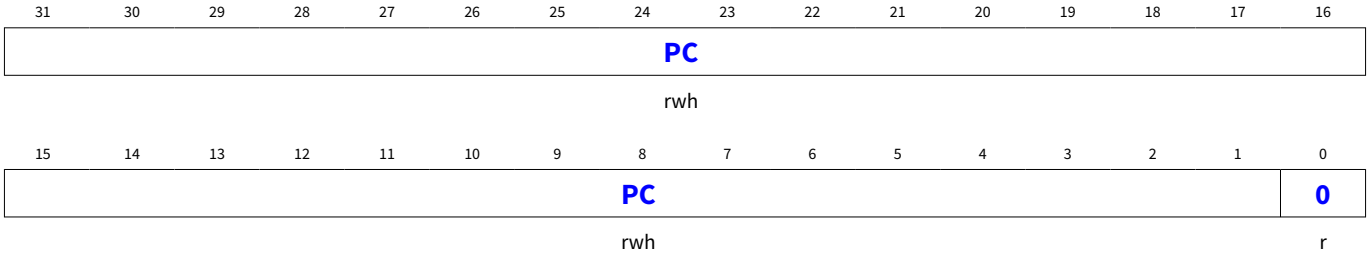
The 32-bit Program Counter (PC) shown below, holds the address of the instruction that is currently running. The Program Counter is part of a task's state information. The PC should only be written via the Core Debug Controller when the core is halted. If the core is not in halt a write will have no effect.

HRA_PC Offset address: $1FE08_H$

CPUx program counter Kernel Reset value: $XXXX\ XXXX_H$

CPU_PC Offset address: $0FE08_H$

Short address for domain CSFR Kernel Reset value: $XXXX\ XXXX_H$



Field	Bits	Type	Description
PC	31:1	rwh	**Program counter**

(table continues...)

(continued)

Field	Bits	Type	Description
0	0	r	**Reserved - RES** Read as 0; should be written as 0.

Table 128 Mirrors of HRA_PC

Short name	Long name	Offset address
HRB_PC	CPUx program counter is an alias.	$2FE08_H$
HRHV_PC	CPUx program counter is an alias.	$3FE08_H$

4.6.1.34 CPUx identification register

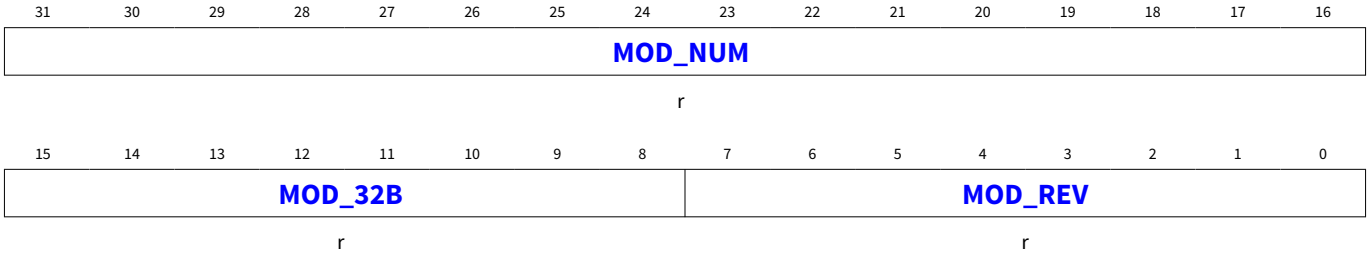
The CPU Identification Register identifies the CPU type and revision.

HRA_CPU_ID Offset address: $1FE18_H$

CPUx identification register Kernel Reset value: $00C0\ C03X_H$

CPU_CPU_ID Offset address: $0FE18_H$

Short address for domain CSFR Kernel Reset value: $00C0\ C03X_H$



Field	Bits	Type	Description
MOD_REV	7:0	r	**Revision number**
MOD_32B	15:8	r	**Module type** A value of $C0_H$ in this field indicates a 32-bit module with a 32-bit module ID register
MOD_NUM	31:16	r	**Module identification number**

Table 129 Mirrors of HRA_CPU_ID

Short name	Long name	Offset address
HRB_CPU_ID	CPUx identification register is an alias.	$2FE18_H$
HRHV_CPU_ID	CPUx identification register is an alias.	$3FE18_H$

4.6.1.35 CPUx core identification register

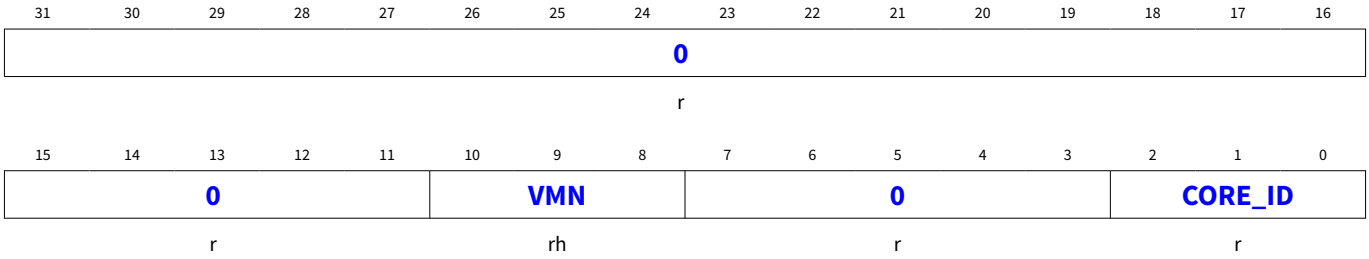
In a multiprocessor system each logical processor core is given a unique identification number. In a system supporting virtualization each logical processor core can be used to run multiple virtual machines. The Core Identification Register holds both the core identification number and the current virtual machine number.

HRA_CORE_ID Offset address: $1FE1C_H$

CPUx core identification register Reset values see: Table 130

CPU_CORE_ID Offset address: $0FE1C_H$

Short address for domain CSFR Reset values see: Table 130



Field	Bits	Type	Description
CORE_ID	2:0	r	**Core identification number** The identification number of the core.
VMN	10:8	rh	**Virutal machine number** The current virutal machine number. When virtualization is disabled or not present this field will read as 1.
0	7:3, 31:11	r	**Reserved - RES** Read as 0.

Table 130 Reset values

Applies to CPU_CORE_ID

Applies to HRA_CORE_ID

Reset type	Reset value	Note
Kernel Reset	0000 0000 0000 0000 0000 0000 0000 $0XXX_B$	Instance specific

Table 131 Mirrors of HRA_CORE_ID

Short name	Long name	Offset address
HRB_CORE_ID	CPUx core identification register is an alias.	$2FE1C_H$
HRHV_CORE_ID	CPUx core identification register is an alias.	$3FE1C_H$

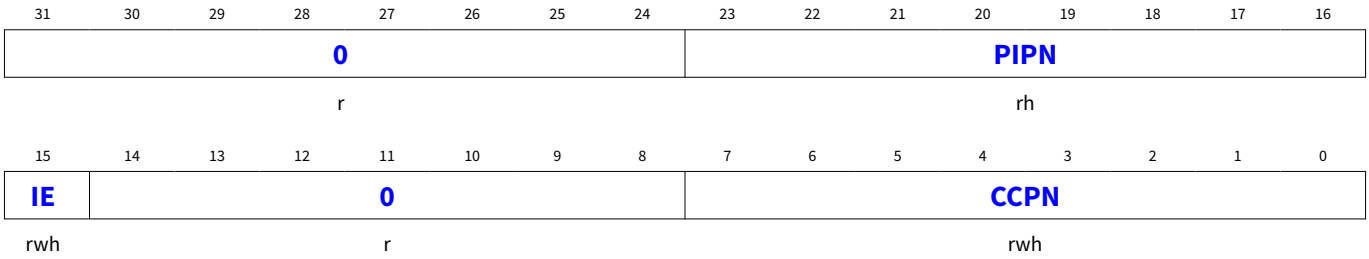
4.6.1.36 CPUx interrupt control register

HRA_ICR Offset address: $1FE2C_H$

CPUx interrupt control register Kernel Reset value: $0000\ 0000_H$

CPU_ICR Offset address: $0FE2C_H$

Short address for domain CSFR Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
CCPN	7:0	rwh	**Current CPU priority number** The Current CPU Priority Number (CCPN) bit field indicates the current priority level of the CPU. It is automatically updated by hardware on entry or exit of Interrupt Service Routines (ISRs) and through the execution of a BISR instruction. CCPN can also be updated through an MTCR instruction.
IE	15	rwh	**Global interrupt enable bit** The interrupt enable bit globally enables the CPU service request system. Whether a service request is delivered to the CPU depends on the individual Service Request Enable Bits (SRE) in the SRNs, and the current state of the CPU. ICR.IE is automatically updated by hardware on entry and exit of an Interrupt Service Routine (ISR). ICR.IE is cleared to 0 when an interrupt is taken, and is restored to the previous value when the ISR executes an RFE instruction to terminate itself. ICR.IE can also be updated through the execution of the ENABLE, DISABLE, MTCR, and BISR instructions. 0_B Interrupt system is globally disabled 1_B Interrupt system is globally enabled
PIPN	23:16	rh	**Pending interrupt priority number** A read-only bit field that is updated by the ICU at the end of each interrupt arbitration process. It indicates the priority number of the pending service request. ICR.PIPN is set to 0 when no request is pending, and at the beginning of each new arbitration process. ... 00_H No valid pending request 01_H Request pending, lowest priority FF_H Request pending, highest priority
0	14:8, 31:24	r	**Reserved - RES** Read as 0; should be written as 0.

Table 132 Mirrors of HRA_ICR

Short name	Long name	Offset address
HRB_ICR	CPUx interrupt control register is an alias.	$2FE2C_H$
HRHV_ICR	CPUx interrupt control register is an alias.	$3FE2C_H$

4.6.1.37 CPUx previous PRS

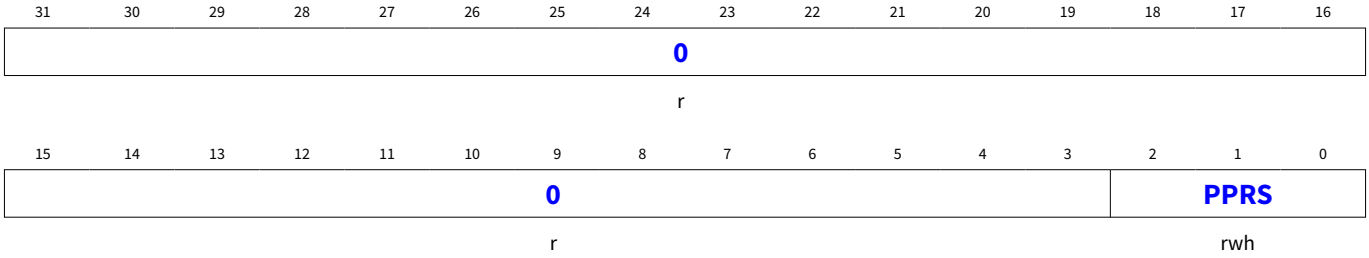
The previous protection register set register (PPRS) contains the value of the protection register set used by the previous execution context. The PPRS is part of a task's state information.

HRA_PPRS Offset address: $1FE34_H$

CPUx previous PRS Kernel Reset value: $0000\ 0000_H$

CPU_PPRS Offset address: $0FE34_H$

Short address for domain CSFR Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
PPRS	2:0	rwh	**Previous PRS**
0	31:3	r	**Reserved - RES** Read as 0; should be written as 0.

Table 133 Mirrors of HRA_PPRS

Short name	Long name	Offset address
HRB_PPRS	CPUx previous PRS is an alias.	$2FE34_H$
HRHV_PPRS	CPUx previous PRS is an alias.	$3FE34_H$

4.6.1.38 CPUx software identification

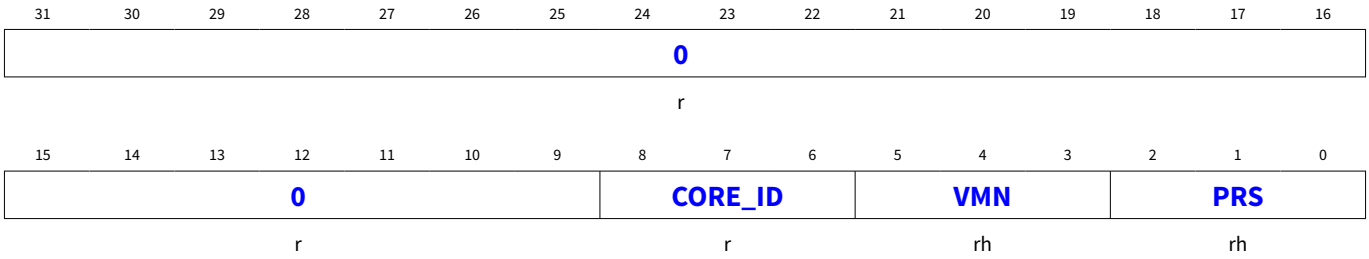
The software identification register holds key identifiers in one single register to ease software configuration.

HRA_SWID Offset address: $1FE40_H$

CPUx software identification Reset values see: Table 134

CPU_SWID Offset address: $0FE40_H$

Short address for domain CSFR Reset values see: Table 134



Field	Bits	Type	Description
PRS	2:0	rh	**Protection register set** Identifies the active level 1 Data and Code Memory Protection Register Set.
VMN	5:3	rh	**Virutal machine number** The current virutal machine number. When virtualization is disabled or not present this field will read as 1.
CORE_ID	8:6	r	**Core identification number** The identification number of the core.
0	31:9	r	**Reserved - RES** Read as 0.

Table 134 Reset values

Applies to CPU_SWID

Applies to HRA_SWID

Reset type	Reset value	Note
Kernel Reset	0000 0000 0000 0000 0000 000X XX00 0000_B	Instance specific

Table 135 Mirrors of HRA_SWID

Short name	Long name	Offset address
HRB_SWID	CPUx software identification is an alias.	$2FE40_H$
HRHV_SWID	CPUx software identification is an alias.	$3FE40_H$

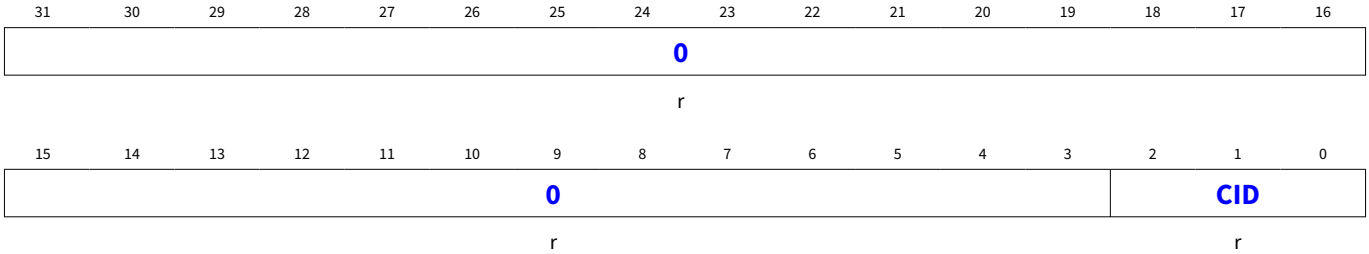
4.6.1.39 CPUx customer ID register

HRA_CUS_ID Offset address: $1FE50_H$

CPUx customer ID register Reset values see: Table 136

CPU_CUS_ID Offset address: $0FE50_H$

Short address for domain CSFR Reset values see: Table 136



Field	Bits	Type	Description
CID	2:0	r	**Customer ID** Alternate core numbering scheme
0	31:3	r	**Reserved - RES** Read as 0.

Table 136 Reset values

Applies to CPU_CUS_ID

Applies to HRA_CUS_ID

Reset type	Reset value	Note
Kernel Reset	0000 0000 0000 0000 0000 0000 0000 $0XXX_B$	Instance specific

Table 137 Mirrors of HRA_CUS_ID

Short name	Long name	Offset address
HRB_CUS_ID	CPUx customer ID register is an alias.	$2FE50_H$
HRHV_CUS_ID	CPUx customer ID register is an alias.	$3FE50_H$

4.6.1.40 CPUx core boot configuration register

The boot configuration register controls the execution behavior following a reset.

HRA_BOOTCON Offset address: $1FE60_H$

CPUx core boot configuration register Reset values see: Table 138

CPU_BOOTCON Offset address: $0FE60_H$

Short address for domain CSFR Reset values see: Table 138

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															BHALT
r															rw

Field	Bits	Type	Description
BHALT	0	rw	**Boot halt** 0_B Core is not in boot halt 1_B Core is in boot halt, write to 0 will exit
0	31:1	r	**Reserved** Read as 0; should be written with 0.

Table 138 Reset values

Applies to CPU_BOOTCON

Applies to HRA_BOOTCON

Reset type	Reset value	Note
Kernel Reset	0000 0000 0000 0000 0000 0000 0000 $000X_B$	Instance specific

Table 139 Mirrors of HRA_BOOTCON

Short name	Long name	Offset address
HRB_BOOTCON	CPUx core boot configuration register is an alias.	$2FE60_H$
HRHV_BOOTCON	CPUx core boot configuration register is an alias.	$3FE60_H$

4.6.1.41 CPUx lockstep control register

Provides control for the CPU Lockstep Comparator Logic blocks.

HRA_LCLCON Offset address: $1FE64_H$

CPUx lockstep control register Cold PowerOn Reset value: $XXXX\ XXXX_H$

CPU_LCLCON Offset address: $0FE64_H$

Short address for domain CSFR Cold PowerOn Reset value: $XXXX\ XXXX_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LSEN	0														LS
rw	r														rh

Field	Bits	Type	Description
LS	0	rh	**Lockstep mode status** This bit indicates whether the CPU is currently running in lockstep monitor mode 0_B Not in lockstep mode 1_B Running in lockstep mode
LSEN	15	rw	**Lockstep enable** This bit may only be written by SSW during boot. Enable lockstep CPU monitoring for the associated processor core. CPU0 is always enabled and cannot be disabled. After cold reset, lockstep is enabled by default. The LSEN bit may be cleared during the boot to disable lockstep mode. SMU lockstep fault reporting should be disabled when lockstep is disabled. 0_B Lockstep is disabled 1_B Lockstep enabled (Default after Cold Power-On Reset)
0	14:1, 31:16	r	**Reserved** Read as 0; should be written with 0.

Table 140 Mirrors of HRA_LCLCON

Short name	Long name	Offset address
HRB_LCLCON	CPUx lockstep control register is an alias.	$2FE64_H$
HRHV_LCLCON	CPUx lockstep control register is an alias.	$3FE64_H$

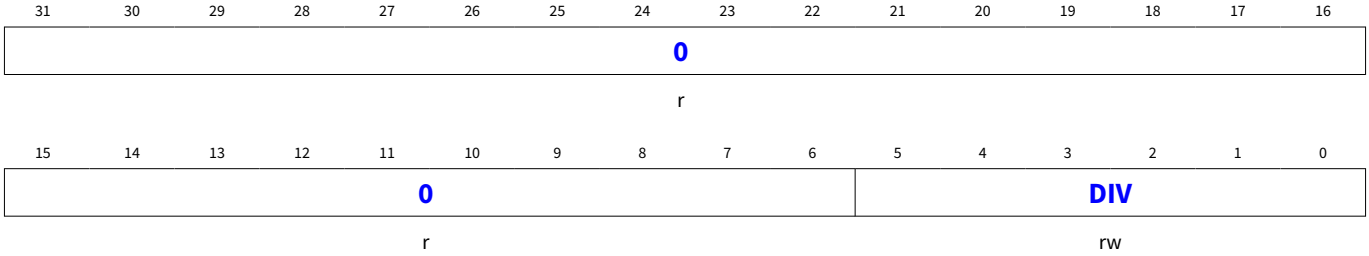
4.6.1.42 CPUx clock control register

HRA_CCON Offset address: $1FE68_H$

CPUx clock control register Kernel Reset value: $0000\ 0000_H$

CPU_CCON Offset address: $0FE68_H$

Short address for domain CSFR Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
DIV	5:0	rw	**Divider reload value** The resulting CPU frequency (performance) is configured to f_{CPU0} = f_{SRI} * (64 - CPU0DIV) / 64. For CPU0DIV = 000000_B, f_{CPU0} = f_{SRI} .
0	31:6	r	**Reserved** Read as 0; Should be written with 0.

Table 141 Mirrors of HRA_CCON

Short name	Long name	Offset address
HRB_CCON	CPUx clock control register is an alias.	$2FE68_H$
HRHV_CCON	CPUx clock control register is an alias.	$3FE68_H$

4.6.1.43 CPUx hardware configuration

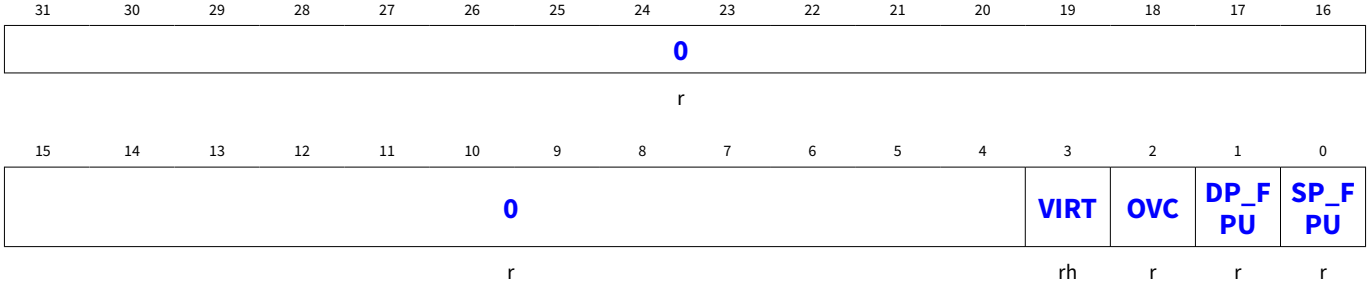
The TriCore core configuration register holds information regarding the hardware support for implementation specific features of the core.

HRA_TCCON Offset address: $1FE6C_H$

CPUx hardware configuration Kernel Reset value: $0000\ 000X_H$

CPU_TCCON Offset address: $0FE6C_H$

Short address for domain CSFR Kernel Reset value: $0000\ 000X_H$



Field	Bits	Type	Description
SP_FPU	0	r	**Single precision FPU hardware support** 0_B Single precision FPU hardware not present 1_B Single precision FPU hardware present
DP_FPU	1	r	**Double precision FPU hardware support** 0_B Double precision FPU hardware not present 1_B Double precision FPU hardware present
OVC	2	r	**Overlay hardware support** 0_B No overlay 1_B Overlay present
VIRT	3	rh	**Virtualization hardware support** 0_B Virtualization hardware extension not present or disabled 1_B Virtualization hardware extension present and enabled
0	31:4	r	**Reserved - RES** Read as 0.

Table 142 Mirrors of HRA_TCCON

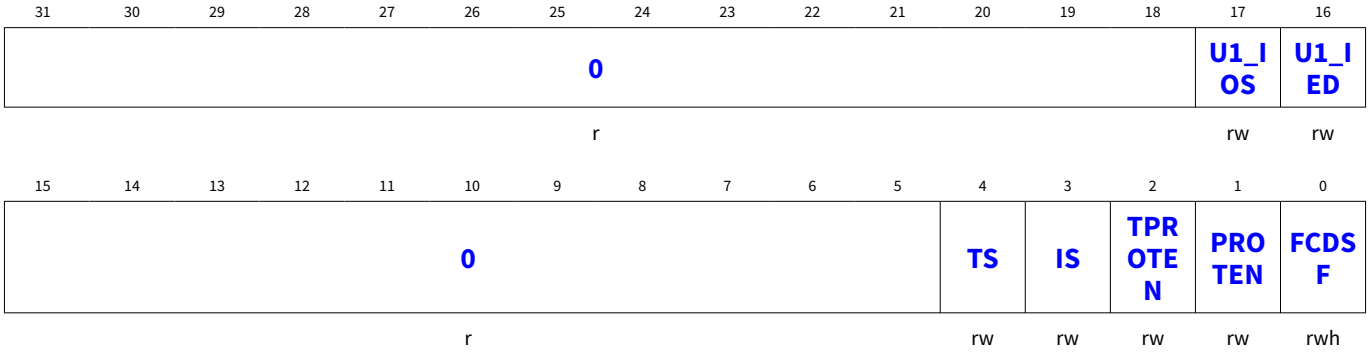
Short name	Long name	Offset address
HRB_TCCON	CPUx hardware configuration is an alias.	$2FE6C_H$
HRHV_TCCON	CPUx hardware configuration is an alias.	$3FE6C_H$

4.6.1.44 CPUx HR specific core configuration register

The Core Configuration Register provides the following functionality:

- Enable bit for temporal protection system
- Enable bit for memory protection system
- Bit for definition of the initial state of the PSW.S bit in interrupt handlers
- Bit for definition of the initial state of the PSW.S bit in trap handlers
- Enable for User-1 IO mode peripheral access as supervisor
- Disable for User-1 IO mode ability to enable and disable interrupts
- Status indicator of the Free Context List Depletion condition

HRA_CORECON	Offset address:	$1FE14_H$
CPUx HR specific core configuration register	Kernel Reset value:	$0000\ 0000_H$
HRB_CORECON	Offset address:	$2FE14_H$
CPUx HR specific core configuration register	Kernel Reset value:	$0000\ 0000_H$
HRHV_CORECON	Offset address:	$3FE14_H$
CPUx HR specific core configuration register	Kernel Reset value:	$0000\ 0000_H$
CPU_CORECON	Offset address:	$0FE14_H$
Short address for domain CSFR	Kernel Reset value:	$0000\ 0000_H$
CPU_HVHRA_CORECON	Offset address:	$0FE15_H$
Short address for domain CSFR	Kernel Reset value:	$0000\ 0000_H$
CPU_HVHRB_CORECON	Offset address:	$0FE16_H$
Short address for domain CSFR	Kernel Reset value:	$0000\ 0000_H$



Field	Bits	Type	Description
FCDSF	0	rwh	**Free context list depleted sticky flag** This sticky bit indicates that a FCD (Free Context List Depleted) trap occurred since the bit was last cleared by software. 0_B No FCD trap occurred since the last clear 1_B An FCD trap occurred since the last clear

(table continues...)

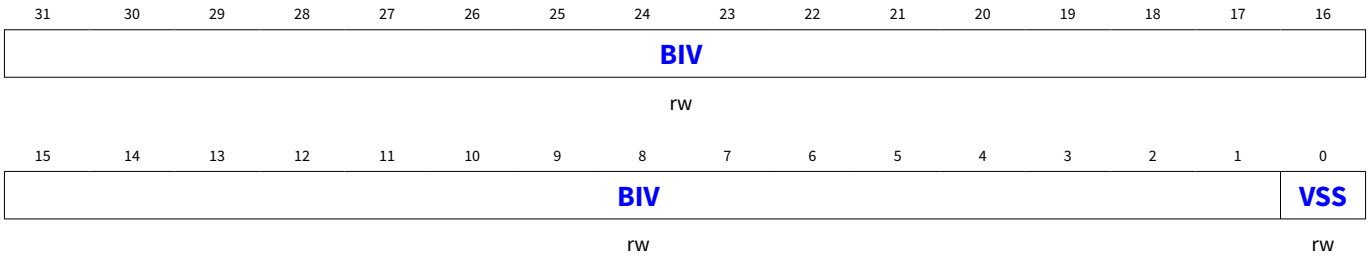
(continued)

Field	Bits	Type	Description
PROTEN	1	rw	**Memory protection enable** Enables the memory protection system. Memory protection is controlled through the memory protection register sets. Note: Initialize the protection register sets prior to setting PROTEN to one. 0_B Memory protection is disabled 1_B Memory protection is enabled
TPROTEN	2	rw	**Temporal protection enable** Enable the temporal protection system. 0_B Temporal protection is disabled 1_B Temporal protection is enabled
IS	3	rw	**Initial state interrupt** Value assigned to the PSW.S bit on interrupt entry
TS	4	rw	**Initial state trap** Value assigned to the PSW.S bit on trap entry
U1_IED	16	rw	**User-1 instruction execution disable** Disable the execution of User-1 mode instructions in User-1 IO mode. Disables User-1 ability to enable and disable interrupts.
U1_IOS	17	rw	**User-1 peripheral access as supervisor** Allow User-1 mode tasks to access peripherals as if in Supervisor mode. Enables User-1 access to all peripheral registers.
0	15:5, 31:18	r	**Reserved - RES** Read as 0; should be written with 0.

4.6.1.45 CPUx HR specific base interrupt vector table pointer

The BIV register contains the base address of the interrupt vector table. When an interrupt is accepted, the entry address into the interrupt vector table is generated from the priority number (taken from the PIPN) of that interrupt, left shifted by either three or five bits, and then ORd with the contents of the BIV register. The left-shift of the interrupt priority number results in a spacing of either eight bytes or 32 bytes between the individual entries in the vector table dependent on the vector spacing selected by the VSS bit.

HRA_BIV	Offset address:	$1FE20_H$
CPUx HR specific base interrupt vector table pointer	Kernel Reset value:	$0000\ 0000_H$
HRB_BIV	Offset address:	$2FE20_H$
CPUx HR specific base interrupt vector table pointer	Kernel Reset value:	$0000\ 0000_H$
HRHV_BIV	Offset address:	$3FE20_H$
CPUx HR specific base interrupt vector table pointer	Kernel Reset value:	$0000\ 0000_H$
CPU_BIV	Offset address:	$0FE20_H$
Short address for domain CSFR	Kernel Reset value:	$0000\ 0000_H$
CPU_HVHRA_BIV	Offset address:	$0FE21_H$
Short address for domain CSFR	Kernel Reset value:	$0000\ 0000_H$
CPU_HVHRB_BIV	Offset address:	$0FE22_H$
Short address for domain CSFR	Kernel Reset value:	$0000\ 0000_H$



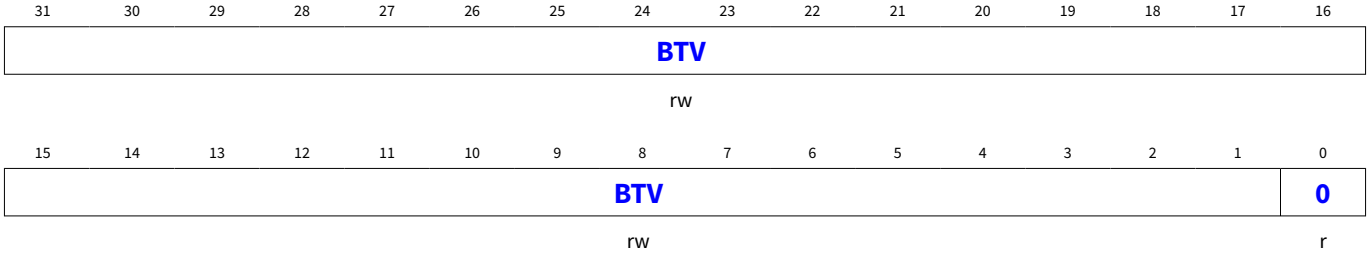
Field	Bits	Type	Description
VSS	0	rw	**Vector spacing select** 0_B 32 byte vector spacing 1_B 8 Byte vector spacing
BIV	31:1	rw	**Base address of interrupt vector table** The address in the BIV register must be aligned to an even byte address (halfword address). Because of the simple ORing of the left-shifted priority number and the contents of the BIV register, the alignment of the base address of the vector table must be to a power of two boundary, dependent on the number of interrupt entries used. For the full range of 256 interrupt entries an alignment to an 8 KByte boundary is required. If fewer sources are used, the alignment requirements are correspondingly relaxed.

4.6.1.46 CPUx HR specific base trap vector table pointer

The BTV contains the base address of the trap vector table. When a trap occurs, the entry address into the trap vector table is generated from the Trap Class of that trap, left-shifted by 5 bits and then ORd with the contents of the BTV register. The left-shift of the Trap Class results in a spacing of 8 words (32 bytes) between the individual entries in the vector table.

Note: This register is protected by CPUx_PROTSFRE.

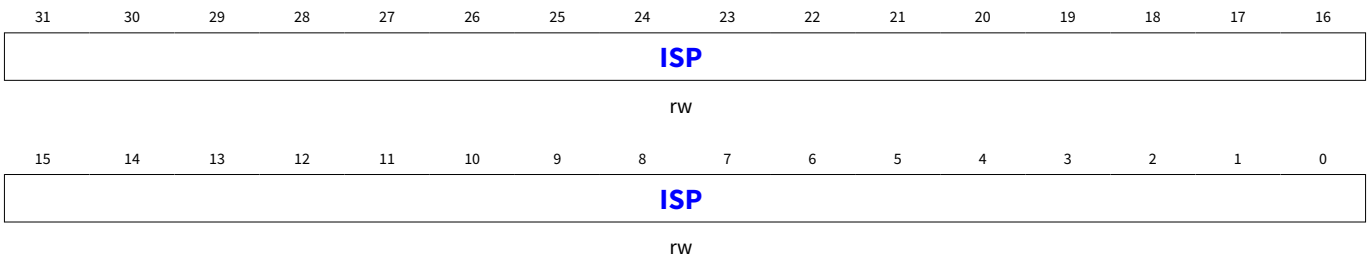
HRA_BTV	Offset address:	$1FE24_H$
CPUx HR specific base trap vector table pointer	Kernel Reset value:	XXXX $FE00_H$
HRB_BTV	Offset address:	$2FE24_H$
CPUx HR specific base trap vector table pointer	Kernel Reset value:	XXXX $FE00_H$
HRHV_BTV	Offset address:	$3FE24_H$
CPUx HR specific base trap vector table pointer	Kernel Reset value:	XXXX $FE00_H$
CPU_BTV	Offset address:	$0FE24_H$
Short address for domain CSFR	Kernel Reset value:	XXXX $FE00_H$
CPU_HVHRA_BTV	Offset address:	$0FE25_H$
Short address for domain CSFR	Kernel Reset value:	XXXX $FE00_H$
CPU_HVHRB_BTV	Offset address:	$0FE26_H$
Short address for domain CSFR	Kernel Reset value:	XXXX $FE00_H$



Field	Bits	Type	Description
BTV	31:1	rw	**Base address of trap vector table** The address in the BTV register must be aligned to an even byte address (halfword address). Also, due to the simple ORing of the left-shifted trap identification number and the contents of the BTV register, the alignment of the base address of the vector table must be to a power of two boundary. There are eight different trap classes, resulting in Trap Classes from 0 to 7. The contents of BTV should therefore be set to at least a 256 byte boundary (8 Trap Classes * 8 word spacing).
0	0	r	**Reserved - RES** Read as 0; should be written as 0.

4.6.1.47 CPUx HR specific interrupt stack pointer

HRA_ISP	Offset address:	$1FE28_H$
CPUx HR specific interrupt stack pointer	Kernel Reset value:	$0000\ 0100_H$
HRB_ISP	Offset address:	$2FE28_H$
CPUx HR specific interrupt stack pointer	Kernel Reset value:	$0000\ 0100_H$
HRHV_ISP	Offset address:	$3FE28_H$
CPUx HR specific interrupt stack pointer	Kernel Reset value:	$0000\ 0100_H$
CPU_ISP	Offset address:	$0FE28_H$
Short address for domain CSFR	Kernel Reset value:	$0000\ 0100_H$
CPU_HVHRA_ISP	Offset address:	$0FE29_H$
Short address for domain CSFR	Kernel Reset value:	$0000\ 0100_H$
CPU_HVHRB_ISP	Offset address:	$0FE2A_H$
Short address for domain CSFR	Kernel Reset value:	$0000\ 0100_H$

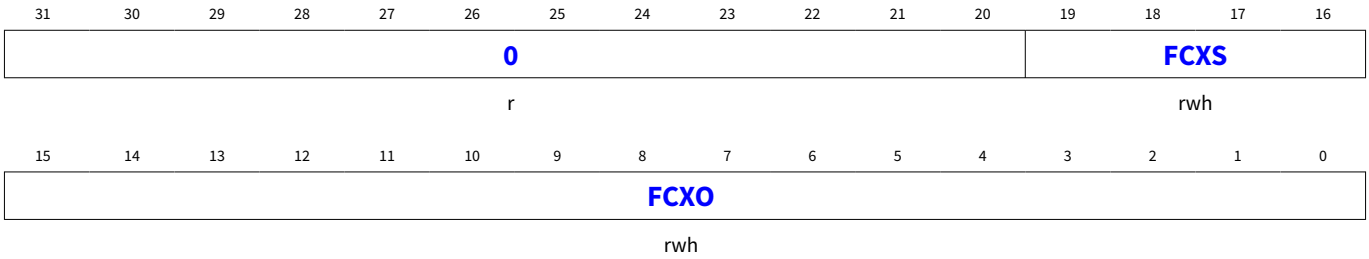


Field	Bits	Type	Description
ISP	31:0	rw	**Interrupt stack pointer**

4.6.1.48 CPUx HR specific free CSA list head pointer

The Free CSA List Head Pointer (FCX) register holds the free CSA list head pointer. This always points to an available CSA.

HRA_FCX	Offset address:	$1FE38_H$
CPUx HR specific free CSA list head pointer	Kernel Reset value:	$0000\ 0000_H$
HRB_FCX	Offset address:	$2FE38_H$
CPUx HR specific free CSA list head pointer	Kernel Reset value:	$0000\ 0000_H$
HRHV_FCX	Offset address:	$3FE38_H$
CPUx HR specific free CSA list head pointer	Kernel Reset value:	$0000\ 0000_H$
CPU_FCX	Offset address:	$0FE38_H$
Short address for domain CSFR	Kernel Reset value:	$0000\ 0000_H$
CPU_HVHRA_FCX	Offset address:	$0FE39_H$
Short address for domain CSFR	Kernel Reset value:	$0000\ 0000_H$
CPU_HVHRB_FCX	Offset address:	$0FE3A_H$
Short address for domain CSFR	Kernel Reset value:	$0000\ 0000_H$



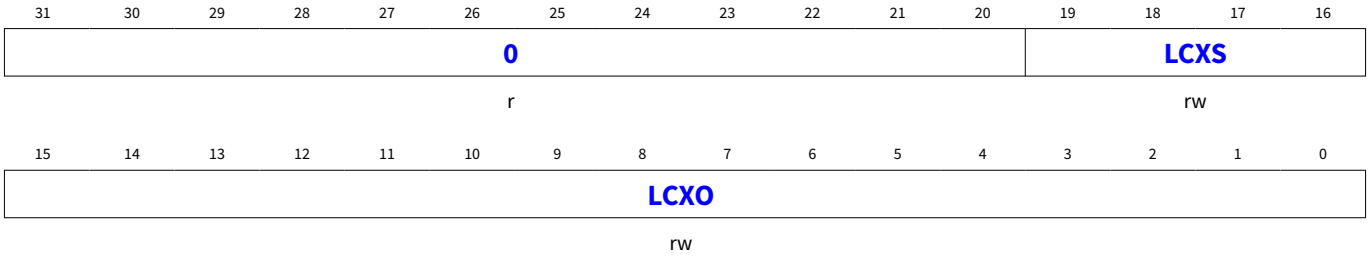
Field	Bits	Type	Description
FCXO	15:0	rwh	**FCX offset address field** The FCXO and FCXS fields together form the FCX pointer, which points to the next available CSA.
FCXS	19:16	rwh	**FCX segment address field** Used in conjunction with the FCXO field.
0	31:20	r	**Reserved - RES** Read as 0; should be written as 0.

4.6.1.49 CPUx HR specific free CSA list limit pointer

The free CSA List Limit Pointer (LCX) register is used to recognize impending free CSA list depletion. If a context save operation occurs and the value of FCX matches LCX then the 'free context depletion' condition is recognized, which triggers an FCD trap immediately after completion of the operation causing the context save; i.e. the return address of the FCD trap is the first instruction of the trap/interrupt/called routine, or the instruction following an SVLCX or BISR instruction.

Note: Please refer to the FCD trap description for details on the use and setting of LCX.

HRA_LCX	Offset address:	$1FE3C_H$
CPUx HR specific free CSA list limit pointer	Kernel Reset value:	$0000\ 0000_H$
HRB_LCX	Offset address:	$2FE3C_H$
CPUx HR specific free CSA list limit pointer	Kernel Reset value:	$0000\ 0000_H$
HRHV_LCX	Offset address:	$3FE3C_H$
CPUx HR specific free CSA list limit pointer	Kernel Reset value:	$0000\ 0000_H$
CPU_LCX	Offset address:	$0FE3C_H$
Short address for domain CSFR	Kernel Reset value:	$0000\ 0000_H$
CPU_HVHRA_LCX	Offset address:	$0FE3D_H$
Short address for domain CSFR	Kernel Reset value:	$0000\ 0000_H$
CPU_HVHRB_LCX	Offset address:	$0FE3E_H$
Short address for domain CSFR	Kernel Reset value:	$0000\ 0000_H$

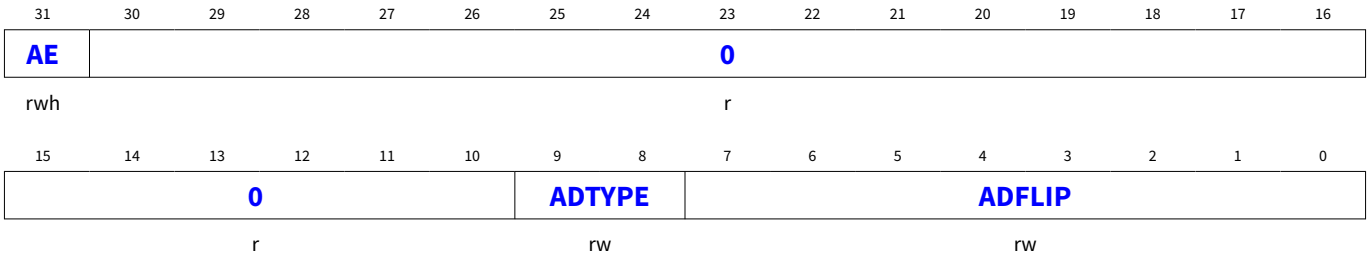


Field	Bits	Type	Description
LCXO	15:0	rw	**LCX offset field** The LCXO and LCXS fields form the pointer LCX, which points to the last available CSA.
LCXS	19:16	rw	**LCX segment address** This field is used in conjunction with the LCXO field.
0	31:20	r	**Reserved - RES** Read as 0; should be written as 0.

4.6.1.50 CPUx SRI error generation register

The SEGEN register controls the injection of SRI errors from the DMI.

HRA_SEGEN	Offset address:	11030_H
CPUx SRI error generation register	Application Reset value:	$0000\ 0000_H$
CPU_SEGEN	Offset address:	01030_H
Short address for domain CSFR	Application Reset value:	$0000\ 0000_H$



Field	Bits	Type	Description
ADFLIP	7:0	rw	**Address ECC bit flip** SRI address ECC bits to be flipped on the next read or write transaction from the DMI when enabled by AE. 00_H No flip 01_H Flip
ADTYPE	9:8	rw	**Type of error** 00_B Data master address phase 01_B Data master write data 10_B Data slave read data 11_B Reserved
AE	31	rwh	**Activate error enable** Enabled the selective inverting of SRI ECC packet bits defined by ADFLIP. This bit will be cleared by hardware after the next SRI read or write transaction from the DMI. 0_B Not enabled 1_B Enabled
0	30:10	r	**Reserved - RES** Read as 0; should be written with 0.

Table 143 Mirrors of HRA_SEGEN

Short name	Long name	Offset address
HRB_SEGEN	CPUx SRI error generation register is an alias.	21030_H
HRHV_SEGEN	CPUx SRI error generation register is an alias.	31030_H

4.6.1.51 CPUx lockstep test register

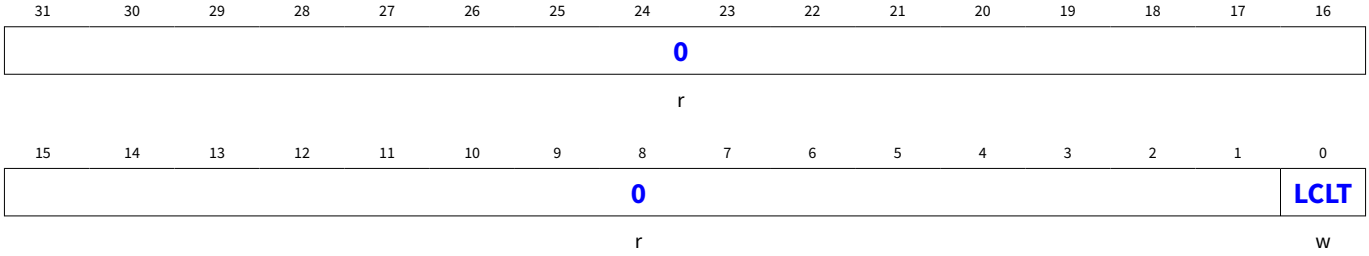
Provides the capability for software to inject a fault condition into the comparators of each Lockstep Comparator Logic block. The implementation should generate a single cycle fault each time the bit is written with '1'.

HRA_LCLTEST Offset address: 11040_H

CPUx lockstep test register Cold PowerOn Reset value: $0000\ 0000_H$

CPU_LCLTEST Offset address: 01040_H

Short address for domain CSFR Cold PowerOn Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
LCLT	0	w	**LCL Lockstep test** Fault injection for LCL. Reads as zero. 0_B No action 1_B Inject single fault in LCL0
0	31:1	r	**Reserved - RES** Read as 0; should be written with 0.

Table 144 Mirrors of HRA_LCLTEST

Short name	Long name	Offset address
HRB_LCLTEST	CPUx lockstep test register is an alias.	21040_H
HRHV_LCLTEST	CPUx lockstep test register is an alias.	31040_H

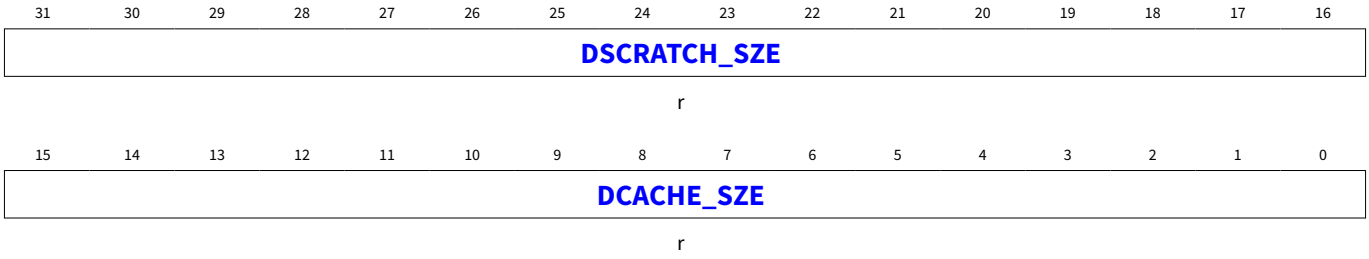
4.6.1.52 CPUx data control register 2

HRA_DCON2 Offset address: 19000_H

CPUx data control register 2 Kernel Reset value: XXXX XXXX$_H$

CPU_DCON2 Offset address: 09000_H

Short address for domain CSFR Kernel Reset value: XXXX XXXX$_H$



Field	Bits	Type	Description
DCACHE_SZE	15:0	r	**Data cache size** In KBytes
DSCRATCH_SZE	31:16	r	**Data scratch size** In KBytes

Table 145 Mirrors of HRA_DCON2

Short name	Long name	Offset address
HRB_DCON2	CPUx data control register 2 is an alias.	29000_H
HRHV_DCON2	CPUx data control register 2 is an alias.	39000_H

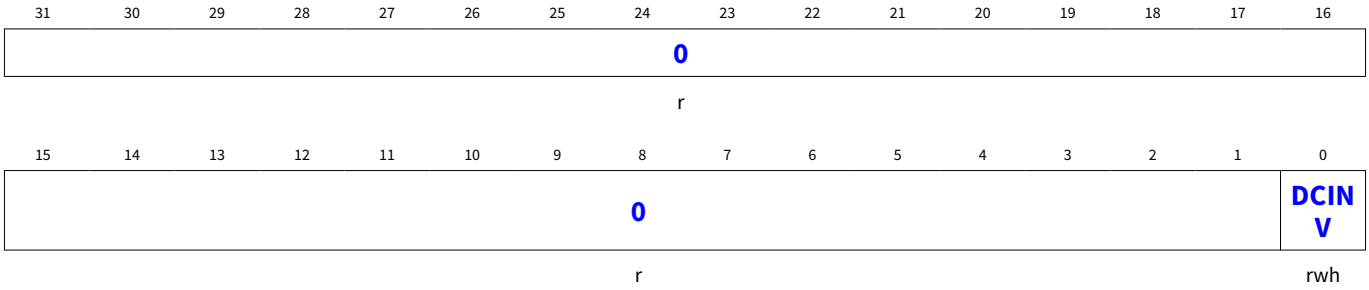
4.6.1.53 CPUx data control register 1

HRA_DCON1 Offset address: 19008_H

CPUx data control register 1 Kernel Reset value: $0000\ 0000_H$

CPU_DCON1 Offset address: 09008_H

Short address for domain CSFR Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
DCINV	0	rwh	**Data cache invalidate** 0_B Write: No effect, normal data cache operation. Read : Normal operation, data cache available 1_B Write : Initiate invalidation of all clean entries in the data cache. Read: Data cache invalidation in progress. Data cache unavailable.
0	31:1	r	**Reserved - RES** Read as 0; should be written with 0.

Table 146 Mirrors of HRA_DCON1

Short name	Long name	Offset address
HRB_DCON1	CPUx data control register 1 is an alias.	29008_H
HRHV_DCON1	CPUx data control register 1 is an alias.	39008_H

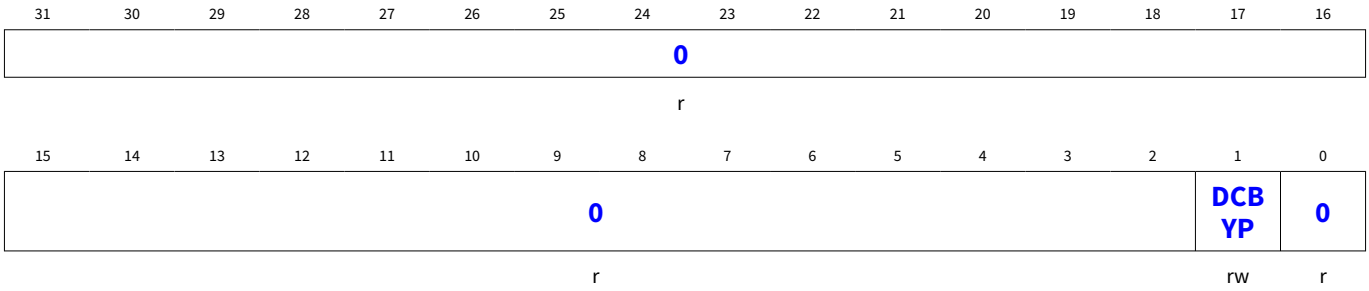
4.6.1.54 CPUx data memory control register

HRA_DCON0 Offset address: 19040_H

CPUx data memory control register Kernel Reset value: $0000\ 0002_H$

CPU_DCON0 Offset address: 09040_H

Short address for domain CSFR Kernel Reset value: $0000\ 0002_H$



Field	Bits	Type	Description
DCBYP	1	rw	**Data cache bypass** 0_B DCache / Data read buffer enabled 1_B DCache / Data read buffer bypass (disabled)
0	0, 31:2	r	**Reserved - RES** Read as 0; should be written with 0.

Table 147 Mirrors of HRA_DCON0

Short name	Long name	Offset address
HRB_DCON0	CPUx data memory control register is an alias.	29040_H
HRHV_DCON0	CPUx data memory control register is an alias.	39040_H

4.6.1.55 CPUx HR specific data synchronous trap register

The DSTR contains synchronous trap information for the data memory system. The register is updated with trap source information to aid the localisation of faults.

The register is updated whenever a valid trap is detected and the register has no bits already set. It is cleared by a write (independent of data value).

HRA_DSTR	Offset address:	19010_H
CPUx HR specific data synchronous trap register	Kernel Reset value:	$0000\ 0000_H$
HRB_DSTR	Offset address:	29010_H
CPUx HR specific data synchronous trap register	Kernel Reset value:	$0000\ 0000_H$
HRHV_DSTR	Offset address:	39010_H
CPUx HR specific data synchronous trap register	Kernel Reset value:	$0000\ 0000_H$
CPU_DSTR	Offset address:	09010_H
Short address for domain CSFR	Kernel Reset value:	$0000\ 0000_H$
CPU_HVHRA_DSTR	Offset address:	09011_H
Short address for domain CSFR	Kernel Reset value:	$0000\ 0000_H$
CPU_HVHRB_DSTR	Offset address:	09012_H
Short address for domain CSFR	Kernel Reset value:	$0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0							ALN	0			CLE	MPE	CAC	SCE	SDE
r							rwh	r			rwh	rwh	rwh	rwh	rwh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LOE	DTME	0			L2_ASYNC	L2_CSA	L2_MPE	0	CRE	0		DRE	LBE	GAE	SRE
rwh	rwh	r			rwh	rwh	rwh	r	rwh	r		rwh	rwh	rwh	rwh

Field	Bits	Type	Description
SRE	0	rwh	**Scratch range error** A scratch range error occurs whenever an access to the data scratch is outside the range of the SRAM.
GAE	1	rwh	**Global address error** Load or store to local code scratch address outside of the lower 1MByte.
LBE	2	rwh	**Load bus error** A Load bus error will be set whenever the SRI flags an error due a load from external memory.
DRE	3	rwh	**Local DLMU range error** A DLMU range error occurs whenever an access to the local DLMU region is outside the physically implemented memory.
CRE	6	rwh	**Cache refill error** A Cache refill error will be set whenever the SRI flags an error due a cache refill from external memory.
L2_MPE	8	rwh	**Level 2 memory protection error** Data access violating level 2 (hypervisor) memory protection.
L2_CSA	9	rwh	**Level 2 CSA memory protection error** Data access violating level 2 (hypervisor) memory protection caused by a CSA operation.
L2_ASYNC	10	rwh	**Level 2 async CSA protection error** Data access violating level 2 (hypervisor) memory protection. During the entry sequence of an asynchrnonous event (trap/interrupt) taken in a VM, the CSA operation triggered a L2 MPE.
DTME	14	rwh	**DTAG MSIST error** Access to memory mapped DTAG range outside of physically implemented memory.
LOE	15	rwh	**Load overlay error** Load to invalid overlay address.
SDE	16	rwh	**Segment difference error** Load or store access where base address is in different segment to access address.
SCE	17	rwh	**Segment crossing error** Load or store access across segment boundary.

(table continues...)

(continued)

Field	Bits	Type	Description
CAC	18	rwh	**CSFR access error** Load or store to local CSFR space.
MPE	19	rwh	**Memory protection error** Data access violating memory protection.
CLE	20	rwh	**Context location error** Context operation to invalid location.
ALN	24	rwh	**Alignment error** Data access causing alignment error.
0	5:4, 7, 13:11, 23:21, 31:25	r	**Reserved - RES** Read as 0; should be written with 0.

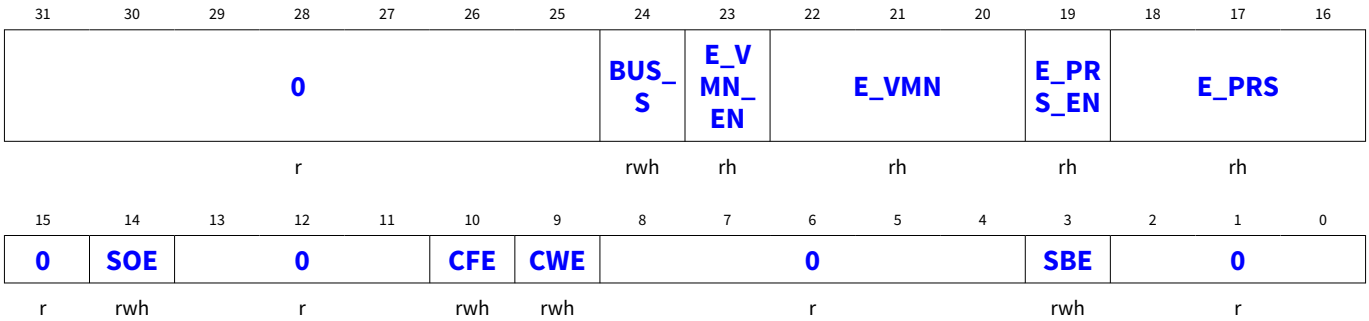
4.6.1.56 CPUx HR specific data asynchronous trap register

The DATR contains asynchronous trap information for the data memory system. The register is updated with trap information for DAE traps to aid the localisation of faults.

The register is updated whenever a valid trap is detected and the register has no bits already set. It is cleared by a write (independent of data value).

DAE traps are inhibited if the DATR register is non-zero.

HRA_DATR	Offset address:	19018_H
CPUx HR specific data asynchronous trap register	Kernel Reset value:	$0000\ 0000_H$
HRB_DATR	Offset address:	29018_H
CPUx HR specific data asynchronous trap register	Kernel Reset value:	$0000\ 0000_H$
HRHV_DATR	Offset address:	39018_H
CPUx HR specific data asynchronous trap register	Kernel Reset value:	$0000\ 0000_H$
CPU_DATR	Offset address:	09018_H
Short address for domain CSFR	Kernel Reset value:	$0000\ 0000_H$
CPU_HVHRA_DATR	Offset address:	09019_H
Short address for domain CSFR	Kernel Reset value:	$0000\ 0000_H$
CPU_HVHRB_DATR	Offset address:	$0901A_H$
Short address for domain CSFR	Kernel Reset value:	$0000\ 0000_H$



Field	Bits	Type	Description
SBE	3	rwh	**Store bus error**
CWE	9	rwh	**Cache writeback error**
CFE	10	rwh	**Cache flush error**
SOE	14	rwh	**Store overlay error**
E_PRS	18:16	rh	**Captured PRS** Only valid if PRS_EN is set.
E_PRS_EN	19	rh	**Captured PRS_EN**
E_VMN	22:20	rh	**Captured VMN** Only valid if VMN_EN is set.
E_VMN_EN	23	rh	**Captured VMN_EN**
BUS_S	24	rwh	**Captured alternate identifier selector bit** Value of the alternate identifier selector bit used to initiate the bus transaction. Either PSW.S or Cache.S.
0	2:0, 8:4, 13:11, 15, 31:25	r	**Reserved - RES** Read as 0; should be written with 0.

4.6.1.57 CPUx HR specific data error address register

DEADD contains trap address information for the Data memory system. The register is updated with trap information for MEM, ALN, DSE or DAE traps to aid the localisation of faults.

The register is only set whenever a trap is detected and either the DATR or DSTR registers have no bits already set.

The register contents are only valid when either the DATR or DSTR register is non-zero and hence should be read prior to clearing these registers.

HRA_DEADD	Offset address:	$1901C_H$
CPUx HR specific data error address register	Kernel Reset value:	$0000\ 0000_H$
HRB_DEADD	Offset address:	$2901C_H$
CPUx HR specific data error address register	Kernel Reset value:	$0000\ 0000_H$
HRHV_DEADD	Offset address:	$3901C_H$
CPUx HR specific data error address register	Kernel Reset value:	$0000\ 0000_H$
CPU_DEADD	Offset address:	$0901C_H$
Short address for domain CSFR	Kernel Reset value:	$0000\ 0000_H$
CPU_HVHRA_DEADD	Offset address:	$0901D_H$
Short address for domain CSFR	Kernel Reset value:	$0000\ 0000_H$
CPU_HVHRB_DEADD	Offset address:	$0901E_H$
Short address for domain CSFR	Kernel Reset value:	$0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ERROR_ADDRESS															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ERROR_ADDRESS															
rh															

Field	Bits	Type	Description
ERROR_ADDRESS	31:0	rh	**Error address**

4.6.1.58 CPUx HR specific trap control register

HRA_FPU_TRAP_CON Offset address: $1A000_H$
CPUx HR specific trap control register Kernel Reset value: $0000\ 0000_H$
HRB_FPU_TRAP_CON Offset address: $2A000_H$
CPUx HR specific trap control register Kernel Reset value: $0000\ 0000_H$
HRHV_FPU_TRAP_CON Offset address: $3A000_H$
CPUx HR specific trap control register Kernel Reset value: $0000\ 0000_H$
CPU_FPU_TRAP_CON Offset address: $0A000_H$
Short address for domain CSFR Kernel Reset value: $0000\ 0000_H$
CPU_HVHRA_FPU_TRAP_CON Offset address: $0A001_H$
Short address for domain CSFR Kernel Reset value: $0000\ 0000_H$
CPU_HVHRB_FPU_TRAP_CON Offset address: $0A002_H$
Short address for domain CSFR Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		FV	0	FU	FX	0				FVE	0	FUE	FXE	0	
r		rh	r	rh	rh	r				rw	r	rw	rw	r	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0						RM		0						TCL	TST
r						rh		r						w	rh

Field	Bits	Type	Description
TST	0	rh	**Trap Status** 0_B No instruction captured The next enabled exception will cause the exceptional instruction to be captured. 1_B Instruction captured No further enabled exceptions will be captured until TST is cleared.

(table continues...)

(continued)

Field	Bits	Type	Description
TCL	1	w	**Trap clear** Read: always reads as 0. 0_B No effect 1_B Clears the trapped instruction (TST will be negated)
RM	9:8	rh	**Captured rounding mode** The rounding mode of the captured instruction. Only valid when TST is asserted. Note that this is the rounding mode supplied to the FPU for the exceptional instruction. UPDFL instructions may cause a trap and change the rounding mode. In this case the RM bits capture the input rounding mode
FXE	18	rw	**FX trap enable** When set, an instruction generating an FX exception will trigger a trap.
FUE	19	rw	**FU trap enable** When set, an instruction generating an FU exception will trigger a trap.
FVE	21	rw	**FV trap enable** When set, an instruction generating an FV exception will trigger a trap.
FX	26	rh	**Captured FX** Asserted if the captured instruction asserted FX. Only valid when TST is asserted.
FU	27	rh	**Captured FU** Asserted if the captured instruction asserted FU. Only valid when TST is asserted.
FV	29	rh	**Captured FV** Asserted if the captured instruction asserted FV. Only valid when TST is asserted
0	7:2, 17:10, 20, 25:22, 28, 31:30	r	**Reserved - RES** **Read as 0; should be written with 0.**

4.6.1.59 CPUx HR specific trapping instruction program counter register

HRA_FPU_TRAP_PC — Offset address: $1A004_H$

CPUx HR specific trapping instruction program counter register — Kernel Reset value: $0000\ 0000_H$

HRB_FPU_TRAP_PC — Offset address: $2A004_H$

CPUx HR specific trapping instruction program counter register — Kernel Reset value: $0000\ 0000_H$

HRHV_FPU_TRAP_PC — Offset address: $3A004_H$

CPUx HR specific trapping instruction program counter register — Kernel Reset value: $0000\ 0000_H$

CPU_FPU_TRAP_PC — Offset address: $0A004_H$

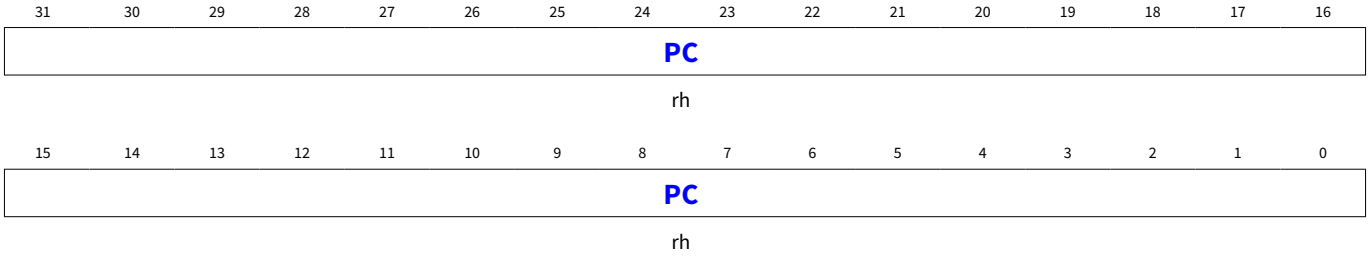
Short address for domain CSFR — Kernel Reset value: $0000\ 0000_H$

CPU_HVHRA_FPU_TRAP_PC — Offset address: $0A005_H$

Short address for domain CSFR — Kernel Reset value: $0000\ 0000_H$

CPU_HVHRB_FPU_TRAP_PC — Offset address: $0A006_H$

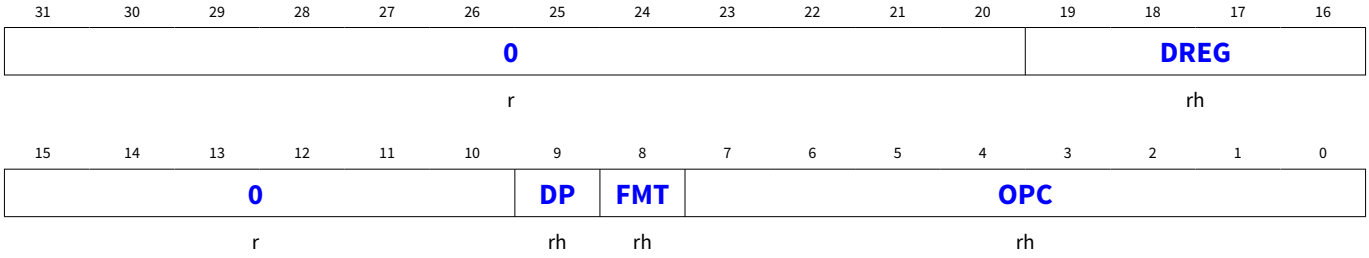
Short address for domain CSFR — Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
PC	31:0	rh	**Captured program counter** The program counter of the captured instruction. Only valid when FPU_TRAP_CON.TST is asserted.

4.6.1.60 CPUx HR specific trapping instruction opcode register

HRA_FPU_TRAP_OPC	Offset address:	$1A008_H$
CPUx HR specific trapping instruction opcode register	Kernel Reset value:	$0000\ 0000_H$
HRB_FPU_TRAP_OPC	Offset address:	$2A008_H$
CPUx HR specific trapping instruction opcode register	Kernel Reset value:	$0000\ 0000_H$
HRHV_FPU_TRAP_OPC	Offset address:	$3A008_H$
CPUx HR specific trapping instruction opcode register	Kernel Reset value:	$0000\ 0000_H$
CPU_FPU_TRAP_OPC	Offset address:	$0A008_H$
Short address for domain CSFR	Kernel Reset value:	$0000\ 0000_H$
CPU_HVHRA_FPU_TRAP_OPC	Offset address:	$0A009_H$
Short address for domain CSFR	Kernel Reset value:	$0000\ 0000_H$
CPU_HVHRB_FPU_TRAP_OPC	Offset address:	$0A00A_H$
Short address for domain CSFR	Kernel Reset value:	$0000\ 0000_H$



Field	Bits	Type	Description
OPC	7:0	rh	**Captured opcode** The secondary opcode of the captured instruction. When FPU_TRAP_OPC.FMT=0 only bits [3:0] are defined. OPC is valid only when FPU_TRAP_CON.TST is asserted.
FMT	8	rh	**Captured instruction format** The format of the captured instruction's opcode. Only valid when FPU_TRAP_CON.TST is asserted. 0_B RRR format 1_B RR format
DP	9	rh	**Captured instruction format - FMT** Single/Double precision identifier. Only valid when FPU_TRAP_CON.TST is asserted. 0_B Single Precision (including HPTOF and FTOHP) 1_B Double precision
DREG	19:16	rh	**Captured destination register** The destination register of the captured instruction. Only valid when FPU_TRAP_CON.TST is asserted. 0_H Data general purpose register 0 F_H Data general purpose register 15

(table continues...)

(continued)

Field	Bits	Type	Description
0	15:10, 31:20	r	**Reserved - RES** **Read as 0; should be written with 0.**

4.6.1.61 CPUx HR specific trapping instruction operand register

HRA_FPU_TRAP_SRC1_L Offset address: $1A010_H$

CPUx HR specific trapping instruction operand register Kernel Reset value: $0000\ 0000_H$

HRB_FPU_TRAP_SRC1_L Offset address: $2A010_H$

CPUx HR specific trapping instruction operand register Kernel Reset value: $0000\ 0000_H$

HRHV_FPU_TRAP_SRC1_L Offset address: $3A010_H$

CPUx HR specific trapping instruction operand register Kernel Reset value: $0000\ 0000_H$

CPU_FPU_TRAP_SRC1_L Offset address: $0A010_H$

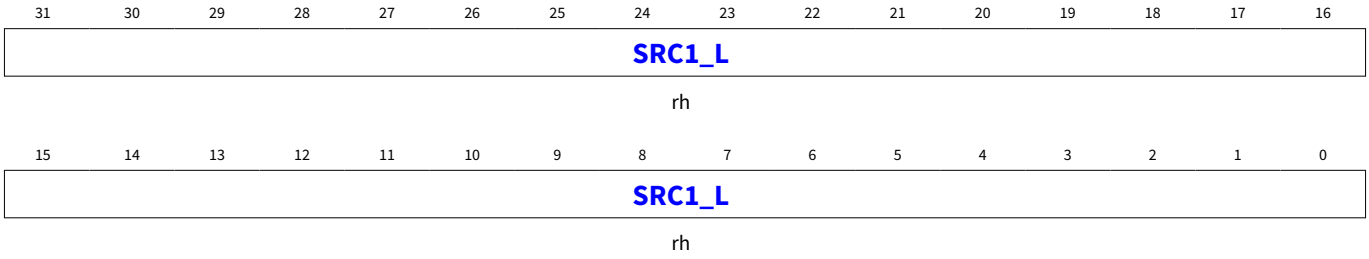
Short address for domain CSFR Kernel Reset value: $0000\ 0000_H$

CPU_HVHRA_FPU_TRAP_SRC1_L Offset address: $0A011_H$

Short address for domain CSFR Kernel Reset value: $0000\ 0000_H$

CPU_HVHRB_FPU_TRAP_SRC1_L Offset address: $0A012_H$

Short address for domain CSFR Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
SRC1_L	31:0	rh	**Captured SRC1 operand** The SRC1[31:0] operand of the captured instruction. Only valid when FPU_TRAP_CON.TST is asserted.

4.6.1.62 CPUx HR specific trapping instruction operand register

HRA_FPU_TRAP_SRC1_U Offset address: $1A014_H$

CPUx HR specific trapping instruction operand register Kernel Reset value: $0000\ 0000_H$

HRB_FPU_TRAP_SRC1_U Offset address: $2A014_H$

CPUx HR specific trapping instruction operand register Kernel Reset value: $0000\ 0000_H$

HRHV_FPU_TRAP_SRC1_U Offset address: $3A014_H$

CPUx HR specific trapping instruction operand register Kernel Reset value: $0000\ 0000_H$

CPU_FPU_TRAP_SRC1_U Offset address: $0A014_H$

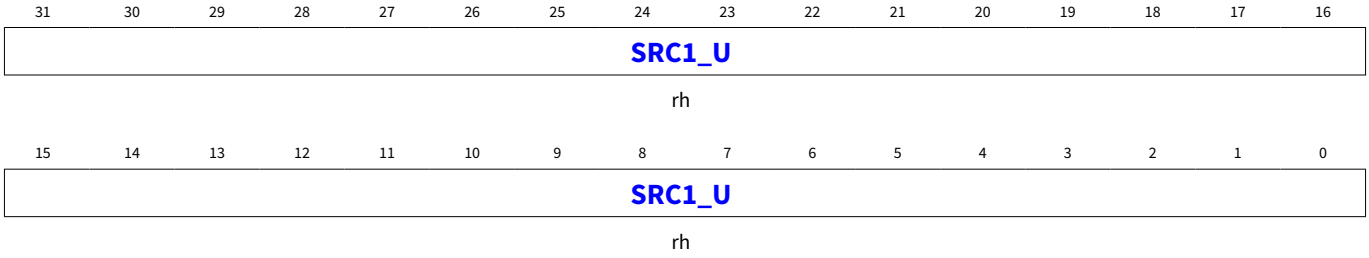
Short address for domain CSFR Kernel Reset value: $0000\ 0000_H$

CPU_HVHRA_FPU_TRAP_SRC1_U Offset address: $0A015_H$

Short address for domain CSFR Kernel Reset value: $0000\ 0000_H$

CPU_HVHRB_FPU_TRAP_SRC1_U Offset address: $0A016_H$

Short address for domain CSFR Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
SRC1_U	31:0	rh	**Captured SRC1 operand** The SRC1[63:32] operand of the captured instruction. For single precision SRC1, this register will be set to 0. Only valid when FPU_TRAP_CON.TST is asserted.

4.6.1.63 CPUx HR specific trapping instruction operand register

HRA_FPU_TRAP_SRC2_L Offset address: $1A018_H$

CPUx HR specific trapping instruction operand register Kernel Reset value: $0000\ 0000_H$

HRB_FPU_TRAP_SRC2_L Offset address: $2A018_H$

CPUx HR specific trapping instruction operand register Kernel Reset value: $0000\ 0000_H$

HRHV_FPU_TRAP_SRC2_L Offset address: $3A018_H$

CPUx HR specific trapping instruction operand register Kernel Reset value: $0000\ 0000_H$

CPU_FPU_TRAP_SRC2_L Offset address: $0A018_H$

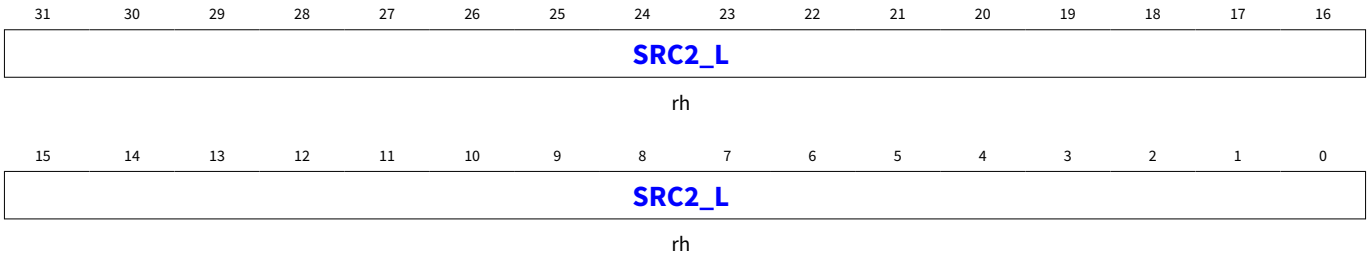
Short address for domain CSFR Kernel Reset value: $0000\ 0000_H$

CPU_HVHRA_FPU_TRAP_SRC2_L Offset address: $0A019_H$

Short address for domain CSFR Kernel Reset value: $0000\ 0000_H$

CPU_HVHRB_FPU_TRAP_SRC2_L Offset address: $0A01A_H$

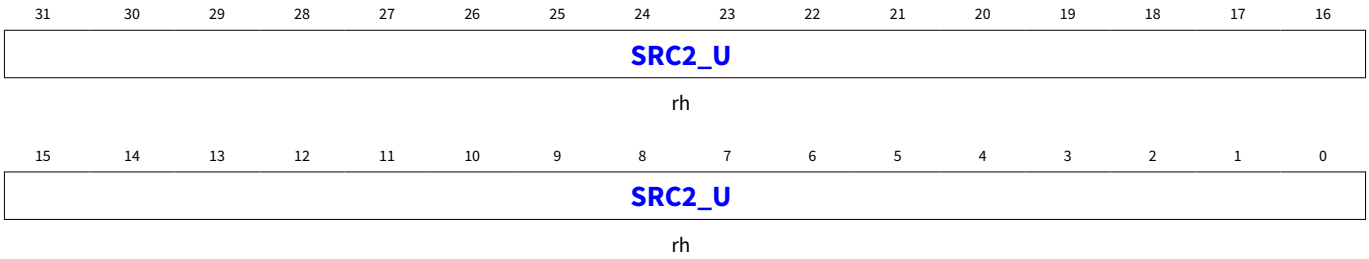
Short address for domain CSFR Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
SRC2_L	31:0	rh	**Captured SRC2 operand** The SRC2[31:0] operand of the captured instruction. Only valid when FPU_TRAP_CON.TST is asserted.

4.6.1.64 CPUx HR specific trapping instruction operand register

HRA_FPU_TRAP_SRC2_U	Offset address:	$1A01C_H$
CPUx HR specific trapping instruction operand register	Kernel Reset value:	$0000\ 0000_H$
HRB_FPU_TRAP_SRC2_U	Offset address:	$2A01C_H$
CPUx HR specific trapping instruction operand register	Kernel Reset value:	$0000\ 0000_H$
HRHV_FPU_TRAP_SRC2_U	Offset address:	$3A01C_H$
CPUx HR specific trapping instruction operand register	Kernel Reset value:	$0000\ 0000_H$
CPU_FPU_TRAP_SRC2_U	Offset address:	$0A01C_H$
Short address for domain CSFR	Kernel Reset value:	$0000\ 0000_H$
CPU_HVHRA_FPU_TRAP_SRC2_U	Offset address:	$0A01D_H$
Short address for domain CSFR	Kernel Reset value:	$0000\ 0000_H$
CPU_HVHRB_FPU_TRAP_SRC2_U	Offset address:	$0A01E_H$
Short address for domain CSFR	Kernel Reset value:	$0000\ 0000_H$



Field	Bits	Type	Description
SRC2_U	31:0	rh	**Captured SRC2 operand** The SRC2[63:32] operand of the captured instruction. For single precision SRC2, this register will be set to 0. Only valid when FPU_TRAP_CON.TST is asserted.

4.6.1.65 CPUx HR specific trapping instruction operand register

HRA_FPU_TRAP_SRC3_L Offset address: $1A020_H$

CPUx HR specific trapping instruction operand register Kernel Reset value: $0000\ 0000_H$

HRB_FPU_TRAP_SRC3_L Offset address: $2A020_H$

CPUx HR specific trapping instruction operand register Kernel Reset value: $0000\ 0000_H$

HRHV_FPU_TRAP_SRC3_L Offset address: $3A020_H$

CPUx HR specific trapping instruction operand register Kernel Reset value: $0000\ 0000_H$

CPU_FPU_TRAP_SRC3_L Offset address: $0A020_H$

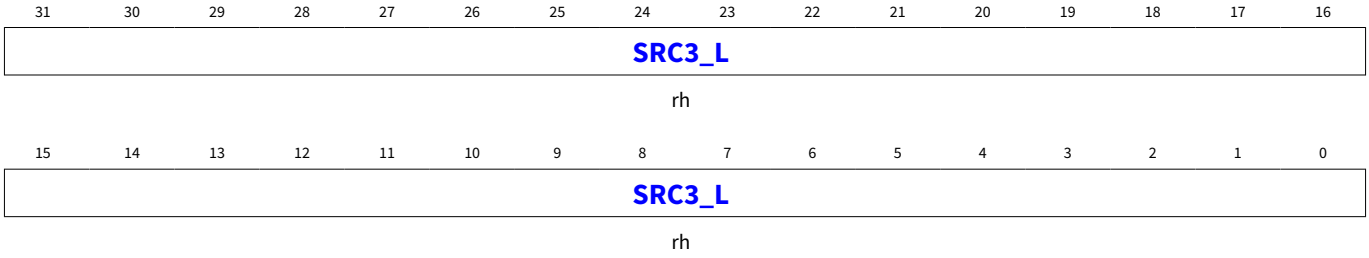
Short address for domain CSFR Kernel Reset value: $0000\ 0000_H$

CPU_HVHRA_FPU_TRAP_SRC3_L Offset address: $0A021_H$

Short address for domain CSFR Kernel Reset value: $0000\ 0000_H$

CPU_HVHRB_FPU_TRAP_SRC3_L Offset address: $0A022_H$

Short address for domain CSFR Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
SRC3_L	31:0	rh	**Captured SRC3 operand** The SRC3[31:0] operand of the captured instruction. Only valid when FPU_TRAP_CON.TST is asserted.

4.6.1.66 CPUx HR specific trapping instruction operand register

HRA_FPU_TRAP_SRC3_U Offset address: $1A024_H$

CPUx HR specific trapping instruction operand register Kernel Reset value: $0000\ 0000_H$

HRB_FPU_TRAP_SRC3_U Offset address: $2A024_H$

CPUx HR specific trapping instruction operand register Kernel Reset value: $0000\ 0000_H$

HRHV_FPU_TRAP_SRC3_U Offset address: $3A024_H$

CPUx HR specific trapping instruction operand register Kernel Reset value: $0000\ 0000_H$

CPU_FPU_TRAP_SRC3_U Offset address: $0A024_H$

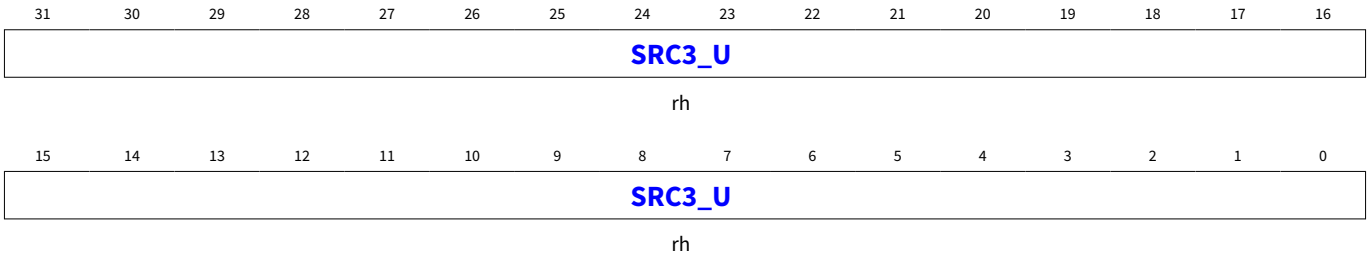
Short address for domain CSFR Kernel Reset value: $0000\ 0000_H$

CPU_HVHRA_FPU_TRAP_SRC3_U Offset address: $0A025_H$

Short address for domain CSFR Kernel Reset value: $0000\ 0000_H$

CPU_HVHRB_FPU_TRAP_SRC3_U Offset address: $0A026_H$

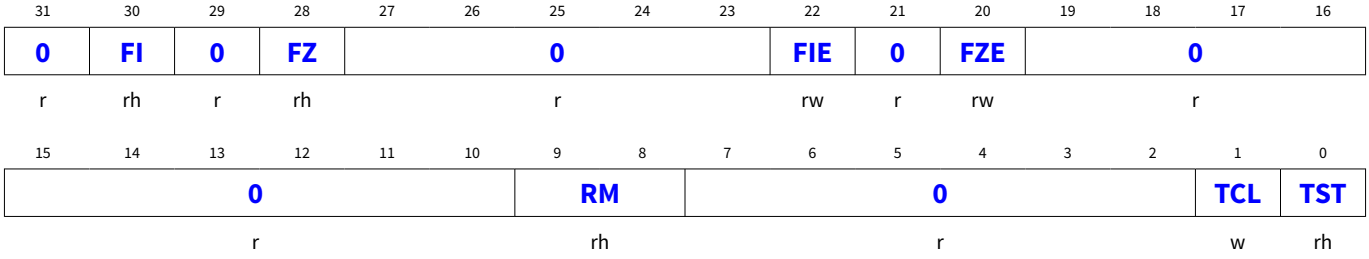
Short address for domain CSFR Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
SRC3_U	31:0	rh	**Captured SRC3 operand** The SRC3[63:32] operand of the captured instruction. For single precision SRC3, this register will be set to 0. Only valid when FPU_TRAP_CON.TST is asserted.

4.6.1.67 CPUx HR specific synchronous trap control register

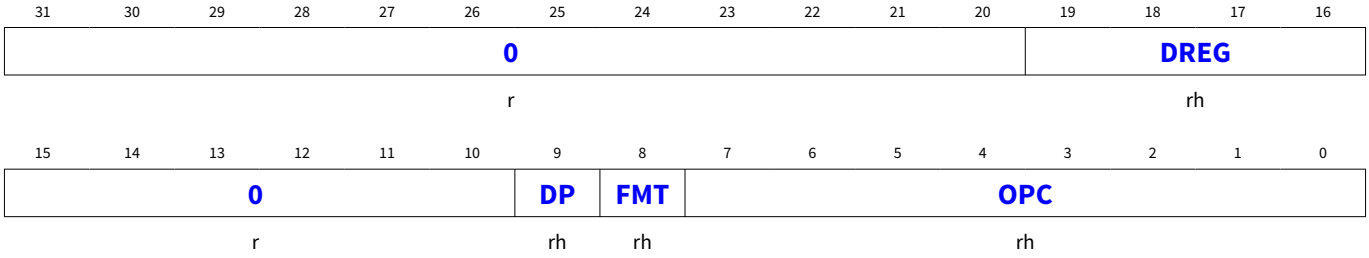
HRA_FPU_SYNC_TRAP_CON	Offset address:	$1A030_H$
CPUx HR specific synchronous trap control register	Kernel Reset value:	$0000\ 0000_H$
HRB_FPU_SYNC_TRAP_CON	Offset address:	$2A030_H$
CPUx HR specific synchronous trap control register	Kernel Reset value:	$0000\ 0000_H$
HRHV_FPU_SYNC_TRAP_CON	Offset address:	$3A030_H$
CPUx HR specific synchronous trap control register	Kernel Reset value:	$0000\ 0000_H$
CPU_FPU_SYNC_TRAP_CON	Offset address:	$0A030_H$
Short address for domain CSFR	Kernel Reset value:	$0000\ 0000_H$
CPU_HVHRA_FPU_SYNC_TRAP_CON	Offset address:	$0A031_H$
Short address for domain CSFR	Kernel Reset value:	$0000\ 0000_H$
CPU_HVHRB_FPU_SYNC_TRAP_CON	Offset address:	$0A032_H$
Short address for domain CSFR	Kernel Reset value:	$0000\ 0000_H$



Field	Bits	Type	Description
TST	0	rh	**Trap status** 0_B No instruction captured The next enabled synchronous exception will cause the exceptional instruction to be captured. 1_B Instruction captured No further enabled synchronous exceptions will be captured until TST is cleared.
TCL	1	w	**Trap clear** Read: always reads as 0. 0_B No effect 1_B Clears the trapped instruction (TST will be negated)
RM	9:8	rh	**Captured rounding mode** The rounding mode of the captured instruction. Only valid when TST is asserted. Note that this is the rounding mode supplied to the FPU for the exceptional instruction. UPDFL instructions may cause a trap and change the rounding mode. In this case the RM bits capture the input rounding mode
FZE	20	rw	**FZ trap enable** When set, an instruction generating an FZ exception will trigger a trap.
FIE	22	rw	**FI trap enable** When set, an instruction generating an FI exception will trigger a trap.
FZ	28	rh	**Captured FZ** Asserted if the captured instruction asserted FZ. Only valid when TST is asserted
FI	30	rh	**Captured FI** Asserted if the captured instruction asserted FI. Only valid when TST is asserted
0	7:2, 19:10, 21, 27:23, 29, 31	r	**Reserved - RES** **Read as 0; should be written with 0.**

4.6.1.68 CPUx HR specific trapping instruction opcode register

HRA_FPU_SYNC_TRAP_OPC	Offset address:	$1A034_H$
CPUx HR specific trapping instruction opcode register	Kernel Reset value:	$0000\ 0000_H$
HRB_FPU_SYNC_TRAP_OPC	Offset address:	$2A034_H$
CPUx HR specific trapping instruction opcode register	Kernel Reset value:	$0000\ 0000_H$
HRHV_FPU_SYNC_TRAP_OPC	Offset address:	$3A034_H$
CPUx HR specific trapping instruction opcode register	Kernel Reset value:	$0000\ 0000_H$
CPU_FPU_SYNC_TRAP_OPC	Offset address:	$0A034_H$
Short address for domain CSFR	Kernel Reset value:	$0000\ 0000_H$
CPU_HVHRA_FPU_SYNC_TRAP_OPC	Offset address:	$0A035_H$
Short address for domain CSFR	Kernel Reset value:	$0000\ 0000_H$
CPU_HVHRB_FPU_SYNC_TRAP_OPC	Offset address:	$0A036_H$
Short address for domain CSFR	Kernel Reset value:	$0000\ 0000_H$



Field	Bits	Type	Description
OPC	7:0	rh	**Captured opcode** The secondary opcode of the captured instruction. When FPU_TRAP_OPC.FMT=0 only bits [3:0] are defined. OPC is valid only when FPU_TRAP_CON.TST is asserted.
FMT	8	rh	**Captured instruction format** The format of the captured instruction's opcode. Only valid when FPU_TRAP_CON.TST is asserted. 0_B RRR format 1_B RR format
DP	9	rh	**Captured instruction format - FMT** Single/Double precision identifier. Only valid when FPU_TRAP_CON.TST is asserted. 0_B Single Precision (including HPTOF and FTOHP) 1_B Double precision
DREG	19:16	rh	**Captured destination register** The destination register of the captured instruction. Only valid when FPU_TRAP_CON.TST is asserted. 0_H Data general purpose register 0 F_H Data general purpose register 15

(table continues...)

(continued)

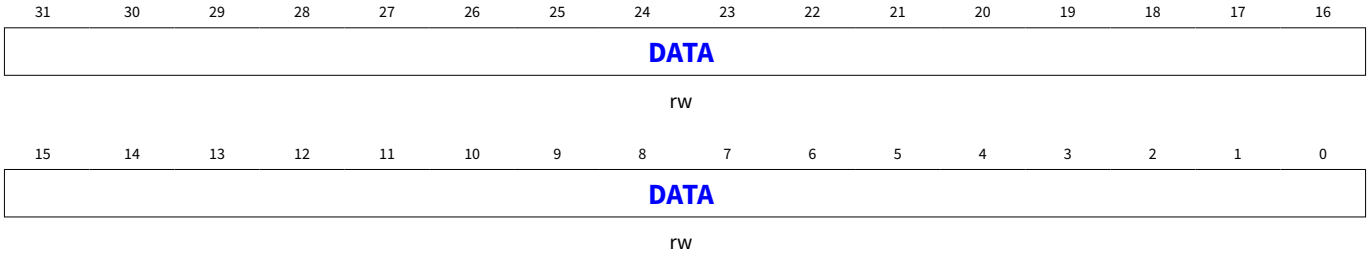
Field	Bits	Type	Description
0	15:10, 31:20	r	**Reserved - RES** **Read as 0; should be written with 0.**

4.6.1.69 CPUx data general purpose register y

General Purpose Register

Dy (y=0-15) Offset address: $1FF00_H$+y*4

CPUx data general purpose register y Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
DATA	31:0	rw	**Data register** General purpose registers

4.6.1.70 CPUx address general purpose register y

General Purpose Register

Ay (y=0-15) Offset address: $1FF80_H$+y*4

CPUx address general purpose register y Kernel Reset value: $0000\ 0000_H$

HRB_Ay (y=0-1) Offset address: $2FF80_H$+y*4

CPUx HR specific address general purpose register y Kernel Reset value: $0000\ 0000_H$

HRB_Ay (y=8-9) Offset address: $2FFA0_H$+(y-8)*4

CPUx HR specific address general purpose register y Kernel Reset value: $0000\ 0000_H$

HRHV_Ay (y=0-1) Offset address: $3FF80_H$+y*4

CPUx HR specific address general purpose register y Kernel Reset value: $0000\ 0000_H$

HRHV_Ay (y=8-9) Offset address: $3FFA0_H$+(y-8)*4

CPUx HR specific address general purpose register y Kernel Reset value: $0000\ 0000_H$

CPU_HVHRA_Ay (y=0-1) Offset address: $0FF81_H$+y*4

Short address for domain CSFR Kernel Reset value: $0000\ 0000_H$

CPU_HVHRA_Ay (y=8-9) Offset address: $0FFA1_H$+(y-8)*4

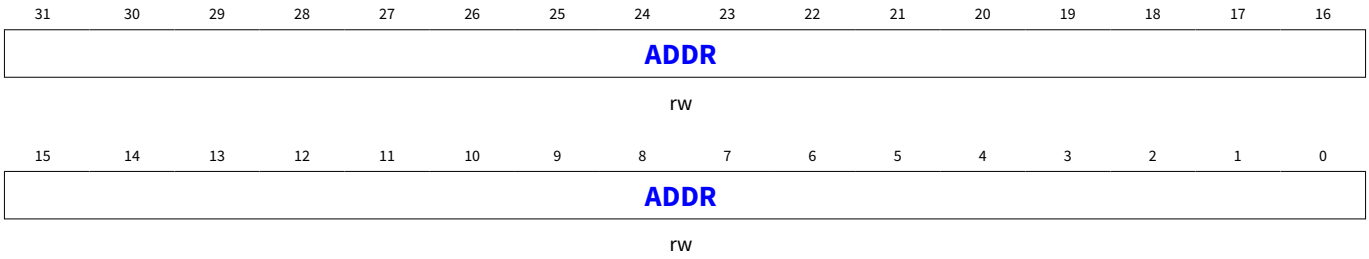
Short address for domain CSFR Kernel Reset value: $0000\ 0000_H$

CPU_HVHRB_Ay (y=0-1) Offset address: $0FF82_H$+y*4

Short address for domain CSFR Kernel Reset value: $0000\ 0000_H$

CPU_HVHRB_Ay (y=8-9) Offset address: $0FFA2_H$+(y-8)*4

Short address for domain CSFR Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
ADDR	31:0	rw	**Address register** General purpose registers

4.6.1.71 CPUx virtualization control register 0

The virtualization extension can be enabled or disabled by writing the bit VCON0.VEN (bit0). This bit is "write once" in that the value is locked following a write until the next reset, subsequent writes will be ignored and have no effect.

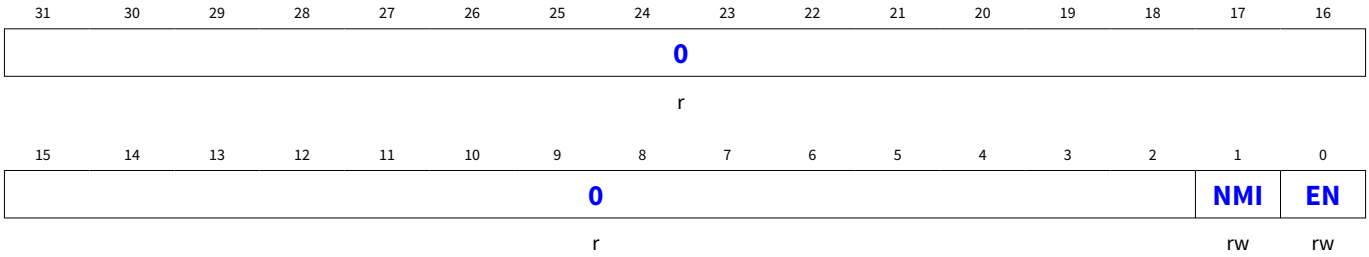
If VCON0.VEN==0 then HVCALL and all other methods of switching to the hypervisor are disabled.

HRHV_VCON0 — Offset address: $3B000_H$

CPUx virtualization control register 0 — Kernel Reset value: $0000\ 0001_H$

CPU_VCON0 — Offset address: $0B000_H$

Short address for domain CSFR — Kernel Reset value: $0000\ 0001_H$



Field	Bits	Type	Description
EN	0	rw	**Virtualization Enable** 0_B Virtualization disabled 1_B Virtualization enabled
NMI	1	rw	**NMI handling** 0_B NMI handled by the currently running VM 1_B NMI handled by the hypervisor
0	31:2	r	**Reserved - RES** Read as 0; should be written with 0.

4.6.1.72 CPUx virtualization control register 1

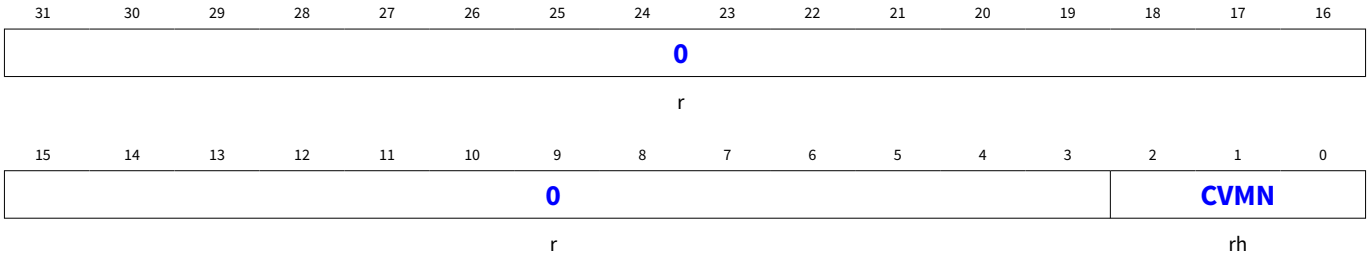
VCON1 contains the number of the currently running virtual machine. This status register is updated from VCON2.VMN on execution of the Return from Hypervisor (RFH) instruction or by a switch to the Hypervisor from some other virtual machine.

HRHV_VCON1 Offset address: $3B004_H$

CPUx virtualization control register 1 Kernel Reset value: $0000\ 0000_H$

CPU_VCON1 Offset address: $0B004_H$

Short address for domain CSFR Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
CVMN	2:0	rh	**Currently running virtual machine number**
0	31:3	r	**Reserved - RES** Read as 0; should be written with 0.

4.6.1.73 CPUx virtualization control register 2

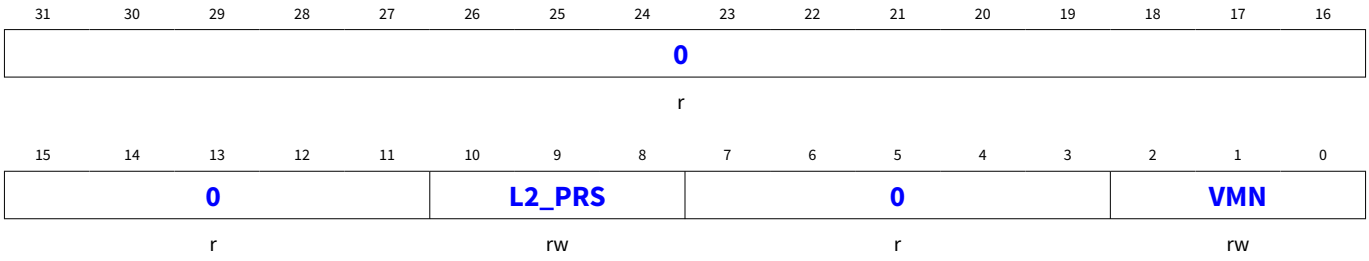
VCON2 is used by the Return from Hypervisor instruction (RFH) to change the execution from VM0 (hypervisor) to a target virtual machine selected by the hypervisor. It provides both the target virtual machine number and the protection register set to be used for the level 2 MPU. VCON2.VMN is also used by the CACHE[A/I].[I/W/WI] instructions to selectively operate on the data cache entries only encached by VM[VCON2.VMN].

HRHV_VCON2 Offset address: $3B008_H$

CPUx virtualization control register 2 Kernel Reset value: $0000\ 0001_H$

CPU_VCON2 Offset address: $0B008_H$

Short address for domain CSFR Kernel Reset value: $0000\ 0001_H$



Field	Bits	Type	Description
VMN	2:0	rw	**VMN** Target virtual machine number. Used by certain CACHE instructions and by the RFH instruction.

(table continues...)

(continued)

Field	Bits	Type	Description
L2_PRS	10:8	rw	**Level 2 MPU PRS** Selection of the protection register set for the level 2 MPU.
0	7:3, 31:11	r	**Reserved - RES** Read as 0; should be written with 0.

4.6.1.74 CPUx base hypervisor vector table pointer

The BHV contains the base address of the hypervisor trap vector table. When a hypervisor trap occurs, the entry address into the hypervisor trap vector table is generated from the hypervisor trap class number (HVTCN) of that trap, left-shifted by 5-bits and then ORd with the contents of the BHV register. The left-shift of the HVTCN results in a spacing of 8-words (32-bytes) between the individual entries in the vector table.

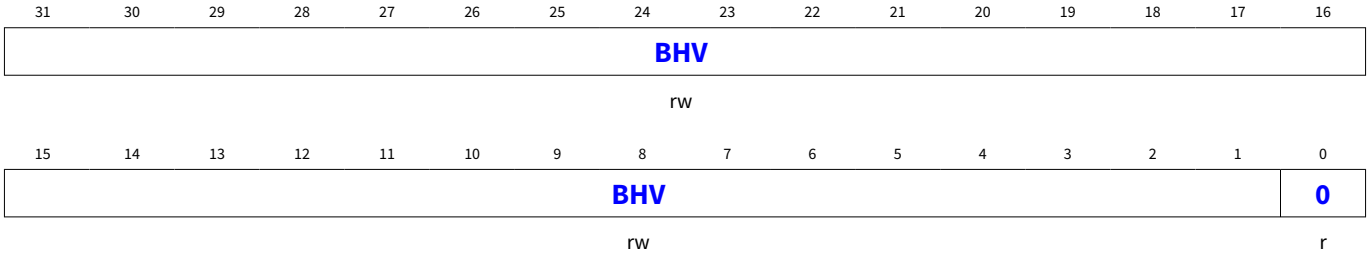
Note: This register is protected by CPUx_PROTSFRE.

HRHV_BHV Offset address: $3B010_H$

CPUx base hypervisor vector table pointer Kernel Reset value: $XXXX\ FD00_H$

CPU_BHV Offset address: $0B010_H$

Short address for domain CSFR Kernel Reset value: $XXXX\ FD00_H$



Field	Bits	Type	Description
BHV	31:1	rw	**Base address of hypervisor trap vector table** The address in the BHV register must be aligned to an even byte address (half-word address). Also, due to the simple ORing of the left-shifted trap identification number and the contents of the BHV register, the alignment of the base address of the vector table must be to a power of two boundary. There are six different hypervisor trap classes, resulting in Trap Classes from 0 to 5. The contents of BHV should therefore be set to at least a 256 byte boundary (8 Trap Classes * 8-word spacing, with 8 being the nearest power of two for 6).
0	0	r	**Reserved - RES** Read as 0; should be written as 0.

4.6.1.75 CPUx Virtual machine n interrupt state

The VMn_ICR registers allow the CPU to evaluate whether an incoming interrupt request can be serviced by a virtual machine (irrespective of its current execution status).

HRHV_VMn_ICR (n=0-7) — Offset address: $3B100_H$+n*4

CPUx Virtual machine n interrupt state — Kernel Reset value: $0000\ 0000_H$

CPU_VMn_ICR (n=0-7) — Offset address: $0B100_H$+n*4

Short address for domain CSFR — Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								PIPN							
r								rh							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IE	0							CCPN							
rwh	r							rwh							

Field	Bits	Type	Description
CCPN	7:0	rwh	**Virtual machine n current priority number** The Current CPU Priority Number (CCPN) bit-field indicates the current priority level of the Virtual Machine. For the currently running virtual machine, it is automatically updated by hardware on entry or exit of Interrupt Service Routines (ISRs) and through the execution of a BISR instruction. CCPN can also be updated through an MTCR instruction.
IE	15	rwh	**VM interrupt enable bit** The interrupt enable bit enables this VM's service request system. VMn_ICR.IE is automatically updated by hardware on entry and exit of an Interrupt Service Routine (ISR). VMn_ICR.IE is cleared to 0 when an interrupt is taken, and is restored to the previous value when the ISR executes an RFE instruction to terminate itself. VMn_ICR.IE can also be updated through the execution of the ENABLE, DISABLE, RESTORE, MTCR, MTDCR and BISR instructions. 0_B Interrupt system is disabled for this VM 1_B Interrupt system is enabled for this VM
PIPN	23:16	rh	**Pending interrupt priority number** A read-only bit-field that is updated by the ICU at the end of each interrupt arbitration process. It indicates the priority number of the pending service request for a specific VM. VMn_ICR.PIPN is set to 0 when no request is pending, and at the beginning of each new arbitration process. ... 00_H No valid pending request 01_H Request pending, lowest priority FF_H Request pending, highest priority
0	14:8, 31:24	r	**Reserved - RES** Read as 0; should be written with 0.

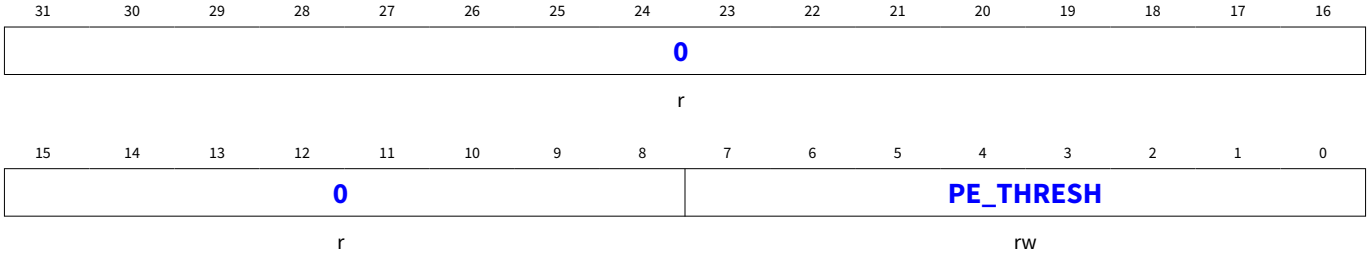
4.6.1.76 CPUx Virtual machine n pre-emption threshold

HRHV_VMn_PETHRESH (n=0-7) Offset address: $3B200_H$+n*4

CPUx Virtual machine n pre-emption threshold Kernel Reset value: $0000\ 0000_H$

CPU_VMn_PETHRESH (n=0-7) Offset address: $0B200_H$+n*4

Short address for domain CSFR Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
PE_THRESH	7:0	rw	**Pre-emption threshold**
0	31:8	r	**Reserved - RES** Read as 0; should be written as 0.

4.6.1.77 CPUx data integrity error address register

This register contains the physical address accessed by the operation that encountered a uncorrectable data memory integrity error. This register is only updated if DIETR.IED is zero.

HRA_DIEAR Offset address: 19020_H

CPUx data integrity error address register Kernel Reset value: $0000\ 0000_H$

HRB_DIEAR Offset address: 29020_H

CPUx data integrity error address register Kernel Reset value: $0000\ 0000_H$

HRHV_DIEAR Offset address: 39020_H

CPUx data integrity error address register Kernel Reset value: $0000\ 0000_H$

CPU_DIEAR Offset address: 09020_H

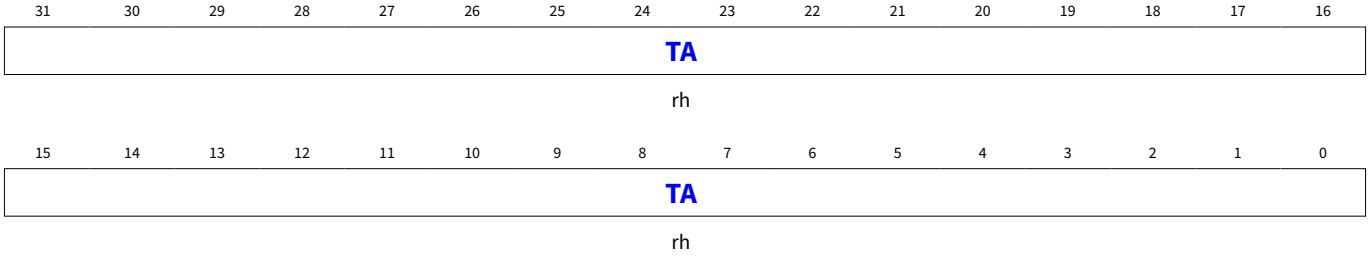
Short address for domain CSFR Kernel Reset value: $0000\ 0000_H$

CPU_HVHRA_DIEAR Offset address: 09021_H

Short address for domain CSFR Kernel Reset value: $0000\ 0000_H$

CPU_HVHRB_DIEAR Offset address: 09022_H

Short address for domain CSFR Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
TA	31:0	rh	**Transaction address** Physical address being accessed by operation that encountered data integrity error.

4.6.1.78 CPUx data integrity error trap register

HRA_DIETR Offset address: 19024_H

CPUx data integrity error trap register Kernel Reset value: $0000\ 0000_H$

HRB_DIETR Offset address: 29024_H

CPUx data integrity error trap register Kernel Reset value: $0000\ 0000_H$

HRHV_DIETR Offset address: 39024_H

CPUx data integrity error trap register Kernel Reset value: $0000\ 0000_H$

CPU_DIETR Offset address: 09024_H

Short address for domain CSFR Kernel Reset value: $0000\ 0000_H$

CPU_HVHRA_DIETR Offset address: 09025_H

Short address for domain CSFR Kernel Reset value: $0000\ 0000_H$

CPU_HVHRB_DIETR Offset address: 09026_H

Short address for domain CSFR Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0				E_VMN_EN	E_VMN			E_PRS_EN	E_PRS			0			IE_MTMV
r				rh	rh			rh	rh			r			rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IE_LPB	IE_DLMU	IE_BS	IE_SP	IE_UNC	E_INFO						IE_BI	IE_S	IE_C	IE_T	IED
rh	rh	rh	rh	rh	rh						rh	rh	rh	rh	rwh

Field	Bits	Type	Description
IED	0	rwh	**Integrity error detected** 0_B Write: Clear IED bit, re-enable DIETR and DIEAR update. Read : No data integrity error condition occurred 1_B Write : No Effect. Read: Data integrity error condition detected. DIETR and DIEAR contents valid, further DIETR and DIEAR updates disabled.
IE_T	1	rh	**Integrity error - TAG memory**
IE_C	2	rh	**Integrity error - cache memory**
IE_S	3	rh	**Integrity error - scratchpad memory**
IE_BI	4	rh	**Integrity error - bus interface**

(table continues...)

(continued)

Field	Bits	Type	Description
E_INFO	10:5	rh	**Error information** If IE_BS = 1: Bus master TAG ID of requesting master. If IE_C = 1: Cache way.
IE_UNC	11	rh	**Integrity error - uncorrectable error detected**
IE_SP	12	rh	**PROT APU protection error detected**
IE_BS	13	rh	**Bus slave access indicator**
IE_DLMU	14	rh	**Integrity error - DLMU**
IE_LPB	15	rh	**Integrity error - local NVM bank**
IE_MTMV	16	rh	**Memory test mode violation detected**
E_PRS	22:20	rh	**Captured PRS**
E_PRS_EN	23	rh	**Captured PRS_EN**
E_VMN	26:24	rh	**Captured VMN**
E_VMN_EN	27	rh	**Captured VMN_EN**
0	19:17, 31:28	r	**Reserved - RES** Read as 0; should be written with 0.

4.6.1.79 CPUx HR specific program integrity error address register

HRA_PIEAR — Offset address: 19210_H

CPUx HR specific program integrity error address register — Kernel Reset value: $0000\ 0000_H$

HRB_PIEAR — Offset address: 29210_H

CPUx HR specific program integrity error address register — Kernel Reset value: $0000\ 0000_H$

HRHV_PIEAR — Offset address: 39210_H

CPUx HR specific program integrity error address register — Kernel Reset value: $0000\ 0000_H$

CPU_PIEAR — Offset address: 09210_H

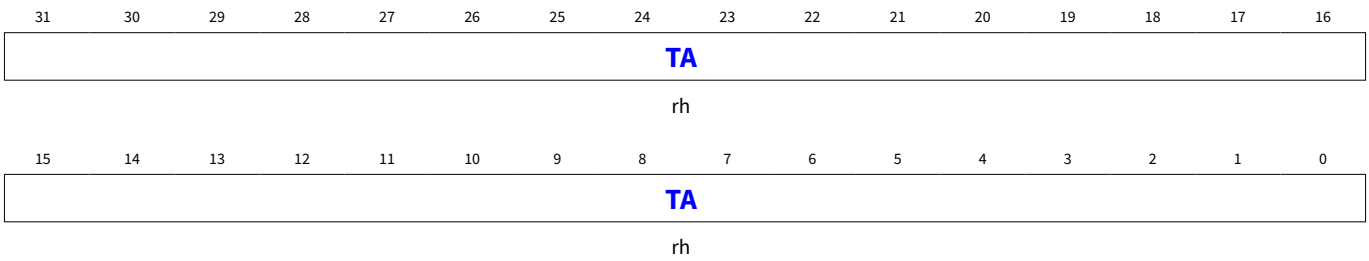
Short address for domain CSFR — Kernel Reset value: $0000\ 0000_H$

CPU_HVHRA_PIEAR — Offset address: 09211_H

Short address for domain CSFR — Kernel Reset value: $0000\ 0000_H$

CPU_HVHRB_PIEAR — Offset address: 09212_H

Short address for domain CSFR — Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
TA	31:0	rh	**Transaction address** Physical address being accessed by operation that encountered data integrity error.

4.6.1.80 CPUx HR specific program integrity error trap register

HRA_PIETR Offset address: 19214_H

CPUx HR specific program integrity error trap register Kernel Reset value: $0000\ 0000_H$

HRB_PIETR Offset address: 29214_H

CPUx HR specific program integrity error trap register Kernel Reset value: $0000\ 0000_H$

HRHV_PIETR Offset address: 39214_H

CPUx HR specific program integrity error trap register Kernel Reset value: $0000\ 0000_H$

CPU_PIETR Offset address: 09214_H

Short address for domain CSFR Kernel Reset value: $0000\ 0000_H$

CPU_HVHRA_PIETR Offset address: 09215_H

Short address for domain CSFR Kernel Reset value: $0000\ 0000_H$

CPU_HVHRB_PIETR Offset address: 09216_H

Short address for domain CSFR Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0				E_VMN_EN	E_VMN			E_PRS_EN	E_PRS			0			IE_MTMV
r				rh	rh			rh	rh			r			rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IE_LPB	IE_ADDR	IE_BS	IE_SP	IE_UNC	E_INFO						IE_BI	IE_S	IE_C	IE_T	IED
rh	rh	rh	rh	rh	rh						rh	rh	rh	rh	rwh

Field	Bits	Type	Description
IED	0	rwh	**Integrity error detected** 0_B Write: Clear IED bit, re-enable PIETR and PIEAR update. Read : No data integrity error condition occurred 1_B Write : No Effect. Read: Data integrity error condition detected. PIETR and PIEAR contents valid, further PIETR and PIEAR updates disabled.
IE_T	1	rh	**Integrity error - TAG memory**
IE_C	2	rh	**Integrity error - cache memory**
IE_S	3	rh	**Integrity error - scratchpad memory**
IE_BI	4	rh	**Integrity error - bus interface**

(table continues...)

(continued)

Field	Bits	Type	Description
E_INFO	10:5	rh	**Error information** If IE_BS= 1: Bus master TAG ID of requesting master. If IE_C = 1: Cache way.
IE_UNC	11	rh	**Integrity error - uncorrectable error detected**
IE_SP	12	rh	**PROT APU protection error detected**
IE_BS	13	rh	**Bus slave access indicator**
IE_ADDR	14	rh	**Address phase error detected at SRI slave interface**
IE_LPB	15	rh	**Integrity error - local NVM bank**
IE_MTMV	16	rh	**Memory test mode violation detected**
E_PRS	22:20	rh	**Captured PRS**
E_PRS_EN	23	rh	**Captured PRS_EN**
E_VMN	26:24	rh	**Captured VMN**
E_VMN_EN	27	rh	**Captured VMN_EN**
0	19:17, 31:28	r	**Reserved - RES** Read as 0; should be written with 0.

4.6.1.81 CPUx HR specific data protection range y, lower bound register

HRA_DPRy_L (y=0-23) — Offset address: $1C000_H$+y*8

CPUx HR specific data protection range y, lower bound register — Kernel Reset value: $0000\ 0000_H$

HRB_DPRy_L (y=0-23) — Offset address: $2C000_H$+y*8

CPUx HR specific data protection range y, lower bound register — Kernel Reset value: $0000\ 0000_H$

HRHV_DPRy_L (y=0-23) — Offset address: $3C000_H$+y*8

CPUx HR specific data protection range y, lower bound register — Kernel Reset value: $0000\ 0000_H$

CPU_DPRy_L (y=0-23) — Offset address: $0C000_H$+y*8

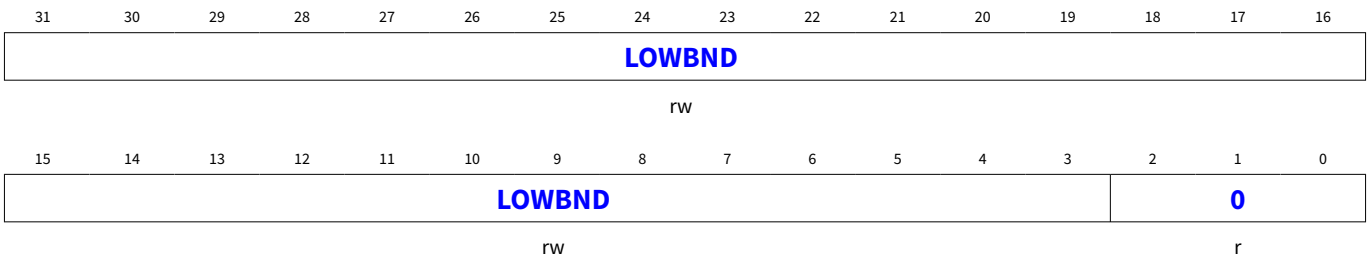
Short address for domain CSFR — Kernel Reset value: $0000\ 0000_H$

CPU_HVHRA_DPRy_L (y=0-23) — Offset address: $0C001_H$+y*8

Short address for domain CSFR — Kernel Reset value: $0000\ 0000_H$

CPU_HVHRB_DPRy_L (y=0-23) — Offset address: $0C002_H$+y*8

Short address for domain CSFR — Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
LOWBND	31:3	rw	**DPRy lower boundary address bits 31:3**
0	2:0	r	**Reserved - RES** Read as 0; should be written with 0.

4.6.1.82 CPUx HR specific data protection range y, upper bound register

HRA_DPRy_U (y=0-23) Offset address: $1C004_H$+y*8

CPUx HR specific data protection range y, upper bound register Kernel Reset value: $0000\ 0000_H$

HRB_DPRy_U (y=0-23) Offset address: $2C004_H$+y*8

CPUx HR specific data protection range y, upper bound register Kernel Reset value: $0000\ 0000_H$

HRHV_DPRy_U (y=0-23) Offset address: $3C004_H$+y*8

CPUx HR specific data protection range y, upper bound register Kernel Reset value: $0000\ 0000_H$

CPU_DPRy_U (y=0-23) Offset address: $0C004_H$+y*8

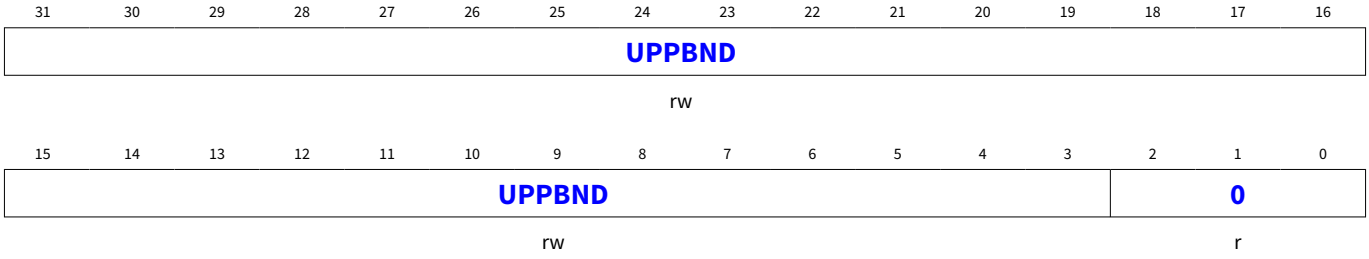
Short address for domain CSFR Kernel Reset value: $0000\ 0000_H$

CPU_HVHRA_DPRy_U (y=0-23) Offset address: $0C005_H$+y*8

Short address for domain CSFR Kernel Reset value: $0000\ 0000_H$

CPU_HVHRB_DPRy_U (y=0-23) Offset address: $0C006_H$+y*8

Short address for domain CSFR Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
UPPBND	31:3	rw	**DPRy upper boundary address bits 31:3**
0	2:0	r	**Reserved - RES** Read as 0; should be written with 0.

4.6.1.83 CPUx HR specific code protection range y, lower bound register

HRA_CPRy_L (y=0-15) — Offset address: $1D000_H$+y*8

CPUx HR specific code protection range y, lower bound register — Kernel Reset value: $0000\ 0000_H$

HRB_CPRy_L (y=0-15) — Offset address: $2D000_H$+y*8

CPUx HR specific code protection range y, lower bound register — Kernel Reset value: $0000\ 0000_H$

HRHV_CPRy_L (y=0-15) — Offset address: $3D000_H$+y*8

CPUx HR specific code protection range y, lower bound register — Kernel Reset value: $0000\ 0000_H$

CPU_CPRy_L (y=0-15) — Offset address: $0D000_H$+y*8

Short address for domain CSFR — Kernel Reset value: $0000\ 0000_H$

CPU_HVHRA_CPRy_L (y=0-15) — Offset address: $0D001_H$+y*8

Short address for domain CSFR — Kernel Reset value: $0000\ 0000_H$

CPU_HVHRB_CPRy_L (y=0-15) — Offset address: $0D002_H$+y*8

Short address for domain CSFR — Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LOWBND															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LOWBND											0				
rw											r				

Field	Bits	Type	Description
LOWBND	31:5	rw	**CPRy lower boundary address bits 31:5**
0	4:0	r	**Reserved - RES** The 5 least significant bits are not writable and always return zero.

4.6.1.84 CPUx HR specific code protection range y, upper bound register

HRA_CPRy_U (y=0-15) — Offset address: $1D004_H$+y*8
CPUx HR specific code protection range y, upper bound register — Kernel Reset value: $0000\ 0000_H$

HRB_CPRy_U (y=0-15) — Offset address: $2D004_H$+y*8
CPUx HR specific code protection range y, upper bound register — Kernel Reset value: $0000\ 0000_H$

HRHV_CPRy_U (y=0-15) — Offset address: $3D004_H$+y*8
CPUx HR specific code protection range y, upper bound register — Kernel Reset value: $0000\ 0000_H$

CPU_CPRy_U (y=0-15) — Offset address: $0D004_H$+y*8
Short address for domain CSFR — Kernel Reset value: $0000\ 0000_H$

CPU_HVHRA_CPRy_U (y=0-15) — Offset address: $0D005_H$+y*8
Short address for domain CSFR — Kernel Reset value: $0000\ 0000_H$

CPU_HVHRB_CPRy_U (y=0-15) — Offset address: $0D006_H$+y*8
Short address for domain CSFR — Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UPPBND															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UPPBND											0				
rw											r				

Field	Bits	Type	Description
UPPBND	31:5	rw	**CPRy upper boundary address 31:5**
0	4:0	r	**Reserved - RES** Read as 0; should be written with 0.

4.6.1.85 CPUx HR specific code protection execute enable register set y (y=0-3)

HRA_CPXE_y (y=0-3) — Offset address: $1E000_H$+y*4

CPUx HR specific code protection execute enable register set y — Kernel Reset value: $0000\ 0000_H$

HRB_CPXE_y (y=0-3) — Offset address: $2E000_H$+y*4

CPUx HR specific code protection execute enable register set y — Kernel Reset value: $0000\ 0000_H$

HRHV_CPXE_y (y=0-3) — Offset address: $3E000_H$+y*4

CPUx HR specific code protection execute enable register set y — Kernel Reset value: $0000\ 0000_H$

CPU_CPXE_y (y=0-3) — Offset address: $0E000_H$+y*4

Short address for domain CSFR — Kernel Reset value: $0000\ 0000_H$

CPU_HVHRA_CPXE_y (y=0-3) — Offset address: $0E001_H$+y*4

Short address for domain CSFR — Kernel Reset value: $0000\ 0000_H$

CPU_HVHRB_CPXE_y (y=0-3) — Offset address: $0E002_H$+y*4

Short address for domain CSFR — Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
XE15	XE14	XE13	XE12	XE11	XE10	XE09	XE08	XE07	XE06	XE05	XE04	XE03	XE02	XE01	XE00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
XEn (n=00-15)	n	rw	**Execute enable range-n select** 0_B Code protection range-n not enabled for execution 1_B Code protection range-n enabled for execution
0	31:16	r	**Reserved - RES** Read as 0; should be written with 0.

4.6.1.86 CPUx HR specific data protection read enable register set y (y=0-3)

HRA_DPRE_y (y=0-3) — Offset address: $1E010_H$+y*4

CPUx HR specific data protection read enable register set y — Kernel Reset value: $0000\ 0000_H$

HRB_DPRE_y (y=0-3) — Offset address: $2E010_H$+y*4

CPUx HR specific data protection read enable register set y — Kernel Reset value: $0000\ 0000_H$

HRHV_DPRE_y (y=0-3) — Offset address: $3E010_H$+y*4

CPUx HR specific data protection read enable register set y — Kernel Reset value: $0000\ 0000_H$

CPU_DPRE_y (y=0-3) — Offset address: $0E010_H$+y*4

Short address for domain CSFR — Kernel Reset value: $0000\ 0000_H$

CPU_HVHRA_DPRE_y (y=0-3) — Offset address: $0E011_H$+y*4

Short address for domain CSFR — Kernel Reset value: $0000\ 0000_H$

CPU_HVHRB_DPRE_y (y=0-3) — Offset address: $0E012_H$+y*4

Short address for domain CSFR — Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								RE23	RE22	RE21	RE20	RE19	RE18	RE17	RE16
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RE15	RE14	RE13	RE12	RE11	RE10	RE09	RE08	RE07	RE06	RE05	RE04	RE03	RE02	RE01	RE00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
REn (n=00-23)	n	rw	**Read enable range-n select** 0_B Data protection range-n not enabled for data read 1_B Data protection range-n enabled for data read
0	31:24	r	**Reserved - RES** Read as 0; should be written with 0.

4.6.1.87 CPUx HR specific data protection write enable register set y (y=0-3)

HRA_DPWE_y (y=0-3) — Offset address: $1E020_H$+y*4

CPUx HR specific data protection write enable register set y — Kernel Reset value: $0000\ 0000_H$

HRB_DPWE_y (y=0-3) — Offset address: $2E020_H$+y*4

CPUx HR specific data protection write enable register set y — Kernel Reset value: $0000\ 0000_H$

HRHV_DPWE_y (y=0-3) — Offset address: $3E020_H$+y*4

CPUx HR specific data protection write enable register set y — Kernel Reset value: $0000\ 0000_H$

CPU_DPWE_y (y=0-3) — Offset address: $0E020_H$+y*4

Short address for domain CSFR — Kernel Reset value: $0000\ 0000_H$

CPU_HVHRA_DPWE_y (y=0-3) — Offset address: $0E021_H$+y*4

Short address for domain CSFR — Kernel Reset value: $0000\ 0000_H$

CPU_HVHRB_DPWE_y (y=0-3) — Offset address: $0E022_H$+y*4

Short address for domain CSFR — Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WE23	WE22	WE21	WE20	WE19	WE18	WE17	WE16
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WE15	WE14	WE13	WE12	WE11	WE10	WE09	WE08	WE07	WE06	WE05	WE04	WE03	WE02	WE01	WE00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
WEn (n=00-23)	n	rw	**Write enable range-n select** 0_B Data protection range-n not enabled for data write 1_B Data protection range-n enabled for data write
0	31:24	r	**Reserved - RES** Read as 0; should be written with 0.

4.6.1.88 CPUx HR specific code protection execute enable register set y (y=4-7)

HRA_CPXE_y (y=4-7) — Offset address: $1E040_H+(y-4)*4$

CPUx HR specific code protection execute enable register set y — Kernel Reset value: $0000\ 0000_H$

HRB_CPXE_y (y=4-7) — Offset address: $2E040_H+(y-4)*4$

CPUx HR specific code protection execute enable register set y — Kernel Reset value: $0000\ 0000_H$

HRHV_CPXE_y (y=4-7) — Offset address: $3E040_H+(y-4)*4$

CPUx HR specific code protection execute enable register set y — Kernel Reset value: $0000\ 0000_H$

CPU_CPXE_y (y=4-7) — Offset address: $0E040_H+(y-4)*4$

Short address for domain CSFR — Kernel Reset value: $0000\ 0000_H$

CPU_HVHRA_CPXE_y (y=4-7) — Offset address: $0E041_H+(y-4)*4$

Short address for domain CSFR — Kernel Reset value: $0000\ 0000_H$

CPU_HVHRB_CPXE_y (y=4-7) — Offset address: $0E042_H+(y-4)*4$

Short address for domain CSFR — Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
XE15	XE14	XE13	XE12	XE11	XE10	XE09	XE08	XE07	XE06	XE05	XE04	XE03	XE02	XE01	XE00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
XEn (n=00-15)	n	rw	**Execute enable range-n select** 0_B Code protection range-n not enabled for execution 1_B Code protection range-n enabled for execution
0	31:16	r	**Reserved - RES** Read as 0; should be written with 0.

4.6.1.89 CPUx HR specific data protection read enable register set y (y=4-7)

HRA_DPRE_y (y=4-7) — Offset address: $1E050_H$+(y-4)*4

CPUx HR specific data protection read enable register set y — Kernel Reset value: $0000\ 0000_H$

HRB_DPRE_y (y=4-7) — Offset address: $2E050_H$+(y-4)*4

CPUx HR specific data protection read enable register set y — Kernel Reset value: $0000\ 0000_H$

HRHV_DPRE_y (y=4-7) — Offset address: $3E050_H$+(y-4)*4

CPUx HR specific data protection read enable register set y — Kernel Reset value: $0000\ 0000_H$

CPU_DPRE_y (y=4-7) — Offset address: $0E050_H$+(y-4)*4

Short address for domain CSFR — Kernel Reset value: $0000\ 0000_H$

CPU_HVHRA_DPRE_y (y=4-7) — Offset address: $0E051_H$+(y-4)*4

Short address for domain CSFR — Kernel Reset value: $0000\ 0000_H$

CPU_HVHRB_DPRE_y (y=4-7) — Offset address: $0E052_H$+(y-4)*4

Short address for domain CSFR — Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								RE23	RE22	RE21	RE20	RE19	RE18	RE17	RE16
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RE15	RE14	RE13	RE12	RE11	RE10	RE09	RE08	RE07	RE06	RE05	RE04	RE03	RE02	RE01	RE00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
REn (n=00-23)	n	rw	**Read enable range-n select** 0_B Data protection range-n not enabled for data read 1_B Data protection range-n enabled for data read
0	31:24	r	**Reserved - RES** Read as 0; should be written with 0.

4.6.1.90 CPUx HR specific data protection write enable register set y (y=4-7)

HRA_DPWE_y (y=4-7) Offset address: $1E060_H$+(y-4)*4

CPUx HR specific data protection write enable register set y Kernel Reset value: $0000\ 0000_H$

HRB_DPWE_y (y=4-7) Offset address: $2E060_H$+(y-4)*4

CPUx HR specific data protection write enable register set y Kernel Reset value: $0000\ 0000_H$

HRHV_DPWE_y (y=4-7) Offset address: $3E060_H$+(y-4)*4

CPUx HR specific data protection write enable register set y Kernel Reset value: $0000\ 0000_H$

CPU_DPWE_y (y=4-7) Offset address: $0E060_H$+(y-4)*4

Short address for domain CSFR Kernel Reset value: $0000\ 0000_H$

CPU_HVHRA_DPWE_y (y=4-7) Offset address: $0E061_H$+(y-4)*4

Short address for domain CSFR Kernel Reset value: $0000\ 0000_H$

CPU_HVHRB_DPWE_y (y=4-7) Offset address: $0E062_H$+(y-4)*4

Short address for domain CSFR Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WE23	WE22	WE21	WE20	WE19	WE18	WE17	WE16
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WE15	WE14	WE13	WE12	WE11	WE10	WE09	WE08	WE07	WE06	WE05	WE04	WE03	WE02	WE01	WE00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
WEn (n=00-23)	n	rw	**Write enable range-n select** 0_B Data protection range-n not enabled for data write 1_B Data protection range-n enabled for data write
0	31:24	r	**Reserved - RES** Read as 0; should be written with 0.

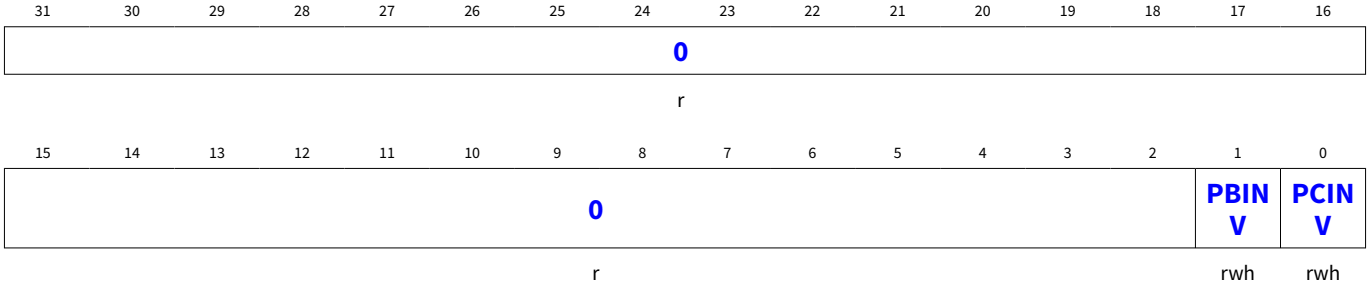
4.6.1.91 CPUx program control 1

HRA_PCON1 Offset address: 19204_H

CPUx program control 1 Kernel Reset value: $0000\ 0000_H$

CPU_PCON1 Offset address: 09204_H

Short address for domain CSFR Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
PCINV	0	rwh	**Program cache invalidate** 0_B Write: No effect, normal instruction cache operation. Read : Normal operation, instruction cache available 1_B Write : Initiate invalidation of entire instruction cache. Read: Instruction cache invalidation in progress. Instruction cache unavailable.
PBINV	1	rwh	**Program buffer invalidate** Write Operation: This field returns 0 when read. 0_B Write: No effect. Normal program line buffer operation. 1_B Write :Invalidate the program line buffer.
0	31:2	r	**Reserved - RES** Read as 0; should be written with 0.

Table 148 Mirrors of HRA_PCON1

Short name	Long name	Offset address
HRB_PCON1	CPUx program control 1 is an alias.	29204_H
HRHV_PCON1	CPUx program control 1 is an alias.	39204_H

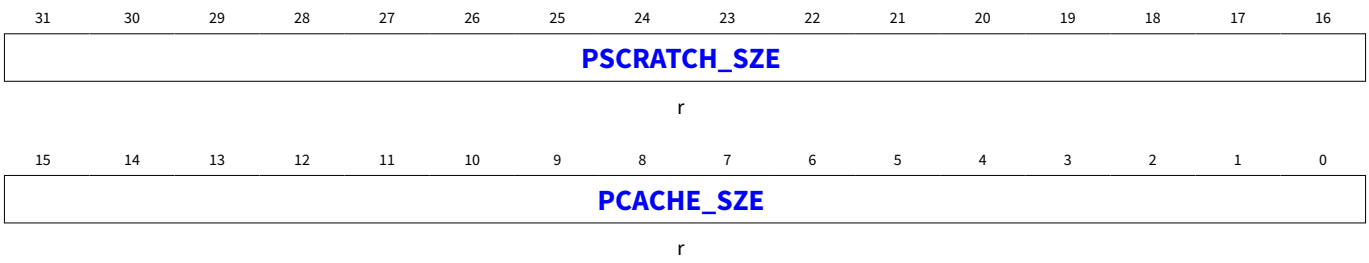
4.6.1.92 CPUx program control 2

HRA_PCON2 Offset address: 19208_H

CPUx program control 2 Kernel Reset value: XXXX XXXX$_H$

CPU_PCON2 Offset address: 09208_H

Short address for domain CSFR Kernel Reset value: XXXX XXXX$_H$



Field	Bits	Type	Description
PCACHE_SZE	15:0	r	**Program cache size** In KBytes
PSCRATCH_SZE	31:16	r	**Program scratch size** In KBytes

Table 149 Mirrors of HRA_PCON2

Short name	Long name	Offset address
HRB_PCON2	CPUx program control 2 is an alias.	29208_H
HRHV_PCON2	CPUx program control 2 is an alias.	39208_H

4.6.1.93 CPUx program control 0

HRA_PCON0 Offset address: $1920C_H$

CPUx program control 0 Kernel Reset value: $0000\ 0002_H$

CPU_PCON0 Offset address: $0920C_H$

Short address for domain CSFR Kernel Reset value: $0000\ 0002_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0														PCBYP	0
r														rw	r

Field	Bits	Type	Description
PCBYP	1	rw	**Program cache bypass** 0_B Cache enabled 1_B Cache bypass (disabled)
0	0, 31:2	r	**Reserved - RES** Read as 0; should be written with 0.

Table 150 Mirrors of HRA_PCON0

Short name	Long name	Offset address
HRB_PCON0	CPUx program control 0 is an alias.	$2920C_H$
HRHV_PCON0	CPUx program control 0 is an alias.	$3920C_H$

4.6.1.94 CPUx HR specific program synchronous trap register

PSTR contains synchronous trap information for the program memory system. The register is updated with trap information for PSE traps to aid the localisation of faults. The register is only set whenever a trap is detected and the register has no bits already set. It is cleared by a CSFR write (independent of data value).

HRA_PSTR	Offset address:	19200_H
CPUx HR specific program synchronous trap register	Kernel Reset value:	$0000\ 0000_H$
HRB_PSTR	Offset address:	29200_H
CPUx HR specific program synchronous trap register	Kernel Reset value:	$0000\ 0000_H$
HRHV_PSTR	Offset address:	39200_H
CPUx HR specific program synchronous trap register	Kernel Reset value:	$0000\ 0000_H$
CPU_PSTR	Offset address:	09200_H
Short address for domain CSFR	Kernel Reset value:	$0000\ 0000_H$
CPU_HVHRA_PSTR	Offset address:	09201_H
Short address for domain CSFR	Kernel Reset value:	$0000\ 0000_H$
CPU_HVHRB_PSTR	Offset address:	09202_H
Short address for domain CSFR	Kernel Reset value:	$0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	FME	0	FPE	0									FBE	0	FRE
r	rwh	r	rwh	r									rwh	r	rwh

Field	Bits	Type	Description
FRE	0	rwh	**Fetch range error** A Fetch Range Error occurs whenever an access to the Program Scratch is outside the range of the SRAM.
FBE	2	rwh	**Fetch bus error** A Fetch bus error will be set whenever the SRI flags an error due a fetch from external memory. This will be set for both direct fetches from the bus and for cache refills.
FPE	12	rwh	**Fetch peripheral error** A Fetch peripheral error will be flagged whenever a fetch is attempted to peripheral space.
FME	14	rwh	**Fetch MSIST error** During SIST mode, a fetch from the PTAG will cause a PSE trap to occur.
0	1, 11:3, 13, 31:15	r	**Reserved - RES** Read as 0; should be written with 0.

4.6.1.95 CPUx HR specific temporal protection system control register

Definition of the temporal protection system control register.

HRA_TPS_CON — Offset address: $1E400_H$
CPUx HR specific temporal protection system control register — Kernel Reset value: $0000\ 0000_H$

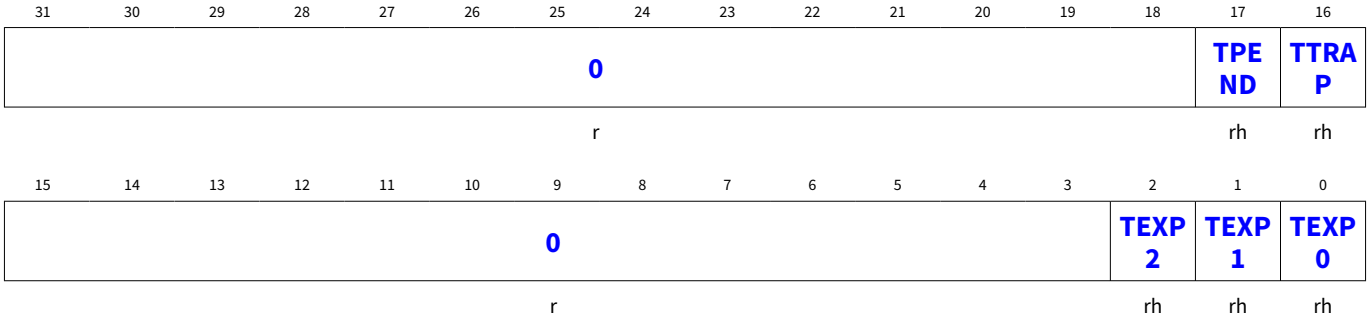
HRB_TPS_CON — Offset address: $2E400_H$
CPUx HR specific temporal protection system control register — Kernel Reset value: $0000\ 0000_H$

HRHV_TPS_CON — Offset address: $3E400_H$
CPUx HR specific temporal protection system control register — Kernel Reset value: $0000\ 0000_H$

CPU_TPS_CON — Offset address: $0E400_H$
Short address for domain CSFR — Kernel Reset value: $0000\ 0000_H$

CPU_HVHRA_TPS_CON — Offset address: $0E401_H$
Short address for domain CSFR — Kernel Reset value: $0000\ 0000_H$

CPU_HVHRB_TPS_CON — Offset address: $0E402_H$
Short address for domain CSFR — Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
TEXPn (n=0-2)	n	rh	**Timer n expired flag** Set when the corresponding timer expires. Cleared on any write to the TPS_TIMERn register.
TTRAP	16	rh	**Temporal protection trap** If set, indicates that a TAE trap has been taken. Any subsequent TAE traps requests are disabled. Any write to the register clears the flag and re-enables TAE traps.
TPEND	17	rh	**TAE pending flag** If set, indicates that a TAE trap has been requested but not yet taken. Any subsequent TAE traps are disabled. TPEND is automatically cleared when TTRAP is set.
0	15:3, 31:18	r	**Reserved - RES** Read as 0; should be written with 0.

4.6.1.96 CPUx HR specific temporal protection system timer register n

Definition of the temporal protection system timer register n.

HRA_TPS_TIMERn (n=0-2) — Offset address: $1E404_H$+n*4

CPUx HR specific temporal protection system timer register n — Kernel Reset value: $0000\ 0000_H$

HRB_TPS_TIMERn (n=0-2) — Offset address: $2E404_H$+n*4

CPUx HR specific temporal protection system timer register n — Kernel Reset value: $0000\ 0000_H$

HRHV_TPS_TIMERn (n=0-2) — Offset address: $3E404_H$+n*4

CPUx HR specific temporal protection system timer register n — Kernel Reset value: $0000\ 0000_H$

CPU_TPS_TIMERn (n=0-2) — Offset address: $0E404_H$+n*4

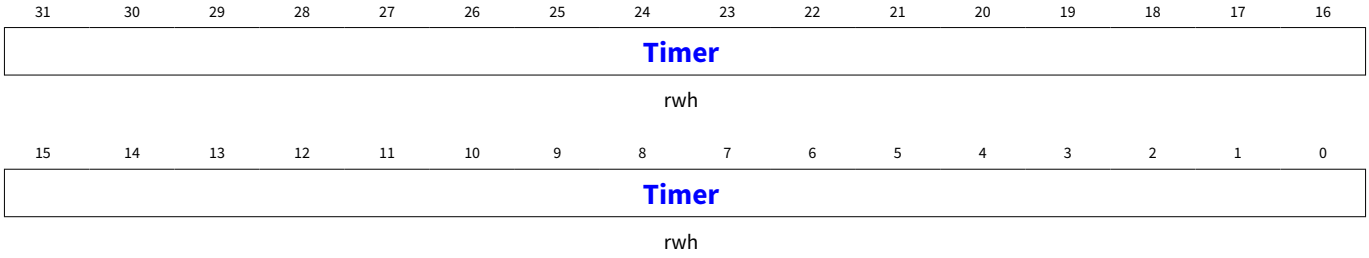
Short address for domain CSFR — Kernel Reset value: $0000\ 0000_H$

CPU_HVHRA_TPS_TIMERn (n=0-2) — Offset address: $0E405_H$+n*4

Short address for domain CSFR — Kernel Reset value: $0000\ 0000_H$

CPU_HVHRB_TPS_TIMERn (n=0-2) — Offset address: $0E406_H$+n*4

Short address for domain CSFR — Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
Timer	31:0	rwh	**Temporal protection timer** Writing zero de-activates the Timer. Writing a non-zero value starts the Timer. Any write clears the corresponding TPS_CON.TEXPn flag. Read returns the current value of Timer.

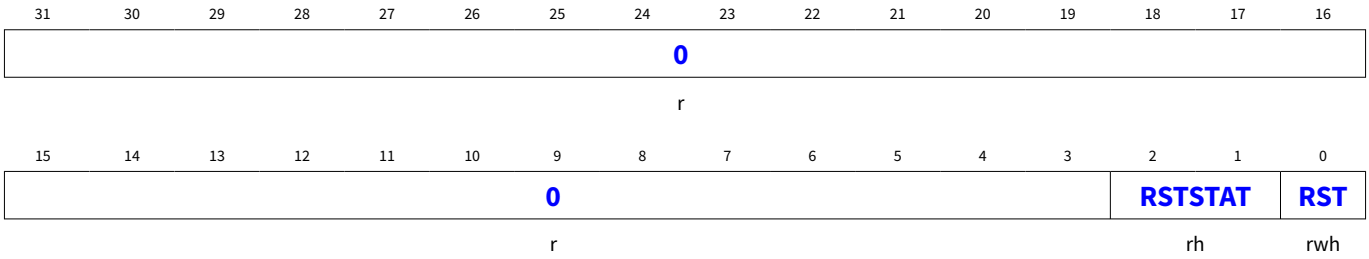
4.6.1.97 CPUx reset register 0

The CPU Kernel Reset Register 0 is used to reset the CPU kernel. CPU registers related to the Debug Reset (Class 1), STM registers and CFI registers (FSFR range) are not influenced. To reset a module kernel it is necessary to set the RST bits by writing with ´1´ in both CPU Kernel Reset Registers. The RST bits will be re-set at the end of the kernel reset sequence.

CPU Kernel Reset Register 0 includes a kernel reset status bits that is set to ´1´ at the start of the reset sequence. These bit can be used to detect that a kernel reset was processed. These bits can be re-set to ´0´ by writing ´1´ to the related KRSTCLRx.CLR register bit.

KRST0 — Offset address: $0D000_H$

CPUx reset register 0 — Kernel Reset value: $0000\ 0000_H$



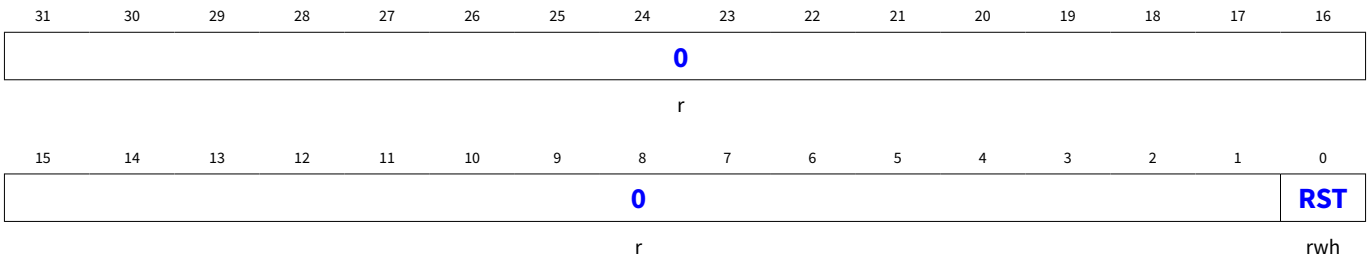
Field	Bits	Type	Description
RST	0	rwh	**Kernel reset** This reset bit can be used to request a kernel reset. The kernel reset will be executed if the reset bits of both kernel registers are set. The RST bit will be cleared (re-set to ´0´) by the BPI_FPI after the kernel reset was executed. 0_B No kernel reset was requested 1_B A kernel reset was requested
RSTSTAT	2:1	rh	**Kernel reset status** This bit indicates whether a kernel reset was executed or not. This bit is set immediately after the execution of a kernel reset. These bits can be cleared by writing ´1´ to the CLR bit in the related KRSTCLR register. 00_B No kernel reset was executed 01_B Kernel reset was requested by hardware since last clear (SMU) 10_B Kernel reset was requested by software since last clear (KRST)
0	31:3	r	**Reserved - RES** Read as 0; should be written with 0.

4.6.1.98 CPUx reset register 1

The CPU kernel reset register 1 is used to reset the CPU kernel. To reset the CPU kernel it is necessary to set the RST bits by writing with ´1´ in both Kernel Reset Registers CPU_KRSTx1.RST and CPU_KRSTx0.RST. The RST bit will be re-set at end of the reset sequence.

KRST1 Offset address: $0D004_H$

CPUx reset register 1 Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
RST	0	rwh	**Kernel reset** This reset bit can be used to request a kernel reset. The kernel reset will be executed if the reset bits of both kernel reset registers is set. The RST bit will be cleared (re-set to ´0´) after the kernel reset was executed. 0_B No kernel reset was requested 1_B A kernel reset was requested
0	31:1	r	**Reserved - RES** Read as 0; should be written with 0.

4.6.1.99 CPUx reset clear register

The kernel reset register clear register is used to clear the Kernel Reset Status bits (CPU_KRST0.RSTSTAT).

KRSTCLR Offset address: $0D008_H$

CPUx reset clear register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															CLR
r															w

Field	Bits	Type	Description
CLR	0	w	**Kernel reset status clear** Read always as 0. 0_B No action 1_B Clear kernel reset status KRST0.RSTSTAT[1:0]
0	31:1	r	**Reserved - RES** Read as 0; should be written with 0.

4.6.1.100 CPUx overlay range select register

OSEL Offset address: $0FB00_H$

CPUx overlay range select register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SHOVEN31	SHOVEN30	SHOVEN29	SHOVEN28	SHOVEN27	SHOVEN26	SHOVEN25	SHOVEN24	SHOVEN23	SHOVEN22	SHOVEN21	SHOVEN20	SHOVEN19	SHOVEN18	SHOVEN17	SHOVEN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SHOVEN15	SHOVEN14	SHOVEN13	SHOVEN12	SHOVEN11	SHOVEN10	SHOVEN09	SHOVEN08	SHOVEN07	SHOVEN06	SHOVEN05	SHOVEN04	SHOVEN03	SHOVEN02	SHOVEN01	SHOVEN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
SHOVENi (i=00-31)	i	rw	**Shadow overlay enable - SHOVEN i** This bit provides shadow configuration for overlay block 'i' 0_B Overlay block 'i' is disabled when OVCCON.OVSTRT is set 1_B Overlay block 'i' is enabled when OVCCON.OVSTRT is set

4.6.1.101 CPUx redirected address base register i (i=0-1)

RABRi (i=0-1) Offset address: $0FB10_H+i*12$

CPUx redirected address base register i Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OVEN	0			OMEM				OBASE							
rwh	r			rw				rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OBASE											0				
rw											r				

Field	Bits	Type	Description
OBASE	23:5	rw	**Overlay block base address (Low priority)** Bits 23..5 of the base address the overlay memory block in the overlay memory. If the corresponding bit in OMASK register is set to one, OBASE bit value is used in the redirection address. If the corresponding bit in OMASK register is set to zero, OBASE bit value is ignored.

(table continues...)

(continued)

Field	Bits	Type	Description
OMEM	27:24	rw	**Overlay memory select** Selects overlay memory used for redirection. Redirection using OMEM values 9 to F_H is only available in emulation devices. For production devices, OMEM values 9 to F_H are reserved 0_H Redirection to Core 0 DSPR/PSPR memory 1_H Redirection to Core 1 DSPR/PSPR memory 2_H Redirection to Core 2 DSPR/PSPR memory 3_H Redirection to Core 3 DSPR/PSPR memory 4_H Redirection to Core 4 DSPR/PSPR memory 5_H Redirection to Core 5 DSPR/PSPR memory 6_H Redirection to Core CS DSPR/PSPR memory 7_H Reserved 8_H Redirection to LMU Low 9_H Redirection to LMU Mid (Emulation device only) A_H Redirection to LMU High (Emulation device only) B_H Reserved C_H Redirection to NVM Low (Emulation device only) D_H Redirection to NVM High (Emulation device only) E_H Redirection to NVM SOTA (Emulation device only) F_H Reserved
OVEN	31	rwh	**Overlay enabled** This bit controls whether the overlay function of overlay block 'i' is enabled. This bit can also be changed when OVCCON.OVSTP or OVCCON.OVSTRT is set. See OVCCON register description. 0_B Overlay function of block 'i' is disabled 1_B Overlay function of block 'i' is enabled
0	4:0, 30:28	r	**Reserved - RES** Reads as 0; should be written with 0.

4.6.1.102 CPUx redirected address base register i (i=2-31)

RABRi (i=2-31) Offset address: $0FB10_H+i*12$

CPUx redirected address base register i Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OVEN	0			OMEM				0		OBASE					
rwh	r			rw				r		rw					

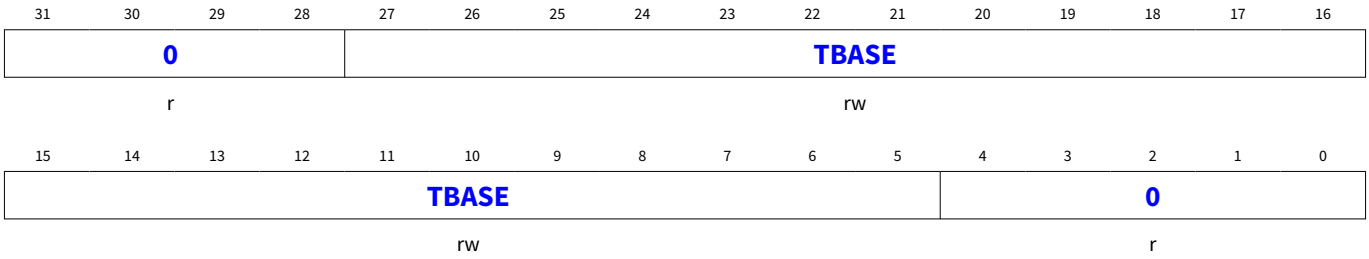
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OBASE											0				
rw											r				

Field	Bits	Type	Description
OBASE	21:5	rw	**Overlay block base address (High priority)** Bits 21..5 of the base address the overlay memory block in the overlay memory. If the corresponding bit in OMASK register is set to one, OBASE bit value is used in the redirection address. If the corresponding bit in OMASK register is set to zero, OBASE bit value is ignored.
OMEM	27:24	rw	**Overlay memory select** Selects overlay memory used for redirection. Redirection using OMEM values 9 to F_H is only available in emulation devices. For production devices, OMEM values 9 to F_H are reserved 0_H Redirection to Core 0 DSPR/PSPR memory 1_H Redirection to Core 1 DSPR/PSPR memory 2_H Redirection to Core 2 DSPR/PSPR memory 3_H Redirection to Core 3 DSPR/PSPR memory 4_H Redirection to Core 4 DSPR/PSPR memory 5_H Redirection to Core 5 DSPR/PSPR memory 6_H Redirection to Core CS DSPR/PSPR memory 7_H Reserved 8_H Redirection to LMU Low 9_H Redirection to LMU Mid (Emulation device only) A_H Redirection to LMU High (Emulation device only) B_H Reserved C_H Redirection to NVM Low (Emulation device only) D_H Redirection to NVM High (Emulation device only) E_H Redirection to NVM SOTA (Emulation device only) F_H Reserved
OVEN	31	rwh	**Overlay enabled** This bit controls whether the overlay function of overlay block 'i' is enabled. This bit can also be changed when OVCCON.OVSTP or OVCCON.OVSTRT is set. See OVCCON register description. 0_B Overlay function of block 'i' is disabled 1_B Overlay function of block 'i' is enabled
0	4:0, 23:22, 30:28	r	**Reserved - RES** Reads as 0; should be written with 0.

4.6.1.103 CPUx overlay target address register i

OTARi (i=0-31) Offset address: $0FB14_H+i*12$

CPUx overlay target address register i Kernel Reset value: $0000\ 0000_H$

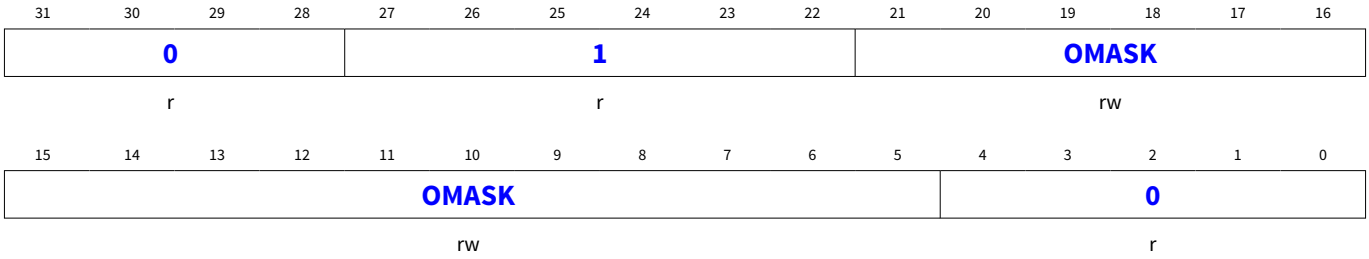


Field	Bits	Type	Description
TBASE	27:5	rw	**Target base** This field holds the base address of the overlay memory block in the target memory. If the corresponding bit in OMASK register is set to one TBASE bit value is used in the address match. If the corresponding bit in OMASK register is set to zero TBASE bit value is ignored.
0	4:0, 31:28	r	**Reserved - RES** Reads as 0; should be written with 0.

4.6.1.104 CPUx overlay mask register i (i=0-1)

OMASKi (i=0-1) Offset address: $0FB18_H+i*12$

CPUx overlay mask register i Kernel Reset value: $0FFF\ FFE0_H$



Field	Bits	Type	Description
OMASK	21:5	rw	**Overlay address mask (Low priority)** This bitfield determines the overlay block size and the bits used for address comparison and translation. "Zero" bits determine the corresponding address bits which are not used in the address comparison and thus determine the block size; corresponding final address bits are derived from the original data address. "One" bits determine the corresponding address bits which are used for the address comparison; corresponding final address bits are derived from RABRi register in case of address match. 00000_H 4096-Kbyte block size 10000_H 2048-Kbyte block size 18000_H 1024-Kbyte block size $1FF00_H$ ... $1FFFE_H$ 64-byte block size $1FFFF_H$ 32-byte block size
0	4:0, 31:28	r	**Reserved - RES** Corresponding address bits are not used in the address comparison. Corresponding final address bits are taken from the original data address.
1	27:22	r	**Fixed 1 Values (Low Priority) - ONE** Corresponding address bits are participating in the address comparison. Corresponding final address bits are taken from RABRi.

4.6.1.105 CPUx overlay mask register i (i=2-31)

OMASKi (i=2-31) Offset address: $0FB18_H+i*12$

CPUx overlay mask register i Kernel Reset value: $0FFF\ FFE0_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0				1									OMASK		
r				r									rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OMASK											0				
rw											r				

Field	Bits	Type	Description
OMASK	18:5	rw	**Overlay address mask (High priority)** This bitfield determines the overlay block size and the bits used for address comparison and translation. "Zero" bits determine the corresponding address bits which are not used in the address comparison and thus determine the block size; corresponding final address bits are derived from the original data address. "One" bits determine the corresponding address bits which are used for the address comparison; corresponding final address bits are derived from RABRi register in case of address match. 0000_H 512-Kbyte block size 2000_H 256-Kbyte block size 3000_H 128-Kbyte block size $3F80_H$ ... $3FFE_H$ 64-byte block size $3FFF_H$ 32-byte block size
0	4:0, 31:28	r	**Reserved - RES** Corresponding address bits are not used in the address comparison. Corresponding final address bits are taken from the original data address.
1	27:19	r	**Fixed 1 Values (High Priority) - ONE** Corresponding address bits are participating in the address comparison. Corresponding final address bits are taken from RABRi.

4.6.1.106 CPUx PROTSPR register safe endinit

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner and in some PROT states, the secure master to update the PROT state as described in the PROT mechanism.

The PROTSPRSE register controls lock / unlock of the local Safe Endinit (SE) protected control registers allocated to the SPR resources.

PROTSPRSE Offset address: $0E008_H$

CPUx PROTSPR register safe endinit Reset values see: Table 152

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OWEN	ODEF	TAGID						PRSEN	PRS			VMEN	VM		
w	rw	rw						rw	rw			rw	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				SEL				0				SWEN	STATE		
r				rw				r				w	rwh		

Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
SEL	11:8	rw	**Range selection** Returns the number of the range being configured when read. Can be written by the PROT owner or CS master to select the range number to be configured if the state transition is valid in the PROT state diagram. 0_H **Const0**: Range 0 selected for configuration … F_H **Const15**: Range 15 selected for configuration
VM	18:16	rw	**Virtual machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual machine definition enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.
PRSEN	23	rw	**Protection set definition enable for PROT owner** 0_B **Disable**: PRS is part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner

(table continues...)

(continued)

Field	Bits	Type	Description
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	7:4, 15:12	r	**Reserved** Read as 0; should be written with 0.

Table 151 Access mode restrictions of PROTSPRSE sorted by descending priority

Mode name	Access mode		Description
write 1 to .**SWEN**	rwh	STATE	
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

Table 152 Reset values of PROTSPRSE

Reset type	Reset value	Note
Application Reset	0000 0000$_H$	Instance specific: Any other CPU
Application Reset	4C00 0000$_H$	Instance specific: CPUcs

4.6.1.107 CPUx shadow SPR, write access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

ACCENSPRCFG_WRA Offset address: 0E020$_H$

CPUx shadow SPR, write access enable register A Reset values see: Table 153

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	**EN30**	**EN29**	**EN28**	**EN27**	**EN26**	**EN25**	**EN24**	**EN23**	**EN22**	**EN21**	**EN20**	**EN19**	**EN18**	**EN17**	**EN16**
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	**EN14**	**EN13**	**EN12**	**EN11**	**EN10**	**EN09**	**EN08**	**EN07**	**EN06**	**EN05**	**EN04**	**EN03**	**EN02**	**EN01**	**EN00**
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access

Table 153 Reset values of ACCENSPRCFG_WRA

Reset type	Reset value	Note
Application Reset	1000 0003$_H$	Instance specific: Any other CPU
Application Reset	0000 1000$_H$	Instance specific: CPUcs, CSRM debug disabled
After Boot-FW Value	1000 3000$_H$	CPUcs, CSRM debug enabled

4.6.1.108 CPUx shadow SPR, write access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

ACCENSPRCFG_WRB Offset address: 0E024$_H$

CPUx shadow SPR, write access enable register B Reset values see: Table 154

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN63	EN62	EN61	EN60	EN59	EN58	EN57	EN56	EN55	EN54	EN53	EN52	EN51	EN50	EN49	EN48
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN47	EN46	EN45	EN44	EN43	EN42	EN41	EN40	EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-63)	q-32	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access

Table 154 Reset values of ACCENSPRCFG_WRB

Reset type	Reset value	Note
Application Reset	0000 0000$_H$	Instance specific: Any other CPU
Application Reset	0000 0000$_H$	Instance specific: CPUcs, CSRM debug disabled
After Boot-FW Value	0000 4000$_H$	CPUcs, CSRM debug enabled

4.6.1.109 CPUx shadow SPR, read access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

ACCENSPRCFG_RDA Offset address: 0E028$_H$

CPUx shadow SPR, read access enable register A Reset values see: Table 155

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access

Table 155 Reset values of ACCENSPRCFG_RDA

Reset type	Reset value	Note
Application Reset	FFFF FFFF$_H$	Instance specific: Any other CPU
Application Reset	0000 1000$_H$	Instance specific: CPUcs, CSRM debug disabled
After Boot-FW Value	1000 3000$_H$	CPUcs, CSRM debug enabled

4.6.1.110 CPUx shadow SPR, read access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

ACCENSPRCFG_RDB Offset address: 0E02C$_H$

CPUx shadow SPR, read access enable register B Reset values see: Table 156

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN63	EN62	EN61	EN60	EN59	EN58	EN57	EN56	EN55	EN54	EN53	EN52	EN51	EN50	EN49	EN48
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN47	EN46	EN45	EN44	EN43	EN42	EN41	EN40	EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-63)	q-32	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access

Table 156 Reset values of ACCENSPRCFG_RDB

Reset type	Reset value	Note
Application Reset	FFFF FFFF$_H$	Instance specific: Any other CPU
Application Reset	0000 4000$_H$	Instance specific: CPUcs

4.6.1.111 CPUx shadow SPR, VM access enable register

This register defines which virtual machine encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each VMx.

ACCENSPRCFG_VM Offset address: 0E030$_H$

CPUx shadow SPR, VM access enable register Application Reset value: 00FF 00FF$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

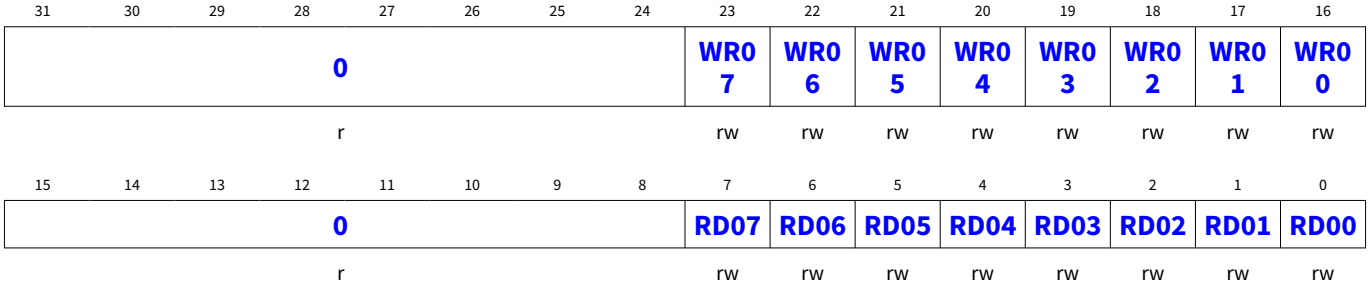
Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for VM ID q** This bit enables read access to the access protected region for transactions with the VM ID q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for VM ID q** This bit enables write access to the access protected region for transactions with the VM ID q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

4.6.1.112 CPUx shadow SPR, PRS access enable register

This register defines which protection register set encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each PRSx.

ACCENSPRCFG_PRS Offset address: 0E034$_H$

CPUx shadow SPR, PRS access enable register Application Reset value: 00FF 00FF$_H$



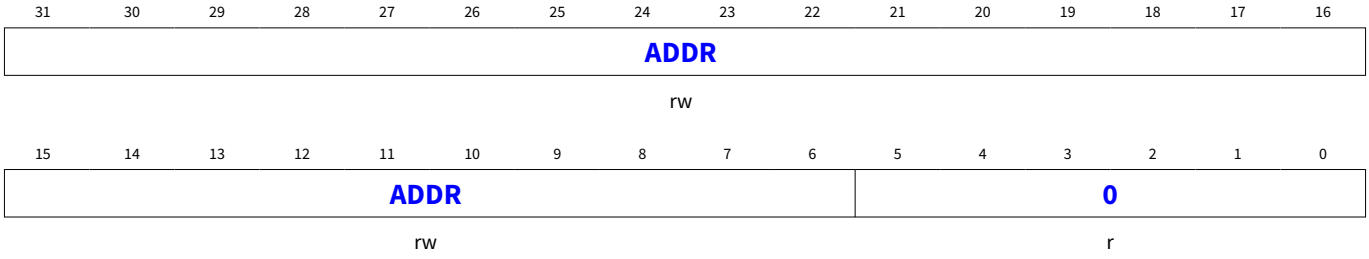
Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for PRS q** This bit enables read access to the access protected region for transactions with the PRS q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for PRS q** This bit enables write access to the access protected region for transactions with the PRS q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

4.6.1.113 CPUx shadow SPR, region lower address register

This register sets the lower bound of the access protected region. Addresses greater than or equal to the lower bound are considered part of the access protected region.

ACCENSPRCFG_RGNLA Offset address: $0E038_H$

CPUx shadow SPR, region lower address register Application Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
ADDR	31:6	rw	**Defines bits 31:6 of the lower bound of the access protected region**
0	5:0	r	**Reserved** Read as 0; should be written with 0.

4.6.1.114 CPUx shadow SPR, region upper address register

This register sets the upper bound of the access protected region. Addresses stricly less than the upper bound are considered part of the access protected region.

ACCENSPRCFG_RGNUA Offset address: $0E03C_H$

CPUx shadow SPR, region upper address register Application Reset value: $FFFF\ FFC0_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADDR															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR										0					
rw										r					

Field	Bits	Type	Description
ADDR	31:6	rw	**Defines bits 31:6 of the upper bound of the access protected region**
0	5:0	r	**Reserved** Read as 0; should be written with 0.

4.6.1.115 CPUx PROTDLMU register safe endinit

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner and in some PROT states, the secure master to update the PROT state as described in the PROT mechanism.

The PROTDLMUSE register controls lock / unlock of the local Safe Endinit (SE) protected control registers allocated to the DLMU resources.

PROTDLMUSE Offset address: $0E048_H$

CPUx PROTDLMU register safe endinit Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OWEN	ODEF	TAGID						PRSEN	PRS			VMEN	VM		
w	rw	rw						rw	rw			rw	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				SEL				0				SWEN	STATE		
r				rw				r				w	rwh		

Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
SEL	11:8	rw	**Range selection** Returns the number of the range being configured when read. Can be written by the PROT owner or CS master to select the range number to be configured if the state transition is valid in the PROT state diagram. 0_H **Const0**: Range 0 selected for configuration … F_H **Const15**: Range 15 selected for configuration
VM	18:16	rw	**Virtual machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual machine definition enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.
PRSEN	23	rw	**Protection set definition enable for PROT owner** 0_B **Disable**: PRS is part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner

(table continues...)

(continued)

Field	Bits	Type	Description
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	7:4, 15:12	r	**Reserved** Read as 0; should be written with 0.

Table 157 Access mode restrictions of PROTDLMUSE sorted by descending priority

Mode name	Access mode		Description
write 1 to .**SWEN**	rwh	STATE	
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

4.6.1.116 CPUx shadow DLMU, write access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

ACCENDLMUCFG_WRA Offset address: $0E060_H$

CPUx shadow DLMU, write access enable register A Application Reset value: $1000\ 0003_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	**EN30**	**EN29**	**EN28**	**EN27**	**EN26**	**EN25**	**EN24**	**EN23**	**EN22**	**EN21**	**EN20**	**EN19**	**EN18**	**EN17**	**EN16**
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	**EN14**	**EN13**	**EN12**	**EN11**	**EN10**	**EN09**	**EN08**	**EN07**	**EN06**	**EN05**	**EN04**	**EN03**	**EN02**	**EN01**	**EN00**
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access

4.6.1.117 CPUx shadow DLMU, write access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

ACCENDLMUCFG_WRB Offset address: $0E064_H$

CPUx shadow DLMU, write access enable register B Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN63	EN62	EN61	EN60	EN59	EN58	EN57	EN56	EN55	EN54	EN53	EN52	EN51	EN50	EN49	EN48
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN47	EN46	EN45	EN44	EN43	EN42	EN41	EN40	EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-63)	q-32	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access

4.6.1.118 CPUx shadow DLMU, read access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

ACCENDLMUCFG_RDA Offset address: $0E068_H$

CPUx shadow DLMU, read access enable register A Application Reset value: $FFFF\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access

4.6.1.119 CPUx shadow DLMU, read access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

ACCENDLMUCFG_RDB Offset address: $0E06C_H$

CPUx shadow DLMU, read access enable register B Application Reset value: $FFFF\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN63	EN62	EN61	EN60	EN59	EN58	EN57	EN56	EN55	EN54	EN53	EN52	EN51	EN50	EN49	EN48
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN47	EN46	EN45	EN44	EN43	EN42	EN41	EN40	EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-63)	q-32	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access

4.6.1.120 CPUx shadow DLMU, VM access enable register

This register defines which virtual machine encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each VMx.

ACCENDLMUCFG_VM Offset address: $0E070_H$

CPUx shadow DLMU, VM access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for VM ID q** This bit enables read access to the access protected region for transactions with the VM ID q. 0_B Disabled for read access 1_B Enabled for read access

(table continues...)

(continued)

Field	Bits	Type	Description
WRq (q=00-07)	q+16	rw	**Write access enable for VM ID q** This bit enables write access to the access protected region for transactions with the VM ID q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

4.6.1.121 CPUx shadow DLMU, PRS access enable register

This register defines which protection register set encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each PRSx.

ACCENDLMUCFG_PRS Offset address: $0E074_H$

CPUx shadow DLMU, PRS access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

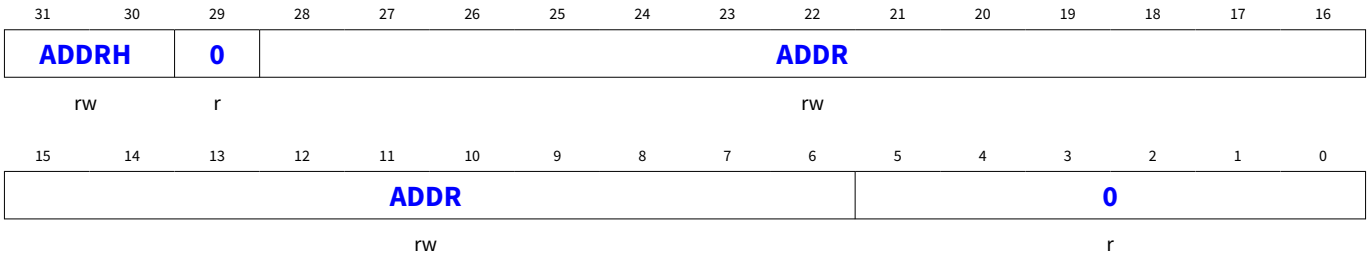
Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for PRS q** This bit enables read access to the access protected region for transactions with the PRS q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for PRS q** This bit enables write access to the access protected region for transactions with the PRS q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

4.6.1.122 CPUx shadow DLMU, region lower address register

This register sets the lower bound of the access protected region. Addresses greater than or equal to the lower bound are considered part of the access protected region.

ACCENDLMUCFG_RGNLA Offset address: $0E078_H$

CPUx shadow DLMU, region lower address register Application Reset value: $0000\ 0000_H$



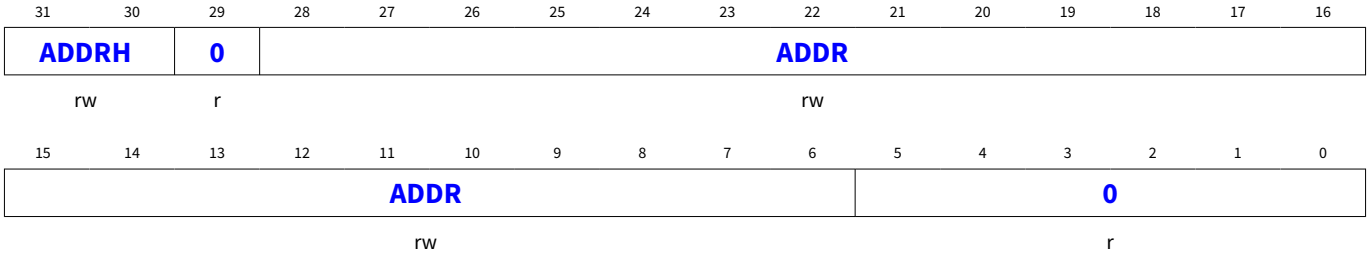
Field	Bits	Type	Description
ADDR	28:6	rw	**Defines bits 28:6 of the lower bound of the access protected region**
ADDRH	31:30	rw	**Defines bits 31:30 of the lower bound of the access protected region**
0	5:0, 29	r	**Reserved** Read as 0; should be written with 0.

4.6.1.123 CPUx shadow DLMU, region upper address register

This register sets the upper bound of the access protected region. Addresses stricly less than the upper bound are considered part of the access protected region.

ACCENDLMUCFG_RGNUA Offset address: $0E07C_H$

CPUx shadow DLMU, region upper address register Application Reset value: $DFFF\ FFC0_H$



Field	Bits	Type	Description
ADDR	28:6	rw	**Defines bits 28:6 of the upper bound of the access protected region**
ADDRH	31:30	rw	**Defines bits 31:30 of the upper bound of the access protected region**
0	5:0, 29	r	**Reserved** Read as 0; should be written with 0.

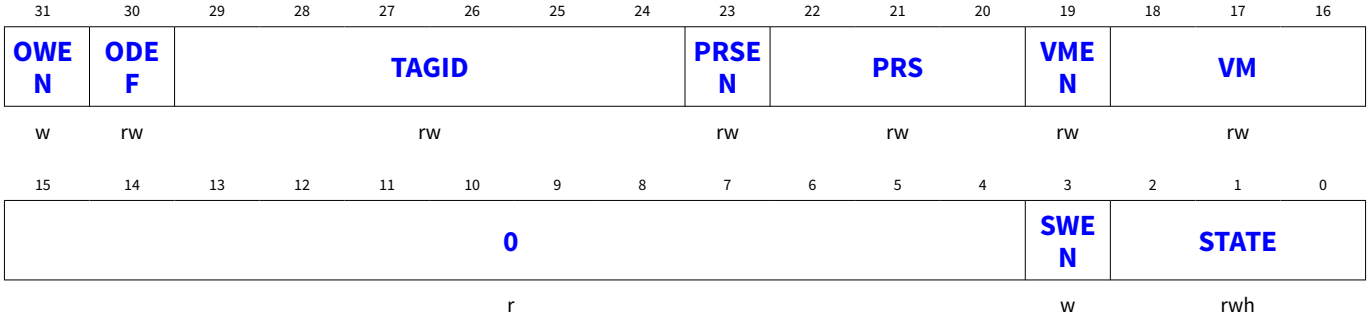
4.6.1.124 CPUx PROTSFR register endinit

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner and in some PROT states, the secure master to update the PROT state as described in the PROT mechanism.

The PROTSFRE register controls lock / unlock of the local Endinit (E) protected control registers.

PROTSFRE Offset address: $0E080_H$

CPUx PROTSFR register endinit Reset values see: Table 159



Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
VM	18:16	rw	**Virtual machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual machine definition enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.
PRSEN	23	rw	**Protection set definition enable for PROT owner** 0_B **Disable**: PRS is part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner

(table continues...)

(continued)

Field	Bits	Type	Description
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; should be written with 0.

Table 158 Access mode restrictions of PROTSFRE sorted by descending priority

Mode name	Access mode		Description
write 1 to .**SWEN**	rwh	STATE	
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

Table 159 Reset values of PROTSFRE

Reset type	Reset value	Note
Application Reset	0000 0000_H	Instance specific: Any other CPU
Application Reset	4C00 0000_H	Instance specific: CPUcs

4.6.1.125 CPUx PROTSFR register safe endinit

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner and in some PROT states, the secure master to update the PROT state as described in the PROT mechanism.

The PROTSFRSE register controls lock / unlock of the local Safe Endinit (SE) protected control registers.

PROTSFRSE Offset address: $0E088_H$

CPUx PROTSFR register safe endinit Reset values see: Table 161

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OWEN	**ODEF**	**TAGID**						**PRSEN**	**PRS**			**VMEN**	**VM**		
w	rw	rw						rw	rw			rw	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												**SWEN**	**STATE**		
r												w	rwh		

Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
VM	18:16	rw	**Virtual machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual machine definition enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.
PRSEN	23	rw	**Protection set definition enable for PROT owner** 0_B **Disable**: PRS is part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; should be written with 0.

Table 160 Access mode restrictions of PROTSFRSE sorted by descending priority

Mode name	Access mode		Description
write 1 to .**SWEN**	rwh	STATE	
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

Table 161 Reset values of PROTSFRSE

Reset type	Reset value	Note
Application Reset	0000 0000$_H$	Instance specific: Any other CPU
Application Reset	4C00 0000$_H$	Instance specific: CPUcs

4.6.1.126 CPUx shadow SFR, write access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

ACCENSFRCFG_WRA Offset address: 0E0A0$_H$

CPUx shadow SFR, write access enable register A Reset values see: Table 162

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	**EN30**	**EN29**	**EN28**	**EN27**	**EN26**	**EN25**	**EN24**	**EN23**	**EN22**	**EN21**	**EN20**	**EN19**	**EN18**	**EN17**	**EN16**
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	**EN14**	**EN13**	**EN12**	**EN11**	**EN10**	**EN09**	**EN08**	**EN07**	**EN06**	**EN05**	**EN04**	**EN03**	**EN02**	**EN01**	**EN00**
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access

Table 162 Reset values of ACCENSFRCFG_WRA

Reset type	Reset value	Note
Application Reset	1000 0003$_H$	Instance specific: Any other CPU
Application Reset	0000 1000$_H$	Instance specific: CPUcs, CSRM debug disabled
After Boot-FW Value	1000 3000$_H$	CPUcs, CSRM debug enabled

4.6.1.127 CPUx shadow SFR, write access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

ACCENSFRCFG_WRB Offset address: $0E0A4_H$

CPUx shadow SFR, write access enable register B Reset values see: Table 163

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN63	EN62	EN61	EN60	EN59	EN58	EN57	EN56	EN55	EN54	EN53	EN52	EN51	EN50	EN49	EN48
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN47	EN46	EN45	EN44	EN43	EN42	EN41	EN40	EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-63)	q-32	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access

Table 163 Reset values of ACCENSFRCFG_WRB

Reset type	Reset value	Note
Application Reset	$0000\ 0000_H$	Instance specific: Any other CPU
Application Reset	$0000\ 0000_H$	Instance specific: CPUcs, CSRM debug disabled
After Boot-FW Value	$0000\ 4000_H$	CPUcs, CSRM debug enabled

4.6.1.128 CPUx shadow SFR, read access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

ACCENSFRCFG_RDA Offset address: $0E0A8_H$

CPUx shadow SFR, read access enable register A Reset values see: Table 164

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access

Table 164 Reset values of ACCENSFRCFG_RDA

Reset type	Reset value	Note
Application Reset	FFFF FFFF$_H$	Instance specific: Any other CPU
Application Reset	0000 1000$_H$	Instance specific: CPUcs, CSRM debug disabled
After Boot-FW Value	1000 3000$_H$	CPUcs, CSRM debug enabled

4.6.1.129 CPUx shadow SFR, read access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

ACCENSFRCFG_RDB Offset address: 0E0AC$_H$

CPUx shadow SFR, read access enable register B Reset values see: Table 165

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN63	EN62	EN61	EN60	EN59	EN58	EN57	EN56	EN55	EN54	EN53	EN52	EN51	EN50	EN49	EN48
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN47	EN46	EN45	EN44	EN43	EN42	EN41	EN40	EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-63)	q-32	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access

Table 165 Reset values of ACCENSFRCFG_RDB

Reset type	Reset value	Note
Application Reset	FFFF FFFF$_H$	Instance specific: Any other CPU
Application Reset	0000 0000$_H$	Instance specific: CPUcs, CSRM debug disabled
After Boot-FW Value	0000 4000$_H$	CPUcs, CSRM debug enabled

4.6.1.130 CPUx shadow SFR, VM access enable register

This register defines which virtual machine encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each VMx.

ACCENSFRCFG_VM Offset address: $0E0B0_H$

CPUx shadow SFR, VM access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for VM ID q** This bit enables read access to the access protected region for transactions with the VM ID q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for VM ID q** This bit enables write access to the access protected region for transactions with the VM ID q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

4.6.1.131 CPUx shadow SFR, PRS access enable register

This register defines which protection register set encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each PRSx.

ACCENSFRCFG_PRS Offset address: $0E0B4_H$

CPUx shadow SFR, PRS access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for PRS q** This bit enables read access to the access protected region for transactions with the PRS q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for PRS q** This bit enables write access to the access protected region for transactions with the PRS q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

4.6.1.132 CPUx PROTRST register endinit

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner and in some PROT states, the secure master to update the PROT state as described in the PROT mechanism.

The PROTRSTE register controls lock / unlock of the local Endinit (E) protected control registers associated with the kernel reset resource.

PROTRSTE Offset address: $0E0C0_H$

CPUx PROTRST register endinit Reset values see: Table 167

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OWEN	ODEF	TAGID						PRSEN	PRS			VMEN	VM		
w	rw	rw						rw	rw			rw	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												SWEN	STATE		
r												w	rwh		

Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)

(table continues...)

(continued)

Field	Bits	Type	Description
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
VM	18:16	rw	**Virtual machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual machine definition enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.
PRSEN	23	rw	**Protection set definition enable for PROT owner** 0_B **Disable**: PRS is part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; should be written with 0.

Table 166 Access mode restrictions of PROTRSTE sorted by descending priority

Mode name	Access mode		Description
write 1 to .**SWEN**	rwh	STATE	
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

Table 167 Reset values of PROTRSTE

Reset type	Reset value	Note
Application Reset	0000 0000$_H$	Instance specific: Any other CPU
Application Reset	4C00 0000$_H$	Instance specific: CPUcs
After Boot-FW Value	7F00 0007$_H$	CPU0

4.6.1.133 CPUx PROTSTM Register Safe Endinit

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner and in some PROT states, the secure master to update the PROT state as described in the PROT mechanism.

The PROTSTMSE register controls lock / unlock of the local Safety Endinit (SE) protected control registers associated with the STM resource.

PROTSTMSE Offset address: 0E0D8$_H$

CPUx PROTSTM Register Safe Endinit Reset values see: Table 169

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OWEN	ODEF	TAGID						PRSEN	PRS			VMEN	VM		
w	rw	rw						rw	rw			rw	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0					SEL			0				SWEN	STATE		
r					rw			r				w	rwh		

Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write

(table continues...)

(continued)

Field	Bits	Type	Description
SEL	10:8	rw	**Range selection** Returns the number of the range being configured when read. Can be written by the PROT owner or CS master to select the range number to be configured if the state transition is valid in the PROT state diagram. 000_B **Const0**: Range 0 selected for configuration … 111_B **Const7**: Range 7 selected for configuration
VM	18:16	rw	**Virtual machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual machine definition enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.
PRSEN	23	rw	**Protection set definition enable for PROT owner** 0_B **Disable**: PRS is part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	7:4, 15:11	r	**Reserved** Read as 0; should be written with 0.

Table 168 Access mode restrictions of PROTSTMSE sorted by descending priority

Mode name	Access mode		Description
write 1 to .**SWEN**	rwh	STATE	
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

Table 169 Reset values of PROTSTMSE

Reset type	Reset value	Note
Application Reset	0000 0000$_H$	Instance specific: Any other CPU
Application Reset	4C00 0000$_H$	Instance specific: CPUcs

4.6.1.134 CPUx shadow STM, write access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

ACCENSTMCFG_WRA Offset address: 0E0E0$_H$

CPUx shadow STM, write access enable register A Reset values see: Table 170

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access

Table 170 Reset values of ACCENSTMCFG_WRA

Reset type	Reset value	Note
Application Reset	1000 0003$_H$	Instance specific: Any other CPU
Application Reset	0000 1000$_H$	Instance specific: CPUcs

4.6.1.135 CPUx shadow STM, write access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

ACCENSTMCFG_WRB Offset address: 0E0E4$_H$

CPUx shadow STM, write access enable register B Application Reset value: 0000 0000$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN63	EN62	EN61	EN60	EN59	EN58	EN57	EN56	EN55	EN54	EN53	EN52	EN51	EN50	EN49	EN48
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN47	EN46	EN45	EN44	EN43	EN42	EN41	EN40	EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-63)	q-32	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access

4.6.1.136 CPUx shadow STM, read access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

ACCENSTMCFG_RDA Offset address: $0E0E8_H$

CPUx shadow STM, read access enable register A Reset values see: Table 171

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access

Table 171 Reset values of ACCENSTMCFG_RDA

Reset type	Reset value	Note
Application Reset	FFFF $FFFF_H$	Instance specific: Any other CPU
Application Reset	0000 1000_H	Instance specific: CPUcs

4.6.1.137 CPUx shadow STM, read access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

ACCENSTMCFG_RDB Offset address: $0E0EC_H$

CPUx shadow STM, read access enable register B Reset values see: Table 172

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN63	EN62	EN61	EN60	EN59	EN58	EN57	EN56	EN55	EN54	EN53	EN52	EN51	EN50	EN49	EN48
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN47	EN46	EN45	EN44	EN43	EN42	EN41	EN40	EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-63)	q-32	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access

Table 172 Reset values of ACCENSTMCFG_RDB

Reset type	Reset value	Note
Application Reset	FFFF FFFF$_H$	Instance specific: Any other CPU
Application Reset	0000 0000$_H$	Instance specific: CPUcs

4.6.1.138 CPUx shadow STM, VM access enable register

This register defines which virtual machine encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each VMx.

ACCENSTMCFG_VM Offset address: 0E0F0$_H$

CPUx shadow STM, VM access enable register Application Reset value: 00FF 00FF$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for VM ID q** This bit enables read access to the access protected region for transactions with the VM ID q. 0_B Disabled for read access 1_B Enabled for read access

(table continues...)

(continued)

Field	Bits	Type	Description
WRq (q=00-07)	q+16	rw	**Write access enable for VM ID q** This bit enables write access to the access protected region for transactions with the VM ID q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

4.6.1.139 CPUx shadow STM, PRS access enable register

This register defines which protection register set encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each PRSx.

ACCENSTMCFG_PRS Offset address: $0E0F4_H$

CPUx shadow STM, PRS access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for PRS q** This bit enables read access to the access protected region for transactions with the PRS q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for PRS q** This bit enables write access to the access protected region for transactions with the PRS q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

4.6.1.140 CPUx SPR range y, write access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

ACCENSPRy_WRA (y=0-15) Offset address: $0E200_H+y*20_H$

CPUx SPR range y, write access enable register A Reset values see: Table 173

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
ENq (q=00-31)	q	rh	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access

Table 173 Reset values of ACCENSPRy_WRA (y=0-15)

Reset type	Reset value	Note
Application Reset	$1000\ 0003_H$	Instance specific: Any other CPU
Application Reset	$0000\ 1000_H$	Instance specific: CPUcs, CSRM debug disabled
After Boot-FW Value	$1000\ 3000_H$	CPUcs, y=0, CSRM debug enabled

4.6.1.141 CPUx SPR range y, write access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

ACCENSPRy_WRB (y=0-15) Offset address: $0E204_H+y*20_H$

CPUx SPR range y, write access enable register B Reset values see: Table 174

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN63	EN62	EN61	EN60	EN59	EN58	EN57	EN56	EN55	EN54	EN53	EN52	EN51	EN50	EN49	EN48
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN47	EN46	EN45	EN44	EN43	EN42	EN41	EN40	EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
ENq (q=32-63)	q-32	rh	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access

Table 174 Reset values of ACCENSPRy_WRB (y=0-15)

Reset type	Reset value	Note
Application Reset	0000 0000$_H$	Instance specific: Any other CPU
Application Reset	0000 0000$_H$	Instance specific: CPUcs, CSRM debug disabled
After Boot-FW Value	0000 4000$_H$	CPUcs, y=0, CSRM debug enabled

4.6.1.142 CPUx SPR range y, read access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

ACCENSPRy_RDA (y=0-15) Offset address: 0E208$_H$+y*20$_H$

CPUx SPR range y, read access enable register A Reset values see: Table 175

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
ENq (q=00-31)	q	rh	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access

Table 175 Reset values of ACCENSPRy_RDA (y=0-15)

Reset type	Reset value	Note
Application Reset	FFFF FFFF$_H$	Instance specific: Any other CPU
Application Reset	0000 1000$_H$	Instance specific: CPUcs, CSRM debug disabled
After Boot-FW Value	1000 3000$_H$	CPUcs, y=0, CSRM debug enabled

4.6.1.143 CPUx SPR range y, read access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

ACCENSPRy_RDB (y=0-15) Offset address: 0E20C$_H$+y*20$_H$

CPUx SPR range y, read access enable register B Reset values see: Table 176

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN63	EN62	EN61	EN60	EN59	EN58	EN57	EN56	EN55	EN54	EN53	EN52	EN51	EN50	EN49	EN48
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN47	EN46	EN45	EN44	EN43	EN42	EN41	EN40	EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
ENq (q=32-63)	q-32	rh	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access

Table 176 Reset values of ACCENSPRy_RDB (y=0-15)

Reset type	Reset value	Note
Application Reset	FFFF FFFF$_H$	Instance specific: Any other CPU
Application Reset	0000 4000$_H$	Instance specific: CPUcs

4.6.1.144 CPUx SPR range y, VM access enable register

This register defines which virtual machine encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each VMx.

ACCENSPRy_VM (y=0-15) Offset address: 0E210$_H$+y*20$_H$

CPUx SPR range y, VM access enable register Application Reset value: 00FF 00FF$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
RDq (q=00-07)	q	rh	**Read access enable for VM ID q** This bit enables read access to the access protected region for transactions with the VM ID q. 0_B Disabled for read access 1_B Enabled for read access

(table continues...)

(continued)

Field	Bits	Type	Description
WRq (q=00-07)	q+16	rh	**Write access enable for VM ID q** This bit enables write access to the access protected region for transactions with the VM ID q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

4.6.1.145 CPUx SPR range y, PRS access enable register

This register defines which protection register set encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each PRSx.

ACCENSPRy_PRS (y=0-15) Offset address: $0E214_H+y*20_H$

CPUx SPR range y, PRS access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rh	rh	rh	rh	rh	rh	rh	rh

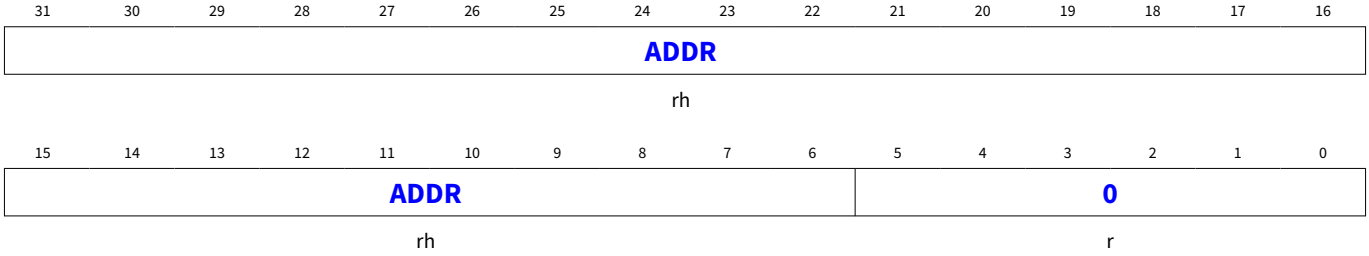
Field	Bits	Type	Description
RDq (q=00-07)	q	rh	**Read access enable for PRS q** This bit enables read access to the access protected region for transactions with the PRS q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rh	**Write access enable for PRS q** This bit enables write access to the access protected region for transactions with the PRS q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

4.6.1.146 CPUx SPR range y, region lower address register

This register sets the lower bound of the access protected region. Addresses greater than or equal to the lower bound are considered part of the access protected region.

ACCENSPRy_RGNLA (y=0-15) Offset address: $0E218_H+y*20_H$

CPUx SPR range y, region lower address register Application Reset value: $0000\ 0000_H$



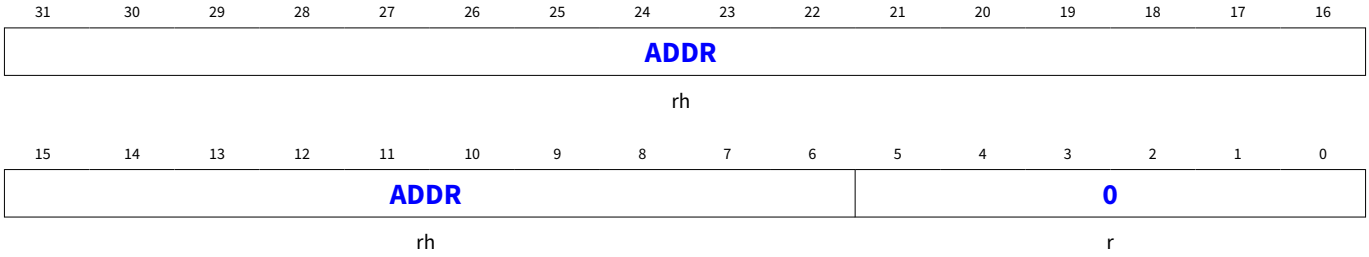
Field	Bits	Type	Description
ADDR	31:6	rh	**Defines bits 31:6 of the lower bound of the access protected region**
0	5:0	r	**Reserved** Read as 0; should be written with 0.

4.6.1.147 CPUx SPR range y, region upper address register

This register sets the upper bound of the access protected region. Addresses stricly less than the upper bound are considered part of the access protected region.

ACCENSPRy_RGNUA (y=0-15) Offset address: $0E21C_H+y*20_H$

CPUx SPR range y, region upper address register Application Reset value: $FFFF\ FFC0_H$



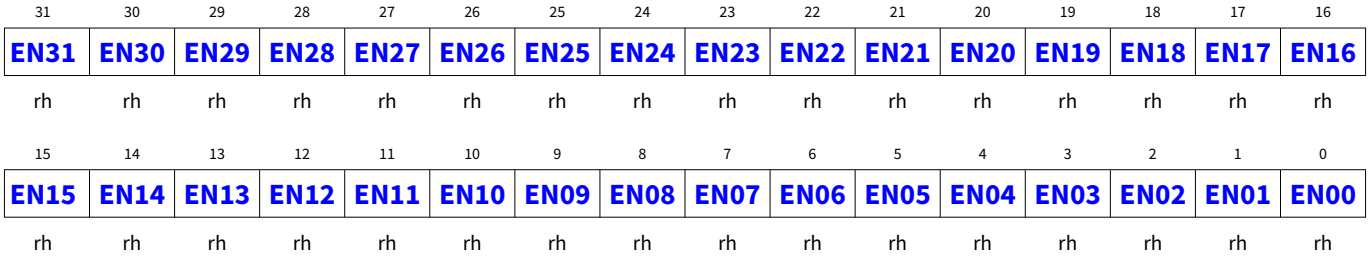
Field	Bits	Type	Description
ADDR	31:6	rh	**Defines bits 31:6 of the upper bound of the access protected region**
0	5:0	r	**Reserved** Read as 0; should be written with 0.

4.6.1.148 CPUx DLMU range y, write access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

ACCENDLMUy_WRA (y=0-15) Offset address: $0E400_H+y*20_H$

CPUx DLMU range y, write access enable register A Application Reset value: $1000\ 0003_H$



Field	Bits	Type	Description
ENq (q=00-31)	q	rh	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access

4.6.1.149 CPUx DLMU range y, write access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

ACCENDLMUy_WRB (y=0-15) Offset address: $0E404_H+y*20_H$

CPUx DLMU range y, write access enable register B Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN63	EN62	EN61	EN60	EN59	EN58	EN57	EN56	EN55	EN54	EN53	EN52	EN51	EN50	EN49	EN48
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN47	EN46	EN45	EN44	EN43	EN42	EN41	EN40	EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
ENq (q=32-63)	q-32	rh	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access

4.6.1.150 CPUx DLMU range y, read access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

ACCENDLMUy_RDA (y=0-15) Offset address: $0E408_H+y*20_H$

CPUx DLMU range y, read access enable register A Application Reset value: $FFFF\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
ENq (q=00-31)	q	rh	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access

4.6.1.151 CPUx DLMU range y, read access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

ACCENDLMUy_RDB (y=0-15) Offset address: $0E40C_H+y*20_H$

CPUx DLMU range y, read access enable register B Application Reset value: FFFF FFFF$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN63	EN62	EN61	EN60	EN59	EN58	EN57	EN56	EN55	EN54	EN53	EN52	EN51	EN50	EN49	EN48
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN47	EN46	EN45	EN44	EN43	EN42	EN41	EN40	EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
ENq (q=32-63)	q-32	rh	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access

4.6.1.152 CPUx DLMU range y, VM access enable register

This register defines which virtual machine encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each VMx.

ACCENDLMUy_VM (y=0-15) Offset address: $0E410_H+y*20_H$

CPUx DLMU range y, VM access enable register Application Reset value: 00FF 00FF$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rh	rh	rh	rh	rh	rh	rh	rh
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
RDq (q=00-07)	q	rh	**Read access enable for VM ID q** This bit enables read access to the access protected region for transactions with the VM ID q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rh	**Write access enable for VM ID q** This bit enables write access to the access protected region for transactions with the VM ID q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

4.6.1.153 CPUx DLMU range y, PRS access enable register

This register defines which protection register set encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each PRSx.

ACCENDLMUy_PRS (y=0-15) Offset address: $0E414_H+y*20_H$

CPUx DLMU range y, PRS access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
RDq (q=00-07)	q	rh	**Read access enable for PRS q** This bit enables read access to the access protected region for transactions with the PRS q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rh	**Write access enable for PRS q** This bit enables write access to the access protected region for transactions with the PRS q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

4.6.1.154 CPUx DLMU range y, region lower address register

This register sets the lower bound of the access protected region. Addresses greater than or equal to the lower bound are considered part of the access protected region.

ACCENDLMUy_RGNLA (y=0-15) Offset address: $0E418_H+y*20_H$

CPUx DLMU range y, region lower address register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADDRH		0	ADDR												
rh		r	rh												

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR										0					
rh										r					

Field	Bits	Type	Description
ADDR	28:6	rh	**Defines bits 28:6 of the lower bound of the access protected region**
ADDRH	31:30	rh	**Defines bits 31:30 of the lower bound of the access protected region**
0	5:0, 29	r	**Reserved** Read as 0; should be written with 0.

4.6.1.155 CPUx DLMU range y, region upper address register

This register sets the upper bound of the access protected region. Addresses stricly less than the upper bound are considered part of the access protected region.

ACCENDLMUy_RGNUA (y=0-15) Offset address: $0E41C_H+y*20_H$

CPUx DLMU range y, region upper address register Application Reset value: $DFFF\ FFC0_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADDRH		0	ADDR												
rh		r	rh												

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR										0					
rh										r					

Field	Bits	Type	Description
ADDR	28:6	rh	**Defines bits 28:6 of the upper bound of the access protected region**
ADDRH	31:30	rh	**Defines bits 31:30 of the upper bound of the access protected region**
0	5:0, 29	r	**Reserved** Read as 0; should be written with 0.

4.6.1.156 CPUx SFR, write access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

ACCENSFR_WRA Offset address: $0E600_H$

CPUx SFR, write access enable register A Reset values see: Table 177

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
ENq (q=00-31)	q	rh	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access

Table 177 Reset values of ACCENSFR_WRA

Reset type	Reset value	Note
Application Reset	$1000\ 0003_H$	Instance specific: Any other CPU
Application Reset	$0000\ 1000_H$	Instance specific: CPUcs, CSRM debug disabled
After Boot-FW Value	$1000\ 3000_H$	CPUcs, CSRM debug enabled

4.6.1.157 CPUx SFR, write access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

ACCENSFR_WRB Offset address: $0E604_H$

CPUx SFR, write access enable register B Reset values see: Table 178

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN63	EN62	EN61	EN60	EN59	EN58	EN57	EN56	EN55	EN54	EN53	EN52	EN51	EN50	EN49	EN48
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN47	EN46	EN45	EN44	EN43	EN42	EN41	EN40	EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
ENq (q=32-63)	q-32	rh	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access

Table 178 Reset values of ACCENSFR_WRB

Reset type	Reset value	Note
Application Reset	0000 0000$_H$	Instance specific: Any other CPU
Application Reset	0000 0000$_H$	Instance specific: CPUcs, CSRM debug disabled
After Boot-FW Value	0000 4000$_H$	CPUcs, CSRM debug enabled

4.6.1.158 CPUx SFR, read access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

ACCENSFR_RDA Offset address: 0E608$_H$

CPUx SFR, read access enable register A Reset values see: Table 179

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
ENq (q=00-31)	q	rh	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access

Table 179 Reset values of ACCENSFR_RDA

Reset type	Reset value	Note
Application Reset	FFFF FFFF$_H$	Instance specific: Any other CPU
Application Reset	0000 1000$_H$	Instance specific: CPUcs, CSRM debug disabled
After Boot-FW Value	1000 3000$_H$	CPUcs, CSRM debug enabled

4.6.1.159 CPUx SFR, read access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

ACCENSFR_RDB Offset address: $0E60C_H$

CPUx SFR, read access enable register B Reset values see: Table 180

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN63	EN62	EN61	EN60	EN59	EN58	EN57	EN56	EN55	EN54	EN53	EN52	EN51	EN50	EN49	EN48
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN47	EN46	EN45	EN44	EN43	EN42	EN41	EN40	EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
ENq (q=32-63)	q-32	rh	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access

Table 180 Reset values of ACCENSFR_RDB

Reset type	Reset value	Note
Application Reset	FFFF FFFF$_H$	Instance specific: Any other CPU
Application Reset	0000 0000$_H$	Instance specific: CPUcs, CSRM debug disabled
After Boot-FW Value	0000 4000$_H$	CPUcs, CSRM debug enabled

4.6.1.160 CPUx SFR, VM access enable register

This register defines which virtual machine encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each VMx.

ACCENSFR_VM Offset address: $0E610_H$

CPUx SFR, VM access enable register Application Reset value: 00FF 00FF$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
RDq (q=00-07)	q	rh	**Read access enable for VM ID q** This bit enables read access to the access protected region for transactions with the VM ID q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rh	**Write access enable for VM ID q** This bit enables write access to the access protected region for transactions with the VM ID q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

4.6.1.161 CPUx SFR, PRS access enable register

This register defines which protection register set encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each PRSx.

ACCENSFR_PRS Offset address: $0E614_H$

CPUx SFR, PRS access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
RDq (q=00-07)	q	rh	**Read access enable for PRS q** This bit enables read access to the access protected region for transactions with the PRS q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rh	**Write access enable for PRS q** This bit enables write access to the access protected region for transactions with the PRS q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

4.6.1.162 CPUx VMy STM, write access enable register A, virtualization view

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

ACCENSTMVMy_WRA (y=0-7) Offset address: $0E700_H+y*20_H$

CPUx VMy STM, write access enable register A, virtualization view Reset values see: Table 181

ACCENSTM_WRA Offset address: $0E720_H$

CPUx STM, write access enable register A Reset values see: Table 181

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
ENq (q=00-31)	q	rh	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access

Table 181 Reset values

Applies to ACCENSTMVMy_WRA (y=0-7)

Applies to ACCENSTM_WRA

Reset type	Reset value	Note
Application Reset	$1000\ 0003_H$	Instance specific: Any other CPU
Application Reset	$0000\ 1000_H$	Instance specific: CPUcs

4.6.1.163 CPUx VMy STM, write access enable register B, virtualization view

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

ACCENSTMVMy_WRB (y=0-7) Offset address: $0E704_H+y*20_H$

CPUx VMy STM, write access enable register B, virtualization view Application Reset value: $0000\ 0000_H$

ACCENSTM_WRB Offset address: $0E724_H$

CPUx STM, write access enable register B Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN63	EN62	EN61	EN60	EN59	EN58	EN57	EN56	EN55	EN54	EN53	EN52	EN51	EN50	EN49	EN48
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN47	EN46	EN45	EN44	EN43	EN42	EN41	EN40	EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
ENq (q=32-63)	q-32	rh	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access

4.6.1.164 CPUx VMy STM, read access enable register A, virtualization view

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

ACCENSTMVMy_RDA (y=0-7) Offset address: $0E708_H+y*20_H$

CPUx VMy STM, read access enable register A, virtualization view Reset values see: Table 182

ACCENSTM_RDA Offset address: $0E728_H$

CPUx STM, read access enable register A Reset values see: Table 182

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
ENq (q=00-31)	q	rh	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access

Table 182 Reset values

Applies to ACCENSTMVMy_RDA (y=0-7)

Applies to ACCENSTM_RDA

Reset type	Reset value	Note
Application Reset	$FFFF\ FFFF_H$	Instance specific: Any other CPU

(table continues...)

Table 182 (continued) Reset values

Applies to ACCENSTMVMy_RDA (y=0-7)

Applies to ACCENSTM_RDA

Reset type	Reset value	Note
Application Reset	0000 1000$_H$	Instance specific: CPUcs

4.6.1.165 CPUx VMy STM, read access enable register B, virtualization view

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

ACCENSTMVMy_RDB (y=0-7) Offset address: 0E70C$_H$+y*20$_H$

CPUx VMy STM, read access enable register B, virtualization view Reset values see: Table 183

ACCENSTM_RDB Offset address: 0E72C$_H$

CPUx STM, read access enable register B Reset values see: Table 183

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN63	EN62	EN61	EN60	EN59	EN58	EN57	EN56	EN55	EN54	EN53	EN52	EN51	EN50	EN49	EN48
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN47	EN46	EN45	EN44	EN43	EN42	EN41	EN40	EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
ENq (q=32-63)	q-32	rh	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access

Table 183 Reset values

Applies to ACCENSTMVMy_RDB (y=0-7)

Applies to ACCENSTM_RDB

Reset type	Reset value	Note
Application Reset	FFFF FFFF$_H$	Instance specific: Any other CPU
Application Reset	0000 0000$_H$	Instance specific: CPUcs

4.6.1.166 CPUx VMy STM, VM access enable register, virtualization view

This register defines which virtual machine encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each VMx.

ACCENSTMVMy_VM (y=0-7) Offset address: $0E710_H+y*20_H$

CPUx VMy STM, VM access enable register, virtualization view Application Reset value: $00FF\ 00FF_H$

ACCENSTM_VM Offset address: $0E730_H$

CPUx STM, VM access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
RDq (q=00-07)	q	rh	**Read access enable for VM ID q** This bit enables read access to the access protected region for transactions with the VM ID q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rh	**Write access enable for VM ID q** This bit enables write access to the access protected region for transactions with the VM ID q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

4.6.1.167 CPUx VMy STM, PRS access enable register, virtualization view

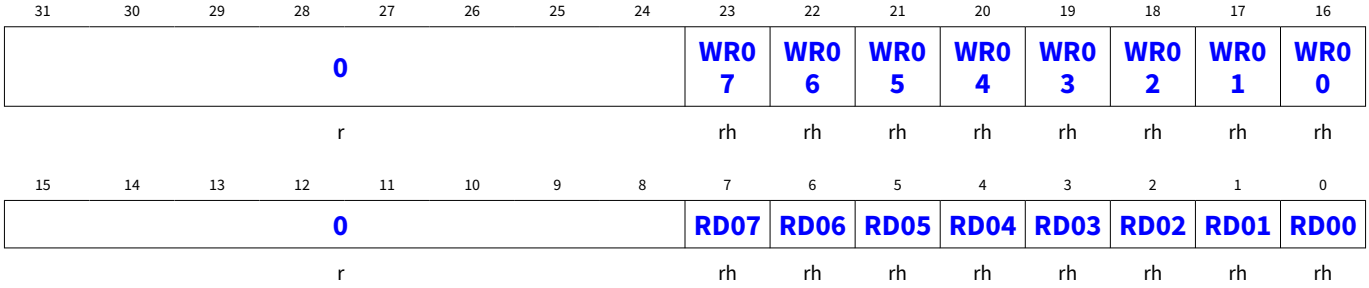
This register defines which protection register set encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each PRSx.

ACCENSTMVMy_PRS (y=0-7) Offset address: $0E714_H+y*20_H$

CPUx VMy STM, PRS access enable register, virtualization view Application Reset value: $00FF\ 00FF_H$

ACCENSTM_PRS Offset address: $0E734_H$

CPUx STM, PRS access enable register Application Reset value: $00FF\ 00FF_H$

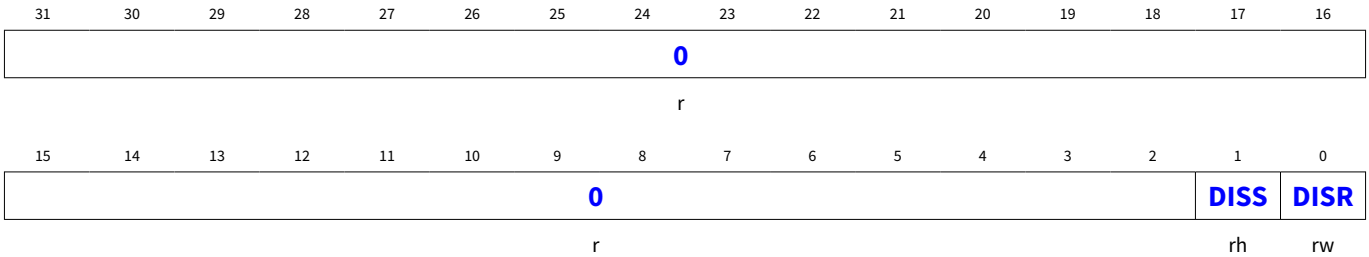


Field	Bits	Type	Description
RDq (q=00-07)	q	rh	**Read access enable for PRS q** This bit enables read access to the access protected region for transactions with the PRS q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rh	**Write access enable for PRS q** This bit enables write access to the access protected region for transactions with the PRS q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

4.6.1.168 CPUx system timer clock control register

The STM clock control register is used to switch the STM on or off and to control its input clock rate. After reset, the STM is always enabled and starts counting. The STM can be disabled by setting bit DISR to 1.

STM_CLC	Offset address:	00000_H
CPUx system timer clock control register	System Reset value:	$0000\ 0000_H$
STMHV_CLC	Offset address:	00000_H
CPUx system timer clock control register, hypervisor view	System Reset value:	$0000\ 0000_H$



Field	Bits	Type	Description
DISR	0	rw	**Module disable request bit** Used for enable and disable control of the STM module. ***Note:*** *f_{STM} is generated by the CCU.* 0_B On request: enable the module clock 1_B Off request: stop the module clock
DISS	1	rh	**Module disable status bit** When STM_CLC.DISS=1_H, registers STM_CLC, PROTSTMSE, and ACCENSTMCFG are still accessible. Accesses to other STM registers are not executed. 0_B STM module is enabled 1_B STM module is disabled
0	31:2	r	**Reserved** Read as 0; should be written with 0.

4.6.1.169 CPUx OCDS control and status register

The OCDS control and status register OCS controls the debug behavior by selecting suspend modes.

Register OCS is cleared by Debug Reset. It can only be written when Debug is enabled.

If Debug is being disabled, the OCS register value will not change. When Debug is disabled the OCS suspend control is ineffective.

If the virtualization extension is not present or disabled:

- Debug is enabled when both DBGSR.DE==1 and DBGCFG.EN==1

If the virtualization extension is present and enabled:

- Debug is enabled when DBGSR.DE==1, DBGCFG.EN==1, and DBGCFG.VMn==1 for the currently active virtual machine

STM_OCS Offset address: 00004_H

CPUx OCDS control and status register Debug Reset value: $0000\ 0000_H$

STMHV_OCS Offset address: 00004_H

CPUx OCDS control and status register, hypervisor view Debug Reset value: $0000\ 0000_H$

31–30	29	28	27–26–25–24	23–22–21–20–19–18–17–16
0	SUSSTA	SUS_P	SUS	0
r	rh	w	rw	r

15–14–13–12–11–10–9–8–7–6–5–4–3–2	1	0
0	0	
r	r	

Field	Bits	Type	Description
SUS	27:24	rw	**OCDS suspend control** Controls the sensitivity of STMx to the suspend signal coming from CPUx (CPUxSUSOUT) 0_H Will not suspend 1_H 64-bit counter will be stopped. Counter value does not change when registers are read 2_H 64-bit counter will be stopped. Counter value does not change when registers are read (backward compatiblity) **others,** Reserved, do not use this combination
SUS_P	28	w	**SUS write protection** SUS is only written when SUS_P is 1, otherwise unchanged. Read as 0.
SUSSTA	29	rh	**Suspend state** 0_B Module is not (yet) suspended 1_B Module is suspended
0	2:0, 23:3, 31:30	r	**Reserved** Read as 0; must be written with 0.

Table 184 Access mode restrictions sorted by descending priority

Applies to STM_OCS

Applies to STMHV_OCS

Mode name	Access mode		Description
write 1 to .**SUS_P**	rw	SUS	
(default)	r	SUS	

4.6.1.170 CPUx system timer absolute timer

STM_ABS Offset address: 00020_H

CPUx system timer absolute timer System Reset value: 0000 0000 0000 0000_H

STMHV_ABS Offset address: 00020_H

CPUx system timer absolute timer, hypervisor view System Reset value: 0000 0000 0000 0000_H

63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48
ABSVAL															
rwh															

47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
ABSVAL															
rwh															

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ABSVAL															
rwh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ABSVAL															
rwh															

Field	Bits	Type	Description
ABSVAL	63:0	rwh	**System timer bits [63:0]** This bit field contains the absolute value of the 64-bit STM.

4.6.1.171 CPUx system timer compare register c

STM_CMPc (c=0-1) Offset address: 00120_H+c*4

CPUx system timer compare register c System Reset value: $0000\ 0000_H$

STMHV_VMn_CMPc (c=0-1;n=0-7) Offset address: 00100_H+n*20_H+c*4

CPUx VMn system timer compare register c, hypervisor view System Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CMPVAL															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CMPVAL															
rw															

Field	Bits	Type	Description
CMPVAL	31:0	rw	**Compare value of compare register cmp** This bit field holds up to 32 bits of the compare value (right-adjusted).

4.6.1.172 CPUx system timer compare match control register

The STM Compare Match Control Register controls the parameters of the compare logic.

STM_CMCON — Offset address: 00128_H

CPUx system timer compare match control register — System Reset value: $0000\ 0000_H$

STMHV_VMn_CMCON (n=0-7) — Offset address: 00108_H+n*20_H

CPUx VMn system timer compare match control register, hypervisor view — System Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RELCOMP1	0		MSTART1					0			MSIZE1				
rw	r		rw					r			rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RELCOMP0	0		MSTART0					0			MSIZE0				
rw	r		rw					r			rw				

Field	Bits	Type	Description
MSIZE0	4:0	rw	**Compare register size for CMP0** This bit field determines the number of bits in register CMP0 (starting from bit 0) that are used for the compare operation with the System Timer. 00_H CMP0[0] used for compare operation 01_H CMP0[1:0] used for compare operation 10_H ... $1E_H$ CMP0[30:0] used for compare operation $1F_H$ CMP0[31:0] used for compare operation
MSTART0	12:8	rw	**Start bit location for CMP0** This bit field determines the lowest bit number of the 64-bit STM that is compared with the content of register CMP0 bit 0. The number of bits to be compared is defined by bit field MSIZE0. 00_H STM[0] is the lowest bit number 01_H STM[1] is the lowest bit number 10_H ... $1E_H$ STM[30] is the lowest bit number $1F_H$ STM[31] is the lowest bit number
RELCOMP0	15	rw	**Relative timer comparison** CMP0 may be selected to compare against either the absolute or the relative timer. A comparator that is set to compare absolute time is constantly active. A comparator that is set to compare relative time is only active when its associated VM is active. 0_B Compare against absolute timer 1_B Compare against relative timer

(table continues...)

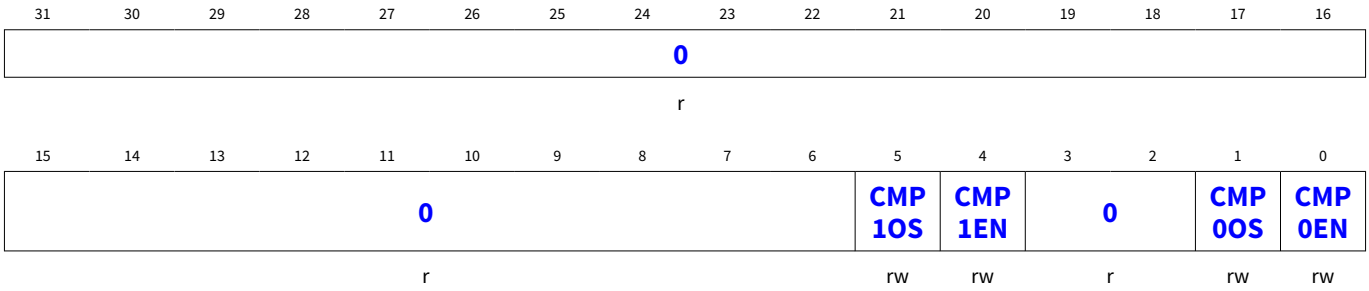
(continued)

Field	Bits	Type	Description
MSIZE1	20:16	rw	**Compare register size for CMP1** This bit field determines the number of bits in register CMP1 (starting from bit 0) that are used for the compare operation with the System Timer. 00_H CMP1[0] used for compare operation 01_H CMP1[1:0] used for compare operation 10_H ... $1E_H$ CMP1[30:0] used for compare operation $1F_H$ CMP1[31:0] used for compare operation
MSTART1	28:24	rw	**Start bit location for CMP1** This bit field determines the lowest bit number of the 64-bit STM that is compared with the content of register CMP1 bit 0. The number of bits to be compared is defined by bit field MSIZE1. 00_H STM[0] is the lowest bit number 01_H STM[1] is the lowest bit number 10_H ... $1E_H$ STM[30] is the lowest bit number $1F_H$ STM[31] is the lowest bit number
RELCOMP1	31	rw	**Relative timer comparison** CMP0 may be selected to compare against either the absolute or the relative timer. A comparator that is set to compare absolute time is constantly active. A comparator that is set to compare relative time is only active when its associated VM is active. 0_B Compare against absolute timer 1_B Compare against relative timer
0	7:5, 14:13, 23:21, 30:29	r	**Reserved** Read as 0; should be written with 0.

4.6.1.173 CPUx system timer interrupt control register

The two compare match interrupts of the STM are controlled by the STM Interrupt Control Register.

STM_ICR	Offset address:	$0012C_H$
CPUx system timer interrupt control register	System Reset value:	$0000\ 0000_H$
STMHV_VMn_ICR (n=0-7)	Offset address:	$0010C_H+n*20_H$
CPUx VMn system timer interrupt control register, hypervisor view	System Reset value:	$0000\ 0000_H$



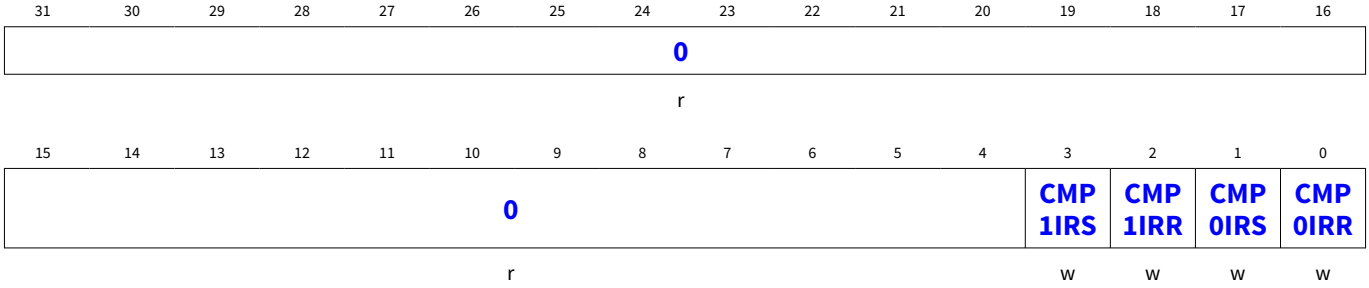
Field	Bits	Type	Description
CMP0EN	0	rw	**Compare register CMP0 interrupt enable control** This bit enables the compare match interrupt with compare register CMP0. 0_B Interrupt on compare match with CMP0 disabled 1_B Interrupt on compare match with CMP0 enabled
CMP0OS	1	rw	**Compare register CMP0 interrupt output selection** This bit determines the interrupt output that is activated on a compare match event of compare register CMP0. 0_B Interrupt output STMIR0 selected 1_B Interrupt output STMIR1 selected
CMP1EN	4	rw	**Compare register CMP1 interrupt enable control** This bit enables the compare match interrupt with compare register CMP1. 0_B Interrupt on compare match with CMP1 disabled 1_B Interrupt on compare match with CMP1 enabled
CMP1OS	5	rw	**Compare register CMP1 interrupt output selection** This bit determines the interrupt output that is activated on a compare match event of compare register CMP1. 0_B Interrupt output STMIR0 selected 1_B Interrupt output STMIR1 selected
0	3:2, 31:6	r	**Reserved** Read as 0; should be written with 0.

4.6.1.174 CPUx system timer interrupt set/clear register

The bits in the STM Interrupt Set/Clear Register make it possible to set or cleared the compare match interrupt request status flags of register ICR.

***Note**: Reading register ISCR always returns 0000 0000$_H$.*

STM_ISCR	Offset address:	00130_H
CPUx system timer interrupt set/clear register	System Reset value:	$0000\ 0000_H$
STMHV_VMn_ISCR (n=0-7)	Offset address:	00110_H+n*20_H
CPUx VMn system timer interrupt set/clear register, hypervisor view	System Reset value:	$0000\ 0000_H$



Field	Bits	Type	Description
CMP0IRR	0	w	**Reset compare register CMP0 interrupt flag** 0_B Bit ISR.CMP0IR is not changed 1_B Bit ISR.CMP0IR is cleared
CMP0IRS	1	w	**Set compare register CMP0 interrupt flag** 0_B Bit ISR.CMP0IR is not changed 1_B Bit ISR.CMP0IR is set. The state of bit CMP0IRR is "don't care" in this case
CMP1IRR	2	w	**Reset compare register CMP1 interrupt flag** 0_B Bit ISR.CMP1IR is not changed 1_B Bit ISR.CMP1IR is cleared.
CMP1IRS	3	w	**Set compare register CMP1 interrupt flag** 0_B Bit ISR.CMP1IR is not changed 1_B Bit ISR.CMP1IR is set. The state of bit CMP1IRR is "don't care" in this case
0	31:4	r	**Reserved** Read as 0; should be written with 0.

4.6.1.175 CPUx system timer interrupt status register

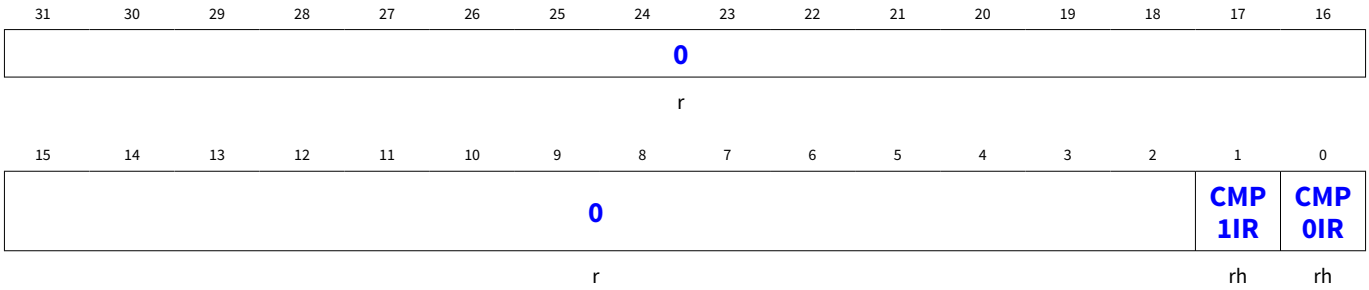
The status of the two compare match interrupts of the STM is provided by the STM Interrupt Status Register.

STM_ISR — Offset address: 00134_H

CPUx system timer interrupt status register — System Reset value: $0000\ 0000_H$

STMHV_VMn_ISR (n=0-7) — Offset address: 00114_H+n*20_H

CPUx VMn system timer interrupt status register, hypervisor view — System Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
CMP0IR	0	rh	**Compare register CMP0 interrupt request flag** This bit indicates whether or not a compare match event of compare register CMP0 has occured. CMP0IR can be cleared by software and can be set by software, too (see ISCR register). After a STM reset operation, CMP0IR is immediately set as a result of a compare match event with the reset values of the STM and the compare register CMP0. ***Note**: This flag does not gate the interrupt generation. i.e., even if this flag is not cleared, compare match event interrupts are forwarded if CMPxEN is set.* 0_B A compare match event has not been detected since the bit was last cleared 1_B A compare match event has been detected
CMP1IR	1	rh	**Compare register CMP1 interrupt request flag** This bit indicates whether or not a compare match event of compare register CMP1 has occured. CMP1IR can be cleared by software and can be set by software, too (see ISCR register). After a STM reset operation, CMP1IR is immediately set as a result of a compare match event with the reset values of the STM and the compare register CMP1. ***Note**: This flag does not gate the interrupt generation. i.e., even if this flag is not cleared, compare match event interrupts are forwarded if CMPxEN is set.* 0_B A compare match event has not been detected since the bit was last cleared 1_B A compare match event has been detected
0	31:2	r	**Reserved** Read as 0; should be written with 0.

4.6.1.176 CPUx system timer relative timer

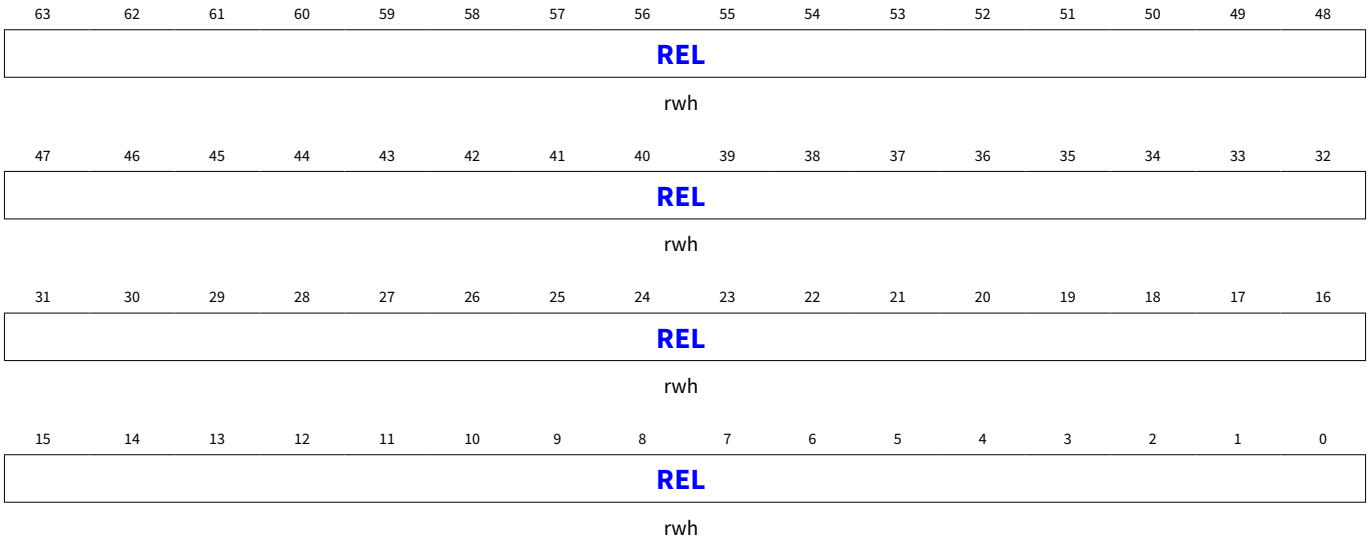
Alias for STM_ABS.

STM_RELTIM — Offset address: 00138_H

CPUx system timer relative timer — System Reset value: 0000 0000 0000 0000_H

STMHV_VMn_RELTIM (n=0-7) — Offset address: 00118_H+n*20_H

CPUx VMn system timer relative timer, hypervisor view — System Reset value: 0000 0000 0000 0000_H



Field	Bits	Type	Description
REL	63:0	rwh	**System timer bits [63:0]** This bit field contains the relative value of the 64-bit STM.

4.6.2 Registers of CPU CFI interface

4.6.2.1 Register overview - access mode glossary

Table 185 Register overview - access mode glossary

Keyword	Description
PFSE	Access protection using PROT register PROTPFSE.
PFSFRE	Access protection using PROT register PROTPFSFRE.
PFSFRSE	Access protection using PROT register PROTPFSFRSE.
APU-P4	Protection group consisting of registers ACCENPFSFRCFG_WRA, ACCENPFSFRCFG_WRB, ACCENPFSFRCFG_RDA, ACCENPFSFRCFG_RDB, ACCENPFSFRCFG_VM, ACCENPFSFRCFG_PRS, ACCENPFSFR_WRA, ACCENPFSFR_WRB, ACCENPFSFR_RDA, ACCENPFSFR_RDB, ACCENPFSFR_VM, ACCENPFSFR_PRS.
P4	Access protection using *APU-P4* registers.
APU-P5	Protection group consisting of registers ACCENPFCFG_RDA, ACCENPFCFG_RDB, ACCENPFCFG_VM, ACCENPFCFG_PRS, ACCENPFCFG_RGNLA, ACCENPFCFG_RGNUA, ACCENPFy_RDA, ACCENPFy_RDB, ACCENPFy_VM, ACCENPFy_PRS, ACCENPFy_RGNLA, ACCENPFy_RGNUA.
P5	Access protection using *APU-P5* registers.
32	Access only when using 32-bit width.
PROT	Access restrictions as defined in the PROT register access rules.

4.6.2.2 Registers overview - CPUCFI (ascending offset address)

Table 186 Registers overview - CPUCFI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PROTPFSFRE	CPUx PROTPFSFR register endinit	00000_H	32	SV, 32, PROT	Application Reset	363
PROTPFSFRSE	CPUx PROTPFSFR register safe endinit	00008_H	32	SV, 32, PROT	Application Reset	364
ACCENPFSFRCFG_WRA	CPUx shadow NVM SFR write access enable register A	00020_H	32	PFSFRSE, SV, 32	See 366	366
ACCENPFSFRCFG_WRB	CPUx shadow NVM SFR write access enable register B	00024_H	32	PFSFRSE, SV, 32	See 367	367
ACCENPFSFRCFG_RDA	CPUx shadow NVM SFR read access enable register A	00028_H	32	PFSFRSE, SV, 32	See 367	367
ACCENPFSFRCFG_RDB	CPUx shadow NVM SFR read access enable register B	$0002C_H$	32	PFSFRSE, SV, 32	See 368	368
ACCENPFSFRCFG_VM	CPUx shadow NVM SFR VM access enable register	00030_H	32	PFSFRSE, SV, 32	Application Reset	369
ACCENPFSFRCFG_PRS	CPUx shadow NVM SFR PRS access enable register	00034_H	32	PFSFRSE, SV, 32	Application Reset	369
PROTPFSE	CPUx PROTPF register safe endinit	00048_H	32	SV, 32, PROT	Application Reset	370
ACCENPFCFG_RDA	CPUx shadow NVM read access enable register A	00068_H	32	PFSE, SV, 32	Application Reset	372
ACCENPFCFG_RDB	CPUx shadow NVM read access enable register B	$0006C_H$	32	PFSE, SV, 32	Application Reset	373
ACCENPFCFG_VM	CPUx shadow NVM VM access enable register	00070_H	32	PFSE, SV, 32	Application Reset	373
ACCENPFCFG_PRS	CPUx shadow NVM PRS access enable register	00074_H	32	PFSE, SV, 32	Application Reset	374
ACCENPFCFG_RGNLA	CPUx shadow NVM region lower address register	00078_H	32	PFSE, SV, 32	Application Reset	374
ACCENPFCFG_RGNUA	CPUx shadow NVM region upper address register	$0007C_H$	32	PFSE, SV, 32	Application Reset	375
ACCENPFSFR_WRA	CPUx NVM SFR write access enable register A	00080_H	32	PFSFRSE, SV, 32	See 375	375
ACCENPFSFR_WRB	CPUx NVM SFR write access enable register B	00084_H	32	PFSFRSE, SV, 32	See 376	376
ACCENPFSFR_RDA	CPUx NVM SFR read access enable register A	00088_H	32	PFSFRSE, SV, 32	See 377	377

(table continues...)

Table 186 (continued) Registers overview - CPUCFI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ACCENPFSFR_RDB	CPUx NVM SFR read access enable register B	$0008C_H$	32	PFSFRSE, SV, 32	See 377	377
ACCENPFSFR_VM	CPUx NVM SFR VM access enable register	00090_H	32	PFSFRSE, SV, 32	Application Reset	378
ACCENPFSFR_PRS	CPUx NVM SFR PRS access enable register	00094_H	32	PFSFRSE, SV, 32	Application Reset	379
ACCENPFy_RDA	CPUx NVM range y read access enable register A	00108_H+ y*20_H	32	PFSE, SV, 32	Application Reset	379
ACCENPFy_RDB	CPUx NVM range y read access enable register B	$0010C_H$+ y*20_H	32	PFSE, SV, 32	Application Reset	380
ACCENPFy_VM	CPUx NVM range y VM access enable register	00110_H+ y*20_H	32	PFSE, SV, 32	Application Reset	381
ACCENPFy_PRS	CPUx NVM range y PRS access enable register	00114_H+ y*20_H	32	PFSE, SV, 32	Application Reset	381
ACCENPFy_RGNLA	CPUx NVM range y region lower address register	00118_H+ y*20_H	32	PFSE, SV, 32	Application Reset	382
ACCENPFy_RGNUA	CPUx NVM range y region upper address register	$0011C_H$+ y*20_H	32	PFSE, SV, 32	Application Reset	382
PFIy_FLASHCON0	CPUx NVM interface y configuration register 0	00200_H+ y*8	P4, 32	PFSFRE, P4, SV, 32	See 383	383
PFIy_FLASHCON1	CPUx NVM interface y configuration register 1	00204_H+ y*8	P4, 32	PFSFRE, P4, SV, 32	Application Reset	384
TS16PNVMy_CTRL0	CPUx NVM interface y MCDS performance counter control register 0	00210_H+ y*8	P4, 32	P4, SV, 32	Debug Reset	384
TS16PNVMy_CTRL1	CPUx NVM interface y MCDS performance counter control register 1	00214_H+ y*8	P4, 32	P4, SV, 32	Debug Reset	385
CFILCLCON	CPUx NVM interface lockstep control register	00240_H	P4, 32	P4, SV, 32	Cold PowerOn Reset	386
CFILCLTEST	CPUx NVM interface lockstep test register	00244_H	P4, 32	P4, SV, 32	Application Reset	387
CFIERR	CPUx CFI error register	00250_H	P4, 32	P4, SV, 32	Application Reset	388
CFIERRAR	CPUx CFI error address register	00254_H	P4, 32	P4, SV, 32	Application Reset	388
PFIBUFDIS	CPUx PFI buffer disable	00320_H	P4, 32	P4, SV, 32	Application Reset	389

4.6.2.3 CPUx PROTPFSFR register endinit

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner and in some PROT states, the secure master to update the PROT state as described in the PROT mechanism.

The PROTPFSFRE register controls lock / unlock of the local Endinit (E) protected control registers.

PROTPFSFRE Offset address: 00000_H

CPUx PROTPFSFR register endinit Reset values see: Table 188

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OWEN	ODEF	TAGID						PRSEN	PRS			VMEN	VM		
w	rw	rw						rw	rw			rw	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												SWEN	STATE		
r												w	rwh		

Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
VM	18:16	rw	**Virtual machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual machine definition enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.

(table continues...)

(continued)

Field	Bits	Type	Description
PRSEN	23	rw	**Protection set definition enable for PROT owner** 0_B **Disable**: PRS is part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; should be written with 0.

Table 187 Access mode restrictions of PROTPFSFRE sorted by descending priority

Mode name	Access mode		Description
write 1 to .**SWEN**	rwh	STATE	
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

Table 188 Reset values of PROTPFSFRE

Reset type	Reset value	Note
Application Reset	0000 0000_H	Instance specific: Non secure NVM
Application Reset	4C00 0000_H	Instance specific: Secure NVM

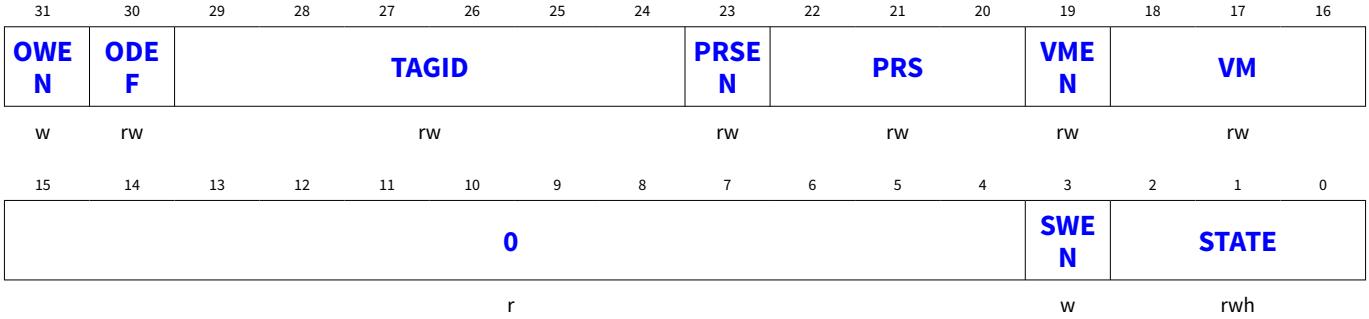
4.6.2.4 CPUx PROTPFSFR register safe endinit

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner and in some PROT states, the secure master to update the PROT state as described in the PROT mechanism.

The PROTPFSFRSE register controls lock / unlock of the local Safe Endinit (SE) protected control registers.

PROTPFSFRSE Offset address: 00008_H

CPUx PROTPFSFR register safe endinit Reset values see: Table 190



Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
VM	18:16	rw	**Virtual machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual machine definition enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.
PRSEN	23	rw	**Protection set definition enable for PROT owner** 0_B **Disable**: PRS is part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner

(table continues...)

(continued)

Field	Bits	Type	Description
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; should be written with 0.

Table 189 Access mode restrictions of PROTPFSFRSE sorted by descending priority

Mode name	Access mode		Description
write 1 to .**SWEN**	rwh	STATE	
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

Table 190 Reset values of PROTPFSFRSE

Reset type	Reset value	Note
Application Reset	0000 0000$_H$	Instance specific: Non secure NVM
Application Reset	4C00 0000$_H$	Instance specific: Secure NVM

4.6.2.5 CPUx shadow NVM SFR write access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

ACCENPFSFRCFG_WRA Offset address: 00020$_H$

CPUx shadow NVM SFR write access enable register A Reset values see: Table 191

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	**EN30**	**EN29**	**EN28**	**EN27**	**EN26**	**EN25**	**EN24**	**EN23**	**EN22**	**EN21**	**EN20**	**EN19**	**EN18**	**EN17**	**EN16**
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	**EN14**	**EN13**	**EN12**	**EN11**	**EN10**	**EN09**	**EN08**	**EN07**	**EN06**	**EN05**	**EN04**	**EN03**	**EN02**	**EN01**	**EN00**
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access

Table 191 Reset values of ACCENPFSFRCFG_WRA

Reset type	Reset value	Note
Application Reset	1000 0003$_H$	Instance specific: Non secure NVM
Application Reset	0000 1000$_H$	Instance specific: Secure NVM, CSRM debug disabled
After Boot-FW Value	1000 3000$_H$	Secure NVM, CSRM debug enabled

4.6.2.6 CPUx shadow NVM SFR write access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

ACCENPFSFRCFG_WRB Offset address: 00024$_H$

CPUx shadow NVM SFR write access enable register B Reset values see: Table 192

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN63	EN62	EN61	EN60	EN59	EN58	EN57	EN56	EN55	EN54	EN53	EN52	EN51	EN50	EN49	EN48
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN47	EN46	EN45	EN44	EN43	EN42	EN41	EN40	EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-63)	q-32	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access

Table 192 Reset values of ACCENPFSFRCFG_WRB

Reset type	Reset value	Note
Application Reset	0000 0000$_H$	Instance specific: Non secure NVM
Application Reset	0000 0000$_H$	Instance specific: Secure NVM, CSRM debug disabled
After Boot-FW Value	0000 4000$_H$	Secure NVM, CSRM debug enabled

4.6.2.7 CPUx shadow NVM SFR read access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

ACCENPFSFRCFG_RDA Offset address: 00028$_H$

CPUx shadow NVM SFR read access enable register A Reset values see: Table 193

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access

Table 193 Reset values of ACCENPFSFRCFG_RDA

Reset type	Reset value	Note
Application Reset	FFFF FFFF$_H$	Instance specific: Non secure NVM
Application Reset	0000 1000$_H$	Instance specific: Secure NVM, CSRM debug enabled
After Boot-FW Value	1000 3000$_H$	Secure NVM, CSRM debug enabled

4.6.2.8 CPUx shadow NVM SFR read access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

ACCENPFSFRCFG_RDB Offset address: 0002C$_H$

CPUx shadow NVM SFR read access enable register B Reset values see: Table 194

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN63	EN62	EN61	EN60	EN59	EN58	EN57	EN56	EN55	EN54	EN53	EN52	EN51	EN50	EN49	EN48
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN47	EN46	EN45	EN44	EN43	EN42	EN41	EN40	EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-63)	q-32	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access

Table 194 Reset values of ACCENPFSFRCFG_RDB

Reset type	Reset value	Note
Application Reset	FFFF FFFF$_H$	Instance specific: Non secure NVM
Application Reset	0000 0000$_H$	Instance specific: Secure NVM, CSRM debug disabled
After Boot-FW Value	0000 4000$_H$	Secure NVM, CSRM debug enabled

4.6.2.9 CPUx shadow NVM SFR VM access enable register

This register defines which virtual machine encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each VMx.

ACCENPFSFRCFG_VM Offset address: 00030$_H$

CPUx shadow NVM SFR VM access enable register Application Reset value: 00FF 00FF$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for VM ID q** This bit enables read access to the access protected region for transactions with the VM ID q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for VM ID q** This bit enables write access to the access protected region for transactions with the VM ID q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

4.6.2.10 CPUx shadow NVM SFR PRS access enable register

This register defines which protection register set encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each PRSx.

ACCENPFSFRCFG_PRS Offset address: 00034$_H$

CPUx shadow NVM SFR PRS access enable register Application Reset value: 00FF 00FF$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for PRS q** This bit enables read access to the access protected region for transactions with the PRS q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for PRS q** This bit enables write access to the access protected region for transactions with the PRS q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

4.6.2.11 CPUx PROTPF register safe endinit

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner and in some PROT states, the secure master to update the PROT state as described in the PROT mechanism.

The PROTPFSE register controls lock / unlock of the local Safe Endinit (SE) protected control registers allocated to the local NVM resources.

PROTPFSE Offset address: 00048_H

CPUx PROTPF register safe endinit Reset values see: Table 196

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OWEN	ODEF	TAGID						PRSEN	PRS			VMEN	VM		
w	rw	rw						rw	rw			rw	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0					SEL			0				SWEN	STATE		
r					rw			r				w	rwh		

Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
SEL	10:8	rw	**Range selection** Returns the number of the range being configured when read. Can be written by the PROT owner or CS master to select the range number to be configured if the state transition is valid in the PROT state diagram. 000_B **Const0**: Range 0 selected for configuration … 111_B **Const7**: Range 7 selected for configuration
VM	18:16	rw	**Virtual machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual machine definition enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.
PRSEN	23	rw	**Protection set definition enable for PROT owner** 0_B **Disable**: PRS is part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner

(table continues...)

(continued)

Field	Bits	Type	Description
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	7:4, 15:11	r	**Reserved** Read as 0; should be written with 0.

Table 195 Access mode restrictions of PROTPFSE sorted by descending priority

Mode name	Access mode		Description
write 1 to .**SWEN**	rwh	STATE	
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

Table 196 Reset values of PROTPFSE

Reset type	Reset value	Note
Application Reset	0000 0000$_H$	Instance specific: Non secure NVM
Application Reset	4C00 0000$_H$	Instance specific: Secure NVM

4.6.2.12 CPUx shadow NVM read access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

ACCENPFCFG_RDA Offset address: 00068$_H$

CPUx shadow NVM read access enable register A Reset values see: Table 197

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	**EN30**	**EN29**	**EN28**	**EN27**	**EN26**	**EN25**	**EN24**	**EN23**	**EN22**	**EN21**	**EN20**	**EN19**	**EN18**	**EN17**	**EN16**
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	**EN14**	**EN13**	**EN12**	**EN11**	**EN10**	**EN09**	**EN08**	**EN07**	**EN06**	**EN05**	**EN04**	**EN03**	**EN02**	**EN01**	**EN00**
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access

Table 197 Reset values of ACCENPFCFG_RDA

Reset type	Reset value	Note
Application Reset	FFFF FFFF$_H$	Instance specific: Non secure NVM
Application Reset	0000 1000$_H$	Instance specific: Secure NVM

4.6.2.13 CPUx shadow NVM read access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

ACCENPFCFG_RDB Offset address: 0006C$_H$

CPUx shadow NVM read access enable register B Reset values see: Table 198

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN63	EN62	EN61	EN60	EN59	EN58	EN57	EN56	EN55	EN54	EN53	EN52	EN51	EN50	EN49	EN48
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN47	EN46	EN45	EN44	EN43	EN42	EN41	EN40	EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-63)	q-32	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access

Table 198 Reset values of ACCENPFCFG_RDB

Reset type	Reset value	Note
Application Reset	FFFF FFFF$_H$	Instance specific: Non secure NVM
Application Reset	0000 4000$_H$	Instance specific: Secure NVM

4.6.2.14 CPUx shadow NVM VM access enable register

This register defines which virtual machine encodings are enabled or disabled for accesses to the access protected region. There is one RDx bit for each VMx.

ACCENPFCFG_VM Offset address: 00070$_H$

CPUx shadow NVM VM access enable register Application Reset value: 0000 00FF$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for VM ID q** This bit enables read access to the access protected region for transactions with the VM ID q. 0_B Disabled for read access 1_B Enabled for read access
0	31:8	r	**Reserved** Read as 0; should be written with 0.

4.6.2.15 CPUx shadow NVM PRS access enable register

This register defines which protection register set encodings are enabled or disabled for accesses to the access protected region. There is one RDx bit for each PRSx.

ACCENPFCFG_PRS Offset address: 00074_H

CPUx shadow NVM PRS access enable register Application Reset value: $0000\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

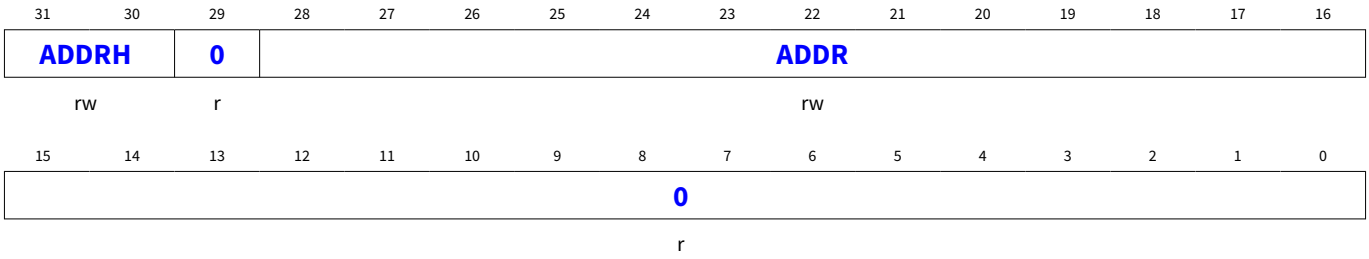
Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for PRS q** This bit enables read access to the access protected region for transactions with the PRS q. 0_B Disabled for read access 1_B Enabled for read access
0	31:8	r	**Reserved** Read as 0; should be written with 0.

4.6.2.16 CPUx shadow NVM region lower address register

This register sets the lower bound of the access protected region. Addresses greater than or equal to the lower bound are considered part of the access protected region.

ACCENPFCFG_RGNLA Offset address: 00078_H

CPUx shadow NVM region lower address register Application Reset value: $0000\ 0000_H$



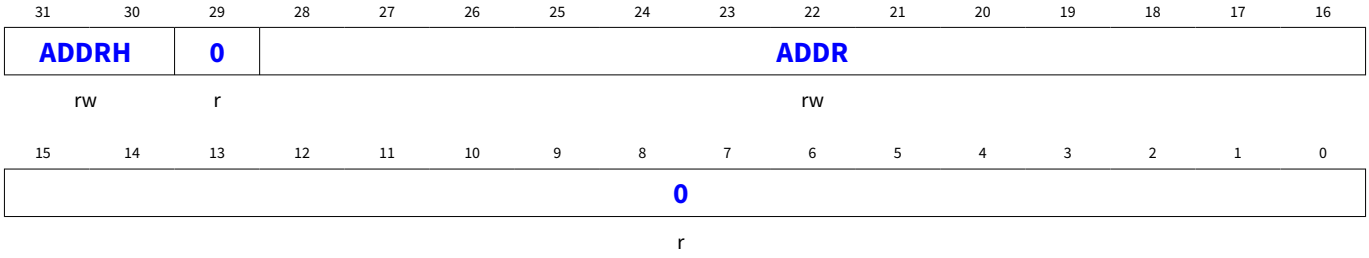
Field	Bits	Type	Description
ADDR	28:16	rw	**Defines bits 28:16 of the lower bound of the access protected region**
ADDRH	31:30	rw	**Defines bits 31:30 of the lower bound of the access protected region**
0	15:0, 29	r	**Reserved** Read as 0; should be written with 0.

4.6.2.17 CPUx shadow NVM region upper address register

This register sets the upper bound of the access protected region. Addresses stricly less than the upper bound are considered part of the access protected region.

ACCENPFCFG_RGNUA Offset address: $0007C_H$

CPUx shadow NVM region upper address register Application Reset value: $DFFF\ 0000_H$



Field	Bits	Type	Description
ADDR	28:16	rw	**Defines bits 28:16 of the upper bound of the access protected region**
ADDRH	31:30	rw	**Defines bits 31:30 of the upper bound of the access protected region**
0	15:0, 29	r	**Reserved** Read as 0; should be written with 0.

4.6.2.18 CPUx NVM SFR write access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

ACCENPFSFR_WRA Offset address: 00080_H

CPUx NVM SFR write access enable register A Reset values see: Table 199

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
ENq (q=00-31)	q	rh	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access

Table 199 Reset values of ACCENPFSFR_WRA

Reset type	Reset value	Note
Application Reset	1000 0003$_H$	Instance specific: Non secure NVM
Application Reset	0000 1000$_H$	Instance specific: Secure NVM, CSRM debug disabled
After Boot-FW Value	1000 3000$_H$	Secure NVM, CSRM debug enabled

4.6.2.19 CPUx NVM SFR write access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

ACCENPFSFR_WRB Offset address: 00084$_H$

CPUx NVM SFR write access enable register B Reset values see: Table 200

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN63	EN62	EN61	EN60	EN59	EN58	EN57	EN56	EN55	EN54	EN53	EN52	EN51	EN50	EN49	EN48
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN47	EN46	EN45	EN44	EN43	EN42	EN41	EN40	EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
ENq (q=32-63)	q-32	rh	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access

Table 200 Reset values of ACCENPFSFR_WRB

Reset type	Reset value	Note
Application Reset	0000 0000$_H$	Instance specific: Non secure NVM
Application Reset	0000 0000$_H$	Instance specific: Secure NVM, CSRM debug disabled
After Boot-FW Value	0000 4000$_H$	Secure NVM, CSRM debug enabled

4.6.2.20 CPUx NVM SFR read access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

ACCENPFSFR_RDA Offset address: 00088$_H$

CPUx NVM SFR read access enable register A Reset values see: Table 201

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
ENq (q=00-31)	q	rh	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access

Table 201 Reset values of ACCENPFSFR_RDA

Reset type	Reset value	Note
Application Reset	FFFF FFFF$_H$	Instance specific: Non secure NVM
Application Reset	0000 1000$_H$	Instance specific: Secure NVM, CSRM debug disabled
After Boot-FW Value	1000 3000$_H$	Secure NVM, CSRM debug enabled

4.6.2.21 CPUx NVM SFR read access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

ACCENPFSFR_RDB Offset address: 0008C$_H$

CPUx NVM SFR read access enable register B Reset values see: Table 202

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN63	EN62	EN61	EN60	EN59	EN58	EN57	EN56	EN55	EN54	EN53	EN52	EN51	EN50	EN49	EN48
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN47	EN46	EN45	EN44	EN43	EN42	EN41	EN40	EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
ENq (q=32-63)	q-32	rh	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access

Table 202 Reset values of ACCENPFSFR_RDB

Reset type	Reset value	Note
Application Reset	FFFF FFFF$_H$	Instance specific: Non secure NVM
Application Reset	0000 0000$_H$	Instance specific: Secure NVM, CSRM debug disabled
After Boot-FW Value	0000 4000$_H$	Secure NVM, CSRM debug enabled

4.6.2.22 CPUx NVM SFR VM access enable register

This register defines which virtual machine encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each VMx.

ACCENPFSFR_VM Offset address: 00090$_H$

CPUx NVM SFR VM access enable register Application Reset value: 00FF 00FF$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
RDq (q=00-07)	q	rh	**Read access enable for VM ID q** This bit enables read access to the access protected region for transactions with the VM ID q. 0_B Disabled for read access 1_B Enabled for read access

(table continues...)

(continued)

Field	Bits	Type	Description
WRq (q=00-07)	q+16	rh	**Write access enable for VM ID q** This bit enables write access to the access protected region for transactions with the VM ID q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

4.6.2.23 CPUx NVM SFR PRS access enable register

This register defines which protection register set encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each PRSx.

ACCENPFSFR_PRS Offset address: 00094_H

CPUx NVM SFR PRS access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
RDq (q=00-07)	q	rh	**Read access enable for PRS q** This bit enables read access to the access protected region for transactions with the PRS q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rh	**Write access enable for PRS q** This bit enables write access to the access protected region for transactions with the PRS q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

4.6.2.24 CPUx NVM range y read access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

ACCENPFy_RDA (y=0-7) Offset address: 00108_H+y*20_H

CPUx NVM range y read access enable register A Reset values see: Table 203

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
ENq (q=00-31)	q	rh	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access

Table 203 Reset values of ACCENPFy_RDA (y=0-7)

Reset type	Reset value	Note
Application Reset	FFFF FFFF$_H$	Instance specific: Non secure NVM
Application Reset	0000 1000$_H$	Instance specific: Secure NVM

4.6.2.25 CPUx NVM range y read access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

ACCENPFy_RDB (y=0-7) Offset address: 0010C$_H$+y*20$_H$

CPUx NVM range y read access enable register B Reset values see: Table 204

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN63	EN62	EN61	EN60	EN59	EN58	EN57	EN56	EN55	EN54	EN53	EN52	EN51	EN50	EN49	EN48
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN47	EN46	EN45	EN44	EN43	EN42	EN41	EN40	EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
ENq (q=32-63)	q-32	rh	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access

Table 204 Reset values of ACCENPFy_RDB (y=0-7)

Reset type	Reset value	Note
Application Reset	FFFF FFFF$_H$	Instance specific: Non secure NVM

(table continues...)

Table 204 (continued) Reset values of ACCENPFy_RDB (y=0-7)

Reset type	Reset value	Note
Application Reset	0000 4000$_H$	Instance specific: Secure NVM

4.6.2.26 CPUx NVM range y VM access enable register

This register defines which virtual machine encodings are enabled or disabled for accesses to the access protected region. There is one RDx bit for each VMx.

ACCENPFy_VM (y=0-7) Offset address: 00110$_H$+y*20$_H$

CPUx NVM range y VM access enable register Application Reset value: 0000 00FF$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
RDq (q=00-07)	q	rh	**Read access enable for VM ID q** This bit enables read access to the access protected region for transactions with the VM ID q. 0_B Disabled for read access 1_B Enabled for read access
0	31:8	r	**Reserved** Read as 0; should be written with 0.

4.6.2.27 CPUx NVM range y PRS access enable register

This register defines which protection register set encodings are enabled or disabled for accesses to the access protected region. There is one RDx bit for each PRSx.

ACCENPFy_PRS (y=0-7) Offset address: 00114$_H$+y*20$_H$

CPUx NVM range y PRS access enable register Application Reset value: 0000 00FF$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
RDq (q=00-07)	q	rh	**Read access enable for PRS q** This bit enables read access to the access protected region for transactions with the PRS q. 0_B Disabled for read access 1_B Enabled for read access
0	31:8	r	**Reserved** Read as 0; should be written with 0.

4.6.2.28 CPUx NVM range y region lower address register

This register sets the lower bound of the access protected region. Addresses greater than or equal to the lower bound are considered part of the access protected region.

ACCENPFy_RGNLA (y=0-7) Offset address: 00118_H+y*20_H

CPUx NVM range y region lower address register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADDRH		0	ADDR												
rh		r	rh												

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															
r															

Field	Bits	Type	Description
ADDR	28:16	rh	**Defines bits 28:16 of the lower bound of the access protected region**
ADDRH	31:30	rh	**Defines bits 31:30 of the lower bound of the access protected region**
0	15:0, 29	r	**Reserved** Read as 0; should be written with 0.

4.6.2.29 CPUx NVM range y region upper address register

This register sets the upper bound of the access protected region. Addresses stricly less than the upper bound are considered part of the access protected region.

ACCENPFy_RGNUA (y=0-7) Offset address: $0011C_H+y*20_H$

CPUx NVM range y region upper address register Application Reset value: $DFFF\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADDRH		0	ADDR												
rh		r	rh												

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															
r															

Field	Bits	Type	Description
ADDR	28:16	rh	**Defines bits 28:16 of the upper bound of the access protected region**
ADDRH	31:30	rh	**Defines bits 31:30 of the upper bound of the access protected region**
0	15:0, 29	r	**Reserved** Read as 0; should be written with 0.

4.6.2.30 CPUx NVM interface y configuration register 0

Software may configure a NVM Prefetch Buffer with a master TAG ID stored in NVM Interface Configuration Register 0.

If a CPU instance does not have a local NVM bank then the FLASHCON0 register associated with that instance will have no functionality.

PFIy_FLASHCON0 (y=0-1) Offset address: 00200_H+y*8

CPUx NVM interface y configuration register 0 Reset values see: Table 205

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		TAG4						0		TAG3					
r		rw						r		rw					

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		TAG2						0		TAG1					
r		rw						r		rw					

Field	Bits	Type	Description
TAG1	5:0	rw	**NVM prefetch buffer 1 configuration** FPB is assigned to on chip master with master TAG ID equal to TAG1.
TAG2	13:8	rw	**NVM prefetch buffer 2 configuration** FPB is assigned to on chip master with master TAG ID equal to TAG2.
TAG3	21:16	rw	**NVM prefetch buffer 3 configuration** FPB is assigned to on chip master with master TAG ID equal to TAG3.
TAG4	29:24	rw	**NVM prefetch buffer 4 configuration** FPB is assigned to on chip master with master TAG ID equal to TAG4.
0	7:6, 15:14, 23:22, 31:30	r	**Reserved - RES** Always read as 0; should be written with 0.

Table 205 Reset values of PFIy_FLASHCON0 (y=0-1)

Reset type	Reset value	Note
After Boot-FW Value	$0000\ 0000_H$	PFI0

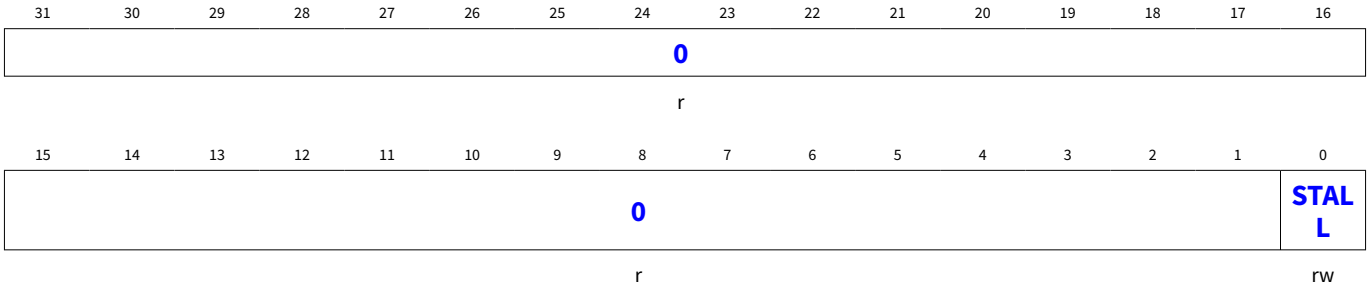
4.6.2.31 CPUx NVM interface y configuration register 1

The CPU NVM Interface Configuration Register 1 controls the stall response.

If a CPU instance does not have a local NVM bank then the FLASHCON1 register associated with that instance will have no functionality.

PFIy_FLASHCON1 (y=0-1) Offset address: 00204_H+y*8

CPUx NVM interface y configuration register 1 Application Reset value: $0000\ 0000_H$

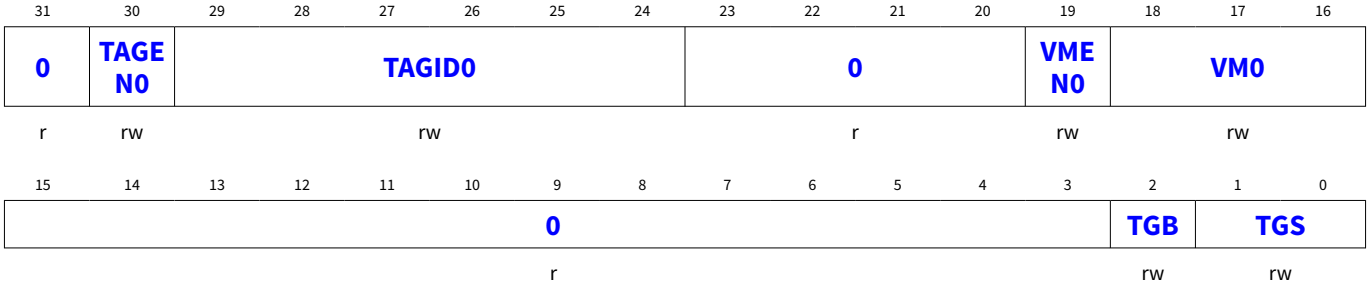


Field	Bits	Type	Description
STALL	0	rw	**Stall Bus Request** This field must not be changed while any bank is busy. The results are unpredictable. It is strongly recommended to configure this field once and avoid changing it during operation. Reading a NVM bank, erase counter (NVM Flash devices), or special sectors (NVM RRAM devices), in Sleep Mode must result in a bus error independent of this field (although it reports "busy"). Reading a status register is possible even while a bank is busy. Writing a user status register while a bank is busy is not possible and results in an error regardless of the value of the STALL field. 0_B **Error**: Reading local NVM bank, erase counter (NVM Flash devices), or special sectors (NVM RRAM devices, when DMU_HCI_STATUS.PxBUSY is set, generates a bus error. 1_B **Stall**: Reading local NVM bank, erase counter (NVM Flash devices), or special sectors (NVM RRAM devices, when DMU_HCI_STATUS.PxBUSY is set delays the response by inserting additional wait cycles until PxBUSY is cleared.
0	31:1	r	**Reserved - RES** Read as 0; should be written with 0.

4.6.2.32 CPUx NVM interface y MCDS performance counter control register 0

TS16PNVMy_CTRL0 (y=0-1) Offset address: 00210_H+y*8

CPUx NVM interface y MCDS performance counter control register 0 Debug Reset value: $0000\ 0000_H$

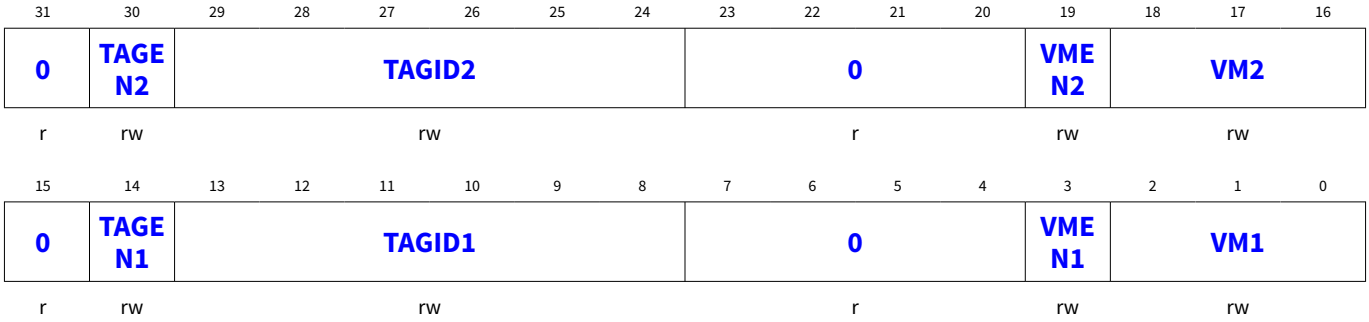


Field	Bits	Type	Description
TGS	1:0	rw	**Trigger set for OTGB0/1** 00_B No trigger set output 01_B Trigger set 'TS16P_NVM' selected **others,** Reserved
TGB	2	rw	**OTGB0/1 bus select** 0_B Trigger set output on OTGB0 1_B Trigger set output on OTGB1
VM0	18:16	rw	**Virtual machine definition for TAG0** Restricts the VM for TAG0. This field is ignored if VMEN0=0_B
VMEN0	19	rw	**Enable for TAG0 virtual machine definition** 0_B Any VM 1_B Only the virtual machine defined by VM0
TAGID0	29:24	rw	**TAG-ID definition for TAG0** Restricts the TAG-ID for TAG0. This field is ignored if TAGEN0=0_B
TAGEN0	30	rw	**Enable for TAG0 TAG-ID definition** 0_B Any master 1_B Only the master defined by TAGID0
0	15:3, 23:20, 31	r	**Reserved** Read as 0; should be written with 0.

4.6.2.33 CPUx NVM interface y MCDS performance counter control register 1

TS16PNVMy_CTRL1 (y=0-1) Offset address: 00214_H+y*8

CPUx NVM interface y MCDS performance counter control register 1 Debug Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
VM1	2:0	rw	**Virtual machine definition for TAG1** Restricts the VM for TAG1. This field is ignored if VMEN1=0_B
VMEN1	3	rw	**Enable for TAG1 virtual machine definition** 0_B Any VM 1_B Only the virtual machine defined by VM1
TAGID1	13:8	rw	**TAG-ID definition for TAG1** Restricts the TAG-ID for TAG1. This field is ignored if TAGEN1=0_B
TAGEN1	14	rw	**Enable for TAG1 TAG-ID definition** 0_B Any master 1_B Only the master defined by TAGID1
VM2	18:16	rw	**Virtual machine definition for TAG2** Restricts the VM for TAG2. This field is ignored if VMEN2=0_B
VMEN2	19	rw	**Enable for TAG2 virtual machine definition** 0_B Any VM 1_B Only the virtual machine defined by VM2
TAGID2	29:24	rw	**TAG-ID definition for TAG2** Restricts the TAG-ID for TAG2. This field is ignored if TAGEN2=0_B
TAGEN2	30	rw	**Enable for TAG2 TAG-ID definition** 0_B Any master 1_B Only the master defined by TAGID2
0	7:4, 15, 23:20, 31	r	**Reserved** Read as 0; should be written with 0.

4.6.2.34 CPUx NVM interface lockstep control register

Provides control for CPU NVM Lockstep Comparator Logic blocks.

CFILCLCON Offset address: 00240_H

CPUx NVM interface lockstep control register Reset values see: Table 206

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LSEN	0														LS
r	r														r

Field	Bits	Type	Description
LS	0	r	**Lockstep mode status** This bit indicates whether the CPU NVM interface is currently running in lockstep monitor mode. 0_B Not in lockstep mode 1_B Running in lockstep mode
LSEN	15	r	**Lockstep enable** Enable lockstep CPU monitoring for the CPU NVM interface. This lockstep is always enabled. 0_B Lockstep is disabled 1_B Lockstep enabled (Default after Cold Power-On Reset)
0	14:1, 31:16	r	**Reserved** Read as 0; should be written with 0.

Table 206 Reset values of CFILCLCON

Reset type	Reset value	Note
Cold PowerOn Reset	$0000\ 8001_H$	Instance specific: Lockstep present
Cold PowerOn Reset	$0000\ 0000_H$	Instance specific: Lockstep not present

4.6.2.35 CPUx NVM interface lockstep test register

Provides the capability for software to inject a fault condition into the comparators of each Lockstep Comparator Logic block. The implementation should generate a single cycle fault each time the bit is written with '1'.

CFILCLTEST Offset address: 00244_H

CPUx NVM interface lockstep test register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															LCLT
r															w

Field	Bits	Type	Description
LCLT	0	w	**LCL lockstep test** Fault injection for LCL. Read as zero. 0_B No action 1_B Inject single fault in LCL0
0	31:1	r	**Reserved - RES** Read as 0; should be written with 0.

4.6.2.36 CPUx CFI error register

CFIERR Offset address: 00250_H

CPUx CFI error register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0				E_VMN_EN	E_VMN			E_PRS_EN	E_PRS			0			
r				rh	rh			rh	rh			r			

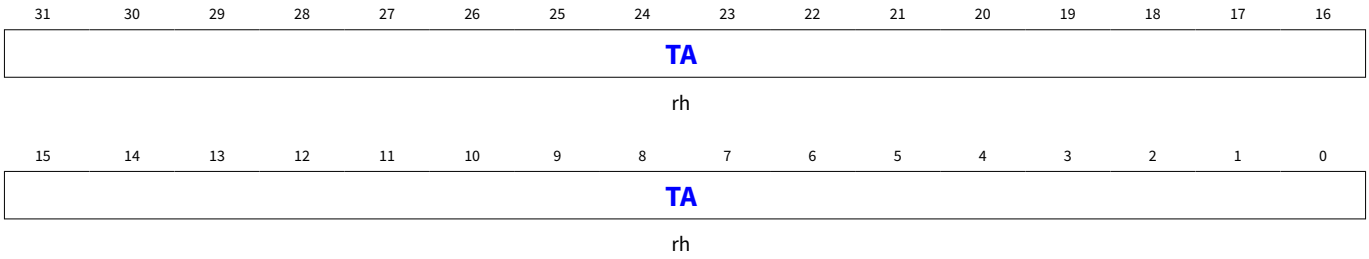
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E_REG	E_WS	E_SOTA	E_SP	0	E_TAG						E_IE	0			ED
rh	rh	rh	rh	r	rh						rh	r			rwh

Field	Bits	Type	Description
ED	0	rwh	**Error detected** 0_B Write: Clear ED bit, re-enable CFIERR and CFIERRAR update. Read : No error condition occurred 1_B Write: No Effect Read : Error captured. CFIERR and CFIERRAR contents valid, further CFIERR and CFIERRAR updates disabled.
E_IE	4	rh	**Integrity error**
E_TAG	10:5	rh	**Captured master TAG ID** Bus master TAG ID of requesting master.
E_SP	12	rh	**PROT APU protection error detected**
E_SOTA	13	rh	**SOTA configuration error detected**
E_WS	14	rh	**PFI wait state ECC error detected**
E_REG	15	rh	**Register access write error detected**
E_PRS	22:20	rh	**Captured PRS**
E_PRS_EN	23	rh	**Captured PRS_EN**
E_VMN	26:24	rh	**Captured VMN**
E_VMN_EN	27	rh	**Captured VMN_EN**
0	3:1, 11, 19:16, 31:28	r	**Reserved - RES** Read as 0; should be written with 0.

4.6.2.37 CPUx CFI error address register

CFIERRAR Offset address: 00254_H

CPUx CFI error address register Application Reset value: $0000\ 0000_H$



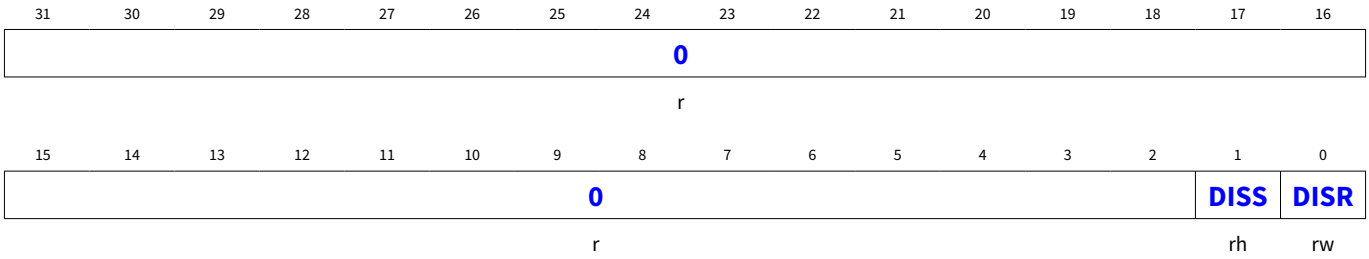
Field	Bits	Type	Description
TA	31:0	rh	**Transaction address** Address being accessed by operation that encountered the error.

4.6.2.38 CPUx PFI buffer disable

A number of buffers (prefetch buffers, data read line buffer) are implemented to reduce the access time of subsequent NVM accesses. When an access, targeting the NVM, is required to read the memory (for example for test), the buffers need to be bypassed to ensure that the data is not retrieved from the buffers. When requested (PFIBUFDIS.DISR=1), PFIBUFDIS.DISS is set, all buffers are invalidated and bypassed ensuring that subsequent accesses, even to the same address are not buffered. The buffers remain disabled until the request is remove (PFIBUFDIS.DISR=0). After reset the buffers are not bypassed.

PFIBUFDIS Offset address: 00320_H

CPUx PFI buffer disable Application Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
DISR	0	rw	**Prefetch and data read line buffers disable request bit** 0_B Enable request: Enable the prefetch and data read line buffers 1_B Bypass request: Disable the prefetch and data read line buffers
DISS	1	rh	**Prefetch and data read line buffers disable status bit** 0_B Pretch buffers and data read line buffers are enabled 1_B Pretch buffers and data read line buffers are invalidated and bypassed. Next access will trigger a new request to the NVM
0	31:2	r	**Reserved - RES** Always read as 0; should be written with 0.

4.7 Debug information

Topics:

- Debug restrictions
- Filtering support for trace based measurement
- Trace bandwidth limiting
- Enabling and disabling the trace using TRCCFG
- TC1.8 CPU and local NVM performance measurement
- System timer suspend

4.7.1 Debug restrictions

For context operations and operations using circular addressing the base address of the operation is used for debug range comparison, rather than the effective address.

When the program cache (PCACHE) is mapped into address space, PCACHE operation can only be enabled when debug is enabled. When the data cache (DCACHE) is mapped into address space DCACHE operation can only be enabled when debug is enabled.

External writes to the processor's GPR registers are not supported unless the processor is in Boot-Halt, Debug-Halt or idle state.

External writes to the processor's PSW and PC registers are not supported unless the processor is in Boot-Halt, Debug-Halt or idle state.

4.7.2 Filtering support for trace based measurement

The current MCDS system produces large amount of trace data. The TC1.8 CPU implements two configuration registers to qualify the data to be traced. Load/Store operations can be filtered on the following conditions in the TRCFILT register:

- Indicate based on a particular base address register used by the Load/Store instruction
- Based on type of load store instruction used 10-bit/16-bit

Additional qualification of trace output required:

- Exclude CSA operations
- Exclude Stack operations from trace

The trace can be further configured to apply to a particular virtual machine or to the whole CPU via the TRCCFG register.

4.7.3 Trace bandwidth limiting

When tracing large amounts of date the MCDS data trace unit (DTU) trace FIFO may saturate. To better match the CPU bandwidth with that of the trace system the CPU implements a mechanism to limit trace output by stalling instruction execution. This feature is controlled through the TRCLIM register.

4.7.4 Enabling and disabling the trace using TRCCFG

The TriCore™ architecture defines multiple multi-cycle operations (for example CALL, Return, …) which can have an effect on the trace data over multiple cycles. When using the TRCCFG register to enable or disable the trace in software, the application should ensure that all previous multi-cycle operation have completed. Configuration of the TRCCFG register is achieved using the MTCR instruction. The architecture requires that an

ISYNC should be placed after the MTCR for the effects to be seen by the CPU. This is not enough in the case of changing the trace configuration.

Enabling the trace

Before enabling the trace, the application software should ensure that the previous instruction is completed and that the store buffers are flushed. The following sequence is recommended in order to avoid instructions being traced partially:

- ISYNC
- DSYNC
- TRCCFG configuration using MTCR
- ISYNC

Disabling the trace

Before disabling the trace, the application software should ensure that the previous instruction is completed and that the store buffers are flushed. The following sequence is recommended in order to avoid incomplete trace of instructions already completed:

- ISYNC
- DSYNC
- TRCCFG configuration using MTCR
- ISYNC

4.7.5 TC1.8 CPU and local NVM performance measurement

Real-time measurement of the core and local NVM performance provides useful insights to system developers. The TC1.8 CPU provides a set of relevant events to the MCDS that can be used to measure different performance aspects without real-time effect on execution.

The performance events that can be measured are described in the following two sections. The 'OTGB core performance counter' and the 'OTGB NVM performance counter' are configured independently and share the same OTGB interfaces. The hardware allows the measurement of a maximum of two sources (out of a maximum of three for CPU instances with two NVM interfaces).

When multiple sources are redirected to the same OTGB interface, the interface will prioritize the core performance over the local NVM0 and over local NVM1 by default.

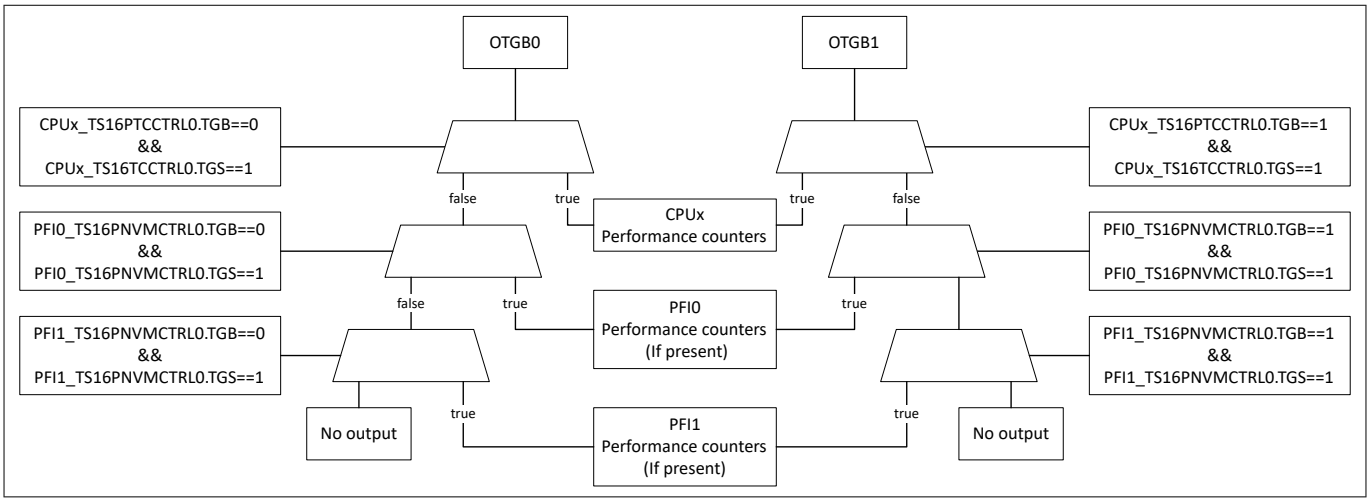


Figure 35 OTGB selection logic

4.7.5.1 OTGB core performance counters

The following table describes the TC1.8 CPU events that can be mapped to counters in MCDS module. Before they can be used in MCDS, they must be activated using the OCS.OTGM function. The list also describes how the event affects a counter selected in the MCDS.

Details of the MCDS counter functions, for example counter width, counter overflow, linkage of counters, are described in the OCMTS chapter.

The OTGB interface is controlled using the HRA_TS16PTCCTRL register which can be used to enable the counters, select the OTGB target and restrict the event to a specific virtual machine number or to the complete TC1.8 CPU instance.

Table 207 TS16P_TC performance counter scheme

Bit	Event function	Resolution [1)]	Description
[1:0]	Instructions executed	2 instructions	Bit 0 indicates that 2 instructions have been executed Bit 1 indicates that 4 instructions have been executed
2	Stall cycles	2 stall cycles	Indication that either the fetch unit or the Load-Store unit are requesting data and are respectively stalled.
3	Correctly predicted branch	2 branches	Indication that a conditional branch instruction is in a branch resolution stage of the pipeline (IP_EX1, LS_DEC, LP_DEC) and is correctly predicted
4	Incorrectly predicted branch	2 branches	Indication that a conditional branch instruction is in a branch resolution stage of the pipeline (IP_EX1, LS_DEC, LP_DEC) and is mispredicted
5	Taken branch	2 branches	Indication that a conditional branch instruction is in a branch resolution stage of the pipeline (IP_EX1, LS_DEC, LP_DEC)
6	Instruction cache hits	2 hits	Indication that the target of a cached fetch request from the fetch unit is found in the program cache
7	Instruction cache misses	2 misses	Indication that the target of a cached fetch request from the fetch unit is not found in the program cache and hence a bus fetch is initiated
8	Data cache hits	2 hits	Indication that the target of a cached request from the Load-Store unit is found in the data cache
9	Data cache misses	2 misses	Indication that the target of a cached request from the Load-Store unit is not found in the data cache and hence a bus fetch is initiated. (This includes clean and dirty misses)
10	PSPR access	2 accesses	Indication of an access to the program scratch pad RAM
11	DSPR access	2 accesses	Indication of an access to the data scratch pad RAM
12	DLMU access	2 accesses	Indication of an access to the local distribute local memory unit RAM
13	0 (not used)	-	
14	Interrupt acknowledge	1 interrupt	Indication that an interrupt request was acknowledged by the local CPU

(table continues...)

Table 207 (continued) TS16P_TC performance counter scheme

Bit	Event function	Resolution [1]	Description
15	Clock indicator	-	This bit is active for one TPB clock cycle every 2 active CPU clock cycles and does consider the CPU‘s fractional divider. This allows a tool to determine the effective clock rate ratio to MCDS considering the CPU‘s fractional divider

1) A resolution of 2 means that the OTGB bit is active for N = E/2 OTGB clock cycles. E is the number of active cycles of the event. Considering also the phase a tool has an uncertainty of +/- 1 event: E = N*2 +/- 1

4.7.5.2 OTGB NVM performance counter

The following table describes the local NVM events that can be mapped to counters in MCDS module. Before they can be used in MCDS, they must be activated using the OCS.OTGM function. The list also describes how the event affects a counter selected in the MCDS.

Details of the MCDS counter functions, for example counter width, counter overflow, linkage of counters, are described in the OCMTS chapter.

The OTGB interface is configured separately per NVM interface (where implemented) using the PFIy_TS16PNVMCTRL0 and PFIy_TS16PNVMCTRL1 registers. These registers can be used to enable the counters, select the OTGB target and restrict the events to a specific virtual machine number, specific master TAG-ID identifiers or to the complete set of transactions of the TC1.8 CPU NVM instance.

Table 208 TS16P_NVM performance counter scheme

Bit	Event function	Resolution [1]	Description
0	TAG0 read access	2 accesses	Indication of a read access from TAG0
1	TAG0 waits for prefetch	2 accesses	Indication that a request from TAG0 waits for a NVM prefetch buffer to complete a prefetch request
2	TAG0 prefetch buffer hit	2 hits	Indication that a request from TAG0 is present in the prefetch buffer
3	TAG0 read line buffer hit	2 hits	Indication that a request from TAG0 is present in the read line buffer
4	TAG1 read access	2 accesses	Indication of a read access from TAG1
5	TAG1 waits for prefetch	2 accesses	Indication that a request from TAG1 waits for a NVM prefetch buffer to complete a prefetch request
6	TAG1 prefetch buffer hit	2 hits	Indication that a request from TAG1 is present in the prefetch buffer
7	TAG1 read line buffer hit	2 hits	Indication that a request from TAG1 is present in the read line buffer
8	TAG2 read access	2 accesses	Indication of a read access from TAG2
9	TAG2 waits for prefetch	2 accesses	Indication that a request from TAG0 waits for a NVM prefetch buffer to complete a prefetch request
10	TAG2 prefetch buffer hit	2 hits	Indication that a request from TAG2 is present in the prefetch buffer

(table continues...)

Table 208 (continued) TS16P_NVM performance counter scheme

Bit	Event function	Resolution [1]	Description
11	TAG2 read line buffer hit	2 hits	Indication that a request from TAG2 is present in the read line buffer
12-14	0 (not used)	-	
15	Clock indicator	-	This bit is active for one TPB clock cycle every 2 CPU clock cycles and does not consider the CPU‘s fractional divider. This allows a tool to determine the clock rate ratio to MCDS

1) A resolution of 2 means that the OTGB bit is active for N = E/2 OTGB clock cycles. E is the number of active cycles of the event. Considering also the phase a tool has an uncertainty of +/- 1 event: E = N*2 +/- 1

4.7.6 System timer suspend

Different configurations of the debug system may be needed in order to ensure synchronicity a CPU and its local STM instance, or between different STM instances.

Single-stepping a single CPU while maintaining local STM synchronicity

In order to ensure the tight coupling of the executing software and the values of the STM timer, the CPU debug system should be configured with DBGACT.SUSP=1. With this setting DBGSR.SUSOUT is set when a debug event is taken. This ensures that as soon as CPUx is suspended, its associated CPUx.STM is also suspended.

***Note**: In this configuration, a small timer drift will be noticed in the other STM instances due to the propagation of the suspend request to the other cores. That is, if CPUx is being single-stepped, STMx is stopped earlier than STMy (y != x) as each breakpoint are taken.*

Debug while maintaining the same time base across all STM instances

In order to ensure common time base across all STM instances, the CPU debug system must be configured with DBGACT.SUSP=0. With this setting, DBGSR.SUSOUT retain its value when a debug event is taken. This ensures that the suspend request is received by all STM instances at the same time, keeping the timebase common across all instances.

***Note**: In this configuration, all STM instances keep the same time base but a small time drift is introduced between CPUx and its STMx instance. That is CPUx will suspend while STMx (and all other STM instances) may keep counting for a few cycles.*

4.8 References

1. TriCore™ TC1.8 architecture manual volume 1
2. TriCore™ TC1.8 architecture manual volume 2
3. IEEE-754-2019 - IEEE Standard for Floating-Point Arithmetic

4.9 CPU revision history

Reference	Description of change(s)
Date range: 2024-08-17 to 2025-02-07	
Functional overview	• Clarified location of the APU protection
HRA_DBGSR	• Replaced incorrect mention of PSW.CDC with "core debug controller"

Reference	Description of change(s)
Erase counter and register accesses	• Added clarification that erase counters are only available for NVM Flash based devices
PFIy_FLASHCON1	• Added clarification that erase counters are only available for NVM Flash based devices
PFIBUFDIS	• PFIBUDIS register now visible in all devices. The register was previously hidden
Online data acquisition (OLDA) space	• Removed incorrect OLDA address and added reference to the system address map
PROTSPRSE, PROTDLMUSE, PROTSTMSE, PROTPFSE	• Removed SEL bit-field from the Access mode restrictions table as the bit-field is not protected by .OWEN

4.10 TC4Dx CPU information

4.10.1 TC4Dx CPU configuration

The different CPU and local memory configurations for the AURIX™ TC4xx family of devices are defined in the following table:

Table 209 Processor and local memory configuration of the TC4Dx

Feature	Core ID						
	0	1	2	3	4	5	6(CS)
Program cache	32 KB	32 KB	32 KB	32 KB	32 KB	32 KB	32 KB
Program scratch-pad RAM	64 KB	64 KB	64 KB	64 KB	64 KB	64 KB	64 KB
Data cache	16 KB	16 KB	16 KB	16 KB	16 KB	16 KB	16 KB
Data scratch-pad RAM	240 KB	240 KB	240 KB	240 KB	240 KB	240 KB	240 KB
DLMU	512 KB[1)]	512 KB [1)]	512 KB	512 KB	512 KB	512 KB	No
Local NVM Banks	2+2 MB	2+2 MB	1+1 MB	2+2 MB	2+2 MB	1+1 MB	1 MB
Lockstep	Yes	Yes	Yes	Yes	Yes	Yes	No
Virtualization	Yes	Yes	Yes	Yes	Yes	Yes	No
Single-precision FPU	Yes	Yes	Yes	Yes	Yes	Yes	No
Double-precision FPU	Yes	Yes	Yes	Yes	Yes	Yes	No
Overlay	Yes	Yes	Yes	Yes	Yes	Yes	No
FPI connectivity	Dual FPI	Dual FPI	Dual FPI	Dual FPI	Dual FPI	Dual FPI	Dual FPI

1) Includes 128 KB Standby SRAM

Table 210 CPU Segment 0xF Peripheral address mapping

Address range	Dual FPI
$F0000000_H$ – $F3FFFFFF_H$	SPB
$F4000000_H$ – $F5FFFFFF_H$	COMPB
$F6000000_H$ – $FFFFFFFF_H$	SRI

4.10.2 TC4Dx CPU features

There are no deviations from the generic specification.

4.10.3 TC4Dx CPU functional description

There are no deviations from the generic specification.

4.10.4 TC4Dx CPU registers

4.10.4.1 Registers of CPU

4.10.4.1.1 Memory overview tables of CPU domain SRI

Table 211 Memory overview - CPU domain SRI (ascending address)

Short name	Long name	Address
CPUCS_DSPR	Data ScratchPad RAM ($3C000_H$ Byte)	10000000_H
CPUCS_DCACHE	Data Cache RAM (04000_H Byte)	$1003C000_H$
CPUCS_PSPR	Program ScratchPad RAM (10000_H Byte)	10100000_H
CPUCS_PCACHE	Program Cache RAM (08000_H Byte)	10110000_H
CPUCS_DTAG	Data Cache Tag RAM (01800_H Byte)	10200000_H
CPUCS_PTAG	Program Cache TAG RAM (03000_H Byte)	10300000_H
CPU5_DSPR	Data ScratchPad RAM ($3C000_H$ Byte)	20000000_H
CPU5_DCACHE	Data Cache RAM (04000_H Byte)	$2003C000_H$
CPU5_PSPR	Program ScratchPad RAM (10000_H Byte)	20100000_H
CPU5_PCACHE	Program Cache RAM (08000_H Byte)	20110000_H
CPU5_DTAG	Data Cache Tag RAM (01800_H Byte)	20200000_H
CPU5_PTAG	Program Cache TAG RAM (03000_H Byte)	20300000_H
CPU4_DSPR	Data ScratchPad RAM ($3C000_H$ Byte)	30000000_H
CPU4_DCACHE	Data Cache RAM (04000_H Byte)	$3003C000_H$
CPU4_PSPR	Program ScratchPad RAM (10000_H Byte)	30100000_H

(table continues...)

Table 211 (continued) Memory overview - CPU domain SRI (ascending address)

Short name	Long name	Address
CPU4_PCACHE	Program Cache RAM (08000_H Byte)	30110000_H
CPU4_DTAG	Data Cache Tag RAM (01800_H Byte)	30200000_H
CPU4_PTAG	Program Cache TAG RAM (03000_H Byte)	30300000_H
CPU3_DSPR	Data ScratchPad RAM ($3C000_H$ Byte)	40000000_H
CPU3_DCACHE	Data Cache RAM (04000_H Byte)	$4003C000_H$
CPU3_PSPR	Program ScratchPad RAM (10000_H Byte)	40100000_H
CPU3_PCACHE	Program Cache RAM (08000_H Byte)	40110000_H
CPU3_DTAG	Data Cache Tag RAM (01800_H Byte)	40200000_H
CPU3_PTAG	Program Cache TAG RAM (03000_H Byte)	40300000_H
CPU2_DSPR	Data ScratchPad RAM ($3C000_H$ Byte)	50000000_H
CPU2_DCACHE	Data Cache RAM (04000_H Byte)	$5003C000_H$
CPU2_PSPR	Program ScratchPad RAM (10000_H Byte)	50100000_H
CPU2_PCACHE	Program Cache RAM (08000_H Byte)	50110000_H
CPU2_DTAG	Data Cache Tag RAM (01800_H Byte)	50200000_H
CPU2_PTAG	Program Cache TAG RAM (03000_H Byte)	50300000_H
CPU1_DSPR	Data ScratchPad RAM ($3C000_H$ Byte)	60000000_H
CPU1_DCACHE	Data Cache RAM (04000_H Byte)	$6003C000_H$
CPU1_PSPR	Program ScratchPad RAM (10000_H Byte)	60100000_H

(table continues...)

Table 211 (continued) Memory overview - CPU domain SRI (ascending address)

Short name	Long name	Address
CPU1_PCACHE	Program Cache RAM (08000_H Byte)	60110000_H
CPU1_DTAG	Data Cache Tag RAM (01800_H Byte)	60200000_H
CPU1_PTAG	Program Cache TAG RAM (03000_H Byte)	60300000_H
CPU0_DSPR	Data ScratchPad RAM ($3C000_H$ Byte)	70000000_H
CPU0_DCACHE	Data Cache RAM (04000_H Byte)	$7003C000_H$
CPU0_PSPR	Program ScratchPad RAM (10000_H Byte)	70100000_H
CPU0_PCACHE	Program Cache RAM (08000_H Byte)	70110000_H
CPU0_DTAG	Data Cache Tag RAM (01800_H Byte)	70200000_H
CPU0_PTAG	Program Cache TAG RAM (03000_H Byte)	70300000_H
CPU0_DLMU_MEM	DLMU RAM (80000_H Byte)	90000000_H
CPU1_DLMU_MEM	DLMU RAM (80000_H Byte)	90080000_H
CPU2_DLMU_MEM	DLMU RAM (80000_H Byte)	90100000_H
CPU3_DLMU_MEM	DLMU RAM (80000_H Byte)	90180000_H
CPU4_DLMU_MEM	DLMU RAM (80000_H Byte)	90200000_H
CPU5_DLMU_MEM	DLMU RAM (80000_H Byte)	90200000_H
CPU0_NCDLMU_MEM	DLMU RAM (80000_H Byte)	$B0000000_H$
CPU1_NCDLMU_MEM	DLMU RAM (80000_H Byte)	$B0080000_H$
CPU2_NCDLMU_MEM	DLMU RAM (80000_H Byte)	$B0100000_H$

(table continues...)

Table 211 (continued) Memory overview - CPU domain SRI (ascending address)

Short name	Long name	Address
CPU3_NCDLMU_MEM	DLMU RAM (80000_H Byte)	$B0180000_H$
CPU4_NCDLMU_MEM	DLMU RAM (80000_H Byte)	$B0200000_H$
CPU5_NCDLMU_MEM	DLMU RAM (80000_H Byte)	$B0280000_H$

4.10.4.1.2 Register address space - CPU domain SRI

Table 212 Registers address space - CPU domain SRI

Key: The module name in brackets () indicates a memory section name. Without brackets the reference is to a functional block.

Module	Domain	Base address	End address	Note
(CPU0)	SRI	70000000_H	$7003BFFF_H$	Data ScratchPad RAM interface
	SRI	$7003C000_H$	$7003FFFF_H$	Data Cache RAM interface
	SRI	70100000_H	$7010FFFF_H$	Program ScratchPad RAM interface
	SRI	70110000_H	$70117FFF_H$	Program Cache RAM interface
	SRI	70200000_H	$702017FF_H$	Data Cache Tag RAM interface
	SRI	70300000_H	$70302FFF_H$	Program Cache TAG RAM interface
	SRI	90000000_H	$9007FFFF_H$	DLMU RAM interface (cached)
	SRI	$B0000000_H$	$B007FFFF_H$	DLMU RAM interface (non-cached)
CPU0	SRI	$F8800000_H$	$F883FFFF_H$	SRI slave interface for SFR+CSFR
(CPU1)	SRI	60000000_H	$6003BFFF_H$	Data ScratchPad RAM interface
	SRI	$6003C000_H$	$6003FFFF_H$	Data Cache RAM interface
	SRI	60100000_H	$6010FFFF_H$	Program ScratchPad RAM interface
	SRI	60110000_H	$60117FFF_H$	Program Cache RAM interface
	SRI	60200000_H	$602017FF_H$	Data Cache Tag RAM interface
	SRI	60300000_H	$60302FFF_H$	Program Cache TAG RAM interface
	SRI	90080000_H	$900FFFFF_H$	DLMU RAM interface (cached)
	SRI	$B0080000_H$	$B00FFFFF_H$	DLMU RAM interface (non-cached)
CPU1	SRI	$F8840000_H$	$F887FFFF_H$	SRI slave interface for SFR+CSFR
(CPU2)	SRI	50000000_H	$5003BFFF_H$	Data ScratchPad RAM interface
	SRI	$5003C000_H$	$5003FFFF_H$	Data Cache RAM interface
	SRI	50100000_H	$5010FFFF_H$	Program ScratchPad RAM interface
	SRI	50110000_H	$50117FFF_H$	Program Cache RAM interface

(table continues...)

Table 212 (continued) Registers address space - CPU domain SRI

Key: The module name in brackets () indicates a memory section name. Without brackets the reference is to a functional block.

Module	Domain	Base address	End address	Note
	SRI	50200000_H	$502017FF_H$	Data Cache Tag RAM interface
	SRI	50300000_H	$50302FFF_H$	Program Cache TAG RAM interface
	SRI	90100000_H	$9017FFFF_H$	DLMU RAM interface (cached)
	SRI	$B0100000_H$	$B017FFFF_H$	DLMU RAM interface (non-cached)
CPU2	SRI	$F8880000_H$	$F88BFFFF_H$	SRI slave interface for SFR+CSFR
(CPU3)	SRI	40000000_H	$4003BFFF_H$	Data ScratchPad RAM interface
	SRI	$4003C000_H$	$4003FFFF_H$	Data Cache RAM interface
	SRI	40100000_H	$4010FFFF_H$	Program ScratchPad RAM interface
	SRI	40110000_H	$40117FFF_H$	Program Cache RAM interface
	SRI	40200000_H	$402017FF_H$	Data Cache Tag RAM interface
	SRI	40300000_H	$40302FFF_H$	Program Cache TAG RAM interface
	SRI	90180000_H	$901FFFFF_H$	DLMU RAM interface (cached)
	SRI	$B0180000_H$	$B01FFFFF_H$	DLMU RAM interface (non-cached)
CPU3	SRI	$F88C0000_H$	$F88FFFFF_H$	SRI slave interface for SFR+CSFR
(CPU4)	SRI	30000000_H	$3003BFFF_H$	Data ScratchPad RAM interface
	SRI	$3003C000_H$	$3003FFFF_H$	Data Cache RAM interface
	SRI	30100000_H	$3010FFFF_H$	Program ScratchPad RAM interface
	SRI	30110000_H	$30117FFF_H$	Program Cache RAM interface
	SRI	30200000_H	$302017FF_H$	Data Cache Tag RAM interface
	SRI	30300000_H	$30302FFF_H$	Program Cache TAG RAM interface
	SRI	90200000_H	$9027FFFF_H$	DLMU RAM interface (cached)
	SRI	$B0200000_H$	$B027FFFF_H$	DLMU RAM interface (non-cached)
CPU4	SRI	$F8900000_H$	$F893FFFF_H$	SRI slave interface for SFR+CSFR
(CPU5)	SRI	20000000_H	$2003BFFF_H$	Data ScratchPad RAM interface
	SRI	$2003C000_H$	$2003FFFF_H$	Data Cache RAM interface
	SRI	20100000_H	$2010FFFF_H$	Program ScratchPad RAM interface
	SRI	20110000_H	$20117FFF_H$	Program Cache RAM interface
	SRI	20200000_H	$202017FF_H$	Data Cache Tag RAM interface
	SRI	20300000_H	$20302FFF_H$	Program Cache TAG RAM interface
	SRI	90200000_H	$9027FFFF_H$	DLMU RAM interface (cached)
	SRI	$B0280000_H$	$B02FFFFF_H$	DLMU RAM interface (non-cached)
CPU5	SRI	$F8940000_H$	$F897FFFF_H$	SRI slave interface for SFR+CSFR

(table continues...)

Table 212 (continued) Registers address space - CPU domain SRI

Key: The module name in brackets () indicates a memory section name. Without brackets the reference is to a functional block.

Module	Domain	Base address	End address	Note
(CPUCS)	SRI	10000000_H	$1003BFFF_H$	Data ScratchPad RAM interface
	SRI	$1003C000_H$	$1003FFFF_H$	Data Cache RAM interface
	SRI	10100000_H	$1010FFFF_H$	Program ScratchPad RAM interface
	SRI	10110000_H	$10117FFF_H$	Program Cache RAM interface
	SRI	10200000_H	$102017FF_H$	Data Cache Tag RAM interface
	SRI	10300000_H	$10302FFF_H$	Program Cache TAG RAM interface
CPUCS	SRI	$F8980000_H$	$F89BFFFF_H$	SRI slave interface for SFR+CSFR

4.10.4.1.3 Register address space - CPU domain CSFR

Table 213 Registers address space - CPU domain CSFR

Module	Domain	Base address	End address	Note
CPU	CSFR	0000_H	$FFFF_H$	Short addresses for MxCR/MxDCR instructions

4.10.4.1.4 Register overview - access mode glossary

Table 214 Register overview - access mode glossary

Keyword	Description
DLMUSE	Access protection using PROT register CPU0_PROTDLMUSE or CPU1_PROTDLMUSE or CPU2_PROTDLMUSE or CPU3_PROTDLMUSE or CPU4_PROTDLMUSE or CPU5_PROTDLMUSE .
RSTE	Access protection using PROT register CPU0_PROTRSTE or CPU1_PROTRSTE or CPU2_PROTRSTE or CPU3_PROTRSTE or CPU4_PROTRSTE or CPU5_PROTRSTE or CPUCS_PROTRSTE.
SFRE	Access protection using PROT register CPU0_PROTSFRE or CPU1_PROTSFRE or CPU2_PROTSFRE or CPU3_PROTSFRE or CPU4_PROTSFRE or CPU5_PROTSFRE or CPUCS_PROTSFRE.
SFRSE	Access protection using PROT register CPU0_PROTSFRSE or CPU1_PROTSFRSE or CPU2_PROTSFRSE or CPU3_PROTSFRSE or CPU4_PROTSFRSE or CPU5_PROTSFRSE or CPUCS_PROTSFRSE.
SPRSE	Access protection using PROT register CPU0_PROTSPRSE or CPU1_PROTSPRSE or CPU2_PROTSPRSE or CPU3_PROTSPRSE or CPU4_PROTSPRSE or CPU5_PROTSPRSE or CPUCS_PROTSPRSE.
STMSE	Access protection using PROT register CPU0_PROTSTMSE or CPU1_PROTSTMSE or CPU2_PROTSTMSE or CPU3_PROTSTMSE or CPU4_PROTSTMSE or CPU5_PROTSTMSE or CPUCS_PROTSTMSE.

(table continues...)

Table 214 (continued) Register overview - access mode glossary

Keyword	Description
APU-P0	Protection group consisting of registers CPU0_ACCENSFRCFG_WRA, CPU0_ACCENSFRCFG_WRB, CPU0_ACCENSFRCFG_RDA, CPU0_ACCENSFRCFG_RDB, CPU0_ACCENSFRCFG_VM , CPU0_ACCENSFRCFG_PRS , CPU0_ACCENSFR_WRA, CPU0_ACCENSFR_WRB, CPU0_ACCENSFR_RDA, CPU0_ACCENSFR_RDB, CPU0_ACCENSFR_VM , CPU0_ACCENSFR_PRS or CPU1_ACCENSFRCFG_WRA, CPU1_ACCENSFRCFG_WRB, CPU1_ACCENSFRCFG_RDA, CPU1_ACCENSFRCFG_RDB, CPU1_ACCENSFRCFG_VM , CPU1_ACCENSFRCFG_PRS , CPU1_ACCENSFR_WRA, CPU1_ACCENSFR_WRB, CPU1_ACCENSFR_RDA, CPU1_ACCENSFR_RDB, CPU1_ACCENSFR_VM , CPU1_ACCENSFR_PRS or CPU2_ACCENSFRCFG_WRA, CPU2_ACCENSFRCFG_WRB, CPU2_ACCENSFRCFG_RDA, CPU2_ACCENSFRCFG_RDB, CPU2_ACCENSFRCFG_VM , CPU2_ACCENSFRCFG_PRS , CPU2_ACCENSFR_WRA, CPU2_ACCENSFR_WRB, CPU2_ACCENSFR_RDA, CPU2_ACCENSFR_RDB, CPU2_ACCENSFR_VM , CPU2_ACCENSFR_PRS or CPU3_ACCENSFRCFG_WRA, CPU3_ACCENSFRCFG_WRB, CPU3_ACCENSFRCFG_RDA, CPU3_ACCENSFRCFG_RDB, CPU3_ACCENSFRCFG_VM , CPU3_ACCENSFRCFG_PRS , CPU3_ACCENSFR_WRA, CPU3_ACCENSFR_WRB, CPU3_ACCENSFR_RDA, CPU3_ACCENSFR_RDB, CPU3_ACCENSFR_VM , CPU3_ACCENSFR_PRS or CPU4_ACCENSFRCFG_WRA, CPU4_ACCENSFRCFG_WRB, CPU4_ACCENSFRCFG_RDA, CPU4_ACCENSFRCFG_RDB, CPU4_ACCENSFRCFG_VM , CPU4_ACCENSFRCFG_PRS , CPU4_ACCENSFR_WRA, CPU4_ACCENSFR_WRB, CPU4_ACCENSFR_RDA, CPU4_ACCENSFR_RDB, CPU4_ACCENSFR_VM , CPU4_ACCENSFR_PRS or CPU5_ACCENSFRCFG_WRA, CPU5_ACCENSFRCFG_WRB, CPU5_ACCENSFRCFG_RDA, CPU5_ACCENSFRCFG_RDB, CPU5_ACCENSFRCFG_VM , CPU5_ACCENSFRCFG_PRS , CPU5_ACCENSFR_WRA, CPU5_ACCENSFR_WRB, CPU5_ACCENSFR_RDA, CPU5_ACCENSFR_RDB, CPU5_ACCENSFR_VM , CPU5_ACCENSFR_PRS or CPUCS_ACCENSFRCFG_WRA, CPUCS_ACCENSFRCFG_WRB, CPUCS_ACCENSFRCFG_RDA, CPUCS_ACCENSFRCFG_RDB, CPUCS_ACCENSFRCFG_VM , CPUCS_ACCENSFRCFG_PRS , CPUCS_ACCENSFR_WRA, CPUCS_ACCENSFR_WRB, CPUCS_ACCENSFR_RDA, CPUCS_ACCENSFR_RDB, CPUCS_ACCENSFR_VM , CPUCS_ACCENSFR_PRS .
P0	Access protection using *APU-P0* registers.

(table continues...)

Table 214 (continued) Register overview - access mode glossary

Keyword	Description
APU-P1	Protection group consisting of registers CPU0_ACCENSTMCFG_WRA, CPU0_ACCENSTMCFG_WRB , CPU0_ACCENSTMCFG_RDA, CPU0_ACCENSTMCFG_RDB, CPU0_ACCENSTMCFG_VM , CPU0_ACCENSTMCFG_PRS , CPU0_ACCENSTMVMy_WRA, CPU0_ACCENSTMVMy_WRB , CPU0_ACCENSTMVMy_RDA, CPU0_ACCENSTMVMy_RDB, CPU0_ACCENSTMVMy_VM , CPU0_ACCENSTMVMy_PRS or CPU1_ACCENSTMCFG_WRA, CPU1_ACCENSTMCFG_WRB , CPU1_ACCENSTMCFG_RDA, CPU1_ACCENSTMCFG_RDB, CPU1_ACCENSTMCFG_VM , CPU1_ACCENSTMCFG_PRS , CPU1_ACCENSTMVMy_WRA, CPU1_ACCENSTMVMy_WRB , CPU1_ACCENSTMVMy_RDA, CPU1_ACCENSTMVMy_RDB, CPU1_ACCENSTMVMy_VM , CPU1_ACCENSTMVMy_PRS or CPU2_ACCENSTMCFG_WRA, CPU2_ACCENSTMCFG_WRB , CPU2_ACCENSTMCFG_RDA, CPU2_ACCENSTMCFG_RDB, CPU2_ACCENSTMCFG_VM , CPU2_ACCENSTMCFG_PRS , CPU2_ACCENSTMVMy_WRA, CPU2_ACCENSTMVMy_WRB , CPU2_ACCENSTMVMy_RDA, CPU2_ACCENSTMVMy_RDB, CPU2_ACCENSTMVMy_VM , CPU2_ACCENSTMVMy_PRS or CPU3_ACCENSTMCFG_WRA, CPU3_ACCENSTMCFG_WRB , CPU3_ACCENSTMCFG_RDA, CPU3_ACCENSTMCFG_RDB, CPU3_ACCENSTMCFG_VM , CPU3_ACCENSTMCFG_PRS , CPU3_ACCENSTMVMy_WRA, CPU3_ACCENSTMVMy_WRB , CPU3_ACCENSTMVMy_RDA, CPU3_ACCENSTMVMy_RDB, CPU3_ACCENSTMVMy_VM , CPU3_ACCENSTMVMy_PRS or CPU4_ACCENSTMCFG_WRA, CPU4_ACCENSTMCFG_WRB , CPU4_ACCENSTMCFG_RDA, CPU4_ACCENSTMCFG_RDB, CPU4_ACCENSTMCFG_VM , CPU4_ACCENSTMCFG_PRS , CPU4_ACCENSTMVMy_WRA, CPU4_ACCENSTMVMy_WRB , CPU4_ACCENSTMVMy_RDA, CPU4_ACCENSTMVMy_RDB, CPU4_ACCENSTMVMy_VM , CPU4_ACCENSTMVMy_PRS or CPU5_ACCENSTMCFG_WRA, CPU5_ACCENSTMCFG_WRB , CPU5_ACCENSTMCFG_RDA, CPU5_ACCENSTMCFG_RDB, CPU5_ACCENSTMCFG_VM , CPU5_ACCENSTMCFG_PRS , CPU5_ACCENSTMVMy_WRA, CPU5_ACCENSTMVMy_WRB , CPU5_ACCENSTMVMy_RDA, CPU5_ACCENSTMVMy_RDB, CPU5_ACCENSTMVMy_VM , CPU5_ACCENSTMVMy_PRS or CPUCS_ACCENSTMCFG_WRA, CPUCS_ACCENSTMCFG_WRB , CPUCS_ACCENSTMCFG_RDA, CPUCS_ACCENSTMCFG_RDB, CPUCS_ACCENSTMCFG_VM , CPUCS_ACCENSTMCFG_PRS , CPUCS_ACCENSTM_WRA, CPUCS_ACCENSTM_WRB , CPUCS_ACCENSTM_RDA, CPUCS_ACCENSTM_RDB, CPUCS_ACCENSTM_VM , CPUCS_ACCENSTM_PRS .
P1	Access protection using *APU-P1* registers.

(table continues...)

Table 214 (continued) Register overview - access mode glossary

Keyword	Description
APU-P2	Protection group consisting of registers CPU0_ACCENSPRCFG_WRA, CPU0_ACCENSPRCFG_WRB, CPU0_ACCENSPRCFG_RDA, CPU0_ACCENSPRCFG_RDB, CPU0_ACCENSPRCFG_VM , CPU0_ACCENSPRCFG_PRS , CPU0_ACCENSPRCFG_RGNLA , CPU0_ACCENSPRCFG_RGNUA , CPU0_ACCENSPRy_WRA, CPU0_ACCENSPRy_WRB, CPU0_ACCENSPRy_RDA, CPU0_ACCENSPRy_RDB, CPU0_ACCENSPRy_VM , CPU0_ACCENSPRy_PRS , CPU0_ACCENSPRy_RGNLA , CPU0_ACCENSPRy_RGNUA or CPU1_ACCENSPRCFG_WRA, CPU1_ACCENSPRCFG_WRB, CPU1_ACCENSPRCFG_RDA, CPU1_ACCENSPRCFG_RDB, CPU1_ACCENSPRCFG_VM , CPU1_ACCENSPRCFG_PRS , CPU1_ACCENSPRCFG_RGNLA , CPU1_ACCENSPRCFG_RGNUA , CPU1_ACCENSPRy_WRA, CPU1_ACCENSPRy_WRB, CPU1_ACCENSPRy_RDA, CPU1_ACCENSPRy_RDB, CPU1_ACCENSPRy_VM , CPU1_ACCENSPRy_PRS , CPU1_ACCENSPRy_RGNLA , CPU1_ACCENSPRy_RGNUA or CPU2_ACCENSPRCFG_WRA, CPU2_ACCENSPRCFG_WRB, CPU2_ACCENSPRCFG_RDA, CPU2_ACCENSPRCFG_RDB, CPU2_ACCENSPRCFG_VM , CPU2_ACCENSPRCFG_PRS , CPU2_ACCENSPRCFG_RGNLA , CPU2_ACCENSPRCFG_RGNUA , CPU2_ACCENSPRy_WRA, CPU2_ACCENSPRy_WRB, CPU2_ACCENSPRy_RDA, CPU2_ACCENSPRy_RDB, CPU2_ACCENSPRy_VM , CPU2_ACCENSPRy_PRS , CPU2_ACCENSPRy_RGNLA , CPU2_ACCENSPRy_RGNUA or CPU3_ACCENSPRCFG_WRA, CPU3_ACCENSPRCFG_WRB, CPU3_ACCENSPRCFG_RDA, CPU3_ACCENSPRCFG_RDB, CPU3_ACCENSPRCFG_VM , CPU3_ACCENSPRCFG_PRS , CPU3_ACCENSPRCFG_RGNLA , CPU3_ACCENSPRCFG_RGNUA , CPU3_ACCENSPRy_WRA, CPU3_ACCENSPRy_WRB, CPU3_ACCENSPRy_RDA, CPU3_ACCENSPRy_RDB, CPU3_ACCENSPRy_VM , CPU3_ACCENSPRy_PRS , CPU3_ACCENSPRy_RGNLA , CPU3_ACCENSPRy_RGNUA or CPU4_ACCENSPRCFG_WRA, CPU4_ACCENSPRCFG_WRB, CPU4_ACCENSPRCFG_RDA, CPU4_ACCENSPRCFG_RDB, CPU4_ACCENSPRCFG_VM , CPU4_ACCENSPRCFG_PRS , CPU4_ACCENSPRCFG_RGNLA , CPU4_ACCENSPRCFG_RGNUA , CPU4_ACCENSPRy_WRA, CPU4_ACCENSPRy_WRB, CPU4_ACCENSPRy_RDA, CPU4_ACCENSPRy_RDB, CPU4_ACCENSPRy_VM , CPU4_ACCENSPRy_PRS , CPU4_ACCENSPRy_RGNLA , CPU4_ACCENSPRy_RGNUA or CPU5_ACCENSPRCFG_WRA, CPU5_ACCENSPRCFG_WRB, CPU5_ACCENSPRCFG_RDA, CPU5_ACCENSPRCFG_RDB, CPU5_ACCENSPRCFG_VM , CPU5_ACCENSPRCFG_PRS , CPU5_ACCENSPRCFG_RGNLA , CPU5_ACCENSPRCFG_RGNUA , CPU5_ACCENSPRy_WRA, CPU5_ACCENSPRy_WRB, CPU5_ACCENSPRy_RDA, CPU5_ACCENSPRy_RDB, CPU5_ACCENSPRy_VM , CPU5_ACCENSPRy_PRS , CPU5_ACCENSPRy_RGNLA , CPU5_ACCENSPRy_RGNUA or CPUCS_ACCENSPRCFG_WRA, CPUCS_ACCENSPRCFG_WRB, CPUCS_ACCENSPRCFG_RDA, CPUCS_ACCENSPRCFG_RDB, CPUCS_ACCENSPRCFG_VM , CPUCS_ACCENSPRCFG_PRS , CPUCS_ACCENSPRCFG_RGNLA , CPUCS_ACCENSPRCFG_RGNUA , CPUCS_ACCENSPRy_WRA, CPUCS_ACCENSPRy_WRB, CPUCS_ACCENSPRy_RDA, CPUCS_ACCENSPRy_RDB, CPUCS_ACCENSPRy_VM , CPUCS_ACCENSPRy_PRS , CPUCS_ACCENSPRy_RGNLA , CPUCS_ACCENSPRy_RGNUA .
P2	Access protection using *APU-P2* registers.

(table continues...)

Table 214 (continued) Register overview - access mode glossary

Keyword	Description
APU-P3	Protection group consisting of registers CPU0_ACCENDLMUCFG_WRA , CPU0_ACCENDLMUCFG_WRB , CPU0_ACCENDLMUCFG_RDA , CPU0_ACCENDLMUCFG_RDB , CPU0_ACCENDLMUCFG_VM , CPU0_ACCENDLMUCFG_PRS , CPU0_ACCENDLMUCFG_RGNLA , CPU0_ACCENDLMUCFG_RGNUA , CPU0_ACCENDLMUy_WRA , CPU0_ACCENDLMUy_WRB , CPU0_ACCENDLMUy_RDA , CPU0_ACCENDLMUy_RDB , CPU0_ACCENDLMUy_VM , CPU0_ACCENDLMUy_PRS , CPU0_ACCENDLMUy_RGNLA , CPU0_ACCENDLMUy_RGNUA or CPU1_ACCENDLMUCFG_WRA , CPU1_ACCENDLMUCFG_WRB , CPU1_ACCENDLMUCFG_RDA , CPU1_ACCENDLMUCFG_RDB , CPU1_ACCENDLMUCFG_VM , CPU1_ACCENDLMUCFG_PRS , CPU1_ACCENDLMUCFG_RGNLA , CPU1_ACCENDLMUCFG_RGNUA , CPU1_ACCENDLMUy_WRA , CPU1_ACCENDLMUy_WRB , CPU1_ACCENDLMUy_RDA , CPU1_ACCENDLMUy_RDB , CPU1_ACCENDLMUy_VM , CPU1_ACCENDLMUy_PRS , CPU1_ACCENDLMUy_RGNLA , CPU1_ACCENDLMUy_RGNUA or CPU2_ACCENDLMUCFG_WRA , CPU2_ACCENDLMUCFG_WRB , CPU2_ACCENDLMUCFG_RDA , CPU2_ACCENDLMUCFG_RDB , CPU2_ACCENDLMUCFG_VM , CPU2_ACCENDLMUCFG_PRS , CPU2_ACCENDLMUCFG_RGNLA , CPU2_ACCENDLMUCFG_RGNUA , CPU2_ACCENDLMUy_WRA , CPU2_ACCENDLMUy_WRB , CPU2_ACCENDLMUy_RDA , CPU2_ACCENDLMUy_RDB , CPU2_ACCENDLMUy_VM , CPU2_ACCENDLMUy_PRS , CPU2_ACCENDLMUy_RGNLA , CPU2_ACCENDLMUy_RGNUA or CPU3_ACCENDLMUCFG_WRA , CPU3_ACCENDLMUCFG_WRB , CPU3_ACCENDLMUCFG_RDA , CPU3_ACCENDLMUCFG_RDB , CPU3_ACCENDLMUCFG_VM , CPU3_ACCENDLMUCFG_PRS , CPU3_ACCENDLMUCFG_RGNLA , CPU3_ACCENDLMUCFG_RGNUA , CPU3_ACCENDLMUy_WRA , CPU3_ACCENDLMUy_WRB , CPU3_ACCENDLMUy_RDA , CPU3_ACCENDLMUy_RDB , CPU3_ACCENDLMUy_VM , CPU3_ACCENDLMUy_PRS , CPU3_ACCENDLMUy_RGNLA , CPU3_ACCENDLMUy_RGNUA or CPU4_ACCENDLMUCFG_WRA , CPU4_ACCENDLMUCFG_WRB , CPU4_ACCENDLMUCFG_RDA , CPU4_ACCENDLMUCFG_RDB , CPU4_ACCENDLMUCFG_VM , CPU4_ACCENDLMUCFG_PRS , CPU4_ACCENDLMUCFG_RGNLA , CPU4_ACCENDLMUCFG_RGNUA , CPU4_ACCENDLMUy_WRA , CPU4_ACCENDLMUy_WRB , CPU4_ACCENDLMUy_RDA , CPU4_ACCENDLMUy_RDB , CPU4_ACCENDLMUy_VM , CPU4_ACCENDLMUy_PRS , CPU4_ACCENDLMUy_RGNLA , CPU4_ACCENDLMUy_RGNUA or CPU5_ACCENDLMUCFG_WRA , CPU5_ACCENDLMUCFG_WRB , CPU5_ACCENDLMUCFG_RDA , CPU5_ACCENDLMUCFG_RDB , CPU5_ACCENDLMUCFG_VM , CPU5_ACCENDLMUCFG_PRS , CPU5_ACCENDLMUCFG_RGNLA , CPU5_ACCENDLMUCFG_RGNUA , CPU5_ACCENDLMUy_WRA , CPU5_ACCENDLMUy_WRB , CPU5_ACCENDLMUy_RDA , CPU5_ACCENDLMUy_RDB , CPU5_ACCENDLMUy_VM , CPU5_ACCENDLMUy_PRS , CPU5_ACCENDLMUy_RGNLA , CPU5_ACCENDLMUy_RGNUA .
P3	Access protection using *APU-P3* registers.
32	Access only when using 32-bit width.
STP	Access only during start-up, before start-up protection is set (SMM_STCON.ST_PROT=1).
PROT	Access restrictions as defined in the PROT register access rules.
DEN	Description can be found in global access mode definition.
64	Access only when using 64-bit width.
nBE	Indicates that no Bus Error is generated when accessing this address range, even though it is either an access to an undefined address or the access does not follow the given rules.

4.10.4.1.5 Register overview - CPU0 domain SRI (ascending offset address)

Table 215 Register overview - CPU0 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU0_STM_CLC	CPU0 system timer clock control register	00000_H	P0, 32	P0, SV, SFRE, 32	System Reset	351
CPU0_STMHV_CLC	CPU0 system timer clock control register, hypervisor view	00000_H	P0, 32	P0, SV, SFRE, 32	System Reset	351
CPU0_STM_OCS	CPU0 OCDS control and status register	00004_H	P0, 32	P0, SV, 32, DEN	Debug Reset	352
CPU0_STMHV_OCS	CPU0 OCDS control and status register, hypervisor view	00004_H	P0, 32	P0, SV, 32, DEN	Debug Reset	352
CPU0_STM_ABS	CPU0 system timer absolute timer	00020_H	(P0, 32)| (P0, 64)	P0, SV, 64, SFRSE	System Reset	353
CPU0_STMHV_ABS	CPU0 system timer absolute timer, hypervisor view	00020_H	(P0, 32)| (P0, 64)	P0, SV, 64, SFRSE	System Reset	353
CPU0_STMHV_VMn_CMPc (c=0-1;n=0-7)	CPU0 VMn system timer compare register c, hypervisor view	00100_H+ n*20_H+c *4	P1, 32	P1, SV, 32	System Reset	354
CPU0_STMHV_VMn_CMCON (n=0-7)	CPU0 VMn system timer compare match control register, hypervisor view	00108_H+ n*20_H	P1, 32	P1, SV, 32	System Reset	355
CPU0_STMHV_VMn_ICR (n=0-7)	CPU0 VMn system timer interrupt control register, hypervisor view	$0010C_H$+ n*20_H	P1, 32	P1, SV, 32	System Reset	356
CPU0_STMHV_VMn_ISCR (n=0-7)	CPU0 VMn system timer interrupt set/clear register, hypervisor view	00110_H+ n*20_H	P1, 32	P1, SV, 32	System Reset	357
CPU0_STMHV_VMn_ISR (n=0-7)	CPU0 VMn system timer interrupt status register, hypervisor view	00114_H+ n*20_H	P1, 32	P1, SV, 32	System Reset	358
CPU0_STMHV_VMn_RELTIM (n=0-7)	CPU0 VMn system timer relative timer, hypervisor view	00118_H+ n*20_H	(P1, 32)| (P1, 64)	P1, SV, 64	System Reset	359
CPU0_STM_CMPc (c=0-1)	CPU0 system timer compare register c	00120_H+ c*4	P1, 32	P1, SV, 32	System Reset	354
CPU0_STM_CMCON	CPU0 system timer compare match control register	00128_H	P1, 32	P1, SV, 32	System Reset	355
CPU0_STM_ICR	CPU0 system timer interrupt control register	$0012C_H$	P1, 32	P1, SV, 32	System Reset	356

(table continues...)

Table 215 (continued) Register overview - CPU0 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU0_STM_ISCR	CPU0 system timer interrupt set/clear register	00130_H	P1, 32	P1, SV, 32	System Reset	357
CPU0_STM_ISR	CPU0 system timer interrupt status register	00134_H	P1, 32	P1, SV, 32	System Reset	358
CPU0_STM_RELTIM	CPU0 system timer relative timer	00138_H	(P1, 32)| (P1, 64)	P1, SV, 64	System Reset	359
CPU0_KRST0	CPU0 reset register 0	$0D000_H$	P0, 32	RSTE, P0, SV, 32	Kernel Reset	297
CPU0_KRST1	CPU0 reset register 1	$0D004_H$	P0, 32	RSTE, P0, SV, 32	Kernel Reset	298
CPU0_KRSTCLR	CPU0 reset clear register	$0D008_H$	P0, 32	RSTE, P0, SV, 32	Kernel Reset	299
CPU0_PROTSPRSE	CPU0 PROTSPR register safe endinit	$0E008_H$	32	SV, 32, PROT	Application Reset	562
CPU0_ACCENSPRCFG_WRA	CPU0 shadow SPR, write access enable register A	$0E020_H$	32	SPRSE, SV, 32	Application Reset	565
CPU0_ACCENSPRCFG_WRB	CPU0 shadow SPR, write access enable register B	$0E024_H$	32	SPRSE, SV, 32	Application Reset	566
CPU0_ACCENSPRCFG_RDA	CPU0 shadow SPR, read access enable register A	$0E028_H$	32	SPRSE, SV, 32	Application Reset	567
CPU0_ACCENSPRCFG_RDB	CPU0 shadow SPR, read access enable register B	$0E02C_H$	32	SPRSE, SV, 32	Application Reset	568
CPU0_ACCENSPRCFG_VM	CPU0 shadow SPR, VM access enable register	$0E030_H$	32	SPRSE, SV, 32	Application Reset	310
CPU0_ACCENSPRCFG_PRS	CPU0 shadow SPR, PRS access enable register	$0E034_H$	32	SPRSE, SV, 32	Application Reset	310
CPU0_ACCENSPRCFG_RGNLA	CPU0 shadow SPR, region lower address register	$0E038_H$	32	SPRSE, SV, 32	Application Reset	311
CPU0_ACCENSPRCFG_RGNUA	CPU0 shadow SPR, region upper address register	$0E03C_H$	32	SPRSE, SV, 32	Application Reset	312
CPU0_PROTDLMUSE	CPU0 PROTDLMU register safe endinit	$0E048_H$	32	SV, 32, PROT	Application Reset	312
CPU0_ACCENDLMUCFG_WRA	CPU0 shadow DLMU, write access enable register A	$0E060_H$	32	DLMUSE, SV, 32	Application Reset	314
CPU0_ACCENDLMUCFG_WRB	CPU0 shadow DLMU, write access enable register B	$0E064_H$	32	DLMUSE, SV, 32	Application Reset	315
CPU0_ACCENDLMUCFG_RDA	CPU0 shadow DLMU, read access enable register A	$0E068_H$	32	DLMUSE, SV, 32	Application Reset	315
CPU0_ACCENDLMUCFG_RDB	CPU0 shadow DLMU, read access enable register B	$0E06C_H$	32	DLMUSE, SV, 32	Application Reset	316
CPU0_ACCENDLMUCFG_VM	CPU0 shadow DLMU, VM access enable register	$0E070_H$	32	DLMUSE, SV, 32	Application Reset	316

(table continues...)

Table 215 (continued) Register overview - CPU0 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU0_ACCENDLMUCFG_PRS	CPU0 shadow DLMU, PRS access enable register	$0E074_H$	32	DLMUSE, SV, 32	Application Reset	317
CPU0_ACCENDLMUCFG_RGNLA	CPU0 shadow DLMU, region lower address register	$0E078_H$	32	DLMUSE, SV, 32	Application Reset	317
CPU0_ACCENDLMUCFG_RGNUA	CPU0 shadow DLMU, region upper address register	$0E07C_H$	32	DLMUSE, SV, 32	Application Reset	318
CPU0_PROTSFRE	CPU0 PROTSFR register endinit	$0E080_H$	32	SV, 32, PROT	Application Reset	569
CPU0_PROTSFRSE	CPU0 PROTSFR register safe endinit	$0E088_H$	32	SV, 32, PROT	Application Reset	571
CPU0_ACCENSFRCFG_WRA	CPU0 shadow SFR, write access enable register A	$0E0A0_H$	32	SFRSE, SV, 32	Application Reset	573
CPU0_ACCENSFRCFG_WRB	CPU0 shadow SFR, write access enable register B	$0E0A4_H$	32	SFRSE, SV, 32	Application Reset	574
CPU0_ACCENSFRCFG_RDA	CPU0 shadow SFR, read access enable register A	$0E0A8_H$	32	SFRSE, SV, 32	Application Reset	575
CPU0_ACCENSFRCFG_RDB	CPU0 shadow SFR, read access enable register B	$0E0AC_H$	32	SFRSE, SV, 32	Application Reset	576
CPU0_ACCENSFRCFG_VM	CPU0 shadow SFR, VM access enable register	$0E0B0_H$	32	SFRSE, SV, 32	Application Reset	325
CPU0_ACCENSFRCFG_PRS	CPU0 shadow SFR, PRS access enable register	$0E0B4_H$	32	SFRSE, SV, 32	Application Reset	325
CPU0_PROTRSTE	CPU0 PROTRST register endinit	$0E0C0_H$	32	SV, 32, PROT	See 577	577
CPU0_PROTSTMSE	CPU0 PROTSTM Register Safe Endinit	$0E0D8_H$	32	SV, 32, PROT	Application Reset	579
CPU0_ACCENSTMCFG_WRA	CPU0 shadow STM, write access enable register A	$0E0E0_H$	32	STMSE, SV, 32	Application Reset	582
CPU0_ACCENSTMCFG_WRB	CPU0 shadow STM, write access enable register B	$0E0E4_H$	32	STMSE, SV, 32	Application Reset	330
CPU0_ACCENSTMCFG_RDA	CPU0 shadow STM, read access enable register A	$0E0E8_H$	32	STMSE, SV, 32	Application Reset	583
CPU0_ACCENSTMCFG_RDB	CPU0 shadow STM, read access enable register B	$0E0EC_H$	32	STMSE, SV, 32	Application Reset	584
CPU0_ACCENSTMCFG_VM	CPU0 shadow STM, VM access enable register	$0E0F0_H$	32	STMSE, SV, 32	Application Reset	332
CPU0_ACCENSTMCFG_PRS	CPU0 shadow STM, PRS access enable register	$0E0F4_H$	32	STMSE, SV, 32	Application Reset	333
CPU0_ACCENSPRy_WRA (y=0-15)	CPU0 SPR range y, write access enable register A	$0E200_H$+ y*20_H	32	SPRSE, SV, 32	Application Reset	585

(table continues...)

Table 215 (continued) Register overview - CPU0 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU0_ACCENSPRy_WRB (y=0-15)	CPU0 SPR range y, write access enable register B	$0E204_H$+ y*20_H	32	SPRSE, SV, 32	Application Reset	586
CPU0_ACCENSPRy_RDA (y=0-15)	CPU0 SPR range y, read access enable register A	$0E208_H$+ y*20_H	32	SPRSE, SV, 32	Application Reset	587
CPU0_ACCENSPRy_RDB (y=0-15)	CPU0 SPR range y, read access enable register B	$0E20C_H$+ y*20_H	32	SPRSE, SV, 32	Application Reset	588
CPU0_ACCENSPRy_VM (y=0-15)	CPU0 SPR range y, VM access enable register	$0E210_H$+ y*20_H	32	SPRSE, SV, 32	Application Reset	336
CPU0_ACCENSPRy_PRS (y=0-15)	CPU0 SPR range y, PRS access enable register	$0E214_H$+ y*20_H	32	SPRSE, SV, 32	Application Reset	337
CPU0_ACCENSPRy_RGNLA (y=0-15)	CPU0 SPR range y, region lower address register	$0E218_H$+ y*20_H	32	SPRSE, SV, 32	Application Reset	337
CPU0_ACCENSPRy_RGNUA (y=0-15)	CPU0 SPR range y, region upper address register	$0E21C_H$+ y*20_H	32	SPRSE, SV, 32	Application Reset	338
CPU0_ACCENDLMUy_WRA (y=0-15)	CPU0 DLMU range y, write access enable register A	$0E400_H$+ y*20_H	32	DLMUSE, SV, 32	Application Reset	338
CPU0_ACCENDLMUy_WRB (y=0-15)	CPU0 DLMU range y, write access enable register B	$0E404_H$+ y*20_H	32	DLMUSE, SV, 32	Application Reset	339
CPU0_ACCENDLMUy_RDA (y=0-15)	CPU0 DLMU range y, read access enable register A	$0E408_H$+ y*20_H	32	DLMUSE, SV, 32	Application Reset	339
CPU0_ACCENDLMUy_RDB (y=0-15)	CPU0 DLMU range y, read access enable register B	$0E40C_H$+ y*20_H	32	DLMUSE, SV, 32	Application Reset	340
CPU0_ACCENDLMUy_VM (y=0-15)	CPU0 DLMU range y, VM access enable register	$0E410_H$+ y*20_H	32	DLMUSE, SV, 32	Application Reset	340
CPU0_ACCENDLMUy_PRS (y=0-15)	CPU0 DLMU range y, PRS access enable register	$0E414_H$+ y*20_H	32	DLMUSE, SV, 32	Application Reset	341
CPU0_ACCENDLMUy_RGNLA (y=0-15)	CPU0 DLMU range y, region lower address register	$0E418_H$+ y*20_H	32	DLMUSE, SV, 32	Application Reset	342

(table continues...)

Table 215 (continued) Register overview - CPU0 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU0_ACCENDLMUy_RGNUA (y=0-15)	CPU0 DLMU range y, region upper address register	$0E41C_H$+ y*20_H	32	DLMUSE, SV, 32	Application Reset	342
CPU0_ACCENSFR_WRA	CPU0 SFR, write access enable register A	$0E600_H$	32	SFRSE, SV, 32	Application Reset	589
CPU0_ACCENSFR_WRB	CPU0 SFR, write access enable register B	$0E604_H$	32	SFRSE, SV, 32	Application Reset	590
CPU0_ACCENSFR_RDA	CPU0 SFR, read access enable register A	$0E608_H$	32	SFRSE, SV, 32	Application Reset	591
CPU0_ACCENSFR_RDB	CPU0 SFR, read access enable register B	$0E60C_H$	32	SFRSE, SV, 32	Application Reset	592
CPU0_ACCENSFR_VM	CPU0 SFR, VM access enable register	$0E610_H$	32	SFRSE, SV, 32	Application Reset	345
CPU0_ACCENSFR_PRS	CPU0 SFR, PRS access enable register	$0E614_H$	32	SFRSE, SV, 32	Application Reset	346
CPU0_ACCENSTMVMy_WRA (y=0-7)	CPU0 VMy STM, write access enable register A, virtualization view	$0E700_H$+ y*20_H	32	STMSE, SV, 32	Application Reset	593
CPU0_ACCENSTMVMy_WRB (y=0-7)	CPU0 VMy STM, write access enable register B, virtualization view	$0E704_H$+ y*20_H	32	STMSE, SV, 32	Application Reset	347
CPU0_ACCENSTMVMy_RDA (y=0-7)	CPU0 VMy STM, read access enable register A, virtualization view	$0E708_H$+ y*20_H	32	STMSE, SV, 32	Application Reset	594
CPU0_ACCENSTMVMy_RDB (y=0-7)	CPU0 VMy STM, read access enable register B, virtualization view	$0E70C_H$+ y*20_H	32	STMSE, SV, 32	Application Reset	595
CPU0_ACCENSTMVMy_VM (y=0-7)	CPU0 VMy STM, VM access enable register, virtualization view	$0E710_H$+ y*20_H	32	STMSE, SV, 32	Application Reset	350
CPU0_ACCENSTMVMy_PRS (y=0-7)	CPU0 VMy STM, PRS access enable register, virtualization view	$0E714_H$+ y*20_H	32	STMSE, SV, 32	Application Reset	350
CPU0_OSEL	CPU0 overlay range select register	$0FB00_H$	P0, 32	P0, SV, 32	Kernel Reset	299
CPU0_RABRi (i=0-31)	CPU0 redirected address base register i	$0FB10_H$+ i*12	P0, 32	P0, SV, 32	Kernel Reset	300
CPU0_OTARi (i=0-31)	CPU0 overlay target address register i	$0FB14_H$+ i*12	P0, 32	P0, SV, 32	Kernel Reset	302
CPU0_OMASKi (i=0-31)	CPU0 overlay mask register i	$0FB18_H$+ i*12	P0, 32	P0, SV, 32	Kernel Reset	303

(table continues...)

Table 215 (continued) Register overview - CPU0 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU0_HRA_SEGEN	CPU0 SRI error generation register	11030_H	P0, 32	SFRE, P0, SV, 32	Application Reset	255
CPU0_HRA_LCLTEST	CPU0 lockstep test register	11040_H	P0, 32	P0, SV, 32	Cold PowerOn Reset	256
CPU0_HRA_PMA0	CPU0 data access cacheability register	18100_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	232
CPU0_HRA_PMA1	CPU0 code access cacheability register	18104_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	233
CPU0_HRA_PMA2	CPU0 peripheral space identifier register	18108_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	234
CPU0_HRA_DCON2	CPU0 data control register 2	19000_H	P0, 32	P0, SV, 32	Kernel Reset	558
CPU0_HRA_DCON1	CPU0 data control register 1	19008_H	P0, 32	P0, SV, 32	Kernel Reset	258
CPU0_HRA_SMACON	CPU0 SIST mode access control register	$1900C_H$	P0, 32	SFRSE, P0, SV, 32	Kernel Reset	234
CPU0_HRA_DSTR	CPU0 HR specific data synchronous trap register	19010_H	P0, 32	P0, SV, 32	Kernel Reset	259
CPU0_HRA_DATR	CPU0 HR specific data asynchronous trap register	19018_H	P0, 32	P0, SV, 32	Kernel Reset	261
CPU0_HRA_DEADD	CPU0 HR specific data error address register	$1901C_H$	P0, 32	P0, SV, 32	Kernel Reset	262
CPU0_HRA_DIEAR	CPU0 data integrity error address register	19020_H	P0, 32	P0, SV, 32	Kernel Reset	279
CPU0_HRA_DIETR	CPU0 data integrity error trap register	19024_H	P0, 32	P0, SV, 32	Kernel Reset	280
CPU0_HRA_DCON0	CPU0 data memory control register	19040_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	258
CPU0_HRA_PSTR	CPU0 HR specific program synchronous trap register	19200_H	P0, 32	P0, SV, 32	Kernel Reset	295
CPU0_HRA_PCON1	CPU0 program control 1	19204_H	P0, 32	P0, SV, 32	Kernel Reset	292
CPU0_HRA_PCON2	CPU0 program control 2	19208_H	P0, 32	P0, SV, 32	Kernel Reset	561
CPU0_HRA_PCON0	CPU0 program control 0	$1920C_H$	P0, 32	SFRE, P0, SV, 32	Kernel Reset	294
CPU0_HRA_PIEAR	CPU0 HR specific program integrity error address register	19210_H	P0, 32	P0, SV, 32	Kernel Reset	281
CPU0_HRA_PIETR	CPU0 HR specific program integrity error trap register	19214_H	P0, 32	P0, SV, 32	Kernel Reset	282

(table continues...)

Table 215 (continued) Register overview - CPU0 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU0_HRA_COMPAT	CPU0 compatibility control register	19400_H	P0, 32	SFRSE, P0, SV, 32	Kernel Reset	235
CPU0_HRA_FPU_TRAP_CON	CPU0 HR specific trap control register	$1A000_H$	P0, 32	P0, SV, 32	Kernel Reset	263
CPU0_HRA_FPU_TRAP_PC	CPU0 HR specific trapping instruction program counter register	$1A004_H$	P0, 32	P0, SV, 32	Kernel Reset	265
CPU0_HRA_FPU_TRAP_OPC	CPU0 HR specific trapping instruction opcode register	$1A008_H$	P0, 32	P0, SV, 32	Kernel Reset	266
CPU0_HRA_FPU_TRAP_SRC1_L	CPU0 HR specific trapping instruction operand register	$1A010_H$	P0, 32	P0, SV, 32	Kernel Reset	267
CPU0_HRA_FPU_TRAP_SRC1_U	CPU0 HR specific trapping instruction operand register	$1A014_H$	P0, 32	P0, SV, 32	Kernel Reset	268
CPU0_HRA_FPU_TRAP_SRC2_L	CPU0 HR specific trapping instruction operand register	$1A018_H$	P0, 32	P0, SV, 32	Kernel Reset	268
CPU0_HRA_FPU_TRAP_SRC2_U	CPU0 HR specific trapping instruction operand register	$1A01C_H$	P0, 32	P0, SV, 32	Kernel Reset	269
CPU0_HRA_FPU_TRAP_SRC3_L	CPU0 HR specific trapping instruction operand register	$1A020_H$	P0, 32	P0, SV, 32	Kernel Reset	270
CPU0_HRA_FPU_TRAP_SRC3_U	CPU0 HR specific trapping instruction operand register	$1A024_H$	P0, 32	P0, SV, 32	Kernel Reset	270
CPU0_HRA_FPU_SYNC_TRAP_CON	CPU0 HR specific synchronous trap control register	$1A030_H$	P0, 32	P0, SV, 32	Kernel Reset	271
CPU0_HRA_FPU_SYNC_TRAP_OPC	CPU0 HR specific trapping instruction opcode register	$1A034_H$	P0, 32	P0, SV, 32	Kernel Reset	273
CPU0_HRA_DPRy_L (y=0-23)	CPU0 HR specific data protection range y, lower bound register	$1C000_H$+y*8	P0, 32	P0, SV, 32	Kernel Reset	283
CPU0_HRA_DPRy_U (y=0-23)	CPU0 HR specific data protection range y, upper bound register	$1C004_H$+y*8	P0, 32	P0, SV, 32	Kernel Reset	284
CPU0_HRA_CPRy_L (y=0-15)	CPU0 HR specific code protection range y, lower bound register	$1D000_H$+y*8	P0, 32	P0, SV, 32	Kernel Reset	285

(table continues...)

Table 215 (continued) Register overview - CPU0 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU0_HRA_CPRy_U (y=0-15)	CPU0 HR specific code protection range y, upper bound register	$1D004_H$+y*8	P0, 32	P0, SV, 32	Kernel Reset	286
CPU0_HRA_CPXE_y (y=0-3)	CPU0 HR specific code protection execute enable register set y	$1E000_H$+y*4	P0, 32	P0, SV, 32	Kernel Reset	287
CPU0_HRA_DPRE_y (y=0-3)	CPU0 HR specific data protection read enable register set y	$1E010_H$+y*4	P0, 32	P0, SV, 32	Kernel Reset	288
CPU0_HRA_DPWE_y (y=0-3)	CPU0 HR specific data protection write enable register set y	$1E020_H$+y*4	P0, 32	P0, SV, 32	Kernel Reset	289
CPU0_HRA_CPXE_y (y=4-7)	CPU0 HR specific code protection execute enable register set y	$1E040_H$+(y-4)*4	P0, 32	P0, SV, 32	Kernel Reset	287
CPU0_HRA_DPRE_y (y=4-7)	CPU0 HR specific data protection read enable register set y	$1E050_H$+(y-4)*4	P0, 32	P0, SV, 32	Kernel Reset	288
CPU0_HRA_DPWE_y (y=4-7)	CPU0 HR specific data protection write enable register set y	$1E060_H$+(y-4)*4	P0, 32	P0, SV, 32	Kernel Reset	289
CPU0_HRA_TPS_CON	CPU0 HR specific temporal protection system control register	$1E400_H$	P0, 32	P0, SV, 32	Kernel Reset	296
CPU0_HRA_TPS_TIMERn (n=0-2)	CPU0 HR specific temporal protection system timer register n	$1E404_H$+n*4	P0, 32	P0, SV, 32	Kernel Reset	297
CPU0_HRA_TRnEVT (n=0-7)	CPU0 trigger event n	$1F000_H$+n*8	P0, 32	P0, SV, 32	Debug Reset	209
CPU0_HRA_TRnADR (n=0-7)	CPU0 trigger address n	$1F004_H$+n*8	P0, 32	P0, SV, 32	Debug Reset	210
CPU0_HRA_CCTRL	CPU0 HR specific counter control	$1FC00_H$	P0, 32	P0, SV, 32	Kernel Reset	226
CPU0_HRA_CCNT	CPU0 HR specific CPU clock cycle count	$1FC04_H$	P0, 32	P0, SV, 32	Kernel Reset	227
CPU0_HRA_ICNT	CPU0 HR specific instruction count	$1FC08_H$	P0, 32	P0, SV, 32	Kernel Reset	228
CPU0_HRA_M1CNT	CPU0 HR specific multi-count register 1	$1FC0C_H$	P0, 32	P0, SV, 32	Kernel Reset	229

(table continues...)

Table 215 (continued) Register overview - CPU0 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU0_HRA_M2CNT	CPU0 HR specific multi-count register 2	$1FC10_H$	P0, 32	P0, SV, 32	Kernel Reset	230
CPU0_HRA_M3CNT	CPU0 HR specific multi-count register 3	$1FC14_H$	P0, 32	P0, SV, 32	Kernel Reset	231
CPU0_HRA_DBGSR	CPU0 debug status register	$1FD00_H$	P0, 32	P0, SV, 32	Debug Reset	211
CPU0_HRA_EXEVT	CPU0 external event register	$1FD08_H$	P0, 32	P0, SV, 32	Debug Reset	212
CPU0_HRA_CREVT	CPU0 core register access event	$1FD0C_H$	P0, 32	P0, SV, 32	Debug Reset	213
CPU0_HRA_SWEVT	CPU0 software debug event	$1FD10_H$	P0, 32	P0, SV, 32	Debug Reset	214
CPU0_HRA_DBGACT	CPU0 debug action configuration	$1FD14_H$	P0, 32	P0, SV, 32	Debug Reset	215
CPU0_HRA_TRIG_ACC	CPU0 trigger event accumulator	$1FD30_H$	P0, 32	P0, SV, 32	Debug Reset	217
CPU0_HRA_DMS	CPU0 debug monitor start address	$1FD40_H$	P0, 32	P0, SV, 32	Debug Reset	534
CPU0_HRA_DCX	CPU0 debug context save area pointer	$1FD44_H$	P0, 32	P0, SV, 32	Debug Reset	536
CPU0_HRA_DBGTCR	CPU0 debug trap control	$1FD48_H$	P0, 32	P0, SV, 32	Application Reset	219
CPU0_HRA_DBGCFG	CPU0 debug configuration	$1FD4C_H$	P0, 32	P0, SV, 32	Debug Reset	220
CPU0_HRA_TRCCFG	CPU0 trace configuration	$1FD50_H$	P0, 32	P0, SV, 32	Debug Reset	221
CPU0_HRA_TRCFILT	CPU0 trace filter	$1FD54_H$	P0, 32	P0, SV, 32	Debug Reset	222
CPU0_HRA_TRCLIM	CPU0 trace bandwidth limiter	$1FD58_H$	P0, 32	P0, SV, 32	Debug Reset	224
CPU0_HRA_TS16PTCCTRL	CPU0 MCDS performance counter control register	$1FD60_H$	P0, 32	P0, SV, 32	Debug Reset	225
CPU0_HRA_PCXI	CPU0 previous context information register	$1FE00_H$	P0, 32	P0, SV, 32	Kernel Reset	236
CPU0_HRA_PSW	CPU0 HR specific program status word	$1FE04_H$	P0, 32	P0, SV, 32	Kernel Reset	237
CPU0_HRA_PC	CPU0 program counter	$1FE08_H$	P0, 32	P0, SV, 32	Kernel Reset	538
CPU0_HRA_CORECON	CPU0 HR specific core configuration register	$1FE14_H$	P0, 32	SFRSE, P0, SV, 32	Kernel Reset	250
CPU0_HRA_CPU_ID	CPU0 identification register	$1FE18_H$	P0, 32	P0, SV, 32	Kernel Reset	540

(table continues...)

Table 215 (continued) Register overview - CPU0 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU0_HRA_CORE_ID	CPU0 core identification register	$1FE1C_H$	P0, 32	P0, SV, 32	Kernel Reset	542
CPU0_HRA_BIV	CPU0 HR specific base interrupt vector table pointer	$1FE20_H$	P0, 32	SFRE, P0, SV, 32	Kernel Reset	251
CPU0_HRA_BTV	CPU0 HR specific base trap vector table pointer	$1FE24_H$	P0, 32	SFRE, P0, SV, 32	Kernel Reset	554
CPU0_HRA_ISP	CPU0 HR specific interrupt stack pointer	$1FE28_H$	P0, 32	SFRE, P0, SV, 32	Kernel Reset	253
CPU0_HRA_ICR	CPU0 interrupt control register	$1FE2C_H$	P0, 32	P0, SV, 32	Kernel Reset	242
CPU0_HRA_PPRS	CPU0 previous PRS	$1FE34_H$	P0, 32	P0, SV, 32	Kernel Reset	244
CPU0_HRA_FCX	CPU0 HR specific free CSA list head pointer	$1FE38_H$	P0, 32	P0, SV, 32	Kernel Reset	254
CPU0_HRA_LCX	CPU0 HR specific free CSA list limit pointer	$1FE3C_H$	P0, 32	P0, SV, 32	Kernel Reset	255
CPU0_HRA_SWID	CPU0 software identification	$1FE40_H$	P0, 32	P0, SV, 32	Kernel Reset	544
CPU0_HRA_CUS_ID	CPU0 customer ID register	$1FE50_H$	P0, 32	P0, SV, 32	Kernel Reset	546
CPU0_HRA_BOOTCON	CPU0 core boot configuration register	$1FE60_H$	P0, 32	P0, SV, 32	Kernel Reset	548
CPU0_HRA_LCLCON	CPU0 lockstep control register	$1FE64_H$	P0, 32	STP, SFRSE, P0, SV, 32	Cold PowerOn Reset	550
CPU0_HRA_CCON	CPU0 clock control register	$1FE68_H$	P0, 32	P0, SV, 32	Kernel Reset	248
CPU0_HRA_TCCON	CPU0 hardware configuration	$1FE6C_H$	P0, 32	P0, SV, 32	Kernel Reset	552
CPU0_Dy (y=0-15)	CPU0 data general purpose register y	$1FF00_H$+ y*4	P0, 32	P0, SV, 32	Kernel Reset	274
CPU0_Ay (y=0-15)	CPU0 address general purpose register y	$1FF80_H$+ y*4	P0, 32	P0, SV, 32	Kernel Reset	274
CPU0_HRB_SEGEN	CPU0 SRI error generation register	21030_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	255
CPU0_HRB_LCLTEST	CPU0 lockstep test register	21040_H	P0, 32	P0, SV, 32	Cold PowerOn Reset	256
CPU0_HRB_PMA0	CPU0 data access cacheability register	28100_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	232
CPU0_HRB_PMA1	CPU0 code access cacheability register	28104_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	233

(table continues...)

Table 215 (continued) Register overview - CPU0 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU0_HRB_PMA2	CPU0 peripheral space identifier register	28108_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	234
CPU0_HRB_DCON2	CPU0 data control register 2	29000_H	P0, 32	P0, SV, 32	Kernel Reset	558
CPU0_HRB_DCON1	CPU0 data control register 1	29008_H	P0, 32	P0, SV, 32	Kernel Reset	258
CPU0_HRB_SMACON	CPU0 SIST mode access control register	$2900C_H$	P0, 32	SFRSE, P0, SV, 32	Kernel Reset	234
CPU0_HRB_DSTR	CPU0 HR specific data synchronous trap register	29010_H	P0, 32	P0, SV, 32	Kernel Reset	259
CPU0_HRB_DATR	CPU0 HR specific data asynchronous trap register	29018_H	P0, 32	P0, SV, 32	Kernel Reset	261
CPU0_HRB_DEADD	CPU0 HR specific data error address register	$2901C_H$	P0, 32	P0, SV, 32	Kernel Reset	262
CPU0_HRB_DIEAR	CPU0 data integrity error address register	29020_H	P0, 32	P0, SV, 32	Kernel Reset	279
CPU0_HRB_DIETR	CPU0 data integrity error trap register	29024_H	P0, 32	P0, SV, 32	Kernel Reset	280
CPU0_HRB_DCON0	CPU0 data memory control register	29040_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	258
CPU0_HRB_PSTR	CPU0 HR specific program synchronous trap register	29200_H	P0, 32	P0, SV, 32	Kernel Reset	295
CPU0_HRB_PCON1	CPU0 program control 1	29204_H	P0, 32	P0, SV, 32	Kernel Reset	292
CPU0_HRB_PCON2	CPU0 program control 2	29208_H	P0, 32	P0, SV, 32	Kernel Reset	561
CPU0_HRB_PCON0	CPU0 program control 0	$2920C_H$	P0, 32	SFRE, P0, SV, 32	Kernel Reset	294
CPU0_HRB_PIEAR	CPU0 HR specific program integrity error address register	29210_H	P0, 32	P0, SV, 32	Kernel Reset	281
CPU0_HRB_PIETR	CPU0 HR specific program integrity error trap register	29214_H	P0, 32	P0, SV, 32	Kernel Reset	282
CPU0_HRB_COMPAT	CPU0 compatibility control register	29400_H	P0, 32	SFRSE, P0, SV, 32	Kernel Reset	235
CPU0_HRB_FPU_TRAP_CON	CPU0 HR specific trap control register	$2A000_H$	P0, 32	P0, SV, 32	Kernel Reset	263
CPU0_HRB_FPU_TRAP_PC	CPU0 HR specific trapping instruction program counter register	$2A004_H$	P0, 32	P0, SV, 32	Kernel Reset	265
CPU0_HRB_FPU_TRAP_OPC	CPU0 HR specific trapping instruction opcode register	$2A008_H$	P0, 32	P0, SV, 32	Kernel Reset	266

(table continues...)

Table 215 (continued) Register overview - CPU0 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU0_HRB_FPU_TRAP_SRC1_L	CPU0 HR specific trapping instruction operand register	$2A010_H$	P0, 32	P0, SV, 32	Kernel Reset	267
CPU0_HRB_FPU_TRAP_SRC1_U	CPU0 HR specific trapping instruction operand register	$2A014_H$	P0, 32	P0, SV, 32	Kernel Reset	268
CPU0_HRB_FPU_TRAP_SRC2_L	CPU0 HR specific trapping instruction operand register	$2A018_H$	P0, 32	P0, SV, 32	Kernel Reset	268
CPU0_HRB_FPU_TRAP_SRC2_U	CPU0 HR specific trapping instruction operand register	$2A01C_H$	P0, 32	P0, SV, 32	Kernel Reset	269
CPU0_HRB_FPU_TRAP_SRC3_L	CPU0 HR specific trapping instruction operand register	$2A020_H$	P0, 32	P0, SV, 32	Kernel Reset	270
CPU0_HRB_FPU_TRAP_SRC3_U	CPU0 HR specific trapping instruction operand register	$2A024_H$	P0, 32	P0, SV, 32	Kernel Reset	270
CPU0_HRB_FPU_SYNC_TRAP_CON	CPU0 HR specific synchronous trap control register	$2A030_H$	P0, 32	P0, SV, 32	Kernel Reset	271
CPU0_HRB_FPU_SYNC_TRAP_OPC	CPU0 HR specific trapping instruction opcode register	$2A034_H$	P0, 32	P0, SV, 32	Kernel Reset	273
CPU0_HRB_DPRy_L (y=0-23)	CPU0 HR specific data protection range y, lower bound register	$2C000_H$+y*8	P0, 32	P0, SV, 32	Kernel Reset	283
CPU0_HRB_DPRy_U (y=0-23)	CPU0 HR specific data protection range y, upper bound register	$2C004_H$+y*8	P0, 32	P0, SV, 32	Kernel Reset	284
CPU0_HRB_CPRy_L (y=0-15)	CPU0 HR specific code protection range y, lower bound register	$2D000_H$+y*8	P0, 32	P0, SV, 32	Kernel Reset	285
CPU0_HRB_CPRy_U (y=0-15)	CPU0 HR specific code protection range y, upper bound register	$2D004_H$+y*8	P0, 32	P0, SV, 32	Kernel Reset	286
CPU0_HRB_CPXE_y (y=0-3)	CPU0 HR specific code protection execute enable register set y	$2E000_H$+y*4	P0, 32	P0, SV, 32	Kernel Reset	287
CPU0_HRB_DPRE_y (y=0-3)	CPU0 HR specific data protection read enable register set y	$2E010_H$+y*4	P0, 32	P0, SV, 32	Kernel Reset	288

(table continues...)

Table 215 (continued) Register overview - CPU0 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU0_HRB_DPWE_y (y=0-3)	CPU0 HR specific data protection write enable register set y	$2E020_H$+y*4	P0, 32	P0, SV, 32	Kernel Reset	289
CPU0_HRB_CPXE_y (y=4-7)	CPU0 HR specific code protection execute enable register set y	$2E040_H$+(y-4)*4	P0, 32	P0, SV, 32	Kernel Reset	287
CPU0_HRB_DPRE_y (y=4-7)	CPU0 HR specific data protection read enable register set y	$2E050_H$+(y-4)*4	P0, 32	P0, SV, 32	Kernel Reset	288
CPU0_HRB_DPWE_y (y=4-7)	CPU0 HR specific data protection write enable register set y	$2E060_H$+(y-4)*4	P0, 32	P0, SV, 32	Kernel Reset	289
CPU0_HRB_TPS_CON	CPU0 HR specific temporal protection system control register	$2E400_H$	P0, 32	P0, SV, 32	Kernel Reset	296
CPU0_HRB_TPS_TIMERn (n=0-2)	CPU0 HR specific temporal protection system timer register n	$2E404_H$+n*4	P0, 32	P0, SV, 32	Kernel Reset	297
CPU0_HRB_TRnEVT (n=0-7)	CPU0 trigger event n	$2F000_H$+n*8	P0, 32	P0, SV, 32	Debug Reset	209
CPU0_HRB_TRnADR (n=0-7)	CPU0 trigger address n	$2F004_H$+n*8	P0, 32	P0, SV, 32	Debug Reset	210
CPU0_HRB_CCTRL	CPU0 HR specific counter control	$2FC00_H$	P0, 32	P0, SV, 32	Kernel Reset	226
CPU0_HRB_CCNT	CPU0 HR specific CPU clock cycle count	$2FC04_H$	P0, 32	P0, SV, 32	Kernel Reset	227
CPU0_HRB_ICNT	CPU0 HR specific instruction count	$2FC08_H$	P0, 32	P0, SV, 32	Kernel Reset	228
CPU0_HRB_M1CNT	CPU0 HR specific multi-count register 1	$2FC0C_H$	P0, 32	P0, SV, 32	Kernel Reset	229
CPU0_HRB_M2CNT	CPU0 HR specific multi-count register 2	$2FC10_H$	P0, 32	P0, SV, 32	Kernel Reset	230
CPU0_HRB_M3CNT	CPU0 HR specific multi-count register 3	$2FC14_H$	P0, 32	P0, SV, 32	Kernel Reset	231
CPU0_HRB_DBGSR	CPU0 debug status register	$2FD00_H$	P0, 32	P0, SV, 32	Debug Reset	211
CPU0_HRB_EXEVT	CPU0 external event register	$2FD08_H$	P0, 32	P0, SV, 32	Debug Reset	212
CPU0_HRB_CREVT	CPU0 core register access event	$2FD0C_H$	P0, 32	P0, SV, 32	Debug Reset	213

(table continues...)

Table 215 (continued) Register overview - CPU0 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU0_HRB_SWEVT	CPU0 software debug event	$2FD10_H$	P0, 32	P0, SV, 32	Debug Reset	214
CPU0_HRB_DBGACT	CPU0 debug action configuration	$2FD14_H$	P0, 32	P0, SV, 32	Debug Reset	215
CPU0_HRB_TRIG_ACC	CPU0 trigger event accumulator	$2FD30_H$	P0, 32	P0, SV, 32	Debug Reset	217
CPU0_HRB_DMS	CPU0 debug monitor start address	$2FD40_H$	P0, 32	P0, SV, 32	Debug Reset	534
CPU0_HRB_DCX	CPU0 debug context save area pointer	$2FD44_H$	P0, 32	P0, SV, 32	Debug Reset	536
CPU0_HRB_DBGTCR	CPU0 debug trap control	$2FD48_H$	P0, 32	P0, SV, 32	Application Reset	219
CPU0_HRB_DBGCFG	CPU0 debug configuration	$2FD4C_H$	P0, 32	P0, SV, 32	Debug Reset	220
CPU0_HRB_TRCCFG	CPU0 trace configuration	$2FD50_H$	P0, 32	P0, SV, 32	Debug Reset	221
CPU0_HRB_TRCFILT	CPU0 trace filter	$2FD54_H$	P0, 32	P0, SV, 32	Debug Reset	222
CPU0_HRB_TRCLIM	CPU0 trace bandwidth limiter	$2FD58_H$	P0, 32	P0, SV, 32	Debug Reset	224
CPU0_HRB_TS16PTCCTRL	CPU0 MCDS performance counter control register	$2FD60_H$	P0, 32	P0, SV, 32	Debug Reset	225
CPU0_HRB_PCXI	CPU0 previous context information register	$2FE00_H$	P0, 32	P0, SV, 32	Kernel Reset	236
CPU0_HRB_PSW	CPU0 HR specific program status word	$2FE04_H$	P0, 32	P0, SV, 32	Kernel Reset	237
CPU0_HRB_PC	CPU0 program counter	$2FE08_H$	P0, 32	P0, SV, 32	Kernel Reset	538
CPU0_HRB_CORECON	CPU0 HR specific core configuration register	$2FE14_H$	P0, 32	SFRSE, P0, SV, 32	Kernel Reset	250
CPU0_HRB_CPU_ID	CPU0 identification register	$2FE18_H$	P0, 32	P0, SV, 32	Kernel Reset	540
CPU0_HRB_CORE_ID	CPU0 core identification register	$2FE1C_H$	P0, 32	P0, SV, 32	Kernel Reset	542
CPU0_HRB_BIV	CPU0 HR specific base interrupt vector table pointer	$2FE20_H$	P0, 32	SFRE, P0, SV, 32	Kernel Reset	251
CPU0_HRB_BTV	CPU0 HR specific base trap vector table pointer	$2FE24_H$	P0, 32	SFRE, P0, SV, 32	Kernel Reset	555
CPU0_HRB_ISP	CPU0 HR specific interrupt stack pointer	$2FE28_H$	P0, 32	SFRE, P0, SV, 32	Kernel Reset	253
CPU0_HRB_ICR	CPU0 interrupt control register	$2FE2C_H$	P0, 32	P0, SV, 32	Kernel Reset	242

(table continues...)

Table 215 (continued) Register overview - CPU0 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU0_HRB_PPRS	CPU0 previous PRS	$2FE34_H$	P0, 32	P0, SV, 32	Kernel Reset	244
CPU0_HRB_FCX	CPU0 HR specific free CSA list head pointer	$2FE38_H$	P0, 32	P0, SV, 32	Kernel Reset	254
CPU0_HRB_LCX	CPU0 HR specific free CSA list limit pointer	$2FE3C_H$	P0, 32	P0, SV, 32	Kernel Reset	255
CPU0_HRB_SWID	CPU0 software identification	$2FE40_H$	P0, 32	P0, SV, 32	Kernel Reset	544
CPU0_HRB_CUS_ID	CPU0 customer ID register	$2FE50_H$	P0, 32	P0, SV, 32	Kernel Reset	546
CPU0_HRB_BOOTCON	CPU0 core boot configuration register	$2FE60_H$	P0, 32	P0, SV, 32	Kernel Reset	548
CPU0_HRB_LCLCON	CPU0 lockstep control register	$2FE64_H$	P0, 32	STP, SFRSE, P0, SV, 32	Cold PowerOn Reset	550
CPU0_HRB_CCON	CPU0 clock control register	$2FE68_H$	P0, 32	P0, SV, 32	Kernel Reset	248
CPU0_HRB_TCCON	CPU0 hardware configuration	$2FE6C_H$	P0, 32	P0, SV, 32	Kernel Reset	552
CPU0_HRB_Ay (y=0-1)	CPU0 HR specific address general purpose register y	$2FF80_H$+ y*4	P0, 32	P0, SV, 32	Kernel Reset	274
CPU0_HRB_Ay (y=8-9)	CPU0 HR specific address general purpose register y	$2FFA0_H$+ (y-8)*4	P0, 32	P0, SV, 32	Kernel Reset	274
CPU0_HRHV_SEGEN	CPU0 SRI error generation register	31030_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	255
CPU0_HRHV_LCLTEST	CPU0 lockstep test register	31040_H	P0, 32	P0, SV, 32	Cold PowerOn Reset	256
CPU0_HRHV_PMA0	CPU0 data access cacheability register	38100_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	232
CPU0_HRHV_PMA1	CPU0 code access cacheability register	38104_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	233
CPU0_HRHV_PMA2	CPU0 peripheral space identifier register	38108_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	234
CPU0_HRHV_DCON2	CPU0 data control register 2	39000_H	P0, 32	P0, SV, 32	Kernel Reset	558
CPU0_HRHV_DCON1	CPU0 data control register 1	39008_H	P0, 32	P0, SV, 32	Kernel Reset	258
CPU0_HRHV_SMACON	CPU0 SIST mode access control register	$3900C_H$	P0, 32	SFRSE, P0, SV, 32	Kernel Reset	234
CPU0_HRHV_DSTR	CPU0 HR specific data synchronous trap register	39010_H	P0, 32	P0, SV, 32	Kernel Reset	259

(table continues...)

Table 215 (continued) Register overview - CPU0 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU0_HRHV_DATR	CPU0 HR specific data asynchronous trap register	39018_H	P0, 32	P0, SV, 32	Kernel Reset	261
CPU0_HRHV_DEADD	CPU0 HR specific data error address register	$3901C_H$	P0, 32	P0, SV, 32	Kernel Reset	262
CPU0_HRHV_DIEAR	CPU0 data integrity error address register	39020_H	P0, 32	P0, SV, 32	Kernel Reset	279
CPU0_HRHV_DIETR	CPU0 data integrity error trap register	39024_H	P0, 32	P0, SV, 32	Kernel Reset	280
CPU0_HRHV_DCON0	CPU0 data memory control register	39040_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	258
CPU0_HRHV_PSTR	CPU0 HR specific program synchronous trap register	39200_H	P0, 32	P0, SV, 32	Kernel Reset	295
CPU0_HRHV_PCON1	CPU0 program control 1	39204_H	P0, 32	P0, SV, 32	Kernel Reset	292
CPU0_HRHV_PCON2	CPU0 program control 2	39208_H	P0, 32	P0, SV, 32	Kernel Reset	561
CPU0_HRHV_PCON0	CPU0 program control 0	$3920C_H$	P0, 32	SFRE, P0, SV, 32	Kernel Reset	294
CPU0_HRHV_PIEAR	CPU0 HR specific program integrity error address register	39210_H	P0, 32	P0, SV, 32	Kernel Reset	281
CPU0_HRHV_PIETR	CPU0 HR specific program integrity error trap register	39214_H	P0, 32	P0, SV, 32	Kernel Reset	282
CPU0_HRHV_COMPAT	CPU0 compatibility control register	39400_H	P0, 32	SFRSE, P0, SV, 32	Kernel Reset	235
CPU0_HRHV_FPU_TRAP_CON	CPU0 HR specific trap control register	$3A000_H$	P0, 32	P0, SV, 32	Kernel Reset	263
CPU0_HRHV_FPU_TRAP_PC	CPU0 HR specific trapping instruction program counter register	$3A004_H$	P0, 32	P0, SV, 32	Kernel Reset	265
CPU0_HRHV_FPU_TRAP_OPC	CPU0 HR specific trapping instruction opcode register	$3A008_H$	P0, 32	P0, SV, 32	Kernel Reset	266
CPU0_HRHV_FPU_TRAP_SRC1_L	CPU0 HR specific trapping instruction operand register	$3A010_H$	P0, 32	P0, SV, 32	Kernel Reset	267
CPU0_HRHV_FPU_TRAP_SRC1_U	CPU0 HR specific trapping instruction operand register	$3A014_H$	P0, 32	P0, SV, 32	Kernel Reset	268
CPU0_HRHV_FPU_TRAP_SRC2_L	CPU0 HR specific trapping instruction operand register	$3A018_H$	P0, 32	P0, SV, 32	Kernel Reset	268

(table continues...)

Table 215 (continued) Register overview - CPU0 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU0_HRHV_FPU_TRAP_SRC2_U	CPU0 HR specific trapping instruction operand register	$3A01C_H$	P0, 32	P0, SV, 32	Kernel Reset	269
CPU0_HRHV_FPU_TRAP_SRC3_L	CPU0 HR specific trapping instruction operand register	$3A020_H$	P0, 32	P0, SV, 32	Kernel Reset	270
CPU0_HRHV_FPU_TRAP_SRC3_U	CPU0 HR specific trapping instruction operand register	$3A024_H$	P0, 32	P0, SV, 32	Kernel Reset	270
CPU0_HRHV_FPU_SYNC_TRAP_CON	CPU0 HR specific synchronous trap control register	$3A030_H$	P0, 32	P0, SV, 32	Kernel Reset	271
CPU0_HRHV_FPU_SYNC_TRAP_OPC	CPU0 HR specific trapping instruction opcode register	$3A034_H$	P0, 32	P0, SV, 32	Kernel Reset	273
CPU0_HRHV_VCON0	CPU0 virtualization control register 0	$3B000_H$	P0, 32	P0, SV, 32	Kernel Reset	275
CPU0_HRHV_VCON1	CPU0 virtualization control register 1	$3B004_H$	P0, 32	P0, SV, 32	Kernel Reset	276
CPU0_HRHV_VCON2	CPU0 virtualization control register 2	$3B008_H$	P0, 32	P0, SV, 32	Kernel Reset	276
CPU0_HRHV_BHV	CPU0 base hypervisor vector table pointer	$3B010_H$	P0, 32	P0, SV, 32, SFRE	Kernel Reset	559
CPU0_HRHV_VMn_ICR (n=0-7)	CPU0 Virtual machine n interrupt state	$3B100_H$+n*4	P0, 32	P0, SV, 32	Kernel Reset	278
CPU0_HRHV_VMn_PETHRESH (n=0-7)	CPU0 Virtual machine n pre-emption threshold	$3B200_H$+n*4	P0, 32	P0, SV, 32	Kernel Reset	279
CPU0_HRHV_DPRy_L (y=0-23)	CPU0 HR specific data protection range y, lower bound register	$3C000_H$+y*8	P0, 32	P0, SV, 32	Kernel Reset	283
CPU0_HRHV_DPRy_U (y=0-23)	CPU0 HR specific data protection range y, upper bound register	$3C004_H$+y*8	P0, 32	P0, SV, 32	Kernel Reset	284
CPU0_HRHV_CPRy_L (y=0-15)	CPU0 HR specific code protection range y, lower bound register	$3D000_H$+y*8	P0, 32	P0, SV, 32	Kernel Reset	285
CPU0_HRHV_CPRy_U (y=0-15)	CPU0 HR specific code protection range y, upper bound register	$3D004_H$+y*8	P0, 32	P0, SV, 32	Kernel Reset	286

(table continues...)

Table 215 (continued) Register overview - CPU0 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU0_HRHV_CPXE_y (y=0-3)	CPU0 HR specific code protection execute enable register set y	$3E000_H$+y*4	P0, 32	P0, SV, 32	Kernel Reset	287
CPU0_HRHV_DPRE_y (y=0-3)	CPU0 HR specific data protection read enable register set y	$3E010_H$+y*4	P0, 32	P0, SV, 32	Kernel Reset	288
CPU0_HRHV_DPWE_y (y=0-3)	CPU0 HR specific data protection write enable register set y	$3E020_H$+y*4	P0, 32	P0, SV, 32	Kernel Reset	289
CPU0_HRHV_CPXE_y (y=4-7)	CPU0 HR specific code protection execute enable register set y	$3E040_H$+(y-4)*4	P0, 32	P0, SV, 32	Kernel Reset	287
CPU0_HRHV_DPRE_y (y=4-7)	CPU0 HR specific data protection read enable register set y	$3E050_H$+(y-4)*4	P0, 32	P0, SV, 32	Kernel Reset	288
CPU0_HRHV_DPWE_y (y=4-7)	CPU0 HR specific data protection write enable register set y	$3E060_H$+(y-4)*4	P0, 32	P0, SV, 32	Kernel Reset	289
CPU0_HRHV_TPS_CON	CPU0 HR specific temporal protection system control register	$3E400_H$	P0, 32	P0, SV, 32	Kernel Reset	296
CPU0_HRHV_TPS_TIMERn (n=0-2)	CPU0 HR specific temporal protection system timer register n	$3E404_H$+n*4	P0, 32	P0, SV, 32	Kernel Reset	297
CPU0_HRHV_TRnEVT (n=0-7)	CPU0 trigger event n	$3F000_H$+n*8	P0, 32	P0, SV, 32	Debug Reset	209
CPU0_HRHV_TRnADR (n=0-7)	CPU0 trigger address n	$3F004_H$+n*8	P0, 32	P0, SV, 32	Debug Reset	210
CPU0_HRHV_CCTRL	CPU0 HR specific counter control	$3FC00_H$	P0, 32	P0, SV, 32	Kernel Reset	226
CPU0_HRHV_CCNT	CPU0 HR specific CPU clock cycle count	$3FC04_H$	P0, 32	P0, SV, 32	Kernel Reset	227
CPU0_HRHV_ICNT	CPU0 HR specific instruction count	$3FC08_H$	P0, 32	P0, SV, 32	Kernel Reset	228
CPU0_HRHV_M1CNT	CPU0 HR specific multi-count register 1	$3FC0C_H$	P0, 32	P0, SV, 32	Kernel Reset	229
CPU0_HRHV_M2CNT	CPU0 HR specific multi-count register 2	$3FC10_H$	P0, 32	P0, SV, 32	Kernel Reset	230
CPU0_HRHV_M3CNT	CPU0 HR specific multi-count register 3	$3FC14_H$	P0, 32	P0, SV, 32	Kernel Reset	231

(table continues...)

Table 215 (continued) Register overview - CPU0 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU0_HRHV_DBGSR	CPU0 debug status register	$3FD00_H$	P0, 32	P0, SV, 32	Debug Reset	211
CPU0_HRHV_EXEVT	CPU0 external event register	$3FD08_H$	P0, 32	P0, SV, 32	Debug Reset	212
CPU0_HRHV_CREVT	CPU0 core register access event	$3FD0C_H$	P0, 32	P0, SV, 32	Debug Reset	213
CPU0_HRHV_SWEVT	CPU0 software debug event	$3FD10_H$	P0, 32	P0, SV, 32	Debug Reset	214
CPU0_HRHV_DBGACT	CPU0 debug action configuration	$3FD14_H$	P0, 32	P0, SV, 32	Debug Reset	215
CPU0_HRHV_TRIG_ACC	CPU0 trigger event accumulator	$3FD30_H$	P0, 32	P0, SV, 32	Debug Reset	217
CPU0_HRHV_DMS	CPU0 debug monitor start address	$3FD40_H$	P0, 32	P0, SV, 32	Debug Reset	534
CPU0_HRHV_DCX	CPU0 debug context save area pointer	$3FD44_H$	P0, 32	P0, SV, 32	Debug Reset	536
CPU0_HRHV_DBGTCR	CPU0 debug trap control	$3FD48_H$	P0, 32	P0, SV, 32	Application Reset	219
CPU0_HRHV_DBGCFG	CPU0 debug configuration	$3FD4C_H$	P0, 32	P0, SV, 32	Debug Reset	220
CPU0_HRHV_TRCCFG	CPU0 trace configuration	$3FD50_H$	P0, 32	P0, SV, 32	Debug Reset	221
CPU0_HRHV_TRCFILT	CPU0 trace filter	$3FD54_H$	P0, 32	P0, SV, 32	Debug Reset	222
CPU0_HRHV_TRCLIM	CPU0 trace bandwidth limiter	$3FD58_H$	P0, 32	P0, SV, 32	Debug Reset	224
CPU0_HRHV_TS16PTCCTRL	CPU0 MCDS performance counter control register	$3FD60_H$	P0, 32	P0, SV, 32	Debug Reset	225
CPU0_HRHV_PCXI	CPU0 previous context information register	$3FE00_H$	P0, 32	P0, SV, 32	Kernel Reset	236
CPU0_HRHV_PSW	CPU0 HR specific program status word	$3FE04_H$	P0, 32	P0, SV, 32	Kernel Reset	237
CPU0_HRHV_PC	CPU0 program counter	$3FE08_H$	P0, 32	P0, SV, 32	Kernel Reset	538
CPU0_HRHV_CORECON	CPU0 HR specific core configuration register	$3FE14_H$	P0, 32	SFRSE, P0, SV, 32	Kernel Reset	250
CPU0_HRHV_CPU_ID	CPU0 identification register	$3FE18_H$	P0, 32	P0, SV, 32	Kernel Reset	540
CPU0_HRHV_CORE_ID	CPU0 core identification register	$3FE1C_H$	P0, 32	P0, SV, 32	Kernel Reset	542
CPU0_HRHV_BIV	CPU0 HR specific base interrupt vector table pointer	$3FE20_H$	P0, 32	SFRE, P0, SV, 32	Kernel Reset	251

(table continues...)

Table 215 (continued) Register overview - CPU0 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU0_HRHV_BTV	CPU0 HR specific base trap vector table pointer	$3FE24_H$	P0, 32	SFRE, P0, SV, 32	Kernel Reset	555
CPU0_HRHV_ISP	CPU0 HR specific interrupt stack pointer	$3FE28_H$	P0, 32	SFRE, P0, SV, 32	Kernel Reset	253
CPU0_HRHV_ICR	CPU0 interrupt control register	$3FE2C_H$	P0, 32	P0, SV, 32	Kernel Reset	242
CPU0_HRHV_PPRS	CPU0 previous PRS	$3FE34_H$	P0, 32	P0, SV, 32	Kernel Reset	244
CPU0_HRHV_FCX	CPU0 HR specific free CSA list head pointer	$3FE38_H$	P0, 32	P0, SV, 32	Kernel Reset	254
CPU0_HRHV_LCX	CPU0 HR specific free CSA list limit pointer	$3FE3C_H$	P0, 32	P0, SV, 32	Kernel Reset	255
CPU0_HRHV_SWID	CPU0 software identification	$3FE40_H$	P0, 32	P0, SV, 32	Kernel Reset	544
CPU0_HRHV_CUS_ID	CPU0 customer ID register	$3FE50_H$	P0, 32	P0, SV, 32	Kernel Reset	546
CPU0_HRHV_BOOTCON	CPU0 core boot configuration register	$3FE60_H$	P0, 32	P0, SV, 32	Kernel Reset	548
CPU0_HRHV_LCLCON	CPU0 lockstep control register	$3FE64_H$	P0, 32	STP, SFRSE, P0, SV, 32	Cold PowerOn Reset	550
CPU0_HRHV_CCON	CPU0 clock control register	$3FE68_H$	P0, 32	P0, SV, 32	Kernel Reset	248
CPU0_HRHV_TCCON	CPU0 hardware configuration	$3FE6C_H$	P0, 32	P0, SV, 32	Kernel Reset	552
CPU0_HRHV_Ay (y=0-1)	CPU0 HR specific address general purpose register y	$3FF80_H$+ y*4	P0, 32	P0, SV, 32	Kernel Reset	274
CPU0_HRHV_Ay (y=8-9)	CPU0 HR specific address general purpose register y	$3FFA0_H$+ (y-8)*4	P0, 32	P0, SV, 32	Kernel Reset	274

4.10.4.1.6 Register overview - CPU1 domain SRI (ascending offset address)

Table 216 Register overview - CPU1 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU1_STM_CLC	CPU1 system timer clock control register	00000_H	P0, 32	P0, SV, SFRE, 32	System Reset	351
CPU1_STMHV_CLC	CPU1 system timer clock control register, hypervisor view	00000_H	P0, 32	P0, SV, SFRE, 32	System Reset	351
CPU1_STM_OCS	CPU1 OCDS control and status register	00004_H	P0, 32	P0, SV, 32, DEN	Debug Reset	352

(table continues...)

Table 216 (continued) Register overview - CPU1 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU1_STMHV_OCS	CPU1 OCDS control and status register, hypervisor view	00004_H	P0, 32	P0, SV, 32, DEN	Debug Reset	352
CPU1_STM_ABS	CPU1 system timer absolute timer	00020_H	(P0, 32)|(P0, 64)	P0, SV, 64, SFRSE	System Reset	353
CPU1_STMHV_ABS	CPU1 system timer absolute timer, hypervisor view	00020_H	(P0, 32)|(P0, 64)	P0, SV, 64, SFRSE	System Reset	353
CPU1_STMHV_VMn_CMPc (c=0-1;n=0-7)	CPU1 VMn system timer compare register c, hypervisor view	00100_H+n*20_H+c*4	P1, 32	P1, SV, 32	System Reset	354
CPU1_STMHV_VMn_CMCON (n=0-7)	CPU1 VMn system timer compare match control register, hypervisor view	00108_H+n*20_H	P1, 32	P1, SV, 32	System Reset	355
CPU1_STMHV_VMn_ICR (n=0-7)	CPU1 VMn system timer interrupt control register, hypervisor view	$0010C_H$+n*20_H	P1, 32	P1, SV, 32	System Reset	356
CPU1_STMHV_VMn_ISCR (n=0-7)	CPU1 VMn system timer interrupt set/clear register, hypervisor view	00110_H+n*20_H	P1, 32	P1, SV, 32	System Reset	357
CPU1_STMHV_VMn_ISR (n=0-7)	CPU1 VMn system timer interrupt status register, hypervisor view	00114_H+n*20_H	P1, 32	P1, SV, 32	System Reset	358
CPU1_STMHV_VMn_RELTIM (n=0-7)	CPU1 VMn system timer relative timer, hypervisor view	00118_H+n*20_H	(P1, 32)|(P1, 64)	P1, SV, 64	System Reset	359
CPU1_STM_CMPc (c=0-1)	CPU1 system timer compare register c	00120_H+c*4	P1, 32	P1, SV, 32	System Reset	354
CPU1_STM_CMCON	CPU1 system timer compare match control register	00128_H	P1, 32	P1, SV, 32	System Reset	355
CPU1_STM_ICR	CPU1 system timer interrupt control register	$0012C_H$	P1, 32	P1, SV, 32	System Reset	356
CPU1_STM_ISCR	CPU1 system timer interrupt set/clear register	00130_H	P1, 32	P1, SV, 32	System Reset	357
CPU1_STM_ISR	CPU1 system timer interrupt status register	00134_H	P1, 32	P1, SV, 32	System Reset	358
CPU1_STM_RELTIM	CPU1 system timer relative timer	00138_H	(P1, 32)|(P1, 64)	P1, SV, 64	System Reset	359
CPU1_KRST0	CPU1 reset register 0	$0D000_H$	P0, 32	RSTE, P0, SV, 32	Kernel Reset	297

(table continues...)

Table 216 (continued) Register overview - CPU1 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU1_KRST1	CPU1 reset register 1	$0D004_H$	P0, 32	RSTE, P0, SV, 32	Kernel Reset	298
CPU1_KRSTCLR	CPU1 reset clear register	$0D008_H$	P0, 32	RSTE, P0, SV, 32	Kernel Reset	299
CPU1_PROTSPRSE	CPU1 PROTSPR register safe endinit	$0E008_H$	32	SV, 32, PROT	Application Reset	562
CPU1_ACCENSPRCFG_WRA	CPU1 shadow SPR, write access enable register A	$0E020_H$	32	SPRSE, SV, 32	Application Reset	565
CPU1_ACCENSPRCFG_WRB	CPU1 shadow SPR, write access enable register B	$0E024_H$	32	SPRSE, SV, 32	Application Reset	566
CPU1_ACCENSPRCFG_RDA	CPU1 shadow SPR, read access enable register A	$0E028_H$	32	SPRSE, SV, 32	Application Reset	567
CPU1_ACCENSPRCFG_RDB	CPU1 shadow SPR, read access enable register B	$0E02C_H$	32	SPRSE, SV, 32	Application Reset	568
CPU1_ACCENSPRCFG_VM	CPU1 shadow SPR, VM access enable register	$0E030_H$	32	SPRSE, SV, 32	Application Reset	310
CPU1_ACCENSPRCFG_PRS	CPU1 shadow SPR, PRS access enable register	$0E034_H$	32	SPRSE, SV, 32	Application Reset	310
CPU1_ACCENSPRCFG_RGNLA	CPU1 shadow SPR, region lower address register	$0E038_H$	32	SPRSE, SV, 32	Application Reset	311
CPU1_ACCENSPRCFG_RGNUA	CPU1 shadow SPR, region upper address register	$0E03C_H$	32	SPRSE, SV, 32	Application Reset	312
CPU1_PROTDLMUSE	CPU1 PROTDLMU register safe endinit	$0E048_H$	32	SV, 32, PROT	Application Reset	312
CPU1_ACCENDLMUCFG_WRA	CPU1 shadow DLMU, write access enable register A	$0E060_H$	32	DLMUSE, SV, 32	Application Reset	314
CPU1_ACCENDLMUCFG_WRB	CPU1 shadow DLMU, write access enable register B	$0E064_H$	32	DLMUSE, SV, 32	Application Reset	315
CPU1_ACCENDLMUCFG_RDA	CPU1 shadow DLMU, read access enable register A	$0E068_H$	32	DLMUSE, SV, 32	Application Reset	315
CPU1_ACCENDLMUCFG_RDB	CPU1 shadow DLMU, read access enable register B	$0E06C_H$	32	DLMUSE, SV, 32	Application Reset	316
CPU1_ACCENDLMUCFG_VM	CPU1 shadow DLMU, VM access enable register	$0E070_H$	32	DLMUSE, SV, 32	Application Reset	316
CPU1_ACCENDLMUCFG_PRS	CPU1 shadow DLMU, PRS access enable register	$0E074_H$	32	DLMUSE, SV, 32	Application Reset	317
CPU1_ACCENDLMUCFG_RGNLA	CPU1 shadow DLMU, region lower address register	$0E078_H$	32	DLMUSE, SV, 32	Application Reset	317
CPU1_ACCENDLMUCFG_RGNUA	CPU1 shadow DLMU, region upper address register	$0E07C_H$	32	DLMUSE, SV, 32	Application Reset	318
CPU1_PROTSFRE	CPU1 PROTSFR register endinit	$0E080_H$	32	SV, 32, PROT	Application Reset	569

(table continues...)

Table 216 (continued) Register overview - CPU1 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU1_PROTSFRSE	CPU1 PROTSFR register safe endinit	$0E088_H$	32	SV, 32, PROT	Application Reset	571
CPU1_ACCENSFRCFG_WRA	CPU1 shadow SFR, write access enable register A	$0E0A0_H$	32	SFRSE, SV, 32	Application Reset	573
CPU1_ACCENSFRCFG_WRB	CPU1 shadow SFR, write access enable register B	$0E0A4_H$	32	SFRSE, SV, 32	Application Reset	574
CPU1_ACCENSFRCFG_RDA	CPU1 shadow SFR, read access enable register A	$0E0A8_H$	32	SFRSE, SV, 32	Application Reset	575
CPU1_ACCENSFRCFG_RDB	CPU1 shadow SFR, read access enable register B	$0E0AC_H$	32	SFRSE, SV, 32	Application Reset	576
CPU1_ACCENSFRCFG_VM	CPU1 shadow SFR, VM access enable register	$0E0B0_H$	32	SFRSE, SV, 32	Application Reset	325
CPU1_ACCENSFRCFG_PRS	CPU1 shadow SFR, PRS access enable register	$0E0B4_H$	32	SFRSE, SV, 32	Application Reset	325
CPU1_PROTRSTE	CPU1 PROTRST register endinit	$0E0C0_H$	32	SV, 32, PROT	Application Reset	577
CPU1_PROTSTMSE	CPU1 PROTSTM Register Safe Endinit	$0E0D8_H$	32	SV, 32, PROT	Application Reset	579
CPU1_ACCENSTMCFG_WRA	CPU1 shadow STM, write access enable register A	$0E0E0_H$	32	STMSE, SV, 32	Application Reset	582
CPU1_ACCENSTMCFG_WRB	CPU1 shadow STM, write access enable register B	$0E0E4_H$	32	STMSE, SV, 32	Application Reset	330
CPU1_ACCENSTMCFG_RDA	CPU1 shadow STM, read access enable register A	$0E0E8_H$	32	STMSE, SV, 32	Application Reset	583
CPU1_ACCENSTMCFG_RDB	CPU1 shadow STM, read access enable register B	$0E0EC_H$	32	STMSE, SV, 32	Application Reset	584
CPU1_ACCENSTMCFG_VM	CPU1 shadow STM, VM access enable register	$0E0F0_H$	32	STMSE, SV, 32	Application Reset	332
CPU1_ACCENSTMCFG_PRS	CPU1 shadow STM, PRS access enable register	$0E0F4_H$	32	STMSE, SV, 32	Application Reset	333
CPU1_ACCENSPRy_WRA (y=0-15)	CPU1 SPR range y, write access enable register A	$0E200_H$+ y*20_H	32	SPRSE, SV, 32	Application Reset	585
CPU1_ACCENSPRy_WRB (y=0-15)	CPU1 SPR range y, write access enable register B	$0E204_H$+ y*20_H	32	SPRSE, SV, 32	Application Reset	586
CPU1_ACCENSPRy_RDA (y=0-15)	CPU1 SPR range y, read access enable register A	$0E208_H$+ y*20_H	32	SPRSE, SV, 32	Application Reset	587
CPU1_ACCENSPRy_RDB (y=0-15)	CPU1 SPR range y, read access enable register B	$0E20C_H$+ y*20_H	32	SPRSE, SV, 32	Application Reset	588

(table continues...)

Table 216 (continued) Register overview - CPU1 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU1_ACCENSPRy_VM (y=0-15)	CPU1 SPR range y, VM access enable register	$0E210_H$+ y*20_H	32	SPRSE, SV, 32	Application Reset	336
CPU1_ACCENSPRy_PRS (y=0-15)	CPU1 SPR range y, PRS access enable register	$0E214_H$+ y*20_H	32	SPRSE, SV, 32	Application Reset	337
CPU1_ACCENSPRy_RGNLA (y=0-15)	CPU1 SPR range y, region lower address register	$0E218_H$+ y*20_H	32	SPRSE, SV, 32	Application Reset	337
CPU1_ACCENSPRy_RGNUA (y=0-15)	CPU1 SPR range y, region upper address register	$0E21C_H$+ y*20_H	32	SPRSE, SV, 32	Application Reset	338
CPU1_ACCENDLMUy_WRA (y=0-15)	CPU1 DLMU range y, write access enable register A	$0E400_H$+ y*20_H	32	DLMUSE, SV, 32	Application Reset	338
CPU1_ACCENDLMUy_WRB (y=0-15)	CPU1 DLMU range y, write access enable register B	$0E404_H$+ y*20_H	32	DLMUSE, SV, 32	Application Reset	339
CPU1_ACCENDLMUy_RDA (y=0-15)	CPU1 DLMU range y, read access enable register A	$0E408_H$+ y*20_H	32	DLMUSE, SV, 32	Application Reset	339
CPU1_ACCENDLMUy_RDB (y=0-15)	CPU1 DLMU range y, read access enable register B	$0E40C_H$+ y*20_H	32	DLMUSE, SV, 32	Application Reset	340
CPU1_ACCENDLMUy_VM (y=0-15)	CPU1 DLMU range y, VM access enable register	$0E410_H$+ y*20_H	32	DLMUSE, SV, 32	Application Reset	340
CPU1_ACCENDLMUy_PRS (y=0-15)	CPU1 DLMU range y, PRS access enable register	$0E414_H$+ y*20_H	32	DLMUSE, SV, 32	Application Reset	341
CPU1_ACCENDLMUy_RGNLA (y=0-15)	CPU1 DLMU range y, region lower address register	$0E418_H$+ y*20_H	32	DLMUSE, SV, 32	Application Reset	342
CPU1_ACCENDLMUy_RGNUA (y=0-15)	CPU1 DLMU range y, region upper address register	$0E41C_H$+ y*20_H	32	DLMUSE, SV, 32	Application Reset	342
CPU1_ACCENSFR_WRA	CPU1 SFR, write access enable register A	$0E600_H$	32	SFRSE, SV, 32	Application Reset	589
CPU1_ACCENSFR_WRB	CPU1 SFR, write access enable register B	$0E604_H$	32	SFRSE, SV, 32	Application Reset	590
CPU1_ACCENSFR_RDA	CPU1 SFR, read access enable register A	$0E608_H$	32	SFRSE, SV, 32	Application Reset	591

(table continues...)

Table 216 (continued) Register overview - CPU1 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU1_ACCENSFR_RDB	CPU1 SFR, read access enable register B	$0E60C_H$	32	SFRSE, SV, 32	Application Reset	592
CPU1_ACCENSFR_VM	CPU1 SFR, VM access enable register	$0E610_H$	32	SFRSE, SV, 32	Application Reset	345
CPU1_ACCENSFR_PRS	CPU1 SFR, PRS access enable register	$0E614_H$	32	SFRSE, SV, 32	Application Reset	346
CPU1_ACCENSTMVMy_WRA (y=0-7)	CPU1 VMy STM, write access enable register A, virtualization view	$0E700_H$+ y*20_H	32	STMSE, SV, 32	Application Reset	593
CPU1_ACCENSTMVMy_WRB (y=0-7)	CPU1 VMy STM, write access enable register B, virtualization view	$0E704_H$+ y*20_H	32	STMSE, SV, 32	Application Reset	347
CPU1_ACCENSTMVMy_RDA (y=0-7)	CPU1 VMy STM, read access enable register A, virtualization view	$0E708_H$+ y*20_H	32	STMSE, SV, 32	Application Reset	594
CPU1_ACCENSTMVMy_RDB (y=0-7)	CPU1 VMy STM, read access enable register B, virtualization view	$0E70C_H$+ y*20_H	32	STMSE, SV, 32	Application Reset	595
CPU1_ACCENSTMVMy_VM (y=0-7)	CPU1 VMy STM, VM access enable register, virtualization view	$0E710_H$+ y*20_H	32	STMSE, SV, 32	Application Reset	350
CPU1_ACCENSTMVMy_PRS (y=0-7)	CPU1 VMy STM, PRS access enable register, virtualization view	$0E714_H$+ y*20_H	32	STMSE, SV, 32	Application Reset	350
CPU1_OSEL	CPU1 overlay range select register	$0FB00_H$	P0, 32	P0, SV, 32	Kernel Reset	299
CPU1_RABRi (i=0-31)	CPU1 redirected address base register i	$0FB10_H$+ i*12	P0, 32	P0, SV, 32	Kernel Reset	300
CPU1_OTARi (i=0-31)	CPU1 overlay target address register i	$0FB14_H$+ i*12	P0, 32	P0, SV, 32	Kernel Reset	302
CPU1_OMASKi (i=0-31)	CPU1 overlay mask register i	$0FB18_H$+ i*12	P0, 32	P0, SV, 32	Kernel Reset	303
CPU1_HRA_SEGEN	CPU1 SRI error generation register	11030_H	P0, 32	SFRE, P0, SV, 32	Application Reset	255
CPU1_HRA_LCLTEST	CPU1 lockstep test register	11040_H	P0, 32	P0, SV, 32	Cold PowerOn Reset	256
CPU1_HRA_PMA0	CPU1 data access cacheability register	18100_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	232
CPU1_HRA_PMA1	CPU1 code access cacheability register	18104_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	233

(table continues...)

Table 216 (continued) Register overview - CPU1 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU1_HRA_PMA2	CPU1 peripheral space identifier register	18108_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	234
CPU1_HRA_DCON2	CPU1 data control register 2	19000_H	P0, 32	P0, SV, 32	Kernel Reset	558
CPU1_HRA_DCON1	CPU1 data control register 1	19008_H	P0, 32	P0, SV, 32	Kernel Reset	258
CPU1_HRA_SMACON	CPU1 SIST mode access control register	$1900C_H$	P0, 32	SFRSE, P0, SV, 32	Kernel Reset	234
CPU1_HRA_DSTR	CPU1 HR specific data synchronous trap register	19010_H	P0, 32	P0, SV, 32	Kernel Reset	259
CPU1_HRA_DATR	CPU1 HR specific data asynchronous trap register	19018_H	P0, 32	P0, SV, 32	Kernel Reset	261
CPU1_HRA_DEADD	CPU1 HR specific data error address register	$1901C_H$	P0, 32	P0, SV, 32	Kernel Reset	262
CPU1_HRA_DIEAR	CPU1 data integrity error address register	19020_H	P0, 32	P0, SV, 32	Kernel Reset	279
CPU1_HRA_DIETR	CPU1 data integrity error trap register	19024_H	P0, 32	P0, SV, 32	Kernel Reset	280
CPU1_HRA_DCON0	CPU1 data memory control register	19040_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	258
CPU1_HRA_PSTR	CPU1 HR specific program synchronous trap register	19200_H	P0, 32	P0, SV, 32	Kernel Reset	295
CPU1_HRA_PCON1	CPU1 program control 1	19204_H	P0, 32	P0, SV, 32	Kernel Reset	292
CPU1_HRA_PCON2	CPU1 program control 2	19208_H	P0, 32	P0, SV, 32	Kernel Reset	561
CPU1_HRA_PCON0	CPU1 program control 0	$1920C_H$	P0, 32	SFRE, P0, SV, 32	Kernel Reset	294
CPU1_HRA_PIEAR	CPU1 HR specific program integrity error address register	19210_H	P0, 32	P0, SV, 32	Kernel Reset	281
CPU1_HRA_PIETR	CPU1 HR specific program integrity error trap register	19214_H	P0, 32	P0, SV, 32	Kernel Reset	282
CPU1_HRA_COMPAT	CPU1 compatibility control register	19400_H	P0, 32	SFRSE, P0, SV, 32	Kernel Reset	235
CPU1_HRA_FPU_TRAP_CON	CPU1 HR specific trap control register	$1A000_H$	P0, 32	P0, SV, 32	Kernel Reset	263
CPU1_HRA_FPU_TRAP_PC	CPU1 HR specific trapping instruction program counter register	$1A004_H$	P0, 32	P0, SV, 32	Kernel Reset	265
CPU1_HRA_FPU_TRAP_OPC	CPU1 HR specific trapping instruction opcode register	$1A008_H$	P0, 32	P0, SV, 32	Kernel Reset	266

(table continues...)

Table 216 (continued) Register overview - CPU1 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU1_HRA_FPU_TRAP_SRC1_L	CPU1 HR specific trapping instruction operand register	$1A010_H$	P0, 32	P0, SV, 32	Kernel Reset	267
CPU1_HRA_FPU_TRAP_SRC1_U	CPU1 HR specific trapping instruction operand register	$1A014_H$	P0, 32	P0, SV, 32	Kernel Reset	268
CPU1_HRA_FPU_TRAP_SRC2_L	CPU1 HR specific trapping instruction operand register	$1A018_H$	P0, 32	P0, SV, 32	Kernel Reset	268
CPU1_HRA_FPU_TRAP_SRC2_U	CPU1 HR specific trapping instruction operand register	$1A01C_H$	P0, 32	P0, SV, 32	Kernel Reset	269
CPU1_HRA_FPU_TRAP_SRC3_L	CPU1 HR specific trapping instruction operand register	$1A020_H$	P0, 32	P0, SV, 32	Kernel Reset	270
CPU1_HRA_FPU_TRAP_SRC3_U	CPU1 HR specific trapping instruction operand register	$1A024_H$	P0, 32	P0, SV, 32	Kernel Reset	270
CPU1_HRA_FPU_SYNC_TRAP_CON	CPU1 HR specific synchronous trap control register	$1A030_H$	P0, 32	P0, SV, 32	Kernel Reset	271
CPU1_HRA_FPU_SYNC_TRAP_OPC	CPU1 HR specific trapping instruction opcode register	$1A034_H$	P0, 32	P0, SV, 32	Kernel Reset	273
CPU1_HRA_DPRy_L (y=0-23)	CPU1 HR specific data protection range y, lower bound register	$1C000_H$+ y*8	P0, 32	P0, SV, 32	Kernel Reset	283
CPU1_HRA_DPRy_U (y=0-23)	CPU1 HR specific data protection range y, upper bound register	$1C004_H$+ y*8	P0, 32	P0, SV, 32	Kernel Reset	284
CPU1_HRA_CPRy_L (y=0-15)	CPU1 HR specific code protection range y, lower bound register	$1D000_H$+ y*8	P0, 32	P0, SV, 32	Kernel Reset	285
CPU1_HRA_CPRy_U (y=0-15)	CPU1 HR specific code protection range y, upper bound register	$1D004_H$+ y*8	P0, 32	P0, SV, 32	Kernel Reset	286
CPU1_HRA_CPXE_y (y=0-3)	CPU1 HR specific code protection execute enable register set y	$1E000_H$+ y*4	P0, 32	P0, SV, 32	Kernel Reset	287
CPU1_HRA_DPRE_y (y=0-3)	CPU1 HR specific data protection read enable register set y	$1E010_H$+ y*4	P0, 32	P0, SV, 32	Kernel Reset	288

(table continues...)

Table 216 (continued) Register overview - CPU1 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU1_HRA_DPWE_y (y=0-3)	CPU1 HR specific data protection write enable register set y	$1E020_H$+y*4	P0, 32	P0, SV, 32	Kernel Reset	289
CPU1_HRA_CPXE_y (y=4-7)	CPU1 HR specific code protection execute enable register set y	$1E040_H$+(y-4)*4	P0, 32	P0, SV, 32	Kernel Reset	287
CPU1_HRA_DPRE_y (y=4-7)	CPU1 HR specific data protection read enable register set y	$1E050_H$+(y-4)*4	P0, 32	P0, SV, 32	Kernel Reset	288
CPU1_HRA_DPWE_y (y=4-7)	CPU1 HR specific data protection write enable register set y	$1E060_H$+(y-4)*4	P0, 32	P0, SV, 32	Kernel Reset	289
CPU1_HRA_TPS_CON	CPU1 HR specific temporal protection system control register	$1E400_H$	P0, 32	P0, SV, 32	Kernel Reset	296
CPU1_HRA_TPS_TIMERn (n=0-2)	CPU1 HR specific temporal protection system timer register n	$1E404_H$+n*4	P0, 32	P0, SV, 32	Kernel Reset	297
CPU1_HRA_TRnEVT (n=0-7)	CPU1 trigger event n	$1F000_H$+n*8	P0, 32	P0, SV, 32	Debug Reset	209
CPU1_HRA_TRnADR (n=0-7)	CPU1 trigger address n	$1F004_H$+n*8	P0, 32	P0, SV, 32	Debug Reset	210
CPU1_HRA_CCTRL	CPU1 HR specific counter control	$1FC00_H$	P0, 32	P0, SV, 32	Kernel Reset	226
CPU1_HRA_CCNT	CPU1 HR specific CPU clock cycle count	$1FC04_H$	P0, 32	P0, SV, 32	Kernel Reset	227
CPU1_HRA_ICNT	CPU1 HR specific instruction count	$1FC08_H$	P0, 32	P0, SV, 32	Kernel Reset	228
CPU1_HRA_M1CNT	CPU1 HR specific multi-count register 1	$1FC0C_H$	P0, 32	P0, SV, 32	Kernel Reset	229
CPU1_HRA_M2CNT	CPU1 HR specific multi-count register 2	$1FC10_H$	P0, 32	P0, SV, 32	Kernel Reset	230
CPU1_HRA_M3CNT	CPU1 HR specific multi-count register 3	$1FC14_H$	P0, 32	P0, SV, 32	Kernel Reset	231
CPU1_HRA_DBGSR	CPU1 debug status register	$1FD00_H$	P0, 32	P0, SV, 32	Debug Reset	211
CPU1_HRA_EXEVT	CPU1 external event register	$1FD08_H$	P0, 32	P0, SV, 32	Debug Reset	212
CPU1_HRA_CREVT	CPU1 core register access event	$1FD0C_H$	P0, 32	P0, SV, 32	Debug Reset	213

(table continues...)

Table 216 (continued) Register overview - CPU1 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU1_HRA_SWEVT	CPU1 software debug event	$1FD10_H$	P0, 32	P0, SV, 32	Debug Reset	214
CPU1_HRA_DBGACT	CPU1 debug action configuration	$1FD14_H$	P0, 32	P0, SV, 32	Debug Reset	215
CPU1_HRA_TRIG_ACC	CPU1 trigger event accumulator	$1FD30_H$	P0, 32	P0, SV, 32	Debug Reset	217
CPU1_HRA_DMS	CPU1 debug monitor start address	$1FD40_H$	P0, 32	P0, SV, 32	Debug Reset	534
CPU1_HRA_DCX	CPU1 debug context save area pointer	$1FD44_H$	P0, 32	P0, SV, 32	Debug Reset	536
CPU1_HRA_DBGTCR	CPU1 debug trap control	$1FD48_H$	P0, 32	P0, SV, 32	Application Reset	219
CPU1_HRA_DBGCFG	CPU1 debug configuration	$1FD4C_H$	P0, 32	P0, SV, 32	Debug Reset	220
CPU1_HRA_TRCCFG	CPU1 trace configuration	$1FD50_H$	P0, 32	P0, SV, 32	Debug Reset	221
CPU1_HRA_TRCFILT	CPU1 trace filter	$1FD54_H$	P0, 32	P0, SV, 32	Debug Reset	222
CPU1_HRA_TRCLIM	CPU1 trace bandwidth limiter	$1FD58_H$	P0, 32	P0, SV, 32	Debug Reset	224
CPU1_HRA_TS16PTCCTRL	CPU1 MCDS performance counter control register	$1FD60_H$	P0, 32	P0, SV, 32	Debug Reset	225
CPU1_HRA_PCXI	CPU1 previous context information register	$1FE00_H$	P0, 32	P0, SV, 32	Kernel Reset	236
CPU1_HRA_PSW	CPU1 HR specific program status word	$1FE04_H$	P0, 32	P0, SV, 32	Kernel Reset	237
CPU1_HRA_PC	CPU1 program counter	$1FE08_H$	P0, 32	P0, SV, 32	Kernel Reset	538
CPU1_HRA_CORECON	CPU1 HR specific core configuration register	$1FE14_H$	P0, 32	SFRSE, P0, SV, 32	Kernel Reset	250
CPU1_HRA_CPU_ID	CPU1 identification register	$1FE18_H$	P0, 32	P0, SV, 32	Kernel Reset	540
CPU1_HRA_CORE_ID	CPU1 core identification register	$1FE1C_H$	P0, 32	P0, SV, 32	Kernel Reset	542
CPU1_HRA_BIV	CPU1 HR specific base interrupt vector table pointer	$1FE20_H$	P0, 32	SFRE, P0, SV, 32	Kernel Reset	251
CPU1_HRA_BTV	CPU1 HR specific base trap vector table pointer	$1FE24_H$	P0, 32	SFRE, P0, SV, 32	Kernel Reset	554
CPU1_HRA_ISP	CPU1 HR specific interrupt stack pointer	$1FE28_H$	P0, 32	SFRE, P0, SV, 32	Kernel Reset	253
CPU1_HRA_ICR	CPU1 interrupt control register	$1FE2C_H$	P0, 32	P0, SV, 32	Kernel Reset	242

(table continues...)

Table 216 (continued) Register overview - CPU1 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU1_HRA_PPRS	CPU1 previous PRS	$1FE34_H$	P0, 32	P0, SV, 32	Kernel Reset	244
CPU1_HRA_FCX	CPU1 HR specific free CSA list head pointer	$1FE38_H$	P0, 32	P0, SV, 32	Kernel Reset	254
CPU1_HRA_LCX	CPU1 HR specific free CSA list limit pointer	$1FE3C_H$	P0, 32	P0, SV, 32	Kernel Reset	255
CPU1_HRA_SWID	CPU1 software identification	$1FE40_H$	P0, 32	P0, SV, 32	Kernel Reset	544
CPU1_HRA_CUS_ID	CPU1 customer ID register	$1FE50_H$	P0, 32	P0, SV, 32	Kernel Reset	546
CPU1_HRA_BOOTCON	CPU1 core boot configuration register	$1FE60_H$	P0, 32	P0, SV, 32	Kernel Reset	548
CPU1_HRA_LCLCON	CPU1 lockstep control register	$1FE64_H$	P0, 32	STP, SFRSE, P0, SV, 32	Cold PowerOn Reset	550
CPU1_HRA_CCON	CPU1 clock control register	$1FE68_H$	P0, 32	P0, SV, 32	Kernel Reset	248
CPU1_HRA_TCCON	CPU1 hardware configuration	$1FE6C_H$	P0, 32	P0, SV, 32	Kernel Reset	552
CPU1_Dy (y=0-15)	CPU1 data general purpose register y	$1FF00_H$+ y*4	P0, 32	P0, SV, 32	Kernel Reset	274
CPU1_Ay (y=0-15)	CPU1 address general purpose register y	$1FF80_H$+ y*4	P0, 32	P0, SV, 32	Kernel Reset	274
CPU1_HRB_SEGEN	CPU1 SRI error generation register	21030_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	255
CPU1_HRB_LCLTEST	CPU1 lockstep test register	21040_H	P0, 32	P0, SV, 32	Cold PowerOn Reset	256
CPU1_HRB_PMA0	CPU1 data access cacheability register	28100_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	232
CPU1_HRB_PMA1	CPU1 code access cacheability register	28104_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	233
CPU1_HRB_PMA2	CPU1 peripheral space identifier register	28108_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	234
CPU1_HRB_DCON2	CPU1 data control register 2	29000_H	P0, 32	P0, SV, 32	Kernel Reset	558
CPU1_HRB_DCON1	CPU1 data control register 1	29008_H	P0, 32	P0, SV, 32	Kernel Reset	258
CPU1_HRB_SMACON	CPU1 SIST mode access control register	$2900C_H$	P0, 32	SFRSE, P0, SV, 32	Kernel Reset	234
CPU1_HRB_DSTR	CPU1 HR specific data synchronous trap register	29010_H	P0, 32	P0, SV, 32	Kernel Reset	259

(table continues...)

Table 216 (continued) Register overview - CPU1 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU1_HRB_DATR	CPU1 HR specific data asynchronous trap register	29018_H	P0, 32	P0, SV, 32	Kernel Reset	261
CPU1_HRB_DEADD	CPU1 HR specific data error address register	$2901C_H$	P0, 32	P0, SV, 32	Kernel Reset	262
CPU1_HRB_DIEAR	CPU1 data integrity error address register	29020_H	P0, 32	P0, SV, 32	Kernel Reset	279
CPU1_HRB_DIETR	CPU1 data integrity error trap register	29024_H	P0, 32	P0, SV, 32	Kernel Reset	280
CPU1_HRB_DCON0	CPU1 data memory control register	29040_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	258
CPU1_HRB_PSTR	CPU1 HR specific program synchronous trap register	29200_H	P0, 32	P0, SV, 32	Kernel Reset	295
CPU1_HRB_PCON1	CPU1 program control 1	29204_H	P0, 32	P0, SV, 32	Kernel Reset	292
CPU1_HRB_PCON2	CPU1 program control 2	29208_H	P0, 32	P0, SV, 32	Kernel Reset	561
CPU1_HRB_PCON0	CPU1 program control 0	$2920C_H$	P0, 32	SFRE, P0, SV, 32	Kernel Reset	294
CPU1_HRB_PIEAR	CPU1 HR specific program integrity error address register	29210_H	P0, 32	P0, SV, 32	Kernel Reset	281
CPU1_HRB_PIETR	CPU1 HR specific program integrity error trap register	29214_H	P0, 32	P0, SV, 32	Kernel Reset	282
CPU1_HRB_COMPAT	CPU1 compatibility control register	29400_H	P0, 32	SFRSE, P0, SV, 32	Kernel Reset	235
CPU1_HRB_FPU_TRAP_CON	CPU1 HR specific trap control register	$2A000_H$	P0, 32	P0, SV, 32	Kernel Reset	263
CPU1_HRB_FPU_TRAP_PC	CPU1 HR specific trapping instruction program counter register	$2A004_H$	P0, 32	P0, SV, 32	Kernel Reset	265
CPU1_HRB_FPU_TRAP_OPC	CPU1 HR specific trapping instruction opcode register	$2A008_H$	P0, 32	P0, SV, 32	Kernel Reset	266
CPU1_HRB_FPU_TRAP_SRC1_L	CPU1 HR specific trapping instruction operand register	$2A010_H$	P0, 32	P0, SV, 32	Kernel Reset	267
CPU1_HRB_FPU_TRAP_SRC1_U	CPU1 HR specific trapping instruction operand register	$2A014_H$	P0, 32	P0, SV, 32	Kernel Reset	268
CPU1_HRB_FPU_TRAP_SRC2_L	CPU1 HR specific trapping instruction operand register	$2A018_H$	P0, 32	P0, SV, 32	Kernel Reset	268

(table continues...)

Table 216 (continued) Register overview - CPU1 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU1_HRB_FPU_TRAP_SRC2_U	CPU1 HR specific trapping instruction operand register	$2A01C_H$	P0, 32	P0, SV, 32	Kernel Reset	269
CPU1_HRB_FPU_TRAP_SRC3_L	CPU1 HR specific trapping instruction operand register	$2A020_H$	P0, 32	P0, SV, 32	Kernel Reset	270
CPU1_HRB_FPU_TRAP_SRC3_U	CPU1 HR specific trapping instruction operand register	$2A024_H$	P0, 32	P0, SV, 32	Kernel Reset	270
CPU1_HRB_FPU_SYNC_TRAP_CON	CPU1 HR specific synchronous trap control register	$2A030_H$	P0, 32	P0, SV, 32	Kernel Reset	271
CPU1_HRB_FPU_SYNC_TRAP_OPC	CPU1 HR specific trapping instruction opcode register	$2A034_H$	P0, 32	P0, SV, 32	Kernel Reset	273
CPU1_HRB_DPRy_L (y=0-23)	CPU1 HR specific data protection range y, lower bound register	$2C000_H$+y*8	P0, 32	P0, SV, 32	Kernel Reset	283
CPU1_HRB_DPRy_U (y=0-23)	CPU1 HR specific data protection range y, upper bound register	$2C004_H$+y*8	P0, 32	P0, SV, 32	Kernel Reset	284
CPU1_HRB_CPRy_L (y=0-15)	CPU1 HR specific code protection range y, lower bound register	$2D000_H$+y*8	P0, 32	P0, SV, 32	Kernel Reset	285
CPU1_HRB_CPRy_U (y=0-15)	CPU1 HR specific code protection range y, upper bound register	$2D004_H$+y*8	P0, 32	P0, SV, 32	Kernel Reset	286
CPU1_HRB_CPXE_y (y=0-3)	CPU1 HR specific code protection execute enable register set y	$2E000_H$+y*4	P0, 32	P0, SV, 32	Kernel Reset	287
CPU1_HRB_DPRE_y (y=0-3)	CPU1 HR specific data protection read enable register set y	$2E010_H$+y*4	P0, 32	P0, SV, 32	Kernel Reset	288
CPU1_HRB_DPWE_y (y=0-3)	CPU1 HR specific data protection write enable register set y	$2E020_H$+y*4	P0, 32	P0, SV, 32	Kernel Reset	289
CPU1_HRB_CPXE_y (y=4-7)	CPU1 HR specific code protection execute enable register set y	$2E040_H$+(y-4)*4	P0, 32	P0, SV, 32	Kernel Reset	287
CPU1_HRB_DPRE_y (y=4-7)	CPU1 HR specific data protection read enable register set y	$2E050_H$+(y-4)*4	P0, 32	P0, SV, 32	Kernel Reset	288

(table continues...)

Table 216 (continued) Register overview - CPU1 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU1_HRB_DPWE_y (y=4-7)	CPU1 HR specific data protection write enable register set y	$2E060_H$+(y-4)*4	P0, 32	P0, SV, 32	Kernel Reset	289
CPU1_HRB_TPS_CON	CPU1 HR specific temporal protection system control register	$2E400_H$	P0, 32	P0, SV, 32	Kernel Reset	296
CPU1_HRB_TPS_TIMERn (n=0-2)	CPU1 HR specific temporal protection system timer register n	$2E404_H$+n*4	P0, 32	P0, SV, 32	Kernel Reset	297
CPU1_HRB_TRnEVT (n=0-7)	CPU1 trigger event n	$2F000_H$+n*8	P0, 32	P0, SV, 32	Debug Reset	209
CPU1_HRB_TRnADR (n=0-7)	CPU1 trigger address n	$2F004_H$+n*8	P0, 32	P0, SV, 32	Debug Reset	210
CPU1_HRB_CCTRL	CPU1 HR specific counter control	$2FC00_H$	P0, 32	P0, SV, 32	Kernel Reset	226
CPU1_HRB_CCNT	CPU1 HR specific CPU clock cycle count	$2FC04_H$	P0, 32	P0, SV, 32	Kernel Reset	227
CPU1_HRB_ICNT	CPU1 HR specific instruction count	$2FC08_H$	P0, 32	P0, SV, 32	Kernel Reset	228
CPU1_HRB_M1CNT	CPU1 HR specific multi-count register 1	$2FC0C_H$	P0, 32	P0, SV, 32	Kernel Reset	229
CPU1_HRB_M2CNT	CPU1 HR specific multi-count register 2	$2FC10_H$	P0, 32	P0, SV, 32	Kernel Reset	230
CPU1_HRB_M3CNT	CPU1 HR specific multi-count register 3	$2FC14_H$	P0, 32	P0, SV, 32	Kernel Reset	231
CPU1_HRB_DBGSR	CPU1 debug status register	$2FD00_H$	P0, 32	P0, SV, 32	Debug Reset	211
CPU1_HRB_EXEVT	CPU1 external event register	$2FD08_H$	P0, 32	P0, SV, 32	Debug Reset	212
CPU1_HRB_CREVT	CPU1 core register access event	$2FD0C_H$	P0, 32	P0, SV, 32	Debug Reset	213
CPU1_HRB_SWEVT	CPU1 software debug event	$2FD10_H$	P0, 32	P0, SV, 32	Debug Reset	214
CPU1_HRB_DBGACT	CPU1 debug action configuration	$2FD14_H$	P0, 32	P0, SV, 32	Debug Reset	215
CPU1_HRB_TRIG_ACC	CPU1 trigger event accumulator	$2FD30_H$	P0, 32	P0, SV, 32	Debug Reset	217
CPU1_HRB_DMS	CPU1 debug monitor start address	$2FD40_H$	P0, 32	P0, SV, 32	Debug Reset	534

(table continues...)

Table 216 (continued) Register overview - CPU1 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU1_HRB_DCX	CPU1 debug context save area pointer	$2FD44_H$	P0, 32	P0, SV, 32	Debug Reset	536
CPU1_HRB_DBGTCR	CPU1 debug trap control	$2FD48_H$	P0, 32	P0, SV, 32	Application Reset	219
CPU1_HRB_DBGCFG	CPU1 debug configuration	$2FD4C_H$	P0, 32	P0, SV, 32	Debug Reset	220
CPU1_HRB_TRCCFG	CPU1 trace configuration	$2FD50_H$	P0, 32	P0, SV, 32	Debug Reset	221
CPU1_HRB_TRCFILT	CPU1 trace filter	$2FD54_H$	P0, 32	P0, SV, 32	Debug Reset	222
CPU1_HRB_TRCLIM	CPU1 trace bandwidth limiter	$2FD58_H$	P0, 32	P0, SV, 32	Debug Reset	224
CPU1_HRB_TS16PTCCTRL	CPU1 MCDS performance counter control register	$2FD60_H$	P0, 32	P0, SV, 32	Debug Reset	225
CPU1_HRB_PCXI	CPU1 previous context information register	$2FE00_H$	P0, 32	P0, SV, 32	Kernel Reset	236
CPU1_HRB_PSW	CPU1 HR specific program status word	$2FE04_H$	P0, 32	P0, SV, 32	Kernel Reset	237
CPU1_HRB_PC	CPU1 program counter	$2FE08_H$	P0, 32	P0, SV, 32	Kernel Reset	538
CPU1_HRB_CORECON	CPU1 HR specific core configuration register	$2FE14_H$	P0, 32	SFRSE, P0, SV, 32	Kernel Reset	250
CPU1_HRB_CPU_ID	CPU1 identification register	$2FE18_H$	P0, 32	P0, SV, 32	Kernel Reset	540
CPU1_HRB_CORE_ID	CPU1 core identification register	$2FE1C_H$	P0, 32	P0, SV, 32	Kernel Reset	542
CPU1_HRB_BIV	CPU1 HR specific base interrupt vector table pointer	$2FE20_H$	P0, 32	SFRE, P0, SV, 32	Kernel Reset	251
CPU1_HRB_BTV	CPU1 HR specific base trap vector table pointer	$2FE24_H$	P0, 32	SFRE, P0, SV, 32	Kernel Reset	555
CPU1_HRB_ISP	CPU1 HR specific interrupt stack pointer	$2FE28_H$	P0, 32	SFRE, P0, SV, 32	Kernel Reset	253
CPU1_HRB_ICR	CPU1 interrupt control register	$2FE2C_H$	P0, 32	P0, SV, 32	Kernel Reset	242
CPU1_HRB_PPRS	CPU1 previous PRS	$2FE34_H$	P0, 32	P0, SV, 32	Kernel Reset	244
CPU1_HRB_FCX	CPU1 HR specific free CSA list head pointer	$2FE38_H$	P0, 32	P0, SV, 32	Kernel Reset	254
CPU1_HRB_LCX	CPU1 HR specific free CSA list limit pointer	$2FE3C_H$	P0, 32	P0, SV, 32	Kernel Reset	255
CPU1_HRB_SWID	CPU1 software identification	$2FE40_H$	P0, 32	P0, SV, 32	Kernel Reset	544

(table continues...)

Table 216 (continued) Register overview - CPU1 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU1_HRB_CUS_ID	CPU1 customer ID register	$2FE50_H$	P0, 32	P0, SV, 32	Kernel Reset	546
CPU1_HRB_BOOTCON	CPU1 core boot configuration register	$2FE60_H$	P0, 32	P0, SV, 32	Kernel Reset	548
CPU1_HRB_LCLCON	CPU1 lockstep control register	$2FE64_H$	P0, 32	STP, SFRSE, P0, SV, 32	Cold PowerOn Reset	550
CPU1_HRB_CCON	CPU1 clock control register	$2FE68_H$	P0, 32	P0, SV, 32	Kernel Reset	248
CPU1_HRB_TCCON	CPU1 hardware configuration	$2FE6C_H$	P0, 32	P0, SV, 32	Kernel Reset	552
CPU1_HRB_Ay (y=0-1)	CPU1 HR specific address general purpose register y	$2FF80_H$+ y*4	P0, 32	P0, SV, 32	Kernel Reset	274
CPU1_HRB_Ay (y=8-9)	CPU1 HR specific address general purpose register y	$2FFA0_H$+ (y-8)*4	P0, 32	P0, SV, 32	Kernel Reset	274
CPU1_HRHV_SEGEN	CPU1 SRI error generation register	31030_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	255
CPU1_HRHV_LCLTEST	CPU1 lockstep test register	31040_H	P0, 32	P0, SV, 32	Cold PowerOn Reset	256
CPU1_HRHV_PMA0	CPU1 data access cacheability register	38100_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	232
CPU1_HRHV_PMA1	CPU1 code access cacheability register	38104_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	233
CPU1_HRHV_PMA2	CPU1 peripheral space identifier register	38108_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	234
CPU1_HRHV_DCON2	CPU1 data control register 2	39000_H	P0, 32	P0, SV, 32	Kernel Reset	558
CPU1_HRHV_DCON1	CPU1 data control register 1	39008_H	P0, 32	P0, SV, 32	Kernel Reset	258
CPU1_HRHV_SMACON	CPU1 SIST mode access control register	$3900C_H$	P0, 32	SFRSE, P0, SV, 32	Kernel Reset	234
CPU1_HRHV_DSTR	CPU1 HR specific data synchronous trap register	39010_H	P0, 32	P0, SV, 32	Kernel Reset	259
CPU1_HRHV_DATR	CPU1 HR specific data asynchronous trap register	39018_H	P0, 32	P0, SV, 32	Kernel Reset	261
CPU1_HRHV_DEADD	CPU1 HR specific data error address register	$3901C_H$	P0, 32	P0, SV, 32	Kernel Reset	262
CPU1_HRHV_DIEAR	CPU1 data integrity error address register	39020_H	P0, 32	P0, SV, 32	Kernel Reset	279
CPU1_HRHV_DIETR	CPU1 data integrity error trap register	39024_H	P0, 32	P0, SV, 32	Kernel Reset	280

(table continues...)

Table 216 (continued) Register overview - CPU1 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU1_HRHV_DCON0	CPU1 data memory control register	39040_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	258
CPU1_HRHV_PSTR	CPU1 HR specific program synchronous trap register	39200_H	P0, 32	P0, SV, 32	Kernel Reset	295
CPU1_HRHV_PCON1	CPU1 program control 1	39204_H	P0, 32	P0, SV, 32	Kernel Reset	292
CPU1_HRHV_PCON2	CPU1 program control 2	39208_H	P0, 32	P0, SV, 32	Kernel Reset	561
CPU1_HRHV_PCON0	CPU1 program control 0	$3920C_H$	P0, 32	SFRE, P0, SV, 32	Kernel Reset	294
CPU1_HRHV_PIEAR	CPU1 HR specific program integrity error address register	39210_H	P0, 32	P0, SV, 32	Kernel Reset	281
CPU1_HRHV_PIETR	CPU1 HR specific program integrity error trap register	39214_H	P0, 32	P0, SV, 32	Kernel Reset	282
CPU1_HRHV_COMPAT	CPU1 compatibility control register	39400_H	P0, 32	SFRSE, P0, SV, 32	Kernel Reset	235
CPU1_HRHV_FPU_TRAP_CON	CPU1 HR specific trap control register	$3A000_H$	P0, 32	P0, SV, 32	Kernel Reset	263
CPU1_HRHV_FPU_TRAP_PC	CPU1 HR specific trapping instruction program counter register	$3A004_H$	P0, 32	P0, SV, 32	Kernel Reset	265
CPU1_HRHV_FPU_TRAP_OPC	CPU1 HR specific trapping instruction opcode register	$3A008_H$	P0, 32	P0, SV, 32	Kernel Reset	266
CPU1_HRHV_FPU_TRAP_SRC1_L	CPU1 HR specific trapping instruction operand register	$3A010_H$	P0, 32	P0, SV, 32	Kernel Reset	267
CPU1_HRHV_FPU_TRAP_SRC1_U	CPU1 HR specific trapping instruction operand register	$3A014_H$	P0, 32	P0, SV, 32	Kernel Reset	268
CPU1_HRHV_FPU_TRAP_SRC2_L	CPU1 HR specific trapping instruction operand register	$3A018_H$	P0, 32	P0, SV, 32	Kernel Reset	268
CPU1_HRHV_FPU_TRAP_SRC2_U	CPU1 HR specific trapping instruction operand register	$3A01C_H$	P0, 32	P0, SV, 32	Kernel Reset	269
CPU1_HRHV_FPU_TRAP_SRC3_L	CPU1 HR specific trapping instruction operand register	$3A020_H$	P0, 32	P0, SV, 32	Kernel Reset	270
CPU1_HRHV_FPU_TRAP_SRC3_U	CPU1 HR specific trapping instruction operand register	$3A024_H$	P0, 32	P0, SV, 32	Kernel Reset	270

(table continues...)

Table 216 (continued) Register overview - CPU1 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU1_HRHV_FPU_SYNC_TRAP_CON	CPU1 HR specific synchronous trap control register	$3A030_H$	P0, 32	P0, SV, 32	Kernel Reset	271
CPU1_HRHV_FPU_SYNC_TRAP_OPC	CPU1 HR specific trapping instruction opcode register	$3A034_H$	P0, 32	P0, SV, 32	Kernel Reset	273
CPU1_HRHV_VCON0	CPU1 virtualization control register 0	$3B000_H$	P0, 32	P0, SV, 32	Kernel Reset	275
CPU1_HRHV_VCON1	CPU1 virtualization control register 1	$3B004_H$	P0, 32	P0, SV, 32	Kernel Reset	276
CPU1_HRHV_VCON2	CPU1 virtualization control register 2	$3B008_H$	P0, 32	P0, SV, 32	Kernel Reset	276
CPU1_HRHV_BHV	CPU1 base hypervisor vector table pointer	$3B010_H$	P0, 32	P0, SV, 32, SFRE	Kernel Reset	559
CPU1_HRHV_VMn_ICR (n=0-7)	CPU1 Virtual machine n interrupt state	$3B100_H$+ n*4	P0, 32	P0, SV, 32	Kernel Reset	278
CPU1_HRHV_VMn_PETHRESH (n=0-7)	CPU1 Virtual machine n pre-emption threshold	$3B200_H$+ n*4	P0, 32	P0, SV, 32	Kernel Reset	279
CPU1_HRHV_DPRy_L (y=0-23)	CPU1 HR specific data protection range y, lower bound register	$3C000_H$+ y*8	P0, 32	P0, SV, 32	Kernel Reset	283
CPU1_HRHV_DPRy_U (y=0-23)	CPU1 HR specific data protection range y, upper bound register	$3C004_H$+ y*8	P0, 32	P0, SV, 32	Kernel Reset	284
CPU1_HRHV_CPRy_L (y=0-15)	CPU1 HR specific code protection range y, lower bound register	$3D000_H$+ y*8	P0, 32	P0, SV, 32	Kernel Reset	285
CPU1_HRHV_CPRy_U (y=0-15)	CPU1 HR specific code protection range y, upper bound register	$3D004_H$+ y*8	P0, 32	P0, SV, 32	Kernel Reset	286
CPU1_HRHV_CPXE_y (y=0-3)	CPU1 HR specific code protection execute enable register set y	$3E000_H$+ y*4	P0, 32	P0, SV, 32	Kernel Reset	287
CPU1_HRHV_DPRE_y (y=0-3)	CPU1 HR specific data protection read enable register set y	$3E010_H$+ y*4	P0, 32	P0, SV, 32	Kernel Reset	288
CPU1_HRHV_DPWE_y (y=0-3)	CPU1 HR specific data protection write enable register set y	$3E020_H$+ y*4	P0, 32	P0, SV, 32	Kernel Reset	289

(table continues...)

Table 216 (continued) Register overview - CPU1 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU1_HRHV_CPXE_y (y=4-7)	CPU1 HR specific code protection execute enable register set y	$3E040_H$+(y-4)*4	P0, 32	P0, SV, 32	Kernel Reset	287
CPU1_HRHV_DPRE_y (y=4-7)	CPU1 HR specific data protection read enable register set y	$3E050_H$+(y-4)*4	P0, 32	P0, SV, 32	Kernel Reset	288
CPU1_HRHV_DPWE_y (y=4-7)	CPU1 HR specific data protection write enable register set y	$3E060_H$+(y-4)*4	P0, 32	P0, SV, 32	Kernel Reset	289
CPU1_HRHV_TPS_CON	CPU1 HR specific temporal protection system control register	$3E400_H$	P0, 32	P0, SV, 32	Kernel Reset	296
CPU1_HRHV_TPS_TIMERn (n=0-2)	CPU1 HR specific temporal protection system timer register n	$3E404_H$+n*4	P0, 32	P0, SV, 32	Kernel Reset	297
CPU1_HRHV_TRnEVT (n=0-7)	CPU1 trigger event n	$3F000_H$+n*8	P0, 32	P0, SV, 32	Debug Reset	209
CPU1_HRHV_TRnADR (n=0-7)	CPU1 trigger address n	$3F004_H$+n*8	P0, 32	P0, SV, 32	Debug Reset	210
CPU1_HRHV_CCTRL	CPU1 HR specific counter control	$3FC00_H$	P0, 32	P0, SV, 32	Kernel Reset	226
CPU1_HRHV_CCNT	CPU1 HR specific CPU clock cycle count	$3FC04_H$	P0, 32	P0, SV, 32	Kernel Reset	227
CPU1_HRHV_ICNT	CPU1 HR specific instruction count	$3FC08_H$	P0, 32	P0, SV, 32	Kernel Reset	228
CPU1_HRHV_M1CNT	CPU1 HR specific multi-count register 1	$3FC0C_H$	P0, 32	P0, SV, 32	Kernel Reset	229
CPU1_HRHV_M2CNT	CPU1 HR specific multi-count register 2	$3FC10_H$	P0, 32	P0, SV, 32	Kernel Reset	230
CPU1_HRHV_M3CNT	CPU1 HR specific multi-count register 3	$3FC14_H$	P0, 32	P0, SV, 32	Kernel Reset	231
CPU1_HRHV_DBGSR	CPU1 debug status register	$3FD00_H$	P0, 32	P0, SV, 32	Debug Reset	211
CPU1_HRHV_EXEVT	CPU1 external event register	$3FD08_H$	P0, 32	P0, SV, 32	Debug Reset	212
CPU1_HRHV_CREVT	CPU1 core register access event	$3FD0C_H$	P0, 32	P0, SV, 32	Debug Reset	213
CPU1_HRHV_SWEVT	CPU1 software debug event	$3FD10_H$	P0, 32	P0, SV, 32	Debug Reset	214

(table continues...)

Table 216 (continued) Register overview - CPU1 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU1_HRHV_DBGACT	CPU1 debug action configuration	$3FD14_H$	P0, 32	P0, SV, 32	Debug Reset	215
CPU1_HRHV_TRIG_ACC	CPU1 trigger event accumulator	$3FD30_H$	P0, 32	P0, SV, 32	Debug Reset	217
CPU1_HRHV_DMS	CPU1 debug monitor start address	$3FD40_H$	P0, 32	P0, SV, 32	Debug Reset	534
CPU1_HRHV_DCX	CPU1 debug context save area pointer	$3FD44_H$	P0, 32	P0, SV, 32	Debug Reset	536
CPU1_HRHV_DBGTCR	CPU1 debug trap control	$3FD48_H$	P0, 32	P0, SV, 32	Application Reset	219
CPU1_HRHV_DBGCFG	CPU1 debug configuration	$3FD4C_H$	P0, 32	P0, SV, 32	Debug Reset	220
CPU1_HRHV_TRCCFG	CPU1 trace configuration	$3FD50_H$	P0, 32	P0, SV, 32	Debug Reset	221
CPU1_HRHV_TRCFILT	CPU1 trace filter	$3FD54_H$	P0, 32	P0, SV, 32	Debug Reset	222
CPU1_HRHV_TRCLIM	CPU1 trace bandwidth limiter	$3FD58_H$	P0, 32	P0, SV, 32	Debug Reset	224
CPU1_HRHV_TS16PTCCTRL	CPU1 MCDS performance counter control register	$3FD60_H$	P0, 32	P0, SV, 32	Debug Reset	225
CPU1_HRHV_PCXI	CPU1 previous context information register	$3FE00_H$	P0, 32	P0, SV, 32	Kernel Reset	236
CPU1_HRHV_PSW	CPU1 HR specific program status word	$3FE04_H$	P0, 32	P0, SV, 32	Kernel Reset	237
CPU1_HRHV_PC	CPU1 program counter	$3FE08_H$	P0, 32	P0, SV, 32	Kernel Reset	538
CPU1_HRHV_CORECON	CPU1 HR specific core configuration register	$3FE14_H$	P0, 32	SFRSE, P0, SV, 32	Kernel Reset	250
CPU1_HRHV_CPU_ID	CPU1 identification register	$3FE18_H$	P0, 32	P0, SV, 32	Kernel Reset	540
CPU1_HRHV_CORE_ID	CPU1 core identification register	$3FE1C_H$	P0, 32	P0, SV, 32	Kernel Reset	542
CPU1_HRHV_BIV	CPU1 HR specific base interrupt vector table pointer	$3FE20_H$	P0, 32	SFRE, P0, SV, 32	Kernel Reset	251
CPU1_HRHV_BTV	CPU1 HR specific base trap vector table pointer	$3FE24_H$	P0, 32	SFRE, P0, SV, 32	Kernel Reset	555
CPU1_HRHV_ISP	CPU1 HR specific interrupt stack pointer	$3FE28_H$	P0, 32	SFRE, P0, SV, 32	Kernel Reset	253
CPU1_HRHV_ICR	CPU1 interrupt control register	$3FE2C_H$	P0, 32	P0, SV, 32	Kernel Reset	242
CPU1_HRHV_PPRS	CPU1 previous PRS	$3FE34_H$	P0, 32	P0, SV, 32	Kernel Reset	244

(table continues...)

Table 216 (continued) Register overview - CPU1 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU1_HRHV_FCX	CPU1 HR specific free CSA list head pointer	$3FE38_H$	P0, 32	P0, SV, 32	Kernel Reset	254
CPU1_HRHV_LCX	CPU1 HR specific free CSA list limit pointer	$3FE3C_H$	P0, 32	P0, SV, 32	Kernel Reset	255
CPU1_HRHV_SWID	CPU1 software identification	$3FE40_H$	P0, 32	P0, SV, 32	Kernel Reset	544
CPU1_HRHV_CUS_ID	CPU1 customer ID register	$3FE50_H$	P0, 32	P0, SV, 32	Kernel Reset	546
CPU1_HRHV_BOOTCON	CPU1 core boot configuration register	$3FE60_H$	P0, 32	P0, SV, 32	Kernel Reset	548
CPU1_HRHV_LCLCON	CPU1 lockstep control register	$3FE64_H$	P0, 32	STP, SFRSE, P0, SV, 32	Cold PowerOn Reset	550
CPU1_HRHV_CCON	CPU1 clock control register	$3FE68_H$	P0, 32	P0, SV, 32	Kernel Reset	248
CPU1_HRHV_TCCON	CPU1 hardware configuration	$3FE6C_H$	P0, 32	P0, SV, 32	Kernel Reset	552
CPU1_HRHV_Ay (y=0-1)	CPU1 HR specific address general purpose register y	$3FF80_H$+ y*4	P0, 32	P0, SV, 32	Kernel Reset	274
CPU1_HRHV_Ay (y=8-9)	CPU1 HR specific address general purpose register y	$3FFA0_H$+ (y-8)*4	P0, 32	P0, SV, 32	Kernel Reset	274

4.10.4.1.7 Register overview - CPU2 domain SRI (ascending offset address)

Table 217 Register overview - CPU2 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU2_STM_CLC	CPU2 system timer clock control register	00000_H	P0, 32	P0, SV, SFRE, 32	System Reset	351
CPU2_STMHV_CLC	CPU2 system timer clock control register, hypervisor view	00000_H	P0, 32	P0, SV, SFRE, 32	System Reset	351
CPU2_STM_OCS	CPU2 OCDS control and status register	00004_H	P0, 32	P0, SV, 32, DEN	Debug Reset	352
CPU2_STMHV_OCS	CPU2 OCDS control and status register, hypervisor view	00004_H	P0, 32	P0, SV, 32, DEN	Debug Reset	352
CPU2_STM_ABS	CPU2 system timer absolute timer	00020_H	(P0, 32)| (P0, 64)	P0, SV, 64, SFRSE	System Reset	353
CPU2_STMHV_ABS	CPU2 system timer absolute timer, hypervisor view	00020_H	(P0, 32)| (P0, 64)	P0, SV, 64, SFRSE	System Reset	353

(table continues...)

Table 217 (continued) Register overview - CPU2 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU2_STMHV_VMn_CMPc (c=0-1;n=0-7)	CPU2 VMn system timer compare register c, hypervisor view	00100_H+ n*20_H+c *4	P1, 32	P1, SV, 32	System Reset	354
CPU2_STMHV_VMn_CMCON (n=0-7)	CPU2 VMn system timer compare match control register, hypervisor view	00108_H+ n*20_H	P1, 32	P1, SV, 32	System Reset	355
CPU2_STMHV_VMn_ICR (n=0-7)	CPU2 VMn system timer interrupt control register, hypervisor view	$0010C_H$+ n*20_H	P1, 32	P1, SV, 32	System Reset	356
CPU2_STMHV_VMn_ISCR (n=0-7)	CPU2 VMn system timer interrupt set/clear register, hypervisor view	00110_H+ n*20_H	P1, 32	P1, SV, 32	System Reset	357
CPU2_STMHV_VMn_ISR (n=0-7)	CPU2 VMn system timer interrupt status register, hypervisor view	00114_H+ n*20_H	P1, 32	P1, SV, 32	System Reset	358
CPU2_STMHV_VMn_RELTIM (n=0-7)	CPU2 VMn system timer relative timer, hypervisor view	00118_H+ n*20_H	(P1, 32)| (P1, 64)	P1, SV, 64	System Reset	359
CPU2_STM_CMPc (c=0-1)	CPU2 system timer compare register c	00120_H+ c*4	P1, 32	P1, SV, 32	System Reset	354
CPU2_STM_CMCON	CPU2 system timer compare match control register	00128_H	P1, 32	P1, SV, 32	System Reset	355
CPU2_STM_ICR	CPU2 system timer interrupt control register	$0012C_H$	P1, 32	P1, SV, 32	System Reset	356
CPU2_STM_ISCR	CPU2 system timer interrupt set/clear register	00130_H	P1, 32	P1, SV, 32	System Reset	357
CPU2_STM_ISR	CPU2 system timer interrupt status register	00134_H	P1, 32	P1, SV, 32	System Reset	358
CPU2_STM_RELTIM	CPU2 system timer relative timer	00138_H	(P1, 32)| (P1, 64)	P1, SV, 64	System Reset	359
CPU2_KRST0	CPU2 reset register 0	$0D000_H$	P0, 32	RSTE, P0, SV, 32	Kernel Reset	297
CPU2_KRST1	CPU2 reset register 1	$0D004_H$	P0, 32	RSTE, P0, SV, 32	Kernel Reset	298
CPU2_KRSTCLR	CPU2 reset clear register	$0D008_H$	P0, 32	RSTE, P0, SV, 32	Kernel Reset	299
CPU2_PROTSPRSE	CPU2 PROTSPR register safe endinit	$0E008_H$	32	SV, 32, PROT	Application Reset	562
CPU2_ACCENSPRCFG_WRA	CPU2 shadow SPR, write access enable register A	$0E020_H$	32	SPRSE, SV, 32	Application Reset	565

(table continues...)

Table 217 (continued) Register overview - CPU2 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU2_ACCENSPRCFG_WRB	CPU2 shadow SPR, write access enable register B	$0E024_H$	32	SPRSE, SV, 32	Application Reset	566
CPU2_ACCENSPRCFG_RDA	CPU2 shadow SPR, read access enable register A	$0E028_H$	32	SPRSE, SV, 32	Application Reset	567
CPU2_ACCENSPRCFG_RDB	CPU2 shadow SPR, read access enable register B	$0E02C_H$	32	SPRSE, SV, 32	Application Reset	568
CPU2_ACCENSPRCFG_VM	CPU2 shadow SPR, VM access enable register	$0E030_H$	32	SPRSE, SV, 32	Application Reset	310
CPU2_ACCENSPRCFG_PRS	CPU2 shadow SPR, PRS access enable register	$0E034_H$	32	SPRSE, SV, 32	Application Reset	310
CPU2_ACCENSPRCFG_RGNLA	CPU2 shadow SPR, region lower address register	$0E038_H$	32	SPRSE, SV, 32	Application Reset	311
CPU2_ACCENSPRCFG_RGNUA	CPU2 shadow SPR, region upper address register	$0E03C_H$	32	SPRSE, SV, 32	Application Reset	312
CPU2_PROTDLMUSE	CPU2 PROTDLMU register safe endinit	$0E048_H$	32	SV, 32, PROT	Application Reset	312
CPU2_ACCENDLMUCFG_WRA	CPU2 shadow DLMU, write access enable register A	$0E060_H$	32	DLMUSE, SV, 32	Application Reset	314
CPU2_ACCENDLMUCFG_WRB	CPU2 shadow DLMU, write access enable register B	$0E064_H$	32	DLMUSE, SV, 32	Application Reset	315
CPU2_ACCENDLMUCFG_RDA	CPU2 shadow DLMU, read access enable register A	$0E068_H$	32	DLMUSE, SV, 32	Application Reset	315
CPU2_ACCENDLMUCFG_RDB	CPU2 shadow DLMU, read access enable register B	$0E06C_H$	32	DLMUSE, SV, 32	Application Reset	316
CPU2_ACCENDLMUCFG_VM	CPU2 shadow DLMU, VM access enable register	$0E070_H$	32	DLMUSE, SV, 32	Application Reset	316
CPU2_ACCENDLMUCFG_PRS	CPU2 shadow DLMU, PRS access enable register	$0E074_H$	32	DLMUSE, SV, 32	Application Reset	317
CPU2_ACCENDLMUCFG_RGNLA	CPU2 shadow DLMU, region lower address register	$0E078_H$	32	DLMUSE, SV, 32	Application Reset	317
CPU2_ACCENDLMUCFG_RGNUA	CPU2 shadow DLMU, region upper address register	$0E07C_H$	32	DLMUSE, SV, 32	Application Reset	318
CPU2_PROTSFRE	CPU2 PROTSFR register endinit	$0E080_H$	32	SV, 32, PROT	Application Reset	569
CPU2_PROTSFRSE	CPU2 PROTSFR register safe endinit	$0E088_H$	32	SV, 32, PROT	Application Reset	571
CPU2_ACCENSFRCFG_WRA	CPU2 shadow SFR, write access enable register A	$0E0A0_H$	32	SFRSE, SV, 32	Application Reset	573
CPU2_ACCENSFRCFG_WRB	CPU2 shadow SFR, write access enable register B	$0E0A4_H$	32	SFRSE, SV, 32	Application Reset	574
CPU2_ACCENSFRCFG_RDA	CPU2 shadow SFR, read access enable register A	$0E0A8_H$	32	SFRSE, SV, 32	Application Reset	575

(table continues...)

Table 217 (continued) Register overview - CPU2 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU2_ACCENSFRCFG_RDB	CPU2 shadow SFR, read access enable register B	$0E0AC_H$	32	SFRSE, SV, 32	Application Reset	576
CPU2_ACCENSFRCFG_VM	CPU2 shadow SFR, VM access enable register	$0E0B0_H$	32	SFRSE, SV, 32	Application Reset	325
CPU2_ACCENSFRCFG_PRS	CPU2 shadow SFR, PRS access enable register	$0E0B4_H$	32	SFRSE, SV, 32	Application Reset	325
CPU2_PROTRSTE	CPU2 PROTRST register endinit	$0E0C0_H$	32	SV, 32, PROT	Application Reset	577
CPU2_PROTSTMSE	CPU2 PROTSTM Register Safe Endinit	$0E0D8_H$	32	SV, 32, PROT	Application Reset	579
CPU2_ACCENSTMCFG_WRA	CPU2 shadow STM, write access enable register A	$0E0E0_H$	32	STMSE, SV, 32	Application Reset	582
CPU2_ACCENSTMCFG_WRB	CPU2 shadow STM, write access enable register B	$0E0E4_H$	32	STMSE, SV, 32	Application Reset	330
CPU2_ACCENSTMCFG_RDA	CPU2 shadow STM, read access enable register A	$0E0E8_H$	32	STMSE, SV, 32	Application Reset	583
CPU2_ACCENSTMCFG_RDB	CPU2 shadow STM, read access enable register B	$0E0EC_H$	32	STMSE, SV, 32	Application Reset	584
CPU2_ACCENSTMCFG_VM	CPU2 shadow STM, VM access enable register	$0E0F0_H$	32	STMSE, SV, 32	Application Reset	332
CPU2_ACCENSTMCFG_PRS	CPU2 shadow STM, PRS access enable register	$0E0F4_H$	32	STMSE, SV, 32	Application Reset	333
CPU2_ACCENSPRy_WRA (y=0-15)	CPU2 SPR range y, write access enable register A	$0E200_H$+ y*20_H	32	SPRSE, SV, 32	Application Reset	585
CPU2_ACCENSPRy_WRB (y=0-15)	CPU2 SPR range y, write access enable register B	$0E204_H$+ y*20_H	32	SPRSE, SV, 32	Application Reset	586
CPU2_ACCENSPRy_RDA (y=0-15)	CPU2 SPR range y, read access enable register A	$0E208_H$+ y*20_H	32	SPRSE, SV, 32	Application Reset	587
CPU2_ACCENSPRy_RDB (y=0-15)	CPU2 SPR range y, read access enable register B	$0E20C_H$+ y*20_H	32	SPRSE, SV, 32	Application Reset	588
CPU2_ACCENSPRy_VM (y=0-15)	CPU2 SPR range y, VM access enable register	$0E210_H$+ y*20_H	32	SPRSE, SV, 32	Application Reset	336
CPU2_ACCENSPRy_PRS (y=0-15)	CPU2 SPR range y, PRS access enable register	$0E214_H$+ y*20_H	32	SPRSE, SV, 32	Application Reset	337

(table continues...)

Table 217 (continued) Register overview - CPU2 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU2_ACCENSPRy_RGNLA (y=0-15)	CPU2 SPR range y, region lower address register	$0E218_H$+ y*20_H	32	SPRSE, SV, 32	Application Reset	337
CPU2_ACCENSPRy_RGNUA (y=0-15)	CPU2 SPR range y, region upper address register	$0E21C_H$+ y*20_H	32	SPRSE, SV, 32	Application Reset	338
CPU2_ACCENDLMUy_WRA (y=0-15)	CPU2 DLMU range y, write access enable register A	$0E400_H$+ y*20_H	32	DLMUSE, SV, 32	Application Reset	338
CPU2_ACCENDLMUy_WRB (y=0-15)	CPU2 DLMU range y, write access enable register B	$0E404_H$+ y*20_H	32	DLMUSE, SV, 32	Application Reset	339
CPU2_ACCENDLMUy_RDA (y=0-15)	CPU2 DLMU range y, read access enable register A	$0E408_H$+ y*20_H	32	DLMUSE, SV, 32	Application Reset	339
CPU2_ACCENDLMUy_RDB (y=0-15)	CPU2 DLMU range y, read access enable register B	$0E40C_H$+ y*20_H	32	DLMUSE, SV, 32	Application Reset	340
CPU2_ACCENDLMUy_VM (y=0-15)	CPU2 DLMU range y, VM access enable register	$0E410_H$+ y*20_H	32	DLMUSE, SV, 32	Application Reset	340
CPU2_ACCENDLMUy_PRS (y=0-15)	CPU2 DLMU range y, PRS access enable register	$0E414_H$+ y*20_H	32	DLMUSE, SV, 32	Application Reset	341
CPU2_ACCENDLMUy_RGNLA (y=0-15)	CPU2 DLMU range y, region lower address register	$0E418_H$+ y*20_H	32	DLMUSE, SV, 32	Application Reset	342
CPU2_ACCENDLMUy_RGNUA (y=0-15)	CPU2 DLMU range y, region upper address register	$0E41C_H$+ y*20_H	32	DLMUSE, SV, 32	Application Reset	342
CPU2_ACCENSFR_WRA	CPU2 SFR, write access enable register A	$0E600_H$	32	SFRSE, SV, 32	Application Reset	589
CPU2_ACCENSFR_WRB	CPU2 SFR, write access enable register B	$0E604_H$	32	SFRSE, SV, 32	Application Reset	590
CPU2_ACCENSFR_RDA	CPU2 SFR, read access enable register A	$0E608_H$	32	SFRSE, SV, 32	Application Reset	591
CPU2_ACCENSFR_RDB	CPU2 SFR, read access enable register B	$0E60C_H$	32	SFRSE, SV, 32	Application Reset	592
CPU2_ACCENSFR_VM	CPU2 SFR, VM access enable register	$0E610_H$	32	SFRSE, SV, 32	Application Reset	345
CPU2_ACCENSFR_PRS	CPU2 SFR, PRS access enable register	$0E614_H$	32	SFRSE, SV, 32	Application Reset	346

(table continues...)

Table 217 (continued) Register overview - CPU2 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU2_ACCENSTMVMy_WRA (y=0-7)	CPU2 VMy STM, write access enable register A, virtualization view	$0E700_H$+ y*20_H	32	STMSE, SV, 32	Application Reset	593
CPU2_ACCENSTMVMy_WRB (y=0-7)	CPU2 VMy STM, write access enable register B, virtualization view	$0E704_H$+ y*20_H	32	STMSE, SV, 32	Application Reset	347
CPU2_ACCENSTMVMy_RDA (y=0-7)	CPU2 VMy STM, read access enable register A, virtualization view	$0E708_H$+ y*20_H	32	STMSE, SV, 32	Application Reset	594
CPU2_ACCENSTMVMy_RDB (y=0-7)	CPU2 VMy STM, read access enable register B, virtualization view	$0E70C_H$+ y*20_H	32	STMSE, SV, 32	Application Reset	595
CPU2_ACCENSTMVMy_VM (y=0-7)	CPU2 VMy STM, VM access enable register, virtualization view	$0E710_H$+ y*20_H	32	STMSE, SV, 32	Application Reset	350
CPU2_ACCENSTMVMy_PRS (y=0-7)	CPU2 VMy STM, PRS access enable register, virtualization view	$0E714_H$+ y*20_H	32	STMSE, SV, 32	Application Reset	350
CPU2_OSEL	CPU2 overlay range select register	$0FB00_H$	P0, 32	P0, SV, 32	Kernel Reset	299
CPU2_RABRi (i=0-31)	CPU2 redirected address base register i	$0FB10_H$+ i*12	P0, 32	P0, SV, 32	Kernel Reset	300
CPU2_OTARi (i=0-31)	CPU2 overlay target address register i	$0FB14_H$+ i*12	P0, 32	P0, SV, 32	Kernel Reset	302
CPU2_OMASKi (i=0-31)	CPU2 overlay mask register i	$0FB18_H$+ i*12	P0, 32	P0, SV, 32	Kernel Reset	303
CPU2_HRA_SEGEN	CPU2 SRI error generation register	11030_H	P0, 32	SFRE, P0, SV, 32	Application Reset	255
CPU2_HRA_LCLTEST	CPU2 lockstep test register	11040_H	P0, 32	P0, SV, 32	Cold PowerOn Reset	256
CPU2_HRA_PMA0	CPU2 data access cacheability register	18100_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	232
CPU2_HRA_PMA1	CPU2 code access cacheability register	18104_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	233
CPU2_HRA_PMA2	CPU2 peripheral space identifier register	18108_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	234
CPU2_HRA_DCON2	CPU2 data control register 2	19000_H	P0, 32	P0, SV, 32	Kernel Reset	558
CPU2_HRA_DCON1	CPU2 data control register 1	19008_H	P0, 32	P0, SV, 32	Kernel Reset	258

(table continues...)

Table 217 (continued) Register overview - CPU2 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU2_HRA_SMACON	CPU2 SIST mode access control register	$1900C_H$	P0, 32	SFRSE, P0, SV, 32	Kernel Reset	234
CPU2_HRA_DSTR	CPU2 HR specific data synchronous trap register	19010_H	P0, 32	P0, SV, 32	Kernel Reset	259
CPU2_HRA_DATR	CPU2 HR specific data asynchronous trap register	19018_H	P0, 32	P0, SV, 32	Kernel Reset	261
CPU2_HRA_DEADD	CPU2 HR specific data error address register	$1901C_H$	P0, 32	P0, SV, 32	Kernel Reset	262
CPU2_HRA_DIEAR	CPU2 data integrity error address register	19020_H	P0, 32	P0, SV, 32	Kernel Reset	279
CPU2_HRA_DIETR	CPU2 data integrity error trap register	19024_H	P0, 32	P0, SV, 32	Kernel Reset	280
CPU2_HRA_DCON0	CPU2 data memory control register	19040_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	258
CPU2_HRA_PSTR	CPU2 HR specific program synchronous trap register	19200_H	P0, 32	P0, SV, 32	Kernel Reset	295
CPU2_HRA_PCON1	CPU2 program control 1	19204_H	P0, 32	P0, SV, 32	Kernel Reset	292
CPU2_HRA_PCON2	CPU2 program control 2	19208_H	P0, 32	P0, SV, 32	Kernel Reset	561
CPU2_HRA_PCON0	CPU2 program control 0	$1920C_H$	P0, 32	SFRE, P0, SV, 32	Kernel Reset	294
CPU2_HRA_PIEAR	CPU2 HR specific program integrity error address register	19210_H	P0, 32	P0, SV, 32	Kernel Reset	281
CPU2_HRA_PIETR	CPU2 HR specific program integrity error trap register	19214_H	P0, 32	P0, SV, 32	Kernel Reset	282
CPU2_HRA_COMPAT	CPU2 compatibility control register	19400_H	P0, 32	SFRSE, P0, SV, 32	Kernel Reset	235
CPU2_HRA_FPU_TRAP_CON	CPU2 HR specific trap control register	$1A000_H$	P0, 32	P0, SV, 32	Kernel Reset	263
CPU2_HRA_FPU_TRAP_PC	CPU2 HR specific trapping instruction program counter register	$1A004_H$	P0, 32	P0, SV, 32	Kernel Reset	265
CPU2_HRA_FPU_TRAP_OPC	CPU2 HR specific trapping instruction opcode register	$1A008_H$	P0, 32	P0, SV, 32	Kernel Reset	266
CPU2_HRA_FPU_TRAP_SRC1_L	CPU2 HR specific trapping instruction operand register	$1A010_H$	P0, 32	P0, SV, 32	Kernel Reset	267
CPU2_HRA_FPU_TRAP_SRC1_U	CPU2 HR specific trapping instruction operand register	$1A014_H$	P0, 32	P0, SV, 32	Kernel Reset	268

(table continues...)

Table 217 (continued) Register overview - CPU2 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU2_HRA_FPU_TRAP_SRC2_L	CPU2 HR specific trapping instruction operand register	$1A018_H$	P0, 32	P0, SV, 32	Kernel Reset	268
CPU2_HRA_FPU_TRAP_SRC2_U	CPU2 HR specific trapping instruction operand register	$1A01C_H$	P0, 32	P0, SV, 32	Kernel Reset	269
CPU2_HRA_FPU_TRAP_SRC3_L	CPU2 HR specific trapping instruction operand register	$1A020_H$	P0, 32	P0, SV, 32	Kernel Reset	270
CPU2_HRA_FPU_TRAP_SRC3_U	CPU2 HR specific trapping instruction operand register	$1A024_H$	P0, 32	P0, SV, 32	Kernel Reset	270
CPU2_HRA_FPU_SYNC_TRAP_CON	CPU2 HR specific synchronous trap control register	$1A030_H$	P0, 32	P0, SV, 32	Kernel Reset	271
CPU2_HRA_FPU_SYNC_TRAP_OPC	CPU2 HR specific trapping instruction opcode register	$1A034_H$	P0, 32	P0, SV, 32	Kernel Reset	273
CPU2_HRA_DPRy_L (y=0-23)	CPU2 HR specific data protection range y, lower bound register	$1C000_H$+y*8	P0, 32	P0, SV, 32	Kernel Reset	283
CPU2_HRA_DPRy_U (y=0-23)	CPU2 HR specific data protection range y, upper bound register	$1C004_H$+y*8	P0, 32	P0, SV, 32	Kernel Reset	284
CPU2_HRA_CPRy_L (y=0-15)	CPU2 HR specific code protection range y, lower bound register	$1D000_H$+y*8	P0, 32	P0, SV, 32	Kernel Reset	285
CPU2_HRA_CPRy_U (y=0-15)	CPU2 HR specific code protection range y, upper bound register	$1D004_H$+y*8	P0, 32	P0, SV, 32	Kernel Reset	286
CPU2_HRA_CPXE_y (y=0-3)	CPU2 HR specific code protection execute enable register set y	$1E000_H$+y*4	P0, 32	P0, SV, 32	Kernel Reset	287
CPU2_HRA_DPRE_y (y=0-3)	CPU2 HR specific data protection read enable register set y	$1E010_H$+y*4	P0, 32	P0, SV, 32	Kernel Reset	288
CPU2_HRA_DPWE_y (y=0-3)	CPU2 HR specific data protection write enable register set y	$1E020_H$+y*4	P0, 32	P0, SV, 32	Kernel Reset	289
CPU2_HRA_CPXE_y (y=4-7)	CPU2 HR specific code protection execute enable register set y	$1E040_H$+(y-4)*4	P0, 32	P0, SV, 32	Kernel Reset	287

(table continues...)

Table 217 (continued) Register overview - CPU2 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU2_HRA_DPRE_y (y=4-7)	CPU2 HR specific data protection read enable register set y	$1E050_H$+(y-4)*4	P0, 32	P0, SV, 32	Kernel Reset	288
CPU2_HRA_DPWE_y (y=4-7)	CPU2 HR specific data protection write enable register set y	$1E060_H$+(y-4)*4	P0, 32	P0, SV, 32	Kernel Reset	289
CPU2_HRA_TPS_CON	CPU2 HR specific temporal protection system control register	$1E400_H$	P0, 32	P0, SV, 32	Kernel Reset	296
CPU2_HRA_TPS_TIMERn (n=0-2)	CPU2 HR specific temporal protection system timer register n	$1E404_H$+n*4	P0, 32	P0, SV, 32	Kernel Reset	297
CPU2_HRA_TRnEVT (n=0-7)	CPU2 trigger event n	$1F000_H$+n*8	P0, 32	P0, SV, 32	Debug Reset	209
CPU2_HRA_TRnADR (n=0-7)	CPU2 trigger address n	$1F004_H$+n*8	P0, 32	P0, SV, 32	Debug Reset	210
CPU2_HRA_CCTRL	CPU2 HR specific counter control	$1FC00_H$	P0, 32	P0, SV, 32	Kernel Reset	226
CPU2_HRA_CCNT	CPU2 HR specific CPU clock cycle count	$1FC04_H$	P0, 32	P0, SV, 32	Kernel Reset	227
CPU2_HRA_ICNT	CPU2 HR specific instruction count	$1FC08_H$	P0, 32	P0, SV, 32	Kernel Reset	228
CPU2_HRA_M1CNT	CPU2 HR specific multi-count register 1	$1FC0C_H$	P0, 32	P0, SV, 32	Kernel Reset	229
CPU2_HRA_M2CNT	CPU2 HR specific multi-count register 2	$1FC10_H$	P0, 32	P0, SV, 32	Kernel Reset	230
CPU2_HRA_M3CNT	CPU2 HR specific multi-count register 3	$1FC14_H$	P0, 32	P0, SV, 32	Kernel Reset	231
CPU2_HRA_DBGSR	CPU2 debug status register	$1FD00_H$	P0, 32	P0, SV, 32	Debug Reset	211
CPU2_HRA_EXEVT	CPU2 external event register	$1FD08_H$	P0, 32	P0, SV, 32	Debug Reset	212
CPU2_HRA_CREVT	CPU2 core register access event	$1FD0C_H$	P0, 32	P0, SV, 32	Debug Reset	213
CPU2_HRA_SWEVT	CPU2 software debug event	$1FD10_H$	P0, 32	P0, SV, 32	Debug Reset	214
CPU2_HRA_DBGACT	CPU2 debug action configuration	$1FD14_H$	P0, 32	P0, SV, 32	Debug Reset	215
CPU2_HRA_TRIG_ACC	CPU2 trigger event accumulator	$1FD30_H$	P0, 32	P0, SV, 32	Debug Reset	217

(table continues...)

Table 217 (continued) Register overview - CPU2 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU2_HRA_DMS	CPU2 debug monitor start address	$1FD40_H$	P0, 32	P0, SV, 32	Debug Reset	534
CPU2_HRA_DCX	CPU2 debug context save area pointer	$1FD44_H$	P0, 32	P0, SV, 32	Debug Reset	536
CPU2_HRA_DBGTCR	CPU2 debug trap control	$1FD48_H$	P0, 32	P0, SV, 32	Application Reset	219
CPU2_HRA_DBGCFG	CPU2 debug configuration	$1FD4C_H$	P0, 32	P0, SV, 32	Debug Reset	220
CPU2_HRA_TRCCFG	CPU2 trace configuration	$1FD50_H$	P0, 32	P0, SV, 32	Debug Reset	221
CPU2_HRA_TRCFILT	CPU2 trace filter	$1FD54_H$	P0, 32	P0, SV, 32	Debug Reset	222
CPU2_HRA_TRCLIM	CPU2 trace bandwidth limiter	$1FD58_H$	P0, 32	P0, SV, 32	Debug Reset	224
CPU2_HRA_TS16PTCCTRL	CPU2 MCDS performance counter control register	$1FD60_H$	P0, 32	P0, SV, 32	Debug Reset	225
CPU2_HRA_PCXI	CPU2 previous context information register	$1FE00_H$	P0, 32	P0, SV, 32	Kernel Reset	236
CPU2_HRA_PSW	CPU2 HR specific program status word	$1FE04_H$	P0, 32	P0, SV, 32	Kernel Reset	237
CPU2_HRA_PC	CPU2 program counter	$1FE08_H$	P0, 32	P0, SV, 32	Kernel Reset	538
CPU2_HRA_CORECON	CPU2 HR specific core configuration register	$1FE14_H$	P0, 32	SFRSE, P0, SV, 32	Kernel Reset	250
CPU2_HRA_CPU_ID	CPU2 identification register	$1FE18_H$	P0, 32	P0, SV, 32	Kernel Reset	540
CPU2_HRA_CORE_ID	CPU2 core identification register	$1FE1C_H$	P0, 32	P0, SV, 32	Kernel Reset	542
CPU2_HRA_BIV	CPU2 HR specific base interrupt vector table pointer	$1FE20_H$	P0, 32	SFRE, P0, SV, 32	Kernel Reset	251
CPU2_HRA_BTV	CPU2 HR specific base trap vector table pointer	$1FE24_H$	P0, 32	SFRE, P0, SV, 32	Kernel Reset	554
CPU2_HRA_ISP	CPU2 HR specific interrupt stack pointer	$1FE28_H$	P0, 32	SFRE, P0, SV, 32	Kernel Reset	253
CPU2_HRA_ICR	CPU2 interrupt control register	$1FE2C_H$	P0, 32	P0, SV, 32	Kernel Reset	242
CPU2_HRA_PPRS	CPU2 previous PRS	$1FE34_H$	P0, 32	P0, SV, 32	Kernel Reset	244
CPU2_HRA_FCX	CPU2 HR specific free CSA list head pointer	$1FE38_H$	P0, 32	P0, SV, 32	Kernel Reset	254
CPU2_HRA_LCX	CPU2 HR specific free CSA list limit pointer	$1FE3C_H$	P0, 32	P0, SV, 32	Kernel Reset	255

(table continues...)

Table 217 (continued) Register overview - CPU2 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU2_HRA_SWID	CPU2 software identification	$1FE40_H$	P0, 32	P0, SV, 32	Kernel Reset	544
CPU2_HRA_CUS_ID	CPU2 customer ID register	$1FE50_H$	P0, 32	P0, SV, 32	Kernel Reset	546
CPU2_HRA_BOOTCON	CPU2 core boot configuration register	$1FE60_H$	P0, 32	P0, SV, 32	Kernel Reset	548
CPU2_HRA_LCLCON	CPU2 lockstep control register	$1FE64_H$	P0, 32	STP, SFRSE, P0, SV, 32	Cold PowerOn Reset	550
CPU2_HRA_CCON	CPU2 clock control register	$1FE68_H$	P0, 32	P0, SV, 32	Kernel Reset	248
CPU2_HRA_TCCON	CPU2 hardware configuration	$1FE6C_H$	P0, 32	P0, SV, 32	Kernel Reset	552
CPU2_Dy (y=0-15)	CPU2 data general purpose register y	$1FF00_H$+ y*4	P0, 32	P0, SV, 32	Kernel Reset	274
CPU2_Ay (y=0-15)	CPU2 address general purpose register y	$1FF80_H$+ y*4	P0, 32	P0, SV, 32	Kernel Reset	274
CPU2_HRB_SEGEN	CPU2 SRI error generation register	21030_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	255
CPU2_HRB_LCLTEST	CPU2 lockstep test register	21040_H	P0, 32	P0, SV, 32	Cold PowerOn Reset	256
CPU2_HRB_PMA0	CPU2 data access cacheability register	28100_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	232
CPU2_HRB_PMA1	CPU2 code access cacheability register	28104_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	233
CPU2_HRB_PMA2	CPU2 peripheral space identifier register	28108_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	234
CPU2_HRB_DCON2	CPU2 data control register 2	29000_H	P0, 32	P0, SV, 32	Kernel Reset	558
CPU2_HRB_DCON1	CPU2 data control register 1	29008_H	P0, 32	P0, SV, 32	Kernel Reset	258
CPU2_HRB_SMACON	CPU2 SIST mode access control register	$2900C_H$	P0, 32	SFRSE, P0, SV, 32	Kernel Reset	234
CPU2_HRB_DSTR	CPU2 HR specific data synchronous trap register	29010_H	P0, 32	P0, SV, 32	Kernel Reset	259
CPU2_HRB_DATR	CPU2 HR specific data asynchronous trap register	29018_H	P0, 32	P0, SV, 32	Kernel Reset	261
CPU2_HRB_DEADD	CPU2 HR specific data error address register	$2901C_H$	P0, 32	P0, SV, 32	Kernel Reset	262
CPU2_HRB_DIEAR	CPU2 data integrity error address register	29020_H	P0, 32	P0, SV, 32	Kernel Reset	279

(table continues...)

Table 217 (continued) Register overview - CPU2 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU2_HRB_DIETR	CPU2 data integrity error trap register	29024_H	P0, 32	P0, SV, 32	Kernel Reset	280
CPU2_HRB_DCON0	CPU2 data memory control register	29040_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	258
CPU2_HRB_PSTR	CPU2 HR specific program synchronous trap register	29200_H	P0, 32	P0, SV, 32	Kernel Reset	295
CPU2_HRB_PCON1	CPU2 program control 1	29204_H	P0, 32	P0, SV, 32	Kernel Reset	292
CPU2_HRB_PCON2	CPU2 program control 2	29208_H	P0, 32	P0, SV, 32	Kernel Reset	561
CPU2_HRB_PCON0	CPU2 program control 0	$2920C_H$	P0, 32	SFRE, P0, SV, 32	Kernel Reset	294
CPU2_HRB_PIEAR	CPU2 HR specific program integrity error address register	29210_H	P0, 32	P0, SV, 32	Kernel Reset	281
CPU2_HRB_PIETR	CPU2 HR specific program integrity error trap register	29214_H	P0, 32	P0, SV, 32	Kernel Reset	282
CPU2_HRB_COMPAT	CPU2 compatibility control register	29400_H	P0, 32	SFRSE, P0, SV, 32	Kernel Reset	235
CPU2_HRB_FPU_TRAP_CON	CPU2 HR specific trap control register	$2A000_H$	P0, 32	P0, SV, 32	Kernel Reset	263
CPU2_HRB_FPU_TRAP_PC	CPU2 HR specific trapping instruction program counter register	$2A004_H$	P0, 32	P0, SV, 32	Kernel Reset	265
CPU2_HRB_FPU_TRAP_OPC	CPU2 HR specific trapping instruction opcode register	$2A008_H$	P0, 32	P0, SV, 32	Kernel Reset	266
CPU2_HRB_FPU_TRAP_SRC1_L	CPU2 HR specific trapping instruction operand register	$2A010_H$	P0, 32	P0, SV, 32	Kernel Reset	267
CPU2_HRB_FPU_TRAP_SRC1_U	CPU2 HR specific trapping instruction operand register	$2A014_H$	P0, 32	P0, SV, 32	Kernel Reset	268
CPU2_HRB_FPU_TRAP_SRC2_L	CPU2 HR specific trapping instruction operand register	$2A018_H$	P0, 32	P0, SV, 32	Kernel Reset	268
CPU2_HRB_FPU_TRAP_SRC2_U	CPU2 HR specific trapping instruction operand register	$2A01C_H$	P0, 32	P0, SV, 32	Kernel Reset	269
CPU2_HRB_FPU_TRAP_SRC3_L	CPU2 HR specific trapping instruction operand register	$2A020_H$	P0, 32	P0, SV, 32	Kernel Reset	270

(table continues...)

Table 217 (continued) Register overview - CPU2 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU2_HRB_FPU_TRAP_SRC3_U	CPU2 HR specific trapping instruction operand register	$2A024_H$	P0, 32	P0, SV, 32	Kernel Reset	270
CPU2_HRB_FPU_SYNC_TRAP_CON	CPU2 HR specific synchronous trap control register	$2A030_H$	P0, 32	P0, SV, 32	Kernel Reset	271
CPU2_HRB_FPU_SYNC_TRAP_OPC	CPU2 HR specific trapping instruction opcode register	$2A034_H$	P0, 32	P0, SV, 32	Kernel Reset	273
CPU2_HRB_DPRy_L (y=0-23)	CPU2 HR specific data protection range y, lower bound register	$2C000_H$+y*8	P0, 32	P0, SV, 32	Kernel Reset	283
CPU2_HRB_DPRy_U (y=0-23)	CPU2 HR specific data protection range y, upper bound register	$2C004_H$+y*8	P0, 32	P0, SV, 32	Kernel Reset	284
CPU2_HRB_CPRy_L (y=0-15)	CPU2 HR specific code protection range y, lower bound register	$2D000_H$+y*8	P0, 32	P0, SV, 32	Kernel Reset	285
CPU2_HRB_CPRy_U (y=0-15)	CPU2 HR specific code protection range y, upper bound register	$2D004_H$+y*8	P0, 32	P0, SV, 32	Kernel Reset	286
CPU2_HRB_CPXE_y (y=0-3)	CPU2 HR specific code protection execute enable register set y	$2E000_H$+y*4	P0, 32	P0, SV, 32	Kernel Reset	287
CPU2_HRB_DPRE_y (y=0-3)	CPU2 HR specific data protection read enable register set y	$2E010_H$+y*4	P0, 32	P0, SV, 32	Kernel Reset	288
CPU2_HRB_DPWE_y (y=0-3)	CPU2 HR specific data protection write enable register set y	$2E020_H$+y*4	P0, 32	P0, SV, 32	Kernel Reset	289
CPU2_HRB_CPXE_y (y=4-7)	CPU2 HR specific code protection execute enable register set y	$2E040_H$+(y-4)*4	P0, 32	P0, SV, 32	Kernel Reset	287
CPU2_HRB_DPRE_y (y=4-7)	CPU2 HR specific data protection read enable register set y	$2E050_H$+(y-4)*4	P0, 32	P0, SV, 32	Kernel Reset	288
CPU2_HRB_DPWE_y (y=4-7)	CPU2 HR specific data protection write enable register set y	$2E060_H$+(y-4)*4	P0, 32	P0, SV, 32	Kernel Reset	289
CPU2_HRB_TPS_CON	CPU2 HR specific temporal protection system control register	$2E400_H$	P0, 32	P0, SV, 32	Kernel Reset	296

(table continues...)

Table 217 (continued) Register overview - CPU2 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU2_HRB_TPS_TIMERn (n=0-2)	CPU2 HR specific temporal protection system timer register n	$2E404_H$+ n*4	P0, 32	P0, SV, 32	Kernel Reset	297
CPU2_HRB_TRnEVT (n=0-7)	CPU2 trigger event n	$2F000_H$+ n*8	P0, 32	P0, SV, 32	Debug Reset	209
CPU2_HRB_TRnADR (n=0-7)	CPU2 trigger address n	$2F004_H$+ n*8	P0, 32	P0, SV, 32	Debug Reset	210
CPU2_HRB_CCTRL	CPU2 HR specific counter control	$2FC00_H$	P0, 32	P0, SV, 32	Kernel Reset	226
CPU2_HRB_CCNT	CPU2 HR specific CPU clock cycle count	$2FC04_H$	P0, 32	P0, SV, 32	Kernel Reset	227
CPU2_HRB_ICNT	CPU2 HR specific instruction count	$2FC08_H$	P0, 32	P0, SV, 32	Kernel Reset	228
CPU2_HRB_M1CNT	CPU2 HR specific multi-count register 1	$2FC0C_H$	P0, 32	P0, SV, 32	Kernel Reset	229
CPU2_HRB_M2CNT	CPU2 HR specific multi-count register 2	$2FC10_H$	P0, 32	P0, SV, 32	Kernel Reset	230
CPU2_HRB_M3CNT	CPU2 HR specific multi-count register 3	$2FC14_H$	P0, 32	P0, SV, 32	Kernel Reset	231
CPU2_HRB_DBGSR	CPU2 debug status register	$2FD00_H$	P0, 32	P0, SV, 32	Debug Reset	211
CPU2_HRB_EXEVT	CPU2 external event register	$2FD08_H$	P0, 32	P0, SV, 32	Debug Reset	212
CPU2_HRB_CREVT	CPU2 core register access event	$2FD0C_H$	P0, 32	P0, SV, 32	Debug Reset	213
CPU2_HRB_SWEVT	CPU2 software debug event	$2FD10_H$	P0, 32	P0, SV, 32	Debug Reset	214
CPU2_HRB_DBGACT	CPU2 debug action configuration	$2FD14_H$	P0, 32	P0, SV, 32	Debug Reset	215
CPU2_HRB_TRIG_ACC	CPU2 trigger event accumulator	$2FD30_H$	P0, 32	P0, SV, 32	Debug Reset	217
CPU2_HRB_DMS	CPU2 debug monitor start address	$2FD40_H$	P0, 32	P0, SV, 32	Debug Reset	534
CPU2_HRB_DCX	CPU2 debug context save area pointer	$2FD44_H$	P0, 32	P0, SV, 32	Debug Reset	536
CPU2_HRB_DBGTCR	CPU2 debug trap control	$2FD48_H$	P0, 32	P0, SV, 32	Application Reset	219
CPU2_HRB_DBGCFG	CPU2 debug configuration	$2FD4C_H$	P0, 32	P0, SV, 32	Debug Reset	220

(table continues...)

Table 217 (continued) Register overview - CPU2 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU2_HRB_TRCCFG	CPU2 trace configuration	$2FD50_H$	P0, 32	P0, SV, 32	Debug Reset	221
CPU2_HRB_TRCFILT	CPU2 trace filter	$2FD54_H$	P0, 32	P0, SV, 32	Debug Reset	222
CPU2_HRB_TRCLIM	CPU2 trace bandwidth limiter	$2FD58_H$	P0, 32	P0, SV, 32	Debug Reset	224
CPU2_HRB_TS16PTCCTRL	CPU2 MCDS performance counter control register	$2FD60_H$	P0, 32	P0, SV, 32	Debug Reset	225
CPU2_HRB_PCXI	CPU2 previous context information register	$2FE00_H$	P0, 32	P0, SV, 32	Kernel Reset	236
CPU2_HRB_PSW	CPU2 HR specific program status word	$2FE04_H$	P0, 32	P0, SV, 32	Kernel Reset	237
CPU2_HRB_PC	CPU2 program counter	$2FE08_H$	P0, 32	P0, SV, 32	Kernel Reset	538
CPU2_HRB_CORECON	CPU2 HR specific core configuration register	$2FE14_H$	P0, 32	SFRSE, P0, SV, 32	Kernel Reset	250
CPU2_HRB_CPU_ID	CPU2 identification register	$2FE18_H$	P0, 32	P0, SV, 32	Kernel Reset	540
CPU2_HRB_CORE_ID	CPU2 core identification register	$2FE1C_H$	P0, 32	P0, SV, 32	Kernel Reset	542
CPU2_HRB_BIV	CPU2 HR specific base interrupt vector table pointer	$2FE20_H$	P0, 32	SFRE, P0, SV, 32	Kernel Reset	251
CPU2_HRB_BTV	CPU2 HR specific base trap vector table pointer	$2FE24_H$	P0, 32	SFRE, P0, SV, 32	Kernel Reset	555
CPU2_HRB_ISP	CPU2 HR specific interrupt stack pointer	$2FE28_H$	P0, 32	SFRE, P0, SV, 32	Kernel Reset	253
CPU2_HRB_ICR	CPU2 interrupt control register	$2FE2C_H$	P0, 32	P0, SV, 32	Kernel Reset	242
CPU2_HRB_PPRS	CPU2 previous PRS	$2FE34_H$	P0, 32	P0, SV, 32	Kernel Reset	244
CPU2_HRB_FCX	CPU2 HR specific free CSA list head pointer	$2FE38_H$	P0, 32	P0, SV, 32	Kernel Reset	254
CPU2_HRB_LCX	CPU2 HR specific free CSA list limit pointer	$2FE3C_H$	P0, 32	P0, SV, 32	Kernel Reset	255
CPU2_HRB_SWID	CPU2 software identification	$2FE40_H$	P0, 32	P0, SV, 32	Kernel Reset	544
CPU2_HRB_CUS_ID	CPU2 customer ID register	$2FE50_H$	P0, 32	P0, SV, 32	Kernel Reset	546
CPU2_HRB_BOOTCON	CPU2 core boot configuration register	$2FE60_H$	P0, 32	P0, SV, 32	Kernel Reset	548
CPU2_HRB_LCLCON	CPU2 lockstep control register	$2FE64_H$	P0, 32	STP, SFRSE, P0, SV, 32	Cold PowerOn Reset	550

(table continues...)

Table 217 (continued) Register overview - CPU2 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU2_HRB_CCON	CPU2 clock control register	$2FE68_H$	P0, 32	P0, SV, 32	Kernel Reset	248
CPU2_HRB_TCCON	CPU2 hardware configuration	$2FE6C_H$	P0, 32	P0, SV, 32	Kernel Reset	552
CPU2_HRB_Ay (y=0-1)	CPU2 HR specific address general purpose register y	$2FF80_H$+ y*4	P0, 32	P0, SV, 32	Kernel Reset	274
CPU2_HRB_Ay (y=8-9)	CPU2 HR specific address general purpose register y	$2FFA0_H$+ (y-8)*4	P0, 32	P0, SV, 32	Kernel Reset	274
CPU2_HRHV_SEGEN	CPU2 SRI error generation register	31030_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	255
CPU2_HRHV_LCLTEST	CPU2 lockstep test register	31040_H	P0, 32	P0, SV, 32	Cold PowerOn Reset	256
CPU2_HRHV_PMA0	CPU2 data access cacheability register	38100_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	232
CPU2_HRHV_PMA1	CPU2 code access cacheability register	38104_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	233
CPU2_HRHV_PMA2	CPU2 peripheral space identifier register	38108_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	234
CPU2_HRHV_DCON2	CPU2 data control register 2	39000_H	P0, 32	P0, SV, 32	Kernel Reset	558
CPU2_HRHV_DCON1	CPU2 data control register 1	39008_H	P0, 32	P0, SV, 32	Kernel Reset	258
CPU2_HRHV_SMACON	CPU2 SIST mode access control register	$3900C_H$	P0, 32	SFRSE, P0, SV, 32	Kernel Reset	234
CPU2_HRHV_DSTR	CPU2 HR specific data synchronous trap register	39010_H	P0, 32	P0, SV, 32	Kernel Reset	259
CPU2_HRHV_DATR	CPU2 HR specific data asynchronous trap register	39018_H	P0, 32	P0, SV, 32	Kernel Reset	261
CPU2_HRHV_DEADD	CPU2 HR specific data error address register	$3901C_H$	P0, 32	P0, SV, 32	Kernel Reset	262
CPU2_HRHV_DIEAR	CPU2 data integrity error address register	39020_H	P0, 32	P0, SV, 32	Kernel Reset	279
CPU2_HRHV_DIETR	CPU2 data integrity error trap register	39024_H	P0, 32	P0, SV, 32	Kernel Reset	280
CPU2_HRHV_DCON0	CPU2 data memory control register	39040_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	258
CPU2_HRHV_PSTR	CPU2 HR specific program synchronous trap register	39200_H	P0, 32	P0, SV, 32	Kernel Reset	295
CPU2_HRHV_PCON1	CPU2 program control 1	39204_H	P0, 32	P0, SV, 32	Kernel Reset	292

(table continues...)

Table 217 (continued) Register overview - CPU2 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU2_HRHV_PCON2	CPU2 program control 2	39208_H	P0, 32	P0, SV, 32	Kernel Reset	561
CPU2_HRHV_PCON0	CPU2 program control 0	$3920C_H$	P0, 32	SFRE, P0, SV, 32	Kernel Reset	294
CPU2_HRHV_PIEAR	CPU2 HR specific program integrity error address register	39210_H	P0, 32	P0, SV, 32	Kernel Reset	281
CPU2_HRHV_PIETR	CPU2 HR specific program integrity error trap register	39214_H	P0, 32	P0, SV, 32	Kernel Reset	282
CPU2_HRHV_COMPAT	CPU2 compatibility control register	39400_H	P0, 32	SFRSE, P0, SV, 32	Kernel Reset	235
CPU2_HRHV_FPU_TRAP_CON	CPU2 HR specific trap control register	$3A000_H$	P0, 32	P0, SV, 32	Kernel Reset	263
CPU2_HRHV_FPU_TRAP_PC	CPU2 HR specific trapping instruction program counter register	$3A004_H$	P0, 32	P0, SV, 32	Kernel Reset	265
CPU2_HRHV_FPU_TRAP_OPC	CPU2 HR specific trapping instruction opcode register	$3A008_H$	P0, 32	P0, SV, 32	Kernel Reset	266
CPU2_HRHV_FPU_TRAP_SRC1_L	CPU2 HR specific trapping instruction operand register	$3A010_H$	P0, 32	P0, SV, 32	Kernel Reset	267
CPU2_HRHV_FPU_TRAP_SRC1_U	CPU2 HR specific trapping instruction operand register	$3A014_H$	P0, 32	P0, SV, 32	Kernel Reset	268
CPU2_HRHV_FPU_TRAP_SRC2_L	CPU2 HR specific trapping instruction operand register	$3A018_H$	P0, 32	P0, SV, 32	Kernel Reset	268
CPU2_HRHV_FPU_TRAP_SRC2_U	CPU2 HR specific trapping instruction operand register	$3A01C_H$	P0, 32	P0, SV, 32	Kernel Reset	269
CPU2_HRHV_FPU_TRAP_SRC3_L	CPU2 HR specific trapping instruction operand register	$3A020_H$	P0, 32	P0, SV, 32	Kernel Reset	270
CPU2_HRHV_FPU_TRAP_SRC3_U	CPU2 HR specific trapping instruction operand register	$3A024_H$	P0, 32	P0, SV, 32	Kernel Reset	270
CPU2_HRHV_FPU_SYNC_TRAP_CON	CPU2 HR specific synchronous trap control register	$3A030_H$	P0, 32	P0, SV, 32	Kernel Reset	271
CPU2_HRHV_FPU_SYNC_TRAP_OPC	CPU2 HR specific trapping instruction opcode register	$3A034_H$	P0, 32	P0, SV, 32	Kernel Reset	273
CPU2_HRHV_VCON0	CPU2 virtualization control register 0	$3B000_H$	P0, 32	P0, SV, 32	Kernel Reset	275

(table continues...)

Table 217 (continued) Register overview - CPU2 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU2_HRHV_VCON1	CPU2 virtualization control register 1	$3B004_H$	P0, 32	P0, SV, 32	Kernel Reset	276
CPU2_HRHV_VCON2	CPU2 virtualization control register 2	$3B008_H$	P0, 32	P0, SV, 32	Kernel Reset	276
CPU2_HRHV_BHV	CPU2 base hypervisor vector table pointer	$3B010_H$	P0, 32	P0, SV, 32, SFRE	Kernel Reset	559
CPU2_HRHV_VMn_ICR (n=0-7)	CPU2 Virtual machine n interrupt state	$3B100_H$+n*4	P0, 32	P0, SV, 32	Kernel Reset	278
CPU2_HRHV_VMn_PETHRESH (n=0-7)	CPU2 Virtual machine n pre-emption threshold	$3B200_H$+n*4	P0, 32	P0, SV, 32	Kernel Reset	279
CPU2_HRHV_DPRy_L (y=0-23)	CPU2 HR specific data protection range y, lower bound register	$3C000_H$+y*8	P0, 32	P0, SV, 32	Kernel Reset	283
CPU2_HRHV_DPRy_U (y=0-23)	CPU2 HR specific data protection range y, upper bound register	$3C004_H$+y*8	P0, 32	P0, SV, 32	Kernel Reset	284
CPU2_HRHV_CPRy_L (y=0-15)	CPU2 HR specific code protection range y, lower bound register	$3D000_H$+y*8	P0, 32	P0, SV, 32	Kernel Reset	285
CPU2_HRHV_CPRy_U (y=0-15)	CPU2 HR specific code protection range y, upper bound register	$3D004_H$+y*8	P0, 32	P0, SV, 32	Kernel Reset	286
CPU2_HRHV_CPXE_y (y=0-3)	CPU2 HR specific code protection execute enable register set y	$3E000_H$+y*4	P0, 32	P0, SV, 32	Kernel Reset	287
CPU2_HRHV_DPRE_y (y=0-3)	CPU2 HR specific data protection read enable register set y	$3E010_H$+y*4	P0, 32	P0, SV, 32	Kernel Reset	288
CPU2_HRHV_DPWE_y (y=0-3)	CPU2 HR specific data protection write enable register set y	$3E020_H$+y*4	P0, 32	P0, SV, 32	Kernel Reset	289
CPU2_HRHV_CPXE_y (y=4-7)	CPU2 HR specific code protection execute enable register set y	$3E040_H$+(y-4)*4	P0, 32	P0, SV, 32	Kernel Reset	287
CPU2_HRHV_DPRE_y (y=4-7)	CPU2 HR specific data protection read enable register set y	$3E050_H$+(y-4)*4	P0, 32	P0, SV, 32	Kernel Reset	288
CPU2_HRHV_DPWE_y (y=4-7)	CPU2 HR specific data protection write enable register set y	$3E060_H$+(y-4)*4	P0, 32	P0, SV, 32	Kernel Reset	289

(table continues...)

Table 217 (continued) Register overview - CPU2 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU2_HRHV_TPS_CON	CPU2 HR specific temporal protection system control register	$3E400_H$	P0, 32	P0, SV, 32	Kernel Reset	296
CPU2_HRHV_TPS_TIMERn (n=0-2)	CPU2 HR specific temporal protection system timer register n	$3E404_H$+ n*4	P0, 32	P0, SV, 32	Kernel Reset	297
CPU2_HRHV_TRnEVT (n=0-7)	CPU2 trigger event n	$3F000_H$+ n*8	P0, 32	P0, SV, 32	Debug Reset	209
CPU2_HRHV_TRnADR (n=0-7)	CPU2 trigger address n	$3F004_H$+ n*8	P0, 32	P0, SV, 32	Debug Reset	210
CPU2_HRHV_CCTRL	CPU2 HR specific counter control	$3FC00_H$	P0, 32	P0, SV, 32	Kernel Reset	226
CPU2_HRHV_CCNT	CPU2 HR specific CPU clock cycle count	$3FC04_H$	P0, 32	P0, SV, 32	Kernel Reset	227
CPU2_HRHV_ICNT	CPU2 HR specific instruction count	$3FC08_H$	P0, 32	P0, SV, 32	Kernel Reset	228
CPU2_HRHV_M1CNT	CPU2 HR specific multi-count register 1	$3FC0C_H$	P0, 32	P0, SV, 32	Kernel Reset	229
CPU2_HRHV_M2CNT	CPU2 HR specific multi-count register 2	$3FC10_H$	P0, 32	P0, SV, 32	Kernel Reset	230
CPU2_HRHV_M3CNT	CPU2 HR specific multi-count register 3	$3FC14_H$	P0, 32	P0, SV, 32	Kernel Reset	231
CPU2_HRHV_DBGSR	CPU2 debug status register	$3FD00_H$	P0, 32	P0, SV, 32	Debug Reset	211
CPU2_HRHV_EXEVT	CPU2 external event register	$3FD08_H$	P0, 32	P0, SV, 32	Debug Reset	212
CPU2_HRHV_CREVT	CPU2 core register access event	$3FD0C_H$	P0, 32	P0, SV, 32	Debug Reset	213
CPU2_HRHV_SWEVT	CPU2 software debug event	$3FD10_H$	P0, 32	P0, SV, 32	Debug Reset	214
CPU2_HRHV_DBGACT	CPU2 debug action configuration	$3FD14_H$	P0, 32	P0, SV, 32	Debug Reset	215
CPU2_HRHV_TRIG_ACC	CPU2 trigger event accumulator	$3FD30_H$	P0, 32	P0, SV, 32	Debug Reset	217
CPU2_HRHV_DMS	CPU2 debug monitor start address	$3FD40_H$	P0, 32	P0, SV, 32	Debug Reset	534
CPU2_HRHV_DCX	CPU2 debug context save area pointer	$3FD44_H$	P0, 32	P0, SV, 32	Debug Reset	536
CPU2_HRHV_DBGTCR	CPU2 debug trap control	$3FD48_H$	P0, 32	P0, SV, 32	Application Reset	219

(table continues...)

Table 217 (continued) Register overview - CPU2 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU2_HRHV_DBGCFG	CPU2 debug configuration	$3FD4C_H$	P0, 32	P0, SV, 32	Debug Reset	220
CPU2_HRHV_TRCCFG	CPU2 trace configuration	$3FD50_H$	P0, 32	P0, SV, 32	Debug Reset	221
CPU2_HRHV_TRCFILT	CPU2 trace filter	$3FD54_H$	P0, 32	P0, SV, 32	Debug Reset	222
CPU2_HRHV_TRCLIM	CPU2 trace bandwidth limiter	$3FD58_H$	P0, 32	P0, SV, 32	Debug Reset	224
CPU2_HRHV_TS16PTCCTRL	CPU2 MCDS performance counter control register	$3FD60_H$	P0, 32	P0, SV, 32	Debug Reset	225
CPU2_HRHV_PCXI	CPU2 previous context information register	$3FE00_H$	P0, 32	P0, SV, 32	Kernel Reset	236
CPU2_HRHV_PSW	CPU2 HR specific program status word	$3FE04_H$	P0, 32	P0, SV, 32	Kernel Reset	237
CPU2_HRHV_PC	CPU2 program counter	$3FE08_H$	P0, 32	P0, SV, 32	Kernel Reset	538
CPU2_HRHV_CORECON	CPU2 HR specific core configuration register	$3FE14_H$	P0, 32	SFRSE, P0, SV, 32	Kernel Reset	250
CPU2_HRHV_CPU_ID	CPU2 identification register	$3FE18_H$	P0, 32	P0, SV, 32	Kernel Reset	540
CPU2_HRHV_CORE_ID	CPU2 core identification register	$3FE1C_H$	P0, 32	P0, SV, 32	Kernel Reset	542
CPU2_HRHV_BIV	CPU2 HR specific base interrupt vector table pointer	$3FE20_H$	P0, 32	SFRE, P0, SV, 32	Kernel Reset	251
CPU2_HRHV_BTV	CPU2 HR specific base trap vector table pointer	$3FE24_H$	P0, 32	SFRE, P0, SV, 32	Kernel Reset	555
CPU2_HRHV_ISP	CPU2 HR specific interrupt stack pointer	$3FE28_H$	P0, 32	SFRE, P0, SV, 32	Kernel Reset	253
CPU2_HRHV_ICR	CPU2 interrupt control register	$3FE2C_H$	P0, 32	P0, SV, 32	Kernel Reset	242
CPU2_HRHV_PPRS	CPU2 previous PRS	$3FE34_H$	P0, 32	P0, SV, 32	Kernel Reset	244
CPU2_HRHV_FCX	CPU2 HR specific free CSA list head pointer	$3FE38_H$	P0, 32	P0, SV, 32	Kernel Reset	254
CPU2_HRHV_LCX	CPU2 HR specific free CSA list limit pointer	$3FE3C_H$	P0, 32	P0, SV, 32	Kernel Reset	255
CPU2_HRHV_SWID	CPU2 software identification	$3FE40_H$	P0, 32	P0, SV, 32	Kernel Reset	544
CPU2_HRHV_CUS_ID	CPU2 customer ID register	$3FE50_H$	P0, 32	P0, SV, 32	Kernel Reset	546
CPU2_HRHV_BOOTCON	CPU2 core boot configuration register	$3FE60_H$	P0, 32	P0, SV, 32	Kernel Reset	548

(table continues...)

Table 217 (continued) Register overview - CPU2 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU2_HRHV_LCLCON	CPU2 lockstep control register	$3FE64_H$	P0, 32	STP, SFRSE, P0, SV, 32	Cold PowerOn Reset	550
CPU2_HRHV_CCON	CPU2 clock control register	$3FE68_H$	P0, 32	P0, SV, 32	Kernel Reset	248
CPU2_HRHV_TCCON	CPU2 hardware configuration	$3FE6C_H$	P0, 32	P0, SV, 32	Kernel Reset	552
CPU2_HRHV_Ay (y=0-1)	CPU2 HR specific address general purpose register y	$3FF80_H$+ y*4	P0, 32	P0, SV, 32	Kernel Reset	274
CPU2_HRHV_Ay (y=8-9)	CPU2 HR specific address general purpose register y	$3FFA0_H$+ (y-8)*4	P0, 32	P0, SV, 32	Kernel Reset	274

4.10.4.1.8 Register overview - CPU3 domain SRI (ascending offset address)

Table 218 Register overview - CPU3 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU3_STM_CLC	CPU3 system timer clock control register	00000_H	P0, 32	P0, SV, SFRE, 32	System Reset	351
CPU3_STMHV_CLC	CPU3 system timer clock control register, hypervisor view	00000_H	P0, 32	P0, SV, SFRE, 32	System Reset	351
CPU3_STM_OCS	CPU3 OCDS control and status register	00004_H	P0, 32	P0, SV, 32, DEN	Debug Reset	352
CPU3_STMHV_OCS	CPU3 OCDS control and status register, hypervisor view	00004_H	P0, 32	P0, SV, 32, DEN	Debug Reset	352
CPU3_STM_ABS	CPU3 system timer absolute timer	00020_H	(P0, 32)| (P0, 64)	P0, SV, 64, SFRSE	System Reset	353
CPU3_STMHV_ABS	CPU3 system timer absolute timer, hypervisor view	00020_H	(P0, 32)| (P0, 64)	P0, SV, 64, SFRSE	System Reset	353
CPU3_STMHV_VMn_CMPc (c=0-1;n=0-7)	CPU3 VMn system timer compare register c, hypervisor view	00100_H+ n*20_H+c *4	P1, 32	P1, SV, 32	System Reset	354
CPU3_STMHV_VMn_CMCON (n=0-7)	CPU3 VMn system timer compare match control register, hypervisor view	00108_H+ n*20_H	P1, 32	P1, SV, 32	System Reset	355
CPU3_STMHV_VMn_ICR (n=0-7)	CPU3 VMn system timer interrupt control register, hypervisor view	$0010C_H$+ n*20_H	P1, 32	P1, SV, 32	System Reset	356

(table continues...)

Table 218 (continued) Register overview - CPU3 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU3_STMHV_VMn_ISCR (n=0-7)	CPU3 VMn system timer interrupt set/clear register, hypervisor view	00110_H+ n*20_H	P1, 32	P1, SV, 32	System Reset	357
CPU3_STMHV_VMn_ISR (n=0-7)	CPU3 VMn system timer interrupt status register, hypervisor view	00114_H+ n*20_H	P1, 32	P1, SV, 32	System Reset	358
CPU3_STMHV_VMn_RELTIM (n=0-7)	CPU3 VMn system timer relative timer, hypervisor view	00118_H+ n*20_H	(P1, 32)| (P1, 64)	P1, SV, 64	System Reset	359
CPU3_STM_CMPc (c=0-1)	CPU3 system timer compare register c	00120_H+ c*4	P1, 32	P1, SV, 32	System Reset	354
CPU3_STM_CMCON	CPU3 system timer compare match control register	00128_H	P1, 32	P1, SV, 32	System Reset	355
CPU3_STM_ICR	CPU3 system timer interrupt control register	$0012C_H$	P1, 32	P1, SV, 32	System Reset	356
CPU3_STM_ISCR	CPU3 system timer interrupt set/clear register	00130_H	P1, 32	P1, SV, 32	System Reset	357
CPU3_STM_ISR	CPU3 system timer interrupt status register	00134_H	P1, 32	P1, SV, 32	System Reset	358
CPU3_STM_RELTIM	CPU3 system timer relative timer	00138_H	(P1, 32)| (P1, 64)	P1, SV, 64	System Reset	359
CPU3_KRST0	CPU3 reset register 0	$0D000_H$	P0, 32	RSTE, P0, SV, 32	Kernel Reset	297
CPU3_KRST1	CPU3 reset register 1	$0D004_H$	P0, 32	RSTE, P0, SV, 32	Kernel Reset	298
CPU3_KRSTCLR	CPU3 reset clear register	$0D008_H$	P0, 32	RSTE, P0, SV, 32	Kernel Reset	299
CPU3_PROTSPRSE	CPU3 PROTSPR register safe endinit	$0E008_H$	32	SV, 32, PROT	Application Reset	562
CPU3_ACCENSPRCFG_WRA	CPU3 shadow SPR, write access enable register A	$0E020_H$	32	SPRSE, SV, 32	Application Reset	565
CPU3_ACCENSPRCFG_WRB	CPU3 shadow SPR, write access enable register B	$0E024_H$	32	SPRSE, SV, 32	Application Reset	566
CPU3_ACCENSPRCFG_RDA	CPU3 shadow SPR, read access enable register A	$0E028_H$	32	SPRSE, SV, 32	Application Reset	567
CPU3_ACCENSPRCFG_RDB	CPU3 shadow SPR, read access enable register B	$0E02C_H$	32	SPRSE, SV, 32	Application Reset	568
CPU3_ACCENSPRCFG_VM	CPU3 shadow SPR, VM access enable register	$0E030_H$	32	SPRSE, SV, 32	Application Reset	310

(table continues...)

Table 218 (continued) Register overview - CPU3 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU3_ACCENSPRCFG_PRS	CPU3 shadow SPR, PRS access enable register	$0E034_H$	32	SPRSE, SV, 32	Application Reset	310
CPU3_ACCENSPRCFG_RGNLA	CPU3 shadow SPR, region lower address register	$0E038_H$	32	SPRSE, SV, 32	Application Reset	311
CPU3_ACCENSPRCFG_RGNUA	CPU3 shadow SPR, region upper address register	$0E03C_H$	32	SPRSE, SV, 32	Application Reset	312
CPU3_PROTDLMUSE	CPU3 PROTDLMU register safe endinit	$0E048_H$	32	SV, 32, PROT	Application Reset	312
CPU3_ACCENDLMUCFG_WRA	CPU3 shadow DLMU, write access enable register A	$0E060_H$	32	DLMUSE, SV, 32	Application Reset	314
CPU3_ACCENDLMUCFG_WRB	CPU3 shadow DLMU, write access enable register B	$0E064_H$	32	DLMUSE, SV, 32	Application Reset	315
CPU3_ACCENDLMUCFG_RDA	CPU3 shadow DLMU, read access enable register A	$0E068_H$	32	DLMUSE, SV, 32	Application Reset	315
CPU3_ACCENDLMUCFG_RDB	CPU3 shadow DLMU, read access enable register B	$0E06C_H$	32	DLMUSE, SV, 32	Application Reset	316
CPU3_ACCENDLMUCFG_VM	CPU3 shadow DLMU, VM access enable register	$0E070_H$	32	DLMUSE, SV, 32	Application Reset	316
CPU3_ACCENDLMUCFG_PRS	CPU3 shadow DLMU, PRS access enable register	$0E074_H$	32	DLMUSE, SV, 32	Application Reset	317
CPU3_ACCENDLMUCFG_RGNLA	CPU3 shadow DLMU, region lower address register	$0E078_H$	32	DLMUSE, SV, 32	Application Reset	317
CPU3_ACCENDLMUCFG_RGNUA	CPU3 shadow DLMU, region upper address register	$0E07C_H$	32	DLMUSE, SV, 32	Application Reset	318
CPU3_PROTSFRE	CPU3 PROTSFR register endinit	$0E080_H$	32	SV, 32, PROT	Application Reset	569
CPU3_PROTSFRSE	CPU3 PROTSFR register safe endinit	$0E088_H$	32	SV, 32, PROT	Application Reset	571
CPU3_ACCENSFRCFG_WRA	CPU3 shadow SFR, write access enable register A	$0E0A0_H$	32	SFRSE, SV, 32	Application Reset	573
CPU3_ACCENSFRCFG_WRB	CPU3 shadow SFR, write access enable register B	$0E0A4_H$	32	SFRSE, SV, 32	Application Reset	574
CPU3_ACCENSFRCFG_RDA	CPU3 shadow SFR, read access enable register A	$0E0A8_H$	32	SFRSE, SV, 32	Application Reset	575
CPU3_ACCENSFRCFG_RDB	CPU3 shadow SFR, read access enable register B	$0E0AC_H$	32	SFRSE, SV, 32	Application Reset	576
CPU3_ACCENSFRCFG_VM	CPU3 shadow SFR, VM access enable register	$0E0B0_H$	32	SFRSE, SV, 32	Application Reset	325
CPU3_ACCENSFRCFG_PRS	CPU3 shadow SFR, PRS access enable register	$0E0B4_H$	32	SFRSE, SV, 32	Application Reset	325
CPU3_PROTRSTE	CPU3 PROTRST register endinit	$0E0C0_H$	32	SV, 32, PROT	Application Reset	577

(table continues...)

Table 218 (continued) Register overview - CPU3 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU3_PROTSTMSE	CPU3 PROTSTM Register Safe Endinit	$0E0D8_H$	32	SV, 32, PROT	Application Reset	579
CPU3_ACCENSTMCFG_WRA	CPU3 shadow STM, write access enable register A	$0E0E0_H$	32	STMSE, SV, 32	Application Reset	582
CPU3_ACCENSTMCFG_WRB	CPU3 shadow STM, write access enable register B	$0E0E4_H$	32	STMSE, SV, 32	Application Reset	330
CPU3_ACCENSTMCFG_RDA	CPU3 shadow STM, read access enable register A	$0E0E8_H$	32	STMSE, SV, 32	Application Reset	583
CPU3_ACCENSTMCFG_RDB	CPU3 shadow STM, read access enable register B	$0E0EC_H$	32	STMSE, SV, 32	Application Reset	584
CPU3_ACCENSTMCFG_VM	CPU3 shadow STM, VM access enable register	$0E0F0_H$	32	STMSE, SV, 32	Application Reset	332
CPU3_ACCENSTMCFG_PRS	CPU3 shadow STM, PRS access enable register	$0E0F4_H$	32	STMSE, SV, 32	Application Reset	333
CPU3_ACCENSPRy_WRA (y=0-15)	CPU3 SPR range y, write access enable register A	$0E200_H$+ y*20_H	32	SPRSE, SV, 32	Application Reset	585
CPU3_ACCENSPRy_WRB (y=0-15)	CPU3 SPR range y, write access enable register B	$0E204_H$+ y*20_H	32	SPRSE, SV, 32	Application Reset	586
CPU3_ACCENSPRy_RDA (y=0-15)	CPU3 SPR range y, read access enable register A	$0E208_H$+ y*20_H	32	SPRSE, SV, 32	Application Reset	587
CPU3_ACCENSPRy_RDB (y=0-15)	CPU3 SPR range y, read access enable register B	$0E20C_H$+ y*20_H	32	SPRSE, SV, 32	Application Reset	588
CPU3_ACCENSPRy_VM (y=0-15)	CPU3 SPR range y, VM access enable register	$0E210_H$+ y*20_H	32	SPRSE, SV, 32	Application Reset	336
CPU3_ACCENSPRy_PRS (y=0-15)	CPU3 SPR range y, PRS access enable register	$0E214_H$+ y*20_H	32	SPRSE, SV, 32	Application Reset	337
CPU3_ACCENSPRy_RGNLA (y=0-15)	CPU3 SPR range y, region lower address register	$0E218_H$+ y*20_H	32	SPRSE, SV, 32	Application Reset	337
CPU3_ACCENSPRy_RGNUA (y=0-15)	CPU3 SPR range y, region upper address register	$0E21C_H$+ y*20_H	32	SPRSE, SV, 32	Application Reset	338
CPU3_ACCENDLMUy_WRA (y=0-15)	CPU3 DLMU range y, write access enable register A	$0E400_H$+ y*20_H	32	DLMUSE, SV, 32	Application Reset	338

(table continues...)

Table 218 (continued) Register overview - CPU3 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU3_ACCENDLMUy_WRB (y=0-15)	CPU3 DLMU range y, write access enable register B	$0E404_H$+ y*20_H	32	DLMUSE, SV, 32	Application Reset	339
CPU3_ACCENDLMUy_RDA (y=0-15)	CPU3 DLMU range y, read access enable register A	$0E408_H$+ y*20_H	32	DLMUSE, SV, 32	Application Reset	339
CPU3_ACCENDLMUy_RDB (y=0-15)	CPU3 DLMU range y, read access enable register B	$0E40C_H$+ y*20_H	32	DLMUSE, SV, 32	Application Reset	340
CPU3_ACCENDLMUy_VM (y=0-15)	CPU3 DLMU range y, VM access enable register	$0E410_H$+ y*20_H	32	DLMUSE, SV, 32	Application Reset	340
CPU3_ACCENDLMUy_PRS (y=0-15)	CPU3 DLMU range y, PRS access enable register	$0E414_H$+ y*20_H	32	DLMUSE, SV, 32	Application Reset	341
CPU3_ACCENDLMUy_RGNLA (y=0-15)	CPU3 DLMU range y, region lower address register	$0E418_H$+ y*20_H	32	DLMUSE, SV, 32	Application Reset	342
CPU3_ACCENDLMUy_RGNUA (y=0-15)	CPU3 DLMU range y, region upper address register	$0E41C_H$+ y*20_H	32	DLMUSE, SV, 32	Application Reset	342
CPU3_ACCENSFR_WRA	CPU3 SFR, write access enable register A	$0E600_H$	32	SFRSE, SV, 32	Application Reset	589
CPU3_ACCENSFR_WRB	CPU3 SFR, write access enable register B	$0E604_H$	32	SFRSE, SV, 32	Application Reset	590
CPU3_ACCENSFR_RDA	CPU3 SFR, read access enable register A	$0E608_H$	32	SFRSE, SV, 32	Application Reset	591
CPU3_ACCENSFR_RDB	CPU3 SFR, read access enable register B	$0E60C_H$	32	SFRSE, SV, 32	Application Reset	592
CPU3_ACCENSFR_VM	CPU3 SFR, VM access enable register	$0E610_H$	32	SFRSE, SV, 32	Application Reset	345
CPU3_ACCENSFR_PRS	CPU3 SFR, PRS access enable register	$0E614_H$	32	SFRSE, SV, 32	Application Reset	346
CPU3_ACCENSTMVMy_WRA (y=0-7)	CPU3 VMy STM, write access enable register A, virtualization view	$0E700_H$+ y*20_H	32	STMSE, SV, 32	Application Reset	593
CPU3_ACCENSTMVMy_WRB (y=0-7)	CPU3 VMy STM, write access enable register B, virtualization view	$0E704_H$+ y*20_H	32	STMSE, SV, 32	Application Reset	347
CPU3_ACCENSTMVMy_RDA (y=0-7)	CPU3 VMy STM, read access enable register A, virtualization view	$0E708_H$+ y*20_H	32	STMSE, SV, 32	Application Reset	594

(table continues...)

Table 218 (continued) Register overview - CPU3 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU3_ACCENSTMVMy_RDB (y=0-7)	CPU3 VMy STM, read access enable register B, virtualization view	$0E70C_H$+ y*20_H	32	STMSE, SV, 32	Application Reset	595
CPU3_ACCENSTMVMy_VM (y=0-7)	CPU3 VMy STM, VM access enable register, virtualization view	$0E710_H$+ y*20_H	32	STMSE, SV, 32	Application Reset	350
CPU3_ACCENSTMVMy_PRS (y=0-7)	CPU3 VMy STM, PRS access enable register, virtualization view	$0E714_H$+ y*20_H	32	STMSE, SV, 32	Application Reset	350
CPU3_OSEL	CPU3 overlay range select register	$0FB00_H$	P0, 32	P0, SV, 32	Kernel Reset	299
CPU3_RABRi (i=0-31)	CPU3 redirected address base register i	$0FB10_H$+ i*12	P0, 32	P0, SV, 32	Kernel Reset	300
CPU3_OTARi (i=0-31)	CPU3 overlay target address register i	$0FB14_H$+ i*12	P0, 32	P0, SV, 32	Kernel Reset	302
CPU3_OMASKi (i=0-31)	CPU3 overlay mask register i	$0FB18_H$+ i*12	P0, 32	P0, SV, 32	Kernel Reset	303
CPU3_HRA_SEGEN	CPU3 SRI error generation register	11030_H	P0, 32	SFRE, P0, SV, 32	Application Reset	255
CPU3_HRA_LCLTEST	CPU3 lockstep test register	11040_H	P0, 32	P0, SV, 32	Cold PowerOn Reset	256
CPU3_HRA_PMA0	CPU3 data access cacheability register	18100_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	232
CPU3_HRA_PMA1	CPU3 code access cacheability register	18104_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	233
CPU3_HRA_PMA2	CPU3 peripheral space identifier register	18108_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	234
CPU3_HRA_DCON2	CPU3 data control register 2	19000_H	P0, 32	P0, SV, 32	Kernel Reset	558
CPU3_HRA_DCON1	CPU3 data control register 1	19008_H	P0, 32	P0, SV, 32	Kernel Reset	258
CPU3_HRA_SMACON	CPU3 SIST mode access control register	$1900C_H$	P0, 32	SFRSE, P0, SV, 32	Kernel Reset	234
CPU3_HRA_DSTR	CPU3 HR specific data synchronous trap register	19010_H	P0, 32	P0, SV, 32	Kernel Reset	259
CPU3_HRA_DATR	CPU3 HR specific data asynchronous trap register	19018_H	P0, 32	P0, SV, 32	Kernel Reset	261
CPU3_HRA_DEADD	CPU3 HR specific data error address register	$1901C_H$	P0, 32	P0, SV, 32	Kernel Reset	262
CPU3_HRA_DIEAR	CPU3 data integrity error address register	19020_H	P0, 32	P0, SV, 32	Kernel Reset	279

(table continues...)

Table 218 (continued) Register overview - CPU3 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU3_HRA_DIETR	CPU3 data integrity error trap register	19024_H	P0, 32	P0, SV, 32	Kernel Reset	280
CPU3_HRA_DCON0	CPU3 data memory control register	19040_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	258
CPU3_HRA_PSTR	CPU3 HR specific program synchronous trap register	19200_H	P0, 32	P0, SV, 32	Kernel Reset	295
CPU3_HRA_PCON1	CPU3 program control 1	19204_H	P0, 32	P0, SV, 32	Kernel Reset	292
CPU3_HRA_PCON2	CPU3 program control 2	19208_H	P0, 32	P0, SV, 32	Kernel Reset	561
CPU3_HRA_PCON0	CPU3 program control 0	$1920C_H$	P0, 32	SFRE, P0, SV, 32	Kernel Reset	294
CPU3_HRA_PIEAR	CPU3 HR specific program integrity error address register	19210_H	P0, 32	P0, SV, 32	Kernel Reset	281
CPU3_HRA_PIETR	CPU3 HR specific program integrity error trap register	19214_H	P0, 32	P0, SV, 32	Kernel Reset	282
CPU3_HRA_COMPAT	CPU3 compatibility control register	19400_H	P0, 32	SFRSE, P0, SV, 32	Kernel Reset	235
CPU3_HRA_FPU_TRAP_CON	CPU3 HR specific trap control register	$1A000_H$	P0, 32	P0, SV, 32	Kernel Reset	263
CPU3_HRA_FPU_TRAP_PC	CPU3 HR specific trapping instruction program counter register	$1A004_H$	P0, 32	P0, SV, 32	Kernel Reset	265
CPU3_HRA_FPU_TRAP_OPC	CPU3 HR specific trapping instruction opcode register	$1A008_H$	P0, 32	P0, SV, 32	Kernel Reset	266
CPU3_HRA_FPU_TRAP_SRC1_L	CPU3 HR specific trapping instruction operand register	$1A010_H$	P0, 32	P0, SV, 32	Kernel Reset	267
CPU3_HRA_FPU_TRAP_SRC1_U	CPU3 HR specific trapping instruction operand register	$1A014_H$	P0, 32	P0, SV, 32	Kernel Reset	268
CPU3_HRA_FPU_TRAP_SRC2_L	CPU3 HR specific trapping instruction operand register	$1A018_H$	P0, 32	P0, SV, 32	Kernel Reset	268
CPU3_HRA_FPU_TRAP_SRC2_U	CPU3 HR specific trapping instruction operand register	$1A01C_H$	P0, 32	P0, SV, 32	Kernel Reset	269
CPU3_HRA_FPU_TRAP_SRC3_L	CPU3 HR specific trapping instruction operand register	$1A020_H$	P0, 32	P0, SV, 32	Kernel Reset	270

(table continues...)

Table 218 (continued) Register overview - CPU3 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU3_HRA_FPU_TRAP_SRC3_U	CPU3 HR specific trapping instruction operand register	$1A024_H$	P0, 32	P0, SV, 32	Kernel Reset	270
CPU3_HRA_FPU_SYNC_TRAP_CON	CPU3 HR specific synchronous trap control register	$1A030_H$	P0, 32	P0, SV, 32	Kernel Reset	271
CPU3_HRA_FPU_SYNC_TRAP_OPC	CPU3 HR specific trapping instruction opcode register	$1A034_H$	P0, 32	P0, SV, 32	Kernel Reset	273
CPU3_HRA_DPRy_L (y=0-23)	CPU3 HR specific data protection range y, lower bound register	$1C000_H$+ y*8	P0, 32	P0, SV, 32	Kernel Reset	283
CPU3_HRA_DPRy_U (y=0-23)	CPU3 HR specific data protection range y, upper bound register	$1C004_H$+ y*8	P0, 32	P0, SV, 32	Kernel Reset	284
CPU3_HRA_CPRy_L (y=0-15)	CPU3 HR specific code protection range y, lower bound register	$1D000_H$+ y*8	P0, 32	P0, SV, 32	Kernel Reset	285
CPU3_HRA_CPRy_U (y=0-15)	CPU3 HR specific code protection range y, upper bound register	$1D004_H$+ y*8	P0, 32	P0, SV, 32	Kernel Reset	286
CPU3_HRA_CPXE_y (y=0-3)	CPU3 HR specific code protection execute enable register set y	$1E000_H$+ y*4	P0, 32	P0, SV, 32	Kernel Reset	287
CPU3_HRA_DPRE_y (y=0-3)	CPU3 HR specific data protection read enable register set y	$1E010_H$+ y*4	P0, 32	P0, SV, 32	Kernel Reset	288
CPU3_HRA_DPWE_y (y=0-3)	CPU3 HR specific data protection write enable register set y	$1E020_H$+ y*4	P0, 32	P0, SV, 32	Kernel Reset	289
CPU3_HRA_CPXE_y (y=4-7)	CPU3 HR specific code protection execute enable register set y	$1E040_H$+ (y-4)*4	P0, 32	P0, SV, 32	Kernel Reset	287
CPU3_HRA_DPRE_y (y=4-7)	CPU3 HR specific data protection read enable register set y	$1E050_H$+ (y-4)*4	P0, 32	P0, SV, 32	Kernel Reset	288
CPU3_HRA_DPWE_y (y=4-7)	CPU3 HR specific data protection write enable register set y	$1E060_H$+ (y-4)*4	P0, 32	P0, SV, 32	Kernel Reset	289
CPU3_HRA_TPS_CON	CPU3 HR specific temporal protection system control register	$1E400_H$	P0, 32	P0, SV, 32	Kernel Reset	296

(table continues...)

Table 218 (continued) Register overview - CPU3 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU3_HRA_TPS_TIMERn (n=0-2)	CPU3 HR specific temporal protection system timer register n	$1E404_H$+ n*4	P0, 32	P0, SV, 32	Kernel Reset	297
CPU3_HRA_TRnEVT (n=0-7)	CPU3 trigger event n	$1F000_H$+ n*8	P0, 32	P0, SV, 32	Debug Reset	209
CPU3_HRA_TRnADR (n=0-7)	CPU3 trigger address n	$1F004_H$+ n*8	P0, 32	P0, SV, 32	Debug Reset	210
CPU3_HRA_CCTRL	CPU3 HR specific counter control	$1FC00_H$	P0, 32	P0, SV, 32	Kernel Reset	226
CPU3_HRA_CCNT	CPU3 HR specific CPU clock cycle count	$1FC04_H$	P0, 32	P0, SV, 32	Kernel Reset	227
CPU3_HRA_ICNT	CPU3 HR specific instruction count	$1FC08_H$	P0, 32	P0, SV, 32	Kernel Reset	228
CPU3_HRA_M1CNT	CPU3 HR specific multi-count register 1	$1FC0C_H$	P0, 32	P0, SV, 32	Kernel Reset	229
CPU3_HRA_M2CNT	CPU3 HR specific multi-count register 2	$1FC10_H$	P0, 32	P0, SV, 32	Kernel Reset	230
CPU3_HRA_M3CNT	CPU3 HR specific multi-count register 3	$1FC14_H$	P0, 32	P0, SV, 32	Kernel Reset	231
CPU3_HRA_DBGSR	CPU3 debug status register	$1FD00_H$	P0, 32	P0, SV, 32	Debug Reset	211
CPU3_HRA_EXEVT	CPU3 external event register	$1FD08_H$	P0, 32	P0, SV, 32	Debug Reset	212
CPU3_HRA_CREVT	CPU3 core register access event	$1FD0C_H$	P0, 32	P0, SV, 32	Debug Reset	213
CPU3_HRA_SWEVT	CPU3 software debug event	$1FD10_H$	P0, 32	P0, SV, 32	Debug Reset	214
CPU3_HRA_DBGACT	CPU3 debug action configuration	$1FD14_H$	P0, 32	P0, SV, 32	Debug Reset	215
CPU3_HRA_TRIG_ACC	CPU3 trigger event accumulator	$1FD30_H$	P0, 32	P0, SV, 32	Debug Reset	217
CPU3_HRA_DMS	CPU3 debug monitor start address	$1FD40_H$	P0, 32	P0, SV, 32	Debug Reset	534
CPU3_HRA_DCX	CPU3 debug context save area pointer	$1FD44_H$	P0, 32	P0, SV, 32	Debug Reset	536
CPU3_HRA_DBGTCR	CPU3 debug trap control	$1FD48_H$	P0, 32	P0, SV, 32	Application Reset	219
CPU3_HRA_DBGCFG	CPU3 debug configuration	$1FD4C_H$	P0, 32	P0, SV, 32	Debug Reset	220

(table continues...)

Table 218 (continued) Register overview - CPU3 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU3_HRA_TRCCFG	CPU3 trace configuration	$1FD50_H$	P0, 32	P0, SV, 32	Debug Reset	221
CPU3_HRA_TRCFILT	CPU3 trace filter	$1FD54_H$	P0, 32	P0, SV, 32	Debug Reset	222
CPU3_HRA_TRCLIM	CPU3 trace bandwidth limiter	$1FD58_H$	P0, 32	P0, SV, 32	Debug Reset	224
CPU3_HRA_TS16PTCCTRL	CPU3 MCDS performance counter control register	$1FD60_H$	P0, 32	P0, SV, 32	Debug Reset	225
CPU3_HRA_PCXI	CPU3 previous context information register	$1FE00_H$	P0, 32	P0, SV, 32	Kernel Reset	236
CPU3_HRA_PSW	CPU3 HR specific program status word	$1FE04_H$	P0, 32	P0, SV, 32	Kernel Reset	237
CPU3_HRA_PC	CPU3 program counter	$1FE08_H$	P0, 32	P0, SV, 32	Kernel Reset	538
CPU3_HRA_CORECON	CPU3 HR specific core configuration register	$1FE14_H$	P0, 32	SFRSE, P0, SV, 32	Kernel Reset	250
CPU3_HRA_CPU_ID	CPU3 identification register	$1FE18_H$	P0, 32	P0, SV, 32	Kernel Reset	540
CPU3_HRA_CORE_ID	CPU3 core identification register	$1FE1C_H$	P0, 32	P0, SV, 32	Kernel Reset	542
CPU3_HRA_BIV	CPU3 HR specific base interrupt vector table pointer	$1FE20_H$	P0, 32	SFRE, P0, SV, 32	Kernel Reset	251
CPU3_HRA_BTV	CPU3 HR specific base trap vector table pointer	$1FE24_H$	P0, 32	SFRE, P0, SV, 32	Kernel Reset	554
CPU3_HRA_ISP	CPU3 HR specific interrupt stack pointer	$1FE28_H$	P0, 32	SFRE, P0, SV, 32	Kernel Reset	253
CPU3_HRA_ICR	CPU3 interrupt control register	$1FE2C_H$	P0, 32	P0, SV, 32	Kernel Reset	242
CPU3_HRA_PPRS	CPU3 previous PRS	$1FE34_H$	P0, 32	P0, SV, 32	Kernel Reset	244
CPU3_HRA_FCX	CPU3 HR specific free CSA list head pointer	$1FE38_H$	P0, 32	P0, SV, 32	Kernel Reset	254
CPU3_HRA_LCX	CPU3 HR specific free CSA list limit pointer	$1FE3C_H$	P0, 32	P0, SV, 32	Kernel Reset	255
CPU3_HRA_SWID	CPU3 software identification	$1FE40_H$	P0, 32	P0, SV, 32	Kernel Reset	544
CPU3_HRA_CUS_ID	CPU3 customer ID register	$1FE50_H$	P0, 32	P0, SV, 32	Kernel Reset	546
CPU3_HRA_BOOTCON	CPU3 core boot configuration register	$1FE60_H$	P0, 32	P0, SV, 32	Kernel Reset	548
CPU3_HRA_LCLCON	CPU3 lockstep control register	$1FE64_H$	P0, 32	STP, SFRSE, P0, SV, 32	Cold PowerOn Reset	550

(table continues...)

Table 218 (continued) Register overview - CPU3 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU3_HRA_CCON	CPU3 clock control register	$1FE68_H$	P0, 32	P0, SV, 32	Kernel Reset	248
CPU3_HRA_TCCON	CPU3 hardware configuration	$1FE6C_H$	P0, 32	P0, SV, 32	Kernel Reset	552
CPU3_Dy (y=0-15)	CPU3 data general purpose register y	$1FF00_H$+ y*4	P0, 32	P0, SV, 32	Kernel Reset	274
CPU3_Ay (y=0-15)	CPU3 address general purpose register y	$1FF80_H$+ y*4	P0, 32	P0, SV, 32	Kernel Reset	274
CPU3_HRB_SEGEN	CPU3 SRI error generation register	21030_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	255
CPU3_HRB_LCLTEST	CPU3 lockstep test register	21040_H	P0, 32	P0, SV, 32	Cold PowerOn Reset	256
CPU3_HRB_PMA0	CPU3 data access cacheability register	28100_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	232
CPU3_HRB_PMA1	CPU3 code access cacheability register	28104_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	233
CPU3_HRB_PMA2	CPU3 peripheral space identifier register	28108_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	234
CPU3_HRB_DCON2	CPU3 data control register 2	29000_H	P0, 32	P0, SV, 32	Kernel Reset	558
CPU3_HRB_DCON1	CPU3 data control register 1	29008_H	P0, 32	P0, SV, 32	Kernel Reset	258
CPU3_HRB_SMACON	CPU3 SIST mode access control register	$2900C_H$	P0, 32	SFRSE, P0, SV, 32	Kernel Reset	234
CPU3_HRB_DSTR	CPU3 HR specific data synchronous trap register	29010_H	P0, 32	P0, SV, 32	Kernel Reset	259
CPU3_HRB_DATR	CPU3 HR specific data asynchronous trap register	29018_H	P0, 32	P0, SV, 32	Kernel Reset	261
CPU3_HRB_DEADD	CPU3 HR specific data error address register	$2901C_H$	P0, 32	P0, SV, 32	Kernel Reset	262
CPU3_HRB_DIEAR	CPU3 data integrity error address register	29020_H	P0, 32	P0, SV, 32	Kernel Reset	279
CPU3_HRB_DIETR	CPU3 data integrity error trap register	29024_H	P0, 32	P0, SV, 32	Kernel Reset	280
CPU3_HRB_DCON0	CPU3 data memory control register	29040_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	258
CPU3_HRB_PSTR	CPU3 HR specific program synchronous trap register	29200_H	P0, 32	P0, SV, 32	Kernel Reset	295
CPU3_HRB_PCON1	CPU3 program control 1	29204_H	P0, 32	P0, SV, 32	Kernel Reset	292

(table continues...)

Table 218 (continued) Register overview - CPU3 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU3_HRB_PCON2	CPU3 program control 2	29208_H	P0, 32	P0, SV, 32	Kernel Reset	561
CPU3_HRB_PCON0	CPU3 program control 0	$2920C_H$	P0, 32	SFRE, P0, SV, 32	Kernel Reset	294
CPU3_HRB_PIEAR	CPU3 HR specific program integrity error address register	29210_H	P0, 32	P0, SV, 32	Kernel Reset	281
CPU3_HRB_PIETR	CPU3 HR specific program integrity error trap register	29214_H	P0, 32	P0, SV, 32	Kernel Reset	282
CPU3_HRB_COMPAT	CPU3 compatibility control register	29400_H	P0, 32	SFRSE, P0, SV, 32	Kernel Reset	235
CPU3_HRB_FPU_TRAP_CON	CPU3 HR specific trap control register	$2A000_H$	P0, 32	P0, SV, 32	Kernel Reset	263
CPU3_HRB_FPU_TRAP_PC	CPU3 HR specific trapping instruction program counter register	$2A004_H$	P0, 32	P0, SV, 32	Kernel Reset	265
CPU3_HRB_FPU_TRAP_OPC	CPU3 HR specific trapping instruction opcode register	$2A008_H$	P0, 32	P0, SV, 32	Kernel Reset	266
CPU3_HRB_FPU_TRAP_SRC1_L	CPU3 HR specific trapping instruction operand register	$2A010_H$	P0, 32	P0, SV, 32	Kernel Reset	267
CPU3_HRB_FPU_TRAP_SRC1_U	CPU3 HR specific trapping instruction operand register	$2A014_H$	P0, 32	P0, SV, 32	Kernel Reset	268
CPU3_HRB_FPU_TRAP_SRC2_L	CPU3 HR specific trapping instruction operand register	$2A018_H$	P0, 32	P0, SV, 32	Kernel Reset	268
CPU3_HRB_FPU_TRAP_SRC2_U	CPU3 HR specific trapping instruction operand register	$2A01C_H$	P0, 32	P0, SV, 32	Kernel Reset	269
CPU3_HRB_FPU_TRAP_SRC3_L	CPU3 HR specific trapping instruction operand register	$2A020_H$	P0, 32	P0, SV, 32	Kernel Reset	270
CPU3_HRB_FPU_TRAP_SRC3_U	CPU3 HR specific trapping instruction operand register	$2A024_H$	P0, 32	P0, SV, 32	Kernel Reset	270
CPU3_HRB_FPU_SYNC_TRAP_CON	CPU3 HR specific synchronous trap control register	$2A030_H$	P0, 32	P0, SV, 32	Kernel Reset	271
CPU3_HRB_FPU_SYNC_TRAP_OPC	CPU3 HR specific trapping instruction opcode register	$2A034_H$	P0, 32	P0, SV, 32	Kernel Reset	273

(table continues...)

Table 218 (continued) Register overview - CPU3 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU3_HRB_DPRy_L (y=0-23)	CPU3 HR specific data protection range y, lower bound register	$2C000_H$+y*8	P0, 32	P0, SV, 32	Kernel Reset	283
CPU3_HRB_DPRy_U (y=0-23)	CPU3 HR specific data protection range y, upper bound register	$2C004_H$+y*8	P0, 32	P0, SV, 32	Kernel Reset	284
CPU3_HRB_CPRy_L (y=0-15)	CPU3 HR specific code protection range y, lower bound register	$2D000_H$+y*8	P0, 32	P0, SV, 32	Kernel Reset	285
CPU3_HRB_CPRy_U (y=0-15)	CPU3 HR specific code protection range y, upper bound register	$2D004_H$+y*8	P0, 32	P0, SV, 32	Kernel Reset	286
CPU3_HRB_CPXE_y (y=0-3)	CPU3 HR specific code protection execute enable register set y	$2E000_H$+y*4	P0, 32	P0, SV, 32	Kernel Reset	287
CPU3_HRB_DPRE_y (y=0-3)	CPU3 HR specific data protection read enable register set y	$2E010_H$+y*4	P0, 32	P0, SV, 32	Kernel Reset	288
CPU3_HRB_DPWE_y (y=0-3)	CPU3 HR specific data protection write enable register set y	$2E020_H$+y*4	P0, 32	P0, SV, 32	Kernel Reset	289
CPU3_HRB_CPXE_y (y=4-7)	CPU3 HR specific code protection execute enable register set y	$2E040_H$+(y-4)*4	P0, 32	P0, SV, 32	Kernel Reset	287
CPU3_HRB_DPRE_y (y=4-7)	CPU3 HR specific data protection read enable register set y	$2E050_H$+(y-4)*4	P0, 32	P0, SV, 32	Kernel Reset	288
CPU3_HRB_DPWE_y (y=4-7)	CPU3 HR specific data protection write enable register set y	$2E060_H$+(y-4)*4	P0, 32	P0, SV, 32	Kernel Reset	289
CPU3_HRB_TPS_CON	CPU3 HR specific temporal protection system control register	$2E400_H$	P0, 32	P0, SV, 32	Kernel Reset	296
CPU3_HRB_TPS_TIMERn (n=0-2)	CPU3 HR specific temporal protection system timer register n	$2E404_H$+n*4	P0, 32	P0, SV, 32	Kernel Reset	297
CPU3_HRB_TRnEVT (n=0-7)	CPU3 trigger event n	$2F000_H$+n*8	P0, 32	P0, SV, 32	Debug Reset	209
CPU3_HRB_TRnADR (n=0-7)	CPU3 trigger address n	$2F004_H$+n*8	P0, 32	P0, SV, 32	Debug Reset	210

(table continues...)

Table 218 (continued) Register overview - CPU3 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU3_HRB_CCTRL	CPU3 HR specific counter control	$2FC00_H$	P0, 32	P0, SV, 32	Kernel Reset	226
CPU3_HRB_CCNT	CPU3 HR specific CPU clock cycle count	$2FC04_H$	P0, 32	P0, SV, 32	Kernel Reset	227
CPU3_HRB_ICNT	CPU3 HR specific instruction count	$2FC08_H$	P0, 32	P0, SV, 32	Kernel Reset	228
CPU3_HRB_M1CNT	CPU3 HR specific multi-count register 1	$2FC0C_H$	P0, 32	P0, SV, 32	Kernel Reset	229
CPU3_HRB_M2CNT	CPU3 HR specific multi-count register 2	$2FC10_H$	P0, 32	P0, SV, 32	Kernel Reset	230
CPU3_HRB_M3CNT	CPU3 HR specific multi-count register 3	$2FC14_H$	P0, 32	P0, SV, 32	Kernel Reset	231
CPU3_HRB_DBGSR	CPU3 debug status register	$2FD00_H$	P0, 32	P0, SV, 32	Debug Reset	211
CPU3_HRB_EXEVT	CPU3 external event register	$2FD08_H$	P0, 32	P0, SV, 32	Debug Reset	212
CPU3_HRB_CREVT	CPU3 core register access event	$2FD0C_H$	P0, 32	P0, SV, 32	Debug Reset	213
CPU3_HRB_SWEVT	CPU3 software debug event	$2FD10_H$	P0, 32	P0, SV, 32	Debug Reset	214
CPU3_HRB_DBGACT	CPU3 debug action configuration	$2FD14_H$	P0, 32	P0, SV, 32	Debug Reset	215
CPU3_HRB_TRIG_ACC	CPU3 trigger event accumulator	$2FD30_H$	P0, 32	P0, SV, 32	Debug Reset	217
CPU3_HRB_DMS	CPU3 debug monitor start address	$2FD40_H$	P0, 32	P0, SV, 32	Debug Reset	534
CPU3_HRB_DCX	CPU3 debug context save area pointer	$2FD44_H$	P0, 32	P0, SV, 32	Debug Reset	536
CPU3_HRB_DBGTCR	CPU3 debug trap control	$2FD48_H$	P0, 32	P0, SV, 32	Application Reset	219
CPU3_HRB_DBGCFG	CPU3 debug configuration	$2FD4C_H$	P0, 32	P0, SV, 32	Debug Reset	220
CPU3_HRB_TRCCFG	CPU3 trace configuration	$2FD50_H$	P0, 32	P0, SV, 32	Debug Reset	221
CPU3_HRB_TRCFILT	CPU3 trace filter	$2FD54_H$	P0, 32	P0, SV, 32	Debug Reset	222
CPU3_HRB_TRCLIM	CPU3 trace bandwidth limiter	$2FD58_H$	P0, 32	P0, SV, 32	Debug Reset	224
CPU3_HRB_TS16PTCCTRL	CPU3 MCDS performance counter control register	$2FD60_H$	P0, 32	P0, SV, 32	Debug Reset	225
CPU3_HRB_PCXI	CPU3 previous context information register	$2FE00_H$	P0, 32	P0, SV, 32	Kernel Reset	236

(table continues...)

Table 218 (continued) Register overview - CPU3 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU3_HRB_PSW	CPU3 HR specific program status word	$2FE04_H$	P0, 32	P0, SV, 32	Kernel Reset	237
CPU3_HRB_PC	CPU3 program counter	$2FE08_H$	P0, 32	P0, SV, 32	Kernel Reset	538
CPU3_HRB_CORECON	CPU3 HR specific core configuration register	$2FE14_H$	P0, 32	SFRSE, P0, SV, 32	Kernel Reset	250
CPU3_HRB_CPU_ID	CPU3 identification register	$2FE18_H$	P0, 32	P0, SV, 32	Kernel Reset	540
CPU3_HRB_CORE_ID	CPU3 core identification register	$2FE1C_H$	P0, 32	P0, SV, 32	Kernel Reset	542
CPU3_HRB_BIV	CPU3 HR specific base interrupt vector table pointer	$2FE20_H$	P0, 32	SFRE, P0, SV, 32	Kernel Reset	251
CPU3_HRB_BTV	CPU3 HR specific base trap vector table pointer	$2FE24_H$	P0, 32	SFRE, P0, SV, 32	Kernel Reset	555
CPU3_HRB_ISP	CPU3 HR specific interrupt stack pointer	$2FE28_H$	P0, 32	SFRE, P0, SV, 32	Kernel Reset	253
CPU3_HRB_ICR	CPU3 interrupt control register	$2FE2C_H$	P0, 32	P0, SV, 32	Kernel Reset	242
CPU3_HRB_PPRS	CPU3 previous PRS	$2FE34_H$	P0, 32	P0, SV, 32	Kernel Reset	244
CPU3_HRB_FCX	CPU3 HR specific free CSA list head pointer	$2FE38_H$	P0, 32	P0, SV, 32	Kernel Reset	254
CPU3_HRB_LCX	CPU3 HR specific free CSA list limit pointer	$2FE3C_H$	P0, 32	P0, SV, 32	Kernel Reset	255
CPU3_HRB_SWID	CPU3 software identification	$2FE40_H$	P0, 32	P0, SV, 32	Kernel Reset	544
CPU3_HRB_CUS_ID	CPU3 customer ID register	$2FE50_H$	P0, 32	P0, SV, 32	Kernel Reset	546
CPU3_HRB_BOOTCON	CPU3 core boot configuration register	$2FE60_H$	P0, 32	P0, SV, 32	Kernel Reset	548
CPU3_HRB_LCLCON	CPU3 lockstep control register	$2FE64_H$	P0, 32	STP, SFRSE, P0, SV, 32	Cold PowerOn Reset	550
CPU3_HRB_CCON	CPU3 clock control register	$2FE68_H$	P0, 32	P0, SV, 32	Kernel Reset	248
CPU3_HRB_TCCON	CPU3 hardware configuration	$2FE6C_H$	P0, 32	P0, SV, 32	Kernel Reset	552
CPU3_HRB_Ay (y=0-1)	CPU3 HR specific address general purpose register y	$2FF80_H$+ y*4	P0, 32	P0, SV, 32	Kernel Reset	274
CPU3_HRB_Ay (y=8-9)	CPU3 HR specific address general purpose register y	$2FFA0_H$+ (y-8)*4	P0, 32	P0, SV, 32	Kernel Reset	274

(table continues...)

Table 218 (continued) Register overview - CPU3 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU3_HRHV_SEGEN	CPU3 SRI error generation register	31030_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	255
CPU3_HRHV_LCLTEST	CPU3 lockstep test register	31040_H	P0, 32	P0, SV, 32	Cold PowerOn Reset	256
CPU3_HRHV_PMA0	CPU3 data access cacheability register	38100_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	232
CPU3_HRHV_PMA1	CPU3 code access cacheability register	38104_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	233
CPU3_HRHV_PMA2	CPU3 peripheral space identifier register	38108_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	234
CPU3_HRHV_DCON2	CPU3 data control register 2	39000_H	P0, 32	P0, SV, 32	Kernel Reset	558
CPU3_HRHV_DCON1	CPU3 data control register 1	39008_H	P0, 32	P0, SV, 32	Kernel Reset	258
CPU3_HRHV_SMACON	CPU3 SIST mode access control register	$3900C_H$	P0, 32	SFRSE, P0, SV, 32	Kernel Reset	234
CPU3_HRHV_DSTR	CPU3 HR specific data synchronous trap register	39010_H	P0, 32	P0, SV, 32	Kernel Reset	259
CPU3_HRHV_DATR	CPU3 HR specific data asynchronous trap register	39018_H	P0, 32	P0, SV, 32	Kernel Reset	261
CPU3_HRHV_DEADD	CPU3 HR specific data error address register	$3901C_H$	P0, 32	P0, SV, 32	Kernel Reset	262
CPU3_HRHV_DIEAR	CPU3 data integrity error address register	39020_H	P0, 32	P0, SV, 32	Kernel Reset	279
CPU3_HRHV_DIETR	CPU3 data integrity error trap register	39024_H	P0, 32	P0, SV, 32	Kernel Reset	280
CPU3_HRHV_DCON0	CPU3 data memory control register	39040_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	258
CPU3_HRHV_PSTR	CPU3 HR specific program synchronous trap register	39200_H	P0, 32	P0, SV, 32	Kernel Reset	295
CPU3_HRHV_PCON1	CPU3 program control 1	39204_H	P0, 32	P0, SV, 32	Kernel Reset	292
CPU3_HRHV_PCON2	CPU3 program control 2	39208_H	P0, 32	P0, SV, 32	Kernel Reset	561
CPU3_HRHV_PCON0	CPU3 program control 0	$3920C_H$	P0, 32	SFRE, P0, SV, 32	Kernel Reset	294
CPU3_HRHV_PIEAR	CPU3 HR specific program integrity error address register	39210_H	P0, 32	P0, SV, 32	Kernel Reset	281
CPU3_HRHV_PIETR	CPU3 HR specific program integrity error trap register	39214_H	P0, 32	P0, SV, 32	Kernel Reset	282

(table continues...)

Table 218 (continued) Register overview - CPU3 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU3_HRHV_COMPAT	CPU3 compatibility control register	39400_H	P0, 32	SFRSE, P0, SV, 32	Kernel Reset	235
CPU3_HRHV_FPU_TRAP_CON	CPU3 HR specific trap control register	$3A000_H$	P0, 32	P0, SV, 32	Kernel Reset	263
CPU3_HRHV_FPU_TRAP_PC	CPU3 HR specific trapping instruction program counter register	$3A004_H$	P0, 32	P0, SV, 32	Kernel Reset	265
CPU3_HRHV_FPU_TRAP_OPC	CPU3 HR specific trapping instruction opcode register	$3A008_H$	P0, 32	P0, SV, 32	Kernel Reset	266
CPU3_HRHV_FPU_TRAP_SRC1_L	CPU3 HR specific trapping instruction operand register	$3A010_H$	P0, 32	P0, SV, 32	Kernel Reset	267
CPU3_HRHV_FPU_TRAP_SRC1_U	CPU3 HR specific trapping instruction operand register	$3A014_H$	P0, 32	P0, SV, 32	Kernel Reset	268
CPU3_HRHV_FPU_TRAP_SRC2_L	CPU3 HR specific trapping instruction operand register	$3A018_H$	P0, 32	P0, SV, 32	Kernel Reset	268
CPU3_HRHV_FPU_TRAP_SRC2_U	CPU3 HR specific trapping instruction operand register	$3A01C_H$	P0, 32	P0, SV, 32	Kernel Reset	269
CPU3_HRHV_FPU_TRAP_SRC3_L	CPU3 HR specific trapping instruction operand register	$3A020_H$	P0, 32	P0, SV, 32	Kernel Reset	270
CPU3_HRHV_FPU_TRAP_SRC3_U	CPU3 HR specific trapping instruction operand register	$3A024_H$	P0, 32	P0, SV, 32	Kernel Reset	270
CPU3_HRHV_FPU_SYNC_TRAP_CON	CPU3 HR specific synchronous trap control register	$3A030_H$	P0, 32	P0, SV, 32	Kernel Reset	271
CPU3_HRHV_FPU_SYNC_TRAP_OPC	CPU3 HR specific trapping instruction opcode register	$3A034_H$	P0, 32	P0, SV, 32	Kernel Reset	273
CPU3_HRHV_VCON0	CPU3 virtualization control register 0	$3B000_H$	P0, 32	P0, SV, 32	Kernel Reset	275
CPU3_HRHV_VCON1	CPU3 virtualization control register 1	$3B004_H$	P0, 32	P0, SV, 32	Kernel Reset	276
CPU3_HRHV_VCON2	CPU3 virtualization control register 2	$3B008_H$	P0, 32	P0, SV, 32	Kernel Reset	276
CPU3_HRHV_BHV	CPU3 base hypervisor vector table pointer	$3B010_H$	P0, 32	P0, SV, 32, SFRE	Kernel Reset	559

(table continues...)

Table 218 (continued) Register overview - CPU3 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU3_HRHV_VMn_ICR (n=0-7)	CPU3 Virtual machine n interrupt state	$3B100_H$+n*4	P0, 32	P0, SV, 32	Kernel Reset	278
CPU3_HRHV_VMn_PETHRESH (n=0-7)	CPU3 Virtual machine n pre-emption threshold	$3B200_H$+n*4	P0, 32	P0, SV, 32	Kernel Reset	279
CPU3_HRHV_DPRy_L (y=0-23)	CPU3 HR specific data protection range y, lower bound register	$3C000_H$+y*8	P0, 32	P0, SV, 32	Kernel Reset	283
CPU3_HRHV_DPRy_U (y=0-23)	CPU3 HR specific data protection range y, upper bound register	$3C004_H$+y*8	P0, 32	P0, SV, 32	Kernel Reset	284
CPU3_HRHV_CPRy_L (y=0-15)	CPU3 HR specific code protection range y, lower bound register	$3D000_H$+y*8	P0, 32	P0, SV, 32	Kernel Reset	285
CPU3_HRHV_CPRy_U (y=0-15)	CPU3 HR specific code protection range y, upper bound register	$3D004_H$+y*8	P0, 32	P0, SV, 32	Kernel Reset	286
CPU3_HRHV_CPXE_y (y=0-3)	CPU3 HR specific code protection execute enable register set y	$3E000_H$+y*4	P0, 32	P0, SV, 32	Kernel Reset	287
CPU3_HRHV_DPRE_y (y=0-3)	CPU3 HR specific data protection read enable register set y	$3E010_H$+y*4	P0, 32	P0, SV, 32	Kernel Reset	288
CPU3_HRHV_DPWE_y (y=0-3)	CPU3 HR specific data protection write enable register set y	$3E020_H$+y*4	P0, 32	P0, SV, 32	Kernel Reset	289
CPU3_HRHV_CPXE_y (y=4-7)	CPU3 HR specific code protection execute enable register set y	$3E040_H$+(y-4)*4	P0, 32	P0, SV, 32	Kernel Reset	287
CPU3_HRHV_DPRE_y (y=4-7)	CPU3 HR specific data protection read enable register set y	$3E050_H$+(y-4)*4	P0, 32	P0, SV, 32	Kernel Reset	288
CPU3_HRHV_DPWE_y (y=4-7)	CPU3 HR specific data protection write enable register set y	$3E060_H$+(y-4)*4	P0, 32	P0, SV, 32	Kernel Reset	289
CPU3_HRHV_TPS_CON	CPU3 HR specific temporal protection system control register	$3E400_H$	P0, 32	P0, SV, 32	Kernel Reset	296
CPU3_HRHV_TPS_TIMERn (n=0-2)	CPU3 HR specific temporal protection system timer register n	$3E404_H$+n*4	P0, 32	P0, SV, 32	Kernel Reset	297

(table continues...)

Table 218 (continued) Register overview - CPU3 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU3_HRHV_TRnEVT (n=0-7)	CPU3 trigger event n	$3F000_H$+n*8	P0, 32	P0, SV, 32	Debug Reset	209
CPU3_HRHV_TRnADR (n=0-7)	CPU3 trigger address n	$3F004_H$+n*8	P0, 32	P0, SV, 32	Debug Reset	210
CPU3_HRHV_CCTRL	CPU3 HR specific counter control	$3FC00_H$	P0, 32	P0, SV, 32	Kernel Reset	226
CPU3_HRHV_CCNT	CPU3 HR specific CPU clock cycle count	$3FC04_H$	P0, 32	P0, SV, 32	Kernel Reset	227
CPU3_HRHV_ICNT	CPU3 HR specific instruction count	$3FC08_H$	P0, 32	P0, SV, 32	Kernel Reset	228
CPU3_HRHV_M1CNT	CPU3 HR specific multi-count register 1	$3FC0C_H$	P0, 32	P0, SV, 32	Kernel Reset	229
CPU3_HRHV_M2CNT	CPU3 HR specific multi-count register 2	$3FC10_H$	P0, 32	P0, SV, 32	Kernel Reset	230
CPU3_HRHV_M3CNT	CPU3 HR specific multi-count register 3	$3FC14_H$	P0, 32	P0, SV, 32	Kernel Reset	231
CPU3_HRHV_DBGSR	CPU3 debug status register	$3FD00_H$	P0, 32	P0, SV, 32	Debug Reset	211
CPU3_HRHV_EXEVT	CPU3 external event register	$3FD08_H$	P0, 32	P0, SV, 32	Debug Reset	212
CPU3_HRHV_CREVT	CPU3 core register access event	$3FD0C_H$	P0, 32	P0, SV, 32	Debug Reset	213
CPU3_HRHV_SWEVT	CPU3 software debug event	$3FD10_H$	P0, 32	P0, SV, 32	Debug Reset	214
CPU3_HRHV_DBGACT	CPU3 debug action configuration	$3FD14_H$	P0, 32	P0, SV, 32	Debug Reset	215
CPU3_HRHV_TRIG_ACC	CPU3 trigger event accumulator	$3FD30_H$	P0, 32	P0, SV, 32	Debug Reset	217
CPU3_HRHV_DMS	CPU3 debug monitor start address	$3FD40_H$	P0, 32	P0, SV, 32	Debug Reset	534
CPU3_HRHV_DCX	CPU3 debug context save area pointer	$3FD44_H$	P0, 32	P0, SV, 32	Debug Reset	536
CPU3_HRHV_DBGTCR	CPU3 debug trap control	$3FD48_H$	P0, 32	P0, SV, 32	Application Reset	219
CPU3_HRHV_DBGCFG	CPU3 debug configuration	$3FD4C_H$	P0, 32	P0, SV, 32	Debug Reset	220
CPU3_HRHV_TRCCFG	CPU3 trace configuration	$3FD50_H$	P0, 32	P0, SV, 32	Debug Reset	221
CPU3_HRHV_TRCFILT	CPU3 trace filter	$3FD54_H$	P0, 32	P0, SV, 32	Debug Reset	222

(table continues...)

Table 218 (continued) Register overview - CPU3 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU3_HRHV_TRCLIM	CPU3 trace bandwidth limiter	$3FD58_H$	P0, 32	P0, SV, 32	Debug Reset	224
CPU3_HRHV_TS16PTCCTRL	CPU3 MCDS performance counter control register	$3FD60_H$	P0, 32	P0, SV, 32	Debug Reset	225
CPU3_HRHV_PCXI	CPU3 previous context information register	$3FE00_H$	P0, 32	P0, SV, 32	Kernel Reset	236
CPU3_HRHV_PSW	CPU3 HR specific program status word	$3FE04_H$	P0, 32	P0, SV, 32	Kernel Reset	237
CPU3_HRHV_PC	CPU3 program counter	$3FE08_H$	P0, 32	P0, SV, 32	Kernel Reset	538
CPU3_HRHV_CORECON	CPU3 HR specific core configuration register	$3FE14_H$	P0, 32	SFRSE, P0, SV, 32	Kernel Reset	250
CPU3_HRHV_CPU_ID	CPU3 identification register	$3FE18_H$	P0, 32	P0, SV, 32	Kernel Reset	540
CPU3_HRHV_CORE_ID	CPU3 core identification register	$3FE1C_H$	P0, 32	P0, SV, 32	Kernel Reset	542
CPU3_HRHV_BIV	CPU3 HR specific base interrupt vector table pointer	$3FE20_H$	P0, 32	SFRE, P0, SV, 32	Kernel Reset	251
CPU3_HRHV_BTV	CPU3 HR specific base trap vector table pointer	$3FE24_H$	P0, 32	SFRE, P0, SV, 32	Kernel Reset	555
CPU3_HRHV_ISP	CPU3 HR specific interrupt stack pointer	$3FE28_H$	P0, 32	SFRE, P0, SV, 32	Kernel Reset	253
CPU3_HRHV_ICR	CPU3 interrupt control register	$3FE2C_H$	P0, 32	P0, SV, 32	Kernel Reset	242
CPU3_HRHV_PPRS	CPU3 previous PRS	$3FE34_H$	P0, 32	P0, SV, 32	Kernel Reset	244
CPU3_HRHV_FCX	CPU3 HR specific free CSA list head pointer	$3FE38_H$	P0, 32	P0, SV, 32	Kernel Reset	254
CPU3_HRHV_LCX	CPU3 HR specific free CSA list limit pointer	$3FE3C_H$	P0, 32	P0, SV, 32	Kernel Reset	255
CPU3_HRHV_SWID	CPU3 software identification	$3FE40_H$	P0, 32	P0, SV, 32	Kernel Reset	544
CPU3_HRHV_CUS_ID	CPU3 customer ID register	$3FE50_H$	P0, 32	P0, SV, 32	Kernel Reset	546
CPU3_HRHV_BOOTCON	CPU3 core boot configuration register	$3FE60_H$	P0, 32	P0, SV, 32	Kernel Reset	548
CPU3_HRHV_LCLCON	CPU3 lockstep control register	$3FE64_H$	P0, 32	STP, SFRSE, P0, SV, 32	Cold PowerOn Reset	550
CPU3_HRHV_CCON	CPU3 clock control register	$3FE68_H$	P0, 32	P0, SV, 32	Kernel Reset	248
CPU3_HRHV_TCON	CPU3 hardware configuration	$3FE6C_H$	P0, 32	P0, SV, 32	Kernel Reset	552

(table continues...)

Table 218 (continued) Register overview - CPU3 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU3_HRHV_Ay (y=0-1)	CPU3 HR specific address general purpose register y	$3FF80_H$+ y*4	P0, 32	P0, SV, 32	Kernel Reset	274
CPU3_HRHV_Ay (y=8-9)	CPU3 HR specific address general purpose register y	$3FFA0_H$+ (y-8)*4	P0, 32	P0, SV, 32	Kernel Reset	274

4.10.4.1.9 Register overview - CPU4 domain SRI (ascending offset address)

Table 219 Register overview - CPU4 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU4_STM_CLC	CPU4 system timer clock control register	00000_H	P0, 32	P0, SV, SFRE, 32	System Reset	351
CPU4_STMHV_CLC	CPU4 system timer clock control register, hypervisor view	00000_H	P0, 32	P0, SV, SFRE, 32	System Reset	351
CPU4_STM_OCS	CPU4 OCDS control and status register	00004_H	P0, 32	P0, SV, 32, DEN	Debug Reset	352
CPU4_STMHV_OCS	CPU4 OCDS control and status register, hypervisor view	00004_H	P0, 32	P0, SV, 32, DEN	Debug Reset	352
CPU4_STM_ABS	CPU4 system timer absolute timer	00020_H	(P0, 32)| (P0, 64)	P0, SV, 64, SFRSE	System Reset	353
CPU4_STMHV_ABS	CPU4 system timer absolute timer, hypervisor view	00020_H	(P0, 32)| (P0, 64)	P0, SV, 64, SFRSE	System Reset	353
CPU4_STMHV_VMn_CMPc (c=0-1;n=0-7)	CPU4 VMn system timer compare register c, hypervisor view	00100_H+ n*20_H+c *4	P1, 32	P1, SV, 32	System Reset	354
CPU4_STMHV_VMn_CMCON (n=0-7)	CPU4 VMn system timer compare match control register, hypervisor view	00108_H+ n*20_H	P1, 32	P1, SV, 32	System Reset	355
CPU4_STMHV_VMn_ICR (n=0-7)	CPU4 VMn system timer interrupt control register, hypervisor view	$0010C_H$+ n*20_H	P1, 32	P1, SV, 32	System Reset	356
CPU4_STMHV_VMn_ISCR (n=0-7)	CPU4 VMn system timer interrupt set/clear register, hypervisor view	00110_H+ n*20_H	P1, 32	P1, SV, 32	System Reset	357
CPU4_STMHV_VMn_ISR (n=0-7)	CPU4 VMn system timer interrupt status register, hypervisor view	00114_H+ n*20_H	P1, 32	P1, SV, 32	System Reset	358

(table continues...)

Table 219 (continued) Register overview - CPU4 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU4_STMHV_VMn_RELTIM (n=0-7)	CPU4 VMn system timer relative timer, hypervisor view	00118_H+ n*20_H	(P1, 32)| (P1, 64)	P1, SV, 64	System Reset	359
CPU4_STM_CMPc (c=0-1)	CPU4 system timer compare register c	00120_H+ c*4	P1, 32	P1, SV, 32	System Reset	354
CPU4_STM_CMCON	CPU4 system timer compare match control register	00128_H	P1, 32	P1, SV, 32	System Reset	355
CPU4_STM_ICR	CPU4 system timer interrupt control register	$0012C_H$	P1, 32	P1, SV, 32	System Reset	356
CPU4_STM_ISCR	CPU4 system timer interrupt set/clear register	00130_H	P1, 32	P1, SV, 32	System Reset	357
CPU4_STM_ISR	CPU4 system timer interrupt status register	00134_H	P1, 32	P1, SV, 32	System Reset	358
CPU4_STM_RELTIM	CPU4 system timer relative timer	00138_H	(P1, 32)| (P1, 64)	P1, SV, 64	System Reset	359
CPU4_KRST0	CPU4 reset register 0	$0D000_H$	P0, 32	RSTE, P0, SV, 32	Kernel Reset	297
CPU4_KRST1	CPU4 reset register 1	$0D004_H$	P0, 32	RSTE, P0, SV, 32	Kernel Reset	298
CPU4_KRSTCLR	CPU4 reset clear register	$0D008_H$	P0, 32	RSTE, P0, SV, 32	Kernel Reset	299
CPU4_PROTSPRSE	CPU4 PROTSPR register safe endinit	$0E008_H$	32	SV, 32, PROT	Application Reset	562
CPU4_ACCENSPRCFG_WRA	CPU4 shadow SPR, write access enable register A	$0E020_H$	32	SPRSE, SV, 32	Application Reset	565
CPU4_ACCENSPRCFG_WRB	CPU4 shadow SPR, write access enable register B	$0E024_H$	32	SPRSE, SV, 32	Application Reset	566
CPU4_ACCENSPRCFG_RDA	CPU4 shadow SPR, read access enable register A	$0E028_H$	32	SPRSE, SV, 32	Application Reset	567
CPU4_ACCENSPRCFG_RDB	CPU4 shadow SPR, read access enable register B	$0E02C_H$	32	SPRSE, SV, 32	Application Reset	568
CPU4_ACCENSPRCFG_VM	CPU4 shadow SPR, VM access enable register	$0E030_H$	32	SPRSE, SV, 32	Application Reset	310
CPU4_ACCENSPRCFG_PRS	CPU4 shadow SPR, PRS access enable register	$0E034_H$	32	SPRSE, SV, 32	Application Reset	310
CPU4_ACCENSPRCFG_RGNLA	CPU4 shadow SPR, region lower address register	$0E038_H$	32	SPRSE, SV, 32	Application Reset	311
CPU4_ACCENSPRCFG_RGNUA	CPU4 shadow SPR, region upper address register	$0E03C_H$	32	SPRSE, SV, 32	Application Reset	312

(table continues...)

Table 219 (continued) Register overview - CPU4 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU4_PROTDLMUSE	CPU4 PROTDLMU register safe endinit	$0E048_H$	32	SV, 32, PROT	Application Reset	312
CPU4_ACCENDLMUCFG_WRA	CPU4 shadow DLMU, write access enable register A	$0E060_H$	32	DLMUSE, SV, 32	Application Reset	314
CPU4_ACCENDLMUCFG_WRB	CPU4 shadow DLMU, write access enable register B	$0E064_H$	32	DLMUSE, SV, 32	Application Reset	315
CPU4_ACCENDLMUCFG_RDA	CPU4 shadow DLMU, read access enable register A	$0E068_H$	32	DLMUSE, SV, 32	Application Reset	315
CPU4_ACCENDLMUCFG_RDB	CPU4 shadow DLMU, read access enable register B	$0E06C_H$	32	DLMUSE, SV, 32	Application Reset	316
CPU4_ACCENDLMUCFG_VM	CPU4 shadow DLMU, VM access enable register	$0E070_H$	32	DLMUSE, SV, 32	Application Reset	316
CPU4_ACCENDLMUCFG_PRS	CPU4 shadow DLMU, PRS access enable register	$0E074_H$	32	DLMUSE, SV, 32	Application Reset	317
CPU4_ACCENDLMUCFG_RGNLA	CPU4 shadow DLMU, region lower address register	$0E078_H$	32	DLMUSE, SV, 32	Application Reset	317
CPU4_ACCENDLMUCFG_RGNUA	CPU4 shadow DLMU, region upper address register	$0E07C_H$	32	DLMUSE, SV, 32	Application Reset	318
CPU4_PROTSFRE	CPU4 PROTSFR register endinit	$0E080_H$	32	SV, 32, PROT	Application Reset	569
CPU4_PROTSFRSE	CPU4 PROTSFR register safe endinit	$0E088_H$	32	SV, 32, PROT	Application Reset	571
CPU4_ACCENSFRCFG_WRA	CPU4 shadow SFR, write access enable register A	$0E0A0_H$	32	SFRSE, SV, 32	Application Reset	573
CPU4_ACCENSFRCFG_WRB	CPU4 shadow SFR, write access enable register B	$0E0A4_H$	32	SFRSE, SV, 32	Application Reset	574
CPU4_ACCENSFRCFG_RDA	CPU4 shadow SFR, read access enable register A	$0E0A8_H$	32	SFRSE, SV, 32	Application Reset	575
CPU4_ACCENSFRCFG_RDB	CPU4 shadow SFR, read access enable register B	$0E0AC_H$	32	SFRSE, SV, 32	Application Reset	576
CPU4_ACCENSFRCFG_VM	CPU4 shadow SFR, VM access enable register	$0E0B0_H$	32	SFRSE, SV, 32	Application Reset	325
CPU4_ACCENSFRCFG_PRS	CPU4 shadow SFR, PRS access enable register	$0E0B4_H$	32	SFRSE, SV, 32	Application Reset	325
CPU4_PROTRSTE	CPU4 PROTRST register endinit	$0E0C0_H$	32	SV, 32, PROT	Application Reset	577
CPU4_PROTSTMSE	CPU4 PROTSTM Register Safe Endinit	$0E0D8_H$	32	SV, 32, PROT	Application Reset	579
CPU4_ACCENSTMCFG_WRA	CPU4 shadow STM, write access enable register A	$0E0E0_H$	32	STMSE, SV, 32	Application Reset	582
CPU4_ACCENSTMCFG_WRB	CPU4 shadow STM, write access enable register B	$0E0E4_H$	32	STMSE, SV, 32	Application Reset	330

(table continues...)

Table 219 (continued) Register overview - CPU4 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU4_ACCENSTMCFG_RDA	CPU4 shadow STM, read access enable register A	$0E0E8_H$	32	STMSE, SV, 32	Application Reset	583
CPU4_ACCENSTMCFG_RDB	CPU4 shadow STM, read access enable register B	$0E0EC_H$	32	STMSE, SV, 32	Application Reset	584
CPU4_ACCENSTMCFG_VM	CPU4 shadow STM, VM access enable register	$0E0F0_H$	32	STMSE, SV, 32	Application Reset	332
CPU4_ACCENSTMCFG_PRS	CPU4 shadow STM, PRS access enable register	$0E0F4_H$	32	STMSE, SV, 32	Application Reset	333
CPU4_ACCENSPRy_WRA (y=0-15)	CPU4 SPR range y, write access enable register A	$0E200_H$+ y*20_H	32	SPRSE, SV, 32	Application Reset	585
CPU4_ACCENSPRy_WRB (y=0-15)	CPU4 SPR range y, write access enable register B	$0E204_H$+ y*20_H	32	SPRSE, SV, 32	Application Reset	586
CPU4_ACCENSPRy_RDA (y=0-15)	CPU4 SPR range y, read access enable register A	$0E208_H$+ y*20_H	32	SPRSE, SV, 32	Application Reset	587
CPU4_ACCENSPRy_RDB (y=0-15)	CPU4 SPR range y, read access enable register B	$0E20C_H$+ y*20_H	32	SPRSE, SV, 32	Application Reset	588
CPU4_ACCENSPRy_VM (y=0-15)	CPU4 SPR range y, VM access enable register	$0E210_H$+ y*20_H	32	SPRSE, SV, 32	Application Reset	336
CPU4_ACCENSPRy_PRS (y=0-15)	CPU4 SPR range y, PRS access enable register	$0E214_H$+ y*20_H	32	SPRSE, SV, 32	Application Reset	337
CPU4_ACCENSPRy_RGNLA (y=0-15)	CPU4 SPR range y, region lower address register	$0E218_H$+ y*20_H	32	SPRSE, SV, 32	Application Reset	337
CPU4_ACCENSPRy_RGNUA (y=0-15)	CPU4 SPR range y, region upper address register	$0E21C_H$+ y*20_H	32	SPRSE, SV, 32	Application Reset	338
CPU4_ACCENDLMUy_WRA (y=0-15)	CPU4 DLMU range y, write access enable register A	$0E400_H$+ y*20_H	32	DLMUSE, SV, 32	Application Reset	338
CPU4_ACCENDLMUy_WRB (y=0-15)	CPU4 DLMU range y, write access enable register B	$0E404_H$+ y*20_H	32	DLMUSE, SV, 32	Application Reset	339
CPU4_ACCENDLMUy_RDA (y=0-15)	CPU4 DLMU range y, read access enable register A	$0E408_H$+ y*20_H	32	DLMUSE, SV, 32	Application Reset	339

(table continues...)

Table 219 (continued) Register overview - CPU4 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU4_ACCENDLMUy_RDB (y=0-15)	CPU4 DLMU range y, read access enable register B	$0E40C_H$+ y*20_H	32	DLMUSE, SV, 32	Application Reset	340
CPU4_ACCENDLMUy_VM (y=0-15)	CPU4 DLMU range y, VM access enable register	$0E410_H$+ y*20_H	32	DLMUSE, SV, 32	Application Reset	340
CPU4_ACCENDLMUy_PRS (y=0-15)	CPU4 DLMU range y, PRS access enable register	$0E414_H$+ y*20_H	32	DLMUSE, SV, 32	Application Reset	341
CPU4_ACCENDLMUy_RGNLA (y=0-15)	CPU4 DLMU range y, region lower address register	$0E418_H$+ y*20_H	32	DLMUSE, SV, 32	Application Reset	342
CPU4_ACCENDLMUy_RGNUA (y=0-15)	CPU4 DLMU range y, region upper address register	$0E41C_H$+ y*20_H	32	DLMUSE, SV, 32	Application Reset	342
CPU4_ACCENSFR_WRA	CPU4 SFR, write access enable register A	$0E600_H$	32	SFRSE, SV, 32	Application Reset	589
CPU4_ACCENSFR_WRB	CPU4 SFR, write access enable register B	$0E604_H$	32	SFRSE, SV, 32	Application Reset	590
CPU4_ACCENSFR_RDA	CPU4 SFR, read access enable register A	$0E608_H$	32	SFRSE, SV, 32	Application Reset	591
CPU4_ACCENSFR_RDB	CPU4 SFR, read access enable register B	$0E60C_H$	32	SFRSE, SV, 32	Application Reset	592
CPU4_ACCENSFR_VM	CPU4 SFR, VM access enable register	$0E610_H$	32	SFRSE, SV, 32	Application Reset	345
CPU4_ACCENSFR_PRS	CPU4 SFR, PRS access enable register	$0E614_H$	32	SFRSE, SV, 32	Application Reset	346
CPU4_ACCENSTMVMy_WRA (y=0-7)	CPU4 VMy STM, write access enable register A, virtualization view	$0E700_H$+ y*20_H	32	STMSE, SV, 32	Application Reset	593
CPU4_ACCENSTMVMy_WRB (y=0-7)	CPU4 VMy STM, write access enable register B, virtualization view	$0E704_H$+ y*20_H	32	STMSE, SV, 32	Application Reset	347
CPU4_ACCENSTMVMy_RDA (y=0-7)	CPU4 VMy STM, read access enable register A, virtualization view	$0E708_H$+ y*20_H	32	STMSE, SV, 32	Application Reset	594
CPU4_ACCENSTMVMy_RDB (y=0-7)	CPU4 VMy STM, read access enable register B, virtualization view	$0E70C_H$+ y*20_H	32	STMSE, SV, 32	Application Reset	595
CPU4_ACCENSTMVMy_VM (y=0-7)	CPU4 VMy STM, VM access enable register, virtualization view	$0E710_H$+ y*20_H	32	STMSE, SV, 32	Application Reset	350

(table continues...)

Table 219 (continued) Register overview - CPU4 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU4_ACCENSTMVMy_PRS (y=0-7)	CPU4 VMy STM, PRS access enable register, virtualization view	$0E714_H$+ y*20_H	32	STMSE, SV, 32	Application Reset	350
CPU4_OSEL	CPU4 overlay range select register	$0FB00_H$	P0, 32	P0, SV, 32	Kernel Reset	299
CPU4_RABRi (i=0-31)	CPU4 redirected address base register i	$0FB10_H$+ i*12	P0, 32	P0, SV, 32	Kernel Reset	300
CPU4_OTARi (i=0-31)	CPU4 overlay target address register i	$0FB14_H$+ i*12	P0, 32	P0, SV, 32	Kernel Reset	302
CPU4_OMASKi (i=0-31)	CPU4 overlay mask register i	$0FB18_H$+ i*12	P0, 32	P0, SV, 32	Kernel Reset	303
CPU4_HRA_SEGEN	CPU4 SRI error generation register	11030_H	P0, 32	SFRE, P0, SV, 32	Application Reset	255
CPU4_HRA_LCLTEST	CPU4 lockstep test register	11040_H	P0, 32	P0, SV, 32	Cold PowerOn Reset	256
CPU4_HRA_PMA0	CPU4 data access cacheability register	18100_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	232
CPU4_HRA_PMA1	CPU4 code access cacheability register	18104_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	233
CPU4_HRA_PMA2	CPU4 peripheral space identifier register	18108_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	234
CPU4_HRA_DCON2	CPU4 data control register 2	19000_H	P0, 32	P0, SV, 32	Kernel Reset	558
CPU4_HRA_DCON1	CPU4 data control register 1	19008_H	P0, 32	P0, SV, 32	Kernel Reset	258
CPU4_HRA_SMACON	CPU4 SIST mode access control register	$1900C_H$	P0, 32	SFRSE, P0, SV, 32	Kernel Reset	234
CPU4_HRA_DSTR	CPU4 HR specific data synchronous trap register	19010_H	P0, 32	P0, SV, 32	Kernel Reset	259
CPU4_HRA_DATR	CPU4 HR specific data asynchronous trap register	19018_H	P0, 32	P0, SV, 32	Kernel Reset	261
CPU4_HRA_DEADD	CPU4 HR specific data error address register	$1901C_H$	P0, 32	P0, SV, 32	Kernel Reset	262
CPU4_HRA_DIEAR	CPU4 data integrity error address register	19020_H	P0, 32	P0, SV, 32	Kernel Reset	279
CPU4_HRA_DIETR	CPU4 data integrity error trap register	19024_H	P0, 32	P0, SV, 32	Kernel Reset	280
CPU4_HRA_DCON0	CPU4 data memory control register	19040_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	258
CPU4_HRA_PSTR	CPU4 HR specific program synchronous trap register	19200_H	P0, 32	P0, SV, 32	Kernel Reset	295

(table continues...)

Table 219 (continued) Register overview - CPU4 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU4_HRA_PCON1	CPU4 program control 1	19204_H	P0, 32	P0, SV, 32	Kernel Reset	292
CPU4_HRA_PCON2	CPU4 program control 2	19208_H	P0, 32	P0, SV, 32	Kernel Reset	561
CPU4_HRA_PCON0	CPU4 program control 0	$1920C_H$	P0, 32	SFRE, P0, SV, 32	Kernel Reset	294
CPU4_HRA_PIEAR	CPU4 HR specific program integrity error address register	19210_H	P0, 32	P0, SV, 32	Kernel Reset	281
CPU4_HRA_PIETR	CPU4 HR specific program integrity error trap register	19214_H	P0, 32	P0, SV, 32	Kernel Reset	282
CPU4_HRA_COMPAT	CPU4 compatibility control register	19400_H	P0, 32	SFRSE, P0, SV, 32	Kernel Reset	235
CPU4_HRA_FPU_TRAP_CON	CPU4 HR specific trap control register	$1A000_H$	P0, 32	P0, SV, 32	Kernel Reset	263
CPU4_HRA_FPU_TRAP_PC	CPU4 HR specific trapping instruction program counter register	$1A004_H$	P0, 32	P0, SV, 32	Kernel Reset	265
CPU4_HRA_FPU_TRAP_OPC	CPU4 HR specific trapping instruction opcode register	$1A008_H$	P0, 32	P0, SV, 32	Kernel Reset	266
CPU4_HRA_FPU_TRAP_SRC1_L	CPU4 HR specific trapping instruction operand register	$1A010_H$	P0, 32	P0, SV, 32	Kernel Reset	267
CPU4_HRA_FPU_TRAP_SRC1_U	CPU4 HR specific trapping instruction operand register	$1A014_H$	P0, 32	P0, SV, 32	Kernel Reset	268
CPU4_HRA_FPU_TRAP_SRC2_L	CPU4 HR specific trapping instruction operand register	$1A018_H$	P0, 32	P0, SV, 32	Kernel Reset	268
CPU4_HRA_FPU_TRAP_SRC2_U	CPU4 HR specific trapping instruction operand register	$1A01C_H$	P0, 32	P0, SV, 32	Kernel Reset	269
CPU4_HRA_FPU_TRAP_SRC3_L	CPU4 HR specific trapping instruction operand register	$1A020_H$	P0, 32	P0, SV, 32	Kernel Reset	270
CPU4_HRA_FPU_TRAP_SRC3_U	CPU4 HR specific trapping instruction operand register	$1A024_H$	P0, 32	P0, SV, 32	Kernel Reset	270
CPU4_HRA_FPU_SYNC_TRAP_CON	CPU4 HR specific synchronous trap control register	$1A030_H$	P0, 32	P0, SV, 32	Kernel Reset	271
CPU4_HRA_FPU_SYNC_TRAP_OPC	CPU4 HR specific trapping instruction opcode register	$1A034_H$	P0, 32	P0, SV, 32	Kernel Reset	273

(table continues...)

Table 219 (continued) Register overview - CPU4 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU4_HRA_DPRy_L (y=0-23)	CPU4 HR specific data protection range y, lower bound register	$1C000_H$+y*8	P0, 32	P0, SV, 32	Kernel Reset	283
CPU4_HRA_DPRy_U (y=0-23)	CPU4 HR specific data protection range y, upper bound register	$1C004_H$+y*8	P0, 32	P0, SV, 32	Kernel Reset	284
CPU4_HRA_CPRy_L (y=0-15)	CPU4 HR specific code protection range y, lower bound register	$1D000_H$+y*8	P0, 32	P0, SV, 32	Kernel Reset	285
CPU4_HRA_CPRy_U (y=0-15)	CPU4 HR specific code protection range y, upper bound register	$1D004_H$+y*8	P0, 32	P0, SV, 32	Kernel Reset	286
CPU4_HRA_CPXE_y (y=0-3)	CPU4 HR specific code protection execute enable register set y	$1E000_H$+y*4	P0, 32	P0, SV, 32	Kernel Reset	287
CPU4_HRA_DPRE_y (y=0-3)	CPU4 HR specific data protection read enable register set y	$1E010_H$+y*4	P0, 32	P0, SV, 32	Kernel Reset	288
CPU4_HRA_DPWE_y (y=0-3)	CPU4 HR specific data protection write enable register set y	$1E020_H$+y*4	P0, 32	P0, SV, 32	Kernel Reset	289
CPU4_HRA_CPXE_y (y=4-7)	CPU4 HR specific code protection execute enable register set y	$1E040_H$+(y-4)*4	P0, 32	P0, SV, 32	Kernel Reset	287
CPU4_HRA_DPRE_y (y=4-7)	CPU4 HR specific data protection read enable register set y	$1E050_H$+(y-4)*4	P0, 32	P0, SV, 32	Kernel Reset	288
CPU4_HRA_DPWE_y (y=4-7)	CPU4 HR specific data protection write enable register set y	$1E060_H$+(y-4)*4	P0, 32	P0, SV, 32	Kernel Reset	289
CPU4_HRA_TPS_CON	CPU4 HR specific temporal protection system control register	$1E400_H$	P0, 32	P0, SV, 32	Kernel Reset	296
CPU4_HRA_TPS_TIMERn (n=0-2)	CPU4 HR specific temporal protection system timer register n	$1E404_H$+n*4	P0, 32	P0, SV, 32	Kernel Reset	297
CPU4_HRA_TRnEVT (n=0-7)	CPU4 trigger event n	$1F000_H$+n*8	P0, 32	P0, SV, 32	Debug Reset	209
CPU4_HRA_TRnADR (n=0-7)	CPU4 trigger address n	$1F004_H$+n*8	P0, 32	P0, SV, 32	Debug Reset	210

(table continues...)

Table 219 (continued) Register overview - CPU4 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU4_HRA_CCTRL	CPU4 HR specific counter control	$1FC00_H$	P0, 32	P0, SV, 32	Kernel Reset	226
CPU4_HRA_CCNT	CPU4 HR specific CPU clock cycle count	$1FC04_H$	P0, 32	P0, SV, 32	Kernel Reset	227
CPU4_HRA_ICNT	CPU4 HR specific instruction count	$1FC08_H$	P0, 32	P0, SV, 32	Kernel Reset	228
CPU4_HRA_M1CNT	CPU4 HR specific multi-count register 1	$1FC0C_H$	P0, 32	P0, SV, 32	Kernel Reset	229
CPU4_HRA_M2CNT	CPU4 HR specific multi-count register 2	$1FC10_H$	P0, 32	P0, SV, 32	Kernel Reset	230
CPU4_HRA_M3CNT	CPU4 HR specific multi-count register 3	$1FC14_H$	P0, 32	P0, SV, 32	Kernel Reset	231
CPU4_HRA_DBGSR	CPU4 debug status register	$1FD00_H$	P0, 32	P0, SV, 32	Debug Reset	211
CPU4_HRA_EXEVT	CPU4 external event register	$1FD08_H$	P0, 32	P0, SV, 32	Debug Reset	212
CPU4_HRA_CREVT	CPU4 core register access event	$1FD0C_H$	P0, 32	P0, SV, 32	Debug Reset	213
CPU4_HRA_SWEVT	CPU4 software debug event	$1FD10_H$	P0, 32	P0, SV, 32	Debug Reset	214
CPU4_HRA_DBGACT	CPU4 debug action configuration	$1FD14_H$	P0, 32	P0, SV, 32	Debug Reset	215
CPU4_HRA_TRIG_ACC	CPU4 trigger event accumulator	$1FD30_H$	P0, 32	P0, SV, 32	Debug Reset	217
CPU4_HRA_DMS	CPU4 debug monitor start address	$1FD40_H$	P0, 32	P0, SV, 32	Debug Reset	534
CPU4_HRA_DCX	CPU4 debug context save area pointer	$1FD44_H$	P0, 32	P0, SV, 32	Debug Reset	536
CPU4_HRA_DBGTCR	CPU4 debug trap control	$1FD48_H$	P0, 32	P0, SV, 32	Application Reset	219
CPU4_HRA_DBGCFG	CPU4 debug configuration	$1FD4C_H$	P0, 32	P0, SV, 32	Debug Reset	220
CPU4_HRA_TRCCFG	CPU4 trace configuration	$1FD50_H$	P0, 32	P0, SV, 32	Debug Reset	221
CPU4_HRA_TRCFILT	CPU4 trace filter	$1FD54_H$	P0, 32	P0, SV, 32	Debug Reset	222
CPU4_HRA_TRCLIM	CPU4 trace bandwidth limiter	$1FD58_H$	P0, 32	P0, SV, 32	Debug Reset	224
CPU4_HRA_TS16PTCCTRL	CPU4 MCDS performance counter control register	$1FD60_H$	P0, 32	P0, SV, 32	Debug Reset	225
CPU4_HRA_PCXI	CPU4 previous context information register	$1FE00_H$	P0, 32	P0, SV, 32	Kernel Reset	236

(table continues...)

Table 219 (continued) Register overview - CPU4 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU4_HRA_PSW	CPU4 HR specific program status word	$1FE04_H$	P0, 32	P0, SV, 32	Kernel Reset	237
CPU4_HRA_PC	CPU4 program counter	$1FE08_H$	P0, 32	P0, SV, 32	Kernel Reset	538
CPU4_HRA_CORECON	CPU4 HR specific core configuration register	$1FE14_H$	P0, 32	SFRSE, P0, SV, 32	Kernel Reset	250
CPU4_HRA_CPU_ID	CPU4 identification register	$1FE18_H$	P0, 32	P0, SV, 32	Kernel Reset	540
CPU4_HRA_CORE_ID	CPU4 core identification register	$1FE1C_H$	P0, 32	P0, SV, 32	Kernel Reset	542
CPU4_HRA_BIV	CPU4 HR specific base interrupt vector table pointer	$1FE20_H$	P0, 32	SFRE, P0, SV, 32	Kernel Reset	251
CPU4_HRA_BTV	CPU4 HR specific base trap vector table pointer	$1FE24_H$	P0, 32	SFRE, P0, SV, 32	Kernel Reset	554
CPU4_HRA_ISP	CPU4 HR specific interrupt stack pointer	$1FE28_H$	P0, 32	SFRE, P0, SV, 32	Kernel Reset	253
CPU4_HRA_ICR	CPU4 interrupt control register	$1FE2C_H$	P0, 32	P0, SV, 32	Kernel Reset	242
CPU4_HRA_PPRS	CPU4 previous PRS	$1FE34_H$	P0, 32	P0, SV, 32	Kernel Reset	244
CPU4_HRA_FCX	CPU4 HR specific free CSA list head pointer	$1FE38_H$	P0, 32	P0, SV, 32	Kernel Reset	254
CPU4_HRA_LCX	CPU4 HR specific free CSA list limit pointer	$1FE3C_H$	P0, 32	P0, SV, 32	Kernel Reset	255
CPU4_HRA_SWID	CPU4 software identification	$1FE40_H$	P0, 32	P0, SV, 32	Kernel Reset	544
CPU4_HRA_CUS_ID	CPU4 customer ID register	$1FE50_H$	P0, 32	P0, SV, 32	Kernel Reset	546
CPU4_HRA_BOOTCON	CPU4 core boot configuration register	$1FE60_H$	P0, 32	P0, SV, 32	Kernel Reset	548
CPU4_HRA_LCLCON	CPU4 lockstep control register	$1FE64_H$	P0, 32	STP, SFRSE, P0, SV, 32	Cold PowerOn Reset	550
CPU4_HRA_CCON	CPU4 clock control register	$1FE68_H$	P0, 32	P0, SV, 32	Kernel Reset	248
CPU4_HRA_TCCON	CPU4 hardware configuration	$1FE6C_H$	P0, 32	P0, SV, 32	Kernel Reset	552
CPU4_Dy (y=0-15)	CPU4 data general purpose register y	$1FF00_H$+ y*4	P0, 32	P0, SV, 32	Kernel Reset	274
CPU4_Ay (y=0-15)	CPU4 address general purpose register y	$1FF80_H$+ y*4	P0, 32	P0, SV, 32	Kernel Reset	274

(table continues...)

Table 219 (continued) Register overview - CPU4 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU4_HRB_SEGEN	CPU4 SRI error generation register	21030_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	255
CPU4_HRB_LCLTEST	CPU4 lockstep test register	21040_H	P0, 32	P0, SV, 32	Cold PowerOn Reset	256
CPU4_HRB_PMA0	CPU4 data access cacheability register	28100_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	232
CPU4_HRB_PMA1	CPU4 code access cacheability register	28104_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	233
CPU4_HRB_PMA2	CPU4 peripheral space identifier register	28108_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	234
CPU4_HRB_DCON2	CPU4 data control register 2	29000_H	P0, 32	P0, SV, 32	Kernel Reset	558
CPU4_HRB_DCON1	CPU4 data control register 1	29008_H	P0, 32	P0, SV, 32	Kernel Reset	258
CPU4_HRB_SMACON	CPU4 SIST mode access control register	$2900C_H$	P0, 32	SFRSE, P0, SV, 32	Kernel Reset	234
CPU4_HRB_DSTR	CPU4 HR specific data synchronous trap register	29010_H	P0, 32	P0, SV, 32	Kernel Reset	259
CPU4_HRB_DATR	CPU4 HR specific data asynchronous trap register	29018_H	P0, 32	P0, SV, 32	Kernel Reset	261
CPU4_HRB_DEADD	CPU4 HR specific data error address register	$2901C_H$	P0, 32	P0, SV, 32	Kernel Reset	262
CPU4_HRB_DIEAR	CPU4 data integrity error address register	29020_H	P0, 32	P0, SV, 32	Kernel Reset	279
CPU4_HRB_DIETR	CPU4 data integrity error trap register	29024_H	P0, 32	P0, SV, 32	Kernel Reset	280
CPU4_HRB_DCON0	CPU4 data memory control register	29040_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	258
CPU4_HRB_PSTR	CPU4 HR specific program synchronous trap register	29200_H	P0, 32	P0, SV, 32	Kernel Reset	295
CPU4_HRB_PCON1	CPU4 program control 1	29204_H	P0, 32	P0, SV, 32	Kernel Reset	292
CPU4_HRB_PCON2	CPU4 program control 2	29208_H	P0, 32	P0, SV, 32	Kernel Reset	561
CPU4_HRB_PCON0	CPU4 program control 0	$2920C_H$	P0, 32	SFRE, P0, SV, 32	Kernel Reset	294
CPU4_HRB_PIEAR	CPU4 HR specific program integrity error address register	29210_H	P0, 32	P0, SV, 32	Kernel Reset	281
CPU4_HRB_PIETR	CPU4 HR specific program integrity error trap register	29214_H	P0, 32	P0, SV, 32	Kernel Reset	282

(table continues...)

Table 219 (continued) Register overview - CPU4 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU4_HRB_COMPAT	CPU4 compatibility control register	29400_H	P0, 32	SFRSE, P0, SV, 32	Kernel Reset	235
CPU4_HRB_FPU_TRAP_CON	CPU4 HR specific trap control register	$2A000_H$	P0, 32	P0, SV, 32	Kernel Reset	263
CPU4_HRB_FPU_TRAP_PC	CPU4 HR specific trapping instruction program counter register	$2A004_H$	P0, 32	P0, SV, 32	Kernel Reset	265
CPU4_HRB_FPU_TRAP_OPC	CPU4 HR specific trapping instruction opcode register	$2A008_H$	P0, 32	P0, SV, 32	Kernel Reset	266
CPU4_HRB_FPU_TRAP_SRC1_L	CPU4 HR specific trapping instruction operand register	$2A010_H$	P0, 32	P0, SV, 32	Kernel Reset	267
CPU4_HRB_FPU_TRAP_SRC1_U	CPU4 HR specific trapping instruction operand register	$2A014_H$	P0, 32	P0, SV, 32	Kernel Reset	268
CPU4_HRB_FPU_TRAP_SRC2_L	CPU4 HR specific trapping instruction operand register	$2A018_H$	P0, 32	P0, SV, 32	Kernel Reset	268
CPU4_HRB_FPU_TRAP_SRC2_U	CPU4 HR specific trapping instruction operand register	$2A01C_H$	P0, 32	P0, SV, 32	Kernel Reset	269
CPU4_HRB_FPU_TRAP_SRC3_L	CPU4 HR specific trapping instruction operand register	$2A020_H$	P0, 32	P0, SV, 32	Kernel Reset	270
CPU4_HRB_FPU_TRAP_SRC3_U	CPU4 HR specific trapping instruction operand register	$2A024_H$	P0, 32	P0, SV, 32	Kernel Reset	270
CPU4_HRB_FPU_SYNC_TRAP_CON	CPU4 HR specific synchronous trap control register	$2A030_H$	P0, 32	P0, SV, 32	Kernel Reset	271
CPU4_HRB_FPU_SYNC_TRAP_OPC	CPU4 HR specific trapping instruction opcode register	$2A034_H$	P0, 32	P0, SV, 32	Kernel Reset	273
CPU4_HRB_DPRy_L (y=0-23)	CPU4 HR specific data protection range y, lower bound register	$2C000_H$+y*8	P0, 32	P0, SV, 32	Kernel Reset	283
CPU4_HRB_DPRy_U (y=0-23)	CPU4 HR specific data protection range y, upper bound register	$2C004_H$+y*8	P0, 32	P0, SV, 32	Kernel Reset	284
CPU4_HRB_CPRy_L (y=0-15)	CPU4 HR specific code protection range y, lower bound register	$2D000_H$+y*8	P0, 32	P0, SV, 32	Kernel Reset	285

(table continues...)

Table 219 (continued) Register overview - CPU4 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU4_HRB_CPRy_U (y=0-15)	CPU4 HR specific code protection range y, upper bound register	$2D004_H$+y*8	P0, 32	P0, SV, 32	Kernel Reset	286
CPU4_HRB_CPXE_y (y=0-3)	CPU4 HR specific code protection execute enable register set y	$2E000_H$+y*4	P0, 32	P0, SV, 32	Kernel Reset	287
CPU4_HRB_DPRE_y (y=0-3)	CPU4 HR specific data protection read enable register set y	$2E010_H$+y*4	P0, 32	P0, SV, 32	Kernel Reset	288
CPU4_HRB_DPWE_y (y=0-3)	CPU4 HR specific data protection write enable register set y	$2E020_H$+y*4	P0, 32	P0, SV, 32	Kernel Reset	289
CPU4_HRB_CPXE_y (y=4-7)	CPU4 HR specific code protection execute enable register set y	$2E040_H$+(y-4)*4	P0, 32	P0, SV, 32	Kernel Reset	287
CPU4_HRB_DPRE_y (y=4-7)	CPU4 HR specific data protection read enable register set y	$2E050_H$+(y-4)*4	P0, 32	P0, SV, 32	Kernel Reset	288
CPU4_HRB_DPWE_y (y=4-7)	CPU4 HR specific data protection write enable register set y	$2E060_H$+(y-4)*4	P0, 32	P0, SV, 32	Kernel Reset	289
CPU4_HRB_TPS_CON	CPU4 HR specific temporal protection system control register	$2E400_H$	P0, 32	P0, SV, 32	Kernel Reset	296
CPU4_HRB_TPS_TIMERn (n=0-2)	CPU4 HR specific temporal protection system timer register n	$2E404_H$+n*4	P0, 32	P0, SV, 32	Kernel Reset	297
CPU4_HRB_TRnEVT (n=0-7)	CPU4 trigger event n	$2F000_H$+n*8	P0, 32	P0, SV, 32	Debug Reset	209
CPU4_HRB_TRnADR (n=0-7)	CPU4 trigger address n	$2F004_H$+n*8	P0, 32	P0, SV, 32	Debug Reset	210
CPU4_HRB_CCTRL	CPU4 HR specific counter control	$2FC00_H$	P0, 32	P0, SV, 32	Kernel Reset	226
CPU4_HRB_CCNT	CPU4 HR specific CPU clock cycle count	$2FC04_H$	P0, 32	P0, SV, 32	Kernel Reset	227
CPU4_HRB_ICNT	CPU4 HR specific instruction count	$2FC08_H$	P0, 32	P0, SV, 32	Kernel Reset	228
CPU4_HRB_M1CNT	CPU4 HR specific multi-count register 1	$2FC0C_H$	P0, 32	P0, SV, 32	Kernel Reset	229

(table continues...)

Table 219 (continued) Register overview - CPU4 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU4_HRB_M2CNT	CPU4 HR specific multi-count register 2	$2FC10_H$	P0, 32	P0, SV, 32	Kernel Reset	230
CPU4_HRB_M3CNT	CPU4 HR specific multi-count register 3	$2FC14_H$	P0, 32	P0, SV, 32	Kernel Reset	231
CPU4_HRB_DBGSR	CPU4 debug status register	$2FD00_H$	P0, 32	P0, SV, 32	Debug Reset	211
CPU4_HRB_EXEVT	CPU4 external event register	$2FD08_H$	P0, 32	P0, SV, 32	Debug Reset	212
CPU4_HRB_CREVT	CPU4 core register access event	$2FD0C_H$	P0, 32	P0, SV, 32	Debug Reset	213
CPU4_HRB_SWEVT	CPU4 software debug event	$2FD10_H$	P0, 32	P0, SV, 32	Debug Reset	214
CPU4_HRB_DBGACT	CPU4 debug action configuration	$2FD14_H$	P0, 32	P0, SV, 32	Debug Reset	215
CPU4_HRB_TRIG_ACC	CPU4 trigger event accumulator	$2FD30_H$	P0, 32	P0, SV, 32	Debug Reset	217
CPU4_HRB_DMS	CPU4 debug monitor start address	$2FD40_H$	P0, 32	P0, SV, 32	Debug Reset	534
CPU4_HRB_DCX	CPU4 debug context save area pointer	$2FD44_H$	P0, 32	P0, SV, 32	Debug Reset	536
CPU4_HRB_DBGTCR	CPU4 debug trap control	$2FD48_H$	P0, 32	P0, SV, 32	Application Reset	219
CPU4_HRB_DBGCFG	CPU4 debug configuration	$2FD4C_H$	P0, 32	P0, SV, 32	Debug Reset	220
CPU4_HRB_TRCCFG	CPU4 trace configuration	$2FD50_H$	P0, 32	P0, SV, 32	Debug Reset	221
CPU4_HRB_TRCFILT	CPU4 trace filter	$2FD54_H$	P0, 32	P0, SV, 32	Debug Reset	222
CPU4_HRB_TRCLIM	CPU4 trace bandwidth limiter	$2FD58_H$	P0, 32	P0, SV, 32	Debug Reset	224
CPU4_HRB_TS16PTCCTRL	CPU4 MCDS performance counter control register	$2FD60_H$	P0, 32	P0, SV, 32	Debug Reset	225
CPU4_HRB_PCXI	CPU4 previous context information register	$2FE00_H$	P0, 32	P0, SV, 32	Kernel Reset	236
CPU4_HRB_PSW	CPU4 HR specific program status word	$2FE04_H$	P0, 32	P0, SV, 32	Kernel Reset	237
CPU4_HRB_PC	CPU4 program counter	$2FE08_H$	P0, 32	P0, SV, 32	Kernel Reset	538
CPU4_HRB_CORECON	CPU4 HR specific core configuration register	$2FE14_H$	P0, 32	SFRSE, P0, SV, 32	Kernel Reset	250
CPU4_HRB_CPU_ID	CPU4 identification register	$2FE18_H$	P0, 32	P0, SV, 32	Kernel Reset	540

(table continues...)

Table 219 (continued) Register overview - CPU4 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU4_HRB_CORE_ID	CPU4 core identification register	$2FE1C_H$	P0, 32	P0, SV, 32	Kernel Reset	542
CPU4_HRB_BIV	CPU4 HR specific base interrupt vector table pointer	$2FE20_H$	P0, 32	SFRE, P0, SV, 32	Kernel Reset	251
CPU4_HRB_BTV	CPU4 HR specific base trap vector table pointer	$2FE24_H$	P0, 32	SFRE, P0, SV, 32	Kernel Reset	555
CPU4_HRB_ISP	CPU4 HR specific interrupt stack pointer	$2FE28_H$	P0, 32	SFRE, P0, SV, 32	Kernel Reset	253
CPU4_HRB_ICR	CPU4 interrupt control register	$2FE2C_H$	P0, 32	P0, SV, 32	Kernel Reset	242
CPU4_HRB_PPRS	CPU4 previous PRS	$2FE34_H$	P0, 32	P0, SV, 32	Kernel Reset	244
CPU4_HRB_FCX	CPU4 HR specific free CSA list head pointer	$2FE38_H$	P0, 32	P0, SV, 32	Kernel Reset	254
CPU4_HRB_LCX	CPU4 HR specific free CSA list limit pointer	$2FE3C_H$	P0, 32	P0, SV, 32	Kernel Reset	255
CPU4_HRB_SWID	CPU4 software identification	$2FE40_H$	P0, 32	P0, SV, 32	Kernel Reset	544
CPU4_HRB_CUS_ID	CPU4 customer ID register	$2FE50_H$	P0, 32	P0, SV, 32	Kernel Reset	546
CPU4_HRB_BOOTCON	CPU4 core boot configuration register	$2FE60_H$	P0, 32	P0, SV, 32	Kernel Reset	548
CPU4_HRB_LCLCON	CPU4 lockstep control register	$2FE64_H$	P0, 32	STP, SFRSE, P0, SV, 32	Cold PowerOn Reset	550
CPU4_HRB_CCON	CPU4 clock control register	$2FE68_H$	P0, 32	P0, SV, 32	Kernel Reset	248
CPU4_HRB_TCCON	CPU4 hardware configuration	$2FE6C_H$	P0, 32	P0, SV, 32	Kernel Reset	552
CPU4_HRB_Ay (y=0-1)	CPU4 HR specific address general purpose register y	$2FF80_H$+y*4	P0, 32	P0, SV, 32	Kernel Reset	274
CPU4_HRB_Ay (y=8-9)	CPU4 HR specific address general purpose register y	$2FFA0_H$+(y-8)*4	P0, 32	P0, SV, 32	Kernel Reset	274
CPU4_HRHV_SEGEN	CPU4 SRI error generation register	31030_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	255
CPU4_HRHV_LCLTEST	CPU4 lockstep test register	31040_H	P0, 32	P0, SV, 32	Cold PowerOn Reset	256
CPU4_HRHV_PMA0	CPU4 data access cacheability register	38100_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	232
CPU4_HRHV_PMA1	CPU4 code access cacheability register	38104_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	233

(table continues...)

Table 219 (continued) Register overview - CPU4 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU4_HRHV_PMA2	CPU4 peripheral space identifier register	38108_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	234
CPU4_HRHV_DCON2	CPU4 data control register 2	39000_H	P0, 32	P0, SV, 32	Kernel Reset	558
CPU4_HRHV_DCON1	CPU4 data control register 1	39008_H	P0, 32	P0, SV, 32	Kernel Reset	258
CPU4_HRHV_SMACON	CPU4 SIST mode access control register	$3900C_H$	P0, 32	SFRSE, P0, SV, 32	Kernel Reset	234
CPU4_HRHV_DSTR	CPU4 HR specific data synchronous trap register	39010_H	P0, 32	P0, SV, 32	Kernel Reset	259
CPU4_HRHV_DATR	CPU4 HR specific data asynchronous trap register	39018_H	P0, 32	P0, SV, 32	Kernel Reset	261
CPU4_HRHV_DEADD	CPU4 HR specific data error address register	$3901C_H$	P0, 32	P0, SV, 32	Kernel Reset	262
CPU4_HRHV_DIEAR	CPU4 data integrity error address register	39020_H	P0, 32	P0, SV, 32	Kernel Reset	279
CPU4_HRHV_DIETR	CPU4 data integrity error trap register	39024_H	P0, 32	P0, SV, 32	Kernel Reset	280
CPU4_HRHV_DCON0	CPU4 data memory control register	39040_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	258
CPU4_HRHV_PSTR	CPU4 HR specific program synchronous trap register	39200_H	P0, 32	P0, SV, 32	Kernel Reset	295
CPU4_HRHV_PCON1	CPU4 program control 1	39204_H	P0, 32	P0, SV, 32	Kernel Reset	292
CPU4_HRHV_PCON2	CPU4 program control 2	39208_H	P0, 32	P0, SV, 32	Kernel Reset	561
CPU4_HRHV_PCON0	CPU4 program control 0	$3920C_H$	P0, 32	SFRE, P0, SV, 32	Kernel Reset	294
CPU4_HRHV_PIEAR	CPU4 HR specific program integrity error address register	39210_H	P0, 32	P0, SV, 32	Kernel Reset	281
CPU4_HRHV_PIETR	CPU4 HR specific program integrity error trap register	39214_H	P0, 32	P0, SV, 32	Kernel Reset	282
CPU4_HRHV_COMPAT	CPU4 compatibility control register	39400_H	P0, 32	SFRSE, P0, SV, 32	Kernel Reset	235
CPU4_HRHV_FPU_TRAP_CON	CPU4 HR specific trap control register	$3A000_H$	P0, 32	P0, SV, 32	Kernel Reset	263
CPU4_HRHV_FPU_TRAP_PC	CPU4 HR specific trapping instruction program counter register	$3A004_H$	P0, 32	P0, SV, 32	Kernel Reset	265
CPU4_HRHV_FPU_TRAP_OPC	CPU4 HR specific trapping instruction opcode register	$3A008_H$	P0, 32	P0, SV, 32	Kernel Reset	266

(table continues...)

Table 219 (continued) Register overview - CPU4 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU4_HRHV_FPU_TRAP_SRC1_L	CPU4 HR specific trapping instruction operand register	$3A010_H$	P0, 32	P0, SV, 32	Kernel Reset	267
CPU4_HRHV_FPU_TRAP_SRC1_U	CPU4 HR specific trapping instruction operand register	$3A014_H$	P0, 32	P0, SV, 32	Kernel Reset	268
CPU4_HRHV_FPU_TRAP_SRC2_L	CPU4 HR specific trapping instruction operand register	$3A018_H$	P0, 32	P0, SV, 32	Kernel Reset	268
CPU4_HRHV_FPU_TRAP_SRC2_U	CPU4 HR specific trapping instruction operand register	$3A01C_H$	P0, 32	P0, SV, 32	Kernel Reset	269
CPU4_HRHV_FPU_TRAP_SRC3_L	CPU4 HR specific trapping instruction operand register	$3A020_H$	P0, 32	P0, SV, 32	Kernel Reset	270
CPU4_HRHV_FPU_TRAP_SRC3_U	CPU4 HR specific trapping instruction operand register	$3A024_H$	P0, 32	P0, SV, 32	Kernel Reset	270
CPU4_HRHV_FPU_SYNC_TRAP_CON	CPU4 HR specific synchronous trap control register	$3A030_H$	P0, 32	P0, SV, 32	Kernel Reset	271
CPU4_HRHV_FPU_SYNC_TRAP_OPC	CPU4 HR specific trapping instruction opcode register	$3A034_H$	P0, 32	P0, SV, 32	Kernel Reset	273
CPU4_HRHV_VCON0	CPU4 virtualization control register 0	$3B000_H$	P0, 32	P0, SV, 32	Kernel Reset	275
CPU4_HRHV_VCON1	CPU4 virtualization control register 1	$3B004_H$	P0, 32	P0, SV, 32	Kernel Reset	276
CPU4_HRHV_VCON2	CPU4 virtualization control register 2	$3B008_H$	P0, 32	P0, SV, 32	Kernel Reset	276
CPU4_HRHV_BHV	CPU4 base hypervisor vector table pointer	$3B010_H$	P0, 32	P0, SV, 32, SFRE	Kernel Reset	559
CPU4_HRHV_VMn_ICR (n=0-7)	CPU4 Virtual machine n interrupt state	$3B100_H$+n*4	P0, 32	P0, SV, 32	Kernel Reset	278
CPU4_HRHV_VMn_PETHRESH (n=0-7)	CPU4 Virtual machine n pre-emption threshold	$3B200_H$+n*4	P0, 32	P0, SV, 32	Kernel Reset	279
CPU4_HRHV_DPRy_L (y=0-23)	CPU4 HR specific data protection range y, lower bound register	$3C000_H$+y*8	P0, 32	P0, SV, 32	Kernel Reset	283

(table continues...)

Table 219 (continued) Register overview - CPU4 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU4_HRHV_DPRy_U (y=0-23)	CPU4 HR specific data protection range y, upper bound register	$3C004_H$+y*8	P0, 32	P0, SV, 32	Kernel Reset	284
CPU4_HRHV_CPRy_L (y=0-15)	CPU4 HR specific code protection range y, lower bound register	$3D000_H$+y*8	P0, 32	P0, SV, 32	Kernel Reset	285
CPU4_HRHV_CPRy_U (y=0-15)	CPU4 HR specific code protection range y, upper bound register	$3D004_H$+y*8	P0, 32	P0, SV, 32	Kernel Reset	286
CPU4_HRHV_CPXE_y (y=0-3)	CPU4 HR specific code protection execute enable register set y	$3E000_H$+y*4	P0, 32	P0, SV, 32	Kernel Reset	287
CPU4_HRHV_DPRE_y (y=0-3)	CPU4 HR specific data protection read enable register set y	$3E010_H$+y*4	P0, 32	P0, SV, 32	Kernel Reset	288
CPU4_HRHV_DPWE_y (y=0-3)	CPU4 HR specific data protection write enable register set y	$3E020_H$+y*4	P0, 32	P0, SV, 32	Kernel Reset	289
CPU4_HRHV_CPXE_y (y=4-7)	CPU4 HR specific code protection execute enable register set y	$3E040_H$+(y-4)*4	P0, 32	P0, SV, 32	Kernel Reset	287
CPU4_HRHV_DPRE_y (y=4-7)	CPU4 HR specific data protection read enable register set y	$3E050_H$+(y-4)*4	P0, 32	P0, SV, 32	Kernel Reset	288
CPU4_HRHV_DPWE_y (y=4-7)	CPU4 HR specific data protection write enable register set y	$3E060_H$+(y-4)*4	P0, 32	P0, SV, 32	Kernel Reset	289
CPU4_HRHV_TPS_CON	CPU4 HR specific temporal protection system control register	$3E400_H$	P0, 32	P0, SV, 32	Kernel Reset	296
CPU4_HRHV_TPS_TIMERn (n=0-2)	CPU4 HR specific temporal protection system timer register n	$3E404_H$+n*4	P0, 32	P0, SV, 32	Kernel Reset	297
CPU4_HRHV_TRnEVT (n=0-7)	CPU4 trigger event n	$3F000_H$+n*8	P0, 32	P0, SV, 32	Debug Reset	209
CPU4_HRHV_TRnADR (n=0-7)	CPU4 trigger address n	$3F004_H$+n*8	P0, 32	P0, SV, 32	Debug Reset	210
CPU4_HRHV_CCTRL	CPU4 HR specific counter control	$3FC00_H$	P0, 32	P0, SV, 32	Kernel Reset	226

(table continues...)

Table 219 (continued) Register overview - CPU4 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU4_HRHV_CCNT	CPU4 HR specific CPU clock cycle count	$3FC04_H$	P0, 32	P0, SV, 32	Kernel Reset	227
CPU4_HRHV_ICNT	CPU4 HR specific instruction count	$3FC08_H$	P0, 32	P0, SV, 32	Kernel Reset	228
CPU4_HRHV_M1CNT	CPU4 HR specific multi-count register 1	$3FC0C_H$	P0, 32	P0, SV, 32	Kernel Reset	229
CPU4_HRHV_M2CNT	CPU4 HR specific multi-count register 2	$3FC10_H$	P0, 32	P0, SV, 32	Kernel Reset	230
CPU4_HRHV_M3CNT	CPU4 HR specific multi-count register 3	$3FC14_H$	P0, 32	P0, SV, 32	Kernel Reset	231
CPU4_HRHV_DBGSR	CPU4 debug status register	$3FD00_H$	P0, 32	P0, SV, 32	Debug Reset	211
CPU4_HRHV_EXEVT	CPU4 external event register	$3FD08_H$	P0, 32	P0, SV, 32	Debug Reset	212
CPU4_HRHV_CREVT	CPU4 core register access event	$3FD0C_H$	P0, 32	P0, SV, 32	Debug Reset	213
CPU4_HRHV_SWEVT	CPU4 software debug event	$3FD10_H$	P0, 32	P0, SV, 32	Debug Reset	214
CPU4_HRHV_DBGACT	CPU4 debug action configuration	$3FD14_H$	P0, 32	P0, SV, 32	Debug Reset	215
CPU4_HRHV_TRIG_ACC	CPU4 trigger event accumulator	$3FD30_H$	P0, 32	P0, SV, 32	Debug Reset	217
CPU4_HRHV_DMS	CPU4 debug monitor start address	$3FD40_H$	P0, 32	P0, SV, 32	Debug Reset	534
CPU4_HRHV_DCX	CPU4 debug context save area pointer	$3FD44_H$	P0, 32	P0, SV, 32	Debug Reset	536
CPU4_HRHV_DBGTCR	CPU4 debug trap control	$3FD48_H$	P0, 32	P0, SV, 32	Application Reset	219
CPU4_HRHV_DBGCFG	CPU4 debug configuration	$3FD4C_H$	P0, 32	P0, SV, 32	Debug Reset	220
CPU4_HRHV_TRCCFG	CPU4 trace configuration	$3FD50_H$	P0, 32	P0, SV, 32	Debug Reset	221
CPU4_HRHV_TRCFILT	CPU4 trace filter	$3FD54_H$	P0, 32	P0, SV, 32	Debug Reset	222
CPU4_HRHV_TRCLIM	CPU4 trace bandwidth limiter	$3FD58_H$	P0, 32	P0, SV, 32	Debug Reset	224
CPU4_HRHV_TS16PTCCTRL	CPU4 MCDS performance counter control register	$3FD60_H$	P0, 32	P0, SV, 32	Debug Reset	225
CPU4_HRHV_PCXI	CPU4 previous context information register	$3FE00_H$	P0, 32	P0, SV, 32	Kernel Reset	236
CPU4_HRHV_PSW	CPU4 HR specific program status word	$3FE04_H$	P0, 32	P0, SV, 32	Kernel Reset	237

(table continues...)

Table 219 (continued) Register overview - CPU4 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU4_HRHV_PC	CPU4 program counter	$3FE08_H$	P0, 32	P0, SV, 32	Kernel Reset	538
CPU4_HRHV_CORECON	CPU4 HR specific core configuration register	$3FE14_H$	P0, 32	SFRSE, P0, SV, 32	Kernel Reset	250
CPU4_HRHV_CPU_ID	CPU4 identification register	$3FE18_H$	P0, 32	P0, SV, 32	Kernel Reset	540
CPU4_HRHV_CORE_ID	CPU4 core identification register	$3FE1C_H$	P0, 32	P0, SV, 32	Kernel Reset	542
CPU4_HRHV_BIV	CPU4 HR specific base interrupt vector table pointer	$3FE20_H$	P0, 32	SFRE, P0, SV, 32	Kernel Reset	251
CPU4_HRHV_BTV	CPU4 HR specific base trap vector table pointer	$3FE24_H$	P0, 32	SFRE, P0, SV, 32	Kernel Reset	557
CPU4_HRHV_ISP	CPU4 HR specific interrupt stack pointer	$3FE28_H$	P0, 32	SFRE, P0, SV, 32	Kernel Reset	253
CPU4_HRHV_ICR	CPU4 interrupt control register	$3FE2C_H$	P0, 32	P0, SV, 32	Kernel Reset	242
CPU4_HRHV_PPRS	CPU4 previous PRS	$3FE34_H$	P0, 32	P0, SV, 32	Kernel Reset	244
CPU4_HRHV_FCX	CPU4 HR specific free CSA list head pointer	$3FE38_H$	P0, 32	P0, SV, 32	Kernel Reset	254
CPU4_HRHV_LCX	CPU4 HR specific free CSA list limit pointer	$3FE3C_H$	P0, 32	P0, SV, 32	Kernel Reset	255
CPU4_HRHV_SWID	CPU4 software identification	$3FE40_H$	P0, 32	P0, SV, 32	Kernel Reset	544
CPU4_HRHV_CUS_ID	CPU4 customer ID register	$3FE50_H$	P0, 32	P0, SV, 32	Kernel Reset	546
CPU4_HRHV_BOOTCON	CPU4 core boot configuration register	$3FE60_H$	P0, 32	P0, SV, 32	Kernel Reset	548
CPU4_HRHV_LCLCON	CPU4 lockstep control register	$3FE64_H$	P0, 32	STP, SFRSE, P0, SV, 32	Cold PowerOn Reset	550
CPU4_HRHV_CCON	CPU4 clock control register	$3FE68_H$	P0, 32	P0, SV, 32	Kernel Reset	248
CPU4_HRHV_TCCON	CPU4 hardware configuration	$3FE6C_H$	P0, 32	P0, SV, 32	Kernel Reset	552
CPU4_HRHV_Ay (y=0-1)	CPU4 HR specific address general purpose register y	$3FF80_H$+ y*4	P0, 32	P0, SV, 32	Kernel Reset	274
CPU4_HRHV_Ay (y=8-9)	CPU4 HR specific address general purpose register y	$3FFA0_H$+ (y-8)*4	P0, 32	P0, SV, 32	Kernel Reset	274

4.10.4.1.10 Register overview - CPU5 domain SRI (ascending offset address)

Table 220 Register overview - CPU5 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU5_STM_CLC	CPU5 system timer clock control register	00000_H	P0, 32	P0, SV, SFRE, 32	System Reset	351
CPU5_STMHV_CLC	CPU5 system timer clock control register, hypervisor view	00000_H	P0, 32	P0, SV, SFRE, 32	System Reset	351
CPU5_STM_OCS	CPU5 OCDS control and status register	00004_H	P0, 32	P0, SV, 32, DEN	Debug Reset	352
CPU5_STMHV_OCS	CPU5 OCDS control and status register, hypervisor view	00004_H	P0, 32	P0, SV, 32, DEN	Debug Reset	352
CPU5_STM_ABS	CPU5 system timer absolute timer	00020_H	(P0, 32)| (P0, 64)	P0, SV, 64, SFRSE	System Reset	353
CPU5_STMHV_ABS	CPU5 system timer absolute timer, hypervisor view	00020_H	(P0, 32)| (P0, 64)	P0, SV, 64, SFRSE	System Reset	353
CPU5_STMHV_VMn_CMPc (c=0-1;n=0-7)	CPU5 VMn system timer compare register c, hypervisor view	00100_H+ n*20_H+c *4	P1, 32	P1, SV, 32	System Reset	354
CPU5_STMHV_VMn_CMCON (n=0-7)	CPU5 VMn system timer compare match control register, hypervisor view	00108_H+ n*20_H	P1, 32	P1, SV, 32	System Reset	355
CPU5_STMHV_VMn_ICR (n=0-7)	CPU5 VMn system timer interrupt control register, hypervisor view	$0010C_H$+ n*20_H	P1, 32	P1, SV, 32	System Reset	356
CPU5_STMHV_VMn_ISCR (n=0-7)	CPU5 VMn system timer interrupt set/clear register, hypervisor view	00110_H+ n*20_H	P1, 32	P1, SV, 32	System Reset	357
CPU5_STMHV_VMn_ISR (n=0-7)	CPU5 VMn system timer interrupt status register, hypervisor view	00114_H+ n*20_H	P1, 32	P1, SV, 32	System Reset	358
CPU5_STMHV_VMn_RELTIM (n=0-7)	CPU5 VMn system timer relative timer, hypervisor view	00118_H+ n*20_H	(P1, 32)| (P1, 64)	P1, SV, 64	System Reset	359
CPU5_STM_CMPc (c=0-1)	CPU5 system timer compare register c	00120_H+ c*4	P1, 32	P1, SV, 32	System Reset	354
CPU5_STM_CMCON	CPU5 system timer compare match control register	00128_H	P1, 32	P1, SV, 32	System Reset	355
CPU5_STM_ICR	CPU5 system timer interrupt control register	$0012C_H$	P1, 32	P1, SV, 32	System Reset	356

(table continues...)

Table 220 (continued) Register overview - CPU5 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU5_STM_ISCR	CPU5 system timer interrupt set/clear register	00130_H	P1, 32	P1, SV, 32	System Reset	357
CPU5_STM_ISR	CPU5 system timer interrupt status register	00134_H	P1, 32	P1, SV, 32	System Reset	358
CPU5_STM_RELTIM	CPU5 system timer relative timer	00138_H	(P1, 32)| (P1, 64)	P1, SV, 64	System Reset	359
CPU5_KRST0	CPU5 reset register 0	$0D000_H$	P0, 32	RSTE, P0, SV, 32	Kernel Reset	297
CPU5_KRST1	CPU5 reset register 1	$0D004_H$	P0, 32	RSTE, P0, SV, 32	Kernel Reset	298
CPU5_KRSTCLR	CPU5 reset clear register	$0D008_H$	P0, 32	RSTE, P0, SV, 32	Kernel Reset	299
CPU5_PROTSPRSE	CPU5 PROTSPR register safe endinit	$0E008_H$	32	SV, 32, PROT	Application Reset	562
CPU5_ACCENSPRCFG_WRA	CPU5 shadow SPR, write access enable register A	$0E020_H$	32	SPRSE, SV, 32	Application Reset	565
CPU5_ACCENSPRCFG_WRB	CPU5 shadow SPR, write access enable register B	$0E024_H$	32	SPRSE, SV, 32	Application Reset	566
CPU5_ACCENSPRCFG_RDA	CPU5 shadow SPR, read access enable register A	$0E028_H$	32	SPRSE, SV, 32	Application Reset	567
CPU5_ACCENSPRCFG_RDB	CPU5 shadow SPR, read access enable register B	$0E02C_H$	32	SPRSE, SV, 32	Application Reset	568
CPU5_ACCENSPRCFG_VM	CPU5 shadow SPR, VM access enable register	$0E030_H$	32	SPRSE, SV, 32	Application Reset	310
CPU5_ACCENSPRCFG_PRS	CPU5 shadow SPR, PRS access enable register	$0E034_H$	32	SPRSE, SV, 32	Application Reset	310
CPU5_ACCENSPRCFG_RGNLA	CPU5 shadow SPR, region lower address register	$0E038_H$	32	SPRSE, SV, 32	Application Reset	311
CPU5_ACCENSPRCFG_RGNUA	CPU5 shadow SPR, region upper address register	$0E03C_H$	32	SPRSE, SV, 32	Application Reset	312
CPU5_PROTDLMUSE	CPU5 PROTDLMU register safe endinit	$0E048_H$	32	SV, 32, PROT	Application Reset	312
CPU5_ACCENDLMUCFG_WRA	CPU5 shadow DLMU, write access enable register A	$0E060_H$	32	DLMUSE, SV, 32	Application Reset	314
CPU5_ACCENDLMUCFG_WRB	CPU5 shadow DLMU, write access enable register B	$0E064_H$	32	DLMUSE, SV, 32	Application Reset	315
CPU5_ACCENDLMUCFG_RDA	CPU5 shadow DLMU, read access enable register A	$0E068_H$	32	DLMUSE, SV, 32	Application Reset	315
CPU5_ACCENDLMUCFG_RDB	CPU5 shadow DLMU, read access enable register B	$0E06C_H$	32	DLMUSE, SV, 32	Application Reset	316
CPU5_ACCENDLMUCFG_VM	CPU5 shadow DLMU, VM access enable register	$0E070_H$	32	DLMUSE, SV, 32	Application Reset	316

(table continues...)

Table 220 (continued) Register overview - CPU5 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU5_ACCENDLMUCFG_PRS	CPU5 shadow DLMU, PRS access enable register	$0E074_H$	32	DLMUSE, SV, 32	Application Reset	317
CPU5_ACCENDLMUCFG_RGNLA	CPU5 shadow DLMU, region lower address register	$0E078_H$	32	DLMUSE, SV, 32	Application Reset	317
CPU5_ACCENDLMUCFG_RGNUA	CPU5 shadow DLMU, region upper address register	$0E07C_H$	32	DLMUSE, SV, 32	Application Reset	318
CPU5_PROTSFRE	CPU5 PROTSFR register endinit	$0E080_H$	32	SV, 32, PROT	Application Reset	569
CPU5_PROTSFRSE	CPU5 PROTSFR register safe endinit	$0E088_H$	32	SV, 32, PROT	Application Reset	571
CPU5_ACCENSFRCFG_WRA	CPU5 shadow SFR, write access enable register A	$0E0A0_H$	32	SFRSE, SV, 32	Application Reset	573
CPU5_ACCENSFRCFG_WRB	CPU5 shadow SFR, write access enable register B	$0E0A4_H$	32	SFRSE, SV, 32	Application Reset	574
CPU5_ACCENSFRCFG_RDA	CPU5 shadow SFR, read access enable register A	$0E0A8_H$	32	SFRSE, SV, 32	Application Reset	575
CPU5_ACCENSFRCFG_RDB	CPU5 shadow SFR, read access enable register B	$0E0AC_H$	32	SFRSE, SV, 32	Application Reset	576
CPU5_ACCENSFRCFG_VM	CPU5 shadow SFR, VM access enable register	$0E0B0_H$	32	SFRSE, SV, 32	Application Reset	325
CPU5_ACCENSFRCFG_PRS	CPU5 shadow SFR, PRS access enable register	$0E0B4_H$	32	SFRSE, SV, 32	Application Reset	325
CPU5_PROTRSTE	CPU5 PROTRST register endinit	$0E0C0_H$	32	SV, 32, PROT	Application Reset	577
CPU5_PROTSTMSE	CPU5 PROTSTM Register Safe Endinit	$0E0D8_H$	32	SV, 32, PROT	Application Reset	579
CPU5_ACCENSTMCFG_WRA	CPU5 shadow STM, write access enable register A	$0E0E0_H$	32	STMSE, SV, 32	Application Reset	582
CPU5_ACCENSTMCFG_WRB	CPU5 shadow STM, write access enable register B	$0E0E4_H$	32	STMSE, SV, 32	Application Reset	330
CPU5_ACCENSTMCFG_RDA	CPU5 shadow STM, read access enable register A	$0E0E8_H$	32	STMSE, SV, 32	Application Reset	583
CPU5_ACCENSTMCFG_RDB	CPU5 shadow STM, read access enable register B	$0E0EC_H$	32	STMSE, SV, 32	Application Reset	584
CPU5_ACCENSTMCFG_VM	CPU5 shadow STM, VM access enable register	$0E0F0_H$	32	STMSE, SV, 32	Application Reset	332
CPU5_ACCENSTMCFG_PRS	CPU5 shadow STM, PRS access enable register	$0E0F4_H$	32	STMSE, SV, 32	Application Reset	333
CPU5_ACCENSPRy_WRA (y=0-15)	CPU5 SPR range y, write access enable register A	$0E200_H$+ y*20_H	32	SPRSE, SV, 32	Application Reset	585

(table continues...)

Table 220 (continued) Register overview - CPU5 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU5_ACCENSPRy_WRB (y=0-15)	CPU5 SPR range y, write access enable register B	$0E204_H$+ y*20_H	32	SPRSE, SV, 32	Application Reset	586
CPU5_ACCENSPRy_RDA (y=0-15)	CPU5 SPR range y, read access enable register A	$0E208_H$+ y*20_H	32	SPRSE, SV, 32	Application Reset	587
CPU5_ACCENSPRy_RDB (y=0-15)	CPU5 SPR range y, read access enable register B	$0E20C_H$+ y*20_H	32	SPRSE, SV, 32	Application Reset	588
CPU5_ACCENSPRy_VM (y=0-15)	CPU5 SPR range y, VM access enable register	$0E210_H$+ y*20_H	32	SPRSE, SV, 32	Application Reset	336
CPU5_ACCENSPRy_PRS (y=0-15)	CPU5 SPR range y, PRS access enable register	$0E214_H$+ y*20_H	32	SPRSE, SV, 32	Application Reset	337
CPU5_ACCENSPRy_RGNLA (y=0-15)	CPU5 SPR range y, region lower address register	$0E218_H$+ y*20_H	32	SPRSE, SV, 32	Application Reset	337
CPU5_ACCENSPRy_RGNUA (y=0-15)	CPU5 SPR range y, region upper address register	$0E21C_H$+ y*20_H	32	SPRSE, SV, 32	Application Reset	338
CPU5_ACCENDLMUy_WRA (y=0-15)	CPU5 DLMU range y, write access enable register A	$0E400_H$+ y*20_H	32	DLMUSE, SV, 32	Application Reset	338
CPU5_ACCENDLMUy_WRB (y=0-15)	CPU5 DLMU range y, write access enable register B	$0E404_H$+ y*20_H	32	DLMUSE, SV, 32	Application Reset	339
CPU5_ACCENDLMUy_RDA (y=0-15)	CPU5 DLMU range y, read access enable register A	$0E408_H$+ y*20_H	32	DLMUSE, SV, 32	Application Reset	339
CPU5_ACCENDLMUy_RDB (y=0-15)	CPU5 DLMU range y, read access enable register B	$0E40C_H$+ y*20_H	32	DLMUSE, SV, 32	Application Reset	340
CPU5_ACCENDLMUy_VM (y=0-15)	CPU5 DLMU range y, VM access enable register	$0E410_H$+ y*20_H	32	DLMUSE, SV, 32	Application Reset	340
CPU5_ACCENDLMUy_PRS (y=0-15)	CPU5 DLMU range y, PRS access enable register	$0E414_H$+ y*20_H	32	DLMUSE, SV, 32	Application Reset	341
CPU5_ACCENDLMUy_RGNLA (y=0-15)	CPU5 DLMU range y, region lower address register	$0E418_H$+ y*20_H	32	DLMUSE, SV, 32	Application Reset	342

(table continues...)

Table 220 (continued) Register overview - CPU5 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU5_ACCENDLMUy_RGNUA (y=0-15)	CPU5 DLMU range y, region upper address register	$0E41C_H$+ y*20_H	32	DLMUSE, SV, 32	Application Reset	342
CPU5_ACCENSFR_WRA	CPU5 SFR, write access enable register A	$0E600_H$	32	SFRSE, SV, 32	Application Reset	589
CPU5_ACCENSFR_WRB	CPU5 SFR, write access enable register B	$0E604_H$	32	SFRSE, SV, 32	Application Reset	590
CPU5_ACCENSFR_RDA	CPU5 SFR, read access enable register A	$0E608_H$	32	SFRSE, SV, 32	Application Reset	591
CPU5_ACCENSFR_RDB	CPU5 SFR, read access enable register B	$0E60C_H$	32	SFRSE, SV, 32	Application Reset	592
CPU5_ACCENSFR_VM	CPU5 SFR, VM access enable register	$0E610_H$	32	SFRSE, SV, 32	Application Reset	345
CPU5_ACCENSFR_PRS	CPU5 SFR, PRS access enable register	$0E614_H$	32	SFRSE, SV, 32	Application Reset	346
CPU5_ACCENSTMVMy_WRA (y=0-7)	CPU5 VMy STM, write access enable register A, virtualization view	$0E700_H$+ y*20_H	32	STMSE, SV, 32	Application Reset	593
CPU5_ACCENSTMVMy_WRB (y=0-7)	CPU5 VMy STM, write access enable register B, virtualization view	$0E704_H$+ y*20_H	32	STMSE, SV, 32	Application Reset	347
CPU5_ACCENSTMVMy_RDA (y=0-7)	CPU5 VMy STM, read access enable register A, virtualization view	$0E708_H$+ y*20_H	32	STMSE, SV, 32	Application Reset	594
CPU5_ACCENSTMVMy_RDB (y=0-7)	CPU5 VMy STM, read access enable register B, virtualization view	$0E70C_H$+ y*20_H	32	STMSE, SV, 32	Application Reset	595
CPU5_ACCENSTMVMy_VM (y=0-7)	CPU5 VMy STM, VM access enable register, virtualization view	$0E710_H$+ y*20_H	32	STMSE, SV, 32	Application Reset	350
CPU5_ACCENSTMVMy_PRS (y=0-7)	CPU5 VMy STM, PRS access enable register, virtualization view	$0E714_H$+ y*20_H	32	STMSE, SV, 32	Application Reset	350
CPU5_OSEL	CPU5 overlay range select register	$0FB00_H$	P0, 32	P0, SV, 32	Kernel Reset	299
CPU5_RABRi (i=0-31)	CPU5 redirected address base register i	$0FB10_H$+ i*12	P0, 32	P0, SV, 32	Kernel Reset	300
CPU5_OTARi (i=0-31)	CPU5 overlay target address register i	$0FB14_H$+ i*12	P0, 32	P0, SV, 32	Kernel Reset	302
CPU5_OMASKi (i=0-31)	CPU5 overlay mask register i	$0FB18_H$+ i*12	P0, 32	P0, SV, 32	Kernel Reset	303

(table continues...)

Table 220 (continued) Register overview - CPU5 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU5_HRA_SEGEN	CPU5 SRI error generation register	11030_H	P0, 32	SFRE, P0, SV, 32	Application Reset	255
CPU5_HRA_LCLTEST	CPU5 lockstep test register	11040_H	P0, 32	P0, SV, 32	Cold PowerOn Reset	256
CPU5_HRA_PMA0	CPU5 data access cacheability register	18100_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	232
CPU5_HRA_PMA1	CPU5 code access cacheability register	18104_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	233
CPU5_HRA_PMA2	CPU5 peripheral space identifier register	18108_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	234
CPU5_HRA_DCON2	CPU5 data control register 2	19000_H	P0, 32	P0, SV, 32	Kernel Reset	558
CPU5_HRA_DCON1	CPU5 data control register 1	19008_H	P0, 32	P0, SV, 32	Kernel Reset	258
CPU5_HRA_SMACON	CPU5 SIST mode access control register	$1900C_H$	P0, 32	SFRSE, P0, SV, 32	Kernel Reset	234
CPU5_HRA_DSTR	CPU5 HR specific data synchronous trap register	19010_H	P0, 32	P0, SV, 32	Kernel Reset	259
CPU5_HRA_DATR	CPU5 HR specific data asynchronous trap register	19018_H	P0, 32	P0, SV, 32	Kernel Reset	261
CPU5_HRA_DEADD	CPU5 HR specific data error address register	$1901C_H$	P0, 32	P0, SV, 32	Kernel Reset	262
CPU5_HRA_DIEAR	CPU5 data integrity error address register	19020_H	P0, 32	P0, SV, 32	Kernel Reset	279
CPU5_HRA_DIETR	CPU5 data integrity error trap register	19024_H	P0, 32	P0, SV, 32	Kernel Reset	280
CPU5_HRA_DCON0	CPU5 data memory control register	19040_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	258
CPU5_HRA_PSTR	CPU5 HR specific program synchronous trap register	19200_H	P0, 32	P0, SV, 32	Kernel Reset	295
CPU5_HRA_PCON1	CPU5 program control 1	19204_H	P0, 32	P0, SV, 32	Kernel Reset	292
CPU5_HRA_PCON2	CPU5 program control 2	19208_H	P0, 32	P0, SV, 32	Kernel Reset	561
CPU5_HRA_PCON0	CPU5 program control 0	$1920C_H$	P0, 32	SFRE, P0, SV, 32	Kernel Reset	294
CPU5_HRA_PIEAR	CPU5 HR specific program integrity error address register	19210_H	P0, 32	P0, SV, 32	Kernel Reset	281
CPU5_HRA_PIETR	CPU5 HR specific program integrity error trap register	19214_H	P0, 32	P0, SV, 32	Kernel Reset	282

(table continues...)

Table 220 (continued) Register overview - CPU5 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU5_HRA_COMPAT	CPU5 compatibility control register	19400_H	P0, 32	SFRSE, P0, SV, 32	Kernel Reset	235
CPU5_HRA_FPU_TRAP_CON	CPU5 HR specific trap control register	$1A000_H$	P0, 32	P0, SV, 32	Kernel Reset	263
CPU5_HRA_FPU_TRAP_PC	CPU5 HR specific trapping instruction program counter register	$1A004_H$	P0, 32	P0, SV, 32	Kernel Reset	265
CPU5_HRA_FPU_TRAP_OPC	CPU5 HR specific trapping instruction opcode register	$1A008_H$	P0, 32	P0, SV, 32	Kernel Reset	266
CPU5_HRA_FPU_TRAP_SRC1_L	CPU5 HR specific trapping instruction operand register	$1A010_H$	P0, 32	P0, SV, 32	Kernel Reset	267
CPU5_HRA_FPU_TRAP_SRC1_U	CPU5 HR specific trapping instruction operand register	$1A014_H$	P0, 32	P0, SV, 32	Kernel Reset	268
CPU5_HRA_FPU_TRAP_SRC2_L	CPU5 HR specific trapping instruction operand register	$1A018_H$	P0, 32	P0, SV, 32	Kernel Reset	268
CPU5_HRA_FPU_TRAP_SRC2_U	CPU5 HR specific trapping instruction operand register	$1A01C_H$	P0, 32	P0, SV, 32	Kernel Reset	269
CPU5_HRA_FPU_TRAP_SRC3_L	CPU5 HR specific trapping instruction operand register	$1A020_H$	P0, 32	P0, SV, 32	Kernel Reset	270
CPU5_HRA_FPU_TRAP_SRC3_U	CPU5 HR specific trapping instruction operand register	$1A024_H$	P0, 32	P0, SV, 32	Kernel Reset	270
CPU5_HRA_FPU_SYNC_TRAP_CON	CPU5 HR specific synchronous trap control register	$1A030_H$	P0, 32	P0, SV, 32	Kernel Reset	271
CPU5_HRA_FPU_SYNC_TRAP_OPC	CPU5 HR specific trapping instruction opcode register	$1A034_H$	P0, 32	P0, SV, 32	Kernel Reset	273
CPU5_HRA_DPRy_L (y=0-23)	CPU5 HR specific data protection range y, lower bound register	$1C000_H$+y*8	P0, 32	P0, SV, 32	Kernel Reset	283
CPU5_HRA_DPRy_U (y=0-23)	CPU5 HR specific data protection range y, upper bound register	$1C004_H$+y*8	P0, 32	P0, SV, 32	Kernel Reset	284
CPU5_HRA_CPRy_L (y=0-15)	CPU5 HR specific code protection range y, lower bound register	$1D000_H$+y*8	P0, 32	P0, SV, 32	Kernel Reset	285

(table continues...)

Table 220 (continued) Register overview - CPU5 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU5_HRA_CPRy_U (y=0-15)	CPU5 HR specific code protection range y, upper bound register	$1D004_H$+y*8	P0, 32	P0, SV, 32	Kernel Reset	286
CPU5_HRA_CPXE_y (y=0-3)	CPU5 HR specific code protection execute enable register set y	$1E000_H$+y*4	P0, 32	P0, SV, 32	Kernel Reset	287
CPU5_HRA_DPRE_y (y=0-3)	CPU5 HR specific data protection read enable register set y	$1E010_H$+y*4	P0, 32	P0, SV, 32	Kernel Reset	288
CPU5_HRA_DPWE_y (y=0-3)	CPU5 HR specific data protection write enable register set y	$1E020_H$+y*4	P0, 32	P0, SV, 32	Kernel Reset	289
CPU5_HRA_CPXE_y (y=4-7)	CPU5 HR specific code protection execute enable register set y	$1E040_H$+(y-4)*4	P0, 32	P0, SV, 32	Kernel Reset	287
CPU5_HRA_DPRE_y (y=4-7)	CPU5 HR specific data protection read enable register set y	$1E050_H$+(y-4)*4	P0, 32	P0, SV, 32	Kernel Reset	288
CPU5_HRA_DPWE_y (y=4-7)	CPU5 HR specific data protection write enable register set y	$1E060_H$+(y-4)*4	P0, 32	P0, SV, 32	Kernel Reset	289
CPU5_HRA_TPS_CON	CPU5 HR specific temporal protection system control register	$1E400_H$	P0, 32	P0, SV, 32	Kernel Reset	296
CPU5_HRA_TPS_TIMERn (n=0-2)	CPU5 HR specific temporal protection system timer register n	$1E404_H$+n*4	P0, 32	P0, SV, 32	Kernel Reset	297
CPU5_HRA_TRnEVT (n=0-7)	CPU5 trigger event n	$1F000_H$+n*8	P0, 32	P0, SV, 32	Debug Reset	209
CPU5_HRA_TRnADR (n=0-7)	CPU5 trigger address n	$1F004_H$+n*8	P0, 32	P0, SV, 32	Debug Reset	210
CPU5_HRA_CCTRL	CPU5 HR specific counter control	$1FC00_H$	P0, 32	P0, SV, 32	Kernel Reset	226
CPU5_HRA_CCNT	CPU5 HR specific CPU clock cycle count	$1FC04_H$	P0, 32	P0, SV, 32	Kernel Reset	227
CPU5_HRA_ICNT	CPU5 HR specific instruction count	$1FC08_H$	P0, 32	P0, SV, 32	Kernel Reset	228
CPU5_HRA_M1CNT	CPU5 HR specific multi-count register 1	$1FC0C_H$	P0, 32	P0, SV, 32	Kernel Reset	229

(table continues...)

Table 220 (continued) Register overview - CPU5 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU5_HRA_M2CNT	CPU5 HR specific multi-count register 2	$1FC10_H$	P0, 32	P0, SV, 32	Kernel Reset	230
CPU5_HRA_M3CNT	CPU5 HR specific multi-count register 3	$1FC14_H$	P0, 32	P0, SV, 32	Kernel Reset	231
CPU5_HRA_DBGSR	CPU5 debug status register	$1FD00_H$	P0, 32	P0, SV, 32	Debug Reset	211
CPU5_HRA_EXEVT	CPU5 external event register	$1FD08_H$	P0, 32	P0, SV, 32	Debug Reset	212
CPU5_HRA_CREVT	CPU5 core register access event	$1FD0C_H$	P0, 32	P0, SV, 32	Debug Reset	213
CPU5_HRA_SWEVT	CPU5 software debug event	$1FD10_H$	P0, 32	P0, SV, 32	Debug Reset	214
CPU5_HRA_DBGACT	CPU5 debug action configuration	$1FD14_H$	P0, 32	P0, SV, 32	Debug Reset	215
CPU5_HRA_TRIG_ACC	CPU5 trigger event accumulator	$1FD30_H$	P0, 32	P0, SV, 32	Debug Reset	217
CPU5_HRA_DMS	CPU5 debug monitor start address	$1FD40_H$	P0, 32	P0, SV, 32	Debug Reset	534
CPU5_HRA_DCX	CPU5 debug context save area pointer	$1FD44_H$	P0, 32	P0, SV, 32	Debug Reset	536
CPU5_HRA_DBGTCR	CPU5 debug trap control	$1FD48_H$	P0, 32	P0, SV, 32	Application Reset	219
CPU5_HRA_DBGCFG	CPU5 debug configuration	$1FD4C_H$	P0, 32	P0, SV, 32	Debug Reset	220
CPU5_HRA_TRCCFG	CPU5 trace configuration	$1FD50_H$	P0, 32	P0, SV, 32	Debug Reset	221
CPU5_HRA_TRCFILT	CPU5 trace filter	$1FD54_H$	P0, 32	P0, SV, 32	Debug Reset	222
CPU5_HRA_TRCLIM	CPU5 trace bandwidth limiter	$1FD58_H$	P0, 32	P0, SV, 32	Debug Reset	224
CPU5_HRA_TS16PTCCTRL	CPU5 MCDS performance counter control register	$1FD60_H$	P0, 32	P0, SV, 32	Debug Reset	225
CPU5_HRA_PCXI	CPU5 previous context information register	$1FE00_H$	P0, 32	P0, SV, 32	Kernel Reset	236
CPU5_HRA_PSW	CPU5 HR specific program status word	$1FE04_H$	P0, 32	P0, SV, 32	Kernel Reset	237
CPU5_HRA_PC	CPU5 program counter	$1FE08_H$	P0, 32	P0, SV, 32	Kernel Reset	538
CPU5_HRA_CORECON	CPU5 HR specific core configuration register	$1FE14_H$	P0, 32	SFRSE, P0, SV, 32	Kernel Reset	250
CPU5_HRA_CPU_ID	CPU5 identification register	$1FE18_H$	P0, 32	P0, SV, 32	Kernel Reset	540

(table continues...)

Table 220 (continued) Register overview - CPU5 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU5_HRA_CORE_ID	CPU5 core identification register	$1FE1C_H$	P0, 32	P0, SV, 32	Kernel Reset	542
CPU5_HRA_BIV	CPU5 HR specific base interrupt vector table pointer	$1FE20_H$	P0, 32	SFRE, P0, SV, 32	Kernel Reset	251
CPU5_HRA_BTV	CPU5 HR specific base trap vector table pointer	$1FE24_H$	P0, 32	SFRE, P0, SV, 32	Kernel Reset	554
CPU5_HRA_ISP	CPU5 HR specific interrupt stack pointer	$1FE28_H$	P0, 32	SFRE, P0, SV, 32	Kernel Reset	253
CPU5_HRA_ICR	CPU5 interrupt control register	$1FE2C_H$	P0, 32	P0, SV, 32	Kernel Reset	242
CPU5_HRA_PPRS	CPU5 previous PRS	$1FE34_H$	P0, 32	P0, SV, 32	Kernel Reset	244
CPU5_HRA_FCX	CPU5 HR specific free CSA list head pointer	$1FE38_H$	P0, 32	P0, SV, 32	Kernel Reset	254
CPU5_HRA_LCX	CPU5 HR specific free CSA list limit pointer	$1FE3C_H$	P0, 32	P0, SV, 32	Kernel Reset	255
CPU5_HRA_SWID	CPU5 software identification	$1FE40_H$	P0, 32	P0, SV, 32	Kernel Reset	544
CPU5_HRA_CUS_ID	CPU5 customer ID register	$1FE50_H$	P0, 32	P0, SV, 32	Kernel Reset	546
CPU5_HRA_BOOTCON	CPU5 core boot configuration register	$1FE60_H$	P0, 32	P0, SV, 32	Kernel Reset	548
CPU5_HRA_LCLCON	CPU5 lockstep control register	$1FE64_H$	P0, 32	STP, SFRSE, P0, SV, 32	Cold PowerOn Reset	550
CPU5_HRA_CCON	CPU5 clock control register	$1FE68_H$	P0, 32	P0, SV, 32	Kernel Reset	248
CPU5_HRA_TCCON	CPU5 hardware configuration	$1FE6C_H$	P0, 32	P0, SV, 32	Kernel Reset	552
CPU5_Dy (y=0-15)	CPU5 data general purpose register y	$1FF00_H$+ y*4	P0, 32	P0, SV, 32	Kernel Reset	274
CPU5_Ay (y=0-15)	CPU5 address general purpose register y	$1FF80_H$+ y*4	P0, 32	P0, SV, 32	Kernel Reset	274
CPU5_HRB_SEGEN	CPU5 SRI error generation register	21030_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	255
CPU5_HRB_LCLTEST	CPU5 lockstep test register	21040_H	P0, 32	P0, SV, 32	Cold PowerOn Reset	256
CPU5_HRB_PMA0	CPU5 data access cacheability register	28100_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	232
CPU5_HRB_PMA1	CPU5 code access cacheability register	28104_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	233

(table continues...)

Table 220 (continued) Register overview - CPU5 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU5_HRB_PMA2	CPU5 peripheral space identifier register	28108_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	234
CPU5_HRB_DCON2	CPU5 data control register 2	29000_H	P0, 32	P0, SV, 32	Kernel Reset	558
CPU5_HRB_DCON1	CPU5 data control register 1	29008_H	P0, 32	P0, SV, 32	Kernel Reset	258
CPU5_HRB_SMACON	CPU5 SIST mode access control register	$2900C_H$	P0, 32	SFRSE, P0, SV, 32	Kernel Reset	234
CPU5_HRB_DSTR	CPU5 HR specific data synchronous trap register	29010_H	P0, 32	P0, SV, 32	Kernel Reset	259
CPU5_HRB_DATR	CPU5 HR specific data asynchronous trap register	29018_H	P0, 32	P0, SV, 32	Kernel Reset	261
CPU5_HRB_DEADD	CPU5 HR specific data error address register	$2901C_H$	P0, 32	P0, SV, 32	Kernel Reset	262
CPU5_HRB_DIEAR	CPU5 data integrity error address register	29020_H	P0, 32	P0, SV, 32	Kernel Reset	279
CPU5_HRB_DIETR	CPU5 data integrity error trap register	29024_H	P0, 32	P0, SV, 32	Kernel Reset	280
CPU5_HRB_DCON0	CPU5 data memory control register	29040_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	258
CPU5_HRB_PSTR	CPU5 HR specific program synchronous trap register	29200_H	P0, 32	P0, SV, 32	Kernel Reset	295
CPU5_HRB_PCON1	CPU5 program control 1	29204_H	P0, 32	P0, SV, 32	Kernel Reset	292
CPU5_HRB_PCON2	CPU5 program control 2	29208_H	P0, 32	P0, SV, 32	Kernel Reset	561
CPU5_HRB_PCON0	CPU5 program control 0	$2920C_H$	P0, 32	SFRE, P0, SV, 32	Kernel Reset	294
CPU5_HRB_PIEAR	CPU5 HR specific program integrity error address register	29210_H	P0, 32	P0, SV, 32	Kernel Reset	281
CPU5_HRB_PIETR	CPU5 HR specific program integrity error trap register	29214_H	P0, 32	P0, SV, 32	Kernel Reset	282
CPU5_HRB_COMPAT	CPU5 compatibility control register	29400_H	P0, 32	SFRSE, P0, SV, 32	Kernel Reset	235
CPU5_HRB_FPU_TRAP_CON	CPU5 HR specific trap control register	$2A000_H$	P0, 32	P0, SV, 32	Kernel Reset	263
CPU5_HRB_FPU_TRAP_PC	CPU5 HR specific trapping instruction program counter register	$2A004_H$	P0, 32	P0, SV, 32	Kernel Reset	265
CPU5_HRB_FPU_TRAP_OPC	CPU5 HR specific trapping instruction opcode register	$2A008_H$	P0, 32	P0, SV, 32	Kernel Reset	266

(table continues...)

Table 220 (continued) Register overview - CPU5 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU5_HRB_FPU_TRAP_SRC1_L	CPU5 HR specific trapping instruction operand register	$2A010_H$	P0, 32	P0, SV, 32	Kernel Reset	267
CPU5_HRB_FPU_TRAP_SRC1_U	CPU5 HR specific trapping instruction operand register	$2A014_H$	P0, 32	P0, SV, 32	Kernel Reset	268
CPU5_HRB_FPU_TRAP_SRC2_L	CPU5 HR specific trapping instruction operand register	$2A018_H$	P0, 32	P0, SV, 32	Kernel Reset	268
CPU5_HRB_FPU_TRAP_SRC2_U	CPU5 HR specific trapping instruction operand register	$2A01C_H$	P0, 32	P0, SV, 32	Kernel Reset	269
CPU5_HRB_FPU_TRAP_SRC3_L	CPU5 HR specific trapping instruction operand register	$2A020_H$	P0, 32	P0, SV, 32	Kernel Reset	270
CPU5_HRB_FPU_TRAP_SRC3_U	CPU5 HR specific trapping instruction operand register	$2A024_H$	P0, 32	P0, SV, 32	Kernel Reset	270
CPU5_HRB_FPU_SYNC_TRAP_CON	CPU5 HR specific synchronous trap control register	$2A030_H$	P0, 32	P0, SV, 32	Kernel Reset	271
CPU5_HRB_FPU_SYNC_TRAP_OPC	CPU5 HR specific trapping instruction opcode register	$2A034_H$	P0, 32	P0, SV, 32	Kernel Reset	273
CPU5_HRB_DPRy_L (y=0-23)	CPU5 HR specific data protection range y, lower bound register	$2C000_H$+y*8	P0, 32	P0, SV, 32	Kernel Reset	283
CPU5_HRB_DPRy_U (y=0-23)	CPU5 HR specific data protection range y, upper bound register	$2C004_H$+y*8	P0, 32	P0, SV, 32	Kernel Reset	284
CPU5_HRB_CPRy_L (y=0-15)	CPU5 HR specific code protection range y, lower bound register	$2D000_H$+y*8	P0, 32	P0, SV, 32	Kernel Reset	285
CPU5_HRB_CPRy_U (y=0-15)	CPU5 HR specific code protection range y, upper bound register	$2D004_H$+y*8	P0, 32	P0, SV, 32	Kernel Reset	286
CPU5_HRB_CPXE_y (y=0-3)	CPU5 HR specific code protection execute enable register set y	$2E000_H$+y*4	P0, 32	P0, SV, 32	Kernel Reset	287
CPU5_HRB_DPRE_y (y=0-3)	CPU5 HR specific data protection read enable register set y	$2E010_H$+y*4	P0, 32	P0, SV, 32	Kernel Reset	288

(table continues...)

Table 220 (continued) Register overview - CPU5 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU5_HRB_DPWE_y (y=0-3)	CPU5 HR specific data protection write enable register set y	$2E020_H$+y*4	P0, 32	P0, SV, 32	Kernel Reset	289
CPU5_HRB_CPXE_y (y=4-7)	CPU5 HR specific code protection execute enable register set y	$2E040_H$+(y-4)*4	P0, 32	P0, SV, 32	Kernel Reset	287
CPU5_HRB_DPRE_y (y=4-7)	CPU5 HR specific data protection read enable register set y	$2E050_H$+(y-4)*4	P0, 32	P0, SV, 32	Kernel Reset	288
CPU5_HRB_DPWE_y (y=4-7)	CPU5 HR specific data protection write enable register set y	$2E060_H$+(y-4)*4	P0, 32	P0, SV, 32	Kernel Reset	289
CPU5_HRB_TPS_CON	CPU5 HR specific temporal protection system control register	$2E400_H$	P0, 32	P0, SV, 32	Kernel Reset	296
CPU5_HRB_TPS_TIMERn (n=0-2)	CPU5 HR specific temporal protection system timer register n	$2E404_H$+n*4	P0, 32	P0, SV, 32	Kernel Reset	297
CPU5_HRB_TRnEVT (n=0-7)	CPU5 trigger event n	$2F000_H$+n*8	P0, 32	P0, SV, 32	Debug Reset	209
CPU5_HRB_TRnADR (n=0-7)	CPU5 trigger address n	$2F004_H$+n*8	P0, 32	P0, SV, 32	Debug Reset	210
CPU5_HRB_CCTRL	CPU5 HR specific counter control	$2FC00_H$	P0, 32	P0, SV, 32	Kernel Reset	226
CPU5_HRB_CCNT	CPU5 HR specific CPU clock cycle count	$2FC04_H$	P0, 32	P0, SV, 32	Kernel Reset	227
CPU5_HRB_ICNT	CPU5 HR specific instruction count	$2FC08_H$	P0, 32	P0, SV, 32	Kernel Reset	228
CPU5_HRB_M1CNT	CPU5 HR specific multi-count register 1	$2FC0C_H$	P0, 32	P0, SV, 32	Kernel Reset	229
CPU5_HRB_M2CNT	CPU5 HR specific multi-count register 2	$2FC10_H$	P0, 32	P0, SV, 32	Kernel Reset	230
CPU5_HRB_M3CNT	CPU5 HR specific multi-count register 3	$2FC14_H$	P0, 32	P0, SV, 32	Kernel Reset	231
CPU5_HRB_DBGSR	CPU5 debug status register	$2FD00_H$	P0, 32	P0, SV, 32	Debug Reset	211
CPU5_HRB_EXEVT	CPU5 external event register	$2FD08_H$	P0, 32	P0, SV, 32	Debug Reset	212
CPU5_HRB_CREVT	CPU5 core register access event	$2FD0C_H$	P0, 32	P0, SV, 32	Debug Reset	213

(table continues...)

Table 220 (continued) Register overview - CPU5 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU5_HRB_SWEVT	CPU5 software debug event	$2FD10_H$	P0, 32	P0, SV, 32	Debug Reset	214
CPU5_HRB_DBGACT	CPU5 debug action configuration	$2FD14_H$	P0, 32	P0, SV, 32	Debug Reset	215
CPU5_HRB_TRIG_ACC	CPU5 trigger event accumulator	$2FD30_H$	P0, 32	P0, SV, 32	Debug Reset	217
CPU5_HRB_DMS	CPU5 debug monitor start address	$2FD40_H$	P0, 32	P0, SV, 32	Debug Reset	534
CPU5_HRB_DCX	CPU5 debug context save area pointer	$2FD44_H$	P0, 32	P0, SV, 32	Debug Reset	536
CPU5_HRB_DBGTCR	CPU5 debug trap control	$2FD48_H$	P0, 32	P0, SV, 32	Application Reset	219
CPU5_HRB_DBGCFG	CPU5 debug configuration	$2FD4C_H$	P0, 32	P0, SV, 32	Debug Reset	220
CPU5_HRB_TRCCFG	CPU5 trace configuration	$2FD50_H$	P0, 32	P0, SV, 32	Debug Reset	221
CPU5_HRB_TRCFILT	CPU5 trace filter	$2FD54_H$	P0, 32	P0, SV, 32	Debug Reset	222
CPU5_HRB_TRCLIM	CPU5 trace bandwidth limiter	$2FD58_H$	P0, 32	P0, SV, 32	Debug Reset	224
CPU5_HRB_TS16PTCCTRL	CPU5 MCDS performance counter control register	$2FD60_H$	P0, 32	P0, SV, 32	Debug Reset	225
CPU5_HRB_PCXI	CPU5 previous context information register	$2FE00_H$	P0, 32	P0, SV, 32	Kernel Reset	236
CPU5_HRB_PSW	CPU5 HR specific program status word	$2FE04_H$	P0, 32	P0, SV, 32	Kernel Reset	237
CPU5_HRB_PC	CPU5 program counter	$2FE08_H$	P0, 32	P0, SV, 32	Kernel Reset	538
CPU5_HRB_CORECON	CPU5 HR specific core configuration register	$2FE14_H$	P0, 32	SFRSE, P0, SV, 32	Kernel Reset	250
CPU5_HRB_CPU_ID	CPU5 identification register	$2FE18_H$	P0, 32	P0, SV, 32	Kernel Reset	540
CPU5_HRB_CORE_ID	CPU5 core identification register	$2FE1C_H$	P0, 32	P0, SV, 32	Kernel Reset	542
CPU5_HRB_BIV	CPU5 HR specific base interrupt vector table pointer	$2FE20_H$	P0, 32	SFRE, P0, SV, 32	Kernel Reset	251
CPU5_HRB_BTV	CPU5 HR specific base trap vector table pointer	$2FE24_H$	P0, 32	SFRE, P0, SV, 32	Kernel Reset	555
CPU5_HRB_ISP	CPU5 HR specific interrupt stack pointer	$2FE28_H$	P0, 32	SFRE, P0, SV, 32	Kernel Reset	253
CPU5_HRB_ICR	CPU5 interrupt control register	$2FE2C_H$	P0, 32	P0, SV, 32	Kernel Reset	242

(table continues...)

Table 220 (continued) Register overview - CPU5 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU5_HRB_PPRS	CPU5 previous PRS	$2FE34_H$	P0, 32	P0, SV, 32	Kernel Reset	244
CPU5_HRB_FCX	CPU5 HR specific free CSA list head pointer	$2FE38_H$	P0, 32	P0, SV, 32	Kernel Reset	254
CPU5_HRB_LCX	CPU5 HR specific free CSA list limit pointer	$2FE3C_H$	P0, 32	P0, SV, 32	Kernel Reset	255
CPU5_HRB_SWID	CPU5 software identification	$2FE40_H$	P0, 32	P0, SV, 32	Kernel Reset	544
CPU5_HRB_CUS_ID	CPU5 customer ID register	$2FE50_H$	P0, 32	P0, SV, 32	Kernel Reset	546
CPU5_HRB_BOOTCON	CPU5 core boot configuration register	$2FE60_H$	P0, 32	P0, SV, 32	Kernel Reset	548
CPU5_HRB_LCLCON	CPU5 lockstep control register	$2FE64_H$	P0, 32	STP, SFRSE, P0, SV, 32	Cold PowerOn Reset	550
CPU5_HRB_CCON	CPU5 clock control register	$2FE68_H$	P0, 32	P0, SV, 32	Kernel Reset	248
CPU5_HRB_TCCON	CPU5 hardware configuration	$2FE6C_H$	P0, 32	P0, SV, 32	Kernel Reset	552
CPU5_HRB_Ay (y=0-1)	CPU5 HR specific address general purpose register y	$2FF80_H$+ y*4	P0, 32	P0, SV, 32	Kernel Reset	274
CPU5_HRB_Ay (y=8-9)	CPU5 HR specific address general purpose register y	$2FFA0_H$+ (y-8)*4	P0, 32	P0, SV, 32	Kernel Reset	274
CPU5_HRHV_SEGEN	CPU5 SRI error generation register	31030_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	255
CPU5_HRHV_LCLTEST	CPU5 lockstep test register	31040_H	P0, 32	P0, SV, 32	Cold PowerOn Reset	256
CPU5_HRHV_PMA0	CPU5 data access cacheability register	38100_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	232
CPU5_HRHV_PMA1	CPU5 code access cacheability register	38104_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	233
CPU5_HRHV_PMA2	CPU5 peripheral space identifier register	38108_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	234
CPU5_HRHV_DCON2	CPU5 data control register 2	39000_H	P0, 32	P0, SV, 32	Kernel Reset	558
CPU5_HRHV_DCON1	CPU5 data control register 1	39008_H	P0, 32	P0, SV, 32	Kernel Reset	258
CPU5_HRHV_SMACON	CPU5 SIST mode access control register	$3900C_H$	P0, 32	SFRSE, P0, SV, 32	Kernel Reset	234
CPU5_HRHV_DSTR	CPU5 HR specific data synchronous trap register	39010_H	P0, 32	P0, SV, 32	Kernel Reset	259

(table continues...)

Table 220 (continued) Register overview - CPU5 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU5_HRHV_DATR	CPU5 HR specific data asynchronous trap register	39018_H	P0, 32	P0, SV, 32	Kernel Reset	261
CPU5_HRHV_DEADD	CPU5 HR specific data error address register	$3901C_H$	P0, 32	P0, SV, 32	Kernel Reset	262
CPU5_HRHV_DIEAR	CPU5 data integrity error address register	39020_H	P0, 32	P0, SV, 32	Kernel Reset	279
CPU5_HRHV_DIETR	CPU5 data integrity error trap register	39024_H	P0, 32	P0, SV, 32	Kernel Reset	280
CPU5_HRHV_DCON0	CPU5 data memory control register	39040_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	258
CPU5_HRHV_PSTR	CPU5 HR specific program synchronous trap register	39200_H	P0, 32	P0, SV, 32	Kernel Reset	295
CPU5_HRHV_PCON1	CPU5 program control 1	39204_H	P0, 32	P0, SV, 32	Kernel Reset	292
CPU5_HRHV_PCON2	CPU5 program control 2	39208_H	P0, 32	P0, SV, 32	Kernel Reset	561
CPU5_HRHV_PCON0	CPU5 program control 0	$3920C_H$	P0, 32	SFRE, P0, SV, 32	Kernel Reset	294
CPU5_HRHV_PIEAR	CPU5 HR specific program integrity error address register	39210_H	P0, 32	P0, SV, 32	Kernel Reset	281
CPU5_HRHV_PIETR	CPU5 HR specific program integrity error trap register	39214_H	P0, 32	P0, SV, 32	Kernel Reset	282
CPU5_HRHV_COMPAT	CPU5 compatibility control register	39400_H	P0, 32	SFRSE, P0, SV, 32	Kernel Reset	235
CPU5_HRHV_FPU_TRAP_CON	CPU5 HR specific trap control register	$3A000_H$	P0, 32	P0, SV, 32	Kernel Reset	263
CPU5_HRHV_FPU_TRAP_PC	CPU5 HR specific trapping instruction program counter register	$3A004_H$	P0, 32	P0, SV, 32	Kernel Reset	265
CPU5_HRHV_FPU_TRAP_OPC	CPU5 HR specific trapping instruction opcode register	$3A008_H$	P0, 32	P0, SV, 32	Kernel Reset	266
CPU5_HRHV_FPU_TRAP_SRC1_L	CPU5 HR specific trapping instruction operand register	$3A010_H$	P0, 32	P0, SV, 32	Kernel Reset	267
CPU5_HRHV_FPU_TRAP_SRC1_U	CPU5 HR specific trapping instruction operand register	$3A014_H$	P0, 32	P0, SV, 32	Kernel Reset	268
CPU5_HRHV_FPU_TRAP_SRC2_L	CPU5 HR specific trapping instruction operand register	$3A018_H$	P0, 32	P0, SV, 32	Kernel Reset	268

(table continues...)

Table 220 (continued) Register overview - CPU5 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU5_HRHV_FPU_TRAP_SRC2_U	CPU5 HR specific trapping instruction operand register	$3A01C_H$	P0, 32	P0, SV, 32	Kernel Reset	269
CPU5_HRHV_FPU_TRAP_SRC3_L	CPU5 HR specific trapping instruction operand register	$3A020_H$	P0, 32	P0, SV, 32	Kernel Reset	270
CPU5_HRHV_FPU_TRAP_SRC3_U	CPU5 HR specific trapping instruction operand register	$3A024_H$	P0, 32	P0, SV, 32	Kernel Reset	270
CPU5_HRHV_FPU_SYNC_TRAP_CON	CPU5 HR specific synchronous trap control register	$3A030_H$	P0, 32	P0, SV, 32	Kernel Reset	271
CPU5_HRHV_FPU_SYNC_TRAP_OPC	CPU5 HR specific trapping instruction opcode register	$3A034_H$	P0, 32	P0, SV, 32	Kernel Reset	273
CPU5_HRHV_VCON0	CPU5 virtualization control register 0	$3B000_H$	P0, 32	P0, SV, 32	Kernel Reset	275
CPU5_HRHV_VCON1	CPU5 virtualization control register 1	$3B004_H$	P0, 32	P0, SV, 32	Kernel Reset	276
CPU5_HRHV_VCON2	CPU5 virtualization control register 2	$3B008_H$	P0, 32	P0, SV, 32	Kernel Reset	276
CPU5_HRHV_BHV	CPU5 base hypervisor vector table pointer	$3B010_H$	P0, 32	P0, SV, 32, SFRE	Kernel Reset	559
CPU5_HRHV_VMn_ICR (n=0-7)	CPU5 Virtual machine n interrupt state	$3B100_H$+n*4	P0, 32	P0, SV, 32	Kernel Reset	278
CPU5_HRHV_VMn_PETHRESH (n=0-7)	CPU5 Virtual machine n pre-emption threshold	$3B200_H$+n*4	P0, 32	P0, SV, 32	Kernel Reset	279
CPU5_HRHV_DPRy_L (y=0-23)	CPU5 HR specific data protection range y, lower bound register	$3C000_H$+y*8	P0, 32	P0, SV, 32	Kernel Reset	283
CPU5_HRHV_DPRy_U (y=0-23)	CPU5 HR specific data protection range y, upper bound register	$3C004_H$+y*8	P0, 32	P0, SV, 32	Kernel Reset	284
CPU5_HRHV_CPRy_L (y=0-15)	CPU5 HR specific code protection range y, lower bound register	$3D000_H$+y*8	P0, 32	P0, SV, 32	Kernel Reset	285
CPU5_HRHV_CPRy_U (y=0-15)	CPU5 HR specific code protection range y, upper bound register	$3D004_H$+y*8	P0, 32	P0, SV, 32	Kernel Reset	286

(table continues...)

Table 220 (continued) Register overview - CPU5 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU5_HRHV_CPXE_y (y=0-3)	CPU5 HR specific code protection execute enable register set y	$3E000_H$+y*4	P0, 32	P0, SV, 32	Kernel Reset	287
CPU5_HRHV_DPRE_y (y=0-3)	CPU5 HR specific data protection read enable register set y	$3E010_H$+y*4	P0, 32	P0, SV, 32	Kernel Reset	288
CPU5_HRHV_DPWE_y (y=0-3)	CPU5 HR specific data protection write enable register set y	$3E020_H$+y*4	P0, 32	P0, SV, 32	Kernel Reset	289
CPU5_HRHV_CPXE_y (y=4-7)	CPU5 HR specific code protection execute enable register set y	$3E040_H$+(y-4)*4	P0, 32	P0, SV, 32	Kernel Reset	287
CPU5_HRHV_DPRE_y (y=4-7)	CPU5 HR specific data protection read enable register set y	$3E050_H$+(y-4)*4	P0, 32	P0, SV, 32	Kernel Reset	288
CPU5_HRHV_DPWE_y (y=4-7)	CPU5 HR specific data protection write enable register set y	$3E060_H$+(y-4)*4	P0, 32	P0, SV, 32	Kernel Reset	289
CPU5_HRHV_TPS_CON	CPU5 HR specific temporal protection system control register	$3E400_H$	P0, 32	P0, SV, 32	Kernel Reset	296
CPU5_HRHV_TPS_TIMERn (n=0-2)	CPU5 HR specific temporal protection system timer register n	$3E404_H$+n*4	P0, 32	P0, SV, 32	Kernel Reset	297
CPU5_HRHV_TRnEVT (n=0-7)	CPU5 trigger event n	$3F000_H$+n*8	P0, 32	P0, SV, 32	Debug Reset	209
CPU5_HRHV_TRnADR (n=0-7)	CPU5 trigger address n	$3F004_H$+n*8	P0, 32	P0, SV, 32	Debug Reset	210
CPU5_HRHV_CCTRL	CPU5 HR specific counter control	$3FC00_H$	P0, 32	P0, SV, 32	Kernel Reset	226
CPU5_HRHV_CCNT	CPU5 HR specific CPU clock cycle count	$3FC04_H$	P0, 32	P0, SV, 32	Kernel Reset	227
CPU5_HRHV_ICNT	CPU5 HR specific instruction count	$3FC08_H$	P0, 32	P0, SV, 32	Kernel Reset	228
CPU5_HRHV_M1CNT	CPU5 HR specific multi-count register 1	$3FC0C_H$	P0, 32	P0, SV, 32	Kernel Reset	229
CPU5_HRHV_M2CNT	CPU5 HR specific multi-count register 2	$3FC10_H$	P0, 32	P0, SV, 32	Kernel Reset	230
CPU5_HRHV_M3CNT	CPU5 HR specific multi-count register 3	$3FC14_H$	P0, 32	P0, SV, 32	Kernel Reset	231

(table continues...)

Table 220 (continued) Register overview - CPU5 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU5_HRHV_DBGSR	CPU5 debug status register	$3FD00_H$	P0, 32	P0, SV, 32	Debug Reset	211
CPU5_HRHV_EXEVT	CPU5 external event register	$3FD08_H$	P0, 32	P0, SV, 32	Debug Reset	212
CPU5_HRHV_CREVT	CPU5 core register access event	$3FD0C_H$	P0, 32	P0, SV, 32	Debug Reset	213
CPU5_HRHV_SWEVT	CPU5 software debug event	$3FD10_H$	P0, 32	P0, SV, 32	Debug Reset	214
CPU5_HRHV_DBGACT	CPU5 debug action configuration	$3FD14_H$	P0, 32	P0, SV, 32	Debug Reset	215
CPU5_HRHV_TRIG_ACC	CPU5 trigger event accumulator	$3FD30_H$	P0, 32	P0, SV, 32	Debug Reset	217
CPU5_HRHV_DMS	CPU5 debug monitor start address	$3FD40_H$	P0, 32	P0, SV, 32	Debug Reset	534
CPU5_HRHV_DCX	CPU5 debug context save area pointer	$3FD44_H$	P0, 32	P0, SV, 32	Debug Reset	536
CPU5_HRHV_DBGTCR	CPU5 debug trap control	$3FD48_H$	P0, 32	P0, SV, 32	Application Reset	219
CPU5_HRHV_DBGCFG	CPU5 debug configuration	$3FD4C_H$	P0, 32	P0, SV, 32	Debug Reset	220
CPU5_HRHV_TRCCFG	CPU5 trace configuration	$3FD50_H$	P0, 32	P0, SV, 32	Debug Reset	221
CPU5_HRHV_TRCFILT	CPU5 trace filter	$3FD54_H$	P0, 32	P0, SV, 32	Debug Reset	222
CPU5_HRHV_TRCLIM	CPU5 trace bandwidth limiter	$3FD58_H$	P0, 32	P0, SV, 32	Debug Reset	224
CPU5_HRHV_TS16PTCCTRL	CPU5 MCDS performance counter control register	$3FD60_H$	P0, 32	P0, SV, 32	Debug Reset	225
CPU5_HRHV_PCXI	CPU5 previous context information register	$3FE00_H$	P0, 32	P0, SV, 32	Kernel Reset	236
CPU5_HRHV_PSW	CPU5 HR specific program status word	$3FE04_H$	P0, 32	P0, SV, 32	Kernel Reset	237
CPU5_HRHV_PC	CPU5 program counter	$3FE08_H$	P0, 32	P0, SV, 32	Kernel Reset	538
CPU5_HRHV_CORECON	CPU5 HR specific core configuration register	$3FE14_H$	P0, 32	SFRSE, P0, SV, 32	Kernel Reset	250
CPU5_HRHV_CPU_ID	CPU5 identification register	$3FE18_H$	P0, 32	P0, SV, 32	Kernel Reset	540
CPU5_HRHV_CORE_ID	CPU5 core identification register	$3FE1C_H$	P0, 32	P0, SV, 32	Kernel Reset	542
CPU5_HRHV_BIV	CPU5 HR specific base interrupt vector table pointer	$3FE20_H$	P0, 32	SFRE, P0, SV, 32	Kernel Reset	251

(table continues...)

Table 220 (continued) Register overview - CPU5 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU5_HRHV_BTV	CPU5 HR specific base trap vector table pointer	$3FE24_H$	P0, 32	SFRE, P0, SV, 32	Kernel Reset	557
CPU5_HRHV_ISP	CPU5 HR specific interrupt stack pointer	$3FE28_H$	P0, 32	SFRE, P0, SV, 32	Kernel Reset	253
CPU5_HRHV_ICR	CPU5 interrupt control register	$3FE2C_H$	P0, 32	P0, SV, 32	Kernel Reset	242
CPU5_HRHV_PPRS	CPU5 previous PRS	$3FE34_H$	P0, 32	P0, SV, 32	Kernel Reset	244
CPU5_HRHV_FCX	CPU5 HR specific free CSA list head pointer	$3FE38_H$	P0, 32	P0, SV, 32	Kernel Reset	254
CPU5_HRHV_LCX	CPU5 HR specific free CSA list limit pointer	$3FE3C_H$	P0, 32	P0, SV, 32	Kernel Reset	255
CPU5_HRHV_SWID	CPU5 software identification	$3FE40_H$	P0, 32	P0, SV, 32	Kernel Reset	544
CPU5_HRHV_CUS_ID	CPU5 customer ID register	$3FE50_H$	P0, 32	P0, SV, 32	Kernel Reset	546
CPU5_HRHV_BOOTCON	CPU5 core boot configuration register	$3FE60_H$	P0, 32	P0, SV, 32	Kernel Reset	548
CPU5_HRHV_LCLCON	CPU5 lockstep control register	$3FE64_H$	P0, 32	STP, SFRSE, P0, SV, 32	Cold PowerOn Reset	550
CPU5_HRHV_CCON	CPU5 clock control register	$3FE68_H$	P0, 32	P0, SV, 32	Kernel Reset	248
CPU5_HRHV_TCCON	CPU5 hardware configuration	$3FE6C_H$	P0, 32	P0, SV, 32	Kernel Reset	552
CPU5_HRHV_Ay (y=0-1)	CPU5 HR specific address general purpose register y	$3FF80_H$+ y*4	P0, 32	P0, SV, 32	Kernel Reset	274
CPU5_HRHV_Ay (y=8-9)	CPU5 HR specific address general purpose register y	$3FFA0_H$+ (y-8)*4	P0, 32	P0, SV, 32	Kernel Reset	274

4.10.4.1.11 Register overview - CPUCS domain SRI (ascending offset address)

Table 221 Register overview - CPUCS domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPUCS_STM_CLC	CPUcs system timer clock control register	00000_H	P0, 32	P0, SV, SFRE, 32	System Reset	351
CPUCS_STM_OCS	CPUcs OCDS control and status register	00004_H	P0, 32	P0, SV, 32, DEN	Debug Reset	352
CPUCS_STM_ABS	CPUcs system timer absolute timer	00020_H	(P0, 32)| (P0, 64)	P0, SV, 64, SFRSE	System Reset	353

(table continues...)

Table 221 (continued) Register overview - CPUCS domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPUCS_STM_CMPc (c=0-1)	CPUcs system timer compare register c	00120_H+c*4	P1, 32	P1, SV, 32	System Reset	354
CPUCS_STM_CMCON	CPUcs system timer compare match control register	00128_H	P1, 32	P1, SV, 32	System Reset	355
CPUCS_STM_ICR	CPUcs system timer interrupt control register	$0012C_H$	P1, 32	P1, SV, 32	System Reset	356
CPUCS_STM_ISCR	CPUcs system timer interrupt set/clear register	00130_H	P1, 32	P1, SV, 32	System Reset	357
CPUCS_STM_ISR	CPUcs system timer interrupt status register	00134_H	P1, 32	P1, SV, 32	System Reset	358
CPUCS_STM_RELTIM	CPUcs system timer relative timer	00138_H	(P1, 32)| (P1, 64)	P1, SV, 64	System Reset	359
CPUCS_KRST0	CPUcs reset register 0	$0D000_H$	P0, 32	RSTE, P0, SV, 32	Kernel Reset	297
CPUCS_KRST1	CPUcs reset register 1	$0D004_H$	P0, 32	RSTE, P0, SV, 32	Kernel Reset	298
CPUCS_KRSTCLR	CPUcs reset clear register	$0D008_H$	P0, 32	RSTE, P0, SV, 32	Kernel Reset	299
CPUCS_PROTSPRSE	CPUcs PROTSPR register safe endinit	$0E008_H$	32	SV, 32, PROT	Application Reset	562
CPUCS_ACCENSPRCFG_WRA	CPUcs shadow SPR, write access enable register A	$0E020_H$	32	SPRSE, SV, 32	See 565	565
CPUCS_ACCENSPRCFG_WRB	CPUcs shadow SPR, write access enable register B	$0E024_H$	32	SPRSE, SV, 32	See 566	566
CPUCS_ACCENSPRCFG_RDA	CPUcs shadow SPR, read access enable register A	$0E028_H$	32	SPRSE, SV, 32	See 567	567
CPUCS_ACCENSPRCFG_RDB	CPUcs shadow SPR, read access enable register B	$0E02C_H$	32	SPRSE, SV, 32	Application Reset	568
CPUCS_ACCENSPRCFG_VM	CPUcs shadow SPR, VM access enable register	$0E030_H$	32	SPRSE, SV, 32	Application Reset	310
CPUCS_ACCENSPRCFG_PRS	CPUcs shadow SPR, PRS access enable register	$0E034_H$	32	SPRSE, SV, 32	Application Reset	310
CPUCS_ACCENSPRCFG_RGNLA	CPUcs shadow SPR, region lower address register	$0E038_H$	32	SPRSE, SV, 32	Application Reset	311
CPUCS_ACCENSPRCFG_RGNUA	CPUcs shadow SPR, region upper address register	$0E03C_H$	32	SPRSE, SV, 32	Application Reset	312
CPUCS_PROTSFRE	CPUcs PROTSFR register endinit	$0E080_H$	32	SV, 32, PROT	Application Reset	569
CPUCS_PROTSFRSE	CPUcs PROTSFR register safe endinit	$0E088_H$	32	SV, 32, PROT	Application Reset	571

(table continues...)

Table 221 (continued) Register overview - CPUCS domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPUCS_ACCENSFRCFG_WRA	CPUcs shadow SFR, write access enable register A	$0E0A0_H$	32	SFRSE, SV, 32	See 573	573
CPUCS_ACCENSFRCFG_WRB	CPUcs shadow SFR, write access enable register B	$0E0A4_H$	32	SFRSE, SV, 32	See 574	574
CPUCS_ACCENSFRCFG_RDA	CPUcs shadow SFR, read access enable register A	$0E0A8_H$	32	SFRSE, SV, 32	See 575	575
CPUCS_ACCENSFRCFG_RDB	CPUcs shadow SFR, read access enable register B	$0E0AC_H$	32	SFRSE, SV, 32	See 576	576
CPUCS_ACCENSFRCFG_VM	CPUcs shadow SFR, VM access enable register	$0E0B0_H$	32	SFRSE, SV, 32	Application Reset	325
CPUCS_ACCENSFRCFG_PRS	CPUcs shadow SFR, PRS access enable register	$0E0B4_H$	32	SFRSE, SV, 32	Application Reset	325
CPUCS_PROTRSTE	CPUcs PROTRST register endinit	$0E0C0_H$	32	SV, 32, PROT	Application Reset	577
CPUCS_PROTSTMSE	CPUcs PROTSTM Register Safe Endinit	$0E0D8_H$	32	SV, 32, PROT	Application Reset	579
CPUCS_ACCENSTMCFG_WRA	CPUcs shadow STM, write access enable register A	$0E0E0_H$	32	STMSE, SV, 32	Application Reset	582
CPUCS_ACCENSTMCFG_WRB	CPUcs shadow STM, write access enable register B	$0E0E4_H$	32	STMSE, SV, 32	Application Reset	330
CPUCS_ACCENSTMCFG_RDA	CPUcs shadow STM, read access enable register A	$0E0E8_H$	32	STMSE, SV, 32	Application Reset	583
CPUCS_ACCENSTMCFG_RDB	CPUcs shadow STM, read access enable register B	$0E0EC_H$	32	STMSE, SV, 32	Application Reset	584
CPUCS_ACCENSTMCFG_VM	CPUcs shadow STM, VM access enable register	$0E0F0_H$	32	STMSE, SV, 32	Application Reset	332
CPUCS_ACCENSTMCFG_PRS	CPUcs shadow STM, PRS access enable register	$0E0F4_H$	32	STMSE, SV, 32	Application Reset	333
CPUCS_ACCENSPRy_WRA (y=0-15)	CPUcs SPR range y, write access enable register A	$0E200_H$+ y*20_H	32	SPRSE, SV, 32	See 585	585
CPUCS_ACCENSPRy_WRB (y=0-15)	CPUcs SPR range y, write access enable register B	$0E204_H$+ y*20_H	32	SPRSE, SV, 32	See 586	586
CPUCS_ACCENSPRy_RDA (y=0-15)	CPUcs SPR range y, read access enable register A	$0E208_H$+ y*20_H	32	SPRSE, SV, 32	See 587	587
CPUCS_ACCENSPRy_RDB (y=0-15)	CPUcs SPR range y, read access enable register B	$0E20C_H$+ y*20_H	32	SPRSE, SV, 32	Application Reset	588

(table continues...)

Table 221 (continued) Register overview - CPUCS domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPUCS_ACCENSPRy_VM (y=0-15)	CPUcs SPR range y, VM access enable register	$0E210_H$+ y*20_H	32	SPRSE, SV, 32	Application Reset	336
CPUCS_ACCENSPRy_PRS (y=0-15)	CPUcs SPR range y, PRS access enable register	$0E214_H$+ y*20_H	32	SPRSE, SV, 32	Application Reset	337
CPUCS_ACCENSPRy_RGNLA (y=0-15)	CPUcs SPR range y, region lower address register	$0E218_H$+ y*20_H	32	SPRSE, SV, 32	Application Reset	337
CPUCS_ACCENSPRy_RGNUA (y=0-15)	CPUcs SPR range y, region upper address register	$0E21C_H$+ y*20_H	32	SPRSE, SV, 32	Application Reset	338
CPUCS_ACCENSFR_WRA	CPUcs SFR, write access enable register A	$0E600_H$	32	SFRSE, SV, 32	See 589	589
CPUCS_ACCENSFR_WRB	CPUcs SFR, write access enable register B	$0E604_H$	32	SFRSE, SV, 32	See 590	590
CPUCS_ACCENSFR_RDA	CPUcs SFR, read access enable register A	$0E608_H$	32	SFRSE, SV, 32	See 591	591
CPUCS_ACCENSFR_RDB	CPUcs SFR, read access enable register B	$0E60C_H$	32	SFRSE, SV, 32	See 592	592
CPUCS_ACCENSFR_VM	CPUcs SFR, VM access enable register	$0E610_H$	32	SFRSE, SV, 32	Application Reset	345
CPUCS_ACCENSFR_PRS	CPUcs SFR, PRS access enable register	$0E614_H$	32	SFRSE, SV, 32	Application Reset	346
CPUCS_ACCENSTM_WRA	CPUcs STM, write access enable register A	$0E720_H$	32	STMSE, SV, 32	Application Reset	593
CPUCS_ACCENSTM_WRB	CPUcs STM, write access enable register B	$0E724_H$	32	STMSE, SV, 32	Application Reset	347
CPUCS_ACCENSTM_RDA	CPUcs STM, read access enable register A	$0E728_H$	32	STMSE, SV, 32	Application Reset	594
CPUCS_ACCENSTM_RDB	CPUcs STM, read access enable register B	$0E72C_H$	32	STMSE, SV, 32	Application Reset	595
CPUCS_ACCENSTM_VM	CPUcs STM, VM access enable register	$0E730_H$	32	STMSE, SV, 32	Application Reset	350
CPUCS_ACCENSTM_PRS	CPUcs STM, PRS access enable register	$0E734_H$	32	STMSE, SV, 32	Application Reset	350
CPUCS_HRA_SEGEN	CPUcs SRI error generation register	11030_H	P0, 32	SFRE, P0, SV, 32	Application Reset	255
CPUCS_HRA_LCLTEST	CPUcs lockstep test register	11040_H	P0, 32	P0, SV, 32	Cold PowerOn Reset	256
CPUCS_HRA_PMA0	CPUcs data access cacheability register	18100_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	232

(table continues...)

Table 221 (continued) Register overview - CPUCS domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPUCS_HRA_PMA1	CPUcs code access cacheability register	18104_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	233
CPUCS_HRA_PMA2	CPUcs peripheral space identifier register	18108_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	234
CPUCS_HRA_DCON2	CPUcs data control register 2	19000_H	P0, 32	P0, SV, 32	Kernel Reset	558
CPUCS_HRA_DCON1	CPUcs data control register 1	19008_H	P0, 32	P0, SV, 32	Kernel Reset	258
CPUCS_HRA_SMACON	CPUcs SIST mode access control register	$1900C_H$	P0, 32	SFRSE, P0, SV, 32	Kernel Reset	234
CPUCS_HRA_DSTR	CPUcs HR specific data synchronous trap register	19010_H	P0, 32	P0, SV, 32	Kernel Reset	259
CPUCS_HRA_DATR	CPUcs HR specific data asynchronous trap register	19018_H	P0, 32	P0, SV, 32	Kernel Reset	261
CPUCS_HRA_DEADD	CPUcs HR specific data error address register	$1901C_H$	P0, 32	P0, SV, 32	Kernel Reset	262
CPUCS_HRA_DIEAR	CPUcs data integrity error address register	19020_H	P0, 32	P0, SV, 32	Kernel Reset	279
CPUCS_HRA_DIETR	CPUcs data integrity error trap register	19024_H	P0, 32	P0, SV, 32	Kernel Reset	280
CPUCS_HRA_DCON0	CPUcs data memory control register	19040_H	P0, 32	SFRE, P0, SV, 32	Kernel Reset	258
CPUCS_HRA_PSTR	CPUcs HR specific program synchronous trap register	19200_H	P0, 32	P0, SV, 32	Kernel Reset	295
CPUCS_HRA_PCON1	CPUcs program control 1	19204_H	P0, 32	P0, SV, 32	Kernel Reset	292
CPUCS_HRA_PCON2	CPUcs program control 2	19208_H	P0, 32	P0, SV, 32	Kernel Reset	561
CPUCS_HRA_PCON0	CPUcs program control 0	$1920C_H$	P0, 32	SFRE, P0, SV, 32	Kernel Reset	294
CPUCS_HRA_PIEAR	CPUcs HR specific program integrity error address register	19210_H	P0, 32	P0, SV, 32	Kernel Reset	281
CPUCS_HRA_PIETR	CPUcs HR specific program integrity error trap register	19214_H	P0, 32	P0, SV, 32	Kernel Reset	282
CPUCS_HRA_COMPAT	CPUcs compatibility control register	19400_H	P0, 32	SFRSE, P0, SV, 32	Kernel Reset	235
CPUCS_HRA_FPU_TRAP_CON	CPUcs HR specific trap control register	$1A000_H$	P0, 32	P0, SV, 32	Kernel Reset	263
CPUCS_HRA_FPU_TRAP_PC	CPUcs HR specific trapping instruction program counter register	$1A004_H$	P0, 32	P0, SV, 32	Kernel Reset	265

(table continues...)

Table 221 (continued) Register overview - CPUCS domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPUCS_HRA_FPU_TRAP_OPC	CPUcs HR specific trapping instruction opcode register	$1A008_H$	P0, 32	P0, SV, 32	Kernel Reset	266
CPUCS_HRA_FPU_TRAP_SRC1_L	CPUcs HR specific trapping instruction operand register	$1A010_H$	P0, 32	P0, SV, 32	Kernel Reset	267
CPUCS_HRA_FPU_TRAP_SRC1_U	CPUcs HR specific trapping instruction operand register	$1A014_H$	P0, 32	P0, SV, 32	Kernel Reset	268
CPUCS_HRA_FPU_TRAP_SRC2_L	CPUcs HR specific trapping instruction operand register	$1A018_H$	P0, 32	P0, SV, 32	Kernel Reset	268
CPUCS_HRA_FPU_TRAP_SRC2_U	CPUcs HR specific trapping instruction operand register	$1A01C_H$	P0, 32	P0, SV, 32	Kernel Reset	269
CPUCS_HRA_FPU_TRAP_SRC3_L	CPUcs HR specific trapping instruction operand register	$1A020_H$	P0, 32	P0, SV, 32	Kernel Reset	270
CPUCS_HRA_FPU_TRAP_SRC3_U	CPUcs HR specific trapping instruction operand register	$1A024_H$	P0, 32	P0, SV, 32	Kernel Reset	270
CPUCS_HRA_FPU_SYNC_TRAP_CON	CPUcs HR specific synchronous trap control register	$1A030_H$	P0, 32	P0, SV, 32	Kernel Reset	271
CPUCS_HRA_FPU_SYNC_TRAP_OPC	CPUcs HR specific trapping instruction opcode register	$1A034_H$	P0, 32	P0, SV, 32	Kernel Reset	273
CPUCS_HRA_DPRy_L (y=0-23)	CPUcs HR specific data protection range y, lower bound register	$1C000_H$+ y*8	P0, 32	P0, SV, 32	Kernel Reset	283
CPUCS_HRA_DPRy_U (y=0-23)	CPUcs HR specific data protection range y, upper bound register	$1C004_H$+ y*8	P0, 32	P0, SV, 32	Kernel Reset	284
CPUCS_HRA_CPRy_L (y=0-15)	CPUcs HR specific code protection range y, lower bound register	$1D000_H$+ y*8	P0, 32	P0, SV, 32	Kernel Reset	285
CPUCS_HRA_CPRy_U (y=0-15)	CPUcs HR specific code protection range y, upper bound register	$1D004_H$+ y*8	P0, 32	P0, SV, 32	Kernel Reset	286
CPUCS_HRA_CPXE_y (y=0-3)	CPUcs HR specific code protection execute enable register set y	$1E000_H$+ y*4	P0, 32	P0, SV, 32	Kernel Reset	287

(table continues...)

Table 221 (continued) Register overview - CPUCS domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPUCS_HRA_DPRE_y (y=0-3)	CPUcs HR specific data protection read enable register set y	$1E010_H$+y*4	P0, 32	P0, SV, 32	Kernel Reset	288
CPUCS_HRA_DPWE_y (y=0-3)	CPUcs HR specific data protection write enable register set y	$1E020_H$+y*4	P0, 32	P0, SV, 32	Kernel Reset	289
CPUCS_HRA_CPXE_y (y=4-7)	CPUcs HR specific code protection execute enable register set y	$1E040_H$+(y-4)*4	P0, 32	P0, SV, 32	Kernel Reset	287
CPUCS_HRA_DPRE_y (y=4-7)	CPUcs HR specific data protection read enable register set y	$1E050_H$+(y-4)*4	P0, 32	P0, SV, 32	Kernel Reset	288
CPUCS_HRA_DPWE_y (y=4-7)	CPUcs HR specific data protection write enable register set y	$1E060_H$+(y-4)*4	P0, 32	P0, SV, 32	Kernel Reset	289
CPUCS_HRA_TPS_CON	CPUcs HR specific temporal protection system control register	$1E400_H$	P0, 32	P0, SV, 32	Kernel Reset	296
CPUCS_HRA_TPS_TIMERn (n=0-2)	CPUcs HR specific temporal protection system timer register n	$1E404_H$+n*4	P0, 32	P0, SV, 32	Kernel Reset	297
CPUCS_HRA_TRnEVT (n=0-7)	CPUcs trigger event n	$1F000_H$+n*8	P0, 32	P0, SV, 32	Debug Reset	209
CPUCS_HRA_TRnADR (n=0-7)	CPUcs trigger address n	$1F004_H$+n*8	P0, 32	P0, SV, 32	Debug Reset	210
CPUCS_HRA_CCTRL	CPUcs HR specific counter control	$1FC00_H$	P0, 32	P0, SV, 32	Kernel Reset	226
CPUCS_HRA_CCNT	CPUcs HR specific CPU clock cycle count	$1FC04_H$	P0, 32	P0, SV, 32	Kernel Reset	227
CPUCS_HRA_ICNT	CPUcs HR specific instruction count	$1FC08_H$	P0, 32	P0, SV, 32	Kernel Reset	228
CPUCS_HRA_M1CNT	CPUcs HR specific multi-count register 1	$1FC0C_H$	P0, 32	P0, SV, 32	Kernel Reset	229
CPUCS_HRA_M2CNT	CPUcs HR specific multi-count register 2	$1FC10_H$	P0, 32	P0, SV, 32	Kernel Reset	230
CPUCS_HRA_M3CNT	CPUcs HR specific multi-count register 3	$1FC14_H$	P0, 32	P0, SV, 32	Kernel Reset	231
CPUCS_HRA_DBGSR	CPUcs debug status register	$1FD00_H$	P0, 32	P0, SV, 32	Debug Reset	211

(table continues...)

Table 221 (continued) Register overview - CPUCS domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPUCS_HRA_EXEVT	CPUcs external event register	$1FD08_H$	P0, 32	P0, SV, 32	Debug Reset	212
CPUCS_HRA_CREVT	CPUcs core register access event	$1FD0C_H$	P0, 32	P0, SV, 32	Debug Reset	213
CPUCS_HRA_SWEVT	CPUcs software debug event	$1FD10_H$	P0, 32	P0, SV, 32	Debug Reset	214
CPUCS_HRA_DBGACT	CPUcs debug action configuration	$1FD14_H$	P0, 32	P0, SV, 32	Debug Reset	215
CPUCS_HRA_TRIG_ACC	CPUcs trigger event accumulator	$1FD30_H$	P0, 32	P0, SV, 32	Debug Reset	217
CPUCS_HRA_DMS	CPUcs debug monitor start address	$1FD40_H$	P0, 32	P0, SV, 32	Debug Reset	534
CPUCS_HRA_DCX	CPUcs debug context save area pointer	$1FD44_H$	P0, 32	P0, SV, 32	Debug Reset	536
CPUCS_HRA_DBGTCR	CPUcs debug trap control	$1FD48_H$	P0, 32	P0, SV, 32	Application Reset	219
CPUCS_HRA_DBGCFG	CPUcs debug configuration	$1FD4C_H$	P0, 32	P0, SV, 32	Debug Reset	220
CPUCS_HRA_TRCCFG	CPUcs trace configuration	$1FD50_H$	P0, 32	P0, SV, 32	Debug Reset	221
CPUCS_HRA_TRCFILT	CPUcs trace filter	$1FD54_H$	P0, 32	P0, SV, 32	Debug Reset	222
CPUCS_HRA_TRCLIM	CPUcs trace bandwidth limiter	$1FD58_H$	P0, 32	P0, SV, 32	Debug Reset	224
CPUCS_HRA_TS16PTCCTRL	CPUcs MCDS performance counter control register	$1FD60_H$	P0, 32	P0, SV, 32	Debug Reset	225
CPUCS_HRA_PCXI	CPUcs previous context information register	$1FE00_H$	P0, 32	P0, SV, 32	Kernel Reset	236
CPUCS_HRA_PSW	CPUcs HR specific program status word	$1FE04_H$	P0, 32	P0, SV, 32	Kernel Reset	237
CPUCS_HRA_PC	CPUcs program counter	$1FE08_H$	P0, 32	P0, SV, 32	Kernel Reset	538
CPUCS_HRA_CORECON	CPUcs HR specific core configuration register	$1FE14_H$	P0, 32	SFRSE, P0, SV, 32	Kernel Reset	250
CPUCS_HRA_CPU_ID	CPUcs identification register	$1FE18_H$	P0, 32	P0, SV, 32	Kernel Reset	540
CPUCS_HRA_CORE_ID	CPUcs core identification register	$1FE1C_H$	P0, 32	P0, SV, 32	Kernel Reset	542
CPUCS_HRA_BIV	CPUcs HR specific base interrupt vector table pointer	$1FE20_H$	P0, 32	SFRE, P0, SV, 32	Kernel Reset	251
CPUCS_HRA_BTV	CPUcs HR specific base trap vector table pointer	$1FE24_H$	P0, 32	SFRE, P0, SV, 32	Kernel Reset	554

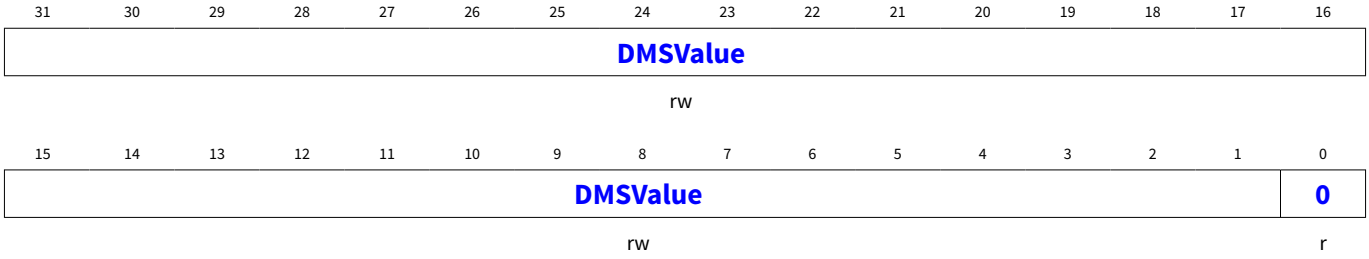
(table continues...)

Table 221 (continued) Register overview - CPUCS domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPUCS_HRA_ISP	CPUcs HR specific interrupt stack pointer	$1FE28_H$	P0, 32	SFRE, P0, SV, 32	Kernel Reset	253
CPUCS_HRA_ICR	CPUcs interrupt control register	$1FE2C_H$	P0, 32	P0, SV, 32	Kernel Reset	242
CPUCS_HRA_PPRS	CPUcs previous PRS	$1FE34_H$	P0, 32	P0, SV, 32	Kernel Reset	244
CPUCS_HRA_FCX	CPUcs HR specific free CSA list head pointer	$1FE38_H$	P0, 32	P0, SV, 32	Kernel Reset	254
CPUCS_HRA_LCX	CPUcs HR specific free CSA list limit pointer	$1FE3C_H$	P0, 32	P0, SV, 32	Kernel Reset	255
CPUCS_HRA_SWID	CPUcs software identification	$1FE40_H$	P0, 32	P0, SV, 32	Kernel Reset	544
CPUCS_HRA_CUS_ID	CPUcs customer ID register	$1FE50_H$	P0, 32	P0, SV, 32	Kernel Reset	546
CPUCS_HRA_BOOTCON	CPUcs core boot configuration register	$1FE60_H$	P0, 32	P0, SV, 32	Kernel Reset	548
CPUCS_HRA_LCLCON	CPUcs lockstep control register	$1FE64_H$	P0, 32	STP, SFRSE, P0, SV, 32	Cold PowerOn Reset	550
CPUCS_HRA_CCON	CPUcs clock control register	$1FE68_H$	P0, 32	P0, SV, 32	Kernel Reset	248
CPUCS_HRA_TCCON	CPUcs hardware configuration	$1FE6C_H$	P0, 32	P0, SV, 32	Kernel Reset	552
CPUCS_Dy (y=0-15)	CPUcs data general purpose register y	$1FF00_H$+ y*4	P0, 32	P0, SV, 32	Kernel Reset	274
CPUCS_Ay (y=0-15)	CPUcs address general purpose register y	$1FF80_H$+ y*4	P0, 32	P0, SV, 32	Kernel Reset	274

4.10.4.1.12 CPU0 debug monitor start address

CPU0_HRA_DMS	Offset address:	$1FD40_H$
CPU0 debug monitor start address	Debug Reset value:	$A000\ 0200_H$
CPU1_HRA_DMS	Offset address:	$1FD40_H$
CPU1 debug monitor start address	Debug Reset value:	$A000\ 0240_H$
CPU2_HRA_DMS	Offset address:	$1FD40_H$
CPU2 debug monitor start address	Debug Reset value:	$A000\ 0280_H$
CPU3_HRA_DMS	Offset address:	$1FD40_H$
CPU3 debug monitor start address	Debug Reset value:	$A000\ 02C0_H$
CPU4_HRA_DMS	Offset address:	$1FD40_H$
CPU4 debug monitor start address	Debug Reset value:	$A000\ 0300_H$
CPU5_HRA_DMS	Offset address:	$1FD40_H$
CPU5 debug monitor start address	Debug Reset value:	$A000\ 0340_H$
CPUCS_HRA_DMS	Offset address:	$1FD40_H$
CPUcs debug monitor start address	Debug Reset value:	$A000\ 0380_H$
CPU_DMS	Offset address:	$0FD40_H$
Short address for domain CSFR	Reset values see:	Table 222



Field	Bits	Type	Description
DMSValue	31:1	rw	**Debug monitor start address** The address at which monitor code execution begins when a breakpoint trap is taken.
0	0	r	**Reserved - RES** Read as 0; should be written with 0.

Table 222 Reset values of CPU_DMS

Reset type	Reset value	Note
Debug Reset	$A000\ 0200_H$	Instance specific: CPU0
Debug Reset	$A000\ 0240_H$	Instance specific: CPU1
Debug Reset	$A000\ 0280_H$	Instance specific: CPU2
Debug Reset	$A000\ 02C0_H$	Instance specific: CPU3
Debug Reset	$A000\ 0300_H$	Instance specific: CPU4
Debug Reset	$A000\ 0340_H$	Instance specific: CPU5

(table continues...)

Table 222 (continued) Reset values of CPU_DMS

Reset type	Reset value	Note
Debug Reset	A000 0380$_H$	Instance specific: CPUcs

Table 223 Mirrors of CPU0_HRA_DMS

Short name	Long name	Offset address
CPU0_HRB_DMS	CPU0 debug monitor start address is an alias.	2FD40$_H$
CPU0_HRHV_DMS	CPU0 debug monitor start address is an alias.	3FD40$_H$

Table 224 Mirrors of CPU1_HRA_DMS

Short name	Long name	Offset address
CPU1_HRB_DMS	CPU1 debug monitor start address is an alias.	2FD40$_H$
CPU1_HRHV_DMS	CPU1 debug monitor start address is an alias.	3FD40$_H$

Table 225 Mirrors of CPU2_HRA_DMS

Short name	Long name	Offset address
CPU2_HRB_DMS	CPU2 debug monitor start address is an alias.	2FD40$_H$
CPU2_HRHV_DMS	CPU2 debug monitor start address is an alias.	3FD40$_H$

Table 226 Mirrors of CPU3_HRA_DMS

Short name	Long name	Offset address
CPU3_HRB_DMS	CPU3 debug monitor start address is an alias.	2FD40$_H$
CPU3_HRHV_DMS	CPU3 debug monitor start address is an alias.	3FD40$_H$

Table 227 Mirrors of CPU4_HRA_DMS

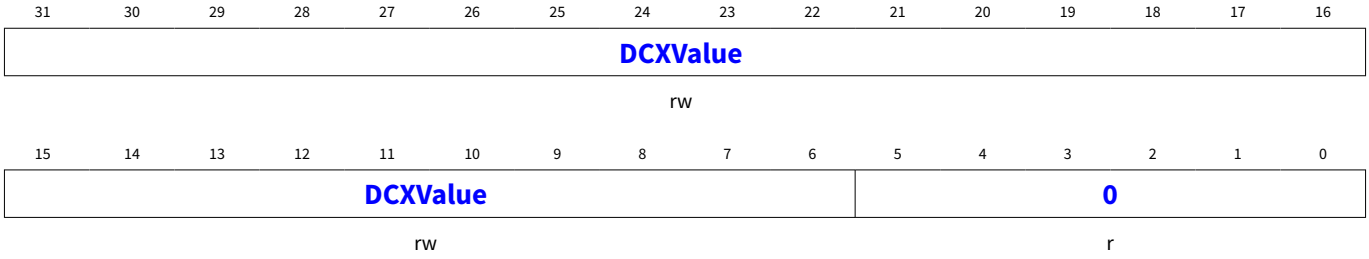
Short name	Long name	Offset address
CPU4_HRB_DMS	CPU4 debug monitor start address is an alias.	2FD40$_H$
CPU4_HRHV_DMS	CPU4 debug monitor start address is an alias.	3FD40$_H$

Table 228 Mirrors of CPU5_HRA_DMS

Short name	Long name	Offset address
CPU5_HRB_DMS	CPU5 debug monitor start address is an alias.	2FD40$_H$
CPU5_HRHV_DMS	CPU5 debug monitor start address is an alias.	3FD40$_H$

4.10.4.1.13 CPU0 debug context save area pointer

CPU0_HRA_DCX	Offset address:	$1FD44_H$
CPU0 debug context save area pointer	Debug Reset value:	$A000\ 0400_H$
CPU1_HRA_DCX	Offset address:	$1FD44_H$
CPU1 debug context save area pointer	Debug Reset value:	$A000\ 0440_H$
CPU2_HRA_DCX	Offset address:	$1FD44_H$
CPU2 debug context save area pointer	Debug Reset value:	$A000\ 0480_H$
CPU3_HRA_DCX	Offset address:	$1FD44_H$
CPU3 debug context save area pointer	Debug Reset value:	$A000\ 04C0_H$
CPU4_HRA_DCX	Offset address:	$1FD44_H$
CPU4 debug context save area pointer	Debug Reset value:	$A000\ 0500_H$
CPU5_HRA_DCX	Offset address:	$1FD44_H$
CPU5 debug context save area pointer	Debug Reset value:	$A000\ 0540_H$
CPUCS_HRA_DCX	Offset address:	$1FD44_H$
CPUcs debug context save area pointer	Debug Reset value:	$A000\ 0580_H$
CPU_DCX	Offset address:	$0FD44_H$
Short address for domain CSFR	Reset values see:	Table 229



Field	Bits	Type	Description
DCXValue	31:6	rw	**Debug context save area pointer** Address where the debug context is stored following a breakpoint trap.
0	5:0	r	**Reserved - RES** Read as 0; should be written with 0.

Table 229 Reset values of CPU_DCX

Reset type	Reset value	Note
Debug Reset	$A000\ 0400_H$	Instance specific: CPU0
Debug Reset	$A000\ 0440_H$	Instance specific: CPU1
Debug Reset	$A000\ 0480_H$	Instance specific: CPU2
Debug Reset	$A000\ 04C0_H$	Instance specific: CPU3
Debug Reset	$A000\ 0500_H$	Instance specific: CPU4
Debug Reset	$A000\ 0540_H$	Instance specific: CPU5
Debug Reset	$A000\ 0580_H$	Instance specific: CPUcs

Table 230 Mirrors of CPU0_HRA_DCX

Short name	Long name	Offset address
CPU0_HRB_DCX	CPU0 debug context save area pointer is an alias.	$2FD44_H$
CPU0_HRHV_DCX	CPU0 debug context save area pointer is an alias.	$3FD44_H$

Table 231 Mirrors of CPU1_HRA_DCX

Short name	Long name	Offset address
CPU1_HRB_DCX	CPU1 debug context save area pointer is an alias.	$2FD44_H$
CPU1_HRHV_DCX	CPU1 debug context save area pointer is an alias.	$3FD44_H$

Table 232 Mirrors of CPU2_HRA_DCX

Short name	Long name	Offset address
CPU2_HRB_DCX	CPU2 debug context save area pointer is an alias.	$2FD44_H$
CPU2_HRHV_DCX	CPU2 debug context save area pointer is an alias.	$3FD44_H$

Table 233 Mirrors of CPU3_HRA_DCX

Short name	Long name	Offset address
CPU3_HRB_DCX	CPU3 debug context save area pointer is an alias.	$2FD44_H$
CPU3_HRHV_DCX	CPU3 debug context save area pointer is an alias.	$3FD44_H$

Table 234 Mirrors of CPU4_HRA_DCX

Short name	Long name	Offset address
CPU4_HRB_DCX	CPU4 debug context save area pointer is an alias.	$2FD44_H$
CPU4_HRHV_DCX	CPU4 debug context save area pointer is an alias.	$3FD44_H$

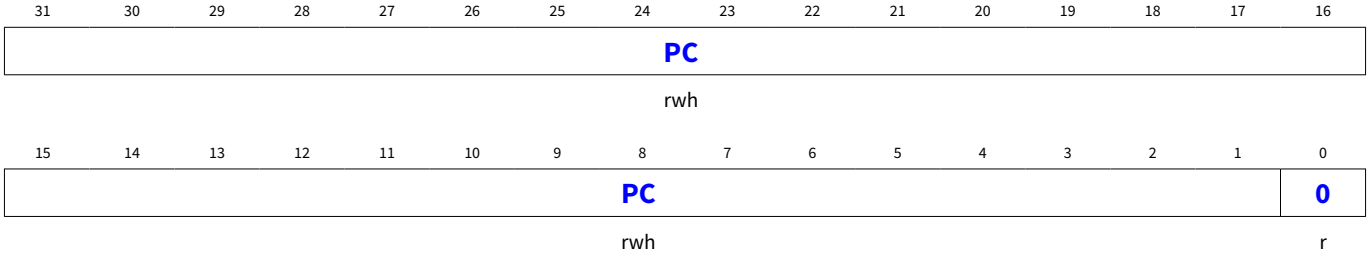
Table 235 Mirrors of CPU5_HRA_DCX

Short name	Long name	Offset address
CPU5_HRB_DCX	CPU5 debug context save area pointer is an alias.	$2FD44_H$
CPU5_HRHV_DCX	CPU5 debug context save area pointer is an alias.	$3FD44_H$

4.10.4.1.14 CPU0 program counter

The 32-bit Program Counter (PC) shown below, holds the address of the instruction that is currently running. The Program Counter is part of a task's state information. The PC should only be written via the Core Debug Controller when the core is halted. If the core is not in halt a write will have no effect.

CPU0_HRA_PC	Offset address:	$1FE08_H$
CPU0 program counter	Kernel Reset value:	$AFFF\ FF00_H$
CPU1_HRA_PC	Offset address:	$1FE08_H$
CPU1 program counter	Kernel Reset value:	$0000\ 0000_H$
CPU2_HRA_PC	Offset address:	$1FE08_H$
CPU2 program counter	Kernel Reset value:	$0000\ 0000_H$
CPU3_HRA_PC	Offset address:	$1FE08_H$
CPU3 program counter	Kernel Reset value:	$0000\ 0000_H$
CPU4_HRA_PC	Offset address:	$1FE08_H$
CPU4 program counter	Kernel Reset value:	$0000\ 0000_H$
CPU5_HRA_PC	Offset address:	$1FE08_H$
CPU5 program counter	Kernel Reset value:	$0000\ 0000_H$
CPUCS_HRA_PC	Offset address:	$1FE08_H$
CPUcs program counter	Kernel Reset value:	$AFFE\ FF00_H$
CPU_PC	Offset address:	$0FE08_H$
Short address for domain CSFR	Reset values see:	Table 236



Field	Bits	Type	Description
PC	31:1	rwh	**Program counter**
0	0	r	**Reserved - RES** Read as 0; should be written as 0.

Table 236 Reset values of CPU_PC

Reset type	Reset value	Note
Kernel Reset	$AFFF\ FF00_H$	Instance specific: CPU0
Kernel Reset	$AFFE\ FF00_H$	Instance specific: CPUcs
Kernel Reset	$0000\ 0000_H$	Instance specific: Any other CPU

Table 237 Mirrors of CPU0_HRA_PC

Short name	Long name	Offset address
CPU0_HRB_PC	CPU0 program counter is an alias.	$2FE08_H$
CPU0_HRHV_PC	CPU0 program counter is an alias.	$3FE08_H$

Table 238 Mirrors of CPU1_HRA_PC

Short name	Long name	Offset address
CPU1_HRB_PC	CPU1 program counter is an alias.	$2FE08_H$
CPU1_HRHV_PC	CPU1 program counter is an alias.	$3FE08_H$

Table 239 Mirrors of CPU2_HRA_PC

Short name	Long name	Offset address
CPU2_HRB_PC	CPU2 program counter is an alias.	$2FE08_H$
CPU2_HRHV_PC	CPU2 program counter is an alias.	$3FE08_H$

Table 240 Mirrors of CPU3_HRA_PC

Short name	Long name	Offset address
CPU3_HRB_PC	CPU3 program counter is an alias.	$2FE08_H$
CPU3_HRHV_PC	CPU3 program counter is an alias.	$3FE08_H$

Table 241 Mirrors of CPU4_HRA_PC

Short name	Long name	Offset address
CPU4_HRB_PC	CPU4 program counter is an alias.	$2FE08_H$
CPU4_HRHV_PC	CPU4 program counter is an alias.	$3FE08_H$

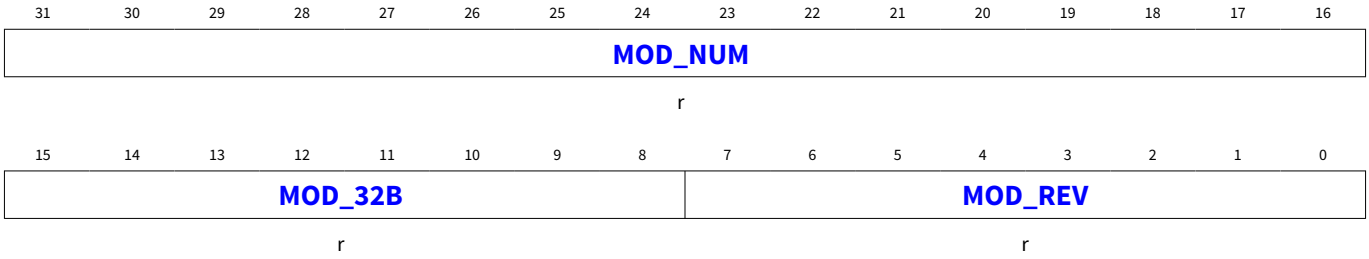
Table 242 Mirrors of CPU5_HRA_PC

Short name	Long name	Offset address
CPU5_HRB_PC	CPU5 program counter is an alias.	$2FE08_H$
CPU5_HRHV_PC	CPU5 program counter is an alias.	$3FE08_H$

4.10.4.1.15 CPU0 identification register

The CPU Identification Register identifies the CPU type and revision.

CPU0_HRA_CPU_ID	Offset address:	$1FE18_H$
CPU0 identification register	Kernel Reset value:	$00C0\ C031_H$
CPU1_HRA_CPU_ID	Offset address:	$1FE18_H$
CPU1 identification register	Kernel Reset value:	$00C0\ C031_H$
CPU2_HRA_CPU_ID	Offset address:	$1FE18_H$
CPU2 identification register	Kernel Reset value:	$00C0\ C031_H$
CPU3_HRA_CPU_ID	Offset address:	$1FE18_H$
CPU3 identification register	Kernel Reset value:	$00C0\ C031_H$
CPU4_HRA_CPU_ID	Offset address:	$1FE18_H$
CPU4 identification register	Kernel Reset value:	$00C0\ C031_H$
CPU5_HRA_CPU_ID	Offset address:	$1FE18_H$
CPU5 identification register	Kernel Reset value:	$00C0\ C031_H$
CPUCS_HRA_CPU_ID	Offset address:	$1FE18_H$
CPUcs identification register	Kernel Reset value:	$00C0\ C031_H$
CPU_CPU_ID	Offset address:	$0FE18_H$
Short address for domain CSFR	Kernel Reset value:	$00C0\ C031_H$



Field	Bits	Type	Description
MOD_REV	7:0	r	**Revision number**
MOD_32B	15:8	r	**Module type** A value of $C0_H$ in this field indicates a 32-bit module with a 32-bit module ID register
MOD_NUM	31:16	r	**Module identification number**

Table 243 Mirrors of CPU0_HRA_CPU_ID

Short name	Long name	Offset address
CPU0_HRB_CPU_ID	CPU0 identification register is an alias.	$2FE18_H$
CPU0_HRHV_CPU_ID	CPU0 identification register is an alias.	$3FE18_H$

Table 244 Mirrors of CPU1_HRA_CPU_ID

Short name	Long name	Offset address
CPU1_HRB_CPU_ID	CPU1 identification register is an alias.	$2FE18_H$
CPU1_HRHV_CPU_ID	CPU1 identification register is an alias.	$3FE18_H$

Table 245 Mirrors of CPU2_HRA_CPU_ID

Short name	Long name	Offset address
CPU2_HRB_CPU_ID	CPU2 identification register is an alias.	$2FE18_H$
CPU2_HRHV_CPU_ID	CPU2 identification register is an alias.	$3FE18_H$

Table 246 Mirrors of CPU3_HRA_CPU_ID

Short name	Long name	Offset address
CPU3_HRB_CPU_ID	CPU3 identification register is an alias.	$2FE18_H$
CPU3_HRHV_CPU_ID	CPU3 identification register is an alias.	$3FE18_H$

Table 247 Mirrors of CPU4_HRA_CPU_ID

Short name	Long name	Offset address
CPU4_HRB_CPU_ID	CPU4 identification register is an alias.	$2FE18_H$
CPU4_HRHV_CPU_ID	CPU4 identification register is an alias.	$3FE18_H$

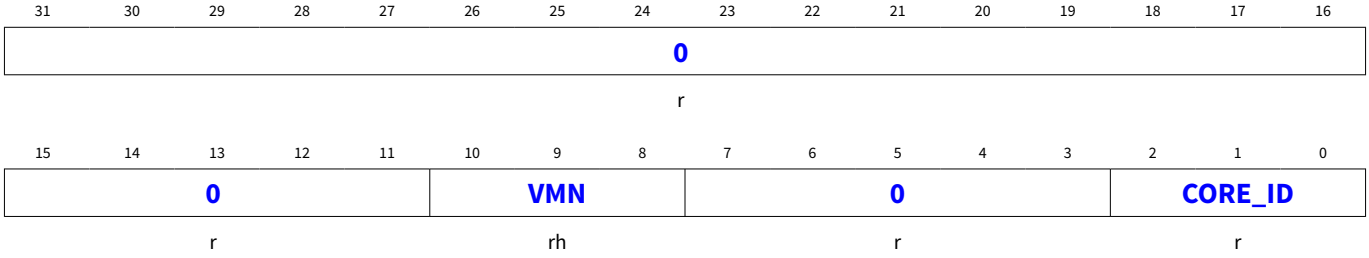
Table 248 Mirrors of CPU5_HRA_CPU_ID

Short name	Long name	Offset address
CPU5_HRB_CPU_ID	CPU5 identification register is an alias.	$2FE18_H$
CPU5_HRHV_CPU_ID	CPU5 identification register is an alias.	$3FE18_H$

4.10.4.1.16 CPU0 core identification register

In a multiprocessor system each logical processor core is given a unique identification number. In a system supporting virtualization each logical processor core can be used to run multiple virtual machines. The Core Identification Register holds both the core identification number and the current virtual machine number.

CPU0_HRA_CORE_ID	Offset address:	$1FE1C_H$
CPU0 core identification register	Kernel Reset value:	$0000\ 0000_H$
CPU1_HRA_CORE_ID	Offset address:	$1FE1C_H$
CPU1 core identification register	Kernel Reset value:	$0000\ 0001_H$
CPU2_HRA_CORE_ID	Offset address:	$1FE1C_H$
CPU2 core identification register	Kernel Reset value:	$0000\ 0002_H$
CPU3_HRA_CORE_ID	Offset address:	$1FE1C_H$
CPU3 core identification register	Kernel Reset value:	$0000\ 0003_H$
CPU4_HRA_CORE_ID	Offset address:	$1FE1C_H$
CPU4 core identification register	Kernel Reset value:	$0000\ 0004_H$
CPU5_HRA_CORE_ID	Offset address:	$1FE1C_H$
CPU5 core identification register	Kernel Reset value:	$0000\ 0005_H$
CPUCS_HRA_CORE_ID	Offset address:	$1FE1C_H$
CPUcs core identification register	Kernel Reset value:	$0000\ 0006_H$
CPU_CORE_ID	Offset address:	$0FE1C_H$
Short address for domain CSFR	Reset values see:	Table 249



Field	Bits	Type	Description
CORE_ID	2:0	r	**Core identification number** The identification number of the core.
VMN	10:8	rh	**Virutal machine number** The current virutal machine number. When virtualization is disabled or not present this field will read as 1.
0	7:3, 31:11	r	**Reserved - RES** Read as 0.

Table 249 Reset values of CPU_CORE_ID

Reset type	Reset value	Note
Kernel Reset	$0000\ 0000_H$	Instance specific: CPU0
Kernel Reset	$0000\ 0001_H$	Instance specific: CPU1

(table continues...)

Table 249 (continued) Reset values of CPU_CORE_ID

Reset type	Reset value	Note
Kernel Reset	$0000\ 0002_H$	Instance specific: CPU2
Kernel Reset	$0000\ 0003_H$	Instance specific: CPU3
Kernel Reset	$0000\ 0004_H$	Instance specific: CPU4
Kernel Reset	$0000\ 0005_H$	Instance specific: CPU5
Kernel Reset	$0000\ 0006_H$	Instance specific: CPUcs

Table 250 Mirrors of CPU0_HRA_CORE_ID

Short name	Long name	Offset address
CPU0_HRB_CORE_ID	CPU0 core identification register is an alias.	$2FE1C_H$
CPU0_HRHV_CORE_ID	CPU0 core identification register is an alias.	$3FE1C_H$

Table 251 Mirrors of CPU1_HRA_CORE_ID

Short name	Long name	Offset address
CPU1_HRB_CORE_ID	CPU1 core identification register is an alias.	$2FE1C_H$
CPU1_HRHV_CORE_ID	CPU1 core identification register is an alias.	$3FE1C_H$

Table 252 Mirrors of CPU2_HRA_CORE_ID

Short name	Long name	Offset address
CPU2_HRB_CORE_ID	CPU2 core identification register is an alias.	$2FE1C_H$
CPU2_HRHV_CORE_ID	CPU2 core identification register is an alias.	$3FE1C_H$

Table 253 Mirrors of CPU3_HRA_CORE_ID

Short name	Long name	Offset address
CPU3_HRB_CORE_ID	CPU3 core identification register is an alias.	$2FE1C_H$
CPU3_HRHV_CORE_ID	CPU3 core identification register is an alias.	$3FE1C_H$

Table 254 Mirrors of CPU4_HRA_CORE_ID

Short name	Long name	Offset address
CPU4_HRB_CORE_ID	CPU4 core identification register is an alias.	$2FE1C_H$
CPU4_HRHV_CORE_ID	CPU4 core identification register is an alias.	$3FE1C_H$

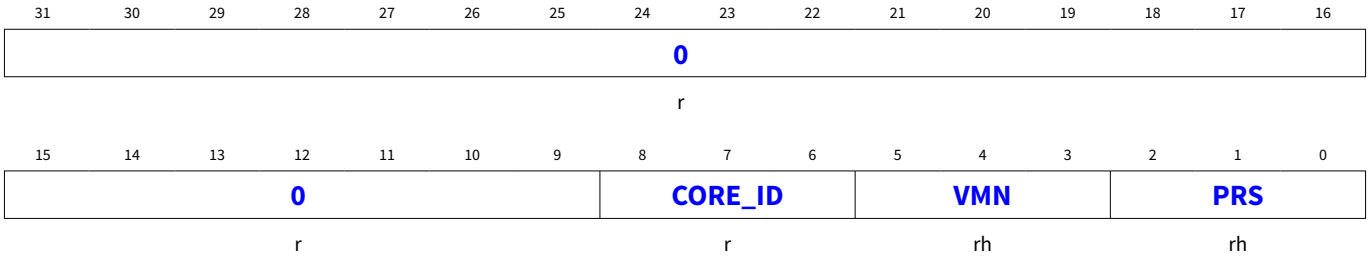
Table 255 Mirrors of CPU5_HRA_CORE_ID

Short name	Long name	Offset address
CPU5_HRB_CORE_ID	CPU5 core identification register is an alias.	$2FE1C_H$
CPU5_HRHV_CORE_ID	CPU5 core identification register is an alias.	$3FE1C_H$

4.10.4.1.17 CPU0 software identification

The software identification register holds key identifiers in one single register to ease software configuration.

CPU0_HRA_SWID	Offset address:	$1FE40_H$
CPU0 software identification	Kernel Reset value:	$0000\ 0000_H$
CPU1_HRA_SWID	Offset address:	$1FE40_H$
CPU1 software identification	Kernel Reset value:	$0000\ 0040_H$
CPU2_HRA_SWID	Offset address:	$1FE40_H$
CPU2 software identification	Kernel Reset value:	$0000\ 0080_H$
CPU3_HRA_SWID	Offset address:	$1FE40_H$
CPU3 software identification	Kernel Reset value:	$0000\ 00C0_H$
CPU4_HRA_SWID	Offset address:	$1FE40_H$
CPU4 software identification	Kernel Reset value:	$0000\ 0100_H$
CPU5_HRA_SWID	Offset address:	$1FE40_H$
CPU5 software identification	Kernel Reset value:	$0000\ 0140_H$
CPUCS_HRA_SWID	Offset address:	$1FE40_H$
CPUcs software identification	Kernel Reset value:	$0000\ 0180_H$
CPU_SWID	Offset address:	$0FE40_H$
Short address for domain CSFR	Reset values see:	Table 256



Field	Bits	Type	Description
PRS	2:0	rh	**Protection register set** Identifies the active level 1 Data and Code Memory Protection Register Set.
VMN	5:3	rh	**Virutal machine number** The current virutal machine number. When virtualization is disabled or not present this field will read as 1.
CORE_ID	8:6	r	**Core identification number** The identification number of the core.
0	31:9	r	**Reserved - RES** Read as 0.

Table 256 Reset values of CPU_SWID

Reset type	Reset value	Note
Kernel Reset	$0000\ 0000_H$	Instance specific: CPU0

(table continues...)

Table 256 (continued) Reset values of CPU_SWID

Reset type	Reset value	Note
Kernel Reset	$0000\ 0040_H$	Instance specific: CPU1
Kernel Reset	$0000\ 0080_H$	Instance specific: CPU2
Kernel Reset	$0000\ 00C0_H$	Instance specific: CPU3
Kernel Reset	$0000\ 0100_H$	Instance specific: CPU4
Kernel Reset	$0000\ 0140_H$	Instance specific: CPU5
Kernel Reset	$0000\ 0180_H$	Instance specific: CPUcs

Table 257 Mirrors of CPU0_HRA_SWID

Short name	Long name	Offset address
CPU0_HRB_SWID	CPU0 software identification is an alias.	$2FE40_H$
CPU0_HRHV_SWID	CPU0 software identification is an alias.	$3FE40_H$

Table 258 Mirrors of CPU1_HRA_SWID

Short name	Long name	Offset address
CPU1_HRB_SWID	CPU1 software identification is an alias.	$2FE40_H$
CPU1_HRHV_SWID	CPU1 software identification is an alias.	$3FE40_H$

Table 259 Mirrors of CPU2_HRA_SWID

Short name	Long name	Offset address
CPU2_HRB_SWID	CPU2 software identification is an alias.	$2FE40_H$
CPU2_HRHV_SWID	CPU2 software identification is an alias.	$3FE40_H$

Table 260 Mirrors of CPU3_HRA_SWID

Short name	Long name	Offset address
CPU3_HRB_SWID	CPU3 software identification is an alias.	$2FE40_H$
CPU3_HRHV_SWID	CPU3 software identification is an alias.	$3FE40_H$

Table 261 Mirrors of CPU4_HRA_SWID

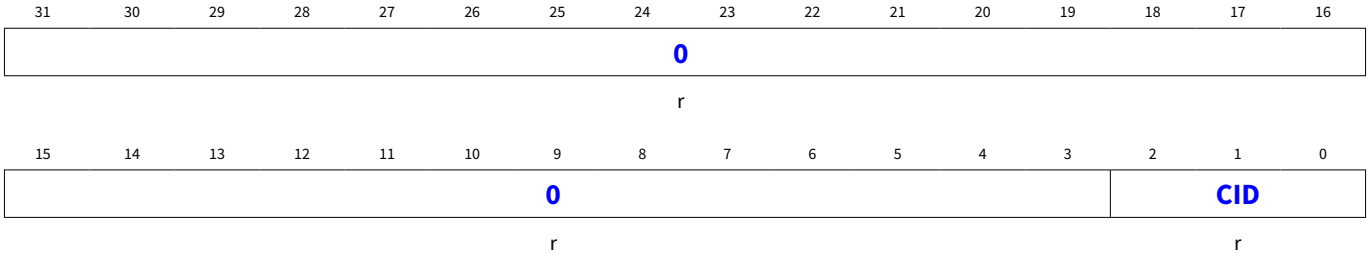
Short name	Long name	Offset address
CPU4_HRB_SWID	CPU4 software identification is an alias.	$2FE40_H$
CPU4_HRHV_SWID	CPU4 software identification is an alias.	$3FE40_H$

Table 262 Mirrors of CPU5_HRA_SWID

Short name	Long name	Offset address
CPU5_HRB_SWID	CPU5 software identification is an alias.	$2FE40_H$
CPU5_HRHV_SWID	CPU5 software identification is an alias.	$3FE40_H$

4.10.4.1.18 CPU0 customer ID register

CPU0_HRA_CUS_ID	Offset address:	$1FE50_H$
CPU0 customer ID register	Kernel Reset value:	$0000\ 0000_H$
CPU1_HRA_CUS_ID	Offset address:	$1FE50_H$
CPU1 customer ID register	Kernel Reset value:	$0000\ 0001_H$
CPU2_HRA_CUS_ID	Offset address:	$1FE50_H$
CPU2 customer ID register	Kernel Reset value:	$0000\ 0002_H$
CPU3_HRA_CUS_ID	Offset address:	$1FE50_H$
CPU3 customer ID register	Kernel Reset value:	$0000\ 0003_H$
CPU4_HRA_CUS_ID	Offset address:	$1FE50_H$
CPU4 customer ID register	Kernel Reset value:	$0000\ 0004_H$
CPU5_HRA_CUS_ID	Offset address:	$1FE50_H$
CPU5 customer ID register	Kernel Reset value:	$0000\ 0005_H$
CPUCS_HRA_CUS_ID	Offset address:	$1FE50_H$
CPUcs customer ID register	Kernel Reset value:	$0000\ 0006_H$
CPU_CUS_ID	Offset address:	$0FE50_H$
Short address for domain CSFR	Reset values see:	Table 263



Field	Bits	Type	Description
CID	2:0	r	**Customer ID** Alternate core numbering scheme
0	31:3	r	**Reserved - RES** Read as 0.

Table 263 Reset values of CPU_CUS_ID

Reset type	Reset value	Note
Kernel Reset	$0000\ 0000_H$	Instance specific: CPU0
Kernel Reset	$0000\ 0001_H$	Instance specific: CPU1
Kernel Reset	$0000\ 0002_H$	Instance specific: CPU2
Kernel Reset	$0000\ 0003_H$	Instance specific: CPU3
Kernel Reset	$0000\ 0004_H$	Instance specific: CPU4
Kernel Reset	$0000\ 0005_H$	Instance specific: CPU5
Kernel Reset	$0000\ 0006_H$	Instance specific: CPUcs

Table 264 Mirrors of CPU0_HRA_CUS_ID

Short name	Long name	Offset address
CPU0_HRB_CUS_ID	CPU0 customer ID register is an alias.	$2FE50_H$
CPU0_HRHV_CUS_ID	CPU0 customer ID register is an alias.	$3FE50_H$

Table 265 Mirrors of CPU1_HRA_CUS_ID

Short name	Long name	Offset address
CPU1_HRB_CUS_ID	CPU1 customer ID register is an alias.	$2FE50_H$
CPU1_HRHV_CUS_ID	CPU1 customer ID register is an alias.	$3FE50_H$

Table 266 Mirrors of CPU2_HRA_CUS_ID

Short name	Long name	Offset address
CPU2_HRB_CUS_ID	CPU2 customer ID register is an alias.	$2FE50_H$
CPU2_HRHV_CUS_ID	CPU2 customer ID register is an alias.	$3FE50_H$

Table 267 Mirrors of CPU3_HRA_CUS_ID

Short name	Long name	Offset address
CPU3_HRB_CUS_ID	CPU3 customer ID register is an alias.	$2FE50_H$
CPU3_HRHV_CUS_ID	CPU3 customer ID register is an alias.	$3FE50_H$

Table 268 Mirrors of CPU4_HRA_CUS_ID

Short name	Long name	Offset address
CPU4_HRB_CUS_ID	CPU4 customer ID register is an alias.	$2FE50_H$
CPU4_HRHV_CUS_ID	CPU4 customer ID register is an alias.	$3FE50_H$

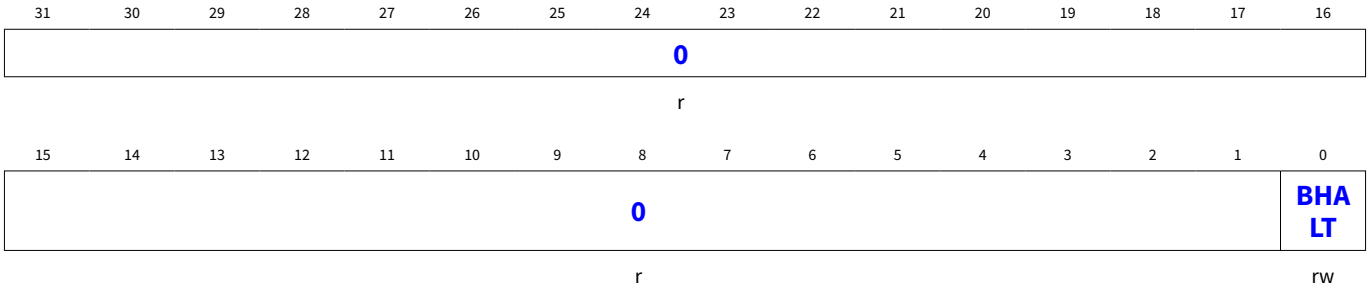
Table 269 Mirrors of CPU5_HRA_CUS_ID

Short name	Long name	Offset address
CPU5_HRB_CUS_ID	CPU5 customer ID register is an alias.	$2FE50_H$
CPU5_HRHV_CUS_ID	CPU5 customer ID register is an alias.	$3FE50_H$

4.10.4.1.19 CPU0 core boot configuration register

The boot configuration register controls the execution behavior following a reset.

CPU0_HRA_BOOTCON	Offset address:	$1FE60_H$
CPU0 core boot configuration register	Kernel Reset value:	$0000\ 0000_H$
CPUCS_HRA_BOOTCON	Offset address:	$1FE60_H$
CPUcs core boot configuration register	Kernel Reset value:	$0000\ 0000_H$
CPU1_HRA_BOOTCON	Offset address:	$1FE60_H$
CPU1 core boot configuration register	Kernel Reset value:	$0000\ 0001_H$
CPU2_HRA_BOOTCON	Offset address:	$1FE60_H$
CPU2 core boot configuration register	Kernel Reset value:	$0000\ 0001_H$
CPU3_HRA_BOOTCON	Offset address:	$1FE60_H$
CPU3 core boot configuration register	Kernel Reset value:	$0000\ 0001_H$
CPU4_HRA_BOOTCON	Offset address:	$1FE60_H$
CPU4 core boot configuration register	Kernel Reset value:	$0000\ 0001_H$
CPU5_HRA_BOOTCON	Offset address:	$1FE60_H$
CPU5 core boot configuration register	Kernel Reset value:	$0000\ 0001_H$
CPU_BOOTCON	Offset address:	$0FE60_H$
Short address for domain CSFR	Reset values see:	Table 270



Field	Bits	Type	Description
BHALT	0	rw	**Boot halt** 0_B Core is not in boot halt 1_B Core is in boot halt, write to 0 will exit
0	31:1	r	**Reserved** Read as 0; should be written with 0.

Table 270 Reset values of CPU_BOOTCON

Reset type	Reset value	Note
Kernel Reset	$0000\ 0000_H$	Instance specific: CPU0 or CPUcs
Kernel Reset	$0000\ 0001_H$	Instance specific: Any other CPU

Table 271 Mirrors of CPU0_HRA_BOOTCON

Short name	Long name	Offset address
CPU0_HRB_BOOTCON	CPU0 core boot configuration register is an alias.	$2FE60_H$
CPU0_HRHV_BOOTCON	CPU0 core boot configuration register is an alias.	$3FE60_H$

Table 272 Mirrors of CPU1_HRA_BOOTCON

Short name	Long name	Offset address
CPU1_HRB_BOOTCON	CPU1 core boot configuration register is an alias.	$2FE60_H$
CPU1_HRHV_BOOTCON	CPU1 core boot configuration register is an alias.	$3FE60_H$

Table 273 Mirrors of CPU2_HRA_BOOTCON

Short name	Long name	Offset address
CPU2_HRB_BOOTCON	CPU2 core boot configuration register is an alias.	$2FE60_H$
CPU2_HRHV_BOOTCON	CPU2 core boot configuration register is an alias.	$3FE60_H$

Table 274 Mirrors of CPU3_HRA_BOOTCON

Short name	Long name	Offset address
CPU3_HRB_BOOTCON	CPU3 core boot configuration register is an alias.	$2FE60_H$
CPU3_HRHV_BOOTCON	CPU3 core boot configuration register is an alias.	$3FE60_H$

Table 275 Mirrors of CPU4_HRA_BOOTCON

Short name	Long name	Offset address
CPU4_HRB_BOOTCON	CPU4 core boot configuration register is an alias.	$2FE60_H$
CPU4_HRHV_BOOTCON	CPU4 core boot configuration register is an alias.	$3FE60_H$

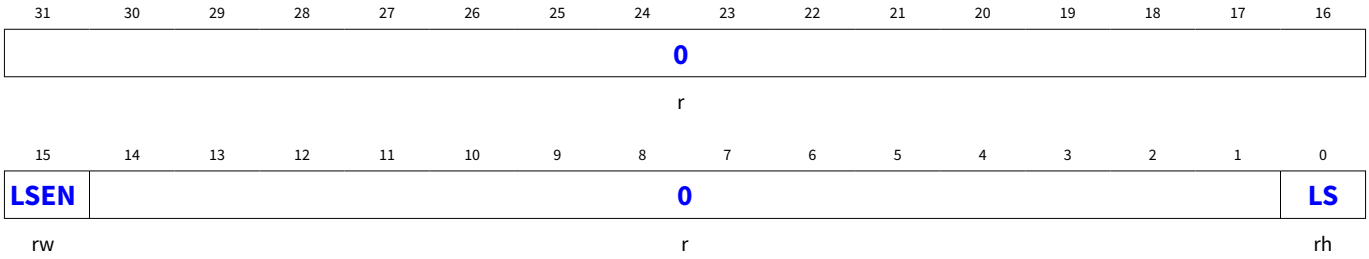
Table 276 Mirrors of CPU5_HRA_BOOTCON

Short name	Long name	Offset address
CPU5_HRB_BOOTCON	CPU5 core boot configuration register is an alias.	$2FE60_H$
CPU5_HRHV_BOOTCON	CPU5 core boot configuration register is an alias.	$3FE60_H$

4.10.4.1.20 CPU0 lockstep control register

Provides control for the CPU Lockstep Comparator Logic blocks.

CPU0_HRA_LCLCON	Offset address:	$1FE64_H$
CPU0 lockstep control register	Cold PowerOn Reset value:	$0000\ 8001_H$
CPU1_HRA_LCLCON	Offset address:	$1FE64_H$
CPU1 lockstep control register	Cold PowerOn Reset value:	$0000\ 8001_H$
CPU2_HRA_LCLCON	Offset address:	$1FE64_H$
CPU2 lockstep control register	Cold PowerOn Reset value:	$0000\ 8001_H$
CPU3_HRA_LCLCON	Offset address:	$1FE64_H$
CPU3 lockstep control register	Cold PowerOn Reset value:	$0000\ 8001_H$
CPU4_HRA_LCLCON	Offset address:	$1FE64_H$
CPU4 lockstep control register	Cold PowerOn Reset value:	$0000\ 8001_H$
CPU5_HRA_LCLCON	Offset address:	$1FE64_H$
CPU5 lockstep control register	Cold PowerOn Reset value:	$0000\ 8001_H$
CPUCS_HRA_LCLCON	Offset address:	$1FE64_H$
CPUcs lockstep control register	Cold PowerOn Reset value:	$0000\ 0000_H$
CPU_LCLCON	Offset address:	$0FE64_H$
Short address for domain CSFR	Reset values see:	Table 277



Field	Bits	Type	Description
LS	0	rh	**Lockstep mode status** This bit indicates whether the CPU is currently running in lockstep monitor mode 0_B Not in lockstep mode 1_B Running in lockstep mode
LSEN	15	rw	**Lockstep enable** This bit may only be written by SSW during boot. Enable lockstep CPU monitoring for the associated processor core. CPU0 is always enabled and cannot be disabled. After cold reset, lockstep is enabled by default. The LSEN bit may be cleared during the boot to disable lockstep mode. SMU lockstep fault reporting should be disabled when lockstep is disabled. 0_B Lockstep is disabled 1_B Lockstep enabled (Default after Cold Power-On Reset)

(table continues...)

(continued)

Field	Bits	Type	Description
0	14:1, 31:16	r	**Reserved** Read as 0; should be written with 0.

Table 277 Reset values of CPU_LCLCON

Reset type	Reset value	Note
Cold PowerOn Reset	$0000\ 8001_H$	Instance specific: Lockstep present
Cold PowerOn Reset	$0000\ 0000_H$	Instance specific: Lockstep not present

Table 278 Mirrors of CPU0_HRA_LCLCON

Short name	Long name	Offset address
CPU0_HRB_LCLCON	CPU0 lockstep control register is an alias.	$2FE64_H$
CPU0_HRHV_LCLCON	CPU0 lockstep control register is an alias.	$3FE64_H$

Table 279 Mirrors of CPU1_HRA_LCLCON

Short name	Long name	Offset address
CPU1_HRB_LCLCON	CPU1 lockstep control register is an alias.	$2FE64_H$
CPU1_HRHV_LCLCON	CPU1 lockstep control register is an alias.	$3FE64_H$

Table 280 Mirrors of CPU2_HRA_LCLCON

Short name	Long name	Offset address
CPU2_HRB_LCLCON	CPU2 lockstep control register is an alias.	$2FE64_H$
CPU2_HRHV_LCLCON	CPU2 lockstep control register is an alias.	$3FE64_H$

Table 281 Mirrors of CPU3_HRA_LCLCON

Short name	Long name	Offset address
CPU3_HRB_LCLCON	CPU3 lockstep control register is an alias.	$2FE64_H$
CPU3_HRHV_LCLCON	CPU3 lockstep control register is an alias.	$3FE64_H$

Table 282 Mirrors of CPU4_HRA_LCLCON

Short name	Long name	Offset address
CPU4_HRB_LCLCON	CPU4 lockstep control register is an alias.	$2FE64_H$
CPU4_HRHV_LCLCON	CPU4 lockstep control register is an alias.	$3FE64_H$

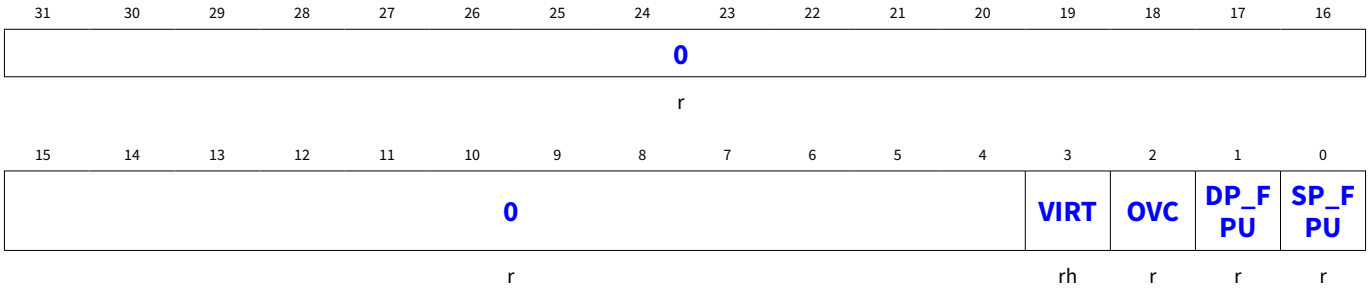
Table 283 Mirrors of CPU5_HRA_LCLCON

Short name	Long name	Offset address
CPU5_HRB_LCLCON	CPU5 lockstep control register is an alias.	$2FE64_H$
CPU5_HRHV_LCLCON	CPU5 lockstep control register is an alias.	$3FE64_H$

4.10.4.1.21 CPU0 hardware configuration

The TriCore core configuration register holds information regarding the hardware support for implementation specific features of the core.

CPU0_HRA_TCCON	Offset address:	$1FE6C_H$
CPU0 hardware configuration	Kernel Reset value:	$0000\ 000F_H$
CPU1_HRA_TCCON	Offset address:	$1FE6C_H$
CPU1 hardware configuration	Kernel Reset value:	$0000\ 000F_H$
CPU2_HRA_TCCON	Offset address:	$1FE6C_H$
CPU2 hardware configuration	Kernel Reset value:	$0000\ 000F_H$
CPU3_HRA_TCCON	Offset address:	$1FE6C_H$
CPU3 hardware configuration	Kernel Reset value:	$0000\ 000F_H$
CPU4_HRA_TCCON	Offset address:	$1FE6C_H$
CPU4 hardware configuration	Kernel Reset value:	$0000\ 000F_H$
CPU5_HRA_TCCON	Offset address:	$1FE6C_H$
CPU5 hardware configuration	Kernel Reset value:	$0000\ 000F_H$
CPUCS_HRA_TCCON	Offset address:	$1FE6C_H$
CPUcs hardware configuration	Kernel Reset value:	$0000\ 0000_H$
CPU_TCCON	Offset address:	$0FE6C_H$
Short address for domain CSFR	Reset values see:	Table 284



Field	Bits	Type	Description
SP_FPU	0	r	**Single precision FPU hardware support** 0_B Single precision FPU hardware not present 1_B Single precision FPU hardware present
DP_FPU	1	r	**Double precision FPU hardware support** 0_B Double precision FPU hardware not present 1_B Double precision FPU hardware present
OVC	2	r	**Overlay hardware support** 0_B No overlay 1_B Overlay present

(table continues...)

(continued)

Field	Bits	Type	Description
VIRT	3	rh	**Virtualization hardware support** 0_B Virtualization hardware extension not present or disabled 1_B Virtualization hardware extension present and enabled
0	31:4	r	**Reserved - RES** Read as 0.

Table 284 Reset values of CPU_TCCON

Reset type	Reset value	Note
Kernel Reset	$0000\ 0000_H$	Instance specific: No Special features
Kernel Reset	$0000\ 000F_H$	Instance specific: VIRT + OVC + DP FPU + SP FPU

Table 285 Mirrors of CPU0_HRA_TCCON

Short name	Long name	Offset address
CPU0_HRB_TCCON	CPU0 hardware configuration is an alias.	$2FE6C_H$
CPU0_HRHV_TCCON	CPU0 hardware configuration is an alias.	$3FE6C_H$

Table 286 Mirrors of CPU1_HRA_TCCON

Short name	Long name	Offset address
CPU1_HRB_TCCON	CPU1 hardware configuration is an alias.	$2FE6C_H$
CPU1_HRHV_TCCON	CPU1 hardware configuration is an alias.	$3FE6C_H$

Table 287 Mirrors of CPU2_HRA_TCCON

Short name	Long name	Offset address
CPU2_HRB_TCCON	CPU2 hardware configuration is an alias.	$2FE6C_H$
CPU2_HRHV_TCCON	CPU2 hardware configuration is an alias.	$3FE6C_H$

Table 288 Mirrors of CPU3_HRA_TCCON

Short name	Long name	Offset address
CPU3_HRB_TCCON	CPU3 hardware configuration is an alias.	$2FE6C_H$
CPU3_HRHV_TCCON	CPU3 hardware configuration is an alias.	$3FE6C_H$

Table 289 Mirrors of CPU4_HRA_TCCON

Short name	Long name	Offset address
CPU4_HRB_TCCON	CPU4 hardware configuration is an alias.	$2FE6C_H$
CPU4_HRHV_TCCON	CPU4 hardware configuration is an alias.	$3FE6C_H$

Table 290 Mirrors of CPU5_HRA_TCCON

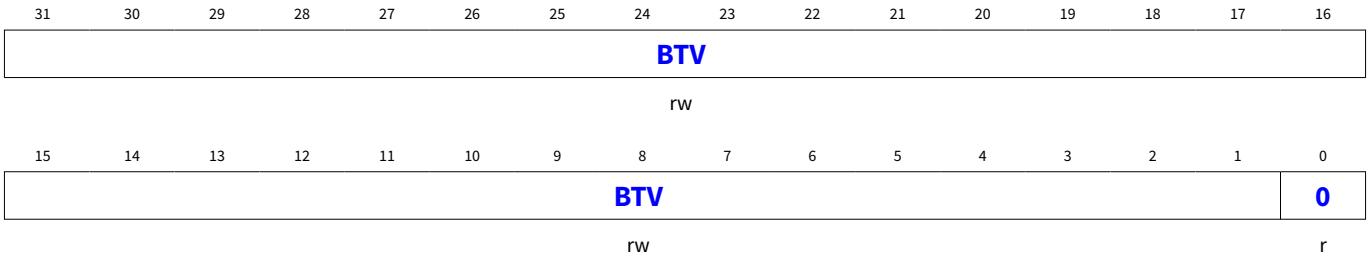
Short name	Long name	Offset address
CPU5_HRB_TCCON	CPU5 hardware configuration is an alias.	$2FE6C_H$
CPU5_HRHV_TCCON	CPU5 hardware configuration is an alias.	$3FE6C_H$

4.10.4.1.22 CPU0 HR specific base trap vector table pointer

The BTV contains the base address of the trap vector table. When a trap occurs, the entry address into the trap vector table is generated from the Trap Class of that trap, left-shifted by 5 bits and then ORd with the contents of the BTV register. The left-shift of the Trap Class results in a spacing of 8 words (32 bytes) between the individual entries in the vector table.

Note: This register is protected by CPUx_PROTSFRE.

CPU0_HRA_BTV	Offset address:	$1FE24_H$
CPU0 HR specific base trap vector table pointer	Kernel Reset value:	$AFFF\ FE00_H$
CPU1_HRA_BTV	Offset address:	$1FE24_H$
CPU1 HR specific base trap vector table pointer	Kernel Reset value:	$0000\ FE00_H$
CPU2_HRA_BTV	Offset address:	$1FE24_H$
CPU2 HR specific base trap vector table pointer	Kernel Reset value:	$0000\ FE00_H$
CPU3_HRA_BTV	Offset address:	$1FE24_H$
CPU3 HR specific base trap vector table pointer	Kernel Reset value:	$0000\ FE00_H$
CPU4_HRA_BTV	Offset address:	$1FE24_H$
CPU4 HR specific base trap vector table pointer	Kernel Reset value:	$0000\ FE00_H$
CPU5_HRA_BTV	Offset address:	$1FE24_H$
CPU5 HR specific base trap vector table pointer	Kernel Reset value:	$0000\ FE00_H$
CPUCS_HRA_BTV	Offset address:	$1FE24_H$
CPUcs HR specific base trap vector table pointer	Kernel Reset value:	$AFFE\ FE00_H$
CPU_BTV	Offset address:	$0FE24_H$
Short address for domain CSFR	Reset values see:	Table 291
CPU_HVHRA_BTV	Offset address:	$0FE25_H$
Short address for domain CSFR	Reset values see:	Table 292
CPU_HVHRB_BTV	Offset address:	$0FE26_H$
Short address for domain CSFR	Reset values see:	Table 293



Field	Bits	Type	Description
BTV	31:1	rw	**Base address of trap vector table** The address in the BTV register must be aligned to an even byte address (halfword address). Also, due to the simple ORing of the left-shifted trap identification number and the contents of the BTV register, the alignment of the base address of the vector table must be to a power of two boundary. There are eight different trap classes, resulting in Trap Classes from 0 to 7. The contents of BTV should therefore be set to at least a 256 byte boundary (8 Trap Classes * 8 word spacing).
0	0	r	**Reserved - RES** Read as 0; should be written as 0.

Table 291 Reset values of CPU_BTV

Reset type	Reset value	Note
Kernel Reset	AFFF FE00$_H$	Instance specific
Kernel Reset	0000 FE00$_H$	Instance specific
Kernel Reset	AFFE FE00$_H$	Instance specific

Table 292 Reset values of CPU_HVHRA_BTV

Reset type	Reset value	Note
Kernel Reset	AFFF FE00$_H$	Instance specific
Kernel Reset	0000 FE00$_H$	Instance specific

Table 293 Reset values of CPU_HVHRB_BTV

Reset type	Reset value	Note
Kernel Reset	AFFF FE00$_H$	Instance specific
Kernel Reset	0000 FE00$_H$	Instance specific

4.10.4.1.23 CPU0 HR specific base trap vector table pointer

CPU0_HRB_BTV Offset address: 2FE24$_H$

CPU0 HR specific base trap vector table pointer Kernel Reset value: AFFF FE00$_H$

CPU1_HRB_BTV Offset address: 2FE24$_H$

CPU1 HR specific base trap vector table pointer Kernel Reset value: 0000 FE00$_H$

CPU2_HRB_BTV Offset address: 2FE24$_H$

CPU2 HR specific base trap vector table pointer Kernel Reset value: 0000 FE00$_H$

CPU3_HRB_BTV Offset address: 2FE24$_H$

CPU3 HR specific base trap vector table pointer Kernel Reset value: 0000 FE00$_H$

CPU4_HRB_BTV Offset address: 2FE24$_H$

CPU4 HR specific base trap vector table pointer Kernel Reset value: 0000 FE00$_H$

CPU5_HRB_BTV Offset address: 2FE24$_H$

CPU5 HR specific base trap vector table pointer Kernel Reset value: 0000 FE00$_H$

CPU0_HRHV_BTV Offset address: $3FE24_H$

CPU0 HR specific base trap vector table pointer Kernel Reset value: $AFFF\ FE00_H$

CPU1_HRHV_BTV Offset address: $3FE24_H$

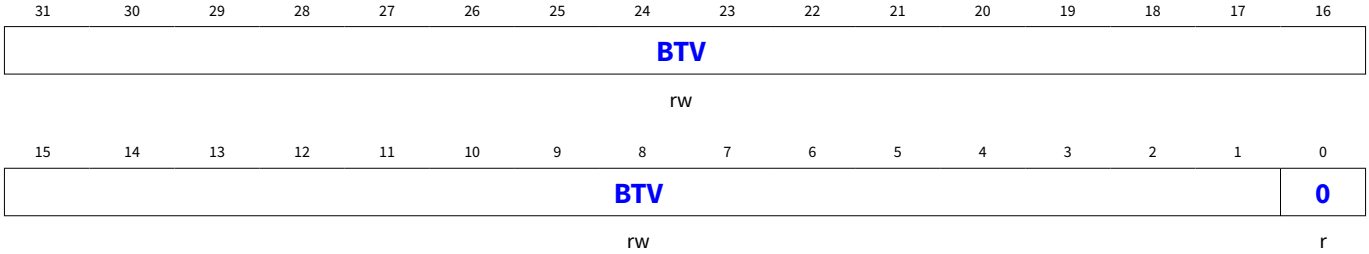
CPU1 HR specific base trap vector table pointer Kernel Reset value: $0000\ FE00_H$

CPU2_HRHV_BTV Offset address: $3FE24_H$

CPU2 HR specific base trap vector table pointer Kernel Reset value: $0000\ FE00_H$

CPU3_HRHV_BTV Offset address: $3FE24_H$

CPU3 HR specific base trap vector table pointer Kernel Reset value: $0000\ FE00_H$



Field	Bits	Type	Description
BTV	31:1	rw	**Base address of trap vector table** The address in the BTV register must be aligned to an even byte address (halfword address). Also, due to the simple ORing of the left-shifted trap identification number and the contents of the BTV register, the alignment of the base address of the vector table must be to a power of two boundary. There are eight different trap classes, resulting in Trap Classes from 0 to 7. The contents of BTV should therefore be set to at least a 256 byte boundary (8 Trap Classes * 8 word spacing).
0	0	r	**Reserved - RES** Read as 0; should be written as 0.

Table 294 Reset values of CPU_BTV

Reset type	Reset value	Note
Kernel Reset	$AFFF\ FE00_H$	Instance specific
Kernel Reset	$0000\ FE00_H$	Instance specific

Table 295 Reset values of CPU_HVHRA_BTV

Reset type	Reset value	Note
Kernel Reset	$AFFF\ FE00_H$	Instance specific
Kernel Reset	$0000\ FE00_H$	Instance specific

Table 296 Reset values of CPU_HVHRB_BTV

Reset type	Reset value	Note
Kernel Reset	$AFFF\ FE00_H$	Instance specific
Kernel Reset	$0000\ FE00_H$	Instance specific

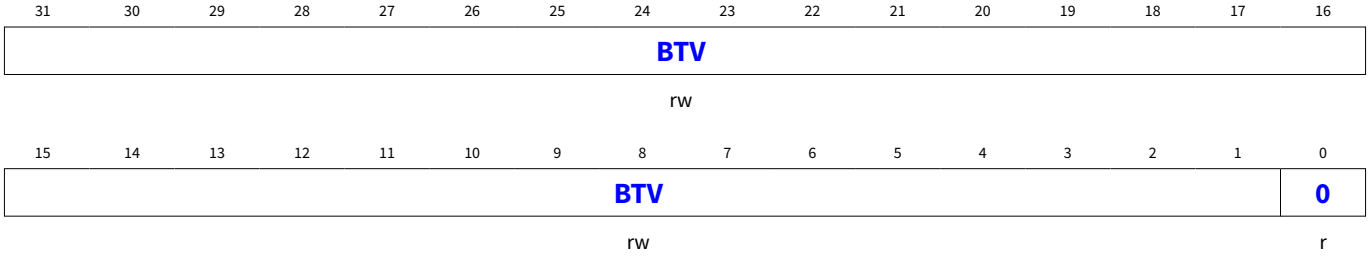
4.10.4.1.24 CPU4 HR specific base trap vector table pointer

CPU4_HRHV_BTV Offset address: $3FE24_H$

CPU4 HR specific base trap vector table pointer Kernel Reset value: $0000\ FE00_H$

CPU5_HRHV_BTV Offset address: $3FE24_H$

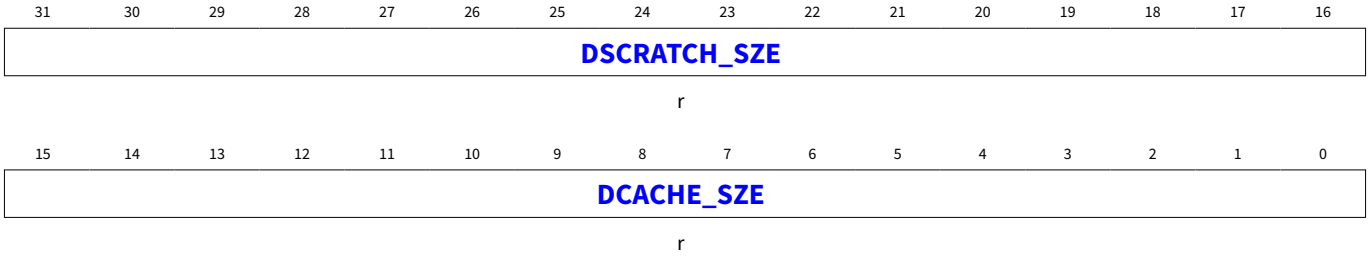
CPU5 HR specific base trap vector table pointer Kernel Reset value: $0000\ FE00_H$



Field	Bits	Type	Description
BTV	31:1	rw	**Base address of trap vector table** The address in the BTV register must be aligned to an even byte address (halfword address). Also, due to the simple ORing of the left-shifted trap identification number and the contents of the BTV register, the alignment of the base address of the vector table must be to a power of two boundary. There are eight different trap classes, resulting in Trap Classes from 0 to 7. The contents of BTV should therefore be set to at least a 256 byte boundary (8 Trap Classes * 8 word spacing).
0	0	r	**Reserved - RES** Read as 0; should be written as 0.

4.10.4.1.25 CPU0 data control register 2

CPU0_HRA_DCON2	Offset address:	19000_H
CPU0 data control register 2	Kernel Reset value:	$00F0\ 0010_H$
CPU1_HRA_DCON2	Offset address:	19000_H
CPU1 data control register 2	Kernel Reset value:	$00F0\ 0010_H$
CPU2_HRA_DCON2	Offset address:	19000_H
CPU2 data control register 2	Kernel Reset value:	$00F0\ 0010_H$
CPU3_HRA_DCON2	Offset address:	19000_H
CPU3 data control register 2	Kernel Reset value:	$00F0\ 0010_H$
CPU4_HRA_DCON2	Offset address:	19000_H
CPU4 data control register 2	Kernel Reset value:	$00F0\ 0010_H$
CPU5_HRA_DCON2	Offset address:	19000_H
CPU5 data control register 2	Kernel Reset value:	$00F0\ 0010_H$
CPUCS_HRA_DCON2	Offset address:	19000_H
CPUcs data control register 2	Kernel Reset value:	$00F0\ 0010_H$
CPU_DCON2	Offset address:	09000_H
Short address for domain CSFR	Kernel Reset value:	$00F0\ 0010_H$



Field	Bits	Type	Description
DCACHE_SZE	15:0	r	**Data cache size** In KBytes
DSCRATCH_SZE	31:16	r	**Data scratch size** In KBytes

Table 297 Mirrors of CPU0_HRA_DCON2

Short name	Long name	Offset address
CPU0_HRB_DCON2	CPU0 data control register 2 is an alias.	29000_H
CPU0_HRHV_DCON2	CPU0 data control register 2 is an alias.	39000_H

Table 298 Mirrors of CPU1_HRA_DCON2

Short name	Long name	Offset address
CPU1_HRB_DCON2	CPU1 data control register 2 is an alias.	29000_H
CPU1_HRHV_DCON2	CPU1 data control register 2 is an alias.	39000_H

Table 299 Mirrors of CPU2_HRA_DCON2

Short name	Long name	Offset address
CPU2_HRB_DCON2	CPU2 data control register 2 is an alias.	29000_H
CPU2_HRHV_DCON2	CPU2 data control register 2 is an alias.	39000_H

Table 300 Mirrors of CPU3_HRA_DCON2

Short name	Long name	Offset address
CPU3_HRB_DCON2	CPU3 data control register 2 is an alias.	29000_H
CPU3_HRHV_DCON2	CPU3 data control register 2 is an alias.	39000_H

Table 301 Mirrors of CPU4_HRA_DCON2

Short name	Long name	Offset address
CPU4_HRB_DCON2	CPU4 data control register 2 is an alias.	29000_H
CPU4_HRHV_DCON2	CPU4 data control register 2 is an alias.	39000_H

Table 302 Mirrors of CPU5_HRA_DCON2

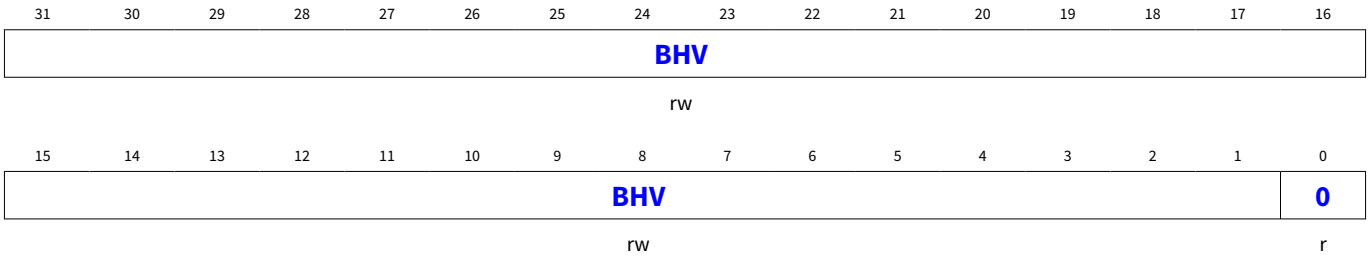
Short name	Long name	Offset address
CPU5_HRB_DCON2	CPU5 data control register 2 is an alias.	29000_H
CPU5_HRHV_DCON2	CPU5 data control register 2 is an alias.	39000_H

4.10.4.1.26 CPU0 base hypervisor vector table pointer

The BHV contains the base address of the hypervisor trap vector table. When a hypervisor trap occurs, the entry address into the hypervisor trap vector table is generated from the hypervisor trap class number (HVTCN) of that trap, left-shifted by 5-bits and then ORd with the contents of the BHV register. The left-shift of the HVTCN results in a spacing of 8-words (32-bytes) between the individual entries in the vector table.

Note: This register is protected by CPUx_PROTSFRE.

CPU0_HRHV_BHV	Offset address:	$3B010_H$
CPU0 base hypervisor vector table pointer	Kernel Reset value:	AFFF FD00$_H$
CPU1_HRHV_BHV	Offset address:	$3B010_H$
CPU1 base hypervisor vector table pointer	Kernel Reset value:	0000 FD00$_H$
CPU2_HRHV_BHV	Offset address:	$3B010_H$
CPU2 base hypervisor vector table pointer	Kernel Reset value:	0000 FD00$_H$
CPU3_HRHV_BHV	Offset address:	$3B010_H$
CPU3 base hypervisor vector table pointer	Kernel Reset value:	0000 FD00$_H$
CPU4_HRHV_BHV	Offset address:	$3B010_H$
CPU4 base hypervisor vector table pointer	Kernel Reset value:	0000 FD00$_H$
CPU5_HRHV_BHV	Offset address:	$3B010_H$
CPU5 base hypervisor vector table pointer	Kernel Reset value:	0000 FD00$_H$
CPU_BHV	Offset address:	$0B010_H$
Short address for domain CSFR	Reset values see:	Table 303



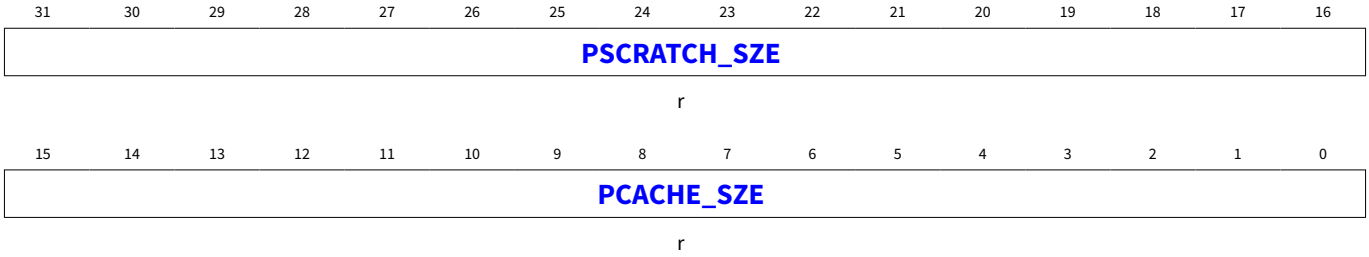
Field	Bits	Type	Description
BHV	31:1	rw	**Base address of hypervisor trap vector table** The address in the BHV register must be aligned to an even byte address (half-word address). Also, due to the simple ORing of the left-shifted trap identification number and the contents of the BHV register, the alignment of the base address of the vector table must be to a power of two boundary. There are six different hypervisor trap classes, resulting in Trap Classes from 0 to 5. The contents of BHV should therefore be set to at least a 256 byte boundary (8 Trap Classes * 8-word spacing, with 8 being the nearest power of two for 6).
0	0	r	**Reserved - RES** Read as 0; should be written as 0.

Table 303 Reset values of CPU_BHV

Reset type	Reset value	Note
Kernel Reset	AFFF FD00$_H$	Instance specific
Kernel Reset	0000 FD00$_H$	Instance specific

4.10.4.1.27 CPU0 program control 2

CPU0_HRA_PCON2	Offset address:	19208_H
CPU0 program control 2	Kernel Reset value:	$0040\ 0020_H$
CPU1_HRA_PCON2	Offset address:	19208_H
CPU1 program control 2	Kernel Reset value:	$0040\ 0020_H$
CPU2_HRA_PCON2	Offset address:	19208_H
CPU2 program control 2	Kernel Reset value:	$0040\ 0020_H$
CPU3_HRA_PCON2	Offset address:	19208_H
CPU3 program control 2	Kernel Reset value:	$0040\ 0020_H$
CPU4_HRA_PCON2	Offset address:	19208_H
CPU4 program control 2	Kernel Reset value:	$0040\ 0020_H$
CPU5_HRA_PCON2	Offset address:	19208_H
CPU5 program control 2	Kernel Reset value:	$0040\ 0020_H$
CPUCS_HRA_PCON2	Offset address:	19208_H
CPUcs program control 2	Kernel Reset value:	$0040\ 0020_H$
CPU_PCON2	Offset address:	09208_H
Short address for domain CSFR	Kernel Reset value:	$0040\ 0020_H$



Field	Bits	Type	Description
PCACHE_SZE	15:0	r	**Program cache size** In KBytes
PSCRATCH_SZE	31:16	r	**Program scratch size** In KBytes

Table 304 Mirrors of CPU0_HRA_PCON2

Short name	Long name	Offset address
CPU0_HRB_PCON2	CPU0 program control 2 is an alias.	29208_H
CPU0_HRHV_PCON2	CPU0 program control 2 is an alias.	39208_H

Table 305 Mirrors of CPU1_HRA_PCON2

Short name	Long name	Offset address
CPU1_HRB_PCON2	CPU1 program control 2 is an alias.	29208_H
CPU1_HRHV_PCON2	CPU1 program control 2 is an alias.	39208_H

Table 306 Mirrors of CPU2_HRA_PCON2

Short name	Long name	Offset address
CPU2_HRB_PCON2	CPU2 program control 2 is an alias.	29208_H
CPU2_HRHV_PCON2	CPU2 program control 2 is an alias.	39208_H

Table 307 Mirrors of CPU3_HRA_PCON2

Short name	Long name	Offset address
CPU3_HRB_PCON2	CPU3 program control 2 is an alias.	29208_H
CPU3_HRHV_PCON2	CPU3 program control 2 is an alias.	39208_H

Table 308 Mirrors of CPU4_HRA_PCON2

Short name	Long name	Offset address
CPU4_HRB_PCON2	CPU4 program control 2 is an alias.	29208_H
CPU4_HRHV_PCON2	CPU4 program control 2 is an alias.	39208_H

Table 309 Mirrors of CPU5_HRA_PCON2

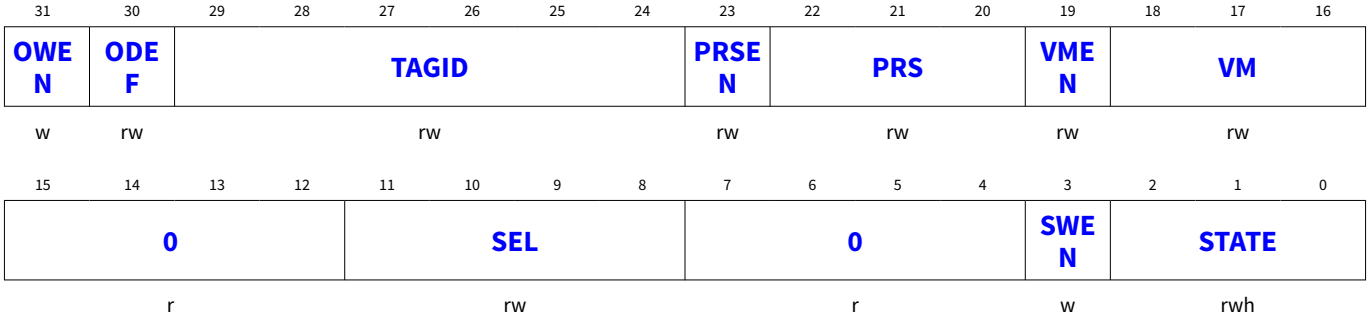
Short name	Long name	Offset address
CPU5_HRB_PCON2	CPU5 program control 2 is an alias.	29208_H
CPU5_HRHV_PCON2	CPU5 program control 2 is an alias.	39208_H

4.10.4.1.28 CPU0 PROTSPR register safe endinit

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner and in some PROT states, the secure master to update the PROT state as described in the PROT mechanism.

The PROTSPRSE register controls lock / unlock of the local Safe Endinit (SE) protected control registers allocated to the SPR resources.

CPU0_PROTSPRSE	Offset address:	$0E008_H$
CPU0 PROTSPR register safe endinit	Application Reset value:	$0000\ 0000_H$
CPU1_PROTSPRSE	Offset address:	$0E008_H$
CPU1 PROTSPR register safe endinit	Application Reset value:	$0000\ 0000_H$
CPU2_PROTSPRSE	Offset address:	$0E008_H$
CPU2 PROTSPR register safe endinit	Application Reset value:	$0000\ 0000_H$
CPU3_PROTSPRSE	Offset address:	$0E008_H$
CPU3 PROTSPR register safe endinit	Application Reset value:	$0000\ 0000_H$
CPU4_PROTSPRSE	Offset address:	$0E008_H$
CPU4 PROTSPR register safe endinit	Application Reset value:	$0000\ 0000_H$
CPU5_PROTSPRSE	Offset address:	$0E008_H$
CPU5 PROTSPR register safe endinit	Application Reset value:	$0000\ 0000_H$
CPUCS_PROTSPRSE	Offset address:	$0E008_H$
CPUcs PROTSPR register safe endinit	Application Reset value:	$4C00\ 0000_H$



Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
SEL	11:8	rw	**Range selection** Returns the number of the range being configured when read. Can be written by the PROT owner or CS master to select the range number to be configured if the state transition is valid in the PROT state diagram. 0_H **Const0**: Range 0 selected for configuration … F_H **Const15**: Range 15 selected for configuration
VM	18:16	rw	**Virtual machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual machine definition enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.

(table continues...)

(continued)

Field	Bits	Type	Description
PRSEN	23	rw	**Protection set definition enable for PROT owner** 0_B **Disable**: PRS is part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	7:4, 15:12	r	**Reserved** Read as 0; should be written with 0.

Table 310 Access mode restrictions sorted by descending priority

Applies to CPU0_PROTSPRSE
Applies to CPU1_PROTSPRSE
Applies to CPU2_PROTSPRSE
Applies to CPU3_PROTSPRSE
Applies to CPU4_PROTSPRSE
Applies to CPU5_PROTSPRSE
Applies to CPUCS_PROTSPRSE

Mode name	Access mode		Description
write 1 to .**SWEN**	rwh	STATE	
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

4.10.4.1.29 CPU0 shadow SPR, write access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

CPU0_ACCENSPRCFG_WRA Offset address: $0E020_H$

CPU0 shadow SPR, write access enable register A Application Reset value: $1000\ 0003_H$

CPU1_ACCENSPRCFG_WRA Offset address: $0E020_H$

CPU1 shadow SPR, write access enable register A Application Reset value: $1000\ 0003_H$

CPU2_ACCENSPRCFG_WRA Offset address: $0E020_H$

CPU2 shadow SPR, write access enable register A Application Reset value: $1000\ 0003_H$

CPU3_ACCENSPRCFG_WRA Offset address: $0E020_H$

CPU3 shadow SPR, write access enable register A Application Reset value: $1000\ 0003_H$

CPU4_ACCENSPRCFG_WRA Offset address: $0E020_H$

CPU4 shadow SPR, write access enable register A Application Reset value: $1000\ 0003_H$

CPU5_ACCENSPRCFG_WRA Offset address: $0E020_H$

CPU5 shadow SPR, write access enable register A Application Reset value: $1000\ 0003_H$

CPUCS_ACCENSPRCFG_WRA Offset address: $0E020_H$

CPUcs shadow SPR, write access enable register A Reset values see: Table 311

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	**EN30**	**EN29**	**EN28**	**EN27**	**EN26**	**EN25**	**EN24**	**EN23**	**EN22**	**EN21**	**EN20**	**EN19**	**EN18**	**EN17**	**EN16**
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	**EN14**	**EN13**	**EN12**	**EN11**	**EN10**	**EN09**	**EN08**	**EN07**	**EN06**	**EN05**	**EN04**	**EN03**	**EN02**	**EN01**	**EN00**
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access

Table 311 Reset values of CPUCS_ACCENSPRCFG_WRA

Reset type	Reset value	Note
Application Reset	$0000\ 1000_H$	CPUcs, CSRM debug disabled
After Boot-FW Value	$1000\ 3000_H$	CPUcs, CSRM debug enabled

4.10.4.1.30 CPU0 shadow SPR, write access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

CPU0_ACCENSPRCFG_WRB Offset address: $0E024_H$

CPU0 shadow SPR, write access enable register B Application Reset value: $0000\ 0000_H$

CPU1_ACCENSPRCFG_WRB Offset address: $0E024_H$

CPU1 shadow SPR, write access enable register B Application Reset value: $0000\ 0000_H$

CPU2_ACCENSPRCFG_WRB Offset address: $0E024_H$

CPU2 shadow SPR, write access enable register B Application Reset value: $0000\ 0000_H$

CPU3_ACCENSPRCFG_WRB Offset address: $0E024_H$

CPU3 shadow SPR, write access enable register B Application Reset value: $0000\ 0000_H$

CPU4_ACCENSPRCFG_WRB Offset address: $0E024_H$

CPU4 shadow SPR, write access enable register B Application Reset value: $0000\ 0000_H$

CPU5_ACCENSPRCFG_WRB Offset address: $0E024_H$

CPU5 shadow SPR, write access enable register B Application Reset value: $0000\ 0000_H$

CPUCS_ACCENSPRCFG_WRB Offset address: $0E024_H$

CPUcs shadow SPR, write access enable register B Reset values see: Table 312

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN63	**EN62**	**EN61**	**EN60**	**EN59**	**EN58**	**EN57**	**EN56**	**EN55**	**EN54**	**EN53**	**EN52**	**EN51**	**EN50**	**EN49**	**EN48**
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN47	**EN46**	**EN45**	**EN44**	**EN43**	**EN42**	**EN41**	**EN40**	**EN39**	**EN38**	**EN37**	**EN36**	**EN35**	**EN34**	**EN33**	**EN32**
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-63)	q-32	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access

Table 312 Reset values of CPUCS_ACCENSPRCFG_WRB

Reset type	Reset value	Note
Application Reset	$0000\ 0000_H$	CPUcs, CSRM debug disabled
After Boot-FW Value	$0000\ 4000_H$	CPUcs, CSRM debug enabled

4.10.4.1.31 CPU0 shadow SPR, read access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

CPU0_ACCENSPRCFG_RDA Offset address: $0E028_H$

CPU0 shadow SPR, read access enable register A Application Reset value: $FFFF\ FFFF_H$

CPU1_ACCENSPRCFG_RDA Offset address: $0E028_H$

CPU1 shadow SPR, read access enable register A Application Reset value: $FFFF\ FFFF_H$

CPU2_ACCENSPRCFG_RDA Offset address: $0E028_H$

CPU2 shadow SPR, read access enable register A Application Reset value: $FFFF\ FFFF_H$

CPU3_ACCENSPRCFG_RDA Offset address: $0E028_H$

CPU3 shadow SPR, read access enable register A Application Reset value: $FFFF\ FFFF_H$

CPU4_ACCENSPRCFG_RDA Offset address: $0E028_H$

CPU4 shadow SPR, read access enable register A Application Reset value: $FFFF\ FFFF_H$

CPU5_ACCENSPRCFG_RDA Offset address: $0E028_H$

CPU5 shadow SPR, read access enable register A Application Reset value: $FFFF\ FFFF_H$

CPUCS_ACCENSPRCFG_RDA Offset address: $0E028_H$

CPUcs shadow SPR, read access enable register A Reset values see: Table 313

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	**EN30**	**EN29**	**EN28**	**EN27**	**EN26**	**EN25**	**EN24**	**EN23**	**EN22**	**EN21**	**EN20**	**EN19**	**EN18**	**EN17**	**EN16**
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	**EN14**	**EN13**	**EN12**	**EN11**	**EN10**	**EN09**	**EN08**	**EN07**	**EN06**	**EN05**	**EN04**	**EN03**	**EN02**	**EN01**	**EN00**
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access

Table 313 Reset values of CPUCS_ACCENSPRCFG_RDA

Reset type	Reset value	Note
Application Reset	$0000\ 1000_H$	CPUcs, CSRM debug disabled
After Boot-FW Value	$1000\ 3000_H$	CPUcs, CSRM debug enabled

4.10.4.1.32 CPU0 shadow SPR, read access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

CPU0_ACCENSPRCFG_RDB Offset address: $0E02C_H$

CPU0 shadow SPR, read access enable register B Application Reset value: $FFFF\ FFFF_H$

CPU1_ACCENSPRCFG_RDB Offset address: $0E02C_H$

CPU1 shadow SPR, read access enable register B Application Reset value: $FFFF\ FFFF_H$

CPU2_ACCENSPRCFG_RDB Offset address: $0E02C_H$

CPU2 shadow SPR, read access enable register B Application Reset value: $FFFF\ FFFF_H$

CPU3_ACCENSPRCFG_RDB Offset address: $0E02C_H$

CPU3 shadow SPR, read access enable register B Application Reset value: $FFFF\ FFFF_H$

CPU4_ACCENSPRCFG_RDB Offset address: $0E02C_H$

CPU4 shadow SPR, read access enable register B Application Reset value: $FFFF\ FFFF_H$

CPU5_ACCENSPRCFG_RDB Offset address: $0E02C_H$

CPU5 shadow SPR, read access enable register B Application Reset value: $FFFF\ FFFF_H$

CPUCS_ACCENSPRCFG_RDB Offset address: $0E02C_H$

CPUcs shadow SPR, read access enable register B Application Reset value: $0000\ 4000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN63	EN62	EN61	EN60	EN59	EN58	EN57	EN56	EN55	EN54	EN53	EN52	EN51	EN50	EN49	EN48
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN47	EN46	EN45	EN44	EN43	EN42	EN41	EN40	EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-63)	q-32	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access

4.10.4.1.33 CPU0 PROTSFR register endinit

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner and in some PROT states, the secure master to update the PROT state as described in the PROT mechanism.

The PROTSFRE register controls lock / unlock of the local Endinit (E) protected control registers.

CPU0_PROTSFRE	Offset address:	$0E080_H$
CPU0 PROTSFR register endinit	Application Reset value:	$0000\ 0000_H$
CPU1_PROTSFRE	Offset address:	$0E080_H$
CPU1 PROTSFR register endinit	Application Reset value:	$0000\ 0000_H$
CPU2_PROTSFRE	Offset address:	$0E080_H$
CPU2 PROTSFR register endinit	Application Reset value:	$0000\ 0000_H$
CPU3_PROTSFRE	Offset address:	$0E080_H$
CPU3 PROTSFR register endinit	Application Reset value:	$0000\ 0000_H$
CPU4_PROTSFRE	Offset address:	$0E080_H$
CPU4 PROTSFR register endinit	Application Reset value:	$0000\ 0000_H$
CPU5_PROTSFRE	Offset address:	$0E080_H$
CPU5 PROTSFR register endinit	Application Reset value:	$0000\ 0000_H$
CPUCS_PROTSFRE	Offset address:	$0E080_H$
CPUcs PROTSFR register endinit	Application Reset value:	$4C00\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OWEN	**ODEF**	**TAGID**						**PRSEN**	**PRS**			**VMEN**	**VM**		
w	rw	rw						rw	rw			rw	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												**SWEN**	**STATE**		
r												w	rwh		

Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)

(table continues...)

(continued)

Field	Bits	Type	Description
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
VM	18:16	rw	**Virtual machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual machine definition enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.
PRSEN	23	rw	**Protection set definition enable for PROT owner** 0_B **Disable**: PRS is part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; should be written with 0.

Table 314 Access mode restrictions sorted by descending priority

Applies to CPU0_PROTSFRE
Applies to CPU1_PROTSFRE
Applies to CPU2_PROTSFRE
Applies to CPU3_PROTSFRE
Applies to CPU4_PROTSFRE
Applies to CPU5_PROTSFRE
Applies to CPUCS_PROTSFRE

Mode name	Access mode		Description
write 1 to .**SWEN**	rwh	STATE	

(table continues...)

Table 314 (continued) Access mode restrictions sorted by descending priority

Applies to CPU0_PROTSFRE
Applies to CPU1_PROTSFRE
Applies to CPU2_PROTSFRE
Applies to CPU3_PROTSFRE
Applies to CPU4_PROTSFRE
Applies to CPU5_PROTSFRE
Applies to CPUCS_PROTSFRE

Mode name	Access mode		Description
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

4.10.4.1.34 CPU0 PROTSFR register safe endinit

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner and in some PROT states, the secure master to update the PROT state as described in the PROT mechanism.

The PROTSFRSE register controls lock / unlock of the local Safe Endinit (SE) protected control registers.

CPU0_PROTSFRSE Offset address: $0E088_H$
CPU0 PROTSFR register safe endinit Application Reset value: $0000\ 0000_H$

CPU1_PROTSFRSE Offset address: $0E088_H$
CPU1 PROTSFR register safe endinit Application Reset value: $0000\ 0000_H$

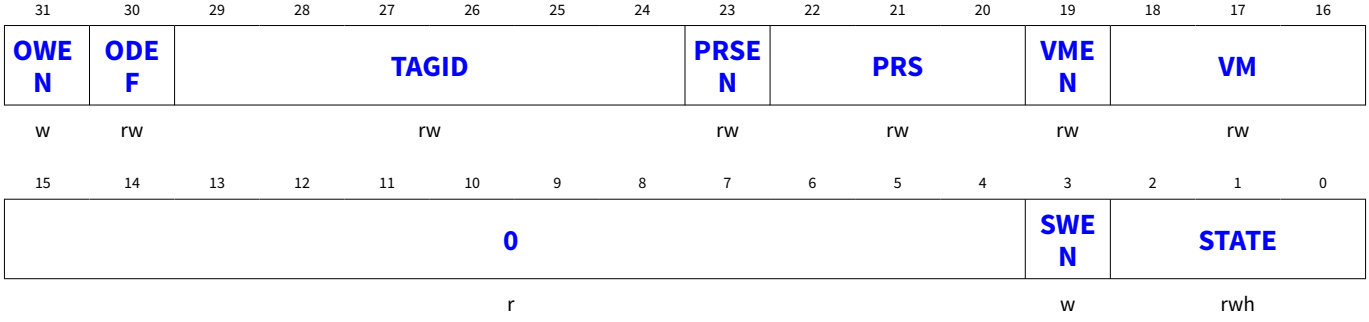
CPU2_PROTSFRSE Offset address: $0E088_H$
CPU2 PROTSFR register safe endinit Application Reset value: $0000\ 0000_H$

CPU3_PROTSFRSE Offset address: $0E088_H$
CPU3 PROTSFR register safe endinit Application Reset value: $0000\ 0000_H$

CPU4_PROTSFRSE Offset address: $0E088_H$
CPU4 PROTSFR register safe endinit Application Reset value: $0000\ 0000_H$

CPU5_PROTSFRSE Offset address: $0E088_H$
CPU5 PROTSFR register safe endinit Application Reset value: $0000\ 0000_H$

CPUCS_PROTSFRSE Offset address: $0E088_H$
CPUcs PROTSFR register safe endinit Application Reset value: $4C00\ 0000_H$



Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
VM	18:16	rw	**Virtual machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual machine definition enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.
PRSEN	23	rw	**Protection set definition enable for PROT owner** 0_B **Disable**: PRS is part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; should be written with 0.

Table 315 Access mode restrictions sorted by descending priority

Applies to CPU0_PROTSFRSE
Applies to CPU1_PROTSFRSE
Applies to CPU2_PROTSFRSE
Applies to CPU3_PROTSFRSE
Applies to CPU4_PROTSFRSE
Applies to CPU5_PROTSFRSE
Applies to CPUCS_PROTSFRSE

Mode name	Access mode		Description
write 1 to .**SWEN**	rwh	STATE	
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

4.10.4.1.35 CPU0 shadow SFR, write access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

CPU0_ACCENSFRCFG_WRA Offset address: $0E0A0_H$
CPU0 shadow SFR, write access enable register A Application Reset value: $1000\ 0003_H$
CPU1_ACCENSFRCFG_WRA Offset address: $0E0A0_H$
CPU1 shadow SFR, write access enable register A Application Reset value: $1000\ 0003_H$
CPU2_ACCENSFRCFG_WRA Offset address: $0E0A0_H$
CPU2 shadow SFR, write access enable register A Application Reset value: $1000\ 0003_H$
CPU3_ACCENSFRCFG_WRA Offset address: $0E0A0_H$
CPU3 shadow SFR, write access enable register A Application Reset value: $1000\ 0003_H$
CPU4_ACCENSFRCFG_WRA Offset address: $0E0A0_H$
CPU4 shadow SFR, write access enable register A Application Reset value: $1000\ 0003_H$
CPU5_ACCENSFRCFG_WRA Offset address: $0E0A0_H$
CPU5 shadow SFR, write access enable register A Application Reset value: $1000\ 0003_H$
CPUCS_ACCENSFRCFG_WRA Offset address: $0E0A0_H$
CPUcs shadow SFR, write access enable register A Reset values see: Table 316

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	**EN30**	**EN29**	**EN28**	**EN27**	**EN26**	**EN25**	**EN24**	**EN23**	**EN22**	**EN21**	**EN20**	**EN19**	**EN18**	**EN17**	**EN16**
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	**EN14**	**EN13**	**EN12**	**EN11**	**EN10**	**EN09**	**EN08**	**EN07**	**EN06**	**EN05**	**EN04**	**EN03**	**EN02**	**EN01**	**EN00**
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access

Table 316 Reset values of CPUCS_ACCENSFRCFG_WRA

Reset type	Reset value	Note
Application Reset	0000 1000$_H$	CPUcs, CSRM debug disabled
After Boot-FW Value	1000 3000$_H$	CPUcs, CSRM debug enabled

4.10.4.1.36 CPU0 shadow SFR, write access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

CPU0_ACCENSFRCFG_WRB Offset address: 0E0A4$_H$

CPU0 shadow SFR, write access enable register B Application Reset value: 0000 0000$_H$

CPU1_ACCENSFRCFG_WRB Offset address: 0E0A4$_H$

CPU1 shadow SFR, write access enable register B Application Reset value: 0000 0000$_H$

CPU2_ACCENSFRCFG_WRB Offset address: 0E0A4$_H$

CPU2 shadow SFR, write access enable register B Application Reset value: 0000 0000$_H$

CPU3_ACCENSFRCFG_WRB Offset address: 0E0A4$_H$

CPU3 shadow SFR, write access enable register B Application Reset value: 0000 0000$_H$

CPU4_ACCENSFRCFG_WRB Offset address: 0E0A4$_H$

CPU4 shadow SFR, write access enable register B Application Reset value: 0000 0000$_H$

CPU5_ACCENSFRCFG_WRB Offset address: 0E0A4$_H$

CPU5 shadow SFR, write access enable register B Application Reset value: 0000 0000$_H$

CPUCS_ACCENSFRCFG_WRB Offset address: 0E0A4$_H$

CPUcs shadow SFR, write access enable register B Reset values see: Table 317

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN63	EN62	EN61	EN60	EN59	EN58	EN57	EN56	EN55	EN54	EN53	EN52	EN51	EN50	EN49	EN48
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN47	EN46	EN45	EN44	EN43	EN42	EN41	EN40	EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-63)	q-32	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access

Table 317 Reset values of CPUCS_ACCENSFRCFG_WRB

Reset type	Reset value	Note
Application Reset	0000 0000$_H$	CPUcs, CSRM debug disabled
After Boot-FW Value	0000 4000$_H$	CPUcs, CSRM debug enabled

4.10.4.1.37 CPU0 shadow SFR, read access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

CPU0_ACCENSFRCFG_RDA Offset address: 0E0A8$_H$
CPU0 shadow SFR, read access enable register A Application Reset value: FFFF FFFF$_H$
CPU1_ACCENSFRCFG_RDA Offset address: 0E0A8$_H$
CPU1 shadow SFR, read access enable register A Application Reset value: FFFF FFFF$_H$
CPU2_ACCENSFRCFG_RDA Offset address: 0E0A8$_H$
CPU2 shadow SFR, read access enable register A Application Reset value: FFFF FFFF$_H$
CPU3_ACCENSFRCFG_RDA Offset address: 0E0A8$_H$
CPU3 shadow SFR, read access enable register A Application Reset value: FFFF FFFF$_H$
CPU4_ACCENSFRCFG_RDA Offset address: 0E0A8$_H$
CPU4 shadow SFR, read access enable register A Application Reset value: FFFF FFFF$_H$
CPU5_ACCENSFRCFG_RDA Offset address: 0E0A8$_H$
CPU5 shadow SFR, read access enable register A Application Reset value: FFFF FFFF$_H$
CPUCS_ACCENSFRCFG_RDA Offset address: 0E0A8$_H$
CPUcs shadow SFR, read access enable register A Reset values see: Table 318

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access

Table 318 Reset values of CPUCS_ACCENSFRCFG_RDA

Reset type	Reset value	Note
Application Reset	0000 1000_H	CPUcs, CSRM debug disabled
After Boot-FW Value	1000 3000_H	CPUcs, CSRM debug enabled

4.10.4.1.38 CPU0 shadow SFR, read access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

CPU0_ACCENSFRCFG_RDB Offset address: $0E0AC_H$

CPU0 shadow SFR, read access enable register B Application Reset value: FFFF $FFFF_H$

CPU1_ACCENSFRCFG_RDB Offset address: $0E0AC_H$

CPU1 shadow SFR, read access enable register B Application Reset value: FFFF $FFFF_H$

CPU2_ACCENSFRCFG_RDB Offset address: $0E0AC_H$

CPU2 shadow SFR, read access enable register B Application Reset value: FFFF $FFFF_H$

CPU3_ACCENSFRCFG_RDB Offset address: $0E0AC_H$

CPU3 shadow SFR, read access enable register B Application Reset value: FFFF $FFFF_H$

CPU4_ACCENSFRCFG_RDB Offset address: $0E0AC_H$

CPU4 shadow SFR, read access enable register B Application Reset value: FFFF $FFFF_H$

CPU5_ACCENSFRCFG_RDB Offset address: $0E0AC_H$

CPU5 shadow SFR, read access enable register B Application Reset value: FFFF $FFFF_H$

CPUCS_ACCENSFRCFG_RDB Offset address: $0E0AC_H$

CPUcs shadow SFR, read access enable register B Reset values see: Table 319

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN63	**EN62**	**EN61**	**EN60**	**EN59**	**EN58**	**EN57**	**EN56**	**EN55**	**EN54**	**EN53**	**EN52**	**EN51**	**EN50**	**EN49**	**EN48**
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN47	**EN46**	**EN45**	**EN44**	**EN43**	**EN42**	**EN41**	**EN40**	**EN39**	**EN38**	**EN37**	**EN36**	**EN35**	**EN34**	**EN33**	**EN32**
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-63)	q-32	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access

Table 319 Reset values of CPUCS_ACCENSFRCFG_RDB

Reset type	Reset value	Note
Application Reset	0000 0000$_H$	CPUcs, CSRM debug disabled
After Boot-FW Value	0000 4000$_H$	CPUcs, CSRM debug enabled

4.10.4.1.39 CPU0 PROTRST register endinit

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner and in some PROT states, the secure master to update the PROT state as described in the PROT mechanism.

The PROTRSTE register controls lock / unlock of the local Endinit (E) protected control registers associated with the kernel reset resource.

CPU0_PROTRSTE Offset address: 0E0C0$_H$

CPU0 PROTRST register endinit Reset values see: Table 321

CPU1_PROTRSTE Offset address: 0E0C0$_H$

CPU1 PROTRST register endinit Application Reset value: 0000 0000$_H$

CPU2_PROTRSTE Offset address: 0E0C0$_H$

CPU2 PROTRST register endinit Application Reset value: 0000 0000$_H$

CPU3_PROTRSTE Offset address: 0E0C0$_H$

CPU3 PROTRST register endinit Application Reset value: 0000 0000$_H$

CPU4_PROTRSTE Offset address: 0E0C0$_H$

CPU4 PROTRST register endinit Application Reset value: 0000 0000$_H$

CPU5_PROTRSTE Offset address: 0E0C0$_H$

CPU5 PROTRST register endinit Application Reset value: 0000 0000$_H$

CPUCS_PROTRSTE Offset address: 0E0C0$_H$

CPUcs PROTRST register endinit Application Reset value: 4C00 0000$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OWEN	ODEF	TAGID						PRSEN	PRS			VMEN	VM		
w	rw	rw						rw	rw			rw	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												SWEN	STATE		
r												w	rwh		

Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
VM	18:16	rw	**Virtual machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual machine definition enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.
PRSEN	23	rw	**Protection set definition enable for PROT owner** 0_B **Disable**: PRS is part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; should be written with 0.

Table 320 Access mode restrictions sorted by descending priority

Applies to CPU0_PROTRSTE
Applies to CPU1_PROTRSTE
Applies to CPU2_PROTRSTE
Applies to CPU3_PROTRSTE
Applies to CPU4_PROTRSTE
Applies to CPU5_PROTRSTE
Applies to CPUCS_PROTRSTE

Mode name	Access mode		Description
write 1 to .**SWEN**	rwh	STATE	
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

Table 321 Reset values of CPU0_PROTRSTE

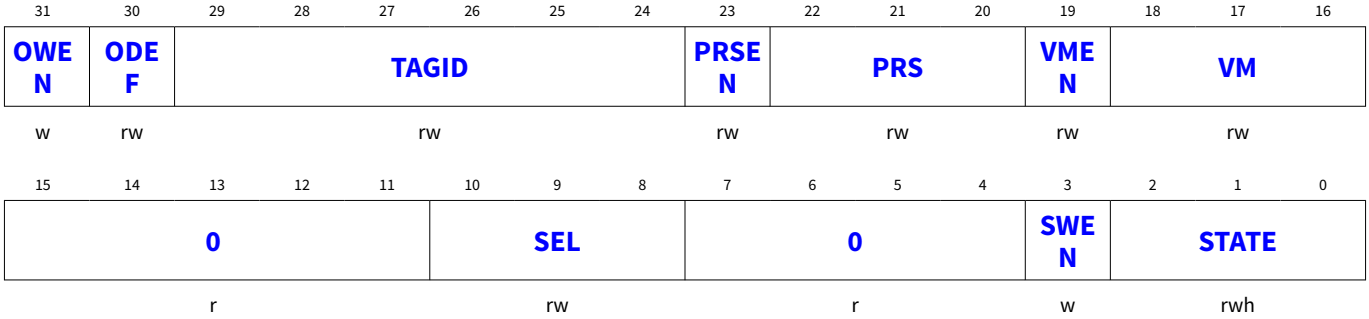
Reset type	Reset value	Note
After Boot-FW Value	$7F00\ 0007_H$	CPU0

4.10.4.1.40 CPU0 PROTSTM Register Safe Endinit

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner and in some PROT states, the secure master to update the PROT state as described in the PROT mechanism.

The PROTSTMSE register controls lock / unlock of the local Safety Endinit (SE) protected control registers associated with the STM resource.

CPU0_PROTSTMSE	Offset address:	$0E0D8_H$
CPU0 PROTSTM Register Safe Endinit	Application Reset value:	$0000\ 0000_H$
CPU1_PROTSTMSE	Offset address:	$0E0D8_H$
CPU1 PROTSTM Register Safe Endinit	Application Reset value:	$0000\ 0000_H$
CPU2_PROTSTMSE	Offset address:	$0E0D8_H$
CPU2 PROTSTM Register Safe Endinit	Application Reset value:	$0000\ 0000_H$
CPU3_PROTSTMSE	Offset address:	$0E0D8_H$
CPU3 PROTSTM Register Safe Endinit	Application Reset value:	$0000\ 0000_H$
CPU4_PROTSTMSE	Offset address:	$0E0D8_H$
CPU4 PROTSTM Register Safe Endinit	Application Reset value:	$0000\ 0000_H$
CPU5_PROTSTMSE	Offset address:	$0E0D8_H$
CPU5 PROTSTM Register Safe Endinit	Application Reset value:	$0000\ 0000_H$
CPUCS_PROTSTMSE	Offset address:	$0E0D8_H$
CPUcs PROTSTM Register Safe Endinit	Application Reset value:	$4C00\ 0000_H$



Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
SEL	10:8	rw	**Range selection** Returns the number of the range being configured when read. Can be written by the PROT owner or CS master to select the range number to be configured if the state transition is valid in the PROT state diagram. 000_B **Const0**: Range 0 selected for configuration … 111_B **Const7**: Range 7 selected for configuration
VM	18:16	rw	**Virtual machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual machine definition enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.

(table continues...)

(continued)

Field	Bits	Type	Description
PRSEN	23	rw	**Protection set definition enable for PROT owner** 0_B **Disable**: PRS is part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	7:4, 15:11	r	**Reserved** Read as 0; should be written with 0.

Table 322 Access mode restrictions sorted by descending priority

Applies to CPU0_PROTSTMSE
Applies to CPU1_PROTSTMSE
Applies to CPU2_PROTSTMSE
Applies to CPU3_PROTSTMSE
Applies to CPU4_PROTSTMSE
Applies to CPU5_PROTSTMSE
Applies to CPUCS_PROTSTMSE

Mode name	Access mode		Description
write 1 to .**SWEN**	rwh	STATE	
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

4.10.4.1.41 CPU0 shadow STM, write access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

CPU0_ACCENSTMCFG_WRA Offset address: $0E0E0_H$

CPU0 shadow STM, write access enable register A Application Reset value: $1000\ 0003_H$

CPU1_ACCENSTMCFG_WRA Offset address: $0E0E0_H$

CPU1 shadow STM, write access enable register A Application Reset value: $1000\ 0003_H$

CPU2_ACCENSTMCFG_WRA Offset address: $0E0E0_H$

CPU2 shadow STM, write access enable register A Application Reset value: $1000\ 0003_H$

CPU3_ACCENSTMCFG_WRA Offset address: $0E0E0_H$

CPU3 shadow STM, write access enable register A Application Reset value: $1000\ 0003_H$

CPU4_ACCENSTMCFG_WRA Offset address: $0E0E0_H$

CPU4 shadow STM, write access enable register A Application Reset value: $1000\ 0003_H$

CPU5_ACCENSTMCFG_WRA Offset address: $0E0E0_H$

CPU5 shadow STM, write access enable register A Application Reset value: $1000\ 0003_H$

CPUCS_ACCENSTMCFG_WRA Offset address: $0E0E0_H$

CPUcs shadow STM, write access enable register A Application Reset value: $0000\ 1000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access

4.10.4.1.42 CPU0 shadow STM, read access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

CPU0_ACCENSTMCFG_RDA	Offset address:	$0E0E8_H$
CPU0 shadow STM, read access enable register A	Application Reset value:	$FFFF\ FFFF_H$
CPU1_ACCENSTMCFG_RDA	Offset address:	$0E0E8_H$
CPU1 shadow STM, read access enable register A	Application Reset value:	$FFFF\ FFFF_H$
CPU2_ACCENSTMCFG_RDA	Offset address:	$0E0E8_H$
CPU2 shadow STM, read access enable register A	Application Reset value:	$FFFF\ FFFF_H$
CPU3_ACCENSTMCFG_RDA	Offset address:	$0E0E8_H$
CPU3 shadow STM, read access enable register A	Application Reset value:	$FFFF\ FFFF_H$
CPU4_ACCENSTMCFG_RDA	Offset address:	$0E0E8_H$
CPU4 shadow STM, read access enable register A	Application Reset value:	$FFFF\ FFFF_H$
CPU5_ACCENSTMCFG_RDA	Offset address:	$0E0E8_H$
CPU5 shadow STM, read access enable register A	Application Reset value:	$FFFF\ FFFF_H$
CPUCS_ACCENSTMCFG_RDA	Offset address:	$0E0E8_H$
CPUcs shadow STM, read access enable register A	Application Reset value:	$0000\ 1000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	**EN30**	**EN29**	**EN28**	**EN27**	**EN26**	**EN25**	**EN24**	**EN23**	**EN22**	**EN21**	**EN20**	**EN19**	**EN18**	**EN17**	**EN16**
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	**EN14**	**EN13**	**EN12**	**EN11**	**EN10**	**EN09**	**EN08**	**EN07**	**EN06**	**EN05**	**EN04**	**EN03**	**EN02**	**EN01**	**EN00**
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access

4.10.4.1.43 CPU0 shadow STM, read access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

CPU0_ACCENSTMCFG_RDB Offset address: $0E0EC_H$

CPU0 shadow STM, read access enable register B Application Reset value: $FFFF\ FFFF_H$

CPU1_ACCENSTMCFG_RDB Offset address: $0E0EC_H$

CPU1 shadow STM, read access enable register B Application Reset value: $FFFF\ FFFF_H$

CPU2_ACCENSTMCFG_RDB Offset address: $0E0EC_H$

CPU2 shadow STM, read access enable register B Application Reset value: $FFFF\ FFFF_H$

CPU3_ACCENSTMCFG_RDB Offset address: $0E0EC_H$

CPU3 shadow STM, read access enable register B Application Reset value: $FFFF\ FFFF_H$

CPU4_ACCENSTMCFG_RDB Offset address: $0E0EC_H$

CPU4 shadow STM, read access enable register B Application Reset value: $FFFF\ FFFF_H$

CPU5_ACCENSTMCFG_RDB Offset address: $0E0EC_H$

CPU5 shadow STM, read access enable register B Application Reset value: $FFFF\ FFFF_H$

CPUCS_ACCENSTMCFG_RDB Offset address: $0E0EC_H$

CPUcs shadow STM, read access enable register B Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN63	EN62	EN61	EN60	EN59	EN58	EN57	EN56	EN55	EN54	EN53	EN52	EN51	EN50	EN49	EN48
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN47	EN46	EN45	EN44	EN43	EN42	EN41	EN40	EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-63)	q-32	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access

4.10.4.1.44 CPU0 SPR range y, write access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

CPU0_ACCENSPRy_WRA (y=0-15) Offset address: $0E200_H+y*20_H$

CPU0 SPR range y, write access enable register A Application Reset value: $1000\ 0003_H$

CPU1_ACCENSPRy_WRA (y=0-15) Offset address: $0E200_H+y*20_H$

CPU1 SPR range y, write access enable register A Application Reset value: $1000\ 0003_H$

CPU2_ACCENSPRy_WRA (y=0-15) Offset address: $0E200_H+y*20_H$

CPU2 SPR range y, write access enable register A Application Reset value: $1000\ 0003_H$

CPU3_ACCENSPRy_WRA (y=0-15) Offset address: $0E200_H+y*20_H$

CPU3 SPR range y, write access enable register A Application Reset value: $1000\ 0003_H$

CPU4_ACCENSPRy_WRA (y=0-15) Offset address: $0E200_H+y*20_H$

CPU4 SPR range y, write access enable register A Application Reset value: $1000\ 0003_H$

CPU5_ACCENSPRy_WRA (y=0-15) Offset address: $0E200_H+y*20_H$

CPU5 SPR range y, write access enable register A Application Reset value: $1000\ 0003_H$

CPUCS_ACCENSPRy_WRA (y=0-15) Offset address: $0E200_H+y*20_H$

CPUcs SPR range y, write access enable register A Reset values see: Table 323

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
ENq (q=00-31)	q	rh	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access

Table 323 Reset values of CPUCS_ACCENSPRy_WRA (y=0-15)

Reset type	Reset value	Note
Application Reset	$0000\ 1000_H$	CPUcs, CSRM debug disabled
After Boot-FW Value	$1000\ 3000_H$	CPUcs, y=0, CSRM debug enabled

4.10.4.1.45 CPU0 SPR range y, write access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

CPU0_ACCENSPRy_WRB (y=0-15) Offset address: $0E204_H+y*20_H$

CPU0 SPR range y, write access enable register B Application Reset value: $0000\ 0000_H$

CPU1_ACCENSPRy_WRB (y=0-15) Offset address: $0E204_H+y*20_H$

CPU1 SPR range y, write access enable register B Application Reset value: $0000\ 0000_H$

CPU2_ACCENSPRy_WRB (y=0-15) Offset address: $0E204_H+y*20_H$

CPU2 SPR range y, write access enable register B Application Reset value: $0000\ 0000_H$

CPU3_ACCENSPRy_WRB (y=0-15) Offset address: $0E204_H+y*20_H$

CPU3 SPR range y, write access enable register B Application Reset value: $0000\ 0000_H$

CPU4_ACCENSPRy_WRB (y=0-15) Offset address: $0E204_H+y*20_H$

CPU4 SPR range y, write access enable register B Application Reset value: $0000\ 0000_H$

CPU5_ACCENSPRy_WRB (y=0-15) Offset address: $0E204_H+y*20_H$

CPU5 SPR range y, write access enable register B Application Reset value: $0000\ 0000_H$

CPUCS_ACCENSPRy_WRB (y=0-15) Offset address: $0E204_H+y*20_H$

CPUcs SPR range y, write access enable register B Reset values see: Table 324

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN63	EN62	EN61	EN60	EN59	EN58	EN57	EN56	EN55	EN54	EN53	EN52	EN51	EN50	EN49	EN48
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN47	EN46	EN45	EN44	EN43	EN42	EN41	EN40	EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
ENq (q=32-63)	q-32	rh	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access

Table 324 Reset values of CPUCS_ACCENSPRy_WRB (y=0-15)

Reset type	Reset value	Note
Application Reset	$0000\ 0000_H$	CPUcs, CSRM debug disabled
After Boot-FW Value	$0000\ 4000_H$	CPUcs, y=0, CSRM debug enabled

4.10.4.1.46 CPU0 SPR range y, read access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

CPU0_ACCENSPRy_RDA (y=0-15) Offset address: $0E208_H+y*20_H$

CPU0 SPR range y, read access enable register A Application Reset value: FFFF FFFF$_H$

CPU1_ACCENSPRy_RDA (y=0-15) Offset address: $0E208_H+y*20_H$

CPU1 SPR range y, read access enable register A Application Reset value: FFFF FFFF$_H$

CPU2_ACCENSPRy_RDA (y=0-15) Offset address: $0E208_H+y*20_H$

CPU2 SPR range y, read access enable register A Application Reset value: FFFF FFFF$_H$

CPU3_ACCENSPRy_RDA (y=0-15) Offset address: $0E208_H+y*20_H$

CPU3 SPR range y, read access enable register A Application Reset value: FFFF FFFF$_H$

CPU4_ACCENSPRy_RDA (y=0-15) Offset address: $0E208_H+y*20_H$

CPU4 SPR range y, read access enable register A Application Reset value: FFFF FFFF$_H$

CPU5_ACCENSPRy_RDA (y=0-15) Offset address: $0E208_H+y*20_H$

CPU5 SPR range y, read access enable register A Application Reset value: FFFF FFFF$_H$

CPUCS_ACCENSPRy_RDA (y=0-15) Offset address: $0E208_H+y*20_H$

CPUcs SPR range y, read access enable register A Reset values see: Table 325

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	**EN30**	**EN29**	**EN28**	**EN27**	**EN26**	**EN25**	**EN24**	**EN23**	**EN22**	**EN21**	**EN20**	**EN19**	**EN18**	**EN17**	**EN16**
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	**EN14**	**EN13**	**EN12**	**EN11**	**EN10**	**EN09**	**EN08**	**EN07**	**EN06**	**EN05**	**EN04**	**EN03**	**EN02**	**EN01**	**EN00**
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
ENq (q=00-31)	q	rh	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access

Table 325 Reset values of CPUCS_ACCENSPRy_RDA (y=0-15)

Reset type	Reset value	Note
Application Reset	0000 1000$_H$	CPUcs, CSRM debug disabled
After Boot-FW Value	1000 3000$_H$	CPUcs, y=0, CSRM debug enabled

4.10.4.1.47 CPU0 SPR range y, read access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

CPU0_ACCENSPRy_RDB (y=0-15)	Offset address:	$0E20C_H+y*20_H$
CPU0 SPR range y, read access enable register B	Application Reset value:	$FFFF\ FFFF_H$
CPU1_ACCENSPRy_RDB (y=0-15)	Offset address:	$0E20C_H+y*20_H$
CPU1 SPR range y, read access enable register B	Application Reset value:	$FFFF\ FFFF_H$
CPU2_ACCENSPRy_RDB (y=0-15)	Offset address:	$0E20C_H+y*20_H$
CPU2 SPR range y, read access enable register B	Application Reset value:	$FFFF\ FFFF_H$
CPU3_ACCENSPRy_RDB (y=0-15)	Offset address:	$0E20C_H+y*20_H$
CPU3 SPR range y, read access enable register B	Application Reset value:	$FFFF\ FFFF_H$
CPU4_ACCENSPRy_RDB (y=0-15)	Offset address:	$0E20C_H+y*20_H$
CPU4 SPR range y, read access enable register B	Application Reset value:	$FFFF\ FFFF_H$
CPU5_ACCENSPRy_RDB (y=0-15)	Offset address:	$0E20C_H+y*20_H$
CPU5 SPR range y, read access enable register B	Application Reset value:	$FFFF\ FFFF_H$
CPUCS_ACCENSPRy_RDB (y=0-15)	Offset address:	$0E20C_H+y*20_H$
CPUcs SPR range y, read access enable register B	Application Reset value:	$0000\ 4000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN63	**EN62**	**EN61**	**EN60**	**EN59**	**EN58**	**EN57**	**EN56**	**EN55**	**EN54**	**EN53**	**EN52**	**EN51**	**EN50**	**EN49**	**EN48**
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN47	**EN46**	**EN45**	**EN44**	**EN43**	**EN42**	**EN41**	**EN40**	**EN39**	**EN38**	**EN37**	**EN36**	**EN35**	**EN34**	**EN33**	**EN32**
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
ENq (q=32-63)	q-32	rh	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access

4.10.4.1.48 CPU0 SFR, write access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

CPU0_ACCENSFR_WRA — Offset address: $0E600_H$
CPU0 SFR, write access enable register A — Application Reset value: $1000\ 0003_H$

CPU1_ACCENSFR_WRA — Offset address: $0E600_H$
CPU1 SFR, write access enable register A — Application Reset value: $1000\ 0003_H$

CPU2_ACCENSFR_WRA — Offset address: $0E600_H$
CPU2 SFR, write access enable register A — Application Reset value: $1000\ 0003_H$

CPU3_ACCENSFR_WRA — Offset address: $0E600_H$
CPU3 SFR, write access enable register A — Application Reset value: $1000\ 0003_H$

CPU4_ACCENSFR_WRA — Offset address: $0E600_H$
CPU4 SFR, write access enable register A — Application Reset value: $1000\ 0003_H$

CPU5_ACCENSFR_WRA — Offset address: $0E600_H$
CPU5 SFR, write access enable register A — Application Reset value: $1000\ 0003_H$

CPUCS_ACCENSFR_WRA — Offset address: $0E600_H$
CPUcs SFR, write access enable register A — Reset values see: Table 326

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	**EN30**	**EN29**	**EN28**	**EN27**	**EN26**	**EN25**	**EN24**	**EN23**	**EN22**	**EN21**	**EN20**	**EN19**	**EN18**	**EN17**	**EN16**
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	**EN14**	**EN13**	**EN12**	**EN11**	**EN10**	**EN09**	**EN08**	**EN07**	**EN06**	**EN05**	**EN04**	**EN03**	**EN02**	**EN01**	**EN00**
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
ENq (q=00-31)	q	rh	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access

Table 326 Reset values of CPUCS_ACCENSFR_WRA

Reset type	Reset value	Note
Application Reset	$0000\ 1000_H$	CPUcs, CSRM debug disabled
After Boot-FW Value	$1000\ 3000_H$	CPUcs, CSRM debug enabled

4.10.4.1.49 CPU0 SFR, write access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

CPU0_ACCENSFR_WRB Offset address: $0E604_H$

CPU0 SFR, write access enable register B Application Reset value: $0000\ 0000_H$

CPU1_ACCENSFR_WRB Offset address: $0E604_H$

CPU1 SFR, write access enable register B Application Reset value: $0000\ 0000_H$

CPU2_ACCENSFR_WRB Offset address: $0E604_H$

CPU2 SFR, write access enable register B Application Reset value: $0000\ 0000_H$

CPU3_ACCENSFR_WRB Offset address: $0E604_H$

CPU3 SFR, write access enable register B Application Reset value: $0000\ 0000_H$

CPU4_ACCENSFR_WRB Offset address: $0E604_H$

CPU4 SFR, write access enable register B Application Reset value: $0000\ 0000_H$

CPU5_ACCENSFR_WRB Offset address: $0E604_H$

CPU5 SFR, write access enable register B Application Reset value: $0000\ 0000_H$

CPUCS_ACCENSFR_WRB Offset address: $0E604_H$

CPUcs SFR, write access enable register B Reset values see: Table 327

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN63	**EN62**	**EN61**	**EN60**	**EN59**	**EN58**	**EN57**	**EN56**	**EN55**	**EN54**	**EN53**	**EN52**	**EN51**	**EN50**	**EN49**	**EN48**
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN47	**EN46**	**EN45**	**EN44**	**EN43**	**EN42**	**EN41**	**EN40**	**EN39**	**EN38**	**EN37**	**EN36**	**EN35**	**EN34**	**EN33**	**EN32**
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
ENq (q=32-63)	q-32	rh	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access

Table 327 Reset values of CPUCS_ACCENSFR_WRB

Reset type	Reset value	Note
Application Reset	$0000\ 0000_H$	CPUcs, CSRM debug disabled
After Boot-FW Value	$0000\ 4000_H$	CPUcs, CSRM debug enabled

4.10.4.1.50 CPU0 SFR, read access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

CPU0_ACCENSFR_RDA Offset address: $0E608_H$

CPU0 SFR, read access enable register A Application Reset value: $FFFF\ FFFF_H$

CPU1_ACCENSFR_RDA Offset address: $0E608_H$

CPU1 SFR, read access enable register A Application Reset value: $FFFF\ FFFF_H$

CPU2_ACCENSFR_RDA Offset address: $0E608_H$

CPU2 SFR, read access enable register A Application Reset value: $FFFF\ FFFF_H$

CPU3_ACCENSFR_RDA Offset address: $0E608_H$

CPU3 SFR, read access enable register A Application Reset value: $FFFF\ FFFF_H$

CPU4_ACCENSFR_RDA Offset address: $0E608_H$

CPU4 SFR, read access enable register A Application Reset value: $FFFF\ FFFF_H$

CPU5_ACCENSFR_RDA Offset address: $0E608_H$

CPU5 SFR, read access enable register A Application Reset value: $FFFF\ FFFF_H$

CPUCS_ACCENSFR_RDA Offset address: $0E608_H$

CPUcs SFR, read access enable register A Reset values see: Table 328

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	**EN30**	**EN29**	**EN28**	**EN27**	**EN26**	**EN25**	**EN24**	**EN23**	**EN22**	**EN21**	**EN20**	**EN19**	**EN18**	**EN17**	**EN16**
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	**EN14**	**EN13**	**EN12**	**EN11**	**EN10**	**EN09**	**EN08**	**EN07**	**EN06**	**EN05**	**EN04**	**EN03**	**EN02**	**EN01**	**EN00**
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
ENq (q=00-31)	q	rh	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access

Table 328 Reset values of CPUCS_ACCENSFR_RDA

Reset type	Reset value	Note
Application Reset	$0000\ 1000_H$	CPUcs, CSRM debug disabled
After Boot-FW Value	$1000\ 3000_H$	CPUcs, CSRM debug enabled

4.10.4.1.51 CPU0 SFR, read access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

CPU0_ACCENSFR_RDB Offset address: $0E60C_H$

CPU0 SFR, read access enable register B Application Reset value: $FFFF\ FFFF_H$

CPU1_ACCENSFR_RDB Offset address: $0E60C_H$

CPU1 SFR, read access enable register B Application Reset value: $FFFF\ FFFF_H$

CPU2_ACCENSFR_RDB Offset address: $0E60C_H$

CPU2 SFR, read access enable register B Application Reset value: $FFFF\ FFFF_H$

CPU3_ACCENSFR_RDB Offset address: $0E60C_H$

CPU3 SFR, read access enable register B Application Reset value: $FFFF\ FFFF_H$

CPU4_ACCENSFR_RDB Offset address: $0E60C_H$

CPU4 SFR, read access enable register B Application Reset value: $FFFF\ FFFF_H$

CPU5_ACCENSFR_RDB Offset address: $0E60C_H$

CPU5 SFR, read access enable register B Application Reset value: $FFFF\ FFFF_H$

CPUCS_ACCENSFR_RDB Offset address: $0E60C_H$

CPUcs SFR, read access enable register B Reset values see: Table 329

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN63	**EN62**	**EN61**	**EN60**	**EN59**	**EN58**	**EN57**	**EN56**	**EN55**	**EN54**	**EN53**	**EN52**	**EN51**	**EN50**	**EN49**	**EN48**
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN47	**EN46**	**EN45**	**EN44**	**EN43**	**EN42**	**EN41**	**EN40**	**EN39**	**EN38**	**EN37**	**EN36**	**EN35**	**EN34**	**EN33**	**EN32**
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
ENq (q=32-63)	q-32	rh	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access

Table 329 Reset values of CPUCS_ACCENSFR_RDB

Reset type	Reset value	Note
Application Reset	$0000\ 0000_H$	CPUcs, CSRM debug disabled
After Boot-FW Value	$0000\ 4000_H$	CPUcs, CSRM debug enabled

4.10.4.1.52 CPU0 VMy STM, write access enable register A, virtualization view

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

CPU0_ACCENSTMVMy_WRA (y=0-7) — Offset address: $0E700_H+y*20_H$

CPU0 VMy STM, write access enable register A, virtualization view — Application Reset value: $1000\ 0003_H$

CPU1_ACCENSTMVMy_WRA (y=0-7) — Offset address: $0E700_H+y*20_H$

CPU1 VMy STM, write access enable register A, virtualization view — Application Reset value: $1000\ 0003_H$

CPU2_ACCENSTMVMy_WRA (y=0-7) — Offset address: $0E700_H+y*20_H$

CPU2 VMy STM, write access enable register A, virtualization view — Application Reset value: $1000\ 0003_H$

CPU3_ACCENSTMVMy_WRA (y=0-7) — Offset address: $0E700_H+y*20_H$

CPU3 VMy STM, write access enable register A, virtualization view — Application Reset value: $1000\ 0003_H$

CPU4_ACCENSTMVMy_WRA (y=0-7) — Offset address: $0E700_H+y*20_H$

CPU4 VMy STM, write access enable register A, virtualization view — Application Reset value: $1000\ 0003_H$

CPU5_ACCENSTMVMy_WRA (y=0-7) — Offset address: $0E700_H+y*20_H$

CPU5 VMy STM, write access enable register A, virtualization view — Application Reset value: $1000\ 0003_H$

CPUCS_ACCENSTM_WRA — Offset address: $0E720_H$

CPUcs STM, write access enable register A — Application Reset value: $0000\ 1000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
ENq (q=00-31)	q	rh	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access

4.10.4.1.53 CPU0 VMy STM, read access enable register A, virtualization view

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

CPU0_ACCENSTMVMy_RDA (y=0-7) — Offset address: $0E708_H+y*20_H$

CPU0 VMy STM, read access enable register A, virtualization view — Application Reset value: $FFFF\ FFFF_H$

CPU1_ACCENSTMVMy_RDA (y=0-7) — Offset address: $0E708_H+y*20_H$

CPU1 VMy STM, read access enable register A, virtualization view — Application Reset value: $FFFF\ FFFF_H$

CPU2_ACCENSTMVMy_RDA (y=0-7) — Offset address: $0E708_H+y*20_H$

CPU2 VMy STM, read access enable register A, virtualization view — Application Reset value: $FFFF\ FFFF_H$

CPU3_ACCENSTMVMy_RDA (y=0-7) — Offset address: $0E708_H+y*20_H$

CPU3 VMy STM, read access enable register A, virtualization view — Application Reset value: $FFFF\ FFFF_H$

CPU4_ACCENSTMVMy_RDA (y=0-7) — Offset address: $0E708_H+y*20_H$

CPU4 VMy STM, read access enable register A, virtualization view — Application Reset value: $FFFF\ FFFF_H$

CPU5_ACCENSTMVMy_RDA (y=0-7) — Offset address: $0E708_H+y*20_H$

CPU5 VMy STM, read access enable register A, virtualization view — Application Reset value: $FFFF\ FFFF_H$

CPUCS_ACCENSTM_RDA — Offset address: $0E728_H$

CPUcs STM, read access enable register A — Application Reset value: $0000\ 1000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
ENq (q=00-31)	q	rh	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access

4.10.4.1.54 CPU0 VMy STM, read access enable register B, virtualization view

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

CPU0_ACCENSTMVMy_RDB (y=0-7) — Offset address: $0E70C_H+y*20_H$

CPU0 VMy STM, read access enable register B, virtualization view — Application Reset value: $FFFF\ FFFF_H$

CPU1_ACCENSTMVMy_RDB (y=0-7) — Offset address: $0E70C_H+y*20_H$

CPU1 VMy STM, read access enable register B, virtualization view — Application Reset value: $FFFF\ FFFF_H$

CPU2_ACCENSTMVMy_RDB (y=0-7) — Offset address: $0E70C_H+y*20_H$

CPU2 VMy STM, read access enable register B, virtualization view — Application Reset value: $FFFF\ FFFF_H$

CPU3_ACCENSTMVMy_RDB (y=0-7) — Offset address: $0E70C_H+y*20_H$

CPU3 VMy STM, read access enable register B, virtualization view — Application Reset value: $FFFF\ FFFF_H$

CPU4_ACCENSTMVMy_RDB (y=0-7) — Offset address: $0E70C_H+y*20_H$

CPU4 VMy STM, read access enable register B, virtualization view — Application Reset value: $FFFF\ FFFF_H$

CPU5_ACCENSTMVMy_RDB (y=0-7) — Offset address: $0E70C_H+y*20_H$

CPU5 VMy STM, read access enable register B, virtualization view — Application Reset value: $FFFF\ FFFF_H$

CPUCS_ACCENSTM_RDB — Offset address: $0E72C_H$

CPUcs STM, read access enable register B — Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN63	EN62	EN61	EN60	EN59	EN58	EN57	EN56	EN55	EN54	EN53	EN52	EN51	EN50	EN49	EN48
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN47	EN46	EN45	EN44	EN43	EN42	EN41	EN40	EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
ENq (q=32-63)	q-32	rh	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access

4.10.4.2 Registers of CPU CFI interface

4.10.4.2.1 Register address space - CPUCFI

Table 330 Registers address space - CPUCFI

Module	Base address	End address	Note
CPU_CFI0	$F8400000_H$	$F840FFFF_H$	SRI slave interface for FSFR
CPU_CFI1	$F8440000_H$	$F844FFFF_H$	SRI slave interface for FSFR
CPU_CFI2	$F8480000_H$	$F848FFFF_H$	SRI slave interface for FSFR
CPU_CFI3	$F84C0000_H$	$F84CFFFF_H$	SRI slave interface for FSFR
CPU_CFI4	$F8500000_H$	$F850FFFF_H$	SRI slave interface for FSFR
CPU_CFI5	$F8540000_H$	$F854FFFF_H$	SRI slave interface for FSFR
CPU_CFICS	$F8580000_H$	$F858FFFF_H$	SRI slave interface for FSFR

4.10.4.2.2 Register overview - access mode glossary

Table 331 Register overview - access mode glossary

Keyword	Description
PFSE	Access protection using PROT register CPU_CFI0_PROTPFSE or CPU_CFI1_PROTPFSE or CPU_CFI2_PROTPFSE or CPU_CFI3_PROTPFSE or CPU_CFI4_PROTPFSE or CPU_CFI5_PROTPFSE or CPU_CFICS_PROTPFSE.
PFSFRE	Access protection using PROT register CPU_CFI0_PROTPFSFRE or CPU_CFI1_PROTPFSFRE or CPU_CFI2_PROTPFSFRE or CPU_CFI3_PROTPFSFRE or CPU_CFI4_PROTPFSFRE or CPU_CFI5_PROTPFSFRE or CPU_CFICS_PROTPFSFRE.
PFSFRSE	Access protection using PROT register CPU_CFI0_PROTPFSFRSE or CPU_CFI1_PROTPFSFRSE or CPU_CFI2_PROTPFSFRSE or CPU_CFI3_PROTPFSFRSE or CPU_CFI4_PROTPFSFRSE or CPU_CFI5_PROTPFSFRSE or CPU_CFICS_PROTPFSFRSE.

(table continues...)

Table 331 (continued) Register overview - access mode glossary

Keyword	Description
APU-P4	Protection group consisting of registers CPU_CFI0_ACCENPFSFRCFG_WRA, CPU_CFI0_ACCENPFSFRCFG_WRB, CPU_CFI0_ACCENPFSFRCFG_RDA, CPU_CFI0_ACCENPFSFRCFG_RDB, CPU_CFI0_ACCENPFSFRCFG_VM , CPU_CFI0_ACCENPFSFRCFG_PRS , CPU_CFI0_ACCENPFSFR_WRA, CPU_CFI0_ACCENPFSFR_WRB, CPU_CFI0_ACCENPFSFR_RDA, CPU_CFI0_ACCENPFSFR_RDB, CPU_CFI0_ACCENPFSFR_VM , CPU_CFI0_ACCENPFSFR_PRS or CPU_CFI1_ACCENPFSFRCFG_WRA, CPU_CFI1_ACCENPFSFRCFG_WRB, CPU_CFI1_ACCENPFSFRCFG_RDA, CPU_CFI1_ACCENPFSFRCFG_RDB, CPU_CFI1_ACCENPFSFRCFG_VM , CPU_CFI1_ACCENPFSFRCFG_PRS , CPU_CFI1_ACCENPFSFR_WRA, CPU_CFI1_ACCENPFSFR_WRB, CPU_CFI1_ACCENPFSFR_RDA, CPU_CFI1_ACCENPFSFR_RDB, CPU_CFI1_ACCENPFSFR_VM , CPU_CFI1_ACCENPFSFR_PRS or CPU_CFI2_ACCENPFSFRCFG_WRA, CPU_CFI2_ACCENPFSFRCFG_WRB, CPU_CFI2_ACCENPFSFRCFG_RDA, CPU_CFI2_ACCENPFSFRCFG_RDB, CPU_CFI2_ACCENPFSFRCFG_VM , CPU_CFI2_ACCENPFSFRCFG_PRS , CPU_CFI2_ACCENPFSFR_WRA, CPU_CFI2_ACCENPFSFR_WRB, CPU_CFI2_ACCENPFSFR_RDA, CPU_CFI2_ACCENPFSFR_RDB, CPU_CFI2_ACCENPFSFR_VM , CPU_CFI2_ACCENPFSFR_PRS or CPU_CFI3_ACCENPFSFRCFG_WRA, CPU_CFI3_ACCENPFSFRCFG_WRB, CPU_CFI3_ACCENPFSFRCFG_RDA, CPU_CFI3_ACCENPFSFRCFG_RDB, CPU_CFI3_ACCENPFSFRCFG_VM , CPU_CFI3_ACCENPFSFRCFG_PRS , CPU_CFI3_ACCENPFSFR_WRA, CPU_CFI3_ACCENPFSFR_WRB, CPU_CFI3_ACCENPFSFR_RDA, CPU_CFI3_ACCENPFSFR_RDB, CPU_CFI3_ACCENPFSFR_VM , CPU_CFI3_ACCENPFSFR_PRS or CPU_CFI4_ACCENPFSFRCFG_WRA, CPU_CFI4_ACCENPFSFRCFG_WRB, CPU_CFI4_ACCENPFSFRCFG_RDA, CPU_CFI4_ACCENPFSFRCFG_RDB, CPU_CFI4_ACCENPFSFRCFG_VM , CPU_CFI4_ACCENPFSFRCFG_PRS , CPU_CFI4_ACCENPFSFR_WRA, CPU_CFI4_ACCENPFSFR_WRB, CPU_CFI4_ACCENPFSFR_RDA, CPU_CFI4_ACCENPFSFR_RDB, CPU_CFI4_ACCENPFSFR_VM , CPU_CFI4_ACCENPFSFR_PRS or CPU_CFI5_ACCENPFSFRCFG_WRA, CPU_CFI5_ACCENPFSFRCFG_WRB, CPU_CFI5_ACCENPFSFRCFG_RDA, CPU_CFI5_ACCENPFSFRCFG_RDB, CPU_CFI5_ACCENPFSFRCFG_VM , CPU_CFI5_ACCENPFSFRCFG_PRS , CPU_CFI5_ACCENPFSFR_WRA, CPU_CFI5_ACCENPFSFR_WRB, CPU_CFI5_ACCENPFSFR_RDA, CPU_CFI5_ACCENPFSFR_RDB, CPU_CFI5_ACCENPFSFR_VM , CPU_CFI5_ACCENPFSFR_PRS or CPU_CFICS_ACCENPFSFRCFG_WRA, CPU_CFICS_ACCENPFSFRCFG_WRB, CPU_CFICS_ACCENPFSFRCFG_RDA, CPU_CFICS_ACCENPFSFRCFG_RDB, CPU_CFICS_ACCENPFSFRCFG_VM , CPU_CFICS_ACCENPFSFRCFG_PRS , CPU_CFICS_ACCENPFSFR_WRA, CPU_CFICS_ACCENPFSFR_WRB, CPU_CFICS_ACCENPFSFR_RDA, CPU_CFICS_ACCENPFSFR_RDB, CPU_CFICS_ACCENPFSFR_VM , CPU_CFICS_ACCENPFSFR_PRS .
P4	Access protection using *APU-P4* registers.

(table continues...)

Table 331 (continued) Register overview - access mode glossary

Keyword	Description
APU-P5	Protection group consisting of registers CPU_CFI0_ACCENPFCFG_RDA, CPU_CFI0_ACCENPFCFG_RDB, CPU_CFI0_ACCENPFCFG_VM , CPU_CFI0_ACCENPFCFG_PRS , CPU_CFI0_ACCENPFCFG_RGNLA , CPU_CFI0_ACCENPFCFG_RGNUA , CPU_CFI0_ACCENPFy_RDA, CPU_CFI0_ACCENPFy_RDB, CPU_CFI0_ACCENPFy_VM , CPU_CFI0_ACCENPFy_PRS , CPU_CFI0_ACCENPFy_RGNLA , CPU_CFI0_ACCENPFy_RGNUA or CPU_CFI1_ACCENPFCFG_RDA, CPU_CFI1_ACCENPFCFG_RDB, CPU_CFI1_ACCENPFCFG_VM , CPU_CFI1_ACCENPFCFG_PRS , CPU_CFI1_ACCENPFCFG_RGNLA , CPU_CFI1_ACCENPFCFG_RGNUA , CPU_CFI1_ACCENPFy_RDA, CPU_CFI1_ACCENPFy_RDB, CPU_CFI1_ACCENPFy_VM , CPU_CFI1_ACCENPFy_PRS , CPU_CFI1_ACCENPFy_RGNLA , CPU_CFI1_ACCENPFy_RGNUA or CPU_CFI2_ACCENPFCFG_RDA, CPU_CFI2_ACCENPFCFG_RDB, CPU_CFI2_ACCENPFCFG_VM , CPU_CFI2_ACCENPFCFG_PRS , CPU_CFI2_ACCENPFCFG_RGNLA , CPU_CFI2_ACCENPFCFG_RGNUA , CPU_CFI2_ACCENPFy_RDA, CPU_CFI2_ACCENPFy_RDB, CPU_CFI2_ACCENPFy_VM , CPU_CFI2_ACCENPFy_PRS , CPU_CFI2_ACCENPFy_RGNLA , CPU_CFI2_ACCENPFy_RGNUA or CPU_CFI3_ACCENPFCFG_RDA, CPU_CFI3_ACCENPFCFG_RDB, CPU_CFI3_ACCENPFCFG_VM , CPU_CFI3_ACCENPFCFG_PRS , CPU_CFI3_ACCENPFCFG_RGNLA , CPU_CFI3_ACCENPFCFG_RGNUA , CPU_CFI3_ACCENPFy_RDA, CPU_CFI3_ACCENPFy_RDB, CPU_CFI3_ACCENPFy_VM , CPU_CFI3_ACCENPFy_PRS , CPU_CFI3_ACCENPFy_RGNLA , CPU_CFI3_ACCENPFy_RGNUA or CPU_CFI4_ACCENPFCFG_RDA, CPU_CFI4_ACCENPFCFG_RDB, CPU_CFI4_ACCENPFCFG_VM , CPU_CFI4_ACCENPFCFG_PRS , CPU_CFI4_ACCENPFCFG_RGNLA , CPU_CFI4_ACCENPFCFG_RGNUA , CPU_CFI4_ACCENPFy_RDA, CPU_CFI4_ACCENPFy_RDB, CPU_CFI4_ACCENPFy_VM , CPU_CFI4_ACCENPFy_PRS , CPU_CFI4_ACCENPFy_RGNLA , CPU_CFI4_ACCENPFy_RGNUA or CPU_CFI5_ACCENPFCFG_RDA, CPU_CFI5_ACCENPFCFG_RDB, CPU_CFI5_ACCENPFCFG_VM , CPU_CFI5_ACCENPFCFG_PRS , CPU_CFI5_ACCENPFCFG_RGNLA , CPU_CFI5_ACCENPFCFG_RGNUA , CPU_CFI5_ACCENPFy_RDA, CPU_CFI5_ACCENPFy_RDB, CPU_CFI5_ACCENPFy_VM , CPU_CFI5_ACCENPFy_PRS , CPU_CFI5_ACCENPFy_RGNLA , CPU_CFI5_ACCENPFy_RGNUA or CPU_CFICS_ACCENPFCFG_RDA, CPU_CFICS_ACCENPFCFG_RDB, CPU_CFICS_ACCENPFCFG_VM , CPU_CFICS_ACCENPFCFG_PRS , CPU_CFICS_ACCENPFCFG_RGNLA , CPU_CFICS_ACCENPFCFG_RGNUA , CPU_CFICS_ACCENPFy_RDA, CPU_CFICS_ACCENPFy_RDB, CPU_CFICS_ACCENPFy_VM , CPU_CFICS_ACCENPFy_PRS , CPU_CFICS_ACCENPFy_RGNLA , CPU_CFICS_ACCENPFy_RGNUA .
P5	Access protection using *APU-P5* registers.
32	Access only when using 32-bit width.
PROT	Access restrictions as defined in the PROT register access rules.

4.10.4.2.3 Register overview - CPU_CFI0 (ascending offset address)

Table 332 Register overview - CPU_CFI0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU_CFI0_PROTPFSFRE	CPU0 PROTPFSFR register endinit	00000_H	32	SV, 32, PROT	Application Reset	614
CPU_CFI0_PROTPFSFRSE	CPU0 PROTPFSFR register safe endinit	00008_H	32	SV, 32, PROT	Application Reset	617
CPU_CFI0_ACCENPFSFRCFG_WRA	CPU0 shadow NVM SFR write access enable register A	00020_H	32	PFSFRSE, SV, 32	Application Reset	619
CPU_CFI0_ACCENPFSFRCFG_WRB	CPU0 shadow NVM SFR write access enable register B	00024_H	32	PFSFRSE, SV, 32	Application Reset	620
CPU_CFI0_ACCENPFSFRCFG_RDA	CPU0 shadow NVM SFR read access enable register A	00028_H	32	PFSFRSE, SV, 32	Application Reset	621
CPU_CFI0_ACCENPFSFRCFG_RDB	CPU0 shadow NVM SFR read access enable register B	$0002C_H$	32	PFSFRSE, SV, 32	Application Reset	622
CPU_CFI0_ACCENPFSFRCFG_VM	CPU0 shadow NVM SFR VM access enable register	00030_H	32	PFSFRSE, SV, 32	Application Reset	369
CPU_CFI0_ACCENPFSFRCFG_PRS	CPU0 shadow NVM SFR PRS access enable register	00034_H	32	PFSFRSE, SV, 32	Application Reset	369
CPU_CFI0_PROTPFSE	CPU0 PROTPF register safe endinit	00048_H	32	SV, 32, PROT	Application Reset	623
CPU_CFI0_ACCENPFCFG_RDA	CPU0 shadow NVM read access enable register A	00068_H	32	PFSE, SV, 32	Application Reset	626
CPU_CFI0_ACCENPFCFG_RDB	CPU0 shadow NVM read access enable register B	$0006C_H$	32	PFSE, SV, 32	Application Reset	627
CPU_CFI0_ACCENPFCFG_VM	CPU0 shadow NVM VM access enable register	00070_H	32	PFSE, SV, 32	Application Reset	373
CPU_CFI0_ACCENPFCFG_PRS	CPU0 shadow NVM PRS access enable register	00074_H	32	PFSE, SV, 32	Application Reset	374
CPU_CFI0_ACCENPFCFG_RGNLA	CPU0 shadow NVM region lower address register	00078_H	32	PFSE, SV, 32	Application Reset	374
CPU_CFI0_ACCENPFCFG_RGNUA	CPU0 shadow NVM region upper address register	$0007C_H$	32	PFSE, SV, 32	Application Reset	375
CPU_CFI0_ACCENPFSFR_WRA	CPU0 NVM SFR write access enable register A	00080_H	32	PFSFRSE, SV, 32	Application Reset	628
CPU_CFI0_ACCENPFSFR_WRB	CPU0 NVM SFR write access enable register B	00084_H	32	PFSFRSE, SV, 32	Application Reset	629
CPU_CFI0_ACCENPFSFR_RDA	CPU0 NVM SFR read access enable register A	00088_H	32	PFSFRSE, SV, 32	Application Reset	630

(table continues...)

Table 332 (continued) Register overview - CPU_CFI0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU_CFI0_ACCENPFSFR_RDB	CPU0 NVM SFR read access enable register B	$0008C_H$	32	PFSFRSE, SV, 32	Application Reset	631
CPU_CFI0_ACCENPFSFR_VM	CPU0 NVM SFR VM access enable register	00090_H	32	PFSFRSE, SV, 32	Application Reset	378
CPU_CFI0_ACCENPFSFR_PRS	CPU0 NVM SFR PRS access enable register	00094_H	32	PFSFRSE, SV, 32	Application Reset	379
CPU_CFI0_ACCENPFy_RDA (y=0-7)	CPU0 NVM range y read access enable register A	00108_H+ y*20_H	32	PFSE, SV, 32	Application Reset	632
CPU_CFI0_ACCENPFy_RDB (y=0-7)	CPU0 NVM range y read access enable register B	$0010C_H$+ y*20_H	32	PFSE, SV, 32	Application Reset	633
CPU_CFI0_ACCENPFy_VM (y=0-7)	CPU0 NVM range y VM access enable register	00110_H+ y*20_H	32	PFSE, SV, 32	Application Reset	381
CPU_CFI0_ACCENPFy_PRS (y=0-7)	CPU0 NVM range y PRS access enable register	00114_H+ y*20_H	32	PFSE, SV, 32	Application Reset	381
CPU_CFI0_ACCENPFy_RGNLA (y=0-7)	CPU0 NVM range y region lower address register	00118_H+ y*20_H	32	PFSE, SV, 32	Application Reset	382
CPU_CFI0_ACCENPFy_RGNUA (y=0-7)	CPU0 NVM range y region upper address register	$0011C_H$+ y*20_H	32	PFSE, SV, 32	Application Reset	382
CPU_CFI0_PFIy_FLASHCON0 (y=0-1)	CPU0 NVM interface y configuration register 0	00200_H+ y*8	P4, 32	PFSFRE, P4, SV, 32	See 634	634
CPU_CFI0_PFIy_FLASHCON1 (y=0-1)	CPU0 NVM interface y configuration register 1	00204_H+ y*8	P4, 32	PFSFRE, P4, SV, 32	Application Reset	384
CPU_CFI0_TS16PNVMy_CTRL0 (y=0-1)	CPU0 NVM interface y MCDS performance counter control register 0	00210_H+ y*8	P4, 32	P4, SV, 32	Debug Reset	384
CPU_CFI0_TS16PNVMy_CTRL1 (y=0-1)	CPU0 NVM interface y MCDS performance counter control register 1	00214_H+ y*8	P4, 32	P4, SV, 32	Debug Reset	385
CPU_CFI0_CFILCLCON	CPU0 NVM interface lockstep control register	00240_H	P4, 32	P4, SV, 32	Cold PowerOn Reset	636
CPU_CFI0_CFILCLTEST	CPU0 NVM interface lockstep test register	00244_H	P4, 32	P4, SV, 32	Application Reset	387
CPU_CFI0_CFIERR	CPU0 CFI error register	00250_H	P4, 32	P4, SV, 32	Application Reset	388

(table continues...)

Table 332 (continued) Register overview - CPU_CFI0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU_CFI0_CFIERRAR	CPU0 CFI error address register	00254_H	P4, 32	P4, SV, 32	Application Reset	388
CPU_CFI0_PFIBUFDIS	CPU0 PFI buffer disable	00320_H	P4, 32	P4, SV, 32	Application Reset	389

4.10.4.2.4 Register overview - CPU_CFI1 (ascending offset address)

Table 333 Register overview - CPU_CFI1 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU_CFI1_PROTPFSFRE	CPU1 PROTPFSFR register endinit	00000_H	32	SV, 32, PROT	Application Reset	614
CPU_CFI1_PROTPFSFRSE	CPU1 PROTPFSFR register safe endinit	00008_H	32	SV, 32, PROT	Application Reset	617
CPU_CFI1_ACCENPFSFRCFG_WRA	CPU1 shadow NVM SFR write access enable register A	00020_H	32	PFSFRSE, SV, 32	Application Reset	619
CPU_CFI1_ACCENPFSFRCFG_WRB	CPU1 shadow NVM SFR write access enable register B	00024_H	32	PFSFRSE, SV, 32	Application Reset	620
CPU_CFI1_ACCENPFSFRCFG_RDA	CPU1 shadow NVM SFR read access enable register A	00028_H	32	PFSFRSE, SV, 32	Application Reset	621
CPU_CFI1_ACCENPFSFRCFG_RDB	CPU1 shadow NVM SFR read access enable register B	$0002C_H$	32	PFSFRSE, SV, 32	Application Reset	622
CPU_CFI1_ACCENPFSFRCFG_VM	CPU1 shadow NVM SFR VM access enable register	00030_H	32	PFSFRSE, SV, 32	Application Reset	369
CPU_CFI1_ACCENPFSFRCFG_PRS	CPU1 shadow NVM SFR PRS access enable register	00034_H	32	PFSFRSE, SV, 32	Application Reset	369
CPU_CFI1_PROTPFSE	CPU1 PROTPF register safe endinit	00048_H	32	SV, 32, PROT	Application Reset	623
CPU_CFI1_ACCENPFCFG_RDA	CPU1 shadow NVM read access enable register A	00068_H	32	PFSE, SV, 32	Application Reset	626
CPU_CFI1_ACCENPFCFG_RDB	CPU1 shadow NVM read access enable register B	$0006C_H$	32	PFSE, SV, 32	Application Reset	627
CPU_CFI1_ACCENPFCFG_VM	CPU1 shadow NVM VM access enable register	00070_H	32	PFSE, SV, 32	Application Reset	373
CPU_CFI1_ACCENPFCFG_PRS	CPU1 shadow NVM PRS access enable register	00074_H	32	PFSE, SV, 32	Application Reset	374
CPU_CFI1_ACCENPFCFG_RGNLA	CPU1 shadow NVM region lower address register	00078_H	32	PFSE, SV, 32	Application Reset	374

(table continues...)

Table 333 (continued) Register overview - CPU_CFI1 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU_CFI1_ACCENPFCFG_RGNUA	CPU1 shadow NVM region upper address register	$0007C_H$	32	PFSE, SV, 32	Application Reset	375
CPU_CFI1_ACCENPFSFR_WRA	CPU1 NVM SFR write access enable register A	00080_H	32	PFSFRSE, SV, 32	Application Reset	628
CPU_CFI1_ACCENPFSFR_WRB	CPU1 NVM SFR write access enable register B	00084_H	32	PFSFRSE, SV, 32	Application Reset	629
CPU_CFI1_ACCENPFSFR_RDA	CPU1 NVM SFR read access enable register A	00088_H	32	PFSFRSE, SV, 32	Application Reset	630
CPU_CFI1_ACCENPFSFR_RDB	CPU1 NVM SFR read access enable register B	$0008C_H$	32	PFSFRSE, SV, 32	Application Reset	631
CPU_CFI1_ACCENPFSFR_VM	CPU1 NVM SFR VM access enable register	00090_H	32	PFSFRSE, SV, 32	Application Reset	378
CPU_CFI1_ACCENPFSFR_PRS	CPU1 NVM SFR PRS access enable register	00094_H	32	PFSFRSE, SV, 32	Application Reset	379
CPU_CFI1_ACCENPFy_RDA (y=0-7)	CPU1 NVM range y read access enable register A	00108_H+ y*20_H	32	PFSE, SV, 32	Application Reset	632
CPU_CFI1_ACCENPFy_RDB (y=0-7)	CPU1 NVM range y read access enable register B	$0010C_H$+ y*20_H	32	PFSE, SV, 32	Application Reset	633
CPU_CFI1_ACCENPFy_VM (y=0-7)	CPU1 NVM range y VM access enable register	00110_H+ y*20_H	32	PFSE, SV, 32	Application Reset	381
CPU_CFI1_ACCENPFy_PRS (y=0-7)	CPU1 NVM range y PRS access enable register	00114_H+ y*20_H	32	PFSE, SV, 32	Application Reset	381
CPU_CFI1_ACCENPFy_RGNLA (y=0-7)	CPU1 NVM range y region lower address register	00118_H+ y*20_H	32	PFSE, SV, 32	Application Reset	382
CPU_CFI1_ACCENPFy_RGNUA (y=0-7)	CPU1 NVM range y region upper address register	$0011C_H$+ y*20_H	32	PFSE, SV, 32	Application Reset	382
CPU_CFI1_PFIy_FLASHCON0 (y=0-1)	CPU1 NVM interface y configuration register 0	00200_H+ y*8	P4, 32	PFSFRE, P4, SV, 32	See 634	634
CPU_CFI1_PFIy_FLASHCON1 (y=0-1)	CPU1 NVM interface y configuration register 1	00204_H+ y*8	P4, 32	PFSFRE, P4, SV, 32	Application Reset	384
CPU_CFI1_TS16PNVMy_CTRL0 (y=0-1)	CPU1 NVM interface y MCDS performance counter control register 0	00210_H+ y*8	P4, 32	P4, SV, 32	Debug Reset	384

(table continues...)

Table 333 (continued) Register overview - CPU_CFI1 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU_CFI1_TS16PNVMy_CTRL1 (y=0-1)	CPU1 NVM interface y MCDS performance counter control register 1	00214_H+ y*8	P4, 32	P4, SV, 32	Debug Reset	385
CPU_CFI1_CFILCLCON	CPU1 NVM interface lockstep control register	00240_H	P4, 32	P4, SV, 32	Cold PowerOn Reset	636
CPU_CFI1_CFILCLTEST	CPU1 NVM interface lockstep test register	00244_H	P4, 32	P4, SV, 32	Application Reset	387
CPU_CFI1_CFIERR	CPU1 CFI error register	00250_H	P4, 32	P4, SV, 32	Application Reset	388
CPU_CFI1_CFIERRAR	CPU1 CFI error address register	00254_H	P4, 32	P4, SV, 32	Application Reset	388
CPU_CFI1_PFIBUFDIS	CPU1 PFI buffer disable	00320_H	P4, 32	P4, SV, 32	Application Reset	389

4.10.4.2.5 Register overview - CPU_CFI2 (ascending offset address)

Table 334 Register overview - CPU_CFI2 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU_CFI2_PROTPFSFRE	CPU2 PROTPFSFR register endinit	00000_H	32	SV, 32, PROT	Application Reset	614
CPU_CFI2_PROTPFSFRSE	CPU2 PROTPFSFR register safe endinit	00008_H	32	SV, 32, PROT	Application Reset	617
CPU_CFI2_ACCENPFSFRCFG_WRA	CPU2 shadow NVM SFR write access enable register A	00020_H	32	PFSFRSE, SV, 32	Application Reset	619
CPU_CFI2_ACCENPFSFRCFG_WRB	CPU2 shadow NVM SFR write access enable register B	00024_H	32	PFSFRSE, SV, 32	Application Reset	620
CPU_CFI2_ACCENPFSFRCFG_RDA	CPU2 shadow NVM SFR read access enable register A	00028_H	32	PFSFRSE, SV, 32	Application Reset	621
CPU_CFI2_ACCENPFSFRCFG_RDB	CPU2 shadow NVM SFR read access enable register B	$0002C_H$	32	PFSFRSE, SV, 32	Application Reset	622
CPU_CFI2_ACCENPFSFRCFG_VM	CPU2 shadow NVM SFR VM access enable register	00030_H	32	PFSFRSE, SV, 32	Application Reset	369
CPU_CFI2_ACCENPFSFRCFG_PRS	CPU2 shadow NVM SFR PRS access enable register	00034_H	32	PFSFRSE, SV, 32	Application Reset	369
CPU_CFI2_PROTPFSE	CPU2 PROTPF register safe endinit	00048_H	32	SV, 32, PROT	Application Reset	623

(table continues...)

Table 334 (continued) Register overview - CPU_CFI2 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU_CFI2_ACCENPFCFG_RDA	CPU2 shadow NVM read access enable register A	00068_H	32	PFSE, SV, 32	Application Reset	626
CPU_CFI2_ACCENPFCFG_RDB	CPU2 shadow NVM read access enable register B	$0006C_H$	32	PFSE, SV, 32	Application Reset	627
CPU_CFI2_ACCENPFCFG_VM	CPU2 shadow NVM VM access enable register	00070_H	32	PFSE, SV, 32	Application Reset	373
CPU_CFI2_ACCENPFCFG_PRS	CPU2 shadow NVM PRS access enable register	00074_H	32	PFSE, SV, 32	Application Reset	374
CPU_CFI2_ACCENPFCFG_RGNLA	CPU2 shadow NVM region lower address register	00078_H	32	PFSE, SV, 32	Application Reset	374
CPU_CFI2_ACCENPFCFG_RGNUA	CPU2 shadow NVM region upper address register	$0007C_H$	32	PFSE, SV, 32	Application Reset	375
CPU_CFI2_ACCENPFSFR_WRA	CPU2 NVM SFR write access enable register A	00080_H	32	PFSFRSE, SV, 32	Application Reset	628
CPU_CFI2_ACCENPFSFR_WRB	CPU2 NVM SFR write access enable register B	00084_H	32	PFSFRSE, SV, 32	Application Reset	629
CPU_CFI2_ACCENPFSFR_RDA	CPU2 NVM SFR read access enable register A	00088_H	32	PFSFRSE, SV, 32	Application Reset	630
CPU_CFI2_ACCENPFSFR_RDB	CPU2 NVM SFR read access enable register B	$0008C_H$	32	PFSFRSE, SV, 32	Application Reset	631
CPU_CFI2_ACCENPFSFR_VM	CPU2 NVM SFR VM access enable register	00090_H	32	PFSFRSE, SV, 32	Application Reset	378
CPU_CFI2_ACCENPFSFR_PRS	CPU2 NVM SFR PRS access enable register	00094_H	32	PFSFRSE, SV, 32	Application Reset	379
CPU_CFI2_ACCENPFy_RDA (y=0-7)	CPU2 NVM range y read access enable register A	00108_H+ y*20_H	32	PFSE, SV, 32	Application Reset	632
CPU_CFI2_ACCENPFy_RDB (y=0-7)	CPU2 NVM range y read access enable register B	$0010C_H$+ y*20_H	32	PFSE, SV, 32	Application Reset	633
CPU_CFI2_ACCENPFy_VM (y=0-7)	CPU2 NVM range y VM access enable register	00110_H+ y*20_H	32	PFSE, SV, 32	Application Reset	381
CPU_CFI2_ACCENPFy_PRS (y=0-7)	CPU2 NVM range y PRS access enable register	00114_H+ y*20_H	32	PFSE, SV, 32	Application Reset	381
CPU_CFI2_ACCENPFy_RGNLA (y=0-7)	CPU2 NVM range y region lower address register	00118_H+ y*20_H	32	PFSE, SV, 32	Application Reset	382
CPU_CFI2_ACCENPFy_RGNUA (y=0-7)	CPU2 NVM range y region upper address register	$0011C_H$+ y*20_H	32	PFSE, SV, 32	Application Reset	382

(table continues...)

Table 334 (continued) Register overview - CPU_CFI2 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU_CFI2_PFIy_FLASHCON0 (y=0-1)	CPU2 NVM interface y configuration register 0	00200_H+ y*8	P4, 32	PFSFRE, P4, SV, 32	See 634	634
CPU_CFI2_PFIy_FLASHCON1 (y=0-1)	CPU2 NVM interface y configuration register 1	00204_H+ y*8	P4, 32	PFSFRE, P4, SV, 32	Application Reset	384
CPU_CFI2_TS16PNVMy_CTRL0 (y=0-1)	CPU2 NVM interface y MCDS performance counter control register 0	00210_H+ y*8	P4, 32	P4, SV, 32	Debug Reset	384
CPU_CFI2_TS16PNVMy_CTRL1 (y=0-1)	CPU2 NVM interface y MCDS performance counter control register 1	00214_H+ y*8	P4, 32	P4, SV, 32	Debug Reset	385
CPU_CFI2_CFILCLCON	CPU2 NVM interface lockstep control register	00240_H	P4, 32	P4, SV, 32	Cold PowerOn Reset	636
CPU_CFI2_CFILCLTEST	CPU2 NVM interface lockstep test register	00244_H	P4, 32	P4, SV, 32	Application Reset	387
CPU_CFI2_CFIERR	CPU2 CFI error register	00250_H	P4, 32	P4, SV, 32	Application Reset	388
CPU_CFI2_CFIERRAR	CPU2 CFI error address register	00254_H	P4, 32	P4, SV, 32	Application Reset	388
CPU_CFI2_PFIBUFDIS	CPU2 PFI buffer disable	00320_H	P4, 32	P4, SV, 32	Application Reset	389

4.10.4.2.6 Register overview - CPU_CFI3 (ascending offset address)

Table 335 Register overview - CPU_CFI3 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU_CFI3_PROTPFSFRE	CPU3 PROTPFSFR register endinit	00000_H	32	SV, 32, PROT	Application Reset	614
CPU_CFI3_PROTPFSFRSE	CPU3 PROTPFSFR register safe endinit	00008_H	32	SV, 32, PROT	Application Reset	617
CPU_CFI3_ACCENPFSFRCFG_WRA	CPU3 shadow NVM SFR write access enable register A	00020_H	32	PFSFRSE, SV, 32	Application Reset	619
CPU_CFI3_ACCENPFSFRCFG_WRB	CPU3 shadow NVM SFR write access enable register B	00024_H	32	PFSFRSE, SV, 32	Application Reset	620
CPU_CFI3_ACCENPFSFRCFG_RDA	CPU3 shadow NVM SFR read access enable register A	00028_H	32	PFSFRSE, SV, 32	Application Reset	621

(table continues...)

Table 335 (continued) Register overview - CPU_CFI3 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU_CFI3_ACCENPFSFRCFG_RDB	CPU3 shadow NVM SFR read access enable register B	$0002C_H$	32	PFSFRSE, SV, 32	Application Reset	622
CPU_CFI3_ACCENPFSFRCFG_VM	CPU3 shadow NVM SFR VM access enable register	00030_H	32	PFSFRSE, SV, 32	Application Reset	369
CPU_CFI3_ACCENPFSFRCFG_PRS	CPU3 shadow NVM SFR PRS access enable register	00034_H	32	PFSFRSE, SV, 32	Application Reset	369
CPU_CFI3_PROTPFSE	CPU3 PROTPF register safe endinit	00048_H	32	SV, 32, PROT	Application Reset	623
CPU_CFI3_ACCENPFCFG_RDA	CPU3 shadow NVM read access enable register A	00068_H	32	PFSE, SV, 32	Application Reset	626
CPU_CFI3_ACCENPFCFG_RDB	CPU3 shadow NVM read access enable register B	$0006C_H$	32	PFSE, SV, 32	Application Reset	627
CPU_CFI3_ACCENPFCFG_VM	CPU3 shadow NVM VM access enable register	00070_H	32	PFSE, SV, 32	Application Reset	373
CPU_CFI3_ACCENPFCFG_PRS	CPU3 shadow NVM PRS access enable register	00074_H	32	PFSE, SV, 32	Application Reset	374
CPU_CFI3_ACCENPFCFG_RGNLA	CPU3 shadow NVM region lower address register	00078_H	32	PFSE, SV, 32	Application Reset	374
CPU_CFI3_ACCENPFCFG_RGNUA	CPU3 shadow NVM region upper address register	$0007C_H$	32	PFSE, SV, 32	Application Reset	375
CPU_CFI3_ACCENPFSFR_WRA	CPU3 NVM SFR write access enable register A	00080_H	32	PFSFRSE, SV, 32	Application Reset	628
CPU_CFI3_ACCENPFSFR_WRB	CPU3 NVM SFR write access enable register B	00084_H	32	PFSFRSE, SV, 32	Application Reset	629
CPU_CFI3_ACCENPFSFR_RDA	CPU3 NVM SFR read access enable register A	00088_H	32	PFSFRSE, SV, 32	Application Reset	630
CPU_CFI3_ACCENPFSFR_RDB	CPU3 NVM SFR read access enable register B	$0008C_H$	32	PFSFRSE, SV, 32	Application Reset	631
CPU_CFI3_ACCENPFSFR_VM	CPU3 NVM SFR VM access enable register	00090_H	32	PFSFRSE, SV, 32	Application Reset	378
CPU_CFI3_ACCENPFSFR_PRS	CPU3 NVM SFR PRS access enable register	00094_H	32	PFSFRSE, SV, 32	Application Reset	379
CPU_CFI3_ACCENPFy_RDA (y=0-7)	CPU3 NVM range y read access enable register A	00108_H+ y*20_H	32	PFSE, SV, 32	Application Reset	632
CPU_CFI3_ACCENPFy_RDB (y=0-7)	CPU3 NVM range y read access enable register B	$0010C_H$+ y*20_H	32	PFSE, SV, 32	Application Reset	633
CPU_CFI3_ACCENPFy_VM (y=0-7)	CPU3 NVM range y VM access enable register	00110_H+ y*20_H	32	PFSE, SV, 32	Application Reset	381

(table continues...)

Table 335 (continued) Register overview - CPU_CFI3 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU_CFI3_ACCENPFy_PRS (y=0-7)	CPU3 NVM range y PRS access enable register	00114_H+ y*20_H	32	PFSE, SV, 32	Application Reset	381
CPU_CFI3_ACCENPFy_RGNLA (y=0-7)	CPU3 NVM range y region lower address register	00118_H+ y*20_H	32	PFSE, SV, 32	Application Reset	382
CPU_CFI3_ACCENPFy_RGNUA (y=0-7)	CPU3 NVM range y region upper address register	$0011C_H$+ y*20_H	32	PFSE, SV, 32	Application Reset	382
CPU_CFI3_PFIy_FLASHCON0 (y=0-1)	CPU3 NVM interface y configuration register 0	00200_H+ y*8	P4, 32	PFSFRE, P4, SV, 32	See 634	634
CPU_CFI3_PFIy_FLASHCON1 (y=0-1)	CPU3 NVM interface y configuration register 1	00204_H+ y*8	P4, 32	PFSFRE, P4, SV, 32	Application Reset	384
CPU_CFI3_TS16PNVMy_CTRL0 (y=0-1)	CPU3 NVM interface y MCDS performance counter control register 0	00210_H+ y*8	P4, 32	P4, SV, 32	Debug Reset	384
CPU_CFI3_TS16PNVMy_CTRL1 (y=0-1)	CPU3 NVM interface y MCDS performance counter control register 1	00214_H+ y*8	P4, 32	P4, SV, 32	Debug Reset	385
CPU_CFI3_CFILCLCON	CPU3 NVM interface lockstep control register	00240_H	P4, 32	P4, SV, 32	Cold PowerOn Reset	636
CPU_CFI3_CFILCLTEST	CPU3 NVM interface lockstep test register	00244_H	P4, 32	P4, SV, 32	Application Reset	387
CPU_CFI3_CFIERR	CPU3 CFI error register	00250_H	P4, 32	P4, SV, 32	Application Reset	388
CPU_CFI3_CFIERRAR	CPU3 CFI error address register	00254_H	P4, 32	P4, SV, 32	Application Reset	388
CPU_CFI3_PFIBUFDIS	CPU3 PFI buffer disable	00320_H	P4, 32	P4, SV, 32	Application Reset	389

4.10.4.2.7 Register overview - CPU_CFI4 (ascending offset address)

Table 336 Register overview - CPU_CFI4 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU_CFI4_PROTPFSFRE	CPU4 PROTPFSFR register endinit	00000_H	32	SV, 32, PROT	Application Reset	614
CPU_CFI4_PROTPFSFRSE	CPU4 PROTPFSFR register safe endinit	00008_H	32	SV, 32, PROT	Application Reset	617

(table continues...)

Table 336 (continued) Register overview - CPU_CFI4 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU_CFI4_ACCENPFSFRCFG_WRA	CPU4 shadow NVM SFR write access enable register A	00020_H	32	PFSFRSE, SV, 32	Application Reset	619
CPU_CFI4_ACCENPFSFRCFG_WRB	CPU4 shadow NVM SFR write access enable register B	00024_H	32	PFSFRSE, SV, 32	Application Reset	620
CPU_CFI4_ACCENPFSFRCFG_RDA	CPU4 shadow NVM SFR read access enable register A	00028_H	32	PFSFRSE, SV, 32	Application Reset	621
CPU_CFI4_ACCENPFSFRCFG_RDB	CPU4 shadow NVM SFR read access enable register B	$0002C_H$	32	PFSFRSE, SV, 32	Application Reset	622
CPU_CFI4_ACCENPFSFRCFG_VM	CPU4 shadow NVM SFR VM access enable register	00030_H	32	PFSFRSE, SV, 32	Application Reset	369
CPU_CFI4_ACCENPFSFRCFG_PRS	CPU4 shadow NVM SFR PRS access enable register	00034_H	32	PFSFRSE, SV, 32	Application Reset	369
CPU_CFI4_PROTPFSE	CPU4 PROTPF register safe endinit	00048_H	32	SV, 32, PROT	Application Reset	623
CPU_CFI4_ACCENPFCFG_RDA	CPU4 shadow NVM read access enable register A	00068_H	32	PFSE, SV, 32	Application Reset	626
CPU_CFI4_ACCENPFCFG_RDB	CPU4 shadow NVM read access enable register B	$0006C_H$	32	PFSE, SV, 32	Application Reset	627
CPU_CFI4_ACCENPFCFG_VM	CPU4 shadow NVM VM access enable register	00070_H	32	PFSE, SV, 32	Application Reset	373
CPU_CFI4_ACCENPFCFG_PRS	CPU4 shadow NVM PRS access enable register	00074_H	32	PFSE, SV, 32	Application Reset	374
CPU_CFI4_ACCENPFCFG_RGNLA	CPU4 shadow NVM region lower address register	00078_H	32	PFSE, SV, 32	Application Reset	374
CPU_CFI4_ACCENPFCFG_RGNUA	CPU4 shadow NVM region upper address register	$0007C_H$	32	PFSE, SV, 32	Application Reset	375
CPU_CFI4_ACCENPFSFR_WRA	CPU4 NVM SFR write access enable register A	00080_H	32	PFSFRSE, SV, 32	Application Reset	628
CPU_CFI4_ACCENPFSFR_WRB	CPU4 NVM SFR write access enable register B	00084_H	32	PFSFRSE, SV, 32	Application Reset	629
CPU_CFI4_ACCENPFSFR_RDA	CPU4 NVM SFR read access enable register A	00088_H	32	PFSFRSE, SV, 32	Application Reset	630
CPU_CFI4_ACCENPFSFR_RDB	CPU4 NVM SFR read access enable register B	$0008C_H$	32	PFSFRSE, SV, 32	Application Reset	631
CPU_CFI4_ACCENPFSFR_VM	CPU4 NVM SFR VM access enable register	00090_H	32	PFSFRSE, SV, 32	Application Reset	378
CPU_CFI4_ACCENPFSFR_PRS	CPU4 NVM SFR PRS access enable register	00094_H	32	PFSFRSE, SV, 32	Application Reset	379

(table continues...)

Table 336 (continued) Register overview - CPU_CFI4 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU_CFI4_ACCENPFy_RDA (y=0-7)	CPU4 NVM range y read access enable register A	00108_H+ y*20_H	32	PFSE, SV, 32	Application Reset	632
CPU_CFI4_ACCENPFy_RDB (y=0-7)	CPU4 NVM range y read access enable register B	$0010C_H$+ y*20_H	32	PFSE, SV, 32	Application Reset	633
CPU_CFI4_ACCENPFy_VM (y=0-7)	CPU4 NVM range y VM access enable register	00110_H+ y*20_H	32	PFSE, SV, 32	Application Reset	381
CPU_CFI4_ACCENPFy_PRS (y=0-7)	CPU4 NVM range y PRS access enable register	00114_H+ y*20_H	32	PFSE, SV, 32	Application Reset	381
CPU_CFI4_ACCENPFy_RGNLA (y=0-7)	CPU4 NVM range y region lower address register	00118_H+ y*20_H	32	PFSE, SV, 32	Application Reset	382
CPU_CFI4_ACCENPFy_RGNUA (y=0-7)	CPU4 NVM range y region upper address register	$0011C_H$+ y*20_H	32	PFSE, SV, 32	Application Reset	382
CPU_CFI4_PFIy_FLASHCON0 (y=0-1)	CPU4 NVM interface y configuration register 0	00200_H+ y*8	P4, 32	PFSFRE, P4, SV, 32	See 634	634
CPU_CFI4_PFIy_FLASHCON1 (y=0-1)	CPU4 NVM interface y configuration register 1	00204_H+ y*8	P4, 32	PFSFRE, P4, SV, 32	Application Reset	384
CPU_CFI4_TS16PNVMy_CTRL0 (y=0-1)	CPU4 NVM interface y MCDS performance counter control register 0	00210_H+ y*8	P4, 32	P4, SV, 32	Debug Reset	384
CPU_CFI4_TS16PNVMy_CTRL1 (y=0-1)	CPU4 NVM interface y MCDS performance counter control register 1	00214_H+ y*8	P4, 32	P4, SV, 32	Debug Reset	385
CPU_CFI4_CFILCLCON	CPU4 NVM interface lockstep control register	00240_H	P4, 32	P4, SV, 32	Cold PowerOn Reset	636
CPU_CFI4_CFILCLTEST	CPU4 NVM interface lockstep test register	00244_H	P4, 32	P4, SV, 32	Application Reset	387
CPU_CFI4_CFIERR	CPU4 CFI error register	00250_H	P4, 32	P4, SV, 32	Application Reset	388
CPU_CFI4_CFIERRAR	CPU4 CFI error address register	00254_H	P4, 32	P4, SV, 32	Application Reset	388
CPU_CFI4_PFIBUFDIS	CPU4 PFI buffer disable	00320_H	P4, 32	P4, SV, 32	Application Reset	389

4.10.4.2.8 Register overview - CPU_CFI5 (ascending offset address)

Table 337 Register overview - CPU_CFI5 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU_CFI5_PROTPFSFRE	CPU5 PROTPFSFR register endinit	00000_H	32	SV, 32, PROT	Application Reset	614
CPU_CFI5_PROTPFSFRSE	CPU5 PROTPFSFR register safe endinit	00008_H	32	SV, 32, PROT	Application Reset	617
CPU_CFI5_ACCENPFSFRCFG_WRA	CPU5 shadow NVM SFR write access enable register A	00020_H	32	PFSFRSE, SV, 32	Application Reset	619
CPU_CFI5_ACCENPFSFRCFG_WRB	CPU5 shadow NVM SFR write access enable register B	00024_H	32	PFSFRSE, SV, 32	Application Reset	620
CPU_CFI5_ACCENPFSFRCFG_RDA	CPU5 shadow NVM SFR read access enable register A	00028_H	32	PFSFRSE, SV, 32	Application Reset	621
CPU_CFI5_ACCENPFSFRCFG_RDB	CPU5 shadow NVM SFR read access enable register B	$0002C_H$	32	PFSFRSE, SV, 32	Application Reset	622
CPU_CFI5_ACCENPFSFRCFG_VM	CPU5 shadow NVM SFR VM access enable register	00030_H	32	PFSFRSE, SV, 32	Application Reset	369
CPU_CFI5_ACCENPFSFRCFG_PRS	CPU5 shadow NVM SFR PRS access enable register	00034_H	32	PFSFRSE, SV, 32	Application Reset	369
CPU_CFI5_PROTPFSE	CPU5 PROTPF register safe endinit	00048_H	32	SV, 32, PROT	Application Reset	623
CPU_CFI5_ACCENPFCFG_RDA	CPU5 shadow NVM read access enable register A	00068_H	32	PFSE, SV, 32	Application Reset	626
CPU_CFI5_ACCENPFCFG_RDB	CPU5 shadow NVM read access enable register B	$0006C_H$	32	PFSE, SV, 32	Application Reset	627
CPU_CFI5_ACCENPFCFG_VM	CPU5 shadow NVM VM access enable register	00070_H	32	PFSE, SV, 32	Application Reset	373
CPU_CFI5_ACCENPFCFG_PRS	CPU5 shadow NVM PRS access enable register	00074_H	32	PFSE, SV, 32	Application Reset	374
CPU_CFI5_ACCENPFCFG_RGNLA	CPU5 shadow NVM region lower address register	00078_H	32	PFSE, SV, 32	Application Reset	374
CPU_CFI5_ACCENPFCFG_RGNUA	CPU5 shadow NVM region upper address register	$0007C_H$	32	PFSE, SV, 32	Application Reset	375
CPU_CFI5_ACCENPFSFR_WRA	CPU5 NVM SFR write access enable register A	00080_H	32	PFSFRSE, SV, 32	Application Reset	628
CPU_CFI5_ACCENPFSFR_WRB	CPU5 NVM SFR write access enable register B	00084_H	32	PFSFRSE, SV, 32	Application Reset	629
CPU_CFI5_ACCENPFSFR_RDA	CPU5 NVM SFR read access enable register A	00088_H	32	PFSFRSE, SV, 32	Application Reset	630

(table continues...)

Table 337 (continued) Register overview - CPU_CFI5 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU_CFI5_ACCENPFSFR_RDB	CPU5 NVM SFR read access enable register B	$0008C_H$	32	PFSFRSE, SV, 32	Application Reset	631
CPU_CFI5_ACCENPFSFR_VM	CPU5 NVM SFR VM access enable register	00090_H	32	PFSFRSE, SV, 32	Application Reset	378
CPU_CFI5_ACCENPFSFR_PRS	CPU5 NVM SFR PRS access enable register	00094_H	32	PFSFRSE, SV, 32	Application Reset	379
CPU_CFI5_ACCENPFy_RDA (y=0-7)	CPU5 NVM range y read access enable register A	00108_H+ y*20_H	32	PFSE, SV, 32	Application Reset	632
CPU_CFI5_ACCENPFy_RDB (y=0-7)	CPU5 NVM range y read access enable register B	$0010C_H$+ y*20_H	32	PFSE, SV, 32	Application Reset	633
CPU_CFI5_ACCENPFy_VM (y=0-7)	CPU5 NVM range y VM access enable register	00110_H+ y*20_H	32	PFSE, SV, 32	Application Reset	381
CPU_CFI5_ACCENPFy_PRS (y=0-7)	CPU5 NVM range y PRS access enable register	00114_H+ y*20_H	32	PFSE, SV, 32	Application Reset	381
CPU_CFI5_ACCENPFy_RGNLA (y=0-7)	CPU5 NVM range y region lower address register	00118_H+ y*20_H	32	PFSE, SV, 32	Application Reset	382
CPU_CFI5_ACCENPFy_RGNUA (y=0-7)	CPU5 NVM range y region upper address register	$0011C_H$+ y*20_H	32	PFSE, SV, 32	Application Reset	382
CPU_CFI5_PFIy_FLASHCON0 (y=0-1)	CPU5 NVM interface y configuration register 0	00200_H+ y*8	P4, 32	PFSFRE, P4, SV, 32	See 634	634
CPU_CFI5_PFIy_FLASHCON1 (y=0-1)	CPU5 NVM interface y configuration register 1	00204_H+ y*8	P4, 32	PFSFRE, P4, SV, 32	Application Reset	384
CPU_CFI5_TS16PNVMy_CTRL0 (y=0-1)	CPU5 NVM interface y MCDS performance counter control register 0	00210_H+ y*8	P4, 32	P4, SV, 32	Debug Reset	384
CPU_CFI5_TS16PNVMy_CTRL1 (y=0-1)	CPU5 NVM interface y MCDS performance counter control register 1	00214_H+ y*8	P4, 32	P4, SV, 32	Debug Reset	385
CPU_CFI5_CFILCLCON	CPU5 NVM interface lockstep control register	00240_H	P4, 32	P4, SV, 32	Cold PowerOn Reset	636
CPU_CFI5_CFILCLTEST	CPU5 NVM interface lockstep test register	00244_H	P4, 32	P4, SV, 32	Application Reset	387
CPU_CFI5_CFIERR	CPU5 CFI error register	00250_H	P4, 32	P4, SV, 32	Application Reset	388

(table continues...)

Table 337 (continued) Register overview - CPU_CFI5 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU_CFI5_CFIERRAR	CPU5 CFI error address register	00254_H	P4, 32	P4, SV, 32	Application Reset	388
CPU_CFI5_PFIBUFDIS	CPU5 PFI buffer disable	00320_H	P4, 32	P4, SV, 32	Application Reset	389

4.10.4.2.9 Register overview - CPU_CFICS (ascending offset address)

Table 338 Register overview - CPU_CFICS (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU_CFICS_PROTPFSFRE	CPUcs PROTPFSFR register endinit	00000_H	32	SV, 32, PROT	Application Reset	614
CPU_CFICS_PROTPFSFRSE	CPUcs PROTPFSFR register safe endinit	00008_H	32	SV, 32, PROT	Application Reset	617
CPU_CFICS_ACCENPFSFRCFG_WRA	CPUcs shadow NVM SFR write access enable register A	00020_H	32	PFSFRSE, SV, 32	See 619	619
CPU_CFICS_ACCENPFSFRCFG_WRB	CPUcs shadow NVM SFR write access enable register B	00024_H	32	PFSFRSE, SV, 32	See 620	620
CPU_CFICS_ACCENPFSFRCFG_RDA	CPUcs shadow NVM SFR read access enable register A	00028_H	32	PFSFRSE, SV, 32	See 621	621
CPU_CFICS_ACCENPFSFRCFG_RDB	CPUcs shadow NVM SFR read access enable register B	$0002C_H$	32	PFSFRSE, SV, 32	See 622	622
CPU_CFICS_ACCENPFSFRCFG_VM	CPUcs shadow NVM SFR VM access enable register	00030_H	32	PFSFRSE, SV, 32	Application Reset	369
CPU_CFICS_ACCENPFSFRCFG_PRS	CPUcs shadow NVM SFR PRS access enable register	00034_H	32	PFSFRSE, SV, 32	Application Reset	369
CPU_CFICS_PROTPFSE	CPUcs PROTPF register safe endinit	00048_H	32	SV, 32, PROT	Application Reset	623
CPU_CFICS_ACCENPFCFG_RDA	CPUcs shadow NVM read access enable register A	00068_H	32	PFSE, SV, 32	Application Reset	626
CPU_CFICS_ACCENPFCFG_RDB	CPUcs shadow NVM read access enable register B	$0006C_H$	32	PFSE, SV, 32	Application Reset	627
CPU_CFICS_ACCENPFCFG_VM	CPUcs shadow NVM VM access enable register	00070_H	32	PFSE, SV, 32	Application Reset	373
CPU_CFICS_ACCENPFCFG_PRS	CPUcs shadow NVM PRS access enable register	00074_H	32	PFSE, SV, 32	Application Reset	374
CPU_CFICS_ACCENPFCFG_RGNLA	CPUcs shadow NVM region lower address register	00078_H	32	PFSE, SV, 32	Application Reset	374

(table continues...)

Table 338 (continued) Register overview - CPU_CFICS (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU_CFICS_ACCENPFCFG_RGNUA	CPUcs shadow NVM region upper address register	$0007C_H$	32	PFSE, SV, 32	Application Reset	375
CPU_CFICS_ACCENPFSFR_WRA	CPUcs NVM SFR write access enable register A	00080_H	32	PFSFRSE, SV, 32	See 628	628
CPU_CFICS_ACCENPFSFR_WRB	CPUcs NVM SFR write access enable register B	00084_H	32	PFSFRSE, SV, 32	See 629	629
CPU_CFICS_ACCENPFSFR_RDA	CPUcs NVM SFR read access enable register A	00088_H	32	PFSFRSE, SV, 32	See 630	630
CPU_CFICS_ACCENPFSFR_RDB	CPUcs NVM SFR read access enable register B	$0008C_H$	32	PFSFRSE, SV, 32	See 631	631
CPU_CFICS_ACCENPFSFR_VM	CPUcs NVM SFR VM access enable register	00090_H	32	PFSFRSE, SV, 32	Application Reset	378
CPU_CFICS_ACCENPFSFR_PRS	CPUcs NVM SFR PRS access enable register	00094_H	32	PFSFRSE, SV, 32	Application Reset	379
CPU_CFICS_ACCENPFy_RDA (y=0-7)	CPUcs NVM range y read access enable register A	00108_H+ y*20_H	32	PFSE, SV, 32	Application Reset	632
CPU_CFICS_ACCENPFy_RDB (y=0-7)	CPUcs NVM range y read access enable register B	$0010C_H$+ y*20_H	32	PFSE, SV, 32	Application Reset	633
CPU_CFICS_ACCENPFy_VM (y=0-7)	CPUcs NVM range y VM access enable register	00110_H+ y*20_H	32	PFSE, SV, 32	Application Reset	381
CPU_CFICS_ACCENPFy_PRS (y=0-7)	CPUcs NVM range y PRS access enable register	00114_H+ y*20_H	32	PFSE, SV, 32	Application Reset	381
CPU_CFICS_ACCENPFy_RGNLA (y=0-7)	CPUcs NVM range y region lower address register	00118_H+ y*20_H	32	PFSE, SV, 32	Application Reset	382
CPU_CFICS_ACCENPFy_RGNUA (y=0-7)	CPUcs NVM range y region upper address register	$0011C_H$+ y*20_H	32	PFSE, SV, 32	Application Reset	382
CPU_CFICS_PFIy_FLASHCON0 (y=0)	CPUcs NVM interface 0 configuration register 0	00200_H	P4, 32	PFSFRE, P4, SV, 32	See 634	634
CPU_CFICS_PFIy_FLASHCON1 (y=0)	CPUcs NVM interface 0 configuration register 1	00204_H	P4, 32	PFSFRE, P4, SV, 32	Application Reset	384
CPU_CFICS_TS16PNVMy_CTRL0 (y=0)	CPUcs NVM interface 0 MCDS performance counter control register 0	00210_H	P4, 32	P4, SV, 32	Debug Reset	384

(table continues...)

Table 338 (continued) Register overview - CPU_CFICS (ascending offset address)

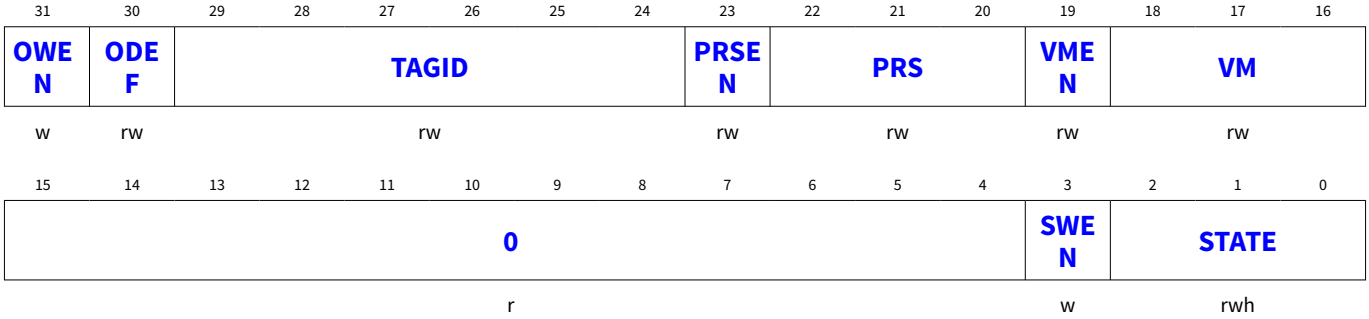
Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CPU_CFICS_TS16PNVMy_CTRL1 (y=0)	CPUcs NVM interface 0 MCDS performance counter control register 1	00214_H	P4, 32	P4, SV, 32	Debug Reset	385
CPU_CFICS_CFILCLCON	CPUcs NVM interface lockstep control register	00240_H	P4, 32	P4, SV, 32	Cold PowerOn Reset	636
CPU_CFICS_CFILCLTEST	CPUcs NVM interface lockstep test register	00244_H	P4, 32	P4, SV, 32	Application Reset	387
CPU_CFICS_CFIERR	CPUcs CFI error register	00250_H	P4, 32	P4, SV, 32	Application Reset	388
CPU_CFICS_CFIERRAR	CPUcs CFI error address register	00254_H	P4, 32	P4, SV, 32	Application Reset	388
CPU_CFICS_PFIBUFDIS	CPUcs PFI buffer disable	00320_H	P4, 32	P4, SV, 32	Application Reset	389

4.10.4.2.10 CPU0 PROTPFSFR register endinit

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner and in some PROT states, the secure master to update the PROT state as described in the PROT mechanism.

The PROTPFSFRE register controls lock / unlock of the local Endinit (E) protected control registers.

CPU_CFI0_PROTPFSFRE	Offset address:	00000_H
CPU0 PROTPFSFR register endinit	Application Reset value:	$0000\ 0000_H$
CPU_CFI1_PROTPFSFRE	Offset address:	00000_H
CPU1 PROTPFSFR register endinit	Application Reset value:	$0000\ 0000_H$
CPU_CFI2_PROTPFSFRE	Offset address:	00000_H
CPU2 PROTPFSFR register endinit	Application Reset value:	$0000\ 0000_H$
CPU_CFI3_PROTPFSFRE	Offset address:	00000_H
CPU3 PROTPFSFR register endinit	Application Reset value:	$0000\ 0000_H$
CPU_CFI4_PROTPFSFRE	Offset address:	00000_H
CPU4 PROTPFSFR register endinit	Application Reset value:	$0000\ 0000_H$
CPU_CFI5_PROTPFSFRE	Offset address:	00000_H
CPU5 PROTPFSFR register endinit	Application Reset value:	$0000\ 0000_H$
CPU_CFICS_PROTPFSFRE	Offset address:	00000_H
CPUcs PROTPFSFR register endinit	Application Reset value:	$4C00\ 0000_H$



Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
VM	18:16	rw	**Virtual machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual machine definition enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.
PRSEN	23	rw	**Protection set definition enable for PROT owner** 0_B **Disable**: PRS is part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner

(table continues...)

(continued)

Field	Bits	Type	Description
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; should be written with 0.

Table 339 Access mode restrictions sorted by descending priority

Applies to CPU_CFI0_PROTPFSFRE
Applies to CPU_CFI1_PROTPFSFRE
Applies to CPU_CFI2_PROTPFSFRE
Applies to CPU_CFI3_PROTPFSFRE
Applies to CPU_CFI4_PROTPFSFRE
Applies to CPU_CFI5_PROTPFSFRE
Applies to CPU_CFICS_PROTPFSFRE

Mode name	Access mode		Description
write 1 to .**SWEN**	rwh	STATE	
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

4.10.4.2.11 CPU0 PROTPFSFR register safe endinit

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner and in some PROT states, the secure master to update the PROT state as described in the PROT mechanism.

The PROTPFSFRSE register controls lock / unlock of the local Safe Endinit (SE) protected control registers.

CPU_CFI0_PROTPFSFRSE	Offset address:	00008_H
CPU0 PROTPFSFR register safe endinit	Application Reset value:	$0000\ 0000_H$
CPU_CFI1_PROTPFSFRSE	Offset address:	00008_H
CPU1 PROTPFSFR register safe endinit	Application Reset value:	$0000\ 0000_H$
CPU_CFI2_PROTPFSFRSE	Offset address:	00008_H
CPU2 PROTPFSFR register safe endinit	Application Reset value:	$0000\ 0000_H$
CPU_CFI3_PROTPFSFRSE	Offset address:	00008_H
CPU3 PROTPFSFR register safe endinit	Application Reset value:	$0000\ 0000_H$
CPU_CFI4_PROTPFSFRSE	Offset address:	00008_H
CPU4 PROTPFSFR register safe endinit	Application Reset value:	$0000\ 0000_H$
CPU_CFI5_PROTPFSFRSE	Offset address:	00008_H
CPU5 PROTPFSFR register safe endinit	Application Reset value:	$0000\ 0000_H$
CPU_CFICS_PROTPFSFRSE	Offset address:	00008_H
CPUcs PROTPFSFR register safe endinit	Application Reset value:	$4C00\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OWEN	ODEF	TAGID						PRSEN	PRS			VMEN	VM		
w	rw	rw						rw	rw			rw	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												SWEN	STATE		
r												w	rwh		

Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)

(table continues...)

(continued)

Field	Bits	Type	Description
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
VM	18:16	rw	**Virtual machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual machine definition enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.
PRSEN	23	rw	**Protection set definition enable for PROT owner** 0_B **Disable**: PRS is part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; should be written with 0.

Table 340 Access mode restrictions sorted by descending priority

Applies to CPU_CFI0_PROTPFSFRSE
Applies to CPU_CFI1_PROTPFSFRSE
Applies to CPU_CFI2_PROTPFSFRSE
Applies to CPU_CFI3_PROTPFSFRSE
Applies to CPU_CFI4_PROTPFSFRSE
Applies to CPU_CFI5_PROTPFSFRSE
Applies to CPU_CFICS_PROTPFSFRSE

Mode name	Access mode		Description
write 1 to .**SWEN**	rwh	STATE	

(table continues...)

Table 340 (continued) Access mode restrictions sorted by descending priority

Applies to CPU_CFI0_PROTPFSFRSE
Applies to CPU_CFI1_PROTPFSFRSE
Applies to CPU_CFI2_PROTPFSFRSE
Applies to CPU_CFI3_PROTPFSFRSE
Applies to CPU_CFI4_PROTPFSFRSE
Applies to CPU_CFI5_PROTPFSFRSE
Applies to CPU_CFICS_PROTPFSFRSE

Mode name	Access mode		Description
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

4.10.4.2.12 CPU0 shadow NVM SFR write access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

CPU_CFI0_ACCENPFSFRCFG_WRA Offset address: 00020_H
CPU0 shadow NVM SFR write access enable register A Application Reset value: $1000\ 0003_H$

CPU_CFI1_ACCENPFSFRCFG_WRA Offset address: 00020_H
CPU1 shadow NVM SFR write access enable register A Application Reset value: $1000\ 0003_H$

CPU_CFI2_ACCENPFSFRCFG_WRA Offset address: 00020_H
CPU2 shadow NVM SFR write access enable register A Application Reset value: $1000\ 0003_H$

CPU_CFI3_ACCENPFSFRCFG_WRA Offset address: 00020_H
CPU3 shadow NVM SFR write access enable register A Application Reset value: $1000\ 0003_H$

CPU_CFI4_ACCENPFSFRCFG_WRA Offset address: 00020_H
CPU4 shadow NVM SFR write access enable register A Application Reset value: $1000\ 0003_H$

CPU_CFI5_ACCENPFSFRCFG_WRA Offset address: 00020_H
CPU5 shadow NVM SFR write access enable register A Application Reset value: $1000\ 0003_H$

CPU_CFICS_ACCENPFSFRCFG_WRA Offset address: 00020_H
CPUcs shadow NVM SFR write access enable register A Reset values see: Table 341

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	**EN30**	**EN29**	**EN28**	**EN27**	**EN26**	**EN25**	**EN24**	**EN23**	**EN22**	**EN21**	**EN20**	**EN19**	**EN18**	**EN17**	**EN16**
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	**EN14**	**EN13**	**EN12**	**EN11**	**EN10**	**EN09**	**EN08**	**EN07**	**EN06**	**EN05**	**EN04**	**EN03**	**EN02**	**EN01**	**EN00**
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access

Table 341 Reset values of CPU_CFICS_ACCENPFSFRCFG_WRA

Reset type	Reset value	Note
Application Reset	$0000\ 1000_H$	Secure NVM, CSRM debug disabled
After Boot-FW Value	$1000\ 3000_H$	Secure NVM, CSRM debug enabled

4.10.4.2.13 CPU0 shadow NVM SFR write access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

CPU_CFI0_ACCENPFSFRCFG_WRB Offset address: 00024_H

CPU0 shadow NVM SFR write access enable register B Application Reset value: $0000\ 0000_H$

CPU_CFI1_ACCENPFSFRCFG_WRB Offset address: 00024_H

CPU1 shadow NVM SFR write access enable register B Application Reset value: $0000\ 0000_H$

CPU_CFI2_ACCENPFSFRCFG_WRB Offset address: 00024_H

CPU2 shadow NVM SFR write access enable register B Application Reset value: $0000\ 0000_H$

CPU_CFI3_ACCENPFSFRCFG_WRB Offset address: 00024_H

CPU3 shadow NVM SFR write access enable register B Application Reset value: $0000\ 0000_H$

CPU_CFI4_ACCENPFSFRCFG_WRB Offset address: 00024_H

CPU4 shadow NVM SFR write access enable register B Application Reset value: $0000\ 0000_H$

CPU_CFI5_ACCENPFSFRCFG_WRB Offset address: 00024_H

CPU5 shadow NVM SFR write access enable register B Application Reset value: $0000\ 0000_H$

CPU_CFICS_ACCENPFSFRCFG_WRB Offset address: 00024_H

CPUcs shadow NVM SFR write access enable register B Reset values see: Table 342

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN63	EN62	EN61	EN60	EN59	EN58	EN57	EN56	EN55	EN54	EN53	EN52	EN51	EN50	EN49	EN48
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN47	EN46	EN45	EN44	EN43	EN42	EN41	EN40	EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-63)	q-32	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access

Table 342 Reset values of CPU_CFICS_ACCENPFSFRCFG_WRB

Reset type	Reset value	Note
Application Reset	0000 0000$_H$	Secure NVM, CSRM debug disabled
After Boot-FW Value	0000 4000$_H$	Secure NVM, CSRM debug enabled

4.10.4.2.14 CPU0 shadow NVM SFR read access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

CPU_CFI0_ACCENPFSFRCFG_RDA Offset address: 00028$_H$
CPU0 shadow NVM SFR read access enable register A Application Reset value: FFFF FFFF$_H$
CPU_CFI1_ACCENPFSFRCFG_RDA Offset address: 00028$_H$
CPU1 shadow NVM SFR read access enable register A Application Reset value: FFFF FFFF$_H$
CPU_CFI2_ACCENPFSFRCFG_RDA Offset address: 00028$_H$
CPU2 shadow NVM SFR read access enable register A Application Reset value: FFFF FFFF$_H$
CPU_CFI3_ACCENPFSFRCFG_RDA Offset address: 00028$_H$
CPU3 shadow NVM SFR read access enable register A Application Reset value: FFFF FFFF$_H$
CPU_CFI4_ACCENPFSFRCFG_RDA Offset address: 00028$_H$
CPU4 shadow NVM SFR read access enable register A Application Reset value: FFFF FFFF$_H$
CPU_CFI5_ACCENPFSFRCFG_RDA Offset address: 00028$_H$
CPU5 shadow NVM SFR read access enable register A Application Reset value: FFFF FFFF$_H$
CPU_CFICS_ACCENPFSFRCFG_RDA Offset address: 00028$_H$
CPUcs shadow NVM SFR read access enable register A Reset values see: Table 343

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access

Table 343 Reset values of CPU_CFICS_ACCENPFSFRCFG_RDA

Reset type	Reset value	Note
Application Reset	0000 1000_H	Secure NVM, CSRM debug enabled
After Boot-FW Value	1000 3000_H	Secure NVM, CSRM debug enabled

4.10.4.2.15 CPU0 shadow NVM SFR read access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

CPU_CFI0_ACCENPFSFRCFG_RDB Offset address: $0002C_H$

CPU0 shadow NVM SFR read access enable register B Application Reset value: FFFF $FFFF_H$

CPU_CFI1_ACCENPFSFRCFG_RDB Offset address: $0002C_H$

CPU1 shadow NVM SFR read access enable register B Application Reset value: FFFF $FFFF_H$

CPU_CFI2_ACCENPFSFRCFG_RDB Offset address: $0002C_H$

CPU2 shadow NVM SFR read access enable register B Application Reset value: FFFF $FFFF_H$

CPU_CFI3_ACCENPFSFRCFG_RDB Offset address: $0002C_H$

CPU3 shadow NVM SFR read access enable register B Application Reset value: FFFF $FFFF_H$

CPU_CFI4_ACCENPFSFRCFG_RDB Offset address: $0002C_H$

CPU4 shadow NVM SFR read access enable register B Application Reset value: FFFF $FFFF_H$

CPU_CFI5_ACCENPFSFRCFG_RDB Offset address: $0002C_H$

CPU5 shadow NVM SFR read access enable register B Application Reset value: FFFF $FFFF_H$

CPU_CFICS_ACCENPFSFRCFG_RDB Offset address: $0002C_H$

CPUcs shadow NVM SFR read access enable register B Reset values see: Table 344

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN63	EN62	EN61	EN60	EN59	EN58	EN57	EN56	EN55	EN54	EN53	EN52	EN51	EN50	EN49	EN48
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN47	EN46	EN45	EN44	EN43	EN42	EN41	EN40	EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-63)	q-32	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access

Table 344 Reset values of CPU_CFICS_ACCENPFSFRCFG_RDB

Reset type	Reset value	Note
Application Reset	0000 0000$_H$	Secure NVM, CSRM debug disabled
After Boot-FW Value	0000 4000$_H$	Secure NVM, CSRM debug enabled

4.10.4.2.16 CPU0 PROTPF register safe endinit

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner and in some PROT states, the secure master to update the PROT state as described in the PROT mechanism.

The PROTPFSE register controls lock / unlock of the local Safe Endinit (SE) protected control registers allocated to the local NVM resources.

CPU_CFI0_PROTPFSE Offset address: 00048$_H$

CPU0 PROTPF register safe endinit Application Reset value: 0000 0000$_H$

CPU_CFI1_PROTPFSE Offset address: 00048$_H$

CPU1 PROTPF register safe endinit Application Reset value: 0000 0000$_H$

CPU_CFI2_PROTPFSE Offset address: 00048$_H$

CPU2 PROTPF register safe endinit Application Reset value: 0000 0000$_H$

CPU_CFI3_PROTPFSE Offset address: 00048$_H$

CPU3 PROTPF register safe endinit Application Reset value: 0000 0000$_H$

CPU_CFI4_PROTPFSE Offset address: 00048$_H$

CPU4 PROTPF register safe endinit Application Reset value: 0000 0000$_H$

CPU_CFI5_PROTPFSE Offset address: 00048$_H$

CPU5 PROTPF register safe endinit Application Reset value: 0000 0000$_H$

CPU_CFICS_PROTPFSE Offset address: 00048$_H$

CPUcs PROTPF register safe endinit Application Reset value: 4C00 0000$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OWEN	ODEF	TAGID						PRSEN	PRS			VMEN	VM		
w	rw	rw						rw	rw			rw	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0					SEL			0				SWEN	STATE		
r					rw			r				w	rwh		

Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
SEL	10:8	rw	**Range selection** Returns the number of the range being configured when read. Can be written by the PROT owner or CS master to select the range number to be configured if the state transition is valid in the PROT state diagram. 000_B **Const0**: Range 0 selected for configuration … 111_B **Const7**: Range 7 selected for configuration
VM	18:16	rw	**Virtual machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual machine definition enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.
PRSEN	23	rw	**Protection set definition enable for PROT owner** 0_B **Disable**: PRS is part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner

(table continues...)

(continued)

Field	Bits	Type	Description
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	7:4, 15:11	r	**Reserved** Read as 0; should be written with 0.

Table 345 Access mode restrictions sorted by descending priority

Applies to CPU_CFI0_PROTPFSE
Applies to CPU_CFI1_PROTPFSE
Applies to CPU_CFI2_PROTPFSE
Applies to CPU_CFI3_PROTPFSE
Applies to CPU_CFI4_PROTPFSE
Applies to CPU_CFI5_PROTPFSE
Applies to CPU_CFICS_PROTPFSE

Mode name	Access mode		Description
write 1 to .**SWEN**	rwh	STATE	
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

4.10.4.2.17 CPU0 shadow NVM read access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

CPU_CFI0_ACCENPFCFG_RDA	Offset address:	00068_H
CPU0 shadow NVM read access enable register A	Application Reset value:	$FFFF\ FFFF_H$
CPU_CFI1_ACCENPFCFG_RDA	Offset address:	00068_H
CPU1 shadow NVM read access enable register A	Application Reset value:	$FFFF\ FFFF_H$
CPU_CFI2_ACCENPFCFG_RDA	Offset address:	00068_H
CPU2 shadow NVM read access enable register A	Application Reset value:	$FFFF\ FFFF_H$
CPU_CFI3_ACCENPFCFG_RDA	Offset address:	00068_H
CPU3 shadow NVM read access enable register A	Application Reset value:	$FFFF\ FFFF_H$
CPU_CFI4_ACCENPFCFG_RDA	Offset address:	00068_H
CPU4 shadow NVM read access enable register A	Application Reset value:	$FFFF\ FFFF_H$
CPU_CFI5_ACCENPFCFG_RDA	Offset address:	00068_H
CPU5 shadow NVM read access enable register A	Application Reset value:	$FFFF\ FFFF_H$
CPU_CFICS_ACCENPFCFG_RDA	Offset address:	00068_H
CPUcs shadow NVM read access enable register A	Application Reset value:	$0000\ 1000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	**EN30**	**EN29**	**EN28**	**EN27**	**EN26**	**EN25**	**EN24**	**EN23**	**EN22**	**EN21**	**EN20**	**EN19**	**EN18**	**EN17**	**EN16**
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	**EN14**	**EN13**	**EN12**	**EN11**	**EN10**	**EN09**	**EN08**	**EN07**	**EN06**	**EN05**	**EN04**	**EN03**	**EN02**	**EN01**	**EN00**
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access

4.10.4.2.18 CPU0 shadow NVM read access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

CPU_CFI0_ACCENPFCFG_RDB Offset address: $0006C_H$

CPU0 shadow NVM read access enable register B Application Reset value: $FFFF\ FFFF_H$

CPU_CFI1_ACCENPFCFG_RDB Offset address: $0006C_H$

CPU1 shadow NVM read access enable register B Application Reset value: $FFFF\ FFFF_H$

CPU_CFI2_ACCENPFCFG_RDB Offset address: $0006C_H$

CPU2 shadow NVM read access enable register B Application Reset value: $FFFF\ FFFF_H$

CPU_CFI3_ACCENPFCFG_RDB Offset address: $0006C_H$

CPU3 shadow NVM read access enable register B Application Reset value: $FFFF\ FFFF_H$

CPU_CFI4_ACCENPFCFG_RDB Offset address: $0006C_H$

CPU4 shadow NVM read access enable register B Application Reset value: $FFFF\ FFFF_H$

CPU_CFI5_ACCENPFCFG_RDB Offset address: $0006C_H$

CPU5 shadow NVM read access enable register B Application Reset value: $FFFF\ FFFF_H$

CPU_CFICS_ACCENPFCFG_RDB Offset address: $0006C_H$

CPUcs shadow NVM read access enable register B Application Reset value: $0000\ 4000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN63	EN62	EN61	EN60	EN59	EN58	EN57	EN56	EN55	EN54	EN53	EN52	EN51	EN50	EN49	EN48
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN47	EN46	EN45	EN44	EN43	EN42	EN41	EN40	EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-63)	q-32	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access

4.10.4.2.19 CPU0 NVM SFR write access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

CPU_CFI0_ACCENPFSFR_WRA	Offset address:	00080_H
CPU0 NVM SFR write access enable register A	Application Reset value:	$1000\ 0003_H$
CPU_CFI1_ACCENPFSFR_WRA	Offset address:	00080_H
CPU1 NVM SFR write access enable register A	Application Reset value:	$1000\ 0003_H$
CPU_CFI2_ACCENPFSFR_WRA	Offset address:	00080_H
CPU2 NVM SFR write access enable register A	Application Reset value:	$1000\ 0003_H$
CPU_CFI3_ACCENPFSFR_WRA	Offset address:	00080_H
CPU3 NVM SFR write access enable register A	Application Reset value:	$1000\ 0003_H$
CPU_CFI4_ACCENPFSFR_WRA	Offset address:	00080_H
CPU4 NVM SFR write access enable register A	Application Reset value:	$1000\ 0003_H$
CPU_CFI5_ACCENPFSFR_WRA	Offset address:	00080_H
CPU5 NVM SFR write access enable register A	Application Reset value:	$1000\ 0003_H$
CPU_CFICS_ACCENPFSFR_WRA	Offset address:	00080_H
CPUcs NVM SFR write access enable register A	Reset values see:	Table 346

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	**EN30**	**EN29**	**EN28**	**EN27**	**EN26**	**EN25**	**EN24**	**EN23**	**EN22**	**EN21**	**EN20**	**EN19**	**EN18**	**EN17**	**EN16**
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	**EN14**	**EN13**	**EN12**	**EN11**	**EN10**	**EN09**	**EN08**	**EN07**	**EN06**	**EN05**	**EN04**	**EN03**	**EN02**	**EN01**	**EN00**
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
ENq (q=00-31)	q	rh	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access

Table 346 Reset values of CPU_CFICS_ACCENPFSFR_WRA

Reset type	Reset value	Note
Application Reset	$0000\ 1000_H$	Secure NVM, CSRM debug disabled
After Boot-FW Value	$1000\ 3000_H$	Secure NVM, CSRM debug enabled

4.10.4.2.20 CPU0 NVM SFR write access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

CPU_CFI0_ACCENPFSFR_WRB Offset address: 00084_H

CPU0 NVM SFR write access enable register B Application Reset value: $0000\ 0000_H$

CPU_CFI1_ACCENPFSFR_WRB Offset address: 00084_H

CPU1 NVM SFR write access enable register B Application Reset value: $0000\ 0000_H$

CPU_CFI2_ACCENPFSFR_WRB Offset address: 00084_H

CPU2 NVM SFR write access enable register B Application Reset value: $0000\ 0000_H$

CPU_CFI3_ACCENPFSFR_WRB Offset address: 00084_H

CPU3 NVM SFR write access enable register B Application Reset value: $0000\ 0000_H$

CPU_CFI4_ACCENPFSFR_WRB Offset address: 00084_H

CPU4 NVM SFR write access enable register B Application Reset value: $0000\ 0000_H$

CPU_CFI5_ACCENPFSFR_WRB Offset address: 00084_H

CPU5 NVM SFR write access enable register B Application Reset value: $0000\ 0000_H$

CPU_CFICS_ACCENPFSFR_WRB Offset address: 00084_H

CPUcs NVM SFR write access enable register B Reset values see: Table 347

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN63	**EN62**	**EN61**	**EN60**	**EN59**	**EN58**	**EN57**	**EN56**	**EN55**	**EN54**	**EN53**	**EN52**	**EN51**	**EN50**	**EN49**	**EN48**
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN47	**EN46**	**EN45**	**EN44**	**EN43**	**EN42**	**EN41**	**EN40**	**EN39**	**EN38**	**EN37**	**EN36**	**EN35**	**EN34**	**EN33**	**EN32**
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
ENq (q=32-63)	q-32	rh	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access

Table 347 Reset values of CPU_CFICS_ACCENPFSFR_WRB

Reset type	Reset value	Note
Application Reset	$0000\ 0000_H$	Secure NVM, CSRM debug disabled
After Boot-FW Value	$0000\ 4000_H$	Secure NVM, CSRM debug enabled

4.10.4.2.21 CPU0 NVM SFR read access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

CPU_CFI0_ACCENPFSFR_RDA Offset address: 00088_H

CPU0 NVM SFR read access enable register A Application Reset value: $FFFF\ FFFF_H$

CPU_CFI1_ACCENPFSFR_RDA Offset address: 00088_H

CPU1 NVM SFR read access enable register A Application Reset value: $FFFF\ FFFF_H$

CPU_CFI2_ACCENPFSFR_RDA Offset address: 00088_H

CPU2 NVM SFR read access enable register A Application Reset value: $FFFF\ FFFF_H$

CPU_CFI3_ACCENPFSFR_RDA Offset address: 00088_H

CPU3 NVM SFR read access enable register A Application Reset value: $FFFF\ FFFF_H$

CPU_CFI4_ACCENPFSFR_RDA Offset address: 00088_H

CPU4 NVM SFR read access enable register A Application Reset value: $FFFF\ FFFF_H$

CPU_CFI5_ACCENPFSFR_RDA Offset address: 00088_H

CPU5 NVM SFR read access enable register A Application Reset value: $FFFF\ FFFF_H$

CPU_CFICS_ACCENPFSFR_RDA Offset address: 00088_H

CPUcs NVM SFR read access enable register A Reset values see: Table 348

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	**EN30**	**EN29**	**EN28**	**EN27**	**EN26**	**EN25**	**EN24**	**EN23**	**EN22**	**EN21**	**EN20**	**EN19**	**EN18**	**EN17**	**EN16**
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	**EN14**	**EN13**	**EN12**	**EN11**	**EN10**	**EN09**	**EN08**	**EN07**	**EN06**	**EN05**	**EN04**	**EN03**	**EN02**	**EN01**	**EN00**
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
ENq (q=00-31)	q	rh	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access

Table 348 Reset values of CPU_CFICS_ACCENPFSFR_RDA

Reset type	Reset value	Note
Application Reset	$0000\ 1000_H$	Secure NVM, CSRM debug disabled
After Boot-FW Value	$1000\ 3000_H$	Secure NVM, CSRM debug enabled

4.10.4.2.22 CPU0 NVM SFR read access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

CPU_CFI0_ACCENPFSFR_RDB Offset address: $0008C_H$
CPU0 NVM SFR read access enable register B Application Reset value: $FFFF\ FFFF_H$

CPU_CFI1_ACCENPFSFR_RDB Offset address: $0008C_H$
CPU1 NVM SFR read access enable register B Application Reset value: $FFFF\ FFFF_H$

CPU_CFI2_ACCENPFSFR_RDB Offset address: $0008C_H$
CPU2 NVM SFR read access enable register B Application Reset value: $FFFF\ FFFF_H$

CPU_CFI3_ACCENPFSFR_RDB Offset address: $0008C_H$
CPU3 NVM SFR read access enable register B Application Reset value: $FFFF\ FFFF_H$

CPU_CFI4_ACCENPFSFR_RDB Offset address: $0008C_H$
CPU4 NVM SFR read access enable register B Application Reset value: $FFFF\ FFFF_H$

CPU_CFI5_ACCENPFSFR_RDB Offset address: $0008C_H$
CPU5 NVM SFR read access enable register B Application Reset value: $FFFF\ FFFF_H$

CPU_CFICS_ACCENPFSFR_RDB Offset address: $0008C_H$
CPUcs NVM SFR read access enable register B Reset values see: Table 349

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN63	**EN62**	**EN61**	**EN60**	**EN59**	**EN58**	**EN57**	**EN56**	**EN55**	**EN54**	**EN53**	**EN52**	**EN51**	**EN50**	**EN49**	**EN48**
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN47	**EN46**	**EN45**	**EN44**	**EN43**	**EN42**	**EN41**	**EN40**	**EN39**	**EN38**	**EN37**	**EN36**	**EN35**	**EN34**	**EN33**	**EN32**
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
ENq (q=32-63)	q-32	rh	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access

Table 349 Reset values of CPU_CFICS_ACCENPFSFR_RDB

Reset type	Reset value	Note
Application Reset	$0000\ 0000_H$	Secure NVM, CSRM debug disabled
After Boot-FW Value	$0000\ 4000_H$	Secure NVM, CSRM debug enabled

4.10.4.2.23 CPU0 NVM range y read access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

CPU_CFI0_ACCENPFy_RDA (y=0-7) — Offset address: 00108_H+y*20_H
CPU0 NVM range y read access enable register A — Application Reset value: FFFF FFFF$_H$

CPU_CFI1_ACCENPFy_RDA (y=0-7) — Offset address: 00108_H+y*20_H
CPU1 NVM range y read access enable register A — Application Reset value: FFFF FFFF$_H$

CPU_CFI2_ACCENPFy_RDA (y=0-7) — Offset address: 00108_H+y*20_H
CPU2 NVM range y read access enable register A — Application Reset value: FFFF FFFF$_H$

CPU_CFI3_ACCENPFy_RDA (y=0-7) — Offset address: 00108_H+y*20_H
CPU3 NVM range y read access enable register A — Application Reset value: FFFF FFFF$_H$

CPU_CFI4_ACCENPFy_RDA (y=0-7) — Offset address: 00108_H+y*20_H
CPU4 NVM range y read access enable register A — Application Reset value: FFFF FFFF$_H$

CPU_CFI5_ACCENPFy_RDA (y=0-7) — Offset address: 00108_H+y*20_H
CPU5 NVM range y read access enable register A — Application Reset value: FFFF FFFF$_H$

CPU_CFICS_ACCENPFy_RDA (y=0-7) — Offset address: 00108_H+y*20_H
CPUcs NVM range y read access enable register A — Application Reset value: 0000 1000$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
ENq (q=00-31)	q	rh	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access

4.10.4.2.24 CPU0 NVM range y read access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

CPU_CFI0_ACCENPFy_RDB (y=0-7) Offset address: $0010C_H+y*20_H$

CPU0 NVM range y read access enable register B Application Reset value: $FFFF\ FFFF_H$

CPU_CFI1_ACCENPFy_RDB (y=0-7) Offset address: $0010C_H+y*20_H$

CPU1 NVM range y read access enable register B Application Reset value: $FFFF\ FFFF_H$

CPU_CFI2_ACCENPFy_RDB (y=0-7) Offset address: $0010C_H+y*20_H$

CPU2 NVM range y read access enable register B Application Reset value: $FFFF\ FFFF_H$

CPU_CFI3_ACCENPFy_RDB (y=0-7) Offset address: $0010C_H+y*20_H$

CPU3 NVM range y read access enable register B Application Reset value: $FFFF\ FFFF_H$

CPU_CFI4_ACCENPFy_RDB (y=0-7) Offset address: $0010C_H+y*20_H$

CPU4 NVM range y read access enable register B Application Reset value: $FFFF\ FFFF_H$

CPU_CFI5_ACCENPFy_RDB (y=0-7) Offset address: $0010C_H+y*20_H$

CPU5 NVM range y read access enable register B Application Reset value: $FFFF\ FFFF_H$

CPU_CFICS_ACCENPFy_RDB (y=0-7) Offset address: $0010C_H+y*20_H$

CPUcs NVM range y read access enable register B Application Reset value: $0000\ 4000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN63	EN62	EN61	EN60	EN59	EN58	EN57	EN56	EN55	EN54	EN53	EN52	EN51	EN50	EN49	EN48
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN47	EN46	EN45	EN44	EN43	EN42	EN41	EN40	EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
ENq (q=32-63)	q-32	rh	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access

4.10.4.2.25 CPU0 NVM interface y configuration register 0

Software may configure a NVM Prefetch Buffer with a master TAG ID stored in NVM Interface Configuration Register 0.

If a CPU instance does not have a local NVM bank then the FLASHCON0 register associated with that instance will have no functionality.

CPU_CFI0_PFIy_FLASHCON0 (y=0-1)	Offset address:	00200_H+y*8
CPU0 NVM interface y configuration register 0	Reset values see:	Table 350
CPU_CFI1_PFIy_FLASHCON0 (y=0-1)	Offset address:	00200_H+y*8
CPU1 NVM interface y configuration register 0	Reset values see:	Table 351
CPU_CFI2_PFIy_FLASHCON0 (y=0-1)	Offset address:	00200_H+y*8
CPU2 NVM interface y configuration register 0	Reset values see:	Table 352
CPU_CFI3_PFIy_FLASHCON0 (y=0-1)	Offset address:	00200_H+y*8
CPU3 NVM interface y configuration register 0	Reset values see:	Table 353
CPU_CFI4_PFIy_FLASHCON0 (y=0-1)	Offset address:	00200_H+y*8
CPU4 NVM interface y configuration register 0	Reset values see:	Table 354
CPU_CFI5_PFIy_FLASHCON0 (y=0-1)	Offset address:	00200_H+y*8
CPU5 NVM interface y configuration register 0	Reset values see:	Table 355
CPU_CFICS_PFI0_FLASHCON0	Offset address:	00200_H
CPUcs NVM interface 0 configuration register 0	Reset values see:	Table 356

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		**TAG4**						**0**		**TAG3**					
r		rw						r		rw					

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		**TAG2**						**0**		**TAG1**					
r		rw						r		rw					

Field	Bits	Type	Description
TAG1	5:0	rw	**NVM prefetch buffer 1 configuration** FPB is assigned to on chip master with master TAG ID equal to TAG1.
TAG2	13:8	rw	**NVM prefetch buffer 2 configuration** FPB is assigned to on chip master with master TAG ID equal to TAG2.
TAG3	21:16	rw	**NVM prefetch buffer 3 configuration** FPB is assigned to on chip master with master TAG ID equal to TAG3.
TAG4	29:24	rw	**NVM prefetch buffer 4 configuration** FPB is assigned to on chip master with master TAG ID equal to TAG4.
0	7:6, 15:14, 23:22, 31:30	r	**Reserved - RES** Always read as 0; should be written with 0.

Table 350 Reset values of CPU_CFI0_PFIy_FLASHCON0 (y=0-1)

Reset type	Reset value	Note
After Boot-FW Value	2B2A 2928$_H$	PFI0

Table 351 Reset values of CPU_CFI1_PFIy_FLASHCON0 (y=0-1)

Reset type	Reset value	Note
After Boot-FW Value	2B2A 2829$_H$	PFI0

Table 352 Reset values of CPU_CFI2_PFIy_FLASHCON0 (y=0-1)

Reset type	Reset value	Note
After Boot-FW Value	2928 2B2A$_H$	PFI0

Table 353 Reset values of CPU_CFI3_PFIy_FLASHCON0 (y=0-1)

Reset type	Reset value	Note
After Boot-FW Value	2928 2A2B$_H$	PFI0

Table 354 Reset values of CPU_CFI4_PFIy_FLASHCON0 (y=0-1)

Reset type	Reset value	Note
After Boot-FW Value	2928 2D2C$_H$	PFI0

Table 355 Reset values of CPU_CFI5_PFIy_FLASHCON0 (y=0-1)

Reset type	Reset value	Note
After Boot-FW Value	2928 2C2D$_H$	PFI0

Table 356 Reset values of CPU_CFICS_PFI0_FLASHCON0

Reset type	Reset value	Note
After Boot-FW Value	2A29 282E$_H$	PFI0

4.10.4.2.26 CPU0 NVM interface lockstep control register

Provides control for CPU NVM Lockstep Comparator Logic blocks.

CPU_CFI0_CFILCLCON	Offset address:	00240_H
CPU0 NVM interface lockstep control register	Cold PowerOn Reset value:	$0000\ 8001_H$
CPU_CFI1_CFILCLCON	Offset address:	00240_H
CPU1 NVM interface lockstep control register	Cold PowerOn Reset value:	$0000\ 8001_H$
CPU_CFI2_CFILCLCON	Offset address:	00240_H
CPU2 NVM interface lockstep control register	Cold PowerOn Reset value:	$0000\ 8001_H$
CPU_CFI3_CFILCLCON	Offset address:	00240_H
CPU3 NVM interface lockstep control register	Cold PowerOn Reset value:	$0000\ 8001_H$
CPU_CFI4_CFILCLCON	Offset address:	00240_H
CPU4 NVM interface lockstep control register	Cold PowerOn Reset value:	$0000\ 8001_H$
CPU_CFI5_CFILCLCON	Offset address:	00240_H
CPU5 NVM interface lockstep control register	Cold PowerOn Reset value:	$0000\ 8001_H$
CPU_CFICS_CFILCLCON	Offset address:	00240_H
CPUcs NVM interface lockstep control register	Cold PowerOn Reset value:	$0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LSEN	0														LS
r	r														r

Field	Bits	Type	Description
LS	0	r	**Lockstep mode status** This bit indicates whether the CPU NVM interface is currently running in lockstep monitor mode. 0_B Not in lockstep mode 1_B Running in lockstep mode
LSEN	15	r	**Lockstep enable** Enable lockstep CPU monitoring for the CPU NVM interface. This lockstep is always enabled. 0_B Lockstep is disabled 1_B Lockstep enabled (Default after Cold Power-On Reset)
0	14:1, 31:16	r	**Reserved** Read as 0; should be written with 0.

4.10.5 TC4Dx CPU connectivity

The table below lists interfaces to and from this functional block to other blocks in the device.

Table 357 List of CPU interface signals

Interface signals	I/O	Description
CLOCK_CPU_fSRI	In	fSRI
CLOCK_CPU_fSPB	In	FPI0 is always clocked at fSPB
CLOCK_CPU_fFPI1	In	FPI1 is device specific. Refer to FPI chapter for device specific interconnect type. Refer to CPU configuration section for number of FPI master connection for each CPU instances
CLOCK_CPU_fSTM	In	fSTM
SMM_CPU_ApplicationReset	In	Application Reset
SMM_CPU_SystemReset	In	System Reset
SMM_CPU_DebugReset	In	Debug Reset
SMU_CPU1_RESET	In	Kernel Reset
SMU_CPU2_RESET	In	Kernel Reset
SMU_CPU3_RESET	In	Kernel Reset
SMU_CPU4_RESET	In	Kernel Reset
SMU_CPU5_RESET	In	Kernel Reset
CPU0_SMM_STM_RST_REQ	Out	Reset Request from STM
CPU1_SMM_STM_RST_REQ	Out	Reset Request from STM
CPU2_SMM_STM_RST_REQ	Out	Reset Request from STM
CPU3_SMM_STM_RST_REQ	Out	Reset Request from STM
CPU4_SMM_STM_RST_REQ	Out	Reset Request from STM
CPU5_SMM_STM_RST_REQ	Out	Reset Request from STM
SMM_CPU0_REQSLP_IDLE	In	Idle Request
SMM_CPU1_REQSLP_IDLE	In	Idle Request
SMM_CPU2_REQSLP_IDLE	In	Idle Request
SMM_CPU3_REQSLP_IDLE	In	Idle Request
SMM_CPU4_REQSLP_IDLE	In	Idle Request
SMM_CPU5_REQSLP_IDLE	In	Idle Request
SMM_CPUCS_REQSLP_IDLE	In	Idle Request
CPU0_SCU_STM_RST_REQ	Out	Reset Request from STM
CPU1_SCU_STM_RST_REQ	Out	Reset Request from STM
CPU2_SCU_STM_RST_REQ	Out	Reset Request from STM
CPU3_SCU_STM_RST_REQ	Out	Reset Request from STM

(table continues...)

Table 357 (continued) List of CPU interface signals

Interface signals	I/O	Description
CPU4_SCU_STM_RST_REQ	Out	Reset Request from STM
CPU5_SCU_STM_RST_REQ	Out	Reset Request from STM
SCU_CPU0_SOTASEL[1:0]	In	Linear and SOTA bank configuration
SCU_CPU1_SOTASEL[1:0]	In	Linear and SOTA bank configuration
SCU_CPU2_SOTASEL[1:0]	In	Linear and SOTA bank configuration
SCU_CPU3_SOTASEL[1:0]	In	Linear and SOTA bank configuration
SCU_CPU4_SOTASEL[1:0]	In	Linear and SOTA bank configuration
SCU_CPU5_SOTASEL[1:0]	In	Linear and SOTA bank configuration
SCU_CPU0_OVEN	In	Enables or disables the data access overlay
SCU_CPU1_OVEN	In	Enables or disables the data access overlay
SCU_CPU2_OVEN	In	Enables or disables the data access overlay
SCU_CPU3_OVEN	In	Enables or disables the data access overlay
SCU_CPU4_OVEN	In	Enables or disables the data access overlay
SCU_CPU5_OVEN	In	Enables or disables the data access overlay
SCU_CPU0_OVSTART	In	Start of overlay
SCU_CPU1_OVSTART	In	Start of overlay
SCU_CPU2_OVSTART	In	Start of overlay
SCU_CPU3_OVSTART	In	Start of overlay
SCU_CPU4_OVSTART	In	Start of overlay
SCU_CPU5_OVSTART	In	Start of overlay
SCU_CPU0_OVSTOP	In	Stop of overlay
SCU_CPU1_OVSTOP	In	Stop of overlay
SCU_CPU2_OVSTOP	In	Stop of overlay
SCU_CPU3_OVSTOP	In	Stop of overlay
SCU_CPU4_OVSTOP	In	Stop of overlay
SCU_CPU5_OVSTOP	In	Stop of overlay
SCU_CPU0_OVDCINVAL	In	Data cache invalidation (clean entries)
SCU_CPU1_OVDCINVAL	In	Data cache invalidation (clean entries)
SCU_CPU2_OVDCINVAL	In	Data cache invalidation (clean entries)
SCU_CPU3_OVDCINVAL	In	Data cache invalidation (clean entries)
SCU_CPU4_OVDCINVAL	In	Data cache invalidation (clean entries)
SCU_CPU5_OVDCINVAL	In	Data cache invalidation (clean entries)
SCU_CPU0_TRAP	In	NMI Event

(table continues...)

Table 357 (continued) List of CPU interface signals

Interface signals	I/O	Description
SCU_CPU1_TRAP	In	NMI Event
SCU_CPU2_TRAP	In	NMI Event
SCU_CPU3_TRAP	In	NMI Event
SCU_CPU4_TRAP	In	NMI Event
SCU_CPU5_TRAP	In	NMI Event
SCU_CPUCS_CSTRAP	In	NMI Event
CSRM.CSCU_CPUCS_CS_VALID	In	Mark CPUCS (CPUCSCS) bus transactions as security relevant.
CSRM.CSCU_CPUCS_SOTA_SELECT[2:0]	In	Linear and SOTA bank configuration
CSRM.CSCU_CPU_CSAV[1:0]	In	CS activity visibility
CPU0_IR_SRC_CPUxSB	Out	CPU0 service request.
CPU1_IR_SRC_CPUxSB	Out	CPU1 service request.
CPU2_IR_SRC_CPUxSB	Out	CPU2 service request.
CPU3_IR_SRC_CPUxSB	Out	CPU3 service request.
CPU4_IR_SRC_CPUxSB	Out	CPU4 service request.
CPU5_IR_SRC_CPUxSB	Out	CPU5 service request.
CPUCS_IR_SRC_CPUxSB	Out	CPUCS service request.
CPU0_IR_SRC_STMCPUxSR	Out	STM service request
CPU1_IR_SRC_STMCPUxSR	Out	STM service request
CPU2_IR_SRC_STMCPUxSR	Out	STM service request
CPU3_IR_SRC_STMCPUxSR	Out	STM service request
CPU4_IR_SRC_STMCPUxSR	Out	STM service request
CPU5_IR_SRC_STMCPUxSR	Out	STM service request
CPUCS_IR_SRC_STMCPUxSR	Out	STM service request
IR_CPU0_ICU0	In	Interrupt Service Request
IR_CPU1_ICU1	In	Interrupt Service Request
IR_CPU2_ICU2	In	Interrupt Service Request
IR_CPU3_ICU3	In	Interrupt Service Request
IR_CPU4_ICU4	In	Interrupt Service Request
IR_CPU5_ICU5	In	Interrupt Service Request
IR_CPUCS_ICU6	In	Interrupt Service Request
SRI_CPU0_S_SRI0	In	SRI Slave Interface
SRI_CPU1_S_SRI0	In	SRI Slave Interface
SRI_CPU2_S_SRI0	In	SRI Slave Interface

(table continues...)

Table 357 (continued) List of CPU interface signals

Interface signals	I/O	Description
SRI_CPU3_S_SRI0	In	SRI Slave Interface
SRI_CPU4_S_SRI0	In	SRI Slave Interface
SRI_CPU5_S_SRI0	In	SRI Slave Interface
SRI_CPUCS_S_SRI0	In	SRI Slave Interface
SRI_CPU0_S_SRI1	In	SRI NVM Slave Interface
SRI_CPU1_S_SRI1	In	SRI NVM Slave Interface
SRI_CPU2_S_SRI1	In	SRI NVM Slave Interface
SRI_CPU3_S_SRI1	In	SRI NVM Slave Interface
SRI_CPU4_S_SRI1	In	SRI NVM Slave Interface
SRI_CPU5_S_SRI1	In	SRI NVM Slave Interface
SRI_CPUCS_S_SRI1	In	SRI NVM Slave Interface
CPU_FPI_M_FPI0	Out	M_FPI0 Master Interface - Connected to the SPB bus
CPU_FPI_M_FPI1	Out	M_FPI1 Master Interface - Refer to CPU configuration section for number of FPI master connection for each CPU instances
CPU0_SRI_M_SRI	Out	M_SRI Master Interface
CPU1_SRI_M_SRI	Out	M_SRI Master Interface
CPU2_SRI_M_SRI	Out	M_SRI Master Interface
CPU3_SRI_M_SRI	Out	M_SRI Master Interface
CPU4_SRI_M_SRI	Out	M_SRI Master Interface
CPU5_SRI_M_SRI	Out	M_SRI Master Interface
CPUCS_SRI_M_SRI	Out	M_SRI Master Interface
CPU_WTU_SUSOUT	Out	Suspend state.
CPU0_CAN0_Nx_TRIG	Out	VM1 STM Service Request 0 transmit trigger input
CPU0_CAN1_Nx_TRIG	Out	VM1 STM Service Request 0 transmit trigger input
CPU0_CAN2_Nx_TRIG	Out	VM1 STM Service Request 0 transmit trigger input
CPU0_CAN3_Nx_TRIG	Out	VM1 STM Service Request 0 transmit trigger input
CPU0_CAN4_Nx_TRIG	Out	VM1 STM Service Request 0 transmit trigger input
CPU0_CANXL_STM_TRIG	Out	VM1 STM Service Request 0 transmit trigger input

4.10.6 TC4Dx CPU revision history

Reference	Description of change(s)
Date range: 2024-08-17 to 2025-02-07	
HRA_DBGSR	• Replaced incorrect mention of PSW.CDC with "core debug controller"
PFIy_FLASHCON1	• Added clarification that erase counters are only available for NVM Flash based devices
PFIBUFDIS	• PFIBUDIS register now visible in all devices. The register was previously hidden
PROTSPRSE, PROTDLMUSE, PROTSTMSE, PROTPFSE	• Removed SEL bit-field from the Access mode restrictions table as the bit-field is not protected by .OWEN

5 Non-Volatile Memory (NVM)

This sub-system provides non-volatile memory for code and data.

The memory is divided into separate Program Flash (PFLASH) Banks to allow parallel read operations and read-while-write functions. For frequent data updates there is a Data Flash (DFLASH) memory region available, which allows a high number of program or erase cycles to emulate an EEPROM.

The NVM also provides User Configuration Blocks (UCB) where System and NVM configuration settings can be stored. For startup the NVM also contains a BootROM to store the startup software.

For further details about NVM and UCB, please refer to the AURIX™ TC4Dx user manual, under the chapters NVM and Firmware.

5.1 Feature list

- Storage of CPU code and data in several Program Flash (PFLASH) banks with fast read access
  - Independent read access to the banks
  - Read-while-Write capability between the banks
- Storage of EEPROM data in dedicated Data Flash (DFLASH) banks
  - High write or erase cycling endurance
  - DFLASH write or erase while PFLASH read possible
- Security code and data can be stored in separate banks with access protection to the security CPU
- User Configuration Block (UCB) for microcontroller and application specific configuration storage
- Write and erase operations include a verify operation to check the correct cell distribution of programmed and erased bits after the operation
- Two different write operations:
  - Write page with minimum Error Correction Code (ECC) width
  - Write burst for high throughput: more data with one operation
- Erase operation with adjustable range from minimum logical sector size of 16 KB PFLASH and 2 KB DFLASH, up to 512 KB PFLASH and 256 KB DFLASH
- Write (burst) and erase operations may be aborted to allow the start of an other operation
- Replace Logical Sector (RLS) repair function usable by application software in the field to replace a failing logical sector by a free redundant sector
- Separate erase counter for each PFLASH bank: each erase operation in the PFLASH bank is counted and up to 500 entries are possible
- Data integrity check operations
- BootROM for microcontroller start-up software (SSW)

5.2 Functional overview

Blocks of the NVM sub-system

The following figure shows the NVM sub-system and the interfaces to CPU and SRI.

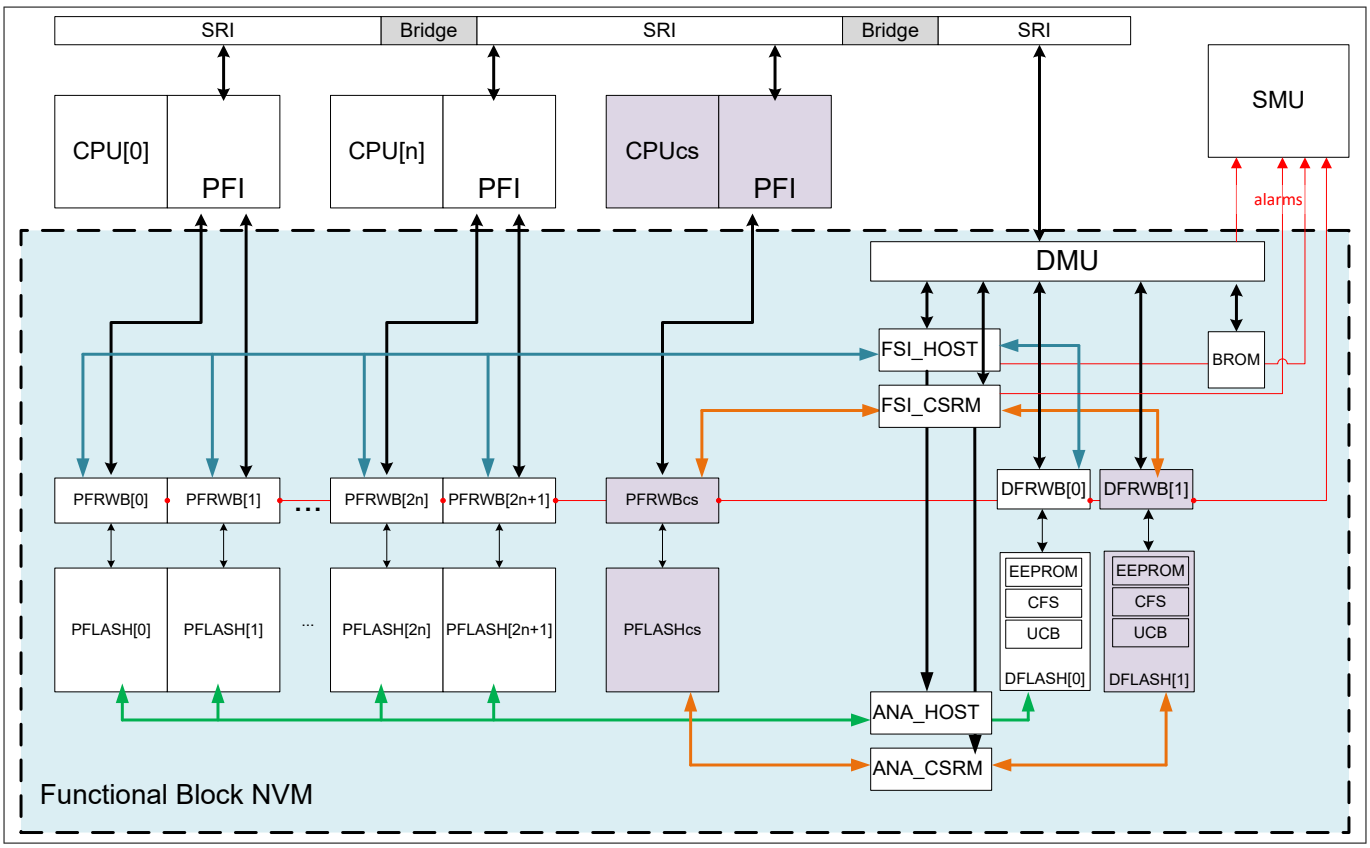


Figure 36 NVM block diagram

For the alarm description please refer to the alarm mapping tables in the SMU functional block and the Safety Manual.

Block descriptions

Table 358 Blocks within the NVM sub-system

Sub-system block	Sub-function	Description
DMU	-	Command Interface Block to System: • Command Decoding of Flash Operation Requests from CPU or other SRI Bus masters • Check of write, read, write_once, and OTP protection • Check of security settings of memory regions • Flash operations to protected memory regions will be prevented Read port and access control for: DFLASH Banks, UCB and BootROM
FSI_x	-	NVM internal CPU for execution of Flash Operation commands such as program, erase, or verify. It has its own storage for firmware for Flash operations
ANALOG_x	-	Central hardmacro block containing chargepumps mainly used for program or erase operation
PFLASH[x]	-	PFLASH bank. A bank containing memory cells for code or data storage of the CPUs, and also containing peripheral circuitry for read, write, or erase operations, 278-bit data output (256-bit data and 22-bit ECC checkbits). Each Bank has a separate address or read path and one bank can be programmed or erased while the other banks are being read (read-while-write support)
PFLASHcs	-	PFLASH bank (as above). Can be set to be exclusively readable, programmable, or erasable by the security CPU
PFRWB[x]	-	PFLASH Read Write Buffers. Digital Interface to the hardmacro PFLASH banks containing address muxes, read buffers, ECC correction, and also control registers for the PFLASH bank
DFLASH[0]	-	Bank containing memory cells for the EEPROM, UCB, and CFS sub-functions. Shared bitline and other read, program or erase peripheral circuitry for operation
	EEPROM	For the storage of data which needs to be frequently updated during its lifetime
	Config Sector (CFS)	Internal parameters and settings of the NVM and the system. Not accessible by the application software
	User Configuration Blocks (UCB)	Infineon and user settings. Part of it can be changed by customer production equipment or application software
DFLASH[1]	-	Separate bank for the EEPROM, UCB, and CFS sub-functions, which can be set to be exclusively readable, programmable, or erasable by the security CPU
	EEPROM	For the storage of data which needs to be frequently updated during its lifetime

(table continues...)

Table 358 (continued) Blocks within the NVM sub-system

Sub-system block	Sub-function	Description
	Config Sector (CFS)	NVM and system internal parameters and settings. Not accessible by the application or security software
	User Configuration Blocks (UCB)	Infineon and user settings. Part of it can be changed by customer production equipment, application and security software
DFRWB[x]	-	DFLASH Read Write Buffers. Digital Interface to the hardmacro DFLASH Banks containing address muxes, read buffers, ECC correction and also control registers for the DFLASH Bank
BROM	-	ROM containing the startup software (SSW)

Related information

TC4Dx SMU alarm mapping tables on page 7233

5.3 Functional description

The following chapters describe the NVM architecture and functions.

5.3.1 NVM sub-functions

Main sub-functions

1. PFLASH safety critical code and data fetch
2. DFLASH data read
3. PFLASH and DFLASH security critical code and data fetch
4. Support of MCU boot
5. Change of NVM content
6. Change of system and NVM configuration

PFLASH safety critical code and data fetch

- Direct fetch path from CPU through CPU PFI Interface to one of the attached PFLASH Bank PFLASH[x]
- Fetch path from CPU to the PFLASH Banks attached to other CPUs through SRI – CPU – CPU PFI

DFLASH data read

- CPU and DMA can read DFLASH[0] EEPROM memory region

PFLASH and DFLASH security critical code and data fetch

- Direct fetch path from CSRM TriCore™ CPU to the attached PFLASH Bank PFLASH[Cx]
- DFLASH[1] can be configured to be only accessible by the CSRM CPU

Support of MCU boot

- BootROM contains the system start-up software which can be read through the SRI connection of the DMU during the system boot phase
- Critical system settings or configurations and trimming values are stored within the config sector and the UCBs and can be read by the start-up software during system boot
- Subset of the UCBs can also be read by the application

Change of NVM content

- Always using a dedicated command sequence through the SRI bus to an allocated memory region (see memory map) which is then decoded in the DMU as a command:
  - Program operations
  - Erase operations
  - Data integrity check operations
  - Replace logical sector operation
- DMU checks the protection or security settings before the execution of a command:
  - If security memory regions have been configured as exclusive then they can be only changed by the security CPU
  - For each logical sector in the PFLASH it is possible to set write, write-once, OTP protection

Change of system and NVM configuration

- During test, the Infineon internal settings and configurations are programmed to config sector and UCB
- Customer production equipment, application or security software changes customer configuration settings like write protection in dedicated UCBs

5.3.2 NVM general terms

Some definitions of terms that are useful to know when reading the NVM chapters.

Table 359 General NVM terms

Application programming interface
Describes all inputs and outputs as well as necessary communication protocols of a software component or a software unit.
Assembly buffer
A special memory area to store data to be programmed into the Flash memory. We distinguish between the FSI SRAM ASB (assembly buffer in the FSI SRAM) and the flash ASB (assembly buffer located in the hardmarco).
Confirmation code
The 32-bit value stored in a predefined NVM memory region, such as a UCB for example, used to identify the state of that memory region. A change of the confirmation value becomes effective after boot or test mode connect. The states that can expressed by the Confirmation Code are: • ERRORED - The state of the confirmation code if it is not CONFIRMED or UNLOCKED. This state indicates that the content of the memory region is somehow corrupted and therefore shall not be used • UNLOCKED - The state where the entire memory region is accessible by read or any NVM operation. In this state the content of that memory region can be modified. The UNLOCKED confirmation code value is 4321_1234_H • CONFIRMED - The state where the content of that memory region is valid and can be used. The access to that memory region is in some way . For example, it is not allowed to modify the memory content. The type of restriction might differ from memory region to memory region. The CONFIRMED confirmation code value is $57B5_327F_H$
Configurable write protection
The removable protection of PFLASH Logical Sector against any NVM operation that can change its content. The write protection is configured in a special UCB and can be removed by software using the password stored in this special UCB. The granularity of the configurable write protection can be selected to be entire PFLASH region or individual PFLASH Banks.
CSRM channel, CSRM command interface
The request channel to execute NVM operations (excluding read operation). Per reset it is exclusive for the CSRM CPU unless the protection is bypassed by UCB setting. The CSRM channel requests are handled by the FSI_{CSRM}
Endurance
The maximum number of program or erase cycles of each flash cell is called "endurance". It is a statistical figure that depends on operating conditions, the use of the Flash cells, and the required quality level.
Erase counter
A special memory region to record the erase events.

(table continues...)

Table 359 (continued) General NVM terms

Erasing
The process of forcing a programmed Flash memory cell to the erased state. The erased state of the Flash cell is logical 0_B. The erased state is not necessarily an ECC codeword. Reading an erased page might result in ECC failures and an ECC correction might by applied. Complete Logical Sectors are erased. All Flash cells in this area incur one "cycle" that counts for the "endurance".
Flash embedded RISC (FLER)
A small processor in the FSI to control FSI operations. The µCode is executed on this processor.
Flash interface - flash standard interface (FSI)
The digital control logic to control the read accesses and any write or erase accesses to the NVM.
HOST channel, HOST command interface
The request channel to execute NVM operations (excluding read operation) mainly for the HOST standard processor(s) [or TriCore™s]. Per reset the HOST command channel access is set to CPU(0) but this can be changed to allow several other CPUs (including the CSRM CPU) access. The HOST channel requests are handled by the FSI_{HOST}
NVM bank idle state
The state where no read and no operation on that NVM Bank is running, and any new read or operation on that bank is possible.
NVM idle state
The state where no NVM read and no NVM operation is running and any new NVM read or new NVM operation is possible.
NVM operation
An operation that is targeting or ongoing at either: **1.** modifying **2.** analyzing the contents of the NVM **3.** changing the redundancy configuration of the NVM **4.** changing the state of the NVM Whenever a NVM operation is executed the FSI and the µCode are involved. ***Note**: 'Operation' means every function of the NVM except the read. Examples are Program and Erase, but also Verify, memory bit counting, integrity checks, or changing redundancy. It also means any operation that changes the status such as start-up, sleep, wake up, or shutdown for example.*
Program erase (p/e) cycle
A sequence consisting of one program and one erase (or vice-versa) in the NVM.
Programming
The programmed state of a Flash cell is logical 1_B. Changing an erased Flash cell to this state is called "programming".
Retention
This is the time during which the data of a Flash cell can be read reliably. The retention time is a statistical figure that depends on the operating conditions of the device (such as temperature profile) and is affected by operations on other flash cells in the same Wordline and physical sector. With an increased number of p/e cycles (see endurance) the retention may be impacted.

(table continues...)

Table 359 (continued) General NVM terms

Test mode
A special set of NVM operations to support the NVM test. Test mode operations are not available to the customer.
Test Pass Marker
The 32-bit value stored in the UCB_TEST; it is programmed to 80658383_H when all Infineon's production tests are passed. A production device with a Test Pass Marker value different than 80658383_H, is specified to be discarded. This is to prevent any possibility of an unknown fault in Infineon production or customer procurement process, resulting in an invalid device used by the customer. In such cases Infineon appreciates a notification.
µCode
Software that is executed on the FSI.
User mode
A well defined set of NVM operations that are provided to support common Flash functions like erase or program. In contrast to the Test Mode operations, all user mode functions are delivered to the custome r,although some user mode functions might be by the DMU. The (full set of) user mode NVM operations are available after NVM boot phase.
Verify
A process to check the programmed and erased state of Flash cells against a given p/e criteria. Different verify operations (Program Verify, Erase Verify, or Tight Margin Reads for example) are possible. This term shall be used as a collection term for all of these verify options.

5.3.3 NVM architecture terms

Definitions of some special terms related to the NVM architecture.

Table 360 NVM architecture terms

Flash module
A module that contains a Non-Volatile Memory (NVM) with its own digital control logic.
Block
A Block consists of multiple (but at least one) Banks physically grouped together. All Banks of a Block share a common "Block analog" region.
Bank
An NVM memory region with own address and read path. A Bank consists of *b* Physical Sectors. The whole NVM is divided into Banks. There are two different bank types: • PFLASH Bank: used for code or data • DFLASH Bank: used for data which is continuously changed during the application run (EEPROM emulation) Both Bank types have different configurations in terms of LOGSEC size, WL size, Page size, and ECC capabilities for example.
Segment
A segment consists of *s* Physical Sectors with the same Bitline redundancy.
Physical sector
A contiguous memory region within a Bank which has separate local bitlines. A Physical Sector consists of *m* Logical Sectors for code and data storage (where *m* depends on the Flash Bank type). A Physical Sector also contains *r* additional Logical Sectors for special purpose: • Redundant Logical Sectors • PFLASH: Erase Counter Logical Sector (only in one Physical Sector of a Bank)
Logical sector
The smallest erasable contiguous memory region within a Physical Sector. It consists of *w* consecutive Wordlines (where *w* depends on the Flash Bank type). In AURIX™ TC4xx a Logical Sector is equal to a Mini Sector. The Logical Sector, Mini Sector and Sector are effectively synonymous terms.
Mini sector
The Logical Sector, Mini Sector, and Sector are synonymous terms in AURIX™ TC4xx.
Sector
The Logical Sector, Mini Sector, and Sector are synonymous terms in AURIX™ TC4xx.
Wordline
A set of *p* consecutive Pages, where *p* depends on the Flash Bank type.
Page
A set of *b* consecutive bytes, where *b* depends on the Flash Bank type. It is the smallest unit that can be programmed. A Page consists of: • Data part that stores the customer data • ECC part that stores an Flash Bank type specific ECC checksum • A well-defined number of spare bits that can be used to repair defect bits within a Page (so-called Bitline redundancy) The exact dimension of these parts depend on the Flash Bank type.

The following figure depicts the relationship between the NVM architecture terms.

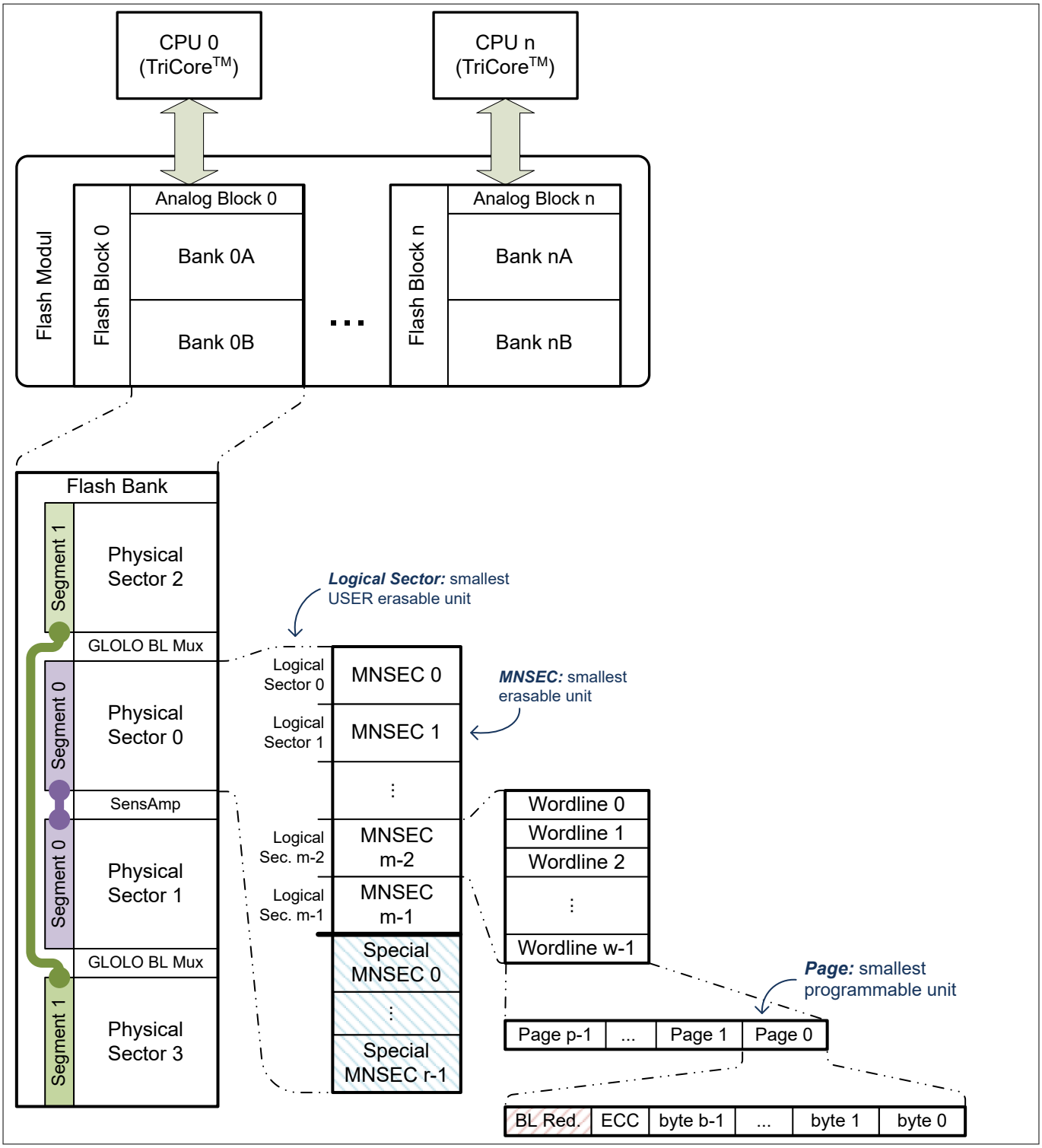


Figure 37 Logical structure of flash memory

Related information

Flash structure on page 656

5.3.4 NVM abbreviations

Table 361 NVM abbreviations

ASB
Assembly Buffer
BL
Bitline
CFS
Configuration Sector
CFTM
CPU Flash Test Mode
CSRM
Cyber Security Real Time Module
DECTED
Double Error Correcting, Triple Error Detecting
DF or DFLASH
Data Flash or Data Memory In the AURIX™ 3rd Generation family of products, the NVM distinguishes between HOST and CSRM Data Flashes, so the following refinement is introduced: • $DFLASH_{CSRM}$ or CSRM DFLASH or csDFLASH - Describes a DFLASH that belongs to the CSRM domain • $DFLASH_{HOST}$ or HOST DFLASH - A DFLASH that belongs to the HOST domain Using the abbreviation DF or DFLASH without an additional domain extension addresses both domains. Enumeration of Data Flashes is performed using the following syntax: • DFLASH(x) or $DFLASH_{CSRM}$(x), where *x* is either a placeholder for all available Data Flashes or a dedicated decimal number that identifies a specific Flash Bank. The following figure illustrates the enumeration using the current POR. Since only one Bank is planned per domain, most of the names are equivalent

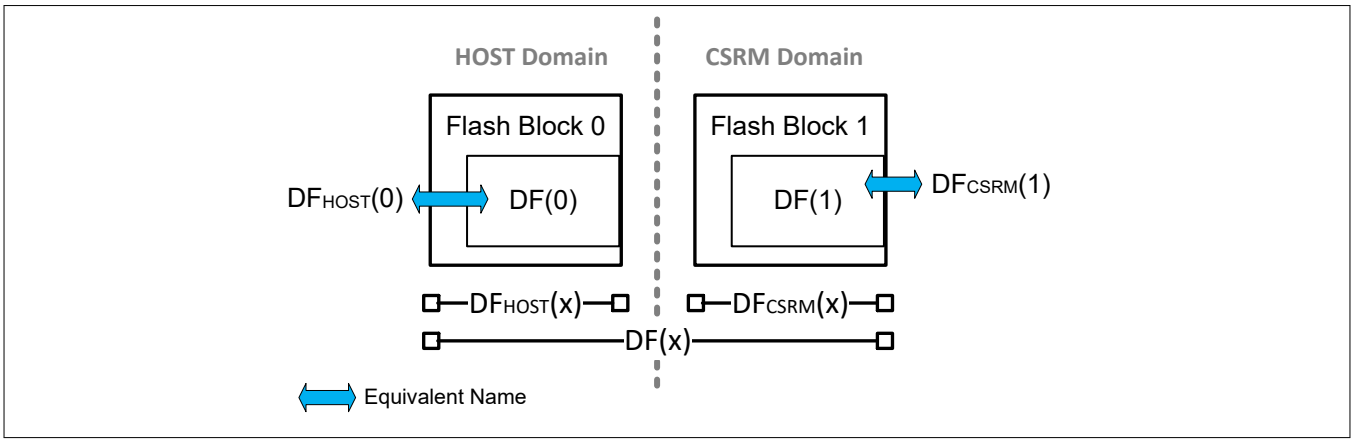


Figure 38 DFLASH enumeration

(table continues...)

Table 361 (continued) NVM abbreviations

DMU
Data Memory Unit
EC
Erase Counter
ECC
Error Correction Code. This is usually not the code itself but the mechanism to detect and correct a failure within a word.
EoR
End of Request
FLER
Flash Embedded RISC
FSI
Flash Standard Interface
FW
FSI Firmware or µCode
GTFM
General Flash Test Mode
GLOLO
Global Local
ISR
Interrupt Service Routine
LOGSEC
Logical Sector
MBP
Multi-Bank Parallel
MNSEC
Mini Sector
NVM
Non-Volatile Memory

(table continues...)

Table 361 (continued) NVM abbreviations

PF or PFLASH
Program Flash or Program Memory In the AURIX™ 3rd Generation product family, the NVM functional block distinguishes between HOST and CSRM Program Flashes, so the following refinement is introduced: • $PFLASH_{CSRM}$/CSRM PFLASH/csPFLASH - Describes a PFLASH that belongs to the CSRM domain • $PFLASH_{HOST}$/HOST PFLASH - A PFLASH that belongs to the HOST domain Using the abbreviation PF or PFLASH without an additional domain extension addresses both domains. Enumeration of Program Flashes is carried out using the following syntax: • PFLASH(x) or $PFLASH_{HOST}$(x): where *x* is either a placeholder for all available Program Flashes or a dedicated identifier that identifies a specific Flash Bank. Since a PFLASH Bank is part of a Flash Block that usually consists of two Banks *A* and *B*, the unique identifier shall be expressed by *x=x'{A|B}*, where *x'* defines the unique ID of a Flash Block (corresponds to CPU ID). Example: $PFLASH_{HOST}$(x) means all HOST Program Flashes while $PFLASH_{HOST}$(0B) means the second PFLASH Bank of the first PFLASH Bock of the HOST domain

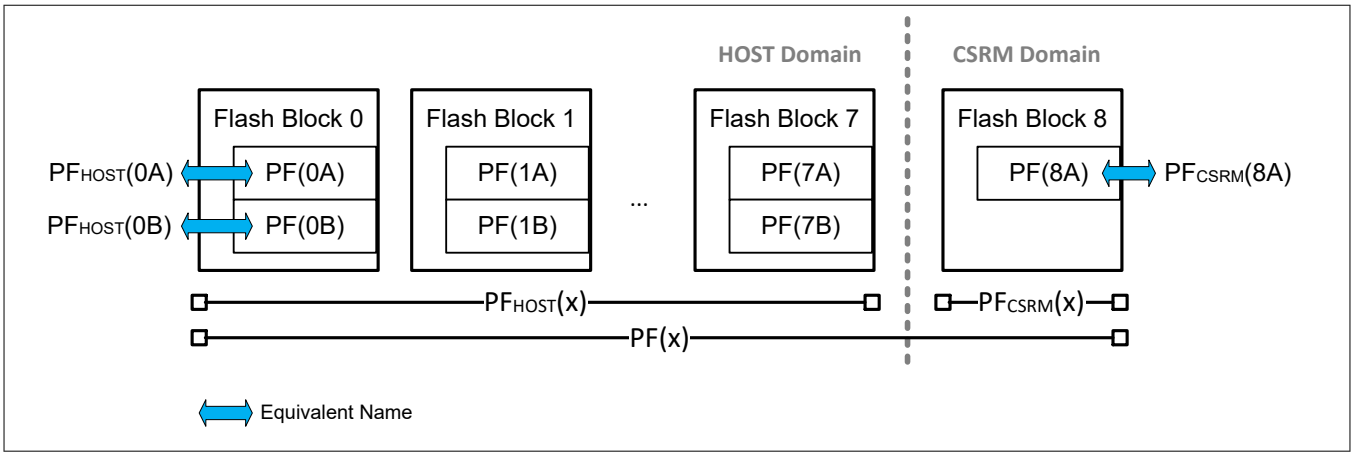


Figure 39 PFLASH enumeration

PFI
Program Flash Interface
PHYSEC
Physical Sector
RED
Redundancy. For example BL RED for bit-line redundancy or MNSEC RED as mini sector redundancy.
RLS
Replace Logical Sector
RWB
Read Write Buffer

(table continues...)

Table 361 (continued) NVM abbreviations

SFR
Special Function Register
TECQED
Triple Error Correcting; Quadruple Error Detecting
TM
Test Mode
UCB
User Configuration Block
UM
User Mode
WL
Wordline

5.3.5 Flash structure

The following chapters describe the sector structure and addresses of the PFLASH - and DFLASH - Banks. Please also refer to chapter NVM architecture terms for a description of the micro-architecture within a Logical Sector (wordlines, pages, bitlines).

Related information

NVM architecture terms on page 650

5.3.5.1 Program flash banks

Sector structure

All PFLASH Banks PFp are based on the same sector structure.

There are different sizes of PFp Banks available. The PFp Banks implement the following sector structure:

- 2 Mbyte Program Flash Bank PFp implements logical sectors S0 to S127
- 1 Mbyte Program Flash Bank PFp implements logical sectors S0 to S63

Table 362 Sector structure of PFp

Logical sector	Physical sector	Offset address	Size	Total
S0	PS0	$00'0000_H$	16 Kbyte	16 Kbyte
S1		$00'4000_H$	16 Kbyte	32 Kbyte
S2		$00'8000_H$	16 Kbyte	48 Kbyte
...		...	...	...
S30		$07'8000_H$	16 Kbyte	496 Kbyte
S31		$07'C000_H$	16 Kbyte	512 Kbyte
S32	PS1	$08'0000_H$	16 Kbyte	528 Kbyte
S33		$08'4000_H$	16 Kbyte	544 Kbyte
S34		$08'8000_H$	16 Kbyte	560 Kbyte
...		...	...	...
S62		$0F'8000_H$	16 Kbyte	1008 Kbyte
S63		$0F'C000_H$	16 Kbyte	1024 Kbyte
S64	PS2	$10'0000_H$	16 Kbyte	1040 Kbyte
S65		$10'4000_H$	16 Kbyte	1056 Kbyte
...		...	...	...
S94		$17'8000_H$	16 Kbyte	1520 Kbyte
S95		$17'C000_H$	16 Kbyte	1536 Kbyte
S96	PS3	$18'0000_H$	16 Kbyte	1552 Kbyte
S97		$18'4000_H$	16 Kbyte	1568 Kbyte
...		...	...	...
S126		$1F'8000_H$	16 Kbyte	2032 Kbyte
S127		$1F'C000_H$	16 Kbyte	2048 byte

***Note**: As option there are also 512 Kbyte Banks planned for certain products.*

Please refer to the system memory map for an overview of how many 2 Mbyte and 1 Mbyte Banks are available for a product and the Bank addresses.

Security code PFLASH bank

There is a separate Program Flash Bank for the Cyber Security CPU.

5.3.5.2 Data flash banks

The AURIX™ TC4x has 2 different Data Flash Banks which are used for different purposes. The following table shows the two banks and namings which can be used as synonyms.

Table 363 Data flash banks

Bank	Synonyms	Available sizes	Used for
DFLASH(0)	DFLASH(HOST), DF(0)	1024 Kbyte, 512 Kbyte, 128 Kbyte	Data, EEPROM emulation for the HOST application
DFLASH(1)	DFLASH(CSRM), DF(1)	128 Kbyte	Data, EEPROM emulation for the CSRM security application

***Note**: The size of a DFLASH Bank is set according to its EEPROM region size (see below).*

***Note**: Access rights for the DFLASH(1) Bank are usually set to the security CPU only. For initial factory programming this can be bypassed by UCB but shall be set for normal operation. Please refer to the description of the chapter about protection for further details.*

A data flash bank consists of 3 parts, which are explained in the following table.

Table 364 Data flash bank parts

Memory region	Description
EEPROM Region	Data flash region which can be used to store data during application run and to emulate an EEPROM
UCB Region	User Configuration Blocks
Config Sector Region (CFS)	Storage of device configuration data (recovered during start-up), not accessible for normal operation

There are different sizes of DFp Banks available. The size of UCB and CFS regions are always the same but the size of the actual EEPROM region differs and is use as classifier of the Bank; for example: a "1 MB" DFLASH Bank contains an EEPROM region of 1 MB but contains also additional regions for UCB and CFS (for the UCB sizes please refer to the UCB description).

The EEPROM region of the DFp Banks implement the following sector structure:

- 1 Mbyte data flash bank DFp implement logical sectors EEPROM0 to EEPROM511
- 512 Kbyte data flash bank DFp implement logical sectors EEPROM0 to EEPROM255

Data flash banks ≥ 256 Kbyte have 128 Kbyte physical sectors and 2 Kbyte logical sectors. Smaller data flash banks have 64 Kbyte physical sectors and 2 Kbyte logical sectors.

Table 365 Sector structure of EEPROM region of a DFLASH bank of 1 Mbyte

Logical sector	Physical sector (128 Kbyte)	Offset address EEPROM region	Size	Total
EEPROM0	PS0	$00'0000_H$	2 Kbyte	2 Kbyte
EEPROM1		$00'0800_H$	2 Kbyte	4 Kbyte
…		…	…	…
EEPROM62		$01'F000_H$	2 Kbyte	126 Kbyte
EEPROM63		$01'F800_H$	2 Kbyte	128 Kbyte
EEPROM64	PS1	$02'0000_H$	2 Kbyte	130 Kbyte
EEPROM65		$02'0800_H$	2 Kbyte	132 Kbyte
…		…	…	…
EEPROM126		$03'F000_H$	2 Kbyte	254 Kbyte
EEPROM127		$03'F800_H$	2 Kbyte	256 Kbyte
EEPROM128	PS2	$04'000_H$	2 Kbyte	258 Kbyte
EEPROM129		$04'0800_H$	2 Kbyte	260 Kbyte
…		…	…	…
EEPROM190		$05'F000_H$	2 Kbyte	382 Kbyte
EEPROM191		$05'F800_H$	2 Kbyte	384 Kbyte
EEPROM192	PS3	$06'0000_H$	2 Kbyte	386 Kbyte
EEPROM193		$06'0800_H$	2 Kbyte	388 Kbyte
…		…	…	…
EEPROM254		$07'F000_H$	2 Kbyte	510 Kbyte
EEPROM255		$07'F800_H$	2 Kbyte	512 Kbyte
EEPROM256	PS4	$08'0000_H$	2 Kbyte	514 Kbyte
EEPROM257		$08'0800_H$	2 Kbyte	516 Kbyte
…		…	…	…
EEPROM318		$09'F000_H$	2 Kbyte	638 Kbyte
EEPROM319		$09'F800_H$	2 Kbyte	640 Kbyte
EEPROM320	PS5	$0A'0000_H$	2 Kbyte	642 Kbyte
EEPROM321		$0A'0800_H$	2 Kbyte	644 Kbyte
…		…	…	…
EEPROM382		$0B'F000_H$	2 Kbyte	766 Kbyte
EEPROM383		$0B'F800_H$	2 Kbyte	768 Kbyte
EEPROM384	PS6	$0C'000_H$	2 Kbyte	770 Kbyte
EEPROM385		$0C'0800_H$	2 Kbyte	772 Kbyte

(table continues...)

Table 365 (continued) Sector structure of EEPROM region of a DFLASH bank of 1 Mbyte

Logical sector	Physical sector (128 Kbyte)	Offset address EEPROM region	Size	Total
…		…	…	…
EEPROM446		$0D'F000_H$	2 Kbyte	894 Kbyte
EEPROM447		$0D'F800_H$	2 Kbyte	896 Kbyte
EEPROM448	PS7	$0E'0000_H$	2 Kbyte	898 Kbyte
EEPROM449		$0E'0800_H$	2 Kbyte	900 Kbyte
…		…	…	…
EEPROM510		$0F'F000_H$	2 Kbyte	1022 Kbyte
EEPROM511		$0F'F800_H$	2 Kbyte	1024 Kbyte

Table 366 Sector structure of EEPROM region of a DFLASH bank of 512 Kbyte

Logical sector	Physical sector (128 Kbyte)	Offset address EEPROM region	Size	Total
EEPROM0	PS0	$00'0000_H$	2 Kbyte	2 Kbyte
EEPROM1		$00'0800_H$	2 Kbyte	4 Kbyte
…		…	…	…
EEPROM62		$01'F000_H$	2 Kbyte	126 Kbyte
EEPROM63		$01'F800_H$	2 Kbyte	128 Kbyte
EEPROM64	PS1	$02'0000_H$	2 Kbyte	130 Kbyte
EEPROM65		$02'0800_H$	2 Kbyte	132 Kbyte
…		…	…	…
EEPROM126		$03'F000_H$	2 Kbyte	254 Kbyte
EEPROM127		$03'F800_H$	2 Kbyte	256 Kbyte
EEPROM128	PS2	$04'000_H$	2 Kbyte	258 Kbyte
EEPROM129		$04'0800_H$	2 Kbyte	260 Kbyte
…		…	…	…
EEPROM190		$05'F000_H$	2 Kbyte	382 Kbyte
EEPROM191		$05'F800_H$	2 Kbyte	384 Kbyte
EEPROM192	PS3	$06'0000_H$	2 Kbyte	386 Kbyte
EEPROM193		$06'0800_H$	2 Kbyte	388 Kbyte
…		…	…	…
EEPROM254		$07'F000_H$	2 Kbyte	510 Kbyte
EEPROM255		$07'F800_H$	2 Kbyte	512 Kbyte

There is also a small DataFlash Bank with 128 Kbyte available which has the following sector structure.

Table 367 Sector structure of EEPROM region of a DFLASH bank of 128 Kbyte

Logical sector	Physical sector (64 Kbyte)	Offset Address EEPROM region	Size	Total
EEPROM0	PS0	$00'0000_H$	2 Kbyte	2 Kbyte
EEPROM1		$00'0800_H$	2 Kbyte	4 Kbyte
…		…	…	…
EEPROM30		$00'F000_H$	2 Kbyte	62 Kbyte
EEPROM31		$00'F800_H$	2 Kbyte	64 Kbyte
EEPROM32	PS1	$01'0000_H$	2 Kbyte	66 Kbyte
EEPROM33		$01'0800_H$	2 Kbyte	68 Kbyte

(table continues...)

Table 367 (continued) Sector structure of EEPROM region of a DFLASH bank of 128 Kbyte

Logical sector	Physical sector (64 Kbyte)	Offset Address EEPROM region	Size	Total
…		…	…	…
EEPROM62		$01'F000_H$	2 Kbyte	126 Kbyte
EEPROM63		$01'F800_H$	2 Kbyte	128 Kbyte

Related information

Operation protection overview on page 759
DFLASH master read protection on page 769

5.3.5.3 Logical sector

A logical sector is the smallest erasable contiguous memory region within a Physical Sector. It consists of *w* consecutive wordlines. Each wordline has *p* consecutive Pages; a Page is the smallest unit that can be programmed using the `Write Page` command. The size of the logical sector, its number of wordlines and pages per wordline depend on the Bank type (PFLASH or DFLASH).

Table 368 Logical sector structure

Bank type	Logical sector size [Kbyte]	Number of wordlines	Wordline size [Byte]	Number of pages per wordline
PFLASH	16	16	1024	32
DFLASH	2	4	512	64

***Note**: one UCB in DFLASH has the same structure as a Logical Sector in DFLASH.*

Table 369 PFLASH logical sector offset addresses

Wordline	Offset address	Total size [Kbyte]
0	0000_H	1
1	0400_H	2
2	0800_H	3
..	...	
14	3800_H	15
15	$3C00_H$	16

Table 370 DFLASH logical sector offset addresses

Wordline	Offset address	Total size [Kbyte]
0	0000_H	0.5
1	0200_H	1
2	0400_H	1.5
3	0600_H	2

Table 371 PFLASH wordline offset addresses

Page	Offset address	Total size [Byte]
0	0000_H	32
1	0020_H	64
2	0040_H	96
..	...	
30	$03C0_H$	992
31	$03E0_H$	1024

Table 372 DFLASH wordline offset addresses

Page	Offset address	Total size [Byte]
0	0000_H	8
1	0008_H	16
2	0010_H	24
..	...	
62	$01F0_H$	504
63	$01F8_H$	512

5.3.5.4 Erase counters

For each PFLASH(PFp) Bank there is a dedicated Erase Counter(EC), which is an additional 16 Kbyte logical sector within each PFLASH(PFp) Bank, but belongs to a separate system address region (see system memory map). The erase counter is read-only in the application.

Erase logging

For every call of an “Erase Logical Sector Range” command sequence the NVM automatically records the logical sector start address and the number of logical sectors in a 256-bit log in the corresponding Erase Counter (ECp) memory region. The sector start address is not the system address, but the address of the selected logical sector in the local PFLASH Bank. For the address of the erase counter region please refer to the system memory map: the entry is "NVM.PFLASHxy (Erase Counter)" with x for CPU number and y = 0 or 1 for one of the 2 local PFLASH banks of a CPU.

The ECp 16 Kbyte erase counter supports the recording of 2 x 250 x 256-bit log entries. The Erase Counter area in each PFLASH is divided into two areas supporting 250 entries each - of which the first entry is pre-programmed and the remaining 249 entries are used to record erase operations. The first 250 entries from the Erase Counter base address is the Low priority area and the remaining is the High priority area. The user can assign priority to a logical sector in the PFLASH using the UCB_RTC_ECPRIO. When an erase encompassing at least one high priority logical sector is triggered, it is recorded in the high priority area of the Erase Counter. Otherwise, the erase is recorded in the low priority area. The first entry of each area is a pre-programmed ECC valid entry, and cannot be used for Erase Counter entries.

For TC4xx the erase counter region does not start at the start address of a 16 Kbyte logical sector but at page with offset address 0180_H (rationale: the memory range from offset address 0000_H to 0160_H is reserved).

Table 373 Erase counter logical sector structure

Page	Offset address	Purpose
0	0000_H	Reserved
1	0020_H	
..	..	..
3		
...		
10	0140_H	
11	0160_H	
12	0180_H	Erase Counter LowPrio Entry 0 (dummy entry to have valid ECC)
13	$01A0_H$	Erase Counter LowPrio Entry 1
		..
260	2080_H	Erase Counter LowPrio Entry 248
261	$20A0_H$	Last possible Erase Counter LowPrio Entry 249
262	$20C0_H$	Erase Counter HighPrio Entry 0 (dummy entry to have valid ECC)
263	$20E0_H$	Erase Counter HighPrio Entry 1
..	..	..
510	$3FC0_H$	Erase Counter HighPrio Entry 248
511	$3FE0_H$	Last possible Erase Counter HighPrio Entry 249

Each EC entry (including the first pre-programmed entry) has a 4 byte log marker. The marker is used to indicate the last valid EC entry. If the marker value of the entry is not equal to 0x00000000, the next EC entry is valid and can be read. If the marker value of entry is equal to 0x00000000, the last valid EC entry is reached. Thus, the next entry on address is empty (erased state) and shall not be read (would generate in most cases an uncorrectable error).

This is the reason why the first entry of a erase counter region is pre-programmed by Infineon during product test and cannot be used for erase counting. The next figure shows the erase counter structure and an application use case where a part of the erase counter entries has already been written due to customer or application erase requests.

Table 374 256-bit erase counter logging entry structure

Bits	Description
[31:0]	Marker (4 byte)
[55:32]	Start address of the first PFp logical sector (3 byte)
[63:56]	Number of logical sectors erased (1 byte)
[255:64]	Reserved

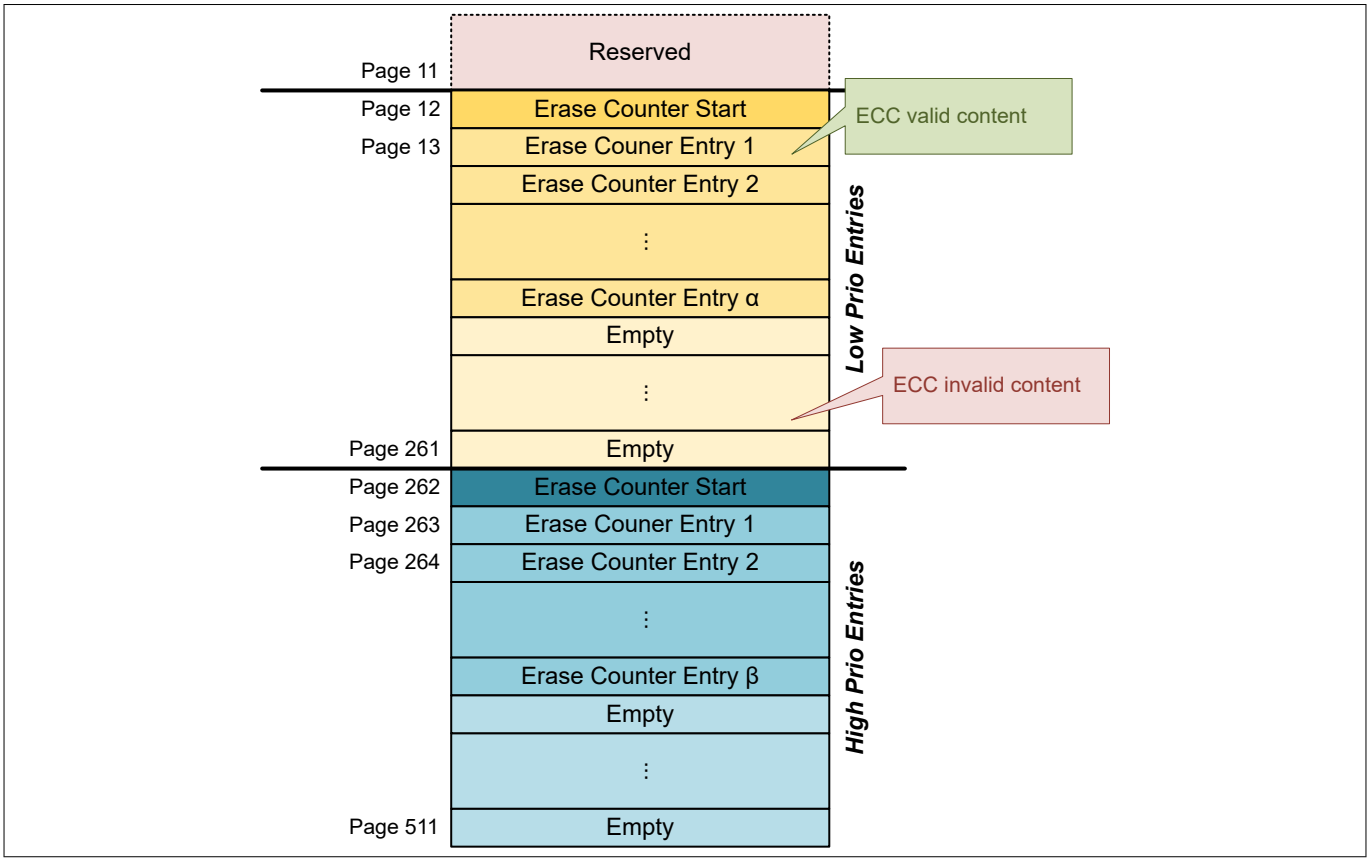


Figure 40 Erase counter structure and use case

Erase counter access

Erase counters can be read by any bus master that has the rights to read the associated PFLASH bank. All reads are performed by the local CPU through the connected flash interface. All other flash operations on the Erase Counter regions fail with an ERROR (see command sequence chapter for more details).

Erase counter registers and configuration

For each logical sector of a PFLASH Bank it is possible to configure it as "low"- or "high"-priority sector for erase counter monitoring. The DMU registers DMU_GP_HOST_PFECPRIOpXn show the settings. The following table explains the "pXn" indexing of the registers:

Table 375 Indexing of bank registers

Index	Description
p	Number of CPU
X	= A or =B for Bank A or Bank B of a CPU
n	Number of physical sector: for a 2 MB Bank n = (0..3) for 4 physical sectors with each having 32 logical sectors (=> 32 bits of register)

The registers DMU_GP_HOST_PFECPRIOpXn are read only and are read after a reset by the DMU from the UCB_RTC_ECPRIO_ORIG or UCB_RTC_ECPRIO_COPY. It is possible to change these UCBs by write or erase commands if the correct password has been applied using the `Disable Protection` command. Please refer to the UCB address map overview chapter for the offset address of UCB and the offset addresses of UCB content, confirmation codes and passwords.

The following table shows an example for the erase counter priority registers for the Bank B of CPU(1):

Table 376 Example of ECPRIO registers for bank B of CPU(1)

Register	High or Low priority settings for
DMU_GP_HOST_PFECPRIO1B0	logical sectors 0..31
DMU_GP_HOST_PFECPRIO1B1	logical sectors 31..63
DMU_GP_HOST_PFECPRIO1B2	logical sectors 64..95
DMU_GP_HOST_PFECPRIO1B3	logical sectors 96..127

CSRM specific erase counter registers and UCB

The erase counter settings for the PFLASH Bank of the CSRM are stored in the registers DMU_GP_CSRM_PFECPRIOn with n = (0..3). These registers are read only and are read after a reset by the DMU from the UCB_CS_ECPRIO_ORIG or UCB_CS_ECPRIO_COPY.

Related information

Error handling on page 685

UCB on page 668

5.3.5.5 UCB

The AURIX™ TC4xx has two regions of UCB (User Configuration Blocks): one region in DFLASH(HOST) Bank and one region in DFLASH(CSRM) Bank.

Table 377 UCB memory regions

DFLASH bank	UCB	Size
DF(0):DFLASH(HOST)	NVM.UCB0	80 Kbyte
DF(1):DFLASH(CSRM)	NVM.UCB1	52 Kbyte

***Note**: please refer to the system memory map for the addresses of the UCB using the references NVM.UCBx mentioned in the table*

Master access restrictions

The following table lists the master access protections for the two UCB regions based on reset values of the APU registers.

Table 378 Master access protections of UCB memory regions

UCB	Read	Write
NVM.UCB0	all	CPU0
NVM.UCB1	CPUcs	CPUcs

***Note**: for a write access it is also necessary that the command interface (HOST or CSRM) may also be accessed by the master*

The access restriction may be bypassed by a specific setting of the APUBYPASS set within the UCB_CS_PFLASH_OTP0 of NVM.UCB1 which is recovered during startup by the DMU.

UCB specific access restrictions

In addition to the master access restrictions there are also additional restrictions concerning read and or write, which are UCB specific; for example UCB with passwords do not allow the read access to password region or UCB for permanent write protection do not allow erase once the UCB is in CONFIRMED state. Please refer to the description of the UCB for UCB specific read and write restrictions.

References to detailed UCB description

Please see the following reference for a detailed description of the UCB offset addresses and content: Chapter **Firmware → Registers**.

References to UCB handling and NVM UCB usage

The following list contains references to chapters describing the handling of the UCB and the usage of UCBs to store NVM block configuration settings like sector protection.

Related information

UCB handling on page 750
UCB confirmation code and UCB state evaluation on page 750
UCB COPY and ORIG update procedure on page 754
Erase counters on page 665
Sector protection configuration and registers on page 775
UCB for APU bypass on page 782

Master access protection life cycle on page 779

5.3.6 Read access

If the Flash Banks are in Read Mode then all flash read accesses are memory mapped reads. Read protection can be used to block a read.

5.3.6.1 Transaction types

Program flash

The transaction types for **local** CPU and **remote** CPU read accesses to a PFLASH are as follows:

Table 379 Local CPU read accesses to PFLASH

Local CPU access	Bus	Code fetch	Data constant
Cacheable address	DPI	Block Transfer 4 (BTR4)	Block Transfer 4 (BTR4)
Non-cacheable address	DPI	Block Transfer 4 (BTR4)	Minimum width

Table 380 Remote CPU read accesses to PFLASH

CPU access	Bus	Code fetch	Data constant
Cacheable address	SRI	Block Transfer 4 (BTR4)	Block Transfer 4 (BTR4)
Non-cacheable address	SRI	Block Transfer 4 (BTR4)	Minimum width

Data flash

Read accesses to DFLASH must be single transfers made across the SRI and are available only in the non-cacheable address range. A Block Transfer will result in a bus error.

5.3.6.2 Configuring read access cycles

Configuring PFLASH read access cycles

Read accesses to the PFLASH have an access delay setting counted with the f_{SRI} clock. The setting is defined in the DMU_GP_HOST_PFWAIT register.

In the datasheet the parameter t_{PF} defines the read access time to a PFLASH Bank in [ns]. This needs to be converted into a number of read cycles based on the actual **SRI** frequency using the formula mentioned in the table below.

Table 381 PFLASH read access cycle calculation

	Minimum value[1)]
PFLASH read cycles	Ceiling(t_{PF} * f_{SRI})

1) The Ceiling(r) function rounds up a real number, i.e., the result is the smallest integer not less than the real argument.

The "number of read cycles - 1" has to be programmed into the following register bit field: DMU_GP_HOST_PFWAIT.RFLASH

***Attention**: In case of using the frequency modulation of the clock the maximum frequency of the f_{SRI} clock has to enter the calculation. This could mean an additional wait cycle in worst case.*

PFLASH read cycles example

Example configurations with different SRI clock frequencies are shown in the table below.

Table 382 Example configurations for PFLASH read access time in DMU_GP_HOST_PFWAIT.RFLASH

Read access time t_{PF}	**f_{SRI} MAX [MHz]**	**Frequency modulation (MA = 2%)**	**Read cycles**	**Wait states**
15	400	OFF	6	5
15	408	ON	7	6
15	450	OFF	7	6
15	459	ON	7	6
15	500	OFF	8	7
15	510	ON	8	7

***Attention**: Values mentioned in the previous table are meant as examples. Please refer to the actual product datasheet for applicable values of t_{PF} and f_{SRI}.*

Configuring DFLASH read access cycles

Read accesses to the DFLASH have an access delay setting counted with the f_{FSI} clock. The setting is defined in the DMU_GP_HOST_DFWAIT register.

In the datasheet the parameter t_{DF} defines the read access time to a DFLASH Bank in [ns]. This needs to be converted into a number of read cycles based on the actual **FSI** frequency using the formula mentioned in the table below.

Table 383 DFLASH read access cycle calculation

	Minimum value[1)]
DFLASH read cycles	Ceiling(t_{DF} * f_{FSI})

1) The Ceiling(r) function rounds up a real number, i.e., the result is the smallest integer not less than the real argument.

The "number of read cycles - 1" has to be programmed into the following register bit field: DMU_GP_HOST_DFWAIT.RFLASH

***Attention**: In case of using the frequency modulation of the clock the maximum frequency of the f_{SRI} clock has to enter the calculation. This could mean an additional wait cycle in worst case.*

DFLASH read cycles example

Example configurations with different FSI clock frequencies are shown in the table below.

Table 384 Example configurations for DFLASH read access time in DMU_GP_HOST_DFWAIT.RFLASH

Read access time t_{DF}	**f_{FSI} MAX [MHz]**	**Frequency modulation (MA = 2%)**	**Read cycles**	**Wait states**
50	100	OFF	5	4
50	102	ON	6	5
50	112.5	OFF	6	5
50	114.75	ON	6	6

***Attention**: Values mentioned in the previous table are meant as examples. Please refer to the actual product datasheet for applicable values of t_{DF} and f_{FSI}.*

Separate wait state settings for the security memory banks

For the security memory Banks there is a separate setting of wait states necessary in the following registers:

Table 385 Wait state settings for the security memory banks

Bank	**Wait state setting in register**
CSRM PFLASH	DMU_GP_CSRM_PFWAIT
CSRM DFLASH (DF(1))	DMU_GP_CSRM_DFWAIT

***Attention**: please check the SRI clock settings for the cyber security real time module (CSRM) from which the applicable wait state settings for the CSRM PFLASH can then be derived. For the CSRM DFLASH this is not an issue because the reference clock for CSRM -DFLASH reading is the FSI clock (same clock as used for HOST DFLASH reading).*

Related information

DFLASH read configuration protection on page 769
Clocks relevant for NVM on page 738

5.3.6.3 PFLASH ECC

Features

The ECC checksum is calculated over 256 data bits and the address bits. The ECC has the following features:

- Correction of 1-bit and 2-bit errors
- Detection of 100% of 1-bit, 2-bit and 3-bit errors
- Detection of >99% of all error vectors in the white noise error model
- Detection of >99% of all-0 and all-1 cases
- Detection of addressing errors

Per safety requirement the error correcting code detects all-0 and all-1 data as uncorrectable errors. Therefore reading from an erased Flash range results in uncorrectable error and also over-programming of Flash ranges with all-1 would create entries with uncorrectable ECC errors.

Please always refer to the device specific safety manual to find out which safety mechanisms of this functional block need to be considered for safety relevant applications.

PFLASH ECC control and error flags

The control and the status of the PFLASH ECC is done by the following PFRWB control and status registers:

- UR_FLASHCON2
- UR_ECCR
- UR_ECCS
- UR_SBABx
- UR_DBABx
- UR_UBABx

For each PFLASH Bank there is a separate set of these registers. The registers can be accessed by CPU; the base address can be found in the system address map at NVM.PFRWB**xy**.UR with:

- **x** = number of the CPU where these PFLASH Bank belongs to
- **y** = 0 for PFLASH Bank 0(A) or **y** = 1 for PFLASH Bank 1(B)

For the offset address of the register please refer to the Registers Overview - PFRWB chapter.

The ECC is automatically generated when programming the Flash. This can be disabled with bit field UR_FLASHCON2.ECCCORDIS. The disabling of the error correction can be selected per Bank as described above.

The ECC is automatically evaluated when reading data. Errors are only reported for 256-bit data blocks for which at least one byte is read by the CPU port.

An internal prefetch with ECC errors (1-bit, 2-bit or uncorrectable) is discarded and is not stored in the prefetch buffer. The ECC errors will not be recorded in status and error address buffer registers. Also a (potential) SMU alarm will not be triggered.

Besides the prefetch the flash interface does not know if the master uses the requested data or discards it (for example due to speculatively fetching code). Therefore speculatively fetched code or data gets an error response in case of a ECC error. This means an error flag setting in the ECC status registers and potentially also an SMU alarm (depending on error number and configuration).

Error reporting and ECC disabling:

- Single-bit error:
  - Is noted in register bit field UR_ECCS.SBER
  - Single Bit Error (SBER) alarm is generated
  - If UR_FLASHCON2.RECDIS = 0_B then the affected address is stored uniquely in the SBAB (see section below), if the SBAB buffer is full an additional safety alarm is generated
  - If UR_FLASHCON2.RECDIS = 1_B then no recording is done and no alarm is generated
  - Data and ECC value are corrected if this is not disabled with UR_FLASHCON2.ECCCORDIS[8)]

[8] Disabling the correction of errors with ECCCORDIS is a test feature. During application run-time the correction must be enabled.

- Double-bit error:
  - Is noted in register bit field UR_ECCS.DBER
  - Double Bit Error (DBER) alarm is generated
  - If UR_FLASHCON2.RECDIS = 0_B then the affected address is stored uniquely in the DBAB (see section below), if the DBAB buffer is full an additional safety alarm is generated
  - If UR_FLASHCON2.RECDIS = 1_B then no recording is done and no alarm is generated
  - Data and ECC value are corrected if this is not disabled UR_FLASHCON2.ECCCORDIS
- Uncorrectable error:
  - Is noted in register bit field UR_ECCS.UCER
  - Causes a bus error
  - If UR_FLASHCON2.RECDIS = 0_B then the affected address is stored uniquely in the UBAB (see section below), if the UBAB buffer is full an additional safety alarm is generated
  - If UR_FLASHCON2.RECDIS = 1_B then no recording is done and no alarm is generated

***Note**: In the UR_ECCS register there are additional status bits available: for example all-0 bit error or accumulated single bit error; please refer to the description of the register for further details.*

Error address buffers SBAB, DBAB and UBAB

If UR_FLASHCON2.RECDIS = 0_B then the affected address is stored uniquely in one of the error address buffers:

Table 386 PFLASH ECC error address buffers

Address buffer	Description
SBABx	address storage for page(s) with correctable single bit errors
DBABx	address storage for page(s) with correctable double bit errors
UBABx	address storage for page(s) with uncorrectable double bit errors

For a description of the address buffers please refer to the register descriptions in the Registers Overview - PFRWB chapter.

For a description of the safety alarms and how to handle these alarms please refer to the safety description: safety mechanism: SM[HW]:NVM:PFLASH_ERROR_MANAGEMENT

Related information

Write page on page 696

5.3.6.4 DFLASH ECC

Features

The ECC is calculated over 64 data bits. Addressing faults (correct data is read from an incorrect address) cannot be detected. The ECC has the following features:

- Correction of 1-bit, 2-bit and 3-bit errors
- Detection of 100% of 1-bit, 2-bit, 3-bit and 4-bit errors
- Detection of >90% of all error vectors in the white noise error model
- All-0 and all-1 data is a valid ECC word

An erased Flash range delivers ECC correct all-0 data. Over-programming with all-1 results in ECC correct all-1 data.

***Attention**: over-programming already existing data in DFLASH is allowed but only to a limited number; please refer to the datasheet parameters ND_PAGE_OVERWR and ND_LOGSEC_BL_PROG.*

DFLASH ECC control and error flags

The control and the status of the DFLASH ECC is done by the following DMU control and status registers:

- HOST DFLASH (DFLASH(0))
  - DMU_GP_HOST_DFECCC
  - DMU_GP_HOST_DFECCS
  - DMU_GP_HOST_DFECCR
  - DMU_GP_HOST_DFECCW
- CSRM DFLASH (DFLASH(1))
  - DMU_GP_CSRM_DFECCC
  - DMU_GP_CSRM_DFECCS
  - DMU_GP_CSRM_DFECCR
  - DMU_GP_CSRM_DFECCW

The registers can be accessed by CPU; the base address can be found in the system address map at NVM.DMU.

For the offset address of the register please refer to the Registers Overview - DMU .

The ECC is automatically generated when programming the Flash. This can be disabled with bit field GP_HOST_DFECCC.ECCCORDIS and GP_CSRM_DFECCC.ECCCORDIS.

The ECC is automatically evaluated when reading data. Errors are only reported for 64-bit data blocks for which at least one byte is read.

Error reporting:

The following ECC error flags are available:

Table 387 DFLASH ECC error flags

Error flag	Description
SBER	Single Bit ECC error
DBER	Double Bit ECC error
TBER	Triple Bit ECC error
UCER	Uncorrectable ECC error
ANER	Any ECC error

In addition for all of these flags also an "accumulated" error flag is available, for example the flag **A**SBER, which reports that at least one single bit error was found in the read accesses since the last reset or clear.

Related information

DFLASH read configuration protection on page 769

5.3.7 Command interfaces

All Flash operations except memory mapped reads are performed with command sequences.

A command sequence is done by a sequence of write accesses to a dedicated memory range of system address segment 15. The DMU has two command interfaces which can be accessed at two different addresses of segment 15:

- Host command interface
- CSRM command interface

Command sequences consist of 1 to 9 command cycles. The command sequence interpreter checks that a command cycle is correct in the current state of command interpretation, else a *Sequence Error (SQER)* is reported.

Triggering an NVM operation using the command sequences takes the DMU into **COMMAND MODE**. The DMU_HCI_STATUS .REQACK is set on the first cycle of a command sequence to acknowledge that a request was received (for CSRM Flash Banks see DMU_CSCI_STATUS.REQACK). During the execution of the command the DMU_HCI_STATUS.BUSYx of the Banks under operation are set (for CSRM Flash Banks see DMU_CSCI_STATUS.BUSYx). In this mode read accesses to a Flash Bank are refused with a *Bus Error* or the ready is suppressed until BUSY clears (see section about "stall function" below).

Upon BUSY assertion the PFI of the CPU invalidates all prefetch and data buffer.

Code that performs any NVM operation on a PFLASH Bank must be executed from an other PFLASH Bank or from SRAM. This is valid for any command executed by the DMU: not only for program or erase commands but also for data integrity check commands like `Verify Erased Logical Sector Range`.

At the end of a flash operation the Flash bank clears the BUSY bit and read accesses are enabled.

The end of the command sequence is signaled by the bit DMU_HCI_STATUS.REQDONE set to 1_B (HOST command interface). Only after this it is possible to apply a new command sequence. For the CSRM command interface the bit DMU_CSCI_STATUS.REQDONE need to be checked.

The command sequence interpreter reports a *sequence error* by setting SQER bit or a *protection error* by setting PROER bit in the command interface error registers DMU_HCI_ERR (for HOST command interface) or DMU_CSCI_ERR (for CSRM command interface). Then the command sequence interpreter is in **ERROR MODE** and a `Reset to Read` command is necessary to reset the command interpreter.

***Note**: In the following chapters the terms "command interface" and "command channel" will be used interchangeably.*

Using CPU stall function

Using the CPU register PFIy_FLASHCON1.STALL = 1_B is not recommended for normal use because the stalled CPU is inoperable. This feature shall only be used under controlled conditions by a Flash loader.

Attention**: **Using the CPU register PFIy_FLASHCON1.STALL = 1_B together with Flash Sleep (on affected Banks) is forbidden as it might lead to device hang-up.

Related information

Command sequences on page 689
Host command interface on page 679
CSRM command interface on page 679

5.3.7.1 Host command interface

- The host command sequence interpreter is accessed by write accesses to the NVM.DMU (Host command sequence interpreter) F808 0000_H - F808 $FFFF_H$ address range in the memory map segment 15
- A on chip bus master may perform command sequences using the host command interface if this is allowed by the access enable protection settings (APU)
- Reports status and errors in DMU_HCI_STATUS and DMU_HCI_ERR
- Own assembly buffer for programming data

Related information

Command interfaces on page 678

5.3.7.2 CSRM command interface

- The CSRM command sequence interpreter is accessed by write accesses to the NVM.DMU (CSRM command sequence interpreter) F80C 0000_H - F80C $FFFF_H$ address range in the memory map segment 15
- A on chip bus master may perform command sequences via the CSRM command interface if this is allowed by the access enable protection settings (APU). Per RESET value these protection settings only allow access by the CSRM CPU
- Reports status and errors in DMU_CSCI_STATUS and DMU_CSCI_ERR
- Own assembly buffer for programming data

Related information

Command interfaces on page 678

5.3.7.3 Concurrent operations

Both command interfaces are independent from each other. They can issue command sequences independently and they have their "own" registers with status-, control- and error-flags.

TC4xx allows a **true** parallel operation on both command interfaces. The following example is a possible concurrent operation scenario:

- Write page on DFLASH(HOST)
- Erase on DFLASH(CSRM)

Attention**: **two parallel NVM operations lead to increased current consumption on the VDDEXT supply. It is possible to reduce the current consumption in field while still taking benefit of this true parallel operation capability by using write operations which split the program pulse into several pulses (see the description in the related information).

Related information

Write settings on page 739

5.3.7.4 Allocation of banks to command interfaces

Each Command Interface allows NVM Operations on a subset of NVM PFLASH and DFLASH **Banks**. The following table shows this allocation.

Table 388 Allocation of NVM banks to the command interfaces

DMU command interface mode	Host command interface	CSRM command interface
Domain Mode	HOST PFLASH PFx HOST DFLASH DF(0)	CSRM PFLASH CSRM DFLASH DF(1)
CSRM_ALL_PFLASH Mode	HOST DFLASH DF(0)	CSRM PFLASH CSRM DFLASH DF(1) HOST PFLASH PF(x)

***Note**: The DEFAULT mode after start-up is domain mode.*

***Note**: A change of the modes can be done by the `Bank Reallocation` command sequence.*

A Bank may only be allocated to one command interface. The modes shown above are the only allowed settings.

Command sequences on allocated memory banks

A NVM operation by a command interface is possible to those Banks which are allocated to that command interface.

The access to the command interface alone is not sufficient. In addition it is necessary that the accen registers for the memory banks are also set to enable the master access. For further details please refer to the Access Protection chapter.

The following sections show graphical representations of the table above. Please note that the access lines from HOST and from CSRM CPUs to the command interfaces show examples for allowed accesses. For an overview of the protection settings and how to change them please refer to the related information mentioned at the end of the chapter.

Domain mode

This is the default mode after start-up. The allocation of the banks to the command interfaces matches with the separation of the domains:

- All HOST memories are allocated to the HOST command interface
- All security memories are allocated to the CSRM command interface

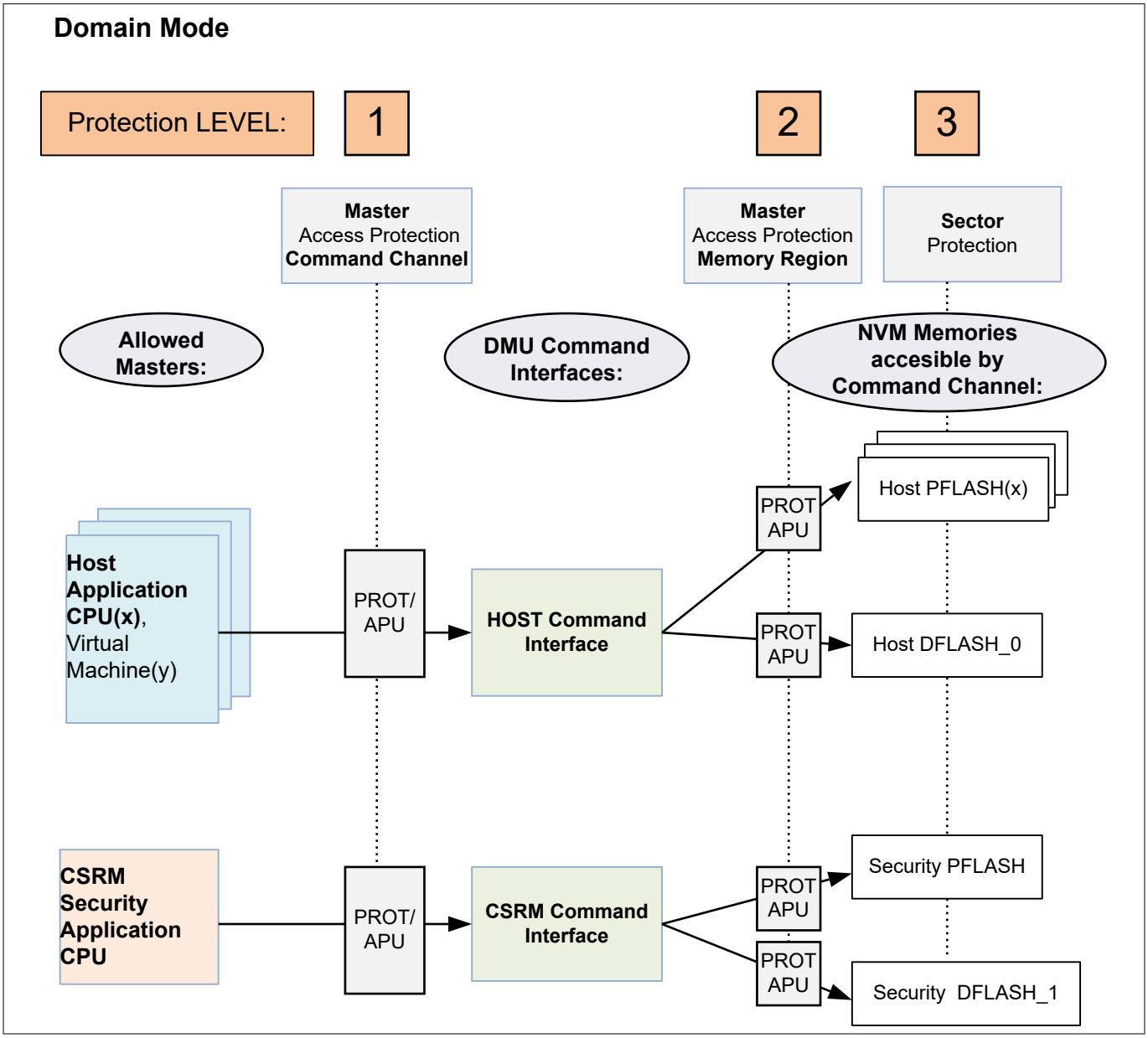


Figure 41 Domain mode

The preceding figure shows the default allocation of memory banks to the command interfaces. The access from different CPUs to the command interfaces is also shown. This CPU → command interface access is not a hardware configuration but is possible to adapt by setting the APU or accen registers.

CSRM_ALL_PFLASH mode

In this mode only the DFLASH(HOST) is allocated to the HOST command interface. All the other Banks are allocated to the CSRM command interface.

This mode is beneficial for concurrent operations on different Bank types. A possible use case is the EEPROM emulation on DFLASH(HOST)) and concurrent updates of PFLASH (HOST) for SOTA purpose. For this use case the security software running on CSRM CPU could do the prog or erase operations on the PFLASH Banks while the EEPROM software may be operated on a HOST CPU and perform prog or erase operations on the DFLASH(HOST) by the HOST command interface.

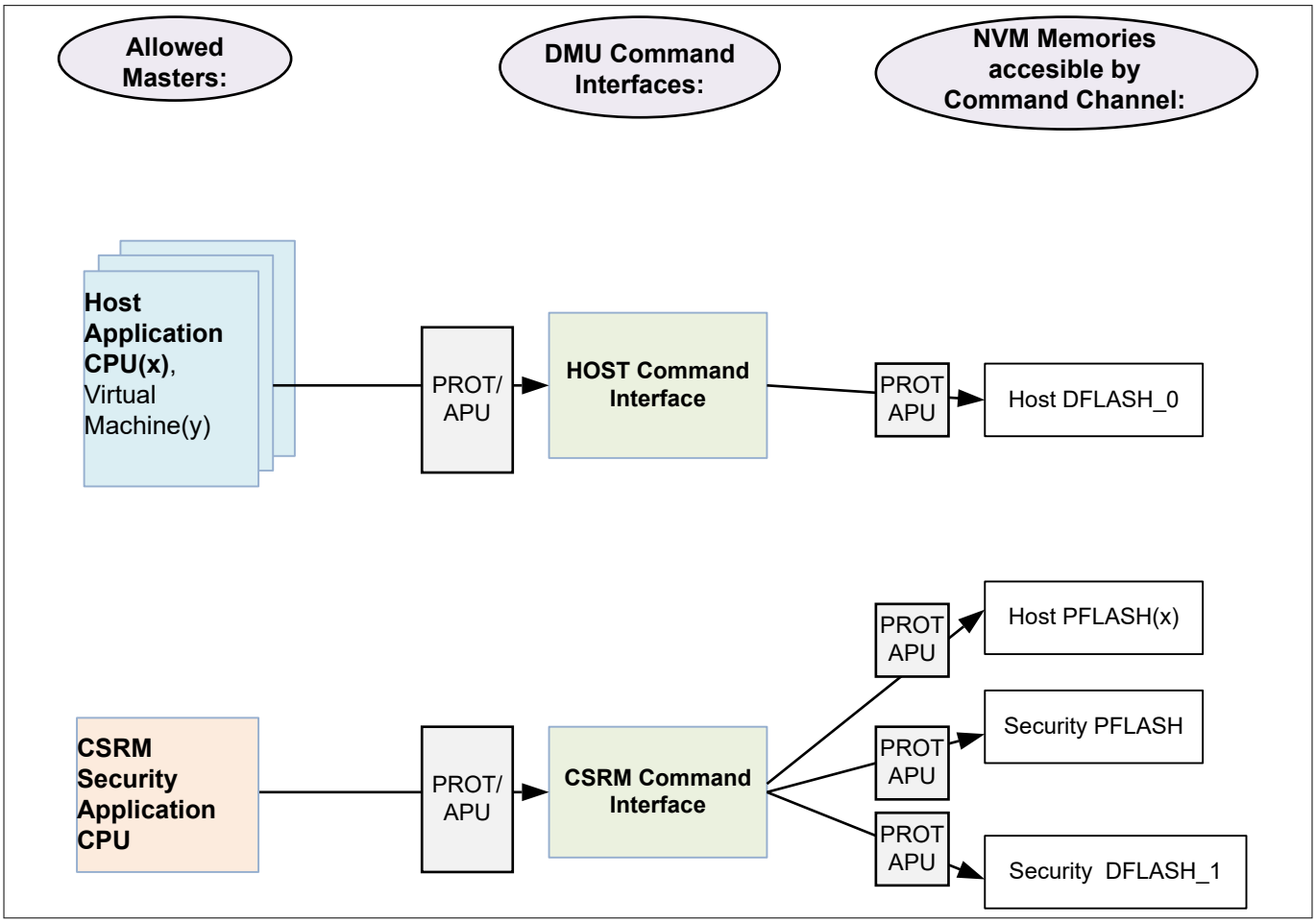


Figure 42 CSRM_ALL_PFLASH mode

Status bits of PFLASH(HOST) banks during the CSRM_ALL_PFLASH mode

If the PFLASH(HOST) Banks are allocated to the CSRM command interface then the status bits for BUSY and SLEEP states for these Banks are updated only in the CSRM status registers (see the following table). Status bits of these Banks in the HOST status registers are not updated.

To enable an interrupt correlated with a NVM operation on the PFLASH (HOST) banks it is also necessary to use the CSCI_INTEN register.

Also any other control setting or error status information for operations on PFLASH(HOST) memories shall be set or checked in the corresponding CSRM control or status registers (for example CSCI_ERR or CSCI_OCONTROL.

Table 389 CSRM command interface registers

Register	Use
DMU_CSCI_STATUS	Status, BUSY indication
DMU_CSCI_ERR	Error flags

CPUcs access to HOST memories

There are two possibilities to grant the CPUcs command access to the HOST memories:

1. CSRM_ALL_PFLASH mode
2. DOMAIN Mode and CPUcs gets access to the HOST command interface

Access to the command interface alone is not sufficient. In addition it is necessary that the accen registers for the PFLASH (HOST) are also set to enable CPUcs access.

Related information

Bank reallocation on page 735
Operation protection overview on page 759

5.3.7.5 Command interface modes

A command interface may be in one of the following states (also called "modes" in other chapters):

- IDLE MODE
- COMMAND MODE
- PAGE MODE
- ERROR MODE

IDLE MODE

During this mode there is no NVM operation ongoing. All NVM Banks may be read (with the exception of Banks which had been sent to SLEEP). A new command sequence may be applied to the command interface.

COMMAND MODE

This mode is started by a command sequence to the NVM. During this mode a NVM operation like write or erase is performed on NVM Bank(s). During the operation on a Bank the BUSY flag of that Bank is set and that Bank cannot be read. The CPU stall setting determines the behavior if a BUSY PFLASH Bank is read. Read access to a BUSY DFLASH Bank always generates a bus error.

PAGE MODE

The command interface must be in PAGE MODE to:

- Load write data in an assembly buffer
- Support a `Write` command to write assembly buffer data in the Flash

The PAGE MODE is started by a `Enter Page Mode` command sequence.

The PAGE MODE is left by any `Write` command sequence; for example `Write Page` command. The PAGE MODE is also left if the subsequent Write command is aborted. The PAGE MODE is also aborted by a `Reset to Read` command.

ERROR MODE

The command interface is in ERROR MODE if a *Sequence Error (SQER)* or a *Protection Error (PROER)*occurs. In this mode the command interface does not accept new commands except `Reset to Read`..

It is necessary to perform a `Reset to Read` command to get the command interface back to IDLE MODE.

OPERATION ERROR MODE

OPERATION ERROR MODE is entered when there is an *Operation Error (OPER)*. In this case the following happens:

- An on-going NVM operation is aborted
- The BUSY flags of bank(s) under operation stay HIGH
- The read of the other banks is not impaired
- All further commands are rejected with *Sequence Error (SQER)*.

The OPERATION ERROR MODE can only be left by a SYSTEM RESET.

Freedom of interference between both command interfaces

Both command interfaces (HOST, CSRM) are independent from each other which means that the one command interface may be in mode or state A while the other command interface may be in mode or state B; for example: HOST command interface may be IDLE while CSRM command interface is in PAGE MODE.

Related information

Error handling on page 685

5.3.7.6 Error handling

This chapter describes:

- Fault conditions
- New state or mode of NVM
- Proposed handling by software

for the following errors:

- *Sequence Error (SQER)*
- *Operation Error (OPER)*
- *Protection Error (PROER)*
- *Erase Verify Error (EVER)*
- *Program Verify Error (PVER)*
- *Original Error (ORIER)*

These errors are reported by the NVM by setting the corresponding bit in the command interface error registers:

- HOST command interface error register DMU_HCI_ERR
- CSRM command interface error register DMU_CSCI_ERR

Sequence error (SQER)

Fault conditions:

- Improper command cycle address or data, that means incorrect command sequence
- New `Enter Page Mode` in **PAGE MODE**
- `Load Page` and not in **PAGE MODE**
- `Load Page` with mixed 32-bit or 64-bit transfers
- `Load Page` with invalid operation code (operation code must be SDTD or SDTW)
- For a 32-bit transfer the first `Load Page` addresses the upper 32-bit word
- Write commands when the assembly buffer contains no data, to few data or too much data was transferred
- Write commands not in **PAGE MODE**
- All commands to unavailable Flash range (for example: Flash range does not physically exist)
- Command sequence with address not aligned to a legal start address (for example page, UCB or logical sector)
- Command sequence not pointing to a PFLASH, DFLASH0(EEPROM), DFLASH0(UCB) or DFLASH1(EEPROM), DFLASH1(UCB) address
- A command sequence to the Configuration Sector (CFS)
- A command sequence to the Erase Counters
- `Write Page Once` targeting DFLASH
- `Write Burst Once` targeting DFLASH
- `Erase Logical Sector Range` or `Verify Erased Logical Sector Range` with range violating the Bank address boundaries
- Any command to a Bank in SLEEP state except a Bank wake-up command
- Unsupported command sequence for ERROR MODE

New state:

The command interface enters **ERROR MODE** and does not accept new commands except `Reset to Read`.

Proposed handling by software:

Usually this bit is only set due to a bug in the software. Therefore in development code the responsible error tracer shall be notified. In production code this error shall not occur.

It is necessary to perform the command `Reset to Read` to get the command interface back to **IDLE MODE**.

Operation error (OPER)

Fault conditions:

The FSI may report an OPER at any time. Possible causes include:

- Double-bit ECC error detected while executing microcode out of FSI SRAM
- Transient event due to alpha-particles or illegal operating conditions
- Permanent error due to a hardware defect

The probability for these faults is very low.

New state:

The Flash operation is aborted and **OPERATION ERROR MODE** is entered.

Proposed handling by software:

The last operation can be determined from the flags in the status registers (for example PROG) flag. In case of an erase operation the affected physical sector must be assumed to be in an invalid state, in case of a program operation only the affected page (or pages in case of `Write Burst` commands. Other physical sectors can still be read. New program or erase commands must not be issued before the next reset.

A system reset must be applied to perform a new Flash startup with initialization of the FSI SRAM and clear the OPER flag. The application must determine from the context which operation failed and react accordingly: it is recommended to erase and rewrite the logical sector which was under operation when the OPER occurred.

Only in case of a permanent defective FSI SRAM or control logic the next operation after system reset will incur again this error.

Protection error (PROER)

Fault conditions:

- Password failure
- Erase or Write to protected sector
- Replace Logical Sector command to a protected sector
- `Erase Logical Sector Range` of UCB with active protection
- Write commands to UCB with active protection
- Verify commands to a protected sector:
  - `Verify Erased Page`, `Verify Erased Wordline`, `Verify Erased Logical Sector Range`
- User commands to a protected sector:
  - `User Margin Count`, `User Vth Count`, `User Content Count`
- During ramp-up if for any single and dual UCB both the ORIG and COPY confirmation codes are **ERRORED**

New state:

The protection violating command is not executed.

The command interface enters **ERROR MODE** and does not accept new commands except `Reset to Read`.

Proposed handling by software:

System integrator to select appropriate further action. It is necessary to perform the command `Reset to Read` to get the command interface back to **IDLE MODE**.

Erase verify error (EVER)

Fault conditions:

This flag is set in one of the following cases:

- `Erase Logical Sector Range` operation was not successful
- Standalone `Verify Erased Logical Sector Range`, `Verify Erased Page`, `Verify Erased Wordline` functions found errors
- `Replace Logical Sector` command did not find a free redundant sector to be used for repair
- `Write Page Once` or `Write Burst Once` functions found that targeted memory range is not erased

New state:

No state change. Just the EVER bit is set.

Proposed handling by software:

This flag can be cleared with the `Clear Status` command.

There are several possible reasons for the failure of an erase operation.

- Permanent or temporal voltage condition below the minimum specification (VDDEXT or VDD Core supply)
- Wrong device configuration
- Hard breakdown of an oxide within the NVM
- Erase in principle functional but the number of weakly erased cells is above a defined fault tolerance threshold level

The operating conditions shall always be checked with regards to voltage and temperature. A stable voltage supply during the whole time of the erase command must be maintained.

It is also recommended to repeat the Erase operation at least one time.

The following advice assumes that a repetition of the erase operation still shows an EVER flag and that the operating conditions and device configuration was correct.

It is possible to "repair" a failing logical sector by using the `Replace Logical Sector` command. For this 2 things are necessary:

- Ensure that there is still a free redundant mini sector for repair
- Identify the address of the failing logical sector

The `Count Free Redundant Sectors` command may be used to check if there are free redundant mini sectors available for a repair.

It is also necessary to identify the failing logical sector address. Usually an erase is done over a wider range: for example up to 256 KB erase size are allowed for Data Flash. It is possible to repeat the erase on smaller regions to make sure that all logical sectors are erased properly and the failing logical sector can be identified.

Once the failing logical sector is identified it is possible to repair this sector by applying the `Replace Logical Sector` command together with the logical sector address.

If the DFLASH EEPROM emulation software is not designed to use the `Replace Logical Sector` command or cannot use it due to whatever reason then the software must be able to "jump" over a logical sector during the operation.

For additional information about a "robust EEPROM emulation" algorithm please refer to the chapter EEPROM emulation with DFLASH.

***Note**:* *It is recommended to clear the EVER flag at the end of the command. Otherwise all following operations — including “sleep” — could trigger an interrupt even when they are successful.*

Program verify error (PVER)

Fault conditions:

This flag is set in one of the following cases:

- `Write Page`, `Write Page Once`, `Write Burst`, or `Write Burst Once` operation was not successful
- Error during the programming of the new redundancy mapping information during the `Replace Logical Sector` command operation
- During the application of the `Erase Logical Sector Range` command on a PFLASH region when it was not possible to write a new erase counter entry

New state:

No state change. Just the PVER bit is set.

Proposed handling by software:

This can be cleared with the `Clear Status` command.

There are several possible reasons for the failure of an program operation.

- Permanent or temporal voltage condition below the minimum specification (VDDEXT or VDD Core supply)

- Wrong device configuration
- Hard breakdown of an oxide within the NVM
- Program in principle functional but the number of weakly programmed cells is above a defined "fault tolerance threshold" level

The operating conditions shall always be checked with regards to voltage and temperature. A stable voltage supply during the whole time of the write command must be maintained.

It is also recommended to repeat the write operation at least one time.

The following advice assumes that a repetition of the write operation still shows a PVER flag and that the operating conditions and device configuration was correct.

All write operations (for example `Write Page`, `Write Page Once`) contain the check of the programmed bits under very tight conditions to have sufficient retention margin for the full operating lifetime. Therefore there could be a *Program Verify Error (PVER)* but a normal read of this data does not show an (ECC) error. But the PVER indicates that the retention of the data may not be sufficient.

***Note**:* *After the write operation the number of failing bits may be read out, see the command sequence description of the `Write Page` command.*

Shall the PVER error flag not disappear after a repetition of the write command then it is recommended to either skip the use of the wordline or repair the logical sector where the error occurred by the `Replace Logical Sector` command. Please refer to the application hints given in the previous section for EVER and the hints given in the chapter EEPROM emulation with DFLASH.

***Note**:* *It is recommended to clear the PVER flag at the end of the command. Otherwise all following operations — including “sleep” — could trigger an interrupt even when they are successful.*

Original error (ORIER)

Fault conditions:

This bit is set by hardware, if during startup, the DMU detects an **ERRORED** value or an uncorrectable ECC error in the ORIG confirmation code of a single UCB (one UCB containing both ORIG and COPY confirmation codes).

Proposed handling:

The start of the MCU is still possible as long as the COPY confirmation code of the UCB is still correct. It is recommended to rewrite this UCB in a stable environment (for example at next inspection).

Related information

Write operation hints on page 739
Command interface modes on page 684
EEPROM emulation with DFLASH on page 792
Erase counters on page 665
Command sequences on page 689
Operation protection overview on page 759
Sector protection on page 763
Write page on page 696
Replace logical sector on page 702

5.3.8 Command sequences

The command sequence interpreter checks for the correctness of request and execution. In case of any error a Sequence Error (SQER) is generated.

The command sequence descriptions use the following nomenclature (symbolic assembly language) STaddr, data. Symbolic representation of a command cycle moving "data" to "addr". The parameter "addr" is defined as follows:

- $CCCC_H$: The "addr" must point into any of the address regions of the Command Sequence Interpreter used. The last 16 address bits must match $CCCC_H$

The parameter "data" can be one of the following attributes:

- **PA**: Absolute start address of the Flash page. It must be aligned to burst size for "Write Burst" or to the page size for "Write Page"
- **WA**: Absolute start address of the Flash word line
- **SA**: Absolute start address of a Flash sector. Allowed are the PFlash sectors S*x* and the DFlash sectors EEPROM*x* and UCB*x*
- **BS**: Bank selection coded as 32-bit vector (refer to the command description to see the allowed combinations):
  - Bit 0: HOST PFLASH0A
  - Bit 1: HOST PFLASH0B
  - Bit 2 : HOST PFLASH1A
  - ..
  - Bit 11: HOST PFLASH5B
  - Bit 12 -15 : reserved - 0
  - Bit 16 : HOST DFLASH (includes UCB)
  - Bit 17: CSRM DFLASH (includes UCB)
  - Bit 18: CSRM PFLASH
  - Bit 19-31 : reserved - 0

  ***Note**: For products with less PFLASH (HOST) Banks, it is necessary to adapt the BS accordingly. Example: a product with 10 PFLASH (HOST) banks would make use of bits 0 to 9 (i.e. HOST PFLASH0A to PFLASH4B) and would have bits 10 - 15 as reserved.*

- **WD64** and **WD32**: 64-bit or 32-bit **write data** to be loaded into the page assembly buffer
- xx**YY**: 8-bit write data as part of a command cycle. Only the byte "YY" is used for command interpretation. The higher order bytes "xx" are ignored
  - xx**5y**: "y" can be 0_H for selecting the PFlash or D_H for DFlash
  - xx**ww**: "ww" defines the number of pages and must not cross a word line border
  - xx**nn**: "nn" defines the number of logical sectors that are erased or verified. This number underlies certain restrictions
- **UC**: Identification of the UCB*x* for which the password checking shall be performed:
  - **HOST command sequence interpreter**
    - xx00H for UCB_RTC_BMHD*n* (n=0 - 3)
    - xx01H for UCB_RTC_USERCFG_ORIG and UCB_RTC_USERCFG_COPY
    - xx02H for UCB_RTC_SWAP_ORIG and UCB_RTC_SWAP_COPY
    - xx03H for UCB_RTC_ECPRIO_ORIG and UCB_RTC_ECPRIO_COPY
    - xx04H for UCB_USER_DATA0
    - xx05H for UCB_USER_DATA1

- xx10H for UCB_RTC_FLASH_ORIG and UCB_RTC_FLASH_COPY to disable global read and write protection for HOST DFLASH and all HOST PFLASHs
- xx20H for UCB_RTC_FLASH_ORIG and UCB_RTC_FLASH_COPY to disable global read protection for HOST DFLASH
- xx21H for UCB_RTC_FLASH_ORIG and UCB_RTC_FLASH_COPY to disable global read protection for HOST PFLASH
- xx30H for UCB_RTC_FLASH_ORIG and UCB_RTC_FLASH_COPY to disable global read protection, and sector specific write protection for HOST PFLASH0A
- xx31H for UCB_RTC_FLASH_ORIG and UCB_RTC_FLASH_COPY to disable global read protection, and sector specific write protection for HOST PFLASH0B
- xx32H for UCB_RTC_FLASH_ORIG and UCB_RTC_FLASH_COPY to disable global read protection, and sector specific write protection for HOST PFLASH1A
- xx33H for UCB_RTC_FLASH_ORIG and UCB_RTC_FLASH_COPY to disable global read protection, and sector specific write protection for HOST PFLASH1B
- xx34H for UCB_RTC_FLASH_ORIG and UCB_RTC_FLASH_COPY to disable global read protection, and sector specific write protection for HOST PFLASH2A
- xx35H for UCB_RTC_FLASH_ORIG and UCB_RTC_FLASH_COPY to disable global read protection, and sector specific write protection for HOST PFLASH2B
- xx36H for UCB_RTC_FLASH_ORIG and UCB_RTC_FLASH_COPY to disable global read protection, and sector specific write protection for HOST PFLASH3A
- xx37H for UCB_RTC_FLASH_ORIG and UCB_RTC_FLASH_COPY to disable global read protection, and sector specific write protection for HOST PFLASH3B
- xx38H for UCB_RTC_FLASH_ORIG and UCB_RTC_FLASH_COPY to disable global read protection, and sector specific write protection for HOST PFLASH4A
- xx39H for UCB_RTC_FLASH_ORIG and UCB_RTC_FLASH_COPY to disable global read protection, and sector specific write protection for HOST PFLASH4B
- xx3AH for UCB_RTC_FLASH_ORIG and UCB_RTC_FLASH_COPY to disable global read protection, and sector specific write protection for HOST PFLASH5A
- xx3BH for UCB_RTC_FLASH_ORIG and UCB_RTC_FLASH_COPY to disable global read protection, and sector specific write protection for HOST PFLASH5B

  ***Note**:* *For products with less PFLASH (HOST) Banks, it is necessary to adapt the UC accordingly. Example: a product with 10 PFLASH (HOST) banks would not make use of UCs xx34H and xx3BH for PFLASH5A and PFLASH5B.*

- **CSRM command sequencer**
  - xx80H for UCB_CS_BMHD*n* (n=0 - 3)
  - xx81H for UCB_CS_USERCFG_ORIG and UCB_CS_USERCFG_COPY
  - xx82H for UCB_CS_SWAP_ORIG and UCB_CS_SWAP_COPY
  - xx83H for UCB_CS_ECPRIO_ORIG and UCB_CS_ECPRIO_COPY
  - xx84H for UCB_CS_USER_DATA0
  - xx85H for UCB_CS_USER_DATA1
  - xx8DH for UCB_DBGCS_ORIGand UCB_DBGCS_COPY
  - xx90H for UCB_CS_FLASH_ORIG and UCB_CS_FLASH_COPY to disable read and write protection for CSRM DFLASH, and to disable sector specific write protection for CSRM PFLASH

- xxA0H for UCB_CS_FLASH_ORIG and UCB_CS_FLASH_COPY to disable read and write protection for CSRM DFLASH
- xxB0H for UCB_CS_FLASH_ORIG and UCB_CS_FLASH_COPY to disable sector specific write protection for CSRM PFLASH

- **PW**x: 32-bit word of a 256-bit password. Note that the entire password consists of 256 bits. During the execution of the command sequence “disable protection”, the password is split into eight transfers of 32 bits each

Several command sequences are supported by the DMU. These are summarized in the following table :

Table 390 Command sequences supported by the DMU

Command sequence		Cycle								
		1	2	3	4	5	6	7	8	9
Reset to Read	Addr	.5554								
	Data	xxF0								
Enter Page Mode	Addr	.5554								
	Data	xx5y								
Load Page (32 bit)	Addr	.55F4								
	Data	WD32								
Load Page (64 bit)	Addr	.55F0								
	Data	WD64								
Write Page	Addr	.AA50	.AA58	.AAA8	.AAA8					
	Data	PA	.xx00	.xxA0	.xxAA					
Write Page Once	Addr	.AA50	.AA58	.AAA8	.AAA8					
	Data	PA	.xx00	.xxA0	.xxA8					
Write Burst	Addr	.AA50	.AA58	.AAA8	.AAA8					
	Data	PA	.xx00	.xxA0	.xxA6					
Write Burst Once	Addr	.AA50	.AA58	.AAA8	.AAA8					
	Data	PA	.xx00	.xxA0	.xxA4					
Count Free Redundant Sectors	Addr	.AAC0								
	Data	BS								
Replace Logical Sector	Addr	.AA50	.AA58	.AAA8	.AAA8					
	Data	PA	.xx00	.xxA0	.xxAC					
Verify Erased Page	Addr	.AA50	.AA58	.AAA8	.AAA8					
	Data	PA	.xx00	.xx80	.xx56					
Verify Erased WL	Addr	.AA50	.AA58	.AAA8	.AAA8					
	Data	WA	.xx00	.xx80	.xx58					
Verify Erased Logical Sector Range	Addr	.AA50	.AA58	.AAA8	.AAA8					
	Data	SA	.xxnn	.xx80	.xx5F					

(table continues...)

Table 390 (continued) Command sequences supported by the DMU

Command sequence		Cycle								
		1	2	3	4	5	6	7	8	9
Erase Logical Sector Range	Addr	.AA50	.AA58	.AAA8	.AAA8					
	Data	SA	.xxnn	.xx80	.xx50					
User Margin Count	Addr	.AA50	.AA58	.AAA8	.AAA8					
	Data	WA	.xx00	.xx60	.xx11					
User Vth Count	Addr	.AA50	.AA58	.AAA8	.AAA8					
	Data	WA	.xx00	.xx60	.xx12					
User Content Count	Addr	.AA50	.AA58	.AAA8	.AAA8					
	Data	WA	.xx00	.xx60	.xx14					
Disable Protection	Addr	.553C	.553C	.553C	.553C	.553C	.553C	.553C	.553C	.553C
	Data	UC	PW0	PW1	PW2	PW3	PW4	PW5	PW6	PW7
Resume Protection	Addr	.5554								
	Data	.xxF5								
Clear Status	Addr	.5554								
	Data	.xxFA								
Bank Sleep	Addr	.AA00								
	Data	BS								
Bank Wake-up	Addr	.AAF0								
	Data	BS								
Bank Reallocation	Addr	.AA90								
	Data	BS								

***Note**: The parameters PA, WA and SA shall be an address of the non-cached address range.*

Related information

Command interfaces on page 678
Error handling on page 685
PFLASH global read protection on page 767
DFLASH global read protection on page 771

5.3.8.1 Reset to read

Calling

- ST 5554_H, xx$F0_H$

Function

This function resets the addressed command sequence interpreter to its initial state (i.e. the next command cycle must be the 1st cycle of a sequence). A **PAGE MODE** is aborted.

This command is the only one that is accepted without *Sequence Error (SQER)* when the command sequence interpreter has already received command cycles of a different sequence. Thus `Reset to Read` can cancel every command sequence before its last command cycle has been received.

A system reset will re-initialize the NVM control circuits.

Host Command Sequence Interpreter

Host command sequence interpreter is reset. The following flags are cleared:

- Command sequence error flag: DMU_HCI_ERR.SQER
- Protection error flag: DMU_HCI_ERR.PROER
- PAGE MODE flags: DMU_HCI_STATUS.DFPAGE and DMU_HCI_STATUS.PFPAGE
- Request acknowledgment : DMU_HCI_STATUS.REQACK and DMU_HCI_STATUS.REQDONE

CSRM Command Sequence Interpreter

CSRM command sequence interpreter is reset. The following flags are cleared:

- Command sequence error flag: DMU_CSCI_ERR.SQER
- Protection error flag: DMU_CSCI_ERR.PROER
- PAGE MODE flags: DMU_CSCI_STATUS.DFPAGE and DMU_CSCI_STATUS.PFPAGE
- Request acknowledgment : DMU_CSCI_STATUS.REQACK and DMU_CSCI_STATUS.REQDONE

5.3.8.2 Enter page mode

Calling

- ST 5554_H, xx5**y**$_H$

Function

The PFLASH or the DFLASH assembly buffer enter **PAGE MODE** selected by the parameter "**y**":

Table 391 Page Mode memory selection

y	Memory which enters Page Mode
0_H	PFLASH
D_H	DFLASH

The write pointer of the page assembly buffer is set to 0, its previous content is maintained.

If a new `Enter Page Mode` command sequence is received while any Flash is already in **PAGE MODE**, then *Sequence Error (SQER)* is set.

PAGE MODE is exited when the next `Write Page` or `Write Burst` command sequence terminates.

If the DMU is in **ERROR MODE** then the command fails with a *Sequence Error (SQER)*.

Host Command Sequence Interpreter

PAGE MODE is signaled by the flags DMU_HCI_STATUS.DFPAGE and DMU_HCI_STATUS.PFPAGE for DFLASH and PFLASH respectively.

CSRM Command Sequence Interpreter

PAGE MODE is signaled by the flags DMU_CSCI_STATUS.DFPAGE and DMU_CSCI_STATUS.PFPAGE for DFLASH and PFLASH respectively. .

5.3.8.3 Load page

Calling

- ST $55F0_H$, **WD64** (for 64-bit transfers)
- ST $55F4_H$, **WD32** (for 32-bit transfers)

Function

Loads the data **WDxx** into the page assembly buffer of the chosen HOST or CSRM channel and increments the write pointer to the next position.

All **WDxx** transfers for one page must have the same width (either all 32-bit or all 64-bit). Else the transfer is refused with *Sequence Error (SQER)*.

The following table shows the number of necessary `Load Page` commands before a `Write` operation depending on Page or Burst mode and PFLASH or DFLASH memory.

Table 392 Number of `Load Page` commands

Command	Data [Byte]	Number of `Load Page` (32 Bit)	Number of `Load Page` (64 Bit)
PFLASH `Write Page`	32	8	4
PFLASH `Write Burst`	512	128	64
DFLASH `Write Page`	8	2	1

(table continues...)

Table 392 (continued) Number of `Load Page` commands

Command	Data [Byte]	Number of `Load Page` (32 Bit)	Number of `Load Page` (64 Bit)
DFLASH `Write Burst`	32	8	4

***Attention**: for 32-bit transfers always use address $55F4_H$, this is different to the command specification in previous product generations.*

If the number of `Load Page` data transfers does not result in the correct data amount for a write operation as specified in the table above, then the following `Write Page` or `Write Burst` command will not execute and show a *Sequence Error (SQER)*.

If `Load Page` is called more often than allowed for the required data, then the overflow data is discarded and **PAGE MODE** is not left. This overflow is reported by the following `Write Page` or `Write Burst` command with *Sequence Error (SQER)* and the write operation to Flash will not be executed (example for this error type: data for Burst is written but a Write Page command is requested)

If the DMU is in ERROR MODE then the command fails with an *Sequence Error (SQER)*.

5.3.8.4 Write page

Calling

- ST $AA50_H$, **PA**
- ST $AA58_H$, $xx00_H$
- ST $AAA8_H$, $xxA0_H$
- ST $AAA8_H$, $xxAA_H$

Function

This function starts the programming process for one page with the data transferred previously by `Load Page` commands. Upon executing the command, **PAGE MODE** is exited (indicated by clearing the corresponding PAGE flag) and the BUSY flag of the bank is set.

This command is refused with *Sequence Error (SQER)* when the addressed Flash is not in **PAGE MODE**.

Sequence Error (SQER) is also issued when **PA** addresses an unavailable Flash range or when **PA** does not point to a correct page start address.

If after `Enter Page Mode` no data, too few data or too much data was transferred to the assembly buffer with `Load Page` then the execution of `Write Page` fails, no data is programmed and *Sequence Error (SQER)* is set.

When the page **PA** is located in a sector with active write protection or the Flash module has an active global read protection the execution fails and *Protection Error (PROER)* is set.

When the programming process incurs an error the flag *Program Verify Error (PVER)* is set (both: '0' and '1' bits are verified). The flag is set when the number of incorrect bits is bigger than a memory region dependent threshold. This threshold is of course below the error correction capability of the memory region.

The same applies to the CSRM command sequence interpreter which maintains its own assembly buffer. After programming to the CSRM sectors the assembly buffer is automatically erased.

If the DMU is in **ERROR MODE** then the command fails with a *Sequence Error (SQER)*.

Additional information

After the execution of the command there is additional information about the number of failing bits ('0' and '1') available. This is described in the following table.

Table 393 Additional information after write page execution

NVM part	Register	Range	Description
FSI	COMM0	0 .. 255	Returns the number of fails detected during program verify. Both ('0' and '1' bits are checked and the value is the sum of the failing bits). This result is reported independently from the *Program Verify Error (PVER)* and may be not equal 0 even in case that *PVER* = 0_B.

The maximum allowed number of failing bits (fault tolerance) for the *Program Verify Error (PVER)* is at least one bit less than the number of bits which can be corrected by the error correcting circuitry. This means that with a *Program Verify Error (PVER)* = 1_B there are two possibilities:

1. number of failing bits is lower or equal the number of bits which can be corrected
2. number of failing bits is higher than the number of bits which can be corrected

By using the failcount available in the COMM0 register it is possible to distinguish the two cases. This information can be used by the software to decide on further appropriate actions depending on the type of data and the memory region (for example: waive of error, write data again, erase and write data again, apply replace logical sector command,..),

Table 394 PFLASH: Examples for fault tolerant PVER assertions

Failing Bits (COMM0)	Fault Tolerance	=> PVER	ECC correctable ?
1	1	0	YES
2	1	1	YES
≥ 3	1	1	NO

Notes for safety critical memory regions

Built in fault tolerance of the Write page command is necessary to support fail operational usage in case of few single bit hard errors. Single bit errors up to a certain threshold may be tolerated. The readout of the COMM0 register provides an immediate check of failing bits. The readout of the whole PFLASH memory region together with the check of the error buffers SBABx, DBABx and UBAB (see related information) allows to monitor the overall state of the memory and perform appropriate action if necessary.

Table 395 DFLASH: Examples for fault tolerant PVER assertions in case of fault tolerance = 1

Failing Bits (COMM0)	Fault Tolerance	=> PVER	ECC correctable ?
1	1	0	YES
2	1	1	YES
3	1	1	YES
≥ 4	1	1	NO

FSI_COMMx register addresses

There are separate FSI_COMMx registers for the HOST and for the CSRM command interface:

- FSI_HOST.COMMx
- FSI_CSRM.COMMx

These COMMx registers do not belong to the standard DMU register region. To access those registers please refer to:

Table 396 FSI COMMx registers

Address	Chapter
Register Base Address FSI_HOST.COMM0	System address map
Register Base Address FSI_CSRM.COMM0	
Register offset addresses for COMMx Registers Note: The offset addresses for FSI_HOST.COMMx and for FSI_CSRM.COMMx are the same	Register Overview - FSI

Related information

Write operation hints on page 739
Error handling on page 685
Write sequence on page 741
PFLASH ECC on page 674

5.3.8.5 Write page once

Calling

- ST $AA50_H$, **PA**
- ST $AA58_H$, $xx00_H$
- ST $AAA8_H$, $xxA0_H$
- ST $AAA8_H$, $xxA8_H$

Function

This function starts the programming process for one page as the normal `Write Page` does. But before programming it checks if the page is erased. If the page is not erased (allowing one correctable error) the command fails with *Erase Verify Error (EVER)*.

When the programming itself incurs an error the flag *Program Verify Error (PVER)* is set.

The `Write Page Once` command is only possible on PFLASH Banks. When applied to DFLASH the command fails with *Sequence Error (SQER)*.

Execution time of the write by the `Write Page Once` command is longer then a normal write command (the adder is defined by the datasheet parameter t_{ADD}).

On sectors with “write-once” protection only the `Write Page Once` and the `Write Burst Once` commands are accepted by DMU.

Other than the specific function features mentioned above the same features and constraints as mentioned for `Write Page` apply.

Related information

Write operation hints on page 739

5.3.8.6 Write burst

Calling

- ST $AA50_H$, **PA**
- ST $AA58_H$, $xx00_H$
- ST $AAA8_H$, $xxA0_H$
- ST $AAA8_H$, $xxA6_H$

Function

This function starts the programming process for an aligned group of pages. The programming process is more efficient than for a single page, achieving a significantly higher throughput (see data sheet).

In the PFLASH 16 pages (512bytes) are programmed and in DFLASH 4 pages (32bytes). The pages are programmed one after the other with increasing addresses.

For further details see the `Write Page` command description.

A *Sequence Error (SQER)* is returned and the execution fails as well if:

- The **PA** is not aligned to a burst address (512bytes PFLASH and 32bytes DFLASH)

Host command sequence interpreter

For disabled ECC generation:

- If DMU_GP_HOST_PFECCW.ENCDIS = '11_B' then the user must not perform a Write Burst to PFLASH
- If DMU_GP_HOST_DFECCW.ENCDIS = '11_B' then the user must not perform a Write Burst to DFLASH

CSRM command sequence interpreter

For disabled ECC generation:

- If DMU_GP_CSRM_PFECCW.ENCDIS = '11_B' then the user must not perform a Write Burst to PFLASH
- If DMU_GP_CSRM_DFECCW.ENCDIS = '11_B' then the user must not perform a Write Burst to DFLASH

Related information

Write operation hints on page 739

5.3.8.7 Write burst once

Calling

- ST $AA50_H$, **PA**
- ST $AA58_H$, $xx00_H$
- ST $AAA8_H$, $xxA0_H$
- ST $AAA8_H$, $xxA4_H$

Function

This function starts the programming process for an aligned group of pages as the normal `Write Burst` does. But before programming it checks if all the target pages are erased. If one page is not erased (allowing one correctable error) the command fails with *Erase Verify Error (EVER)*.

When the programming itself incurs an error the flag *Program Verify Error (PVER)* is set.

The `Write Burst Once` command is only possible on PFLASH Banks. When applied to DFLASH the command fails with *Sequence Error (SQER)*.

Execution time of the write by the `Write Burst Once` command is longer then a normal `Write Burst` command (adder by number of pages x t_{ADD}).

On sectors with “write-once” protection only the `Write Page Once` and the `Write Burst Once` commands are accepted by DMU.

For further details see `Write Burst`.

Host Command Sequence Interpreter

For disabled ECC generation:

- If DMU_GP_HOST_PFECCW.ENCDIS = ‘11_B’ then the user must not perform a Write Burst to PFLASH
- If DMU_GP_HOST_DFECCW.ENCDIS = ‘11_B’ then the user must not perform a Write Burst to DFLASH

HSM Command Sequence Interpreter

For disabled ECC generation:

- If DMU_GP_CSRM_PFECCW.ENCDIS = ‘11_B’ then the user must not perform a Write Burst to PFLASH
- If DMU_GP_CSRM_DFECCW.ENCDIS = ‘11_B’ then the user must not perform a Write Burst to DFLASH

Related information

Write operation hints on page 739

5.3.8.8 Count free redundant sectors

Calling

- ST $AAC0_H$, **BS**

Use Case

This command sequence can be used to retrieve the number of free redundant sector in a Bank before executing a `Replace Logical Sector` to replace failing logical sectors.

Function

The argument "**BS**" is used to select the Bank in which the number of free redundant sectors shall be counted.

Only one bank at a time is accepted, other values for **BS** will fail with a *Sequence Error (SQER)*.

The number of free redundant sectors will be returned in the FSI HOST or FSI CSRM COMM0 register. This number will be comprised between 0 and the number of maximum available redundant sectors described in the following tables :

Table 397 PFLASH maximum number of available redundant sectors

PFLASH Bank size	Maximum number of free redundant sectors
1MB	3
2MB	7

Table 398 DFLASH maximum number of available redundant sectors

DFLASH Bank size	Maximum number of free redundant sectors
128kB	7
0.5MB	15
1MB	31

If the `Count Free Redundant Sectors` returns 0, it is useless to execute a `Replace Logical Sector` in this Bank as no redundant sectors are available anymore and thus the device cannot be repaired anymore.

FSI_COMMx register addresses

There are separate FSI_COMMx registers for the HOST and for the CSRM command interface:

- FSI_HOST.COMMx
- FSI_CSRM.COMMx

These COMMx registers do not belong to the standard DMU register region. To access those registers please refer to:

Table 399 FSI COMMx registers

Address	Chapter
Register Base Address FSI_HOST.COMM0	System address map
Register Base Address FSI_CSRM.COMM0	
Register offset addresses for COMMx Registers Note: The offset addresses for FSI_HOST.COMMx and for FSI_CSRM.COMMx are the same	Register Overview - FSI

5.3.8.9 Replace logical sector

Calling

- ST $AA50_H$, **PA**
- ST $AA58_H$, $xx00_H$
- ST $AAA8_H$, $xxA0_H$
- ST $AAA8_H$, $xxAC_H$

Use case

`Replace Logical Sector` is a command sequence that enables the replacement of a logical sector with hard fails with an available redundant logical sector. It can be used if a sector fails during an erase or program.

The AURIX™ TC4xx allows the replacements of logical sectors in PFLASH and in DFLASH Banks.

Every Bank has a certain number of redundant logical sectors. This number depends on the type (PFLASH, DFLASH) and on the size (2 MB, 1 MB, … 128 KB.). The redundant logical sectors are available both for production test, and for the application. Depending on the number of redundant logical sectors used for production test there may be a reduced number of sectors free for repair by the application. The number of already used sectors may be different for different Banks. In worst case there are no more free redundant sectors available. The probability for this is low and might happen rather for smaller Banks than for large Banks. The number of available free redundant sectors can be retrieved by using the `Count Free Redundant Sectors` command.

A failure in a logical sector that has already been replaced cannot be repaired. If the application software uses Replace Logical Sector on a sector that has already been replaced, no error will be reported, but the replacement will not occur, and a redundant sector will be consumed.

After the successful execution of the `Replace Logical Sector` command the previously failing logical sector is mapped to a free redundant functional logical sector which is accessible at the same address. In addition the mapping information of the replacement is programmed into a non volatile memory region so that also at the next reset the mapping information is recovered.

If the DMU is in **ERROR MODE** then the command fails with an *Sequence Error (SQER)*

Function

The argument “**PA**” is used to select a page address aligned to the start of the logical sector (in PFLASH or in DFLASH) to be re-mapped.

Status and error flags

The *Sequence Error (SQER)* is set when the **PA** is not aligned to the start address of a logical sector page address or when **PA** addresses an unavailable Flash range. Logical sectors of the following Flash ranges cannot be repaired:

- UCB
- Erase Counter

The *Protection Error (PROER)* is set when there is an attempt to re-map a logical sector with enabled Write Protection or when the master requesting the command is not allowed to do so.

During the execution of the command the BUSY bit of the Bank under operation is set.

An *Erase Verify Error (EVER)* is generated if there is no free redundant logical sector available in that Bank anymore.

A *Program Verify Error (PVER)* is generated if there is a programming error during the programming of the redundancy information.

Related information

Error handling on page 685

5.3.8.10 Verify erased page

Calling

- ST $AA50_H$, **PA**
- ST $AA58_H$, $xx00_H$
- ST $AAA8_H$, $xx80_H$
- ST $AAA8_H$, $xx56_H$

Function

This command verifies if one page addressed by **PA** is correctly erased, i.e. contain 0 data and ECC bits. Upon executing the command, the BUSY flag of the corresponding bank is set.

As the PFLASH has Safety ECC, the PFLASH cannot be checked for “0” content by direct reads.

The *Erase Verify Error (EVER)* is set in case of an erase verification error.

Sequence error

A *Sequence Error (SQER)* is returned and the execution fails for the following conditions:

- The **PA** is not aligned to a page address
- The DMU is in **ERROR MODE**

5.3.8.11 Verify erased wordline

Calling

- ST $AA50_H$, **WA**
- ST $AA58_H$, $xx00_H$
- ST $AAA8_H$, $xx80_H$
- ST $AAA8_H$, $xx58_H$

Function

This command verifies if one wordline addressed by **WA** is correctly erased, i.e. contain 0 data and ECC bits. Upon executing the command, the BUSY flag of the corresponding bank is set.

As the PFLASH has Safety ECC, the PFLASH cannot be checked for “0” content by direct reads.

The *Erase Verify Error (EVER)* is set in case of an erase verification error.

Sequence error

A *Sequence Error (SQER)* is returned and the execution fails for the following conditions:

- The **WA** is not aligned to a wordline address
- The DMU is in **ERROR MODE**

5.3.8.12 Verify erased logical sector range

Calling

- ST $AA50_H$, **SA**
- ST $AA58_H$, xx**nn**$_H$
- ST $AAA8_H$, $xx80_H$
- ST $AAA8_H$, $xx5F_H$

Function

This command verifies if **nn** sectors starting at the sector addressed by **SA** are correctly erased, i.e. contain 0 data and ECC bits. Upon executing the command, the BUSY flag of the corresponding bank is set.

As the PFLASH has Safety ECC, the PFLASH cannot be checked for “0” content by direct reads.

The *Erase Verify Error (EVER)* is set in case of an erase verification error.

A *Sequence Error (SQER)* is returned and the execution fails for the following conditions:

- The **SA** is not within the logical sector range and not page aligned or it points to an unavailable sector
- For DFLASH, if the range of logical sectors exceeds 256 KByte addressable memory
- For PFLASH, if the range of logical sectors exceeds 512 KByte
- For UCB, if the range of logical sectors exceeds 1
- If the range of logical sector is not contained in a bank
- The DMU is in **ERROR MODE**

A *Protection Error (PROER)* is returned and the execution fails for the following conditions:

- If **SA** or any of the following (**nn** - 1) logical sectors has an active write protection
- If **SA** or any of the following (**nn** - 1) logical sectors belong to a memory region where the APU does not allow access to the master requesting the Erase
- If the Erase request is given to a command interface where the APU does not allow access to the master requesting the Erase
- The Flash module has an active global read protection set

Additional Information in case of erase verify error

In case of *Erase Verify Error (EVER)* there is additional information available. This is described in the following table.

Table 400 Additional information in case of erase verify error

NVM part	Register	Range	Description
FSI	COMM1	0 .. 255	Offset address of first failing logical sector within the sector range: 0: 1st logical sector has erase verify error 1: 2nd logical sector has erase verify error ...

FSI_COMMx register addresses

There are separate FSI_COMMx registers for the HOST and for the CSRM command interface:

- FSI_HOST.COMMx
- FSI_CSRM.COMMx

These COMMx registers do not belong to the standard DMU register region. To access those registers please refer to:

Table 401 FSI COMMx registers

Address	Chapter
Register Base Address FSI_HOST.COMM0	System address map
Register Base Address FSI_CSRM.COMM0	
Register offset addresses for COMMx Registers Note: The offset addresses for FSI_HOST.COMMx and for FSI_CSRM.COMMx are the same	Register Overview - FSI

5.3.8.13 Erase logical sector range

Calling

- ST $AA50_H$, **SA**
- ST $AA58_H$, xx**nn**$_H$
- ST $AAA8_H$, $xx80_H$
- ST $AAA8_H$, $xx50_H$

Function

This command erases **nn** logical sectors starting at the sector addressed by **SA**. Upon executing the command, the BUSY flag of the corresponding bank is set.

A *Sequence Error (SQER)* is returned and the execution fails for the following conditions:

- For PFLASH, if the range of logical sectors exceeds 512 KByte
- For DFLASH, if the range of logical sectors exceeds 256 KByte addressable memory
- For UCB, if the range of logical sectors exceeds 1
- If the range of logical sector is not contained in a bank
- If **SA** is not within the logical sector range and not page aligned
- If **SA** aligns to an unavailable sector
- If **nn** logical sectors is set to 0
- If the DMU is in **ERROR MODE**

A *Protection Error (PROER)* is returned and the execution fails for the following conditions:

- If **SA** or any of the following (**nn** - 1) logical sectors has an active write protection
- If **SA** or any of the following (**nn** - 1) logical sectors belong to a memory region where the APU does not allow access to the master requesting the Erase
- If the erase request is given to a command interface where the APU does not allow access to the master requesting the erase
- The Flash module has an active global read protection set

The *Erase Verify Error (EVER)* is set to indicate an error during the erase process.

The *Program Verify Error (PVER)* is set when the erase counter programming incurs an error (only for PFLASH erase).

***Note**: The duration of an erase command can be much shorter than documented in the datasheet when the range is already erased or partly erased.*

Additional Information in case of erase verify error

In case of *Erase Verify Error (EVER)* there is additional information available. This is described in the following table.

Table 402 Additional information in case of erase verify error

NVM part	Register	Range	Description
FSI	COMM0	0 .. 255	Number of failing logical sectors
	COMM1	0 .. 255	Offset address of first failing logical sector within the sector range: 0: 1st logical sector has erase verify error 1: 2nd logical sector has erase verify error ...

FSI_COMMx register addresses

There are separate FSI_COMMx registers for the HOST and for the CSRM command interface:

- FSI_HOST.COMMx
- FSI_CSRM.COMMx

These COMMx registers do not belong to the standard DMU register region. To access those registers please refer to:

Table 403 FSI COMMx registers

Address	Chapter
Register Base Address FSI_HOST.COMM0	System address map
Register Base Address FSI_CSRM.COMM0	
Register offset addresses for COMMx Registers Note: The offset addresses for FSI_HOST.COMMx and for FSI_CSRM.COMMx are the same	Register Overview - FSI

Related information

Erase operation hints on page 743

5.3.8.14 User margin count

Calling

- ST $AA50_H$, **WA**
- ST $AA58_H$, $xx00_H$
- ST $AAA8_H$, $xx60_H$
- ST $AAA8_H$, $xx11_H$

Function

Attention**: **This command is mainly for characterization and analysis purpose.

With this command the distribution of NVM cell currents can be investigated. The scope is either a single page or a complete wordline; this can be selected by the DMU_HCI_OCONTROL.UCG (or DMU_CSCI_OCONTROL.UCG for CSRM command channel). Depending on the scope the argument **WA** must then be used to select a page or wordline aligned address.

The result of this function is the number N_dv of logic 1 bits in the selected wordline or page which have less cell current than the selected reference current. Calling this function with different reference currents allows calculation of the cell current distribution (please refer to the 2 figures below).

The reference current can be specified by setting an **REF_CURRENT_OFFSET** by bits [6:0] of the FSI_COMM0 register (for register address see section at the end of chapter).

The actual reference current used within the function is calculated using the following formulas:

- PFLASH: reference_current_pflash:= default_current_pflash + **REF_CURRENT_OFFSET**
- DFLASH: reference_current _dflash := default_current_dflash + **REF_CURRENT_OFFSET**

Attention**: **Default values base on characterization and qualification. Please contact your local Infineon representative to check for latest values.

The values for **REF_CURRENT_OFFSET**are used as signed integer (two's complement) to either increase or decrease the default current values; please refer to the following table for reference:

Table 404 Setting table for reference current

REF_CURRENT_OFFSET		Offset Value
hex	bin	[µA]
3F	0111111	15,75
...	...	...
9	0001001	2,25
8	0001000	2
7	0000111	1,75
6	0000110	1,5
5	0000101	1,25
4	0000100	1
3	0000011	0,75
2	0000010	0,5
1	0000001	0,25
0	0000000	0
7F	1111111	-0,25
7E	1111110	-0,5
7D	1111101	-0,75
7C	1111100	-1
7B	1111011	-1,25
7A	1111010	-1,5
79	1111001	-1,75
...	...	...
40	1000000	-16

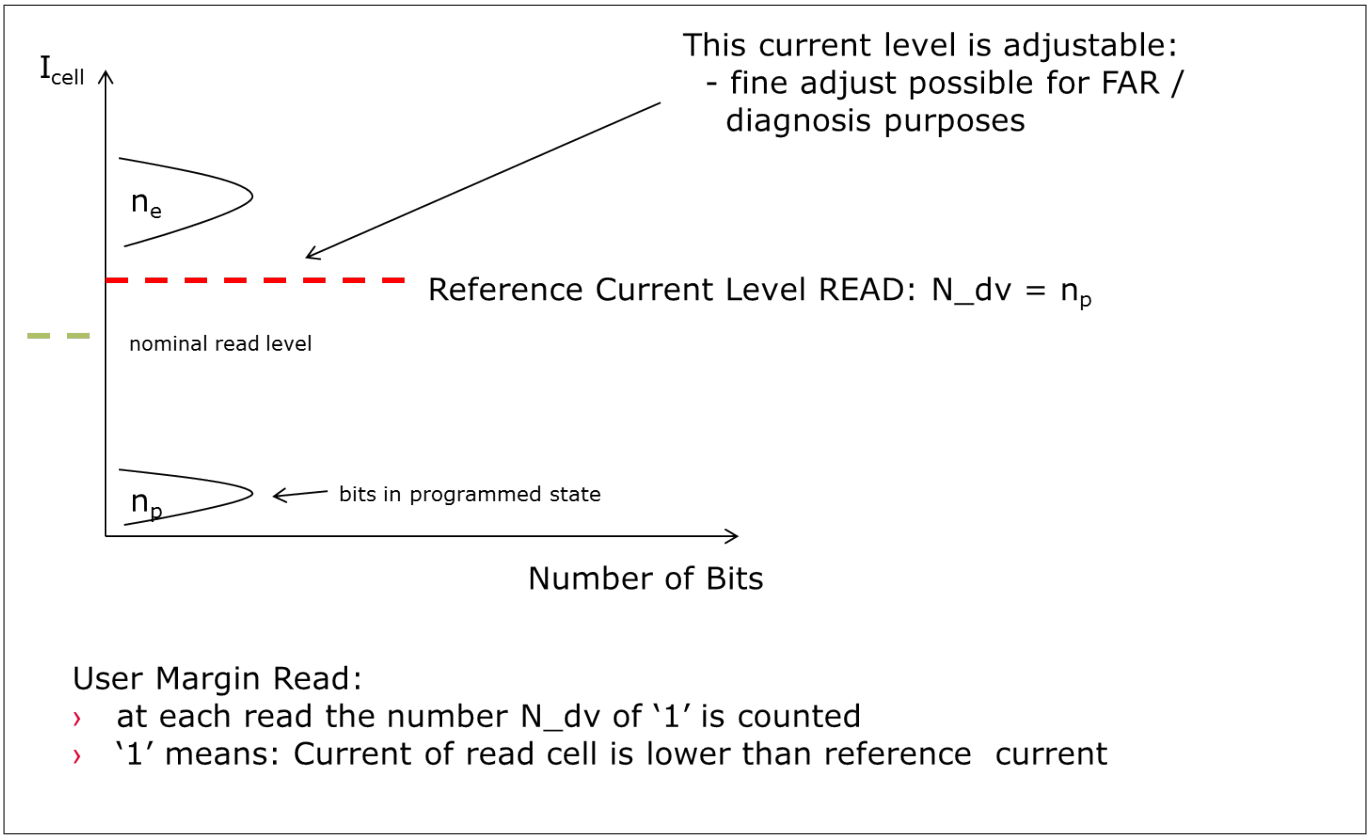


Figure 43 Basic principle of user margin count

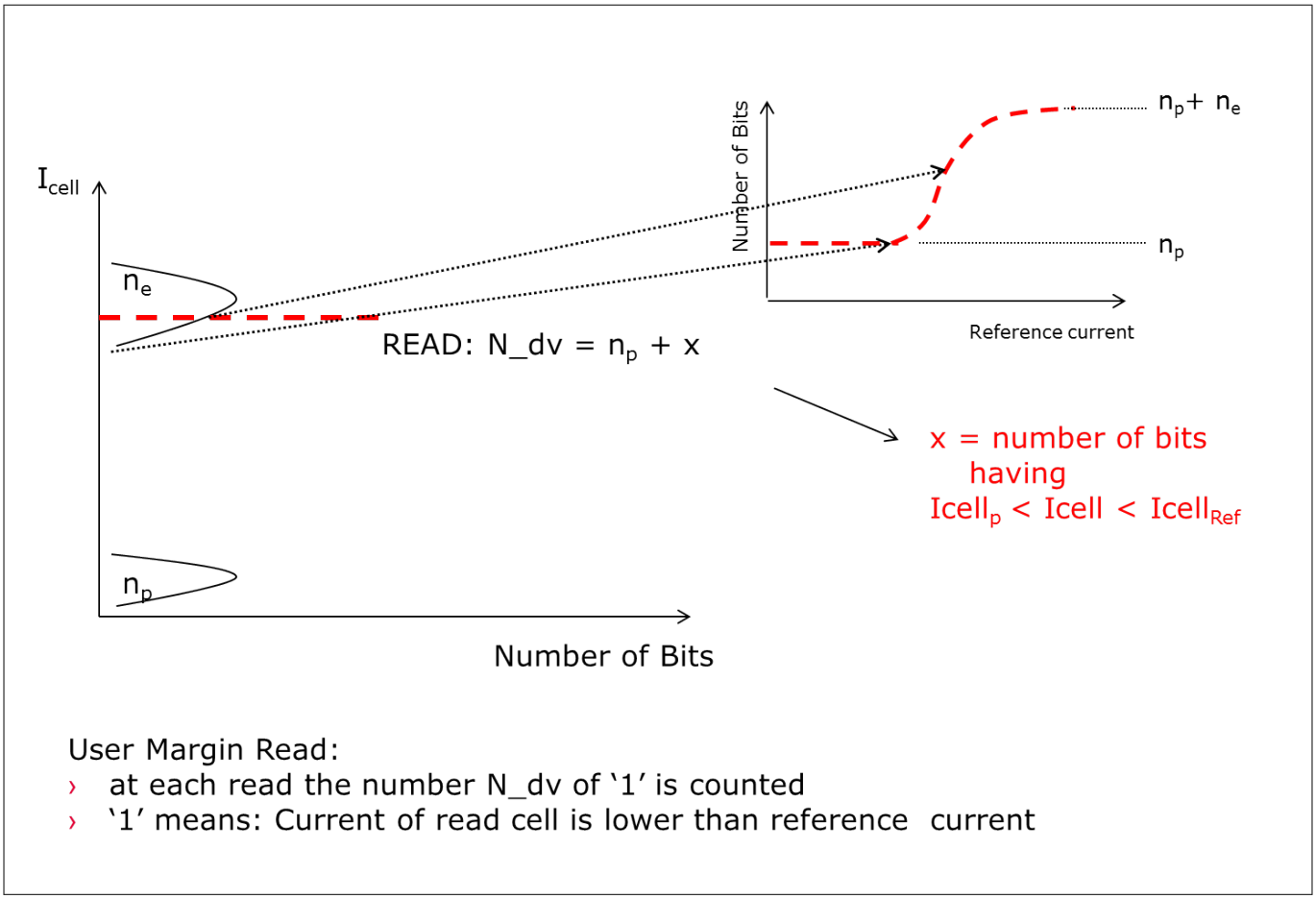


Figure 44 Example of shifted reference current of user margin count

Results

- DFLASH
  - The result is returned as 13-bit unsigned integer with bits [7:0] in FSI_COMM0 [7:0] and bits [12:8] in FSI_COMM1[4:0]
  - The ECC errors are evaluated and stored in FSI_COMM1 bit 5 for triple bit error, bit 6 for double bit error, bit 7 for uncorrectable error
- PFLASH
  - The result is returned as 14-bit unsigned integer with bits [7:0] in FSI_COMM0 [7:0] and bits [13:8] in FSI_COMM1 [5:0]
  - The ECC errors are evaluated and stored in FSI_COMM1 bit 6 for double bit error, bit 7 for uncorrectable error

***Note**: The registers FSI_COMM0 and FSI_COMM1 support only byte wide accesses. Other access sizes generate a bus error. For register address see section at the end of the chapter.*

Error flags

A *Sequence Error (SQER)* is set if **WA** address alignment does not fit to the scope selected (wordline or page).

A *Protection Error (PROER)* is set if:

- Write protection for target memory area is active
- PROT/APU access protection forbids an access of the requesting master

FSI_COMMx register addresses

There are separate FSI_COMMx registers for the HOST and for the CSRM command interface:

- FSI_HOST.COMMx
- FSI_CSRM.COMMx

These COMMx registers do not belong to the standard DMU register region. To access those registers please refer to:

Table 405 FSI COMMx registers

Address	Chapter
Register Base Address FSI_HOST.COMM0	System address map
Register Base Address FSI_CSRM.COMM0	
Register offset addresses for COMMx Registers Note: The offset addresses for FSI_HOST.COMMx and for FSI_CSRM.COMMx are the same	Register Overview - FSI

Application hints

- During the execution of a `User Margin Count`, `User Vth Count` or `User Content Count` command the BUSY flag of the corresponding bank is set in the status registers DMU_HCI_STATUS (HOST) or DMU_CSCI_STATUS (CSRM)
- The COMMx registers must not be changed during the execution of the commands
- The default content 0 of the COMM0 and COMM1 registers shall be set after the end of the execution

5.3.8.15 User Vth count

Calling

- ST $AA50_H$, **WA**
- ST $AA58_H$, $xx00_H$
- ST $AAA8_H$, $xx60_H$
- ST $AAA8_H$, $xx12_H$

Function

Attention: ***This command is ONLY for characterization and analysis purpose. If this command is used many times on the same memory location then a disturb may occur.***

With this command the distribution of the cell threshold voltage Vth of erased cells can be investigated. The scope is either a single page or a complete wordline; this can be selected by the DMU_HCI_OCONTROL.UCG (or DMU_CSCI_OCONTROL.UCG for CSRM command channel). Depending on the scope the argument **WA** must then be used to select a page or wordline aligned address.

The result of this function is the number N_pv of logic 1 bits in the selected wordline or page having a threshold voltage Vth higher than the selected wordline voltage. Calling this function with different wordline voltages allows the calculation of the Vth distribution (please refer to the following figures for examples).

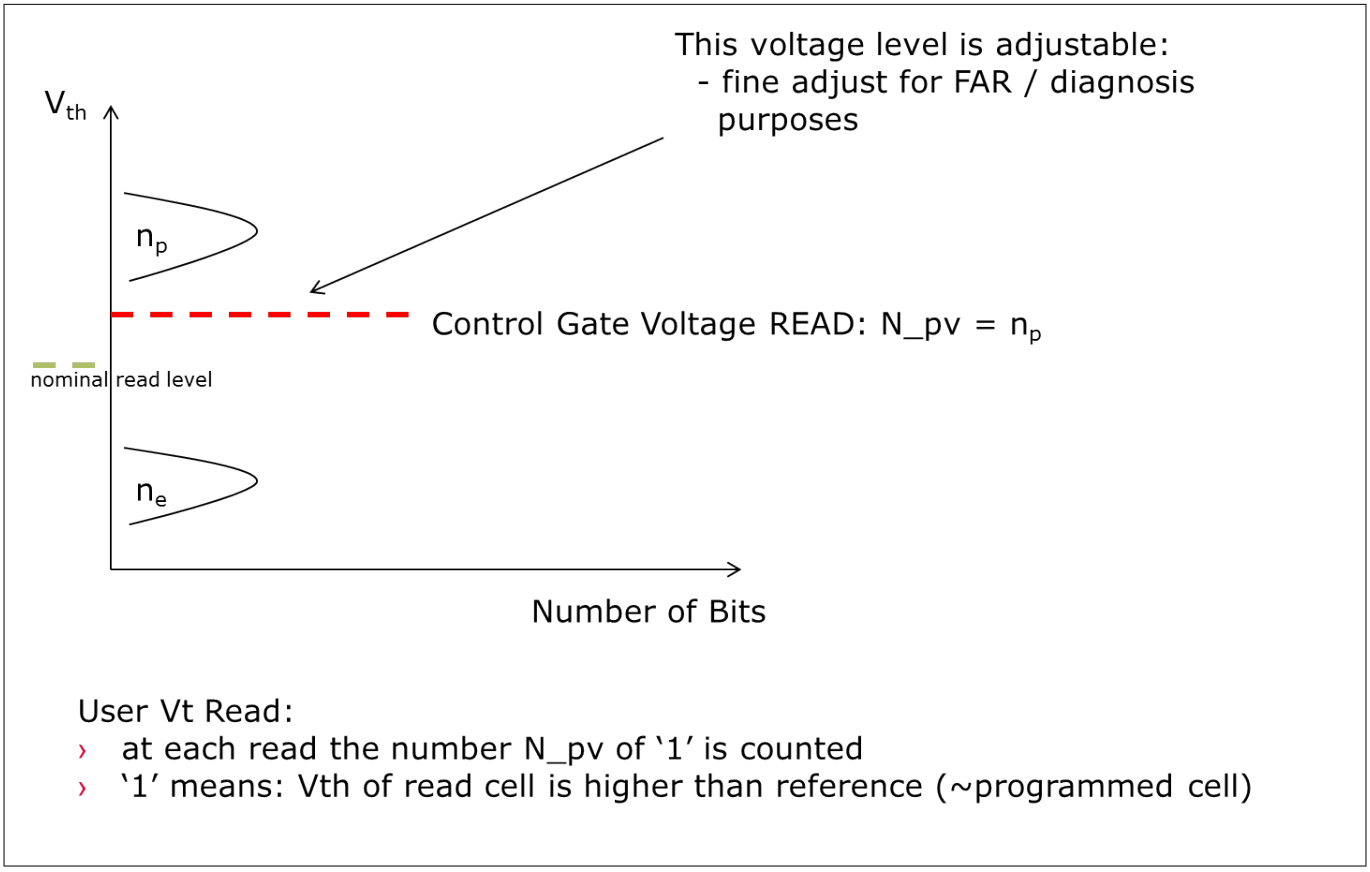


Figure 45 Basic principle of user Vth count

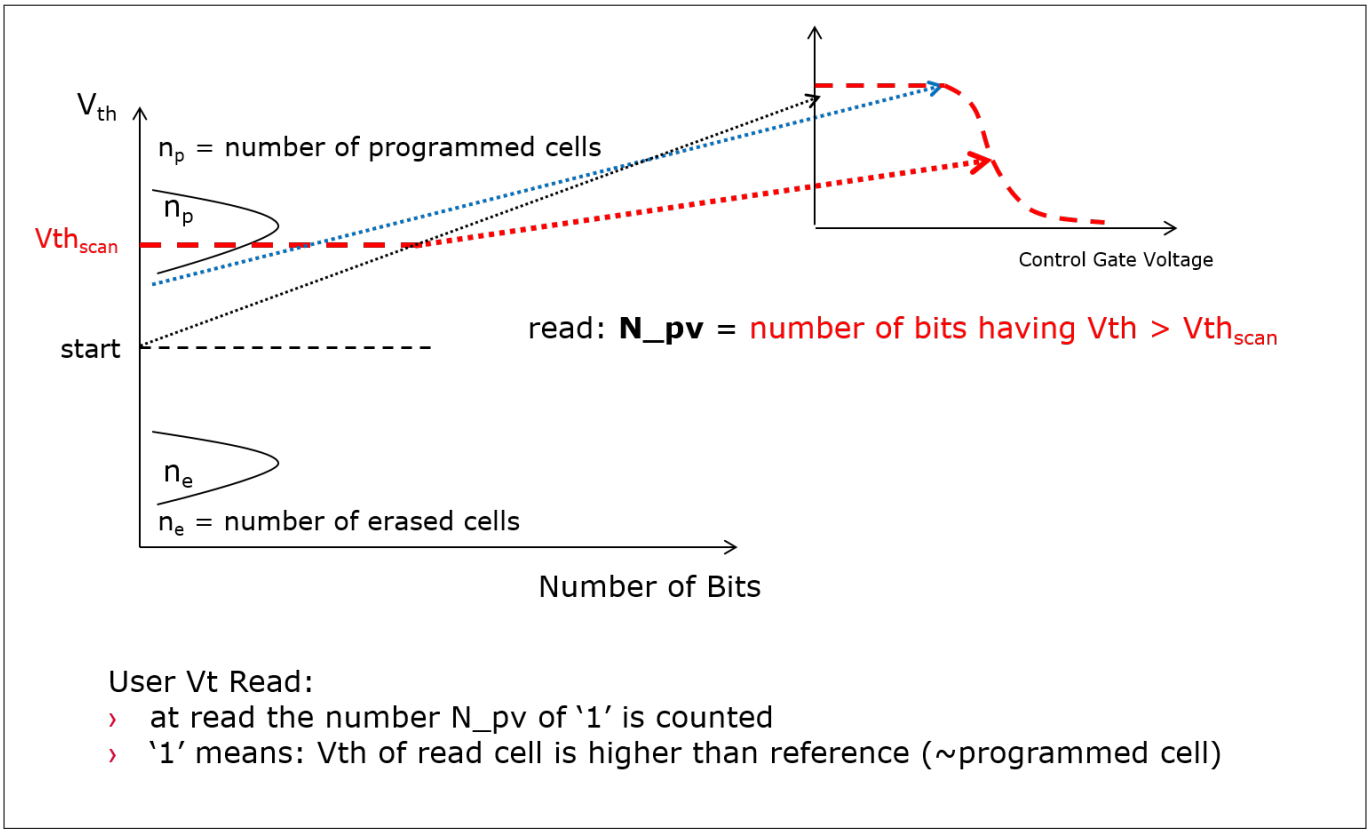


Figure 46 Example of shifted wordline voltage of user Vth count

The **wordline voltage** can be specified by bits [5:0] of FSI_COMM0.
The valid settings are described in the table below.

Table 406 Wordline voltage settings

Hex	Dec	Bin	Voltage
00	0	000000	2,1
01	1	000001	2,2
02	2	000010	2,3
03	3	000011	2,4
04	4	000100	2,5
05	5	000101	2,6
06	6	000110	2,7
07	7	000111	2,8
08	8	001000	2,9
09	9	001001	3,0
0A	10	001010	3,1
0B	11	001011	3,2
0C	12	001100	3,3
0D	13	001101	3,4
0E	14	001110	3,5

(table continues...)

Table 406 (continued) Wordline voltage settings

Hex	Dec	Bin	Voltage
0F	15	001111	3,6
10	16	010000	3,7
11	17	010001	3,8
12	18	010010	3,9
13	19	010011	4,0
14	20	010100	4,1
15	21	010101	4,2
16	22	010110	4,3
17	23	010111	4,4
18	24	011000	4,5
19	25	011001	4,6
1A	26	011010	4,7
1B	27	011011	4,8
1C	28	011100	4,9
1D	29	011101	5,0
1E	30	011110	5,1
1F	31	011111	5,2
20	32	100000	5,3
21	33	100001	5,4
22	34	100010	5,5

Results

- DFLASH
  - The result is returned as 13-bit unsigned integer with bits [7:0] in FSI_COMM0 [7:0] and bits [12:8] in FSI_COMM1[4:0]
  - The ECC errors are evaluated and stored in FSI_COMM1 bit 5 for triple bit error, bit 6 for double bit error, bit 7 for uncorrectable error
- PFLASH
  - The result is returned as 14-bit unsigned integer with bits [7:0] in FSI_COMM0 [7:0] and bits [13:8] in FSI_COMM1 [5:0]
  - The ECC errors are evaluated and stored in FSI_COMM1 bit 6 for double bit error, bit 7 for uncorrectable error

***Note**: The registers FSI_COMM0 and FSI_COMM1 support only byte wide accesses. Other access sizes generate a bus error. For register address see section at the end of the chapter.*

Disturbs by Vth count

Frequent use of the `User Vth Count` Function on one wordline might lead to a disturb of the data within the wordline; especially if high voltages are used. In a worst case this disturb might lead to a change for some cells. However occasional use for characterization or analysis (about 10 times a wordline with the max voltage of 5.5 V) will not lead to a problem. Please consult your local Infineon technical customer service in case of questions.

Error flags

A *Sequence Error (SQER)* is set if **WA** address alignment does not fit to the scope selected (wordline or page).

A *Protection Error (PROER)* is set if:

- Write protection for target memory area is active
- PROT/APU access protection forbids an access of the requesting master

FSI_COMMx register addresses

There are separate FSI_COMMx registers for the HOST and for the CSRM command interface:

- FSI_HOST.COMMx
- FSI_CSRM.COMMx

These COMMx registers do not belong to the standard DMU register region. To access those registers please refer to:

Table 407 FSI COMMx registers

Address	Chapter
Register Base Address FSI_HOST.COMM0	System address map
Register Base Address FSI_CSRM.COMM0	
Register offset addresses for COMMx Registers Note: The offset addresses for FSI_HOST.COMMx and for FSI_CSRM.COMMx are the same	Register Overview - FSI

Application hints

- During the execution of a `User Margin Count`, `User Vth Count` or `User Content Count` command the BUSY flag of the corresponding bank is set in the status registers DMU_HCI_STATUS (HOST) or DMU_CSCI_STATUS (CSRM)

- The COMMx registers must not be changed during the execution of the commands
- The default content 0 of the COMM0 and COMM1 registers shall be set after the end of the execution

5.3.8.16 User content count

Calling

[Function description is not yet complete]

- ST $AA50_H$, **WA**
- ST $AA58_H$, $xx00_H$
- ST $AAA8_H$, $xx60_H$
- ST $AAA8_H$, $xx14_H$

Function

Attention**:* ***This command is mainly for characterization and analysis purpose. Limited use during run-time for data integrity checks may be possible but applicable disturb parameters and default configuration need to be taken into account. Please check with your local Infineon representative in case that it is intended to use this command during runtime

The result of the User Content Count function is the difference of the logic 1 bits counted by the User Vth Count and the User Margin Count functions:

result:= N_dv - N_pv

The User Content Count function is a data integrity check which detects bits with a cell current level between the erase verify- and an adjustable wordline verify- level (please refer to the following figures). This function may be called by setting different wordline (control gate) voltages. Changing of the erase verify level is not possible; the current level is the same as used for the `Verify Erased Logical Sector Range` function.

The scope is either a single page or a complete wordline; this can be selected by the DMU_HCI_OCONTROL.UCG (or DMU_CSCI_OCONTROL.UCG for CSRM command channel). Depending on the scope the argument **WA** must then be used to select a page or wordline aligned address.

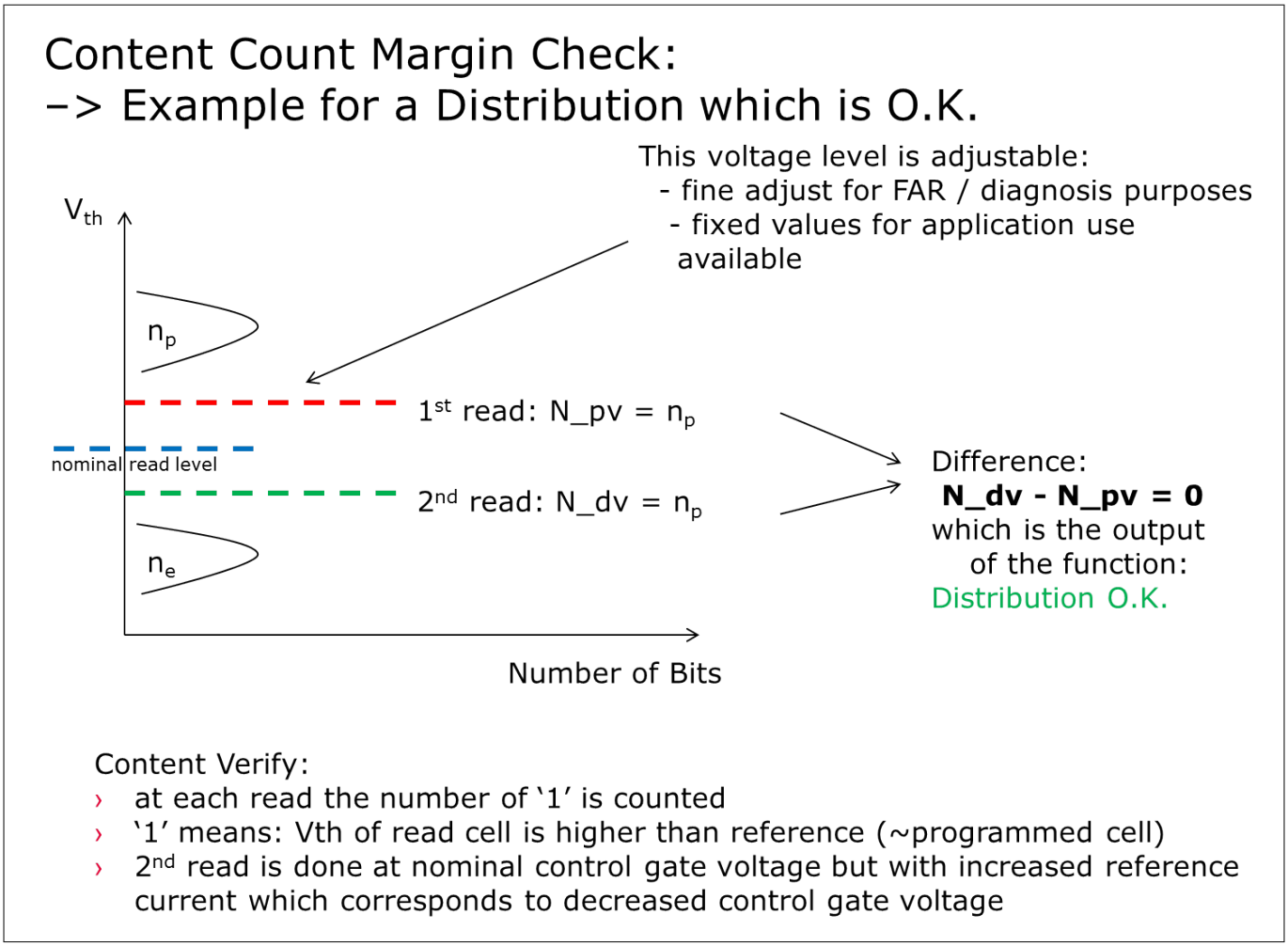


Figure 47 Basic principle of user content count

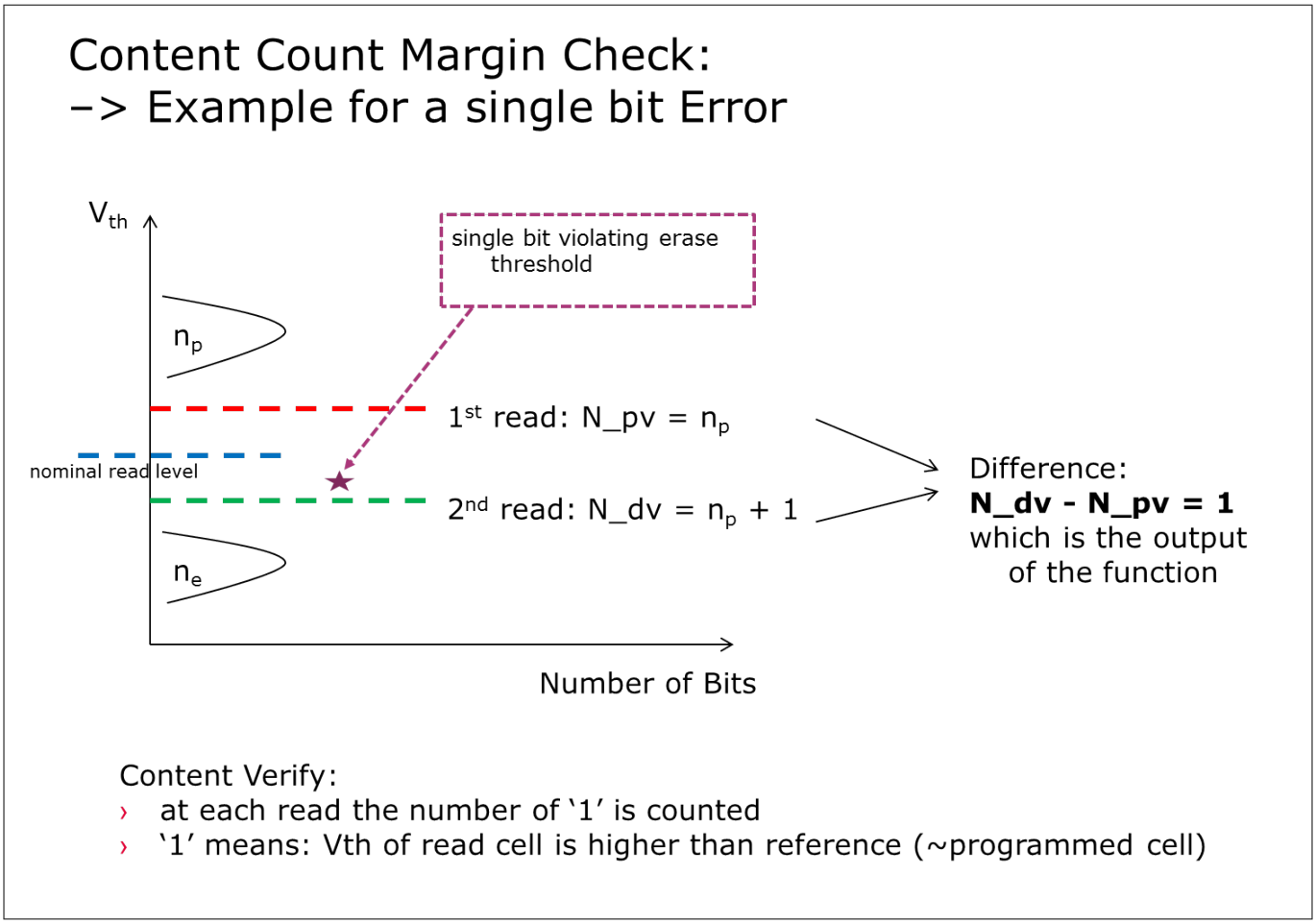


Figure 48 Example of user content count with 1 bit error

The **wordline voltage** can be specified by bits [5:0] of FSI_COMM0.
The valid settings are described in the table below.

Table 408 Wordline voltage settings

hex	dec	bin	voltage
00	0	000000	2,1
01	1	000001	2,2
02	2	000010	2,3
03	3	000011	2,4
04	4	000100	2,5
05	5	000101	2,6
06	6	000110	2,7
07	7	000111	2,8
08	8	001000	2,9
09	9	001001	3,0
0A	10	001010	3,1
0B	11	001011	3,2

(table continues...)

Table 408 (continued) Wordline voltage settings

hex	dec	bin	voltage
0C	12	001100	3,3
0D	13	001101	3,4
0E	14	001110	3,5
0F	15	001111	3,6
10	16	010000	3,7
11	17	010001	3,8
12	18	010010	3,9
13	19	010011	4,0
14	20	010100	4,1
15	21	010101	4,2
16	22	010110	4,3
17	23	010111	4,4
18	24	011000	4,5
19	25	011001	4,6
1A	26	011010	4,7
1B	27	011011	4,8
1C	28	011100	4,9
1D	29	011101	5,0
1E	30	011110	5,1
1F	31	011111	5,2
20	32	100000	5,3
21	33	100001	5,4
22	34	100010	5,5

Results

- DFLASH
  - The result is returned as 13-bit unsigned integer with bits [7:0] in FSI_COMM0 [7:0] and bits [12:8] in FSI_COMM1[4:0]
  - The ECC errors are evaluated and stored in FSI_COMM1 bit 5 for triple bit error, bit 6 for double bit error, bit 7 for uncorrectable error
- PFLASH
  - The result is returned as 14-bit unsigned integer with bits [7:0] in FSI_COMM0 [7:0] and bits [13:8] in FSI_COMM1 [5:0]
  - The ECC errors are evaluated and stored in FSI_COMM1 bit 6 for double bit error, bit 7 for uncorrectable error

***Note**: The registers FSI_COMM0 and FSI_COMM1 support only byte wide accesses. Other access sizes generate a bus error. For register address see section at the end of the chapter.*

Disturbs by user content count

Frequent use of the `User Content Count` function on one wordline might lead to a disturb of the data within the wordline; especially if high voltages are used. In a worst case this disturb might lead to a change for some cells. However occasional use for characterization or analysis (about 10 times a wordline with the max voltage of 5.5 V) will not lead to a problem. Please consult your local Infineon technical customer service in case of questions.

Error flags

A *Sequence Error (SQER)* is set if **WA** address alignment does not fit to the scope selected (wordline or page).

A *Protection Error (PROER)* is set if:

- Write protection for target memory area is active
- PROT/APU access protection forbids an access of the requesting master

FSI_COMMx register addresses

There are separate FSI_COMMx registers for the HOST and for the CSRM command interface:

- FSI_HOST.COMMx
- FSI_CSRM.COMMx

These COMMx registers do not belong to the standard DMU register region. To access those registers please refer to:

Table 409 FSI COMMx registers

Address	Chapter
Register Base Address FSI_HOST.COMM0	System address map
Register Base Address FSI_CSRM.COMM0	
Register offset addresses for COMMx Registers Note: The offset addresses for FSI_HOST.COMMx and for FSI_CSRM.COMMx are the same	Register Overview - FSI

Application hints

- During the execution of a `User Margin Count`, `User Vth Count` or `User Content Count` command the BUSY flag of the corresponding bank is set in the status registers DMU_HCI_STATUS (HOST) or DMU_CSCI_STATUS (CSRM)

- The COMMx registers must not be changed during the execution of the commands
- The default content 0 of the COMM0 and COMM1 registers shall be set after the end of the execution

User of user content count during runtime

The User Content Count function may be used during runtime for data integrity checks with a fixed single value for the wordline voltage. For this purpose recommended values for PFLASH and DFLASH (HOST or CSRM) are available.

Attention**: **Recommended values for use during runtime may be changed based on final characterization and qualification results. Please contact your local Infineon representative.

5.3.8.17 Disable protection

Calling

- ST $553C_H$, **UC**
- ST.W $553C_H$, **PW0**
- ST.W $553C_H$, **PW1**
- ST.W $553C_H$, **PW2**
- ST.W $553C_H$, **PW3**
- ST.W $553C_H$, **PW4**
- ST.W $553C_H$, **PW5**
- ST.W $553C_H$, **PW6**
- ST.W $553C_H$, **PW7**

Function

The password protection of the selected UCB (if this UCB offers this feature) is temporarily disabled and the respective bit in GP_HOST_PROTECT or GP_CSRM_PROTECT is set. when all the passwords **PW0** .. **PW7** match their configured values in the corresponding UCB.

The following protections may be disabled temporarily by using the Disable Protection command:

- "Global" Read and "Global" Write Protection
- Write Protection of a specific Bank

The following protections cannot be disabled anymore:

- Permanent Write ("OTP") Protection
- Permanent Write Once Protection

This command sequence generates a PROER if

1. any supplied **PWx** is incorrect or
2. if the confirmation code of the target UCB is not in UNLOCKED or CONFIRMED state (see UCB confirmation code registers CONFIRMA, CONFIRMB, and CONFIRMC)

In such cases, an application reset is needed before trying this command sequence again. Otherwise, all further calls of the Disable Protection will fail with a PROER regardless of the supplied password.

Before comparing passwords, the DMU reads again the confirmation codes from DF and if necessary, it updates the corresponding registers GP_HOST_CONFIRMA, GP_HOST_CONFIRMB, GP_HOST_CONFIRMC, GP_CSRM_CONFIRMA, GP_CSRM_CONFIRMB or GP_CSRM_CONFIRMC.

***Note**:* *Host and CSRM BHMDs have a specific handling.*

Every disable protection sequence issued on the BMHDs shall prompt the DMU to read the BMHDs confirmation codes (CC) in ascending order, from BMHD0 to BMHD3, regardless of their CC values.

The overall status of the BMHDs, that ultimately controls their access and protection, is evaluated based on the status of the first valid BMHD. The table below describes the status of the BMHD UCBs after the programming phase and following a reset.

***Note**:* *during the UCBs programming phase, it is recommended to avoid any resets.*

***Note**:* *the ERASED CC is a special case of ERRORED and therefore the behavior is the same. ERRORED can mean ERASED or virgin, that is never been written*

Table 410 Handling of BMHDs CC and PWD combination

BMHD0 CC	BMHD1 CC	BMHD2 CC	BMHD3 CC	BMHDs overall status	Access	PWD protection
UNLOCKED	Any state	Any state	Any state	UNLOCKED	Full read and write access on all BMHDs, regardless of their status.	Disabled
CONFIRMED	Any state	Any state	Any state	CONFIRMED	BMHDs PWDs not readable. Content of all BMHDs can be read but it cannot be written.	Enabled
ERRORED	UNLOCKED	Any state	Any state	UNLOCKED	Full read and write access on all BMHDs, regardless of their status.	Disabled
ERRORED	CONFIRMED	Any state	Any state	CONFIRMED	BMHDs PWDs not readable. Content of all BMHDs can be read but it cannot be written.	Enabled
ERRORED	ERRORED	UNLOCKED	Any state	UNLOCKED	Full read and write access on all BMHDs, regardless of their status.	Disabled
ERRORED	ERRORED	CONFIRMED	Any state	CONFIRMED	BMHDs PWDs not readable. Content of all BMHDs can be read but it cannot be written.	Enabled
ERRORED	ERRORED	ERRORED	UNLOCKED	UNLOCKED	Full read and write access on all BMHDs, regardless of their status.	Disabled
ERRORED	ERRORED	ERRORED	CONFIRMED	CONFIRMED	BMHDs PWDs not readable. Content of all BMHDs can be read but it cannot be written.	Enabled

(table continues...)

Table 410 (continued) Handling of BMHDs CC and PWD combination

BMHD0 CC	BMHD1 CC	BMHD2 CC	BMHD3 CC	BMHDs overall status	Access	PWD protection
ERRORED	ERRORED	ERRORED	ERRORED	Braindead	None of the BMHDs PWDs can be read. No write access.	Enabled

***Note**: the PWD protection column in the table above indicates the protection status following a reset. It is evaluated based on the GP_HOST_PROTECT or GP_CSRM_PROTECT registers and of the BMHDs overall status. During the programming phase, changing the confirmation code of the UCBs does not enable or disable the protection.*

***Note**: the value of the CCs does not reflect their protection status. A disable protection command sequence does not change the CC value but changes the relevant status bit in the GP_HOST_PROTECT or GP_CSRM_PROTECT registers.*

To disable the protection of all BHMDs, the disable protection command sequence shall be issued with the password of the first CONFIRMED BMHD, providing that the preceding BMHD (where applicable) is not UNLOCKED.

A disable protection issued with the correct PWD on an BMHD with UNLOCKED CC shall set the BMHD dedicated bit in the GP_HOST_PROTECT or GP_CSRM_PROTECT register, providing that the preceding BMHDs (where applicable) are not UNLOCKED or CONFIRMED and that BMHDs PWDs differ. Otherwise, a PROER shall be returned.

A disable protection issued with the correct PWD on the second or subsequent valid BMHD shall return a PROER providing that all BMHDs PWDs differ.

***Note**: successfully disabling the protection of the BMHDs allows full read and write access to them, regardless of their confirmation code.*

In addition, it is advised that the confirmation code of the first valid BMHD shall always be set to CONFIRMED to prevent reads of the BMHDs password that would otherwise remove the protection to all BMHDs

In the current implementation of the design, every disable protection sequence issued on the dual UCBs shall prompt the DMU to read the ORIG CC first followed by the COPY CC, regardless of their CC values.

The overall status of the dual UCBs, that ultimately controls their access and protection, is evaluated based status of the first valid dual UCB. The table below describes the status of the dual UCBs after the programming phase and following a reset.

***Note**: during the dual UCBs programming phase, it is recommended to avoid any resets.*

Table 411 Handling of dual UCBs CC and PWD combination

ORIG UCB CC	COPY UCB CC	UCB overall status	Access	PWD protection
UNLOCKED	UNLOCKED	UNLOCKED	Full read and write access on both ORIG and COPY, regardless of their status.	Disabled

(table continues...)

Table 411 (continued) Handling of dual UCBs CC and PWD combination

ORIG UCB CC	COPY UCB CC	UCB overall status	Access	PWD protection
UNLOCKED	CONFIRMED	UNLOCKED	Full read and write access on both ORIG and COPY, regardless of their status. The protection on COPY UCB shall not be enabled because the main UCB is ORIG	
UNLOCKED	ERRORED	UNLOCKED	Full read and write access on both ORIG and COPY, regardless of their status.	
CONFIRMED	UNLOCKED	CONFIRMED	Neither ORIG nor COPY PWDs can be read. Content of ORIG and COPY can be read but it cannot be written. Disabling the protection on ORIG allows write access to both UCBs regardless of the status of the COPY UCB.	Enabled
CONFIRMED	CONFIRMED	CONFIRMED		
CONFIRMED	ERRORED	CONFIRMED		
ERRORED	UNLOCKED	UNLOCKED	Full read and write access on both ORIG and COPY, regardless of their status.	Disabled
ERRORED	CONFIRMED	CONFIRMED	PWD of neither ORIG nor COPY can be read. Content of ORIG and COPY can be read but it cannot be written. Disabling the protection on COPY allows write access to both UCBs.	Enabled
ERRORED	ERRORED	ERRORED	UCBs PWDs cannot be read. No write access	Enabled

To disable the protection of the dual UCBs, when the ORIG CC is CONFIRMED, the disable protection command sequence shall be issued with the password of the ORIG UCB, regardless of the value of the COPY CC.

Otherwise, disabling the protection of the dual UCBs when the ORIG CC is ERRORED and the COPY CC is CONFIRMED, the disable protection command sequence shall be issued with the password of the COPY UCB.

In either case, the ERRORED UCB shall be provided full read and write access.

***Note**: PWD protection is evaluated based on the GP_HOST_PROTECT or GP_CSRM_PROTECT registers and of the dual UCBs overall status. The value of the dual UCBs CCs does not reflect their protection status. A disable protection command sequence does not change the CC value but changes the relevant status bit in the GP_HOST_PROTECT or GP_CSRM_PROTECT registers.*

***Note**: Limitations in the AURIX™ 3rd Generation*

- *Single-UCBs: UCB_USER_DATA0, UCB_USER_DATA1, UCB_CS_USER_DATA0, and UCB_CS_USER_DATA1*
  - *If the UCB is in ERRORED state, the UCB cannot be recovered*

***Note**: In order to avoid this shortcoming, during the programming phase the following shall not occur:*

- *writing the incorrect confirmation code in the UCBs*
- *brownout or voltage losses*
- *resets while writing confirmation codes*

but also ensure that:

- *setups with compiler generating the binary files are generated correctly*
- *no issues with the debug tools present*

5.3.8.18 Resume protection

Calling

- ST 5554_H, xxF5$_H$

Function

This command clears all DMU_GP_HOST_PROTECT (Host Command Interface) or DMU_GP_CSRM_PROTECT (CSRM Command Interface) and enables again the Flash protection as it was configured.

5.3.8.19 Clear status

Calling

- ST 5554_H, xxFA_H

Function

Clear operation and error flags.
Clearing of error flags will not solve the underlying problem that caused the error flag.

Host Command Sequence interpreter

The following flags of the register DMU_HCI_STATUS are cleared:
- PROG
- ERASE
- USER
- REQACK
- REQDONE

The following flags of the register DMU_HCI_ERR are cleared:
- PVER
- EVER

CSRM Command Sequence interpreter

The following flags of the register DMU_CSCI_STATUS are cleared:
- PROG
- ERASE
- USER
- REQACK
- REQDONE

The following flags of the register DMU_CSCI_ERR are cleared:
- PVER
- EVER

5.3.8.20 Bank sleep

Calling

- ST $AA00_H$, **BS**

Function

Sets all banks defined by **BS** into an off - state ("`Sleep`") for power reduction during which no read or operation on these banks is possible.

- **BS**: Bank selection coded as 32-bit vector:
  - Bit 0: HOST PFLASH0A
  - Bit 1: HOST PFLASH0B
  - Bit 2 : HOST PFLASH1A
  - ...
  - Bit 11: HOST PFLASH5B
  - Bit 12 -15 : reserved - 0
  - Bit 16 : HOST DFLASH (includes UCB)
  - Bit 17: CSRM DFLASH (includes UCB)
  - Bit 18: CSRM PFLASH
  - Bit 19-31 : reserved - 0

***Note**: For products with less PFLASH (HOST) Banks, it is necessary to adapt the BS accordingly. Example: a product with 10 PFLASH (HOST) banks would make use of bits 0 to 9 (i.e. HOST PFLASH0A to PFLASH4B) and would have bits 10 - 15 as reserved.*

Please note the following hints:

- The command interface can only set to `Sleep` those Banks which are allocated to this command channel
- A bank under operation (for example erase) cannot be set into `Sleep` mode[9].
- A Bank in `Sleep` mode gets a BUSY indication in the status register DMU_HCI_STATUS or DMU_CSCI_STATUS
- A bank in `Sleep` mode gets an indication in the status register DMU_HCI_SLEEP or DMU_CSCI_SLEEP

The command fails with an error flag in the following situations:

- A part of the addressed banks is already in sleep → *Sequence Error (SQER)*
- An invalid bank is selected (for example, reserved bits), → *Sequence Error (SQER)*
- An operation is running → *Bus Error (BE)*
- Master is not allowed to operate on the command interface → *Bus Error (BE)*

***Note**: If all banks of both command interfaces are set to `Sleep` then the NVM will enter a "deep sleep" state during which also common analog resources are switched off.*

HOST command interface registers

Table 412 HOST command interface registers

Register	Use
DMU_HCI_STATUS	Status, BUSY indication
DMU_HCI_ERR	Error flags
DMU_HCI_SLEEP	Sleep status indication

[9] application of sleep would only be possible if the NVM operation was aborted

CSRM command interface registers

Table 413 CSRM command interface registers

Register	Use
DMU_CSCI_STATUS	Status, BUSY indication
DMU_CSCI_ERR	Error flags
DMU_CSCI_SLEEP	Sleep status indication

5.3.8.21 Bank wake-up

Calling

- ST $AAF0_H$, **BS**

Function

Wakes up all (sleeping) banks defined by **BS** into an idle state during which a read or operation on these banks is possible.

- **BS**: Bank selection coded as 32-bit vector:
  - Bit 0: HOST PFLASH0A
  - Bit 1: HOST PFLASH0B
  - Bit 2 : HOST PFLASH1A
  - …
  - Bit 11: HOST PFLASH5B
  - Bit 12 -15 : reserved - 0
  - Bit 16 : HOST DFLASH (includes UCB)
  - Bit 17: CSRM DFLASH (includes UCB)
  - Bit 18: CSRM PFLASH
  - Bit 19-31 : reserved - 0

***Note**: For products with less PFLASH (HOST) Banks, it is necessary to adapt the BS accordingly. Example: a product with 10 PFLASH (HOST) banks would make use of bits 0 to 9 (i.e. HOST PFLASH0A to PFLASH4B) and would have bits 10 - 15 as reserved.*

Please note the following hints:

- The command interface can only wake up those banks which are allocated to this command channel
- A Bank under read or operation cannot be waken up

The command fails with an error flag in the following situations:

- A part of the banks is already in read or operation state → *Sequence Error (SQER)*
- An invalid bank is selected (for example, reserved bits), → *Sequence Error (SQER)*
- An operation is running → *Bus Error (BE)*
- Master is not allowed to operate on the command interface → *Bus Error (BE)*

Attention**: **For waking up n banks it is necessary to apply the command n times.

***Note**: If the NVM is in Deep Sleep then the wake up timing of the first bank is longer because first the common analog resources are switched on, then the Bank is woken up.*

HOST command interface registers

Table 414 HOST command interface registers

Register	Use
DMU_HCI_STATUS	Status, BUSY indication
DMU_HCI_ERR	Error Flags
DMU_HCI_SLEEP	Sleep status indication

CSRM command interface registers

Table 415 CSRM command interface registers

Register	Use
DMU_CSCI_STATUS	Status, BUSY indication
DMU_CSCI_ERR	Error Flags
DMU_CSCI_SLEEP	Sleep status indication

5.3.8.22 Bank reallocation

Calling

- ST $AA90_H$, **BS**

Function

Moves all Banks defined by **BS** to the other command interface.

- **BS**: Bank selection coded as 32-bit vector:
  - Bit 0: HOST PFLASH0A
  - Bit 1: HOST PFLASH0B
  - Bit 2 : HOST PFLASH1A
  - ..
  - Bit 11: HOST PFLASH5B
  - Bit 12 -15 : reserved - 0
  - Bit 16 : HOST DFLASH (includes UCB)
  - Bit 17: reserved - 0
  - Bit 18: reserved - 0
  - Bit 19-31 : reserved - 0

***Note**: For products with less PFLASH (HOST) Banks, it is necessary to adapt the BS accordingly. Example: a product with 10 PFLASH (HOST) banks would make use of bits 0 to 9 (i.e. HOST PFLASH0A to PFLASH4B) and would have bits 10 - 15 as reserved.*

Please note the following hints:

- A command interface can only move those banks to the other command interface which currently belong to the interface executing the move command
- Security memory banks cannot be moved
- Only two different modes (allocation sets) are possible
- If one or more of the banks selected to move are under operation (for example erase) then a reallocation is not possible
- If one or more of the banks selected to move are in `Sleep` then a reallocation is not possible
- Banks with ongoing read - accesses may be re-allocated without impact on the reading
- It is only possible to move all PFLASH HOST Banks together but not a subset of it

Table 416 Allowed BS settings

BS Setting for products with 12 PFLASH HOST Banks	Selection
0000 $0FFF_H$	All HOST PFLASH Banks selected

***Note**: For products with less PFLASH (HOST) Banks it is necessary to adapt the **BS**. Example: A product with 8 PFLASH(HOST) banks would need **BS** = 0000 $00FF_H$ to select all PFLASH(HOST) Banks .*

Table 417 Allocation of NVM banks to the command interfaces

DMU Command interface mode	Host command interface	CSRM command interface
Domain mode	HOST PFLASH PFx HOST DFLASH DF(0)	CSRM PFLASH CSRM DFLASH DF(1)
CSRM_ALL_PFLASH mode	HOST DFLASH DF(0)	CSRM PFLASH CSRM DFLASH DF(1) HOST PFLASH PF(x)

The following table and picture shows the allowed transitions between the different Bank allocation; the table also mentions which command interface (CI: HOST or CSRM) may execute the command.

Table 418 Possible transitions

Initial allocation	Command interface involved (CI)	Final allocation
Domain	HOST CI: reallocate PFLASH(HOST) → CSRM CI	CSRM_ALL_PFLASH
CSRM_ALL_PFLASH	CSRM CI: reallocate PFLASH(HOST) → HOST CI	Domain

***Note**: A mode with DFLASH(HOST) on CSRM CI but PFLASH(HOST) on HOST CI is not possible.*

The command fails with an error flag in the following situations:

- A wrong **BS** selection → *Sequence Error (SQER)*
- One or several banks selected for reallocation are under operation (for example `Erase` ongoing) → *Sequence Error (SQER)*
- One or several banks selected for reallocation are in `Sleep` → *Sequence Error (SQER)*
- Master is not allowed to operate on one or several banks selected for reallocation → *Protection Error (PROER)*

Common bank allocation register

The following register shows the current allocation of the memory banks: DMU_GP_BKALLOC

Status bits of PFLASH(HOST) banks during the CSRM_ALL_PFLASH mode

If the PFLASH(HOST) Banks are allocated to the CSRM command interface then the status bits for BUSY and SLEEP states for these Banks are updated only in the CSRM status registers (see the following table). Status bits of these Banks in the HOST status registers are not updated.

To enable an interrupt correlated with a NVM operation on the PFLASH (HOST) banks it is also necessary to use the CSCI_INTEN register.

Also any other control setting or error status information for operations on PFLASH(HOST) memories shall be set or checked in the corresponding CSRM control or status registers (for example CSCI_ERR or CSCI_OCONTROL.

Table 419 CSRM command interface registers

Register	Use
DMU_CSCI_STATUS	Status, BUSY indication
DMU_CSCI_ERR	Error flags

Concurrent NVM operations

Concurrent Operations in Domain Mode:

In Domain Mode the NVM shall provide the software actors with the ability to perform each of the concurrent operation combinations mentioned in the columns of the following table.

HOST channel	HOST PF (bankx)	read	read	read	operation	read	read	read	operation	read	read	read	operation
	HOST PF (banky)	read	read	operation	read	read	read	operation	read	read	read	operation	read
	HOST DF / UCB / CFS	read*	operation	read*	read*	read	operation	read	read	read*	operation	read*	read*
CSRM channel	CSRM PF	read	read	read	read	read	read	read	read	operation	operation	operation	operation
	CSRM DF / UCB / CFS	read*	read	read*	read*	operation	operation	operation	operation	read*	read	read*	read*

Figure 49 Domain Mode Concurrent Operations

Notes:

- "read*" read may be delayed on system to the single SRI port
- "operation" means all ucode controlled sequences, so also a read for a prog verify is included here

Concurrent Operations in CSRM ALL PFLASH Mode:

In CSRM ALL PFLASH mode the NVM shall provide the software actors with the ability to perform each of the concurrent operation combinations mentioned in the columns of the following table.

HOST channel	HOST DF / UCB / CFS	read*	read	operation	read*	operation	read*	operation	read*	operation
CSRM channel	HOST PF (bankx)	read	read	read	read	read	read	read	operation	operation
	HOST PF (banky)	read	read	read	read	read	operation	operation	read	read
	CSRM PF	read	read	read	operation	operation	read	read	read	read
	CSRM DF / UCB / CFS	read*	operation	operation	read*	read	read*	read*	read*	read*

Figure 50 CSRM ALL PFLASH Mode Concurrent Operations

Notes:

- "read*" read may be delayed on system to the single SRI port
- "operation" means all ucode controlled sequences, so also a read for a prog verify is included here

Related information

Allocation of banks to command interfaces on page 680

5.3.9 Clocks relevant for NVM

The SRI and FSI clocks are relevant for the NVM functions.

SRI clock

SRI clock is relevant for:

- Counting of wait states during a PFLASH read access
- Duration of data transfers to NVM for write
- Duration of read data transfer from DataFlash

Changing SRI clock

In addition to possible other constraints (see other module chapters and SRI description) it must be ensured that before, during and after the divider change the configured number of PFLASH wait cycles is sufficient for the selected clock frequency. For a change to a higher SRI clock it is necessary to change wait states first to a higher value necessary for the higher clock; then a clock increase can be done.

Attention**: **the SRI clock for the cyber security real time module (CSRM) might be different to the SRI clock of the HOST CPU SRI clusters. This need to be taken into account when configuring the PFLASH read wait states for the CSRM PFLASH Bank. Therefore please check the SRI clock settings for the CSRM from which the applicable wait state settings for the CSRM PFLASH can then be derived.

FSI clock

FSI clock is relevant for:

- Counting of wait states during a DFLASH read access
- Duration of command sequences like `Write Burst` or `Erase Logical Sector Range`

Attention**: **All the command timing parameters in the datasheet (for example: erase timing) are specified for the condition that f_{FSI} = 100 MHz.

Changing FSI clock

It must be ensured that before, during and after the divider change the configured number of DFLASH wait cycles is sufficient for the selected clock frequency. For a change to a higher FSI clock it is necessary to change wait states first to a higher value necessary for the higher clock; then a clock increase can be done.

Attention**: **a FSI clock change during a running NVM operation is NOT possible.

Related information

Configuring read access cycles on page 671

5.3.10 Write operation hints

The following chapters describe the possible options for write commands and show an example sequence for write, which also includes the use of status and error flags. Please refer to the chapter Error Handling for a description of how to handle errors.

Related information

Write page on page 696
Write page once on page 698
Write burst on page 699
Write burst once on page 699
Error handling on page 685

5.3.10.1 Write settings

Overview

The write operation to the NVM may be controlled by the following settings:

1. Page or Burst mode
2. Program pulse width: 256 bit or 4 x 64 bit

Page versus burst mode

In page mode there is only one page written to the NVM (PFLASH: 32 bytes, DFLASH: 8 bytes).

In the burst mode there are 512 bytes (PFLASH) and 32 bytes (DFLASH) written to the NVM.

The benefit of the burst mode is a higher program throughput because internal NVM voltages do not need to be changed so many times.

- The burst mode is available for PFLASH and DFLASH
- Burst or page mode are selected by choosing the dedicated commands
  - Page mode: `Write Page` or `Write Page Once`
  - Burst mode: `Write Burst` or `Write Burst Once`

Write pulse width

For the PFLASH there is an additional option available to increase the program throughput: this can be done by increasing the number of bits which are written with one program pulse to the memory.

Normally only 64 bits are programmed. So a page write for PFLASH would need four program pulses to write the page size of 256 bits (4 x 64 bits). Also for the burst mode there would be 16 x 4 = 64 pulses to achieve 512 bytes.

When the 256 bit pulse register bit is set to 1_B the program pulse for the page mode would be one pulse only with 256 bits being programmed.

Consequences:

- Higher program throughput
- Higher current consumption

The write pulse width increase is only available for PFLASH and can be selected by the following settings:

- Host command interface: set DMU_HCI_OCONTROL.PRMODE to 1_B
- CSRM command interface: set DMU_CSCI_OCONTROL.PRMODE to 1_B

***Note**: The reset value is 0_B which means that per default four pulses are necessary per page.*

***Note**: Once changed the selection stays valid also for the next write commands until it is changed again.*

For the timings please refer to the following datasheet parameters:

Table 420 PFLASH program timing parameters

Parameter	Program pulse	Page or burst
$t_{PRG_P64_P}$	4 x 64 bit	Page (256 bit)
$t_{PRG_P256_P}$	1 x 256 bit	Page (256 bit)
$t_{PRG_P64_B}$	4 x 64 bit	Burst (16 x 256 bit)
$t_{PRG_P256_B}$	1 x 256 bit	Burst (16 x 256 bit)

Attention**: **Please refer to the datasheet for potential additional restrictions for the 256 bit one pulse programming.

***Note**: Selection of 256 bit one pulse programming cause a higher VDDEXT current consumption than the default 4 x 64 bit pulse program mode.*

Use case: High program throughput

To achieve a high program throughput the following settings shall be used:

- Write burst commands
- 256 bit write pulse

This mode may be beneficial for fast programming during fabrication or garage updates. Applicable restrictions (VDDEXT minimum voltage and current consumption) can be taken into account.

Use case: Low VDDEXT current consumption

In order to achieve a low current consumption at the cost of longer execution timing the following settings shall be used:

- Write burst commands
- 4 x 64 bit write pulse setting

This mode may be beneficial to reduce the VDDEXT current consumption for cases where two concurrent write operations occur: for example a PFLASH-Write together with a DFLASH-Write.

Related information

Concurrent operations on page 679

5.3.10.2 Write sequence

Recommended write sequence

The following sequence is recommended for the write operation.

- `Clear Status` command to clear flags
- `Enter Page Mode` command
- Wait until DMU_HCI_STATUS.PFPAGE = ‘1’ or DMU_HCI_STATUS.DFPAGE = ‘1’ depending upon the Flash target, or fail if DMU_HCI_ERR.SQER = ‘1’ or DMU_HCI_ERR.PROER = ‘1’’
- `Clear Status` command to clear flags
- Repeat `Load Page` command until the page is filled (check HCI_ERR.SQER)
- `Clear Status` command to clear flags
- `Write Page` command
- information: the NVM sets DMU_HCI_STATUS.REQACK = ‘1’ as soon as the NVM receives the first cycle of the command sequence
- Wait until DMU_HCI_STATUS.REQDONE = ‘1’
- Error Flag Checks:
  - Fail if DMU_HCI_ERR.PVER flag
  - Fail if DMU_HCI_ERR.OPER = ‘1’
  - Fail if DMU_HCI_ERR.SQER = ‘1’
  - Fail if DMU_HCI_ERR.PROER = ‘1’
- For information about how to handle the errors refer to Error handling
- Recommended: read programmed content and evaluate it by CRC and or error flags of the memory error correction circuit (ECC)
- Clear error flags either with “Clear Status” or by directly writing to DMU_HCI_CLRERR register

***Note**:* *The steps may also be done when using the CSRM command interface; in this case use the registers DMU_CSCI_STATUS, DMU_CSCI_ERR and DMU_CSCI_CLRERR.*

Program Verify Error (PVER)

There are different ways to react on a *Program Verify Error (PVER)*. Please refer to the following two chapters for further information:

- Write page (Write page)
- Error handling (Error handling)

Critical section

The following commands shall be applied by the software in a sequence which is not interrupted by another software accessing the same command interface:

- `Clear Status`
- `Enter Page Mode`
- `Load Page` as often as necessary to fill buffer
- `Clear Status`
- `Write Page` (or `Write Burst`)

Alternative write sequence using several interrupts

The following sequence is a possible alternative to the write sequence. For this alternative several interrupts are used.

- `Clear Status` command to clear flags

- `Enter Page Mode` command
- Wait until DMU_HCI_STATUS.PFPAGE = ‘1’ or DMU_HCI_STATUS.DFPAGE = ‘1’ depending upon the Flash target, or fail if DMU_HCI_ERR.SQER = ‘1’ or DMU_HCI_ERR.PROER = ‘1’’
- `Clear Status` command to clear flags
- Repeat `Load Page` command until the page is filled
- `Write Page` command
- Wait until one of the following interrupts[10]:
  - Sequence error interrupt
  - Protection error interrupt
  - Operation error interrupt
  - End of busy interrupt
- Error Flag Checks:
  - Fail if DMU_HCI_ERR.PVER flag
  - Fail if DMU_HCI_ERR.OPER = ‘1’
  - Fail if DMU_HCI_ERR.SQER = ‘1’
  - Fail if DMU_HCI_ERR.PROER = ‘1’
- For information about how to handle the errors refer to Error handling, see also the remarks given in the previous section about the handling of PVER.
- Recommended: read programmed content and evaluate it either CRC, error flags of the memory error correction circuit (ECC) or both
- Clear error flags either with “Clear Status” or by directly writing to DMU_HCI_CLRERR register

***Note**: The steps may also be done when using the CSRM command interface; in this case use the registers DMU_CSCI_STATUS, DMU_CSCI_ERR and DMU_CSCI_CLRERR.*

Timeout

The datasheet parameters (for example $t_{PRG_P256_B}$) can be used for a timeout check after the application of the `Write Page` command. To account for potential system related additional timing (for example command transfer from CPU via SRI to DMU) it is recommended to use: (datasheet value) + (5 µs) as timeout value.

Further notes

At the end of the write operation the busy flag of a bank is set back to "0" which indicates that all operations on the bank are finished and a read from the bank is possible again. The "end of busy" interrupt is asserted by the falling edge transition of the busy flag in DMU_HCI_STATUS or DMU_CSCI_STATUS.

However the command sequence might not yet been finished: for example there may be a write of FSI_COMMx registers ongoing. The real end of the command sequence execution is signaled by DMU_HCI_STATUS.REQDONE = ‘1’

Related information

Write page on page 696

10 prerequisite: all of these interrupts are enabled before the start of the write sequence

5.3.11 Erase operation hints

The following chapters describe applicable erase sizes and show an example sequence for erase, which also includes the use of status and error flags.

Related information

Erase logical sector range on page 707

5.3.11.1 Erase range sizes

PFLASH erase range sizes

Table 421 PFLASH erase range sizes

Size	Description	Size [Kbyte]
Minimum size	Logical Sector	16
Maximum size	Contiguous Range of Logical Sectors	512

***Note**: For the AURIX™ TC4xx products it is allowed that the range of logical sectors may cross a physical sector boundary.*

DFLASH erase range sizes

Table 422 DFLASH erase range sizes

Size	Description	Size [Kbyte]
Minimum size	Logical Sector	2
Maximum size	Contiguous Range of Logical Sectors	256

***Note**: For the AURIX™ TC4xx products it is allowed that the range of logical sectors may cross a physical sector boundary.*

***Note**: One UCB comprise also a logical sector with 2 Kbyte.*

5.3.11.2 Erase sequence

Recommended erase sequence

The following sequence is recommended for the erase operation.

- `Clear Status` command to clear flags
- `Erase Logical Sector Range` command
- information: the NVM sets DMU_HCI_STATUS.REQACK = '1' as soon as the NVM receives the first cycle of the command sequence
- Wait until DMU_HCI_STATUS.REQDONE = '1'
- Error Flag Checks:
  - Optional: Check for DMU_HCI_ERR.PVER flag (PFLASH: Erase counter program fail, would not block the erase)
  - Fail if DMU_HCI_ERR.EVER flag
  - Fail if DMU_HCI_ERR.OPER = '1'
  - Fail if DMU_HCI_ERR.SQER = '1'
  - Fail if DMU_HCI_ERR.PROER = '1'
- For information about how to handle the errors refer to Error handling
- Clear error flags either with "Clear Status" or by directly writing to DMU_HCI_CLRERR register

***Note**: The steps may also be done when using the CSRM command interface; in this case use the registers DMU_CSCI_STATUS, DMU_CSCI_ERR and DMU_CSCI_CLRERR.*

Alternative erase sequence using several interrupts

The following sequence is a possible alternative to the erase sequence. For this alternative several interrupts are used.

- `Clear Status` command to clear flags
- `Erase Logical Sector Range` command
- Wait until one of the following interrupts[11]:
  - Sequence error interrupt
  - Protection error interrupt
  - Operation error interrupt
  - End of busy interrupt
- Error Flag Checks:
  - Optional: Check for DMU_HCI_ERR.PVER flag (PFLASH: Erase counter program fail, would not block the erase)
  - Fail if DMU_HCI_ERR.EVER flag
  - Fail if DMU_HCI_ERR.OPER = '1'
  - Fail if DMU_HCI_ERR.SQER = '1'
  - Fail if DMU_HCI_ERR.PROER = '1'
- For information about how to handle the errors refer to Error handling
- Clear error flags either with "Clear Status" or by directly writing to DMU_HCI_CLRERR register

***Note**: The steps may also be done when using the CSRM command interface; in this case use the registers DMU_CSCI_STATUS, DMU_CSCI_ERR and DMU_CSCI_CLRERR.*

11 prerequisite: all of these interrupts are enabled before the start of the erase sequence

Timeout

The datasheet parameters (for example t_{RANGE_ERP}) can be used for a timeout check after the application of the `Erase Logical Sector Range` command. To account for potential system related additional timing (for example command transfer from CPU via SRI to DMU) it is recommended to use: (datasheet value) + (5 µs) as timeout value.

Further notes

At the end of the write operation the busy flag of a bank is set back to "0" which indicates that all operations on the bank are finished and a read from the bank is possible again. The "end of busy" interrupt is asserted by the falling edge transition of the busy flag in DMU_HCI_STATUS or DMU_CSCI_STATUS.

However the command sequence might not yet been finished: for example there may be a write of FSI_COMMx registers ongoing. The real end of the command sequence execution is signaled by DMU_HCI_STATUS.REQDONE = ‘1’

5.3.12 Abort

This section details the use of the abort command.

5.3.12.1 Command overview

This chapter describes the general concept of the abort feature.

Table 423 NVM abbreviations

Abort
A running NVM operation can be aborted. This means that the operation is terminated immediately . ***Note**: An aborted NVM operation might result in an undefined Flash memory content (with respect to the operation range of the aborted operation).* Software may abort a running NVM operation by writing 1_B to DMU_HCI_OCONTROL.ABORT (HOST command interface) or DMU_CSCI_OCONTROL.ABORT (CSRM command interface) ***Note**: Abort may need a certain time until read or another operation is possible. The maximum time is defined by the datasheet parameter t_{SPNDP} for PFLASH or t_{SPNDD} for DFLASH.*

5.3.12.2 Command interface states

The commands described above are possible for each command interface. The state of one command interface is independent from the state of the other command interface.

The state of a command interface can be modeled as *m* with *m* ϵ {*'idle'*, *'active'*}. The value of *m* denotes the current operation mode of the channel.

Table 424 Command interfaces stack and active or idle states

State	m (operation mode)	Description
A	Idle	• No abort command • No command is currently executed, any new command may be started
B	Active	• Abort is possible • A command is currently executed

Example: A running ERASE request on PFLASH Bank can be expressed by the state (active)

The following figure shows the state transition graph.

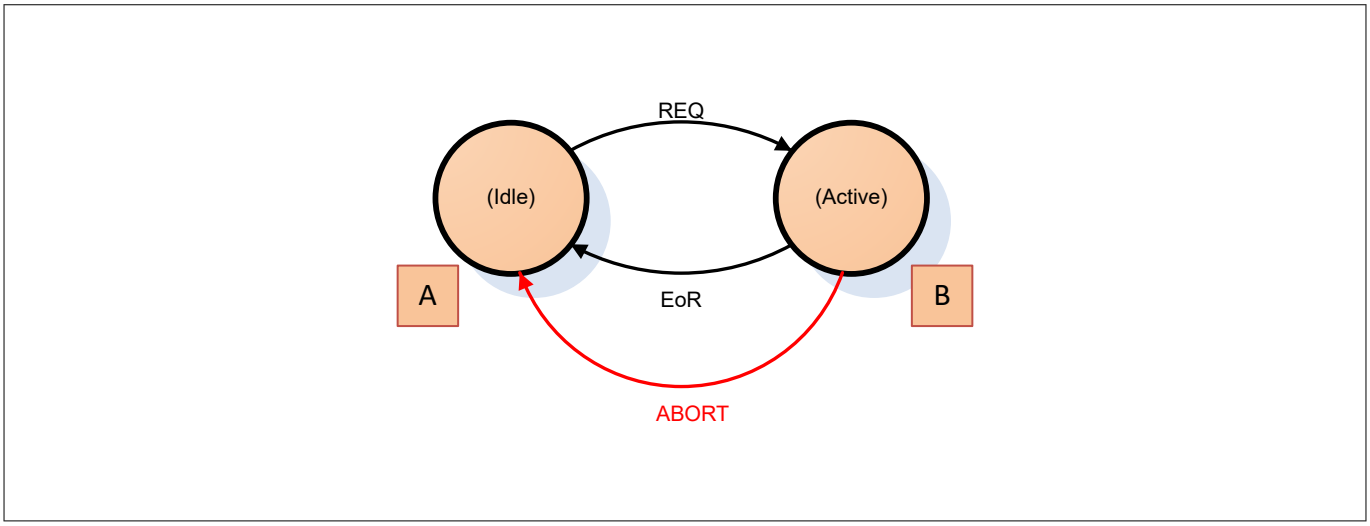


Figure 51 Command interface state transitions

Table 425 Abbreviations used in previous figure

Abbreviation	Description
REQ	Request for operation is applied on command interface (for example: Write Page sequence)
EoR	The operation is finished
ABORT	The operation is aborted

The new feature of AURIX™ TC4xx NVM, namely *Abort* is also shown in the figure. As already mentioned in a previous section an *Abort* terminates any running operation. Thus, an *Abort* creates a state transition from state *('active')*to state *('idle')*. These transitions are highlighted in red.

5.3.12.3 Abort operation

List of operations which may be aborted

The following operations may be aborted:

- `Write Page, Write Page Once, Write Burst, Write Burst Once`
- `Erase Logical Sector Range`
- `Verify Erased Page, Verify Erased WL, Verify Erased Logical Sector Range`
- `User Margin Count, User Vth Count, Content Count`

Erroneous abort requests are detected and prevented by the DMU.

Abort

Software may abort a NVM operation by writing 1_B to DMU_HCI_OCONTROL.ABORT.

- The command interface checks if there is a flash operation requested or ongoing
  - If not, then the abort request is ignored and DMU_HCI_OCONTROL.ABORT is cleared
- The command interface checks if the ongoing flash operation may be aborted
  - If not, then the DMU:
    - Reports an abort error (DMU_HCI_ERR.ABER = 1_B)
    - Waits for the flash operation to be finished
    - Then clears the abort request (DMU_HCI_OCONTROL.ABORT = 0_B)
- In the case of a valid abort request, the DMU executes the abort and proceeds as follows:
  - The DMU waits for the NVM operation to reach a state where an abort can be done
  - The running operation is aborted and the bank which was under operation is set back to read state
  - Then the busy flag of the bank is cleared
  - DMU_HCI_OCONTROL.ABORT is cleared
- After a command has been aborted:
  - A new NVM operation may be started

***Note**: The abort of a `Write` command leads to an exit of the PAGE MODE*

CSRM command Interface

The Abort operation is also possible on the CSRM command interface for operations which are operated by this interface. Abort on the CSRM command interface is completely independent from the Abort on the Host command interface. For operation please use the following registers of the CSRM Command Interface:

- DMU_CSCI_OCONTROL.ABORT
- DMU_CSCI_ERR.ABER

5.3.12.4 Abort application hints

Abort operation => undefined memory state

Aborting a write or erase operation leaves the target memory range of this operation in an **undefined state**. Reading this range may deliver unpredictable results.

Minimum time between start and abort recommendation

There is no minimum time necessary if a restart of the operation after the abort is not planned. If however the abort is used to start a new operation "B" and after the completion of "B" there is a restart of the aborted operation planned then the time between the start of a Flash operation and the abort request shall be ~10 ms. Timings mentioned above are valid for f_{FSI} = 100 MHz and need to be scaled up accordingly when using a lower FSI frequency.

Abort an ongoing NVM operation

- Check if the corresponding BUSY flag has already been cleared
  - If BUSY = 0_B:
    - the command was finished already
- Request abort with the control flag DMU_HCI_OCONTROL.ABORT set to 1_B
- Wait until the corresponding BUSY flag clears
- Now, a new Flash operation is allowed

Attention**: **by using the abort feature the complete information about the aborted command is deleted. Instead a new command sequence is necessary if the aborted operation need to be done.

CSRM command Interface

The same guidance is applicable for CSRM Command Interface using the following registers:

- DMU_CSCI_OCONTROL.ABORT
- DMU_CSCI_ERR.ABER

5.3.13 UCB handling

This chapter details the handling of UCBs during startup and re-programming.

Attention**: **The Erase operation of the UCB takes more time than the 'normal' Erase operation of a logical sector in the DFLASH(EEPROM) region, for UCB erase please refer to the datasheet timing $t_{SECTOR_ERDM_LC_UCB}$.

Related information

UCB on page 668

5.3.13.1 UCB confirmation code and UCB state evaluation

The state of a UCB is determined by the evaluation of the confirmation codes.

Table 426 UCB confirmation codes

Confirmation code	Value	Description
ERASED	0000 0000$_H$	Erased State The UCB confirmation code was erased. The behavior is the same as for the **ERRORED** state. ***Note**: The **ERASED** state is considered a special case of **ERRORED** and will not be used in the rest of the UCB state descriptions.*
UNLOCKED	4321 1234$_H$	Delivery State The UCB confirmation code is programmed with the **UNLOCKED** value.
CONFIRMED	57B5 327F$_H$	Operational State The UCB confirmation code is programmed with the **CONFIRMED** value. ***Note**: The **UNLOCKED** value can be over programmed with the **CONFIRMED** value.*
ERRORED	Others	Errored State The UCB confirmation code stored is not the **CONFIRMED** or **UNLOCKED** value.

The erased state is considered as **ERRORED** to ensure that a complete loss of UCB content is not leading to the installation of wrong configurations and wrong or no protections.

The device is delivered with UCBs in **UNLOCKED** state. After user configuration, the UCBs shall be changed to **CONFIRMED** state by over-programming the **UNLOCKED** confirmation code value with the **CONFIRMED** confirmation code value (keeping the 4 bytes following the confirmation code to 0000 0000$_H$ to ensure an ECC clean page).

In fact the state of the UCBs is determined by evaluating a pair of confirmation codes called ORIG and COPY confirmation codes.

Single UCB

A single UCB contains full content and a pair of ORIG and COPY confirmation codes in the same UCB.
The state of the UCB is evaluated as follow :

Table 427 Evaluation of the pair of confirmation codes in single UCB

ORIG confirmation code	COPY confirmation code	UCB state
UNLOCKED without uncorrectable ECC error	any state	Only ORIG is evaluated. The UCB is considered **UNLOCKED** and the content can be used.
CONFIRMED without uncorrectable ECC error	any state	Only ORIG is evaluated. The UCB is considered **CONFIRMED** and the content can be used.
ERRORED or uncorrectable ECC error detected	**UNLOCKED**	ORIG and COPY are evaluated. The UCB is considered **UNLOCKED** and the content can be used. An *original error (ORIER)* is set.
ERRORED or uncorrectable ECC error detected	**CONFIRMED**	ORIG and COPY are evaluated. The UCB is considered **CONFIRMED** and the content can be used. An *original error (ORIER)* is set.
ERRORED or uncorrectable ECC error detected	**ERRORED** or uncorrectable ECC error detected	ORIG and COPY are evaluated The UCB is considered **ERRORED** and the content can not be used. An *original error (ORIER)* is set.

Note the *original error (ORIER)* is visible per channel in the respective registers : DMU_HCI_ERR for HOST UCBs or DMU_CSCI_ERR for CSRM UCBs.

***Note**: to increase UCB's security, it is recommended to set both ORIG and COPY confirmation codes to CONFIRMED to avoid unexpected changes to the UCB's state when UCERs occur on the ORIG confirmation code.*

Dual UCB

A dual UCB is a pair of UCBs containing redundant content. The UCBs are called ORIG and COPY UCBs. In this case the ORIG UCB contains a ORIG confirmation code and the COPY UCB the COPY confirmation code.

The state of the dual UCB is evaluated as follow :

Table 428 Evaluation of the pair of confirmation codes in dual UCBs

ORIG confirmation code from the ORIG UCB	COPY confirmation code from the COPY UCB	UCB state
UNLOCKED without uncorrectable ECC error	any state	Only ORIG is evaluated. The UCB is considered **UNLOCKED** and the content of the ORIG UCB can be used. ***Note**: COPY UCB has full read and write access, regardless of its state.*
CONFIRMED without uncorrectable ECC error	any state	Only ORIG is evaluated. The UCB is considered **CONFIRMED** and the content of the ORIG UCB can be used. ***Note**: when COPY UCB is ERRORED, it can be fully recoverable, providing that the password protection is disabled on ORIG UCB.*
ERRORED or uncorrectable ECC error detected	**UNLOCKED**	ORIG and COPY are evaluated. The UCB is considered **UNLOCKED** and the content of the COPY UCB can be used. ***Note**: ORIG UCB is fully recoverable.*
ERRORED or uncorrectable ECC error detected	**CONFIRMED**	ORIG and COPY are evaluated. The UCB is considered **CONFIRMED** and the content of the COPY UCB can be used. ***Note**: ORIG UCB is fully recoverable, providing that the password protection is disabled on COPY UCB.*
ERRORED or uncorrectable ECC error detected	**ERRORED** or uncorrectable ECC error detected	ORIG and COPY are evaluated The UCB is considered **ERRORED** and the content can not be used.

***Note**: to increase UCBs' security, it is recommended to set the confirmation code of both ORIG and COPY UCBs to CONFIRMED to avoid unexpected changes to the UCB's state when UCERs occur on the UCB ORIG confirmation code.*

Related registers

After evaluation of the confirmation codes while startup the DMU will store the status of the HOST UCBs in the DMU_GP_HOST_CONFIRMA, DMU_GP_HOST_CONFIRMB, DMU_GP_HOST_CONFIRMC and the status of the CSRM UCBs in the DMU_GP_CSRM_CONFIRMA, DMU_GP_CSRM_CONFIRMB, DMU_GP_CSRM_CONFIRMC.

Note after an update of the content or the state of a UCB a reset is necessary for the DMU to recover the new information.

***Note**: while in delivery state and during the programming phase of the UCBs, any reset is to be avoided until all UCBs have been programmed to prevent inconsistencies in the life cycle state.*

Related information

UCB on page 668

Sector protection configuration and registers on page 775

5.3.13.2 UCB COPY and ORIG update procedure

Dual UCB with password: ORIG and COPY reprogramming

The data stored in the ORIG and COPY of a UCB pair shall be identical. If there is a need to change the data then the following sequence shall be followed:

- Check if UCB ORIG is either **UNLOCKED** or **CONFIRMED**:
  - The data and *password* are recovered from the UCB ORIG
- If UCB ORIG is **ERRORED** then:
  - The data and *password* are recovered from the UCB COPY, but only if UCB COPY is either **UNLOCKED** or **CONFIRMED**
  - If UCB COPY was also **ERRORED** then there was a startup error
- Apply the *password* with the command sequence `Disable Protection`. This disables protection for both: ORIG and COPY UCBs
- Erase COPY UCB - the confirmation code is treated as **ERRORED** for the installation
- Program in COPY UCB the data and the confirmation code to **CONFIRMED**
- Erase ORIG UCB - the confirmation code is treated as **ERRORED** for the installation
  - After a reset the data and the confirmation code would be sourced now from the COPY UCB
- Program in ORIG UCB the data and the confirmation code to **CONFIRMED**
  - After a reset the data and the confirmation code will be sourced from the ORIG UCB
- Enabling of protection can be done either by:
  - Command sequence Resume Protection
  - Or MCU reboot (system reset)

The ORIG UCB is always evaluated. The above sequence ensures that the new data is confirmed in COPY before the ORIG is re-programmed. It avoids a scenario where the ORIG confirmation state is **ERRORED** and the COPY confirmation state is also **ERRORED**.

The following flowchart details the flow for use cases where one of both UCBs is errored.

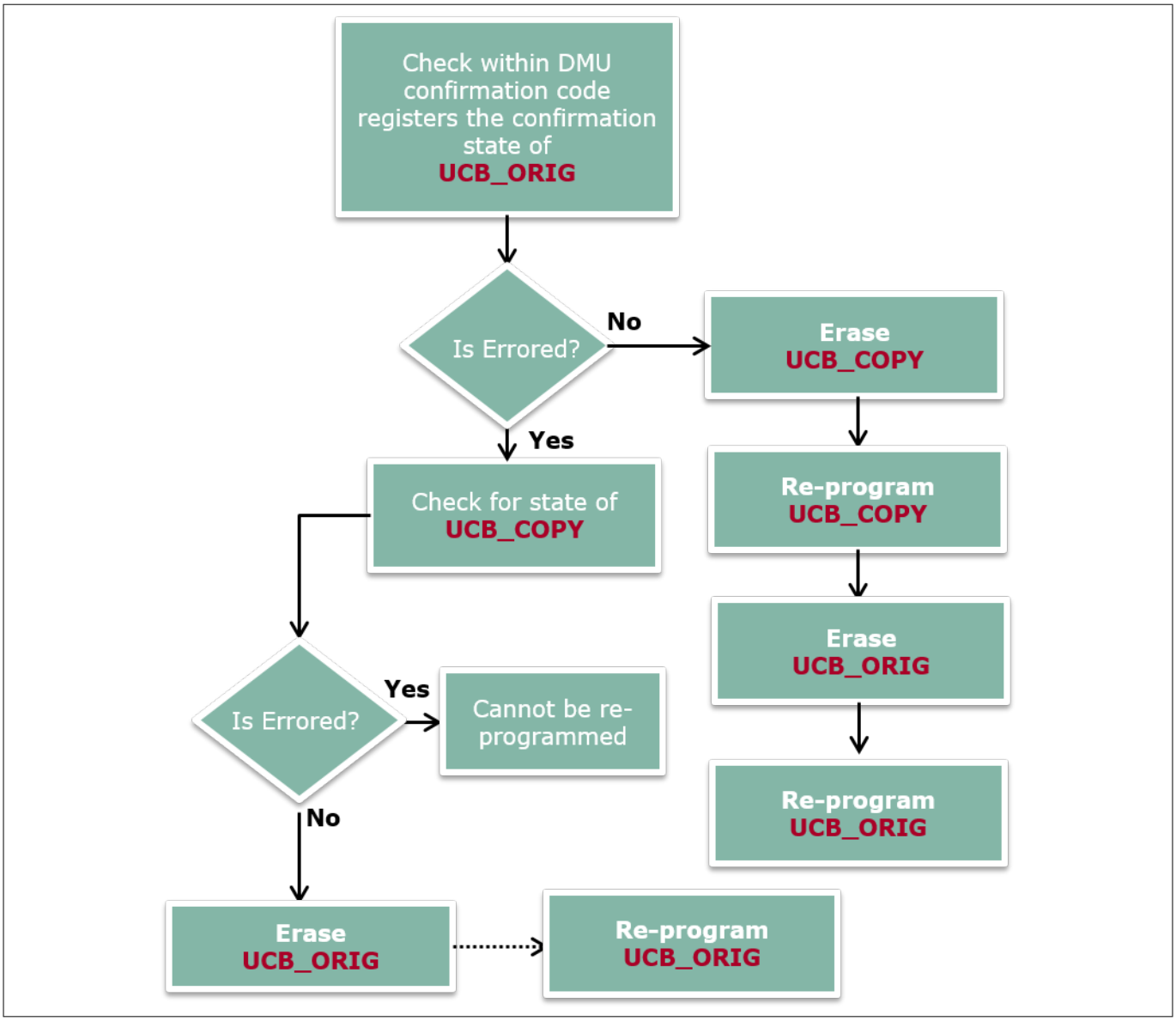


Figure 52 Dual password UCB update flow

Dual UCB without password: OTPx ORIG and COPY reprogramming

The normal use case of the UCB OTPx is different to the standard password protection UCBs: UCB OTPx are normally programmed in a sequential manner and the final protection setting is an OR function of all OTPxy protection sets.

There is no password protection but once a single OTPxy protection set gets his confirmation code to **CONFIRMED** programmed then this UCB OTPxy set cannot be programmed again any longer and the whole UCB OTPx cannot be erased anymore.

So as long as all OTPxy sets have confirmation code in **UNLOCKED** state it is possible to erase the UCB OTPx and re-program it.

Beside these constraints the flow of updating the UCB OTPx ORIG and COPY is the same as mentioned in the previous section for the password protected UCBs.

Related information

UCB on page 668

5.3.13.3 Delivery state

The MCU is delivered with the customer changeable UCBs in UNLOCKED state. The following table list all those UCBs which have the UNLOCKED confirmation code.

Table 429 Device state table

UCB Number	UCB Name	Delivery State
UCB0_09	UCB_RTC_BMHD0	UNLOCKED
UCB0_10	UCB_RTC_BMHD1	UNLOCKED
UCB0_11	UCB_RTC_BMHD2	UNLOCKED
UCB0_12	UCB_RTC_BMHD3	UNLOCKED
UCB0_15	UCB_USER_DATA0	UNLOCKED
UCB0_16	UCB_USER_DATA1	UNLOCKED
UCB0_17	UCB_RTC_USERCFG_ORIG	UNLOCKED
UCB0_18	UCB_RTC_USERCFG_COPY	UNLOCKED
UCB0_19	UCB_RTC_SWAP_ORIG	UNLOCKED
UCB0_20	UCB_RTC_SWAP_COPY	UNLOCKED
UCB0_21	UCB_RTC_FLASH_ORIG	UNLOCKED
UCB0_22	UCB_RTC_FLASH_COPY	UNLOCKED
UCB0_23	UCB_RTC_ECPRIO_ORIG	UNLOCKED
UCB0_24	UCB_RTC_ECPRIO_COPY	UNLOCKED
UCB0_25	UCB_RTC_PFLASH_OTP0_ORIG	UNLOCKED
UCB0_26	UCB_RTC_PFLASH_OTP1_ORIG	UNLOCKED
UCB0_27	UCB_RTC_PFLASH_OTP0_COPY	UNLOCKED
UCB0_28	UCB_RTC_PFLASH_OTP1_COPY	UNLOCKED
UCB1_1	UCB_DBGCS_ORIG	UNLOCKED
UCB1_2	UCB_DBGCS_COPY	UNLOCKED
UCB1_3	UCB_CS_BMHD0	UNLOCKED
UCB1_4	UCB_CS_BMHD1	UNLOCKED
UCB1_5	UCB_CS_BMHD2	UNLOCKED
UCB1_6	UCB_CS_BMHD3	UNLOCKED
UCB1_7	UCB_CS_PFLASH_OTP0_ORIG	UNLOCKED
UCB1_8	UCB_CS_ECPRIO_ORIG	UNLOCKED
UCB1_9	UCB_CS_PFLASH_OTP0_COPY	UNLOCKED
UCB1_10	UCB_CS_ECPRIO_COPY	UNLOCKED
UCB1_11	UCB_CS_FLASH_ORIG	UNLOCKED
UCB1_12	UCB_CS_SWAP_ORIG	UNLOCKED
UCB1_13	UCB_CS_SWAP_COPY	UNLOCKED

(table continues...)

Table 429 (continued) Device state table

UCB Number	UCB Name	Delivery State
UCB1_14	UCB_CS_FLASH_COPY	UNLOCKED
UCB1_15	UCB_CSRM_ORIG	UNLOCKED
UCB1_16	UCB_CSRM_COPY	UNLOCKED
UCB1_24	UCB_CS_USER_DATA0	UNLOCKED
UCB1_25	UCB_CS_USER_DATA1	UNLOCKED

***Note**: The detailed description of the UCB is available in the Firmware chapter of this manual.*

Attention**: **For a secure operation in field it is mandatory to set to CONFIRMED state at least one of those UCBs which cannot be changed back to UNLOCKED state by using Disable Protection Command and erase+write operation. Examples for UCBs being unchangeable with CONFIRMED state are: UCB_RTC_PFLASH_OTP0_ORIG or UCB_CS_PFLASH_OTP0_ORIG.

Related information

UCB for APU bypass on page 782
Device configuration on page 787

5.3.14 Access protection

This section details the use of protection for NVM memory ranges, command interfaces and configuration registers.

5.3.14.1 Operation protection

A change of the NVM content is done by performing a NVM operation like `Write Page` or `Erase Logical Sector Range`. It is necessary to protect the NVM memory regions against accidental and unauthorized changes. For this there are different protection mechanisms available, which are described in the following chapters.

If activated the operation protection mechanisms block **any** NVM operation on a specific memory region. This also includes operations which may not (directly) change the content of the NVM but are used for other purpose: for example data integrity check operations like `Verify Erased Wordline` or the `Replace Logical Sector` function. Only masters which are allowed to perform a change of the memory may analyze the memory or perform a replacement of a sector.

5.3.14.1.1 Operation protection overview

The following figure shows an overview of the three protection levels used during normal operation in field.

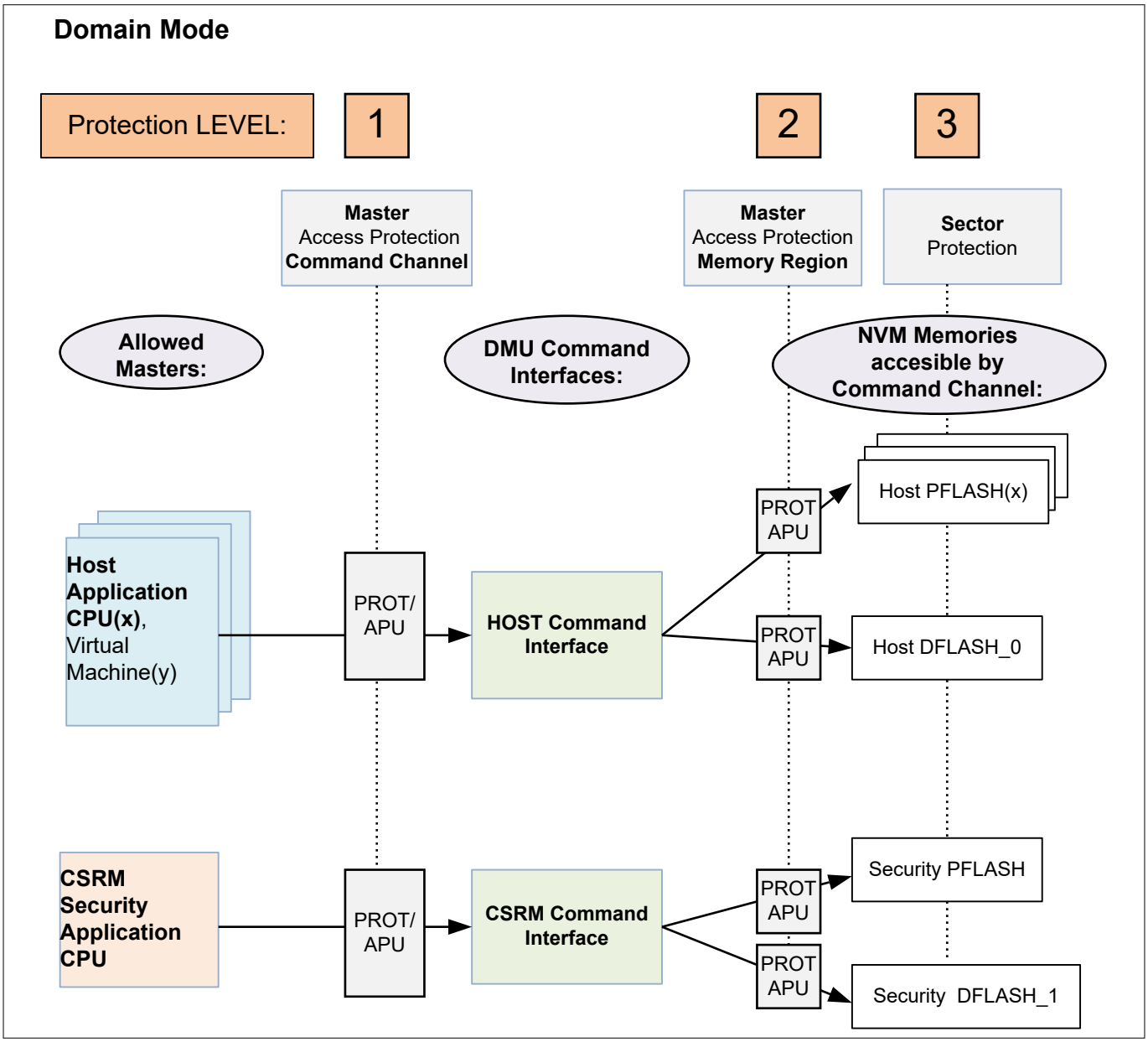


Figure 53 Protection levels for NVM operation

Table 430 Brief description of the three protection levels

Level	Name	Description	Usage examples
1	Master Acces Protection on Command Channel	Master → Resource[1)] Protection	One Command Channel exclusive for CSRM CPU operating on secure memories
2	Master Access Protection on Memory Region	Master → Memory Region Protection	Blocking of accidental PFLASH(Host) Writes, Access Protection to secure memories
3	Sector Protection	Master independent Protection of individual (PFLASH) logical sectors or regions	Write protection of selected PFLASH logical sectors independent from a master
		Master independent Protection of the whole PFLASH (HOST) region and DFLASH (HOST)	Write protection of the whole PFLASH with one password

1) Command Channel

The NVM shall perform a NVM Operation only if all of the three protection levels allow this:

1. Augmented Master ID matches with the IDs allowed by the APU of the command channel used
2. Augmented Master ID matches with the IDs allowed by the APU for the addressed memory region
3. Protection of all sectors of the memory region targeted by the NVM Operation is either not set or disabled

***Note**: in addition to the restrictions mentioned above there are certain memory regions which never can be changed by the application, for example: erase counter regions, certain UCBs with Infineon configuration, customer UCBs for Permanent Write (OTP) protection after program of confirmation code. Please refer to the description of a memory or function for a check whether a change of this memory is possible or not.*

Related information

Data flash banks on page 658
Error handling on page 685
Allocation of banks to command interfaces on page 680

5.3.14.1.2 Master access protection on command channels

Each command channel has a separate set of PROT and APUs to configure the access protection.

Table 431 Command channel access protections

Resource	Function	Number of PROT	Number of APU
HOST Command Channel	NVM Operation	1	1
HOST Command Channel Configuration Registers	Read or Write Register		
CSRM Command Channel	NVM Operation	1	1
CSRM Command Channel Configuration Registers	Read or Write Register		

There is a common protection for the **command channel** itself and its **configuration registers**.

The setting for WRITE access means:

- A master may perform a NVM Operation using this command channel
- A master may change the command channel configuration registers

The setting for READ access means:

- A master can read the command channel configuration registers

Upper or lower address region definition registers in the APU are not used here. The protection is valid for both: command channel and all configuration registers for this channel.

The following list contains examples for command channel configuration registers which are protected by this mechanism:

- Command configuration registers (for example program modes)
- Status registers (for example: busy)
- Program and erase error registers (for example: program verify error)
- Communication registers (for example: communication registers used for data integrity check commands)

Use Case: Different settings for register read and write

Within one domain (for example: HOST domain) there is normally no need to restrict the READ access to the registers of the command channel. On the other hand the WRITE access to registers is usually only needed for those masters which are also allowed to write a memory range and to use this command channel for the write operation to the memory range.

Different PROT/APU for HOST and CSRM command channel and register protection ensure that the read or write access to CSRM command channel configuration registers can be limited to secure masters.

5.3.14.1.3 Master access protection on memory regions

For each memory region there is an independent set of PROT/APU available; please refer to the following table.

Table 432 Memory protections by PROT/APU

Memory	Function	Number of PROT	Number of APU
PFLASH (HOST) Banks	NVM Operation	1	product dependent: = number of Banks
PFLASH (CSRM) Bank	NVM Operation	1	4
DFLASH (HOST)	Read and NVM Operation	1	8
UCB of DFLASH (HOST)	Read and NVM Operation		1
DFLASH (CSRM)	Read and NVM Operation	1	4
UCB of DFLASH (CSRM)	Read and NVM Operation		1

***Note**: The access protection for PFLASH (HOST and CSRM) READ is within the CPU; please refer to the description of the CPU.*

DFLASH and UCB protection

For the DFLASH (HOST) Bank there is one common PROT but different APUs:

- 8 APUs for the DFLASH (HOST) EEPROM Memory Range (APU upper and lower address registers can be used)
- 1 APU which is dedicated solely for the UCBs of the DFLASH (HOST), no address range definition necessary

For the DFLASH (CSRM) Bank there is one common PROT but different APUs:

- 4 APUs for the DFLASH (CSRM) EEPROM Memory Range (APU upper and lower address registers can be used)
- 1 APU which is dedicated solely for the UCBs of the DFLASH (CSRM), no address range definition necessary

Attention**: **In addition the UCB dependent access restrictions are still valid: if a UCB cannot be read or changed (for example UCB OTP in confirmed state) then the setting of PROT/APU is irrelevant. On the other hand if a read or write access to a UCB is possible then also the APU settings for the UCBs need to allow the master to do this.

Related information

DFLASH master read protection on page 769

5.3.14.1.4 Sector protection

For each logical sector of the PFLASH it is possible to set one of 3 different write protection levels. These protection levels are independent from the master accessing the memory: even if a master is allowed to access a memory range the NVM may refuse the request with protection error (PROER) if at least one logical sector has a write protection which forbids this operation.

The protection levels may be set by writing the information into a specific UCB. During startup and after application reset the protections will be read out from the UCB and be set into the protection registers.

The following table describes the 3 protection levels.

Table 433 Sector protection levels

Level	Blocked	Allowed	Disable Protection during run-time?
Write Protection	All `Erase` and `Write` commands `Replace Logical Sector` command	-	Yes, temporarily possible by `Disable Protection` command
Permanent Write Once Protection	All `Erase` commands All standard `Write` commands `Replace Logical Sector` command	`Write Once` commands (page and burst)	Not possible
Permanent Write Protection (synonym: OTP)	All `Erase` and `Write` commands `Replace Logical Sector` command	-	Not possible

Attention**: **the table above describes the possible operations and changes during runtime. The actual protection setting is defined by the UCB. If a UCB for Write Once Protection or Permanent Write Protection has a confirmation code written to CONFIRMED then this protection cannot be changed anymore

***Note**: the write once commands may also be applied to sectors without any sector protection set: in this case the command does the same function as it does on write-once protected sectors.*

***Note**: The sector protections are available for the PFLASH(HOST) and the PFLASH(CSRM) banks (see the related information for a description).*

PFLASH (HOST) global write protection

It is possible to disable the write access to **all** PFLASH(HOST) Banks. For this the UCB bit has to be programmed as described in the table below. The UCB will be recovered during the startup and after an application reset and the register bit will indicate if the global write protection is set.

Table 434 Register and UCB for global PFLASH read protection

Domain	Register	UCB	UCB_Offset_Address
HOST	DMU_GP_HOST_PFPROCON.RWPRO	DFLASH(HOST): UCB_RTC_FLASH	refer to UCB overview

The global write protection may be temporarily removed by using the `Disable Protection` command with the correct **UC** setting as described in the chapter "Command Sequences".

***Note**: the RWPRO bit shall be also be used if the read from PFLASH shall be blocked in case that the MCU startup is NOT from the internal Flash (please refer to the related information for more details).*

DFLASH (HOST) write protections

It is possible to disable the write access to the complete DFLASH(HOST) bank. For this the UCB bits have to be programmed as described in the table below. The UCB will be recovered during the startup and after an application reset and the register bits will indicate if the write protection is enabled.

Table 435 Register and UCB for global DFLASH read protection

Domain	Register	UCB	UCB_Offset_Address
HOST	DMU_GP_HOST_DFPROCON.WP	DFLASH(HOST): UCB_RTC_FLASH	refer to UCB overview
	DMU_GP_HOST_DFPROCON.RWPRO		

The write protection is enabled if WP or RWPRO is set to 1_B.

Difference between RWPRO and WP: the RWPRO is also the configuration for the DFLASH(HOST) read protection in case that the MCU startup is NOT from the internal Flash, whereas the WP is only a configuration of the write protection.

The write protection may be temporarily removed by using the `Disable Protection` command with the correct **UC** setting as described in the chapter "Command Sequences".

DFLASH (CSRM) write protections

It is possible to disable the write access to the complete DFLASH(CSRM) bank. For this the UCB bits have to be programmed as described in the table below. The UCB will be recovered during the startup and after an application reset and the register bits will indicate if the write protection is enabled.

Table 436 Register and UCB for global DFLASH read protection

Domain	Register	UCB	UCB_Offset_Address
CSRM	DMU_GP_CSRM_DFPROCON.WP	DFLASH(CSRM): UCB_CS_FLASH	refer to UCB overview

The write protection is enabled if WP is set to 1_B.

The write protection may be temporarily removed by using the `Disable Protection` command with the correct **UC** setting as described in the chapter "Command Sequences".

Related information

Error handling on page 685

Sector protection configuration and registers on page 775

PFLASH global read protection on page 767

DFLASH global read protection on page 771

5.3.14.1.5 Error signaling

If a master requests an access but is not allowed to do the following error signaling is available.

Table 437 Error signaling

Blocked access	Error
Master → Command Channel	Bus Error
Master → Command Channel - Configuration Register Change	Bus Error
Master → Memory Region	Protection Error (see DMU_HCI_ERR.PROER)

***Note**: Protection error on PFLASH(CSRM) is signaled as: DMU_CSCI_ERR.PROER.*

If a master is allowed by the access enable registers (APU) to perform an access but the target memory region has an additional (master independent) active **sector protection** there is also a *Protection Error (PROER)* set.

5.3.14.2 Read protection

The following chapters describe the master access protection on read from NVM memories and on the read configuration (for example wait state settings).

5.3.14.2.1 PFLASH master read protection

The PFLASH read access is controlled by the CPU to which the PFLASH Bank is allocated. Each CPU contains PROT/APU registers for the master access protection on the PFLASH read access.

Related information

PFLASH global read protection on page 767

5.3.14.2.2 PFLASH read configuration protection

The PFLASH read configuration settings can be split into two groups:

- Local PFLASH Bank specific settings like ECC ON and OFF control
- Global PFLASH settings like wait state settings

Local PFLASH bank specific read configuration protection

The local read configuration of a PFLASH Bank is done by PFRWB Flash configuration registers. The following overview lists some of the Bank specific critical read configurations for PFLASH.

- ECC correction ON and OFF
- ECC trap disable
- ECC status register clear
- Tight read margin settings for robustness checks

***Note**: This list contains only examples, for a list of registers please refer to the NVM PFRWB register overview.*

The access to these registers is protected by one PROT and one APU per CPU, please refer to the access protection description of the CPU.

Global PFLASH read configuration protection

There is a central setting of the PFLASH read wait states in the DMU. As a consequence it is necessary to have PROT/APU registers in the DMU for the master access protection on these settings. For this purpose are used the same PROT/APU as for the DFLASH read configuration protection.

Related information

DFLASH read configuration protection on page 769

5.3.14.2.3 PFLASH global read protection

It is possible to disable the read access to **all** PFLASH(HOST) Banks to prevent the readout of NVM content, for example by downloaded code. For this the UCB has to be programmed as described in the tables below. The UCB will be recovered during the startup and after application reset and the register bit will indicate if the global read protection is configured.

Table 438 Register and UCB for global PFLASH read protection

Domain	Register	UCB	UCB_Offset_Address
HOST	DMU_GP_HOST_PFPROCON.RWPRO	DFLASH(HOST): UCB_RTC_FLASH	refer to UCB overview

The register bit DMU_HCI_BCONTROL.DDFP shows the setting:

- DDFP = 0_B: Read access enabled
- DDFP = 1_B: Read access blocked

The following 2 conditions must be met to switch the global read protection on:

1. UCB_RTC_FLASH for DMU_GP_HOST_PFPROCON.RWPRO = 1_B
2. Startup Mode is NOT from internal Flash

The following table shows the possible settings. The next section explains how the protection may be disabled temporarily by the `Disable Protection` command.

Table 439 Global read protection based on RWPRO and internal boot

UCB RWPRO	Boot from Internal Flash	DDFP	Read possible ?	Comment
0	YES	0	YES	
0	NO	0	YES	Fab-configuration: to be used only for initial programming of the device
1	YES	0	YES	**Recommended Default**
1	NO	1	NO	Not recommended for productive devices

Disabling of global read protection if RWPRO = 1 and DDFP = 1

To allow read access it is necessary to change the DDFP register bit to 0_B. The following steps are necessary to do that:

1. Apply the `Disable Protection` command sequence
2. Change the DDFP register bit from 1_B to 0_B

All of the following command sequence must be issued to disable the global read protection:

- Disable the global read protection for all PFLASH HOST Banks
- Disable the global read protection for all PFLASH HOST Banks and for DFLASH HOST
- Disable the read protection and the write protection of a specific PFLASH Bank pAx or pBx

These options can be selected by the **UC** parameter of the `Disable Protection` command; for the description of the possible **UC** settings please refer to chapter Command Sequences.

Please note that even if the global protection is disabled there may be additional APU protections set.

Resume and lock of global read protection if RWPRO = 1

By setting the DDFP register bit to 1_B the read protection is enabled. The bit access can then be locked by the `Resume Protection` command.

Enabling global read protection if RWPRO = 1 and DDFP = 0

In case of RWPRO = 1 and boot from internal Flash the value of DDFP is 0_B to enable the read from PFLASH.
It is possible to activate the global read protection by setting DMU_HCI_BCONTROL.DDFP = 1.
After this a disable of the Global Read Protection is only possible with the Disable Protection Command as described above.

Change of Global Read Protection if RWPRO = 0

If RWPRO is 0 then the direct register access to DMU_HCI_BCONTROL.DDFP is possible to do the following changes:

- Change 0 → 1 (protection enabled)
- Change 1 → 0 (protection disabled)

Summary of direct HCI_BCONTROL register change

The following table summarizes the possibilities for a change of the DMU_HCI_BCONTROL.DDFP.

Table 440 Direct Change of DDFP

RWPRO	DDFP 0 → 1 ?	DDFP 1 → 0 ?
0	YES	YES
1	YES	after Disable Protection

Use of RWPRO for other purpose

The RWPRO configuration is also used to activate a global write protection for all PFLASH Banks which is effective in any case independent from the boot and access mode. The write protection may also be disabled temporarily by the `Disable Protection` command.

Notes on PFLASH(CSRM)

PFLASH(CSRM) does not have a configurable global read and write protection by RWPRO. Per reset the PFLASH(CSRM) has master access control set to CPUcs for read and write. It is also possible to configure sector write protections for the PFLASH (CSRM) per UCB.
In addition it is possible to block the PFLASH (CSRM) reading by setting the DMU_CSCI_BCONTROL.DDFP = 1_B.

***Note**: once set, DMU_CSCI_BCONTROL.DDFP can only be cleared by the next start up.*

Related information

Command sequences on page 689
PFLASH master read protection on page 766
DFLASH global read protection on page 771
Sector protection on page 763

5.3.14.2.4 DFLASH master read protection

The DFLASH read access is controlled by the DMU. For the master access protection on the DFLASH **read** access the same PROT/APU register sets as for the memory region **operation** protection are used.

These APU registers allow to configure the read and the write access by separate settings. Read access means the actual read from the DFLASH memory region and write access means the command operation on the DFLASH memory region (for example: `Write Page`).

Related information

Master access protection on memory regions on page 762
Data flash banks on page 658
DFLASH global read protection on page 771

5.3.14.2.5 DFLASH read configuration protection

The read configuration of DFLASH is done by DMU registers. The following overview lists some of the critical read configurations for DFLASH.

- ECC correction ON and OFF
- ECC trap disable
- ECC status register clear
- Tight read margin settings for robustness checks
- Wait state settings

***Note**: This list contains examples, for a list of registers please refer to the chapters about the functions and the chapter of DMU register overview for the description of the read and write access modes.*

For each DFLASH Bank (HOST, CSRM) there is a **separate** PROT/APU set as described in the table below.

Table 441 Access protection to DFLASH bank read configurations

Resource	Function	Number of PROT	Number of APU
DFLASH (0) (HOST) Read Configuration Registers	Read and write of registers	1	1
PFLASH (HOST) Wait State Setting (see section below)			
DFLASH (1) (CSRM) Read Configuration Registers	Read and write of registers	1	1
PFLASH(CSRM) Wait State Setting (see section below)			

***Note**: Wait state settings shall be configured during software startup and not be changed anymore. Therefore it is recommended to enable the access only during software startup and then lock any access to protect the settings against accidental changes.*

***Note**: APU allows to configure read access to read configuration registers for many actors whereas the write access may be limited to few masters.*

Use of DFLASH read configuration protection for PFLASH wait state setting protection

There is a central setting of the PFLASH read wait states in the NVM.DMU (see related information). As a consequence it is necessary to have master access protection on these settings. For this purpose the same PROT/APU sets are used as for the DFLASH read configuration protection (see table above).

Related information

DFLASH ECC on page 676
Configuring read access cycles on page 671
PFLASH read configuration protection on page 766

5.3.14.2.6 DFLASH global read protection

It is possible to disable the read access to the DFLASH(HOST) Bank to prevent the readout of NVM content, for example by downloaded code. For this the UCB bit has to be programmed as described in the tables below. The UCB will be recovered during the startup and after the application reset and the register bit will indicate if the global read protection is configured.

Table 442 Register and UCB for global DFLASH read protection

Domain	Register	UCB	UCB_Offset_Address
HOST	DMU_GP_HOST_DFPROCON.RWPRO	DFLASH(HOST): UCB_RTC_FLASH	refer to UCB overview

The register bit DMU_HCI_BCONTROL.DDFD shows the setting:

- DDFD = 0_B: Read access enabled
- DDFD = 1_B: Read access blocked

The following 2 conditions must be met to switch the global read protection on:

1. DMU_GP_HOST_DFPROCON.RWPRO = 1_B
2. Startup Mode is NOT from internal Flash

The following table shows the possible settings. The next section explains how the protection may be disabled temporarily by the `Disable Protection` command.

Table 443 Global read protection based on RWPRO and internal boot

UCB RWPRO	Boot from Internal Flash	DDFD	Read possible ?	Comment
0	YES	0	YES	
0	NO	0	YES	Fab-configuration: to be used only for initial programming of the device
1	YES	0	YES	**Recommended Default**
1	NO	1	NO	Not recommended for productive devices

Disabling of global read protection if RWPRO = 1 and DDFD = 1

To allow read access it is necessary to change the DDFD register bit to 0_B. The following steps are necessary to do that:

1. Apply the `Disable Protection` command sequence
2. Change the DDFD register bit from 1_B to 0_B

There are the following possibilities to disable the global read protection:

- Disable the global read protection for DFLASH HOST Banks

These options can be selected by the **UC** parameter of the `Disable Protection` command; for the description of the possible **UC** settings please refer to chapter Command Sequences.

Please note that even if the global protection is disabled there may be additional APU protections set.

Resume and lock of global read protection if RWPRO = 1

By setting the DDFP register bit to 1_B the read protection is enabled. The bit access can then be locked by the `Resume Protection` command.

Enabling global read protection if RWPRO = 1 and DDFD = 0

In case of RWPRO = 1 and boot from internal Flash the value of DDFD is 0_B to enable the read from PFLASH.
It is possible to activate the global read protection by setting DMU_HCI_BCONTROL.DDFD = 1.
After this a disable of the Global Read Protection is only possible with the Disable Protection Command as described above.

Change of Global Read Protection if RWPRO = 0

If RWPRO is 0 then the direct register access to DMU_HCI_BCONTROL.DDFD is possible to do the following changes:

- Change 0 → 1 (protection enabled)
- Change 1 → 0 (protection disabled)

Summary of direct HCI_BCONTROL register change

The following table summarizes the possibilities for a change of the DMU_HCI_BCONTROL.DDFD.

Table 444 Direct Change of DDFD

RWPRO	DDFD 0 → 1 ?	DDFD 1 → 0 ?
0	YES	YES
1	YES	after Disable Protection

Use of RWPRO for other purpose

The RWPRO configuration is also used to activate a write protection for the DFLASH(HOST) which is effective in any case independent from the boot and access mode. The write protection may also be disabled temporarily by the `Disable Protection` command.

Notes on DFLASH(CSRM)

DFLASH(CSRM) does not have a configurable global read and write protection by RWPRO. Per reset the DFLASH(CSRM) has master access control set to CPUcs for read and write. It is also possible to configure a write protection for the DFLASH (CSRM) per UCB.
In addition it is possible to block the DFLASH (CSRM) reading by setting the DMU_CSCI_BCONTROL.DDFD = 1_B.

***Note**: once set, DMU_CSCI_BCONTROL.DDFD can only be cleared by the next start up.*

Related information

PFLASH global read protection on page 767
Command sequences on page 689
DFLASH master read protection on page 769
Sector protection on page 763

5.3.14.2.7 Error signaling

If a master requests an access but is not allowed to do, then the following error signaling is available:

Table 445 Error signaling

Blocked access	Error signal
Read from PFLASH	see chapter PFLASH protection
Change PFLASH read configuration in FLASHCON registers	
Change PFLASH wait state setting	Bus error
Change DFLASH read configuration	Bus error
Read from DFLASH	Bus error

5.3.14.3 Protection configuration and registers

This section details the registers which contain the protection information. For those registers which are recovered from a UCB during device startup there is also a list of the UCBs.

5.3.14.3.1 Master access protection registers

The PROT and APU registers are all contained in the DMU_PROT register group.

The following table documents the relation between the PROT registers, ACCEN registers and the protected resources :

Table 446 Protection scheme

PROT	Related ACCEN	Related resource
PROTHCI	ACCENHCI	HOST command interface and registers
PROTCSCI	ACCENCSCI	CSRM command interface and registers
PROTHRDC	ACCENHRDC	HOST read configurations registers
PROTCSRDC	ACCENCSRDC	CSRM read configurations registers
PROTHPF	ACCENHPF	HOST PFLASH
PROTHDF	ACCENHDF ACCENHDFUCB	HOST DFLASH HOST UCB
PROTCSPF	ACCENCSPF	CSRM PFLASH
PROTCSDF	ACCENCDF ACCENCDFUCB	CSRM DFLASH CSRM UCB

Please see the address map overview of the registers to know which registers is protected by which ACCEN/APU. After a reset PROT and ACCEN registers have default values which can be changed by the masters having default access. In delivery state the ACCEN/APU are bypassed due to a setting within the APUBYPASS set of the UCB UCB_CS_PFLASH_OTP0. Further details about the life cycle and how to permanently enable the use of ACCEN/APU can be found in the chapter Master access protection life cycle.

Related information

Master access protection life cycle on page 779

5.3.14.3.2 Sector protection configuration and registers

During the device startup the DMU reads the sector protection settings from UCBs into the protection registers (which are read-only). The following table shows which registers and UCBs are used for the 3 different sector protections:

Table 447 Registers and UCBs for sector protection

Protection	Registers	UCB
Write protection	DMU_GP_HOST_PFPROCONWPpXn	UCB_RTC_FLASH_ORIG, UCB_RTC_FLASH_COPY
Permanent write once protection	DMU_GP_HOST_PFPROCONPWOPpXn	UCB_RTC_PFLASH_OTP0_ORIG, UCB_RTC_PFLASH_OTP0_COPY, UCB_RTC_PFLASH_OTP1_ORIG, UCB_RTC_PFLASH_OTP1_COPY
Permanent write protection	DMU_GP_HOST_PFPROCONPWPpXn	

The following table explains the indexing "pXn" of the registers:

Table 448 Indexing of bank registers

Index	Description
p	Number of CPU
X	= A or =B for Bank A or Bank B of a CPU
n	Number of physical sector: for a 2 MB Bank n = (0..3) for 4 physical sectors with each having 32 logical sectors (=> 32 bits of register)

The following table shows an example for the write protection registers for the Bank B of CPU(1):

Table 449 Example of DMU_GP_HOST_PFPROCONWPpXn registers for Bank B of CPU(1) if the Bank has 2 MB size

Register	write protection setting for
DMU_GP_HOST_PFPROCONWP1B0	logical sectors 0..31
DMU_GP_HOST_PFPROCONWP1B1	logical sectors 31..63
DMU_GP_HOST_PFPROCONWP1B2	logical sectors 64..95
DMU_GP_HOST_PFPROCONWP1B3	logical sectors 96..127

Table 450 Example of DMU_GP_HOST_PFPROCONWPpXn registers for Bank B of CPU(1) if the Bank has 1 MB size

Register	write protection setting for
DMU_GP_HOST_PFPROCONWP1B0	logical sectors 0..31
DMU_GP_HOST_PFPROCONWP1B1	logical sectors 31..63

***Attention**: The register description chapter always shows all possible registers for a maximum bank size of 2 MB. For Banks with a smaller size there are no registers for those logical sectors which do not exist. The UCB storage area for those registers is "do not care" and not recovered during the startup.*

Configuration of write protection in the UCB_RTC_FLASH_ORIG and UCB_RTC_FLASH_COPY

The UCB_RTC_FLASH_ORIG and UCB_RTC_FLASH_COPY is password protected. Once the password protection is disabled by using the `Disable Protection` command with the correct password it is possible to erase or write content to the UCB. Please refer to the UCB address map overview chapter for the offset address of UCB and the offset addresses of UCB content, confirmation codes and passwords.

Note that the Disable Protection command does two things:

- Disables sector specific write protection of the Bank addressed with the command settings
- Disables the write protection of the UCB_RTC_FLASH

Configuration of permanent write once and permanent write protection in the UCB_RTC_PFLASH_OTP0_ORIG, UCB_RTC_PFLASH_OTP0_COPY, UCB_RTC_PFLASH_OTP1_ORIG and UCB_RTC_PFLASH_OTP1_COPY

The handling of the permanent write once protection and permanent write protection is different compared to the write protection handling. There is no password for the permanent write once and permanent write protection. Instead there are different "OTP_sets(y) with y = 1..6".

- UCB_RTC_PFLASH_OTP0 contains the OTP_sets 1-3
- UCB_RTC_PFLASH_OTP1 contains the OTP_sets 4-6

Each OTP_set contains storage place to define for all logical sectors of all PFLASH Banks:

- DMU_GP_HOST_PFPROCONPWPpXn
- DMU_GP_HOST_PFPROCONPWOPpXn

Each OTP_set has its own confirmation code. As long as the confirmation codes of all OTP_sets of a UCB are in **UNLOCKED** state it is possible to erase this UCB. If one or more OTP_sets have a confirmation code **CONFIRMED** it is no longer possible to erase the UCB. A write operation of a OTP_set(y) is possible if the confirmation code of this OTP_set is **UNLOCKED**. If the confirmation code of a OTP_set is **CONFIRMED** then a write is not possible.

The PFPROCON registers contain the OR'ed information of all OTP_set. If a OTP_set(y) is **CONFIRMED** then its content cannot be longer changed and the resulting permanent write (once) protection is set forever. By the combination of several OTP_sets it is only possible to add additional logical sectors to the protection. The following table shows some examples of the lifecycle for the UCB_RTC_PFLASH_OTP0.

Table 451 Example for OTP set life cycle of UCB_RTC_PFLASH_OTP0

OTP_set1	OTP_set2	OTP_set3	Erase UCB?	Write OTP sets?
UNLOCKED	UNLOCKED	UNLOCKED	yes	1,2,3
CONFIRMED	UNLOCKED	UNLOCKED	no	2,3
CONFIRMED	CONFIRMED	UNLOCKED	no	3

The status (**UNLOCKED**, **CONFIRMED**, **ERRORED**) information of each OTP_set**y** is available in the register DMU_GP_HOST_CONFIRMC.

HOST OTP protection installation

Initial protection

- If UCB_RTC_PFLASH_OTP0_**ORIG**:OTP_**Set_0** is **UNLOCKED** or **CONFIRMED** then:
  - GP_HOST_PFPROCONPWPpXn and GP_HOST_PFPROCONPWOPpXn registers are initialized to the UCB_RTC_PFLASH_OTP0_**ORIG**:OTP_**Set_0** configuration set
- Else if UCB_RTC_PFLASH_OTP0_**ORIG**:OTP_**Set_0** is **ERRORED** and UCB_RTC_PFLASH_OTP0_**COPY**:OTP_**Set_0** is **UNLOCKED** or **CONFIRMED** then:
  - GP_HOST_PFPROCONPWPpXn and GP_HOST_PFPROCONPWOPpXn registers are initialized to the UCB_RTC_PFLASH_OTP0_**COPY**:OTP_**Set_0** configuration set

Subsequent protection

- If UCB_RTC_PFLASH_OTP0_**ORIG**:OTP_**Set_1** is **UNLOCKED** or **CONFIRMED** then:
  - GP_HOST_PFPROCONPWPpXn and GP_HOST_PFPROCONPWOPpXn registers are set to the **OR** (UCB_RTC_PFLASH_OTP0_xxxx:OTP_**Set_0**, UCB_RTC_PFLASH_OTP0_ORIG:OTP_**Set_1**) configuration set
  - Note: OTP0_xxxx means either OTP0_ORIG or OTP0_COPY (see "Initial Protection")

- Else if UCB_RTC_PFLASH_OTP0_**ORIG**:OTP_**Set_1** is **ERRORED** and UCB_RTC_PFLASH_OTP0_**COPY**:OTP_**Set_1** is **UNLOCKED** or **CONFIRMED** then:
  - GP_HOST_PFPROCONPWPpXn and GP_HOST_PFPROCONPWOPpXn registers are set to the **OR** (UCB_RTC_PFLASH_OTP0_xxxx:OTP_**Set_0**, UCB_RTC_PFLASH_OTP0_**COPY**:OTP_**Set_1**) configuration set
  - Note: OTP0_xxxx means either OTP0_ORIG or OTP0_COPY (see "Initial Protection")
- If subsequent OTP_Set **y** is **UNLOCKED** or **CONFIRMED** then:
  - GP_HOST_PFPROCONPWPpXn and GP_HOST_PFPROCONPWOPpXn registers are set to the OR of all preceding OTP configuration sets including Set **y**

Error:

- If (for a common y) both OTP_set**y** of the ORIG and the COPY UCB are **ERRORED** then:
  - All sector protection bits of GP_HOST_PFPROCONPWPpXn and GP_HOST_PFPROCONPWOPpXn registers are set to 1_B which means that all sectors are protected

CSRM specific write protection registers and UCB

The write protection settings for the PFLASH Bank of the CSRM are stored in the following registers and UCB.

Table 452 Registers and UCBs for sector protection

Protection	Registers	UCB
Write protection	DMU_GP_CSRM_PFPROCONWPn	UCB_CS_FLASH_ORIG and UCB_CS_FLASH_COPY
Permanent write once protection	DMU_GP_CSRM_PFPROCONPWOPn	UCB_CS_PFLASH_OTP0_ORIG and UCB_CS_PFLASH_OTP0_COPY
Permanent write protection	DMU_GP_CSRM_PFPROCONPWPn	

There is only one PFLASH Bank for the CSRM and therefore only one index n is necessary. For the 1 MB PFLASH(CSRM) Bank n is 0 or 1.

The UCB_CS_PFLASH_OTP0_ORIG and UCB_CS_PFLASH_OTP0_COPY contain the OTP sets 1-3 (for CSRM PFLASH there are only 3 OTP sets available).

The status (**UNLOCKED**, **CONFIRMED**, **ERRORED**) information of each OTP_set **y** is available in the register DMU_GP_CSRM_CONFIRMC.

The configuration of CSRM OTP UCBs and the installation of the information from the UCBs into the registers is similar to the way described above for the HOST UCBs.

Related information

UCB on page 668

UCB confirmation code and UCB state evaluation on page 750

Sector protection on page 763

5.3.14.4 Master access protection life cycle

Master access protections are disabled at the delivery state to support an initial configuration of the NVM memory regions. During the code development phase it is possible to activate and deactivate the protections for debugging purpose. For a usage in field a secure protection can be permanently activated, for example the access to the security memories.

Related information

UCB on page 668
Master access protection registers on page 774

5.3.14.4.1 Overview life cycle

The following figure is a simplified view of the standard MCU life cycle.

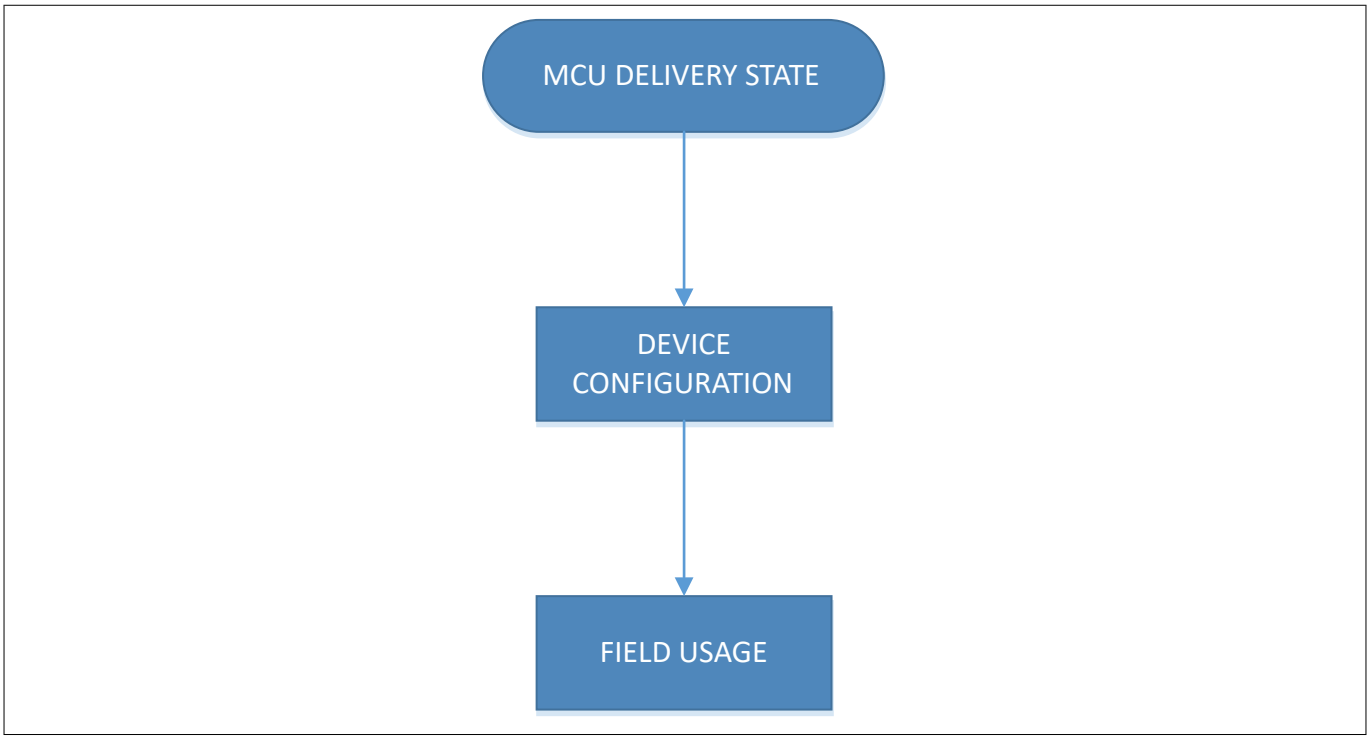


Figure 54 Standard MCU life cycle

Delivery state

In the delivery state, the APU protections are bypassed as described in the following chapter.

Device configuration

The device configuration is done at customer production environment. During this phase the bypass of the protection can be switched off by changing the UCB entry APUBYPASS in the UCB_CS_PFLASH_OTP0 to APU_USE so that the device is ready for field usage.

Field usage

The device has a safe and secure basic configuration after MCU startup or after Reset. Based on this, the application host software and the security software can do a refinement of the APU protections for the HOST and for the secure memories during their startup phase.

5.3.14.4.2 Delivery state APU bypass

The UCB entry APUBYPASS of the UCB_CS_PFLASH_OTP0 in DELIVERY state has the following content and confirmation code:

Table 453 Delivery state

UCB APUBYPASS confirmation code	UCB APUBYPASS content	Effect
UNLOCKED	APU_BYPASS	**1.** UCB may be changed **2.** APU protection is not used (this means it is bypassed)

Bypass of all APUs means:

- All masters have READ access to all NVM regions
- All masters can perform a NVM Operation to all NVM regions

The following restrictions apply under all circumstances (therefore there are also valid in the APU BYPASS situation):

- No read or write access to configuration sector
- No write access to erase counter
- No read or write access to certain UCBs (please refer to UCB description)

Attention**: **in DELIVERY state, the owner of the PROT for the security memories is the CPUcs, this means that a change of the APU (or the PROT) is only possible by a code executed on the CPUcs (either from secure SRAM or from PFLASH(CSRM)).

The purpose of this bypass mode is to enable an initial programming and read access to all memories before the access is locked to a subset of masters. The following chapters describe how this initial bypass state can be changed to support safe and secure operation by protecting the memories from unauthorized access by using the PROT and APU mechanisms.

The following figure shows a sketch of the bypass of the APU of the command channels and the memory region.

Attention**: **The BYPASS is only at the APU of the command channel and the memory region. In the delivery state the sector protection of the Banks is not set but once these sector protections are set they become valid. Since the sector protections are independent from the PROT/APU protections the BYPASS of the APU protections does not change the sector protection.

During the startup of the NVM the content of the UCB entry APUBYPASS is read and the following register field is set accordingly: DMU_PROT_APUBYPASS.BYPASS.

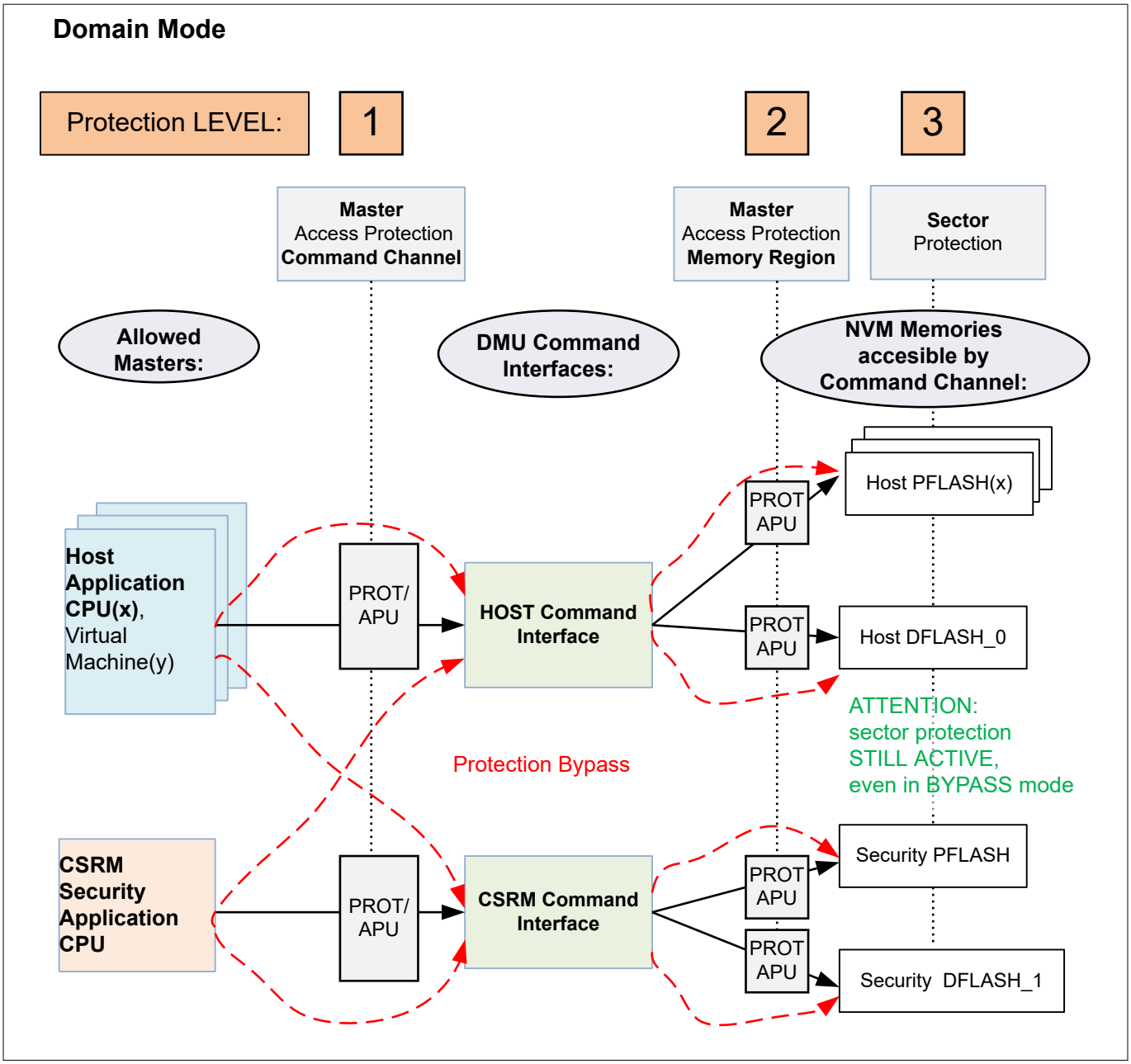


Figure 55 Bypass of APU

Related information

UCB for APU bypass on page 782

5.3.14.4.3 UCB for APU bypass

Overview

There is a UCB entry necessary to define if the APUs for the memories are used or bypassed. The UCB entry is named APUBYPASS and is part of the UCB_CS_PFLASH_OTP0. After UCB recovery the DMU register DMU_PROT_APUBYPASS contains the protection state APU_BYPASS or APU_USE.

***Note**: The access to the UCB_CS_PFLASH_OTP0 is if the APU is used because this UCB is part of theDFLASH(CSRM) UCB region.*

The UCB has a content and a place for status information. The status of the UCB may be:

- ERRORED:
  - UCB content errored
  - UCB confirmation code uncorrectable errored
  - invalid confirmation code values written
- UNLOCKED: unlock code is read
- CONFIRMED: confirmation code is read

The content of the UCB may have 2 states:

- APU_BYPASS
- APU_USE

Table 454 Definition of page for the UCB content

UCB entry APUBYPASS content	Pattern for one page (64 bit)
APU_BYPASS	DMU checks Byte 0: 55_H , Rest: 0
APU_USE	DMU checks Byte 0: FF_H ; overall pattern has to be 4 x $FFFF_H$ to get ECC valid 8 Byte word (can be achieved by over-programming the APU BYPASS pattern)

The UCB state is ERRORED if one of the following conditions occurs:

- Uncorrectable error when reading state or content
- Content different to the patterns mentioned in the previous table

APUBYPASS UCB State and Content Truth Table

The consequences of the UCB state and the UCB content are described in the following truth table.

Table 455 UCB entry APUBYPASS state and content => NVM APU use or bypass

UCB state	UCB content	=> NVM APU protection	=> UCB Change possible?
ERRORED	"do not care"	used	no
UNLOCKED	APU_BYPASS	bypassed	yes
	APU_USE	used	yes, but access may be restricted to CPUcs
	ECC uncorrectable error or any other pattern than APU_BYPASS	used	
CONFIRMED	APU_BYPASS	bypassed	NO, UCB content is OTP
	APU_USE	used	
	ECC uncorrectable error or any other pattern than APU_BYPASS	used	

***Attention**: If the UCB state for the APUBYASS is set to CONFIRMED then a change of this configuration is not possible anymore. Therefore for field usage the UCB content must be set to APU_USE before the confirmation code is set to CONFIRMED to make sure that the APU protections become effective.*

***Attention**: The UCB entry APUBYPASS is part of the UCB_CS_PFLASH_OTP0; therefore changes of this APUBYPASS entry may be limited to program only if one of the OTP sets are already confirmed. Please refer to the section "Use case scenarios" within this chapter.*

***Attention**: Once the APUBYPASS content is written to APU_USE (0xFF) then a change of this content is only possible by an erase of the UCB. If the erase of the UCB is not possible anymore due to an CONFIRMED OTP set then this APUBYPASS content becomes already fixed (even if the actual confirmation code is still UNLOCKED.*

Use case scenarios for the UCB_CS_PFLASH_OTP0

The UCB UCB_CS_PFLASH_OTP0 contains the following information:

- OTP sets: settings for OTP (permanent write) protection of PFLASH(CSRM) logical sectors
- APUBYPASS entry

Each OTP set and the APUBYPASS entry have their own confirmation code. This enables to lock each information independently from the other information.

If one (or several) of the OTP sets or the APUBYPASS entry have a confirmation code CONFIRMED then the following rules apply:

- It is not possible anymore to erase the UCB_CS_PFLASH_OTP0
- It is not possible anymore to program a UCB set content which has the confirmation code CONFIRMED
- It is possible to program a UCB set entry (content and confirmation code) which has the confirmation code UNLOCKED

The following figure describes possible use cases during the life cycle (please note that the table is only an example and describes a functional view, please refer to the UCB memory map for the number and the location (addresses) of the UCB entries).

Action/State	OTP set 1 content	OTP set 1 Confirmation code	OTP set 2 content	OTP set 2 Confirmation code	APUBYPASS content	APUBYPASS Confirmation code	=> UCB Overall State	=> PROG UCB possible ?	=> ERASE UCB possible ?
DELIVERY State	erased	UNLOCKED	erased	UNLOCKED	APU_BYPASS	UNLOCKED	UNLOCKED	YES	YES
Prog OTP_1 content	OTP-data	UNLOCKED	erased	UNLOCKED	APU_BYPASS	UNLOCKED	UNLOCKED	YES	YES
Prog Confirmation code OTP 1	OTP-data	CONFIRMED	erased	UNLOCKED	APU_BYPASS	UNLOCKED	CONFIRMED	only OTP set 2 and APUBYPASS	NO
Prog APUBYPASS content	OTP-data	CONFIRMED	erased	UNLOCKED	APU_USE	UNLOCKED	CONFIRMED	only OTP set 2	NO
Read Fail	uncorrectable data	wrong code or uncorrectable data	x	x	x	x	ERRORED	NO	NO

Figure 56 UCB_CS_PFLASH_OTP0 scenarios

***Note**: In the figure above the change as a result of an action is marked red.*

Development phase at customer

The following figure describes the different states of the UCB_APU during the life cycle.

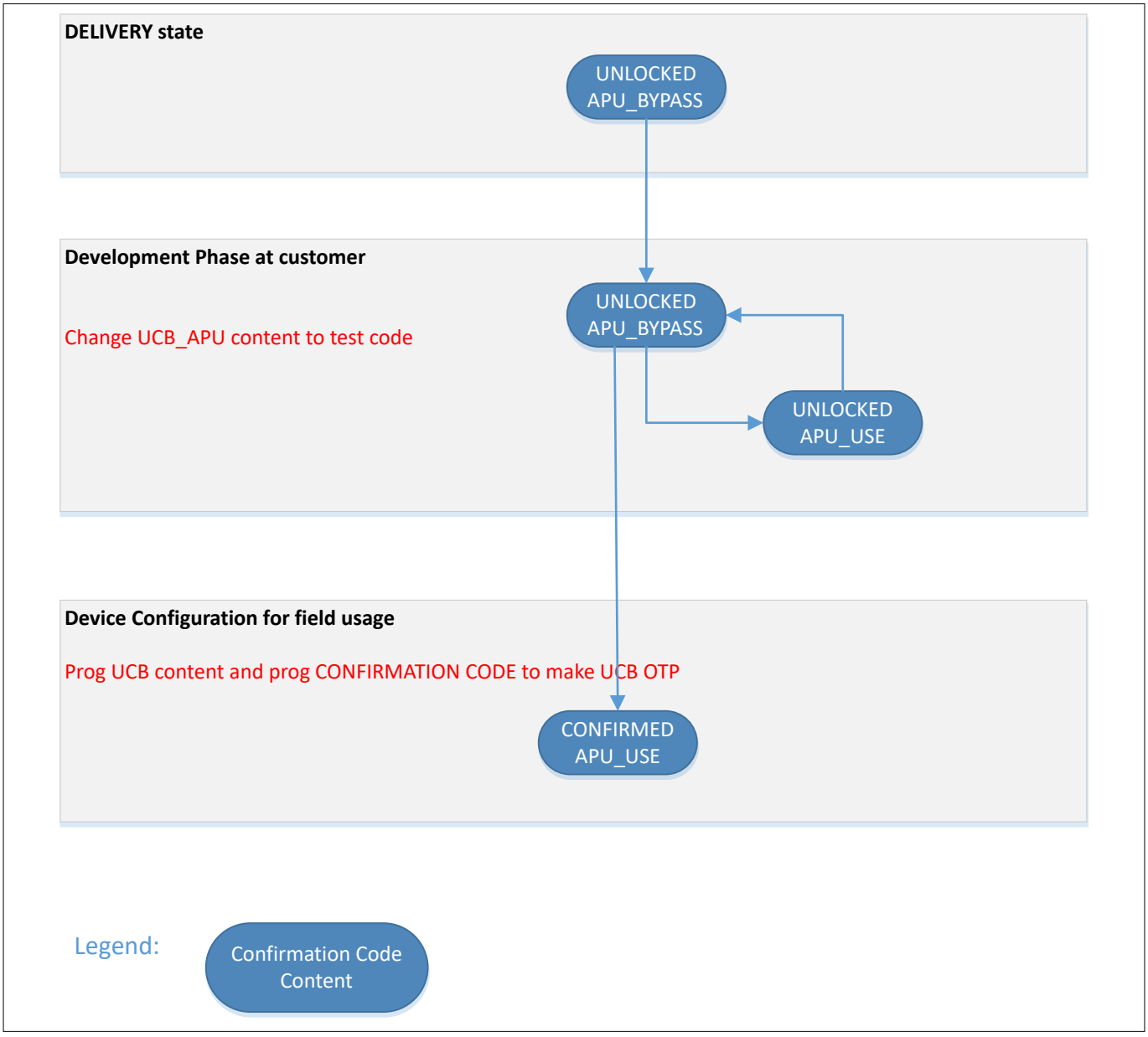


Figure 57 States of UCB_APU during the life cycle

For the development phase at customer, the UCB entry APUBYPASS may be programmed to UNLOCKED and its content be programmed to either APU_BYPASS or APU_USE to test both configurations. If the UCB content is APU_USE then the master access protection to the APU_UCB is active which means that a change of the UCB can only be done by:

- CPUcs executing code from PFLASH(CSRM) which rewrites the UCB entry APUBYPASS
- CPUcs having configured the APU for DFLASH(CSRM) so that another master can change the UCB entry APUBYPASS

***Note**: The delivery state of the microcontroller has all customer changeable UCBs in **UNLOCKED** state. In this delivery state the APU is automatically bypassed independent from the content of the APUBYPASS UCB. The content of the APUBYPASS UCB is relevant only if at least one UCB is **CONFIRMED**, because then the delivery state is left.*

***Note**: For the development phase it is recommended to set a UCB in **CONFIRMED** state which still can be changed, for example the setting of the UCB_RTC_FLASH (sector write protection) to **CONFIRMED** still allows a change of the write protection settings by the use of the Disable Protection command with the correct password.*

Related information

UCB on page 668
Delivery state APU bypass on page 780
Delivery state on page 756

5.3.14.4.4 Device configuration

The device configuration is done at customer production environment. During this phase, the read and write access to all memories is possible as long as the UCB APUBYPASS content is APU_BYPASS.

After the final programming of code and data, the UCBBYPASS content may be changed to APU_USE.

After a program of the CONFIRMATION code to the UCB APUBYPASS this UCB will be locked forever and cannot be changed anymore.

The following tables assume:

1. APUBYPASS is in UNLOCKED state
2. MCU shall be configured for use in field

MCU PROT/APU initial device configuration

Table 456 HOST application: Command channel and memories

State and transition	Command channel(HOST)		PFLASH(HOST)		DFLASH(HOST)		UCB APUBYPASS entry: UCB state UCB content
	PROT	APU	PROT	APU	PROT	APU	
RESET Values	ALL	READ: ALL WRITE: CPU0,CBS	ALL	READ:ALL WRITE:CPU0,CBS	ALL	READ: ALL WRITE:CPU0,CBS	
UCB Recovery							UNLOCKED APU_BYPASS
After MCU Startup	ALL	BYPASS	ALL	BYPASS	ALL	BYPASS	UNLOCKED APU_BYPASS
Device Configuration: UCB_Change							CONFIRMED APU_USE
After Reset and MCU Startup	ALL	READ: ALL WRITE: CPU0,CBS	ALL	READ: ALL WRITE: CPU0,CBS	ALL	READ: ALL WRITE: CPU0,CBS	CONFIRMED APU_USE

Table 457 Abbreviations and terms

Abbreviation and term	Description
ALL	For PROT: ODEF = 0_B, for APU: all masters enabled
READ: ALL WRITE: CPU0,CBS	For APU: APU set so that READ is possible for all masters, WRITE is ONLY possible for CPU0 and Cerberus
BYPASS	APU is not used, memory access possible to all masters

Table 458 Secure application: Command channel and memories

State and transition	Command channel(CSRM)		PFLASH(CSRM)		DFLASH(CSRM)		UCB APUBYPASS entry: UCB state UCB content
	PROT	APU	PROT	APU	PROT	APU	
RESET Values	CPUcs	CPUcs	CPUcs	CPUcs	CPUcs	CPUcs	
UCB Recovery							UNLOCKED APU_BYPASS
After MCU Startup	CPUcs	BYPASS	CPUcs	BYPASS	CPUcs	BYPASS	UNLOCKED APU_BYPASS
Device Configuration: UCB_Change							CONFIRMED APU_USE
After Reset and MCU Startup	CPUcs	CPUcs	CPUcs	CPUcs	CPUcs	CPUcs	CONFIRMED APU_USE

Table 459 Abbreviations and terms

Abbreviation and term	Description
CPUcs	For PROT: CPUcs is the owner of PROT, for APU: CPUcs only has access to command channel and memories
BYPASS	APU is not used, memory access possible to all masters

The tables above only mean the transition by programming the UCB. In addition, there would be initial startup code be programmed into the HOST PFLASH and the CSRM PFLASH memories. This startup code is always executed after RESET and MCU startup and may do a specific setting of the APU registers according to the software structure being used. This is described in the next chapter.

In addition to the configurations described above there will also be configuration of the other UCBs to confirmed state which will determine the read and write access to these UCBs together with the applicable PROT/APU protections.

The following sequence is recommended:

1. Program PFLASH code into PFLASH(HOST) and PFLASH(CSRM)
2. Program initial DFLASH data into DFLASH(HOST) and DFLASH(CSRM)
3. Program content and confirmation codes to the user UCBs (see UCB documentation)
4. Program content and confirmation code to the UCB entry (APUBYPASS) which is part of the UCB_CS_PFLASH_OTP0
5. Device is ready for field usage

Related information

Delivery state on page 756

5.3.14.4.5 Field usage

Overview

The following tables describe a typical use case of a MCU which has been configured for safety and security during production and which is run in field.

The description focus on the NVM Operation like program or erase to a memory range; commonalities and differences with regards to a read access to the memory are explained at the end of the chapter.

MCU PROT/APU configuration during field usage

Table 460 Host application: Command channel and memories

State and transition	Command channel(HOST)		PFLASH(HOST)		DFLASH(HOST)		UCB APUBYPASS entry: UCB state UCB content
	PROT	APU	PROT	APU	PROT	APU	
After MCU Startup	ALL	READ:ALL WRITE:CPU0,CBS	ALL	READ:ALL WRITE:CPU0,CBS	ALL	READ:ALL WRITE:CPU0,CBS	CONFIRMED APU_USE
Host Software: Initialization	owner_safety	authorized_safety and authorized_eeprom	owner_safety	authorized_safety	owner_eeprom	authorized_eeprom	CONFIRMED APU_USE

Table 461 Abbreviations and terms

Abbreviation and term	Description
ALL	for PROT: ODEF = 0_B, for APU: all masters enabled
READ:ALL WRITE:CPU0,CBS	for APU: APU set so that READ is possible for all masters, WRITE is only possible for CPU0 and Cerberus.
owner_safety	for PROT: Application Start Software defines the owner for the safe configuration
authorized_safety	for APU: Application Start Software or owner of safe application defines which CPU, VM or PRS has access
owner_eeprom	for PROT: Application Start Software defines the owner for the eeprom configuration
authorized_eeprom	for APU: Application Start Software or owner of eeprom application defines which CPU, VM or PRS has access

Table 462 Secure application: Command channel and memories

State and transition	Command channel(CSRM)		PFLASH(CSRM)		DFLASH(CSRM)		UCB APUBYPASS entry: UCB state UCB content
	PROT	APU	PROT	APU	PROT	APU	
After MCU Startup	CPUcs	CPUcs	CPUcs	CPUcs	CPUcs	CPUcs	CONFIRMED APU_USE
Secure Software: Initialization	CPUcs	authorized_ security	CPUcs	authorized_ security	CPUcs	authorized_ security	CONFIRMED APU_USE

Table 463 Abbreviations and terms

Abbreviation and term	Description
CPUcs	For PROT: CPUcs is the owner of PROT, for APU: CPUcs only has access to command channel and memories
authorized_security	For APU: Secure Application Start Software may extend or limit further the access to (part of) security memories

Safety: Protections against accidental PFLASH programming

Accidental change of PFLASH(HOST) memories by programming or erasing must be avoided for a safe operation.

An easy way to achieve this is to block all masters for an NVM Operation access to PFLASH(HOST) memories during normal operation.

In case that a program or erase operation is needed for a sub range of the PFLASH(HOST):

- The safety owner changes the APU
  - Grant write access to the software component doing the PFLASH(HOST) change
  - On a dedicated memory range

This scheme is a replacement of the use of "safety_endinit" scheme of TC3xx. An example application may be to allow the prog or erase access to an (inactive) bank for SOTA updates or the prog or erase access to a PFLASH(HOST) Bank range during offline state for incremental updates.

PROT/APU for read access

The description in the previous sections is for a NVM Operation like `Write` or `Erase`. The principal scheme of PROT/APU setting for *reading* is the same but the following differences need to be taken into account:

- Command channel is not necessary for *reading*
- There are different PROT/APU sets to protect PFLASH *reading*
- PROT/APU for PFLASH (HOST and CSRM) *reading* is within the CPU

PROT/APU for read configuration registers

The description in the previous sections is for a NVM Operation like Prog or Erase. The principal scheme of PROT/APU setting for READ and WRITE of the Read Configuration registers is the same but the following differences need to be taken into account:

- Read configuration registers are independent from the command channel configuration registers

- There are different PROT/APU sets to protect the READ configuration registers
- A part of the Read configuration registers is accessible by CPU and therefore the PROT/APU for those registers is within the CPU

5.3.15 EEPROM emulation with DFLASH

The Data Flash can be used for the emulation of an EEPROM or the storage of "quasi static" data, which is seldom changed. This chapter describes parameters, flags and additional commands to achieve a reliable data storage.

EEPROM emulation purpose

The term “EEPROM Emulation” designates an algorithm with the following features:

- It maps a logical EEPROM memory address to a physical Flash address
- It increases the effective endurance by spreading the EEPROM write accesses over a larger range of Flash memory
- It ensures that all Flash cells incur a similar number of cycles independent of the update frequency of the EEPROM data (“wear leveling”)
- It manages the allocation of EEPROM data to Flash ranges so that stale data can be erased

Handling of quasi static data

For the AURIX™ TC4xx products a logical sector within the Data Flash does not get disturbed during the program or erase operations within other Data Flash ranges. Thus, 'quasi-static' data may be placed in the Data Flash without the need to include this in the EEPROM emulation data (which could cost storage place and update cycles).

However the retention of Data Flash cells may be lower then those of Program Flash cells (due to other algorithms being used).

Therefore it is necessary to use the following measures for quasi static data:

1. Definition of quasi-static data:
   - less than 1000 updates
2. Permanent partitioning of DFLASH EEPROM necessary:
   a. EEPROM region for data updated less than the maximum allowed number of cycles per logical sector (parameters N_{E_CSRM} and N_{E_EEPROM})
   b. Quasi static data region for data updated less than 1000 times
3. Before the start of product usage the partitioning must be fixed for the whole lifetime:
   - It is not allowed to change an EEPROM region with more than 1000 cycles done into a quasi static data region
   - It is not allowed to change a quasi static data region into an EEPROM region (quasi static data region only allows less than 1000 updates)
4. Use the `Erase Logical Sector Range` command with 'quasi static data' option as described below:
   - Set the FSI COMM0 register to 0x05
   - Start the `Erase Logical Sector Range` command
   - Set the FSI COMM0 register to 0x00
5. Use the `Write Page` or `Write Burst` command with 'quasi static data' option as described below:
   - Set the FSI COMM0 register to 0x05
   - Start the `Write Page` or `Write Burst` command
   - Set the FSI COMM0 register to 0x00

Attention**: **The Erase and Write commands with the 'quasi static data' option shall only be used for the low-cycled (< 1000 updates) data.

Attention**: **The Erase operation with the 'quasi static data' option takes more time than the 'normal' Erase operation, see the datasheet timings $t_{SECTOR_ERDM_LC_UCB}$ for one logical sector and $t_{RANGE_ERDM_LC}$ for the maximum allowed erase size. The abbreviation "LC" in the parameter name stands for "low cycle".

***Note**: For UCB erasing and programming the use of the 'quasi static data' option is not necessary because the NVM automatically detects a UCB operation and adapts the sequence accordingly (an unintentional use of the option would not do harm). The erase and write operation on a UCB uses the same sequence as the erase and write sequence on a 'quasi static data" logical sector (therefore the parameter name $t_{SECTOR_ERDM_LC_UCB}$ contains also "UCB").*

***Note**: During the maximum 1000 updates of quasi static data the probability of a failure is low; nevertheless it is possible to perform a sector replacement if a logical sector cannot be erased or programmed. For this purpose the Replace Logical Sector command can be used.*

FSI_COMMx register addresses

There are separate FSI_COMMx registers for the HOST and for the CSRM command interface:

- FSI_HOST.COMMx
- FSI_CSRM.COMMx

These COMMx registers do not belong to the standard DMU register region. To access those registers please refer to:

Table 464 FSI COMMx registers

Address	Chapter
Register Base Address FSI_HOST.COMM0	System address map
Register Base Address FSI_CSRM.COMM0	
Register offset addresses for COMMx Registers Note: The offset addresses for FSI_HOST.COMMx and for FSI_CSRM.COMMx are the same	Register Overview - FSI

Related information

Error handling on page 685

5.3.15.1 Segmentation and parameters

Definition of parameters

The DFLASH supports "multi-sector" EEPROM emulation. Each of these "EEPROM sectors" may consist of a range of Flash logical sectors. Usually an "EEPROM sector" is erased completely during the EEPROM emulation; this is possible as long as the total size of the "EEPROM sector" is lower or equal to the maximum erase size of the Data Flash.

The number of erase operations in the Data Flash is limited by the two parameters shown in the table below:

Table 465 Limits for number of erase operations

Parameter HOST and CSRM	Description
NE_EEPROM or *NE_CSRM*	Total number of erase operations on a logical sector of DFLASH(HOST)(0) EEPROM or DFLASH(CSRM)(1)
NERD0S or *NERD1S*	Total number of erase operations on DFLASH(HOST) (0) or DFLASH(CSRM) (1) over lifetime

The parameter *NE_EEPROM* defines the maximum allowed erase operations per cell and must be observed in all cases. It is defined per logical sector but it is also valid for an "EEPROM sector", because the "EEPROM sector" is defined as a logical sector range which is always erased completely using the `Erase Logical Sector Range` command.

The parameter *NERD0S* is not memory cell related but relate to the memory periphery. Each erase operation uses high internal voltages, which induce a small stress on transistors in periphery switches or other circuits of the memory periphery. The overall accumulated stress within lifetime must not exceed a certain limit. Therefore the total number of erase operations must be limited.

Impact of parameters for different DFLASH segmentations

For the AURIX™ TC4xx the following formula is valid: *NERD0S* = 8 x *NE_EEPROM*.

This means that if DFLASH is split in 8 "EEPROM sectors", the total number of erase operations for each "EEPROM sector" is equal to *NE_EEPROM* and the total number of erase operations on the entire DFLASH is equal to *NERD0S*.

A division of the DFLASH region into a number of "EEPROM sectors" different than 8 is always possible, but the following equations must be observed:

- Number of erase operations of each "EEPROM sector" ≤ *NE_EEPROM*
- Number of "EEPROM sectors" **x** number of erase operation on each "EEPROM sectors" ≤ *NERD0S*

If the number of “EEPROM sectors” is less than 8, the limiting constrain is always *NE_EEPROM*..

If the number of “EEPROM sectors” is greater than 8, the limiting constraint is always *NERD0S*.

It is also possible to define "EEPROM sectors" of different sizes and with different number of updates as long as the limits *NE_EEPROM* and *NERD0S* are not violated.

Please note that the maximum size of each "EEPROM sector" must be less or equal the maximum allowed erase size for Data Flash.

Example for a "two sector emulation"

Example: If the 1 MB DFLASH is divided into two EEPROM regions; then the size of each EEPRPOM region would be 512 KB. Considering that the maximum erase size of the Data Flash is 256 KB, each EEPROM region needs two erase operations to be fully erased. For this reason, this case has the same behavior as the case where the DFLASH is divided in 4 “EEPROM sectors”.

Examples of possible segmentations

The following table shows an overview of different segmentations.

Table 466 Examples of possible segmentations for a 1 MB Data Flash EEPROM region

Number of "EEPROM_sectors"	Size of "EEPROM_sector"	Max number of "EEPROM sector" erase operations
4	256 KB	NE_EEPROM
8	128 KB	NE_EEPROM
16	64 KB	NERD0S/16

***Note**: In case of four "EEPROM_sectors" the parameter NE_EEPROM becomes the limiting constraint.*

***Note**: In case of sixteen "EEPROM_sectors" the parameter NERD0S is the limiting constraint.*

***Note**: Same constraints apply to the DFLASH(CSRM), for which the parameters NE_CSRM and NERD1S have to be used.*

5.3.15.2 Robust EEPROM emulation

A key requirement for an EEPROM emulation algorithm is the reliability of the stored data. This chapter provides information about possible faults and how to prevent and repair them.

Fault types and resulting failures

A memory array may have:

- Bitline oriented faults
- Wordline oriented faults
- Sector oriented faults

Bitline oriented faults mean a limited number of failing bits within the read data. The error correction circuit (ECC) can correct up to 3 failing bits within 64 bit and can detect all possible 4 failing bit patterns and an arbitrary amount of uncorrectable (multibit) errors with a detection coverage ~90%.

Wordline and sector oriented faults may result in a failure of:

- Reading from Wordline
- Write (Page or Burst) on Wordline
- Erase Logical sector

The ECC mechanism cannot correct wordline or sector oriented failures.

Wordline or sector oriented failures are most likely to occur during the high voltage conditions present during writing or erase operations. A hard breakdown of the following may occur:

- oxide between cell nodes
- oxide between contacts
- metal layer

Repair function

The AURIX™ TC4xx products offer a logical sector repair for the Data Flash: a failing logical sector may be replaced by a free redundant logical sector by the `Replace Logical Sector` command.

The `Replace Logical Sector` command needs to be started with the address of the failing logical sector; it replaces the complete logical sector (in Data Flash 2 KB, four wordlines) by an (erased) redundant logical sector, which is accessible at the same logical address after the command execution.

Repair after an erase operation

To detect a failure of an erase operation the check of the *Erase Verify Error (EVER)* flag is necessary (register for HOST command interface: DMU_HCI_ERR, register for CSRM command interface: DMU_CSCI_ERR).

Before repairing a logical sector it is necessary to ensure that no other reason for the erase failure exists: a stable voltage supply (VDDEXT) during the erase operation is mandatory.

It is recommended to repeat the erase operation at least one time. This may help to shift (potentially few) remaining cells to the target erase state.

If the *Erase Verify Error (EVER)* persists, then a repair of the failing logical sector shall be done, because otherwise write and read operations on the wordlines of the failing sector will likely fail. Furthermore a failing erase may have left the wordlines of the concerned logical sector in an undefined (erroneous) state or an old state with old data. The repair by the `Replace Logical Sector` command resolves these issues by providing an erased functional logical sector at the same address.

Usually during an erase in the Data Flash a contiguous range of n logical sectors is erased with one execution of the `Erase Logical Sector Range` command. It is necessary to identify the failing logical sector. This can be done by erase each logical sector in the range separately.

By doing so it is ensured that all the other logical sectors are erased sufficiently and the failing logical sector address is known.

The repair of a failing logical sector can then be done by performing the `Replace Logical Sector` command with the address of the failing sector.

***Note**: in order to speed up the operation it is possible to operate the Erase Logical Sector commands on half of the original region and continue on the failing region similar to a "binary search".*

***Note**: The erase command sequence checks the state of a logical sector before the start of the command execution. The actual erase does not start if a logical sector is already erased. Thus, unnecessary stress and execution time are avoided.*

Fault mitigation after a write operation

Based on the used cell voltage conditions it is expected that the likelihood of a hard fail during erase is significantly higher than the likelihood of a hard fail during a write operation. A quantitative assessment of this is only possible after product qualification with sufficient statistics.

It is therefore recommended that the EEPROM emulation algorithm is able to work with faulty wordlines and protect against data loss during a write operation.

The following mechanisms are possible to mitigate the effects of a faulty wordline.

Before writing a page in a wordline, save the content of those pages (of the wordline) which already contain valid data to SRAM.

To detect a failure of a write operation it is necessary to:

- Check the *Program Verify Error (PVER)* flag (see DMU_HCI_ERR or DMU_CSCI_ERR)
- Read pages of the wordline and compare with target content
- Correctable 1-bit and 2-bit errors may be accepted

***Note**: Erase and write operations have a built-in fault tolerance to prevent the repetition of needless operations in the case of a few "stuck-at programmed state or stuck-at erased state" bits. This means that after a write operation there may be a (correctable) single bit or double bit error.*

It is recommended to repeat a failing write operation at least one time. This may help to shift (potentially few) remaining cells to the target program state.

If the *Program Verify Error (PVER)* persists or there is a mismatch between read data and target data, there are three options:

1. Skip the failing wordline and write the wordline content in the next wordline
2. Skip the failing wordline and write the wordline content in the next wordline and store the logical sector address of the failing wordline for a later repair
3. Repair the whole logical sector containing the failing wordline immediately by the use of the `Replace Logical Sector` command

Writing the content in the next wordline leaves a faulty wordline which need to be excluded from further data processing.

Replacing the whole logical sector immediately (option 3) means that the faulty wordline is no longer visible. However such an immediate repair requires the temporary backup of the already existing content of the logical sector into SRAM, replace the sector and then write back the valid content (see the description in the next section). This takes a considerable amount of additional time. If a reset occurs during this process there may be a loss of data. Option 2 mitigates this risk by delaying the repair to a situation where the sector data is no more necessary, however in this case the application must be able to skip faulty wordlines until the repair is possible.

Repair after a write operation

It is necessary to make a backup of the already existing valid data pages within the sector before the application of the `Replace Logical Sector` command.

The repair sequence shall be as follows:

1. Read all valid data from the broken logical sector
   - Use backup data in SRAM in case the data from the failing wordline cannot be read anymore. Data from functional wordlines of the logical sectors can be read[12)]

2. Apply the *Replace Logical Sector* command, using the sector address of the logical sector where the write has failed
3. After successful command execution, the current logical sector is replaced by an (erased) redundant logical sector, which is accessible at the same logical address
4. Write the stored data at the same address again

Resets during EEPROM emulation

A reset during a prog or erase operation stops the running sequence and may leave the targeted memory range in an undefined state.

A detection of such an event can be done by performing two operations in sequence and determine after reset from the correctness of the second the completeness of the first operation.

Example: after erasing a Data Flash logical sector a page is programmed. After reset the existence of correct data in this page proves that the erase process was performed completely.

The detection of aborted programming processes can be handled similarly. After a successful programming of a data block an additional page is programmed as marker. When after reset the block of data is readable and the marker is existent it is ensured that the block of data was programmed without interruption.

Instead the use of a marker the rewrite of the last written page or burst after startup may be done. This is possible if the data can be read and has only correctable errors.

Attention**: **"Over-programming" already existing data in DFLASH again is allowed but only to a limited number; please refer to the datasheet parameters $N_{D_PAGE_OVERWR}$ and $N_{D_LOGSEC_BL_PROG}$.

The probability of aborting a programming process due to power on reset can be minimized by voltage monitoring: this can be used to get an under voltage warning early enough so that an ongoing programming process can be finished and no new one is started.

In addition to the means described above there is a `User Content Count` command which checks the data integrity of a wordline or page: the command checks that the programmed and erased cells are above or below critical threshold levels to make sure that a reading is possible and the retention of the cell state is sufficient.

For the check of erase state it is possible to use the `Verify Erase Page`, `Verify Erase Wordline` or `Verify Erase Logical Sector Range` commands.

Invalidation of data

A reliable invalidation of data is possible by "over-programming" a page with "all-1" data. The datasheet parameters $N_{D_PAGE_OVERWR}$ and $N_{D_LOGSEC_BL_PROG}$ need to be taken into account as limits for such "over-programming" activities within one logical sector. The dastasheet parameters need not to be taken into account if the purpose is only to "destroy" existing data and the whole data of the logical sector is not used anymore (for example invalidation of markers before erase).

12

5.3.16 Resets during flash operation

A reset or power failure during an ongoing Flash operation (i.e. program or erase) must be considered as a violation of stable operating conditions. However, the Flash was designed to prevent damage to non-addressed Flash ranges when the reset is applied as defined in the datasheet. The addressed Flash range is left in an undefined state.

When an erase operation is aborted the previously programmed bits (‘1’) in the addressed Flash range can be in any state between ‘0’ and ‘1’. When reading this range all-0 can be returned, the old data, or something in between. The result can be unstable. Due to the ECC correction there may even appear ‘1’ bits at positions which contained ‘0’ bits before erase start.

When a page programming operation is aborted the page can still appear as erased (but contain slightly programmed bits), it can appear as being correctly programmed (but the data has a lowered retention) or the page contains garbage data. It is also possible that the read data is unstable so that depending on the operating conditions different data is read.

When Flash processes are aborted by system resets or power-on resets, this is not indicated by any flags. It is not possible to detect an aborted operation simply by reading the Flash range (please note that the ECC is not a reliable means to detect an aborted Flash operation). Also the margin reads don’t offer a reliable indication.

When erasing or programming the PFlash usually an external instance can notice the reset and restart the operation by erasing the Flash range and programming it again.

In addition there is a `User Content Count` command which checks the data integrity of a wordline or page: the command checks that the programmed and erased cells are above or below critical threshold levels to make sure that a reading is possible and the retention of the cell state is sufficient. This command can be used to check the integrity of data if it is not sure whether this data has been erased or programmed properly.

5.3.17 Interrupts

The NVM can be configured to trigger interrupt service requests on certain events. Both command interfaces request independent interrupts as described below.

HOST command interface

The following events can trigger an interrupt service request from the HOST command interface to the interrupt router (IR):

- End of BUSY : if the interrupt is enabled on "end of BUSY" using DMU_HCI_INTEN.EOBM = 1_B and one of the BUSY flags in DMU_HCI_STATUS goes low, then the end of a command sequence is detected and a interrupt is requested
- *Protection Error (PROER)* : if DMU_HCI_INTEN.PROERM = 1_B and the DMU_HCI_ERR.PROER flag is set
- *Sequence Error (SQER)* : if DMU_HCI_INTEN.SQERM = 1_B and the DMU_HCI_ERR.SQER flag is set
- *Operation Error (OPER)* : at the end of the operation, if DMU_HCI_INTEN.OPERM = 1_B and the DMU_HCI_ERR.OPER flag is set
- *Erase Verify Error (EVER)* : at the end of the operation, if DMU_HCI_INTEN.EVERM = 1_B and the DMU_HCI_ERR.EVER flag is set
- *Program Verify Error (PVER)* : at the end of the operation, if DMU_HCI_INTEN.PVERM = 1_B and the DMU_HCI_ERR.PVER flag is set

In case several conditions are enabled, the source of the interrupt can be determined by reading the DMU_HCI_STATUS and DMU_HCI_ERR registers.

CSRM command interface

The following events can trigger an interrupt service request from the CSRM command interface to the interrupt router (IR):

- End of BUSY : if the interrupt is enabled on "end of BUSY" using DMU_CSCI_INTEN.EOBM = 1_B and one of the BUSY flags in DMU_CSCI_STATUS goes low, then the end of a command sequence is detected and a interrupt is requested
- *Protection Error (PROER)* : if DMU_CSCI_INTEN.PROERM = 1_B and the DMU_CSCI_ERR.PROER flag is set
- *Sequence Error (SQER)* : if DMU_CSCI_INTEN.SQERM = 1_B and the DMU_CSCI_ERR.SQER flag is set
- *Operation Error (OPER)* : at the end of the operation, if DMU_CSCI_INTEN.OPERM = 1_B and the DMU_CSCI_ERR.OPER flag is set
- *Erase Verify Error (EVER)* : at the end of the operation, if DMU_CSCI_INTEN.EVERM = 1_B and the DMU_CSCI_ERR.EVER flag is set
- *Program Verify Error (PVER)* : at the end of the operation, if DMU_CSCI_INTEN.PVERM = 1_B and the DMU_CSCI_ERR.PVER flag is set

In case several conditions are enabled, the source of the interrupt can be determined by reading the DMU_CSCI_STATUS and DMU_CSCI_ERR registers.

5.3.18 Traps

The following fatal errors create a bus error which is translated by the CPU into a trap.

The conditions for reporting a bus error are:

- Uncorrectable ECC error
- Write access to Boot ROM (BROM)
- Write access to read-only register
- Read or write access to unoccupied register address
- Register Read or Write using double word or Block Transfer
- Write access to an access controlled register or Flash address range by a master without allowance by the register access protection
- Not allowed write access to a protected register (for example supervisor, startup protected)
- Not allowed Flash read access with active read protection
- Command Sequence to not available Flash memory
- Command Sequence like Write, Erase to a Bank which is in Sleep state (only wake up is allowed for sleeping Banks)
- Command Sequence on a command interface which is already operating a command sequence
- Command Sequence using data size other than Word (except the Double-word option for the Load Page (64 bit) command options
- Read-modify-write access to the Flash memory
- Read access to a Bank which is in Sleep state
- Read access to not available Flash memory
- Read access to a busy PFLASH bank (if not disabled by with CPU register PFly_FLASHCON1.STALL)
- Read access to a busy DFLASH bank (DFLASH0 or DFLASH1 EEPROM,UCB)
- Block Transfer for reading from a DFLASH bank

5.4 Registers

5.4.1 DMU registers

5.4.1.1 Register overview - access mode glossary

Table 467 Register overview - access mode glossary

Keyword	Description
CSCI	Access protection using PROT register PROT_PROTCSCI.
CSDF	Access protection using PROT register PROT_PROTCSDF.
CSPF	Access protection using PROT register PROT_PROTCSPF.
CSRDC	Access protection using PROT register PROT_PROTCSRDC.
HCI	Access protection using PROT register PROT_PROTHCI.
HDF	Access protection using PROT register PROT_PROTHDF.
HPF	Access protection using PROT register PROT_PROTHPF.
HRDC	Access protection using PROT register PROT_PROTHRDC.
APU-PCSCI	Protection group consisting of registers PROT_ACCENCSCI_WRA, PROT_ACCENCSCI_WRB, PROT_ACCENCSCI_RDA, PROT_ACCENCSCI_RDB, PROT_ACCENCSCI_VM, PROT_ACCENCSCI_PRS.
PCSCI	Access protection using *APU-PCSCI* registers.
APU-PCSDFUCB	Protection group consisting of registers PROT_ACCENCSDFUCB_WRA, PROT_ACCENCSDFUCB_WRB, PROT_ACCENCSDFUCB_RDA, PROT_ACCENCSDFUCB_RDB, PROT_ACCENCSDFUCB_VM, PROT_ACCENCSDFUCB_PRS.
PCSDFUCB	Access protection using *APU-PCSDFUCB* registers.
APU-PCSDFd (d=0-3)	Protection group consisting of registers PROT_ACCENCSDFd_WRA, PROT_ACCENCSDFd_WRB, PROT_ACCENCSDFd_RDA, PROT_ACCENCSDFd_RDB, PROT_ACCENCSDFd_VM, PROT_ACCENCSDFd_PRS, PROT_ACCENCSDFd_RGNLA, PROT_ACCENCSDFd_RGNUA.
PCSDFd	Access protection using *APU-PCSDFd* registers.
APU-PCSPFp (p=0-3)	Protection group consisting of registers PROT_ACCENCSPFp_WRA, PROT_ACCENCSPFp_WRB, PROT_ACCENCSPFp_VM, PROT_ACCENCSPFp_PRS, PROT_ACCENCSPFp_RGNLA, PROT_ACCENCSPFp_RGNUA.
PCSPFp	Access protection using *APU-PCSPFp* registers.
APU-PCSRDC	Protection group consisting of registers PROT_ACCENCSRDC_WRA, PROT_ACCENCSRDC_WRB, PROT_ACCENCSRDC_RDA, PROT_ACCENCSRDC_RDB, PROT_ACCENCSRDC_VM, PROT_ACCENCSRDC_PRS.
PCSRDC	Access protection using *APU-PCSRDC* registers.
APU-PHCI	Protection group consisting of registers PROT_ACCENHCI_WRA, PROT_ACCENHCI_WRB, PROT_ACCENHCI_RDA, PROT_ACCENHCI_RDB, PROT_ACCENHCI_VM, PROT_ACCENHCI_PRS.
PHCI	Access protection using *APU-PHCI* registers.

(table continues...)

Table 467 (continued) Register overview - access mode glossary

Keyword	Description
APU-PHDFUCB	Protection group consisting of registers PROT_ACCENHDFUCB_WRA, PROT_ACCENHDFUCB_WRB, PROT_ACCENHDFUCB_RDA, PROT_ACCENHDFUCB_RDB, PROT_ACCENHDFUCB_VM, PROT_ACCENHDFUCB_PRS.
PHDFUCB	Access protection using *APU-PHDFUCB* registers.
APU-PHDFd (d=0-7)	Protection group consisting of registers PROT_ACCENHDFd_WRA, PROT_ACCENHDFd_WRB, PROT_ACCENHDFd_RDA, PROT_ACCENHDFd_RDB, PROT_ACCENHDFd_VM, PROT_ACCENHDFd_PRS, PROT_ACCENHDFd_RGNLA, PROT_ACCENHDFd_RGNUA.
PHDFd	Access protection using *APU-PHDFd* registers.
APU-PHPFp (p=0-11)	Protection group consisting of registers PROT_ACCENHPFp_WRA, PROT_ACCENHPFp_WRB, PROT_ACCENHPFp_VM, PROT_ACCENHPFp_PRS, PROT_ACCENHPFp_RGNLA, PROT_ACCENHPFp_RGNUA.
PHPFp	Access protection using *APU-PHPFp* registers.
APU-PHRDC	Protection group consisting of registers PROT_ACCENHRDC_WRA, PROT_ACCENHRDC_WRB, PROT_ACCENHRDC_RDA, PROT_ACCENHRDC_RDB, PROT_ACCENHRDC_VM, PROT_ACCENHRDC_PRS.
PHRDC	Access protection using *APU-PHRDC* registers.
32	Access only when using 32-bit width.
BE	Always returns a Bus Error.
U	No access restrictions.
STP	Access only during start-up, before start-up protection is set (SMM_STCON.ST_PROT=1).
M	Indicates a module specific access condition. Refer to the register description for details of the specific access condition.

5.4.1.2 Registers overview - DMU (ascending offset address)

Table 468 Registers overview - DMU (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
HCI_STATUS	HOST status register	0004_H	PHCI	BE	Application Reset	813
HCI_BCONTROL	HOST boot control register	0008_H	PHCI	PHCI	See 816	816
HCI_OCONTROL	HOST operation control register	$000C_H$	PHCI	PHCI	Application Reset	817
HCI_ERR	HOST command interface error register	0010_H	PHCI	BE	See 818	818
HCI_CLRERR	HOST command interface clear error register	0014_H	PHCI, 32	PHCI, 32	Application Reset	821
HCI_INTEN	HOST command interface interrupt enable register	0018_H	PHCI	PHCI	Application Reset	822

(table continues...)

Table 468 (continued) Registers overview - DMU (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
HCI_SLEEP	HOST sleep register	$001C_H$	PHCI, 32	PHCI, 32	Application Reset	822
CSCI_STATUS	CSRM status register	0084_H	PCSCI	BE	Application Reset	823
CSCI_BCONTROL	CSRM boot control register	0088_H	PCSCI	PCSCI	See 826	826
CSCI_OCONTROL	CSRM operation control register	$008C_H$	PCSCI	PCSCI	Application Reset	827
CSCI_ERR	CSRM command interface error register	0090_H	PCSCI	BE	See 828	828
CSCI_CLRERR	CSRM command interface clear error register	0094_H	PCSCI, 32	PCSCI, 32	Application Reset	831
CSCI_INTEN	CSRM command interface interrupt enable register	0098_H	PCSCI	PCSCI	Application Reset	832
CSCI_SLEEP	CSRM sleep register	$009C_H$	PCSCI, 32	PCSCI, 32	Application Reset	832
GP_HOST_DFWAIT	HOST DFLASH wait cycle register	0100_H	PHRDC	PHRDC	See 834	834
GP_HOST_DFMARGIN	HOST DFLASH margin control register	0104_H	PHRDC	PHRDC	Application Reset	834
GP_HOST_DFECCR	HOST DFLASH ECC read register	0108_H	PHRDC, 32	BE	Application Reset	835
GP_HOST_DFECCS	HOST DFLASH ECC status register	$010C_H$	PHRDC, 32	BE	See 836	836
GP_HOST_DFECCC	HOST DFLASH ECC control register	0110_H	PHRDC	PHRDC	Application Reset	837
GP_HOST_DFECCW	HOST DFLASH ECC write register	0114_H	PHCI	PHCI	Application Reset	838
GP_HOST_DFPROCON	HOST DFLASH protection register	0118_H	U	32	See 839	839
GP_HOST_PFWAIT	HOST PFLASH wait cycle register	$011C_H$	PHRDC	PHRDC	See 840	840
GP_HOST_PFERRINJ	HOST PFLASH error injection register	0120_H	PHRDC	PHRDC	Application Reset	840
GP_HOST_PFECCW	HOST PFLASH ECC write register	0124_H	PHCI	PHCI	Application Reset	841
GP_HOST_PFPROCON	HOST PFLASH protection register	0128_H	U	BE	See 842	842
GP_HOST_PFPROCONWPpA0	HOST PFLASH bank PpA write protection configuration 0	$012C_H$+p *80_H	U	BE	See 842	842

(table continues...)

Table 468 (continued) Registers overview - DMU (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
GP_HOST_PFPROCONWPpA1	HOST PFLASH bank PpA write protection configuration 1	0130_H+p*80_H	U	BE	See 843	843
GP_HOST_PFPROCONWPpA2	HOST PFLASH bank PpA write protection configuration 2	0134_H+p*80_H	U	BE	See 844	844
GP_HOST_PFPROCONWPpA3	HOST PFLASH bank PpA write protection configuration 3	0138_H+p*80_H	U	BE	See 844	844
GP_HOST_PFPROCONPWPpA0	HOST PFLASH bank PpA permanent write protection configuration 0	$013C_H$+p*80_H	U	BE	See 845	845
GP_HOST_PFPROCONPWPpA1	HOST PFLASH bank PpA permanent write protection configuration 1	0140_H+p*80_H	U	BE	See 846	846
GP_HOST_PFPROCONPWPpA2	HOST PFLASH bank PpA permanent write protection configuration 2	0144_H+p*80_H	U	BE	See 846	846
GP_HOST_PFPROCONPWPpA3	HOST PFLASH bank PpA permanent write protection configuration 3	0148_H+p*80_H	U	BE	See 847	847
GP_HOST_PFPROCONPWOPpA0	HOST PFLASH bank PpA permanent write once protection configuration 0	$014C_H$+p*80_H	U	BE	See 848	848
GP_HOST_PFPROCONPWOPpA1	HOST PFLASH bank PpA permanent write once protection configuration 1	0150_H+p*80_H	U	BE	See 848	848
GP_HOST_PFPROCONPWOPpA2	HOST PFLASH bank PpA permanent write once protection configuration 2	0154_H+p*80_H	U	BE	See 849	849
GP_HOST_PFPROCONPWOPpA3	HOST PFLASH bank PpA permanent write once protection configuration 3	0158_H+p*80_H	U	BE	See 850	850
GP_HOST_PFECPRIOpA0	HOST PFLASH bank PpA erase counter priority configuration 0	$015C_H$+p*80_H	U	BE	See 850	850
GP_HOST_PFECPRIOpA1	HOST PFLASH bank PpA erase counter priority configuration 1	0160_H+p*80_H	U	BE	See 851	851
GP_HOST_PFECPRIOpA2	HOST PFLASH bank PpA erase counter priority configuration 2	0164_H+p*80_H	U	BE	See 852	852
GP_HOST_PFECPRIOpA3	HOST PFLASH bank PpA erase counter priority configuration 3	0168_H+p*80_H	U	BE	See 852	852

(table continues...)

Table 468 (continued) Registers overview - DMU (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
GP_HOST_PFPROCONWPpB0	HOST PFLASH bank PpB write protection configuration 0	$016C_H$+p*80_H	U	BE	See 853	853
GP_HOST_PFPROCONWPpB1	HOST PFLASH bank PpB write protection configuration 1	0170_H+p*80_H	U	BE	See 854	854
GP_HOST_PFPROCONWPpB2	HOST PFLASH bank PpB write protection configuration 2	0174_H+p*80_H	U	BE	See 854	854
GP_HOST_PFPROCONWPpB3	HOST PFLASH bank PpB write protection configuration 3	0178_H+p*80_H	U	BE	See 855	855
GP_HOST_PFPROCONPWPpB0	HOST PFLASH bank PpB permanent write protection configuration 0	$017C_H$+p*80_H	U	BE	See 856	856
GP_HOST_PFPROCONPWPpB1	HOST PFLASH bank PpB permanent write protection configuration 1	0180_H+p*80_H	U	BE	See 856	856
GP_HOST_PFPROCONPWPpB2	HOST PFLASH bank PpB permanent write protection configuration 2	0184_H+p*80_H	U	BE	See 857	857
GP_HOST_PFPROCONPWPpB3	HOST PFLASH bank PpB permanent write protection configuration 3	0188_H+p*80_H	U	BE	See 858	858
GP_HOST_PFPROCONPWOPpB0	HOST PFLASH bank PpB permanent write once protection configuration 0	$018C_H$+p*80_H	U	BE	See 858	858
GP_HOST_PFPROCONPWOPpB1	HOST PFLASH bank PpB permanent write once protection configuration 1	0190_H+p*80_H	U	BE	See 859	859
GP_HOST_PFPROCONPWOPpB2	HOST PFLASH bank PpB permanent write once protection configuration 2	0194_H+p*80_H	U	BE	See 860	860
GP_HOST_PFPROCONPWOPpB3	HOST PFLASH bank PpB permanent write once protection configuration 3	0198_H+p*80_H	U	BE	See 860	860
GP_HOST_PFECPRIOpB0	HOST PFLASH bank PpB erase counter priority configuration 0	$019C_H$+p*80_H	U	BE	See 861	861
GP_HOST_PFECPRIOpB1	HOST PFLASH bank PpB erase counter priority configuration 1	$01A0_H$+p*80_H	U	BE	See 862	862
GP_HOST_PFECPRIOpB2	HOST PFLASH bank PpB erase counter priority configuration 2	$01A4_H$+p*80_H	U	BE	See 862	862

(table continues...)

Table 468 (continued) Registers overview - DMU (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
GP_HOST_PFECPRIOpB3	HOST PFLASH bank PpB erase counter priority configuration 3	$01A8_H$+p*80_H	U	BE	See 863	863
GP_HOST_PROTECT	HOST UCB password protection register	$052C_H$	U	BE	Application Reset	864
GP_HOST_CONFIRMA	HOST UCB confirmation codes register A	0530_H	U	BE	See 866	866
GP_HOST_CONFIRMB	HOST UCB confirmation codes register B	0534_H	U	BE	See 868	868
GP_HOST_CONFIRMC	HOST UCB confirmation codes register C	0538_H	U	BE	See 870	870
GP_CSRM_DFWAIT	CSRM DFLASH wait cycle register	0900_H	PCSRDC	PCSRDC	See 873	873
GP_CSRM_DFMARGIN	CSRM DFLASH margin control register	0904_H	PCSRDC	PCSRDC	Application Reset	873
GP_CSRM_DFECCR	CSRM DFLASH ECC read register	0908_H	PCSRDC, 32	BE	Application Reset	874
GP_CSRM_DFECCS	CSRM DFLASH ECC status register	$090C_H$	PCSRDC, 32	BE	See 875	875
GP_CSRM_DFECCC	CSRM DFLASH ECC control register	0910_H	PCSRDC	PCSRDC	Application Reset	877
GP_CSRM_DFECCW	CSRM DFLASH ECC write register	0914_H	PCSCI	PCSCI	Application Reset	878
GP_CSRM_DFPROCON	CSRM DFLASH protection register	0918_H	U	BE	See 878	878
GP_CSRM_PFWAIT	CSRM PFLASH wait cycle register	0920_H	PCSRDC	PCSRDC	See 879	879
GP_CSRM_PFERRINJ	CSRM PFLASH error injection register	0924_H	PCSRDC	PCSRDC	Application Reset	880
GP_CSRM_PFECCW	CSRM PFLASH ECC write register	0928_H	PCSCI	PCSCI	Application Reset	880
GP_CSRM_PFPROCONWP0	CSRM PFLASH bank write protection configuration 0	$092C_H$	U	BE	See 881	881
GP_CSRM_PFPROCONWP1	CSRM PFLASH bank write protection configuration 1	0930_H	U	BE	See 881	881
GP_CSRM_PFPROCONWP2	CSRM PFLASH bank write protection configuration 2	0934_H	U	BE	See 882	882
GP_CSRM_PFPROCONWP3	CSRM PFLASH bank write protection configuration 3	0938_H	U	BE	See 883	883
GP_CSRM_PFPROCONPWP0	CSRM PFLASH bank permanent write protection configuration 0	$093C_H$	U	BE	See 883	883

(table continues...)

Table 468 (continued) Registers overview - DMU (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
GP_CSRM_PFPROCONPWP1	CSRM PFLASH bank permanent write protection configuration 1	0940_H	U	BE	See 884	884
GP_CSRM_PFPROCONPWP2	CSRM PFLASH bank permanent write protection configuration 2	0944_H	U	BE	See 885	885
GP_CSRM_PFPROCONPWP3	CSRM PFLASH bank permanent write protection configuration 3	0948_H	U	BE	See 885	885
GP_CSRM_PFPROCONPWOP0	CSRM PFLASH bank permanent write once protection configuration 0	$094C_H$	U	BE	See 886	886
GP_CSRM_PFPROCONPWOP1	CSRM PFLASH bank permanent write once protection configuration 1	0950_H	U	BE	See 887	887
GP_CSRM_PFPROCONPWOP2	CSRM PFLASH bank permanent write once protection configuration 2	0954_H	U	BE	See 887	887
GP_CSRM_PFPROCONPWOP3	CSRM PFLASH bank permanent write once protection configuration 3	0958_H	U	BE	See 888	888
GP_CSRM_PFECPRIO0	CSRM PFLASH bank erase counter priority configuration 0	$095C_H$	U	BE	See 889	889
GP_CSRM_PFECPRIO1	CSRM PFLASH bank erase counter priority configuration 1	0960_H	U	BE	See 889	889
GP_CSRM_PFECPRIO2	CSRM PFLASH bank erase counter priority configuration 2	0964_H	U	BE	See 890	890
GP_CSRM_PFECPRIO3	CSRM PFLASH bank erase counter priority configuration 3	0968_H	U	BE	See 891	891
GP_CSRM_PROTECT	CSRM UCB password protection register	$096C_H$	U	BE	Application Reset	891
GP_CSRM_CONFIRMA	CSRM UCB confirmation codes register A	0970_H	U	BE	See 893	893
GP_CSRM_CONFIRMB	CSRM UCB confirmation codes register B	0974_H	U	BE	See 895	895
GP_CSRM_CONFIRMC	CSRM UCB confirmation codes register C	0978_H	U	BE	See 897	897
GP_CSRM_DBGCFG	CSRM debug configuration	$097C_H$	U	BE	See 899	899

(table continues...)

Table 468 (continued) Registers overview - DMU (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
GP_CSRM_DBGMODE	CSRM debug mode	0980_H	U	BE	See 900	900
GP_BKALLOC	Bank allocation register	$0A00_H$	U	BE	Application Reset	901
GP_PCONTROL	Power control register	$0A14_H$	U	BE	See 902	902
PROT_PROTHCI	PROT register for HOST command interface and HCI registers	$0B00_H$	U	SV, 32	Application Reset	902
PROT_ACCENHCI_WRA	HOST command interface and HCI registers, write access enable register A	$0B08_H$	U	HCI, 32	Application Reset	904
PROT_ACCENHCI_WRB	HOST command interface and HCI registers, write access enable register B	$0B0C_H$	U	HCI, 32	Application Reset	904
PROT_ACCENHCI_RDA	HOST command interface and HCI registers, read access enable register A	$0B10_H$	U	HCI, 32	Application Reset	905
PROT_ACCENHCI_RDB	HOST command interface and HCI registers, read access enable register B	$0B14_H$	U	HCI, 32	Application Reset	905
PROT_ACCENHCI_VM	HOST command interface and HCI registers, VM access enable register	$0B18_H$	U	HCI, 32	Application Reset	906
PROT_ACCENHCI_PRS	HOST command interface and HCI registers, PRS access enable register	$0B1C_H$	U	HCI, 32	Application Reset	907
PROT_PROTCSCI	PROT register for CSRM command interface and CSCI registers	$0B28_H$	U	SV, 32	Application Reset	907
PROT_ACCENCSCI_WRA	CSRM command interface and CSCI registers, write access enable register A	$0B30_H$	U	CSCI, 32	Application Reset	909
PROT_ACCENCSCI_WRB	CSRM command interface and CSCI registers, write access enable register B	$0B34_H$	U	CSCI, 32	Application Reset	910
PROT_ACCENCSCI_RDA	CSRM command interface and CSCI registers, read access enable register A	$0B38_H$	U	CSCI, 32	Application Reset	910
PROT_ACCENCSCI_RDB	CSRM command interface and CSCI registers, read access enable register B	$0B3C_H$	U	CSCI, 32	Application Reset	911
PROT_ACCENCSCI_VM	CSRM command interface and CSCI registers, VM access enable register	$0B40_H$	U	CSCI, 32	Application Reset	911

(table continues...)

Table 468 (continued) Registers overview - DMU (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PROT_ACCENCSCI_PRS	CSRM command interface and CSCI registers, PRS access enable register	$0B44_H$	U	CSCI, 32	Application Reset	912
PROT_PROTHRDC	PROT register for HOST read configurations registers	$0B50_H$	U	SV, 32	Application Reset	913
PROT_ACCENHRDC_WRA	HOST read configurations registers, write access enable register A	$0B58_H$	U	HRDC, 32	Application Reset	914
PROT_ACCENHRDC_WRB	HOST read configurations registers, write access enable register B	$0B5C_H$	U	HRDC, 32	Application Reset	915
PROT_ACCENHRDC_RDA	HOST read configurations registers, read access enable register A	$0B60_H$	U	HRDC, 32	Application Reset	915
PROT_ACCENHRDC_RDB	HOST read configurations registers, read access enable register B	$0B64_H$	U	HRDC, 32	Application Reset	916
PROT_ACCENHRDC_VM	HOST read configurations registers, VM access enable register	$0B68_H$	U	HRDC, 32	Application Reset	916
PROT_ACCENHRDC_PRS	HOST read configurations registers, PRS access enable register	$0B6C_H$	U	HRDC, 32	Application Reset	917
PROT_PROTCSRDC	PROT register for CSRM read configurations registers	$0B78_H$	U	SV, 32	Application Reset	918
PROT_ACCENCSRDC_WRA	CSRM read configurations registers, write access enable register A	$0B80_H$	U	CSRDC, 32	Application Reset	920
PROT_ACCENCSRDC_WRB	CSRM read configurations registers, write access enable register B	$0B84_H$	U	CSRDC, 32	Application Reset	920
PROT_ACCENCSRDC_RDA	CSRM read configurations registers, read access enable register A	$0B88_H$	U	CSRDC, 32	Application Reset	921
PROT_ACCENCSRDC_RDB	CSRM read configurations registers, read access enable register B	$0B8C_H$	U	CSRDC, 32	Application Reset	921
PROT_ACCENCSRDC_VM	CSRM read configurations registers, VM access enable register	$0B90_H$	U	CSRDC, 32	Application Reset	922
PROT_ACCENCSRDC_PRS	CSRM read configurations registers, PRS access enable register	$0B94_H$	U	CSRDC, 32	Application Reset	922

(table continues...)

Table 468 (continued) Registers overview - DMU (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PROT_PROTHPF	PROT register for HOST PFLASH memory	$0BA0_H$	U	SV, 32	Application Reset	923
PROT_ACCENHPFp_WRA	HOST PFLASH memory Pp, write access enable register A	$0BA8_H$+p*20_H	U	HPF, 32	Application Reset	925
PROT_ACCENHPFp_WRB	HOST PFLASH memory Pp, write access enable register B	$0BAC_H$+p*20_H	U	HPF, 32	Application Reset	925
PROT_ACCENHPFp_VM	HOST PFLASH memory Pp, VM access enable register	$0BB8_H$+p*20_H	U	HPF, 32	Application Reset	926
PROT_ACCENHPFp_PRS	HOST PFLASH memory Pp, PRS access enable register	$0BBC_H$+p*20_H	U	HPF, 32	Application Reset	926
PROT_ACCENHPFp_RGNLA	HOST PFLASH memory Pp, region lower address register	$0BC0_H$+p*20_H	U	HPF, 32	Application Reset	927
PROT_ACCENHPFp_RGNUA	HOST PFLASH memory Pp, region upper address register	$0BC4_H$+p*20_H	U	HPF, 32	Application Reset	927
PROT_PROTHDF	PROT register for HOST DFLASH (EEPROM) and HOST UCB memory	$0DA8_H$	U	SV, 32	Application Reset	928
PROT_ACCENHDFd_WRA	HOST DFLASH (EEPROM) memory Dd, write access enable register A	$0DB0_H$+d*20_H	U	HDF, 32	Application Reset	930
PROT_ACCENHDFd_WRB	HOST DFLASH (EEPROM) memory Dd, write access enable register B	$0DB4_H$+d*20_H	U	HDF, 32	Application Reset	930
PROT_ACCENHDFd_RDA	HOST DFLASH (EEPROM) memory Dd, read access enable register A	$0DB8_H$+d*20_H	U	HDF, 32	Application Reset	931
PROT_ACCENHDFd_RDB	HOST DFLASH (EEPROM) memory Dd, read access enable register B	$0DBC_H$+d*20_H	U	HDF, 32	Application Reset	931
PROT_ACCENHDFd_VM	HOST DFLASH (EEPROM) memory Dd, VM access enable register	$0DC0_H$+d*20_H	U	HDF, 32	Application Reset	932
PROT_ACCENHDFd_PRS	HOST DFLASH (EEPROM) memory Dd, PRS access enable register	$0DC4_H$+d*20_H	U	HDF, 32	Application Reset	933
PROT_ACCENHDFd_RGNLA	HOST DFLASH (EEPROM) memory Dd, region lower address register	$0DC8_H$+d*20_H	U	HDF, 32	Application Reset	933

(table continues...)

Table 468 (continued) Registers overview - DMU (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PROT_ACCENHDFd_RGNUA	HOST DFLASH (EEPROM) memory Dd, region upper address register	$0DCC_H$+d*20_H	U	HDF, 32	Application Reset	934
PROT_ACCENHDFUCB_WRA	HOST UCB memory, write access enable register A	$0EB0_H$	U	HDF, 32	Application Reset	934
PROT_ACCENHDFUCB_WRB	HOST UCB memory, write access enable register B	$0EB4_H$	U	HDF, 32	Application Reset	935
PROT_ACCENHDFUCB_RDA	HOST UCB memory, read access enable register A	$0EB8_H$	U	HDF, 32	Application Reset	935
PROT_ACCENHDFUCB_RDB	HOST UCB memory, read access enable register B	$0EBC_H$	U	HDF, 32	Application Reset	936
PROT_ACCENHDFUCB_VM	HOST UCB memory, VM access enable register	$0EC0_H$	U	HDF, 32	Application Reset	936
PROT_ACCENHDFUCB_PRS	HOST UCB memory, PRS access enable register	$0EC4_H$	U	HDF, 32	Application Reset	937
PROT_PROTCSPF	PROT register for CSRM PFLASH memory	$0ED0_H$	U	SV, 32	Application Reset	938
PROT_ACCENCSPFp_WRA	CSRM PFLASH memory Pp, write access enable register A	$0ED8_H$+p*20_H	U	CSPF, 32	Application Reset	939
PROT_ACCENCSPFp_WRB	CSRM PFLASH memory Pp, write access enable register B	$0EDC_H$+p*20_H	U	CSPF, 32	Application Reset	940
PROT_ACCENCSPFp_VM	CSRM PFLASH memory Pp, VM access enable register	$0EE8_H$+p*20_H	U	CSPF, 32	Application Reset	940
PROT_ACCENCSPFp_PRS	CSRM PFLASH memory Pp, PRS access enable register	$0EEC_H$+p*20_H	U	CSPF, 32	Application Reset	941
PROT_ACCENCSPFp_RGNLA	CSRM PFLASH memory Pp, region lower address register	$0EF0_H$+p*20_H	U	CSPF, 32	Application Reset	942
PROT_ACCENCSPFp_RGNUA	CSRM PFLASH memory Pp, region upper address register	$0EF4_H$+p*20_H	U	CSPF, 32	Application Reset	942
PROT_PROTCSDF	PROT register for CSRM DFLASH (EEPROM) and CSRM UCB memory	$0F58_H$	U	SV, 32	Application Reset	943
PROT_ACCENCSDFd_WRA	CSRM DFLASH (EEPROM) memory Dd, write access enable register A	$0F60_H$+d*20_H	U	CSDF, 32	Application Reset	945
PROT_ACCENCSDFd_WRB	CSRM DFLASH (EEPROM) memory Dd, write access enable register B	$0F64_H$+d*20_H	U	CSDF, 32	Application Reset	945

(table continues...)

Table 468 (continued) Registers overview - DMU (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PROT_ACCENCSDFd_RDA	CSRM DFLASH (EEPROM) memory Dd, read access enable register A	$0F68_H$+d*20_H	U	CSDF, 32	Application Reset	946
PROT_ACCENCSDFd_RDB	CSRM DFLASH (EEPROM) memory Dd, read access enable register B	$0F6C_H$+d*20_H	U	CSDF, 32	Application Reset	946
PROT_ACCENCSDFd_VM	CSRM DFLASH (EEPROM) memory Dd, VM access enable register	$0F70_H$+d*20_H	U	CSDF, 32	Application Reset	947
PROT_ACCENCSDFd_PRS	CSRM DFLASH (EEPROM) memory Dd, PRS access enable register	$0F74_H$+d*20_H	U	CSDF, 32	Application Reset	947
PROT_ACCENCSDFd_RGNLA	CSRM DFLASH (EEPROM) memory Dd, region lower address register	$0F78_H$+d*20_H	U	CSDF, 32	Application Reset	948
PROT_ACCENCSDFd_RGNUA	CSRM DFLASH (EEPROM) memory Dd, region upper address register	$0F7C_H$+d*20_H	U	CSDF, 32	Application Reset	948
PROT_ACCENCSDFUCB_WRA	CSRM UCB memory, write access enable register A	$0FE0_H$	U	CSDF, 32	Application Reset	949
PROT_ACCENCSDFUCB_WRB	CSRM UCB memory, write access enable register B	$0FE4_H$	U	CSDF, 32	Application Reset	949
PROT_ACCENCSDFUCB_RDA	CSRM UCB memory, read access enable register A	$0FE8_H$	U	CSDF, 32	Application Reset	950
PROT_ACCENCSDFUCB_RDB	CSRM UCB memory, read access enable register B	$0FEC_H$	U	CSDF, 32	Application Reset	950
PROT_ACCENCSDFUCB_VM	CSRM UCB memory, VM access enable register	$0FF0_H$	U	CSDF, 32	Application Reset	951
PROT_ACCENCSDFUCB_PRS	CSRM UCB memory, PRS access enable register	$0FF4_H$	U	CSDF, 32	Application Reset	952
PROT_APUBYPASS	APU bypass	1000_H	U	32	See 952	952

5.4.1.3 HOST status register

HOST command interface registers

HCI_STATUS Offset address: 0004_H

HOST status register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
REQDONE	REQACK	0	USER	ERASE	PROG	PFPAGE	DFPAGE	0				FSIACTIVE	0		BUSYHOSTDF
rh	rh	r	rh	rh	rh	rh	rh	r				rh	r		rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		0		BUSYP5B	BUSYP5A	BUSYP4B	BUSYP4A	BUSYP3B	BUSYP3A	BUSYP2B	BUSYP2A	BUSYP1B	BUSYP1A	BUSYP0B	BUSYP0A
r		r		rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
BUSYPxA (x=0-5)	2*x	rh	**HOST PFLASH PxA BUSY setting** HW-controlled status flag. Indication of busy state of PFxA because of active execution of an operation; PFxA busy state is also indicated during Flash startup after reset or in sleep mode; while in busy state the PFxA does not allow read access. 0_B Bank ready for read 1_B Bank busy; no read possible (NVM is not fully started or bank is under operation or bank is sleeping)
BUSYPxB (x=0-5)	2*x+1	rh	**HOST PFLASH PxB BUSY setting** HW-controlled status flag. Indication of busy state of PFxB because of active execution of an operation; PFxB busy state is also indicated during Flash startup after reset or in sleep mode; while in busy state the PFxB does not allow read access. 0_B Bank ready for read 1_B Bank busy; no read possible (NVM is not fully started or bank is under operation or bank is sleeping)
BUSYHOSTDF	16	rh	**HOST DFLASH BUSY setting** HW-controlled status flag. Indication of busy state of HOST DFLASH because of active execution of an operation; HOST DFLASH busy state is also indicated during Flash startup after reset or in sleep mode; while in busy state the HOST DFLASH does not allow read access. 0_B Bank ready for read 1_B Bank busy; no read possible (NVM is not fully started or bank is under operation or bank is sleeping)
FSIACTIVE	19	rh	**HOST FSI active** This register bit shows to the system if the FSI is active.
DFPAGE	24	rh	**Data Flash in Page Mode** HW-controlled status flag. Set with Enter Page Mode for DFLASH, cleared with Write Page command. Read accesses are allowed while in page mode. 0_B DFLASH not in page mode 1_B DFLASH in page mode

(table continues...)

(continued)

Field	Bits	Type	Description
PFPAGE	25	rh	**Program Flash in Page Mode** HW-controlled status flag. Set with Enter Page Mode for Flash, cleared with Write Page command. Read accesses are allowed while in page mode. 0_B PFLASH not in page mode 1_B PFLASH in page mode
PROG	26	rh	**Programming State** This bitfield indicates that the last received operation was a programming operation. This is set per HW once the programming operation is received and stays until cleared with DMU_HCI_OCONTROL.CPROG or with the "clear status" command. It is set with last cycle of write page or write burst and replace logical sector command sequences. 0_B Programming state is cleared (default) 1_B Program operation on going or finished.
ERASE	27	rh	**Erasing State** This bitfield indicates that the last received operation was an erase or a verify erase operation. This is set per HW once the erase or verify erase operation is received and stays until cleared with DMU_HCI_OCONTROL.CERASE or with the "clear status" command. It is set with last cycle of erase or verify erase command sequences. 0_B Erasing state is cleared (default) 1_B Erase operation on going or finished.
USER	28	rh	**User State** This bitfield indicates that the last received operation was an user operation. This is set per HW once the user operation is received and stays until cleared with DMU_HCI_OCONTROL.CUSER or with the "clear status" command. It is set with last cycle of user ommand sequences 0_B User state is cleared (default) 1_B User operation on going or finished.
REQACK	30	rh	**Request acknowledgement** This bitfield is set on the first cycle of a command sequence to acknowledge that a request was received. The request acknowledge information stays until cleared by "clear status" command. 0_B No request received (default) 1_B Request received

(table continues...)

(continued)

Field	Bits	Type	Description
REQDONE	31	rh	**Request done** This bitfield is set at the end of the execution of a command sequence to acknowledge that the request is executed. The request done information stays until cleared by "clear status" command. 0_B No request executed (default) 1_B Request execution done
0	15:14, 13:12, 18:17, 23:20, 29	r	**Reserved** Read as 0; shall be written with 0.

5.4.1.4 HOST boot control register

HOST command interface registers

HCI_BCONTROL Offset address: 0008_H

HOST boot control register Reset values see: Table 469

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								DDFP	DDFD	0	RES				
r								rwh	rwh	r	r				

Field	Bits	Type	Description
RES	4:0	r	**Reserved** Read as all 1's.
DDFD	6	rwh	**Disable read access from DFLASH** This bit enables or disables the read access to DFLASH. This bit is automatically set with reset and is cleared during startup, if no read protection installed, and during startup SSW in case of internal start out of flash. This bit has specific access condition, making also use of the startup protection (STP) functionality. 0_B Read access from DFLASH is allowed 1_B Read access from DFLASH is not allowed

(table continues...)

(continued)

Field	Bits	Type	Description
DDFP	7	rwh	**Disable read access from PFLASH** This bit enables or disables the read access to PFLASH. This bit is automatically set with reset and is cleared during startup, if no read protection installed, and during startup SSW in case of internal start out of flash. This bit has specific access condition, making also use of the startup protection (STP) functionality. 0_B Read access from PFLASH is allowed 1_B Read access from PFLASH is not allowed
0	5, 31:8	r	**Reserved** Read as 0; shall be written with 0.

Table 469 Reset values of HCI_BCONTROL

Reset type	Reset value	Note
System Reset	0000 0000 0000 0000 0000 0000 1100 $X000_B$	
Application Reset	XXXX XXXX XXXX XXXX XXXX XXXX XXXX $0XXX_B$	
After Boot-FW Value	XXXX XXXX XXXX XXXX XXXX XXXX ––X1 1111_B	DDFP and DDFD are set depending on the GP_HOST_PFPROCON.RWPRO and the startup mode selected in the boot mode header

5.4.1.5 HOST operation control register

HOST command interface registers

HCI_OCONTROL Offset address: $000C_H$

HOST operation control register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0			CUSER	CERASE	CPROG	0									
r			w	w	w	r									

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0					PMPMODE	PRMODE	UCG	0					CLEAR	ABORT	0
r					rw	rw	rw	r					rwh	rwh	r

Field	Bits	Type	Description
ABORT	1	rwh	**Abort request** 0_B No action 1_B Request to abort

(table continues...)

(continued)

Field	Bits	Type	Description
CLEAR	2	rwh	**Clear Request** 0_B No action 1_B Request to clear
UCG	8	rw	**User Command Granularity** Granularity configuration of User commands. 0_B Wordline (default) 1_B Page
PRMODE	9	rw	**Programming mode** This bitfield configures the programming mode for the PFLASH. Note the DFLASH is always programmed in one pulse independently from this setting. 0_B Program in 4 pulses of 64 bits (default) 1_B Program in 1 pulse of 256 bits (please refer to the datasheet for potential additional restrictions)
PMPMODE	10	rw	**Pump mode** This bitfield configures the pump mode for the operations. 0_B Full mode (default) 1_B Half mode (test setting, do not use without clarification with Infineon FAE)
CPROG	26	w	**Clear programming State** 0_B No action 1_B Clear programming state flag STATUS.PROG
CERASE	27	w	**Clear erase State** 0_B No action 1_B Clear erase state flag STATUS.ERASE
CUSER	28	w	**Clear user State** 0_B No action 1_B Clear user state flag STATUS.USER
0	0, 7:3, 25:11, 31:29	r	**Reserved** Read as 0; shall be written with 0.

5.4.1.6 HOST command interface error register

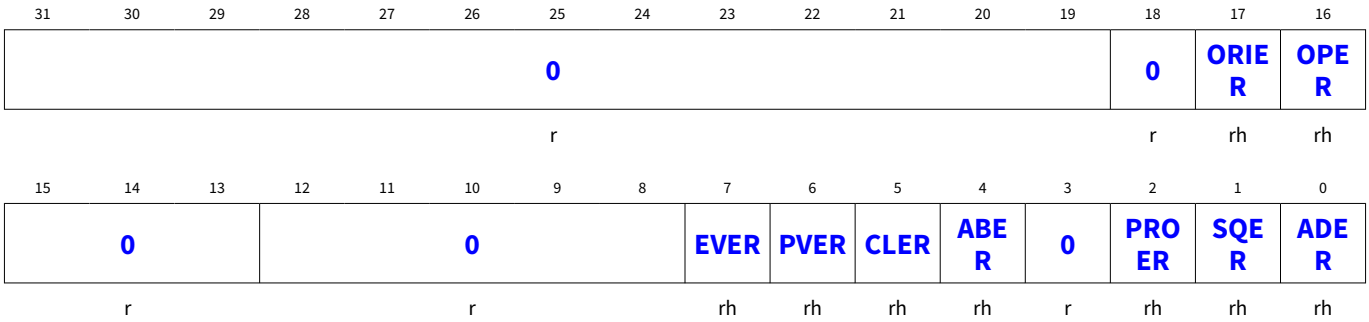
HOST command interface registers

This register stores the errors occuring while execution of NVM operations through the HOST command interface.

Note: the setting of any of these error bits takes priority over the clear operation performed using HCI_CLRERR.

HCI_ERR Offset address: 0010_H

HOST command interface error register Reset values see: Table 470



Field	Bits	Type	Description
ADER	0	rh	**SRI bus Address error** This bit is set when the DMU detects an ECC error in the address phase of the SRI transaction. *Note : Cleared with application reset or writting HCI_CLRERR.CADER=1$_B$.* 0_B No SRI address error was detected 1_B SRI address error detected
SQER	1	rh	**Command sequence error** This bit is set when the DMU detects a sequence error. *Note : Cleared with application reset, command "Reset to Read" or writing HCI_CLRERR.CSQER=1$_B$.* 0_B No sequence error was detected 1_B A sequence error was detected
PROER	2	rh	**Protection error** This bit is set when the DMU detects a protection error. *Note : Cleared with application reset, command "Reset to Read" or writing HCI_CLRERR.CPROER=1$_B$.* 0_B No protection error was detected 1_B A protection error was detected
ABER	4	rh	**Abort error** This bit is set when the DMU detects an abort error. *Note : Cleared with application reset or writting HCI_CLRERR.CABER=1$_B$.* 0_B No abort error was detected 1_B An abort error was detected
CLER	5	rh	**Clear error** This bit is set when the DMU detects a clear error. *Note : Cleared with application reset or writting HCI_CLRERR.CCLER=1$_B$.* 0_B No clear error was detected 1_B A clear error was detected

(table continues...)

(continued)

Field	Bits	Type	Description
PVER	6	rh	**Program verify error** This bit is set when, while execution of an NVM operation, the programming part failed, means the programmed data are not the expected ones. This includes error while write operations as well as hidden program operations for example : error in erase counter update while erase. The meaning of the error bits is detailed as well in the command sequence description. *Note : Cleared with application reset, command "Clear Status" or writting HCI_CLRERR.CPVER=1_B .* 0_B No program verify error was detected 1_B An program verify error was detected
EVER	7	rh	**Erase verify error** This bit is set when, while execution of an NVM operation, the expected data are not erased state. This includes error while erase operations as well as hidden erase verify checks. Example : no free entry found while replace logical sector. The meaning of the error bits is detailed as well in the command sequence description. *Note : Cleared with application reset, command "Clear Status" or writting HCI_CLRERR.CEVER=1_B .* 0_B No erase verify error was detected 1_B An erase verify error was detected
OPER	16	rh	**Flash operation error** This bit is set if a flash operation error occurred. *Note : Cleared with system reset.* 0_B No operation error was detected 1_B An operation error was detected
ORIER	17	rh	**Original error** This bit is set by hardware, if during startup, the DMU detects an ERRORED value or an uncorrectable ECC error in the ORIG confirmation code of a single UCB (one UCB containing both ORIG and COPY confirmation codes). *Note : Cleared with application reset.* 0_B No original error was detected 1_B An original error was detected
0	3, 12:8, 15:13, 18, 31:19	r	**Reserved** Read as 0; shall be written with 0.

Table 470 Reset values of HCI_ERR

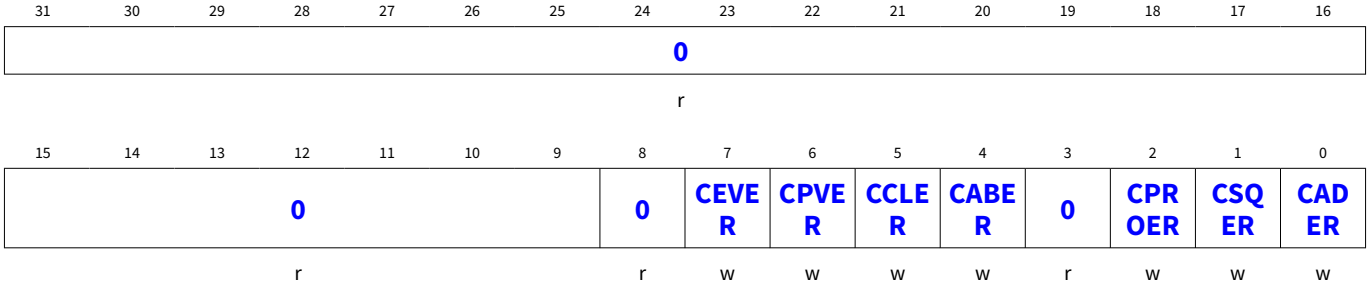
Reset type	Reset value	Note
Application Reset	0000 0000 0000 000X 0000 0000 0000 0000_B	Bit 16 (OPER) is only cleared with system reset

5.4.1.7 HOST command interface clear error register

HOST command interface registers

HCI_CLRERR Offset address: 0014_H

HOST command interface clear error register Application Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
CADER	0	w	**Clear SRI bus Address error** 0_B No action 1_B Clears the SRI address error flag DMU_HCI_ERR.ADER
CSQER	1	w	**Clears command sequence error** 0_B No action 1_B Clears the sequence error flag DMU_HCI_ERR.SQER
CPROER	2	w	**Clears protection error** 0_B No action 1_B Clears the protection error flag DMU_HCI_ERR.PROER
CABER	4	w	**Clears abort error** 0_B No action 1_B Clears the abort error flag DMU_HCI_ERR.ABER
CCLER	5	w	**Clear clear error** 0_B No action 1_B Clears the clear error flag DMU_HCI_ERR.CLER
CPVER	6	w	**Clear program verify error** 0_B No action 1_B Clears the program verify error flag DMU_HCI_ERR.PVER
CEVER	7	w	**Clear erase verify error** 0_B No action 1_B Clears the erase verify error flag DMU_HCI_ERR.EVER
0	3, 8, 31:9	r	**Reserved** Read as 0; shall be written with 0.

5.4.1.8 HOST command interface interrupt enable register

HOST command interface registers

HCI_INTEN Offset address: 0018_H

HOST command interface interrupt enable register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EOBM	0														OPERM
rw	r														rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EVERM	PVERM	0			PROERM	SQERM	0
r								rw	rw	r			rw	rw	r

Field	Bits	Type	Description
SQERM	1	rw	**Command sequence error interrupt mask** 0_B No interrupt on sequence error 1_B Interrupt enabled on sequence error
PROERM	2	rw	**Protection error interrupt mask** 0_B No interrupt on protection error 1_B Interrupt enabled on protection error
PVERM	6	rw	**Program verify error interrupt mask** 0_B No interrupt on program verify error 1_B Interrupt enabled on program verify error
EVERM	7	rw	**Erase verify error interrupt mask** 0_B No interrupt on erase verify error 1_B Interrupt enabled on erase verify error
OPERM	16	rw	**Flash operation error interrupt mask** 0_B No interrupt on flash operation error 1_B Interrupt enabled on flash operation error
EOBM	31	rw	**End of busy interrupt mask** 0_B No interrupt on end of busy 1_B Interrupt enabled on end of busy
0	0, 5:3, 15:8, 30:17	r	**Reserved** Read as 0; shall be written with 0.

5.4.1.9 HOST sleep register

HOST command interface registers

HCI_SLEEP Offset address: $001C_H$

HOST sleep register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															SLEEPHOSTDF
r															rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		0		SLEEPP5B	SLEEPP5A	SLEEPP4B	SLEEPP4A	SLEEPP3B	SLEEPP3A	SLEEPP2B	SLEEPP2A	SLEEPP1B	SLEEPP1A	SLEEPP0B	SLEEPP0A
r		r		rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
SLEEPPxA (x=0-5)	2*x	rh	**HOST PFLASH PxA sleeping** HW-controlled status flag. Indication that PFxA is sleeping thus no read nor operation can be executed on that bank. 0_B Bank awake 1_B Bank sleeping; no read, no operation possible
SLEEPPxB (x=0-5)	2*x+1	rh	**HOST PFLASH PxB sleeping** HW-controlled status flag. Indication that PFxB is sleeping thus no read nor operation can be executed on that bank. 0_B Bank awake 1_B Bank sleeping; no read, no operation possible
SLEEPHOSTDF	16	rh	**HOST DFLASH sleeping** HW-controlled status flag. Indication that HOST DFLASH is sleeping thus no read nor operation can be executed on that bank 0_B Bank awake 1_B Bank sleeping; no read, no operation possible
0	15:14, 13:12, 31:17	r	**Reserved** Read as 0; shall be written with 0.

5.4.1.10 CSRM status register

CSRM command interface registers

CSCI_STATUS Offset address: 0084_H

CSRM status register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
REQDONE	REQACK	0	USER	ERASE	PROG	PFPAGE	DFPAGE	0				FSIACTIVE	BUSYCSRMPF	BUSYCSRMDF	BUSYHOSTDF
rh	rh	r	rh	rh	rh	rh	rh	r				rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		0		BUSYP5B	BUSYP5A	BUSYP4B	BUSYP4A	BUSYP3B	BUSYP3A	BUSYP2B	BUSYP2A	BUSYP1B	BUSYP1A	BUSYP0B	BUSYP0A
r		r		rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
BUSYPxA (x=0-5)	2*x	rh	**HOST PFLASH PxA BUSY setting** HW-controlled status flag. Indication of busy state of PFxA because of active execution of an operation; PFxA busy state is also indicated during Flash startup after reset or in sleep mode; while in busy state the PFxA does not allow read access. 0_B Bank ready for read 1_B Bank busy; no read possible (NVM is not fully started or bank is under operation or bank is sleeping)
BUSYPxB (x=0-5)	2*x+1	rh	**HOST PFLASH PxB BUSY setting** HW-controlled status flag. Indication of busy state of PFxB because of active execution of an operation; PFxB busy state is also indicated during Flash startup after reset or in sleep mode; while in busy state the PFxB does not allow read access. 0_B Bank ready for read 1_B Bank busy; no read possible (NVM is not fully started or bank is under operation or bank is sleeping)
BUSYHOSTDF	16	rh	**HOST DFLASH BUSY setting** HW-controlled status flag. Indication of busy state of HOST DFLASH because of active execution of an operation; HOST DFLASH busy state is also indicated during Flash startup after reset or in sleep mode; while in busy state the HOST DFLASH does not allow read access. 0_B Bank ready for read 1_B Bank busy; no read possible (NVM is not fully started or bank is under operation or bank is sleeping)
BUSYCSRMDF	17	rh	**CSRM DFLASH BUSY setting** HW-controlled status flag. Indication of busy state of CSRM DFLASH because of active execution of an operation; CSRM DFLASH busy state is also indicated during Flash startup after reset or in sleep mode; while in busy state the CSRM DFLASH does not allow read access 0_B Bank ready for read 1_B Bank busy; no read possible (NVM is not fully started or bank is under operation or bank is sleeping)
BUSYCSRMPF	18	rh	**CSRM PFLASH BUSY setting** HW-controlled status flag. Indication of busy state of CSRM PFLASH because of active execution of an operation; CSRM PFLASH busy state is also indicated during Flash startup after reset or in sleep mode; while in busy state the CSRM PFLASH does not allow read access 0_B Bank ready for read 1_B Bank busy; no read possible (NVM is not fully started or bank is under operation or bank is sleeping)
FSIACTIVE	19	rh	**CSRM FSI active** This register bit shows to the system if the FSI is active.

(table continues...)

(continued)

Field	Bits	Type	Description
DFPAGE	24	rh	**Data Flash in Page Mode** HW-controlled status flag. Set with Enter Page Mode for DFLASH, cleared with Write Page command. Read accesses are allowed while in page mode. 0_B DFLASH not in page mode 1_B DFLASH in page mode
PFPAGE	25	rh	**Program Flash in Page Mode** HW-controlled status flag. Set with Enter Page Mode for Flash, cleared with Write Page command. Read accesses are allowed while in page mode. 0_B PFLASH not in page mode 1_B PFLASH in page mode
PROG	26	rh	**Programming State** This bitfield indicates that the last received operation was a programming operation. This is set per HW once the programming operation is received and stays until cleared with DMU_CSCI_OCONTROL.CPROG or with the "clear status" command. It is set with last cycle of write page or write burst and replace logical sector command sequences. 0_B Programming state is cleared (default) 1_B Program operation on going or finished.
ERASE	27	rh	**Erasing State** This bitfield indicates that the last received operation was an erase or a verify erase operation. This is set per HW once the erase or verify erase operation is received and stays until cleared with DMU_CSCI_OCONTROL.CERASE or with the "clear status" command. It is set with last cycle of erase or verify erase command sequences. 0_B Erasing state is cleared (default) 1_B Erase operation on going or finished.
USER	28	rh	**User State** This bitfield indicates that the last received operation was an user operation. This is set per HW once the user operation is received and stays until cleared with DMU_CSCI_OCONTROL.CUSER or with the "clear status" command. It is set with last cycle of user ommand sequences 0_B User state is cleared (default) 1_B User operation on going or finished.

(table continues...)

(continued)

Field	Bits	Type	Description
REQACK	30	rh	**Request acknowledgement** This bitfield is set on the first cycle of a command sequence to acknowledge that a request was received. The request acknowledge information stays until cleared by "clear status" command. 0_B No request received (default) 1_B Request received
REQDONE	31	rh	**Request done** This bitfield is set at the end of the execution of a command sequence to acknowledge that the request is executed. The request done information stays until cleared by "clear status" command. 0_B No request executed (default) 1_B Request execution done
0	15:14, 13:12, 23:20, 29	r	**Reserved** Read as 0; shall be written with 0.

5.4.1.11 CSRM boot control register

CSRM command interface registers

CSCI_BCONTROL Offset address: 0088_H

CSRM boot control register Reset values see: Table 471

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								DDFP	DDFD	0	RES				
r								rwh	rwh	r	r				

Field	Bits	Type	Description
RES	4:0	r	**Reserved** Read as all 1's.
DDFD	6	rwh	**Disable read access from DFLASH** This bit enables or disables the read access to DFLASH. This bit is automatically set with reset and is cleared during startup, if no read protection installed, and during startup SSW in case of internal start out of flash. This bit has specific access condition, making also use of the startup protection (STP) functionality. 0_B Read access from DFLASH is allowed 1_B Read access from DFLASH is not allowed

(table continues...)

(continued)

Field	Bits	Type	Description
DDFP	7	rwh	**Disable read access from PFLASH** This bit enables or disables the read access to PFLASH. This bit is automatically set with reset and is cleared during startup, if no read protection installed, and during startup SSW in case of internal start out of flash. This bit has specific access condition, making also use of the startup protection (STP) functionality. 0_B Read access from PFLASH is allowed 1_B Read access from PFLASH is not allowed
0	5, 31:8	r	**Reserved** Read as 0; shall be written with 0.

Table 471 Reset values of CSCI_BCONTROL

Reset type	Reset value	Note
System Reset	0000 0000 0000 0000 0000 0000 1100 $X000_B$	
Application Reset	XXXX XXXX XXXX XXXX XXXX XXXX XXXX $0XXX_B$	
After Boot-FW Value	XXXX XXXX XXXX XXXX XXXX XXXX 00X1 1111_B	system firmware sets the DDFP and DDFD bits unconditionally to 0 unless an error has happened

5.4.1.12 CSRM operation control register

CSRM command interface registers

CSCI_OCONTROL Offset address: $008C_H$

CSRM operation control register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0			CUSER	CERASE	CPROG	0									
r			w	w	w	r									

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0					PMPMODE	PRMODE	UCG	0					CLEAR	ABORT	0
r					rw	rw	rw	r					rwh	rwh	r

Field	Bits	Type	Description
ABORT	1	rwh	**Abort request** 0_B No action 1_B Request to abort

(table continues...)

(continued)

Field	Bits	Type	Description
CLEAR	2	rwh	**Clear Request** 0_B No action 1_B Request to clear
UCG	8	rw	**User Command Granularity** Granularity configuration of User commands. 0_B Wordline (default) 1_B Page
PRMODE	9	rw	**Programming mode** This bitfield configures the programming mode for the PFLASH. Note the DFLASH is always programmed in one pulse independently from this setting. 0_B Program in 4 pulses of 64 bits (default) 1_B Program in 1 pulse of 256 bits (please refer to the datasheet for potential additional restrictions)
PMPMODE	10	rw	**Pump mode** This bitfield configures the pump mode for the operations. 0_B Full mode (default) 1_B Half mode (test setting, do not use without clarification with Infineon FAE)
CPROG	26	w	**Clear programming State** 0_B No action 1_B Clear programing state flag STATUS.PROG
CERASE	27	w	**Clear erase State** 0_B No action 1_B Clear erase state flag STATUS.ERASE
CUSER	28	w	**Clear user State** 0_B No action 1_B Clear user state flag STATUS.USER
0	0, 7:3, 25:11, 31:29	r	**Reserved** Read as 0; shall be written with 0.

5.4.1.13 CSRM command interface error register

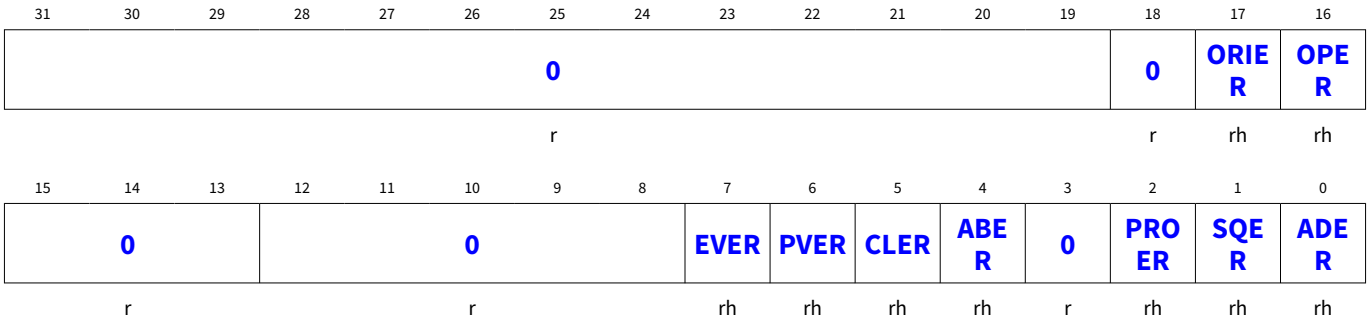
CSRM command interface registers

This register stores the errors occuring while execution of NVM operations through the CSRM command interface.

Note: the setting of any of these error bits takes priority over the clear operation performed using CSCI_CLRERR.

CSCI_ERR Offset address: 0090_H

CSRM command interface error register Reset values see: Table 472



Field	Bits	Type	Description
ADER	0	rh	**SRI bus Address error** This bit is set when the DMU detects an ECC error in the address phase of the SRI transaction. *Note : Cleared with application reset or writting CSCI_CLRERR.CADER=1$_B$.* 0_B No SRI address error was detected 1_B SRI address error detected
SQER	1	rh	**Command sequence error** This bit is set when the DMU detects a sequence error. *Note : Cleared with application reset, command "Reset to Read" or writing CSCI_CLRERR.CSQER=1$_B$.* 0_B No sequence error was detected 1_B A sequence error was detected
PROER	2	rh	**Protection error** This bit is set when the DMU detects a protection error. *Note : Cleared with application reset, command "Reset to Read" or writing CSCI_CLRERR.CPROER=1$_B$.* 0_B No protection error was detected 1_B A protection error was detected
ABER	4	rh	**Abort error** This bit is set when the DMU detects an abort error. *Note : Cleared with application reset or writting CSCI_CLRERR.CABER=1$_B$.* 0_B No abort error was detected 1_B An abort error was detected
CLER	5	rh	**Clear error** This bit is set when the DMU detects a clear error. *Note : Cleared with application reset or writting CSCI_CLRERR.CCLER=1$_B$.* 0_B No clear error was detected 1_B A clear error was detected

(table continues...)

(continued)

Field	Bits	Type	Description
PVER	6	rh	**Program verify error** This bit is set when, while execution of an NVM operation, the programming part failed, means the programmed data are not the expected ones. This includes error while write operations as well as hidden program operations for example : error in erase counter update while erase. The meaning of the error bits is detailed as well in the command sequence description. *Note : Cleared with application reset, command "Clear Status" or writting CSCI_CLRERR.CPVER=1_B .* 0_B No program verify error was detected 1_B An program verify error was detected
EVER	7	rh	**Erase verify error** This bit is set when, while execution of an NVM operation, the expected data are not erased state. This includes error while erase operations as well as hidden erase verify checks. Example : no free entry found while replace logical sector. The meaning of the error bits is detailed as well in the command sequence description. *Note : Cleared with application reset, command "Clear Status" or writting CSCI_CLRERR.CEVER=1_B .* 0_B No erase verify error was detected 1_B An erase verify error was detected
OPER	16	rh	**Flash operation error** This bit is set if a flash operation error occurred. *Note : Cleared with system reset.* 0_B No operation error was detected 1_B An operation error was detected
ORIER	17	rh	**Original error** This bit is set by hardware, if during startup, the DMU detects an ERRORED value or an uncorrectable ECC error in the ORIG confirmation code of a single UCB (one UCB containing both ORIG and COPY confirmation codes). *Note : Cleared with application reset.* 0_B No original error was detected 1_B An original error was detected
0	3, 12:8, 15:13, 18, 31:19	r	**Reserved** Read as 0; shall be written with 0.

Table 472 Reset values of CSCI_ERR

Reset type	Reset value	Note
Application Reset	0000 0000 0000 000X 0000 0000 0000 0000_B	Bit 16 (OPER) is only cleared with system reset

5.4.1.14 CSRM command interface clear error register

CSRM command interface registers

CSCI_CLRERR Offset address: 0094_H

CSRM command interface clear error register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0							0	CEVER	CPVER	CCLER	CABER	0	CPROER	CSQER	CADER
r							r	w	w	w	w	r	w	w	w

Field	Bits	Type	Description
CADER	0	w	**Clear SRI bus Address error** 0_B No action 1_B Clears the SRI address error flag DMU_CSCI_ERR.ADER
CSQER	1	w	**Clears command sequence error** 0_B No action 1_B Clears the sequence error flag DMU_CSCI_ERR.SQER
CPROER	2	w	**Clears protection error** 0_B No action 1_B Clears the protection error flag DMU_CSCI_ERR.PROER
CABER	4	w	**Clears abort error** 0_B No action 1_B Clears the abort error flag DMU_CSCI_ERR.ABER
CCLER	5	w	**Clear clear error** 0_B No action 1_B Clears the clear error flag DMU_CSCI_ERR.CLER
CPVER	6	w	**Clear program verify error** 0_B No action 1_B Clears the program verify error flag DMU_CSCI_ERR.PVER
CEVER	7	w	**Clear erase verify error** 0_B No action 1_B Clears the erase verify error flag DMU_CSCI_ERR.EVER
0	3, 8, 31:9	r	**Reserved** Read as 0; shall be written with 0.

5.4.1.15 CSRM command interface interrupt enable register

CSRM command interface registers

CSCI_INTEN Offset address: 0098_H

CSRM command interface interrupt enable register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EOBM	0														OPERM
rw	r														rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EVERM	PVERM	0			PROERM	SQERM	0
r								rw	rw	r			rw	rw	r

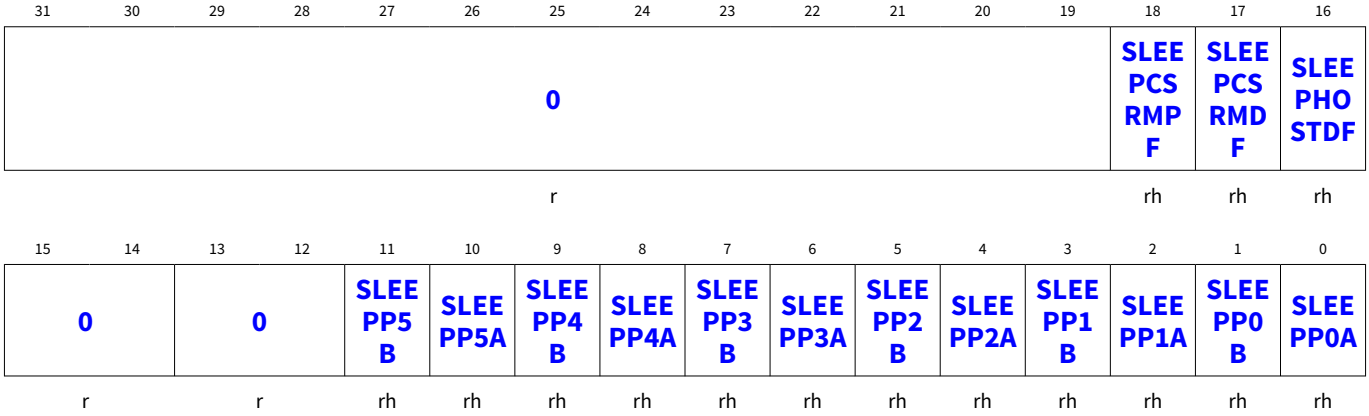
Field	Bits	Type	Description
SQERM	1	rw	**Command sequence error interrupt mask** 0_B No interrupt on sequence error 1_B Interrupt enabled on sequence error
PROERM	2	rw	**Protection error interrupt mask** 0_B No interrupt on protection error 1_B Interrupt enabled on protection error
PVERM	6	rw	**Program verify error interrupt mask** 0_B No interrupt on program verify error 1_B Interrupt enabled on program verify error
EVERM	7	rw	**Erase verify error interrupt mask** 0_B No interrupt on erase verify error 1_B Interrupt enabled on erase verify error
OPERM	16	rw	**Flash operation error interrupt mask** 0_B No interrupt on flash operation error 1_B Interrupt enabled on flash operation error
EOBM	31	rw	**End of busy interrupt mask** 0_B No interrupt on end of busy 1_B Interrupt enabled on end of busy
0	0, 5:3, 15:8, 30:17	r	**Reserved** Read as 0; shall be written with 0.

5.4.1.16 CSRM sleep register

CSRM command interface registers

CSCI_SLEEP Offset address: $009C_H$

CSRM sleep register Application Reset value: $0000\ 0000_H$

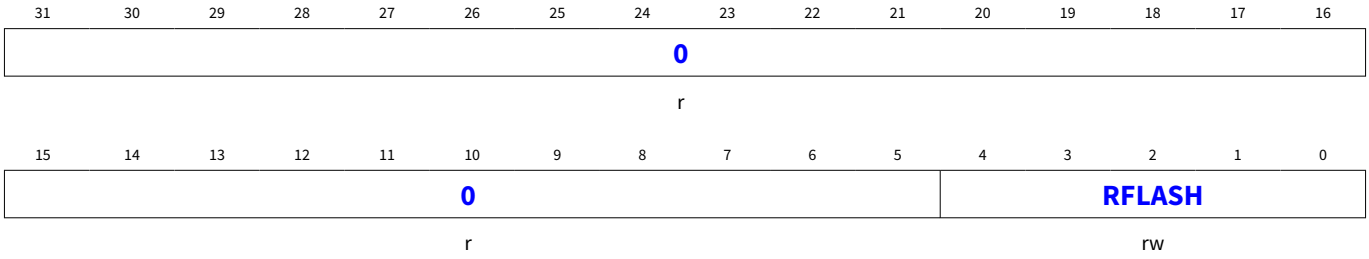


Field	Bits	Type	Description
SLEEPPxA (x=0-5)	2*x	rh	**HOST PFLASH PxA sleeping** HW-controlled status flag. Indication that PFxA is sleeping thus no read nor operation can be executed on that bank. 0_B Bank awake 1_B Bank sleeping; no read, no operation possible
SLEEPPxB (x=0-5)	2*x+1	rh	**HOST PFLASH PxB sleeping** HW-controlled status flag. Indication that PFxB is sleeping thus no read nor operation can be executed on that bank. 0_B Bank awake 1_B Bank sleeping; no read, no operation possible
SLEEPHOSTDF	16	rh	**HOST DFLASH sleeping** HW-controlled status flag. Indication that HOST DFLASH is sleeping thus no read nor operation can be executed on that bank 0_B Bank awake 1_B Bank sleeping; no read, no operation possible
SLEEPCSRMDF	17	rh	**CSRM DFLASH sleeping** HW-controlled status flag. Indication that CSRM DFLASH is sleeping thus no read nor operation can be executed on that bank 0_B Bank awake 1_B Bank sleeping; no read, no operation possible
SLEEPCSRMPF	18	rh	**CSRM PFLASH sleeping** HW-controlled status flag. Indication that CSRM PFLASH is sleeping thus no read nor operation can be executed on that bank 0_B Bank awake 1_B Bank sleeping; no read, no operation possible
0	15:14, 13:12, 31:19	r	**Reserved** Read as 0; shall be written with 0.

5.4.1.17 HOST DFLASH wait cycle register

GP_HOST_DFWAIT Offset address: 0100_H

HOST DFLASH wait cycle register Reset values see: Table 473



Field	Bits	Type	Description
RFLASH	4:0	rw	**Wait cycles for read** These registers defines the number of FSI clock cycles needed for a HOST DFLASH, UCB or CFS read access. 00_H Read access takes 1 fFSI clock cycles … $1F_H$ Read access takes 32 fFSI clock cycles
0	31:5	r	**Reserved** Read as 0; shall be written with 0.

Table 473 Reset values of GP_HOST_DFWAIT

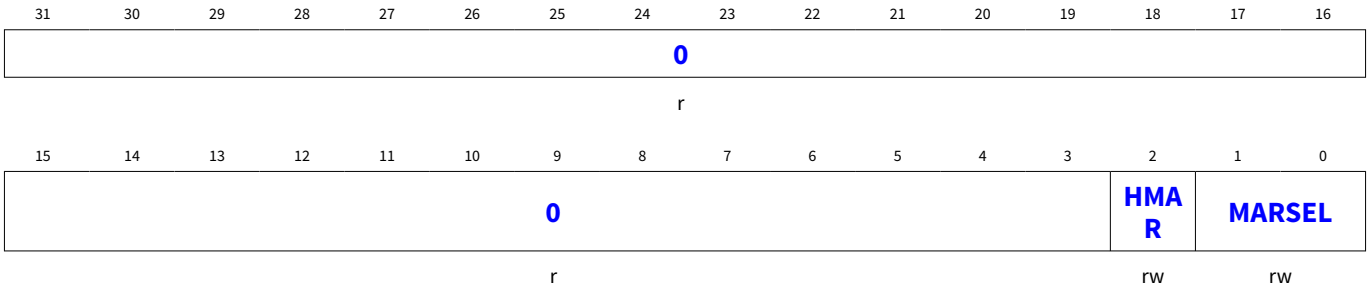
Reset type	Reset value	Note
System Reset	$0000\ 0005_H$	
After Boot-FW Value	$0000\ 0005_H$	

5.4.1.18 HOST DFLASH margin control register

If a change between the read margin settings is done using HMAR or MARSEL, the system must wait a delay time $t_{FL_MarginDel}$ until the next read is requested.

GP_HOST_DFMARGIN Offset address: 0104_H

HOST DFLASH margin control register Application Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
MARSEL	1:0	rw	**Read margin selection** This register enables to use the hard margins to read out of the HOST DFLASH, UCB and CFS memory regions. 00_B Enables the standard read margins for reading the HOST DFLASH, UCB or CFS bank. ... 10_B Enables the standard read margins for reading the HOST DFLASH, UCB or CFS bank. 11_B Enables the read margins chosen by DMU_GP_HOST_DFMARGIN.HMAR for reading the HOST DFLASH, UCB or CFS bank
HMAR	2	rw	**Hard margin selection** These registers enable to chose between the thigh-0 and the tight-1 margin for reading out of the HOST DFLASH, UCB or CFS memory regions. This setting is effective only if DMU_GP_HOST_DFMARGIN.MARSEL=11_B. 0_B Enables the tight-0 read margins (sub-optimal 0 read as 1) for reading the HOST DFLASH, UCB or CFS bank. 1_B Enables the tight-1 read margins (sub-optimal 1 read as 0) for reading the HOST DFLASH, UCB or CFS bank.
0	31:3	r	**Reserved** Read as 0; shall be written with 0.

5.4.1.19 HOST DFLASH ECC read register

The ECC read register stores the ECC checksum corresponding to the last read on the HOST DFLASH, UCB or CFS bank when the read is initiated through the DMU SRI slave interface.

GP_HOST_DFECCR Offset address: 0108_H

HOST DFLASH ECC read register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								RES		RCODE					
r								rh		rh					

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RCODE															
rh															

Field	Bits	Type	Description
RCODE	21:0	rh	**Error correction code from the last read.** These registers contain the ECC checksum corresponding to the last HOST DFLASH, UCB or CFS read access.
RES	23:22	rh	**Reserved** Reserved. Internal
0	31:24	r	**Reserved** Read as 0; shall be written with 0.

5.4.1.20 HOST DFLASH ECC status register

The ECC status register stores the errors found by the ECC corresponding to the last read(s) on the HOST DFLASH, UCB or CFS bank when the read is initiated through the DMU SRI slave interface.

GP_HOST_DFECCS Offset address: $010C_H$

HOST DFLASH ECC status register Reset values see: Table 474

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0							AANER	0				AUCER	ATBER	ADBER	ASBER
r							rh	r				rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0							ANER	0				UCER	TBER	DBER	SBER
r							rh	r				rh	rh	rh	rh

Field	Bits	Type	Description
SBER	0	rh	**Single Bit ECC Error** This register reports a single bit error found in the last read access. 0_B No single bit error was detected by the ECC in the last read access. 1_B A single bit error was detected by the ECC in the last read access.
DBER	1	rh	**Double Bit ECC Error** This register reports a double bit error found in the last read access. 0_B No double bit error was detected by the ECC in the last read access. 1_B A double bit error was detected by the ECC in the last read access.
TBER	2	rh	**Triple Bit ECC Error** This register reports a triple bit error found in the last read access. 0_B No triple bit error was detected by the ECC in the last read access. 1_B A triple bit error was detected by the ECC in the last read access.
UCER	3	rh	**Uncorrectable ECC Error** This register reports an uncorrectable error found in the last read access. 0_B No uncorrectable error was detected by the ECC in the last read access. 1_B An uncorrectable error was detected by the ECC in the last read access.
ANER	8	rh	**Any ECC error** This register reports an error found in the last read access. 0_B No error was detected by the ECC in the last read access. 1_B An error was detected by the ECC in the last read access.

(table continues...)

(continued)

Field	Bits	Type	Description
ASBER	16	rh	**Accumulated Single Bit ECC Error** This register reports that at least one single bit error was found in the read accesses since the last reset or clear. 0_B No single bit error was detected by the ECC in the read accesses. 1_B A single bit error was detected by the ECC in the read accesses.
ADBER	17	rh	**Accumulated Double Bit ECC Error** This register reports that at least one double bit error was found in the read accesses since the last reset or clear. 0_B No double bit error was detected by the ECC in the read accesses. 1_B A double bit error was detected by the ECC in the read accesses.
ATBER	18	rh	**Accumulated Triple Bit ECC Error** This register reports that at least one triple bit error was found in the read accesses since the last reset or clear. 0_B No triple bit error was detected by the ECC in the read accesses. 1_B A triple bit error was detected by the ECC in the read accesses.
AUCER	19	rh	**Accumulated Uncorrectable ECC Error** This register reports that at least one uncorrectable error was found in the read accesses since the last reset or clear. 0_B No uncorrectable error was detected by the ECC in the read accesses. 1_B A uncorrectable error was detected by the ECC in the read accesses.
AANER	24	rh	**Accumulated Any ECC error** This register reports that at least one error was found in the read accesses since the last reset or clear. 0_B No error was detected by the ECC in the read accesses. 1_B An error was detected by the ECC in the read accesses.
0	7:4, 15:9, 23:20, 31:25	r	**Reserved** Read as 0; shall be written with 0.

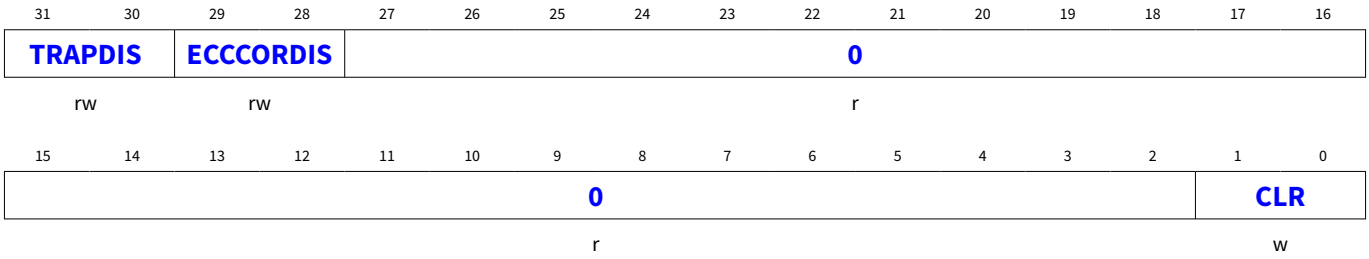
Table 474 Reset values of GP_HOST_DFECCS

Reset type	Reset value	Note
Application Reset	$0000\ 0000_H$	
After Boot-FW Value	$0000\ 0000_H$	

5.4.1.21 HOST DFLASH ECC control register

GP_HOST_DFECCC Offset address: 0110_H

HOST DFLASH ECC control register Application Reset value: $C000\ 0000_H$



Field	Bits	Type	Description
CLR	1:0	w	**Clear ECC status bits** These registers clears the DMU_GP_HOST_DFECCS registers. 00_B No action … 10_B No action 11_B Clears the DMU_GP_HOST_DFECCS
ECCCORDIS	29:28	rw	**ECC correction disable** These registers disable the ECC correction for the HOST DFLASH, UCB and CFS bank. 00_B ECC correction is enabled for HOST DFLASH, UCB and CFS … 10_B ECC correction is enabled for HOST DFLASH, UCB and CFS 11_B ECC correction is disabled for HOST DFLASH, UCB and CFS
TRAPDIS	31:30	rw	**ECC trap disable** These registers disable the bus error trap on ECC uncorrectable errors for the HOST DFLASH, UCB and CFS bank 00_B Bus error trap is enabled for HOST DFLASH, UCB and CFS uncorrectable ECC errors … 10_B Bus error trap is enabled for HOST DFLASH, UCB and CFS uncorrectable ECC errors 11_B Bus error trap is disabled for HOST DFLASH, UCB and CFS uncorrectable ECC errors
0	27:2	r	**Reserved** Read as 0; shall be written with 0.

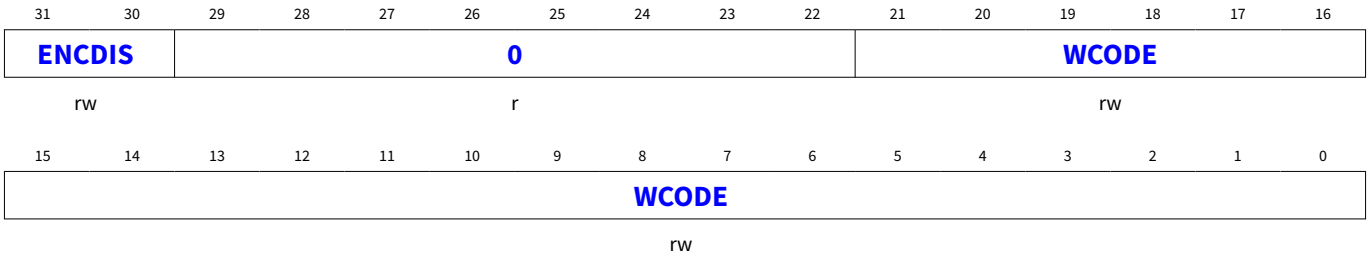
5.4.1.22 HOST DFLASH ECC write register

These registers control the behavior of the ECC for write.

When disabling the ECC encoding with DMU_GP_HOST_DFECCW.ENCDIS='11_B' the ECC code for the next 64-bit data block transferred from the DMU to the HOST DFLASH is taken from the DMU_GP_HOST_DFECCW.WCODE.

GP_HOST_DFECCW Offset address: 0114_H

HOST DFLASH ECC write register Application Reset value: $0000\ 0000_H$

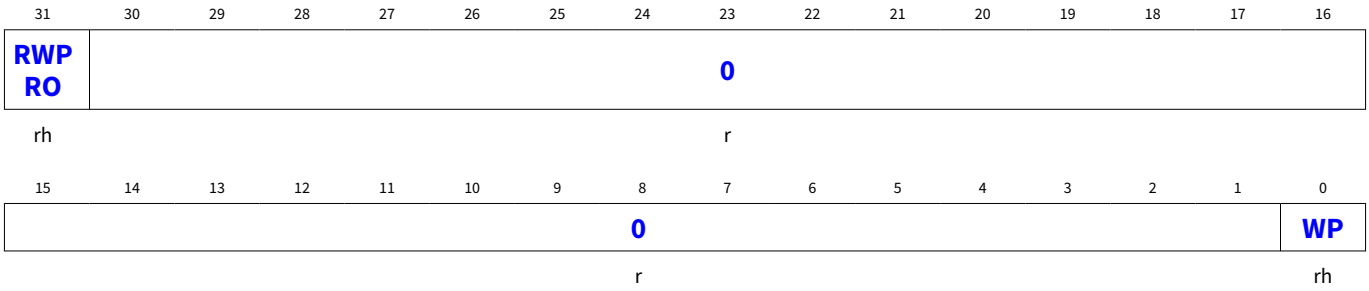


Field	Bits	Type	Description
WCODE	21:0	rw	**Error correction code for write** These registers contain the ECC checksum for the next write on HOST DFLASH.
ENCDIS	31:30	rw	**ECC encoding disable** 00_B The ECC code is calculated automatically. … 10_B The ECC code is calculated automatically. 11_B The ECC code is taken from WCODE
0	29:22	r	**Reserved** Read as 0; shall be written with 0.

5.4.1.23 HOST DFLASH protection register

GP_HOST_DFPROCON Offset address: 0118_H

HOST DFLASH protection register Reset values see: Table 475



Field	Bits	Type	Description
WP	0	rh	**Write protection** This bit indicates whether the HOST DFLASH is write protected. 0_B No write protection is configured for HOST DFLASH. 1_B Write protection is configured for HOST DFLASH.
RWPRO	31	rh	**Read and write protections configuration** This bit indicates whether read and write protection are configured for HOST DFLASH. 0_B No read and no write protection are configured for HOST DFLASH. 1_B Read and write protections are configured for HOST DFLASH.

(table continues...)

(continued)

Field	Bits	Type	Description
0	30:1	r	**Reserved** Read as 0; shall be written with 0.

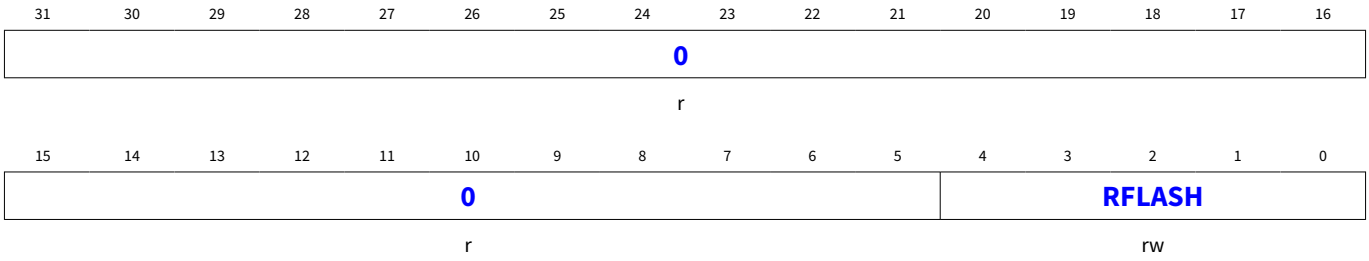
Table 475 Reset values of GP_HOST_DFPROCON

Reset type	Reset value	Note
Application Reset	8000 0000$_H$	
After Boot-FW Value	–000 0000 0000 0000 0000 0000 0000 000–$_B$	Values recovered from UCB

5.4.1.24 HOST PFLASH wait cycle register

GP_HOST_PFWAIT Offset address: 011C$_H$

HOST PFLASH wait cycle register Reset values see: Table 476



Field	Bits	Type	Description
RFLASH	4:0	rw	**HOST PFLASH wait cycles for read** These registers defines the number of SRI clock cycles needed for a HOST PFLASH read access. 00$_H$ Read access takes 2 f_{SRI} clock cycles. 01$_H$ Read access takes 2 fSRI clock cycles … 1F$_H$ Read access takes 32 fSRI clock cycles
0	31:5	r	**Reserved** Read as 0; shall be written with 0.

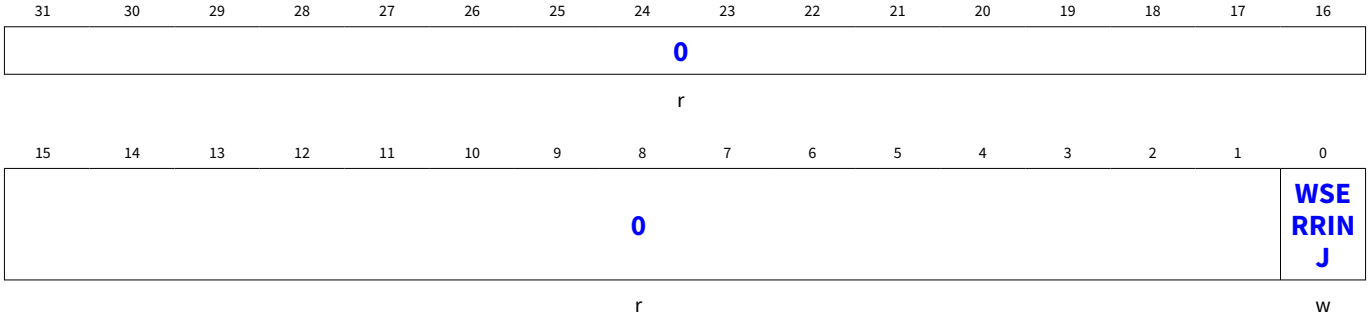
Table 476 Reset values of GP_HOST_PFWAIT

Reset type	Reset value	Note
System Reset	0000 000B$_H$	
After Boot-FW Value	0000 000B$_H$	

5.4.1.25 HOST PFLASH error injection register

GP_HOST_PFERRINJ Offset address: 0120$_H$

HOST PFLASH error injection register Application Reset value: 0000 0000$_H$



Field	Bits	Type	Description
WSERRINJ	0	w	**HOST PFLASH Wait State ECC error injection** Error injection into the ECC logic protecting PFlash Wait states to trigger an alarm. 0_B No action 1_B An error is injected in the ECC checksum protecting the PFLASH wait states
0	31:1	r	**Reserved** Read as 0; shall be written with 0.

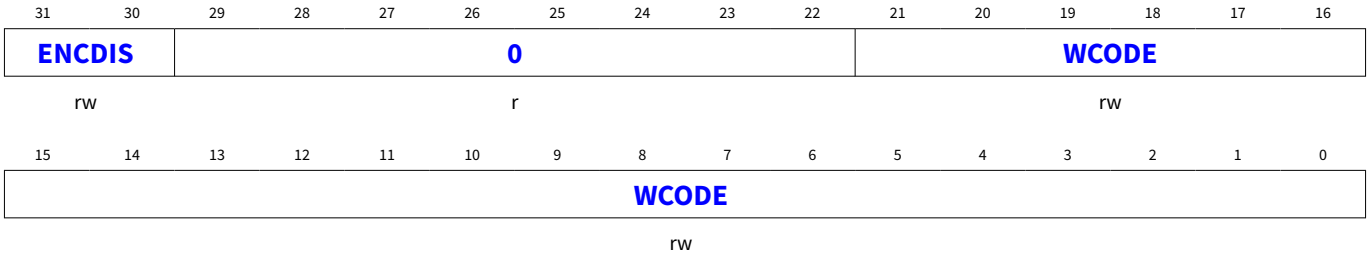
5.4.1.26 HOST PFLASH ECC write register

These registers control the behavior of the ECC for write.

When disabling the ECC correction with DMU_GP_HOST_PFECCW.ENCDIS='11_B' the ECC code for the next 256-bit data block transferred from the DMU to the HOST PFLASH is taken from the DMU_GP_HOST_PFECCW.WCODE.

GP_HOST_PFECCW Offset address: 0124_H

HOST PFLASH ECC write register Application Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
WCODE	21:0	rw	**Error correction code for write** These registers contain the ECC checksum for the next write on HOST PFLASH.
ENCDIS	31:30	rw	**ECC encoding disable** 00_B The ECC code is calculated automatically. … 10_B The ECC code is calculated automatically. 11_B The ECC code is taken from WCODE

(table continues...)

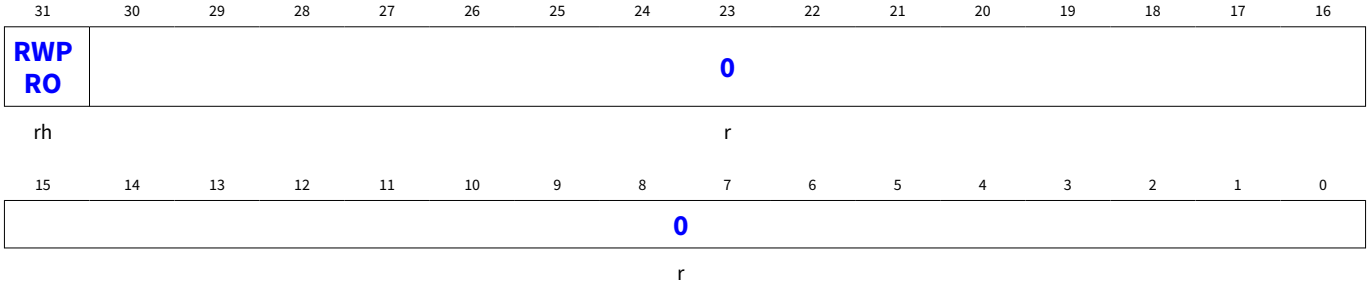
(continued)

Field	Bits	Type	Description
0	29:22	r	**Reserved** Read as 0; shall be written with 0.

5.4.1.27 HOST PFLASH protection register

GP_HOST_PFPROCON Offset address: 0128_H

HOST PFLASH protection register Reset values see: Table 477



Field	Bits	Type	Description
RWPRO	31	rh	**Read and write protections configuration** This bit indicates whether read and write protection are configured for HOST PFLASH. 0_B No read and no write protection are configured for HOST PFLASH. 1_B Read and write protections are configured for HOST PFLASH.
0	30:0	r	**Reserved** Read as 0; shall be written with 0.

Table 477 Reset values of GP_HOST_PFPROCON

Reset type	Reset value	Note
Application Reset	$8000\ 0000_H$	
After Boot-FW Value	–000 0000 0000 0000 0000 0000 0000 0000_B	Value recovered from UCB

5.4.1.28 HOST PFLASH bank PpA write protection configuration 0

GP_HOST_PFPROCONWPpA0 (p=0-5) Offset address: $012C_H+p*80_H$

HOST PFLASH bank PpA write protection configuration 0 Reset values see: Table 478

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
S31WP	S30WP	S29WP	S28WP	S27WP	S26WP	S25WP	S24WP	S23WP	S22WP	S21WP	S20WP	S19WP	S18WP	S17WP	S16WP
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
S15WP	S14WP	S13WP	S12WP	S11WP	S10WP	S9WP	S8WP	S7WP	S6WP	S5WP	S4WP	S3WP	S2WP	S1WP	S0WP
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
SxWP (x=0-31)	x	rh	**PFLASH pA sector x write protection** These bits indicates whether PFLASH pA sector x is write protected 0_B No write protection is configured for sector x 1_B Write protection is configured for sector x

Table 478 Reset values of GP_HOST_PFPROCONWPpA0 (p=0-5)

Reset type	Reset value	Note
Application Reset	0000 0000$_H$	
After Boot-FW Value	---- ----$_H$	Values recovered from UCB

5.4.1.29 HOST PFLASH bank PpA write protection configuration 1

GP_HOST_PFPROCONWPpA1 (p=0-5) Offset address: 0130_H+p*80_H

HOST PFLASH bank PpA write protection configuration 1 Reset values see: Table 479

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
S63WP	S62WP	S61WP	S60WP	S59WP	S58WP	S57WP	S56WP	S55WP	S54WP	S53WP	S52WP	S51WP	S50WP	S49WP	S48WP
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
S47WP	S46WP	S45WP	S44WP	S43WP	S42WP	S41WP	S40WP	S39WP	S38WP	S37WP	S36WP	S35WP	S34WP	S33WP	S32WP
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
SxWP (x=32-63)	x-32	rh	**PFLASH pA sector x write protection** These bits indicates whether PFLASH pA sector x is write protected 0_B No write protection is configured for sector x 1_B Write protection is configured for sector x

Table 479 Reset values of GP_HOST_PFPROCONWPpA1 (p=0-5)

Reset type	Reset value	Note
Application Reset	0000 0000$_H$	
After Boot-FW Value	---- ----$_H$	Values recovered from UCB

5.4.1.30 HOST PFLASH bank PpA write protection configuration 2

GP_HOST_PFPROCONWPpA2 (p=0-5) Offset address: 0134$_H$+p*80$_H$

HOST PFLASH bank PpA write protection configuration 2 Reset values see: Table 480

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
S95WP	S94WP	S93WP	S92WP	S91WP	S90WP	S89WP	S88WP	S87WP	S86WP	S85WP	S84WP	S83WP	S82WP	S81WP	S80WP
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
S79WP	S78WP	S77WP	S76WP	S75WP	S74WP	S73WP	S72WP	S71WP	S70WP	S69WP	S68WP	S67WP	S66WP	S65WP	S64WP
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
SxWP (x=64-95)	x-64	rh	**PFLASH pA sector x write protection** These bits indicates whether PFLASH pA sector x is write protected 0_B No write protection is configured for sector x 1_B Write protection is configured for sector x

Table 480 Reset values of GP_HOST_PFPROCONWPpA2 (p=0-5)

Reset type	Reset value	Note
Application Reset	0000 0000$_H$	
After Boot-FW Value	---- ----$_H$	Values recovered from UCB

5.4.1.31 HOST PFLASH bank PpA write protection configuration 3

GP_HOST_PFPROCONWPpA3 (p=0-5) Offset address: 0138$_H$+p*80$_H$

HOST PFLASH bank PpA write protection configuration 3 Reset values see: Table 481

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
S127 WP	S126 WP	S125 WP	S124 WP	S123 WP	S122 WP	S121 WP	S120 WP	S119 WP	S118 WP	S117 WP	S116 WP	S115 WP	S114 WP	S113 WP	S112 WP
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
S111 WP	S110 WP	S109 WP	S108 WP	S107 WP	S106 WP	S105 WP	S104 WP	S103 WP	S102 WP	S101 WP	S100 WP	S99 WP	S98 WP	S97 WP	S96 WP
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
SxWP (x=96-127)	x-96	rh	**PFLASH pA sector x write protection** These bits indicates whether PFLASH pA sector x is write protected 0_B No write protection is configured for sector x 1_B Write protection is configured for sector x

Table 481 Reset values of GP_HOST_PFPROCONWPpA3 (p=0-5)

Reset type	Reset value	Note
Application Reset	0000 0000$_H$	
After Boot-FW Value	---- ----$_H$	Values recovered from UCB

5.4.1.32 HOST PFLASH bank PpA permanent write protection configuration 0

GP_HOST_PFPROCONPWPpA0 (p=0-5) Offset address: 013C$_H$+p*80$_H$

HOST PFLASH bank PpA permanent write protection configuration 0 Reset values see: Table 482

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
S31P WP	S30P WP	S29P WP	S28P WP	S27P WP	S26P WP	S25P WP	S24P WP	S23P WP	S22P WP	S21P WP	S20P WP	S19P WP	S18P WP	S17P WP	S16P WP
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
S15P WP	S14P WP	S13P WP	S12P WP	S11P WP	S10P WP	S9P WP	S8P WP	S7P WP	S6P WP	S5P WP	S4P WP	S3P WP	S2P WP	S1P WP	S0P WP
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
SxPWP (x=0-31)	x	rh	**PFLASH pA sector x permanent write protection** These bits indicates whether PFLASH pA sector x is permanently write protected 0_B No PWP protection is configured for sector x 1_B PWP protection is configured for sector x

Table 482 Reset values of GP_HOST_PFPROCONPWPpA0 (p=0-5)

Reset type	Reset value	Note
Application Reset	0000 0000$_H$	
After Boot-FW Value	---- ----$_H$	Values recovered from UCB

5.4.1.33 HOST PFLASH bank PpA permanent write protection configuration 1

GP_HOST_PFPROCONPWPpA1 (p=0-5) Offset address: 0140$_H$+p*80$_H$

HOST PFLASH bank PpA permanent write protection configuration 1 Reset values see: Table 483

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
S63PWP	S62PWP	S61PWP	S60PWP	S59PWP	S58PWP	S57PWP	S56PWP	S55PWP	S54PWP	S53PWP	S52PWP	S51PWP	S50PWP	S49PWP	S48PWP
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
S47PWP	S46PWP	S45PWP	S44PWP	S43PWP	S42PWP	S41PWP	S40PWP	S39PWP	S38PWP	S37PWP	S36PWP	S35PWP	S34PWP	S33PWP	S32PWP
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
SxPWP (x=32-63)	x-32	rh	**PFLASH pA sector x permanent write protection** These bits indicates whether PFLASH pA sector x is permanently write protected 0_B No PWP protection is configured for sector x 1_B PWP protection is configured for sector x

Table 483 Reset values of GP_HOST_PFPROCONPWPpA1 (p=0-5)

Reset type	Reset value	Note
Application Reset	0000 0000$_H$	
After Boot-FW Value	---- ----$_H$	Values recovered from UCB

5.4.1.34 HOST PFLASH bank PpA permanent write protection configuration 2

GP_HOST_PFPROCONPWPpA2 (p=0-5) Offset address: 0144$_H$+p*80$_H$

HOST PFLASH bank PpA permanent write protection configuration 2 Reset values see: Table 484

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
S95PWP	S94PWP	S93PWP	S92PWP	S91PWP	S90PWP	S89PWP	S88PWP	S87PWP	S86PWP	S85PWP	S84PWP	S83PWP	S82PWP	S81PWP	S80PWP
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
S79PWP	S78PWP	S77PWP	S76PWP	S75PWP	S74PWP	S73PWP	S72PWP	S71PWP	S70PWP	S69PWP	S68PWP	S67PWP	S66PWP	S65PWP	S64PWP
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
SxPWP (x=64-95)	x-64	rh	**PFLASH pA sector x permanent write protection** These bits indicates whether PFLASH pA sector x is permanently write protected 0_B No PWP protection is configured for sector x 1_B PWP protection is configured for sector x

Table 484 Reset values of GP_HOST_PFPROCONPWPpA2 (p=0-5)

Reset type	Reset value	Note
Application Reset	0000 0000$_H$	
After Boot-FW Value	---- ----$_H$	Values recovered from UCB

5.4.1.35 HOST PFLASH bank PpA permanent write protection configuration 3

GP_HOST_PFPROCONPWPpA3 (p=0-5) Offset address: 0148$_H$+p*80$_H$

HOST PFLASH bank PpA permanent write protection configuration 3 Reset values see: Table 485

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
S127PWP	S126PWP	S125PWP	S124PWP	S123PWP	S122PWP	S121PWP	S120PWP	S119PWP	S118PWP	S117PWP	S116PWP	S115PWP	S114PWP	S113PWP	S112PWP
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
S111PWP	S110PWP	S109PWP	S108PWP	S107PWP	S106PWP	S105PWP	S104PWP	S103PWP	S102PWP	S101PWP	S100PWP	S99PWP	S98PWP	S97PWP	S96PWP
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
SxPWP (x=96-127)	x-96	rh	**PFLASH pA sector x permanent write protection** These bits indicates whether PFLASH pA sector x is permanently write protected 0_B No PWP protection is configured for sector x 1_B PWP protection is configured for sector x

Table 485 **Reset values of** GP_HOST_PFPROCONPWPpA3 (p=0-5)

Reset type	Reset value	Note
Application Reset	0000 0000$_H$	
After Boot-FW Value	---- ----$_H$	Values recovered from UCB

5.4.1.36 HOST PFLASH bank PpA permanent write once protection configuration 0

GP_HOST_PFPROCONPWOPpA0 (p=0-5)

Offset address: 014C$_H$+p*80$_H$

HOST PFLASH bank PpA permanent write once protection configuration 0

Reset values see: Table 486

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
S31WOP	S30WOP	S29WOP	S28WOP	S27WOP	S26WOP	S25WOP	S24WOP	S23WOP	S22WOP	S21WOP	S20WOP	S19WOP	S18WOP	S17WOP	S16WOP
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
S15WOP	S14WOP	S13WOP	S12WOP	S11WOP	S10WOP	S9WOP	S8WOP	S7WOP	S6WOP	S5WOP	S4WOP	S3WOP	S2WOP	S1WOP	S0WOP
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
SxWOP (x=0-31)	x	rh	**PFLASH pA sector x permanent write once protection** These bits indicates whether PFLASH pA sector x is permanently write once protected 0_B No permanent write once protection is configured for sector x 1_B Permanent write once protection is configured for sector x

Table 486 **Reset values of** GP_HOST_PFPROCONPWOPpA0 (p=0-5)

Reset type	Reset value	Note
Application Reset	0000 0000$_H$	
After Boot-FW Value	---- ----$_H$	Values recovered from UCB

5.4.1.37 HOST PFLASH bank PpA permanent write once protection configuration 1

GP_HOST_PFPROCONPWOPpA1 (p=0-5)

Offset address: 0150$_H$+p*80$_H$

HOST PFLASH bank PpA permanent write once protection configuration 1

Reset values see: Table 487

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
S63WOP	S62WOP	S61WOP	S60WOP	S59WOP	S58WOP	S57WOP	S56WOP	S55WOP	S54WOP	S53WOP	S52WOP	S51WOP	S50WOP	S49WOP	S48WOP
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
S47WOP	S46WOP	S45WOP	S44WOP	S43WOP	S42WOP	S41WOP	S40WOP	S39WOP	S38WOP	S37WOP	S36WOP	S35WOP	S34WOP	S33WOP	S32WOP
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
SxWOP (x=32-63)	x-32	rh	**PFLASH pA sector x permanent write once protection** These bits indicates whether PFLASH pA sector x is permanently write once protected 0_B No permanent write once protection is configured for sector x 1_B Permanent write once protection is configured for sector x

Table 487 Reset values of GP_HOST_PFPROCONPWOPpA1 (p=0-5)

Reset type	Reset value	Note
Application Reset	0000 0000_H	
After Boot-FW Value	---- $----_H$	Values recovered from UCB

5.4.1.38 HOST PFLASH bank PpA permanent write once protection configuration 2

GP_HOST_PFPROCONPWOPpA2 (p=0-5)

HOST PFLASH bank PpA permanent write once protection configuration 2

Offset address: 0154_H+p*80_H

Reset values see: Table 488

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
S95WOP	S94WOP	S93WOP	S92WOP	S91WOP	S90WOP	S89WOP	S88WOP	S87WOP	S86WOP	S85WOP	S84WOP	S83WOP	S82WOP	S81WOP	S80WOP
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
S79WOP	S78WOP	S77WOP	S76WOP	S75WOP	S74WOP	S73WOP	S72WOP	S71WOP	S70WOP	S69WOP	S68WOP	S67WOP	S66WOP	S65WOP	S64WOP
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
SxWOP (x=64-95)	x-64	rh	**PFLASH pA sector x permanent write once protection** These bits indicates whether PFLASH pA sector x is permanently write once protected 0_B No permanent write once protection is configured for sector x 1_B Permanent write once protection is configured for sector x

Table 488 Reset values of GP_HOST_PFPROCONPWOPpA2 (p=0-5)

Reset type	Reset value	Note
Application Reset	0000 0000$_H$	
After Boot-FW Value	---- ----$_H$	Values recovered from UCB

5.4.1.39 HOST PFLASH bank PpA permanent write once protection configuration 3

GP_HOST_PFPROCONPWOPpA3 (p=0-5) Offset address: 0158$_H$+p*80$_H$

HOST PFLASH bank PpA permanent write once protection configuration 3 Reset values see: Table 489

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
S127 WOP	S126 WOP	S125 WOP	S124 WOP	S123 WOP	S122 WOP	S121 WOP	S120 WOP	S119 WOP	S118 WOP	S117 WOP	S116 WOP	S115 WOP	S114 WOP	S113 WOP	S112 WOP
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
S111 WOP	S110 WOP	S109 WOP	S108 WOP	S107 WOP	S106 WOP	S105 WOP	S104 WOP	S103 WOP	S102 WOP	S101 WOP	S100 WOP	S99 WOP	S98 WOP	S97 WOP	S96 WOP
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
SxWOP (x=96-127)	x-96	rh	**PFLASH pA sector x permanent write once protection** These bits indicates whether PFLASH pA sector x is permanently write once protected 0_B No permanent write once protection is configured for sector x 1_B Permanent write once protection is configured for sector x

Table 489 Reset values of GP_HOST_PFPROCONPWOPpA3 (p=0-5)

Reset type	Reset value	Note
Application Reset	0000 0000$_H$	
After Boot-FW Value	---- ----$_H$	Values recovered from UCB

5.4.1.40 HOST PFLASH bank PpA erase counter priority configuration 0

GP_HOST_PFECPRIOpA0 (p=0-5) Offset address: 015C$_H$+p*80$_H$

HOST PFLASH bank PpA erase counter priority configuration 0 Reset values see: Table 490

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
S31LH	S30LH	S29LH	S28LH	S27LH	S26LH	S25LH	S24LH	S23LH	S22LH	S21LH	S20LH	S19LH	S18LH	S17LH	S16LH
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
S15LH	S14LH	S13LH	S12LH	S11LH	S10LH	S9LH	S8LH	S7LH	S6LH	S5LH	S4LH	S3LH	S2LH	S1LH	S0LH
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
SxLH (x=0-31)	x	rh	**PFLASH pA sector x erase counter priority** These bits indicates whether PFLASH pA sector x has high or low priority for erase counter recording 0_B Low priority is configured for sector x 1_B High priority is configured for sector x

Table 490 Reset values of GP_HOST_PFECPRIOpA0 (p=0-5)

Reset type	Reset value	Note
Application Reset	0000 0000_H	
After Boot-FW Value	---- $----_H$	Values recovered from UCB

5.4.1.41 HOST PFLASH bank PpA erase counter priority configuration 1

GP_HOST_PFECPRIOpA1 (p=0-5) Offset address: 0160_H+p*80_H

HOST PFLASH bank PpA erase counter priority configuration 1 Reset values see: Table 491

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
S63LH	S62LH	S61LH	S60LH	S59LH	S58LH	S57LH	S56LH	S55LH	S54LH	S53LH	S52LH	S51LH	S50LH	S49LH	S48LH
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
S47LH	S46LH	S45LH	S44LH	S43LH	S42LH	S41LH	S40LH	S39LH	S38LH	S37LH	S36LH	S35LH	S34LH	S33LH	S32LH
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
SxLH (x=32-63)	x-32	rh	**PFLASH pA sector x erase counter priority** These bits indicates whether PFLASH pA sector x has high or low priority for erase counter recording 0_B Low priority is configured for sector x 1_B High priority is configured for sector x

Table 491 Reset values of GP_HOST_PFECPRIOpA1 (p=0-5)

Reset type	Reset value	Note
Application Reset	0000 0000$_H$	
After Boot-FW Value	---- ----$_H$	Values recovered from UCB

5.4.1.42 HOST PFLASH bank PpA erase counter priority configuration 2

GP_HOST_PFECPRIOpA2 (p=0-5) Offset address: 0164$_H$+p*80$_H$

HOST PFLASH bank PpA erase counter priority configuration 2 Reset values see: Table 492

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
S95LH	S94LH	S93LH	S92LH	S91LH	S90LH	S89LH	S88LH	S87LH	S86LH	S85LH	S84LH	S83LH	S82LH	S81LH	S80LH
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
S79LH	S78LH	S77LH	S76LH	S75LH	S74LH	S73LH	S72LH	S71LH	S70LH	S69LH	S68LH	S67LH	S66LH	S65LH	S64LH
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
SxLH (x=64-95)	x-64	rh	**PFLASH pA sector x erase counter priority** These bits indicates whether PFLASH pA sector x has high or low priority for erase counter recording 0_B Low priority is configured for sector x 1_B High priority is configured for sector x

Table 492 Reset values of GP_HOST_PFECPRIOpA2 (p=0-5)

Reset type	Reset value	Note
Application Reset	0000 0000$_H$	
After Boot-FW Value	---- ----$_H$	Values recovered from UCB

5.4.1.43 HOST PFLASH bank PpA erase counter priority configuration 3

GP_HOST_PFECPRIOpA3 (p=0-5) Offset address: 0168$_H$+p*80$_H$

HOST PFLASH bank PpA erase counter priority configuration 3 Reset values see: Table 493

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
S127LH	S126LH	S125LH	S124LH	S123LH	S122LH	S121LH	S120LH	S119LH	S118LH	S117LH	S116LH	S115LH	S114LH	S113LH	S112LH
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
S111LH	S110LH	S109LH	S108LH	S107LH	S106LH	S105LH	S104LH	S103LH	S102LH	S101LH	S100LH	S99LH	S98LH	S97LH	S96LH
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
SxLH (x=96-127)	x-96	rh	**PFLASH pA sector x erase counter priority** These bits indicates whether PFLASH pA sector x has high or low priority for erase counter recording 0_B Low priority is configured for sector x 1_B High priority is configured for sector x

Table 493 Reset values of GP_HOST_PFECPRIOpA3 (p=0-5)

Reset type	Reset value	Note
Application Reset	0000 0000_H	
After Boot-FW Value	---- $----_H$	Values recovered from UCB

5.4.1.44 HOST PFLASH bank PpB write protection configuration 0

GP_HOST_PFPROCONWPpB0 (p=0-5) Offset address: $016C_H+p*80_H$

HOST PFLASH bank PpB write protection configuration 0 Reset values see: Table 494

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
S31WP	S30WP	S29WP	S28WP	S27WP	S26WP	S25WP	S24WP	S23WP	S22WP	S21WP	S20WP	S19WP	S18WP	S17WP	S16WP
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
S15WP	S14WP	S13WP	S12WP	S11WP	S10WP	S9WP	S8WP	S7WP	S6WP	S5WP	S4WP	S3WP	S2WP	S1WP	S0WP
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
SxWP (x=0-31)	x	rh	**PFLASH pB sector x write protection** These bits indicates whether PFLASH pB sector x is write protected 0_B No write protection is configured for sector x 1_B Write protection is configured for sector x

Table 494 Reset values of GP_HOST_PFPROCONWPpB0 (p=0-5)

Reset type	Reset value	Note
Application Reset	0000 0000$_H$	
After Boot-FW Value	---- ----$_H$	Values recovered from UCB

5.4.1.45 HOST PFLASH bank PpB write protection configuration 1

GP_HOST_PFPROCONWPpB1 (p=0-5) Offset address: 0170$_H$+p*80$_H$

HOST PFLASH bank PpB write protection configuration 1 Reset values see: Table 495

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
S63WP	S62WP	S61WP	S60WP	S59WP	S58WP	S57WP	S56WP	S55WP	S54WP	S53WP	S52WP	S51WP	S50WP	S49WP	S48WP
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
S47WP	S46WP	S45WP	S44WP	S43WP	S42WP	S41WP	S40WP	S39WP	S38WP	S37WP	S36WP	S35WP	S34WP	S33WP	S32WP
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
SxWP (x=32-63)	x-32	rh	**PFLASH pB sector x write protection** These bits indicates whether PFLASH pB sector x is write protected 0_B No write protection is configured for sector x 1_B Write protection is configured for sector x

Table 495 Reset values of GP_HOST_PFPROCONWPpB1 (p=0-5)

Reset type	Reset value	Note
Application Reset	0000 0000$_H$	
After Boot-FW Value	---- ----$_H$	Values recovered from UCB

5.4.1.46 HOST PFLASH bank PpB write protection configuration 2

GP_HOST_PFPROCONWPpB2 (p=0-5) Offset address: 0174$_H$+p*80$_H$

HOST PFLASH bank PpB write protection configuration 2 Reset values see: Table 496

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
S95WP	S94WP	S93WP	S92WP	S91WP	S90WP	S89WP	S88WP	S87WP	S86WP	S85WP	S84WP	S83WP	S82WP	S81WP	S80WP
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
S79WP	S78WP	S77WP	S76WP	S75WP	S74WP	S73WP	S72WP	S71WP	S70WP	S69WP	S68WP	S67WP	S66WP	S65WP	S64WP
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
SxWP (x=64-95)	x-64	rh	**PFLASH pB sector x write protection** These bits indicates whether PFLASH pB sector x is write protected 0_B No write protection is configured for sector x 1_B Write protection is configured for sector x

Table 496 Reset values of GP_HOST_PFPROCONWPpB2 (p=0-5)

Reset type	Reset value	Note
Application Reset	0000 0000$_H$	
After Boot-FW Value	---- ----$_H$	Values recovered from UCB

5.4.1.47 HOST PFLASH bank PpB write protection configuration 3

GP_HOST_PFPROCONWPpB3 (p=0-5)

HOST PFLASH bank PpB write protection configuration 3

Offset address: 0178$_H$+p*80$_H$

Reset values see: Table 497

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
S127WP	S126WP	S125WP	S124WP	S123WP	S122WP	S121WP	S120WP	S119WP	S118WP	S117WP	S116WP	S115WP	S114WP	S113WP	S112WP
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
S111WP	S110WP	S109WP	S108WP	S107WP	S106WP	S105WP	S104WP	S103WP	S102WP	S101WP	S100WP	S99WP	S98WP	S97WP	S96WP
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
SxWP (x=96-127)	x-96	rh	**PFLASH pB sector x write protection** These bits indicates whether PFLASH pB sector x is write protected 0_B No write protection is configured for sector x 1_B Write protection is configured for sector x

Table 497 **Reset values of** GP_HOST_PFPROCONWPpB3 (p=0-5)

Reset type	Reset value	Note
Application Reset	0000 0000$_H$	
After Boot-FW Value	---- ----$_H$	Values recovered from UCB

5.4.1.48 HOST PFLASH bank PpB permanent write protection configuration 0

GP_HOST_PFPROCONPWPpB0 (p=0-5) Offset address: 017C$_H$+p*80$_H$

HOST PFLASH bank PpB permanent write protection configuration 0 Reset values see: Table 498

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
S31PWP	S30PWP	S29PWP	S28PWP	S27PWP	S26PWP	S25PWP	S24PWP	S23PWP	S22PWP	S21PWP	S20PWP	S19PWP	S18PWP	S17PWP	S16PWP
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
S15PWP	S14PWP	S13PWP	S12PWP	S11PWP	S10PWP	S9PWP	S8PWP	S7PWP	S6PWP	S5PWP	S4PWP	S3PWP	S2PWP	S1PWP	S0PWP
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
SxPWP (x=0-31)	x	rh	**PFLASH pB sector x permanent write protection** These bits indicates whether PFLASH pB sector x is permanently write protected 0_B No PWP protection is configured for sector x 1_B PWP protection is configured for sector x

Table 498 **Reset values of** GP_HOST_PFPROCONPWPpB0 (p=0-5)

Reset type	Reset value	Note
Application Reset	0000 0000$_H$	
After Boot-FW Value	---- ----$_H$	Values recovered from UCB

5.4.1.49 HOST PFLASH bank PpB permanent write protection configuration 1

GP_HOST_PFPROCONPWPpB1 (p=0-5) Offset address: 0180$_H$+p*80$_H$

HOST PFLASH bank PpB permanent write protection configuration 1 Reset values see: Table 499

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
S63PWP	S62PWP	S61PWP	S60PWP	S59PWP	S58PWP	S57PWP	S56PWP	S55PWP	S54PWP	S53PWP	S52PWP	S51PWP	S50PWP	S49PWP	S48PWP
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
S47PWP	S46PWP	S45PWP	S44PWP	S43PWP	S42PWP	S41PWP	S40PWP	S39PWP	S38PWP	S37PWP	S36PWP	S35PWP	S34PWP	S33PWP	S32PWP
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
SxPWP (x=32-63)	x-32	rh	**PFLASH pB sector x permanent write protection** These bits indicates whether PFLASH pB sector x is permanently write protected 0_B No PWP protection is configured for sector x 1_B PWP protection is configured for sector x

Table 499 Reset values of GP_HOST_PFPROCONPWPpB1 (p=0-5)

Reset type	Reset value	Note
Application Reset	$0000\ 0000_H$	
After Boot-FW Value	$----\ ----_H$	Values recovered from UCB

5.4.1.50 HOST PFLASH bank PpB permanent write protection configuration 2

GP_HOST_PFPROCONPWPpB2 (p=0-5) Offset address: 0184_H+p*80_H

HOST PFLASH bank PpB permanent write protection configuration 2 Reset values see: Table 500

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
S95PWP	S94PWP	S93PWP	S92PWP	S91PWP	S90PWP	S89PWP	S88PWP	S87PWP	S86PWP	S85PWP	S84PWP	S83PWP	S82PWP	S81PWP	S80PWP
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
S79PWP	S78PWP	S77PWP	S76PWP	S75PWP	S74PWP	S73PWP	S72PWP	S71PWP	S70PWP	S69PWP	S68PWP	S67PWP	S66PWP	S65PWP	S64PWP
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
SxPWP (x=64-95)	x-64	rh	**PFLASH pB sector x permanent write protection** These bits indicates whether PFLASH pB sector x is permanently write protected 0_B No PWP protection is configured for sector x 1_B PWP protection is configured for sector x

Table 500 **Reset values of** GP_HOST_PFPROCONPWPpB2 (p=0-5)

Reset type	Reset value	Note
Application Reset	0000 0000_H	
After Boot-FW Value	---- $----_H$	Values recovered from UCB

5.4.1.51 HOST PFLASH bank PpB permanent write protection configuration 3

GP_HOST_PFPROCONPWPpB3 (p=0-5)

Offset address: 0188_H+p*80_H

HOST PFLASH bank PpB permanent write protection configuration 3

Reset values see: Table 501

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
S127 PWP	S126 PWP	S125 PWP	S124 PWP	S123 PWP	S122 PWP	S121 PWP	S120 PWP	S119 PWP	S118 PWP	S117 PWP	S116 PWP	S115 PWP	S114 PWP	S113 PWP	S112 PWP
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
S111 PWP	S110 PWP	S109 PWP	S108 PWP	S107 PWP	S106 PWP	S105 PWP	S104 PWP	S103 PWP	S102 PWP	S101 PWP	S100 PWP	S99P WP	S98P WP	S97P WP	S96P WP
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
SxPWP (x=96-127)	x-96	rh	**PFLASH pB sector x permanent write protection** These bits indicates whether PFLASH pB sector x is permanently write protected 0_B No PWP protection is configured for sector x 1_B PWP protection is configured for sector x

Table 501 **Reset values of** GP_HOST_PFPROCONPWPpB3 (p=0-5)

Reset type	Reset value	Note
Application Reset	0000 0000_H	
After Boot-FW Value	---- $----_H$	Values recovered from UCB

5.4.1.52 HOST PFLASH bank PpB permanent write once protection configuration 0

GP_HOST_PFPROCONPWOPpB0 (p=0-5)

Offset address: $018C_H+p*80_H$

HOST PFLASH bank PpB permanent write once protection configuration 0

Reset values see: Table 502

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
S31PWOP	S30PWOP	S29PWOP	S28PWOP	S27PWOP	S26PWOP	S25PWOP	S24PWOP	S23PWOP	S22PWOP	S21PWOP	S20PWOP	S19PWOP	S18PWOP	S17PWOP	S16PWOP
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
S15PWOP	S14PWOP	S13PWOP	S12PWOP	S11PWOP	S10PWOP	S9PWOP	S8PWOP	S7PWOP	S6PWOP	S5PWOP	S4PWOP	S3PWOP	S2PWOP	S1PWOP	S0PWOP
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
SxPWOP (x=0-31)	x	rh	**PFLASH pB sector x permanent write once protection** These bits indicates whether PFLASH pB sector x is permanently write once protected 0_B No permanent write once protection is configured for sector x 1_B Permanent write once protection is configured for sector x

Table 502 Reset values of GP_HOST_PFPROCONPWOPpB0 (p=0-5)

Reset type	Reset value	Note
Application Reset	0000 0000_H	
After Boot-FW Value	---- $----_H$	Values recovered from UCB

5.4.1.53 HOST PFLASH bank PpB permanent write once protection configuration 1

GP_HOST_PFPROCONPWOPpB1 (p=0-5)

Offset address: 0190_H+p*80_H

HOST PFLASH bank PpB permanent write once protection configuration 1

Reset values see: Table 503

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
S63PWOP	S62PWOP	S61PWOP	S60PWOP	S59PWOP	S58PWOP	S57PWOP	S56PWOP	S55PWOP	S54PWOP	S53PWOP	S52PWOP	S51PWOP	S50PWOP	S49PWOP	S48PWOP
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
S47PWOP	S46PWOP	S45PWOP	S44PWOP	S43PWOP	S42PWOP	S41PWOP	S40PWOP	S39PWOP	S38PWOP	S37PWOP	S36PWOP	S35PWOP	S34PWOP	S33PWOP	S32PWOP
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
SxPWOP (x=32-63)	x-32	rh	**PFLASH pB sector x permanent write once protection** These bits indicates whether PFLASH pB sector x is permanently write once protected 0_B No permanent write once protection is configured for sector x 1_B Permanent write once protection is configured for sector x

Table 503 **Reset values of** GP_HOST_PFPROCONPWOPpB1 (p=0-5)

Reset type	Reset value	Note
Application Reset	0000 0000$_H$	
After Boot-FW Value	---- ----$_H$	Values recovered from UCB

5.4.1.54 HOST PFLASH bank PpB permanent write once protection configuration 2

GP_HOST_PFPROCONPWOPpB2 (p=0-5)

Offset address: 0194$_H$+p*80$_H$

HOST PFLASH bank PpB permanent write once protection configuration 2

Reset values see: Table 504

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
S95PWOP	S94PWOP	S93PWOP	S92PWOP	S91PWOP	S90PWOP	S89PWOP	S88PWOP	S87PWOP	S86PWOP	S85PWOP	S84PWOP	S83PWOP	S82PWOP	S81PWOP	S80PWOP
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
S79PWOP	S78PWOP	S77PWOP	S76PWOP	S75PWOP	S74PWOP	S73PWOP	S72PWOP	S71PWOP	S70PWOP	S69PWOP	S68PWOP	S67PWOP	S66PWOP	S65PWOP	S64PWOP
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
SxPWOP (x=64-95)	x-64	rh	**PFLASH pB sector x permanent write once protection** These bits indicates whether PFLASH pB sector x is permanently write once protected 0_B No permanent write once protection is configured for sector x 1_B Permanent write once protection is configured for sector x

Table 504 **Reset values of** GP_HOST_PFPROCONPWOPpB2 (p=0-5)

Reset type	Reset value	Note
Application Reset	0000 0000$_H$	
After Boot-FW Value	---- ----$_H$	Values recovered from UCB

5.4.1.55 HOST PFLASH bank PpB permanent write once protection configuration 3

GP_HOST_PFPROCONPWOPpB3 (p=0-5)

Offset address: 0198$_H$+p*80$_H$

HOST PFLASH bank PpB permanent write once protection configuration 3

Reset values see: Table 505

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
S127PWOP	S126PWOP	S125PWOP	S124PWOP	S123PWOP	S122PWOP	S121PWOP	S120PWOP	S119PWOP	S118PWOP	S117PWOP	S116PWOP	S115PWOP	S114PWOP	S113PWOP	S112PWOP
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
S111PWOP	S110PWOP	S109PWOP	S108PWOP	S107PWOP	S106PWOP	S105PWOP	S104PWOP	S103PWOP	S102PWOP	S101PWOP	S100PWOP	S99PWOP	S98PWOP	S97PWOP	S96PWOP
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
SxPWOP (x=96-127)	x-96	rh	**PFLASH pB sector x permanent write once protection** These bits indicates whether PFLASH pB sector x is permanently write once protected 0_B No permanent write once protection is configured for sector x 1_B Permanent write once protection is configured for sector x

Table 505 Reset values of GP_HOST_PFPROCONPWOPpB3 (p=0-5)

Reset type	Reset value	Note
Application Reset	0000 0000_H	
After Boot-FW Value	---- $----_H$	Values recovered from UCB

5.4.1.56 HOST PFLASH bank PpB erase counter priority configuration 0

GP_HOST_PFECPRIOpB0 (p=0-5)

HOST PFLASH bank PpB erase counter priority configuration 0

Offset address: $019C_H+p*80_H$

Reset values see: Table 506

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
S31LH	S30LH	S29LH	S28LH	S27LH	S26LH	S25LH	S24LH	S23LH	S22LH	S21LH	S20LH	S19LH	S18LH	S17LH	S16LH
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
S15LH	S14LH	S13LH	S12LH	S11LH	S10LH	S9LH	S8LH	S7LH	S6LH	S5LH	S4LH	S3LH	S2LH	S1LH	S0LH
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
SxLH (x=0-31)	x	rh	**PFLASH pB sector x erase counter priority** These bits indicates whether PFLASH pB sector x has high or low priority for erase counter recording 0_B Low priority is configured for sector x 1_B High priority is configured for sector x

Table 506 Reset values of GP_HOST_PFECPRIOpB0 (p=0-5)

Reset type	Reset value	Note
Application Reset	0000 0000$_H$	
After Boot-FW Value	---- ----$_H$	Values recovered from UCB

5.4.1.57 HOST PFLASH bank PpB erase counter priority configuration 1

GP_HOST_PFECPRIOpB1 (p=0-5) Offset address: 01A0$_H$+p*80$_H$

HOST PFLASH bank PpB erase counter priority configuration 1 Reset values see: Table 507

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
S63LH	S62LH	S61LH	S60LH	S59LH	S58LH	S57LH	S56LH	S55LH	S54LH	S53LH	S52LH	S51LH	S50LH	S49LH	S48LH
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
S47LH	S46LH	S45LH	S44LH	S43LH	S42LH	S41LH	S40LH	S39LH	S38LH	S37LH	S36LH	S35LH	S34LH	S33LH	S32LH
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
SxLH (x=32-63)	x-32	rh	**PFLASH pB sector x erase counter priority** These bits indicates whether PFLASH pB sector x has high or low priority for erase counter recording 0_B Low priority is configured for sector x 1_B High priority is configured for sector x

Table 507 Reset values of GP_HOST_PFECPRIOpB1 (p=0-5)

Reset type	Reset value	Note
Application Reset	0000 0000$_H$	
After Boot-FW Value	---- ----$_H$	Values recovered from UCB

5.4.1.58 HOST PFLASH bank PpB erase counter priority configuration 2

GP_HOST_PFECPRIOpB2 (p=0-5) Offset address: 01A4$_H$+p*80$_H$

HOST PFLASH bank PpB erase counter priority configuration 2 Reset values see: Table 508

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
S95LH	S94LH	S93LH	S92LH	S91LH	S90LH	S89LH	S88LH	S87LH	S86LH	S85LH	S84LH	S83LH	S82LH	S81LH	S80LH
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
S79LH	S78LH	S77LH	S76LH	S75LH	S74LH	S73LH	S72LH	S71LH	S70LH	S69LH	S68LH	S67LH	S66LH	S65LH	S64LH
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
SxLH (x=64-95)	x-64	rh	**PFLASH pB sector x erase counter priority** These bits indicates whether PFLASH pB sector x has high or low priority for erase counter recording 0_B Low priority is configured for sector x 1_B High priority is configured for sector x

Table 508 Reset values of GP_HOST_PFECPRIOpB2 (p=0-5)

Reset type	Reset value	Note
Application Reset	0000 0000$_H$	
After Boot-FW Value	---- ----$_H$	Values recovered from UCB

5.4.1.59 HOST PFLASH bank PpB erase counter priority configuration 3

GP_HOST_PFECPRIOpB3 (p=0-5)

Offset address: 01A8$_H$+p*80$_H$

HOST PFLASH bank PpB erase counter priority configuration 3

Reset values see: Table 509

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
S127LH	S126LH	S125LH	S124LH	S123LH	S122LH	S121LH	S120LH	S119LH	S118LH	S117LH	S116LH	S115LH	S114LH	S113LH	S112LH
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
S111LH	S110LH	S109LH	S108LH	S107LH	S106LH	S105LH	S104LH	S103LH	S102LH	S101LH	S100LH	S99LH	S98LH	S97LH	S96LH
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
SxLH (x=96-127)	x-96	rh	**PFLASH pB sector x erase counter priority** These bits indicates whether PFLASH pB sector x has high or low priority for erase counter recording 0_B Low priority is configured for sector x 1_B High priority is configured for sector x

Table 509 Reset values of GP_HOST_PFECPRIOpB3 (p=0-5)

Reset type	Reset value	Note
Application Reset	0000 0000$_H$	
After Boot-FW Value	---- ----$_H$	Values recovered from UCB

5.4.1.60 HOST UCB password protection register

This register reports the state of the HOST FLASH protection and contains protection relevant control fields.

GP_HOST_PROTECT Offset address: 052C$_H$

HOST UCB password protection register Application Reset value: 0000 0000$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		0		PRODISP5B	PRODISP5A	PRODISP4B	PRODISP4A	PRODISP3B	PRODISP3A	PRODISP2B	PRODISP2A	PRODISP1B	PRODISP1A	PRODISP0B	PRODISP0A
r		r		rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PRODISP	PRODISD	PRODISFL	0					PRODISUDATA1	PRODISUDATA0	0		PRODISECPRIO	PRODISSWAP	PRODISUSERCFG	PRODISBMHD
rh	rh	rh	r					rh	rh	r		rh	rh	rh	rh

Field	Bits	Type	Description
PRODISBMHD	0	rh	**UCB_RTC_BMHD protection disabled** The protection configured by UCB_RTC_BMHD0, UCB_RTC_BMHD1, UCB_RTC_BMHD2 and UCB_RTC_BMHD3 was successfully disabled by supplying the correct password to "Disable Protection". Note: Cleared with command "Resume Protection".
PRODISUSERCFG	1	rh	**UCB_RTC_USERCFG protection disabled** The protection configured by UCB_RTC_USERCFG_ORIG and UCB_RTC_USERCFG_COPY was successfully disabled by supplying the correct password to "Disable Protection". Note: Cleared with command "Resume Protection".
PRODISSWAP	2	rh	**UCB_RTC_SWAP protection disabled** The protection configured by UCB_RTC_SWAP_ORIG and UCB_RTC_SWAP_COPY was successfully disabled by supplying the correct password to "Disable Protection". Note: Cleared with command "Resume Protection".
PRODISECPRIO	3	rh	**UCB_RTC_ECPRIO protection disabled** The protection configured by UCB_RTC_ECPRIO_ORIG and UCB_RTC_ECPRIO_COPY was successfully disabled by supplying the correct password to "Disable Protection". Note: Cleared with command "Resume Protection".

(table continues...)

(continued)

Field	Bits	Type	Description
PRODISUDATA0	6	rh	**UCB_USER_DATA0 protection disabled** The protection configured by UCB_USER_DATA0 was successfully disabled by supplying the correct password to "Disable Protection". Note: Cleared with command "Resume Protection".
PRODISUDATA1	7	rh	**UCB_USER_DATA1 protection disabled** The protection configured by UCB_USER_DATA1 was successfully disabled by supplying the correct password to "Disable Protection". Note: Cleared with command "Resume Protection".
PRODISFL	13	rh	**HOST FLASH Protection Disabled** The read and write protections for HOST PFLASH and HOST DFLASH configured by UCB_RTC_FLASH_ORIG and UCB_RTC_FLASH_COPY were successfully disabled by supplying the correct password to "Disable Protection". Note: Cleared with command "Resume Protection". Note: When this bit is set, the protection status of all the HOST PFLASHs is not reflected in bitfields PRODISPxA and PRODISPxB.
PRODISD	14	rh	**HOST DFLASH Protection Disabled** The read and write protections for HOST DFLASH configured by UCB_RTC_FLASH_ORIG and UCB_RTC_FLASH_COPY were successfully disabled by supplying the correct password to "Disable Protection". Note: Cleared with command "Resume Protection".
PRODISP	15	rh	**HOST PFLASH Protection Disabled** The read and write protections for HOST PFLASH configured by UCB_RTC_FLASH_ORIG and UCB_RTC_FLASH_COPY were successfully disabled by supplying the correct password to "Disable Protection". Note: Cleared with command "Resume Protection". Note: When this bit is set, the protection status of all the HOST PFLASHs is not reflected in bitfields PRODISPxA and PRODISPxB.
PRODISPxA (x=0-5)	2*x+16	rh	**HOST PFLASH PxA protection disable** The write protections configured for HOST PFLASHxA bank by UCB_RTC_FLASH_ORIG and UCB_RTC_FLASH_COPY were successfully disabled by supplying the correct password to "Disable Protection". Note: Cleared with command "Resume Protection".
PRODISPxB (x=0-5)	2*x+17	rh	**HOST PFLASH PxB protection disable** The write protections configured for HOST PFLASHxB bank by UCB_RTC_FLASH_ORIG and UCB_RTC_FLASH_COPY were successfully disabled by supplying the correct password to "Disable Protection". Note: Cleared with command "Resume Protection".
0	5:4, 12:8, 31:30, 29:28	r	**Reserved** Read as 0; shall be written with 0.

5.4.1.61 HOST UCB confirmation codes register A

GP_HOST_CONFIRMA Offset address: 0530_H

HOST UCB confirmation codes register A Reset values see: Table 510

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								PROINBMHD3		PROINBMHD2		PROINBMHD1		PROINBMHD0	
r								rh		rh		rh		rh	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PROINUDATA1		PROINUDATA0		PROINADCTRIM		PROINRTCCFG		PROININFO		PROINTEST		PROINMOCFG		PROINMOID	
rh		rh		rh		rh		rh		rh		rh		rh	

Field	Bits	Type	Description
PROINMOID	1:0	rh	**UCB_RTC_MOID confirmation state** This bit reflects the confirmation codes of UCB_RTC_MOID. 00_B UCB was not read yet. 01_B UCB state is UNLOCKED. 10_B UCB state is CONFIRMED. 11_B UCB state is ERRORED.
PROINMOCFG	3:2	rh	**UCB_RTC_MOCFG confirmation state** This bit reflects the confirmation codes of UCB_RTC_MOCFG. 00_B UCB was not read yet. 01_B UCB state is UNLOCKED. 10_B UCB state is CONFIRMED. 11_B UCB state is ERRORED.
PROINTEST	5:4	rh	**UCB_TEST confirmation state** This bit reflects the confirmation codes of UCB_TEST. 00_B UCB was not read yet. 01_B UCB state is UNLOCKED. 10_B UCB state is CONFIRMED. 11_B UCB state is ERRORED.
PROININFO	7:6	rh	**UCB_INFO confirmation state** This bit reflects the confirmation codes of UCB_INFO. 00_B UCB was not read yet. 01_B UCB state is UNLOCKED. 10_B UCB state is CONFIRMED. 11_B UCB state is ERRORED.

(table continues...)

(continued)

Field	Bits	Type	Description
PROINRTCCFG	9:8	rh	**UCB_RTCCFG Confirmation** This bit reflects the confirmation codes of UCB_RTCCFG. 00_B UCB was not read yet. 01_B UCB state is UNLOCKED. 10_B UCB state is CONFIRMED. 11_B UCB state is ERRORED.
PROINADCTRIM	11:10	rh	**UCB_ADCTRIM confirmation state** This bit reflects the confirmation codes of UCB_ADCTRIM. 00_B UCB was not read yet. 01_B UCB state is UNLOCKED. 10_B UCB state is CONFIRMED. 11_B UCB state is ERRORED.
PROINUDATA0	13:12	rh	**UCB_USER_DATA0 confirmation state** This bit reflects the confirmation codes of UCB_USER_DATA0. Note: This is a single-UCB. If in ERRORED state, the UCB cannot be recovered. 00_B UCB was not read yet. 01_B UCB state is UNLOCKED. 10_B UCB state is CONFIRMED. 11_B UCB state is ERRORED.
PROINUDATA1	15:14	rh	**UCB_USER_DATA1 confirmation state** This bit reflects the confirmation codes of UCB_USER_DATA1. Note: This is a single-UCB. If in ERRORED state, the UCB cannot be recovered. 00_B UCB was not read yet. 01_B UCB state is UNLOCKED. 10_B UCB state is CONFIRMED. 11_B UCB state is ERRORED.
PROINBMHD0	17:16	rh	**UCB_RTC_BMHD0 confirmation state** This bit reflects the confirmation codes of UCB_RTC_BMHD0. Note: if the confirmation code of BMHD0 is ERRORED, all BMHDs cannot be unlocked even BMHD1, BMHD2 or BMHD3 are in CONFIRMED state 00_B UCB was not read yet. 01_B UCB state is UNLOCKED. 10_B UCB state is CONFIRMED. 11_B UCB state is ERRORED.

(table continues...)

(continued)

Field	Bits	Type	Description
PROINBMHD1	19:18	rh	**UCB_RTC_BMHD1 confirmation state** This bit reflects the confirmation codes of UCB_RTC_BMHD1. Note: if the confirmation code of BMHD0 is ERRORED, this BMHD cannot be unlocked even if it is in CONFIRMED state 00_B UCB was not read yet. 01_B UCB state is UNLOCKED. 10_B UCB state is CONFIRMED. 11_B UCB state is ERRORED.
PROINBMHD2	21:20	rh	**UCB_RTC_BMHD2 confirmation state** This bit reflects the confirmation codes of UCB_RTC_BMHD2. Note: if the confirmation code of BMHD0 is ERRORED, this BMHD cannot be unlocked even if it is in CONFIRMED state 00_B UCB was not read yet. 01_B UCB state is UNLOCKED. 10_B UCB state is CONFIRMED. 11_B UCB state is ERRORED.
PROINBMHD3	23:22	rh	**UCB_RTC_BMHD3 confirmation state** This bit reflects the confirmation codes of UCB_RTC_BMHD3. Note: if the confirmation code of BMHD0 is ERRORED, this BMHD cannot be unlocked even if it is in CONFIRMED state 00_B UCB was not read yet. 01_B UCB state is UNLOCKED. 10_B UCB state is CONFIRMED. 11_B UCB state is ERRORED.
0	31:24	r	**Reserved** Read as 0; shall be written with 0.

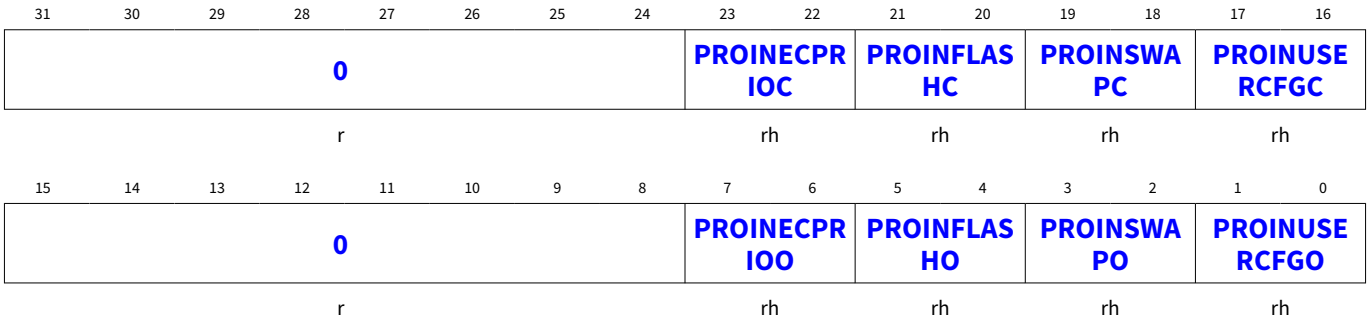
Table 510 Reset values of GP_HOST_CONFIRMA

Reset type	Reset value	Note
Application Reset	$0000\ 0000_H$	
After Boot-FW Value	$XX{-}{-}\ {-}{-}{-}{-}_H$	

5.4.1.62 HOST UCB confirmation codes register B

GP_HOST_CONFIRMB Offset address: 0534_H

HOST UCB confirmation codes register B Reset values see: Table 511



Field	Bits	Type	Description
PROINUSERCFGO	1:0	rh	**UCB_RTC_USERCFG_ORIG confirmation state** This bit reflects the confirmation codes of UCB_RTC_USERCFG_ORIG. 00_B ORIG UCB was not read yet. 01_B ORIG UCB state is UNLOCKED. 10_B ORIG UCB state is CONFIRMED. 11_B ORIG UCB state is ERRORED.
PROINSWAPO	3:2	rh	**UCB_RTC_SWAP_ORIG confirmation state** This bit reflects the confirmation codes of UCB_RTC_SWAP_ORIG. 00_B ORIG UCB was not read yet. 01_B ORIG UCB state is UNLOCKED. 10_B ORIG UCB state is CONFIRMED. 11_B ORIG UCB state is ERRORED.
PROINFLASHO	5:4	rh	**UCB_RTC_FLASH_ORIG confirmation state** This bit reflects the confirmation codes of UCB_RTC_FLASH_ORIG. 00_B ORIG UCB was not read yet. 01_B ORIG UCB state is UNLOCKED. 10_B ORIG UCB state is CONFIRMED. 11_B ORIG UCB state is ERRORED.
PROINECPRIOO	7:6	rh	**UCB_RTC_ECPRIO_ORIG confirmation state** This bit reflects the confirmation codes of UCB_RTC_ECPRIO_ORIG. 00_B ORIG UCB was not read yet. 01_B ORIG UCB state is UNLOCKED. 10_B ORIG UCB state is CONFIRMED. 11_B ORIG UCB state is ERRORED.
PROINUSERCFGC	17:16	rh	**UCB_RTC_USERCFG_COPY confirmation state** This bit reflects the confirmation codes of UCB_RTC_USERCFG_COPY. 00_B COPY UCB was not read yet. 01_B COPY UCB state is UNLOCKED. 10_B COPY UCB state is CONFIRMED. 11_B COPY UCB state is ERRORED.

(table continues...)

(continued)

Field	Bits	Type	Description
PROINSWAPC	19:18	rh	**UCB_RTC_SWAP_COPY confirmation state** This bit reflects the confirmation codes of UCB_RTC_SWAP_COPY. 00_B COPY UCB was not read yet. 01_B COPY UCB state is UNLOCKED. 10_B COPY UCB state is CONFIRMED. 11_B COPY UCB state is ERRORED.
PROINFLASHC	21:20	rh	**UCB_RTC_FLASH_COPY confirmation state** This bit reflects the confirmation codes of UCB_RTC_FLASH_COPY. 00_B COPY UCB was not read yet. 01_B COPY UCB state is UNLOCKED. 10_B COPY UCB state is CONFIRMED. 11_B COPY UCB state is ERRORED.
PROINECPRIOC	23:22	rh	**UCB_RTC_ECPRIO_COPY confirmation state** This bit reflects the confirmation codes of UCB_RTC_ECPRIO_COPY. 00_B COPY UCB was not read yet. 01_B COPY UCB state is UNLOCKED. 10_B COPY UCB state is CONFIRMED. 11_B COPY UCB state is ERRORED.
0	15:8, 31:24	r	**Reserved** Read as 0; shall be written with 0.

Table 511 Reset values of GP_HOST_CONFIRMB

Reset type	Reset value	Note
Application Reset	$0000\ 0000_H$	
After Boot-FW Value	$XX\text{--}\ XX\text{--}_H$	

5.4.1.63 HOST UCB confirmation codes register C

GP_HOST_CONFIRMC Offset address: 0538_H

HOST UCB confirmation codes register C Reset values see: Table 512

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0				PROINOTP16C		PROINOTP15C		PROINOTP14C		PROINOTP03C		PROINOTP02C		PROINOTP01C	
r				rh		rh		rh		rh		rh		rh	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				PROINOTP16O		PROINOTP15O		PROINOTP14O		PROINOTP03O		PROINOTP02O		PROINOTP01O	
r				rh		rh		rh		rh		rh		rh	

Field	Bits	Type	Description
PROINOTP01O	1:0	rh	**UCB_RTC_PFLASH_OTP0_ORIG OTP set 1 confirmation state** This bit reflects the confirmation codes of UCB_RTC_PFLASH_OTP0_ORIG for OTP set 1. 00_B ORIG OTP set was not read yet. 01_B ORIG OTP set state is UNLOCKED. 10_B ORIG OTP set state is CONFIRMED. 11_B ORIG OTP set state is ERRORED.
PROINOTP02O	3:2	rh	**UCB_RTC_PFLASH_OTP0_ORIG OTP set 2 confirmation state** This bit reflects the confirmation codes of UCB_RTC_PFLASH_OTP0_ORIG for OTP set 2. 00_B ORIG OTP set was not read yet. 01_B ORIG OTP set state is UNLOCKED. 10_B ORIG OTP set state is CONFIRMED. 11_B ORIG OTP set state is ERRORED.
PROINOTP03O	5:4	rh	**UCB_RTC_PFLASH_OTP0_ORIG OTP set 3 confirmation state** This bit reflects the confirmation codes of UCB_RTC_PFLASH_OTP0_ORIG for OTP set 3. 00_B ORIG OTP set was not read yet. 01_B ORIG OTP set state is UNLOCKED. 10_B ORIG OTP set state is CONFIRMED. 11_B ORIG OTP set state is ERRORED.
PROINOTP14O	7:6	rh	**UCB_RTC_PFLASH_OTP1_ORIG OTP set 4 confirmation state** This bit reflects the confirmation codes of UCB_RTC_PFLASH_OTP1_ORIG for OTP set 4. 00_B ORIG OTP set was not read yet. 01_B ORIG OTP set state is UNLOCKED. 10_B ORIG OTP set state is CONFIRMED. 11_B ORIG OTP set state is ERRORED.
PROINOTP15O	9:8	rh	**UCB_RTC_PFLASH_OTP1_ORIG OTP set 5 confirmation state** This bit reflects the confirmation codes of UCB_RTC_PFLASH_OTP1_ORIG for OTP set 5. 00_B ORIG OTP set was not read yet. 01_B ORIG OTP set state is UNLOCKED. 10_B ORIG OTP set state is CONFIRMED. 11_B ORIG OTP set state is ERRORED.
PROINOTP16O	11:10	rh	**UCB_RTC_PFLASH_OTP1_ORIG OTP set 6 confirmation state** This bit reflects the confirmation codes of UCB_RTC_PFLASH_OTP1_ORIG for OTP set 6. 00_B ORIG OTP set was not read yet. 01_B ORIG OTP set state is UNLOCKED. 10_B ORIG OTP set state is CONFIRMED. 11_B ORIG OTP set state is ERRORED.

(table continues...)

(continued)

Field	Bits	Type	Description
PROINOTP01C	17:16	rh	**UCB_RTC_PFLASH_OTP0_COPY OTP set 1 confirmation state** This bit reflects the confirmation codes of UCB_RTC_PFLASH_OTP0_COPY for OTP set 1. 00_B COPY OTP set was not read yet. 01_B COPY OTP set state is UNLOCKED. 10_B COPY OTP set state is CONFIRMED. 11_B COPY OTP set state is ERRORED.
PROINOTP02C	19:18	rh	**UCB_RTC_PFLASH_OTP0_COPY OTP set 2 confirmation state** This bit reflects the confirmation codes of UCB_RTC_PFLASH_OTP0_COPY for OTP set 2. 00_B COPY OTP set was not read yet. 01_B COPY OTP set state is UNLOCKED. 10_B COPY OTP set state is CONFIRMED. 11_B COPY OTP set state is ERRORED.
PROINOTP03C	21:20	rh	**UCB_RTC_PFLASH_OTP0_COPY OTP set 3 confirmation state** This bit reflects the confirmation codes of UCB_RTC_PFLASH_OTP0_COPY for OTP set 3. 00_B COPY OTP set was not read yet. 01_B COPY OTP set state is UNLOCKED. 10_B COPY OTP set state is CONFIRMED. 11_B COPY OTP set state is ERRORED.
PROINOTP14C	23:22	rh	**UCB_RTC_PFLASH_OTP1_COPY OTP set 4 confirmation state** This bit reflects the confirmation codes of UCB_RTC_PFLASH_OTP1_COPY for OTP set 4. 00_B COPY OTP set was not read yet. 01_B COPY OTP set state is UNLOCKED. 10_B COPY OTP set state is CONFIRMED. 11_B COPY OTP set state is ERRORED.
PROINOTP15C	25:24	rh	**UCB_RTC_PFLASH_OTP1_COPY OTP set 5 confirmation state** This bit reflects the confirmation codes of UCB_RTC_PFLASH_OTP1_COPY for OTP set 5. 00_B COPY OTP set was not read yet. 01_B COPY OTP set state is UNLOCKED. 10_B COPY OTP set state is CONFIRMED. 11_B COPY OTP set state is ERRORED.
PROINOTP16C	27:26	rh	**UCB_RTC_PFLASH_OTP1_COPY OTP set 6 confirmation state** This bit reflects the confirmation codes of UCB_RTC_PFLASH_OTP1_COPY for OTP set 6. 00_B COPY OTP set was not read yet. 01_B COPY OTP set state is UNLOCKED. 10_B COPY OTP set state is CONFIRMED. 11_B COPY OTP set state is ERRORED.

(table continues...)

(continued)

Field	Bits	Type	Description
0	15:12, 31:28	r	**Reserved** Read as 0; shall be written with 0.

Table 512 Reset values of GP_HOST_CONFIRMC

Reset type	Reset value	Note
Application Reset	0000 0000$_H$	
After Boot-FW Value	X--- X---$_H$	

5.4.1.64 CSRM DFLASH wait cycle register

GP_CSRM_DFWAIT Offset address: 0900$_H$

CSRM DFLASH wait cycle register Reset values see: Table 513

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0											RFLASH				
r											rw				

Field	Bits	Type	Description
RFLASH	4:0	rw	**CSRM DFLASH wait cycles for read** These registers defines the number of FSI clock cycles needed for a CSRM DFLASH, UCB or CFS read access. 00_H Read access takes 1 fFSI clock cycles … $1F_H$ Read access takes 32 fFSI clock cycles
0	31:5	r	**Reserved** Read as 0; shall be written with 0.

Table 513 Reset values of GP_CSRM_DFWAIT

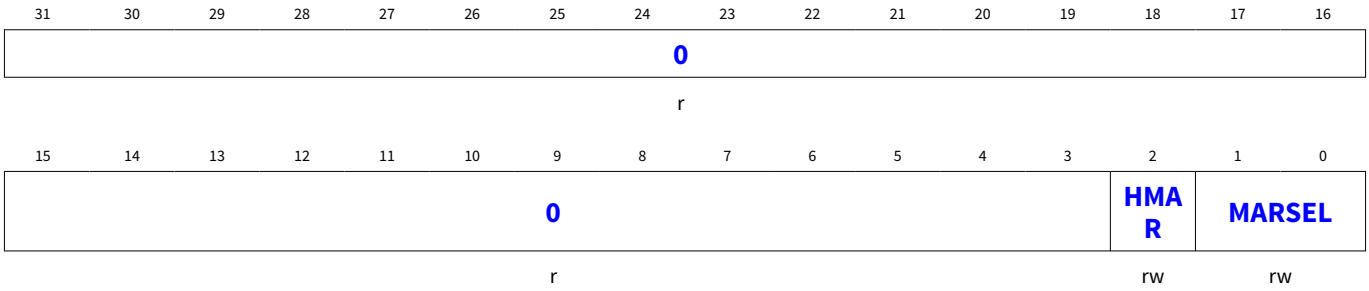
Reset type	Reset value	Note
System Reset	0000 0005$_H$	
After Boot-FW Value	0000 0005$_H$	

5.4.1.65 CSRM DFLASH margin control register

If a change between the read margin settings is done using HMAR or MARSEL, the system must wait a delay time $t_{FL_MarginDel}$ until the next read is requested.

GP_CSRM_DFMARGIN Offset address: 0904$_H$

CSRM DFLASH margin control register Application Reset value: 0000 0000$_H$



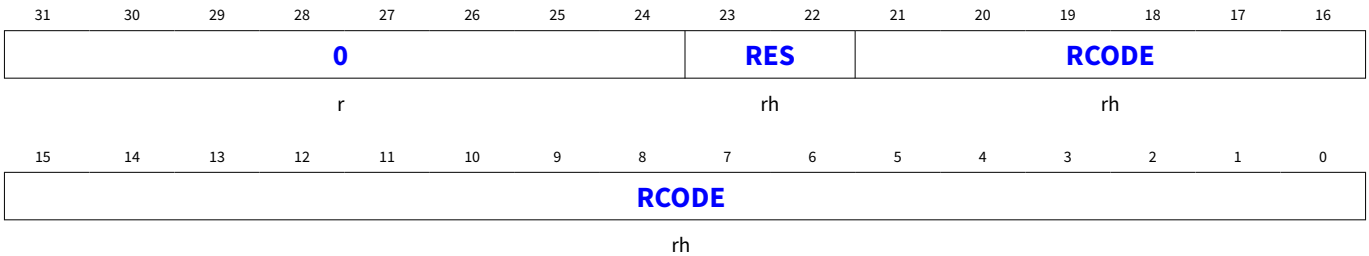
Field	Bits	Type	Description
MARSEL	1:0	rw	**Read margin selection** This register enables to use the hard margins to read out of the CSRM DFLASH, UCB and CFS memory regions. 00_B Enables the standard read margins for reading the CSRM DFLASH, UCB or CFS bank. … 10_B Enables the standard read margins for reading the CSRM DFLASH, UCB or CFS bank. 11_B Enables the read margins chosen by DMU_GP_CSRM_DFMARGIN.HMAR for reading the CSRM DFLASH, UCB or CFS bank
HMAR	2	rw	**Hard margin selection** These registers enable to chose between the thigh-0 and the tight-1 margin for reading out of the CSRM DFLASH, UCB or CFS memory regions. This setting is effective only if DMU_GP_CSRM_DFMARGIN.MARSEL=11_B. 0_B Enables the tight-0 read margins (sub-optimal 0 read as 1) for reading the CSRM DFLASH, UCB or CFS bank. 1_B Enables the tight-1 read margins (sub-optimal 1 read as 0) for reading the CSRM DFLASH, UCB or CFS bank.
0	31:3	r	**Reserved** Read as 0; shall be written with 0.

5.4.1.66 CSRM DFLASH ECC read register

The ECC read register stores the ECC checksum corresponding to the last read on the CSRM DFLASH, UCB or CFS bank when the read is initiated through the DMU SRI slave interface.

GP_CSRM_DFECCR Offset address: 0908_H

CSRM DFLASH ECC read register Application Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
RCODE	21:0	rh	**Error correction code from the last read.** These registers contain the ECC checksum corresponding to the last CSRM DFLASH, UCB or CFS read access.
RES	23:22	rh	**Reserved** Reserved. Internal
0	31:24	r	**Reserved** Read as 0; shall be written with 0.

5.4.1.67 CSRM DFLASH ECC status register

The ECC status register stores the errors found by the ECC corresponding to the last read(s) on the CSRM DFLASH, UCB or CFS bank when the read is initiated through the DMU SRI slave interface.

GP_CSRM_DFECCS Offset address: $090C_H$

CSRM DFLASH ECC status register Reset values see: Table 514

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0							AANER	0				AUCER	ATBER	ADBER	ASBER
r							rh	r				rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0							ANER	0				UCER	TBER	DBER	SBER
r							rh	r				rh	rh	rh	rh

Field	Bits	Type	Description
SBER	0	rh	**Single Bit ECC Error** This register reports a single bit error found in the last read access. 0_B No single bit error was detected by the ECC in the last read access. 1_B A single bit error was detected by the ECC in the last read access.
DBER	1	rh	**Double Bit ECC Error** This register reports a double bit error found in the last read access. 0_B No double bit error was detected by the ECC in the last read access. 1_B A double bit error was detected by the ECC in the last read access.
TBER	2	rh	**Triple Bit ECC Error** This register reports a triple bit error found in the last read access. 0_B No triple bit error was detected by the ECC in the last read access. 1_B A triple bit error was detected by the ECC in the last read access.

(table continues...)

(continued)

Field	Bits	Type	Description
UCER	3	rh	**Uncorrectable ECC Error** This register reports an uncorrectable error found in the last read access. 0_B No uncorrectable error was detected by the ECC in the last read access. 1_B An uncorrectable error was detected by the ECC in the last read access.
ANER	8	rh	**Any ECC error** This register reports an error found in the last read access. 0_B No error was detected by the ECC in the last read access. 1_B An error was detected by the ECC in the last read access.
ASBER	16	rh	**Accumulated Single Bit ECC Error** This register reports that at least one single bit error was found in the read accesses since the last reset or clear. 0_B No single bit error was detected by the ECC in the read accesses. 1_B A single bit error was detected by the ECC in the read accesses.
ADBER	17	rh	**Accumulated Double Bit ECC Error** This register reports that at least one double bit error was found in the read accesses since the last reset or clear. 0_B No double bit error was detected by the ECC in the read accesses. 1_B A double bit error was detected by the ECC in the read accesses.
ATBER	18	rh	**Accumulated Triple Bit ECC Error** This register reports that at least one triple bit error was found in the read accesses since the last reset or clear. 0_B No triple bit error was detected by the ECC in the read accesses. 1_B A triple bit error was detected by the ECC in the read accesses.
AUCER	19	rh	**Accumulated Uncorrectable ECC Error** This register reports that at least one uncorrectable error was found in the read accesses since the last reset or clear. 0_B No uncorrectable error was detected by the ECC in the read accesses. 1_B A uncorrectable error was detected by the ECC in the read accesses.
AANER	24	rh	**Accumulated Any ECC error** This register reports that at least one error was found in the read accesses since the last reset or clear. 0_B No error was detected by the ECC in the read accesses. 1_B An error was detected by the ECC in the read accesses.
0	7:4, 15:9, 23:20, 31:25	r	**Reserved** Read as 0; shall be written with 0.

Table 514 Reset values of GP_CSRM_DFECCS

Reset type	Reset value	Note
Application Reset	0000 0000$_H$	
After Boot-FW Value	0000 0000$_H$	

5.4.1.68 CSRM DFLASH ECC control register

GP_CSRM_DFECCC Offset address: 0910$_H$

CSRM DFLASH ECC control register Application Reset value: C000 0000$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TRAPDIS		ECCCORDIS		0											
rw		rw		r											

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0														CLR	
r														w	

Field	Bits	Type	Description
CLR	1:0	w	**Clear ECC status bits** These registers clears the DMU_GP_CSRM_DFECCS registers. 00_B No action … 10_B No action 11_B Clears the DMU_GP_CSRM_DFECCS
ECCCORDIS	29:28	rw	**ECC correction disable** These registers disable the ECC correction for the CSRM DFLASH, UCB and CFS bank. 00_B ECC correction is enabled for CSRM DFLASH, UCB and CFS … 10_B ECC correction is enabled for CSRM DFLASH, UCB and CFS 11_B ECC correction is disabled for CSRM DFLASH, UCB and CFS
TRAPDIS	31:30	rw	**ECC trap disable** These registers disable the bus error trap on ECC uncorrectable errors for the CSRM DFLASH, UCB and CFS bank 00_B Bus error trap is enabled for CSRM DFLASH, UCB and CFS uncorrectable ECC errors … 10_B Bus error trap is enabled for CSRM DFLASH, UCB and CFS uncorrectable ECC errors 11_B Bus error trap is disabled for CSRM DFLASH, UCB and CFS uncorrectable ECC errors
0	27:2	r	**Reserved** Read as 0; shall be written with 0.

5.4.1.69 CSRM DFLASH ECC write register

These registers control the behavior of the ECC for write.

When disabling the ECC encoding with DMU_GP_CSRM_DFECCW.ENCDIS='11_B' the ECC code for the next 64-bit data block transferred from the DMU to the CSRM DFLASH is taken from the DMU_GP_CSRM_DFECCW.WCODE

GP_CSRM_DFECCW Offset address: 0914_H

CSRM DFLASH ECC write register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ENCDIS		0								WCODE					
rw		r								rw					

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WCODE															
rw															

Field	Bits	Type	Description
WCODE	21:0	rw	**Error correction code for write** These registers contain the ECC checksum for the next write on CSRM DFLASH.
ENCDIS	31:30	rw	**ECC encoding disable** 00_B The ECC code is calculated automatically. … 10_B The ECC code is calculated automatically. 11_B The ECC code is taken from WCODE
0	29:22	r	**Reserved** Read as 0; shall be written with 0.

5.4.1.70 CSRM DFLASH protection register

GP_CSRM_DFPROCON Offset address: 0918_H

CSRM DFLASH protection register Reset values see: Table 515

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															WP
r															rh

Field	Bits	Type	Description
WP	0	rh	**Write protection** This bit indicates whether the CSRM DFLASH is write protected. 0_B No write protection is configured for CSRM DFLASH. 1_B Write protection is configured for CSRM DFLASH.
0	31:1	r	**Reserved** Read as 0; shall be written with 0.

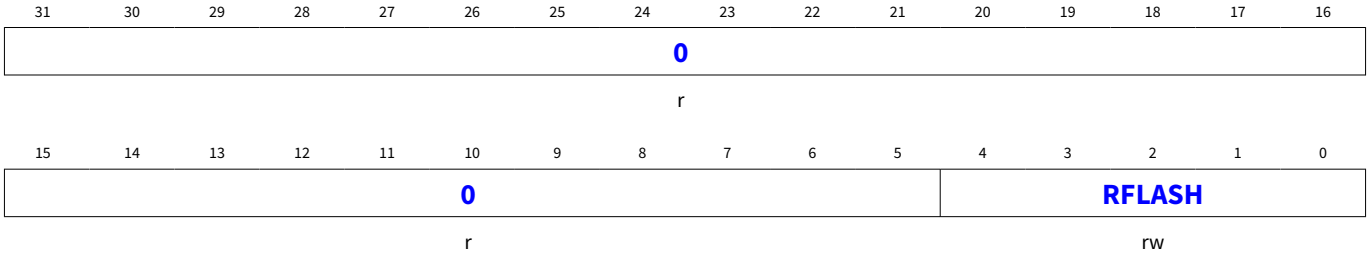
Table 515 Reset values of GP_CSRM_DFPROCON

Reset type	Reset value	Note
Application Reset	0000 0000$_H$	
After Boot-FW Value	0000 0000$_H$	

5.4.1.71 CSRM PFLASH wait cycle register

GP_CSRM_PFWAIT Offset address: 0920$_H$

CSRM PFLASH wait cycle register Reset values see: Table 516



Field	Bits	Type	Description
RFLASH	4:0	rw	**CSRM PFLASH wait cycles for read** These registers defines the number of SRI clock cycles needed for a CSRM PFLASH read access. 00_H Read access takes 2 fSRI clock cycles. 01_H Read access takes 2 fSRI clock cycles … $1F_H$ Read access takes 32 fSRI clock cycles
0	31:5	r	**Reserved** Read as 0; shall be written with 0.

Table 516 Reset values of GP_CSRM_PFWAIT

Reset type	Reset value	Note
System Reset	0000 000B$_H$	
After Boot-FW Value	0000 000B$_H$	

5.4.1.72 CSRM PFLASH error injection register

GP_CSRM_PFERRINJ Offset address: 0924_H

CSRM PFLASH error injection register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															WSERRINJ
r															w

Field	Bits	Type	Description
WSERRINJ	0	w	**CSRM PFLASH Wait State ECC error injection** Error injection into the ECC logic protecting PFlash Wait states to trigger an alarm. 0_B No action 1_B An error is injected in the ECC checksum protecting the PFLASH wait states
0	31:1	r	**Reserved** Read as 0; shall be written with 0.

5.4.1.73 CSRM PFLASH ECC write register

These registers control the behavior of the ECC for write.

When disabling the ECC correction with DMU_GP_CSRM_PFECCW.ENCDIS='11_B' the ECC code for the next 256-bit data block transferred from the DMU to the CSRM PFLASH is taken from the DMU_GP_CSRM_PFECCW.WCODE.

GP_CSRM_PFECCW Offset address: 0928_H

CSRM PFLASH ECC write register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ENCDIS		0								WCODE					
rw		r								rw					

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WCODE															
rw															

Field	Bits	Type	Description
WCODE	21:0	rw	**Error correction code for write** These registers contain the ECC checksum for the next write on the CSRM PFLASH.

(table continues...)

(continued)

Field	Bits	Type	Description
ENCDIS	31:30	rw	**ECC encoding disable** 00_B The ECC code is calculated automatically. … 10_B The ECC code is calculated automatically. 11_B The ECC code is taken from WCODE
0	29:22	r	**Reserved** Read as 0; shall be written with 0.

5.4.1.74 CSRM PFLASH bank write protection configuration 0

GP_CSRM_PFPROCONWP0 Offset address: $092C_H$

CSRM PFLASH bank write protection configuration 0 Reset values see: Table 517

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
S31WP	S30WP	S29WP	S28WP	S27WP	S26WP	S25WP	S24WP	S23WP	S22WP	S21WP	S20WP	S19WP	S18WP	S17WP	S16WP
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
S15WP	S14WP	S13WP	S12WP	S11WP	S10WP	S9WP	S8WP	S7WP	S6WP	S5WP	S4WP	S3WP	S2WP	S1WP	S0WP
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
SxWP (x=0-31)	x	rh	**PFLASH sector x write protection** These bits indicates whether PFLASH sector x is write protected 0_B No write protection is configured for sector x 1_B Write protection is configured for sector x

Table 517 Reset values of GP_CSRM_PFPROCONWP0

Reset type	Reset value	Note
Application Reset	$0000\ 0000_H$	
After Boot-FW Value	$----\ ----_H$	Values recovered from UCB

5.4.1.75 CSRM PFLASH bank write protection configuration 1

GP_CSRM_PFPROCONWP1 Offset address: 0930_H

CSRM PFLASH bank write protection configuration 1 Reset values see: Table 518

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
S63WP	S62WP	S61WP	S60WP	S59WP	S58WP	S57WP	S56WP	S55WP	S54WP	S53WP	S52WP	S51WP	S50WP	S49WP	S48WP
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
S47WP	S46WP	S45WP	S44WP	S43WP	S42WP	S41WP	S40WP	S39WP	S38WP	S37WP	S36WP	S35WP	S34WP	S33WP	S32WP
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
SxWP (x=32-63)	x-32	rh	**PFLASH sector x write protection** These bits indicates whether PFLASH sector x is write protected 0_B No write protection is configured for sector x 1_B Write protection is configured for sector x

Table 518 Reset values of GP_CSRM_PFPROCONWP1

Reset type	Reset value	Note
Application Reset	0000 0000$_H$	
After Boot-FW Value	---- ----$_H$	Values recovered from UCB

5.4.1.76 CSRM PFLASH bank write protection configuration 2

GP_CSRM_PFPROCONWP2 Offset address: 0934$_H$

CSRM PFLASH bank write protection configuration 2 Reset values see: Table 519

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
S95WP	S94WP	S93WP	S92WP	S91WP	S90WP	S89WP	S88WP	S87WP	S86WP	S85WP	S84WP	S83WP	S82WP	S81WP	S80WP
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
S79WP	S78WP	S77WP	S76WP	S75WP	S74WP	S73WP	S72WP	S71WP	S70WP	S69WP	S68WP	S67WP	S66WP	S65WP	S64WP
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
SxWP (x=64-95)	x-64	rh	**PFLASH Sector x write protection** These bits indicates whether PFLASH sector x is write protected 0_B No write protection is configured for sector x 1_B Write protection is configured for sector x

Table 519 Reset values of GP_CSRM_PFPROCONWP2

Reset type	Reset value	Note
Application Reset	0000 0000$_H$	
After Boot-FW Value	---- ----$_H$	Values recovered from UCB

5.4.1.77 CSRM PFLASH bank write protection configuration 3

GP_CSRM_PFPROCONWP3 Offset address: 0938$_H$

CSRM PFLASH bank write protection configuration 3 Reset values see: Table 520

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
S127WP	S126WP	S125WP	S124WP	S123WP	S122WP	S121WP	S120WP	S119WP	S118WP	S117WP	S116WP	S115WP	S114WP	S113WP	S112WP
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
S111WP	S110WP	S109WP	S108WP	S107WP	S106WP	S105WP	S104WP	S103WP	S102WP	S101WP	S100WP	S99WP	S98WP	S97WP	S96WP
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
SxWP (x=96-127)	x-96	rh	**PFLASH sector x write protection** These bits indicates whether PFLASH sector x is write protected 0_B No write protection is configured for sector x 1_B Write protection is configured for sector x

Table 520 Reset values of GP_CSRM_PFPROCONWP3

Reset type	Reset value	Note
Application Reset	0000 0000$_H$	
After Boot-FW Value	---- ----$_H$	Values recovered from UCB

5.4.1.78 CSRM PFLASH bank permanent write protection configuration 0

GP_CSRM_PFPROCONPWP0 Offset address: 093C$_H$

CSRM PFLASH bank permanent write protection configuration 0 Reset values see: Table 521

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
S31PWP	S30PWP	S29PWP	S28PWP	S27PWP	S26PWP	S25PWP	S24PWP	S23PWP	S22PWP	S21PWP	S20PWP	S19PWP	S18PWP	S17PWP	S16PWP
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
S15PWP	S14PWP	S13PWP	S12PWP	S11PWP	S10PWP	S9PWP	S8PWP	S7PWP	S6PWP	S5PWP	S4PWP	S3PWP	S2PWP	S1PWP	S0PWP
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
SxPWP (x=0-31)	x	rh	**PFLASH sector x permanent write protection** These bits indicates whether PFLASH sector x is permanently write protected 0_B No PWP protection is configured for sector x 1_B PWP protection is configured for sector x

Table 521 Reset values of GP_CSRM_PFPROCONPWP0

Reset type	Reset value	Note
Application Reset	0000 0000$_H$	
After Boot-FW Value	---- ----$_H$	Values recovered from UCB

5.4.1.79 CSRM PFLASH bank permanent write protection configuration 1

GP_CSRM_PFPROCONPWP1 Offset address: 0940$_H$

CSRM PFLASH bank permanent write protection configuration 1 Reset values see: Table 522

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
S63PWP	S62PWP	S61PWP	S60PWP	S59PWP	S58PWP	S57PWP	S56PWP	S55PWP	S54PWP	S53PWP	S52PWP	S51PWP	S50PWP	S49PWP	S48PWP
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
S47PWP	S46PWP	S45PWP	S44PWP	S43PWP	S42PWP	S41PWP	S40PWP	S39PWP	S38PWP	S37PWP	S36PWP	S35PWP	S34PWP	S33PWP	S32PWP
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
SxPWP (x=32-63)	x-32	rh	**PFLASH sector x permanent write protection** These bits indicates whether PFLASH sector x is permanently write protected 0_B No PWP protection is configured for sector x 1_B PWP protection is configured for sector x

Table 522 Reset values of GP_CSRM_PFPROCONPWP1

Reset type	Reset value	Note
Application Reset	0000 0000$_H$	
After Boot-FW Value	---- ----$_H$	Values recovered from UCB

5.4.1.80 CSRM PFLASH bank permanent write protection configuration 2

GP_CSRM_PFPROCONPWP2 Offset address: 0944$_H$

CSRM PFLASH bank permanent write protection configuration 2 Reset values see: Table 523

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
S95PWP	S94PWP	S93PWP	S92PWP	S91PWP	S90PWP	S89PWP	S88PWP	S87PWP	S86PWP	S85PWP	S84PWP	S83PWP	S82PWP	S81PWP	S80PWP
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
S79PWP	S78PWP	S77PWP	S76PWP	S75PWP	S74PWP	S73PWP	S72PWP	S71PWP	S70PWP	S69PWP	S68PWP	S67PWP	S66PWP	S65PWP	S64PWP
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
SxPWP (x=64-95)	x-64	rh	**PFLASH sector x permanent write protection** These bits indicates whether PFLASH sector x is permanently write protected 0_B No PWP protection is configured for sector x 1_B PWP protection is configured for sector x

Table 523 Reset values of GP_CSRM_PFPROCONPWP2

Reset type	Reset value	Note
Application Reset	0000 0000$_H$	
After Boot-FW Value	---- ----$_H$	Values recovered from UCB

5.4.1.81 CSRM PFLASH bank permanent write protection configuration 3

GP_CSRM_PFPROCONPWP3 Offset address: 0948$_H$

CSRM PFLASH bank permanent write protection configuration 3 Reset values see: Table 524

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
S127PWP	S126PWP	S125PWP	S124PWP	S123PWP	S122PWP	S121PWP	S120PWP	S119PWP	S118PWP	S117PWP	S116PWP	S115PWP	S114PWP	S113PWP	S112PWP
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
S111PWP	S110PWP	S109PWP	S108PWP	S107PWP	S106PWP	S105PWP	S104PWP	S103PWP	S102PWP	S101PWP	S100PWP	S99PWP	S98PWP	S97PWP	S96PWP
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
SxPWP (x=96-127)	x-96	rh	**PFLASH sector x permanent write protection** These bits indicates whether PFLASH sector x is permanently write protected 0_B No PWP protection is configured for sector x 1_B PWP protection is configured for sector x

Table 524 Reset values of GP_CSRM_PFPROCONPWP3

Reset type	Reset value	Note
Application Reset	0000 0000_H	
After Boot-FW Value	---- $----_H$	Values recovered from UCB

5.4.1.82 CSRM PFLASH bank permanent write once protection configuration 0

GP_CSRM_PFPROCONPWOP0 Offset address: $094C_H$

CSRM PFLASH bank permanent write once protection configuration 0 Reset values see: Table 525

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
S31PWOP	S30PWOP	S29PWOP	S28PWOP	S27PWOP	S26PWOP	S25PWOP	S24PWOP	S23PWOP	S22PWOP	S21PWOP	S20PWOP	S19PWOP	S18PWOP	S17PWOP	S16PWOP
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
S15PWOP	S14PWOP	S13PWOP	S12PWOP	S11PWOP	S10PWOP	S9PWOP	S8PWOP	S7PWOP	S6PWOP	S5PWOP	S4PWOP	S3PWOP	S2PWOP	S1PWOP	S0PWOP
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
SxPWOP (x=0-31)	x	rh	**PFLASH sector x permanent write once protection** These bits indicates whether PFLASH sector x is write once protected 0_B No permanent write once protection is configured for sector x 1_B Permanent write once protection is configured for sector x

Table 525 Reset values of GP_CSRM_PFPROCONPWOP0

Reset type	Reset value	Note
Application Reset	0000 0000$_H$	
After Boot-FW Value	---- ----$_H$	Values recovered from UCB

5.4.1.83 CSRM PFLASH bank permanent write once protection configuration 1

GP_CSRM_PFPROCONPWOP1 Offset address: 0950$_H$

CSRM PFLASH bank permanent write once protection configuration 1 Reset values see: Table 526

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
S63PWOP	S62PWOP	S61PWOP	S60PWOP	S59PWOP	S58PWOP	S57PWOP	S56PWOP	S55PWOP	S54PWOP	S53PWOP	S52PWOP	S51PWOP	S50PWOP	S49PWOP	S48PWOP
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
S47PWOP	S46PWOP	S45PWOP	S44PWOP	S43PWOP	S42PWOP	S41PWOP	S40PWOP	S39PWOP	S38PWOP	S37PWOP	S36PWOP	S35PWOP	S34PWOP	S33PWOP	S32PWOP
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
SxPWOP (x=32-63)	x-32	rh	**PFLASH sector x permanent write once protection** These bits indicates whether PFLASH sector x is write once protected 0_B No permanent write once protection is configured for sector x 1_B Permanent write once protection is configured for sector x

Table 526 Reset values of GP_CSRM_PFPROCONPWOP1

Reset type	Reset value	Note
Application Reset	0000 0000$_H$	
After Boot-FW Value	---- ----$_H$	Values recovered from UCB

5.4.1.84 CSRM PFLASH bank permanent write once protection configuration 2

GP_CSRM_PFPROCONPWOP2 Offset address: 0954$_H$

CSRM PFLASH bank permanent write once protection configuration 2 Reset values see: Table 527

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
S95PWOP	S94PWOP	S93PWOP	S92PWOP	S91PWOP	S90PWOP	S89PWOP	S88PWOP	S87PWOP	S86PWOP	S85PWOP	S84PWOP	S83PWOP	S82PWOP	S81PWOP	S80PWOP
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
S79PWOP	S78PWOP	S77PWOP	S76PWOP	S75PWOP	S74PWOP	S73PWOP	S72PWOP	S71PWOP	S70PWOP	S69PWOP	S68PWOP	S67PWOP	S66PWOP	S65PWOP	S64PWOP
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
SxPWOP (x=64-95)	x-64	rh	**PFLASH sector x permanent write once protection** These bits indicates whether PFLASH sector x is write once protected 0_B No permanent write once protection is configured for sector x 1_B Permanent write once protection is configured for sector x

Table 527 Reset values of GP_CSRM_PFPROCONPWOP2

Reset type	Reset value	Note
Application Reset	0000 0000$_H$	
After Boot-FW Value	---- ----$_H$	Values recovered from UCB

5.4.1.85 CSRM PFLASH bank permanent write once protection configuration 3

GP_CSRM_PFPROCONPWOP3 Offset address: 0958$_H$

CSRM PFLASH bank permanent write once protection configuration 3 Reset values see: Table 528

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
S127PWOP	S126PWOP	S125PWOP	S124PWOP	S123PWOP	S122PWOP	S121PWOP	S120PWOP	S119PWOP	S118PWOP	S117PWOP	S116PWOP	S115PWOP	S114PWOP	S113PWOP	S112PWOP
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
S111PWOP	S110PWOP	S109PWOP	S108PWOP	S107PWOP	S106PWOP	S105PWOP	S104PWOP	S103PWOP	S102PWOP	S101PWOP	S100PWOP	S99PWOP	S98PWOP	S97PWOP	S96PWOP
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
SxPWOP (x=96-127)	x-96	rh	**PFLASH sector x permanent write once protection** These bits indicates whether PFLASH sector x is write once protected 0_B No permanent write once protection is configured for sector x 1_B Permanent write once protection is configured for sector x

Table 528 Reset values of GP_CSRM_PFPROCONPWOP3

Reset type	Reset value	Note
Application Reset	0000 0000$_H$	
After Boot-FW Value	---- ----$_H$	Values recovered from UCB

5.4.1.86 CSRM PFLASH bank erase counter priority configuration 0

GP_CSRM_PFECPRIO0 Offset address: 095C$_H$

CSRM PFLASH bank erase counter priority configuration 0 Reset values see: Table 529

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
S31LH	**S30LH**	**S29LH**	**S28LH**	**S27LH**	**S26LH**	**S25LH**	**S24LH**	**S23LH**	**S22LH**	**S21LH**	**S20LH**	**S19LH**	**S18LH**	**S17LH**	**S16LH**
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
S15LH	**S14LH**	**S13LH**	**S12LH**	**S11LH**	**S10LH**	**S9LH**	**S8LH**	**S7LH**	**S6LH**	**S5LH**	**S4LH**	**S3LH**	**S2LH**	**S1LH**	**S0LH**
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
SxLH (x=0-31)	x	rh	**PFLASH sector x erase counter priority** These bits indicates whether PFLASH sector x has high or low priority for erase counter recording 0_B Low priority is configured for sector x 1_B High priority is configured for sector x

Table 529 Reset values of GP_CSRM_PFECPRIO0

Reset type	Reset value	Note
Application Reset	0000 0000$_H$	
After Boot-FW Value	---- ----$_H$	Values recovered from UCB

5.4.1.87 CSRM PFLASH bank erase counter priority configuration 1

GP_CSRM_PFECPRIO1 Offset address: 0960$_H$

CSRM PFLASH bank erase counter priority configuration 1 Reset values see: Table 530

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
S63LH	S62LH	S61LH	S60LH	S59LH	S58LH	S57LH	S56LH	S55LH	S54LH	S53LH	S52LH	S51LH	S50LH	S49LH	S48LH
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
S47LH	S46LH	S45LH	S44LH	S43LH	S42LH	S41LH	S40LH	S39LH	S38LH	S37LH	S36LH	S35LH	S34LH	S33LH	S32LH
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
SxLH (x=32-63)	x-32	rh	**PFLASH sector x erase counter priority** These bits indicates whether PFLASH sector x has high or low priority for erase counter recording 0_B Low priority is configured for sector x 1_B High priority is configured for sector x

Table 530 Reset values of GP_CSRM_PFECPRIO1

Reset type	Reset value	Note
Application Reset	$0000\ 0000_H$	
After Boot-FW Value	$----\ ----_H$	Values recovered from UCB

5.4.1.88 CSRM PFLASH bank erase counter priority configuration 2

GP_CSRM_PFECPRIO2

CSRM PFLASH bank erase counter priority configuration 2

Offset address: 0964_H

Reset values see: Table 531

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
S95LH	S94LH	S93LH	S92LH	S91LH	S90LH	S89LH	S88LH	S87LH	S86LH	S85LH	S84LH	S83LH	S82LH	S81LH	S80LH
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
S79LH	S78LH	S77LH	S76LH	S75LH	S74LH	S73LH	S72LH	S71LH	S70LH	S69LH	S68LH	S67LH	S66LH	S65LH	S64LH
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
SxLH (x=64-95)	x-64	rh	**PFLASH sector x erase counter priority** These bits indicates whether PFLASH sector x has high or low priority for erase counter recording 0_B Low priority is configured for sector x 1_B High priority is configured for sector x

Table 531 Reset values of GP_CSRM_PFECPRIO2

Reset type	Reset value	Note
Application Reset	0000 0000$_H$	
After Boot-FW Value	---- ----$_H$	Values recovered from UCB

5.4.1.89 CSRM PFLASH bank erase counter priority configuration 3

GP_CSRM_PFECPRIO3 Offset address: 0968$_H$

CSRM PFLASH bank erase counter priority configuration 3 Reset values see: Table 532

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
S127LH	S126LH	S125LH	S124LH	S123LH	S122LH	S121LH	S120LH	S119LH	S118LH	S117LH	S116LH	S115LH	S114LH	S113LH	S112LH
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
S111LH	S110LH	S109LH	S108LH	S107LH	S106LH	S105LH	S104LH	S103LH	S102LH	S101LH	S100LH	S99LH	S98LH	S97LH	S96LH
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
SxLH (x=96-127)	x-96	rh	**PFLASH sector x erase counter priority** These bits indicates whether PFLASH sector x has high or low priority for erase counter recording 0_B Low priority is configured for sector x 1_B High priority is configured for sector x

Table 532 Reset values of GP_CSRM_PFECPRIO3

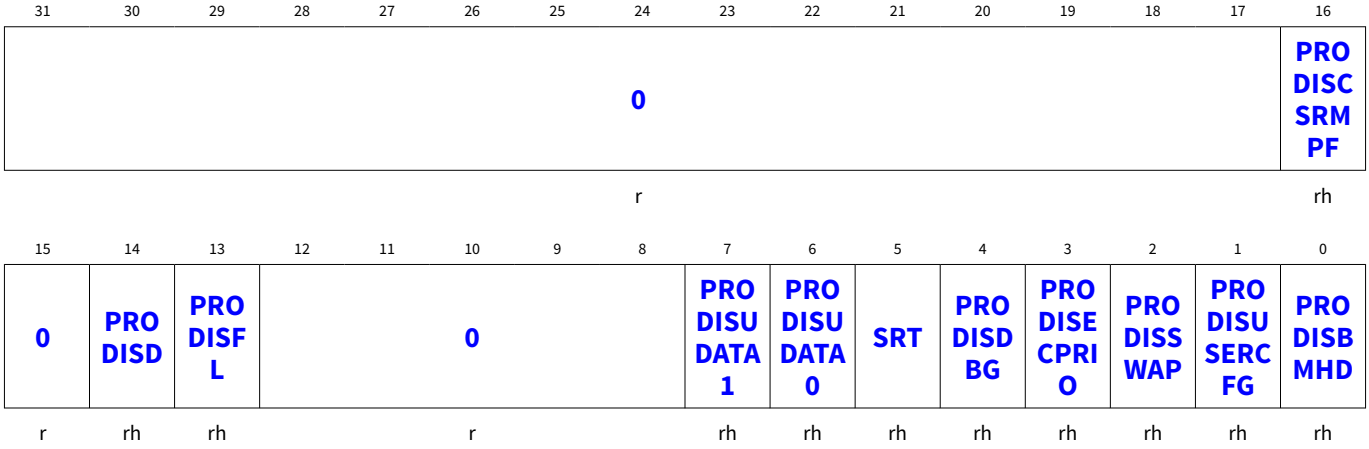
Reset type	Reset value	Note
Application Reset	0000 0000$_H$	
After Boot-FW Value	---- ----$_H$	Values recovered from UCB

5.4.1.90 CSRM UCB password protection register

This register reports the state of the CSRM FLASH protection and contains protection relevant control fields.

GP_CSRM_PROTECT Offset address: 096C$_H$

CSRM UCB password protection register Application Reset value: 0000 0000$_H$



Field	Bits	Type	Description
PRODISBMHD	0	rh	**UCB_CS_BMHD protection disabled** The protection configured by UCB_CS_BMHD0, UCB_CS_BMHD1, UCB_CS_BMHD2 and UCB_CS_BMHD3 was successfully disabled by supplying the correct password to "Disable Protection". Note: Cleared with command "Resume Protection".
PRODISUSERCFG	1	rh	**UCB_CSRM protection disabled** The protection configured by UCB_CS_USERCFG_ORIG and UCB_CS_USERCFG_COPY was successfully disabled by supplying the correct password to "Disable Protection". Note: Cleared with command "Resume Protection".
PRODISSWAP	2	rh	**UCB_CS_SWAP protection disabled** The protection configured by UCB_CS_SWAP_ORIG and UCB_CS_SWAP_COPY was successfully disabled by supplying the correct password to "Disable Protection". Note: Cleared with command "Resume Protection".
PRODISECPRIO	3	rh	**UCB_CS_ECPRIO protection disabled** The protection configured by UCB_CS_ECPRIO_ORIG and UCB_CS_ECPRIO_COPY was successfully disabled by supplying the correct password to "Disable Protection". Note: Cleared with command "Resume Protection".
PRODISDBG	4	rh	**UCB_CS_DBGCS protection disabled** The protection configured by UCB_DBGCS_ORIG and UCB_DBGCS_COPY was successfully disabled by supplying the correct password to "Disable Protection". Note: Cleared with command "Resume Protection".
SRT	5	rh	**Secure retest password protection disabled** The protection configured by UCB_RETEST was successfully disabled by supplying the correct password to "Disable Protection". Note: Cleared with command "Resume Protection".

(table continues...)

(continued)

Field	Bits	Type	Description
PRODISUDATA0	6	rh	**UCB_CS_USER_DATA0 protection disabled** The protection configured by UCB_CS_USER_DATA0 was successfully disabled by supplying the correct password to "Disable Protection". Note: Cleared with command "Resume Protection".
PRODISUDATA1	7	rh	**UCB_CS_USER_DATA1 protection disabled** The protection configured by UCB_CS_USER_DATA1 was successfully disabled by supplying the correct password to "Disable Protection". Note: Cleared with command "Resume Protection".
PRODISFL	13	rh	**HOST FLASH Protection Disabled** The read and write protections for CSRM PFLASH and CSRM DFLASH configured by UCB_CS_FLASH_ORIG and UCB_CS_FLASH_COPY were successfully disabled by supplying the correct password to "Disable Protection". Note: Cleared with command "Resume Protection".
PRODISD	14	rh	**CSRM DFLASH Protection Disabled** The read and write protections for CSRM DFLASH configured by UCB_CS_FLASH_ORIG and UCB_CS_FLASH_COPY were successfully disabled by supplying the correct password to "Disable Protection". Note: Cleared with command "Resume Protection".
PRODISCSRMPF	16	rh	**CSRM PFLASH protection disable** The protection configured for CSRM PFLASH by UCB_CS_FLASH_ORIG and UCB_CS_FLASH_COPY was successfully disabled by supplying the correct password to "Disable Protection". Note: Cleared with command "Resume Protection".
0	12:8, 15, 31:17	r	**Reserved** Read as 0; shall be written with 0.

5.4.1.91 CSRM UCB confirmation codes register A

GP_CSRM_CONFIRMA Offset address: 0970_H

CSRM UCB confirmation codes register A Reset values see: Table 533

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								PROINBMHD3		PROINBMHD2		PROINBMHD1		PROINBMHD0	
r								rh		rh		rh		rh	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PROINRETEST		0						PROINUDATA1		PROINUDATA0		PROINCFG		PROINMOCFG	
rh		r						rh		rh		rh		rh	

Field	Bits	Type	Description
PROINMOCFG	1:0	rh	**UCB_CS_MOCFG confirmation state** This bit reflects the confirmation codes of UCB_CS_MOCFG. 00_B UCB was not read yet. 01_B UCB state is UNLOCKED. 10_B UCB state is CONFIRMED. 11_B UCB state is ERRORED.
PROINCFG	3:2	rh	**UCB_CS_CFG confirmation state** This bit reflects the confirmation codes of UCB_CS_CFG. 00_B UCB was not read yet. 01_B UCB state is UNLOCKED. 10_B UCB state is CONFIRMED. 11_B UCB state is ERRORED.
PROINUDATA0	5:4	rh	**UCB_CS_USER_DATA0 confirmation state** This bit reflects the confirmation codes of UCB_CS_USER_DATA0. Note: This is a single-UCB. If in ERRORED state, the UCB cannot be recovered. 00_B UCB was not read yet. 01_B UCB state is UNLOCKED. 10_B UCB state is CONFIRMED. 11_B UCB state is ERRORED.
PROINUDATA1	7:6	rh	**UCB_CS_USER_DATA1 confirmation state** This bit reflects the confirmation codes of UCB_CS_USER_DATA1. Note: This is a single-UCB. If in ERRORED state, the UCB cannot be recovered. 00_B UCB was not read yet. 01_B UCB state is UNLOCKED. 10_B UCB state is CONFIRMED. 11_B UCB state is ERRORED.
PROINRETEST	15:14	rh	**UCB_RETEST confirmation state** This bit reflects the confirmation codes of UCB_RETEST. 00_B UCB was not read yet. 01_B UCB state is UNLOCKED. 10_B UCB state is CONFIRMED. 11_B UCB state is ERRORED.
PROINBMHD0	17:16	rh	**UCB_CS_BMHD0 confirmation state** This bit reflects the confirmation codes of UCB_CS_BMHD0. Note: if the confirmation code of BMHD0 is ERRORED, all BMHDs cannot be unlocked even BMHD1, BMHD2 and BMHD3 are in CONFIRMED state 00_B UCB was not read yet. 01_B UCB state is UNLOCKED. 10_B UCB state is CONFIRMED. 11_B UCB state is ERRORED.

(table continues...)

(continued)

Field	Bits	Type	Description
PROINBMHD1	19:18	rh	**UCB_CS_BMHD1 confirmation state** This bit reflects the confirmation codes of UCB_CS_BMHD1. Note: if the confirmation code of BMHD0 is ERRORED, this BMHD cannot be unlocked even if it is in CONFIRMED state 00_B UCB was not read yet. 01_B UCB state is UNLOCKED. 10_B UCB state is CONFIRMED. 11_B UCB state is ERRORED.
PROINBMHD2	21:20	rh	**UCB_CS_BMHD2 confirmation state** This bit reflects the confirmation codes of UCB_CS_BMHD2. Note: if the confirmation code of BMHD0 is ERRORED, this BMHD cannot be unlocked even if it is in CONFIRMED state 00_B UCB was not read yet. 01_B UCB state is UNLOCKED. 10_B UCB state is CONFIRMED. 11_B UCB state is ERRORED.
PROINBMHD3	23:22	rh	**UCB_CS_BMHD3 confirmation state** This bit reflects the confirmation codes of UCB_CS_BMHD3. Note: if the confirmation code of BMHD0 is ERRORED, this BMHD cannot be unlocked even if it is in CONFIRMED state 00_B UCB was not read yet. 01_B UCB state is UNLOCKED. 10_B UCB state is CONFIRMED. 11_B UCB state is ERRORED.
0	13:8, 31:24	r	**Reserved** Read as 0; shall be written with 0.

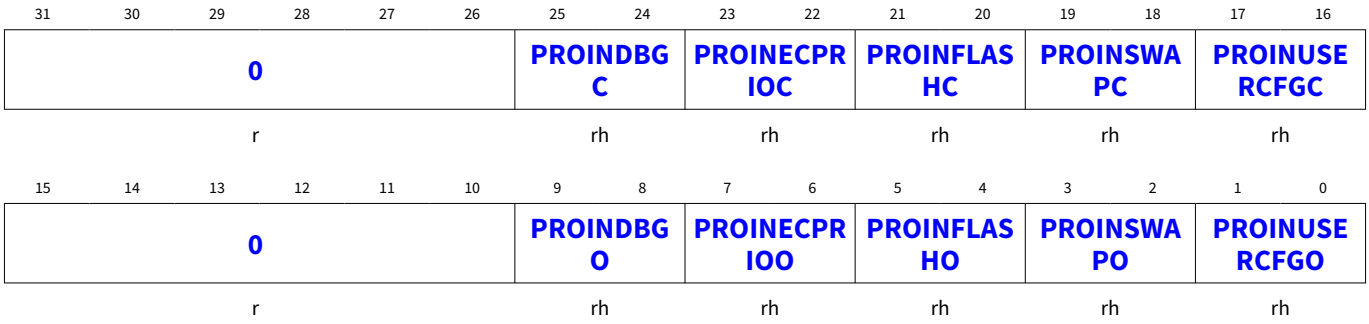
Table 533 Reset values of GP_CSRM_CONFIRMA

Reset type	Reset value	Note
Application Reset	$0000\ 0000_H$	
After Boot-FW Value	XXXX XXXX ---- ---- --XX XXXX ---- $----_B$	

5.4.1.92 CSRM UCB confirmation codes register B

GP_CSRM_CONFIRMB Offset address: 0974_H

CSRM UCB confirmation codes register B Reset values see: Table 534



Field	Bits	Type	Description
PROINUSERCFGO	1:0	rh	**UCB_CS_USERCFG_ORIG confirmation state** This bit reflects the confirmation codes of UCB_CS_USERCFG_ORIG. 00_B ORIG UCB was not read yet. 01_B ORIG UCB state is UNLOCKED. 10_B ORIG UCB state is CONFIRMED. 11_B ORIG UCB state is ERRORED.
PROINSWAPO	3:2	rh	**UCB_CS_SWAP_ORIG confirmation state** This bit reflects the confirmation codes of UCB_CS_SWAP_ORIG. 00_B ORIG UCB was not read yet. 01_B ORIG UCB state is UNLOCKED. 10_B ORIG UCB state is CONFIRMED. 11_B ORIG UCB state is ERRORED.
PROINFLASHO	5:4	rh	**UCB_CS_FLASH_ORIG confirmation state** This bit reflects the confirmation codes of UCB_CS_FLASH_ORIG. 00_B ORIG UCB was not read yet. 01_B ORIG UCB state is UNLOCKED. 10_B ORIG UCB state is CONFIRMED. 11_B ORIG UCB state is ERRORED.
PROINECPRIOO	7:6	rh	**UCB_CS_ECPRIO_ORIG confirmation state** This bit reflects the confirmation codes of UCB_CS_ECPRIO_ORIG. 00_B ORIG UCB was not read yet. 01_B ORIG UCB state is UNLOCKED. 10_B ORIG UCB state is CONFIRMED. 11_B ORIG UCB state is ERRORED.
PROINDBGO	9:8	rh	**UCB_DBGCS_ORIG confirmation state** This bit reflects the confirmation codes of UCB_DBGCS_ORIG. 00_B ORIG UCB was not read yet. 01_B ORIG UCB state is UNLOCKED. 10_B ORIG UCB state is CONFIRMED. 11_B ORIG UCB state is ERRORED.

(table continues...)

(continued)

Field	Bits	Type	Description
PROINUSERCFGC	17:16	rh	**UCB_CS_USERCFG_COPY confirmation state** This bit reflects the confirmation codes of UCB_CS_USERCFG_COPY. 00_B COPY UCB was not read yet. 01_B COPY UCB state is UNLOCKED. 10_B COPY UCB state is CONFIRMED. 11_B COPY UCB state is ERRORED.
PROINSWAPC	19:18	rh	**UCB_CS_SWAP_COPY confirmation state** This bit reflects the confirmation codes of UCB_CS_SWAP_COPY. 00_B COPY UCB was not read yet. 01_B COPY UCB state is UNLOCKED. 10_B COPY UCB state is CONFIRMED. 11_B COPY UCB state is ERRORED.
PROINFLASHC	21:20	rh	**UCB_CS_FLASH_COPY confirmation state** This bit reflects the confirmation codes of UCB_CS_FLASH_COPY. 00_B COPY UCB was not read yet. 01_B COPY UCB state is UNLOCKED. 10_B COPY UCB state is CONFIRMED. 11_B COPY UCB state is ERRORED.
PROINECPRIOC	23:22	rh	**UCB_CS_ECPRIO_COPY confirmation state** This bit reflects the confirmation codes of UCB_CS_ECPRIO_COPY. 00_B COPY UCB was not read yet. 01_B COPY UCB state is UNLOCKED. 10_B COPY UCB state is CONFIRMED. 11_B COPY UCB state is ERRORED.
PROINDBGC	25:24	rh	**UCB_DBGCS_COPY confirmation state** This bit reflects the confirmation codes of UCB_DBGCS_COPY. 00_B COPY UCB was not read yet. 01_B COPY UCB state is UNLOCKED. 10_B COPY UCB state is CONFIRMED. 11_B COPY UCB state is ERRORED.
0	15:10, 31:26	r	**Reserved** Read as 0; shall be written with 0.

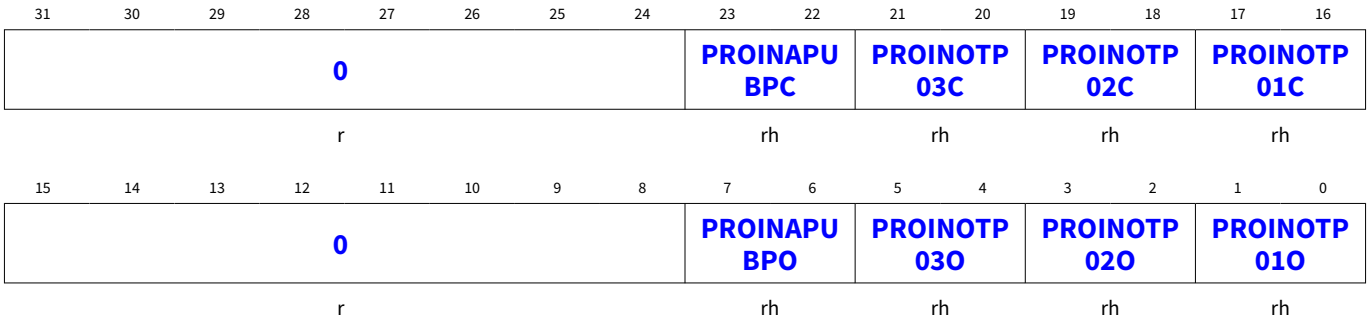
Table 534 Reset values of GP_CSRM_CONFIRMB

Reset type	Reset value	Note
Application Reset	$0000\ 0000_H$	
After Boot-FW Value	XXXX XX-- ---- ---- $XXXX\ XX--\ ----\ ----_B$	

5.4.1.93 CSRM UCB confirmation codes register C

GP_CSRM_CONFIRMC Offset address: 0978_H

CSRM UCB confirmation codes register C Reset values see: Table 535



Field	Bits	Type	Description
PROINOTP01O	1:0	rh	**UCB_CS_PFLASH_OTP0_ORIG OTP set 1 confirmation state** This bit reflects the confirmation codes of UCB_CS_PFLASH_OTP0_ORIG for OTP set 1. 00_B ORIG OTP set was not read yet. 01_B ORIG OTP set state is UNLOCKED. 10_B ORIG OTP set state is CONFIRMED. 11_B ORIG OTP set state is ERRORED.
PROINOTP02O	3:2	rh	**UCB_CS_PFLASH_OTP0_ORIG OTP set 2 confirmation state** This bit reflects the confirmation codes of UCB_CS_PFLASH_OTP0_ORIG for OTP set 2. 00_B ORIG OTP set was not read yet. 01_B ORIG OTP set state is UNLOCKED. 10_B ORIG OTP set state is CONFIRMED. 11_B ORIG OTP set state is ERRORED.
PROINOTP03O	5:4	rh	**UCB_CS_PFLASH_OTP0_ORIG OTP set 3 confirmation state** This bit reflects the confirmation codes of UCB_CS_PFLASH_OTP0_ORIG for OTP set 3. 00_B ORIG OTP set was not read yet. 01_B ORIG OTP set state is UNLOCKED. 10_B ORIG OTP set state is CONFIRMED. 11_B ORIG OTP set state is ERRORED.
PROINAPUBPO	7:6	rh	**UCB_CS_PFLASH_OTP0_ORIG APU bypass confirmation state** This bit reflects the confirmation codes of UCB_CS_PFLASH_OTP0_ORIG for APU bypass. 00_B ORIG OTP set was not read yet. 01_B ORIG OTP set state is UNLOCKED. 10_B ORIG OTP set state is CONFIRMED. 11_B ORIG OTP set state is ERRORED.
PROINOTP01C	17:16	rh	**UCB_CS_PFLASH_OTP0_COPY OTP set 1 confirmation state** This bit reflects the confirmation codes of UCB_CS_PFLASH_OTP0_COPY for OTP set 1. 00_B COPY OTP set was not read yet. 01_B COPY OTP set state is UNLOCKED. 10_B COPY OTP set state is CONFIRMED. 11_B COPY OTP set state is ERRORED.

(table continues...)

(continued)

Field	Bits	Type	Description
PROINOTP02C	19:18	rh	**UCB_CS_PFLASH_OTP0_COPY OTP set 2 confirmation state** This bit reflects the confirmation codes of UCB_CS_PFLASH_OTP0_COPY for OTP set 2. 00_B COPY OTP set was not read yet. 01_B COPY OTP set state is UNLOCKED. 10_B COPY OTP set state is CONFIRMED. 11_B COPY OTP set state is ERRORED.
PROINOTP03C	21:20	rh	**UCB_CS_PFLASH_OTP0_COPY OTP set 3 confirmation state** This bit reflects the confirmation codes of UCB_CS_PFLASH_OTP0_COPY for OTP set 3. 00_B COPY OTP set was not read yet. 01_B COPY OTP set state is UNLOCKED. 10_B COPY OTP set state is CONFIRMED. 11_B COPY OTP set state is ERRORED.
PROINAPUBPC	23:22	rh	**UCB_CS_PFLASH_OTP0_COPY APU bypass confirmation state** This bit reflects the confirmation codes of UCB_CS_PFLASH_OTP0_COPY for APU bypass. 00_B COPY OTP set was not read yet. 01_B COPY OTP set state is UNLOCKED. 10_B COPY OTP set state is CONFIRMED. 11_B COPY OTP set state is ERRORED.
0	15:8, 31:24	r	**Reserved** Read as 0; shall be written with 0.

Table 535 Reset values of GP_CSRM_CONFIRMC

Reset type	Reset value	Note
Application Reset	$0000\ 0000_H$	
After Boot-FW Value	$XX\text{--}\ XX\text{--}_H$	

5.4.1.94 CSRM debug configuration

GP_CSRM_DBGCFG Offset address: $097C_H$

CSRM debug configuration Reset values see: Table 536

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0														DESTDBG	
r														rh	

Field	Bits	Type	Description
DESTDBG	1:0	rh	**Destructive Debug Entry** This field configures the destructive debug entry. 00_B Debug entry is non-destructive. … 10_B Debug entry is non-destructive. 11_B Debug entry is destructive.
0	31:2	r	**Reserved** Read as 0; shall be written with 0.

Table 536 Reset values of GP_CSRM_DBGCFG

Reset type	Reset value	Note
Application Reset	$0000\ 0000_H$	
After Boot-FW Value	$0000\ 0000_H$	

5.4.1.95 CSRM debug mode

This register configures the destructive debug mode.

GP_CSRM_DBGMODE Offset address: 0980_H

CSRM debug mode Reset values see: Table 537

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0														EDDM	
r														rh	

Field	Bits	Type	Description
EDDM	1:0	rh	**Entered Destructive Debug Mode** This bit indicates whether the debug interface has been opened via Destructive Debug Entry. Consequently the CAN and FlexRay operation is made impossible 00_B Destructive debug not entered, device operation not affected. … 10_B Destructive debug not entered, device operation not affected. 11_B Destructive debug entered, CAN and FlexRay operation affected.
0	31:2	r	**Reserved** Read as 0; shall be written with 0.

Table 537 Reset values of GP_CSRM_DBGMODE

Reset type	Reset value	Note
Application Reset	$0000\ 0000_H$	
After Boot-FW Value	0000 0000 0000 0000 0000 0000 0000 00––$_B$	Value recovered from UCB

5.4.1.96 Bank allocation register

This register shows the bank allocation chosen by mode switch

GP_BKALLOC Offset address: $0A00_H$

Bank allocation register Application Reset value: $0006\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0													CSRMPF	CSRMDF	HOSTDF
r													rh	rh	rh

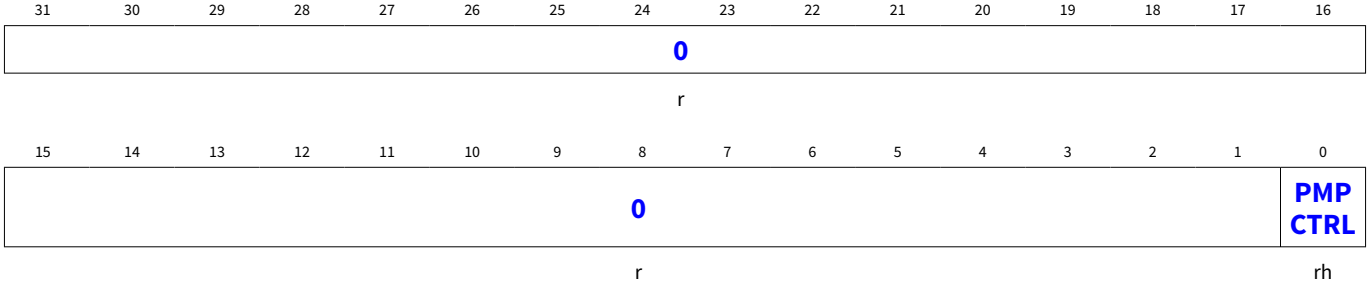
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		0		P5B	P5A	P4B	P4A	P3B	P3A	P2B	P2A	P1B	P1A	P0B	P0A
r		r		rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
PxA (x=0-5)	2*x	rh	**HOST PFLASH PxA bank allocation setting** Setting for PxA bank 0_B Bank is allocated to the HOST command interface and HOST FSI 1_B Bank is allocated to the CSRM command interface and CSRM FSI
PxB (x=0-5)	2*x+1	rh	**HOST PFLASH PxB bank allocation setting** Setting for PxB bank 0_B Bank is allocated to the HOST command interface and HOST FSI 1_B Bank is allocated to the CSRM command interface and CSRM FSI
HOSTDF	16	rh	**HOST DFLASH bank allocation setting** 0_B Bank is allocated to the HOST command interface and HOST FSI 1_B Bank is allocated to the CSRM command interface and CSRM FSI
CSRMDF	17	rh	**CSRM DFLASH bank allocation setting** 0_B Bank is allocated to the HOST command interface and HOST FSI 1_B Bank is allocated to the CSRM command interface and CSRM FSI
CSRMPF	18	rh	**CSRM PFLASH bank allocation setting** 0_B Bank is allocated to the HOST command interface and HOST FSI 1_B Bank is allocated to the CSRM command interface and CSRM FSI
0	15:14, 13:12, 31:19	r	**Reserved** Read as 0; shall be written with 0.

5.4.1.97 Power control register

GP_PCONTROL Offset address: $0A14_H$

Power control register Reset values see: Table 538



Field	Bits	Type	Description
PMPCTRL	0	rh	**Pump mode control** This bitfield control the overall configuration for the pump mode for the operations. 0_B Manual mode 1_B Automatic mode
0	31:1	r	**Reserved** Read as 0; shall be written with 0.

Table 538 Reset values of GP_PCONTROL

Reset type	Reset value	Note
Application Reset	$0000\ 0000_H$	
After Boot-FW Value	0000 0000 0000 0000 0000 0000 0000 $000-_B$	Value recovered from UCB

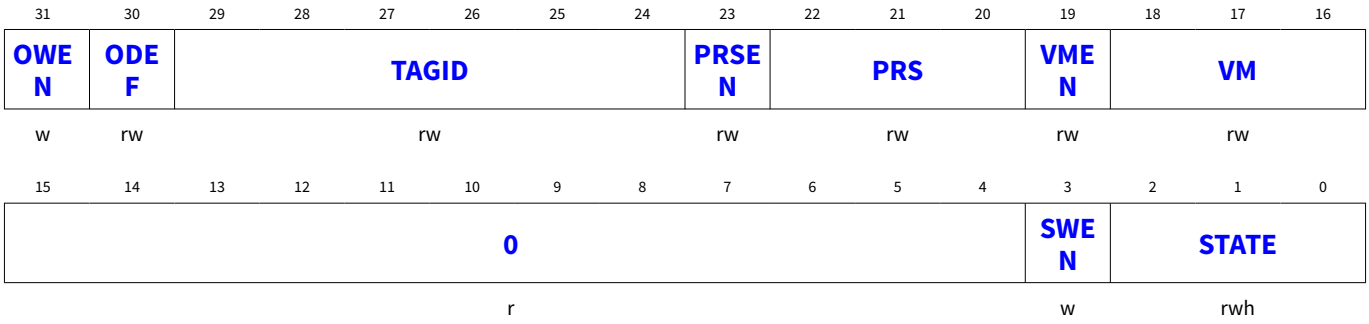
5.4.1.98 PROT register for HOST command interface and HCI registers

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner or CS master to update the PROT state as described in the PROT mechanism.

The PROTHCI register controls lock and unlock of the locally protected HOST command interface and HCI registers.

PROT_PROTHCI Offset address: $0B00_H$

PROT register for HOST command interface and HCI registers Application Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
VM	18:16	rw	**Virtual Machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual Machine definition Enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection Set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.
PRSEN	23	rw	**Protection Set definition Enable for PROT owner** 0_B **Disable**: PRS is not part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; shall be written with 0.

Table 539 Access mode restrictions of PROT_PROTHCI sorted by descending priority

Mode name	Access mode		Description
PROT_OWNER and write 1 to .**SWEN**	rwh	STATE	PROT owner can modify the PROT state (for allowed and valid transitions)
PROT_OWNER and write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	PROT owner can modify the existing PROT owner
SECURE_MASTER and write 1 to .**SWEN**	rwh	STATE	Secure master can modify the PROT state (for allowed and valid transitions)
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

5.4.1.99 HOST command interface and HCI registers, write access enable register A

This register defines which master TAG ID encodings are enabled or disabled for write accesses to the access protected region.

PROT_ACCENHCI_WRA

HOST command interface and HCI registers, write access enable register A

Offset address: $0B08_H$

Application Reset value: $1000\ 0003_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Write access enable for Master TAG ID q** This bit enables write access to the access protected region for transactions with the Master TAG ID q. 0_B Disabled for write access 1_B Enabled for write access

5.4.1.100 HOST command interface and HCI registers, write access enable register B

This register defines which master TAG ID encodings are enabled or disabled for write accesses to the access protected region.

PROT_ACCENHCI_WRB

HOST command interface and HCI registers, write access enable register B

Offset address: $0B0C_H$

Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN63	EN62	EN61	EN60	EN59	EN58	EN57	EN56	EN55	EN54	EN53	EN52	EN51	EN50	EN49	EN48
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN47	EN46	EN45	EN44	EN43	EN42	EN41	EN40	EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-63)	q-32	rw	**Write access enable for Master TAG ID q** This bit enables write access to the access protected region for transactions with the Master TAG ID q. 0_B Disabled for write access 1_B Enabled for write access

5.4.1.101 HOST command interface and HCI registers, read access enable register A

This register defines which master TAG ID encodings are enabled or disabled for read accesses to the access protected region.

PROT_ACCENHCI_RDA

Offset address: $0B10_H$

HOST command interface and HCI registers, read access enable register A

Application Reset value: $FFFF\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Read access enable for Master TAG ID q** This bit enables read access to the access protected region for transactions with the Master TAG ID q. 0_B Disabled for read access 1_B Enabled for read access

5.4.1.102 HOST command interface and HCI registers, read access enable register B

This register defines which master TAG ID encodings are enabled or disabled for read accesses to the access protected region.

PROT_ACCENHCI_RDB

Offset address: $0B14_H$

HOST command interface and HCI registers, read access enable register B

Application Reset value: $FFFF\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN63	EN62	EN61	EN60	EN59	EN58	EN57	EN56	EN55	EN54	EN53	EN52	EN51	EN50	EN49	EN48
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN47	EN46	EN45	EN44	EN43	EN42	EN41	EN40	EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-63)	q-32	rw	**Read access enable for Master TAG ID q** This bit enables read access to the access protected region for transactions with the Master TAG ID q. 0_B Disabled for read access 1_B Enabled for read access

5.4.1.103 HOST command interface and HCI registers, VM access enable register

This register defines which virtual machine encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each VMx.

PROT_ACCENHCI_VM Offset address: $0B18_H$

HOST command interface and HCI registers, VM access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for VM ID q** This bit enables read access to the access protected region for transactions with the VM ID q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for VM ID q** This bit enables write access to the access protected region for transactions with the VM ID q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; shall be written with 0.

5.4.1.104 HOST command interface and HCI registers, PRS access enable register

This register defines which protection register set encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each PRSx.

PROT_ACCENHCI_PRS Offset address: $0B1C_H$

HOST command interface and HCI registers, PRS access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for PRS q** This bit enables read access to the access protected region for transactions with the PRS q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for PRS q** This bit enables write access to the access protected region for transactions with the PRS q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; shall be written with 0.

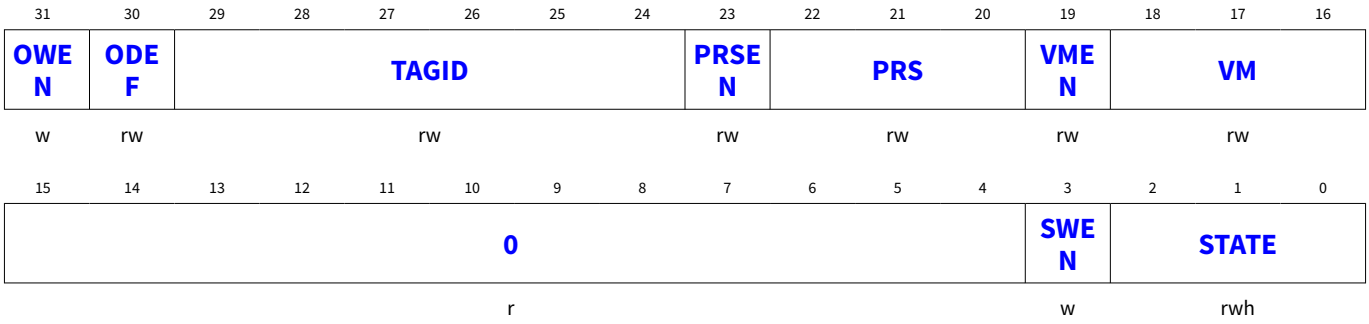
5.4.1.105 PROT register for CSRM command interface and CSCI registers

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner or CS master to update the PROT state as described in the PROT mechanism.

The PROTCSCI register controls lock and unlock of the locally protected CSRM command interface and CSCI registers.

PROT_PROTCSCI Offset address: $0B28_H$

PROT register for CSRM command interface and CSCI registers Application Reset value: $4C00\ 0000_H$



Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
VM	18:16	rw	**Virtual Machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual Machine definition Enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection Set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.
PRSEN	23	rw	**Protection Set definition Enable for PROT owner** 0_B **Disable**: PRS is not part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner

(table continues...)

(continued)

Field	Bits	Type	Description
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; shall be written with 0.

Table 540 Access mode restrictions of PROT_PROTCSCI sorted by descending priority

Mode name	Access mode		Description
PROT_OWNER and write 1 to .**SWEN**	rwh	STATE	PROT owner can modify the PROT state (for allowed and valid transitions)
PROT_OWNER and write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	PROT owner can modify the existing PROT owner
SECURE_MASTER and write 1 to .**SWEN**	rwh	STATE	Secure master can modify the PROT state (for allowed and valid transitions)
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

5.4.1.106 CSRM command interface and CSCI registers, write access enable register A

This register defines which master TAG ID encodings are enabled or disabled for write accesses to the access protected region.

PROT_ACCENCSCI_WRA

CSRM command interface and CSCI registers, write access enable register A

Offset address: $0B30_H$

Application Reset value: $0000\ 1000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Write access enable for Master TAG ID q** This bit enables write access to the access protected region for transactions with the Master TAG ID q. 0_B Disabled for write access 1_B Enabled for write access

5.4.1.107 CSRM command interface and CSCI registers, write access enable register B

This register defines which master TAG ID encodings are enabled or disabled for write accesses to the access protected region.

PROT_ACCENCSCI_WRB

CSRM command interface and CSCI registers, write access enable register B

Offset address: $0B34_H$

Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN63	EN62	EN61	EN60	EN59	EN58	EN57	EN56	EN55	EN54	EN53	EN52	EN51	EN50	EN49	EN48
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN47	EN46	EN45	EN44	EN43	EN42	EN41	EN40	EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-63)	q-32	rw	**Write access enable for Master TAG ID q** This bit enables write access to the access protected region for transactions with the Master TAG ID q. 0_B Disabled for write access 1_B Enabled for write access

5.4.1.108 CSRM command interface and CSCI registers, read access enable register A

This register defines which master TAG ID encodings are enabled or disabled for read accesses to the access protected region.

PROT_ACCENCSCI_RDA

CSRM command interface and CSCI registers, read access enable register A

Offset address: $0B38_H$

Application Reset value: $0000\ 1000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Read access enable for Master TAG ID q** This bit enables read access to the access protected region for transactions with the Master TAG ID q. 0_B Disabled for read access 1_B Enabled for read access

5.4.1.109 CSRM command interface and CSCI registers, read access enable register B

This register defines which master TAG ID encodings are enabled or disabled for read accesses to the access protected region.

PROT_ACCENCSCI_RDB

CSRM command interface and CSCI registers, read access enable register B

Offset address: $0B3C_H$

Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN63	EN62	EN61	EN60	EN59	EN58	EN57	EN56	EN55	EN54	EN53	EN52	EN51	EN50	EN49	EN48
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN47	EN46	EN45	EN44	EN43	EN42	EN41	EN40	EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-63)	q-32	rw	**Read access enable for Master TAG ID q** This bit enables read access to the access protected region for transactions with the Master TAG ID q. 0_B Disabled for read access 1_B Enabled for read access

5.4.1.110 CSRM command interface and CSCI registers, VM access enable register

This register defines which virtual machine encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each VMx.

PROT_ACCENCSCI_VM

CSRM command interface and CSCI registers, VM access enable register

Offset address: $0B40_H$

Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for VM ID q** This bit enables read access to the access protected region for transactions with the VM ID q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for VM ID q** This bit enables write access to the access protected region for transactions with the VM ID q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; shall be written with 0.

5.4.1.111 CSRM command interface and CSCI registers, PRS access enable register

This register defines which protection register set encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each PRSx.

PROT_ACCENCSCI_PRS

CSRM command interface and CSCI registers, PRS access enable register

Offset address: $0B44_H$

Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for PRS q** This bit enables read access to the access protected region for transactions with the PRS q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for PRS q** This bit enables write access to the access protected region for transactions with the PRS q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; shall be written with 0.

5.4.1.112 PROT register for HOST read configurations registers

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner or CS master to update the PROT state as described in the PROT mechanism.

The PROTHRDC register controls lock or unlock of the locally protected HOST read configurations registers.

PROT_PROTHRDC Offset address: $0B50_H$

PROT register for HOST read configurations registers Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OWEN	ODEF	TAGID						PRSEN	PRS			VMEN	VM		
w	rw	rw						rw	rw			rw	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												SWEN	STATE		
r												w	rwh		

Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
VM	18:16	rw	**Virtual Machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual Machine definition Enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection Set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.

(table continues...)

(continued)

Field	Bits	Type	Description
PRSEN	23	rw	**Protection Set definition Enable for PROT owner** 0_B **Disable**: PRS is not part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; shall be written with 0.

Table 541 Access mode restrictions of PROT_PROTHRDC sorted by descending priority

Mode name	Access mode		Description
PROT_OWNER and write 1 to .**SWEN**	rwh	STATE	PROT owner can modify the PROT state (for allowed and valid transitions)
PROT_OWNER and write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	PROT owner can modify the existing PROT owner
SECURE_MASTER and write 1 to .**SWEN**	rwh	STATE	Secure master can modify the PROT state (for allowed and valid transitions)
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

5.4.1.113 HOST read configurations registers, write access enable register A

This register defines which master TAG ID encodings are enabled or disabled for write accesses to the access protected region.

PROT_ACCENHRDC_WRA

HOST read configurations registers, write access enable register A

Offset address: $0B58_H$

Application Reset value: $1000\ 0003_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Write access enable for Master TAG ID q** This bit enables write access to the access protected region for transactions with the Master TAG ID q. 0_B Disabled for write access 1_B Enabled for write access

5.4.1.114 HOST read configurations registers, write access enable register B

This register defines which master TAG ID encodings are enabled or disabled for write accesses to the access protected region.

PROT_ACCENHRDC_WRB Offset address: $0B5C_H$

HOST read configurations registers, write access enable register B Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN63	EN62	EN61	EN60	EN59	EN58	EN57	EN56	EN55	EN54	EN53	EN52	EN51	EN50	EN49	EN48
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN47	EN46	EN45	EN44	EN43	EN42	EN41	EN40	EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-63)	q-32	rw	**Write access enable for Master TAG ID q** This bit enables write access to the access protected region for transactions with the Master TAG ID q. 0_B Disabled for write access 1_B Enabled for write access

5.4.1.115 HOST read configurations registers, read access enable register A

This register defines which master TAG ID encodings are enabled or disabled for read accesses to the access protected region.

PROT_ACCENHRDC_RDA Offset address: $0B60_H$

HOST read configurations registers, read access enable register A Application Reset value: $FFFF\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Read access enable for Master TAG ID q** This bit enables read access to the access protected region for transactions with the Master TAG ID q. 0_B Disabled for read access 1_B Enabled for read access

5.4.1.116 HOST read configurations registers, read access enable register B

This register defines which master TAG ID encodings are enabled or disabled for read accesses to the access protected region.

PROT_ACCENHRDC_RDB Offset address: $0B64_H$

HOST read configurations registers, read access enable register B Application Reset value: $FFFF\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN63	EN62	EN61	EN60	EN59	EN58	EN57	EN56	EN55	EN54	EN53	EN52	EN51	EN50	EN49	EN48
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN47	EN46	EN45	EN44	EN43	EN42	EN41	EN40	EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-63)	q-32	rw	**Read access enable for Master TAG ID q** This bit enables read access to the access protected region for transactions with the Master TAG ID q. 0_B Disabled for read access 1_B Enabled for read access

5.4.1.117 HOST read configurations registers, VM access enable register

This register defines which virtual machine encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each VMx.

PROT_ACCENHRDC_VM Offset address: $0B68_H$

HOST read configurations registers, VM access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for VM ID q** This bit enables read access to the access protected region for transactions with the VM ID q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for VM ID q** This bit enables write access to the access protected region for transactions with the VM ID q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; shall be written with 0.

5.4.1.118 HOST read configurations registers, PRS access enable register

This register defines which protection register set encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each PRSx.

PROT_ACCENHRDC_PRS

HOST read configurations registers, PRS access enable register

Offset address: $0B6C_H$

Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for PRS q** This bit enables read access to the access protected region for transactions with the PRS q. 0_B Disabled for read access 1_B Enabled for read access

(table continues...)

(continued)

Field	Bits	Type	Description
WRq (q=00-07)	q+16	rw	**Write access enable for PRS q** This bit enables write access to the access protected region for transactions with the PRS q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; shall be written with 0.

5.4.1.119 PROT register for CSRM read configurations registers

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner or CS master to update the PROT state as described in the PROT mechanism.

The PROTCSRDC register controls lock and unlock of the locally protected CSRM read configurations registers.

PROT_PROTCSRDC Offset address: $0B78_H$

PROT register for CSRM read configurations registers Application Reset value: $4C00\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OWEN	ODEF	TAGID						PRSEN	PRS			VMEN	VM		
w	rw	rw						rw	rw			rw	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												SWEN	STATE		
r												w	rwh		

Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write

(table continues...)

(continued)

Field	Bits	Type	Description
VM	18:16	rw	**Virtual Machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual Machine definition Enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection Set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.
PRSEN	23	rw	**Protection Set definition Enable for PROT owner** 0_B **Disable**: PRS is not part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; shall be written with 0.

Table 542 Access mode restrictions of PROT_PROTCSRDC sorted by descending priority

Mode name	Access mode		Description
PROT_OWNER and write 1 to .**SWEN**	rwh	STATE	PROT owner can modify the PROT state (for allowed and valid transitions)
PROT_OWNER and write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	PROT owner can modify the existing PROT owner
SECURE_MASTER and write 1 to .**SWEN**	rwh	STATE	Secure master can modify the PROT state (for allowed and valid transitions)
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

5.4.1.120 CSRM read configurations registers, write access enable register A

This register defines which master TAG ID encodings are enabled or disabled for write accesses to the access protected region.

PROT_ACCENCSRDC_WRA

CSRM read configurations registers, write access enable register A

Offset address: $0B80_H$

Application Reset value: $0000\ 1000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Write access enable for Master TAG ID q** This bit enables write access to the access protected region for transactions with the Master TAG ID q. 0_B Disabled for write access 1_B Enabled for write access

5.4.1.121 CSRM read configurations registers, write access enable register B

This register defines which master TAG ID encodings are enabled or disabled for write accesses to the access protected region.

PROT_ACCENCSRDC_WRB

CSRM read configurations registers, write access enable register B

Offset address: $0B84_H$

Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN63	EN62	EN61	EN60	EN59	EN58	EN57	EN56	EN55	EN54	EN53	EN52	EN51	EN50	EN49	EN48
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN47	EN46	EN45	EN44	EN43	EN42	EN41	EN40	EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-63)	q-32	rw	**Write access enable for Master TAG ID q** This bit enables write access to the access protected region for transactions with the Master TAG ID q. 0_B Disabled for write access 1_B Enabled for write access

5.4.1.122 CSRM read configurations registers, read access enable register A

This register defines which master TAG ID encodings are enabled or disabled for read accesses to the access protected region.

PROT_ACCENCSRDC_RDA Offset address: $0B88_H$

CSRM read configurations registers, read access enable register A Application Reset value: $0000\ 1000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Read access enable for Master TAG ID q** This bit enables read access to the access protected region for transactions with the Master TAG ID q. 0_B Disabled for read access 1_B Enabled for read access

5.4.1.123 CSRM read configurations registers, read access enable register B

This register defines which master TAG ID encodings are enabled or disabled for read accesses to the access protected region.

PROT_ACCENCSRDC_RDB Offset address: $0B8C_H$

CSRM read configurations registers, read access enable register B Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN63	EN62	EN61	EN60	EN59	EN58	EN57	EN56	EN55	EN54	EN53	EN52	EN51	EN50	EN49	EN48
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN47	EN46	EN45	EN44	EN43	EN42	EN41	EN40	EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-63)	q-32	rw	**Read access enable for Master TAG ID q** This bit enables read access to the access protected region for transactions with the Master TAG ID q. 0_B Disabled for read access 1_B Enabled for read access

5.4.1.124 CSRM read configurations registers, VM access enable register

This register defines which virtual machine encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each VMx.

PROT_ACCENCSRDC_VM Offset address: $0B90_H$

CSRM read configurations registers, VM access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for VM ID q** This bit enables read access to the access protected region for transactions with the VM ID q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for VM ID q** This bit enables write access to the access protected region for transactions with the VM ID q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; shall be written with 0.

5.4.1.125 CSRM read configurations registers, PRS access enable register

This register defines which protection register set encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each PRSx.

PROT_ACCENCSRDC_PRS Offset address: $0B94_H$

CSRM read configurations registers, PRS access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for PRS q** This bit enables read access to the access protected region for transactions with the PRS q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for PRS q** This bit enables write access to the access protected region for transactions with the PRS q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; shall be written with 0.

5.4.1.126 PROT register for HOST PFLASH memory

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner or CS master to update the PROT state as described in the PROT mechanism.

The PROTHPF register controls lock and unlock of the locally protected HOST PFLASH memory.

PROT_PROTHPF Offset address: $0BA0_H$

PROT register for HOST PFLASH memory Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OWEN	ODEF	TAGID						PRSEN	PRS			VMEN	VM		
w	rw	rw						rw	rw			rw	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												SWEN	STATE		
r												w	rwh		

Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)

(table continues...)

(continued)

Field	Bits	Type	Description
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
VM	18:16	rw	**Virtual Machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual Machine definition Enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection Set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.
PRSEN	23	rw	**Protection Set definition Enable for PROT owner** 0_B **Disable**: PRS is not part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; shall be written with 0.

Table 543 Access mode restrictions of PROT_PROTHPF sorted by descending priority

Mode name	Access mode		Description
PROT_OWNER and write 1 to .**SWEN**	rwh	STATE	PROT owner can modify the PROT state (for allowed and valid transitions)
PROT_OWNER and write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	PROT owner can modify the existing PROT owner
SECURE_MASTER and write 1 to .**SWEN**	rwh	STATE	Secure master can modify the PROT state (for allowed and valid transitions)
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

5.4.1.127 HOST PFLASH memory Pp, write access enable register A

This register defines which master TAG ID encodings are enabled or disabled for write accesses to the access protected region.

PROT_ACCENHPFp_WRA (p=0-11) Offset address: $0BA8_H+p*20_H$

HOST PFLASH memory Pp, write access enable register A Application Reset value: $1000\ 0003_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Write access enable for Master TAG ID q** This bit enables write access to the access protected region for transactions with the Master TAG ID q. 0_B Disabled for write access 1_B Enabled for write access

5.4.1.128 HOST PFLASH memory Pp, write access enable register B

This register defines which master TAG ID encodings are enabled or disabled for write accesses to the access protected region.

PROT_ACCENHPFp_WRB (p=0-11) Offset address: $0BAC_H+p*20_H$

HOST PFLASH memory Pp, write access enable register B Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN63	EN62	EN61	EN60	EN59	EN58	EN57	EN56	EN55	EN54	EN53	EN52	EN51	EN50	EN49	EN48
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN47	EN46	EN45	EN44	EN43	EN42	EN41	EN40	EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-63)	q-32	rw	**Write access enable for Master TAG ID q** This bit enables write access to the access protected region for transactions with the Master TAG ID q. 0_B Disabled for write access 1_B Enabled for write access

5.4.1.129 HOST PFLASH memory Pp, VM access enable register

This register defines which virtual machine encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each VMx.

PROT_ACCENHPFp_VM (p=0-11) Offset address: $0BB8_H+p*20_H$

HOST PFLASH memory Pp, VM access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for VM ID q** This bit enables read access to the access protected region for transactions with the VM ID q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for VM ID q** This bit enables write access to the access protected region for transactions with the VM ID q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; shall be written with 0.

5.4.1.130 HOST PFLASH memory Pp, PRS access enable register

This register defines which protection register set encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each PRSx.

PROT_ACCENHPFp_PRS (p=0-11) Offset address: $0BBC_H+p*20_H$

HOST PFLASH memory Pp, PRS access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for PRS q** This bit enables read access to the access protected region for transactions with the PRS q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for PRS q** This bit enables write access to the access protected region for transactions with the PRS q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; shall be written with 0.

5.4.1.131 HOST PFLASH memory Pp, region lower address register

This register sets the lower bound of the access protected region. Addresses greater than or equal to the lower bound are considered part of the access protected region.

PROT_ACCENHPFp_RGNLA (p=0-11) Offset address: $0BC0_H+p*20_H$

HOST PFLASH memory Pp, region lower address register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0				ADDR											
r				rw											

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR								0							
rw								r							

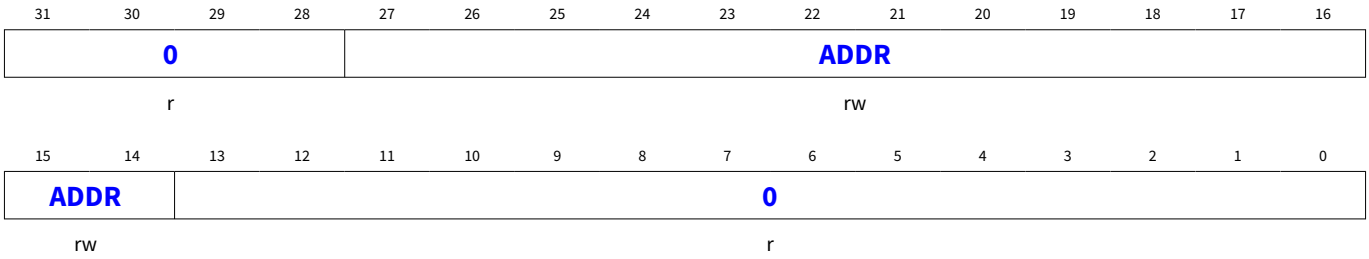
Field	Bits	Type	Description
ADDR	27:14	rw	**Region lower address** Defines bits 27:14 of the lower bound of the access protected region in PFLASH.
0	13:0, 31:28	r	**Reserved** Read as 0; shall be written with 0.

5.4.1.132 HOST PFLASH memory Pp, region upper address register

This register sets the upper bound of the access protected region. Addresses stricly less than the upper bound are considered part of the access protected region.

PROT_ACCENHPFp_RGNUA (p=0-11) Offset address: $0BC4_H+p*20_H$

HOST PFLASH memory Pp, region upper address register Application Reset value: $0FFF\ C000_H$



Field	Bits	Type	Description
ADDR	27:14	rw	**Region upper address** Defines bits 27:14 of the upper bound of the access protected region in PFLASH.
0	13:0, 31:28	r	**Reserved** Read as 0; shall be written with 0.

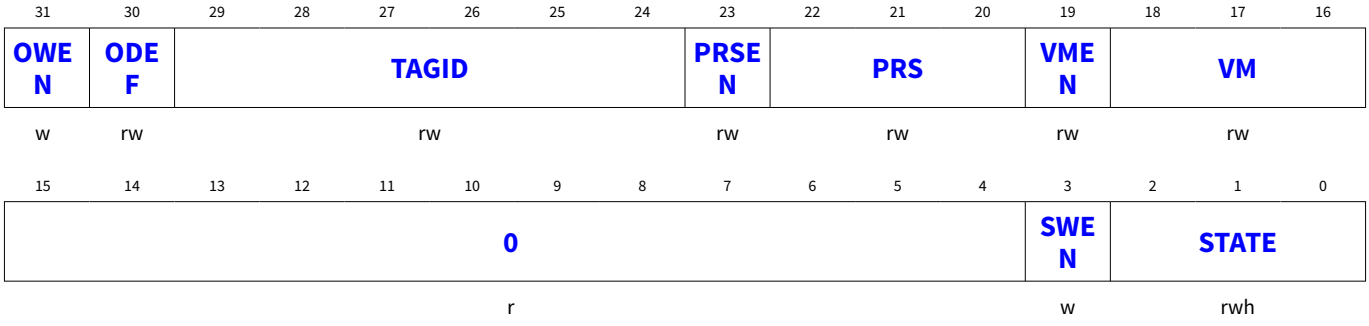
5.4.1.133 PROT register for HOST DFLASH (EEPROM) and HOST UCB memory

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner or CS master to update the PROT state as described in the PROT mechanism.

The PROTHDF register controls lock and unlock of the locally protected HOST DFLASH (EEPROM) and HOST UCB memory.

PROT_PROTHDF Offset address: $0DA8_H$

PROT register for HOST DFLASH [EEPROM) and HOST UCB memory Application Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
VM	18:16	rw	**Virtual Machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual Machine definition Enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection Set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.
PRSEN	23	rw	**Protection Set definition Enable for PROT owner** 0_B **Disable**: PRS is not part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; shall be written with 0.

Table 544 Access mode restrictions of PROT_PROTHDF sorted by descending priority

Mode name	Access mode		Description
PROT_OWNER and write 1 to .**SWEN**	rwh	STATE	PROT owner can modify the PROT state (for allowed and valid transitions)
PROT_OWNER and write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	PROT owner can modify the existing PROT owner
SECURE_MASTER and write 1 to .**SWEN**	rwh	STATE	Secure master can modify the PROT state (for allowed and valid transitions)
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

5.4.1.134 HOST DFLASH (EEPROM) memory Dd, write access enable register A

This register defines which master TAG ID encodings are enabled or disabled for write accesses to the access protected region.

PROT_ACCENHDFd_WRA (d=0-7) Offset address: $0DB0_H+d*20_H$

HOST DFLASH [EEPROM) memory Dd, write access enable register A Application Reset value: $1000\ 0003_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	**EN30**	**EN29**	**EN28**	**EN27**	**EN26**	**EN25**	**EN24**	**EN23**	**EN22**	**EN21**	**EN20**	**EN19**	**EN18**	**EN17**	**EN16**
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	**EN14**	**EN13**	**EN12**	**EN11**	**EN10**	**EN09**	**EN08**	**EN07**	**EN06**	**EN05**	**EN04**	**EN03**	**EN02**	**EN01**	**EN00**
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Write access enable for Master TAG ID q** This bit enables write access to the access protected region for transactions with the Master TAG ID q. 0_B Disabled for write access 1_B Enabled for write access

5.4.1.135 HOST DFLASH (EEPROM) memory Dd, write access enable register B

This register defines which master TAG ID encodings are enabled or disabled for write accesses to the access protected region.

PROT_ACCENHDFd_WRB (d=0-7) Offset address: $0DB4_H+d*20_H$

HOST DFLASH [EEPROM) memory Dd, write access enable register B Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN63	EN62	EN61	EN60	EN59	EN58	EN57	EN56	EN55	EN54	EN53	EN52	EN51	EN50	EN49	EN48
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN47	EN46	EN45	EN44	EN43	EN42	EN41	EN40	EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-63)	q-32	rw	**Write access enable for Master TAG ID q** This bit enables write access to the access protected region for transactions with the Master TAG ID q. 0_B Disabled for write access 1_B Enabled for write access

5.4.1.136 HOST DFLASH (EEPROM) memory Dd, read access enable register A

This register defines which master TAG ID encodings are enabled or disabled for read accesses to the access protected region.

PROT_ACCENHDFd_RDA (d=0-7)

Offset address: $0DB8_H+d*20_H$

HOST DFLASH [EEPROM) memory Dd, read access enable register A

Application Reset value: $FFFF\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Read access enable for Master TAG ID q** This bit enables read access to the access protected region for transactions with the Master TAG ID q. 0_B Disabled for read access 1_B Enabled for read access

5.4.1.137 HOST DFLASH (EEPROM) memory Dd, read access enable register B

This register defines which master TAG ID encodings are enabled or disabled for read accesses to the access protected region.

PROT_ACCENHDFd_RDB (d=0-7)

Offset address: $0DBC_H+d*20_H$

HOST DFLASH [EEPROM) memory Dd, read access enable register B

Application Reset value: $FFFF\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN63	EN62	EN61	EN60	EN59	EN58	EN57	EN56	EN55	EN54	EN53	EN52	EN51	EN50	EN49	EN48
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN47	EN46	EN45	EN44	EN43	EN42	EN41	EN40	EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-63)	q-32	rw	**Read access enable for Master TAG ID q** This bit enables read access to the access protected region for transactions with the Master TAG ID q. 0_B Disabled for read access 1_B Enabled for read access

5.4.1.138 HOST DFLASH (EEPROM) memory Dd, VM access enable register

This register defines which virtual machine encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each VMx.

PROT_ACCENHDFd_VM (d=0-7)

HOST DFLASH [EEPROM) memory Dd, VM access enable register

Offset address: $0DC0_H+d*20_H$

Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for VM ID q** This bit enables read access to the access protected region for transactions with the VM ID q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for VM ID q** This bit enables write access to the access protected region for transactions with the VM ID q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; shall be written with 0.

5.4.1.139 HOST DFLASH (EEPROM) memory Dd, PRS access enable register

This register defines which protection register set encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each PRSx.

PROT_ACCENHDFd_PRS (d=0-7)

HOST DFLASH [EEPROM) memory Dd, PRS access enable register

Offset address: $0DC4_H+d*20_H$

Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for PRS q** This bit enables read access to the access protected region for transactions with the PRS q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for PRS q** This bit enables write access to the access protected region for transactions with the PRS q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; shall be written with 0.

5.4.1.140 HOST DFLASH (EEPROM) memory Dd, region lower address register

This register sets the lower bound of the access protected region. Addresses greater than or equal to the lower bound are considered part of the access protected region.

PROT_ACCENHDFd_RGNLA (d=0-7)

HOST DFLASH [EEPROM) memory Dd, region lower address register

Offset address: $0DC8_H+d*20_H$

Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0											ADDR				
r											rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR					0										
rw					r										

Field	Bits	Type	Description
ADDR	20:11	rw	**Region lower address** Defines bits 20:11 of the lower bound of the access protected region in DFLASH.
0	10:0, 31:21	r	**Reserved** Read as 0; shall be written with 0.

5.4.1.141 HOST DFLASH (EEPROM) memory Dd, region upper address register

This register sets the upper bound of the access protected region. Addresses stricly less than the upper bound are considered part of the access protected region.

PROT_ACCENHDFd_RGNUA (d=0-7) Offset address: $0DCC_H+d*20_H$

HOST DFLASH [EEPROM) memory Dd, region upper address register Application Reset value: $001F\ F800_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0											ADDR				
r											rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR					0										
rw					r										

Field	Bits	Type	Description
ADDR	20:11	rw	**Region upper address** Defines bits 20:11 of the upper bound of the access protected region in DFLASH.
0	10:0, 31:21	r	**Reserved** Read as 0; shall be written with 0.

5.4.1.142 HOST UCB memory, write access enable register A

This register defines which master TAG ID encodings are enabled or disabled for write accesses to the access protected region.

PROT_ACCENHDFUCB_WRA Offset address: $0EB0_H$

HOST UCB memory, write access enable register A Application Reset value: $1000\ 0003_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Write access enable for Master TAG ID q** This bit enables write access to the access protected region for transactions with the Master TAG ID q. 0_B Disabled for write access 1_B Enabled for write access

5.4.1.143 HOST UCB memory, write access enable register B

This register defines which master TAG ID encodings are enabled or disabled for write accesses to the access protected region.

PROT_ACCENHDFUCB_WRB Offset address: $0EB4_H$

HOST UCB memory, write access enable register B Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN63	EN62	EN61	EN60	EN59	EN58	EN57	EN56	EN55	EN54	EN53	EN52	EN51	EN50	EN49	EN48
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN47	EN46	EN45	EN44	EN43	EN42	EN41	EN40	EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-63)	q-32	rw	**Write access enable for Master TAG ID q** This bit enables write access to the access protected region for transactions with the Master TAG ID q. 0_B Disabled for write access 1_B Enabled for write access

5.4.1.144 HOST UCB memory, read access enable register A

This register defines which master TAG ID encodings are enabled or disabled for read accesses to the access protected region.

PROT_ACCENHDFUCB_RDA Offset address: $0EB8_H$

HOST UCB memory, read access enable register A Application Reset value: $FFFF\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Read access enable for Master TAG ID q** This bit enables read access to the access protected region for transactions with the Master TAG ID q. 0_B Disabled for read access 1_B Enabled for read access

5.4.1.145 HOST UCB memory, read access enable register B

This register defines which master TAG ID encodings are enabled or disabled for read accesses to the access protected region.

PROT_ACCENHDFUCB_RDB Offset address: $0EBC_H$

HOST UCB memory, read access enable register B Application Reset value: $FFFF\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN63	EN62	EN61	EN60	EN59	EN58	EN57	EN56	EN55	EN54	EN53	EN52	EN51	EN50	EN49	EN48
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN47	EN46	EN45	EN44	EN43	EN42	EN41	EN40	EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-63)	q-32	rw	**Read access enable for Master TAG ID q** This bit enables read access to the access protected region for transactions with the Master TAG ID q. 0_B Disabled for read access 1_B Enabled for read access

5.4.1.146 HOST UCB memory, VM access enable register

This register defines which virtual machine encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each VMx.

PROT_ACCENHDFUCB_VM Offset address: $0EC0_H$

HOST UCB memory, VM access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for VM ID q** This bit enables read access to the access protected region for transactions with the VM ID q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for VM ID q** This bit enables write access to the access protected region for transactions with the VM ID q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; shall be written with 0.

5.4.1.147 HOST UCB memory, PRS access enable register

This register defines which protection register set encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each PRSx.

PROT_ACCENHDFUCB_PRS Offset address: $0EC4_H$

HOST UCB memory, PRS access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for PRS q** This bit enables read access to the access protected region for transactions with the PRS q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for PRS q** This bit enables write access to the access protected region for transactions with the PRS q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; shall be written with 0.

5.4.1.148 PROT register for CSRM PFLASH memory

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner or CS master to update the PROT state as described in the PROT mechanism.

The PROTCSPF register controls lock and unlock of the locally protected CSRM PFLASH memory.

PROT_PROTCSPF Offset address: $0ED0_H$

PROT register for CSRM PFLASH memory Application Reset value: $4C00\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OWEN	ODEF	TAGID						PRSEN	PRS			VMEN	VM		
w	rw	rw						rw	rw			rw	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												SWEN	STATE		
r												w	rwh		

Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
VM	18:16	rw	**Virtual Machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual Machine definition Enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection Set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.

(table continues...)

(continued)

Field	Bits	Type	Description
PRSEN	23	rw	**Protection Set definition Enable for PROT owner** 0_B **Disable**: PRS is not part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; shall be written with 0.

Table 545 Access mode restrictions of PROT_PROTCSPF sorted by descending priority

Mode name	Access mode		Description
PROT_OWNER and write 1 to .**SWEN**	rwh	STATE	PROT owner can modify the PROT state (for allowed and valid transitions)
PROT_OWNER and write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	PROT owner can modify the existing PROT owner
SECURE_MASTER and write 1 to .**SWEN**	rwh	STATE	Secure master can modify the PROT state (for allowed and valid transitions)
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

5.4.1.149 CSRM PFLASH memory Pp, write access enable register A

This register defines which master TAG ID encodings are enabled or disabled for write accesses to the access protected region.

PROT_ACCENCSPFp_WRA (p=0-3) Offset address: $0ED8_H+p*20_H$

CSRM PFLASH memory Pp, write access enable register A Application Reset value: $0000\ 1000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Write access enable for Master TAG ID q** This bit enables write access to the access protected region for transactions with the Master TAG ID q. 0_B Disabled for write access 1_B Enabled for write access

5.4.1.150 CSRM PFLASH memory Pp, write access enable register B

This register defines which master TAG ID encodings are enabled or disabled for write accesses to the access protected region.

PROT_ACCENCSPFp_WRB (p=0-3)

Offset address: $0EDC_H+p*20_H$

CSRM PFLASH memory Pp, write access enable register B

Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN63	EN62	EN61	EN60	EN59	EN58	EN57	EN56	EN55	EN54	EN53	EN52	EN51	EN50	EN49	EN48
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN47	EN46	EN45	EN44	EN43	EN42	EN41	EN40	EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-63)	q-32	rw	**Write access enable for Master TAG ID q** This bit enables write access to the access protected region for transactions with the Master TAG ID q. 0_B Disabled for write access 1_B Enabled for write access

5.4.1.151 CSRM PFLASH memory Pp, VM access enable register

This register defines which virtual machine encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each VMx.

PROT_ACCENCSPFp_VM (p=0-3)

Offset address: $0EE8_H+p*20_H$

CSRM PFLASH memory Pp, VM access enable register

Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for VM ID q** This bit enables read access to the access protected region for transactions with the VM ID q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for VM ID q** This bit enables write access to the access protected region for transactions with the VM ID q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; shall be written with 0.

5.4.1.152 CSRM PFLASH memory Pp, PRS access enable register

This register defines which protection register set encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each PRSx.

PROT_ACCENCSPFp_PRS (p=0-3) Offset address: $0EEC_H+p*20_H$

CSRM PFLASH memory Pp, PRS access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for PRS q** This bit enables read access to the access protected region for transactions with the PRS q. 0_B Disabled for read access 1_B Enabled for read access

(table continues...)

(continued)

Field	Bits	Type	Description
WRq (q=00-07)	q+16	rw	**Write access enable for PRS q** This bit enables write access to the access protected region for transactions with the PRS q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; shall be written with 0.

5.4.1.153 CSRM PFLASH memory Pp, region lower address register

This register sets the lower bound of the access protected region. Addresses greater than or equal to the lower bound are considered part of the access protected region.

PROT_ACCENCSPFp_RGNLA (p=0-3)

CSRM PFLASH memory Pp, region lower address register

Offset address: $0EF0_H+p*20_H$

Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0				ADDR											
r				rw											

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR								0							
rw								r							

Field	Bits	Type	Description
ADDR	27:14	rw	**Region lower address** Defines bits 27:14 of the lower bound of the access protected region in PFLASH.
0	13:0, 31:28	r	**Reserved** Read as 0; shall be written with 0.

5.4.1.154 CSRM PFLASH memory Pp, region upper address register

This register sets the upper bound of the access protected region. Addresses stricly less than the upper bound are considered part of the access protected region.

PROT_ACCENCSPFp_RGNUA (p=0-3)

CSRM PFLASH memory Pp, region upper address register

Offset address: $0EF4_H+p*20_H$

Application Reset value: $0FFF\ C000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0				ADDR											
r				rw											

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR								0							
rw								r							

Field	Bits	Type	Description
ADDR	27:14	rw	**Region upper address** Defines bits 27:14 of the upper bound of the access protected region in PFLASH.
0	13:0, 31:28	r	**Reserved** Read as 0; shall be written with 0.

5.4.1.155 PROT register for CSRM DFLASH (EEPROM) and CSRM UCB memory

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner or CS master to update the PROT state as described in the PROT mechanism.

The PROTCSDF register controls lock and unlock of the locally protected CSRM DFLASH (EEPROM) and CSRM UCB memory.

PROT_PROTCSDF Offset address: $0F58_H$

PROT register for CSRM DFLASH [EEPROM) and CSRM UCB memory Application Reset value: $4C00\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OWEN	ODEF	TAGID						PRSEN	PRS			VMEN	VM		
w	rw	rw						rw	rw			rw	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												SWEN	STATE		
r												w	rwh		

Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write

(table continues...)

(continued)

Field	Bits	Type	Description
VM	18:16	rw	**Virtual Machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual Machine definition Enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection Set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.
PRSEN	23	rw	**Protection Set definition Enable for PROT owner** 0_B **Disable**: PRS is not part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; shall be written with 0.

Table 546 Access mode restrictions of PROT_PROTCSDF sorted by descending priority

Mode name	Access mode		Description
PROT_OWNER and write 1 to .**SWEN**	rwh	STATE	PROT owner can modify the PROT state (for allowed and valid transitions)
PROT_OWNER and write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	PROT owner can modify the existing PROT owner
SECURE_MASTER and write 1 to .**SWEN**	rwh	STATE	Secure master can modify the PROT state (for allowed and valid transitions)
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

5.4.1.156 CSRM DFLASH (EEPROM) memory Dd, write access enable register A

This register defines which master TAG ID encodings are enabled or disabled for write accesses to the access protected region.

PROT_ACCENCSDFd_WRA (d=0-3)

CSRM DFLASH [EEPROM) memory Dd, write access enable register A

Offset address: $0F60_H+d*20_H$

Application Reset value: $0000\ 1000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Write access enable for Master TAG ID q** This bit enables write access to the access protected region for transactions with the Master TAG ID q. 0_B Disabled for write access 1_B Enabled for write access

5.4.1.157 CSRM DFLASH (EEPROM) memory Dd, write access enable register B

This register defines which master TAG ID encodings are enabled or disabled for write accesses to the access protected region.

PROT_ACCENCSDFd_WRB (d=0-3)

CSRM DFLASH [EEPROM) memory Dd, write access enable register B

Offset address: $0F64_H+d*20_H$

Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN63	EN62	EN61	EN60	EN59	EN58	EN57	EN56	EN55	EN54	EN53	EN52	EN51	EN50	EN49	EN48
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN47	EN46	EN45	EN44	EN43	EN42	EN41	EN40	EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-63)	q-32	rw	**Write access enable for Master TAG ID q** This bit enables write access to the access protected region for transactions with the Master TAG ID q. 0_B Disabled for write access 1_B Enabled for write access

5.4.1.158 CSRM DFLASH (EEPROM) memory Dd, read access enable register A

This register defines which master TAG ID encodings are enabled or disabled for read accesses to the access protected region.

PROT_ACCENCSDFd_RDA (d=0-3)

CSRM DFLASH [EEPROM) memory Dd, read access enable register A

Offset address: $0F68_H+d*20_H$

Application Reset value: $0000\ 1000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Read access enable for Master TAG ID q** This bit enables read access to the access protected region for transactions with the Master TAG ID q. 0_B Disabled for read access 1_B Enabled for read access

5.4.1.159 CSRM DFLASH (EEPROM) memory Dd, read access enable register B

This register defines which master TAG ID encodings are enabled or disabled for read accesses to the access protected region.

PROT_ACCENCSDFd_RDB (d=0-3)

CSRM DFLASH [EEPROM) memory Dd, read access enable register B

Offset address: $0F6C_H+d*20_H$

Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN63	EN62	EN61	EN60	EN59	EN58	EN57	EN56	EN55	EN54	EN53	EN52	EN51	EN50	EN49	EN48
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN47	EN46	EN45	EN44	EN43	EN42	EN41	EN40	EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-63)	q-32	rw	**Read access enable for Master TAG ID q** This bit enables read access to the access protected region for transactions with the Master TAG ID q. 0_B Disabled for read access 1_B Enabled for read access

5.4.1.160 CSRM DFLASH (EEPROM) memory Dd, VM access enable register

This register defines which virtual machine encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each VMx.

PROT_ACCENCSDFd_VM (d=0-3)

CSRM DFLASH [EEPROM] memory Dd, VM access enable register

Offset address: $0F70_H+d*20_H$

Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for VM ID q** This bit enables read access to the access protected region for transactions with the VM ID q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for VM ID q** This bit enables write access to the access protected region for transactions with the VM ID q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; shall be written with 0.

5.4.1.161 CSRM DFLASH (EEPROM) memory Dd, PRS access enable register

This register defines which protection register set encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each PRSx.

PROT_ACCENCSDFd_PRS (d=0-3)

CSRM DFLASH [EEPROM] memory Dd, PRS access enable register

Offset address: $0F74_H+d*20_H$

Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for PRS q** This bit enables read access to the access protected region for transactions with the PRS q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for PRS q** This bit enables write access to the access protected region for transactions with the PRS q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; shall be written with 0.

5.4.1.162 CSRM DFLASH (EEPROM) memory Dd, region lower address register

This register sets the lower bound of the access protected region. Addresses greater than or equal to the lower bound are considered part of the access protected region.

PROT_ACCENCSDFd_RGNLA (d=0-3)

CSRM DFLASH [EEPROM) memory Dd, region lower address register

Offset address: $0F78_H+d*20_H$

Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0											ADDR				
r											rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR					0										
rw					r										

Field	Bits	Type	Description
ADDR	20:11	rw	**Region lower address** Defines bits 20:11 of the lower bound of the access protected region in DFLASH.
0	10:0, 31:21	r	**Reserved** Read as 0; shall be written with 0.

5.4.1.163 CSRM DFLASH (EEPROM) memory Dd, region upper address register

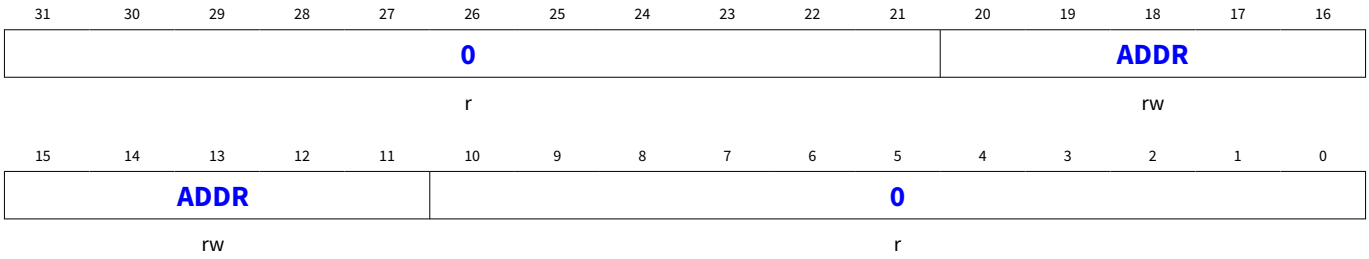
This register sets the upper bound of the access protected region. Addresses stricly less than the upper bound are considered part of the access protected region.

PROT_ACCENCSDFd_RGNUA (d=0-3)

CSRM DFLASH [EEPROM) memory Dd, region upper address register

Offset address: $0F7C_H+d*20_H$

Application Reset value: $001F\ F800_H$



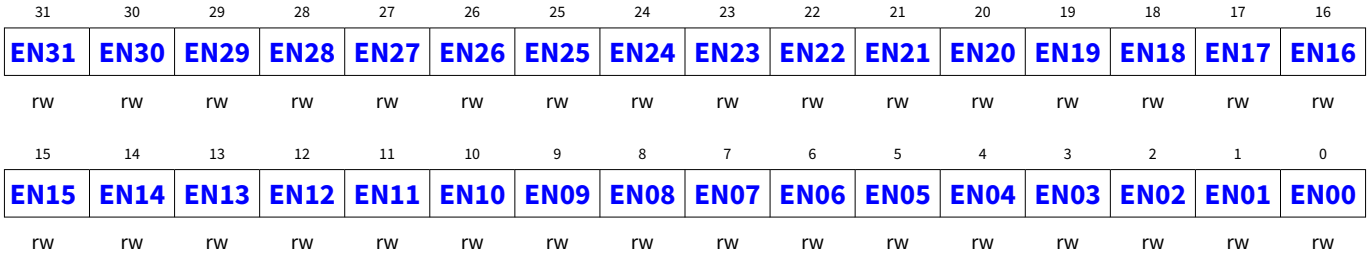
Field	Bits	Type	Description
ADDR	20:11	rw	**Region upper address** Defines bits 20:11 of the upper bound of the access protected region in DFLASH.
0	10:0, 31:21	r	**Reserved** Read as 0; shall be written with 0.

5.4.1.164 CSRM UCB memory, write access enable register A

This register defines which master TAG ID encodings are enabled or disabled for write accesses to the access protected region.

PROT_ACCENCSDFUCB_WRA Offset address: $0FE0_H$

CSRM UCB memory, write access enable register A Application Reset value: $0000\ 1000_H$



Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Write access enable for Master TAG ID q** This bit enables write access to the access protected region for transactions with the Master TAG ID q. 0_B Disabled for write access 1_B Enabled for write access

5.4.1.165 CSRM UCB memory, write access enable register B

This register defines which master TAG ID encodings are enabled or disabled for write accesses to the access protected region.

PROT_ACCENCSDFUCB_WRB Offset address: $0FE4_H$

CSRM UCB memory, write access enable register B Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN63	EN62	EN61	EN60	EN59	EN58	EN57	EN56	EN55	EN54	EN53	EN52	EN51	EN50	EN49	EN48
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN47	EN46	EN45	EN44	EN43	EN42	EN41	EN40	EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-63)	q-32	rw	**Write access enable for Master TAG ID q** This bit enables write access to the access protected region for transactions with the Master TAG ID q. 0_B Disabled for write access 1_B Enabled for write access

5.4.1.166 CSRM UCB memory, read access enable register A

This register defines which master TAG ID encodings are enabled or disabled for read accesses to the access protected region.

PROT_ACCENCSDFUCB_RDA Offset address: $0FE8_H$

CSRM UCB memory, read access enable register A Application Reset value: $0000\ 1000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Read access enable for Master TAG ID q** This bit enables read access to the access protected region for transactions with the Master TAG ID q. 0_B Disabled for read access 1_B Enabled for read access

5.4.1.167 CSRM UCB memory, read access enable register B

This register defines which master TAG ID encodings are enabled or disabled for read accesses to the access protected region.

PROT_ACCENCSDFUCB_RDB Offset address: $0FEC_H$

CSRM UCB memory, read access enable register B Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN63	EN62	EN61	EN60	EN59	EN58	EN57	EN56	EN55	EN54	EN53	EN52	EN51	EN50	EN49	EN48
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN47	EN46	EN45	EN44	EN43	EN42	EN41	EN40	EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-63)	q-32	rw	**Read access enable for Master TAG ID q** This bit enables read access to the access protected region for transactions with the Master TAG ID q. 0_B Disabled for read access 1_B Enabled for read access

5.4.1.168 CSRM UCB memory, VM access enable register

This register defines which virtual machine encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each VMx.

PROT_ACCENCSDFUCB_VM Offset address: $0FF0_H$

CSRM UCB memory, VM access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for VM ID q** This bit enables read access to the access protected region for transactions with the VM ID q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for VM ID q** This bit enables write access to the access protected region for transactions with the VM ID q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; shall be written with 0.

5.4.1.169 CSRM UCB memory, PRS access enable register

This register defines which protection register set encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each PRSx.

PROT_ACCENCSDFUCB_PRS Offset address: $0FF4_H$

CSRM UCB memory, PRS access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for PRS q** This bit enables read access to the access protected region for transactions with the PRS q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for PRS q** This bit enables write access to the access protected region for transactions with the PRS q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; shall be written with 0.

5.4.1.170 APU bypass

The APU protection needs to be bypassed at customer delivery state and until the customer content is provided for the first time.

This register allow to bypass the APU protection and then activate the usage of the APU by modifying the content of the UCB_CS_PFLASH_OTPx_COPY and UCB_CS_PFLASH_OTPx_ORIG.

PROT_APUBYPASS Offset address: 1000_H

APU bypass Reset values see: Table 547

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								BYPASS							
r								rh							

Field	Bits	Type	Description
BYPASS	7:0	rh	**BYPASS** These registers contains the value read out of the UCB and evaluated to authorise the bypass of the APUs. Note that the expected values for this registers are 0x55 and 0xFF but all other values behaves like 0xFF (APU in use). 55_H APU bypassed FF_H APU in use **others,** Reserved
0	31:8	r	**Reserved** Read as 0; shall be written with 0.

Table 547 Reset values of PROT_APUBYPASS

Reset type	Reset value	Note
Application Reset	0000 00FF_H	
After Boot-FW Value	0000 00$--_H$	Value recovered from UCB

5.4.2 FSI registers

5.4.2.1 Register overview - access mode glossary

Table 548 Register overview - access mode glossary

Keyword	Description
M	Indicates a module specific access condition. Refer to the register description for details of the specific access condition.

5.4.2.2 Registers overview - FSI (ascending offset address)

Table 549 Registers overview - FSI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
COMM0	Communication register 0	0000_H	M	M	System Reset	954
COMM1	Communication register 1	0001_H	M	M	System Reset	954
COMM2	Communication register 2	0002_H	M	M	System Reset	955

5.4.2.3 Communication register 0

This 8 bit register can be written by the DMU and by the FSI and is used to transfer options for the command sequences to FSI or status information back to the system.

Properties:

[[NVM_MISR : NO]]
[[NVM_ROM : NO]]
[[NVM_RECOVER : NO]]
[[NVM_CRITICAL : NO]]
[[NVM_TYPE : FSI_SFR_REQ]]

COMM0 Offset address: 0000_H

Communication register 0 System Reset value: 00_H

7	6	5	4	3	2	1	0
COMM0							
rwh							

Field	Bits	Type	Description
COMM0	7:0	rwh	**Communication register 0** This 8 bit register can be written by the DMU and by the FSI and is used to transfer options for the command sequences to FSI or status information back to the system.

5.4.2.4 Communication register 1

This 8 bit register can be written by the DMU and by the FSI and is used to transfer options for the command sequences to FSI or status information back to the system.

Properties:

[[NVM_MISR : NO]]
[[NVM_ROM : NO]]
[[NVM_RECOVER : NO]]
[[NVM_CRITICAL : NO]]
[[NVM_TYPE : FSI_SFR_REQ]]

COMM1 Offset address: 0001_H

Communication register 1 System Reset value: 00_H

7	6	5	4	3	2	1	0
COMM1							
rwh							

Field	Bits	Type	Description
COMM1	7:0	rwh	**Communication register 1** This 8 bit register can be written by the DMU and by the FSI and is used to transfer options for the command sequences to FSI or status information back to the system.

5.4.2.5 Communication register 2

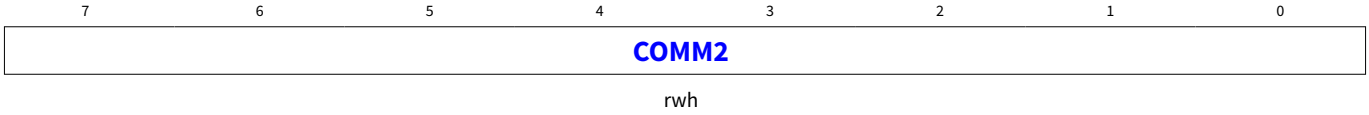
This 8 bit register can be written by the DMU and by the FSI and is used to transfer options for the command sequences to FSI or status information back to the system.

Properties:

[[NVM_MISR : NO]]
[[NVM_ROM : NO]]
[[NVM_RECOVER : NO]]
[[NVM_CRITICAL : NO]]
[[NVM_TYPE : FSI_SFR_REQ]]

COMM2 Offset address: 0002_H

Communication register 2 System Reset value: 00_H



Field	Bits	Type	Description
COMM2	7:0	rwh	**Communication register 2** This 8 bit register can be written by the DMU and by the FSI and is used to transfer options for the command sequences to FSI or status information back to the system.

5.4.3 PFRWB registers

5.4.3.1 Register overview - access mode glossary

Table 550 Register overview - access mode glossary

Keyword	Description
M	Indicates a module specific access condition. Refer to the register description for details of the specific access condition.
32	Access only when using 32-bit width.
BE	Always returns a Bus Error.

5.4.3.2 Registers overview - PFRWB (ascending offset address)

Table 551 Registers overview - PFRWB (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
UR_FLASHCON2	PFRWB FLASH configuration register 2	000000_H	M, 32	M, 32	Application Reset	956
UR_FLASHCON3	PFRWB FLASH configuration register 3	000004_H	M, 32	M, 32	Application Reset	958
UR_ECCR	ECC read register	002000_H	M, 32	BE	System Reset	960

(table continues...)

Table 551 (continued) Registers overview - PFRWB (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
UR_ECCS	ECC status register	002004_H	M, 32	BE	System Reset	960
UR_SBABx	Single bit error address buffer x	002080_H +x*4	M, 32	BE	System Reset	963
UR_DBABx	Double bit error address buffer x	002100_H +x*4	M, 32	BE	System Reset	964
UR_UBAB0	Uncorrectable error address buffer x	002180_H	M, 32	BE	System Reset	964

5.4.3.3 PFRWB FLASH configuration register 2

The Flash Configuration register 2 enables to control the read margins, the ECC behavior and the recording in the xBAB (ECC error address buffer) registers.

If a change between the read margin settings is done using HMAR or MARSEL, the system must wait a delay time $t_{FL_MarginDel}$ until the next read is requested.

UR_FLASHCON2 Offset address: 000000_H

PFRWB FLASH configuration register 2 Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	UBABCLR	0	DBABCLR	0	SBABCLR	0	ECCSCLR	0							RECDIS
r	w	r	w	r	w	r	w	r							rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			TDISUDATA	0	MASKUECC	0	ECCCORDIS	0					HMAR	0	MARSEL
r			rw	r	rw	r	rw	r					rw	r	rw

Field	Bits	Type	Description
MARSEL	0	rw	**Margin Read Selection** This bitfield enables to use the hard margins to read out of the local PFLASH bank. This bitfield is protected by a SFF. 0_B Enables the standard read margins for reading the local PFLASH bank. 1_B Enables the read margins chosen by PFRWBx_FLASHCON2.HMAR for reading the local PFLASH bank.

(table continues...)

(continued)

Field	Bits	Type	Description
HMAR	2	rw	**Hard Margin Selection** This bitfield enables to chose between the thigh-0 and the tight-1 margin for reading out of the local PFLASH bank. This setting is effective only if PFRWBx_FLASCON2.MARSEL=1_B. This bitfield is protected by a SFF. 0_B Enables the tight-0 read margins (sub-optimal 0 read as 1) for reading the local PFLASH bank. 1_B Enables the tight-1 read margins (sub-optimal 1 read as 0) for reading the local PFLASH bank.
ECCCORDIS	8	rw	**ECC correction disable** This bitfield disables the ECC correction for the local PFLASH bank. This bitfield is protected by a SFF. 0_B Enable the ECC correction for reading the local PFLASH bank. 1_B Disable the ECC correction for reading the local PFLASH bank.
MASKUECC	10	rw	**Mask PFLASH ECC uncorrectable errors** This bitfield enables to mask the uncorrectable ECC errors. Note that the ECCS and UBAB registers will still store the uncorrectable error, and an alarm is generated but the master will not trap. This bitfield is protected by a SFF. 0_B Enable the propagation of the uncorrectable ECC error to any master reading the local PFLASH. 1_B Disable the propagation of the uncorrectable ECC error to any master reading the local PFLASH bank.
TDISUDATA	12	rw	**Trap disable and uncorrected data** Note: Test setting, do not use without clarification with Infineon FAE. This bitfield shall be set to 0_B in field and used only for test purpose. It disable the traps and always send uncorrected data to the CPU. This bitfield is protected by a SFF. Note that setting this bit leads to safety alarms from the ECC online monitor on correctable error and from the UBAB on uncorrectable error found. 0_B User mode : enable the propagation of ECC errors and always provide the corrected data when necessary to any master reading the local PFLASH. 1_B Test mode : disable the propagation of the any ECC error and always provide the uncorrected data to any master reading the local PFLASH bank.
RECDIS	16	rw	**Address Buffer Record Disable** These bitfield disables the recording in the xBaBs registers if set to 1_B. SBAB, DBAB and UBAB alarms will not be generated if RECDIS is 1_B This bitfield is protected by a SFF. 0_B Enable local PFlash bank ECC error recording in SBAB, DBAB, UBAB. 1_B Disable local PFLASH bank ECC error recording in SBAB, DBAB, and UBAB.

(table continues...)

(continued)

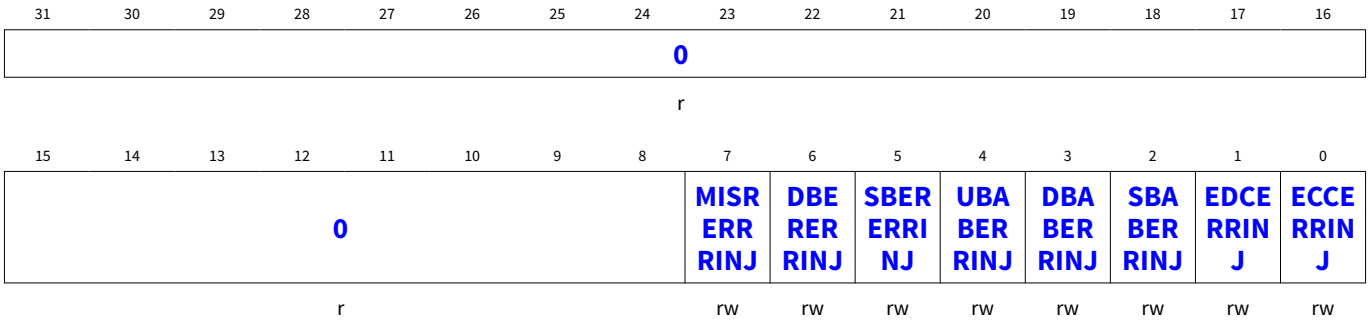
Field	Bits	Type	Description
ECCSCLR	24	w	**Clear ECCS status register** This bitfield enables to clear the ECC status register. This bitfield is protected by a SFF. 0_B No action. 1_B Clear the ECCS registers and the SBER and DBER alarms.
SBABCLR	26	w	**Clear SBAB record registers** This bitfield enables to clear the SBAB registers. This bitfield is protected by a SFF. 0_B No action. 1_B Clear the SBAB registers and the SBAB alarm.
DBABCLR	28	w	**Clear DBAB record registers** This bitfield enables to clear the DBAB registers, This bitfield is protected by a SFF. 0_B No action. 1_B Clear the DBAB registers and the DBAB alarm.
UBABCLR	30	w	**Clear UBAB record registers** This bitfield enables to clear the UBAB registers. This bitfield is protected by a SFF. 0_B No action. 1_B Clear the UBAB registers and the UBAB alarm.
0	1, 7:3, 9, 11, 15:13, 23:17, 25, 27, 29, 31	r	**Reserved** Read as 0; shall be written with 0.

5.4.3.4 PFRWB FLASH configuration register 3

The Flash Configuration register 3 enables to control the error injections to the PFRWB safety mechanisms.

UR_FLASHCON3 Offset address: 000004_H

PFRWB FLASH configuration register 3 Application Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
ECCERRINJ	0	rw	**ECC Error Injection** Setting this bit enforces an error in the ECC error correction monitoring circuit. This can be used to check the alarm line to the SMU. 0_B No error injected, ECC monitor operates normally. 1_B An error is injected in the output of the ECC monitoring logic to mimic an alarm.
EDCERRINJ	1	rw	**EDC Error Injection** Setting this bit enforces an error in the EDC checker logic. This can be used to check the alarm line to the SMU. 0_B No error injected, EDC checker logic operates normally. 1_B An error is injected in the output of the EDC checker logic to mimic an alarm.
SBABERRINJ	2	rw	**SBAB Error Injection** Setting this bit generates a SBAB full SMU alarm, and can be used to check the correct function of this alarm. 0_B No error injected, SBAB logic operates normally. 1_B Inject a SBAB full alarm to the SMU.
DBABERRINJ	3	rw	**DBAB Error Injection** Setting this bit generates a DBAB full SMU alarm, and can be used to check the correct function of this alarm 0_B No error injected, DBAB logic operates normally. 1_B Inject a DBAB full alarm to the SMU.
UBABERRINJ	4	rw	**UBAB Error Injection** Setting this bit generates an UBAB full SMU alarm, and can be used to check the correct function of this alarm. 0_B No error injected, UBAB logic operates normally. 1_B Inject an UBAB full alarm to the SMU.
SBERERRINJ	5	rw	**ECC Single Bit error error injection** Setting this bit generates SBER SMU alarm, and can be used to check the correct function of this alarm 0_B No error injected, SBER logic operates normally. 1_B Inject a SBER alarm to the SMU.

(table continues...)

(continued)

Field	Bits	Type	Description
DBERERRINJ	6	rw	**ECC Double Bit error error injection** Setting this bit generates DBER SMU alarm, and can be used to check the correct function of this alarm 0_B No error injected, DBER logic operates normally. 1_B Inject a DBER alarm to the SMU.
MISRERRRINJ	7	rw	**MISR error injection** Setting this bit generates a MISR SMU alarm, and can be used to check the correct function of this alarm 0_B No error injected, MISR logic operates normally. 1_B Inject a MISR alarm to the SMU.
0	31:8	r	**Reserved** Read as 0; shall be written with 0.

5.4.3.5 ECC read register

The ECC Read register stores the ECC checksum corresponding to the last read on the local PFLASH bank.

UR_ECCR Offset address: 002000_H

ECC read register System Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								0		RCODE					
r								r		rh					

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RCODE															
rh															

Field	Bits	Type	Description
RCODE	21:0	rh	**Error Correction Code from the last read.** These registers contain the ECC checksum corresponding to the last local PFLASH bank read access.
0	23:22, 31:24	r	**Reserved** Read as 0; shall be written with 0.

5.4.3.6 ECC status register

The ECC status register stores the errors found by the ECC while reading the local PFLASH bank.

UR_ECCS Offset address: 002004_H

ECC status register System Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0							AANER	0		AALL1	AALL0	AUCER	0	ADBER	ASBER
r							rh	r		rh	rh	rh	r	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0							ANER	0		ALL1	ALL0	UCER	0	DBER	SBER
r							rh	r		rh	rh	rh	r	rh	rh

Field	Bits	Type	Description
SBER	0	rh	**Single Bit ECC Error** This register reports a single bit error found in the last local PFLASH bank read access. 0_B No single bit error was detected by the ECC in the last local PFLASH bank read. 1_B A single bit error was detected by the ECC in the last local PFLASH bank read.
DBER	1	rh	**Double Bit ECC Error** This register reports a double bit error found in the last local PFLASH bank read access. 0_B No double bit error was detected by the ECC in the last local PFLASH bank read. 1_B A double bit error was detected by the ECC in the last local PFLASH bank read.
UCER	3	rh	**Uncorrectable ECC Error** This register reports an uncorrectable error found in the last local PFLASH bank read access. 0_B No uncorrectable error was detected by the ECC in the last local PFLASH bank read. 1_B An uncorrectable error was detected by the ECC in the last local PFLASH bank read.
ALL0	4	rh	**All 0 bit error** This register reports that all bits are zeros in the last local PFLASH bank read access. 0_B Not all zeros detected in the last local PFLASH bank read. 1_B All zeros detected in the last local PFLASH bank read.
ALL1	5	rh	**All 1 bit error** This register reports that all bits are ones in the last local PFLASH bank read access. 0_B Not all ones detected in the last local PFLASH bank read. 1_B All ones detected in the last local PFLASH bank read.

(table continues...)

(continued)

Field	Bits	Type	Description
ANER	8	rh	**Any ECC error** This register reports an error found in the last local PFLASH bank read access. 0_B No error was detected by the ECC in the last local PFLASH bank read. 1_B An error was detected by the ECC in the last local PFLASH bank read.
ASBER	16	rh	**Accumulated Single Bit ECC Error** This register reports that at least one single bit error was found in the local PFLASH bank read accesses since the last reset or clear. 0_B No single bit error was detected by the ECC in the local PFLASH bank read accesses. 1_B A single bit error was detected by the ECC in the local PFLASH bank read accesses.
ADBER	17	rh	**Accumulated Double Bit ECC Error** This register reports that at least one double bit error was found in the local PFLASH bank read accesses since the last reset or clear. 0_B No double bit error was detected by the ECC in the local PFLASH bank read accesses. 1_B A double bit error was detected by the ECC in the local PFLASH bank read accesses.
AUCER	19	rh	**Accumulated Uncorrectable ECC Error** This register reports that at least one uncorrectable error was found in the local PFLASH bank read accesses since the last reset or clear. 0_B No uncorrectable error was detected by the ECC in the local PFLASH bank read accesses. 1_B A uncorrectable error was detected by the ECC in the local PFLASH bank read accesses.
AALL0	20	rh	**Acumulated all 0 bit error** This register reports that at least one all zeros was found in the local PFLASH bank read accesses since the last reset or clear. 0_B Not all zeros detected in the local PFLASH bank read accesses. 1_B All zeros detected in the local PFLASH bank read accesses.
AALL1	21	rh	**Accumulated all 1 bit error** This register reports that at least one all ones was found in the local PFLASH bank read accesses since the last reset or clear. 0_B Not all ones detected in the local PFLASH bank read accesses. 1_B All ones detected in the local PFLASH bank read accesses.
AANER	24	rh	**Accumulated Any ECC error** This register reports that at least one error was found in the local PFLASH bank read accesses since the last reset or clea 0_B No error was detected by the ECC in the local PFLASH bank read accesses. 1_B An error was detected by the ECC in the local PFLASH bank read accesses.

(table continues...)

(continued)

Field	Bits	Type	Description
0	2, 7:6, 15:9, 18, 23:22, 31:25	r	**Reserved** Read as 0; shall be written with 0.

5.4.3.7 Single bit error address buffer x

These registers records the single bit error detected by the ECC in the local PFLASH bank reads. Once all entries are valid, the buffer is considered full and an alarm is generated.

UR_SBABx (x=0-16) Offset address: 002080_H+x*4

Single bit error address buffer x System Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
VLD		0		ECF			0					ADDR			
rh		r		rh			r					rh			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR											0				
rh											r				

Field	Bits	Type	Description
ADDR	20:5	rh	**Address** This register stores the address of the detected local PFLASH bank read error.
ECF	27	rh	**Erase Counter Flag** This register stores the information that the error was detected in the erase counter. 0_B The error was found in the PFLASH 1_B The error was found in the erase counter.
VLD	31	rh	**Valid** This register stores the information that this 32 bit entry of the SBAB is valid. This bitfield is protected by a SFF. 0_B Entry is not valid. 1_B A valid entry was recorded.
0	4:0, 26:21, 30:28	r	**Reserved** Read as 0; shall be written with 0.

5.4.3.8 Double bit error address buffer x

These registers records the double bit error detected by the ECC in the local PFLASH bank reads. Once all entries are valid, the buffer is considered full and an alarm is generated.

UR_DBABx (x=0-1) Offset address: 002100_H+x*4

Double bit error address buffer x System Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
VLD	0			ECF	0						ADDR				
rh	r			rh	r						rh				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR											0				
rh											r				

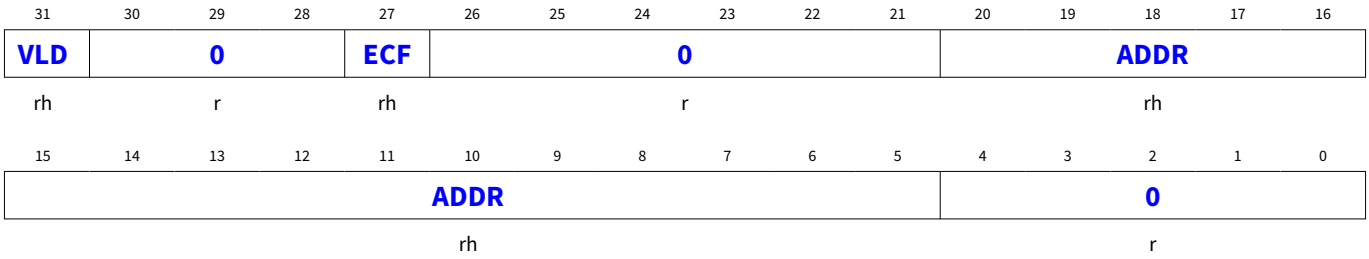
Field	Bits	Type	Description
ADDR	20:5	rh	**Address** This register stores the address of the detected local PFLASH bank read error.
ECF	27	rh	**Erase Counter Flag** This register stores the information that the error was detected in the erase counter. 0_B The error was found in the PFLASH 1_B The error was found in the erase counter.
VLD	31	rh	**Valid** This register stores the information that this 32 bit entry of the SBAB is valid. This bitfield is protected by a SFF. 0_B Entry is not valid. 1_B A valid entry was recorded.
0	4:0, 26:21, 30:28	r	**Reserved** Read as 0; shall be written with 0.

5.4.3.9 Uncorrectable error address buffer x

These registers records the uncorrectable bit error detected by the ECC in the local PFLASH bank reads. Once all entries are valid, the buffer is considered full and an alarm is generated.

UR_UBAB0 Offset address: 002180_H

Uncorrectable error address buffer x System Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
ADDR	20:5	rh	**Address** This register stores the address of the detected local PFLASH bank read error.
ECF	27	rh	**Erase Counter Flag** This register stores the information that the error was detected in the erase counter. 0_B The error was found in the PFLASH 1_B The error was found in the erase counter.
VLD	31	rh	**Valid** This register stores the information that this 32 bit entry of the SBAB is valid. This bitfield is protected by a SFF. 0_B Entry is not valid. 1_B A valid entry was recorded.
0	4:0, 26:21, 30:28	r	**Reserved** Read as 0; shall be written with 0.

5.5 Debug information

This section is not applicable for the module being described.

5.6 References

This section is not applicable for the module being described.

5.7 NVM revision history

Initial release of the chapter.

5.8 TC4Dx NVM information

5.8.1 TC4Dx NVM configuration

The following tables details the NVM Bank configuration of the TC4Dx product.

Table 552 TC4Dx Program Flash Banks

CPU	PFLASH Bank	Size	System Address
CPU0	0A	2 Mbyte	NVM.PFLASH00 (Program Flash)
	0B	2 Mbyte	NVM.PFLASH01 (Program Flash)
CPU1	1A	2 Mbyte	NVM.PFLASH10 (Program Flash)
	1B	2 Mbyte	NVM.PFLASH11 (Program Flash)
CPU2	2A	1 Mbyte	NVM.PFLASH20 (Program Flash)
	2B	1 Mbyte	NVM.PFLASH21 (Program Flash)
CPU3	3A	2 Mbyte	NVM.PFLASH30 (Program Flash)
	3B	2 Mbyte	NVM.PFLASH31 (Program Flash)
CPU4	4A	2 Mbyte	NVM.PFLASH40 (Program Flash)
	4B	2 Mbyte	NVM.PFLASH41 (Program Flash)
CPU5	5A	1 Mbyte	NVM.PFLASH50 (Program Flash)
	5B	1 Mbyte	NVM.PFLASH51 (Program Flash)
CPUcs	CSRM	1 Mbyte	NVM.PFLASHcs (Program Flash)

***Note**: for the actual memory address please refer to the system address chapter. Each Program Flash Bank has also a separate erase counter (16 Kbyte) which is accessible at the system address NVM.PFLASHxx (Erase Counter).*

***Note**: PFLASH Banks for the CPU2 and for the CPU5 have a size of 1 MB. Therefore less write protection registers are necessary. The register description chapters and the UCB description always contain the maximum number of possible write protection registers (which is for a 2 MB Bank). Therefore for the PFLASH Banks for CPU2 and CPU5 only the write protection registers for up to 1 MB are available and the remaining UCB space is "do not care" and will not be recovered.*

Table 553 TC4Dx Data Flash Banks

DFLASH Bank	Memory Region	Size	System Address
DFLASH(0) - HOST	EEPROM Emulation HOST	1024 Kbyte	NVM.EEPROM0
	User Configuration Blocks UCB(0)	80 Kbyte	NVM.UCB0
	Configuration Sector(0)	80 Kbyte	NVM.CFS0 (not accessible in user mode)

(table continues...)

Table 553 (continued) TC4Dx Data Flash Banks

DFLASH Bank	Memory Region	Size	System Address
DFLASH(1) - CSRM	EEPROM Emulation CSRM	128 Kbyte	NVM.EEPROM1
	User Configuration Blocks UCB(1)	52 Kbyte	NVM.UCB1
	Configuration Sector(1)	44 Kbyte	NVM.CFS1 (not accessible in user mode)

***Note**: for the actual memory address please refer to the system address chapter.*

5.8.2 TC4Dx NVM features

There are no deviations from the generic specification.

5.8.3 TC4Dx NVM functional description

The number of APU for PFLASH (HOST) banks is 12.

5.8.4 TC4Dx NVM registers

PFLASH Banks for the CPU2 and for the CPU5 have a size of 1 MB. Therefore less write protection registers are necessary. The register description chapters and the UCB description always contain the maximum number of possible write protection registers (which is for a 2 MB Bank). Therefore for the PFLASH Banks for CPU2 and CPU5 only the write protection registers for up to 1 MB are available and the UCB memory for the upper 1 MB space is "do not care" and is not recovered during the startup phase. This concerns the following registers for the HOST and the CSRM command interfaces:

- GP_HOST_PFPROCONWPpXn (p=2 or p=5 (CPU2,5), X=A or B, n = 2,3 for upper 1 MB not existing))
- GP_HOST_PFPROCONPWPpXn (p=2 or p=5 (CPU2,5), X=A or B, n = 2,3 for upper 1 MB not existing))
- GP_HOST_PFPROCONPWOPpXn ((p=2 or p=5 (CPU2,5), X=A or B, n = 2,3 for upper 1 MB not existing))
- GP_HOST_PFECPRIOpXn ((p=2 or p=5 (CPU2,5), X=A or B, n = 2,3 for upper 1 MB not existing))

5.8.4.1 Register address space - DMU

Table 554 Registers address space - DMU

Module	Base address	End address	Note
DMU	$F8040000_H$	$F807FFFF_H$	

5.8.4.2 Register overview - access mode glossary

Table 555 Register overview - access mode glossary

Keyword	Description
CSCI	Access protection using PROT register DMU_PROT_PROTCSCI .
CSDF	Access protection using PROT register DMU_PROT_PROTCSDF .
CSPF	Access protection using PROT register DMU_PROT_PROTCSPF .
CSRDC	Access protection using PROT register DMU_PROT_PROTCSRDC .
HCI	Access protection using PROT register DMU_PROT_PROTHCI .
HDF	Access protection using PROT register DMU_PROT_PROTHDF .
HPF	Access protection using PROT register DMU_PROT_PROTHPF .
HRDC	Access protection using PROT register DMU_PROT_PROTHRDC .
APU-PCSCI	Protection group consisting of registers DMU_PROT_ACCENCSCI_WRA , DMU_PROT_ACCENCSCI_WRB , DMU_PROT_ACCENCSCI_RDA , DMU_PROT_ACCENCSCI_RDB , DMU_PROT_ACCENCSCI_VM , DMU_PROT_ACCENCSCI_PRS .
PCSCI	Access protection using *APU-PCSCI* registers.
APU-PCSDFUCB	Protection group consisting of registers DMU_PROT_ACCENCSDFUCB_WRA , DMU_PROT_ACCENCSDFUCB_WRB , DMU_PROT_ACCENCSDFUCB_RDA , DMU_PROT_ACCENCSDFUCB_RDB , DMU_PROT_ACCENCSDFUCB_VM , DMU_PROT_ACCENCSDFUCB_PRS .
PCSDFUCB	Access protection using *APU-PCSDFUCB* registers.

(table continues...)

Table 555 (continued) Register overview - access mode glossary

Keyword	Description
APU-PCSDFd (d=0-3)	Protection group consisting of registers DMU_PROT_ACCENCSDFd_WRA , DMU_PROT_ACCENCSDFd_WRB , DMU_PROT_ACCENCSDFd_RDA , DMU_PROT_ACCENCSDFd_RDB , DMU_PROT_ACCENCSDFd_VM , DMU_PROT_ACCENCSDFd_PRS , DMU_PROT_ACCENCSDFd_RGNLA , DMU_PROT_ACCENCSDFd_RGNUA .
PCSDFd	Access protection using *APU-PCSDFd* registers.
APU-PCSPFp (p=0-3)	Protection group consisting of registers DMU_PROT_ACCENCSPFp_WRA , DMU_PROT_ACCENCSPFp_WRB , DMU_PROT_ACCENCSPFp_VM , DMU_PROT_ACCENCSPFp_PRS , DMU_PROT_ACCENCSPFp_RGNLA , DMU_PROT_ACCENCSPFp_RGNUA .
PCSPFp	Access protection using *APU-PCSPFp* registers.
APU-PCSRDC	Protection group consisting of registers DMU_PROT_ACCENCSRDC_WRA , DMU_PROT_ACCENCSRDC_WRB , DMU_PROT_ACCENCSRDC_RDA , DMU_PROT_ACCENCSRDC_RDB , DMU_PROT_ACCENCSRDC_VM , DMU_PROT_ACCENCSRDC_PRS .
PCSRDC	Access protection using *APU-PCSRDC* registers.
APU-PHCI	Protection group consisting of registers DMU_PROT_ACCENHCI_WRA , DMU_PROT_ACCENHCI_WRB , DMU_PROT_ACCENHCI_RDA , DMU_PROT_ACCENHCI_RDB , DMU_PROT_ACCENHCI_VM , DMU_PROT_ACCENHCI_PRS .
PHCI	Access protection using *APU-PHCI* registers.
APU-PHDFUCB	Protection group consisting of registers DMU_PROT_ACCENHDFUCB_WRA , DMU_PROT_ACCENHDFUCB_WRB , DMU_PROT_ACCENHDFUCB_RDA , DMU_PROT_ACCENHDFUCB_RDB , DMU_PROT_ACCENHDFUCB_VM , DMU_PROT_ACCENHDFUCB_PRS .
PHDFUCB	Access protection using *APU-PHDFUCB* registers.
APU-PHDFd (d=0-7)	Protection group consisting of registers DMU_PROT_ACCENHDFd_WRA , DMU_PROT_ACCENHDFd_WRB , DMU_PROT_ACCENHDFd_RDA , DMU_PROT_ACCENHDFd_RDB , DMU_PROT_ACCENHDFd_VM , DMU_PROT_ACCENHDFd_PRS , DMU_PROT_ACCENHDFd_RGNLA , DMU_PROT_ACCENHDFd_RGNUA .
PHDFd	Access protection using *APU-PHDFd* registers.
APU-PHPFp (p=0-11)	Protection group consisting of registers DMU_PROT_ACCENHPFp_WRA , DMU_PROT_ACCENHPFp_WRB , DMU_PROT_ACCENHPFp_VM , DMU_PROT_ACCENHPFp_PRS , DMU_PROT_ACCENHPFp_RGNLA , DMU_PROT_ACCENHPFp_RGNUA .
PHPFp	Access protection using *APU-PHPFp* registers.
APU-PHRDC	Protection group consisting of registers DMU_PROT_ACCENHRDC_WRA , DMU_PROT_ACCENHRDC_WRB , DMU_PROT_ACCENHRDC_RDA , DMU_PROT_ACCENHRDC_RDB , DMU_PROT_ACCENHRDC_VM , DMU_PROT_ACCENHRDC_PRS .
PHRDC	Access protection using *APU-PHRDC* registers.

(table continues...)

Table 555 (continued) Register overview - access mode glossary

Keyword	Description
32	Access only when using 32-bit width.
BE	Always returns a Bus Error.
U	No access restrictions.
STP	Access only during start-up, before start-up protection is set (SMM_STCON.ST_PROT=1).
M	Indicates a module specific access condition. Refer to the register description for details of the specific access condition.

5.8.4.3 Registers overview - DMU (ascending offset address)

Table 556 Registers overview - DMU (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
DMU_HCI_STATUS	HOST status register	0004_H	PHCI	BE	Application Reset	813
DMU_HCI_BCONTROL	HOST boot control register	0008_H	PHCI	PHCI	816	816
DMU_HCI_OCONTROL	HOST operation control register	$000C_H$	PHCI	PHCI	Application Reset	817
DMU_HCI_ERR	HOST command interface error register	0010_H	PHCI	BE	818	818
DMU_HCI_CLRERR	HOST command interface clear error register	0014_H	PHCI, 32	PHCI, 32	Application Reset	821
DMU_HCI_INTEN	HOST command interface interrupt enable register	0018_H	PHCI	PHCI	Application Reset	822
DMU_HCI_SLEEP	HOST sleep register	$001C_H$	PHCI, 32	PHCI, 32	Application Reset	822
DMU_CSCI_STATUS	CSRM status register	0084_H	PCSCI	BE	Application Reset	823
DMU_CSCI_BCONTROL	CSRM boot control register	0088_H	PCSCI	PCSCI	826	826
DMU_CSCI_OCONTROL	CSRM operation control register	$008C_H$	PCSCI	PCSCI	Application Reset	827
DMU_CSCI_ERR	CSRM command interface error register	0090_H	PCSCI	BE	828	828
DMU_CSCI_CLRERR	CSRM command interface clear error register	0094_H	PCSCI, 32	PCSCI, 32	Application Reset	831
DMU_CSCI_INTEN	CSRM command interface interrupt enable register	0098_H	PCSCI	PCSCI	Application Reset	832
DMU_CSCI_SLEEP	CSRM sleep register	$009C_H$	PCSCI, 32	PCSCI, 32	Application Reset	832

(table continues...)

Table 556 (continued) Registers overview - DMU (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
DMU_GP_HOST_DFWAIT	HOST DFLASH wait cycle register	0100_H	PHRDC	PHRDC	834	834
DMU_GP_HOST_DFMARGIN	HOST DFLASH margin control register	0104_H	PHRDC	PHRDC	Application Reset	834
DMU_GP_HOST_DFECCR	HOST DFLASH ECC read register	0108_H	PHRDC, 32	BE	Application Reset	835
DMU_GP_HOST_DFECCS	HOST DFLASH ECC status register	$010C_H$	PHRDC, 32	BE	836	836
DMU_GP_HOST_DFECCC	HOST DFLASH ECC control register	0110_H	PHRDC	PHRDC	Application Reset	837
DMU_GP_HOST_DFECCW	HOST DFLASH ECC write register	0114_H	PHCI	PHCI	Application Reset	838
DMU_GP_HOST_DFPROCON	HOST DFLASH protection register	0118_H	U	32	839	839
DMU_GP_HOST_PFWAIT	HOST PFLASH wait cycle register	$011C_H$	PHRDC	PHRDC	840	840
DMU_GP_HOST_PFERRINJ	HOST PFLASH error injection register	0120_H	PHRDC	PHRDC	Application Reset	840
DMU_GP_HOST_PFECCW	HOST PFLASH ECC write register	0124_H	PHCI	PHCI	Application Reset	841
DMU_GP_HOST_PFPROCON	HOST PFLASH protection register	0128_H	U	BE	842	842
DMU_GP_HOST_PFPROCONWPpA0 (p=0-5)	HOST PFLASH bank PpA write protection configuration 0	$012C_H$+p *80_H	U	BE	842	842
DMU_GP_HOST_PFPROCONWPpA1 (p=0-5)	HOST PFLASH bank PpA write protection configuration 1	0130_H+p *80_H	U	BE	843	843
DMU_GP_HOST_PFPROCONWPpA2 (p=0-5)	HOST PFLASH bank PpA write protection configuration 2	0134_H+p *80_H	U	BE	844	844
DMU_GP_HOST_PFPROCONWPpA3 (p=0-5)	HOST PFLASH bank PpA write protection configuration 3	0138_H+p *80_H	U	BE	844	844
DMU_GP_HOST_PFPROCONPWPpA0 (p=0-5)	HOST PFLASH bank PpA permanent write protection configuration 0	$013C_H$+p *80_H	U	BE	845	845

(table continues...)

Table 556 (continued) Registers overview - DMU (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
DMU_GP_HOST_PFPROCONPWPpA1 (p=0-5)	HOST PFLASH bank PpA permanent write protection configuration 1	0140_H+p*80_H	U	BE	846	846
DMU_GP_HOST_PFPROCONPWPpA2 (p=0-5)	HOST PFLASH bank PpA permanent write protection configuration 2	0144_H+p*80_H	U	BE	846	846
DMU_GP_HOST_PFPROCONPWPpA3 (p=0-5)	HOST PFLASH bank PpA permanent write protection configuration 3	0148_H+p*80_H	U	BE	847	847
DMU_GP_HOST_PFPROCONPWOPpA0 (p=0-5)	HOST PFLASH bank PpA permanent write once protection configuration 0	$014C_H$+p*80_H	U	BE	848	848
DMU_GP_HOST_PFPROCONPWOPpA1 (p=0-5)	HOST PFLASH bank PpA permanent write once protection configuration 1	0150_H+p*80_H	U	BE	848	848
DMU_GP_HOST_PFPROCONPWOPpA2 (p=0-5)	HOST PFLASH bank PpA permanent write once protection configuration 2	0154_H+p*80_H	U	BE	849	849
DMU_GP_HOST_PFPROCONPWOPpA3 (p=0-5)	HOST PFLASH bank PpA permanent write once protection configuration 3	0158_H+p*80_H	U	BE	850	850
DMU_GP_HOST_PFECPRIOpA0 (p=0-5)	HOST PFLASH bank PpA erase counter priority configuration 0	$015C_H$+p*80_H	U	BE	850	850
DMU_GP_HOST_PFECPRIOpA1 (p=0-5)	HOST PFLASH bank PpA erase counter priority configuration 1	0160_H+p*80_H	U	BE	851	851
DMU_GP_HOST_PFECPRIOpA2 (p=0-5)	HOST PFLASH bank PpA erase counter priority configuration 2	0164_H+p*80_H	U	BE	852	852
DMU_GP_HOST_PFECPRIOpA3 (p=0-5)	HOST PFLASH bank PpA erase counter priority configuration 3	0168_H+p*80_H	U	BE	852	852

(table continues...)

Table 556 (continued) Registers overview - DMU (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
DMU_GP_HOST_PFPROCONWPpB0 (p=0-5)	HOST PFLASH bank PpB write protection configuration 0	$016C_H$+p*80_H	U	BE	853	853
DMU_GP_HOST_PFPROCONWPpB1 (p=0-5)	HOST PFLASH bank PpB write protection configuration 1	0170_H+p*80_H	U	BE	854	854
DMU_GP_HOST_PFPROCONWPpB2 (p=0-5)	HOST PFLASH bank PpB write protection configuration 2	0174_H+p*80_H	U	BE	854	854
DMU_GP_HOST_PFPROCONWPpB3 (p=0-5)	HOST PFLASH bank PpB write protection configuration 3	0178_H+p*80_H	U	BE	855	855
DMU_GP_HOST_PFPROCONPWPpB0 (p=0-5)	HOST PFLASH bank PpB permanent write protection configuration 0	$017C_H$+p*80_H	U	BE	856	856
DMU_GP_HOST_PFPROCONPWPpB1 (p=0-5)	HOST PFLASH bank PpB permanent write protection configuration 1	0180_H+p*80_H	U	BE	856	856
DMU_GP_HOST_PFPROCONPWPpB2 (p=0-5)	HOST PFLASH bank PpB permanent write protection configuration 2	0184_H+p*80_H	U	BE	857	857
DMU_GP_HOST_PFPROCONPWPpB3 (p=0-5)	HOST PFLASH bank PpB permanent write protection configuration 3	0188_H+p*80_H	U	BE	858	858
DMU_GP_HOST_PFPROCONPWOPpB0 (p=0-5)	HOST PFLASH bank PpB permanent write once protection configuration 0	$018C_H$+p*80_H	U	BE	858	858
DMU_GP_HOST_PFPROCONPWOPpB1 (p=0-5)	HOST PFLASH bank PpB permanent write once protection configuration 1	0190_H+p*80_H	U	BE	859	859
DMU_GP_HOST_PFPROCONPWOPpB2 (p=0-5)	HOST PFLASH bank PpB permanent write once protection configuration 2	0194_H+p*80_H	U	BE	860	860

(table continues...)

Table 556 (continued) Registers overview - DMU (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
DMU_GP_HOST_PFPROCONPWOPpB3 (p=0-5)	HOST PFLASH bank PpB permanent write once protection configuration 3	0198_H+p*80_H	U	BE	860	860
DMU_GP_HOST_PFECPRIOpB0 (p=0-5)	HOST PFLASH bank PpB erase counter priority configuration 0	$019C_H$+p*80_H	U	BE	861	861
DMU_GP_HOST_PFECPRIOpB1 (p=0-5)	HOST PFLASH bank PpB erase counter priority configuration 1	$01A0_H$+p*80_H	U	BE	862	862
DMU_GP_HOST_PFECPRIOpB2 (p=0-5)	HOST PFLASH bank PpB erase counter priority configuration 2	$01A4_H$+p*80_H	U	BE	862	862
DMU_GP_HOST_PFECPRIOpB3 (p=0-5)	HOST PFLASH bank PpB erase counter priority configuration 3	$01A8_H$+p*80_H	U	BE	863	863
DMU_GP_HOST_PROTECT	HOST UCB password protection register	$052C_H$	U	BE	Application Reset	864
DMU_GP_HOST_CONFIRMA	HOST UCB confirmation codes register A	0530_H	U	BE	866	866
DMU_GP_HOST_CONFIRMB	HOST UCB confirmation codes register B	0534_H	U	BE	868	868
DMU_GP_HOST_CONFIRMC	HOST UCB confirmation codes register C	0538_H	U	BE	870	870
DMU_GP_CSRM_DFWAIT	CSRM DFLASH wait cycle register	0900_H	PCSRDC	PCSRDC	873	873
DMU_GP_CSRM_DFMARGIN	CSRM DFLASH margin control register	0904_H	PCSRDC	PCSRDC	Application Reset	873
DMU_GP_CSRM_DFECCR	CSRM DFLASH ECC read register	0908_H	PCSRDC, 32	BE	Application Reset	874
DMU_GP_CSRM_DFECCS	CSRM DFLASH ECC status register	$090C_H$	PCSRDC, 32	BE	875	875
DMU_GP_CSRM_DFECCC	CSRM DFLASH ECC control register	0910_H	PCSRDC	PCSRDC	Application Reset	877
DMU_GP_CSRM_DFECCW	CSRM DFLASH ECC write register	0914_H	PCSCI	PCSCI	Application Reset	878
DMU_GP_CSRM_DFPROCON	CSRM DFLASH protection register	0918_H	U	BE	878	878
DMU_GP_CSRM_PFWAIT	CSRM PFLASH wait cycle register	0920_H	PCSRDC	PCSRDC	879	879
DMU_GP_CSRM_PFERRINJ	CSRM PFLASH error injection register	0924_H	PCSRDC	PCSRDC	Application Reset	880

(table continues...)

Table 556 (continued) Registers overview - DMU (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
DMU_GP_CSRM_PFECCW	CSRM PFLASH ECC write register	0928_H	PCSCI	PCSCI	Application Reset	880
DMU_GP_CSRM_PFPROCONWP0	CSRM PFLASH bank write protection configuration 0	$092C_H$	U	BE	881	881
DMU_GP_CSRM_PFPROCONWP1	CSRM PFLASH bank write protection configuration 1	0930_H	U	BE	881	881
DMU_GP_CSRM_PFPROCONWP2	CSRM PFLASH bank write protection configuration 2	0934_H	U	BE	882	882
DMU_GP_CSRM_PFPROCONWP3	CSRM PFLASH bank write protection configuration 3	0938_H	U	BE	883	883
DMU_GP_CSRM_PFPROCONPWP0	CSRM PFLASH bank permanent write protection configuration 0	$093C_H$	U	BE	883	883
DMU_GP_CSRM_PFPROCONPWP1	CSRM PFLASH bank permanent write protection configuration 1	0940_H	U	BE	884	884
DMU_GP_CSRM_PFPROCONPWP2	CSRM PFLASH bank permanent write protection configuration 2	0944_H	U	BE	885	885
DMU_GP_CSRM_PFPROCONPWP3	CSRM PFLASH bank permanent write protection configuration 3	0948_H	U	BE	885	885
DMU_GP_CSRM_PFPROCONPWOP0	CSRM PFLASH bank permanent write once protection configuration 0	$094C_H$	U	BE	886	886
DMU_GP_CSRM_PFPROCONPWOP1	CSRM PFLASH bank permanent write once protection configuration 1	0950_H	U	BE	887	887
DMU_GP_CSRM_PFPROCONPWOP2	CSRM PFLASH bank permanent write once protection configuration 2	0954_H	U	BE	887	887
DMU_GP_CSRM_PFPROCONPWOP3	CSRM PFLASH bank permanent write once protection configuration 3	0958_H	U	BE	888	888
DMU_GP_CSRM_PFECPRIO0	CSRM PFLASH bank erase counter priority configuration 0	$095C_H$	U	BE	889	889
DMU_GP_CSRM_PFECPRIO1	CSRM PFLASH bank erase counter priority configuration 1	0960_H	U	BE	889	889
DMU_GP_CSRM_PFECPRIO2	CSRM PFLASH bank erase counter priority configuration 2	0964_H	U	BE	890	890

(table continues...)

Table 556 (continued) Registers overview - DMU (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
DMU_GP_CSRM_PFECPRIO3	CSRM PFLASH bank erase counter priority configuration 3	0968_H	U	BE	891	891
DMU_GP_CSRM_PROTECT	CSRM UCB password protection register	$096C_H$	U	BE	Application Reset	891
DMU_GP_CSRM_CONFIRMA	CSRM UCB confirmation codes register A	0970_H	U	BE	893	893
DMU_GP_CSRM_CONFIRMB	CSRM UCB confirmation codes register B	0974_H	U	BE	895	895
DMU_GP_CSRM_CONFIRMC	CSRM UCB confirmation codes register C	0978_H	U	BE	897	897
DMU_GP_CSRM_DBGCFG	CSRM debug configuration	$097C_H$	U	BE	899	899
DMU_GP_CSRM_DBGMODE	CSRM debug mode	0980_H	U	BE	900	900
DMU_GP_BKALLOC	Bank allocation register	$0A00_H$	U	BE	Application Reset	901
DMU_GP_PCONTROL	Power control register	$0A14_H$	U	BE	902	902
DMU_PROT_PROTHCI	PROT register for HOST command interface and HCI registers	$0B00_H$	U	SV, 32	Application Reset	902
DMU_PROT_ACCENHCI_WRA	HOST command interface and HCI registers, write access enable register A	$0B08_H$	U	HCI, 32	Application Reset	904
DMU_PROT_ACCENHCI_WRB	HOST command interface and HCI registers, write access enable register B	$0B0C_H$	U	HCI, 32	Application Reset	904
DMU_PROT_ACCENHCI_RDA	HOST command interface and HCI registers, read access enable register A	$0B10_H$	U	HCI, 32	Application Reset	905
DMU_PROT_ACCENHCI_RDB	HOST command interface and HCI registers, read access enable register B	$0B14_H$	U	HCI, 32	Application Reset	905
DMU_PROT_ACCENHCI_VM	HOST command interface and HCI registers, VM access enable register	$0B18_H$	U	HCI, 32	Application Reset	906
DMU_PROT_ACCENHCI_PRS	HOST command interface and HCI registers, PRS access enable register	$0B1C_H$	U	HCI, 32	Application Reset	907
DMU_PROT_PROTCSCI	PROT register for CSRM command interface and CSCI registers	$0B28_H$	U	SV, 32	Application Reset	907

(table continues...)

Table 556 (continued) Registers overview - DMU (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
DMU_PROT_ACCENCSCI_WRA	CSRM command interface and CSCI registers, write access enable register A	$0B30_H$	U	CSCI, 32	Application Reset	909
DMU_PROT_ACCENCSCI_WRB	CSRM command interface and CSCI registers, write access enable register B	$0B34_H$	U	CSCI, 32	Application Reset	910
DMU_PROT_ACCENCSCI_RDA	CSRM command interface and CSCI registers, read access enable register A	$0B38_H$	U	CSCI, 32	Application Reset	910
DMU_PROT_ACCENCSCI_RDB	CSRM command interface and CSCI registers, read access enable register B	$0B3C_H$	U	CSCI, 32	Application Reset	911
DMU_PROT_ACCENCSCI_VM	CSRM command interface and CSCI registers, VM access enable register	$0B40_H$	U	CSCI, 32	Application Reset	911
DMU_PROT_ACCENCSCI_PRS	CSRM command interface and CSCI registers, PRS access enable register	$0B44_H$	U	CSCI, 32	Application Reset	912
DMU_PROT_PROTHRDC	PROT register for HOST read configurations registers	$0B50_H$	U	SV, 32	Application Reset	913
DMU_PROT_ACCENHRDC_WRA	HOST read configurations registers, write access enable register A	$0B58_H$	U	HRDC, 32	Application Reset	914
DMU_PROT_ACCENHRDC_WRB	HOST read configurations registers, write access enable register B	$0B5C_H$	U	HRDC, 32	Application Reset	915
DMU_PROT_ACCENHRDC_RDA	HOST read configurations registers, read access enable register A	$0B60_H$	U	HRDC, 32	Application Reset	915
DMU_PROT_ACCENHRDC_RDB	HOST read configurations registers, read access enable register B	$0B64_H$	U	HRDC, 32	Application Reset	916
DMU_PROT_ACCENHRDC_VM	HOST read configurations registers, VM access enable register	$0B68_H$	U	HRDC, 32	Application Reset	916
DMU_PROT_ACCENHRDC_PRS	HOST read configurations registers, PRS access enable register	$0B6C_H$	U	HRDC, 32	Application Reset	917
DMU_PROT_PROTCSRDC	PROT register for CSRM read configurations registers	$0B78_H$	U	SV, 32	Application Reset	918
DMU_PROT_ACCENCSRDC_WRA	CSRM read configurations registers, write access enable register A	$0B80_H$	U	CSRDC, 32	Application Reset	920

(table continues...)

Table 556 (continued) Registers overview - DMU (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
DMU_PROT_ACCENCSRDC_WRB	CSRM read configurations registers, write access enable register B	$0B84_H$	U	CSRDC, 32	Application Reset	920
DMU_PROT_ACCENCSRDC_RDA	CSRM read configurations registers, read access enable register A	$0B88_H$	U	CSRDC, 32	Application Reset	921
DMU_PROT_ACCENCSRDC_RDB	CSRM read configurations registers, read access enable register B	$0B8C_H$	U	CSRDC, 32	Application Reset	921
DMU_PROT_ACCENCSRDC_VM	CSRM read configurations registers, VM access enable register	$0B90_H$	U	CSRDC, 32	Application Reset	922
DMU_PROT_ACCENCSRDC_PRS	CSRM read configurations registers, PRS access enable register	$0B94_H$	U	CSRDC, 32	Application Reset	922
DMU_PROT_PROTHPF	PROT register for HOST PFLASH memory	$0BA0_H$	U	SV, 32	Application Reset	923
DMU_PROT_ACCENHPFp_WRA (p=0-11)	HOST PFLASH memory Pp, write access enable register A	$0BA8_H$+p*20_H	U	HPF, 32	Application Reset	925
DMU_PROT_ACCENHPFp_WRB (p=0-11)	HOST PFLASH memory Pp, write access enable register B	$0BAC_H$+p*20_H	U	HPF, 32	Application Reset	925
DMU_PROT_ACCENHPFp_VM (p=0-11)	HOST PFLASH memory Pp, VM access enable register	$0BB8_H$+p*20_H	U	HPF, 32	Application Reset	926
DMU_PROT_ACCENHPFp_PRS (p=0-11)	HOST PFLASH memory Pp, PRS access enable register	$0BBC_H$+p*20_H	U	HPF, 32	Application Reset	926
DMU_PROT_ACCENHPFp_RGNLA (p=0-11)	HOST PFLASH memory Pp, region lower address register	$0BC0_H$+p*20_H	U	HPF, 32	Application Reset	927
DMU_PROT_ACCENHPFp_RGNUA (p=0-11)	HOST PFLASH memory Pp, region upper address register	$0BC4_H$+p*20_H	U	HPF, 32	Application Reset	927
DMU_PROT_PROTHDF	PROT register for HOST DFLASH (EEPROM) and HOST UCB memory	$0DA8_H$	U	SV, 32	Application Reset	928
DMU_PROT_ACCENHDFd_WRA (d=0-7)	HOST DFLASH (EEPROM) memory Dd, write access enable register A	$0DB0_H$+d*20_H	U	HDF, 32	Application Reset	930

(table continues...)

Table 556 (continued) Registers overview - DMU (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
DMU_PROT_ACCENHDFd_WRB (d=0-7)	HOST DFLASH (EEPROM) memory Dd, write access enable register B	$0DB4_H$+d*20_H	U	HDF, 32	Application Reset	930
DMU_PROT_ACCENHDFd_RDA (d=0-7)	HOST DFLASH (EEPROM) memory Dd, read access enable register A	$0DB8_H$+d*20_H	U	HDF, 32	Application Reset	931
DMU_PROT_ACCENHDFd_RDB (d=0-7)	HOST DFLASH (EEPROM) memory Dd, read access enable register B	$0DBC_H$+d*20_H	U	HDF, 32	Application Reset	931
DMU_PROT_ACCENHDFd_VM (d=0-7)	HOST DFLASH (EEPROM) memory Dd, VM access enable register	$0DC0_H$+d*20_H	U	HDF, 32	Application Reset	932
DMU_PROT_ACCENHDFd_PRS (d=0-7)	HOST DFLASH (EEPROM) memory Dd, PRS access enable register	$0DC4_H$+d*20_H	U	HDF, 32	Application Reset	933
DMU_PROT_ACCENHDFd_RGNLA (d=0-7)	HOST DFLASH (EEPROM) memory Dd, region lower address register	$0DC8_H$+d*20_H	U	HDF, 32	Application Reset	933
DMU_PROT_ACCENHDFd_RGNUA (d=0-7)	HOST DFLASH (EEPROM) memory Dd, region upper address register	$0DCC_H$+d*20_H	U	HDF, 32	Application Reset	934
DMU_PROT_ACCENHDFUCB_WRA	HOST UCB memory, write access enable register A	$0EB0_H$	U	HDF, 32	Application Reset	934
DMU_PROT_ACCENHDFUCB_WRB	HOST UCB memory, write access enable register B	$0EB4_H$	U	HDF, 32	Application Reset	935
DMU_PROT_ACCENHDFUCB_RDA	HOST UCB memory, read access enable register A	$0EB8_H$	U	HDF, 32	Application Reset	935
DMU_PROT_ACCENHDFUCB_RDB	HOST UCB memory, read access enable register B	$0EBC_H$	U	HDF, 32	Application Reset	936
DMU_PROT_ACCENHDFUCB_VM	HOST UCB memory, VM access enable register	$0EC0_H$	U	HDF, 32	Application Reset	936
DMU_PROT_ACCENHDFUCB_PRS	HOST UCB memory, PRS access enable register	$0EC4_H$	U	HDF, 32	Application Reset	937
DMU_PROT_PROTCSPF	PROT register for CSRM PFLASH memory	$0ED0_H$	U	SV, 32	Application Reset	938
DMU_PROT_ACCENCSPFp_WRA (p=0-3)	CSRM PFLASH memory Pp, write access enable register A	$0ED8_H$+p*20_H	U	CSPF, 32	Application Reset	939
DMU_PROT_ACCENCSPFp_WRB (p=0-3)	CSRM PFLASH memory Pp, write access enable register B	$0EDC_H$+p*20_H	U	CSPF, 32	Application Reset	940

(table continues...)

Table 556 (continued) Registers overview - DMU (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
DMU_PROT_ACCENCSPFp_VM (p=0-3)	CSRM PFLASH memory Pp, VM access enable register	$0EE8_H$+p*20_H	U	CSPF, 32	Application Reset	940
DMU_PROT_ACCENCSPFp_PRS (p=0-3)	CSRM PFLASH memory Pp, PRS access enable register	$0EEC_H$+p*20_H	U	CSPF, 32	Application Reset	941
DMU_PROT_ACCENCSPFp_RGNLA (p=0-3)	CSRM PFLASH memory Pp, region lower address register	$0EF0_H$+p*20_H	U	CSPF, 32	Application Reset	942
DMU_PROT_ACCENCSPFp_RGNUA (p=0-3)	CSRM PFLASH memory Pp, region upper address register	$0EF4_H$+p*20_H	U	CSPF, 32	Application Reset	942
DMU_PROT_PROTCSDF	PROT register for CSRM DFLASH (EEPROM) and CSRM UCB memory	$0F58_H$	U	SV, 32	Application Reset	943
DMU_PROT_ACCENCSDFd_WRA (d=0-3)	CSRM DFLASH (EEPROM) memory Dd, write access enable register A	$0F60_H$+d*20_H	U	CSDF, 32	Application Reset	945
DMU_PROT_ACCENCSDFd_WRB (d=0-3)	CSRM DFLASH (EEPROM) memory Dd, write access enable register B	$0F64_H$+d*20_H	U	CSDF, 32	Application Reset	945
DMU_PROT_ACCENCSDFd_RDA (d=0-3)	CSRM DFLASH (EEPROM) memory Dd, read access enable register A	$0F68_H$+d*20_H	U	CSDF, 32	Application Reset	946
DMU_PROT_ACCENCSDFd_RDB (d=0-3)	CSRM DFLASH (EEPROM) memory Dd, read access enable register B	$0F6C_H$+d*20_H	U	CSDF, 32	Application Reset	946
DMU_PROT_ACCENCSDFd_VM (d=0-3)	CSRM DFLASH (EEPROM) memory Dd, VM access enable register	$0F70_H$+d*20_H	U	CSDF, 32	Application Reset	947
DMU_PROT_ACCENCSDFd_PRS (d=0-3)	CSRM DFLASH (EEPROM) memory Dd, PRS access enable register	$0F74_H$+d*20_H	U	CSDF, 32	Application Reset	947
DMU_PROT_ACCENCSDFd_RGNLA (d=0-3)	CSRM DFLASH (EEPROM) memory Dd, region lower address register	$0F78_H$+d*20_H	U	CSDF, 32	Application Reset	948
DMU_PROT_ACCENCSDFd_RGNUA (d=0-3)	CSRM DFLASH (EEPROM) memory Dd, region upper address register	$0F7C_H$+d*20_H	U	CSDF, 32	Application Reset	948

(table continues...)

Table 556 (continued) Registers overview - DMU (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
DMU_PROT_ACCENCSDFUCB_WRA	CSRM UCB memory, write access enable register A	$0FE0_H$	U	CSDF, 32	Application Reset	949
DMU_PROT_ACCENCSDFUCB_WRB	CSRM UCB memory, write access enable register B	$0FE4_H$	U	CSDF, 32	Application Reset	949
DMU_PROT_ACCENCSDFUCB_RDA	CSRM UCB memory, read access enable register A	$0FE8_H$	U	CSDF, 32	Application Reset	950
DMU_PROT_ACCENCSDFUCB_RDB	CSRM UCB memory, read access enable register B	$0FEC_H$	U	CSDF, 32	Application Reset	950
DMU_PROT_ACCENCSDFUCB_VM	CSRM UCB memory, VM access enable register	$0FF0_H$	U	CSDF, 32	Application Reset	951
DMU_PROT_ACCENCSDFUCB_PRS	CSRM UCB memory, PRS access enable register	$0FF4_H$	U	CSDF, 32	Application Reset	952
DMU_PROT_APUBYPASS	APU bypass	1000_H	U	32	952	952

5.8.4.4 Device specific registers

There are no device specific register changes.

5.8.4.5 Register address space - FSI

Table 557 Registers address space - FSI

Key: The module name in brackets () indicates a memory section name. Without brackets the reference is to a functional block.

Module	Base address	End address	Note
(FSI_CSRM)	$F8020000_H$	$F8027FFF_H$	FSI SRAM
FSI_CSRM	$F8028000_H$	$F80280FF_H$	FSI SFRs
(FSI_HOST)	$F8000000_H$	$F8007FFF_H$	FSI SRAM
FSI_HOST	$F8008000_H$	$F80080FF_H$	FSI SFRs

5.8.4.6 Register overview - access mode glossary

Table 558 Register overview - access mode glossary

Keyword	Description
M	Indicates a module specific access condition. Refer to the register description for details of the specific access condition.

5.8.4.7 Register overview - FSI_CSRM (ascending offset address)

Table 559 Register overview - FSI_CSRM (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
FSI_CSRM_COM M0	Communication register 0	0000_H	M	M	System Reset	954
FSI_CSRM_COM M1	Communication register 1	0001_H	M	M	System Reset	954
FSI_CSRM_COM M2	Communication register 2	0002_H	M	M	System Reset	955

5.8.4.8 Register overview - FSI_HOST (ascending offset address)

Table 560 Register overview - FSI_HOST (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
FSI_HOST_COM M0	Communication register 0	0000_H	M	M	System Reset	954
FSI_HOST_COM M1	Communication register 1	0001_H	M	M	System Reset	954
FSI_HOST_COM M2	Communication register 2	0002_H	M	M	System Reset	955

5.8.4.9 Device specific registers

There are no device specific register changes.

5.8.4.10 Register address space - PFRWB

Table 561 Registers address space - PFRWB

Module	Base address	End address	Note
PFRWB0A	$F8410000_H$	$F8417FFF_H$	
PFRWB0B	$F8418000_H$	$F841FFFF_H$	
PFRWB1A	$F8450000_H$	$F8457FFF_H$	
PFRWB1B	$F8458000_H$	$F845FFFF_H$	
PFRWB2A	$F8490000_H$	$F8497FFF_H$	
PFRWB2B	$F8498000_H$	$F849FFFF_H$	
PFRWB3A	$F84D0000_H$	$F84D7FFF_H$	
PFRWB3B	$F84D8000_H$	$F84DFFFF_H$	
PFRWB4A	$F8510000_H$	$F8517FFF_H$	
PFRWB4B	$F8518000_H$	$F851FFFF_H$	
PFRWB5A	$F8550000_H$	$F8557FFF_H$	
PFRWB5B	$F8558000_H$	$F855FFFF_H$	
PFRWBCS	$F8590000_H$	$F8597FFF_H$	

5.8.4.11 Register overview - access mode glossary

Table 562 Register overview - access mode glossary

Keyword	Description
M	Indicates a module specific access condition. Refer to the register description for details of the specific access condition.
32	Access only when using 32-bit width.
BE	Always returns a Bus Error.

5.8.4.12 Register overview - PFRWB0A (ascending offset address)

Table 563 Register overview - PFRWB0A (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PFRWB0A_UR_FLASHCON2	PFRWB FLASH configuration register 2	000000_H	M, 32	M, 32	Application Reset	956
PFRWB0A_UR_FLASHCON3	PFRWB FLASH configuration register 3	000004_H	M, 32	M, 32	Application Reset	958
PFRWB0A_UR_ECCR	ECC read register	002000_H	M, 32	BE	System Reset	960

(table continues...)

Table 563 (continued) Register overview - PFRWB0A (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PFRWB0A_UR_ECCS	ECC status register	002004_H	M, 32	BE	System Reset	960
PFRWB0A_UR_SBABx (x=0-16)	Single bit error address buffer x	002080_H +x*4	M, 32	BE	System Reset	963
PFRWB0A_UR_DBABx (x=0-1)	Double bit error address buffer x	002100_H +x*4	M, 32	BE	System Reset	964
PFRWB0A_UR_UBABx (x=0)	Uncorrectable error address buffer x	002180_H	M, 32	BE	System Reset	964

5.8.4.13 Register overview - PFRWB1A (ascending offset address)

Table 564 Register overview - PFRWB1A (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PFRWB1A_UR_FLASHCON2	PFRWB FLASH configuration register 2	000000_H	M, 32	M, 32	Application Reset	956
PFRWB1A_UR_FLASHCON3	PFRWB FLASH configuration register 3	000004_H	M, 32	M, 32	Application Reset	958
PFRWB1A_UR_ECCR	ECC read register	002000_H	M, 32	BE	System Reset	960
PFRWB1A_UR_ECCS	ECC status register	002004_H	M, 32	BE	System Reset	960
PFRWB1A_UR_SBABx (x=0-16)	Single bit error address buffer x	002080_H +x*4	M, 32	BE	System Reset	963
PFRWB1A_UR_DBABx (x=0-1)	Double bit error address buffer x	002100_H +x*4	M, 32	BE	System Reset	964
PFRWB1A_UR_UBABx (x=0)	Uncorrectable error address buffer x	002180_H	M, 32	BE	System Reset	964

5.8.4.14 Register overview - PFRWB2A (ascending offset address)

Table 565 Register overview - PFRWB2A (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PFRWB2A_UR_FLASHCON2	PFRWB FLASH configuration register 2	000000_H	M, 32	M, 32	Application Reset	956

(table continues...)

Table 565 (continued) Register overview - PFRWB2A (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PFRWB2A_UR_FLASHCON3	PFRWB FLASH configuration register 3	000004_H	M, 32	M, 32	Application Reset	958
PFRWB2A_UR_ECCR	ECC read register	002000_H	M, 32	BE	System Reset	960
PFRWB2A_UR_ECCS	ECC status register	002004_H	M, 32	BE	System Reset	960
PFRWB2A_UR_SBABx (x=0-16)	Single bit error address buffer x	002080_H +x*4	M, 32	BE	System Reset	963
PFRWB2A_UR_DBABx (x=0-1)	Double bit error address buffer x	002100_H +x*4	M, 32	BE	System Reset	964
PFRWB2A_UR_UBABx (x=0)	Uncorrectable error address buffer x	002180_H	M, 32	BE	System Reset	964

5.8.4.15 Register overview - PFRWB3A (ascending offset address)

Table 566 Register overview - PFRWB3A (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PFRWB3A_UR_FLASHCON2	PFRWB FLASH configuration register 2	000000_H	M, 32	M, 32	Application Reset	956
PFRWB3A_UR_FLASHCON3	PFRWB FLASH configuration register 3	000004_H	M, 32	M, 32	Application Reset	958
PFRWB3A_UR_ECCR	ECC read register	002000_H	M, 32	BE	System Reset	960
PFRWB3A_UR_ECCS	ECC status register	002004_H	M, 32	BE	System Reset	960
PFRWB3A_UR_SBABx (x=0-16)	Single bit error address buffer x	002080_H +x*4	M, 32	BE	System Reset	963
PFRWB3A_UR_DBABx (x=0-1)	Double bit error address buffer x	002100_H +x*4	M, 32	BE	System Reset	964
PFRWB3A_UR_UBABx (x=0)	Uncorrectable error address buffer x	002180_H	M, 32	BE	System Reset	964

5.8.4.16 Register overview - PFRWB4A (ascending offset address)

Table 567 Register overview - PFRWB4A (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PFRWB4A_UR_FLASHCON2	PFRWB FLASH configuration register 2	000000_H	M, 32	M, 32	Application Reset	956
PFRWB4A_UR_FLASHCON3	PFRWB FLASH configuration register 3	000004_H	M, 32	M, 32	Application Reset	958
PFRWB4A_UR_ECCR	ECC read register	002000_H	M, 32	BE	System Reset	960
PFRWB4A_UR_ECCS	ECC status register	002004_H	M, 32	BE	System Reset	960
PFRWB4A_UR_SBABx (x=0-16)	Single bit error address buffer x	002080_H +x*4	M, 32	BE	System Reset	963
PFRWB4A_UR_DBABx (x=0-1)	Double bit error address buffer x	002100_H +x*4	M, 32	BE	System Reset	964
PFRWB4A_UR_UBABx (x=0)	Uncorrectable error address buffer x	002180_H	M, 32	BE	System Reset	964

5.8.4.17 Register overview - PFRWB5A (ascending offset address)

Table 568 Register overview - PFRWB5A (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PFRWB5A_UR_FLASHCON2	PFRWB FLASH configuration register 2	000000_H	M, 32	M, 32	Application Reset	956
PFRWB5A_UR_FLASHCON3	PFRWB FLASH configuration register 3	000004_H	M, 32	M, 32	Application Reset	958
PFRWB5A_UR_ECCR	ECC read register	002000_H	M, 32	BE	System Reset	960
PFRWB5A_UR_ECCS	ECC status register	002004_H	M, 32	BE	System Reset	960
PFRWB5A_UR_SBABx (x=0-16)	Single bit error address buffer x	002080_H +x*4	M, 32	BE	System Reset	963
PFRWB5A_UR_DBABx (x=0-1)	Double bit error address buffer x	002100_H +x*4	M, 32	BE	System Reset	964
PFRWB5A_UR_UBABx (x=0)	Uncorrectable error address buffer x	002180_H	M, 32	BE	System Reset	964

5.8.4.18 Register overview - PFRWB0B (ascending offset address)

Table 569 Register overview - PFRWB0B (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PFRWB0B_UR_FLASHCON2	PFRWB FLASH configuration register 2	000000_H	M, 32	M, 32	Application Reset	956
PFRWB0B_UR_FLASHCON3	PFRWB FLASH configuration register 3	000004_H	M, 32	M, 32	Application Reset	958
PFRWB0B_UR_ECCR	ECC read register	002000_H	M, 32	BE	System Reset	960
PFRWB0B_UR_ECCS	ECC status register	002004_H	M, 32	BE	System Reset	960
PFRWB0B_UR_SBABx (x=0-16)	Single bit error address buffer x	002080_H +x*4	M, 32	BE	System Reset	963
PFRWB0B_UR_DBABx (x=0-1)	Double bit error address buffer x	002100_H +x*4	M, 32	BE	System Reset	964
PFRWB0B_UR_UBABx (x=0)	Uncorrectable error address buffer x	002180_H	M, 32	BE	System Reset	964

5.8.4.19 Register overview - PFRWB1B (ascending offset address)

Table 570 Register overview - PFRWB1B (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PFRWB1B_UR_FLASHCON2	PFRWB FLASH configuration register 2	000000_H	M, 32	M, 32	Application Reset	956
PFRWB1B_UR_FLASHCON3	PFRWB FLASH configuration register 3	000004_H	M, 32	M, 32	Application Reset	958
PFRWB1B_UR_ECCR	ECC read register	002000_H	M, 32	BE	System Reset	960
PFRWB1B_UR_ECCS	ECC status register	002004_H	M, 32	BE	System Reset	960
PFRWB1B_UR_SBABx (x=0-16)	Single bit error address buffer x	002080_H +x*4	M, 32	BE	System Reset	963
PFRWB1B_UR_DBABx (x=0-1)	Double bit error address buffer x	002100_H +x*4	M, 32	BE	System Reset	964
PFRWB1B_UR_UBABx (x=0)	Uncorrectable error address buffer x	002180_H	M, 32	BE	System Reset	964

5.8.4.20 Register overview - PFRWB2B (ascending offset address)

Table 571 Register overview - PFRWB2B (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PFRWB2B_UR_FLASHCON2	PFRWB FLASH configuration register 2	000000_H	M, 32	M, 32	Application Reset	956
PFRWB2B_UR_FLASHCON3	PFRWB FLASH configuration register 3	000004_H	M, 32	M, 32	Application Reset	958
PFRWB2B_UR_ECCR	ECC read register	002000_H	M, 32	BE	System Reset	960
PFRWB2B_UR_ECCS	ECC status register	002004_H	M, 32	BE	System Reset	960
PFRWB2B_UR_SBABx (x=0-16)	Single bit error address buffer x	002080_H +x*4	M, 32	BE	System Reset	963
PFRWB2B_UR_DBABx (x=0-1)	Double bit error address buffer x	002100_H +x*4	M, 32	BE	System Reset	964
PFRWB2B_UR_UBABx (x=0)	Uncorrectable error address buffer x	002180_H	M, 32	BE	System Reset	964

5.8.4.21 Register overview - PFRWB3B (ascending offset address)

Table 572 Register overview - PFRWB3B (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PFRWB3B_UR_FLASHCON2	PFRWB FLASH configuration register 2	000000_H	M, 32	M, 32	Application Reset	956
PFRWB3B_UR_FLASHCON3	PFRWB FLASH configuration register 3	000004_H	M, 32	M, 32	Application Reset	958
PFRWB3B_UR_ECCR	ECC read register	002000_H	M, 32	BE	System Reset	960
PFRWB3B_UR_ECCS	ECC status register	002004_H	M, 32	BE	System Reset	960
PFRWB3B_UR_SBABx (x=0-16)	Single bit error address buffer x	002080_H +x*4	M, 32	BE	System Reset	963
PFRWB3B_UR_DBABx (x=0-1)	Double bit error address buffer x	002100_H +x*4	M, 32	BE	System Reset	964
PFRWB3B_UR_UBABx (x=0)	Uncorrectable error address buffer x	002180_H	M, 32	BE	System Reset	964

5.8.4.22 Register overview - PFRWB4B (ascending offset address)

Table 573 Register overview - PFRWB4B (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PFRWB4B_UR_FLASHCON2	PFRWB FLASH configuration register 2	000000_H	M, 32	M, 32	Application Reset	956
PFRWB4B_UR_FLASHCON3	PFRWB FLASH configuration register 3	000004_H	M, 32	M, 32	Application Reset	958
PFRWB4B_UR_ECCR	ECC read register	002000_H	M, 32	BE	System Reset	960
PFRWB4B_UR_ECCS	ECC status register	002004_H	M, 32	BE	System Reset	960
PFRWB4B_UR_SBABx (x=0-16)	Single bit error address buffer x	002080_H +x*4	M, 32	BE	System Reset	963
PFRWB4B_UR_DBABx (x=0-1)	Double bit error address buffer x	002100_H +x*4	M, 32	BE	System Reset	964
PFRWB4B_UR_UBABx (x=0)	Uncorrectable error address buffer x	002180_H	M, 32	BE	System Reset	964

5.8.4.23 Register overview - PFRWB5B (ascending offset address)

Table 574 Register overview - PFRWB5B (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PFRWB5B_UR_FLASHCON2	PFRWB FLASH configuration register 2	000000_H	M, 32	M, 32	Application Reset	956
PFRWB5B_UR_FLASHCON3	PFRWB FLASH configuration register 3	000004_H	M, 32	M, 32	Application Reset	958
PFRWB5B_UR_ECCR	ECC read register	002000_H	M, 32	BE	System Reset	960
PFRWB5B_UR_ECCS	ECC status register	002004_H	M, 32	BE	System Reset	960
PFRWB5B_UR_SBABx (x=0-16)	Single bit error address buffer x	002080_H +x*4	M, 32	BE	System Reset	963
PFRWB5B_UR_DBABx (x=0-1)	Double bit error address buffer x	002100_H +x*4	M, 32	BE	System Reset	964
PFRWB5B_UR_UBABx (x=0)	Uncorrectable error address buffer x	002180_H	M, 32	BE	System Reset	964

5.8.4.24 Register overview - PFRWBCS (ascending offset address)

Table 575 Register overview - PFRWBCS (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PFRWBCS_UR_FLASHCON2	PFRWB FLASH configuration register 2	000000_H	M, 32	M, 32	Application Reset	956
PFRWBCS_UR_FLASHCON3	PFRWB FLASH configuration register 3	000004_H	M, 32	M, 32	Application Reset	958
PFRWBCS_UR_ECCR	ECC read register	002000_H	M, 32	BE	System Reset	960
PFRWBCS_UR_ECCS	ECC status register	002004_H	M, 32	BE	System Reset	960
PFRWBCS_UR_SBABx (x=0-16)	Single bit error address buffer x	002080_H +x*4	M, 32	BE	System Reset	963
PFRWBCS_UR_DBABx (x=0-1)	Double bit error address buffer x	002100_H +x*4	M, 32	BE	System Reset	964
PFRWBCS_UR_UBABx (x=0)	Uncorrectable error address buffer x	002180_H	M, 32	BE	System Reset	964

5.8.4.25 Device specific registers

There are no device specific register changes.

5.8.5 TC4Dx NVM connectivity

There are no deviations from the generic specification.

5.8.6 TC4Dx NVM revision history

Initial release of the chapter.

6 Shared Resource Interconnect (SRI)

The Shared Resource Interconnect (SRI) connects modules with native SRI interfaces such as TriCore™ CPUs and high performance modules such as the DMA with memories and peripherals. The SRI crossbar connects all the agents in one SRI domain with a deterministic performance objective. The fabric is extended with bridges, to connect one SRI domain to another as well as to other components and interconnects operating on different protocols.

The SRI crossbar supports concurrent transactions between different SRI master and SRI slave agents. When transacting to the same slave, the master can achieve very high utilization over the link by employing pipelining. In addition, a master can conduct read and write transfers to different slaves in parallel.

6.1 Feature list

- Single (8-bit, 16-bit, 32-bit and 64-bit) and block read and write transactions (blocks up to 8 x 64-bit beats)
- Atomic read write transaction sequence supported
- Pipelined transactions from SRI masters to SRI slaves
  - Improved read write parallelism compared to previous generations
- Arbiter for each SRI slave, with individual configuration
  - Two round-robin groups, high and low priority
  - Control of opportunity for the high priority group
- EDC (Error Detection Code) on all address, data and control information transferred between SRI masters and SRI slaves
  - EDC on all data and control information transferred from an SRI master to an SRI slave for a write
  - EDC on all data and control information transferred from an SRI slave to an SRI master for a read
- Device specific capability to monitor and record bandwidth utilization of resources for Quality of Service (QoS) purposes

6.2 Functional overview

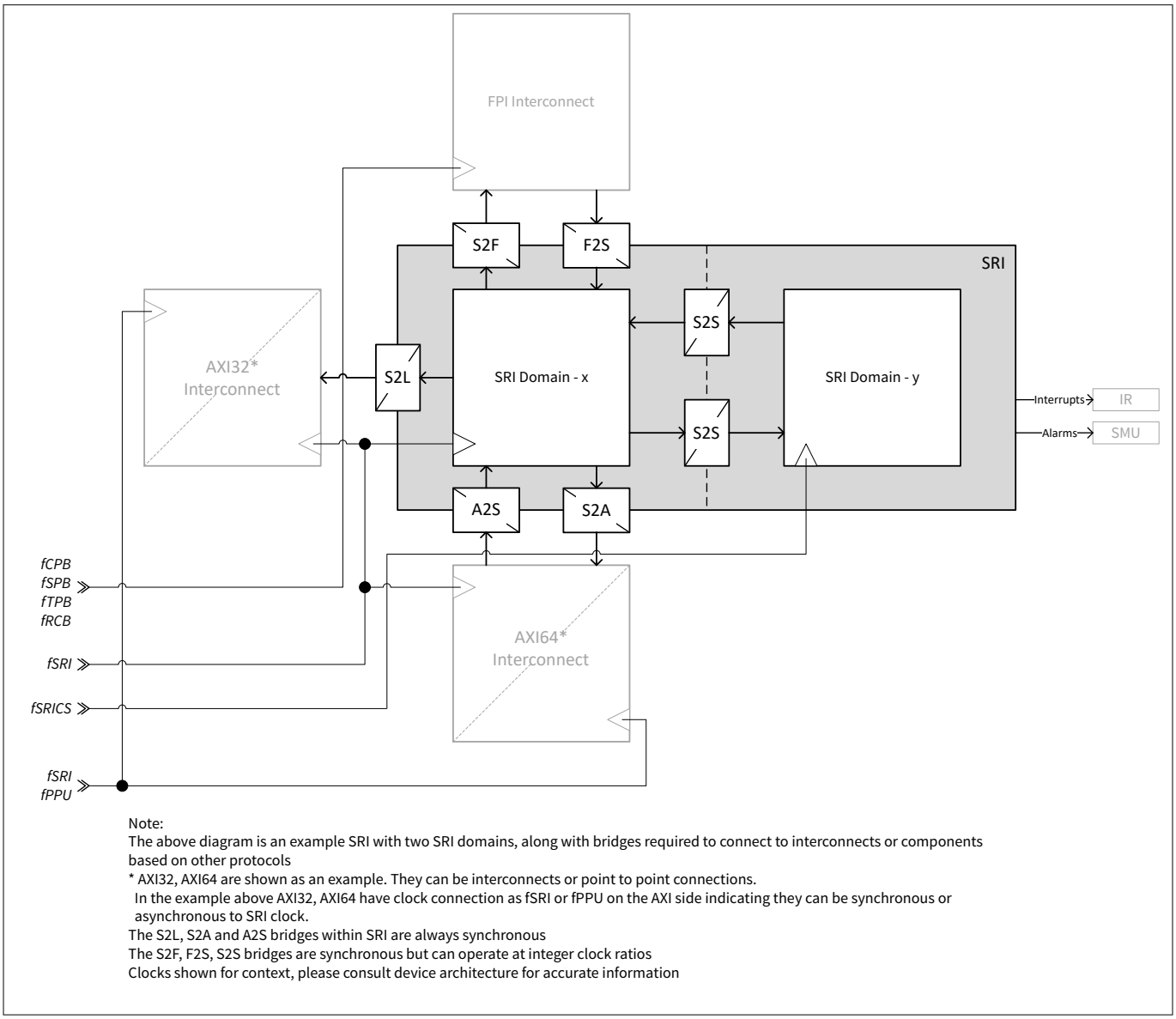


Figure 58 SRI block diagram

For the alarm description please refer to the alarm mapping tables in the SMU functional block and the Safety Manual.

The ERAY consists of one or more crossbars implementing a high bandwidth, deterministic latency, on-chip fabric to interconnect master agents to slave agents through the SRI protocol. On complex systems with numerous masters and slaves, the SRI fabric is subdivided into domains. Each domain implements a crossbar to interconnect a subset of unique masters and slaves. SRI domains connect to other SRI domains, or other interconnects such as FPI and AXI (Advanced extensible interface) through bridges such as:

- S2S - SRI to SRI bridge
- S2A - SRI to AXI64 bridge
- S2L - SRI to AXI32 bridge
- S2F - SRI to FPI bridge
- A2S - AXI64 to SRI bridge
- F2S - FPI to SRI bridge

The bridges permit a master to reach all slaves if desired. Equally, masters may not be connected to all slaves within a single SRI domain if there is no functional need for them to communicate. The interconnected domains

all have clocks derived from the same root, therefore operate either on the same clock or a limited set of integer clock ratios. The bridges implement the clock domain crossing points within the interconnect. The number of SRI domains varies per device.

Related information

TC4Dx SMU alarm mapping tables on page 7233

6.3 Functional description

The SRI fabric includes one or more SRI domains each with an SRI crossbar, along with the bridges necessary to connect between interconnect domains.

6.3.1 Operational overview

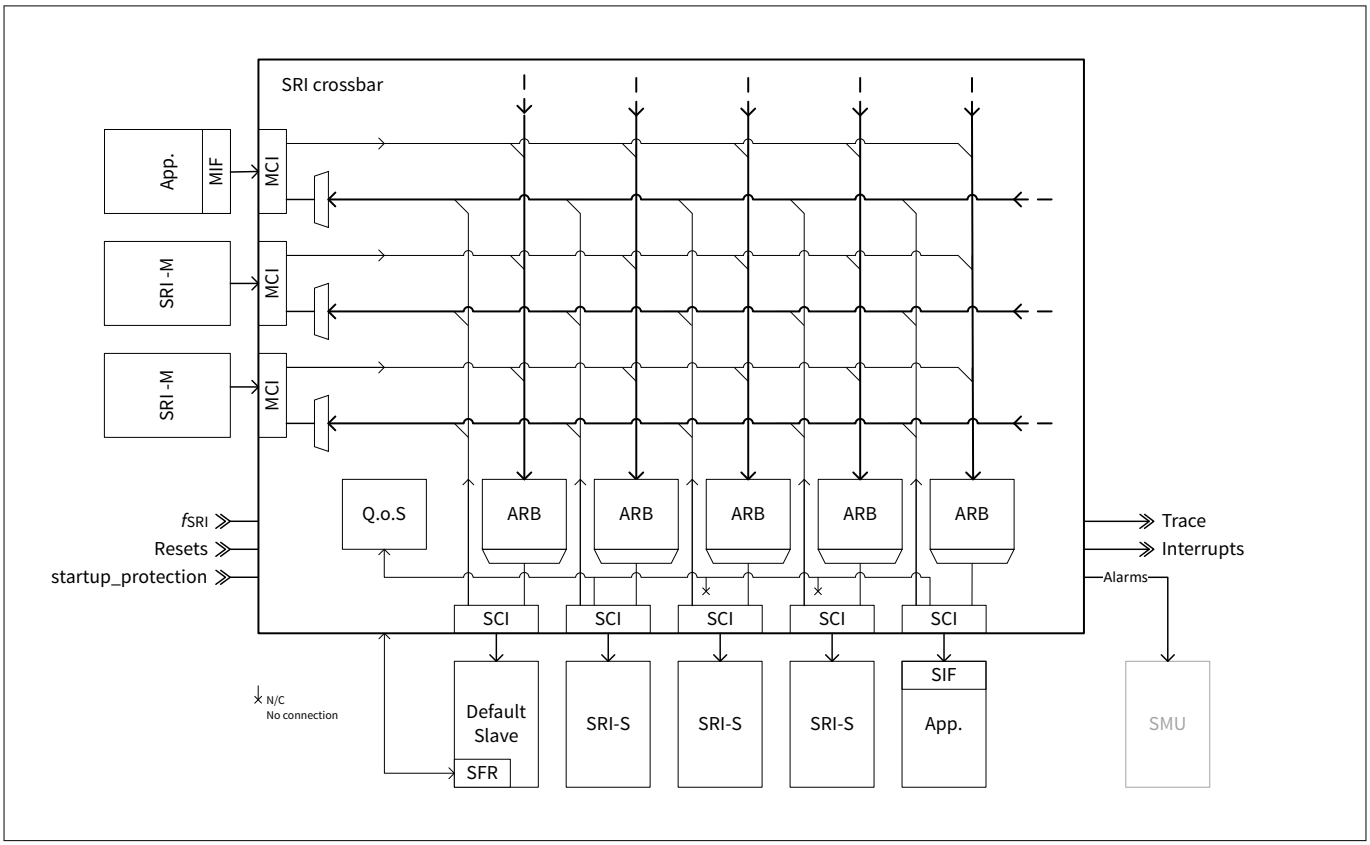


Figure 59 Example of an SRI crossbar

The SRI crossbar provides SRI master connection interfaces (MCIx) to connect SRI master modules and SRI slave connection interfaces (SCIx) to connect SRI slave modules to the SRI crossbar. There is one arbiter per connected SRI slave module and the infrastructure for the enabled read and write data paths.

Each SRI domain will always include at least one slave, called the default slave, to provide housekeeping functions such as:

- domain specific control and status registers
- sink transactions that are to reserved memory regions in the address map

The SRI domain provides arbitration that permits the independent configuration of SRI master priorities for every SRI slave. The SRI includes resources to capture and report various error conditions required for system level diagnostics. Quality of service (QoS) observation capability may also be provided within an SRI domain. The availability of QoS monitoring is device specific and not all connected resources implement QoS monitoring

Table 576 SRI fabric terms

Term	Description
Agent	An SRI agent is any master or slave interface which is connected to the SRI fabric.

(table continues...)

Table 576 (continued) SRI fabric terms

Term	Description
Master agent	An SRI master agent is able to initiate transactions on the SRI through an SRI master interface. More than one master agent function can drive through an SRI master agent. CPU or DMA for example. A master agent is identified as SRI-M in the SRI crossbar diagram.
App.	An application kernel which implements an application interface that connects to a MIF or SIF.
MIF	• Master Interface, abstracts the SRI protocol to an application interface simplifying the integration of master agents • A MIF with an application kernel (App.), also makes a master agent
Slave agent	An SRI slave agent is able to respond to transactions on the SRI received through an SRI slave interface. Memories or register interfaces for example. A slave agent is identified as SRI-S in the SRI crossbar diagram
SIF	• Slave Interface, abstracts the SRI protocol to an application interface simplifying the integration of slave agents • A SIF with an application kernel (App.), also makes a master agent
SRI crossbar	The SRI crossbar provides the connectivity between master and slave agents in the same domain. The SRI crossbar includes arbitration mechanisms and error capture capabilities.
MCI	Each master is connected through one master connection interface.
SCI	Each slave is connected through one slave connection interface. The SRI contains control registers which allow configuration for each SCI of the high or low priority of each MCI for arbitration.
Domain	An SRI domain consists of those agents which are connected to a specific SRI crossbar. There will be at least one master agent and at least one slave agent, and an instance of a crossbar providing full or partial connectivity between all the agents in the domain.
Arbiter	If two or more master agents attempt to access the same slave agent, the arbiter provides the order in which master agents are serviced. The order is determined by the two-level round-robin mechanism implemented by the arbiter and the configuration of master agent priority. The arbiter is identified as ARB in the SRI crossbar diagram.
Default slave	A slave agent which both provides error acknowledgment for unknown addresses and provides access to the configuration and error capture registers of the domain.
QoS	Quality of Service observation point. Provides a method to monitor the interconnect usage on a specific SCI port.

The SRI crossbar can perform multiple transactions from different master agents in parallel if the master agents are accessing different slave agents. A master can also carry out parallel read write transactions to different slaves. When multiple masters compete to access the same resource, an arbiter resolves the ensuing resource contention.

6.3.1.1 Master connection interface (MCI)

Each SRI master in the domain is mapped to master related control and status register bits and bit-fields through its MCI number.

The MCI number is completely distinct from the master tag(s) used by the SRI master connected to that MCI. However, the set of master tags presented by a specific MCI is fixed for a given device.

6.3.1.2 Slave connection interface (SCI)

Each SRI Slave in the system is mapped to slave related control and status register bits and bit-fields through its SCI number.

6.3.1.3 Slave arbiter

Each SCI has an associated arbiter which includes all the functionality for the following tasks:

- Address decoding
- Arbitration to permit a master to access a shared resource in a mult-master environment
- Error capture
  - Errors detected by a slave (some errors are detected by masters) will have transaction information captured where possible and the condition, if enabled, signaled to the system through the Interrupt Router (IR)

6.3.1.4 Default slave

A default slave is an SRI slave which implements necessary housekeeping functions. As an SRI slave, it has its own SCI and arbiter.

A default slave services a single domain. For an SRI fabric with multiple domains there will be one default slave per domain.

A default slave provides the following functions:

- Access to domain specific control and status registers
- Any transactions that fall into a reserved region as determined by the local decoders are directed to the default slave, which can record and complete the transaction
  - recording and reporting of such transactions is further controlled by the cyber security activity visibility (CSAV) attributes
- One default slave supports the online data acquisition (OLDA) feature for the complete interconnect

Each default slave is allocated its unique address space, which is used for accesses to its allocated fabric control and status registers. For accesses to the control and status registers, only single data transactions of word size (32-bit) are permitted. Unsupported transactions are acknowledged with an error.

A default slave completes SRI transactions to non-existent addresses (both reads and write) with an SRI error, which will activate the error capturing mechanism of the default slave's arbiter.

***Note**: A read by an SRI master to a non-existent address in another domain or interconnect (for example, the SPB) may be reported both by the other interconnect, and also reported in the local SRI domain by the bridge to the other interconnect. In the ERAY, this transaction information will be captured by the bridge's arbiter, rather than the default slave for the domain. A write by an SRI master to a non-existent address which is located in another domain or interconnect is posted. A posted write's error will only be reported directly or indirectly in the interconnect segment that generated the error.*

Related information

System address map on page 14
Online data acquisition (OLDA) on page 999

6.3.1.5 Online data acquisition (OLDA)

OLDA is a defined region in the address map, which can be used for data logging during debug and development. Additionally, a physical memory region can be mapped to the OLDA using the Data access

overlay feature of the TriCore™ CPUs. If the overlay feature is enabled, any system memory can be used as overlay memory. OLDA region is allocated to the default slave in domain 0.

When OLDA is enabled, while executing code without a physical memory mapped to the OLDA region, the designated default slave completes all write transfers to the OLDA region by silently dropping them. Therefore OLDA permits production code to incorporate logging and debug, enabling seamless transition between diagnostic and mission mode profiles with a common code base.

The OLDA region has the following characteristics:

- The base address of the OLDA memory range is defined in the system address map
- OLDA support is enabled by setting bit OLDAEN to 1 in the BRCON register
  - If OLDA support is enabled, write accesses do not generate an SRI error
  - If OLDA support is disabled, write accesses will generate an SRI error
  - Read access to an OLDA region will result in bus error when the overlay does not remap it to a physical memory
- OLDA address range is visible from both cached and un-cached segments
  - Reads due to write allocation on the cache, will generate bus errors when data overlay is not enabled

Switching on or off the OLDA function takes a number of system clock cycles after the BRCON register write completes. It is therefore possible for an immediately subsequent access to be pipelined to the OLDA address space, and be acknowledged with either the “on” or “off” behavior. A method of synchronization such as read back of BRCON or another enhanced method should be considered to ensure that the mode change has made effect. However in a multi master environment, more advanced synchronization methods may be required.

***Note**: If enabled in the device, any target address can be selected for redirection in the OTARx registers, including addresses in the OLDA range. However, the handling of direct accesses to the OLDA range is completely controlled by the accessed default slave.*

6.3.1.6 Control and status register access protection

The ERAY control and status registers are protected by a master agent function ID which consists of a unique 6-bit TAG-ID, the optional 3-bit VM information, and the optional 3-bit PRS information.

Each on-chip resource with direct or indirect bus master capability has a unique master TAG ID that can be used to identify the master of an on-chip bus transaction. The protection mechanism and requirements are described in the Access Protection chapter.

After application reset, the access protection mechanism must be configured to put the system into the state that is required by the application.

6.3.2 Arbitration details

An SRI crossbar provides a dedicated SRI arbiter for each connected slave agent. On the crossbar, multiple parallel requests for the same slave agent are arbitrated by the respective arbiter and executed in a sequence. The arbiters consider a request as a unit item, therefore ignore transaction sizes and disregard the actual claimed bandwidth from arbitration decisions.

6.3.2.1 Transaction request arbitration

The arbiter used in the SRI work on a work-conserving round robin (RR) scheme. Round robin provides a predictable starvation free mechanism wherein each request is chosen in a cyclical order. The work conserving scheme ensures that no arbitration slots are wasted when the nominal winner is inactive. Inactive masters are skipped in the current round and the next higher priority master is selected.

With the basic scheme as outlined above, arbitration is further broken down into two priority groups; high priority round robin group (HPRR) and low priority round robin group (LPRR). An arbitration cycle executes both rounds in parallel. The specific allocation of a master to a round robin group is configured by software in PRIORITYx.

A further refinement to the algorithm considers the outcome of both rounds. The high priority round share parameter (PRIORITYx.HPRS) monitors the number of successive HPRR winning slots, and inserts an LPRR slot if it has an active request. If there are no HPRR winners in a specific slot, the LPRR is the default winner. PRIORITYx.HPRS permits an asymmetric bandwidth distribution among the masters in a domain.

The arbitration scheme operates at the SRI request granularity. It does not take any other attribute of that request into consideration. Therefore any observation of bandwidth share between the masters may indicate a non-uniform bandwidth allocation.

Related information

Arbitration configuration on page 1001
Request latency on page 1001

6.3.2.2 Arbitration configuration

The arbitration configuration provides software with the ability to map an MCI (and therefore the master agents connected to it) to the HPRR or LPRR arbitration group. The configuration is domain and SCI specific. On a given domain, a certain master agent can be mapped to different levels in different SCI arbiters. PRIORITYx.MCIn_P configuration directs a request from an MCI with the corresponding index to HPRR or LPRR. The PRIORITYx.HPRS determines the high priority round share for the domain.

After reset most present MCIs are set to the low priority level, although bridges and DMAs are typically set to high priority.

For error diagnosis purposes, when permitted by CSAV configuration, the arbiter samples all the necessary transaction information and provides this in the ERRADDRx and ERRx registers if an SRI protocol error occurs.

Related information

SRI protocol errors on page 1003
Error tracking capability on page 1005

6.3.2.3 Request latency

If another transaction is not pending at the SCI, a transaction from an MCI will be granted in the next clock cycle after that master's request. If there are multiple requests pending, or arrive at the same time at the same SCI, then the latency depends on whether the MCI is in the HPRR or LPRR group at the slave agent, and how many other requests are in each round.

The following table shows an example of the worst case number of transactions that a master might have to wait before its transaction can commence in an example of a domain with 8 masters.

Table 577 Master request transactions

Number of high / low priority master agents	High priority round share(HPRS) for parallel requests to the same slave agent (FB or bridge)	Worst case delay in arbitration for a high priority master[1)]	Worst case delay in arbitration for a low priority master
0 / 8	(0)	N.A.	7
1 / 7	1	1	13

(table continues...)

Table 577 (continued) Master request transactions

Number of high / low priority master agents	High priority round share(HPRS) for parallel requests to the same slave agent (FB or bridge)	Worst case delay in arbitration for a high priority master[1)]	Worst case delay in arbitration for a low priority master
	2	1	20
	3	1	27
	4	1	34
	5	1	41
	6	1	48
	7	1	55
2 / 6	2	2	17
	3	2	23
	4	2	29
	5	2	35
	6	2	41
	7	2	47
3 / 5	3	3	19
	4	3	24
	5	3	29
	6	3	34
	7	3	39
4 / 4	4	4	19
	5	4	23
	6	4	27
	7	4	31
5 / 3	5	5	17
	6	5	20
	7	5	23
6 / 2	6	6	13
	7	6	15
7 / 1	7	7	7
8 / 0	(0)	7	N.A.

1) Refers to winning arbitration slots, and not directly clocks

The worst case number of transactions to wait for access to a slave with fewer masters can be computed as follows:

If there is only a single round of arbitration (high or low), then;

```
max_delay    = number_of_masters-1
```

If there are two rounds (high and low) of arbitration, then for a high priority master;

```
HP_max_delay    = number_of_high_priority_masters
```

and for a low priority master;

```
LP_max_delay    = ((HPRS+1) x number_of_low_priority_masters)-1
```

The worst case number of clock cycles a master waits will depend on the response time of the respective slave. So if for example the SRI Slave has a response time of 10 clocks, the worst case wait in clock cycles for a low priority master in a 2 high, 6 low configuration with a high priority share set to 7, is 470 clocks.

Usually each SRI slave will have a deterministic response time to transactions, although it might vary between read and write transactions, and the data width to be transferred. It is typically only in the case of off-chip accesses, such as the xSPI, where there might be high variation in transaction times.

6.3.3 SRI errors

SRI error conditions can be detected by both masters and slaves depending on the condition. Errors are always reported through alarms to the SMU and optionally interrupts, or signaled directly from a slave to the master within a domain. Some masters, such as the CPUs, can directly respond by taking a trap.

For some errors, a slave can signal to the corresponding master that an error has occurred, which results in an immediate termination of the current transaction (and the status is also captured in the interconnect). For other errors during a transaction, the transaction continues, but the error is captured by the SRI and signaled through an alarm and optionally an interrupt if enabled in both the SRI and interrupt router (IR). The following sections describe the behavior of each error type in detail.

There are three error types:

- SRI Protocol Errors
- SRI Transaction ID Errors
- SRI EDC Errors

SRI interrupts

A crossbar has two interrupt trigger outputs for errors, an SRI_IR_SRC_SRIx and SRI_IR_SRC_SRIxSM. These are only generated if enabled.

6.3.3.1 SRI protocol errors

The following error conditions are detected by SRI slaves and result in error termination of the transaction (SRI protocol error - signaled as an SRI error) to the SRI master and an alarm:

- Non-existent address region accessed by an SRI master
  - The accesses to non-existent address regions are checked by a default slave
- Reserved address within an existing memory region
- Unsupported or reserved op-codes for the addressed slave
- On detection of an access protection violation, such as a supervisor register accessed in user mode

- Access protection is configured to not allow the current SRI Master read access
- Access protection is configured to not allow the current SRI Master write access
  - Some slaves do not issue a bus error. In that case the write is prevented within the slave and an alarm is issued
- Special master indication is incorrect. On SRI, special master(SM) is the cyber security master(CS). For example a special master region accessed by a non-special master such as CPU0 accessing CSRM PFlash
- Unsupported operation to a specific register, such as a write to a read-only register for example
- EDC error detected for the address phase

For testing purposes, an SRI protocol error can be generated by the application software with an access to a reserved address (refer to the Memory Map chapter).

Those SRI Slaves which have detected a protocol error can be determined by inspection of the domain specific PESTAT register.

6.3.3.2 SRI transaction ID errors

A transaction ID is an identifier connected to all phases of a transaction, in order to make the transaction unique in the SRI during the transaction's lifetime.

The transaction ID is used by SRI Masters and SRI Slaves to identify errors in the SRI that might result in data packets received by a master or slave that do not correspond to the relevant transaction. If the read or write transaction identifier does not match the transaction identifier sent during the address phase, this is a transaction ID error and reported by the SRI crossbar.

In addition to SRI Fabric errors, forcing a transaction ID error is used in situations where at the data source side (the SRI master for write transactions and the SRI slave for read transactions), a data phase has to be invalidated (detection of an uncorrectable error from an accessed memory for example). In these situations, an invalid transaction ID is sent in order to invalidate the data phase.

For testing purposes an SRI transaction ID error condition can be generated by injecting a non-correctable error into one of the SRAMs (CPU scratch pad SRAM for example), and then reading the corrupted data by an SRI master.

Those SRI masters and SRI slaves which have detected a transaction ID error can be determined by inspection of the TIDSTAT register.

6.3.3.3 SRI EDC errors

The SRI provides EDC protection for the:

- Address and control signals of an SRI transaction
- Transmitted read and write data

SRI address phase EDC errors

- If an SRI Slave detects an SRI address EDC error, it finishes the transaction with SRI error (a protocol error) and does not process it further
- In addition to the SRI protocol error termination, each SRI Slave signals an address EDC error to the SMU as an alarm (Address Phase EDC Error Alarm)

SRI write data EDC errors

An SRI Slave will check each data beat of a write transaction for EDC validity.

If an SRI Slave detects an SRI write data EDC error, it is only signaled to the SMU as an alarm.

Whether an SRI Slave only monitors the EDC error condition or attempts to prevent the state update, is not defined by the SRI infrastructure, but is described in each SRI functional block or sub-block chapter.

SRI read data EDC errors

An SRI Master will check each received data beat of a read transaction for EDC validity.

If an SRI Master detects an SRI read data EDC error, it is only signaled to the SMU as an alarm.

Whether an SRI Master only monitors the EDC error condition or attempts to prevent the use of the returned data is not defined by the SRI infrastructure, but is described in each SRI functional block or sub-block chapter.

Because of their key role, the behavior of CPUs is reiterated here. A trap will be taken by the CPU if it receives a data item containing an EDC error and the data would be immediately used by the CPU. The data might not be used and no trap taken if a cache line is requested and the beat containing the EDC error is not required for the instruction stream or the data load instruction. If any part of a received memory line contains an EDC error, a possible cache update is prevented and the received data is discarded.

6.3.3.4 Error tracking capability

The SRI crossbar tracks all SRI transactions for SRI protocol errors, and it tracks information about SRI transaction ID errors, but not EDC errors. This is done by all arbiters in parallel, since the SRI crossbar supports the processing of multiple transactions from masters and slaves in parallel, which can result in concurrent events at different SCIs.

When an arbiter recognizes that its associated SRI Slave has signaled an SRI protocol error for a transaction, the arbiter can sample all relevant information of the transaction in the diagnostic registers (ERRx and ERRADDRx). An indication of this error can be provided to the interrupt router (IR), and this indication can be controlled, through the PECONx.PEEN control bit. Only when this bit is set is the error information captured and the trigger condition signaled to the interrupt router.

***Notes**:*

1. *The two error registers ERRx and ERRADDRx in each arbiter are updated with the content of the currently processed transaction. The registers are only updated if they are not locked due to an earlier protocol error*
2. *No master with SM asserted/set will have information captured unless so enabled by CSAV. Transaction ID errors are exempt from CSAV filtering*
3. *SM accesses will be signaled through a separate trigger*

The error capture register field OPC contains information about the type of transaction for which information has been captured. For single data transactions, this field indicates the number of bytes. For block transactions, this field indicates the number of expected data beats. The encoding is shown in the following table.

Table 578 SRI operation code (opcode) encoding

OPC[3:0]	Identifier	Description
0000	SDTB	Single Data Transfer Byte (1 Byte)
0001	SDTH	Single Data Transfer Half-Word (2 Bytes)
0010	SDTW	Single Data Transfer Word (4 Bytes)
0011	SDTD	Single Data Transfer Double Word (8 Bytes)
0100-0111		Reserved
1000	BTR2	Block Transfer Request (2 Beats)
1001	BTR4	Block Transfer Request (4 Beats)
1010	BTR8	Block Transfer Request (8 Beats)
1011-1111		Reserved

The error capture registers will not be updated again until the indication is acknowledged to the slave arbiter module by software (set the respective PECONx.PEACK).

The error capture registers will be locked after the first error. This results in the non-capture of subsequent error information for this slave agent until the write releasing the lock is processed.

Further protocol errors will be ignored by an arbiter that has detected a protocol error until the capture mechanism is re-activated through the arbiter internal control register. A detected protocol error is signaled by the arbiter to the per-domain error status register PESTAT. Each error signal can be masked individually by the PECONx control registers in the arbiter modules for SCIx. In addition the ERRx and ERRADDRx registers are accessible in each SCIx.

A master or a slave can detect a transaction ID error. An indication of this error can be provided to the interrupt router (IR), and this indication can be controlled through the TIDEN.ENMCI and TIDEN.ENSCI control bits. Only if the relevant enable bit is set is the error condition captured in TIDSTAT, and the trigger condition signaled to the interrupt router.

For transaction ID errors of write transactions, the SCIs propagates errors to the associated bit in TIDSTAT.TIDSCIx. For transaction ID errors of read transactions, the MCIs propagate errors to the assigned bit inTIDSTAT.TIDMCIx.

Once an error is signaled from an arbiter to the system, the system can then perform more detailed diagnostics by reading out the error registers(as detailed above) in the respective arbiter . The error information captured for an SRI protocol error allows the identification of the master through the sampled master tag ID, and the final destination through the captured target address. In addition, the ERRx register captures the opcode and other control information about the transaction.

After reading all relevant error information, the error capture mechanism should be re-enabled by clearing the corresponding bits in PESTAT or TIDSTAT as appropriate.

***Notes**:*

1. *While a status bit for an error source is set in either the PESTAT or TIDSTAT register, a new error from this particular source does not generate a new trigger to the Interrupt Router (IR)*
2. *Each SRI agent in the system has two error conditions: protocol errors, and transaction ID errors. Protocol errors are classified as SM and non-SM, where as transaction ID errors are always classified as non-SM. All non-SM errors from an SRI domain share a single SRN in the interrupt router (IR). All SM errors from an SRI domain share a separate single SRN in the IR*

6.3.3.5 Indication event interactions

There are some interactions between indication events. In general, several indications from the same or a different source (arbiter, MCI, or SCI) can occur at the same time. One indication could occur several times before a service request routine services it.

The following examples describe the system behavior without re-arming from a service routine when all consecutive indications or events occur within the same domain.

Two consecutive protocol errors

The first protocol error is captured, together with the generation of an indication to the system. The second protocol error is not captured, since the registers ERR and ERRADDx are already locked, and no indication is generated.

Two consecutive transaction ID errors

The first transaction ID error generates an indication to the system. The second transaction ID error will not generate an indication to the system.

6.3.3.6 Releasing the lock from ERR and ERRADDR registers

When the ERR and ERRADDR registers are locked, the lock can be released by writing a 1 to PECONx.PEACK.

6.3.4 Quality of Service application interconnect usage observation

The QoS observation mechanism enables the system software to monitor the consumed interconnect cycles at key resources for up to 8 application partitions within a QoS time interval.

The application partitions can be defined by assigning master agent functions through its TAG-ID and PRS encoding to the application resource partition functions (ARPx).

For each ARP function:

- The system can define the usage threshold for it
- The interconnect cycles consumed by the assigned master agent functions within a QoS interval are logged
- If the consumed cycles within a QoS time interval reaches the defined usage threshold, an interrupt is signaled at the end of the interval for this ARP function, the cycles consumed until the end of the interval is stored in a capture register and the capture register is locked

***Note**: Setting the threshold value to 0 is considered illegal and should be avoided.*

QoS time interval

A QoS time interval starts as soon as QOS_ICTRL.QOSEN is enabled. At the start of a QoS time interval, QOS_ICNT.QOSICNT is initialized with the value from QOS_ICTRL.QOSINT and partition cycle counter registers are cleared and set to 0. As long as the QOS_ICTRL.QOSEN is set -QoS time intervals with the configured duration (QOS_ICTRL.QOSINT) are generated.

Measuring the cycle consumption during a QoS interval

During a QoS time interval each ARP mechanism counts for the master agent function IDs assigned to the ARP function the number of clocks the master agents occupies the observed resource within the time interval by counting the number of data phase clock cycles.

At the FPI the observed shared resource is the local FPI interconnect, at the SRI it is the SRI respective slave agent to which the ARP function belongs.

Interconnect usage includes the data phase along with wait states, where ongoing activity from a master prevents other masters from using the resource.

***Note**: The interconnect usage of CS-related transactions (with SM= 1) is only taken into account if CSAV = full or CSAV=limited.*

End of QoS interval

At the end of a QoS Interval (QOSICNT=0), the cycles consumed by the assigned master agent functions is copied into the Partition cycle count capture register of the respective ARP. For each ARP function where the interconnect cycles consumed by the assigned master agent functions is above the value configured in its Partition threshold register, an interrupt is signaled and the Partition cycle count capture register is locked after update.

The value of the locked partition cycle counter capture register is then not updated by the hardware until it is unlocked by software.

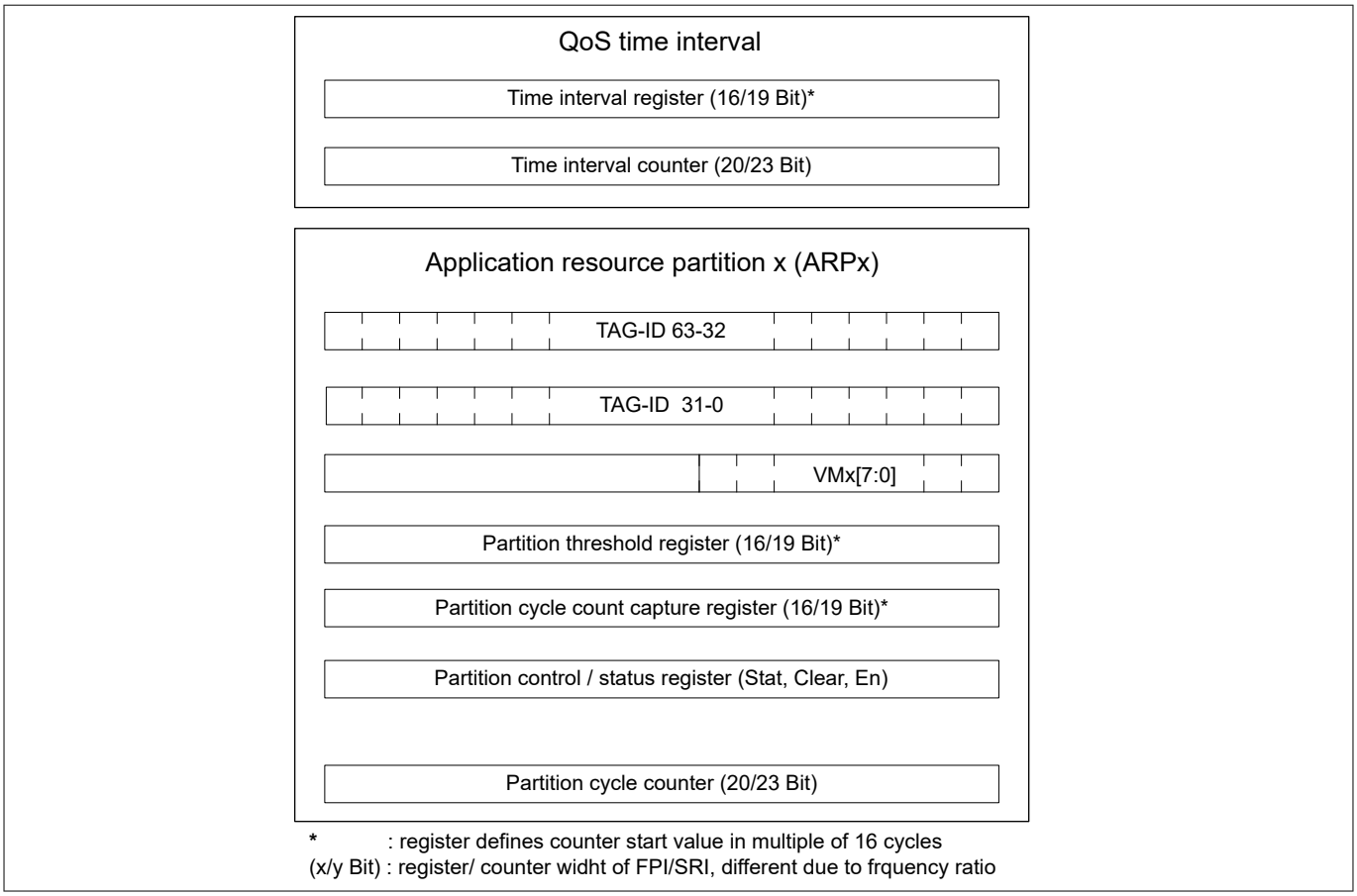


Figure 60 QoS time interval and ARPx register structure

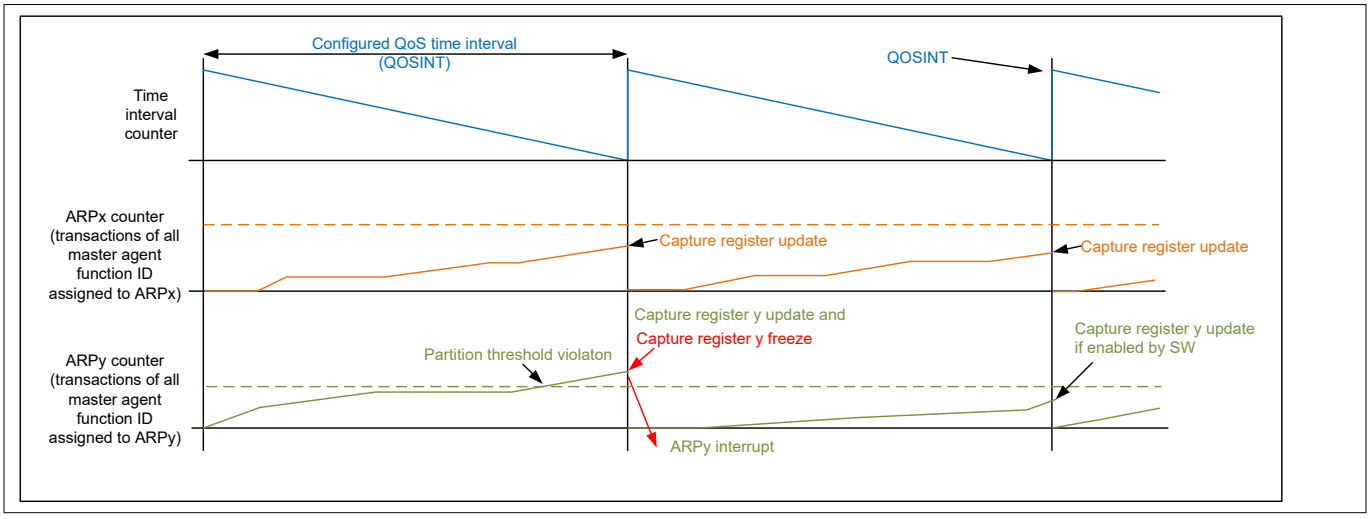


Figure 61 Temporal description of the ARP observation mechanism

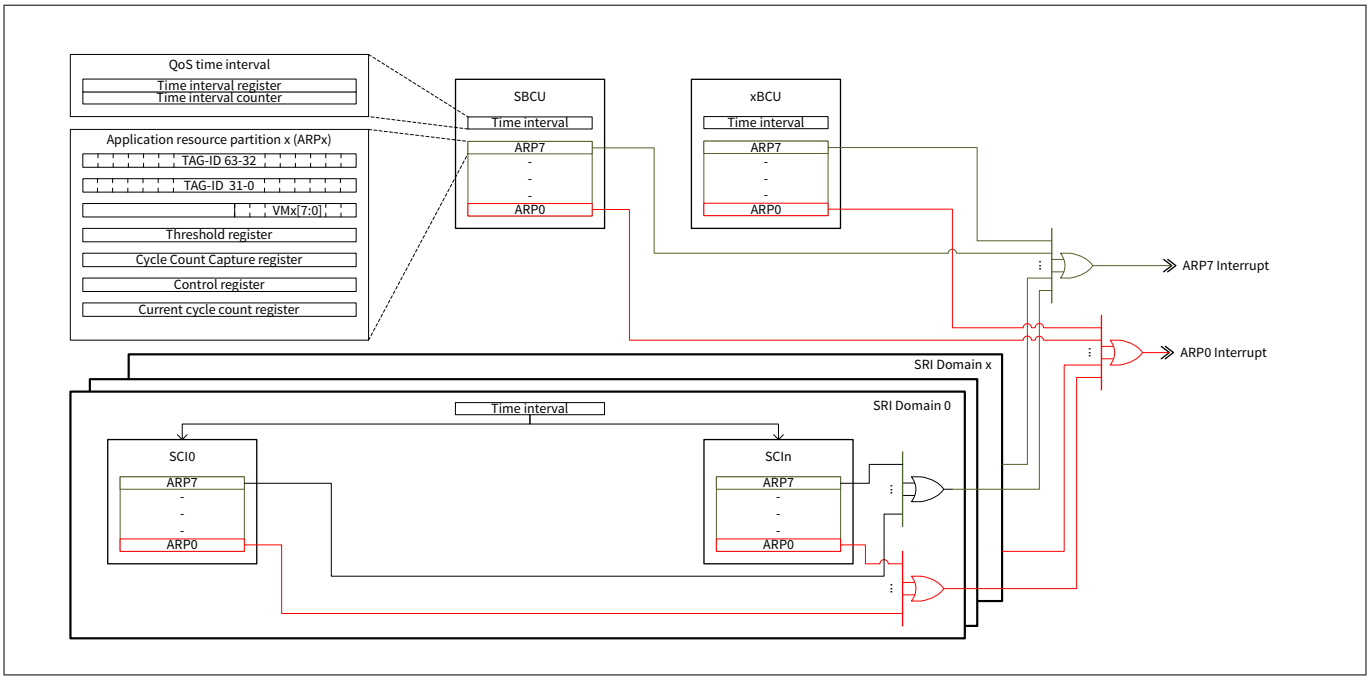


Figure 62 Hardware connection of the local ARP events to 8 ARP system interrupts

6.4 SRI to SRI bridge (S2S)

The S2S bridge facilitate master agents from one SRI domain to initiate transactions to slaves on another domain.

6.4.1 Feature list

- Bridges all transaction types from a master in one domain to a slave in another
- Implements posted writes
- Operates at fSRI frequency; with integer clock ratios of 1:1, 2:1, 1:2, between the source and destination

6.4.2 Functional overview

The S2S bridge facilitate master agents from one SRI domain to initiate transactions to slaves on another domain. The direction of a bridge refers to the request flow direction that is master to slave. Therefore to achieve connectivity between masters and slaves across two domains, two S2S bridges are required in opposite directions between the two SRI domains. The S2S bridge can carry both read and write transactions between the two domains.

S2S bridges transmit all SRI transactions transparently (without modifying either master agent function ID, transaction type including opcode or address).

6.4.3 Functional description

The S2S bridges transaction requests from masters in one domain(origin) to slaves in a target domain. It also bridges the transaction responses from the slaves in the target domain to the origin. The S2S bridge is special in its handling of errors, as it transmits most errors between the domains and does not raise alarms for them itself.

6.4.3.1 EDC errors

The S2S bridge does not perform EDC checks, instead the detection is implemented at the master or slave.

Address EDC errors are passed from the initiating domain and left to the receiving slave.

Read data with EDC errors are transferred from the slave domain to the initiating master.

Write data with EDC errors will be transferred from the initiating domain to the receiving slave.

6.4.3.2 Protocol errors

SRI read transaction protocol errors are transferred from the slave domain to the initiating master, to interrupt or raise alarms as configured.

As write transactions are posted, SRI write transaction protocol errors will not be signalled in the initiating SRI domain by the S2S bridge, as it has no knowledge of the error condition. In this instance, the receiving domain handles the error and signals an alarm.

6.4.3.3 Transaction errors for posted writes

Write transactions from one SRI domain to another SRI domain are handled as posted writes. This means that a write operation from a SRI Master through the S2S Bridge can finish on the source domain, and the S2S completes the write later on the destination SRI domain. If the S2S write transaction results in an error on the destination domain, the error information is not passed back to the source SRI domain.

This behavior occurs only for write operations over the S2S Bridge.

6.4.3.4 S2S clocking

The S2S can bridge two SRI domains which have a clock ratio of 1 or 2 between them.

6.5 FPI to SRI bridge (SFI_F2S)

The SFI_F2S bridge implements a uni-directional bridge that forwards transactions from an implementation of an FPI protocol bus to the SRI.

6.5.1 Feature list

- Bridges transactions from an FPI bus to an SRI domain
- Converts FPI transactions into one or more equivalent SRI transactions
- Signals EDC errors through system alarms
- Operates at f_{TPB} and f_{SRI} frequencies

6.5.2 Functional overview

The SFI_S2S bridge implements a uni-directional bridge that forwards transactions from an implementation of an FPI protocol bus to the SRI . The bridge supports all FPI transactions and that subset of transactions of the SRI required to implement them.

6.5.3 Functional description

The bridge is transparent for the address of a transaction and the master agent information of the FPI transaction. However, transaction types are mapped as described in the following table. For performance reasons, write transactions from FPI masters to SRI resources are handled as posted writes.

Table 579 FPI transaction to SRI transaction mapping

FPI transaction (32-bit data bus)	SRI transaction(s) (64-bit data bus)
Single Byte Transfer (8 bit)	Single Data Transfer Byte (SDTB) (8 bit)
Single Half-Word Transfer (16 bit)	Single Data Transfer Half-Word (SDTH) (16 bit)
Single Word Transfer (32 bit)	Single Data Transfer Word (SDTW) (32 bit)
2-Word Block Transfer	Single Data Transfer Double-Word (SDTD) (64 bit)
4-Word Block Transfer (aligned to 64 bit address boundary)	Block Transfer Request (BTR2) (2 transfers)
4-Word Block Transfer (starting address offset by 04_H,$0C_H$)	SDTW, SDTD, SDTW
8-Word Block Transfer (aligned to 64 bit address boundary)	Block Transfer Request (BTR4) (4 transfers)
8-Word Block Transfer (starting address offset by 04_H, 14_H)	SDTW, SDTD, BTR2, SDTW
8-Word Block Transfer (starting address offset by $0C_H$,$1C_H$)	SDTW, BTR2, SDTD, SDTW
Atomic Read Write 8/16/32 bit sequence	Atomic Read Write 8/16/32 bit sequence

6.5.4 Transaction errors for posted writes

Write transactions from FPI masters to SRI resources are handled as posted writes. This means that a write operation from an FPI master through the SFI_F2S bridge will finish on the FPI bus, and the SFI_F2S completes the write later on the SRI. If the SRI write transaction results in a SRI protocol error, the error information is not passed back to the FPI bus. The error condition will be processed by the receiving SRI slave or domain to interrupt or raise alarms as configured.

6.5.5 Data phase transaction ID errors

A transaction ID mismatch during the SRI read data phase may indicate the mis-steering of an SRI response or the presence of an uncorrectable ECC error at the source of that data. In either case, when a transaction ID error is observed on the read response to a SRI transaction type SDTx, the F2S bridge will convert that SRI read response to an FPI error.

When the read part of a read-with-intent-to-write transaction on the SRI receives such an error, the error is propagated to the FPI. As a consequence, the transaction will terminate on the FPI. The bridge will then complete the write phase on the SRI side as follows

- The data value received during the read data phase will be used as the write data
- The write data phase will be invalidated by transaction ID corruption

Such an error propagation will not happen for block transfers on SRI.

6.6 SRI to FPI bridge (SFI_S2F)

This section describes the functionality of the SFI_S2F bridge, a bridge from the SRI to an FPI based bus system.

6.6.1 Feature list

- Bridges transactions from an SRI domain to an FPI bus
- Converts SRI transactions, except for block transfers of 8 double words and atomic transactions of 64-bits, into equivalent FPI sequences
- Signals FPI EDC errors through system alarms
- Operates at f_{FPI} and f_{SRI} frequencies
  - f_{FPI} is one of f_{CPB}, f_{SPB}, f_{RCB}, f_{TPB}

6.6.2 Functional overview

The SFI_S2F bridge is implemented as a uni-directional bridge that forwards transactions from the SRI to an FPI bus. The bridge supports all SRI transactions on the SRI except for block transfers of 8 beats and atomic read write sequences of 64 bits.

6.6.3 Functional description

The bridge is transparent for the address of a transaction and the master agent information of the SRI transaction. However transactions are mapped as described in the following table. For performance reasons, write transactions from the SRI to FPI resources are handled as posted writes.

Table 580 SRI transaction to FPI transaction mapping

SRI transaction (64 bit data bus)	FPI transaction (32 bit data bus)
Single Data Transfer Byte (8 bit)	Single Byte Transfer (8 bit)
Single Data Transfer Half-Word (16 bit)	Single Half-Word Transfer (16 bit)
Single Data Transfer Word (32 bit)	Single Word Transfer (32 bit)
Single Data Transfer Double-Word (64 bit)	2-Word Block Transfer
Block Transfer Request (2 transfers)	4-Word Block Transfer
Block Transfer Request (4 transfers)	8-Word Block Transfer
Block Transfer Request (8 transfers)	Not converted - errored at the SRI interface.
Atomic Read Write 8/16/32 bit sequence	Atomic Read Write 8/16/32 bit sequence
Atomic Read Write 64 bit sequence	Not converted - errored at the SRI interface,

6.6.4 Transaction errors for posted writes

Write transactions from SRI masters to FPI resources are handled as posted writes. This means that a write operation from a SRI master through the SFI_S2F bridge can finish on the SRI, and the SFI_S2F completes the write later on the FPI. If the FPI write transaction results in an error on the FPI, the error information is not passed back to the SRI. The error condition is detected by the control logic of the FPI (BCU on the FPI), to interrupt or raise alarms as configured. This behavior occurs only for write operations through the SFI_S2F.

6.7 SRI to AXI bridge (S2A)

This section describes the functionality of the S2A bridge, a bridge from the ERAY to slaves using a 64-bit AXI interconnect.

6.7.1 Feature list

- Operates at fSRI frequency with the AXI interconnect at the same frequency
- 64-bit AXI interface support
  - All SRI transactions are converted to equivalent AXI transactions
  - Examples are used by the PCIe and PPU
- Provides PROT and ACCEN registers for the connected AXI slaves

6.7.2 Functional overview

This section describes the functionality of the S2A bridge, a bridge from the SRI to slaves on a 64-bit data AXI interconnect.

6.7.3 Functional description

The S2A bridge is implemented as a uni-directional bridge that forwards transactions from the SRI to a 64-bit data AXI interconnect. The bridge supports all SRI transactions on the SRI, and those transactions of the AXI interconnect required to implement them.

The bridge is transparent for the address of a transaction. Transactions are mapped as described in the table below. For performance reasons write transactions from the SRI to AXI resources are handled as posted writes.

***Note**: An INCR burst is a transaction of 1 or more beats where the address for each beat increments depending on the size of the transfer. A WRAP burst wraps around to a lower address if the transaction size boundary is reached.*

Table 581 SRI transaction to 64-bit data AXI transaction mapping

SRI transaction (64-bit data bus)	AXI transaction (64-bit data bus)
Single Data Transfer Byte (8 bit)	INCR Burst (Length=1, Size=1 byte)
Single Data Transfer Half-Word (16 bit)	INCR Burst (Length=1, Size=2 bytes)
Single Data Transfer Word (32 bit)	INCR Burst (Length=1, Size=4 bytes)
Single Data Transfer Double-Word (64 bit)	INCR Burst (Length=1, Size=8 bytes)
Block Transfer Request (2 transfers)	WRAP Burst (Length=2, Size=8 bytes)
Block Transfer Request (4 transfers)	WRAP Burst (Length=4, Size=8 bytes)
Block Transfer Request (8 transfers)	WRAP Burst (Length=8, Size=8 bytes)
Atomic Read Write sequence (8 bit)	INCR Burst Read followed by INCR Burst write (Length=1, Size=1 byte)[1)]

(table continues...)

Table 581 (continued) SRI transaction to 64-bit data AXI transaction mapping

SRI transaction (64-bit data bus)	AXI transaction (64-bit data bus)
Atomic Read Write sequence (16 bit)	INCR Burst Read followed by INCR Burst write (Length=1, Size=2 bytes) 1)
Atomic Read Write sequence (32 bit)	INCR Burst Read followed by INCR Burst write (Length=1, Size=4 bytes) 1)
Atomic Read Write sequence (64 bit)	INCR Burst Read followed by INCR Burst write (Length=1, Size=8 bytes) 1)

1) This sequence is not atomic on the AXI interconnect.

6.7.3.1 Transaction errors for posted writes

Write transactions from SRI masters to AXI resources are handled as posted writes. This means that a write operation from an SRI master through the S2A bridge can finish on the SRI, and the S2A completes the write later on the AXI interconnect. If the AXI write transaction results in an error on the AXI interconnect, the error information is not passed back to the SRI. The error condition is detected by the S2A bridge and an alarm is raised.

Note that this alarm behavior occurs only for write operations through the S2A bridge.

Any read or write response on the AXI interconnect which is not OKAY is an error.

6.7.3.2 Data phase EDC errors

The S2A bridge checks received write data for EDC correctness. If an EDC error is detected in the write data beats, a data phase alarm is raised. For the resulting AXI transaction the write strobes are all set disabled for the respective data beat and all remaining data beats, so that incorrect data is not written into the target slave.

If the S2A bridge receives read data from the AXI interface with an EDC error, the read transaction ID bits 7 and 6 returned to the SRI are inverted, to force an error at the receiving SRI master agent.

6.7.3.3 Parity errors

The S2A AXI interface implements parity across address and read and write data control signals. If a parity error is detected for a valid input, the S2A raises an alarm. Parity errors detected in the AXI read and write address channels result in an address phase alarm, any other parity error results in a data phase alarm.

6.7.3.4 Protection registers

The S2A provides the PROT and ACCEN registers for the connected AXI slaves. The registers are accessed from the SRI interface, and are used by the S2A to determine whether a master agent function is allowed to access the AXI target.

If an SRI master agent function is not allowed access to the target AXI slave, the transaction is acknowledged with an SRI error. One set of protection registers is provided for each slave target accessible by the S2A bridge. The registers, behavior and specific addresses for the PROT and APU registers implemented per slave are described in the individual slave module chapters accessed by an S2A bridge.

6.7.3.5 Target unavailability

If the target module is inaccessible through the AXI interface for any reason, including clock gated off or power gated off, the S2A will error all accesses to the address space allocated to the module. The SRI error indication will, as usual, raise an alarm and interrupt if enabled.

6.8 SRI to LLI bridge (S2L)

This section describes the functionality of the S2L bridge, a bridge from the SRI to the LLI.

6.8.1 Feature list

- 32-bit data AXI interface support
  - Single SRI transactions up to word size are converted to equivalent AXI transactions
- Provides PROT and ACCEN registers for connected LLI slaves
- Error injection support
- Supports SRI and LLI operating at the same frequency only

6.8.2 Functional overview

S2L is a bridge translating SRI transactions to equivalent AXI transactions for the AXI based LLI interconnect (see LLI chapter).

6.8.3 Functional description

The S2L bridge is implemented as a uni-directional bridge that forwards transactions from the SRI to the 32-bit data AXI LLI interconnect. The bridge supports all single SRI transactions on the SRI, and those transactions of the AXI interconnect required to implement them.

The bridge is transparent for the address and tag of an SRI transaction, and transactions are mapped as described in the table below. For performance reasons write transactions from the SRI to LLI resources are handled as posted writes.

Table 582 SRI transaction to 32-bit data AXI transaction mapping

SRI transaction (64-bit data bus)	AXI transaction (32-bit data bus)
Single Data Transfer Byte (8 bit)	INCR Burst (Length=1, Size=8 bits)
Single Data Transfer Half-Word (16 bit)	INCR Burst (Length=1, Size=16 bits)
Single Data Transfer Word (32 bit)	INCR Burst (Length=1, Size=32 bits)
Single Data Transfer Double-Word (64 bit)	Not converted - errored at the SRI interface.
Block Transfer Request (2 transfers)	Not converted - errored at the SRI interface.
Block Transfer Request (4 transfers)	Not converted - errored at the SRI interface.
Block Transfer Request (8 transfers)	Not converted - errored at the SRI interface.
Atomic Read Write sequence (8 bit)	INCR Burst Read followed by INCR Burst write (Length=1, Size=1 byte)[1)]
Atomic Read Write sequence (16 bit)	INCR Burst Read followed by INCR Burst write (Length=1, Size=2 bytes)[1)]
Atomic Read Write sequence (32 bit)	INCR Burst Read followed by INCR Burst write (Length=1, Size=4 bytes)[1)]
Atomic Read Write sequence (64 bit)	Not converted - errored at the SRI interface.

1) This sequence is not atomic on the AXI interconnect.

6.8.3.1 Transaction errors for posted writes

Write transactions from SRI masters to LLI resources are handled as posted writes. This means that a write operation from an SRI master through the S2L bridge will finish on the SRI, and the S2L completes the write later on the LLI interconnect. If the AXI write transaction results in an error on the LLI interconnect, the error

information is not passed back to the SRI. The error condition is detected by the S2L bridge and an alarm is raised.

Note that this alarm behavior occurs only for write operations through the S2L bridge.

Any read or write response on the AXI interconnect which is not OKAY is an error.

6.8.3.2 Data phase EDC errors

The S2L bridge checks received write data for EDC correctness. If an EDC error is detected in the write data beat, a data phase alarm is raised. For the resulting AXI transaction the write strobes are all set disabled, so that incorrect data is not written into the target slave.

6.8.3.3 Parity errors

The S2L AXI interface implements parity across address and read and write data control signals. If a parity error is detected for a valid input, the S2L raises an alarm. Parity errors detected in the AXI read and write address channels result in an address phase alarm, any other parity error results in a data phase alarm.

6.8.3.4 Protection registers

The S2L provides the PROT and ACCEN registers for the connected AXI slaves. The registers are accessed from the SRI interface, and are used by the S2L to determine whether a master agent function is allowed to access the AXI target. If an SRI master agent function is not allowed access to the target AXI slave, the transaction is acknowledged with an SRI error. One set of protection registers is provided for each slave target accessible by the S2L bridge. The registers behavior and specific addresses for the PROT and APU registers implemented per slave are described in the individual slave module chapters accessed by an S2L bridge.

6.8.3.5 Target unavailability

If the target module is inaccessible through the AXI interface for any reason, including clock gated off or power gated off, the S2L will error all accesses to the address space allocated to the module. The SRI error indication will, as usual, raise an alarm and interrupt if enabled.

6.9 AXI to SRI bridge (A2S)

This section describes the functionality of the A2S bridge, a bridge from a 64-bit AXI interconnect to the SRI.

6.9.1 Feature list

- Converts AXI transactions into one or more equivalent SRI transactions
- Signals EDC errors through system alarms
- Provides posted writes from AXI to the SRI fabric
- Supports SRI and LLI operating at the same frequency only

6.9.2 Functional overview

The A2S bridge is a uni-directional bridge that forwards transactions from a 64-bit data AXI interconnect to the SRI.

The bridge supports most transactions of the 64-bit AXI interconnect and all SRI transactions on the SRI Fabric required to implement them. A small subset of AXI transactions are not supported and receive a SLVERR error indication.

6.9.3 Functional description

The conversion of AXI read transactions which are‘regular’ is straightforward and described in the table below. In addition, the conversion of AXI write ‘regular transactions’; with the additional restriction for burst transactions that all bytes of a beat are written; is straightforward and also described in the table below. AXI transactions which are not ‘straightforward’ are described in the section non-straightforward transactions.

The A2S bridge is transparent for the address of a straightforward transaction, but not for the AxID of the AXI master. The AxID is mapped to an SRI tag in the SRI.

***Note**: For AXI write transactions, a master can use the low-order address lines to provide an unaligned start address or provide an aligned address and use the byte lane strobes to provide the unaligned start address. It is the provided address which is used for the converted transaction or first transaction of a sequence.*

Transactions are converted as described in the table below. For performance reasons write transactions from the AXI interconnect to SRI resources are handled as posted writes.

***Note**: An INCR burst is a transaction of 1 or more beats where the address for each beat increments depending on the size of the transfer. A WRAP burst wraps around to a lower address if the transaction size boundary is reached.*

Table 583 AXI ‘straightforward’ transaction to SRI transaction conversions

AXI 64-bit transaction	SRI transaction (64-bit data bus)
INCR (Length=1, Size=8 bits) (1 byte)	Single Data Transfer Byte (SDTB) (8-bit)
INCR (Length=1, Size=16 bits, Start address aligned to size) (2 bytes)	Single Data Transfer Half-Word (SDTH) (16-bit)
INCR (Length=1, Size=16 bits, Start address unaligned to size) (2 bytes)	Sequence SDTB, SDTB
INCR (Length=1, Size=32 bits, Start address aligned to size) (4 bytes)	Single Data Transfer Word (SDTW) (32-bit)
INCR (Length=1, Size=32 bits, Start address unaligned to size) (4 bytes)	Sequence SDTH, SDTH
INCR (Length=1, Size=32 bits, Start address unaligned to size) (4 bytes)	Sequence SDTB, SDTH, SDTB
INCR (Length=1, Size=64 bits, Start address unaligned to size) (8 bytes)	Single Data Transfer Double-Word (SDTD) (64-bit)
INCR (Length=1, Size=64 bits, Start address unaligned to size) (7 valid bytes)	Sequence SDTB, SDTH, SDTW
INCR (Length=1, Size=64 bits, Start address unaligned to size)(5 valid bytes)	Sequence SDTB, SDTW
INCR (Length=1, Size=64 bits, Start address unaligned to size)(3 valid bytes)	Sequence SDTB, SDTH
INCR (Length=1, Size=64 bits, Start address unaligned to size)(6 valid bytes)	Sequence SDTH, SDTW
Locked INCR Read followed by unlocked INCR write (Both length=1, both Byte/Half-Word/Word/Double-Word Aligned Transfer (1/2/4/8 bytes respectively)	Atomic Read Write SDTB/SDTH/SDTW/SDTD sequence

(table continues...)

Table 583 (continued) AXI 'straightforward' transaction to SRI transaction conversions

AXI 64-bit transaction	SRI transaction (64-bit data bus)
Locked Read Half-Word/Word/Double-Word Non-Aligned Transfer (2/4/8 bytes)	Not converted – errored at AXI interface
Locked INCR Write	Not converted – errored at AXI interface
Locked INCR or WRAP Burst (Length = 2-16)	Not converted – errored at AXI interface
INCR Burst regular = 16 byte aligned (Length=2, Size=8 bytes)	Block Transfer Request (BTR2) (2 transfers)
INCR Burst regular = 32 byte aligned (Length=4, Size=8 bytes)	Block Transfer Request (BTR4) (4 transfers)
INCR Burst regular = 64 byte aligned (Length=8, Size=8 bytes)	Block Transfer Request (BTR8) (8 transfers)
INCR Burst regular = 128 byte aligned (Length=16, Size=8 bytes)	Sequence BTR8, BTR8
WRAP Burst regular = 8 byte aligned (Length=2, Size=8 bytes)	Block Transfer Request (BTR2) (2 transfers)
WRAP Burst regular = 8 byte aligned (Length=4, Size=8 bytes)	Block Transfer Request (BTR4) (4 transfers)
WRAP Burst regular = 8 byte aligned (Length=8, Size=8 bytes)	Block Transfer Request (BTR8) (8 transfers)
WRAP Burst regular = 8 byte aligned (Length=16, Size=8 bytes)	Sequence BTR8, BTR8

6.9.3.1 Transaction errors for reads

When the A2S converts a read transaction which is errored in the SRI, it returns data beats to the AXI master with dummy read data and a read response of an error (RRESP=SLVERR).

When the A2S converts a read transaction which for a data beat receives a transaction ID error, or the A2S explicitly detects an EDC error, it returns the received (faulty) data and a read response of an error (RRESP=SLVERR) with each faulty beat or beats. In addition the A2S raises an EDC error alarm.

In addition, if the A2S in returning read data detects an EDC error in the data to be placed on the AXI interconnect, the A2S provides an error response (RRESP=SLVERR) with the faulty beat or beats of data. In addition the A2S raises an EDC error alarm.

6.9.3.2 Transaction errors for posted writes

Write transactions from AXI Masters to SRI resources are handled as posted writes. This means that a write operation from an AXI Master through the A2S Bridge will always have the data beats finished on the AXI interconnect, and the A2S completes the write later on the SRI.

If the SRI write transaction results in an error on the SRI, the error information is not passed back to the AXI interconnect. The error condition will be left to the receiving SRI slave to interrupt or raise alarms as configured.

This behavior occurs only for write operations using the A2S bridge.

If one or more of the AXI write data beats is detected as having an EDC error then the A2S returns an error to the AXI master (BRESP=SLVERR). As the transaction will be executed on the SRI, the corresponding beat in error is annotated with a transaction ID error so the slave is aware of the EDC condition.

6.9.3.3 Locked transfers

The only permitted sequence is a single locked Read of a particular size followed by a single Write of the same size. This pair of transactions occur in an uninterrupted sequence and provides a 1:1 map to the SRI atomic sequence.If the initial SRI read transaction of the read write sequence terminates on the SRI with an error, the error will be propagated to the AXI read. The counter part write transaction is still expected and will be terminated with an error and be not translated to the SRI.

The use of this locked sequence is reserved for the PPU master when it implements the EX instruction required to enable management of synchronization primitives. The use of this feature is discouraged in all other circumstances.

6.9.3.4 Parity errors

The A2S AXI interface implements parity across address and read and write data control signals. If a parity error is detected for a valid input, the A2S raises an alarm, but does not halt the transaction.

6.9.3.5 Non-straightforward transactions

AXI transactions which are not straightforward will require more than one separate SRI transactions for their conversion.

AXI transactions presented to the A2S which are not straightforward are:

- FIXED burst transactions
- INCR or WRAP bursts where the length is not 2, 4, 8, or 16, or the data in each beat is not 8-bytes
- INCR or WRAP bursts where the starting address is not naturally aligned

If the written bytes presented in any beat are discontiguous the AXI transaction will be errored and not presented on the SRI.

FIXED burst transactions of length ‘n’ will be converted to a sequence of ‘n’ element conversions. The element conversion may itself be a sequence of SRI transactions if the AXI transaction is not aligned. The element conversion will be one of the single transaction conversions described in the table “straightforward conversions”.

INCR or WRAP burst transactions will be mapped into sequences of SRI transactions which will vary in size and length depending on the length, alignment, and bytes presented during each beat. The SRI transactions resulting from the conversion will be the minimum required respecting the SRI natural alignment restrictions.

6.9.3.6 Converting AXI transaction IDs

This section describes how AXI transaction IDs are mapped onto transactions within the SRI.

The AXI interface will receive either an AwID or an ArID which identifies the originating master of the transaction. Each master agent which uses an A2S to access the SRI has an SRI master tag assigned to it, and the A2S will identify the driving master from the received ID. The A2S then drives the mapped transaction with the assigned SRI master tag.

***Note**: Most AXI masters only contain one master agent, and therefore have one tag assigned. The GETH DMA is one master, but has 9 master agents, one per read or write channel and descriptor, and therefore the A2S attached to this can drive any one of 9 SRI master tags.*

6.9.3.7 Low power capability

The AXI side of the bridge implements the low power capability as defined by the AXI3 low power interface. Only controller initiated wake up and sleep is required to be supported. An external controller can put the bridge into a low power state to isolate the AXI side from the SRI.

The clocks may be switched off after successfully entering low power mode.

6.10 GTM to SRI bridge (G2S)

The G2S bridge implements a uni-directional bridge that forwards transactions from a single beat 32-bit AXI master to the SRI.

6.10.1 Feature list

- Converts GTM AXI transactions into equivalent SRI transactions
- Signals EDC errors through system alarms
- Provides posted writes from AXI to the SRI Fabric
- Operates at f_{CPB} and f_{SRI} clock frequencies

6.10.2 Functional overview

The G2S bridge implements a uni-directional bridge that forwards transactions from the single beat 32-bit AXI master (GTM) to the SRI Fabric. The bridge supports the subset of AXI transactions that the GTM presents and that subset of transactions of the SRI required to implement them.

6.10.3 Functional description

The bridge is transparent for the address of a transaction, but not for the AxID of the AXI master. The AxID is mapped to an SRI tag in the SRI. Transactions are converted as described in the following table.

Table 584 AXI single fully enabled transaction to SRI transaction mapping

AXI transaction	SRI transaction (64-bit data bus)
Single Byte Transfer (8-bit)	Single Data Transfer Byte (8-bit)
Single Half-Word Transfer (16-bit)	Single Data Transfer Half-Word (16-bit)
Single Word Transfer (32-bit)	Single Data Transfer Word (32-bit)

6.10.3.1 Converting AXI transaction IDs

The AXI interface will receive an AXI ID which identifies the originating master of the transaction.

Each master agent which uses a G2S to access the SRI has an SRI master tag assigned to it, and the G2S will identify the driving master from the received ID. The G2S then drives the converted transaction with the assigned SRI master tag.

VM and PRS information on the SRI is populated with a 1:1 mapping from the information received on the designated AXI side user signals.

6.10.3.2 Transaction errors for posted writes

Write transactions from the GTM to SRI resources are handled as posted writes. This means that a write operation from the GTM through the G2S bridge can finish on the AXI bus, and the G2S completes the write later on the SRI.

If the SRI write transaction results in an error, the error information is not passed back to the AXI interconnect. The error condition will be left to the receiving SRI slave to interrupt or raise alarms as configured.

This behavior occurs only for write operations using the G2S bridge.

If the AXI write data beat is detected as having an EDC error then the G2S returns an error to the AXI master (BRESP=SLVERR). As the transaction will be executed on the SRI, the corresponding beat in error is annotated with a transaction ID error so the slave is aware of the EDC condition.

6.11 Set to use latency for access protection updates through S2A and S2L bridges

Additional considerations to update PROT and APU registers that are accessed through S2A and S2L bridges

- Some of the access protection checks for resources downstream of a bridge are implemented within the bridge itself
- PROT and APU register updates in the bridges can take up-to 5 clock cycles before they are used for subsequent permission checks
- When the access permissions are modified on-the-fly amidst live traffic scenarios, transactions closely following the update may be subjected to permission checks using the old configuration
- In order to avoid the ambiguous permission check around the time of the configuration change, the application software must suitably regulate the sensitive traffic through the bridge during that time

The affected slaves include eGTM.SIF, S2L, PPU.CSM, PPU.VCCM, GETH.SIF, LETH0/1.SIF, PCI0/1.SIF (not all slaves may be present in all devices).

Related information

SRI to AXI bridge (S2A) on page 1013

6.12 Registers

6.12.1 Register overview - access mode glossary

Table 585 Register overview - access mode glossary

Keyword	Description
SE	Access protection using PROT register PROTSE.
APU-P	Protection group consisting of registers ACCEN_WRA, ACCEN_WRB, ACCEN_RDA, ACCEN_RDB, ACCEN_VM, ACCEN_PRS.
P	Access protection using *APU-P* registers.
32	Access only when using 32-bit width.
SV	Access only when supervisor mode is active on the interconnect.
BE	Always returns a Bus Error.
STP	Access only during start-up, before start-up protection is set (SMM_STCON.ST_PROT=1).
U	No access restrictions.
PROT	Access restrictions as defined in the PROT register access rules.

6.12.2 Registers overview - DOM (ascending offset address)

Table 586 Registers overview - DOM (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PECONx	Protocol error control register x	00000_H+ x*20_H	32, P	P, SV, 32	Application Reset	1023
PRIORITYx	SCIx arbiter priority register	00008_H+ x*20_H	32, P	P, SV, 32	Undefined	1025
ERRADDRx	SCIx error address capture register	00010_H+ x*20_H	32, P	P, SV, 32	Application Reset	1025
ERRx	SCIx error capture register	00018_H+ x*20_H	32, P	P, SV, 32	Application Reset	1026
ID	Identification register	00408_H	32, P	BE	Application Reset	1027
PESTAT	Protocol error status register	00410_H	32, P	P, SV, 32	Application Reset	1028
TIDSTAT	Transaction ID status register	00418_H	32, P	P, SV, 32	Application Reset	1028
TIDEN	Transaction ID enable register	00420_H	32, P	P, SV, 32	Application Reset	1029
BRCON	Bridge control register	00430_H	32, P	P, SV, 32	Application Reset	1030
ACCEN_WRA	Write access enable register A	$004C0_H$	U	SV, SE	Application Reset	1030

(table continues...)

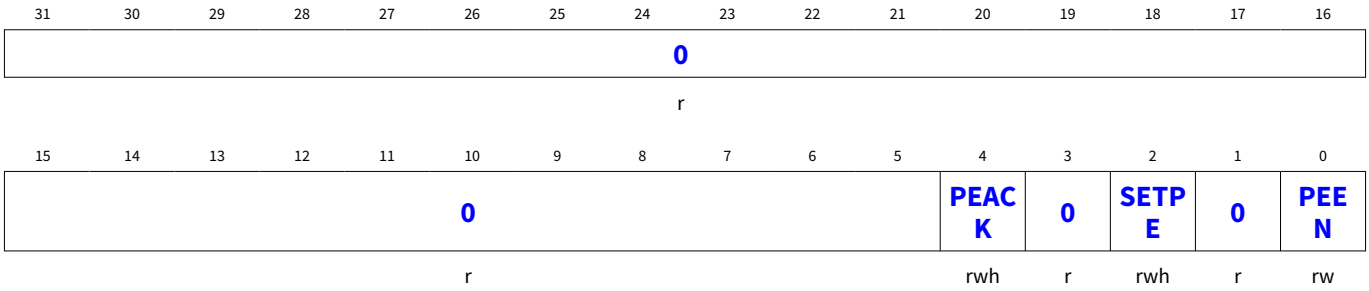
Table 586 (continued) Registers overview - DOM (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ACCEN_WRB	Write access enable register B	$004C4_H$	U	SV, SE	Application Reset	1031
ACCEN_RDA	Read access enable register A	$004C8_H$	U	SV, SE	Application Reset	1031
ACCEN_RDB	Read access enable register B	$004CC_H$	U	SV, SE	Application Reset	1032
ACCEN_VM	VM access enable register	$004D0_H$	U	SV, SE	Application Reset	1032
ACCEN_PRS	PRS access enable register	$004D4_H$	U	SV, SE	Application Reset	1033
PROTSE	Resource protection register	$004E0_H$	U	SV, PROT	Application Reset	1034
QOS_ICTRL	QoS time interval control register	01000_H	P	SV, P	Application Reset	1035
QOS_ICNT	QoS time interval counter	01004_H	P	BE	Application Reset	1036
QOSx_ARPp_TAGIDA	QOSx ARPp TAG-ID assignment register A	01020_H+ x*200_H+ p*20_H	P	P, SV	Application Reset	1036
QOSx_ARPp_TAGIDB	QOSx ARPp TAG-ID assignment register B	01024_H+ x*200_H+ p*20_H	P	SV, P	Application Reset	1037
QOSx_ARPp_VM	QOSx ARPp VM assignment register	01028_H+ x*200_H+ p*20_H	P	SV, P	Application Reset	1037
QOSx_ARPp_THR	QOSx ARPp threshold register	$0102C_H$+ x*200_H+ p*20_H	P	SV, P	Application Reset	1038
QOSx_ARPp_CTRL	QOSx ARPp control register	01030_H+ x*200_H+ p*20_H	P	SV, P	Application Reset	1039
QOSx_ARPp_CCC	QOSx ARPp cycle count capture register	01034_H+ x*200_H+ p*20_H	P	BE	Application Reset	1039
QOSx_ARPp_CCV	QOSx ARPp current counter value register	01038_H+ x*200_H+ p*20_H	P	BE	Application Reset	1040

6.12.3 Protocol error control register x

PECONx (x=0-15) Offset address: 00000_H + x*20_H

Protocol error control register x Application Reset value: 0000 0001_H



Field	Bits	Type	Description
PEEN	0	rw	**Protocol error enable** 0_B Protocol errors are not indicated and no information is captured. 1_B Protocol errors are indicated and information is captured.
SETPE	2	rwh	**Set protocol error** This allows SW to mimic a protocol error and present an indication similar to a hardware detected error. After setting this bit, it is automatically cleared by the hardware in the cycle after the write. 0_B No protocol error indication is generated 1_B A protocol error indication is generated
PEACK	4	rwh	**Protocol error acknowledge** Writing a one to this bit while it's set has the following results: The error lock of the corresponding ERRADDRx and ERRx are released, and the registers will be updated for the next protocol error detected (see chapter "Error tracking capability"). After setting this bit, it is automatically cleared by the hardware in the cycle after the write. 0_B Default value 1_B A Protocol Error for this arbiter has been indicated. The corresponding ERRADDR and ERR registers are not updated for new errors.
0	1, 3, 31:5	r	**Reserved** Read as 0; should be written with 0.

Related information

Error tracking capability on page 1005

6.12.4 SCIx arbiter priority register

Identical functionality can be achieved by setting all priorities to high compared with setting all priorities to low.

HPRS only affects arbitration when high priority requests arrive so frequently that there are no arbitration rounds with only low priority requests. When this occurs (no high priority free arbitration rounds), a maximum of HPRS grants are given to high priority masters, and then a single grant is given to the low priority round robin winner. An example configuration would be to assign the DMA to high priority and the CPUs to low priority so that the DMA does not have to wait too long for accesses and thus lose samples when moving data from RX/TX FIFOs.

***Note**: Only the MCIn_P bits assigned to implemented MCIs are present. Bits corresponding to unimplemented MCIs are treated as reserved bits.*

Reset values are device and instance specific. Refer to the device specific register table for reset values.

PRIORITYx (x=0-15) Offset address: 00008_H+x*20_H

SCIx arbiter priority register Reset value: Undefined

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0												HPRS			
r												rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MCI15_P	MCI14_P	MCI13_P	MCI12_P	MCI11_P	MCI10_P	MCI9_P	MCI8_P	MCI7_P	MCI6_P	MCI5_P	MCI4_P	MCI3_P	MCI2_P	MCI1_P	MCI0_P
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

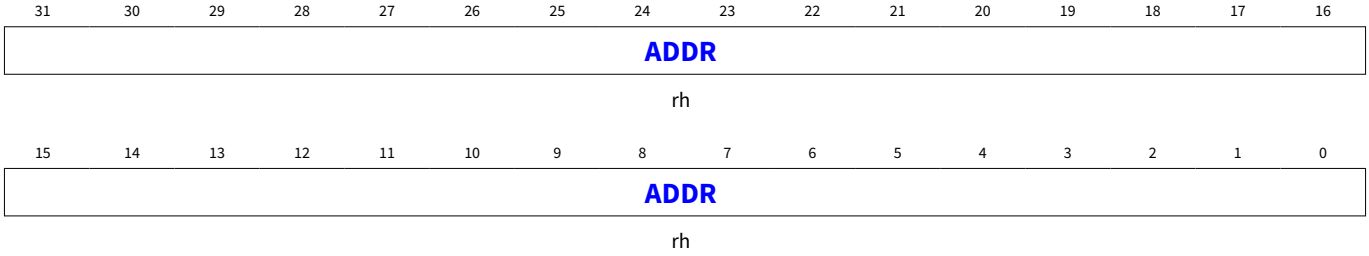
Field	Bits	Type	Description
MCIn_P (n=0-15)	n	rw	**MCIn Priority** 0_B MCIn requests are arbitrated in the low priority round robin. 1_B MCIn requests are arbitrated in the high priority round robin.
HPRS	19:16	rw	**High priority round share** Number of transactions to give to the high priority round robin before a transaction from low priority round (when request saturated). This number may not be less than the number of high priority MCI programmed via. MCIn_P.
0	31:20	r	**Reserved** Read as 0; should be written with 0.

6.12.5 SCIx error address capture register

***Note**: The default value can differ from the one shown here, because a constant can be used to reduce the number of compared bits in the arbitration if a slave occupies only a limited address area. For more details, see the design specification of the SRI crossbar.*

ERRADDRx (x=0-15) Offset address: 00010_H+x*20_H

SCIx error address capture register Application Reset value: $0000\ 0000_H$

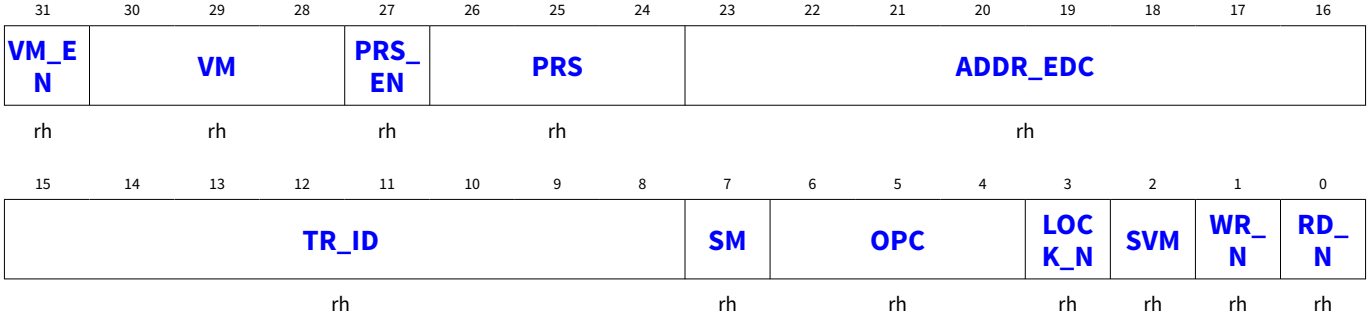


Field	Bits	Type	Description
ADDR	31:0	rh	**Transaction address** This bitfield contains the address of the erroneous transaction from the address phase

6.12.6 SCIx error capture register

ERRx (x=0-15) Offset address: 00018_H+x*20_H

SCIx error capture register Application Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
RD_N	0	rh	**Read status - RD** 0_B The read signal line was asserted (read or start of RMW transaction) 1_B The read signal line was deasserted (no read transaction)
WR_N	1	rh	**Write status - WR** 0_B The write signal line was asserted (write or start of RMW transaction) 1_B The write signal line was deasserted (no write transaction)
SVM	2	rh	**Supervisor mode status** 0_B The supervisor mode signal line was deasserted 1_B The supervisor mode signal line was asserted
LOCK_N	3	rh	**Lock status - LK** 0_B The lock signal line was asserted 1_B The lock signal line was deasserted

(table continues...)

(continued)

Field	Bits	Type	Description
OPC	6:4	rh	**Operation code** This field contains bits 3, 1 and 0 concatenated of the opcode of the erroneous transaction. Bit 2 is unused for valid opcodes.
SM	7	rh	**Special master** 0_B The special master signal line was deasserted 1_B The special master signal line was asserted
TR_ID	15:8	rh	**Transaction ID** This field contains the Master's transaction ID of the erroneous transaction. The Transaction ID is built out of the Master's 6-bit unique TAG ID (TR_ID[5:0]), and a 2-bit running number TR_ID[7:6].
ADDR_EDC	23:16	rh	**Address phase error detection information** This field contains the Address Phase Error Detection Information of the erroneous transaction.
PRS	26:24	rh	**Protection register set** This field contains the Master's PRS number of the erroneous transaction
PRS_EN	27	rh	**PRS enable** This field contains the Master's PRS enable of the erroneous transaction
VM	30:28	rh	**Virtual machine** This field contains the Master's Virtual Machine number of the erroneous transaction
VM_EN	31	rh	**VM Enable** This field contains the Master's VM enable of the erroneous transaction

6.12.7 Identification register

The identification register identifies the module, and provides revision and type information. The table below describes the identification register, which is implemented per domain.

ID Offset address: 00408_H

Identification register Application Reset value: $0004\ D006_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MOD_NUM															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MOD_TYPE								MOD_REV							
r								r							

Field	Bits	Type	Description
MOD_REV	7:0	r	**Module revision number** This bit field defines the module revision number. The value of a module revision starts with 01H (first revision).
MOD_TYPE	15:8	r	**Module type** The bit field is set to D0H which defines the module as a 64-bit module.
MOD_NUM	31:16	r	**Module number value** This bit field defines a module identification number. The value for the SRI SRI Fabric is 0004H.

6.12.8 Protocol error status register

***Note**: Only the bits assigned to implemented SCIs are present. Bits corresponding to unimplemented SCIs are treated as reserved bits, which will be read as 0, and should be written with 0.*

PESTAT Offset address: 00410_H

Protocol error status register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PESCI15	PESCI14	PESCI13	PESCI12	PESCI11	PESCI10	PESCI9	PESCI8	PESCI7	PESCI6	PESCI5	PESCI4	PESCI3	PESCI2	PESCI1	PESCI0
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															
r															

Field	Bits	Type	Description
PESCIn (n=0-15)	n+16	rwh	**Protocol error status of SCIn** Writing 0 to the bit leaves the content unchanged, while writing 1 to the bit clears it. In case the bit is simultaneously cleared via software and set via hardware, the bit remains set and is not cleared. 0_B No protocol error has been indicated by SCIn 1_B A protocol error has been indicated by SCIn
0	15:0	r	**Reserved** Read as 0; should be written with 0.

6.12.9 Transaction ID status register

Only the bits assigned to implemented SCIs and MCIs are present. Bits corresponding to unimplemented SCIs and MCIs are treated as reserved bits, which will be read as 0, and should be written with 0. Reset values for bits/bit fields referencing MCIs or SCIs that are not implemented or enabled are 0.

TIDSTAT Offset address: 00418_H

Transaction ID status register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TIDMCI15	TIDMCI14	TIDMCI13	TIDMCI12	TIDMCI11	TIDMCI10	TIDMCI9	TIDMCI8	TIDMCI7	TIDMCI6	TIDMCI5	TIDMCI4	TIDMCI3	TIDMCI2	TIDMCI1	TIDMCI0
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TIDSCI15	TIDSCI14	TIDSCI13	TIDSCI12	TIDSCI11	TIDSCI10	TIDSCI9	TIDSCI8	TIDSCI7	TIDSCI6	TIDSCI5	TIDSCI4	TIDSCI3	TIDSCI2	TIDSCI1	TIDSCI0
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
TIDSCIn (n=0-15)	n	rwh	**Transaction ID error from SCIn status** Writing 0 to the bit has no effect. Writing 1 to the bit clears it. ***Note**: If the bit is simultaneously cleared via software and set due to a hardware error, the bit remains set and is not cleared.* 0_B No transaction ID error has been indicated by SCIn 1_B A transaction ID error has been indicated by SCIn
TIDMCIn (n=0-15)	n+16	rwh	**Trasaction ID Error from MCIn status** Writing 0 to the bit has no effect. Writing 1 to the bit clears it. ***Note**: If the bit is simultaneously cleared via software and set due to a hardware error, the bit remains set and is not cleared.* 0_B No transaction ID error has been indicated by MCIn 1_B A transaction ID error has been indicated by MCIn

6.12.10 Transaction ID enable register

***Note**: Only the bits assigned to implemented SCIs and MCIs are present. Bits corresponding to unimplemented SCIs and MCIs are treated as reserved bits, which will be read as ´0´, and should be written with ´0´.*

Reset values for bits/bit fields referencing MCIs or SCIs that are not implemented or enabled are zero.

TIDEN Offset address: 00420_H

Transaction ID enable register Application Reset value: $FFFF\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ENMCI15	ENMCI14	ENMCI13	ENMCI12	ENMCI11	ENMCI10	ENMCI9	ENMCI8	ENMCI7	ENMCI6	ENMCI5	ENMCI4	ENMCI3	ENMCI2	ENMCI1	ENMCI0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ENSCI15	ENSCI14	ENSCI13	ENSCI12	ENSCI11	ENSCI10	ENSCI9	ENSCI8	ENSCI7	ENSCI6	ENSCI5	ENSCI4	ENSCI3	ENSCI2	ENSCI1	ENSCI0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENSCIn (n=0-15)	n	rw	**Enable transaction ID error from SCIn** 0_B Transaction ID errors from SCIn are not indicated 1_B Transaction ID errors from SCIn are indicated
ENMCIn (n=0-15)	n+16	rw	**Enable transaction ID error from MCIn** 0_B Transaction ID errors from MCIn are not indicated 1_B Transaction ID errors from MCIn are indicated

6.12.11 Bridge control register

BRCON Offset address: 00430_H

Bridge control register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															OLDAEN
r															rw

Field	Bits	Type	Description
OLDAEN	0	rw	**Online data acquisition enable** This bit is used to control trap generated for write accesses to the OLDA address range associated with this domain. 0_B Trap generated on a write access to the OLDA memory range. 1_B No trap generated on a write access to the OLDA memory range.
0	31:1	r	**Reserved** Read as 0; should be written with 0.

6.12.12 Write access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

ACCEN_WRA Offset address: $004C0_H$

Write access enable register A Application Reset value: $1000\ 0003_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Write access enable for TAG ID q** This bit enables write access to the access protected region for transactions with the TAG ID q. 0_B Disabled for write access 1_B Enabled for write access

6.12.13 Write access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

ACCEN_WRB Offset address: $004C4_H$

Write access enable register B Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN63	EN62	EN61	EN60	EN59	EN58	EN57	EN56	EN55	EN54	EN53	EN52	EN51	EN50	EN49	EN48
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN47	EN46	EN45	EN44	EN43	EN42	EN41	EN40	EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-63)	q-32	rw	**Write access enable for TAG ID q** This bit enables write access to the access protected region for transactions with the TAG ID q. 0_B Disabled for write access 1_B Enabled for write access

6.12.14 Read access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

ACCEN_RDA Offset address: $004C8_H$

Read access enable register A Application Reset value: $FFFF\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Read access enable for TAG ID q** This bit enables read access to the access protected region for transactions with the TAG ID q. 0_B Disabled for read access 1_B Enabled for read access

6.12.15 Read access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

ACCEN_RDB Offset address: 004CC$_H$

Read access enable register B Application Reset value: FFFF FFFF$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN63	EN62	EN61	EN60	EN59	EN58	EN57	EN56	EN55	EN54	EN53	EN52	EN51	EN50	EN49	EN48
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN47	EN46	EN45	EN44	EN43	EN42	EN41	EN40	EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-63)	q-32	rw	**Read access enable for TAG ID q** This bit enables read access to the access protected region for transactions with the TAG ID q. 0_B Disabled for read access 1_B Enabled for read access

6.12.16 VM access enable register

This register provides separate controls to enable or disable read and write accesses to registers for transactions with VM corresponding to the numbered bits. Bits WR0 to WR7 control write access and bits RD0 to RD7 control read access.

ACCEN_VM Offset address: 004D0$_H$

VM access enable register Application Reset value: 00FF 00FF$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for VM ID q** This bit enables read access to the access protected region for transactions with the VM ID q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for VM ID q** This bit enables write access to the access protected region for transactions with the VM ID q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

6.12.17 PRS access enable register

This register provides separate controls to enable or disable read and write accesses to registers for transactions with PRS corresponding to the numbered bits. Bits WR0 to WR7 control write access and bits RD0 to RD7 control read access.

ACCEN_PRS Offset address: $004D4_H$

PRS access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for PRS q** This bit enables read access to the access protected region for transactions in the PRS q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for PRS q** This bit enables write access to the access protected region for transactions in the PRS q. 0_B Disabled for write access Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

6.12.18 Resource protection register

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner and in some PROT states, the secure master to update the PROT state as described in the PROT mechanism.

PROTSE Offset address: $004E0_H$

Resource protection register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OWEN	ODEF	TAGID						PRSEN	PRS			VMEN	VM		
w	rw	rw						rw	rw			rw	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												SWEN	STATE		
r												w	rwh		

Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
VM	18:16	rw	**Virtual Machine definition for PROT owner** Defines the VM portion of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual Machine definition Enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection Set definition for PROT owner** Defines the PRS portion of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.
PRSEN	23	rw	**Protection Set definition Enable for PROT owner** 0_B **Disable**: PRS is part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition

(table continues...)

(continued)

Field	Bits	Type	Description
TAGID	29:24	rw	**TAG ID definition for PROT owner** Defines the TAGID portion of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner
OWEN	31	w	**Owner write enable** 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; should be written with 0.

Table 587 Access mode restrictions of PROTSE sorted by descending priority

Mode name	Access mode		Description
write 1 to .**SWEN**	rwh	STATE	
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

6.12.19 QoS time interval control register

The QOSICTRL register allows the enabling and disabling of the QoS function and the defining of the QoS time interval for this interconnect.The four lower order bits are reserved and always zero so that the register as a whole provides the time interval at a granularity of 16 clocks.

QOS_ICTRL Offset address: 01000_H

QoS time interval control register Application Reset value: $0007\ FFF0_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
QOSEN	0								QOSINT						
rw	r								rw						

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
QOSINT												0			
rw												r			

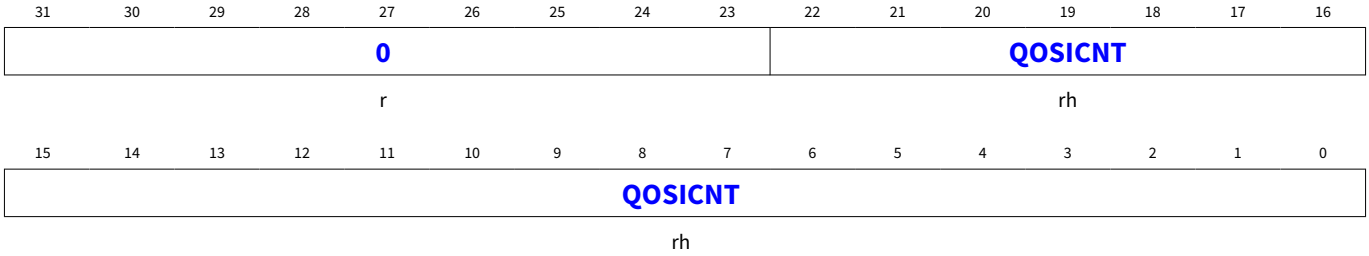
Field	Bits	Type	Description
QOSINT	22:4	rw	**QoS time interval** The bit field determines the QoS time interval in multiples of 16 local interconnect clock cycles. The lower order reserved bits within the register account for the 16 cycle granularity.
QOSEN	31	rw	**QoS enable** 0_B QoS function is disabled for the local interconnect 1_B QoS function is enabled for the local interconnect
0	3:0, 30:23	r	**Reserved** Read as 0; should be written with 0.

6.12.20 QoS time interval counter

Indicates the current value of the QoS time interval counter.

QOS_ICNT Offset address: 01004_H

QoS time interval counter Application Reset value: $007F\ FFFF_H$



Field	Bits	Type	Description
QOSICNT	22:0	rh	**QoS time interval counter** The bit field shows the current values of the QoS time interval counter. In QoS mode, the decrementing counter indicates the number of local clock cycles before the current time interval has expired. when the counter is counted down to 0, it is reloaded with the counter value at 16 cycle granularity as derived from QOS_ICTRL.
0	31:23	r	**Reserved** Read as 0; should be written with 0.

6.12.21 QOSx ARPp TAG-ID assignment register A

QOSx_ARPp_TAGIDA (p=0-7;x=1-15) Offset address: $01020_H+x*200_H+p*20_H$

QOSx ARPp TAG-ID assignment register A Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**TAG-ID q** This bit enables usage monitoring of master agent TAG-ID q 0_B TAG-ID not assigned to ARPp 1_B TAG-ID assigned to ARPp

6.12.22 QOSx ARPp TAG-ID assignment register B

QOSx_ARPp_TAGIDB (p=0-7;x=1-15) Offset address: $01024_H+x*200_H+p*20_H$

QOSx ARPp TAG-ID assignment register B Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN63	EN62	EN61	EN60	EN59	EN58	EN57	EN56	EN55	EN54	EN53	EN52	EN51	EN50	EN49	EN48
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

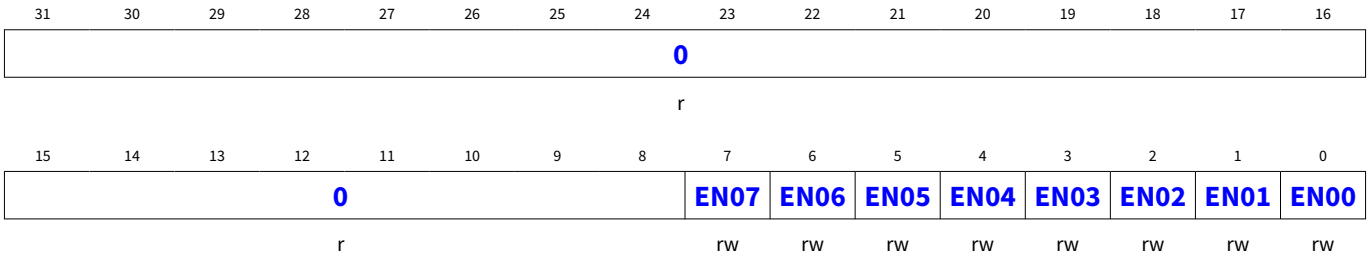
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN47	EN46	EN45	EN44	EN43	EN42	EN41	EN40	EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-63)	q-32	rw	**TAG-ID q** This bit enables usage monitoring of master agent TAG-ID q 0_B TAG-ID not assigned to ARPp 1_B TAG-ID assigned to ARPp

6.12.23 QOSx ARPp VM assignment register

QOSx_ARPp_VM (p=0-7;x=1-15) Offset address: $01028_H+x*200_H+p*20_H$

QOSx ARPp VM assignment register Application Reset value: $0000\ 0000_H$



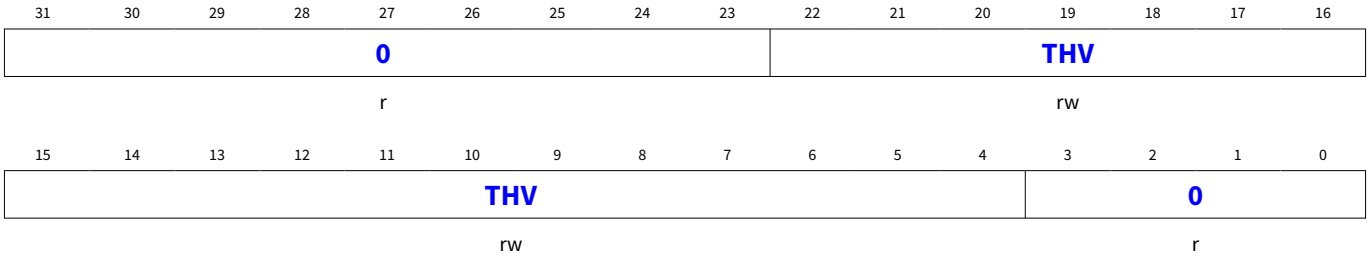
Field	Bits	Type	Description
ENq (q=00-07)	q	rw	**VM-ID q** This bit enables usage monitoring of master agent VM-ID q 0_B VM-ID not assigned to ARPp 1_B VM-ID assigned to ARPp
0	31:8	r	**Reserved** Read as 0; should be written with 0.

6.12.24 QOSx ARPp threshold register

Register defines the interconnect usage budget for the master agent functions mapped to ARPp. The four lower order bits are reserved and always read zero so that the register as a whole defines the cycle count budget at a granularity of 16 clocks.

QOSx_ARPp_THR (p=0-7;x=1-15) Offset address: $0102C_H+x*200_H+p*20_H$

QOSx ARPp threshold register Application Reset value: $0007\ FFF0_H$



Field	Bits	Type	Description
THV	22:4	rw	**ARPp threshold value** Bit field defines how many clock cycles the master agent functions assigned to ARPp can use the interconnect in the QoS time interval before it is identified as an ARPp usage violation. The bit field, along with the reserved lower order bits, defines the number in multiples of 16 local interconnect clock cycles.
0	3:0, 31:23	r	**Reserved** Read as 0; should be written with 0.

6.12.25 QOSx ARPp control register

Register provides control for the ARPp function and indicates the status of ARPp.

QOSx_ARPp_CTRL (p=0-7;x=1-15) Offset address: 01030_H+x*200_H+p*20_H

QOSx ARPp control register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
STAT	CLR	0													
rh	w	r													

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															EN
r															rw

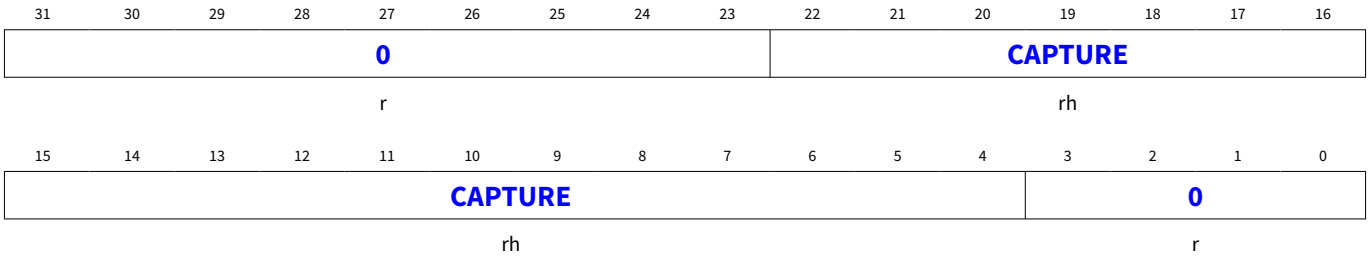
Field	Bits	Type	Description
EN	0	rw	**ARPp enable** 0_B ARPp function is disabled 1_B ARPp function is enabled
CLR	30	w	**ARPp status clear** Bit can be used to clear the ARPp status bit (STAT). Bit is always read as 0. 0_B No action 1_B STAT bit is cleared.
STAT	31	rh	**ARPp status** The STAT bit is set by the ARPp function at the end of a QoS interval if an ARPp interconnect usage violation was identified in the QoS interval. If the STAT bit is not already set at that time, the CAPTURE register is simultaneously updated and its contents are protected against being overwritten by the hardware until the STAT bit is reset by the application software. 0_B This partition has not had a usage violation 1_B This partition has had a usage violation
0	29:1	r	**Reserved-0** Read as 0; should be written with 0.

6.12.26 QOSx ARPp cycle count capture register

Register shows the interconnect cycles used by the master agent functions mapped to ARPp in a QoS interval. The four lower order bits are reserved and always read zero so that the register as a whole provides the cycle count usage at a granularity of 16 clocks.

QOSx_ARPp_CCC (p=0-7;x=1-15) Offset address: 01034_H+x*200_H+p*20_H

QOSx ARPp cycle count capture register Application Reset value: $0000\ 0000_H$



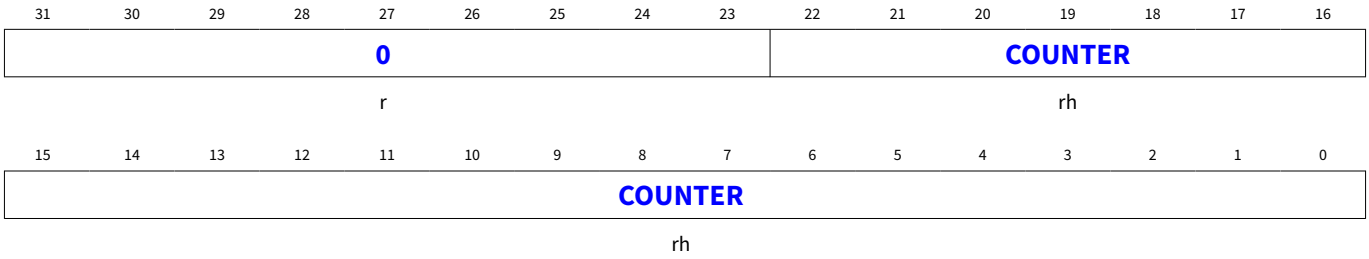
Field	Bits	Type	Description
CAPTURE	22:4	rh	**ARPp captured cycle count** Bit field indicates how many clock cycles the master agent functions assigned to the ARPp have used the interconnect in a QoS time interval. The register is updated at the end of a QoS time interval if the CTRL.STAT bit is not set. The bit field, along with the four lower order reserved bits from the register, provide the result in multiples of 16 local interconnect clock cycles.
0	3:0, 31:23	r	**Reserved-0** Read as 0; should be written with 0.

6.12.27 QOSx ARPp current counter value register

Register shows the state of the ARPp counter with which the clocks are counted in which ARPp master agent functions have used the interconnect in the current QoS interval.

QOSx_ARPp_CCV (p=0-7;x=1-15) Offset address: 01038_H+x*200_H+p*20_H

QOSx ARPp current counter value register Application Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
COUNTER	22:0	rh	**ARPp cycle count** Bit field shows the current state of the ARPp clock cycle counter and thus how many clock cycles the master agent functions assigned to the ARPp have used the interconnect so far in the current QoS time interval. Note: the ARPp clock cycle counter counts only if ARPp is enabled.
0	31:23	r	**Reserved-0** Read as 0; should be written with 0.

6.13 Debug information

This section is not applicable for the module being described.

6.14 References

1. SRI Protocol Specification v5.2

6.15 SRI revision history

Initial release of the chapter.

6.16 TC4Dx SRI information

6.16.1 TC4Dx SRI configuration

There are no deviations from the generic specification.

6.16.2 TC4Dx SRI features

There are no deviations from the generic specification.

6.16.3 TC4Dx SRI functional description

6.16.3.1 Set to use latency for access protection updates through S2A and S2L bridges

Additional considerations to update PROT and APU registers that are accessed through S2A and S2L bridges

- Some of the access protection checks for resources downstream of a bridge are implemented within the bridge itself
- PROT and APU register updates in the bridges can take up-to 5 clock cycles before they are used for subsequent permission checks
- When the access permissions are modified on-the-fly amidst live traffic scenarios, transactions closely following the update may be subjected to permission checks using the old configuration
- In order to avoid the ambiguous permission check around the time of the configuration change, the application software must suitably regulate the sensitive traffic through the bridge during that time

The affected slaves include eGTM.SIF, S2L, PPU.CSM, PPU.VCCM, GETH.SIF, LETH0/1.SIF, PCI0/1.SIF (not all slaves may be present in all devices).

Related information

SRI to AXI bridge (S2A) on page 1013

6.16.3.2 TC4Dx OLDA address space restrictions

The following restriction applies to the 512KB OLDA region

- OLDA region is split into 32 16KB sections
- In each 16KB section, 64B starting at offset 0x4C0 is not available for OLDA and writes should not be targeted to this region

Related information

Online data acquisition (OLDA) on page 999

6.16.3.3 TC4Dx incorrect write completion status from S2L following RMW write address EDC error

For SRI read with intent to write (RMW) transaction types, which are often the result of atomic CPU instructions , a soft error could affect the address phase of the write. On detecting such errors, the S2L will generate an alarm to signal an address phase EDC error to the SMU. The detection of the address phase EDC error terminates the transaction. Such terminated transaction should also be indicated as complete by the S2L. However, for the specific case as discussed above, the S2L would not indicate the completion of this transfer

back to the initiating master. Therefore, if a DSYNC appears in the instruction stream following the above soft error scenario, the CPU pipeline could stall indefinitely (which would eventually be detected by the expiration of a watchdog timer). Such a DSYNC may also appear within the interrupt service routine which is a direct response to the SMU alarm. If the ISR is executed on the same CPU as the one which triggered the S2L alarm, the ISR and therefore alarm reaction may not complete.

The recommendation is to handle S2L address phase EDC errors by invoking an application reset, or to execute the associated SMU initiated ISR on a CPU that does not normally access the S2L. Application reset is guaranteed to reinitialize the completion status at the S2L and also recover stalled CPUs.

6.16.3.4 TC4Dx transactions closely following a RMW may break strict round robin arbitration

For transaction sequences closely following RMW transactions as observed at the SRI crossbar arbiter, the arbiter might unfairly grant access to the same master even though other higher priority masters are pending. The exact sequence depends on the internal state of the arbiter and in the worst case the arbiter unfairly grants two additional transactions.

Therefore, when the arbiters witness a high density of RMW, the instantaneous worst case arbitration delay of other participating masters would be extended by up-to two arbitration delay slots, or the equivalent clock cycles.

Related information

Request latency on page 1001

6.16.3.5 Legend to connectivity matrix

Table 588 Legend to connectivity matrix

Name	Guide to naming convention
<FB>.MIF<*n*>	n^{th} master port instance belonging to the functional block(FB) where the FB has more than one master port. It is omitted when there is a single mater port.
<FB>.<named_port>	Where a unique name is associated with a FB master or slave, such as PPU.CBU, or PPU.VMEM, CPU.SSF(alternately P, S_SRI1), or CPU.SSX (alternately S, S_SRI0), for example.
<FB>.SIF<*n*>	n^{th} slave port instance belonging to the functional block(FB) where the FB has more than one slave port. It may be omitted when there is a single slave port, or alternately identified with n as *0*.
S2S_<*i*><*j*>_<*k*>	An S2S bridge from domain *i* to domain *j*. *k* stands for the instance of the bridge for the specified connection. *k* stands for the instance of the bridge when multiple bridges exist. *k* is normally 0, or only one bridge exists.
S2F_<*i*><named_fpi>_<*k*>	An SRI to FPI bridge from domain *i* to one of the FPI domains. *k* stands for the instance of the bridge. *k* stands for the instance of the bridge when multiple bridges exist. *k* is normally 0, or only one bridge exists.
F2S_<named_fpi><*j*>_<*k*>	An FPI to SRI bridge from one of the FPI domains to domain *j*. *k* stands for the instance of the bridge when multiple bridges exist. *k* is normally 0, or only one bridge exists.

6.16.3.6 SRI crossbar 0 MCI to SCI connectivity

r/w MCI has read write connectivity to SCI
r.o MCI has only read connectivity to SCI
x MCI has no connectivity to SCI
\+ SCI has QoS Observation

		DEFS	S2S_03_0	CPU0.SSF	CPU0.SSX	CPU1.SSF	CPU1.SSX	CPU2.SSF	CPU2.SSX	LMU0	LMU1	S2S_01_0	S2S_02_0	S2S_06_0	S2S_07_0	CSS
		SCI0	SCI1	SCI2	SCI3	SCI4	SCI5	SCI6	SCI7	SCI8	SCI9	SCI10	SCI11	SCI12	SCI13	SCI14
			+		+		+		+	+	+	+	+	+	+	
SDMA0.MIF0	MCI0	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	x	x	r/w
CPU0	MCI1	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w
CPU1	MCI2	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w
CPU2	MCI3	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w
S2S_10_0	MCI4	r/w	x	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	x	x	x	x	r/w
S2S_20_0	MCI5	r/w	x	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	x	x	x	x	r/w
SDMA1.MIF1	MCI6	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	x	r/w	x	x	r/w
GETH.MIF0	MCI7	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w
S2S_30_0	MCI8	r/w	x	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	x	r/w	x	x	r/w

Figure 63 Domain 0 connectivity

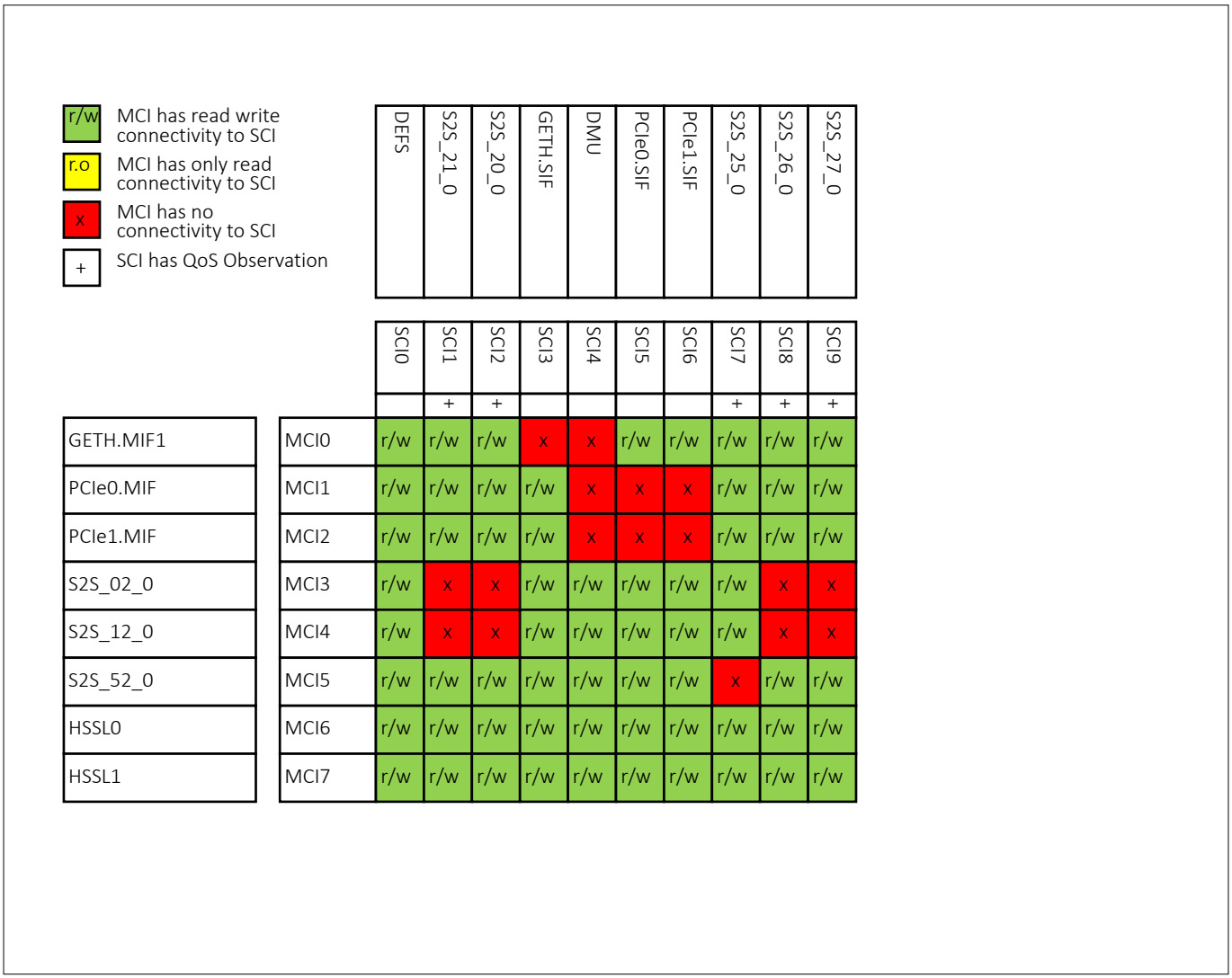
6.16.3.7 SRI crossbar 1 MCI to SCI connectivity

r/w MCI has read write connectivity to SCI
r.o MCI has only read connectivity to SCI
x MCI has no connectivity to SCI
\+ SCI has QoS Observation

		DEFS	CPU3.SSF	CPU3.SSX	CPU4.SSF	CPU4.SSX	CPU5.SSF	CPU5.SSX	LMU2	LMU3	S2S_10_0	S2S_13_0	S2S_12_0	S2S_16_0	S2S_17_0
		SCI0	SCI1	SCI2	SCI3	SCI4	SCI5	SCI6	SCI7	SCI8	SCI9	SCI10	SCI11	SCI12	SCI13
				+		+		+	+	+	+	+	+	+	+
CPU3	MCI0	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w
CPU4	MCI1	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w
CPU5	MCI2	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w
SDMA0.MIF1	MCI3	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	x	r/w	r/w	x	x
SDMA1.MIF0	MCI4	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	x	x
S2S_01_0	MCI5	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	x	x	x	x	x
S2S_21_0	MCI6	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	x	r/w	x	x	x
S2S_31_0	MCI7	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	x	x	x	x	x

Figure 64 Domain 1 connectivity

6.16.3.8 SRI crossbar 2 MCI to SCI connectivity



		DEFS	S2S_21_0	S2S_20_0	GETH.SIF	DMU	PCIe0.SIF	PCIe1.SIF	S2S_25_0	S2S_26_0	S2S_27_0
		SCI0	SCI1	SCI2	SCI3	SCI4	SCI5	SCI6	SCI7	SCI8	SCI9
			+	+					+	+	+
GETH.MIF1	MCI0	r/w	r/w	r/w	x	x	r/w	r/w	r/w	r/w	r/w
PCIe0.MIF	MCI1	r/w	r/w	r/w	r/w	x	x	x	r/w	r/w	r/w
PCIe1.MIF	MCI2	r/w	r/w	r/w	r/w	x	x	x	r/w	r/w	r/w
S2S_02_0	MCI3	r/w	x	x	r/w	r/w	r/w	r/w	r/w	x	x
S2S_12_0	MCI4	r/w	x	x	r/w	r/w	r/w	r/w	r/w	x	x
S2S_52_0	MCI5	r/w	r/w	r/w	r/w	r/w	r/w	r/w	x	r/w	r/w
HSSL0	MCI6	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w
HSSL1	MCI7	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

Figure 65 Domain 2 connectivity

6.16.3.9 SRI crossbar 3 MCI to SCI connectivity

r/w	MCI has read write connectivity to SCI
r.o	MCI has only read connectivity to SCI
x	MCI has no connectivity to SCI
+	SCI has QoS Observation

		DEFS	S2S_31_0	S2S_30_0	S2S_34_0	PPU.CSM	PPU.VCCM	S2L	eGTM	S2F_3SPB_0	S2F_3COMPB_0	S2S_36_0	S2S_37_0
		SCI0	SCI1	SCI2	SCI3	SCI4	SCI5	SCI6	SCI7	SCI8	SCI9	SCI10	SCI11
			+	+	+	+	+					+	+
PPU.STU	MCI0	r/w	r/w	r/w	r/w	r/w	r/w	x	r/w	r/w	r/w	r/w	r/w
PPU.CBU	MCI1	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w
S2S_13_0	MCI2	r/w	x	x	r/w	r/w	r/w	r/w	r/w	r/w	r/w	x	x
S2S_03_0	MCI3	r/w	x	x	r/w	r/w	r/w	r/w	r/w	r/w	r/w	x	x
S2S_43_0	MCI4	r/w	r/w	r/w	x	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w
S2S_73_0	MCI5	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	x	x

Figure 66 Domain 3 connectivity

6.16.3.10 SRI crossbar 4 MCI to SCI connectivity

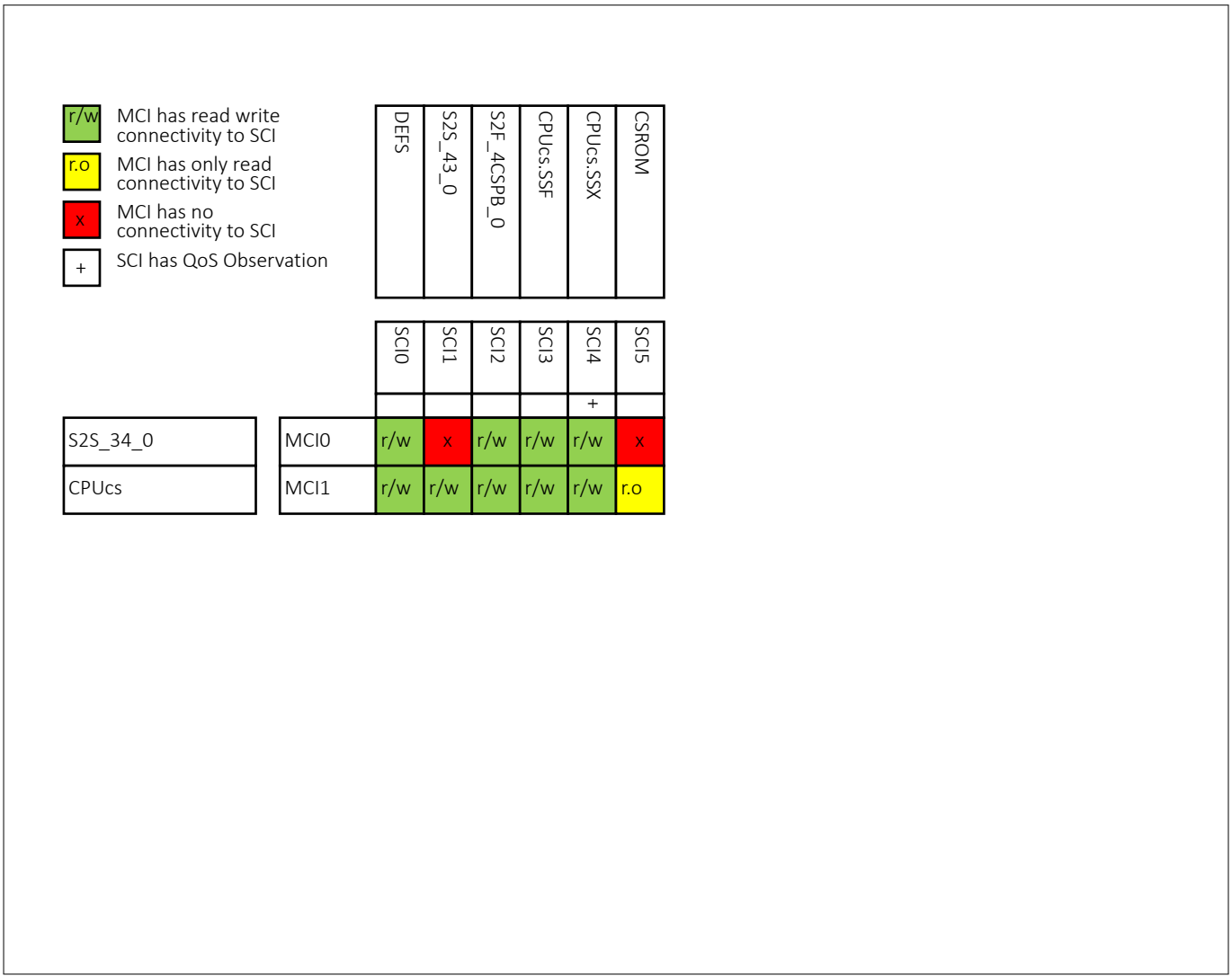


Figure 67 Domain 4 connectivity

6.16.3.11 SRI crossbar 5 MCI to SCI connectivity

r/w	MCI has read write connectivity to SCI
r.o	MCI has only read connectivity to SCI
x	MCI has no connectivity to SCI
+	SCI has QoS Observation

		DEFS	S2S_52_0	LETH.SIF	DRE.SIF	xSPI.SIF
		SCI0	SCI1	SCI2	SCI3	SCI4
S2S_25_0	MCI0	r/w	x	r/w	r/w	r/w
LETH.MIF	MCI1	r/w	r/w	x	r/w	x
DRE.MIF	MCI2	r/w	r/w	r/w	r/w	r/w
xSPI.MIF	MCI3	r/w	r/w	x	x	x
SDMMC	MCI4	r/w	r/w	x	x	x
CANXL	MCI5	r/w	r/w	x	x	x

Figure 68 Domain 5 connectivity

6.16.3.12 SRI crossbar 6 MCI to SCI connectivity

r/w MCI has read write connectivity to SCI
r.o MCI has only read connectivity to SCI
x MCI has no connectivity to SCI
\+ SCI has QoS Observation

		DEFS	LMU4	LMU5	LMU6	S2S_67_0
		SCI0	SCI1	SCI2	SCI3	SCI4
			+	+	+	
S2S_06_0	MCI0	r/w	r/w	r/w	r/w	x
S2S_16_0	MCI1	r/w	r/w	r/w	r/w	x
S2S_26_0	MCI2	r/w	r/w	r/w	r/w	x
S2S_36_0	MCI3	r/w	r/w	r/w	r/w	x
S2S_76_0	MCI4	r/w	r/w	r/w	r/w	x
SDMA0.MIF2	MCI5	r/w	r/w	r/w	r/w	r/w

Figure 69 Domain 6 connectivity

6.16.3.13 SRI crossbar 7 MCI to SCI connectivity

r/w MCI has read write connectivity to SCI
r.o MCI has only read connectivity to SCI
x MCI has no connectivity to SCI
\+ SCI has QoS Observation

		DEFS	LMU7	LMU8	LMU9	S2S_76_0	S2F_7TPB_0	S2S_73_0
		SCI0	SCI1	SCI2	SCI3	SCI4	SCI5	SCI6
			+	+	+			
S2S_07_0	MCI0	r/w	r/w	r/w	r/w	x	r/w	x
S2S_17_0	MCI1	r/w	r/w	r/w	r/w	x	r/w	x
S2S_27_0	MCI2	r/w	r/w	r/w	r/w	x	r/w	x
S2S_37_0	MCI3	r/w	r/w	r/w	r/w	x	r/w	x
S2S_67_0	MCI4	r/w	r/w	r/w	r/w	x	r/w	x
SDMA1.MIF2	MCI5	r/w	r/w	r/w	r/w	r/w	r/w	x
TRIF.MIF0	MCI6	r/w	r/w	r/w	r/w	r/w	x	r/w
F2S_TPB7_0	MCI7	r/w	r/w	r/w	r/w	r/w	x	r/w

Figure 70 Domain 7 connectivity

6.16.4 TC4Dx SRI registers

6.16.4.1 Register address space - DOM

Table 589 Registers address space - DOM

Module	Base address	End address	Note
DOM0	$F8F00000_H$	$F8F0FFFF_H$	SRI slave interface
DOM1	$F8F10000_H$	$F8F1FFFF_H$	SRI slave interface
DOM2	$F8F20000_H$	$F8F2FFFF_H$	SRI slave interface
DOM3	$F8F30000_H$	$F8F3FFFF_H$	SRI slave interface
DOM4	$F8F40000_H$	$F8F4FFFF_H$	SRI slave interface
DOM5	$F8F50000_H$	$F8F5FFFF_H$	SRI slave interface
DOM6	$F8F60000_H$	$F8F6FFFF_H$	SRI slave interface
DOM7	$F8F70000_H$	$F8F7FFFF_H$	SRI slave interface

6.16.4.2 Register overview - access mode glossary

Table 590 Register overview - access mode glossary

Keyword	Description
SE	Access protection using PROT register DOM0_PROTSE or DOM1_PROTSE or DOM2_PROTSE or DOM3_PROTSE or DOM4_PROTSE or DOM5_PROTSE or DOM6_PROTSE or DOM7_PROTSE .
APU-P	Protection group consisting of registers DOM0_ACCEN_WRA , DOM0_ACCEN_WRB , DOM0_ACCEN_RDA , DOM0_ACCEN_RDB , DOM0_ACCEN_VM , DOM0_ACCEN_PRS or DOM1_ACCEN_WRA , DOM1_ACCEN_WRB , DOM1_ACCEN_RDA , DOM1_ACCEN_RDB , DOM1_ACCEN_VM , DOM1_ACCEN_PRS or DOM2_ACCEN_WRA , DOM2_ACCEN_WRB , DOM2_ACCEN_RDA , DOM2_ACCEN_RDB , DOM2_ACCEN_VM , DOM2_ACCEN_PRS or DOM3_ACCEN_WRA , DOM3_ACCEN_WRB , DOM3_ACCEN_RDA , DOM3_ACCEN_RDB , DOM3_ACCEN_VM , DOM3_ACCEN_PRS or DOM4_ACCEN_WRA , DOM4_ACCEN_WRB , DOM4_ACCEN_RDA , DOM4_ACCEN_RDB , DOM4_ACCEN_VM , DOM4_ACCEN_PRS or DOM5_ACCEN_WRA , DOM5_ACCEN_WRB , DOM5_ACCEN_RDA , DOM5_ACCEN_RDB , DOM5_ACCEN_VM , DOM5_ACCEN_PRS or DOM6_ACCEN_WRA , DOM6_ACCEN_WRB , DOM6_ACCEN_RDA , DOM6_ACCEN_RDB , DOM6_ACCEN_VM , DOM6_ACCEN_PRS or DOM7_ACCEN_WRA , DOM7_ACCEN_WRB , DOM7_ACCEN_RDA , DOM7_ACCEN_RDB , DOM7_ACCEN_VM , DOM7_ACCEN_PRS .
P	Access protection using *APU-P* registers.
32	Access only when using 32-bit width.
SV	Access only when supervisor mode is active on the interconnect.
BE	Always returns a Bus Error.
STP	Access only during start-up, before start-up protection is set (SMM_STCON.ST_PROT=1).

(table continues...)

Table 590 (continued) Register overview - access mode glossary

Keyword	Description
U	No access restrictions.
PROT	Access restrictions as defined in the PROT register access rules.

6.16.4.3 Register overview - DOM0 (ascending offset address)

Table 591 Register overview - DOM0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
DOM0_PECONx (x=0-14)	Protocol error control register x	00000_H+ x*20_H	32, P	P, SV, 32	Application Reset	1023
DOM0_PRIORITYx (x=0-14)	SCIx arbiter priority register	00008_H+ x*20_H	32, P	P, SV, 32	Application Reset	1065
DOM0_ERRADDRx (x=0-14)	SCIx error address capture register	00010_H+ x*20_H	32, P	P, SV, 32	Application Reset	1025
DOM0_ERRx (x=0-14)	SCIx error capture register	00018_H+ x*20_H	32, P	P, SV, 32	Application Reset	1026
DOM0_ID	Identification register	00408_H	32, P	BE	Application Reset	1027
DOM0_PESTAT	Protocol error status register	00410_H	32, P	P, SV, 32	Application Reset	1068
DOM0_TIDSTAT	Transaction ID status register	00418_H	32, P	P, SV, 32	Application Reset	1072
DOM0_TIDEN	Transaction ID enable register	00420_H	32, P	P, SV, 32	Application Reset	1078
DOM0_BRCON	Bridge control register	00430_H	32, P	P, SV, 32	Application Reset	1030
DOM0_ACCEN_WRA	Write access enable register A	$004C0_H$	U	SV, SE	Application Reset	1030
DOM0_ACCEN_WRB	Write access enable register B	$004C4_H$	U	SV, SE	Application Reset	1031
DOM0_ACCEN_RDA	Read access enable register A	$004C8_H$	U	SV, SE	Application Reset	1031
DOM0_ACCEN_RDB	Read access enable register B	$004CC_H$	U	SV, SE	Application Reset	1032
DOM0_ACCEN_VM	VM access enable register	$004D0_H$	U	SV, SE	Application Reset	1032
DOM0_ACCEN_PRS	PRS access enable register	$004D4_H$	U	SV, SE	Application Reset	1033
DOM0_PROTSE	Resource protection register	$004E0_H$	U	SV, PROT	Application Reset	1034

(table continues...)

Table 591 (continued) Register overview - DOM0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
DOM0_QOS_ICTRL	QoS time interval control register	01000_H	P	SV, P	Application Reset	1035
DOM0_QOS_ICNT	QoS time interval counter	01004_H	P	BE	Application Reset	1036
DOM0_QOSx_ARPp_TAGIDA (p=0-7;x=1,3,5,7-13)	QOSx ARPp TAG-ID assignment register A	01020_H+ x*200_H+ p*20_H	P	P, SV	Application Reset	1036
DOM0_QOSx_ARPp_TAGIDB (p=0-7;x=1,3,5,7-13)	QOSx ARPp TAG-ID assignment register B	01024_H+ x*200_H+ p*20_H	P	SV, P	Application Reset	1037
DOM0_QOSx_ARPp_VM (p=0-7;x=1,3,5,7-13)	QOSx ARPp VM assignment register	01028_H+ x*200_H+ p*20_H	P	SV, P	Application Reset	1037
DOM0_QOSx_ARPp_THR (p=0-7;x=1,3,5,7-13)	QOSx ARPp threshold register	$0102C_H$+ x*200_H+ p*20_H	P	SV, P	Application Reset	1038
DOM0_QOSx_ARPp_CTRL (p=0-7;x=1,3,5,7-13)	QOSx ARPp control register	01030_H+ x*200_H+ p*20_H	P	SV, P	Application Reset	1039
DOM0_QOSx_ARPp_CCC (p=0-7;x=1,3,5,7-13)	QOSx ARPp cycle count capture register	01034_H+ x*200_H+ p*20_H	P	BE	Application Reset	1039
DOM0_QOSx_ARPp_CCV (p=0-7;x=1,3,5,7-13)	QOSx ARPp current counter value register	01038_H+ x*200_H+ p*20_H	P	BE	Application Reset	1040

6.16.4.4 Register overview - DOM1 (ascending offset address)

Table 592 Register overview - DOM1 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
DOM1_PECONx (x=0-13)	Protocol error control register x	00000_H+ x*20_H	32, P	P, SV, 32	Application Reset	1023
DOM1_PRIORITYx (x=0-13)	SCIx arbiter priority register	00008_H+ x*20_H	32, P	P, SV, 32	Application Reset	1066

(table continues...)

Table 592 (continued) Register overview - DOM1 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
DOM1_ERRADDRx (x=0-13)	SCIx error address capture register	00010_H+ x*20_H	32, P	P, SV, 32	Application Reset	1025
DOM1_ERRx (x=0-13)	SCIx error capture register	00018_H+ x*20_H	32, P	P, SV, 32	Application Reset	1026
DOM1_ID	Identification register	00408_H	32, P	BE	Application Reset	1027
DOM1_PESTAT	Protocol error status register	00410_H	32, P	P, SV, 32	Application Reset	1069
DOM1_TIDSTAT	Transaction ID status register	00418_H	32, P	P, SV, 32	Application Reset	1073
DOM1_TIDEN	Transaction ID enable register	00420_H	32, P	P, SV, 32	Application Reset	1079
DOM1_BRCON	Bridge control register	00430_H	32, P	P, SV, 32	Application Reset	1030
DOM1_ACCEN_WRA	Write access enable register A	$004C0_H$	U	SV, SE	Application Reset	1030
DOM1_ACCEN_WRB	Write access enable register B	$004C4_H$	U	SV, SE	Application Reset	1031
DOM1_ACCEN_RDA	Read access enable register A	$004C8_H$	U	SV, SE	Application Reset	1031
DOM1_ACCEN_RDB	Read access enable register B	$004CC_H$	U	SV, SE	Application Reset	1032
DOM1_ACCEN_VM	VM access enable register	$004D0_H$	U	SV, SE	Application Reset	1032
DOM1_ACCEN_PRS	PRS access enable register	$004D4_H$	U	SV, SE	Application Reset	1033
DOM1_PROTSE	Resource protection register	$004E0_H$	U	SV, PROT	Application Reset	1034
DOM1_QOS_ICTRL	QoS time interval control register	01000_H	P	SV, P	Application Reset	1035
DOM1_QOS_ICNT	QoS time interval counter	01004_H	P	BE	Application Reset	1036
DOM1_QOSx_ARPp_TAGIDA (p=0-7;x=2,4,6-13)	QOSx ARPp TAG-ID assignment register A	01020_H+ x*200_H+ p*20_H	P	P, SV	Application Reset	1036
DOM1_QOSx_ARPp_TAGIDB (p=0-7;x=2,4,6-13)	QOSx ARPp TAG-ID assignment register B	01024_H+ x*200_H+ p*20_H	P	SV, P	Application Reset	1037

(table continues...)

Table 592 (continued) Register overview - DOM1 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
DOM1_QOSx_ARPp_VM (p=0-7;x=2,4,6-13)	QOSx ARPp VM assignment register	01028_H+ x*200_H+ p*20_H	P	SV, P	Application Reset	1037
DOM1_QOSx_ARPp_THR (p=0-7;x=2,4,6-13)	QOSx ARPp threshold register	$0102C_H$+ x*200_H+ p*20_H	P	SV, P	Application Reset	1038
DOM1_QOSx_ARPp_CTRL (p=0-7;x=2,4,6-13)	QOSx ARPp control register	01030_H+ x*200_H+ p*20_H	P	SV, P	Application Reset	1039
DOM1_QOSx_ARPp_CCC (p=0-7;x=2,4,6-13)	QOSx ARPp cycle count capture register	01034_H+ x*200_H+ p*20_H	P	BE	Application Reset	1039
DOM1_QOSx_ARPp_CCV (p=0-7;x=2,4,6-13)	QOSx ARPp current counter value register	01038_H+ x*200_H+ p*20_H	P	BE	Application Reset	1040

6.16.4.5 Register overview - DOM2 (ascending offset address)

Table 593 Register overview - DOM2 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
DOM2_PECONx (x=0-9)	Protocol error control register x	00000_H+ x*20_H	32, P	P, SV, 32	Application Reset	1023
DOM2_PRIORITYx (x=0-9)	SCIx arbiter priority register	00008_H+ x*20_H	32, P	P, SV, 32	Application Reset	1066
DOM2_ERRADDRx (x=0-9)	SCIx error address capture register	00010_H+ x*20_H	32, P	P, SV, 32	Application Reset	1025
DOM2_ERRx (x=0-9)	SCIx error capture register	00018_H+ x*20_H	32, P	P, SV, 32	Application Reset	1026
DOM2_ID	Identification register	00408_H	32, P	BE	Application Reset	1027
DOM2_PESTAT	Protocol error status register	00410_H	32, P	P, SV, 32	Application Reset	1069
DOM2_TIDSTAT	Transaction ID status register	00418_H	32, P	P, SV, 32	Application Reset	1074

(table continues...)

Table 593 (continued) Register overview - DOM2 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
DOM2_TIDEN	Transaction ID enable register	00420_H	32, P	P, SV, 32	Application Reset	1079
DOM2_BRCON	Bridge control register	00430_H	32, P	P, SV, 32	Application Reset	1030
DOM2_ACCEN_WRA	Write access enable register A	$004C0_H$	U	SV, SE	Application Reset	1030
DOM2_ACCEN_WRB	Write access enable register B	$004C4_H$	U	SV, SE	Application Reset	1031
DOM2_ACCEN_RDA	Read access enable register A	$004C8_H$	U	SV, SE	Application Reset	1031
DOM2_ACCEN_RDB	Read access enable register B	$004CC_H$	U	SV, SE	Application Reset	1032
DOM2_ACCEN_VM	VM access enable register	$004D0_H$	U	SV, SE	Application Reset	1032
DOM2_ACCEN_PRS	PRS access enable register	$004D4_H$	U	SV, SE	Application Reset	1033
DOM2_PROTSE	Resource protection register	$004E0_H$	U	SV, PROT	Application Reset	1034
DOM2_QOS_ICTRL	QoS time interval control register	01000_H	P	SV, P	Application Reset	1035
DOM2_QOS_ICNT	QoS time interval counter	01004_H	P	BE	Application Reset	1036
DOM2_QOSx_ARPp_TAGIDA (p=0-7;x=1-2,7-9)	QOSx ARPp TAG-ID assignment register A	01020_H+ x*200_H+ p*20_H	P	P, SV	Application Reset	1036
DOM2_QOSx_ARPp_TAGIDB (p=0-7;x=1-2,7-9)	QOSx ARPp TAG-ID assignment register B	01024_H+ x*200_H+ p*20_H	P	SV, P	Application Reset	1037
DOM2_QOSx_ARPp_VM (p=0-7;x=1-2,7-9)	QOSx ARPp VM assignment register	01028_H+ x*200_H+ p*20_H	P	SV, P	Application Reset	1037
DOM2_QOSx_ARPp_THR (p=0-7;x=1-2,7-9)	QOSx ARPp threshold register	$0102C_H$+ x*200_H+ p*20_H	P	SV, P	Application Reset	1038
DOM2_QOSx_ARPp_CTRL (p=0-7;x=1-2,7-9)	QOSx ARPp control register	01030_H+ x*200_H+ p*20_H	P	SV, P	Application Reset	1039
DOM2_QOSx_ARPp_CCC (p=0-7;x=1-2,7-9)	QOSx ARPp cycle count capture register	01034_H+ x*200_H+ p*20_H	P	BE	Application Reset	1039
DOM2_QOSx_ARPp_CCV (p=0-7;x=1-2,7-9)	QOSx ARPp current counter value register	01038_H+ x*200_H+ p*20_H	P	BE	Application Reset	1040

(table continues...)

Table 593 (continued) Register overview - DOM2 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		

6.16.4.6 Register overview - DOM3 (ascending offset address)

Table 594 Register overview - DOM3 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
DOM3_PECONx (x=0-11)	Protocol error control register x	00000_H+ x*20_H	32, P	P, SV, 32	Application Reset	1023
DOM3_PRIORITYx (x=0-11)	SCIx arbiter priority register	00008_H+ x*20_H	32, P	P, SV, 32	Application Reset	1067
DOM3_ERRADDRx (x=0-11)	SCIx error address capture register	00010_H+ x*20_H	32, P	P, SV, 32	Application Reset	1025
DOM3_ERRx (x=0-11)	SCIx error capture register	00018_H+ x*20_H	32, P	P, SV, 32	Application Reset	1026
DOM3_ID	Identification register	00408_H	32, P	BE	Application Reset	1027
DOM3_PESTAT	Protocol error status register	00410_H	32, P	P, SV, 32	Application Reset	1070
DOM3_TIDSTAT	Transaction ID status register	00418_H	32, P	P, SV, 32	Application Reset	1075
DOM3_TIDEN	Transaction ID enable register	00420_H	32, P	P, SV, 32	Application Reset	1080
DOM3_BRCON	Bridge control register	00430_H	32, P	P, SV, 32	Application Reset	1030
DOM3_ACCEN_WRA	Write access enable register A	$004C0_H$	U	SV, SE	Application Reset	1030
DOM3_ACCEN_WRB	Write access enable register B	$004C4_H$	U	SV, SE	Application Reset	1031
DOM3_ACCEN_RDA	Read access enable register A	$004C8_H$	U	SV, SE	Application Reset	1031
DOM3_ACCEN_RDB	Read access enable register B	$004CC_H$	U	SV, SE	Application Reset	1032
DOM3_ACCEN_VM	VM access enable register	$004D0_H$	U	SV, SE	Application Reset	1032
DOM3_ACCEN_PRS	PRS access enable register	$004D4_H$	U	SV, SE	Application Reset	1033
DOM3_PROTSE	Resource protection register	$004E0_H$	U	SV, PROT	Application Reset	1034

(table continues...)

Table 594 (continued) Register overview - DOM3 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
DOM3_QOS_ICTRL	QoS time interval control register	01000_H	P	SV, P	Application Reset	1035
DOM3_QOS_ICNT	QoS time interval counter	01004_H	P	BE	Application Reset	1036
DOM3_QOSx_ARPp_TAGIDA (p=0-7;x=1-5,10-11)	QOSx ARPp TAG-ID assignment register A	01020_H+ x*200_H+ p*20_H	P	P, SV	Application Reset	1036
DOM3_QOSx_ARPp_TAGIDB (p=0-7;x=1-5,10-11)	QOSx ARPp TAG-ID assignment register B	01024_H+ x*200_H+ p*20_H	P	SV, P	Application Reset	1037
DOM3_QOSx_ARPp_VM (p=0-7;x=1-5,10-11)	QOSx ARPp VM assignment register	01028_H+ x*200_H+ p*20_H	P	SV, P	Application Reset	1037
DOM3_QOSx_ARPp_THR (p=0-7;x=1-5,10-11)	QOSx ARPp threshold register	$0102C_H$+ x*200_H+ p*20_H	P	SV, P	Application Reset	1038
DOM3_QOSx_ARPp_CTRL (p=0-7;x=1-5,10-11)	QOSx ARPp control register	01030_H+ x*200_H+ p*20_H	P	SV, P	Application Reset	1039
DOM3_QOSx_ARPp_CCC (p=0-7;x=1-5,10-11)	QOSx ARPp cycle count capture register	01034_H+ x*200_H+ p*20_H	P	BE	Application Reset	1039
DOM3_QOSx_ARPp_CCV (p=0-7;x=1-5,10-11)	QOSx ARPp current counter value register	01038_H+ x*200_H+ p*20_H	P	BE	Application Reset	1040

6.16.4.7 Register overview - DOM4 (ascending offset address)

Table 595 Register overview - DOM4 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
DOM4_PECONx (x=0-5)	Protocol error control register x	00000_H+ x*20_H	32, P	P, SV, 32	Application Reset	1023
DOM4_PRIORITYx (x=0-5)	SCIx arbiter priority register	00008_H+ x*20_H	32, P	P, SV, 32	Application Reset	1067

(table continues...)

Table 595 (continued) Register overview - DOM4 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
DOM4_ERRADDRx (x=0-5)	SCIx error address capture register	00010_H+x*20_H	32, P	P, SV, 32	Application Reset	1025
DOM4_ERRx (x=0-5)	SCIx error capture register	00018_H+x*20_H	32, P	P, SV, 32	Application Reset	1026
DOM4_ID	Identification register	00408_H	32, P	BE	Application Reset	1027
DOM4_PESTAT	Protocol error status register	00410_H	32, P	P, SV, 32	Application Reset	1070
DOM4_TIDSTAT	Transaction ID status register	00418_H	32, P	P, SV, 32	Application Reset	1075
DOM4_TIDEN	Transaction ID enable register	00420_H	32, P	P, SV, 32	Application Reset	1080
DOM4_BRCON	Bridge control register	00430_H	32, P	P, SV, 32	Application Reset	1030
DOM4_ACCEN_WRA	Write access enable register A	$004C0_H$	U	SV, SE	Application Reset	1030
DOM4_ACCEN_WRB	Write access enable register B	$004C4_H$	U	SV, SE	Application Reset	1031
DOM4_ACCEN_RDA	Read access enable register A	$004C8_H$	U	SV, SE	Application Reset	1031
DOM4_ACCEN_RDB	Read access enable register B	$004CC_H$	U	SV, SE	Application Reset	1032
DOM4_ACCEN_VM	VM access enable register	$004D0_H$	U	SV, SE	Application Reset	1032
DOM4_ACCEN_PRS	PRS access enable register	$004D4_H$	U	SV, SE	Application Reset	1033
DOM4_PROTSE	Resource protection register	$004E0_H$	U	SV, PROT	Application Reset	1034
DOM4_QOS_ICTRL	QoS time interval control register	01000_H	P	SV, P	Application Reset	1035
DOM4_QOS_ICNT	QoS time interval counter	01004_H	P	BE	Application Reset	1036
DOM4_QOSx_ARPp_TAGIDA (p=0-7;x=4)	QOSx ARPp TAG-ID assignment register A	01020_H+x*200_H+p*20_H	P	P, SV	Application Reset	1036
DOM4_QOSx_ARPp_TAGIDB (p=0-7;x=4)	QOSx ARPp TAG-ID assignment register B	01024_H+x*200_H+p*20_H	P	SV, P	Application Reset	1037
DOM4_QOSx_ARPp_VM (p=0-7;x=4)	QOSx ARPp VM assignment register	01028_H+x*200_H+p*20_H	P	SV, P	Application Reset	1037

(table continues...)

Table 595 (continued) Register overview - DOM4 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
DOM4_QOSx_ARPp_THR (p=0-7;x=4)	QOSx ARPp threshold register	$0102C_H$+ x*200_H+ p*20_H	P	SV, P	Application Reset	1038
DOM4_QOSx_ARPp_CTRL (p=0-7;x=4)	QOSx ARPp control register	01030_H+ x*200_H+ p*20_H	P	SV, P	Application Reset	1039
DOM4_QOSx_ARPp_CCC (p=0-7;x=4)	QOSx ARPp cycle count capture register	01034_H+ x*200_H+ p*20_H	P	BE	Application Reset	1039
DOM4_QOSx_ARPp_CCV (p=0-7;x=4)	QOSx ARPp current counter value register	01038_H+ x*200_H+ p*20_H	P	BE	Application Reset	1040

6.16.4.8 Register overview - DOM5 (ascending offset address)

Table 596 Register overview - DOM5 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
DOM5_PECONx (x=0-4)	Protocol error control register x	00000_H+ x*20_H	32, P	P, SV, 32	Application Reset	1023
DOM5_PRIORITYx (x=0-4)	SCIx arbiter priority register	00008_H+ x*20_H	32, P	P, SV, 32	Application Reset	1067
DOM5_ERRADDRx (x=0-4)	SCIx error address capture register	00010_H+ x*20_H	32, P	P, SV, 32	Application Reset	1025
DOM5_ERRx (x=0-4)	SCIx error capture register	00018_H+ x*20_H	32, P	P, SV, 32	Application Reset	1026
DOM5_ID	Identification register	00408_H	32, P	BE	Application Reset	1027
DOM5_PESTAT	Protocol error status register	00410_H	32, P	P, SV, 32	Application Reset	1071
DOM5_TIDSTAT	Transaction ID status register	00418_H	32, P	P, SV, 32	Application Reset	1076
DOM5_TIDEN	Transaction ID enable register	00420_H	32, P	P, SV, 32	Application Reset	1081
DOM5_BRCON	Bridge control register	00430_H	32, P	P, SV, 32	Application Reset	1030
DOM5_ACCEN_WRA	Write access enable register A	$004C0_H$	U	SV, SE	Application Reset	1030

(table continues...)

Table 596 (continued) Register overview - DOM5 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
DOM5_ACCEN_WRB	Write access enable register B	$004C4_H$	U	SV, SE	Application Reset	1031
DOM5_ACCEN_RDA	Read access enable register A	$004C8_H$	U	SV, SE	Application Reset	1031
DOM5_ACCEN_RDB	Read access enable register B	$004CC_H$	U	SV, SE	Application Reset	1032
DOM5_ACCEN_VM	VM access enable register	$004D0_H$	U	SV, SE	Application Reset	1032
DOM5_ACCEN_PRS	PRS access enable register	$004D4_H$	U	SV, SE	Application Reset	1033
DOM5_PROTSE	Resource protection register	$004E0_H$	U	SV, PROT	Application Reset	1034

6.16.4.9 Register overview - DOM6 (ascending offset address)

Table 597 Register overview - DOM6 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
DOM6_PECONx (x=0-4)	Protocol error control register x	00000_H+ x*20_H	32, P	P, SV, 32	Application Reset	1023
DOM6_PRIORITYx (x=0-4)	SCIx arbiter priority register	00008_H+ x*20_H	32, P	P, SV, 32	Application Reset	1067
DOM6_ERRADDRx (x=0-4)	SCIx error address capture register	00010_H+ x*20_H	32, P	P, SV, 32	Application Reset	1025
DOM6_ERRx (x=0-4)	SCIx error capture register	00018_H+ x*20_H	32, P	P, SV, 32	Application Reset	1026
DOM6_ID	Identification register	00408_H	32, P	BE	Application Reset	1027
DOM6_PESTAT	Protocol error status register	00410_H	32, P	P, SV, 32	Application Reset	1071
DOM6_TIDSTAT	Transaction ID status register	00418_H	32, P	P, SV, 32	Application Reset	1076
DOM6_TIDEN	Transaction ID enable register	00420_H	32, P	P, SV, 32	Application Reset	1081
DOM6_BRCON	Bridge control register	00430_H	32, P	P, SV, 32	Application Reset	1030
DOM6_ACCEN_WRA	Write access enable register A	$004C0_H$	U	SV, SE	Application Reset	1030

(table continues...)

Table 597 (continued) Register overview - DOM6 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
DOM6_ACCEN_WRB	Write access enable register B	$004C4_H$	U	SV, SE	Application Reset	1031
DOM6_ACCEN_RDA	Read access enable register A	$004C8_H$	U	SV, SE	Application Reset	1031
DOM6_ACCEN_RDB	Read access enable register B	$004CC_H$	U	SV, SE	Application Reset	1032
DOM6_ACCEN_VM	VM access enable register	$004D0_H$	U	SV, SE	Application Reset	1032
DOM6_ACCEN_PRS	PRS access enable register	$004D4_H$	U	SV, SE	Application Reset	1033
DOM6_PROTSE	Resource protection register	$004E0_H$	U	SV, PROT	Application Reset	1034
DOM6_QOS_ICTRL	QoS time interval control register	01000_H	P	SV, P	Application Reset	1035
DOM6_QOS_ICNT	QoS time interval counter	01004_H	P	BE	Application Reset	1036
DOM6_QOSx_ARPp_TAGIDA (p=0-7;x=1-3)	QOSx ARPp TAG-ID assignment register A	01020_H+ x*200_H+ p*20_H	P	P, SV	Application Reset	1036
DOM6_QOSx_ARPp_TAGIDB (p=0-7;x=1-3)	QOSx ARPp TAG-ID assignment register B	01024_H+ x*200_H+ p*20_H	P	SV, P	Application Reset	1037
DOM6_QOSx_ARPp_VM (p=0-7;x=1-3)	QOSx ARPp VM assignment register	01028_H+ x*200_H+ p*20_H	P	SV, P	Application Reset	1037
DOM6_QOSx_ARPp_THR (p=0-7;x=1-3)	QOSx ARPp threshold register	$0102C_H$+ x*200_H+ p*20_H	P	SV, P	Application Reset	1038
DOM6_QOSx_ARPp_CTRL (p=0-7;x=1-3)	QOSx ARPp control register	01030_H+ x*200_H+ p*20_H	P	SV, P	Application Reset	1039
DOM6_QOSx_ARPp_CCC (p=0-7;x=1-3)	QOSx ARPp cycle count capture register	01034_H+ x*200_H+ p*20_H	P	BE	Application Reset	1039
DOM6_QOSx_ARPp_CCV (p=0-7;x=1-3)	QOSx ARPp current counter value register	01038_H+ x*200_H+ p*20_H	P	BE	Application Reset	1040

6.16.4.10 Register overview - DOM7 (ascending offset address)

Table 598 Register overview - DOM7 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
DOM7_PECONx (x=0-6)	Protocol error control register x	00000_H+ x*20_H	32, P	P, SV, 32	Application Reset	1023
DOM7_PRIORITYx (x=0-6)	SCIx arbiter priority register	00008_H+ x*20_H	32, P	P, SV, 32	Application Reset	1066
DOM7_ERRADDRx (x=0-6)	SCIx error address capture register	00010_H+ x*20_H	32, P	P, SV, 32	Application Reset	1025
DOM7_ERRx (x=0-6)	SCIx error capture register	00018_H+ x*20_H	32, P	P, SV, 32	Application Reset	1026
DOM7_ID	Identification register	00408_H	32, P	BE	Application Reset	1027
DOM7_PESTAT	Protocol error status register	00410_H	32, P	P, SV, 32	Application Reset	1072
DOM7_TIDSTAT	Transaction ID status register	00418_H	32, P	P, SV, 32	Application Reset	1077
DOM7_TIDEN	Transaction ID enable register	00420_H	32, P	P, SV, 32	Application Reset	1081
DOM7_BRCON	Bridge control register	00430_H	32, P	P, SV, 32	Application Reset	1030
DOM7_ACCEN_WRA	Write access enable register A	$004C0_H$	U	SV, SE	Application Reset	1030
DOM7_ACCEN_WRB	Write access enable register B	$004C4_H$	U	SV, SE	Application Reset	1031
DOM7_ACCEN_RDA	Read access enable register A	$004C8_H$	U	SV, SE	Application Reset	1031
DOM7_ACCEN_RDB	Read access enable register B	$004CC_H$	U	SV, SE	Application Reset	1032
DOM7_ACCEN_VM	VM access enable register	$004D0_H$	U	SV, SE	Application Reset	1032
DOM7_ACCEN_PRS	PRS access enable register	$004D4_H$	U	SV, SE	Application Reset	1033
DOM7_PROTSE	Resource protection register	$004E0_H$	U	SV, PROT	Application Reset	1034
DOM7_QOS_ICTRL	QoS time interval control register	01000_H	P	SV, P	Application Reset	1035
DOM7_QOS_ICNT	QoS time interval counter	01004_H	P	BE	Application Reset	1036

(table continues...)

Table 598 (continued) Register overview - DOM7 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
DOM7_QOSx_ARPp_TAGIDA (p=0-7;x=1-3)	QOSx ARPp TAG-ID assignment register A	01020_H+x*200_H+p*20_H	P	P, SV	Application Reset	1036
DOM7_QOSx_ARPp_TAGIDB (p=0-7;x=1-3)	QOSx ARPp TAG-ID assignment register B	01024_H+x*200_H+p*20_H	P	SV, P	Application Reset	1037
DOM7_QOSx_ARPp_VM (p=0-7;x=1-3)	QOSx ARPp VM assignment register	01028_H+x*200_H+p*20_H	P	SV, P	Application Reset	1037
DOM7_QOSx_ARPp_THR (p=0-7;x=1-3)	QOSx ARPp threshold register	$0102C_H$+x*200_H+p*20_H	P	SV, P	Application Reset	1038
DOM7_QOSx_ARPp_CTRL (p=0-7;x=1-3)	QOSx ARPp control register	01030_H+x*200_H+p*20_H	P	SV, P	Application Reset	1039
DOM7_QOSx_ARPp_CCC (p=0-7;x=1-3)	QOSx ARPp cycle count capture register	01034_H+x*200_H+p*20_H	P	BE	Application Reset	1039
DOM7_QOSx_ARPp_CCV (p=0-7;x=1-3)	QOSx ARPp current counter value register	01038_H+x*200_H+p*20_H	P	BE	Application Reset	1040

6.16.4.11 SCIx arbiter priority register (x=0-14)

Identical functionality can be achieved by setting all priorities to high compared with setting all priorities to low.

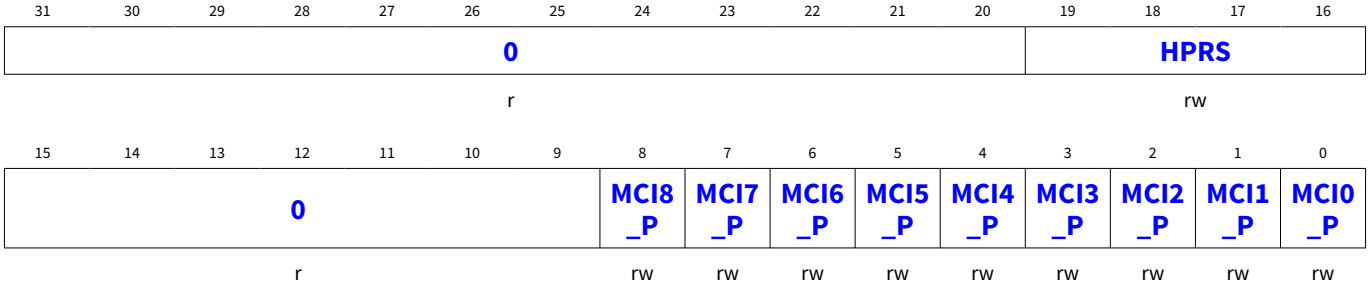
HPRS only affects arbitration when high priority requests arrive so frequently that there are no arbitration rounds with only low priority requests. When this occurs (no high priority free arbitration rounds), a maximum of HPRS grants are given to high priority masters, and then a single grant is given to the low priority round robin winner. An example configuration would be to assign the DMA to high priority and the CPUs to low priority so that the DMA does not have to wait too long for accesses and thus lose samples when moving data from RX/TX FIFOs.

***Note**: Only the MCIn_P bits assigned to implemented MCIs are present. Bits corresponding to unimplemented MCIs are treated as reserved bits.*

Reset values are device and instance specific. Refer to the device specific register table for reset values.

DOM0_PRIORITYx (x=0-14) Offset address: 00008_H+x*20_H

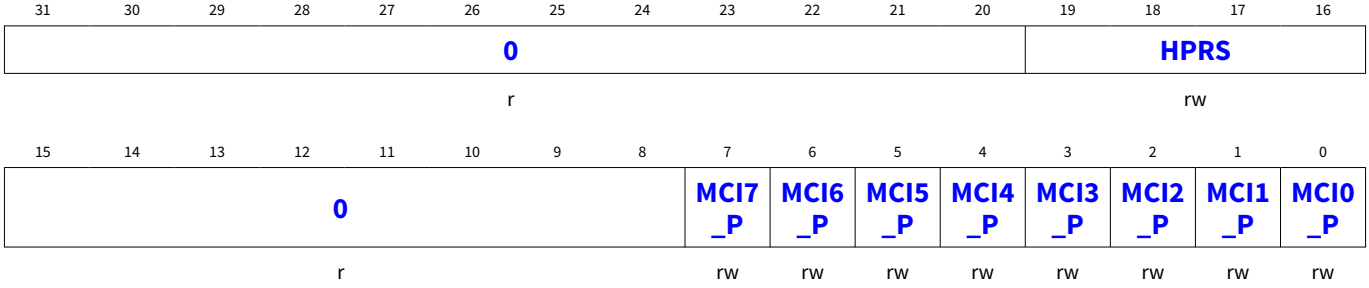
SCIx arbiter priority register Application Reset value: 0005 0171_H



Field	Bits	Type	Description
MCIn_P (n=0-8)	n	rw	**MCIn Priority** 0_B MCIn requests are arbitrated in the low priority round robin. 1_B MCIn requests are arbitrated in the high priority round robin.
HPRS	19:16	rw	**High priority round share** Number of transactions to give to the high priority round robin before a transaction from low priority round (when request saturated). This number may not be less than the number of high priority MCI programmed via. MCIn_P.
0	15:9, 31:20	r	**Reserved** Read as 0; should be written with 0.

6.16.4.12 SCIx arbiter priority register (x=0-13)

DOM1_PRIORITYx (x=0-13)	Offset address:	00008_H+x*20_H
SCIx arbiter priority register	Application Reset value:	$0005\ 00F8_H$
DOM2_PRIORITYx (x=0-9)	Offset address:	00008_H+x*20_H
SCIx arbiter priority register	Application Reset value:	$0003\ 0038_H$
DOM7_PRIORITYx (x=0-6)	Offset address:	00008_H+x*20_H
SCIx arbiter priority register	Application Reset value:	$0006\ 003F_H$



Field	Bits	Type	Description
MCIn_P (n=0-7)	n	rw	**MCIn Priority** 0_B MCIn requests are arbitrated in the low priority round robin. 1_B MCIn requests are arbitrated in the high priority round robin.

(table continues...)

(continued)

Field	Bits	Type	Description
HPRS	19:16	rw	**High priority round share** Number of transactions to give to the high priority round robin before a transaction from low priority round (when request saturated). This number may not be less than the number of high priority MCI programmed via. MCIn_P.
0	15:8, 31:20	r	**Reserved** Read as 0; should be written with 0.

6.16.4.13 SCIx arbiter priority register (x=0-11)

DOM3_PRIORITYx (x=0-11) Offset address: 00008_H+x*20_H

SCIx arbiter priority register Application Reset value: $0004\ 003C_H$

DOM5_PRIORITYx (x=0-4) Offset address: 00008_H+x*20_H

SCIx arbiter priority register Application Reset value: $0001\ 0001_H$

DOM6_PRIORITYx (x=0-4) Offset address: 00008_H+x*20_H

SCIx arbiter priority register Application Reset value: $0006\ 003F_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0												HPRS			
r												rw			

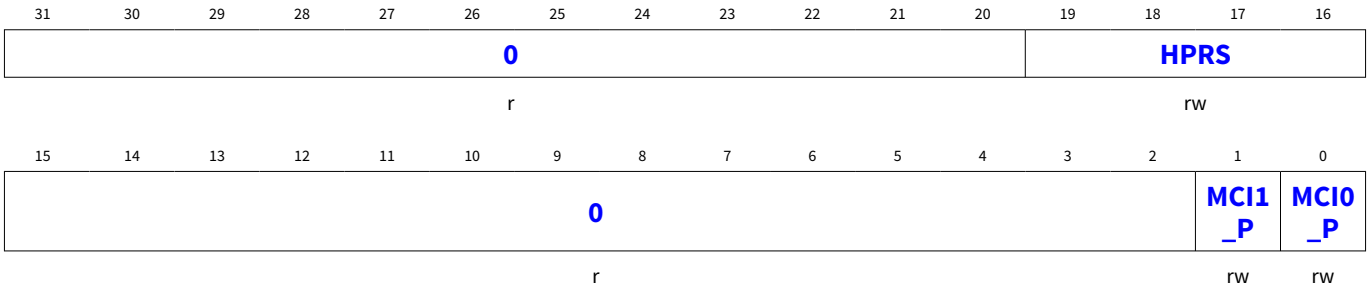
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0										MCI5_P	MCI4_P	MCI3_P	MCI2_P	MCI1_P	MCI0_P
r										rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
MCIn_P (n=0-5)	n	rw	**MCIn Priority** 0_B MCIn requests are arbitrated in the low priority round robin. 1_B MCIn requests are arbitrated in the high priority round robin.
HPRS	19:16	rw	**High priority round share** Number of transactions to give to the high priority round robin before a transaction from low priority round (when request saturated). This number may not be less than the number of high priority MCI programmed via. MCIn_P.
0	15:6, 31:20	r	**Reserved** Read as 0; should be written with 0.

6.16.4.14 SCIx arbiter priority register (x=0-5)

DOM4_PRIORITYx (x=0-5) Offset address: 00008_H+x*20_H

SCIx arbiter priority register Application Reset value: $0001\ 0001_H$



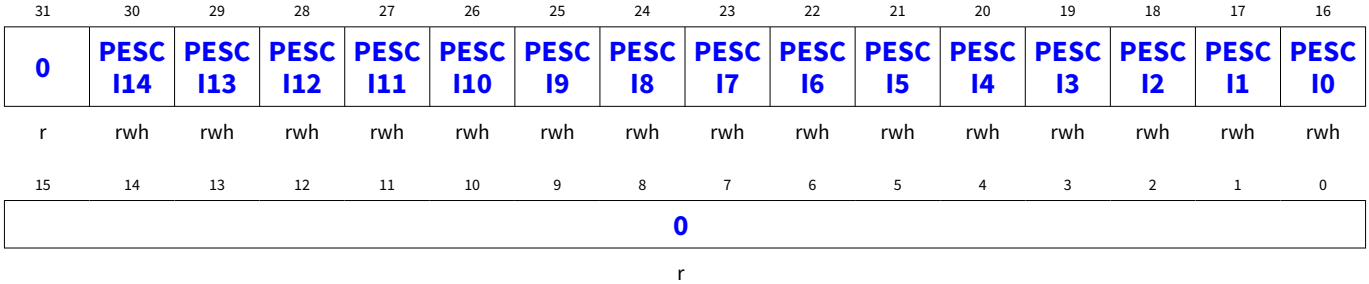
Field	Bits	Type	Description
MCIn_P (n=0-1)	n	rw	**MCIn Priority** 0_B MCIn requests are arbitrated in the low priority round robin. 1_B MCIn requests are arbitrated in the high priority round robin.
HPRS	19:16	rw	**High priority round share** Number of transactions to give to the high priority round robin before a transaction from low priority round (when request saturated). This number may not be less than the number of high priority MCI programmed via. MCIn_P.
0	15:2, 31:20	r	**Reserved** Read as 0; should be written with 0.

6.16.4.15 Protocol error status register

***Note**: Only the bits assigned to implemented SCIs are present. Bits corresponding to unimplemented SCIs are treated as reserved bits, which will be read as 0, and should be written with 0.*

DOM0_PESTAT Offset address: 00410_H

Protocol error status register Application Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
PESCIn (n=0-14)	n+16	rwh	**Protocol error status of SCIn** Writing 0 to the bit leaves the content unchanged, while writing 1 to the bit clears it. In case the bit is simultaneously cleared via software and set via hardware, the bit remains set and is not cleared. 0_B No protocol error has been indicated by SCIn 1_B A protocol error has been indicated by SCIn

(table continues...)

(continued)

Field	Bits	Type	Description
0	15:0, 31	r	**Reserved** Read as 0; should be written with 0.

6.16.4.16 Protocol error status register

DOM1_PESTAT Offset address: 00410_H

Protocol error status register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		PESCI13	PESCI12	PESCI11	PESCI10	PESCI9	PESCI8	PESCI7	PESCI6	PESCI5	PESCI4	PESCI3	PESCI2	PESCI1	PESCI0
r		rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															
r															

Field	Bits	Type	Description
PESCIn (n=0-13)	n+16	rwh	**Protocol error status of SCIn** Writing 0 to the bit leaves the content unchanged, while writing 1 to the bit clears it. In case the bit is simultaneously cleared via software and set via hardware, the bit remains set and is not cleared. 0_B No protocol error has been indicated by SCIn 1_B A protocol error has been indicated by SCIn
0	15:0, 31:30	r	**Reserved** Read as 0; should be written with 0.

6.16.4.17 Protocol error status register

DOM2_PESTAT Offset address: 00410_H

Protocol error status register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0						PESCI9	PESCI8	PESCI7	PESCI6	PESCI5	PESCI4	PESCI3	PESCI2	PESCI1	PESCI0
r						rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															
r															

Field	Bits	Type	Description
PESCIn (n=0-9)	n+16	rwh	**Protocol error status of SCIn** Writing 0 to the bit leaves the content unchanged, while writing 1 to the bit clears it. In case the bit is simultaneously cleared via software and set via hardware, the bit remains set and is not cleared. 0_B No protocol error has been indicated by SCIn 1_B A protocol error has been indicated by SCIn
0	15:0, 31:26	r	**Reserved** Read as 0; should be written with 0.

6.16.4.18 Protocol error status register

DOM3_PESTAT Offset address: 00410_H

Protocol error status register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0				PESCI11	PESCI10	PESCI9	PESCI8	PESCI7	PESCI6	PESCI5	PESCI4	PESCI3	PESCI2	PESCI1	PESCI0
r				rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

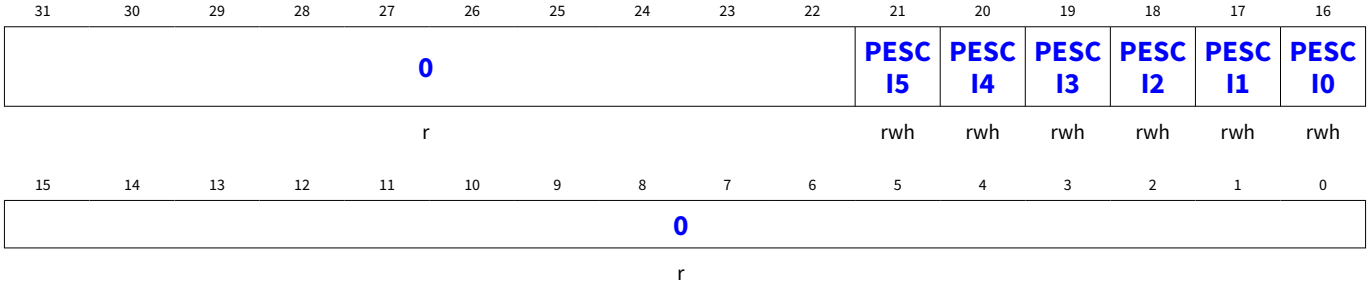
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															
r															

Field	Bits	Type	Description
PESCIn (n=0-11)	n+16	rwh	**Protocol error status of SCIn** Writing 0 to the bit leaves the content unchanged, while writing 1 to the bit clears it. In case the bit is simultaneously cleared via software and set via hardware, the bit remains set and is not cleared. 0_B No protocol error has been indicated by SCIn 1_B A protocol error has been indicated by SCIn
0	15:0, 31:28	r	**Reserved** Read as 0; should be written with 0.

6.16.4.19 Protocol error status register

DOM4_PESTAT Offset address: 00410_H

Protocol error status register Application Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
PESCIn (n=0-5)	n+16	rwh	**Protocol error status of SCIn** Writing 0 to the bit leaves the content unchanged, while writing 1 to the bit clears it. In case the bit is simultaneously cleared via software and set via hardware, the bit remains set and is not cleared. 0_B No protocol error has been indicated by SCIn 1_B A protocol error has been indicated by SCIn
0	15:0, 31:22	r	**Reserved** Read as 0; should be written with 0.

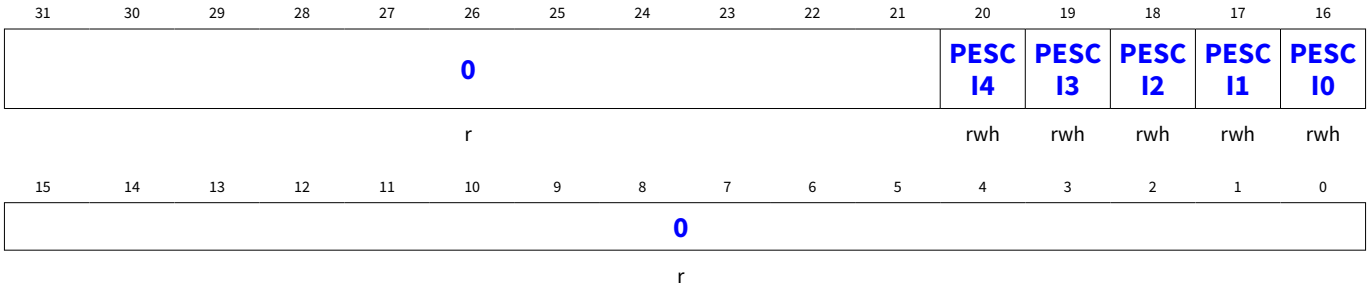
6.16.4.20 Protocol error status register

DOM5_PESTAT Offset address: 00410_H

Protocol error status register Application Reset value: $0000\ 0000_H$

DOM6_PESTAT Offset address: 00410_H

Protocol error status register Application Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
PESCIn (n=0-4)	n+16	rwh	**Protocol error status of SCIn** Writing 0 to the bit leaves the content unchanged, while writing 1 to the bit clears it. In case the bit is simultaneously cleared via software and set via hardware, the bit remains set and is not cleared. 0_B No protocol error has been indicated by SCIn 1_B A protocol error has been indicated by SCIn
0	15:0, 31:21	r	**Reserved** Read as 0; should be written with 0.

6.16.4.21 Protocol error status register

DOM7_PESTAT Offset address: 00410_H

Protocol error status register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0									PESCI6	PESCI5	PESCI4	PESCI3	PESCI2	PESCI1	PESCI0
r									rwh	rwh	rwh	rwh	rwh	rwh	rwh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															
r															

Field	Bits	Type	Description
PESCIn (n=0-6)	n+16	rwh	**Protocol error status of SCIn** Writing 0 to the bit leaves the content unchanged, while writing 1 to the bit clears it. In case the bit is simultaneously cleared via software and set via hardware, the bit remains set and is not cleared. 0_B No protocol error has been indicated by SCIn 1_B A protocol error has been indicated by SCIn
0	15:0, 31:23	r	**Reserved** Read as 0; should be written with 0.

6.16.4.22 Transaction ID status register

Only the bits assigned to implemented SCIs and MCIs are present. Bits corresponding to unimplemented SCIs and MCIs are treated as reserved bits, which will be read as 0, and should be written with 0. Reset values for bits/bit fields referencing MCIs or SCIs that are not implemented or enabled are 0.

DOM0_TIDSTAT Offset address: 00418_H

Transaction ID status register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0							TIDMCI8	TIDMCI7	TIDMCI6	TIDMCI5	TIDMCI4	TIDMCI3	TIDMCI2	TIDMCI1	TIDMCI0
r							rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	TIDSCI14	TIDSCI13	TIDSCI12	TIDSCI11	TIDSCI10	TIDSCI9	TIDSCI8	TIDSCI7	TIDSCI6	TIDSCI5	TIDSCI4	TIDSCI3	TIDSCI2	TIDSCI1	TIDSCI0
r	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
TIDSCIn (n=0-14)	n	rwh	**Transaction ID error from SCIn status** Writing 0 to the bit has no effect. Writing 1 to the bit clears it. ***Note**: If the bit is simultaneously cleared via software and set due to a hardware error, the bit remains set and is not cleared.* 0_B No transaction ID error has been indicated by SCIn 1_B A transaction ID error has been indicated by SCIn
TIDMCIn (n=0-8)	n+16	rwh	**Trasaction ID Error from MCIn status** Writing 0 to the bit has no effect. Writing 1 to the bit clears it. ***Note**: If the bit is simultaneously cleared via software and set due to a hardware error, the bit remains set and is not cleared.* 0_B No transaction ID error has been indicated by MCIn 1_B A transaction ID error has been indicated by MCIn
0	15, 31:25	r	**Reserved** Read as 0; should be written with 0.

6.16.4.23 Transaction ID status register

DOM1_TIDSTAT Offset address: 00418_H

Transaction ID status register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								TIDMCI7	TIDMCI6	TIDMCI5	TIDMCI4	TIDMCI3	TIDMCI2	TIDMCI1	TIDMCI0
r								rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		TIDSCI13	TIDSCI12	TIDSCI11	TIDSCI10	TIDSCI9	TIDSCI8	TIDSCI7	TIDSCI6	TIDSCI5	TIDSCI4	TIDSCI3	TIDSCI2	TIDSCI1	TIDSCI0
r		rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
TIDSCIn (n=0-13)	n	rwh	**Transaction ID error from SCIn status** Writing 0 to the bit has no effect. Writing 1 to the bit clears it. ***Note**: If the bit is simultaneously cleared via software and set due to a hardware error, the bit remains set and is not cleared.* 0_B No transaction ID error has been indicated by SCIn 1_B A transaction ID error has been indicated by SCIn

(table continues...)

(continued)

Field	Bits	Type	Description
TIDMCIn (n=0-7)	n+16	rwh	**Trasaction ID Error from MCIn status** Writing 0 to the bit has no effect. Writing 1 to the bit clears it. ***Note**: If the bit is simultaneously cleared via software and set due to a hardware error, the bit remains set and is not cleared.* 0_B No transaction ID error has been indicated by MCIn 1_B A transaction ID error has been indicated by MCIn
0	15:14, 31:24	r	**Reserved** Read as 0; should be written with 0.

6.16.4.24 Transaction ID status register

DOM2_TIDSTAT Offset address: 00418_H

Transaction ID status register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								TIDMCI7	TIDMCI6	TIDMCI5	TIDMCI4	TIDMCI3	TIDMCI2	TIDMCI1	TIDMCI0
r								rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0						TIDSCI9	TIDSCI8	TIDSCI7	TIDSCI6	TIDSCI5	TIDSCI4	TIDSCI3	TIDSCI2	TIDSCI1	TIDSCI0
r						rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
TIDSCIn (n=0-9)	n	rwh	**Transaction ID error from SCIn status** Writing 0 to the bit has no effect. Writing 1 to the bit clears it. ***Note**: If the bit is simultaneously cleared via software and set due to a hardware error, the bit remains set and is not cleared.* 0_B No transaction ID error has been indicated by SCIn 1_B A transaction ID error has been indicated by SCIn
TIDMCIn (n=0-7)	n+16	rwh	**Trasaction ID Error from MCIn status** Writing 0 to the bit has no effect. Writing 1 to the bit clears it. ***Note**: If the bit is simultaneously cleared via software and set due to a hardware error, the bit remains set and is not cleared.* 0_B No transaction ID error has been indicated by MCIn 1_B A transaction ID error has been indicated by MCIn

(table continues...)

(continued)

Field	Bits	Type	Description
0	15:10, 31:24	r	**Reserved** Read as 0; should be written with 0.

6.16.4.25 Transaction ID status register

DOM3_TIDSTAT Offset address: 00418_H

Transaction ID status register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0										TIDMCI5	TIDMCI4	TIDMCI3	TIDMCI2	TIDMCI1	TIDMCI0
r										rwh	rwh	rwh	rwh	rwh	rwh

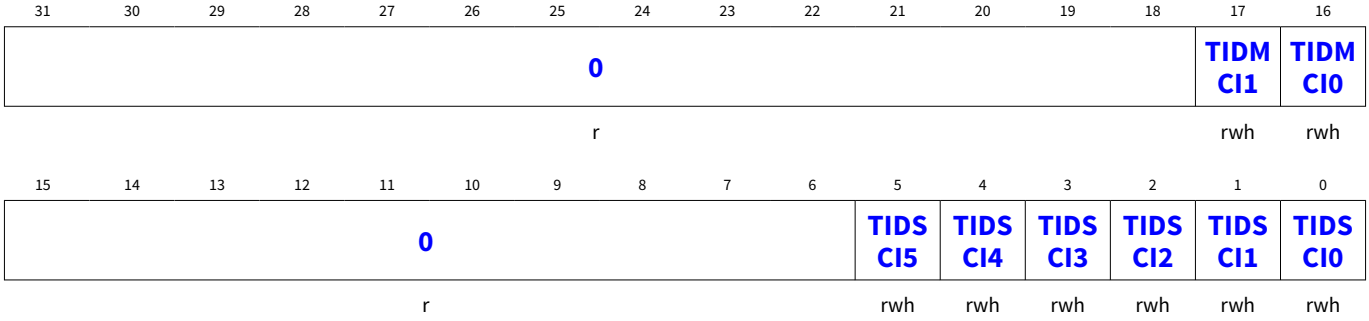
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				TIDSCI11	TIDSCI10	TIDSCI9	TIDSCI8	TIDSCI7	TIDSCI6	TIDSCI5	TIDSCI4	TIDSCI3	TIDSCI2	TIDSCI1	TIDSCI0
r				rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
TIDSCIn (n=0-11)	n	rwh	**Transaction ID error from SCIn status** Writing 0 to the bit has no effect. Writing 1 to the bit clears it. ***Note**: If the bit is simultaneously cleared via software and set due to a hardware error, the bit remains set and is not cleared.* 0_B No transaction ID error has been indicated by SCIn 1_B A transaction ID error has been indicated by SCIn
TIDMCIn (n=0-5)	n+16	rwh	**Trasaction ID Error from MCIn status** Writing 0 to the bit has no effect. Writing 1 to the bit clears it. ***Note**: If the bit is simultaneously cleared via software and set due to a hardware error, the bit remains set and is not cleared.* 0_B No transaction ID error has been indicated by MCIn 1_B A transaction ID error has been indicated by MCIn
0	15:12, 31:22	r	**Reserved** Read as 0; should be written with 0.

6.16.4.26 Transaction ID status register

DOM4_TIDSTAT Offset address: 00418_H

Transaction ID status register Application Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
TIDSCIn (n=0-5)	n	rwh	**Transaction ID error from SCIn status** Writing 0 to the bit has no effect. Writing 1 to the bit clears it. ***Note**: If the bit is simultaneously cleared via software and set due to a hardware error, the bit remains set and is not cleared.* 0_B No transaction ID error has been indicated by SCIn 1_B A transaction ID error has been indicated by SCIn
TIDMCIn (n=0-1)	n+16	rwh	**Trasaction ID Error from MCIn status** Writing 0 to the bit has no effect. Writing 1 to the bit clears it. ***Note**: If the bit is simultaneously cleared via software and set due to a hardware error, the bit remains set and is not cleared.* 0_B No transaction ID error has been indicated by MCIn 1_B A transaction ID error has been indicated by MCIn
0	15:6, 31:18	r	**Reserved** Read as 0; should be written with 0.

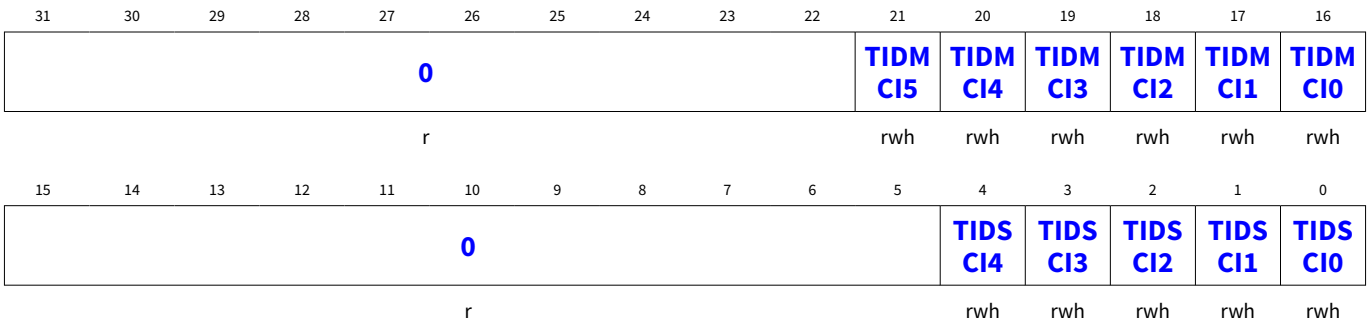
6.16.4.27 Transaction ID status register

DOM5_TIDSTAT Offset address: 00418_H

Transaction ID status register Application Reset value: $0000\ 0000_H$

DOM6_TIDSTAT Offset address: 00418_H

Transaction ID status register Application Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
TIDSCIn (n=0-4)	n	rwh	**Transaction ID error from SCIn status** Writing 0 to the bit has no effect. Writing 1 to the bit clears it. ***Note**: If the bit is simultaneously cleared via software and set due to a hardware error, the bit remains set and is not cleared.* 0_B No transaction ID error has been indicated by SCIn 1_B A transaction ID error has been indicated by SCIn
TIDMCIn (n=0-5)	n+16	rwh	**Trasaction ID Error from MCIn status** Writing 0 to the bit has no effect. Writing 1 to the bit clears it. ***Note**: If the bit is simultaneously cleared via software and set due to a hardware error, the bit remains set and is not cleared.* 0_B No transaction ID error has been indicated by MCIn 1_B A transaction ID error has been indicated by MCIn
0	15:5, 31:22	r	**Reserved** Read as 0; should be written with 0.

6.16.4.28 Transaction ID status register

DOM7_TIDSTAT Offset address: 00418_H

Transaction ID status register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								TIDMCI7	TIDMCI6	TIDMCI5	TIDMCI4	TIDMCI3	TIDMCI2	TIDMCI1	TIDMCI0
r								rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0									TIDSCI6	TIDSCI5	TIDSCI4	TIDSCI3	TIDSCI2	TIDSCI1	TIDSCI0
r									rwh	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
TIDSCIn (n=0-6)	n	rwh	**Transaction ID error from SCIn status** Writing 0 to the bit has no effect. Writing 1 to the bit clears it. ***Note**: If the bit is simultaneously cleared via software and set due to a hardware error, the bit remains set and is not cleared.* 0_B No transaction ID error has been indicated by SCIn 1_B A transaction ID error has been indicated by SCIn

(table continues...)

(continued)

Field	Bits	Type	Description
TIDMCIn (n=0-7)	n+16	rwh	**Trasaction ID Error from MCIn status** Writing 0 to the bit has no effect. Writing 1 to the bit clears it. ***Note**: If the bit is simultaneously cleared via software and set due to a hardware error, the bit remains set and is not cleared.* 0_B No transaction ID error has been indicated by MCIn 1_B A transaction ID error has been indicated by MCIn
0	15:7, 31:24	r	**Reserved** Read as 0; should be written with 0.

6.16.4.29 Transaction ID enable register

***Note**: Only the bits assigned to implemented SCIs and MCIs are present. Bits corresponding to unimplemented SCIs and MCIs are treated as reserved bits, which will be read as ´0´, and should be written with ´0´.*

Reset values for bits/bit fields referencing MCIs or SCIs that are not implemented or enabled are zero.

DOM0_TIDEN Offset address: 00420_H

Transaction ID enable register Application Reset value: $01FF\ 7FFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0							ENMCI8	ENMCI7	ENMCI6	ENMCI5	ENMCI4	ENMCI3	ENMCI2	ENMCI1	ENMCI0
r							rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	ENSCI14	ENSCI13	ENSCI12	ENSCI11	ENSCI10	ENSCI9	ENSCI8	ENSCI7	ENSCI6	ENSCI5	ENSCI4	ENSCI3	ENSCI2	ENSCI1	ENSCI0
r	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENSCIn (n=0-14)	n	rw	**Enable transaction ID error from SCIn** 0_B Transaction ID errors from SCIn are not indicated 1_B Transaction ID errors from SCIn are indicated
ENMCIn (n=0-8)	n+16	rw	**Enable transaction ID error from MCIn** 0_B Transaction ID errors from MCIn are not indicated 1_B Transaction ID errors from MCIn are indicated
0	15, 31:25	r	**Reserved** Read as 0; should be written with 0.

6.16.4.30 Transaction ID enable register

DOM1_TIDEN Offset address: 00420_H

Transaction ID enable register Application Reset value: $00FF\ 3FFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								ENMCI7	ENMCI6	ENMCI5	ENMCI4	ENMCI3	ENMCI2	ENMCI1	ENMCI0
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		ENSCI13	ENSCI12	ENSCI11	ENSCI10	ENSCI9	ENSCI8	ENSCI7	ENSCI6	ENSCI5	ENSCI4	ENSCI3	ENSCI2	ENSCI1	ENSCI0
r		rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENSCIn (n=0-13)	n	rw	**Enable transaction ID error from SCIn** 0_B Transaction ID errors from SCIn are not indicated 1_B Transaction ID errors from SCIn are indicated
ENMCIn (n=0-7)	n+16	rw	**Enable transaction ID error from MCIn** 0_B Transaction ID errors from MCIn are not indicated 1_B Transaction ID errors from MCIn are indicated
0	15:14, 31:24	r	**Reserved** Read as 0; should be written with 0.

6.16.4.31 Transaction ID enable register

DOM2_TIDEN Offset address: 00420_H

Transaction ID enable register Application Reset value: $00FF\ 03FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								ENMCI7	ENMCI6	ENMCI5	ENMCI4	ENMCI3	ENMCI2	ENMCI1	ENMCI0
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0						ENSCI9	ENSCI8	ENSCI7	ENSCI6	ENSCI5	ENSCI4	ENSCI3	ENSCI2	ENSCI1	ENSCI0
r						rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENSCIn (n=0-9)	n	rw	**Enable transaction ID error from SCIn** 0_B Transaction ID errors from SCIn are not indicated 1_B Transaction ID errors from SCIn are indicated

(table continues...)

(continued)

Field	Bits	Type	Description
ENMCIn (n=0-7)	n+16	rw	**Enable transaction ID error from MCIn** 0_B Transaction ID errors from MCIn are not indicated 1_B Transaction ID errors from MCIn are indicated
0	15:10, 31:24	r	**Reserved** Read as 0; should be written with 0.

6.16.4.32 Transaction ID enable register

DOM3_TIDEN Offset address: 00420_H

Transaction ID enable register Application Reset value: $003F\ 0FFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0										ENMCI5	ENMCI4	ENMCI3	ENMCI2	ENMCI1	ENMCI0
r										rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				ENSCI11	ENSCI10	ENSCI9	ENSCI8	ENSCI7	ENSCI6	ENSCI5	ENSCI4	ENSCI3	ENSCI2	ENSCI1	ENSCI0
r				rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENSCIn (n=0-11)	n	rw	**Enable transaction ID error from SCIn** 0_B Transaction ID errors from SCIn are not indicated 1_B Transaction ID errors from SCIn are indicated
ENMCIn (n=0-5)	n+16	rw	**Enable transaction ID error from MCIn** 0_B Transaction ID errors from MCIn are not indicated 1_B Transaction ID errors from MCIn are indicated
0	15:12, 31:22	r	**Reserved** Read as 0; should be written with 0.

6.16.4.33 Transaction ID enable register

DOM4_TIDEN Offset address: 00420_H

Transaction ID enable register Application Reset value: $0003\ 003F_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0														ENMCI1	ENMCI0
r														rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0										ENSCI5	ENSCI4	ENSCI3	ENSCI2	ENSCI1	ENSCI0
r										rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENSCIn (n=0-5)	n	rw	**Enable transaction ID error from SCIn** 0_B Transaction ID errors from SCIn are not indicated 1_B Transaction ID errors from SCIn are indicated
ENMCIn (n=0-1)	n+16	rw	**Enable transaction ID error from MCIn** 0_B Transaction ID errors from MCIn are not indicated 1_B Transaction ID errors from MCIn are indicated
0	15:6, 31:18	r	**Reserved** Read as 0; should be written with 0.

6.16.4.34 Transaction ID enable register

DOM5_TIDEN Offset address: 00420_H

Transaction ID enable register Application Reset value: $003F\ 001F_H$

DOM6_TIDEN Offset address: 00420_H

Transaction ID enable register Application Reset value: $003F\ 001F_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0										ENMCI5	ENMCI4	ENMCI3	ENMCI2	ENMCI1	ENMCI0
r										rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0											ENSCI4	ENSCI3	ENSCI2	ENSCI1	ENSCI0
r											rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENSCIn (n=0-4)	n	rw	**Enable transaction ID error from SCIn** 0_B Transaction ID errors from SCIn are not indicated 1_B Transaction ID errors from SCIn are indicated
ENMCIn (n=0-5)	n+16	rw	**Enable transaction ID error from MCIn** 0_B Transaction ID errors from MCIn are not indicated 1_B Transaction ID errors from MCIn are indicated
0	15:5, 31:22	r	**Reserved** Read as 0; should be written with 0.

6.16.4.35 Transaction ID enable register

DOM7_TIDEN Offset address: 00420_H

Transaction ID enable register Application Reset value: $00FF\ 007F_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								ENMCI7	ENMCI6	ENMCI5	ENMCI4	ENMCI3	ENMCI2	ENMCI1	ENMCI0
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0									ENSCI6	ENSCI5	ENSCI4	ENSCI3	ENSCI2	ENSCI1	ENSCI0
r									rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENSCIn (n=0-6)	n	rw	**Enable transaction ID error from SCIn** 0_B Transaction ID errors from SCIn are not indicated 1_B Transaction ID errors from SCIn are indicated
ENMCIn (n=0-7)	n+16	rw	**Enable transaction ID error from MCIn** 0_B Transaction ID errors from MCIn are not indicated 1_B Transaction ID errors from MCIn are indicated
0	15:7, 31:24	r	**Reserved** Read as 0; should be written with 0.

6.16.5 TC4Dx SRI connectivity

The table below lists interfaces to and from this functional block to other blocks in the device.

Table 599 List of SRI interface signals

Interface signals	I/O	Description
CLOCK_SRI_fCPB	In	CPB clock input
CLOCK_SRI_fRCB	In	RCB clock input
CLOCK_SRI_fSPB	In	SPB clock input
CLOCK_SRI_fSRI	In	SRI clock input
CLOCK_SRI_fSRICS	In	SRICS clock input
SMM_SRI_ApplicationReset	In	Application reset
SMM_SRI_SystemReset	In	System reset
SRI_IR_SRC_SRIx	Out	XBAR service request
SRI_SMU_ALARM_NCS	Out	SRI XBAR bus error event alarm
CANXL0_SRI_MIF	In	CANXL master interface
CPU0_SRI_M_SRI	In	CPU SRI master interface
CPU1_SRI_M_SRI	In	CPU SRI master interface
CPU2_SRI_M_SRI	In	CPU SRI master interface
CPU3_SRI_M_SRI	In	CPU SRI master interface
CPU4_SRI_M_SRI	In	CPU SRI master interface

(table continues...)

Table 599 (continued) List of SRI interface signals

Interface signals	I/O	Description
CPU5_SRI_M_SRI	In	CPU SRI master interface
SRI_CPU0_S_SRI1	Out	CPU slave interface to NVM regions
SRI_CPU1_S_SRI1	Out	CPU slave interface to NVM regions
SRI_CPU2_S_SRI1	Out	CPU slave interface to NVM regions
SRI_CPU3_S_SRI1	Out	CPU slave interface to NVM regions
SRI_CPU4_S_SRI1	Out	CPU slave interface to NVM regions
SRI_CPU5_S_SRI1	Out	CPU slave interface to NVM regions
SRI_CPU0_S_SRI0	Out	CPU slave interface to memory, SFR, STM
SRI_CPU1_S_SRI0	Out	CPU slave interface to memory, SFR, STM
SRI_CPU2_S_SRI0	Out	CPU slave interface to memory, SFR, STM
SRI_CPU3_S_SRI0	Out	CPU slave interface to memory, SFR, STM
SRI_CPU4_S_SRI0	Out	CPU slave interface to memory, SFR, STM
SRI_CPU5_S_SRI0	Out	CPU slave interface to memory, SFR, STM
CPUCS_SRI_M_SRI	In	CPU SRI master interface
SRI_CPUCS_S_SRI1	Out	CPU slave interface to NVM regions
SRI_CPUCS_S_SRI0	Out	CPU slave interface to memory, SFR, STM
SRI_NVM_SIF	Out	Slave interfaces to DMU or DMUR instances
DRE_SRI_MIF	In	DRE master interface
SRI_DRE_SIF	Out	DRE slave interface
SRI_EGTM_SIF	Out	eGTM slave interface
GETH0_SRI_MIF[1:0]	In	GETH master interface
SRI_GETH0_SIF	Out	GETH slave interface
HSSL0_SRI_MIF	In	HSSL master interface
HSSL1_SRI_MIF	In	HSSL master interface
LETH0_SRI_MIF	In	LETH master interface
SRI_LETH0_SIF	Out	LETH slave interface
SRI_LLI_SIF	Out	SRI to LLI bridge
SRI_LMU0_SIF	Out	LMU slave interface
SRI_LMU1_SIF	Out	LMU slave interface
SRI_LMU2_SIF	Out	LMU slave interface
SRI_LMU3_SIF	Out	LMU slave interface
SRI_LMU4_SIF	Out	LMU slave interface
SRI_LMU5_SIF	Out	LMU slave interface

(table continues...)

Table 599 (continued) List of SRI interface signals

Interface signals	I/O	Description
SRI_LMU6_SIF	Out	LMU slave interface
SRI_LMU7_SIF	Out	LMU slave interface
SRI_LMU8_SIF	Out	LMU slave interface
SRI_LMU9_SIF	Out	LMU slave interface
PCIE0_SRI_MIF	In	PCIe master interface
PCIE1_SRI_MIF	In	PCIe master interface
SRI_PCIE0_SIF	Out	PCIe slave interface
SRI_PCIE1_SIF	Out	PCIe slave interface
PPU_SRI_CBU	In	PPU CBU master
PPU_SRI_STU	In	PPU STU master
SRI_PPU_CSM-DMI	Out	PPU CSM slave
SRI_PPU_VMEM-DMI	Out	PPU VMEM slave
SDMMC0_SRI_MIF	In	SDMMC master interface
TRACE_SRI_MIF	In	TRIF master interface
XSPI_SRI_MIF	In	xSPI master interface
SRI_XSPI_SIF	Out	xSPI slave interface
SRI_CSS_SRI	Out	CSS slave interface
CSRM.CSCU_SRI_CSAV[1:0]	In	CS activity visibility (Trace enable for security related transactions)

6.16.6 TC4Dx SRI revision history

Initial release of the chapter.

7 Flexible Peripheral Interconnect (FPI)

The system on chip (SoC) communication is mainly based on three interconnect protocols: SRI, FPI, and AXI that used for the LLI interconnect.

The Flexible peripheral interconnect (FPI) is multi-master interconnect, 32-bit data width, based on FPI interconnect protocol. The FPI connects the high speed peripherals, such as the TC1.8 CPUs and DMA modules to the medium and low bandwidth functional blocks.

7.1 Feature list

- Optimized for high-speed and high-performance
  - Support of multiple master agents and back-to-back transaction execution
  - Transaction pipelining support to support up to 1 data transmission per cycle
  - 32-bit address and 32-bit data width
  - 8, 16, and 32-bit data transfers
  - 64, 128, and 256-bit block transfers with wrap around addressing
- Slave-controlled wait state and retry insertion
- Timeout detection and handling
- Support of atomic TriCore™ operations for example LDMST, ST.T, SWAP.W, CMPSWP, and SWPMSK
- Flexible arbitration schemes
  - Priority based, two round-robin groups and starvation prevention mechanism
- QoS master agent interconnect usage control mechanism
- QoS application interconnect usage observation mechanism
- Round-robin group control mechanism (round share mechanism)
- Starvation prevention mechanism that can take care that even low priority requests will be granted after a configurable number of arbitration cycles, permanently enabled
- Default slave mechanism that responds to a transactions to a not implemented addresses
- Debug support where transaction information are captured in case of a FPI timeout or FPI error
- Address phase includes master identifier, supervisor mode and security mode information
- All slave agents implemented with master agent identifier based access protection
- Transactions protected through end to end error detection code (EDC)

7.2 Functional overview

The Flexible Peripheral Interconnect (FPI) is a multi-master interconnect with a 32-bit interconnect supporting 8-, 16-, 32-, 64- 128- and 256-bit data transfers as well as atomic Read-Modify-Write transactions. Multi-master interconnect here means that exactly one data transfer between the connected function blocks can be executed at any time. The FPI connects the high speed functional blocks, such as the TC1.8 CPU´s and DMA modules to the medium and low bandwidth system and communication functional blocks.

The TC4x System interconnect architecture defines multiple FPI interconnect instances where each FPI interconnect instance provides a dedicated FPI Bus Control Unit (BCU):

- SBCU related to the system peripheral interconnect (SPB)
- CBCU related to the converter peripheral interconnect (CPB)
- COMBCU related to communication peripheral interconnect (COMPB)
- RBCU related to the radar control peripheral interconnect (RCB)
- CSBCU related to the cyber Security peripheral interconnect (CSPB)
- TBCU related to the tool peripheral interconnect (TPB)

***Note**: Which FPI instances are instantiated in the respective product is described in chapter 'Characteristic of FPI interconnect'. The distribution of FBs to SPB, COMPB and CPB is product-specific.*

***Note**: The distribution of FBs on SPB, COMPB and CPB is product-specific and visible through the product specific BCUs ALSTAT register bits. The product-specific BCU ALSTAT registers show which FBs are connected to the respective FPI with their FPI EDC alarm signals.*

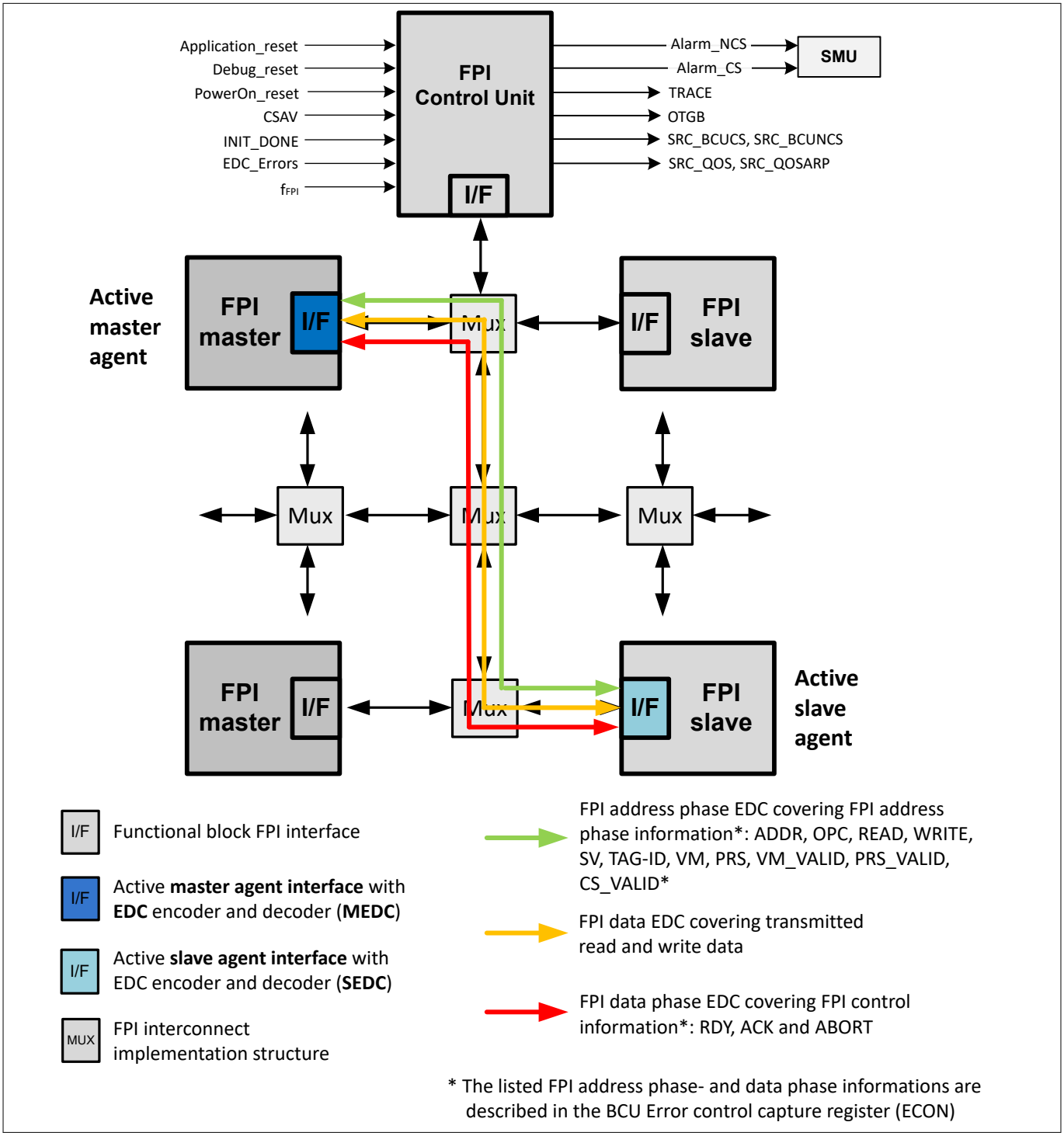


Figure 71 FPI block diagram

Functional blocks (FB) can be connected to an FPI interconnect through an FPI slave interface, also called a slave agent, and through FPI master interface that is also called the master agent. An FPI interconnect allows

the connected master agents to access the connected slave agents address range. In contrast to the SRI interconnect, which provides a cross-bar connection that supports parallel access between the connected master and slave agents, an FPI interconnect is a shared interconnect where all master and slave agents share the connections. Therefore an FPI interconnect is limited to only one connected master agent accessing one connected slave agent at a time.

There are two types of agents with which functional blocks are connected to an FPI interconnect:

- FPI master agent can request for access to the FPI interconnect, starting a FPI transaction when it is granted by the local BCU arbitration function
- FPI slave agent can only react and respond to FPI transactions initiated by FPI master agents in order to read or write internal registers of slave modules and for example memories

When an FPI master agent attempts to initiate a transaction on the FPI, it first signals a request for interconnect ownership to the local BCU. When the master agent is granted by the BCU it becomes the active master agent and starts the read or write transaction that consists of one or multiple address phase and data phase sequences. The slave agent addressed by the transaction address phase becomes the active FPI slave agent that responds with either the requested data transfer or with a transaction error.

FPI interconnect arbitration is performed by the BCU of the FPI. In case of a transaction error, the BCU signals an alarm to the Safety Management Unit (SMU) . Additionally it can generate an interrupt request and it can capture information about the transaction that lead to the transaction error.

For the alarm description please refer to the alarm mapping tables in the SMU functional block and the Safety Manual.

Related information

TC4Dx SMU alarm mapping tables on page 7233

7.3 Functional description

This section gives an overview of the on-chip FPI interconnect which is necessary for understanding behavior and debug information. This chapter describes also the Bus Control Unit (BCU) modules including arbitration, scheduling, prioritizing, error conditions, and debugging support.

7.3.1 FPI transaction types

This section describes the FPI transaction types.

Single transfers

Single transfers are byte, half-word (16-bit) and word (32-bit) transactions that target any slave connected to the system interconnect.

Block transfers

The FPI also supports transactions with 64-, 128- and 256-bit transferred data bits. Since the FPI is a 32-bit (word) interconnect, these transactions are executed as block transfers that are also called burst transactions or multi beat transactions.

Block transfers operate in principle in the same way as single transfers do, but one address phase is followed by multiple data phases. Block transfers can be composed of 2-word, 4-word, or 8-word transfers.

***Note**: Block transfer that are not started with a block size aligned start address are executed in wrap around addressing mode. Linear addressing mode is not supported by the current FPI version .*

Block transfers can be initiated directly through for example DMA channel configuration but also indirectly through CPU instructions. For example by the following TriCore™ instructions: LD.D, LD.DA, LD.DD, MOV.D, ST.D, ST.DA, ST.DD.

Atomic read-modify-write transfers

Atomic read modify write (RMW) transfers are generated by some TriCore™ instructions like, for example, LDMST, ST.T and SWAP.W and are executed through the interconnect with two single transfers which is a read followed by a write transfer. The read and write transfers of an atomic transfer are always locked and cannot be interrupted by another FPI master agent. Atomic transfers are also referenced as read-modify-write transfers.

***Note**: See also chapter ´FPI error handling´ for available FPI transfer types.*

***Note**: For the complete list of TriCore™ instructions that are executed through the interconnect with read-modify-write please refer the TriCore™ architecture manual.*

7.3.2 Reaction of a busy slave agent

If an FPI slave agent is busy at an incoming FPI transaction request, it can delay the execution of the FPI transaction through two different methods: Retry and Wait states.

- Retry: If the slave agent answers a transaction with Retry, the current transaction without data transfer is finished and the FPI is free for other transactions. The requesting master agent that has received a retry must send another transaction request and repeat the transaction. Retry is used by an FPI2SRI bridge that is still occupied by a previous access and does not know how long it will be occupied. Other FPI slaves do not use retry during normal application flow
- Wait States: If a slave agent requires several clocks to deliver a read data or to accept a write data, it can use wait states to extend the data phase

7.3.3 FPI address alignment rules

FPI address generation is compliant with the following rules:

- Half-word transactions must have a half-word aligned address (A[0] = 0_B). Half-word accesses using byte lanes 1 and 2 to transfer data are illegal
- Word transactions must always have word-aligned addresses (A[1:0] = 00_B)
- Block transactions must always have word-aligned start address
- Block transactions are executed with wrap around addressing mode in case of a non-block aligned start address
- Read-Modify-Write transactions must always have a word aligned start address

7.3.4 FPI basic operations

This section describes some basic transactions on the FPI.

Each transaction consists of one or more data transfers. Each data transfer consists of 3 phases that are executed sequentially on the FPI interconnect:

1. Request and grant phase:
   - An FPI master agent attempts to perform a read or write transfer and requests for FPI access. If the FPI interconnect is available, it is granted in the same clock cycle by the Bus Control Unit (BCU)

2. Address phase:
   - After the request or grant phase, the master puts the address on the FPI interconnect and all connected FPI slave agents check whether they are addressed for the following data phase. The addressed slave agent becomes the active slave agent for this transaction
3. Data phase:
   - In the data phase, in case of a write data phase master puts write data on the FPI which is taken over by the active FPI slave agent, and in case of a read data phase the slave agent puts the read data on the FPI which is taken over at the end of the data phase by the active master agent

Up to three transfers can be executed parallel on the FPI interconnect with different transfer phases:

- The data phase of a first transfer
- The address phase of a second transfer, the next data phase belongs to this transfer
- The request/grant phase, at the end of which the BCU decides which master agent to assign next unless it is part of a burst or RMW transaction that has already started

On the FPI interconnect the three phases that are in parallel on the FPI interconnect are synchronized with the FPI_RDY signal. All three phases end together with the FPI clock cycle in which FPI_RDY=1. The FPI_RDY signal can be used by the active FPI slave agent to extend a data phase, and through this also the parallel address and request phase, with wait states by signaling FPI_RDY=0. If no slave agent is active FPI_RDY=1 is signaled by the FPI interconnect.

A maximum number of wait states, consecutive FPI clock cycles with FPI_RDY=0, before a transaction is terminated by the BCU with FPI Timeout can be configured in the BCU.

The figure below shows 3 fully pipelined transfers which means that there is no IDLE transfer in between these 3 transfers. Transfer 1 is granted. Transfers 2 and 3 show the conflict when two master agent requesting in parallel for FPI interconnect access and how the conflict is resolved. In the example, the FPI master agent of transfer 2 has a higher priority than the FPI master agent of transfer 3 and is therefore granted first.

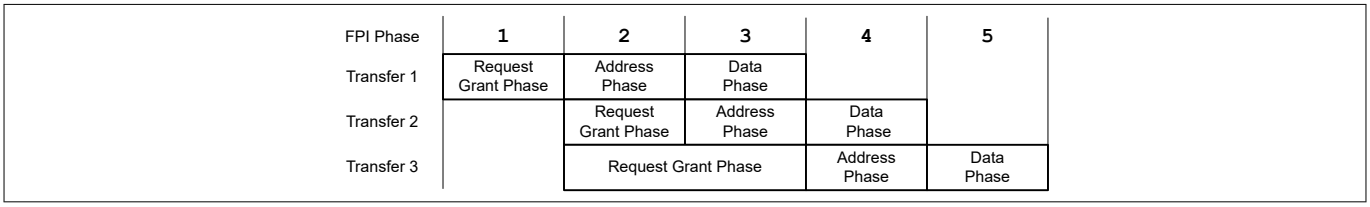


Figure 72 Basic FPI transactions

7.3.5 FPI interconnect arbitration

The arbitration unit of the BCU arbitrates among the master agents that are requesting for FPI access. The priority for each master agent connected to the FPI is defined in the BCU registers PRIOH and PRIOL. During arbitration, the FPI is granted to the requesting master agent with the highest priority. If no master agent request is pending, the FPI is granted to a default master.

The mapping of implemented master agents to the BCU registers PRIOH and PRIOL is described in the product specific BCU chapter.

The arbitration is implemented as a late arbitration scheme. This means that the decision which master agent will be granted next is always done at the last FPI clock cycle of a data phase, which is the cycle with FPI_RDY=1.

7.3.5.1 Configurable master agent priorities

Arbitration only becomes active if several master agents request interconnect access at the same time. If this happens the order of access is determined by the priority of the master agents or by the arbitration mechanisms.

- Each master agent connected to the FPI is assigned to a dedicated 4-bit priority bit field in the BCU registers PRIOH and PRIOL
- In addition, each PRIOH and PRIOL 4-bit field is assigned to an index in the range from 0 to 15:
  - PRIOL[3:0] → index 0 up to PRIOH[31:28] → index 15
  - The index is used to define the arbitration sequence within a round-robin group arbitration
- The value in a priority bit-field defines the related FPI master agent priority used for the priority driven FPI arbitration
- A lower priority number has a higher priority, 0 is the highest priority

7.3.5.2 Arbitration algorithms

The arbitration algorithm implemented in the BCU combines three layers of arbitration:

- Priority driven arbitration of master agents with a pending request with priority 0 - 15, 0 = highest priority
- Round-robin group arbitration mechanism on priority 4 covering all master agents assigned to priority 4
- Round-robin group arbitration mechanism on priority 8 covering all master agents assigned to priority 8
- Starvation Prevention mechanism that can increase master priorities during Starvation Prevention process

The default priority of each connected FPI master agent can be changed during run-time through its related bit-field in the registers PRIOH and PRIOL (see also section 'Changing master agent priorities' in this chapter).

Application software must be ensure that two FPI master agents do not have the same priority, with the exception of priority 8 and priority 4 which are the round-robin group priorities.

If more than one master agent is configured with the same priority but not with the round-robin group priorities 8 or 4, the arbitration will still work but the sequence how these master agents will be granted can not be guaranteed.

Priority driven arbitration (priority 0- priority 15)

The general arbitration algorithm is priority driven where priority 0 is the highest priority and 15 the lowest one. If multiple master agents are requesting, the master agent with the highest priority will win the next arbitration round (see also section 'Starvation prevention' in this chapter).

Round-robin groups (priority 4 and priority 8)

Priorities 8 and 4 are implemented as a round-robin groups. The two round-robin groups work independently from each other.

Example for round-robin group mechanism on priority 8, similar mechanism also applies to priority 4:

- The round-robin group algorithm arbitrates among all master agents configured with priority 8 that are requesting for interconnect access. The winner of the round-robin group arbitration will be treated as a normal master agent request with priority 8 within the priority driven arbitration algorithm
- Master agents assigned to priority 8 without pending request for interconnect access are not considered by the round robin mechanism and will not delay other master agents from being granted
- If only one master agent is mapped to a round-robin group priority, the master agent’s request will be treated as normal master agent request with priority 8 within the priority driven arbitration algorithm
- After the winner of a round-robin group has been granted, the requesting Master Agent configured on priority 8 with the next higher index number will be selected as the next winner of the round-robin group. If there is no requesting master agent with a higher index number in the round-robin group, the algorithm will start with the requesting master agent with the lowest index number in the round-robin group, going to the next higher index number and so on. The index numbers are defined in the registers PRIOH, PRIOL

Round-robin group control (round share mechanism)

Each round-robin group also provides a mechanism to prevent the master agents of the round-robin group in combination with master agents with a higher priority from excluding the master agents with low priority from arbitration for too long.

For this it can be configured per round-robin group after how many consecutive granted transactions in a round-robin group and from master agents with a higher priority one master agent request with low priority is granted.

This allows a fine granular adjustment to temporary interconnect overload situations to the application requirements. These are periods with a high master agent request density.

The round share mechanism of a round-robin group is disabled if the related register RRCTRL.RRx bit-field is configured with 0.

If not disabled, the Round Share mechanism of an round-robin group:

- Becomes active, starts to count, as soon as at least one master agent of the round-robin group or with a higher priority has a request pending AND at least one master agent with a lower priority has a request pending
- When active, it counts the consecutive granted transactions to the masters agents of the round-robin group and to master agents with higher priority
- Resets its grant counter if a master agent with lower priority than that of the round-robin group was granted
- Resets its grant counter if no master agent of the round-robin group or master agent with a higher priority has a request pending
- Allows a grant to a lower priority master agent when the configured number of successive granted transactions has been reached

In addition to the round share mechanism there is the starvation prevention mechanism that works independent of the round share mechanisms and provides a safety net for all non-granted master agents in a configured period of time.

Master agent priorities after reset

The default priority settings should be optimal for most applications. Where required, the arbitration setting can be adapted to the specific application needs (see also section 'Changing master agent priorities' in this chapter).

The master agent priorities are described in the device specific chapter, section BCU.

Changing master agent priorities

The FPI master agent priorities can be reconfigured at any time. The BCU arbitration mechanism always uses the current PRIOH and PRIOL configuration. This means that after a change in one of the PRIOH and PRIOL registers, the new master agent priorities take effect immediately and also affect the current pending master requests if more than one master agent requests access for the FPI.

Starvation prevention

Starvation prevention takes care that even requesting low priority master agents will be granted after a period, where the period length can be controlled by Bus control unit (BCU) control registers. Because of the priority driven arbitration algorithm, it is possible that a lower-priority master agent may never be granted if one or more higher-priority master agents continuously requesting for FPI access. To protect against starvation of lower-priority master agents, the starvation prevention mechanism of the BCU will detect such cases and momentarily raise the priority of the lower-priority requester to the highest priority (above all other priorities), thereby guaranteeing it access.

Starvation prevention employs a counter that is decremented with each clock cycle on the related FPI interconnect, for example on SPB with fSPB and CPB with fCPB. When the counter is counted down to zero it is always re-loaded with the starvation period value in the CON.SPC bit-field. When the counter is counted down

to zero, for each active master agent request a request flag is stored in the BCU. This flag is cleared automatically when a master agent is granted by the BCU.

When the next counter period is finished and not all request flags where cleared, a starvation event happened. These master agents related to the remaining request flags will now be set to the highest priority and will be granted in the order of their configured priority ranking.

If a master agent that is processing its transaction under starvation condition is retried by the addressed slave (means: slave was busy, transaction was not executed), its corresponding request flag is automatically set again.

Starvation protection is permanently enabled. The sample period of the counter is programmed through register bit-field CON.SPC. The configured value in the SPC bit-field must be larger than the number of master agents. Its reset value is FFF_H.

7.3.5.3 Default master mechanism

If no master agent is requesting for FPI access, the FPI master agent that was most recently granted will be granted as the default master. An exception is if the master agent that was most recently granted is now switched off or it is in Request delay mode. In this case, the BCU is the default master until another master has requested and was granted.

Any master agent can act as default master.

The BCU has no functional master agent interface and therefore has no request or grant signal. However, the BCU can act as the default master on the local interconnect. If the BCU acts as default master, it gives itself a default master grant internally that is not visible outside the BCU nor visible through the BCU error capture registers.

After a reset the BCU is the default master until a master agent is requesting for interconnect access.

A disabled master agent and a master agent in Request delay mode is not granted as default master. In this situation, the BCU becomes default master until a master agent is granted.

Related information

QoS master agent interconnect usage control on page 1099

7.3.6 FPI error handling

When an FPI error occurs on an FPI interconnect, its BCU captures and stores information about the erroneous condition, signals an alarm to the SMU and generates an interrupt if enabled to do so. The FPI error conditions that force an error-capture are:

- Error acknowledge: A FPI transaction is terminated by the slave agent in a data phase with FPI error acknowledge. A slave agent may terminate a transaction with FPI Error acknowledge due to an access protection violation, access with an op-code not supported for this address or an access to a reserved system address that is not assigned to register or memory
- Access to a Reserved system address: No FPI slave agent responds to a transaction request
- Timeout: A slave does not respond to a transaction request within a certain time window. The number of interconnect clock cycles that can elapse until a FPI time-out is generated is defined by bit-field CON.TOUT

When a transaction causes an error, the address and data phase signals of the transaction causing the error can be captured by the BCU and traced through On chip multi tool support (OCMTS) if enabled for capturing and tracing.

Capturing and tracing of Cyber security real-time module (CSRM) transaction information (CS_VALID=1) is controlled by CSRM through the functional block Cyber security system control unit (CSCU). See also register CSSTAT.

- The Error Address Capture Register (EADD) stores the 32-bit FPI address information that has been captured during the erroneous FPI transaction

- The Error Data Capture Registers (EDAT) stores the 32-bit FPI data information that has been captured during the erroneous FPI transaction
- The Error Control Capture Register (ECON) stores status information of the error event

In case of an EDC error the BCU provides additional information in the ALSTAT registers to identify the connected FPI slave or master interface where the FPI EDC error was detected.

If more than one FPI transaction generates a error, only the first error is captured. After a error has been captured, the capture mechanism must be released again by software. The lock is removed by reading the register ECON which clears the ECON.ERRCNT bit-field.

***Note**: It is recommended to read in a debug session register ECON last as this removes the lock and a new error can already modify the content of the other two registers EDAT and EADD.*

If a write transaction from a master function that is not directly connected to an FPI instance causes an error on this FPI instance, for example a posted write transaction forwarded through an S2F bridge, the originating master is not informed about this error.

With each error-and error capture event,

- A service request can be generated if enabled and configured in the corresponding service request control register in the Interrupt Router function. In the interrupt router, the corresponding service request control registers are named SRC_<BCU>_NCS, SRC_<BCU>_CS where <BCU> is the placeholder for the respective BCU instance name: SBCU, CBCU, COMBCU, CSBCU, TBCU and RBCU
- An alarm is signaled to the SMU (Alarm_CS or Alarm_NCS)

7.3.6.1 Interpreting the BCU control register error information

Although the address and data values captured in registers EADD and EDAT, respectively, are self-explanatory, the captured FPI control information needs some more explanation.

Register ECON captures the state of the read (RDN), write (WRN), Supervisor Mode (SVM), acknowledge (ACK), ready (RDY), abort (ABT), timeout (TOUT), master identification lines (TAG, VM, PRS, VM_VALID, PRS_VALID), cyber security information (CS_VALID) and transaction operation code (OPC) lines of the FPI .

The SVM signal is set to 1 for an access in Supervisor Mode and set to 0 for an access in User Mode.

The timeout signal indicates when a slave agent has inserted more wait states than defined by BCU_TOUT in a data phase and the BCU has aborted the data phase with signaling TOUT=1.

An acknowledge code has to be driven by the selected slave agent during each data cycle of an access. These codes are listed below in table ´FPI acknowledge codes´.

Table 600 FPI acknowledge codes

Code (ACK)	Description
00_B	NSC: No special condition
01_B	Reserved
10_B	RTY: Retry. Slave can currently not respond to the access. Master agent must repeat the access later
11_B	ERR: Error, last data cycle is aborted

Transactions on the FPI are classified through a 4-bit operation code.

Table 601 FPI operation codes (OPC)

OPC	Description
0000_B	Single Byte Transfer (8-bit)

(table continues...)

Table 601 (continued) FPI operation codes (OPC)

OPC	Description
0001_B	Single Half-Word Transfer (16-bit)
0010_B	Single Word Transfer (32-bit)
0100_B	2-Word Block Transfer
0101_B	4-Word Block Transfer
0110_B	8-Word Block Transfer
1111	No operation
0011_B, 0111_B, 1000_B - 1110_B	Reserved

Related information

FPI error handling on page 1093

7.3.6.2 FPI error capture mechanism and BCU interrupts

The BCU provides two interfaces for observation and software debugging:

- A capture mechanism captures the first transaction with a fatal error and a trace port which sends trace information to the MCDS. Fatal error means that an intended transaction was not executed because it leads to an FPI Error, an FPI timeout or when no slave responding to an access. After a transaction information was captured, software can release the mechanism again
- A trace port which sends trace information to the MCDS

The visibility of CS related transactions through these interfaces is controlled by the CSRM through the CSAV signals. As long CSRM does not enable the capturing of CS related transactions through CSAV=Full or CSAV=Limited, only transactions with CS_VALID=0 can be captured in the BCU and traced through OCMTS.

The interfaces cover all relevant transaction information, including:

- Master agent grant signal status
- 32-bit address
- OP-code, Read and Write information
- TAG ID, VM, PRS
- Supervisor (SV) and Security (CS_VALID) information
- 32-bit data
- Slave agent acknowledge code
- FPI protocol control signals of the transactions address phase and data phase

CSRM control of the debug and trace interfaces:

The visibility of CS related transactions is controlled by CSRM tracing through a two bit enable signal (CSAV) driven from the CSRM block. The signal value decode is shown in the Table 602 table.

Table 602 CSAV Decode

Value	Meaning
0	No CSRM tracing (default reset value)
1	Limited CSRM tracing
2	Reserved - same behavior as 0
3	Full CSRM tracing

The definition of limited and full tracing for errored and non-errored transactions are shown in tables Table 603 and Table 604 .

Table 603 CSAV levels for the trace of the local FPI interconnect traffic

Level	Behavior for CS transactions
Full	Fully visible in the trace
Limited	Visible in the trace, but address and data are 0
No	Not visible in the trace

Table 604 CSAV levels for BCU error capturing on fatal transaction errors

Level	Behavior for CS transactions
Full	Transactions are captured
Limited	Transactions are not captured
No	Transactions are not captured

Related information

BCU interrupts on page 1096
BCU alarms to SMU on page 1096

7.3.7 BCU interrupts

Each BCU provides multiple interrupts. The interrupts are signaled to the functional block Interrupt router (IR) where each interrupt is assigned to dedicated Service request control register in the interrupt router.

The names of the BCU related service request control registers in the IR for the BCU instance <BCU> :

- SRC_<BCU>CS: Transaction with CS_VALID = 1 that leads to an FPI error, FPI timeout or access to a reserved address
- SRC_<BCU>NCS: Transaction with CS_VALID = 0 that leads to an FPI error, FPI timeout or access to a reserved address
- SRC_<BCU>QOS: Interrupt is signaled whenever one or multiple of the QOSMx_STAT.MSB bits changes from 0 to 1. The QoS support is not implemented in all BCU instances
- SRC_QOSARPx; x=0-7: Interrupts signaled by a BCU with QoS ARP function whenever ARPx_CTRL.STAT bit changes from 0 to 1. The QOSARPx interrupt outputs from all BCUs and SRI interconnects with QoS observing function are bit-wise OR combined and the resulting interrupt trigger signals are mapped in the IR to SRC_QOSARPx, x=0-7. The QoS support is not implemented in all BCU instances

A BCU can be configured to generate interrupts for fatal errors of Secure (CS_VALID=1) and Non-Secure (CS_VALID=0) transaction on detection of an FPI error situation an FPI timeout situation or an FPI access to a reserved address.

***Note**: The product specific FPI chapters show the available FPI interconnects, the corresponding BCU control registers and the information which BCU is implemented with QoS functionality.*

Related information

BCU alarms to SMU on page 1096

7.3.8 BCU alarms to SMU

BCU alarm signals to SMU

Each instance of the FPI interconnect provides two alarm signals:

- Alarm_CS for an alarm event detected for a security related transaction with CS_VALID=1 in the transaction address phase
- Alarm_NCS for an alarm event related to a non security related transaction with CS_VALID=0 in the transaction address phase

On detection of a FPI transaction terminated with FPI error acknowledge or a FPI timeout on a security related transaction (CS_VALID = 1):

- Alarm_CS is signaled to the SMU_CS

On detection of a FPI transaction terminated with FPI error acknowledge or a FPI timeout on a non security related transaction (CS_VALID = 0):

- Alarm_NCS is signaled to the SMU_SAFEx

On detection of a FPI EDC error during a FPI transaction the CS_VALID signal can no longer be considered valid, safety related transaction is assumed:

- Alarm_NCS is signaled to the SMU_SAFEx

The alarm signals are covering the following three events:

- Alarm on detection of a FPI transaction finished with FPI error (e.g. access protection violation, access to reserved address)
- Alarm on detection of a FPI timeout situation
- Alarm on detection of a FPI EDC error

Each BCU enables the application software to test the EDC decoders of all connected FPI master and slave agents through the FPI Error Generation (FEGEN) register in the BCU. See also related information.

The Alarm_CS and Alarm_NCS signal paths to the SMU can be tested in different ways with access to an address related to the respective FPI instance. For example by an CPU access to reserved address that results in an FPI error and an Alarm_NCS and by a CPUcs access to reserved address that results in an FPI error and an Alarm_CS.

Related information

FPI error injection on page 1108

7.3.9 Interconnect performance measurement and analysis

Real-time measurement of the interconnect performance provides useful insights to system developers, architects, application developers.

The BCU provides a set of relevant events to the MCDS that can be used to measure different performance aspects of the local interconnect without any real-time effect on execution.

The performance events that can be measured are described in the table 'TS16_BCU,
BCU Performance Events' .

The forwarding of the BCU performance events is controlled by the OCS register. The signals are then forwarded to the MCDS module via the OTGB interface.

The performance measurement events can be used to measure basic parameters such as:

- Data Phase count. The counter is incremented on end of a data phases with NSC, ERR or RTY. NOP transactions not counted
- Local FPI clock cycles count. The counter is incremented with local FPI clock cycles
- Wait state cycle count. The counter is incremented on every wait state clock cycle
- Retry Phase count. The counter is incremented on every data phase ending with RTY
- Interconnect contention cycles. The counter is incremented on every clock cycle where x requests are pending in parallel to the actual transaction. Number of parallel requests X is configurable. A single request that is granted in a request phase is not counted
- Default master data phases. The counter is incremented with any data phase of transaction in default master mode. NOP transactions not counted

- Starvation prevention active. The counter is incremented with every interconnect clock cycle where starvation prevention is active
- Starvation prevention transaction. Can be used to count the number of transactions granted by starvation prevention mechanism

7.3.9.1 Typical performance event usage

The performance parameters can be mapped in the Multi Core Debug Support (MCDS) module to its counter functions.

Typically two parameters are always counted for base line measurement:

- Local FPI clock cycle count (CCNT)
- Data Phase count (TCNT)

Additional parameters can be counted to derive the desired performance values, for example:

- Interconnect load
- Transaction rate
- Wait states per transaction
- Transaction retry rate
- Interconnect contention rate
- Starvation prevention active rate

7.3.9.2 BCU performance parameters

The following list describes the local FPI interconnect events that can be mapped to counters in MCDS module. Before they can be used in MCDS, they must be activated using the OCS.OTGM function. The list also describes how the event affects a counter selected in the MCDS.

Details of the MCDS counter functions, for example counter width, counter overflow, linkage of counters, are described in the OCMTS chapter.

Table 605 TS16_BCU, BCU performance events

Bit	Event function	Description
0	Data_phase_count (DPC)	Counter is incremented whenever a transaction data phase ends with NSC, ERR or RTY. NOP transactions not counted. Aborted Data Phases are not counted
1	Waits_state_count (WSC)	Counter is incremented with any local FPI clock cycle where a transaction is delayed by wait state clock cycle
2	Transaction_count (TC)	Counter is incremented with any transaction. NOP transactions not counted
3	Transaction_retry_count (TRC)	Counter is incremented when a data phase is terminated with RTY by a slave agent which means that the slave agent was busy, transaction not executed

(table continues...)

Table 605 (continued) TS16_BCU, BCU performance events

Bit	Event function	Description
4	Transaction_default_master_count (TDMC)	Counter is incremented when a transaction is started in Default Master mode. NOP transactions not counted
5	Transaction_starvation_prevention_count (TSPC)	Counter is incremented when a transaction is granted by the starvation prevention mechanism
6	Transaction_aborted_count (TAC)	Counter is incremented when a transaction is aborted by master agent
7	Transaction_Error_Count (TEC)	Counter is incremented when a transaction is terminated with ERR
8	Interconnect_contention_cycles (ICC)	Counter is incremented on every clock cycle where >= x requests are pending in parallel to the actual transaction. Number of parallel requests x is configurable through register PSELECT.PMAR. A single request that is granted in a request phase is not counted
9	Starvation_prevention_active_cycles (SPAC)	Counter is incremented with every local FPI clock cycle at which the starvation prevention mechanism is active which means that one or more transactions are pending for grant by the starvation prevention mechanism
10	QOS_event_cycles (QEC)	Counter is incremented whenever one of the QOSMx_CTRL.MSB bits changes from 0 to 1. This means that master agent x is or was in Request Delay Mode because it has consumed its assigned FPI clock cycles in a QoS time interval.
11	Unused_FPI_cycles (UFC)	Counter is incremented with any local FPI clock cycle in which the local FPI interconnect is not used for a transfer.
12 - 13	Reserved	Counter is not counting.
14	PSELECT	The function of this bit will be defined by the register PSELECT.PSEL allows to map an event from this list to this bit
15	Local_FPI_clock (LFC)	This signal is active for one f_{TPB} clock cycle per local FPI clock cycle. It is used by the tool to determine the clock ratio between local FPI and MCDS to evaluate the other TS16_BCU event information.

7.3.10 QoS master agent interconnect usage control

The QoS control mechanism enables the application software to assign a master agent a number of interconnect clock cycles that it can consume in a QoS interval before its activities on the interconnect are until the end of the QoS interval. The way in which the master agent is is also configurableby the application software.

- For each FPI interconnect a QoS time interval (QOSICNT.QOSINT) can be defined with the QOSICTRL.QOSINT register in multiples of 16 local interconnect clock cycles. The time interval counter (QOSISTAT.QOSICNT) width is 20-bit
- For each connected master agent a master interconnect cycle count can be defined with the QOSMx_CTRL.MICC register in multiples of 16 local FPI clock cycles. The QOSMx_CTRL.MICC is used to define the number of FPI clock cycles a master agent can consume on the interconnect within a QoS time interval before the request delay mechanism becomes active for this master agent until the end of the QoS time interval. The master cycle counter (QOSMx_STAT.MCNT) width is 19-bit
- For each connected master agent a master request delay (QOSMx_CTRL.MRD) can be defined with the QOSMx_CTRL.MRD register (11-bit). The MRD defines the number of local clocks by which a request of the master is delayed in request delay mode before it is considered in the arbitration
- If the QoS control mechanism is available in a FPI interconnect instance, it can be activated there separately for each connected master agent
- Master agent transactions are considered independently of the transactions CS_VALID status. This means that CSAV has no influence here
- The FPI master agent of a SRI2FPI bridge is treated like any other FPI master agent. An FPI2SRI does not have its own transaction ID but transparently forwards the transaction ID of accesses through the bridge

QoS time interval

A QoS time interval starts as soon as QOSICTRL.QOSEN is enabled.

At the start of a QoS time interval, QOSISTAT.QOSICNT is initialized with the value from QOS_ICTRL.QOSINT.

As long as the QOSICTRL.QOSEN is set QoS time intervals with the configured duration (QOSICTRL.QOSINT) are generated.

At the start of a QoS time interval, for each master agent for which the QoS control feature is enabled through QOSMx_STAT.CTRL.MEN=1_B,

- QOSMx_STAT.MSTAT is set to0_B which means ´No request delay mode´
- QOSMx_CTRL.MICC register value is loaded into QOSMx_STAT.MCNT

***Note**: The assignment of the connected master agents to an index ´x´ is defined in the product-specific registers PRIOH, PRIOL*

Measuring the master agent FPI cycle consumption during a QoS interval

During a QoS time interval the mechanism counts for each connected master agent for which the QoS usage control function is enabled, the number of data phase FPI clock cycles the master agent uses the local FPI interconnect within the time interval. This includes also the data phase wait state clock cycles.

Usage here means that the FPI interconnect cannot be used by other master agents for data transmission. A master agent in the ´No request delay mode (QOSMx_STAT.MSTAT=0 while QOSMx_STAT.MCNT!=0) has not consumed the number of interconnect clock cycles assigned to it. Its requests for FPI access are taken into account without delay by the arbitration.

Master in request delay mode

As soon as a master agent has consumed its allocated interconnect cycles within the current QoS time interval, which means QOSISTAT.QOSICNT!=0_B and QOSMx_STAT.MCNT=0_B and QOSMx_STAT.MSTAT=0_B, the hardware sets QOSMx_STAT.MSTAT=1_B and the master agent is arbitrated in request delay mode until the end of the current QoS time interval (QOSISTAT.QOSICNT=0_B).

In request delay mode (QOSMx_STAT.MSTAT=1_B), each request of the master for interconnect access is delayed by the number of clocks configured in the master request delay register (QOSMx_CTRL.MRD) before the request is forwarded to the arbiter and there arbitrated according to configured master agents priority. This reduces the number of conflicts on the FPI interconnect of other master agents with the master agent in request delay

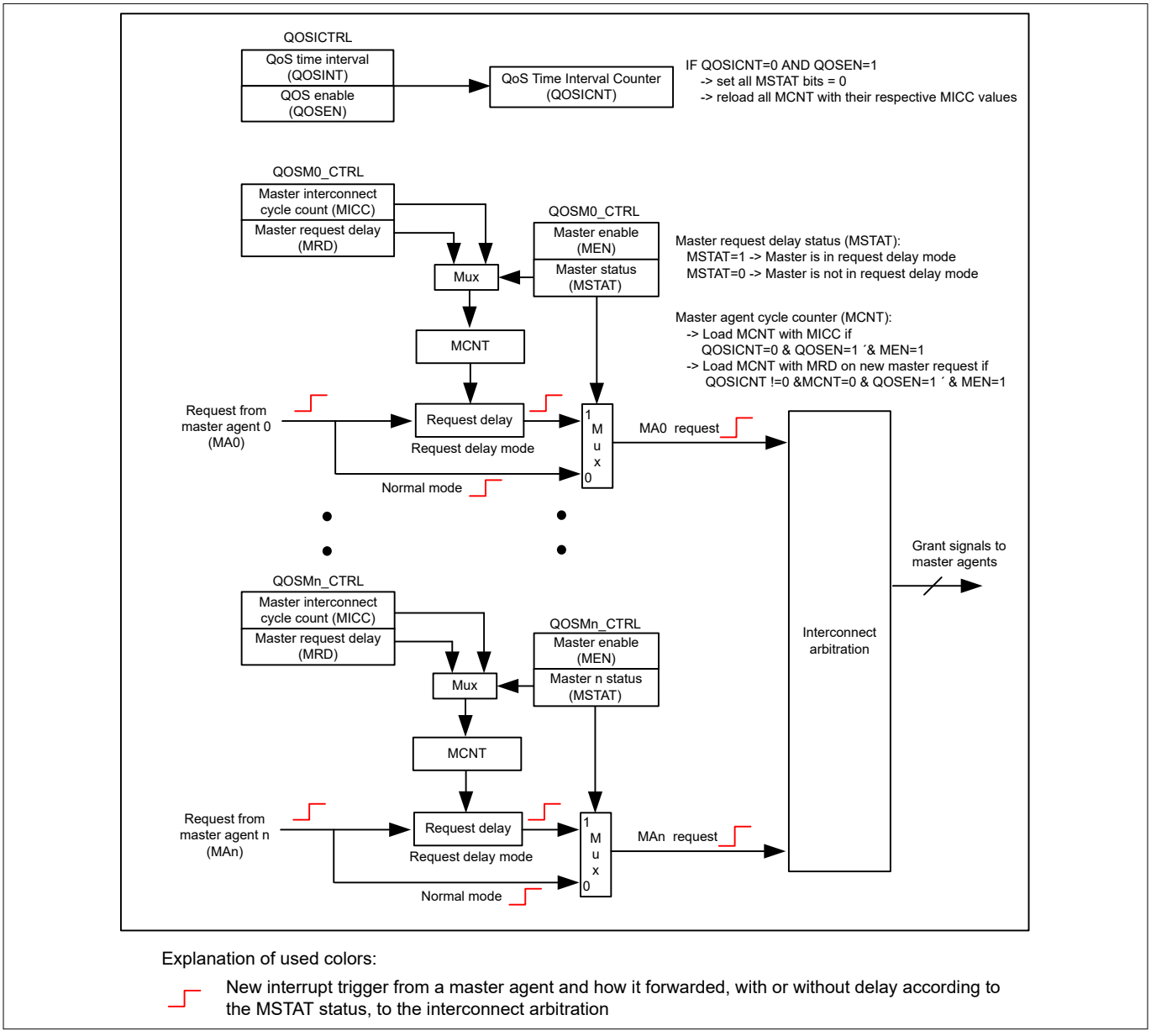
mode at the interconnect. Nevertheless, a reduced access rate at the interconnect can still be enabled for the master in request delay mode.

End of QoS interval

At the end of an QoS time interval (QOSISTAT.QOSICNT=0_B),

- all MSTAT bits are cleared (QOSMx_STAT.MSTAT=0_B)
- all master agents QOSMx_STAT.MCNT counters are re-loaded with the respective Master interconnect cycle count (QOSMx_CTRL.MICC) value
- the QoS time interval counter (QOSISTAT.QOSICNT) is loaded with the QOSICTRL.QOSINT value
- if QOSMx_STAT.MSTAT is set to 1_B at the end of the time interval, QOSMx_STAT.MSB is set to 1_B

A QoS control interrupt is signaled to the related SRC_BCUQOS service request node in the IR whenever a QOSMx_STAT.MSB bit changes from 0 to 1



Explanation of used colors:

New interrupt trigger from a master agent and how it forwarded, with or without delay according to the MSTAT status, to the interconnect arbitration

Figure 73 QoS: Structure of the request delay mechanism control registers

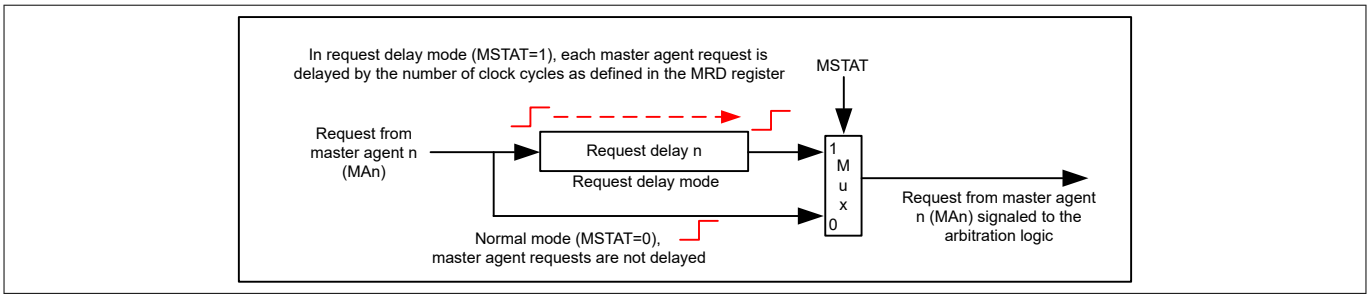


Figure 74 QoS: Structure of the request delay mode

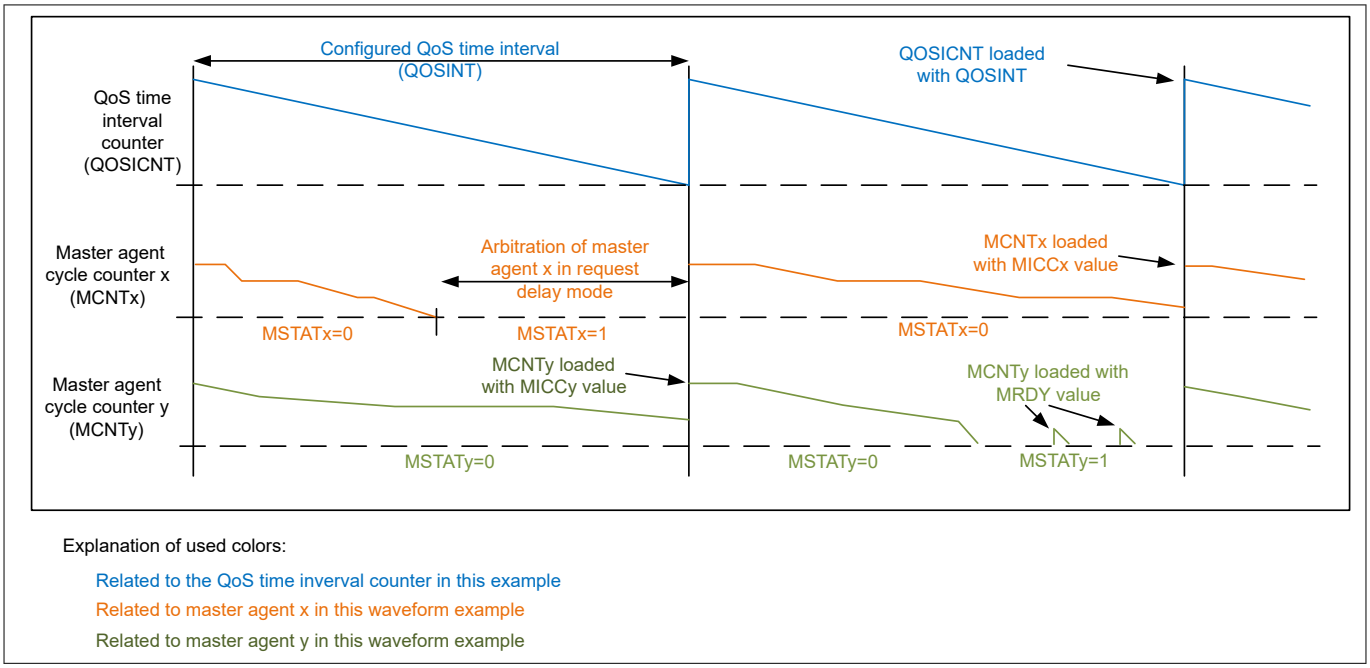


Figure 75 QoS: Temporal description of the request delay mechanism

7.3.11 Quality of Service application interconnect usage observation

The QoS observation mechanism enables the system software to monitor the consumed interconnect cycles at key resources for up to 8 application partitions within a QoS time interval.

The application partitions can be defined by assigning master agent functions through its TAG-ID and PRS encoding to the application resource partition functions (ARPx).

For each ARP function:

- The system can define the usage threshold for it
- The interconnect cycles consumed by the assigned master agent functions within a QoS interval are logged
- If the consumed cycles within a QoS time interval reaches the defined usage threshold, an interrupt is signaled at the end of the interval for this ARP function, the cycles consumed until the end of the interval is stored in a capture register and the capture register is locked

***Note**: Setting the threshold value to 0 is considered illegal and should be avoided.*

QoS time interval

A QoS time interval starts as soon as QOS_ICTRL.QOSEN is enabled. At the start of a QoS time interval, QOS_ICNT.QOSICNT is initialized with the value from QOS_ICTRL.QOSINT and partition cycle counter registers

are cleared and set to 0. As long as the QOS_ICTRL.QOSEN is set -QoS time intervals with the configured duration (QOS_ICTRL.QOSINT) are generated.

Measuring the cycle consumption during a QoS interval

During a QoS time interval each ARP mechanism counts for the master agent function IDs assigned to the ARP function the number of clocks the master agents occupies the observed resource within the time interval by counting the number of data phase clock cycles.

At the FPI the observed shared resource is the local FPI interconnect, at the SRI it is the SRI respective slave agent to which the ARP function belongs.

Interconnect usage includes the data phase along with wait states, where ongoing activity from a master prevents other masters from using the resource.

***Note**: The interconnect usage of CS-related transactions (with SM= 1) is only taken into account if CSAV = full or CSAV=limited.*

End of QoS interval

At the end of a QoS Interval (QOSICNT=0), the cycles consumed by the assigned master agent functions is copied into the Partition cycle count capture register of the respective ARP. For each ARP function where the interconnect cycles consumed by the assigned master agent functions is above the value configured in its Partition threshold register, an interrupt is signaled and the Partition cycle count capture register is locked after update.

The value of the locked partition cycle counter capture register is then not updated by the hardware until it is unlocked by software.

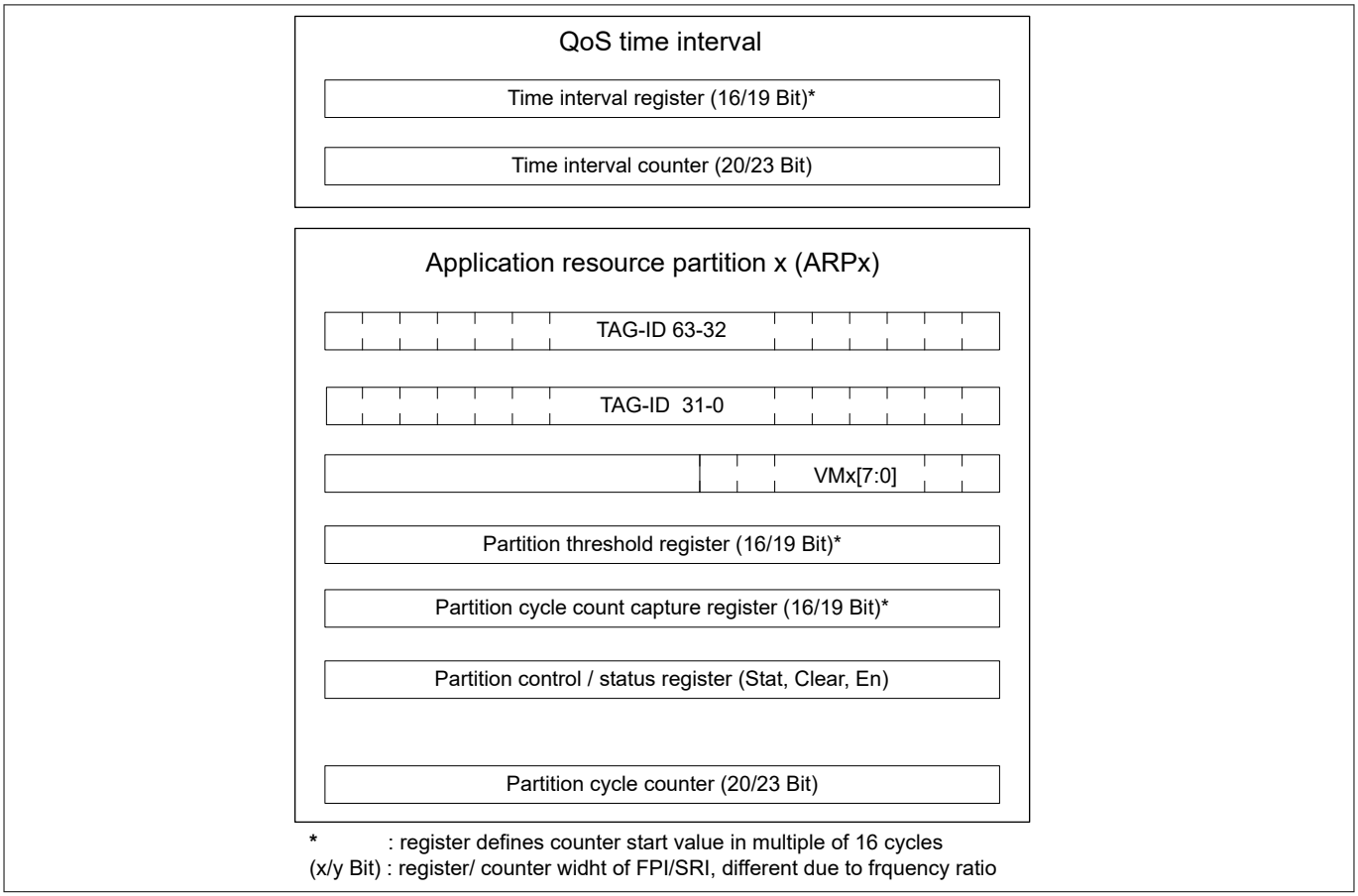


Figure 76 QoS time interval and ARPx register structure

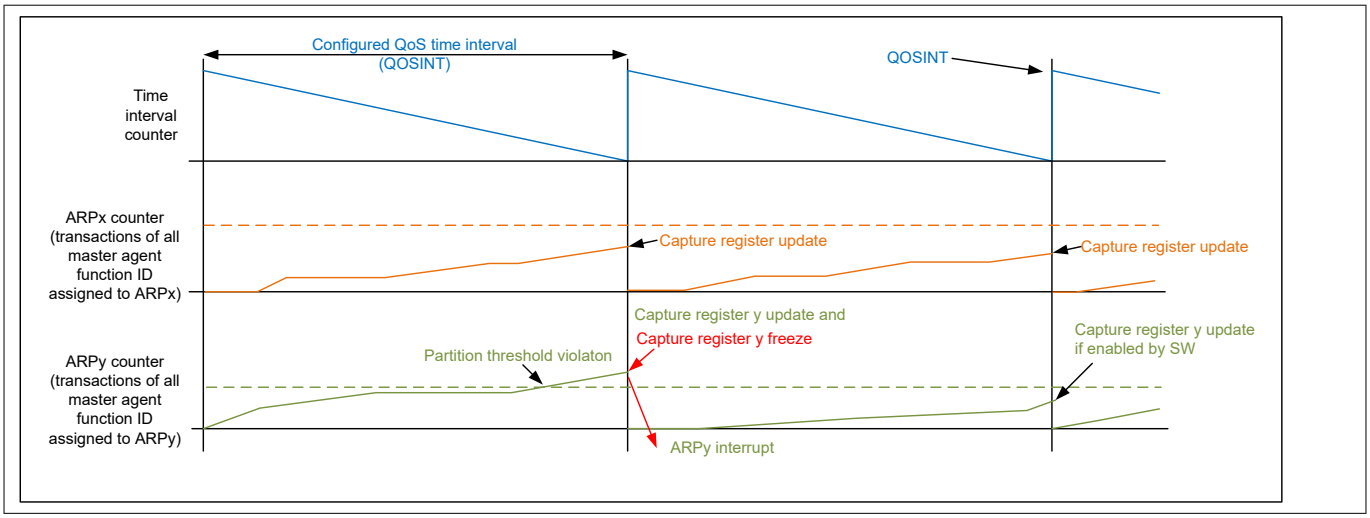


Figure 77 Temporal description of the ARP observation mechanism

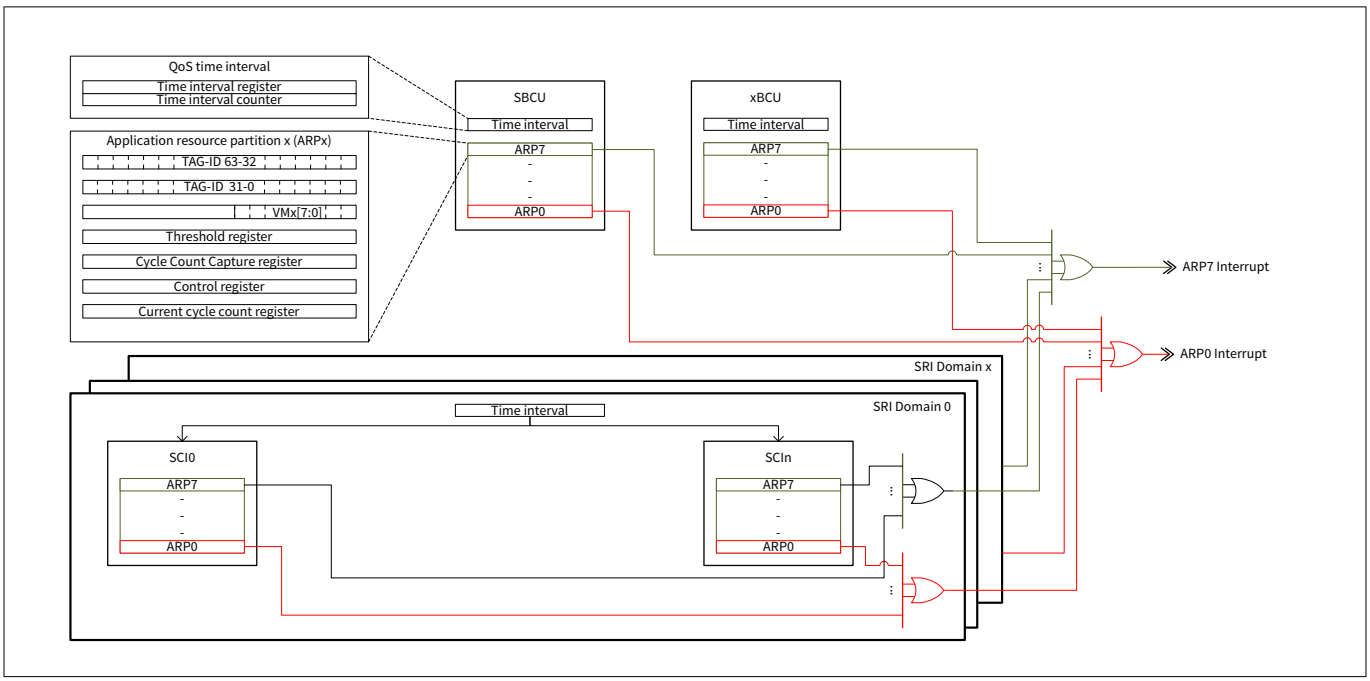


Figure 78 Hardware connection of the local ARP events to 8 ARP system interrupts

7.3.12 FPI integrity support

All transactions through the FPI interconnect are protected with an error detection mechanism (EDC). The mechanism is defined and implemented in order to detect errors on the signaled information in between active FPI master and slave agents during a transaction for example:

- Transaction address phase errors
- Transaction data phase errors
- Errors in the transmitted Data
- Situations where no slave agent, the wrong slave agent or multiple slave agents respond to a transaction
- Situations where multiple master agents are initiating transactions in parallel
- Situations where a granted master agent is not initiating a transaction

- Active master agent: master agent that initiates a transaction
- Active slave agent: slave agent that is selected in the Address Phase

The FPI EDC is distributed. Each master and slave agent performs the EDC check independently and signals a detected error to the interconnect fabric. The fabric can only forward one detected error per clock pulse to the BCU. If more than one error is signaled by the master and slave agents in a clock cycle, only the ALCTRL.PALOV bit is set. This scenario does not affect errors in address and data phases. However, it can be generated by errors in some FPI control signals.

The FPI implementation provides at any time the FPI address, data and control signals are driven by exactly one master agent or slave agent as required by the FPI specification. The active FPI slave and master agents drive the signals with its output enable signals. For the groups of the respective output enable signals, the BCU checks whether there are no or several active output enable signals at any time, reports this as an alarm and sets the corresponding ALSTAT register bits. This is done permanently by the BCU in parallel to the master and slave agents EDC checks. On detection of such an event (no or multiple output enables active), the related bits are set in the ALSTAT3 register:

- A_EN, no or multiple FPI address phase output enables active, mapped to ALSTAT3.AL0
- ABORT_EN, no or multiple FPI abort output enables active, mapped to ALSTAT3.AL1
- ACK_EN, no or multiple FPI acknowledge and FPI RDY output enables active, mapped to ALSTAT3.AL2
- D_EN, no or multiple FPI data phase output enables active, mapped to ALSTAT3.AL3

On detection of an FPI EDC error the Bus Control Unit (BCU) signals an Alarm_NCS to the functional block Safety management unit (SMU). The BCU ALSTAT registers provide detailed information about which FPI master or slave agent detected the EDC error.

The implemented solution ensures that only relevant errors will generate an alarm.

This includes:

- The interconnect signals from the active master agent through the interconnect
- The interconnect signals from the active slave through the interconnect
- General control signals during a transaction to detect erroneous active slave and master agent interfaces
- FPI_RDY = 1 is used as qualifier for valid Data Phase and Address Phase ECC information

***Note**:* *In this implementation, the FPI integrity support only offers error detection. Error correction codes (ECC) are used for extended error detection but not for error correction.*

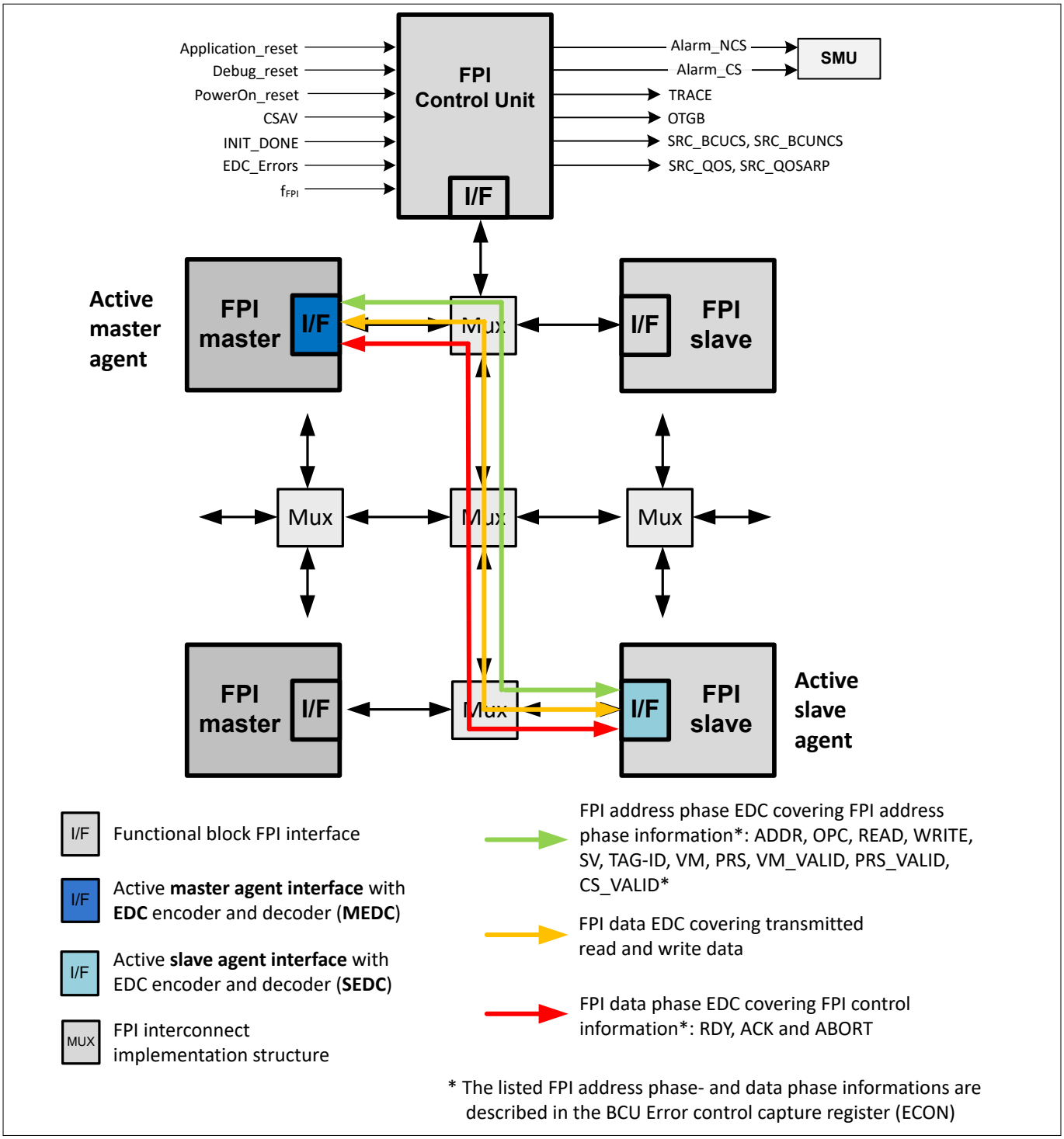


Figure 79 FPI EDC: Protection of active master agent to active slave data path through interconnect fabric

7.3.12.1 Safety support

BCU alarm and interrupts

Each FPI interconnect instances is implemented with one dedicated BCU. Each BCU provides:

- Two alarm signals ALARM_CS and ALARM_NCS to the functional block SMU

- Two interrupt signals SRC_CS and SRC_NCS to the functional block IR
- Additional 9 interrupt signals if the BCU is implemented with QoS support (SRC_QOS and SRC_QOSARP[7:0]). Each BCU with QoS support has a dedicated SRC_QOS service request node in the IR while the SRC_QOSARP[7:0] interrupts are combined with all other BCUs and SRI SCIs SRC_QOSARP[7:0] signals and mapped to one set of SRC_QOSARP[7:0] service request nodes in the IR

Capturing of FPI transaction error information

On detection of a FPI error and FPI timeout the BCU signals an alarm to the SMU. See chapter ´FPI error handling´.

FPI starvation prevention

The FPI arbitration algorithm in the BCU provides a starvation prevention mechanism. This feature of the BCU ensures that even requesting low priority master agents will be granted after a period, where the period length is defined through BCU control registers. See chapter ´Arbitration algorithmss´.

The starvation prevention is permanently enabled.

FPI Timeout

Each BCU provides a FPI timeout detection mechanism for its FPI interconnect instance. This mechanism detects if a slave agent does not finish a data phase within a configurable time window. The timeout mechanism is also implemented in the FPI protocol. A dedicated FPI TIMEOUT protocol signal forces the active master and active slave agent to release the FPI interconnect if the BCU has detected a timeout situation. On detection of timeout situation the BCU signals an alarm to the SMU (Alarm_NCS or Alarm_CS).

Register access protection

The BCU control registers are protected by the register access protection. On detection of an access protection violation the BCU terminates the transaction with FPI error and signals an Alarm_NCS or Alarm_CS to the SMU.

Access to reserved FPI address (default slave mechanism)

As long no slave agent is active, the FPI interconnect fabric implementation shows the following FPI slave agent control and EDC information:

- FPI_RDY = 1
- FPI_ACK = 11. This is equal to a slave FPI error acknowledge
- FPI_DP_ECC = 111. This is an invalid FPI EDC for the above RDY and ACK default information on the interconnect

This means that with these default values the following error situations can trigger an EDC alarm for the transaction of the active master agent although no EDC relevant signal error has occurred:

- The master agent access a reserved address where no slave agent becomes active. As a result the BCU signals alarm to the SMU
- The addressed FPI slave does not become active. As a result the BCU signals alarm to the SMU
- A master agent does not react to the default master grant and is not driving the data phase signals during the default master data phase while the next requesting and granted master agent looks at FPI_RDY to identify the start and end of its address phase. This means that an EDC error in a default master phase is only signaled as an alarm if it affects a following real transaction. This situation can result in FPI EDC alarms from the next transaction active master agent and from the active slave interface to BCU, where the BCU signals an Alarm_CS or Alarm_NCS to the SMU

In these cases two errors are displayed by the BCU: a FPI error acknowledge with the captured data and the EDC alarm in the respective ALSTAT register.

Related information

BCU alarms to SMU on page 1096

BCU interrupts on page 1096

FPI error handling on page 1093

7.3.12.2 FPI EDC overview

The FPI EDC covers the FPI address, FPI data and control signals from a active FPI master agent output to the active FPI slave agent input during a transaction.

The FPI EDC is implemented as end to end protection in between the master agents and the slave agents connected to one FPI instance. The FPI EDC encoder and decoder functions are implemented in the master and slave agents:

- A Master EDC encoder and decoder module (MEDM) is implemented in each master agent
- A Slave EDC encoder decoder module (SEDM) is implemented in each slave agent

The active master agent and slave agent perform an EDC check of the received transaction information at the relevant times of the address phases and data phases of the transaction.

On detection of EDC error the master agent or slave agent signals an EDC alarm to the local BCU.

Each FPI interconnect and the related local Interconnect control unit (BCU) includes:

- A mechanism to inject EDC errors and to localize detected errors
- A mechanism to detect error situations where more than master agent or more than one slave agent becomes active at a time at the same FPI interconnect
- A mechanism to detect EDC error situations where a granted FPI master agent does not become active

***Note**: The FPI EDC is not working in the first 4 FPI clock cycles after a reset of the FPI interconnect.*

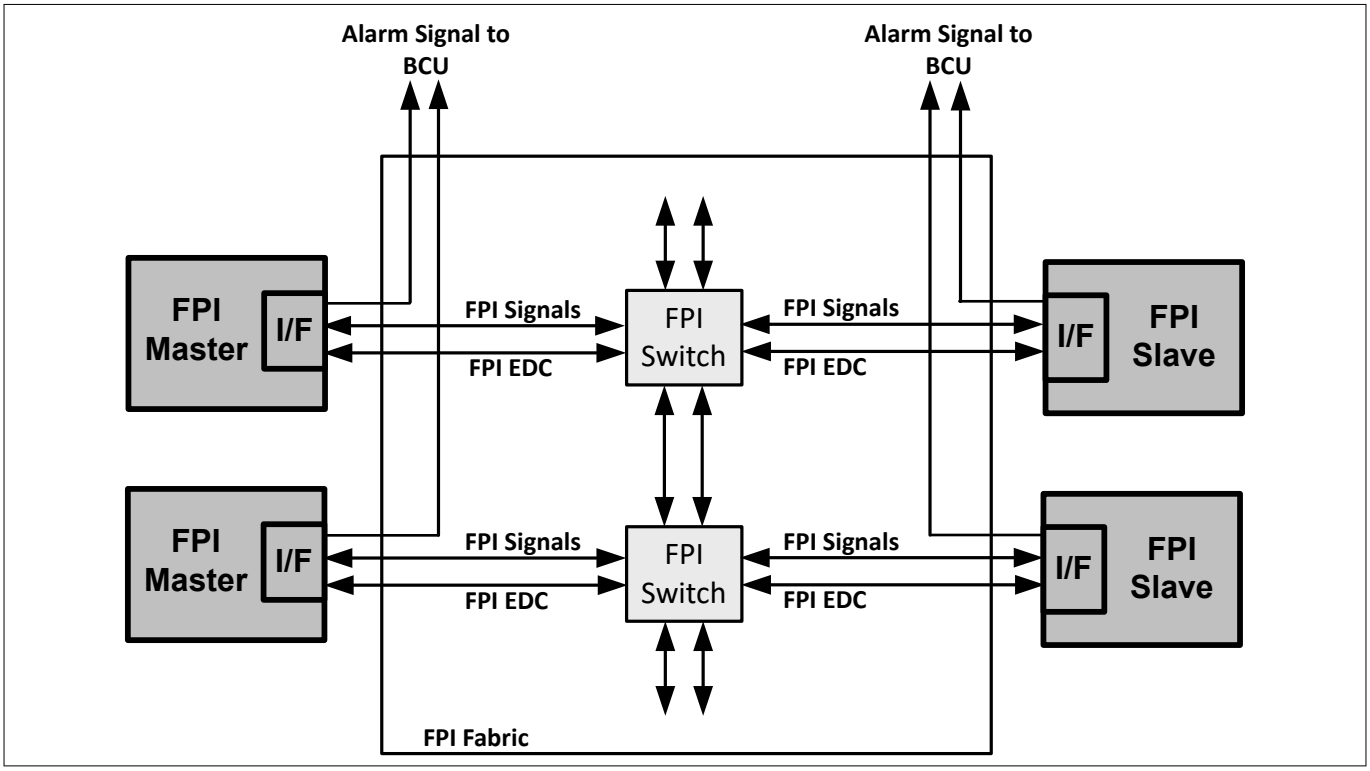


Figure 80 FPI EDC: Master and slave agent EDC alarm signals to the BCU

7.3.12.3 FPI error injection

The FPI EDC implementation allows to test all MEDM and SEDM decoder functions in the master and slave agents and the related alarm signal paths to the SMU by error injection.

At each FPI interconnect errors can be injected in the EDC encoder of one MEDM and one SEDM each. To test an EDC decoder in a slave agent and to trigger an alarm for it, a master agent access to this slave agent is initiated where an error is injected into the corresponding Master EDC Encoder (MEDM). Vice versa for the test of a master agent EDC decoder.

The error injection is controlled through the FPI Error Generation (FEGEN) in the BCU.

- Slave EDC encoder error injection (FEGEN.SEDM): The slave agent SEDM through which the error can be injected is the slave agent of the local BCU functional block. This can then be used to check all master agent error decoders (MEDM) at the local FPI. For example the SBCU with its FGEN register can be used to check the MEDM of all master agents connected to the SPB
- Master EDC encoder error injection (FEGEN.MEDM): The master agent MEDM through which the error can be injected is the one defined in the register bit-field PRIOH[3:0] (index 8) of the respective BCU. This can then be used to check all slave error decoders (SEDM) at the local FPI
- VMEN.EN allows to inject EDC errors into the output-enable signals for address-phase, data-phase and data informations signaled from the BCU slave agent and from the master agent assigned to PRIOH[3:0] to the FPI. Also here: the errors are affecting only the EDC information of these signals, not the output-signals directly therefore the normal communication is not affected but the related EDC error is detected and the corresponding ALSTAT bit in the BCU is set
- VMEN.BCU allows for injection errors in the EDC information of the FPI grant signal from the BCU to the master agent assigned to PRIOH[3:0] (master index 8) and in the EDC information of the FPI timeout (TOUT) signaled by the BCU

MEDM decoder test

To test a master agents MEDM decoder and the alarm signal path to the SMU the following sequence can be used:

- Enable the error injection related to the encoder where an error has to be injected in BCU SEDM through the FEGEN.SEDM register , for example to test a master agents read data EDC decoder
- The FPI master related to the MEDM where the decoder alarm signal path has to be tested, must access the BCU with an appropriate transaction. For example on SPB: DMA0 reads a SBCU control register in order to get an DMA related FPI EDC alarm from the SBCU
- Disable the error injection by clearing the error injection by setting FEGEN.SEDM to 000_H

SEDM decoder test

To test a slave agents SEDM decoder and the alarm signal path to the SMU the following sequence can be used:

- Enable the error injection related to the master agent that is mapped on the BCU register bit-field PRIOH[3:0] (index 8). This is the master agent where an FPI EDC error can to be injected through the register FEGEN.MEDM register. For example to test a slave agents FPI address phase EDC decoder
- The master agent that is mapped to PRIOH[3:0] has to access now the FPI slave agent related to the SEDM where the decoder alarm signal path has to be tested. For example on SPB: CPU0 data access to FCE to get an alarm from the FCE related alarm signal on the SPB
- Disable the error injection by setting FEGEN.SEDM to 000_H

7.3.12.4 Mapping of EDC ALARM signals to BCU ALSTAT and ALCLR registers

Each BCU provides a set of EDC Alarm Status registers (ALSTATx) and Alarm Clear Registers (ALCLRx):

- For each master agent and each slave on the local FPI there is one fix bit in the ALSTATx register that is set by hardware when an EDC error was detected for this master or slave agent
- For each master agent and each slave on the local FPI there is one fix bit in the ALCLRx register that can be used by software to clear the corresponding bit in the ALSTATx register
- The ALCTRL register provides additional status and clear functions to detect if there was more than one alarm while the corresponding ALSTATx register bit was already set (important if errors are injected during

run-time) or if multiple alarm events were signaled in the same cycle (then the ALSTATx bits may be corrupted)

The mapping of the FPI EDC alarm signals from the connected FPI master and slave agents and the alarm events from the BCU internal output enable checks to the Alarm Status (ALSTATx; x = 0-4) and Alarm Clear (ALCLRx; x = 0-4) registers is described in the device specific chapter.

The mapping of the module alarm signals to the ALSTATx[y] and ALCLRx[y] in the different devices is compatible. This means that the alarm of module x is always on a fixed register bit. If module x is not present in a derivative then the bit is reserved.

***Note**: The register ALSTATx, ALCLRx and ALCTRL do not influence the signaling of a detected EDC error through ALARM_NCS to SMU_SAFE.*

7.3.13 FPI to FPI interconnect register stage (F2F)

The FPI to FPI bridge implements a uni-directional bridge that forwards FPI transactions from one part of the one FPI interconnect instance to a second part of the same FPI interconnect instance.

The F2F bridge can be used as a register stage in an FPI interconnect to connect one or more distant slave agents to the respective FPI interconnect.

Each F2F bridge shows up with a dedicated alarm signal in the ALSTATx registers of the respective BCU. The respective product users manual describes in which FPI interconnects F2F bridges are used and how they are used there, which master and which slave agents are affected.

***Note**: If a transaction to a slave agent behind the F2F Bridge gets a FPI timeout, the F2F Bridge is blocked until application reset. All further accesses to the slave agents behind the bridge are terminated with FPI Error until the next application reset.*

7.3.14 Characteristic of the FPI interconnect

The following table shows the FPI interconnects as implemented in the different devices.

The table enables the customer to write generic software that can be easily ported between the targeted derivatives and to recognize the differences between the derivatives.

All BCU instances marked with an asterisk ´yes*´ are implemented with QoS control and QoS observation functions. BCUs not marked with asterisk are implemented without QoS functions because they do not need them for functional reasons or from the product point of view. Example: CSBCU does not need the QoS functions because the CSPB is essentially only used by the secure application. TBCU does not need the QoS functions since the TPB is essentially only used by the MCDS.

Table 606 FPI interconnects and interconnect instance names

	SPB	CPB	CSPB	TPB	COMPB	RCB
TC4Zx	yes*	no	yes	yes	yes*	no
TC48x	yes*	no	yes	yes	yes*	no
TC4Dx	yes*	no	yes	yes	yes*	no
TC49xN	yes*	yes*	yes	yes	no	no
TC46x	yes*	yes*	yes	yes	no	no
TC44x	yes*	yes*	yes	yes	no	no

(table continues...)

Table 606 (continued) FPI interconnects and interconnect instance names

	SPB	CPB	CSPB	TPB	COMPB	RCB
TC4Ex	yes*	yes*	yes	yes	no	no
TC4Px	yes*	yes*	yes	yes	no	no
TC42x	yes*	yes*	yes	yes	no	no
TC45x	yes	no	yes	yes	no	yes

Each interconnect instance provides a dedicated BCU:

- SPB → SBCU
- CPB → CBCU
- CSPB → CSBCU
- TPB → TBCU
- COMPB → COMBCU
- RCB → RBCU

7.4 Registers

7.4.1 Register overview - access mode glossary

Table 607 Register overview - access mode glossary

Keyword	Description
E	Access protection using PROT register PROTE.
SE	Access protection using PROT register PROTSE.
APU-P	Protection group consisting of registers ACCEN_WRA, ACCEN_WRB, ACCEN_RDA, ACCEN_RDB, ACCEN_VM, ACCEN_PRS.
P	Access protection using *APU-P* registers.
BE	Always returns a Bus Error.
U	No access restrictions.
PROT	Access restrictions as defined in the PROT register access rules.

7.4.2 Registers overview - BCU (ascending offset address)

Table 608 Registers overview - BCU (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
OCS	OCDS control and status register	004_H	P	SV, P	Debug Reset	1114
ID	Module identification register	008_H	P	BE	PowerOn Reset	1115
PROTE	PROT register endinit	$00C_H$	U	SV, PROT	Application Reset	1115
PROTSE	PROT register safe endinit (CS)	010_H	U	SV, PROT	Application Reset	1117
ACCEN_WRA	Write access enable register A	014_H	U	SE, SV	Application Reset	1118
ACCEN_WRB	Write access enable register B	018_H	U	SE, SV	Application Reset	1119
ACCEN_RDA	Read access enable register A	$01C_H$	U	SE, SV	Application Reset	1119
ACCEN_RDB	Read access enable register B	020_H	U	SE, SV	Application Reset	1120
ACCEN_VM	VM access enable register	024_H	U	SE, SV	Application Reset	1120
ACCEN_PRS	PRS access enable register	028_H	U	SE, SV	Application Reset	1121
RRCTRL	Round-robin group control	030_H	P	SV, E, P	Application Reset	1122

(table continues...)

Table 608 (continued) Registers overview - BCU (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PRIOH	Master agent priority register high	034_H	P	SV, E, P	Application Reset	1123
PRIOL	Master agent priority register low	038_H	P	SV, E, P	Application Reset	1123
CON	BCU control register	$03C_H$	P	SV, E, P	Application Reset	1124
ECON	BCU error control capture register	040_H	P	SV, P	Application Reset	1125
EADD	BCU error address capture register	044_H	P	SV, P	Application Reset	1127
EDAT	BCU error data capture register	048_H	P	SV, P	Application Reset	1127
CSAVSTAT	CSAV status register	$04C_H$	P	BE	Debug Reset	1128
ALSTATx	BCU EDC alarm status register x.	050_H+x*4	P	BE	Application Reset	1128
ALCLRx	BCU EDC alarm clear register x	070_H+x*4	P	SV, P	Application Reset	1129
ALCTRL	BCU EDC alarm control register	090_H	P	SV, P	Application Reset	1129
FEGEN	FPI error generation control register	094_H	P	SV, SE	Application Reset	1131
PSELECT	BCU performance event select	098_H	P	SV, E, P	Application Reset	1133
QOSICTRL	QoS time interval control register	100_H	P	SV, E, P	Application Reset	1133
QOSISTAT	QoS time interval counter status	104_H	P	BE	Application Reset	1134
QOSMx_CTRL	QoS master agent x control register	110_H+x*8	P	SV, E, P	Application Reset	1134
QOSMx_STAT	QoS master agent x status register	114_H+x*8	P	BE	Application Reset	1135
ARPx_TAGIDA	ARP x TAG-ID assignment register A	190_H+x*$1C_H$	P	P, SV, E	Application Reset	1136
ARPx_TAGIDB	ARP x TAG-ID assignment register B	194_H+x*$1C_H$	P	SV, P, E	Application Reset	1137
ARPx_VM	ARP x VM assignment register	198_H+x*$1C_H$	P	SV, P, E	Application Reset	1137
ARPx_THR	ARP x threshold register	$19C_H$+x*$1C_H$	P	SV, E, P	Application Reset	1138
ARPx_CTRL	ARP x control register	$1A0_H$+x*$1C_H$	P	SV, E, P	Application Reset	1138

(table continues...)

Table 608 (continued) Registers overview - BCU (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ARPx_CCC	ARP x cycle count capture register	$1A4_H$+x* $1C_H$	P	SV, E, P	Application Reset	1139
ARPx_CCV	ARP x current counter value register	$1A8_H$+x* $1C_H$	P	BE	Application Reset	1140

7.4.3 OCDS control and status register

The OCDS control and status register controls the OTGB trigger sets.

OCS Offset address: 004_H

OCDS control and status register Debug Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												TG_P	TGB	TGS	
r												w	rw	rw	

Field	Bits	Type	Description
TGS	1:0	rw	**Trigger set for OTGB0/1** 00_B No trigger set output 01_B Trigger set 1: ´TS16_BCU_Performance_Events´ selected 10_B Trigger set 2: Not used, no trigger set output 11_B Reserved
TGB	2	rw	**OTGB0/1 bus select** 0_B Trigger set is output on OTGB0 1_B Trigger set is output on OTGB1
TG_P	3	w	**TGS, TGB write protection** TGS and TGB are only written when TG_P is 1, otherwise unchanged. Read as 0. 0_B No action 1_B TGS and TGB are written.
0	31:4	r	**Reserved** Read as 0; should be written with 0.

Table 609 Access mode restrictions of OCS sorted by descending priority

Mode name	Access mode		Description
write 1 to .**TG_P**	rw	TGB, TGS	Set TG_P during write access
(default)	r	TGB, TGS	

7.4.4 Module identification register

ID Offset address: 008_H

Module identification register PowerOn Reset value: $006A\ C00F_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MOD_NUM															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MOD_TYPE								MOD_REV							
r								r							

Field	Bits	Type	Description
MOD_REV	7:0	r	**Module revision** Indicates the revision number of the implementation.
MOD_TYPE	15:8	r	**Module type** This internal marker is fixed to $C0_H$.
MOD_NUM	31:16	r	**Module number** Indicates the module identification number.

7.4.5 PROT register endinit

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner and in some PROT states, the secure master to update the PROT state as described in the PROT mechanism.

The PROTE register controls lock and unlock of the local Endinit (E) protected control registers.

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner and in some PROT states, the secure master to update the PROT state as described in the PROT mechanism

PROTE Offset address: $00C_H$

PROT register endinit Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OWEN	ODEF	TAGID						PRSEN	PRS			VMEN	VM		
w	rw	rw						rw	rw			rw	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												SWEN	STATE		
r												w	rwh		

Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
VM	18:16	rw	**Virtual machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual machine definition enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.
PRSEN	23	rw	**Protection set definition enable for PROT owner** 0_B **Disable**: PRS is not part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG-ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; should be written with 0.

Table 610 Access mode restrictions of PROTE sorted by descending priority

Mode name	Access mode		Description
write 1 to .**SWEN**	rwh	STATE	
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

7.4.6 PROT register safe endinit (CS)

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner and in some PROT states, the secure master to update the PROT state as described in the PROT mechanism.

The PROTSE register controls lock and unlock of the local Safe Endinit (SE) protected control registers.

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner and in some PROT states, the secure master to update the PROT state as described in the PROT mechanism

PROTSE Offset address: 010_H

PROT register safe endinit [CS] Application Reset value: $4C00\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OWEN	**ODEF**	**TAGID**						**PRSEN**	**PRS**			**VMEN**	**VM**		
w	rw	rw						rw	rw			rw	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												**SWEN**	**STATE**		
r												w	rwh		

Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write

(table continues...)

(continued)

Field	Bits	Type	Description
VM	18:16	rw	**Virtual machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual machine definition enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.
PRSEN	23	rw	**Protection set definition enable for PROT owner** 0_B **Disable**: PRS is not part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG-ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; should be written with 0.

Table 611 Access mode restrictions of PROTSE sorted by descending priority

Mode name	Access mode		Description
write 1 to .**SWEN**	rwh	STATE	
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

7.4.7 Write access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

ACCEN_WRA Offset address: 014_H

Write access enable register A Application Reset value: $1000\ 0003_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q 0_B Disabled for write access 1_B Enabled for write access

7.4.8 Write access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

ACCEN_WRB Offset address: 018_H

Write access enable register B Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q 0_B Disabled for write access 1_B Enabled for write access
0	31:8	r	**Reserved** Read as 0; should be written with 0.

7.4.9 Read access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

ACCEN_RDA Offset address: $01C_H$

Read access enable register A Application Reset value: $FFFF\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q 0_B Disabled for read access 1_B Enabled for read access

7.4.10 Read access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

ACCEN_RDB Offset address: 020_H

Read access enable register B Application Reset value: $0000\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q 0_B Disabled for read access 1_B Enabled for read access
0	31:8	r	**Reserved** Read as 0; should be written with 0.

7.4.11 VM access enable register

This register defines which virtual machine encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each VMx.

ACCEN_VM Offset address: 024_H

VM access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for VM ID q** This bit enables read access to the access protected region for transactions with the VM ID q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for VM ID q** This bit enables write access to the access protected region for transactions with the VM ID q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

7.4.12 PRS access enable register

This register defines which protection register set encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each PRSx.

ACCEN_PRS Offset address: 028_H

PRS access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for PRS q** This bit enables read access to the access protected region for transactions in the PRS q. 0_B Disabled for read access 1_B Enabled for read access

(table continues...)

(continued)

Field	Bits	Type	Description
WRq (q=00-07)	q+16	rw	**Write access enable for PRS q** This bit enables write access to the access protected region for transactions in the PRS q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

7.4.13 Round-robin group control

The Round Share mechanism of a round-robin group only affects arbitration when requests of master agents assigned to the related round-robin group or higher priority master agents arrives so frequently that pending master agent request with lower priority are not granted.

When this occurs a maximum of RRx grants are given to the master agents of the related round-robin group and master agents with a higher priority, and then a single grant is given to the lower priority master agents if there is any request pending.

RRCTRL Offset address: 030_H

Round-robin group control Application Reset value: $0006\ 0006_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0												RR4			0
r												rw			r

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												RR8			0
r												rw			r

Field	Bits	Type	Description
RR8	3:1	rw	**RR Group 8 Priority Round Share** The Register defines the number of consecutive transactions granted to master agents of this round-robin group and master agents with higher priority before a request for a master agent with lower priority is granted. The register defines the number in multiples of 2. The mechanism is inactive if the register is configured with 0. A detailed description of the arbitration mechanisms can be found in the System Interconnect FPI chapter.
RR4	19:17	rw	**RR Group 4 Priority Round Share** The register defines the number of consecutive transactions granted to master agents of this round-robin group and master agents with higher priority before a request for a master agent with lower priority is granted. The register defines the number in multiples of 2. The mechanism is inactive if the register is configured with 0. A detailed description of the arbitration mechanisms can be found in the System Interconnect FPI chapter.

(table continues...)

(continued)

Field	Bits	Type	Description
0	0, 16:4, 31:20	r	**Reserved** Read as 0; should be written with 0.

7.4.14 Master agent priority register high

Priority 4 and priority 8 are implemented as round-robin groups.
Master agents can be mapped to a round-robin group by configuration with priority 4 or priority 8.
A detailed description of the arbitration mechanisms can be found in the System Interconnect FPI chapter.

PRIOH Offset address: 034_H

Master agent priority register high Application Reset value: $0888\ 8888_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MASTER15				MASTER14				MASTER13				MASTER12			
rw				rw				rw				rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MASTER11				MASTER10				MASTER9				MASTER8			
rw				rw				rw				rw			

Field	Bits	Type	Description
MASTERi (i=8-15)	4*i-29:4*i-32	rw	**Master i Priority** Master agent i priority for access on the FPI. A lower number has a higher priority in the arbitration round than a higher one.

7.4.15 Master agent priority register low

Priority 4 and priority 8 are implemented as round-robin groups.
Master agents can be mapped to a round-robin group by configuration with priority 4 or priority 8.
A detailed description of the arbitration mechanisms can be found in the System Interconnect FPI chapter.

PRIOL Offset address: 038_H

Master agent priority register low Application Reset value: $0444\ 4444_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MASTER7				MASTER6				MASTER5				MASTER4			
rw				rw				rw				rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MASTER3				MASTER2				MASTER1				MASTER0			
rw				rw				rw				rw			

Field	Bits	Type	Description
MASTERi (i=0-7)	4*i+3:4*i	rw	**Master i Priority** Master agent i priority for access on the FPI. A lower number has a higher priority in the arbitration round than a higher one.

7.4.16 BCU control register

The BCU control register controls the overall operation of the BCU, including setting the starvation sample period, the timeout period, enabling starvation-protection mode, and error handling.

CON Offset address: $03C_H$

BCU control register Application Reset value: $FFF1\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SPC												0			DBG
rw												r			rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TOUT															
rw															

Field	Bits	Type	Description
TOUT	15:0	rw	**BCU timeout value** The bit field determines the number of wait state cycles before the BCU terminates a data phase and the related transaction with a timeout. Default value after reset is $FFFF_H$ (= 65536 local interconnect cycles). ***Note**: TOUT value must be >= 5 and shall be configured with a value that it is not violated during normal operation.*
DBG	16	rw	**BCU Debug Trace Enable** The bit enables and disables the error capture mechanism for the registers BCU_ECON, BCU_EADD, BCU_EDAT. The bit has no affect on the signalling of alarms and interrupts when an error condition is detected. 0_B BCU debug trace disabled 1_B BCU debug trace enabled (default after reset)
SPC	31:20	rw	**Starvation period control** Determines the sample period for the starvation counter. Must be larger than the number of masters. The reset value is FFF_H.
0	19:17	r	**Reserved** Read as 0; should be written with 0.

7.4.17 BCU error control capture register

Application software can clear the register by writing the content.

ECON Offset address: 040_H

BCU error control capture register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OPC				TAG						RDN	WRN	SVM	ACK		ABT
rwh				rwh						rwh	rwh	rwh	rwh		rwh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RDY	TOUT	ERRCNT					CS_VALID	PRS_VALID	PRS			VM_VALID	VM		
rwh	rwh	rwh					rwh	rwh	rwh			rwh	rwh		

Field	Bits	Type	Description
VM	2:0	rwh	**FPI virtual machine information** This bit indicates the state of the Virtual machine information at an FBI error
VM_VALID	3	rwh	**FPI virtual machine information valid signal** This bit indicates the state of the Virtual Machine valid signal at an FBI error. 0_B Virtual machine information is not valid 1_B Virtual machine information is valid
PRS	6:4	rwh	**FPI Protection Set Information** This bit field indicates the protection set information of the FPI transaction at an FPI error.
PRS_VALID	7	rwh	**FPI protection set information valid signal** This bit indicates the state of the protection set valid signal at an FBI error. 0_B Protection set information is not valid 1_B Protection set information is valid
CS_VALID	8	rwh	**FPI CS valid signal** This bit indicates the state of the CS valid signal at an FBI error. 0_B Wait state(s) have been inserted. Ready signal was active 1_B Ready signal was inactive
ERRCNT	13:9	rwh	**FPI error counter** ERRCNT is incremented with each Error or Timeout event. If it is not reset by reading, it counts the events up to the max. counter value and then stops at this value. ERRCNT is reset to 0 with a read of the ECON register. A write access is not executed in the register if an error is detected in parallel. 1)

(table continues...)

(continued)

Field	Bits	Type	Description
TOUT	14	rwh	**State of FPI timeout signal** This bit indicates the state of the timeout signal at an FBI error. 0_B No timeout occurred 1_B Timeout has occurred
RDY	15	rwh	**State of FPI ready signal** This bit indicates the state of the ready signal at an FBI error. 0_B Wait state(s) have been inserted. Ready signal was active 1_B Ready signal was inactive
ABT	16	rwh	**State of FPI abort signal** This bit indicates the state of the abort signal at an FBI error. 0_B Master has aborted an FPI transfer. Abort signal was active 1_B Abort signal was inactive
ACK	18:17	rwh	**State of FPI acknowledge signals** This bit field indicates the acknowledge code that has been output by the selected slave at an FPI error. 00_B NSC: No special case 01_B Reserved 10_B RTY: Retry 11_B ERR: FPI error
SVM	19	rwh	**State of FPI supervisor mode signal** This bit indicates whether the FPI error occurred in supervisor mode or in user mode. 0_B Transfer was initiated in user mode 1_B Transfer was initiated in supervisor mode
WRN	20	rwh	**State of FPI write signal** This bit indicates whether the FPI error occurred at a write transaction.
RDN	21	rwh	**State of FPI read signal** This bit indicates whether the FPI error occurred at a read transaction.
TAG	27:22	rwh	**FPI master agent function TAG-ID** This bit field indicates the FPI master agent function identifier (TAG-ID).
OPC	31:28	rwh	**FPI operation code signals** The FPI operation codes are defined in .

1) Aborted accesses to a 0 wait state SPB slave may also increment ERRCNT when the slave generates an error acknowledge.

Table 612 FPI read and write error indication

$\overline{RD}$	$\overline{WR}$	FPI Cycle
0	0	FPI error occurred at the read transfer of a read-modify-write transfer.
0	1	FPI error occurred at a read cycle of a single transfer.
1	0	FPI error occurred at a write cycle of a single transfer or at the write cycle of a read-modify-write transfer.
1	1	Does not occur.

7.4.18 BCU error address capture register

Application software can clear the register by writing the content.

EADD Offset address: 044_H

BCU error address capture register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FPIADR															
rwh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FPIADR															
rwh															

Field	Bits	Type	Description
FPIADR	31:0	rwh	**Captured FPI address** This bit field holds the 32-bit FPI address of the FPI address phase that corresponds to the FPI data phase that was acknowledged by the slave agent with FPI error.. Note that if multiple FPI errors occurred, only the address of the first FPI error is captured.

7.4.19 BCU error data capture register

Application software can clear the register by writing the content.

EDAT Offset address: 048_H

BCU error data capture register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FPIDAT															
rwh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FPIDAT															
rwh															

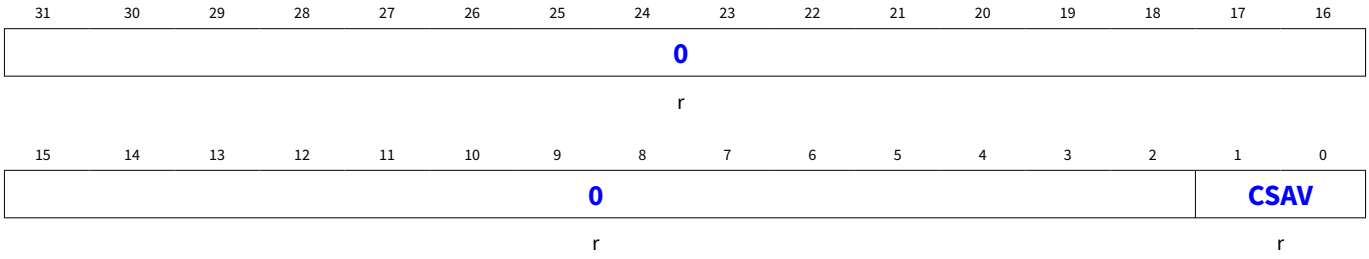
Field	Bits	Type	Description
FPIDAT	31:0	rwh	**Captured FPI Data** This bit field holds the 32-bit FPI data that has been captured in the data phase that was acknowledged with FPI error by the active slave agent. Note that if multiple FPI errors occurred, only the data of the first FPI error is captured.

7.4.20 CSAV status register

Register shows the status of cyber security activity visibility (CSAV) signals.
CS related transactions are identified in the transaction address phase through CS_VALID=1.

CSAVSTAT Offset address: $04C_H$

CSAV status register Debug Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
CSAV	1:0	r	**Status of the CSAV signals** This bitfield shows the current CSAV status. CSAV is controlled through the CSCU. The CSAV status controls if and how information of CS related transactions (CSVALID=1) can be traced through OCMTS and captured by the BCU. 00_B No - Tracing of CS transactions and capturing of CS transactions in the BCU is disabled 01_B Limited - Tracing of CS transactions through OCMTS and capturing of CS transactions in the BCU is enabled with limited information. Address, data, VM and PRS information are masked and read with 0. 10_B Reserved - Tracing of CS transactions and capturing of CS transaction information in the BCU is disabled 11_B Full - Tracing of CS transactions through OCMTS and capturing of CS transaction information in the BCU is enabled
0	31:2	r	**Reserved** Read as 0; should be written with 0.

7.4.21 BCU EDC alarm status register x.

The BCU provides one alarm status register bit for each implemented FPI master agent and FPI slave agent. The assignment of the FPI master-slave and master-agent alarm signals is contained in the bit descriptions of the product-specific ALSTAT register description. The source of an alarm signal is described there using <module name>_S for a slave agent interface, <module name>_M for a master agent interface.

ALSTATx (x=0-4) Offset address: 050_H+x*4

BCU EDC alarm status register x. Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AL31	AL30	AL29	AL28	AL27	AL26	AL25	AL24	AL23	AL22	AL21	AL20	AL19	AL18	AL17	AL16
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AL15	AL14	AL13	AL12	AL11	AL10	AL09	AL08	AL07	AL06	AL05	AL04	AL03	AL02	AL01	AL00
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
ALy (y=00-31)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 0_B No error was detected 1_B An error was detected

7.4.22 BCU EDC alarm clear register x

The BCU provides one alarm clear register bit for each implemented FPI master and FPI slave.

ALCLRx (x=0-4) Offset address: 070_H+x*4

BCU EDC alarm clear register x Application Reset value: $0000\ 0000_H$

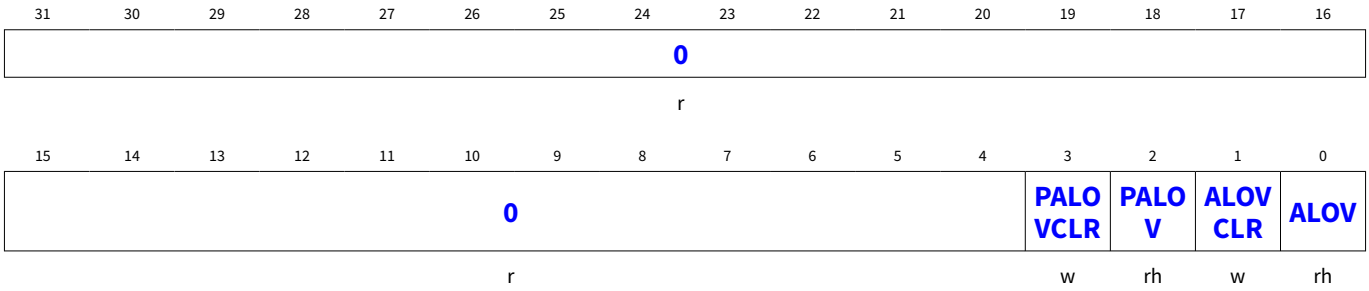
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CLR31	CLR30	CLR29	CLR28	CLR27	CLR26	CLR25	CLR24	CLR23	CLR22	CLR21	CLR20	CLR19	CLR18	CLR17	CLR16
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CLR15	CLR14	CLR13	CLR12	CLR11	CLR10	CLR09	CLR08	CLR07	CLR06	CLR05	CLR04	CLR03	CLR02	CLR01	CLR00
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

Field	Bits	Type	Description
CLRy (y=00-31)	y	w	**Clear alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI slave and master interface. 0_B No action 1_B Clears related SBC_ALSTATx.[y] (clear = set to ´0´); read always returns 0

7.4.23 BCU EDC alarm control register

ALCTRL Offset address: 090_H

BCU EDC alarm control register Application Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
ALOV	0	rh	**Alarm overflow** The ALOV bit is set if multiple FPI EDC alarms for the same FPI slave or the same FPI master were detected while the related ALSTATx[y] bit was still set. ***Note**: Some errors result in a static fault situation, for example address phase, data phase or data enable signal faults. Static faults do not generate multiple alarms and will not set the ALOV bit.* 0_B No alarm overflow for any FPI EDC alarm detected 1_B Alarm overflow detected for at least one of the set ALSTATx[y] bits
ALOVCLR	1	w	**Alarm overflow clear** The ALOVCLR bit is required to clear the ALOV bit. 0_B No action 1_B Clear ALOV (clear = set to 0); bit value is not stored; read always returns 0
PALOV	2	rh	**Parallel alarm overflow** The PALOV bit is set if multiple FPI EDC alarms from different slave agents or master agents are signaled in the same clock cycle. Corresponding ALSTAT Bits are not set. ***Note**: Some errors result in a static fault situation, for example address phase, data phase or data enable signal faults. Static faults do not generate multiple alarms and will not set the ALOV bit.* 0_B No alarm overflow for any FPI EDC alarm detected 1_B Alarm overflow detected for at least one of the set ALSTATx[y] bits
PALOVCLR	3	w	**Parallel alarm overflow clear** The PALOVCLR bit is required to clear the PALOV bit. 0_B No action 1_B Clear ALOV (clear = set to 0); bit value is not stored; read always returns 0
0	31:4	r	**Reserved** Read as 0; should be written with 0.

7.4.24 FPI error generation control register

The FEGEN register controls the injection of errors in the FPI EDC mechanism to test the alarm path from the implemented EDC decoders to the SMU.

Errors can be injected in the FPI EDC related encoders and the mechanism that checks the module output enable signal:

- BCU FPI slave agent EDC encoder (SEDM)
- FPI master agent EDC encoder (MEDM) for the master agent mapped on PRIOH[3:0] (master index 8)
- BCU FPI timeout signal
- BCU FPI grant signals
- All groups of FPI slave and master output enable signals that are checked in the BCU

FEGEN Offset address: 094_H

FPI error generation control register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0						BCU		0					EN		
r						rw		r					rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0					MEDM			0					SEDM		
r					rw			r					rw		

Field	Bits	Type	Description
SEDM	2:0	rw	**SEDM (slave encoder)** The errors are injected into the FPI-EDC encoder information of the SEDM which is assigned to the BCU. The injected errors are detected by the master agents MEDM, which access the BCU registers. Other bit combinations are reserved and do not inject errors. 000_B SEDM error injection is disabled 011_B Slave agent data phase (disable inversion of the FPI_ACK[0] signal) 101_B Slave read data (ECC LSB flip) **others,** Reserved, no SEDM errors are injected

(table continues...)

(continued)

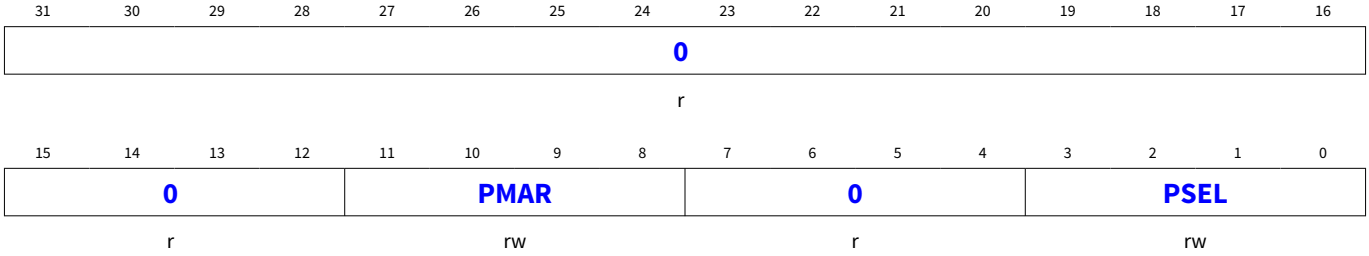
Field	Bits	Type	Description
MEDM	10:8	rw	**MEDM (master encoder) type of error** The errors are injected into the MEDM's FPI-EDC coding associated with the PRIOH[3:0] (master index 8) assigned master agent. The injected errors are detected by the SEDM of the slave agent accessed by the master agent. Other bit combinations are reserved and do not inject errors. 000_B MEDM error injection is disabled 001_B Master agent REQUEST signal (disable REQUEST signal inversion) 010_B Master agent LOCK signal (disable LOCK signal inversion) 011_B Master agent address phase (invert ECC LSB) 101_B Master agent data phase (disable inversion of the FPI_ACK[0] signal, default master agent) 110_B Master agent write data (invert ECC LSB) 111_B Master agent abort signal (disable abort signal inversion)
EN	18:16	rw	**Enable signal type of error** The errors of the enable signals are injected into the BCU without affecting the transactions executed on the FPI. For each group of enable signals, it is checked that no more than one signal is active at any time. For the error injection, the BCU activates the respective enable signals, going into the error checking mechanisms, related to the BCU and the master agent assigned to PRIOH[3:0] (master index 8). Other bit combinations are reserved and do not inject errors. 000_B EN error injection is disabled 011_B Address phase enable (A_EN_N, master agent) 101_B Data phase enable (ABORT_EN_N, master agent) 110_B Data enable (D_EN_N, master and slave agent) 111_B Data phase output enable (ACK_EN_N, master and slave agent) **others,** Reserved, no EN errors are injected
BCU	25:24	rw	**BCU type of error** The errors are injected into the FPI-EDC information. Other bit combinations are reserved and do not inject errors. 00_B BCU timout and grant error injection is disabled 10_B BCU grant: disable inversion of grant signal to the master agent mapped to PRIOH[3:0] (master index 8). Error is detected by the MEDM related to this master agent. 11_B BCU timeout: disable BCU timeout signal inversion inside the BCU in front of the error checking mechanism. The injected error is detected by the BCU. **others,** BCU timout and grant error injection is disabled
0	7:3, 15:11, 23:19, 31:26	r	**Reserved** Read as 0; should be written with 0.

7.4.25 BCU performance event select

This function can be used when only one MCDS counter is available to control which event is routed to the MCDS counter through the BCU.

PSELECT Offset address: 098_H

BCU performance event select Application Reset value: $0000\ 000F_H$



Field	Bits	Type	Description
PSEL	3:0	rw	**TS16_BCU_performance_events[15] selector** Register controls which event from the TS16_BCU_performance_events list is mapped to PSELECT that it can be observed by the MCDS. PSELECT= x : TS16_BCU[x] mapped to PSELECT.
PMAR	11:8	rw	**Parallel master agent requests** The register controls the threshold value from which a pulse is signalled through TS16_BCU_performance_events[8] (Interconnect_contention_cycles). PMAR= 0 : no pulse is generated PMAR = x: pulse is generated on every local FPI cycle where >= x master agent requests for FPI access are active in parallel to the actual transaction.
0	7:4, 31:12	r	**Reserved** Read as 0; should be written with 0.

7.4.26 QoS time interval control register

The QOSICTRL register allows the enabling and disabling of the QoS function and the defining of the QoS time interval for this interconnect.

QOSICTRL Offset address: 100_H

QoS time interval control register Application Reset value: $0000\ FFFF_H$

Field	Bits	Type	Description
QOSINT	15:0	rw	**QoS time interval** The bit field determines the QoS time interval in multiples of 16 local interconnect clock cycles.
QOSEN	31	rw	**QoS enable** 0_B QoS function is disabled for the local interconnect 1_B QoS function is enabled for the local interconnect
0	30:16	r	**Reserved** Read as 0; should be written with 0.

7.4.27 QoS time interval counter status

Indicates the status of the QoS time interval counter.

QOSISTAT Offset address: 104_H

QoS time interval counter status Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0												QOSICNT			
r												rh			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
QOSICNT															
rh															

Field	Bits	Type	Description
QOSICNT	19:0	rh	**QoS time interval counter status** The bit field shows the current values of the QoS time interval counter. In QoS mode, the decrementing counter indicates the number of local clock cycles before the current time interval has expired. when the counter is counted down to 0, it is reloaded with QOSINT.
0	31:20	r	**Reserved** Read as 0; should be written with 0.

7.4.28 QoS master agent x control register

The BCU provides one QOSM CTRL register for each master agent connected to the local interconnect.

QOSMx_CTRL (x=0-15) Offset address: 110_H+x*8

QoS master agent x control register Application Reset value: $07FF\ 7FFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MEN	MSB CLR	0			MRD										
rw	w	r			rw										

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	MICC														
r	rw														

Field	Bits	Type	Description
MICC	14:0	rw	**Master agent interconnnect cycle count** Bit field defines how many clock cycles the master agent can use the interconnect in the QoS time interval before it is set to request delay mode. The bit field defines the number in multiples of 16 local FPI clock cycles.
MRD	26:16	rw	**Master agent request delay** Bit field determines in request delay mode the number of FPI clock cycles a master request is delayed before it is taken into account in arbitration.
MSBCLR	30	w	**QoS master agent sticky bit clear** Bit can be used to clear the master agent sticky bit (MSB). Bit is always read as 0. 0_B No action 1_B Bit MSB bit is cleared.
MEN	31	rw	**Master agent enable** 0_B QoS function is disabled for this master agent 1_B QoS function is enabled for this master agent
0	15, 29:27	r	**Reserved** Read as 0; should be written with 0.

7.4.29 QoS master agent x status register

The BCU provides one QOSM STAT register for each master agent connected to the local interconnect.

QOSMx_STAT (x=0-15) Offset address: 114_H+x*8

QoS master agent x status register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MSTAT	MSB	0											MCNT		
rh	rh	r											rh		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MCNT															
rh															

Field	Bits	Type	Description
MCNT	18:0	rh	**Master agent cycle counter** In QoS mode the current usage of this counter is indicated by the MSTAT bit. Depending on the mode, the counter is loaded with MICC or MRD and then decremented according to the mode. MSTAT=0 This counter shows the remaining clock cycles for the master agent that it can still use in the current QoS time interval before it switches to request delay mode. MSTAT=1 the master agent is in request delay mode until the end of the QoS time interval. With each new request, the counter is loaded with the MRD value and counted down. If the counter value is 0, the master agent request is considered in the arbitration.
MSB	30	rh	**QoS master agent sticky bit** Determines whether the master agent was in request delay mode in one or more of the previous QoS time intervals. Note: The bit is set from 0 to 1 at the end of a QoS time interval if MSTAT=1. In parallel a QoS interrupt is triggered whenever any MSB bit changes from 0 to 1. 0_B Master agent was not in request delay mode 1_B Master agent was in request delay mode
MSTAT	31	rh	**Master agent status** Determines whether the master agent is in request delay mode in the current QoS time interval. Note: the MSTAT bit changes from 0 to 1 when MCNT is counted down from 1 to 0. 0_B Master agent is not in request delay mode 1_B Master agent is in request delay mode
0	29:19	r	**Reserved** Read as 0; should be written with 0.

7.4.30 ARP x TAG-ID assignment register A

ARPx_TAGIDA (x=0-7) Offset address: 190_H+x*1C_H

ARP x TAG-ID assignment register A Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Bit controls the assignment of TAG-ID q to ARP x** 0_B TAG-ID not assigned to ARP x 1_B TAG-ID selected for ARP x

7.4.31 ARP x TAG-ID assignment register B

ARPx_TAGIDB (x=0-7) Offset address: 194_H+x*1C_H

ARP x TAG-ID assignment register B Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN63	EN62	EN61	EN60	EN59	EN58	EN57	EN56	EN55	EN54	EN53	EN52	EN51	EN50	EN49	EN48
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN47	EN46	EN45	EN44	EN43	EN42	EN41	EN40	EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-63)	q-32	rw	**Bit controls the assignment of TAG-ID q to ARP x** 0_B TAG-ID not assigned to ARP x 1_B TAG-ID assigned to ARP x

7.4.32 ARP x VM assignment register

ARPx_VM (x=0-7) Offset address: 198_H+x*1C_H

ARP x VM assignment register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-07)	q	rw	**Bit controls the assignment of VM-ID q to ARP x** 0_B VM-ID not assigned to ARP x 1_B VM-ID assigned to ARP x
0	31:8	r	**Reserved** Read as 0; should be written with 0.

7.4.33 ARP x threshold register

Register defines the interconnect bandwidth usage for the master agent functions mapped to ARPx.

ARPx_THR (x=0-7) Offset address: $19C_H+x*1C_H$

ARP x threshold register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
THV															
rw															

Field	Bits	Type	Description
THV	15:0	rw	**ARP x threshold value** Bit field defines how many clock cycles the master agent functions assigned to ARPx can use the observed resource in the QoS time interval before it is identified as an ARPx usage violation. The bit field defines the number in multiples of 16 local interconnect clock cycles.
0	31:16	r	**Reserved-0** Read as 0; should be written with 0.

7.4.34 ARP x control register

Register provides control for the ARPx function and indicates the status of ARPx.

ARPx_CTRL (x=0-7) Offset address: $1A0_H+x*1C_H$

ARP x control register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
STAT	CLR	0													
rh	w	r													

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															EN
r															rw

Field	Bits	Type	Description
EN	0	rw	**ARPx enable** 0_B ARPx function is disabled 1_B ARPx function is enabled

(table continues...)

(continued)

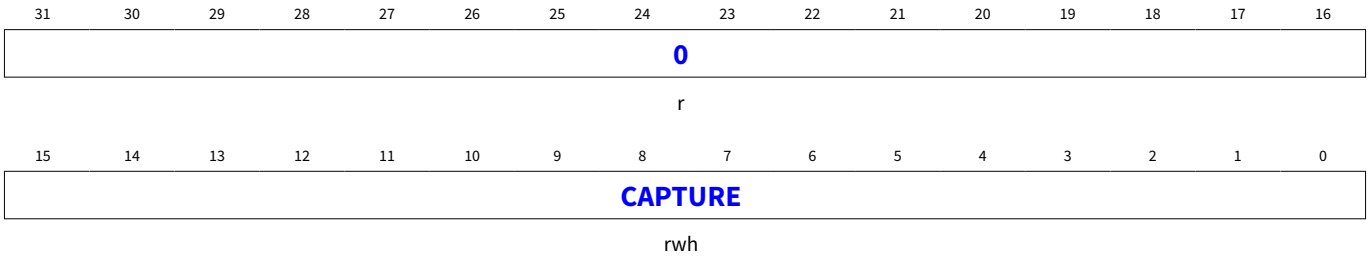
Field	Bits	Type	Description
CLR	30	w	**ARPx status clear** Bit can be used to clear the ARPx status bit (STAT). Bit is always read as 0. 0_B No action 1_B STAT bit is cleared
STAT	31	rh	**ARPx status** The STAT bit is set by the ARPx function at the end of a QoS interval if an ARPx usage violation was identified in the QoS interval. If the STAT bit is not already set at that time, the CAPTURE register is simultaneously updated and its contents are protected against being overwritten by the hardware until the STAT bit is reset by the application software.. 0_B This partition has not had a usage violation 1_B This partition has had a usage violation
0	29:1	r	**Reserved-0** Read as 0; should be written with 0.

7.4.35 ARP x cycle count capture register

Register shows the bandwidth used by the master agent functions mapped to ARPx in a QoS interval.

ARPx_CCC (x=0-7) Offset address: $1A4_H+x*1C_H$

ARP x cycle count capture register Application Reset value: $0000\ 0000_H$



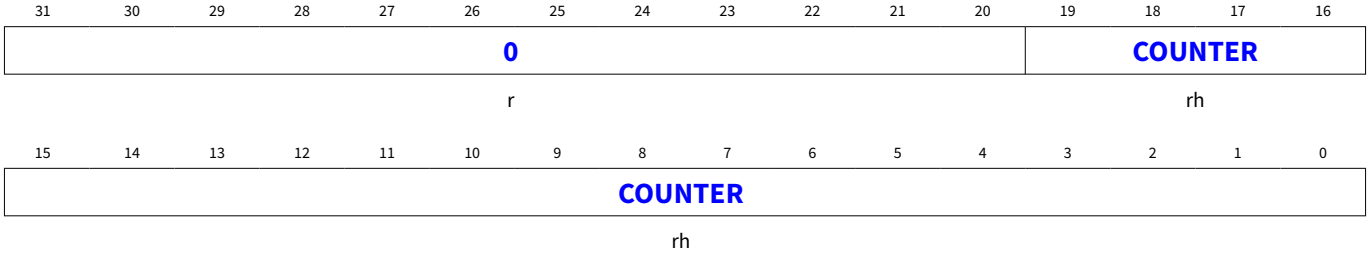
Field	Bits	Type	Description
CAPTURE	15:0	rwh	**ARP x captured cycle count** Bit field indicates how many clock cycles the master agent functions assigned to the ARPx have used the observed resource in a QoS time interval. The register is updated at the end of a QoS time interval if the CTRL.STAT bit is not set. The bit field provides the number in multiples of 16 local interconnect clock cycles.
0	31:16	r	**Reserved-0** Read as 0; should be written with 0.

7.4.36 ARP x current counter value register

Register shows the state of the ARPx counter with which the clocks are counted in which ARPx master agent functions have occupied the interconnect in the current QoS interval.

ARPx_CCV (x=0-7) Offset address: $1A8_H+x*1C_H$

ARP x current counter value register Application Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
COUNTER	19:0	rh	**ARP x cycle count** Bit field shows the current state of the ARPx clock cycle counter and thus how many clock cycles the master agent functions assigned to the ARPx have used the observed resouce in the current QoS time interval so far. Note: The ARPx clock cycle counter counts only when ARPx is enabled.
0	31:20	r	**Reserved-0** Read as 0; should be written with 0.

7.5 Debug information

The BCU contains an interface to the OCMTS through which the activities on the local FPI can be monitored. The observation must be released through the global CSAV signals to activate the interface. In addition, the BCU has an error capture register that captures information about the transaction that ended with a FPI error or FPI timeout. Cyber Security related transactions, which are transactions with CS_VALID=1, are only captured if previously enabled by CSRM. A detailed description of these functions can be found in the respective BCU chapters.

Related information

FPI error handling on page 1093
FPI error capture mechanism and BCU interrupts on page 1095
BCU interrupts on page 1096
Interconnect performance measurement and analysis on page 1097

7.6 References

This section is not applicable for the module being described.

7.7 FPI revision history

Reference	Description of change(s)
Date range: 2024-08-17 to 2025-02-10	
CON	• Typo fixed in bit field TOUT
ALSTATx	• Concretization in register description
PSELECT	• Concretization in register description

7.8 TC4Dx FPI information

7.8.1 TC4Dx FPI configuration

There are no deviations from the generic specification.

7.8.2 TC4Dx FPI features

There are no deviations from the generic specification.

7.8.3 TC4Dx FPI functional description

There are no deviations from the generic specification.

7.8.4 TC4Dx FPI registers

7.8.4.1 Register address space - BCU

Table 613 Registers address space - BCU

Module	Base address	End address	Note
COMBCU	$F5FFFC00_H$	$F5FFFFFF_H$	
CSBCU	$F6FFFC00_H$	$F6FFFFFF_H$	
SBCU	$F1FFFC00_H$	$F1FFFFFF_H$	
TBCU	$FAFFFC00_H$	$FAFFFFFF_H$	

7.8.4.2 Register overview - access mode glossary

Table 614 Register overview - access mode glossary

Keyword	Description
E	Access protection using PROT register SBCU_PROTE or COMBCU_PROTE or TBCU_PROTE or CSBCU_PROTE.
SE	Access protection using PROT register SBCU_PROTSE or COMBCU_PROTSE or TBCU_PROTSE or CSBCU_PROTSE.
APU-P	Protection group consisting of registers SBCU_ACCEN_WRA, SBCU_ACCEN_WRB, SBCU_ACCEN_RDA, SBCU_ACCEN_RDB, SBCU_ACCEN_VM, SBCU_ACCEN_PRS or COMBCU_ACCEN_WRA, COMBCU_ACCEN_WRB, COMBCU_ACCEN_RDA, COMBCU_ACCEN_RDB, COMBCU_ACCEN_VM, COMBCU_ACCEN_PRS or TBCU_ACCEN_WRA, TBCU_ACCEN_WRB, TBCU_ACCEN_RDA, TBCU_ACCEN_RDB, TBCU_ACCEN_VM, TBCU_ACCEN_PRS or CSBCU_ACCEN_WRA, CSBCU_ACCEN_WRB, CSBCU_ACCEN_RDA, CSBCU_ACCEN_RDB, CSBCU_ACCEN_VM, CSBCU_ACCEN_PRS.
P	Access protection using *APU-P* registers.
BE	Always returns a Bus Error.
U	No access restrictions.
PROT	Access restrictions as defined in the PROT register access rules.

7.8.4.3 Register overview - COMBCU (ascending offset address)

Table 615 Register overview - COMBCU (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
COMBCU_OCS	OCDS control and status register	004_H	P	SV, P	Debug Reset	1114
COMBCU_ID	Module identification register	008_H	P	BE	PowerOn Reset	1115
COMBCU_PROTE	PROT register endinit	$00C_H$	U	SV, PROT	Application Reset	1150

(table continues...)

Table 615 (continued) Register overview - COMBCU (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
COMBCU_PROTSE	PROT register safe endinit	010_H	U	SV, PROT	Application Reset	1154
COMBCU_ACCEN_WRA	Write access enable register A	014_H	U	SE, SV	Application Reset	1157
COMBCU_ACCEN_WRB	Write access enable register B	018_H	U	SE, SV	Application Reset	1158
COMBCU_ACCEN_RDA	Read access enable register A	$01C_H$	U	SE, SV	Application Reset	1159
COMBCU_ACCEN_RDB	Read access enable register B	020_H	U	SE, SV	Application Reset	1159
COMBCU_ACCEN_VM	VM access enable register	024_H	U	SE, SV	Application Reset	1160
COMBCU_ACCEN_PRS	PRS access enable register	028_H	U	SE, SV	Application Reset	1161
COMBCU_RRCTRL	Round-robin group control	030_H	P	SV, E, P	Application Reset	1122
COMBCU_PRIOH	Master agent priority register high	034_H	P	SV, E, P	Application Reset	1167
COMBCU_PRIOL	Master agent priority register low	038_H	P	SV, E, P	Application Reset	1170
COMBCU_CON	BCU control register	$03C_H$	P	SV, E, P	Application Reset	1124
COMBCU_ECON	BCU error control capture register	040_H	P	SV, P	Application Reset	1125
COMBCU_EADD	BCU error address capture register	044_H	P	SV, P	Application Reset	1127
COMBCU_EDAT	BCU error data capture register	048_H	P	SV, P	Application Reset	1127
COMBCU_CSAVSTAT	CSAV status register	$04C_H$	P	BE	Debug Reset	1128
COMBCU_ALSTATx (x=0-4)	BCU EDC alarm status register x.	050_H+x*4	P	BE	Application Reset	1180
COMBCU_ALCLRx (x=0-4)	BCU EDC alarm clear register x	070_H+x*4	P	SV, P	Application Reset	1129
COMBCU_ALCTRL	BCU EDC alarm control register	090_H	P	SV, P	Application Reset	1129
COMBCU_FEGEN	FPI error generation control register	094_H	P	SV, SE	Application Reset	1131
COMBCU_PSELECT	BCU performance event select	098_H	P	SV, E, P	Application Reset	1133

(table continues...)

Table 615 (continued) Register overview - COMBCU (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
COMBCU_QOSICTRL	QoS time interval control register	100_H	P	SV, E, P	Application Reset	1133
COMBCU_QOSISTAT	QoS time interval counter status	104_H	P	BE	Application Reset	1134
COMBCU_QOSMx_CTRL (x=0-15)	QoS master agent x control register	110_H+x*8	P	SV, E, P	Application Reset	1134
COMBCU_QOSMx_STAT (x=0-15)	QoS master agent x status register	114_H+x*8	P	BE	Application Reset	1135
COMBCU_ARPx_TAGIDA (x=0-7)	ARP x TAG-ID assignment register A	190_H+x*$1C_H$	P	P, SV, E	Application Reset	1136
COMBCU_ARPx_TAGIDB (x=0-7)	ARP x TAG-ID assignment register B	194_H+x*$1C_H$	P	SV, P, E	Application Reset	1137
COMBCU_ARPx_VM (x=0-7)	ARP x VM assignment register	198_H+x*$1C_H$	P	SV, P, E	Application Reset	1137
COMBCU_ARPx_THR (x=0-7)	ARP x threshold register	$19C_H$+x*$1C_H$	P	SV, E, P	Application Reset	1138
COMBCU_ARPx_CTRL (x=0-7)	ARP x control register	$1A0_H$+x*$1C_H$	P	SV, E, P	Application Reset	1138
COMBCU_ARPx_CCC (x=0-7)	ARP x cycle count capture register	$1A4_H$+x*$1C_H$	P	SV, E, P	Application Reset	1139
COMBCU_ARPx_CCV (x=0-7)	ARP x current counter value register	$1A8_H$+x*$1C_H$	P	BE	Application Reset	1140

7.8.4.4 Register overview - CSBCU (ascending offset address)

Table 616 Register overview - CSBCU (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CSBCU_OCS	OCDS control and status register	004_H	P	SV, P	Debug Reset	1114
CSBCU_ID	Module identification register	008_H	P	BE	PowerOn Reset	1115

(table continues...)

Table 616 (continued) Register overview - CSBCU (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CSBCU_PROTE	PROT register endinit (CS specific default configuration)	$00C_H$	U	SV, PROT	Application Reset	1152
CSBCU_PROTSE	PROT register safe endinit (CS specific default configuration)	010_H	U	SV, PROT	Application Reset	1156
CSBCU_ACCEN_WRA	Write access enable register A (CS specific default configuration)	014_H	U	SE, SV	Application Reset	1161
CSBCU_ACCEN_WRB	Write access enable register B	018_H	U	SE, SV	Application Reset	1162
CSBCU_ACCEN_RDA	Read access enable register A (CS specific default configuration)	$01C_H$	U	SE, SV	Application Reset	1162
CSBCU_ACCEN_RDB	Read access enable register B (CS specific default configuration)	020_H	U	SE, SV	Application Reset	1163
CSBCU_ACCEN_VM	VM access enable register	024_H	U	SE, SV	Application Reset	1163
CSBCU_ACCEN_PRS	PRS access enable register	028_H	U	SE, SV	Application Reset	1164
CSBCU_RRCTRL	Round-robin group control	030_H	P	SV, E, P	Application Reset	1122
CSBCU_PRIOH	Master agent priority register high	034_H	P	SV, E, P	Application Reset	1166
CSBCU_PRIOL	Master agent priority register low	038_H	P	SV, E, P	Application Reset	1169
CSBCU_CON	BCU control register	$03C_H$	P	SV, E, P	Application Reset	1124
CSBCU_ECON	BCU error control capture register	040_H	P	SV, P	Application Reset	1125
CSBCU_EADD	BCU error address capture register	044_H	P	SV, P	Application Reset	1127
CSBCU_EDAT	BCU error data capture register	048_H	P	SV, P	Application Reset	1127
CSBCU_CSAVSTAT	CSAV status register	$04C_H$	P	BE	Debug Reset	1128
CSBCU_ALSTATx (x=0-4)	BCU EDC alarm status register x.	050_H+x*4	P	BE	Application Reset	1173
CSBCU_ALCLRx (x=0-4)	BCU EDC alarm clear register x	070_H+x*4	P	SV, P	Application Reset	1129

(table continues...)

Table 616 (continued) Register overview - CSBCU (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CSBCU_ALCTRL	BCU EDC alarm control register	090_H	P	SV, P	Application Reset	1129
CSBCU_FEGEN	FPI error generation control register	094_H	P	SV, SE	Application Reset	1131
CSBCU_PSELECT	BCU performance event select	098_H	P	SV, E, P	Application Reset	1133

7.8.4.5 Register overview - SBCU (ascending offset address)

Table 617 Register overview - SBCU (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
SBCU_OCS	OCDS control and status register	004_H	P	SV, P	Debug Reset	1114
SBCU_ID	Module identification register	008_H	P	BE	PowerOn Reset	1115
SBCU_PROTE	PROT register endinit	$00C_H$	U	SV, PROT	Application Reset	1150
SBCU_PROTSE	PROT register safe endinit	010_H	U	SV, PROT	Application Reset	1154
SBCU_ACCEN_WRA	Write access enable register A	014_H	U	SE, SV	Application Reset	1157
SBCU_ACCEN_WRB	Write access enable register B	018_H	U	SE, SV	Application Reset	1158
SBCU_ACCEN_RDA	Read access enable register A	$01C_H$	U	SE, SV	Application Reset	1159
SBCU_ACCEN_RDB	Read access enable register B	020_H	U	SE, SV	Application Reset	1159
SBCU_ACCEN_VM	VM access enable register	024_H	U	SE, SV	Application Reset	1160
SBCU_ACCEN_PRS	PRS access enable register	028_H	U	SE, SV	Application Reset	1161
SBCU_RRCTRL	Round-robin group control	030_H	P	SV, E, P	Application Reset	1122
SBCU_PRIOH	Master agent priority register high	034_H	P	SV, E, P	Application Reset	1165
SBCU_PRIOL	Master agent priority register low	038_H	P	SV, E, P	Application Reset	1168
SBCU_CON	BCU control register	$03C_H$	P	SV, E, P	Application Reset	1124
SBCU_ECON	BCU error control capture register	040_H	P	SV, P	Application Reset	1125

(table continues...)

Table 617 (continued) Register overview - SBCU (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
SBCU_EADD	BCU error address capture register	044_H	P	SV, P	Application Reset	1127
SBCU_EDAT	BCU error data capture register	048_H	P	SV, P	Application Reset	1127
SBCU_CSAVSTAT	CSAV status register	$04C_H$	P	BE	Debug Reset	1128
SBCU_ALSTATx (x=0-4)	BCU EDC alarm status register x.	050_H+x*4	P	BE	Application Reset	1171
SBCU_ALCLRx (x=0-4)	BCU EDC alarm clear register x	070_H+x*4	P	SV, P	Application Reset	1129
SBCU_ALCTRL	BCU EDC alarm control register	090_H	P	SV, P	Application Reset	1129
SBCU_FEGEN	FPI error generation control register	094_H	P	SV, SE	Application Reset	1131
SBCU_PSELECT	BCU performance event select	098_H	P	SV, E, P	Application Reset	1133
SBCU_QOSICTRL	QoS time interval control register	100_H	P	SV, E, P	Application Reset	1133
SBCU_QOSISTAT	QoS time interval counter status	104_H	P	BE	Application Reset	1134
SBCU_QOSMx_CTRL (x=0-15)	QoS master agent x control register	110_H+x*8	P	SV, E, P	Application Reset	1134
SBCU_QOSMx_STAT (x=0-15)	QoS master agent x status register	114_H+x*8	P	BE	Application Reset	1135
SBCU_ARPx_TAGIDA (x=0-7)	ARP x TAG-ID assignment register A	190_H+x*$1C_H$	P	P, SV, E	Application Reset	1136
SBCU_ARPx_TAGIDB (x=0-7)	ARP x TAG-ID assignment register B	194_H+x*$1C_H$	P	SV, P, E	Application Reset	1137
SBCU_ARPx_VM (x=0-7)	ARP x VM assignment register	198_H+x*$1C_H$	P	SV, P, E	Application Reset	1137
SBCU_ARPx_THR (x=0-7)	ARP x threshold register	$19C_H$+x*$1C_H$	P	SV, E, P	Application Reset	1138
SBCU_ARPx_CTRL (x=0-7)	ARP x control register	$1A0_H$+x*$1C_H$	P	SV, E, P	Application Reset	1138
SBCU_ARPx_CCC (x=0-7)	ARP x cycle count capture register	$1A4_H$+x*$1C_H$	P	SV, E, P	Application Reset	1139

(table continues...)

Table 617 (continued) Register overview - SBCU (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
SBCU_ARPx_CCV (x=0-7)	ARP x current counter value register	$1A8_H$+x* $1C_H$	P	BE	Application Reset	1140

7.8.4.6 Register overview - TBCU (ascending offset address)

Table 618 Register overview - TBCU (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
TBCU_OCS	OCDS control and status register	004_H	P	SV, P	Debug Reset	1114
TBCU_ID	Module identification register	008_H	P	BE	PowerOn Reset	1115
TBCU_PROTE	PROT register endinit	$00C_H$	U	SV, PROT	Application Reset	1150
TBCU_PROTSE	PROT register safe endinit	010_H	U	SV, PROT	Application Reset	1154
TBCU_ACCEN_WRA	Write access enable register A	014_H	U	SE, SV	Application Reset	1157
TBCU_ACCEN_WRB	Write access enable register B	018_H	U	SE, SV	Application Reset	1158
TBCU_ACCEN_RDA	Read access enable register A	$01C_H$	U	SE, SV	Application Reset	1159
TBCU_ACCEN_RDB	Read access enable register B	020_H	U	SE, SV	Application Reset	1159
TBCU_ACCEN_VM	VM access enable register	024_H	U	SE, SV	Application Reset	1160
TBCU_ACCEN_PRS	PRS access enable register	028_H	U	SE, SV	Application Reset	1161
TBCU_RRCTRL	Round-robin group control	030_H	P	SV, E, P	Application Reset	1122
TBCU_PRIOH	Master agent priority register high	034_H	P	SV, E, P	Application Reset	1166
TBCU_PRIOL	Master agent priority register low	038_H	P	SV, E, P	Application Reset	1169
TBCU_CON	BCU control register	$03C_H$	P	SV, E, P	Application Reset	1124
TBCU_ECON	BCU error control capture register	040_H	P	SV, P	Application Reset	1125
TBCU_EADD	BCU error address capture register	044_H	P	SV, P	Application Reset	1127
TBCU_EDAT	BCU error data capture register	048_H	P	SV, P	Application Reset	1127

(table continues...)

Table 618 (continued) Register overview - TBCU (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
TBCU_CSAVSTAT	CSAV status register	$04C_H$	P	BE	Debug Reset	1128
TBCU_ALSTATx (x=0-4)	BCU EDC alarm status register x.	050_H+x*4	P	BE	Application Reset	1173
TBCU_ALCLRx (x=0-4)	BCU EDC alarm clear register x	070_H+x*4	P	SV, P	Application Reset	1129
TBCU_ALCTRL	BCU EDC alarm control register	090_H	P	SV, P	Application Reset	1129
TBCU_FEGEN	FPI error generation control register	094_H	P	SV, SE	Application Reset	1131
TBCU_PSELECT	BCU performance event select	098_H	P	SV, E, P	Application Reset	1133

7.8.4.7 PROT register endinit

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner and in some PROT states, the secure master to update the PROT state as described in the PROT mechanism.

The PROTE register controls lock and unlock of the local Endinit (E) protected control registers.

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner and in some PROT states, the secure master to update the PROT state as described in the PROT mechanism

SBCU_PROTE Offset address: $00C_H$

PROT register endinit Application Reset value: $0000\ 0000_H$

COMBCU_PROTE Offset address: $00C_H$

PROT register endinit Application Reset value: $0000\ 0000_H$

TBCU_PROTE Offset address: $00C_H$

PROT register endinit Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OWEN	**ODEF**	**TAGID**						**PRSEN**	**PRS**			**VMEN**	**VM**		
w	rw	rw						rw	rw			rw	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												**SWEN**	**STATE**		
r												w	rwh		

Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
VM	18:16	rw	**Virtual machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual machine definition enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.
PRSEN	23	rw	**Protection set definition enable for PROT owner** 0_B **Disable**: PRS is not part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG-ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; should be written with 0.

Table 619 Access mode restrictions sorted by descending priority

Applies to SBCU_PROTE

Applies to COMBCU_PROTE

Applies to TBCU_PROTE

Mode name	Access mode		Description
write 1 to .**SWEN**	rwh	STATE	
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

7.8.4.8 PROT register endinit (CS specific default configuration)

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner and in some PROT states, the secure master to update the PROT state as described in the PROT mechanism.

The PROTE register controls lock and unlock of the local Endinit (E) protected control registers.

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner and in some PROT states, the secure master to update the PROT state as described in the PROT mechanism

CSBCU_PROTE Offset address: $00C_H$

PROT register endinit [CS specific default configuration) Application Reset value: $4C00\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OWEN	**ODEF**	**TAGID**						**PRSEN**	**PRS**			**VMEN**	**VM**		
w	rw	rw						rw	rw			rw	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												**SWEN**	**STATE**		
r												w	rwh		

Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)

(table continues...)

(continued)

Field	Bits	Type	Description
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
VM	18:16	rw	**Virtual machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual machine definition enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.
PRSEN	23	rw	**Protection set definition enable for PROT owner** 0_B **Disable**: PRS is not part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG-ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; should be written with 0.

Table 620 Access mode restrictions of CSBCU_PROTE sorted by descending priority

Mode name	Access mode		Description
write 1 to .**SWEN**	rwh	STATE	
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

7.8.4.9 PROT register safe endinit

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner and in some PROT states, the secure master to update the PROT state as described in the PROT mechanism.

The PROTSE register controls lock and unlock of the local Safe Endinit (SE) protected control registers.

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner and in some PROT states, the secure master to update the PROT state as described in the PROT mechanism

SBCU_PROTSE Offset address: 010_H

PROT register safe endinit Application Reset value: $0000\ 0000_H$

COMBCU_PROTSE Offset address: 010_H

PROT register safe endinit Application Reset value: $0000\ 0000_H$

TBCU_PROTSE Offset address: 010_H

PROT register safe endinit Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OWEN	ODEF	TAGID						PRSEN	PRS			VMEN	VM		
w	rw	rw						rw	rw			rw	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												SWEN	STATE		
r												w	rwh		

Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
VM	18:16	rw	**Virtual machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.

(table continues...)

(continued)

Field	Bits	Type	Description
VMEN	19	rw	**Virtual machine definition enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.
PRSEN	23	rw	**Protection set definition enable for PROT owner** 0_B **Disable**: PRS is not part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG-ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; should be written with 0.

Table 621 Access mode restrictions sorted by descending priority

Applies to SBCU_PROTSE

Applies to COMBCU_PROTSE

Applies to TBCU_PROTSE

Mode name	Access mode		Description
write 1 to .**SWEN**	rwh	STATE	
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

7.8.4.10 PROT register safe endinit (CS specific default configuration)

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner and in some PROT states, the secure master to update the PROT state as described in the PROT mechanism.

The PROTSE register controls lock and unlock of the local Safe Endinit (SE) protected control registers.

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner and in some PROT states, the secure master to update the PROT state as described in the PROT mechanism

CSBCU_PROTSE Offset address: 010_H

PROT register safe endinit [CS specific default configuration) Application Reset value: $4C00\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OWEN	ODEF	TAGID						PRSEN	PRS			VMEN	VM		
w	rw	rw						rw	rw			rw	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												SWEN	STATE		
r												w	rwh		

Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
VM	18:16	rw	**Virtual machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual machine definition enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.

(table continues...)

(continued)

Field	Bits	Type	Description
PRSEN	23	rw	**Protection set definition enable for PROT owner** 0_B **Disable**: PRS is not part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG-ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; should be written with 0.

Table 622 Access mode restrictions of CSBCU_PROTSE sorted by descending priority

Mode name	Access mode		Description
write 1 to .**SWEN**	rwh	STATE	
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

7.8.4.11 Write access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

SBCU_ACCEN_WRA Offset address: 014_H

Write access enable register A Application Reset value: $1000\ 0003_H$

COMBCU_ACCEN_WRA Offset address: 014_H

Write access enable register A Application Reset value: $1000\ 0003_H$

TBCU_ACCEN_WRA Offset address: 014_H

Write access enable register A Application Reset value: $1000\ 0003_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q 0_B Disabled for write access 1_B Enabled for write access

7.8.4.12 Write access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

SBCU_ACCEN_WRB Offset address: 018_H

Write access enable register B Application Reset value: $0000\ 0000_H$

COMBCU_ACCEN_WRB Offset address: 018_H

Write access enable register B Application Reset value: $0000\ 0000_H$

TBCU_ACCEN_WRB Offset address: 018_H

Write access enable register B Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q 0_B Disabled for write access 1_B Enabled for write access
0	31:8	r	**Reserved** Read as 0; should be written with 0.

7.8.4.13 Read access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

SBCU_ACCEN_RDA Offset address: $01C_H$

Read access enable register A Application Reset value: FFFF FFFF$_H$

COMBCU_ACCEN_RDA Offset address: $01C_H$

Read access enable register A Application Reset value: FFFF FFFF$_H$

TBCU_ACCEN_RDA Offset address: $01C_H$

Read access enable register A Application Reset value: FFFF FFFF$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q 0_B Disabled for read access 1_B Enabled for read access

7.8.4.14 Read access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

SBCU_ACCEN_RDB Offset address: 020_H

Read access enable register B Application Reset value: 0000 00FF$_H$

COMBCU_ACCEN_RDB Offset address: 020_H

Read access enable register B Application Reset value: 0000 00FF$_H$

TBCU_ACCEN_RDB Offset address: 020_H

Read access enable register B Application Reset value: 0000 00FF$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q 0_B Disabled for read access 1_B Enabled for read access
0	31:8	r	**Reserved** Read as 0; should be written with 0.

7.8.4.15 VM access enable register

This register defines which virtual machine encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each VMx.

SBCU_ACCEN_VM Offset address: 024_H

VM access enable register Application Reset value: $00FF\ 00FF_H$

COMBCU_ACCEN_VM Offset address: 024_H

VM access enable register Application Reset value: $00FF\ 00FF_H$

TBCU_ACCEN_VM Offset address: 024_H

VM access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for VM ID q** This bit enables read access to the access protected region for transactions with the VM ID q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for VM ID q** This bit enables write access to the access protected region for transactions with the VM ID q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

7.8.4.16 PRS access enable register

This register defines which protection register set encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each PRSx.

SBCU_ACCEN_PRS — Offset address: 028_H

PRS access enable register — Application Reset value: $00FF\ 00FF_H$

COMBCU_ACCEN_PRS — Offset address: 028_H

PRS access enable register — Application Reset value: $00FF\ 00FF_H$

TBCU_ACCEN_PRS — Offset address: 028_H

PRS access enable register — Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for PRS q** This bit enables read access to the access protected region for transactions in the PRS q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for PRS q** This bit enables write access to the access protected region for transactions in the PRS q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

7.8.4.17 Write access enable register A (CS specific default configuration)

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

CSBCU_ACCEN_WRA — Offset address: 014_H

Write access enable register A [CS specific default configuration) — Application Reset value: $0000\ 1000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q 0_B Disabled for write access 1_B Enabled for write access

7.8.4.18 Write access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

CSBCU_ACCEN_WRB Offset address: 018_H

Write access enable register B Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q 0_B Disabled for write access 1_B Enabled for write access
0	31:8	r	**Reserved** Read as 0; should be written with 0.

7.8.4.19 Read access enable register A (CS specific default configuration)

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

CSBCU_ACCEN_RDA Offset address: $01C_H$

Read access enable register A [CS specific default configuration) Application Reset value: $0000\ 1000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q 0_B Disabled for read access 1_B Enabled for read access

7.8.4.20 Read access enable register B (CS specific default configuration)

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

CSBCU_ACCEN_RDB Offset address: 020_H

Read access enable register B [CS specific default configuration) Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q 0_B Disabled for read access 1_B Enabled for read access
0	31:8	r	**Reserved** Read as 0; should be written with 0.

7.8.4.21 VM access enable register

This register defines which virtual machine encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each VMx.

CSBCU_ACCEN_VM Offset address: 024_H

VM access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for VM ID q** This bit enables read access to the access protected region for transactions with the VM ID q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for VM ID q** This bit enables write access to the access protected region for transactions with the VM ID q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

7.8.4.22 PRS access enable register

This register defines which protection register set encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each PRSx.

CSBCU_ACCEN_PRS Offset address: 028_H

PRS access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for PRS q** This bit enables read access to the access protected region for transactions in the PRS q. 0_B Disabled for read access 1_B Enabled for read access

(table continues...)

(continued)

Field	Bits	Type	Description
WRq (q=00-07)	q+16	rw	**Write access enable for PRS q** This bit enables write access to the access protected region for transactions in the PRS q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

7.8.4.23 Master agent priority register high

Priority 4 and priority 8 are implemented as round-robin groups.

Master agents can be mapped to a round-robin group by configuration with priority 4 or priority 8.

A detailed description of the arbitration mechanisms can be found in the System Interconnect FPI chapter.

SBCU_PRIOH Offset address: 034_H

Master agent priority register high Application Reset value: $0888\ 8888_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVD28				MASTER14				MASTER13				MASTER12			
rw				rw				rw				rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MASTER11				MASTER10				MASTER9				MASTER8			
rw				rw				rw				rw			

Field	Bits	Type	Description
MASTER8	3:0	rw	**CPU0 Priority (Index 8)** CPU0 priority for access on the SPB (Index 8). On the SPB, this Master Agent is used in the EDC implementation for Req/Lock error injection.
MASTER9	7:4	rw	**CPU1 Priority (Index 9)** CPU1 priority for access on the SPB (Index 9).
MASTER10	11:8	rw	**CPU2 Priority (Index 10)** CPU2 priority for access on the SPB (Index 10).
MASTER11	15:12	rw	**CPU3 Priority (Index 11)** CPU3 priority for access on the SPB (Index 11).
MASTER12	19:16	rw	**CPU4 Priority (Index 12)** CPU4 priority for access on the SPB (Index 12).
MASTER13	23:20	rw	**CPU5 Priority (Index 13)** CPU5 priority for access on the SPB (Index 13).
MASTER14	27:24	rw	**CPUCS Priority (Index 14)** CPUCS priority for access on the SPB (Index 14).
RSVD28	31:28	rw	**Reserved** Read as reset value or last written value; should be written with 0.

7.8.4.24 Master agent priority register high

Priority 4 and priority 8 are implemented as round-robin groups.
Master agents can be mapped to a round-robin group by configuration with priority 4 or priority 8.
A detailed description of the arbitration mechanisms can be found in the System Interconnect FPI chapter.

TBCU_PRIOH Offset address: 034_H

Master agent priority register high Application Reset value: $0000\ 0888_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVD28				RSVD24				RSVD20				RSVD16			
rw				rw				rw				rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD12				MASTER10				MASTER9				MASTER8			
rw				rw				rw				rw			

Field	Bits	Type	Description
MASTER8	3:0	rw	**SRI4 to TPB Bridge Master Agent (Index 8)** S2F Bridge priority for access on the TPB (Index 8). On the TPB, this Master Agent is used in the EDC implementation for Req/Lock error injection.
MASTER9	7:4	rw	**Cerberus/IOC Master Agent (Index 9)** Cerberus priority for access on the TPB (Index 9).
MASTER10	11:8	rw	**IOCE Master Agent (Index 10)** IOCE priority for access on the TPB (Index 10).
RSVD12	15:12	rw	**Reserved** Read as reset value or last written value; should be written with 0.
RSVD16	19:16	rw	**Reserved** Read as reset value or last written value; should be written with 0.
RSVD20	23:20	rw	**Reserved** Read as reset value or last written value; should be written with 0.
RSVD24	27:24	rw	**Reserved** Read as reset value or last written value; should be written with 0.
RSVD28	31:28	rw	**Reserved** Read as reset value or last written value; should be written with 0.

7.8.4.25 Master agent priority register high

Priority 4 and priority 8 are implemented as round-robin groups.
Master agents can be mapped to a round-robin group by configuration with priority 4 or priority 8.
A detailed description of the arbitration mechanisms can be found in the System Interconnect FPI chapter.

CSBCU_PRIOH Offset address: 034_H

Master agent priority register high Application Reset value: $0000\ 0008_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVD28				RSVD24				RSVD20				RSVD16			
rw				rw				rw				rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD12				RSVD8				RSVD4				MASTER8			
rw				rw				rw				rw			

Field	Bits	Type	Description
MASTER8	3:0	rw	**SRI4 to CSPB Bridge Master Agent (Index 8)** S2F Bridge priority for access on the CSPB (Index 8). On the CSPB, this Master Agent is used in the EDC implementation for Req/Lock error injection.
RSVD4	7:4	rw	**Reserved** Read as reset value or last written value; should be written with 0.
RSVD8	11:8	rw	**Reserved** Read as reset value or last written value; should be written with 0.
RSVD12	15:12	rw	**Reserved** Read as reset value or last written value; should be written with 0.
RSVD16	19:16	rw	**Reserved** Read as reset value or last written value; should be written with 0.
RSVD20	23:20	rw	**Reserved** Read as reset value or last written value; should be written with 0.
RSVD24	27:24	rw	**Reserved** Read as reset value or last written value; should be written with 0.
RSVD28	31:28	rw	**Reserved** Read as reset value or last written value; should be written with 0.

7.8.4.26 Master agent priority register high

Priority 4 and priority 8 are implemented as round-robin groups.
Master agents can be mapped to a round-robin group by configuration with priority 4 or priority 8.
A detailed description of the arbitration mechanisms can be found in the System Interconnect FPI chapter.

COMBCU_PRIOH Offset address: 034_H

Master agent priority register high Application Reset value: $0888\ 8888_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVD28				MASTER14				MASTER13				MASTER12			
rw				rw				rw				rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MASTER11				MASTER10				MASTER9				MASTER8			
rw				rw				rw				rw			

Field	Bits	Type	Description
MASTER8	3:0	rw	**CPU0 Priority (Index 8)** CPU0 priority for access on the COMPB (Index 8). On the COMPB, this Master Agent is used in the EDC implementation for Req/Lock error injection.
MASTER9	7:4	rw	**CPU1 Priority (Index 9)** CPU1 priority for access on the COMPB (Index 9).
MASTER10	11:8	rw	**CPU2 Priority (Index 10)** CPU2 priority for access on the COMPB (Index 10).
MASTER11	15:12	rw	**CPU3 Priority (Index 11)** CPU3 priority for access on the COMPB (Index 11).
MASTER12	19:16	rw	**CPU4 Priority (Index 12)** CPU4 priority for access on the COMPB (Index 12).
MASTER13	23:20	rw	**CPU5 Priority (Index 13)** CPU5 priority for access on the COMPB (Index 13).
MASTER14	27:24	rw	**CPUCS Priority (Index 14)** CPUCS priority for access on the COMPB (Index 14).
RSVD28	31:28	rw	**Reserved** Read as reset value or last written value; should be written with 0.

7.8.4.27 Master agent priority register low

Priority 4 and priority 8 are implemented as round-robin groups.
Master agents can be mapped to a round-robin group by configuration with priority 4 or priority 8.
A detailed description of the arbitration mechanisms can be found in the System Interconnect FPI chapter.

SBCU_PRIOL Offset address: 038_H

Master agent priority register low Application Reset value: $0444\ 4444_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVD28				RSVD24				RSVD20				RSVD16			
rw				rw				rw				rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD12				MASTER2				MASTER1				MASTER0			
rw				rw				rw				rw			

Field	Bits	Type	Description
MASTER0	3:0	rw	**DMA0 Priority (Index 0)** DMA0 priority for access on the SPB (Index 0).
MASTER1	7:4	rw	**DMA1 Priority (Index 1)** DMA1 priority for access on the SPB (Index 1).
MASTER2	11:8	rw	**SRI to SPB Bridge Master Agent (Index 2)** S2F Bridge priority for access on the SPB (Index 2).

(table continues...)

(continued)

Field	Bits	Type	Description
RSVD12	15:12	rw	**Reserved** Read as reset value or last written value; should be written with 0.
RSVD16	19:16	rw	**Reserved** Read as reset value or last written value; should be written with 0.
RSVD20	23:20	rw	**Reserved** Read as reset value or last written value; should be written with 0.
RSVD24	27:24	rw	**Reserved** Read as reset value or last written value; should be written with 0.
RSVD28	31:28	rw	**Reserved** Read as reset value or last written value; should be written with 0.

7.8.4.28 Master agent priority register low

Priority 4 and priority 8 are implemented as round-robin groups.
Master agents can be mapped to a round-robin group by configuration with priority 4 or priority 8.
A detailed description of the arbitration mechanisms can be found in the System Interconnect FPI chapter.

CSBCU_PRIOL Offset address: 038_H

Master agent priority register low Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVD15				RSVD14				RSVD13				RSVD12			
rw				rw				rw				rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD11				RSVD10				RSVD9				RSVD8			
rw				rw				rw				rw			

Field	Bits	Type	Description
RSVDi (i=8-15)	4*i-29:4*i-32	rw	**Reserved** Read as reset value or last written value; should be written with 0.

7.8.4.29 Master agent priority register low

Priority 4 and priority 8 are implemented as round-robin groups.
Master agents can be mapped to a round-robin group by configuration with priority 4 or priority 8.
A detailed description of the arbitration mechanisms can be found in the System Interconnect FPI chapter.

TBCU_PRIOL Offset address: 038_H

Master agent priority register low Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVD15				RSVD14				RSVD13				RSVD12			
rw				rw				rw				rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD11				RSVD10				RSVD9				RSVD8			
rw				rw				rw				rw			

Field	Bits	Type	Description
RSVDi (i=8-15)	4*i-29:4*i-32	rw	**Reserved** Read as reset value or last written value; should be written with 0.

7.8.4.30 Master agent priority register low

Priority 4 and priority 8 are implemented as round-robin groups.
Master agents can be mapped to a round-robin group by configuration with priority 4 or priority 8.
A detailed description of the arbitration mechanisms can be found in the System Interconnect FPI chapter.

COMBCU_PRIOL Offset address: 038_H

Master agent priority register low Application Reset value: $0444\ 4444_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVD28				MASTER6				MASTER5				MASTER4			
rw				rw				rw				rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD12				MASTER2				MASTER1				MASTER0			
rw				rw				rw				rw			

Field	Bits	Type	Description
MASTER0	3:0	rw	**DMA0 Priority (Index 0)** DMA0 priority for access on the COMPB (Index 0).
MASTER1	7:4	rw	**DMA1 Priority (Index 1)** DMA1 priority for access on the COMPB (Index 1).
MASTER2	11:8	rw	**SRI to COMPB Bridge Master Agent (Index 2)** S2F Bridge priority for access on the COMPB (Index 2).
RSVD12	15:12	rw	**Reserved** Read as reset value or last written value; should be written with 0.
MASTER4	19:16	rw	**DRE Priority (Index 4)** DRE priority for access on the SPB (Index 4).
MASTER5	23:20	rw	**HSSL0 Priority (Index 5)** HSSL0 priority for access on the SPB (Index 5).
MASTER6	27:24	rw	**HSSL1 Priority (Index 6)** HSSL1 priority for access on the SPB (Index 6).

(table continues...)

(continued)

Field	Bits	Type	Description
RSVD28	31:28	rw	**Reserved** Read as reset value or last written value; should be written with 0.

7.8.4.31 BCU EDC alarm status register x. (x=0)

The BCU provides one alarm status register bit for each implemented FPI master agent and FPI slave agent. The assignment of the FPI master-slave and master-agent alarm signals is contained in the bit descriptions of the product-specific ALSTAT register description. The source of an alarm signal is described there using <module name>_S for a slave agent interface, <module name>_M for a master agent interface.

SBCU_ALSTATx (x=0) Offset address: 050_H+x*4

BCU EDC alarm status register x. Application Reset value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AL31	AL30	AL29	AL28	AL27	AL26	AL25	AL24	AL23	AL22	AL21	AL20	AL19	AL18	AL17	AL16
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AL15	AL14	AL13	AL12	AL11	AL10	AL09	AL08	AL07	AL06	AL05	AL04	AL03	AL02	AL01	AL00
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
ALy (y=00)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **SBCU_S**: EDC error was detected by the SBCU slave agent.
ALy (y=01-03,05,11,13-14,22-23,25-31)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected.
ALy (y=04)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **SCU_S**: EDC error was detected by the SCU slave agent.
ALy (y=06)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **WTU_S**: EDC error was detected by the WTU slave agent.

(table continues...)

(continued)

Field	Bits	Type	Description
ALy (y=07)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **PMS_S**: EDC error was detected by the PMS slave agent that also provides access to the SCR.
ALy (y=08)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **SMM_S**: EDC error was detected by the SMM slave agent.
ALy (y=09)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **TRI_S**: EDC error was detected by the TRI slave agent.
ALy (y=10)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **VTMON_S**: EDC error was detected by the VTMON slave agent.
ALy (y=12)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **CLOCK_S**: EDC error was detected by the CLOCK slave agent.
ALy (y=15)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **VMT0_S**: EDC error was detected by the VMT0 slave agent.
ALy (y=16)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **VMT1_S**: EDC error was detected by the VMT1 slave agent.

(table continues...)

(continued)

Field	Bits	Type	Description
ALy (y=17)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **VMT2_S**: EDC error was detected by the VMT2 slave agent.
ALy (y=18)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **VMT3_S**: EDC error was detected by the VMT3 slave agent.
ALy (y=19)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **VMT4_S**: EDC error was detected by the VMT4 slave agent.
ALy (y=20)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **VMT5_S**: EDC error was detected by the VMT5 slave agent.
ALy (y=21)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **VMT6_S**: EDC error was detected by the VMT6 slave agent.
ALy (y=24)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **HSPHY_S**: EDC error was detected by the HSPHY slave agent.

7.8.4.32 BCU EDC alarm status register x. (x=1,4)

SBCU_ALSTATx (x=1,4)	Offset address:	050_H+x*4
BCU EDC alarm status register x.	Application Reset value:	0000 0000_H
CSBCU_ALSTATx (x=1,4)	Offset address:	050_H+x*4
BCU EDC alarm status register x.	Application Reset value:	0000 0000_H
TBCU_ALSTATx (x=1,4)	Offset address:	050_H+x*4
BCU EDC alarm status register x.	Application Reset value:	0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AL31	AL30	AL29	AL28	AL27	AL26	AL25	AL24	AL23	AL22	AL21	AL20	AL19	AL18	AL17	AL16
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AL15	AL14	AL13	AL12	AL11	AL10	AL09	AL08	AL07	AL06	AL05	AL04	AL03	AL02	AL01	AL00
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
ALy (y=00-31)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected.

7.8.4.33 BCU EDC alarm status register x. (x=2)

SBCU_ALSTATx (x=2) Offset address: 050_H+x*4

BCU EDC alarm status register x. Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AL31	AL30	AL29	AL28	AL27	AL26	AL25	AL24	AL23	AL22	AL21	AL20	AL19	AL18	AL17	AL16
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AL15	AL14	AL13	AL12	AL11	AL10	AL09	AL08	AL07	AL06	AL05	AL04	AL03	AL02	AL01	AL00
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
ALy (y=00)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **P00_S**: EDC error was detected by the PORT 00 slave agent.
ALy (y=01)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **P01_S**: EDC error was detected by the PORT 01 slave agent.

(table continues...)

(continued)

Field	Bits	Type	Description
ALy (y=02)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **P02_S**: EDC error was detected by the PORT 02 slave agent.
ALy (y=03)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **P03_S**: EDC error was detected by the PORT 03 slave agent.
ALy (y=04)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **P04_S**: EDC error was detected by the PORT 04 slave agent.
ALy (y=05,07-08,13-14,19,21-22,30)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected.
ALy (y=06)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **P10_S**: EDC error was detected by the PORT 10 slave agent.
ALy (y=09)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **P13_S**: EDC error was detected by the PORT 13 slave agent.
ALy (y=10)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **P14_S**: EDC error was detected by the PORT 14 slave agent.

(table continues...)

(continued)

Field	Bits	Type	Description
ALy (y=11)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **P15_S**: EDC error was detected by the PORT 15 slave agent.
ALy (y=12)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **P16_S**: EDC error was detected by the PORT 16 slave agent.
ALy (y=15)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **P20_S**: EDC error was detected by the PORT 20 slave agent.
ALy (y=16)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **P21_S**: EDC error was detected by the PORT 21 slave agent.
ALy (y=17)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **P22_S**: EDC error was detected by the PORT 22 slave agent.
ALy (y=18)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **P23_S**: EDC error was detected by the PORT 23 slave agent.
ALy (y=20)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **P25_S**: EDC error was detected by the PORT 25 slave agent.

(table continues...)

(continued)

Field	Bits	Type	Description
ALy (y=23)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **P30_S**: EDC error was detected by the PORT 30 slave agent.
ALy (y=24)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **P31_S**: EDC error was detected by the PORT 31 slave agent.
ALy (y=25)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **P32_S**: EDC error was detected by the PORT 32 slave agent.
ALy (y=26)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **P33_S**: EDC error was detected by the PORT 33 slave agent.
ALy (y=27)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **P34_S**: EDC error was detected by the PORT 34 slave agent.
ALy (y=28)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **P35_S**: EDC error was detected by the PORT 35 slave agent.
ALy (y=29)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **P40_S**: EDC error was detected by the PORT 40 slave agent.

(table continues...)

(continued)

Field	Bits	Type	Description
ALy (y=31)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **SBCU_M**: EDC error was detected by the SBCU master agent.

7.8.4.34 BCU EDC alarm status register x. (x=3)

SBCU_ALSTATx (x=3) Offset address: 050_H+x*4

BCU EDC alarm status register x. Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AL31	AL30	AL29	AL28	AL27	AL26	AL25	AL24	AL23	AL22	AL21	AL20	AL19	AL18	AL17	AL16
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AL15	AL14	AL13	AL12	AL11	AL10	AL09	AL08	AL07	AL06	AL05	AL04	AL03	AL02	AL01	AL00
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
ALy (y=00)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **A_EN**: No or multiple output enables active: A_EN_N (master agent)
ALy (y=01)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **ABORT_EN**: No or multiple output enables active: ABORT_EN_N (master agent)
ALy (y=02)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **ACK_EN**: No or multiple output enables active: ACK_EN_N (default master and slave agent)

(table continues...)

(continued)

Field	Bits	Type	Description
ALy (y=03)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **D_EN**: No or multiple output enables active: D_EN_N (master and slave agent)
ALy (y=04-15,19-23,31)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected.
ALy (y=16)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **DMA0_M**: EDC error was detected by the DMA0 master agent.
ALy (y=17)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **DMA1_M**: EDC error was detected by the DMA1 master agent.
ALy (y=18)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **S2F_3SPB_M**: EDC error was detected by the SRI to SPB bridge master agent.
ALy (y=24)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **CPU0_M**: EDC error was detected by the CPU0 master agent.
ALy (y=25)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **CPU1_M**: EDC error was detected by the CPU1 master agent.

(table continues...)

(continued)

Field	Bits	Type	Description
ALy (y=26)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **CPU2_M**: EDC error was detected by the CPU2 master agent.
ALy (y=27)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **CPU3_M**: EDC error was detected by the CPU3 master agent.
ALy (y=28)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **CPU4_M**: EDC error was detected by the CPU4 master agent.
ALy (y=29)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **CPU5_M**: EDC error was detected by the CPU5 master agent.
ALy (y=30)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **CPUCS_M**: EDC error was detected by the CPUCS master agent.

7.8.4.35 BCU EDC alarm status register x. (x=0)

COMBCU_ALSTATx (x=0) Offset address: 050_H+x*4

BCU EDC alarm status register x. Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AL31	AL30	AL29	AL28	AL27	AL26	AL25	AL24	AL23	AL22	AL21	AL20	AL19	AL18	AL17	AL16
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AL15	AL14	AL13	AL12	AL11	AL10	AL09	AL08	AL07	AL06	AL05	AL04	AL03	AL02	AL01	AL00
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
ALy (y=00)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **COMBCU_S**: EDC error was detected by the COMBCU slave agent.
ALy (y=01)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **DMA0_S**: EDC error was detected by the DMA0 slave agent.
ALy (y=02)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **DMA1_S**: EDC error was detected by the DMA1 slave agent.
ALy (y=03)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **IR_S**: EDC error was detected by the IR slave agent.
ALy (y=04-10,12-13,15-22,24,27,30-31)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected.
ALy (y=11)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **SDMMC_S**: EDC error was detected by the SDMMC slave agent.
ALy (y=14)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **FCE_S**: EDC error was detected by the FCE slave agent.

(table continues...)

(continued)

Field	Bits	Type	Description
ALy (y=23)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **TMADC_S**: EDC error was detected by the TMADC slave agent.
ALy (y=25)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **PSI5_S**: EDC error was detected by the PSI5 slave agent.
ALy (y=26)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **PSI5S0_S**: EDC error was detected by the PSI5S0 slave agent.
ALy (y=28)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **ERAY0_S**: EDC error was detected by the ERAY0 slave agent.
ALy (y=29)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **ERAY1_S**: EDC error was detected by the ERAY1 slave agent.

7.8.4.36 BCU EDC alarm status register x. (x=1)

COMBCU_ALSTATx (x=1) Offset address: 050_H+x*4

BCU EDC alarm status register x. Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AL31	AL30	AL29	AL28	AL27	AL26	AL25	AL24	AL23	AL22	AL21	AL20	AL19	AL18	AL17	AL16
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AL15	AL14	AL13	AL12	AL11	AL10	AL09	AL08	AL07	AL06	AL05	AL04	AL03	AL02	AL01	AL00
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
ALy (y=00)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **QSPI0_S**: EDC error was detected by the QSPI0 slave agent.
ALy (y=01)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **QSPI1_S**: EDC error was detected by the QSPI1 slave agent.
ALy (y=02)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **QSPI2_S**: EDC error was detected by the QSPI2 slave agent.
ALy (y=03)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **QSPI3_S**: EDC error was detected by the QSPI3 slave agent.
ALy (y=04)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **QSPI4_S**: EDC error was detected by the QSPI4 slave agent.
ALy (y=05)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **QSPI5_S**: EDC error was detected by the QSPI5 slave agent.
ALy (y=06)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **QSPI6_S**: EDC error was detected by the QSPI6 slave agent.

(table continues...)

(continued)

Field	Bits	Type	Description
ALy (y=07)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **QSPI7_S**: EDC error was detected by the QSPI7 slave agent.
ALy (y=08)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **MSC0_S**: EDC error was detected by the MSC0 slave agent.
ALy (y=09-11,15-16,31)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected.
ALy (y=12)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **I2C0_S**: EDC error was detected by the I2C0 slave agent.
ALy (y=13)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **I2C1_S**: EDC error was detected by the I2C1 slave agent.
ALy (y=14)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **I2C2_S**: EDC error was detected by the I2C2 slave agent.
ALy (y=17)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **CANXL**: EDC error was detected by the CANXL slave agent.

(table continues...)

(continued)

Field	Bits	Type	Description
ALy (y=18)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **SENT0_S**: EDC error was detected by the SENT0 slave agent.
ALy (y=19)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **SENT1_S**: EDC error was detected by the SENT1 slave agent.
ALy (y=20)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **MCMCAN0_S**: EDC error was detected by the MCMCAN0 slave agent.
ALy (y=21)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **MCMCAN1_S**: EDC error was detected by the MCMCAN1 slave agent.
ALy (y=22)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **MCMCAN2_S**: EDC error was detected by the MCMCAN2 slave agent.
ALy (y=23)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **MCMCAN3_S**: EDC error was detected by the MCMCAN3 slave agent.

(table continues...)

(continued)

Field	Bits	Type	Description
ALy (y=24)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **MCMCAN4_S**: EDC error was detected by the MCMCAN4 slave agent.
ALy (y=25)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **HSSL0_S**: EDC error was detected by the HSSL0 slave agent.
ALy (y=26)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **HSSL1_S**: EDC error was detected by the HSSL1 slave agent.
ALy (y=27)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **HSCT0_S**: EDC error was detected by the HSCT0 slave agent.
ALy (y=28)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **HSCT1_S**: EDC error was detected by the HSCT1 slave agent.
ALy (y=29)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **PCIE0_S**: EDC error was detected by the PCIE0 slave agent.
ALy (y=30)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **PCIE1_S**: EDC error was detected by the PCIE1 slave agent.

7.8.4.37 BCU EDC alarm status register x. (x=2)

COMBCU_ALSTATx (x=2) Offset address: 050_H+x*4

BCU EDC alarm status register x. Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AL31	AL30	AL29	AL28	AL27	AL26	AL25	AL24	AL23	AL22	AL21	AL20	AL19	AL18	AL17	AL16
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AL15	AL14	AL13	AL12	AL11	AL10	AL09	AL08	AL07	AL06	AL05	AL04	AL03	AL02	AL01	AL00
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
ALy (y=00-30)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected.
ALy (y=31)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **SBCU_M**: EDC error was detected by the SBCU master agent.

7.8.4.38 BCU EDC alarm status register x. (x=3)

COMBCU_ALSTATx (x=3) Offset address: 050_H+x*4

BCU EDC alarm status register x. Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AL31	AL30	AL29	AL28	AL27	AL26	AL25	AL24	AL23	AL22	AL21	AL20	AL19	AL18	AL17	AL16
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AL15	AL14	AL13	AL12	AL11	AL10	AL09	AL08	AL07	AL06	AL05	AL04	AL03	AL02	AL01	AL00
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
ALy (y=00)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **A_EN**: No or multiple output enables active: A_EN_N (master agent)

(table continues...)

(continued)

Field	Bits	Type	Description
ALy (y=01)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **ABORT_EN**: No or multiple output enables active: ABORT_EN_N (master agent)
ALy (y=02)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **ACK_EN**: No or multiple output enables active: ACK_EN_N (default master and slave agent)
ALy (y=03)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **D_EN**: No or multiple output enables active: D_EN_N (master and slave agent)
ALy (y=04-15,19,23,31)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected.
ALy (y=16)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **DMA0_M**: EDC error was detected by the DMA0 master agent.
ALy (y=17)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **DMA1_M**: EDC error was detected by the DMA1 master agent.
ALy (y=18)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **S2F_3COMPB_M**: EDC error was detected by the SRI to SPB bridge master agent.

(table continues...)

(continued)

Field	Bits	Type	Description
ALy (y=20)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **DRE_M**: EDC error was detected by the DRE master agent.
ALy (y=21)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **HSSL0_M**: EDC error was detected by the HSSL0 master agent.
ALy (y=22)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **HSSL1_M**: EDC error was detected by the HSSL1 master agent.
ALy (y=24)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **CPU0_M**: EDC error was detected by the CPU0 master agent.
ALy (y=25)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **CPU1_M**: EDC error was detected by the CPU1 master agent.
ALy (y=26)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **CPU2_M**: EDC error was detected by the CPU2 master agent.
ALy (y=27)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **CPU3_M**: EDC error was detected by the CPU3 master agent.

(table continues...)

(continued)

Field	Bits	Type	Description
ALy (y=28)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **CPU4_M**: EDC error was detected by the CPU4 master agent.
ALy (y=29)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **CPU5_M**: EDC error was detected by the CPU5 master agent.
ALy (y=30)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **CPUCS_M**: EDC error was detected by the CPUCS master agent.

7.8.4.39 BCU EDC alarm status register x. (x=4)

COMBCU_ALSTATx (x=4) Offset address: 050_H+x*4

BCU EDC alarm status register x. Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AL31	AL30	AL29	AL28	AL27	AL26	AL25	AL24	AL23	AL22	AL21	AL20	AL19	AL18	AL17	AL16
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AL15	AL14	AL13	AL12	AL11	AL10	AL09	AL08	AL07	AL06	AL05	AL04	AL03	AL02	AL01	AL00
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
ALy (y=00)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **ASCLIN0_S**: EDC error was detected by the ASCLIN0 slave agent.
ALy (y=01)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **ASCLIN1_S**: EDC error was detected by the ASCLIN1 slave agent.

(table continues...)

(continued)

Field	Bits	Type	Description
ALy (y=02)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **ASCLIN2_S**: EDC error was detected by the ASCLIN2 slave agent.
ALy (y=03)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **ASCLIN3_S**: EDC error was detected by the ASCLIN3 slave agent.
ALy (y=04)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **ASCLIN4_S**: EDC error was detected by the ASCLIN4 slave agent.
ALy (y=05)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **ASCLIN5_S**: EDC error was detected by the ASCLIN5 slave agent.
ALy (y=06)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **ASCLIN6_S**: EDC error was detected by the ASCLIN6 slave agent.
ALy (y=07)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **ASCLIN7_S**: EDC error was detected by the ASCLIN7 slave agent.
ALy (y=08)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **ASCLIN8_S**: EDC error was detected by the ASCLIN8 slave agent.

(table continues...)

(continued)

Field	Bits	Type	Description
ALy (y=09)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **ASCLIN9_S**: EDC error was detected by the ASCLIN9 slave agent.
ALy (y=10)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **ASCLIN10_S**: EDC error was detected by the ASCLIN10 slave agent.
ALy (y=11)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **ASCLIN11_S**: EDC error was detected by the ASCLIN11 slave agent.
ALy (y=12)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **ASCLIN12_S**: EDC error was detected by the ASCLIN12 slave agent.
ALy (y=13)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **ASCLIN13_S**: EDC error was detected by the ASCLIN13 slave agent.
ALy (y=14)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **ASCLIN14_S**: EDC error was detected by the ASCLIN14 slave agent.
ALy (y=15)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **ASCLIN15_S**: EDC error was detected by the ASCLIN15 slave agent.

(table continues...)

(continued)

Field	Bits	Type	Description
ALy (y=16)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **ASCLIN16_S**: EDC error was detected by the ASCLIN16 slave agent.
ALy (y=17)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **ASCLIN17_S**: EDC error was detected by the ASCLIN17 slave agent.
ALy (y=18)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **ASCLIN18_S**: EDC error was detected by the ASCLIN18 slave agent.
ALy (y=19)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **ASCLIN19_S**: EDC error was detected by the ASCLIN19 slave agent.
ALy (y=20)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **ASCLIN20_S**: EDC error was detected by the ASCLIN20 slave agent.
ALy (y=21)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **ASCLIN21_S**: EDC error was detected by the ASCLIN21 slave agent.
ALy (y=22)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **ASCLIN22_S**: EDC error was detected by the ASCLIN22 slave agent.

(table continues...)

(continued)

Field	Bits	Type	Description
ALy (y=23)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **ASCLIN23_S**: EDC error was detected by the ASCLIN23 slave agent.
ALy (y=24)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **ASCLIN24_S**: EDC error was detected by the ASCLIN24 slave agent.
ALy (y=25)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **ASCLIN25_S**: EDC error was detected by the ASCLIN25 slave agent.
ALy (y=26)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **ASCLIN26_S**: EDC error was detected by the ASCLIN26 slave agent.
ALy (y=27)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **ASCLIN27_S**: EDC error was detected by the ASCLIN27 slave agent.
ALy (y=28-31)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected.

7.8.4.40 BCU EDC alarm status register x. (x=0)

CSBCU_ALSTATx (x=0) Offset address: 050_H+x*4

BCU EDC alarm status register x. Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AL31	AL30	AL29	AL28	AL27	AL26	AL25	AL24	AL23	AL22	AL21	AL20	AL19	AL18	AL17	AL16
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AL15	AL14	AL13	AL12	AL11	AL10	AL09	AL08	AL07	AL06	AL05	AL04	AL03	AL02	AL01	AL00
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
ALy (y=00)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **CSBCU_S**: EDC error was detected by the CSBCU slave agent.
ALy (y=01)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **CSCU_S**: EDC error was detected by the CSCU slave agent.
ALy (y=02)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **RNG_S**: EDC error was detected by the RNG slave agent.
ALy (y=03)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **PKC_S**: EDC error was detected by the PKC slave agent.
ALy (y=04-31)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected.

7.8.4.41 BCU EDC alarm status register x. (x=2)

CSBCU_ALSTATx (x=2)

BCU EDC alarm status register x.

Offset address: 050_H+x*4

Application Reset value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AL31	AL30	AL29	AL28	AL27	AL26	AL25	AL24	AL23	AL22	AL21	AL20	AL19	AL18	AL17	AL16
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AL15	AL14	AL13	AL12	AL11	AL10	AL09	AL08	AL07	AL06	AL05	AL04	AL03	AL02	AL01	AL00
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
ALy (y=00-30)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected.
ALy (y=31)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **CSBCU_M**: EDC error was detected by the CSBCU master interface.

7.8.4.42 BCU EDC alarm status register x. (x=3)

CSBCU_ALSTATx (x=3) Offset address: 050_H+x*4

BCU EDC alarm status register x. Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AL31	AL30	AL29	AL28	AL27	AL26	AL25	AL24	AL23	AL22	AL21	AL20	AL19	AL18	AL17	AL16
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AL15	AL14	AL13	AL12	AL11	AL10	AL09	AL08	AL07	AL06	AL05	AL04	AL03	AL02	AL01	AL00
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
ALy (y=00)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **A_EN**: No or multiple output enables active: A_EN_N (master agent)

(table continues...)

(continued)

Field	Bits	Type	Description
ALy (y=01)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **ABORT_EN**: No or multiple output enables active: ABORT_EN_N (master agent)
ALy (y=02)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **ACK_EN**: No or multiple output enables active: ACK_EN_N (default master and slave agent)
ALy (y=03)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **D_EN**: No or multiple output enables active: D_EN_N (master and slave agent)
ALy (y=04-23,25-31)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected.
ALy (y=24)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **S2F_4CSPB_M**: EDC error was detected by the SRI4 to CSPB bridge master agent.

7.8.4.43 BCU EDC alarm status register x. (x=0)

TBCU_ALSTATx (x=0) Offset address: 050_H+x*4

BCU EDC alarm status register x. Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AL31	AL30	AL29	AL28	AL27	AL26	AL25	AL24	AL23	AL22	AL21	AL20	AL19	AL18	AL17	AL16
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AL15	AL14	AL13	AL12	AL11	AL10	AL09	AL08	AL07	AL06	AL05	AL04	AL03	AL02	AL01	AL00
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
ALy (y=00)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **TBCU_S**: EDC error was detected by the TBCU slave agent.
ALy (y=01)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **CBS_S**: EDC error was detected by the CBS slave agent.
ALy (y=02)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **TRIF_S**: EDC error was detected by the TRIF slave agent.
ALy (y=03)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **MCDS4P_S**: EDC error was detected by the MCDS4P slave agent.
ALy (y=04)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **MCDS2P_S**: EDC error was detected by the MCDS2P slave agent.
ALy (y=05)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **F2S_TPB7_S**: EDC error was detected by the TPB to SRI bridge slave agent.
ALy (y=06-31)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected.

7.8.4.44 BCU EDC alarm status register x. (x=2)

TBCU_ALSTATx (x=2) Offset address: 050_H+x*4

BCU EDC alarm status register x. Application Reset value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AL31	AL30	AL29	AL28	AL27	AL26	AL25	AL24	AL23	AL22	AL21	AL20	AL19	AL18	AL17	AL16
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AL15	AL14	AL13	AL12	AL11	AL10	AL09	AL08	AL07	AL06	AL05	AL04	AL03	AL02	AL01	AL00
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
ALy (y=00-30)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected.
ALy (y=31)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **TBCU_M**: EDC error was detected by the CBCU master interface.

7.8.4.45 BCU EDC alarm status register x. (x=3)

TBCU_ALSTATx (x=3) Offset address: 050_H+x*4

BCU EDC alarm status register x. Application Reset value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AL31	AL30	AL29	AL28	AL27	AL26	AL25	AL24	AL23	AL22	AL21	AL20	AL19	AL18	AL17	AL16
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AL15	AL14	AL13	AL12	AL11	AL10	AL09	AL08	AL07	AL06	AL05	AL04	AL03	AL02	AL01	AL00
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
ALy (y=00)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **A_EN**: No or multiple output enables active: A_EN_N (master agent)

(table continues...)

(continued)

Field	Bits	Type	Description
ALy (y=01)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **ABORT_EN**: No or multiple output enables active: ABORT_EN_N (master agent)
ALy (y=02)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **ACK_EN**: No or multiple output enables active: ACK_EN_N (default master and slave agent)
ALy (y=03)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **D_EN**: No or multiple output enables active: D_EN_N (master and slave agent)
ALy (y=04-23,27-31)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected.
ALy (y=24)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **S2F_7TPB_M**: EDC error was detected by the SRI4 to TPB bridge master agent.
ALy (y=25)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **CBS_M**: EDC error was detected by the Cerberus master agent.
ALy (y=26)	y	rh	**Alarm y** The alarm bit shows if an EDC error was detected in an active phase of the related FPI Slave and Master interface. Register bits without constant definition are reserved in this product. 0 means: no EDC error detected. 1_B **IOCE_M**: EDC error was detected by the IOCE master agent.

7.8.5 TC4Dx FPI connectivity

The table below lists interfaces to and from this functional block to other blocks in the device.

Table 623 List of FPI interface signals

Interface signals	I/O	Description
CLOCK_SBCU_fSPB	In	fSPB
CLOCK_COMBCU_fSPB	In	fSPB
CLOCK_CSBCU_fSPB	In	fSPB
CLOCK_TBCU_fTPB	In	fTPB
SMM_SBCU_Applicationreset	In	Application reset input
SMM_COMBCU_Applicationreset	In	Application reset input
SMM_CSBCU_Applicationreset	In	Application reset input
SMM_TBCU_Applicationreset	In	Application reset input
SMM_SBCU_Debugreset	In	Debug reset input
SMM_COMBCU_Debugreset	In	Debug reset input
SMM_CSBCU_Debugreset	In	Debug reset input
SMM_TBCU_Debugreset	In	Debug reset input
SMM_SBCU_Poweronreset	In	Power on reset input
SMM_COMBCU_Poweronreset	In	Power on reset input
SMM_CSBCU_Poweronreset	In	Power on reset input
SMM_TBCU_Poweronreset	In	Power on reset input
SPB_SBCU_OTGB	In	SBCU OTGB interface
COMPB_COMBCU_OTGB	In	COMBCU OTGB interface
CSPB_CSBCU_OTGB	In	CSBCU OTGB interface
SBCU_DEBUG_TRACE	Out	SBCU FPI trace data output
COMBCU_DEBUG_TRACE	Out	COMBCU FPI trace data output
CSBCU_DEBUG_TRACE	Out	CSBCU FPI trace data output
CSRM.CSCU_SBCU_CSAV[1:0]	In	CS activity visibility inputs
CSRM.CSCU_COMBCU_CSAV[1:0]	In	CS activity visibility inputs
CSRM.CSCU_CSBCU_CSAV[1:0]	In	CS activity visibility inputs
CSRM.CSCU_TBCU_CSAV[1:0]	In	CS activity visibility inputs
SBCU_SMU_ALARM_NCS	Out	SBCU non-secure alarm output
COMBCU_SMU_ALARM_NCS	Out	COMBCU non-secure alarm output
CSBCU_SMU_ALARM_NCS	Out	CSBCU non-secure alarm output
TBCU_SMU_ALARM_NCS	Out	TBCU non-secure alarm output
SBCU_SMU_ALARM_CS	Out	SBCU security alarm output

(table continues...)

Table 623 (continued) List of FPI interface signals

Interface signals	I/O	Description
COMBCU_SMU_ALARM_CS	Out	COMBCU security alarm output
CSBCU_SMU_ALARM_CS	Out	CSBCU security alarm output
TBCU_SMU_ALARM_CS	Out	TBCU security alarm output
SBCU_INT_SRC_BCUNCS	Out	SBCU NCS service request output
COMBCU_INT_SRC_BCUNCS	Out	COMBCU NCS service request output
CSBCU_INT_SRC_BCUNCS	Out	CSBCU NCS service request output
TBCU_INT_SRC_BCUNCS	Out	TBCU NCS service request output
SBCU_INT_SRC_BCUCS	Out	SBCU CS service request output
COMBCU_INT_SRC_BCUCS	Out	COMBCU CS service request output
CSBCU_INT_SRC_BCUCS	Out	CSBCU CS service request output
TBCU_INT_SRC_BCUCS	Out	TBCU CS service request output
SBCU_IR_SRC_SBCUQOSC	Out	QOS control service request output
COMBCU_IR_SRC_COMBCUQOSC	Out	QOS control service request output
SBCU_IR_SRC_QOSCARP[7:0]	Out	QOS ARP service request outputs
COMBCU_IR_SRC_QOSCARP[7:0]	Out	QOS ARP service request output
FB_SBCU_EDC_Errors[n:0]	In	SBCU FPI EDC error signals from FPI master and slave agents connected to the SPB FPI interconnect
FB_COMBCU_EDC_Errors[n:0]	In	COMBCU FPI EDC error signals from FPI master and slave agents connected to the COMPB FPI interconnect
FB_CSBCU_EDC_Errors[n:0]	In	CSBCU FPI EDC error signals from FPI master and slave agents connected to the CSPB FPI interconnect
FB_TBCU_EDC_Errors[n:0]	In	TBCU FPI EDC error signals from FPI master and slave agents connected to the TPB FPI interconnect
FPI_SBCU_FPI_SIF	In	SBCU SW interface
FPI_COMBCU_FPI_SIF	In	COMBCU SW interface
FPI_CSBCU_FPI_SIF	In	CSBCU SW interface
FPI_TBCU_FPI_SIF	In	TBCU SW interface

7.8.6 TC4Dx FPI revision history

Initial release of the chapter.

8 LLI - NDA only

Information for this functional block is only available under NDA.

9 System Direct Memory Access (SDMA)

The System DMA is a safe and flexible DMA controller capable of mastering data move operations without direct CPU involvement.

The System DMA, more commonly referred to as just the DMA, implements multiple master interfaces to the system interconnect, and contains two move engines to allow two data movement operations to run simultaneously. It also implements multiple DMA channels which undergo priority arbitration to become active in the move engines.

9.1 Feature list

- Safe movement of data
  - The DMA implements a lockstep mechanism using a redundant checker core for safety
  - The DMA implements ECC protection on the DMARAM
- Ability to detect patterns in DMA move data
  - Pattern detection can be configured on 8-bit, 16-bit, and 32-bit data moves
- Virtualization and resource partitions to provide freedom from interference
  - The DMA implements virtualization using the concept of resource partitions (RPs). Each DMA instance implements up to 16 RPs
  - An application in one RP running a set of defined DMA move functions has functional freedom from interference from another application running another set of defined DMA move functions in a different RP
  - Each RP has independent access protection controls which enable individual Master TAG identifiers, virtual machine IDs, and protection register sets (PRS) to have read and write access to the RP and assigned DMA channels
  - Each RP may be assigned to a cyber-secure or non-cyber-secure group and only interrupt requests from the same group may trigger DMA hardware requests
  - Each RP can be configured with a master agent function tag identifier, a virtual machine number (VM), a protection register set (PRS), user or supervisor mode, and secure master identification to be driven onto the system interconnect during a DMA move
- Multiple independent DMA channels which can be individually programmed and scheduled
  - Each DMA channel may be assigned to a resource partition
  - Each DMA channel's transaction control set is stored in the DMARAM
  - The DMA channel source and destination address pointers support 32-bit wide address counters
  - Programmable data width of DMA moves
- Multiple move engines to support parallel servicing of DMA requests
  - Any move engine can service a DMA request from any DMA channel
  - Arbitration rules to pick the highest priority DMA channels with pending DMA requests for service by the move engines
  - SRI-source to SRI-destination data block moves up to 16 Mbyte DMA moves per DMA transaction
  - FPI-source to FPI-destination data block moves up to 8 Mbyte DMA moves per DMA transaction
- Multiple ways to request DMA move operations or chains of DMA move operations
  - DMA software requests
  - DMA hardware requests triggered by hardware interrupts
  - DMA daisy-chain requests to automatically trigger successive channels
  - Linked list and conditional linked list operations - the current DMA transaction can load the next DMA channel transaction control set (TCS) into the DMARAM by overwriting the existing DMA channel TCS. The next DMA transaction can be auto-started
- Generation of interrupts and alarms
  - Channel completion, pattern match and circular buffer wrap interrupts

  - Resource partition, and global error interrupts
  - Safety alarms generated on all safety events such as lockstep mismatches and EDC errors
- DMA timestamp
  - The DMA channel can be configured to write out a timestamp when the DMA transaction is completed. This timestamp can be used by application software to implement temporal monitoring of data transfers
- DMA CRC computation
  - The DMA channel can be configured to compute a CRC on the addresses or data transferred during a DMA transaction using the IEEE 802.3 CRC32 algorithm

9.2 Functional overview

The DMA moves data from source locations to destination locations without the intervention of the CPU or other on chip devices.

A data move is controlled by the transaction control set of an active DMA channel and is executed by a move engine. The transaction control sets are loaded into the DMARAM either directly through software writes through the configuration interface, or indirectly through linked list operations.

A DMA channel is activated by a DMA request, which can be a software request, a hardware request or a daisy chain request. DMA channels can be configured to raise an interrupt on transaction completion or other events. DMA channels are each assigned to a unique resource partition and inherit the access enable rules and master tag identifiers of that resource partition.

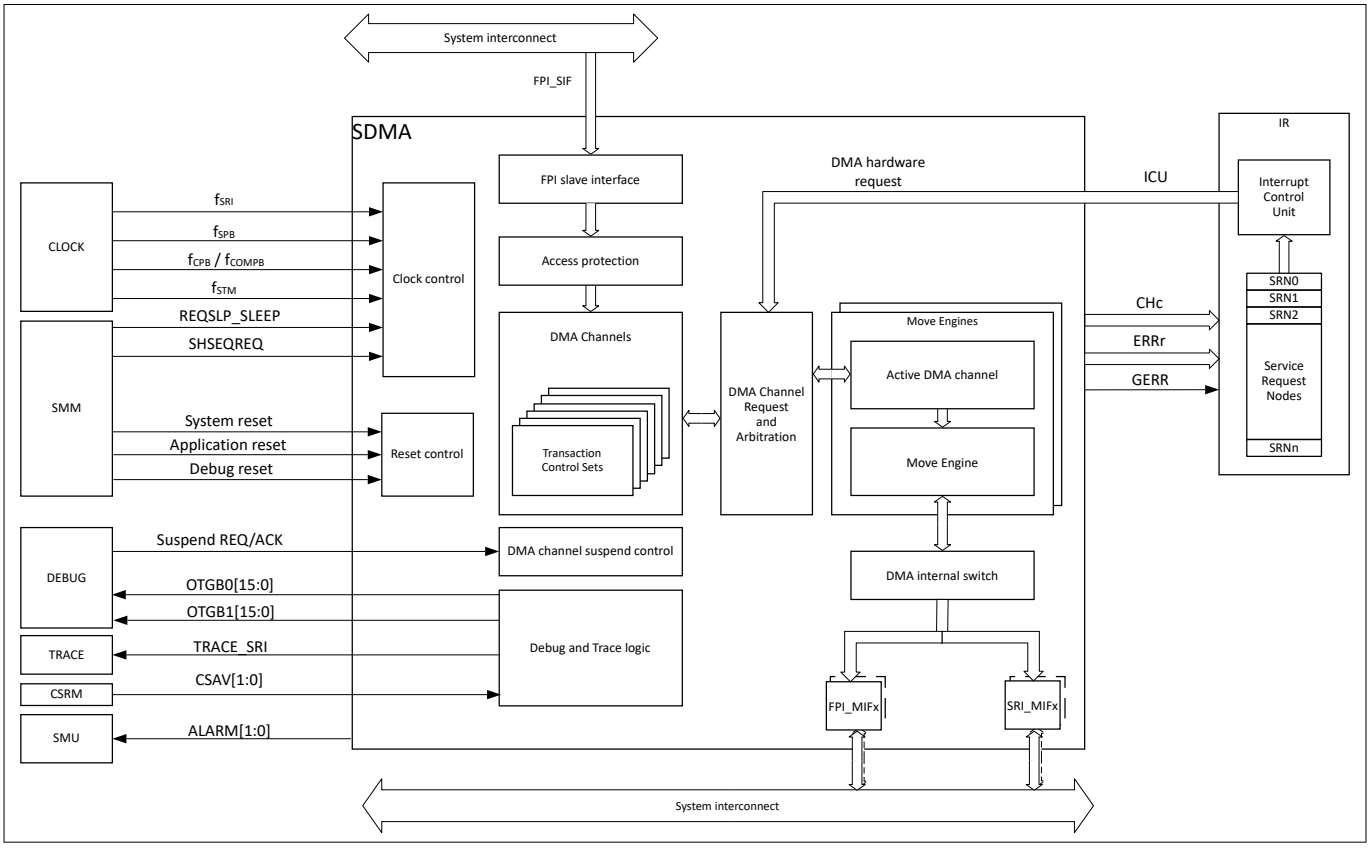


Figure 81 SDMA block diagram

The DMA is configured by software over the FPI_SIF interface and implements access protection to provide functional freedom from interference between different applications using the DMA.

Each DMA channel stores the context of an independent DMA operation in a transaction control set (TCS). Each application can configure and use one or more DMA channels. The DMA channels can be triggered by software requests, through hardware requests coming from the interrupt router (IR), or through other internal triggers. All DMA channels with pending DMA requests are then arbitrated so they can begin processing on one of the available move engines (MEs). DMA moves generated by the MEs result in read and write transactions being sent out through the DMA master agents FPI_MIFx and SRI_MIFx onto the system interconnect.

Each DMA channel can be configured to generate interrupts to the IR for completion and other events on the CHc interface. The DMA resource partitions (RPs) can also generate error interrupts to the IR on the ERRr interface. Errors which cannot be attributed to a single resource partition are reported to the IR through the GERR interface.

The DMA also implements interfaces to various infrastructure components for clocks, resets, sleep and shutdown requests. It implements debug and trace functionality through the interfaces with the DEBUG and TRACE functional blocks. The DMA debug and trace functions for CS resource partitions can be disabled by the CSRM through the CSAV interface.

The DMA also generates alarms for safety relevant events to the SMU.

For the alarm description please refer to the alarm mapping tables in the SMU functional block and the Safety Manual.

Related information

TC4Dx SMU alarm mapping tables on page 7233

9.3 Functional description

Please refer to the following tables of information to help in your understanding of the SDMA functional description.

DMA glossary

The following tables introduce some DMA acronyms and terms that are used in the rest of the chapter.

Table 624 DMA acronyms

Acronym	Description
ACCLL	Accumulated Linked List
CONLL	Conditional Linked List
CH	Channel
DER	Destination Error
DLLER	Linked List Error
DMA	Direct Memory Access
DMALL	Direct Memory Access Linked List
DMARAM	Direct Memory Access Random Access Memory
EH	Error Handler
ME	Move Engine
PATDET	Pattern Detection
RAMER	RAM Error
RDCRC	Read Data Cyclic Redundancy Check
RP	Resource Partition
RROAT	Reset Request Only After Transaction
CHKLL	Checked Linked List
SDCRC	Source Destination Cyclic Redundancy Check
SER	Source Error
CLLER	Checked Linked List Error
TCS	Transaction Control Set
TRL	Transaction Request Lost
TS	Trigger Set

Table 625 DMA terms

Term	Description
DMA configuration data	The DMA configuration data configures the DMA configuration structure.

(table continues...)

Table 625 (continued) DMA terms

Term	Description
DMA configuration structure	The DMA configuration structure is used to configure CH, RP and ME. The DMA configuration structure is defined by the DMA register address map. ***Note**: The DMA configuration structure does not include the configuration of other functional blocks such as the IR and the SMU which are needed to make the DMA function within the system.* The DMA configuration structure and DMA configuration data defines all DMA move functions.
DMA software request	The CPU initiates a DMA transfer or a DMA transaction in a DMA channel.
DMA hardware request	The Interrupt Router initiates a DMA transfer or a DMA transaction in a DMA channel.
DMA daisy chain request	As soon as the current DMA transaction completes, the DMA channel initiates a DMA transfer or a DMA transaction in the next lower priority DMA channel.
DMA auto start request	As soon as the current DMA transaction completes, a linked list operation loads a new TCS and initiates a DMA transfer or a DMA transaction.
DMA request	DMA software request, DMA hardware request, DMA daisy chain request or DMA auto start request.
DMA read move data	Data read from the source module.
DMA write move data	Data written to the destination module.
DMA channel interrupt triggers	DMA signaling of the successful completion of a DMA move function.
DMA RP error interrupt triggers	DMA signaling of a DMA move function error.
DMA alarms	DMA signaling of a DMA safety error.
DMA address checksum	Cyclic redundancy checksum calculated according to the IEEE 802.3 standard for addresses generated during a DMA transaction.
DMA data checksum	Cyclic redundancy checksum calculated according to the IEEE 802.3 standard for data moved during a DMA transaction.
DMA timestamp	Appendage of timestamp to a DMA transaction.
DMA linked list	A DMA linked list is a series of DMA transactions executed in the same DMA channel.
DMA channel	A DMA channel stores the context of each independent DMA transaction.

Table 626 Structure of data move function

Term	Description
DMA move	A DMA move is an operation that always consists of two parts: **1.** A DMA read move that loads DMA read move data from a source module to the DMA **2.** A DMA write move that stores DMA write move data from the DMA to a destination module
DMA transfer	A DMA transfer is composed of 1, 2, 3, 4, 5, 8, 9 or 16 DMA moves. Once a DMA transfer is started on a DMA Move Engine, it cannot be interrupted unless an error occurs.
DMA transaction	A DMA transaction is composed of at least one DMA transfer.

Table 627 DMA channel states

State	Comments
Idle state	No DMA request is pending in this DMA channel
Reset state	No DMA request is pending in this DMA channel. DMA channel TCS bits cleared
Halt state	The DMA channel is halted by software configuration of the DMA channel registers. A DMA request will not be serviced by a ME
Suspend state	The DMA channel is suspended by the debug logic. A DMA request will not be serviced by a ME
Pending state	A DMA request in this DMA channel is waiting to be serviced by a ME
Active state	The ME is servicing a DMA request in this DMA channel

9.3.1 Configuration interface

The DMA implements a standard FPI slave interface (FPI_SIF) compliant with the FPI bus protocol. The DMA configuration interface supports single and block data transfers.

9.3.2 DMA operating modes

Each DMA channel can be configured to operate in any of the operating modes described in DMA channel operation. Each operating mode is described in the following sections.

Related information

DMA channel operation on page 1251

9.3.2.1 Move operation

The DMA moves data from source locations to destination locations. The source and destination addresses and other characteristics of the transaction like the size are all configured in the transaction control set for the current channel.

The size of a single DMA move within a transaction is defined by the channel data width (CHCFGRc.CHDW). The number of DMA moves in a transaction may be calculated from the following:

- Block mode (CHCFGRc.BLKM) defines the number of DMA moves in a DMA transfer
- Transfer reload (CHCFGRc.TREL) defines the number of DMA transfers in a DMA transaction

After a DMA move, the next source and destination addresses are calculated. Source and destination addresses are calculated independently of each other. The following address calculation parameters may be selected:

- The address offset, which is a multiple of the selected data width
- The offset direction: Addition, subtraction, or none (unchanged address)

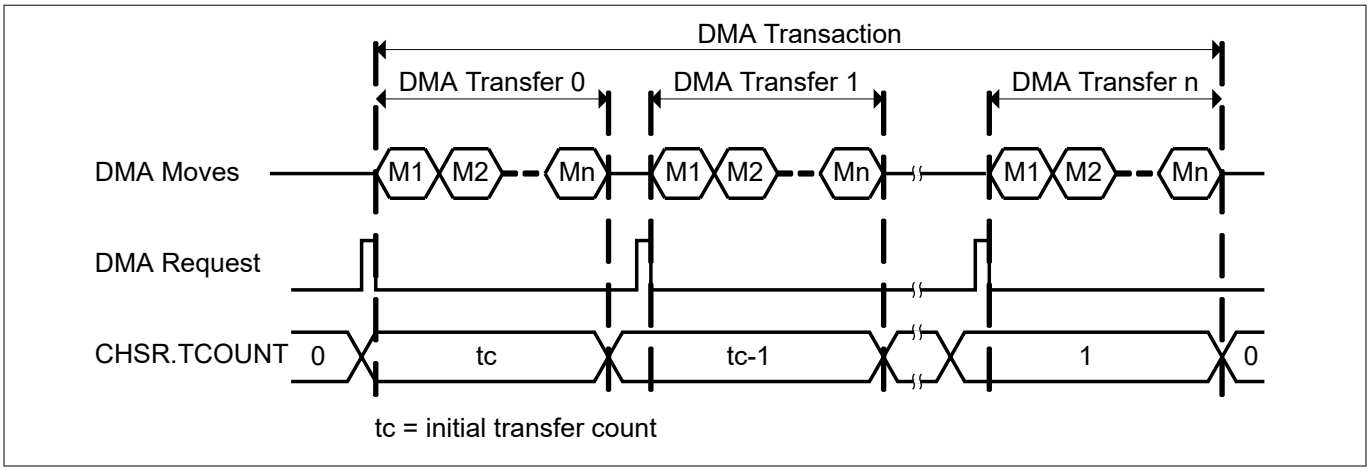


Figure 82 Block mode and transfer count

9.3.2.1.1 Address generation

Address control bits in DMA channel ADICRc determine how the addresses increment or decrement. The data width is defined in DMA channel CHCFGRc.CHDW and is taken into account during the address calculation. The address offset calculation tables below show the offset values that are added to or subtracted from the source address based on the ADICRc.SMF and ADICRc.INCS parameters, or the destination address based on the ADICRc.DMF and ADICRc.INCD parameters after each DMA move.

Attention: ***Software shall ensure that a DMA transaction is not configured such that the source or destination addresses cross from an SRI address space to an FPI address space or vice versa.***

Table 628 Address offset calculation table - 8-bit, 16-bit and 32-bit

CHCFGRc.CHDW = 000_B (8-bit data width)			**CHCFGRc.CHDW = 001_B (16-bit data width)**			**CHCFGRc.CHDW = 010_B (32-bit data width)**		
SMF DMF	**INCS INCD**	**Address offset**	**SMF DMF**	**INCS INCD**	**Address offset**	**SMF DMF**	**INCS INCD**	**Address offset**
000_B	0_B	-1	000_B	0_B	-2	000_B	0_B	-4
	1_B	+1		1_B	+2		1_B	+4
001_B	0_B	-2	001_B	0_B	-4	001_B	0_B	-8
	1_B	+2		1_B	+4		1_B	+8
010_B	0_B	-4	010_B	0_B	-8	010_B	0_B	-16
	1_B	+4		1_B	+8		1_B	+16
011_B	0_B	-8	011_B	0_B	-16	011_B	0_B	-32
	1_B	+8		1_B	+16		1_B	+32
100_B	0_B	-16	100_B	0_B	-32	100_B	0_B	-64
	1_B	+16		1_B	+32		1_B	+64
101_B	0_B	-32	101_B	0_B	-64	101_B	0_B	-128
	1_B	+32		1_B	+64		1_B	+128

(table continues...)

Table 628 (continued) Address offset calculation table - 8-bit, 16-bit and 32-bit

CHCFGRc.CHDW = 000_B (8-bit data width)			CHCFGRc.CHDW = 001_B (16-bit data width)			CHCFGRc.CHDW = 010_B (32-bit data width)		
SMF DMF	INCS INCD	Address offset	SMF DMF	INCS INCD	Address offset	SMF DMF	INCS INCD	Address offset
110_B	0_B	-64	110_B	0_B	-128	110_B	0_B	-256
	1_B	+64		1_B	+128		1_B	+256
111_B	0_B	-128	111_B	0_B	-256	111_B	0_B	-512
	1_B	+128		1_B	+256		1_B	+512

Table 629 Address offset calculation table - 64-bit, 128-bit and 256-bit

CHCFGRc.CHDW = 011_B (64-bit data width)			CHCFGRc.CHDW = 100_B (128-bit data width)			CHCFGRc.CHDW = 101_B (256-bit data width)		
SMF DMF	INCS INCD	Address offset	SMF DMF	INCS INCD	Address offset	SMF DMF	INCS INCD	Address offset
000_B	0_B	-8	000_B	0_B	-16	000_B	0_B	-32
	1_B	+8		1_B	+16		1_B	+32
001_B	0_B	-16	001_B	0_B	-32	001_B	0_B	-64
	1_B	+16		1_B	+32		1_B	+64
010_B	0_B	-32	010_B	0_B	-64	010_B	0_B	-128
	1_B	+32		1_B	+64		1_B	+128
011_B	0_B	-64	011_B	0_B	-128	011_B	0_B	-256
	1_B	+64		1_B	+128		1_B	+256
100_B	0_B	-128	100_B	0_B	-256	100_B	0_B	-512
	1_B	+128		1_B	+256		1_B	+512
101_B	0_B	-256	101_B	0_B	-512	101_B	0_B	-1024
	1_B	+256		1_B	+512		1_B	+1024
110_B	0_B	-512	110_B	0_B	-1024	110_B	0_B	-2048
	1_B	+512		1_B	+1024		1_B	+2048
111_B	0_B	-1024	111_B	0_B	-2048	111_B	0_B	-4096
	1_B	+1024		1_B	+2048		1_B	+4096

Table 630 Address offset calculation table - 512-bit

CHCFGRc.CHDW = 110_B (512-bit data width)		
SMF DMF	**INCS INCD**	**Address offset**
000_B	0_B	-64
	1_B	+64
001_B	0_B	-128
	1_B	+128
010_B	0_B	-256
	1_B	+256
011_B	0_B	-512
	1_B	+512
100_B	0_B	-1024
	1_B	+1024
101_B	0_B	-2048
	1_B	+2048
110_B	0_B	-4096
	1_B	+4096
111_B	0_B	-8192
	1_B	+8192

Address calculation examples

The following examples demonstrate address generation for a data width of 16-bit (CHCFGRc.CHDW = 001_B).

Programmable address generation - example 1

16-bit half-words are moved from a source memory with an incrementing source address offset of 10_H to a destination memory with a decrementing destination address offset of 08_H.

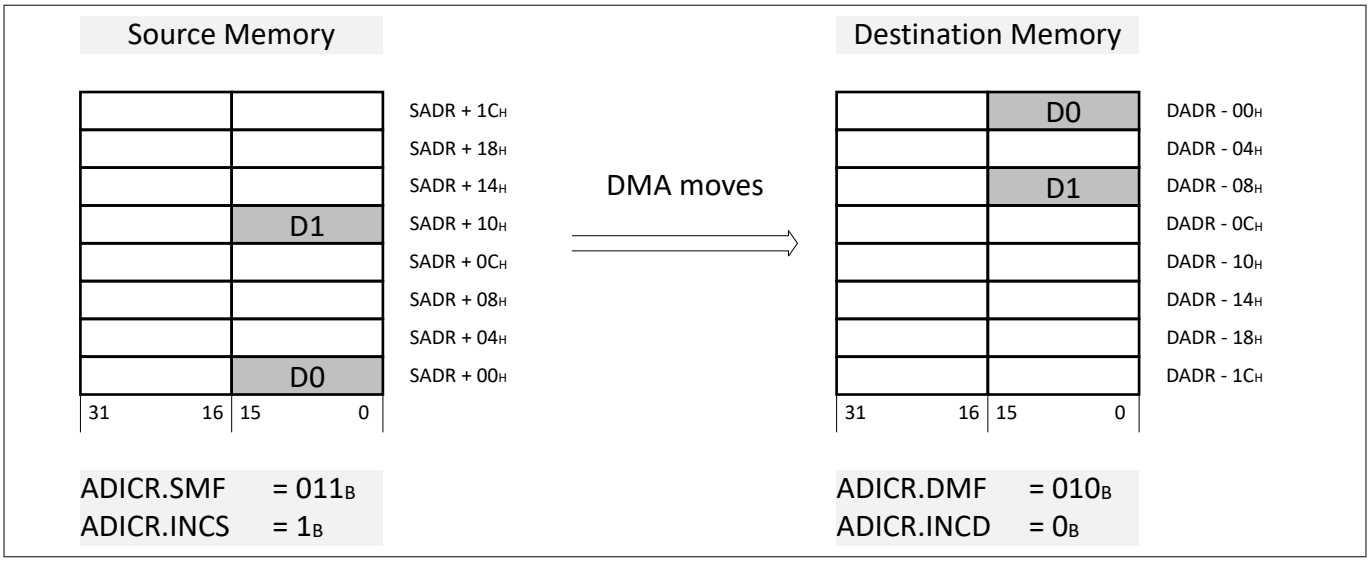


Figure 83 Programmable address generation - example 1

Programmable address generation - example 2

16-bit half-words are moved from a source memory with an incrementing source address offset of 02_H to a destination memory with an incrementing destination address offset of 04_H.

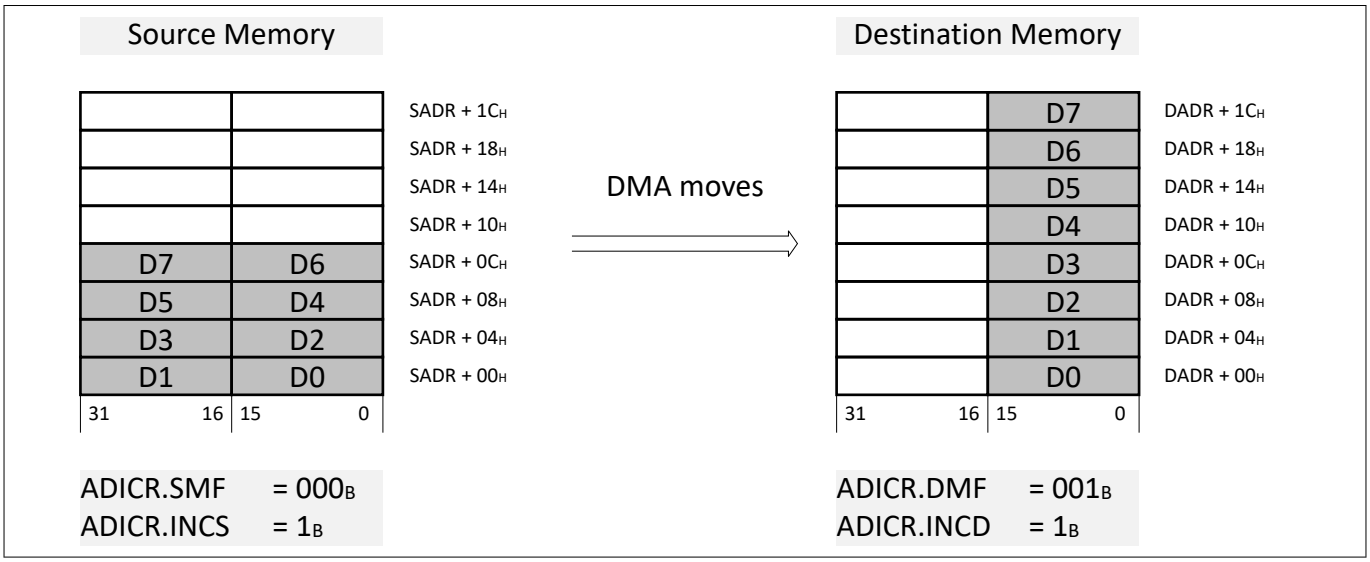


Figure 84 Programmable address generation - example 2

Circular buffer

The source and destination circular buffers are enabled by setting the DMA channel circular buffer enable bits ADICRc.SCBE and ADICRc.DCBE respectively. The source address and destination address may be configured to build a circular buffer separately for source and destination data. Within a circular buffer, the addresses are updated within the circular buffer wrap-around limits. The circular buffer length is determined by bit-fields ADICRc.CBLS for the source buffer and ADICRc.CBLD for the destination buffer. These 4-bit wide bit-fields determine which bits of the 32-bit address remain unchanged during an address update. Possible buffer sizes of the circular buffers can be 2^{CBLS} or 2^{CBLD} bytes (= 1, 2, 4, 8, 16, … up to 32k bytes). Source and destination addresses are incremented or decremented during a DMA move, all upper bits [31:CBLS] of source address and [31:CBLD] of destination address are frozen and remain unchanged, even if a wrap-around from the lower address bits [CBLS-1:0] or [CBLD-1:0] occurred. This address-freezing mechanism always causes the circular

buffers to be aligned to a multiple integer value of its size. If the circular buffer size is less than or equal to the selected address offset then the same circular buffer address will always be accessed.

Address alignment

The DMA can initiate transactions on SRI and FPI interconnects. Software must ensure that the address alignment rules for each type of interconnect are observed when configuring the address generation parameters.

FPI address alignment

The FPI interconnect interface width is 32 bits. Burst transaction are supported up to a burst length of 8, which is a 256 bit move operation. All source addresses and destination addresses shall be aligned to the correct address boundary as follows:

- Single transfers: the source address and destination address boundaries shall align with the channel data width (CHCFGRc.CHDW set to byte, half-word, word)
- Block transfers: the source address and destination address boundaries shall align to a word boundary. The FPI protocol defines a wrap around addressing scheme for block transfers not starting with a block transfer sized start address. The effect on a DMA transaction is demonstrated by different scenarios for BTR2 , BTR4 and BTR8 transactions on an FPI interconnect below

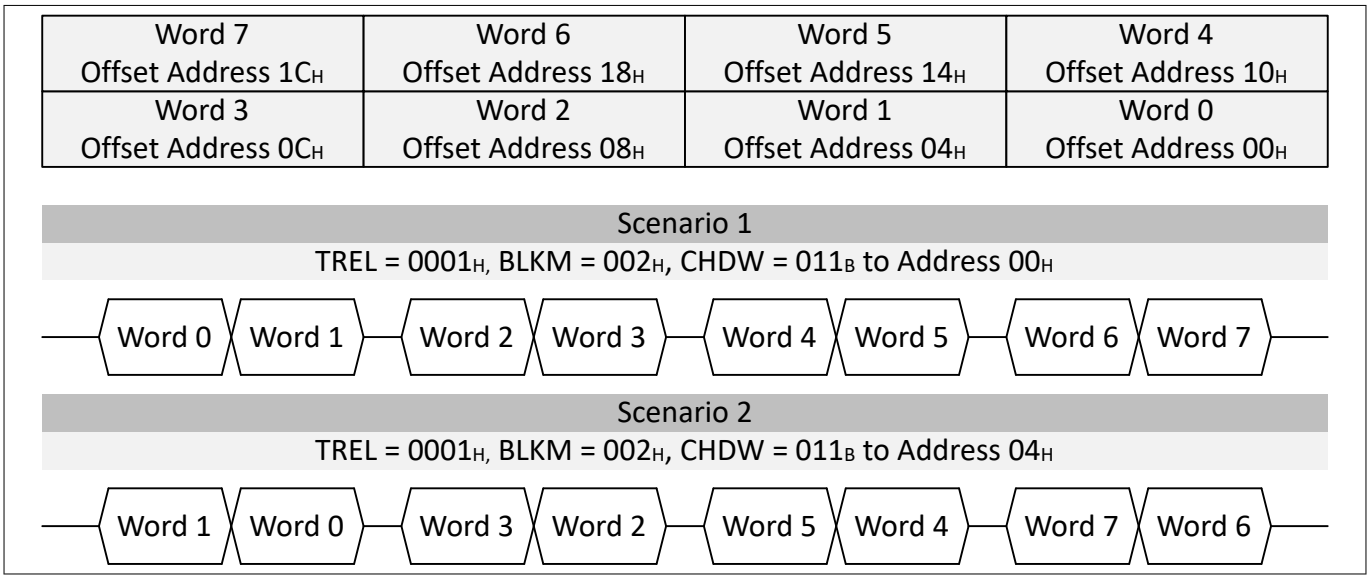


Figure 85 FPI protocol block transfer request - 2 transfers

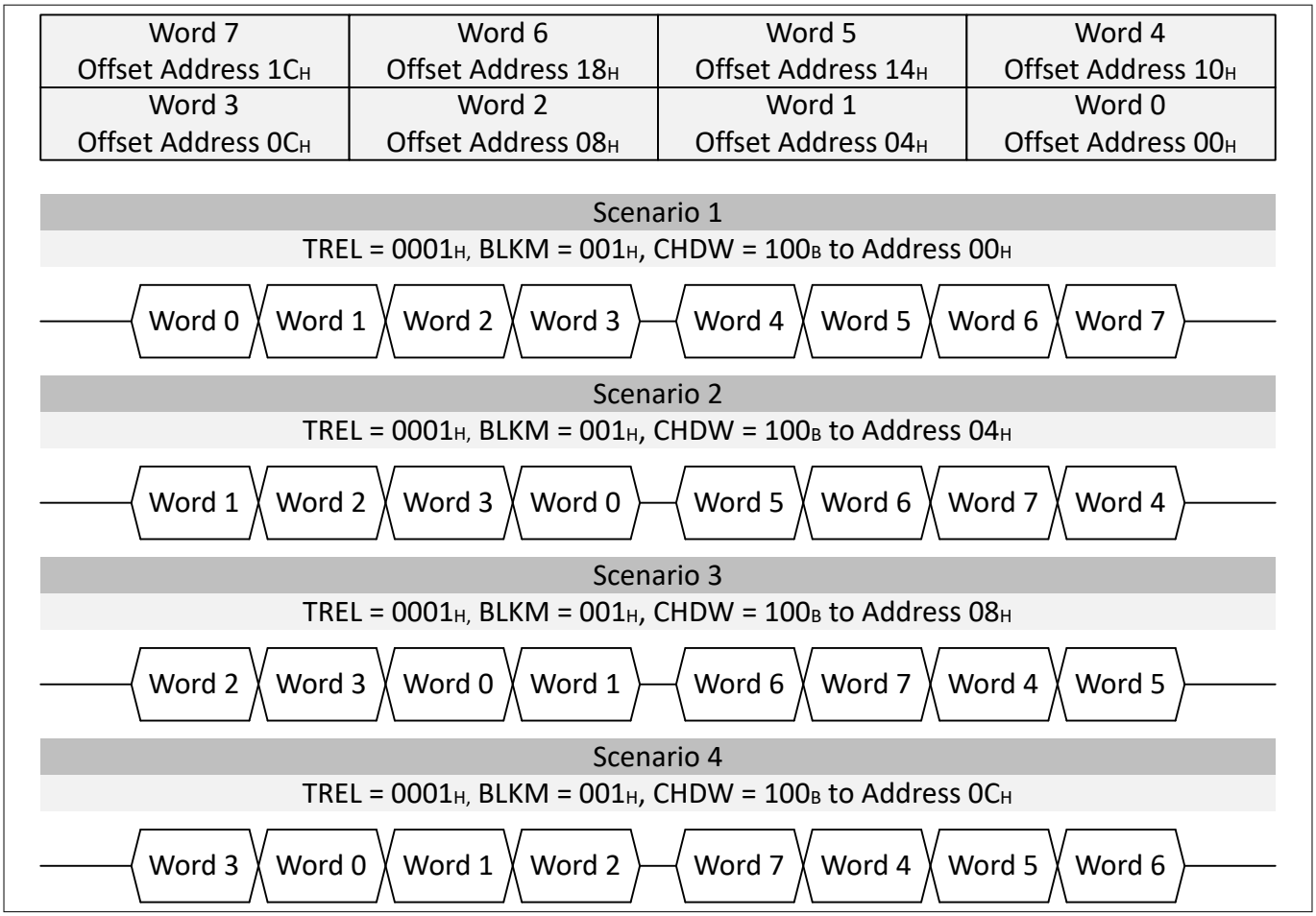


Figure 86 FPI protocol block transfer request - 4 transfers

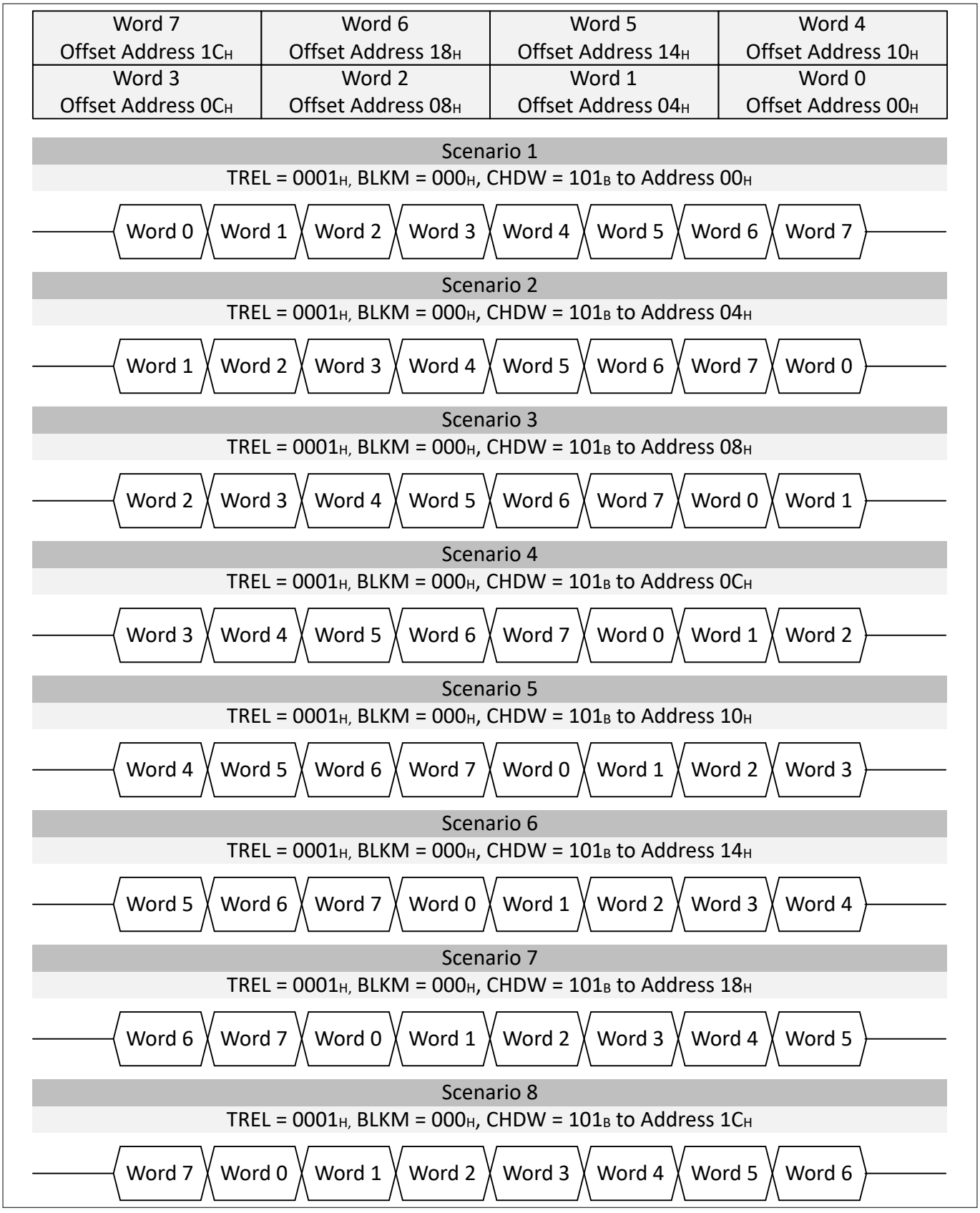


Figure 87 FPI protocol block transfer request - 8 transfers

SRI address alignment

The SRI interconnect interface width is 64 bits. Burst transaction are supported up to a burst length of 8, which is a 512 bit move operation. All source addresses and destination addresses shall be aligned to the correct address boundary as follows:

- Single transfers: The source address and destination address boundaries shall align with the channel data width (CHCFGRc.CHDW set to byte, half-word, word or double-word)
- Block transfers: The source address and destination address boundaries shall align to a double-word boundary. The SRI protocol defines a wrap around addressing scheme for block transfers not starting with a block transfer sized start address. The effect on a DMA transaction is demonstrated by different scenarios for BTR2 , BTR4 and BTR8 transactions on an SRI interconnect below

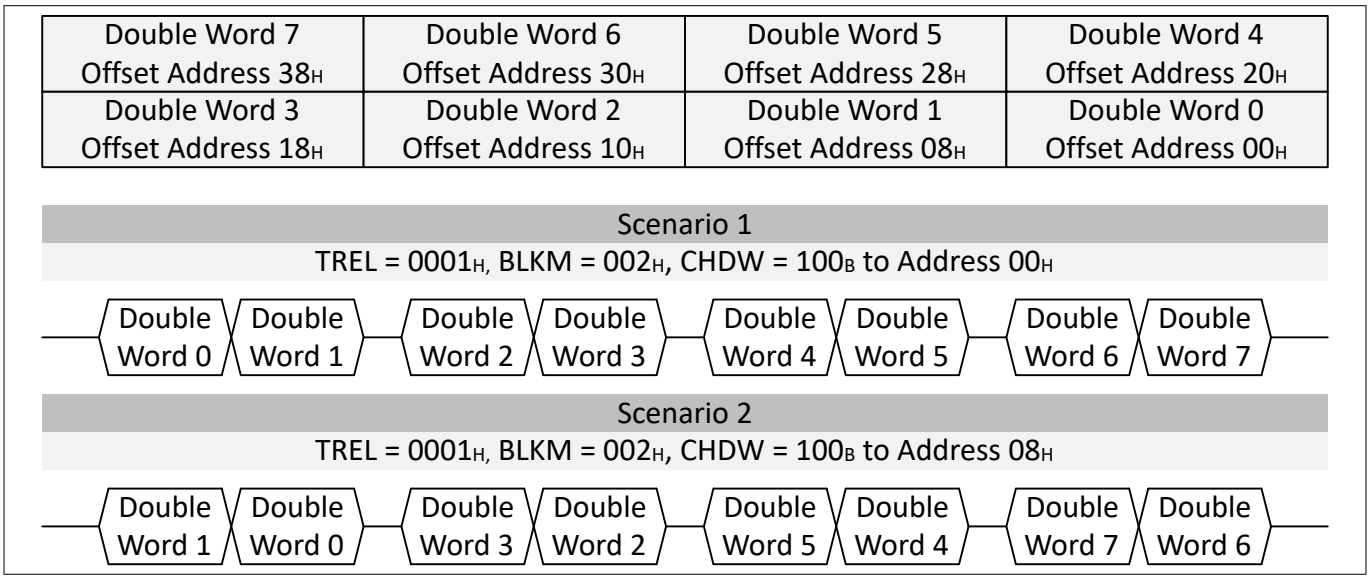


Figure 88 SRI protocol block transfer request - 2 transfers

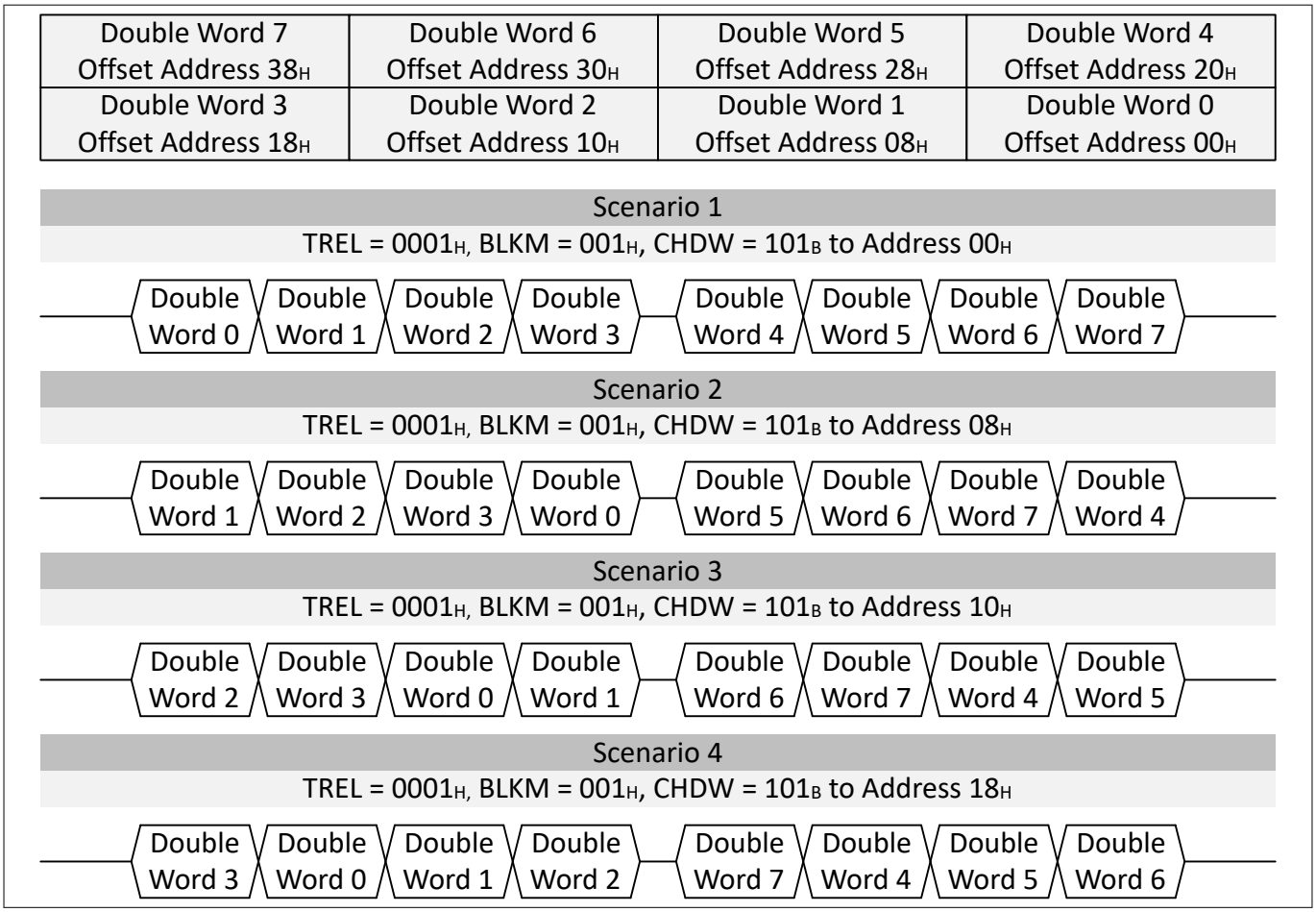


Figure 89 SRI protocol block transfer request - 4 transfers

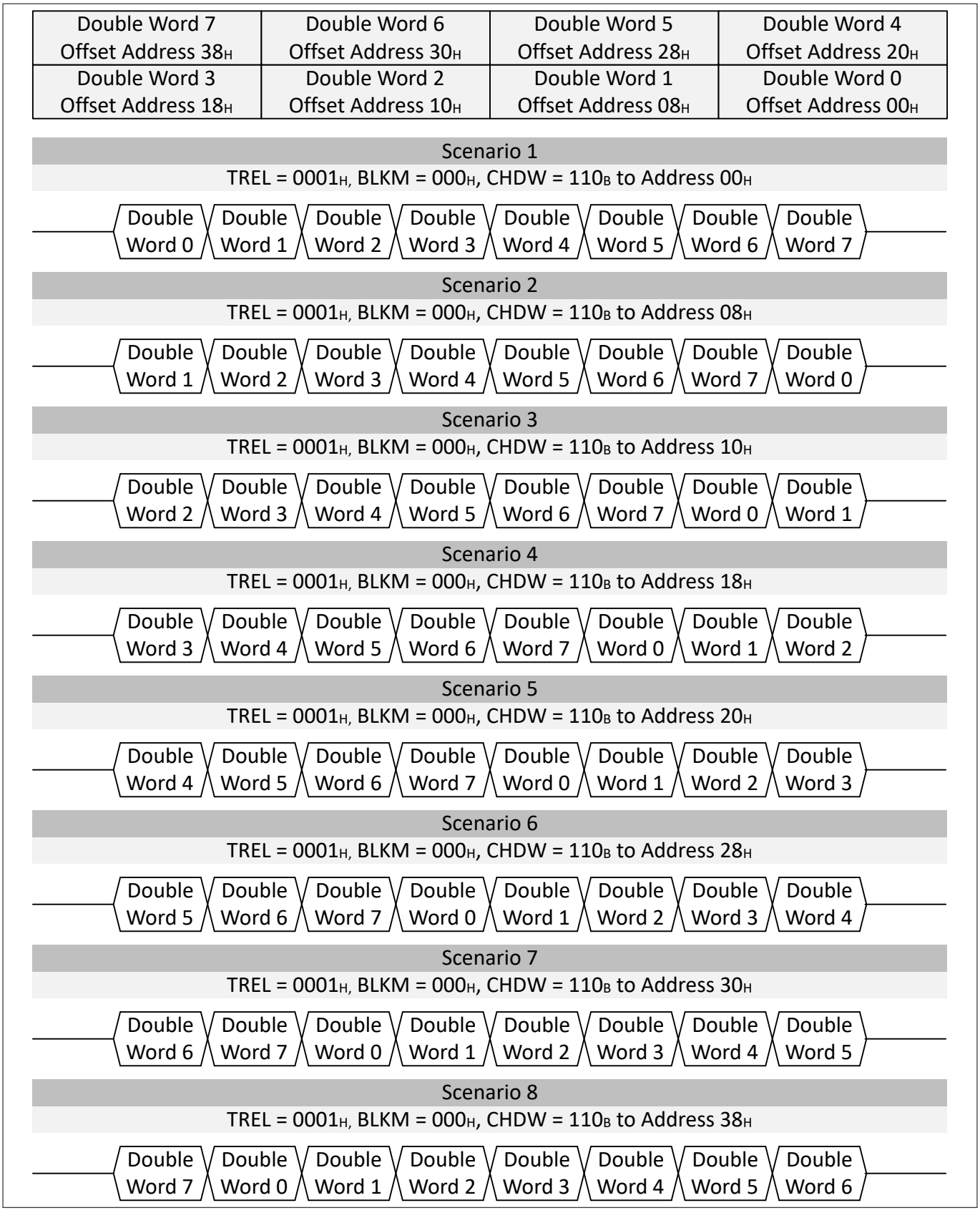


Figure 90 SRI protocol block transfer request - 8 transfers

Address counter

If the source circular buffer is not enabled (ADICRc.SCBE = 0_B) then the source address will increment or decrement across the entire 32-bit address field. If the destination circular buffer is not enabled (ADICRc.DCBE = 0_B) then the destination address will increment or decrement across the entire 32-bit address field. The address offset is determined by the DMA channel CHCFGRc.CHDW. The address will wrap around on the 32-bit address boundary:

- If the address is incrementing, it will wrap from $FFFFFFFF_H$ to 00000000_H
- If the address is decrementing, it will wrap from 00000000_H to $FFFFFFFF_H$

9.3.2.1.2 DMA address checksum

The DMA calculates a SDCRC checksum in accordance with the IEEE 802.3 standard using the source and destination addresses presented to the system interconnect as input data. Software can compare the calculated DMA address checksum (SDCRC) with an expected DMA address checksum stored in memory to validate the address generation during a DMA transaction.

Attention**: **DMA address checksum calculation is considered deprecated functionality in the AURIX™ TC4xx family. This functionality may be removed in a future AURIX™ generation.

To check the DMA channel address generation use the following steps:

- Software shall initialize the DMA address checksum stored in DMARAM to a known initial value
- The DMA channel receives a DMA request
- If the DMA channel is in the active state and no error or retry event is reported, the ME calculates an updated DMA address checksum for each DMA read move and DMA write move
- On completion of the DMA transaction, the DMA shall store the final DMA address checksum in DMARAM
- Software shall compare the calculated DMA address checksum
  - If the calculated and expected DMA address checksum match, the DMA has correctly calculated the source and destination address. If the checksums do not match, software shall report a fault

The DMA address checksum is not available for the following configurations of DMA channel operation:

- DMA double source buffering Software switch only
- DMA double source buffering Software switch and Automatic hardware switch
- DMA double destination buffering Software switch only
- DMA double destination buffering Software switch and Automatic hardware switch
- Conditional Linked List (CONLL)

Related information

DMA channel states on page 1262
DMA channel operation on page 1251
DMA double source buffering on page 1234
Software switch on page 1236
Automatic hardware switch on page 1236
Conditional linked list (CONLL) on page 1246
DMA double destination buffering on page 1234

9.3.2.1.3 DMA data checksum

To support enhanced data integrity checking the DMA calculates a 32-bit read data CRC (RDCRCRc) in accordance with the IEEE 802.3 standard on DMA move data as it passes through the DMA.

Move operation or shadow operation

To calculate a DMA data checksum for DMA move data the DMA channel RDCRCRc must be initialized, for example written with 00000000_H or another desired initial value, in addition to the standard DMA transaction configuration set for the size of DMA move data.

If no error or retry event is reported during DMA move then the ME calculates an updated DMA data checksum value for each DMA move. On completion of the DMA transaction the ME stores the DMA data checksum in the DMA channel RDCRCRc. Software may compare the calculated DMA data checksum with an expected DMA data checksum to verify the integrity of DMA move data.

Swap byte

The DMA data checksum is calculated for all DMA channel data widths and may be configured to swap the byte order (at the following figure). If DMA channel swap byte is selected (CHCFGRc.SWAP = 1_B) then the order of bytes are swapped before DMA data checksum computation. Big-endian input is converted to little-endian and vice versa. If the DMA channel is configured for 8-bit channel data width (CHCFGRc.CHDW = 000_B) then swap byte has no effect.

Swap byte does not change the byte order of data between a DMA read move and a DMA write move.

DMA timestamp

If the DMA channel is configured to append a DMA timestamp then the DMA timestamp is not part of the DMA data checksum calculation.

Double buffering operations

The DMA data checksum is accumulated across all the DMA move data moved into both buffers. Software may re-initialize the DMA data checksum after a Software switch and before the next data sample is received.

DMA linked List (DMALL)

The DMA data checksum is initialized to 00000000_H at the start of each DMA transaction.

Accumulated linked list (ACCLL), checked linked list (CHKLL) and conditional linked list (CONLL)

The DMA data checksum is accumulated across all the DMA transactions in the linked list.

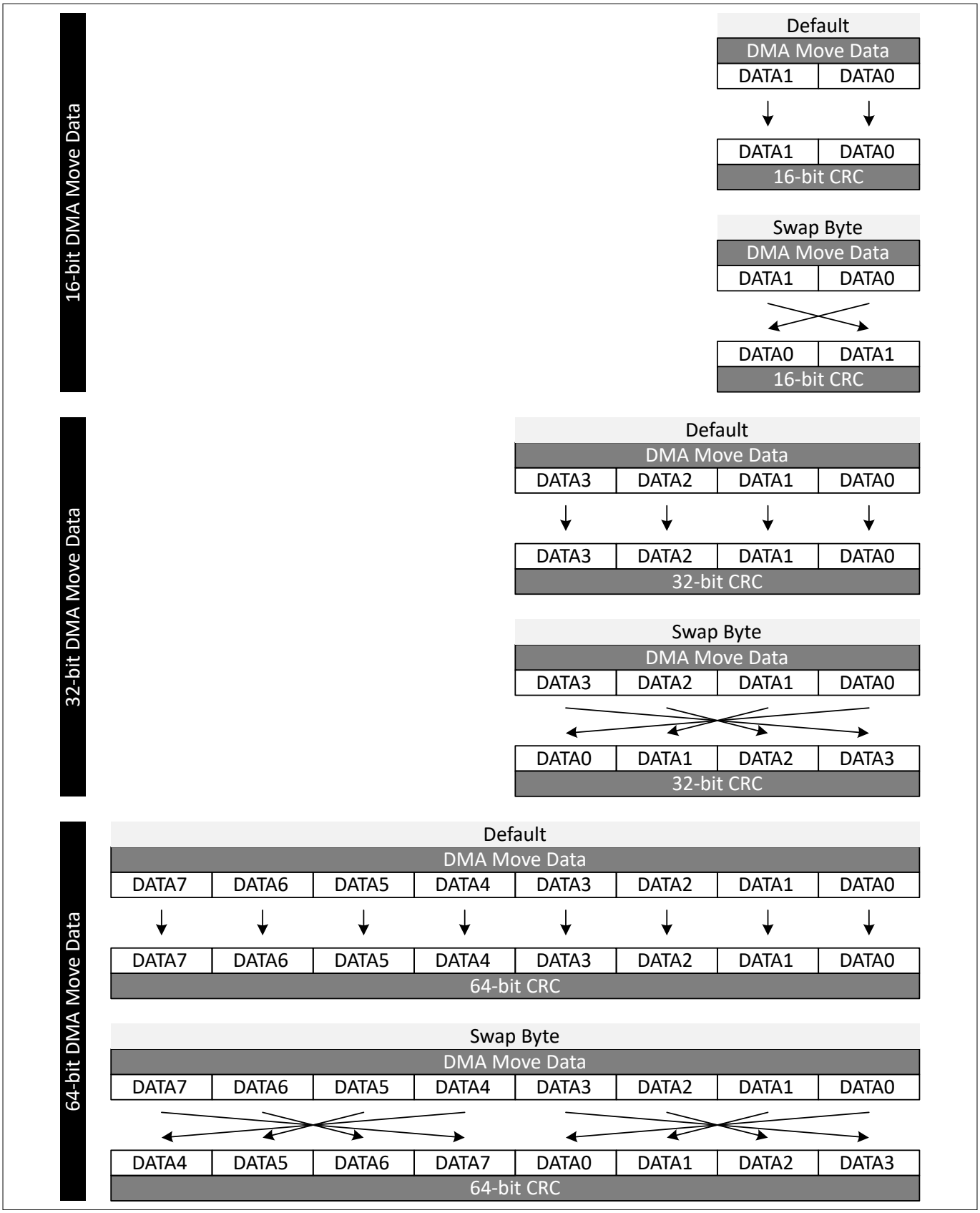


Figure 91 DMA data checksum - swap byte order

Related information

Move operation on page 1211

Shadow operation on page 1230
DMA timestamp on page 1248
Double buffering operations on page 1233
DMA linked list (DMALL) on page 1241
Accumulated linked list (ACCLL) on page 1243
Checked linked list (CHKLL) on page 1243
Conditional linked list (CONLL) on page 1246
Software switch on page 1236

9.3.2.2 Pattern detection

Each DMA resource partition (RP) provides two pattern detection registers that can be used by the DMA channels assigned to the RP. If a DMA channel is configured for pattern detection then after each DMA read move, the DMA move engine (ME) compares move data with a value stored in the Pattern Read Registers (PRR) belonging to the corresponding resource partition. The pattern detection operation depends on the channel data width (CHCFGRc.CHDW). The control field CHCFGRc.PATSEL selects different pattern detection operations for a specific value of CHDW.

The pattern compare logic compares the DMA move data stored in the ME read buffer to the selected PRR register. If the bytes of move data to be compared are spread across different words of move data, the ME may combine the pattern match result of the current DMA move and the pattern match result stored in MEmCHSR.LXO of the previous DMA move.

Configuration

If a DMA channel is configured for pattern detection:

- The channel data width (CHCFGRc.CHDW) shall be configured to 8-bit, 16-bit or 32-bit
- The source address shall not be configured to a cached address (segments 8 and 9). A non-cached address (segments A and B) shall be configured to prevent the ME read buffer from re-aligning the move data

Interrupt service request

If a DMA channel is configured for pattern detection and a pattern match is detected:

- The DMA transaction ends with the current DMA move
- DMA channel TSRc.HTRE is cleared to stop DMA transfers in the current active DMA channel
- DMA channel TSRc.CH will clear when the DMA write move has completed
- The ME writes back the next DMA move values of SADRc and DADRc
  - If the DMA channel is configured for Shadow operation Read only mode, the ME does not perform a shadow address update. The ME writes backs the current value of SHADRc into the DMARAM
- A pattern detection interrupt service request is triggered

Software may read the DMA channel status to identify the location of the pattern match in the data stream

Errors

If a SER is reported during the current DMA read move, the ME does not perform a pattern match

Related information

ME read buffer on page 1280
Shadow operation on page 1230
Read only mode on page 1231

9.3.2.2.1 Application of pattern detection

In a typical application, there is an incoming data stream from one of the communication interfaces which the application has to scan for a fixed pattern which might indicate, for example, the end of the communication. The application can then offload this task to the DMA by configuring a DMA channel in pattern detection mode. If the DMA detects the configured pattern in the data stream, the DMA channel activates the DMA channel pattern match interrupt service request. The application may then read the SADRc register to determine the address at which the pattern match occurred after allowing for the increment or decrement of the source address after the matching DMA move completes.

9.3.2.2.2 Pattern compare logic

The ME COMPare (COMP) logic shown below compares move data against a reference pattern. One COMP block controls either 8 bits or 16 bits of data. The COMP block may be configured to mask each data bit for the pattern compare.

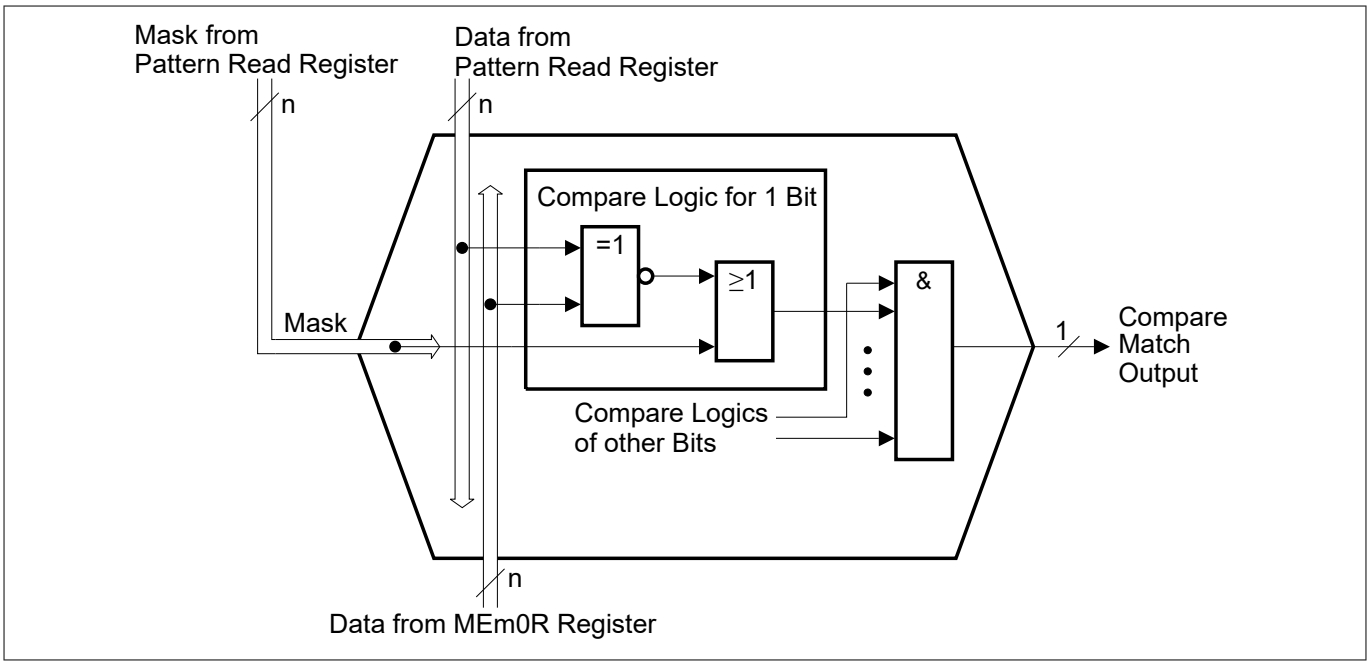


Figure 92 Pattern compare logic (COMP block)

In the COMP logic, one data bit from a ME read register is compared to the corresponding pattern bit stored in the selected PRR. If both bits are equal and a pattern mask bit stored in another PRR bit-field is 0_B then the compare matched condition becomes active. When the pattern mask bit is set to 1_B, the compare matched condition is always active (set) for the related bit. When the compare matched conditions for each bit within the COMP logic are true, the compare match output line of the COMP block becomes active.

9.3.2.2.3 Pattern detection for 8-bit channel data width

If a DMA channel is configured for Pattern detection for 8-bit channel data width, the ME uses the source address to select byte(s) of move data to be compared with patterns stored in PRRAr or PRRBr.

The pattern compare logic allows a byte of DMA move data to be compared with two different patterns. A mask operation of each compared bit is possible.

Example of pattern detection for 8-bit channel data width

The DMA channel is configured for pattern detection for 8-bit channel data width (CHCFGRc.CHDW = 000_B).
If the current DMA read move stores the move data byte in the lowest byte of the ME read buffer:

- The ME uses the source address to select the move data byte stored in the lowest byte of the ME read buffer
- The ME compares move data byte stored in the lowest byte of the ME read buffer to PRRAr

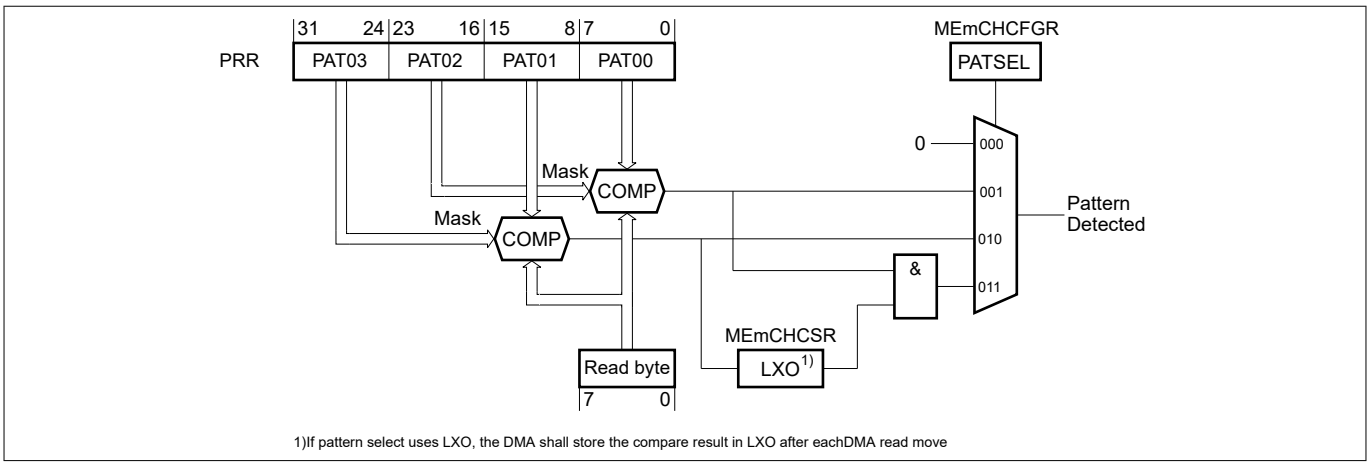


Figure 93 Pattern detection for 8-bit channel data width

The DMA channel may be configured for one of three pattern compare operations.

Table 631 Pattern detection for 8-bit channel data width

PATSEL	Pattern detection operating modes
000_B	Pattern detection disabled
001_B	Pattern compare of the lowest byte of the DMA read move data to PRRAr.PAT00, masked by PRRAr.PAT02
010_B	Pattern compare of the lowest byte of the DMA read move data to PRRAr.PAT01, masked by PRRAr.PAT03
011_B	Pattern compare of the lowest byte of the current DMA read move data to PRRAr.PAT00, masked by PRRAr.PAT02 and Pattern compare of the lowest byte of the previous DMA read move data to PRRAr.PAT01, masked by PRRAr.PAT03
100_B	Pattern detection disabled
101_B	Pattern compare of the lowest byte of the DMA read move data to PRRBr.PAT10, masked by PRRBr.PAT12
110_B	Pattern compare of the lowest byte of the DMA read move data to PRRBr.PAT11, masked by PRRBr.PAT13
111_B	Pattern compare of the lowest byte of the current DMA read move data to PRRBr.PAT10, masked by PRRBr.PAT12 and Pattern compare of the lowest byte of the previous DMA read move data to PRRBr.PAT11, masked by PRRBr.PAT13

Example of two byte pattern detection sequence

If a DMA channel is configured for a two byte pattern detection sequence compared to PRRAr (CHCFGRc.PATSEL[2:0] = 011_B) then after each DMA move the pattern match result of move data compared with PRRAr.PAT01, masked by PRRAr.PAT03 is stored in bit MEmCHSR.LXO.

This DMA channel configuration allows, for example, two-byte sequences to be detected in an 8-bit data stream coming from a serial module unit with 8-bit data width - for example recognition of carriage-return and line-feed characters.

9.3.2.2.4 Pattern detection for 16-bit channel data width

If a DMA channel is configured for Pattern detection for 16-bit channel data width, the ME uses the source address to select bytes of move data to be compared with patterns stored in PRRAr or PRRBr. Three pattern match configurations are possible defined by the DMA channel pattern select (CHCFGRc.PATSEL[1:0]):

- Aligned mode: compare complete half-word of one DMA move only
- Unaligned mode: compare upper and lower byte of two consecutive DMA moves
- Combined mode: combine Aligned mode and Unaligned mode

A mask operation of each compared bit is possible.

Example of pattern detection for 16-bit channel data width

The DMA channel is configured for pattern detection for 16-bit channel data width (CHCFGRc.CHDW = 001_B).

If the current DMA read move stores the move data half word in the lowest half word of the ME read buffer:

- The ME uses the source address to select the move data half word stored in the lowest half word of the ME read buffer
- The ME compares move data half word stored in the lowest half word of the ME read buffer to PRRAr

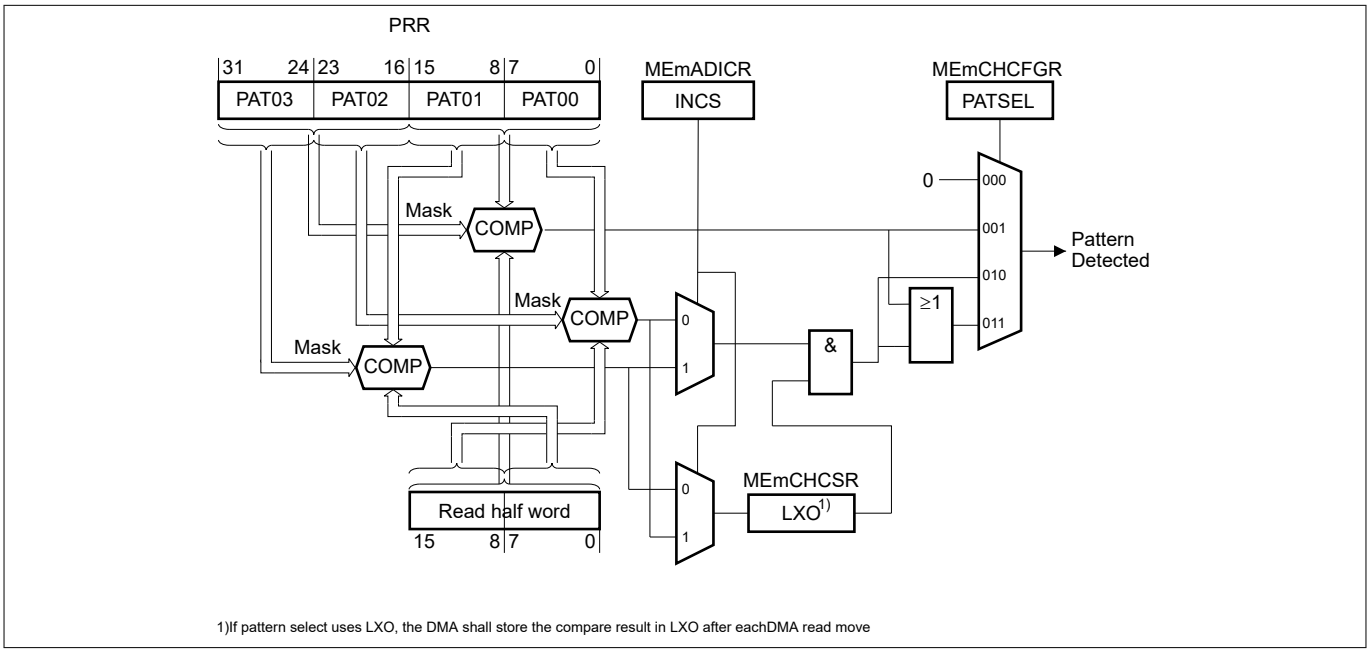


Figure 94 Pattern detection for 16-bit channel data width

The DMA channel may be configured for one of four pattern compare operations.

- Aligned mode
- Unaligned mode 1 (source address decrement)
  - The high byte of the current 16-bit DMA read move data and the low byte of the previous 16-bit DMA read move data are compared
- Unaligned mode 2 (source address increment)
  - The low byte of the current 16-bit DMA read move data and the high byte of the previous 16-bit DMA read move data are compared
- Combined mode
  - If it is not known on which byte boundary (even or odd address) the 16-bit pattern to be detected is located, the DMA channel should be configured for combined mode
  - The combined mode is the most flexible mode that combines the pattern search capability for aligned and unaligned 16-bit data searches

Table 632 Pattern detection for 16-bit channel data width

PATSEL	INCS	Pattern detection operating modes
000_B	–	Pattern detection disabled
001_B	–	Aligned mode Pattern compare of the lowest half word of the DMA read move data to PRRAr.PAT01||PAT00, masked by PRRAr.PAT03||PAT02
010_B	0	Unaligned mode 1 (source address decrement) Pattern compare of the second lowest byte of the current DMA read move data to PRRAr.PAT00, masked by PRRAr.PAT02 and Pattern compare of the lowest byte of the previous DMA read move data to PRRAr.PAT01, masked by PRRAr.PAT03 (LXO)
	1	Unaligned mode 2 (source address increment) Pattern compare of the lowest byte of the current DMA read move data to PRRAr.PAT01, masked by PRRAr.PAT03 and Pattern compare of the second lowest byte of the previous DMA read move data to PRRAr.PAT00, masked by PRRAr.PAT02 (LXO)
011_B	0 or 1	Combined mode Pattern compare for aligned mode (PATSEL = 001_B) or unaligned modes (PATSEL = 010_B)
100_B	–	Pattern detection disabled
101_B	–	Aligned mode Pattern compare of the lowest half word of the DMA read move data to PRRBr.PAT11||PAT10, masked by PRRBr.PAT13||PAT12
110_B	0	Unaligned mode 1 (source address decrement) Pattern compare of the second lowest byte of the current DMA read move data to PRRBr.PAT10, masked by PRRBr.PAT12 and Pattern compare of the lowest byte of the previous DMA read move data to PRRBr.PAT11, masked by PRRBr.PAT13 (LXO)
	1	Unaligned mode 2 (source address increment) Pattern compare of the lowest byte of the current DMA read move data to PRRBr.PAT11, masked by PRRBr.PAT13 and Pattern compare of the second lowest byte of the previous DMA read move data to PRRBr.PAT10, masked by PRRBr.PAT12 (LXO)
111_B	0 or 1	Combined mode Pattern compare for aligned mode (PATSEL = 101_B) or unaligned modes (PATSEL = 110_B)

9.3.2.2.5 Pattern detection for 32-bit channel data width

If a DMA channel is configured for Pattern detection for 32-bit channel data width, the ME compares a move data word with patterns stored in PRRAr or PRRBr. Three pattern compare configurations are possible defined by the DMA channel pattern select (CHCFGRc.PATSEL[1:0]). A mask operation is not possible.

Example of pattern detection for 32-bit channel data width

The DMA channel is configured for pattern detection and 32-bit channel data width (CHCFGRc.CHDW = 010_B).
If the current DMA read move stores the move data word in the lowest word then

- The ME uses the source address to select the move data word stored in the lowest word of the ME read buffer
- The ME compares move data word stored in the lowest word of the ME read buffer to PRRAr

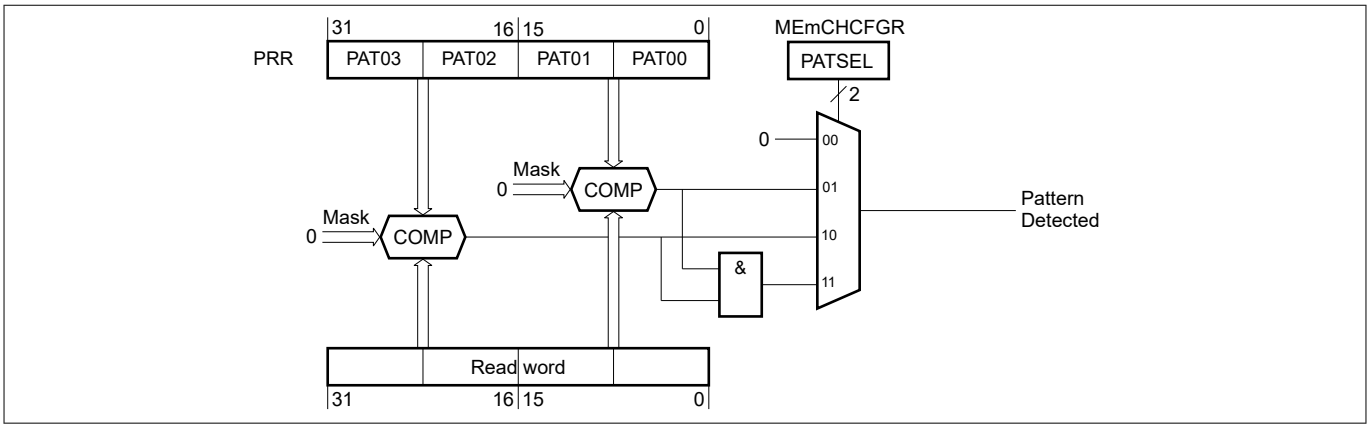


Figure 95 Pattern detection for 32-bit channel data width

The DMA channel may be configured for one of three pattern compare operations.

- Lower half-word only compared to pattern stored in PRRAr or PRRBr
- Upper half-word only compared to pattern stored in PRRAr or PRRBr
- Complete 32-bit word compared to pattern stored in PRRAr or PRRBr

Table 633 Pattern detection for 32-bit channel data width

PATSEL	Pattern detection operating modes
000_B	Pattern detection disabled
001_B	Unmasked pattern compare of lowest half word of the DMA read move data to PRRAr.PAT01||PAT00
010_B	Unmasked pattern compare of second lowest half word of the DMA read move data to PRRAr.PAT03||PAT02
011_B	Unmasked pattern compare of lowest word of the DMA read move data to PRRAr.PAT03||PAT02||PAT01||PAT00
100_B	Pattern detection disabled
101_B	Unmasked pattern compare of lowest half word of the DMA read move data to PRRBr.PAT11||PAT10
110_B	Unmasked pattern compare of second lowest half word of the DMA read move data to PRRBr.PAT13||PAT12
111_B	Unmasked pattern compare of lowest word of the DMA read move data to PRRBr.PAT13||PAT12||PAT11||PAT10

9.3.2.3 Shadow operation

If a DMA channel is configured for shadow operation, the DMA buffers a variable source or destination address.

9.3.2.3.1 Application of shadow operation

In a typical application, an ASCLIN module that receives data (fixed DMA channel source address) has to deliver the data to a memory buffer using a DMA transaction (variable DMA channel destination address). After a certain amount of data has been transferred, a new DMA transaction should be initiated to deliver further ASC data into another memory buffer. While the destination address register is updated during a running DMA transaction with the actual destination address, a shadow mechanism allows programming of a new destination address without disturbing the content of the destination address register. In this case, the new

destination address is written and buffered in the shadow address register. At the start of the next DMA transaction, the destination address register uses the new address without CPU intervention.

The shadow operation enables the application to immediately set up the next transaction after the current one has started, and without having to wait for the current one to complete. This can give the software a larger time window to configure the next transaction and therefore enable better application performance.

9.3.2.3.2 Shadowed address register

The shadow address register SHADRc stores either a new source address, or a new destination address. If both the source address and destination address have to be updated for the next DMA transaction, a running DMA transaction for this DMA channel must be finished. After that, source and destination address registers may be written before the next DMA transaction is started.

Only one address register may be shadowed while a DMA transaction is running, because the shadow register can only be assigned either to the source address or to the destination address. Note that the shadow address transfer mechanism has the same behavior in Single Mode and Continuous Mode.

The function of the shadow address register is defined by the mode of the shadow operation:

- Read only mode: If software writes to the shadow address register, the DMA does not execute the write and store a RP TCS configuration error
- Direct write mode: If software writes to the shadow address register, the DMA stores a new shadow address in the shadow address register

Related information

DMA hardware request on page 1257

TCS configuration error interrupt service request on page 1279

9.3.2.3.3 Read only mode

If software writes a new DMA channel source or destination address value, the DMA stores the new address value as follows:

- Idle state (with DMA channel CHCSRc.TCOUNT=0)
  - If software writes a new DMA channel source address value, the DMA stores the new source address value in the DMA channel SADRc
  - If software writes a new DMA channel destination address value, the DMA stores the new destination address value in the DMA channel DADRc
- Idle state (with DMA channel CHCSRc.TCOUNT!=0) or pending state
  - If software writes a new DMA channel source or destination address value, the DMA stores the new address value in the DMA channel SHADRc
- Active state
  - If software writes a new DMA channel source or destination address value, the DMA stores the new address value in the MEmSHADR

End of DMA transaction

As soon as the DMA completes a DMA transaction, the DMA checks the value of MEmSHADR.

If ME SHADR is not equal to 0000 0000_H, the ME writes back data to the DMARAM as follows:

- Shadow operation read-only mode source address
  - The DMA stores the MEmSHADR value into DMA channel SADRc
  - The DMA stores 0000 0000$_H$ into DMA channel SHADRc
- Shadow operation read-only mode destination address
  - The DMA stores the MEmSHADR value into DMA channel DADRc
  - The DMA stores 0000 0000$_H$ into DMA channel SHADRc

If software has not executed a shadow address update, MEmSHADR is equal to 0000 0000$_H$. The ME writes back data to the DMARAM as follows:

- Shadow operation read only mode source address
  - The DMA stores 0000 0000$_H$ into DMA channel SADRc (see Error conditions)
- Shadow operation read only mode destination address
  - The DMA stores 0000 0000$_H$ into DMA channel DADRc (see Error conditions)

Related information

DMA channel states on page 1262

Error conditions on page 1232

9.3.2.3.4 Direct write mode

If software writes a new DMA channel shadow address value, the DMA stores the new shadow address value as follows:

- Idle state or Pending state:
  - If software writes a new DMA channel shadow address value, the DMA stores the new shadow address value in the DMA channel SHADRc
- Active state:
  - If software writes a new DMA channel shadow address value, the DMA stores the new shadow address value in the MEmSHADR. As soon as the ME writes back data to the DMA RAM, the DMA stores the MEmSHADR value in the DMA channel SHADRc word

Start of DMA transaction

As soon as the DMA services a DMA request, the DMA performs a shadow address transfer as follows:

- Shadow operation direct write mode source address:
  - If DMA channel SHADRc is not 0000 0000$_H$, the DMA stores the DMA channel SHADRc value into MEmSADR
- Shadow operation direct write mode destination address:
  - If DMA channel SHADRc is not 0000 0000$_H$, the DMA stores the DMA channel SHADRc value into MEmDADR

End of shadow operation

If software writes a new DMA channel shadow address value of 0000 0000$_H$ or applies a DMA channel reset, the DMA shall not perform a shadow address update at the start of the next DMA transaction.

9.3.2.3.5 Error conditions

If a DMA channel is configured for Shadow operation and a new DMA request is serviced before software updates the shadow address, the DMA performs a DMA move to address 0000 0000$_H$ resulting in a bus error.

Related information

Shadow operation on page 1230

9.3.2.3.6 Transfer count update

The transfer count of a DMA transaction, stored in the DMA channel bit-field CHCFGRc.TREL, may be programmed if a DMA transaction is running. At the start of a DMA transaction, CHCFGRc.TREL is transferred to the MEmCHSR.TCOUNT. No reload of address or counter will be done if MEmCHSR.TCOUNT is not equal to 0.

The reprogramming of channel specific values except for the selected shadow address register must be avoided while a DMA channel is active as the data transfer may be corrupted.

Transfer count and source address update - example

The contents of the ME transfer count and the ME source address register are updated as shown in the figure below during two DMA transactions with a transfer count and a shadowed source address update.

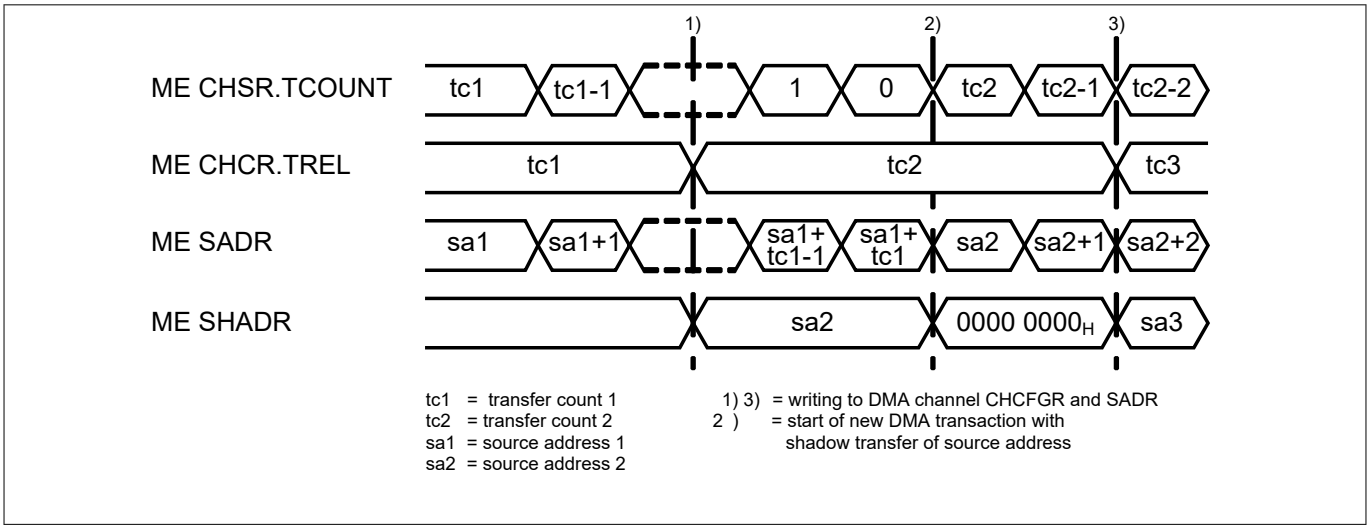


Figure 96 Shadow operation read only mode source address: Transfer count and source address update

At reference point 2) the DMA transaction 1 is finished and DMA transaction 2 is started. At 1) the DMA channel is reprogrammed with two new parameters for the next DMA transaction: Transfer count tc2 and source address sa2. Source address sa2 is buffered in MEmSHADR and transferred to MEmSADR when the new DMA transaction is started at 2). At this time, transfer count tc2 is also transferred to MEmCHSR.TCOUNT. Note that the shadow address register is only reset by hardware to 0000 0000_H as shown in this example, if read-only mode is selected.

In the event that DMA channel CHCFGRc.TREL is written while DMA channel is active:

- A write to DMA channel CHCFGRc.TREL will update the MEmCHCR.TREL value
- On write back DMA channel CHCFGRc.TREL is updated with the latest MEmCHCR.TREL value
- MEmCHSR.TCOUNT is updated to the new DMA channel CHCFGRc.TREL value at the start of the next DMA transaction

9.3.2.4 Double buffering operations

The DMA supports double buffering, which can be used for tasks which are schedule driven rather than being interrupt driven. For example, double destination buffering is as follows:

- DMA read moves transfer a continuous data stream from a module to the DMA
- DMA write moves transfer the read data from the DMA to one of two destination buffers stored in memory

The dormant buffer is frozen and available for cyclic software tasks to consume while the other buffer continues to be filled.

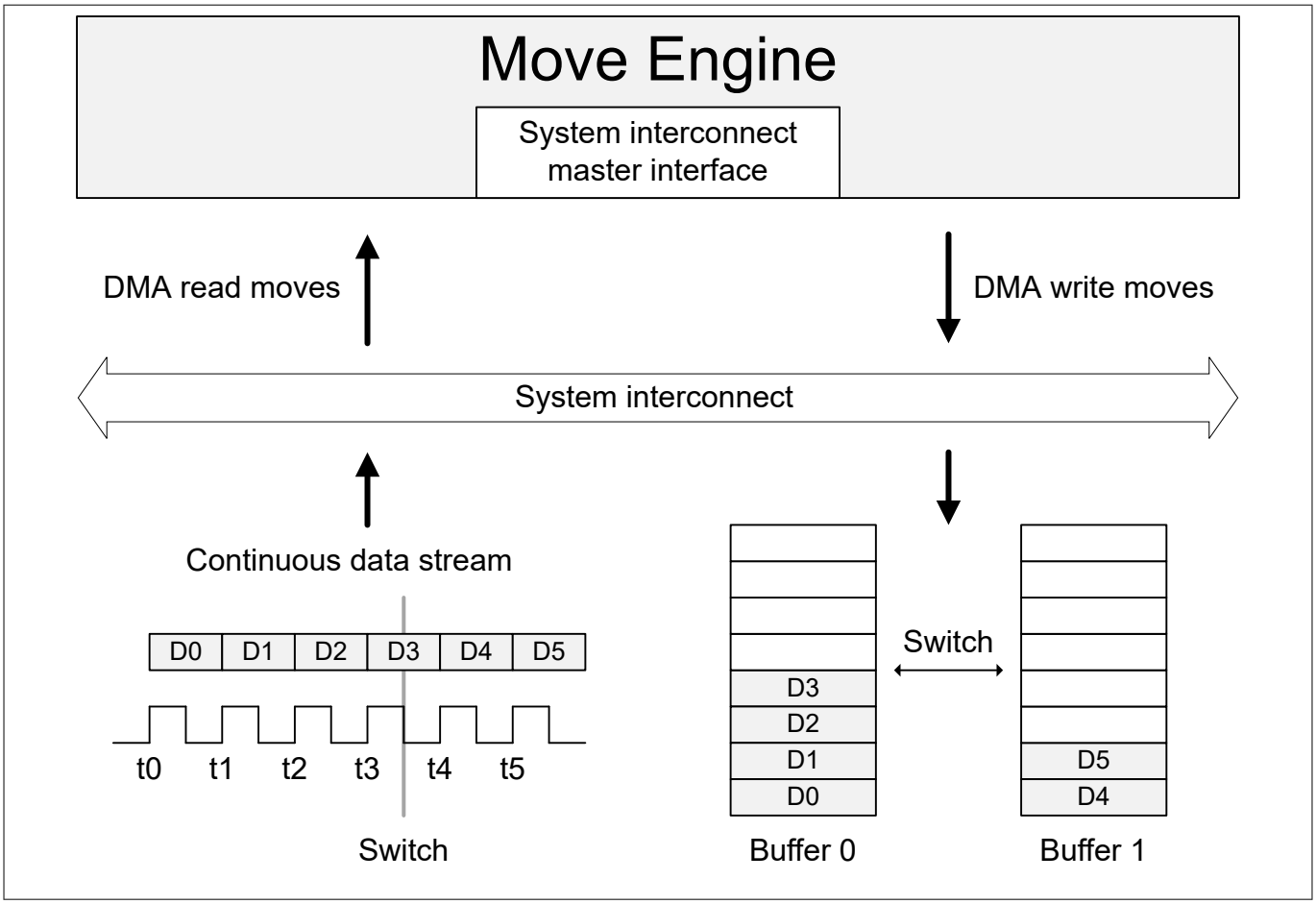


Figure 97 DMA double buffering

If a DMA channel is configured for one of the Double buffering operations and is triggered by DMA hardware request, software should configure the DMA channel for continuous mode (DMA channel CHCFGRc.CHMODE = 1_B).

Related information

DMA hardware request on page 1257

9.3.2.4.1 DMA double source buffering

Data is streamed by the DMA from one of two data buffers (buffer 0 and buffer 1) stored in memory to a continuous output data stream. The double source buffering addresses are:

- The source address SADRc addresses source buffer 0
- The shadow address SHADRc addresses source buffer 1

9.3.2.4.2 DMA double destination buffering

A continuous input data stream is directed by the DMA to one of two data buffers (buffer 0 and buffer 1) stored in memory. The double destination buffering addresses are:

- The destination address DADRc addresses destination buffer 0
- The shadow address SHADRc addresses destination buffer 1

9.3.2.4.3 Size of buffer

The data size of the two source and destination buffers is equal and determined by the following TCS parameters:

- The channel data width (CHCFGRc.CHDW)
- The block mode value (CHCFGRc.BLKM)
- The transfer reload value (CHCFGRc.TREL)

The memory size of the two source and destination buffers is determined by the additional TCS parameters:

- The source circular buffer enable or disable control (ADICRc.SCBE)
- The destination circular buffer enable or disable control (ADICRc.DCBE)
- The source address modification factor (ADICRc.SMF)
- The destination address modification factor (ADICRc.DMF)
- The increment of the source address (ADICRc.INCS)
- The increment of the destination address (ADICRc.INCD)

The size of one buffer is equal to the size of one DMA transaction.

9.3.2.4.4 Buffer switch

The re-direction of the data stream from one buffer to the other buffer is controlled by a software switch and additionally an automatic hardware switch. During a buffer switch:

- No DMA requests are lost by the DMA
- There is no loss, duplication or split of data across the two buffers

Active buffer status

During DMA double buffering the DMA channel CHCSRc.BUFFER bit is used to indicate which buffer is active.

- DMA channel CHCSRc.BUFFER = 0_B: Buffer 0 is read or filled by the DMA
- DMA channel CHCSRc.BUFFER = 1_B: Buffer 1 is read or filled by the DMA

Frozen buffer status

When the DMA switches buffers the DMA channel frozen bit is set (CHCSRc.FROZEN= 1_B). The frozen buffer is then available for a cyclic software task. On completion of the software task the status bit CHCSRc.FROZEN is cleared by software and the buffer address pointer is re-programmed to the start address ready for the next DMA transaction that reads or fills the buffer.

Active buffer empty or full

If the active buffer is emptied (in the case of DMA double source buffering) or filled (in the case of DMA double destination buffering) and no automatic hardware switch occurs before a software buffer switch is received then the DMA channel will stop and no more DMA transfers are made. When the DMA channel stops, the transfer count (CHCSRc.TCOUNT) is equal to 0000_H.

A buffer empty or full event may be signaled by a DMA channel interrupt service request. The DMA channel interrupt control ADICRc.INTCT should be configured to 10_B and the interrupt raise detect value ADICRc.IRDV to 0000_B. When the transfer count equals zero then the DMA channel will raise a DMA channel interrupt service request.

The DMA channel interrupt handler can interrogate the DMA to identify which DMA channel generated the interrupt service request. Software may initialize the DMA channel ready to restart double buffering operations.

Active buffer overflow

If the data rate is faster than the DMA is able to transfer the data to one of the buffers then the DMA reports a double buffer TRL error (DMA channel TSRc.TRL = 1_B and RP ERRSRr.DBTRLER = 1_B). If the DMA channel enable TRL bit is set (TSRc.ETRL = 1_B), the DMA triggers a DMA RP error interrupt service request

Related information

Software switch on page 1236
Automatic hardware switch on page 1236
DMA channel interrupt service request on page 1267
DMA RP error interrupt service request on page 1275

9.3.2.4.5 Software switch

Before a DMA transaction completes and the active buffer is full software shall re-direct the data stream from the active buffer to the other buffer by setting the DMA channel CHCSRc.SWB. If a ME is actively servicing a DMA channel configured for double buffering then the current DMA transfer completes before the buffer switch is made. On completion of the DMA transfer the DMA will:

- Automatically re-load DMA channel CHCSRc.TCOUNT with CHCFGRc.TREL
- Switch the address pointer:
  - Buffer 0 address pointer (source address or destination address)
  - Buffer 1 address pointer (shadow address)

The TCS address control factors remain the same. The next DMA transaction controlling the filling or reading of the new buffer will start when a DMA request is received.

If software re-directs the data stream by setting DMA channel CHCSRc.SWB = 1_B, software shall poll DMA channel CHCSRc.FROZEN AND DMA channel CHCSRc.BUFFER. If software reads DMA channel CHCSRc.FROZEN = 1_B and an active buffer transition, software may start to analyze the frozen buffer.

If a DMA channel is configured for software switch only and the DMA transaction is completed before the DMA receives a software switch, the DMA reports a Double Buffer TRL error (DMA channel TSRc.TRL = 1_B and RP ERRSR.DBTRLER = 1_B). If the DMA channel enable TRL bit is set (TSRc.ETRL = 1_B), the DMA triggers a DMA RP error interrupt service request.

Related information

DMA RP error interrupt service request on page 1275

9.3.2.4.6 Automatic hardware switch

The DMA will automatically switch from the active buffer to the other buffer when the DMA transaction reading or filling the active buffer completes. On switching buffers the DMA channel CHCSRc.FROZEN bit is set to indicate that one buffer is frozen and available for cyclic software tasks. The completion of a DMA transaction may be signaled by a DMA channel interrupt service request. On completion of the DMA transaction when MEmCHSR.TCOUNT equals 0000_H the DMA will:

- Automatically re-load DMA channel CHCSRc.TCOUNT with CHCFGRc.TREL
- Switch the address pointer to select the other buffer:
  - Buffer 0 address pointer (source address or destination address)
  - Buffer 1 address pointer (shadow address)

The DMA channel TCS address control factors remain the same. The next DMA transaction controlling the writing or reading of the new buffer will start when a DMA request is received.

If DMA channel CHCSRc.FROZEN is equal to 1_B, the automatic hardware switch will not occur and the DMA will report a double buffer TRL error (DMA channel TSRc.TRL = 1_B and RP ERRSRr.DBTRLER = 1_B). If the DMA channel enable TRL bit is set (TSRc.ETRL = 1_B), the DMA triggers a DMA RP error interrupt service request.

Related information

DMA channel interrupt service request on page 1267
DMA RP error interrupt service request on page 1275

9.3.2.4.7 Application of double buffering

A typical DMA double buffer application is shown in the following figure:

- DMA channel m is configured for DMA double destination buffering. The DMA channel moves the input data generated by a sensor into the active destination buffer
- A cyclic software task processes the input data stored in the frozen destination buffer and writes the output into the frozen source buffer. The CPU controls both the source and destination buffer switching
- DMA channel n is configured for DMA double source buffering. The DMA channel moves the processed data from the frozen source buffer to the actuator

The system configuration shall balance the input data rate, the duration of cyclic software tasks and the buffer size in order to prevent TRL events. In particular, enabling wait for write completion using the TSRc.WWC bit can delay the completion of the DMA transfers or transactions and this delay should be considered while determining the maximum data rate.

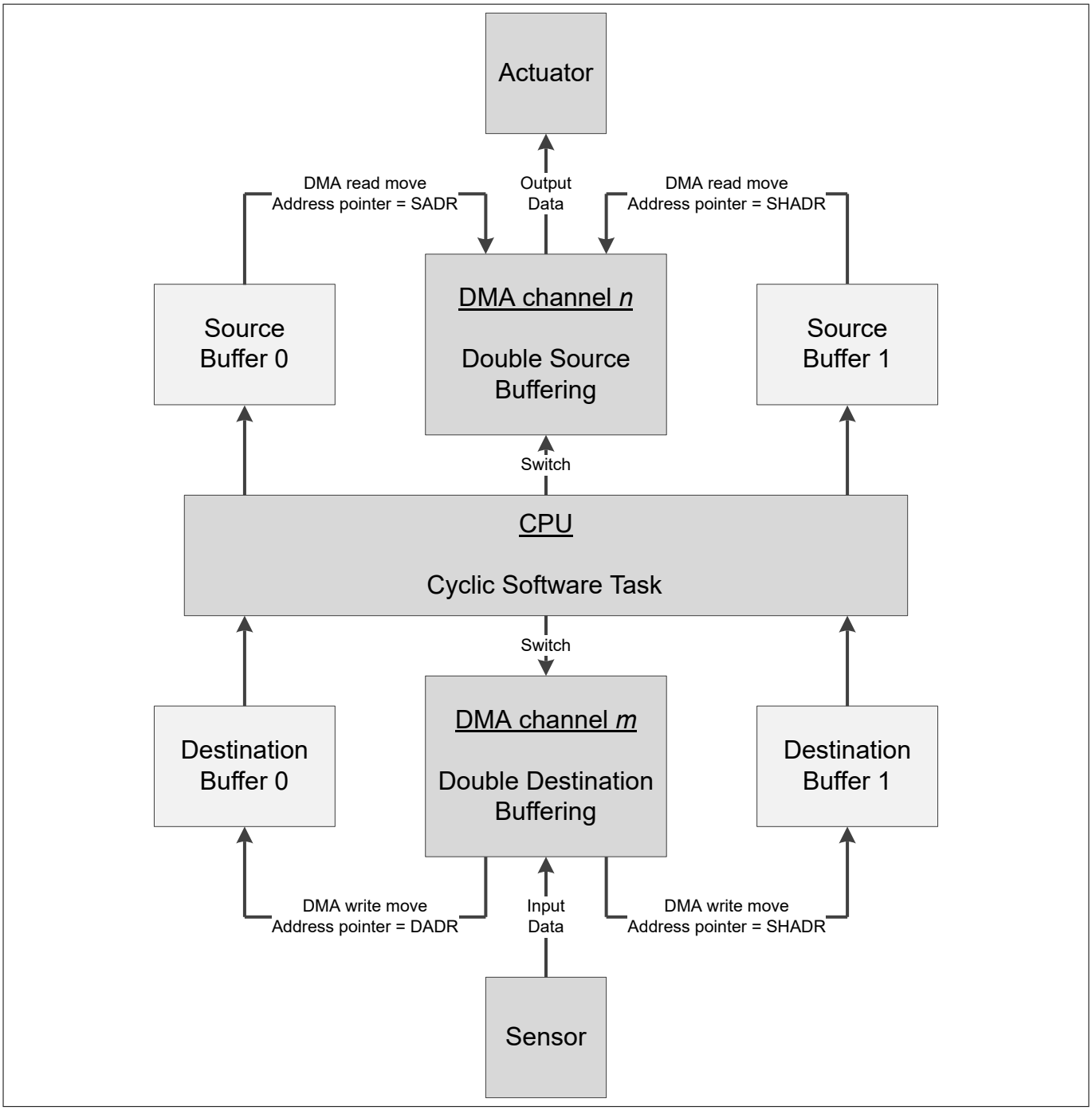


Figure 98 Application of double buffering

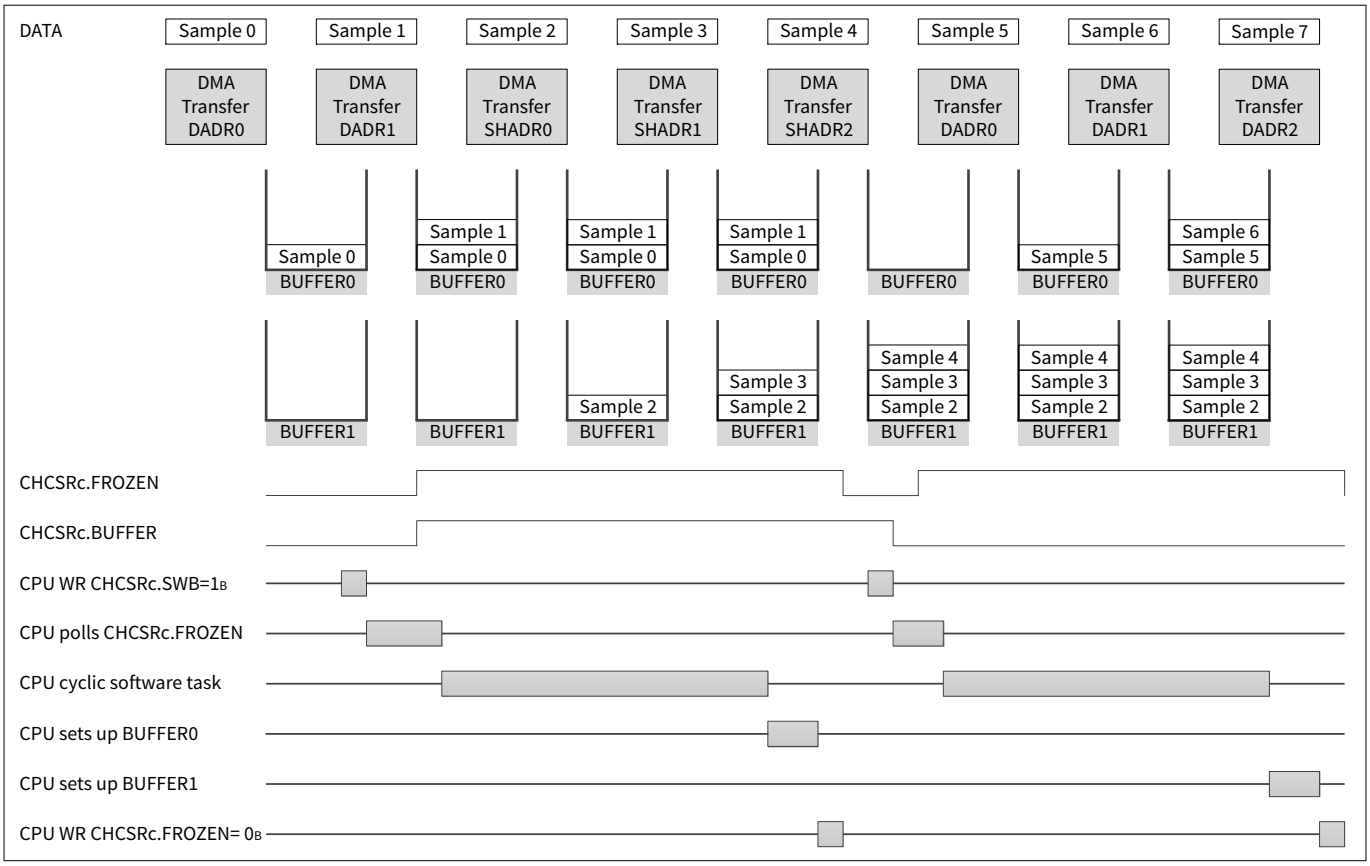


Figure 99 DMA double destination buffering software switch only

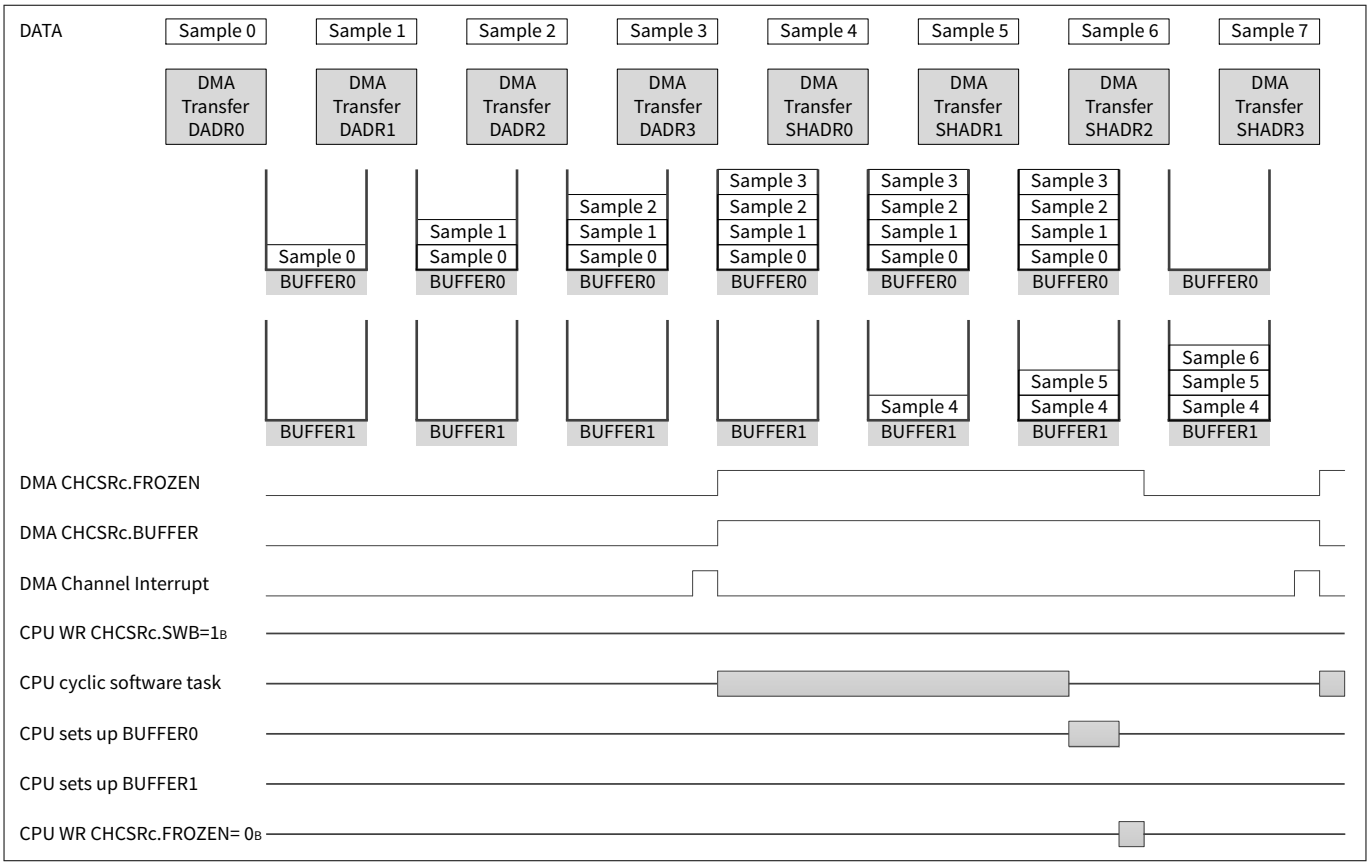


Figure 100 DMA double destination buffering software switch and automatic hardware switch

Related information

DMA double destination buffering on page 1234

DMA double source buffering on page 1234

9.3.2.5 Linked list operations

Linked list operations are an extension of DMA channel functionality to support the more flexible use of the DMA.

A linked list operation consists of a series of DMA transactions executed by the same DMA channel. Each DMA transaction has a unique TCS. The source and destination areas do not have to exist in contiguous areas of memory.

If a DMA channel is configured for linked list operation then as soon as the ME completes the current DMA transaction, the ME reads the next TCS from memory and overwrite the current TCS at the DMA channel location in DMARAM. The current DMA transaction uses a 32-byte aligned address pointer to point to the next TCS stored in either internal or external memory. The DMA starts reading the next TCS from word 0 upwards. There is no limit to the number of DMA transactions in a linked list operation. The first DMA transaction in a linked list operation has to be initiated by a DMA hardware request or a DMA software request. Subsequent DMA transactions may be initiated by a DMA auto start request.

Each DMA channel supports the following types of linked list operation:

- DMA Linked List (DMALL)
- Accumulated Linked List (ACCLL)
- Checked Linked List (CHKLL)
- Conditional Linked List (CONLL)

Related information

DMA auto start request on page 1261

Accumulated linked list (ACCLL) on page 1243

DMA linked list (DMALL) on page 1241

Checked linked list (CHKLL) on page 1243

Conditional linked list (CONLL) on page 1246

9.3.2.5.1 Last DMA transaction

A DMA linked list chain of operations may be terminated in one of the following two ways.

- With a non-linked list TCS: If the last TCS is not configured for linked list operation, the last TCS should be configured for a move operation instead by setting ADICRc.SHCT = 0_H. In this case, the DMA channel exits linked list operation. The last TCS will be loaded into the DMA channel, but a DMA auto start request will not initiate the transaction
- With a linked list TCS: If the last TCS is configured for a linked list operation but has the DMA auto start request bit (CHCSRc.SCH) bit clear, the last TCS is loaded into the DMA channel but is not triggered

A DMA channel interrupt service request may be used to signal the completion of the last DMA transaction in a sequence of linked list operations.

Related information

Move operation on page 1211

DMA auto start request on page 1261

DMA channel interrupt service request on page 1267

9.3.2.5.2 Circular linked list operations

A sequence of linked list operations may be configured for circular operation. The same series of DMA transactions is repeated.

9.3.2.5.3 DMA linked list (DMALL)

The DMA channel shadow address register (SHADRc) stores the 32-bit address pointer to the next TCS. As soon as the current DMA transaction completes, the ME reads the next TCS and stores it in DMARAM.

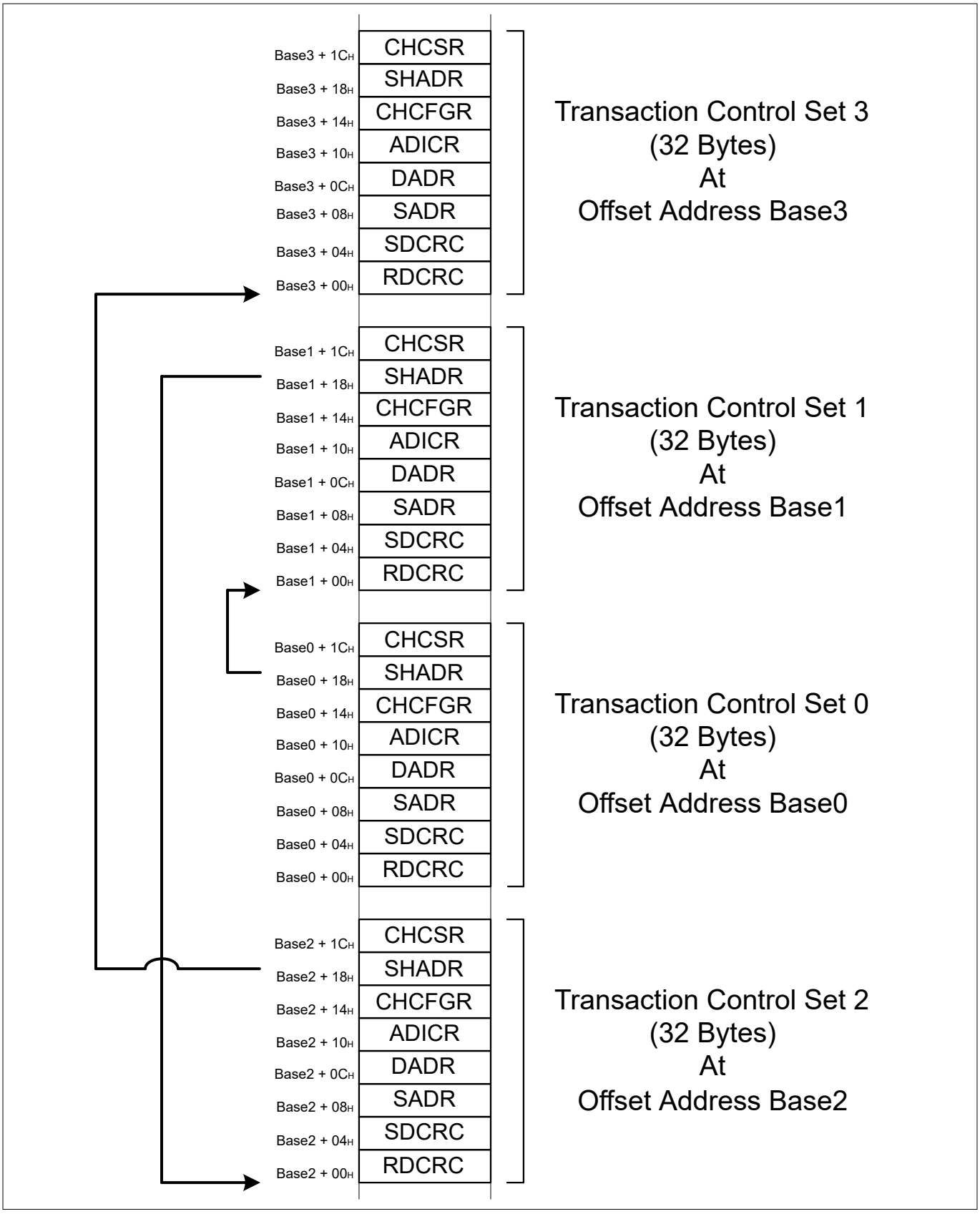


Figure 101 DMA linked list

DMA address checksum and DMA data checksum

Software shall configure the initial value of DMA address checksum and DMA data checksum used during the first DMA transaction. When the ME reads the next TCS, the DMA address checksum and DMA data checksum is initialized to 00000000_H.

Related information

DMA address checksum on page 1222
DMA data checksum on page 1223

9.3.2.5.4 Accumulated linked list (ACCLL)

ACCLL is a variant of DMALL and has an identical footprint in memory (size and structure). The SDCRC and RDCRC checksums are accumulated across DMA transactions.

DMA address checksum and DMA data checksum

The DMA address checksum and DMA data checksum are not overwritten when the next TCS is loaded into the DMA channel. As long as the DMA channel is configured for ACCLL, the DMA address checksum and DMA data checksum are calculated across all DMA transactions.

Related information

DMA address checksum on page 1222
DMA data checksum on page 1223

9.3.2.5.5 Checked linked list (CHKLL)

CHKLL provides protection against software errors. If the address pointer in a sequence of linked list operations is incorrectly written then the address pointer may point to any random area of memory and load the next TCS from the wrong location and execute it. As long as a DMA channel is configured for CHKLL, the ME checks that the calculated DMA address checksum matches an expected DMA address checksum stored in the next TCS.

Attention**: **Checked Linked List support is considered deprecated functionality in the AURIX™ TC4xx family. This functionality may be removed in a future AURIX™ generation.

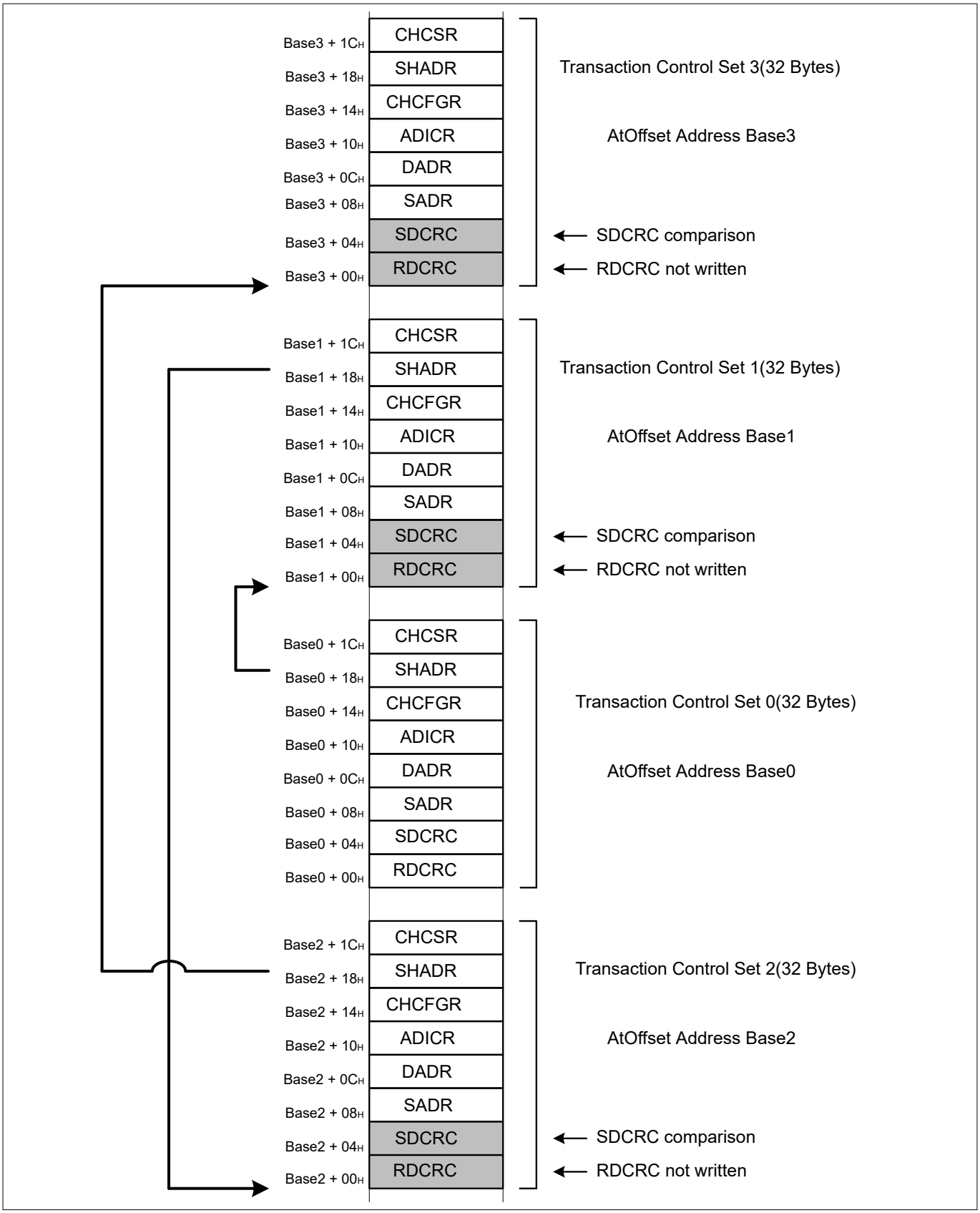


Figure 102 Checked linked list

DMA address checksum

The user is required to load the SDCRC word with an expected DMA address checksum value. As soon as the ME has read the next TCS from memory, the ME compares the DMA address checksum calculated by the current DMA transaction against the expected DMA address checksum stored in the next TCS. If the checksums match then the DMA proceeds with the execution of the next TCS. If the checksums do not match then the ME records a CHKLL DMA address checksum error and triggers a DMA RP error interrupt service request. The DMA stops the execution of the linked list operation. Assuming all the DMA address checksum values match during the sequence of linked list operations then the DMA address checksum is calculated across all DMA transactions.

DMA data checksum

The DMA data checksum is not overwritten when the new TCS is loaded into the DMA channel. The DMA data checksum is calculated across all DMA transactions.

Related information

DMA address checksum on page 1222
DMA RP error interrupt service request on page 1275
DMA data checksum on page 1223

9.3.2.5.6 Conditional linked list (CONLL)

A special use of a linked list is CONLL. Selection of the address pointer to the next TCS is determined by a conditional state. CONLL uses the shadow address (SHADRc) and the source and destination address CRC registers (SDCRCRc) as address pointers.

A CONLL operation is limited to 8-bit DMA moves (CHCFGRc.CHDW = 000_B). The source address shall not be configured to a cached address (segments 8 and 9). A non-cached address (segments A and B) shall be configured to prevent the ME read buffer re-aligning the move data.

CONLL pattern detection

The DMA channel in the figure below is configured for CONLL. In this example, the TCS is configured to select PRRAr for a pattern compare.

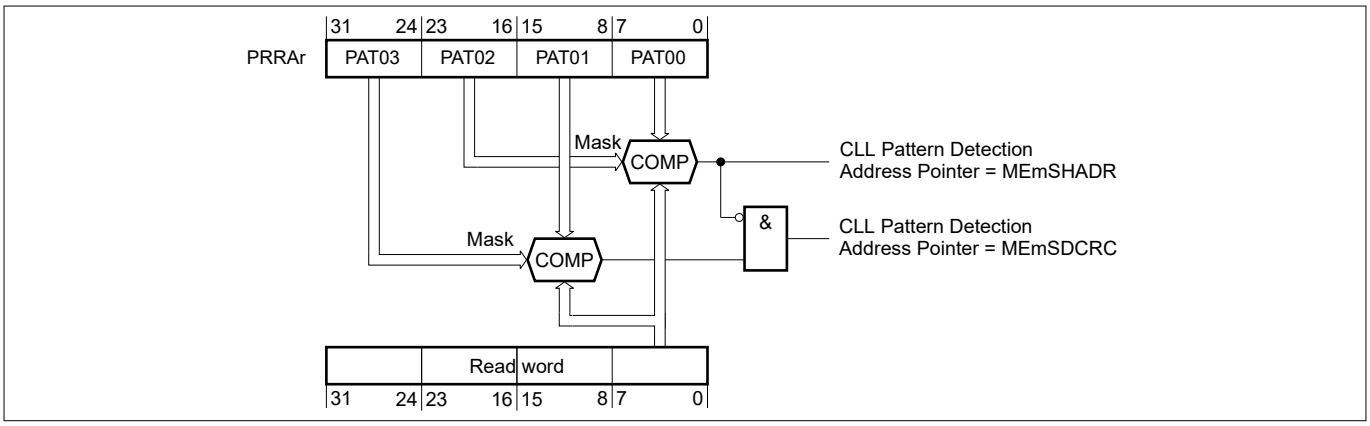


Figure 103 Conditional linked list and pattern compare logic

During each DMA move the pattern detection logic determines the selection of the address pointer:

- If PAT00 masked by PAT02 matches the lowest byte of the DMA read move data then as soon as the DMA move completes the ME loads the DMA channel with the next TCS read from an address stored in SHADRc
- If PAT01 masked by PAT03 matches the lowest byte of the DMA read move data then as soon as the DMA move completes the ME loads the DMA channel with the next TCS read from an address stored in SDCRCRc
- If both PAT00 masked by PAT02 and PAT01 masked by PAT03 match the lowest byte of the DMA read move data then as soon as the DMA move completes, the ME loads the DMA channel with the next TCS read from an address stored in SHADRc
- If there is no pattern match then the ME continues with the DMA transaction

If a pattern match is detected:

- The DMA transaction ends with the current DMA move
- The DMA channel TSRc.CH will be cleared when the DMA write move has completed

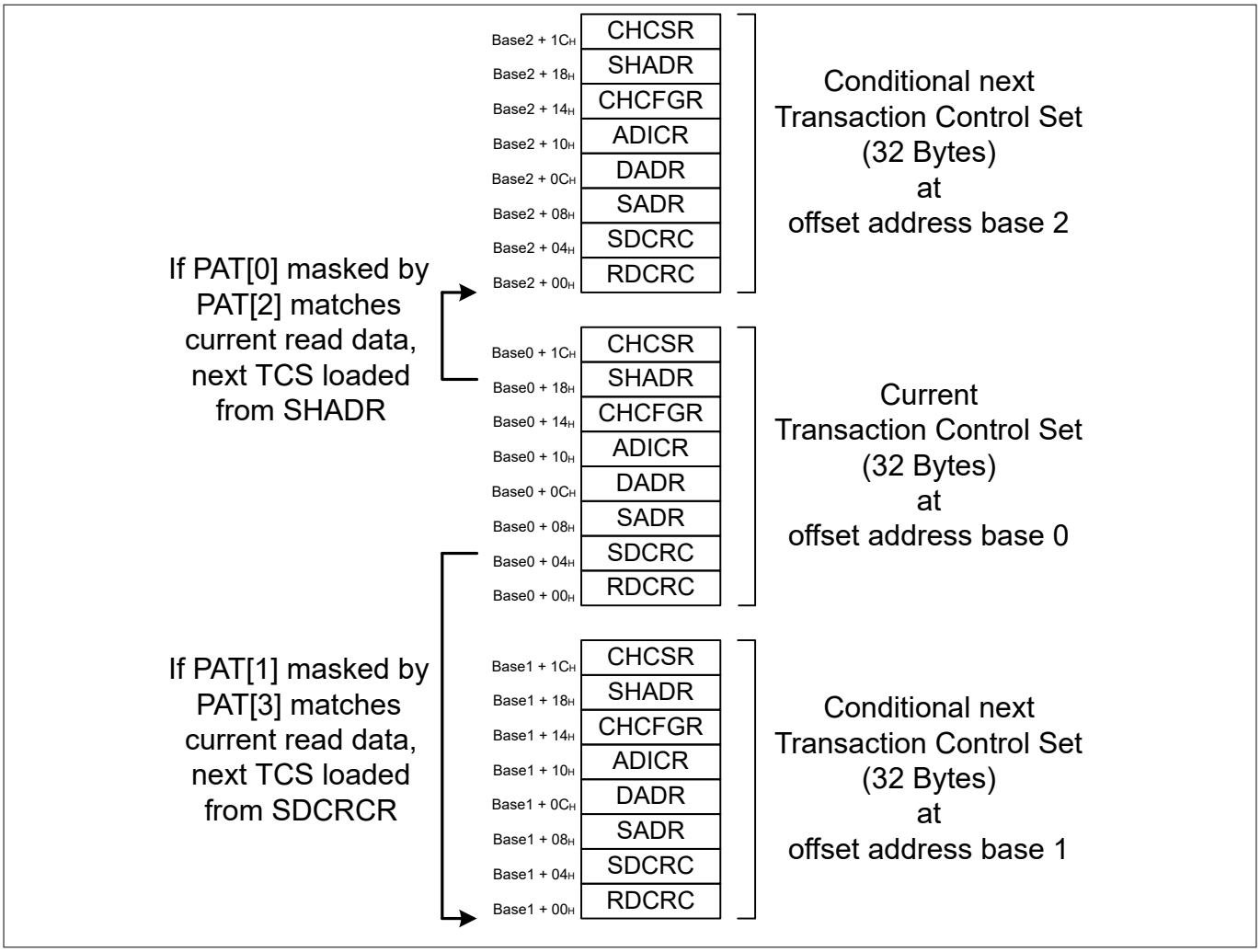


Figure 104 Conditional linked list operation

Completion of current DMA transaction

If the current DMA transaction completes when MEmCHSR.TCOUNT = 0_D and there is no pattern match then the CONLL will load the DMA channel with the next TCS read from an address stored in SHADRc.

Multiple pattern detection conditions

The user may intentionally program PAT00 masked by PAT02 shall not match with the lowest byte of the DMA read move data and transition across a series of DMA transactions in the CONLL. For each DMA transaction the user may program a series of different PAT01 masked by PAT03 values and test if each value matches with the current lowest byte of the DMA read move data. If PAT01 masked by PAT03 matches the current lowest byte of the DMA read move data, then as soon as the DMA move is completed, the ME loads the DMA channel with the next TCS read from an address stored in SDCRCRc.

Related information

ME read buffer on page 1280

9.3.2.6 DMA timestamp

As soon as the last DMA transfer of a DMA transaction has completed, a 32-bit DMA timestamp may be appended to record the occurrence of a DMA operation.

The DMA timestamp is the time at which the event was recorded by the DMA and not the real time. The DMA transaction completes after the DMA timestamp is written.

9.3.2.6.1 Generation of DMA timestamp

The DMA timestamp is generated from a 32-bit binary upwards synchronous counter clocked by the global system timer clock (f_{STM}). The counter is reset by system reset and counts as long as the f_{STM} clock is running and has not been suspended through the OCDS suspend input and the DMA TIMEOCS register. The current 32-bit timestamp value may be read or updated through the DMA TIME register.

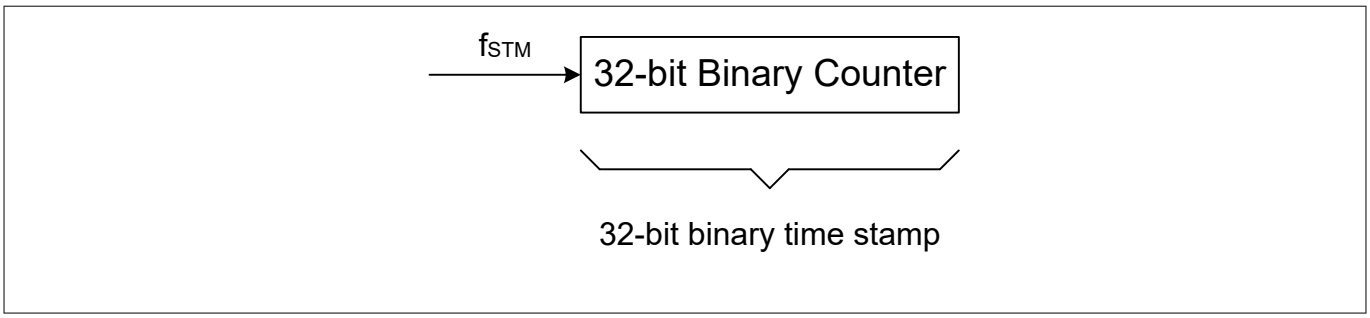


Figure 105 Generation of DMA timestamp

9.3.2.6.2 Appendage of DMA timestamp to non-destination circular buffer

The appendage of a DMA timestamp to the destination data is controlled by the DMA channel ADICRc.STAMP bit.

If a DMA channel is configured for the appendage of a DMA timestamp and not destination circular buffer operation (ADICRc.DCBE = 0_B), the ME writes a DMA timestamp at a destination address defined by:

- If ADICRc.INCD = 1_B, the address of the DMA timestamp is the next higher word aligned destination address
- If ADICRc.INCD = 0_B, the address of the DMA timestamp is the next lower word aligned destination address

Appendage of DMA timestamp - example 1

16-bit half-words are moved from a source memory with an incrementing source address offset of 10_H to a destination memory with an incrementing destination address offset of 08_H.

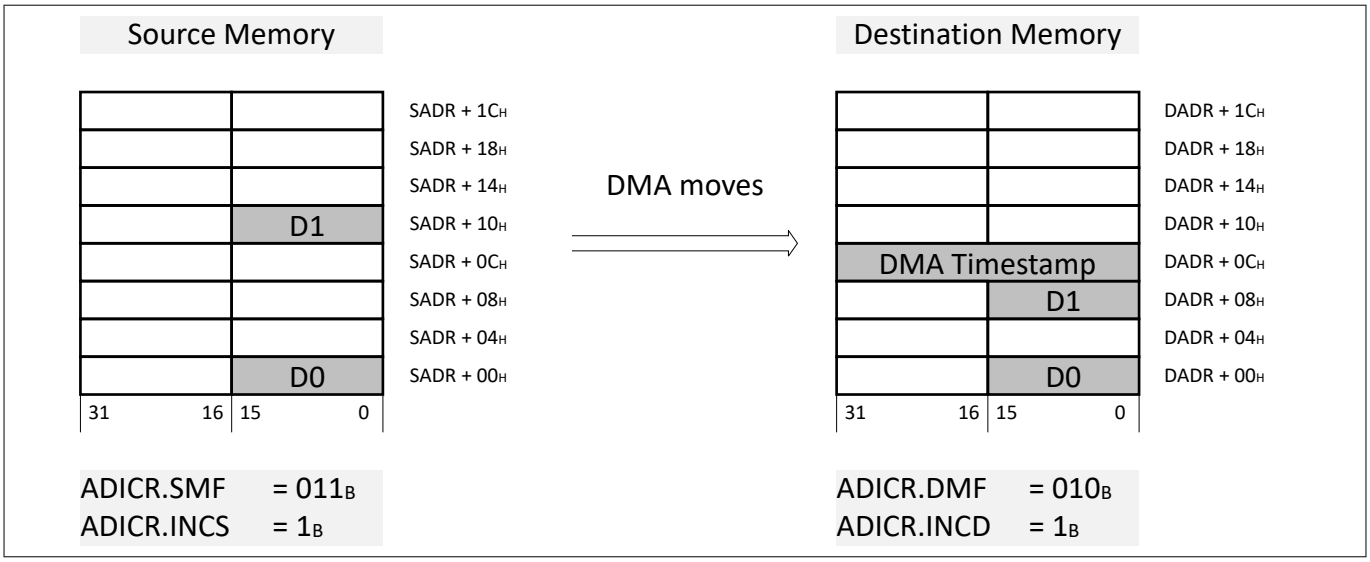


Figure 106 Incrementing destination address: DMA timestamp appended above DMA write move data

Appendage of DMA timestamp - example 2

16-bit half-words are moved from a source memory with an incrementing source address offset of 02_H to a destination memory with a decrementing destination address offset of 04_H.

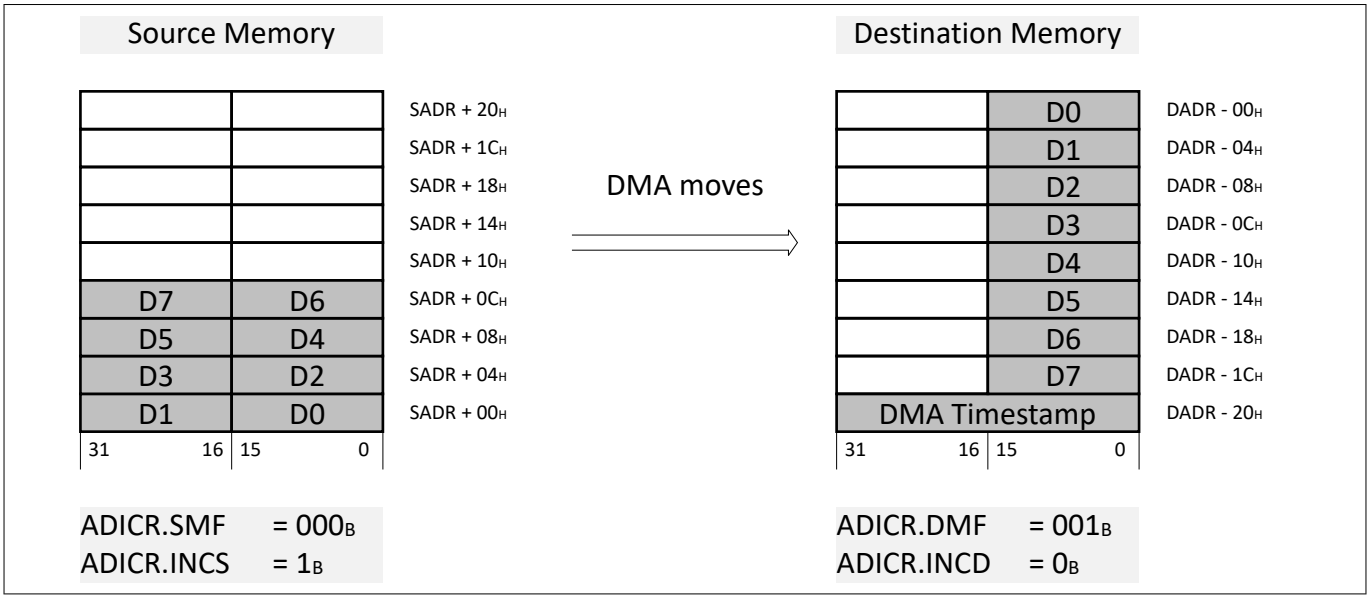


Figure 107 Decrementing destination address: DMA timestamp appended below DMA write move data

Destination address generation

The destination address of the next DMA write move of the next DMA transaction is calculated using:

- The destination address of the last DMA write move of the current DMA transaction
- The destination address offset defined by ADICRc.INCD, ADICRc.DMF and CHCFGRc.CHDW

The ME checks the destination address of the next DMA write move.

If the currently calculated destination address of the next DMA write move results in DMA write move data overwriting the DMA timestamp of the current DMA transaction, the ME calculates a new destination address:

- If DMA channel ADICRc.INCD = 1_B, the ME selects the greater value of the following:
  - Add (2_D x destination address offset) to the destination address of last DMA write move
  - Add 8_D to the destination address of last DMA write move of the current DMA transaction
- If DMA channel ADICRc.INCD = 0_B, the ME selects the lesser value of the following:
  - Subtract (2_D x destination address offset) from the destination address of last DMA write move
  - Subtract 8_D from the destination address of last DMA write move of the current DMA transaction

The ME writes back the DMA channel TCS including the updated destination address to the DMARAM.

9.3.2.6.3 Appendage of DMA timestamp to destination circular buffer

If a DMA channel is configured for the appendage of a DMA timestamp and destination circular buffer operation (ADICRc.DCBE = 1_B), the ME writes a DMA timestamp at a destination address defined by:

- If ADICRc.INCD = 1_B AND ADICRc.CBLD = 1 or 0, the address of the DMA timestamp = (DADRc[31:2] + 1) << 2
- If ADICRc.INCD = 1_B AND ADICRc.CBLD > 1, the address of the DMA timestamp = (DADRc[31:CBLD] + 1) << CBLD
- If ADICRc.INCD = 0_B AND ADICRc.CBLD = 1 or 0, the address of the DMA timestamp = (DADRc[31:2] << 2) - 4
- If ADICRc.INCD = 0_B AND ADICRc.CBLD > 1, the address of the DMA timestamp = (DADRc[31:CBLD]) << CBLD) - 4

The ME writes back to the DMARAM an incremented and wrapped destination address.

9.3.2.6.4 Application of DMA timestamp

The appendage of a DMA timestamp may be used by software to perform a temporal analysis of DMA transaction completion times. Software may for example analyze relative DMA timestamp values to determine the correct sequencing of DMA transactions.

9.3.3 DMA channels

Each DMA channel stores the context of an independent DMA operation.

9.3.3.1 Transaction control set

Each DMA channel provides eight 32-bit control registers, together called the DMA channel transaction control set (TCS) that define the DMA channel operation. Software may configure the DMA channel TCS to define the move function of each DMA channel.

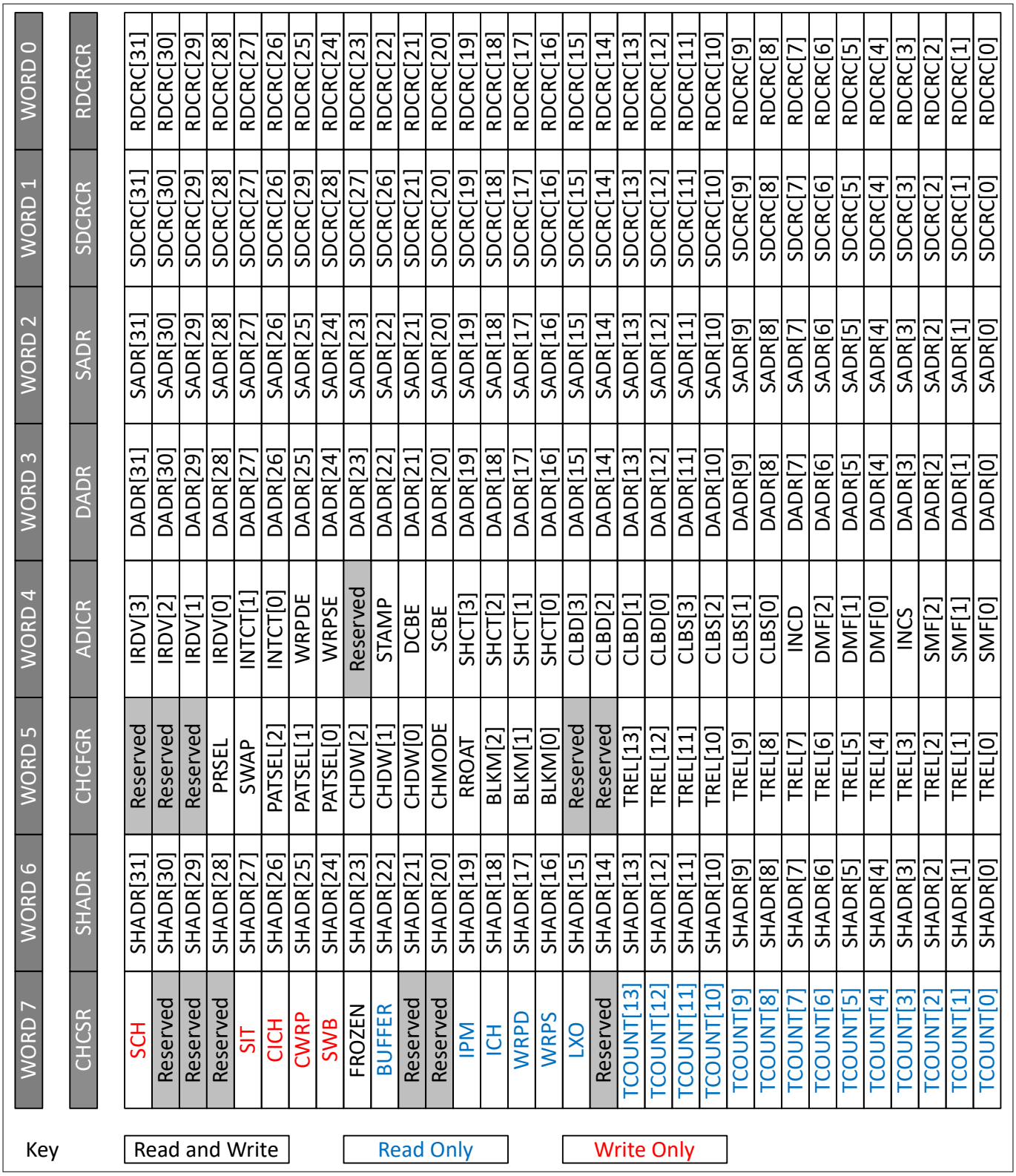


Figure 108 DMA channel TCS organization

9.3.3.1.1 DMA channel operation

The following table lists the supported types of DMA channel operation and system interconnect access sizes that can be configured in the DMA channel TCS.

Table 634 DMA channel operation

PATSEL	STAMP	SHCT	Description	BTR8	BTR4	BTR2	SDTD	SDTW	SDTH	SDTB
000_B	0_B	0000_B	Move operation	Yes	Yes	Yes	Yes	Yes	Yes	Yes
000_B	0_B	0001_B	Shadow operation read only mode source address	Yes	Yes	Yes	Yes	Yes	Yes	Yes
000_B	0_B	0010_B	Shadow operation read only mode destination address	Yes	Yes	Yes	Yes	Yes	Yes	Yes
000_B	0_B	0101_B	Shadow operation direct write mode source address	Yes	Yes	Yes	Yes	Yes	Yes	Yes
000_B	0_B	0110_B	Shadow operation direct write mode destination address	Yes	Yes	Yes	Yes	Yes	Yes	Yes
000_B	0_B	1000_B	DMA double source buffering Software switch only	Yes	Yes	Yes	Yes	Yes	Yes	Yes
000_B	0_B	1001_B	DMA double source buffering Software switch and Automatic hardware switch	Yes	Yes	Yes	Yes	Yes	Yes	Yes
000_B	0_B	1010_B	DMA double destination buffering Software switch only	Yes	Yes	Yes	Yes	Yes	Yes	Yes
000_B	0_B	1011_B	DMA double destination buffering Software switch and Automatic hardware switch	Yes	Yes	Yes	Yes	Yes	Yes	Yes
000_B	0_B	1100_B	DMA Linked List (DMALL)	Yes	Yes	Yes	Yes	Yes	Yes	Yes
000_B	0_B	1101_B	Accumulated Linked List (ACCLL)	Yes	Yes	Yes	Yes	Yes	Yes	Yes
000_B	0_B	1110_B	Checked Linked List (CHKLL)	Yes	Yes	Yes	Yes	Yes	Yes	Yes
000_B	0_B	1111_B	Conditional Linked List (CONLL) - DMA read move data compared to PRRAr	No	No	No	No	No	No	Yes
100_B	0_B	1111_B	Conditional Linked List (CONLL) - DMA read move data compared to PRRBr	No	No	No	No	No	No	Yes
000_B	1_B	0000_B	Move operation with DMA timestamp	Yes	Yes	Yes	Yes	Yes	Yes	Yes
000_B	1_B	0001_B	Shadow operation read only mode source address with DMA timestamp	Yes	Yes	Yes	Yes	Yes	Yes	Yes
000_B	1_B	0010_B	Shadow operation read only mode destination address with DMA timestamp	Yes	Yes	Yes	Yes	Yes	Yes	Yes
000_B	1_B	0101_B	Shadow operation direct write mode source address with DMA timestamp	Yes	Yes	Yes	Yes	Yes	Yes	Yes
000_B	1_B	0110_B	Shadow operation direct write mode destination address with DMA timestamp	Yes	Yes	Yes	Yes	Yes	Yes	Yes
000_B	1_B	1001_B	DMA double source buffering with DMA timestamp[1] - Software switch and Automatic hardware switch	Yes	Yes	Yes	Yes	Yes	Yes	Yes
000_B	1_B	1011_B	DMA double destination buffering with DMA timestamp[1] - Software switch and Automatic hardware switch	Yes	Yes	Yes	Yes	Yes	Yes	Yes
000_B	1_B	1100_B	DMA Linked List (DMALL) with DMA timestamp	Yes	Yes	Yes	Yes	Yes	Yes	Yes
000_B	1_B	1101_B	Accumulated Linked List (ACCLL) with DMA timestamp	Yes	Yes	Yes	Yes	Yes	Yes	Yes
000_B	1_B	1110_B	Checked Linked List (CHKLL) with DMA timestamp	Yes	Yes	Yes	Yes	Yes	Yes	Yes

(table continues...)

Table 634 (continued) DMA channel operation

PATSEL	STAMP	SHCT	Description	BTR8	BTR4	BTR2	SDTD	SDTW	SDTH	SDTB
001_B	0_B	0000_B	Move operation[2)] Pattern detection for 8-bit channel data width - lower pattern compare to PRRAr	No	No	No	No	No	No	Yes
010_B	0_B	0000_B	Move operation[2)] Pattern detection for 8-bit channel data width - upper pattern compare to PRRAr	No	No	No	No	No	No	Yes
011_B	0_B	0000_B	Move operation[2)] Pattern detection for 8-bit channel data width - two byte pattern detection sequence compared to PRRAr	No	No	No	No	No	No	Yes
001_B	0_B	0000_B	Move operation[2)] Pattern detection for 16-bit channel data width - aligned mode compared to PRRAr	No	No	No	No	No	Yes	No
010_B	0_B	0000_B	Move operation[2)] Pattern detection for 16-bit channel data width - unaligned mode 1 or unaligned mode 2 compared to PRRAr	No	No	No	No	No	Yes	No
011_B	0_B	0000_B	Move operation[2)] Pattern detection for 16-bit channel data width - combined mode compared to PRRAr	No	No	No	No	No	Yes	No
001_B	0_B	0000_B	Move operation[2)] Pattern detection for 32-bit channel data width - lower data half word compared to PRRAr	No	No	No	No	Yes	No	No
010_B	0_B	0000_B	Move operation[2)] Pattern detection for 32-bit channel data width - upper data half word compared to PRRAr	No	No	No	No	Yes	No	No
011_B	0_B	0000_B	Move operation[2)] Pattern detection for 32-bit channel data width - complete data word compared to PRRAr	No	No	No	No	Yes	No	No
101_B	0_B	0000_B	Move operation[2)] Pattern detection for 8-bit channel data width - lower pattern compare to PRRBr	No	No	No	No	No	No	Yes
110_B	0_B	0000_B	Move operation[2)] Pattern detection for 8-bit channel data width - upper pattern compare to PRRBr	No	No	No	No	No	No	Yes
111_B	0_B	0000_B	Move operation[2)] Pattern detection for 8-bit channel data width - two byte pattern detection sequence compared to PRRBr	No	No	No	No	No	No	Yes
101_B	0_B	0000_B	Move operation[2)] Pattern detection for 16-bit channel data width - aligned mode compared to PRRBr	No	No	No	No	No	Yes	No
110_B	0_B	0000_B	Move operation[2)] Pattern detection for 16-bit channel data width - unaligned mode 1 or unaligned mode 2 compared to PRRBr	No	No	No	No	No	Yes	No
111_B	0_B	0000_B	Move operation[2)] Pattern detection for 16-bit channel data width - combined mode compared to PRRBr	No	No	No	No	No	Yes	No
101_B	0_B	0000_B	Move operation[2)] Pattern detection for 32-bit channel data width - lower data half word compared to PRRBr	No	No	No	No	Yes	No	No
110_B	0_B	0000_B	Move operation[2)] Pattern detection for 32-bit channel data width - upper data half word compared to PRRBr	No	No	No	No	Yes	No	No

(table continues...)

Table 634 (continued) DMA channel operation

PATSEL	STAMP	SHCT	Description	BTR8	BTR4	BTR2	SDTD	SDTW	SDTH	SDTB
111_B	0_B	0000_B	Move operation[2] Pattern detection for 32-bit channel data width - complete data word compared to PRRBr	No	No	No	No	Yes	No	No
Others	Reserved [3]	N/A	N/A	N/A	N/A	N/A	N/A	N/A		

1) DMA timestamp is only appended on completion of a DMA transaction.
2) DMA channel operation “pattern detection” functions for all types of DMA channel shadow operation.
3) If a DMA channel is configured for a reserved value of DMA channel ADICRc.SHCT and a DMA request is received, the DMA clears DMA channel TSRc.CH and performs no DMA moves.

Related information

Move operation on page 1211
Shadow operation on page 1230
DMA double source buffering on page 1234
Software switch on page 1236
DMA double destination buffering on page 1234
Automatic hardware switch on page 1236
Accumulated linked list (ACCLL) on page 1243
DMA linked list (DMALL) on page 1241
Checked linked list (CHKLL) on page 1243
Conditional linked list (CONLL) on page 1246
DMA timestamp on page 1248
Pattern detection for 8-bit channel data width on page 1226
Pattern detection for 16-bit channel data width on page 1228
Pattern detection for 32-bit channel data width on page 1229

9.3.3.1.2 TCS initialization

On the assertion of an application reset, the DMA initializes the TCS configuration data of all channels to all zeros to prevent the triggering of DMARAM data integrity errors in the application.

9.3.3.2 DMA channel request control

DMA channel request control logic as shown in the following figure is implemented for each DMA channel.

The DMA channel operation is individually programmable. The following types of DMA requests are possible:

- DMA software request initiated by a CPU
- DMA hardware request initiated by the Interrupt Router (IR) Interrupt Control Unit (ICU)
- DMA daisy chain request initiated by the next higher priority DMA channel
- DMA auto start request initiated by the loading of the next TCS during a DMA Linked List (DMALL), Accumulated Linked List (ACCLL), Checked Linked List (CHKLL), or Conditional Linked List (CONLL) operation

The DMA channel status flag TSRc.CH indicates if a DMA request is pending. TSRc.CH may be cleared at the start of a DMA transfer or at the end of a DMA transaction depending on the value of CHCFGRc.RROAT. It follows that a DMA request may trigger a single DMA transfer or one complete DMA transaction.

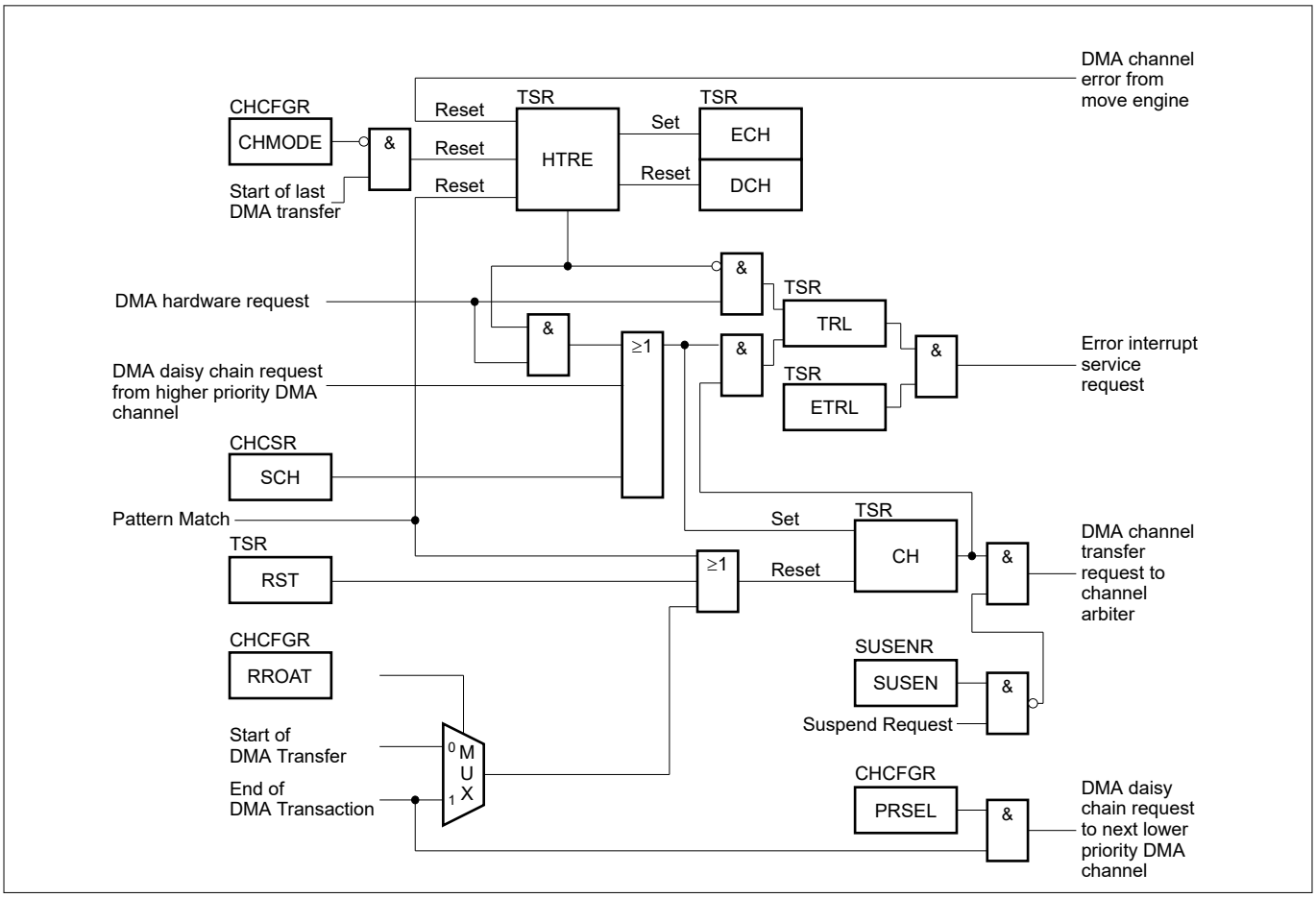


Figure 109 DMA channel request control

Related information

DMA software request on page 1256
DMA hardware request on page 1257
DMA daisy chain request on page 1260
DMA auto start request on page 1261
DMA linked list (DMALL) on page 1241
Accumulated linked list (ACCLL) on page 1243
Checked linked list (CHKLL) on page 1243
Conditional linked list (CONLL) on page 1246

9.3.3.2.1 Reset request only after transaction (RROAT)

The clearing of the DMA channel TSRc.CH bit is controlled as follows:

- CHCFGRc.RROAT = 0_B
  - DMA channel TSRc.CH is cleared at the start of each DMA transfer
  - One DMA request starts one single DMA transfer
- CHCFGRc.RROAT = 1_B
  - DMA channel TSRc.CH is cleared at the end of each DMA transaction
  - One DMA request starts one complete DMA transaction

9.3.3.2.2 DMA software request

One DMA software request may start one complete DMA transaction or one single DMA transfer.

Software may configure the DMA channel to be triggered only by software by disabling DMA hardware requests. Software must set TSRc.DCH to 1_B to disable a DMA hardware request (TSRc.HTRE = 0_B).

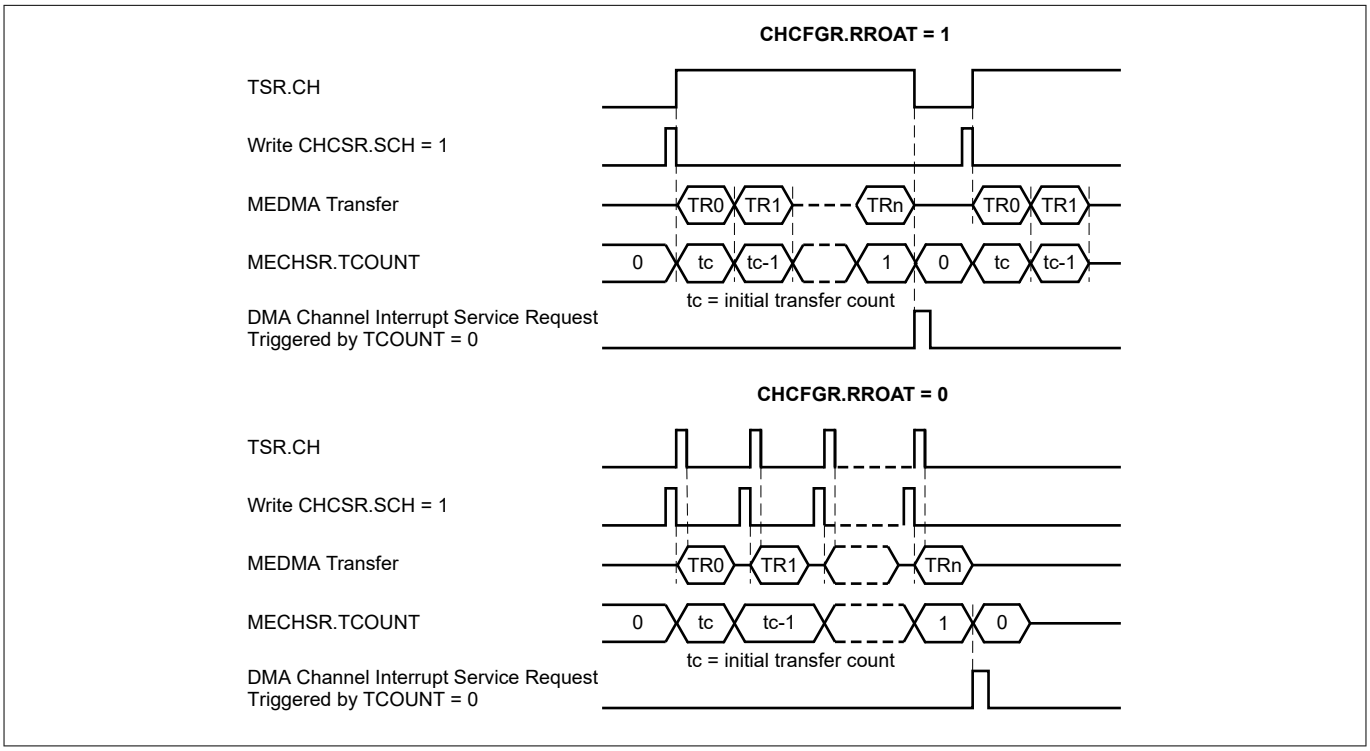


Figure 110 Software control

The following DMA channel configuration is required to initiate one complete DMA transaction under software control:

- DMA channel CHCFGRc.RROAT = 1_B

A DMA software request is initiated by setting the DMA channel CHCSRc.SCH to 1_B with the result that TSRc.CH is set to 1_B. At the start of the DMA transaction, the value of the DMA channel reload value CHCFGRc.TREL is loaded into MEmCHSR.TCOUNT and DMA transfers are executed. After each DMA transfer, MEmCHSR.TCOUNT is decremented and the next source and destination addresses are calculated. When TCOUNT decrements to 0_D, the DMA channel status flag TSRc.CH is reset. Setting the DMA channel CHCSRc.SCH = 1_B again starts a new DMA transaction of the DMA channel with the stored TCS parameters.

The following DMA channel configuration is required to initiate each single DMA transfer under software control:

- DMA channel CHCFGRc.RROAT = 0_B

A DMA software request must be initiated for each DMA transfer by setting the DMA channel CHCSRc.SCH to 1_B.

Related information

DMA hardware request on page 1257
DMA software request on page 1256

9.3.3.2.3 DMA hardware request

Interrupt Requests are prioritized by the Interrupt Router and processed by one of the Service Providers (CPU, DMA or other modules). The DMA interfaces to an interrupt control unit (ICU) instantiated in the interrupt router (IR). Interrupts processed by the DMA ICU interface are treated as DMA hardware requests.

DMA channels are associated with the Service Request Priority Number (SRPN) bit-field programmed in the IR service request control (SRC) register. For example:

- DMA channel 000 equates to SRPN = 0_D programmed in IR
- DMA channel 001 equates to SRPN = 1_D programmed in IR
- DMA channel 002 equates to SPRN = 2_D programmed in IR
- DMA channel 003 equates to SRPN = 3_D programmed in IR
- DMA channel 004 equates to SPRN = 4_D programmed in IR

The DMA will acknowledge all service requests from the IR. If the value programmed in the SRC.SRPN is for an invalid DMA channel then the DMA will take no action. The user must program valid SRC.SRPN values for the DMA.

The interrupt request from the IR contains a CS bit which indicates whether the corresponding interrupt has been configured as part of the cyber-security (CS) or non-cyber-security (non-CS) groups in the IR. The DMA takes no action for a DMA hardware service request where the CS bit from the IR does not match the MODEr.CS bit from the resource partition the triggered DMA channel belongs to.

A DMA hardware request is enabled or disabled by the DMA channel TSRc.HTRE bit. The HTRE functionality is as follows:

- Software may set (TSRc.ECH = 1_B) or clear (TSRc.DCH = 1_B) DMA channel TSRc.HTRE
- Cleared as a result of the ME reporting an error for the DMA channel
- Single mode: At the start of the last DMA transfer of a DMA transaction

Table 635 Conditions to set or reset DMA channel TSRc.HTRE

DMA channel TSRc.ECH	DMA channel TSRc.DCH	Start of last DMA transfer of a DMA transaction[1)]	Modification of DMA channel TSRc.HTRE
0	0	0	Unchanged
1	0	0	Set
X	1	X	Reset
X	X	1	Reset

1) In single mode only. In continuous mode, the end of a DMA transaction has no impact.

DMA channel mode

The DMA channel mode bit CHCFGRc.CHMODE controls the DMA channel mode as follows:

- Single mode (DMA channel CHCFGRc.CHMODE = 0_B)
  - DMA hardware request is disabled by hardware at the start of the last DMA transfer of a DMA transaction
- Continuous mode (DMA channel CHCFGRc.CHMODE = 1_B)
  - DMA hardware request is not disabled by hardware on completion of a DMA transaction

Single mode

The following DMA channel configuration is required to initiate one complete DMA transaction under hardware control in single mode:

- DMA channel CHCFGRc.CHMODE = 0_B
- DMA channel CHCFGRc.RROAT = 1_B
- DMA channel CHCFGRc.PRSEL = 0_B
- DMA channel TSRc.ECH = 1_B

Setting DMA channel TSRc.ECH to 1_B enables a DMA hardware request (TSRc.HTRE = 1_B) for the DMA channel. When the ICU generates a DMA hardware request, TSRc.CH is set high. If the DMA channel wins channel arbitration then the DMA channel transitions to the active state. The value of CHCFGRc.TREL is loaded into the MEmCHSR.TCOUNT and the DMA transaction is started by executing the first DMA transfer. After each DMA transfer, MEmCHSR.TCOUNT is decremented and the next source and destination addresses are calculated. At the start of the last DMA transfer of a DMA transaction, further DMA hardware requests are disabled and TSRc.HTRE is reset. In order to start a new hardware-controlled DMA transaction, a DMA hardware request must be enabled again by software writing TSRc.ECH = 1_B to set TSRc.HTRE.

The following DMA channel configuration is required to initiate each single DMA transfer under hardware control in single mode:

- DMA channel CHCFGRc.CHMODE = 0_B
- DMA channel CHCFGRc.RROAT = 0_B
- DMA channel CHCFGRc.PRSEL = 0_B
- DMA channel TSRc.ECH = 1_B

In this DMA channel configuration, TSRc.CH is cleared at the start of each DMA transfer and a new DMA hardware request must be generated to start the next DMA transfer.

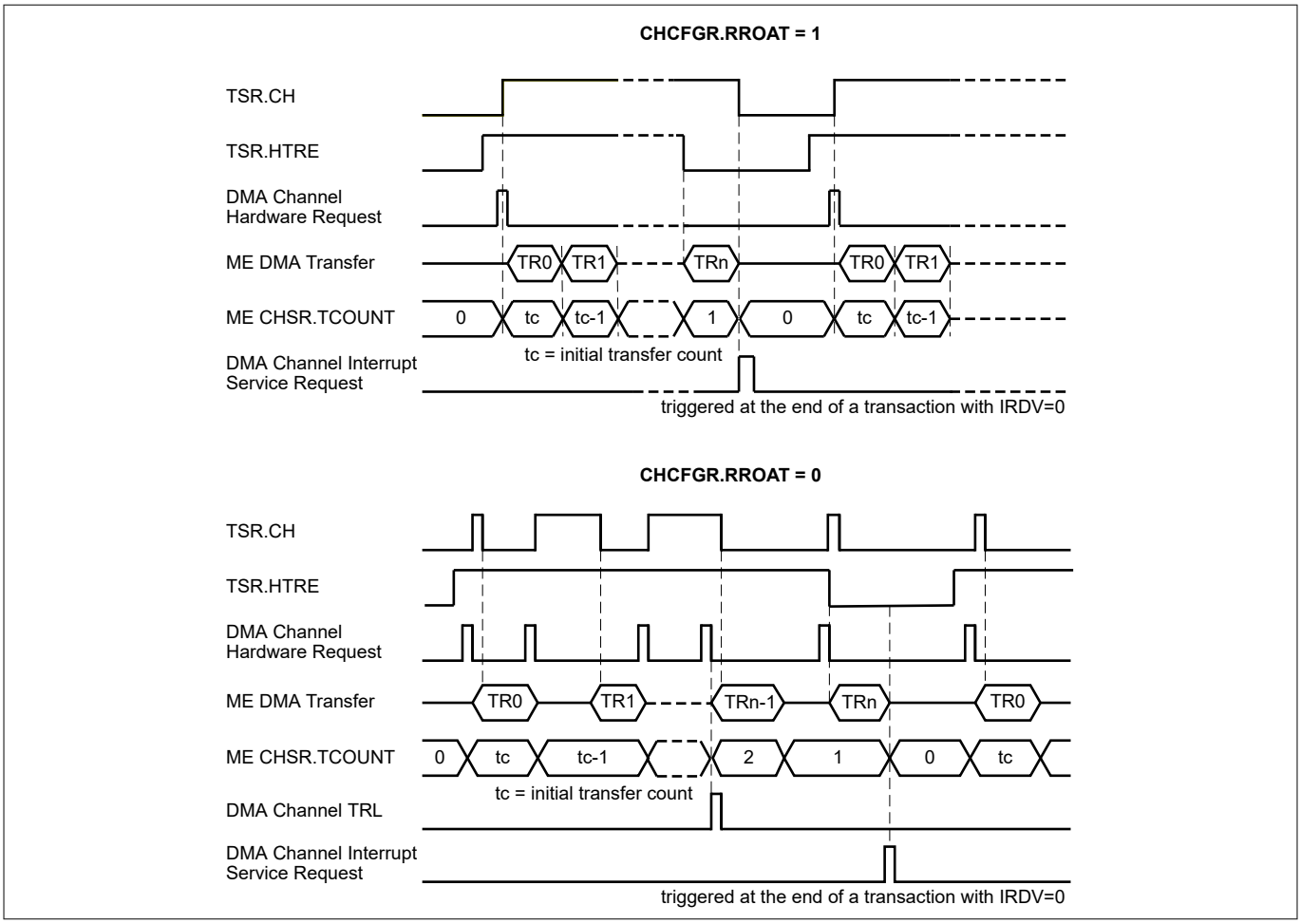


Figure 111 Hardware control in single mode

Continuous mode

If the DMA channel is configured for continuous mode (CHCFGRc.CHMODE = 1_B), then TSRc.HTRE is not reset at the start of the last DMA transfer in a DMA transaction. On completion of the current DMA transaction (MEmCHSR.TCOUNT = 0_D) each new DMA hardware request will start a new DMA transaction with the stored DMA channel TCS.

Related information

DMA hardware request on page 1257

9.3.3.2.4 Combined DMA software request and DMA hardware request

A DMA channel may be operated under combined software and hardware control. The example in the figure below demonstrates combined control:

- The first DMA transfer is triggered by a DMA software request setting the DMA channel CHCSRc.SCH = 1_B
- A DMA hardware request is still disabled (DMA channel TSRc.HTRE = 0_B)
- Software enables a DMA hardware request by setting the DMA channel TSRc.ECH = 1_B
- Each subsequent DMA transfer is triggered by a DMA hardware request from the ICU

In the example, the DMA channel operates in single mode (DMA channel CHCFGRc.CHMODE = 0_B). In single mode, the DMA channel TSRc.HTRE is reset by hardware at the start of the last DMA transfer in the DMA transaction.

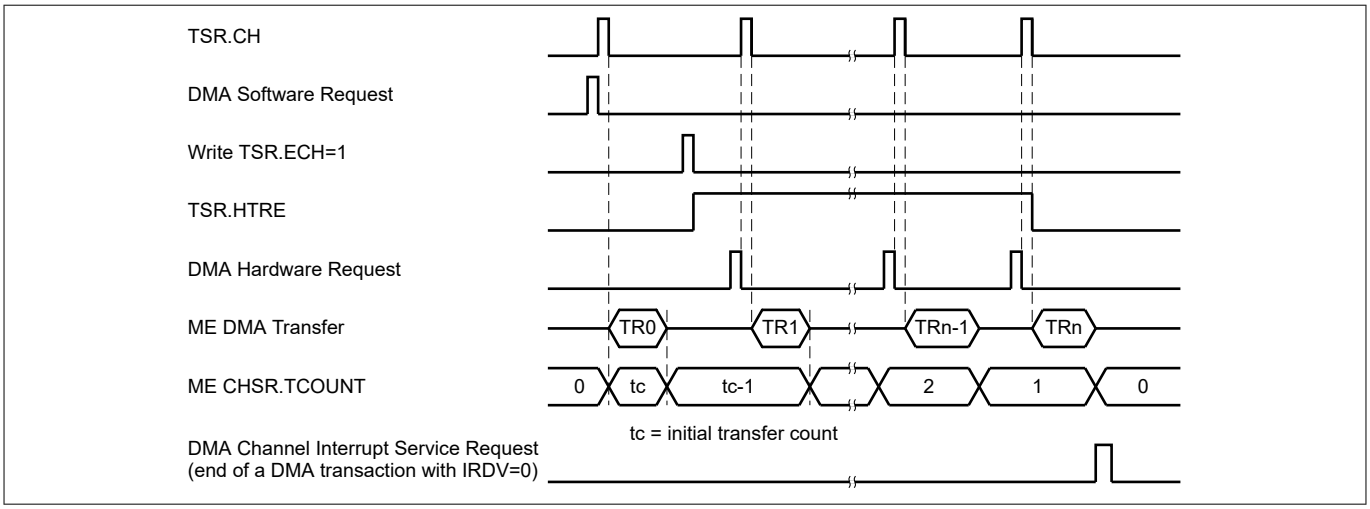


Figure 112 Transaction start by software, continuation by hardware

Transaction request lost (TRL)

When a DMA channel is configured to be triggered by parallel hardware and software requests, if a DMA software request and a DMA hardware request collide in the same clock cycle then a CBTRLER event is flagged.

Related information

DMA hardware request on page 1257
DMA software request on page 1256
DMA channel TRL error interrupt service request on page 1276

9.3.3.2.5 DMA daisy chain request

DMA channels are configured for DMA daisy chain request by setting DMA channel CHCFGRc.PRSEL.

Attention*: *DMA daisy chain requests are considered deprecated functionality in the AURIX™ TC4xx family. This functionality may be removed in a future AURIX™ generation.

When a higher priority DMA channel completes a DMA transaction it will initiate a DMA transaction in the next lower priority DMA channel by setting the channel request pending bit (TSRc.CH). DMA daisy chain request is limited to a higher priority DMA channel initiating a DMA request in the next lower priority DMA channel. The DMA daisy chain request from a higher priority channel will not trigger the next lower priority channel if the lower priority channel is assigned to a different resource partition or no resource partition and will instead set TSRc.TRL in the higher priority channel as well as set ERRSRr.DCTRLER in the RP that the higher priority channel belongs to. If TSR.ETRL is 1_B in the higher priority channel, an RP error interrupt will also be signaled.

Enabling the daisy chain disables the DMA channel interrupt service request trigger in the higher priority DMA channel. In a typical DMA daisy chain application only the lowest priority DMA channel in the daisy chain is required to generate a DMA channel interrupt service request. When the sequence of DMA transactions from the highest to lowest priority DMA channel has completed then the lowest priority DMA channel in the daisy chain can be configured to generate a DMA channel interrupt service request to signal the end of the DMA operation. DMA channel 0 will never trigger a DMA daisy chain request and the PRSEL setting will be ignored.

If DMA channels are configured in a daisy chain, a DMA transfer or DMA transaction in the highest priority DMA channel is initiated by a DMA software request or a DMA hardware request. DMA transactions in the lower priority DMA channels are triggered by a DMA daisy chain request in order to improve DMA latency.

***Note**: Software should ensure that a DMA channel configured to trigger the next lower priority DMA channel through a daisy chain request does not receive a new DMA request before the previous execution of the daisy chain is complete. If such a DMA request is generated, it is possible that the lower priority DMA channels in the daisy chain do not complete their pending DMA requests.*

Related information

DMA channel TRL error interrupt service request on page 1276
DMA channel interrupt service request on page 1267
DMA hardware request on page 1257
DMA software request on page 1256
TC4Dx DMA daisy chain request on page 1335

9.3.3.2.6 DMA auto start request

If a DMA auto start request is selected in the next TCS (CHCSRc.SCH = 1_B) during a linked list operation and the next TCS is configured for linked list operation, the request pending bit TSRc.CH for the DMA channel is set. The DMA then performs a new DMA request arbitration sequence to determine the highest priority DMA request. A DMA auto start request bypasses the Interrupt Router so reducing the cumulative latency over a number of DMA transactions.

Related information

DMA request arbitration on page 1261

9.3.3.3 DMA request arbitration

The DMA arbiter continuously monitors all DMA channels for a pending DMA request when the DMA channel is not in the halt state (TSRc.HLTACK = 1_B) or the suspend state (SUSACRc.SUSAC = 1_B).

The highest number DMA channel with a pending DMA request wins the DMA channel arbitration. The DMA channel which wins the DMA channel arbitration is loaded onto the highest numbered ME which is not busy servicing other DMA channels. If both MEs are busy, the DMA channel which has won the DMA channel arbitration waits for one of the MEs to complete an ongoing DMA transfer. If the DMA channel which has won the DMA channel arbitration has a higher priority than the DMA channel being serviced on the ME, the ME stops servicing the lower priority DMA channel at the end of the ongoing DMA transfer and after any required interrupts have been signaled. The ME then reads the new DMA channel TCS from the DMARAM, loads the TCS into the ME active channel register set and executes a DMA transfer.

To prevent a lower ASIL task from interfering (for example delayed DMA transaction) with a higher ASIL task, the system configuration should assign the higher ASIL tasks to higher number DMA channels.

Arbitration sequence

- Rule 1: If there is a higher priority DMA channel with a pending DMA request, then:
  - Rule 1.1: An ME which is idle or running the lowest priority DMA channel completes any ongoing DMA transfer and then writes back the active DMA channel updated TCS to the DMARAM
  - Rule 1.2: The ME reads the higher priority DMA channel TCS from the DMARAM to the ME
  - Rule 1.3: The ME executes a DMA transfer
  - Rule 1.4: On completion of the DMA transfer, the DMA performs an arbitration sequence
- Rule 2: If there is not a higher priority DMA channel pending DMA request and for the ME active DMA channel CHCFGRc.RROAT = 0_B, then:
  - Rule 2.1: The updated TCS is written back from the ME to the DMARAM
  - Rule 2.2: During each clock cycle the DMA performs an arbitration sequence

  - Rule 2.3: If a DMA Request is received then the ME executes a DMA transfer
  - Rule 2.4: On completion of the DMA transfer, the DMA performs an arbitration sequence
- Rule 3: If there is not a higher priority DMA channel pending DMA request and for the ME active DMA channel CHCFGRc.RROAT = 1_B, then:
  - Rule 3.1: The ME executes a DMA transfer
  - Rule 3.2: On completion of the DMA transfer, the DMA performs an arbitration sequence
- Rule 4: On completion of a DMA transfer, if the ME sets the active DMA channel MEmCHSR.ICH, MEmCHSR.WRPS or MEmCHSR.WRPD bits, the updated TCS is written back from the ME to the DMARAM
- Rule 5: On completion of the last DMA transfer, the DMA channel request bit is cleared (TSRc.CH = 0_B)
- Rule 6: The arbitration sequence continues until all DMA channels with pending DMA requests are serviced by an ME

Related information

Write completion signaling on page 1269

9.3.3.4 DMA channel states

The table below describes the possible DMA channel states.

Table 636 DMA channel states

State	Status	Comments
Idle state	DMA channel TSRc.CH = 0_B	No DMA request is pending in this DMA channel
Reset state	DMA channel TSRc.CH = 0_B	No DMA request is pending in this DMA channel. DMA channel TCS bits cleared
Halt state	DMA channel TSRc.HLTACK = 1_B	The DMA channel is halted by software configuration of the DMA channel registers. A DMA request will not be serviced by a ME
Suspend state	DMA channel SUSACRc.SUSAC = 1_B	The DMA channel is suspended by the debug logic. A DMA request will not be serviced by a ME
Pending state	DMA channel TSRc.CH = 1_B	A DMA request in this DMA channel is waiting to be serviced by a ME
Active state		The ME is servicing a DMA request in this DMA channel

Related information

DMA channel reset on page 1262
DMA channel halt and resume on page 1263
DMA channel suspend on page 1330

9.3.3.4.1 DMA channel reset

Software shall reset an individual DMA channel by setting the DMA channel reset bit (TSRc.RST = 1_B). When a DMA channel reset is applied to a DMA channel, the transition to the reset state (TSRc.RST = 0_B) is as follows:

- Idle state, halt state, suspend state and pending state: The DMA channel transitions to the reset state
- Active state: On completion of the current DMA transfer, the ME writes back the updated DMA channel TCS to the DMARAM. The DMA channel then transitions to the reset state

Reset state

On completion of a DMA channel reset the DMA channel enters the reset state defined as:

- The following DMA channel bits are reset:
  - DMA transaction state register: TSRc.HLTREQ, TSRc.HLTACK, TSRc.HTRE, TSRc.CH and TSRc.TRL
  - DMA channel TCS: CHCFGRc.PRSEL, CHCSRc.ICH, CHCSRc.IPM, CHCSRc.WRPD, CHCSRc.WRPS, CHCSRc.FROZEN, CHCSRc.BUFFER, CHCSRc.LXO and CHCSRc.TCOUNT
- If a circular buffer ADICRc.SCBE is enabled for the DMA channel, the source address register will be set to the wrap boundary else the source address register is cleared
- If a circular buffer ADICRc.DCBE is enabled for the DMA channel, the destination address register will be set to the wrap boundary else the destination address register is cleared
- DMA channel shadow address register (SHADRc) is cleared

Resetting a DMA channel

A user program must execute the following steps to reset a DMA channel:

1. If a DMA hardware request is enabled, disable hardware requests (TSRc.DCH = 1_B)
2. Software requests a DMA channel reset (TSRc.RST = 1_B)
3. Software shall monitor the DMA channel reset and the DMA channel SADRc, DADRc and SHADRc registers
4. As soon as the DMA has cleared the DMA channel reset (TSRc.RST = 0_B) and the DMA has reset the DMA channel SADRc, DADRc and SHADRc registers, the DMA channel reset has completed

During a DMA channel reset operation, a DMA software request must not be initiated by setting CHCSRc.SCH = 1_B.

Restarting a DMA channel

A user program must execute the following steps to restart a DMA channel after a DMA channel reset:

1. Configure the DMA channel TCS including:
   a. DMA double buffering: Program address pointers to the base address
2. The DMA channel is restarted as follows:
   a. DMA software request: Software initiates a DMA request (DMA channel CHCSRc.SCH = 1_B)
   b. DMA hardware request: Software enables a DMA request (DMA channel TSRc.ECH = 1_B)

Related information

DMA channel states on page 1262
DMA hardware request on page 1257
DMA software request on page 1256

9.3.3.4.2 DMA channel halt and resume

A DMA channel may be halted during a DMA transaction and the state frozen to allow a background RAM test to be run over a destination memory in order to detect stuck bits and distinguish between static errors and transient errors. On completion of the RAM test the DMA channel may be re-started and the DMA transaction completed.

The DMA channel halt logic shown below utilizes a set and clear mechanism to request the DMA channel transitions to and from the HALT state on completion of the current DMA transfer. Only writing a logic ‘1’ to set or clear a halt request to a DMA channel has an effect. The status of other DMA channels is ignored.

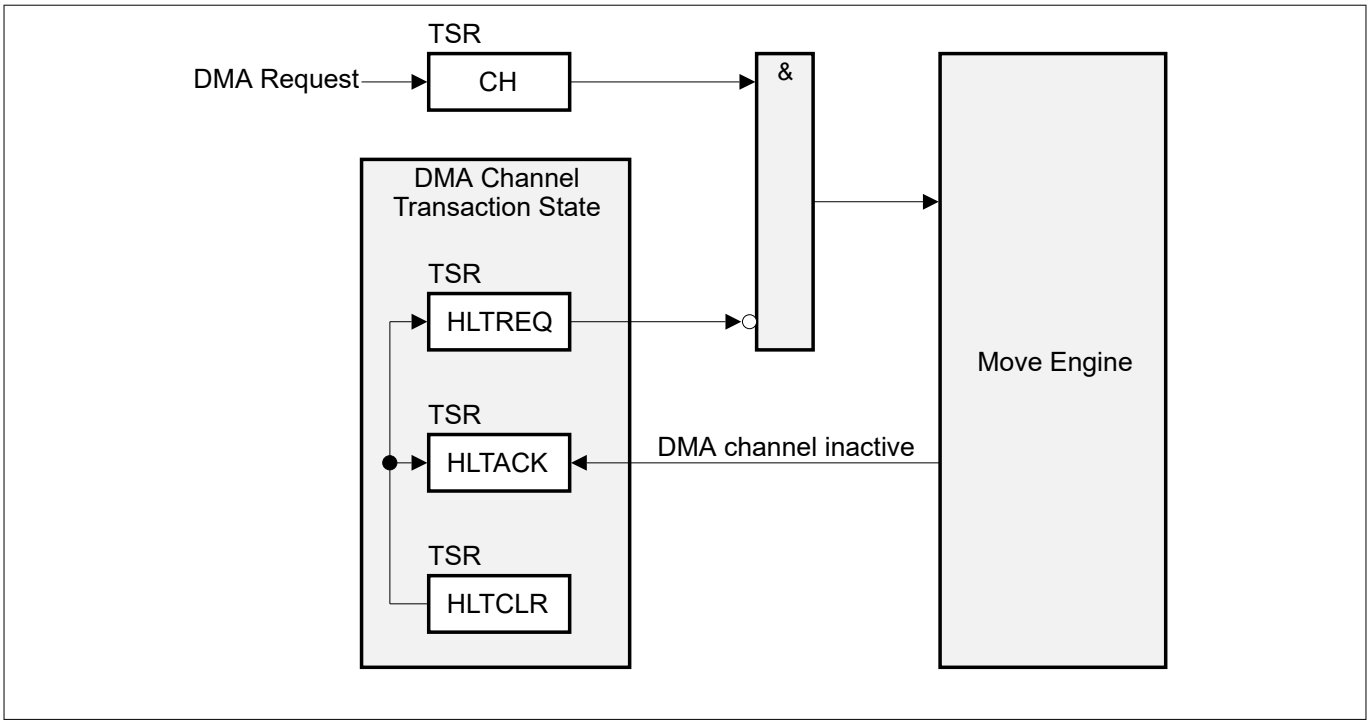


Figure 113 DMA channel halt logic

Halt state

A DMA channel is defined to be in the halt state when DMA channel TSRc.HLTACK = 1_B.

Entering DMA channel halt

A DMA channel is halted by software writing the DMA channel halt request bit (TSRc.HLTREQ = 1_B). The DMA channel enters the halt state as follows:

- Idle state, reset state and pending state: As soon as the DMA channel receives the halt request, the DMA channel transitions to the halt state
- Active state: On completion of the current DMA transfer, the DMA channel transitions to the halt state

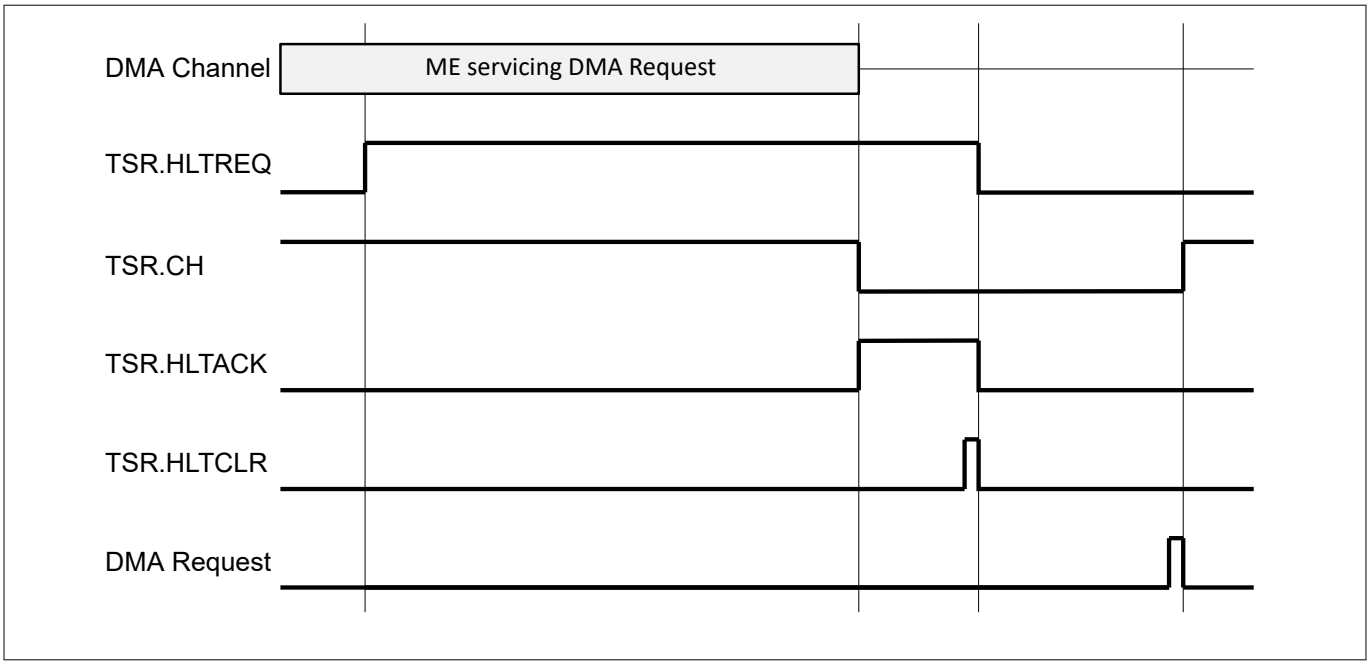


Figure 114 DMA channel halt operation

Exiting DMA channel halt

A DMA channel is released from the halt state by software writing the DMA channel halt clear bit (TSRc.HLTCLR = 1_B). The DMA operation is resumed. If the halt request is cleared before it is acknowledged then there is no effect on DMA operation.

DMA channel hardware request during DMA channel halt

If a DMA channel is in the halt state and Hardware Transaction Request is enabled (TSRc.HTRE = 1_B) then the DMA channel will respond to a DMA hardware request as follows:

- No DMA request pending (TSRc.CH = 0_B): The DMA channel sets the access pending bit (TSRc.CH = 1_B). The DMA hardware request is serviced when DMA channel exits the halt state
- DMA request pending (TSRc.CH = 1_B): The DMA channel records a TRL event. If the DMA channel TRL enable bit is set (TSRc.ETRL = 1_B), the DMA triggers a DMA RP error interrupt service request

Related information

DMA channel states on page 1262
DMA hardware request on page 1257
DMA RP error interrupt service request on page 1275

9.3.3.5 DMA channel updates

Software shall only configure the DMA channel TCS when the DMA channel is in the idle state (with DMA channel CHCSRc.TCOUNT=0_D), suspend state or reset state. If the DMA channel is in the idle state (with DMA channel CHCSRc.TCOUNT!=0_D), pending state or active state, software shall not update the DMA channel TCS. The following DMA channel update exceptions apply for Shadow operations and Double buffering operations.

Related information

DMA channel states on page 1262
DMA channel reset on page 1262
Shadow operation on page 1230

Double buffering operations on page 1233

9.3.3.5.1 Shadow operations

If the DMA channel TCS is configured for Shadow operation and the DMA channel is in the idle state (DMA channel CHCSRc.TCOUNT!=0_D), pending state or active state, software shall be to updating the DMA channel TCS words marked 'X':

Table 637 DMA channel TCS updates during shadow operation

SHCT	Description	CHCSRc	SHADRc	CHCFGRc	ADICRc	DADRc	SADRc	SDCRCRc	RDCRCRc
0001_B	Shadow operation read only mode source address						X		
0010_B	Shadow operation read only mode destination address					X			
0101_B	Shadow operation direct write mode source address		X						
0110_B	Shadow operation direct write mode destination address		X						

Related information

Shadow operation on page 1230

DMA channel states on page 1262

9.3.3.5.2 Double buffering operations

If the DMA channel TCS is configured for Double buffering operations and the DMA channel is in the idle state(DMA channel CHCSRc.TCOUNT!=0_D), pending state or active state, software is to updating the DMA channel TCS words marked 'X':

Table 638 DMA channel TCS updates during double buffering operations

SHCT	Description	CHCSRc	SHADRc	CHCFGRc	ADICRc	DADRc	SADRc	SDCRCRc	RDCRCRc
1000_B	DMA double source buffering Software switch only - DMA channel CHCSRc.BUFFER = 0_B	X	X						
1000_B	DMA double source buffering Software switch only - DMA channel CHCSRc.BUFFER = 1_B	X					X		
1001_B	DMA double source buffering Software switch and Automatic hardware switch - DMA channel CHCSRc.BUFFER = 0_B	X	X						
1001_B	DMA double source buffering Software switch and Automatic hardware switch - DMA channel CHCSRc.BUFFER = 1_B	X					X		
1010_B	DMA double destination buffering Software switch only - DMA channel CHCSRc.BUFFER = 0_B	X	X						
1010_B	DMA double destination buffering Software switch only - DMA channel CHCSRc.BUFFER = 1_B	X				X			

(table continues...)

Table 638 (continued) DMA channel TCS updates during double buffering operations

SHCT	Description	CHCSRc	SHADRc	CHCFGRc	ADICRc	DADRc	SADRc	SDCRCRc	RDCRCRc
1011_B	DMA double destination buffering Software switch and Automatic hardware switch - DMA channel CHCSRc.BUFFER = 0_B	X	X						
1011_B	DMA double destination buffering Software switch and Automatic hardware switch - DMA channel CHCSRc.BUFFER = 1_B	X				X			

Related information

Double buffering operations on page 1233
DMA channel states on page 1262
DMA double source buffering on page 1234
Software switch on page 1236
Automatic hardware switch on page 1236
DMA double destination buffering on page 1234

9.3.3.6 DMA channel reconfiguration

If a DMA channel is to be configured, software shall apply a DMA channel reset to initialize the DMA channel. After the DMA channel has entered reset state, software shall configure the DMA channel for a DMA operation.

Related information

DMA channel reset on page 1262

9.3.3.7 DMA channel interrupt service request

Each DMA channel generates an interrupt service request (CHc) to report the following DMA channel events:

- DMA channel transfer interrupt service request (ICH)
- DMA channel pattern match interrupt service request (IPM)
- DMA channel wrap buffer interrupt service request (WRPS and WRPD)

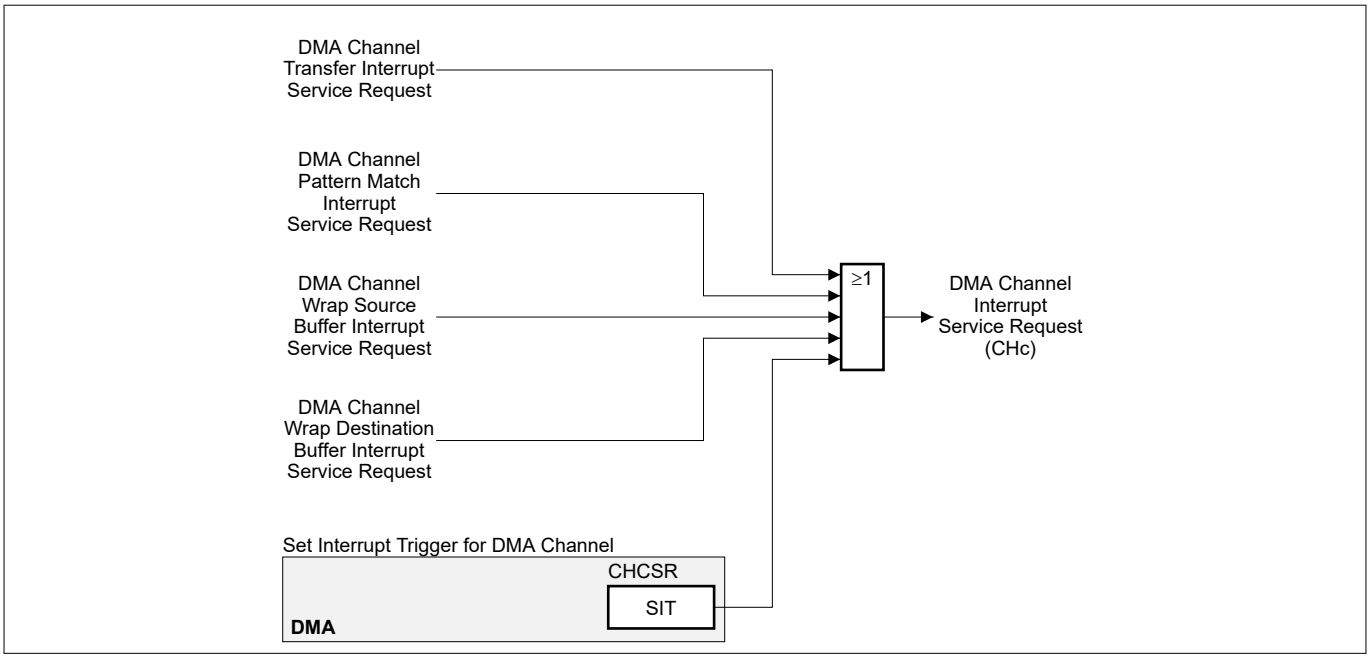


Figure 115 DMA channel interrupt service request

Related information

DMA channel transfer interrupt service request on page 1268
DMA channel pattern match interrupt service request on page 1270
DMA channel wrap buffer interrupt service request on page 1270

9.3.3.7.1 DMA channel transfer interrupt service request

If a DMA channel is in the active state then a DMA channel transfer interrupt service request may be activated on completion of a DMA transfer, or when MEmCHSR.TCOUNT matches with the value of bit-field ADICRc.IRDV after it has been decremented after a DMA transfer. An interrupt service request from a DMA channel is indicated when status flag CHCSRc.ICH is set. The status flag is reset by software setting DMA channel CHCSRc.CICH = 1_B (or TSRc.RST = 1_B). The DMA channel interrupt service request is enabled when DMA channel ADICRc.INTCT[1] is set.

Bit ADICRc.INTCT[0] selects one of two types of interrupt source. For the compare operation, bit-field ADICRc.IRDV (4-bit) is zero-extended to 14-bit and then compared with the 14-bit TCOUNT value. This means that a TCOUNT match interrupt may be generated after one of the last 16 DMA transfers of a DMA transaction. Note that with IRDV = 0000_B, the match interrupt is generated at the end of a DMA transaction (after the last DMA transfer).

The DMA channel transfer interrupt service request may also be activated or deactivated during a linked list load operation. The DMA CHCSRc.ICH bit is set if the ICH bit in the new TCS is set when the CICH bit in the new TCS is clear. The DMA CHCSRc.ICH bit is cleared if the ICH bit in the new TCS is clear (irrespective of the CICH bit in the new TCS). If both the ICH and CICH bits in the new TCS are set, the TCS load does not affect the DMA CHCSRc.ICH bit and the current value is retained as-is. If the linked list load operation causes the DMA CHCSRc.ICH bit to be set, the DMA Channel transfer interrupt service request is activated depending on the value of CHCFGRc.PRSEL and ADICRc.INTCT[1] from the new TCS.

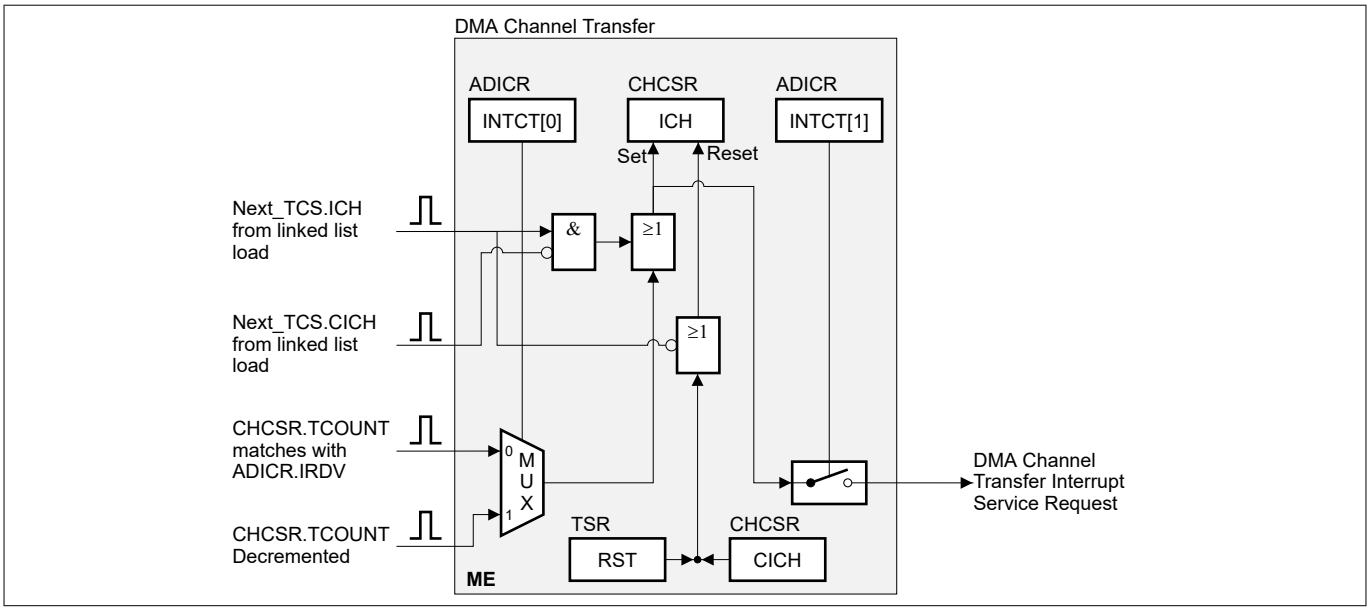


Figure 116 DMA channel transfer interrupt service request

Related information

DMA channel states on page 1262
Write completion signaling on page 1269
TC4Dx DMA channel transfer interrupt service request on page 1335

Write completion signaling

When the application software is operating in a model where the destination location has to be accessed immediately after the DMA channel transfer interrupt service request is received and there is no higher level coordination between producer and consumer, it may be necessary to ensure that the DMA write moves have reached their final destination before the DMA channel transfer interrupt service request is activated.

When TSRc.WWC is set to 0_B, the DMA channel activates the DMA channel transfer interrupt service request as soon as it has issued the write data for the last DMA write move of the DMA transfer or transaction as configured by ADICRc.INTCT. When TSRc.WWC is set to 1_B, the DMA channel waits for write completion signaling from the system interconnect to ensure that all previous writes from the DMA channel have reached their final destination before updating the CHCSRc.ICH bit and if enabled in ADICRc.INTCT, activating the DMA channel transfer interrupt service request.

***Note**: Software shall only change the value of the TSRc.WWC bit when the DMA channel is in the idle, reset, halted or suspend states.*

The system interconnect offers write completion signaling independently for each system interconnect master tag identifier. A master agent on the system interconnect shall not issue further writes onto the system interconnect using the same master tag identifier while it is waiting for the write completion signal for that master tag identifier. When TSRc.WWC is set to 1_B and while a DMA channel is waiting for the write completion signaling from the system interconnect, a different DMA channel that wins DMA request arbitration may begin execution in the DMA move engine. However, the new DMA channel cannot start a DMA write move if it uses the same master tag identifier as another DMA channel (regardless of its priority) which is waiting for a write completion signal.

***Note**: Because the granularity of write completion signaling is for a given master tag identifier, there is the possibility of temporal interference between different DMA channels using the same master tag identifier if even one of them uses the write completion signaling feature. Software should therefore restrict the usage of DMA channels with TSRc.WWC set to 1_B and instead prefer the usage of higher level coordination between the producer and consumer where possible.*

Related information

DMA channel transfer interrupt service request on page 1268
DMA request arbitration on page 1261

9.3.3.7.2 DMA channel pattern match interrupt service request

The DMA channel pattern detection interrupt service request is enabled when CHCFGRc.PATSEL[1:0] is not equal to 00_B. The pattern detection interrupt is indicated when status flag CHCSRc.IPM is set. The status flag CHCSRc.IPM is reset by software setting bit DMA channel CHCSRc.CICH = 1_B (or TSRc.RST = 1_B).

The DMA CHCSRc.IPM bit is cleared by a linked list load operation since a pattern match operation cannot be combined with a linked list operation.

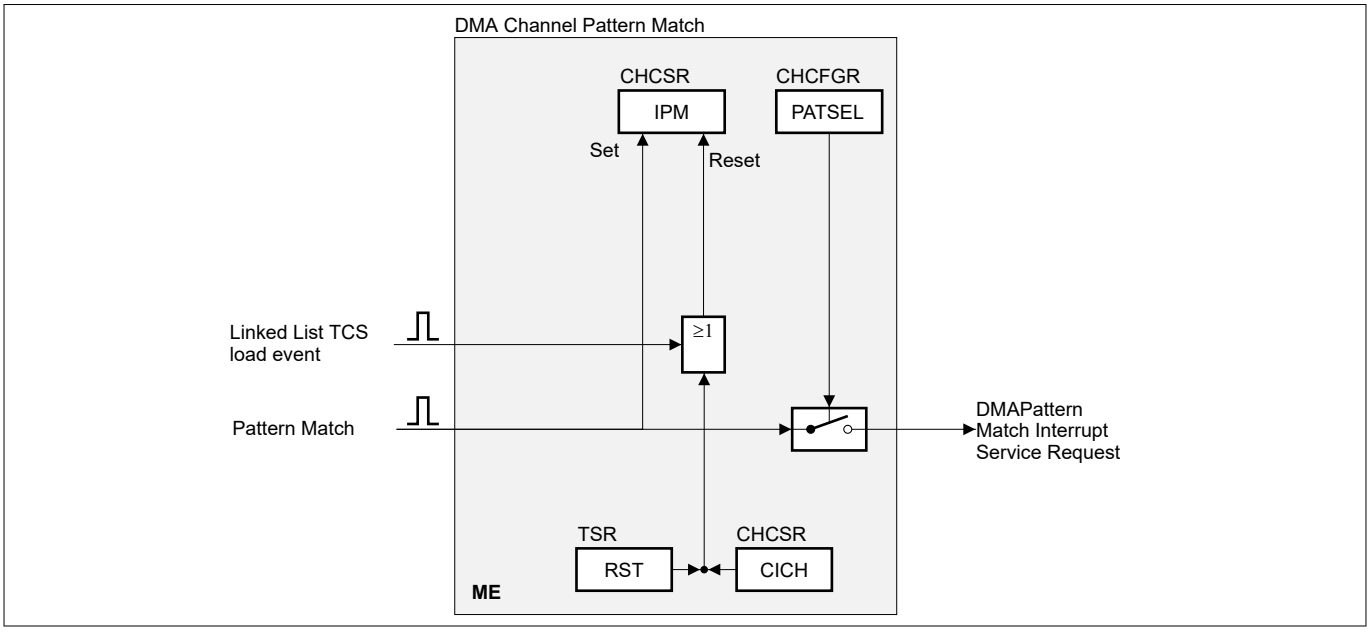


Figure 117 DMA channel pattern match interrupt service request

9.3.3.7.3 DMA channel wrap buffer interrupt service request

Each DMA channel reports a source buffer wraparound or a destination buffer wraparound as follows:

- A wrap source buffer wraparound is indicated by status flag CHCSRc.WRPS
- A wrap destination buffer wraparound is indicated by the status flag CHCSRc.WRPD

Software may reset both buffer wraparound status flags by setting CHCSRc.CWRP = 1_B (or TSRc.RST = 1_B).

A DMA channel wrap buffer interrupt service request is enabled as follows:

- A DMA channel wrap source buffer interrupt service request is enabled when bit ADICRc.WRPSE is set
- A DMA channel wrap destination buffer interrupt service request is enabled when bit ADICRc.WRPDE is set

The DMA channel wrap source or destination interrupt service request may also be activated or deactivated during a linked list load operation.

- The DMA CHCSRc.WRPS and CHCSRc.WRPD bits are set if the WRPS or WRPD bits respectively in the new TCS are set when the CWRP bit in the new TCS is clear
- The DMA CHCSRc.WRPS and CHCSRc.WRPD bits are cleared if the WRPS or WRPD bits respectively in the new TCS are clear (irrespective of the CWRP bit in the new TCS)
- If both the WRPS and CWRP bits in the new TCS are set, the TCS load does not affect the DMA CHCSRc.WRPS bit and the current value is retained as-is
- If both the WRPD and CWRP bits in the new TCS are set, the TCS load does not affect the DMA CHCSRc.WRPD bit and the current value is retained as-is
- If the linked list load operation causes the DMA CHCSRc.WRPS bit to be set, a DMA channel wrap source buffer interrupt service request is generated depending on the values of ADICRc.WRPSE from the new TCS
- If the linked list load operation causes the DMA CHCSRc.WRPD bit to be set, a DMA channel wrap source buffer interrupt service request is generated depending on the values of ADICRc.WRPDE from the new TCS

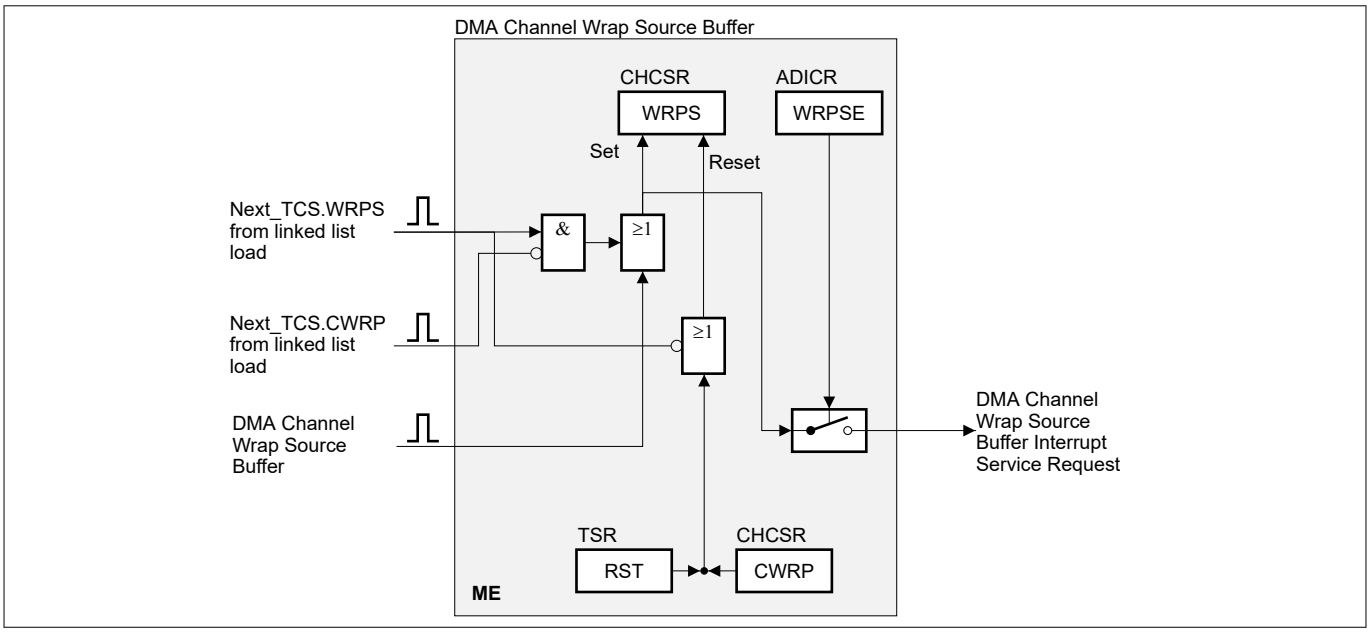


Figure 118 DMA channel wrap source buffer interrupt service request

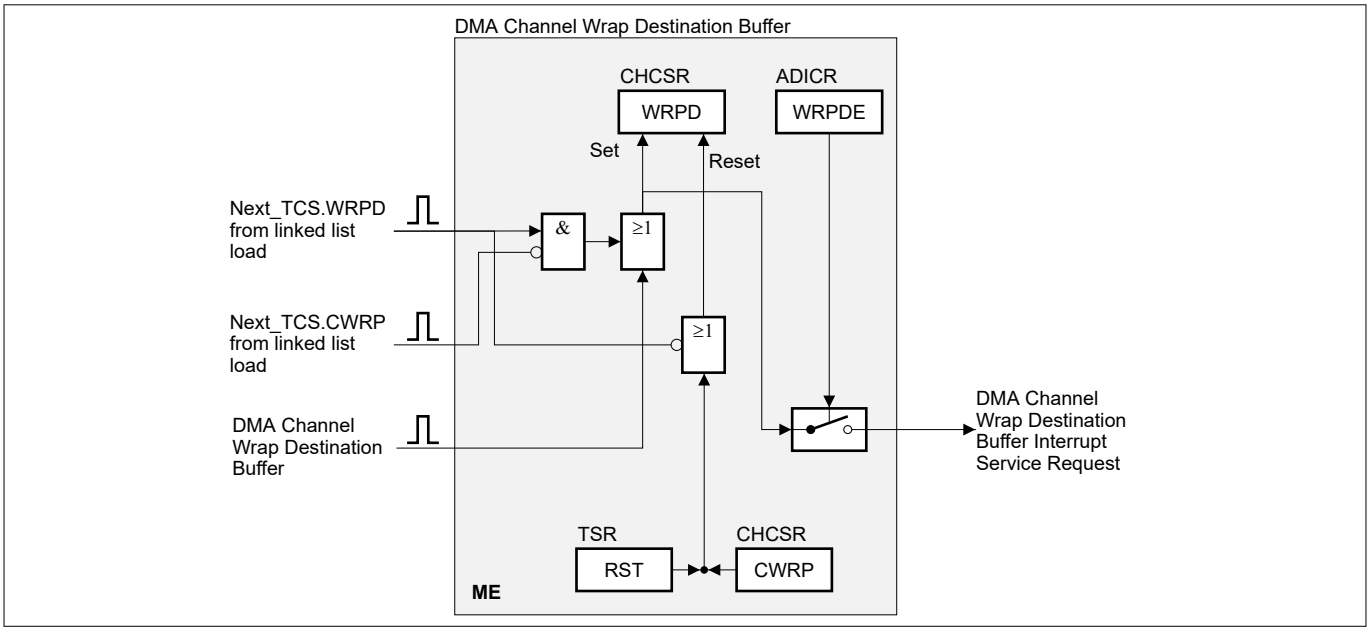


Figure 119 DMA channel wrap destination buffer interrupt service request

9.3.4 Resource partitions

Resource partitions (RPs) are used to group sets of DMA channels together.

Each resource partition may be assigned to a software application or a set of software applications using the PROTRPr and ACCENRPr registers. Each DMA channel may then be assigned to a resource partition using the HRRc register.

Once assigned and configured, the RP access protection mechanisms can be used to isolate the set of DMA channels assigned to that RP from other software applications using other RPs. The DMA channels assigned to an RP also share certain common configuration parameters like those in the MODEr register, including RP TAG-ID assignment.

9.3.4.1 Access protection

Each RP has its own RP PROT mechanism (PROTRPr) and RP APU (ACCENRPr). The RP PROT mechanism provides write protection to the RP APU registers, the channel assignment registers (HRRc) and the channel suspend control registers (SUSENRc). The RP APU in turn provides read and write protection to all the RP specific registers as well as all other channel specific registers for channels that are assigned to that RP.

Software shall use an appropriate synchronization mechanism such as a read-back or synchronization instruction after updating the contents of the PROTRPr or ACCENRPr registers in order to ensure that subsequent accesses to the protected registers use the updated access protection configuration.

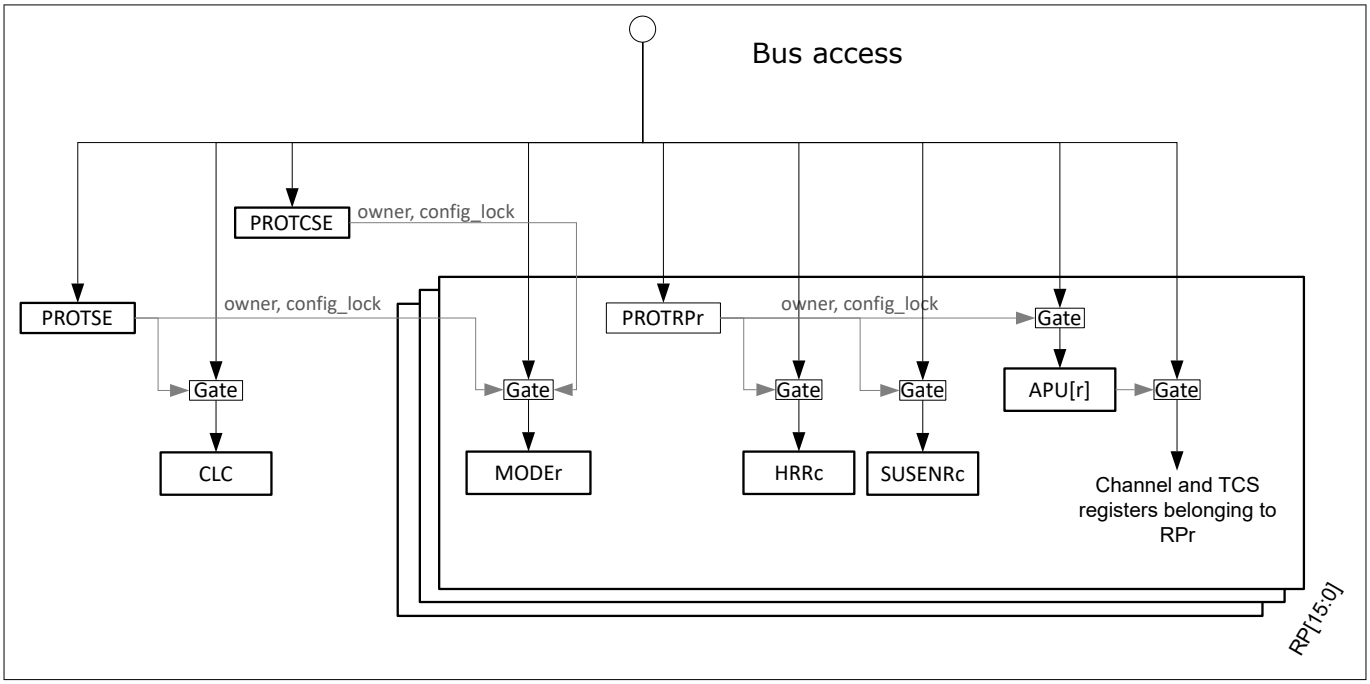


Figure 120 DMA access protection

The system configuration should define the access protection of each RP to prevent a lower ASIL application from issuing a DMA software request, DMA channel reset or DMA channel halt to a DMA channel assigned to a higher ASIL application.

9.3.4.2 Resource partition configuration

Each RP is configured with a master agent function identifier consisting of a TAG ID from the group of DMA TAG IDs and optionally with VM and PRS information. The DMA channels assigned to an RP use this master agent function identifier for all DMA read or write requests on the system interconnect.

Each DMA instance in a device is provided up to eight unique TAG-IDs as listed in the system interconnect chapter. Each RP in each DMA instance can be configured with one of these eight TAG-IDs by configuring the MODEr.TAGOFF bit-field. Each RP can also be configured with VM and PRS information using the MODEr.VMEN, MODEr.VM, MODEr.PRSEN and MODEr.PRS bit-fields.

The MODEr.MODE bit selects if a DMA move system interconnect access is made in supervisor mode or user mode.

The MODEr.CS bit selects if the RP is marked as a CS or a non-CS RP. It restricts hardware requests to DMA channels in the RP to be triggered from the correct CS or non-CS applications. The CS bit is also used on every DMA read or write request to set the cs_valid bit on FPI interconnects and sri_sm bit on SRI interconnects to indicate whether the move has been triggered by a CS or non-CS application.

The system configuration should define the access protection and privilege level (supervisor or user mode) of source modules and destination modules and to isolate DMA moves between RPs.

Access protection of RP MODEr registers

The RP MODEr registers define the tags that DMA channels in each RP present on the interconnect and therefore may have to be configured in a safe and secure manner. The PROTCSE owner can configure the resource partitions assigned to cybersecurity tasks, and the PROTSE owner can configure the resource partitions assigned to non-cybersecurity tasks. These two owners can independently assign and configure the MODE register for their respective resource partitions.

Each RP can be in one of three states:

1. Unassigned: When MODEr.VALID = 0_B, the RP is unassigned. Either the PROTSE or the PROTCSE owner can claim and configure the resource partition
2. Marked as a non-CS RP: When MODEr.VALID = 1_B and MODEr.CS = 0_B, the RP is marked as a non-CS RP. Only the enabled PROTSE owner can reconfigure the fields in the RP MODEr register
3. Marked as a CS RP: When MODEr.VALID = 1_B and MODEr.CS = 1_B, the RP is marked as a CS RP. Only the enabled PROTCSE owner can reconfigure the fields in the RP MODEr register

The table below shows the master agent functions which are permitted to write to the RP MODEr registers for different combinations of the current RP state and the new write data. Writes to the RP MODEr registers are only executed when the respective PROT state machine is in an unlocked state, and any writes which are not executed terminate with a bus error.

Table 639 Allowed master agents for write to RP MODEr register

PROTCSE.CSEN	Current MODEr.VALID	Current MODEr.CS	Wdata.VALID	Wdata.CS	Wdata.TAGOFF == PROTCSE.CSTAGOFF?	Allowed master agents for write
0	X	X	0	0	X	PROTSE owner[1)]
0	X	X	1	0	X	PROTSE owner[1)]
0	X	X	X	1	X	None
1	0	X	0	0	0	PROTSE owner[1)]
1	0	X	0	0	1	None
1	0	X	0	1	0	None
1	0	X	0	1	1	PROTCSE owner[2)]
1	0	X	1	0	0	PROTSE owner[1)]

(table continues...)

Table 639 (continued) Allowed master agents for write to RP MODEr register

PROTCSE.CSEN	Current MODEr.VALID	Current MODEr.CS	Wdata.VALID	Wdata.CS	Wdata.TAGOFF == PROTCSE.CSTAGOFF?	Allowed master agents for write
1	0	X	1	0	1	None
1	0	X	1	1	0	None
1	0	X	1	1	1	PROTCSE owner[2)]
1	1	0	0	0	0	PROTSE owner[1)]
1	1	0	0	0	1	None
1	1	0	0	1	X	None
1	1	0	1	0	0	PROTSE owner[1)]
1	1	0	1	0	1	None
1	1	0	1	1	X	None
1	1	1	0	X	X	PROTCSE owner[2)]
1	1	1	1	0	X	None
1	1	1	1	1	0	None
1	1	1	1	1	1	PROTCSE owner[2)]

1) When PROTSE.ODEF=0, any master (including the PROTCSE owner if one is defined) can behave as the PROTSE owner.
2) When PROTCSE.ODEF=0, any master (including the PROTSE owner if one is defined) can behave as the PROTCSE owner.

9.3.4.3 DMA channel assignment

After an application reset, DMA channels are not assigned to any resource partition and cannot be triggered. During DMA configuration, each used DMA channel can be assigned to a valid resource partition.

DMA channels are assigned to a resource partition using the HRRc register. The assignment of channels to resource partitions has to be achieved in a safe and secure manner. There are therefore a few restrictions on the assignment of channels to resource partitions.

The table below shows the master agent functions which are permitted to write to the HRRc registers for different combinations of the current HRRc state and the new write data. Writes to the HRRc registers are only executed when the respective PROT state machine is in an unlocked state, and any writes which are not executed terminate with a bus error.

Software shall use an appropriate synchronization mechanism such as a read-back or synchronization instruction after updating the contents of the HRRc registers in order to ensure that subsequent accesses to the respective DMA channel TSRc and DMA channel 'c' TCS registers use the updated access protection configuration.

Table 640 Allowed master agents for writes to HRRc register

Channel 'c' state active or pending?	TSRc.HTRE	Current HRRc.HRPV	Current HRRc.HRP	Wdata.HRPV	Wdata.HRP	Allowed master agents for write to HRRc
1	X	X	X	X	X	None
0	1	X	X	X	X	None
0	0	0	X	0	X	Any PROTRPr owner[1)]
0	0	0	X	1	'r'	PROTRPr owner[1) 2)]
0	0	1	'a'	1	'r'	• None, if 'r' is not equal to 'a' • PROTRPr owner[1)], if 'r' is equal to 'a'
0	0	1	'r'	0	X	PROTRPr owner[1)]

1) When PROTRPr.ODEF=0, any master agent can behave as the PROTRPr owner
2) The write is not executed if 'r' does not correspond to an implemented RP in the current DMA instance

The table below shows the master agents which are permitted to access the DMA channel TSRc and DMA channel 'c' TCS registers for different combinations of the current HRRc state. Any accesses which are not executed terminate with a bus error.

Table 641 Allowed master agents for accesses to DMA TSRc and DMA channel 'c' TCS registers

HRRc.HRPV	HRRc.HRP	Allowed master agents for accesses to DMA TSRc and DMA channel 'c' TCS registers
0	X	None
1	'r'	Master agents enabled for access in ACCENRPr* registers

9.3.4.4 DMA RP error interrupt service request

Each RP generates one error interrupt service request (ERRr) to cover all error events for DMA channels assigned to that RP including servicing of DMA requests by the ME. Each application software task using a DMA RP should include an error handler to resolve RP and DMA error conditions.

- DMA channel TRL error interrupt service request that can be attributed to an RP (CBTRLER, CDTRLER, DCTRLER, DBTRLER, IRTRLER)
- RP DMARAM integrity error interrupt service request
- RP SER and DER error interrupt service request
- RP linked list operation TCS load error interrupt service request

- RP CHKLL DMA address checksum error interrupt service request
- RP TCS configuration error interrupt service request

If a DMA RP error interrupt service request is triggered then the software RP application error handler shall read the contents of the error status registers to identify the origin of the error.

In the case of multiple errors occurring at the same time, the error bits are set according to the error conditions (that is more than one error flag may be set).If a previous RP error status flag is still set when a new error occurs that requires an update of the RP error status bits, the RP error status overflow bit ERRSRr.OVFLER is set. Software may clear the RP error status overflow bit by writing a 1_B to the CLREr.COVFLER bit-field.

For all error conditions, the DMA performs the following actions:

- The RP records the number of the DMA channel in the RP last error channel bit-field ERRSRr.LEC
- If the DMA channel is enabled for DMA channel hardware request then DMA channel TSRc.HTRE is cleared

Related information

DMA channel TRL error interrupt service request on page 1276

9.3.4.4.1 DMA channel TRL error interrupt service request

DMA channel TRL events that can be attributed to a certain resource partition occur for the following conditions.

- Channel Busy TRL Error (CBTRLER): If a DMA request is detected and the DMA channel TSRc.CH is already set or multiple DMA requests occur simultaneously on a DMA channel
- Channel Disabled TRL Error (CDTRLER): If a DMA hardware request is detected and the DMA channel is disabled for hardware requests
- Daisy Chain TRL Error (DCTRLER): If a DMA daisy chain request from the current channel attempts to trigger the next lower priority channel assigned to a different resource partition or no resource partition (the TRL event is processed on the current channel)
- Double Buffer TRL Error (DBTRLER): If a DMA channel is configured for a double buffering operation and one of the following conditions occur:
  - The DMA channel is configured for software switch only AND the DMA transaction is completed before the DMA receives a software switch
  - The DMA channel is configured for automatic hardware switch AND the DMA transaction is completed when CHCSRc.FROZEN is still 1_B
- Invalid Resource partition TRL Error (IRTRLER): If a DMA request is detected and the resource partition that the DMA channel is assigned to is marked invalid (MODEr.VALID is set to 0_B)

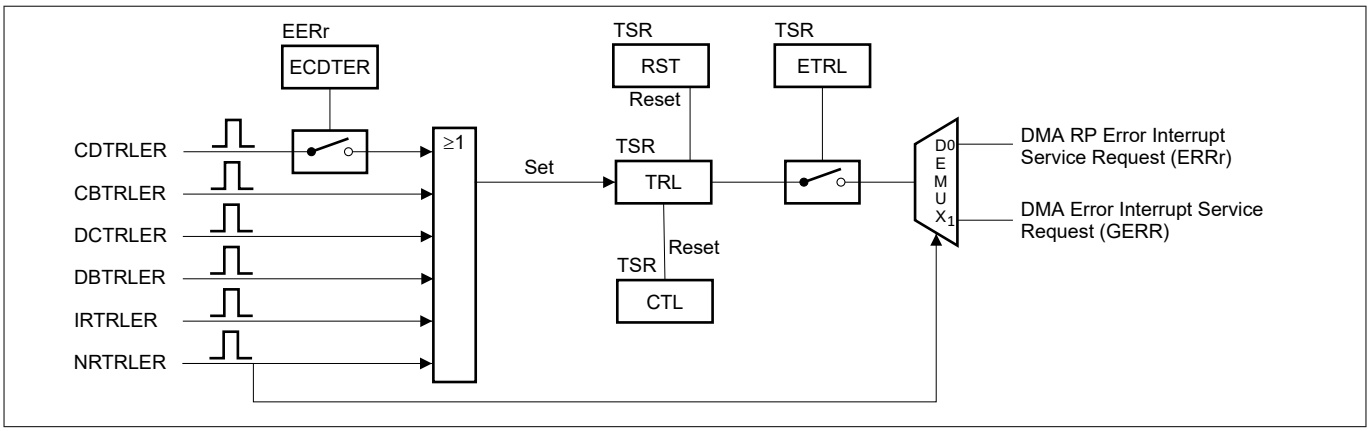


Figure 121 DMA channel transaction request lost interrupt service request

For error conditions CBTRLER, DCTRLER, DBTRLER and IRTRLER, the DMA sets the DMA channel TRL bit (TSRc.TRL = 1_B) and if the DMA channel enable TRL bit is set (TSRc.ETRL = 1_B), the DMA triggers a DMA RP error interrupt service request.

Error condition CDTRLER is enabled or disabled by the RP CDTRLER enable bit (EERr.ECDTER). If CDTRLER is enabled and the error condition CDTRLER occurs, the DMA sets the DMA channel TRL bit (TSRc.TRL = 1_B) and if the DMA channel enable TRL bit (TSRc.ETRL) is set, the DMA triggers a DMA RP error interrupt service request.

An error handler shall interrogate the DMA channels to identify the source of the error. Software may clear TSRc.TRL by setting the DMA channel TSRc.CTL or TSRc.RST.

Related information

DMA hardware request on page 1257
DMA daisy chain request on page 1260
Software switch on page 1236
Automatic hardware switch on page 1236
DMA auto start request on page 1261
DMA RP error interrupt service request on page 1275
No resource partition TRL error interrupt service request on page 1282

9.3.4.4.2 DMARAM integrity error interrupt service request

When a DMA channel wins arbitration, a ME accesses the DMARAM to read the TCS. If an ECC error is detected, the RP sets the error flag ERRSRr.RAMER. The DMA transaction does not take place.

- A RAMER is indicated by the RP error status flag ERRSRr.RAMER
- The DMA triggers a DMA RP error interrupt service request
- If software sets CLREr.CRAMER, the DMA clears error status flag ERRSRr.RAMER

Related information

DMA RP error interrupt service request on page 1275
TC4Dx DMARAM integrity error interrupt service request on page 1336

9.3.4.4.3 Source and destination error interrupt service request

SER and DER include access protection errors and unsupported types of bus transaction.

The RP detects the following error conditions:

- An SER indicates a bus error occurred during a DMA read move from a source address
  - An SER is indicated by the RP error status flag ERRSRr.SER
  - If EERr.ESER is set, the DMA triggers a DMA RP error interrupt service request
  - If software sets CLREr.CSER, the DMA clears error status flag ERRSRr.SER
- A DER indicates a bus error that occurred during a DMA write move to a destination address
  - A DER is indicated by the RP error status flag ERRSRr.DER
  - If EERr.EDER is set, the DMA triggers a DMA RP error interrupt service request
  - If software sets CLREr.CDER, the DMA clears error status flag ERRSRr.DER

If a SER or DER is reported then the ME completes the DMA transaction. If a SER is reported during a DMA read move then the DMA write move is not executed, but the destination address is updated.

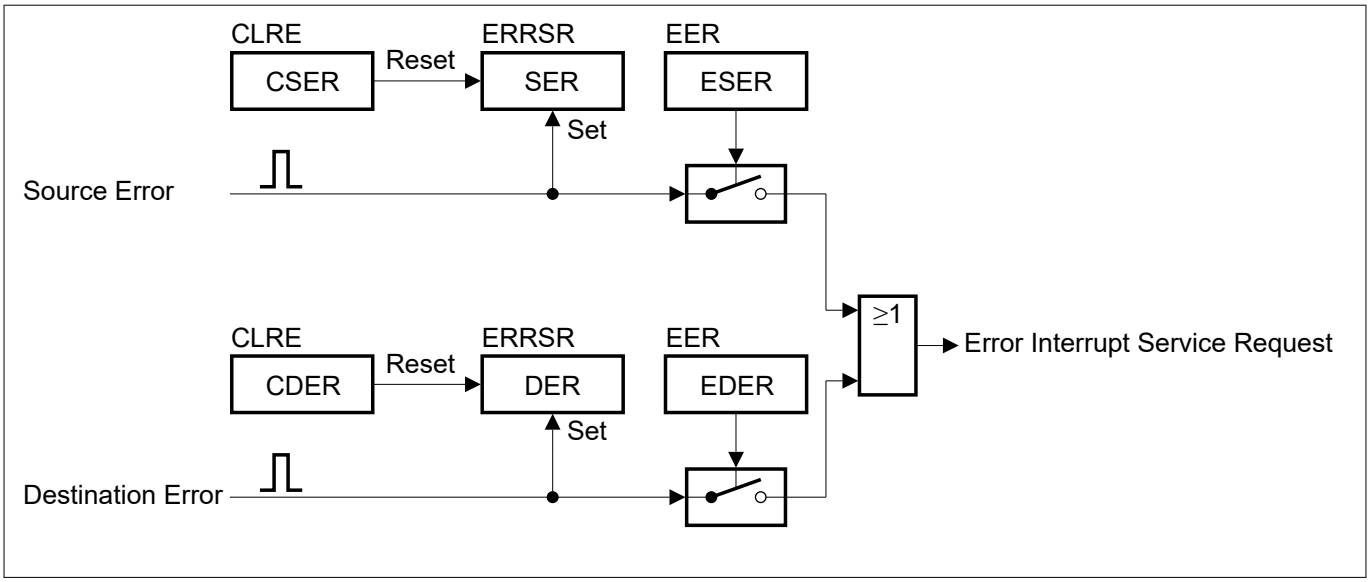


Figure 122 Source and destination error interrupt service request

Related information

DMA RP error interrupt service request on page 1275

9.3.4.4.4 Linked list operation TCS error interrupt service request

During DMALL, ACCL, CHKLL and CONLL operations if an error is reported during the loading of the next TCS from the system interconnect, the RP sets the error flag ERRSRr.DLLER. The linked list operation is aborted and the DMA channel transaction request state bit TSRc.CH bit cleared.

- A DLLER is indicated by the RP error status flag ERRSRr.DLLER
- If EERr.ELER is set, the DMA triggers a DMA RP error interrupt service request
- If software sets CLREr.CDLLER, the DMA clears error status flag ERRSRr.DLLER

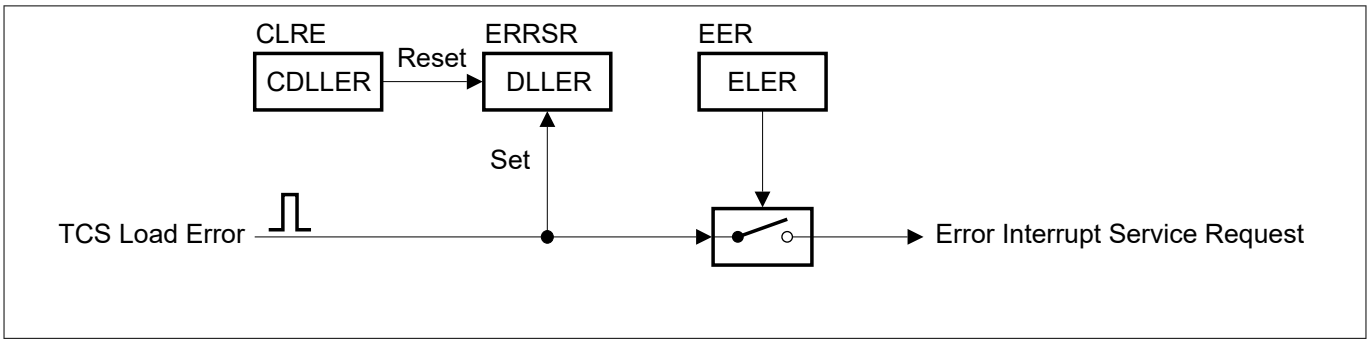


Figure 123 Linked list operation TCS error interrupt service request

Related information

DMA RP error interrupt service request on page 1275

9.3.4.4.5 CHKLL DMA address checksum error interrupt service request

The ME compares the DMA address checksum calculated by the current DMA transaction against the expected DMA address checksum stored in the next TCS. If the checksums do not match then the RP sets the error flag ERRSRr.CLLER.

- A CLLER is indicated by the RP error status flag ERRSRr.CLLER

- The DMA triggers a DMA RP error interrupt service request
- If software sets CLREr.CCLLER, the DMA clears error status flag ERRSRr.CLLER

Related information

DMA address checksum on page 1222
DMA RP error interrupt service request on page 1275

9.3.4.4.6 TCS configuration error interrupt service request

During DMA channel TCS configuration, some configuration errors are detected:

- If software writes to the shadow address register when shadow operation read only mode is configured, the DMA stores a RP TCS configuration error

On detection of such an error, the RP sets the error flag ERRSRr.TCSCER. The write is aborted.

- An TCSCER is indicated by the RP error status flag ERRSRr.TCSCER
- The DMA triggers a DMA RP error interrupt service request
- If software sets CLREr.CTCSCER, the DMA clears error status flag ERRSRr.TCSCER

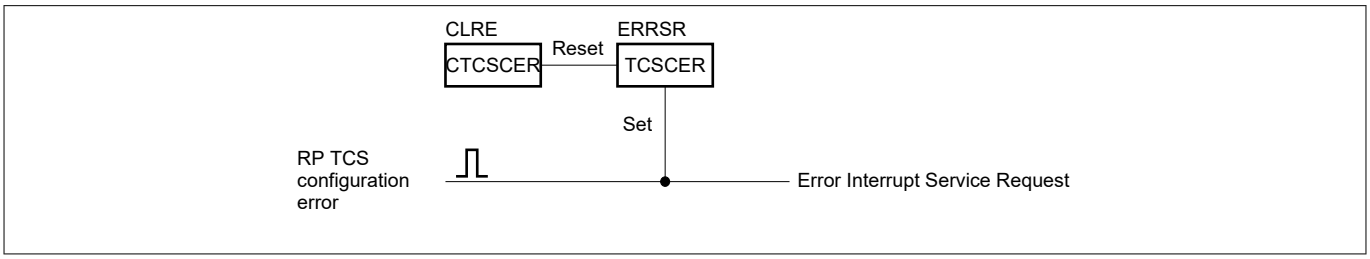


Figure 124 TCS configuration error interrupt service request

Related information

DMA RP error interrupt service request on page 1275

9.3.5 DMA move engine

The DMA move engine services the move operations as defined by the TCS of the active channel.

Any ME may service a DMA request from any DMA channel. As soon as a DMA channel wins arbitration, the DMA selects one of the available MEs to execute the pending DMA request. When the selected ME is idle, the TCS is copied from the DMARAM into the channel registers of the ME and the DMA request is serviced. The ME requests the required system interconnect master interfaces and reads or writes data according to the TCS parameters of the active DMA channel. The ME is able to wait if the source or destination target is not available.

The processing of a DMA transfer (composed of several DMA moves) by the ME cannot be interrupted and is always completed, except for the case where pattern detection is enabled and a pattern match is detected. A DMA channel interrupt, reset, halt request or debug suspend only becomes active when the current DMA transfer is complete and the TCS has written back to the DMARAM. Error conditions are reported. On completion of a DMA transaction or when a DMA channel loses channel arbitration the ME writes the TCS back to the DMARAM.

Related information

DMA request arbitration on page 1261

9.3.5.1 ME read buffer

The ME stores the DMA read move data in eight 32-bit read buffer locations. If a DMA channel is configured for 8-bit, 16-bit, 32-bit, 64-bit or 128-bit channel data width and a DMA read move is to a cached address[13], the ME translates the read access on the system interconnect into an SRI BTR4 access to a 32-byte aligned address. The ME uses the DMA read move data in the ME read buffer together with the 32-byte aligned address as follows:

- Hit, if the 32-byte aligned address of the next DMA read move matches the 32-byte aligned address stored in the ME read buffer then the DMA read move data is read from the ME read buffer. No read access to the system interconnect occurs
- Miss, if the 32-byte aligned address of the next DMA read move does not match the 32-byte aligned address stored in the ME read buffer then the ME invalidates the contents of the ME buffer. The ME executes the DMA read move by performing a read access to the system interconnect

The ME invalidates the ME read buffer at the start of a DMA transfer.

9.3.5.1.1 DMA address checksum

If a DMA read move is to a cached address, then the ME calculates the DMA address checksum using the 32-byte aligned address source address.

Related information

DMA address checksum on page 1222

9.3.6 DMA internal switch

The DMA internal switch arbitrates between each system interconnect request from the move engines. Address routing across the whole memory map directs a request to a system interconnect master interface.

The DMA internal switch also implements buffering capabilities for move actions on the system interconnect (at least one DMA move per master interface is buffered).

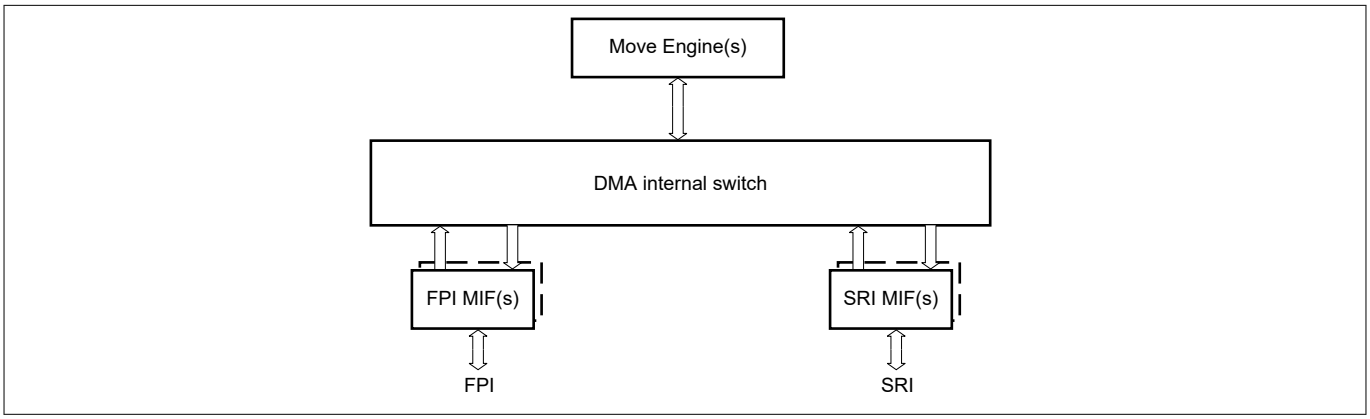


Figure 125 DMA internal switch

9.3.6.1 DMA master agents

The DMA instances master agents to the SRI and FPI system interconnect instances.

The DMA FPI master agents support

- Single data read and write transactions (8-bit, 16-bit, 32-bit)
- Block transfer read and write transactions (64-bit, 128-bit, 256-bit)

13 Segments 8 and 9 (see system address map for details).

- Accesses in supervisor mode according to the setting in the MODEr.MODE bit-field
- Generation of interleaved FPI transactions from different access requesters (MEs)
- De-assertion of request after retry in order to prevent bus blocking
- Out of order transactions from different sources in order to avoid side effects (blocking) between the different access requesters (MEs)

The DMA SRI master agents support

- Single data read and write transactions (8-bit, 16-bit, 32-bit, 64-bit)
- Block transfer read and write transactions (128-bit, 256-bit, 512-bit)[14)]
- Accesses in supervisor mode according to the setting in the MODEr.MODE bit-field
- Generation of interleaved SRI transactions from different access requesters (MEs)
- Data pacing to transmit data on the interface only on cycles when the slave indicates that it is ready to accept the data

Read modify write (RMW) Support

The DMA does not support RMW accesses to the system interconnect.

9.3.6.2 DMA internal switch arbitration

If requests from both move engines conflict, the DMA internal switch priorities listed in the table below determine arbitration.

Table 642 DMA internal switch priorities

Priority	Agent requests	Comment
Highest	ME write	For concurrent write requests, the highest DMA channel wins.
Lowest	ME read	For concurrent read requests, the highest DMA channel wins.

The DMA internal switch supports pipelining. High and low priority ME0 and ME1 requests may be pipelined.

9.3.7 DMA error interrupt service request

The DMA generates one error interrupt service request (GERR) to cover all error events that cannot be attributed to a single RP.

- DMA channel TRL error interrupt service request that cannot be attributed to a single RP (NRTRLER)

If a DMA error interrupt service request is triggered then the software application error handler shall read the contents of the error status registers to identify the origin of the error.

For all such error conditions, the DMA performs the following actions:

- The DMA sets the corresponding error status bit in the ERRSR register
- The DMA records the number of the DMA channel in the DMA last error channel bit-field ERRSR.LEC
- If the DMA channel is enabled for DMA channel hardware request then DMA channel TSRc.HTRE is cleared
- If software sets a bit in the CLRE register, the corresponding status bit in the ERRSR register is cleared

If a previous DMA error status flag is still set when a new error occurs that requires an update of the DMA error status bits, the DMA error status overflow bit ERRSR.OVFLER is set. Software may clear the DMA error status overflow bit by writing a 1_B to the CLRE.COVFLER bit-field.

14 512-bit block transfer DMA move supported for DMA read move from SRI source to DMA write move to SRI destination.

Related information

No resource partition TRL error interrupt service request on page 1282

9.3.7.1 No resource partition TRL error interrupt service request

A DMA channel No resource partition TRL event occurs for the following conditions.

- No Resource partition TRL Error (NRTRLER): If a DMA hardware request or a DMA auto-start request is detected and the DMA channel has not been assigned to a resource partition (HRRc.HRPV is set to 0_B)

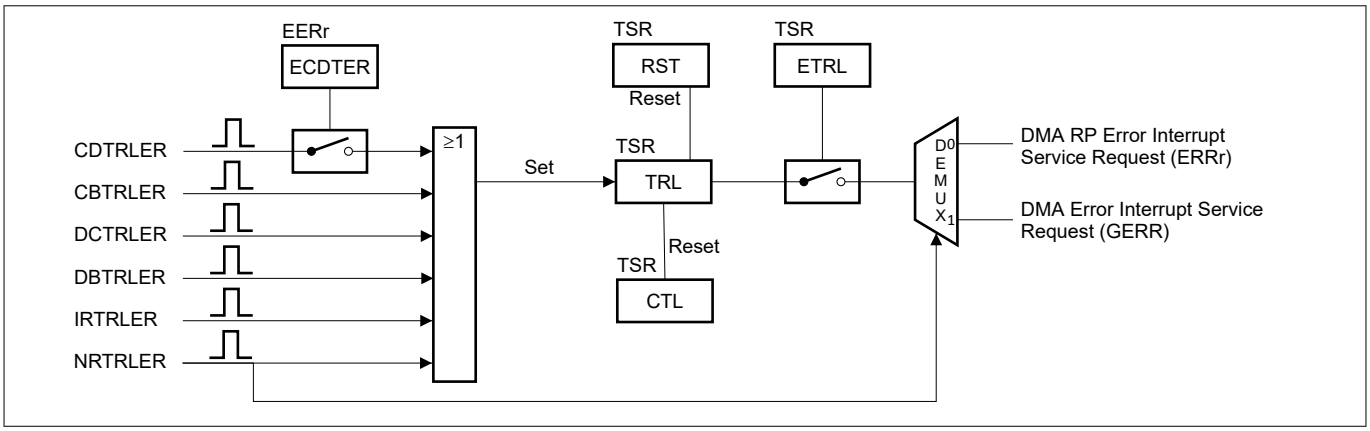


Figure 126 DMA channel transaction request lost interrupt service request

For error condition NRTRLER, the DMA sets the DMA channel TRL bit (TSRc.TRL = 1_B) and if the DMA channel enable TRL bit is set (TSRc.ETRL = 1_B), the DMA triggers a DMA error interrupt service request.

An error handler shall interrogate the DMA channels to identify the source of the error. Software may clear TSRc.TRL by setting the DMA channel TSRc.CTL or TSRc.RST.

Related information

DMA channel TRL error interrupt service request on page 1276

9.3.8 Power modes

The DMA disable status bit (CLC.DISS) reports the state of the DMA module clock.

After the assertion of an application reset, the DMA module clock is disabled (CLC.DISS = 1_B).

9.3.8.1 Sleep mode

The DMA enters sleep mode as follows:

- If the sleep mode enable control bit is enabled (CLC.EDIS = 0_B) and the system sleep request control (REQSLP_SLEEP) is activated, the DMA enters sleep mode when the ME(s) have completed any ongoing DMA transfers and any DMA channel interrupts, DMA RP error interrupts or DMA error interrupts have been sent to the Interrupt Router
- If the sleep mode enable control bit is enabled (CLC.EDIS = 0_B) and the system shutdown request control (SHSEQREQx) is activated, the DMA enters sleep mode when the ME(s) have completed any pending DMA transfers and any DMA channel, DMA RP error or DMA error interrupts have been sent to the Interrupt Router
- If software writes a DMA disable request (CLC.DISR = 1_B), the DMA enters sleep mode when the ME(s) have completed any pending DMA transfers and any DMA channel, DMA RP error or DMA error interrupts have been sent to the Interrupt Router

***Note**: Software should ensure that CLC.EDIS is set to 0_B as described above if the DMA should react to the system shutdown request.*

If DMA sleep mode is requested as in any of the conditions above:

- DMA SFRs
  - Software accesses to the CLC, OTSS, ID, TIMEOCS, PROTSE, PROTCSE, PROTRPr and ACCENRPr SFRs are executed
  - Software accesses to the other DMA registers result in a bus error
- DMARAM
  - Software accesses the to the DMARAM results in a bus error

If the DMA is in sleep mode:

- The DMA disables the DMA module clock to minimize DMA power consumption
- DMA SFRs
  - Software accesses to the CLC, OTSS, ID, TIMEOCS, PROTSE, PROTCSE, PROTRPr and ACCENRPr SFRs are executed
  - Software accesses to the other DMA registers result in a bus error
- DMARAM
  - Software accesses the to the DMARAM results in a bus error

If the system sleep request control and the system shutdown request control are deactivated and software clears the DMA disable request (CLC.DISR = 1_B):

- The DMA enables the DMA module clock
- The DMA resumes the servicing of DMA requests

9.4 Registers

9.4.1 Register overview - access mode glossary

Table 643 Register overview - access mode glossary

Keyword	Description
CSE	Access protection using PROT register PROTCSE.
RPr (r=0-15)	Access protection using PROT register PROTRPr.
SE	Access protection using PROT register PROTSE.
APU-Pr (r=0-15)	Protection group consisting of registers ACCENRPrWRA, ACCENRPrWRB, ACCENRPrRDA, ACCENRPrRDB, ACCENRPrVM, ACCENRPrPRS.
Pr	Access protection using *APU-Pr* registers.
U	No access restrictions.
BE	Always returns a Bus Error.
PROT	Access restrictions as defined in the PROT register access rules.
32	Access only when using 32-bit width.
OEN	Access only when OCDS is enabled.
M	Indicates a module specific access condition. Refer to the register description for details of the specific access condition.

9.4.2 Registers overview - DMA (ascending offset address)

Table 644 Registers overview - DMA (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CLC	Clock control register	0000_H	U	SV, SE	Application Reset	1286
OTSS	OCDS trigger set select	0004_H	U	SV, SE	Debug Reset	1287
ID	Module identification register	0008_H	U	BE	PowerOn Reset	1288
PROTSE	PROT register safe endinit	0024_H	U	SV, PROT	Application Reset	1289
PROTCSE	PROT register cyber-security endinit	0028_H	U	SV, PROT	Application Reset	1291
TIME	DMA time register	0080_H	U	SV, SE, 32	System Reset	1293
ERRSR	DMA error status register	0084_H	U	BE	Application Reset	1293
CLRE	DMA clear error register	0088_H	U	SV, SE	Application Reset	1294
TIMEOCS	DMA time OCDS control and status register	$008C_H$	U	SV, SE	Debug Reset	1295

(table continues...)

Table 644 (continued) Registers overview - DMA (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
LCLTEST	DMA LCL test register	0090_H	U	SV, SE	Application Reset	1295
MEmSR	ME m status register	0410_H+ m*200_H	OEN	BE	Application Reset	1296
MEmRDCRC	ME m channel read data CRC register	0460_H+ m*200_H	OEN	BE	Application Reset	1297
MEmSDCRC	ME m channel source and destination address CRC register	0464_H+ m*200_H	OEN	BE	Application Reset	1297
MEmSADR	ME m channel source address register	0468_H+ m*200_H	OEN	BE	Application Reset	1297
MEmDADR	ME m channel destination address register	$046C_H$+ m*200_H	OEN	BE	Application Reset	1298
MEmADICR	ME m channel address and interrupt control register	0470_H+ m*200_H	OEN	BE	Application Reset	1298
MEmCHCR	ME m channel control register	0474_H+ m*200_H	OEN	BE	Application Reset	1300
MEmSHADR	ME m channel shadow address register	0478_H+ m*200_H	OEN	BE	Application Reset	1300
MEmCHSR	ME m channel status register	$047C_H$+ m*200_H	OEN	BE	Application Reset	1301
ACCENRPrWRA	RP r write access enable register A	1000_H+r* 80_H	U	SV, RPr	Application Reset	1302
ACCENRPrWRB	RP r write access enable register B	1004_H+r* 80_H	U	SV, RPr	Application Reset	1302
ACCENRPrRDA	RP r read access enable register A	1008_H+r* 80_H	U	SV, RPr	Application Reset	1303
ACCENRPrRDB	RP r read access enable register B	$100C_H$+r *80_H	U	SV, RPr	Application Reset	1303
ACCENRPrVM	RP r VM access enable register	1010_H+r* 80_H	U	SV, RPr	Application Reset	1304
ACCENRPrPRS	RP r PRS access enable register	1014_H+r* 80_H	U	SV, RPr	Application Reset	1305
PROTRPr	PROT register RP r	1020_H+r* 80_H	U	SV, PROT	Application Reset	1305
MODEr	RP r mode register	1030_H+r* 80_H	U	SV, M, 32	Application Reset	1307
PRRAr	RP r pattern read register 0	1038_H+r* 80_H	Pr	SV, Pr	Application Reset	1308
PRRBr	RP r pattern read register 1	$103C_H$+r *80_H	Pr	SV, Pr	Application Reset	1309

(table continues...)

Table 644 (continued) Registers overview - DMA (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ERRINTRr	RP r error interrupt set register	1040_H+r* 80_H	Pr	SV, Pr	Application Reset	1309
EERr	RP r enable error register	1044_H+r* 80_H	Pr	SV, Pr	Application Reset	1310
ERRSRr	RP r error status register	1048_H+r* 80_H	Pr	BE	Application Reset	1310
CLREr	RP r clear error register	$104C_H$+r *80_H	Pr	SV, Pr	Application Reset	1313
HRRc	DMA channel c resource partition assignment register	1800_H+c *4	U	SV, M, 32	Application Reset	1314
SUSENRc	DMA channel c suspend enable register	$1A00_H$+c *4	U	SV, M	Debug Reset	1315
SUSACRc	DMA channel c suspend acknowledge register	$1C00_H$+c *4	U	BE	Debug Reset	1316
TSRc	DMA channel c transaction state register	$1E00_H$+c *4	M	SV, M	Application Reset	1316
RDCRCRc	DMA channel c read data CRC register	2000_H+c *20_H	M	SV, M	Application Reset	1318
SDCRCRc	DMA channel c source and destination address CRC register	2004_H+c *20_H	M	SV, M	Application Reset	1319
SADRc	DMA channel c source address register	2008_H+c *20_H	M	SV, M	Application Reset	1319
DADRc	DMA channel c destination address register	$200C_H$+c *20_H	M	SV, M	Application Reset	1320
ADICRc	DMA channel c address and interrupt control register	2010_H+c *20_H	M	SV, M	Application Reset	1320
CHCFGRc	DMA channel c configuration register	2014_H+c *20_H	M	SV, M	Application Reset	1324
SHADRc	DMA channel c shadow address register	2018_H+c *20_H	M	SV, M	Application Reset	1326
CHCSRc	DMA channel c control and status register	$201C_H$+c *20_H	M	SV, M	Application Reset	1327

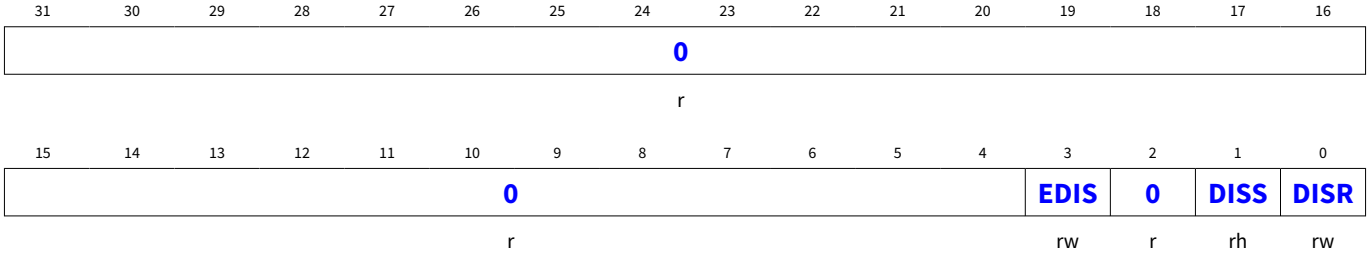
9.4.3 Clock control register

The clock control register CLC allows the programmer to adapt the functionality and power consumption of the module to the requirements of the application.

Register CLC controls the module clock signal and the reactivity to the sleep request control signal (REQSLP_SLEEP).

CLC Offset address: 0000_H

Clock control register Application Reset value: $0000\ 0001_H$



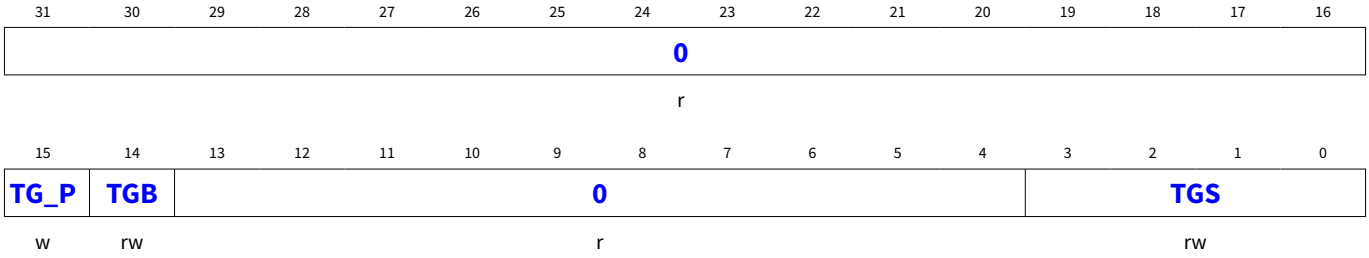
Field	Bits	Type	Description
DISR	0	rw	**Module disable request bit** Used for enable and disable control of the module. 0_B On request: enable the module clock 1_B Off request: stop the module clock
DISS	1	rh	**Module disable status bit** When CLC.DISS=1_B, registers CLC, OTSS, ID, TIMEOCS, PROTSE, PROTCSE, PROTRPr and ACCENRPr are still accessible. Accesses to other DMA registers are not executed and will result in a bus error. 0_B Module clock is enabled 1_B Module is not clocked
EDIS	3	rw	**Sleep mode enable control** Used to control the module's reaction to sleep mode. 0_B Sleep mode request is enabled and functional 1_B Module disregards the sleep request control signal (REQSLP_SLEEP)
0	2, 31:4	r	**Reserved** Read as 0; should be written with 0.

9.4.4 OCDS trigger set select

The OTSS register controls the selection of the OCDS Trigger Bus (OTGB).

OTSS Offset address: 0004_H

OCDS trigger set select Debug Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
TGS	3:0	rw	**Trigger set for OTGB0 or OTGB1** 0_H No trigger set output 1_H Trigger set 1 2_H Trigger set 2 3_H Reserved … 7_H Reserved 8_H Trigger set 8 9_H Trigger set 9 A_H Trigger set 10 B_H Trigger set 11 C_H Trigger set 12 D_H Trigger set 13 E_H Trigger set 14 F_H Trigger set 15
TGB	14	rw	**OTGB0 or OTGB1 bus select** 0_B Trigger set is output on OTGB0 1_B Trigger set is output on OTGB1
TG_P	15	w	**TGS, TGB write protection** TGS and TGB are only written when TG_P is 1, otherwise unchanged. Read as 0.
0	13:4, 31:16	r	**Reserved** Read as 0; should be written with 0.

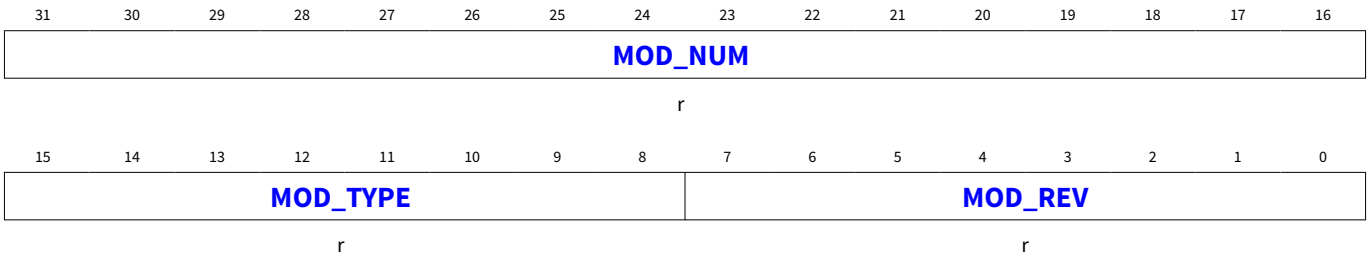
Table 645 Access mode restrictions of OTSS sorted by descending priority

Mode name	Access mode		Description
write 1 to .**TG_P**	rw	TGB, TGS	Set TG_P during write access
(default)	r	TGB, TGS	

9.4.5 Module identification register

ID Offset address: 0008_H

Module identification register PowerOn Reset value: $0087\ C005_H$



Field	Bits	Type	Description
MOD_REV	7:0	r	**Module revision** Indicates the revision number of the implementation.
MOD_TYPE	15:8	r	**Module type** This internal marker is fixed to $C0_H$.
MOD_NUM	31:16	r	**Module number** Indicates the module identification number (0087_H = DMA).

9.4.6 PROT register safe endinit

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner and in some PROT states, the secure master to update the PROT state as described in the PROT mechanism.

The PROTSE register controls lock and unlock of the local Safe Endinit (SE) protected control registers.

PROTSE Offset address: 0024_H

PROT register safe endinit Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OWEN	ODEF	TAGID						PRSEN	PRS			VMEN	VM		
w	rw	rw						rw	rw			rw	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												SWEN	STATE		
r												w	rwh		

Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write

(table continues...)

(continued)

Field	Bits	Type	Description
VM	18:16	rw	**Virtual machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual machine definition enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.
PRSEN	23	rw	**Protection set definition enable for PROT owner** 0_B **Disable**: PRS is not part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG-ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; should be written with 0.

Table 646 Access mode restrictions of PROTSE sorted by descending priority

Mode name	Access mode		Description
write 1 to .**SWEN**	rwh	STATE	
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

9.4.7 PROT register cyber-security endinit

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner and in some PROT states, the secure master to update the PROT state as described in the PROT mechanism.

This register defines the cyber-security master that coordinates with the PROTSE owner to configure the DMA resource partitions.

PROTCSE Offset address: 0028_H

PROT register cyber-security endinit Application Reset value: $4C00\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OWEN	ODEF	TAGID						PRSEN	PRS			VMEN	VM		
w	rw	rw						rw	rw			rw	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				CSEN	CSTAGOFF			0				SWEN	STATE		
r				rw	rw			r				w	rwh		

Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
CSTAGOFF	10:8	rw	**CS asset tag offset** When CSEN is set, defines the tag offset that is reserved for the CyberSecurity master. As long as CSEN remains set, the hardware ensures that resource partitions configured as a non-CS asset cannot re-use the CS asset tag offset.
CSEN	11	rw	**Enable for CSTAGOFF reservation** 0_B CSTAGOFF is ignored. All tag offsets may be used in any resource partition. 1_B Tag offset in CSTAGOFF is reserved for use in RPs which are CS assets.

(table continues...)

(continued)

Field	Bits	Type	Description
VM	18:16	rw	**Virtual machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual machine definition enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.
PRSEN	23	rw	**Protection set definition enable for PROT owner** 0_B **Disable**: PRS is not part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	7:4, 15:12	r	**Reserved** Read as 0; should be written with 0.

Table 647 Access mode restrictions of PROTCSE sorted by descending priority

Mode name	Access mode		Description
write 1 to .**SWEN**	rwh	STATE	
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
PROT_OWNER and .**STATE** == 0x0	rw	CSEN, CSTAGOFF	PROT owner can modify the CS tag offset in state Init
PROT_OWNER and .**STATE** == 0x1	rw	CSEN, CSTAGOFF	PROT owner can modify the CS tag offset in state Config
PROT_OWNER and write 1 to .**SWEN** and value written to .**STATE** == 0x1	rw	CSEN, CSTAGOFF	PROT owner can modify the CS tag offset while moving to state Config
PROT_OWNER and .**STATE** == 0x2	rw	CSEN, CSTAGOFF	PROT owner can modify the CS tag offset in state ConfigSec

(table continues...)

Table 647 (continued) Access mode restrictions of PROTCSE sorted by descending priority

Mode name	Access mode		Description
PROT_OWNER and write 1 to .**SWEN** and value written to .**STATE** == 0x2	rw	CSEN, CSTAGOFF	PROT owner can modify the CS tag offset while moving to state ConfigSec
(default)	r	CSEN, CSTAGOFF, ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

9.4.8 DMA time register

The time register stores the 32-bit count value used for the appendage of DMA timestamps.

TIME Offset address: 0080_H

DMA time register System Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
COUNT															
rwh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
COUNT															
rwh															

Field	Bits	Type	Description
COUNT	31:0	rwh	**Timestamp count** The count value used during the appendage of DMA timestamps.

9.4.9 DMA error status register

The error status register indicates if the DMA controller could not service a DMA request that cannot be attributed to a single RP. The channel number of the channel that last triggered an error is recorded in ERRSR.LEC.

ERRSR Offset address: 0084_H

DMA error status register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0													**OVFL ER**	**0**	**NRT RLER**
r													rh	r	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0									**LEC**						
r									rh						

Field	Bits	Type	Description
LEC	6:0	rh	**DMA last error channel** This bit field indicates the DMA channel number of the last DMA channel generating an error reported in the ERRSR register.
NRTRLER	16	rh	**No resource partition TRL error** This bit is set when a DMA request is detected and the DMA channel has not been assigned to a resource partition (HRRc.HRPV is set to 0_B). 0_B No error occurred. 1_B A TRL error has occurred.
OVFLER	18	rh	**DMA error status overflow** This bit is set whenever the DMA error status overflows - that is, when any previously reported DMA error status remains set in the ERRSR register and a new DMA error is reported. 0_B No error overflow occurred. 1_B An error overflow occurred.
0	15:7, 17, 31:19	r	**Reserved** Read as 0; should be written with 0.

9.4.10 DMA clear error register

The clear error register contains bits to clear the corresponding DMA error flags.

CLRE Offset address: 0088_H

DMA clear error register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0													COVFLER	0	CNRTRLER
r													w	r	w

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															
r															

Field	Bits	Type	Description
CNRTRLER	16	w	**Clear no resource partition TRL error** 0_B No action 1_B Clear DMA no resource partition TRL error flag ERRSR.NRTRLER.
COVFLER	18	w	**Clear DMA error status overflow** 0_B No action 1_B Clear DMA error status overflow flag ERRSR.OVFLER.
0	15:0, 17, 31:19	r	**Reserved** Read as 0; should be written with 0.

9.4.11 DMA time OCDS control and status register

The DMA time OCDS control and status register (TIMEOCS) controls the DMA timestamp behavior in suspend mode. When OCDS is disabled the TIMEOCS suspend control is ineffective.

TIMEOCS Offset address: $008C_H$

DMA time OCDS control and status register Debug Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		SUSSTA	SUS_P	SUS				0							
r		rh	w	rw				r							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															
r															

Field	Bits	Type	Description
SUS	27:24	rw	**OCDS suspend control** Controls the sensitivity of the DMA timestamp counter to the OCDS suspend signal 0_H Will not suspend 1_H Reserved, do not use this combination 2_H 32-bit DMA timestamp counter will be stopped **others,** Reserved, do not use this combination
SUS_P	28	w	**SUS write protection** SUS is only written when SUS_P is 1, otherwise unchanged. Read as 0.
SUSSTA	29	rh	**Suspend state** 0_B DMA timestamp counter is not (yet) suspended 1_B DMA timestamp counter is suspended
0	23:0, 31:30	r	**Reserved** Read as 0; should be written with 0.

Table 648 Access mode restrictions of TIMEOCS sorted by descending priority

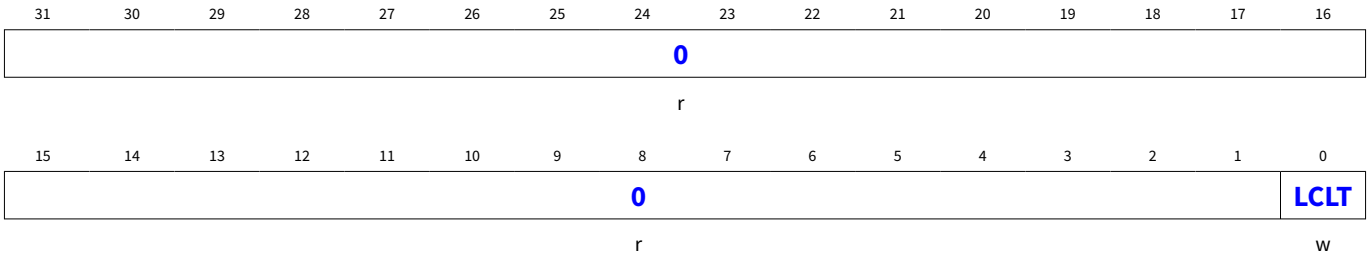
Mode name	Access mode		Description
write 1 to .**SUS_P**	rw	SUS	
(default)	r	SUS	

9.4.12 DMA LCL test register

Provides the capability for software to inject a fault condition into the comparators of each lockstep comparator logic block. The implementation should generate a single cycle fault each time the bit is written with '1'.

LCLTEST Offset address: 0090_H

DMA LCL test register Application Reset value: $0000\ 0000_H$



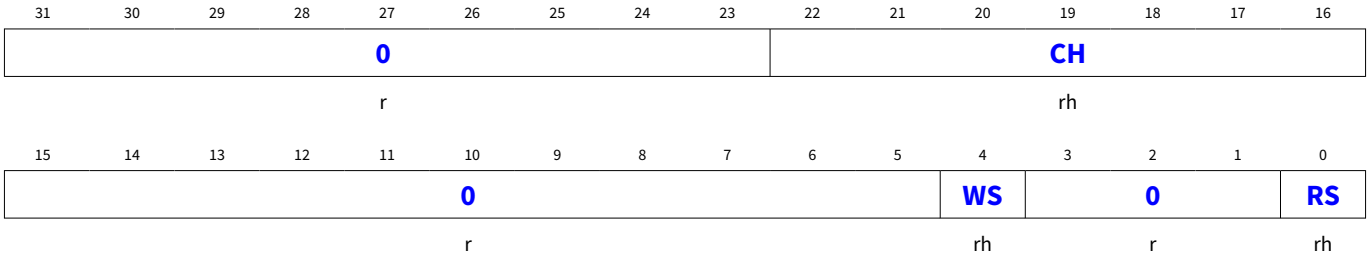
Field	Bits	Type	Description
LCLT	0	w	**LCL lockstep test** Fault injection for LCL. Reads as zero. 0_B No action 1_B Inject single fault in the LCL
0	31:1	r	**Reserved** Read as 0; should be written with 0.

9.4.13 ME m status register

The ME status register holds status information about the DMA transaction handled by the ME.

MEmSR (m=0-1) Offset address: $0410_H+m*200_H$

ME m status register Application Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
RS	0	rh	**ME read status** 0_B ME is not performing a DMA read move. 1_B ME is performing a DMA read move.
WS	4	rh	**ME write status** 0_B ME is not performing a DMA write move. 1_B ME is performing a DMA write move.
CH	22:16	rh	**ME active channel** Indicates the number of the DMA channel currently processed by the ME.
0	3:1, 15:5, 31:23	r	**Reserved** Read as 0; should be written with 0.

9.4.14 ME m channel read data CRC register

The read only DMA active channel read data CRC register will store one polynomial calculation during each DMA read move provided that there is no retry or error condition flagged.

MEmRDCRC (m=0-1) Offset address: $0460_H+m*200_H$

ME m channel read data CRC register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RDCRC															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RDCRC															
rh															

Field	Bits	Type	Description
RDCRC	31:0	rh	**Read data CRC** Active DMA channel read data CRC32 ethernet polynomial checksum.

9.4.15 ME m channel source and destination address CRC register

The read only DMA active channel source and destination address CRC Register will store one polynomial calculation during each DMA read and each DMA write move provided there is no retry or error condition flagged.

MEmSDCRC (m=0-1) Offset address: $0464_H+m*200_H$

ME m channel source and destination address CRC register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SDCRC															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SDCRC															
rh															

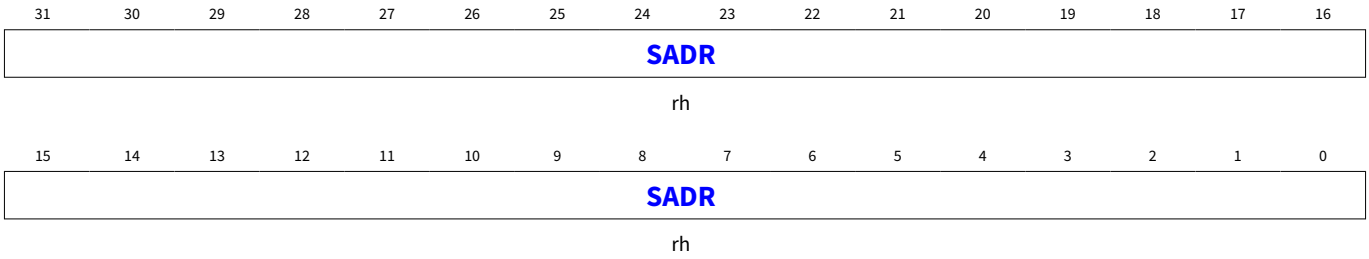
Field	Bits	Type	Description
SDCRC	31:0	rh	**Source and destination address CRC** Active DMA channel source and destination address CRC32 ethernet polynomial checksum.

9.4.16 ME m channel source address register

The read only DMA active channel source address register holds the 32-bit source address. If a DMA channel is active, MEmSADR is updated continuously and shows the actual source address used for DMA read moves.

MEmSADR (m=0-1) Offset address: $0468_H+m*200_H$

ME m channel source address register Application Reset value: $0000\ 0000_H$



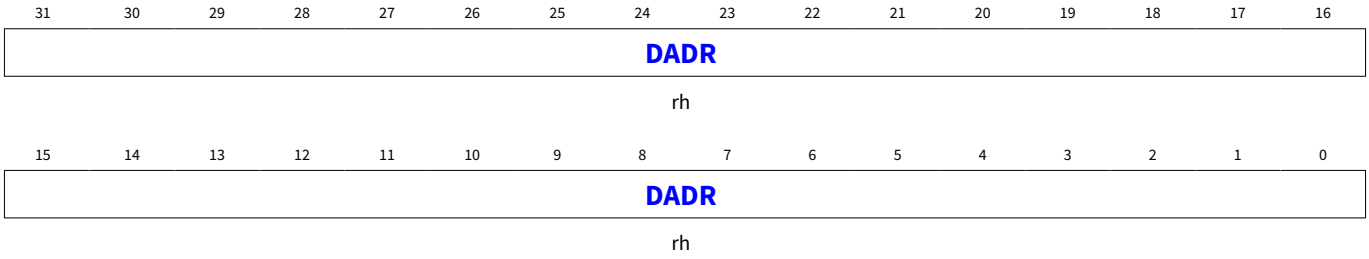
Field	Bits	Type	Description
SADR	31:0	rh	**Source address** Active DMA channel 32-bit source address used for DMA read moves.

9.4.17 ME m channel destination address register

The read only DMA active channel destination address register holds the 32-bit destination address. If a DMA channel is active, MEmDADR is updated continuously and shows the actual destination address used for DMA write moves.

MEmDADR (m=0-1) Offset address: $046C_H+m*200_H$

ME m channel destination address register Application Reset value: $0000\ 0000_H$



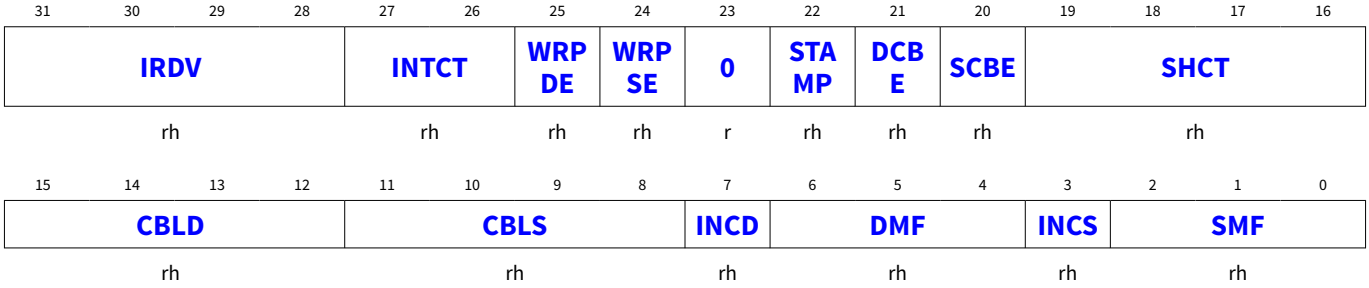
Field	Bits	Type	Description
DADR	31:0	rh	**Destination address** Active DMA channel 32-bit destination address used for DMA write moves.

9.4.18 ME m channel address and interrupt control register

The read only DMA active channel address and interrupt control register shows the address update and interrupt configuration of the active DMA channel.

MEmADICR (m=0-1) Offset address: $0470_H+m*200_H$

ME m channel address and interrupt control register Application Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
SMF	2:0	rh	**Source address modification factor** Active DMA channel source address modification factor.
INCS	3	rh	**Increment of source address** Active DMA channel increment of source address control.
DMF	6:4	rh	**Destination address modification factor** Active DMA channel destination address modification factor.
INCD	7	rh	**Increment of destination address** Active DMA channel increment of destination address control.
CBLS	11:8	rh	**Source circular buffer length** Active DMA channel circular source buffer control.
CBLD	15:12	rh	**Destination circular buffer length** Active DMA channel circular destination buffer control.
SHCT	19:16	rh	**Shadow control** Active DMA channel control of shadow address register function.
SCBE	20	rh	**Source circular buffer enable** Active DMA channel circular source buffer enable or disable.
DCBE	21	rh	**Destination circular buffer enable** Active DMA channel circular destination buffer enable or disable.
STAMP	22	rh	**Timestamp** Active DMA channel control to enable the appendage of a timestamp after the end of the last DMA move during a DMA transaction.
WRPSE	24	rh	**Wrap source enable** Active DMA channel source buffer interrupt trigger enable or disable.
WRPDE	25	rh	**Wrap destination enable** Active DMA channel destination buffer interrupt trigger enable or disable.
INTCT	27:26	rh	**Interrupt control** Active DMA channel interrupt service request control.
IRDV	31:28	rh	**Interrupt raise detect value** Active DMA channel control of the threshold limit of of DMA channel CHSR.TCOUNT for triggering a channel interrupt service request.
0	23	r	**Reserved** Read as 0; should be written with 0.

9.4.19 ME m channel control register

The read only DMA active channel channel control register shows the configuration and control bits of the active DMA channel.

MEmCHCR (m=0-1) Offset address: $0474_H+m*200_H$

ME m channel control register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0			PRSEL	SWAP	PATSEL			CHDW			CHMODE	RROAT	BLKM		
r			rh	rh	rh			rh			rh	rh	rh		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		TREL													
r		rh													

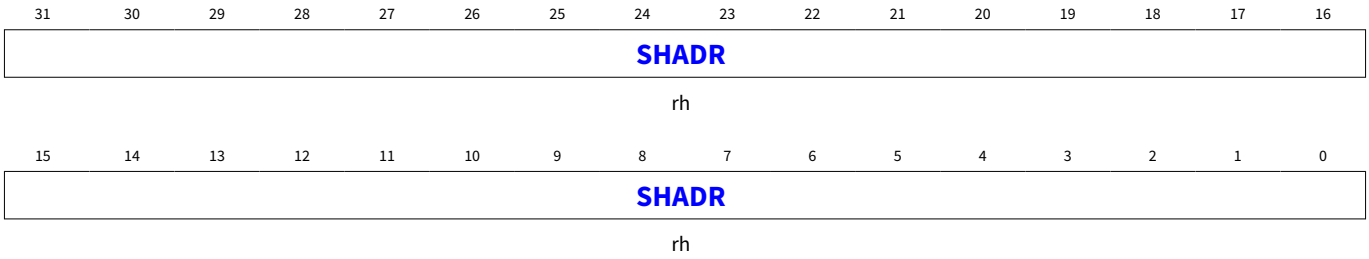
Field	Bits	Type	Description
TREL	13:0	rh	**Transfer reload value** Active DMA channel transfer reload value.
BLKM	18:16	rh	**Block mode** Active DMA channel block mode.
RROAT	19	rh	**Reset request only after transaction** Active DMA channel reset request only after transaction.
CHMODE	20	rh	**Channel operation mode** Active DMA channel channel operation mode.
CHDW	23:21	rh	**Channel data width** Active DMA channel DMA move channel data width.
PATSEL	26:24	rh	**Pattern select** Active DMA channel pattern select control.
SWAP	27	rh	**Swap data CRC byte order** Active DMA channel swap data CRC byte order.
PRSEL	28	rh	**Peripheral request select** Active DMA channel peripheral request select.
0	15:14, 31:29	r	**Reserved** Read as 0; should be written with 0.

9.4.20 ME m channel shadow address register

The read only DMA active channel shadow address register holds the 32-bit shadow address.

MEmSHADR (m=0-1) Offset address: $0478_H+m*200_H$

ME m channel shadow address register Application Reset value: $0000\ 0000_H$



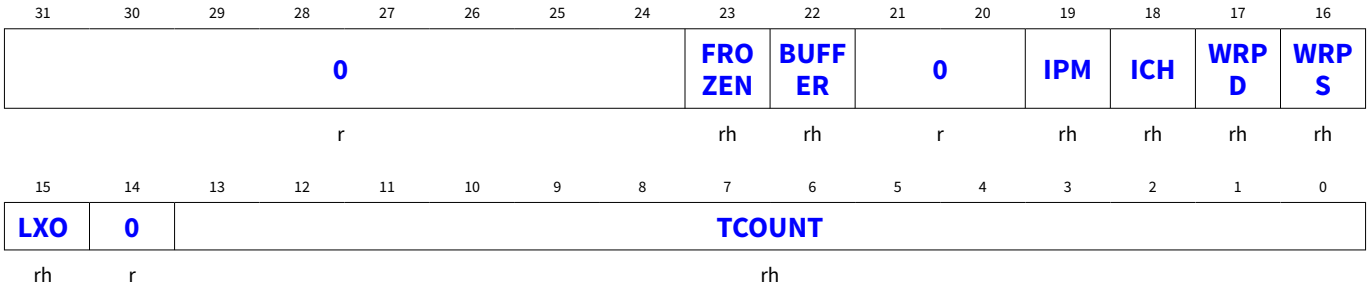
Field	Bits	Type	Description
SHADR	31:0	rh	**Shadowed address** This bit field holds the 32-bit shadow address of the active DMA channel. The function of the shadow address is set by the shadow control settings.

9.4.21 ME m channel status register

The read only DMA active channel channel status register contains the current transfer count, pattern detection compare result and the status of traffic management interrupt triggers.

MEmCHSR (m=0-1) Offset address: $047C_H+m*200_H$

ME m channel status register Application Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
TCOUNT	13:0	rh	**Transfer count status** Active DMA channel count of the number of DMA transfers. TCOUNT is loaded with the DMA channel value of CHCFGRc.TREL when TSRc.CH becomes set (and TCOUNT = 0). After each DMA transfer, TCOUNT is decremented by 1.
LXO	15	rh	**Old value of pattern detection** Active DMA channel compare result of a pattern compare operation when 8-bit or 16-bit data width is selected.
WRPS	16	rh	**Wrap source buffer** Active DMA channel wrap source buffer status bit.
WRPD	17	rh	**Wrap destination buffer** Active DMA channel wrap destination buffer status bit.
ICH	18	rh	**Interrupt from channel** Active DMA channel detection of channel interrupt service request.

(table continues...)

(continued)

Field	Bits	Type	Description
IPM	19	rh	**Pattern detection from channel** Active DMA channel detection of pattern match interrupt service request.
BUFFER	22	rh	**DMA double buffering active buffer** Active DMA channel DMA double buffering active buffer status bit. 0_B Buffer 0 read or filled by DMA. 1_B Buffer 1 read or filled by DMA.
FROZEN	23	rh	**DMA double buffering frozen buffer** Active DMA channel DMA double buffering frozen buffer status bit.
0	14, 21:20, 31:24	r	**Reserved** Read as 0; should be written with 0.

9.4.22 RP r write access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

ACCENRPrWRA (r=0-15) Offset address: 1000_H+r*80_H

RP r write access enable register A Application Reset value: $1000\ 0003_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

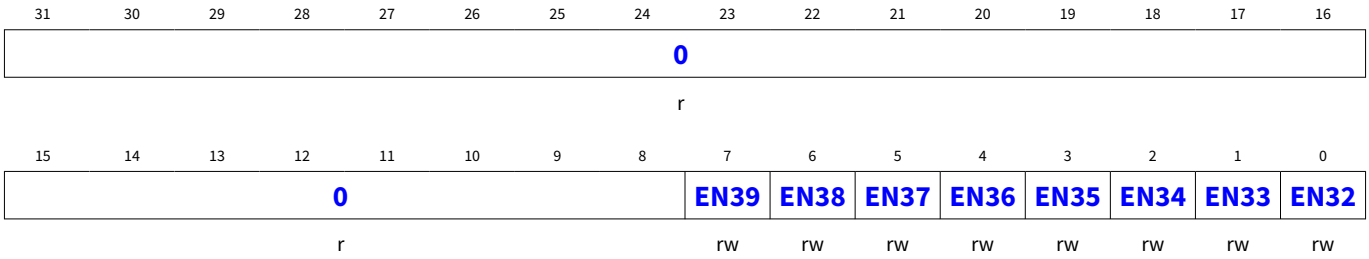
Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access

9.4.23 RP r write access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

ACCENRPrWRB (r=0-15) Offset address: 1004_H+r*80_H

RP r write access enable register B Application Reset value: $0000\ 0000_H$



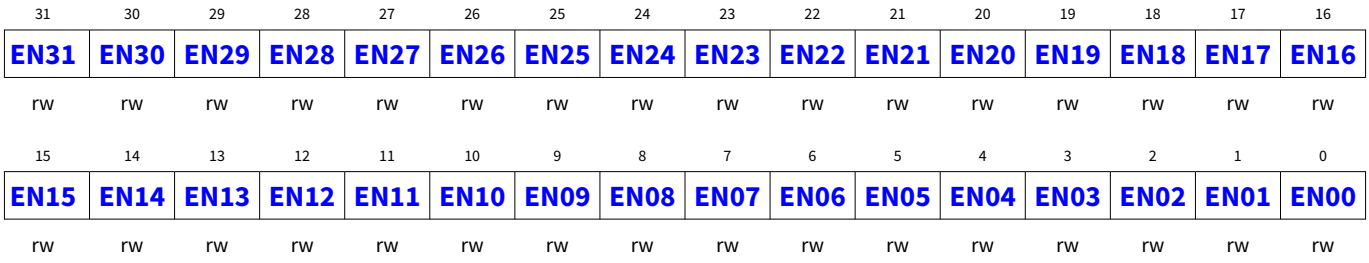
Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access
0	31:8	r	**Reserved** Read as 0; should be written with 0.

9.4.24 RP r read access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

ACCENRPrRDA (r=0-15) Offset address: 1008_H+r*80_H

RP r read access enable register A Application Reset value: FFFF $FFFF_H$



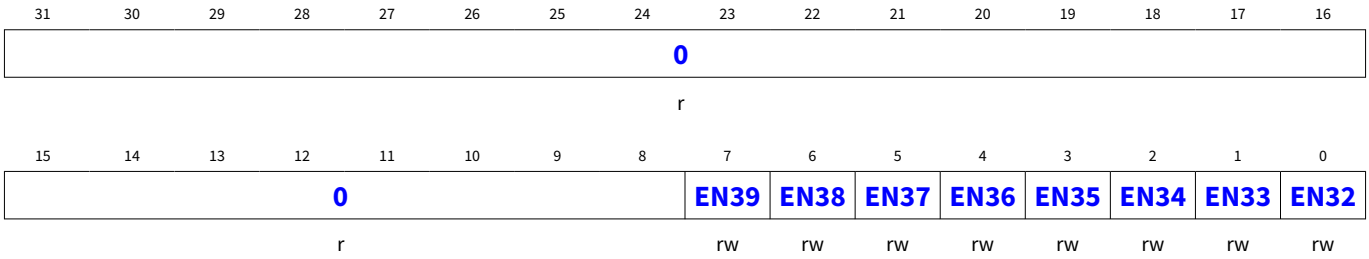
Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access

9.4.25 RP r read access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

ACCENRPrRDB (r=0-15) Offset address: $100C_H+r*80_H$

RP r read access enable register B Application Reset value: 0000 $00FF_H$



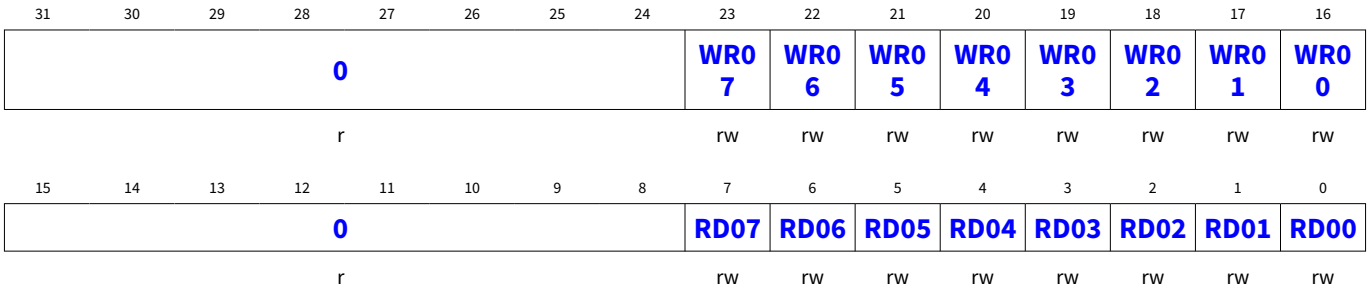
Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access
0	31:8	r	**Reserved** Read as 0; should be written with 0.

9.4.26 RP r VM access enable register

This register defines which virtual machine encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each VMx.

ACCENRPrVM (r=0-15) Offset address: 1010_H+r*80_H

RP r VM access enable register Application Reset value: $00FF\ 00FF_H$



Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for VM ID q** This bit enables read access to the access protected region for transactions with the VM ID q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for VM ID q** This bit enables write access to the access protected region for transactions with the VM ID q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

9.4.27 RP r PRS access enable register

This register defines which protection register set encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each PRSx.

ACCENRPrPRS (r=0-15) Offset address: 1014_H+r*80_H

RP r PRS access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for PRS q** This bit enables read access to the access protected region for transactions with the PRS q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for PRS q** This bit enables write access to the access protected region for transactions with the PRS q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

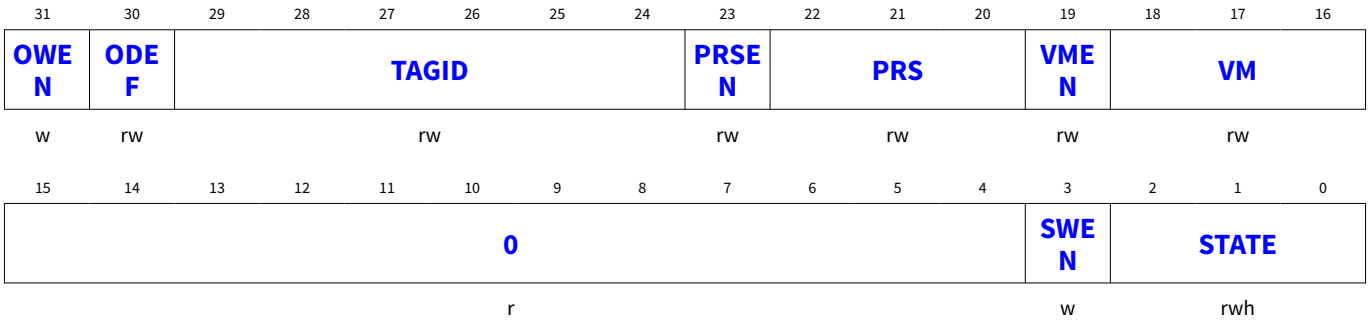
9.4.28 PROT register RP r

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner and in some PROT states, the secure master to update the PROT state as described in the PROT mechanism.

This register provides assignment and write protection to the RP registers as well as the channel registers for channels assigned to the RP. The RP PROT owner can assign or deassign channels to that RP, set up the APU for the channels in the RP etc.

PROTRPr (r=0-15) Offset address: 1020_H+r*80_H

PROT register RP r Application Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
VM	18:16	rw	**Virtual machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual machine definition enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.
PRSEN	23	rw	**Protection set definition enable for PROT owner** 0_B **Disable**: PRS is not part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG-ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner

(table continues...)

(continued)

Field	Bits	Type	Description
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; should be written with 0.

Table 649 Access mode restrictions of PROTRPr (r=0-15) sorted by descending priority

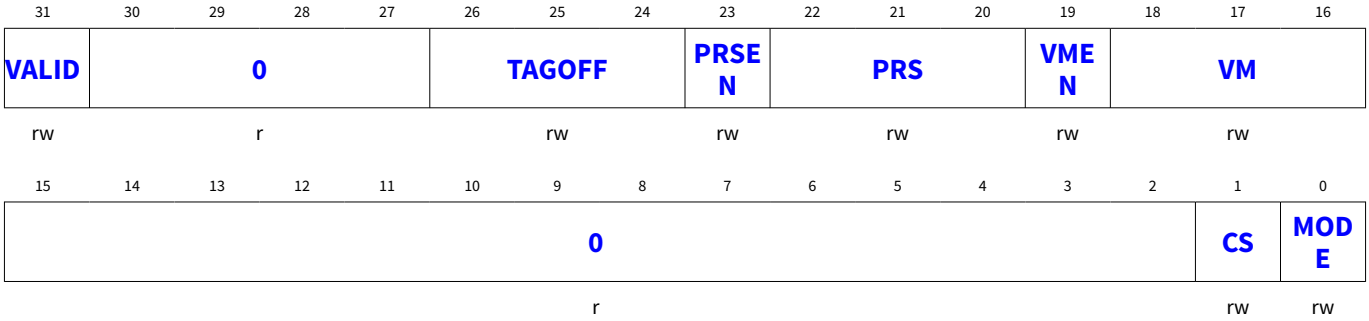
Mode name	Access mode		Description
write 1 to .**SWEN**	rwh	STATE	
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

9.4.29 RP r mode register

The mode register defines the attributes used by the DMA system interconnect master interfaces when a channel in the RP accesses the system interconnect. See section "Resource partition configuration" for details of the module specific access protection rules for this register.

MODEr (r=0-15) Offset address: 1030_H+r*80_H

RP r mode register Application Reset value: $0000\ 0001_H$



Field	Bits	Type	Description
MODE	0	rw	**Resource partition supervisor mode** 0_B System interconnect master interface accesses system interconnect in user mode. 1_B System interconnect master interface accesses system interconnect in supervisor mode.

(table continues...)

(continued)

Field	Bits	Type	Description
CS	1	rw	**Cyber-security assignment** 0_B Resource partition is marked as a non-CS RP. 1_B Resource partition is marked as a CS RP.
VM	18:16	rw	**Virtual machine information**
VMEN	19	rw	**Virtual machine enable** 0_B **Disable**: VM disabled 1_B **Enable**: VM enabled
PRS	22:20	rw	**Protection set information**
PRSEN	23	rw	**Protection set enable** 0_B **Disable**: PRS disabled 1_B **Enable**: PRS enabled
TAGOFF	26:24	rw	**Tag offset** Selection of TAG-ID to be used by DMA channels assigned to this RP when accessing the system interconnect
VALID	31	rw	**Resource partition valid** 0_B Resource partition is disabled. 1_B Resource partition is enabled.
0	15:2, 30:27	r	**Reserved** Read as 0; should be written with 0.

9.4.30 RP r pattern read register 0

The pattern read register 0 stores the pattern data (mask and compare bits) to be processed by the pattern compare logic.

PRRAr (r=0-15) Offset address: 1038_H+r*80_H

RP r pattern read register 0 Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PAT03								PAT02							
rw								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PAT01								PAT00							
rw								rw							

Field	Bits	Type	Description
PAT00	7:0	rw	**Pattern data byte**
PAT01	15:8	rw	**Pattern data byte**
PAT02	23:16	rw	**Pattern data byte**
PAT03	31:24	rw	**Pattern data byte**

9.4.31 RP r pattern read register 1

The pattern read register 1 stores the pattern data (mask and compare bits) to be processed by the pattern compare logic.

PRRBr (r=0-15) Offset address: $103C_H+r*80_H$

RP r pattern read register 1 Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PAT13								PAT12							
rw								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PAT11								PAT10							
rw								rw							

Field	Bits	Type	Description
PAT10	7:0	rw	**Pattern data byte**
PAT11	15:8	rw	**Pattern data byte**
PAT12	23:16	rw	**Pattern data byte**
PAT13	31:24	rw	**Pattern data byte**

9.4.32 RP r error interrupt set register

ERRINTRr (r=0-15) Offset address: 1040_H+r*80_H

RP r error interrupt set register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															SIT
r															w

Field	Bits	Type	Description
SIT	0	w	**Set error interrupt service request** Reading this bit returns a 0. 0_B No action. 1_B DMA error interrupt service request will be activated.
0	31:1	r	**Reserved** Read as 0; should be written with 0.

9.4.33 RP r enable error register

The enable error register enables an error interrupt service request for RP errors.

EERr (r=0-15) Offset address: 1044_H+r*80_H

RP r enable error register Application Reset value: $2403\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		ECDTER	0		ELER	0								EDER	ESER
r		rw	r		rw	r								rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															
r															

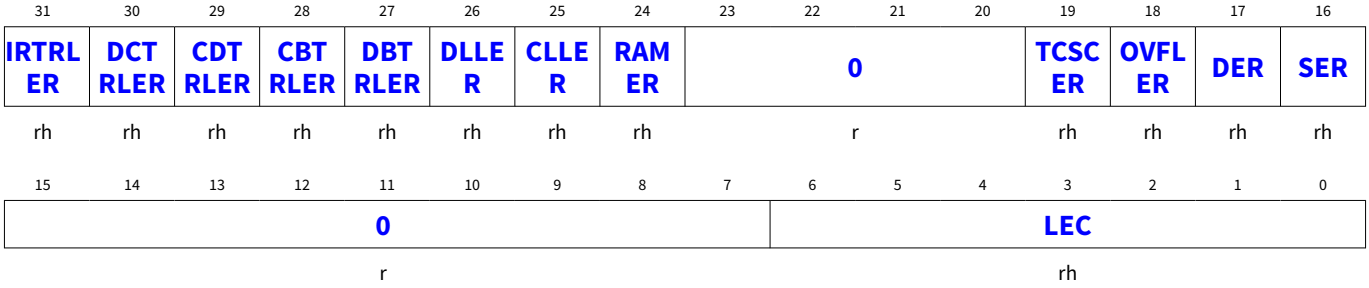
Field	Bits	Type	Description
ESER	16	rw	**Enable RP source error** This bit enables the generation of a RP source error interrupt. 0_B RP source error interrupt is disabled. 1_B RP source error interrupt is enabled.
EDER	17	rw	**Enable RP destination error** This bit enables the generation of a RP destination error interrupt. 0_B RP destination error interrupt is disabled. 1_B RP destination error interrupt is enabled.
ELER	26	rw	**Enable RP DMA linked list error** This bit enables the generation of a RP DMA linked list error interrupt. 0_B RP DMA linked list error interrupt is disabled. 1_B RP DMA linked list error interrupt is enabled.
ECDTER	29	rw	**Enable RP channel disabled TRL error** This bit enables the generation of a RP channel disabled TRL error. 0_B RP channel disabled TRL error is disabled. 1_B RP channel disabled TRL error is enabled.
0	15:0, 25:18, 28:27, 31:30	r	**Reserved** Read as 0; should be written with 0.

9.4.34 RP r error status register

The error status register indicates if the DMA controller could not service a DMA request in a particular RP. The channel number of the channel that last triggered an error is recorded in ERRSRr.LEC.

ERRSRr (r=0-15) Offset address: 1048_H+r*80_H

RP r error status register Application Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
LEC	6:0	rh	**RP last error channel** This bit field indicates the DMA channel number of the last DMA channel in the RP generating an error reported in the ERRSRr register.
SER	16	rh	**RP source error** This bit is set whenever an RP error occurs during a source (read) move of a DMA transfer, or a request cannot be serviced due to the access protection. 0_B No RP source error has occurred. 1_B RP source error has occurred.
DER	17	rh	**RP destination error** This bit is set whenever an RP error occurs during a destination (write) move of a DMA transfer, or a request cannot be serviced due to the access protection. 0_B No RP destination error has occurred. 1_B RP destination error has occurred.
OVFLER	18	rh	**RP error status overflow** This bit is set whenever the RP error status overflows - that is, when any previously reported RP error status remains set in the ERRSRr register and a new RP error is reported. 0_B No error overflow occurred. 1_B An error overflow occurred.
TCSCER	19	rh	**RP TCS configuration error** This bit is set when a TCS configuration error is detected as follows: • The SHADRc register is written to when the channel is in DMA shadow operation read only mode. 0_B No RP TCS configuration error has occurred. 1_B RP TCS configuration error has occurred.
RAMER	24	rh	**RP RAM error** This bit is set whenever an RP error occurs during the loading of a TCS from the DMARAM to the ME channel registers. 0_B No error occurred. 1_B An error occurred during the load of a TCS.

(table continues...)

(continued)

Field	Bits	Type	Description
CLLER	25	rh	**RP checked linked list error** This bit is set when an RP error occurs during the comparison of a SDCRC checksum as part of a checked linked list operation. 0_B No error occurred. 1_B An error occurred during the SDCRC checksum comparison.
DLLER	26	rh	**RP DMA linked list error** This bit is set when an RP error occurs during a DMALL, ACCLL, CHKLL or CONLL operation when a new TCS is loaded from memory to overwrite the current TCS stored in the DMARAM. 0_B No error occurred. 1_B An error occurred during the loading of a new TCS.
DBTRLER	27	rh	**Double buffer TRL error** This bit is set when a DMA channel is configured for a double buffering operation and one of the following conditions occur. • The DMA channel is configured for software switch only AND the DMA transaction is completed before the DMA receives a software switch. • The DMA channel is configured for automatic hardware switch AND the DMA transaction is completed when CHCSRc.FROZEN is still . 0_B No error occurred. 1_B A TRL error has occurred.
CBTRLER	28	rh	**Channel busy TRL error** This bit is set when a DMA request is detected and the DMA channel TSRc.CH is set. 0_B No error occurred. 1_B A TRL error has occurred.
CDTRLER	29	rh	**Channel disabled TRL error** This bit is set when a DMA hardware request is detected and the DMA channel is disabled for hardware requests. 0_B No error occurred. 1_B A TRL error has occurred.
DCTRLER	30	rh	**Daisy chain TRL error** This bit is set when a DMA daisy chain request from the current channel attempts to trigger the next lower priority channel assigned to a different resource partition (the TRL event is processed on the current channel). 0_B No error occurred. 1_B A TRL error has occurred.
IRTRLER	31	rh	**Invalid resource partition TRL error** This bit is set when a DMA request is detected and the resource partition that the DMA channel is assigned to is not configured (MODEr.VALID is set to 0_B). 0_B No error occurred. 1_B A TRL error has occurred.

(table continues...)

(continued)

Field	Bits	Type	Description
0	15:7, 23:20	r	**Reserved** Read as 0; should be written with 0.

9.4.35 RP r clear error register

The clear error register contains bits to clear the corresponding RP error flags.

CLREr (r=0-15) Offset address: $104C_H+r*80_H$

RP r clear error register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CIRTRLER	CDCTRLER	CCDTRLER	CCBTRLER	CDBTRLER	CDLLER	CCLLER	CRAMER	0				CTCSCER	COVFLER	CDER	CSER
w	w	w	w	w	w	w	w	r				w	w	w	w

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															
r															

Field	Bits	Type	Description
CSER	16	w	**Clear RP source error** 0_B No action 1_B Clear source error flag ERRSRr.SER.
CDER	17	w	**Clear RP destination error** 0_B No action 1_B Clear destination error flag ERRSRr.DER.
COVFLER	18	w	**Clear RP error status overflow** 0_B No action 1_B Clear RP error status overflow flag ERRSRr.OVFLER.
CTCSCER	19	w	**Clear RP TCS configuration error** 0_B No action 1_B Clear TCS configuration error flag ERRSRr.TCSCER.
CRAMER	24	w	**Clear RAM error** 0_B No action 1_B Clear error flag ERRSRr.RAMER.
CCLLER	25	w	**Clear CLL error** 0_B No action 1_B Clear error flag ERRSRr.CLLER.
CDLLER	26	w	**Clear DLL error** 0_B No action 1_B Clear error flag ERRSRr.DLLER.

(table continues...)

(continued)

Field	Bits	Type	Description
CDBTRLER	27	w	**Clear double buffer TRL error** 0_B No action 1_B Clear RP double buffer TRL error flag ERRSRr.DBTRLER.
CCBTRLER	28	w	**Clear channel busy TRL error** 0_B No action 1_B Clear RP channel busy TRL error flag ERRSRr.CBTRLER.
CCDTRLER	29	w	**Clear channel disabled TRL error** 0_B No action 1_B Clear RP channel disabled TRL error flag ERRSRr.CDTRLER.
CDCTRLER	30	w	**Clear daisy chain TRL error** 0_B No action 1_B Clear RP daisy chain TRL error flag ERRSRr.DCTRLER.
CIRTRLER	31	w	**Clear invalid resource partition TRL error** 0_B No action 1_B Clear RP invalid resource partition TRL error flag ERRSRr.IRTRLER.
0	15:0, 23:20	r	**Reserved** Read as 0; should be written with 0.

9.4.36 DMA channel c resource partition assignment register

The DMA channel resource partition assignment register assigns a dma channel to a resource partition. Refer to section "DMA channel assignment" for the module specific access protection rules on this register.

HRRc (c=000-127) Offset address: 1800_H+c*4

DMA channel c resource partition assignment register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0											HRPV	HRP			
r											rw	rw			

Field	Bits	Type	Description
HRP	3:0	rw	**DMA channel resource partition assignment** 0_H Resource Partition 0 (RP0). 1_H Resource Partition 1 (RP1). 2_H Resource Partition 2 (RP2). 3_H Resource Partition 3 (RP3). 4_H Resource Partition 4 (RP4). 5_H Resource Partition 5 (RP5). 6_H Resource Partition 6 (RP6). 7_H Resource Partition 7 (RP7). 8_H Resource Partition 8 (RP8). 9_H Resource Partition 9 (RP9). A_H Resource Partition 10 (RP10). B_H Resource Partition 11 (RP11). C_H Resource Partition 12 (RP12). D_H Resource Partition 13 (RP13). E_H Resource Partition 14 (RP14). F_H Resource Partition 15 (RP15).
HRPV	4	rw	**DMA channel resource partition assignment valid** 0_B Channel is unassigned. Channel registers other than HRRc cannot be accessed. 1_B Channel is assigned to the resource partition specified in HRP. Channel registers are protected by the resource partition APU.
0	31:5	r	**Reserved** Read as 0; should be written with 0.

9.4.37 DMA channel c suspend enable register

The suspend enable register enables or disables soft suspend state capability. This register is write-protected by the PROTRP register in the RP to which the current channel is assigned.

SUSENRc (c=000-127) Offset address: $1A00_H$+c*4

DMA channel c suspend enable register Debug Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															SUSEN
r															rw

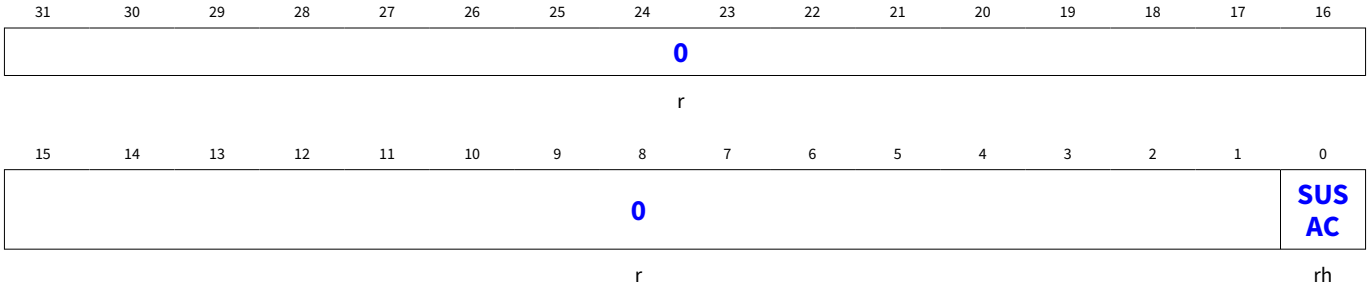
Field	Bits	Type	Description
SUSEN	0	rw	**Channel suspend enable for DMA channel** 0_B DMA channel is disabled for DMA channel suspend. The DMA channel does not react on an active suspend request signal (Suspend REQ). 1_B DMA channel is enabled for DMA channel suspend. If the suspend request signal (Suspend REQ) becomes active, a DMA transaction of the DMA channel is stopped after the current DMA transfer has completed.
0	31:1	r	**Reserved** Read as 0; should be written with 0.

9.4.38 DMA channel c suspend acknowledge register

The suspend acknowledge register indicates the DMA channel soft suspend status.

SUSACRc (c=000-127) Offset address: $1C00_H$+c*4

DMA channel c suspend acknowledge register Debug Reset value: $0000\ 0000_H$



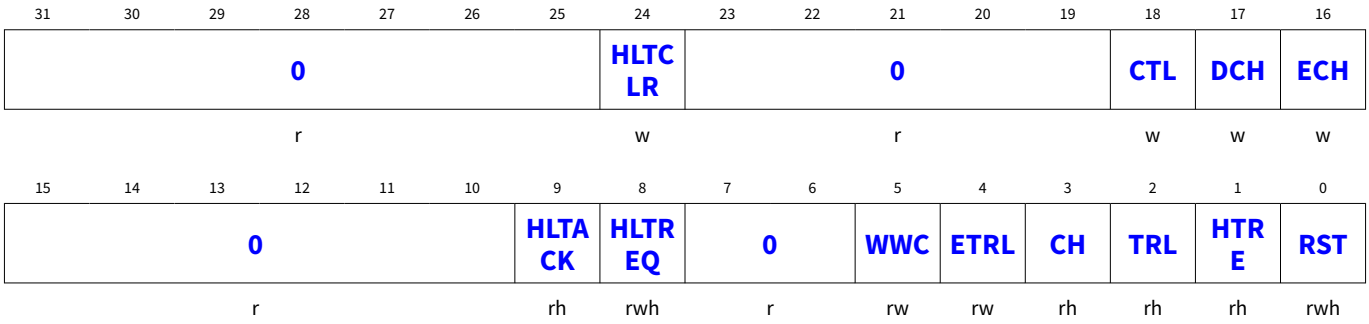
Field	Bits	Type	Description
SUSAC	0	rh	**DMA channel suspend state or frozen state active for DMA channel** Indicates when a DMA channel is in channel suspend state. 0_B DMA channel is not in channel suspend state, frozen state or internal actions are not completed after the channel suspend state or frozen state was requested. 1_B DMA channel is in channel suspend state or frozen state.
0	31:1	r	**Reserved** Read as 0; should be written with 0.

9.4.39 DMA channel c transaction state register

This register is protected by the APU in the RP to which the current channel is assigned.

TSRc (c=000-127) Offset address: $1E00_H$+c*4

DMA channel c transaction state register Application Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
RST	0	rwh	**DMA channel reset** The DMA channel reset bit is set by software (DMA channel TSRc.RST=1_B) and cleared by hardware when the DMA channel has been reset. 0_B After the application of a DMA channel reset, the DMA channel is in the DMA channel reset state. Software write to 0 has no effect. 1_B A DMA channel reset is pending.
HTRE	1	rh	**DMA channel hardware request enable** 0_B DMA hardware request is disabled for DMA channel. 1_B DMA hardware request is enabled for DMA channel.
TRL	2	rh	**DMA channel transaction request lost** This bit is set by hardware in response to a TRL event. This bit is reset by software clearing TRL (writing DMA channel TSRc.CTL = 1_B) or resetting the DMA channel (writing DMA channel TSRc.RST = 1_B). 0_B No TRL event has been detected for DMA channel. 1_B TRL event has been detected for DMA channel.
CH	3	rh	**DMA channel transaction request state** Indicates if a DMA request is pending. 0_B No DMA request is pending for DMA channel. 1_B DMA request is pending for DMA channel.
ETRL	4	rw	**Enable DMA channel transaction request lost interrupt** DMA channel control bit to enable the generation of an error interrupt service request when DMA channel TSRc.TRL is set. 0_B Interrupt generation for DMA channel TRL event is disabled. 1_B Interrupt generation for DMA channel TRL event is enabled.
WWC	5	rw	**Wait for write completion** DMA channel control bit to enable waiting for write completion signaling from the system interconnect before the generation of a DMA channel interrupt service request. 0_B **DISABLE**: DMA channel does not wait for write completion signaling 1_B **ENABLE**: DMA channel waits for write completion signaling

(table continues...)

(continued)

Field	Bits	Type	Description
HLTREQ	8	rwh	**DMA channel halt request** A DMA channel may be halted by software writing the DMA channel halt request bit (TSRc.HLTREQ = 1_B). Software can check when the halt request has completed by polling on the TSRc.HLTACK bit. 0_B No action. 1_B Halt request.
HLTACK	9	rh	**DMA channel halt acknowledge** 0_B DMA channel is not halted. 1_B DMA channel is halted.
ECH	16	w	**Enable DMA channel hardware transaction request** Setting DMA channel TSRc.ECH to 1_B enables DMA hardware requests (TSRc.HTRE = 1_B) for the DMA channel. Reading this bit returns a 0.
DCH	17	w	**Disable DMA channel hardware transaction request** Setting DMA channel TSRc.DCH to 1_B disables DMA hardware requests (TSRc.HTRE = 0_B) for the DMA channel. Reading this bit returns a 0.
CTL	18	w	**Clear DMA channel transaction request lost** Software clear of the DMA channel TRL status flag. Reading this bit returns a 0. 0_B No action. 1_B Clear DMA channel TRL flag (TSRc.TRL).
HLTCLR	24	w	**Clear DMA channel halt request and acknowledge** A DMA channel is released from the halt state by software writing the DMA channel halt clear bit (TSRc.HLTCLR = 1_B). The DMA operation is resumed. If the halt request is cleared before it is acknowledged then there is no effect on DMA operation. 0_B No action. 1_B Clear DMA channel halt request (TSRc.HLTREQ) and halt acknowledge (TSRc.HLTACK).
0	7:6, 15:10, 23:19, 31:25	r	**Reserved** Read as 0; should be written with 0.

9.4.40 DMA channel c read data CRC register

This register is protected by the APU in the RP to which the current channel is assigned.

RDCRCRc (c=000-127) Offset address: 2000_H+c*20_H

DMA channel c read data CRC register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RDCRC															
rwh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RDCRC															
rwh															

Field	Bits	Type	Description
RDCRC	31:0	rwh	**Read data CRC** Checksum calculated for DMA read move data.

9.4.41 DMA channel c source and destination address CRC register

This register is protected by the APU in the RP to which the current channel is assigned.

SDCRCRc (c=000-127) Offset address: 2004_H+c*20_H

DMA channel c source and destination address CRC register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SDCRC															
rwh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SDCRC															
rwh															

Field	Bits	Type	Description
SDCRC	31:0	rwh	**Source and destination address CRC** Checksum calculated for DMA move source and destination addresses.

9.4.42 DMA channel c source address register

This register is protected by the APU in the RP to which the current channel is assigned.

SADRc (c=000-127) Offset address: 2008_H+c*20_H

DMA channel c source address register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SADR															
rwh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SADR															
rwh															

Field	Bits	Type	Description
SADR	31:0	rwh	**Source address** 32-bit source address.

9.4.43 DMA channel c destination address register

This register is protected by the APU in the RP to which the current channel is assigned.

DADRc (c=000-127) Offset address: $200C_H+c*20_H$

DMA channel c destination address register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DADR															
rwh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DADR															
rwh															

Field	Bits	Type	Description
DADR	31:0	rwh	**Destination address** 32-bit destination address.

9.4.44 DMA channel c address and interrupt control register

This register is protected by the APU in the RP to which the current channel is assigned.

ADICRc (c=000-127) Offset address: 2010_H+c*20_H

DMA channel c address and interrupt control register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
IRDV				INTCT		WRPDE	WRPSE	0	STAMP	DCBE	SCBE	SHCT			
rwh				rwh		rwh	rwh	r	rwh	rwh	rwh	rwh			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CBLD				CBLS				INCD	DMF			INCS	SMF		
rwh				rwh				rwh	rwh			rwh	rwh		

Field	Bits	Type	Description
SMF	2:0	rwh	**Source address modification factor** DMA channel TCS 32-bit source address modification factor and the channel data width CHDW determines an address offset value by which the source address is modified after each DMA move. If SCBE = 1_B and CBLS = 0000_B then the source address is not modified. 000_B Address offset is 1 x CHDW 001_B Address offset is 2 x CHDW 010_B Address offset is 4 x CHDW 011_B Address offset is 8 x CHDW 100_B Address offset is 16 x CHDW 101_B Address offset is 32 x CHDW 110_B Address offset is 64 x CHDW 111_B Address offset is 128 x CHDW
INCS	3	rwh	**Increment of source address** DMA channel TCS control bit to determine if the address offset selected by SMF will be added to or subtracted from the source address after each DMA move. If SCBE = 1_B and CBLS = 0000_B then the source address is not modified. 0_B Address offset will be subtracted. 1_B Address offset will be added.
DMF	6:4	rwh	**Destination address modification factor** DMA channel TCS 32-bit destination address modification factor and the channel data width CHDW determines an address offset value by which the destination address is modified after each DMA move. If DCBE = 1_B and CBLD = 0000_B then the destination address is not modified. 000_B Address offset is 1 x CHDW 001_B Address offset is 2 x CHDW 010_B Address offset is 4 x CHDW 011_B Address offset is 8 x CHDW 100_B Address offset is 16 x CHDW 101_B Address offset is 32 x CHDW 110_B Address offset is 64 x CHDW 111_B Address offset is 128 x CHDW
INCD	7	rwh	**Increment of destination address** DMA channel TCS control bit to determine if the address offset selected by DMF will be added to or subtracted from the destination address after each DMA move. If DCBE = 1_B and CBLD = 0000_B the destination address is not modified. 0_B Address offset will be subtracted. 1_B Address offset will be added.

(table continues...)

(continued)

Field	Bits	Type	Description
CBLS	11:8	rwh	**Source circular buffer length** DMA channel TCS circular buffer source address update control bit determines which part of the 32-bit source address register remains unchanged and is not updated after a DMA move operation. ***Note**: CBLS determines the size of the circular source buffer.* 0_H Source address SADRc[31:0] is not updated 1_H Source address SADRc[31:1] is not updated 2_H Source address SADRc[31:2] is not updated 3_H Source address SADRc[31:3] is not updated 4_H Source address SADRc[31:4] is not updated 5_H Source address SADRc[31:5] is not updated 6_H Source address SADRc[31:6] is not updated 7_H Source address SADRc[31:7] is not updated 8_H Source address SADRc[31:8] is not updated 9_H Source address SADRc[31:9] is not updated A_H Source address SADRc[31:10] is not updated B_H Source address SADRc[31:11] is not updated C_H Source address SADRc[31:12] is not updated D_H Source address SADRc[31:13] is not updated E_H Source address SADRc[31:14] is not updated F_H Source address SADRc[31:15] is not updated
CBLD	15:12	rwh	**Destination circular buffer length** DMA channel TCS circular buffer destination address update control bit determines which part of the 32-bit destination address register remains unchanged and is not updated after a DMA move operation. ***Note**: CBLD determines the size of the circular destination buffer.* 0_H Destination address DADRc[31:0] is not updated 1_H Destination address DADRc[31:1] is not updated 2_H Destination address DADRc[31:2] is not updated 3_H Destination address DADRc[31:3] is not updated 4_H Destination address DADRc[31:4] is not updated 5_H Destination address DADRc[31:5] is not updated 6_H Destination address DADRc[31:6] is not updated 7_H Destination address DADRc[31:7] is not updated 8_H Destination address DADRc[31:8] is not updated 9_H Destination address DADRc[31:9] is not updated A_H Destination address DADRc[31:10] is not updated B_H Destination address DADRc[31:11] is not updated C_H Destination address DADRc[31:12] is not updated D_H Destination address DADRc[31:13] is not updated E_H Destination address DADRc[31:14] is not updated F_H Destination address DADRc[31:15] is not updated

(table continues...)

(continued)

Field	Bits	Type	Description
SHCT	19:16	rwh	**Shadow control** DMA channel TCS shadow control determines the function of the shadow address register. 0_H Move Operation. 1_H Shadow operation read only mode source address. 2_H Shadow operation read only mode destination address. 3_H Reserved. 4_H Reserved. 5_H Shadow operation direct write mode source address. 6_H Shadow operation direct write mode destination address. 7_H Reserved. 8_H DMA double source buffering with software switch only. 9_H DMA double source buffering with software switch and automatic hardware switch. A_H DMA double destination buffering with software switch only. B_H DMA double destination buffering with software switch and automatic hardware switch. C_H DMA linked list (DMALL). D_H Accumulated linked list (ACCLL). E_H Checked linked list (CHKLL). F_H Conditional linked list (CONLL).
SCBE	20	rwh	**Source circular buffer enable** DMA channel TCS source circular buffer enable. 0_B Source circular buffer disabled. 1_B Source circular buffer enabled.
DCBE	21	rwh	**Destination circular buffer enable** DMA channel TCS destination circular buffer enable. 0_B Destination circular buffer disabled. 1_B Destination circular buffer enabled.
STAMP	22	rwh	**Timestamp** DMA channel TCS control bit to enable the appendage of a timestamp after the end of the last DMA move during a DMA transaction. 0_B No action. 1_B DMA timestamp is appended.
WRPSE	24	rwh	**Wrap source enable** DMA channel TCS source buffer interrupt trigger enable or disable. 0_B Wrap source buffer interrupt trigger disabled. 1_B Wrap source buffer interrupt trigger enabled.
WRPDE	25	rwh	**Wrap destination enable** DMA channel TCS destination buffer interrupt trigger enable or disable. 0_B Wrap destination buffer interrupt trigger disabled. 1_B Wrap destination buffer interrupt trigger enabled.

(table continues...)

(continued)

Field	Bits	Type	Description
INTCT	27:26	rwh	**Interrupt control** DMA channel TCS interrupt control. ***Note**: If DMA channel CHCFGRc.PRSEL = 1_B for the next lower priority channel then the channel transfer trigger interrupt is disabled.* 00_B No interrupt trigger will be generated on changing the TCOUNT value. The DMA channel CHSR.ICH is set when TCOUNT equals IRDV. 01_B No interrupt trigger will be generated on changing the TCOUNT value. The DMA channel CHSR.ICH is set when TCOUNT is decremented 10_B Interrupt trigger is generated and DMA channel CHSR.ICH is set on changing the TCOUNT value and TCOUNT equals IRDV 11_B Interrupt trigger is generated and DMA channel CHSR.ICH is set each time TCOUNT is decremented
IRDV	31:28	rwh	**Interrupt raise detect value** DMA channel TCS interrupt threshold value defines the threshold limit of CHSR.TCOUNT for which a channel interrupt trigger will be raised.
0	23	r	**Reserved** Read as 0; should be written with 0.

9.4.45 DMA channel c configuration register

This register is protected by the APU in the RP to which the current channel is assigned.

CHCFGRc (c=000-127) Offset address: 2014_H+c*20_H

DMA channel c configuration register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0			PRSEL	SWAP	PATSEL			CHDW			CHMODE	RROAT	BLKM		
r			rwh	rwh	rwh			rwh			rwh	rwh	rwh		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		TREL													
r		rwh													

Field	Bits	Type	Description
TREL	13:0	rwh	**Transfer reload value** DMA channel TCS transfer reload value to control the number of DMA transfers in a DMA transaction. The 14-bit transfer count value is loaded into ME CHSR.TCOUNT at the start of a DMA transaction (when TSRc.CH becomes set and CHSR.TCOUNT = 0). A write to CHCFGRc.TREL during a running DMA transaction has no influence on the running DMA transaction. If CHCFGRc.TREL = 0 or if CHCFGRc.TREL = 1 then ME CHSR.TCOUNT will be loaded with 1 when a new DMA transaction is started (at least one DMA transfer must be executed per DMA transaction).
BLKM	18:16	rwh	**Block mode** Defines the number of DMA moves executed during one DMA transfer. 000_B One DMA transfer has 1 DMA move 001_B One DMA transfer has 2 DMA moves 010_B One DMA transfer has 4 DMA moves 011_B One DMA transfer has 8 DMA moves 100_B One DMA transfer has 16 DMA moves 101_B One DMA transfer has 3 DMA moves 110_B One DMA transfer has 5 DMA moves 111_B One DMA transfer has 9 DMA moves
RROAT	19	rwh	**Reset request only after transaction** DMA channel control bit to determine if the DMA request state flag (DMA channel TSRc.CH) is reset after each DMA transfer. 0_B DMA channel TSRc.CH is reset after the start of each DMA transfer. A DMA request is required for each DMA transfer. 1_B DMA channel TSRc.CH is reset when CHSR.TCOUNT = 0 and after the completion of the last DMA transfer (i.e. on completion of the DMA transaction). One DMA request starts a complete DMA transaction.
CHMODE	20	rwh	**Channel operation mode** DMA channel TCS control to determine TSRc.HTRE reset condition. 0_B **Single_Mode**: At the start of the last DMA trasfter of a DMA transaction, DMA channel is disabled for further hardware requests (TSRc.HTRE is reset by hardware). TSRc.HTRE must be set again by software for starting a new transaction. 1_B **Continuous_Mode**: TSRc.HTRE is not reset at the start of the last DMA transfer of a DMA transaction.

(table continues...)

(continued)

Field	Bits	Type	Description
CHDW	23:21	rwh	**Channel data width** DMA channel TCS data width for DMA read moves and DMA write moves. 000_B 8-bit data width for moves selected Single Data Transfer Byte (SDTB) 001_B 16-bit data width for moves selected Single Data Transfer Half-Word (SDTH) 010_B 32-bit data width for moves selected Single Data Transfer Word (SDTW) 011_B 64-bit data width transaction selected SRI: Single Data Transfer Double Word (SDTD) FPI: Block Transfer Request - 2 transfers (BTR2) 100_B 128-bit data width transaction selected SRI: Block Transfer Request - 2 transfers (BTR2) FPI: Block Transfer Request - 4 transfers (BTR4) 101_B 256-bit data width transaction selected SRI: Block Transfer Request - 4 transfers (BTR4) FPI: Block Transfer Request - 8 transfers (BTR8) 110_B 512-bit data width transaction selected SRI: Block Transfer Request - 8 transfers (BTR8) FPI: Not supported. 111_B Reserved.
PATSEL	26:24	rwh	**Pattern select** DMA channel TCS bit field to select the pattern detection operation (see functional description). If PATSEL[1:0] is not equal to 00_B then a pattern detection operation defined by the channel data width (CHDW) will be performed. PATSEL[2] selects the pattern read register. If a pattern match is detected then a DMA channel interrupt shall be triggered.
SWAP	27	rwh	**Swap data CRC byte order** DMA channel TCS swap data CRC byte order. 0_B Byte order is not swapped 1_B Byte order is swapped.
PRSEL	28	rwh	**Peripheral request select** DMA channel TCS control bit field to select the source of a DMA request. 0_B DMA hardware request selected. 1_B DMA daisy chain request selected.
0	15:14, 31:29	r	**Reserved** Read as 0; should be written with 0.

9.4.46 DMA channel c shadow address register

This register is protected by the APU in the RP to which the current channel is assigned.

SHADRc (c=000-127) Offset address: 2018_H+c*20_H

DMA channel c shadow address register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SHADR															
rwh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SHADR															
rwh															

Field	Bits	Type	Description
SHADR	31:0	rwh	**Shadowed address** 32-bit shadow address.

9.4.47 DMA channel c control and status register

The CHCSRc bit fields are used for two functions:
1. Storing DMA channel status bits.
2. Write only triggers to set and clear DMA channel configuration and status.

This register is protected by the APU in the RP to which the current channel is assigned.

CHCSRc (c=000-127) Offset address: $201C_H+c*20_H$

DMA channel c control and status register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SCH	0			SIT	CICH	CWRP	SWB	FROZEN	BUFFER	0		IPM	ICH	WRPD	WRPS
w	r			w	w	w	w	rwh	rh	r		rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LXO	0	TCOUNT													
rh	r	rh													

Field	Bits	Type	Description
TCOUNT	13:0	rh	**Transfer count** DMA channel status transfer count updated after DMARAM write back.
LXO	15	rh	**Old value of pattern detection match status** DMA channel status bit to store the result of a previous pattern detection operation when 8-bit or 16-bit data width is selected. 0_B The corresponding pattern compare operation did not find a pattern match during the previous DMA read move. 1_B The corresponding pattern compare operation found a pattern match during the previous DMA read move.

(table continues...)

(continued)

Field	Bits	Type	Description
WRPS	16	rh	**Wrap source buffer** Status bit indicates that a DMA channel has reached a wrap source buffer boundary. Bit is reset by software writing DMA channel CHCSRc.CWRP = 1_B or DMA channel reset TSRc.RST = 1_B. 0_B No wrap source buffer occurred. 1_B Wrap source buffer occurred.
WRPD	17	rh	**Wrap destination buffer** Status bit indicates that a DMA channel has reached a wrap destination buffer boundary. Bit is reset by software writing DMA channel CHCSRc.CWRP = 1_B or DMA channel reset TSRc.RST = 1_B. 0_B No wrap destination buffer occurred. 1_B Wrap destination buffer occurred.
ICH	18	rh	**Interrupt from channel** Status bit indicates when a channel interrupt has been raised (see also ADICRc.INTCT, ADICRc.IRDV). Bit is reset by software writing DMA channel CHCSRc.CICH = 1_B or by a DMA channel reset TSRc.RST = 1_B. 0_B An interrupt from channel has not been detected. 1_B An interrupt from channel has been detected.
IPM	19	rh	**Pattern detection from channel** Status bit indicates that a pattern match has been detected for the DMA channel when pattern detection is enabled. This bit is reset by software writing DMA channel CHCSRc.CICH = 1_B or DMA channel reset TSRc.RST = 1_B. 0_B A pattern match has not been detected. 1_B A pattern match has been detected.
BUFFER	22	rh	**DMA double buffering active buffer** During a DMA double buffering operation, the status bit indicates which buffer is read or filled. 0_B Buffer 0 read or filled by DMA. 1_B Buffer 1 read or filled by DMA.
FROZEN	23	rwh	**DMA double buffering frozen buffer** If a DMA channel is configured for double buffering operation, the FROZEN bit indicates that one of the buffers is frozen and available for processing by a cyclic software task. ***Note**: FROZEN bit shall only be set by the hardware and shall only be cleared by software.* 0_B Buffer is not frozen. 1_B Buffer is frozen.

(table continues...)

(continued)

Field	Bits	Type	Description
SWB	24	w	**DMA double buffering switch buffer** When DMA double buffering is configured, the control bit is used to re-direct data from one buffer to the other buffer. ***Note**: If a DMALL, ACCLL, CHKLL or CONLL operation is configured then SWB shall be 0_B.* 0_B No action. 1_B Switch from buffer.
CWRP	25	w	**Clear wrap buffer interrupt** Software clear of the DMA channel source and destination wrap buffer flags stored at CHCSRc.WRPS and CHCSRc.WRPD. If the DMA channel is active in an ME then clear ME bit-fields CHSR.WRPS and CHSR.WRPD. Reading this bit returns a 0. 0_B No action. 1_B Clear DMA channel bits CSR.WRPS and CSR.WRPD.
CICH	26	w	**Clear interrupt for DMA channel** Software clear of the DMA channel flags stored at CHCSRc.ICH and CHCSRc.IPM. If the DMA channel is active in a ME then clear ME bit fields CHSR.ICH and CHSR.IPM. Reading this bit returns a 0. 0_B No action. 1_B Clear DMA channel bits CSR.ICH and CSR.IPM.
SIT	27	w	**Set interrupt trigger for DMA channel** Reading this bit returns a 0. ***Note**: If a DMALL, ACCLL, CHKLL or CONLL operation is configured then SIT must be 0_B.* 0_B No action. 1_B DMA channel interrupt trigger will be activated.
SCH	31	w	**Set transaction request** Triggers a DMA software request when written with a 1_B. All other TCS registers (RDCRCRc, SDCRCRc, SADRc, DADRc, ADICRc, CHCFGRc, SHADRc) should be written with the required channel configuration before the DMA software request is triggered. Reading this bit returns a 0. 0_B No action. 1_B DMA software request initiated for DMA channel. When SCH is set, DMA channel TSRc.CH is set to indicate a DMA request is pending.
0	14, 21:20, 30:28	r	**Reserved** Read as 0; should be written with 0.

9.5 Debug information

This section describes the following topics:

- DMA channel suspend
- Software activation of DMA channel interrupt service requests
- Software activation of DMA RP error interrupt service requests
- OCDS trigger bus (OTGB) interface
- MCDS trace interface

9.5.1 DMA channel suspend

Two DMA channel registers control and report the channel suspend operation:

- The suspend enable register (SUSENRc) enables and disables DMA channel soft suspend state capability
- The suspend acknowledge register (SUSACRc) indicates the DMA channel soft suspend status

The On Chip Debug System (OCDS) is able to generate a DMA suspend request (Suspend REQ). When the suspend request becomes active, the DMA channel enters the suspend state where the DMA channel state is frozen to ensure that it can be analyzed by reading the DMA channel register contents. If a DMA channel is active then the current DMA transfer is completed before the DMA channel enters the suspend state. The DMA signals the suspend state status back to the OCDS through a suspend acknowledge (Suspend ACK) signal.

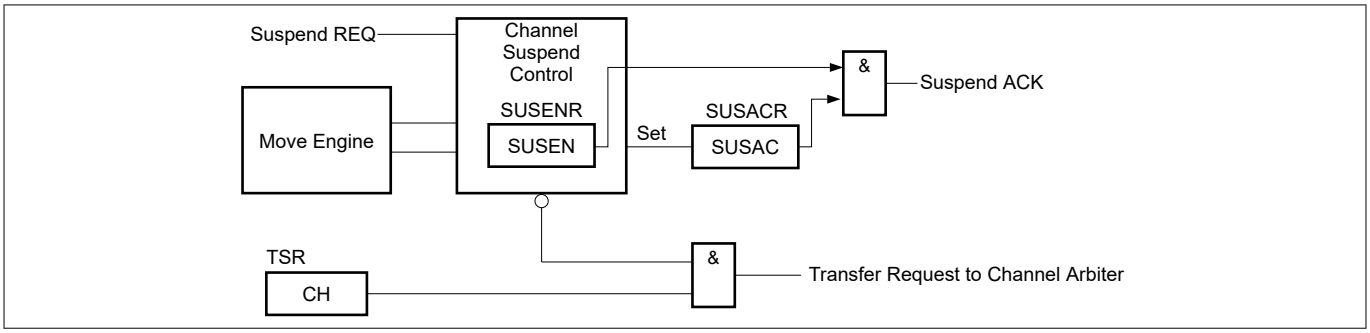


Figure 127 DMA channel suspend

Entering DMA channel suspend state

A DMA channel is enabled for suspend state when the DMA channel suspend enable bit in the Suspend State Enable Register SUSENRc.SUSEN is set. When Suspend REQ becomes active, for DMA channels enabled for DMA channel suspend, the operation is as follows:

- Idle state, reset state, halt state and pending state: The DMA channel suspend active status flag SUSACRc.SUSAC is set and the DMA channel enters the suspend state
- Active state: The DMA channel suspend active status flag SUSACRc.SUSAC is set when the current DMA transfer is complete, the updated TCS is written back to the DMARAM, and the DMA channel enters the suspend state

The DMA suspend acknowledge flag becomes active when all DMA channels that are enabled for DMA channel suspend have set their DMA channel suspend active status flag SUSACRc.SUSAC. In the suspend state, the DMA channel contents may be modified. These modifications are taken into account for further DMA transfers or DMA transactions of the related DMA channel after the suspend state has been left.

DMA channels that are disabled for suspend state (SUSENRc.SUSEN = 0) continue to execute the DMA channel operation.

Exiting DMA channel suspend state

The DMA channel suspend state is left and the DMA channel operation is resumed if either the Suspend REQ signal becomes inactive, or if the enable bit SUSENRc.SUSEN is reset by software.

9.5.2 Software activation of DMA channel interrupt service requests

Each DMA channel interrupt service request may be activated for testing purposes by programming the DMA channel CHCSRc.SIT = 1_B.

9.5.3 Software activation of DMA RP error interrupt service requests

Each DMA RP error interrupt service request may be activated for testing purposes by programming the RP ERRINTr.SIT = 1_B.

9.5.4 OCDS trigger bus (OTGB) interface

A Trigger Set (TS) is a collection of signals which supports a specific debug use case. The OCDS Trigger Set Select (OTSS) register controls which TS is applied to one of two DMA OTGB interfaces: OTGB0 or OTGB1. A TS is routed by the OCDS Trigger Multiplexer (OTGM) to the OCDS Trigger Switch (OTGS).

DMA trigger sets

The DMA trigger sets are selected and routed to the 16-bit OTGB0 or OTGB1 interface by programming the OTSS register. Only one TS shall be output at a time either on OTGB0 or on OTGB1. The OTGB0 and OTGB1 have no dependency on the OCDS enabled state.

***Note**: The OTGB interfaces may operate at a lower clock frequency than the signals being traced in the various DMA trigger sets. Certain higher frequency transitions on the traced signals may therefore not be visible on the OTGB interface.*

DMA TS are unique to each DMA configuration and dependent on the number of DMA channels and ME.

Table 650 DMA trigger sets

Index	Description	Details
0	No trigger set selected	
1	Channels (TS16_PF) Active channels	Table 651
2	Channels (TS16_ERR) Error channel number and flags	Table 652
8	Transaction request state (TS16_C15) Transaction request state channel [15:0]	
9	Transaction request state (TS16_C31) Transaction request state channel [31:16]	
10	Transaction request state (TS16_C47) Transaction request state channel [47:32]	
11	Transaction request state (TS16_C63) Transaction request state channel [63:48]	
12	Transaction request state (TS16_C79) Transaction request state channel [79:64]	
13	Transaction request state (TS16_C95) Transaction request state channel [95:80]	

(table continues...)

Table 650 (continued) DMA trigger sets

Index	Description	Details
14	Transaction request state (TS16_C111) Transaction request state channel [111:96]	
15	Transaction request state (TS16_C127) Transaction request state channel [127:112]	
Other	Reserved. No trigger set selected	

The information in the TS is suppressed depending on the value of the CSAV input signal. See chapter CSRM for details of the CSAV signal.

- When CSAV is "Full", no TS information is suppressed
- When CSAV is "Limited", no TS information is suppressed
- When CSAV is "No" and for channels belonging to a resource partition assigned as a CS asset, all information about the channel or ME executing the channel is suppressed from the TS

Performance trigger set

The performance TS (TS16_PF) continuously monitors ME activity. While a ME is servicing a DMA request, the ME activity status bit TS16_PF.MEm is set and the TS16_PF.CHm field is set to the number of the DMA channel. When the ME is idle the activity status bit TS16_PF.MEm goes low. The TS16_PF.CHm bit field retains the value of the last active DMA channel.

Table 651 TS16_PF trigger set channels

Bits	Name	Description
[6:0]	CH0	Number of DMA channel active in ME0
7	ME0	ME0 active 0_B ME0 is idle 1_B ME0 is active
[14:8]	CH1	Number of DMA channel active in ME1
15	ME1	ME1 active 0_B ME1 is idle 1_B ME1 is active

Error trigger set

The error TS (TS16_ERR) triggers an event for DMA source errors and DMA destination errors.

If an ME detects an SER or DER, the DMA sets TS16_ERR as follows:

- If MEm detects an SER, the DMA sets TS16_ERR.MEmSE for one trigger cycle
- If MEm detects a DER, the DMA sets TS16_ERR.MEmDE for one trigger cycle
- The DMA sets TS16_ERR.LEC to the last error channel for one trigger cycle. If both MEs generate an SER or a DER in the same clock cycle, the DMA shall set TS16_ERR.LEC to the ME0 last error channel

Table 652 TS16_ERR trigger set channels

Bits	Name	Description
[6:0]	LEC	Last error channel number
[11:7]		Reserved
12	ME0SE	ME0 source error

(table continues...)

Table 652 (continued) TS16_ERR trigger set channels

Bits	Name	Description
13	ME0DE	ME0 destination error
14	ME1SE	ME1 source error
15	ME1DE	ME1 destination error

Request trigger sets

The request trigger sets (TS16_C15, etc.) continuously monitor the state of the DMA channel TSRc.CH bits.

9.5.5 MCDS trace interface

The DMA implements trace ports for SRI master interface 0 and SRI master interface 1 (for instances which have SRI master interface 1) which send trace information to the MCDS.

The trace ports send transaction information to the trace data multiplexer (DMUX), which selects what information is then traced by the MCDS. The following information is sent to the DMUX:

- Address
- Data (either read or write data depending on the type of transaction)
- Read transaction ID
- Error
- Opcode
- Read
- Write
- Ready
- Write data valid
- Write transaction ID
- Supervisor mode
- Virtual machine number
- Protection register set number
- Virtual machine valid
- Protection register set valid
- Special master
- Sideband signals

Sideband signals are used in the DMA to identify the channel number and move engine used for the transactions seen by the FPI BCUs and the SRI XBARs as follows.

Table 653 DMA side band signal definition

Bits	Trace
[6:0]	DMA Channel
[7]	Move Engine 0_B ME0 1_B ME1

The information in the trace ports is suppressed depending on the value of the CSAV input signal. See chapter CSRM for details of the CSAV signal and its encoding.

- When CSAV is "Full", no trace port information is suppressed

- When CSAV is "Limited" and for channels belonging to a resource partition assigned as a CS asset, address and data information in the trace port are suppressed and set to 0
- When CSAV is "No" and for channels belonging to a resource partition assigned as a CS asset, all trace port information is suppressed

9.6 References

This section is not applicable for the module being described.

9.7 SDMA revision history

Initial release of the chapter.

9.8 TC4Dx SDMA information

9.8.1 TC4Dx SDMA configuration

The following table describes the TC4Dx SDMA configuration.

Table 654 TC4Dx SDMA configuration

Feature	DMA0	DMA1
Number of DMA channels	128	64
Number of DMA resource partitions	16	8
Number of DMA move engines	2	2
Number of FPI MIFs	2	2
Number of SRI MIFs	3	3

9.8.2 TC4Dx SDMA features

- Two SDMA instances - SDMA0 and SDMA1
  - Similar connectivity on both SDMA instances for maximum flexibility
- SDMA0 features 128 DMA channels and 16 DMA resource partitions
- SDMA1 features 64 DMA channels and 8 DMA resource partitions

9.8.3 TC4Dx SDMA functional description

9.8.3.1 TC4Dx DMA daisy chain request

When a DMA daisy chain request from a higher priority channel attempts to trigger the next lower priority channel and the lower priority channel is assigned to a different resource partition or no resource partition, the DMA should take the following actions:

- The lower priority DMA channel is not triggered
- The TSRc.TRL status bit is set
- The ERRSRr.DCTRLER status bit is set
- If TSRc.ETRL is 1_B in the higher priority channel, an RP error interrupt is signaled

On the AURIX™ TC4Dx device, the ERRSRr.DCTRLER status bit is not always set depending on which of the two MEs the higher priority DMA channel executes on. As a result, software should not rely on the ERRSRr.DCTRLER status bit and should instead infer the cause of the TRL error from the configuration of the DMA channel.

Related information

DMA daisy chain request on page 1260

9.8.3.2 TC4Dx DMA channel transfer interrupt service request

On the AURIX™ TC4Dx device, a DMA channel transfer interrupt service request may not be signaled from the DMA to the IR when all of the following conditions are true:

- The DMA channel is enabled to wait for write completion signaling (TSRc.WWC = 1_B)

- The DMA channel is configured to signal a DMA channel transfer interrupt service request at the end of every DMA transfer (ADICRc.INTCT = 11_B)
- The DMA channel receives an error from the interconnect for a DMA read move
- There are other DMA write requests pending on the interconnect with the same SRI master agent function identifier, even if these are from other DMA resource partitions

On an AURIX™ TC4Dx device, the DMA will merge the DMA channel transfer interrupt service request for the recently errored transfer with the DMA channel transfer interrupt service request for the next transfer and signal a single interrupt. The read error received by the DMA channel will cause the DMA to report a DMA RP error interrupt if configured to do so.

Related information

DMA channel transfer interrupt service request on page 1268

9.8.3.3 TC4Dx DMARAM integrity error interrupt service request

When a double-bit ECC error is detected during a DMA move engine access to the DMARAM to read the TCS on a AURIX™ TC4Dx device, the DMA transaction is not aborted under the following circumstances.

- The SPB clock frequency is configured to be the same as the SRI clock frequency AND
- The ECC error is preceded shortly by one of the following
  - A system sleep request
  - A DMA channel halt request
  - A DMA channel reset request
  - A suspend request from the debug system

The DMARAM ECC error is still reported to the SMU as an alarm, and software should select an appropriate safe reaction to this event.

Related information

DMARAM integrity error interrupt service request on page 1277

9.8.3.4 TC4Dx SDMA master interfaces

The following list describes the TC4Dx SDMA system interconnect master interfaces and the usage of each.

- SDMA0
  - FPI-MIF0: SPB address range
  - FPI-MIF1: ComPB address range
  - SRI-MIF0: All other addresses
  - SRI-MIF1
    - Addresses in system address map segments 2, 3 and 4
    - Cacheable and non-cacheable addresses in ranges CPU3.DLMU, CPU4.DLMU and CPU5.DLMU
    - Cacheable and non-cacheable addresses in ranges LMU2 and LMU3
    - Addresses in ranges CPUx.SFR, CPUx.HR0.CSFR, CPUx.HR1.CSFR and CPUx.HR2.CSFR (x=3, 4, 5); and SRI1.SFR
  - SRI-MIF2
    - Cacheable and non-cacheable addresses in ranges LMUx and LMUx.SFR (x=4, 5, 6, 7, 8, 9)

    - Addresses in ranges SRI6.SFR and SRI7.SFR
    - Addresses in ranges MCDS4P (incl. 32 Kbyte TBUF), MCDS2P (incl. 16 Kbyte TBUF), TRIF, CBS and TBCU
- SDMA1
  - FPI-MIF0: SPB address range
  - FPI-MIF1: ComPB address range
  - SRI-MIF0: All other addresses
  - SRI-MIF1
    - Addresses in system address map segments 5, 6 and 7
    - Cacheable and non-cacheable addresses in ranges CPU0.DLMU, CPU1.DLMU and CPU2.DLMU
    - Cacheable and non-cacheable addresses in ranges LMU0 and LMU1
    - Addresses in ranges CPUx.SFR, CPUx.HR0.CSFR, CPUx.HR1.CSFR and CPUx.HR2.CSFR (x=0, 1, 2); and SRI0.SFR
    - Addresses in range CSS
  - SRI-MIF2
    - Cacheable and non-cacheable addresses in ranges LMUx and LMUx.SFR (x=4, 5, 6, 7, 8, 9)
    - Addresses in ranges SRI6.SFR and SRI7.SFR
    - Addresses in ranges MCDS4P (incl. 32 Kbyte TBUF), MCDS2P (incl. 16 Kbyte TBUF), TRIF, CBS and TBCU

Related information

System address map on page 14

9.8.4 TC4Dx SDMA registers

9.8.4.1 Register address space - DMA

Table 655 Registers address space - DMA

Module	Base address	End address	Note
DMA0	$F4410000_H$	$F4413FFF_H$	FPI slave interface
DMA1	$F4414000_H$	$F4417FFF_H$	FPI slave interface

9.8.4.2 Register overview - access mode glossary

Table 656 Register overview - access mode glossary

Keyword	Description
CSE	Access protection using PROT register DMA0_PROTCSE or DMA1_PROTCSE .
RPr (r=0-7)	Access protection using PROT register DMA0_PROTRPr or DMA1_PROTRPr .
RPr (r=8-15)	Access protection using PROT register DMA0_PROTRPr .
SE	Access protection using PROT register DMA0_PROTSE or DMA1_PROTSE .
APU-Pr (r=0-7)	Protection group consisting of registers DMA0_ACCENRPrWRA , DMA0_ACCENRPrWRB , DMA0_ACCENRPrRDA , DMA0_ACCENRPrRDB , DMA0_ACCENRPrVM , DMA0_ACCENRPrPRS or DMA1_ACCENRPrWRA , DMA1_ACCENRPrWRB , DMA1_ACCENRPrRDA , DMA1_ACCENRPrRDB , DMA1_ACCENRPrVM , DMA1_ACCENRPrPRS .
Pr	Access protection using *APU-Pr* registers.
APU-Pr (r=8-15)	Protection group consisting of registers DMA0_ACCENRPrWRA , DMA0_ACCENRPrWRB , DMA0_ACCENRPrRDA , DMA0_ACCENRPrRDB , DMA0_ACCENRPrVM , DMA0_ACCENRPrPRS .
U	No access restrictions.
BE	Always returns a Bus Error.
PROT	Access restrictions as defined in the PROT register access rules.
32	Access only when using 32-bit width.
OEN	Access only when OCDS is enabled.
M	Indicates a module specific access condition. Refer to the register description for details of the specific access condition.

9.8.4.3 Register overview - DMA0 (ascending offset address)

Table 657 Register overview - DMA0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
DMA0_CLC	Clock control register	0000_H	U	SV, SE	Application Reset	1286
DMA0_OTSS	OCDS trigger set select	0004_H	U	SV, SE	Debug Reset	1287

(table continues...)

Table 657 (continued) Register overview - DMA0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
DMA0_ID	Module identification register	0008_H	U	BE	PowerOn Reset	1288
DMA0_PROTSE	PROT register safe endinit	0024_H	U	SV, PROT	Application Reset	1289
DMA0_PROTCSE	PROT register cyber-security endinit	0028_H	U	SV, PROT	Application Reset	1291
DMA0_TIME	DMA time register	0080_H	U	SV, SE, 32	System Reset	1293
DMA0_ERRSR	DMA error status register	0084_H	U	BE	Application Reset	1293
DMA0_CLRE	DMA clear error register	0088_H	U	SV, SE	Application Reset	1294
DMA0_TIMEOCS	DMA time OCDS control and status register	$008C_H$	U	SV, SE	Debug Reset	1295
DMA0_LCLTEST	DMA LCL test register	0090_H	U	SV, SE	Application Reset	1295
DMA0_MEmSR (m=0-1)	ME m status register	0410_H+ m*200_H	OEN	BE	Application Reset	1296
DMA0_MEmRDCRC (m=0-1)	ME m channel read data CRC register	0460_H+ m*200_H	OEN	BE	Application Reset	1297
DMA0_MEmSDCRC (m=0-1)	ME m channel source and destination address CRC register	0464_H+ m*200_H	OEN	BE	Application Reset	1297
DMA0_MEmSADR (m=0-1)	ME m channel source address register	0468_H+ m*200_H	OEN	BE	Application Reset	1297
DMA0_MEmDADR (m=0-1)	ME m channel destination address register	$046C_H$+ m*200_H	OEN	BE	Application Reset	1298
DMA0_MEmADICR (m=0-1)	ME m channel address and interrupt control register	0470_H+ m*200_H	OEN	BE	Application Reset	1298
DMA0_MEmCHCR (m=0-1)	ME m channel control register	0474_H+ m*200_H	OEN	BE	Application Reset	1300
DMA0_MEmSHADR (m=0-1)	ME m channel shadow address register	0478_H+ m*200_H	OEN	BE	Application Reset	1300
DMA0_MEmCHSR (m=0-1)	ME m channel status register	$047C_H$+ m*200_H	OEN	BE	Application Reset	1301

(table continues...)

Table 657 (continued) Register overview - DMA0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
DMA0_ACCENRPrWRA (r=0-15)	RP r write access enable register A	1000_H+r*80_H	U	SV, RPr	Application Reset	1302
DMA0_ACCENRPrWRB (r=0-15)	RP r write access enable register B	1004_H+r*80_H	U	SV, RPr	Application Reset	1302
DMA0_ACCENRPrRDA (r=0-15)	RP r read access enable register A	1008_H+r*80_H	U	SV, RPr	Application Reset	1303
DMA0_ACCENRPrRDB (r=0-15)	RP r read access enable register B	$100C_H$+r*80_H	U	SV, RPr	Application Reset	1303
DMA0_ACCENRPrVM (r=0-15)	RP r VM access enable register	1010_H+r*80_H	U	SV, RPr	Application Reset	1304
DMA0_ACCENRPrPRS (r=0-15)	RP r PRS access enable register	1014_H+r*80_H	U	SV, RPr	Application Reset	1305
DMA0_PROTRPr (r=0-15)	PROT register RP r	1020_H+r*80_H	U	SV, PROT	Application Reset	1305
DMA0_MODEr (r=0-15)	RP r mode register	1030_H+r*80_H	U	SV, M, 32	Application Reset	1307
DMA0_PRRAr (r=0-15)	RP r pattern read register 0	1038_H+r*80_H	Pr	SV, Pr	Application Reset	1308
DMA0_PRRBr (r=0-15)	RP r pattern read register 1	$103C_H$+r*80_H	Pr	SV, Pr	Application Reset	1309
DMA0_ERRINTRr (r=0-15)	RP r error interrupt set register	1040_H+r*80_H	Pr	SV, Pr	Application Reset	1309
DMA0_EERr (r=0-15)	RP r enable error register	1044_H+r*80_H	Pr	SV, Pr	Application Reset	1310
DMA0_ERRSRr (r=0-15)	RP r error status register	1048_H+r*80_H	Pr	BE	Application Reset	1310
DMA0_CLREr (r=0-15)	RP r clear error register	$104C_H$+r*80_H	Pr	SV, Pr	Application Reset	1313
DMA0_HRRc (c=000-127)	DMA channel c resource partition assignment register	1800_H+c*4	U	SV, M, 32	Application Reset	1314
DMA0_SUSENRc (c=000-127)	DMA channel c suspend enable register	$1A00_H$+c*4	U	SV, M	Debug Reset	1315

(table continues...)

Table 657 (continued) Register overview - DMA0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
DMA0_SUSACRc (c=000-127)	DMA channel c suspend acknowledge register	$1C00_H$+c *4	U	BE	Debug Reset	1316
DMA0_TSRc (c=000-127)	DMA channel c transaction state register	$1E00_H$+c *4	M	SV, M	Application Reset	1316
DMA0_RDCRCRc (c=000-127)	DMA channel c read data CRC register	2000_H+c *20_H	M	SV, M	Application Reset	1318
DMA0_SDCRCRc (c=000-127)	DMA channel c source and destination address CRC register	2004_H+c *20_H	M	SV, M	Application Reset	1319
DMA0_SADRc (c=000-127)	DMA channel c source address register	2008_H+c *20_H	M	SV, M	Application Reset	1319
DMA0_DADRc (c=000-127)	DMA channel c destination address register	$200C_H$+c *20_H	M	SV, M	Application Reset	1320
DMA0_ADICRc (c=000-127)	DMA channel c address and interrupt control register	2010_H+c *20_H	M	SV, M	Application Reset	1320
DMA0_CHCFGRc (c=000-127)	DMA channel c configuration register	2014_H+c *20_H	M	SV, M	Application Reset	1324
DMA0_SHADRc (c=000-127)	DMA channel c shadow address register	2018_H+c *20_H	M	SV, M	Application Reset	1326
DMA0_CHCSRc (c=000-127)	DMA channel c control and status register	$201C_H$+c *20_H	M	SV, M	Application Reset	1327

9.8.4.4 Register overview - DMA1 (ascending offset address)

Table 658 Register overview - DMA1 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
DMA1_CLC	Clock control register	0000_H	U	SV, SE	Application Reset	1286
DMA1_OTSS	OCDS trigger set select	0004_H	U	SV, SE	Debug Reset	1287
DMA1_ID	Module identification register	0008_H	U	BE	PowerOn Reset	1288
DMA1_PROTSE	PROT register safe endinit	0024_H	U	SV, PROT	Application Reset	1289
DMA1_PROTCSE	PROT register cyber-security endinit	0028_H	U	SV, PROT	Application Reset	1291
DMA1_TIME	DMA time register	0080_H	U	SV, SE, 32	System Reset	1293
DMA1_ERRSR	DMA error status register	0084_H	U	BE	Application Reset	1293

(table continues...)

Table 658 (continued) Register overview - DMA1 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
DMA1_CLRE	DMA clear error register	0088_H	U	SV, SE	Application Reset	1294
DMA1_TIMEOCS	DMA time OCDS control and status register	$008C_H$	U	SV, SE	Debug Reset	1295
DMA1_LCLTEST	DMA LCL test register	0090_H	U	SV, SE	Application Reset	1295
DMA1_MEmSR (m=0-1)	ME m status register	0410_H+ m*200_H	OEN	BE	Application Reset	1296
DMA1_MEmRDCRC (m=0-1)	ME m channel read data CRC register	0460_H+ m*200_H	OEN	BE	Application Reset	1297
DMA1_MEmSDCRC (m=0-1)	ME m channel source and destination address CRC register	0464_H+ m*200_H	OEN	BE	Application Reset	1297
DMA1_MEmSADR (m=0-1)	ME m channel source address register	0468_H+ m*200_H	OEN	BE	Application Reset	1297
DMA1_MEmDADR (m=0-1)	ME m channel destination address register	$046C_H$+ m*200_H	OEN	BE	Application Reset	1298
DMA1_MEmADICR (m=0-1)	ME m channel address and interrupt control register	0470_H+ m*200_H	OEN	BE	Application Reset	1298
DMA1_MEmCHCR (m=0-1)	ME m channel control register	0474_H+ m*200_H	OEN	BE	Application Reset	1300
DMA1_MEmSHADR (m=0-1)	ME m channel shadow address register	0478_H+ m*200_H	OEN	BE	Application Reset	1300
DMA1_MEmCHSR (m=0-1)	ME m channel status register	$047C_H$+ m*200_H	OEN	BE	Application Reset	1301
DMA1_ACCENRPrWRA (r=0-7)	RP r write access enable register A	1000_H+r* 80_H	U	SV, RPr	Application Reset	1302
DMA1_ACCENRPrWRB (r=0-7)	RP r write access enable register B	1004_H+r* 80_H	U	SV, RPr	Application Reset	1302
DMA1_ACCENRPrRDA (r=0-7)	RP r read access enable register A	1008_H+r* 80_H	U	SV, RPr	Application Reset	1303

(table continues...)

Table 658 (continued) Register overview - DMA1 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
DMA1_ACCENRPrRDB (r=0-7)	RP r read access enable register B	$100C_H$+r*80_H	U	SV, RPr	Application Reset	1303
DMA1_ACCENRPrVM (r=0-7)	RP r VM access enable register	1010_H+r*80_H	U	SV, RPr	Application Reset	1304
DMA1_ACCENRPrPRS (r=0-7)	RP r PRS access enable register	1014_H+r*80_H	U	SV, RPr	Application Reset	1305
DMA1_PROTRPr (r=0-7)	PROT register RP r	1020_H+r*80_H	U	SV, PROT	Application Reset	1305
DMA1_MODEr (r=0-7)	RP r mode register	1030_H+r*80_H	U	SV, M, 32	Application Reset	1307
DMA1_PRRAr (r=0-7)	RP r pattern read register 0	1038_H+r*80_H	Pr	SV, Pr	Application Reset	1308
DMA1_PRRBr (r=0-7)	RP r pattern read register 1	$103C_H$+r*80_H	Pr	SV, Pr	Application Reset	1309
DMA1_ERRINTRr (r=0-7)	RP r error interrupt set register	1040_H+r*80_H	Pr	SV, Pr	Application Reset	1309
DMA1_EERr (r=0-7)	RP r enable error register	1044_H+r*80_H	Pr	SV, Pr	Application Reset	1310
DMA1_ERRSRr (r=0-7)	RP r error status register	1048_H+r*80_H	Pr	BE	Application Reset	1310
DMA1_CLREr (r=0-7)	RP r clear error register	$104C_H$+r*80_H	Pr	SV, Pr	Application Reset	1313
DMA1_HRRc (c=000-063)	DMA channel c resource partition assignment register	1800_H+c*4	U	SV, M, 32	Application Reset	1314
DMA1_SUSENRc (c=000-063)	DMA channel c suspend enable register	$1A00_H$+c*4	U	SV, M	Debug Reset	1315
DMA1_SUSACRc (c=000-063)	DMA channel c suspend acknowledge register	$1C00_H$+c*4	U	BE	Debug Reset	1316
DMA1_TSRc (c=000-063)	DMA channel c transaction state register	$1E00_H$+c*4	M	SV, M	Application Reset	1316
DMA1_RDCRCRc (c=000-063)	DMA channel c read data CRC register	2000_H+c*20_H	M	SV, M	Application Reset	1318
DMA1_SDCRCRc (c=000-063)	DMA channel c source and destination address CRC register	2004_H+c*20_H	M	SV, M	Application Reset	1319

(table continues...)

Table 658 (continued) Register overview - DMA1 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
DMA1_SADRc (c=000-063)	DMA channel c source address register	2008_H+c *20_H	M	SV, M	Application Reset	1319
DMA1_DADRc (c=000-063)	DMA channel c destination address register	$200C_H$+c *20_H	M	SV, M	Application Reset	1320
DMA1_ADICRc (c=000-063)	DMA channel c address and interrupt control register	2010_H+c *20_H	M	SV, M	Application Reset	1320
DMA1_CHCFGRc (c=000-063)	DMA channel c configuration register	2014_H+c *20_H	M	SV, M	Application Reset	1324
DMA1_SHADRc (c=000-063)	DMA channel c shadow address register	2018_H+c *20_H	M	SV, M	Application Reset	1326
DMA1_CHCSRc (c=000-063)	DMA channel c control and status register	$201C_H$+c *20_H	M	SV, M	Application Reset	1327

9.8.4.5 Device specific registers

There are no device specific register changes.

9.8.5 TC4Dx SDMA connectivity

The table below lists interfaces to and from this functional block to other blocks in the device.

Table 659 List of SDMA interface signals

Interface signals	I/O	Description
CLOCK_DMA_fSRI	In	fSRI
CLOCK_DMA_fSPB	In	fSPB
CLOCK_DMA_fCPB	In	fCPB (for devices with a CPB bus)
CLOCK_DMA_fSTM	In	fSTM
SMM_DMA_ApplicationReset	In	Application Reset
SMM_DMA_SystemReset	In	System Reset
SMM_DMA_DebugReset	In	Debug Reset
DEBUG_DMA_SuspendREQ/ACK	In	DMA OCDS suspend request/ack
CSRM.CSCU_DMA_CSAV[1:0]	In	CSAV
DMA0_DEBUG_OTGB0[15:0]	Out	DMA0 OTGB Interface
DMA1_DEBUG_OTGB0[15:0]	Out	DMA1 OTGB Interface
DMA0_DEBUG_OTGB1[15:0]	Out	DMA0 OTGB Interface
DMA1_DEBUG_OTGB1[15:0]	Out	DMA1 OTGB Interface
DMA0_TRACE_TRACE_SRIx	Out	DMA0 trace interface(s). 'x' is the index of each SRI master interface on the DMA0 instance
DMA1_TRACE_TRACE_SRIx	Out	DMA1 trace interface(s). 'x' is the index of each SRI master interface on the DMA1 instance
DMA0_SMU_ALARM[1:0]	Out	DMA0 alarms
DMA1_SMU_ALARM[1:0]	Out	DMA1 alarms
IR_DMA0_ICU7	In	DMA0 Hardware Request
IR_DMA1_ICU8	In	DMA1 Hardware Request
DMA0_IR_CHc	Out	DMA0 Channel Service Request. 'c' is the index of each DMA0 channel on the DMA0 instance
DMA1_IR_CHc	Out	DMA1 Channel Service Request. 'c' is the index of each DMA1 channel on the DMA1 instance
DMA0_IR_ERRr	Out	DMA0 RP Error Service Request. 'r' is the index of each DMA0 resource partition on the DMA0 instance
DMA1_IR_ERRr	Out	DMA1 RP Error Service Request. 'r' is the index of each DMA1 resource partition on the DMA1 instance
DMA0_IR_GERR	Out	DMA0 Global Error Service Request
DMA1_IR_GERR	Out	DMA1 Global Error Service Request

(table continues...)

Table 659 (continued) List of SDMA interface signals

Interface signals	I/O	Description
FPI_DMA0_FPI_SIF	In	DMA0 Configuration Interface
FPI_DMA1_FPI_SIF	In	DMA1 Configuration Interface
DMA0_FPI_FPI_MIFx	Out	DMA0 FPI Master Interface(s). 'x' is the index of each FPI master interface on the DMA0 instance
DMA1_FPI_FPI_MIFx	Out	DMA1 FPI Master Interface(s). 'x' is the index of each FPI master interface on the DMA1 instance
DMA0_SRI_SRI_MIFx	Out	DMA0 SRI Master Interface(s). 'x' is the index of each SRI master interface on the DMA0 instance
DMA1_SRI_SRI_MIFx	Out	DMA1 SRI Master Interface(s). 'x' is the index of each SRI master interface on the DMA1 instance

9.8.6 TC4Dx SDMA revision history

Initial release of the chapter.

10 Local Memory Unit (LMU)

The local memory unit is a resource attached to the System Resource Interconnect (SRI) that provides SRAM for general purpose use. A product may contain multiple instances of the LMU, and each instance has its own set of configuration registers.

Data stored in the SRAM is protected by error-correcting codes (ECC). Master access to areas of each LMU can be permitted or denied by configuring up to 16 address ranges using an Access Protection Unit (APU) in the LMU. The ranges can be sized in 64-byte increments and the master agent functions can be granted access for read and write independently. A master agent function is identified by its number (ID), Protection Register Set (PRS) and Virtual Machine (VM) identifier.

***Note**: Please refer to the system address map for the availability of LMU instances in a specific product.*

10.1 Feature list

- Each LMU instance has 512 KByte of SRAM
- Provides byte, half-word, word, and double-word accesses, as well as double-word burst transactions of length 2,4, or 8 to SRAM
- Provides word and double-word access to most configuration registers
- Each LMU has sixteen Access Protection Units (APUs) for memory access protection

10.2 Functional overview

The LMU SRAM can be used for code execution, data storage or overlay memory. The address range of the memory for each LMU instance is defined in the system memory map. The memory supports being aliased to more than one address range, for example, cached and non-cached segments. A block diagram of the LMU is shown below.

Access to the SRAM is protected by 16 Access Protection Units (APUs), each of which can be configured to limit access by masters to a configurable memory range. By default (after reset), read access is allowed by any master to the whole memory and write access is to certain masters, details of which can be found inthe reset value of RGNx_ACCEN_WRA. The safety configuration and monitoring registers are protected by PROTSE.

The memory implements memory integrity checking for error detection and correction. This means that the memory must be initialized before reading with the integrity checking enabled to avoid generating spurious data corruption errors. Initializing with the memory integrity logic disabled allows the LMU SRAM to support initialization using word (32-bit) or smaller writes. Initialization using 64-bit writes is always possible.

If memory integrity checking is enabled, a read access which fails the integrity check will cause an error condition to be flagged to the initiating bus master. This behavior can be changed by setting the MEMCON.ERRDIS bit to 1_B. If ERRDIS is set, the access will terminate without an integrity error being reported to the SRI. The ERRDIS bit-field of the MEMCON register is protected by the MEMCON.ERRDISWE bit. If the data written to the register has the MEMCON.ERRDISWE bit set to 0_B, no change will be made to ERRDIS regardless of the data written to the field.

An alarm will be triggered whenever there is an ECC error, regardless of the value of MEMCON.ERRDIS.

The SRAM can be written as a 64-bit memory without using internal Read-Modify-Write (iRMW) operations. Any write access of less than 64 bits of data results in an iRMW operation to correctly write the data and update the ECC data. This happens transparently to the rest of the system unless an uncorrected ECC error is detected during the read phase of the iRMW. Any such ECC error will be flagged with an alarm in the same way as a data ECC error occurring during a normal read. In addition, the MEMCON.RMWERR flag will be set. The write operation will not take place as this would write potentially incorrect data to the memory.

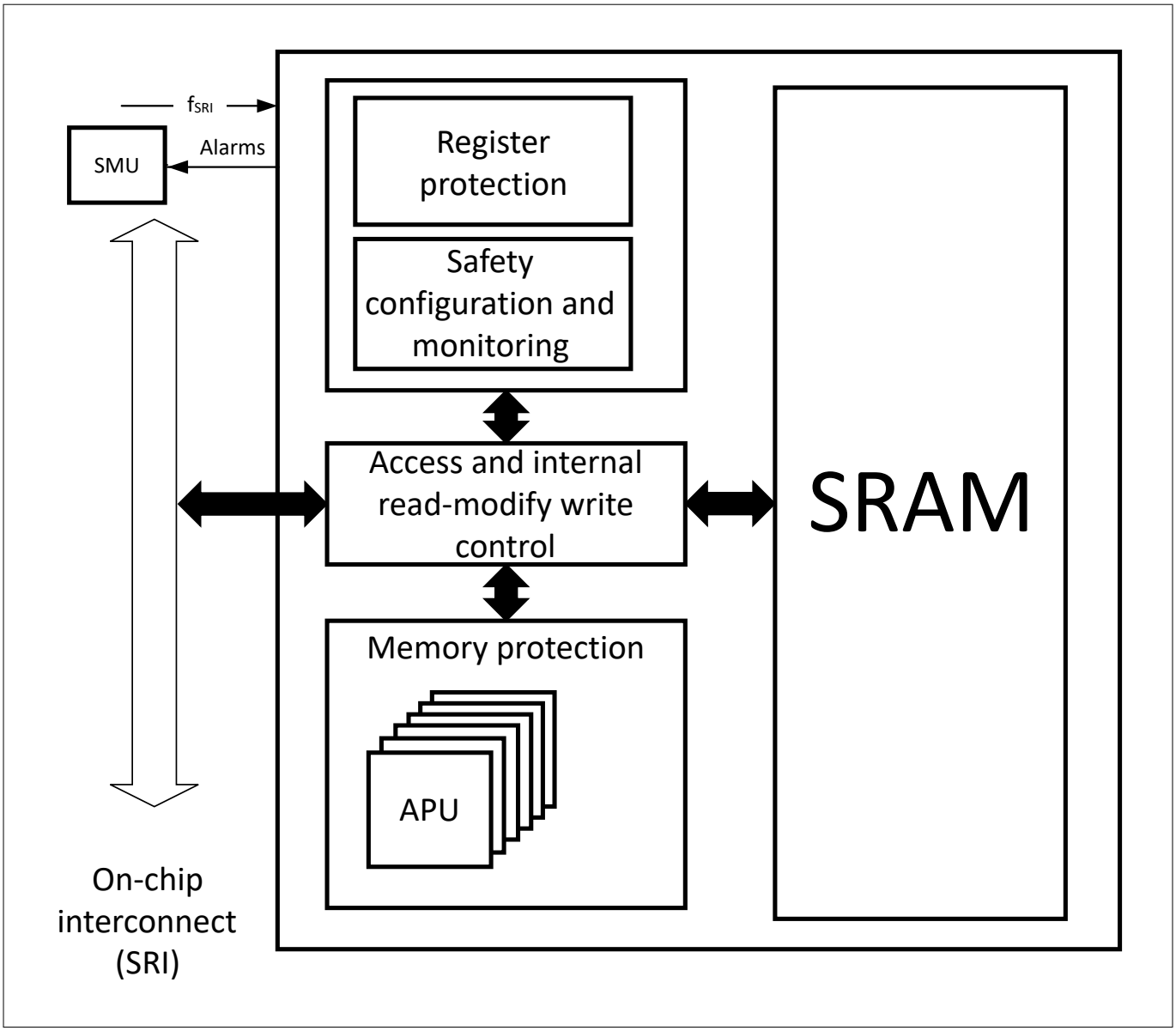


Figure 128 LMU block diagram

For the alarm description please refer to the alarm mapping tables in the SMU functional block and the Safety Manual.

Related information

TC4Dx SMU alarm mapping tables on page 7233

10.3 Functional description

The following sections describe the functions of the LMU.

10.3.1 Memory protection

The LMU allows for the definition of 16, 64-byte aligned, protected regions of SRAM memory defined by an array of access protection unit (APU) instances. This protection applies only to accesses to SRAM included in the LMU, not to registers.

The configuration of the protection regions is held in the set of registers named with the prefix RGNx_, where x is the number of the APU instance. After reset, the protection region range registers will be set to include the whole of the LMU SRAM address space, and read access by all masters will be enabled but write access is to CPU0 and Cerberus. If overlapping regions are defined, then an access only needs to be permittedby one of the overlapping regions for it to succeed.

An APU can only be configured by writing to the equivalent shadow register in the set named with the prefix ACCENCFG_. The shadow register set is protected by a mechanism controlled by the PROTRGN register. The SEL field in this register selects which region is to be configured and write access to the shadow registers is only given when the PROTRGN.STATE field is in an unlocked state. For full details of how the protection mechanism operates, see the access protection section of the manual.

If the access protection is violated by a read access, the LMU will trigger an APU alarm and cause an SRI error on the data phase. If the access protection is violated by a write access, the LMU will trigger an APU alarm and inhibit the write to the SRAM.

In the case of an SRI RMW access, the APU checks for both the read and the write part of the access are done in the address phase of the read part. If either the read or write part would violate the access protection then the LMU will trigger an APU alarm and cause an SRI error on the read data phase.

Related information

PROT mechanism on page 64

Access protection unit (APU) on page 76

10.3.2 Register protection

The registers that control and monitor safety mechanisms are protected by the PROTSE register. Write access to these registers is only given when the PROTSE.STATE field is in an unlocked state.

Any access to an address in the configuration register space of the LMU that is not to a defined register will cause an SRI error which will trigger an alarm in the SRI. Any access to an SFR that does not comply with the access mode for that SFR will cause an SRI error, which will trigger an alarm in the SRI. Any access to the PROTSE or PROTRGN registers that violates the PROT register access rules will trigger an APU alarm.

10.3.3 Clock control

The LMU contains power saving logic that only clocks the SRAM and other functions when an access is being performed.

10.4 Registers

The registers in the LMU are accessible using 32-bit-and 64-bit SRI read and write transactions, except where documented as 32-bit only in the register overview table.

10.4.1 Register overview - access mode glossary

Table 660 Register overview - access mode glossary

Keyword	Description
RGN	Access protection using PROT register PROTRGN.
SE	Access protection using PROT register PROTSE.
32	Access only when using 32-bit width.
BE	Always returns a Bus Error.
PROT	Access restrictions as defined in the PROT register access rules.
U	No access restrictions.

10.4.2 Registers overview - LMU (ascending offset address)

Table 661 Registers overview - LMU (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ID	Module identification register	00008_H	32	BE	PowerOn Reset	1352
PROTSE	Protection for safety registers	00028_H	32	SV, 32, PROT	Application Reset	1353
MEMCON	Memory control register	00060_H	32	SV, SE, 32	Application Reset	1354
SCTRL	Safety control register	00064_H	32	SV, SE, 32	Application Reset	1356
PROTRGN	Protection for memory regions	00070_H	32	SV, 32, PROT	Application Reset	1357
RGNx_ACCEN_WRA	Region write access enable register A x	00080_H+ x*20_H	U	BE	Application Reset	1359
RGNx_ACCEN_WRB	Region write access enable register B x	00084_H+ x*20_H	U	BE	Application Reset	1359
RGNx_ACCEN_RDA	Region read access enable register A x	00088_H+ x*20_H	U	BE	Application Reset	1360
RGNx_ACCEN_RDB	Region read access enable register B x	$0008C_H$+ x*20_H	U	BE	Application Reset	1360

(table continues...)

Table 661 (continued) Registers overview - LMU (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
RGNx_ACCEN_VM	Region VM access enable register x	00090_H+ x*20_H	U	BE	Application Reset	1361
RGNx_ACCEN_PRS	Region PRS access enable register x	00094_H+ x*20_H	U	BE	Application Reset	1361
RGNx_ACCEN_RGNLA	Region lower address register x	00098_H+ x*20_H	U	BE	Application Reset	1362
RGNx_ACCEN_RGNUA	Region upper address register x	$0009C_H$+ x*20_H	U	BE	Application Reset	1363
ACCENCFG_WRA	Configure region write access enable register A	00300_H	U	RGN, SV	Application Reset	1363
ACCENCFG_WRB	Configure region write access enable register B	00304_H	U	RGN, SV	Application Reset	1364
ACCENCFG_RDA	Configure region read access enable register A	00308_H	U	RGN, SV	Application Reset	1364
ACCENCFG_RDB	Configure region read access enable register B	$0030C_H$	U	RGN, SV	Application Reset	1365
ACCENCFG_VM	Configure region VM access enable register	00310_H	U	RGN, SV	Application Reset	1365
ACCENCFG_PRS	Configure region PRS access enable register	00314_H	U	RGN, SV	Application Reset	1366
ACCENCFG_RGNLA	Configure region lower address register	00318_H	U	RGN, SV	Application Reset	1367
ACCENCFG_RGNUA	Configure region upper address register	$0031C_H$	U	RGN, SV	Application Reset	1368

10.4.3 Module identification register

ID Offset address: 00008_H

Module identification register PowerOn Reset value: $0088\ C005_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MOD_NUM															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MOD_TYPE								MOD_REV							
r								r							

Field	Bits	Type	Description
MOD_REV	7:0	r	**Module Revision** Indicates the revision number of the implementation. This information depends on the design step.

(table continues...)

(continued)

Field	Bits	Type	Description
MOD_TYPE	15:8	r	**Module Type** This internal marker is fixed to $C0_H$.
MOD_NUM	31:16	r	**Module Number** Indicates the module identification number (0088_H = LMU).

10.4.4 Protection for safety registers

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner and in some PROT states, the secure master to update the PROT state as described in the PROT mechanism.

PROTSE Offset address: 00028_H

Protection for safety registers Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OWEN	ODEF	TAGID						PRSEN	PRS			VMEN	VM		
w	rw	rw						rw	rw			rw	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												SWEN	STATE		
r												w	rwh		

Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
VM	18:16	rw	**Virtual machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.

(table continues...)

(continued)

Field	Bits	Type	Description
VMEN	19	rw	**Virtual machine definition enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.
PRSEN	23	rw	**Protection set definition enable for PROT owner** 0_B **Disable**: PRS is not part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG-ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; should be written with 0.

Table 662 Access mode restrictions of PROTSE sorted by descending priority

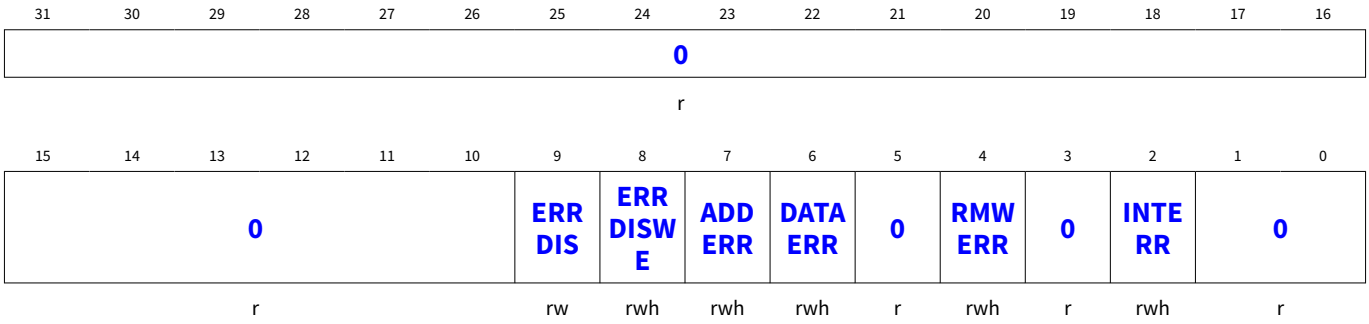
Mode name	Access mode		Description
write 1 to .**SWEN**	rwh	STATE	
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

10.4.5 Memory control register

This register allows software to monitor both the memory integrity error checking and the error signalling to the SMU.

MEMCON Offset address: 00060_H

Memory control register Application Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
INTERR	2	rwh	**Internal ECC error** Flag set by hardware when the LMU detects an ECC error on the internal data path registers while accessing the SRAM. This bit is cleared by writing 0_B but cannot be set by software. 0_B No error has occurred. 1_B An error has been observed during a RAM access.
RMWERR	4	rwh	**Internal read modify write error** Flag set by hardware when an uncorrected ECC error is reported by the SRAM on the read phase of an internal RMW operation. This bit is cleared by writing 0_B but cannot be set by software. 0_B No error has occurred. 1_B An error has been observed during an internal RMW operation.
DATAERR	6	rwh	**SRI write data phase ECC error** Flag set by hardware when the SRI interface detects an ECC error in the data phase of an incoming write transaction. This bit is cleared by writing 0_B but cannot be set by software. 0_B No error has occurred. 1_B An ECC error has been observed on an SRI transaction addressed to the LMU.
ADDERR	7	rwh	**SRI address phase ECC error** Flag set by hardware when the SRI interface detects an ECC error in the address phase of an incoming transaction. This bit is cleared by writing 0_B but cannot be set by software. 0_B No error has occurred. 1_B An ECC address error has been observed on an SRI transaction addressed to the LMU.
ERRDISWE	8	rwh	**ERRDIS write enable** Will always return 0_B when read. 0_B Bit Protected: ERRDIS bit remains unchanged after MEMCON write. 1_B ERRDIS bit will be updated by the current write to MEMCON.

(table continues...)

(continued)

Field	Bits	Type	Description
ERRDIS	9	rw	**ECC error disable** When set to 1_B, SRI error reporting of ECC errors in data read from the SRAM will be disabled, and an ECC error on the read phase of an internal RMW will not cause the write phase to be aborted. The value of this bit does not affect the generation of alarms. 0_B Normal behavior. SRI read errors will occur for read transactions with SRAM ECC errors. Default after reset. 1_B Test or initialisation mode. SRI read errors will not be generated on SRAM ECC errors.
0	1:0, 3, 5, 31:10	r	**Reserved** Read as 0; should be written with 0.

Table 663 Access mode restrictions of MEMCON sorted by descending priority

Mode name	Access mode		Description
write 1 to .**ERRDISWE**	rw	ERRDIS	ERDISS will only be written if ERRDISWE is written with a 1 at the same time
(default)	r	ERRDIS	Field can always be read

10.4.6 Safety control register

This register provides control of the user-configurable safety mechanisms of the LMU.

SCTRL Offset address: 00064_H

Safety control register Application Reset value: $0000\ 0400_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				CMPTST		0									GED
r				rw		r									w

Field	Bits	Type	Description
GED	0	w	**Generate error in EDC for data protection** The data paths between the SRAM and the LMU SRI interface are protected by error detection code (EDC) logic. This bit is used to inject an error into the EDC code on the next write access so that the EDC check and SMU alarm (internal data transfer EDC error) can be tested. The next write will not be inhibited as a result of this error injection. Reading this bit always returns 0_B. Writing works as follows: 0_B No effect 1_B Inject an error into the LMU data path EDC on the next write access to the LMU.

(table continues...)

(continued)

Field	Bits	Type	Description
CMPTST	11:10	rw	**Comparator test** Setting this bit-field to ON will inject an error in to the comparators checking the duplicated logic area. This will allow the correct operation of the control redundancy mismatch alarm to be verified. An error will continue to be injected until this field is set to OFF or the module is reset. Writing either of the invalid values (00_b or 11_b) to this redundantly encoded control field will trigger the control redundancy self-test mismatch alarm allowing it to be tested. 00_B **RES0**: Invalid 01_B **OFF**: No error injected 10_B **ON**: Error injected 11_B **RES3**: Invalid
0	9:1, 31:12	r	**Reserved** Read as 0; should be written with 0.

10.4.7 Protection for memory regions

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner and in some PROT states, the secure master to update the PROT state as described in the PROT mechanism.

The SEL bit-field is an index into the APU array selecting the APU to be configured.

PROTRGN Offset address: 00070_H

Protection for memory regions Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OWEN	ODEF	TAGID						PRSEN	PRS			VMEN	VM		
w	rw	rw						rw	rw			rw	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				SEL				0				SWEN	STATE		
r				rw				r				w	rwh		

Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. 0_B **Disable**: STATE field will not be modified 1_B **Enable**: STATE field may be modified by the PROT owner
SEL	11:8	rw	**Protection Region Select** Selects which set of region protection registers shall be written from the shadow set.
VM	18:16	rw	**Virtual Machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual Machine definition Enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection Set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.
PRSEN	23	rw	**Protection Set definition Enable for PROT owner** 0_B **Disable**: PRS is not part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner
OWEN	31	w	**Owner write enable** 0_B **Disable**: PROT owner fields are not modified 1_B **Enable**: PROT owner fields may be modified by the PROT owner
0	7:4, 15:12	r	**Reserved** Read as 0; should be written with 0.

Table 664 Access mode restrictions of PROTRGN sorted by descending priority

Mode name	Access mode		Description
write 1 to .**SWEN**	rwh	STATE	
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

10.4.8 Region write access enable register A x

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

RGNx_ACCEN_WRA (x=0-15) Offset address: 00080_H+x*20_H

Region write access enable register A x Application Reset value: $1000\ 0003_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	**EN30**	**EN29**	**EN28**	**EN27**	**EN26**	**EN25**	**EN24**	**EN23**	**EN22**	**EN21**	**EN20**	**EN19**	**EN18**	**EN17**	**EN16**
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	**EN14**	**EN13**	**EN12**	**EN11**	**EN10**	**EN09**	**EN08**	**EN07**	**EN06**	**EN05**	**EN04**	**EN03**	**EN02**	**EN01**	**EN00**
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
ENq (q=00-31)	q	rh	**Write access enable for Master TAG ID q** This bit enables write access to the access protected region for transactions with the Master TAG ID q. 0_B Disabled for write access 1_B Enabled for write access

10.4.9 Region write access enable register B x

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

RGNx_ACCEN_WRB (x=0-15) Offset address: 00084_H+x*20_H

Region write access enable register B x Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN63	**EN62**	**EN61**	**EN60**	**EN59**	**EN58**	**EN57**	**EN56**	**EN55**	**EN54**	**EN53**	**EN52**	**EN51**	**EN50**	**EN49**	**EN48**
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN47	**EN46**	**EN45**	**EN44**	**EN43**	**EN42**	**EN41**	**EN40**	**EN39**	**EN38**	**EN37**	**EN36**	**EN35**	**EN34**	**EN33**	**EN32**
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
ENq (q=32-63)	q-32	rh	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access

10.4.10 Region read access enable register A x

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

RGNx_ACCEN_RDA (x=0-15) Offset address: 00088_H+x*20_H

Region read access enable register A x Application Reset value: FFFF FFFF$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
ENq (q=00-31)	q	rh	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access

10.4.11 Region read access enable register B x

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

RGNx_ACCEN_RDB (x=0-15) Offset address: $0008C_H+x*20_H$

Region read access enable register B x Application Reset value: FFFF FFFF$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN63	EN62	EN61	EN60	EN59	EN58	EN57	EN56	EN55	EN54	EN53	EN52	EN51	EN50	EN49	EN48
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN47	EN46	EN45	EN44	EN43	EN42	EN41	EN40	EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
ENq (q=32-63)	q-32	rh	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access

10.4.12 Region VM access enable register x

This register defines which virtual machine encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each VMx.

RGNx_ACCEN_VM (x=0-15) Offset address: 00090_H+x*20_H

Region VM access enable register x Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
RDq (q=00-07)	q	rh	**Read access enable for VM ID q** This bit enables read access to the access protected region for transactions with the VM ID q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rh	**Write access enable for VM ID q** This bit enables write access to the access protected region for transactions with the VM ID q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

10.4.13 Region PRS access enable register x

This register defines which protection register set encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each PRSx.

RGNx_ACCEN_PRS (x=0-15) Offset address: 00094_H+x*20_H

Region PRS access enable register x Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
RDq (q=00-07)	q	rh	**Read access enable for PRS q** This bit enables read access to the access protected region for transactions with the PRS q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rh	**Write access enable for PRS q** This bit enables write access to the access protected region for transactions with the PRS q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

10.4.14 Region lower address register x

This register sets the lower bound of the access protected region. Addresses greater than or equal to the lower bound are considered part of the access protected region.

RGNx_ACCEN_RGNLA (x=0-15) Offset address: 00098_H+x*20_H

Region lower address register x Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADDRH		0	ADDR												
rh		r	rh												

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR										0					
rh										r					

Field	Bits	Type	Description
ADDR	28:6	rh	**Defines bits 28:6 of the lower bound of the access protected region**
ADDRH	31:30	rh	**Defines bits 31:30 of the lower bound of the access protected region**
0	5:0, 29	r	**Reserved** Read as 0; should be written with 0.

10.4.15 Region upper address register x

This register sets the upper bound of the access protected region. Addresses stricly less than the upper bound are considered part of the access protected region.

RGNx_ACCEN_RGNUA (x=0-15) Offset address: $0009C_H+x*20_H$

Region upper address register x Application Reset value: $DFFF\ FFC0_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADDRH		0	ADDR												
rh		r	rh												

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR										0					
rh										r					

Field	Bits	Type	Description
ADDR	28:6	rh	**Defines bits 28:6 of the upper bound of the access protected region**
ADDRH	31:30	rh	**Defines bits 31:30 of the upper bound of the access protected region**
0	5:0, 29	r	**Reserved** Read as 0; should be written with 0.

10.4.16 Configure region write access enable register A

The shadow region access enable registers allow atomic update of one region configuration under the control of the PROTRGN state machine. The value of the PROTRGN.SEL register bit-field determines which APU is configured.

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

ACCENCFG_WRA Offset address: 00300_H

Configure region write access enable register A Application Reset value: $1000\ 0003_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access

10.4.17 Configure region write access enable register B

The shadow region access enable registers allow atomic update of one region configuration under the control of the PROTRGN state machine. The value of the PROTRGN.SEL register bit-field determines which APU is configured.

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

ACCENCFG_WRB Offset address: 00304_H

Configure region write access enable register B Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN63	EN62	EN61	EN60	EN59	EN58	EN57	EN56	EN55	EN54	EN53	EN52	EN51	EN50	EN49	EN48
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN47	EN46	EN45	EN44	EN43	EN42	EN41	EN40	EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-63)	q-32	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access

10.4.18 Configure region read access enable register A

The shadow region access enable registers allow atomic update of one region configuration under the control of the PROTRGN state machine. The value of the PROTRGN.SEL register bit-field determines which APU is configured.

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

ACCENCFG_RDA Offset address: 00308_H

Configure region read access enable register A Application Reset value: $FFFF\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access

10.4.19 Configure region read access enable register B

The shadow region access enable registers allow atomic update of one region configuration under the control of the PROTRGN state machine. The value of the PROTRGN.SEL register bit-field determines which APU is configured.

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

ACCENCFG_RDB Offset address: $0030C_H$

Configure region read access enable register B Application Reset value: $FFFF\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN63	EN62	EN61	EN60	EN59	EN58	EN57	EN56	EN55	EN54	EN53	EN52	EN51	EN50	EN49	EN48
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN47	EN46	EN45	EN44	EN43	EN42	EN41	EN40	EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-63)	q-32	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access

10.4.20 Configure region VM access enable register

The shadow region access enable registers allow atomic update of one region configuration under the control of the PROTRGN state machine. The value of the PROTRGN.SEL register bit-field determines which APU is configured.

This register defines which virtual machine encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each VMx.

ACCENCFG_VM Offset address: 00310_H

Configure region VM access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for VM ID q** This bit enables read access to the access protected region for transactions with the VM ID q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for VM ID q** This bit enables write access to the access protected region for transactions with the VM ID q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

10.4.21 Configure region PRS access enable register

The shadow region access enable registers allow atomic update of one region configuration under the control of the PROTRGN state machine. The value of the PROTRGN.SEL register bit-field determines which APU is configured.

This register defines which protection register set encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each PRSx.

ACCENCFG_PRS Offset address: 00314_H

Configure region PRS access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for PRS q** This bit enables read access to the access protected region for transactions with the PRS q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for PRS q** This bit enables write access to the access protected region for transactions with the PRS q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

10.4.22 Configure region lower address register

The shadow region access enable registers allow atomic update of one region configuration under the control of the PROTRGN state machine. The value of the PROTRGN.SEL register bit-field determines which APU is configured.

This register sets the lower bound of the access protected region. Addresses greater than or equal to the lower bound are considered part of the access protected region.

ACCENCFG_RGNLA Offset address: 00318_H

Configure region lower address register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADDRH		0	ADDR												
rw		r	rw												

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR										0					
rw										r					

Field	Bits	Type	Description
ADDR	28:6	rw	**Defines bits 28:6 of the lower bound of the access protected region**
ADDRH	31:30	rw	**Defines bits 31:30 of the lower bound of the access protected region**
0	5:0, 29	r	**Reserved** Read as 0; should be written with 0.

10.4.23 Configure region upper address register

The shadow region access enable registers allow atomic update of one region configuration under the control of the PROTRGN state machine. The value of the PROTRGN.SEL register bit-field determines which APU is configured.

This register sets the upper bound of the access protected region. Addresses stricly less than the upper bound are considered part of the access protected region.

ACCENCFG_RGNUA Offset address: $0031C_H$

Configure region upper address register Application Reset value: $DFFF\ FFC0_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADDRH		0	ADDR												
rw		r	rw												

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR										0					
rw										r					

Field	Bits	Type	Description
ADDR	28:6	rw	**Defines bits 28:6 of the upper bound of the access protected region**
ADDRH	31:30	rw	**Defines bits 31:30 of the upper bound of the access protected region**
0	5:0, 29	r	**Reserved** Read as 0; should be written with 0.

10.5 Debug information

This section is not applicable for the module being described.

10.6 References

This section is not applicable for the module being described.

10.7 LMU revision history

Initial release of the chapter.

10.8 TC4Dx LMU information

10.8.1 TC4Dx LMU configuration

The TC4Dx device has ten instances of the LMU named LMU0 to LMU9. Six of these, LMU4 to LMU9, are capable of being powered off.

10.8.2 TC4Dx LMU features

Six instances of the LMU (LMU4 to LMU9) are capable of being powered off to save power if they are not required for the application. None, three or six instances can be powered off after reset if they are not intended to be used. The power is controlled and monitored in two groups of instances powered by the domains named as follows:

- LMU4, LMU5 and LMU6: V_{DDLMU0}
- LMU7, LMU8, and LMU9: V_{DDLMU1}

***Note**: although these two groups of LMU instances can be powered off, they are only powered on after a power-on reset.*

10.8.3 TC4Dx LMU functional description

Instances LMU4 to LMU9 of the LMU have the ability to power down their SRAM according to the controls in the power management system (PMS). When an LMU instance is powered down it will respond to any access to its SRI interface with an SRI error which will trigger an alarm from the SRI.

10.8.4 TC4Dx LMU registers

10.8.4.1 Memory overview tables of LMU

Table 665 Memory overview - LMU (ascending address)

Short name	Long name	Address
LMU0_LMU	LMU0 memory (80000_H Byte)	90400000_H
LMU1_LMU	LMU1 memory (80000_H Byte)	90480000_H
LMU2_LMU	LMU2 memory (80000_H Byte)	90500000_H
LMU3_LMU	LMU3 memory (80000_H Byte)	90580000_H
LMU4_LMU	LMU4 memory (80000_H Byte)	90600000_H
LMU5_LMU	LMU5 memory (80000_H Byte)	90680000_H
LMU6_LMU	LMU6 memory (80000_H Byte)	90700000_H
LMU7_LMU	LMU7 memory (80000_H Byte)	90780000_H
LMU8_LMU	LMU8 memory (80000_H Byte)	90800000_H
LMU9_LMU	LMU9 memory (80000_H Byte)	90880000_H
LMU0_LMU	LMU0 memory (80000_H Byte)	$B0400000_H$
LMU1_LMU	LMU1 memory (80000_H Byte)	$B0480000_H$
LMU2_LMU	LMU2 memory (80000_H Byte)	$B0500000_H$
LMU3_LMU	LMU3 memory (80000_H Byte)	$B0580000_H$
LMU4_LMU	LMU4 memory (80000_H Byte)	$B0600000_H$
LMU5_LMU	LMU5 memory (80000_H Byte)	$B0680000_H$

(table continues...)

Table 665 (continued) Memory overview - LMU (ascending address)

Short name	Long name	Address
LMU6_LMU	LMU6 memory (80000_H Byte)	$B0700000_H$
LMU7_LMU	LMU7 memory (80000_H Byte)	$B0780000_H$
LMU8_LMU	LMU8 memory (80000_H Byte)	$B0800000_H$
LMU9_LMU	LMU9 memory (80000_H Byte)	$B0880000_H$

10.8.4.2 Register address space - LMU

Table 666 Registers address space - LMU

Key: The module name in brackets () indicates a memory section name. Without brackets the reference is to a functional block.

Module	Base address	End address	Note
(LMU0)	90400000_H	$9047FFFF_H$	LMU0 cached RAM address range
	$B0400000_H$	$B047FFFF_H$	LMU0 non-cached RAM address range
LMU0	$FB000000_H$	$FB00FFFF_H$	LMU0 SFR address range
(LMU1)	90480000_H	$904FFFFF_H$	LMU1 cached RAM address range
	$B0480000_H$	$B04FFFFF_H$	LMU1 non-cached RAM address range
LMU1	$FB010000_H$	$FB01FFFF_H$	LMU1 SFR address range
(LMU2)	90500000_H	$9057FFFF_H$	LMU2 cached RAM address range
	$B0500000_H$	$B057FFFF_H$	LMU2 non-cached RAM address range
LMU2	$FB020000_H$	$FB02FFFF_H$	LMU2 SFR address range
(LMU3)	90580000_H	$905FFFFF_H$	LMU3 cached RAM address range
	$B0580000_H$	$B05FFFFF_H$	LMU3 non-cached RAM address range
LMU3	$FB030000_H$	$FB03FFFF_H$	LMU3 SFR address range
(LMU4)	90600000_H	$9067FFFF_H$	LMU4 cached RAM address range
	$B0600000_H$	$B067FFFF_H$	LMU4 non-cached RAM address range
LMU4	$FB040000_H$	$FB04FFFF_H$	LMU4 SFR address range
(LMU5)	90680000_H	$906FFFFF_H$	LMU5 cached RAM address range
	$B0680000_H$	$B06FFFFF_H$	LMU5 non-cached RAM address range
LMU5	$FB050000_H$	$FB05FFFF_H$	LMU5 SFR address range
(LMU6)	90700000_H	$9077FFFF_H$	LMU6 cached RAM address range
	$B0700000_H$	$B077FFFF_H$	LMU6 non-cached RAM address range

(table continues...)

Table 666 (continued) Registers address space - LMU

Key: The module name in brackets () indicates a memory section name. Without brackets the reference is to a functional block.

Module	Base address	End address	Note
LMU6	$FB060000_H$	$FB06FFFF_H$	LMU6 SFR address range
(LMU7)	90780000_H	$907FFFFF_H$	LMU7 cached RAM address range
	$B0780000_H$	$B07FFFFF_H$	LMU7 non-cached RAM address range
LMU7	$FB070000_H$	$FB07FFFF_H$	LMU7 SFR address range
(LMU8)	90800000_H	$9087FFFF_H$	LMU8 cached RAM address range
	$B0800000_H$	$B087FFFF_H$	LMU8 non-cached RAM address range
LMU8	$FB080000_H$	$FB08FFFF_H$	LMU8 SFR address range
(LMU9)	90880000_H	$908FFFFF_H$	LMU9 cached RAM address range
	$B0880000_H$	$B08FFFFF_H$	LMU9 non-cached RAM address range
LMU9	$FB090000_H$	$FB09FFFF_H$	LMU9 SFR address range

10.8.4.3 Register overview - access mode glossary

Table 667 Register overview - access mode glossary

Keyword	Description
RGN	Access protection using PROT register LMU0_PROTRGN or LMU1_PROTRGN or LMU2_PROTRGN or LMU3_PROTRGN or LMU4_PROTRGN or LMU5_PROTRGN or LMU6_PROTRGN or LMU7_PROTRGN or LMU8_PROTRGN or LMU9_PROTRGN .
SE	Access protection using PROT register LMU0_PROTSE or LMU1_PROTSE or LMU2_PROTSE or LMU3_PROTSE or LMU4_PROTSE or LMU5_PROTSE or LMU6_PROTSE or LMU7_PROTSE or LMU8_PROTSE or LMU9_PROTSE .
32	Access only when using 32-bit width.
BE	Always returns a Bus Error.
PROT	Access restrictions as defined in the PROT register access rules.
U	No access restrictions.

10.8.4.4 Register overview - LMU0 (ascending offset address)

Table 668 Register overview - LMU0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
LMU0_ID	Module identification register	00008_H	32	BE	PowerOn Reset	1352
LMU0_PROTSE	Protection for safety registers	00028_H	32	SV, 32, PROT	Application Reset	1353
LMU0_MEMCON	Memory control register	00060_H	32	SV, SE, 32	Application Reset	1354

(table continues...)

Table 668 (continued) Register overview - LMU0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
LMU0_SCTRL	Safety control register	00064_H	32	SV, SE, 32	Application Reset	1356
LMU0_PROTRGN	Protection for memory regions	00070_H	32	SV, 32, PROT	Application Reset	1357
LMU0_RGNx_ACCEN_WRA (x=0-15)	Region write access enable register A x	00080_H+ x*20_H	U	BE	Application Reset	1359
LMU0_RGNx_ACCEN_WRB (x=0-15)	Region write access enable register B x	00084_H+ x*20_H	U	BE	Application Reset	1359
LMU0_RGNx_ACCEN_RDA (x=0-15)	Region read access enable register A x	00088_H+ x*20_H	U	BE	Application Reset	1360
LMU0_RGNx_ACCEN_RDB (x=0-15)	Region read access enable register B x	$0008C_H$+ x*20_H	U	BE	Application Reset	1360
LMU0_RGNx_ACCEN_VM (x=0-15)	Region VM access enable register x	00090_H+ x*20_H	U	BE	Application Reset	1361
LMU0_RGNx_ACCEN_PRS (x=0-15)	Region PRS access enable register x	00094_H+ x*20_H	U	BE	Application Reset	1361
LMU0_RGNx_ACCEN_RGNLA (x=0-15)	Region lower address register x	00098_H+ x*20_H	U	BE	Application Reset	1362
LMU0_RGNx_ACCEN_RGNUA (x=0-15)	Region upper address register x	$0009C_H$+ x*20_H	U	BE	Application Reset	1363
LMU0_ACCENCFG_WRA	Configure region write access enable register A	00300_H	U	RGN, SV	Application Reset	1363
LMU0_ACCENCFG_WRB	Configure region write access enable register B	00304_H	U	RGN, SV	Application Reset	1364
LMU0_ACCENCFG_RDA	Configure region read access enable register A	00308_H	U	RGN, SV	Application Reset	1364
LMU0_ACCENCFG_RDB	Configure region read access enable register B	$0030C_H$	U	RGN, SV	Application Reset	1365
LMU0_ACCENCFG_VM	Configure region VM access enable register	00310_H	U	RGN, SV	Application Reset	1365
LMU0_ACCENCFG_PRS	Configure region PRS access enable register	00314_H	U	RGN, SV	Application Reset	1366
LMU0_ACCENCFG_RGNLA	Configure region lower address register	00318_H	U	RGN, SV	Application Reset	1367

(table continues...)

Table 668 (continued) Register overview - LMU0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
LMU0_ACCENCFG_RGNUA	Configure region upper address register	$0031C_H$	U	RGN, SV	Application Reset	1368

10.8.4.5 Register overview - LMU1 (ascending offset address)

Table 669 Register overview - LMU1 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
LMU1_ID	Module identification register	00008_H	32	BE	PowerOn Reset	1352
LMU1_PROTSE	Protection for safety registers	00028_H	32	SV, 32, PROT	Application Reset	1353
LMU1_MEMCON	Memory control register	00060_H	32	SV, SE, 32	Application Reset	1354
LMU1_SCTRL	Safety control register	00064_H	32	SV, SE, 32	Application Reset	1356
LMU1_PROTRGN	Protection for memory regions	00070_H	32	SV, 32, PROT	Application Reset	1357
LMU1_RGNx_ACCEN_WRA (x=0-15)	Region write access enable register A x	00080_H+ x*20_H	U	BE	Application Reset	1359
LMU1_RGNx_ACCEN_WRB (x=0-15)	Region write access enable register B x	00084_H+ x*20_H	U	BE	Application Reset	1359
LMU1_RGNx_ACCEN_RDA (x=0-15)	Region read access enable register A x	00088_H+ x*20_H	U	BE	Application Reset	1360
LMU1_RGNx_ACCEN_RDB (x=0-15)	Region read access enable register B x	$0008C_H$+ x*20_H	U	BE	Application Reset	1360
LMU1_RGNx_ACCEN_VM (x=0-15)	Region VM access enable register x	00090_H+ x*20_H	U	BE	Application Reset	1361
LMU1_RGNx_ACCEN_PRS (x=0-15)	Region PRS access enable register x	00094_H+ x*20_H	U	BE	Application Reset	1361
LMU1_RGNx_ACCEN_RGNLA (x=0-15)	Region lower address register x	00098_H+ x*20_H	U	BE	Application Reset	1362
LMU1_RGNx_ACCEN_RGNUA (x=0-15)	Region upper address register x	$0009C_H$+ x*20_H	U	BE	Application Reset	1363

(table continues...)

Table 669 (continued) Register overview - LMU1 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
LMU1_ACCENCFG_WRA	Configure region write access enable register A	00300_H	U	RGN, SV	Application Reset	1363
LMU1_ACCENCFG_WRB	Configure region write access enable register B	00304_H	U	RGN, SV	Application Reset	1364
LMU1_ACCENCFG_RDA	Configure region read access enable register A	00308_H	U	RGN, SV	Application Reset	1364
LMU1_ACCENCFG_RDB	Configure region read access enable register B	$0030C_H$	U	RGN, SV	Application Reset	1365
LMU1_ACCENCFG_VM	Configure region VM access enable register	00310_H	U	RGN, SV	Application Reset	1365
LMU1_ACCENCFG_PRS	Configure region PRS access enable register	00314_H	U	RGN, SV	Application Reset	1366
LMU1_ACCENCFG_RGNLA	Configure region lower address register	00318_H	U	RGN, SV	Application Reset	1367
LMU1_ACCENCFG_RGNUA	Configure region upper address register	$0031C_H$	U	RGN, SV	Application Reset	1368

10.8.4.6 Register overview - LMU2 (ascending offset address)

Table 670 Register overview - LMU2 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
LMU2_ID	Module identification register	00008_H	32	BE	PowerOn Reset	1352
LMU2_PROTSE	Protection for safety registers	00028_H	32	SV, 32, PROT	Application Reset	1353
LMU2_MEMCON	Memory control register	00060_H	32	SV, SE, 32	Application Reset	1354
LMU2_SCTRL	Safety control register	00064_H	32	SV, SE, 32	Application Reset	1356
LMU2_PROTRGN	Protection for memory regions	00070_H	32	SV, 32, PROT	Application Reset	1357
LMU2_RGNx_ACCEN_WRA (x=0-15)	Region write access enable register A x	00080_H+ x*20_H	U	BE	Application Reset	1359
LMU2_RGNx_ACCEN_WRB (x=0-15)	Region write access enable register B x	00084_H+ x*20_H	U	BE	Application Reset	1359
LMU2_RGNx_ACCEN_RDA (x=0-15)	Region read access enable register A x	00088_H+ x*20_H	U	BE	Application Reset	1360

(table continues...)

Table 670 (continued) Register overview - LMU2 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
LMU2_RGNx_ACCEN_RDB (x=0-15)	Region read access enable register B x	$0008C_H$+ x*20_H	U	BE	Application Reset	1360
LMU2_RGNx_ACCEN_VM (x=0-15)	Region VM access enable register x	00090_H+ x*20_H	U	BE	Application Reset	1361
LMU2_RGNx_ACCEN_PRS (x=0-15)	Region PRS access enable register x	00094_H+ x*20_H	U	BE	Application Reset	1361
LMU2_RGNx_ACCEN_RGNLA (x=0-15)	Region lower address register x	00098_H+ x*20_H	U	BE	Application Reset	1362
LMU2_RGNx_ACCEN_RGNUA (x=0-15)	Region upper address register x	$0009C_H$+ x*20_H	U	BE	Application Reset	1363
LMU2_ACCENCFG_WRA	Configure region write access enable register A	00300_H	U	RGN, SV	Application Reset	1363
LMU2_ACCENCFG_WRB	Configure region write access enable register B	00304_H	U	RGN, SV	Application Reset	1364
LMU2_ACCENCFG_RDA	Configure region read access enable register A	00308_H	U	RGN, SV	Application Reset	1364
LMU2_ACCENCFG_RDB	Configure region read access enable register B	$0030C_H$	U	RGN, SV	Application Reset	1365
LMU2_ACCENCFG_VM	Configure region VM access enable register	00310_H	U	RGN, SV	Application Reset	1365
LMU2_ACCENCFG_PRS	Configure region PRS access enable register	00314_H	U	RGN, SV	Application Reset	1366
LMU2_ACCENCFG_RGNLA	Configure region lower address register	00318_H	U	RGN, SV	Application Reset	1367
LMU2_ACCENCFG_RGNUA	Configure region upper address register	$0031C_H$	U	RGN, SV	Application Reset	1368

10.8.4.7 Register overview - LMU3 (ascending offset address)

Table 671 Register overview - LMU3 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
LMU3_ID	Module identification register	00008_H	32	BE	PowerOn Reset	1352
LMU3_PROTSE	Protection for safety registers	00028_H	32	SV, 32, PROT	Application Reset	1353

(table continues...)

Table 671 (continued) Register overview - LMU3 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
LMU3_MEMCON	Memory control register	00060_H	32	SV, SE, 32	Application Reset	1354
LMU3_SCTRL	Safety control register	00064_H	32	SV, SE, 32	Application Reset	1356
LMU3_PROTRGN	Protection for memory regions	00070_H	32	SV, 32, PROT	Application Reset	1357
LMU3_RGNx_ACCEN_WRA (x=0-15)	Region write access enable register A x	00080_H+ x*20_H	U	BE	Application Reset	1359
LMU3_RGNx_ACCEN_WRB (x=0-15)	Region write access enable register B x	00084_H+ x*20_H	U	BE	Application Reset	1359
LMU3_RGNx_ACCEN_RDA (x=0-15)	Region read access enable register A x	00088_H+ x*20_H	U	BE	Application Reset	1360
LMU3_RGNx_ACCEN_RDB (x=0-15)	Region read access enable register B x	$0008C_H$+ x*20_H	U	BE	Application Reset	1360
LMU3_RGNx_ACCEN_VM (x=0-15)	Region VM access enable register x	00090_H+ x*20_H	U	BE	Application Reset	1361
LMU3_RGNx_ACCEN_PRS (x=0-15)	Region PRS access enable register x	00094_H+ x*20_H	U	BE	Application Reset	1361
LMU3_RGNx_ACCEN_RGNLA (x=0-15)	Region lower address register x	00098_H+ x*20_H	U	BE	Application Reset	1362
LMU3_RGNx_ACCEN_RGNUA (x=0-15)	Region upper address register x	$0009C_H$+ x*20_H	U	BE	Application Reset	1363
LMU3_ACCENCFG_WRA	Configure region write access enable register A	00300_H	U	RGN, SV	Application Reset	1363
LMU3_ACCENCFG_WRB	Configure region write access enable register B	00304_H	U	RGN, SV	Application Reset	1364
LMU3_ACCENCFG_RDA	Configure region read access enable register A	00308_H	U	RGN, SV	Application Reset	1364
LMU3_ACCENCFG_RDB	Configure region read access enable register B	$0030C_H$	U	RGN, SV	Application Reset	1365
LMU3_ACCENCFG_VM	Configure region VM access enable register	00310_H	U	RGN, SV	Application Reset	1365
LMU3_ACCENCFG_PRS	Configure region PRS access enable register	00314_H	U	RGN, SV	Application Reset	1366

(table continues...)

Table 671 (continued) Register overview - LMU3 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
LMU3_ACCENCFG_RGNLA	Configure region lower address register	00318_H	U	RGN, SV	Application Reset	1367
LMU3_ACCENCFG_RGNUA	Configure region upper address register	$0031C_H$	U	RGN, SV	Application Reset	1368

10.8.4.8 Register overview - LMU4 (ascending offset address)

Table 672 Register overview - LMU4 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
LMU4_ID	Module identification register	00008_H	32	BE	PowerOn Reset	1352
LMU4_PROTSE	Protection for safety registers	00028_H	32	SV, 32, PROT	Application Reset	1353
LMU4_MEMCON	Memory control register	00060_H	32	SV, SE, 32	Application Reset	1354
LMU4_SCTRL	Safety control register	00064_H	32	SV, SE, 32	Application Reset	1356
LMU4_PROTRGN	Protection for memory regions	00070_H	32	SV, 32, PROT	Application Reset	1357
LMU4_RGNx_ACCEN_WRA (x=0-15)	Region write access enable register A x	00080_H+ x*20_H	U	BE	Application Reset	1359
LMU4_RGNx_ACCEN_WRB (x=0-15)	Region write access enable register B x	00084_H+ x*20_H	U	BE	Application Reset	1359
LMU4_RGNx_ACCEN_RDA (x=0-15)	Region read access enable register A x	00088_H+ x*20_H	U	BE	Application Reset	1360
LMU4_RGNx_ACCEN_RDB (x=0-15)	Region read access enable register B x	$0008C_H$+ x*20_H	U	BE	Application Reset	1360
LMU4_RGNx_ACCEN_VM (x=0-15)	Region VM access enable register x	00090_H+ x*20_H	U	BE	Application Reset	1361
LMU4_RGNx_ACCEN_PRS (x=0-15)	Region PRS access enable register x	00094_H+ x*20_H	U	BE	Application Reset	1361
LMU4_RGNx_ACCEN_RGNLA (x=0-15)	Region lower address register x	00098_H+ x*20_H	U	BE	Application Reset	1362

(table continues...)

Table 672 (continued) Register overview - LMU4 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
LMU4_RGNx_ACCEN_RGNUA (x=0-15)	Region upper address register x	$0009C_H$+ x*20_H	U	BE	Application Reset	1363
LMU4_ACCENCFG_WRA	Configure region write access enable register A	00300_H	U	RGN, SV	Application Reset	1363
LMU4_ACCENCFG_WRB	Configure region write access enable register B	00304_H	U	RGN, SV	Application Reset	1364
LMU4_ACCENCFG_RDA	Configure region read access enable register A	00308_H	U	RGN, SV	Application Reset	1364
LMU4_ACCENCFG_RDB	Configure region read access enable register B	$0030C_H$	U	RGN, SV	Application Reset	1365
LMU4_ACCENCFG_VM	Configure region VM access enable register	00310_H	U	RGN, SV	Application Reset	1365
LMU4_ACCENCFG_PRS	Configure region PRS access enable register	00314_H	U	RGN, SV	Application Reset	1366
LMU4_ACCENCFG_RGNLA	Configure region lower address register	00318_H	U	RGN, SV	Application Reset	1367
LMU4_ACCENCFG_RGNUA	Configure region upper address register	$0031C_H$	U	RGN, SV	Application Reset	1368

10.8.4.9 Register overview - LMU5 (ascending offset address)

Table 673 Register overview - LMU5 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
LMU5_ID	Module identification register	00008_H	32	BE	PowerOn Reset	1352
LMU5_PROTSE	Protection for safety registers	00028_H	32	SV, 32, PROT	Application Reset	1353
LMU5_MEMCON	Memory control register	00060_H	32	SV, SE, 32	Application Reset	1354
LMU5_SCTRL	Safety control register	00064_H	32	SV, SE, 32	Application Reset	1356
LMU5_PROTRGN	Protection for memory regions	00070_H	32	SV, 32, PROT	Application Reset	1357
LMU5_RGNx_ACCEN_WRA (x=0-15)	Region write access enable register A x	00080_H+ x*20_H	U	BE	Application Reset	1359
LMU5_RGNx_ACCEN_WRB (x=0-15)	Region write access enable register B x	00084_H+ x*20_H	U	BE	Application Reset	1359

(table continues...)

Table 673 (continued) Register overview - LMU5 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
LMU5_RGNx_ACCEN_RDA (x=0-15)	Region read access enable register A x	00088_H+ x*20_H	U	BE	Application Reset	1360
LMU5_RGNx_ACCEN_RDB (x=0-15)	Region read access enable register B x	$0008C_H$+ x*20_H	U	BE	Application Reset	1360
LMU5_RGNx_ACCEN_VM (x=0-15)	Region VM access enable register x	00090_H+ x*20_H	U	BE	Application Reset	1361
LMU5_RGNx_ACCEN_PRS (x=0-15)	Region PRS access enable register x	00094_H+ x*20_H	U	BE	Application Reset	1361
LMU5_RGNx_ACCEN_RGNLA (x=0-15)	Region lower address register x	00098_H+ x*20_H	U	BE	Application Reset	1362
LMU5_RGNx_ACCEN_RGNUA (x=0-15)	Region upper address register x	$0009C_H$+ x*20_H	U	BE	Application Reset	1363
LMU5_ACCENCFG_WRA	Configure region write access enable register A	00300_H	U	RGN, SV	Application Reset	1363
LMU5_ACCENCFG_WRB	Configure region write access enable register B	00304_H	U	RGN, SV	Application Reset	1364
LMU5_ACCENCFG_RDA	Configure region read access enable register A	00308_H	U	RGN, SV	Application Reset	1364
LMU5_ACCENCFG_RDB	Configure region read access enable register B	$0030C_H$	U	RGN, SV	Application Reset	1365
LMU5_ACCENCFG_VM	Configure region VM access enable register	00310_H	U	RGN, SV	Application Reset	1365
LMU5_ACCENCFG_PRS	Configure region PRS access enable register	00314_H	U	RGN, SV	Application Reset	1366
LMU5_ACCENCFG_RGNLA	Configure region lower address register	00318_H	U	RGN, SV	Application Reset	1367
LMU5_ACCENCFG_RGNUA	Configure region upper address register	$0031C_H$	U	RGN, SV	Application Reset	1368

10.8.4.10 Register overview - LMU6 (ascending offset address)

Table 674 Register overview - LMU6 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
LMU6_ID	Module identification register	00008_H	32	BE	PowerOn Reset	1352

(table continues...)

Table 674 (continued) Register overview - LMU6 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
LMU6_PROTSE	Protection for safety registers	00028_H	32	SV, 32, PROT	Application Reset	1353
LMU6_MEMCON	Memory control register	00060_H	32	SV, SE, 32	Application Reset	1354
LMU6_SCTRL	Safety control register	00064_H	32	SV, SE, 32	Application Reset	1356
LMU6_PROTRGN	Protection for memory regions	00070_H	32	SV, 32, PROT	Application Reset	1357
LMU6_RGNx_ACCEN_WRA (x=0-15)	Region write access enable register A x	00080_H+ x*20_H	U	BE	Application Reset	1359
LMU6_RGNx_ACCEN_WRB (x=0-15)	Region write access enable register B x	00084_H+ x*20_H	U	BE	Application Reset	1359
LMU6_RGNx_ACCEN_RDA (x=0-15)	Region read access enable register A x	00088_H+ x*20_H	U	BE	Application Reset	1360
LMU6_RGNx_ACCEN_RDB (x=0-15)	Region read access enable register B x	$0008C_H$+ x*20_H	U	BE	Application Reset	1360
LMU6_RGNx_ACCEN_VM (x=0-15)	Region VM access enable register x	00090_H+ x*20_H	U	BE	Application Reset	1361
LMU6_RGNx_ACCEN_PRS (x=0-15)	Region PRS access enable register x	00094_H+ x*20_H	U	BE	Application Reset	1361
LMU6_RGNx_ACCEN_RGNLA (x=0-15)	Region lower address register x	00098_H+ x*20_H	U	BE	Application Reset	1362
LMU6_RGNx_ACCEN_RGNUA (x=0-15)	Region upper address register x	$0009C_H$+ x*20_H	U	BE	Application Reset	1363
LMU6_ACCENCFG_WRA	Configure region write access enable register A	00300_H	U	RGN, SV	Application Reset	1363
LMU6_ACCENCFG_WRB	Configure region write access enable register B	00304_H	U	RGN, SV	Application Reset	1364
LMU6_ACCENCFG_RDA	Configure region read access enable register A	00308_H	U	RGN, SV	Application Reset	1364
LMU6_ACCENCFG_RDB	Configure region read access enable register B	$0030C_H$	U	RGN, SV	Application Reset	1365
LMU6_ACCENCFG_VM	Configure region VM access enable register	00310_H	U	RGN, SV	Application Reset	1365

(table continues...)

Table 674 (continued) Register overview - LMU6 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
LMU6_ACCENCFG_PRS	Configure region PRS access enable register	00314_H	U	RGN, SV	Application Reset	1366
LMU6_ACCENCFG_RGNLA	Configure region lower address register	00318_H	U	RGN, SV	Application Reset	1367
LMU6_ACCENCFG_RGNUA	Configure region upper address register	$0031C_H$	U	RGN, SV	Application Reset	1368

10.8.4.11 Register overview - LMU7 (ascending offset address)

Table 675 Register overview - LMU7 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
LMU7_ID	Module identification register	00008_H	32	BE	PowerOn Reset	1352
LMU7_PROTSE	Protection for safety registers	00028_H	32	SV, 32, PROT	Application Reset	1353
LMU7_MEMCON	Memory control register	00060_H	32	SV, SE, 32	Application Reset	1354
LMU7_SCTRL	Safety control register	00064_H	32	SV, SE, 32	Application Reset	1356
LMU7_PROTRGN	Protection for memory regions	00070_H	32	SV, 32, PROT	Application Reset	1357
LMU7_RGNx_ACCEN_WRA (x=0-15)	Region write access enable register A x	00080_H+ x*20_H	U	BE	Application Reset	1359
LMU7_RGNx_ACCEN_WRB (x=0-15)	Region write access enable register B x	00084_H+ x*20_H	U	BE	Application Reset	1359
LMU7_RGNx_ACCEN_RDA (x=0-15)	Region read access enable register A x	00088_H+ x*20_H	U	BE	Application Reset	1360
LMU7_RGNx_ACCEN_RDB (x=0-15)	Region read access enable register B x	$0008C_H$+ x*20_H	U	BE	Application Reset	1360
LMU7_RGNx_ACCEN_VM (x=0-15)	Region VM access enable register x	00090_H+ x*20_H	U	BE	Application Reset	1361
LMU7_RGNx_ACCEN_PRS (x=0-15)	Region PRS access enable register x	00094_H+ x*20_H	U	BE	Application Reset	1361

(table continues...)

Table 675 (continued) Register overview - LMU7 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
LMU7_RGNx_ACCEN_RGNLA (x=0-15)	Region lower address register x	00098_H+ x*20_H	U	BE	Application Reset	1362
LMU7_RGNx_ACCEN_RGNUA (x=0-15)	Region upper address register x	$0009C_H$+ x*20_H	U	BE	Application Reset	1363
LMU7_ACCENCFG_WRA	Configure region write access enable register A	00300_H	U	RGN, SV	Application Reset	1363
LMU7_ACCENCFG_WRB	Configure region write access enable register B	00304_H	U	RGN, SV	Application Reset	1364
LMU7_ACCENCFG_RDA	Configure region read access enable register A	00308_H	U	RGN, SV	Application Reset	1364
LMU7_ACCENCFG_RDB	Configure region read access enable register B	$0030C_H$	U	RGN, SV	Application Reset	1365
LMU7_ACCENCFG_VM	Configure region VM access enable register	00310_H	U	RGN, SV	Application Reset	1365
LMU7_ACCENCFG_PRS	Configure region PRS access enable register	00314_H	U	RGN, SV	Application Reset	1366
LMU7_ACCENCFG_RGNLA	Configure region lower address register	00318_H	U	RGN, SV	Application Reset	1367
LMU7_ACCENCFG_RGNUA	Configure region upper address register	$0031C_H$	U	RGN, SV	Application Reset	1368

10.8.4.12 Register overview - LMU8 (ascending offset address)

Table 676 Register overview - LMU8 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
LMU8_ID	Module identification register	00008_H	32	BE	PowerOn Reset	1352
LMU8_PROTSE	Protection for safety registers	00028_H	32	SV, 32, PROT	Application Reset	1353
LMU8_MEMCON	Memory control register	00060_H	32	SV, SE, 32	Application Reset	1354
LMU8_SCTRL	Safety control register	00064_H	32	SV, SE, 32	Application Reset	1356
LMU8_PROTRGN	Protection for memory regions	00070_H	32	SV, 32, PROT	Application Reset	1357
LMU8_RGNx_ACCEN_WRA (x=0-15)	Region write access enable register A x	00080_H+ x*20_H	U	BE	Application Reset	1359

(table continues...)

Table 676 (continued) Register overview - LMU8 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
LMU8_RGNx_ACCEN_WRB (x=0-15)	Region write access enable register B x	00084_H+ x*20_H	U	BE	Application Reset	1359
LMU8_RGNx_ACCEN_RDA (x=0-15)	Region read access enable register A x	00088_H+ x*20_H	U	BE	Application Reset	1360
LMU8_RGNx_ACCEN_RDB (x=0-15)	Region read access enable register B x	$0008C_H$+ x*20_H	U	BE	Application Reset	1360
LMU8_RGNx_ACCEN_VM (x=0-15)	Region VM access enable register x	00090_H+ x*20_H	U	BE	Application Reset	1361
LMU8_RGNx_ACCEN_PRS (x=0-15)	Region PRS access enable register x	00094_H+ x*20_H	U	BE	Application Reset	1361
LMU8_RGNx_ACCEN_RGNLA (x=0-15)	Region lower address register x	00098_H+ x*20_H	U	BE	Application Reset	1362
LMU8_RGNx_ACCEN_RGNUA (x=0-15)	Region upper address register x	$0009C_H$+ x*20_H	U	BE	Application Reset	1363
LMU8_ACCENCFG_WRA	Configure region write access enable register A	00300_H	U	RGN, SV	Application Reset	1363
LMU8_ACCENCFG_WRB	Configure region write access enable register B	00304_H	U	RGN, SV	Application Reset	1364
LMU8_ACCENCFG_RDA	Configure region read access enable register A	00308_H	U	RGN, SV	Application Reset	1364
LMU8_ACCENCFG_RDB	Configure region read access enable register B	$0030C_H$	U	RGN, SV	Application Reset	1365
LMU8_ACCENCFG_VM	Configure region VM access enable register	00310_H	U	RGN, SV	Application Reset	1365
LMU8_ACCENCFG_PRS	Configure region PRS access enable register	00314_H	U	RGN, SV	Application Reset	1366
LMU8_ACCENCFG_RGNLA	Configure region lower address register	00318_H	U	RGN, SV	Application Reset	1367
LMU8_ACCENCFG_RGNUA	Configure region upper address register	$0031C_H$	U	RGN, SV	Application Reset	1368

10.8.4.13 Register overview - LMU9 (ascending offset address)

Table 677 Register overview - LMU9 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
LMU9_ID	Module identification register	00008_H	32	BE	PowerOn Reset	1352
LMU9_PROTSE	Protection for safety registers	00028_H	32	SV, 32, PROT	Application Reset	1353
LMU9_MEMCON	Memory control register	00060_H	32	SV, SE, 32	Application Reset	1354
LMU9_SCTRL	Safety control register	00064_H	32	SV, SE, 32	Application Reset	1356
LMU9_PROTRGN	Protection for memory regions	00070_H	32	SV, 32, PROT	Application Reset	1357
LMU9_RGNx_ACCEN_WRA (x=0-15)	Region write access enable register A x	00080_H+ x*20_H	U	BE	Application Reset	1359
LMU9_RGNx_ACCEN_WRB (x=0-15)	Region write access enable register B x	00084_H+ x*20_H	U	BE	Application Reset	1359
LMU9_RGNx_ACCEN_RDA (x=0-15)	Region read access enable register A x	00088_H+ x*20_H	U	BE	Application Reset	1360
LMU9_RGNx_ACCEN_RDB (x=0-15)	Region read access enable register B x	$0008C_H$+ x*20_H	U	BE	Application Reset	1360
LMU9_RGNx_ACCEN_VM (x=0-15)	Region VM access enable register x	00090_H+ x*20_H	U	BE	Application Reset	1361
LMU9_RGNx_ACCEN_PRS (x=0-15)	Region PRS access enable register x	00094_H+ x*20_H	U	BE	Application Reset	1361
LMU9_RGNx_ACCEN_RGNLA (x=0-15)	Region lower address register x	00098_H+ x*20_H	U	BE	Application Reset	1362
LMU9_RGNx_ACCEN_RGNUA (x=0-15)	Region upper address register x	$0009C_H$+ x*20_H	U	BE	Application Reset	1363
LMU9_ACCENCFG_WRA	Configure region write access enable register A	00300_H	U	RGN, SV	Application Reset	1363
LMU9_ACCENCFG_WRB	Configure region write access enable register B	00304_H	U	RGN, SV	Application Reset	1364
LMU9_ACCENCFG_RDA	Configure region read access enable register A	00308_H	U	RGN, SV	Application Reset	1364

(table continues...)

Table 677 (continued) Register overview - LMU9 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
LMU9_ACCENCFG_RDB	Configure region read access enable register B	$0030C_H$	U	RGN, SV	Application Reset	1365
LMU9_ACCENCFG_VM	Configure region VM access enable register	00310_H	U	RGN, SV	Application Reset	1365
LMU9_ACCENCFG_PRS	Configure region PRS access enable register	00314_H	U	RGN, SV	Application Reset	1366
LMU9_ACCENCFG_RGNLA	Configure region lower address register	00318_H	U	RGN, SV	Application Reset	1367
LMU9_ACCENCFG_RGNUA	Configure region upper address register	$0031C_H$	U	RGN, SV	Application Reset	1368

10.8.4.14 Device specific registers

There are no device specific register changes.

10.8.5 TC4Dx LMU connectivity

The table below lists interfaces to and from this functional block to other blocks in the device.

Table 678 List of LMU interface signals

Interface signals	I/O	Description
LMU0_SMU_ALARM	Out	LMU0 alarm
LMU1_SMU_ALARM	Out	LMU1 alarm
LMU2_SMU_ALARM	Out	LMU2 alarm
LMU3_SMU_ALARM	Out	LMU3 alarm
LMU4_SMU_ALARM	Out	LMU4 alarm
LMU5_SMU_ALARM	Out	LMU5 alarm
LMU6_SMU_ALARM	Out	LMU6 alarm
LMU7_SMU_ALARM	Out	LMU7 alarm
LMU8_SMU_ALARM	Out	LMU8 alarm
LMU9_SMU_ALARM	Out	LMU9 alarm

10.8.6 TC4Dx LMU revision history

Initial release of the chapter.

11 CSRM - NDA only

Information for this functional block is only available under NDA.

12 CSS - NDA only

Information for this functional block is only available under NDA.

13 PPU - NDA only

Information for this functional block is only available under NDA.

14 Gigabit Ethernet (GETH)

The Gigabit Ethernet (GETH) is a functional block that implements Ethernet functions. GETH is connected to SRI with a master interface and with a slave interface with a set of special function registers (SFRs).

14.1 Feature list

- Ethernet port capable of supporting 10M, 100M, 1G, 2.5G, 5G speed in full-duplex mode
- Ethernet port capable of supporting half-duplex mode in 10M and 100M speed
- Host port with single master interface (64-bit wide data bus) for data transfer
- Multichannel DMA Engine on the host interface to transfer data with minimal software intervention
- Single 32-bit Slave interface for configuration by the application
- Ethernet port supports the hardware requirements of IEEE 802.1 AVB and TSN specifications as listed below:
  - IEEE 802.1Qav – Forwarding and Queueing enhancements for Time Sensitive Streams
  - IEEE 802.1AS 2020 – Timing and Synchronization for Time Sensitive Applications
  - IEEE 802.1Qbu – Frame Preemption
  - IEEE 802.1Qbv – Enhancements for Scheduled Traffic
- Classification of traffic between the host port and the ingress/egress port
- Flexible/programmable packet header inspection for filtering, monitoring schemes and enable firewall protection and intrusion detection service
- Bridge function (in products with Ethernet Bridge only):
  - The bridge connects two Ethernet ports and one common host interface
  - It allows to statically establish the data paths between these three blocks. This is called forwarding of Ethernet Frames
- Automotive Safety Features:

  The automotive safety features improve the reliability and reduce the time taken to detect faults. GETH supports the following features for automotive safety:
  - Error correction code (ECC) protection for memories
  - FSM parity and timeout protection
  - Application/CSR interface timeout protection

14.2 Functional overview

This module provides a communication capability according to the Ethernet IEEE 802.1 standard.

The following figure shows the basic architecture.

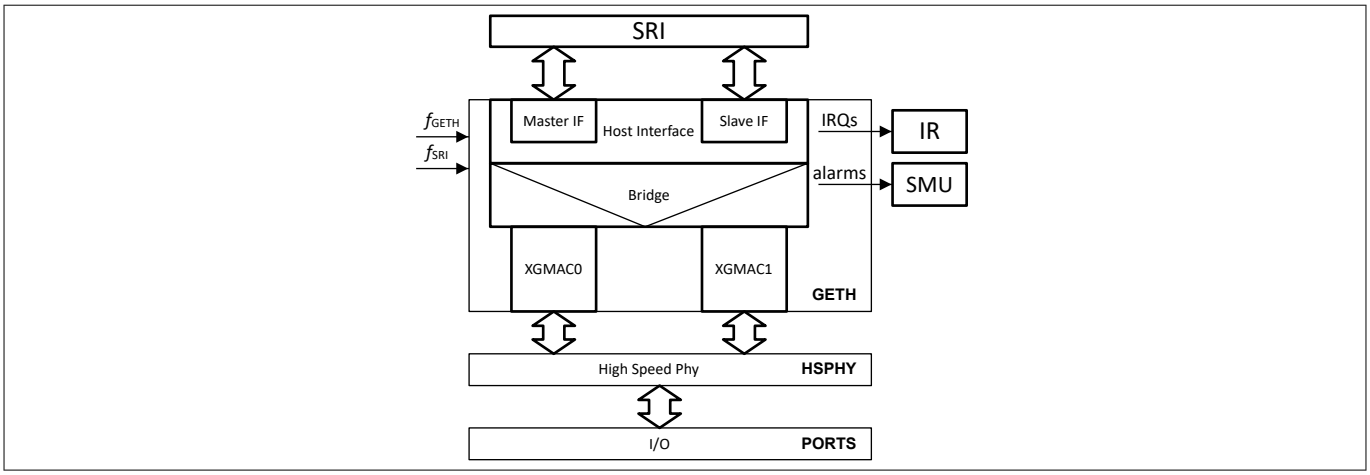


Figure 129 GETH block diagram

For the alarm description please refer to the alarm mapping tables in the SMU functional block and the Safety Manual.

The module consists of the following blocks:

- Host Interface
  - Master interface for transfer of Ethernet-frame-data to and from host
  - Slave interface for configuration
- Bridge (exists in products with Ethernet Bridge only)
  - Connects two XGMACs and the host
  - Forwards Ethernet frames
    - From the host to a XGMAC
    - From the two XGMACs to the host
    - From one XGMAC to the other XGMAC
- XGMAC core
  - Contains the blocks for protocol handling
  - Contains Ethernet frame filters and forwarding rules
  - Contains buffers for Ethernet frames

Note that GETH has no own connection (with the exception of PPS outputs) to I/O but they are routed to the High Speed Phy (HSPHY) which controls the connections with the actual I/O in the PORTS. PPS outputs are connected directly to PORTS output. Please refer to High Speed Physical Layer (HSPHY) chapter for correct setup.

HSPHY provides connection to physical pads for the following interfaces.

- SGMII
- UXSGMII
- RGMII
- RMII
- MII
- MDIO

***Note**: Kindly refer to HSPHY device specific chapter for availability of an interface in the corresponding device.*

Related information

TC4Dx SMU alarm mapping tables on page 7233

14.3 Functional description

The functional description is divided in two parts:

- XGMAC (10 gigabit media access controller)
- Bridge (exists in products with Ethernet Bridge only)

Related information

XGMAC on page 1396
Bridge on page 1594

14.3.1 Detailed architecture

This section describes the architecture of GETH in detail. This information is needed only if performance needs to be optimized.

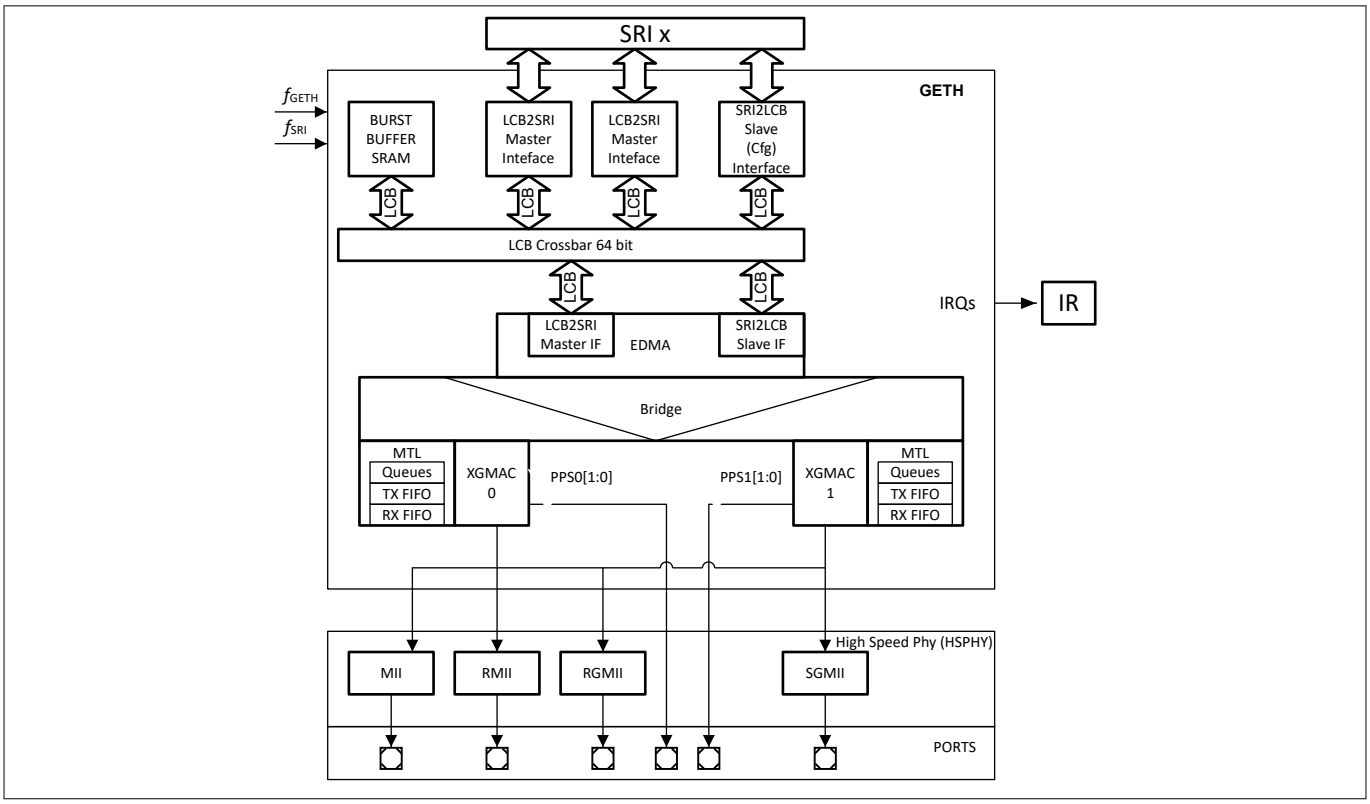


Figure 130 GETH detailed block diagram

The module consists of the following blocks:

- Local Cross Bar (LCB)
  - 64 bit wide
  - Two master interfaces to SRI
  - One slave interface to SRI

  - One SRAM with 64 KBytes (for products with Ethernet Bridge only)
  - One EDMA with
    - One master interface for transfer of Ethernet-frame-data to and from SRI through the local cross bar
    - One slave interface for configuration
- DMA
  - Eight communication channels
  - Communicates through two data structures:
    - Control and Status registers (CSR)
    - Descriptor lists and data buffers
  - Packet transfer is optimized by the following tasks in parallel:
    - Multiple Descriptor Fetch
    - Data Transfer
    - Descriptor Write-back
- Bridge (for products with Ethernet Bridge only) provides:
  - Forwarding of frames from each MAC to the other
  - Transfer of RX frames from each MAC to DMA
  - Transfer of TX frames from the DMA to each MAC
- MAC transaction layer provides
  - Data transmit storage (TX FIFO)
  - Data receive storage (RX FIFO)
  - Data queues for prioritized traffic
- MAC core
  - Containing the blocks for protocol handling

GETH has two LCB2SRI connections. This can be used by applications where sustained data throughput to and from the SRI system is very critical. One LCB2SRI can be used to perform only read transfers and the other to perform only write transfers. This can be achieved by locating the data buffers for TX frames in the address space that is addressed by the first master and the data buffers for RX frames in the address space that is addressed by the second master. This way the whole buffer depth of an LCB2SRI can be fully used by frame data of one direction. Ideally the descriptors shall be located in the local buffer RAM. This will avoid blocking situations on the LCB2SRI caused by single word transfers. Such transfers are performed by the EDAM for example to write back the status of the descriptor or to clear the OWN bit of the descriptor.

***Note**:* *The local buffer RAM provides performance gain. It does not support full channel-separation ("virtualization support"). The APUs are located in the slave interface from SRI to GETH. They protect the address space of the GETH against outside accesses. For example, they protect against accesses from a CPU to a MAC or the local buffer RAM. These APUs cannot protect accesses from MAC DMAs to addresses inside GETH like the local buffer RAM. Still, it is possible to setup eight segments with individual ACCEN protection that will protect each segment of the local buffer RAM against accesses from outside GETH. In addition, MAC DMAs cannot address MAC registers. Such a DMA receives an error response and stop before the access takes place.*

14.3.2 Access protection

The address space of GETH is divided into 11 areas that can be protected with individual APUs.

The APUs are located in the slave interface from SRI to GETH. They protect the address space of the GETH from outside accesses, for example from a CPU or the system DMA. Accesses from MAC DMAs to addresses inside GETH can not be protected by these APUs.

The following APUs are available:

- ACCENGLOBAL allows protection of the Bridge specific CSRs (in products with Ethernet Bridge only) and the bus peripheral interface (BPI) registers (CLC, RST_CTRLA, RST_CTRLB, RST_STAT, MACEN and ID)
- ACCENMAC[1:0] allows per MAC protection of the CSRs except the DMA channel specific ones
- ACCENCH[7:0] allows protection of the CSRs per channel consisting of DMA_CHx, MTL_TCQx and memory region of the local buffer RAM in terms of 8 kByte segments. The first 8Kbyte of RAM segment protected by ACCENCH0 and correspondingly the last 8 kByte protected by ACCENCH7

***Note**: All registers shall be declared as "volatile". Note that volatile bits are not marked as"rh", "wh" or "rwh" in the bit access restriction field but as "r", "w" and "rw" respectively. The bit description field defines the actual behavior verbally*

14.3.3 Master TAG ID

The module has 8 DMA Channels that share one master interface connecting them to the SRI bus. In order to distinguish the 8 DMAs from each other in the system, the master tag ID will dynamically be changed depending on the currently active DMA. The following table details which ID is presented for each DMA. Accesses to descriptors are tagged according to the DMA channel requesting the access.

Table 679 Master TAG IDs for Gigabit Ethernet

DMA	Master TAG ID
DMA0	$0x30_H$
DMA1	$0x31_H$
DMA2	$0x32_H$
DMA3	$0x33_H$
DMA4	$0x34_H$
DMA5	$0x35_H$
DMA6	$0x36_H$
DMA7	$0x37_H$

14.3.4 SRI slave and master interface error notification

The following tables describe the Error detection methods and corresponding notification method for SRI slave and master interface of GETH.

Table 680 SRI slave interface error notification method

Description of error	Notification method
SRI Address phase ECC errors	SRI transaction error and SRI XBAR Bus Error Event Alarm
AXI write error	SRI Write Data phase alarm

Table 681 SRI master interface error notification method

Description of error	Notification method
SRI read data ECC errors	SRC_GETHSE alarm

14.3.5 System related considerations

The following chapters gives recommendation on system related topics.

14.3.5.1 System start-up

Before using the bridge (exists in products with Ethernet Bridge only) with one or both of the MACs the application has to execute the following steps:

- High-Speed Physical Layer (HSPHY) needs to be configured first
- Switch on f_{GETH} by configuring in the SCU bit-field SYSCCUCON1.GETHDIV
- Clear GETH_ CLC.DISR and
- Set GETH_MACEN.MAC0 or MAC1 or both depending on the MAC to use

If the application wants to use the IP in RGMII mode, the application has to execute the following steps:

- Prior to the application reset, the application must switch on f_{GETH} by configuring in the SCU SYSCCUCON1.GETHDIV
- Attach an external 125 MHz clock to input GREFCLK
- Activate the application reset
- Wait for 10 µs

***Note**: CLC.DISR is used to pause the GETH, whilst MACEN.MAC0 and MACEN.MAC1 are used to completely turn off the MAC cores.*

- *CLC.DISR = 1: GETH disables the application clock (on SRI side) and the asynchronous Tx clocks (on PHY side).*

  CLC.DISR does not disable the asynchronous Rx clock, because it is needed in order to wake-up the rest of the GETH in the event that wake-up frames are received. In this event, the small receive portion of the MAC, which is still active and clocked with the Rx clock will generate an interrupt and the software can turn on the rest of the GETH (CLC.DISR = 0).

  There is only one CLC.DISR for the two MAC cores. Writing it to 1 disables for both MAC cores the application clock as well as the Tx asynchronous clocks
- *MACEN.MAC0 and MACEN.MAC1: Used to enable or disable the MAC cores individually. If any of this register fields are zero, the corresponding MAC core is disabled and its asynchronous Rx clock is turned off. The corresponding MAC core is turned off completely and cannot be awaken even in the case of wake-up frames*

14.3.5.2 Reset

This chapter explains how to bring the IP to reset state.

There are two sources of reset provided by the module for it's operation:

- Kernel reset: The complete functional block GETH supports a general reset, the so called kernel reset by setting KRST_CTRLA.KRST and RST_CTRLB.KRST. This resets all registers in the wrapper and the kernel
- Software Reset: The XGMAC kernel IP inside GETH supports an own software reset by setting DMA_Mode.SWR. When this bit is set, the MAC and the DMA controller reset the logic and all internal registers of the DMA, MTL, and MAC

Any reset operation is completed only after all resets in all active clock domains are de-asserted. Therefore, it is essential that all PHY inputs clocks (applicable for the selected PHY interface) are present for reset completion.

If one of these resets (SW reset or kernel reset) is applied this reset will propagate to SSH (SRAM support hardware) controlling the FIFO RAMs of GETH. The RAM content will be no longer reliable. The kernel logic will have lost all information about its content.

Note that in both cases both MACs (Port0_MAC... and Port1_MAC...) will be reset. It is not possible to individually reset only one MAC port or execute a partial reset for MTL and MAC port (without the DMA).

The recommended way to achieve an equivalent result is to gracefully stop all the DMA channels allocated/mapped to a MAC port, their connected TxQ/RxQ and MAC port through programming and bring them to the default/reset state.

For TX side, follow this programming sequence:

1. Stop the associated TxDMA connected/mapped to a MAC port
2. Check the status of the TxDMA. If all the TxDMA went to STOP state, it means that it has gracefully completed the ongoing packet transfers
3. Check the status of the MTL TxQ of that port. If they are empty, it means that all pending packets are transmitted
4. Disable the MTL TxQs and MAC TX

This moves the required MAC port (MAC + MTL + DMA) to idle/initial state (equivalent to reset).

For RX side, adopt a similar strategy but in reverse way:

1. Stop the MAC receiver of the port
2. Check the status (fill-level) of all the MTL RxQ of that port. If it is empty, all the packets of that port have been transferred to the host
3. Stop all the RxDMAs associated with that port

This moves the required MAC port (MAC + MTL + DMA) to idle/initial state (equivalent to reset).

14.3.5.3 Buffer and descriptor alignment

The GETH is implemented as a 32-bit peripheral. Nevertheless it is connected to 64-bit wide bus (SRI). To make full use of the possible performance of SRI and its bridges, the data buffers and the descriptors need to be aligned to 64-bit addresses.

The start address and length of descriptors (containing pointers to data buffers) and data buffers (with the actual frame data) must be aligned to the burst size for performance optimization. This is needed by the AXI2SRI bridges to forward the burst. Others alignments will result in single (word) transfers.

14.3.5.4 DMA burst lengths limitations by the system

Other than specified in the XGMAC IP only the following burst lengths are supported by the system memories: SINGLE, INCR2, INCR4, INCR8, INCR16. As consequence, in DMA_SysBus_Mode only the following burst length modes are allowed: BLEN4, BLEN8 and BLEN16. Do not choose UBL, BLEN32 through BLEN128, they can not be processed by the system and an alarm (SRC_GETHSE) is generated.

Note that the INCR16 is translated into two BTR8 on SRI.

Note that the AXI standard defines that a burst must not cross a 4 KB address boundary. You can set this limit in DMA_SysBus_Mode.ONEKBBE.

In addition, the system does not support outstanding transactions nor bursts reordering or data interleaving.

14.4 XGMAC

This chapter gives the overview of the configuration, features and detailed specifications of the XGMAC.

Software configuration is done by the application by writing to registers.

- Up to eight Tx queues or DMA channels
- Up to eight Rx queues in Receive FIFO
- Audio Video (AV) Bridging
- Up to eight Rx DMA channels
- 32-bit addressing support
- Up to 32 kBytes of receive memory in Transaction layer
- Up to 32 kBytes of transmit memory in Transaction layer
- 64-bit application interface data bus width (Master IF)
- VLAN Tag based perfect and hash filtering in Rx
- SA and VLAN insertion in Tx
- Up to 32 MAC address registers for filtering
- RMON counters (MMC) with individual register selection
- Power Management Module (PMT) with magic packet and remote wake-up frame processing options
- Checksum Offload Engine (TCP/IP layer) for the Transmit and Receive paths
- IEEE 1588 time-stamping and PTP support
- Energy Efficient Ethernet
- Station Management Agent (SMA) MDIO module
- Transmit Packet Status
  - Maximum number of packets in Transmit FIFO = 2
- Transmit Time stamping FIFO depth = 4
- Time-Based Scheduling (TBS) enabled on all DMA channels
- RGMII PHY interface support
- RMII PHY interface support
- Number of synchronizer stages in CDC path = 2

***Note**: Products with a bridge use two identical instances of this XGMAC. This is why the description is done only one time here. The two XGMACs in a subsystem with a bridge are referenced as Port0 and Port1 in the register chapter.*

14.4.1 Feature list

The features of XGMAC are described in the chapters below.

Related information

XGMAC-CORE on page 1397
DMA on page 1399
AXI interface on page 1400
Audio video support on page 1400
Time Sensitive Networking Features on page 1401
Automotive safety features on page 1401
Monitoring, testing, and debugging support on page 1401

14.4.1.1 XGMAC-CORE

The XGMAC (XGMAC controller) includes the following features:

- Full-duplex operation at 5 Gbps/2.5 Gbps (XGMII), 2.5 Gbps/1 Gbps (GMII), 100 Mbps/10 Mbps(MII)
- Full-duplex RGMII support (1 Gbps/100 Mbps/10 Mbps)
- Full-duplex RMII support (100 Mbps/10 Mbps)
- Half-duplex operation in 10 Mbps/100 Mbps modes. Supports:
  - CSMA/CD protocol
  - Flow control using back pressure support
- Communication with an external 5 gigabit/2.5 gigabit PHY provided by an XGMII interface that fully complies with Clause 46 of the IEEE 802.3-2008 standard
- Link fault detection in the XGMII receiver and automatic fault signaling in the XGMII Transmit interface
- Communication with an external 1 Gbps/100 Mbps/10 Mbps PHY provided by a GMII/MII interface that fully complies with Clause 35/72 of the IEEE 802.3-2008 standard. Existing XGMII I/O can be used for GMII/MII operation, or a separate GMII/MII I/O can be configured
- 2.5 Gbps operation using an 8-bit GMII interface operating at 312.5 MHz
- 1 Gbps/100 Mbps/10 Mbpps operation using 4-bit RGMII interface working at 125/25/2.5 MHz
- Clause 78 (Energy Efficient Ethernet (EEE) feature) of the IEEE 802.3az, standard support
- Separate transmission, reception, and slave (control and status register) interfaces to the application
- 64-bit data transfer interface on the system side
- Preamble, SFD, and other control character insertion (in transmit path) and deletion (in receive path)
- Programmable minimum Inter-Packet Gap (IPG) from 64 bit times to 224 bit times
- Padding in transmit side for short (less-than-64 byte long) frames received from the application. Pad stripping on the receive side for short frames
- Programmable source address inclusion or replacement for Transmit frames
- Programmable frame length, supporting standard or jumbo (extendable to 16 KB) Ethernet frames if the communication rate is equal to or below 100 Mbit/s
  - Support for jumbo frames up to 9 KB expandable up to 16 KB
  - Truncation support for oversized frame reception and transmission at fixed thresholds of 2,048, 10,240, or 16,383 bytes. Alternatively, the truncation thresholds can be programmed in steps of 1 KB from 2 KB onwards
- Supports IEEE 802.3 flow-control
  - Forwarding of received PAUSE control frames to the application
  - Automatic transmission of zero-quanta PAUSE frames on flow-control input deassertion
  - Support for unicast PAUSE frame reception
  - Transmission of PAUSE frames triggered by software controlled register bit or based on queue fill level
- Support for VLAN-tagged frame processing in compliance with the IEEE 802.1Q standard
  - Support for VLAN-tagged frame detection, stripping, and filtering
  - Support for VLAN tag inclusion, replacement, or deletion in transmit frames
  - Support for up to two Stacked-VLAN tagged packets
- Support for the following flexible address filtering modes for received frames:
  - Up to 32 destination address (DA) perfect match address filters with masks for each byte
  - Up to 32 source address (SA) perfect match filters with masks for each byte
  - Posibility to pass all multicast-addressed frames
  - Promiscuous mode to pass all frames without any filtering for network monitoring
  - Up to eight VLAN-based perfect matched and hash filtering

  - Up to eight Layer 3 and Layer 4-based (TCP or UDP over IPv4 or IPv6) match filters
  - Pass all incoming packets with a status report as determined by the filter
- Forward all control frames other than PAUSE control frames
- Filter unicast control frames that fail the DA address perfect match filter
- Network statistics with RMON or MIB Counters (RFC2819/RFC2665)
- Features to support Ethernet packet timestamping as described in IEEE 1588-2008 (64-bit timestamps given in the Tx or Rx status of each packet)
  - Possibility to store up to 16 timestamps with packet identifier in the CSR
  - One-step timestamping in Tx direction with the host providing a 64-bit timestamp along with TX packet. XGMAC-CORE parses PTP (over Ethernet) packets and insert or correct timestamp in appropriate header field
  - Parses received frames for PTP packets and takes timestamps as per configured node
  - Provides registers for asymmetric time correction
  - Supports internal System time generation and update
  - Support for dynamic selection of reference time source through control bit in CSR
  - Flexibility to control the Pulse-Per-Second (PPS) output signal (ptp_pps_o)
  - Support for capturing timestamp based on external trigger and providing it in CSR
- Loopback capability at XGMII/GMII (transmit data to receive path)
- Detection of LAN wake-up frames and AMD Magic Packet frames during the power-down mode. Wake-on LAN support up to four filters
- MDIO Master interface for multiple PHY devices and their configuration and management
- Interrupt generation mechanism for MDIO
- IPv4 header checksum offload on reception
- TCP, UDP, or ICMP checksum offload (IPv4 and IPv6) for received packets

14.4.1.2 DMA

The DMA block exchanges the data between the MTL block and the host memory. The host can use a set of registers (DMA CSR) to control the DMA operations. The DMA block supports the following features:

- 64-bit data transfers
- Multi-channel Transmit and Receive engines (eight Transmit and eight Receive channels)
- Separate DMA channel in the Transmit path for each queue in MTL. Single or multiple DMA channels for any number of queues in MTL Receive path
- Descriptor structure to support the following:
  - Byte-aligned addressing for data buffer
  - Dual-buffer descriptor ring and support for 32-bit addressing in descriptor structure. Expandable descriptor with context descriptor for additional control and status information
  - Descriptor architecture to allow large blocks of data transfer with minimum CPU intervention (each descriptor can transfer up to 8 KB of data)
  - IEEE 1588 timestamps
  - One-step PTP time correction in Tx, that is, input 64-bit timestamp provided in descriptor
  - Possibility to provide 64-bit timestamp for received packets
  - Possibility to provide up to two VLAN tags in descriptor for insertion or replacement in Tx and stripped VLAN tags in Rx
  - Possibility to split the packet header (Layer 3 and Layer 4) and Layer 4 payload in different buffers

- Individual programmable burst size for Tx DMA and Rx DMA engines for optimal host bus utilization
- Routing of received packets to the DMA channels based on the Ethernet DA or VLAN Priority in multi-channel DMA configurations
- Register programmable RX buffer sizes for each DMA channel
- Additional per-DMA channel interrupt for normal packet transfers
- Per-packet Transmit and Receive Interrupt control in descriptors
- Supports pre-fetching and caching up to 16 descriptors for each DMA channel
- Descriptor fetches, Data transfers and Descriptor writes are de-linked and performed independently so as to reduce gaps between data transfers of each DMA
- TxDMA engines have capability to issue up to 2 outstanding requests for packet/burst transfer
- DMA descriptor writes can be “posted”, so that next descriptor can be processed immediately for next packet
- Descriptor memory cache
- Time-sensitive conditional packet transmission by comparing the Slot Time information provided in the descriptor (useful for AV applications)

14.4.1.3 AXI interface

This section discusses the AXI Interface features.

AXI master interface

The AXI master interface is compatible to interface with AXI-4 slave ports. It supports the following features:

- 32-bit address width, little-endian
- 64-bit data
- OKAY, SLVERR, and DECERR responses from AXI slaves
- Software-selected type of AXI burst
- AXI 4 KB boundary burst splitting and option to limit burst splitting boundary to 1 KB boundary
- Outstanding read and write transactions with register programmable control. Two outstanding requests per channel in configuration of eight TxDMA and eight RxDMA channels
- One of the outstanding requests is reserved for high priority Descriptor operation
- Posted writes during multiple data transfers from the AXI master interface to maximize the bus utilization
- AXI-ID based on DMA channel number. Each TX and RX DMA has its own ID for data transfer

The AXI master interface does not generate the following:

- Atomic, exclusive, locked, and protected accesses
- EXOKAY access

AXI slave interface

The AXI Slave interface supports the following features:

- 32-bit for CSR access
- 32-bit address width
- Little-endian
- No narrow burst and FIXED or INCR burst

The AXI slave interface does not support the Exclusive, Atomic, Locked, Cache, and Protected accesses.

14.4.1.4 Audio video support

XGMAC supports the following Audio Video (AV) features:

- Separate channels or queues for AV data transfer
- Up to eight queues on the Receive paths for AV traffic and 3 queues on the Transmit path for AV traffic
- IEEE 802.1-Qav specified credit-based shaper (CBS) algorithm for Transmit channels
- Single Tx FIFO and Rx FIFO (MTL) for all selected queues

14.4.1.5 Time Sensitive Networking Features

XGMAC supports the following Time Sensitive Networking (TSN) features:

- IEEE 802.1Qbv-2015, Enhancements to Scheduling Traffic
- IEEE 802.1Qbu/802.3br, Frame preemption and Interspersing Express Traffic
- IEEE 802.1Qci-2017, Stream-Gate Filtering and Policing
- Time Based Scheduling

14.4.1.6 Automotive safety features

The automotive safety features are targeted to improve the reliability and reduce the time taken to detect faults in the controller:

XGMAC supports the following features for automotive safety:

- Error correction code (ECC) protection for memories
- FSM parity and timeout protection
- Application/CSR interface timeout protection
- On chip data path parity and AXI ECC are not connected in this device

14.4.1.7 Monitoring, testing, and debugging support

- Support for internal loopback on the XGMII/GMII for debugging
- DMA states (Tx and Rx) as status bits
- Current Tx or Rx Buffer pointer as status registers
- Current Tx or Rx Descriptor pointer as status registers
- Tx or Rx memory accessible through Slave port for debug
- Provides debug status register that gives the status of FSMs in the Transmit and Receive data paths and FIFO fill-levels
- Provides complete network ethernet statistics) with RMON-MIB counters (RFCs 2819 and 2665)

14.4.2 Functional overview

This chapter provides an overview of the XGMAC and describes its components and features.

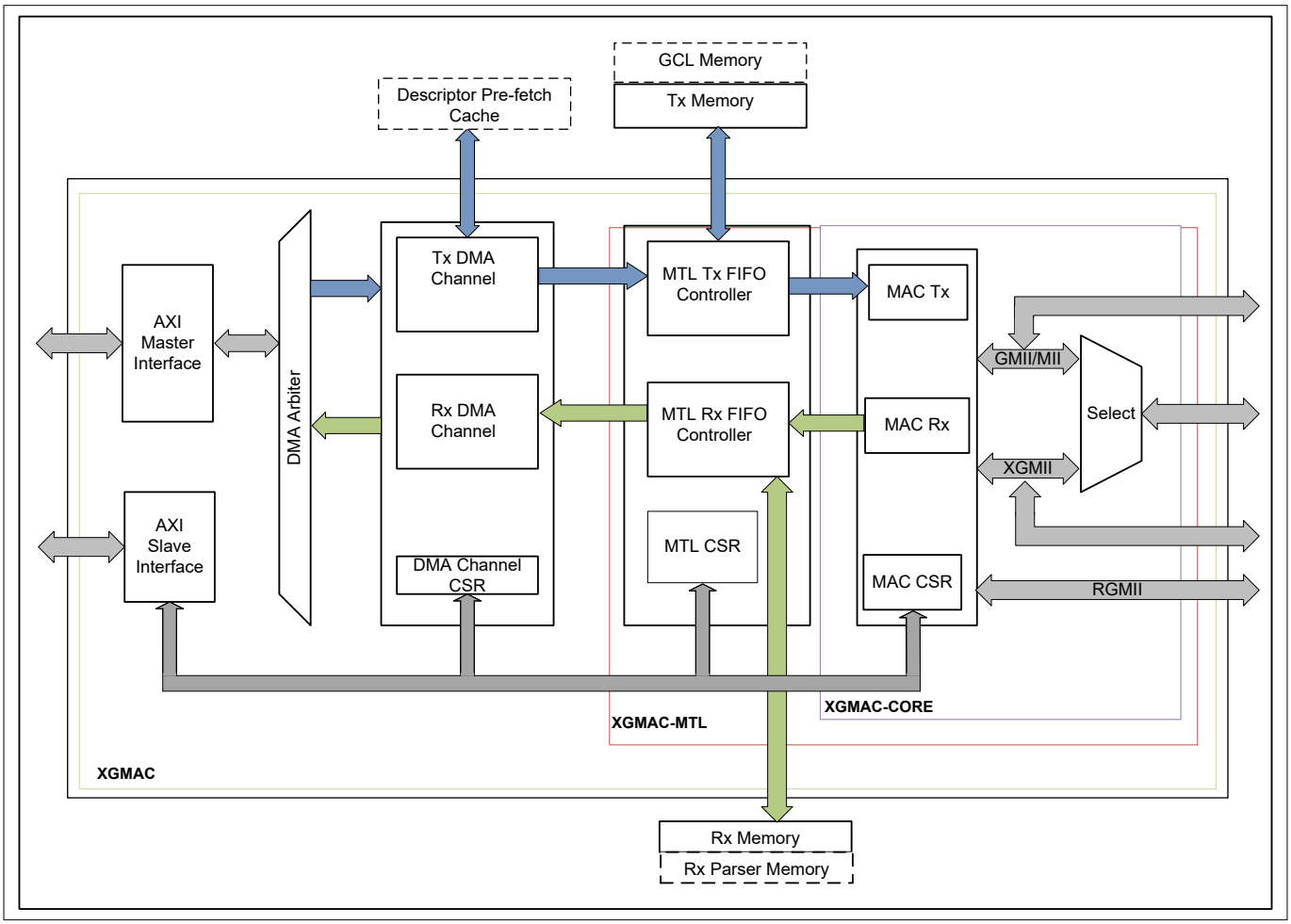


Figure 131 XGMAC block diagram

The XGMAC enables a host to transmit and receive data over Ethernet in compliance with the IEEE 802.3-2008 standard.

Interfaces

The XGMAC has the following interfaces.

AXI interface

The AXI interface transfers the data to and from the system memory as a master interface. The host CPU uses the AXI slave interface to access the Control and Status registers (CSRs).

14.4.2.1 Standard Compliance

In addition to the default interfaces defined in the IEEE 802.3 specifications, the XGMAC supports several industry standard interfaces to the PHY. The XGMAC is compliant with the following standards:

- IEEE 802.1AS 2020 and 802.1-Qav-2009 for Audio Video (AV) traffic
- IEEE 802.3-2015 for Ethernet MAC, Gigabit Media Independent Interface (GMII), 10G Media Independent Interface (XGMII).
- IEEE 1588-2008 for precision networked clock synchronization
- IEEE 802.3az-2010 for Energy Efficient Ethernet (EEE)

- AMBA 3.0 and APB3 slave ports
- AMBA4 AXI and ACE protocol specification, February 2013, ARM Ltd for AXI4 and APB4 interface
- NBASE-T Alliance 2.5 and 5 Gigabit Ethernet (USXGMII)
- Reduced Gigabit Media Independent Interface (RGMII), Version 2.6, Broadcom Corp., Hewlett-Packard Co., Marvell Technology Group, Ltd.
- RMII specification version 1.2 from RMII consortium

14.4.2.2 Buffers and memories

This gives an overview of the buffers and memories used by one XGMAC.

Transmit and receive FIFOs

The Transmit FIFO (Tx FIFO) buffers the data transferred from the application to the XGMAC. Similarly, the Receive FIFO (Rx FIFO) stores the Ethernet frames received from the line until they can be transferred to the application. These are asynchronous FIFOs because they also transfer the data between the application clock and the MAC line clocks. Tx memory and Rx memory are of 64-bit data width.

The size of the Tx FIFO is 32 kBytes

The size of the Rx FIFO is 32 kBytes

Descriptor cache

The memory is used to store the pre-fetched descriptors of all the DMAs active in the controller. Each DMA (Tx or Rx) has allocated space for caching 16 descriptors (256 bytes).

Enhancements to scheduled traffic (EST)

This memory is used for the Gate Control List (The list in the controller memory that holds the gate controls and the associated time intervals) in the 802.1Qbv EST function. The memory depth is 256 lines.

14.4.3 Functional description

The following sub-section give the details on the functions of XGMAC.

14.4.3.1 Introduction

The Figure 131 provides a block diagram of the XGMAC subsystem of 2 ports with a bridge.

The main building layers of the XGMAC are:

- XGMAC-CORE:

  Using CSR, software can configure the XGMAC-CORE to operate at 5/2.5G(XGMII), 1/2.5G (GMII), 100M/10M (MII and RMII) or 1G/100M/10M (RGMII) speeds. The XGMAC-CORE contains the MAC Transmitter and Receiver functions specified in the IEEE 802.3-2008/RMII/RGMII standard. It interfaces to the line side with the XGMII/GMII/MII/RGMII/RMII and with a native interface to the system side. These system side interfaces, called the MAC Transmit Interface (MTI) and the MAC Receive Interface (MRI) are synchronous, respectively, to the XGMII/GMII/MII/RGMII/RMII Transmit and XGMII/GMII/MII/RGMII/RMII Receive clocks. In addition to these interfaces, the application can access the Control and Status Registers (CSR) through a MAC Control Interface (MCI) or APB3
- XGMAC-MTL:

  This layer introduces a FIFO buffer into the transmit and receive paths. It interfaces to the system with native interfaces called the Application Transmit Interface (ATI) and the Application Receive Interface (ARI). These interfaces are synchronous to the system clock
- XGMAC-AXI

  The AXI interface transfers the data to and from the system memory through AXI4-compatible master interface. The host CPU uses the AXI slave interface to access the Control and Status registers (CSRs) of the XGMAC subsystem

14.4.3.2 DMA Controller

The DMA has independent Transmit (Tx) and Receive (Rx) engines, and a CSR space. The Tx engine transfers data from the system memory to the device port (MTL), whereas the Rx engine transfers data from the device port to the system memory. The controller uses descriptors to efficiently move data from source to destination with minimal application CPU intervention. The DMA is designed for packet-oriented data transfers such as packets in Ethernet. The controller can be programmed to interrupt the application CPU for situations such as Packet Transmit and Receive Transfer completion, and other normal or error conditions.

The DMA and the application communicate through the following two data structures:

- Control and Status registers (CSR)
- Descriptor lists and data buffers

The DMA transfers the data packets received by the MAC to the Rx Buffer in system memory and Tx data packets from the Tx Buffer in the system memory. The descriptors that reside in the system memory contains the pointers to these buffers.

The DMA supports up to 8 Tx and 8 Rx Descriptor lists (or DMA channels). The base address of each list is written to the respective Tx Descriptor List Address register and Rx Descriptor List Address register. The descriptor list is forward linked and the next descriptor is always considered at a fixed offset to the current one. The offset is controlled by the programming of the DSL field of DMA_Ch(#i)_Control register. The number of descriptors in the list is programmed in the respective Tx (or Rx) Descriptor Ring Length register. After the DMA processes the last descriptor in the list, it jumps back to the descriptor in the List Address register to create a descriptor ring.

The descriptor lists reside in the physical memory address space of the application. Each descriptor can point to a maximum of two buffers. This enables two buffers to be used, physically addressed, rather than contiguous buffers in memory.

A data buffer resides in the application physical memory space and consists of an entire packet or part of a packet but cannot exceed a single packet. Buffers contain only data. Buffer status is maintained in the descriptor. Data chaining refers to packets that span multiple data buffers. However, a single descriptor cannot span multiple packets. The DMA skips to the data buffer of next packet when EOP is detected.

The XGMAC supports the ring structure for the DMA descriptor. For details about descriptor structure and how the DMA accesses the descriptors, see chapter "Descriptors".

The XGMAC-AXI always uses DMA controller for descriptor fetch, descriptor write-back and data transfer. In DMA, the efficiency of packet transfer is optimized by the following tasks in parallel:

- Multiple Descriptor Fetch
- Data Transfer
- Descriptor write-back.

All these DMA phases work independently in a pipelined fashion such that the descriptor fetch of a new packet can be performed along with data transfer of the previous packet and the descriptor write-back of the last to previous packet. In this way, packet transfers throughput is improved with minimal gaps between them.

***Note**: In case of Rx DMAs, the Data transfer Phase and Descriptor write-back phase cannot be made totally independent, since the order of the data transfer and descriptor write-back has to be strictly followed for maintaining synchronization between the software processing of descriptors and the packet transfers. This ensures that the software processes the closed descriptors only after the corresponding packet buffer is transferred and written into system memory.*

When DSPW bit in the DMA_Mode register is set, the descriptor writes are made as posted writes. XGMAC does not wait for the Write response from the AXI slave to take further actions. This makes the interrupt generation based on the IOC bit in the Descriptor unreliable. Therefore, when DSPW is set, the IOC bit in the Rx Descriptor should not be set.

Related information

Descriptors on page 1559

14.4.3.2.1 Application Bus Burst Access

When configured through Bit 0 of the DMA_SysBus_Mode register, the DMA attempts to execute the fixed-length burst transfers on the AXI master interface. The PBL field of Transmit Control and Receive Control registers of respective DMA channel indicates and limits the maximum burst length. The Rx and Tx descriptors are always accessed in the maximum possible (limited by PBL or 16 * 8/64) burst length for 16 bytes to be read. For more details please refer to DMA_SysBus_Mode register.

The Tx DMA initiates a data transfer only when sufficient space is available in the MTL Tx Queue to accommodate the minimum of the following:

- Bytes corresponding to the configured burst (PBL * 64/8)
- Remaining bytes in Tx Buffer without EOP
- Number of bytes till EOP

The DMA indicates the start address and the number of transfers required to the AXI master interface. When the AXI Interface is configured for fixed-length burst, it transfers the data using the best combination of INCR4, INCR8, or INCR16 or higher if enabled and SINGLE transactions. Note that the SRI system supports fixed-length burst only and only up to BTR16. You must not configure the DMA for using any undefined burst length (INCRx) transactions.

The Rx DMA initiates a data transfer in the following conditions:

- Sufficient data is available in MTL Rx Queue to accommodate the configured burst
- EOP (when it is less than the configured burst length) is detected in the Rx Queue (possible only in store and forward mode)

The DMA indicates the start address and the number of transfers required to the AXI master interface. If EOP is reached before the fixed-burst ends on the AXI interface, dummy transfers are performed in-order to complete the fixed-burst. Otherwise (Bit 0 of DMA_SysBus_Mode register is reset), the DMA transfers the data using INCRx transactions.

When the AXI interface is configured for address-aligned beats, both DMA engines ensure that the first burst transfer initiated by the AXI is less than or equal to the size of the configured PBL. Therefore, all subsequent beats start at an address that is aligned to the configured PBL.

***Note**: The DMA can align the address for up to 256 beats because the AXI interface does support up to BLEN 256. But if a system memory is addressed only SINGLE, BLEN4, BLEN8 and BLEN16 are allowed as the SRI bridge is limited to BTR16.*

14.4.3.2.2 Application Data Buffer Alignment

The Tx and Rx data buffers do not have any restrictions on start address alignment. The start address for buffers can be aligned to an unaligned address. However, the DMA always initiates write transfers with address aligned to the bus width and dummy data (old data) in the invalid byte lanes. This typically happens during the transfer of the beginning or end of an Ethernet packet. The software driver should discard the dummy bytes based on the start address of the buffer and size of the packet.

Table 682 Application Data Buffer Alignment Examples

Examples	
Buffer Read	If the Tx buffer address is 32'h00000FF2 (for 128-bit data bus), and 32 bytes need to be transferred, the DMA reads three full words from address 32'h00000FF0, but when transferring data to the MTL Tx queue, the additional bytes (the first two bytes) are dropped or ignored. Similarly, the last 14 bytes of the last transfer are also ignored. The DMA always ensures that it transfers a full 32-bit data to the MTL Tx queue, unless it is the end of packet.
Buffer Write	If the Rx buffer address is 32'h0000FF2 (for 64-bit data bus) and 16 bytes of a received packet need to be transferred, the DMA writes 3 full words from address 32'h00000FF0. However, the first 2 bytes of the first transfer and the last 6 bytes of the third transfer have dummy data. The DMA considers the offset address only if it is the first Rx buffer of the packet. The DMA ignores the offset address and performs full word writes for the middle and the last Rx buffer of the packet.

14.4.3.2.3 Buffer Size Calculations

The DMA does not provide Buffer size in the Rx descriptors. The DMA updates only the status fields (RDES) of the descriptors. The driver has to perform the size calculations.

The Tx DMA transfers the exact number of bytes (indicated by buffer size field of TDES2) towards the MAC. If a descriptor is marked as first (FD bit of TDES3 is set), the DMA marks the first transfer from the buffer as SOP. If a descriptor is marked as last (LD bit of TDES3), the DMA marks the last transfer from that data buffer as EOP to the MTL.

The Rx DMA transfers data to a buffer until the buffer is full or the end of packet is received from the MTL. When the FD bit of a descriptor is set, the amount of valid data in a buffer is accurately indicated by its buffer size field minus the data buffer pointer offset. The offset is zero when the data buffer pointer is aligned to the data bus width. If a descriptor is marked as last, the buffer may not be full (as indicated by the buffer size in Bits [14:1] of

Receive Control Register). To compute the amount of valid data in this final buffer, the driver must read the packet length (PL bits of RDES3[13:0]) and subtract the sum of the buffer sizes of the preceding buffers in this packet. The Rx DMA always transfers the start of next packet with a new descriptor.

***Note**: Even when the start address of a Rx buffer is not aligned to the data width of system bus, the system should allocate a Rx buffer of a size aligned to the system bus width. For example, if the system allocates a Rx buffer of 1,024 bytes (1 KB) starting from address 0x1000, the software can program the buffer start address in the Rx descriptor to 0x1002.*

The Rx DMA writes the packet to this buffer with dummy data in the first two locations (0x1000 and 0x1001). The actual packet is written from location 0x1002. Therefore, the actual useful space in this buffer is 1,022 bytes, even though the buffer size is programmed as 1,024 bytes, because of the start address offset.

14.4.3.2.4 Error response to DMA

For any data transfer initiated by a DMA channel, if the slave replies with an error response, the DMA stops all operations and updates the error bits and the Fatal Bus Error bit in the Status Register of corresponding DMA channel. The application can either perform a reset to XGMAC or re-initialize the DMA descriptor list and start again. The rest of the DMA channels are not affected by such errors.

14.4.3.2.5 DMA Descriptor Fetch Operation

Each DMA (TX or RX) has allocated space for caching 16 descriptors (256 bytes).

Therefore, the DMA attempts to fetch multiple descriptors in a burst. The maximum burst length during descriptor fetches for a DMA Channel, is dependent on

- Programmed Burst Length in DMA_CH(#i)_Tx_Control register (TxPBL)
- Space available in the Descriptor Cache for that DMA Channel
- Number of descriptors created by the Host (Offset of the Tail Pointer with respect to the last fetched descriptor)
- The Address Alignment requirement with respect to current descriptor address in case bit[12] AAL of DMA_SysBus_Mode is set to 1

The XGMAC-AXI selects minimum of these, to calculate the burst length for descriptor fetches and fill-up the cache. After the current descriptor burst fetch is completed, the TxDMA/RxDMA engines do not initiate further descriptor fetches until the space available in the descriptor cache falls below the threshold size programmed in the respective DMA_Tx_EDMA_Control or DMA_Rx_EDMA_Control register.

The DMA engine also checks the OWN bit (xDES3[31]) of the descriptor to confirm whether the descriptor contents are valid and can be processed by the DMA. In case any of the descriptor has its OWN bit = 0, then, that descriptor and the rest of the descriptors in that burst are dropped and not written into the cache. The DMA tries to fetch the descriptors again in a burst of length decided as per the conditions, from that point onwards.

The descriptors in the cache are read and given to the Data Transfer engine. Once the Data Transfer engine accepts the descriptor, the next descriptor is read from the cache and thus free up the cache space.The Buffer Unavailable (TBU/RBU) interrupt is generated when all the descriptors created by the host (as indicated by the Tail Pointer) are fetched. It is asserted even if some descriptors are still available in the cache for processing. This advance interrupt provides the opportunity for the software to create more descriptors in advance before the cache gets emptied out completely.

The xBU interrupt is also generated when the DMA engine finds that the OWN bit = 0 for two successive fetches of the same descriptor. It then goes into SUSPEND mode and waits for the software to wake it up by updating/ writing into the Tail Pointer register.

***Note**: A descriptor ring must provide a minimum of four descriptors or more to the XGMAC.*

14.4.3.2.6 DMA Tx Data Transfer Operation

The data transfer operation by TxDMA is as follows:

1. The Descriptor fetch engine reads a valid descriptor (OWN=1) either from the system memory or the descriptor pre-fetch cache and gives it to the Data transfer engine.
2. It processes the descriptor control bits, buffer sizes and calculates the size of data transfer to be requested as per the settings of the registers (TxPBL). It checks if space is available in the corresponding TxQ before it issues the request for data transfer.
3. Once the request is accepted by the AXI master (for scheduling on the bus), this engine calculates the next data transfer size and issues another request. The second request can be for one of the following:
   a. Next burst of data transfer from the same buffer in case the first request is only for part of the buffer
   b. Burst of data transfer from Buffer 2 in case the first request completed the transfer of Buffer 1
   c. Burst of data transfer from Buffer 1 of next descriptor in case the first request completed Buffer 2 (or Buffer 2 was empty)
   d. Start of new packet transfer from Buffer of next descriptor in case the first request completes a packet transfer.
4. At any given time, up to two requests can be outstanding for data transfer completion.
5. A request is considered to be completed once the requested data is fetched and written to the corresponding TxQ in MTL. After this, the engine can issue the next request as given in step 3.
6. After the valid buffers of a descriptor is fetched, the descriptor is freed-up and pushed to the Descriptor write-back engine for closure and the next descriptor from the pre-fetched cache is accepted for processing (Step 2).

All these operations are implemented in a pipe-lined fashion so as to avoid any bottlenecks and overheads due to descriptor reads or writes. Moreover, the engine can request for two packet transfers in sequence (for packets of size smaller than the programmed PBL) without completing the fetch of the first one. This allows the hiding of the system latencies of a read command behind the data transfer of the previous command and thus improve the throughput.

14.4.3.2.7 DMA Rx Data Transfer Operation

The data transfer operation by the RxDMA is different from that of the TxDMA and is as follows:

1. The Descriptor fetch engine reads a valid descriptor (OWN=1) either from the system memory or the descriptor pre-fetch cache and gives it to the Data transfer engine.
2. The data transfer engine indicates its readiness to transfer data to the MTL RxQ read Controller along with the supported burst-size length as calculated from the buffer sizes and as per the register settings (RxPBL).
3. When the RxQ Read controller selects an RxQ (in case of multiple RxQ configuration) and triggers the RxDMA to start the data transfer.
4. The RxDMA engine issues the request which is then accepted and driven by the AXI master. This requested data is then transferred to the buffer in system memory through the AXI write channel.
5. Once the requested data transfer is internally completed (driven on the AXI bus), the data transfer engine jumps back to Step 2 in case the packet transfer is not over (no EOP transferred) in step 4.

6. If both the RxBuffers, pointed in the descriptor are full, it pushes the descriptor to the “Descriptor write-back engine” along with the intermediate status and then accept the next descriptor from the pre-fetch cache and go to Step 2.
7. If the EOP is transferred during the last burst transfer, the engine pushes the descriptor to be closed along with the final packet status read from the RxQ after the EOP and then jump to step 2.

The descriptor pre-fetch and the descriptor closure activities are implemented in a pipe-lined fashion so as to avoid any bottlenecks and overheads due to descriptor reads. However, in step 2, the engine does not initiate another data transfer request unless the previous descriptor write transfer is indicated to be complete as explained in the next section.

***Note**: Instead of DMA_CHj_Rx_Data_Xfer_Ring_Offset register indicating the ring offset of the descriptor currently processed by the receive DMA and is transferring the packet data to the buffers specified in the descriptor, it indicates the ring offset of the descriptor for which the packet data transfer is completed by the receive DMA. Hence software must interpret the value read from the DMA_CHj_Rx_Data_Xfer_Ring_Offset status register as one less (except when the value read is 0 in which case the software will not be able to distinguish whether the current ring offset is 0 or 1), unless the receive DMA is suspended due to the unavailability of the descriptors or is in the stopped state (the RDFS, RDTS, and RDWS fields of the DMA_CHj_Debug status register indicate the STOP or SUSPEND state).*

14.4.3.2.8 DMA Descriptor Write-Back Operation

The DMA descriptor write-back operation is triggered when the descriptor is passed on to it by the Data Transfer Engine after the corresponding buffer contents are transferred. This engine closes the descriptor with OWN bit = 0 to indicate to the software that the descriptor is freed-up.

For TxDMA, the Descriptor write-back operation clears TDES3 while the other TDES0-TDES2 are not modified. This write-back request is issued independently of the Buffer data transfer requests (pointed by the next descriptor) by the TxDMA engine.

For RxDMA, the Descriptor is closed with the status fields also updated in RDES0-RDES3 fields. To ensure that the buffer transfer to system memory is complete before the descriptor is closed, these transfer operations are performed in sequence with the same AXI-ID. In other words, the RxDMA engine starts performing the next descriptor's buffer transfer only after the current descriptor is written back and closed. This sequencing is done to ensure that the software does not process the descriptor and read the buffer contents before the buffer data transfers are completed by the hardware.

The transfer complete Interrupts TI (bit[0]) in case of TxDMA and RI (bit[6]) in the case of RxDMA is generated when the descriptor corresponding to the last segment of the packet data is closed and also has the Interrupt On Completion bit set. Note that the interrupt is generated only after the AXI response to the descriptor write transfer is received from the AXI slave.

If the interrupt generation is not enabled for that packet, the descriptor write operation is deemed to be completed once the descriptor write request is accepted by the AXI slave (without waiting for the response from the slave). This reduces the latency of descriptor write completion and improves the throughput.

14.4.3.2.9 DMA Error Handling

Tx DMA Bus Error Handling Flow

If AXI Bus Error is detected by XGMAC while fetching the Tx Descriptors from system memory, it stops fetching descriptors and ignores all the descriptors it would have fetched as a part of burst. However, it processes all the normal pre-fetched descriptors (if any) in the descriptor Cache memory. While processing the descriptor where

the Bus Error is detected, the packet transfer is terminated forcibly along with a control to tag, which enables MAC transmitter to automatically insert CRC error for that packet. The XGMAC completes pending outstanding read requests on the AXI of that Tx Channel. Also, it makes an attempt to close the descriptor with the Error Status (bit[27] of TDES3) and then automatically STOPs the corresponding Tx DMA Channel.

If AXI Bus Error is detected by the XGMAC while transferring Tx packet data from system memory, it stops fetching descriptors, stops packet data reads and ignores all the descriptors it would have fetched as a part of burst. The packet transfer is also terminated forcibly along with a control to tag which enables MAC transmitter to automatically insert CRC error for that packet. It completes pending outstanding read requests on the AXI of that Tx Channel. Also, XGMAC makes an attempt to close the descriptor with the Error Status (bit[27] of TDES3) and then automatically STOPs the corresponding Tx DMA Channel.

If AXI Bus Error is detected by the XGMAC while writing back the Descriptor to the System memory, it stops fetching descriptors, stops packet data reads and ignores all the descriptors it would have fetched as a part of burst. The packet transfer is also terminated forcibly along with a control to tag which enables MAC transmitter to automatically insert CRC error for that packet. It completes pending outstanding read requests on the AXI of that Tx Channel. Then it STOPs the corresponding Tx DMA Channel. In this case the Error Status (bit[27] of TDES3) write may not be successful as Bus Error is detected while writing back the descriptor itself.

When a Tx DMA channel is stopped by the XGMAC due to Bus Error, the software driver is expected to follow the steps as described in Bus Error Handling.

Rx DMA Bus Error Handling Flow

If AXI Bus Error is detected by the XGMAC while fetching the Rx Descriptors from system memory, it stops fetching descriptors and ignores all the descriptors it would have fetched as a part of burst. However, it processes all the normal pre-fetched descriptors (if any) in the descriptor cache memory. While processing the descriptor where the Bus Error is detected, the packet transfer to system memory is terminated forcibly by flushing the current packet in the Rx Queue. The XGMAC completes any pending outstanding read/write requests on the AXI of the Rx channel. Also, XGMAC attempts to close the descriptor with the Error Status (bit[15] =1 and bit[19:16] = 4’h1000 of RDES3) and then automatically STOPs the corresponding Rx DMA channel.

If AXI Bus Error is detected by the XGMAC while transferring Rx packet data to the system memory, XGMAC stops fetching descriptors, stops packet data writes and also ignores all the descriptors it would have fetched as a part of burst. The packet transfer to system memory is terminated forcibly by flushing the current packet in the Rx Queue. The XGMAC completes any pending outstanding read/write requests on the AXI of that Rx Channel. Also, it makes an attempt to close the descriptor with the Error Status (bit[15] =1 and bit[19:16] = 4’h1000 of RDES3) and then automatically STOPs the corresponding Rx DMA Channel.

If AXI Bus Error is detected by the XGMAC while writing back the Descriptor to the system memory, XGMAC stops fetching descriptors, stops packet data writes and ignores all the descriptors it would have fetched as a part of burst. The packet transfer to system memory is terminated forcibly by flushing the current packet in the Rx Queue. The XGMAC completes pending outstanding read/write requests on the AXI of that Rx Channel. Then it STOPs the corresponding Rx DMA Channel. In this case the Error Status (bit[15] =1 and bit[19:16] = 4’h1000 RDES3) write may not be successful as Bus Error is detected while writing back the descriptor itself.

When a Rx DMA channel is stopped by the XGMAC due to Bus Error, the software driver is expected to follow the steps as described in Bus Error Handling The software driver can optionally program the bit[31] (RPF) of DMA_CHx_Rx_Control Register to prevent any Head of the Line (HOL) blocking of the other DMA Channels which are not stopped. When this bit is set and corresponding DMA is stopped (either by software or XGMAC) the Rx packets routed to that Rx channel is automatically flushed in the Rx Queues. When the DMA is started by the software driver later, the new packets that would have received in the XGMAC after the DMA is started are routed to the Rx DMA Channel. For more information see, Receive Path.

***Notes**:*

1. *When RPF=1, dynamic mapping causes flushing of packets from unintended queues to the stopped Rx DMA*
2. *When RPF=0, when one of the Rx DMA channels is stopped, it causes unintended stopping of Rx DMA channels*

14.4.3.2.10 DMA related Interrupts

An interrupt can be generated as a result of various events in the XGMAC. These events are captured in status registers and interrupt enables are provided for each source of interrupt such that the interrupt signal (sbd_intr_o) is asserted for an event only when the corresponding interrupt enable is set.

The interrupt status and corresponding enable registers are organized in an hierarchical manner so that it is easier for software to traverse and identify the source of interrupt event quickly. When sbd_intr_o is asserted, the DMA_Interrupt_Status register is first level that indicates the major blocks for the interrupt event source. This register is read-only, and it contains bits corresponding to each DMA channel (TX & RX pair), the MTL, and the MAC. The software application must then read one (or more) of the following registers corresponding to the bits that are set:

- DMA_CH(#i)_Status register
- MTL_Interrupt_Status register
- MAC_Interrupt_Status register

The sbd_intr_o interrupt is a level signal and gets de-asserted only when all the enabled interrupt events are cleared in their respective status registers and correspondingly all the bits in the DMA_Interrupt_Status register are cleared.

The DMA_CH(#i)_Status register captures all the interrupt events of that TxDMA and RxDMA channel pair. The DMA_CH(#i)_Interrupt_Enable register contains the corresponding enable bits for each of the interrupt event. There are two groups of interrupts in the DMA channel namely: Normal and Abnormal interrupts. They are indicated by Bits[15:14] of DMA_CH(#i)_Status register respectively. The normal group is for events that happen during the normal transfer of packets (TI, RI, TBU) while the abnormal interrupt events are for error events. Interrupt events are cleared by writing 1'b1 to the corresponding bit position. When all the enabled interrupt events are cleared (including the NIS and AIS), the interrupt source from the DMA Channel is cleared and the corresponding bit in DMA_Interrupt_Status register is also cleared.

Interrupts are not queued. If the same interrupt event occurs again before the driver responds to the previous one, no additional interrupts are generated. For example, Receive Interrupt Bit[6] of DMA_CH(#i)_Status register indicates that one or more packets were transferred to the application buffer. The driver must scan all descriptors, from the last recorded position to the first one, owned by the DMA to determine how many packets are received.

An interrupt is generated only once for multiple events. The driver must scan the DMA_Interrupt_Status register for the cause of the interrupt and clear the source in the respective Status register. The sbd_intr_o is cleared only when all the bits of DMA_Interrupt_Status register are cleared.

Periodic scheduling of transmit and receive interrupt

It is not preferable to generate interrupts for every packet transferred by DMA (RI and TI) for system throughput performance reasons. The XGMAC gives the flexibility to schedule the interrupt at regular intervals using two methods:

1. Set Interrupt on Completion bit in Transmit descriptor (TDES2[31]) once for every "required" number of packets to be transmitted
2. Similarly, set the IOC (RDES3[30] in Table 758 bit only at some specific intervals of Receive descriptors. This way, whenever a received packet transfer to system memory is complete and any of the descriptors used for that packet transfer has the IOC bit set, only then the RI event is generated

In addition, an interrupt timer (DMA_CH(#i)_Rx_Interrupt_Watchdog_Timer) is given for flexible control and periodic scheduling of Receive Interrupt as described:

1. Rx Interrupt Watchdog Timer based on number of application clock cycles: When the RWT field of DMA_CH(#i)_Rx_Interrupt_Watchdog_Timer is programmed with a non-zero value, interrupt timer gets activated as soon as the Rx DMA completes a transfer of a received packet to system memory without asserting the Receive Interrupt because the corresponding interrupt of completion IOC bit (RDES3[30]) is not set. When this timer runs out as per the programmed value, RI bit is set and the interrupt is asserted if the corresponding RIE is enabled in DMA_CH(#i)_Interrupt_Enable register. The 2-bit RWTU field in the same register selects the granularity of RWT (8-bit) field. The granularity can be programmed to 256, 512, 1024, or 2048 clock cycles, giving a maximum range of 522240 ($= 255 \times 2048$) application clock cycles
2. Rx Interrupt Watchdog Timer based on number of transferred bytes: When RBCT field is programmed with a non-zero value, the interrupt timer accumulates the packet size in bytes (extracted from the received packet status), beginning with the packet for which the RxDMA does not assert the Receive Interrupt because the corresponding interrupt of completion IOC bit (RDES3[30]) is not set. When the accumulated bytes equals or exceeds the programmed value, RI bit is set and the interrupt is asserted if the corresponding RIE is enabled in DMA_CH(#i)_Interrupt_Enable register. The active range of the RBCT field in DMA_CH(#i)_Rx_Interrupt_Watchdog_Timer register is 1 to 1023. The granularity is 1 KB, which gives a maximum threshold of 1023 KB for the interrupt timer

Both the interrupt timers are stopped and cleared, if the RI is set for a packet transfer whose descriptor's IOC was set or due to expiry of any one of the interrupt timer. The timer is reactivated automatically after the next packet transfer is complete for which the RI event is not generated.

***Note**: When DSPW bit in the DMA_Mode register is set,the generation of the Rx interrupt based on IOC in the Rx Descriptoris not reliable. Therefore, if the DSPW bit set, IOC bit shouldnot be enabled in the Rx Descriptor.*

Per channel transfer complete interrupt

The Transmit Transfer complete interrupt (TI) and Receive Transfer complete interrupt (RI) are reflected in DMA_CH(#i)_Status register. The TI bit is set whenever the Tx DMA channel closes a descriptor in which the IOC (Interrupt On Completion - TDES2[30]) bit is set. Similarly, the RI bit is set whenever the Rx DMA channel closes the descriptor with LD bit set and in any of the descriptors used for transferring that packet, IOC (Interrupt Enable on completion - RDES3[30]) bit is set.

The common sbd_intr_o output signal is asserted for the Transfer complete interrupts only when the corresponding interrupts are enabled in DMA_CH(#i)_Interrupt_Enable register.

The XGMAC also supports the following per Channel Transfer Complete interrupt signals.

- sbd_perch_tx_intr_o[] (Transmit Per Channel Interrupts)
- sbd_perch_rx_intr_o[] (Receive Per Channel Interrupts)

The behavior of the RI/TI/ sbd_perch_tx_intr_o[]/sbd_perch_rx_intr_o[] changes depending on the settings of INTM field bit[13:12] in DMA_Mode Register.

14.4.3.3 MAC transaction layer (MTL)

The MAC Transaction Layer provides FIFO memory to buffer and regulate the frames between the application system memory and the XGMAC IP. The native interface enables data transfer between the host and the MAC transaction layer through a simple handshake protocol.

The MTL block also provides a reliable synchronization mechanism for data transfer between the application and XGMAC clock domains. The MTL has an asynchronous FIFO on the transmit and receive paths, which are 68 bits wide in this 64-bit system. It contains 64 bits of data and 4 control bits.

The MTL block communicates with the application through the Application Transmit Interface (ATI), Application Receive Interface (ARI), and XGMAC Control Interface (MCI).

14.4.3.3.1 Transmit Path

The application SW pushes the Ethernet packets into the corresponding queue via TX DMA channels. The packet is then popped out and transferred to the MAC when the queue threshold is reached (threshold mode) or complete packet is in the queue (store-and-forward mode).

14.4.3.3.2 Receive Path

The Rx module receives the packets given by the MAC and pushes them into the Rx Queue. The status (fill level, through PBL and watermark) of the queue is indicated to the DMA when it crosses the configured Receive threshold (Bits[1:0] of MTL_RxQ(#i)_Operation_Mode register), in threshold mode or when a complete packet is received in store and forward mode. The DMA can use the fill level of the queue to initiate pre-configured burst transfers towards the AXI interface.

14.4.3.4 The MAC layer

The MAC layer complies fully with the IEEE 802.3-2008/industry standard, implementing the XGMII/GMII/MII/ RGMII/RMII full duplex interface for communication with the physical coding sub-layer. The MAC layer communicates with the application side through the MAC Transmit Interface (MTI), MAC Receive Interface (MRI) and the MAC Control Interface (MCI).

PHY interface selection

The XGMAC-CORE supports the following PHY interfaces. You can select from the following supported modes, by programing the MAC_Tx_Configuration register as shown in Table 683.

Table 683 PHY interfaces supported by XGMAC

Speed	Interface	Data width	Interface clock frequency	Programming MAC_Tx_Configuration.SS
5 Gbps	XGMII	32-bit	156.25 MHz	3'b101
2.5 Gbps	XGMII	32-bit	78.125 MHz	3'b110
1 Gbps	GMII	8-bit	125 MHz	3'b011
2.5 Gbps	GMII	8-bit	312.5 MHz	3'b010
100 Mbps	MII	4-bit	25 MHz	3'b100
10 Mbps	MII	4-bit	2.5 MHz	3'b111

- RGMII interface is selected by configuring RGMII mode in the module HSPHY before a software reset of the MAC or a kernel reset of GETH. Write HSPHY register CTRL0_PHY.EPR = 3'b001. After that, a software reset of the MAC or a kernel reset of GETH needs to be applied. RGMII speed modes (2.5 G/1 G/100 M/10 M) is based on SS[2:0] values of 3’b010, 3'b011, 3'b100, and 3'b111 respectively
- RMII interface is selected by configuring RMII mode in the module HSPHY before a software reset of the MAC or a kernel reset of GETH. Write HSPHY register CTRL0_PHY.EPR = 3'b010. After that, a software reset of the MAC or a kernel reset of GETH needs to be applied. RMII speed modes (100 M/10 M) is based on ss[2:0] values 3’b100 and 3’b111

14.4.3.4.1 Transmission

Transmission is initiated when the MTL or application pushes data towards XGMAC with the SOP signal asserted. When the SOP signal is detected, XGMAC accepts the data and begins transmitting to the XGMII/ GMII/MII/RGMII/RMII. The time required to transmit the frame data to the XGMII/GMII/MII/RGMII/RMII after the application initiates transmission is variable, depending on such delay factors as inter-frame gap (IFG) delay, pause transmission requests from the XGMAC receiver, and the time required to transmit preamble and SFD and

any back-off delays for half-duplex mode. As long as XGMAC can accommodate the data from the application in the internal buffer, XGMAC accepts data received from the MTL.

After the end-of-frame is transferred to the XGMAC-CORE, the application can wait until the controller completes normal transmission, or start transferring the next frame to the XGMAC, then receive the transmission status.

In the half-duplex mode, if a normal collision occurs during transmission, the MAC sends the Transmit Status to MTL, with the retry bit set to 1. The MAC sends retry request, until one of the following condition is true:

- Packet is successfully transmitted
- Maximum number of retry requests are completed

When maximum number of retry requests are completed, the MAC aborts the packet transmission with Excessive Collision Transmit Status. The MAC accepts and drops all further data until the next SOP is received. The MTL block should retransmit the same packet from SOP, when it observes a Retry request (in the Status) from the MAC. If any one of the following occurs, the MAC aborts the packet transmission:

- No carrier (half-duplex mode)
- Loss of carrier (half-duplex mode)
- Excessive deferral (half-duplex mode)
- Late collisions (half-duplex mode)
- Jabber

The MAC accepts and drops all further data until the next SOP is received.

If the MTL cannot provide the data continuously during the transmission, XGMAC issues an underflow status. During the normal transfer of a frame from MTL, if the XGMAC receives an SOF without receiving an EOF for the previous frame, the SOF is ignored and the new frame is considered a continuation of the previous frame.

Related information

Transmit bus interface module (TBU) on page 1414
Transmit frame controller module (TFC) on page 1415
Transmit protocol engine module (TPE) on page 1416
Transmit scheduler module (STX) on page 1417
Transmit CRC generator module (CTX) on page 1419
Transmit flow control module (FTX) on page 1420

Transmit bus interface module (TBU)

This section describes the transmit bus interface module (TBU).

The TBU adds a VLAN tag and/or SA bytes, replaces the VLAN tag, or deletes VLAN tag from the input frame data. The XGMAC provides for VLAN tag inclusion/deletion or replacement and source address inclusion or replacement. Both the VLAN tagging and untagging feature, and the source address inclusion or replacement feature can be controlled through CSR.

VLAN Tag Modification

VLAN tag inclusion

The TBU module adds the VLAN tag to the transmitted packet. It uses the VLT bits of the VLAN Tag Inclusion/Replacement (MAC_VLAN_Incl) register to add the VLAN tag value. The VLAN tag can be included on a per-frame basis. MAC_VLAN_Incl register also provides options for inclusion, deletion, or replacement of VLAN tag. The application can program MAC_VLAN_Incl register to include the VLAN tag for all the transmit frames.

VLAN tag replacement

The TBU module replaces the transmitted packet's existing VLAN tag with the contents of MAC_VLAN_Incl register. The VLAN tag can be replaced on a per-frame basis.

VLAN tag deletion

The TBU module also deletes the VLAN tag for each transmitted frame. MAC_VLAN_Incl register also provides options for VLAN tag inclusion/deletion or replacement. The application can program MAC_VLAN_Incl register to remove the VLAN tag for all transmit frames.

Source Address Modification

SA inclusion

The TBU module adds the source address to the packet transferred from the application, using the contents of MAC_Address0_High and MAC_Address0_Low registers to add the SA to the SA field. If the frame transferred to XGMAC has FCS, it is replaced with an accurate FCS value. The source address inclusion request can be triggered on a per-frame basis by the source address inclusion/replacement control bits in the MAC_Tx_Configuration register.

SA replacement

The TBU module replaces the source address of the transmitted packet with the contents of MAC_Address0_High and MAC_Address0_Low registers to the SA field. The source address replacement request is serviced in the same way as an SA inclusion.

Transmit frame controller module (TFC)

This section describes the transmit frame controller module (TFC).

The Transmit Frame Controller (TFC) consists of a two-stage register, which holds the data and byte enables received from the TBU module. This register regulates data flow to TPE by providing a buffer between the application and the TPE. The TFC also requires this buffer for adding CRC and pad bytes to the frame data input the TBU module sends to the TPE (Transmit Protocol Engine).

The TBU module performs the following types of CRC and Pad insertion related operations:

CRC and pad insertion

The MAC appends a cyclic redundancy check (CRC) to the end of the transmitted packet of length greater than or equal to 60 bytes. When the number of bytes received from the application is less than 60 (DA + SA + LT + Data < 60), the MAC automatically appends zeros to make the packet length exactly 60 bytes and 4 bytes of CRC.

CRC insertion (disable pad insertion)

The MAC appends cyclic redundancy check (CRC) to the end of the transmitted packet, but does not append padding. The application should ensure that the padding bytes are present in the packet being transferred, that is, that the packet being transferred is of length greater than or equal to 60 bytes.

Disable CRC insertion

The MAC does not append a cyclic redundancy check (CRC) to the end of the transmitted packet. The application should ensure that the padding and CRC bytes are present in the packet being transferred.

CRC replacement

The MAC replaces the last four bytes of the transmitted packet with recalculated CRC bytes. The application should ensure that the padding and CRC bytes are present in the packet being transferred.

The application must select the correct type of CRC and Pad insertion operation based on the transmit packet modifications expected/enabled in the MAC.

The TFC handles the XGMAC transmit pause operation because of reception of PAUSE frames. The TFC gives higher priority to transmit a PAUSE frame request from FTX block than the request received from TBU block.

Transmit protocol engine module (TPE)

This section describes the transmit protocol engine module (TPE).

The Transmit Protocol Engine module (TPE) consists of a transmit state machine that controls Ethernet frame transmission. In XGMII and GMII/MII mode, the transmit state machine of this module contains the following features to conform to IEEE 802.3/802.3z specifications.

- Generates preamble and SFD
- Generates jam pattern after normal collision, in the half-duplex mode
- Supports jabber timeout
- Supports flow control for the half-duplex mode (backpressure)
- Generates transmit packet status
- Contains timestamp snapshot logic for IEEE 1588 support

When the TPC module requests the TPE module for a new packet transmission, the Transmit State Machine sends out the preamble and SFD, followed by the data received. The preamble is defined as 7 bytes of 8’b10101010 pattern and the SFD is defined as 1 byte of 8b’10101011 pattern. The collision window is defined as 1 slot time (512-bit times for 10/100 Mbps Ethernet). The jam pattern generation is applicable only to half-duplex mode, not to full duplex mode. In full-duplex mode, the Transmit State Machine ignores the collision signal from the PHY.

In MII mode, if a collision occurs any time from the beginning of the packet to the end of the CRC field, the Transmit State Machine sends a 32-bit jam pattern of 32’h55555555 on MII to inform all other stations that a collision has occurred. If the collision is seen during the preamble transmission phase, the Transmit State Machine completes the transmission of preamble and SFD, and then sends the jam pattern. If the collision occurs after the collision window and before the end of the FCS field the Transmit State Machine sends a 32-bit jam pattern and sets the late collision bit in the transmit packet status.

The TPE module maintains a jabber timer to stop the transmission of Ethernet packets if the TFC module transfers more than 2,048 (default) bytes. The timeout is changed to 10,240 bytes when the Jumbo packet is enabled. The Transmit State Machine uses the deferral mechanism for flow control (backpressure) in the half-duplex mode. When the application requests to stop receiving packets, the Transmit State Machine sends a JAM pattern of (8'h55) 32 bytes when it senses a reception of a packet, provided the Transmit flow control is enabled. This results in a collision and the remote station backs off. The application requests the flow control by setting the FCB bit of the Flow Control Register of the corresponding MTL queue. If the application requests a packet to be transmitted, it is scheduled and transmitted even when the backpressure is activated. If the backpressure continues to be active for a long time (and more than 16 consecutive collision events occur), the remote stations abort their transmissions because of excessive collisions.

If IEEE 1588 timestamp is enabled for the Transmit packet, this block takes a snapshot of the system time when the SFD is put onto the Transmit GMII or MII bus. The system time source is either an external input or generated internally based on the configuration selected.

XGMII mode

The TPE module’s transmit state machine performs the following functions to conform to Clause 46 of IEEE 802.3-2008:

- Generates Start Control Character (SCC), preamble, and SFD
- Generates transmit frame status

- Blocks frame data from the TFC when the STX blocks transmission due to an IFG delay
- Based on the link fault status from the FSM in the XGMAC receiver, aborts frame transmission and transmits the remote fault messages/idles during the IFG

When a new frame transmission from the TFC is requested, the transmit state machine sends the SCC, preamble, and SFD, followed by the received data. The SCC is defined as 1 byte of the 8'b11011111 pattern, the preamble is defined as 7 bytes of the 8'b10101010 pattern, and the SFD is defined as 1 byte of the 8b'10101011 pattern. The Transmit Control signal (TXC[3:0]) is generated synchronously with the TXD[31:0] from this block. This includes the control character during frame transmission and fault messaging.

GMII/MII mode

The TPE module's transmit state machine performs the following functions to conform to Clause 35 of IEEE 802.3-2008:

- Generates preamble and SFD
- Generates transmit frame status
- Blocks frame data from the TFC when the STX blocks transmission due to IFG delay
- Converts the 32-bit data from the TFC to 8-bit or 4-bit data (depending on GMII or MII mode) that can be sent on the GMII data bus (TXD[7:0])

When a new frame transmission from the TFC is requested, the transmit state machine sends the preamble and SFD, followed by the received data. The preamble is defined as 7 bytes of the 8'b10101010 pattern and the SFD is defined as 1 byte of the 8b'10101011 pattern. The TPE state machine transmits a single byte every clock over TXD[7:0]. The transmit control signals TX_EN and TX_ER are generated synchronously with the TXD[7:0] from this block. If an underflow occurs (no valid data available for transmission even though end-of-frame data is not received), the state machine ends the frame transmission on GMII by asserting the TXER signal on the last byte.

Transmit scheduler module (STX)

This section describes the transmit scheduler module (STX).

The Transmit scheduler module (STX) schedules frame transmission on the GMII or MII interface. It provides an enable signal to the TPE module after satisfying the IPG and back-off delays. The STX module performs the following functions:

- Maintains the inter-packet-gap (IPG) between two transmitted packets:

  When operating in XGMII mode, this module maintains an average inter-frame gap (as controlled by the DDIC bit of MAC_Tx_configuration register) between two transmitted frames using the Deficit Idle Counter (DIC) algorithm. When operating in GMII mode, this module ensures that there is a minimum inter-frame gap between two transmit frames (controlled by the IFG bits of MAC_Tx_Configuration register)

  In the half-duplex mode, when IPG is configured for 96-bit times, the STX module follows the rule of deference specified in the IEEE802.3,Section 4.2.3.2.1. The module resets its IPG counter if a carrier is detected during the first two thirds (64-bit times for all IPG values) of the IPG interval. If the carrier is detected during the final third of the IPG interval, the STX module continues the IPG count and enables the transmitter after the IPG interval
- Implements the Truncated Binary Exponential Back-off algorithm:

  The STX module implements the Truncated Binary Exponential Back-off algorithm when it operates in the half-duplex mode
- In XGMII mode, this module can reduce the data transmission rate to SONET/SDH rates, using a rate adaptation algorithm that stretches the IFG. The rate adaptation can be enabled through MAC_Tx_Configuration register
- This module also ensures that the transmitter defers transmission (of data frames) when a PAUSE frame with non-zero pause quanta is received

After satisfying IFG requirements, the STX enables the TPE to transmit the frame.

In XGMII mode, the STX module maintains an idle period according to the DIC Algorithm and performs IFG stretching for rate adaptation if the SONET/SDH rate match bit in MAC_Tx_Configuration register is enabled. If frames from the TFC arrive at the TPE module sooner than the calculated IFG time, the TPE waits for the enable signal from the STX before starting the transmission on the XGMII. The STX starts its IFG counter after the EOF, followed by the IDLE, is transmitted on the line. At the end of the IFG count (when the count is zero) the module issues an enable signal to the TPE.

In GMII mode, the STX module maintains an idle period of the configured inter-frame gap (IFG bits of MAC_Tx_Configuration register) between any two transmitted frames. If frames from the TFC arrive at the TPE module sooner than the configured IFG time, the TPE module waits for the Enable signal from the STX module before starting transmission on the GMII. The STX module starts its IFG counter as soon as the carrier signal of the GMII goes inactive. At the end of the programmed IFG value, the module issues an Enable signal to the TPE.

Link fault signaling

The link fault detection and signaling feature of XGMAC is fully compliant to the Link Fault Signaling FSM described in Section 46.3.4.3 of IEEE 802.3. XGMAC receiver performs link fault detection and indicates the link status to XGMAC transmitter. When the XGMAC receiver detects a local fault during frame reception, XGMAC aborts the frame transmission with an error control character and instead sends Remote Fault messages continuously on the XGMII lanes.

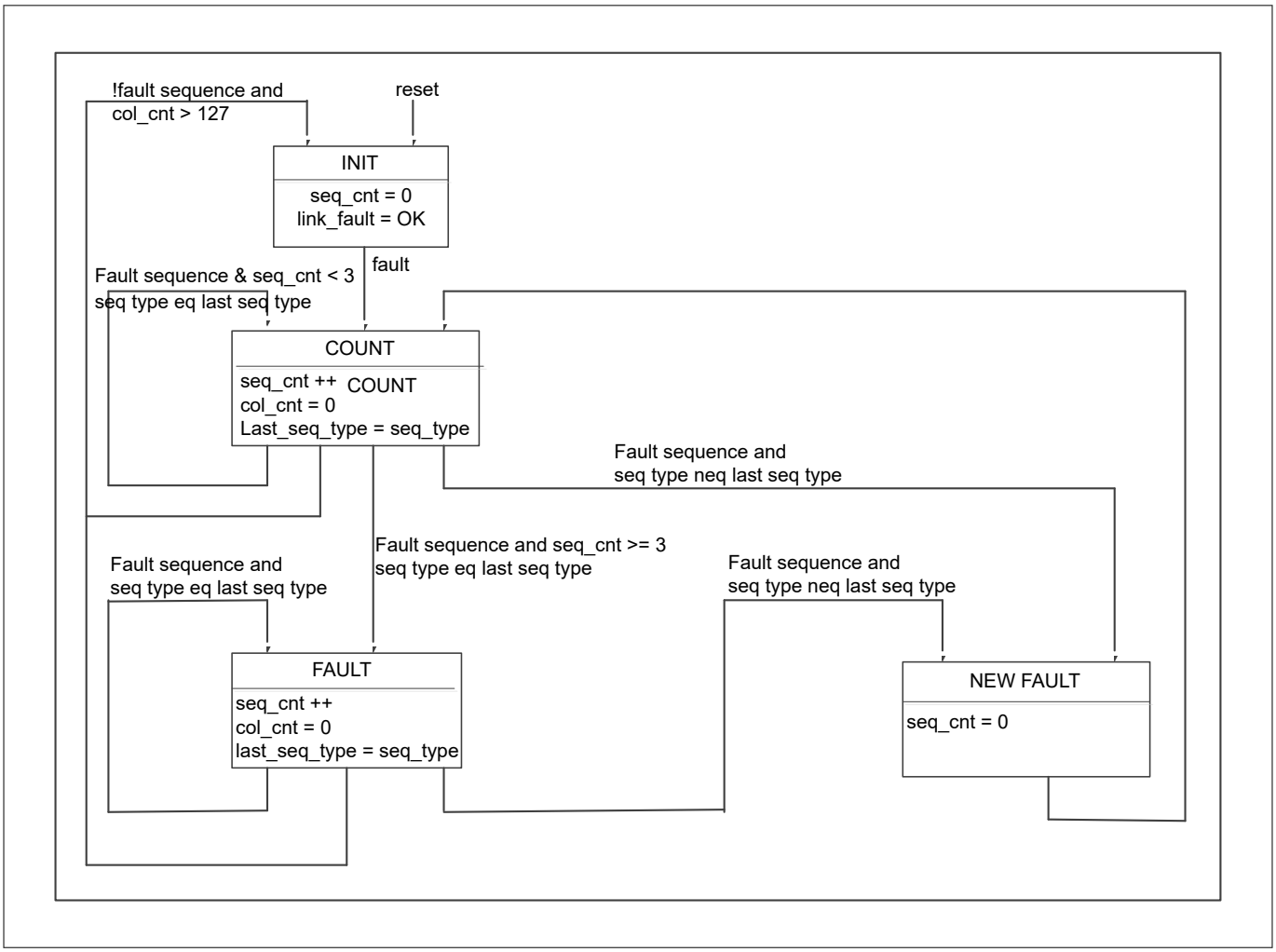


Figure 132 Link fault signaling

When it detects Remote Fault messages on the receive side, XGMAC starts sending Idle control characters continuously on the transmit XGMII lines. If a remote fault is detected during a normal frame transmission, the frame is aborted by sending an Error control character. No data is accepted from the application until the link status is OK.

***Note**:* *Detection of local or remote fault events are validated with the reception of 4 fault_sequences containing the same fault value, with each pair of fault sequences separated by less than 128 columns and no intervening fault_sequences of a different fault value as described in IEEE 802.3 Section 4, Chapter 46.3.4.3.*

XGMAC only supports sequence order set (0x9c) for the remote and local fault detection on the XGMII. If XGMII receives signal order set (0x5c), it is ignored. Section 4 of IEEE 802.3-2008 specification mentions this as RESERVED for INCITS T11 Fiber Channel use

Transmit CRC generator module (CTX)

This section describes the transmit CRC generator module (CTX).

The Transmit CRC Generator module interfaces with the TFC module to calculate CRC for insertion into the Ethernet frame's FCS field. The figure below explains how CRC is transmitted.

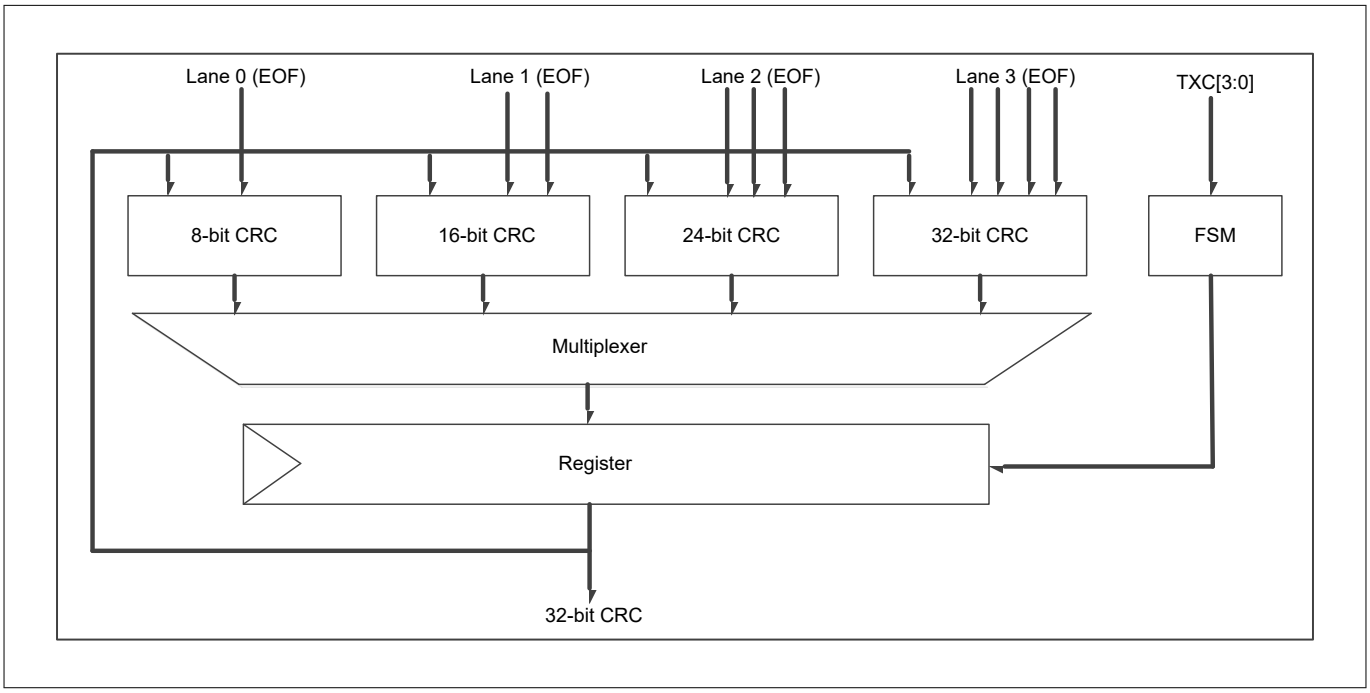


Figure 133 CRC calculation and transmission

The CTX calculates the 32-bit CRC for the Ethernet frame's FCS field. The encoding is defined by the following generating polynomial.

G (x) = x32 + x26 + x23 + x22 + x16 + x12 + x11 + x10 + x8 + x7 + x5 + x4 + x2 + x + 1

This module gets the Ethernet frame as 32-bit words from the TFC module which includes (DA + SA + LT + Data + PAD) qualified with a Data Valid signal. The TFC module issues the Start command to reset the CRC before starting the CRC calculation for the new frame.

TXC[3:0] (Transmit Control)indicates whether data or control characters are being driven Transmit Data TXD lanes. If any of these bits are de-asserted, data is driven to the corresponding Transmit data lane. If a bit remains asserted, control characters are driven to the corresponding Transmit data lane

Transmit flow control module (FTX)

This section describes the transmit flow control module (FTX).

The Transmit Flow Control module (FTX) generates PAUSE frames and drives them to the TFC module. The TFC receives PAUSE frames from the FTX, appends the calculated CRC, and sends the frames to the TPE.

The application initiates PAUSE frame generation by requesting the FTX module to send a PAUSE frame in one of the following ways:

- Setting the Flow Control Busy (FCB) bit in the Flow Control register
- Asserting the mti_flowctrl_i signal in response to an MTL Receive FIFO Full condition or due to the assertion of the sbd_flowctrl_i sideband signal in the XGMAC Subsystem

If the application requests flow control by setting the FCB bit of Flow Control register, the FTX generates and transmits a PAUSE frame to the TFC. The generated frame contains the Pause Time value programmed in MAC_Q(#i)_Tx_Flow_Ctrl register. To extend or end the pause prior to the time specified in the previously transmitted PAUSE frame, the application must request another PAUSE frame transmission after programming the appropriate value in the Pause Time register.

If the request for PAUSE goes inactive prior to the sampling time, the FTX transmits a PAUSE frame with 0 pause time to indicate the other end (a switch, for example) that the Receive buffer is ready to receive new data frames. Zero-quanta PAUSE frame generation is controlled using the Flow Control register's DZQP bit (bit[7] of Flow Control register).

14.4.3.4.2 Reception

When MAC detects an SFD on the XGMII/GMII, a receive operation is initiated. The controller strips the preamble and SFD before processing the frame.

The frame is dropped in the following scenarios:

- The Start control character occurs in any lane other than Lane 0
- The SFD and preamble are less than 8
- The SFD is not received on Lane 3 after the start control character

***Note**:*

- *TPRE field of the `MAC_Extended_Configuration` register is not supported for preemptive packets*
- *In configurations with Frame Preemption (FPE), for packet reception in XGMII mode, the minimum gap between the last valid byte of previous and first valid byte after SFD of current packet is 4 cycles of the receiver clock (IPG columns + Preamble columns)*

However, when operating in the 2.5G speed in XGMII mode, if the TPRE field of `MAC_Extended_Configuration` register is set, packets with SCC and 5 or 6 preamble bytes are accepted.

When operating in GMII mode, the MAC receiver can receive a frame with a variable preamble and does not drop the frame if preamble bytes are missing.

The header fields are checked for filtering and the FCS field is used to verify the frame's CRC. The received frame is stored in a shallow buffer until the address filtering is performed. XGMAC drops the frame if it fails the address filter.

Related information

Transmission on page 1413
Receive protocol engine module (RPE) on page 1421
Receive CRC module (CRX) on page 1422
Receive frame controller module (RFC) on page 1423
Receive flow control module (FRX) on page 1423

Receive bus interface unit module (RBU) on page 1424
Address filtering module (AFM) on page 1424
Packet filtering on page 1432

Receive protocol engine module (RPE)

The RPE consists of the Receive state machines (one for XGMII and another for GMII), which strip the preamble and SFD. At any time only one FSM is enabled, depending on the configuration programmed by software through CSR (the SS field of MAC_TX_Configuration register).

XGMII mode

This section describes the XGMII mode.

In XGMII mode, the RPE's XGMII receive state machine seeks the Start control character on Lane 0, then seeks the SFD field in the XGMII lanes. Until the SFD field is detected, the state machine drops receive preambles. Once the SFD is detected on Lane 3, the state machine begins sending the Ethernet frame's data to the RFC, beginning with the first byte after the SFD (that is, the destination address). The RPE FSM stops sending data to the RFC when any control character other than the Error control character is received in any lane.

The RPE module detects the local and remote fault messages received on the XGMII lines, implements the Link fault signaling protocol, and sends requests to the XGMAC transmitter to send Remote Fault or Idle messages. It also detects invalid control characters (except the Terminate control character) received on any lane during frame reception and ensures that the frame checksum error status is set. The figure in the "Link fault signaling" section shows how the XGMAC monitors the Link Fault message from the PHY or remote device(s) to determine the link fault status. The link fault status from the RPE determines whether the MAC transmitter transmits a Normal Frame message or a Fault message.

Related information

Link fault signaling on page 1418

GMII/MII mode

This section describes the GMII/MII mode.

In GMII mode, when the RX_DV signal of the GMII/XGMII becomes active, the receive state machine of the RPE seeks the SFD field from the receive modifier logic. Until the SFD field is detected, the state machine drops receive preambles. When the SFD is detected, the state machine sends the Ethernet frame data to the RFC module, beginning with the first byte following the SFD (destination address).

Auto pad/CRC stripping

The RPE module's receive state machine decodes the Length/Type field of the Ethernet frame being received. If the Length/Type field is less than 64 bytes and the MAC is programmed with the Auto Pad Stripping or CRC Stripping options, the state machine sends the frame's data up to the count specified in the Length/Type field, then drops bytes (including the FCS field). If the received Length/Type field is greater than or equal to 64 bytes, the state machine performs no pad stripping, but does strip the FCS. The state machine sends the RFC the appropriate byte enables, along with the last word transferred after receiving the Terminate control character. The state machine enters the status phase, where it sends the received frame status. If the Length/Type field is greater than 600 (hexadecimal), the RPE module sends all received Ethernet frame data to the RFC module, irrespective of the value on the programmed auto-pad or CRC stripping option.

Watchdog timer

The XGMAC has an internal watchdog timer, which is programmed to truncate the receive frame. This timer is enabled by default; therefore, frames bigger than 2,048 bytes (10,240 bytes if Jumbo Frame is enabled) are cut off after 2,048 bytes (10,240 bytes) are received at the RPE. This feature can be disabled by programming the watchdog timer disable in the MAC_Rx_Configuration register. However, even if the watchdog timer is disabled, larger-than-16383 bytes frames are cut off and a watchdog time-out status is given. There is a programmable watchdog timer option provided, where cut-off limit can be specified in steps of 1 KB from 2 KB to 16383 bytes.

Loopback support

The XGMAC supports the loopback of transmitted frames onto its receiver in XGMII mode. By default, the XGMAC loopback function is disabled, but this feature can be enabled by programming the Loopback bit in the MAC_Rx_Configuration register.

Enable the loopback only in the full-duplex mode. In the half-duplex mode, the carrier sense signal (csr) or collision (col) signal inputs are sampled which might result in packet drop.

The transmit and receive clocks are asynchronous, so a free-running, asynchronous FIFO is used to loop transmit data back onto the receive path.

In GMII mode, the same FIFO is used for loopback (5 deep, 36 bits wide). The write and read pointers are re-initialized to have an offset of 2 at the start of each frame read from the FIFO. This helps avoid over- or underflow during the transfer of a frame, and ensures that the over- or underflow occurs only during the IFG between the frames.

A FIFO depth of 5 is sufficient to prevent data corruption for frame sizes up to 9,022 bytes with a difference of 200 ppm between the GMII transmit and receive clock frequencies. Hence, larger frames should not be looped back in GMII mode, as they may become corrupted in this loopback FIFO.

***Note**: The XGMAC uses the XGMII receive clock in the receive path during loopback operation.*

Frame status generation

At the end of every received frame, the RPE module generates a received frame status and sends it to the RFC module. Control, missed frame, and filter fail statuses are added to the receive status in the RFC module.

Receive CRC module (CRX)

This section describes the receive CRC module (CRX).

The Receive CRC (CRX) interfaces to the RPE to check for CRC errors in the frame being received. The upper 4 bits of VLAN tag CRC are also used for VLAN-based hash filtering of VLAN frames.

This module calculates the 32-bit CRC for the received frame, which includes the Destination Address field through the FCS field. The following polynomial is used to calculate the CRC.

G (x) = x32 + x26 + x23 + x22 + x16 + x12 + x11 + x10 + x8 + x7 + x5 + x4 + x2 + x + 1

The module takes its data from the RPE (that is, DA + SA + LT + Data + PAD + FCS). The RPE also sends a control signal that indicates the validity of the data. The CRC calculation architecture is same as that of the Transmit CRC described in the "Transmit CRC generator module (CTX)" section. Irrespective of whether Auto Pad or CRC stripping is invoked, the CRX module receives the entire frame to compute the CRC check for a received frame.

Related information

Transmit CRC generator module (CTX) on page 1419

Receive frame controller module (RFC)

This section describes the receive frame controller module (RFC).

The Receive Frame Controller (RFC) receives Ethernet frame data and status from the RPE. The RFC consists of a five-deep FIFO, to delay the data until the filter status is received from the Address Filter block. XGMAC supports the following address filtering options:

- Perfect match destination address filtering for all frames
- Perfect source address filtering for all frames
- VLAN tag hash-based frame filtering for VLAN frames
- VLAN-based perfect filtering of VLAN frames
- Filtering of PAUSE control frames based on the pass control frame setting in the MAC_Packet_Filter register

If the RA bit of the MAC_Packet_Filter register is set, the RFC does not wait for the address filter result and transfers all received frames to the RBU. At the end of data transfer, the RFC sends the received frame status, which includes the Frame Filter bits (SA Filter Fail, DA Filter Fail, VLAN Filter Fail) and the status from the RFC module. These bits are generated based on the AFM filter-fail signals. The RFC does not drop frames in this mode.

If the RA bit is reset, the RFC module waits up to five clocks for the packet filtering result from the AFM module. If it detects a filter-fail from AFM or a runt packet is received (of size <16 bytes), the RFC drops the frame and does not transfer it to the application.

If the filter response from the AFM is delayed more than 5 clocks, the RFC proceeds with frame transfer to the RBU. It still accepts the delayed response from AFM on a DA or SA filter failure and VLAN filter failure, but does not drop the rest of the frame. The Rx status word indicates the filter-fail status. If there is no response from the AFM until the end-of-frame is transmitted, the Rx status word is updated as Filter Fail.

Receive flow control module (FRX)

This section describes the receive flow control module (FRX).

The Receive Flow Controller (FRX) detects the receiving PAUSE frame and passes the pause time received with the frame to the XGMAC transmitter. PAUSE frame detection can be enabled or disabled with the Receive Flow Control Enable (RFE) bit of the MAC_Rx_Flow_Ctrl.

Once receive flow control is enabled, the FRX begins monitoring the received frame destination address for any match with the control frame's multicast address (48'h0180C2000001). If the FRX detects a match, it indicates to the RFC module that the destination address of the received frame matches the reserved control frame destination address. The RFC then determines whether to transfer the received control frame to the application, based on the Pass Control Frames (PCF) bit setting of the MAC_Packet_Filter register.

The FRX also decodes the Type, Opcode, and Pause Timer fields of the control frame being received. At the end of a received frame, the FRX gets the received frame status from the RPE. If the byte count of the status indicates 64 bytes and there is no CRC error, the FRX requests the MAC transmitter to pause data frame transmission for the duration of the decoded pause time value, multiplied by the slot time (64 byte times). Meanwhile, if another PAUSE frame is detected with a zero pause time value, the FRX transfers the pause time value to the transmitter, which exits the pause state immediately. If the received control frame matches neither the Type field (16'h8808), the opcode (16'h00001), nor the byte length (64 bytes), or if there is a CRC error, the FRX does not generate a pause request to the transmitter.

For a PAUSE frame with a unicast destination address, filtering in the FRX module depends on whether the DA matched the contents of MAC Address0 register and the Unicast Detect bit of the MAC_Rx_Flow_Ctrl register is set (detecting a PAUSE frame even with a unicast destination address). The PCF field (MAC_Packet_Filter register) controls the filtering for control frames in addition to the Address filter.

The RFC can also detect the MAC control frames that have opcode fields other than the PAUSE frame Opcode field, which can be called control frames. The status is updated as a Control frame received.

Receive bus interface unit module (RBU)

The RBU interfaces with the application through the MAC receive interface (MRI).

Address filtering module (AFM)

This section describes the address filtering module (AFM).

The Address Filtering module (AFM) checks the destination and/or source address on all received frames and reports the address filtering status to the RFC. It also performs perfect VLAN and VLAN-based hashing for VLAN frames.

Address filtering is based on different parameters programmed by the application. These parameters are input to the AFM as control signals, and the AFM reports the address filtering status based on the combination of these inputs. The AFM does not itself filter the receive frames; rather, it reports the address filtering status (whether or not to drop the frame) to the RFC. The AFM also reports whether the receiving frame is a multicast or broadcast frame. The detailed operation of the packet filter in various is described in chapter "Packet Filtering".

14.4.3.5 Interrupt hierarchy

Interrupts can be generated as a result of various events in the XGMAC. These events are captured in status registers and interrupt enables are provided for each source of interrupt such that the interrupt signal (sbd_intr_o) is asserted for an event only when the corresponding interrupt enable is set.

The MTL_Interrupt_Status register identifies the top level modules that can cause the interrupt to be asserted.

- Bits[7:0] identify eight Queues (Tx or Rx). They are read-only bits and the application should read the corresponding MTL_Q(#i)_Interrupt_Status register to identify the exact cause and set the corresponding bits to 1’b1 to clear the interrupt event. The assertion of sbd_intr_o due to these events is enabled by the corresponding enable bits in MTL_Q(#i)_Interrupt_Enable register
- Bit[16] is made high in case the event is present in the MAC. The application should in turn read the MAC_Interrupt_Status register to identify the cause

Bit[17] is related to indirect access completion events to the TxFIFO and RxFIFO memory in Debug Mode. For more details, see chapter "Memory Access in Debug Mode".

The SRC_GETHNCE (sbd_sfty_ue_intr_o) interrupt is generated when unrecoverable errors are detected and these errors cannot be masked. Appropriate status is indicated in the following registers:

- DMA_AM/DMA_AS_Sfty_ECC_Intr_Sts
- DMA_AM/DMA_AS_Sfty_DPP_Intr_Sts
- DMA/MTL_Safety_Interrupt_Status
- DMA/MAC_DPP_FSM_Interrupt_Status

The SRC_GETHCE (sbd_sfty_ce_intr_o) interrupt is generated when correctable errors are detected. XGMAC sets the respective interrupt enable bits (in the DMA_AM/DMA_AS_Sfty_Intr_En and DMA/MTL_ECC_Interrupt_enable register). To find the root cause of the error, read the DMA_AM/DMA_AS_Sfty_ECC_Intr_Sts, DMA/MTL_Safety_Interrupt_Status and DMA/MTL_ECC_Interrupt_Status registers.

The MAC_Interrupt_Status register bits[1:0] identify the modules that can cause an interrupt from XGMAC. Each event can be prevented from asserting an interrupt on sbd_intr_o by programming the MAC_Interrupt_Enable register.

Some of the interrupt events are enabled (such as, MMC and MDIO) and controlled by the respective Interrupt Enable register inside the module (MMC, MDIO, and so on). The figure below shows the interrupt hierarchy.

In summary, the XGMAC can assert an interrupt on the following events:

- Any change in XGMII link fault status (XGMAC receiver)

- Any event in the SMA (MDIO) function
  - Hot-plugging or removal of PHY port0 to port3 (SMA)
  - Alive status change in any of the MMD devices of PHY port0 to port3 (SMA)
  - Link status change in any of the MMD devices of PHY port0 to port3 (SMA)
  - Completion of a continuous write command (SMA)
  - Completion of a single read/write command (SMA)
- Any RMON counter reaching a value equal to the threshold value programmed in the MMC Control register (MMC Rx/Tx)
- MAC receiving an expected magic packet or remote wake-up packet in the power-down mode
- MAC receiver or transmitter enters or exits the Low-Power IDLE mode when Energy Efficient feature is enabled
- Any event on the timestamp functions
- Any packet error event in the MAC Transmitter or Receiver

***Note**: By default, the MAC interrupt status bits are cleared when the register that contains the source of the interrupt is read. If RCWE bit of the MAC_CSR_SW_Ctrl register is programmed to 1, the MAC interrupt status bits are cleared when the bit that contains the source of the interrupt is explicitly written to 1.*

***Note**: If multiple interrupt events are generated simultaneously and they are grouped to a single interrupt trigger, then only one interrupt trigger event is generated.*

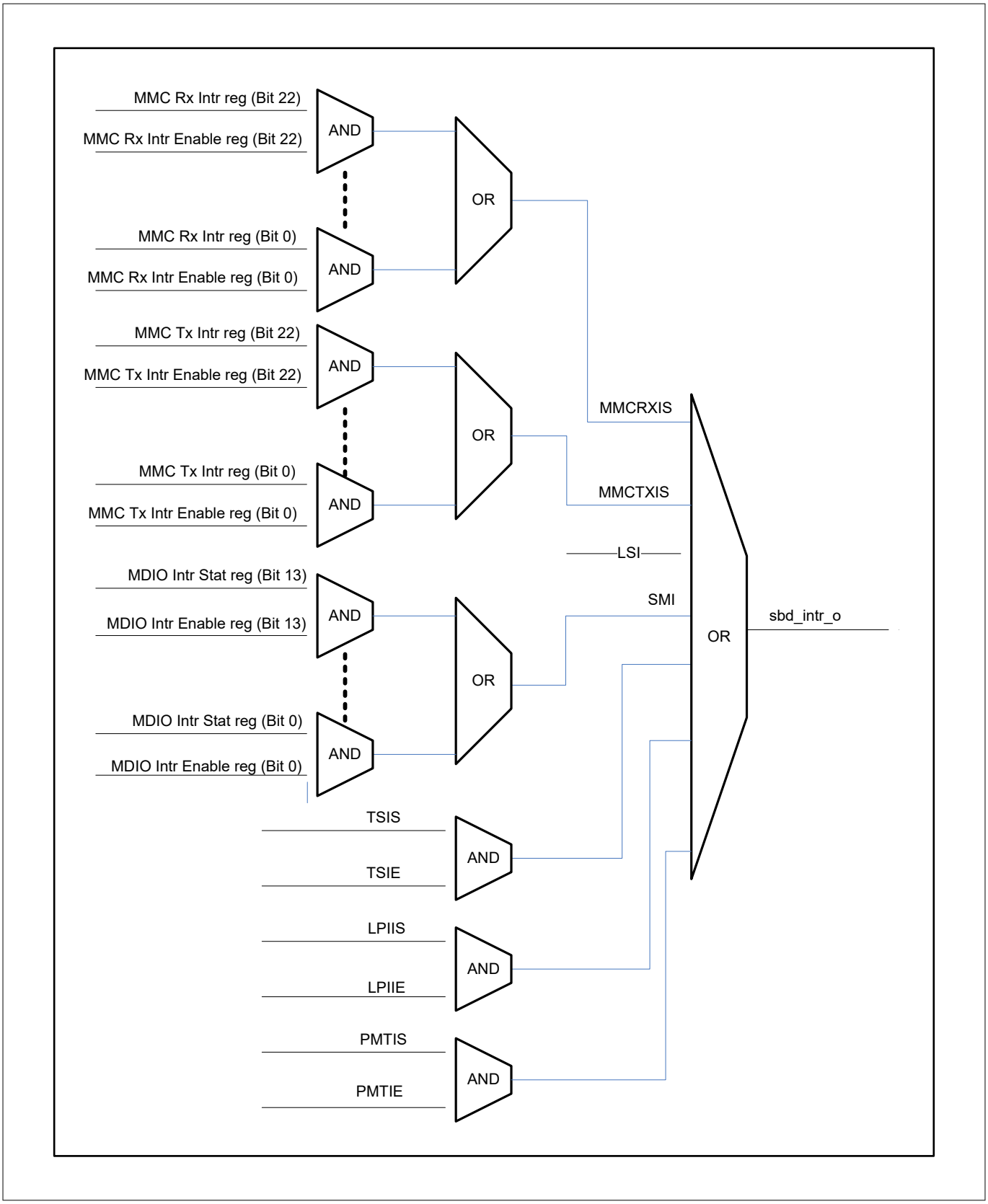


Figure 134 XGMAC interrupt hierarchy

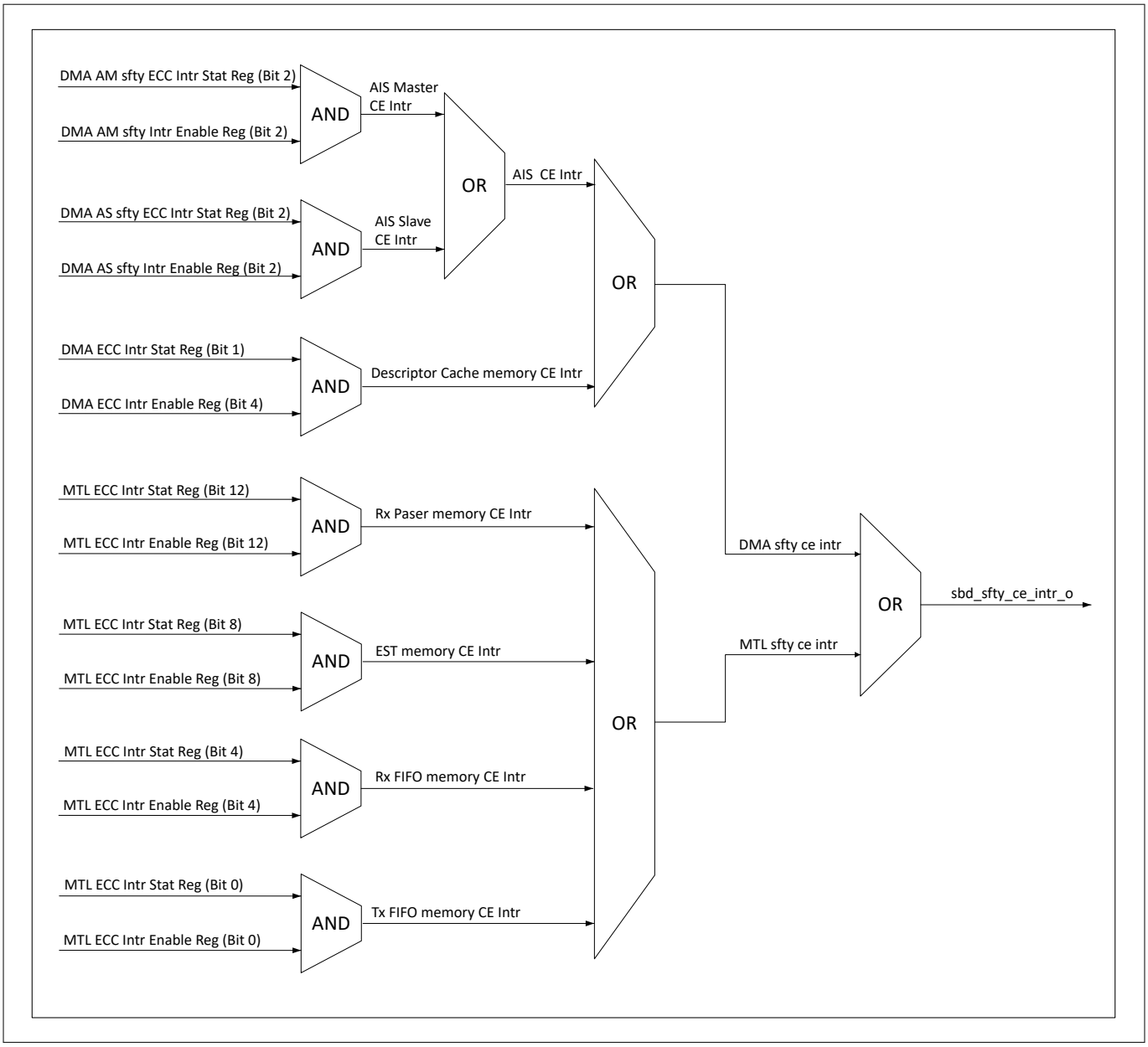


Figure 135 XGMAC interrupt hierarchy with correctable errors

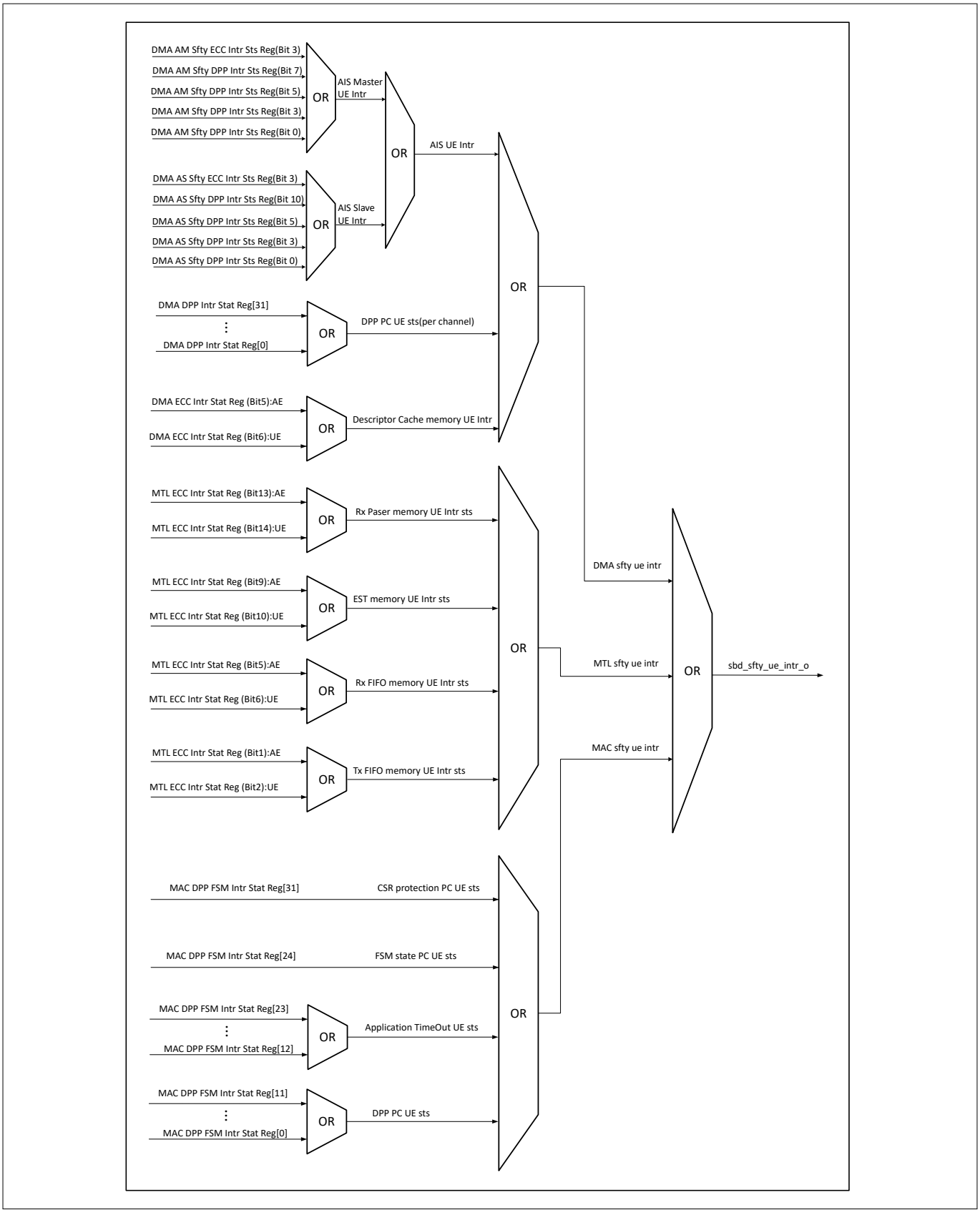


Figure 136 XGMAC interrupt hierarchy with unrecoverable errors

Related information

Memory Access in Debug Mode on page 1472

14.4.3.6 Source Address or VLAN Insertion, Replacement or Deletion

You can enable the Source Address (SA) and VLAN insertion by selecting the Enable SA and VLAN Insertion on Tx option.

The XGMAC supports the following functions for Transmit packets.

Related information

Source Address Insertion or Replacement on page 1429
VLAN Insertion, Replacement, or Deletion on page 1429
Queue/Channel based VLAN Insertion on Tx on page 1430

14.4.3.6.1 Source Address Insertion or Replacement

The application can use the SA insertion or replacement feature to instruct the MAC to do the following for Tx packets:

- Insert the content of the MAC Address Registers in the SA field
- Replace the content of the SA field with the content of the MAC Address registers

When SA insertion is enabled, the application must ensure that the packets sent to the MAC do not have the SA field. The MAC does not check whether the SA field is present in the Transmit packet and it inserts the content of MAC Address registers in the SA field. Similarly, when SA replacement is enabled, the application must ensure that the SA field is present in the packets sent to the MAC. The MAC replaces the six bytes following the Destination Address field in the Transmit packet with the content of the MAC Address registers.

You can enable the SA insertion or replacement feature for all Transmit packets or selective packets:

- Enabling SA insertion or replacement for all packets

  To enable this feature for all packets, program the SARC field of the MAC_Tx_Configuration register.
- Enabling SA insertion or replacement for selective packets
  - Program the SA Insertion Control field (Bits[25:23] of TDES3) in the first Transmit descriptor of the packet. When Bit 25 of TDES3 is set, the SA Insertion Control field indicates insertion or replacement by MAC_Address1 registers. When Bit 25 of TDES3 is reset, it indicates insertion or replacement by MAC_Address0 registers.

If MAC_Address1 registers are not enabled, the MAC_Address0 registers are used for insertion or replacement irrespective of the value of the most-significant bit of the SA Insertion Control field.

14.4.3.6.2 VLAN Insertion, Replacement, or Deletion

The software can use the VLAN insertion, replacement, or deletion feature to instruct the MAC to do the following for Tx packets:

- Delete the VLAN Type and VLAN Tag fields
- Insert or replace the VLAN Type and VLAN Tag fields

  Insertion or replacement is done based on the setting of VLTI bit in the MAC_VLAN_Incl register as described in Table 684 .

Table 684 VLAN Insertion or Replacement Based on VLTI Bit

Condition	Description
VLTI bit is set	The MAC inserts or replaces the following: - VLAN Type field (C-VLAN or S-VLAN as indicated by the CSVL bit of the MAC_VLAN_Incl register) - VLAN Tag field with Content of the VT field of Transmit context descriptor of the packet
VLTI bit is reset	The MAC inserts or replaces the following: - VLAN Type field (C-VLAN or S-VLAN as indicated by the CSVL bit of the MAC_VLAN_Incl register) - VLAN Tag field with the VLT field of the MAC_VLAN_Incl register

When VLAN replacement or deletion is enabled, the MAC checks if the VLAN Type field (0x8100 or 0x88a8) is present after the DA and SA fields in the Transmit packet. The replace or delete operation does not occur if the VLAN Type field is not detected in two bytes following the DA and SA fields. However, when VLAN insertion is enabled, the MAC does not check the presence of VLAN Type field in the Transmit packet and just inserts the VLAN Type and VLAN Tag fields.

You can enable the VLAN insertion, replacement, or deletion feature for all Tx packets or selective packets:

- To enable this feature for all packets, program the VLC field of the MAC_VLAN_Incl register.
- To enable this feature for selective packets,
  - Program the VTIR field of TDES2 Normal Descriptor

14.4.3.6.3 Queue/Channel based VLAN Insertion on Tx

XGMAC supports channel/queue based VLAN tag insertion on all transmitted packets. Queue/Channel specific VLAN tag registers are accessed using indirect addressing through the MAC_VLAN_Incl register. VLAN type and tag value can be independently programmed for each queue/channel.

14.4.3.7 Double VLAN Processing

The XGMAC supports the double VLAN tagging feature in which the MAC can process up to two VLAN tags (inner and outer). When you enable this feature, the XGMAC supports the following:

- Insertion, replacement, or deletion of up to two VLAN tags in the Transmit path
- Packet filtering and stripping based on any one of the two VLAN Tags in the Receive path. Stripping and providing up to two VLAN Tags in the Receive path as a part of the Receive status

Enable this feature by selecting the Enable Double VLAN Processing option.

14.4.3.7.1 Transmit Path

Table 685 describes the features supported by the MAC on the Transmit side.

Table 685 Double VLAN Processing Features in Tx Path

Feature	Description
Support for C-VLAN and S-VLAN Tag types	The inner or outer VLAN tag can be of C-VLAN and S-VLAN type. The VLAN type is specified through the CSVL bit of MAC_VLAN_Incl and registers. The XGMAC supports processing of any sequence of outer and inner VLAN tags. Note: The XGMAC does not support the C-VLAN S-VLAN sequence. The MAC does not check whether the packet provided by the application has a valid sequence of the VLAN Tag types or the insertion or replacement operation results in invalid sequence of VLAN Tag type. Therefore, the application must provide correct sequence of VLAN Tag types and program the MAC in such a way that it results in correct sequence of VLAN Tag types in the transmitted packet. The application must ensure the following: • The inner tag should not be S-VLAN when outer C-VLAN Tag insertion is enabled. • The outer tag should not be C-VLAN when inner S-VLAN Tag insertion is enabled. • The inner tag should not be S-VLAN when outer tag should be replaced with C-VLAN. • The outer tag should not be C-VLAN when inner tag should be replaced with S-VLAN.
VLAN Tag deletion	You can enable the VLAN tag deletion for outer or inner tag through VLC field in the MAC_VLAN_Incl and register, respectively. When VLAN deletion is enabled, the MAC deletes the tag present at the corresponding position. When a packet has only one tag, it is considered as the outer tag. If inner tag deletion is enabled and the packet has only one tag, the MAC does not delete the tag.
VLAN Tag Insertion or Replacement	You can enable the VLAN tag insertion or replacement for outer or inner tag through VLC field in the MAC_VLAN_Incl and register, respectively. When VLAN tag insertion or replacement is enabled, the VLTI bit in the MAC_VLAN_Incl or MAC_Inner_VLAN_Incl register is used to determine whether the VLAN tag should be taken from the register.

14.4.3.7.2 Receive Path

Table 686 describes the features supported by the MAC on the Receive side and the corresponding bits in the MAC_VLAN_Tag register.

Table 686 Double VLAN Processing in Rx Path

Feature	Description
Outer or inner VLAN tag-based filtering	The MAC can filter packets based on the outer or inner VLAN tag through the ERIVLT bit.
C-VLAN or S-VLAN tag-based filtering	The MAC can filter packets based on the C-VLAN or S-VLAN type based on the ERSVLM bit.
Outer and Inner VLAN Tag stripping	The MAC can strip the outer and inner VLAN Tags from received frame based on the EVLS and EIVLS bits.
16-bit outer and inner VLAN Tag and Type in Rx status	The MAC can provide the 16-bit outer and inner VLAN Tag and Type in the Rx status based on the EVLRXS and EIVLRXS bits, respectively.
Disabling or skipping checking of outer VLAN Tag type	The MAC can disable or skip checking of outer VLAN Tag type to match C-VLAN or S-VLAN based on the DOVLTC bit.

14.4.3.8 Packet filtering

The XGMAC supports the following types of filtering for Rx packets:

- Source Address or Destination Address Filtering:

  The Address Filtering Module (AFM) checks the source address and destination address fields of each incoming packet
- Extended VLAN Filtering:

  The XGMAC supports multiple VLAN tag-based and VLAN hash filtering
- Layer 3 and layer 4 match filtering:

  Layer 3 filtering refers to source address and destination address filtering. Layer 4 filtering refers to source port and destination port filtering

The figure below shows the filtering sequence for Rx packets.

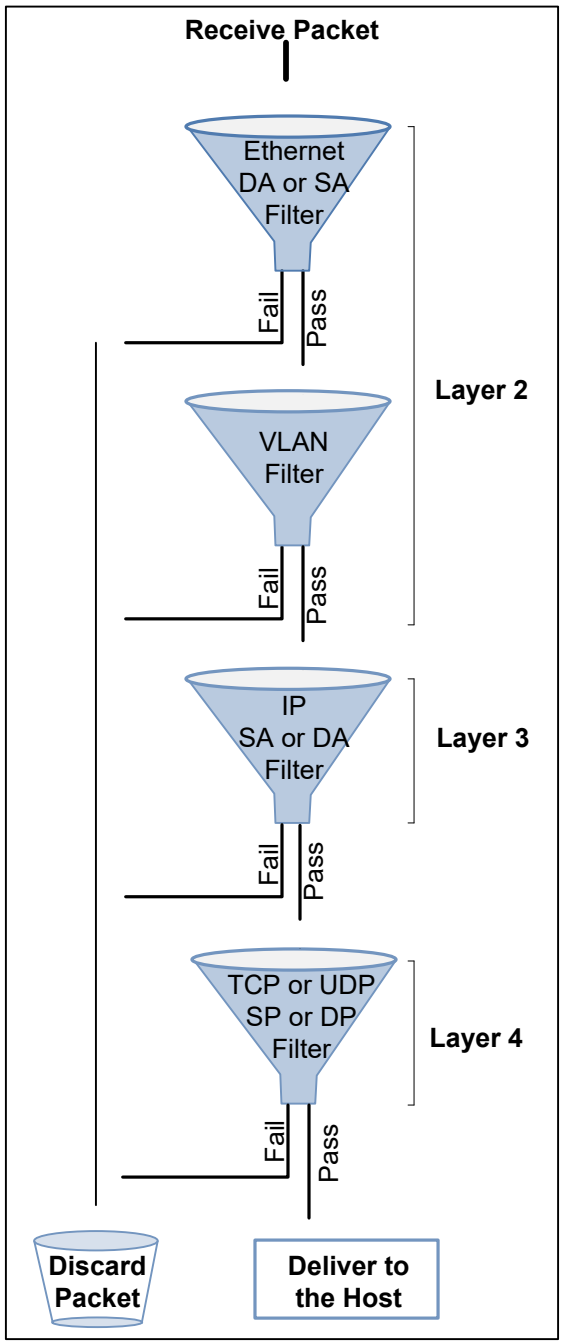


Figure 137 Packet filtering sequence

14.4.3.8.1 Source address or destination address filtering

The Address Filtering Module of the MAC checks the source address (SA) and destination address (DA) fields of each incoming packet.

Unicast destination address filtering

The MAC supports up to 32 MAC addresses for unicast perfect filtering. The MAC compares all 48 bits of received unicast address with the programmed MAC address for any match. The default MacAddr0 is always enabled.

The MacAddr1 to MacAddr31 addresses are selected with an individual enable bit. For MacAddr1 to MacAddr31 addresses, you can mask each byte during comparison with corresponding received DA byte by setting the corresponding Mask Byte Control bit in the register. This enables group address filtering for the DA.

Multicast destination address filtering

To program the MAC to pass all multicast packets, set the PM bit in the MAC_Packet_Filter register. If the PM bit is reset, the multicast address is compared with the programmed MAC Destination Address registers (1-31). Group address filtering is also supported.

In Perfect filtering mode, the multicast address is compared with the programmed MAC Destination Address registers (1-31). Group address filtering is also supported.

Broadcast address filtering

The MAC does not filter any broadcast packets by default. To program the MAC to reject all broadcast packets, set the DBF bit in the MAC_Packet_Filter register.

Unicast source address filtering

The MAC can perform perfect filtering based on the source address field of received packets. By default, the MAC compares the SA field with the values programmed in the SA registers. You can configure the MAC Address registers [1-31] to use SA instead of DA for comparison by setting Bit 30 of the corresponding register.

The MAC also supports group filtering with SA. You can filter a group of addresses by masking one or more bytes of the address. The MAC drops the packets that fail the SA filter if the SAF bit is set in the MAC_Packet_Filter register. Otherwise, the result of the SA filter is given as a status bit in the Receive Status word. When the SAF bit is set, the SA filter and DA filter result is ANDed to decide whether the packet needs to be forwarded. This means that the packet is dropped if either filter fails. The packet is forwarded to the application only if the packet passes both filters in-order. In this case, the DMA channel is selected based on the corresponding DA filter configuration.

Inverse filtering

For DA and SA filtering, you can invert the filter-match result at the final output by setting the DAIF and SAIF bits of the MAC_Packet_Filter register. The DAIF bit is applicable for both unicast and multicast DA packets. The result of the unicast or multicast destination address filter is inverted in this mode. Similarly, when the SAIF bit is set, the result of unicast SA filter is reversed.

The following tables summarize the DA and SA filtering based on the type of packets received.

***Note**: When the RA bit of the MAC_Packet_Filter register is set, all packets are forwarded to the system along with the correct result of the address filtering in the Rx Status.*

Table 687 Destination address filtering

Packet type	PR	DAIF	PM	DBF	DA filter operation
Broadcast	1	X	X	X	Pass
	0	X	X	0	Pass
	0	X	X	1	Fail
Unicast	1	X	X	X	Pass all packets
	0	0	X	X	Pass on Perfect/Group filter match
	0	1	X	X	Fail on Perfect/Group filter match
Multicast	1	X	X	X	Pass all packets
	X	X	1	X	Pass all packets

(table continues...)

Table 687 (continued) Destination address filtering

Packet type	PR	DAIF	PM	DBF	DA filter operation
	0	0	0	X	Pass on Perfect/Group filter match and drop Pause packets if PCF = 0x
	0	1	0	X	Fail on Perfect/Group filter match and drop Pause packets if PCF = 0x

Table 688 Source address filtering

Packet type	PR	SAIF	SAF	SA filter operation
Unicast	1	X	X	Pass all packets
	0	0	0	Pass status on Perfect or Group filter match but do not drop packets that fail
	0	1	0	Fail status on Perfect or Group filter match but do not drop packet
	0	0	1	Pass on Perfect or Group filter match and drop packets that fail
	0	1	1	Fail on Perfect or Group filter match and drop packets that fail

14.4.3.8.2 Extended VLAN filtering

Perfect filtering is done for the VLAN Tag Filters. For each VLAN Tag Filter, the VLAN Tag ID is compared. The combined result of all the enabled VLAN Tag Filters determines the overall VLAN Tag Filter result. The filter result is passed to the application as part of the status bits.

When the VLAN Tag Filter gives a pass result, the frame is considered to have passed the VLAN Tag Filters. If the frame fails all the filters, the frame is considered to have failed the VLAN filter.

In addition to Perfect Filtering, Hash Filtering can also be enabled. In Hash Filtering, the overall VLAN Filter result is determined by the Hash Filter result and the perfect filter result. The overall filter result determines if the frame is dropped or forwarded to the application based on the programming on the VTFE and RA bits of the MAC Filter Register.

Comparison modes of extended receive VLAN filtering

For each VLAN Tag filter, the application has the following comparison modes:

- Program the MAC to compare an outer VLAN Tag or an inner VLAN Tag with the programmed VID
- Choose the width of the VID field, 12 or 16 bits, for the comparison
- Enable or disable type check for each filter. When type check is enabled, the application can choose if the VID comparison should be only for the SVLAN or CVLAN types frames. For example, when a filter is enabled for 16 bit comparison, type is SVLAN, and outer VLAN tag, any single or double VLAN tagged frames with outer SVLAN tags are compared with this filter, and result is obtained

Filtering

When Extended RX VLAN Filtering & Routing feature is enabled, the application can enable both Perfect and Hash Filtering. The overall VLAN Filter Result is based on the perfect filter result and the Hash Filter result (if enabled). The filter result is passed to the application as part of the status bits.

Perfect Filtering is done based on the MAC_VLAN_Tag_Filter registers. For each VLAN Tag Filter, the MAC compares the relevant VLAN Tag ID and gives a result. If any one of the VLAN Tag Filters gives a match then the frame is considered to have passed the VLAN Tag Filters. If the frame mismatches all the filters, then the frame is considered to have failed the VLAN filter. This behavior is applicable only when the Inverse Filtering is not enabled in MAC_VLAN_Tag_Ctrl Register.

If inverse Filtering is enabled and the frame has mis-matched all the relevant filters then it is considered to have passed the VLAN filter. If the frame matches any one of the relevant filters then it is considered as a fail. If none

of the enabled filters can perform a comparison or if none of the filters are enabled, then the frame is bypassed to the application.

The overall filter result and the programming on the VTFE and RA bits of the MAC Filter Register determine if the frame is dropped or forwarded to the application. If RA is set, it does not matter if the filter result is a pass or fail. The frame is always forwarded. If RA is 0 and VTFE is set, then only if the VLAN Tag Filter result is a pass does the MAC forward the frame. If RA is 0 and VTFE is also 0, then the frame is forwarded always irrespective of the filter result. If the frame is forwarded to the application, then the relevant filter result is indicated through the Status bits.

Filter status

Currently the MAC indicates the VLAN Filter Status through one bit in the status - VF in RDES2. When Extended RX VLAN Filtering & Routing is enabled, two status bits are used to indicate the comparison result of VLAN tags. The Outer VLAN Tag Filter Pass and Inner VLAN Tag Filter Pass bits are defined in the following positions in various configurations. The status indicated through these bits is highly dependent on the programming:

In RDES2:

- Bit 15 - Outer VLAN Tag Filter Status
- Bit 13 - Inner VLAN Tag Filter Status

In ARI status: MAC Filter Status

- Bit 15 - Outer VLAN Tag Filter Status
- Bit 13 - Inner VLAN Tag Filter Status

MRI Status:

- Bit 47 - Outer VLAN Tag Filter Status
- Bit 45 - Inner VLAN Tag Filter Status

Outer VLAN Tag Filter Status (OTS)

- In Perfect filtering, without inverse filtering enabled, if this bit is set, it indicates that the frame's Outer VLAN Tag has matched one of the VLAN Tag Filters
- If this bit is reset, it indicates that the frame's Outer VLAN Tag has either failed the relevant Outer VLAN Tag Filters or bypassed them
- If none of the filters are enabled for Outer VLAN Tag Comparison, this bit is reset
- If Inverse Filtering is enabled and this bit is set, then the frame's VLAN Tag has passed all the relevant VLAN Tag Filters. If it is reset, it has failed at least one of or bypassed all the Filters programmed for Outer VLAN Tag Comparison
- This bit is valid for both Single and Double VLAN Tagged frames

Inner VLAN Tag Filter Status (ITS)

- In Perfect Filtering, without inverse filtering enabled, if this bit is set, it indicates that the frame's Inner VLAN Tag has matched one of the VLAN Tag Filters
- If this bit is reset, it indicates that the frame's Inner VLAN Tag has either failed the relevant Inner VLAN Tag Filters or bypassed them
- If none of the filters are enabled for Inner VLAN Tag Comparison, this bit is reset
- If Inverse Filtering is enabled and this bit is set, the frame's VLAN Tag has passed all the relevant VLAN Tag Filters. If it is reset, it has failed at least one of or bypassed all the Filters programmed for Inner VLAN Tag Comparison
- This bit is valid for only Double VLAN Tagged frames when Double VLAN Processing is enabled

The application must look at the status bits and the programming to determine if the Frame has passed or failed the VLAN Filter.

The following two tables show the possible Filter combinations and the corresponding filter results. These tables explain the scenarios when Double VLAN Processing and Hash VLAN filter are enabled in the design.

Legend for the Table:

- VTIM – VLAN Tag Inverse Match Enable – bit 17 in VLAN_Tag_Ctrl Register
- HFO – Hash Filter enabled for Outer VLAN Tag Comparison – bit 25 and bit 27 in VLAN_Tag_Ctrl Register

- HFI – Hash Filter enabled for Inner VLAN Tag Comparison – bit 25 and bit 27 in VLAN_Tag_Ctrl Register
- PFO – Perfect Filter comparison enabled for Outer VLAN Tag - Any of the MAC_VLAN_Tag_Filter Registers is enabled (bit 16 is set) and programmed for Outer VLAN Tag comparison (bit 20 is set to 0)
- PFI – Perfect Filter comparison enabled for Inner VLAN Tag - Any of the MAC_VLAN_Tag_Filter Registers is enabled (bit 16 is set) and programmed for Inner VLAN Tag comparison (bit 20 is set to 1)
- OTS – Outer VLAN Tag Filter Status
- ITS – Inner VLAN Tag Filter Status

The table below shows the possible values of status bits (OTS and ITS) when at least one Perfect filter is enabled.

Table 689 OTS and ITS bit values with at least 1 perfect filter enabled

VTIM	HFO	HFI	PFO	PFI	OTS	ITS
0	0	0	0	1	0	1/0
0	0	0	1	0	1/0	0
0	0	0	1	1	1/0	1/0
0	1	0	1	1	1/0	1/0
0	1	0	1	0	1/0	0
0	1	0	0	1	1/0	1/0
0	0	1	1	1	1/0	1/0
0	0	1	1	0	1/0	1/0
0	0	1	0	1	0	1/0
1	0	0	0	1	0	1/0
1	0	0	1	0	1/0	0
1	0	0	1	1	1/0	1/0
1	1	0	1	1	1/0	1/0
1	1	0	1	0	1/0	0
1	1	0	0	1	1/0	1/0
1	0	1	1	1	1/0	1/0
1	0	1	1	0	1/0	1/0
1	0	1	0	1	0	1/0

The table below shows the possible values of status bits (OTS and ITS) when none of the perfect filters are enabled and only the VLAN Hash Filter is enabled.

Table 690 OTS and ITS bit values with only VLAN hash filter enabled

VTIM	HFO	HFI	OTS	ITS
0	0	0	0	0
0	1	0	1/0	0
0	0	1	0	1/0
1	0	0	0	0
1	1	0	1/0	0
1	0	1	0	1/0

With no perfect filters enabled, any VLAN packet is considered to have bypassed the perfect filter. If Hash Filter is enabled for one of the Tags, then the respective Status bit depends on the Filter's result. The Status bits are set to 0 if VLAN Hash Filter is also not enabled.

If the value of ITS/OTS is shown as 1/0; then it indicates that the final result is dependent on the enabled relevant filter's result.

Example 1: The second row of Table 689 indicates that at least one Perfect Filter is enabled for Outer VLAN tag comparison and none of the filters are enabled for Inner VLAN tag comparison. Inverse VLAN Filtering is not enabled. The bit OTS is given as 1/0. If the received frame passes atleast one of the enabled Outer VLAN Tag filters then the bit is set to 1. If the frame doesn't pass any of the enabled Outer VLAN Tag filters, then the bit is set to 0.

Example 2: Last Row of Table 689 indicates that Inverse Filtering is enabled, Hash Filter and atleast one perfect filter is enabled for Inner VLAN Tag comparison, then if the received frame's Inner VLAN tag mismatches with both the Hash Filter and all the enabled Perfect filters, then the frame has the ITS bit set to 1; else it is set to 0. OTS is set to 0 as no comparison is performed.

Limitations:

The status bits in the descriptor for Extended RX VLAN Filtering & Routing Feature have some restrictions based on other features enabled in the MAC. Certain configurations, the entire VLAN Filter result is provided as VF instead of two separate bits – OTS & ITS. The following points describe the scenarios where a single VLAN Filter status is provided.

- The bits OTS and ITS both are valid
- The IVT and OVT fields of the RDES3 hold the VLAN Tag values if programmed accordingly

When the Status bit 15 of RDES2 is defined as VF instead of OTS; the behavior is as follows:

- When Inverse Filtering is not enabled:
  - The Status Bit is set if any of the VLAN Tag Filter passes
  - If the frame’s VLAN tags do not match any of the relevant VLAN Filters, the VF bit is set to 0. In this case, if VTFE bit of the Packet Filter register is set, the frame is not forwarded to the application
  - If both the VLAN tags bypass the relevant filters, the VF bit is set to 0. In this case, if VTFE bit of the Packet Filter register is set, the frame is forwarded to the application
- When Inverse Filtering is enabled
  - VF bit is set only when both the Tags (if applicable) have mismatched the relevant enabled filters
  - If one of the tags has mismatched the relevant filters and the other tag has bypassed it, the VF bit is set to 1
  - If either of the tags matches any of the relevant filters, the VF bit is set to 0
  - If both the tags have bypassed the relevant filters, the frame is forwarded to the application with the status bit VF set to 0

The VLAN Tag stripping is still dependent on the individual tags match or mismatch. For example, if the MAC has “Strip on Tag Match” enabled for the Outer VLAN Tag; even though the Over All Filter Result as indicated by the Bit VF is a fail (set to 0); this tag is stripped.

Stripping

Each of the VLAN Tags has individual control over stripping. The programming options of Always strip, never strip, strip on pass and strip on fail are available. Inner or Outer VLAN Tag Stripping is based on the individual tags’ pass or fail results. If a tag is bypassed by all the relevant filters then stripping is not applicable for it.

- If strip on Pass is enabled for the outer VLAN Tag, the stripping occurs only if the Outer VLAN tag has passed the relevant Filters. The Outer VLAN Tag Filter Result bit is set
- If strip on Fail is enabled for the outer VLAN Tag, stripping occurs only if the Outer VLAN Tag has failed relevant filters. The Outer VLAN Tag Filter Result Bit is reset
- If the Outer VLAN tag of the received frame is bypassed by the entire filter (no comparison has been made), the tag is not stripped, though the Status Bit is still 0

- As multiple filters are enabled, it is possible that the received VLAN frame could have matched any one or more of the filters. The VLAN Tag's value is not always deterministic from the filter status bits
- So if the application strips the VLAN Tag based on the filter result, it might lose the VID. So the suggested use is: if Stripping is enabled for any of the tags, then the tag can be put in the status. For this, the application must enable the respective "VLAN Tag in Status" bit – 24 or 31 in the MAC VLAN Tag Control Register

14.4.3.8.3 Layer 3 and layer 4 match filtering

The XGMAC supports Layer 3 and Layer 4 based packet filtering. The Layer 3 filtering refers to the IP Source or Destination Address filtering in the IPv4 or IPv6 packets whereas Layer 4 filtering refers to the Source or Destination Port number filtering in TCP or UDP.

For Layer 3 or Layer 4 filtering operation, you must set the IPC bit of the MAC_Rx_Configuration register to enable the Rx Checksum Offload Engine.

When Layer 3 and Layer 4 filtering is enabled, the packets are filtered in the following way:

- Matched Packets: The MAC forwards the packets that match all enabled fields to the application along with the status. The MAC forwards the matched field status only if the IPC bit of the MAC_Rx_Configuration register is set and one of the following conditions is true:
  - All enabled Layer 3 and Layer 4 fields match
  - At least one of the enabled field matches and other fields are bypassed or disabled

  When multiple Layer 3 and Layer 4 filters are enabled, any filter match is considered as a match. If more than one filter matches, the MAC provides the status of the lowest filter where Filter 0 is the lowest filter and Filter 7 is the highest filter. For example, if Filter 0 and Filter 1 match, the MAC gives the status corresponding to filter 0.

***Note**: The source or destination address and VLAN tag filters (if enabled) have precedence over Layer 3 and Layer 4 filter. This means that a packet which fails the source or destination address or VLAN tag filter is dropped irrespective of the Layer 3 and Layer 4 filter results.*

- Unmatched Packets: The MAC drops the packets that do not match any of the enabled fields. The user can use the inverse match feature to block or drop a packet with specific TCP or UDP over IP fields and forward all other packets.

  Aborted or partial packets are dropped in the MTL Rx FIFO. If the Rx FIFO operates in the threshold (cut-through) mode and the threshold is programmed to a small value, such that packet transfer to application starts before the failed Layer 3 and Layer 4 filter results are available, the application may receive a partial packet with appropriate abort status
- Non-TCP or UDP IP packets: By default, all non-TCP or UDP IP packets are bypassed from the Layer 3 and Layer 4 filters. The user can optionally program the MAC to drop all non-TCP or UDP over IP packets

Layer 3 and layer 4 filters register set

The MAC implements a set of indirect registers for Layer 3 and Layer 4 based packet filtering. The indirect register set contains a control register namely, MAC_L3_L4_Control0, to control the packet filtering. In addition, there are five address registers to program the Layer 3 and Layer 4 fields to be matched, such as:

- MAC_Layer4_Address0
- MAC_Layer3_Addr0_Reg0
- MAC_Layer3_Addr1_Reg0
- MAC_Layer3_Addr2_Reg0
- MAC_Layer3_Addr3_Reg0

XGMAC has eight such independent set of registers.

Layer 3 filtering

The XGMAC supports perfect matching or inverse matching for IP Source Address and Destination Address. In addition, you can match the complete IP address or mask the lower bits matching, that is, compare all bits of the address except the specified lower mask bits.

For IPv6 packets filtering, you can the 4 MAC_Layer3_Addr[0:3]_Reg(#i) to contain the 128-bit IP Source Address or IP Destination Address. The IP Source Address or Destination Address should be programmed in the order defined in the IPv6 specification, that is, the first byte of the IP Source Address or Destination Address in the received packet is in the higher byte of the register and the subsequent registers follow the same order.

For IPv4 packet filtering, you can use the MAC_Layer3_Addr0_Reg#(i) and MAC_Layer3_Addr1_Reg#(i) registers to contain the 32-bit IP Source Address and IP Destination Address, respectively. The remaining two registers are reserved. The IP Source Address or Destination Address should be programmed in the order defined in the IPv4 specification, that is, the first byte of IP Source Address and Destination Address in the received packet in the higher byte of the respective register.

Layer 4 filtering

The XGMAC supports perfect matching or inverse matching for TCP or UDP Source and Destination Port numbers. However, you can program only one type (TCP or UDP) at a time. The MAC_Layer4_Address(#i) register contains the 16-bit Source and Destination Port numbers of TCP or UDP, that is, the lower 16 bits for Source Port number and higher 16 bits for Destination Port number.

The TCP or UDP Source and Destination Port numbers should be programmed in the order defined in the TCP or UDP specification, that is, the first byte of TCP or UDP Source and Destination Port number in the received packet is in the higher byte of the register.

14.4.3.9 Flexible Receive Parser

14.4.3.9.1 Overview of Flexible Receive Parser

The XGMAC controller has in-built programmable parser to parse the Ethernet packet based on a software controlled/programmable rule-set. This helps to support filtering and packet steering decision (DMA channel selection) of the received packet, based on any header field of existing protocols or custom/future protocols.

The parser uses a software programmed lookup table that contains instructions and filtering and routing decisions to be taken by the controller on the received packets.

- The maximum length of the packet data that the parser can operate on, is 256B.
- Support for up to 256 instructions, each processing 32-bit frame data with a mask
  - Match and Inverse Match support per instruction
  - Packet data compared with programmed data and mask bits
- Frame filtering based on any of the packet fields
  - Drop/accept packet or bypass packet filtering.
- Determine the DMA channel for packet parsing
- Support for out-of-order packet parsing
  - Decision tree can be out of order; decision tree need not be based on the same order as the packet data arrival.

- Flexible receive parser always operates in receive clock domain
- Flexible receive parser supports stream gate filtering, a subset of IEEE 802.1 Qci specification; stream filter classification and stream gate instance. Stream gate filtering is supported only in
  - Full duplex mode
  - 100 Mbps or higher link speed

14.4.3.9.2 Description of Flexible Receive Parser

This section describes the block diagram and the details of the flexible receive parser.

Flexible Receive Parser Instruction Table (IT) Format

Before enabling the Flexible Receive Parser, the software must build the 128-bit wide instruction table for the parser. Each instruction is programmed as an entry in the instruction memory.

The following figure shows the graphical representation of the instruction table.

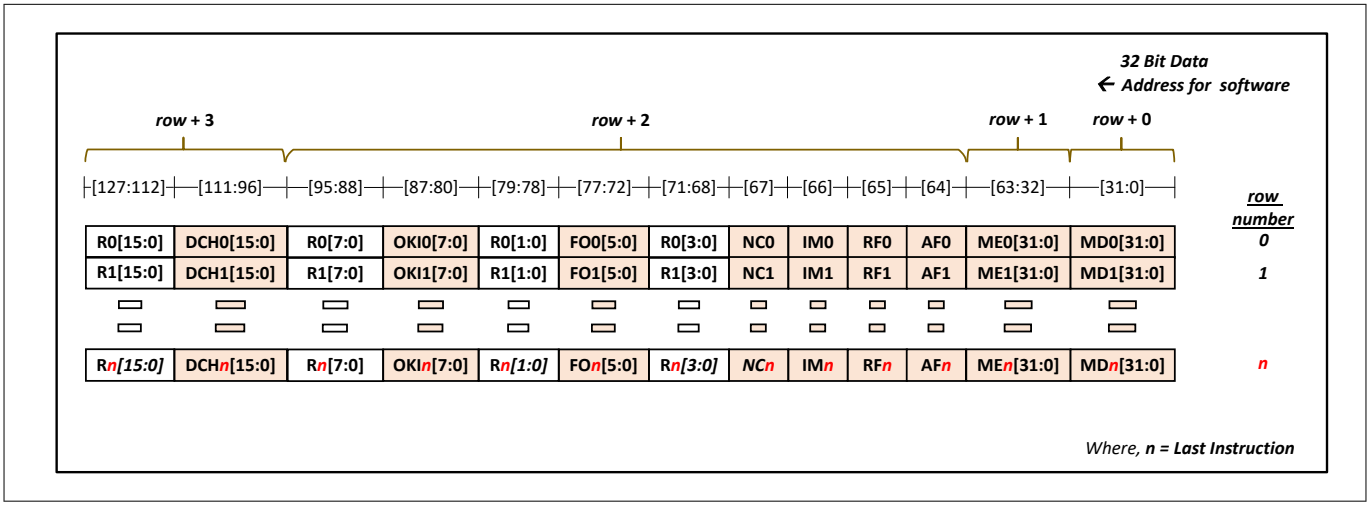


Figure 138 Flexible Receive Parser Instruction Table

The format of each entry in the instruction table is shown below.

Table 691 Flexible Receive Parser Instruction Entry Format

Field	Field Name	Description
31:0	MD	Match Data The 4-byte data. This data is used for comparing with incoming packet data starts at the frame offset as defined in [77:72] of the entry. The comparison is done only on those bits whose corresponding mask bits are set to 1 (ME bits of [63:32]).
63:32	ME	Match Enable When the ME bit is set to 1, the corresponding packet data bit is used for comparing. Otherwise, corresponding data bit is ignored.
64	AF	Accept Frame
65	RF	Reject Frame
66	IM	Inverse Match

(table continues...)

Table 691 (continued) Flexible Receive Parser Instruction Entry Format

Field	Field Name	Description
67	NIC	Next Instruction Control match = MD & ME Bits == (Packet data [frame offset* 8*4]) & ME bits If IM = 0; ENTRY_MATCH = match If IM = 1; ENTRY_MATCH = !match • When ENTRY_MATCH == 1 - if AF == 1 and RF == 0, (PASS result) Accept and route the frame to DMA channel given by the DCH field (Bits[111:96]) of this entry - else if AF == 0 and RF = 1, reject or drop the frame - else if AF == 1 and RF = 1, (Routing BYPASS result) Accept the frame, but ignore DCH but route the frame to DMA Channel number given by MAC - else if AF == 0 and RF == 0, (intermediate result) - If NIC == 1 Continue to next sequential entry in the Instruction Table - else NIC = 0 Jump to next entry given by OKI field (Bits[87:80]) in this entry • When ENTRY_MATCH == 0 (intermediate result) - if NIC == 1 Jump to next entry given by OKI field (Bits[87:80]) in this entry - else NIC == 0 Continue to next sequential entry in the instruction table
71:68	Reserved	Reserved, shall be written as 0 and read as 0.
77:72	FO	Frame Offset [77:72] indicates the frame offset in terms of 4 bytes Frame offset in the packet data to be compared for match, in terms of 4 bytes. • 0 (value): 0 (frame offset in bytes) • 1: 4 • 2: 8 • ... • 63: 252
79:78	Reserved	

(table continues...)

Table 691 (continued) Flexible Receive Parser Instruction Entry Format

Field	Field Name	Description
87:80	OKI/PCV, PCN / GIV,GID	OK Index (OKI) This field gives the next entry in the instruction table, to be processed, OKI field is valid only when, • ENTRY_MATCH = 0 and NIC = 1 • ENTRY_MATCH = 1 and NIC = 0, AF = 0, RF = 0. • OKI is 8 bits wide When you enable the Policing Counter feature, interpret this field as follows: • Bit [80]: PCV (Policing Counter Valid) • Bits [84:81]: PCN (Policing Counter Number) • Bits [87: 85]: Reserved When you do not enable Policing Counter feature, but enable Stream-Gate filtering feature, and when AF = 1 and NIC = 0, interpret this field as follows: • Bit [80]: GIV • Bits [84:81]: GID • Bits [87: 85]: Reserved
88	GIV	Gate-ID Valid When this field is 1, ENTRY_MATCH = 1, and AF = 1, indicates that the packet is further checked against the Stream Gate Filter, given by GID. **Note:** Only when both Stream-Gate Filtering and Policing Counter features are enabled, interpret this field as GIV. Otherwise, this field is reserved.
91:89	GID	Gate-ID This field represents the Stream-Gate filter number to be applied on the packet, when GIV = 1. **Note:** Only when both Stream-Gate Filtering and Policing Counter features are enabled, interpret this field as GID. Otherwise, this field is reserved.
95:92	Reserved	Shall be written and read as 0.
111:96	DCH	DMA Channel Number Indicates the DMA Channel Number when ENTRY_MATCH==1 and AF == 1. The encoding is as follows: • bit[0]: DMA channel number 0 • bit[1]: DMA channel number 1 • bit[2]: DMA channel number 2 • ... • bit[7]: DMA channel number 7 bits[15:8]: Reserved, shall be written as 0. ***Note**: The DMA Channel number is encoded bitwise. XGMAC IP supports multicast/broadcast across DMA channels.*

(table continues...)

Table 691 (continued) Flexible Receive Parser Instruction Entry Format

Field	Field Name	Description
127:112	Reserved	These bits are not implemented. This field is only for software to view.

***Note**:*
- *Instruction Table Width: 128 bits*
- *Instruction Table Depth: 256*
- *Byte Ordering: The first byte received on the line is compared with MD[7:0] of the instruction table.*
- *Entry and instruction are used synonymously.*
- *All the reserved bits are read as zeros.*

***Note**: Error Correction Code (ECC) is used only for security purpose. During programming, ECC is calculated for the whole instruction data (except MSB, reserved bits) as well its address. After programming, XGMAC recomputes the ECC and compares with existing ECC bits. If it finds a mismatch, it generates safety interrupt.*

The parser begins parsing from the 0^{th} entry, for each received packet. The subsequent parsing entry location (next entry or OKI) depends on the parser result of the current entry. The figure below shows the flexible receive parser flow.

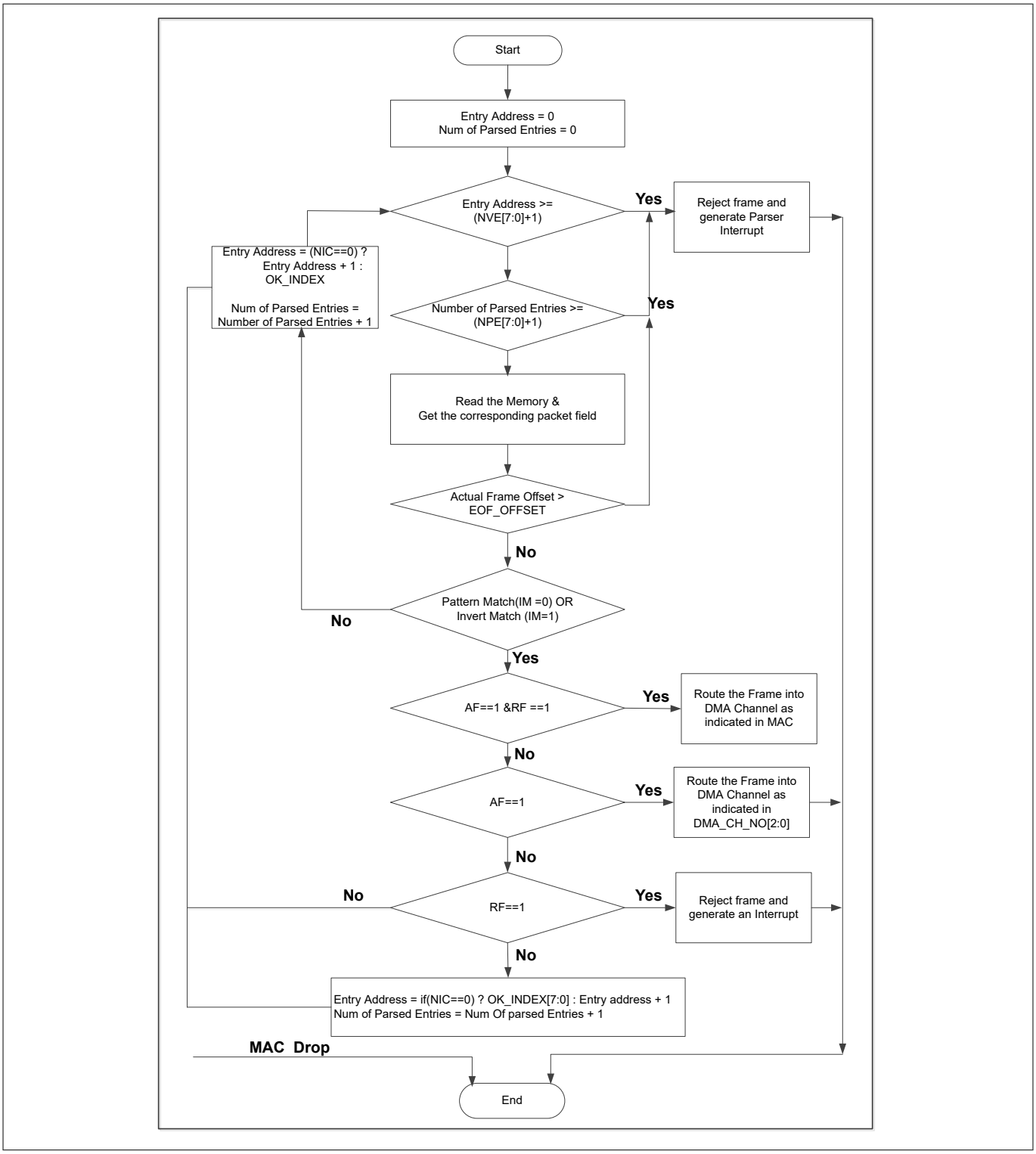


Figure 139 Flexible Receive Parser Flow Chart

Software Access to the Instruction Table

The software can write/read the flexible receive parser instruction table using the following register sets through indirect addressing:

- MTL_RXP_Indirect_Acc_Control_Status
- MTL_RXP_Indirect_Acc_data

***Note**: The indirect registers are shared by multiple indirect accessible blocks. Set the ACCSEL field of the MTL_RXP_Indirect_Acc_Control_Status register to access the Instruction Table.*

It is recommended not to update valid entries in the instruction table when frame reception and frame parsing is in progress. This can cause unintended results.

For more details, see the section "Registers".

***Note**: To program multiple parsing scheme, program them across instruction table and select either of it to run it first by programming the following at instruction 0:*

- *Set all bits to 0 except OKI[]. Set OKI to the first instruction location of the parsing scheme which you want to run first*

It is possible to reprogram the unused instructions when parsing is in progress, to build a new instruction list (or sequence).

Related information

Registers on page 802

Number of Valid Entries in Instruction Table (NVE)

This is the number of valid entries in the Instruction Table starting from the 0th instruction. The default value is set to number of supported instructions minus 1. Software can override the default value by writing into the NVE field of the MTL_RXP_Control_Status register (Number of Valid Entries). Possible values can be from 0 to number of supported instructions minus 1; 0 implies 1st instruction, 1 implies 2nd instruction, and so on.

While parsing, if the instruction address is greater than NVE, an error is flagged in the NVEOVIS field of MTL_RXP_Interrupt_Control_Status register. Also, XGMAC generates an interrupt status, MTLPIS field of the MTL_Interrupt_Status register.

***Note**: NVE Overflow error status is reported as consolidated error status in the receive descriptor (write-back format). For more details, see the "Receive normal descriptor (write-back format)" section.*

Related information

Receive Normal Descriptor (Write-Back Format) on page 1572

Number of Parsable Entries (NPE)

Indicates the number of entries/instructions that can be parsed on an incoming packet. The default value is set to number of supported instructions minus 1. Software can override the default value by writing into MTL_RXP_Control_Status registers field (Number of Parsable Entries). Possible values can be from 0 to number of supported instructions minus 1; 0 implies 1st parsable instruction and so on.

During parsing, if the number of parsed entries is more than NPE+1, an error is flagged in the NPIOVIS field of MTL_RXP_Interrupt_Control_Status register. Also, XGMAC generates an interrupt status in the MTLPIS field of MT_ Interrupt_Status register.

***Note**: NVE indicates the number of valid entries starting from 0th entry of the instruction table built by the software,whereas NPE indicates the worst case number of parsable entries considering the various possible paths that can be taken.*

NVE Overflow error status is reported as consolidated error status in the receive descriptor (write-back format). For more details, see the "Receive normal descriptor (write-back format)" section.

Related information

Receive Normal Descriptor (Write-Back Format) on page 1572

Frame Offset

Indicates the frame offset, in terms of 4 bytes; used to compare respective receive frame offset data with the instruction's match data.

In an instruction, an error is flagged in the MTL_RXP_Interrupt_Control_Status register, FOOVIS (Frame Offset Overflow Interrupt Status) when Frame Offset is more than packet's EOF size or supported number of parsable bytes.

Also, XGMAC generates an interrupt status in the MTLPIS field of MTL_Interrupt_Status register.

***Note**:*

- *If the frame ends after the (frame_offset *4) bytes but before ((frame_offset *4)+3) bytes, the Receive parser declares Frame Offset Error if comparison is enabled (using MASK bits) for the non-received bytes; from (frame_offset *4) to ((frame_offset *4)+3).*
- *Frame Offset Overflow error status is reported as a consolidated error status in the receive descriptor (write-back format). For more details, see the "Receive normal descriptor (write-back format)" section*

Related information

Receive Normal Descriptor (Write-Back Format) on page 1572

VLAN Strip Handling

The MAC supports stripping the outer as well as inner VLAN tags. When MAC is programmed to strip the VLAN tags, the Flexible Receive parser considers the stripped VLAN tags as part of the incoming packet.

PAD Strip/CRC Strip Handling

The pad strip and CRC strip are controlled by the software and applicable to all the received packets. When these features are enabled, software must build the flexible receive parser instructions accordingly.

***Note**:* *It is recommended to enable PAD and CRC stripping.*

Runt Packet Handling

The flexible receive parser is designed for minimum packet size to be of 64B. If the runt packet is received without an additional IPG (at least 64B minus Runt packet size), the ongoing parsing of preceding packet could be impacted.

Out of Order Processing

The flexible receive parser flow can be out of order within the bytes of a received frame. The decision tree need not be based on the order in which packet arrives. This implies that the next entry and next frame offset (FO[]) can be less than the current instruction number and current Frame Offset.

For example, you can look for matching/mismatching IP header and then look for Ethernet header to decide on filtering/channel selection. Also, it can skip from higher address entry to lower address entry in the table.

Ethernet line speed dependency

The flexible receive parser is implemented in the receive (Rx) clock domain.

Example: For 1G Line Speed and 125 MHz receive (Rx) clock, assuming worst case example of back-to-back 64 bytes frame reception.

The processor has 672 ns (including 64B of next frame, 12B of IPG, 8B of Preamble time of next to next frame) to process the current frame.

Number of parser cycles available at 125 MHz: 672 ns/8 ns = 84 cycles.

The flexible receive parser stops the current parsing when it detects SOF of the next packet. To account for the implementation latency, actual parsing cycles are: 84 - 3 = 81.

So, with one Parallel Instruction Processor Engine, the parser can process up to 81 instructions.

Four Parallel Instruction Processor Engines (PIPEs) are implemented, in the worst-case example, theoretically the parser can parse a total of 83 * 4 = 324 instructions. Practically this means, the full parser instruction table with 256 entries can be used in this case.

When the link speed changes, it is assumed that software does not program the number of parsable entries/ instructions beyond the intended limit as per the calculations above. If programmed incorrectly, parser does not parse the packet further and flags a parsing error and is treated as NPE[] overflow.

There are no constraints on the number of JUMPs in the Instruction Table. The architecture does not have additional latency to execute the JUMPs.

Frame Accept

In case of an Entry Match (considering the Inverse match) and Frame Accept bit is set and RF is not set, the frame is accepted and routed to the corresponding instruction's DMA channel number.

Frame Reject

In case of an Entry Match (considering the Inverse match) and Frame Accept bit is not set and Frame Reject bit is set, the frame is dropped.

An interrupt is generated when the frame is dropped due to AF = 0 and RF = 1;
MTL_RXP_Interrupt_Control_Status register bit PDRFIE is set.

XGMAC also generates an interrupt status, MTLPIS bit of the MTL_Interrupt_Status register.

Flexible Receive Parser Bypass Routing

In case of an Entry Match (considering the Inverse match) and AF and RF bits are set, accept the frame, but bypass the DMA channel routing selected by the flexible receive parser. In such cases DMA channel number provided by MAC/MTL, based on one of the filtering and routing mechanisms is selected.

DMA Channel Selection

The MAC/MTL also determines the DMA Channel number based on incoming packet fields.

When the parser is enabled (FRPE bit is set to 1 in MTL_Operation_Mode register) the parser overrides the DMA selection criteria of the MAC.

When the parser is disabled (FRPE is set to 0 in MTL_Operation_Mode register) the MAC/MTL determines the DMA channel number.

Frames that pass the FRP filtering, are routed based on the Instruction Table. All the other frames that meet any of the following criteria, are treated as residual frames.

- Parsing errors such as NVE, NPE, FOO, and so on

- Incomplete parsing
- Any other frames that could not be dropped despite the drop result from FRP. For more details, see the "Receive packet drop indication" section

Residual frames are routed to

- Channel 0, for express traffic
- Channel number configured in the FPE_RCH field of MTL_RXP_Control_Status register, for preemption traffic

***Note**: DMA channel number in the instruction has one-hot encoding. So, the frame is routed to the channel whose respective bit is set.*

Related information

Receive Packet Drop Indication on page 1451

Receive Queue Selection

The receive queue is determined by the existing MAC functionality, irrespective of the enabling/disabling of the flexible receive parser. This is because the flexible receive parser is mainly designed for selecting the DMA channel and Rx queuing is decided by the MAC to enable the proper functioning of the Pause feature.

***Note**: When there are multiple queues with multiple DMA Channels enabled, a specific DMA channel might receive packets out-of-order. This is because, queue selection is based on MAC criteria (VLAN Priority and other criteria) and Rx DMA channel selection is based on flexible receive parser. If in-order packet reception is necessary, you must program Single Queue configuration.*

MAC (L2) Packet Filtering/Drop/Error Handling

The MAC does the packet filtering based on the received packet fields. MAC filtering/L2 filtering can also be enabled with flexible receive parser. But MAC filtering has higher priority over flexible receiver parser.

Flexible receiver parser 's result has impact on RA (Receive All) and L2 Filtering (such as SA, DA, and VLAN).

Table 692 RXP Impact on RA and L2 Filtering

RA (Receive All)	L2 Filter Pass (MAC)	Impact on Flexible Receive Parser (FRP) Result
No	No (L2 Drop)	Bypass FRP result.
No	Yes (L2 Accept)	FRP result is used.
Yes	No (L2 Drop)	Bypass FRP result.
Yes	Yes (L2 Accept)	If FRP Result is Accept Use FRP Result. If the result from flexible receive parser is Reject, bypass the FRP result and route the frame to residual channel based on the traffic. For more details, see the "DMA channel selection" section.

***Note**: When you enable flexible receive parser, it is recommended to use only L2 filtering and routing (DA, SA, and VLAN).*

Related information

DMA Channel Selection on page 1448

Multicast and Broadcast Support

XGMAC supports multicast and broadcast only when DDS (DA duplication select) is enabled and the packet is routed to the highest queue number.

When flexible receive parser is enabled, the packet routing is decided by the DMA Channel Numbers (DCH[]) in the instruction table. In other words, such multicast packets are transferred to all the DMA channels for which the DCH bits are set.

***Note**: When a frame does not fulfill the requirement for broadcast/multicast, Flexible Receive Parser routes the frame to the lowest selected DMA Channel Number in the instruction table.*

Statistical Counters

When Flexible Receive Parser is enabled, following counters are implemented in the indirect IFR register block:

1. MTL_RXP_Drop_Cnt

   Incremented when flexible receive parser drops a packet due to AF = 0 and RF = 1
2. DMA_RXP_Error_Cnt

   Incremented when a RXP encounters error. The error scenarios are:

   - Current Instruction Number > NVE[]
   - Number Parsed Instructions > NPE[] + 1
   - Current Instruction's Frame offset > EOF frame offset or last parsable frame offset
   - Parsing is not complete even after receiving the next to next packet SOF
3. MTL_RXP_Bypass_cnt

   This is incremented when Rx Parser bypasses a packet due to AF = 1 and RF = 1
4. DMA_CH(#i)_RXP_Accept_cnt (i=0-7)

   This is per DMA Channel register. It is incremented when a Rx Parser accept a packet due to AF = 1 and RF = 0.

   For more details, see the "Software access to indirect FRP register (IFR) block" section

Related information

Software Access to Indirect FRP Register (IFR) Block on page 1450

Software Access to Indirect FRP Register (IFR) Block

The software can write/read the flexible receive parser's Indirect Registers using the following register sets through indirect addressing:

- MTL_RXP_Indirect_Acc_Control_Status
- MTL_RXP_Indirect_Acc_data

***Note**: All indirect Registers of flexible receive parser are 32-bit wide.*

The indirect registers are shared by multiple indirect accessible blocks. Set the ACCSEL field of the MTL_RXP_Indirect_Acc_Control_Status register to access the indirect Register block of flexible receive parser.

For more details, see the section "Registers".

Related information

Registers on page 802

Receive Cut-Through Functionality

In the cut-through mode, the Rx controller attempts to transfer the received packets to DMA just after the cut-through threshold amount of packet data is received.

However due to flexible receive parser's result, worst case arrival time can be more than the programmed cut-through threshold. So, the Rx controller delays the packet transfer to DMA accordingly. So, the cut-through functionality is re-defined as, the Rx controller attempts to transfer the packet to DMA after at least RTC number of bytes are received and flexible receive parser results are available.

Receive Packet Drop Indication

In most cases, the packet drops internally in the Rx Queue. However, back-to-back packets can arrive for the same queue and first packet's flexible receive parser result is not available when the second packet arrives. In such cases, the packet cannot be flushed internally and forwarded to DMA or the Application Interface and status is indicated accordingly.

For more details, see the FPS[2:0] bits of RDES2 and RDES3 described in the "Receive normal descriptor (write-back format)" section.

Related information

Receive Normal Descriptor (Write-Back Format) on page 1572

Uncorrectable ECC Error Handling

When the flexible receive parser detects an uncorrectable ECC error while checking the instruction table, software gets the interrupt for unrecoverable error. In such a case, software should re-program the instruction table.

In such a case, FRPE bit is reset and parsing is suspended for all subsequent packets.

14.4.3.9.3 Enabling and disabling the flexible receive parser

To enable the Flexible Receive Parser, set FRPE bit to 1, in the MTL_Operation_Mode register.

It is not required to disable the L2 filtering when flexible receive parser is enabled, though L2 filtering result has an effect on flexible receiver parser result.

After setting FRPE bit to 1, optionally, software can wait until RXPI field of MTL_RXP_Control_Status register becomes 1 (active).

For details about these options, see the "MAC (L2) packet filtering/drop/error handling" section.

Related information

MAC (L2) Packet Filtering/Drop/Error Handling on page 1449

14.4.3.9.4 Registers Related to the Flexible Receive Parser

- MTL_Operation_Mode
- MTL_Interrupt_Status
- MTL_RXP_Control_Status
- MTL_RXP_Interrupt_Control_Status
- MTL_RXP_Indirect_Acc_Control_Status
- MTL_RXP_Indirect_Acc_Data
- MTL_RXP_Drop_Cnt (Indirect Access Register)

- MTL_RXP_Error_Cnt (Indirect Access Register)
- DMA_CH_RXP_Accept_Cnt (Indirect Access Register)

Indirect Access Registers are implemented in the Indirect FRP (IFR) register block. For more details, see the "Software access to indirect FRP register (IFR) block" section.

For more details, see the section "Registers".

Related information

Software Access to Indirect FRP Register (IFR) Block on page 1450

Registers on page 802

14.4.3.10 IEEE 1588 Timestamps

The IEEE 1588 defines a Precision Time Protocol (PTP) which enables precise synchronization of clocks in measurement and control systems implemented with technologies such as network communication, local computing, and distributed objects. The PTP applies to systems communicating by local area networks supporting multicast messaging, including (but not limited to) Ethernet. This protocol enables heterogeneous systems that include clocks of varying inherent precision, resolution, and stability to synchronize. The protocol supports system-wide synchronization accuracy in the sub-microsecond range with minimal network and local clock computing resources.

The XGMAC supports the IEEE 1588-2002 (version 1) and IEEE 1588-2008 (version 2). The IEEE 1588-2002 supports PTP transported over UDP/IP. The IEEE 1588-2008 supports PTP transported over Ethernet. The XGMAC provides programmable support for both standards. It supports the following features:

- Supports both timestamp formats
- Provides an option to take snapshot of all packets or only PTP type packets
- Provides an option to take snapshot of only event messages
- Provides an option to take the snapshot based on the clock type: ordinary, boundary, end-to-end transparent, and peer-to-peer transparent
- Provides an option to select the node to be a master or slave for ordinary and boundary clock
- Identifies the PTP message type, version, and PTP payload in packets sent directly over Ethernet and sends the status
- Provides an option to measure sub-second time in digital or binary format

14.4.3.10.1 Delay Request-Response Mechanism

The system or network is classified into the master and slave nodes for distributing the timing and clock information. Figure 140 shows the process that PTP uses for synchronizing a slave node to a master node by exchanging PTP messages.

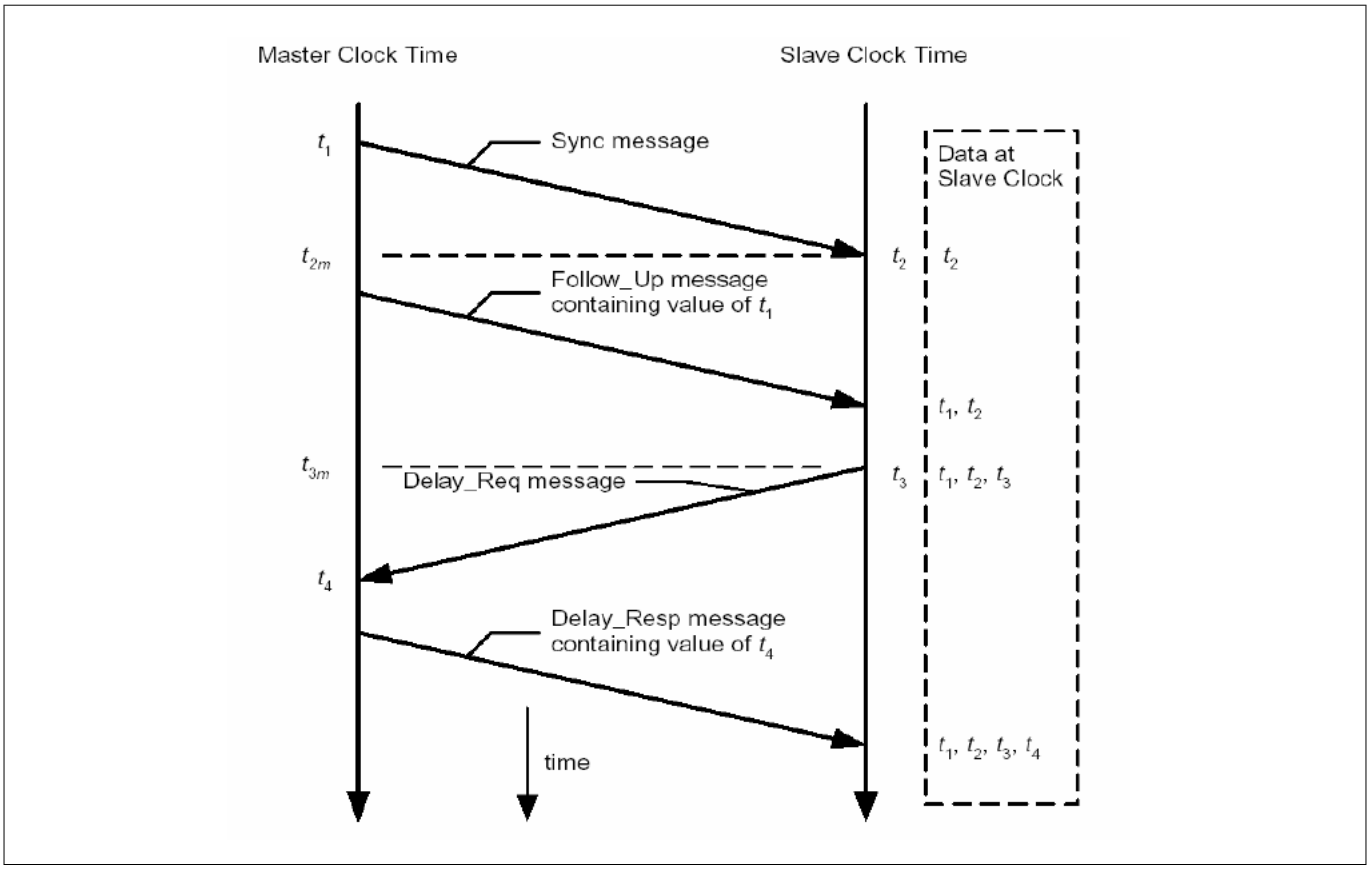


Figure 140 Networked Time Synchronization

As shown in Figure 140, the PTP uses the following process:

1. The master broadcasts the PTP Sync messages to all its nodes. The Sync message contains the reference time information of the master. This message leaves the system of the master at t1. This time must be captured for Ethernet ports at GMII or XGMII.
2. The slave receives the Sync message and also captures the exact time, t2, using its timing reference.
3. The master sends a Follow_up message to the slave, which contains t1 information for later use.
4. The slave sends a Delay_Req message to the master and notes the exact time, t3, at which this packet leaves the GMII or XGMII interface.
5. The master receives the message, capturing the exact time t4, at which the message enters its system.
6. The master sends the t4 information to the slave in the Delay_Resp message.
7. The slave uses the four values of t1, t2, t3, and t4 to synchronize its local timing reference to the timing reference of the master.

Most of the PTP implementation is done in the software that is above the UDP layer. However, the hardware support is required to capture the exact time when specific PTP packets enter or leave the Ethernet port at the GMII or XGMII interface. This timing information must be captured and returned to the software for proper implementation of PTP with high accuracy.

14.4.3.10.2 Peer-to-Peer PTP Transparent Clock (P2P TC) Message Support

The IEEE 1588-2008 supports peer-to-peer PTP (Pdelay) message in addition to the SYNC, Delay Request, Follow-up, and Delay Response messages. Figure 141 shows the method to calculate the propagation delay in clocks supporting peer-to-peer path correction.

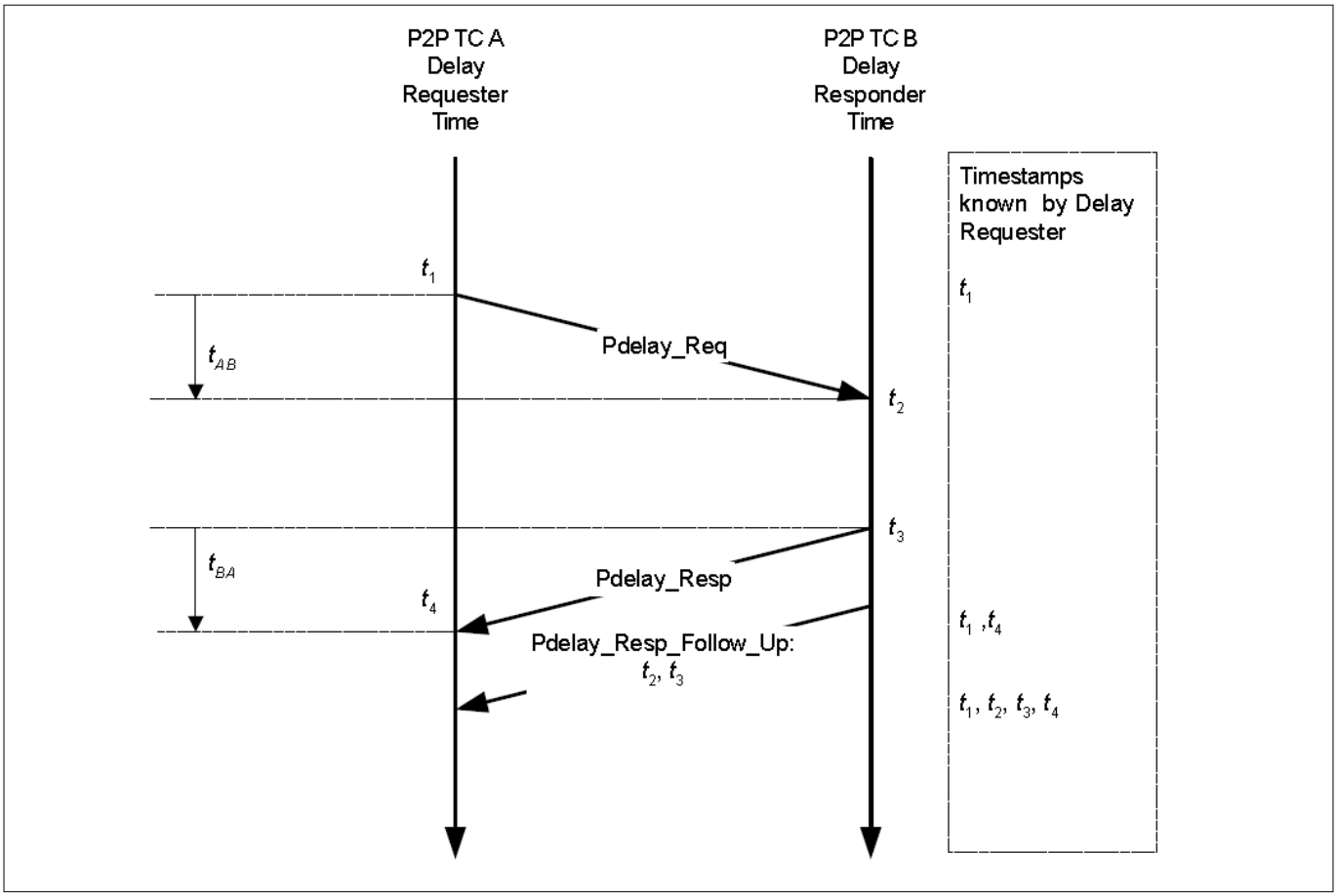


Figure 141 Propagation Delay Calculation in Clocks Supporting Peer-to-Peer Path Correction

As shown in Figure 141, the propagation delay is calculated in the following way:

1. Port 1 issues a Pdelay_Req message and generates a timestamp (t1) for the Pdelay_Req message.
2. Port 2 receives the Pdelay_Req message and generates a timestamp (t2) for this message.
3. Port 2 returns a Pdelay_Resp message and generates a timestamp (t3) for this message.

   To minimize errors because of any frequency offset between the two ports, Port 2 returns the Pdelay_Resp message as quickly as possible after the receipt of the Pdelay_Req message. Port 2 returns any one of the following:

   - Difference between the timestamps t2 and t3 in the Pdelay_Resp message
   - Difference between the timestamps t2 and t3 in the Pdelay_Resp_Follow_Up message
   - Timestamps t2 and t3 in the Pdelay_Resp and Pdelay_Resp_Follow_Up messages, respectively
4. Port 1 generates a timestamp (t4) on receiving the Pdelay_Resp message.
5. Port 1 uses all four timestamps to compute the mean link delay.

14.4.3.10.3 Clock Types

The XGMAC supports the following clock types defined in the IEEE 1588-2008:

- Ordinary Clock
- Boundary Clock
- End-to-End Transparent Clock
- Peer-to-Peer Transparent Clock

Ordinary Clock

The ordinary clock in a domain supports a single copy of the protocol. The ordinary clock has a single PTP state and a single physical port. In typical industrial automation applications, an ordinary clock is associated with an application device such as a sensor or an actuator. In telecoms applications, the ordinary clock can be associated with a timing demarcation device.

The ordinary clock can be a grandmaster or a slave clock. It supports the following features:

- Sends and receives PTP messages. The timestamp snapshot can be controlled as described in Portj_MAC_Timestamp_Control in the section "Registers".
- Maintains the data sets such as timestamp values

Table 693 shows the messages for which you can take the timestamp snapshot on the receive side for master and slave nodes.

Table 693 Ordinary Clock: PTP Messages for Snapshot

Master	Slave
Delay_Req	SYNC

For an ordinary clock, you can take the snapshot of either of the following PTP message types: version 1 or version 2. You cannot take the snapshots for both PTP message types. You can take the snapshot by setting the TSVER2ENA bit and selecting the snapshot mode in Port j MAC Timestamp Control.

Boundary Clock

The boundary clock typically has several physical ports communicating with the network. The messages related to synchronization, master-slave hierarchy, and signaling terminate in the protocol engine of the boundary clock and such messages are not forwarded. The PTP message type status given by the MAC helps you to identify the type of message and take appropriate action.

The boundary clock is similar to the ordinary clock except for the following features:

- The clock data sets are common to all ports of the boundary clock
- The local clock is common to all ports of the boundary clock

End-to-End Transparent Clock

The end-to-end transparent clock supports the end-to-end delay measurement mechanism between the slave clocks and the master clock. The end-to-end transparent clock forwards all messages like normal bridge, router, or repeater. The residence time of a PTP packet is the time taken by the PTP packet from the Ingress port to the Egress port.

The residence time of a SYNC packet inside the end-to-end transparent clock is updated in the correction field of the associated Follow_Up PTP packet before it is transmitted. Similarly, the residence time of a Delay_Req packet, inside the end-to-end transparent clock, is updated in the correction field of the associated Delay_Resp PTP packet before it is transmitted. Therefore, the snapshot needs to be taken at both Ingress and Egress ports only for the messages mentioned in Table 694. You can take the snapshot by setting the SNAPTYPSEL bits to 10 in the Portj_MAC_Timestamp_Control register as shown in Table 702.

Table 694 End to End Transparent Clock: PTP Messages for Snapshot

PTP Messages
SYNC
Delay_Req

Peer-to-Peer Transparent Clock

The peer-to-peer transparent clock differs from the end-to-end transparent clock in the way it corrects and handles the PTP timing messages. In all other aspects, it is identical to the end-to-end transparent clock.

In the peer-to-peer transparent clock, the computation of the link delay is based on an exchange of Pdelay_Req, Pdelay_Resp, and Pdelay_Resp_Follow_Up messages with the link peer. The residence time of the Pdelay_Req and the associated Pdelay_Resp packets is added and inserted into the correction field of the associated Pdely_Resp_Followup packet. Therefore, support for taking snapshot for the event messages related to Pdelay is added as shown in Table 695.

Table 695 Peer-to-Peer Transparent Clock: PTP Messages for Snapshot

PTP Messages
SYNC
Pdelay_Req
Pdelay_Resp

You can take the snapshot by setting the SNAPTYPESEL bit to 11 in the Portj_MAC_Timestamp_Control register.

Related information

Registers on page 802

14.4.3.10.4 Reference timing source

To get a snapshot of the time, the MAC requires a reference time in 64-bit format as defined in the IEEE 1588-2002 (80-bit format as defined in the IEEE 1588-2008). The XGMAC provides the following method for using the reference timing source in a node:

- Internal Reference Time (80-bit)

  It takes only the reference clock input and uses it to internally generate the reference time (also called the system time) and capture timestamps. The timestamp has the following fields:

  - UInteger48 secondsField:

    The seconds field is the integer portion of the timestamp in units of seconds. It is 48-bits wide. For example, 2.000000001 seconds are represented as secondsField = 0x0000_0000_0002
  - UInteger32 nanosecondsField:

    The nanoseconds field is the fractional portion of the timestamp in units of nanoseconds. For example, 2.000000001 seconds are represented as nanosecondsField = 0x0000_0001

    The nanoseconds field supports the following two modes:

    - Digital rollover mode: In this mode, the maximum value in the nanoseconds field is 0x3B9A_C9FF, that is, (10e9-1) nanoseconds
    - Binary rollover mode: In this mode, the nanoseconds field rolls over and increments the seconds field after value 0x7FFF_FFFF. Accuracy is ~0.466 ns per bit

      You can set these modes through TSCTRLSSR bit in the Port j MAC Timestamp Control register
- External Reference Time (80-bit):

  If this is selected by setting MAC_Timestamp_Control.ESTI, the reference clock output from the other XGMAC (only available in products with two XGMACs) is used as the reference time (also called the system time)to capture timestamps. This allows to have the identical time on both XGMACs

The timestamp maintained in the MAC is 64-bit wide. The upper 16-bits of the seconds field can be read from the Port j MAC System Time Higher Word Seconds register.

The XGMAC supports the following reference timing source features:

- Fixed pulse-per-second output
- Flexible pulse-per-second output

Related information

Fixed pulse-per-second output on page 1457
Flexible pulse-per-second output on page 1457

Fixed pulse-per-second output

The XGMAC supports the pulse-per-second (PPS) output that is given to indicate 1 second interval (default). You can change the frequency of the PPS output by setting the PPSCTRL0 field in the Port j MAC PPS Control register.

Flexible pulse-per-second output

The XGMAC also provides the flexibility to program the start or stop time, width, and interval of the pulse generated on the ptp_pps_o output. You can configure up to two separate PPS outputs.

The XGMAC provides the following features with the flexible PPS output option:

- Supports programming the start or stop time in terms of system time
- Supports programming the start point of the single pulse and start and stop points of the pulse train in terms of 64-bit system time. The Target Time registers are used to program the start and stop time
- Supports programming the stop time in advance, that is you can program the stop time before the actual start time has elapsed
- Supports programming the width between the rising edge and corresponding falling edge of PPS signal output in terms of number of units of sub-second increment value programmed in the Port j MAC Sub Second Increment register. You can program the width of pulse from 1 to 2^{32}-1 units of sub-second increment value
- Supports programming the interval between the rising edges of PPS signal, in terms of number of units of sub-second increment value. You can program the interval between pulses from 1 to 2^{32}-1 units of sub-second increment value
- Provides the option to cancel the programmed PPS start or stop request
- Indicates error if the start or stop time being programmed has already elapsed

***Note**: The PTP Reference clock mentioned in the following sections is the clock at which the system time gets updated. It is defined in the "Clocks" section. When the TSCFUPDT bit of the Port j MAC Timestamp Control register is set to 0, this clock is similar to the clk_ptp_ref_i clock. In the Fine Correction mode, this is the clock tick at which the system time gets updated using incr_sub_sec_reg as described in the "System time register module" section.*

Related information

System Time Register Module on page 1458
Clocks on page 1592

PPS start or stop time

You can initially program the start time in the Target Time registers, Port j MAC PPS0 Target Time Seconds and Port j MAC PPS0 Target Time Nanoseconds. Similarly also for the second Flexible PPS output.

If required, you can again program the start or stop time but you can do it only after the earlier programmed value is synchronized to the PTP clock domain. Bit 31 of MAC_PPS(#i)_Target_Time_Nanoseconds register indicates that the synchronization is complete. This enables you to program the start or stop time in advance even before the earlier stop or start time has elapsed.

To ensure proper PPS signal output, you should program advanced system time for the start or stop time. If the application programs a start or stop time that has already elapsed, the MAC sets an error status bit indicating

the programming error. If enabled, the MAC also sets the Target Time Reached interrupt event. The application can cancel the start or stop request only if the corresponding start or stop time has not elapsed. If the time has elapsed, the cancel command has no effect.

PPS width and interval

The PPS width and interval are programmed in terms of granularity of system time, that is, number of the units of sub-second increment value. For example, to have a PPS pulse width of 40 ns and interval of 100 ns with PTP reference clock of 50 MHz, you should program the width and interval to values 2 and 5, respectively. You can achieve smaller granularity by using a faster PTP reference clock.

Before giving the command to trigger a pulse or pulse train on the PPS output, you should program or update the interval and width of the PPS signal output.

14.4.3.10.5 System Time Register Module

The system time generator module maintains the 64-bit time and updated using the input reference clock (clk_ptp_ref_i). This time is the source for taking snapshots (timestamps) of Ethernet packets being transmitted or received at the GMII interface.

The system time counter can be initialized or corrected using the coarse correction method. In this method, the initial value or the offset value is written to the Timestamp Update register. For initialization, the system time counter is written with the value in the Timestamp Update register. For system time correction, the offset value is added to or subtracted from the system time.

In the fine correction method, the frequency drift of a slave clock (clk_ptp_ref_i) with respect to the master clock (as defined in IEEE 1588) is corrected over a period of time instead of in one clock, as in coarse correction. This helps maintain linear time and does not introduce drastic changes (or a large jitter) in the reference time between PTP Sync message intervals. In this method, an accumulator sums up the contents of the Addend register, as shown in Figure 142. The arithmetic carry that the accumulator generates is used as a pulse to increment the system time counter. The accumulator and the addend are 32-bit registers. The accumulator acts as a high-precision frequency multiplier or divider.

This algorithm is shown in Figure 142.

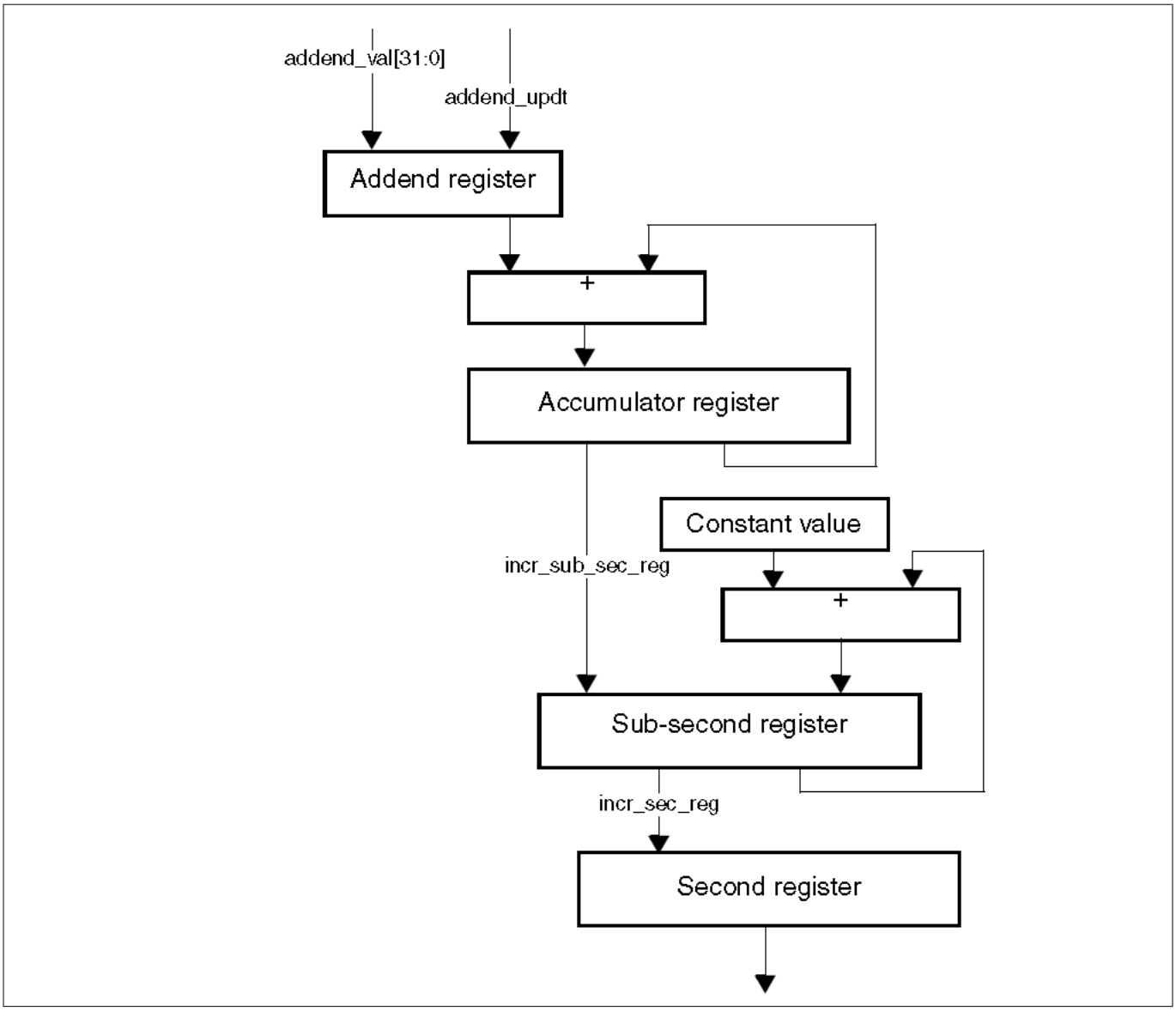


Figure 142 System Time Update Using Fine Method

The System Time Update logic requires a 50 MHz clock frequency to achieve 20 ns accuracy. The frequency division (FreqDivisionRatio) is the ratio of the reference clock frequency to the required clock frequency. For example, if the reference clock (clk_ptp_ref_i) is 250 MHz, this ratio is calculated as 250 MHz / 50 MHz = 5. Therefore, the default addend value to be set in the register is 2^{32} / 5, 0x33333333.

If the reference clock drifts lower, for example, to 248 MHz, the ratio is 248 / 50, or 4.96 and the value to set in the addend register is 2^{32} / 4.96, or 0x339CE739. If the clock drifts higher, for example, to 252 MHz, the addend register must be set to 0x32CB2CB2. When the clock drift is nil, the default addend value of 0x33333333 (2^{32} / 5) must be programmed.

In Figure 142, the constant value used to accumulate the sub-second register is decimal 43, which achieves an accuracy of 20 ns in the system time (in other words, it is incremented in 20 ns steps). When External Time Update is enabled, the optional System Time module is not available. Two different methods are used to update the System Time register depending on the configuration.

The software must calculate the drift in frequency based on the Sync messages and accordingly update the Addend register.

Initially, the slave clock is set with FreqCompensationValue0 in the Addend register. This value is as follows:

$FreqCompensationValue_0 = 2^{32}$ / FreqDivisionRatio

If MasterToSlaveDelay is initially assumed to be the same for consecutive Sync messages, the algorithm given in this section must be applied. After a few Sync cycles, frequency lock occurs. The slave clock can then determine a precise MasterToSlaveDelay value and synchronize again with the master, using the new value.

The algorithm is as follows:

- At time $MasterSyncTime_n$ the master sends the slave clock a Sync message. The slave receives this message when its local clock is $SlaveClockTime_n$ and computes$MasterClockTime_n$ as
  $MasterClockTime_n = MasterSyncTime_n + MasterToSlaveDelay_n$
- The master clock count for current Sync cycle, $MasterClockCount_n$ is
  $MasterClockCount_n = MasterClockTime_n - MasterClockTime_{n-1}$ (assuming that MasterToSlaveDelay is the same for Sync cycles n and n – 1)
- The slave clock count for current Sync cycle, $SlaveClockCount_n$ is
  $SlaveClockCount_n = SlaveClockTime_n - SlaveClockTime_{n-1}$
- The difference between master and slave clock counts for current Sync cycle, $ClockDiffCount_n$ is
  $ClockDiffCount_n = MasterClockTime_n - SlaveClockTime_n$
- The frequency-scaling factor for slave clock, $FreqScaleFactor_n$ is
  $FreqScaleFactor_n = (MasterClockCount_n + ClockDiffCount_n)/ SlaveClockCount_n$
- The frequency compensation value for Addend register, $FreqCompensationValue_n$ is
  $FreqCompensationValue_n = FreqScaleFactor_n * FreqCompensationValue_{n-1}$

In theory, this algorithm achieves lock in one Sync cycle. However, it might take several cycles, because of changing network propagation delays and operating conditions. This algorithm is self-correcting. If the slave clock is initially set to an incorrect value from the master, the algorithm corrects it at the cost of more Sync cycles.

14.4.3.10.6 Transmit Path Functions

The section text below enclosed in a comment is the 3rd party read-only source text. To make it editable, it has been cut and pasted below the comment. The text remains as a reference, should it change in the future.

The MAC captures a timestamp when the Start Frame Delimiter (SFD) of a packet is sent on the GMII or XGMII interface. The packets, for which you want to capture timestamps, can be controlled on per-packet basis. Each Transmit packet can be marked to indicate whether a timestamp should be captured for it.

The MAC does not process the transmitted packets to identify the PTP packets. You must specify the packets for which you want to capture timestamps. You can do this as follows: The MAC captures a timestamp when the Start Packet Delimiter (SFD) of a packet is sent on the GMII or XGMII interface. The packets, for which you want to capture timestamps, can be controlled on per-packet basis. Each Transmit packet can be marked to indicate whether a timestamp should be captured for

You can specify the packets by using the control bits in the Transmit descriptor. The 64-bit timestamp information is written to the Port j MAC Tx Timestamp Status Seconds and Port j MAC Tx Timestamp Status Nanoseconds register. The MAC generates "Tx Timestamp Captured" interrupt when 64-bit timestamp information is written to these registers.

Tx Timestamp Status FIFO with Packet Identifier

The MAC provides option to store 16 captured timestamps with associated packet identifier provided by the application. These can be accessed through Port j MAC Tx Timestamp Status PktID, MAC_TxTimestamp_Status_Seconds and Port j MAC Tx Timestamp Status Nanoseconds registers. The packet identifier helps the application to correlate the timestamp with the packet.

Table 696 Details of the Operating Modes

TTSE	OSTC	OSTPA	ATI Control Word 1 (TTSL & TTSH)	Timestamp Captured & Given in TS StatusFIFO	Comments
0	0	X	Not Valid	No	Default non-PTP packet
0	1	0	Not Valid	No	1-step PTP packet to insert Origin TS only (for example, SYNC, DelayResp. PdelayResp)
0	1	1	Ingress TS	No	1-step PTP packet to insert Origin TS and/or update Correction
1	0	0	Not Valid	Yes (PID = 0)	PTP 2-step without PID (for example, any PTP Event Message)
1	1	0	Not Valid	Yes (PID = 0)	1-step PTP packet to insert Origin TS (Example: SYNC, DelayResp. PdelayResp)
1	0	1	Packet ID	Yes with PID	PTP 2-step with PID (Example: Any PTP Event Message)
1	1	1	Ingress TS	Yes (PID = 0)	1-step PTP packet to insert Origin TS, update Correction & also give Origin TS it to software (DelayReq, PDelayReq)

14.4.3.10.7 Receive Path Functions

The MAC can be programmed to capture the timestamp of all packets received on the GMII or XGMII interface or to process packets to identify the valid PTP messages. You can control the snapshot of the time to be sent to the application by using the following options of the MAC_Timestamp_Control register.

- Enable snapshot for all packets
- Enable snapshot for IEEE 1588 version 1 or version 2 timestamp
- Enable snapshot for PTP packets transmitted directly over Ethernet or UDP-IP-Ethernet
- Enable timestamp snapshot for the received packet for IPv4 or IPv6
- Enable timestamp snapshot only for EVENT messages (SYNC, DELAY_REQ, PDELAY_REQ, or PDELAY_RESP)
- Enable the node to be a master or slave and select the snapshot type

  This feature controls the type of messages for which snapshots are taken.

***Note**: The XGMAC also supports the PTP messages over VLAN packets.*

The timestamp is provided as follows:

- The DMA returns the timestamp to the software inside the context Receive Descriptor. The timestamp message status is written in RDES3 and the snapshot of the timestamp is written in RDES0 and RDES1 fields of context descriptor. The RDES0 field holds the 32 least significant bits of the timestamp.

  ***Note**: The timestamp on RX packet is captured and provided in Rx Status only for valid Ethernet packets without any of the following errors:*

  - *Giant packet error*
  - *Length error*
  - *CRC error*
  - *Invalid code error*
  - *Watchdog timeout error*

14.4.3.10.8 Timestamp Correction

According to the IEEE 1588 standard, a timestamp must be captured when the message timestamp point (leading edge of the first bit of the octet immediately following the Start Frame Delimiter octet) crosses the boundary between the node and the network. Because the reference timing source (the PTP clock clk_ptp_ref_i) is different from the MAC Tx/Rx clock, correction needs to be performed to the captured timestamp for latency due to synchronization. Also, correction due to the latency between the internal snapshot point (XGMII/GMII) and the recommended capture point (the boundary between the node and the network) needs to be performed.

Ingress Correction

In the receive side, the timestamp captured at the internal snapshot point is corrected for latency and synchronization by adding the correction value (Ingress Correction Value) programmed in the Port j MAC Timestamp Ingress Corr Nanosecond. The value that needs to be programmed in the ingress correction register is calculated as described as follows.

The timestamp correction due to synchronization is compensated by adding INGRESS_SYNC_CORR to the synchronized timestamp value, which is as follows: `INGRESS_SYNC_CORR = (1*PTP_CLK_PER + 4*RX_CLK_PER)`

The latency correction between the message timestamp point and the XGMII receive interface is done by subtracting the latency value (INGRESS_LATENCY) with the captured timestamp.

Thus Ingress correction is as follows:

`Ingress Correction = INGRESS_SYNC_CORR - INGRESS_LATENCY - (3*RX_CLK_PER)`

(3*RC_CLK_PER) is due to the internal latency between the XGMII receive interface and the internal timestamp snapshot point. Ingress correction is performed by programming the TSIC field in the Port j MAC Timestamp Ingress Corr Nanosecond register. The ingress correction can be positive/negative and has an unit of nanoseconds. A negative value is represented in complement form as described as follows.

When the TSCTRLSSR bit in the Port j MAC Timestamp Control register is set, this has an accuracy of 1 ns. If the correction is positive, it is represented by setting bit 31 to 0 and bits 30:0 containing <ingress_correction_value> represented in binary. The value should not exceed 0x 3B9A_C9FF (1 second or 10^9 nanoseconds). If the correction is negative, it is represented by setting bit 31 to 1 and bits 30:0 containing 10^9 – <ingress_correction_value> represented in binary.

For example, if the required correction value is -5 ns, then the programmed value should be 0xBB9A_C9FB.

When the TSCTRLSSR bit in the Port j MAC Timestamp Control register is reset, this has an accuracy of ~0.466 ns. If the correction is positive, it is represented by setting bit 31 to 0 and bits 30:0 containing <ingress_correction_value> represented in binary. The maximum value is 0x7FFF_FFFF. If the correction is negative, it is represented by setting bit 31 to 1 and bits 30:0 containing 2^{31} – <ingress_correction_value> represented in binary.

Egress Correction

In the transmit side, the timestamp captured at the internal snapshot point is corrected for latency and synchronization by adding the correction value (Egress Correction Value) programmed in the Port j MAC Timestamp Egress Asym Corr register. The value that needs to be programmed in the egress correction register is calculated as described as follows.

The timestamp correction due to synchronization is compensated by adding EGRESS_SYNC_CORR to the synchronized timestamp value, which is as follows:

`EGRESS_SYNC_CORR = (1*PTP_CLK_PER + 4*TX_CLK_PER)`

The latency correction between the recommended capture point and the internal timestamp snapshot point is done by adding the latency value (EGRESS_LATENCY) with the captured timestamp.

Thus, Egress correction is as follows:

`Egress Correction = EGRESS_SYNC_CORR + EGRESS_LATENCY`

Egress correction is performed by programming the TSEC field in the Port j MAC Timestamp Egress Corr Nanosecond register. The egress correction is always positive and has an unit of nanoseconds.

When TSCTRLSSR bit in Port j MAC Timestamp Control register is set, this has an accuracy of 1 ns.

The value should not exceed 0x 3B9A_C9FF. When TSCTRLSSR bit in Port j MAC Timestamp Control register is reset, this has an accuracy of ~0.466 ns. The maximum value is 0x7FFF_FFFF.

Sub-Nanosecond Accuracy

When Enable IEEE 1588 Sub-Nanoseconds Timestamp Support is selected, then ingress and egress correction can have sub-nanoseconds accuracy. The sub-nanosecond part of the ingress and egress correction is programmed in the TSICSNS and TSECSNS field of the Port j MAC Timestamp Ingress Corr Subnanosecond and Port j MAC Timestamp Egress Corr Subnanosecond registers, respectively.

In this case, the correction has a unit of nanosecond, multiplied by 2^8. The least significant 8 bits of the value must be programmed in the sub-nanoseconds register.

For example, if the required egress correction is 3.6 ns and the TSCTRLSSR bit is set, the TSEC should be 0x3 and the TSECSNS should be 0x99 (3.6 * 256). Similarly, if the required ingress correction is -3.6 ns and the TSCTRLSSR bit is set, the TSIC should be 0xBB9A_C9FC and the TSICSNS should be 0x66 ((10^9 – 3.6) * 256).

Maximum PTP Clock Frequency

The resolution or granularity of the reference time source determines the accuracy of the synchronization. Therefore, a higher PTP clock frequency gives better system performance.

Minimum PTP Clock Frequency

The minimum PTP clock frequency depends on the time required between two consecutive SFD bytes and the time taken for synchronizing the time to the GMII or MII clock domain. This relationship is given in the following equation:

```
3 * PTP clock period + 4 * GMII/MII clock period <= Minimum gap between two SFDs
```

The GMII or MII clock frequency is fixed by IEEE specification. Therefore, the minimum PTP clock frequency required for proper operation depends on the operating mode and operating speed of the MAC as shown in Table 697.

***Note**:*

- *Use a clock that has constant frequency, preferably a divided application clock*
- *When IEEE 1588 timestamp feature is enabled with internal timestamp, use a PTP clock frequency which is greater than 5 MHz. This is because the 8-bit Port j MAC Sub Second Increment register limits the minimum PTP frequency that can be used to ~4 MHz*

Table 697 Minimum PTP Clock Frequency Example

Mode	Minimum Gap Between Two SFDs	Minimum PTP Frequency with External Timestamp Input	Minimum PTP Frequency with Internal Timestamp
10 Mbps full duplex	168 MII clocks (128 clocks for a 64-byte packet + 24 clocks of min IFG + 16 clocks of preamble)	~45 KHz	5 MHz
100 Mbps full duplex	168 MII clocks (128 clocks for a 64-byte packet + 24 clocks of min IFG + 16 clocks of preamble)	~0.5 MHz	5 MHz

(table continues...)

Table 697 (continued) Minimum PTP Clock Frequency Example

Mode	Minimum Gap Between Two SFDs	Minimum PTP Frequency with External Timestamp Input	Minimum PTP Frequency with Internal Timestamp
1G Mbps full duplex	84 GMII clocks (64 clocks for a 64-byte packet + 12 clocks of min IFG + 8 clocks of preamble)	~4.7 MHz	5 MHz
2.5G Full duplex (GMII)	84 GMII clocks (64 clocks for a 64-byte packet + 12 clocks of min IFG + 8 clocks of preamble)	12 MHz	12 MHz
2.5G Full duplex (XGMII)	21 XGMII clocks (64 clocks for a 64-byte packet + 12 clocks of min IFG + 8 clocks of preamble)	14 MHz	14 MHz
5G full duplex	21 XGMII clocks (16 clocks for a 64-byte packet + 3 clocks of min IFG + 2 clocks of preamble)	28 MHz	28 MHz

14.4.3.10.9 PTP Processing and Control

Table 698 shows the common message header for the PTP messages. This format is taken from the IEEE 1588-2008.

Table 698 Message Format Defined in IEEE 1588-2008

Bits								Octets	Offset
7	6	5	4	3	2	1	0		
transportSpecific				messageType				1	0
Reserved				versionPTP				1	1
messageLength								2	2
domainNumber								1	4
Reserved								1	5
flagField								2	6
correctionField								8	8
Reserved								4	16
sourcePortIdentity								10	20
sequenceId								2	30
controlField (*)								1	32
logMessageInterval								1	33

(*) – controlField is used in version 1. In version 2, messageType field is used for detecting different message types.

There are some fields in the Ethernet payload that you can use to detect the PTP packet type and control the snapshot to be taken. These fields are different for the following PTP packets:

- PTP Packets Over IPv4
- PTP Frames Over IPv6
- PTP Packets Over Ethernet

PTP Packets Over IPv4

Table 699 shows the fields that are matched to control snapshot for the PTP packets sent over UDP over IPv4 for IEEE 1588 version 1 and 2. The octet positions for the tagged packets are offset by 4. This is based on the IEEE 1588-2008, Annex D and the message format defined in Table 699.

Table 699 IPv4-UDP PTP Packet Fields Required for Control and Status

Field Matched	Octet Position	Matched Value	Description
MAC Packet Type	12, 13	0x0800	IPv4 datagram
IP version and Header Length	14	0x45	IP version is IPv4
Layer 4 Protocol	23	0x11	UDP
IP Multicast Address (IEEE 1588 version 1)	30, 31, 32, 33	0xE0, 0x00, 0x01, 0x81 (or 0x82 or 0x83 or 0x84)	Multicast IPv4 addresses allowed: • 224.0.1.129 • 224.0.1.130 • 224.0.1.131 • 224.0.1.132
IP Multicast Address (IEEE 1588 version 2)	30, 31, 32, 33	0xE0, 0x00, 0x01, 0x81 (Hex) 0xE0, 0x00, 0x00, 0x6B (Hex)	• PTP Primary multicast address: 224.0.1.129 • PTP Pdelay multicast address: 224.0.0.107
UDP Destination Port	36, 37	0x013F, 0x0140	• 0x013F: PTP event message[1)] • 0x0140: PTP general messages
PTP Control Field (IEEE 1588 version 1)	74	0x00, 0x01, 0x02, 0x03, 0x04	• 0x00: SYNC • 0x01: Delay_Req • 0x02: Follow_Up • 0x03: Delay_Resp • 0x04: Management

(table continues...)

Table 699 (continued) IPv4-UDP PTP Packet Fields Required for Control and Status

Field Matched	Octet Position	Matched Value	Description
PTP Message Type Field (IEEE 1588 version 2)	42 (nibble)	0x0, 0x1, 0x2, 0x3, 0x8, 0x9, 0xB, 0xC, 0xD	• 0x0: SYNC • 0x1: Delay_Req • 0x2: Pdelay_Req • 0x3: Pdelay_Resp • 0x8: Follow_Up • 0x9: Delay_Resp • 0xA: Pdelay_Resp_Follow_Up • 0xB: Announce • 0xC: Signaling • 0xD: Management
PTP Version	43 (nibble)	0x1 or 0x2	• 0x1: Supports PTP version 1 • 0x2: Supports PTP version 2

1) PTP event messages are SYNC, Delay_Req (IEEE 1588 version 1 and 2) or Pdelay_Req, Pdelay_Resp (IEEE 1588 version 2 only)

PTP Frames Over IPv6

Table 700 provides information about the fields that are matched to control the snapshots for the PTP packets sent over UDP over IPv6 for IEEE 1588 version 1 and 2. The octet positions for the tagged packets are offset by 4. This is based on the IEEE 1588-2008, Annex D and the message format defined in Table 698.

Table 700 IPv6-UDP PTP Packet Fields Required for Control and Status

Field Matched	Octet Position	Matched Value	Description
MAC Packet Type	12, 13	0x86DD	IP datagram
IP Version	14 (Bits [7:4])	0x6	IP version is IPv6
Layer 4 Protocol	20[1)] (*)	0x11	UDP
PTP Multicast Address	38 – 53	FF0x:0:0:0:0:0:0:181 (Hex) FF02:0:0:0:0:0:0:6B (Hex)	• PTP Primary multicast address: FF0x:0:0:0:0:0:0:0:181 (Hex) • PTP Pdelay multicast address: FF02:0:0:0:0:0:0:0:6B (Hex)
UDP Destination Port	56, 57[a]	0x013F, 0x140	• 0x013F: PTP event message • 0x0140: PTP general messages
PTP Control Field (IEEE 1588 version 1)	94[a]	0x00, 0x01, 0x02, 0x03, or 0x04	• 0x00: SYNC • 0x01: Delay_Req • 0x02: Follow_Up • 0x03: Delay_Resp • 0x04: Management (version1)

(table continues...)

Table 700 (continued) IPv6-UDP PTP Packet Fields Required for Control and Status

Field Matched	Octet Position	Matched Value	Description
PTP Message Type Field (IEEE 1588 version 2)	62[a] (nibble)	0x0, 0x1, 0x2, 0x3, 0x8, 0x9, 0xB, 0xC, or 0xD	• 0x0: SYNC • 0x1: Delay_Req • 0x2: Pdelay_Req • 0x3: Pdelay_Resp • 0x8: Follow_Up • 0x9: Delay_Resp • 0xA: Pdelay_Resp_Follow_Up • 0xB: Announce • 0xC: Signaling • 0xD: Management
PTP Version	63 (nibble)	0x1 or 0x2	• 0x1: Supports PTP version 1 • 0x2: Supports PTP version 2

1) The Extension Header is not defined for PTP packets.

PTP Packets Over Ethernet

Table 701 provides information about the fields that are matched to control the snapshots for the PTP packets sent over Ethernet for IEEE 1588 version 1 and 2. The octet positions for the tagged packets are offset by 4. This is based on the IEEE 1588-2008, Annex D and the message format defined in Table 698.

Table 701 Ethernet PTP Packet Fields Required for Control And Status

Field Matched	Octet Position	Matched Value	Description
MAC Destination Multicast Address[1)]	0–5	01-1B-19-00-00-00 01-80-C2-00-00-0E	All PTP messages can use any of the following multicast addresses[2)]: • 01-1B-19-00-00-00 • 01-80-C2-00-00-0E[3)]
MAC Packet Type	12, 13	0x88F7	PTP Ethernet packet
PTP Control Field (IEEE 1588 version 1)	46	0x00, 0x01, 0x02, 0x03, or 0x04	• 0x00: SYNC • 0x01: Delay_Req • 0x02: Follow_Up • 0x03: Delay_Resp • 0x04: Management

(table continues...)

Table 701 (continued) Ethernet PTP Packet Fields Required for Control And Status

Field Matched	Octet Position	Matched Value	Description
PTP Message Type Field (IEEE 1588 version 2)	14 (nibble)	0x0, 0x1, 0x2, 0x3, 0x8, 0x9, 0xB, 0xC, or 0xD	• 0x0: SYNC • 0x1: Delay_Req • 0x2: Pdelay_Req • 0x3: Pdelay_Resp • 0x8: Follow_Up • 0x9: Delay_Resp • 0xA: Pdelay_Resp_Follow_Up • 0xB: Announce • 0xC: Signaling • 0xD: Management
PTP Version	15 (nibble)	0x1 or 0x2	• 0x1: Supports PTP version 1 • 0x2: Supports PTP version 2

1) The unicast address match of destination addresses (DA), programmed in MAC address 0 to 31, is used if the TSENMACADDR bit of the MAC_Timestamp_Control register is set.

2) IEEE 1588-2008, Annex F

3) The MAC does not consider the PTP version 1 messages with Peer delay multicast address (01-80-C2-00-00-0E) as valid PTP messages.

14.4.3.11 One-Step Timestamp

The XGMAC supports the one-step timestamp feature. When this feature is enabled, the MAC identifies the offset in the packet and inserts the timestamp received from the application or the timestamp taken for the packet being transmitted at that offset.

You can enable the one-step timestamp feature by selecting the Enable One-Step Timestamp Feature option. You can enable the one-step timestamp feature for a packet as follows:

- Set the OSTC bit in the transmit context descriptor

For updating the correction field in certain PTP packets, the ingress timestamp is to be given in the TSSL and TSSH fields. The timestamp is 80-bit and it also contains the content of higher 16 bits of Higher Timestamp Word Register.

***Note**: One-step timestamps require the sub-second time is programmed to operate in digital format (TSCTRLSSR bit in Port j MAC Timestamp Control register).*

MAC Transmit PTP Mode

Depending upon the type of message and its mode, the MAC updates the following fields of Transmit PTP packets:

- correctionField in the PTP header of messages
- originTimestamp in SYNC, Delay_Req, and Pdelay_Req messages

***Note**: For residence/turnaround time correction, it is assumed that the differences between egress and ingress timestamp may be a few seconds, therefore, the XGMAC-CORE supports a difference up to eight seconds. Do not use this feature if your implementations requires a residence/turnaround time greater than eight seconds.*

The table below shows how the PTP mode is selected for a message based on the settings of SNAPTYPSEL, TSMSTRENA, and TSEVNTENA bits of the Port j MAC Timestamp Control register. The MAC updates the corresponding fields (originTimestamp, correction field) in the transmit PTP packets received from the application.

Table 702 MAC Transmit PTP Mode

Programming			Mode	Per Packet Control[1]			Messages Processed on Tx
SNAPTYPSEL	TSMSTRENA	TSEVNTENA		TTSE[2]	OSTC[3]	TTS[4]	
2'b00	X	0	End-to-end transparent	0	1	Ingress TS	Sync (correction field for residence time and Ingress Asym cor)
							Delay_Req (correction field for residence time and Egress Asym Cor)
2'b00	0	1	Ordinary or Boundary Slave	1	1	X	Delay_Req (originTimestamp field) Delay_Req (correction field for Egress Asym cor)
2'b00	1	1	Ordinary or Boundary Master	0	1	X	Sync (originTimestamp field) Sync (correction field for sub-nanosecond cor)
2’b01	X	0	End-to-End Transparent with support for peer delay mechanism	0	1	Ingress TS	Sync (correction field for residence time and Ingress Asym cor)
						Ingress TS	Pdelay_Req (correction field for residence time and Egress Asym Cor)
						Ingress TS	Pdelay_Resp (correction field for residence time and Ingress Asym Cor)
2’b01	0	1	Ordinary or Boundary Slave with support for peer delay mechanism or Peer to Peer Transparent	0	1	Ingress TS	Sync (correction field for residence time and Ingress Asym cor) (applicable only for Peer to Peer transparent clock operation)
				1	1	X	Delay_Req (originTimestamp field) Delay_Req (correction field for Egress Asym cor)
				1	1	X	Pdelay_Req (originTimestamp field) Pdelay_Req (correction field for Egress Asym Cor)
				0	1	Ingress TS for Pdelay_Req	Pdelay_Resp (correction field for turnaround time and Ingress Asym Cor)
2’b01	1	1	Ordinary or Boundary Master with support for	0	1	X	Sync (originTimestamp field) Sync (correction field for sub-nanosecond cor)

(table continues...)

Table 702 (continued) MAC Transmit PTP Mode

Programming			Mode	Per Packet Control[1]			Messages Processed on Tx
SNAPTYPSEL	TSMSTRENA	TSEVNTENA		TTSE[2]	OSTC[3]	TTS[4]	
			peer delay mechanism	1	1	X	Pdelay_Req (originTimestamp field) Pdelay_Req (correction field for Egress Asym Cor)
				0	1	Ingress TS for Pdelay_Req	Pdelay_Resp (correction field for turnaround time and Ingress Asym Cor)
2'b10	X	X	End-to-End Transparent	0	1	Ingress TS	Sync (correction field for residence time and Ingress Asym cor)
						Ingress TS	Delay_Req (correction field for residence time and Egress Asym Cor)
2'b11	X	X	Peer-to-Peer Transparent	0	1	Ingress TS	Sync (correction field for residence time and Ingress Asym cor)

1) The per packet control values provided here are the recommended settings used by devices in typical PTP operation, for the programmed mode.
2) TTSE represents TTSE bit of transmit descriptor. The TTSE function is independent of the OST (One Step Timestamp) function and the programmed operation mode for OST. The MAC captures and returns the timestamp when the TTSE bit is set.
3) OSTC represents OSTC bit of transmit descriptor. When OSTC is set to 0, OST operation is not performed; that is PTP packet is not modified.
4) TTTS represents the timestamp value provided in the TTSH, TTSL fields of transmit descriptor.

***Notes**:*

1. *Residence time/turnaround time is calculated as the difference between the captured timestamp (egress timestamp) and the ingress timestamp*
2. *When sub-nanosecond feature is enabled, residence time calculation includes sub-nanosecond accuracy*
3. *Clocks supporting peer delay mechanism does not use delay request/response, but it is included in OST for flexibility*

Enable One-Step Time Stamping for PTP Over UDP

One step timestamp for PTP over UDP/IP is a configurable feature available in configurations with Enable IEEE 1588 Timestamp Support option selected. When this feature is selected, XGMAC supports the one step timestamp operation for PTP messages sent over UDP/IPv4 and UDP/IPv6. Because origin timestamp and/or correction field (based on the PTP message type and the programmed PTP mode) in the PTP message are updated as part of the one step timestamp operation, it also requires update to the checksum field of the UDP header encapsulating the PTP message payload.

In XGMAC, checksum is updated for a packet, in the MTL before the packet is sent to the MAC where the PTP message is updated for one step timestamp operation. PTP requires the timestamp to be captured while the packet (SFD of the packet) is on the line and one step timestamp requires the origin timestamp and/or correction field to be updated in the PTP header while the packet is being transmitted on the line. Because the PTP message is present in the UDP payload, it is not possible to update the checksum field in the UDP header for modifications made to the PTP message, because it is already transmitted by then. Instead, the following options are supported that are in line with specification mentioned in Annex D & E of IEEE 1588-2008:

- When PTP message is sent over UDP/IPv4, the UDP checksum may be set to 0. The application must set the UDP checksum to 0 and set CIC to 0 or 1 while providing the packet to the MAC and the MAC does not update checksum for changes in origin timestamp or correction field
- When PTP message is sent over UDP/IPv6, a transmitting node extends the UDP payload of all PTP messages by 2 octets beyond the end of the PTP message. The contents of the UDP checksum field or the final 2 octets of the UDP payload might be modified to ensure that the UDP checksum remains intact after modification of PTP fields. The application must add the two additional bytes at the end of the PTP message while providing the packet to the MAC and the MAC updates these two bytes to keep the checksum correct

A programmable option (CSC field in Port j MAC Timestamp Control register) supports UDP with non-zero checksum for IPv4. When set, the two bytes at the end of the PTP message are updated to keep the UDP checksum correct, similar to PTP over UDP/IPv6. So, the application must add two pad bytes at the end of the PTP message sent to the XGMAC and XGMAC updates these two bytes to keep the checksum correct.

Updating the pad bytes to keep the checksum correct requires the old value of correction field and/or origin timestamp in the packet. To get better latency and area, the application can set the origin timestamp field to zero, for PTP message in particular modes in which XGMAC overwrites the origin timestamp. For example, XGMAC operating in Ordinary/Boundary master mode overwrites the origin timestamp field of the sync message, when OSTC is set. Because the sync message is originated by the port operating in Ordinary/ Boundary master mode, it is possible for the application generating the message to set the origin timestamp field to zero.

The transmit checksum engine is automatically enabled when this feature is selected. XGMAC uses the transmit checksum engine to identify the start of the PTP message in the UDP/IP packet. The checksum engine has the capability to detect some kind of errors in the header (L3) and payload (L4). Also checksum is not computed for some type of packets, which is indicated by "No" in the table "Transmit Checksum Offload Engine Functions for Different Packet Types". For one step operation the packet is free of those errors and have "Yes/Not applicable" for the corresponding packet type in the same table. If an error is reported by the checksum engine or there is a "No" in the table, for PTP messages sent over UDP/IP, no update is performed on the PTP message.

For more details about the errors detected by transmit checksum offload engine, see the section "Transmit Checksum Offload Engine" (see link in Related information).

Note:
- *XGMAC assumes an IP packet with OSTC set as a PTP packet. Therefore it is the responsibility of the application to set the OSTC bit only for IP packets which contains PTP messages*
- *For PTP over UPD/IPv6 and PTP over UDP/IPv4 with CSC set, the XGMAC does not verify the presence of the two pad bytes, instead it overwrites the last two bytes of the UDP payload*
- *When Enable One step timestamp for PTP over UDP/IP feature is selected, Enable Transmit TCP/IP Checksum Offload feature is automatically enabled and this is enabled only for Queue 0 in multiple transmit queue configurations. If the application requires OST support for PTP messages sent over UDP/IP for packets sent from queues other than 0, transmit checksum engine must be manually enabled for those transmit queues. If checksum engine is not enabled, no updates are done for PTP messages sent over UDP/IP even though OSTC is set*
- *When Enable One step timestamp for PTP over UDP/IP feature is selected and OSTC is set, the Transmit Checksum Offload Engine processes the UDP/IP packet for identifying the PTP Message Header start offset irrespective of value in CIC field. The CIC field needs to be appropriately set based on whether Transmit Checksum Offload Engine is expected to perform checksum insertion for that packet*
- *One step timestamp operation is not supported for tunneled packets*

When Enable One step timestamp for PTP over UDP/IP feature is selected and OSTC is set, the Transmit Checksum Offload Engine processes the UDP/IP packet for identifying the PTP Message Header start offset irrespective of value in CIC field. The CIC field needs to be appropriately set based on whether Transmit Checksum Offload Engine is expected to perform checksum insertion for that packet.

1. TTSE represents
- TTSE bit of transmit descriptor

2. OSTC represents
   - OSTC bit of transmit descriptor
3. TTS represents the timestamp value provided in the
   - TTSH and TTSL fields of transmit descriptor
4. The per packet control values specified are the recommended settings used by devices in a typical PTP operation, for the programmed mode
5. The TTSE function is independent of the OST function and the programmed operation mode for OST. The MAC captures and returns the timestamp when the TTSE bit is set
6. Residence time/turnaround time is calculated as the difference between the captured timestamp (egress timestamp) and the ingress timestamp
7. When sub-nanosecond feature is enabled, residence time calculation includes sub-nanosecond accuracy
8. Clocks supporting peer delay mechanism does not use delay request/response, but it is included in OST for flexibility

Related information

Transmit checksum offload engine on page 1486

14.4.3.12 Memory Access in Debug Mode

The XGMAC supports the debug access to the FIFO memory. When you enable this feature in Register MTL_Debug_Control, you must assign the entire selected Tx memory to Tx Queue 0 and Rx memory to Rx Queue 0. In addition, the offload features such as Checksum Offload Engine and IEEE timestamp should not be enabled with this feature.

When this feature is enabled, the MAC insert the pad (if required) and computed CRC in the Transmit packet. The FIFOs operate in the store-and-forward mode.

***Note**:*

- *The debug access mode is not designed for dynamic switching from normal operation mode to debug access mode. The software/application must ensure that the Transmit and Receive paths are idle (All the debug status registers in CSR space are read as all-zeros). The software/ application should wait for idle state or use mechanism such as Transmit Queue flush or soft reset to reach the idle state faster, prior to switching to debug access mode.*
- *While switching from debug access mode to the normal mode of operation, DMA should be reconfigured.*
- *In both the memory debug access modes, that is, Debug Mode Access and Slave Mode Access for applicable memories, the respective whole memory is accessed as a single memory; there is no per queue or hardware or software owned lists based logical partitioning. For example, the whole of Transmit Queue memory must be allocated to Transmit Queue 0 by programming TQS field of MTL_TxQ0_Operation_Mode register. Similarly, the whole of Receive Queue memory must be allocated to Receive Queue 0 by programming RQS field of MTL_RxQ0_Operation_Mode register. Size for other memories such as EST memory and Receive Parser Memory cannot be programmed by user. In debug access mode, the MAC internally switches to accessing the respective entire memory as a single memory.*

14.4.3.12.1 Slave Mode Access

You can use this mode to send or receive a packet using the Register interface. The host controller can push a complete packet into the Tx FIFO and then schedule it for transmission. Similarly, it can read the packet received and stored in the Rx FIFO through the registers.

The following section describes the sequence of programming to transmit or receive a packet in this mode.

Transmitting Packets through a Slave Interface

To write a packet to the Tx memory through the slave interface, complete the steps given in Table 703.

Table 703 Steps to Write a Packet to Tx Memory Through a Slave Interface

Register Name	Step Number and Description
MTL_Debug_Control	1. Select the FIFO for slave access by using the DBGMOD, FDBGEN, and FIFOSEL fields.
MTL_FIFO_Debug_Data	2. Write the control word in the FDBGDATA field.
MTL_Debug_Control	3. Complete the following: a. Write the Control Word encoding to the PKTSTATE field. b. Set the FIFOSEL field to Tx FIFO. c. Set the FIFOWREN bit. Note: The Control Word must contain the valid packet size.
MTL_Debug_Status	4. Read the FIFOBUSY bit and wait till this bit is low. When this bit is low, read the LOCR field (if required).
MTL_FIFO_Debug_Data	5. Write the SOP data in the FDBGDATA field.
MTL_Debug_Control	6. Complete the following steps: a. Write the SOP encoding in the PKTSTATE field. b. Set the FIFOSEL field to Tx FIFO. c. Set the FIFOWREN bit.
MTL_Debug_Status	7. Read the FIFOBUSY bit and wait till this bit is low. When this bit is low, read the LOCR field (if required).
MTL_FIFO_Debug_Data	8. Write subsequent data words in the FDBGDATA field.
MTL_Debug_Control	9. Complete the following steps: a. Write the packet data encoding in the PKTSTATE field. b. Set the FIFOSEL field to Tx FIFO. c. Set the FIFOWREN bit.
MTL_Debug_Status	10. Read the FIFOBUSY bit and wait till this bit is low. When this bit is low, read the LOCR field (if required).
MTL_Debug_Control and MTL_Debug_Status	11. Repeat, Step 8, Step 9, and Step 10 and till the last word of the packet data.
MTL_FIFO_Debug_Data	12. Write the last data word in the FDBGDATA field.
MTL_Debug_Control	13. Complete the following steps: a. Write the EOP encoding in the PKTSTATE field. b. Set the FIFOSEL field to Tx FIFO. c. Set the FIFOWREN bit. d. Set the number of valid bytes in the BYTEEN field.
MTL_Debug_Status	14. Read the FIFOBUSY bit and wait till this bit is low. When this bit is low, read the LOCR field (if required). The packet is transmitted after the EOP is written.

Receiving Packets through a Slave Interface

To read a packet from the Rx memory through a slave interface, complete the steps given in Table 704.

Table 704 Steps to Read a Packet from Rx Memory Through a Slave Interface

Register Name	Step Number and Description
MTL_Debug_Control	1. Select the FIFO for slave access by programming the PKTIE, DBGMOD, and FDBGEN fields.
MTL_Debug_Status	2. Wait for the PKTI interrupt.
MTL_Debug_Control	3. Complete the following: a. Set the FIFOSEL field to Rx FIFO. b. Set the FIFORDEN bit.
MTL_Debug_Status	4. Read the FIFOBUSY bit and wait till this bit is low.
MTL_Debug_Control	5. Read the PKTSTATE and BYTEEN fields.
MTL_FIFO_Debug_Data	6. Read the data in the FDBGDATA field.
MTL_Debug_Status	7. Read the LOCR field (if required).
MTL_Debug_Status, MTL_Debug_Control, MTL_FIFO_Debug_Data	8. Repeat Step 3 through Step 7 till the complete packet, along with the Rx status, is received.

14.4.3.12.2 Debug Mode Access

You can use this mode to test the FIFO memories by performing writes operations in a location and read back the same. You can switch the mode when the complete word is read or written in one location. The following sections describe how to write to or read from FIFO memory in debug mode.

Writing to FIFO Memory in Debug Mode

To write to FIFO memory in the debug mode, complete the steps given in Table 705.

Table 705 Steps to Write to FIFO memory in Debug Mode

Register Name	Step Number and Description
Port j MTL Debug Control	1. Select the FIFO for debug access by using the DBGMOD and FDBGEN fields. Set RSTALL to clear all the pointers to the start location (0x0). If you want to clear the pointers of a specific memory, select using FIFOSEL and set RSTSEL.
Port j MTL FIFO Debug Data	2. Write the data in the FDBGDATA field.
Port j MTL Debug Control	3. Complete the following: a. Set the FIFOSEL field to Tx FIFO, Rx FIFO. b. Set the FIFOWREN bit.
Port j MTL Debug Status	4. Read the FIFOBUSY bit and wait till this bit is low. When this bit is low, read the LOCR field (if required).
Port j MTL Debug Control Port j MTL Debug Status	5. Repeat Step 2, Step 3, and Step 4 twice to write one complete data word in 64-bit mode. Repeat these steps four times in 128-bit mode. The LOCR field is incremented to the next location when a complete data word is written. You can repeat the Step 2 to Step 5 for writing in to the subsequent locations.

Reading from FIFO Memory in Debug Mode

To read from FIFO memory in the debug mode, complete the steps given in Table 706.

Table 706 Steps to Read from FIFO Memory in Debug Mode

Register Name	Step Number and Description
Port j MTL Debug Control	**1.** Select the FIFO for debug access by using the DBGMOD and FDBGEN fields. Set RSTALL to clear all the pointers to the start location (0x0). If you want to clear the pointers of a specific memory, select using FIFOSEL and set RSTSEL **2.** Complete the following: a. Set the FIFOSEL field to Tx FIFO, Rx FIFO b. Set the FIFORDEN bit
Port j MTL Debug Status	3. Read the FIFOBUSY bit and wait till this bit is low. When this bit is low, read the LOCR field (if required)
Port j MTL FIFO Debug Data	4. Read the data in the FDBGDATA field
Port j MTL Debug Status	5. Read the LOCR field (if required)
Port j MTL Debug Control, Port j MTL Debug Status, Port j MTL FIFO Debug Data	6. Repeat Step 2 and Step 5 twice to read one complete data word in 64-bit mode. Repeat these steps four times in 128-bit mode. The LOCR field is incremented to the next location when a complete data word is read. You can repeat the Step 2 to Step 5 for reading from the subsequent locations

14.4.3.13 Flow Control

This section describes the flow control for Transmit and Receive paths.

14.4.3.13.1 Transmit flow control

The Transmit Flow Control involves transmitting Pause packets in full-duplex mode and backpressure in half-duplex mode, to control the flow of packets from the remote end.

The Transmit Flow Control is enabled, when the TFE bit is set in the MAC_Q(#i)_Tx_Flow_Ctrl register. Only PAUSE packets are transmitted.

Flow control trigger

The Transmit Flow Control involves transmitting Pause packets in full-duplex mode, and back pressure in half-duplex mode, to control the flow of packets from the remote end. The application can request the MAC to send a Pause packet by using the following method:

- Software Trigger: In this method, the application sets the FCB bit in the corresponding `MAC_Q(#i)_Tx_Flow_Ctrl` register

The flow control is triggered based on the following:

- Rx Queue Threshold: The flow control operation of the MAC is enabled when the EHFC bit of corresponding MTL_RxQ(#i)_Operation_Mode register is set. The flow control signal to the MAC is asserted when the fill level of the Rx queue crosses the threshold configured in RFA field of MTL_RxQ(#i)_Flow_Control register. This flow control signal is de-asserted when the fill-level of the queue falls below the threshold configured in the RFD field

  The hardware flow control generated based on the Rx Queue threshold crossing condition is applicable only when the Rx queue size is 4,096 bytes or more

***Note**: The RFS sets the threshold at which the flowcontrol is triggered and the MAC schedules to send a PAUSE frame to be sent by the local transmitter. The remote transmitter can continue to send packets until it receives the PAUSE packet. So the RXQ needs space to take in this data even after the flow-control is triggered,to avoid overflow and loss of packets. The max-space required that is calculated theoretically can be 2 max-sized packets, assuming that the read from the RXFIFO does not occur during this time. Theoretically, this makes RFA=4 or more, which implies that RxQ must have at the least 4 KB size. Practically, the system works with RFA=2 (Full – 2 KB, with very little probability of overflow).If you have FIFO size of less than 4 KB, with RFA=2, the flow control is triggered with just 1000 B in the RxQ which is even less than maxpacket size. The RFA field of MTL_RxQ(#i)_Flow_Control register is used for activating the Flow Control. These bits control the threshold(fill-level of Rx queue) at which the flow control is activated. For example, if your RXQ size is 2 KB and you set the RFA to 1 KB; as soon as you receive 1 KB, the Flow Control is activated. If you send a packet of 1500 bytes (which is allowed as per spec), this triggers,flow-control even before you receive a complete packet, which reduces the throughput considerably. If your RXQ size is 4 KB, you can set the RFA to 2 KB, so that you are able to receive the whole packet*

Flow control in half-duplex mode

In half-duplex mode, the MAC uses the deferral mechanism for the flow control (backpressure). When the application requests to stop receiving packets, the MAC sends a JAM pattern of 32 bytes when it senses a packet reception, provided the transmit flow control is enabled. This results in a collision and the remote station backs off. If the application requests a packet to be transmitted, it is scheduled and transmitted even when the backpressure is activated. If the backpressure is active for a long time (more than 16 consecutive collision events occur), the remote stations abort the transmission because of excessive collisions.

Table 708 describes the flow control in the Tx path for Queue0 based on the setting of the following bits:

- EHFC bit of the MTL_RxQ0_Operation_Mode register
- TFE and FCB bits of the MAC_Q0_Tx_Flow_Ctrl register
- HD (Half-Duplex mode) bit of MAC_Extended_Configuration register

Flow control is similar for all queues.

Flow control in full duplex mode

The XGMAC uses one of the following packet types for flow control:

- IEEE 802.3x Pause packets

MAC can send only PAUSE packet as per the programming of the MAC_Q0_Tx_Flow_Ctrl register.

Pause Packet Structure

Table 707 describes the fields of a Pause packet.

Table 707 Pause packet fields

Octets	Field	Pause packet
0 - 5	DA	Special Multicast address = 0180-c200-0001
6 - 11	SA	MAC_Adddress0 register
12 - 13	Type	Control type = 0x8808
14 - 15	Opcode	0x0001
16 - 17	PT/PEV	PT field of MAC_Q0_Tx_Flow_Ctrl register
	...	Reserved
60 - 63	FCS	CRC

Pause packet control

When the FCB bit is set, the MAC generates and transmits a single Pause packet. If the FCB bit is set again after the Pause packet transmission is complete, the MAC sends another Pause packet irrespective of whether the pause time is complete or not. To extend the pause or terminate the pause prior to the time specified in the previously-transmitted Pause packet, the application should program the Pause Time register with appropriate value and then again set the FCB bit.

Table 708 describes the flow control based on the setting of the following bits:

- EHFC bit of the MTL_RxQ0_Operation_Mode register
- TFE bit of the MAC_Q0_Tx_Flow_Ctrl register

Flow control is similar for all queues.

Table 708 Flow control in Tx path

EHFC	TFE	HD	Description
x	0	x	The MAC transmitter does not perform the flowcontrol or back-pressure operation
0	1	1	The MAC transmitter performs back-pressure when Bit 0 of MAC_Q0_Tx_Flow_Ctrl register is set
1	1	1	The MAC transmitter performs back-pressure when Bit 0 of MAC_Q0_Tx_Flow_Ctrl register is set. In addition, the MAC Tx performs back-pressure when Rx Queue level crosses the threshold set by RFA field of MTL_RxQ#_Flow_Controlregister
0	1	0	The MAC transmitter sends the Pause packet when Bit 0 of the MAC_Q0_Tx_Flow_Ctrl register is set
1	1	0	The MAC transmitter sends the Pause packet when Bit 0 of the MAC_Q0_Tx_Flow_Ctrl register is set

14.4.3.13.2 Receive flow control

In the receive path, the flow control is functional only in full-duplex mode. If pause packet is received in half-duplex mode, the packet is considered as a normal control packet.

Table 709 describes the flow control in the Rx path based on the setting of the following bits:

- RFE bit of the `MAC_Rx_Flow_Ctrl` register

Table 709 Flow control in Rx path

RFE	Description
0	The MAC receiver does not detect the received Pause packets
1	The MAC receiver detects or processes the Pause packets and responds to such packets by stopping the MAC transmitter

The following list describes the Rx flow control:

1. The MAC checks the destination address of the received Pause packet for either of the following:

- Multicast destination address: The DA matches the unique multicast address specified for the control packet (48'h0180C2000001)
- Unicast destination address: The DA matches the content of the MAC Address Register 0 and the UP bit of the `MAC_Rx_Flow_Ctrl` register is set.

  If the UP bit is set and the MAC processes Pause packets with unicast destination address in addition to the unique multicast address

2. The MAC decodes the following fields of the received packet:
   - Type field: This field is checked for 16'h8808
   - Opcode field: This field is checked for 16'h0001 (Pause packet)
   - Pause Time field: The Pause time (for Pause packet) is captured to determine the time for which transmitter needs to be blocked
3. If the byte count of the status indicates 64 bytes and there is no CRC error, the MAC transmitter does the following:
   - For 802.3x Pause packets, the MAC pauses the transmission of any data packet for the duration of the decoded Pause Time value multiplied by the slot time (64 byte times)
4. The MAC transfers the received control packet to the application based on the setting of the PCF field in the MAC_Packet_Filter register
5. If subsequent Pause packets are received before the earlier Pause Time expires, the MAC updates the Pause Timer with new value

14.4.3.14 Power management

The XGMAC supports the IEEE 1801 standard, also known as the Unified Power Format (UPF), for power management. In XGMAC, the power management functionality is implemented through Power Management block which supports the reception and detection of magic packets and remote wake-up (or wake-on-LAN) frames.

The power management functionality of XGMAC has the following features:

- Supports Power management block for coming out of the power-down mode
- Supports Destination Address (DA) filtering with perfect unicast MAC 0 address match for magic packets and remote wake-up frames
- Supports VLAN-tagged magic packets and remote wake-up frames
- Provides interrupts to allow software to switch off major blocks or during the power-down mode

14.4.3.14.1 Power management block

The Power Management (PMT) block is available in the receive path of XGMAC. The PMT block does not perform the clock gate function, but generates interrupts for remote wake-up frames and magic packets received by the MAC.

You can select any one or both power management frames (remote wake-up and magic packet). When you enable the power-down mode by switching off the application clock in register CLC, the MAC drops all received frames and does not forward these frames to the application. If software does not switch off the application clock during power down mode in register CLC, the magic packet is not dropped. The MAC comes out of the power-down mode only when a magic packet or a remote wake-up frame is received and the corresponding detection is enabled. After coming out of power-down mode, the software needs to switch on the application clock in register CLC. To generate power management events, the application can program the bits of the MAC_PMT_Control_Status.

Remote wake-up frame detection

When the MAC is in sleep mode and the remote wake-up bit is enabled in the MAC_PMT_Control_Status (0x00C0), the normal operation is resumed after a remote wake-up frame is received. The application writes all eight wake-up filter registers (two registers for each of the 4 wake-up filters), by performing a sequential Write to address (0x00C4). The application enables remote wake-up by writing 1 to bit 2 (RWKPKTEN) of the MAC_PMT_Control_Status.

The PMT block supports the following four programmable filters that enable you to define different receive frame patterns:

- The Filter Offset determines the offset from which the frame is examined
- The Filter Byte Mask determines the bytes of the frame that must be examined
- The Filter Command controls the filter operation
- The Filter CRC-16 contains the CRC_16 value calculated from the pattern

The MAC identifies a frame as remote wake-up frame if the following conditions are true:

- The destination address (DA) matches the value in MAC Address 0 for unicast packets, or the incoming packet is a multicast packet (including broadcast)
- The Filter CRC-16 matches the CRC of the incoming pattern. The remote wake-up CRC block determines the CRC value that is compared with Filter CRC-16

The remote wake-up frame is checked only for length error, FCS error, and XGMII error to ensure that it is not a runt frame. The remote wake-up frame detection is updated in the PMT Control and Status register for every remote wake-up frame received. A PMT interrupt to the application triggers a read operation to the PMT Control and Status register to determine reception of a remote wake-up frame.

VLAN frame support

The PMT block also supports the VLAN frames. The VLAN frame and a remote wake-up frame have the same wake-up pattern but at a different offset (4 bytes difference). In a VLAN frame, a four byte VLAN header is inserted between the Source Address and Frame Type field.

The VLAN header includes the following:

- Type ID assigned to bytes 12:13
- VLAN identifier, VID, assigned to bytes 14:15

The PMT block logic reads the VLAN type in the type ID field of offset 12 in bytes 12:13. If the VLAN type is detected, the PMT block increments the offset (by 4 bytes) to value 16. The hardware logic in the PMT block handles the VLAN packet identification and offset increment functionality.

Magic packet detection

The magic packet is based on a method that uses Advanced Micro Device's Magic Packet technology to power up the sleeping device on the network.

The MAC receives a specific packet of information, called a magic packet, addressed to the node on the network. The MAC checks the following types of packets to determine whether these packets meet the wake-up requirements:

- Unicast: The packets that are addressed to the MAC. For unicast packets, the Destination Address (DA) matches the value in MAC Address 0
- Multicast: The packets with multicast address (including broadcast)

The packets that pass the address filtering (unicast or multicast) are checked to determine whether they meet the remote magic packet data format of 6 bytes of all ones followed by a MAC address (Unicast MAC Address that matches the value in MAC Address0) appearing 16 times.

The application enables the magic packet detection by writing 1 to bit 1 (MGKPKTEN) of the MAC_PMT_Control_Status register. The PMT block constantly monitors each frame addressed to the node for a specific magic packet pattern. Each frame received is checked for a 48'hFF_FF_FF_FF_FF_FF pattern following the destination and source address fields, and the Length or Type fields. The PMT block then checks the frame for 16 repetitions of the MAC address without any breaks or interruptions. In case of a break in the 16 repetitions of the address, the PMT block again scans the 48'hFF_FF_FF_FF_FF_FF pattern in the incoming frame. The 16 repetitions can be anywhere in the frame, but must be preceded by the synchronization stream (48'hFF_FF_FF_FF_FF_FF). The device can also accept a multicast frame, as long as the 16 duplications of the MAC address are detected. If the number of repetitions of 8'hFF are more than 6, the PMT block checks for 16 repetitions of the MAC address without any breaks or interruptions, after the last 6 repetitions of 8'hFF.

If the MAC address of a node is 48'h00_11_22_33_44_55, then the PMT block scans the address for the following data sequence:

Destination Address Source Address Length/Type…….. FF FF FF FF FF FF

00 11 22 33 44 55 00 11 22 33 44 55 00 11 22 33 44 55 00 11 22 33 44 55

00 11 22 33 44 55 00 11 22 33 44 55 00 11 22 33 44 55 00 11 22 33 44 55

00 11 22 33 44 55 00 11 22 33 44 55 00 11 22 33 44 55 00 11 22 33 44 55
00 11 22 33 44 55 00 11 22 33 44 55 00 11 22 33 44 55 00 11 22 33 44 55
…CRC

The magic packet is checked only for length error, FCS error, and XGMII error to ensure that it is not a runt frame.

The magic packet detection is updated in the MAC_PMT_Control_Status for the received magic packet. A PMT interrupt to the application triggers a read to the MAC_PMT_Control_Status to determine whether a magic packet frame has been received.

14.4.3.14.2 PMT Registers

You can access the following PMT registers in similar manner as you access the XGMAC CSR registers. For mapping information, see MAC Registers Map in the section "Registers".

- MAC_PMT_Control_Status
- MAC_RWK_Packet_Filter

***Note**: The features defined in the bits PWRDWN and RWKPFE are not functional because power gating is not implemented. When power gating is not enabled, these features are non-functional.*

Related information

Registers on page 802

14.4.3.15 Energy Efficient Ethernet

Energy Efficient Ethernet (EEE) is an optional operational mode that enables the IEEE 802.3 Media Access Control (MAC) sub-layer along with a family of Physical layers to operate in the Low-Power Idle (LPI) mode. The EEE operational mode supports the IEEE 802.3 MAC operation at 100 Mbps, 1000 Mbps, and 10 Gbps.

The LPI mode allows power saving by switching off the parts of the communication device functionality when there is no data to be transmitted and received. The systems on the both sides of the link can disable some functionality and save power during low link-utilization period. The MAC controls whether the system should enter or exit the LPI mode and communicates this to the PHY.

The EEE specifies the capabilities negotiation methods that the link partners can use to determine whether EEE is supported and then select the best set of parameters common to both devices.

14.4.3.15.1 LPI transmit path functions

In the transmit path, the software must set the LPITXEN bit of the Port j MAC LPI Control Status register to indicate to the MAC to stop the transmission and initiate the LPI protocol. The MAC completes the transmission in progress and generates its transmission status. The MAC starts transmitting the LPI pattern instead of the IDLE pattern if the link status has been UP continuously for a period specified in the LPI LS TIMER field of Port j MAC LPI Timers Control register. The PHY Link Status bit of the LPI Control and Status Register indicates the link status of the PHY.

To make the PHY enter the LPI state, the MAC performs the tasks described in the following table.

Table 710 Transmit path functions for the GMII and XGMII interfaces

GMII/MII interface	XGMII interface
1. De-asserts TX_EN **2.** Asserts TX_ER **3.** Sets TXD[7:0] to 0x01 in GMII and TXD[3:0] in MII **4.** Updates the LPI entry status in the Port j MAC LPI Control Status register and generates the sbd_intr_o interrupt if the LPIIE bit of the Port j MAC Interrupt Enable register is set to 1	**1.** Drives XGMII_TXC[3:0] to 4'b1111 **2.** Drives XGMII_TXD[31:0] to 32'h06060606 **3.** Updates the LPI entry status in the Port j MAC LPI Control Status register and generates the sbd_intr_o interrupt if the LPIIE bit of the Port j MAC Interrupt Enable register is set to 1

***Note**: The MAC maintains the same state for the specified signals while it remains in the LPI state.*

To bring the PHY out of the LPI state, that is, when the software resets the LPITXEN bit, the MAC performs the following tasks:

1. Stops transmitting the LPI pattern and starts transmitting the IDLE pattern
2. Starts the LPI TW TIMER

   The MAC cannot start the normal frame transmission until the wake-up time specified for the PHY expires. The auto-negotiated wake-up interval is programmed in the TWT field (bits [15:0]) of `Port j MAC LPI Timers Control` register
3. Updates the LPI exit status of the `Port j MAC LPI Control Status` register and generates the `sbd_intr_o` interrupt if the LPIIE bit of the `Port j MAC Interrupt Enable` register is set to 1

The following figure shows the behavior of the TX_EN, TX_ER, and TXD[7:0] signals during the LPI mode transitions.

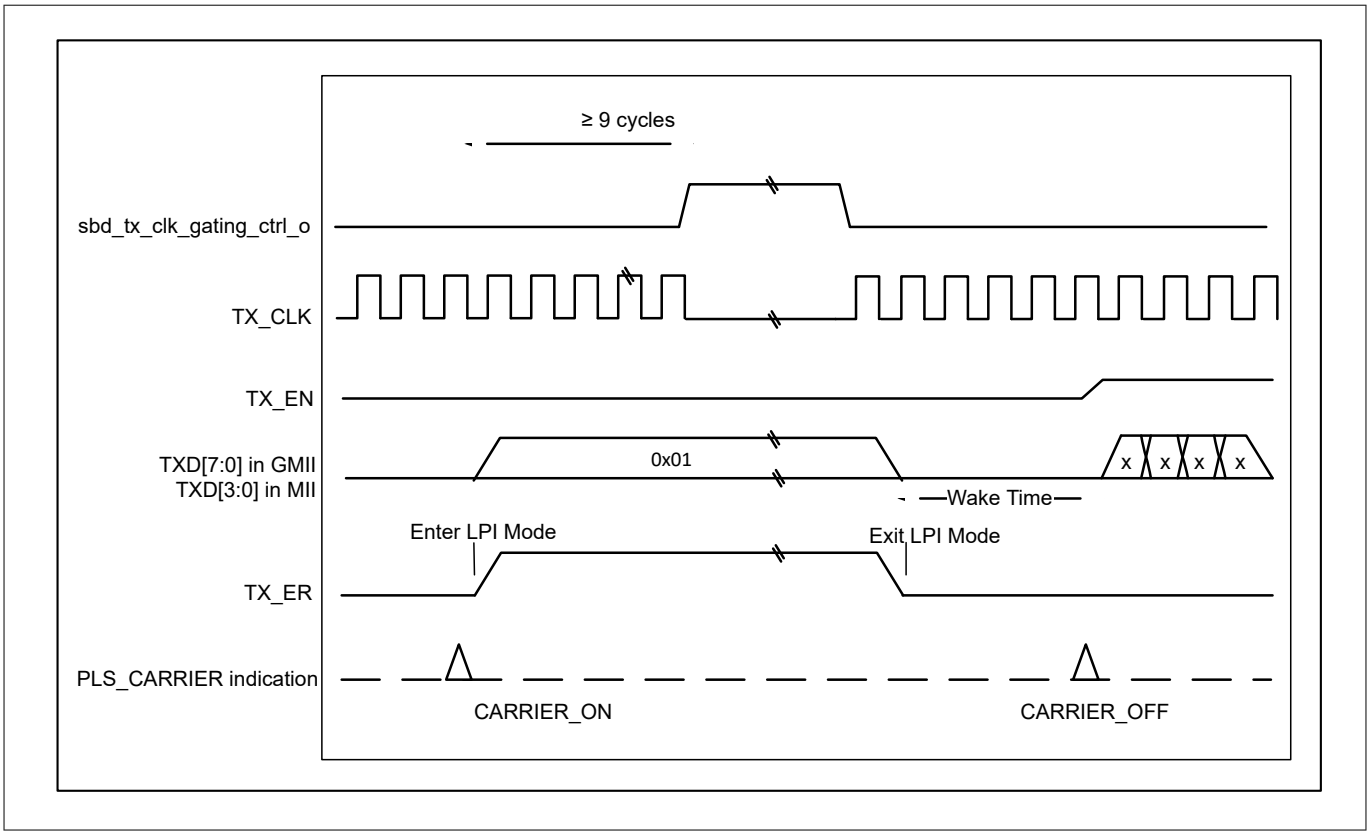


Figure 143 LPI transitions for GMII/MII (transmit)

The following figure shows the behavior of the XGMII_TXD[31:0] and XGMII_TXC[3:0] signals during the LPI mode transitions.

***Note**: When the MAC drives the LPI pattern, it does not internally generate the PLS_CARRIER indication (required by the Energy Efficient Ethernet standard) to defer normal transmission. The internal logic of the MAC ensures that it does not transmit normal frame when it is driving the LPI pattern and also during the subsequent LPI TW time.*

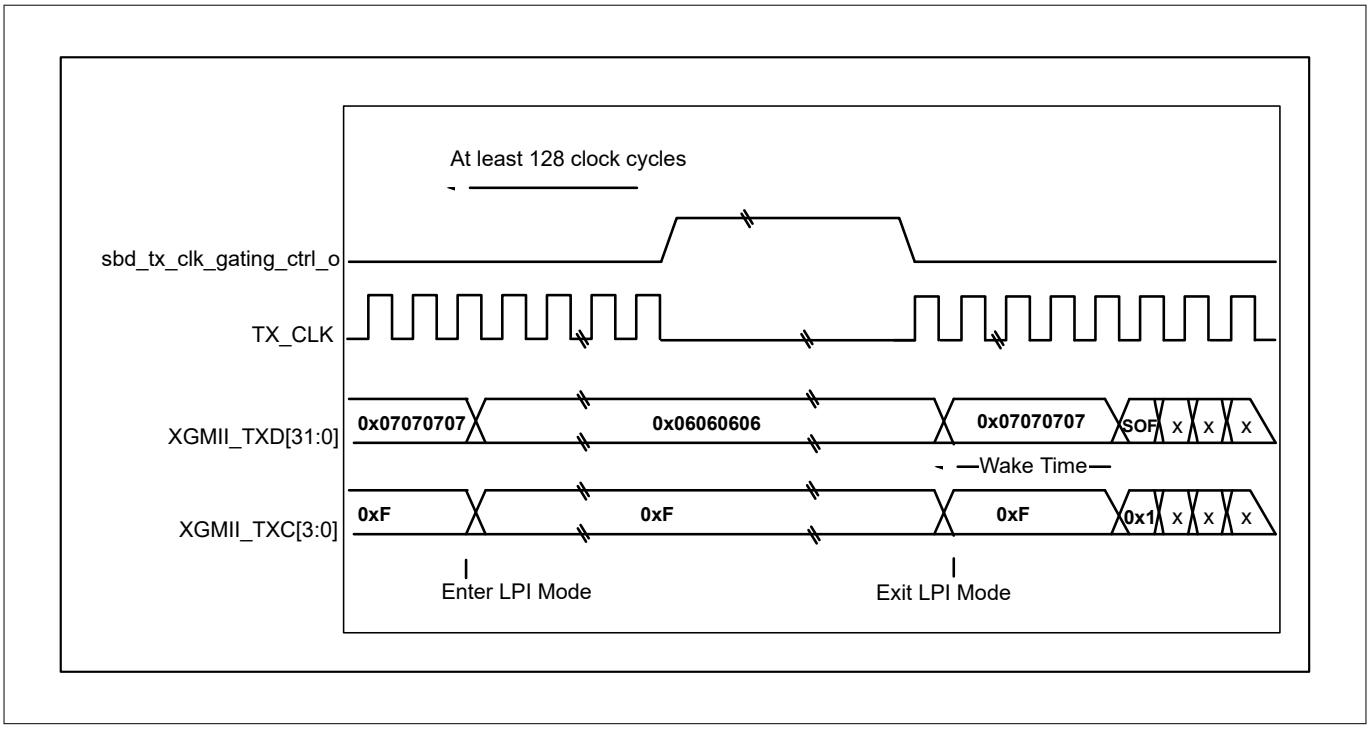


Figure 144 LPI transitions for XGMII (transmit)

Automated entry/exit of LPI mode in transmit path

The MAC transmitter can be programmed to enter and exit LPI IDLE mode automatically based on whether it is IDLE for a specific period of time or has a packet to transfer. These modes are enabled and controlled by Port j MAC LPI Control Status register.

When LPITXA (Bit[19]) and LPITXEN (Bit[16]) of Port j MAC LPI Control Status register are set, the MAC transmitter enters LPI IDLE state when the MAC transmit path (including the MTL layers and DMA layers) are idle. The MAC transmitter exits the LPI IDLE state and clear the LPITXEN bit as soon as any of functions in the TX path (DMA, MTL or MAC) becomes non-idle due to initiation of a packet transfer.

In addition, when Bit[20] (LPIATE) is also set, the MAC transmitter enters LPI IDLE state only if the Transmit path remains in idle state (no activity) for the time period indicated by the value in Port j MAC LPI Auto Entry Timer. In this mode also, the MAC transmitter exits the LPI IDLE state as soon as any of the functions becomes non-idle. However, the LPITXEN bit is not cleared but remains active so that re-entry to LPI IDLE state is possible without any software intervention when the MAC becomes idle again.

When both LPIATE and LPITXA bits are clear, the entry and exit of LPI IDLE state is directly controlled by the software by programming the LPITXEN bit.

14.4.3.15.2 LPI receive path functions

When the PHY receives the signals from the link partner to enter the LPI state, the PHY and MAC perform the tasks mentioned in the following table.

Table 711 Receive path functions for the GMII and XGMII interfaces

GMII/MII interface	XGMII interface
1. The PHY asserts RX_ER 2. The PHY sets RXD[7:0] to 0x01 in GMII and RXD[3:0] in MII 3. The PHY de-asserts RX_DV 4. The MAC updates the RLPIEN bit of the MAC_LPI_Control_Status register and generates the sbd_intr_o interrupt if the LPIIE bit of the MAC_Interrupt_Enable register is set to 1	1. The PHY drives XGMII_RXC[3:0] to 4'b1111 2. The PHY drives XGMII_RXD[31:0] to 32'h06060606 3. The MAC updates the RLPIEN bit of the MAC_LPI_Control_Status register and generates the sbd_intr_o interrupt if the LPIIE bit of the MAC_Interrupt_Enable register is set to 1
Note: The PHY maintains the specified signals while it remains in the LPI state	

When the PHY receives the signals from the link partner to exit the LPI state, the PHY and MAC perform the following tasks:

1. The PHY de-asserts RX_ER
2. The PHY sets the RXD[7:0] in GMII/RXD[3:0] in MII to 8'h00 for GMII or XGMII_RXD[31:0] to 32'h07070707 for XGMII
3. The PHY returns to a normal interframe state
4. The MAC updates the RLPIEX bit of the MAC_LPI_Control_Status register and generates the sbd_intr_o interrupt if the LPIIE bit of the MAC_Interrupt_Enable register is set to 1. The MAC also generates the sbd_intr_o interrupt on the additional sideband signal synchronized to Rx clock (lpi_intr_o)

The following table shows the behavior of the RX_ER, RX_DV, and RXD[7:0] signals during the LPI transitions.

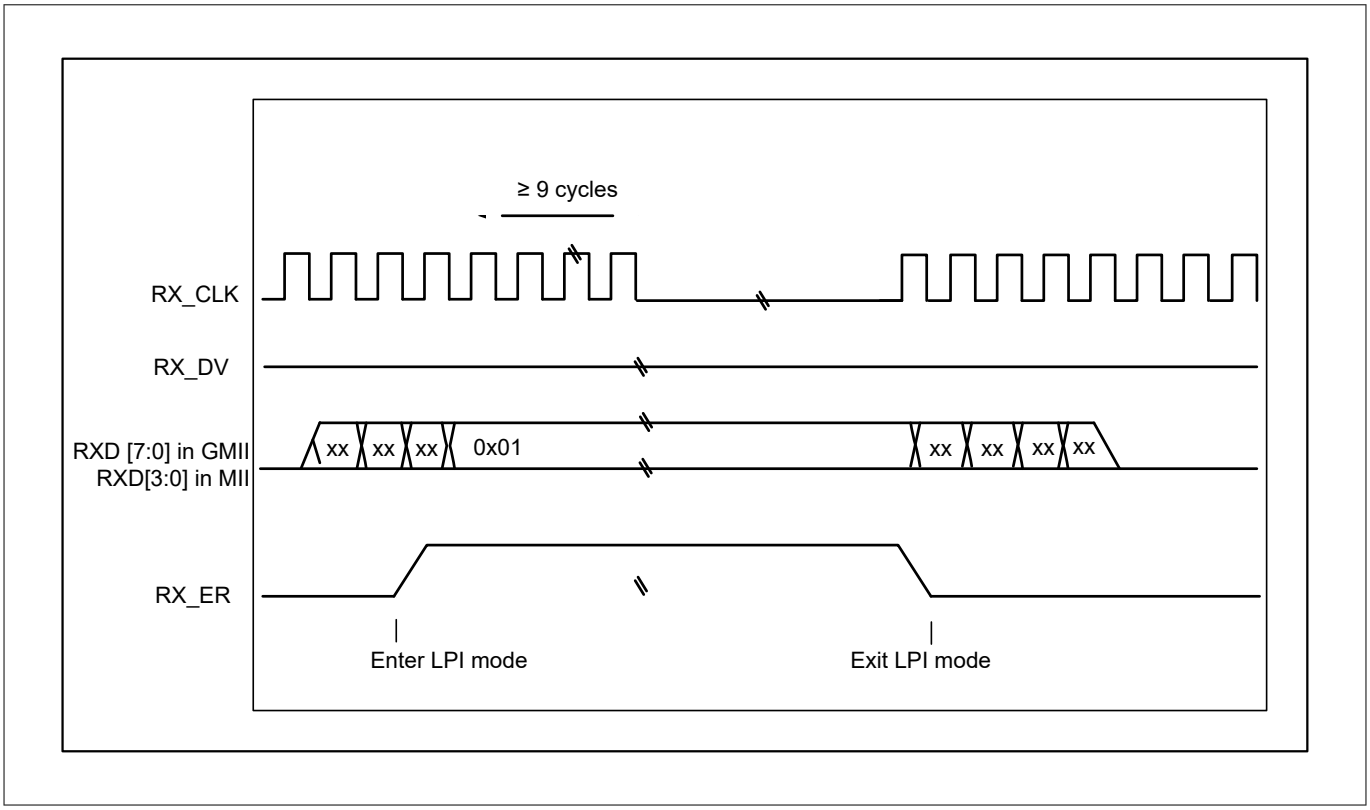


Figure 145 LPI transitions for GMII/MII (receive)

The following table shows the behavior of XGMII_RXD[31:0] and XGMII_RXC[3:0] signals during the LPI mode transitions.

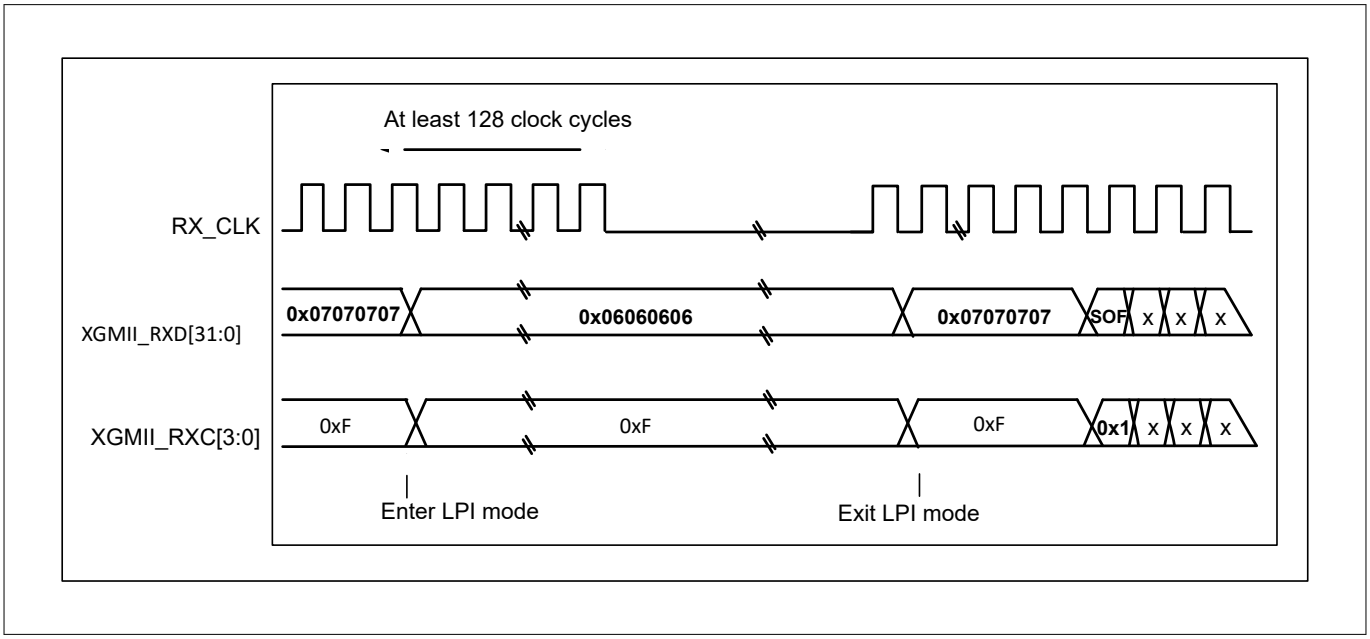


Figure 146 LPI transitions for XGMII (receive)

***Note**:*

- *If the RX_CLK_stoppable bit (in PHY register written through MDIO) is asserted when the PHY is indicating LPI to the MAC, the PHY may halt the RX_CLK at any time more than 128 clock cycles (for XGMII) and 9 clock cycles (for GMII) after the start of the LPI state as shown in figure "LPI Transitions for GMII/MII (Receive)" above*
- *If the MAC is in RX LPI mode and the RX clock is stopped by PHY, the application should not write to CSR registers that are synchronized to RX clock domain*

14.4.3.15.3 LPI Timers

The transmitter maintains the following timers that are loaded with the respective values from MAC_LPI_Timers_Control and MAC_LPI_Auto_Entry_Timer registers:

- LPI LS TIMER

  The LPI LS TIMER counts, in milliseconds, the time expired since the link status is UP. This timer is cleared every time the link goes down and is incremented when the link is UP.

  The XGMAC-CORE does not drive the LPI pattern unless the terminal count, programmed by the software, is reached. This ensures a minimum time during which no LPI pattern is driven after a link is established with the remote station. This period is defined as 1 second in the IEEE standard 802.3-az, version 2.0. The LPI LS TIMER is 10-bit wide. Therefore, the software can program up to 1023 milliseconds.
- LPI TW TIMER

  The LPI TW TIMER counts, in microseconds, the time expired since the de-assertion of Transmit LPI. The MAC can resume the normal transmit operation after the Transmit TW time. The software programs the Transmit TW time based on auto-negotiation.

  The MAC supports the LPI TW TIMER in units of microsecond. The LPI TW TIMER is 16-bit wide. Therefore, the software can program up to 65535 µs.
- LPI AUTO ENTRY TIMER

This timer counts in steps of eight microseconds, the time for which the MAC transmit path has to remain in idle state (no activity), before the MAC Transmitter enters the LPI IDLE state and starts transmitting the LPI pattern. This timer is enabled when LPIATE field of the `MAC_LPI_Control_Status` register is set.

14.4.3.16 Checksum Offload Engine

Communication protocols such as TCP/UDP implement checksum fields, which are useful to determine the integrity of the data transmitted over a network. Because the most widespread use of Ethernet is to encapsulate TCP/UDP over IP datagrams, the XGMAC Subsystem has an optional Checksum Offload Engine to support checksum calculation and insertion in the transmit path, and error detection in the receive path.

14.4.3.16.1 Transmit checksum offload engine

The checksum offload engine supports two types of checksum calculation and insertion.

- IP header checksum engine
- TCP/UDP/ICMP checksum engine

The checksum engine can be controlled for each packet by the CIC bits:

- TDES3 Bits[17:16]

***Note**:*

- *The checksum for TCP, UDP, or ICMP is calculated over a complete packet, and then inserted into its corresponding header field. Because of this requirement, when this function is enabled in a packet, the corresponding TxQ automatically operates in the store-and-forward mode even if the XGMAC is configured for Threshold (cut-through) mode*
- *You must make sure that the TxQ is deep enough to store a complete packet before that packet is transferred to the MAC transmitter. The reason being that when space is not available to accept the programmed burst length of data, then the MTL TxQ starts reading out the packet to avoid dead-lock. Therefore, you must enable the check-sum insertion only in the packets that are less than the following number of bytes in size:*

  TXQ SIZE in bytes – $((PBL + 5) \times 8)$,where TxQ SIZE is given by the TQS field in MTL_TxQ(#i)_Operation_Mode register.

  This calculation is also applicable for the normal and jumbo packets. The PBL is the TxPBL field in DMA_CH(#i)_Control_Register

***Note**: See IETF specifications RFC 791, RFC 793, RFC768, RFC 792, RFC 2460, and RFC 4443 for IPv4, TCP, UDP, ICMP, IPv6,and ICMPv6 packet header specifications, respectively.*

IP header checksum engine

In IPv4 datagrams, the integrity of the header fields is indicated by the 16-bit Header Checksum field (the eleventh and twelfth bytes of the IPv4 datagram). The COE detects an IPv4 datagram when the Type field of Ethernet packet has the value 0x0800 and the Version field of IP datagram has the value 0x4. The checksum field of the input packet is ignored during calculation and replaced with the calculated value.

IPv6 headers do not have a checksum field. Therefore, the COE does not modify the IPv6 header fields.

The IP Header Checksum engine considers the packet having an error for the following conditions:

- For IPv4 datagrams:
  - The received Ethernet type is 0x0800, but the Version field of IP header is not equal to 0x4

- The IPv4 Header Length field indicates a value less than 0x5 (20 bytes)
- The total packet length is less than the value given in the IPv4 Header Length field

- For IPv6 datagrams:
  - The Ethernet type is 0x86dd but the IP header Version field is not equal to 0x6
  - The packet ends before the IPv6 header (40 bytes) or extension header (as given in the corresponding Header Length field in an extension header) is completely received

If COE detects an IP header error, it still inserts an IPv4 header checksum if the Ethernet Type field indicates an IPv4 payload. The IP Header Error and Payload Checksum Error is provided in the MAC_Rx_Tx_Status register; an interrupt is generated when enabled.

The Payload Checksum engine declares an error when it detects that the packet has been forwarded to the MAC Transmitter engine in the store-and-forward mode without the end of packet (EOP) being written to the FIFO, or when the packet ends before the number of bytes indicated by the Payload Length field in the IP Header is received or when the Packet size (as indicated by IP header) is such that it can not accommodate minimum header size of TCP, UDP or ICMP header. When this engine detects these type of errors, it does not modify the TCP, UDP, or ICMP header.

When the packet is longer than the indicated payload length, the COE ignores them as stuff bytes, and no error is reported. When this engine detects this, it still inserts the calculated checksum into the corresponding TCP, UDP, or ICMP header field.

TCP/UDP/ICMP checksum engine

The TCP/UDP/ICMP Checksum Engine processes the IPv4 or IPv6 header (including extension headers) and determines whether the encapsulated payload is TCP, UDP, or ICMP. The checksum is calculated for the TCP, UDP, or ICMP payload and inserted into its corresponding field in the header. The Tx COE can work in the following two modes:

- The TCP, UDP, or ICMPv6 pseudo-header is not included in the checksum calculation and is assumed to be present in the checksum field of the input packet. This engine includes the checksum field in the checksum calculation, and then replaces the checksum field with the final calculated checksum
- The engine ignores the checksum field, includes the TCP, UDP, or ICMPv6 pseudo-header data into the checksum calculation, and overwrites the checksum field with the final calculated value

***Note**:* *For ICMP-over-IPv4 packets, the Checksum field in the ICMP packet must always be 16'h0000 in both modes, because pseudo-headers are not defined for such packets. If it does not equal 16'h0000, an incorrect checksum may be inserted into the packet.*

The Payload Checksum engine declares an error when it detects that the packet has been forwarded to the MAC Transmitter engine in the store-and-forward mode without the end of packet (EOP) being written to the FIFO, or when the packet ends before the number of bytes indicated by the Payload Length field in the IP Header is received or when the Packet size (as indicated by IP header) is such that it can not accommodate minimum header size of TCP, UDP or ICMP header. When the packet is longer than the indicated payload length, the COE ignores them as stuff bytes, and no error is reported. When this engine detects the first type of error, it does not modify the TCP, UDP, or ICMP header. For the second error type, it still inserts the calculated checksum into the corresponding header field.

Table 712 describes the functions supported by Transmit Checksum Offload engine based on the packet type. When the MAC does not insert the checksum, it is indicated as “No” in the table.

***Note**:* *Checksum insertion for IPv4 or IPv6 packets that are greater than the frame size constraint specified in Transmit Checksum Offload Engine should not be enabled because it may result in incorrect checksum insertion or unexpected behavior.*

If the total packet size from the ATI interface or DMA engine exceeds the limit of 14'h3FFF, the checksum is not inserted for that packet.

Table 712 Transmit checksum offload engine functions for different packet types

Packet type	Hardware IP header checksum insertion	Hardware TCP/UDP checksum insertion
Non-IPv4 or IPv6 packet	No	No
IPv4 packet is greater than 1,522 bytes (2,000 bytes when IEEE 802.3ad Support for 2K Packets is enabled in MAC) but less than or equal to the frame size constraint specified in Transmit Checksum Offload Engine.	Yes	Yes
IPv6 packet is greater than 1,522 bytes (2,000 bytes when IEEE 802.3ad Support for 2K Packets is enabled in MAC) but less than or equal to the frame size constraint specified in Transmit Checksum Offload Engine.	Not applicable	Yes
IPv4 with TCP, UDP, or ICMP	Yes	Yes
IPv4 packet has IP options (IP header is longer than 20 bytes)	Yes	Yes
Packet is an IPv4 fragment	Yes	No
IPv6 packet with the following next header fields in main or extension headers: • Hop-by-hop options (in IPv6 main header) • Hop-by-hop options (in IPv6 extension header) • Destinations options • Routing (with segment left 0) • Routing (with segment left > 0) • TCP, UDP, or ICMP • Authentication • Any other next header field in main or extension headers	• Not applicable • Not applicable • Not applicable • Not applicable • Not applicable • Not applicable • Not applicable • Not applicable	• Yes • No • Yes • No • No • Yes • No • No
IPv4 packet has TCP header with Options fields	Yes	Yes
IPv4 packet has 802.3ac tag (with C-VLAN Tag or S-VLAN Tag when enabled).	Yes	Yes
IPv6 packet has 802.3ac tag (with C-VLAN Tag or S-VLAN Tag when enabled).	Not applicable	Yes
IPv4 frames with security features (such as encapsulated security payload)	Yes	No
IPv6 frames with security features (such as encapsulated security payload)	Not applicable	No

14.4.3.16.2 Receive checksum offload engine

You can enable the Receive Checksum Offload Engine (Rx COE) by setting the IPC bit of the MAC_Rx_Configuration register. When this option is selected, both IPv4 and IPv6 packet in the received Ethernet packets are detected and processed for data integrity. The MAC receiver identifies IPv4 or IPv6 packets by checking for value 0x0800 or 0x86DD, respectively, in the Type field of the received Ethernet packet. This identification is applicable to single VLAN-tagged packets. It is also applicable to double VLAN-tagged packets when the Enable Double VLAN Processing option is selected and the EDVLP bit of the MAC_VLAN_Tag register is set.

The Rx COE calculates the IPv4 header checksums and checks that they match the received IPv4 header checksums. The result of this operation (pass or fail) is given to the RFC module for insertion into the receive status word. The IP Header Error bit is set for any mismatch between the indicated payload type (Ethernet Type field) and the IP header version, or when the received packet does not have enough bytes, as indicated by the Length field of the IPv4 header (or when fewer than 20 bytes are available in an IPv4 header), when DIS_TCP_EF bit is set in the MTL_RxQ(#i)_Operation_Mode register.

This engine also identifies a TCP, UDP, or ICMP payload in the received IP datagrams (IPv4 or IPv6) and calculates the checksum of such payloads properly, as defined in the TCP, UDP, or ICMP specifications. This engine includes the TCP, UDP, or ICMPv6 pseudo-header bytes for checksum calculation and checks whether the received checksum field matches the calculated value. The result of this operation is given as a Payload Checksum Error bit in the receive status word. This status bit is also set if the length of the TCP, UDP, or ICMP payload does not match the expected payload length given in the IP header.

Table 713 describes the functions supported by the Rx COE based on the packet type. When the payload of an IP packet is not processed (indicated as “No” in the table), the information (whether the checksum engine is bypassed or not) is given in the receive status.

***Note**: The MAC does not append any payload checksum bytes to the received Ethernet packets.*

Table 713 Receive checksum offload engine functions for different packet types

Packet type	Hardware IP header checksum checking	Hardware TCP/UDP/ICMP checksum checking
Non-IPv4 or IPv6	No	No
IPv4 packet is greater than 1,522 bytes (2,000 bytes when IEEE 802.3ad Support for 2K Packets is enabled in the MAC)	Yes	Yes
IPv6 packet is greater than 1,522 bytes (2,000 bytes when IEEE 802.3ad Support for 2K Packets is enabled in the MAC)	Not applicable	Yes
IPv4 with TCP, UDP, or ICMP	Yes	Yes
IPv4 header's protocol field contains a protocol other than TCP, UDP, or ICMP	Yes	No
IPv4 packet has IP options (IP header is longer than 20 bytes)	Yes	Yes
Packet is an IPv4 fragment	Yes	No
Authentication	Not applicable	Yes
IPv6 packet with the following next header fields in main or extension headers: • Hop-by-hop options (in IPv6 main header) • Hop-by-hop options (in IPv6 extension header) • Destinations options • Routing (with segment left 0) • Routing (with segment left > 0) • TCP, UDP, or ICMP • Any other next header field in main or extension headers	• Not applicable • Not applicable • Not applicable • Not applicable • Not applicable • Not applicable • Not applicable	• Yes • No • Yes • Yes • No • Yes • No

(table continues...)

Table 713 (continued) Receive checksum offload engine functions for different packet types

Packet type	Hardware IP header checksum checking	Hardware TCP/UDP/ICMP checksum checking
IPv4 packet has TCP header with Options fields	Yes	Yes
IPv4 packet has 802.3ac tag (with C-VLAN Tag or S-VLAN Tag when enabled).	Yes	Yes
IPv6 packet has 802.3ac tag (with C-VLAN Tag or S-VLAN Tag when enabled).	Not applicable	Yes
IPv4 frames with security features (such as encapsulated security payload)	Yes	No
IPv6 frames with security features (such as encapsulated security payload)	Not applicable	No

14.4.3.17 Management counters

The MMC maintains a set of registers for statistics on the received and transmitted frames. These include a control register for controlling the behavior of the MMC counters, two 32-bit registers containing interrupts generated (receive and transmit), and two 32-bit registers containing masks for the Receive and Transmit Interrupt registers. The application can access these registers through the MAC Control Interface (MCI). The counter can be 32-bit or 64-bit wide. Non-32 bit accesses are allowed only when the address is word-aligned.

The organization of these registers is shown in the section "Registers". MMC registers are accessed using transactions in the same way that CSR addresses are accessed. This section describes the statistics counters and lists the address for each. Counters, which are 64-bits wide, are divided into two 32-bit registers for access through the MCI.

The receive MMC counters are updated only for frames passed by the Destination Address Filter (AFM) block. Statistics of frames that are dropped by the AFM are not updated unless they are runt frames of less than 6 bytes (that is, DA bytes that are not fully received).

Related information

Registers on page 802

14.4.3.17.1 Address assignments

The MMC register naming convention is as follows:

- “tx” as a prefix or suffix indicates a transmit counter
- “rx” as a prefix or suffix indicates a receive counter
- “_g” as a suffix indicates a register that counts good frames only
- “_gb” as a suffix indicates a register that counts all frames, whether they are good or bad
- “_lo” as a suffix indicates the lower 32 bits of a 64-bit counter
- “_hi” as a suffix indicates the upper 32 bits of a 64-bit counter

The following explanations pertain to terminology used in Table 714:

- Transmitted frames are considered good frames when none of the following errors is observed:
  - Frame Underflow error
  - Local/remote fault detected

  - Jabber time-out
  - No carrier or loss of carrier
  - Late collision
  - Excessive Deferral
  - Excessive Collision
- Received frames are considered good when none of the following errors is observed:
  - CRC error
  - Length error
  - Runt frames (with or without CRC errors)
  - Out-of-range frames
  - Watchdog time-out
- The maximum transmit frame size depends on the frame type, as follows:
  - Untagged frame maxsize = 1,518
  - VLAN Frame maxsize = 1,522
  - Jumbo Frame maxsize = 9,018
  - JumboVLAN Frame maxsize = 9,022
- The maximum receive frame size depends on the frame type and control bits (JE, S2KP, GPSLCE and EDVLP) as follows:

Table 714 Maximum frame size dependency

JE	S2KP	GPSLCE	EDVLP	Untagged framemaxsize in bytes	Single VLAN framemaxsize in bytes	Double VLAN framemaxsize in bytes
1	X	X	1	9018	9022	9026
0	1	X	X	2000	2000	2000
0	0	1	1	GPSL	GPSL+4	GPSL+8
0	0	0	1	1518	1522	1526
1	X	X	0	9018	9022	9022
0	0	1	0	GPSL	GPSL+4	GPSL+4
0	0	0	0	1518	1522	1522

***Note**: Preambles of frames are not counted in the octet counters.*

14.4.3.18 Station Management Agent

The Station Management Agent (SMA) allows the application to access any PHY register through a two-wire Station Management Interface (SMI). To increase the number of MDIO Managed Devices (MMD) per port, the existing MDIO interface (IEEE 802.3, Clause 22) has been extended to support more device registers while still retaining logical compatibility with earlier MDIO interfaces. This extended MDIO interface is specified in IEEE 802.3, Clause 45.

An extension to the existing MDIO management frame structure defines indirect addressing to extend the address space, allowing more registers within each MMD to be addressed. The MDIO address space is orthogonal to the earlier MII management address space; that is, the start-of-frame delimiter is different from the one used for MII management frame structure.

The new management frame structure is as follows:

Table 715 Management Frame Fields

Frame	PRE	ST	OP	PORTAD	DEVAD	TA	REGAD	IDLE
Address	1…1	00	00	PPPPP	EEEEE	10	4'hAAAA	Z
Write	1…1	00	01	PPPPP	EEEEE	10	4'hDDDD	Z
Read	1…1	00	11	PPPPP	EEEEE	Z0	4'hDDDD	Z
Post-Read Increment Address	1…1	00	10	PPPPP	EEEEE	Z0	4'hDDDD	Z

Using this extended MDIO interface, the application can address one of the 32 PHYs, any one of 32 devices within the PHY, and one of the many registers within the given device and send control data or receive status information. Only one register in one PHY can be addressed at a time. For more details on frame protocol see Clause 45 of IEEE 802.ae.

The application sends control data to the PHY (MIM write cycle) and receives status information from the PHY (MIM read cycle) through the SMA.

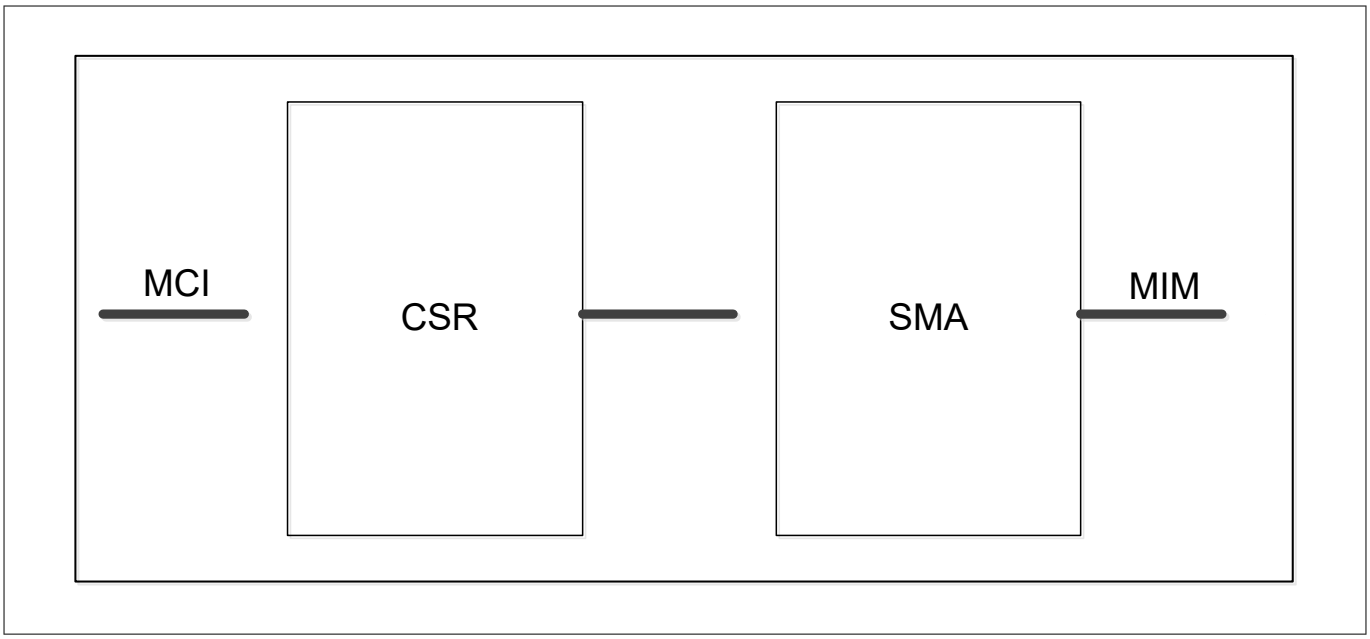


Figure 147 SMA Interface Block

14.4.3.18.1 Functional description

Note that HSPHY provides connection to physical pads of MDIO.

For MIM accesses, the maximum operating frequency of the MDC is 2.5 MHz, as specified in IEEE 802.3. In the XGMAC, the MDC clock is derived from the application clock, clk_csr_i, using a divider-counter. The divide factor depends on the clock range setting (CR field and CRS field) in the MDIO Single Command Control/Data register.

You can select the clock divide factor as recommended in Table 716 to meet IEEE specifications. However, if your system supports higher clock frequencies on the MIM interface, you can select a different divider.

Table 716 MDC clock generation with CRS = 0

CRS field of MDIO single command control/data	CR[2:0] field of MDIO single command control/ data	clk_csr_i frequency (MHz)	Generated MDC clock frequency
0	3’b000	100-150	clk_csr_i/62
0	3’b001	150-250	clk_csr_i/102
0	3’b010	250-300	clk_csr_i/122
0	3’b011	300-450	clk_csr_i/142
0	3’b100	350-400	clk_csr_i/162
0	3’b101	400-500	clk_csr_i/202
0	3’b110 - 3’b111	Reserved	Reserved

If you require higher MDC clock frequency, set the CRS field of the MDIO Single Command Control/Data register. Table 717 show the MDC clock generation based on CR field when CRS is set to 1.

Table 717 MDC clock generation with CRS = 1

CRS field of MDIO single command control/data	CR[2:0] field of MDIO single command control/ data	clk_csr_i frequency (MHz)	Generated MDC clock frequency
1	3’b000	-	clk_csr_i/4
1	3’b001	-	clk_csr_i/6
1	3’b010	-	clk_csr_i/8
1	3’b011	-	clk_csr_i/10
1	3’b100	-	clk_csr_i/12
1	3’b101	-	clk_csr_i/14
1	3’b110	-	clk_csr_i/16
1	3’b111	-	clk_csr_i/18

Table 718 shows the MDIO frame structure:

Table 718 MDIO clause 45 frame structure

Frame	Description
IDLE	The MDIO line is three-state; there is no clock on MDC
PREAMBLE	32 bits of continuous ones
START	Start of frame is 2’00
OPCODE	• 2’b00: Address • 2’b01: Write • 2’b10: Post-read increment address • 2’b11: Read
PHY ADDR	5-bit address select for one of 32 PHYs

(table continues...)

Table 718 (continued) MDIO clause 45 frame structure

Frame	Description
DEV ADDR	5-bit address select for one of 32 devices
TA	Turnaround • 2'bZ0: Read and post-read increment address • 2'b10: Write and address MDIO accesses Where Z is the tri-state level
DATA/ADDRESS	16-bit value: For an address cycle (OPCODE = 2'b00), this frame contains the address of the register to be accessed on the next cycle. For the data cycle of a write frame, this field contains the data to be written to the register. For read or post-read increment address frames, this field contains the contents of the register read from the PHY. • In address and data write cycles, the XGMAC drives the MDIO line during the transfer of these 16 bits • In read and post-read increment address cycles, the PHY drives the MDIO line during the transfer of these 16 bits

The frame structure for Clause 22 frames is also supported and shown in Table 719. The control register (MDIO Clause 22 Port) indicates which ports are Clause 22 and which are Clause 45. The Clause 22 frame format is shown in Table 719. For Clause 22 frames, only the lower five bits of the RA field (MDIO Single Command Address Register) are used as the frame's REG ADDR field.

Table 719 MDIO clause 22 frame structure

Frame	Description
IDLE	The MDIO line is three-state; there is no clock on MDC
PREAMBLE	32 bits of continuous ones
START	Start-of-frame is 2'01
OPCODE	• 2'b01 for Write • 2'b10 for Read
PHY ADDR	5-bit address select for one of 32 PHYs
REG ADDR	5-bit register address to select the register within each MMD
TA	Turnaround • 2'bZ0: Read • 2'b10: Write Where Z is the tri-state level
DATA	Any 16-bit value • In a write operation, the XGMAC drives MDIO • In a read operation, the PHY drives MDIO

In addition to normal single-read and single-write operations, the SMA also supports continuous write, continuous scan, and post-read increment address. These operations are described the following sections.

***Note**: The SMA module supports accesses to four PHY ports (addresses 0–3) only in the Continuous Scan and Continuous Write modes of operation (See Continuous write operation and Continuous scan operation).*

MDIO Preamble Suppression

XGMAC supports 32-bit preamble generation on all Read or Write operation. Some PHYs support reduction in the preamble content which can operate with only 1-bit preamble. When PSE (Preamble Suppression Enable, bit[30] of MDIO Single Command Control/Data Register) is set, preamble suppression mode is enabled and XGMAC drives only 1-bit preamble for all MDIO operations.

Single read/write operations

The software can perform a read or write operation to a specific register of any device of any port by programming SMA MDIO Single Command Address register and MDIO Single Command Control/Data register. If the PHY port is specified (in MDIO clause 22 port) as conforming to IEEE 802.3, Clause 45, the SMA automatically initiates the transfer, as the clause specifies. Otherwise, the SMA initiates a read/write transfer in the frame format specified in Clause 22. The SMA assumes all ports conform to Clause 45 by default.

The Single Read/Write command usually has the highest priority among the three possible operations in the SMA (single read/write, continuous write and continuous scan) followed by the continuous write operation. On receiving a single read/write command, the SMA suspends continuous write/scan operation at the end of the current write/read cycle (access to a register is over) and initiates this command.

The sequence for initiating this operation (programming guidelines) is as follows:

1. Program the MDIO Single Command Address register with the appropriate PHY port number, device ID and register address
2. Program the MDIO Single Command Control/Data register to initiate the required operation. If you are not performing 32-bit accesses to the CSR, make sure you write the Busy bit after writing the other bits, because setting the Busy bit triggers the command
3. The SMA automatically performs the address cycle (if not disabled by SADDR bit in MDIO Single Command Control/Data register) followed by the read/write/post-increment read data transfer cycle. The Busy bit remains high during this period
4. Wait for the Busy bit to clear (for polling) or for the Single Command Completion interrupt to be set
5. On receiving the interrupt (or when Busy bit is cleared), the given single-command operation is completed and the data bits in MDIO Single Command Control/Data register have valid data read from the required PHY Device register
6. The SMA is ready for another single command.

Continuous write operation

This command helps reduce the software's load when it must perform write operations to the same register in multiple devices or to multiple PHY ports. The software can give a single command and the SMA initiates writes to the target registers sequentially.

Continuous write operation is available for ports 0 to 31, and this command only accesses the devices enabled in the corresponding MDIO Port Device In-Use register. Thus, these registers must be configured properly for correct and fast completion of this command.

The programming sequence is as follows:

1. Program the MDIO Continuous Write Data register with the required data value
2. Configure the required operation in the MDIO Continuous Write Address register by programming the CPRT and CADDR bits
   - If CPRT is low, CADDR is taken as the common device address in all ports to which register write operations are performed
   - If CPRT is set high, CADDR is taken as the port number whose devices are all accessed in the register write operation. CADDR must have values of 0–3 in this mode: other PHY port numbers are invalid

3. Configure the register address in the CREGADDR bits and set the Continuous Write Busy bit. This step and Step 2 can be done in a single 32-bit write access to the register
4. The SMA schedules the continuous write operation to all the ports or devices. It performs the two-transfer cycle if the PHY port is configured as a Clause 45 port or a single cycle if it is a Clause 22 port
5. Wait until the Continuous Write Busy bit is cleared (for polling) or the Continuous Write Completion interrupt is set
6. When the SMA asserts the interrupt (or the Busy bit is cleared), the given command is complete and the SMA is ready for another continuous write command to a different port, device, or register

Continuous scan operation

This command helps reduce the load on the software for checking the status of the devices on each port. It also helps in automatic connectivity detection (hot-plug-in) of PHY devices (on a XENPAK module). This command usually has the lowest priority and can be enabled to always run in the background. This command performs read accesses to two specific registers (MDIO Single Command Address register and MDIO Port Connect/ Disconnect Status register) of each device on all ports to check their link and device statuses and to update the corresponding status registers in the SMA (MDIO Port Nx4P0 Link and MDIO Port Nx4P0 Alive status registers).

Continuous scan operation only applies to ports and this command only accesses the devices enabled in the corresponding MDIO Port Device In Use register. These registers must be configured properly for correct and fast completion of a scan cycle.

The programming sequence is as follows:

1. Configure the MDIO Port Nx4P0 Device In Use registers to enable the devices present in the PHY port. If a particular port does not have any PHY connected to it (that is, its current status is disconnected), enable only one of the devices (for example, PMDPMA) to check for connectivity
2. Enable the continuous scan to the required ports by setting the bits in the MDIO Continuous Scan Port Enable register. The SMA scans the ports in sequence
3. Enable the required interrupts in the MDIO Interrupt Enable register (bits 11:0)
4. The SMA scans all the ports and updates the corresponding status registers
5. The MDIO Port Connect-Disconnect Status register is updated as soon as the read access to any device register to that port is completed. The corresponding interrupt is asserted if any bit's value changes
6. The MDIO Port Nx4P0 Link and MDIO Port Nx4P0 Alive status registers are updated at the end of each scan cycle and if any bits in the corresponding ports change, the corresponding interrupt is asserted
7. A new sequence is started automatically from Port 0
8. The scan continues as long as any MDIO Continuous Scan Port Enable register bit is set

14.4.3.19 Flexible header feature

This chapter describes the support for additional bytes in the frame header. This is for non standard, proprietary frame formats only.

Some applications require proprietary information to be transported in the Ethernet packet in the form of a custom header (referred to as Flexible Header (FH) in this document).

When the FH feature is enabled in the controller, it is expected that all the packets passing through the MAC (both Rx, and Tx) will have the custom header.

The Flexible Header position can be any of the following within the Ethernet packet:

- Byte 0: Between preamble/SFD and DA field of Ethernet packet
- Byte 12: After MAC SA field in the Ethernet packet
- Byte 14 to 29: Any position in this range (inclusive)

This is programmed in FHL field of MAC_Flex_Hdr_Cfg register.

Flexible Header length can be programmed to any value from 1 to 8 bytes as per the FHL bits in MAC_Flex_Hdr_Cfg register.

The Flexible Header feature has programmable control bits (FHTX and FHRX in MAC_Flex_Hdr_Cfg register) to enable support in TX and RX path separately.

When enabled, proprietary Ethernet packets in which extra “custom” header bytes are present in Ethernet packets will be sent/received in the respective TX or RX path.

When disabled, normal Ethernet packets as specified in IEEE 802.3 standard will be transferred on the line. An additional programmable control bit is available to include FH support in MAC/Link-layer auto-generated packets like PAUSE packets, Preemption Verify/Response packets.

The content for this FH is taken from programmable registers (MAC_Flex_Hdr_High/Low).

All the existing offload features of the XGMAC IP like VLAN tagging, Packet filtering and queueing based on L2/L3/L4 headers, etc. will be supported when the FH is present. RMON/MMC counters, Giant Packet, Watchdog timeout, min/max packet size checks will ignore the presence of Flexible Header bytes.

14.4.3.19.1 Transmit path

When Flexible Header (FH) for Tx path is enabled, software must insert the header in the first 8 bytes of the Packet Buffer irrespective of programmed FH length and position, as shown in the below figure. Packet’s first buffer size) programming should account for extra 8 bytes. The packet length should accurately account for the actual length of the FH (corresponding to FHL field of MAC_Flex_Hdr_Cfg register). TxCOE offload will work as-is without being impacted by the presence of FH.

For the Link Layer auto-generated packets such as Pause frames, static custom header will be inserted in the packet only if EFLL bit is set. The VLAN/SA Insertion/Replacement/Deletion features are supported as-is independent of the presence of Flexible Header.

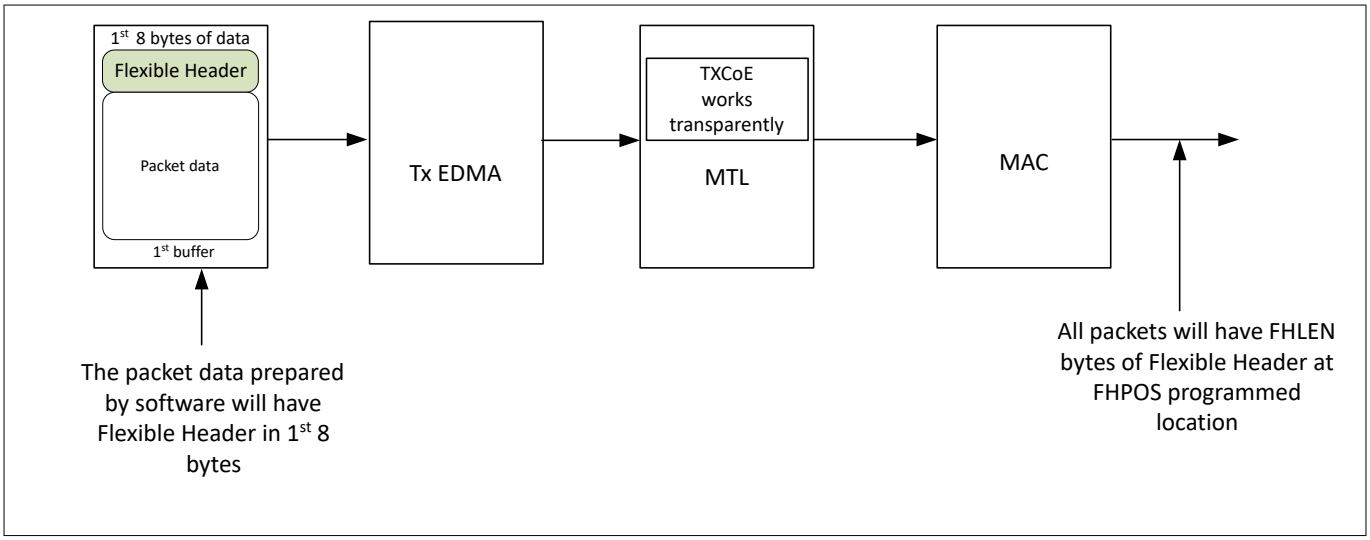


Figure 148 1 Tx path with flexible header high level block diagram

***Note**: First eight bytes of data will contain FH independent of the Buff1 addresses alignment.*

For example: If FH length is five bytes and Buff1 start address is 0x2, even then FH will occupy first eight addresses of the buffer and the actual packet data will start from address 0x2 + 0x8 = 0xA. (This format is applicable for FH in both Tx and Rx paths)

When FH is configured and FHTX bit is set to 1, the Transmit Descriptor **read format fields**’ interpretation will change as follows

TDES2[9:0]	reserved
TDES2[13:0]	Buf1 Length should include the 8 bytes of FH location
TDES2[29:16]	If B1L=0, then Buf2 Length should include the 8 bytes of FH location

TDES3[14:0]	Packet Length

***Note**: FH will be part of CRC computation and will not be part of L3/L4 checksum calculation/Tx MMC counters, and will be placed at the predetermined location in the packets transmitted on the line.*

14.4.3.19.2 Receive path

When FH for Rx path is enabled, ingress packets (including the ones consumed in the MAC for offloads) must have FH in the programmed location and length. Link layer auto-generated packets such as Pause packets are expected to have FH only when EFLL bit is set.

When FH is enabled, the MAC Receiver will always strip the Flexible Header present at the programmed start position (only exception being EFLL related packets). The stripped Flexible Header is sent to the Software as first 8 bytes of data (irrespective of FH length) before start of packet in first buffer of the packet, as shown in the figure below. The actual packet data will start from the given Buffer Address pointer plus 8 bytes, and the packet length, header length and 1st buffer size (if applicable) of the packet status includes the additional 8 bytes consumed by FH (independent of the programmed flexible header length).

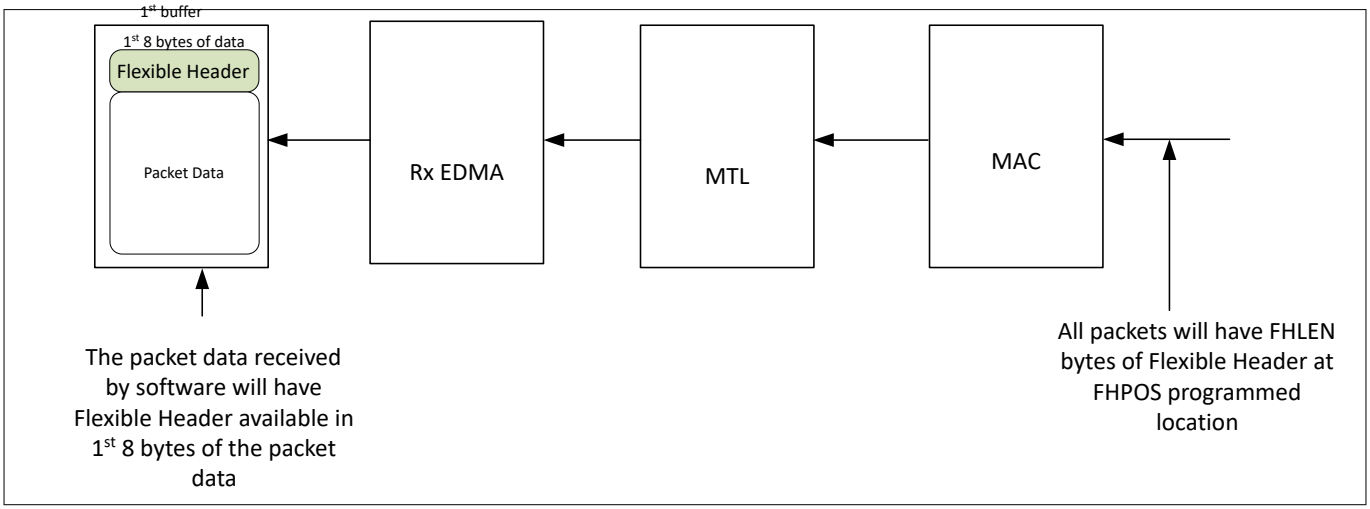


Figure 149 Rx path with flexible header high level block diagram

Filtering, Queueing, channel routing, RxParser and RxCoE functions, when enabled, work as intended with the FH feature enabled. In other words, these functions process the normal packets after the FH is stripped and therefore are not impacted when FH is enabled.

***Note**:*
- *The FH is stripped and discarded for all the packets that are not sent to the application (that is consumed in the MAC or dropped)*
- *In this mode, the FH bytes are blindly stripped (except for PAUSE). Far-end must not send any packets without the FH that are processed by any of the filtering functions. In case, such packets (without FH) are present in this mode, filtering must not be enabled*
- *When FHRX bit is set, packet length; PL field of (RDES3[13:0]) in Descriptor Write-back format will be reported as follows*

Table 720 Packet length status with FH enabled

Packet length with added 8 bytes of flexible header	PL [13:0]
16384	*1*

(table continues...)

Table 720 (continued) Packet length status with FH enabled

16385	*2*
16386	*3*
16387	*4*
16388	*5*
16388	*6*
16390	*7*
16391	*8*

When FH is configured and FHRX bit is set to 1, the Receive Descriptor write back fields will be written as follows to account for the eight additional FH bytes:

RDES2[9:0]	Header Length + 8
RDES3[13:0]	Packet Length + 8

14.4.3.19.3 Format of the FH to be written/read by the software

Flexible header will be written by software (SW) in the memory for the Transmit packets and will be made available to SW in RX path as described below. For example, if the Flexible Header is 4 bytes, and if 0x0a0b0c0d is to be programmed into a transmit packet then in the memory, the 8 Byte FH should be written as 0x000000000a0b0c0d.

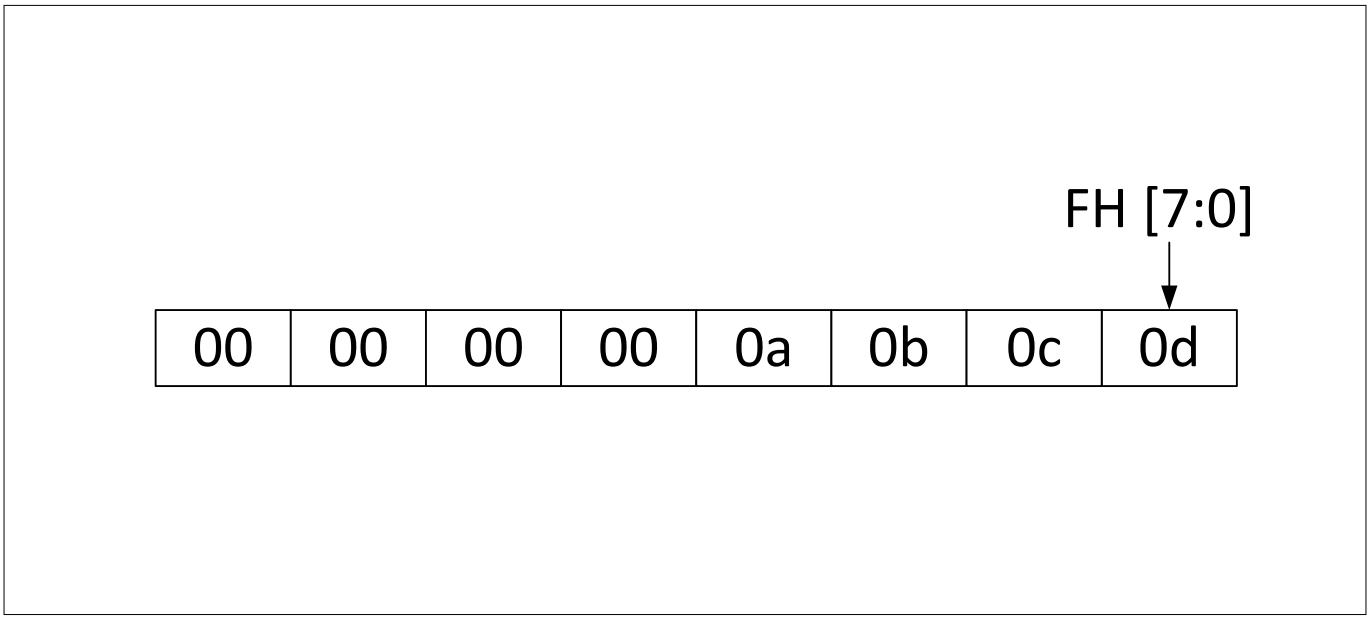


Figure 150 Tx packet FH to be written/Rx packet FH to be read by software

This FH will be written in SW as explained below

0a	0b	0c	0d	Addr offset=0x4			
DA3	DA2	DA1	DA0	4 Dummy FH bytes			
SA5	SA4	SA3	SA2	SA1	SA0	DA5	DA4
P	K	T	*	D	A	T	A

On the line side, if the FH position is Byte 12 (after DA-SA), while transmitting/receiving a packet on the line FH is embedded as shown below:

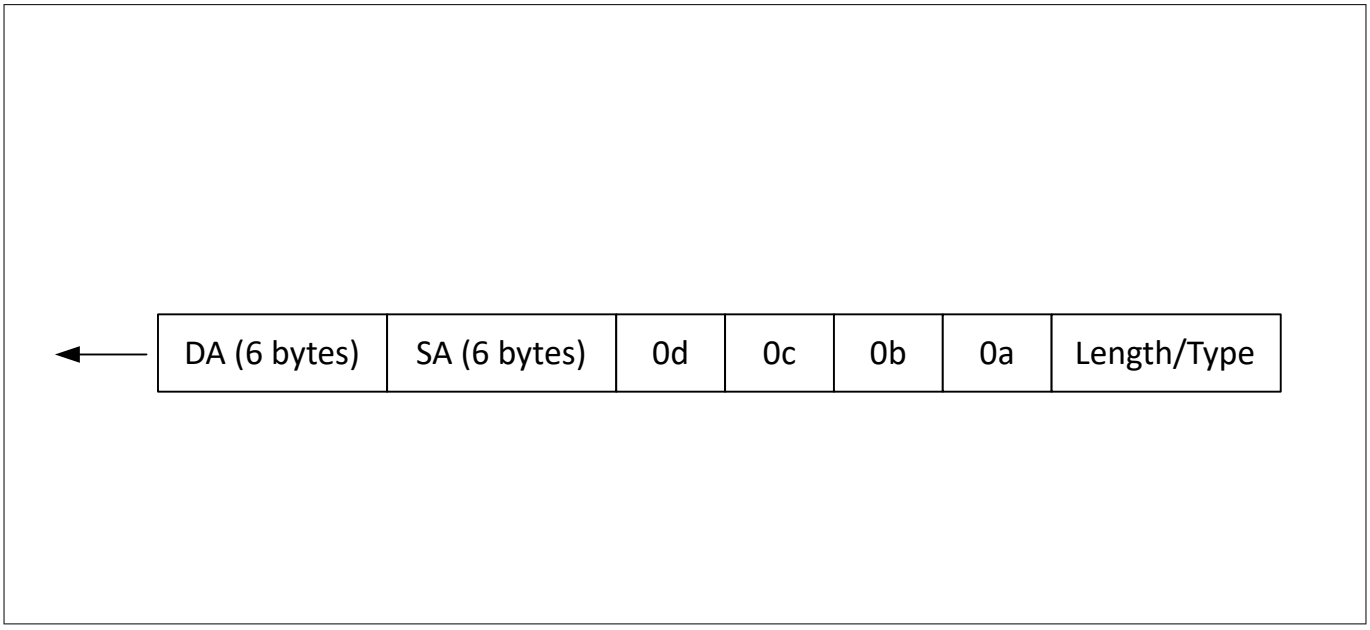


Figure 151 Packet transmitted/received on the line with FH inserted on GMII

FH [7:0] bytes will be the first flexible header bytes to go on the line/to be stored in the buffer address on Rx and Tx paths respectively, followed by subsequent higher bytes.

In case of MAC-generated / Link layer generated packets, the Static Flexible Header can be programmed in the MAC_Flex_Hdr_Low and MAC_Flex_Hdr_High registers respectively as follows,

FH of length 7 bytes and value 0x06050403020100 will be programmed as follows

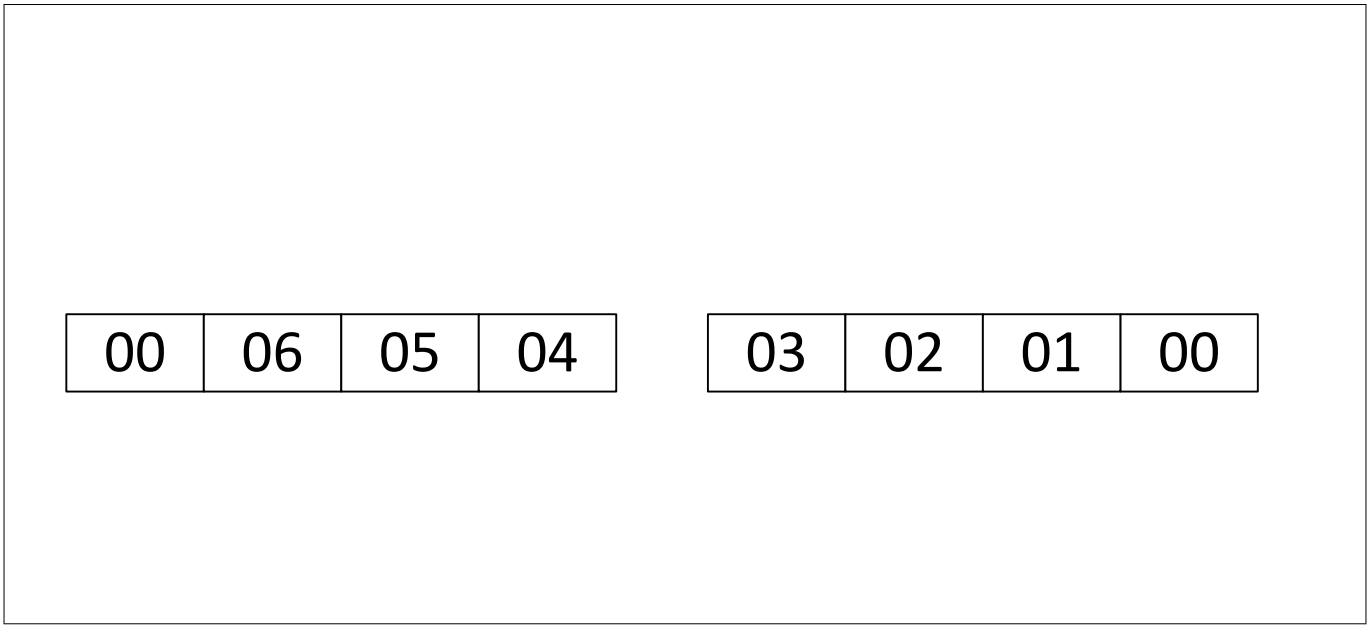


Figure 152 MAC_Flex_Hdr_High and MAC_Flex_Hdr_Low register programming

Here MAC_Flex_Hdr_Low [7:0] will be the first FH byte to go on the line followed by [15:8], [23:16] and [31:24] respectively.

If FH length is more than 4 bytes, then MAC_Flex_Hdr_High will follow MAC_Flex_Hdr_Low in similar way.

In this example, MAC_Flex_Hdr_Low register and only bytes 0x060504 from MAC_Flex_Hdr_High register will be sent on the line as MAC_Flex_Hdr_High[31:24] is not a part of the FH.

14.4.3.19.4 Restrictions

1. For Length packets, length value of Ethernet header (LT field) will not include FH length when FH is enabled
2. When FHRX is enabled and when EFLL=0, Unicast Pause frames will not be supported as the ingress PAUSE packet is identified (for the decision to strip FH or not) by the special multicast DA address only
3. In Rx path, when MAC_Packet_Filter[7:6], PCF field is set to 2'b10 or 2'b11, PAUSE packets can be forwarded to application. In this case, if EFLL bit is not set, then the first 8 bytes of such Pause packets transferred to the memory by RxDMA will contain dummy/invalid bytes(all-zero), even though FH was not stripped in such packets
4. Pause packets with special multicast DA (01:80:C2:00:00:01) is supported when FH is enabled (FHRX and EFLL is set) except for the following cases
   a. In Tx path, if the DMA fetches a Pause packet with special multicast DA, FH (written by the software) will be inserted at programmed position, even if EFLL bit is not set
   b. In Rx path, when EFLL bit is cleared to 0, and FH position is programmed at 0th byte location, FH will be stripped only if the first 6 bytes (FH and subsequent DA bytes) do not make up special multicast DA (01:80:C2:00:00:01) applicable for Pause frames

      For example: FH length = 3 bytes, FH position = 0th byte and EFLL=0

      1. If FH = 0x0180C2 and DA = 0x000102030405

         The 3 FH bytes and the 3 DA bytes, {0180c2,000102} != {0180C2000001}. So FH will be stripped from the received packet
      2. If FH = 0x0180C2 and DA = 0x000001020304

         The 3 FH bytes and the 3 DA bytes, {0180C2,000001} == {0180C2000001}. So FH will not be stripped from the received packet as 3 FH bytes and 3 DA bytes made up special multicast DA identified for Pause packets

In other words, if Flexible Header bytes and subsequent DA bytes are equal to Pause special multicast DA, then the packet will be recognized as a Pause packet when EFLL bit is set to 0 and hence the FH will not be stripped from such packets.

14.4.3.20 Multiple channels and queues support

This chapter describes how the XGMAC controller supports multiple queues and channels.

The topics are as follows:

Related information

Block diagram on page 1502
DMA on page 1502
Multiple queues support in the MAC on page 1504
Multiple queue support in the MAC transaction layer (MTL) on page 1504
TX Side Arbitration Between DMA and MTL on page 1509
Rx Side Arbitration between MTL and DMA on page 1509
Rx queue to DMA mapping on page 1509
Selection of tag priorities assigned to Tx and Rx queues on page 1512
Distribution of Rx Packets from MAC to Rx Queues on page 1513
Rx side routing from MAC to queues on page 1515

Receive Queue Selection Based on Filter Fail on page 1516
VLAN based DMA Mapping on page 1517

14.4.3.20.1 Block diagram

The XGMAC supports up to 8 queues and channels on Tx and Rx paths. The figure below shows the architecture of XGMAC with multiple queues and channels.

***Note**: In multiple channel/queue configurations, for :*

- *Half-duplex operations, enable only the Q0/CH0 on Tx*
- *Full-duplex operations, for single queue or channel, enable any queue/channel*

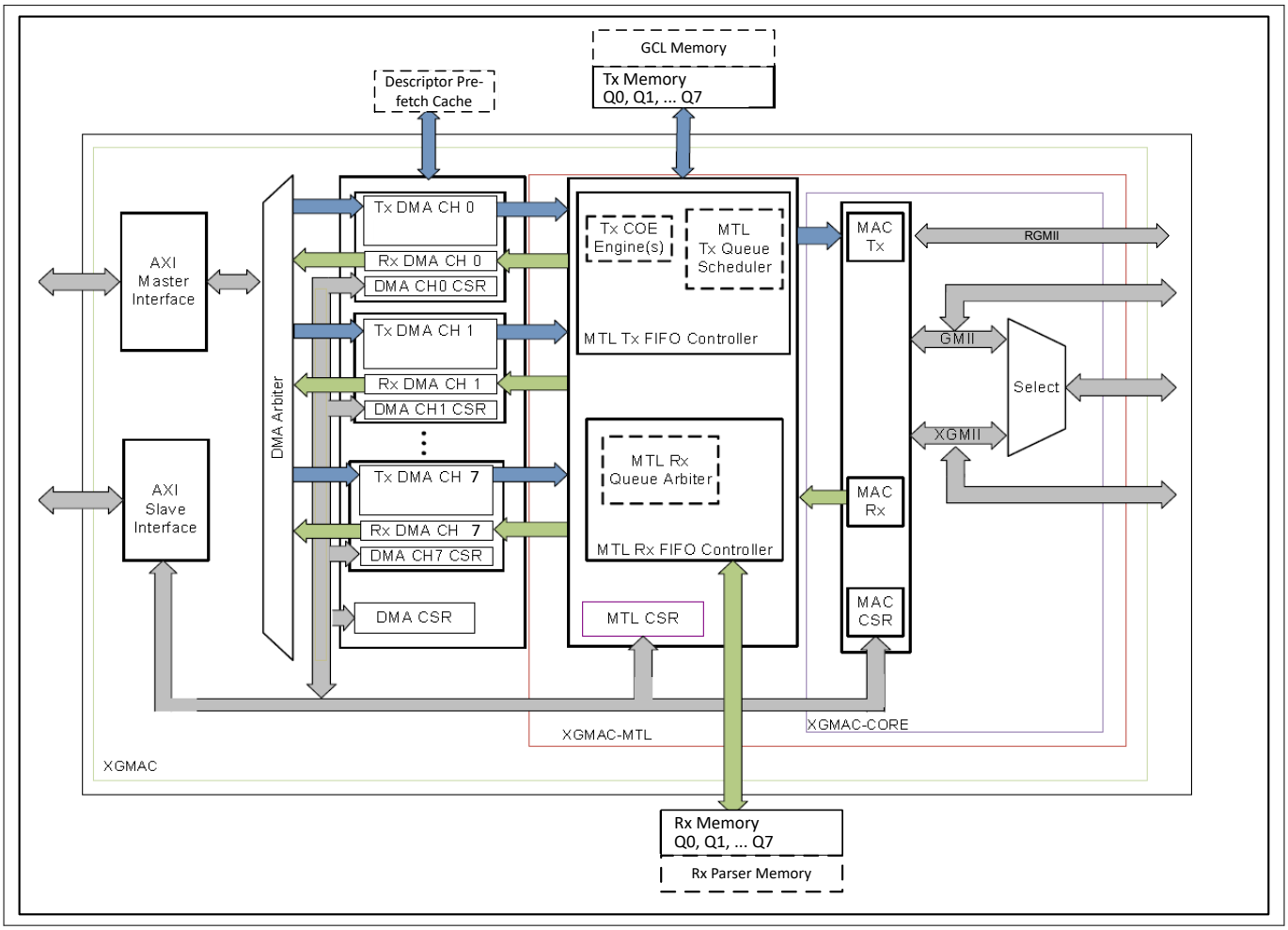


Figure 153 XGMAC with multiple channels and queues

14.4.3.20.2 DMA

This section describes how multiple channels are supported in the Transmit and Receive paths.

Transmit path

The XGMAC supports eight Tx channels. The fixed priority scheme is used in the Transmit path to arbitrate between the multiple Tx DMA channels to access the common AXI bus. In this scheme, the channel with highest

priority always wins the arbitration when it requests the bus. Table 721 provides information about the priority levels of Tx DMA channels.

Table 721 Fixed priority scheme for Tx DMA channels

Priority level	Channel
0 (low)	Channel 0
1	Channel 1
2	Channel 2
3	Channel 3
...	...
7 (high)	Channel 7

The Fixed Priority scheme does not necessarily cause “starving” of lower priority channels for the AXI bus. The AXI4 protocol capabilities and the Tx DMA characteristics are as follows:

- Read command requests from a Tx DMA for a burst of data transfers is driven on a separate AXI channel and gets executed in one clock cycle. The actual data transfer starts on a separate AXI channel only after the request is accepted and takes multiple clock cycles to complete with relatively large read access latencies
- AXI master is free to drive out subsequent requests even before the previous data transfers are completed
- The Tx DMA requests for another data transfer only after the data transfer for its previous request is completed and transferred to the MTL TxQ and space is available in TxQ to take in the next request
- The XGMAC can be programmed to control the maximum number of outstanding requests, Requested Burst sizes by DMA, and the burst sizes of each data transfer on AXI

Considering these, XGMAC can be programmed such that the requests of all the Tx DMA are placed and executed on the AXI bus before the data transfer of the first Tx DMA is completed and it raises the next request to the arbiter.

Receive path

The XGMAC supports up to eight Rx channels. In the Rx direction, the MTL Rx Controller selects the Rx DMA for which it is transferring or reading the data from the Rx FIFO memory. This scheduling is based on the programming done in the respective MTL_RxQ[x]_Control register.

Each Rx DMA indicates when it is ready to transfer data and the size of the burst-length (number of beats) that is has to transfer. The scheduler checks whether sufficient data (of requested burst length) is available to be transferred to these DMAs and then selects the Rx DMA that gets serviced using the programmed priorities.

Priority Scheme for Tx DMA and Rx DMA

The DMA arbiter performs the fixed priority arbitration between the Tx and Rx paths of DMA channels for accessing descriptors and data buffers. Descriptor accesses of a Tx or Rx channel is given higher priority when requests from both are active together.

WRR Arbitration Support for AXI Data-Reads on Tx DMA

The Weighted Round Robin arbitration is supported for Transmit DMA channel read access to AXI Read Data Channel. The weights corresponding to each Transmit DMA Channel is programmed in the indirect registers using the DMA CH Ind Ctrl and DMA CH Ind Data registers. The default weights in the indirect register correspond to Round Robin arbitration.

14.4.3.20.3 Multiple queues support in the MAC

You can enable or disable a Tx Queue by programming the TXQEN field MTL_TxQ(#i)_Operation_Mode register of corresponding queue.

The queuing is based on the CBS or SP algorithms.

You can enable or disable a Rx Queue by programming the RXQ#EN field of corresponding queue in the MAC_RxQ_Ctrl0 register. The following list explains how Rx queues are enabled based on the selected features:

1. Received tagged packets can be routed to any of the Rx Queues from Q0 to Q7
2. Each queue has a corresponding field (PSRQ fields in the MAC_RxQ_Ctrl2/3 registers) by which you can indicate the packets of specific VLAN priority values that should be routed/pushed into that queue
3. It should be ensured that each priority value can be routed to a specific queue within the set of enabled queues. That means bits corresponding to the same priority value should not be set in multiple PSRQx fields within the enabled queues, but the user can route multiple priorities into the same queue
4. Other types of packets such as untagged packets, multicast packets, PTP and AV control packets, are routed to specific Rx queues as programmed in the MAC_RxQ_Ctrl1 register.
5. A pause frame is generated when the flow control is triggered in any of the Rx Queues
6. AV is enabled for all selected Rx queues. The queuing is done based on the VLAN Tag priority for AV packets. The VLAN Tag priority should match the PSRQ field of the MAC_RxQ_Ctrl2 and MAC_RxQ_Ctrl3 registers. The Rx packets can also be routed to a particular DMA channel based on the DCS field of perfectly matched MAC address register. The AV control packets (tagged or untagged) are routed based on the AVCPQ field of the MAC_RxQ_Ctrl1 register. The PTP over Ethernet packets are routed based on the AVPTPQ field of MAC_RxQ_Ctrl1 register

14.4.3.20.4 Multiple queue support in the MAC transaction layer (MTL)

The XGMAC supports eight queues in the Tx direction and eight queues in the Rx direction. The number of active Tx queues and Rx queues can be configured independently by software. You can control a Tx queue through MTL_TxQ(#i)_Operation_Mode register. You can control an Rx queue through MTL_RxQ(#i)_Operation_Mode register.

Queue memory

All MTL Tx queues share one Tx memory and all Rx queues share one Rx memory with programmable individual queue sizes in blocks of 256 bytes. The application can program the Tx queue size in the TQS field of MTL_TxQ(#i)_Operation_Mode register and the Rx queue size in the RQS field of the MTL_RxQ(#i)_Operation_Mode register.

The XGMAC supports the total Tx memory size of 32 KB and total Rx memory size of 32 KB.

Traffic classes and transmit queues

The XGMAC controller supports independent configuration of transmit queues and traffic classes (TC), where you can select up to eight traffic classes and up to eight transmit queues. You can program the bandwidth required for each traffic class by programming the MTL_TCn_Quantum_Weight register corresponding to a traffic class.

A queue must be mapped to a traffic class by programming the Q2TCMAP field in the MTL_Txn_Operation_Mode register for a particular queue.

The scheduling between traffic classes is determined by the scheduling algorithm implemented in the transmit scheduler.

***Note**: All Queues within a traffic class are selected in a round robin fashion (when packets are available) when the traffic class is selected by the scheduler for packet transmission. This is true for any of the scheduling algorithms.*

The following scheduling algorithms are supported:

- Weighted Round Robin
- Strict Priority

Weighted Round Robin

In Weighted Round Robin (WRR) algorithm, each traffic class is assigned a weight based on the percentage of configured bandwidth. All traffic classes are serviced in the round-robin order according to the programmed weights. Large packets get more bandwidth in this algorithm.

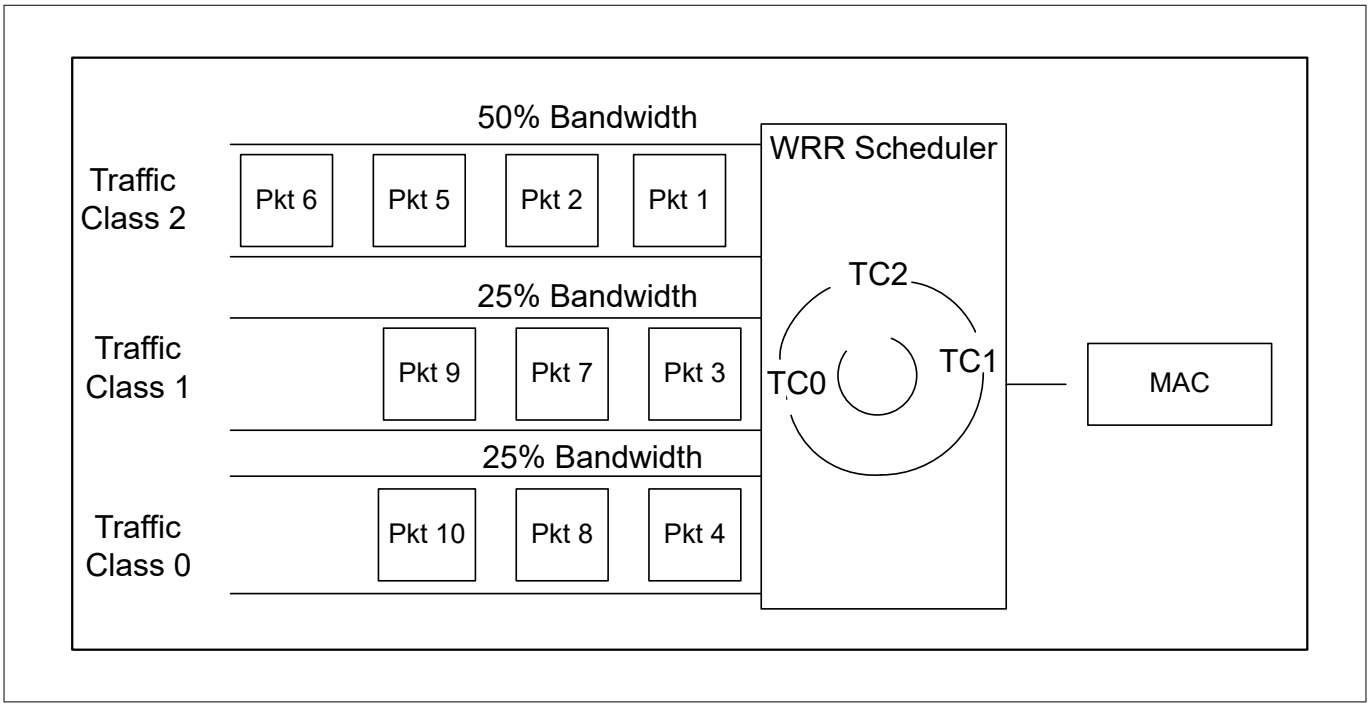


Figure 154 Weighted Round Robin Example

In Figure 154, Traffic Class 2, Traffic Class1 and Traffic Class 0 are configured for 50%, 25%, and 25% bandwidth respectively. The corresponding weights are 2, 1, and 1. Before the WRR scheduler is started, Traffic Class 2 has four packets; Traffic Class 1 has three packets; and Traffic Class 0 has three packets. The packets are transmitted in the following order:

Table 722 Packet transmission order

Packet sequence	Traffic class number
Packet 1	Traffic Class 2
Packet 2	Traffic Class 2
Packet 3	Traffic Class 1
Packet 4	Traffic Class 0
Packet 5	Traffic Class 2
Packet 6	Traffic Class 2
Packet 7	Traffic Class 1

(table continues...)

Table 722 (continued) Packet transmission order

Packet sequence	Traffic class number
Packet 8	Traffic Class 0
Packet 9	Traffic Class 1
Packet 10	Traffic Class 0

Strict priority

In the strict priority algorithm, Traffic Class 0 has the lowest priority and the priority increases with the Traffic Class number. For example, in a four-TC configuration, TC3 has the highest priority, followed by TC2, TC1, and finally TC0.

The highest priority Traffic Class is serviced first. If there is no packet in the highest priority Traffic Class, then the next priority Traffic Class is serviced.

Audio Video Bridging

Overview of Audio Video Feature

XGMAC supports the AV data transfer in all speed modes. The AV feature enables transmission of time-sensitive traffic over bridged local area networks (LANs). The following standards define various aspects of the AV feature implementation:

- IEEE 802.1Qav-2009: Allows the bridges to provide time-sensitive and loss-sensitive real-time audio video data transmission (AV traffic). It specifies the priority regeneration and controlled bandwidth queue draining algorithms that are used in bridges and AV traffic sources
- IEEE 802.1Qat-2009: Allows the network resources to be reserved for specific traffic streams traversing a bridged local area network
- IEEE 802.1AS-2020: Specifies the protocol and procedures used to ensure that the synchronization requirements are met for time-sensitive applications such as audio and video across bridged and virtual-bridged LANs consisting of LAN media where the transmission delays are fixed and symmetrical. For example, IEEE 802.3 full-duplex links include the maintenance of synchronized time during normal operation followed by addition, removal, or failure of network components and network reconfiguration

Transmit Path Functions

The Transmit path of Queue 0 supports the strict priority algorithm, and it is used for best-effort traffic. For a queue, the strict priority algorithm determines that a packet is available for transmission if the queue contains one or more packets. When the threshold mode for MTL Tx FIFO is enabled, the strict priority algorithm determines that a packet is available for transmission if the queue contains a partial packet of size equal to the programmed threshold limit.

The Transmit paths of additional queues support traffic management by using the credit-based shaper algorithm. For a queue, the credit-based shaper algorithm determines that a queue is available for transmission if the following conditions are true:

- The queue contains one or more packets.
- The credit for the queue is positive as per the algorithm.

By default Credit Based Shaper (CBS) is disabled for all queues. CBS support for the queues must be enabled from higher Queue index to lower Queue index, that is enable CBS from Queue 7 before Queue 6 and so on.

Each Transmit DMA has a separate descriptor chain for fetching the transmit data. The Transmit channel that gets the access to the system bus depends on the DMA arbiter.

The Transmit path has a shared FIFO (MTL layer) for each queue, as shown in Figure 131. The data fetched by the DMA is put in the respective part of the FIFO. The MTL Tx Queue Scheduler controls which part of the FIFO data is transmitted by the MAC. If the credit-based shaper algorithm is enabled for a queue, the corresponding queue is selected for transmission if the following conditions are true:

- If the packet is available in the channel and has a positive or zero credit.
- If the higher priority queue has no packet waiting in the FIFO.

If the credit-based shaper algorithm is disabled for all queues, the packet to be transmitted from a queue is selected based on the priority scheme described in Figure 154.

Receive Path Functions

XGMAC having 8 RxQs allows user to demultiplex the received data to send the packets into separate receive channels.

To differentiate between the AV and non-AV traffic, the MAC provides a status that indicates if it is an AV packet and its corresponding VLAN Priority tag value. This status is updated in the Extended Status field of the Receive descriptor. All received packets with the EtherType field of 0x22F0 are detected as AV packets. The AV packets can be of the following two types:

- AV data packets: The AV data packets are always tagged. The tagged AV data packets are received based on the programmed priority value. To specify the queue to which an AV packet with a given priority must be sent, program respective priority bit in PSRQ(#i) field of MAC_RxQ_Ctrl2 or MAC_RxQ_Ctrl3 register corresponding to the queue.
- AV control packets: The AV control packets can be either tagged or untagged. The untagged AV control packets are received on Queue 0 by default. To receive these packets on any other queue, you can program Bits[2:0] of the MAC_RxQ_Ctrl1 register. Similar to the AV data packets, the tagged AV control packets are received based on the programmed priority value.

In addition to the AV packets, you can receive the untagged PTP packets on any queue. By default, the PTP packets (tagged or untagged) are received on Queue 0. To receive these packets on any other queue, you need to program Bits[6:4] of the MAC_RxQ_Ctrl1 register.

Credit-Based Shaper Algorithm

The MTL Queue Scheduler uses the credit-based shaper algorithm to arbitrate the AV traffic in all AV enabled Traffic Classes and the legacy Ethernet traffic in Traffic Class 0. You can program the additional queues to use the credit-based shaper algorithm.

The following sections provide information about configuration of credit-based shaper algorithm.

Credit Value

The credit value is accumulated every transmit clock cycle. The credit to be added or subtracted per cycle can be fractional based on the required idleSlope and sendSlope values. The credit for a TC builds up only when the packet is available but it cannot be transmitted because the MAC is sending a packet from another TC. The XGMAC supports a mode (CC field of ETS control register) in which the credit can build up in advance for a TC which has no outstanding packets. This enables sending a burst of high priority traffic in a TC as soon as the data is available. When the CC field is reset, the accumulated credit parameter in the credit-based shaper algorithm is set to zero if there is positive credit, and there is no packet to transmit in a TC. The credit does not accumulate when there is no packet waiting in a queue and other queues are transmitting. When set, the accumulated credit parameter in the credit-based shaper algorithm is not reset to zero if there is positive credit and no packet to transmit in a queue. The credit accumulates even when there is no packet waiting in a queue and other queues are transmitting.

idleSlopeCredit and sendSlopeCredit Values

The software must program the idleSlopeCredit and sendSlopeCredit values. The programmed values should be the credit accumulated or drained per clock cycle scaled by 1024, such as, 320 x 1024 = 327680 and 480 x 1024 = 491520. In addition, the software must program the hiCredit and loCredit values, scaled by 1024, to adjust for scaling of the idleSlopeCredit and sendSlopeCredit values.

Programming example for idleSlope, sendSlope, hiCredit, and loCredit

The idleSlope, sendSlope, hiCredit and loCredit is computed as below:

idleSlope = BW * portTransmitRate

sendSlope = idleSlope – portTransmitRate

hiCredit = maxInterferenceSize * BW * 8bits

loCredit = maxFrameSize * (sendSlope/portTransmitRate) * 8bits

The portTransmitRate = 4, 8 and 32 for 10/100Mbps, 1 / 2.5Gbps and 5Gbps modes respectively.

maxInterferenceSize is:

-For highest priority queue, it is maximum frame size supported by the MAC

-For lower priority queue, it is the sum of maximum frame sizes supported by higher priority queues and maximum frame size supported by the MAC

It can be calculated using (total number of queues on which traffic is sent - the particular queue number) * maxFrameSize

For example: if traffic is sent on 8 queues, the maxInterference for respective queue will be:

maxInterferenceSize for Q7 = (8 - 7) * maxFrameSize

maxInterferenceSize for Q6 = (8 - 6) * maxFrameSize

maxInterferenceSize for Q5 = (8 - 5) * maxFrameSize

The above formulae's can be used for determining the values to be programmed in the per queue AV related MTL registers.

For example: In 1 Gbps mode, if TC7 (highest priority) is allocated BW=60%, TC6 (lower priority) is allocated BW=25% and Q0 is best effort queue. The maximum frame size supported by MAC in each of the queues is 1500 bytes.

TC7 Programming

idleSlope: (60/100) * 8 = 4.8 (i.e., program (4.8 * 1024) = 0x1333 in the idleSlope register)

sendSlope: 4.8 – 8 = -3.2 ((3.2 * 1024) = 0xCCD (value programmed in the sendSlope register)

hiCredit: 1500 * (60/100) * 8 = 7200 (i.e., program (7200 * 1024) = 0x708000 in the hiCredit register)

loCredit: 1500 * (-3.2/8) * 8 = -4800 ((4800 * 1024) = 0x4B0000, program twos complement which is 0x1FB50000)

TC6 Programming

idleSlope: (25/100) * 8 = 2 (i.e., program (2 * 1024) = 0x800 in the idleSlope register)

sendSlope: 2 – 8 = -6 ((6 * 1024) = 0x1800 (program this value in the sendSlope register)

hiCredit: 1500 * 2 * (25/100) * 8 = 6000 (i.e., program (6000 * 1024) = 0x5DC000 in the hiCredit register)

loCredit: 1500 (-6/8) * 8 = -9000 ((-9000 * 1024) = 0x8CA000, program twos complement which is 0x1F736000)

The 2's complement value should be programmed for loCredit field because it is always a negative value.

Bandwidth Status

The hardware maintains the status of the actual bandwidth consumed by each higher priority queue in the CBS status registers. This enables the software to estimate the average bandwidth consumed by numerically higher traffic classes as compared to the reserved bandwidth.

The CBS status register gives the average number of bits transmitted during the previous programmed slot interval (1, 2, 4, 8, or 16 slots of 125 µs) in a queue. The status register is updated even if the credit-based shaper algorithm is not enabled for a queue. The number of slots over which the average bits transmitted per slot are computed is programmed in Bits[6:4] of the Porti_MTL_TCQ[i]_TC[i]_ETS_Control register of the respective traffic class. For example, if you have programmed two slots, the average bits are computed over slot numbers 0–1, 2–3, 4–5, and so on.

The value programmed in the idleSlopeCredit register of a queue is proportional to the bandwidth reserved for the queue. The software can allocate any bandwidth that is not used by the higher priority queue to the reserved bandwidth of the lower priority queue.

A lower priority queue, which is using the credit-based shaper algorithm, cannot use the unused reserved bandwidth of any higher priority queue that is using the credit-based shaper algorithm. However, a lower priority queue, which is using the strict-priority algorithm, can use the unused reserved bandwidth of any higher priority queue that uses the credit-based shaper algorithm. For example, Queue 1 and Queue 2 use the credit-based shaper algorithm (with reserved bandwidth of 50% and 25%, respectively) and Queue 0 uses the strict-priority algorithm. If Queue 1 uses only 40% of reserved bandwidth, the remaining 10% is used by Queue 0. Queue 2 cannot exceed the reserved bandwidth of 25%.

14.4.3.20.5 TX Side Arbitration Between DMA and MTL

As the number of Tx DMA channels is always equal to the number of Tx Queues in the MTL, they are always mapped directly. For instance, each Tx DMA pushes data into its respective Tx Queue assigned to it.

14.4.3.20.6 Rx Side Arbitration between MTL and DMA

After completing the current packet processing, the DMA Channel controller fetches the next descriptor. Once the descriptor fetching is complete, it evaluates the amount of data to be transferred to the Rx Buffer based on the programmed PBL and Rx Buffer Length, and accordingly requests the MTL to transfer the data.

After servicing the current request, the MTL Rx arbiter selects a Rx Queue based on the arbitration scheme (RAA field of the MTL_Operation_Mode register) and the weights programmed in the MTL_RxQn_Control Register. Arbitration is done among Queues that have the packet to be transferred to a DMA, which are ready to transfer data. For example, if the DMA is in the process of fetching or closing a descriptor, it is not in the “ready” state. It is ready to transfer data only after it reads in a valid descriptor with non-empty buffer locations. Once the Rx Queue is selected, a complete frame/PBL (programmable burst length) amount of data (based on the RXQ_FRM_ARBIT field of the MTL_RxQn_Control register) is read out from that Queue and routed to the Rx DMA Channel based on the Rx Channel selection criteria defined in the "Rx queue to DMA mapping" section.

Related information

Rx queue to DMA mapping on page 1509

14.4.3.20.7 Rx queue to DMA mapping

The packets in the Rx Queue can be routed to any one of the multiple DMA channels by programming the following registers:

- MTL_RxQ_DMA_Map0 Register (for queues 0, 1, 2 and 3)
- MTL_RxQ_DMA_Map1 Register (for queues 4, 5, 6 and 7)

Two types of Rx Queue to DMA mapping is possible through programming.

Static Mapping

In this mode, all the packets of a Rx Queue are routed to a specific DMA channel. For example, all the packets from Rx Queue 0 can be routed to a DMA channel by programming Q0MDMACH (bit[2:0]) and Q0DDMACH (bit[7] = 0) of the Port j MTL RxQ DMA Map0 register.

Similarly, packets from other Rx Queues can be routed to any DMA channel by programming register fields corresponding to each Queue.

Dynamic (per packet) mapping

In this mode, the destination DMA channel of a packet being read from a Rx Queue is not constant but decided independently for each packet. For example, if the Q1DDMACH bit of the MTL_RxQ_DMA_Map0 register is set, the static mapping is disabled for Rx Queue 1 and the value in Q1MDMACH is ignored. The destination DMA channel is decided by the controller receiver for each packet, depending on the following, in the decreasing order of priority.

1. L3-L4 filter Based DMA Selection

   The TCP/UDP and IP header fields of the received packet are matched against the corresponding values programmed and enabled for comparison in the MAC_L3_L4_Controlx register. If the match is successful, the DMA channel number programmed in the DMCHN field of the MAC_L3_L4_Controlx register is selected as the destination DMA channel number provided DMCHEN bit of the same register is set.

If none of the L3-L4 Registers give a comparison match, the next category, makes the DMA channel selection

2. Extended VLAN Based DMA Selection

   The MAC uses the VLAN Tags of the received frame to determine the destination DMA Channel number. For more details, see the "VLAN based DMA mapping" section.

   If none of the VLAN filters result in a comparison match, the next category Ethernet DA-Based DMA Selection, makes the DMA Channel selection.

   Routing is applicable only if the entire VLAN Filter has passed. Routing is only based on Perfect Filter Result. Each perfect filter has a DMA Channel Enable and a DMA Channel Number field which can be programmed. Only if the DMA Channel Enable bit is set, routing is applicable for that filter.

   The frame is routed to the DMA Channel of the smallest Matching Filter, if enabled. If the DMA Channel number of the filter is not enabled, the frame gets routed to Channel 0. For example, if the VLAN tag of frame matches filters 7, 3, and 1, the MAC checks if the DMA Channel Number of Filter 1 is enabled through programming. If yes, the frame is routed to the programmed value; otherwise it is routed to the DA filter based DMA channel or to channel 0. When the inverse filter is enabled; it is routed to the DMA channel number of the least mismatched filter, if enabled.

   If VLAN Hash Filter is also enabled, it is used to determine the Filter result only. For non-tunneled packets, routing depends on the enabled Perfect Filters. If none of the perfect filters are enabled or if all of them are bypassed, the VLAN Filter based routing does not take place. If all the perfect filters give a fail result and the Hash Filter has passed, the VLAN Filter result is a pass but routing is based on DA Based Addressing or to Channel 0. Similar behavior is seen when inverse Filtering is enabled as well.

   The overall VLAN Filter is produced and it decides if the frame can be dropped or forwarded. If the frame gets a fail result from the VLAN Filter, then it is dropped if VTFE = 1 & RA = 0.

   RDES0 cannot be used to give the outer/& inner VLAN Tags of the outer header. If stripping is enabled, the matching tag value is lost as it cannot be passed through the status. The application needs to be cautious before enabling stripping.

3. Ethernet DA-Based DMA Selection

   The DA address of the received packet is compared against the programmed DA values in MAC Address Registers. If the address matches any of the programmed values, the corresponding DCS field determines the destination DMA channel number

   If none of the previous operations are able to make a successful match/decision, then the packet is routed to DMA Channel 0 by default. A summary of the filtering result and the DMA channel selection in this mode is given in Table 723

Table 723 L3L4 filter results

L2/VLAN filter match	MAC_Packet _Filter[IPFE]	L3/L4 filter match	Action on packet	DMCHEN	DMA channel assigned
1	0	Fail	Forward	X	MACAddr-DCS[1)]
1	X	Fail	Drop	X	MACAddr-DCS
1	X	Pass	Forward	1	L3L4-DMCHN[2)]
1	X	Pass	Forward	0	MACAddr-DCS
1	X	Bypass	Forward	X	MACAddr-DCS
1	X	Bypass	Drop	X	MACAddr-DCS
1	1	Fail	Drop	X	MACAddr-DCS
0	X	Pass	Drop	1	L3L4-DMCHN
0	X	X	Drop	X	0

1) MACAddr-DCS: Field of the MAC_Addressx_High register that gave a DA match
2) L3L4-DMCHN: Field (DMCHN) of the MAC_L3_L4_Controlx register that gave a match

L2/VLAN Filter match means the combined result of the Ethernet DA, SA, VLAN perfect & hash match filters. When all of the enabled filters give a match, then this is taken as “1”, otherwise “0”

L3-L4 Filter match has following three values:

- Fail: All enabled L3L4 filter failed
- Pass: Any of enabled L3L4 filter gave a match
- Bypass: No L3L4 filter enabled for received packet type

Action on Packet: DMA channel number is indicated even if packet is “Drop” because of packet may reach the application because of MAC_packet_Filter[RA] = 1

DMCHEN: Field of the MAC_L3_L4_Controlx register that gave a match

***Note**: You can also assign multiple queues to one DMA channel. In this case the order of the packets that gets to the DMA channel from different Queues are affected and does not follow the Scheduling scheme programmed in the RAA field of the Port j MTL Operation Mode register. For more details, see the "Tx side arbitration between DMA and MTL" section.*

Related information

VLAN based DMA Mapping on page 1517
TX Side Arbitration Between DMA and MTL on page 1509

Broadcast/Multicast Packet Duplication

Broadcast/Multicast Packet Duplication supports sending and receiving Broadcast/Multicast packets to multiple DMA channels.

In a virtual system that supports multiple guest OS's (for instance, each OS owning a DMA), where each DMA acts as an independent Ethernet Node, there can be requirement to route the received multicast/broadcast packets to all/multiple DMA channels. This is implemented by reading the packet from the same queue multiple times and route the same packet to the specified multiple DMA channels.

To support multiple DMA channels through software control, program the Duplication Selection of the DA (DDS field of MAC_Extended_Configuration register).

When DDS is 0_B, the DCS value holds the binary number of the destination channel (unique destination with no duplication).

When DDS is set to 1, the XDCS field in the MAC_DChSel_IndReg(#i) Register is taken as the destination Rx DMA for the packet matching the associated DA/SA filter register. The XDCS field is specified as one-hot format where each bit when set, represents the RxDMA channel to which the packet should be transferred. In other words, when multiple bits are set, the packet is duplicated and distributed to all the corresponding RxDMA channels.

Programmable Unicast Packet Duplication

Program the UDC field, bit[31] of the MAC_RxQ_Ctrl4 register to enable the unicast packet duplication.

The unicast packet duplication occurs only when the following conditions are met:

- Highest receive queue is selected and enabled in the configuration
- DA, SA, and VLAN filters pass
- More than 1 bit is set in the DMA channel number that is specified in the XDCS field of the MAC_DchSel_IndReg(#i) register that matches the unicast designation address of the received packet

The XGMAC controller performs packet duplication by default when the received packets have multicast or broadcast destination address.

The following table shows various scenarios when duplication occurs and when it does not.

Table 724 Unicast packet duplication scenarios

UDC	Unicast packet received	DDS = 1 & multiple bits of XDCS field of MAC_DChSel_IndReg(#i) matching the Packet DA, are set to 1	Packet mapped to highest RxQ	Duplication	Behavior
X	Yes	No (either of the condition is false)	No	No	If DDS =0, then packet routed to RxDMA indicated by DCS field of MAC DA register else as per lowest bit set in XDCS field
0	Yes	Yes	X (Do not care)	No	Packet is only routed to DMA corresponding to lowest numbered bit set to 1
1	Yes	Yes	Yes	Yes	Packet routed to multiple RxDMA as programmed

Packet flush for Duplicated packets

When RPF bit of DMA_CH(#i)Rx_Control remains set and the DMA is re-started by the software driver the packets residing in the Rx Queues that were received when the DMA is stopped, still gets flushed out when the packet is not enabled for duplication to multiple Rx DMA.

When packet is enabled for duplication to multiple Rx DMA, the packet will not be flushed by Rx DMA in active state irrespective of RPF bit of DMA_CH(#i)Rx_Control register.

14.4.3.20.8 Selection of tag priorities assigned to Tx and Rx queues

In the Receive direction, the VLAN Tag priorities can be assigned to Rx Queues by programming the PSRQ# field in the corresponding MAC_RxQ_Ctrl2/3 registers. The bit corresponding to the VLAN Tag priority can be set in the PSRQ field for assigning that priority to the corresponding Rx Queue. For example, to assign VLAN Tag priority of 3 to Rx Queue 0, set bit [3] in the PSRQ0 field of the MAC_RxQ_Ctrl2 register. More than one Rx Queue cannot be assigned the same VLAN Tag priority. However, more than one VLAN Tag priorities can be assigned to the same Rx Queue. A maximum of eight priority queues are supported as there are only eight priorities available (3-bit field in the VLAN Tag). A programmable PRQSO field of the MAC_RxQ_Ctrl5 register provides the flexibility to select the start offset of the priority queue.

***Note**:*

- *When PRQSO is programmed to a non-zero value, the highest valid PSRQ(#i) field is only up to the one corresponding to highest Receive Queue selected in the configuration. Otherwise, the received packets corresponding to programmed VLAN Tag priorities in the unmapped PSRQ(#i) field are routed to Receive Queue 0. For example, when eight Receive Queues are selected in the configuration and PRQSO is programmed to 4, the highest PSRQ(#i) that can be programmed is PSRQ3*
- *The following requirements of PRQSO = 0 apply even when it is programmed to a non-zero value:*
  - *The Express packets priority queue and Preempt packets special queues must not be mapped to the same Receive Queue*
  - *The Preempt packets priority queue and Express packets special queues must not be mapped to the same Receive Queue*

The received VLAN Tagged Rx packet is routed to the Rx Queue that has the VLAN Tag priority match.

14.4.3.20.9 Distribution of Rx Packets from MAC to Rx Queues

The XGMAC controller provides multiple Rx queues to support classification and distribution of ingress packets. The MAC receiver pushes the ingress packets to the Rx queues based on packet type (tagged, untagged, multicast, PTP, AVB and so on), tag priorities, MAC packet filter results, preemptable or express packet, and so on. The distribution is controlled by the settings of the MAC_RxQ_Ctrl* and MAC_Packet_Filter register fields.

Based on the packet filter settings, the MAC receiver decides to either drop or forward the Ingress packet. However, the results of Layer 3 and Layer 4 filtering are not considered for the distribution of ingress packets into the Rx queues. Only the results of DA/SA/VLAN filters mentioned in Source address or destination address filtering and Extended VLAN Filtering are applicable.

Filtering of the broadcast packets is exclusively controlled by the DBF field, bit[5] of the MAC_Packet_Filter register. The filtering of multicast packets is controlled by the PM field, bit[4] of the MAC_Packet_Filter register, in addition to the DA/SA match.

By default, packets that fail any of the DA/SA filters, are dropped by the MAC Receiver. However, packets that fail the DA/SA/VLAN filter can still be forwarded to the application when the RA field of the MAC_Packet_Filter register is 1. Even when RA field is 0, VLAN filter fail packets can be forwarded if the VTFE field of the MAC_Packet_Filter register is 0.

The ingress packets go through the following operations in the decreasing order of priority.

1. RA = 0: Ingress packets are dropped if any one of the following condition is true
   - DA/SA filter fail
   - VLAN filter fail and VTFE field of the MAC_Packet_Filter register is 1
2. RA = 1: Ingress packets that fail the DA/SA filter are also forwarded to the Rx queue as mentioned in Table 725

Table 725 Routing DA/SA Failed Packets to Rx Queue

Packet Type	TSN Type	Destination Rx Queue Determined By
Broadcast/Multicast	Express	if MFFQE = 1, MFFQ; else if VFFQE = 1, VFFQ; (only for tagged packets) else RxQ0
Unicast	Express	if UFFQE = 1, UFFQ; else if VFFQE = 1, VFFQ; (only for tagged packets) else RxQ0
All	Preemptable	RQ field

3. RA = 1 and VTFE = 0: Ingress tagged packets that pass the DA/SA filter but fail the VLAN filter, are forwarded to the Rx queue as mentioned in Table 726.

Table 726 Routing DA/SA Pass But VLAN Filter Fail Packets to Rx Queue

TSN Packet Type	Destination Rx Queue Determined By
Express	if VFFQE = 1, VFFQ; else RxQ0
Preemptable	RQ field

4. Ingress packets that pass the DA/SA/VLAN filters are forwarded to the Rx queue

- Table 727 shows the decreasing order of priority when OMCBCQ field of the MAC_RxQ_Ctrl1 register is 0

Table 727 Priority Routing of Filter Pass Packets When OMCBCQ = 0

Packet Type	TSN Type	Tagged	Destination Rx Queue Determined By
Broadcast/Multicast	Express	Yes	if MCBCQEN = 1, MCBCQ; else PSRQ
		No	if MCBCQEN = 1, MCBCQ; else UPQ
	Preemptable	Yes	if MCBCQEN = 1, PMCBCQ; else PSRQ
		No	if MCBCQEN = 1, PMCBCQ; else RQ
Unicast AVTP Control	Express	No	AVCPQ
		Yes	if TACPQE = 1, AVCPQ; else PSRQ
Unicast PTP over Ethernet	Express	No	PTPQ
		Yes	PTPQ or PSRQ based on TPQC
Remaining Unicast	Express	No	UPQ
		Yes	PSRQ
	Preemptable	No	RQ
		Yes	PSRQ

- Table 728 shows decreasing order of priority when OMCBCQ field of the MAC_RxQ_Ctrl1 register is 1

Table 728 Priority Routing of Filter Pass Packets When OMCBCQ = 1

Packet Type	TSN Type	Tagged	Destination Rx Queue Determined By
All AVTP Control	Express	No	AVCPQ
		Yes	if TACPQE = 1, AVCPQ; else PSRQ
All PTP over Ethernet	Express	No	PTPQ
		Yes	PTPQ or PSRQ based on TPQC
Remaining Broadcast/Multicast	Express	Yes	if MCBCQEN = 1, MCBCQ; else PSRQ
		No	if MCBCQEN = 1, MCBCQ; else UPQ
	Preemptable	No	if MCBCQEN = 1, PMCBCQ; else PSRQ
		Yes	if MCBCQEN = 1, PMCBCQ; else RQ
Remaining Unicast	Express	No	UPQ
		Yes	PSRQ

(table continues...)

Table 728 (continued) Priority Routing of Filter Pass Packets When OMCBCQ = 1

Packet Type	TSN Type	Tagged	Destination Rx Queue Determined By
	Preemptable	No	RQ
		Yes	PSRQ

***Note**:*

- *If the RxQ pointed by the various fields in Table 725 through Table 728 is not enabled in the RxQ_Ctrl0 register as per the packet type, the packet is written to*
  - *RxQ0 for Express packets*
  - *Port j MAC RxQ Ctrl1.RQ for Preemptable packets*

  If RxQ0/RQ is not enabled, the packet is dropped
- *If the AV8021ASMEN field of the MAC_Timestamp_Control register is 1, MAC receiver checks only for untagged PTP packet. Tagged packet with PTP payload are considered as Remaining (Unicast/Broadcast/Multicast) tagged packets in Table 725 through Table 728*

14.4.3.20.10 Rx side routing from MAC to queues

The MAC routes the Rx packets to the Rx Queues based on the following packet types specified in that order:

1. Unicast/Multicast destination address packets that fail the destination address filter
2. Multicast/Broadcast destination address packets that pass the destination address filter
3. VLAN Tag Priority field in VLAN Tagged AV data packets
4. AV Control packets
5. VLAN Tagged IEEE 1588 PTP over Ethernet packets
6. Untagged IEEE1588 PTP over Ethernet packets
7. VLAN Tag Priority field in VLAN Tagged packets
8. Untagged packets

The outer C-VLAN Tagged AV data Rx packets can be routed based on the priorities assigned to Rx Queues through PSRQ field in corresponding `Port j MAC RxQ Ctrl2` and `Port j MAC RxQ Ctrl3` registers and the corresponding Rx Queue is enabled for AV through RXQ#EN field in `Port j MAC RxQ Ctrl0` register. These packets may be single VLAN Tagged with C-VLAN type or Double VLAN Tagged with outer VLAN Tag of C-VLAN type when Double VLAN feature is enabled (EDVLP bit in `Port j MAC VLAN Tag Ctrl` register is set) with inner C-VLAN Tagged or inner S-VLAN Tagged when SVLAN processing is enabled (ESVL bit in `Port j MAC VLAN Tag Ctrl` register is set). This type of Rx packet routing is available when AV feature and Multiple Rx Queues are selected in the configuration.

The AV Control Rx packets can be routed based on the Rx Queue number specified in the AVCPQ field in `Port j MAC RxQ Ctrl1` register and corresponding Rx Queue is enabled for AV through RXQ#EN field in `Port j MAC RxQ Ctrl0` register. These packets may be single VLAN Tagged with C-VLAN type or Double VLAN Tagged with outer VLAN Tag of C-VLAN type when Double VLAN feature is enabled (EDVLP bit in `Port j MAC VLAN Tag Ctrl` register is set) with inner C-VLAN Tagged or inner S-VLAN Tagged when SVLAN processing is enabled (ESVL bit in `Port j MAC VLAN Tag Ctrl` register is set). This type of Rx packet routing is available when AV feature and Multiple Rx Queues are selected in the configuration.

The VLAN Tagged Rx packets can be routed based on the priorities assigned to Rx Queues through PSRQ field in corresponding `Port j MAC RxQ Ctrl2` and `Port j MAC RxQ Ctrl3` registers and the corresponding Rx Queue is enabled through RXQ#EN field in `Port j MAC RxQ Ctrl0` register. This type of Rx packet routing is available when Multiple Rx Queues are selected in the configuration.

The untagged IEEE 1588 PTP over Ethernet Rx packets can be routed based on the Rx Queue number specified in the PTPQ field in `Port j MAC RxQ Ctrl1` register and the corresponding Rx Queue is enabled through RXQ#EN field in `Port j MAC RxQ Ctrl0` register.

The VLAN tagged IEEE 1588 PTP over Ethernet Rx packets can be routed based on the priorities assigned to Rx Queues through PSRQ field in corresponding Port j MAC RxQ Ctrl2 and Port j MAC RxQ Ctrl3 registers or the Rx Queue number specified in the PTPQ field in Port j MAC RxQ Ctrl1 register and the corresponding Rx Queue is enabled through RXQ#EN field in Port j MAC RxQ Ctrl0 register. This is determined by programming in TPQC field of Port j MAC RxQ Ctrl1 register. This type of Rx packet routing is available when IEEE 1588 Timestamp feature support and Multiple Rx Queues are selected in the configuration. The VLAN Tagged IEEE 1588 PTP over Ethernet Rx packets are detected only when 802.1AS mode is disabled (AV8021ASMEN bit in Port j MAC Timestamp Control register is set to 0), otherwise this type of Rx packets are routed as generic VLAN Tagged Rx packets.

The multicast/broadcast destination address Rx packets that pass the destination address filter can be routed based on the Rx Queue number specified in the MCBCQ field of Port j MAC RxQ Ctrl1 register when enabled through MCBCQEN bit of Port j MAC RxQ Ctrl1 register and the corresponding Rx Queue is enabled through RXQ#EN field in Port j MAC RxQ Ctrl0 register. This type of Rx packet routing is available when Multiple Rx Queues are selected in the configuration. XGMAC supports a programmable option to de-prioritize the MCBC queue priority over the packet type based queue routing (PTP/AV) for Multicast and Broadcast packets. MCBC queue priority can be changed by setting OMCBCQ field [bit 20] of the Port j MAC RxQ Ctrl1 register.

The untagged Rx packets can be routed based on the Rx Queue number specified in the UPQ field of Port j MAC RxQ Ctrl1 register and the corresponding Rx Queue is enabled through RXQ#EN field in Port j MAC RxQ Ctrl0 register. This type of Rx packet routing is available when Multiple Rx Queues are selected in the configuration.

The unicast destination address Rx packets that fail the destination address filter can be routed based on the Rx Queue number specified in the UFFQ field of MAC_RxQ_Routing_Ctrl register when enabled through UFFQE bit of MAC_RxQ_Routing_Ctrl register, RA bit of Port j MAC Packet Filter register is set to 1 and the corresponding Rx Queue is enabled through RXQ#EN field in Port j MAC RxQ Ctrl0 register. This type of Rx packet routing is available when Multiple Rx Queues are selected in the configuration.

The multicast destination address Rx packets that fail the destination address filter can be routed based on the Rx Queue number specified in the MFFQ field of MAC_RxQ_Routing_Ctrl register when enabled through MFFQE bit of MAC_RxQ_Routing_Ctrl register, RA bit of Port j MAC Packet Filter register is set to 1 and the corresponding Rx Queue is enabled through RXQ#EN field in Port j MAC RxQ Ctrl0 register. This type of Rx packet routing is available when multiple Rx Queues are selected in the configuration.

***Note**: Queue routing behavior for the received runt packet cannot be predicted*

If the Rx packet cannot be classified in any of the defined packet type for routing, it is routed through Rx Queue 0.

For more details, see the section "Registers".

Related information

Registers on page 802

14.4.3.20.11 Receive Queue Selection Based on Filter Fail

This feature enables the selection of RxQ when the unicast/multicast filter fails and VLAN filter also fails.

The multicast destination address Rx packets that fail the destination or source address filter are routed based on the Rx queue number specified in the MFFQ field of Port j MAC RxQ Ctrl4 register. When enabled through MFFQE bit of Port j MAC RxQ Ctrl4 register, RA bit of Port j MAC Packet Filter register is set to 1 and the corresponding Rx queue is enabled through RXQ#EN field in the Port j MAC RxQ Ctrl0 register. This type of Rx packet routing is available when multiple Rx queues are selected in the configuration.

The unicast destination address Rx packets that fail the destination or source address filter can be routed based on the Rx queue number specified in the UFFQ field of Port j MAC RxQ Ctrl4 register. When enabled through

UFFQE bit of Port j MAC RxQ Ctrl4 register, RA bit of Port j MAC Packet Filter register is set to 1 and the corresponding Rx queue is enabled through RXQ#EN field in Port j MAC RxQ Ctrl0 register. This type of Rx packet routing is available when Multiple Rx queues are selected in the configuration.

14.4.3.20.12 VLAN based DMA Mapping

This section discusses the Extended RX VLAN Filtering function with regards to the DMA Channel mapping. Routing is applicable only if the entire VLAN Filter has passed. Routing is only based on Perfect Filter Result. Each perfect filter has a DMA Channel Enable and a DMA Channel Number field which can be programmed. Only if the DMA Channel Enable bit is set, routing is applicable for that filter.

The frame is routed to the smallest Matching Filter's DMA Channel provided the DMA Channel is enabled. If that filter's DMA Channel number is not enabled, the frame gets routed to Channel 0. For example, if a frame's VLAN tag matches filters 7, 3, and 1. Then the MAC checks if Filter 1's DMA Channel Number is enabled through programming. If yes, the frame gets routed to the programmed value; otherwise it gets routed to 0. When the inverse filter is enabled; is routed to the least mismatched filter's DMA channel number provided the DMA Channel is enabled.

If Hash Filter is also enabled, it is used to determine the Filter result only. Routing still depends on the Perfect Filters enabled. If none of the perfect filters are enabled or if all of them are bypassed, then the VLAN Filter based routing does not take place. If all the perfect filters give a fail result and the Hash Filter has passed, then the VLAN Filter result is a pass but routing is based on DA Based Addressing or to Channel 0. Similar behavior is seen when inverse Filtering is enabled as well. This behavior is for non-tunnel frames.

The overall VLAN Filter is produced and it decides if the frame can be dropped or forwarded. If the frame gets a fail result from the VLAN Filter, then it is dropped if VTFE = 1 & RA = 0.

If stripping is enabled, the matching tag value is lost as it cannot be passed through the status. The application needs to be cautious before enabling stripping.

14.4.3.21 Time-Sensitive Networking

Time sensitive network (TSN) is a set of standards developed by the IEEE 802.1 work group. The standards define mechanisms for the time-sensitive transmission of data over Ethernet.

Related information

Policing Counters on page 1517
Stream-Gate Filtering on page 1521
Enhancements to Scheduled Traffic (EST) on page 1523
Frame Preemption (FPE) on page 1534
Time-Based Scheduling on page 1546

14.4.3.21.1 Policing Counters

Overview of Policing Counters

Policing counters feature counts the filter events and gathers the diagnostic information of the received packets. This feature can monitor events associated with filters such as DA/SA, VLAN, L3_L4, flexible receive parser, and stream-gate filtering. This feature maintains a set of registers to gather the filter events information that include, register for controlling the behavior of the 32-bit policing counters and policing counter status registers for providing status of the filter events. When a counter value crosses the programmed threshold, all the subsequent packets which asserts the filter events associated to the counter are dropped as a corrective measure. Interrupts are generated based on the programmed threshold and the selected time window.

Description of policing counters

This feature supports sixteen 32-bit policing counters to track the filter event. The policing counters can track the following filter events, only for perfect filtering:

- DA/SA filter pass
- DA/SA filter fail
- VLAN filter pass
- VLAN filter fail
- L3_L4 filter pass
- L3_L4 filter fail
- Flexible receive parser, accept frame
- Flexible receive parser, reject frame
- stream-gate filter pass

***Note**: Do not associate policing counter with inverse filtering of DA, VLAN, and L3_L4, as this can cause unpredictable behavior.*

You can program the filter events to be associated with any of these policing counters. A policing counter can be associated with more than one filter event. Each policing counter has a programmable time window and threshold.

A time window is provided as a sideband input, using the `sbd_pc_tim_wdw_i` input signal. The connections (with PPS outputs) on system level is described in chapter "Connectivity".

The rising edge of the signal represents the start of the time window, and the falling edge represents the end of the time window. You can program the counters to select any one of the four sideband time windows. You can program the policing counter to increment

- Irrespective of the selected time window (free running mode)
- Within the selected time window (time-window mode)

The programmable threshold is controlled by the `MAC_PCCtrl_IndReg` register. When the count reaches the programmed threshold, an interrupt is generated. You can program the register to restart the counters at the end of each time window.

Event Generators

The policing counter captures the filter events in the status registers, to indicate that an event is generated. Each filter event has an enable and a counter selection programming as described in the "Policing counter selection and filter status" section. When multiple filter events are associated with the same counter, the events are merged and updated in the corresponding counter.

Setting/resetting of the IPFE, VTFE, and SAF fields of the `MAC_Packet_Filter` register has no impact on the policing counter.

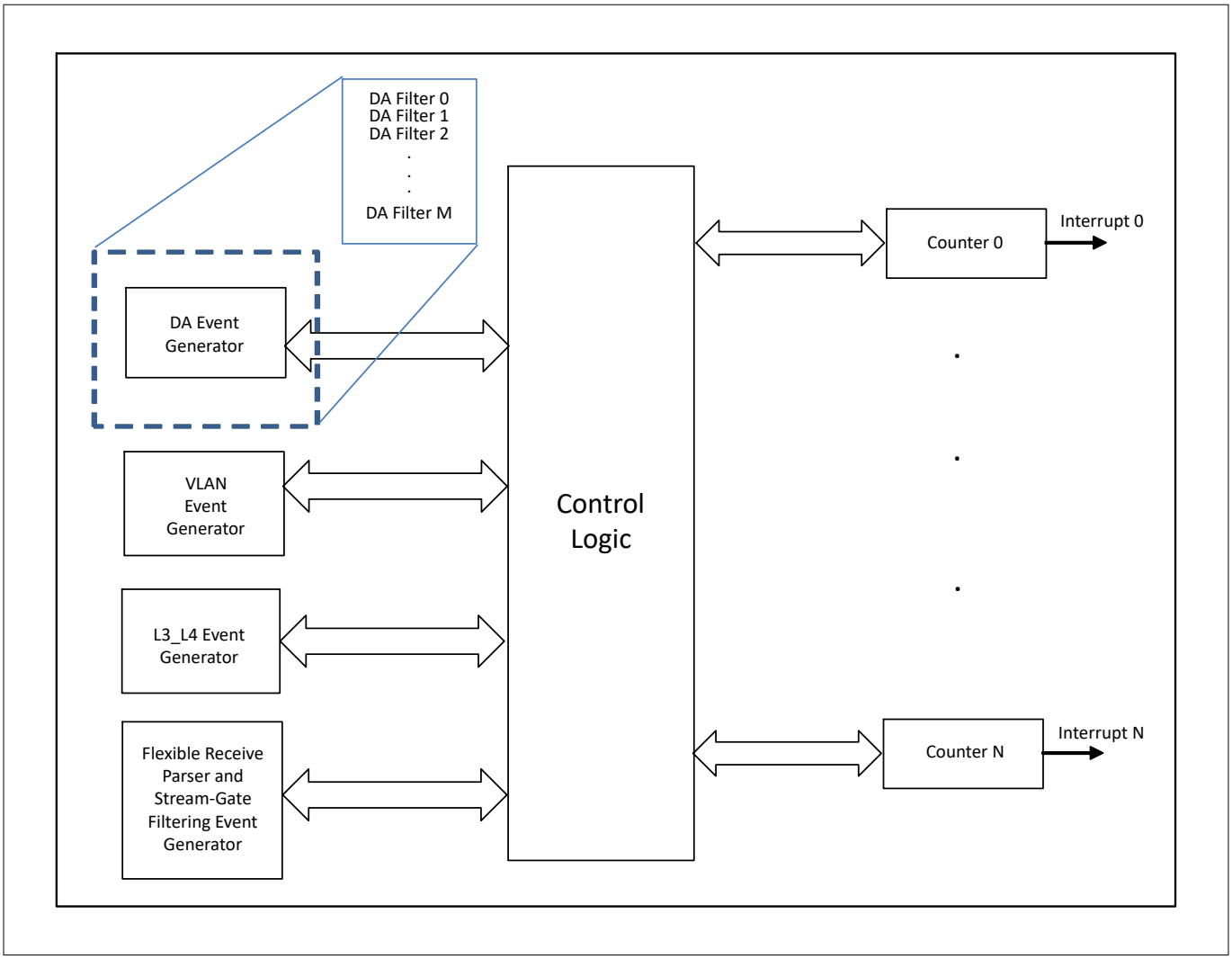


Figure 155 Policing Counter Flow Diagram

Related information

Policing Counter Selection and Filter Status on page 1520

Policing Counter Interrupts

Policing counters support 2 types of interrupts:

- Time window based interrupt

  When you enable time window based interrupt, the interrupt is triggered at the end of a time window. Each policing counter has a time window based interrupt status bit, which is either write-1 to clear or clear-on-read based on the value programmed in the RCWE field of the `Port j MAC CSR SW Ctrl` register
- Threshold based interrupt

  When the counter reaches the threshold that you programmed, an interrupt is asserted. Each policing counter has a threshold-based interrupt status bit, which is either write-1 to clear or clear-on-read based on the value programmed in the RCWE field of the `Port j MAC CSR SW Ctrl` register

The combined status of the policing counter interrupts is indicated in the PCIS field of the `Port j MAC Interrupt Status` register.

Policing Counter Selection and Filter Status

Table 729 shows the registers to be programmed to enable policing counters for various filters and where the filter event status is indicated. The filter event status is set when at the least one corresponding filter event is triggered. Based on the value of the RCWE field of the MAC_CSR_SW_Ctrl register, the status bits of the various policing counter status registers can be cleared either by writing 1 or by clear-on-read.

Table 729 Policing Counter Selection and Filter Status Indication

Filter Type	Register for Enabling/Selection of Policing Counter	Indicating the Filter Event Status in the Policing Counter Status Register
SA/DA pass	MAC_DPCSel_IndReg	MAC_PCStatus_DA_IndReg
VLAN pass	MAC_VPCSel_IndReg	MAC_PCStatus_VLAN_IndReg
L3_L4 pass	MAC_LPCSel_IndReg	MAC_PCStatus_L3L4_IndReg
Flexible Receive Parser and Stream-gate filtering	See the description of the bit position 91:80 in the "Flexible receive parser instruction table (IT) format" section.	Receive Parser, Receive Status
Filter fail	MAC_FPCSel_IndReg0	FFAIL field of the MAC_PCStatus_Filter_Event_IndReg

Related information

Flexible Receive Parser Instruction Table (IT) Format on page 1441

Policing the Bandwidth

The policing counters mapped to various filter pass events also provide bandwidth estimates of number of packets and the associated traffic type that are forwarded to the application. When a filter pass event's policing counter value exceeds the programmed threshold value, and the FFAT field of the MAC_PCCtrl_IndReg is set, the packets are treated as filter fail, and processed accordingly, until the associated counter is cleared. Therefore, the bandwidth policing is in terms of number of packets and not in terms of packet sizes.

When a packet is considered filter fail, based on the FFAT field of the MAC_PCCtrl_IndReg,

- A filter fail event is not asserted. This is because, there is no actual filter failure, but only a threshold exceed.
- Policing counters associated with the filter pass events are incremented and the status for the corresponding filter is also updated.
- If the filter corresponding to a filter pass event is not enabled to drop the packets using IPFE/VTFE and SAF fields of the MAC_Packet_Filter register, the packets are not dropped. Instead, the packets are forwarded to the application with a fail status.
- Multicast pass-all packets are forwarded to the application.

Registers Related to Policing Counters

- MAC_PCntr_IndReg
- MAC_PCStatus_Filter_Event_IndReg
- MAC_PCStatus_L3L4_IndReg
- MAC_VPCSel_IndReg
- MAC_DPCSel_IndReg

- MAC_LPCSel_IndReg
- MAC_PCCtrl_IndReg
- MAC_FPCSel_IndReg0
- MAC_PCTH_Intr_Enable
- MAC_PCTH_Intr_Status

14.4.3.21.2 Stream-Gate Filtering

The flexible receive parser supports a subset of IEEE 802.1 Qci specification, stream-gate filtering. The stream-gate filtering support is based on stream-gate instance and stream filter instance. Policing or metering functions of the flow meter instance should be implemented by the application, as shown in the figure below.

- **Stream Filter**

  Identifies the Stream ID and maps to one of the eight Gate IDs; Only eight Gate IDs are supported
- **Stream Gate Instance**

  All Gate IDs share the same GCL

Based on the configuration, the flexible receive parser can parse the first 256 bytes of a frame. When you enable stream-gate filtering in the configuration, in addition to filtering and DMA channel routing, the parser performs stream-gate filtering. The result of the stream-gate filtering is based on the Gate ID of the received frames and the GCL programming. For details of the Gate ID and Gate ID valid fields, see the section "Flexible Receive Parser Instruction Table (IT) Format".

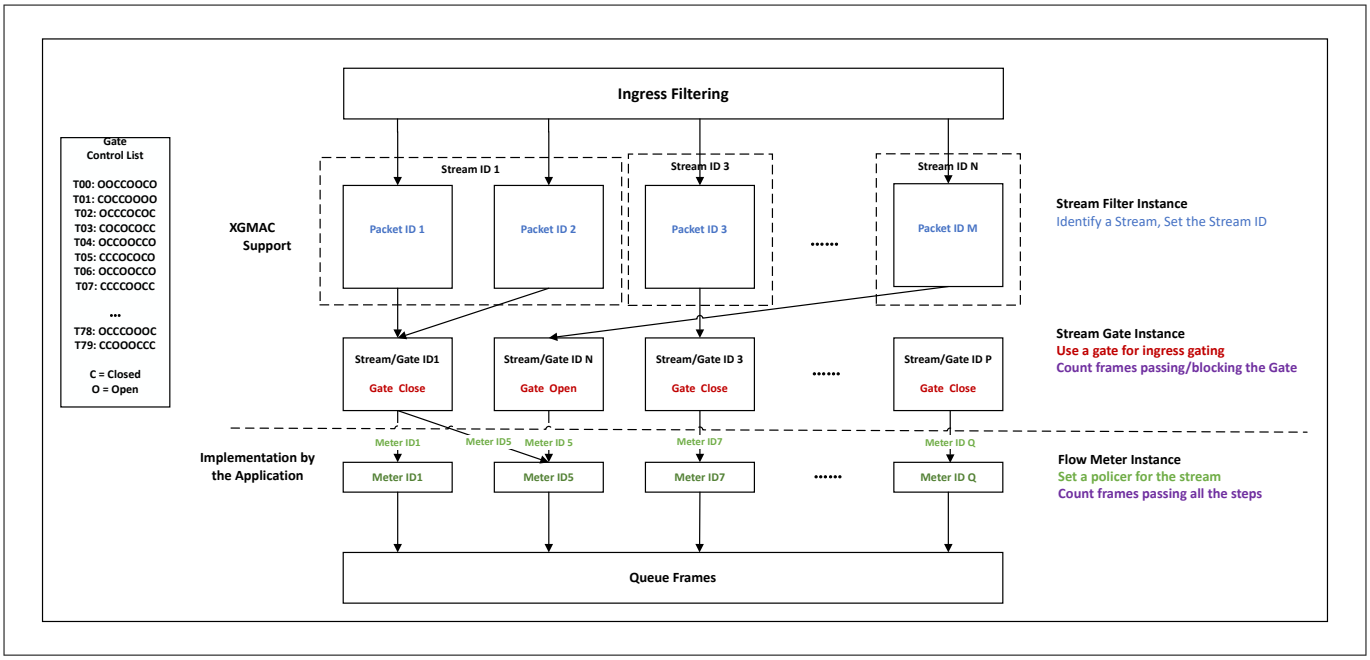


Figure 156 Block Diagram of Stream-Gate Filtering

When the Flexible receive parser result is available, the parser uses the Gate ID (GID field) of the Instruction Table and the Gate Open/Close status of the stream-gate filtering, to determine if the result is pass or fail. When the stream-gate filtering fails, the failed traffic is routed in one of the following ways:

- DMA channel, channel 0 for express traffic
- Channel number configured in the FPE_RCH field of Port j MTL RXP Control Status register, for preemption traffic

The details of the fail/drop are indicated by FPS[2:0] bits of RDES2 and RDES3 as shown in the table "Flexible Receive Parser Receive Status" in the section "Receive Normal Descriptor (Writ-Back Format).

When the stream-gate filtering passes, the frames are processed by the flexible receive parser.

If the receive management counter is enabled in the configuration, the stream-gate filtering tracks the pass/fail packets in the MMC_Stream_Gate_Passed_Packets and MMC_Stream_Gate_Failed packets counters respectively.

Related information

Flexible Receive Parser Instruction Table (IT) Format on page 1441

Receive Normal Descriptor (Write-Back Format) on page 1572

Stream-Gate Filtering with EST

To support stream-gate filtering with EST, flexible receive parser implements one external GCL memory that can be used by both EST and stream-gate filtering. A memory arbiter is used to avoid EST and stream-gate filtering accessing the same GCL memory simultaneously, as shown in Figure 157.

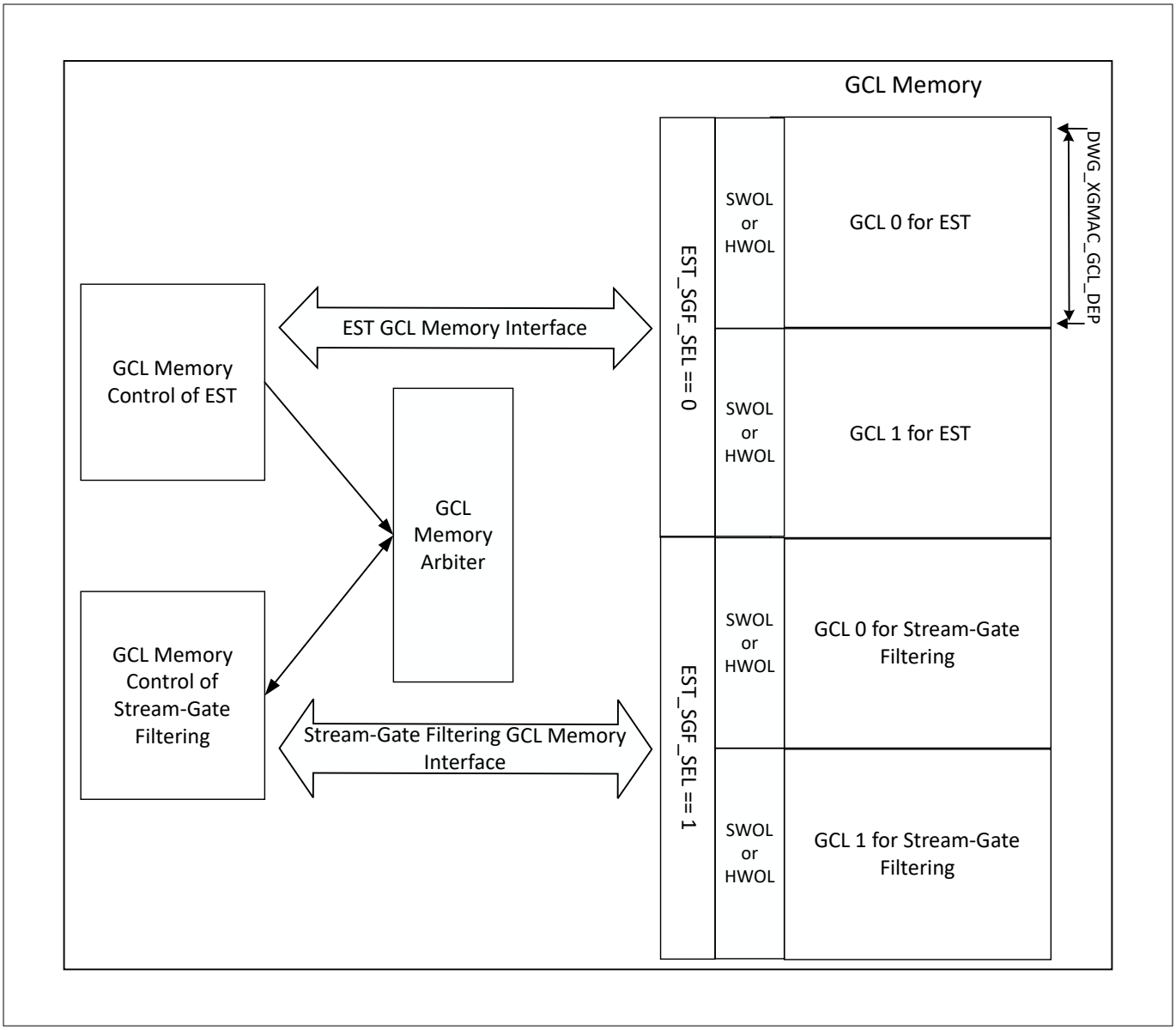


Figure 157 Memory Implementation for Stream-Gate Filtering with EST

Stream-Gate Filtering with Frame Preemption

When you enable stream-gate filtering for preemption frames, the filtering is performed on the preemption frames. For fragmented frames, stream-gate filtering is performed only on the first fragment. Continuation fragments are not considered for stream-gate filtering.

14.4.3.21.3 Enhancements to Scheduled Traffic (EST)

The IEEE 802.1Qbv-2015 defines the schedule for each of the queues on every egress port which makes the implementation aware of traffic arrival schedule. This information can be used to block the lower priority traffic from transmission in this time window/slot. This ensures that scheduled traffic is forwarded from sender to receiver through all the network nodes with a deterministic delay.

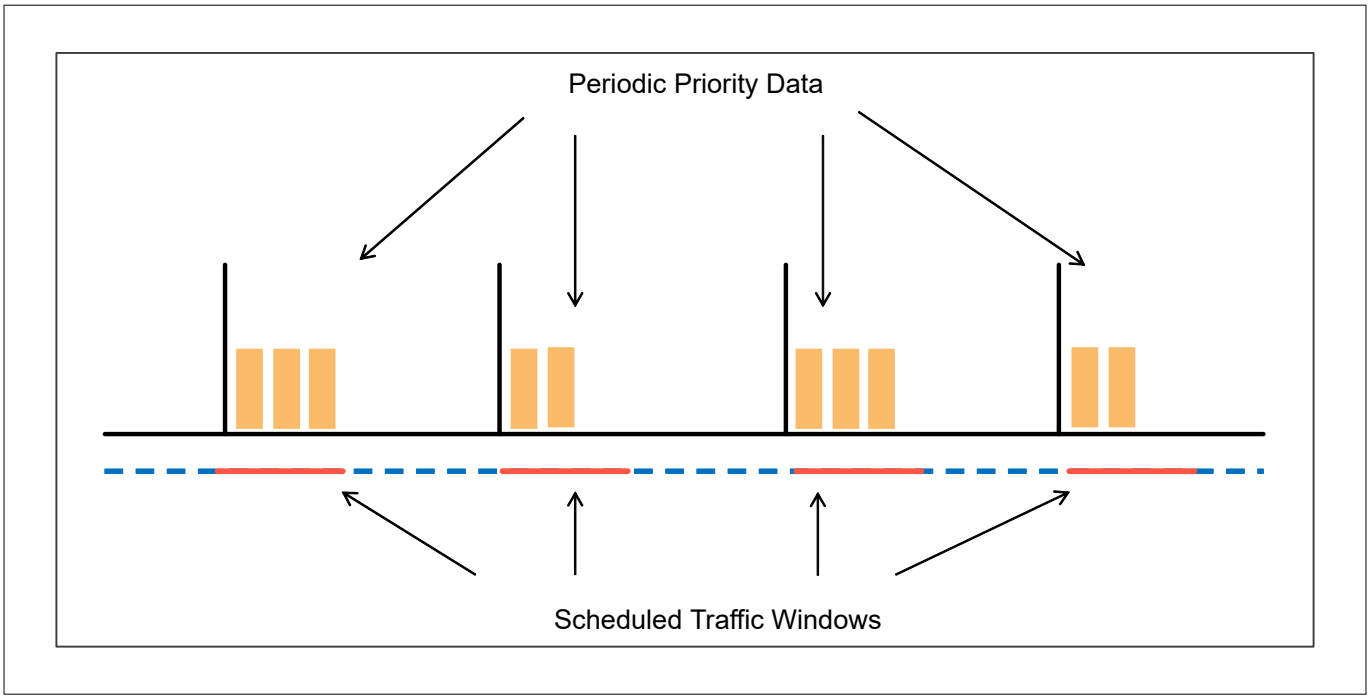


Figure 158 Time Aware Shaper Implementation for Traffic Scheduling

One of the important requirements to achieve a low latency is to ensure there are no interfering frames during the scheduled windows that are reserved for high priority traffic. The use of scheduled traffic imposes limitations while starting a transmission.

As shown in Figure 159, if an interfering frame begins transmission just before the start of a reserved time period, it can extend transmission into the reserved window, and potentially interfere with higher priority traffic. Therefore, a guard band whose size is equal to the largest possible interfering frame is required before the window starts.

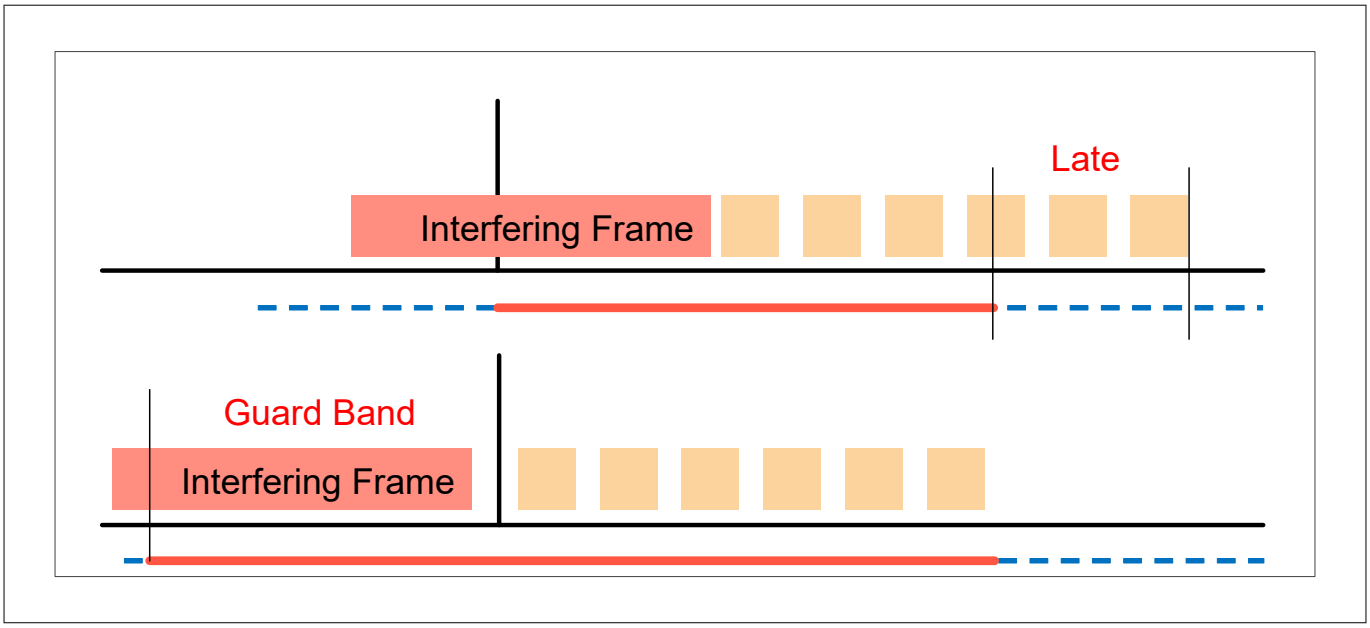


Figure 159 Implementing a Guard Band to Avoid Delays Due to Interfering Frames

A larger guard band equates to a less efficient use of network bandwidth. However, with the implementation of IEEE802.1Qbu (frame preemption), this issue is addressed. Frame preemption breaks the interfering frame into smaller fragments. Therefore, the guard band needs to be only as large as the largest possible interfering fragment instead of the largest possible interfering frame.

During the guard band, only the frames that can complete the transmission of the entire frame before the next gate close event are permitted. This ensures that the high priority traffic can always start at the beginning of the window reserved for it.

Frequently Used Terms in EST

The following are some of the frequently used terms in the EST support and their definitions.

- Gate Control List

  The list in the hardware memory that holds the gate controls and the associated time intervals
- Gate Controls

  For a given schedule (row in gate control list) there is a Gate Open (O) and Gate Close (C) state associated with each Transmit Channel (TC). The set of O or C values, whose width is same as configured TCs is called the Gate Controls. For example, CCOCOOCO means TC7=C, TC6=C, TC5=O and so on
- Time Interval

  Time interval (in nanoseconds) is a 24-bit configurable field in the Gate Control list that indicates the time for which the associated gate controls are valid
- Base Time Register

  Each Gate Control List is associated with a 64-bit Base Time Register that holds the start time (in PTP format) for the list

Updates to the Transmit Scheduling to Support EST

To support EST, following updates are required to the transmit scheduling:

- Implementation of Gate Control List (GCL)
- Enforcing gate controls in the scheduler
- Accounting for Gate Open (O) duty cycle in the computation of idleSlope (CBS)

Implementation of Gate Control List (GCL)

Figure 160 shows block diagram from the IEEE 802.1Qbv spec that illustrates how the gate control list governs the gate close (C) and open (O) events based on the schedule provided for each event.

GCL has the following two parts:

- Time Interval

  Defines the time in nanoseconds for which the gate controls are valid and should be applied before reading the next gate controls from the list. Configurable width of 24-bits to represent a maximum of 16 ms schedule interval. Supports left shift of the time interval up to 7 bits (Portj_MTL_EST_Control.TILS) to be able to apply a multiplication factor from 1 to 128 ns (in steps of 2^n). The maximum value (post shifting) of this field is 999,999,999 ns

  ***Note**: The minimum time interval for CGL entry should be equal to the time required to transmit fragment_size + IPG/EIPG + preamble overheads*

- Gate Control

  Defines the Open (O) represented by logic 1 or Close (C) represented by logic 0 state for the gate of each TC

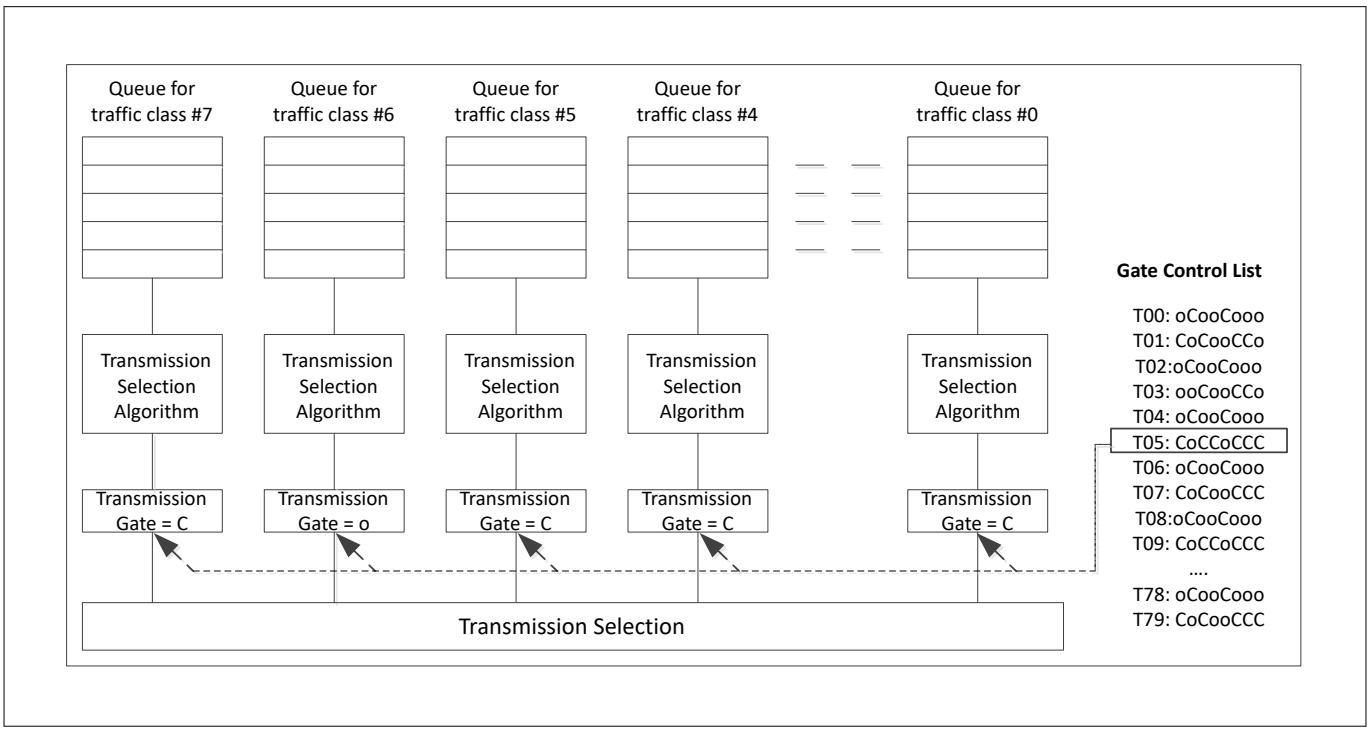


Figure 160 Block Diagram from IEEE 802.1 Qbv Specification - GCL Governing Gate Close and Open Events

The implementation of GCL consists of the following two Gate Control Lists:

- HWOL - Hardware Owned List which is a list for hardware access
- SWOL - Software Owned List which is a list for software access

The access to these lists is mutually exclusive. The ownership to the list is set by hardware in the SWOL field of Port j MTL EST Status register. The following table provides the implementation details of GCL.

Table 730 External Memory Used for Holding the Two Gate-Control Lists

Gate Control (up to 8 bits)	Time Interval (ns) 24 bits	
OOCCCCCC	T0	HWOL or SWOL
OOOOCCCC	T1	
CCCCOOCC	T2	
CCCCCCOO	T3	
…	…	
…	…	
OCOCOCOO		
OOOOCCCC	T last	
OOOOCCCC	T0	SWOL or HWOL
OOOOCCCC	T1	
OOCCCCCC	T2	
OOOOCCCC	T3	
…	…	
…	…	
OCOCCCCC		
OOCCCCCC	T last	

Registers Related to Gate Control List

Following are the set of 4 registers (one for each GCL) related to GCL. These registers are implemented through Indirect Addressing using the GCL_Control and GCL_Data registers.

1. 64-bit Base Time Register (BTR)
2. 40-bit Cycle Time (CTR)
3. m-bit Time Extension (TER)
4. n-bit [Gate Control] List Length (LLR) (n depends of the configured GCL depth)

Base Time Register (BTR)

Base Time Registers is a 64-bit register that holds the start time to execute the GCL. The format of the BTR is same as the PTP format (upper 32-bits holds time in seconds and lower 32 bits hold time in nano seconds). Once the execution of a given list begins, the implementation can update the value in BTR to indicate the next list execution begin time.

Cycle Time Register (CTR)

Cycle Time Register is a 40-bit register that holds the time at which the execution of the GCL should be repeated. The Cycle Time Register consists of an 8-bit value in seconds, and a 32-bit value in nano seconds (similar to the PTP time format with truncated seconds register).

For a given Gate Control List the start time is "Base Time" + N * "Cycle Time" where N is an integer value representing the iteration number starting with 0 for first iteration. If the Gate Control List execution takes longer than the Cycle time, then the list is truncated at the Cycle Time and the subsequent loop begins at Cycle Time.

Time Extension Register (TER)

Time Extension Register (TER) is a m-bit (where m = Configured Time Interval width + 7) register that holds the amount of time (in nano seconds) the current Gate Control List can be extended before switching to the new Gate Control List. This avoids smaller fragments of the current list being executed before switching to a new list.

List Length Register (LLR)

List Length Register (LLR) is an n-bit register (when n is 7, 8, 9, 10, or 11 for a GCL configured depth of 64, 128, 256, 512, or 1024 respectively) that holds the integer value of the length of the GCL (that is the number of valid rows in each GCL). The processing of the GCL stops after the number of rows read equals to the LLR value.

Transmission Gating Implementation

A Bridge or an end station can be enhanced to allow transmission from each TC that is yet to be scheduled relative to a known timescale. To achieve this, a transmission gate is associated with each TC; the state of the transmission gate determines if the queued frames can be selected for transmission.

For a TC, the transmission gate can be in one of the following two states:

1. OPEN

Queued frames are selected for transmission, in accordance with the definition of the transmission selection algorithm associated with the TC.

2. CLOSED

Queued frames are not selected for transmission.

A frame on a traffic class queue is not available for transmission if the transmission gate is in the closed state or if there is insufficient time available to transmit the entirety of that frame before the next gate-close event associated with that queue.

The implementation has visibility to the current schedule of gate controls and the immediate next schedule of the gate controls. So the maximum Gate Open period does not exceed the sum of the two Time Intervals. This is because, a frame is selected for transmission only if the gate is currently Open and the duration of gate open interval is greater than the time taken to transfer the entire frame.

The implementation must know the frame size before the transmission, so that the transmission overruns can be avoided and only the frames that can complete are scheduled at all times.

The implementation adequately compensates for the implementation delays in the data transfer from the buffer to the line by offsetting the current time with all the relevant delays (provided by the CTOV field of MTL_EST_Control register). This ensures that the schedule provided is always accurately implemented at the line.

***Note**:*

- *A 20 Byte overhead is added to the packet size to account for the IFG and Preamble overheads on the line. Therefore for a 128 Byte frame to be transmitted, the Gate Open window should at least be able to accommodate148 Bytes.*
- *In 100 Mbps mode of operation, the rounding down error is about 0.4%. For example, the Gate open duration must be at least 1024 Bytes for transmitting 1000 Byte Frames at a speed of 100 Mbps (20 Bytes for line overheads, and 4 Bytes for rounding down error)*

Idle Slope Computation Updates

When EST is enabled, credit is accumulated only when the gate is open; therefore, the effective data rate of the idleSlope must be increased to reflect the duty cycle for the transmission gate associated with the queue.

The idleSlope is computed based on the GateOpenTime and OperCycleTime values. Program the idleSlope registers (implemented one per CBS enabled TC) based on the following equation. The existing MTL register has sufficient bit width to accommodate the new values for idleSlope.

```
idleSlope = (operIdleSlope(N) * OperCycle/GateOpenTime)
```

Operational Details of GCL

Set the switch to software List, the SSWL field of the MTL_EST_Control Register, so that hardware can access the programmed gate control list. The first set of gate controls are applied when the current time is equal to the value in the Base Time Register (BTR) and is held until the programmed "Time Interval" value.

To avoid transmission overruns, one additional gate control event is always read ahead from the list. This enables the GCL to determine the next gate close events (if any) for the TCs that are open.

The scheduling based on the Gate Open state and Time Interval of only the current and subsequent schedule. An internal accumulator is used to add the time intervals when gate controls are applied. BTR + Accumulator holds the time at which the next set of gate controls are to be applied.

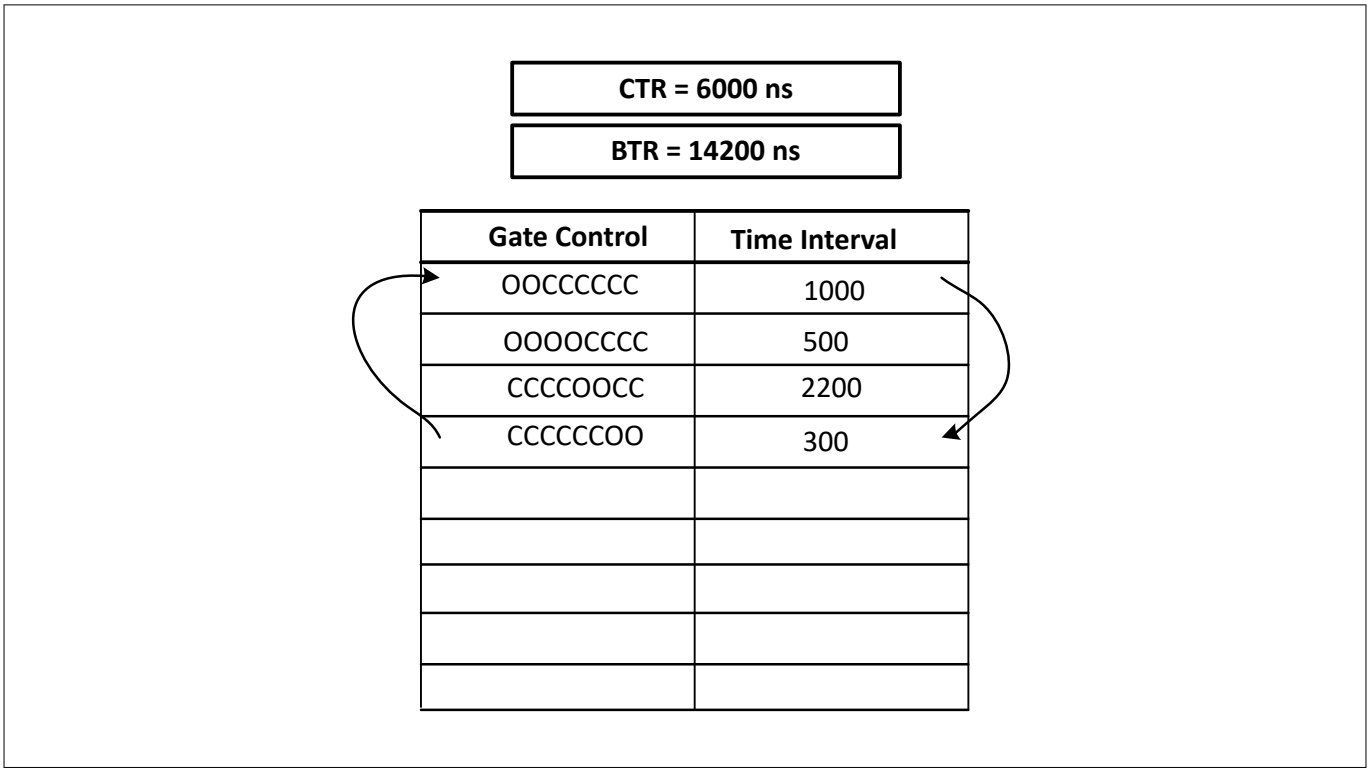


Figure 161 GCL and Associated Registers - BaseTime and CycleTime

The GCL is read progressively from the first row adhering to the schedule. The read operations continue until the List Length (from LLR Register) is reached and the execution of the list restarts at BTR + CTR time. At this point the BTR is incremented by value in CTR to mark the beginning of a new cycle. In the absence of any gate controls, all the gates are deemed to be in Open state, during the execution of the list.

In cases where the execution time of the list is greater than the CycleTime, the list is truncated and the next iteration starts when the current time equals BTR + CTR.

Table 731 GCL and Associated Registers - BTR and CTR

Current Time	Gate Control Applied	Accumulator Value	BTR (with updates)
14200	OOCCCCCC	1000	14200
15200	OOOOCCCC	1500	14200
15700	CCCCOOCC	3700	14200
17900	CCCCCCOO	4000	14200
18200	OOOOOOOO	0	20200
20200	OOCCCCCC	1000	20200
21200	OOOOCCCC	1500	20200

In the example in Table 731, the execution starts at 14200 and the first set of Gate Controls "OOCCCCCC" are applied immediately. As the time interval is 1000ns the next set of gate controls are applied at 14200 (BTR) + 1000 (Accumulator) = 15200 ns as shown in Figure 162. As there are no gate controls available after the execution of the last gate control and before the next iteration of the loop, the gates are deemed to be in open state during that time period as depicted at time 18200 in Table 731.

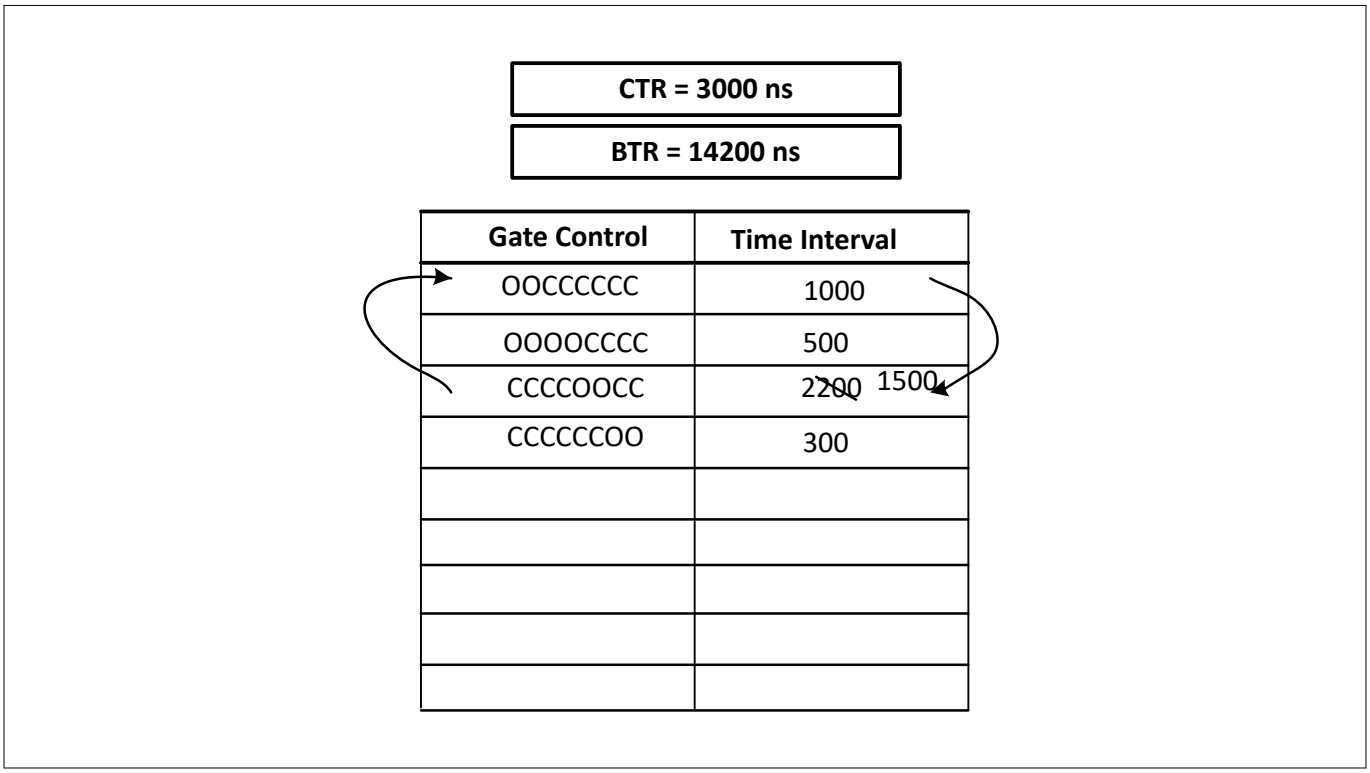


Figure 162 GCL and Associated Registers - BaseTime and CycleTime List Execution Time > CycleTime

As the list execution takes longer than the allocated CycleTime, the list is truncated and the list is started from the BTR+CTR as shown in Table 732.

Table 732 GCL and Associated Registers - BTR and CTR, Execution Time > Cycle Time

Current Time	Gate Control Applied	Accumulator Value	BTR (with updates)
14200	OOCCCCCC	1000	14200

(table continues...)

Table 732 (continued) GCL and Associated Registers - BTR and CTR, Execution Time > Cycle Time

Current Time	Gate Control Applied	Accumulator Value	BTR (with updates)
15200	OOOOCCCC	1500	14200
15700	CCCCOOCC	3700	14200
17200	OOCCCCCC	1000	17200
18200	OOOOCCCC	1500	17200
18700	CCCCOOCC	3700	17200

While applying the third set of gate controls, the BTR + CycleTime (17200) < BTR + Accumulator (17900), so the list is truncated and execution switches to a new iteration at 17200.

***Note**: When the CTR value is less than the GCL loop execution time, and the difference between the CTR and the sum of the BTR and Time Intervals of completely executed GCL rows is less than 8 PTP clock periods, then the first row of the first iteration of the next GCL is skipped. In order to avoid that, software must program the CTR, BTR, and Time Intervals of the GCL rows such that the difference between the CTR and the sum of the BTR and time intervals of fully executed GCL rows must be greater than 8 PTP clock period. If the above cannot be guaranteed by software, then the CTR must be equal to the sum of the BTR and Time Intervals of the fully executed GCL rows.*

Installing a New GCL

When a new software programmed GCL is available and must be executed at the new BTR value, the switch to the new GCL can happen in one of the following ways.

- New Base Time aligned with current schedule
- New Base Time unaligned with current schedule

New Base Time Aligned with Current Schedule

If the choice of cycle time for the new gating cycle is unchanged from the cycle time for the current gating cycle, and if the BTR chosen for the new gating cycle (new BTR) is an integer multiple of the current cycle time (+ current BTR), then the new gating cycle exactly lines up with the old gating cycle, that is, the cycle start times for the new gating cycle is same as they would have been for the old configuration. This could be considered to be the ideal case and allows the new gating cycle to be installed and executed with no timing issues. The implementation completes the execution of an iteration of the current list and switches to the new list at the beginning of the BaseTime listed in the new list.

If (New BaseTime > = Current Time) then,

1. ConfigChangeTime = New BaseTime

Else If (New BaseTime < Current Time)

1. Set the BTRError

2. ConfigChangeTime = (New BaseTime + N* New CycleTime)

where N is the smallest integer for which the relation ConfigChangeTime >= CurrentTime and (N =< 512) is TRUE

When N > 512, the hardware cannot auto recover and the loop count value in BTRL bit-field is set to 11111111 requiring the software to reprogram the New Base Time

Figure 163 illustrates the installation of the new GCL along with the timelines

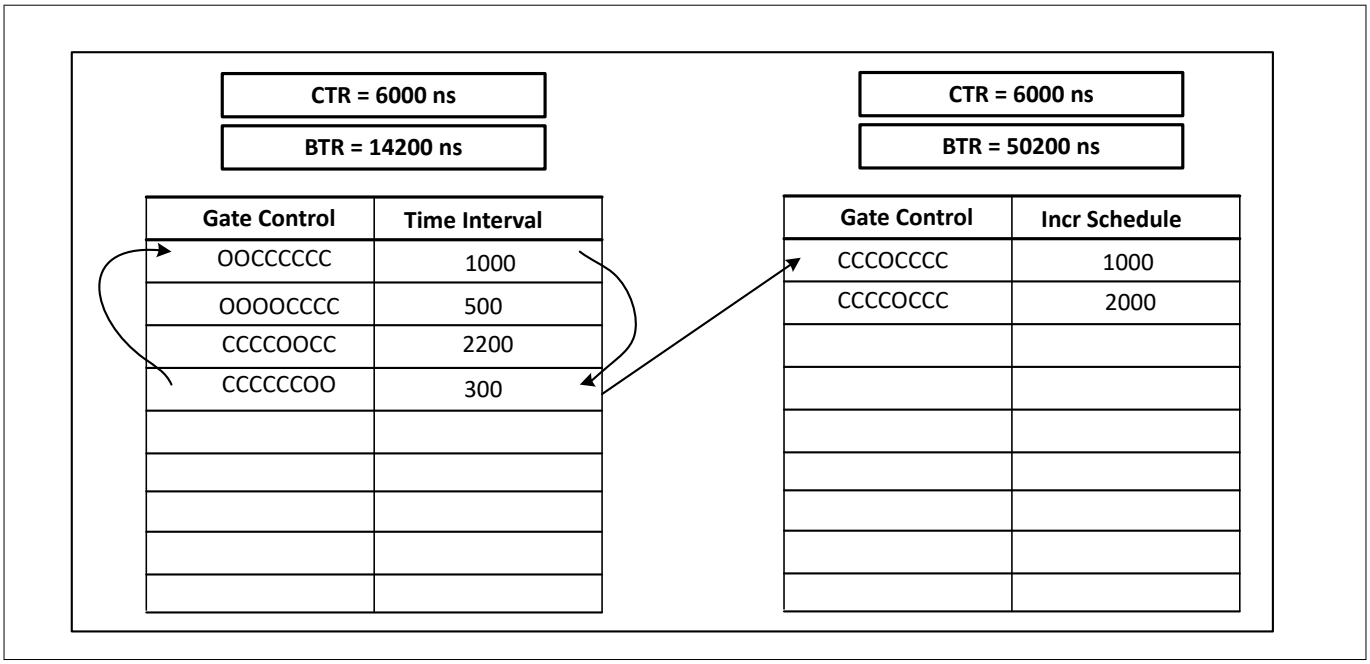


Figure 163 Switching to a New Configuration that is Aligned with the Existing Configuration

From the example in Figure 163, after the sixth iteration of the first GCL, the BTR values of the old and new GCL are equal. At that point the new GCL is processed as a natural extension to the existing GCL.

Table 733 GCL and Associated Registers - BTR and CTR

Current Time	Gate Control Applied	Accumulator Value	BTR (with updates)
44200	OOCCCCCC	1000	44200
45200	OOOOCCCC	1500	44200
45700	CCCCOOCC	3700	44200
47900	CCCCCCOO	4000	44200
48200	OOOOOOOO	0	50200
50200	CCCOCCCC	1000	50200
51200	CCCCOCCC	3000	50200
53200	OOOOOOOO	0	56200

New Base Time Unaligned with Current Schedule

If new CycleTime differs from current CycleTime or New BaseTime is in the future and is not an integer multiple of current CycleTime, then the old and new cycles do not line up; when new BaseTime is reached (when the new configuration is installed and starts to execute), the last old cycle is normally truncated to start the first new cycle. This could be undesirable if it results in a very short last old cycle; arguably it would be better to simply extend the penultimate old cycle by that small amount, rather than starting a very short cycle. The Cycle Time Extension Register (related to the current config list) allows this extension of the last old cycle to be done in a defined way; if the last complete old cycle ends normally in less than current Cycle Time Extension (TER) ns before the new base time, then the last complete cycle before new BaseTime is reached is extended so that it ends at new BaseTime.

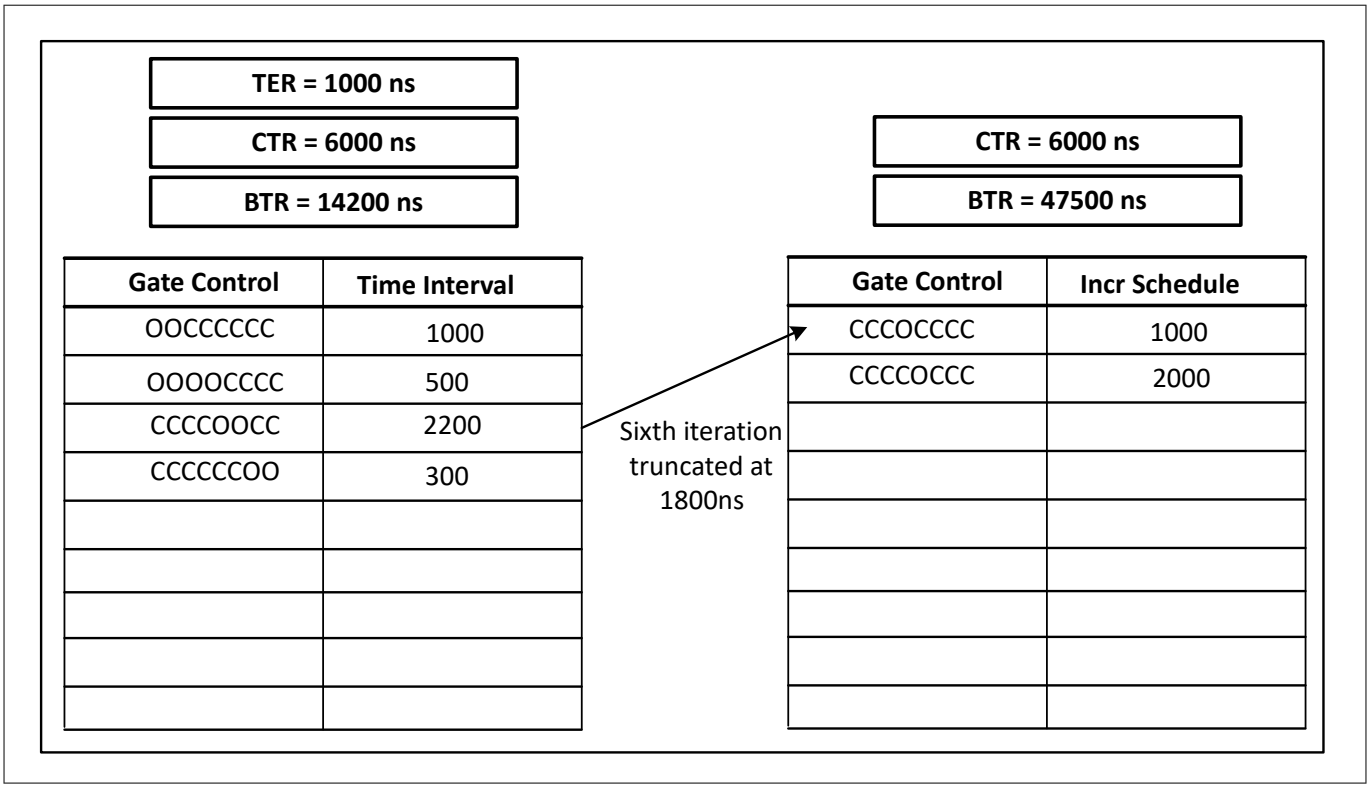


Figure 164 Switching to New List Truncating the Current List

At the end of the fifth iteration the Current time + Cycle time extension (TER) < New BTR so the sixth iteration of current configuration is started. During the sixth iteration of the current list when the new BTR value is smaller than the next schedule in the current list, it switches to the new list.

Table 734 Extending to New List By Truncating the Current List

Current Time	Gate Control Applied	Accumulator Value	BTR (with updates)
44200	OOCCCCCC	1000	44200
45200	OOOOCCCC	1500	44200
45700	CCCCOOCC	3700	44200
47500	CCCOCCCC	4000	44200
48500	CCCCOCCC	0	50200
50500	OOOOOOOO	1000	50200

In Figure 165, the current configuration list is extended instead of starting a new iteration as the extension time of 800 ns is less than the allowed cycle extension time (TER) of 1000 ns.

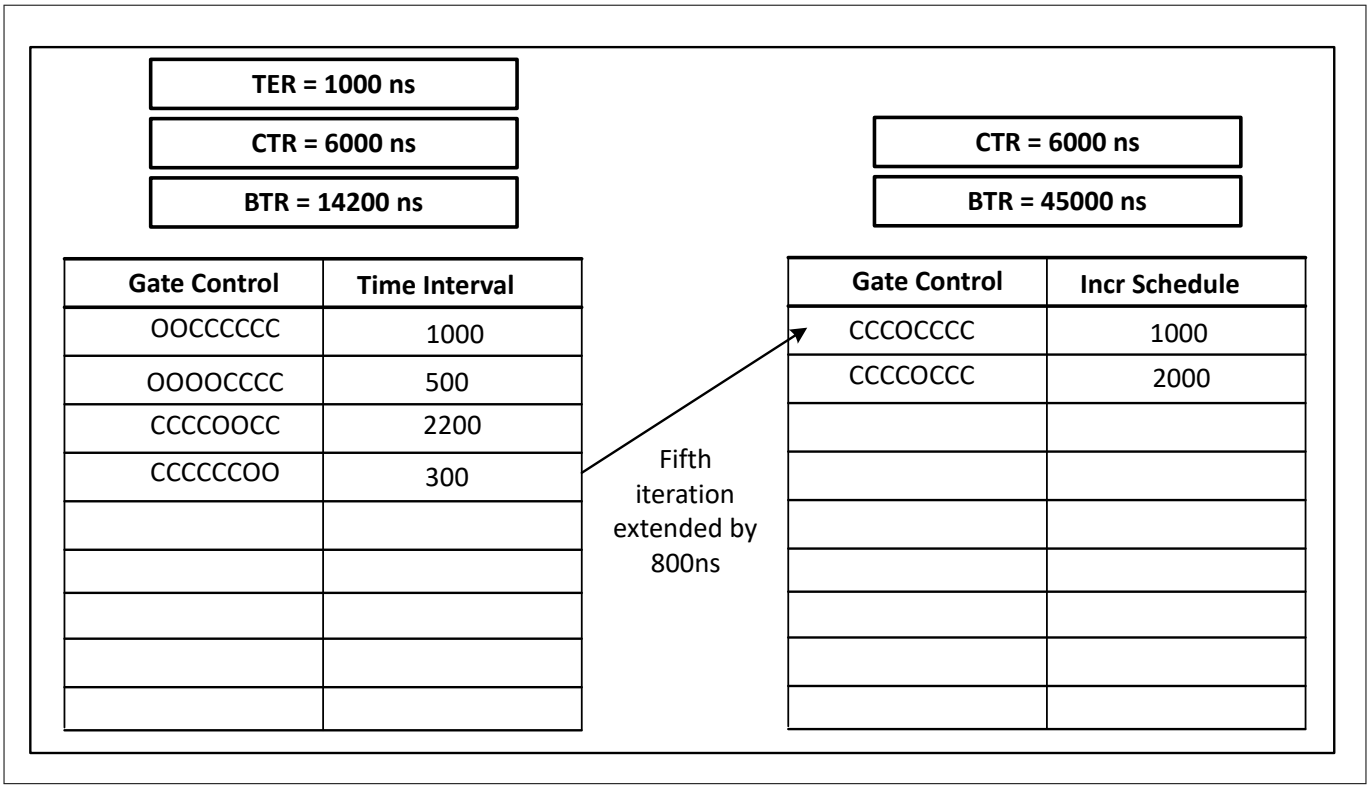


Figure 165 Switching to New List By Extending the Current List

Table 735 Switching to New List By Extending the Current List

Current Time	Gate Control Applied	Accumulator Value	BTR (with updates)
38200	OOCCCCCC	1000	38200
39200	OOOOCCCC	1500	38200
39700	CCCCOOCC	3700	38200
41900	CCCCCCOO	4000	38200
42200	OOOOOOOO	0	44200
45000	CCCOCCCC	1000	45000
46000	CCCCOCCC	3000	45000
48000	OOOOOOOO	0	51000

Impact of Transmit Scheduling Algorithms on EST

When EST is used in isolation, the Gate Control List manages the final open/close state of the Transmit Class along with the checks carried out by the Transmission Selection Algorithm in MTL. As the Gate Controls operate on a predefined repetitive schedule, it is recommended to use Strict Priority or Credit Based Shaper (CBS) scheduling algorithms.

Weighted Round Robin (WRR) implements masking of the transmit channels based on the current winning transmit channel. The algorithm is applied only among the group of transmit channels that open simultaneously. To ensure transmit channels whose gates are "Open" get priority, these algorithms are modified to treat "gate open" transmit channels and "gate closed" transmit channels as separate groups giving priority to gate open transmit channels.

For example, consider four transmit channels (TC3, TC2, TC1, TC0) with weights 4:3:2:1; TC3 & TC2 are in an Open state in one slot, while TC1 & TC0 are in an Open state in another slot. In this case, the scheduler works as follows:

1. In the first slot, the TC3 & TC2 are serviced in the ratio of 4:3 for the duration the slot is open
2. In the second slot, the transmit channels TC1 & TC0 are serviced in the ratio of 2:1
3. Fresh arbitration is started every time a slot is opened

In other words, the traffic does not get distributed in the intended ratio of 4:3:2:1; but as two groups with different ratios and only for the duration of the slot when the gates are open continuously.

When multiple queues map to one TC, XGMAC has a first-stage scheduler (Round Robin) for the queue to TC selection. When EST is enabled, XGMAC uses an enhanced Round Robin. Based on the open window size of EST, the scheduler masks the round robin for queues whose frames do not fit in the window size. This ensures that the scheduler picks the correct queue in the Round Robin, considering the frame sizes too.

Error Reported in EST

Following are the errors reported in EST

- BTR Programming Error

  This error indicates that the value of the New BTR is less than the current time. XGMAC attempts to recover from this error by recursively adding the New Cycle Time to the New BTR such that

  For more details, see the BTRL field of the MTL_EST_Status register in the *"Register Description"* chapter.
- Constant Gate Control Error

  This error occurs when the list length (LLR) is 1 and the programmed Time Interval (TI) value after the optional Left Shifting is more than or equal to the Cycle Time (CTR).

  This implies that the Gates are either always Closed or always Open based on the Gate Control values. As this is not supported, it is reported as an error.

Interrupts in EST for Stream-Gate Filtering

Program the MTL_SGF_Interrupt_Enable register appropriately, to enable interrupts. The field position in this register to enable the interrupts correspond to the field positions in the MTL_SGF_Status register. When interrupt generation is enabled, the following events trigger the generation of an interrupt.

14.4.3.21.4 Frame Preemption (FPE)

Overview of the Frame Preemption

The IEEE 802.1Qbv-2015 defines the schedule for each of the queues on every egress port which makes the implementation aware of traffic arrival schedule. This information can be used to block the lower priority traffic from transmission in this time window/slot. This ensures that scheduled traffic is forwarded from sender to receiver through all the network nodes with a deterministic delay.

One of the important requirements to achieve a low latency is to ensure there are no interfering frames during the scheduled windows that are reserved for high priority traffic. The use of scheduled traffic imposes limitations while starting a transmission.

If an interfering frame begins transmission just before the start of a reserved time period, it can extend transmission into the reserved window, and potentially interfere with higher priority traffic. Therefore, a guard band whose size is equal to the largest possible interfering frame is required before the window starts.

A larger guard band equates to a less efficient use of network bandwidth. However, with the implementation of IEEE802.1Qbu (frame preemption), this issue is addressed. Frame preemption breaks the interfering frame into

smaller fragments. Therefore, the guard band needs to be only as large as the largest possible interfering fragment instead of the largest possible interfering frame.

During the guard band, only the frames that can complete the transmission of the entire frame before the next gate close event are permitted. This ensures that the high priority traffic can always start at the beginning of the window reserved for it.

Description of the Frame Preemption

Preemption allows one or more higher priority (express) frames to interrupt the transmission of a lower priority (preemptable) frame; the preemptable frame transmission is resumed and completed after the express frame transmission is complete. To support frame preemption, the following two abstractions of the MAC are used:

- A preemptable MAC, called pMAC, which carries the preemptable traffic
- An express MAC, called eMAC, which carries the express traffic

In the implementation, only parts of the MAC that holds the state during preemption is replicated and represented as pMAC and eMAC. The MAC uses the following two ways to puts on hold, the transmission of the preemptable traffic, in the presence of express traffic:

- The MTL scheduler interrupts the preemptable traffic that is currently being transmitted

  When the preemption capability is active, the MAC interrupts the transmission and reception of preemptable frames. A preempted fragment can be followed by zero or more express frames, before the continuation fragments. The preemptable frame can be fragmented any number of times, however, the minimum final and non-final fragment length criterion must be met.

  However, interleaving of more than 1 preemptable packet is not permitted. This implies that if a preemptable packet is fragmented by an express packet, another preemptable packet cannot be transferred until all the remaining fragments of the first preempted packet are transferred.
- The MTL scheduler prevents starting the transmission of preemptable traffic

  When the preemption capability is inactive, the pMAC entity is disabled and only express traffic is transmitted or received.

Enabling the Frame Preemption

Enable the frame preemption feature by setting EFPE field of the MAC_FPE_CTRL_STS register.

GCL Modification to Support FPE

In the EST-only configuration, the GCL entry has up to 24 bits of Time Interval and up to eight high order bits representing the Gate Open/Close state requirements as shown in Figure 166.

Gate Control (up to 8 bits)	Time Interval(ns) 16, 20 , or 24 bits
OOCCCCCC	T0
OOOOCCCC	T1
CCCCOOCC	T2
CCCCCCOO	T3
|	|
|	|
OCOCOCOO	T126
OOOOCCCC	T127

Figure 166 Gate Control List When FPE is Disabled

EST only supports SetGate operation, which implies that the gates are set to either open or close at a given time interval.

However, when both Frame Preemption (FPE) and EST are enabled, the GCL also supports Set-and-Hold-MAC and Set-and-Release-MAC operations, in addition to the SetGate operation.

To enable the support of hold and release operations, the format of the GCL is slightly changed while configuring and enabling the FPE. The first Queue (Q0) is always programmed to carry preemption traffic and therefore it is always Open. The bit corresponding to Q0 is renamed as Release/Hold MAC control.

The TX Queues whose packets are preemptable are indicated by setting the PEC field of the MTL_FPE_CTRL_STS register. The GCL bit of the corresponding Queue is ignored and considered as always "Open". So, even if the software writes a "0" ("C"), it is ignored for such queues.

Gate Control (up to 7 bits)	Release/Hold Indication	Time Interval(ns) 16, 20, or 24 bits
CCCOOOO	0	T0
CCCCOOO	0	T1
OCCCOOO	1	T2
COCCOOO	1	T3
CCOCOOO	1	T4
CCCOOOO	0	T5
CCCCOOO	0	T6

Figure 167 Gate Control List When FPE is Enabled

When the Release/Hold bit transitions from a 0_B to 1_B, a Set-and-Hold-MAC operation is performed. This marks the end of the preemptable traffic. This is achieved by sending a Hold request to MTL "ha" ns in advance (where

ha is the time interval mentioned in the Hold Advance (HADV) field of the MTL_FPE_Advance register).When the Release/Hold bit transitions from a 1_B to 0_B a Set-and-Release-MAC operation is performed. This marks the resuming of the preemptable traffic. This is achieved by sending a Release request to MTL "ra" ns in advance (where ra is the time interval mentioned in the Release Advance (RADV) field of the MTL_FPE_Advance register).The preemptable traffic is blocked for the time duration the Release/Hold bit is set to a 1_B in the Gate Control List.

***Note**: The value of HADV should be equal to the time required to transmit minimum_fragment_size + IPG/EIPG + Preamble, in nanoseconds.*

Impact of Preemption on CBS

In CBS, the definition of "Transmit" is as follows:

- TRUE for the duration of frame transmission from the queue;
- FALSE when frame transmission from the queue is complete.

When CBS algorithm is used along with frame preemption, the value of "Transmit" is TRUE only while the frame is being transmitted by the MAC. If the frame transmission is delayed or interrupted (for instance, a preemptable frame transmission is interrupted to allow the transmission of an express frame from a different queue, or the start of express frame is delayed because a preemptable frame is being transmitted) the value of "Transmit" is FALSE until transmission of the frame commences or is resumed.

Also, the value of "Transmit" is FALSE during the transmission of any overhead that is a consequence of frame preemption. For example, additional frame overhead (mCRC, Fragment Count) that is added to the preemptable frame.

At any given time, if there are no frames in the queue, and the value of Transmit is FALSE, and credit is positive value, the credit value is set to zero if here is no preemptable frame from the queue for which transmission is in progress but has been interrupted.

mPacket Format

When the preemption capability is active, MAC sends mPackets to the PHY. An mPacket can be one of the following:

1. A express packet
2. A preemptable packet
3. An initial fragment of a preemptable packet
4. A continuation fragment of a preemptable packet

Figure 168 shows the format of the mPacket. It contains an express packet, a complete preemptable packet or the initial fragment of a preemptable packet. Figure 168(b) shows the format of an mPacket containing a continuation fragment of a packet.

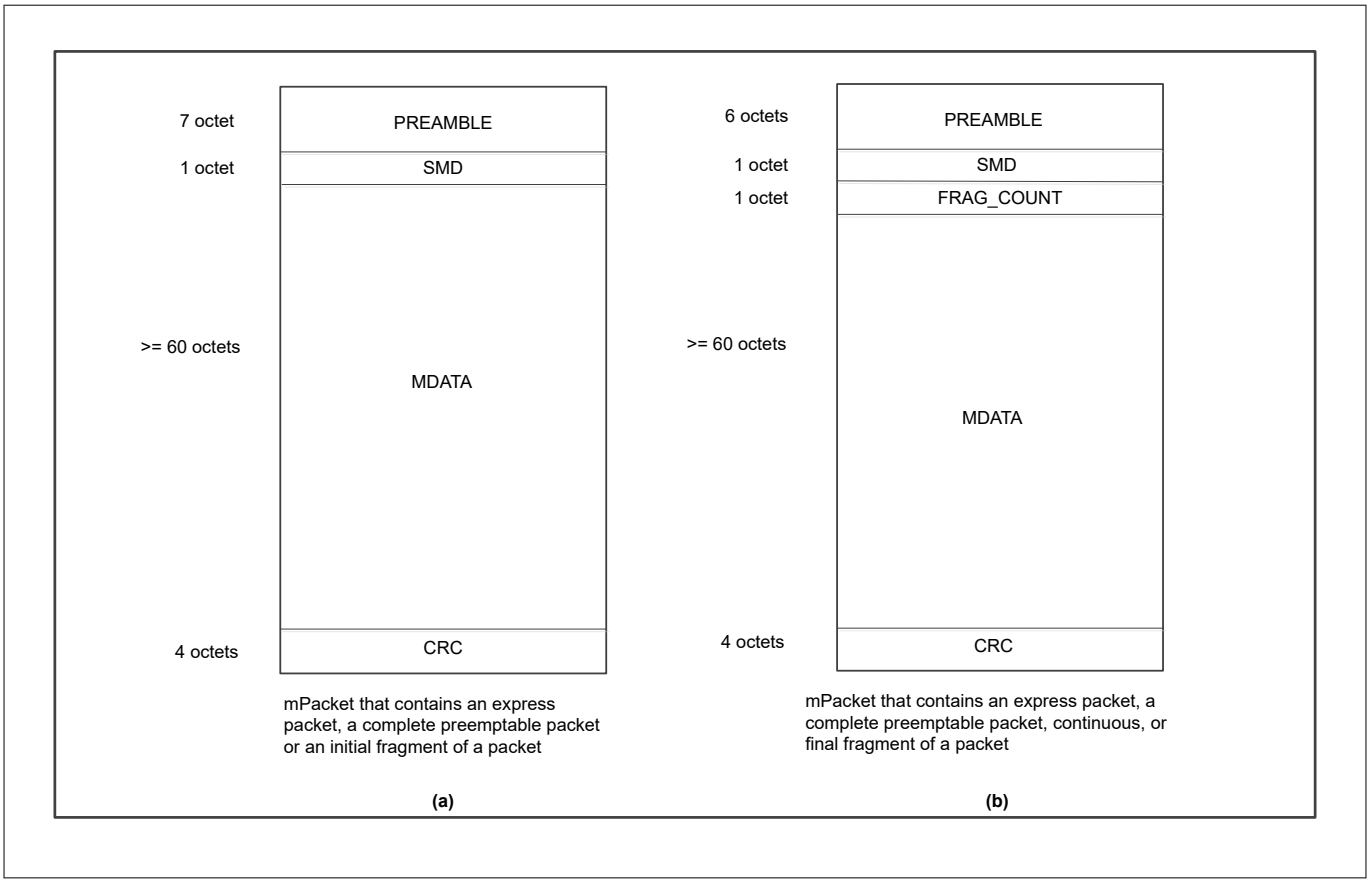


Figure 168 mPacket Formats

Preamble

The preamble in the mPacket shown in Figure 168 (a) contains seven octets. The preamble in the mPacket shown in Figure 168 (b) contains six octets. Each octet contains the value of 0x55 (transmitted in order from left to right 10101010).

Start mPacket Delimiter (SMD)

The value of the SMD indicates whether the mPacket contains an express packet, the start of a preemptable packet (initial fragment or complete packet), or any of continuation fragments of a preemptable packet. Table 736 shows the valid SMD values.

Table 736 Possible SMD Values of mPacket

mPacket Type	Notation	Frame Count	Value
verify packet	SMD-V	-	0x07
respond packet	SMD-R	-	0x19
express packet	SMD-E	-	0xD5
preemptable packet start	SMD-S0	0	0xE6
	SMD-S1	1	0x4C
	SMD-S2	2	0x7F
	SMD-S3	3	0xB3
continuation fragment	SMD-C0	0	0x61

(table continues...)

Table 736 (continued) Possible SMD Values of mPacket

mPacket Type	Notation	Frame Count	Value
	SMD-C1	1	0x52
	SMD-C2	2	0x9E
	SMD-C3	3	0x2A

frag_count

A frag_count is a modulo-4 counter that increments for each continuation fragment of the preemptable packet. The frag_count protects against mPacket reassembly errors by enabling detection of the loss of up to 3 packet fragments.

The frag_count field is present only in mPackets with SMD-C notation (continuation fragment). The frag_count is zero in the first continuation fragment of each preemptable packet.

Table 737 Possible frag_count Values

frag_count	Value
0	0xE6
1	0x4C
2	0x7F
3	0xB3

mData

The contents of the packet from the MAC, starting with the first byte after the SFD to the last byte before the FCS are sent in the mData fields of one or more mPackets for that frame. The minimum size of the mData field is 60 bytes.

CRC

The CRC field contains a cyclic redundancy check (CRC) and has an indication of the final mPacket of a frame. In the final mPacket of a frame, the CRC field contains the last 4 octets of the MAC frame (the FCS field).

For other mPackets, the CRC field contains an mCRC value. The mCRC is calculated on the octets of the packet from the first octet of the frame (the octet following the SFD of preemption frames) to the last octet of the packet transmitted in that mPacket by performing an XOR of the calculated 32 bit CRC value of the fragment and a value of 0x0000FFFF.

Summary of Packet Formats

- Express Packet
  7 bytes of PREAMBLE, SMD-E, Data, and CRC
- Complete Preemptable Packet
  7 bytes of PREAMBLE, <Current Preemptable packet SMD>, Data, CRC
- Initial Fragment (non-final) of Preemptable Packet
  7 bytes of PREAMBLE, <Current Preemptable packet SMD>, Data, mCRC
- Continuation fragments (non-final) of Preemptable packet
  6 bytes of PREAMBLE, <Current Preemptable continuation fragment SMD>, <Current Preemptable continuation fragment FC>, Data, mCRC
- Final fragment of Preemptable packet
  6 bytes of PREAMBLE, <Current Preemptable continuation fragment SMD>, <Current Preemptable continuation fragment FC>, Data, CRC

***Note**: When Frame Preemption is enabled, the MAC receiver communicates the exceptions related to received preempted fragments (incorrect fragment sequencing, missed fragment, and so on) to MTL layer by reporting it as CRC error. Software should not set the DCRCC (Disable CRC Checking for Received Packets) bit of MAC_Ext_Configuration register to 1; otherwise the MTL layer might forward the unintended preempted fragments to the application.*

Table 738 Current and Previous SMD Values for Preemtable Packet

Previous Preemptable packet SMD	Current Preemptable packet SMD
SMD-S0	SMD-S1
SMD-S1	SMD-S2
SMD-S2	SMD-S3
SMD-S3	SMD-S0

Table 739 Current and Previous SMD Values for Premptable Fragment

Previous Preemptable fragment SMD	Previous Preemptable fragment FC	Current Preemptable continuation fragment SMD	Current Preemptable continuation fragment FC
SMD-S0	NA	SMD-C0	FC0
SMD-S1	NA	SMD-C1	FC0
SMD-S2	NA	SMD-C2	FC0
SMD-S3	NA	SMD-C3	FC0
SMD-C0	FC0	SMD-C0	FC1
SMD-C0	FC1	SMD-C0	FC2
SMD-C0	FC2	SMD-C0	FC3
SMD-C0	FC3	SMD-C0	FC0
SMD-C1	FC0	SMD-C1	FC1
SMD-C1	FC1	SMD-C1	FC2
SMD-C1	FC2	SMD-C1	FC3
SMD-C1	FC3	SMD-C1	FC0
SMD-C2	FC0	SMD-C2	FC1
SMD-C2	FC1	SMD-C2	FC2
SMD-C2	FC2	SMD-C2	FC3
SMD-C2	FC3	SMD-C2	FC0
SMD-C3	FC0	SMD-C3	FC1
SMD-C3	FC1	SMD-C3	FC2
SMD-C3	FC2	SMD-C3	FC3
SMD-C3	FC3	SMD-C3	FC0

Transmit Preemption

MTL Tx Preemption

To support preemption MAC should have more than 1 TX queue with at least 1 Queue designated as Express Queue. (and 1 queue designated as preemption queue). When FPE is enabled (setting EFPE=1 in MAC_FPE_CTRL_STS register), the MTL preempts a preemptable frame, when a “hold” request is asserted (EST/Qbv configured and enabled) express frames are available for transmission, that is, frame is present in MTL FIFO and is qualified for arbitration, after ensuring that the minimum mPacket mData field size is met. Therefore, preemption occurs only if at least 60 bytes of the preemptable frame have been transmitted and at least 64 bytes (including the frame CRC) remain to be transmitted.

The earliest starting position of preemption is controlled by the AFSZ field of the MTL_ FPE_CTRL_STS Register. Preemption does not occur until at least 64 * (1+ AFSZ) - 4 bytes of the preemptable frame have been sent.

When preemption occurs, all the preemptable queues are blocked and only the express queues are allowed to arbitrate (if more than one express queue has traffic) and transmit.

Continuation fragment of the preempted frame is the first frame to be transmitted after “release” request is asserted (EST/Qbv configured and enabled) and all the express traffic transmission completes.

***Note**: All the PTP packets should be transmitted as express packets.*

MTL communicates the following frame-type information to the MAC using a 2-bit preemption control signal on the MTI interface qualified by SoF and EoF.

- Express Frame
- Preemption Frame (Full or Fragment)
- Continuation Fragment (non-Final or Final)

Table 740 Preemption Control Values on MTI Interface for Various Frame Types

Qualifier	Preemption Control Value	Frame Type
SoF	00	Start Express
SoF	01	Start Preemption
SoF	10	Continuation Fragment
SoF	11	Reserved
EoF	00	End of Frame
EoF	01	End of Fragment

If the mti_sof_o and mti_eof_o are sent in the same cycle on the MTI interface, the mti_preemp_ctrl (Preemption Control Value) for EoF is taken as "00", End of Frame.

MTL should wait for the previous fragment status to be received before resuming the continuation fragments of a preempted frame. Fragment status is described in detail in the Tx Fragment Status section

MAC Tx Preemption

MAC supports preemption by implementing the functionality needed to generate the mPackets as described in the section "mPacket Format".

Based on the Preemption Control value received on the MTI interface (qualified with SoF and EoF), MAC determines the frame type (shown in the table "Possible SMD values of mPacket") and generates mPackets accordingly.

When the preemption capability is active, MAC replaces the SFD of a preemption packet with an SMD-S value. A 2-bit rolling frame count is encoded in the SMD-S value.

The SMD-E value is the same as the SFD value so the SFD of an express packet does not need to be replaced.

Related information

mPacket Format on page 1537

Tx Fragment Status

MAC sends Tx fragment status to indicate successful transmission of fragmented mPackets. To indicate a fragment status, bit-31 of the mti_txstatus_o field is set and all other values in the status field is set to '0'.

In case of a transmission error (underflow, jabber, and so on) the frame status is sent (and not a fragment status) with an error indication along with all other relevant status fields. The bit-31 of mti_txstatus_o is set to '0' to indicate a Frame Status. In case of receiving an error status for a transmitted fragment, the MTL drops the remaining fragments and does not send any more continuation fragments.

Receive Preemption

MAC Receive Preemption

When FPE is enabled, the MAC Receiver pases the incoming packets and differentiates between Express packets and preemptable packets. An SMD containing an SMD-E indicates express packet, and SMD containing an SMD-S indicates the first mPacket of a preemptable packet.

If an mPacket containing an SMD-S is received when MAC has not completed receiving the previous preempted packet, MAC sets a CRC Error status for the previously received partial packet.

When an SMD-S is detected, MAC records the frame count indicated by the SMD and then begins sending data on the MRI interface.

The MAC checks the last four bytes of the mPacket . If the last four bytes of the mPacket do not match CRC, that indicates the end of the packet with or without a CRC error as per the CRC check result. If the last four octets of the mPacket match, that indicates that the packet was preempted.

An SMD containing an SMD-C indicates an mPacket that continues the data for a preempted packet. Upon receiving an SMD value of SMD-C, MAC checks the following:

1. A preempted packet is in progress
2. The frame count indicated by the SMD matches the frame count of the packet in progress
3. The frag_count value indicates the next fragment count.

If any of these checks fail, the mPacket is discarded and MAC sets a CRC Error status for the partially received packet.

If all the checks pass, the next fragment count is incremented modulo 4.

When a packet is preempted, the MAC saves the state of the partially received packet (filter check status, timestamp, length fields and so on) and processes the received Express packets until the continuation fragment is received.

The MAC Receiver sends a "dummy status" (all zeros in mri_rxstatus_o) for all the mPacket fragments successfully received and sends the Rx status with the final fragment. If an error is detected during any of the fragments the Rx status is sent and the fragment is marked as final fragment. All subsequent continuation fragments received for this packet are dropped in the MAC.

The MAC communicates the following frame type information to the MTL using a 2-bit preemption control signal on the MRI interface qualified by mri_sof_o and mri_eof_o.

1. Express Frame
2. Preemption Frame (Full or Fragment)
3. Continuation Fragment (non-Final or Final)

Table 741 Preemption Control Values on MRI Interface for Various Frame Types

Qualifier	Preemption Control Value	Frame Type
SoF	00	Start Express
SoF	01	Start Preemption
SoF	10	Continuation Fragment
SoF	11	Reserved
EoF	00	End of Frame
EoF	01	End of Fragment

If the mri_sof_o and mri_eof_o are sent in the same cycle on the MRI interface, the mri_preemp_ctrl (Preemption Control Value) for EoF is taken as '0' indicating End of Frame.

***Note**: Minimum Preemptive fragment size supported on Rx is 48 bytes*

Data Alignment

When a received frame cannot be fragmented on any byte boundary, MAC retains the unaligned bytes of data in the previous fragment and resends them with the next fragment as shown in Figure 169.

MTL can choose to ignore the partial data (based on Byte Enable value) that comes with End of Fragment and instead use the "data width aligned" data received in the first beat of the next continuation fragment.

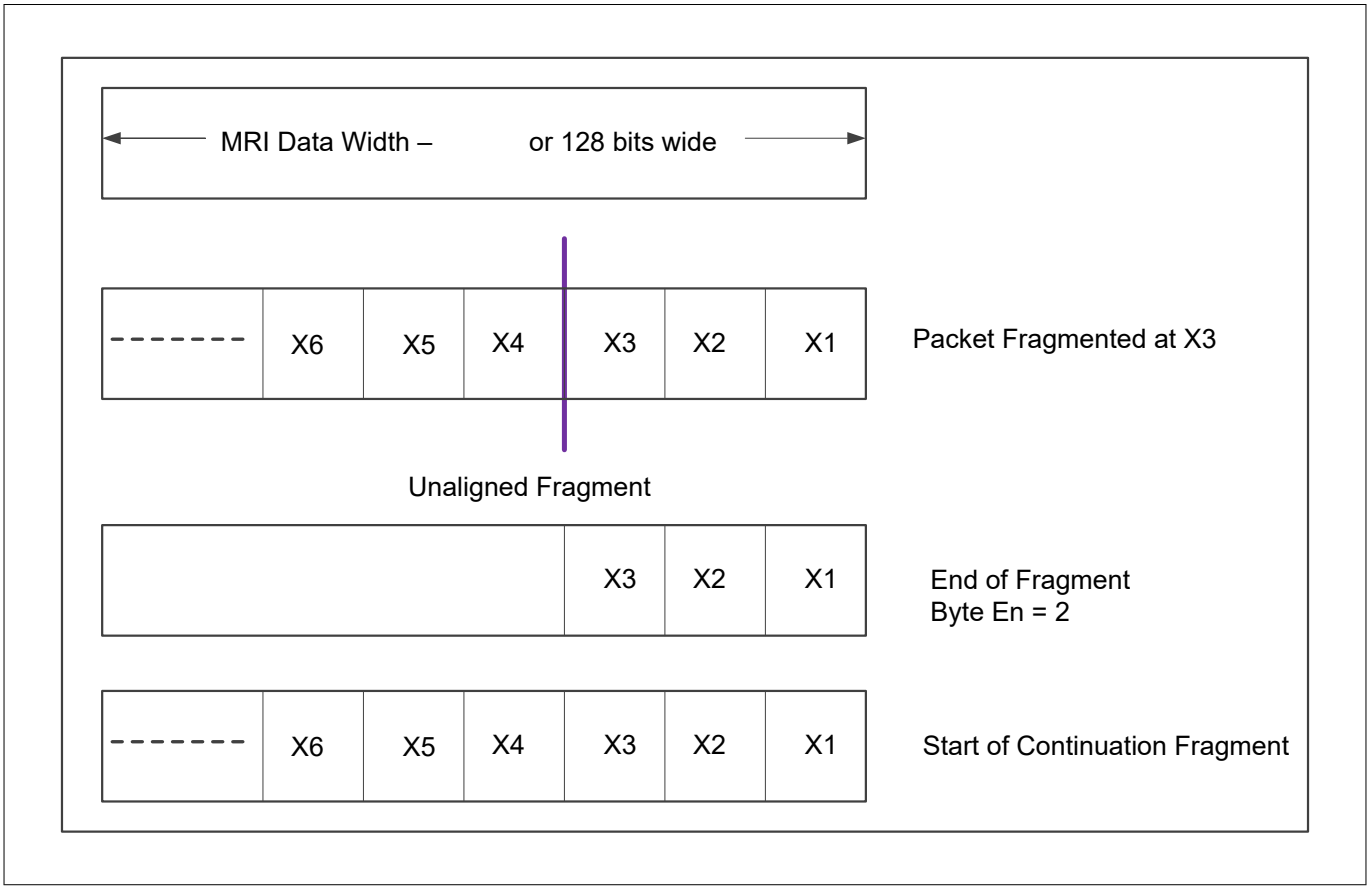


Figure 169 Data Alignment Feature of MAC

MTL Receive Preemption

The MTL RX must have at least 2 Receive Queues to support FPE function as the preemptable packets and the express packets must be routed to separate RX Queues. The destination RX Queue of a received packet is controlled by MAC_RxQ_Ctrl(#i) registers. These must be carefully programmed such that Preemptable traffic and express traffic are not routed to the same RX queue. As the Queue mapping for tagged packets is based on VLAN user-priority field, this means that priority of preemptable and express packets are exclusive to each other. In other words, packets of a certain priority (traffic class) are either express or preemptable but cannot be both.

If MCBCQEN of MAC_RxQ_Ctrl1 register is set, preemptive Multicast/Broadcast packets are routed to PMCBCQ as programmed in MAC_RxQ_Ctrl4. Therefore, ensure that PMCBCQ is not same as any of the queues programmed for express queues.

Preemption traffic routing is carried out as shown in Table 742.

Table 742 Preemption Control Values on MRI Interface for Various Frame Types

Preemption Packet Type	Queue Routing
Multicast and Broadcast Preemptive Frames passing the SA/DA and VLAN filters with MCBCQEN field of MAC_RxQ_Ctrl1 register is set	PMCBCQ[1)]
AV Tagged Packets passing the SA/DA and VLAN filters	PSRQ[2)]
Generic Tagged Packets passing the SA/DA and VLAN filters	PSRQ[3)]
Tagged Packets failing the SA/DA and VLAN filters without setting the Receive All (RA) or setting VTFE	Dropped
All other packets	RQ

1) Only if the queue is enabled for AV feature
2) Only if the queue is enabled for generic VLAN feature
3) Only if the queue is enabled for AV feature

In addition to the PSRQ (Priority Selected in Rx Queue) based Queue routing and PMCBCQ (preemptive multicast/broadcast queue), a programmable “Residue Queue” (RQ) is provided to route all other preemptive packets received (Untagged, SA/ DA or VLAN Filter Fails forwarded due to RA (receive all) being set or VTFE being reset). In case of filter fail, Queue 0 is used as the default queue. So Q0 should not be used by the preemptive traffic routing. Therefore, RQ cannot be programmed as 0.

MTL Receive Arbitration

On the ARI Interface, data is fetched from the MTL Rx FIFO based on the arbitration selected in the MTL_Operation_Mode register. Frame based arbitration can be used only when all the MTL Preemption Queues are operating in Store and Forward mode. Otherwise there is loss of bandwidth on the ARI interface because all the fragments of a preemptable packet are not available. Therefore, express packets received between the fragments are blocked until all the fragments are received and transferred; this defeats the purpose of express traffic.

When operating in either Threshold (cut through) or Store and forward modes of operation, PBL based arbitration is recommended over Frame based arbitration. In case of PBL based arbitration, the watermark check is always performed and the arbitration/transfer of data in terms of chunks of “PBL” size of data which is almost similar to the concepts of “fragments”. Therefore the express queue packets get blocked for less time as well as the ARI interface transfers the data without loss of efficiency.

Received preemption frames support in offload engines

When Frame Preemption is enabled, all PMT, PTP and PAUSE packets must be received as express/normal packets. These offload engine logic and functions do not recognize preemption frames.

Verify and Respond mPackets

When FPE function is present, the MAC can receive and detect the Verify and Respond mPackets, even when FPE is not enabled by software. When MAC detects valid Verify/Respond mPackets, it notifies the software by setting the RVER and RRSP fields of MAC_FPE_CTRL_STS register respectively. Optionally an interrupt can be generated. As such packets have empty (all-zero) data payload, they are dropped inside the MAC and not forwarded to the MRI.

Software can set the SVER and SRSP fields of MAC_FPE_CTRL_STS register to request MAC to transmit Verify and Respond mPackets respectively. Upon successful transmission of these frames, MAC clears the SVER/SRSP bits and sets the TVER & TRSP fields of MAC_FPE_CTRL_STS register. Optionally an interrupt can be generated when these events occur.

When Frame Preemption is enabled,

- For the preemption frames received with CRC error indication, ignore the following status fields
  - Packet length
  - Dribble error indication
  - Runt frame indication
- The preemption frames received with CRC error and with sizes ranging from 64 to 67 bytes are treated as runt packets. So, the corresponding MMC Rx Frame Counter for Runt Frames is incremented
- Do not disable CRC checking on receive. This is required because, internally, FPE error fragments are identified using CRC checks. For more details, see the DCRCC field in the MAC_Ext_Configuration Register in the section "Registers"
- Forward Undersized Good Packets (FUP) field of MTL_RxQ_Operation_Mode register should not be set. A new programmable mode bit (ARV) to autogenerate a Respond packet up on receiving a Verify packet is added

Related information

Registers on page 802

Frame Preemption and MMC Counter and Interrupt Registers

The following MMC counters and associated Interrupt registers are instantiated/present in the MAC when Frame Preemption feature and any one of the existing MMC counters are selected during RTL configuration.

Table 743 MMC Counters and Associated Interrupt Registers

Frame Assembly Error Counter	Description	Associated MMC Counter
Frame Assembly Error Counter	A 32-bit counter that provides the number of MAC frames with reassembly errors on the Receiver, due to mismatch in the Fragment Count value.	MMC_Rx_Packet_Assembly_Err_Cntr
Frame SMD Error Counter	A 32-bit counter that provides the number of received MAC frames rejected due to arriving with an SMD-C when there was no preceding preempted frame	MMC_Rx_Packet_SMD_Err_Cntr

(table continues...)

Table 743 (continued) MMC Counters and Associated Interrupt Registers

Frame Assembly Error Counter	Description	Associated MMC Counter
Frame Assembly OK Counter	A 32-bit counter that provides the number of MAC frames that were successfully reassembled	MMC_Rx_Packet_Assembly_Ok_Cntr
MAC Rx Fragment Counter	A 32-bit counter that provides the number of additional mPackets received due to preemption	MMC_Rx_FPE_Fragment_Cntr
MAC Tx Fragment Counter	A 32-bit counter that provides the number of additional mPackets transmitted due to preemption	MMC_Tx_FPE_Fragment_Cntr
Hold Request Counter	A 32-bit counter that maintains the count of number of times a hold request is given to MAC	MMC_Tx_Hold_Req_Cntr

Additional Registers Associated With MMC Interrupts

Following are the additional registers associated with MMC interrupts for the MMC error counters:

- MMC_FPE_Tx_Interrupt
- MMC_FPE_Tx_Interrupt_Mask
- MMC_Rx_Interrupt
- MMC_FPE_Rx_Interrupt_Mask

14.4.3.21.5 Time-Based Scheduling

Time-based scheduling feature is suitable for traffic whose periodicity and rate are predictable. This is an optional feature. To improve the quality-of-service of such traffic,

- The transmit DMA fetches the packet from the host memory for transmission at designated time. This helps the software to setup the Transmit descriptors in advance even before packet is ready/available. It reduces the overhead on the software and avoids constant monitoring of the time and preparing descriptors just in time when the packet is targeted to be transmitted.
- The MAC transmits the packet only at the designated/pre-determined time even if the packets are fetched in advance. This helps in maintaining a constant transmission rate that can be consumed by the receiver station; therefore avoiding congestion and excessive buffering in the network.

Description of Time-Based Scheduling

The time-based scheduling feature supports fetching and launching an Ethernet packet at (or after) a pre-determined time. The time-based scheduling is supported only in the following modes/configurations:

- Full duplex mode
- Link speed is 100 Mbps or higher
- Time-stamping is enabled.

***Note**:*

- *Do not enable time-based scheduling (or enhanced descriptors) for the channel for which transmit timestamp or one-step timestamping control correction feature is enabled.*
- *When fetch time is enabled, overrides the audio-video slot function.*
- *In EST enabled configurations, the ESTM mode bit setting impacts the time-based scheduling feature only when the DUT processes the GCL list.*
- *When an unaligned new GCL list (with a different CTR) is installed, ensure that there is no traffic during the installation of the new list. Traffic during the switching of the lists might result in unpredictable behavior regarding Fetch, Launch, and Launch expiry as the CTR and BTR values are updated while the frame is being processed.*

Definitions Used

Following are the definitions specific to the time-based scheduling feature:

- Launch Time

  The time beyond which MTL can schedule the packet for transmission
- Fetch Time

  The time beyond which the Tx DMA can schedule a packet-fetch from the host memory
- Expiry Time

  The time beyond which the packet is dropped by MTL

Launch Time

The launch time is specified in the Transmit Enhanced Descriptor in the DMA configurations. For more details, see the section "Enhanced Descriptor for Time-Based Scheduling".

The launch time can also be specified as a Control Word for each packet in the MTL configurations.

When the launch time is specified in the DMA configuration, it is valid only for the specific frame that is scheduled for transmission. The launch time is reset after the transmission of one frame.

Following are the two formats of the launch time.

- Normal/Absolute Format
- EST/Offset Format

Normal/Absolute Format

In this format, the Launch time is an absolute time value at which the packet is launched for transmission. The launch time is interpreted in the Normal/Absolute format if the ESTM bit of the MTL_TBS_CTRL register is set to 0.

The launch time is a 32-bit value, where most-significant 8-bits represent the time in seconds and the rest 24 bits represent the time in 256 ns. The launch time is compared against the IEEE 1588 based System/PTP Time (bits[39:8]) and rolls over after 256 seconds.

The maximum value of the lower 32-bits of System Time is 999,999,999 decimal (0x3B9AC9FF) and it wraps to 0 when reaching this value (representing a full second). Therefore, the maximum value of the lower 24 bits of the Launch Time (after multiplying by 256) must be 0x3B9AC9.

As the maximum value of Launch time is 256 seconds,

- Launch Time is greater than current System Time when its value is between System Time[39:8] and System Time[39:8] + 128 sec
- Launch Time is less than current System Time when its value is between "System Time[39:8] + 128sec" and "System Time[39:8] + 256 sec", because this is a modulo 256 computation

EST/Offset Format

In this format, the Launch time is a offset value relative to the time indicated by the Base Time Register (BTR) of the GCL list provided in the GCL Slot Number (GSN). The value in the BTR is always updated to the start time of the current loop of the GCL.

For each packet, the GSN value and the Launch Time value are specified in the Transmit Context Descriptor in the DMA configurations, or as Control Word in the MTL configurations.

The launch time offset is a 32-bit value; the upper 8-bits represent the time in seconds and the rest 24 bits represent the time in 256 ns, which is added to the BTR value corresponding to GCL Slot Number. The value of the Launch Time Offset must be smaller than the value of the Cycle Time (specified in the CTR register that is implemented when EST is enabled).

If the CTR is greater than or equal to 1s, the maximum value of the lower-24 bits of Launch Time offset should be 999,999,999 decimal (0x3B9AC9FF).

In this format, the launch time is a 64-bit value, which is interpreted as,

Launch Time = Launch GSN BTR[63:0] + (Launch Time Offset[31:0] << 8), which is compared with the System Time [63:0].

GSN BTR is the Base Time value at which the Launch GSN loop started execution.

GCL Slot Number (GSN)

A modulo 16 count of the GCL loop count is implemented and known as GSN. The count is incremented for every new GCL loop; installation of new GCL list does not impact the count. The current GCL slot number can be obtained by reading the CGSN field of the MTL_EST_Status register.

The maximum value of GSN is 15. Therefore GSN values between Current GSN (CGSN field of MTL_EST_Status register) and CGSN+8 represent current or future slots; all other GSN values are interpreted as elapsed slots or past slots. So, for the correct interpretation of time, GSN value must be between CGSN and CGSN+8, for correct interpretation of time.

Launch Expiry Time

Normal/Absolute Mode

In the normal mode, when the LEOV (Launch Expiry Offset Valid bit of MTL_TBS_CTRL register) is set, the Launch Expiry Offset (LEOS field of MTL_TBS_CTRL register) determines the maximum amount of time a frame is eligible for launch, starting from the time the frame becomes eligible for launch.

The Launch Expiry Offset is a 24-bit value defined in 256ns units, with a maximum possible value of 999,999,999 ns (0x3B9AC9FF).

Launch Expiry Time = (Launch Time[39:8] + LEOS[32:8]) * 256ns

The packet with a specific Launch time is considered eligible for transmission when the Launch time is less than the System Time and (if LEOV is set) the System Time is less than the Launch Expiry time.

When the System time is greater than the Launch Expiry time, the frame is categorized as expired and is dropped from the MTL FIFO.

***Notes**:*

1. *For correct interpretation and meaningful operation, the Fetch, Launch, and Launch Expiry time should never be set to a value larger than Current System Time + 128 seconds; such a value is interpreted as time that has already elapsed*
2. *In full duplex mode, the frames dropped from the MTL FIFO have Error Summary (Bit 15) and TBS Drop (bit 3) of Transmit Status set*

EST/Offset Mode

In the EST mode of operation, when the LEOV field of the MTL_TBS_CTRL register is set, the launch expiry GSN offset (LEGOS field of the MTL_TBS_CTRL register) and Launch Expiry Offset (LEOS field of MTL_TBS_CTRL register) determine the maximum amount of time a frame remains eligible for launch, starting from the time the frame becomes eligible for launch.

Launch Expiry Offset = (LEGOS: LEOS)

LEGOS holds the GSN offset (multiples of CTR time) and LEOS holds maximum value of CTR (sub CTR values) in ns.

Launch Expiry offset is a 24-bit value defined in the units of 256 ns, with a maximum possible value of the smaller of 999,999,999ns or CTR-1 ns.

The Launch Expiry GSN is computed as follows:

Launch Expiry GSN = (Launch GSN + LEGOS + CCMA)

Where,

CCMA is the CTR Carry due to Modulo addition. This value is 1 if ((Launch Time Offset + LEOS) << 8) is equal to or greater than CTR. When CCMA = 1,

Launch Expiry Offset = (Launch Time Offset+ LEOS) - CTR

When CCMA = 0,

Launch Expiry Offset = (Launch Time Offset + LEOS).

Launch Expiry Time = Launch Expiry GSN BTR[63:0] + Launch Expiry Time Offset.

When LEOV is not set, Launch Expiry Time is not checked.

When LEOV is set, and

- the system time is greater than the Launch Expiry time, the frame is dropped from MTL FIFO. The frame is considered as expired
- the launch time is smaller than the System Time and Launch Expiry Time is greater than system time, the frame is considered eligible for launch

***Notes**:*

1. *Max value of LEGOS is 7. This implies that when LEOV is set, the frame has a maximum life time of <8 GCL loop iterations after it becomes eligible for Launch*

2. *The slot number of the First GCL List executed each time after EST is enabled, is zero*

3. *All computation in TBS EST mode is done with a granularity of 256 ns*

Fetch Time

The Fetch Time can also be indicated for each packet by the software. This is done by setting FTOV field of DMA_TBS_CTRLn (where n = 0, 1, 2, 3) register.

If the FTOV filed is not set, the Fetch Time Offset is not valid and the DMA fetches packets without any time constraints.

The fetch time accounts for all possible delays in the DMA fetch operation and ensures that the frame is present in the MTL FIFO before the launch time.

***Note**:* *The application must program the respective channel's TFSEL bit of DMA_CH(#i)_TX_Control register to select the specific DMA_TBS_CTRLn register to use.*

Normal/Absolute Mode

In the Normal mode fetch time derived/calculated by reducing the time specified in the Fetch Time Offset (FTOS) field of DMA_TBS_CTRL register from the given Launch time.

Normal mode of operation, the Fetch Launch Time is computed as:

Fetch Time[39:8] = (Launch Time[39:8] - FTOS[31:8])

The Fetch time is 32-bits and is compared against System Time[39:8] to determine the Eligible for fetching the frame:

- The Fetch time is defined as "greater" than System Time if the Fetch Time is in the range of "System Time[39:8]" and "System Time[39:8] + 128 seconds". The frame is considered as not-eligible for fetch
- The Fetch time is defined as "smaller" than the System Time if the Fetch time is in the range of "System Time[39:8] + 128 sec" and "System Time[39:8] + 256 seconds". The frame is considered as eligible for fetch

This is a modulo 256 computation.

EST/Offset Mode

In the EST/Offset mode, the Fetch GSN Offset (FGOS field of MTL_TBS_CTRL register) provides the Slot Number offset to be deducted from the Launch GSN. In this case the FTOS value should be the smaller of 999,999,999ns or CTR-1 ns.

If (Launch Time Offset >= FTOS):

Fetch Time Offset = ((Launch Time Offset - FTOS) * 256ns)

CBFS(CTR Borrow for Fetch Subtraction) = 0

If (Launch Time Offset < FTOS)

Fetch Time Offset = CTR + ((Launch Time Offset - FTOS) * 256ns)

CBFS (CTR Borrow for Fetch Subtraction) = 1

The Fetch GSN is computed as follows:

Fetch GSN = Launch GSN - FGOS - CBFS

Fetch Time = Fetch GSN BTR[63:0] + Fetch Time Offset

The frame is marked eligible for DMA fetch when the fetch time is smaller than the System Time.

Note:
- *Maximum value of FGOS is 7. This implies that when FTOV is set, the frame can be fetched at a maximum of < 8 GCL loop iterations before it becomes eligible for Launch*
- *An interrupt is generated when a frame is dropped due to Launch time expiry. For more details, see the section "Registers"*

DMA Operations Sequence

Enhanced Descriptor should be used for Time-Based Scheduling. For more details, see the section "Enhanced Descriptor for Time-Based Scheduling.

Following is the sequence of operation when FTOV = 1.

1. Fetch the first Transmit Enhanced Normal Descriptor. (Check the FD in the Normal Descriptor)
2. If LTV is set and Fetch is enabled, compute the Fetch Time based on the Launch Time. Wait for the System Time to be greater than Fetch Time
3. Read the Frame (Data) from the host memory and transfer to MTL FIFO
4. Close the Enhanced Normal Descriptor
5. Fetch the next Normal Descriptor (if the previous Descriptor was not the Last)
6. Repeat steps 4 to 6, until the last descriptor of the frame (LD is set)

After the Last Descriptor of a frame, program the next Transmit Enhanced Normal Descriptor with a new Launch Time and with LTV bit set. Otherwise, the subsequent frames are processed without any time restrictions.

Control Word (MTL Configurations)

Time-based scheduling ensures that a packet is allowed to arbitrate for transmission only if the System Time is greater than the Launch Time.

In MTL configuration,

- Launch Time and GSN, use the same field that is used to hold the 64-bit timestamp value for OSTC (One Step Timestamping Correction). If the LTV (Bit 31, TBSDW0) bit is set and OSTC and OSTPA bits are not set (bit 19 and 15 of first control word), the Time Stamp field is interpreted as Launch Time
- The last two Control D-words is considered to the Time Stamp for OSTC, when bit-19 of Control-Word0 (OSTC) is set

When the MTL TXFIFO Read Controller reads the first control word, both bit 19 and bit 15 are reset, it fetches the next control word. When the next control word's bit 31 (TBSDW0) is set, the frame has a valid Launch Time (as per the description of TBSDW0 and TBSDW1). If bit-31 is set, it reads the subsequent control words to get the Launch Time. F.

Impact on Transmission Selection Algorithm

Time-Based Scheduler does not directly influence the Transmission Selection Algorithm but has an indirect influence as it determines if a queue is eligible to participate in the scheduling.

If a frame of a specific queue has a valid Launch Time, the Time Based Scheduler ensures that the frame participates in Transmit Scheduling (TRC Scheduler) only after the Launch time has elapsed.

The gating done by Time-Base scheduler is only for the Scheduling (picking frame for Transmission). It might not alter any other characteristics and behavior of the frame. For example, the Frame waiting for the Launch time to lapse continues to gain credits in CBS. The CBS credits are not lost, as the frame is marked as available but not ready. Strict Priority or CBS are recommended Transmit Scheduling algorithms for the TBS implementation because, scheduling for TBS enabled queues are more predictable.

Related information

Enhanced descriptor for time-based scheduling on page 1586
Transmit Context Descriptor on page 1566
Registers on page 802

14.4.3.22 Automotive safety features

This chapter describes the automotive safety features.

***Note**: On chip data path parity and AXI ECC are not connected in this device.*

Related information

Description of ECC protection for memories on page 1551
Finite State Machine (FSM) parity and timeout protection on page 1556
Application/CSR interface timeout protection on page 1558

14.4.3.22.1 Description of ECC protection for memories

The Error Correction Code (ECC) block can correct single-bit error and detect double-bit error.

At the write interface, ECC check bits are generated by computing ECC on the contents of the data bus and respective address is appended with the data that is written to the memory.

At the read interface, ECC check bits are recomputed on the content of the read data and the respective address is compared with the received check bits in the memory.

The figure below shows the block diagram of ECC protection for memories

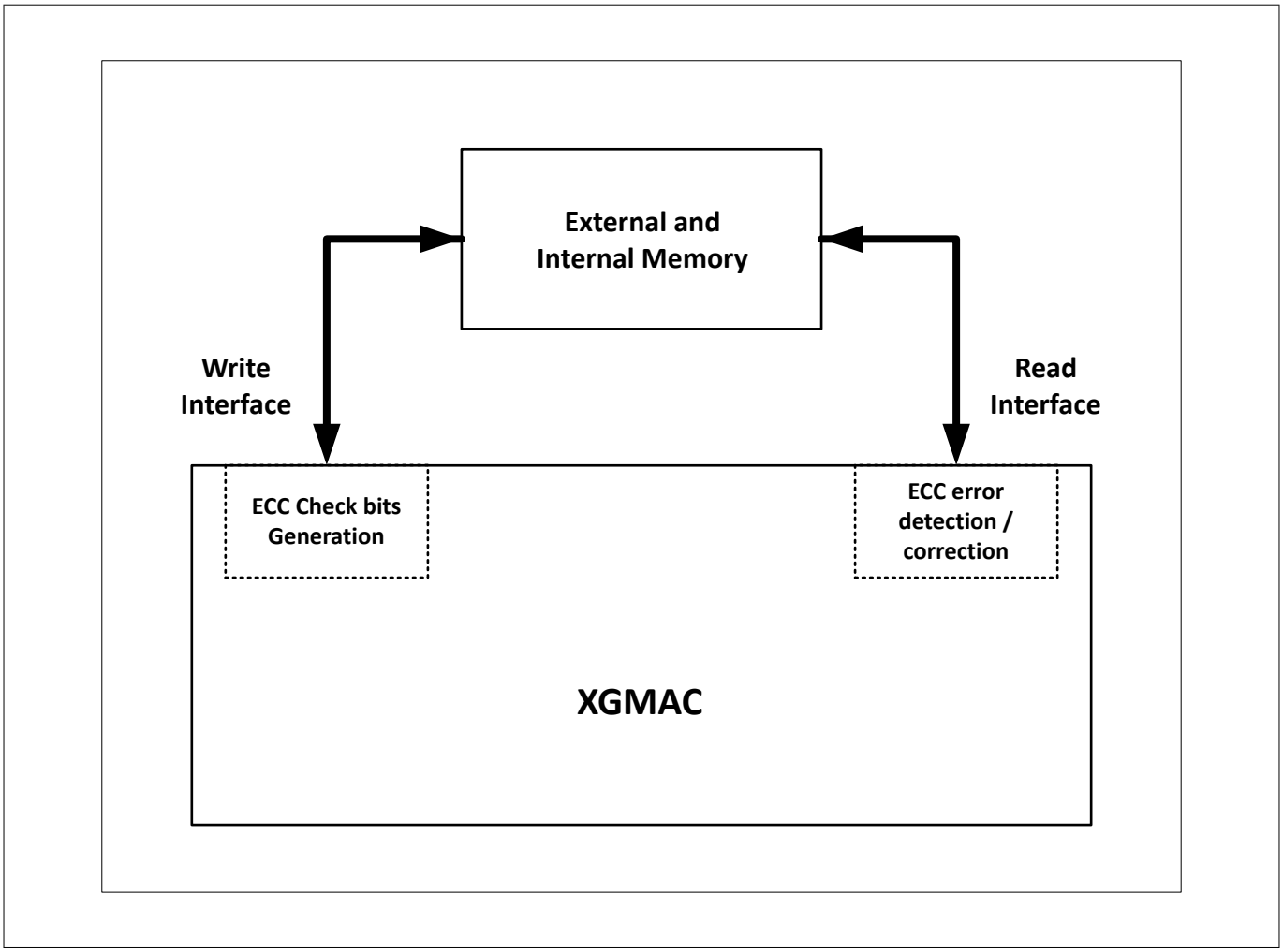


Figure 170 Block diagram of ECC protection for memories

The number of ECC check bits generated for each data block depends on the size of the data bus and address bus.

ECC generator is added to all the memory interfaces in XGMAC. These are:

- Two-port RAM/single-port RAM memories used for the TX Queues and RX Queues in the MTL
- Single-port RAM memory used for the Gate Control List in the 802.1Qbv EST function
- Register array without ECC used for the Instruction Table of rules for the Flexible RX Parser
- Single-port RAM memory used for the descriptor cache

The table below describes the memory bus widths, ECC code size and the generated check bits size.

Table 744 Memory bus width, ECC code size, and checkbits size

Memory block	Configured data width	Additional control bits	Total data width (without ECC)	Address width (Depends on memory size)	ECC code reference range	Required check bits	Total data width (with ECC)
MTL Tx Memory (2 memories)	64	4	68 bits	12	Total Data width + Address width <= 120bits	8 bits	76 bits
MTL Rx Memory (2 memories)	64	4	68 bits	12	Total Data width + Address width <= 120bits	8 bits	76 bits
EST / GCL Memory	32	0	32 bits	8	Total Data width + Address width <= 26 bits Total Data width + Address width <= 57 bits	7 bits	39 bits
Descriptor cache memory	64	0	64 bits	9	Data width + Address width <= 120 bits	8 bits	72 bits

Handling uncorrectable errors

Recovery from an uncorrectable error is based on the memory block and the operating mode of XGMAC.

- MTL Tx memory:

  If an uncorrectable error is detected, the packet is terminated with a forcible EOF. A CRC error (`mti_ecc_err`) is introduced by MAC for the transmitted packet, and the remaining packets in the FIFO are flushed and flush status (bit 13) is indicated to the application.

  When an uncorrectable error is detected, the application should reset XGMAC IP to ensure proper functionality.

  ***Note**: The MTL FIFO flush is issued for XGMAC to recover with limited support.*

  XGMAC generates an interrupt to indicate to the application that a correctable, uncorrectable or address mismatch error has occurred and the status is indicated in `MTL_ECC_Interrupt_Status` register
- MTL Rx memory:

  When an uncorrectable error is detected, reset XGMAC IP to ensure proper functionality.

XGMAC generates an interrupt to indicate to the application that correctable, uncorrectable or address mismatch errors are detected and the status is indicated in MTL_ECC_Interrupt_Status register

***Note**: Issue soft reset*

- MTL EST memory:

  When an uncorrectable error is detected in the MTL EST memory, XGMAC follows these sequence of actions for recovery:

  1. XGMAC disables the EST feature by resetting the EEST bit of MTL_EST_Control register. It sets all the gates to open state, generates an interrupt, and sets the status in the MTL_ECC_Interrupt_status register. This allows the software to reprogram the correct value in the EST memory
  2. The location of the uncorrected error in the EST memory is available in the EUEAS/ECEAS field of the MTL_ECC_Err_Addr_Status register

  XGMAC generates an interrupt to indicate to the application if any correctable, uncorrectable and address mismatch errors are detected and the appropriate status is indicated in MTL_ECC_Interrupt_Status register.

  Software must reprogram and restart EST after an uncorrectable error is reported in the EST memory. Issuing a reset is not mandatory in this case, but other factors might cause a need to reset
- Internal Register Based Instruction Table of MTL Rx Parser Memory:

  When an uncorrectable error is detected in the Instruction Table (IT) of MTL Rx Parser, the parser halts the parsing of all subsequent packets by disabling the parser. After the software completes the necessary steps against the detected ECC error, it can re-enable the FRPE field of the MTL_Operation_Mode register, to continue parsing.

  XGMAC generates an interrupt to indicate to the application that correctable, uncorrectable, or address mismatch errors are detected and the status is indicated in the MTL_ECC_Interrupt_Status register

  ***Note**: Same error could be detected and reported by the periodic flexible receive parser memory checks performed by XGMAC. To avoid recursive reporting of injected ECC UE error, perform a valid write to the corrupted memory location*

Diagnostic Support for the Error Management

Following statistics are provided for monitoring the error behavior on each of the memory blocks.

- Error status provided to management
  - A separate status for correctable, uncorrectable, and address mismatch is specified in the MTL_ECC_Interrupt_Status register
  - Memory locations at which correctable and uncorrectable errors are detected is specified in the MTL_ECC_Err_Addr_Status register. In addition, a control bit (MEEAO field of MTL_ECC_Control register) decides, whether the first errored memory address is reported or the latest errored memory address is reported
  - MTL_ECC_Err_Cntr_Status register has separate counters to count the number of correctable and uncorrectable errors
- Interrupts provided to management
  - Separate interrupts are generated for correctable and unrecoverable errors
  - The sbd_sfty_ue_intr_o interrupt is generated when unrecoverable errors are detected and these errors cannot be masked

- The sbd_sfty_ce_intr_o interrupt is generated when correctable errors are detected. XGMAC sets the respective interrupt enable bits (in the DMA/MTL_ECC_Interrupt_enable register)
- To find the root cause of the error, read the DMA/MTL_Safety_Interrupt_Status and DMA/MTL_ECC_Interrupt_Status registers
- To clear the interrupt, write 1 in the respective interrupt status bit in DMA/MTL_ECC_Interrupt_Status registers

For more details, see the section "Registers".

***Note**:*
- *The status/counters/interrupts are generated per memory*
- *MTL Tx and Rx memory is logically divided into multiple queues. But, ECC diagnostics (status/counter/interrupts) are common to all queues*

Related information

Registers on page 802

ECC Error Injection Capabilities

XGMAC supports error injection capabilities for each memory as a static configuration. You can program the position where the errors are injected in the data word/address/ECC bits. This depends on the EIM/BLEI field of the MTL_DPP_ECC_EIC and 3 control bits of the MTL_Debug_Control/MTL_EST_CGL_Control registers. For each memory, four control bits and eight error locations bits are provided to inject errors:

- One bit to enable error injection (EIEE/ESTEIEE field of the MTL_Debug_Control/MTL_EST_GCL_Control register).
- One bit to enable address error injection (EIAEE/ESTEIAEE bit in MTL_Debug_Control/MTL_EST_GCL_Control)
- One bit to enable data or ECC error injection (EIM field of the MTL_DPP_ECC_EIC register).
- Eight bits to indicate the bit location of error injection (BLEI field of MTL_DPP_ECC_EIC register).
- One bit to indicate the type of error to be injected (EIEC/ESTEIEC field of MTL_Debug_Control/MTL_EST_GCL_Control register).

  Position of the error bit should be programmed ahead of error injection.
- One bit to indicate the type of error to be injected
  - 0: 1 bit error
  - 1: 2 bit error

ECC address error injection supports only 1-bit error injection. DUT does not detect multiple bit address error. However, in the ECC 2-bit error injection mode,

- 1-bit error is always inserted at the 0^{th} bit.
- Other 1-bit error is inserted at the location indicated by the BLEI field of MTL_DPP_ECC_EIC register. If BLEI is configured to 0, XGMAC inserts only 1-bit error, at the 0^{th} bit.

While using ECC error injection in the debug mode,

- There should be no traffic in XGMAC
- When multiple CSR writes are required for writing single data word into the memory, the application should ensure that all the CSR writes corresponding to one memory write maintains the same value for the error injection control word.

The control bit descriptions for

- MTL Tx/Rx descriptor cache memories are specified in the MTL_Debug_Control register. MTL_DPP_ECC_EIC register should be programmed before MTL/DMA_DPP_Control register.

- MTL EST memory are specified in the `MTL_EST_GCL_Control` register
- Rx parser are specified in the `MTL_RXP_Indirect_Acc_Control_Status` register

14.4.3.22.2 Finite State Machine (FSM) parity and timeout protection

Overview

The FSM protection feature supports FSM parity and time out protection.

FSM State Parity Protection

XGMAC implements one-hot encoding scheme for holding the state register. Odd parity is implemented on all the FSM state register bits. Parity is monitored for every clock, after the reset is de-asserted. When a bit flips due to transient errors or permanent faults, the erroneous FSM is set to its default state and an unrecoverable error is indicated.

When the FSM state parity protection is enabled, by setting PRTYEN field of MAC_FSM_Control Register, the FSM state error is indicated by setting the FSMPES field of the MAC_DPP_FSM_Interrupt_Status register. Also, the Safety Interrupt (sbd_sfty_ue_intr_o) is asserted when an FSM Error is detected.

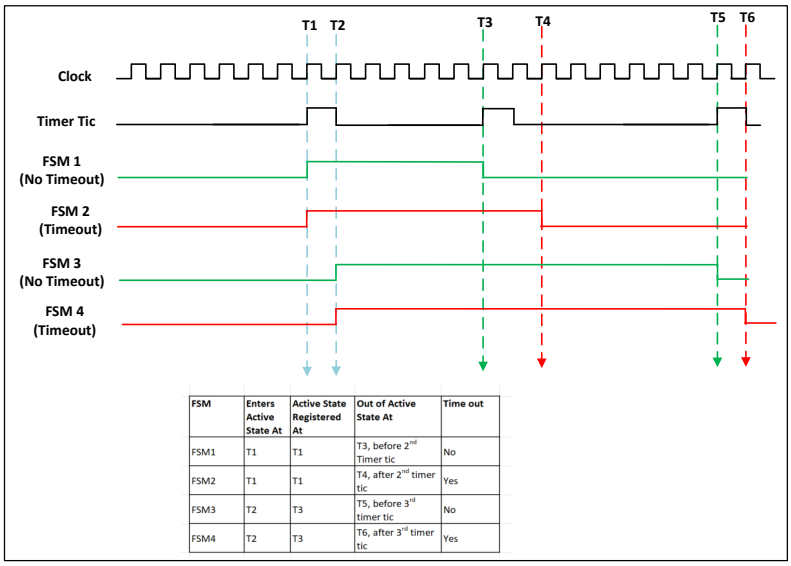
Error Injection mode is also supported for FSM parity error check. Program the Error Injection enable for the respective clock domain denoted by [23:16] bits of MAC_FSM_Control Register.

***Note**: The FSM Parity error is classified as non-recoverable error and software intervention (IP reset) is required to recover from the error state.*

FSM Timeout Protection

The FSM Timeout feature provides a mechanism to ensure that all FSMs in XGMAC can complete a transaction and return to a known completion state (IDLE or any other state that indicates completion of a transaction/ transfer) within the programmed timeout value.

Program the TMR field of MAC_FSM_ACT_Timer register with a value that indicates the number of CSR cycles required to generate a 1 microsecond tic. This microsecond tic is internally used to generate the programmed timeout duration. Two timeout tics (normal mode timeout and large mode timeouts) are generated to provide flexibility to choose one per clock domain, by using the bits [31:24] of MAC_FSM_Control Register. Supported values for Timeout duration are 1us, 1.024ms, 16.384ms, 65.5536ms, 262.144ms, 1048.576ms (~1sec), 4194.304ms (~4s), 8.388s, 16.777s, and 33.554s which is based on the programmable bits NTMRMD, LTMRMD fields of MAC_FSM_ACT_Timer register. Interface timeouts are based on the normal mode tic generation; only the FSM timeouts depend on either normal or large tick selection.



FSM	Enters Active State At	Active State Registered At	Out of Active State At	Time out
FSM1	T1	T1	T3, before 2nd Timer tic	No
FSM2	T1	T1	T4, after 2nd timer tic	Yes
FSM3	T2	T3	T5, before 3rd timer tic	No
FSM4	T2	T3	T6, after 3rd timer tic	Yes

Figure 171 FSM Timeout Timing Diagram

In Figure 171, the timer ticks are generated based on the programmable timeout value. The timer ticks are synchronized and made available in all the clock domains that have the FSMs.

At every timer tick, all the FSMs are monitored for active state (non-IDLE state, which indicates the FSM is actively processing transactions or hand-shakes) and an FSM active flag bit is set. The active flag bit is implemented for every FSM that is monitored for timeout. When the FSM reaches an IDLE or transaction completion state, the active flag bit of that FSM is reset. At the subsequent timer tick, all the FSMs' active flag bits are monitored and any flag that is set indicates a timeout for that FSM. This process of setting the flag and checking at subsequent timer tick is repeated at every tick.

Timeout Error Injection per clock domain is enabled by programming [15:8] fields of MAC_FSM_Control register. When Timeout error injection enable is set for a clock domain, the FSMs in that clock domain automatically timeout and generate an interrupt even without traffic. As error injection is a debug mode, TMR value need not indicate 1us, but can be programmed to a smaller value at which debug mode ticks are needed. FSM Timeout and Parity Error Injection modes can be used for testing at key-on/key-off.

The FSM timeout status is specified in the [15:8] field of MAC_DPP_FSM_Interrupt_Status register as per the respective clock domains. One FSM timeout status bit is made available per clock domain. The safety interrupt (sbd_sfty_ue_intr_o) is asserted when the timeout error is set for any of the clock domains. XGMAC does not attempt to recover from a FSM timeout condition and relies on the application to take the corrective action (resetting XGMAC).

***Note**:*

- *Based on the time at which the FSM enters the active state relative to the timer ticks the timeout period could be any value between the programmed timeout and 2x programmed timeout.*
- *FSM related to PPS (when configured) will not be protected by timeouts as the delays involved in this FSM state transitions can be huge.*

Recovering from FSM Parity/Timeout Errors

If a FSM parity/timeout error is reported, it is recommended to soft-reset the IP. This is because, data path FSM parity/timeout errors are non-recoverable and the functionality of the IP is not guaranteed after a safety non-recoverable interrupt is reported.

14.4.3.22.3 Application/CSR interface timeout protection

This chapter contains the description of the timeout protection.

Overview

This feature provides timeout protection to the application/CSR Interface.

Description

All the interfaces which has the handshake mechanism in XGMAC are monitored for potential hangs due to the external agent (Master/Slave/Interconnect/Application client) not responding to the requests/transfers initiated by XGMAC. After the request is initiated the response arrival interval is monitored. If the response does not arrive within a programmed time (TMR field of `MAC_FSM_ACT_Timer`) the timeout is triggered, using similar trigger generation as explained in the Section FSM protection.

The hardware does not attempt to recover from Application/CSR Interface hangs and relies on software to take appropriate corrective action.

The Application Interface timeout status is provided in the [23:16] field of `MAC_DPP_FSM_Interrupt_Status` register. The Safety interrupt (`sbd_sfty_ue_intr_o`) is asserted when application timeout status is set.

ATI and ARI interface

The timeout is triggered under the following scenarios:

- XGMAC requests for the Transmit packet status transfer and the DMA does not respond within the programmed time
- XGMAC requests for a data/status of receive packet and the DMA does not respond within the programmed time

The Timeout status for the ATI and ARI interface is set in the ATITES and ARITES fields of `MAC_DPP_FSM_Interrupt_Status` register.

AXI Master Interface

The timeout is triggered under the following scenarios:

- The XGMAC initiates Write or Read ADDRESS transfer (by asserting AWVALID or ARVALID) and corresponding ready (AWREADY or ARREADY) is not asserted within the programmed time.
- The XGMAC initiates Write DATA transfer (by asserting WVALID) and corresponding ready (WREADY) is not asserted within a programmed time.

The Timeout status for the AXI master interface is set in the MSTTES field of `MAC_DPP_FSM_Interrupt_Status` register.

AXI Slave Interface

The timeout is triggered under the following scenarios:

- The XGMAC responds to a Read request (by asserting RVALID) and external Master does not accept the response data, by asserting (RREADY) within a programmed time.
- The XGMAC responds to a Write request (by asserting BVALID) and external Master does not accept the response by asserting (BREADY) within a programmed time.

This feature is available in AXI-SLAVE configuration

The Timeout status for the AXI slave interface is set in "SLVTES" field of `MAC_DPP_FSM_Interrupt_Status` register.

***Note**:*
- *Protection is not needed for other Slave interfaces (MCI/APB/AHB) because potential hangs can be only when XGMAC does not respond to the requests. In such cases the XGMAC internal protection logic (FSM timeout) detects such defects.*
- *Based on the time at which the transfer is initiated relative to the timer ticks, the timeout period could be any value between the programmed timeout and 2x times the programmed timeout.*
- *When an unrecoverable safety interrupt is issued and the read of the Interrupt status register returns all zeros, it implies that the CSR read is not functional. As soft reset of XGMAC is not possible without the CSR access, and therefore, a hard reset of XGMAC is recommended.*

14.4.3.23 Descriptors

14.4.3.23.1 Overview

The DMA in the Ethernet subsystem transfers data based on a linked list of descriptors. The application creates the descriptors in the system memory. The XGMAC supports the following two types of descriptors:

- Normal Descriptor: Normal descriptors are used for exchanging packet data and packet control or status information applicable between the application and the XGMAC controller.
- Context Descriptor: Context descriptors are used to provide additional control information applicable to the packets to be transmitted or additional status information for received packets.
  - Tx side: The application prepares the context descriptors and puts the context descriptors in the linked list before the normal descriptor. The XGMAC-AXI interface processes the controls given in the context descriptor for the packet(s) following the context descriptor.
  - Rx side: The application does not create context descriptors. It creates a linked-list of normal descriptors. When the Rx DMA has additional status information for a received packet, it closes the next descriptor with this additional status and marks it as “context descriptor”.

Each normal descriptor contains two address pointers to two buffers.

***Note**:* *A single packet can use any number of descriptors upto the ring length.*
The reserved fields in the descriptor must be set to 0, otherwise the behavior is unpredictable.

14.4.3.23.2 Descriptor Structure

The XGMAC supports the ring structure for DMA descriptor as shown in Figure 172.

***Note**:* *A descriptor ring must provide a minimum of four descriptors or more to the XGMAC.*

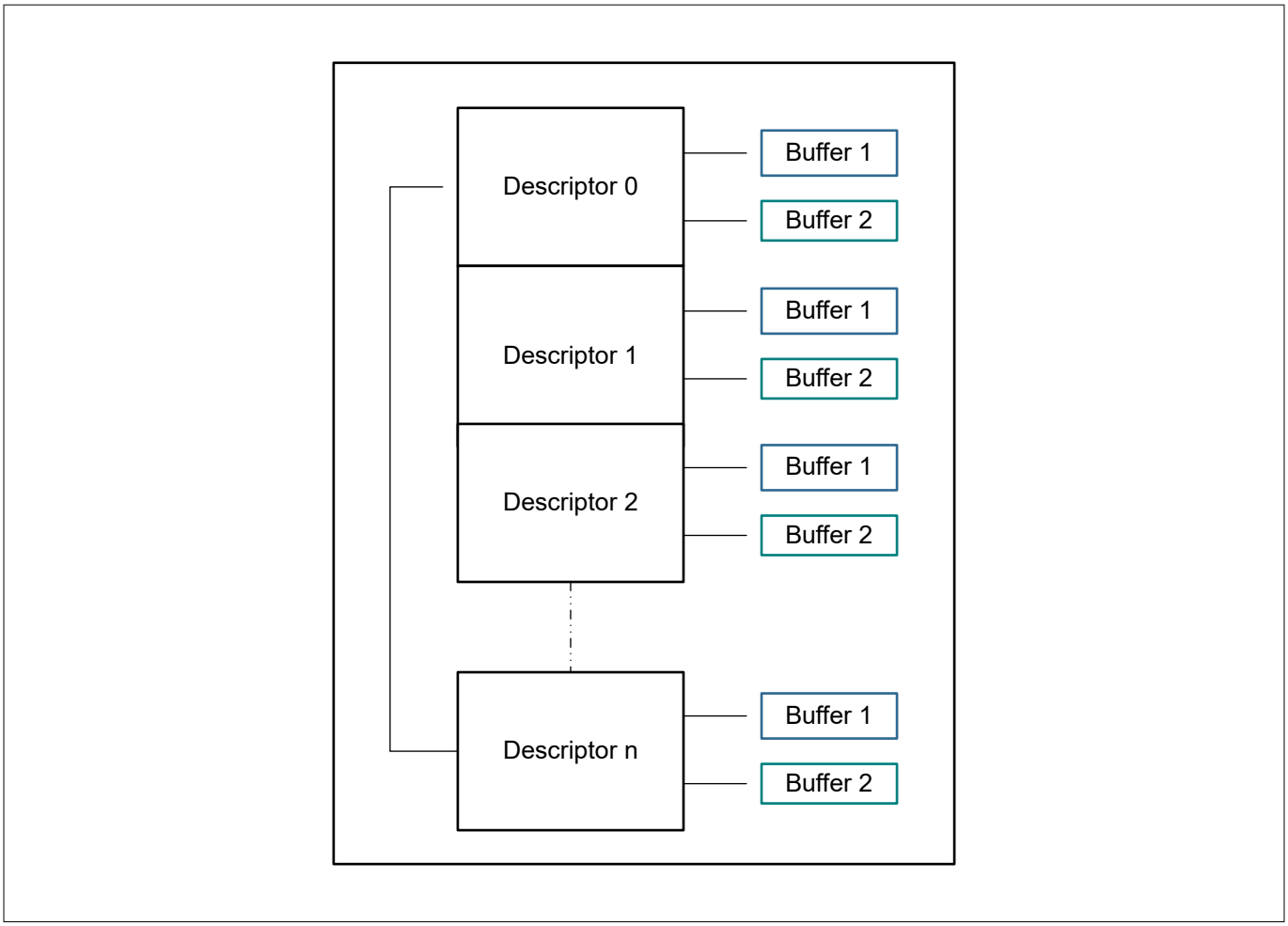


Figure 172 Descriptor Ring Structure

In the ring structure, descriptors are separated by number of 64-bit (DWord) programmed in the DSL field of the DMA_CH(#i)_Control register. The application needs to program the total ring length, that is, the total number of descriptors in ring span in the following registers of a DMA channel:

- Transmit Descriptor Ring Length Register (DMA_CH(#i)_Tx_Control2)
- Receive Descriptor Ring Length Register (DMA_CH(#i)_Rx_Control2))

The Descriptor Tail Pointer Register contains the pointer to the descriptor address (i). The descriptors up to one location less than the one indicated by the descriptor tail pointer (i – 1) are owned by the DMA. The DMA continues to process the descriptors until the following condition occurs:

```
Current Descriptor Pointer == Descriptor Tail Pointer
```

The DMA goes into the Suspend mode when this condition occurs. The application must perform a write to the Descriptor Tail pointer register and increase the offset so that the following condition is true:

```
Current Descriptor < Pointer Descriptor Tail Pointer
```

The DMA automatically wraps around the base address when the last descriptor in the ring is processed, as shown in Figure 173.

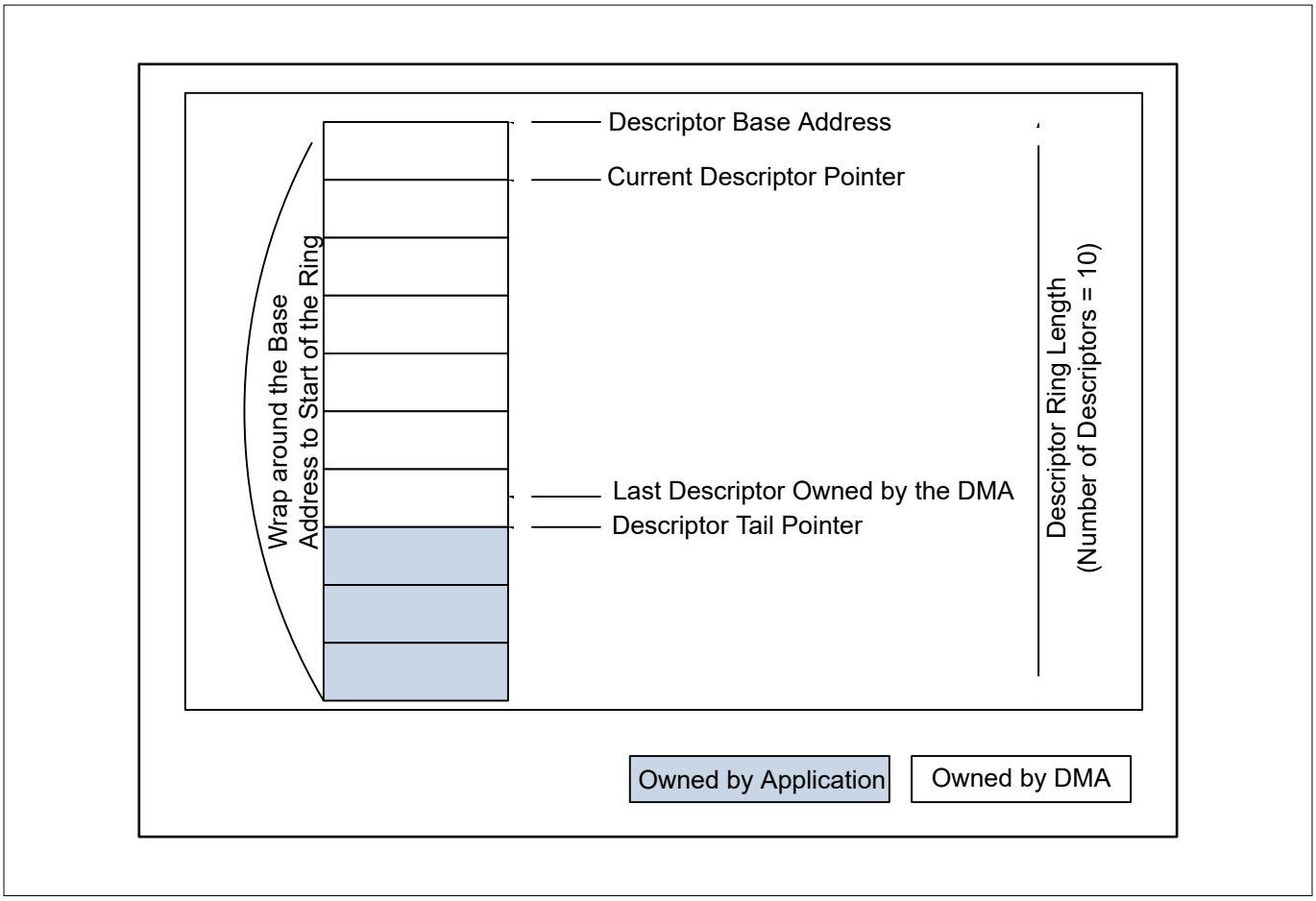


Figure 173 DMA Descriptor Ring

For descriptors owned by the application, the OWN bit of DES3 is reset to 0. For descriptors owned by the DMA, the OWN bit is set to 1. If the application has only two descriptors in the beginning, the application sets the last descriptor address (tail pointer) to Descriptor Base Address + 1. The DMA processes the first descriptor and then waits for the application to advance change the tail pointer back to Descriptor Base Address.

***Note**: The descriptor list address (DMA_CHi_TxDesc_List_LAddress and DMA_CHi_RxDesc_List_LAddress) and descriptor ring lengths (DMA_CHi_Tx_Control2.TDRL and DMA_CHi_Rx_Control2.RDRL) should be programmed such that the descriptor address does not cross a 4 GB address boundary.*

14.4.3.23.3 Transmit Descriptor

The DMA in XGMAC requires at least one descriptor for a transmit packet. In addition to two buffer length fields, and two address pointers, the transmit descriptor has control fields which can be used to control the packet processing on per-transmit packet basis. The Transmit Normal descriptor has the following two formats: Read format and Write-Back format

Transmit Normal Descriptor (Read Format)

Figure 174 shows the Read Format for a Transmit normal descriptor. This is the format in which the software can prepare the descriptor contents and hand it over to the DMA Engines of the XGMAC.

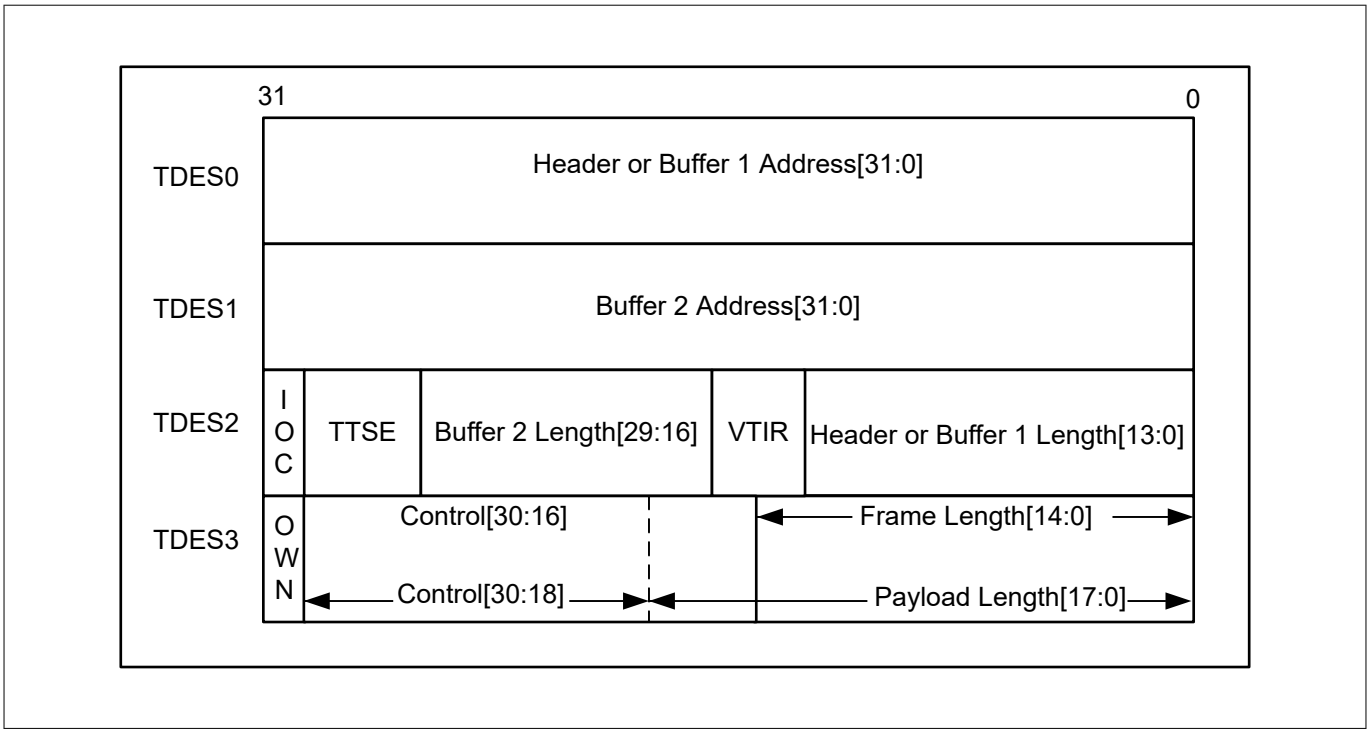


Figure 174 Transmit Descriptor Read Format

TDES0 Normal Descriptor (Read Format)

Table 745 TDES0 Normal Descriptor (Read Format) (Continued)

Bit	Name	Description
31:0	BUF1AP	Buffer 1 Address Pointer These bits indicate the physical address of Buffer 1.

TDES1 Normal Descriptor (Read Format)

Table 746 TDES1 Normal Descriptor (Read Format)

Bit	Name	Description
31:0	BUF2AP	Buffer 2 Address Pointer These bits indicate the physical address of Buffer 2 when a descriptor ring structure is used (as shown in Figure 172). There is no limitation for the buffer address alignment.

TDES2 Normal Descriptor (Read Format)

31	30	29:16	15:14	13:0
IOC	TTSE	B2L	VTIR	B1L

Table 747 TDES2 Normal Descriptor (Read Format)

Bits	Name	Description
31	IOC	Interrupt on Completion This bit sets the TI bit in the DMA_CH(#i)_Status register after the present packet has been transmitted.

(table continues...)

Table 747 (continued) TDES2 Normal Descriptor (Read Format)

Bits	Name	Description
30	TTSE	Transmit Timestamp Enable This bit enables the IEEE1588 time stamping for Transmit packet referenced by the descriptor.
29:16	B2L	Buffer 2 Length The driver sets this field. When set, this field indicates Buffer 2 length.
15:14	VTIR	VLAN Tag Insertion or Replacement These bits request the MAC to perform VLAN tagging or untagging before transmitting the packets. If the packet is modified for VLAN tags, the MAC automatically recalculates and replaces the CRC bytes. The following list describes the values of these bits: • 2'b00: Do not add a VLAN tag. • 2'b01: Remove the VLAN tag from the packets before transmission. This option should be used only with the VLAN packets. • 2'b10: Insert a VLAN tag with the tag value programmed in the MAC_VLAN_Incl register or context descriptor. • 2'b11: Replace the VLAN tag in packets with the tag value programmed in the MAC_VLAN_Incl register or context descriptor. This option should be used only with the VLAN packets.
13:0	B1L	Buffer 1 Length

TDES3 Normal Descriptor (Read Format)

31	30	29	28	27:26	25:23	22:19	18	17:16	15	14:0
OWN	CTXT	FD	LD	CPC	SAIC	SLOTNUM	Rsvd	CIC	Rsvd	FL

Table 748 TDES3 Normal Descriptor (Read Format)

Bits	Name	Description
31	OWN	Own Bit When this bit is set, it indicates that the DMA owns the descriptor. When this bit is reset, it indicates that the application owns the descriptor. The DMA clears this bit after it completes the transfer of data given in the associated buffers.
30	CTXT	Context Type This bit should be set to 1'b0 for normal descriptor.
29	FD	First Descriptor When this bit is set, it indicates that the buffer contains the first segment of a packet.
28	LD	Last Descriptor When this bit is set, it indicates that the buffer contains the last segment of the packet. When this bit is set, the B1L or B2L field must have a non-zero value, else behavior is undefined.

(table continues...)

Table 748 (continued) TDES3 Normal Descriptor (Read Format)

Bits	Name	Description
27:26	CPC	CRC Pad Control This field controls the CRC and Pad Insertion for Tx packet. This field is valid only when the first descriptor bit (TDES3[29]) is set. The following list describes the values of Bits[27:26]: • 2'b00: CRC and Pad Insertion The MAC appends the cyclic redundancy check (CRC) at the end of the transmitted packet of length greater than or equal to 60 bytes. The MAC automatically adds padding and CRC to a frame shorter than 60 bytes. • 2'b01: CRC Insertion (Disable Pad Insertion) The MAC appends the CRC at the end of the transmitted packet but it does not append padding. The application should ensure that the padding bytes are present in the packet being transferred from the Transmit Buffer, that is, the packet being transferred from the Transmit Buffer is of length greater than or equal to 60 bytes. • 2'b10: Disable CRC Insertion The MAC does not append the CRC at the end of the transmitted packet. The application should ensure that the padding and CRC bytes are present in the packet being transferred from the Transmit Buffer. • 2'b11: CRC Replacement The MAC replaces the last four bytes of the transmitted packet with recalculated CRC bytes. The application should ensure that the padding and CRC bytes are present in the packet being transferred from the Transmit Buffer.
25:23	SAIC	SA Insertion Control These bits request the MAC to add or replace the Source Address field in the Ethernet packet with the value given in the MAC Address 0 register. If the Source Address field is modified in a packet, the MAC automatically recalculates and replaces the CRC bytes. Bit 25 specifies the MAC Address Register (1 or 0) value that is used for Source Address insertion or replacement. The following list describes the values of Bits[24:23]: • 2'b00: Do not include the source address • 2'b01: Include or insert the source address. For reliable transmission, the application must provide frames without source addresses. • 2'b10: Replace the source address. For reliable transmission, the application must provide frames with source addresses. • 2'b11: Reserved These bits are valid when the First Segment control bit (TDES3 [29]) is set.
22:19	SLOTNUM	SLOTNUM: Slot Number Control Bits in AV Mode These bits indicate the slot interval in which the data should be fetched from the corresponding buffers addressed by TDES0 or TDES1. When the Transmit descriptor is fetched, the DMA compares the slot number value in this field with the slot interval maintained in the RSN field DMA_CH(#i)_Slot_Function_Control_Status. It fetches the data from the buffers only if a value matches. These bits are valid only for the AV channels.
18	Rsvd	Reserved

(table continues...)

Table 748 (continued) TDES3 Normal Descriptor (Read Format)

Bits	Name	Description
17:16	CIC	Checksum Insertion Control or TCP Payload Length These bits control the checksum calculation and insertion. The following list describes the bit encoding: • 2'b00: Checksum Insertion Disabled. • 2'b01: Only IP header checksum calculation and insertion are enabled • 2'b10: IP header checksum and payload checksum calculation and insertion are enabled, but pseudo-header checksum is not calculated in hardware • 2'b11: IP Header checksum and payload checksum calculation and insertion are enabled, and pseudo-header checksum is calculated in hardware When Timestamping is enabled without One Step Timestamping Feature and OSTC bit is set in the context Descriptor, the content of this field is not effective and Checksum is updated as though CIC field is set to 2'b11.
15	Rsvd	Reserved
14:0	FL	Packet Length This field is equal to the length of the packet to be transmitted in bytes. This field is equal to the total length of the packet to be transmitted: `Ethernet Header Length + TCP /IP Header Length – Preamble Length – SFD Length + Ethernet Payload Length`

Transmit Normal Descriptor (Write-Band Format)

The XGMAC processes the descriptor, acts on it and then writes back the status content. This operation indicates to the software that the descriptor is free and no longer required by the hardware. The XGMAC writes only the bits[31:24] of TDES3 during this operation and all the other fields of the descriptor in the host memory are not modified.

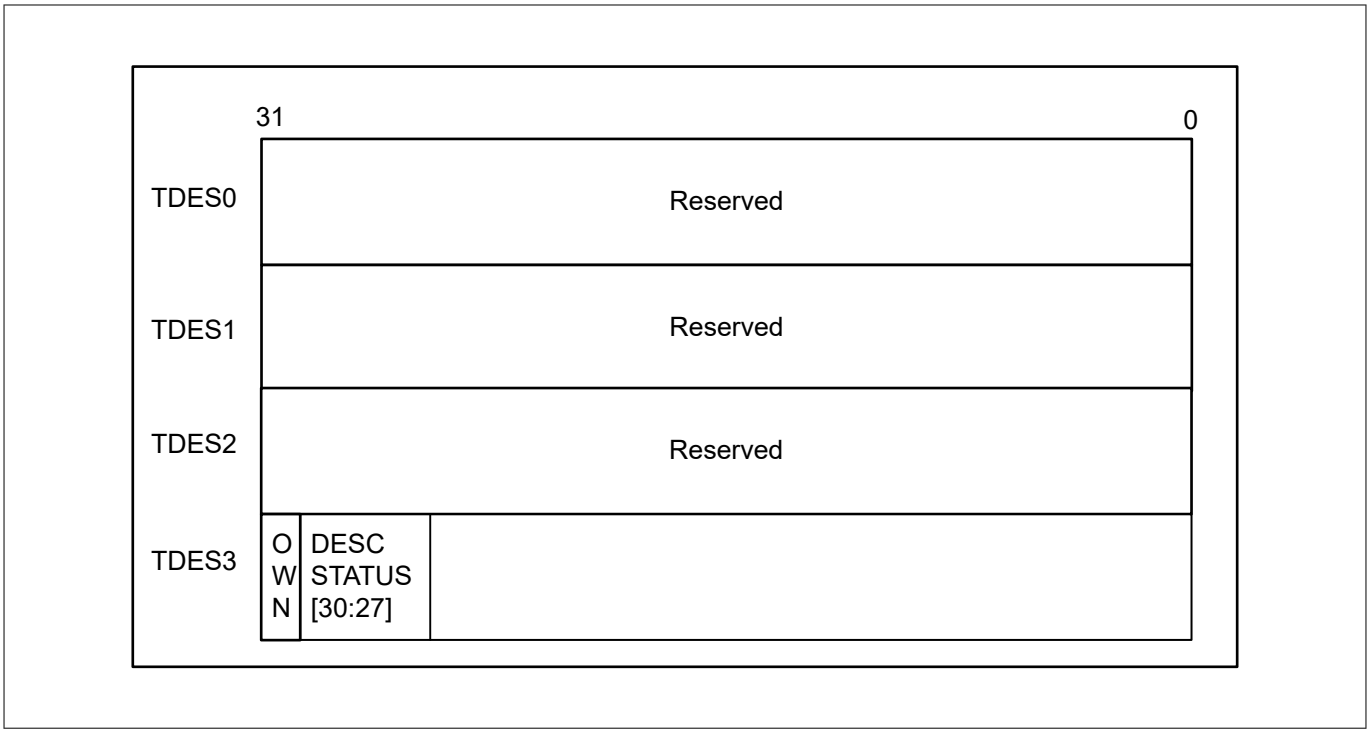


Figure 175 Transmit Descriptor Write-Back Format

TDES3 Normal Descriptor (Write-Back Format)

31	30	29	28	27	26:0
OWN	CTXT	FD	LD	DERR	Rsvd

Table 749 TDES3 Normal Descriptor (Write-Back Format)

Bit	Name	Description
31	OWN	Own Bit When this bit is set, it indicates that the XGMAC DMA owns the descriptor. The DMA clears this bit when it writes back the descriptor. The DMA closes the descriptor when either of the following conditions are true: • The DMA completes the transfer of data indicated by the buffers from the host memory to the corresponding Tx Queue • The B1L and B2L values are zero, indicating empty buffers
30	CTXT	Context Type This bit is set to 'b0 for Normal descriptor.
29	FD	First Descriptor This bit indicates that the buffer contains the first segment of a packet.
28	LD	Last Descriptor This bit is set 'b1 for last descriptor of a packet. The DMA writes the status fields only in the last descriptor of the packet.
27	DERR	Descriptor Error When this bit is set, it indicates that the descriptor content is incorrect. The DMA sets this bit during write-back while closing the descriptor. The descriptor errors can be: • AXI Bus Error: This bit is asserted when AXI Bus Error response is received while processing the descriptor. This can be either when the descriptor itself is fetched or when the packet data indicated by its buffer pointers are fetched. • Descriptor Definition Error: 1 This bit is asserted if CTXT and FD or LD bits are set to 1 (CTXT & (FD | LD)). • All 1s
26:0	Rsvd	Reserved

Transmit Context Descriptor

The Transmit Context descriptor can be provided any time before a packet descriptor. The context is valid for the current packet and subsequent packets. The context descriptor is used to provide the timestamps for one-step timestamp correction, VLAN Tag ID for VLAN insertion feature, and SA insertion bit for SA insertion. Write back is done on a context descriptor only to reset the OWN bit and to set the CDE bit if necessary.

***Note**: The VLAN Tag IDs provided by the application in a context descriptor with their corresponding Valid bits set, are stored internally by the DMA.*

When the outer or inner VLAN tag is provided with the Valid bit set, the DMA always passes the last valid VLAN tag to the MTL. The application cannot invalidate the valid VLAN tag stored by the DMA. The VLAN tag is inserted or replaced based on the control inputs provided for the packet.

The Inner VLAN Tag Control input is used only for the next packet that immediately follows the context descriptor. The application must provide a context descriptor before the normal descriptor of each packet for which the DMA should use the inner VLAN Tag control input.

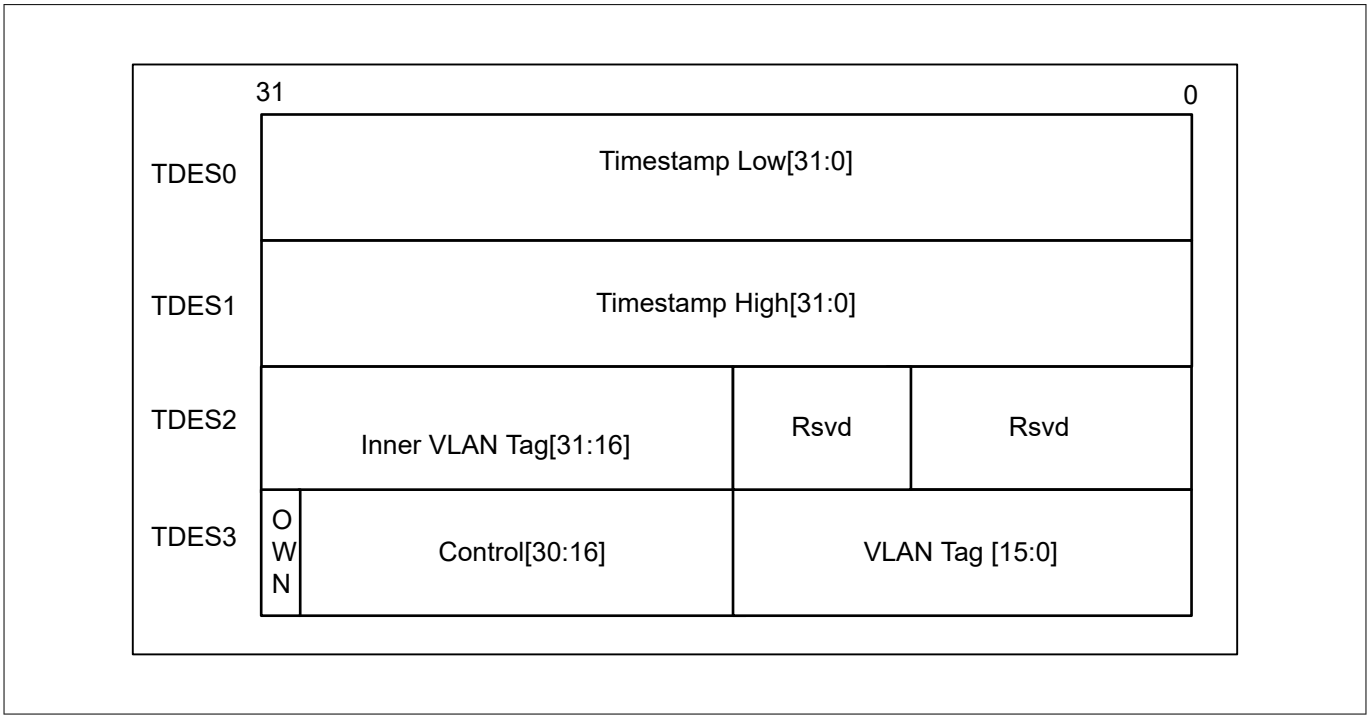


Figure 176 Transmit Context Descriptor Format

TDES0 Context Descriptor

Table 750 TDES0 Context Descriptor

Bit	Name	Description
31:0	TTSL	Transmit Packet Timestamp Low or Packet ID For one-step correction, the driver can provide the lower 32 bits of timestamp in this descriptor word. The DMA uses this value as the low word for doing one-step timestamp correction. This field is valid only if the OSTC and TCMSSV bits of TDES3 context descriptor are set. When the OSTC bit is reset and PIDV is set, the lower 10-bits [9:0] indicate the packet ID which is used for 2-step time stamping.

TDES1 Context Descriptor

Table 751 TDES1 Context Descriptor

Bit	Name	Description
31:0	TTSH	Transmit Packet Timestamp High or Packet ID For one-step correction, the driver can provide the upper 32 bits of timestamp in this descriptor. The DMA uses this value as the high word for doing one-step timestamp correction. This field is valid only if the OSTC and TCMSSV bits of TDES3 context descriptor are set.

TDES2 Context Descriptor

Table 752 TDES2 Context Descriptor

Bit	Name	Description
31:16	IVT	Inner VLAN Tag When the IVLTV bit of TDES3 context descriptor is set and the TCMSSV and OSTC bits of TDES3 context descriptor are reset, TDES2[31:16] contains the inner VLAN Tag to be inserted in the subsequent Transmit packets.
15:14	Rsvd	Reserved
13:0	Rsvd	Reserved

TDES3 Context Descriptor (Read Format)

31	30	29	28	27	26	25	24	23:20	19:18	17	16	15:0
OWN	CTXT	CDE	Rsvd	OSTC	TCMSSV	PIDV	Rsvd	Rsvd	IVTIR	IVLTV	VLTV	VT

Table 753 TDES3 Context Descriptor (Read format)

Bit	Name	Description
31	OWN	Own Bit When this bit is set, it indicates that the XGMAC DMA owns the descriptor. When this bit is reset, it indicates that the application owns the descriptor. The DMA clears this bit after it accepts the context descriptor contents.
30	CTXT	Context Type This bit is set to 1’b1 for Context descriptor.
29:28	Rsvd	Reserved
27	OSTC	One-Step Timestamp Correction Enable When this bit is set, the DMA performs a one-step timestamp correction with reference to the timestamp values provided in TDES0 and TDES1.
26	TCMSSV	One-Step Timestamp Correction When this bit and the OSTC bit are set, it indicates that the Timestamp Correction input provided in TDES0 and TDES1 is valid. When the OSTC bit is reset and this bit and the TSE bit of TDES3 are set in subsequent normal descriptor, it indicates that the MSS input in TDES2 is valid.

(table continues...)

Table 753 (continued) TDES3 Context Descriptor (Read format)

Bit	Name	Description
25	PIDV	Packet ID valid When this bit is set and OSTC bit is reset, the bits [9:0] of TDES0 Context Descriptor indicates the Packet ID for 2-step Time stamping.
24	Rsvd	Reserved
23:20	Rsvd	Reserved
19:18	IVTIR	Inner VLAN Tag Insert or Replace When this bit is set, these bits request the MAC to perform Inner VLAN tagging or untagging before transmitting the packets. If the packet is modified for VLAN tags, the MAC automatically recalculates and replaces the CRC bytes. The following list describes the values of these bits: • 2'b00: Do not add the inner VLAN tag. • 2'b01: Remove the inner VLAN tag from the packets before transmission. This option should be used only with the VLAN frames. • 2'b10: Insert an inner VLAN tag with the tag value programmed in the register or context descriptor. • 2'b11: Replace the inner VLAN tag in packets with the tag value programmed in the register or context descriptor. This option should be used only with the VLAN frames. These bits are valid when the Enable SA and VLAN Insertion on Tx and Enable Double VLAN Processing options are selected.
17	IVLTV	Inner VLAN Tag Valid When this bit is set, it indicates that the IVT field of TDES2 is valid.
16	VLTV	VLAN Tag Valid When this bit is set, it indicates that the VT field of TDES3 is valid.
15:0	VT	VLAN Tag This field contains the VLAN Tag to be inserted or replaced in the packet. This field is used as VLAN Tag only when the VLTI bit of the MAC_VLAN_Incl register is set.

TDES3 Context Descriptor (Write-Back Format)

Table 754 TDES3 Context Descriptor (Write-back format

Bit	Name	Description
31	OWN	Own Bit When this bit is set, it indicates that the XGMAC DMA owns the descriptor. When this bit is reset, it indicates that the application owns the descriptor. The DMA clears this bit after it accepts the context descriptor contents.
30	CTXT	Context Type This bit is set to 1’b1 for Context descriptor.
29	CDE	Context Descriptor Error When this bit is set, it indicates that the context descriptor was provided in the incorrect sequence and the DMA ignored it. The DMA sets this bit during write-back while closing the context descriptor. When the context descriptor is read by DMA, this bit is reserved and must be having a value 1‘b0.
28:0	Rsvd	Reserved

14.4.3.23.4 Receive Descriptor

The DMA in the XGMAC attempts to read a descriptor only if the Tail Pointer is different from the Base Pointer or current pointer. It is recommended to have a descriptor ring with a length that can accommodate at least two complete packets received by the MAC. Otherwise, the performance of the DMA is impacted greatly because of the unavailability of the descriptors. In such situations, the RxFIFO in MTL becomes full and starts dropping packets.

There are two receive descriptor type: Normal and Context descriptors. All RX descriptors are prepared by the software and given to the DMA as Normal Descriptors with the content. as shown in figure "Receive Normal Descriptor (Read Format)". The DMA reads this descriptor and after transferring a received packet (or part of) to the buffers indicated by the descriptor, the Rx DMA closes the descriptor with the corresponding packet status. The format of this status is given in the section "Receive Normal Descriptor (Write-Back Format)".

For some packets, the normal descriptor bits are not enough to write the complete status. For such packets, the RX DMA writes the extended status to the next descriptor (without processing or using the Buffers Pointers embedded in that descriptor). The format and content of this descriptor write back is described in the section "Receive Context Descriptor".

Related information

Receive Normal Descriptor (Write-Back Format) on page 1572
Receive context descriptor on page 1580
Receive Normal Descriptor (Read Format) on page 1570

Receive Normal Descriptor (Read Format)

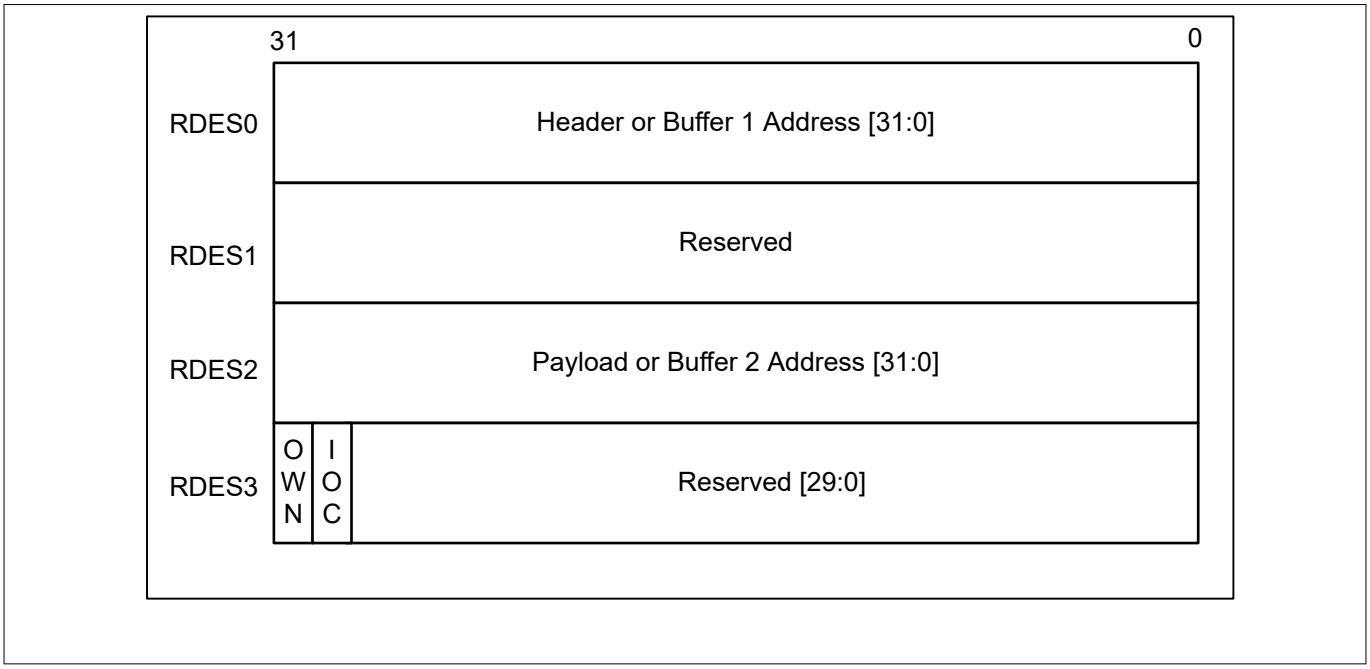


Figure 177 Receive Normal Descriptor (Read Format)

***Note**: In the Receive Descriptor (Read Format), if the Buffer Address field is all 0s, XGMAC does not transfer data to that buffer and skips to the next buffer or next descriptor.*

RDES0 Normal Descriptor (Read Format)

Table 755 RDES0 Normal Descriptor (Read Format)

Bit	Name	Description
31:0	BUF1AP	Header or Buffer 1 Address Pointer When the SPH bit of DMA_CHi_Control register of a channel is reset, these bits indicate the physical address of Buffer 1. When the SPH bit is set, these bits indicate the physical address of Header Buffer where the Rx DMA writes the L2/L3/L4 header bytes of the received packet. The software can program a byte-aligned address for this buffer which means that the LS bits of this field can be non-zero. However, while transferring the start of packet, the DMA performs a Write operation with RDES0[3:0] (or RDES0[2:0] in case of 64-bit configuration) as zero. However, the packet data is shifted as per actual offset as given by buffer address pointer. If the address pointer points to a buffer where the middle or last part of the packet is stored, the DMA ignores the offset address and writes to the full location as indicated by the data-width.

RDES1 Normal Descriptor (Read Format)

Table 756 RDES1 Normal Descriptor (Read Format)

Bit	Name	Description
31:0	BUF1AP	Reserved

RDES2 Normal Descriptor (Read Format)

Table 757 RDES2 Normal Descriptor (Read Format)

Bit	Name	Description
31:0	BUF2AP	Buffer 2 Address Pointer These bits indicate the physical address of Buffer 2. When the SPH bit of the DMA_CH(#i)_Control register is set, the buffer address pointer must be bus width-aligned, that is, RDES2[3:0 or 2:0] = 0 corresponding to 128 or 64 bus width. LSBs are ignored internally. When the SPH bit of the DMA_CH(#i)_Control register is reset, the BUF2AP can be of any byte-aligned value. However, the RxDMA uses the LS Bits of the pointer address only while transferring the start bytes of a packet. If the BUF2AP is giving the address of a buffer in which the middle or last part of a packet is stored, the DMA ignores BUF2AP[3:0 or 2:0] (corresponding to 128- or 64-bit databus) and writes to the complete location.

RDES3 Normal Descriptor (Read Format)

31	30	29:0
OWN	IOC	BUF2AP

Table 758 RDES3 Normal Descriptor (Read Format)

Bit	Name	Description
31	OWN	Own Bit When this bit is set, it indicates that the XGMAC DMA owns the descriptor. When this bit is reset, it indicates that the application owns the descriptor. The DMA clears this bit when either of the following conditions are true: • The DMA completes the packet reception • The buffers associated with the descriptor are full
30	IOC	Interrupt on Completion When this bit is set, an interrupt is issued to the application when the DMA closes this descriptor.
29:0	BUF2AP	Reserved

***Note**: In the Receive Descriptor (Read Format), if the Buffer Address field is all 0s, the XGMAC does not transfer data to that buffer and skips to the next buffer or next descriptor.*

Receive Normal Descriptor (Write-Back Format)

The RxDMA frees up a descriptor by writing it back to the system memory after transferring the received packet contents to the buffers pointed by the descriptor (Read Format). At the same time, it also writes the status information of the current packet in the format as shown in Figure 178.

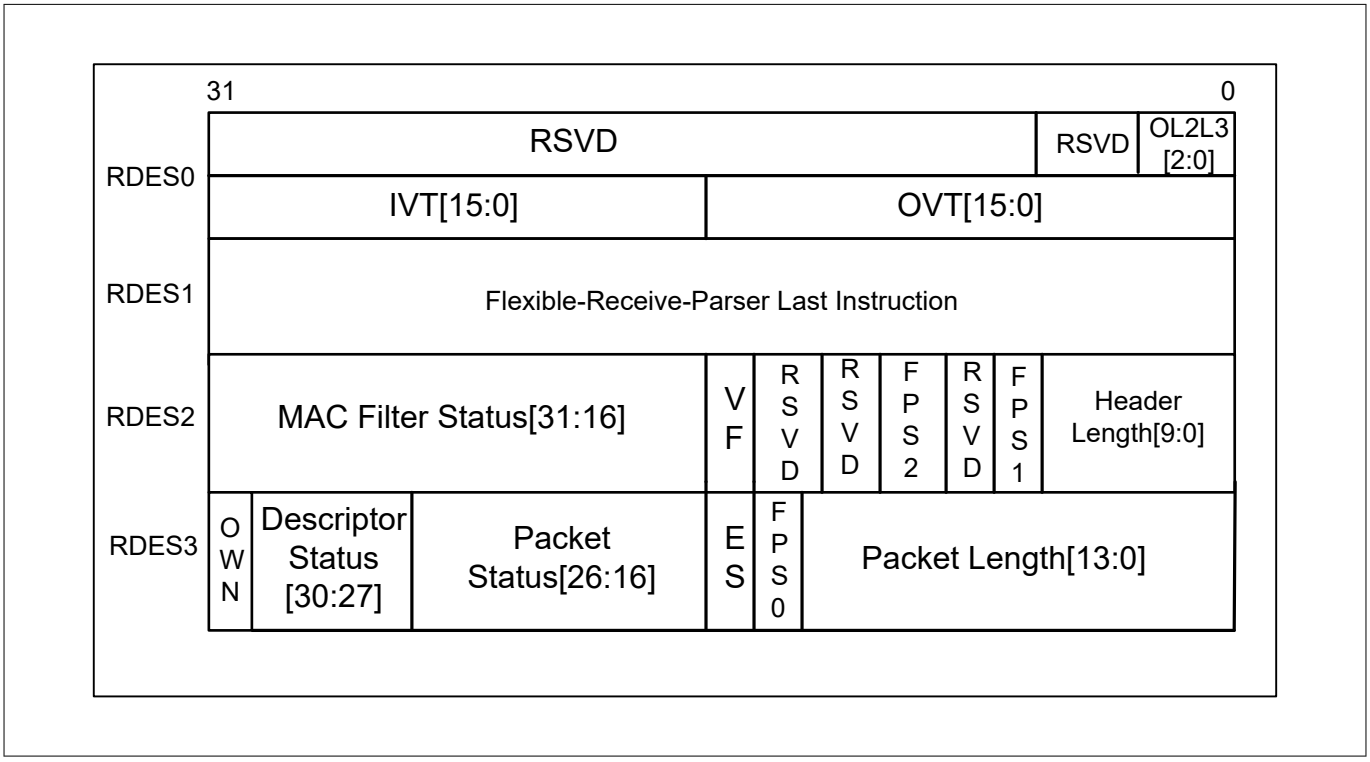


Figure 178 Receive Normal Descriptor (Write-Back Format)

Table 759 Flexible Receive Parser Receive Status

RDES2 Bit 12 FPS2	RDES2 Bit 10 FPS1	RDES3 Bit 14 FPS0	FRP Status
0	0	0	FRP ACCEPT, or result-bypass FRP not enabled (default) or FRP PASS (ACCEPT result); Destination RxDMA determined by the DCH field of the entry that gives accept result.
0	0	1	FRP Reject/Drop FRP Filter FAIL (Drop/Reject result hit during parsing)
0	1	0	Found Parsing Error • Current Instruction Number > NVE[] • Number of Parsed Instructions > NPE[] + 1 • Current Instruction's Frame offset > EOF frame offset or last parsable frame offset
0	1	1	Incomplete Parsing • Parsing is not complete FRP parsing incomplete (no result) without hitting the FRP Parsing error because the parsing of next packet cannot be delayed further.
1	0	0	FRP routing by-pass FRP PASS except that Destination RxDMA (routing) not determined by DCH field of FRP entry but taken from MAC Register filter that also declared pass
1	0	1	Stream Gate Filter fail/drop FRP PASS but corresponding StreamGate filter declared FAIL due to packet arriving outside "Gate Open" window
1	1	0	Policing Counter fail/drop FRP PASS but corresponding Policing counter declared FAIL due to crossing of set threshold
1	1	1	Safety Error Safety error (Hit ECC error while reading the FRP table).

RDES0 Normal Descriptor (Write-Back Format)

Table 760 RDES0 Normal Descriptor (Write-Back Format)

Bit	Name	Description
31:16	IVT	Inner VLAN Tag This field contains the Inner VLAN tag of the received packet. This is valid only when Double VLAN tag processing and VLAN tag stripping are enabled. This field is valid only when the L2T field of RDES0 indicates that the received packet is a Doubled VLAN tagged Packet.

(table continues...)

Table 760 (continued) RDES0 Normal Descriptor (Write-Back Format)

Bit	Name	Description
15:0	OVT	Outer VLAN Tag. This field contains the Outer VLAN tag of the received packet. This is valid only when VLAN tag stripping is enabled. This field is valid only when the L2T Field of RDES0 indicates that the received packet is a VLAN or Doubled VLAN tagged Packet.

RDES1 Normal Descriptor (Write-Back Format)

The Status fields in write-back format are valid only for the last descriptor (RDES3[26] is set).

Table 761 RDES1 Normal Descriptor (Write-Back Format)

Bit	Name	Description
31:0	FRPLI	FRP last instruction This field contains the FRP Last Instruction number where FRP result is generated.

RDES2 Normal Descriptor (Write-Back Format)

31:29	28	27	26:19	18	17	16	15	14	13	12	11	10	9:2	1	0
L3L4FM	L4FM	L3FM	MADRM	Rsvd	DAF	SAF	VF	Rsvd	Rsvd	FPS2	Rsvd	FPS1	HL	AVTDP	AVTCP

Table 762 RDES2 Normal Descriptor (Write-Back Format)

Bit	Name	Description
31:29	L3L4FM	Layer 3 and Layer 4 Filter Number Matched These bits indicate the number of the Layer 3 and Layer 4 Filter that matched the received packet: • 000: Filter 0 • 001: Filter 1 • 010: Filter 2 • 011: Filter 3 • 100: Filter 4 • 101: Filter 5 • 110: Filter 6 • 111: Filter 7 This field is valid only when Bit 28 or Bit 27 is set high. When more than one filter matches, these bits give the number of lowest filter. For a tunneled packet, only the outer L3-L4 header fields are used for comparison.

(table continues...)

Table 762 (continued) RDES2 Normal Descriptor (Write-Back Format)

Bit	Name	Description
28	L4FM	Layer 4 Filter Match When this bit is set, it indicates that the received packet matches one of the enabled Layer 4 Port Number fields. This status is given only when one of the following conditions is true: • Layer 3 filter fields are not enabled and all enabled Layer 4 filter fields as configured in Portj_MAC_L3_L4_Control(0-7) SFRs match • All enabled Layer 3 and Layer 4 filter fields in Portj_MAC_L3_L4_Control(0-7) SFRs match When more than one filter matches, this bit gives the layer 4 filter status of filter indicated by Bits[31:29]. For a tunneled packet, only the outer L3-L4 header fields are used for filter comparison.
27	L3FM	Layer 3 Filter Match When this bit is set, it indicates that the received packet matches one of the enabled Layer 3 IP Address fields. This status is given only when one of the following conditions is true: • All enabled Layer 3 fields in Portj_MAC_L3_L4_Control(0-7) SFRs match and all enabled Layer 4 fields in Portj_MAC_L3_L4_Control(0-7) SFRs are bypassed • All enabled filter fields in Portj_MAC_L3_L4_Control(0-7) SFRs match When more than one filter matches, this bit gives the layer 3 filter status of filter indicated by Bits[31:29]. For a tunneled packet, only the outer L3-L4 header fields are used for filter comparison.
26:19	MADRM	MAC Address Match Value This field contains the MAC address register number that matched the Destination address of the received packet. This field is valid only if the DAF bit is reset.
18	Rsvd	Reserved
17	DAF	Destination Address Filter Fail When this bit is set, it indicates that the packet failed the DA Filter in the MAC. For a tunneled packet, only the outer Ethernet header fields are used for filtering.
16	SAF	SA Address Filter Fail When this bit is set, it indicates that the packet failed the SA Filter in the MAC. For a tunneled packet, only the outer Ethernet header fields are used for filtering.
15	VF	VLAN Filter Status When this bit is set, it indicates that the VLAN Tag of received packet passed the VLAN filter. For a tunneled packet, only the outer Ethernet header fields are used for filtering.
14	Rsvd	Reserved
13	Rsvd	Reserved
12	FPS2	Flexible Receive Parser Status bit
11	Rsvd	Reserved
10	FPS1	Flexible Receive Parser Status bit

(table continues...)

Table 762 (continued) RDES2 Normal Descriptor (Write-Back Format)

Bit	Name	Description
9:2	HL	L3/L4 Header Length This field contains the length of the header of the packet split by the MAC at L3 or L4 header boundary as identified by the MAC receiver. This field is valid only when the first descriptor bit is set (FD = 1). The header data is written to the Buffer 1 address of corresponding descriptor. If header length is zero, it implies that the MAC did not identify and split the header and both Buffer 1 and Buffer 2 are used for storing the packet.
1	AVTDP	AV Tagged Data Packet When L34T=0000, this bit is set to 1 when AV Tagged Data packet is received When L34T is non-zero, this bit is HL[1] for packet split by the MAC at L3 or L4 header.
0	AVTCP	AV Tagged Control Packet When L34T=0000, this bit is set to 1 when AV Tagged Control packet is received When L34T is non-zero, this bit is HL[0] for packet split by the MAC at L3 or L4 header.

RDES3 Normal Descriptor (Write-Back Format)

31	30	29	28	27	26	25	24	23:20	19:16	15	14	13:0
OWN	CTXT	FD	LD	CDA	Rsvd	Rsvd	ETM	L34T	ET/LT	ES	FPS0	PL

Table 763 RDES3 Normal Descriptor (Write-Back Format)

Bit	Name	Description
31	OWN	Own Bit When this bit is set, it indicates that the XGMAC DMA owns the descriptor. When this bit is reset, it indicates that the application owns the descriptor. The DMA clears this bit when either of the following conditions is true: • The DMA completes the packet reception • The buffers associated with the descriptor are full
30	CTXT	Receive Context Descriptor When this bit is set, it indicates that the current descriptor is a context type descriptor. The DMA writes 1'b0 to this bit for normal receive descriptor.
29	FD	First Descriptor When this bit is set, it indicates that this descriptor contains the first buffer of the packet. If the size of the first buffer is 0, the second buffer contains the beginning of the packet. If the size of the second buffer is also 0, the next descriptor contains the beginning of the packet. CTXT, FD, and LD bits together indicate Descriptor Definition Error. During write-back, all three bits are set to 1 {CTXT, FD, LD} = 3'b111 to indicate Descriptor Definition Error in the Rx Descriptor. Descriptor Definition Error for Rx Descriptor is indicated when both buffer 1 and buffer 2 are all 1s.
28	LD	Last Descriptor When this bit is set, it indicates that the buffers to which this descriptor is pointing are the last buffers of the packet.
27	CDA	Context Descriptor Available When this bit is set, the next descriptor following this descriptor is a Context Descriptor.
26	Rsvd	Reserved

(table continues...)

Table 763 (continued) RDES3 Normal Descriptor (Write-Back Format)

Bit	Name	Description
25	Rsvd	Reserved
24	ETM	Ether Type Match/No Coagulation Packet When L34T=000, this bit is set to 1 (ETM Status) when the “Type” field of the received packet matches the value programmed in the MAC_Rx_Eth_Type_Match register. When L34T has any value other than 000, this bit is 0.
23:20	L34T	Layer3/Layer4 packet Type These bits indicate the type of payload encapsulated in the IP datagram and processed by the Receive Checksum Offload Engine (RxCOE). For a tunneled packet, the Layer3/Layer4 packet type refers to the L3/L4 type of the inner packet. • 0000: Not an IP packet • 0001: IPv4 packet with TCP Payload as Layer4 • 0010: IPv4 packet with UDP Payload as Layer4 • 0011: IPv4 packet with ICMP Payload as Layer4 • 0100: IPv4 packet with IGMP Payload is Layer4 • 0101 to 0110: Reserved • 0111: IPv4 with unknown (unsupported) payload • 1000: Reserved • 1001: IPv6 packet with TCP Payload as Layer4 • 1010: IPv6 packet with UDP Payload as Layer4 • 1011: IPv6 packet with ICMP Payload as Layer4 • 1100 to 1110: Reserved • 1111: IPv6 with unknown (unsupported) payload

(table continues...)

Table 763 (continued) RDES3 Normal Descriptor (Write-Back Format)

Bit	Name	Description
19:16	ET/LT	Error Type or L2 Type When the 15th bit of this descriptor (ES) is set to 1, this field indicates the Error Type. When the 15th bit of this descriptor (ES) is set to 0, this field indicates the L2 Packet Type. Following is the encoding: When ES = 1 (Error Type Encoding) • 0000: Reserved • 0001: Watchdog Timeout Error It indicates that the Receive Watchdog Timer has expired while receiving the current packet. The current packet is truncated after watchdog timeout • 0010: Invalid Code Error or GMII Error In XGMII mode, it indicates that the frame was terminated due to the reception of an invalid code character. In GMII mode, it indicates that the RX_ER and RX_DV signals were asserted for the frame, indicating a receive error signaled to the GMII. • 0011: CRC Error It indicates that a Cyclic Redundancy Check (CRC) Error occurred on the received packet. • 0100: Giant Packet Error It indicates that the packet length exceeds the specified maximum Ethernet size of 1518, 1522, or 2000 bytes (9018 or 9022 bytes if jumbo packet enable is set). Note: Giant packet indicates only the packet length. It does not cause any packet truncation. • 0101: IP Header Error It indicates that the 16-bit IPv4 Header checksum calculated by the XGMAC controller does not match the received Checksum bytes or that the IP datagram version is not consistent with the Ethernet Type value. This error is possible only when you select the Enable Receive TCP/IP Checksum Check feature • 0110: Payload (L4) Checksum Error It indicates that the 16-bit IP payload checksum (that is, the TCP, UDP, or ICMP checksum) calculated by the XGMAC controller does not match the received TCP, UDP, or ICMP checksum field. • 0111: Overflow Error It indicates that the received packet is damaged because of buffer overflow in Rx FIFO. Note: This bit is set only when the DMA transfers a partial packet to the application. This happens only when the Rx FIFO is operating in the threshold mode. In the store-and-forward mode, all partial packets are dropped completely in Rx FIFO. • 1000: Bus Error It indicates that AXI Bus Error response is received while fetching this descriptor or when packet is being transferred to its buffers. Continued on next page

(table continues...)

Table 763 (continued) RDES3 Normal Descriptor (Write-Back Format)

Bit	Name	Description
19:16	Cont'd	• 1001: Length Error It indicates that the Length value of the current frame is inconsistent with the total number of bytes received in the current frame. This is applicable when Frame Length value is less than 1500. This error is reported when the following type of frames are received: - Frames in which the value of the Length field is less than 46 bytes and the length of the received frame is less than the following: Length field + 18 bytes - Frames in which the value of the Length field is equal to or greater than 46 bytes and the length of the received frame is not equal to the following: Length field + 18 bytes • 1010: Good runt packet. Packet less than 64 bytes but with good CRC • 1011: Reserved • 1100: Dribble Error Indicates that, in the 10/100Mbps speed mode, the packet has ended without a byte boundary • 1101 -1111: Reserved for non-tunneled packets In the case of Tunneled packet, the Error Status is extended to differentiate between the errors in outer Header or Inner Header. In the status types, the following values are applicable for the Outer Header fields: • 0101: Outer IP Header Error • 0110: Outer Header's Payload (L4) Checksum Error Additionally, the following Error Status types are applicable • 1001: Inner Packet's IP Header Error • 1010: Inner Packet's Payload (L4) Checksum Error • 1011: Reserved • 1101-1111: Reserved When ES = 0 (L2 Packet Type Encoding) • 4'b0000: The packet is a length packet • 4'b0001: The packet is a MAC Control packet • 4'b0010: Reserved • 4'b0011: The packet is of ARP Request Packet Type • 4'b0100: The packet is of OAM packet type.(Note: Inner header of a tunneled packet is not parsed for OAM Control type) • 4'b0101: The untagged packet has a Type field which matched the value in the MAC_Rx_Eth_Type_Match register • 4'b0110: Untagged AV Control Packet • 4'b0111: The packet is a other Type Packet • 4'b1000: The packet is type packet with Single SVLAN tag • 4'b1001: The packet is type packet with Single CVLAN tag • 4'b1010: The packet is type packet with Double VLAN tag with outer VLAN is CVLAN and inner is also CVLAN Continued on next page

(table continues...)

Table 763 (continued) RDES3 Normal Descriptor (Write-Back Format)

Bit	Name	Description
19:16	Cont'd	• 4'b1011: The packet is type packet with Double VLAN tag with outer VLAN is SVLAN and inner is SVLAN • 4'b1100: The packet is type packet with Double VLAN tag with outer VLAN is SVLAN and inner is CVLAN • 4'b1101: The packet is type packet with Double VLAN tag with outer VLAN is CVLAN and inner is SVLAN • 4'b1110 – 4'b1111: Reserved For a tunneled packet, the L2 packet type encoding refers to the L2 type of the Inner packet.
15	ES	Error Summary This bit indicates that the received packet has an error. The type of error is indicated by bits [19:16]. This field is valid only when the LD bit of RDES3 is set.
14	FPS0	Flexible Receive Parser Status bit
13:0	PL	Packet Length These bits indicate the byte length of the received packet that was transferred to system memory. This does not include the VLAN/Pad/CRC bytes if VLAN/Pad/CRC stripping is enabled. This field is valid when the LD (bit[28]) field of RDES3 is set there is no overflow error indication in ET/LT(bits[19:16]).

Receive context descriptor

The Receive context descriptor provides information about the extended status of the last received packet. The Bit 30 of RDES3 indicates the context type descriptor.

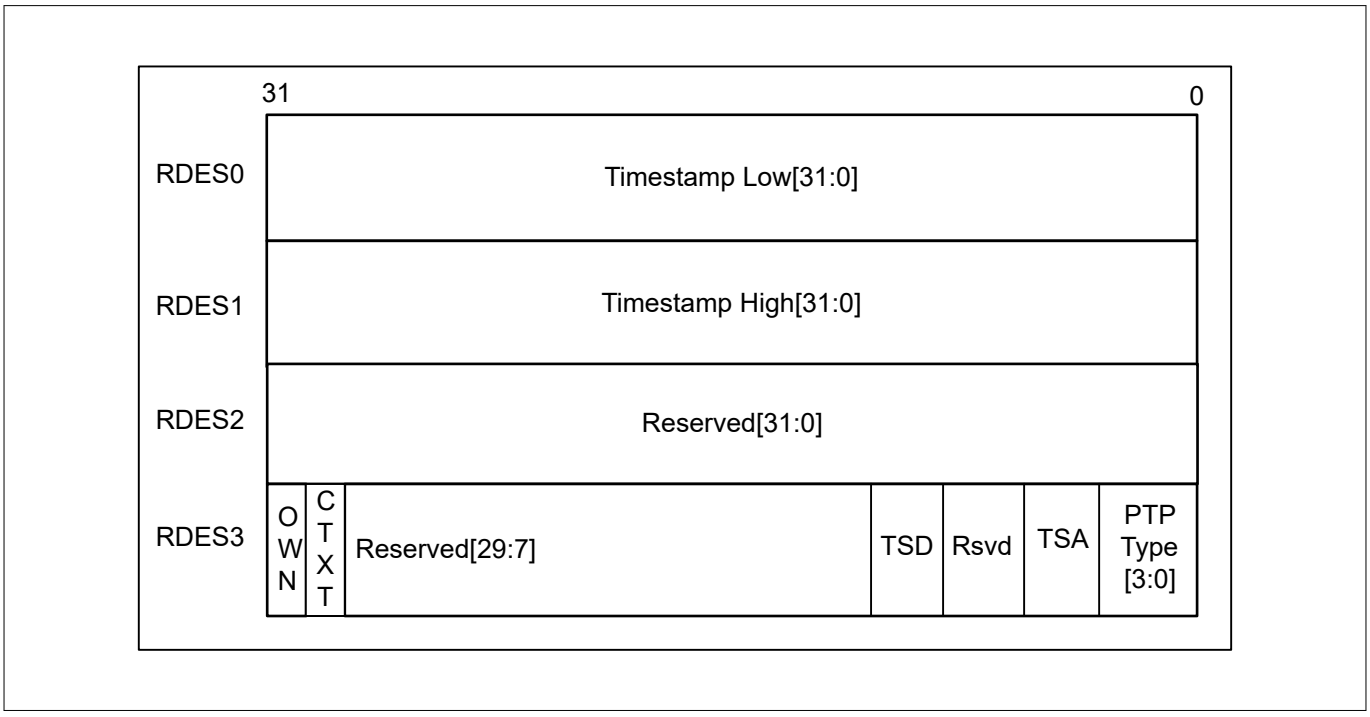


Figure 179 Receive context descriptor

RDES0 context descriptor

Table 764 RDES0 context descriptor

Bit	Name	Description
31:0	RTSL	Receive Packet Timestamp Low The DMA updates this field with least significant 32 bits of the timestamp captured for corresponding Receive packet. When this field and the RTSH field of RDES1 show all-ones value, the timestamp must be considered as corrupt.

RDES1 context descriptor

Table 765 RDES1 Context Descriptor

Bit	Field	Description
31:0	RTSH	Receive Packet Timestamp High The DMA updates this field with most significant 32 bits of the timestamp captured for corresponding receive packet. When this field and the RTSL field of RDES0 show all-ones value, the timestamp must be considered as corrupt.

RDES2 context descriptor

Table 766 RDES2 context descriptor

Bit	Description
31:0	Reserved

RDES3 context descriptor

31	30	29:7	6	5	4	3:0
OWN	CTXT	Rsvd	TSD	Rsvd	TSA	PMT

Table 767 RDES3 context descriptor

Bit	Name	Description
31	OWN	Own Bit When this bit is set, it indicates that the DMA owns the descriptor. When this bit is reset, it indicates that the application owns the descriptor. The DMA clears this bit when either of the following conditions is true: • The DMA completes the packet reception • The buffers associated with the descriptor are full
30	CTXT	Receive Context Descriptor When this bit is set, it indicates that the current descriptor is a context descriptor. The DMA writes 1'b1 to this bit for context descriptor.
29:7	Rsvd	Reserved
6	TSD	Timestamp Dropped When set, it indicates that a timestamp was captured for this packet but was dropped due to the lack of space in RxFIFO. The host should consider the RTSL and RTSH as valid only when TSD=0 and TSA=1

(table continues...)

Table 767 (continued) RDES3 context descriptor

Bit	Name	Description
5	Rsvd	Reserved
4	TSA	Timestamp Available When set, this bit indicates that the timestamp value is available in RDES0 and RDES1 of the context descriptor.
3:0	PMT	PTP Message Type: These bits are encoded to give the type of the PTP message received: • 0000: No PTP message received • 0001: SYNC (all clock types) • 0010: Follow-up (all clock types) • 0011: Delay_Req (all clock types) • 0100: Delay_Resp (all clock types) • 0101: Pdelay_Req (in peer-to-peer transparent clock) • 0110: Pdelay_Resp (in peer-to-peer transparent clock) • 0111: Pdelay_Resp_Follow_Up (in peer-to-peer transparent clock) • 1000: Announce • 1001: Management • 1010: Signaling • 1011-1110: Reserved • 1111: PTP packet with Reserved message type

14.4.3.23.5 Split Header Support

This feature supports splitting the packet header and the payload of received packets in separate buffers (buffer 1 and buffer 2). You can enable the split header feature by selecting the Enable Split Header Structure option and setting the SPH bit in the DMA_CH(#i)_Control register.

HDSMS field of Port j MAC Rx Configuration register indicates maximum header size allowed for splitting the header data in the received packet based on the value programmed. When the ARBS field of the DMA_CHj_Rx_Control2 register is programmed to a non-zero value, ARBS field indicates Receive Buffer size for buffer 1 and the RBSZ field indicates the receive buffer size for buffer 2. The ARBS field can be set to 0, for backward compatibility.

The value in the HDSMS field should be greater than or equal to the buffer 1 size indicated by ARBS field for all the channels.

Table 768 shows how DMA processes a packet based on the packet type.

Table 768 Split Header Support Depending on the Packet Type

Packet Type	SPLM	Description
TCP or UDP Packet	00(L3/L4 Split)	The DMA writes the Ethernet header + IP header + TCP or UDP header into the header buffer.
IP packet (not TCP/ UDP)		The DMA writes the Ethernet header + IP header into the header buffer.
Non-IP packet		The DMA does not split the header and payload

(table continues...)

Table 768 (continued) Split Header Support Depending on the Packet Type

Packet Type	SPLM	Description
Any packet	01 (L2 Split)	The DMA writes the Ethernet headerbased on split offset (SPLOFST) into the header buffer.
IP packet	10(Combination of L2 or L3/L4 Split)	L3/L4 split
Non-IP packet		L2 split
NA	11	Reserved

The IP header includes IPv4 options in case of IPv4 packet and IPv6 extension headers in case of IPv6 frames. The points at which the header is split are shown in Figure 180.

***Note**:*

- *The VLAN tag stripping must be set for the split function. For instance, the DMA separates the header and payload of an untagged packet only. So, when a tagged packet is received, then program XGMAC such that the VLAN tags are deleted/stripped from the received packets*
- *L3/L4 split is applicable for IP packets that are either untagged or VLAN stripped. If VLAN tag is retained in the packet forwarded to the DMA, L3-L4 split is not performed. However, if SPLM field is set to 2, L2 split is performed for VLAN tagged IP packets*
- *For AV packets, to specify an alternative L2 split value, use the SAVE and SAVO fields of the Port j MAC Ext Cfg1 register*
- *L2 Split is not supported for packets with variable preamble*

Table 769 shows the header length availability (HL) in RDES2. This field is valid only when FD = 1 (First Descriptor).

Table 769 Header Length Availability in RDES2

Value of FD	Value of LD	Header Length (HL) Availability in RDES2
0	1	HL not available
1	0	HL is [9:0]
1	1	• IP packets, HL is [9:0] • Non-IP packets, HL is [9:2]

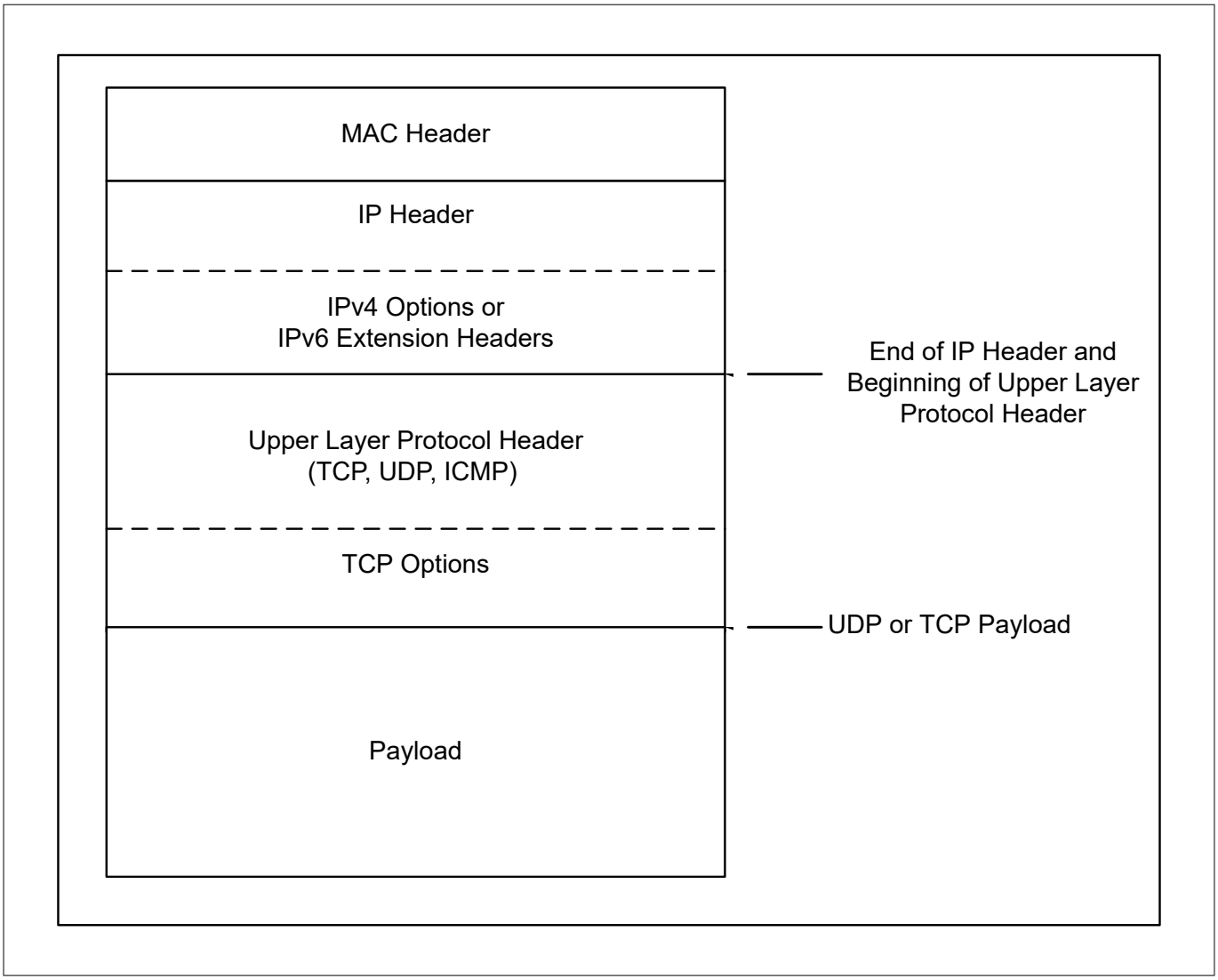


Figure 180 Header Split Points

Descriptor Structure

Figure 181 shows the descriptor structure without the Split Header feature.

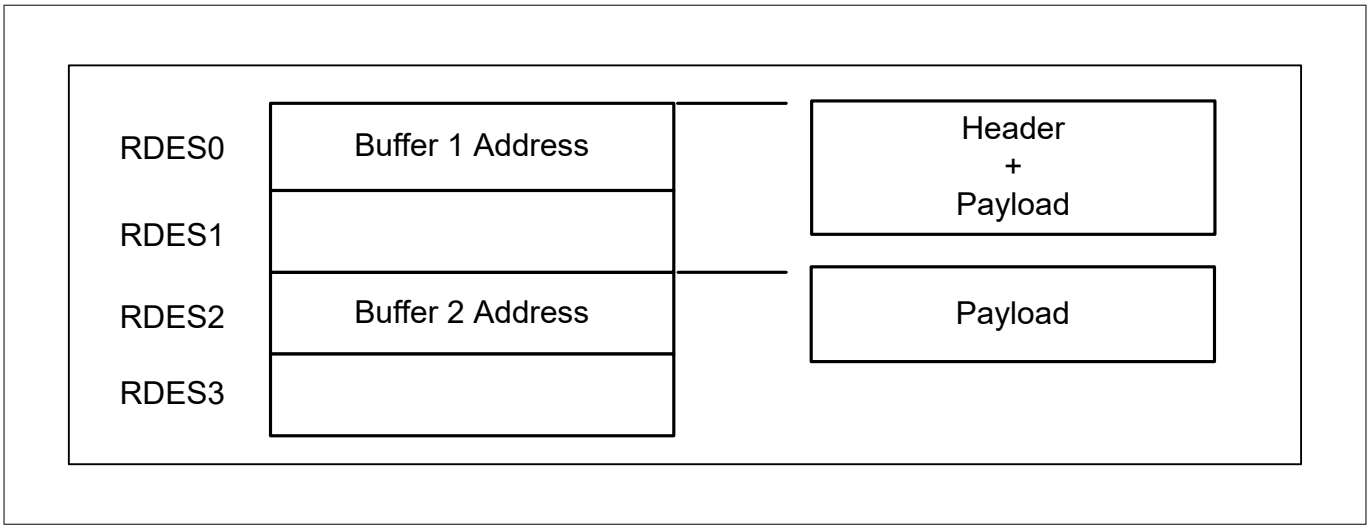


Figure 181 Descriptors without Split Header Feature

Figure 182 shows the descriptor structure with the Split Header feature.

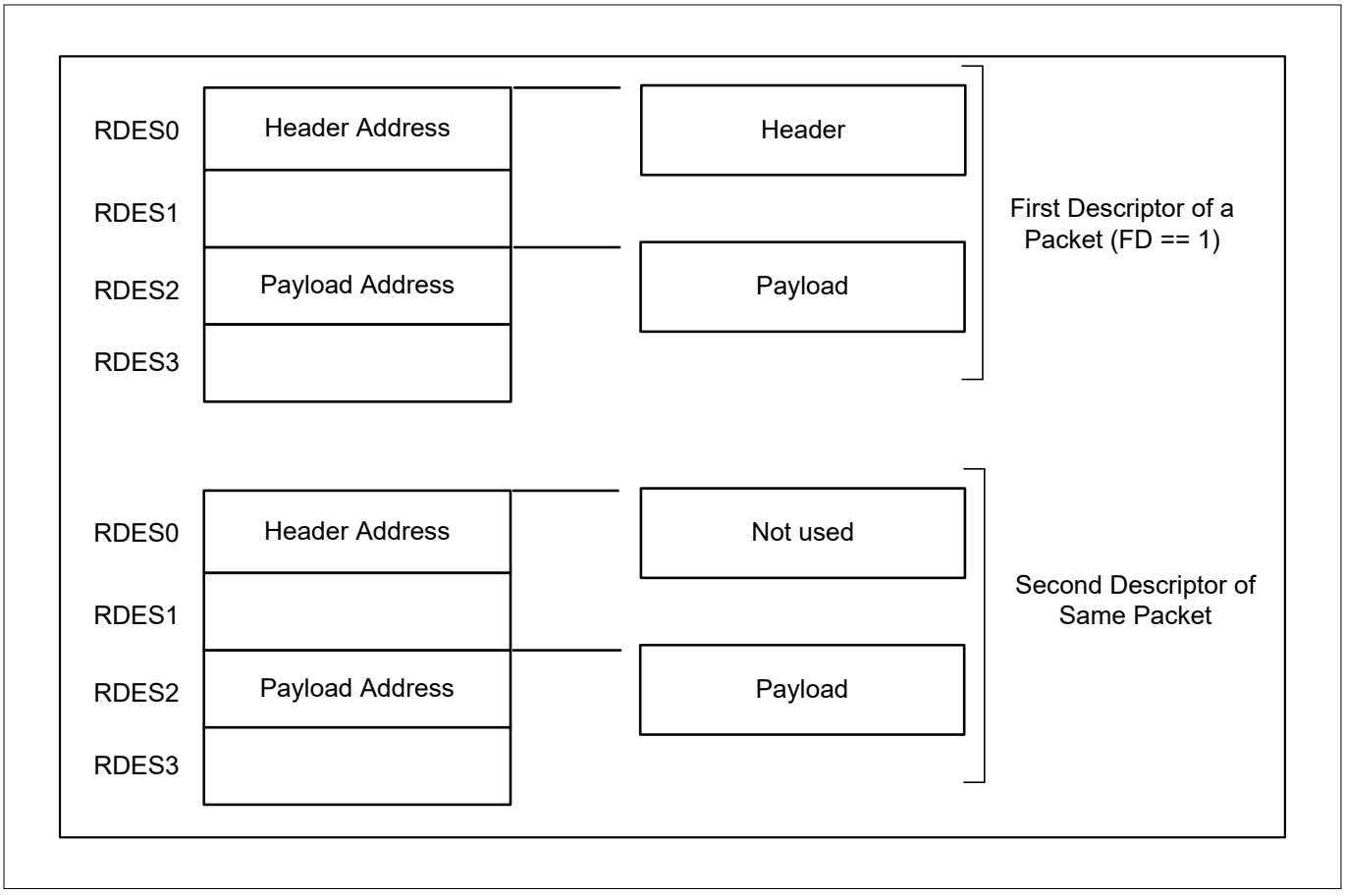


Figure 182 Descriptors with Split Header Feature

The DMA writes the header of the received packet by using the header address to which the RDES0 in the first descriptor is pointing (FD bit of RDES3 is set). The DMA writes the payload of the received packet into the buffer address to which the RDES2 is pointing. For subsequent descriptors (FD is set to 0), the address to which RDES0 (Header address) is pointing is not used. The payload is written only to buffers to which the RDES2 (payload address) is pointing.

The DMA writes the header length in RDES2 of the first receive descriptor (RDES3[29] (FD Bit) is set) for the packet. The packet length is written in RDES3 of the last receive descriptor (RDES3[28] (LD Bit) set). The buffer length for the payload is set by the driver through the RBSZ field in the corresponding DMA Channel Receive Control Register. The DMA fills receive buffers fully in all descriptors except the last receive descriptor. The maximum header length is limited by the value programmed in the HDSMS field of MAC_Rx_Configuration register.

Figure 183 shows the descriptor structure with the Split Header feature for a tunneled packet.

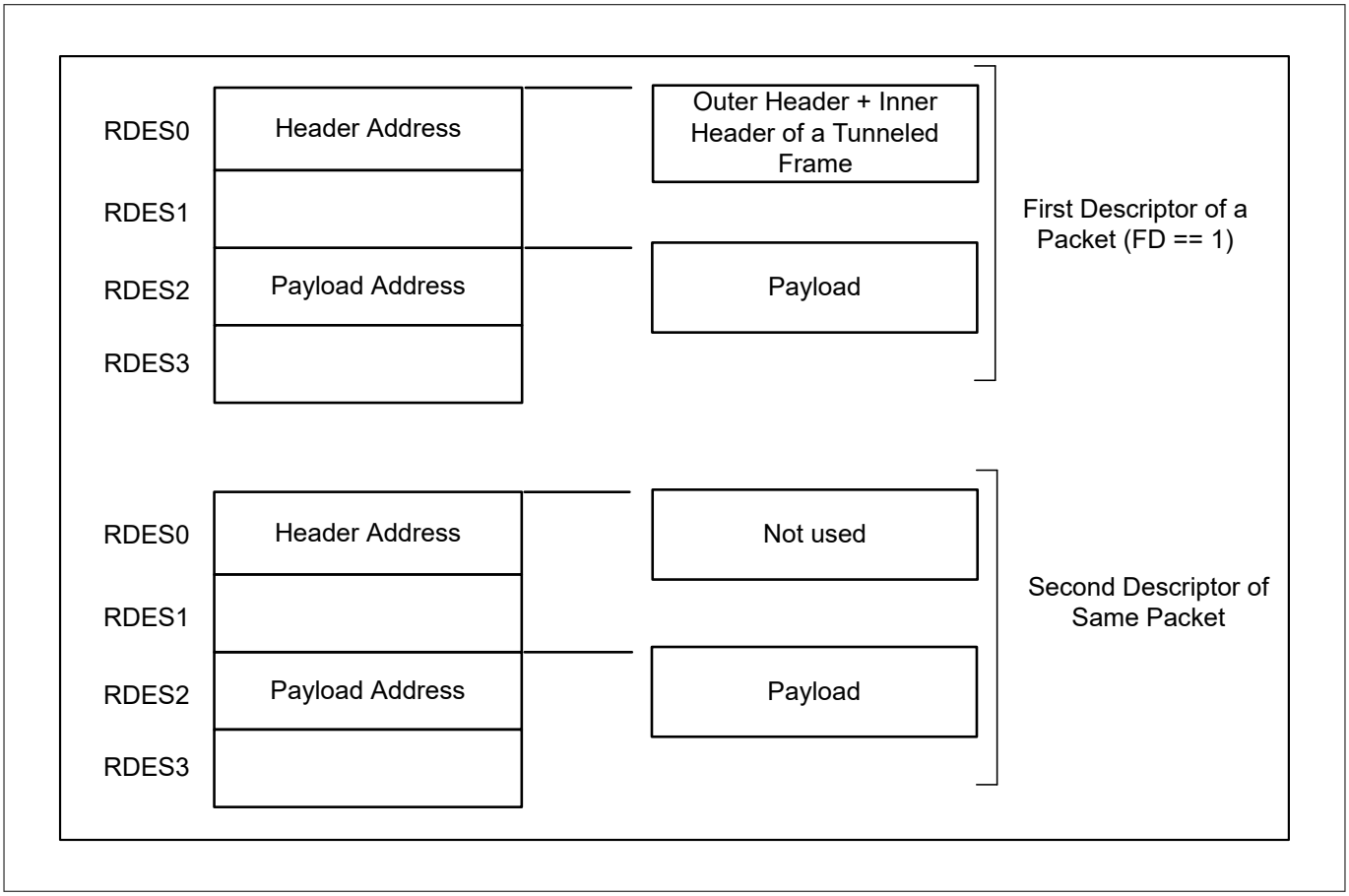


Figure 183 Descriptors with Split Header for Tunneled Packets

The header of Outer Packet and Inner Packet are part of data written in the Header buffer. Only the Payload of the Inner packet is written into the buffers pointed by RDES2.

14.4.3.23.6 Enhanced descriptor for time-based scheduling

The time-based scheduling feature needs Enhanced Descriptors (that are 32 bytes) to be enabled on all the required DMA channels that intend to use the feature by setting the EDSE bit of DMA_CH(#i)_TX_Control register.

The structure of 32-byte Descriptor for the Context and the Normal Descriptor in Read and Write formats are described in the following sections:

Enhanced normal descriptor - read (32-bit mode)

The fields present in the first 16 bytes of the Enhanced Descriptor format of normal descriptor are as follows:

- LTV (Launch Time Valid)
  When FD = 0, this field indicates that the Launch Time (LT) and GSN fields present in the descriptor are valid. Set this field to 1, only if the FD field of the descriptor is 0.

- GSN (GCL Slot Number):
  This field indicates the GCL slot number associated with the packet
- Launch Time (LT):
  The launch time is a 32-bit value, where most-significant 8-bits represent the time in seconds and the rest LT[23:0] represents the launch time in increments of 256 ns. The launch time is compared against the IEEE 1588 based System/PTP Time (bits[39:8]) and rolls over after 256 seconds.

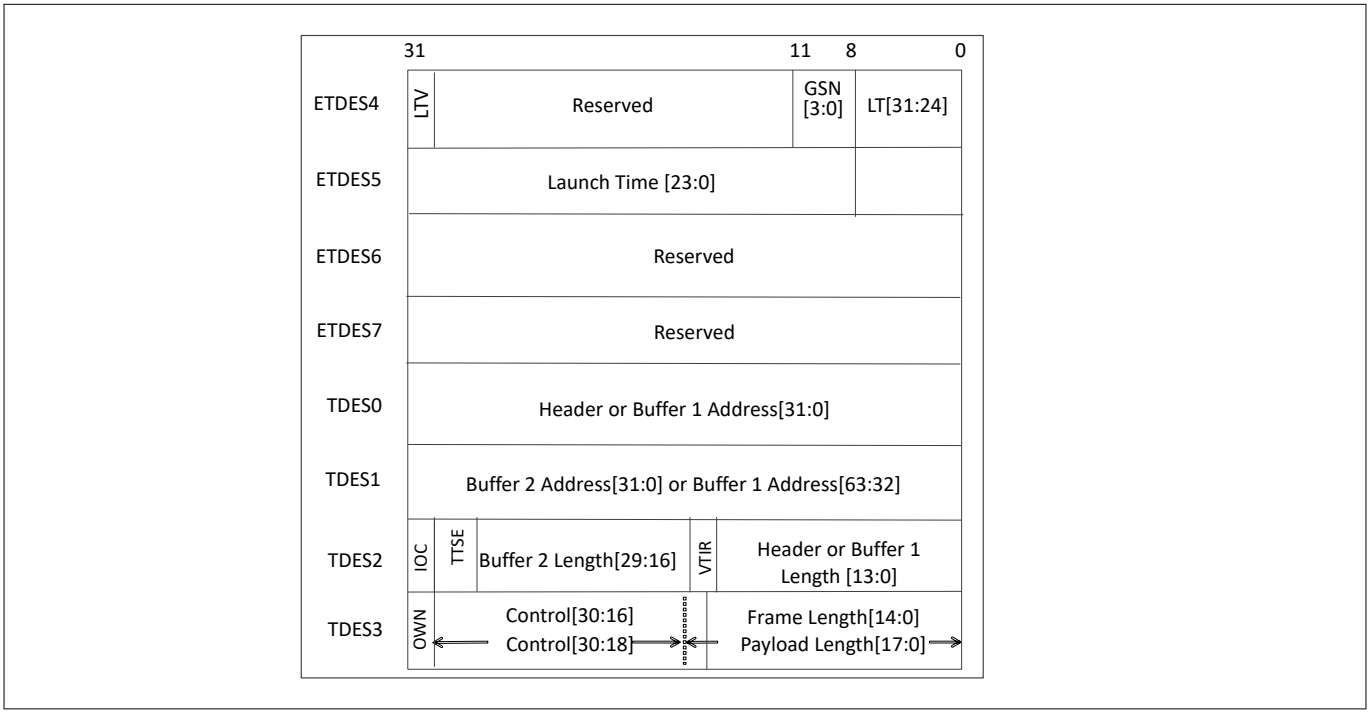


Figure 184 Enhanced normal Descriptor - read (32-bit mode)

Enhanced normal descriptor (write, 32-bit mode)

When the DMA closes the enhanced descriptor, it updates only the TDES0, TDES1, TDES2, and TDES3 fields. So, the values in the TDES4-TDES7 fields are retained in the host memory.

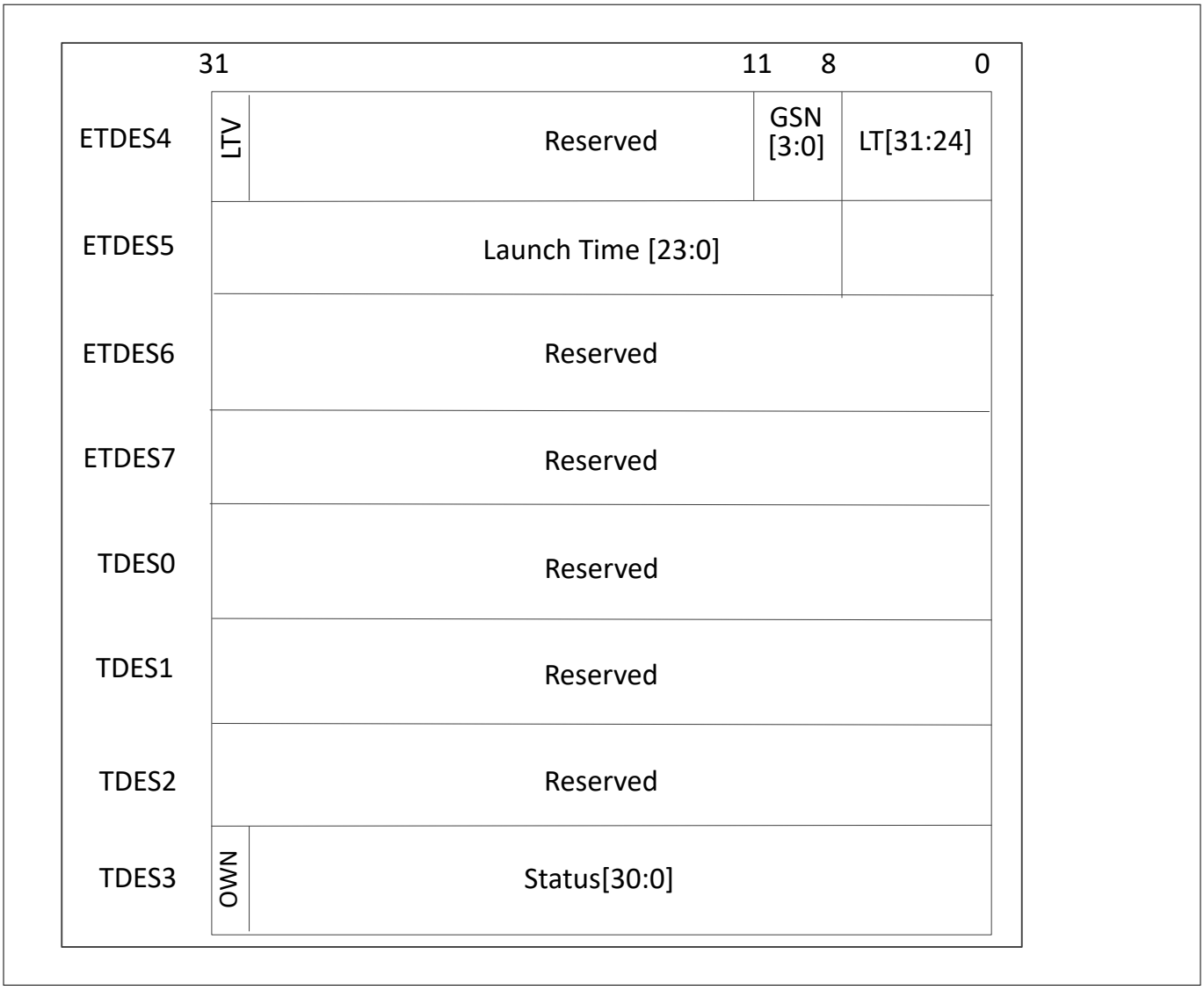


Figure 185 Enhanced normal descriptor (write, 32-bit mode)

Note:

- *All the enhanced descriptor examples assume a 32-bit data width configurations*
- *In both the write back formats, no modifications can be done to the extended 16 bytes. The rest of the 16 bytes (TDESC0 to TDESC3) are written back as per the previous 16 bytes Descriptor format*
- *When enabled, fetch time overrides the AV slot function*
- *When an unaligned new GCL list (with a different CTR) is installed, it is recommended not to have traffic during the installation of the new list. Any traffic during the switching of the lists might have unpredictable behavior regarding Fetch, Launch, and Launch expiry as the CTR and BTR values get updated while the frame is being processed*

Enhanced context descriptor (read, 32-bit mode)

The first 16 bytes of the Enhanced context descriptor (ETDESC4 to ETDESC7) are reserved and must be zeros. The fields present in last bytes of the Descriptor (TDESC0 to TDESC3) are same as the context descriptor (TDESC0 to TDESC3) in 16 byte format.

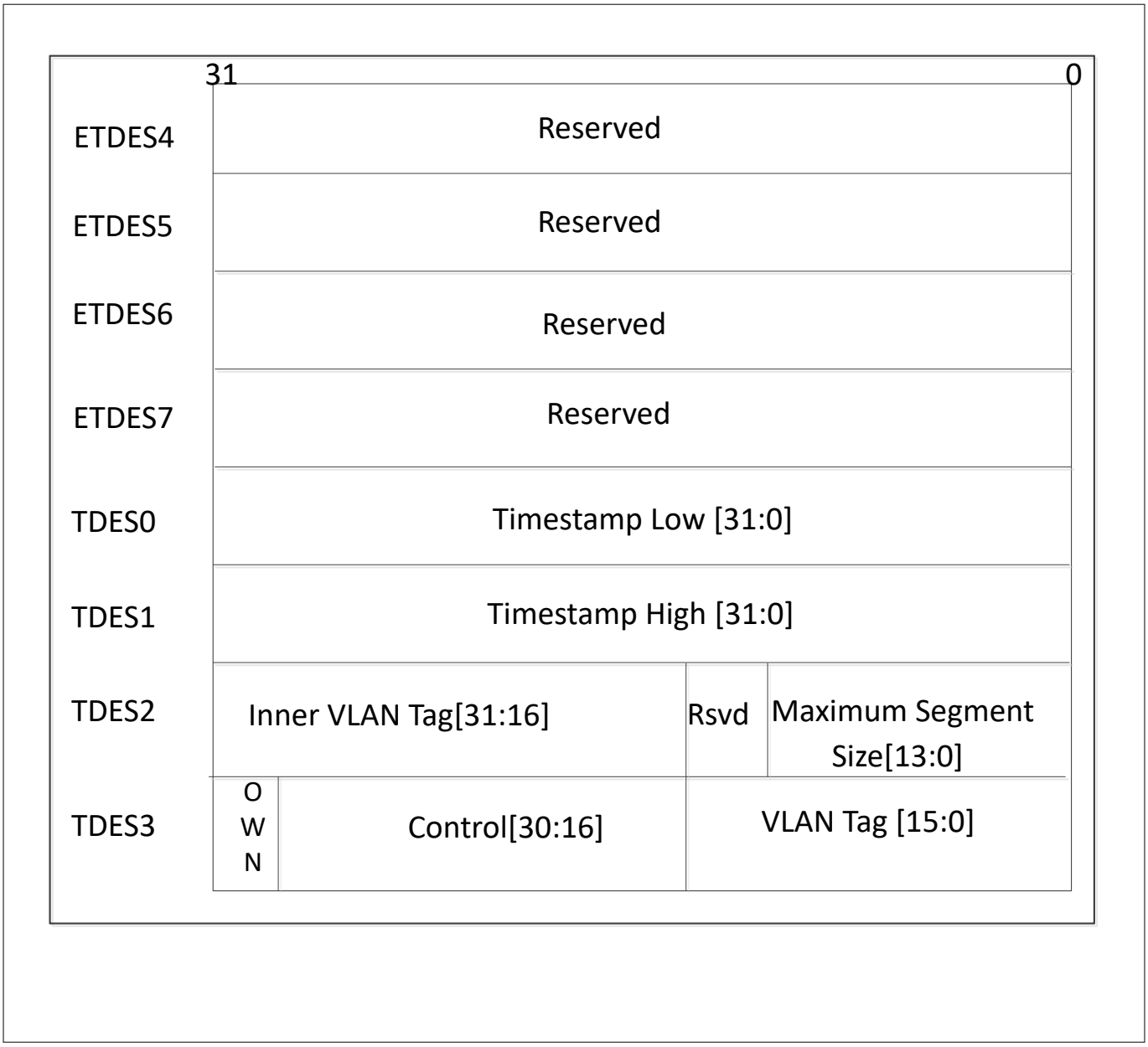


Figure 186 Enhanced context descriptor - read

Enhanced context descriptor (write, 32-bit mode)

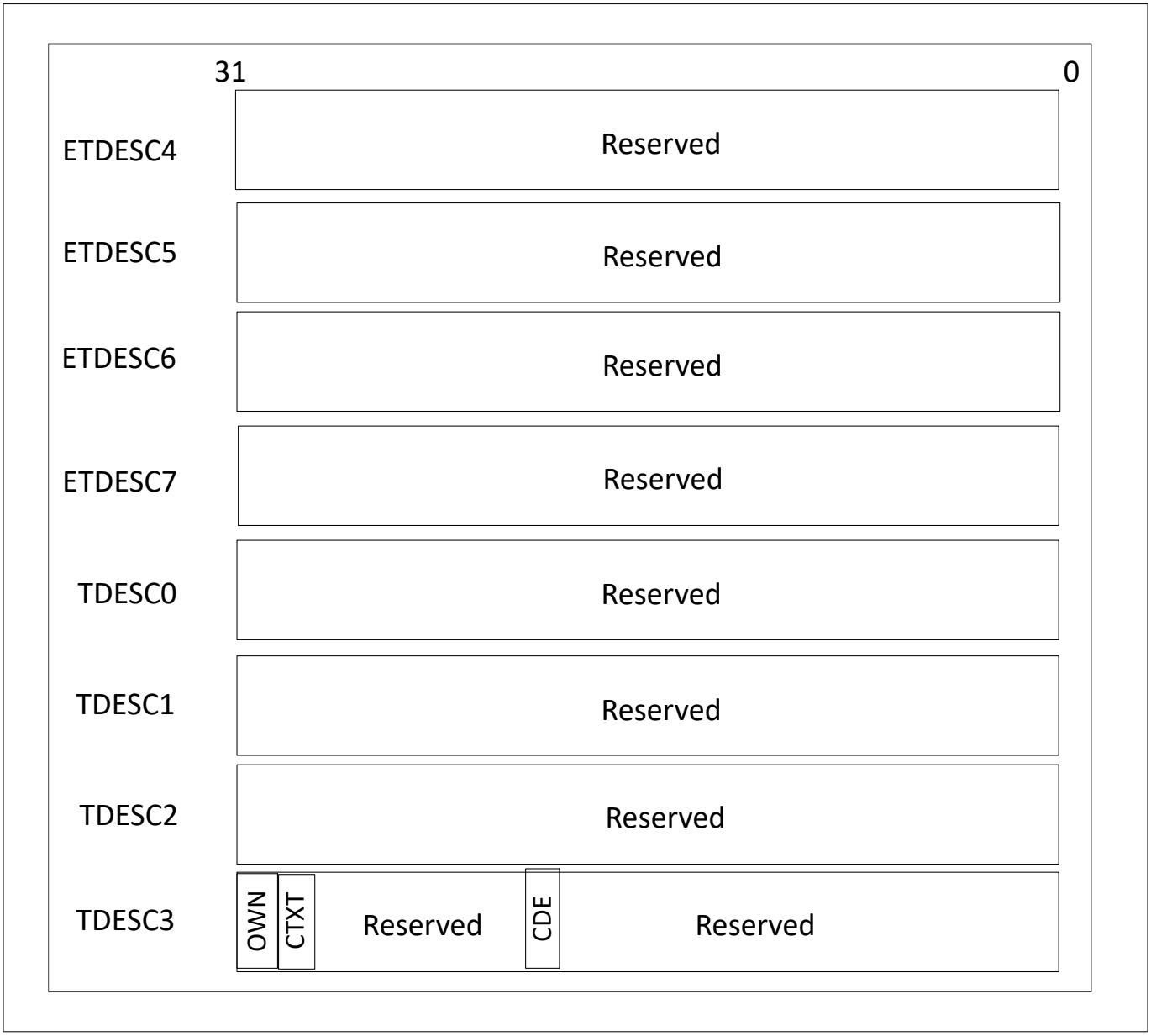


Figure 187 Enhanced context descriptor - write

***Note**: In both the write back formats, no modifications are done to the extended 16 bytes. The rest of the 16 bytes (TDESC0 to TDESC3) are written back as per the previous 16 bytes of the Descriptor format.*

14.4.3.24 IO Interfaces

14.4.3.24.1 Connectivity

Definition of the PHY Interfaces (MII, RMII, RGMII and (UX)SGMII) and the connection of the policing counter inputs from the PTP sub-module.

The module has up to four PHY interfaces. Some products may contain less.

HSPHY controls all of these interfaces. HSPHY needs to be configured before GETH. All connections to external pins are routed from GETH to HSPHY. Refer to High Speed Physical Layer (HSPHY) for correct setup.

Table 770 PHY interfaces

IF	Signal name in standard	GETH signal name	I/O
MII (via HSPHY)	TX_EN	TXEN	O
	TXD[3:0]	TXD[3:0]	O
	TX_ER	TXER	O
	TX_CLK	TXCLK	I
	COL	COL	I
	CRS	CRS	I
	RX_DV	RXDV	I
	RXD[3:0]	RXD[3:0]	I
	RX_ER	RXER	I
	RX_CLK	RXCLK	I
RMII (via HSPHY)	TX_EN	TXEN	O
	TXD[1:0]	TXD[1:0]	O
	CRS_DV	CRSDV	I
	RXD[1:0]	RXD[1:0]	I
	REF_CLK	REFCLK	I
RGMII (via HSPHY)	TD[3:0]	TXD[3:0]	O
	TX_CTL	TCTL	O
	TXC	TXCLK	O
	n. a.	GREFCLK (Reference input)	I
	RD[3:0]	RXD[3:0]	I
	RX_CTL	RCTL	I
	RXC	RXCLK	I
SGMII or UXSGMII (via HSPHY)	TX	SOP (serial out positive signal of the differential pair)	O
	TX	SON (serial out negative signal of the differential pair)	O
	RX	SIP (serial in positive signal of the differential pair)	I
	RX	SIN (serial in negative signal of the differential pair)	I
MDIO (via HSPHY)	MDC	MDC	O
	MDIO	MDIO	I/O
Pulse Per Second signal from Precision Time Protocol (ptp_pps_o[0][0]) MAC0, PPS0 (via PORTS)			O

(table continues...)

Table 770 (continued) PHY interfaces

IF	Signal name in standard	GETH signal name	I/O
Pulse Per Second signal from Precision Time Protocol (ptp_pps_o[0][1]) MAC0, PPS1 (via PORTS)			O
Pulse Per Second signal from Precision Time Protocol (ptp_pps_o[1][0]) MAC1, PPS0 (via PORTS)			O
Pulse Per Second signal from Precision Time Protocol (ptp_pps_o[1][1]) MAC1, PPS1 (via PORTS)			O

Each MAC has 16 policing counters. Each of them can use timer inputs to operate in a time windows mode. The time base for this time windows mode can be selected from the Pulse Per Second signals (PPS). These PPS signals are generated in the precision time protocol sub-module (PTP) described in chapter "IEEE 1588 Time Stamps". Each MAC generates two PPS outputs. So in sum there are four PPS outputs available in the system. Each of them can be selected as time window base for any of the policing counters by programming Register MAC_PCCtrl_IndReg[policing-counter-number].TWSEL (Time window Select) as described below:

Table 771 Connections of the policing counter inputs from the PTP sub-module (PPS to time window select)

PPS output	PORT[1:0].MAC_PCCtrl_IndReg[policing-counter-number].TWSEL
Pulse Per Second signal from Precision Time Protocol (ptp_pps_o[0][0]) MAC0, PPS0	00_B
Pulse Per Second signal from Precision Time Protocol (ptp_pps_o[0][1]) MAC0, PPS1	01_B
Pulse Per Second signal from Precision Time Protocol (ptp_pps_o[1][0]) MAC1, PPS0	10_B
Pulse Per Second signal from Precision Time Protocol (ptp_pps_o[1][1]) MAC1, PPS1	11_B

14.4.3.24.2 Clocks

The module has multiple clock inputs and outputs connecting it to the system.

Table 772 Clock lines of ethernet MAC

Clock line	Connected to	Description
aclk_i, clk_csr_i	f_{GETH}	This is the free-running clock input provided by the clock generation unit CGU. The local AXI, the DMA and the MTL modules also operate with this clock of maximum 250 MHz for full performance. For 2.5 Gbps and 5 Gbps this clock must be 200 MHz or higher.
f_{SRI}	f_{SRI}	AXI2SRI bridges are connected to this free-running clock provided by the clock generation unit (CGU) of maximum 250 MHz for full performance.
clk_tx_i	GETH_TXCLK (port pin)	MII transmit clock Input from PHY (10/100 Mbps) The external PHY or oscillator provides this transmission clock. TXD[3:0] is synchronous to this clock. This is 25 MHz in 100 Mbps mode and 2.5 MHz in 10 Mbps mode. Needs a clock input during SW reset (GETH_DMA_MODE.SWR)

(table continues...)

Table 772 (continued) Clock lines of ethernet MAC

Clock line	Connected to	Description
clk_gref_i	GETH_GREFCLK (port pin)	RGMII transmit clock Reference Input from PHY (1000 Mbps). The external PHY or oscillator provides this clock as reference. TXD[3:0] is not necessarily synchronous to this clock! This is always 125 MHz in 1000/100/10 Mbps/s mode. Needs a clock input during SW reset (GETH_DMA_MODE.SWR)
clk_tx_o	GETH_TXCLK (port pin)	RGMII transmission clock Output to PHY (1000 Mbps) . TXD[3:0] is synchronous (DDR) to this 125 MHz clock to the PHY. If RGMII is downgraded to 100 Mbps it is divided internally to 25 MHz and for 10 Mbps to 2.5 MHz. In each case it is generated from clk_gref_i by respective division. This division is controlled automatically by evaluation of mac_speed_o[1:0] and phy_intf_sel_i[2:0]. TXCLK has the same on chip delay and output pad delay as the TXD signals and TCTL unless clock skew is selected in SKEW.
clk_rx_i	GETH_RXCLK (port pin)	Receive clock from PHY. The external PHY provides this receive clock for RGMII and MII interfaces. This clock is 125 MHz in 1 Gbps mode, 25 MHz in 100 Mbps mode, 2.5 MHz in 10 Mbps mode. Needs a clock input during SW reset (GETH_DMA_MODE.SWR)
clk_rmii_i	GETH_REFCLK (port pin)	50-MHz clock used by the RMII from PHY. In 100 Mbps Mode, it is divided internally to 25 MHz and provided to the internal MII interface. If RMII is downgraded to 10 Mbps it is divided internally to 2.5 MHz. This division is controlled automatically by evaluation of mac_speed_o[1:0] and phy_intf_sel_i[2:0]. Needs a clock input during SW reset (GETH_DMA_MODE.SWR).
clk_ptp_ref_i	f_{GETH}	Reference Clock for the Time Stamp Update Logic of maximum 250 MHz for full performance
sbd_pc_tim_wdw_i[3:0]	ptp_pps_o[1:0][1:0]	The Policing Counters (PC) of the Intrusion Detection System have 4 inputs which are used as Time Window signals (TW). This allows to implement checks in hard ware for a maximum rate of frames. These time window signals are connected to the highly programmable Pulse Per Second (PTP EEE1588) outputs of the MACs. Each MAC has 2 PPS outputs ptp_pps_o[number of MAC] [number of output]. In sum there are 4 PPS signals in a bridge with 2 MACs. These PPS outputs are connected to the Policing Counter system which has 4 time window inputs pc_tw_i[3:0].

Table 773 List of phy interfaces and their clocks

Speed	Interface at MAC	Data line width	Interface at PHY	Interface clock frequency	Clock signal name at PHY
5 Gbps	XGMII	32	UXSGMII	156.25 MHz	No dedicated clock. Clock is always embedded in data.
2.5 Gbps	XGMII	32	SGMII	78.125 MHz	

(table continues...)

Table 773 (continued) List of phy interfaces and their clocks

2.5 Gbps	GMII	8	SGMII	312.5 MHz	
1 Gbps	GMII	8	SGMII	125 MHz	
100 Mbps	MII	4	SGMII	25 MHz	
100 Mbps	MII	4	MII	25 MHz	RXCLK, TXCLK
10 Mbps	MII	4	MII	2.5 MHz	RXCLK, TXCLK
1 Gbps	RGMII	4 (DDR)	RGMII	125 MHz	RXCLK, TXCLK, GREFCLK
100 Mbps	RGMII	4	RGMII	25 MHz	RXCLK, TXCLK, GREFCLK
10 Mbps	RGMII	4	RGMII	2.5 MHz	RXCLK, TXCLK, GREFCLK
100 Mbps	RMII	2	RMII	50 MHz	REFCLK
10 Mbps	RMII	2	RMII	5 MHz	REFCLK

***Note**: Half-duplex is supported in 10/100 MII and RMII modes only. All other modes use full-duplex only.*

***Note**: Clause 37 auto-negotiation is not supported. It is assumed that applications are deterministic and know the speed grade before implementation.*

14.5 Bridge

This chapter gives the overview of features and detailed specifications of the two-port Ethernet Bridge Subsystem.

The Bridge is designed for applications in which an end-station is connected to the Ethernet network using two Ethernet ports. It is capable of terminating traffic from either of the ports to the host through a multichannel DMA interface as well as forwarding traffic between the ports. The latter mainly targets a daisy-chained Ethernet network.

14.5.1 Feature list

This chapter gives the overview of features.

- The bridge connects two XGMACs and one common host interface
- It allows to statically establish the following data paths between these three blocks
- Transmit Ethernet frames on two Ethernet end points (two Ethernet endpoints exist in products with Ethernet Bridge only); the TX data is read from any address and stored in an internal FIFO buffer before transmission
- Receive Ethernet frames from any of the two Ethernet endpoints (two Ethernet endpoints exist in products with Ethernet Bridge only), store them in an internal FIFO and forward them to the other Ethernet endpoint in bridged operation mode or to the application (software side)
- Automotive Safety Features

  The automotive safety features improve the reliability and reduce the time taken to detect faults in the Ethernet Bridge. The Bridge supports the following features for automotive safety:

  - Error correction code (ECC) protection for memories
  - FSM parity and timeout protection
  - Application/CSR interface timeout protection

14.5.2 Functional overview

The block diagram below shows the two-port Ethernet Bridge Subsystem with one host port.

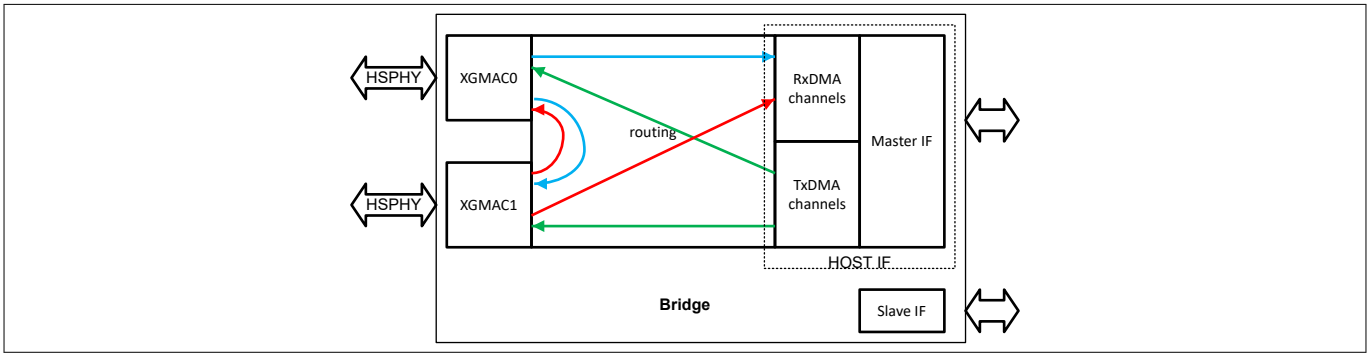


Figure 188 Bridge Overview

Main functionality

The Ethernet Bridge is built with a distributed memory architecture in which ingress/egress buffers are present on each line port. The Ethernet Bridge performs the transfer of packets using a cross-connect concept:

- From ingress to egress port (packet forwarding)
- From ingress to host port
- From host port to egress ports

Main features of the Ethernet Bridge are listed below:

- Each port of the Ethernet Bridge consists of a pair of bidirectional packet transfer interface
- The two Ethernet line ports are on the left side of the Ethernet Bridge as shown in Figure 188. The Ethernet port consist of the MAC and MTL blocks of the XGMAC IP. It has two interfaces to the Ethernet Bridge:
  - ARI-M (master) interface transfers the packets from ingress MAC to the Ethernet Bridge
  - ATI-S (slave) interface accepts the packets from the Ethernet Bridge and transfers to the egress MAC
- The host port is on the right side of the Ethernet Bridge as shown in Figure 188. It connects to the system bus on which the host CPU and host memory reside, through a multi-channel DMA engine. The host port consists of the DMA and master interface blocks of the XGMAC IP. Other than the line ports the host port of the Ethernet Bridge does not have any ingress/egress buffers. The host port will utilize the buffers on the line ports. It has two interfaces to the Ethernet Bridge:
  - ARI-S (slave) interface on the host port accepts the packets from the Ethernet Bridge and transfers to the RX DMA engine which in turn transfers them to the host memory
  - ATI-M (master) interface transfers the packets from the TX DMA engine (host memory) to the Ethernet Bridge

List of sub-blocks

The bridge subsystem is built with XGMAC IP and additional custom blocks

- XGMAC for line and host side ports
- Additional blocks:
  - Arbiter block:
    Prevents contentions at slave side
  - Control status registers
    Allows the application configuration and status read out
  - Clock and reset logic:
    Allows the application to configure the clock and to issue a reset
  - Packet transfer interfaces based on ARI and ATI:
    Allows packet based data transfer inside the bridge
  - Slave Interface:

Provides the application CPU with the ability to access all registers in the various ports of the Bridge through a single slave port

- Ethernet Bridge:
  Provides connection-based classification of traffic between the host port and the ingress/egress ports

14.5.3 Functional description

The following sub-section give more details on the operation modes of the bridge and its configuration. Furthermore the details of forwarding will be described in more detail.

14.5.3.1 Operation mode

The bridge can be configured either in one-port mode or in 2 port mode

In one port mode, only one MAC is active and no bridging available.

In two port mode, two MACs are active and the bridge can either forward frames between ports or send them from ports to the host or from the host to the ports.

One port mode

This is achieved by directly connecting the:

- ARI-M interface of Ethernet line port to ARI-S interface of host (DMA) port
- ATI-S interface of Ethernet line port to ATI-M interface of host (DMA) port

In this mode, the Ethernet line port operates with up to eight TxQ and up to eight RxQs connected to Tx/RxDMA towards the Host port.

You can program the Ethernet Bridge to operate in the one-port MAC mode by programming the Q_CH_MAPPING_EN bit to zero, and select the Ethernet line port through the PORT_SEL field of the Forward_Control register.

Two port mode

The Ethernet Bridge is connecting the two Ethernet ports with the host port.

Within the Ethernet Bridge, there will be three types of simultaneous traffic flows with possible contention in all the three flows at the destination:

- Traffic from the host port (TX DMA) to the egress of the two Ethernet ports. These are shown in Figure 188 as green lines and arrows from ATI-M interface of DMA to ATI-S of Ethernet ports within the Ethernet Bridge
- Traffic from the ingress (RX) interface of each Ethernet port to the host port (RX DMA). These are shown in Figure 188 as blue and red arrows from ARI-M interface of Ethernet ports to ARI-S of the DMA port within the Ethernet Bridge
- Traffic from the ingress port of one Ethernet port to the egress port of the other Ethernet port (forwarding between ports). These are shown in Figure 188 as blue and red arrows from ARI-M interface of Ethernet ingress ports to ATI-S of the Ethernet egress port within the Ethernet Bridge

  In this bridge, the packets from the host port to the Ethernet port (and vice versa), go through the same path as in a 1-port Controller. So, all the features and functions described in the XGMAC Controller hold true with respect to traffic shaping (AVB and TSN), flow control, filtering, and so on

You can program the Ethernet Bridge to operate in the two-port MAC mode by setting the Q_CH_MAPPING_EN bit of the Forward_Control register to 1'b1 and enabling the Ethernet port's TxQ and RxC channel mappings of the two Ethernet ports through the respective Port_i_Control register, TXQ_Map_Port_i register and RXC_Map_Port_i register.

14.5.3.2 Configuration of the bridge

This chapter explains, how queues and DMA channels can be mapped. It gives examples and also reflects the purpose of each configuration.

Naming convention

The following naming convention is used:

- Port: MAC inside bridge

- Host port: interface to the application side containing the DMA
- TxDMA: TxDMA channel in host port
- RxDMA: Rx DMA in host port
- TxQ: Tx Queue in Ethernet line port
- RxQ: Rx Queue in Ethernet line port
  - Inside XGMAC MTL, not used in the bridge for mapping
  - RxQ to DMA channel mapping is done by software configuration of the XGMAC MTL
  - This DMA channel number is used in the bridge as RxC
- RxC: Rx Channel in Ethernet line port
  - Frame filtering or RxQ to DMA mapping results in a DMA number (dynamic mapping)
  - In the bridge, this DMA number is used as RxC number for further bridge mapping
- Bridge mapping
  - Each RxC can be mapped to either a RxDMA of the host port or a TxQ of the other port for forwarding
  - The TxQ can be mapped to either a TxDMA of the host port or an RxC of the other port for forwarding

Configuration

Each MAC (port) has eight TxQs, eight RxCs and eight RxQs. The host port has eight TxDMAs and eight RxDMAs

For full flexibility, use “dynamic mapping” on the Rx path (see XGMAC description). RxDMA number will be based on:

- DA/SA and VLAN register filter engine in the MAC Receiver and/or
- Flexible receive parser engine

This RxDMA number is reused in the bridge configuration and interpreted as RxC number. A mapping table in the Ethernet Bridge allows now to configure two kinds of destination from this RxC: Either to a:

- RxDMA or to a
- TxQ of the other port

On the Tx path, a mapping table can be configured by programming the following registers:

- Port_0_Control Register. The user decides if:
  - TxQi and RxCi are used for forwarding by setting Port_0_Control.FWD_EN
    - TxQi is mapped to an RxC in Port 1, the RxC number is defined in TXQ_Map_Port_0.TXQi
  - Or TxQi and RxCi are each mapped to a TxDMA and RxDMA channel by clearing Port_0_Control.FWD_EN
    - TxQi is mapped to a TxDMA, the TxDMA number is defined in TXQ_Map_Port_0.TXQi
    - RxCi is mapped to an RxDMA, the RxDMA number is defined in RXC_Map_Port_0.RXCi
- Port_1_Control Register. The user decides if:
  - TxQi and RxCi are used for forwarding by setting Port_1_Control.FWD_EN
    - TxQi is mapped to an RxC in Port 0, the RxC number is defined in TXQ_Map_Port_1.TXQi
  - Or TxQi and RxCi are each mapped to a TxDMA and RxDMA channel by clearing Port_1_Control.FWD_EN
    - TxQi is mapped to a TxDMA, the TxDMA number is defined in TXQ_Map_Port_1.TXQi
    - RxCi is mapped to an RxDMA, the RxDMA number is defined in RXC_Map_Port_1.RXCi

***Note**:* *Registers Port_0_Control and Port_1_Control must be programmed after TXQ_Map_Port_0, RXC_Map_Port_0, TXQ_Map_Port_1 and RXC_Map_Port_1!*

TxQ Mapping details

The Ethernet Bridge supports flexible mapping of TxQ:

- TxQ to TxDMA for egress traffic and
- TxQ to RxC for forwarding traffic

The application can configure this through registers.

TxDMA to TxQ static mapping enables virtualization as traffic from TxDMA channel to the TxQ of egress port are isolated from the forwarding traffic between the ports. This way forwarded frames don't need to go to the application interface. AXI bandwidth is efficiently utilized without any backpressure.

RxC Mapping

Similarly, for virtualization purpose, it is recommended to allocate a fixed number of RxDMA channels for the ingress traffic of the two ports. For example RxDMA#0-3 are allocated to Ethernet Port0 RxC#0-3 while RxDMA#4-7 are allocated to Ethernet Port1 RxC#0-3.

For the RxQ to RxDMA mapping, we recommend using dynamic (packet based) mapping to give maximum flexibility. One of the reasons is also that packets from the Ingress MAC are distributed into the RxQs based on "VLAN priority" and not based on destination. Hence packets to be forwarded to other Ethernet port and packets destined for host port can be queued in the same RxQ if they belong to the same traffic class (VLAN tag priority).

***Note**: In single-port mode, the RxQ to RxDMA mapping is not configured in the bridge but it is configured inside the XGMAC, by programming* Port j MTL RxQ DMA Map0 *and* Port j MTL RxQ DMA Map1*.*

The programming of the "destination DMA" numbers in the filter-registers or flexible receive parser should be programmed accordingly, in each of the Ethernet ports, so that host traffic is segregated and goes to its allocated RxDMA channel.

Users can configure in the filters that a specific RxDMA number is assigned to a packet. This number is assigned by the first passing filter. In the bridge configurations, this RxDMA number is reused and interpreted as RxC number. A mapping table in the Ethernet Bridge allows now to configure two kinds of destination from this RxC: Either to a RxDMA or to a TxQ of the other port. This will allow flexibility to do the mapping while retaining common software programming in each Ethernet port.

14.5.3.3 Arbiters in the bridge

Traffic at the destination (egress) ports will have contention between traffic coming from the other ingress ports.

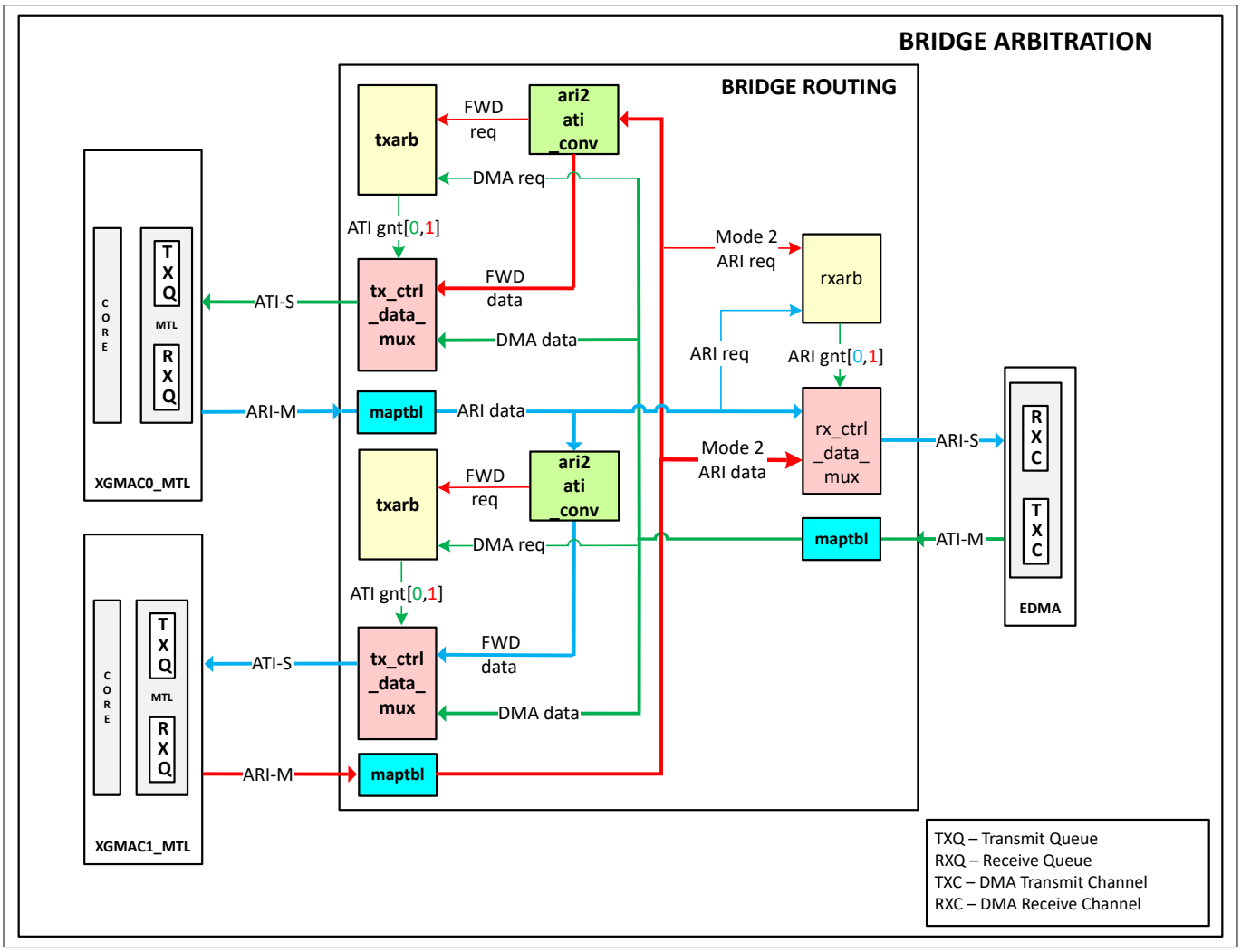


Figure 189 Bridge detailed block diagram

For example

1. ATI-S interface of Ethernet Port 0 can get packets from host port (Tx DMA) and packets forwarded from ARI-M of Ethernet Port 1
2. ATI-S interface of Ethernet Port 1 can get packets from host port (Tx DMA) and packets forwarded from ARI-M of Ethernet Port 0
3. ARI-S interface of host port (Rx DMA) can get packets from ARI-M of Ethernet Port 0 and Port 1

Hence arbiters are required to perform the packet transfers without losing the integrity of the packet.

Arbitration on Ethernet ports: Arbiters for 1. and 2. are simple priority arbiter with traffic from TxDMA (host port) given highest priority (to avoid backpressure on AXI). Arbitration is done every clock so that transactions from host port (AXI) are never blocked. The transactions from the other Ethernet port can be blocked temporarily because the packet is buffered in its RxQ.

Arbitration on host ports: Arbiters for 3. depends on whether RxC are statically allocated between Ethernet Port 1 and 2. If yes, then the arbitration can be Round-Robin between the transactions from the two Ethernet ports. Arbitration cycle is done after every burst transfer initiated from a port is completed.

The specific functionality of the arbiters are as follows:

- In TX path, the bridge is a pass-through with 1-1 connections between the EDMA and corresponding TxQ. The maximum latency that can get added is 1-2 fGETH clock cycles if it gets added to meet any timing requirements. This will not be a factor affecting TBS fetch time
- Port forwarded transfer will have lower priority than host fetched packets. The reason is that port-forwarded packets have ingress buffers while packets fetched from AXI/host does not have buffers and will block the AXI if not serviced immediately. The arbitration on ATI is “clock-cycle” based, that means ATI from EDMA will get priority and ATI from other MAC port will get serviced in free clock cycle windows
- TxQs are statically mapped. That means they are either connected to host/EDMA or connected to port-forwarded traffic from RX of other MAC ports

14.5.3.4 DMA

This section describes how multiple queues are supported in the Transmit and Receive paths in the DMA block. For the architecture of DMA Controller, please refer to chapter XGMAC.

TransmitPath

The bridge supports up to eight Tx channels in the DMA block.

The bridge can be programmed such that the requests of all the TxDMAs are placed and executed on the AXI bus before the data transfer of the first TxDMA is completed and it raises the next request to the arbiter.

Effectively, all the TxDMA channels get equal access to AXI bus and the corresponding TxQs gets filled up at the same rate. The data gets read out from the TxQs as per the Traffic Class scheduler in the MTL. Therefore, at a steady state (assuming that the AXI bandwidth available to fetch data from system memory is always more than the XGMAC operating line rate), the bandwidth for each channel is controlled by the priorities and settings of the Traffic Scheduler.

ReceivePath

The bridge supports up to eight Rx channels in the DMA block.

In the Rx direction, the MTL Rx Controller selects the Rx DMA for which it is transferring or reading the data from the Rx FIFO memory. This scheduling is based on the programming done in the respective MTL_RxQ[x]_Control register.

Each Rx DMA indicates when it is ready to transfer data and the size of the burst-length (number of beats) that it has to transfer. The scheduler checks whether sufficient data (of requested burst length) is available to be transferred to these DMAs and then selects the Rx DMA that gets serviced using the programmed priorities.

Priority scheme for Tx DMA and RxDMA

The DMA arbiter performs the fixed priority arbitration between the Tx and Rx paths of DMA channels for accessing descriptors and data buffers. Descriptor accesses of a Tx or Rx channel is given higher priority when requests from both are active together.

14.6 Registers

14.6.1 Register overview - access mode glossary

Table 774 Register overview - access mode glossary

Keyword	Description
SE	Access protection using PROT register PROTSE.
APU-PCHy (y=0-7)	Protection group consisting of registers ACCENCHy_WRA, ACCENCHy_WRB, ACCENCHy_RDA, ACCENCHy_RDB, ACCENCHy_VM, ACCENCHy_PRS.
PCHy	Access protection using *APU-PCHy* registers.
APU-PG	Protection group consisting of registers ACCENGLOBAL_WRA, ACCENGLOBAL_WRB, ACCENGLOBAL_RDA, ACCENGLOBAL_RDB, ACCENGLOBAL_VM, ACCENGLOBAL_PRS.
PG	Access protection using *APU-PG* registers.
APU-PMy (y=0-1)	Protection group consisting of registers ACCENMACy_WRA, ACCENMACy_WRB, ACCENMACy_RDA, ACCENMACy_RDB, ACCENMACy_VM, ACCENMACy_PRS.
PMy	Access protection using *APU-PMy* registers.
BE	Always returns a Bus Error.
32	Access only when using 32-bit width.
PROT	Access restrictions as defined in the PROT register access rules.

14.6.2 Registers overview - GETH (ascending offset address)

Table 775 Registers overview - GETH (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CLC	Clock control register	00000_H	PG	PG, SV	Application Reset	1642
ID	Module identification register	00008_H	PG	BE	PowerOn Reset	1642
RST_CTRLA	Reset control register A	$0000C_H$	PG	PG, SV	Application Reset	1643
RST_CTRLB	Reset Control Register B	00010_H	PG	PG, SV	Application Reset	1644
RST_STAT	Reset status register	00014_H	PG	BE	Application Reset	1644
MACEN	Clock control register for MAC	00018_H	PG	PG, SV	Application Reset	1645
PROTSE	PROT register safe endinit	00024_H	32	SV, 32, PROT	Application Reset	1646
ACCENGLOBAL_WRA	Write access enable register A	00040_H	32	SE, SV, 32	Application Reset	1647
ACCENGLOBAL_WRB	Write access enable register B	00044_H	32	SE, SV, 32	Application Reset	1648

(table continues...)

Table 775 (continued) Registers overview - GETH (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ACCENGLOBAL_RDA	Read access enable register A	00048_H	32	SE, SV, 32	Application Reset	1648
ACCENGLOBAL_RDB	Read access enable register B	$0004C_H$	32	SE, SV, 32	Application Reset	1649
ACCENGLOBAL_VM	VM access enable register	00050_H	32	SE, SV, 32	Application Reset	1649
ACCENGLOBAL_PRS	PRS access enable register	00054_H	32	SE, SV, 32	Application Reset	1650
ACCENMACy_WRA	MAC y Write access enable register A	00060_H+ y*20_H	32	SE, SV, 32	Application Reset	1651
ACCENMACy_WRB	MAC y Write access enable register B	00064_H+ y*20_H	32	SE, SV, 32	Application Reset	1651
ACCENMACy_RDA	MAC y Read access enable register A	00068_H+ y*20_H	32	SE, SV, 32	Application Reset	1652
ACCENMACy_RDB	MAC y Read access enable register B	$0006C_H$+ y*20_H	32	SE, SV, 32	Application Reset	1652
ACCENMACy_VM	MAC y VM access enable register	00070_H+ y*20_H	32	SE, SV, 32	Application Reset	1653
ACCENMACy_PRS	MAC y PRS access enable register	00074_H+ y*20_H	32	SE, SV, 32	Application Reset	1653
ACCENCHy_WRA	CH y Write access enable register A	$000A0_H$+ y*20_H	32	SE, SV, 32	Application Reset	1654
ACCENCHy_WRB	CH y Write access enable register B	$000A4_H$+ y*20_H	32	SE, SV, 32	Application Reset	1654
ACCENCHy_RDA	CH y Read access enable register A	$000A8_H$+ y*20_H	32	SE, SV, 32	Application Reset	1655
ACCENCHy_RDB	CH y Read access enable register B	$000AC_H$+ y*20_H	32	SE, SV, 32	Application Reset	1655
ACCENCHy_VM	CH y VM access enable register	$000B0_H$+ y*20_H	32	SE, SV, 32	Application Reset	1656
ACCENCHy_PRS	CH y PRS access enable register	$000B4_H$+ y*20_H	32	SE, SV, 32	Application Reset	1657
Portj_MAC_Tx_Configuration	Port j MAC Tx Configuration	10000_H+j *2000_H	PMj	PMj	Kernel Reset	1657
Portj_MAC_Rx_Configuration	Port j MAC Rx Configuration	10004_H+j *2000_H	PMj	PMj	Kernel Reset	1661
Portj_MAC_Packet_Filter	Port j MAC Packet Filter	10008_H+j *2000_H	PMj	PMj	Kernel Reset	1664
Portj_MAC_WD_JB_Timeout	Port j MAC WD JB Timeout	$1000C_H$+ j*2000_H	PMj	PMj	Kernel Reset	1667
Portj_MAC_VLAN_Tag_Ctrl	Port j MAC VLAN Tag Ctrl	10050_H+j *2000_H	PMj	PMj	Kernel Reset	1668

(table continues...)

Table 775 (continued) Registers overview - GETH (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
Portj_MAC_VLAN_Tag_Data	Port j MAC VLAN Tag Data	10054_H+j *2000_H	PMj	PMj	Kernel Reset	1671
Portj_MAC_VLAN_Tag_Filter0	Port j MAC VLAN Tag Filter0	10054_H+j *2000_H	PMj	PMj	Kernel Reset	1672
Portj_MAC_VLAN_Tag_Filter1	Port j MAC VLAN Tag Filter1	10054_H+j *2000_H	PMj	PMj	Kernel Reset	1673
Portj_MAC_VLAN_Tag_Filter2	Port j MAC VLAN Tag Filter2	10054_H+j *2000_H	PMj	PMj	Kernel Reset	1675
Portj_MAC_VLAN_Tag_Filter3	Port j MAC VLAN Tag Filter3	10054_H+j *2000_H	PMj	PMj	Kernel Reset	1676
Portj_MAC_VLAN_Tag_Filter4	Port j MAC VLAN Tag Filter4	10054_H+j *2000_H	PMj	PMj	Kernel Reset	1678
Portj_MAC_VLAN_Tag_Filter5	Port j MAC VLAN Tag Filter5	10054_H+j *2000_H	PMj	PMj	Kernel Reset	1679
Portj_MAC_VLAN_Tag_Filter6	Port j MAC VLAN Tag Filter6	10054_H+j *2000_H	PMj	PMj	Kernel Reset	1681
Portj_MAC_VLAN_Tag_Filter7	Port j MAC VLAN Tag Filter7	10054_H+j *2000_H	PMj	PMj	Kernel Reset	1682
Portj_MAC_VLAN_Hash_Table	Port j MAC VLAN Hash Table	10058_H+j *2000_H	PMj	PMj	Kernel Reset	1684
Portj_MAC_VLAN_Incl	Port j MAC VLAN Incl	10060_H+j *2000_H	PMj	PMj	Kernel Reset	1684
Portj_MAC_VLAN_Incl0	Port j MAC VLAN Incl0	10060_H+j *2000_H	PMj	PMj	Kernel Reset	1686
Portj_MAC_VLAN_Incl1	Port j MAC VLAN Incl1	10060_H+j *2000_H	PMj	PMj	Kernel Reset	1687
Portj_MAC_VLAN_Incl2	Port j MAC VLAN Incl2	10060_H+j *2000_H	PMj	PMj	Kernel Reset	1688
Portj_MAC_VLAN_Incl3	Port j MAC VLAN Incl3	10060_H+j *2000_H	PMj	PMj	Kernel Reset	1689
Portj_MAC_VLAN_Incl4	Port j MAC VLAN Incl4	10060_H+j *2000_H	PMj	PMj	Kernel Reset	1689
Portj_MAC_VLAN_Incl5	Port j MAC VLAN Incl5	10060_H+j *2000_H	PMj	PMj	Kernel Reset	1690
Portj_MAC_VLAN_Incl6	Port j MAC VLAN Incl6	10060_H+j *2000_H	PMj	PMj	Kernel Reset	1691
Portj_MAC_VLAN_Incl7	Port j MAC VLAN Incl7	10060_H+j *2000_H	PMj	PMj	Kernel Reset	1692
Portj_MAC_Inner_VLAN_Incl	Port j MAC Inner VLAN Incl	10064_H+j *2000_H	PMj	PMj	Kernel Reset	1692
Portj_MAC_Rx_Eth_Type_Match	Port j MAC Rx Eth Type Match	$1006C_H$+ j*2000_H	PMj	PMj	Kernel Reset	1694

(table continues...)

Table 775 (continued) Registers overview - GETH (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
Portj_MAC_Q0_Tx_Flow_Ctrl	Port j MAC Q0 Tx Flow Ctrl	10070_H+j *2000_H	PMj	PMj	Kernel Reset	1694
Portj_MAC_Rx_Flow_Ctrl	Port j MAC Rx Flow Ctrl	10090_H+j *2000_H	PMj	PMj	Kernel Reset	1696
Portj_MAC_RxQ_Ctrl4	Port j MAC RxQ Ctrl4	10094_H+j *2000_H	PMj	PMj	Kernel Reset	1697
Portj_MAC_RxQ_Ctrl5	Port j MAC RxQ Ctrl5	10098_H+j *2000_H	PMj	PMj	Kernel Reset	1699
Portj_MAC_RxQ_Ctrl0	Port j MAC RxQ Ctrl0	$100A0_H$+ j*2000_H	PMj	PMj	Kernel Reset	1700
Portj_MAC_RxQ_Ctrl1	Port j MAC RxQ Ctrl1	$100A4_H$+ j*2000_H	PMj	PMj	Kernel Reset	1701
Portj_MAC_RxQ_Ctrl2	Port j MAC RxQ Ctrl2	$100A8_H$+ j*2000_H	PMj	PMj	Kernel Reset	1703
Portj_MAC_RxQ_Ctrl3	Port j MAC RxQ Ctrl3	$100AC_H$+ j*2000_H	PMj	PMj	Kernel Reset	1704
Portj_MAC_Interrupt_Status	Port j MAC Interrupt Status	$100B0_H$+ j*2000_H	PMj	PMj	Kernel Reset	1705
Portj_MAC_Interrupt_Enable	Port j MAC Interrupt Enable	$100B4_H$+ j*2000_H	PMj	PMj	Kernel Reset	1709
Portj_MAC_Rx_Tx_Status	Port j MAC Rx Tx Status	$100B8_H$+ j*2000_H	PMj	PMj	Kernel Reset	1710
Portj_MAC_PMT_Control_Status	Port j MAC PMT Control Status	$100C0_H$+ j*2000_H	PMj	PMj	Kernel Reset	1713
Portj_MAC_RWK_Packet_Filter	Port j MAC RWK Packet Filter	$100C4_H$+ j*2000_H	PMj	PMj	Kernel Reset	1715
Portj_RWK_Filter0_1_CRC	Port j RWK Filter0 1 CRC	$100C4_H$+ j*2000_H	PMj	PMj	Kernel Reset	1716
Portj_RWK_Filter0_3_Command	Port j RWK Filter0 3 Command	$100C4_H$+ j*2000_H	PMj	PMj	Kernel Reset	1716
Portj_RWK_Filter0_3_Offset	Port j RWK Filter0 3 Offset	$100C4_H$+ j*2000_H	PMj	PMj	Kernel Reset	1718
Portj_RWK_Filter0_Byte_Mask	Port j RWK Filter0 Byte Mask	$100C4_H$+ j*2000_H	PMj	PMj	Kernel Reset	1719
Portj_RWK_Filter1_Byte_Mask	Port j RWK Filter1 Byte Mask	$100C4_H$+ j*2000_H	PMj	PMj	Kernel Reset	1720
Portj_RWK_Filter2_3_CRC	Port j RWK Filter2 3 CRC	$100C4_H$+ j*2000_H	PMj	PMj	Kernel Reset	1720
Portj_RWK_Filter2_Byte_Mask	Port j RWK Filter2 Byte Mask	$100C4_H$+ j*2000_H	PMj	PMj	Kernel Reset	1721
Portj_RWK_Filter3_Byte_Mask	Port j RWK Filter3 Byte Mask	$100C4_H$+ j*2000_H	PMj	PMj	Kernel Reset	1722

(table continues...)

Table 775 (continued) Registers overview - GETH (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
Portj_MAC_LPI_Control_Status	Port j MAC LPI Control Status	$100D0_H$+ j*2000_H	PMj	PMj	Kernel Reset	1722
Portj_MAC_LPI_Timers_Control	Port j MAC LPI Timers Control	$100D4_H$+ j*2000_H	PMj	PMj	Kernel Reset	1725
Portj_MAC_LPI_Auto_Entry_Timer	Port j MAC LPI Auto Entry Timer	$100D8_H$+ j*2000_H	PMj	PMj	Kernel Reset	1725
Portj_MAC_1US_Tic_Counter	Port j MAC 1US Tic Counter	$100DC_H$+ j*2000_H	PMj	PMj	Kernel Reset	1726
Portj_MAC_Version	Port j MAC Version	10110_H+j *2000_H	PMj	PMj	Kernel Reset	1726
Portj_MAC_Debug	Port j MAC Debug	10114_H+j *2000_H	PMj	PMj	Kernel Reset	1727
Portj_MAC_HW_Feature0	Port j MAC HW Feature0	$1011C_H$+ j*2000_H	PMj	PMj	Kernel Reset	1728
Portj_MAC_HW_Feature1	Port j MAC HW Feature1	10120_H+j *2000_H	PMj	PMj	Kernel Reset	1730
Portj_MAC_HW_Feature2	Port j MAC HW Feature2	10124_H+j *2000_H	PMj	PMj	Kernel Reset	1733
Portj_MAC_HW_Feature3	Port j MAC HW Feature3	10128_H+j *2000_H	PMj	PMj	Kernel Reset	1734
Portj_MAC_HW_Feature4	Port j MAC HW Feature4	$1012C_H$+ j*2000_H	PMj	PMj	Kernel Reset	1738
Portj_MAC_Extended_Configuration	Port j MAC Extended Configuration	10140_H+j *2000_H	PMj	PMj	Kernel Reset	1738
Portj_MAC_Ext_Cfg1	Port j MAC Ext Cfg1	10144_H+j *2000_H	PMj	PMj	Kernel Reset	1741
Portj_MAC_DPP_FSM_Interrupt_Status	Port j MAC DPP FSM Interrupt Status	10150_H+j *2000_H	PMj	PMj	Kernel Reset	1742
Portj_MAC_AXI_SLV_DPE_Addr_Status	Port j MAC AXI SLV DPE Addr Status	10154_H+j *2000_H	PMj	PMj	Kernel Reset	1746
Portj_MAC_FSM_Control	Port j MAC FSM Control	10158_H+j *2000_H	PMj	PMj	Kernel Reset	1747
Portj_MAC_FSM_ACT_Timer	Port j MAC FSM ACT Timer	$1015C_H$+ j*2000_H	PMj	PMj	Kernel Reset	1749
Portj_SNPS_SCS_REG1	Port j SNPS SCS REG1	10160_H+j *2000_H	PMj	PMj	Kernel Reset	1751
Portj_MAC_SCSR_Control	Port j MAC SCSR Control	10164_H+j *2000_H	PMj	PMj	Kernel Reset	1751

(table continues...)

Table 775 (continued) Registers overview - GETH (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
Portj_MAC_SCSR_Parity_Status	Port j MAC SCSR Parity Status	10168_H+j *2000_H	PMj	PMj	Kernel Reset	1752
Portj_MAC_SCSR_Parity_Err_Count	Port j MAC SCSR Parity Err Count	$1016C_H$+ j*2000_H	PMj	PMj	Kernel Reset	1753
Portj_MAC_Flex_Hdr_Cfg	Port j MAC Flex Hdr Cfg	10194_H+j *2000_H	PMj	PMj	Kernel Reset	1753
Portj_MAC_Flex_Hdr_Low	Port j MAC Flex Hdr Low	10198_H+j *2000_H	PMj	PMj	Kernel Reset	1754
Portj_MAC_Flex_Hdr_High	Port j MAC Flex Hdr High	$1019C_H$+ j*2000_H	PMj	PMj	Kernel Reset	1755
Portj_MDIO_Single_Command_Address	Port j MDIO Single Command Address	10200_H+j *2000_H	PMj	PMj	Kernel Reset	1755
Portj_MDIO_Single_Command_Control_Data	Port j MDIO Single Command Control Data	10204_H+j *2000_H	PMj	PMj	Kernel Reset	1756
Portj_MDIO_Continuous_Write_Address	Port j MDIO Continuous Write Address	10208_H+j *2000_H	PMj	PMj	Kernel Reset	1758
Portj_MDIO_Continuous_Write_Data	Port j MDIO Continuous Write Data	$1020C_H$+ j*2000_H	PMj	PMj	Kernel Reset	1759
Portj_MDIO_Continuous_Scan_Port_Enable	Port j MDIO Continuous Scan Port Enable	10210_H+j *2000_H	PMj	PMj	Kernel Reset	1760
Portj_MDIO_Interrupt_Status	Port j MDIO Interrupt Status	10214_H+j *2000_H	PMj	PMj	Kernel Reset	1762
Portj_MDIO_Interrupt_Enable	Port j MDIO Interrupt Enable	10218_H+j *2000_H	PMj	PMj	Kernel Reset	1765
Portj_MDIO_Port_Connect_Disconnect_Status	Port j MDIO Port Connect Disconnect Status	$1021C_H$+ j*2000_H	PMj	PMj	Kernel Reset	1767
Portj_MDIO_Clause_22_Port	Port j MDIO Clause 22 Port	10220_H+j *2000_H	PMj	PMj	Kernel Reset	1770
Portj_MDIO_Port_Nx4_Indirect_Control	Port j MDIO Port Nx4 Indirect Control	10224_H+j *2000_H	PMj	PMj	Kernel Reset	1772
Portj_MDIO_PortNx4P0_Device_In_Use	Port j MDIO PortNx4P0 Device In Use	10230_H+j *2000_H	PMj	PMj	Kernel Reset	1773
Portj_MDIO_PortNx4P0_Link_Status	Port j MDIO PortNx4P0 Link Status	10234_H+j *2000_H	PMj	PMj	Kernel Reset	1775

(table continues...)

Table 775 (continued) Registers overview - GETH (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
Portj_MDIO_PortNx4P0_Alive_Status	Port j MDIO PortNx4P0 Alive Status	10238_H+j *2000_H	PMj	PMj	Kernel Reset	1776
Portj_MDIO_PortNx4P1_Device_In_Use	Port j MDIO PortNx4P1 Device In Use	10240_H+j *2000_H	PMj	PMj	Kernel Reset	1777
Portj_MDIO_PortNx4P1_Link_Status	Port j MDIO PortNx4P1 Link Status	10244_H+j *2000_H	PMj	PMj	Kernel Reset	1778
Portj_MDIO_PortNx4P1_Alive_Status	Port j MDIO PortNx4P1 Alive Status	10248_H+j *2000_H	PMj	PMj	Kernel Reset	1780
Portj_MDIO_PortNx4P2_Device_In_Use	Port j MDIO PortNx4P2 Device In Use	10250_H+j *2000_H	PMj	PMj	Kernel Reset	1781
Portj_MDIO_PortNx4P2_Link_Status	Port j MDIO PortNx4P2 Link Status	10254_H+j *2000_H	PMj	PMj	Kernel Reset	1782
Portj_MDIO_PortNx4P2_Alive_Status	Port j MDIO PortNx4P2 Alive Status	10258_H+j *2000_H	PMj	PMj	Kernel Reset	1783
Portj_MDIO_PortNx4P3_Device_In_Use	Port j MDIO PortNx4P3 Device In Use	10260_H+j *2000_H	PMj	PMj	Kernel Reset	1785
Portj_MDIO_PortNx4P3_Link_Status	Port j MDIO PortNx4P3 Link Status	10264_H+j *2000_H	PMj	PMj	Kernel Reset	1786
Portj_MDIO_PortNx4P3_Alive_Status	Port j MDIO PortNx4P3 Alive Status	10268_H+j *2000_H	PMj	PMj	Kernel Reset	1787
Portj_MAC_FPE_CTRL_STS	Port j MAC FPE CTRL STS	10280_H+j *2000_H	PMj	PMj	Kernel Reset	1788
Portj_MAC_CSR_SW_Ctrl	Port j MAC CSR SW Ctrl	10290_H+j *2000_H	PMj	PMj	Kernel Reset	1790
Portj_MAC_Address0_High	Port j MAC Address0 High	10300_H+j *2000_H	PMj	PMj	Kernel Reset	1791
Portj_MAC_Address0_Low	Port j MAC Address0 Low	10304_H+j *2000_H	PMj	PMj	Kernel Reset	1792
Portj_MAC_Address1_High	Port j MAC Address1 High	10308_H+j *2000_H	PMj	PMj	Kernel Reset	1793
Portj_MAC_Address1_Low	Port j MAC Address1 Low	$1030C_H$+ j*2000_H	PMj	PMj	Kernel Reset	1794
Portj_MAC_Address2_High	Port j MAC Address2 High	10310_H+j *2000_H	PMj	PMj	Kernel Reset	1795

(table continues...)

Table 775 (continued) Registers overview - GETH (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
Portj_MAC_Address2_Low	Port j MAC Address2 Low	10314_H+j *2000_H	PMj	PMj	Kernel Reset	1796
Portj_MAC_Address3_High	Port j MAC Address3 High	10318_H+j *2000_H	PMj	PMj	Kernel Reset	1797
Portj_MAC_Address3_Low	Port j MAC Address3 Low	$1031C_H$+ j*2000_H	PMj	PMj	Kernel Reset	1798
Portj_MAC_Address4_High	Port j MAC Address4 High	10320_H+j *2000_H	PMj	PMj	Kernel Reset	1799
Portj_MAC_Address4_Low	Port j MAC Address4 Low	10324_H+j *2000_H	PMj	PMj	Kernel Reset	1800
Portj_MAC_Address5_High	Port j MAC Address5 High	10328_H+j *2000_H	PMj	PMj	Kernel Reset	1801
Portj_MAC_Address5_Low	Port j MAC Address5 Low	$1032C_H$+ j*2000_H	PMj	PMj	Kernel Reset	1802
Portj_MAC_Address6_High	Port j MAC Address6 High	10330_H+j *2000_H	PMj	PMj	Kernel Reset	1803
Portj_MAC_Address6_Low	Port j MAC Address6 Low	10334_H+j *2000_H	PMj	PMj	Kernel Reset	1804
Portj_MAC_Address7_High	Port j MAC Address7 High	10338_H+j *2000_H	PMj	PMj	Kernel Reset	1805
Portj_MAC_Address7_Low	Port j MAC Address7 Low	$1033C_H$+ j*2000_H	PMj	PMj	Kernel Reset	1806
Portj_MAC_Address8_High	Port j MAC Address8 High	10340_H+j *2000_H	PMj	PMj	Kernel Reset	1807
Portj_MAC_Address8_Low	Port j MAC Address8 Low	10344_H+j *2000_H	PMj	PMj	Kernel Reset	1808
Portj_MAC_Address9_High	Port j MAC Address9 High	10348_H+j *2000_H	PMj	PMj	Kernel Reset	1809
Portj_MAC_Address9_Low	Port j MAC Address9 Low	$1034C_H$+ j*2000_H	PMj	PMj	Kernel Reset	1810
Portj_MAC_Address10_High	Port j MAC Address10 High	10350_H+j *2000_H	PMj	PMj	Kernel Reset	1811
Portj_MAC_Address10_Low	Port j MAC Address10 Low	10354_H+j *2000_H	PMj	PMj	Kernel Reset	1812
Portj_MAC_Address11_High	Port j MAC Address11 High	10358_H+j *2000_H	PMj	PMj	Kernel Reset	1813
Portj_MAC_Address11_Low	Port j MAC Address11 Low	$1035C_H$+ j*2000_H	PMj	PMj	Kernel Reset	1814
Portj_MAC_Address12_High	Port j MAC Address12 High	10360_H+j *2000_H	PMj	PMj	Kernel Reset	1815
Portj_MAC_Address12_Low	Port j MAC Address12 Low	10364_H+j *2000_H	PMj	PMj	Kernel Reset	1816

(table continues...)

Table 775 (continued) Registers overview - GETH (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
Portj_MAC_Address13_High	Port j MAC Address13 High	10368_H+j *2000_H	PMj	PMj	Kernel Reset	1817
Portj_MAC_Address13_Low	Port j MAC Address13 Low	$1036C_H$+ j*2000_H	PMj	PMj	Kernel Reset	1818
Portj_MAC_Address14_High	Port j MAC Address14 High	10370_H+j *2000_H	PMj	PMj	Kernel Reset	1819
Portj_MAC_Address14_Low	Port j MAC Address14 Low	10374_H+j *2000_H	PMj	PMj	Kernel Reset	1820
Portj_MAC_Address15_High	Port j MAC Address15 High	10378_H+j *2000_H	PMj	PMj	Kernel Reset	1821
Portj_MAC_Address15_Low	Port j MAC Address15 Low	$1037C_H$+ j*2000_H	PMj	PMj	Kernel Reset	1822
Portj_MAC_Address16_High	Port j MAC Address16 High	10380_H+j *2000_H	PMj	PMj	Kernel Reset	1823
Portj_MAC_Address16_Low	Port j MAC Address16 Low	10384_H+j *2000_H	PMj	PMj	Kernel Reset	1824
Portj_MAC_Address17_High	Port j MAC Address17 High	10388_H+j *2000_H	PMj	PMj	Kernel Reset	1825
Portj_MAC_Address17_Low	Port j MAC Address17 Low	$1038C_H$+ j*2000_H	PMj	PMj	Kernel Reset	1826
Portj_MAC_Address18_High	Port j MAC Address18 High	10390_H+j *2000_H	PMj	PMj	Kernel Reset	1827
Portj_MAC_Address18_Low	Port j MAC Address18 Low	10394_H+j *2000_H	PMj	PMj	Kernel Reset	1828
Portj_MAC_Address19_High	Port j MAC Address19 High	10398_H+j *2000_H	PMj	PMj	Kernel Reset	1829
Portj_MAC_Address19_Low	Port j MAC Address19 Low	$1039C_H$+ j*2000_H	PMj	PMj	Kernel Reset	1830
Portj_MAC_Address20_High	Port j MAC Address20 High	$103A0_H$+ j*2000_H	PMj	PMj	Kernel Reset	1831
Portj_MAC_Address20_Low	Port j MAC Address20 Low	$103A4_H$+ j*2000_H	PMj	PMj	Kernel Reset	1832
Portj_MAC_Address21_High	Port j MAC Address21 High	$103A8_H$+ j*2000_H	PMj	PMj	Kernel Reset	1833
Portj_MAC_Address21_Low	Port j MAC Address21 Low	$103AC_H$+ j*2000_H	PMj	PMj	Kernel Reset	1834
Portj_MAC_Address22_High	Port j MAC Address22 High	$103B0_H$+ j*2000_H	PMj	PMj	Kernel Reset	1835
Portj_MAC_Address22_Low	Port j MAC Address22 Low	$103B4_H$+ j*2000_H	PMj	PMj	Kernel Reset	1836
Portj_MAC_Address23_High	Port j MAC Address23 High	$103B8_H$+ j*2000_H	PMj	PMj	Kernel Reset	1837

(table continues...)

Table 775 (continued) Registers overview - GETH (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
Portj_MAC_Address23_Low	Port j MAC Address23 Low	$103BC_H$+ j*2000_H	PMj	PMj	Kernel Reset	1838
Portj_MAC_Address24_High	Port j MAC Address24 High	$103C0_H$+ j*2000_H	PMj	PMj	Kernel Reset	1839
Portj_MAC_Address24_Low	Port j MAC Address24 Low	$103C4_H$+ j*2000_H	PMj	PMj	Kernel Reset	1840
Portj_MAC_Address25_High	Port j MAC Address25 High	$103C8_H$+ j*2000_H	PMj	PMj	Kernel Reset	1841
Portj_MAC_Address25_Low	Port j MAC Address25 Low	$103CC_H$+ j*2000_H	PMj	PMj	Kernel Reset	1842
Portj_MAC_Address26_High	Port j MAC Address26 High	$103D0_H$+ j*2000_H	PMj	PMj	Kernel Reset	1843
Portj_MAC_Address26_Low	Port j MAC Address26 Low	$103D4_H$+ j*2000_H	PMj	PMj	Kernel Reset	1844
Portj_MAC_Address27_High	Port j MAC Address27 High	$103D8_H$+ j*2000_H	PMj	PMj	Kernel Reset	1845
Portj_MAC_Address27_Low	Port j MAC Address27 Low	$103DC_H$+ j*2000_H	PMj	PMj	Kernel Reset	1846
Portj_MAC_Address28_High	Port j MAC Address28 High	$103E0_H$+j *2000_H	PMj	PMj	Kernel Reset	1847
Portj_MAC_Address28_Low	Port j MAC Address28 Low	$103E4_H$+j *2000_H	PMj	PMj	Kernel Reset	1848
Portj_MAC_Address29_High	Port j MAC Address29 High	$103E8_H$+j *2000_H	PMj	PMj	Kernel Reset	1849
Portj_MAC_Address29_Low	Port j MAC Address29 Low	$103EC_H$+ j*2000_H	PMj	PMj	Kernel Reset	1850
Portj_MAC_Address30_High	Port j MAC Address30 High	$103F0_H$+j *2000_H	PMj	PMj	Kernel Reset	1851
Portj_MAC_Address30_Low	Port j MAC Address30 Low	$103F4_H$+j *2000_H	PMj	PMj	Kernel Reset	1852
Portj_MAC_Address31_High	Port j MAC Address31 High	$103F8_H$+j *2000_H	PMj	PMj	Kernel Reset	1853
Portj_MAC_Address31_Low	Port j MAC Address31 Low	$103FC_H$+ j*2000_H	PMj	PMj	Kernel Reset	1854
Portj_MAC_Indir_Access_Ctrl	Port j MAC Indir Access Ctrl	10700_H+j *2000_H	PMj	PMj	Kernel Reset	1855
Portj_MAC_Indir_Access_Data	Port j MAC Indir Access Data	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1857
Portj_MAC_DChSel_IndReg0	Port j MAC DChSel IndReg0	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1857
Portj_MAC_DChSel_IndReg1	Port j MAC DChSel IndReg1	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1857

(table continues...)

Table 775 (continued) Registers overview - GETH (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
Portj_MAC_DChSel_IndReg10	Port j MAC DChSel IndReg10	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1858
Portj_MAC_DChSel_IndReg11	Port j MAC DChSel IndReg11	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1858
Portj_MAC_DChSel_IndReg12	Port j MAC DChSel IndReg12	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1859
Portj_MAC_DChSel_IndReg13	Port j MAC DChSel IndReg13	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1859
Portj_MAC_DChSel_IndReg14	Port j MAC DChSel IndReg14	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1860
Portj_MAC_DChSel_IndReg15	Port j MAC DChSel IndReg15	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1860
Portj_MAC_DChSel_IndReg16	Port j MAC DChSel IndReg16	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1861
Portj_MAC_DChSel_IndReg17	Port j MAC DChSel IndReg17	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1861
Portj_MAC_DChSel_IndReg18	Port j MAC DChSel IndReg18	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1862
Portj_MAC_DChSel_IndReg19	Port j MAC DChSel IndReg19	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1862
Portj_MAC_DChSel_IndReg2	Port j MAC DChSel IndReg2	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1863
Portj_MAC_DChSel_IndReg20	Port j MAC DChSel IndReg20	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1863
Portj_MAC_DChSel_IndReg21	Port j MAC DChSel IndReg21	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1864
Portj_MAC_DChSel_IndReg22	Port j MAC DChSel IndReg22	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1864
Portj_MAC_DChSel_IndReg23	Port j MAC DChSel IndReg23	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1865
Portj_MAC_DChSel_IndReg24	Port j MAC DChSel IndReg24	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1865
Portj_MAC_DChSel_IndReg25	Port j MAC DChSel IndReg25	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1866
Portj_MAC_DChSel_IndReg26	Port j MAC DChSel IndReg26	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1866
Portj_MAC_DChSel_IndReg27	Port j MAC DChSel IndReg27	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1867
Portj_MAC_DChSel_IndReg28	Port j MAC DChSel IndReg28	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1867
Portj_MAC_DChSel_IndReg29	Port j MAC DChSel IndReg29	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1868

(table continues...)

Table 775 (continued) Registers overview - GETH (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
Portj_MAC_DChSel_IndReg3	Port j MAC DChSel IndReg3	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1868
Portj_MAC_DChSel_IndReg30	Port j MAC DChSel IndReg30	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1869
Portj_MAC_DChSel_IndReg31	Port j MAC DChSel IndReg31	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1869
Portj_MAC_DChSel_IndReg4	Port j MAC DChSel IndReg4	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1870
Portj_MAC_DChSel_IndReg5	Port j MAC DChSel IndReg5	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1870
Portj_MAC_DChSel_IndReg6	Port j MAC DChSel IndReg6	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1871
Portj_MAC_DChSel_IndReg7	Port j MAC DChSel IndReg7	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1871
Portj_MAC_DChSel_IndReg8	Port j MAC DChSel IndReg8	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1872
Portj_MAC_DChSel_IndReg9	Port j MAC DChSel IndReg9	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1872
Portj_MAC_DPCSel_IndReg0	Port j MAC DPCSel IndReg0	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1873
Portj_MAC_DPCSel_IndReg1	Port j MAC DPCSel IndReg1	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1873
Portj_MAC_DPCSel_IndReg10	Port j MAC DPCSel IndReg10	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1874
Portj_MAC_DPCSel_IndReg11	Port j MAC DPCSel IndReg11	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1875
Portj_MAC_DPCSel_IndReg12	Port j MAC DPCSel IndReg12	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1875
Portj_MAC_DPCSel_IndReg13	Port j MAC DPCSel IndReg13	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1876
Portj_MAC_DPCSel_IndReg14	Port j MAC DPCSel IndReg14	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1876
Portj_MAC_DPCSel_IndReg15	Port j MAC DPCSel IndReg15	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1877
Portj_MAC_DPCSel_IndReg16	Port j MAC DPCSel IndReg16	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1877
Portj_MAC_DPCSel_IndReg17	Port j MAC DPCSel IndReg17	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1878
Portj_MAC_DPCSel_IndReg18	Port j MAC DPCSel IndReg18	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1879
Portj_MAC_DPCSel_IndReg19	Port j MAC DPCSel IndReg19	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1879

(table continues...)

Table 775 (continued) Registers overview - GETH (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
Portj_MAC_DPCSel_IndReg2	Port j MAC DPCSel IndReg2	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1880
Portj_MAC_DPCSel_IndReg20	Port j MAC DPCSel IndReg20	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1880
Portj_MAC_DPCSel_IndReg21	Port j MAC DPCSel IndReg21	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1881
Portj_MAC_DPCSel_IndReg22	Port j MAC DPCSel IndReg22	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1881
Portj_MAC_DPCSel_IndReg23	Port j MAC DPCSel IndReg23	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1882
Portj_MAC_DPCSel_IndReg24	Port j MAC DPCSel IndReg24	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1883
Portj_MAC_DPCSel_IndReg25	Port j MAC DPCSel IndReg25	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1883
Portj_MAC_DPCSel_IndReg26	Port j MAC DPCSel IndReg26	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1884
Portj_MAC_DPCSel_IndReg27	Port j MAC DPCSel IndReg27	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1884
Portj_MAC_DPCSel_IndReg28	Port j MAC DPCSel IndReg28	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1885
Portj_MAC_DPCSel_IndReg29	Port j MAC DPCSel IndReg29	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1885
Portj_MAC_DPCSel_IndReg3	Port j MAC DPCSel IndReg3	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1886
Portj_MAC_DPCSel_IndReg30	Port j MAC DPCSel IndReg30	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1887
Portj_MAC_DPCSel_IndReg31	Port j MAC DPCSel IndReg31	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1887
Portj_MAC_DPCSel_IndReg4	Port j MAC DPCSel IndReg4	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1888
Portj_MAC_DPCSel_IndReg5	Port j MAC DPCSel IndReg5	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1888
Portj_MAC_DPCSel_IndReg6	Port j MAC DPCSel IndReg6	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1889
Portj_MAC_DPCSel_IndReg7	Port j MAC DPCSel IndReg7	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1889
Portj_MAC_DPCSel_IndReg8	Port j MAC DPCSel IndReg8	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1890
Portj_MAC_DPCSel_IndReg9	Port j MAC DPCSel IndReg9	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1891
Portj_MAC_FPCSel_IndReg0	Port j MAC FPCSel IndReg0	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1891

(table continues...)

Table 775 (continued) Registers overview - GETH (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
Portj_MAC_LPCSel_IndReg0	Port j MAC LPCSel IndReg0	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1892
Portj_MAC_LPCSel_IndReg1	Port j MAC LPCSel IndReg1	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1892
Portj_MAC_LPCSel_IndReg2	Port j MAC LPCSel IndReg2	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1893
Portj_MAC_LPCSel_IndReg3	Port j MAC LPCSel IndReg3	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1893
Portj_MAC_LPCSel_IndReg4	Port j MAC LPCSel IndReg4	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1894
Portj_MAC_LPCSel_IndReg5	Port j MAC LPCSel IndReg5	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1895
Portj_MAC_LPCSel_IndReg6	Port j MAC LPCSel IndReg6	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1895
Portj_MAC_LPCSel_IndReg7	Port j MAC LPCSel IndReg7	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1896
Portj_MAC_PCCtrl_IndReg0	Port j MAC PCCtrl IndReg0	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1896
Portj_MAC_PCCtrl_IndReg1	Port j MAC PCCtrl IndReg1	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1897
Portj_MAC_PCCtrl_IndReg10	Port j MAC PCCtrl IndReg10	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1899
Portj_MAC_PCCtrl_IndReg11	Port j MAC PCCtrl IndReg11	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1900
Portj_MAC_PCCtrl_IndReg12	Port j MAC PCCtrl IndReg12	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1901
Portj_MAC_PCCtrl_IndReg13	Port j MAC PCCtrl IndReg13	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1902
Portj_MAC_PCCtrl_IndReg14	Port j MAC PCCtrl IndReg14	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1903
Portj_MAC_PCCtrl_IndReg15	Port j MAC PCCtrl IndReg15	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1905
Portj_MAC_PCCtrl_IndReg2	Port j MAC PCCtrl IndReg2	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1906
Portj_MAC_PCCtrl_IndReg3	Port j MAC PCCtrl IndReg3	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1907
Portj_MAC_PCCtrl_IndReg4	Port j MAC PCCtrl IndReg4	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1908
Portj_MAC_PCCtrl_IndReg5	Port j MAC PCCtrl IndReg5	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1909
Portj_MAC_PCCtrl_IndReg6	Port j MAC PCCtrl IndReg6	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1911

(table continues...)

Table 775 (continued) Registers overview - GETH (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
Portj_MAC_PCCtrl_IndReg7	Port j MAC PCCtrl IndReg7	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1912
Portj_MAC_PCCtrl_IndReg8	Port j MAC PCCtrl IndReg8	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1913
Portj_MAC_PCCtrl_IndReg9	Port j MAC PCCtrl IndReg9	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1914
Portj_MAC_PCStatus_DA_IndReg	Port j MAC PCStatus DA IndReg	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1915
Portj_MAC_PCStatus_Filter_Event_IndReg	Port j MAC PCStatus Filter Event IndReg	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1916
Portj_MAC_PCStatus_L3L4_IndReg	Port j MAC PCStatus L3L4 IndReg	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1916
Portj_MAC_PCStatus_VLAN_IndReg	Port j MAC PCStatus VLAN IndReg	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1917
Portj_MAC_PCntr_IndReg0	Port j MAC PCntr IndReg0	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1917
Portj_MAC_PCntr_IndReg1	Port j MAC PCntr IndReg1	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1918
Portj_MAC_PCntr_IndReg10	Port j MAC PCntr IndReg10	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1919
Portj_MAC_PCntr_IndReg11	Port j MAC PCntr IndReg11	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1919
Portj_MAC_PCntr_IndReg12	Port j MAC PCntr IndReg12	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1920
Portj_MAC_PCntr_IndReg13	Port j MAC PCntr IndReg13	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1920
Portj_MAC_PCntr_IndReg14	Port j MAC PCntr IndReg14	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1921
Portj_MAC_PCntr_IndReg15	Port j MAC PCntr IndReg15	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1922
Portj_MAC_PCntr_IndReg2	Port j MAC PCntr IndReg2	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1922
Portj_MAC_PCntr_IndReg3	Port j MAC PCntr IndReg3	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1923
Portj_MAC_PCntr_IndReg4	Port j MAC PCntr IndReg4	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1923
Portj_MAC_PCntr_IndReg5	Port j MAC PCntr IndReg5	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1924

(table continues...)

Table 775 (continued) Registers overview - GETH (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
Portj_MAC_PCntr_IndReg6	Port j MAC PCntr IndReg6	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1925
Portj_MAC_PCntr_IndReg7	Port j MAC PCntr IndReg7	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1925
Portj_MAC_PCntr_IndReg8	Port j MAC PCntr IndReg8	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1926
Portj_MAC_PCntr_IndReg9	Port j MAC PCntr IndReg9	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1926
Portj_MAC_VPCSel_IndReg0	Port j MAC VPCSel IndReg0	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1927
Portj_MAC_VPCSel_IndReg1	Port j MAC VPCSel IndReg1	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1928
Portj_MAC_VPCSel_IndReg2	Port j MAC VPCSel IndReg2	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1928
Portj_MAC_VPCSel_IndReg3	Port j MAC VPCSel IndReg3	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1929
Portj_MAC_VPCSel_IndReg4	Port j MAC VPCSel IndReg4	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1929
Portj_MAC_VPCSel_IndReg5	Port j MAC VPCSel IndReg5	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1930
Portj_MAC_VPCSel_IndReg6	Port j MAC VPCSel IndReg6	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1931
Portj_MAC_VPCSel_IndReg7	Port j MAC VPCSel IndReg7	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1931
Portj_MAC_PCTH_Intr_Enable	Port j MAC PCTH Intr Enable	10708_H+j *2000_H	PMj	PMj	Kernel Reset	1932
Portj_MAC_PCTH_Intr_Status	Port j MAC PCTH Intr Status	$1070C_H$+ j*2000_H	PMj	PMj	Kernel Reset	1933
Portj_MAC_PCTW_Intr_Enable	Port j MAC PCTW Intr Enable	10730_H+j *2000_H	PMj	PMj	Kernel Reset	1933
Portj_MAC_PCTW_Intr_Status	Port j MAC PCTW Intr Status	10734_H+j *2000_H	PMj	PMj	Kernel Reset	1934
Portj_MMC_Control	Port j MMC Control	10800_H+j *2000_H	PMj	PMj	Kernel Reset	1934
Portj_MMC_Rx_Interrupt	Port j MMC Rx Interrupt	10804_H+j *2000_H	PMj	PMj	Kernel Reset	1936
Portj_MMC_Tx_Interrupt	Port j MMC Tx Interrupt	10808_H+j *2000_H	PMj	PMj	Kernel Reset	1939
Portj_MMC_Receive_Interrupt_Enable	Port j MMC Receive Interrupt Enable	$1080C_H$+ j*2000_H	PMj	PMj	Kernel Reset	1944

(table continues...)

Table 775 (continued) Registers overview - GETH (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
Portj_MMC_Transmit_Interrupt_Enable	Port j MMC Transmit Interrupt Enable	10810_H+j *2000_H	PMj	PMj	Kernel Reset	1948
Portj_Tx_Octet_Count_Good_Bad_Low	Port j Tx Octet Count Good Bad Low	10814_H+j *2000_H	PMj	PMj	Kernel Reset	1952
Portj_Tx_Octet_Count_Good_Bad_High	Port j Tx Octet Count Good Bad High	10818_H+j *2000_H	PMj	PMj	Kernel Reset	1952
Portj_Tx_Packet_Count_Good_Bad_Low	Port j Tx Packet Count Good Bad Low	$1081C_H$+ j*2000_H	PMj	PMj	Kernel Reset	1953
Portj_Tx_Broadcast_Packets_Good_Low	Port j Tx Broadcast Packets Good Low	10824_H+j *2000_H	PMj	PMj	Kernel Reset	1953
Portj_Tx_Multicast_Packets_Good_Low	Port j Tx Multicast Packets Good Low	$1082C_H$+ j*2000_H	PMj	PMj	Kernel Reset	1954
Portj_Tx_64Octets_Packets_Good_Bad_Low	Port j Tx 64Octets Packets Good Bad Low	10834_H+j *2000_H	PMj	PMj	Kernel Reset	1954
Portj_Tx_65To127Octets_Packets_Good_Bad_Low	Port j Tx 65To127Octets Packets Good Bad Low	$1083C_H$+ j*2000_H	PMj	PMj	Kernel Reset	1954
Portj_Tx_128To255Octets_Packets_Good_Bad_Low	Port j Tx 128To255Octets Packets Good Bad Low	10844_H+j *2000_H	PMj	PMj	Kernel Reset	1955
Portj_Tx_256To511Octets_Packets_Good_Bad_Low	Port j Tx 256To511Octets Packets Good Bad Low	$1084C_H$+ j*2000_H	PMj	PMj	Kernel Reset	1955
Portj_Tx_512To1023Octets_Packets_Good_Bad_Low	Port j Tx 512To1023Octets Packets Good Bad Low	10854_H+j *2000_H	PMj	PMj	Kernel Reset	1956
Portj_Tx_1024ToMaxOctets_Packets_Good_Bad_Low	Port j Tx 1024ToMaxOctets Packets Good Bad Low	$1085C_H$+ j*2000_H	PMj	PMj	Kernel Reset	1956
Portj_Tx_Unicast_Packets_Good_Bad_Low	Port j Tx Unicast Packets Good Bad Low	10864_H+j *2000_H	PMj	PMj	Kernel Reset	1957

(table continues...)

Table 775 (continued) Registers overview - GETH (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
Portj_Tx_Multicast_Packets_Good_Bad_Low	Port j Tx Multicast Packets Good Bad Low	$1086C_H$+j*2000_H	PMj	PMj	Kernel Reset	1957
Portj_Tx_Broadcast_Packets_Good_Bad_Low	Port j Tx Broadcast Packets Good Bad Low	10874_H+j*2000_H	PMj	PMj	Kernel Reset	1958
Portj_Tx_Underflow_Error_Packets_Low	Port j Tx Underflow Error Packets Low	$1087C_H$+j*2000_H	PMj	PMj	Kernel Reset	1958
Portj_Tx_Octet_Count_Good_Low	Port j Tx Octet Count Good Low	10884_H+j*2000_H	PMj	PMj	Kernel Reset	1959
Portj_Tx_Octet_Count_Good_High	Port j Tx Octet Count Good High	10888_H+j*2000_H	PMj	PMj	Kernel Reset	1959
Portj_Tx_Packet_Count_Good_Low	Port j Tx Packet Count Good Low	$1088C_H$+j*2000_H	PMj	PMj	Kernel Reset	1959
Portj_Tx_Pause_Packets_Low	Port j Tx Pause Packets Low	10894_H+j*2000_H	PMj	PMj	Kernel Reset	1960
Portj_Tx_VLAN_Packets_Good_Low	Port j Tx VLAN Packets Good Low	$1089C_H$+j*2000_H	PMj	PMj	Kernel Reset	1960
Portj_Tx_LPI_USEC_Cntr	Port j Tx LPI USEC Cntr	$108A4_H$+j*2000_H	PMj	PMj	Kernel Reset	1961
Portj_Tx_LPI_Tran_Cntr	Port j Tx LPI Tran Cntr	$108A8_H$+j*2000_H	PMj	PMj	Kernel Reset	1961
Portj_Rx_Packet_Count_Good_Bad_Low	Port j Rx Packet Count Good Bad Low	10900_H+j*2000_H	PMj	PMj	Kernel Reset	1962
Portj_Rx_Octet_Count_Good_Bad_Low	Port j Rx Octet Count Good Bad Low	10908_H+j*2000_H	PMj	PMj	Kernel Reset	1962
Portj_Rx_Octet_Count_Good_Bad_High	Port j Rx Octet Count Good Bad High	$1090C_H$+j*2000_H	PMj	PMj	Kernel Reset	1962
Portj_Rx_Octet_Count_Good_Low	Port j Rx Octet Count Good Low	10910_H+j*2000_H	PMj	PMj	Kernel Reset	1963
Portj_Rx_Octet_Count_Good_High	Port j Rx Octet Count Good High	10914_H+j*2000_H	PMj	PMj	Kernel Reset	1963

(table continues...)

Table 775 (continued) Registers overview - GETH (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
Portj_Rx_Broadcast_Packets_Good_Low	Port j Rx Broadcast Packets Good Low	10918_H+j *2000_H	PMj	PMj	Kernel Reset	1964
Portj_Rx_Multicast_Packets_Good_Low	Port j Rx Multicast Packets Good Low	10920_H+j *2000_H	PMj	PMj	Kernel Reset	1964
Portj_Rx_CRC_Error_Packets_Low	Port j Rx CRC Error Packets Low	10928_H+j *2000_H	PMj	PMj	Kernel Reset	1965
Portj_Rx_Runt_Error_Packets	Port j Rx Runt Error Packets	10930_H+j *2000_H	PMj	PMj	Kernel Reset	1965
Portj_Rx_Jabber_Error_Packets	Port j Rx Jabber Error Packets	10934_H+j *2000_H	PMj	PMj	Kernel Reset	1965
Portj_Rx_Undersize_Packets_Good	Port j Rx Undersize Packets Good	10938_H+j *2000_H	PMj	PMj	Kernel Reset	1966
Portj_Rx_Oversize_Packets_Good	Port j Rx Oversize Packets Good	$1093C_H$+ j*2000_H	PMj	PMj	Kernel Reset	1966
Portj_Rx_64Octets_Packets_Good_Bad_Low	Port j Rx 64Octets Packets Good Bad Low	10940_H+j *2000_H	PMj	PMj	Kernel Reset	1967
Portj_Rx_65To127Octets_Packets_Good_Bad_Low	Port j Rx 65To127Octets Packets Good Bad Low	10948_H+j *2000_H	PMj	PMj	Kernel Reset	1967
Portj_Rx_128To255Octets_Packets_Good_Bad_Low	Port j Rx 128To255Octets Packets Good Bad Low	10950_H+j *2000_H	PMj	PMj	Kernel Reset	1968
Portj_Rx_256To511Octets_Packets_Good_Bad_Low	Port j Rx 256To511Octets Packets Good Bad Low	10958_H+j *2000_H	PMj	PMj	Kernel Reset	1968
Portj_Rx_512To1023Octets_Packets_Good_Bad_Low	Port j Rx 512To1023Octets Packets Good Bad Low	10960_H+j *2000_H	PMj	PMj	Kernel Reset	1968
Portj_Rx_1024ToMaxOctets_Packets_Good_Bad_Low	Port j Rx 1024ToMaxOctets Packets Good Bad Low	10968_H+j *2000_H	PMj	PMj	Kernel Reset	1969
Portj_Rx_Unicast_Packets_Good_Low	Port j Rx Unicast Packets Good Low	10970_H+j *2000_H	PMj	PMj	Kernel Reset	1969
Portj_Rx_Length_Error_Packets_Low	Port j Rx Length Error Packets Low	10978_H+j *2000_H	PMj	PMj	Kernel Reset	1970

(table continues...)

Table 775 (continued) Registers overview - GETH (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
Portj_Rx_OutofRange_Packets_Low	Port j Rx OutofRange Packets Low	10980_H+j *2000_H	PMj	PMj	Kernel Reset	1970
Portj_Rx_Pause_Packets_Low	Port j Rx Pause Packets Low	10988_H+j *2000_H	PMj	PMj	Kernel Reset	1971
Portj_Rx_FIFOOverflow_Packets_Low	Port j Rx FIFOOverflow Packets Low	10990_H+j *2000_H	PMj	PMj	Kernel Reset	1971
Portj_Rx_VLAN_Packets_Good_Bad_Low	Port j Rx VLAN Packets Good Bad Low	10998_H+j *2000_H	PMj	PMj	Kernel Reset	1972
Portj_Rx_Watchdog_Error_Packets	Port j Rx Watchdog Error Packets	$109A0_H$+ j*2000_H	PMj	PMj	Kernel Reset	1972
Portj_Rx_LPI_USEC_Cntr	Port j Rx LPI USEC Cntr	$109A4_H$+ j*2000_H	PMj	PMj	Kernel Reset	1972
Portj_Rx_LPI_Tran_Cntr	Port j Rx LPI Tran Cntr	$109A8_H$+ j*2000_H	PMj	PMj	Kernel Reset	1973
Portj_Rx_Discard_Packets_Good_Bad_Low	Port j Rx Discard Packets Good Bad Low	$109AC_H$+ j*2000_H	PMj	PMj	Kernel Reset	1973
Portj_Rx_Discard_Octets_Good_Bad_Low	Port j Rx Discard Octets Good Bad Low	$109B4_H$+ j*2000_H	PMj	PMj	Kernel Reset	1974
Portj_Rx_Discard_Octets_Good_Bad_High	Port j Rx Discard Octets Good Bad High	$109B8_H$+ j*2000_H	PMj	PMj	Kernel Reset	1974
Portj_Rx_Alignment_Error_Packets	Port j Rx Alignment Error Packets	$109BC_H$+ j*2000_H	PMj	PMj	Kernel Reset	1975
Portj_SGF_Pass_PktCnt	Port j SGF Pass PktCnt	$109F0_H$+j *2000_H	PMj	PMj	Kernel Reset	1975
Portj_SGF_Fail_PktCnt	Port j SGF Fail PktCnt	$109F4_H$+j *2000_H	PMj	PMj	Kernel Reset	1976
Portj_MMC_FPE_Tx_Interrupt	Port j MMC FPE Tx Interrupt	$10A00_H$+ j*2000_H	PMj	PMj	Kernel Reset	1976
Portj_MMC_FPE_Tx_Interrupt_Mask	Port j MMC FPE Tx Interrupt Mask	$10A04_H$+ j*2000_H	PMj	PMj	Kernel Reset	1977
Portj_MMC_Tx_FPE_Fragment_Cntr	Port j MMC Tx FPE Fragment Cntr	$10A08_H$+ j*2000_H	PMj	PMj	Kernel Reset	1978
Portj_MMC_Tx_Hold_Req_Cntr	Port j MMC Tx Hold Req Cntr	$10A0C_H$+ j*2000_H	PMj	PMj	Kernel Reset	1979

(table continues...)

Table 775 (continued) Registers overview - GETH (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
Portj_MMC_Tx_Gate_Orun_Cntr_Low	Port j MMC Tx Gate Orun Cntr Low	$10A10_H$+ j*2000_H	PMj	PMj	Kernel Reset	1979
Portj_MMC_Tx_Gate_Orun_Cntr_High	Port j MMC Tx Gate Orun Cntr High	$10A14_H$+ j*2000_H	PMj	PMj	Kernel Reset	1980
Portj_MMC_FPE_Rx_Interrupt	Port j MMC FPE Rx Interrupt	$10A20_H$+ j*2000_H	PMj	PMj	Kernel Reset	1980
Portj_MMC_FPE_Rx_Interrupt_Mask	Port j MMC FPE Rx Interrupt Mask	$10A24_H$+ j*2000_H	PMj	PMj	Kernel Reset	1982
Portj_MMC_Rx_Packet_Assembly_Err_Cntr	Port j MMC Rx Packet Assembly Err Cntr	$10A28_H$+ j*2000_H	PMj	PMj	Kernel Reset	1983
Portj_MMC_Rx_Packet_SMD_Err_Cntr	Port j MMC Rx Packet SMD Err Cntr	$10A2C_H$+ j*2000_H	PMj	PMj	Kernel Reset	1984
Portj_MMC_Rx_Packet_Assembly_OK_Cntr	Port j MMC Rx Packet Assembly OK Cntr	$10A30_H$+ j*2000_H	PMj	PMj	Kernel Reset	1984
Portj_MMC_Rx_FPE_Fragment_Cntr	Port j MMC Rx FPE Fragment Cntr	$10A34_H$+ j*2000_H	PMj	PMj	Kernel Reset	1985
Portj_Tx_Single_Collision_Good_Packets	Port j Tx Single Collision Good Packets	$10A40_H$+ j*2000_H	PMj	PMj	Kernel Reset	1985
Portj_Tx_Multiple_Collision_Good_Packets	Port j Tx Multiple Collision Good Packets	$10A44_H$+ j*2000_H	PMj	PMj	Kernel Reset	1986
Portj_Tx_Deferred_Packets	Port j Tx Deferred Packets	$10A48_H$+ j*2000_H	PMj	PMj	Kernel Reset	1986
Portj_Tx_Late_Collision_Packets	Port j Tx Late Collision Packets	$10A4C_H$+ j*2000_H	PMj	PMj	Kernel Reset	1987
Portj_Tx_Excessive_Collision_Packets	Port j Tx Excessive Collision Packets	$10A50_H$+ j*2000_H	PMj	PMj	Kernel Reset	1987
Portj_Tx_Carrier_Error_Packets	Port j Tx Carrier Error Packets	$10A54_H$+ j*2000_H	PMj	PMj	Kernel Reset	1987
Portj_Tx_Excessive_Deferral_Error	Port j Tx Excessive Deferral Error	$10A58_H$+ j*2000_H	PMj	PMj	Kernel Reset	1988
Portj_MMC_IPC_Rx_Interrupt_Mask	Port j MMC IPC Rx Interrupt Mask	$10A5C_H$+ j*2000_H	PMj	PMj	Kernel Reset	1988

(table continues...)

Table 775 (continued) Registers overview - GETH (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
Portj_MMC_IPC_Rx_Interrupt	Port j MMC IPC Rx Interrupt	$10A60_H$+ j*2000_H	PMj	PMj	Kernel Reset	1994
Portj_RxIPv4_Good_Packets_Low	Port j RxIPv4 Good Packets Low	$10A64_H$+ j*2000_H	PMj	PMj	Kernel Reset	2000
Portj_RxIPv4_Header_Error_Packets_Low	Port j RxIPv4 Header Error Packets Low	$10A6C_H$+ j*2000_H	PMj	PMj	Kernel Reset	2001
Portj_RxIPv4_No_Payload_Packets_Low	Port j RxIPv4 No Payload Packets Low	$10A74_H$+ j*2000_H	PMj	PMj	Kernel Reset	2001
Portj_RxIPv4_Fragmented_Packets_Low	Port j RxIPv4 Fragmented Packets Low	$10A7C_H$+ j*2000_H	PMj	PMj	Kernel Reset	2002
Portj_RxIPv4_UDP_Checksum_Disabled_Packets_Low	Port j RxIPv4 UDP Checksum Disabled Packets Low	$10A84_H$+ j*2000_H	PMj	PMj	Kernel Reset	2002
Portj_RxIPv6_Good_Packets_Low	Port j RxIPv6 Good Packets Low	$10A8C_H$+ j*2000_H	PMj	PMj	Kernel Reset	2002
Portj_RxIPv6_Header_Error_Packets_Low	Port j RxIPv6 Header Error Packets Low	$10A94_H$+ j*2000_H	PMj	PMj	Kernel Reset	2003
Portj_RxIPv6_No_Payload_Packets_Low	Port j RxIPv6 No Payload Packets Low	$10A9C_H$+ j*2000_H	PMj	PMj	Kernel Reset	2003
Portj_RxUDP_Good_Packets_Low	Port j RxUDP Good Packets Low	$10AA4_H$+ j*2000_H	PMj	PMj	Kernel Reset	2004
Portj_RxUDP_Error_Packets_Low	Port j RxUDP Error Packets Low	$10AAC_H$+ j*2000_H	PMj	PMj	Kernel Reset	2004
Portj_RxTCP_Good_Packets_Low	Port j RxTCP Good Packets Low	$10AB4_H$+ j*2000_H	PMj	PMj	Kernel Reset	2005
Portj_RxTCP_Error_Packets_Low	Port j RxTCP Error Packets Low	$10ABC_H$+ j*2000_H	PMj	PMj	Kernel Reset	2005
Portj_RxICMP_Good_Packets_Low	Port j RxICMP Good Packets Low	$10AC4_H$+ j*2000_H	PMj	PMj	Kernel Reset	2005
Portj_RxICMP_Error_Packets_Low	Port j RxICMP Error Packets Low	$10ACC_H$+ j*2000_H	PMj	PMj	Kernel Reset	2006
Portj_RxIPv4_Good_Octets_Low	Port j RxIPv4 Good Octets Low	$10AD4_H$+ j*2000_H	PMj	PMj	Kernel Reset	2006
Portj_RxIPv4_Header_Error_Octets_Low	Port j RxIPv4 Header Error Octets Low	$10ADC_H$+ j*2000_H	PMj	PMj	Kernel Reset	2007

(table continues...)

Table 775 (continued) Registers overview - GETH (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
Portj_RxIPv4_No_Payload_Octets_Low	Port j RxIPv4 No Payload Octets Low	$10AE4_H$+ j*2000_H	PMj	PMj	Kernel Reset	2007
Portj_RxIPv4_Fragmented_Octets_Low	Port j RxIPv4 Fragmented Octets Low	$10AEC_H$+ j*2000_H	PMj	PMj	Kernel Reset	2008
Portj_RxIPv4_UDP_Checksum_Disable_Octets_Low	Port j RxIPv4 UDP Checksum Disable Octets Low	$10AF4_H$+ j*2000_H	PMj	PMj	Kernel Reset	2008
Portj_RxIPv6_Good_Octets_Low	Port j RxIPv6 Good Octets Low	$10AFC_H$+ j*2000_H	PMj	PMj	Kernel Reset	2009
Portj_RxIPv6_Header_Error_Octets_Low	Port j RxIPv6 Header Error Octets Low	$10B04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2009
Portj_RxIPv6_No_Payload_Octets_Low	Port j RxIPv6 No Payload Octets Low	$10B0C_H$+ j*2000_H	PMj	PMj	Kernel Reset	2010
Portj_RxUDP_Good_Octets_Low	Port j RxUDP Good Octets Low	$10B14_H$+ j*2000_H	PMj	PMj	Kernel Reset	2010
Portj_RxUDP_Error_Octets_Low	Port j RxUDP Error Octets Low	$10B1C_H$+ j*2000_H	PMj	PMj	Kernel Reset	2011
Portj_RxTCP_Good_Octets_Low	Port j RxTCP Good Octets Low	$10B24_H$+ j*2000_H	PMj	PMj	Kernel Reset	2011
Portj_RxTCP_Error_Octets_Low	Port j RxTCP Error Octets Low	$10B2C_H$+ j*2000_H	PMj	PMj	Kernel Reset	2012
Portj_RxICMP_Good_Octets_Low	Port j RxICMP Good Octets Low	$10B34_H$+ j*2000_H	PMj	PMj	Kernel Reset	2012
Portj_RxICMP_Error_Octets_Low	Port j RxICMP Error Octets Low	$10B3C_H$+ j*2000_H	PMj	PMj	Kernel Reset	2012
Portj_MAC_L3_L4_Address_Control	Port j MAC L3 L4 Address Control	$10C00_H$+ j*2000_H	PMj	PMj	Kernel Reset	2013
Portj_MAC_L3_L4_Data	Port j MAC L3 L4 Data	$10C04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2014
Portj_MAC_L3_L4_Control0	Port j MAC L3 L4 Control0	$10C04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2014
Portj_MAC_L3_L4_Control1	Port j MAC L3 L4 Control1	$10C04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2017
Portj_MAC_L3_L4_Control2	Port j MAC L3 L4 Control2	$10C04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2020
Portj_MAC_L3_L4_Control3	Port j MAC L3 L4 Control3	$10C04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2023

(table continues...)

Table 775 (continued) Registers overview - GETH (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
Portj_MAC_L3_L4_Control4	Port j MAC L3 L4 Control4	$10C04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2026
Portj_MAC_L3_L4_Control5	Port j MAC L3 L4 Control5	$10C04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2029
Portj_MAC_L3_L4_Control6	Port j MAC L3 L4 Control6	$10C04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2032
Portj_MAC_L3_L4_Control7	Port j MAC L3 L4 Control7	$10C04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2035
Portj_MAC_Layer3_Addr0_Reg0	Port j MAC Layer3 Addr0 Reg0	$10C04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2038
Portj_MAC_Layer3_Addr0_Reg1	Port j MAC Layer3 Addr0 Reg1	$10C04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2039
Portj_MAC_Layer3_Addr0_Reg2	Port j MAC Layer3 Addr0 Reg2	$10C04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2040
Portj_MAC_Layer3_Addr0_Reg3	Port j MAC Layer3 Addr0 Reg3	$10C04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2040
Portj_MAC_Layer3_Addr0_Reg4	Port j MAC Layer3 Addr0 Reg4	$10C04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2041
Portj_MAC_Layer3_Addr0_Reg5	Port j MAC Layer3 Addr0 Reg5	$10C04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2041
Portj_MAC_Layer3_Addr0_Reg6	Port j MAC Layer3 Addr0 Reg6	$10C04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2042
Portj_MAC_Layer3_Addr0_Reg7	Port j MAC Layer3 Addr0 Reg7	$10C04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2043
Portj_MAC_Layer3_Addr1_Reg0	Port j MAC Layer3 Addr1 Reg0	$10C04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2043
Portj_MAC_Layer3_Addr1_Reg1	Port j MAC Layer3 Addr1 Reg1	$10C04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2044
Portj_MAC_Layer3_Addr1_Reg2	Port j MAC Layer3 Addr1 Reg2	$10C04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2044
Portj_MAC_Layer3_Addr1_Reg3	Port j MAC Layer3 Addr1 Reg3	$10C04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2045
Portj_MAC_Layer3_Addr1_Reg4	Port j MAC Layer3 Addr1 Reg4	$10C04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2046
Portj_MAC_Layer3_Addr1_Reg5	Port j MAC Layer3 Addr1 Reg5	$10C04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2046
Portj_MAC_Layer3_Addr1_Reg6	Port j MAC Layer3 Addr1 Reg6	$10C04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2047
Portj_MAC_Layer3_Addr1_Reg7	Port j MAC Layer3 Addr1 Reg7	$10C04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2047
Portj_MAC_Layer3_Addr2_Reg0	Port j MAC Layer3 Addr2 Reg0	$10C04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2048

(table continues...)

Table 775 (continued) Registers overview - GETH (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
Portj_MAC_Layer3_Addr2_Reg1	Port j MAC Layer3 Addr2 Reg1	$10C04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2049
Portj_MAC_Layer3_Addr2_Reg2	Port j MAC Layer3 Addr2 Reg2	$10C04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2049
Portj_MAC_Layer3_Addr2_Reg3	Port j MAC Layer3 Addr2 Reg3	$10C04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2050
Portj_MAC_Layer3_Addr2_Reg4	Port j MAC Layer3 Addr2 Reg4	$10C04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2050
Portj_MAC_Layer3_Addr2_Reg5	Port j MAC Layer3 Addr2 Reg5	$10C04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2051
Portj_MAC_Layer3_Addr2_Reg6	Port j MAC Layer3 Addr2 Reg6	$10C04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2052
Portj_MAC_Layer3_Addr2_Reg7	Port j MAC Layer3 Addr2 Reg7	$10C04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2052
Portj_MAC_Layer3_Addr3_Reg0	Port j MAC Layer3 Addr3 Reg0	$10C04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2053
Portj_MAC_Layer3_Addr3_Reg1	Port j MAC Layer3 Addr3 Reg1	$10C04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2053
Portj_MAC_Layer3_Addr3_Reg2	Port j MAC Layer3 Addr3 Reg2	$10C04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2054
Portj_MAC_Layer3_Addr3_Reg3	Port j MAC Layer3 Addr3 Reg3	$10C04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2055
Portj_MAC_Layer3_Addr3_Reg4	Port j MAC Layer3 Addr3 Reg4	$10C04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2055
Portj_MAC_Layer3_Addr3_Reg5	Port j MAC Layer3 Addr3 Reg5	$10C04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2056
Portj_MAC_Layer3_Addr3_Reg6	Port j MAC Layer3 Addr3 Reg6	$10C04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2056
Portj_MAC_Layer3_Addr3_Reg7	Port j MAC Layer3 Addr3 Reg7	$10C04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2057
Portj_MAC_Layer4_Address0	Port j MAC Layer4 Address0	$10C04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2058
Portj_MAC_Layer4_Address1	Port j MAC Layer4 Address1	$10C04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2058
Portj_MAC_Layer4_Address2	Port j MAC Layer4 Address2	$10C04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2059
Portj_MAC_Layer4_Address3	Port j MAC Layer4 Address3	$10C04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2060
Portj_MAC_Layer4_Address4	Port j MAC Layer4 Address4	$10C04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2061
Portj_MAC_Layer4_Address5	Port j MAC Layer4 Address5	$10C04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2061

(table continues...)

Table 775 (continued) Registers overview - GETH (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
Portj_MAC_Layer4_Address6	Port j MAC Layer4 Address6	$10C04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2062
Portj_MAC_Layer4_Address7	Port j MAC Layer4 Address7	$10C04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2063
Portj_MAC_Timestamp_Control	Port j MAC Timestamp Control	$10D00_H$+ j*2000_H	PMj	PMj	Kernel Reset	2064
Portj_MAC_Sub_Second_Increment	Port j MAC Sub Second Increment	$10D04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2067
Portj_MAC_System_Time_Seconds	Port j MAC System Time Seconds	$10D08_H$+ j*2000_H	PMj	PMj	Kernel Reset	2068
Portj_MAC_System_Time_Nanoseconds	Port j MAC System Time Nanoseconds	$10D0C_H$+ j*2000_H	PMj	PMj	Kernel Reset	2068
Portj_MAC_System_Time_Seconds_Update	Port j MAC System Time Seconds Update	$10D10_H$+ j*2000_H	PMj	PMj	Kernel Reset	2069
Portj_MAC_System_Time_Nanoseconds_Update	Port j MAC System Time Nanoseconds Update	$10D14_H$+ j*2000_H	PMj	PMj	Kernel Reset	2069
Portj_MAC_Timestamp_Addend	Port j MAC Timestamp Addend	$10D18_H$+ j*2000_H	PMj	PMj	Kernel Reset	2070
Portj_MAC_System_Time_Higher_Word_Seconds	Port j MAC System Time Higher Word Seconds	$10D1C_H$+ j*2000_H	PMj	PMj	Kernel Reset	2070
Portj_MAC_Timestamp_Status	Port j MAC Timestamp Status	$10D20_H$+ j*2000_H	PMj	PMj	Kernel Reset	2071
Portj_MAC_Rx_Domain_Time_Incr	Port j MAC Rx Domain Time Incr	$10D24_H$+ j*2000_H	PMj	PMj	Kernel Reset	2074
Portj_MAC_Tx_Domain_Time_Incr	Port j MAC Tx Domain Time Incr	$10D28_H$+ j*2000_H	PMj	PMj	Kernel Reset	2075
Portj_MAC_Tx_Timestamp_Status_Nanoseconds	Port j MAC Tx Timestamp Status Nanoseconds	$10D30_H$+ j*2000_H	PMj	PMj	Kernel Reset	2076
Portj_MAC_Tx_Timestamp_Status_Seconds	Port j MAC Tx Timestamp Status Seconds	$10D34_H$+ j*2000_H	PMj	PMj	Kernel Reset	2077
Portj_MAC_Tx_Timestamp_Status_PktID	Port j MAC Tx Timestamp Status PktID	$10D38_H$+ j*2000_H	PMj	PMj	Kernel Reset	2077

(table continues...)

Table 775 (continued) Registers overview - GETH (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
Portj_MAC_Timestamp_Ingress_Asym_Corr	Port j MAC Timestamp Ingress Asym Corr	$10D50_H$+ j*2000_H	PMj	PMj	Kernel Reset	2078
Portj_MAC_Timestamp_Egress_Asym_Corr	Port j MAC Timestamp Egress Asym Corr	$10D54_H$+ j*2000_H	PMj	PMj	Kernel Reset	2079
Portj_MAC_Timestamp_Ingress_Corr_Nanosecond	Port j MAC Timestamp Ingress Corr Nanosecond	$10D58_H$+ j*2000_H	PMj	PMj	Kernel Reset	2079
Portj_MAC_Timestamp_Ingress_Corr_Subnanosecond	Port j MAC Timestamp Ingress Corr Subnanosecond	$10D5C_H$+ j*2000_H	PMj	PMj	Kernel Reset	2080
Portj_MAC_Timestamp_Egress_Corr_Nanosecond	Port j MAC Timestamp Egress Corr Nanosecond	$10D60_H$+ j*2000_H	PMj	PMj	Kernel Reset	2080
Portj_MAC_Timestamp_Egress_Corr_Subnanosecond	Port j MAC Timestamp Egress Corr Subnanosecond	$10D64_H$+ j*2000_H	PMj	PMj	Kernel Reset	2081
Portj_MAC_PPS_Control	Port j MAC PPS Control	$10D70_H$+ j*2000_H	PMj	PMj	Kernel Reset	2081
Portj_MAC_PPS_Extnd_Ctrl	Port j MAC PPS Extnd Ctrl	$10D74_H$+ j*2000_H	PMj	PMj	Kernel Reset	2086
Portj_MAC_PPS0_Target_Time_Seconds	Port j MAC PPS0 Target Time Seconds	$10D80_H$+ j*2000_H	PMj	PMj	Kernel Reset	2086
Portj_MAC_PPS0_Target_Time_Nanoseconds	Port j MAC PPS0 Target Time Nanoseconds	$10D84_H$+ j*2000_H	PMj	PMj	Kernel Reset	2087
Portj_MAC_PPS0_Interval	Port j MAC PPS0 Interval	$10D88_H$+ j*2000_H	PMj	PMj	Kernel Reset	2088
Portj_MAC_PPS0_Width	Port j MAC PPS0 Width	$10D8C_H$+ j*2000_H	PMj	PMj	Kernel Reset	2089
Portj_MAC_PPS1_Target_Time_Seconds	Port j MAC PPS1 Target Time Seconds	$10D90_H$+ j*2000_H	PMj	PMj	Kernel Reset	2089
Portj_MAC_PPS1_Target_Time_Nanoseconds	Port j MAC PPS1 Target Time Nanoseconds	$10D94_H$+ j*2000_H	PMj	PMj	Kernel Reset	2090
Portj_MAC_PPS1_Interval	Port j MAC PPS1 Interval	$10D98_H$+ j*2000_H	PMj	PMj	Kernel Reset	2091
Portj_MAC_PPS1_Width	Port j MAC PPS1 Width	$10D9C_H$+ j*2000_H	PMj	PMj	Kernel Reset	2091

(table continues...)

Table 775 (continued) Registers overview - GETH (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
Portj_MAC_Presn_Time_ns	Port j MAC Presn Time ns	$10DE0_H$+j*2000_H	PMj	PMj	Kernel Reset	2092
Portj_MAC_Presn_Time_Updt	Port j MAC Presn Time Updt	$10DE4_H$+j*2000_H	PMj	PMj	Kernel Reset	2092
Portj_MTL_Operation_Mode	Port j MTL Operation Mode	11000_H+j*2000_H	PMj	PMj	Kernel Reset	2093
Portj_MTL_Debug_Control	Port j MTL Debug Control	11008_H+j*2000_H	PMj	PMj	Kernel Reset	2094
Portj_MTL_Debug_Status	Port j MTL Debug Status	$1100C_H$+j*2000_H	PMj	PMj	Kernel Reset	2096
Portj_MTL_FIFO_Debug_Data	Port j MTL FIFO Debug Data	11010_H+j*2000_H	PMj	PMj	Kernel Reset	2098
Portj_MTL_Interrupt_Status	Port j MTL Interrupt Status	11020_H+j*2000_H	PMj	PMj	Kernel Reset	2098
Portj_MTL_RxQ_DMA_Map0	Port j MTL RxQ DMA Map0	11030_H+j*2000_H	PMj	PMj	Kernel Reset	2101
Portj_MTL_RxQ_DMA_Map1	Port j MTL RxQ DMA Map1	11034_H+j*2000_H	PMj	PMj	Kernel Reset	2103
Portj_MTL_TBS_CTRL	Port j MTL TBS CTRL	11048_H+j*2000_H	PMj	PMj	Kernel Reset	2104
Portj_MTL_TBS_STATS	Port j MTL TBS STATS	$1104C_H$+j*2000_H	PMj	PMj	Kernel Reset	2105
Portj_MTL_EST_Control	Port j MTL EST Control	11050_H+j*2000_H	PMj	PMj	Kernel Reset	2106
Portj_MTL_EST_Overhead	Port j MTL EST Overhead	11054_H+j*2000_H	PMj	PMj	Kernel Reset	2108
Portj_MTL_EST_Status	Port j MTL EST Status	11058_H+j*2000_H	PMj	PMj	Kernel Reset	2109
Portj_MTL_EST_Sch_Error	Port j MTL EST Sch Error	11060_H+j*2000_H	PMj	PMj	Kernel Reset	2111
Portj_MTL_EST_Frm_Size_Error	Port j MTL EST Frm Size Error	11064_H+j*2000_H	PMj	PMj	Kernel Reset	2112
Portj_MTL_EST_Frm_Size_Capture	Port j MTL EST Frm Size Capture	11068_H+j*2000_H	PMj	PMj	Kernel Reset	2112
Portj_MTL_EST_Intr_Enable	Port j MTL EST Intr Enable	11070_H+j*2000_H	PMj	PMj	Kernel Reset	2113
Portj_MTL_GCL_Control	Port j MTL GCL Control	11080_H+j*2000_H	PMj	PMj	Kernel Reset	2114
Portj_MTL_GCL_Data	Port j MTL GCL Data	11084_H+j*2000_H	PMj	PMj	Kernel Reset	2117

(table continues...)

Table 775 (continued) Registers overview - GETH (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
Portj_MTL_FPE_CTRL_STS	Port j MTL FPE CTRL STS	11090_H+j *2000_H	PMj	PMj	Kernel Reset	2117
Portj_MTL_FPE_Advance	Port j MTL FPE Advance	11094_H+j *2000_H	PMj	PMj	Kernel Reset	2118
Portj_MTL_RXP_Control_Status	Port j MTL RXP Control Status	$110A0_H$+ j*2000_H	PMj	PMj	Kernel Reset	2119
Portj_MTL_RXP_Interrupt_Control_Status	Port j MTL RXP Interrupt Control Status	$110A4_H$+ j*2000_H	PMj	PMj	Kernel Reset	2120
Portj_MTL_Indirect_Acc_Control_Status	Port j MTL Indirect Acc Control Status	$110B0_H$+ j*2000_H	PMj	PMj	Kernel Reset	2122
Portj_MTL_Indirect_Acc_Data	Port j MTL Indirect Acc Data	$110B4_H$+ j*2000_H	PMj	PMj	Kernel Reset	2124
Portj_MTL_FRP_ACNTR0	Port j MTL FRP ACNTR0	$110B4_H$+ j*2000_H	PMj	PMj	Kernel Reset	2125
Portj_MTL_FRP_ACNTR1	Port j MTL FRP ACNTR1	$110B4_H$+ j*2000_H	PMj	PMj	Kernel Reset	2126
Portj_MTL_FRP_ACNTR2	Port j MTL FRP ACNTR2	$110B4_H$+ j*2000_H	PMj	PMj	Kernel Reset	2126
Portj_MTL_FRP_ACNTR3	Port j MTL FRP ACNTR3	$110B4_H$+ j*2000_H	PMj	PMj	Kernel Reset	2127
Portj_MTL_FRP_ACNTR4	Port j MTL FRP ACNTR4	$110B4_H$+ j*2000_H	PMj	PMj	Kernel Reset	2128
Portj_MTL_FRP_ACNTR5	Port j MTL FRP ACNTR5	$110B4_H$+ j*2000_H	PMj	PMj	Kernel Reset	2129
Portj_MTL_FRP_ACNTR6	Port j MTL FRP ACNTR6	$110B4_H$+ j*2000_H	PMj	PMj	Kernel Reset	2129
Portj_MTL_FRP_ACNTR7	Port j MTL FRP ACNTR7	$110B4_H$+ j*2000_H	PMj	PMj	Kernel Reset	2130
Portj_MTL_RXP_Bypass_Cnt	Port j MTL RXP Bypass Cnt	$110B4_H$+ j*2000_H	PMj	PMj	Kernel Reset	2131
Portj_MTL_RXP_Drop_Cnt	Port j MTL RXP Drop Cnt	$110B4_H$+ j*2000_H	PMj	PMj	Kernel Reset	2132
Portj_MTL_RXP_Error_Cnt	Port j MTL RXP Error Cnt	$110B4_H$+ j*2000_H	PMj	PMj	Kernel Reset	2132
Portj_MTL_ECC_Control	Port j MTL ECC Control	$110C0_H$+ j*2000_H	PMj	PMj	Kernel Reset	2133
Portj_MTL_Safety_Interrupt_Status	Port j MTL Safety Interrupt Status	$110C4_H$+ j*2000_H	PMj	PMj	Kernel Reset	2134

(table continues...)

Table 775 (continued) Registers overview - GETH (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
Portj_MTL_ECC_Interrupt_Enable	Port j MTL ECC Interrupt Enable	$110C8_H$+ j*2000_H	PMj	PMj	Kernel Reset	2135
Portj_MTL_ECC_Interrupt_Status	Port j MTL ECC Interrupt Status	$110CC_H$+ j*2000_H	PMj	PMj	Kernel Reset	2136
Portj_MTL_ECC_Err_Sts_Rctl	Port j MTL ECC Err Sts Rctl	$110D0_H$+ j*2000_H	PMj	PMj	Kernel Reset	2139
Portj_MTL_ECC_Err_Addr_Status	Port j MTL ECC Err Addr Status	$110D4_H$+ j*2000_H	PMj	PMj	Kernel Reset	2140
Portj_MTL_ECC_Err_Cntr_Status	Port j MTL ECC Err Cntr Status	$110D8_H$+ j*2000_H	PMj	PMj	Kernel Reset	2141
Portj_MTL_DPP_Control	Port j MTL DPP Control	$110E0_H$+j *2000_H	PMj	PMj	Kernel Reset	2142
Portj_MTL_DPP_ECC_EIC	Port j MTL DPP ECC EIC	$110E4_H$+j *2000_H	PMj	PMj	Kernel Reset	2145
Portj_MTL_SGF_Control	Port j MTL SGF Control	$110E8_H$+j *2000_H	PMj	PMj	Kernel Reset	2145
Portj_MTL_SGF_Extended_Control	Port j MTL SGF Extended Control	$110EC_H$+ j*2000_H	PMj	PMj	Kernel Reset	2147
Portj_MTL_SGF_Status	Port j MTL SGF Status	$110F0_H$+j *2000_H	PMj	PMj	Kernel Reset	2148
Portj_MTL_SGF_Intr_Enable	Port j MTL SGF Interrupt Enable	$110F4_H$+j *2000_H	PMj	PMj	Kernel Reset	2150
Portj_MTL_TxQ0_Operation_Mode	Port j MTL TxQ0 Operation Mode	11100_H+j *2000_H	PMj	PMj	Kernel Reset	2151
Portj_MTL_TxQ0_Underflow	Port j MTL TxQ0 Underflow	11104_H+j *2000_H	PMj	PMj	Kernel Reset	2152
Portj_MTL_TxQ0_Debug	Port j MTL TxQ0 Debug	11108_H+j *2000_H	PMj	PMj	Kernel Reset	2153
Portj_MTL_TC0_ETS_Control	Port j MTL TC0 ETS Control	11110_H+j *2000_H	PMj	PMj	Kernel Reset	2154
Portj_MTL_TC0_Quantum_Weight	Port j MTL TC0 Quantum Weight	11118_H+j *2000_H	PMj	PMj	Kernel Reset	2155
Portj_MTL_RxQ0_Operation_Mode	Port j MTL RxQ0 Operation Mode	11140_H+j *2000_H	PMj	PMj	Kernel Reset	2155
Portj_MTL_RxQ0_Missed_Pkt_Overflow_Cnt	Port j MTL RxQ0 Missed Pkt Overflow Cnt	11144_H+j *2000_H	PMj	PMj	Kernel Reset	2157
Portj_MTL_RxQ0_Debug	Port j MTL RxQ0 Debug	11148_H+j *2000_H	PMj	PMj	Kernel Reset	2158

(table continues...)

Table 775 (continued) Registers overview - GETH (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
Portj_MTL_RxQ0_Control	Port j MTL RxQ0 Control	$1114C_H$+j*2000_H	PMj	PMj	Kernel Reset	2159
Portj_MTL_RxQ0_Flow_Control	Port j MTL RxQ0 Flow Control	11150_H+j*2000_H	PMj	PMj	Kernel Reset	2160
Portj_MTL_Q0_Interrupt_Enable	Port j MTL Q0 Interrupt Enable	11170_H+j*2000_H	PMj	PMj	Kernel Reset	2161
Portj_MTL_Q0_Interrupt_Status	Port j MTL Q0 Interrupt Status	11174_H+j*2000_H	PMj	PMj	Kernel Reset	2161
Portj_MTL_TxQ1_Operation_Mode	Port j MTL TxQ1 Operation Mode	11180_H+j*2000_H	PMj	PMj	Kernel Reset	2162
Portj_MTL_TxQ1_Underflow	Port j MTL TxQ1 Underflow	11184_H+j*2000_H	PMj	PMj	Kernel Reset	2164
Portj_MTL_TxQ1_Debug	Port j MTL TxQ1 Debug	11188_H+j*2000_H	PMj	PMj	Kernel Reset	2164
Portj_MTL_TC1_ETS_Control	Port j MTL TC1 ETS Control	11190_H+j*2000_H	PMj	PMj	Kernel Reset	2165
Portj_MTL_TC1_Quantum_Weight	Port j MTL TC1 Quantum Weight	11198_H+j*2000_H	PMj	PMj	Kernel Reset	2166
Portj_MTL_RxQ1_Operation_Mode	Port j MTL RxQ1 Operation Mode	$111C0_H$+j*2000_H	PMj	PMj	Kernel Reset	2167
Portj_MTL_RxQ1_Missed_Pkt_Overflow_Cnt	Port j MTL RxQ1 Missed Pkt Overflow Cnt	$111C4_H$+j*2000_H	PMj	PMj	Kernel Reset	2168
Portj_MTL_RxQ1_Debug	Port j MTL RxQ1 Debug	$111C8_H$+j*2000_H	PMj	PMj	Kernel Reset	2169
Portj_MTL_RxQ1_Control	Port j MTL RxQ1 Control	$111CC_H$+j*2000_H	PMj	PMj	Kernel Reset	2170
Portj_MTL_RxQ1_Flow_Control	Port j MTL RxQ1 Flow Control	$111D0_H$+j*2000_H	PMj	PMj	Kernel Reset	2171
Portj_MTL_Q1_Interrupt_Enable	Port j MTL Q1 Interrupt Enable	$111F0_H$+j*2000_H	PMj	PMj	Kernel Reset	2172
Portj_MTL_Q1_Interrupt_Status	Port j MTL Q1 Interrupt Status	$111F4_H$+j*2000_H	PMj	PMj	Kernel Reset	2172
Portj_MTL_TxQ2_Operation_Mode	Port j MTL TxQ2 Operation Mode	11200_H+j*2000_H	PMj	PMj	Kernel Reset	2173
Portj_MTL_TxQ2_Underflow	Port j MTL TxQ2 Underflow	11204_H+j*2000_H	PMj	PMj	Kernel Reset	2175
Portj_MTL_TxQ2_Debug	Port j MTL TxQ2 Debug	11208_H+j*2000_H	PMj	PMj	Kernel Reset	2175

(table continues...)

Table 775 (continued) Registers overview - GETH (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
Portj_MTL_TC2_ETS_Control	Port j MTL TC2 ETS Control	11210_H+j *2000_H	PMj	PMj	Kernel Reset	2176
Portj_MTL_TC2_Quantum_Weight	Port j MTL TC2 Quantum Weight	11218_H+j *2000_H	PMj	PMj	Kernel Reset	2177
Portj_MTL_RxQ2_Operation_Mode	Port j MTL RxQ2 Operation Mode	11240_H+j *2000_H	PMj	PMj	Kernel Reset	2178
Portj_MTL_RxQ2_Missed_Pkt_Overflow_Cnt	Port j MTL RxQ2 Missed Pkt Overflow Cnt	11244_H+j *2000_H	PMj	PMj	Kernel Reset	2179
Portj_MTL_RxQ2_Debug	Port j MTL RxQ2 Debug	11248_H+j *2000_H	PMj	PMj	Kernel Reset	2180
Portj_MTL_RxQ2_Control	Port j MTL RxQ2 Control	$1124C_H$+ j*2000_H	PMj	PMj	Kernel Reset	2181
Portj_MTL_RxQ2_Flow_Control	Port j MTL RxQ2 Flow Control	11250_H+j *2000_H	PMj	PMj	Kernel Reset	2182
Portj_MTL_Q2_Interrupt_Enable	Port j MTL Q2 Interrupt Enable	11270_H+j *2000_H	PMj	PMj	Kernel Reset	2183
Portj_MTL_Q2_Interrupt_Status	Port j MTL Q2 Interrupt Status	11274_H+j *2000_H	PMj	PMj	Kernel Reset	2183
Portj_MTL_TxQ3_Operation_Mode	Port j MTL TxQ3 Operation Mode	11280_H+j *2000_H	PMj	PMj	Kernel Reset	2184
Portj_MTL_TxQ3_Underflow	Port j MTL TxQ3 Underflow	11284_H+j *2000_H	PMj	PMj	Kernel Reset	2186
Portj_MTL_TxQ3_Debug	Port j MTL TxQ3 Debug	11288_H+j *2000_H	PMj	PMj	Kernel Reset	2186
Portj_MTL_TC3_ETS_Control	Port j MTL TC3 ETS Control	11290_H+j *2000_H	PMj	PMj	Kernel Reset	2187
Portj_MTL_TC3_Quantum_Weight	Port j MTL TC3 Quantum Weight	11298_H+j *2000_H	PMj	PMj	Kernel Reset	2188
Portj_MTL_RxQ3_Operation_Mode	Port j MTL RxQ3 Operation Mode	$112C0_H$+ j*2000_H	PMj	PMj	Kernel Reset	2189
Portj_MTL_RxQ3_Missed_Pkt_Overflow_Cnt	Port j MTL RxQ3 Missed Pkt Overflow Cnt	$112C4_H$+ j*2000_H	PMj	PMj	Kernel Reset	2190
Portj_MTL_RxQ3_Debug	Port j MTL RxQ3 Debug	$112C8_H$+ j*2000_H	PMj	PMj	Kernel Reset	2191
Portj_MTL_RxQ3_Control	Port j MTL RxQ3 Control	$112CC_H$+ j*2000_H	PMj	PMj	Kernel Reset	2192

(table continues...)

Table 775 (continued) Registers overview - GETH (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
Portj_MTL_RxQ3_Flow_Control	Port j MTL RxQ3 Flow Control	$112D0_H$+j*2000_H	PMj	PMj	Kernel Reset	2193
Portj_MTL_Q3_Interrupt_Enable	Port j MTL Q3 Interrupt Enable	$112F0_H$+j*2000_H	PMj	PMj	Kernel Reset	2194
Portj_MTL_Q3_Interrupt_Status	Port j MTL Q3 Interrupt Status	$112F4_H$+j*2000_H	PMj	PMj	Kernel Reset	2194
Portj_MTL_TxQ4_Operation_Mode	Port j MTL TxQ4 Operation Mode	11300_H+j*2000_H	PMj	PMj	Kernel Reset	2195
Portj_MTL_TxQ4_Underflow	Port j MTL TxQ4 Underflow	11304_H+j*2000_H	PMj	PMj	Kernel Reset	2197
Portj_MTL_TxQ4_Debug	Port j MTL TxQ4 Debug	11308_H+j*2000_H	PMj	PMj	Kernel Reset	2197
Portj_MTL_TC4_ETS_Control	Port j MTL TC4 ETS Control	11310_H+j*2000_H	PMj	PMj	Kernel Reset	2198
Portj_MTL_TC4_Quantum_Weight	Port j MTL TC4 Quantum Weight	11318_H+j*2000_H	PMj	PMj	Kernel Reset	2199
Portj_MTL_RxQ4_Operation_Mode	Port j MTL RxQ4 Operation Mode	11340_H+j*2000_H	PMj	PMj	Kernel Reset	2200
Portj_MTL_RxQ4_Missed_Pkt_Overflow_Cnt	Port j MTL RxQ4 Missed Pkt Overflow Cnt	11344_H+j*2000_H	PMj	PMj	Kernel Reset	2201
Portj_MTL_RxQ4_Debug	Port j MTL RxQ4 Debug	11348_H+j*2000_H	PMj	PMj	Kernel Reset	2202
Portj_MTL_RxQ4_Control	Port j MTL RxQ4 Control	$1134C_H$+j*2000_H	PMj	PMj	Kernel Reset	2203
Portj_MTL_RxQ4_Flow_Control	Port j MTL RxQ4 Flow Control	11350_H+j*2000_H	PMj	PMj	Kernel Reset	2204
Portj_MTL_Q4_Interrupt_Enable	Port j MTL Q4 Interrupt Enable	11370_H+j*2000_H	PMj	PMj	Kernel Reset	2205
Portj_MTL_Q4_Interrupt_Status	Port j MTL Q4 Interrupt Status	11374_H+j*2000_H	PMj	PMj	Kernel Reset	2205
Portj_MTL_TxQ5_Operation_Mode	Port j MTL TxQ5 Operation Mode	11380_H+j*2000_H	PMj	PMj	Kernel Reset	2206
Portj_MTL_TxQ5_Underflow	Port j MTL TxQ5 Underflow	11384_H+j*2000_H	PMj	PMj	Kernel Reset	2208
Portj_MTL_TxQ5_Debug	Port j MTL TxQ5 Debug	11388_H+j*2000_H	PMj	PMj	Kernel Reset	2208
Portj_MTL_TC5_ETS_Control	Port j MTL TC5 ETS Control	11390_H+j*2000_H	PMj	PMj	Kernel Reset	2209

(table continues...)

Table 775 (continued) Registers overview - GETH (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
Portj_MTL_TC5_Quantum_Weight	Port j MTL TC5 Quantum Weight	11398_H+j*2000_H	PMj	PMj	Kernel Reset	2210
Portj_MTL_TC5_SendSlopeCredit	Port j MTL TC5 SendSlopeCredit	$1139C_H$+j*2000_H	PMj	PMj	Kernel Reset	2211
Portj_MTL_TC5_HiCredit	Port j MTL TC5 HiCredit	$113A0_H$+j*2000_H	PMj	PMj	Kernel Reset	2212
Portj_MTL_TC5_LoCredit	Port j MTL TC5 LoCredit	$113A4_H$+j*2000_H	PMj	PMj	Kernel Reset	2212
Portj_MTL_RxQ5_Operation_Mode	Port j MTL RxQ5 Operation Mode	$113C0_H$+j*2000_H	PMj	PMj	Kernel Reset	2213
Portj_MTL_RxQ5_Missed_Pkt_Overflow_Cnt	Port j MTL RxQ5 Missed Pkt Overflow Cnt	$113C4_H$+j*2000_H	PMj	PMj	Kernel Reset	2215
Portj_MTL_RxQ5_Debug	Port j MTL RxQ5 Debug	$113C8_H$+j*2000_H	PMj	PMj	Kernel Reset	2216
Portj_MTL_RxQ5_Control	Port j MTL RxQ5 Control	$113CC_H$+j*2000_H	PMj	PMj	Kernel Reset	2217
Portj_MTL_RxQ5_Flow_Control	Port j MTL RxQ5 Flow Control	$113D0_H$+j*2000_H	PMj	PMj	Kernel Reset	2217
Portj_MTL_Q5_Interrupt_Enable	Port j MTL Q5 Interrupt Enable	$113F0_H$+j*2000_H	PMj	PMj	Kernel Reset	2218
Portj_MTL_Q5_Interrupt_Status	Port j MTL Q5 Interrupt Status	$113F4_H$+j*2000_H	PMj	PMj	Kernel Reset	2219
Portj_MTL_TxQ6_Operation_Mode	Port j MTL TxQ6 Operation Mode	11400_H+j*2000_H	PMj	PMj	Kernel Reset	2220
Portj_MTL_TxQ6_Underflow	Port j MTL TxQ6 Underflow	11404_H+j*2000_H	PMj	PMj	Kernel Reset	2222
Portj_MTL_TxQ6_Debug	Port j MTL TxQ6 Debug	11408_H+j*2000_H	PMj	PMj	Kernel Reset	2222
Portj_MTL_TC6_ETS_Control	Port j MTL TC6 ETS Control	11410_H+j*2000_H	PMj	PMj	Kernel Reset	2223
Portj_MTL_TC6_Quantum_Weight	Port j MTL TC6 Quantum Weight	11418_H+j*2000_H	PMj	PMj	Kernel Reset	2224
Portj_MTL_TC6_SendSlopeCredit	Port j MTL TC6 SendSlopeCredit	$1141C_H$+j*2000_H	PMj	PMj	Kernel Reset	2225
Portj_MTL_TC6_HiCredit	Port j MTL TC6 HiCredit	11420_H+j*2000_H	PMj	PMj	Kernel Reset	2226
Portj_MTL_TC6_LoCredit	Port j MTL TC6 LoCredit	11424_H+j*2000_H	PMj	PMj	Kernel Reset	2226

(table continues...)

Table 775 (continued) Registers overview - GETH (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
Portj_MTL_RxQ6_Operation_Mode	Port j MTL RxQ6 Operation Mode	11440_H+j*2000_H	PMj	PMj	Kernel Reset	2227
Portj_MTL_RxQ6_Missed_Pkt_Overflow_Cnt	Port j MTL RxQ6 Missed Pkt Overflow Cnt	11444_H+j*2000_H	PMj	PMj	Kernel Reset	2229
Portj_MTL_RxQ6_Debug	Port j MTL RxQ6 Debug	11448_H+j*2000_H	PMj	PMj	Kernel Reset	2230
Portj_MTL_RxQ6_Control	Port j MTL RxQ6 Control	$1144C_H$+j*2000_H	PMj	PMj	Kernel Reset	2231
Portj_MTL_RxQ6_Flow_Control	Port j MTL RxQ6 Flow Control	11450_H+j*2000_H	PMj	PMj	Kernel Reset	2231
Portj_MTL_Q6_Interrupt_Enable	Port j MTL Q6 Interrupt Enable	11470_H+j*2000_H	PMj	PMj	Kernel Reset	2232
Portj_MTL_Q6_Interrupt_Status	Port j MTL Q6 Interrupt Status	11474_H+j*2000_H	PMj	PMj	Kernel Reset	2233
Portj_MTL_TxQ7_Operation_Mode	Port j MTL TxQ7 Operation Mode	11480_H+j*2000_H	PMj	PMj	Kernel Reset	2234
Portj_MTL_TxQ7_Underflow	Port j MTL TxQ7 Underflow	11484_H+j*2000_H	PMj	PMj	Kernel Reset	2236
Portj_MTL_TxQ7_Debug	Port j MTL TxQ7 Debug	11488_H+j*2000_H	PMj	PMj	Kernel Reset	2236
Portj_MTL_TC7_ETS_Control	Port j MTL TC7 ETS Control	11490_H+j*2000_H	PMj	PMj	Kernel Reset	2237
Portj_MTL_TC7_Quantum_Weight	Port j MTL TC7 Quantum Weight	11498_H+j*2000_H	PMj	PMj	Kernel Reset	2238
Portj_MTL_TC7_SendSlopeCredit	Port j MTL TC7 SendSlopeCredit	$1149C_H$+j*2000_H	PMj	PMj	Kernel Reset	2239
Portj_MTL_TC7_HiCredit	Port j MTL TC7 HiCredit	$114A0_H$+j*2000_H	PMj	PMj	Kernel Reset	2240
Portj_MTL_TC7_LoCredit	Port j MTL TC7 LoCredit	$114A4_H$+j*2000_H	PMj	PMj	Kernel Reset	2240
Portj_MTL_RxQ7_Operation_Mode	Port j MTL RxQ7 Operation Mode	$114C0_H$+j*2000_H	PMj	PMj	Kernel Reset	2241
Portj_MTL_RxQ7_Missed_Pkt_Overflow_Cnt	Port j MTL RxQ7 Missed Pkt Overflow Cnt	$114C4_H$+j*2000_H	PMj	PMj	Kernel Reset	2243
Portj_MTL_RxQ7_Debug	Port j MTL RxQ7 Debug	$114C8_H$+j*2000_H	PMj	PMj	Kernel Reset	2244

(table continues...)

Table 775 (continued) Registers overview - GETH (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
Portj_MTL_RxQ7_Control	Port j MTL RxQ7 Control	$114CC_H$+ j*2000_H	PMj	PMj	Kernel Reset	2245
Portj_MTL_RxQ7_Flow_Control	Port j MTL RxQ7 Flow Control	$114D0_H$+ j*2000_H	PMj	PMj	Kernel Reset	2245
Portj_MTL_Q7_Interrupt_Enable	Port j MTL Q7 Interrupt Enable	$114F0_H$+j *2000_H	PMj	PMj	Kernel Reset	2246
Portj_MTL_Q7_Interrupt_Status	Port j MTL Q7 Interrupt Status	$114F4_H$+j *2000_H	PMj	PMj	Kernel Reset	2247
Forward_Control	Forward Control	$1E000_H$	PG	PG	Kernel Reset	2248
Port_Flush_and_Loopback_Control	Port Flush and Loopback Control	$1E004_H$	PG	PG	Kernel Reset	2249
Port_0_Control	Port 0 Control	$1E00C_H$	PG	PG	Kernel Reset	2250
TXQ_Map_Port_0	TXQ Map Port 0	$1E010_H$	PG	PG	Kernel Reset	2251
RXC_Map_Port_0	RXC Map Port 0	$1E014_H$	PG	PG	Kernel Reset	2255
Port_1_Control	Port 1 Control	$1E018_H$	PG	PG	Kernel Reset	2258
TXQ_Map_Port_1	TXQ Map Port 1	$1E01C_H$	PG	PG	Kernel Reset	2259
RXC_Map_Port_1	RXC Map Port 1	$1E020_H$	PG	PG	Kernel Reset	2263
CSR_SW_Control	CSR SW Control	$1E100_H$	PG	PG	Kernel Reset	2266
Interrupt_Status	Interrupt Status	$1E104_H$	PG	PG	Kernel Reset	2267
Safety_Interrupt_Status	Safety Interrupt Status	$1E108_H$	PG	PG	Kernel Reset	2268
DMA_Port_Selection	DMA Port Selection	$1E110_H$	PG	PG	Kernel Reset	2271
PMT_Control	PMT Control	$1E114_H$	PG	PG	Kernel Reset	2271
DPP_FSM_Interrupt_Status	DPP FSM Interrupt Status	$1E150_H$	PG	PG	Kernel Reset	2272
AXI_Slave_DPP_Error_Address_Status	AXI Slave DPP Error Address Status	$1E154_H$	PG	PG	Kernel Reset	2276
FSM_Control	FSM Control	$1E158_H$	PG	PG	Kernel Reset	2276
FSM_ACT_Timer	FSM ACT Timer	$1E15C_H$	PG	PG	Kernel Reset	2278
SCSR_Control	SCSR Control	$1E164_H$	PG	PG	Kernel Reset	2279
SCSR_Parity_Status	SCSR Parity Status	$1E168_H$	PG	PG	Kernel Reset	2280
SCSR_Parity_Err_Count	SCSR Parity Err Count	$1E16C_H$	PG	PG	Kernel Reset	2281
Debug_Control	Debug Control	$1E208_H$	PG	PG	Kernel Reset	2281
ECC_Control	ECC Control	$1E2C0_H$	PG	PG	Kernel Reset	2282

(table continues...)

Table 775 (continued) Registers overview - GETH (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ECC_Error_Status_Capture_Control	ECC Err Sts RCtl	$1E2D0_H$	PG	PG	Kernel Reset	2283
ECC_Error_Address_Status	ECC Err Addr Status	$1E2D4_H$	PG	PG	Kernel Reset	2284
ECC_Error_Count_Status	ECC Err Cntr Status	$1E2D8_H$	PG	PG	Kernel Reset	2285
DPP_Control	DPP Control	$1E2E0_H$	PG	PG	Kernel Reset	2286
DPP_ECC_Error_Injection_Control	DPP ECC EIC	$1E2E4_H$	PG	PG	Kernel Reset	2288
DMA_Mode	DMA Mode	$1F000_H$	PG	PG	Kernel Reset	2289
DMA_SysBus_Mode	DMA SysBus Mode	$1F004_H$	PG	PG	Kernel Reset	2290
DMA_Interrupt_Status	DMA Interrupt Status	$1F008_H$	PG	PG	Kernel Reset	2292
AXI_Tx_AR_ACE_Control	AXI Tx AR ACE Control	$1F010_H$	PG	PG	Kernel Reset	2294
AXI_Rx_AW_ACE_Control	AXI Rx AW ACE Control	$1F018_H$	PG	PG	Kernel Reset	2295
AXI_TxRx_AWAR_ACE_Control	AXI TxRx AWAR ACE Control	$1F01C_H$	PG	PG	Kernel Reset	2296
DMA_Debug_Status0	DMA Debug Status0	$1F020_H$	PG	PG	Kernel Reset	2297
DMA_Debug_Status1	DMA Debug Status1	$1F024_H$	PG	PG	Kernel Reset	2298
DMA_Debug_Status3	DMA Debug Status3	$1F02C_H$	PG	PG	Kernel Reset	2298
DMA_Tx_EDMA_Control	DMA Tx EDMA Control	$1F040_H$	PG	PG	Kernel Reset	2299
DMA_Rx_EDMA_Control	DMA Rx EDMA Control	$1F044_H$	PG	PG	Kernel Reset	2300
AXI_LPI_Entry_Interval	AXI LPI Entry Interval	$1F050_H$	PG	PG	Kernel Reset	2302
DMA_TBS_CTRL0	DMA TBS CTRL0	$1F054_H$	PG	PG	Kernel Reset	2302
DMA_TBS_CTRL1	DMA TBS CTRL1	$1F058_H$	PG	PG	Kernel Reset	2303
DMA_TBS_CTRL2	DMA TBS CTRL2	$1F05C_H$	PG	PG	Kernel Reset	2303
DMA_TBS_CTRL3	DMA TBS CTRL3	$1F060_H$	PG	PG	Kernel Reset	2304
DMA_Safety_Interrupt_Status	DMA Safety Interrupt Status	$1F064_H$	PG	PG	Kernel Reset	2305
DMA_ECC_Interrupt_enable	DMA ECC Interrupt enable	$1F068_H$	PG	PG	Kernel Reset	2307

(table continues...)

Table 775 (continued) Registers overview - GETH (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
DMA_ECC_Interrupt_Status	DMA ECC Interrupt Status	$1F06C_H$	PG	PG	Kernel Reset	2308
DMA_DPP_Control	DMA DPP Control	$1F070_H$	PG	PG	Kernel Reset	2308
DMA_DPP_Interrupt_Status	DMA DPP Interrupt Status	$1F074_H$	PG	PG	Kernel Reset	2309
DMA_CH_Ind_Ctrl	DMA CH Ind Ctrl	$1F080_H$	PG	PG	Kernel Reset	2310
DMA_CH_Ind_Data	DMA CH Ind Data	$1F084_H$	PG	PG	Kernel Reset	2311
DMA_CH0_TxExtCfg	DMA Channel j TxExtCfg	$1F084_H$	PG	PG	Kernel Reset	2312
DMA_CH1_TxExtCfg	DMA Channel j TxExtCfg	$1F084_H$	PG	PG	Kernel Reset	2312
DMA_CH2_TxExtCfg	DMA Channel j TxExtCfg	$1F084_H$	PG	PG	Kernel Reset	2313
DMA_CH3_TxExtCfg	DMA Channel j TxExtCfg	$1F084_H$	PG	PG	Kernel Reset	2313
DMA_CH4_TxExtCfg	DMA Channel j TxExtCfg	$1F084_H$	PG	PG	Kernel Reset	2314
DMA_CH5_TxExtCfg	DMA Channel j TxExtCfg	$1F084_H$	PG	PG	Kernel Reset	2314
DMA_CH6_TxExtCfg	DMA Channel j TxExtCfg	$1F084_H$	PG	PG	Kernel Reset	2315
DMA_CH7_TxExtCfg	DMA Channel j TxExtCfg	$1F084_H$	PG	PG	Kernel Reset	2315
DMA_Sfty_Ind_Ctrl	DMA Sfty Ind Ctrl	$1F088_H$	PG	PG	Kernel Reset	2316
DMA_Sfty_Ind_Data	DMA Sfty Ind Data	$1F08C_H$	PG	PG	Kernel Reset	2318
DMA_AM_Sfty_Ctrl	DMA AM Sfty Ctrl	$1F08C_H$	PG	PG	Kernel Reset	2318
DMA_AM_Sfty_DPP_Intr_Sts	DMA AM Sfty DPP Intr Sts	$1F08C_H$	PG	PG	Kernel Reset	2319
DMA_AM_Sfty_ECC_Intr_Sts	DMA AM Sfty ECC Intr Sts	$1F08C_H$	PG	PG	Kernel Reset	2320
DMA_AM_Sfty_Intr_En	DMA AM Sfty Intr En	$1F08C_H$	PG	PG	Kernel Reset	2320
DMA_AM_Sfty_RDC_Dbg_Ctrl	DMA AM Sfty RDC Dbg Ctrl	$1F08C_H$	PG	PG	Kernel Reset	2321
DMA_AM_Sfty_RDC_Dbg_Data	DMA AM Sfty RDC Dbg Data	$1F08C_H$	PG	PG	Kernel Reset	2322

(table continues...)

Table 775 (continued) Registers overview - GETH (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
DMA_AM_Sfty_RDC_ECC_Err_Cntr_Sts	DMA AM Sfty RDC ECC Err Cntr Sts	$1F08C_H$	PG	PG	Kernel Reset	2322
DMA_AM_Sfty_WRC_Dbg_Ctrl	DMA AM Sfty WRC Dbg Ctrl	$1F08C_H$	PG	PG	Kernel Reset	2323
DMA_AS_Sfty_Ctrl	DMA AS Sfty Ctrl	$1F08C_H$	PG	PG	Kernel Reset	2323
DMA_AS_Sfty_DPP_Intr_Sts	DMA AS Sfty DPP Intr Sts	$1F08C_H$	PG	PG	Kernel Reset	2324
DMA_AS_Sfty_ECC_Intr_Sts	DMA AS Sfty ECC Intr Sts	$1F08C_H$	PG	PG	Kernel Reset	2325
DMA_AS_Sfty_Intr_En	DMA AS Sfty Intr En	$1F08C_H$	PG	PG	Kernel Reset	2325
DMA_AS_Sfty_RAC_Dbg_Addr	DMA AS Sfty RAC Dbg Addr	$1F08C_H$	PG	PG	Kernel Reset	2326
DMA_AS_Sfty_RAC_Dbg_Ctrl	DMA AS Sfty RAC Dbg Ctrl	$1F08C_H$	PG	PG	Kernel Reset	2326
DMA_AS_Sfty_WAC_Dbg_Addr	DMA AS Sfty WAC Dbg Addr	$1F08C_H$	PG	PG	Kernel Reset	2327
DMA_AS_Sfty_WAC_Dbg_Ctrl	DMA AS Sfty WAC Dbg Ctrl	$1F08C_H$	PG	PG	Kernel Reset	2328
DMA_AS_Sfty_WDC_Dbg_Ctrl	DMA AS Sfty WDC Dbg Ctrl	$1F08C_H$	PG	PG	Kernel Reset	2328
DMA_AS_Sfty_WDC_Dbg_Data	DMA AS Sfty WDC Dbg Data	$1F08C_H$	PG	PG	Kernel Reset	2329
DMA_AS_Sfty_WDC_ECC_Err_Cntr_Sts	DMA AS Sfty WDC ECC Err Cntr Sts	$1F08C_H$	PG	PG	Kernel Reset	2329
DMA_Sfty_Dbg_EIC	DMA Sfty Dbg EIC	$1F08C_H$	PG	PG	Kernel Reset	2330
DMA_CHj_Control	DMA Channel j Control	$1F100_H$+j *80_H	PCHj	PCHj	Kernel Reset	2331
DMA_CHj_Tx_Control	DMA Channel j Tx Control	$1F104_H$+j *80_H	PCHj	PCHj	Kernel Reset	2332
DMA_CHj_Rx_Control	DMA Channel j Rx Control	$1F108_H$+j *80_H	PCHj	PCHj	Kernel Reset	2334
DMA_CHj_Slot_Function_Control_Status	DMA Channel j Slot Function Control Status	$1F10C_H$+ j*80_H	PCHj	PCHj	Kernel Reset	2337
DMA_CHj_TxDesc_List_LAddress	DMA Channel j TxDesc List LAddress	$1F114_H$+j *80_H	PCHj	PCHj	Kernel Reset	2338

(table continues...)

Table 775 (continued) Registers overview - GETH (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
DMA_CHj_RxDesc_List_LAddress	DMA Channel j RxDesc List LAddress	$1F11C_H$+j*80_H	PCHj	PCHj	Kernel Reset	2338
DMA_CHj_TxDesc_Tail_LPointer	DMA Channel j TxDesc Tail LPointer	$1F124_H$+j*80_H	PCHj	PCHj	Kernel Reset	2339
DMA_CHj_RxDesc_Tail_LPointer	DMA Channel j RxDesc Tail LPointer	$1F12C_H$+j*80_H	PCHj	PCHj	Kernel Reset	2339
DMA_CHj_Tx_Control2	DMA Channel j Tx Control2	$1F130_H$+j*80_H	PCHj	PCHj	Kernel Reset	2340
DMA_CHj_Rx_Control2	DMA Channel j Rx Control2	$1F134_H$+j*80_H	PCHj	PCHj	Kernel Reset	2341
DMA_CHj_Interrupt_Enable	DMA Channel j Interrupt Enable	$1F138_H$+j*80_H	PCHj	PCHj	Kernel Reset	2342
DMA_CHj_Rx_Interrupt_Watchdog_Timer	DMA Channel j Rx Interrupt Watchdog Timer	$1F13C_H$+j*80_H	PCHj	PCHj	Kernel Reset	2343
DMA_CHj_Current_App_TxDesc_L	DMA Channel j Current App TxDesc L	$1F144_H$+j*80_H	PCHj	PCHj	Kernel Reset	2345
DMA_CHj_Current_App_RxDesc_L	DMA Channel j Current App RxDesc L	$1F14C_H$+j*80_H	PCHj	PCHj	Kernel Reset	2345
DMA_CHj_Current_App_TxBuffer_L	DMA Channel j Current App TxBuffer L	$1F154_H$+j*80_H	PCHj	PCHj	Kernel Reset	2346
DMA_CHj_Current_App_RxBuffer_L	DMA Channel j Current App RxBuffer L	$1F15C_H$+j*80_H	PCHj	PCHj	Kernel Reset	2346
DMA_CHj_Status	DMA Channel j Status	$1F160_H$+j*80_H	PCHj	PCHj	Kernel Reset	2347
DMA_CHj_Debug_Status	DMA Channel j Debug Status	$1F164_H$+j*80_H	PCHj	PCHj	Kernel Reset	2350
DMA_CHj_Desc_Mem_Cache_Fill_Level	DMA Channel j Desc Mem Cache Fill Level	$1F168_H$+j*80_H	PCHj	PCHj	Kernel Reset	2352
DMA_CHj_Miss_Packet_Cnt	DMA Channel j Miss Packet Cnt	$1F16C_H$+j*80_H	PCHj	PCHj	Kernel Reset	2353
DMA_CHj_Tx_Data_Xfer_Ring_Offset	DMA Channel j Tx Data Xfer Ring Offset	$1F170_H$+j*80_H	PCHj	PCHj	Kernel Reset	2354
DMA_CHj_Rx_Data_Xfer_Ring_Offset	DMA Channel j Rx Data Xfer Ring Offset	$1F174_H$+j*80_H	PCHj	PCHj	Kernel Reset	2354
DMA_CHj_Tx_Desc_Write_Ring_Offset	DMA Channel j Tx Desc Write Ring Offset	$1F178_H$+j*80_H	PCHj	PCHj	Kernel Reset	2355

(table continues...)

Table 775 (continued) Registers overview - GETH (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
DMA_CHj_Rx_Desc_Write_Ring_Offset	DMA Channel j Rx Desc Write Ring Offset	$1F17C_H$+ j*80_H	PCHj	PCHj	Kernel Reset	2355

14.6.3 Clock control register

The clock control register CLC allows the programmer to adapt the functionality and power consumption of the module to the requirements of the application.

Register CLC controls the module clock signal and the reactivity to the sleep signal.

CLC Offset address: 00000_H

Clock control register Application Reset value: $0000\ 0003_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

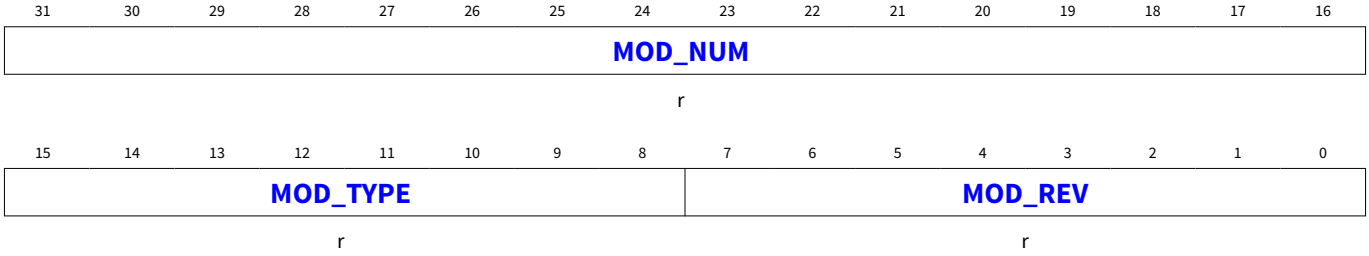
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												EDIS	0	DISS	DISR
r												rw	r	rh	rw

Field	Bits	Type	Description
DISR	0	rw	**Module disable request bit** Used for enable and disable control of the module. 0_B On request: enable the module clock 1_B Off request: stop the module clock
DISS	1	rh	**Module disable status bit** 0_B Module clock is enabled 1_B Module is not clocked
EDIS	3	rw	**Sleep mode enable control** GETH module does not support reaction to sleep mode. 0_B Not applicable 1_B Not applicable
0	2, 31:4	r	**Reserved, write 0, read as 0**

14.6.4 Module identification register

ID Offset address: 00008_H

Module identification register PowerOn Reset value: $00F1\ C001_H$

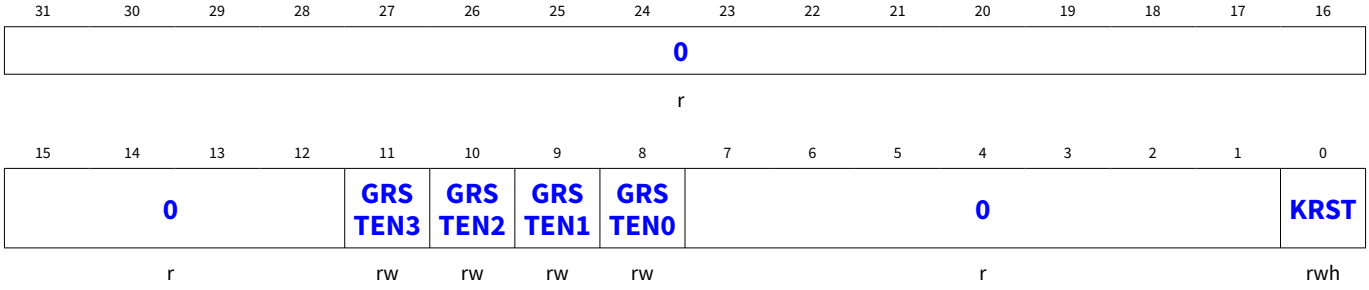


Field	Bits	Type	Description
MOD_REV	7:0	r	**Module revision** Indicates the revision number of the implementation.
MOD_TYPE	15:8	r	**Module type** This internal marker is fixed to $C0_H$.
MOD_NUM	31:16	r	**Module number** Indicates the module identification number.

14.6.5 Reset control register A

RST_CTRLA Offset address: $0000C_H$

Reset control register A Application Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
KRST	0	rwh	**Kernel reset** Request a kernel reset. The requested reset is executed if CTRLA.KRST=1 and CTRLB.KRST=1. KRST is cleared after the kernel reset was executed. 0_B No action 1_B A kernel reset was requested
GRSTENx (x=0-3)	x+8	rw	**Enable for global module reset group x** These bit fields have no functionality, KRST is executed if CTRLA.KRST=1 and CTRLB.KRST=1 independently from GRSTEN bits. 0_B Global module reset group x does not have any effect 1_B Global module reset group x results in a kernel reset
0	7:1, 31:12	r	**Reserved** Read as 0; should be written with 0.

14.6.6 Reset Control Register B

RST_CTRLB Offset address: 00010_H

Reset Control Register B Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
STATCLR								0							
w								r							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															KRST
r															rwh

Field	Bits	Type	Description
KRST	0	rwh	**Kernel reset** Request a kernel reset. The requested reset is executed if CTRLA.KRST=1 and CTRLB.KRST=1. KRST is cleared after the kernel reset was executed. 0_B No action 1_B A kernel reset was requested
STATCLR	31	w	**Kernel reset status clear** Clears all status bits in RST_STAT when 1 is written. Read will return 0. 0_B No action 1_B Write with ´1´ clears bit STAT.KRST .
0	30:1	r	**Reserved** Read as 0; should be written with 0.

14.6.7 Reset status register

The reset status register contains the status bits for kernel reset (KRST) and the global module reset groups (GRSTx).

RST_STAT Offset address: 00014_H

Reset status register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				GRST3	GRST2	GRST1	GRST0	0							KRST
r				rh	rh	rh	rh	r							rh

Field	Bits	Type	Description
KRST	0	rh	**Kernel reset status** Indicates an executed kernel reset. RST_STAT.KRST is set after the execution of a kernel reset in the same clock cycle in which the reset bits are cleared. Clear KRST by setting bit STATCLR in register RST_CTRLB.
GRSTx (x=0-3)	x+8	rh	**Status for global module reset group x** 0_B Reset was not triggered by global reset group x 1_B Reset was triggered by global reset group x
0	7:1, 31:12	r	**Reserved** Read as 0.

14.6.8 Clock control register for MAC

Register used to turn MAC Cores on and off individually.

MACEN Offset address: 00018_H

Clock control register for MAC Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0														MAC1	MAC0
r														rw	rw

Field	Bits	Type	Description
MAC0	0	rw	**Used for enabling and disabling MAC0** Used for enabling and disabling MAC0 0_B Off request: Disable MAC0 1_B On Request: Enable MAC0
MAC1	1	rw	**Used for enabling and disabling MAC1** Used for enabling and disabling MAC1 0_B Off request: Disable MAC1 1_B On Request: Enable MAC1
0	31:2	r	**Reserved** Read as 0; should be written with 0.

14.6.9 PROT register safe endinit

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner and in some PROT states, the secure master to update the PROT state as described in the PROT mechanism.

The PROTSE register controls lock and unlock of the local Safe Endinit (SE) protected control registers.

PROTSE Offset address: 00024_H

PROT register safe endinit Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OWEN	ODEF	TAGID						PRSEN	PRS			VMEN	VM		
w	rw	rw						rw	rw			rw	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												SWEN	STATE		
r												w	rwh		

Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
VM	18:16	rw	**Virtual machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual machine definition enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.

(table continues...)

(continued)

Field	Bits	Type	Description
PRSEN	23	rw	**Protection set definition enable for PROT owner** 0_B **Disable**: PRS is not part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner
OWEN	31	w	**Owner write enable** 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; should be written with 0.

Table 776 Access mode restrictions of PROTSE sorted by descending priority

Mode name	Access mode		Description
write 1 to .**SWEN**	rwh	STATE	
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

14.6.10 Write access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

ACCENGLOBAL_WRA Offset address: 00040_H

Write access enable register A Application Reset value: $1000\ 0003_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Write access enable for TAG ID q** This bit enables write access to the access protected region for transactions with the master TAG ID q 0_B Disabled for write access 1_B Enabled for write access

14.6.11 Write access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

ACCENGLOBAL_WRB Offset address: 00044_H

Write access enable register B Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN63	EN62	EN61	EN60	EN59	EN58	EN57	EN56	EN55	EN54	EN53	EN52	EN51	EN50	EN49	EN48
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN47	EN46	EN45	EN44	EN43	EN42	EN41	EN40	EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-63)	q-32	rw	**Write access enable for TAG ID q** This bit enables write access to the access protected region for transactions with the master TAG ID q 0_B Disabled for write access 1_B Enabled for write access

14.6.12 Read access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

ACCENGLOBAL_RDA Offset address: 00048_H

Read access enable register A Application Reset value: $FFFF\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Read access enable for TAG ID q** This bit enables read access to the access protected region for transactions with the master TAG ID q 0_B Disabled for read access 1_B Enabled for read access

14.6.13 Read access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

ACCENGLOBAL_RDB Offset address: $0004C_H$

Read access enable register B Application Reset value: $FFFF\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN63	EN62	EN61	EN60	EN59	EN58	EN57	EN56	EN55	EN54	EN53	EN52	EN51	EN50	EN49	EN48
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN47	EN46	EN45	EN44	EN43	EN42	EN41	EN40	EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-63)	q-32	rw	**Read access enable for TAG ID q** This bit enables read access to the access protected region for transactions with the master TAG ID q 0_B Disabled for read access 1_B Enabled for read access

14.6.14 VM access enable register

This register defines which virtual machine encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each VMx.

ACCENGLOBAL_VM Offset address: 00050_H

VM access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for VM ID q** This bit enables read access to the access protected region for transactions with the VM ID q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for VM ID q** This bit enables write access to the access protected region for transactions with the VM ID q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

14.6.15 PRS access enable register

This register defines which protection register set encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each PRSx.

ACCENGLOBAL_PRS Offset address: 00054_H

PRS access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for PRS q** This bit enables read access to the access protected region for transactions with the PRS q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for PRS q** This bit enables write access to the access protected region for transactions with the PRS q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

14.6.16 MAC y Write access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

ACCENMACy_WRA (y=0-1) Offset address: 00060_H+y*20_H

MAC y Write access enable register A Application Reset value: $1000\ 0003_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Write access enable for TAG ID q** This bit enables write access to the access protected region for transactions with the master TAG ID q 0_B Disabled for write access 1_B Enabled for write access

14.6.17 MAC y Write access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

ACCENMACy_WRB (y=0-1) Offset address: 00064_H+y*20_H

MAC y Write access enable register B Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN63	EN62	EN61	EN60	EN59	EN58	EN57	EN56	EN55	EN54	EN53	EN52	EN51	EN50	EN49	EN48
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN47	EN46	EN45	EN44	EN43	EN42	EN41	EN40	EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-63)	q-32	rw	**Write access enable for TAG ID q** This bit enables write access to the access protected region for transactions with the master TAG ID q 0_B Disabled for write access 1_B Enabled for write access

14.6.18 MAC y Read access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

ACCENMACy_RDA (y=0-1) Offset address: 00068_H+y*20_H

MAC y Read access enable register A Application Reset value: FFFF FFFF$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Read access enable for TAG ID q** This bit enables read access to the access protected region for transactions with the master TAG ID q 0_B Disabled for read access 1_B Enabled for read access

14.6.19 MAC y Read access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

ACCENMACy_RDB (y=0-1) Offset address: $0006C_H+y*20_H$

MAC y Read access enable register B Application Reset value: FFFF FFFF$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN63	EN62	EN61	EN60	EN59	EN58	EN57	EN56	EN55	EN54	EN53	EN52	EN51	EN50	EN49	EN48
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN47	EN46	EN45	EN44	EN43	EN42	EN41	EN40	EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-63)	q-32	rw	**Read access enable for TAG ID q** This bit enables read access to the access protected region for transactions with the master TAG ID q 0_B Disabled for read access 1_B Enabled for read access

14.6.20 MAC y VM access enable register

This register defines which virtual machine encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each VMx.

ACCENMACy_VM (y=0-1) Offset address: 00070_H+y*20_H

MAC y VM access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for VM ID q** This bit enables read access to the access protected region for transactions with the VM ID q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for VM ID q** This bit enables write access to the access protected region for transactions with the VM ID q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

14.6.21 MAC y PRS access enable register

This register defines which protection register set encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each PRSx.

ACCENMACy_PRS (y=0-1) Offset address: 00074_H+y*20_H

MAC y PRS access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for PRS q** This bit enables read access to the access protected region for transactions with the PRS q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for PRS q** This bit enables write access to the access protected region for transactions with the PRS q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

14.6.22 CH y Write access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

ACCENCHy_WRA (y=0-7) Offset address: $000A0_H+y*20_H$

CH y Write access enable register A Application Reset value: $1000\ 0003_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Write access enable for TAG ID q** This bit enables write access to the access protected region for transactions with the master TAG ID q 0_B Disabled for write access 1_B Enabled for write access

14.6.23 CH y Write access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

ACCENCHy_WRB (y=0-7) Offset address: $000A4_H+y*20_H$

CH y Write access enable register B Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN63	EN62	EN61	EN60	EN59	EN58	EN57	EN56	EN55	EN54	EN53	EN52	EN51	EN50	EN49	EN48
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN47	EN46	EN45	EN44	EN43	EN42	EN41	EN40	EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-63)	q-32	rw	**Write access enable for TAG ID q** This bit enables write access to the access protected region for transactions with the master TAG ID q 0_B Disabled for write access 1_B Enabled for write access

14.6.24 CH y Read access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

ACCENCHy_RDA (y=0-7) Offset address: $000A8_H+y*20_H$

CH y Read access enable register A Application Reset value: FFFF $FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Read access enable for TAG ID q** This bit enables read access to the access protected region for transactions with the master TAG ID q 0_B Disabled for read access 1_B Enabled for read access

14.6.25 CH y Read access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

ACCENCHy_RDB (y=0-7) Offset address: $000AC_H+y*20_H$

CH y Read access enable register B Application Reset value: FFFF $FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN63	EN62	EN61	EN60	EN59	EN58	EN57	EN56	EN55	EN54	EN53	EN52	EN51	EN50	EN49	EN48
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN47	EN46	EN45	EN44	EN43	EN42	EN41	EN40	EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-63)	q-32	rw	**Read access enable for TAG ID q** This bit enables read access to the access protected region for transactions with the master TAG ID q 0_B Disabled for read access 1_B Enabled for read access

14.6.26 CH y VM access enable register

This register defines which virtual machine encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each VMx.

ACCENCHy_VM (y=0-7) Offset address: $000B0_H+y*20_H$

CH y VM access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for VM ID q** This bit enables read access to the access protected region for transactions with the VM ID q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for VM ID q** This bit enables write access to the access protected region for transactions with the VM ID q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

14.6.27 CH y PRS access enable register

This register defines which protection register set encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each PRSx.

ACCENCHy_PRS (y=0-7) Offset address: $000B4_H+y*20_H$

CH y PRS access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for PRS q** This bit enables read access to the access protected region for transactions with the PRS q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for PRS q** This bit enables write access to the access protected region for transactions with the PRS q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

14.6.28 Port j MAC Tx Configuration

The MAC Transmit Configuration register establishes the operating mode of the MAC transmitter.

Portj_MAC_Tx_Configuration (j=0-1) Offset address: $10000_H+j*2000_H$

Port j MAC Tx Configuration Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SS			0	0	0	0	0	0	SARC			0	0	0	JD
rw			r	r	r	r	r	r	rw			r	r	r	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		LUD	TC	IFP	IPG			ISR				ISM	0	DDIC	TE
r		rw	rw	rw	rw			rw				rw	r	rw	rw

Field	Bits	Type	Description
TE	0	rw	**Transmitter Enable** When this bit is set, the Tx state machine of the MAC is enabled for transmission on the GMII or XGMII interface. When this bit is reset, the MAC Tx state machine is disabled after it completes the transmission of the current packet. The Tx state machine does not transmit any more packets.
DDIC	1	rw	**Disable DIC Algorithm** When this bit is set, the Deficit Idle Count (DIC) algorithm is disabled and DWC_xgmac always has a minimum IFG of 12 bytes between transmitted packets. When reset to zero, the DIC is activated and the minimum IFG varies between 9 and 15 bytes.
ISM	3	rw	**IFG Stretch Mode** When this bit is set, DWC_xgmac enforces extended inter-packet spacing (one extra octet of IDLE for every ISR (IFG Stretch Ratio) number of bytes) to support less-than-10-Gbps data rates. When this bit is reset, inter-packet gap extension is not performed.
ISR	7:4	rw	**IFG Stretch Ratio** This value controls the number of bytes in a transmitted packet for which one octet of IDLE is added to the IFG. To operate DWC_xgmac for 10GBASE-W (WIS layer), the value must be 13 (decimal). Note that 0 is not a valid value and must not be written. Before enabling ISM (IFG Stretch Mode) bit, this ISR number must be programmed to a non-zero value.

(table continues...)

(continued)

Field	Bits	Type	Description
IPG	10:8	rw	**Inter-Packet Gap** These bits control the minimum IPG between packets during transmission. When IFP is cleared, the minimum IPG between transmitted packets is reduced in both GMII and XGMII mode. The corresponding IPG values are given as follows. - 3'b000: 96 bit times - 3'b001: 88 bit times - 3'b010: 80 bit times - 3'b011: 72 bit times - 3'b100: 64 bit times - 3'b101: 3'b111: Reserved In the half-duplex mode, the minimum IPG can be configured only for 64-bit times (IPG = 100). Lower values are not considered. When a JAM pattern is transmitted because of backpressure activation, the MAC does not consider the minimum IPG. When IFP is set, the minimum IPG is increased in steps of 32 bits as follows. In 10G (XGMII) mode, when IFP is set, it overrides the functions of DDIC and ISM bits. The EIPG bits (in MAC_Extended_Configuration register) along with IPG bits gives a 10-bit value to control the required IPG as follows: - {EIPG,IPG}: Minimum Inter-Packet Gap - 10'h000: 96 bit times - 10'h001: 128 bit times - 10'h002: 160 bit times - 10'h003: 192 bit times - 10'h004: 224 bit times - ... - 10'h3FF: 32832 bit times
IFP	11	rw	**IPG Control** When this bit is set, the minimum IPG is increased in steps of 32 bits from the default 96 bits, depending on the programming of the IPG field. In the XGMII mode, this bit has higher priority than the ISM bit. When this bit is reset, the minimum IFG is decreased in steps of 8 bits from the default 96 bits, depending on the programming of the IPG field.
TC	12	rw	**Transmit Configuration in RGMII** When set, this bit enables the transmission of duplex mode, link speed, and link up or down information to the PHY in the RGMII, port. When this bit is reset, no such information is driven to the PHY. - 1'b0: Disable Transmit Configuration in RGMII - 1'b1: Enable Transmit Configuration in RGMII

(table continues...)

(continued)

Field	Bits	Type	Description
LUD	13	rw	**Link Up or Down** This bit indicates whether the link is up or down during transmission of configuration in the RGMII interface. - 1'b0: Link down - 1'b1: Link up
JD	16	rw	**Jabber Disable** When this bit is set, the DWC_xgmac disables the jabber timer on the transmitter. Transmission of up to 16383-byte packets is supported. When this bit is reset, DWC_xgmac cuts off the transmitter if the application sends more than 2048 bytes of data (10240 bytes if JE (in MAC_Rx_Configuration register is set high) during transmission.
SARC	22:20	rw	**Source Address Insertion or Replacement Control** This field controls the source address insertion or replacement for all transmitted packets. Bit 22 specifies which MAC Address register (0 or 1) is used for source address insertion or replacement based on the values of Bits[21:20]: 2'b0x: The SA insertion-replacement is controlled by SAIC field (Bits[25:23]) of TDES3 Normal Descriptor 2'b10: - If Bit 22 is set to 0, the MAC inserts the content of the MAC Address 0 registers (MAC_Address0_High and MAC_Address0_Low) in the SA field of all transmitted packets. - If Bit 22 is set to 1 the MAC inserts the content of the MAC Address 1 registers (MAC_Address1_High and MAC_Address1_Low) in the SA field of all transmitted packets. 2'b11: - If Bit 22 is set to 0, the MAC replaces the content of the MAC Address 0 registers (MAC_Address0_High and MAC_Address0_Low) in the SA field of all transmitted packets. - If Bit 22 is set to 1 and the MAC Address Register 1 is enabled, the MAC replaces the content of the MAC Address 1 registers (MAC_Address1_High and MAC_Address1_Low) in the SA field of all transmitted packets. Note: - Changes to this field take effect only on the start of a packet. If you write to this register field when a packet is being transmitted, only the subsequent packet can use the updated value, that is, the current packet does not use the updated value.

(table continues...)

(continued)

Field	Bits	Type	Description
SS	31:29	rw	**Speed Selection** Software programs these bits to instruct the MAC to operate in one of the following possible speeds: - 3'b000 Reserved - 3'b001 Reserved - 3'b010 2.5G GMII - 3'b011 1G GMII - 3'b100 100M MII - 3'b101 5G XGMII - 3'b110 2.5G XGMII - 3'b111 10M MII These bits must be programmed only once; after a hardware reset and before the transmitter and receiver are enabled (by bit 0 of this register and bit 0 of the MAC_Rx_Configuration register). Refer to device specific configuration table for the speed modes supported by GETH in corresponding device.
0	2, 15:14, 17, 18, 19, 23, 24, 25, 26, 27, 28	r	**Reserved** Read as 0; should be written with 0.

14.6.29 Port j MAC Rx Configuration

The MAC Receive Configuration register establishes the operating mode of the MAC receiver.

Portj_MAC_Rx_Configuration (j=0-1) Offset address: $10004_H+j*2000_H$

Port j MAC Rx Configuration Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	GPSL													
r	r	rw													

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	HDSMS			S2KP	LM	IPC	JE	WD	GPSLCE	USP	SPEN	DCRCC	CST	ACS	RE
r	rw			rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RE	0	rw	**Receiver Enable** When this bit is set, the Rx state machine of the MAC is enabled for receiving packets from the GMII or XGMII interface. When this bit is reset, the MAC Rx state machine is disabled after it completes the reception of the current packet. The Rx state machine does not receive any more packets from the GMII or XGMII interface.
ACS	1	rw	**Automatic Pad or CRC Stripping** When this bit is set, the MAC strips the Pad or FCS field on the incoming packets, only if the value of the length field is less than 1536 bytes. All received packets with length field greater than or equal to 1536 bytes are passed to the application without stripping the Pad or FCS field. When this bit is reset, the MAC passes all the incoming packets to the application, without any modification.
CST	2	rw	**CRC stripping for Type packets** When this bit is set, the last four bytes (FCS) of all packets of Ether type (field greater than 1536) are stripped and dropped before forwarding the packet to the application. When this bit is reset,CRC field of Ethernet packet is not stripped.
DCRCC	3	rw	**Disable CRC Checking for Received Packets** When this bit is set, the MAC receiver does not check the CRC field in the received packets. When this bit is reset, the MAC receiver always checks the CRC field in the received packets.
SPEN	4	rw	**Slow Protocol Detection Enable** When this bit is set, MAC processes the Slow Protocol packets (Ether Type 0x8809) and provides the Rx status. The MAC discards the Slow Protocol packets with invalid sub-types. When this bit is reset, the MAC forwards all error-free Slow Protocol packets to the application. The MAC considers such packets as normal Type packets.
USP	5	rw	**Unicast Slow Protocol Packet Detect** When this bit is set, the MAC detects the Slow Protocol packets with unicast address of the station specified in the MAC_Address0_High and MAC_Address0_Low registers. The MAC also detects the Slow Protocol packets with the Slow Protocols multicast address (01-80-C2-00-00-02). When this bit is reset, the MAC detects only Slow Protocol packets with the Slow Protocol multicast address specified in the IEEE 802.3-2018, Section 5.

(table continues...)

(continued)

Field	Bits	Type	Description
GPSLCE	6	rw	**Giant Packet Size Limit Control Enable** When this bit is set, the MAC considers the value in GPSL field in MAC_Rx_Configuration register to declare a received packet as Giant packet. This field must be programmed to more than 1518 bytes. Otherwise, the MAC considers 1518 bytes as giant packet limit. When this bit is reset, the MAC considers a received packet as Giant packet when its size is greater than 1518 bytes (1522 bytes for tagged packet). The watchdog timeout limit, Jumbo Packet Enable and 2K Packet Enable have higher precedence over this bit, that is the MAC considers a received packet as Giant packet when its size is greater than 9018 bytes (9022 bytes for tagged packet) with Jumbo Packet Enabled and greater than 2000 bytes with 2K Packet Enabled. The watchdog timeout, if enabled, terminates the received packet when watchdog limit is reached. Therefore, the programmed giant packet limit must be less than the watchdog limit to get the giant packet status.
WD	7	rw	**Watchdog Disable** When this bit is set, the MAC disables the watchdog timer on the receiver. The MAC can receive packets of up to 16383 bytes. When this bit is reset, the MAC does not allow more than 2048 bytes (10240 if JE is set high) of the packet being received. The MAC cuts off any bytes received after 2048 bytes.
JE	8	rw	**Jumbo Packet Enable** When this bit is set, the MAC allows jumbo packets of 9018 bytes (9022 bytes for VLAN tagged packets) without reporting a giant packet error in the Rx packet status.
IPC	9	rw	**Checksum Offload** When set, this bit enables the IPv4 header checksum checking and IPv4 or IPv6 TCP, UDP, or ICMP payload checksum checking. When this bit is reset, the COE function in the receiver is disabled.
LM	10	rw	**Loopback Mode** When this bit is set, the MAC operates in the loopback mode at GMII or XGMII. The (X)GMII Rx clock input is required for the loopback to work properly. This is because the Tx clock is not internally looped back.
S2KP	11	rw	**IEEE 802.3as Support for 2K Packets** When this bit is set, the MAC considers all packets with up to 2000 bytes length as normal packets. When the JE bit is not set, the MAC considers all the received packets of size more than 2K bytes as Giant packets. When this bit is reset and the JE bit is not set, the MAC considers all the received packets of size more than 1518 bytes (1522 bytes for tagged) as giant packets. Note: When the JE bit is set, setting this bit has no effect on the giant packet status.

(table continues...)

(continued)

Field	Bits	Type	Description
HDSMS	14:12	rw	**Maximum Size for Splitting the Header Data** These bits indicate the maximum header size allowed for splitting the header data in the received packet: - 3'b000: 64 bytes - 3'b001: 128 bytes - 3'b010: 256 bytes - 3'b011: 512 bytes - 3'b100: 1023 bytes - 3'b101-3'b111: Reserved
GPSL	29:16	rw	**Giant Packet Size Limit** If the received packet size is greater than the value programmed in this field in units of bytes, the MAC declares the received packet as Giant packet. The value programmed in this field must be greater than or equal to 1518 bytes. Any other programmed value is considered as 1518 bytes. For VLAN tagged packets, the MAC adds 4 bytes to the programmed value. The value in this field is applicable when the GPSLCE bit is set in MAC_Rx_Configuration register.
0	15, 30, 31	r	**Reserved** Read as 0; should be written with 0.

14.6.30 Port j MAC Packet Filter

The MAC Packet Filter register contains the filter controls for receiving packets. Some of the controls from this register go to the address check block of the MAC which performs the first level of address filtering. The second level of filtering is performed on the incoming packet based on other controls such as Pass Bad Packets and Pass Control Packets.

Portj_MAC_Packet_Filter (j=0-1) Offset address: $10008_H+j*2000_H$

Port j MAC Packet Filter Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RA	0								0	DNTU	IPFE	0			VTFE
rw	r								r	rw	rw	r			rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			DHLFRS		0	SAF	SAIF	PCF		DBF	PM	DAIF	0	0	PR
r			rw		r	rw	rw	rw		rw	rw	rw	r	r	rw

Field	Bits	Type	Description
PR	0	rw	**Promiscuous Mode** When this bit is set, the Address Filtering module passes all incoming packets irrespective of the destination or source address. The MAC clears the SA or DA Filter Fail status bits of the Rx Status Word when PR is set.
DAIF	3	rw	**DA Inverse Filtering** When this bit is set, the Address Check block operates in inverse filtering mode for the DA address comparison for both unicast and multicast packets. When this bit is reset, normal filtering of packets is performed.
PM	4	rw	**Pass All Multicast** When this bit is set, it indicates that all the received packets with a multicast destination address (first bit in the destination address field is 1) are passed. When this bit is reset, filtering of multicast packet is done
DBF	5	rw	**Disable Broadcast Packets** When this bit is set, the AFM module blocks all incoming broadcast packets. In addition, it overrides all other filter settings. When this bit is reset, the AFM module passes all the received broadcast packets.
PCF	7:6	rw	**Pass Control Packets** These bits control the forwarding of all control packets (including unicast and multicast Pause packets). - 00: The MAC filters all control packets from reaching the application. - 01: The MAC forwards all control packets except Pause packets to the application even if they fail the Address filter. - 10: The MAC forwards all control packets to the application even if they fail the Address filter. - 11: The MAC forwards the control packets that pass the Address filter.
SAIF	8	rw	**SA Inverse Filtering** When this bit is set, the Address Check block operates in the inverse filtering mode for SA address comparison. If the SA of a packet matches the values programmed in the MAC_Address registers, it is marked as failing the SA Address filter. When this bit is reset, if the SA of a packet does not match the values programmed in the MAC_Address registers, it is marked as failing the SA Address filter.

(table continues...)

(continued)

Field	Bits	Type	Description
SAF	9	rw	**Source Address Filter Enable** When this bit is set, the MAC compares the SA field of the received packets with the values programmed in the enabled MAC_Address registers. If the comparison fails, the MAC drops the packet. When this bit is reset, the MAC forwards the received packet to the application with updated SAF bit of the Rx Status depending on the SA address comparison. Note: According to the IEEE specification, Bit 47 of the SA is reserved. However, in DWC_xgmac, the MAC compares all 48 bits. The software driver must consider this while programming the MAC address registers for SA.
DHLFRS	12:11	rw	**L3/L4 Filter Number in Receive Status** This field selects L3/L4 filter number information to provide in receive descriptor. Values: 0x0: Use backward compatible L3/L4 filter number field (RDES2[31:29]) in receive status. 0x1: Reserved. 0x2: Use combined RDES2[26:19] and RDES2[31:29] fields to provide L3/L4 filter number. 0x3: Reserved.
VTFE	16	rw	**VLAN Tag Filter Enable** When this bit is set, the MAC drops the VLAN tagged packets that do not match the VLAN Tag. When this bit is reset, the MAC forwards all packets irrespective of the match status of the VLAN Tag.
IPFE	20	rw	**Layer 3 and Layer 4 Filter Enable** When this bit is set, the MAC drops packets that do not match the enabled Layer 3 and Layer 4 filters. If Layer 3 or Layer 4 filters are not enabled for matching, this bit does not have any effect. When this bit is reset, the MAC forwards all packets irrespective of the match status of the Layer 3 and Layer 4 fields.
DNTU	21	rw	**Drop Non-TCP/UDP over IP Packets** When this bit is set, the MAC drops the non-TCP or UDP over IP packets. The MAC forward only those packets that are processed by the Layer 4 filter. When this bit is reset, the MAC forwards all non-TCP or UDP over IP packets.
RA	31	rw	**Receive All** When this bit is set, the MAC Receiver module passes all the received packets to the application, irrespective of whether they pass the address filter or not. The result of the SA or DA filtering is updated (pass or fail) in the corresponding bit in the Rx Status Word. When this bit is reset, the Receiver module passes only those packets to the application that pass the SA or DA address filter.

(table continues...)

(continued)

Field	Bits	Type	Description
0	1, 2, 10, 15:13, 19:17, 22, 30:23	r	**Reserved** Read as 0; should be written with 0.

14.6.31 Port j MAC WD JB Timeout

The Watchdog and Jabber Timeout register controls the watchdog timeout limit for the received packets and jabber timeout limit for transmitted packets.

Portj_MAC_WD_JB_Timeout (j=0-1) Offset address: $1000C_H+j*2000_H$

Port j MAC WD JB Timeout Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0							PJE	0				JTO			
r							rw	r				rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0							PWE	0				WTO			
r							rw	r				rw			

Field	Bits	Type	Description
WTO	3:0	rw	**Watchdog Timeout** When the PWE bit is set and the WD bit of the MAC_Rx_Configuration register is reset, this field is used as watchdog timeout limit for a received packet. If the length of a received packet exceeds the value of this field, such a packet is terminated and declared as an error packet. Encoding is as follows: - 4'h0: 2 KB - 4'h1: 3 KB - 4'h2: 4 KB - 4'h3: 5 KB - ... - 4'hC: 14 KB - 4'hD: 15 KB - 4'hE: 16383 Bytes - 4'hF: Reserved

(table continues...)

(continued)

Field	Bits	Type	Description
PWE	8	rw	**Programmable Watchdog Enable** When this bit is set and the WD bit of the MAC_Rx_Configuration register is reset, the WTO field is used as watchdog timeout limit for a received packet. When this bit is cleared, the watchdog timeout for a received packet is controlled by setting of WD and JE bits in MAC_Rx_Configuration register.
JTO	19:16	rw	**Jabber Timeout** When the PJE bit is set and the JD bit of the MAC_Tx_Configuration register is reset, this field is used as jabber timeout limit for a transmitted packet. If the length of a transmitted packet exceeds the value of this field, such a packet is terminated and declared as an error packet. Encoding is as follows: - 4'h0: 2 KB - 4'h1: 3 KB - 4'h2: 4 KB - 4'h3: 5 KB - ... - 4'hC: 14 KB - 4'hD: 15 KB - 4'hE: 16383 Bytes - 4'hF: Reserved
PJE	24	rw	**Programmable Jabber Enable** When this bit is set and the JD bit of the MAC_Tx_Configuration register is reset, the JTO field is used as jabber timeout limit for a transmitted packet. When this bit is cleared, the jabber timeout limit for a transmitted packet is controlled by setting of JD and JE bits in MAC_Tx_Configuration and MAC_Rx_Configuration registers, respectively.
0	7:4, 15:9, 23:20, 31:25	r	**Reserved** Read as 0; should be written with 0.

14.6.32 Port j MAC VLAN Tag Ctrl

This register holds the control and addressing fields required for indirect accessing of the MAC_VLAN_Tag_Filter registers. And also it holds the control fields for MAC VLAN Hash Filter and perfect filtering.

Portj_MAC_VLAN_Tag_Ctrl (j=0-1) Offset address: $10050_H+j*2000_H$

Port j MAC VLAN Tag Ctrl Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EIVLRXS	0	EIVLS		ERIVLT	EDVLP	VTHM	EVLRXS	0	EVLS		DOVLTC	ERSVLM	ESVL	VTIM	ETV
rw	r	rw		rw	rw	rw	rw	r	rw		rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		0		0					OFS					CT	OB
r		r		r					rw					rw	rw

Field	Bits	Type	Description
OB	0	rw	**Operation Busy** This bit is set along with a read or write command for initiating the indirect access to per VLAN Tag Filter register. The next indirect register access can be initiated only after this bit is reset. During a write operation, the bit is reset only after the data has been written into the Per VLAN Tag register. During a read operation, the data must be read from the MAC_VLAN_Tag_Data register only after this bit is reset.
CT	1	rw	**Command Type** This bit indicates if the current register access is a read or a write. When set, it indicate a read operation. When reset, it indicates a write operation.
OFS	6:2	rw	**Offset** This field holds the address offset of the MAC VLAN Tag Filter register which the application is trying to access. The width of the field depends on the number of MAC VLAN Tag registers enabled.
ETV	16	rw	**Enable 12-Bit VLAN Tag Comparison for VLAN hash filtering** When this bit is set, a 12-bit VLAN identifier is used for VLAN hash filtering instead of the complete 16-bit VLAN tag. When this bit is reset, all 16 bits of the 15th and 16th bytes of the received VLAN packet are used for VLAN hash filtering.
VTIM	17	rw	**VLAN Tag Inverse Match/Result Enable** When this bit is set, it enables the VLAN Tag inverse matching . A match is marked when Packets have non-matching VLAN tags. When reset, it enables the VLAN Tag perfect matching . A match is marked when packets have matching VLAN tags.
ESVL	18	rw	**Enable S-VLAN** When this bit is set, the MAC transmitter and receiver consider the S-VLAN packets (Type = 0x88A8) as valid VLAN tagged packets.
ERSVLM	19	rw	**Enable Receive S-VLAN Match for VLAN hash filtering** When this bit is set, the MAC receiver enables filtering or matching for S-VLAN (Type = 0x88A8) packets. When this bit is reset, the MAC receiver enables filtering or matching for C-VLAN (Type = 0x8100) packets. The ERIVLT bit determines the VLAN tag position considered for filtering or matching.

(table continues...)

(continued)

Field	Bits	Type	Description
DOVLTC	20	rw	**Disable VLAN Type Check for VLAN hash filtering** When this bit is set, the MAC does not check whether the VLAN Tag specified by the ERIVLT bit is of type S-VLAN or C-VLAN. When this bit is reset, the MAC filters or matches the VLAN Tag specified by the ERIVLT bit only when VLAN Tag type is similar to the one specified by the ERSVLM bit.
EVLS	22:21	rw	**Enable VLAN Tag Stripping on Receive** This field indicates the stripping operation on the outer VLAN Tag in received packet: - 2'b00: Do not strip - 2'b01: Strip if VLAN filter passes - 2'b10: Strip if VLAN filter fails - 2'b11: Always strip
EVLRXS	24	rw	**Enable VLAN Tag in Rx status** When this bit is set, MAC provides the outer VLAN Tag in the Rx status. When this bit is reset, the MAC does not provide the outer VLAN Tag in Rx status.
VTHM	25	rw	**VLAN Tag Hash Table Match Enable** When this bit is set, the most significant four bits of CRC of VLAN Tag are used to index the content of the MAC_VLAN_Hash_Table register. A value of 1 in the VLAN Hash Table register, corresponding to the index, indicates that the packet matched the VLAN hash table. When the ETV bit is set, the CRC of the 12-bit VLAN Identifier (VID) is used for comparison. When the ETV bit is reset, the CRC of the 16-bit VLAN tag is used for comparison. When this bit is reset, the VLAN Hash Match operation is not performed.
EDVLP	26	rw	**Enable Double VLAN Processing** When this bit is set, the MAC enables processing of up to two VLAN Tags on Tx and Rx (if present). When this bit is reset, the MAC enables processing of up to one VLAN Tag on Tx and Rx (if present).
ERIVLT	27	rw	**Enable Inner VLAN Tag for VLAN hash filtering** When this bit and the EDVLP field are set, the MAC receiver enables hash operation on the inner VLAN Tag (if present). When this bit is reset, the MAC receiver enables hash operation on the outer VLAN Tag (if present). The ERSVLM bit determines which VLAN type is enabled for filtering or matching.
EIVLS	29:28	rw	**Enable Inner VLAN Tag Stripping on Receive** This field indicates the stripping operation on inner VLAN Tag in the received packet: - 2'b00: Do not strip - 2'b01: Strip if VLAN filter passes - 2'b10: Strip if VLAN filter fails - 2'b11: Always strip

(table continues...)

(continued)

Field	Bits	Type	Description
EIVLRXS	31	rw	**Enable Inner VLAN Tag in Rx Status** When this bit is set, the MAC provides the inner VLAN Tag in the Rx status. When this bit is reset, the MAC does not provide the inner VLAN Tag in Rx status.
0	11:7, 13:12, 15:14, 23, 30	r	**Reserved** Read as 0; should be written with 0.

14.6.33 Port j MAC VLAN Tag Data

This register holds the read/write data for Indirect Access of the Per MAC_VLAN_Tag_Filter registers. During the read access, this field contains valid read data only after the OB bit is reset. During the write access, this field must be valid prior to setting the OB bit in the MAC_VLAN_Tag_Ctrl register.

Portj_MAC_VLAN_Tag_Data (j=0-1) Offset address: $10054_H+j*2000_H$

Port j MAC VLAN Tag Data Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0				DMACHN			DMACHEN	0			ERIVLT	ERSVLM	DOVLTC	ETV	VEN
r				rw			rw	r			rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VID															
rw															

Field	Bits	Type	Description
VID	15:0	rw	**Refer VID field of MAC_VLAN_Tag_Filter(#i) indirect register**
VEN	16	rw	**Refer VEN field of MAC_VLAN_Tag_Filter(#i) indirect register**
ETV	17	rw	**Refer ETV field of MAC_VLAN_Tag_Filter(#i) indirect register**
DOVLTC	18	rw	**Refer DOVLTC field of MAC_VLAN_Tag_Filter(#i) indirect register**
ERSVLM	19	rw	**Refer ERSVLM field of MAC_VLAN_Tag_Filter(#i) indirect register**
ERIVLT	20	rw	**Refer ERIVLT field of MAC_VLAN_Tag_Filter(#i) indirect register**
DMACHEN	24	rw	**Refer DMACHEN field of MAC_VLAN_Tag_Filter(#i) indirect register**
DMACHN	27:25	rw	**Refer DMACHN field of MAC_VLAN_Tag_Filter(#i) indirect register**
0	23:21, 31:28	r	**Reserved** Read as 0; should be written with 0.

14.6.34 Port j MAC VLAN Tag Filter0

This register contains per VLAN Tag filter control information.

Portj_MAC_VLAN_Tag_Filter0 (j=0-1) Offset address: $10054_H+j*2000_H$

Port j MAC VLAN Tag Filter0 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0				DMACHN			DMACHEN	0			ERIVLT	ERSVLM	DOVLTC	ETV	VEN
r				rw			rw	r			rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VID															
rw															

Field	Bits	Type	Description
VID	15:0	rw	**VLAN Tag ID** This field holds the VLAN Tag value which is used by the MAC for perfect comparison. It is valid when VLAN Tag Enable is set.
VEN	16	rw	**VLAN Tag Enable** This bit is used to enable or disable the VLAN Tag. When this bit is set, the MAC compares the VLAN Tag of received packet with the VLAN Tag ID. When this bit is not set, no comparison is performed irrespective of the programming of the other fields.
ETV	17	rw	**Enable 12-Bit VLAN Tag Comparison** When this bit is set, a 12-bit VLAN identifier is used for comparing and filtering instead of the complete 16-bit VLAN tag. Bits[11:0] of VLAN tag are compared with the corresponding field in the received VLAN-tagged packet. Similarly, when enabled, only 12 bits of the VLAN tag in the received packet are used . When this bit is reset, all 16 bits of the 15th and 16th bytes of the received VLAN packet are used for comparison
DOVLTC	18	rw	**Disable VLAN Type Check** When this bit is set, a 12-bit VLAN identifier is used for comparing and filtering instead of the complete 16-bit VLAN tag. Bits[11:0] of VLAN tag are compared with the corresponding field in the received VLAN-tagged packet. Similarly, when enabled, only 12 bits of the VLAN tag in the received packet are used for hash-based VLAN filtering. When this bit is reset, all 16 bits of the 15th and 16th bytes of the received VLAN packet are used for comparison and VLAN hash filtering.
ERSVLM	19	rw	**Enable Receive S-VLAN Match** When this bit is set, the MAC receiver enables filtering or matching for S-VLAN (Type = 0x88A8) packets. When this bit is reset, the MAC receiver enables filtering or matching for CVLAN (Type = 0x8100) packets. The ERIVLT bit determines the VLAN tag position considered for filtering or matching.

(table continues...)

(continued)

Field	Bits	Type	Description
ERIVLT	20	rw	**Enable Inner VLAN Tag** When this bit and the EDVLP field are set, the MAC receiver enables operation on the inner VLAN Tag (if present). When this bit is reset, the MAC receiver enables operation on the outer VLAN Tag (if present). The ERSVLM bit determines which VLAN type is enabled for filtering or matching.
DMACHEN	24	rw	**DMA Channel Number Enable** If this bit is set,it enables the DMA Channel Number value programmed in the field DMACH. If this is reset, then the Routing does not occur based on the VLAN Filter result. The packet is routed based on the DA Based DMA Channel Routing.
DMACHN	27:25	rw	**DMA Channel Number** The DMA Channel number to which the VLAN Tagged Packet is to be routed if it passes this VLAN Tag Filter is programmed in this field. This field is valid only when DMACHEN=1. If the Routing based on the VLAN Tag Filter is not necessary, this field need not be programmed.
0	23:21, 31:28	r	**Reserved** Read as 0; should be written with 0.

14.6.35 Port j MAC VLAN Tag Filter1

This register contains per VLAN Tag filter control information.

Portj_MAC_VLAN_Tag_Filter1 (j=0-1) Offset address: $10054_H+j*2000_H$

Port j MAC VLAN Tag Filter1 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0				DMACHN			DMACHEN	0			ERIVLT	ERSVLM	DOVLTC	ETV	VEN
r				rw			rw	r			rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VID															
rw															

Field	Bits	Type	Description
VID	15:0	rw	**VLAN Tag ID** This field holds the VLAN Tag value which is used by the MAC for perfect comparison. It is valid when VLAN Tag Enable is set.

(table continues...)

(continued)

Field	Bits	Type	Description
VEN	16	rw	**VLAN Tag Enable** This bit is used to enable or disable the VLAN Tag. When this bit is set, the MAC compares the VLAN Tag of received packet with the VLAN Tag ID. When this bit is not set, no comparison is performed irrespective of the programming of the other fields.
ETV	17	rw	**Enable 12-Bit VLAN Tag Comparison** When this bit is set, a 12-bit VLAN identifier is used for comparing and filtering instead of the complete 16-bit VLAN tag. Bits[11:0] of VLAN tag are compared with the corresponding field in the received VLAN-tagged packet. Similarly, when enabled, only 12 bits of the VLAN tag in the received packet are used . When this bit is reset, all 16 bits of the 15th and 16th bytes of the received VLAN packet are used for comparison
DOVLTC	18	rw	**Disable VLAN Type Check** When this bit is set, a 12-bit VLAN identifier is used for comparing and filtering instead of the complete 16-bit VLAN tag. Bits[11:0] of VLAN tag are compared with the corresponding field in the received VLAN-tagged packet. Similarly, when enabled, only 12 bits of the VLAN tag in the received packet are used for hash-based VLAN filtering. When this bit is reset, all 16 bits of the 15th and 16th bytes of the received VLAN packet are used for comparison and VLAN hash filtering.
ERSVLM	19	rw	**Enable Receive S-VLAN Match** When this bit is set, the MAC receiver enables filtering or matching for S-VLAN (Type = 0x88A8) packets. When this bit is reset, the MAC receiver enables filtering or matching for CVLAN (Type = 0x8100) packets. The ERIVLT bit determines the VLAN tag position considered for filtering or matching.
ERIVLT	20	rw	**Enable Inner VLAN Tag** When this bit and the EDVLP field are set, the MAC receiver enables operation on the inner VLAN Tag (if present). When this bit is reset, the MAC receiver enables operation on the outer VLAN Tag (if present). The ERSVLM bit determines which VLAN type is enabled for filtering or matching.
DMACHEN	24	rw	**DMA Channel Number Enable** If this bit is set,it enables the DMA Channel Number value programmed in the field DMACH. If this is reset, then the Routing does not occur based on the VLAN Filter result. The packet is routed based on the DA Based DMA Channel Routing.
DMACHN	27:25	rw	**DMA Channel Number** The DMA Channel number to which the VLAN Tagged Packet is to be routed if it passes this VLAN Tag Filter is programmed in this field. This field is valid only when DMACHEN=1. If the Routing based on the VLAN Tag Filter is not necessary, this field need not be programmed.
0	23:21, 31:28	r	**Reserved** Read as 0; should be written with 0.

14.6.36 Port j MAC VLAN Tag Filter2

This register contains per VLAN Tag filter control information.

Portj_MAC_VLAN_Tag_Filter2 (j=0-1) Offset address: $10054_H+j*2000_H$

Port j MAC VLAN Tag Filter2 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0				DMACHN			DMACHEN	0			ERIVLT	ERSVLM	DOVLTC	ETV	VEN
r				rw			rw	r			rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VID															
rw															

Field	Bits	Type	Description
VID	15:0	rw	**VLAN Tag ID** This field holds the VLAN Tag value which is used by the MAC for perfect comparison. It is valid when VLAN Tag Enable is set.
VEN	16	rw	**VLAN Tag Enable** This bit is used to enable or disable the VLAN Tag. When this bit is set, the MAC compares the VLAN Tag of received packet with the VLAN Tag ID. When this bit is not set, no comparison is performed irrespective of the programming of the other fields.
ETV	17	rw	**Enable 12-Bit VLAN Tag Comparison** When this bit is set, a 12-bit VLAN identifier is used for comparing and filtering instead of the complete 16-bit VLAN tag. Bits[11:0] of VLAN tag are compared with the corresponding field in the received VLAN-tagged packet. Similarly, when enabled, only 12 bits of the VLAN tag in the received packet are used . When this bit is reset, all 16 bits of the 15th and 16th bytes of the received VLAN packet are used for comparison
DOVLTC	18	rw	**Disable VLAN Type Check** When this bit is set, a 12-bit VLAN identifier is used for comparing and filtering instead of the complete 16-bit VLAN tag. Bits[11:0] of VLAN tag are compared with the corresponding field in the received VLAN-tagged packet. Similarly, when enabled, only 12 bits of the VLAN tag in the received packet are used for hash-based VLAN filtering. When this bit is reset, all 16 bits of the 15th and 16th bytes of the received VLAN packet are used for comparison and VLAN hash filtering.
ERSVLM	19	rw	**Enable Receive S-VLAN Match** When this bit is set, the MAC receiver enables filtering or matching for S-VLAN (Type = 0x88A8) packets. When this bit is reset, the MAC receiver enables filtering or matching for CVLAN (Type = 0x8100) packets. The ERIVLT bit determines the VLAN tag position considered for filtering or matching.

(table continues...)

(continued)

Field	Bits	Type	Description
ERIVLT	20	rw	**Enable Inner VLAN Tag** When this bit and the EDVLP field are set, the MAC receiver enables operation on the inner VLAN Tag (if present). When this bit is reset, the MAC receiver enables operation on the outer VLAN Tag (if present). The ERSVLM bit determines which VLAN type is enabled for filtering or matching.
DMACHEN	24	rw	**DMA Channel Number Enable** If this bit is set,it enables the DMA Channel Number value programmed in the field DMACH. If this is reset, then the Routing does not occur based on the VLAN Filter result. The packet is routed based on the DA Based DMA Channel Routing.
DMACHN	27:25	rw	**DMA Channel Number** The DMA Channel number to which the VLAN Tagged Packet is to be routed if it passes this VLAN Tag Filter is programmed in this field. This field is valid only when DMACHEN=1. If the Routing based on the VLAN Tag Filter is not necessary, this field need not be programmed.
0	23:21, 31:28	r	**Reserved** Read as 0; should be written with 0.

14.6.37 Port j MAC VLAN Tag Filter3

This register contains per VLAN Tag filter control information.

Portj_MAC_VLAN_Tag_Filter3 (j=0-1) Offset address: $10054_H+j*2000_H$

Port j MAC VLAN Tag Filter3 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0				DMACHN			DMACHEN	0			ERIVLT	ERSVLM	DOVLTC	ETV	VEN
r				rw			rw	r			rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VID															
rw															

Field	Bits	Type	Description
VID	15:0	rw	**VLAN Tag ID** This field holds the VLAN Tag value which is used by the MAC for perfect comparison. It is valid when VLAN Tag Enable is set.

(table continues...)

(continued)

Field	Bits	Type	Description
VEN	16	rw	**VLAN Tag Enable** This bit is used to enable or disable the VLAN Tag. When this bit is set, the MAC compares the VLAN Tag of received packet with the VLAN Tag ID. When this bit is not set, no comparison is performed irrespective of the programming of the other fields.
ETV	17	rw	**Enable 12-Bit VLAN Tag Comparison** When this bit is set, a 12-bit VLAN identifier is used for comparing and filtering instead of the complete 16-bit VLAN tag. Bits[11:0] of VLAN tag are compared with the corresponding field in the received VLAN-tagged packet. Similarly, when enabled, only 12 bits of the VLAN tag in the received packet are used . When this bit is reset, all 16 bits of the 15th and 16th bytes of the received VLAN packet are used for comparison
DOVLTC	18	rw	**Disable VLAN Type Check** When this bit is set, a 12-bit VLAN identifier is used for comparing and filtering instead of the complete 16-bit VLAN tag. Bits[11:0] of VLAN tag are compared with the corresponding field in the received VLAN-tagged packet. Similarly, when enabled, only 12 bits of the VLAN tag in the received packet are used for hash-based VLAN filtering. When this bit is reset, all 16 bits of the 15th and 16th bytes of the received VLAN packet are used for comparison and VLAN hash filtering.
ERSVLM	19	rw	**Enable Receive S-VLAN Match** When this bit is set, the MAC receiver enables filtering or matching for S-VLAN (Type = 0x88A8) packets. When this bit is reset, the MAC receiver enables filtering or matching for CVLAN (Type = 0x8100) packets. The ERIVLT bit determines the VLAN tag position considered for filtering or matching.
ERIVLT	20	rw	**Enable Inner VLAN Tag** When this bit and the EDVLP field are set, the MAC receiver enables operation on the inner VLAN Tag (if present). When this bit is reset, the MAC receiver enables operation on the outer VLAN Tag (if present). The ERSVLM bit determines which VLAN type is enabled for filtering or matching.
DMACHEN	24	rw	**DMA Channel Number Enable** If this bit is set,it enables the DMA Channel Number value programmed in the field DMACH. If this is reset, then the Routing does not occur based on the VLAN Filter result. The packet is routed based on the DA Based DMA Channel Routing.
DMACHN	27:25	rw	**DMA Channel Number** The DMA Channel number to which the VLAN Tagged Packet is to be routed if it passes this VLAN Tag Filter is programmed in this field. This field is valid only when DMACHEN=1. If the Routing based on the VLAN Tag Filter is not necessary, this field need not be programmed.
0	23:21, 31:28	r	**Reserved** Read as 0; should be written with 0.

14.6.38 Port j MAC VLAN Tag Filter4

This register contains per VLAN Tag filter control information.

Portj_MAC_VLAN_Tag_Filter4 (j=0-1) Offset address: $10054_H+j*2000_H$

Port j MAC VLAN Tag Filter4 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0				DMACHN			DMACHEN	0			ERIVLT	ERSVLM	DOVLTC	ETV	VEN
r				rw			rw	r			rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VID															
rw															

Field	Bits	Type	Description
VID	15:0	rw	**VLAN Tag ID** This field holds the VLAN Tag value which is used by the MAC for perfect comparison. It is valid when VLAN Tag Enable is set.
VEN	16	rw	**VLAN Tag Enable** This bit is used to enable or disable the VLAN Tag. When this bit is set, the MAC compares the VLAN Tag of received packet with the VLAN Tag ID. When this bit is not set, no comparison is performed irrespective of the programming of the other fields.
ETV	17	rw	**Enable 12-Bit VLAN Tag Comparison** When this bit is set, a 12-bit VLAN identifier is used for comparing and filtering instead of the complete 16-bit VLAN tag. Bits[11:0] of VLAN tag are compared with the corresponding field in the received VLAN-tagged packet. Similarly, when enabled, only 12 bits of the VLAN tag in the received packet are used . When this bit is reset, all 16 bits of the 15th and 16th bytes of the received VLAN packet are used for comparison
DOVLTC	18	rw	**Disable VLAN Type Check** When this bit is set, a 12-bit VLAN identifier is used for comparing and filtering instead of the complete 16-bit VLAN tag. Bits[11:0] of VLAN tag are compared with the corresponding field in the received VLAN-tagged packet. Similarly, when enabled, only 12 bits of the VLAN tag in the received packet are used for hash-based VLAN filtering. When this bit is reset, all 16 bits of the 15th and 16th bytes of the received VLAN packet are used for comparison and VLAN hash filtering.
ERSVLM	19	rw	**Enable Receive S-VLAN Match** When this bit is set, the MAC receiver enables filtering or matching for S-VLAN (Type = 0x88A8) packets. When this bit is reset, the MAC receiver enables filtering or matching for CVLAN (Type = 0x8100) packets. The ERIVLT bit determines the VLAN tag position considered for filtering or matching.

(table continues...)

(continued)

Field	Bits	Type	Description
ERIVLT	20	rw	**Enable Inner VLAN Tag** When this bit and the EDVLP field are set, the MAC receiver enables operation on the inner VLAN Tag (if present). When this bit is reset, the MAC receiver enables operation on the outer VLAN Tag (if present). The ERSVLM bit determines which VLAN type is enabled for filtering or matching.
DMACHEN	24	rw	**DMA Channel Number Enable** If this bit is set,it enables the DMA Channel Number value programmed in the field DMACH. If this is reset, then the Routing does not occur based on the VLAN Filter result. The packet is routed based on the DA Based DMA Channel Routing.
DMACHN	27:25	rw	**DMA Channel Number** The DMA Channel number to which the VLAN Tagged Packet is to be routed if it passes this VLAN Tag Filter is programmed in this field. This field is valid only when DMACHEN=1. If the Routing based on the VLAN Tag Filter is not necessary, this field need not be programmed.
0	23:21, 31:28	r	**Reserved** Read as 0; should be written with 0.

14.6.39 Port j MAC VLAN Tag Filter5

This register contains per VLAN Tag filter control information.

Portj_MAC_VLAN_Tag_Filter5 (j=0-1) Offset address: $10054_H+j*2000_H$

Port j MAC VLAN Tag Filter5 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0				DMACHN			DMACHEN	0			ERIVLT	ERSVLM	DOVLTC	ETV	VEN
r				rw			rw	r			rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VID															
rw															

Field	Bits	Type	Description
VID	15:0	rw	**VLAN Tag ID** This field holds the VLAN Tag value which is used by the MAC for perfect comparison. It is valid when VLAN Tag Enable is set.

(table continues...)

(continued)

Field	Bits	Type	Description
VEN	16	rw	**VLAN Tag Enable** This bit is used to enable or disable the VLAN Tag. When this bit is set, the MAC compares the VLAN Tag of received packet with the VLAN Tag ID. When this bit is not set, no comparison is performed irrespective of the programming of the other fields.
ETV	17	rw	**Enable 12-Bit VLAN Tag Comparison** When this bit is set, a 12-bit VLAN identifier is used for comparing and filtering instead of the complete 16-bit VLAN tag. Bits[11:0] of VLAN tag are compared with the corresponding field in the received VLAN-tagged packet. Similarly, when enabled, only 12 bits of the VLAN tag in the received packet are used . When this bit is reset, all 16 bits of the 15th and 16th bytes of the received VLAN packet are used for comparison
DOVLTC	18	rw	**Disable VLAN Type Check** When this bit is set, a 12-bit VLAN identifier is used for comparing and filtering instead of the complete 16-bit VLAN tag. Bits[11:0] of VLAN tag are compared with the corresponding field in the received VLAN-tagged packet. Similarly, when enabled, only 12 bits of the VLAN tag in the received packet are used for hash-based VLAN filtering. When this bit is reset, all 16 bits of the 15th and 16th bytes of the received VLAN packet are used for comparison and VLAN hash filtering.
ERSVLM	19	rw	**Enable Receive S-VLAN Match** When this bit is set, the MAC receiver enables filtering or matching for S-VLAN (Type = 0x88A8) packets. When this bit is reset, the MAC receiver enables filtering or matching for CVLAN (Type = 0x8100) packets. The ERIVLT bit determines the VLAN tag position considered for filtering or matching.
ERIVLT	20	rw	**Enable Inner VLAN Tag** When this bit and the EDVLP field are set, the MAC receiver enables operation on the inner VLAN Tag (if present). When this bit is reset, the MAC receiver enables operation on the outer VLAN Tag (if present). The ERSVLM bit determines which VLAN type is enabled for filtering or matching.
DMACHEN	24	rw	**DMA Channel Number Enable** If this bit is set,it enables the DMA Channel Number value programmed in the field DMACH. If this is reset, then the Routing does not occur based on the VLAN Filter result. The packet is routed based on the DA Based DMA Channel Routing.
DMACHN	27:25	rw	**DMA Channel Number** The DMA Channel number to which the VLAN Tagged Packet is to be routed if it passes this VLAN Tag Filter is programmed in this field. This field is valid only when DMACHEN=1. If the Routing based on the VLAN Tag Filter is not necessary, this field need not be programmed.
0	23:21, 31:28	r	**Reserved** Read as 0; should be written with 0.

14.6.40 Port j MAC VLAN Tag Filter6

This register contains per VLAN Tag filter control information.

Portj_MAC_VLAN_Tag_Filter6 (j=0-1) Offset address: $10054_H+j*2000_H$

Port j MAC VLAN Tag Filter6 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0				DMACHN			DMACHEN	0			ERIVLT	ERSVLM	DOVLTC	ETV	VEN
r				rw			rw	r			rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VID															
rw															

Field	Bits	Type	Description
VID	15:0	rw	**VLAN Tag ID** This field holds the VLAN Tag value which is used by the MAC for perfect comparison. It is valid when VLAN Tag Enable is set.
VEN	16	rw	**VLAN Tag Enable** This bit is used to enable or disable the VLAN Tag. When this bit is set, the MAC compares the VLAN Tag of received packet with the VLAN Tag ID. When this bit is not set, no comparison is performed irrespective of the programming of the other fields.
ETV	17	rw	**Enable 12-Bit VLAN Tag Comparison** When this bit is set, a 12-bit VLAN identifier is used for comparing and filtering instead of the complete 16-bit VLAN tag. Bits[11:0] of VLAN tag are compared with the corresponding field in the received VLAN-tagged packet. Similarly, when enabled, only 12 bits of the VLAN tag in the received packet are used . When this bit is reset, all 16 bits of the 15th and 16th bytes of the received VLAN packet are used for comparison
DOVLTC	18	rw	**Disable VLAN Type Check** When this bit is set, a 12-bit VLAN identifier is used for comparing and filtering instead of the complete 16-bit VLAN tag. Bits[11:0] of VLAN tag are compared with the corresponding field in the received VLAN-tagged packet. Similarly, when enabled, only 12 bits of the VLAN tag in the received packet are used for hash-based VLAN filtering. When this bit is reset, all 16 bits of the 15th and 16th bytes of the received VLAN packet are used for comparison and VLAN hash filtering.
ERSVLM	19	rw	**Enable Receive S-VLAN Match** When this bit is set, the MAC receiver enables filtering or matching for S-VLAN (Type = 0x88A8) packets. When this bit is reset, the MAC receiver enables filtering or matching for CVLAN (Type = 0x8100) packets. The ERIVLT bit determines the VLAN tag position considered for filtering or matching.

(table continues...)

(continued)

Field	Bits	Type	Description
ERIVLT	20	rw	**Enable Inner VLAN Tag** When this bit and the EDVLP field are set, the MAC receiver enables operation on the inner VLAN Tag (if present). When this bit is reset, the MAC receiver enables operation on the outer VLAN Tag (if present). The ERSVLM bit determines which VLAN type is enabled for filtering or matching.
DMACHEN	24	rw	**DMA Channel Number Enable** If this bit is set,it enables the DMA Channel Number value programmed in the field DMACH. If this is reset, then the Routing does not occur based on the VLAN Filter result. The packet is routed based on the DA Based DMA Channel Routing.
DMACHN	27:25	rw	**DMA Channel Number** The DMA Channel number to which the VLAN Tagged Packet is to be routed if it passes this VLAN Tag Filter is programmed in this field. This field is valid only when DMACHEN=1. If the Routing based on the VLAN Tag Filter is not necessary, this field need not be programmed.
0	23:21, 31:28	r	**Reserved** Read as 0; should be written with 0.

14.6.41 Port j MAC VLAN Tag Filter7

This register contains per VLAN Tag filter control information.

Portj_MAC_VLAN_Tag_Filter7 (j=0-1) Offset address: $10054_H+j*2000_H$

Port j MAC VLAN Tag Filter7 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0				DMACHN			DMACHEN	0			ERIVLT	ERSVLM	DOVLTC	ETV	VEN
r				rw			rw	r			rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VID															
rw															

Field	Bits	Type	Description
VID	15:0	rw	**VLAN Tag ID** This field holds the VLAN Tag value which is used by the MAC for perfect comparison. It is valid when VLAN Tag Enable is set.

(table continues...)

(continued)

Field	Bits	Type	Description
VEN	16	rw	**VLAN Tag Enable** This bit is used to enable or disable the VLAN Tag. When this bit is set, the MAC compares the VLAN Tag of received packet with the VLAN Tag ID. When this bit is not set, no comparison is performed irrespective of the programming of the other fields.
ETV	17	rw	**Enable 12-Bit VLAN Tag Comparison** When this bit is set, a 12-bit VLAN identifier is used for comparing and filtering instead of the complete 16-bit VLAN tag. Bits[11:0] of VLAN tag are compared with the corresponding field in the received VLAN-tagged packet. Similarly, when enabled, only 12 bits of the VLAN tag in the received packet are used . When this bit is reset, all 16 bits of the 15th and 16th bytes of the received VLAN packet are used for comparison
DOVLTC	18	rw	**Disable VLAN Type Check** When this bit is set, a 12-bit VLAN identifier is used for comparing and filtering instead of the complete 16-bit VLAN tag. Bits[11:0] of VLAN tag are compared with the corresponding field in the received VLAN-tagged packet. Similarly, when enabled, only 12 bits of the VLAN tag in the received packet are used for hash-based VLAN filtering. When this bit is reset, all 16 bits of the 15th and 16th bytes of the received VLAN packet are used for comparison and VLAN hash filtering.
ERSVLM	19	rw	**Enable Receive S-VLAN Match** When this bit is set, the MAC receiver enables filtering or matching for S-VLAN (Type = 0x88A8) packets. When this bit is reset, the MAC receiver enables filtering or matching for CVLAN (Type = 0x8100) packets. The ERIVLT bit determines the VLAN tag position considered for filtering or matching.
ERIVLT	20	rw	**Enable Inner VLAN Tag** When this bit and the EDVLP field are set, the MAC receiver enables operation on the inner VLAN Tag (if present). When this bit is reset, the MAC receiver enables operation on the outer VLAN Tag (if present). The ERSVLM bit determines which VLAN type is enabled for filtering or matching.
DMACHEN	24	rw	**DMA Channel Number Enable** If this bit is set,it enables the DMA Channel Number value programmed in the field DMACH. If this is reset, then the Routing does not occur based on the VLAN Filter result. The packet is routed based on the DA Based DMA Channel Routing.
DMACHN	27:25	rw	**DMA Channel Number** The DMA Channel number to which the VLAN Tagged Packet is to be routed if it passes this VLAN Tag Filter is programmed in this field. This field is valid only when DMACHEN=1. If the Routing based on the VLAN Tag Filter is not necessary, this field need not be programmed.
0	23:21, 31:28	r	**Reserved** Read as 0; should be written with 0.

14.6.42 Port j MAC VLAN Hash Table

When the ERSVLM bit is set of MAC_VLAN_Tag_Ctrl, the 16-bit VLAN Hash Table register is used for group address filtering based on the VLAN tag. For hash filtering, the content of the 16-bit VLAN tag or 12-bit VLAN ID (based on the ETV bit of MAC_VLAN_Tag_Ctrl register) in the incoming packet is passed through the CRC logic. The upper four bits of the calculated CRC are used to index the contents of the VLAN Hash table. For example, a hash value of 4b'1000 selects Bit 8 of the VLAN Hash table.

The hash value of the destination address is calculated in the following way:

- Calculate the 32-bit CRC for the VLAN tag or ID (For steps to calculate CRC32, see Section 3.2.8 of IEEE 802.3-2018).
- Perform bit-wise reversal for the value obtained in step 1.
- Take the upper four bits from the value obtained in step 2.

If the VLAN hash Table register is configured to be double-synchronized to the (X)GMII clock domain, the synchronization is triggered only when Bits[15:8] (in little-endian mode) or Bits[7:0] (in big-endian mode) of this register are written.

If double-synchronization is enabled, consecutive writes to this register must be performed after at least four clock cycles in the destination clock domain.

Portj_MAC_VLAN_Hash_Table (j=0-1) Offset address: $10058_H+j*2000_H$

Port j MAC VLAN Hash Table Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VLHT															
rw															

Field	Bits	Type	Description
VLHT	15:0	rw	**VLAN Hash Table** This field contains the 16-bit VLAN Hash Table.
0	31:16	r	**Reserved** Read as 0; should be written with 0.

14.6.43 Port j MAC VLAN Incl

The VLAN Tag Inclusion or Replacement register contains the VLAN tag for insertion or replacement in the Transmit packets. It also contains the VLAN tag insertion controls.

Portj_MAC_VLAN_Incl (j=0-1) Offset address: $10060_H+j*2000_H$

Port j MAC VLAN Incl Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BUSY	RDWR	0			ADDR			0		CBTI	VLTI	CSVL	VLP	VLC	
r	rw	r			rw			r		rw	rw	rw	rw	rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VLT															
rw															

Field	Bits	Type	Description
VLT	15:0	rw	**VLAN Tag for Transmit Packets** This field contains the value of the VLAN tag to be inserted or replaced. The value must only be changed when the transmit lines are inactive or during the initialization phase. Bits[15:13] are the User Priority field, Bit 12 is the CFI/DEI field, and Bits[11:0] are the VID field in the VLAN tag. The following list describes the bits of this field: - Bits[15:13]: User Priority - Bit 12: Canonical Format Indicator (CFI) or Drop Eligible Indicator (DEI) - Bits[11:0]: VLAN Identifier (VID) field of VLAN tag
VLC	17:16	rw	**VLAN Tag Control in Transmit Packets** - 2'b00: No VLAN tag deletion, insertion, or replacement - 2'b01: VLAN tag deletion The MAC removes the VLAN type (bytes 13 and 14) and VLAN tag (bytes 15 and 16) of all transmitted packets with VLAN tags. - 2'b10: VLAN tag insertion The MAC inserts VLT in bytes 15 and 16 of the packet after inserting the Type value (0x8100 or 0x88a8) in bytes 13 and 14. This operation is performed on all transmitted packets, irrespective of whether they already have a VLAN tag. - 2'b11: VLAN tag replacement The MAC replaces VLT in bytes 15 and 16 of all VLAN-type transmitted packets (Bytes 13 and 14 are 0x8100 or 0x88a8). Note: Changes to this field take effect only on the start of a packet. If you write this register field when a packet is being transmitted, only the subsequent packet can use the updated value, that is, the current packet does not use the updated value.
VLP	18	rw	**VLAN Priority Control** When this bit is set, the control bits[17:16] are used for VLAN deletion, insertion, or replacement. When this bit is reset, bits[17:16] are ignored and the Tx descriptor is used.
CSVL	19	rw	**C-VLAN or S-VLAN** When this bit is set, S-VLAN type (0x88A8) is inserted or replaced in the 13th and 14th bytes of transmitted packets. When this bit is reset, C-VLAN type (0x8100) is inserted or replaced in the 13th and 14th bytes of transmitted packets.
VLTI	20	rw	**VLAN Tag Input** When this bit is set, it indicates that the VLAN tag to be inserted or replaced in Tx packet must be taken from Tx descriptor. When this bit is reset, it indicates that the VLAN tag to be inserted or replaced in Tx packet must be taken from the VLT field of this register.

(table continues...)

(continued)

Field	Bits	Type	Description
CBTI	21	rw	**Channel Based Tag Insertion** When this bit is set, outer VLAN tag is inserted for every packets transmitted by the MAC. The tag value is taken from the queue/channel specific VLAN tag register. This overrides VLTI, VLP, VLC and VLT fields of this register When this bit is set, a write operation to byte 3 of this register initiates the read/write access to the indirect register. When reset, outer VLAN operation is based on the setting of VLTI, VLP, VLC, and VLT fields of this register.
ADDR	26:24	rw	**Address** This field selects one of the queue/channel specific VLAN Inclusion register for read/write access. This does not have any effect when CBTI is reset.
RDWR	30	rw	**Read Write Control** This bit controls the read or write operation for indirectly accessing the queue/channel specific VLAN Inclusion register. When set, indicates write operation and when reset indicates read operation This does not have any effect when CBTI is reset.
BUSY	31	r	**BUSY** This bit indicates the status of the read/write operation of indirect access to the queue/channel specific VLAN inclusion register. When this bit is reset, write to this register is complete in write operation or read data is valid in read operation When this bit is set, No further writes are allowed to this register. This does not have any effect when CBTI is reset.
0	23:22, 29:27	r	**Reserved** Read as 0; should be written with 0.

14.6.44 Port j MAC VLAN Incl0

The per Tx Queue VLAN Tag Inclusion register contains the VLAN tag for insertion in the Transmit packets from respective Tx Queue. It also contains the VLAN tag insertion controls.

Portj_MAC_VLAN_Incl0 (j=0-1) Offset address: $10060_H+j*2000_H$

Port j MAC VLAN Incl0 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0												CSVL	0		
r												rw	r		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VLT															
rw															

Field	Bits	Type	Description
VLT	15:0	rw	**VLAN Tag for Transmit Packets** This field contains the value of the VLAN tag to be inserted or replaced. The value must only be changed when the transmit lines are inactive or during the initialization phase. Bits[15:13] are the User Priority field, Bit 12 is the CFI/DEI field, and Bits[11:0] are the VID field in the VLAN tag. The following list describes the bits of this field: - Bits[15:13]: User Priority - Bit 12: Canonical Format Indicator (CFI) or Drop Eligible Indicator (DEI) - Bits[11:0]: VLAN Identifier (VID) field of VLAN tag
CSVL	19	rw	**C-VLAN or S-VLAN** When this bit is set, S-VLAN type (0x88A8) is inserted or replaced in the 13th and 14th bytes of transmitted packets. When this bit is reset, C-VLAN type (0x8100) is inserted or replaced in the 13th and 14th bytes of transmitted packets.
0	18:16, 31:20	r	**Reserved** Read as 0; should be written with 0.

14.6.45 Port j MAC VLAN Incl1

The per Tx Queue VLAN Tag Inclusion register contains the VLAN tag for insertion in the Transmit packets from respective Tx Queue. It also contains the VLAN tag insertion controls.

Portj_MAC_VLAN_Incl1 (j=0-1) Offset address: $10060_H+j*2000_H$

Port j MAC VLAN Incl1 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0												CSVL	0		
r												rw	r		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VLT															
rw															

Field	Bits	Type	Description
VLT	15:0	rw	**VLAN Tag for Transmit Packets** This field contains the value of the VLAN tag to be inserted or replaced. The value must only be changed when the transmit lines are inactive or during the initialization phase. Bits[15:13] are the User Priority field, Bit 12 is the CFI/DEI field, and Bits[11:0] are the VID field in the VLAN tag. The following list describes the bits of this field: - Bits[15:13]: User Priority - Bit 12: Canonical Format Indicator (CFI) or Drop Eligible Indicator (DEI) - Bits[11:0]: VLAN Identifier (VID) field of VLAN tag

(table continues...)

(continued)

Field	Bits	Type	Description
CSVL	19	rw	**C-VLAN or S-VLAN** When this bit is set, S-VLAN type (0x88A8) is inserted or replaced in the 13th and 14th bytes of transmitted packets. When this bit is reset, C-VLAN type (0x8100) is inserted or replaced in the 13th and 14th bytes of transmitted packets.
0	18:16, 31:20	r	**Reserved** Read as 0; should be written with 0.

14.6.46 Port j MAC VLAN Incl2

The per Tx Queue VLAN Tag Inclusion register contains the VLAN tag for insertion in the Transmit packets from respective Tx Queue. It also contains the VLAN tag insertion controls.

Portj_MAC_VLAN_Incl2 (j=0-1) Offset address: $10060_H+j*2000_H$

Port j MAC VLAN Incl2 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0												CSVL	0		
r												rw	r		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VLT															
rw															

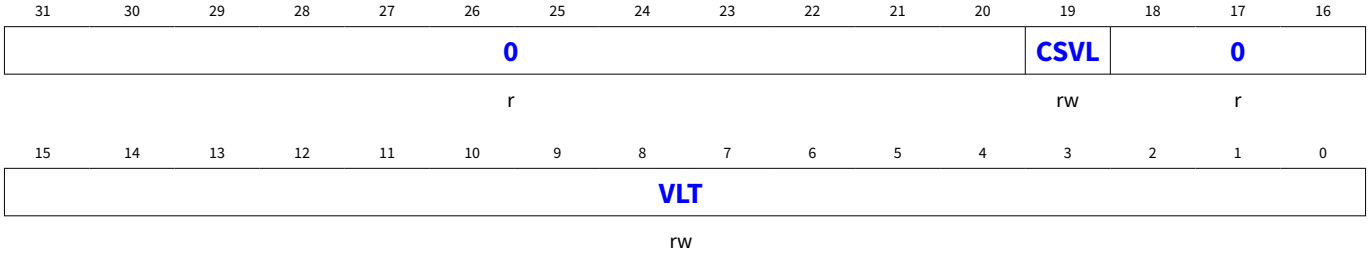
Field	Bits	Type	Description
VLT	15:0	rw	**VLAN Tag for Transmit Packets** This field contains the value of the VLAN tag to be inserted or replaced. The value must only be changed when the transmit lines are inactive or during the initialization phase. Bits[15:13] are the User Priority field, Bit 12 is the CFI/DEI field, and Bits[11:0] are the VID field in the VLAN tag. The following list describes the bits of this field: - Bits[15:13]: User Priority - Bit 12: Canonical Format Indicator (CFI) or Drop Eligible Indicator (DEI) - Bits[11:0]: VLAN Identifier (VID) field of VLAN tag
CSVL	19	rw	**C-VLAN or S-VLAN** When this bit is set, S-VLAN type (0x88A8) is inserted or replaced in the 13th and 14th bytes of transmitted packets. When this bit is reset, C-VLAN type (0x8100) is inserted or replaced in the 13th and 14th bytes of transmitted packets.
0	18:16, 31:20	r	**Reserved** Read as 0; should be written with 0.

14.6.47 Port j MAC VLAN Incl3

The per Tx Queue VLAN Tag Inclusion register contains the VLAN tag for insertion in the Transmit packets from respective Tx Queue. It also contains the VLAN tag insertion controls.

Portj_MAC_VLAN_Incl3 (j=0-1) Offset address: $10060_H+j*2000_H$

Port j MAC VLAN Incl3 Kernel Reset value: $0000\ 0000_H$



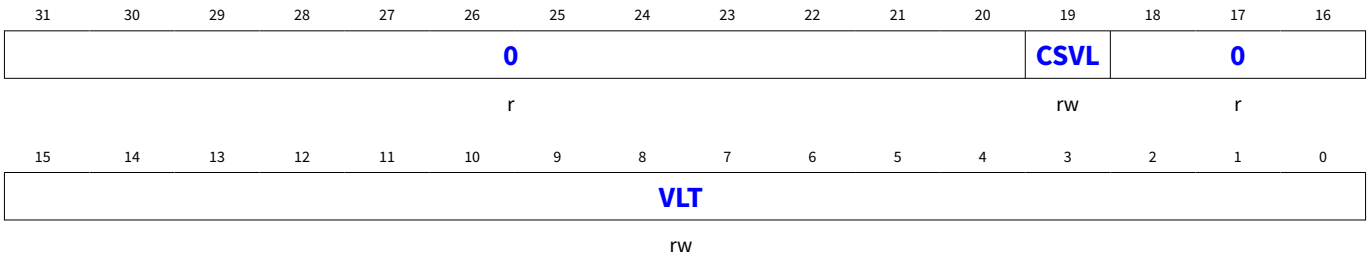
Field	Bits	Type	Description
VLT	15:0	rw	**VLAN Tag for Transmit Packets** This field contains the value of the VLAN tag to be inserted or replaced. The value must only be changed when the transmit lines are inactive or during the initialization phase. Bits[15:13] are the User Priority field, Bit 12 is the CFI/DEI field, and Bits[11:0] are the VID field in the VLAN tag. The following list describes the bits of this field: - Bits[15:13]: User Priority - Bit 12: Canonical Format Indicator (CFI) or Drop Eligible Indicator (DEI) - Bits[11:0]: VLAN Identifier (VID) field of VLAN tag
CSVL	19	rw	**C-VLAN or S-VLAN** When this bit is set, S-VLAN type (0x88A8) is inserted or replaced in the 13th and 14th bytes of transmitted packets. When this bit is reset, C-VLAN type (0x8100) is inserted or replaced in the 13th and 14th bytes of transmitted packets.
0	18:16, 31:20	r	**Reserved** Read as 0; should be written with 0.

14.6.48 Port j MAC VLAN Incl4

The per Tx Queue VLAN Tag Inclusion register contains the VLAN tag for insertion in the Transmit packets from respective Tx Queue. It also contains the VLAN tag insertion controls.

Portj_MAC_VLAN_Incl4 (j=0-1) Offset address: $10060_H+j*2000_H$

Port j MAC VLAN Incl4 Kernel Reset value: $0000\ 0000_H$



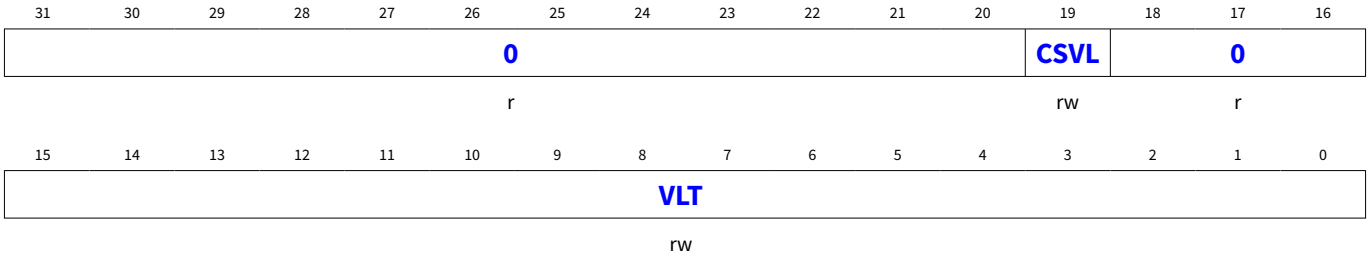
Field	Bits	Type	Description
VLT	15:0	rw	**VLAN Tag for Transmit Packets** This field contains the value of the VLAN tag to be inserted or replaced. The value must only be changed when the transmit lines are inactive or during the initialization phase. Bits[15:13] are the User Priority field, Bit 12 is the CFI/DEI field, and Bits[11:0] are the VID field in the VLAN tag. The following list describes the bits of this field: - Bits[15:13]: User Priority - Bit 12: Canonical Format Indicator (CFI) or Drop Eligible Indicator (DEI) - Bits[11:0]: VLAN Identifier (VID) field of VLAN tag
CSVL	19	rw	**C-VLAN or S-VLAN** When this bit is set, S-VLAN type (0x88A8) is inserted or replaced in the 13th and 14th bytes of transmitted packets. When this bit is reset, C-VLAN type (0x8100) is inserted or replaced in the 13th and 14th bytes of transmitted packets.
0	18:16, 31:20	r	**Reserved** Read as 0; should be written with 0.

14.6.49 Port j MAC VLAN Incl5

The per Tx Queue VLAN Tag Inclusion register contains the VLAN tag for insertion in the Transmit packets from respective Tx Queue. It also contains the VLAN tag insertion controls.

Portj_MAC_VLAN_Incl5 (j=0-1) Offset address: $10060_H+j*2000_H$

Port j MAC VLAN Incl5 Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
VLT	15:0	rw	**VLAN Tag for Transmit Packets** This field contains the value of the VLAN tag to be inserted or replaced. The value must only be changed when the transmit lines are inactive or during the initialization phase. Bits[15:13] are the User Priority field, Bit 12 is the CFI/DEI field, and Bits[11:0] are the VID field in the VLAN tag. The following list describes the bits of this field: - Bits[15:13]: User Priority - Bit 12: Canonical Format Indicator (CFI) or Drop Eligible Indicator (DEI) - Bits[11:0]: VLAN Identifier (VID) field of VLAN tag

(table continues...)

(continued)

Field	Bits	Type	Description
CSVL	19	rw	**C-VLAN or S-VLAN** When this bit is set, S-VLAN type (0x88A8) is inserted or replaced in the 13th and 14th bytes of transmitted packets. When this bit is reset, C-VLAN type (0x8100) is inserted or replaced in the 13th and 14th bytes of transmitted packets.
0	18:16, 31:20	r	**Reserved** Read as 0; should be written with 0.

14.6.50 Port j MAC VLAN Incl6

The per Tx Queue VLAN Tag Inclusion register contains the VLAN tag for insertion in the Transmit packets from respective Tx Queue. It also contains the VLAN tag insertion controls.

Portj_MAC_VLAN_Incl6 (j=0-1) Offset address: $10060_H+j*2000_H$

Port j MAC VLAN Incl6 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0												CSVL	0		
r												rw	r		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VLT															
rw															

Field	Bits	Type	Description
VLT	15:0	rw	**VLAN Tag for Transmit Packets** This field contains the value of the VLAN tag to be inserted or replaced. The value must only be changed when the transmit lines are inactive or during the initialization phase. Bits[15:13] are the User Priority field, Bit 12 is the CFI/DEI field, and Bits[11:0] are the VID field in the VLAN tag. The following list describes the bits of this field: - Bits[15:13]: User Priority - Bit 12: Canonical Format Indicator (CFI) or Drop Eligible Indicator (DEI) - Bits[11:0]: VLAN Identifier (VID) field of VLAN tag
CSVL	19	rw	**C-VLAN or S-VLAN** When this bit is set, S-VLAN type (0x88A8) is inserted or replaced in the 13th and 14th bytes of transmitted packets. When this bit is reset, C-VLAN type (0x8100) is inserted or replaced in the 13th and 14th bytes of transmitted packets.
0	18:16, 31:20	r	**Reserved** Read as 0; should be written with 0.

14.6.51 Port j MAC VLAN Incl7

The per Tx Queue VLAN Tag Inclusion register contains the VLAN tag for insertion in the Transmit packets from respective Tx Queue. It also contains the VLAN tag insertion controls.

Portj_MAC_VLAN_Incl7 (j=0-1) Offset address: $10060_H+j*2000_H$

Port j MAC VLAN Incl7 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0												CSVL	0		
r												rw	r		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VLT															
rw															

Field	Bits	Type	Description
VLT	15:0	rw	**VLAN Tag for Transmit Packets** This field contains the value of the VLAN tag to be inserted or replaced. The value must only be changed when the transmit lines are inactive or during the initialization phase. Bits[15:13] are the User Priority field, Bit 12 is the CFI/DEI field, and Bits[11:0] are the VID field in the VLAN tag. The following list describes the bits of this field: - Bits[15:13]: User Priority - Bit 12: Canonical Format Indicator (CFI) or Drop Eligible Indicator (DEI) - Bits[11:0]: VLAN Identifier (VID) field of VLAN tag
CSVL	19	rw	**C-VLAN or S-VLAN** When this bit is set, S-VLAN type (0x88A8) is inserted or replaced in the 13th and 14th bytes of transmitted packets. When this bit is reset, C-VLAN type (0x8100) is inserted or replaced in the 13th and 14th bytes of transmitted packets.
0	18:16, 31:20	r	**Reserved** Read as 0; should be written with 0.

14.6.52 Port j MAC Inner VLAN Incl

The Inner VLAN Tag Inclusion or Replacement register contains the inner VLAN tag to be inserted or replaced in the Transmit packet. It also contains the inner VLAN tag insertion controls.

Portj_MAC_Inner_VLAN_Incl (j=0-1) Offset address: $10064_H+j*2000_H$

Port j MAC Inner VLAN Incl Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0											VLTI	CSVL	VLP	VLC	
r											rw	rw	rw	rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VLT															
rw															

Field	Bits	Type	Description
VLT	15:0	rw	**VLAN Tag for Transmit Packets** This field contains the value of the VLAN tag to be inserted or replaced. The value must only be changed when the transmit lines are inactive or during the initialization phase. Bits[15:13] are the User Priority field, Bit 12 is the CFI/DEI field, and Bits[11:0] are the VID field in the VLAN tag. The following list describes the bits of this field: - Bits[15:13]: User Priority - Bit 12: Canonical Format Indicator (CFI) or Drop Eligible Indicator (DEI) - Bits[11:0]: VLAN Identifier (VID) field of VLAN tag
VLC	17:16	rw	**VLAN Tag Control in Transmit Packets** - 2'b00: No VLAN tag deletion, insertion, or replacement - 2'b01: VLAN tag deletion The MAC removes the VLAN type (bytes 17 and 18) and VLAN tag (bytes 19 and 20) of all transmitted packets with VLAN tags. - 2'b10: VLAN tag insertion The MAC inserts VLT in bytes 19 and 20 of the packet after inserting the Type value (0x8100 or 0x88a8) in bytes 17 and 18. This operation is performed on all transmitted packets, irrespective of whether they already have a VLAN tag. - 2'b11: VLAN tag replacement The MAC replaces VLT in bytes 19 and 20 of all VLAN-type transmitted packets (Bytes 17 and 18 are 0x8100 or 0x88a8). Note: Changes to this field take effect only on the start of a packet. If you write this register field when a packet is being transmitted, only the subsequent packet can use the updated value, that is, the current packet does not use the updated value.
VLP	18	rw	**VLAN Priority Control** When this bit is set, the VLC field is used for VLAN deletion, insertion, or replacement. When this bit is reset,the VLC field is ignored and the Tx descriptor is used.
CSVL	19	rw	**C-VLAN or S-VLAN** When this bit is set, S-VLAN type (0x88A8) is inserted or replaced in the 13th and 14th bytes of transmitted packets. When this bit is reset, C-VLAN type (0x8100) is inserted or replaced in the 13th and 14th bytes of transmitted packets.
VLTI	20	rw	**VLAN Tag Input** When this bit is set, it indicates that the VLAN tag to be inserted or replaced in Tx packet must be taken from Tx descriptor
0	31:21	r	**Reserved** Read as 0; should be written with 0.

14.6.53 Port j MAC Rx Eth Type Match

The ET field of this register contains the value that needs to be compared with the Ethernet type field of the received packet. The result is indicated in the ET/LT field of receive status word. This field must be programmed with values greater than or equal to 1536. Values less than these are ignored.

Portj_MAC_Rx_Eth_Type_Match (j=0-1) Offset address: $1006C_H+j*2000_H$

Port j MAC Rx Eth Type Match Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ET															
rw															

Field	Bits	Type	Description
ET	15:0	rw	**Ethernet Type** This field holds the Ethernet Type value used for length/type matching.
0	31:16	r	**Reserved** Read as 0; should be written with 0.

14.6.54 Port j MAC Q0 Tx Flow Ctrl

The MAC_Q0_Tx_Flow_Ctrl register controls the generation of the Control (Pause Command) packets by the Flow control module of the MAC. A Write to a register with the Busy bit(FCB) set to 1 triggers the Flow Control block to generate a Pause packet. The fields of the control packet are selected as specified in the IEEE 802.3-2018 specification, and the Pause Time value from this register is used in the Pause Time field of the control packet. The Busy bit remains set until the control packet is transmitted by the MAC. The application must make sure that the Busy bit is cleared before writing to the register.

Portj_MAC_Q0_Tx_Flow_Ctrl (j=0-1) Offset address: $10070_H+j*2000_H$

Port j MAC Q0 Tx Flow Ctrl Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PT															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								DZPQ	PLT			0		TFE	FCB
r								rw	rw			r		rw	rw

Field	Bits	Type	Description
FCB	0	rw	**Flow Control Busy or Backpressure Activate** This bit initiates a Pause packet in the full-duplex mode and activates the backpressure function in the half-duplex mode if the TFE bit is set. In the full-duplex mode, this bit must be read as 1'b0 before writing to this register.To initiate a Pause packet, the application must set this bit to 1'b1. During Control packet transfer, this bit continues to be set to indicate that a packet transmission is in progress. When Pause packet transmission is complete, the MAC resets this bit to 1'b0. You must not write to this register until this bit is cleared. When this bit is set (and TFE bit is set) in the half-duplex mode, the MAC asserts the backpressure. During backpressure, when the MAC receives a new packet, the transmitter starts sending a JAM pattern resulting in a collision. For the backpressure function, this control bit is logically ORed with the H/W transmit pause frame request signal.When the MAC is configured for the full-duplex mode, the BPA is automatically disabled.
TFE	1	rw	**Transmit Flow Control Enable** In the full-duplex mode, when this bit is set, the MAC enables the flow control operation based on the Tx Pause packets. When this bit is reset, the flow control operation in the MAC is disabled, and the MAC does not transmit any Pause packets by itself. In the half-duplex mode, when this bit is set, the MAC enables the backpressure operation. When this bit is reset, the backpressure feature is disabled.
PLT	6:4	rw	**Pause Low Threshold** This field configures the threshold of the Pause timer at which the input flow control signal from the FIFO layer is checked for automatic retransmission of the Pause packet. The threshold values must be always less than the Pause Time configured in Bits[31:16]. For example, if PT = 100H (256 slot times), and PLT = 001, a second Pause packet is automatically transmitted if the transmit pause frame request signal is asserted at 228 (256 - 28) slot times after the first Pause packet is transmitted. The following list provides the threshold values for different values: - 3'b000: Pause Time minus 4 Slot Times (PT - 4 slot times) - 3'b001: Pause Time minus 28 Slot Times (PT - 28 slot times) - 3'b010: Pause Time minus 36 Slot Times (PT - 36 slot times) - 3'b011: Pause Time minus 144 Slot Times (PT - 144 slot times) - 3'b100: Pause Time minus 256 Slot Times (PT - 256 slot times) - 3'b101 - 3'b111: Reserved The slot time is defined as the time taken to transmit 512 bits (64 bytes) on the GMII or XGMII interface. This (approximate) computation is based on the packet size (64, 1518, 2000, 9018, or 16384) + 2 Pause Packet Size + IPG in Slot Times.

(table continues...)

(continued)

Field	Bits	Type	Description
DZPQ	7	rw	**Disable Zero-Quanta Pause** When this bit is set, it disables the automatic generation of the zero-quanta Pause packets on de-assertion of the flow-control signal from the FIFO layer When this bit is reset, normal operation with automatic zero-quanta Pause packet generation is enabled.
PT	31:16	rw	**Pause Time** This field holds the value to be used in the Pause Time field in the Tx control packet. If the Pause Time bits are configured to be double-synchronized to the (X)GMII clock domain, consecutive writes to this register must be performed only after at least four clock cycles in the destination clock domain.
0	3:2, 15:8	r	**Reserved** Read as 0; should be written with 0.

14.6.55 Port j MAC Rx Flow Ctrl

The Receive Flow Control register controls the pausing of MAC Transmit based on the received Pause packet.

Portj_MAC_Rx_Flow_Ctrl (j=0-1) Offset address: $10090_H+j*2000_H$

Port j MAC Rx Flow Ctrl Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0							0	0						UP	RFE
r							r	r						rw	rw

Field	Bits	Type	Description
RFE	0	rw	**Receive Flow Control Enable** - 1: When MAC operates in full-duplex mode, the MAC decodes the received Pause packet and disables its transmitter for a specified (Pause) time. - 0: When MAC operates in half-duplex mode, the decode function of the Pause packet is disabled.

(table continues...)

(continued)

Field	Bits	Type	Description
UP	1	rw	**Unicast Pause Packet Detect** A pause packet is processed when it has the unique multicast address specified in the IEEE 802.3-2018. When this bit is set, the MAC can also detect Pause packets with unicast address of the station. This unicast address must be as specified in MAC_Address0_High and MAC_Address0_Low. When this bit is reset, the MAC only detects Pause packets with unique multicast address. Note: The MAC does not process a Pause packet if the multicast address is different from the unique multicast address. The unique multicast address (0x01_80_C2_00_00_01) is as specified in IEEE 802.1Qbb-2011.
0	7:2, 8, 31:9	r	**Reserved** Read as 0; should be written with 0.

14.6.56 Port j MAC RxQ Ctrl4

The Receive Queue Control 4 register controls the routing of

- Packets which fail the Destination or Source Address filter and VLAN Filter. Packets refer to Express packets only when FPE is configured
- Preemptive Multicast/Broadcast packets when MCBCQEN of MAC_RxQ_Ctrl1 is enabled to a separate Rx Queue

Portj_MAC_RxQ_Ctrl4 (j=0-1) Offset address: $10094_H+j*2000_H$

Port j MAC RxQ Ctrl4 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UDC	0				PMCBCQ			0				VFFQ			VFFQE
rw	r				rw			r				rw			rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				MFFQ			MFFQE	0				UFFQ			UFFQE
r				rw			rw	r				rw			rw

Field	Bits	Type	Description
UFFQE	0	rw	**Unicast Filter Fail Packets Queuing Enable** When this bit is set, the Unicast packets which fail the Destination or Source address filter are routed to the Rx Queue Number programmed in the UFFQ. When this bit is reset, the Unicast packets which fail the Destination or Source address filter are routed based on other routing options. This bit is valid only when the RA bit of the MAC_Packet_Filter register is set. values: - 0: Unicast Address Filter Fail Packets Queuing is disabled. - 1: Unicast Address Filter Fail Packets Queuing is enabled.

(table continues...)

(continued)

Field	Bits	Type	Description
UFFQ	3:1	rw	**Unicast Address Filter Fail Packets Queue** This field holds the Rx queue number to which Unicast Packets failing the Destination or Source Address filter must be routed. This field is valid only when UFFQE the bit is set.
MFFQE	8	rw	**Multicast Filter Fail Packets Queuing Enable** When this bit is set, the Multicast packets which fail the Destination or Source address filter are routed to the Rx Queue Number programmed in the MFFQ. When this bit is reset, the Multicast packets which fail the Destination or Source address filter are routed based on other routing options. This bit is valid only when the RA bit of the MAC_Packet_Filter register is set. values: - 0: Multicast Address Filter Fail Packets Queuing is disabled. - 1: Multicast Address Filter Fail Packets Queuing is enabled.
MFFQ	11:9	rw	**Multicast Address Filter Fail Packets Queue** This field holds the Rx queue number to which Multicast Packets failing the Destination or Source Address filter must be routed. This field is valid only when MFFQE the bit is set.
VFFQE	16	rw	**VLAN Tag Filter Fail Packets Queuing Enable** When this bit is set, the tagged packets which fail the Destination or Source address filter or fail the VLAN tag filter, are routed to the Rx Queue Number programmed in the VFFQ. When this bit is reset, the tagged packets which fail the Destination or Source address filter or fail the VLAN tag filter are routed based on other routing options. This bit is valid only when the RA bit of the MAC_Packet_Filter register is set. values: - 0: VLAN tag Filter Fail Packets Queuing is disabled. - 1: VLAN tag Filter Fail Packets Queuing is enabled.
VFFQ	19:17	rw	**VLAN Tag Filter Fail Packets Queue** This field holds the Rx queue number to which the tagged packets failing the Destination or Source Address filter (and UFFQE/MFFQE not enabled) or failing the VLAN tag filter must be routed. This field is valid only when the VFFQE bit is set.
PMCBCQ	26:24	rw	**Preemption Multicast/Broadcast Queue** This field holds the Rx queue number to which the preemptive Multicast/Broadcast packets passing the Destination or Source Address filter or the VLAN tag filter must be routed. This field is valid only when the MCBCQEN bit of MAC_RxQ_Ctrl1 register is set.

(table continues...)

(continued)

Field	Bits	Type	Description
UDC	31	rw	**Unicast Packet Duplication Control** When this bit is set along with DDS bit of the MAC_Extended_Configuration register, the MAC routes the received packet with Unicast Destination Address to the highest Receive Queue selected in configuration, provided that more than one bit are set in the DMA Channel number specified in XDCS field of MAC_DChSel_IndReg(#i) register that matched the Unicast Destination Address of the received packet. Note: When the above condition is satisfied for unicast packet but the highest Receive Queue selected in configuration is not enabled or when DA/SA/VLAN filter fails and corresponding filter fail Receive Queue is enabled, it will be routed to Receive Queue determined by other queuing methods. When this bit is reset, the MAC routes the received packet with Unicast Destination Address based other queuing methods, the DMA Channel number selected is binary value from DCS field of MAC_Address(#i)_High register (if DDS bit is reset) or lowest bit set in one-hot value from XDCS field of MAC_DChSel_IndReg(#i) indirect register (if DDS is set) that matched the Unicast Destination Address of the received packet.
0	7:4, 15:12, 23:20, 30:27	r	**Reserved** Read as 0; should be written with 0.

14.6.57 Port j MAC RxQ Ctrl5

The Receive Queue Control 5 register allows selection of start offset of the Receive queue from which received packets are routed based on VLAN Tag Priority field.

Portj_MAC_RxQ_Ctrl5 (j=0-1) Offset address: $10098_H+j*2000_H$

Port j MAC RxQ Ctrl5 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												PRQSO			
r												rw			

Field	Bits	Type	Description
PRQSO	3:0	rw	**Priority Receive Queue Start Offset** This field holds the Receive queue number from which received packets are routed based on the VLAN Tag Priority field. For example, value of 0 indicates PSRQ0, PSRQ1, .. PSRQ7 fields in MAC_RxQ_Ctrl2 and MAC_RxQ_Ctrl3 registers are mapped to Receive Queue 0, Receive Queue 1, .. Receive Queue 7 respectively. Similarly, value of 3 indicates PSRQ0, PSRQ1, .. PSRQ7 fields in MAC_RxQ_Ctrl2 and MAC_RxQ_Ctrl3 registers are mapped to Receive Queue 3, Receive Queue 4, .. Receive Queue 10 respectively. The Receive queue number programmed in this field must not be more than number of Receive queues selected in configuration. Note: When this field is programmed to a value such that Receive queues selected in configuration are lesser than the sum of programmed value and 7, all the PSRQx fields cannot be mapped to the Receive queues. For example, if number of Receive queues selected in configuration is 8 and this field is programmed to 5, only the PSRQ0, PSRQ1 and PSRQ2 can be mapped to Receive Queue 5, Receive Queue 6 and Receive Queue 7 respectively.
0	31:4	r	**Reserved** Read as 0; should be written with 0.

14.6.58 Port j MAC RxQ Ctrl0

The Receive Queue Control 0 register activates the queue management in the MAC Receiver. This register is present only when you select multiple queues in the Receive path.

Portj_MAC_RxQ_Ctrl0 (j=0-1) Offset address: $100A0_H+j*2000_H$

Port j MAC RxQ Ctrl0 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		0		0		0		0		0		0		0	
r		r		r		r		r		r		r		r	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXQ7EN		RXQ6EN		RXQ5EN		RXQ4EN		RXQ3EN		RXQ2EN		RXQ1EN		RXQ0EN	
rw		rw		rw		rw		rw		rw		rw		rw	

Field	Bits	Type	Description
RXQ0EN	1:0	rw	**Receive Queue 0 Enable** This field indicates whether Rx Queue 0 is enabled. - 2'b00: Not enabled - 2'b01: Queue 0 enabled for Audio Video Bridging - 2'b10: Queue 0 enabled for Generic - 2'b11: Reserved
RXQ1EN	3:2	rw	**Receive Queue 1 Enable** This field is similar to the RXQ0EN field.

(table continues...)

(continued)

Field	Bits	Type	Description
RXQ2EN	5:4	rw	**Receive Queue 2 Enable** This field is similar to the RXQ0EN field.
RXQ3EN	7:6	rw	**Receive Queue 3 Enable** This field is similar to the RXQ0EN field.
RXQ4EN	9:8	rw	**Receive Queue 4 Enable** This field is similar to the RXQ0EN field.
RXQ5EN	11:10	rw	**Receive Queue 5 Enable** This field is similar to the RXQ0EN field.
RXQ6EN	13:12	rw	**Receive Queue 6 Enable** This field is similar to the RXQ0EN field.
RXQ7EN	15:14	rw	**Receive Queue 7 Enable** This field is similar to the RXQ0EN field.
0	17:16, 19:18, 21:20, 23:22, 25:24, 27:26, 29:28, 31:30	r	**Reserved** Read as 0; should be written with 0.

14.6.59 Port j MAC RxQ Ctrl1

This register allows the selection of the Receive queues to which the received untagged or special packets are routed or written. This register is present only when you select multiple queues in the Receive path.

Note:

- Program the queue number in the fields within the range of Rx Queues that you have selected in your controller.

Portj_MAC_RxQ_Ctrl1 (j=0-1) Offset address: $100A4_H+j*2000_H$

Port j MAC RxQ Ctrl1 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AVCPQ				PTPQ				TACPQE	TPQC		OMCBCQ	0			
rw				rw				rw	rw		rw	r			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MCBCQEN	0			MCBCQ				RQ				UPQ			
rw	r			rw				rw				rw			

Field	Bits	Type	Description
UPQ	3:0	rw	**Untagged Packet Queue** This field specifies the Rx queue on which the received untagged packets are routed. - 4'b0000: Rx Queue 0 - 4'b0001: Rx Queue 1 - ... - 4'b0111: Rx Queue 7 - 4'b1000 - 4'b1111: Reserved
RQ	7:4	rw	**Frame Preemption Residue Queue** This field holds the Rx queue number to which the residual preemption frames must be forwarded. Preemption frames that are tagged and pass the SA/DA/VLAN filtering are routed based on the PSRQ field. All other frames are treated as residual frames and are routed to the queue number mentioned in this field. Queue-0 is used as a default queue for express frames, so this field cannot be programmed to a value 0.
MCBCQ	11:8	rw	**Multicast or Broadcast Queue** This field specifies the Rx queue on which the Multicast or Broadcast packets are routed. - 4'b0000: Rx Queue 0 - 4'b0001: Rx Queue 1 - ... - 4'b0111: Rx Queue 7 - 4'b1000 - 4'b1111: Reserved
MCBCQEN	15	rw	**Multicast or Broadcast Queue Enable** This bit specifies that Multicast or Broadcast packets routing to the Rx Queue is enabled and the Multicast or Broadcast packets must be routed to Rx Queue specified in MCBCQ field.
OMCBCQ	20	rw	**Programmable control for Over-riding MCBCQ Priority** When set to 1, this field enables over-riding the MCBC Queue priority. When set to 0, this field disables over-riding the MCBC Queue priority.
TPQC	22:21	rw	**Tagged PTP over Ethernet Packets Queuing Control** This field controls the routing of the VLAN Tagged PTP over Ethernet packets. The following programmable options are allowed. - 2'b00: VLAN Tagged PTP over Ethernet packets are routed as generic VLAN Tagged packet (based on the PSRQ field for only the non-AV enabled Rx Queues). - 2'b01: VLAN Tagged PTP over Ethernet packets are routed to Rx Queue specified by PTPQ field (That Rx Queue can be enabled for AV or non-AV traffic). - 2'b10: VLAN Tagged PTP over Ethernet packets are routed to only AV enabled Rx Queues based on the PSRQ field. - 2'b11: Reserved

(table continues...)

(continued)

Field	Bits	Type	Description
TACPQE	23	rw	**Tagged AV Control Packets Queuing Enable** When set, the MAC routes the received Tagged AV Control packets to the Rx queue specified by AVCPQ field. If Tagged AV Control packets received are expected to be preempted then do not set the TACPQE bit to enable a separate queue. When reset, the MAC routes the received Tagged AV Control packets based on the tag priority matching the PSRQ fields in MAC_RxQ_Ctrl2 and MAC_RxQ_Ctrl3 registers.
PTPQ	27:24	rw	**PTP Packets Queue** This field specifies the Rx queue on which the untagged PTP packets sent over the Ethernet payload (not over IPv4 or IPv6) are routed. - 4'b0000: Rx Queue 0 - 4'b0001: Rx Queue 1 - ... - 4'b0111: Rx Queue 7 - 4'b1000 - 4'b1111: Reserved When the AV8021ASMEN bit of MAC_Timestamp_Control register is reset, even though VLAN tagged PTP over Ethernet packets are processed, it is routed based on the VLAN Tag priority.
AVCPQ	31:28	rw	**AV Control Packets Queue** This field specifies the Rx queue on which the received AV Control packets (tagged and untagged) are routed. - 4'b0000: Rx Queue 0 - 4'b0001: Rx Queue 1 - ... - 4'b0111: Rx Queue 7 - 4'b1000 - 4'b1111: Reserved
0	14:12, 19:16	r	**Reserved** Read as 0; should be written with 0.

14.6.60 Port j MAC RxQ Ctrl2

This register controls the routing of tagged packets based on the USP (user Priority) field of the received packets to the Rx Queues 0 to 3. This register is present when multiple Rx Queues are selected

Portj_MAC_RxQ_Ctrl2 (j=0-1) Offset address: $100A8_H+j*2000_H$

Port j MAC RxQ Ctrl2 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PSRQ3								PSRQ2							
rw								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PSRQ1								PSRQ0							
rw								rw							

Field	Bits	Type	Description
PSRQ0	7:0	rw	**Priorities Selected in the Receive Queue 0** This field decides the priorities assigned to Rx Queue 0. All packets with priorities that match the values set in this field are routed to Rx Queue 0. For example, if PSRQ0[5] is set, packets with USP field equal to 5 are routed to Rx Queue 0. The software must ensure that the content of this field is mutually exclusive to the PSRQ fields for other queues, that is, the same priority is not mapped to multiple Rx queues.
PSRQ1	15:8	rw	**Priorities Selected in the Receive Queue 1** This field decides the priorities assigned to Rx Queue 1. All packets with priorities that match the values set in this field are routed to Rx Queue 1. For example, if PSRQ1[4] is set, packets with USP field equal to 4 are routed to Rx Queue 1. The software must ensure that the content of this field is mutually exclusive to the PSRQ fields for other queues, that is, the same priority is not mapped to multiple Rx queues.
PSRQ2	23:16	rw	**Priorities Selected in the Receive Queue 2** This field decides the priorities assigned to Rx Queue 2. All packets with priorities that match the values set in this field are routed to Rx Queue 2. For example, if PSRQ2[1, 0] are set, packets with USP field equal to 1 or 0 are routed to Rx Queue 2. The software must ensure that the content of this field is mutually exclusive to the PSRQ fields for other queues, that is, the same priority is not mapped to multiple Rx queues.
PSRQ3	31:24	rw	**Priorities Selected in the Receive Queue 3** This field decides the priorities assigned to Rx Queue 3. All packets with priorities that match the values set in this field are routed to Rx Queue 3. For example, if PSRQ3[6, 3] are set, packets with USP field equal to 3 or 6 are routed to Rx Queue 3. The software must ensure that the content of this field is mutually exclusive to the PSRQ fields for other queues, that is, the same priority is not mapped to multiple Rx queues.

14.6.61 Port j MAC RxQ Ctrl3

This register controls the routing of tagged packets based on the USP (user Priority) field of the received packets to the Rx Queues 4 to 7. This register is present with more than four Rx queues.

Portj_MAC_RxQ_Ctrl3 (j=0-1) Offset address: $100AC_H+j*2000_H$

Port j MAC RxQ Ctrl3 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PSRQ7								PSRQ6							
rw								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PSRQ5								PSRQ4							
rw								rw							

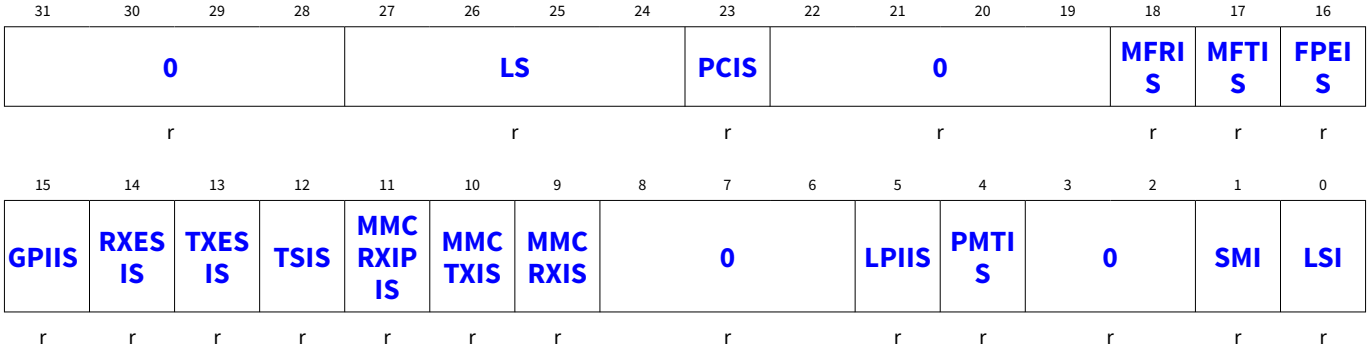
Field	Bits	Type	Description
PSRQ4	7:0	rw	**Priorities Selected in the Receive Queue 4** This field decides the priorities assigned to Rx Queue 4. All packets with priorities that match the values set in this field are routed to Rx Queue 4. For example, if PSRQ4[7:4] is set, packets with USP field equal to 7, 6, 5, or 4 are routed to Rx Queue 4. The software must ensure that the content of this field is mutually exclusive to the PSRQ fields for other queues, that is, the same priority is not mapped to multiple Rx queues.
PSRQ5	15:8	rw	**Priorities Selected in the Receive Queue 5** This field decides the priorities assigned to Rx Queue 5. All packets with priorities that match the values set in this field are routed to Rx Queue 5. For example, if PSRQ5[6] is set, packets with USP field equal to 6 are routed to Rx Queue 5. The software must ensure that the content of this field is mutually exclusive to the PSRQ fields for other queues, that is, the same priority is not mapped to multiple Rx queues.
PSRQ6	23:16	rw	**Priorities Selected in the Receive Queue 6** This field decides the priorities assigned to Rx Queue 6. All packets with priorities that match the values set in this field are routed to Rx Queue 6. For example, if PSRQ6[5] are set, packets with USP field equal to 5 are routed to Rx Queue 6. The software must ensure that the content of this field is mutually exclusive to the PSRQ fields for other queues, that is, the same priority is not mapped to multiple Rx queues.
PSRQ7	31:24	rw	**Priorities Selected in the Receive Queue 7** This field decides the priorities assigned to Rx Queue 7. All packets with priorities that match the values set in this field are routed to Rx Queue 7. For example, if PSRQ7[7, 4] are set, packets with USP field equal to 7 or 4 are routed to Rx Queue 7. The software must ensure that the content of this field is mutually exclusive to the PSRQ fields for other queues, that is, the same priority is not mapped to multiple Rx queues.

14.6.62 Port j MAC Interrupt Status

The Interrupt Status register. The register contains status of the events which generates interrupt when corresponding interrupt is enabled in MAC_Interrupt_Enable register

Portj_MAC_Interrupt_Status (j=0-1) Offset address: $100B0_H+j*2000_H$

Port j MAC Interrupt Status Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
LSI	0	r	**Link Status change Interrupt** This bit is set when the LS (Link Status) bits change their value. It is cleared when this register is read. This interrupt is generated in RGMII mode when the link status change is detected in the RGMII Rx Interface. Access restriction applies. Clears on read (or write of 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event.
SMI	1	r	**SMA Interrupt** This bit is set when any of the bits in the MDIO Interrupt Status register is set. This bit is the logical OR of all the bits in that register and is cleared only when all the bits in the MDIO Interrupt register are cleared
PMTIS	4	r	**PMT Interrupt Status** This bit is set when a Magic packet or Wake-on-LAN packet is received in the power-down mode (RWKPRCVD and MGKPRCVD bits in MAC_PMT_Control_Status register). This bit is cleared when Bits[6:5] are cleared because of a Read operation to the MAC_PMT_Control_Status register. Access restriction applies.This bit is cleared when Bits[6:5] of MAC_PMT_Control_Status register is read (or corresponding interrupt source bit of MAC_PMT_Control_Status register is written to 1 when RCWE bit of MAC_CSR_SW_Ctrl register is set).
LPIIS	5	r	**LPI Interrupt Status** This bit is set for any LPI state entry or exit in the MAC Transmitter or Receiver. Note: On the transmit side the Exit Interrupt is generate only when the XGMII/GMII interface comes out of the LPI and TW timer expires. Access restriction applies.This bit is cleared when the corresponding interrupt source bit of MAC_LPI_Control_Status register is read (or corresponding interrupt source bit of MAC_LPI_Control_Status register is written to 1 when RCWE bit of MAC_CSR_SW_Ctrl register is set).
MMCRXIS	9	r	**MMC Receive Interrupt Status** This bit is set high when an interrupt is generated in the MMC Receive Interrupt register. This bit is cleared when all bits in this interrupt register are cleared.

(table continues...)

(continued)

Field	Bits	Type	Description
MMCTXIS	10	r	**MMC Transmit Interrupt Status** This bit is set high when an interrupt is generated in the MMC Transmit Interrupt register. This bit is cleared when all bits in this interrupt register are cleared.
MMCRXIPIS	11	r	**MMC Receive Checksum Offload Interrupt Status** This bit is set high when an interrupt is generated in the MMC Receive Checksum Offload Interrupt Register. This bit is cleared when all bits in this interrupt register are cleared. 0_B **INACTIVE**: MMC Receive Checksum Offload Interrupt status not active 1_B **ACTIVE**: MMC Receive Checksum Offload Interrupt status active
TSIS	12	r	**Timestamp Interrupt Status** This bit is set when any of the following conditions is true: - The system time value is equal to or exceeds the value specified in the Target Time High and Low registers. - There is an overflow in the Seconds register. - The Target Time Error occurred, that is, programmed target time already elapsed. - When a timestamp of a transmitted PTP packet is available in the MAC_TxTimestamp_Status_Seconds register. Access restriction applies.This bit is cleared when the corresponding interrupt source bit is read (or corresponding interrupt source bit is written to 1 when RCWE bit of MAC_CSR_SW_Ctrl register is set) in the MAC_Timestamp_Status register.
TXESIS	13	r	**Transmit Error Status Interrupt** - This bit indicates the error status of transmitted packets. This bit is set when any of the following bits is set in the MAC_Rx_Tx_Status during transmission. - Excessive Collision (EXCOL) - Late Collision (LCOL) - Excessive Deferral (EXDEF) - Loss of Carrier (LCARR) - No Carrier (NCARR) - Jabber Timeout (TJT) Access restriction applies.This bit is cleared when the corresponding interrupt source bit is read (or corresponding interrupt source bit is written to 1 when RCWE bit of MAC_CSR_SW_Ctrl register is set) in the MAC_Rx_Tx_Status register.
RXESIS	14	r	**Receive Error Status Interrupt** This bit indicates the error status of received packets. This bit is set when the Watchdog Timeout (WDT) event occurs during reception. Access restriction applies.This bit is cleared when the corresponding interrupt source bit is read (or corresponding interrupt source bit is written to 1 when RCWE bit of MAC_CSR_SW_Ctrl register is set) in the MAC_Rx_Tx_Status register.

(table continues...)

(continued)

Field	Bits	Type	Description
GPIIS	15	r	**GPI Interrupt Status** This bit is set when an active event (LL or LH) occurs on the GPIS field of the MAC_GPIO_Status register and the corresponding GPIE bit is enabled in the MAC_GPIO_Control register. This bit is cleared on reading lane 0 (GPIS) of the MAC_GPIO_Status register. Access restriction applies.This bit is cleared when this bit is read (or this bit is written to 1 when RCWE bit of MAC_CSR_SW_Ctrl register is set)
FPEIS	16	r	**Frame Preemption Interrupt Status** This bit indicates an interrupt event during the operation of Frame Preemption (Bits[19:16] of MAC_FPE_CTRL_STS register is set). To reset this bit, the application must clear the event in MAC_FPE_CTRL_STS that has caused the interrupt. 0_B **INACTIVE**: Frame Preemption Interrupt status not active 1_B **ACTIVE**: Frame Preemption Interrupt status active
MFTIS	17	r	**MMC FPE Transmit Interrupt Status** This bit is set high when an interrupt is generated in the MMC FPE Transmit Interrupt Register. This bit is cleared when all bits in this interrupt register are cleared. 0_B **INACTIVE**: MMC FPE Transmit Interrupt status not active 1_B **ACTIVE**: MMC FPE Transmit Interrupt status active
MFRIS	18	r	**MMC FPE Receive Interrupt Status** This bit is set high when an interrupt is generated in the MMC FPE Receive Interrupt Register. This bit is cleared when all bits in this interrupt register are cleared. 0_B **INACTIVE**: MMC FPE Receive Interrupt status not active 1_B **ACTIVE**: MMC FPE Receive Interrupt status active
PCIS	23	r	**Policing Counter Interrupt Status** This bit is set high when an interrupt is generated in the MAC_PCTH_Intr_Status or MAC_PCTW_Intr_Status Register. This bit is cleared when all bits in MAC_PCTH_Intr_Status or MAC_PCTW_Intr_Status register are cleared. 0_B **INACTIVE**: MAC_PCTH_Intr_Status or MAC_PCTW_Intr_Status is not active 1_B **ACTIVE**: MAC_PCTH_Intr_Status or MAC_PCTW_Intr_Status is active

(table continues...)

(continued)

Field	Bits	Type	Description
LS	27:24	r	**Link Status** When current Mode is not RGMII : Bit[25:24] indicate the current link status received on the XGMII receive interface. - 2'b00: Link OK - 2'b01: Reserved - 2'b10: Local link fault - 2'b11: Remote link fault The bits [27:26] are reserved. When current mode is RGMII: Bit[27]: - 0: Link Down - 1: Link Up Bit[26:25]: - 2'b00: 10Mbps - 2'b01: 100Mbps - 2'b10: 1Gbps - 2'b11: Reserved Bit[24]: - 0: Half Duplex - 1: Full Duplex
0	3:2, 8:6, 22:19, 31:28	r	**Reserved** Read as 0; should be written with 0.

14.6.63 Port j MAC Interrupt Enable

The Interrupt Enable register contains the masks for generating the interrupts.

Portj_MAC_Interrupt_Enable (j=0-1) Offset address: $100B4_H+j*2000_H$

Port j MAC Interrupt Enable Kernel Reset value: $0000\ 0001_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FPEIE	RXESIE	TXESIE	TSIE	0						LPIIE	PMTIE	0			LSIE
rw	rw	rw	rw	r						rw	rw	r			rw

Field	Bits	Type	Description
LSIE	0	rw	**Link Status Change Interrupt Enable** When this bit is set, it enables the assertion of the interrupt signal because of the setting of LSI bit in MAC_Interrupt_Status register.
PMTIE	4	rw	**PMT Interrupt Enable** When this bit is set, it enables the assertion of the interrupt signal because of the setting of PMTIS bit in MAC_Interrupt_Status register.
LPIIE	5	rw	**LPI Interrupt Enable** When this bit is set, it enables the assertion of the interrupt signal because of the setting of LPIIS bit in MAC_Interrupt_Status register.
TSIE	12	rw	**Timestamp Interrupt Enable** When this bit is set, it enables the assertion of the interrupt signal because of the setting of TSIS bit in MAC_Interrupt_Status register.
TXESIE	13	rw	**Transmit Error Status Interrupt Enable** When this bit is set, it enables the assertion of the interrupt signal because of the setting of the TXESIS bit in the MAC_Interrupt_Status register.
RXESIE	14	rw	**Receive Error Status Interrupt Enable** When this bit is set, it enables the assertion of the interrupt signal because of the setting of the RXESIS bit in the MAC_Interrupt_Status register.
FPEIE	15	rw	**Frame Preemption Interrupt Enable** When this bit is set, it enables the assertion of the interrupt when FPEIS field is set in the MAC_Interrupt_Status register. 0_B **DISABLE**: Frame Preemption Interrupt is disabled 1_B **ENABLE**: Frame Preemption Interrupt is enabled
0	3:1, 11:6, 31:16	r	**Reserved** Read as 0; should be written with 0.

14.6.64 Port j MAC Rx Tx Status

The Receive Transmit Status register contains the Receive and Transmit Error status.

Portj_MAC_Rx_Tx_Status (j=0-1) Offset address: $100B8_H+j*2000_H$

Port j MAC Rx Tx Status Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		PCE	IHE	0			RWT	0		EXCOL	LCOL	EXDEF	LCARR	NCARR	TJT
r		r	r	r			r	r		r	r	r	r	r	r

Field	Bits	Type	Description
TJT	0	r	**Transmit Jabber Timeout** This bit indicates that the Transmit Jabber Timer expired, which happens when the packet size exceeds 2048 bytes (10240 bytes when the Jumbo packet is enabled) and the JD bit is reset in the MAC_Tx_Configuration register. This bit is set when the packet size exceeds 16383 bytes and the JD bit is set in the MAC_Tx_Configuration register. Access restriction applies. Clears on read (or write of 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event.
NCARR	1	r	**No Carrier** When the FTS bit is reset in the MTL_Operation_Mode register, this bit indicates that the carrier signal from the PHY is not present at the end of preamble transmission. Access restriction applies. Clears on read (or write of 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event. 0_B **INACTIVE**: Carrier is present 1_B **ACTIVE**: No carrier
LCARR	2	r	**Loss of Carrier** When the FTS bit is reset in the MTL_Operation_Mode register, this bit indicates that the loss of carrier occurred during packet transmission, that is, the Phy Carrier interface signal was inactive for one or more transmission clock periods during packet transmission. This bit is valid only for packets transmitted without collision. Access restriction applies. Clears on read (or write of 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event. 0_B **INACTIVE**: Carrier is present 1_B **ACTIVE**: Loss of carrier
EXDEF	3	r	**Excessive Deferral** When the FTS bit is reset in the MTL_Operation_Mode register and the DC bit is set in the MAC_Extended_Configuration register, this bit indicates that the transmission ended because of excessive deferral of over 24,288 bit times (when Jumbo packet is enabled). Access restriction applies. Clears on read (or write of 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event. 0_B **INACTIVE**: No Excessive deferral 1_B **ACTIVE**: Excessive deferral
LCOL	4	r	**Late Collision** When the FTS bit is reset in the MTL_Operation_Mode register, this bit indicates that the packet transmission aborted because a collision occurred after the collision window (64 bytes including Preamble in MII mode). This bit is not valid if the Underflow error occurs. Access restriction applies. Clears on read (or write of 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event. 0_B **INACTIVE**: No collision 1_B **ACTIVE**: Late collision is sensed

(table continues...)

(continued)

Field	Bits	Type	Description
EXCOL	5	r	**Excessive Collisions** When the FTS bit is reset in the MTL_Operation_Mode register, this bit indicates that the transmission aborted after 16 successive collisions while attempting to transmit the current packet. If the DR bit is set in the MAC_Configuration register, this bit is set after the first collision and the packet transmission is aborted. Access restriction applies. Clears on read (or write of 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event. 0_B **INACTIVE**: No collision 1_B **ACTIVE**: Excessive collision is sensed
RWT	8	r	**Receive Watchdog Timeout** This bit is set when a packet with length greater than 2048 bytes is received (10240 bytes when Jumbo Packet mode is enabled) and the WD bit is reset in the MAC_Rx_Configuration register or when a packet with length greater than 16383 bytes is received and the WD bit is set in the MAC_Rx_Configuration register. Access restriction applies. Clears on read (or write of 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event.
IHE	12	r	**IP Header Error** This bit indicates that the Checksum Offload engine detected IP header error. This failure can be either because of received Ethernet type is 0x0800 but Version field of IPv4 header is not equal to 0x4, IPv4 Header Length field indicates a value less than 0x5 (20 bytes), total packet length is less than the value given in the IPv4 Header Length field, received Ethernet type is 0x86dd but IPv6 header Version field is not equal to 0x6, packet ends before the IPv6 header (40 bytes) or extension header (as given in the corresponding Header Length field in an extension header) is completely received. Access restriction applies. Clears on read (or write of 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event.
PCE	13	r	**Payload Checksum Error** This bit indicates that the Checksum Offload engine detected IP payload error and did not insert any checksum into the encapsulated TCP, UDP, or ICMP payload. This failure can be either because of insufficient bytes, as indicated by the Payload Length field of the IP Header or the MTL starting to forward the packet to the MAC transmitter in Store-and-Forward mode without the checksum having been calculated yet. This second error condition only occurs when the Transmit Queue depth is less than the length of the IP packet being transmitted to avoid deadlock, the MTL starts forwarding the packet when the FIFO is full, even in the store-and-forward mode. Access restriction applies. Clears on read (or write of 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event.
0	7:6, 11:9, 31:14	r	**Reserved** Read as 0; should be written with 0.

14.6.65 Port j MAC PMT Control Status

The PMT Control and Status register.

Portj_MAC_PMT_Control_Status (j=0-1) Offset address: $100C0_H+j*2000_H$

Port j MAC PMT Control Status Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RWKFILTRST	0		RWKPTR					0							
rw	r		r					r							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0					RWKPFE	GLBLUCAST	0		RWKPRCVD	MGKPRCVD	0		RWKPKTEN	MGKPKTEN	PWRDWN
r					rw	rw	r		r	r	r		rw	rw	rw

Field	Bits	Type	Description
PWRDWN	0	rw	**Power Down** When this bit is set, the MAC receiver drops all the received packets until it receives the expected magic packet or remote wake-up packet. This bit is then self-cleared and the power-down mode is disabled. The software can clear this bit before the expected magic packet or remote wake-up packet is received. The packets received by the MAC after this bit is cleared are forwarded to the application. This bit must only be set when the Magic Packet Enable, Global Unicast, or Remote Wake-Up Packet Enable bit is set high. Note: You can gate-off the CSR clock during the power-down mode. However, when the CSR clock is gated-off, you cannot perform any read or write operations on this register. Therefore, the Software cannot clear this bit.
MGKPKTEN	1	rw	**Magic Packet Enable** When this bit is set, a power management event is generated when the MAC receives a magic packet.
RWKPKTEN	2	rw	**Remote Wake-Up Packet Enable** When this bit is set, a power management event is generated when the MAC receives a remote wake-up packet.
MGKPRCVD	5	r	**Magic Packet Received** When this bit is set, it indicates that the power management event is generated because of the reception of a magic packet. This bit is cleared when this register is read. Access restriction applies. Clears on read (or write of 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event.

(table continues...)

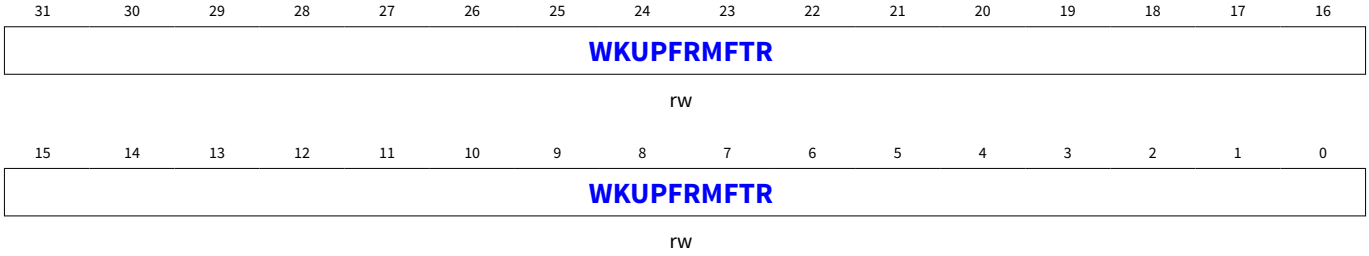
(continued)

Field	Bits	Type	Description
RWKPRCVD	6	r	**Remote Wake-Up Packet Received** When this bit is set, it indicates that the power management event is generated because of the reception of a remote wake-up packet. This bit is cleared when this register is read. Access restriction applies. Clears on read (or write of 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set).Self-set to 1 on internal event.
GLBLUCAST	9	rw	**Global Unicast** When this bit is set, any unicast packet filtered by the MAC (DA matches MAC_Address0 register) address recognition is detected as a remote wake-up packet.
RWKPFE	10	rw	**Remote Wake-up Packet Forwarding Enable** When this bit is set along with RWKPKTEN, the MAC receiver drops all the received packets until it receives the expected Wake-up packet. All packets after that event including the received wake-up packet are forwarded to application. This bit is then self-cleared on receiving the wake-up packet. The application can also clear this bit before the expected wake-up packet is received. In such cases, the MAC reverts to the default behavior where packets received are forwarded to the application. This bit must only be set when RWKPKTEN is set high and PWRDWN bit is set low. The setting of this bit has no effect when PWRDWN bit is set high. Note: If Magic Packet Enable and Wake-Up Packet Enable are both set along with setting of this bit and Magic Packet is received prior to wake-up packet, this bit is self-cleared on receiving Magic Packet, the received Magic packet is dropped, and all packets that are received after the Magic Packet are forwarded to application.
RWKPTR	28:24	r	**Remote Wake-up FIFO Pointer** This field gives the current value of the Remote Wake-up Packet Filter register pointer. When the value of this pointer reaches the maximum for the selected number of filters (which equals 2 times the number of RWK filters -1), the contents of the Remote Wake-up Packet Filter register are transferred to the Rx clock domain when a Write occurs to that register.
RWKFILTRST	31	rw	**Remote Wake-Up Packet Filter Register Pointer Reset** When this bit is set, the remote wake-up packet filter register pointer (RWKPTR) is reset to 0. It is automatically cleared after 1 clock cycle.
0	4:3, 8:7, 23:11, 30:29	r	**Reserved** Read as 0; should be written with 0.

14.6.66 Port j MAC RWK Packet Filter

The Remote Wakeup Filter registers are implemented as 8 indirect access registers (wkuppktfilter_reg#i) and accessed by application through MAC_RWK_Packet_Filter register.. When the Remote Wakeup Filters are to be programmed, the entire set of wkuppktfilter_reg registers must be written. The wkuppktfilter_reg register is programmed by sequentially writing the eight register values in MAC_RWK_Packet_Filter register for wkuppktfilter_reg0, wkuppktfilter_reg1, ..., wkuppktfilter_reg7 respectively. The wkuppktfilter_reg register is read in a similar way. The MAC updates the wkuppktfilter_reg register current pointer value in RWKPTR field of MAC_PMT_Control_Status register. The fields of Remote Wakeup Filter are described as follows: Filter i Byte Mask: The filter i byte mask register defines the bytes of the packet that are examined by filter i (0, 1, 2, .., 3) to determine whether or not a packet is a wake-up packet. - The MSB (31st bit) must be zero. - Bit j[30:0] is the byte mask. - If Bit j (byte number) of the byte mask is set, the CRC block processes the Filter i Offset + j of the incoming packet; otherwise Filter i Offset + j is ignored. Filter i Command: The 4-bit filter i command controls the filter i operation. - Bit 3 specifies the address type, defining the destination address type of the pattern. When the bit is set, the pattern applies to only multicast packets; when the bit is reset, the pattern applies only to unicast packet. - Bit 2 (Inverse Mode), when set, reverses the logic of the CRC16 hash function signal, to reject a packet with matching CRC-16 value. Bit 2, along with Bit 1, allows a MAC to reject a subset of remote wake-up packets by creating filter logic such as "Pattern 1 AND NOT Pattern 2". - Bit 1 (And_Previous) implements the Boolean logic. When set, the result of the current entry is logically ANDed with the result of the previous filter. This AND logic allows a filter pattern longer than 32 bytes by splitting the mask among two, three, or four filters. This depends on the number of filters that have the And_Previous bit set. The details are, -- The And_Previous bit setting is applicable within a set of 4 filters. -- Setting of And_Previous bit of filter that is not enabled has no effect, that is setting And_Previous bit of lowest number filter in the set of 4 filters has no effect. For example, setting of And_Previous bit of Filter 0 has no effect. -- If And_Previous bit is set for filter to form AND chained filter, the AND chain breaks at the point any filter is not enabled. For example: If Filter 2 And_Previous bit is set (bit 1 in Filter 2 command is set) but Filter 1 is not enabled (bit 0 in Filter 1 command is reset), then only Filter 2 result is considered. If Filter 2 And_Previous bit is set (bit 1 in Filter 2 command is set), Filter 3 And_Previous bit is set (bit 1 in Filter 3 command is set), but Filter 1 is not enabled (bit 0 in Filter 1 command is reset), then only Filter 2 result ANDed with Filter 3 result is considered. If Filter 2 And_Previous bit is set (bit 1 in Filter 2 command is set), Filter 3 And_Previous bit is set (bit 1 in Filter 3 command is set), but Filter 2 is not enabled (bit 0 in Filter 2 command is reset), then since setting of Filter 2 And_Previous bit has no effect only Filter 1 result ORed with Filter 3 result is considered. -- If filters chained by And_Previous bit setting have complementary programming, then a frame might never pass the AND chained filter. For example, if Filter 2 And_Previous bit is set (bit 1 in Filter 2 command is set), Filter 1 Address_Type bit is set (bit 3 in Filter 1 command is set) indicating multicast detection and Filter 2 Address_Type bit is reset (bit 3 in Filter 2 command is reset) indicating unicast detection or vice versa, a remote wakeup frame does not pass the AND chained filter as a remote wakeup frame cannot be of both unicast and multicast address type. - Bit 0 is the enable for filter i. If Bit 0 is not set, filter i is disabled. Filter i Offset: The filter i offset register defines the offset (within the packet) from which the filter i examines the packets. - This 8-bit pattern-offset is the offset for the filter i first byte to be examined. - The minimum allowed offset is 12, which refers to the 13th byte of the packet. - The offset value 0 refers to the first byte of the packet. Filter i CRC-16: The filter i CRC-16 register contains the CRC-16 value calculated from the pattern and the byte mask programmed in the Remote Wakeup filter register. - The 16-bit CRC calculation uses the following polynomial: G(x) = x^16 + x^15 + x^2 + 1 - Each mask, used in the hash function calculation, is compared with a 16-bit value associated with that mask. Each filter has the following: -- 32-bit Mask: Each bit in this mask corresponds to one byte in the detected packet. If the bit is 1, the corresponding byte is taken into the CRC-16 calculation. -- 8-bit Offset Pointer: Specifies the byte to start the CRC-16 computation. The pointer and the mask are used together to locate the bytes to be used in the CRC-16 calculations.

Portj_MAC_RWK_Packet_Filter (j=0-1)	Offset address:	$100C4_H+j*2000_H$
Port j MAC RWK Packet Filter	Kernel Reset value:	$0000\ 0000_H$



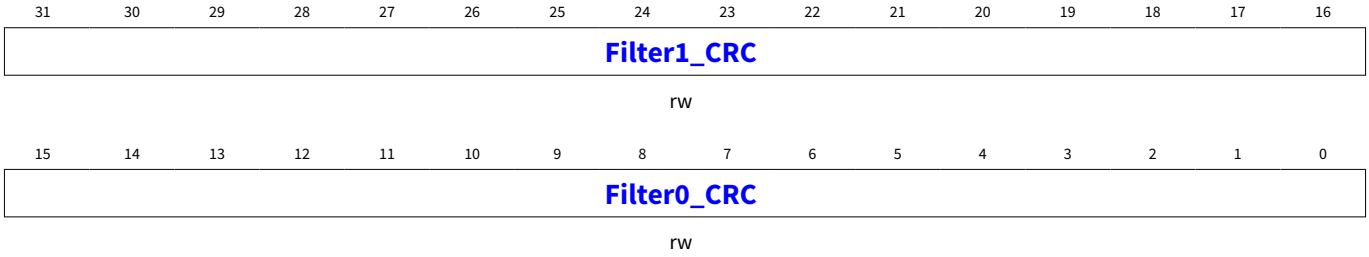
Field	Bits	Type	Description
WKUPFRMFTR	31:0	rw	**Wake-up Packet Filter** This field contains the various controls of RWK Packet filter.

14.6.67 Port j RWK Filter0 1 CRC

Filter i CRC-16. This register contains the CRC-16 to be matched.

Portj_RWK_Filter0_1_CRC (j=0-1) Offset address: $100C4_H+j*2000_H$

Port j RWK Filter0 1 CRC Kernel Reset value: $0000\ 0000_H$



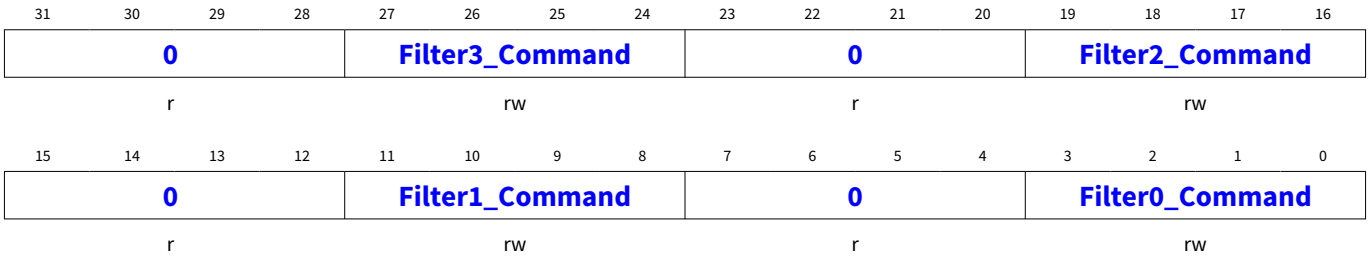
Field	Bits	Type	Description
Filter0_CRC	15:0	rw	**CRC0 This filter CRC-16 contains the CRC_16 value of the pattern** - The 16-bit CRC calculation uses the following polynomial: G(x) = x^16 + x^15 + x^2 + 1
Filter1_CRC	31:16	rw	**CRC1 This filter CRC-16 contains the CRC_16 value of the pattern** - The 16-bit CRC calculation uses the following polynomial: G(x) = x^16 + x^15 + x^2 + 1

14.6.68 Port j RWK Filter0 3 Command

This register controls the filter operation.

Portj_RWK_Filter0_3_Command (j=0-1) Offset address: $100C4_H+j*2000_H$

Port j RWK Filter0 3 Command Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
Filter0_Command	3:0	rw	**Command 0** The 4-bit filter command controls the filter operation. - Bit 3 specifies the address type, defining the destination address type of the pattern. When the bit is set, the pattern applies to only multicast packets; when the bit is reset, the pattern applies only to unicast packet. - Bit 2 (Inverse Mode), when set, reverses the logic of the CRC16 hash function signal, to reject a packet with matching CRC_16 value. - Bit 2, along with Bit 1, allows a MAC to reject a subset of remote wake-up packets by creating filter logic such as "Pattern 1 AND NOT Pattern 2". - Bit 1 (And_Previous) implements the Boolean logic. When set, the result of the current entry is logically ANDed with the result of the previous filter. This AND logic allows a filter pattern longer than 32 bytes by splitting the mask among two, three, or four filters. This depends on the number of filters that have the And_Previous bit set. - Bit 0 is the enable for filter. If Bit 0 is not set, filter is disabled.
Filter1_Command	11:8	rw	**Command1** The 4-bit filter command controls the filter operation. - Bit 3 specifies the address type, defining the destination address type of the pattern. When the bit is set, the pattern applies to only multicast packets; when the bit is reset, the pattern applies only to unicast packet. - Bit 2 (Inverse Mode), when set, reverses the logic of the CRC16 hash function signal, to reject a packet with matching CRC_16 value. - Bit 2, along with Bit 1, allows a MAC to reject a subset of remote wake-up packets by creating filter logic such as "Pattern 1 AND NOT Pattern 2". - Bit 1 (And_Previous) implements the Boolean logic. When set, the result of the current entry is logically ANDed with the result of the previous filter. This AND logic allows a filter pattern longer than 32 bytes by splitting the mask among two, three, or four filters. This depends on the number of filters that have the And_Previous bit set. - Bit 0 is the enable for filter. If Bit 0 is not set, filter is disabled.

(table continues...)

(continued)

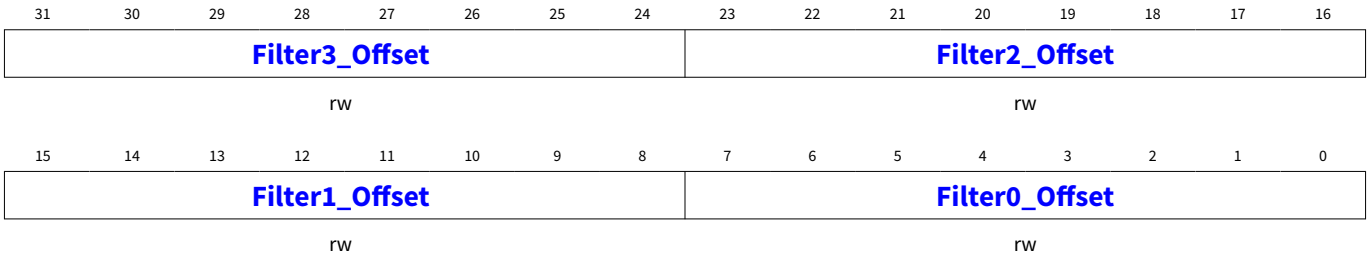
Field	Bits	Type	Description
Filter2_Command	19:16	rw	**Command2** The 4-bit filter command controls the filter operation. - Bit 3 specifies the address type, defining the destination address type of the pattern. When the bit is set, the pattern applies to only multicast packets; when the bit is reset, the pattern applies only to unicast packet. - Bit 2 (Inverse Mode), when set, reverses the logic of the CRC16 hash function signal, to reject a packet with matching CRC_16 value. - Bit 2, along with Bit 1, allows a MAC to reject a subset of remote wake-up packets by creating filter logic such as "Pattern 1 AND NOT Pattern 2". - Bit 1 (And_Previous) implements the Boolean logic. When set, the result of the current entry is logically ANDed with the result of the previous filter. This AND logic allows a filter pattern longer than 32 bytes by splitting the mask among two, three, or four filters. This depends on the number of filters that have the And_Previous bit set. - Bit 0 is the enable for filter. If Bit 0 is not set, filter is disabled.
Filter3_Command	27:24	rw	**Command3 The 4-bit filter command controls the filter operation** - Bit 3 specifies the address type, defining the destination address type of the pattern. When the bit is set, the pattern applies to only multicast packets; when the bit is reset, the pattern applies only to unicast packet. - Bit 2 (Inverse Mode), when set, reverses the logic of the CRC16 hash function signal, to reject a packet with matching CRC_16 value. - Bit 2, along with Bit 1, allows a MAC to reject a subset of remote wake-up packets by creating filter logic such as "Pattern 1 AND NOT Pattern 2". - Bit 1 (And_Previous) implements the Boolean logic. When set, the result of the current entry is logically ANDed with the result of the previous filter. This AND logic allows a filter pattern longer than 32 bytes by splitting the mask among two, three, or four filters. This depends on the number of filters that have the And_Previous bit set. - Bit 0 is the enable for filter. If Bit 0 is not set, filter is disabled.
0	7:4, 15:12, 23:20, 31:28	r	**Reserved** Read as 0; should be written with 0.

14.6.69 Port j RWK Filter0 3 Offset

This register defines the offset (within the packet) from which the filter examines the packet.

Portj_RWK_Filter0_3_Offset (j=0-1) Offset address: $100C4_H+j*2000_H$

Port j RWK Filter0 3 Offset Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
Filter0_Offset	7:0	rw	**Offset0** This filter offset defines the offset (within the packet) from which the filter examines the packets. - This 8-bit pattern-offset is the offset for the filter first byte to be examined. - The minimum allowed offset is 12, which refers to the 13th byte of the packet. - The offset value 0 refers to the first byte of the packet.
Filter1_Offset	15:8	rw	**Offset1 This filter offset defines the offset (within the packet) from which the filter examines the packets** - This 8-bit pattern-offset is the offset for the filter first byte to be examined. - The minimum allowed offset is 12, which refers to the 13th byte of the packet. - The offset value 0 refers to the first byte of the packet.
Filter2_Offset	23:16	rw	**Offset2 This filter offset defines the offset (within the packet) from which the filter examines the packets** - This 8-bit pattern-offset is the offset for the filter first byte to be examined. - The minimum allowed offset is 12, which refers to the 13th byte of the packet. - The offset value 0 refers to the first byte of the packet.
Filter3_Offset	31:24	rw	**Offset3 This filter offset defines the offset (within the packet) from which the filter examines the packets** - This 8-bit pattern-offset is the offset for the filter first byte to be examined. - The minimum allowed offset is 12, which refers to the 13th byte of the packet. - The offset value 0 refers to the first byte of the packet.

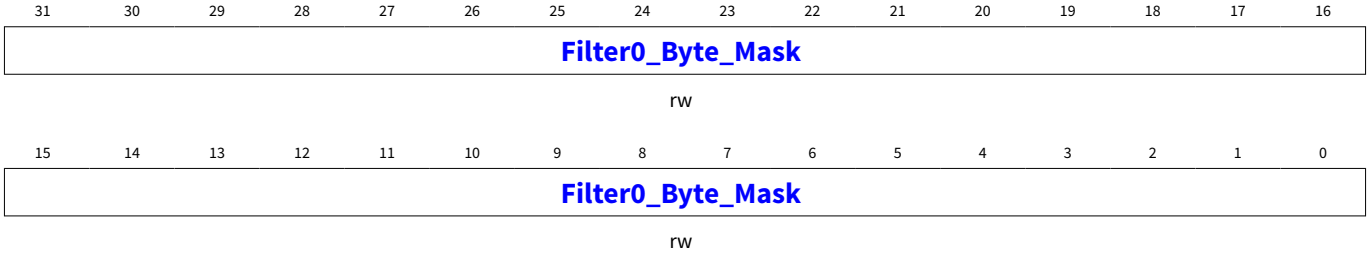
14.6.70 Port j RWK Filter0 Byte Mask

Filter i Byte Mask. The filter i byte mask register defines the bytes of the packet that are examined by filter i to determine whether or not a packet is a wake-up packet. The MSB (31st bit) must be zero. Bit j[30:0] is the byte mask. If Bit j (byte number) of the byte mask is set, the CRC block processes the byte pointed by Filter i Offset + j of the incoming packet; otherwise Filter i Offset + j is ignored.

You can access these registers indirectly through the MAC_RWK_Packet_Filter register.

Portj_RWK_Filter0_Byte_Mask (j=0-1) Offset address: $100C4_H+j*2000_H$

Port j RWK Filter0 Byte Mask Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
Filter0_Byte_Mask	31:0	rw	**Byte Mask** Each bit in this mask corresponds to one byte in the detected packet. If the bit is 1', the corresponding byte is taken into the CRC16 calculation.

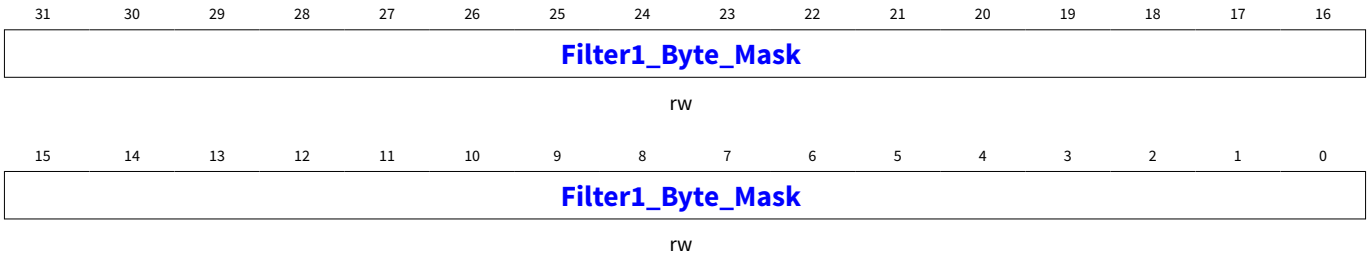
14.6.71 Port j RWK Filter1 Byte Mask

Filter i Byte Mask. The filter i byte mask register defines the bytes of the packet that are examined by filter i to determine whether or not a packet is a wake-up packet. The MSB (31st bit) must be zero. Bit j[30:0] is the byte mask. If Bit j (byte number) of the byte mask is set, the CRC block processes the byte pointed by Filter i Offset + j of the incoming packet; otherwise Filter i Offset + j is ignored.

You can access these registers indirectly through the MAC_RWK_Packet_Filter register.

Portj_RWK_Filter1_Byte_Mask (j=0-1) Offset address: $100C4_H+j*2000_H$

Port j RWK Filter1 Byte Mask Kernel Reset value: $0000\ 0000_H$



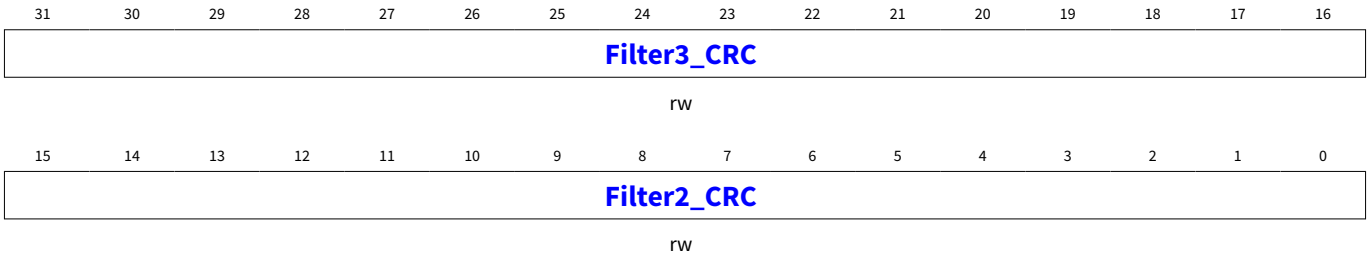
Field	Bits	Type	Description
Filter1_Byte_Mask	31:0	rw	**Byte Mask** Each bit in this mask corresponds to one byte in the detected packet. If the bit is 1', the corresponding byte is taken into the CRC16 calculation.

14.6.72 Port j RWK Filter2 3 CRC

Filter i CRC-16. This register contains the CRC-16 to be matched.

Portj_RWK_Filter2_3_CRC (j=0-1) Offset address: $100C4_H+j*2000_H$

Port j RWK Filter2 3 CRC Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
Filter2_CRC	15:0	rw	**CRC0 This filter CRC-16 contains the CRC_16 value of the pattern** - The 16-bit CRC calculation uses the following polynomial: G(x) = x^16 + x^15 + x^2 + 1
Filter3_CRC	31:16	rw	**CRC1 This filter CRC-16 contains the CRC_16 value of the pattern** - The 16-bit CRC calculation uses the following polynomial: G(x) = x^16 + x^15 + x^2 + 1

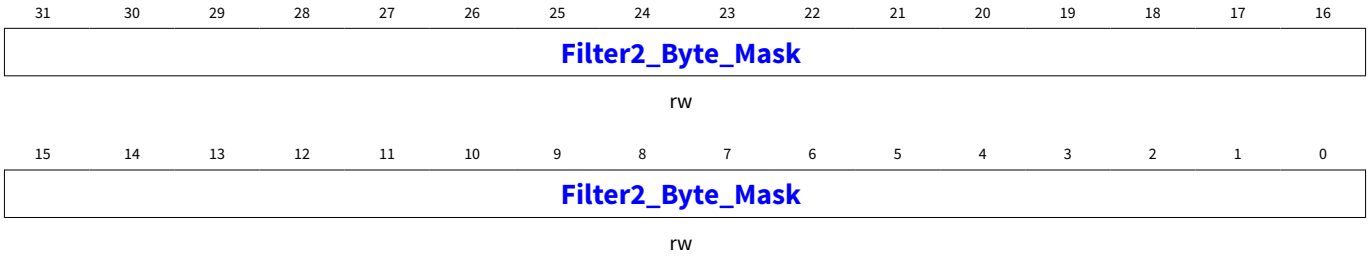
14.6.73 Port j RWK Filter2 Byte Mask

Filter i Byte Mask. The filter i byte mask register defines the bytes of the packet that are examined by filter i to determine whether or not a packet is a wake-up packet. The MSB (31st bit) must be zero. Bit j[30:0] is the byte mask. If Bit j (byte number) of the byte mask is set, the CRC block processes the byte pointed by Filter i Offset + j of the incoming packet; otherwise Filter i Offset + j is ignored.

You can access these registers indirectly through the MAC_RWK_Packet_Filter register.

Portj_RWK_Filter2_Byte_Mask (j=0-1) Offset address: $100C4_H+j*2000_H$

Port j RWK Filter2 Byte Mask Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
Filter2_Byte_Mask	31:0	rw	**Byte Mask** Each bit in this mask corresponds to one byte in the detected packet. If the bit is 1', the corresponding byte is taken into the CRC16 calculation.

14.6.74 Port j RWK Filter3 Byte Mask

Filter i Byte Mask. The filter i byte mask register defines the bytes of the packet that are examined by filter i to determine whether or not a packet is a wake-up packet. The MSB (31st bit) must be zero. Bit j[30:0] is the byte mask. If Bit j (byte number) of the byte mask is set, the CRC block processes the byte pointed by Filter i Offset + j of the incoming packet; otherwise Filter i Offset + j is ignored.

You can access these registers indirectly through the MAC_RWK_Packet_Filter register.

Portj_RWK_Filter3_Byte_Mask (j=0-1) Offset address: $100C4_H+j*2000_H$

Port j RWK Filter3 Byte Mask Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Filter3_Byte_Mask															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Filter3_Byte_Mask															
rw															

Field	Bits	Type	Description
Filter3_Byte_Mask	31:0	rw	**Byte Mask** Each bit in this mask corresponds to one byte in the detected packet. If the bit is 1', the corresponding byte is taken into the CRC16 calculation.

14.6.75 Port j MAC LPI Control Status

The LPI Control and Status register controls the LPI functions and provides the LPI interrupt status. The status bits are cleared when this register is read.

Portj_MAC_LPI_Control_Status (j=0-1) Offset address: $100D0_H+j*2000_H$

Port j MAC LPI Control Status Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0										LPITCSE	LPIATE	LPITXA	PLSDIS	PLS	LPITXEN
r										rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				0	0	RLPIST	TLPIST	0				RLPIEX	RLPIEN	TLPIEX	TLPIEN
r				r	r	r	r	r				r	r	r	r

Field	Bits	Type	Description
TLPIEN	0	r	**Transmit LPI Entry** When this bit is set, it indicates that the MAC Transmitter has entered the LPI state because of the setting of the LPITXEN bit. Access restriction applies.This bit is cleared by a read into this register (or this bit is written to 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set).

(table continues...)

(continued)

Field	Bits	Type	Description
TLPIEX	1	r	**Transmit LPI Exit** When this bit is set, it indicates that the MAC transmitter exited the LPI state after the application cleared the LPITXEN bit and the LPI TW Timer has expired. Access restriction applies.This bit is cleared by a read into this register (or this bit is written to 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set).
RLPIEN	2	r	**Receive LPI Entry** When this bit is set, it indicates that the MAC Receiver has received an LPI pattern and entered the LPI state. Access restriction applies.This bit is cleared by a read into this register (or this bit is written to 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Note: This bit might not be set if the MAC receives the LPI pattern for a very short duration, such as, less than three clock cycles of CSR clock.
RLPIEX	3	r	**Receive LPI Exit** When this bit is set, it indicates that the MAC Receiver has stopped receiving the LPI pattern on the GMII or XGMII interface, exited the LPI state, and resumed the normal reception. Access restriction applies.This bit is cleared by a read into this register (or this bit is written to 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Note: This bit might not be set if the MAC stops receiving the LPI pattern for a very short duration, such as, less than three clock cycles of CSR clock.
TLPIST	8	r	**Transmit LPI State** When this bit is set, it indicates that the MAC is transmitting the LPI pattern on the GMII or XGMII interface.
RLPIST	9	r	**Receive LPI State** When this bit is set, it indicates that the MAC is receiving the LPI pattern on the GMII or XGMII interface.
LPITXEN	16	rw	**LPI Transmit Enable** When this bit is set, it instructs the MAC Transmitter to enter the LPI state. When this bit is reset, it instructs the MAC to exit the LPI state and resume normal transmission. This bit is cleared when the LPITXA bit is set and the MAC exits the LPI state because of the arrival of a new packet for transmission.
PLS	17	rw	**PHY Link Status** This bit indicates the link status of the PHY. The MAC Transmitter asserts the LPI pattern only when the link status is up (OKAY) at least for the time indicated by the LPI LS TIMER. When this bit is set, the link is considered to be okay (UP) and when this bit is reset, the link is considered to be down.

(table continues...)

(continued)

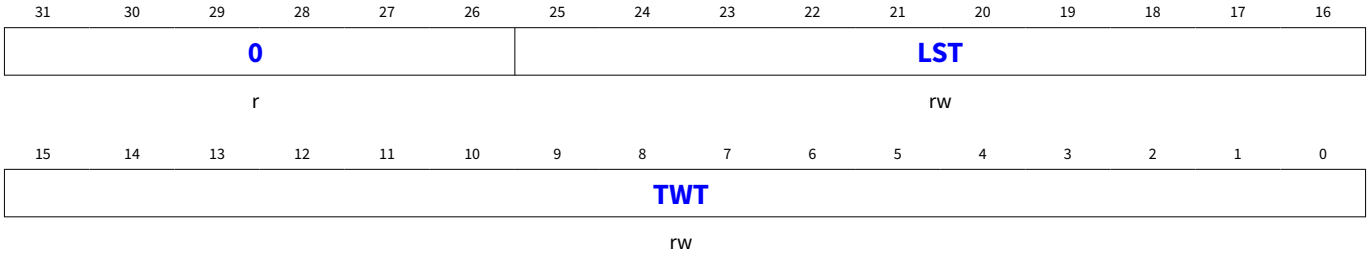
Field	Bits	Type	Description
PLSDIS	18	rw	**PHY Link Status Disable** This bit disables the link status received on the XGMII receive paths, which are used for activating the LPI LS TIMER. When set to 1, the MAC ignores the link status from the XGMII interface and uses the link status from bit 17 (PLS). When set to 0, the MAC uses the link status from the XGMII in the 10G mode. In the GMII or 1G mode, the controller does not use this bit. The controller depends on bit 17 (PLS) for activating the LPI LS TIMER.
LPITXA	19	rw	**LPI Tx Automate** This bit controls the behavior of the MAC when it is entering or coming out of the LPI mode on the Transmit side. If the LPITXA and LPITXEN bits are set to 1, the MAC enters the LPI mode only after all outstanding packets (in the controller) and pending packets (in the application interface) have been transmitted. The MAC comes out of the LPI mode when the application sends any packet for transmission or the application issues a Tx FIFO Flush command. In addition, the MAC automatically clears the LPITXEN bit when it exits the LPI state. If Tx FIFO Flush is set in the FTQ bit of MTL_TxQ0_Operation_Mode register, when the MAC is in the LPI mode, it exits the LPI mode. When this bit is 0, the LPITXEN bit directly controls the behavior of the MAC when it enters or comes out of the LPI mode.
LPIATE	20	rw	**LPI Auto Timer Enable** This bit controls the automatic entry of the MAC Transmitter into and exit out of the LPI state. When LPIATE, LPITXA, and LPITXEN bits are set, the MAC Transmitter enters LPI state only when the complete MAC Tx data path is IDLE for a period indicated by the MAC_LPI_Auto_Entry_Timer register. After entering LPI state, if the data path becomes non-IDLE (due to a new packet being accepted for transmission), the Transmitter exits LPI state but does not clear LPITXEN bit. This enables the re-entry into LPI state when it is IDLE again. When LPIATE is 0, the LPI Auto timer is disabled and MAC Transmitter enters LPI state based on the settings of LPITXA and LPITXEN bit descriptions.
LPITCSE	21	rw	**LPI Tx Clock Stop Enable** When this bit is 0, the output transmit clock gating signal is not asserted. When this bit is 1, the output transmit clock gating signal is asserted during Transmit LPI mode which can be used to Gate the Transmit Clock.
0	7:4, 10, 11, 15:12, 31:22	r	**Reserved** Read as 0; should be written with 0.

14.6.76 Port j MAC LPI Timers Control

The LPI Timers Control register controls the timeout values in the LPI states. It specifies the time for which the MAC transmits the LPI pattern and also the time for which the MAC waits before resuming the normal transmission.

Portj_MAC_LPI_Timers_Control (j=0-1) Offset address: $100D4_H+j*2000_H$

Port j MAC LPI Timers Control Kernel Reset value: $03E8\ 0000_H$



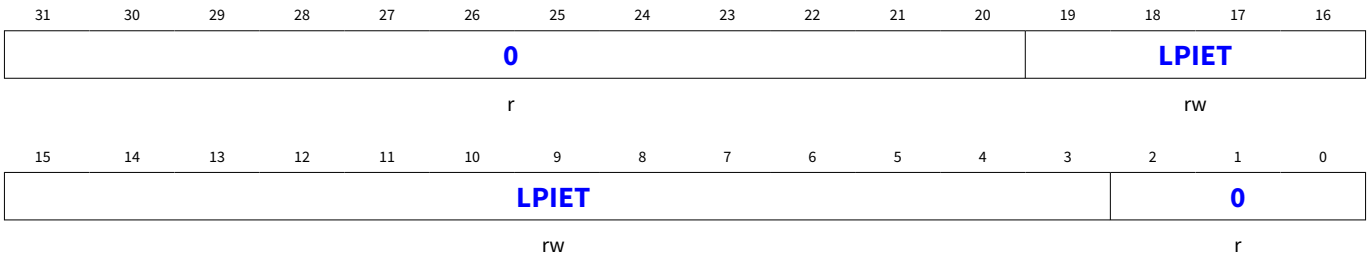
Field	Bits	Type	Description
TWT	15:0	rw	**LPI TW TIMER** This field specifies the minimum time (in microseconds) for which the MAC waits after it stops transmitting the LPI pattern to the PHY and before it resumes the normal transmission. The TLPIEX status bit is set after the expiry of this timer.
LST	25:16	rw	**LPI LS TIMER** This field specifies the minimum time (in milliseconds) for which the link status from the PHY must be up (OKAY) before the LPI pattern can be transmitted to the PHY. The MAC does not transmit the LPI pattern even when the LPITXEN bit is set unless the LPI LS Timer reaches the programmed terminal count. The default value of the LPI LS Timer is 1000 (1 sec) as defined in the IEEE standard.
0	31:26	r	**Reserved** Read as 0; should be written with 0.

14.6.77 Port j MAC LPI Auto Entry Timer

This register is used to store the LPI Auto Entry Timer Value in Micro-Seconds.

Portj_MAC_LPI_Auto_Entry_Timer (j=0-1) Offset address: $100D8_H+j*2000_H$

Port j MAC LPI Auto Entry Timer Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
LPIET	19:3	rw	**LPI Entry Timer** This field is programmed with a value in micro-seconds. When LPIATE, LPITXA, and LPITXEN bits of the MAC_LPI_Control_Status are set, DWC_xgmac automatically enters Tx LPI mode after transmitter is IDLE for LPIET amount of time. Bits[2:0] are read-only such that the granularity of this timer is 8 micro-seconds.
0	2:0, 31:20	r	**Reserved** Read as 0; should be written with 0.

14.6.78 Port j MAC 1US Tic Counter

This register controls the generation of the Reference time (1micro-second tic) for all the LPI timers. This timer has to be programmed by the software initially.

Portj_MAC_1US_Tic_Counter (j=0-1) Offset address: $100DC_H+j*2000_H$

Port j MAC 1US Tic Counter Kernel Reset value: $0000\ 0063_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				TIC_1US_CNTR											
r				rw											

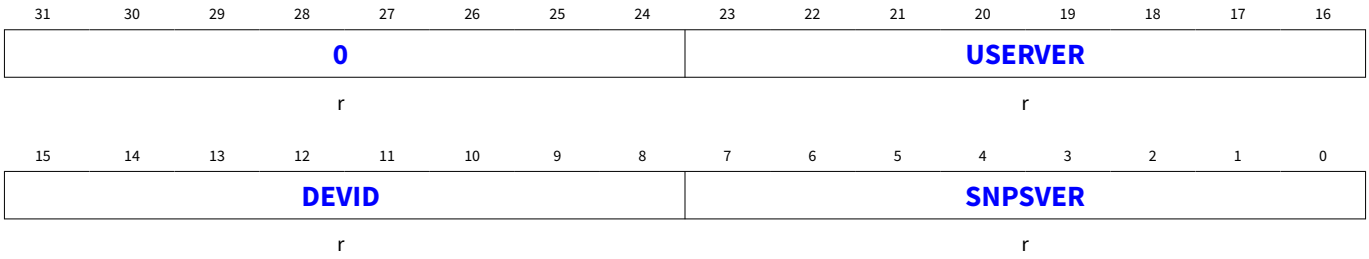
Field	Bits	Type	Description
TIC_1US_CNTR	11:0	rw	**1US TIC Counter** This counter must be programmed with the number of clock cycles of CSR clock that constitutes a period of 1us. (Subtract 1 from the value before programming). For example if the CSR clock frequency is 100MHz, then this field needs to be programmed to value 100 - 1 = 99 (which is 0x63). This is required to generate the 1US events which are used to update some of the EEE related timers and counters.
0	31:12	r	**Reserved** Read as 0; should be written with 0.

14.6.79 Port j MAC Version

The version register identifies the version of DWC_xgmac. This register contains two bytes: one that Synopsys uses to identify the controller release number, and the other is user defined

Portj_MAC_Version (j=0-1) Offset address: $10110_H+j*2000_H$

Port j MAC Version Kernel Reset value: $0010\ 7632_H$



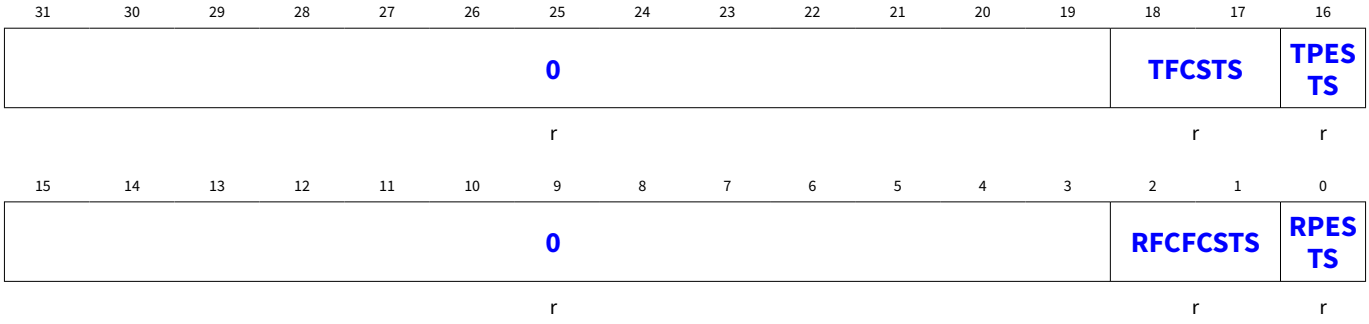
Field	Bits	Type	Description
SNPSVER	7:0	r	**Synopsys-defined Version**
DEVID	15:8	r	**Indicates the Device family**
USERVER	23:16	r	**User-defined Version**
0	31:24	r	**Reserved** Read as 0; should be written with 0.

14.6.80 Port j MAC Debug

The Debug register provides the debug status of various MAC blocks.

Portj_MAC_Debug (j=0-1) Offset address: $10114_H+j*2000_H$

Port j MAC Debug Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
RPESTS	0	r	**MAC, GMII, or XGMII Receive Protocol Engine Status** When this bit is set, it indicates that the MAC, GMII, or XGMII receive protocol engine is actively receiving data, and it is not in the Idle state.
RFCFCSTS	2:1	r	**MAC Receive Packet Controller FIFO Status** This field indicates the active state of the small FIFO Read and Write controllers of the MAC Receive Packet Controller module. When bit 2 is set, small FIFO Read controller is active. When bit 1 is set, small FIFO Write controller is active.
TPESTS	16	r	**MAC GMII or XGMII Transmit Protocol Engine Status** When this bit is set, it indicates that the MAC GMII or XGMII transmit protocol engine is actively transmitting data, and it is not in the Idle state.

(table continues...)

(continued)

Field	Bits	Type	Description
TFCSTS	18:17	r	**MAC Transmit Packet Controller Status** This field indicates the state of the MAC Transmit Packet Controller module: - 2'b00: Idle state - 2'b01: Waiting for one of the following: -- Status of the previous packet or -- IPG period to be over - 2'b10: Generating and transmitting a Pause control packet - 2'b11: Transferring input packet for transmission
0	15:3, 31:19	r	**Reserved** Read as 0; should be written with 0.

14.6.81 Port j MAC HW Feature0

This register indicates the presence of the first set of optional features or functions of DWC_xgmac. The software driver can use this register to dynamically enable or disable the programs related to the optional blocks.

Portj_MAC_HW_Feature0 (j=0-1) Offset address: $1011C_H+j*2000_H$

Port j MAC HW Feature0 Kernel Reset value: $8E7D\ 79FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EDMA	EDIFFC	VXN	0	SAVLANINS	TSTSSEL		PHYSEL		ADDMACADRSEL					0	RXCOESEL
r	r	r	r	r	r		r		r					r	r

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	TXCOESEL	EEESEL	TSSEL	AVSEL	RAVSEL	ARPOFFSEL	MMCSEL	MGKSEL	RWKSEL	SMASEL	VLHASH	HDSEL	RMIISEL	GMIISEL	RGMIISEL
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

Field	Bits	Type	Description
RGMIISEL	0	r	**RGMII Support** This bit is set to 1 when the RGMII Interface option is selected.
GMIISEL	1	r	**1000/100/10 Mbps Support** This bit is set to 1 when the GMII Interface option is selected.
RMIISEL	2	r	**RMII Support** This bit is set to 1 when the RMII Interface option is selected.
HDSEL	3	r	**Half-duplex Support** This bit is set to 1 when the half-duplex mode is selected. 0_B **INACTIVE**: No Half-duplex support 1_B **ACTIVE**: Half-duplex support

(table continues...)

(continued)

Field	Bits	Type	Description
VLHASH	4	r	**VLAN Hash Filter Selected** This bit is set to 1 when the Enable Address Filter VLAN Hash Table option is selected.
SMASEL	5	r	**SMA (MDIO) Interface** This bit is set to 1 when the Enable Station Management Block (MDIO Interface) option is selected.
RWKSEL	6	r	**PMT Remote Wake-up Packet Enable** This bit is set to 1 when the Enable Remote Wake-Up Packet Detection option is selected.
MGKSEL	7	r	**PMT Magic Packet Enable** This bit is set to 1 when the Enable Magic Packet Detection option is selected.
MMCSEL	8	r	**RMON Module Enable** This bit is set to 1 when the Enable XGMAC Management Counter (MMC) option is selected.
ARPOFFSEL	9	r	**ARP Offload Enabled** This bit is set to 1 when the Enable IPv4 ARP Offload option is selected.
RAVSEL	10	r	**Rx Side Only AV Feature Enable** This bit is set to 1 when the Enable Audio Video Bridging option on Rx Side Only is selected.
AVSEL	11	r	**AV Feature Enabled** This bit is set to 1 when the Enable Audio Video Bridging option is selected.
TSSEL	12	r	**IEEE 1588-2008 Timestamp Enabled** This bit is set to 1 when the Enable IEEE 1588 Timestamp Support option is selected.
EEESEL	13	r	**Energy Efficient Ethernet Enabled** This bit is set to 1 when the Enable Energy Efficient Ethernet (EEE) option is selected.
TXCOESEL	14	r	**Transmit Checksum Offload Enabled** This bit is set to 1 when the Enable Transmit TCP/IP Checksum Offload option is selected.
RXCOESEL	16	r	**Receive Checksum Offload Enabled** This bit is set to 1
ADDMACADRSEL	22:18	r	**MAC Addresses 1-31 Selected** This field gives the number of additional MAC addresses selected using the Enable Additional 1-31 MAC Address Registers option.

(table continues...)

(continued)

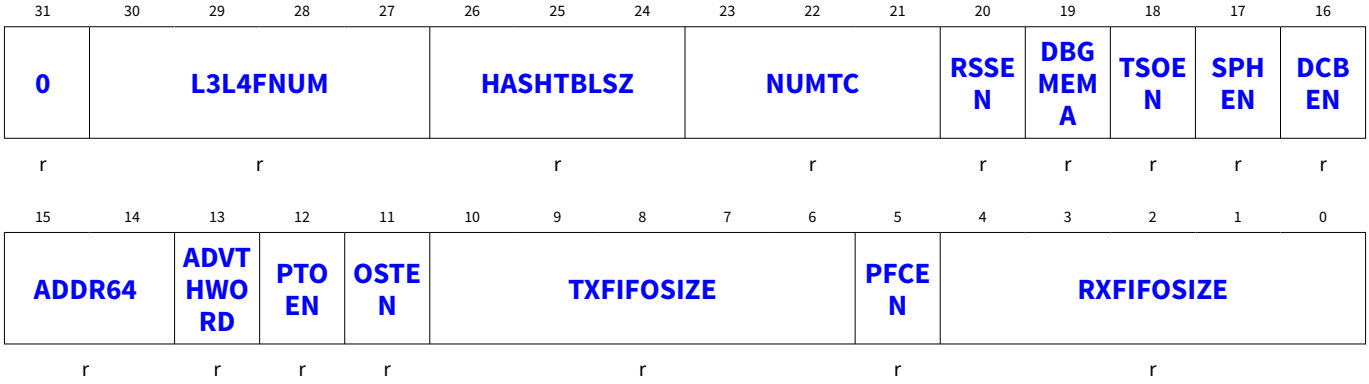
Field	Bits	Type	Description
PHYSEL	24:23	r	**RGMII Interface Select** This field reflect the IO value of RGMII Interface signal which when set to 1 indicates RGMII interface is selected. - 00: Reserved - 01: RGMII interface - 10: RMII Interface - 11: Reserved
TSSTSSEL	26:25	r	**Timestamp System Time Source** This bit indicates the source of the Timestamp system time: - 01: Internal - 10: External - 11: Both - 00: Reserved
SAVLANINS	27	r	**Source Address or VLAN Insertion Enable** This bit is set to 1 when the Enable SA and VLAN Insertion on Tx option is selected.
VXN	29	r	**Reserved**
EDIFFC	30	r	**Different Descriptor Cache** When set to 1, then DMA mode Separate Memory is selected for the Descriptor Cache.
EDMA	31	r	**Enhanced DMA** This bit is always set to 1 when AXI-DMA interface is present in DWC_xgmac.
0	15, 17, 28	r	**Reserved** Read as 0; should be written with 0.

14.6.82 Port j MAC HW Feature1

This register indicates the presence of second set of the optional features or functions of DWC_xgmac. The software driver can use this register to dynamically enable or disable the programs related to the optional blocks.

Portj_MAC_HW_Feature1 (j=0-1) Offset address: $10120_H+j*2000_H$

Port j MAC HW Feature1 Kernel Reset value: $40EA\ 2A08_H$



Field	Bits	Type	Description
RXFIFOSIZE	4:0	r	**MTL Receive FIFO Size** This field contains the configured value of MTL Rx FIFO in bytes expressed as Log to base 2 minus 7, that is, Log2(MTL Receive FIFO Size in bytes) - 7: - 4'b00011: 1024 bytes - 4'b00100: 2048 bytes - 4'b00101: 4096 bytes - 4'b00110: 8192 bytes - 4'b00111: 16384 bytes - 4'b01000: 32 KB - 4'b01001: 64 KB - 4'b01010: 128 KB - 4'b01011: 256 KB - 4'b01100-4'b11111: Reserved
PFCEN	5	r	**Reserved**
TXFIFOSIZE	10:6	r	**MTL Transmit FIFO Size** This field contains the configured value of MTL Tx FIFO in bytes expressed as Log to base 2 minus 7, that is, Log2(MTL Transmit FIFO Size in bytes) - 7: - 4'b00011: 1024 bytes - 4'b00100: 2048 bytes - 4'b00101: 4096 bytes - 4'b00110: 8192 bytes - 4'b00111: 16384 bytes - 4'b01000: 32 KB - 4'b01001: 64 KB - 4'b01010: 128 KB - 4'b01011: 256 KB - 4'b01100-4'b11111: Reserved
OSTEN	11	r	**One-Step Timestamping Enable** This bit is set to 1 when the Enable One-Step Timestamp Feature is selected.
PTOEN	12	r	**Reserved**

(table continues...)

(continued)

Field	Bits	Type	Description
ADVTHWORD	13	r	**IEEE 1588 High Word Register Enable** This bit is set to 1 when the Add IEEE 1588 Higher Word Register option is selected.
ADDR64	15:14	r	**Address Width** This field indicates the configured address width: - 2'b00: 32 - 2'b01: 40 - 2'b10: 48 - 2'b11: Reserved
DCBEN	16	r	**Reserved**
SPHEN	17	r	**Header-Payload Split Feature Enable** This bit is set to 1
TSOEN	18	r	**Reserved**
DBGMEMA	19	r	**Debug Memory Interface Enabled** This bit is set to 1 when the Enable Debug Memory Access option is selected.
RSSEN	20	r	**Reserved**
NUMTC	23:21	r	**Number of Traffic Classes** This field indicates the number of traffic classes selected: - 3'b000: 1 Traffic Class - 3'b001: 2 Traffic Classes - 3'b010: 3 Traffic Classes - ... - 3'b111: 8 Traffic
HASHTBLSZ	26:24	r	**Hash Table Size** This field indicates the size of the hash table: - 3'b000: No hash table selected - 3'b001: 64 - 3'b010: 128 - 3'b011: 256 - 3'b100: 512 - 3'b101: 1024 - 3'b110: 2048 - 3'b111: 4096

(table continues...)

(continued)

Field	Bits	Type	Description
L3L4FNUM	30:27	r	**Total number of L3 or L4 Filters** This field indicates the total number of L3 or L4 filters: - 4'b0000: No L3 or L4 Filter - 4'b0001: 1 L3 or L4 Filter - 4'b0010: 2 L3 or L4 Filters - ... - 4'b1000: 8 L3 or L4 - 4'b1001: 16 L3 or L4 - 4'b1010: 32 L3 or L4
0	31	r	**Reserved** Read as 0; should be written with 0.

14.6.83 Port j MAC HW Feature2

This register indicates the presence of the third set of optional features or functions of DWC_xgmac. The software driver can use this register to dynamically enable or disable the programs related to the optional blocks.

Portj_MAC_HW_Feature2 (j=0-1) Offset address: $10124_H+j*2000_H$

Port j MAC HW Feature2 Kernel Reset value: $021C\ 71C7_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	AUXSNAPNUM			0	PPSOUTNUM			0		TXCHCNT				0	
r	r			r	r			r		r				r	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXCHCNT				0		TXQCNT				0		RXQCNT			
r				r		r				r		r			

Field	Bits	Type	Description
RXQCNT	3:0	r	**Number of MTL Receive Queues** This field indicates the number of MTL Receive queues: - 4'b0000: 1 MTL Rx Queue - 4'b0001: 2 MTL Rx Queues - ... - 4'b1011: 12 MTL Rx Queues
TXQCNT	9:6	r	**Number of MTL Transmit Queues** This field indicates the number of MTL Transmit queues: - 4'b0000: 1 MTL Tx Queue - 4'b0001: 2 MTL Tx Queues - ... - 4'b1111: 16 MTL Tx Queues

(table continues...)

(continued)

Field	Bits	Type	Description
RXCHCNT	15:12	r	**Number of DMA Receive Channels** This field indicates the number of DMA Receive channels: - 4'b0000: 1 DMA Rx Channel - 4'b0001: 2 DMA Rx Channels - ... - 4'b1111: 16 DMA Rx Channels
TXCHCNT	21:18	r	**Number of DMA Transmit Channels** This field indicates the number of DMA Transmit channels: - 4'b0000: 1 DMA Tx Channel - 4'b0001: 2 DMA Tx Channels - ... - 4'b1111: 16 DMA Tx Channels
PPSOUTNUM	26:24	r	**Number of PPS Outputs** This field indicates the number of PPS outputs: - 3'b000: No PPS output - 3'b001: 1 PPS output - 3'b010: 2 PPS outputs - 3'b011: 3 PPS outputs - 3'b100: 4 PPS outputs - 3'b101 - 3'b111: Reserved
AUXSNAPNUM	30:28	r	**Reserved**
0	5:4, 11:10, 17:16, 23:22, 27, 31	r	**Reserved** Read as 0; should be written with 0.

14.6.84 Port j MAC HW Feature3

This register indicates the presence of the fourth set of optional features or functions of DWC_xgmac. The software driver can use this register to dynamically enable or disable the programs related to the optional blocks.

Portj_MAC_HW_Feature3 (j=0-1) Offset address: $10128_H+j*2000_H$

Port j MAC HW Feature3 Kernel Reset value: $7FBD\ 759A_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TBS_CH				TBSSEL	FPESEL	SGFSEL	GCLWID		GCLDEP			ESTSEL	TTSFD		
r				r	r	r	r		r			r	r		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ASP		DVLAN	FRPES		FRPPB		POUOST	FRPPIPE			CBTISEL	FRPSEL	NRVF		
r		r	r		r		r	r			r	r	r		

Field	Bits	Type	Description
NRVF	2:0	r	**Enabled number of Extended VLAN Tag Filters or External VLAN tag lookup size** This field indicates selected Number of Extended VLAN Tag Filters or External VLAN tag lookup size: - 3'b000: No Extended Rx VLAN Filters - 3'b001: 4 Extended Rx VLAN Filters - 3'b010: 8 Extended Rx VLAN Filters - 3'b011: 16 Extended Rx VLAN Filters - 3'b100: 24 Extended Rx VLAN Filters - 3'b101: 32 Extended Rx VLAN Filters - 3'b110: Reserved - 3'b111: External Receive 12bit VLAN Tag Lookup is selected
FRPSEL	3	r	**Supported Flexible Receive Parser** This bit is set to 1 when the Enable Flexible Programmable Receive Parser option is selected. 0_B **INACTIVE**: Flexible Receive Parser feature is not selected 1_B **ACTIVE**: Flexible Receive Parser feature is selected
CBTISEL	4	r	**Queue/Channel based VLAN tag insertion on Tx Enable** This bit is set to 1 when the Enable Queue/Channel based VLAN tag insertion on Tx Feature is selected.
FRPPIPE	7:5	r	**Supported Parallel Instruction Processor Engines (PIPEs)** This field indicates the maximum number of Instruction Processors supported by flexible receive parser. Values: - 0x0: 1 PIPE - 0x1: 2 PIPEs - ... - 0x15: 16 PIPEs
POUOST	8	r	**One Step for PTP over UDP/IP Feature Enable** This bit is set to 1 when the Enable One step timestamp for PTP over UDP/IP feature is selected.
FRPPB	10:9	r	**Supported Flexible Receive Parser Parsable Bytes** This field indicates the supported Max Number of bytes of the packet data to be Parsed by Flexible Receive Parser. 00_B **M_64BYTES**: 64 Bytes 01_B **M_128BYTES**: 128 Bytes 10_B **M_256BYTES**: 256 Bytes 11_B **RSVD**: Reserved

(table continues...)

(continued)

Field	Bits	Type	Description
FRPES	12:11	r	**Supported Flexible Receive Parser Instructions** This field indicates the Max Number of Parser Instructions supported by Flexible Receive Parser. 00_B **M_64ENTR**: 64 Instructions 01_B **M_128ENTR**: 128 Instructions 10_B **M_256ENTR**: 256 Instructions 11_B **RSVD**: Reserved
DVLAN	13	r	**Double VLAN Processing Enabled** This bit is set to 1 when the Enable Double VLAN Processing feature is selected.
ASP	15:14	r	**Automotive Safety Package** This field along with EASP field of MAC_HW_Feature4 register indicates the encoded value for the different Safety features: {EASP,ASP}: Description - 3'b000: No Safety features selected - 3'b001: All the Automotive Safety features are selected with ECC and Parity Ports on External Application Interface for Data and Control respectively - 3'b010: All the Automotive Safety features are selected without ECC or Parity Ports on External Application Interface - 3'b011: All the Automotive Safety features are selected with the Parity Ports on External Application Interface - 3'b100: Only ECC feature selected
TTSFD	18:16	r	**Tx Timestamp FIFO Depth** This value indicates the depth of the Tx Timestamp FIFO - 3'b000: Reserved - 3'b001: 1 - 3'b010: 2 - 3'b011: 4 - 3'b100: 8 - 3'b101: 16 - 3'b110: Reserved - 3'b111: Reserved
ESTSEL	19	r	**Enhancements to Scheduling Traffic Enable** This bit is set to 1 when the Enable Enhancements to Scheduling Traffic feature is selected. 0_B **INACTIVE**: Enable Enhancements to Scheduling Traffic feature is not selected 1_B **ACTIVE**: Enable Enhancements to Scheduling Traffic feature is selected

(table continues...)

(continued)

Field	Bits	Type	Description
GCLDEP	22:20	r	**Depth of the Gate Control List** This field indicates the depth of Gate Control list expressed as Gate Control List Depth selected x2 000_B **NODEPTH**: No Depth configured 001_B **DEPTH64**: 64 010_B **DEPTH128**: 128 011_B **DEPTH256**: 256 100_B **DEPTH512**: 512 101_B **DEPTH1024**: 1024 110_B **RSVD**: Reserved **others,** Reserved
GCLWID	24:23	r	**Width of the Time Interval field in the Gate Control List** This field indicates the width of the Configured Time Interval Field 00_B **NOWIDTH**: Width not configured 01_B **WIDTH16**: 16 10_B **WIDTH20**: 20 11_B **WIDTH24**: 24
SGFSEL	25	r	**Per-Stream Filtering Select** This bit is set to 1 when the Per-Stream Filtering feature is selected. 0_B **INACTIVE**: Per Stream Filtering feature is not selected 1_B **ACTIVE**: Per Stream Filtering feature is selected
FPESEL	26	r	**Frame Preemption Enable** This bit is set to 1 0_B **INACTIVE**: Frame Preemption Enable feature is not selected 1_B **ACTIVE**: Frame Preemption Enable feature is selected
TBSSEL	27	r	**Time Based Scheduling Enable** This bit is set to 1 when the Time Based Scheduling feature is selected. 0_B **INACTIVE**: Time Based Scheduling Enable feature is not selected 1_B **ACTIVE**: Time Based Scheduling Enable feature is selected
TBS_CH	31:28	r	**The number of DMA channels enabled for TBS (starting from the highest Tx Channel, in descending order)** This field provides the number of DMA channels enabled for TBS (starting from the highest Tx Channel, in descending order): - 0000: 1 DMA Tx Channel enabled for TBS - 0001: 2 DMA Tx Channels enabled for TBS - 0010: 3 DMA Tx Channels enabled for TBS - ... - 1111: 16 DMA Tx Channels enabled for TBS Based on the configuration, one or more bits of this field must be treated as Reserved/Read-Only.

14.6.85 Port j MAC HW Feature4

This register indicates the presence of the fifth set of optional features or functions of DWC_xgmac. The software driver can use this register to dynamically enable or disable the programs related to the optional blocks.

Portj_MAC_HW_Feature4 (j=0-1) Offset address: $1012C_H+j*2000_H$

Port j MAC HW Feature4 Kernel Reset value: $0000\ 0001_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0							DSTS	0			EASP	0		PCSEL	
r							r	r			r	r		r	

Field	Bits	Type	Description
PCSEL	1:0	r	**Policing Counters** This value indicates the number of policing counters implemented when PC is enabled 00_B **INACTIVE**: Policing counter feature is not selected 01_B **PCNUM_16**: 16 Policing counters are selected 10_B **PCNUM_32**: 32 Policing counters are selected **others,** Reserved
EASP	4	r	**Extended Automotive Safety Package** This field along with ASP field of MAC_HW_Feature3 register indicates the encoded value for the different Safety features
DSTS	8	r	**Additional Free Running Timestamping Source** This bit is set to 1 after reset when the Additional Free Running Timestamping Source feature is selected.
0	3:2, 7:5, 31:9	r	**Reserved** Read as 0; should be written with 0.

14.6.86 Port j MAC Extended Configuration

The MAC Extended Configuration register establishes the operating mode of the MAC transmitter.

Portj_MAC_Extended_Configuration (j=0-1) Offset address: $10140_H+j*2000_H$

Port j MAC Extended Configuration Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0			0		0		HD	ECRSFD	DO	DCRS	DR	BL		DC	0
r			r		r		rw	rw	rw	rw	rw	rw		rw	r

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0						TPRE	0	DDS	EIPG						
r						rw	r	rw	rw						

Field	Bits	Type	Description
EIPG	6:0	rw	**Extended Inter-Packet Gap** When IFP bit in MAC_Tx_configuration register is set, the minimum IPG is increased in steps of 32 bits, depending on the programming of this field and IPG field in MAC_Tx_Configuration register.EIPG, IPG: Minimum Inter-Packet Gap - 10'h000: 96 bit times - 10'h001: 128 bit times - 10'h002: 160 bit times - 10'h003: 192 bit times - 10'h004: 224 bit times - ... - 10'h3FF: 32832 bit times Note: The extended inter-packet gap (IPG) feature must be enabled only in full-duplex mode. There might be undesirable effects on back-pressure function and frame transmission if it is enabled in half-duplex mode.
DDS	7	rw	**DA Duplication Select** - 1: The received packet with Multicast/Broadcast (and Unicast when UDC bit of MAC_RxQ_Ctrl4 register is set) Destination address is routed to multiple Receive DMA Channels. The Receive DMA Channels is identified by XDCS field of MAC_DChSel_IndReg(#i) register corresponding to the MAC Address register that matches the Multicast/ Broadcast (and Unciast when UDC bit of MAC_RxQ_Ctrl4 register is set) DA. The XDCS field is interpreted to be a one-hot value, each bit corresponding to the Receive DMA Channel. - 0: The received packet is routed to a single Receive DMA Channel. The Receive DMA Channel is identified by the DCS field of MAC_Address_High register corresponding to the MAC Address register that matches the Destination address in the received packet. The DCS field is interpreted as a binary value. Values: - 0x0: DCS field in MAC_Address_High register is interpreted as binary value - 0x1: XDCS field in MAC_DChSel_IndReg register represent the target DMA Channels using a one-hot like encoding.
TPRE	9	rw	**Truncated Preamble Enable** When set, MAC supports processing of packets with 7 bytes preamble on RX side such that SFD is seen on lane2. On reset, packet with truncated preamble are dropped. This field has an impact only if the speed select is set to 3'b110 (XGMII 2.5G mode) TPRE = 1 For express packets, the supported format is - /S/ + Preamble (6/5)+ SFD (lane3/lane2) For preemptive packets, truncated preamble is not supported. Therefore, the supported format is - /S/ + Preamble (6)+ SFD (lane3)

(table continues...)

(continued)

Field	Bits	Type	Description
DC	17	rw	**Deferral Check** When this bit is set, the deferral check function is enabled in the MAC. The MAC issues a Packet Abort status, along with the excessive deferral error bit set in the Tx packet status, when the Tx state machine is deferred for more than 24,288 bit times in 10 or 100 Mbps mode. Deferral begins when the transmitter is ready to transmit, but it is prevented because of an active carrier sense signal (CRS) on MII. The defer time is not cumulative. For example, if the transmitter defers for 10,000 bit times, because the CRS signal is active and the CRS signal becomes inactive, the transmitter transmits and collision occurs. Because of collision, the transmitter needs to back off and then defer again after back off is complete. In such a scenario, the deferral timer is reset to 0, and it is restarted. When this bit is reset, the deferral check function is disabled and the MAC defers until the CRS signal goes inactive. This bit is applicable only in the half-duplex mode. 0_B **DISABLE**: Deferral check function is disabled 1_B **ENABLE**: Deferral check function is enabled
BL	19:18	rw	**Back-Off Limit** The back-off limit determines the random integer number (r) of slot time delays (512 bit times for 10/100 Mbps) for which the MAC waits before rescheduling a transmission attempt during retries after a collision. n = retransmission attempt. The random integer r takes the value in the range 0 to 2^k This bit is applicable only in the half-duplex mode. 00_B **MIN_N_10**: k = min(n,10) 01_B **MIN_N_8**: k = min(n,8) 10_B **MIN_N_4**: k = min(n,4) 11_B **MIN_N_1**: k = min(n,1)
DR	20	rw	**Disable Retry** When this bit is set, the MAC attempts only one transmission. When a collision occurs on the GMII or MII interface, the MAC ignores the current packet transmission and reports a Packet Abort with excessive collision error in the Tx packet status. When this bit is reset, the MAC retries based on the settings of the BL field. This bit is applicable only in the half-duplex mode. 0_B **ENABLE**: Enable Retry 1_B **DISABLE**: Disable Retry

(table continues...)

(continued)

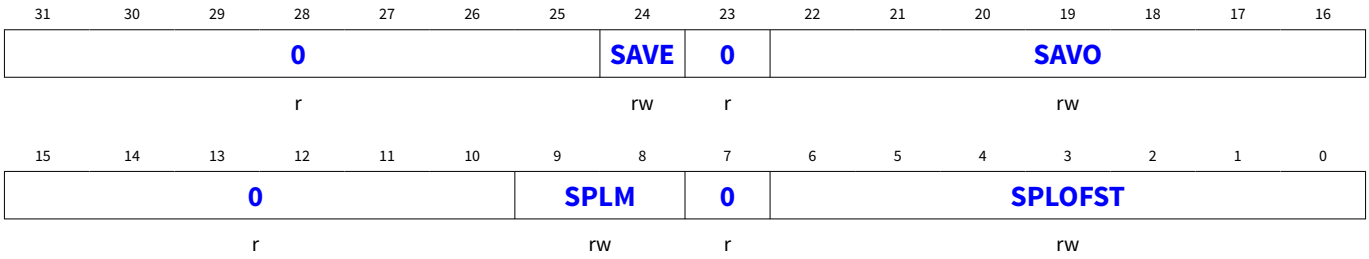
Field	Bits	Type	Description
DCRS	21	rw	**Disable Carrier Sense During Transmission** When this bit is set, the MAC transmitter ignores the MII CRS signal during packet transmission in the half-duplex mode. As a result, no errors are generated because of Loss of Carrier or No Carrier during transmission. When this bit is reset, the MAC transmitter generates errors because of Carrier Sense. The MAC can even abort the transmission. 0_B **ENABLE**: Enable Carrier Sense During Transmission 1_B **DISABLE**: Disable Carrier Sense During Transmission
DO	22	rw	**Disable Receive Own** When this bit is set, the MAC disables the reception of packets when the GMII/MII Transmit Data Enable signal is asserted in the half-duplex mode. When this bit is reset, the MAC receives all packets given by the PHY. This bit is not applicable in the full-duplex mode. 0_B **ENABLE**: Enable Receive Own 1_B **DISABLE**: Disable Receive Own
ECRSFD	23	rw	**Enable Carrier Sense before Transmission in Full-Duplex mode** When this bit is set, the MAC transmitter checks the CRS signal before packet transmission in the full-duplex mode. The MAC starts the transmission only when the CRS signal is low. When this bit is reset, the MAC transmitter ignores the status of the CRS signal. 0_B **DISABLE**: ECRSFD is disabled 1_B **ENABLE**: ECRSFD is enabled
HD	24	rw	**Half-Duplex Mode** When this bit is set, the MAC operates in the half-duplex mode in which it can transmit and receive simultaneously. This bit is RO with default value of 1'b0 in the full-duplex-only configurations. This bit is enabled only for 10/100Mbps. 0_B **FDUPLX**: Full-duplex mode 1_B **HDUPLX**: Half-duplex mode
0	8, 15:10, 16, 27:25, 28, 31:29	r	**Reserved** Read as 0; should be written with 0.

14.6.87 Port j MAC Ext Cfg1

This register contains Split mode control field and offset field for Header-Payload Split feature.

Portj_MAC_Ext_Cfg1 (j=0-1) Offset address: $10144_H+j*2000_H$

Port j MAC Ext Cfg1 Kernel Reset value: $0002\ 0002_H$



Field	Bits	Type	Description
SPLOFST	6:0	rw	**Split Offset** These bits indicate the value of offset from the beginning of Length/ Type field at which header split must take place when the appropriate SPLM is selected. The reset value of this field is 2 bytes indicating a split at L2 header. Value is in terms of bytes.
SPLM	9:8	rw	**Split Mode** These bits control the mode of splitting the incoming Rx packets. They are 00_B **L3L4**: Split at L3/L4 header 01_B **L2OFST**: Split at L2 header with an offset. Always Split at SPLOFST bytes from the beginning of Length/Type field of the Frame 10_B **COMBN**: Combination mode: Split similar to SPLM=00 for IP packets that are untagged or tagged and VLAN stripped and Split similar to SPLM=01 for non-IP packets 11_B **RSVD**: Reserved
SAVO	22:16	rw	**Split AV Offset** When SAVE bit is set to 1, and the received packet is an AV Type packet, these bits indicate the value of the offset from the beginning of Length/ Type field at which header must be split, when appropriate SPLM is selected. The reset value of this field is 2 bytes, indicating a split at L2 header. Value is in terms of bytes.
SAVE	24	rw	**Split AV Enable - When this bit is set to 1, and the received packet is an AV Type packet, the header is split at SAVO bytes from the beginning of Length/Type field of the packet, for L2 Split** - When this bit is set to 0, header is split at SPLOFST bytes from the beginning of Length/Type field of the frame, for L2 Split.
0	7, 15:10, 23, 31:25	r	**Reserved** Read as 0; should be written with 0.

14.6.88 Port j MAC DPP FSM Interrupt Status

This register contains the status of Automotive Safety related Data Path Parity Errors, Interface Timeout Errors, FSM State Parity Errors, and FSM State Timeout Errors.

Portj_MAC_DPP_FSM_Interrupt_Status (j=0-1) Offset address: $10150_H+j*2000_H$

Port j MAC DPP FSM Interrupt Status Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CPI	0					FSMPES	ARITES	ATITES	SLVTES	MSTTES	0	0	0	PTES	ATES
rw			r			rw	rw	rw	rw	rw	r	r	r	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	RTES	TTES	ASRPES	CWPES	MRRCPES	MRWCPES	ARPES	MTFCPES	MTBUPES	0	MTPES	0	0	DPES	0
r	rw	rw	rw	rw	rw	rw	rw	rw	rw	r	rw	r	r	rw	r

Field	Bits	Type	Description
DPES	1	rw	**Descriptor Cache write data path Parity checker Error Status** When this bit is set, it indicates that a parity error is detected at the Descriptor Cache parity checker (or at PC-desc). Access restriction applies. Clears on read (or write of 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event.
MTPES	4	rw	**MTL data path Parity checker Error Status** When this bit is set, it indicates that a parity error is detected at the MTL transmit write controller parity checker (or at PC2). Access restriction applies. Clears on read (or write of 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event. 0_B **INACTIVE**: MTL data path Parity checker Error Status not detected 1_B **ACTIVE**: MTL data path Parity checker Error Status detected
MTBUPES	6	rw	**MAC TBU data path Parity checker Error Status** When this bit is set, it indicates that a parity error is detected on the MAC TBU (or at PC4). Access restriction applies. Clears on read (or write of 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event. 0_B **INACTIVE**: MAC TBU data path Parity Error Status not detected 1_B **ACTIVE**: MAC TBU data path Parity Error Status detected
MTFCPES	7	rw	**MAC TFC data path Parity checker Error Status** When this bit is set, it indicates that a parity error is detected on the MAC TFC (or at PC5). Access restriction applies. Clears on read (or write of 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event. 0_B **INACTIVE**: MAC TFC data path Parity checker Error Status not detected 1_B **ACTIVE**: MAC TFC data path Parity checker Error Status detected

(table continues...)

(continued)

Field	Bits	Type	Description
ARPES	8	rw	**Application Receive interface data path Parity Error Status** When this bit is set, it indicates that a parity error is detected at the parity checker (PC-a) at AXI master interface. Access restriction applies. Clears on read (or write of 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event. 0_B **INACTIVE**: Application Receive interface data path Parity checker Error Status not detected 1_B **ACTIVE**: Application Receive interface data path Parity checker Error Status detected
MRWCPES	9	rw	**MTL RWC data path Parity checker Error Status** When this bit is set, it indicates that a parity error is detected at the MTL RWC input data parity checker (or at PC6). Access restriction applies. Clears on read (or write of 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event. 0_B **INACTIVE**: MTL RWC data path Parity checker Error Status not detected 1_B **ACTIVE**: MTL RWC data path Parity checker Error Status detected
MRRCPES	10	rw	**MTL RRC data path Parity checker Error Status** When this bit is set, it indicates that a parity error is detected at the MTL RRC output data parity checker (or at PC7). Access restriction applies. Clears on read (or write of 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event. 0_B **INACTIVE**: MTL RRC data path Parity checker Error Status not detected 1_B **ACTIVE**: MTL RRC data path Parity checker Error Status detected
CWPES	11	rw	**CSR Write data path Parity checker Error Status** When this bit is set, it indicates that parity error is detected at the CSR write data interface This bit is set to 1 when EPSI bit of MTL_DPP_Control register is set and a parity mismatch is detected on the input slave parity ports (or at PC-axi/apb/mci slave checker). Access restriction applies. Clears on read (or write of 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event. 0_B **INACTIVE**: CSR Write data path Parity checker Error Status not detected 1_B **ACTIVE**: CSR Write data path Parity checker Error Status detected
ASRPES	12	rw	**AXI Slave Read data path Parity checker Error Status** When this bit is set, it indicates that parity error is detected at the AXI Slave read data interface . Access restriction applies. Clears on read (or write of 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event. 0_B **INACTIVE**: AXI Slave Read data path Parity checker Error Status not detected 1_B **ACTIVE**: AXI Slave Read data path Parity checker Error Status detected

(table continues...)

(continued)

Field	Bits	Type	Description
TTES	13	rw	**Tx FSM Timeout Error Status** When this bit is set, it indicates that one of the Tx FSM timeout has occurred. Access restriction applies. Clears on read (or write of 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event. 0_B **INACTIVE**: Tx FSM timeout has not been detected 1_B **ACTIVE**: Tx FSM timeout has been detected
RTES	14	rw	**Rx FSM Timeout Error Status** When this bit is set, it indicates that one of the Rx FSM timeout has occurred. Access restriction applies. Clears on read (or write of 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event. 0_B **INACTIVE**: Rx FSM timeout has not been detected 1_B **ACTIVE**: Rx FSM timeout has been detected
ATES	16	rw	**APP FSM Timeout Error Status** When this bit is set, it indicates that one of the APP FSM timeout has occurred. Access restriction applies. Clears on read (or write of 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event. 0_B **INACTIVE**: APP FSM Timeout Error Status not detected 1_B **ACTIVE**: APP FSM Timeout Error Status detected
PTES	17	rw	**PTP FSM Timeout Error Status** When this bit is set, it indicates that one of the PTP FSM timeout has occurred. Access restriction applies. Clears on read (or write of 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event. 0_B **INACTIVE**: PTP FSM Timeout Error Status not detected 1_B **ACTIVE**: PTP FSM Timeout Error Status detected
MSTTES	21	rw	**Master Read/Write Timeout Error Status** When this bit is set, it indicates that an Application/CSR timeout has occurred on the master (AXI) interface. Access restriction applies. Clears on read (or write of 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event. 0_B **INACTIVE**: Master Read/Write Timeout Error Status not detected 1_B **ACTIVE**: Master Read/Write Timeout Error Status detected
SLVTES	22	rw	**Slave Read/Write Timeout Error Status** When this bit is set, it indicates that an Application/CSR timeout has occurred on the AXI slave interface. Access restriction applies. Clears on read (or write of 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event. 0_B **INACTIVE**: Slave Read/Write Timeout Error Status not detected 1_B **ACTIVE**: Slave Read/Write Timeout Error Status detected

(table continues...)

(continued)

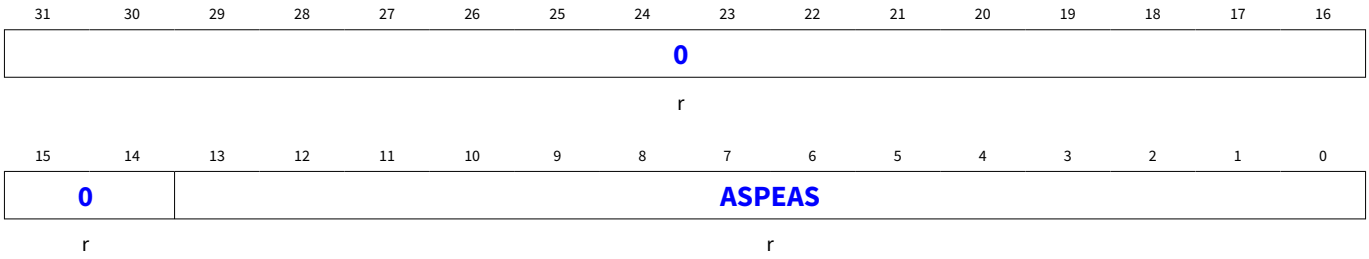
Field	Bits	Type	Description
ATITES	23	rw	**ATI Interface Timeout Error Status** When this bit is set, it indicates that an Application timeout has occurred on ATI Interface. Access restriction applies. Clears on read (or write of 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event. 0_B **INACTIVE**: Timeout not detected on ATI Interface 1_B **ACTIVE**: Timeout detected on ATI Interface
ARITES	24	rw	**ARI Interface Timeout Error Status** When this bit is set, it indicates that an Application timeout has occurred on ARI Interface. Access restriction applies. Clears on read (or write of 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event. 0_B **INACTIVE**: Timeout not detected on ARI Interface 1_B **ACTIVE**: Timeout detected on ARI Interface
FSMPES	25	rw	**FSM State Parity Error Status** Indicates that parity error is detected in one of the FSM State registers. Access restriction applies. Clears on read (or write of 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event. 0_B **INACTIVE**: Slave Read/Write Timeout Error Status not detected 1_B **ACTIVE**: Slave Read/Write Timeout Error Status detected
CPI	31	rw	**Control Register Parity Check Error** When this bit is set, it indicates a parity check error for CSR registers. Access restriction applies. Clears on read (or write of 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event.
0	0, 2, 3, 5, 15, 18, 19, 20, 30:26	r	**Reserved** Read as 0; should be written with 0.

14.6.89 Port j MAC AXI SLV DPE Addr Status

This register gives the address of the Register Write transaction in which parity error occurred.

Portj_MAC_AXI_SLV_DPE_Addr_Status (j=0-1) Offset address: $10154_H+j*2000_H$

Port j MAC AXI SLV DPE Addr Status Kernel Reset value: $0000\ 0000_H$



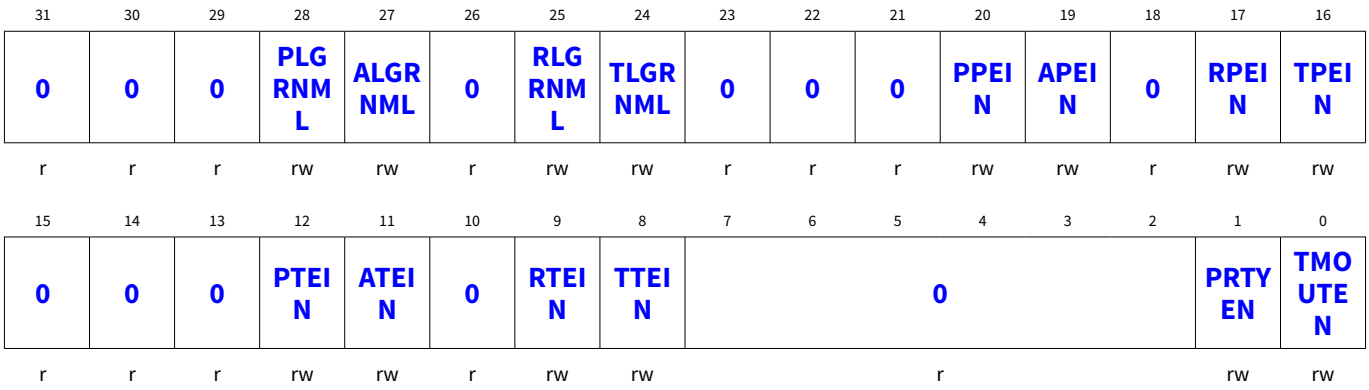
Field	Bits	Type	Description
ASPEAS	13:0	r	**AXI Slave data path Parity Error Address Status** This field holds the CSR address for which parity error is detected on the CSR write data. This field holds the first address for which parity error is detected on the write data which is cleared on read.
0	31:14	r	**Reserved** Read as 0; should be written with 0.

14.6.90 Port j MAC FSM Control

This register is used to control the FSM State parity and timeout error injection in Debug mode.

Portj_MAC_FSM_Control (j=0-1) Offset address: $10158_H+j*2000_H$

Port j MAC FSM Control Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
TMOUTEN	0	rw	**FSM Timeout Checking Enable** When this bit is set, FSM timeout feature is enabled. 0_B **DISABLE**: FSM timeout feature is disabled 1_B **ENABLE**: FSM timeout feature is enabled
PRTYEN	1	rw	**FSM Parity Checking Enable** When this bit is set, FSM parity feature is enabled. 0_B **DISABLE**: FSM Parity feature is disabled 1_B **ENABLE**: FSM Parity feature is enabled

(table continues...)

(continued)

Field	Bits	Type	Description
TTEIN	8	rw	**Tx FSM Timeout Error Injection** When this bit is set, Error Injection for Tx FSM timeout is enabled. 0_B **DISABLE**: Tx FSM Timeout Error Injection is disabled 1_B **ENABLE**: Tx FSM Timeout Error Injection is enabled
RTEIN	9	rw	**Rx FSM Timeout Error Injection** When this bit is set, Error Injection for Rx FSM timeout is enabled. 0_B **DISABLE**: Rx FSM Timeout Error Injection is disabled 1_B **ENABLE**: Rx FSM Timeout Error Injection is enabled
ATEIN	11	rw	**APP FSM Timeout Error Injection** When this bit is set, Error Injection for APP FSM timeout is enabled. 0_B **DISABLE**: APP FSM Timeout Error Injection is disabled 1_B **ENABLE**: APP FSM Timeout Error Injection is enabled
PTEIN	12	rw	**PTP FSM Timeout Error Injection** When this bit is set, Error Injection for PTP FSM timeout is enabled. 0_B **DISABLE**: PTP FSM Timeout Error Injection is disabled 1_B **ENABLE**: PTP FSM Timeout Error Injection is enabled
TPEIN	16	rw	**Tx FSM Parity Error Injection** When this bit is set, Error Injection for Tx FSM Parity is enabled. 0_B **DISABLE**: Tx FSM Parity Error Injection is disabled 1_B **ENABLE**: Tx FSM Parity Error Injection is enabled
RPEIN	17	rw	**Rx FSM Parity Error Injection** When this bit is set, Error Injection for Rx FSM Parity is enabled. 0_B **DISABLE**: Rx FSM Parity Error Injection is disabled 1_B **ENABLE**: Rx FSM Parity Error Injection is enabled
APEIN	19	rw	**APP FSM Parity Error Injection** When this bit is set, Error Injection for APP FSM Parity is enabled. 0_B **DISABLE**: APP FSM Parity Error Injection is disabled 1_B **ENABLE**: APP FSM Parity Error Injection is enabled
PPEIN	20	rw	**PTP FSM Parity Error Injection** When this bit is set, Error Injection for PTP FSM Parity is enabled. 0_B **DISABLE**: PTP FSM Parity Error Injection is disabled 1_B **ENABLE**: PTP FSM Parity Error Injection is enabled
TLGRNML	24	rw	**Tx Large/Normal Mode Select** When this bit is set, large mode tic generation is used for Tx domain; otherwise, normal mode tic generation is used. 0_B **DISABLE**: normal mode tic generation is used for Tx domain 1_B **ENABLE**: large mode tic generation is used for Tx domain

(table continues...)

(continued)

Field	Bits	Type	Description
RLGRNML	25	rw	**Rx Large/Normal Mode Select** When this bit is set, large mode tic generation is used for Rx domain; otherwise, normal mode tic generation is used. 0_B **DISABLE**: normal mode tic generation is used for Rx domain 1_B **ENABLE**: large mode tic generation is used for Rx domain
ALGRNML	27	rw	**APP Large/Normal Mode Select** When this bit is set, large mode tic generation is used for APP domain; otherwise, normal mode tic generation is used. 0_B **DISABLE**: normal mode tic generation is used for APP domain 1_B **ENABLE**: large mode tic generation is used for APP domain
PLGRNML	28	rw	**PTP Large/Normal Mode Select** When this bit is set, large mode tic generation is used for PTP domain; otherwise, normal mode tic generation is used. 0_B **DISABLE**: normal mode tic generation is used for PTP domain 1_B **ENABLE**: large mode tic generation is used for PTP domain
0	7:2, 10, 13, 14, 15, 18, 21, 22, 23, 26, 29, 30, 31	r	**Reserved** Read as 0; should be written with 0.

14.6.91 Port j MAC FSM ACT Timer

This register is used to select the FSM and Interface Timeout values.

Portj_MAC_FSM_ACT_Timer (j=0-1) Offset address: $1015C_H+j*2000_H$

Port j MAC FSM ACT Timer Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	CTMR			0				LTMRMD				NTMRMD			
r	rw			r				rw				rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0						TMR									
r						rw									

Field	Bits	Type	Description
TMR	9:0	rw	**CSR Clocks for 1us Tic** This field indicates the number of CSR clocks required to generate 1us tic.
NTMRMD	19:16	rw	**Normal Mode Timeout Value** This field provides the value to be used for normal mode FSM and other interface time outs. The timeout duration based on the mode value is as follows: 0_H **DISABLE**: Timer disabled 1_H **M_1MICRO_SEC**: 1us 2_H **M_4MILLI_SEC**: 1.024ms (~4ms) 3_H **M_16MILLI_SEC**: 16.384ms (~16ms) 4_H **M_64MILLI_SEC**: 65.536ms (~64ms) 5_H **M_256MILLI_SEC**: 262.144ms (~256ms) 6_H **M_1SEC**: 1.048sec (~1sec) 7_H **M_4SEC**: 4.194sec (~4sec) 8_H **M_16SEC**: 16.777sec (~16sec) 9_H **M_32SEC**: 33.554sec (~32sec) A_H **M_64SEC**: 67.108sec (~64sec) B_H **RSVD**: Reserved **others,** Reserved
LTMRMD	23:20	rw	**Large Mode Timeout Value** This field provides the mode value to be used for large mode FSM and other interface time outs. The timeout duration based on the mode value is as follows: 0_H **DISABLE**: Timer disabled 1_H **M_1MICRO_SEC**: 1us 2_H **M_4MILLI_SEC**: 1.024ms (~4ms) 3_H **M_16MILLI_SEC**: 16.384ms (~16ms) 4_H **M_64MILLI_SEC**: 65.536ms (~64ms) 5_H **M_256MILLI_SEC**: 262.144ms (~256ms) 6_H **M_1SEC**: 1.048sec (~1sec) 7_H **M_4SEC**: 4.194sec (~4sec) 8_H **M_16SEC**: 16.777sec (~16sec) 9_H **M_32SEC**: 33.554sec (~32sec) A_H **M_64SEC**: 67.108sec (~64sec) B_H **RSVD**: Reserved **others,** Reserved

(table continues...)

(continued)

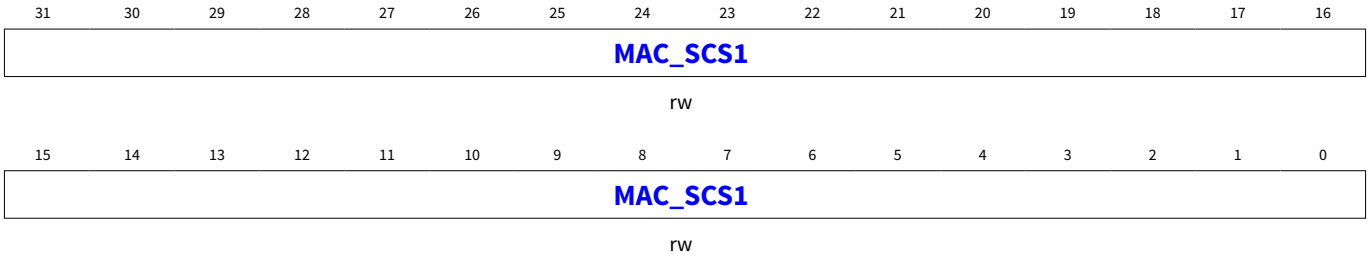
Field	Bits	Type	Description
CTMR	30:28	rw	**CSR Parity Or Rx Parser Instruction Table ECC Check Interval Value** This register is used for setting timer value for generating timer tic for CSR parity or Rx Parser Instruction Table ECC check. 000_B **M_1MILLI_SEC**: 1ms 001_B **M_1MICRO_SEC_CSR**: 1us 010_B **M_4MILLI_SEC_CSR**: 4.096ms (~4ms) 011_B **M_16MILLI_SEC_CSR**: 16.384ms (~16ms) 100_B **M_64MILLI_SEC_CSR**: 65.536ms (~64ms) 101_B **M_256MILLI_SEC_CSR**: 262.144ms (~256ms) 110_B **M_1SEC_CSR**: 1.048sec (~1sec) 111_B **ALWAYS_EN**: Always Enabled
0	15:10, 27:24, 31	r	**Reserved** Read as 0; should be written with 0.

14.6.92 Port j SNPS SCS REG1

Synopsys Reserved Register.

Portj_SNPS_SCS_REG1 (j=0-1) Offset address: $10160_H+j*2000_H$

Port j SNPS SCS REG1 Kernel Reset value: $0000\ 0000_H$



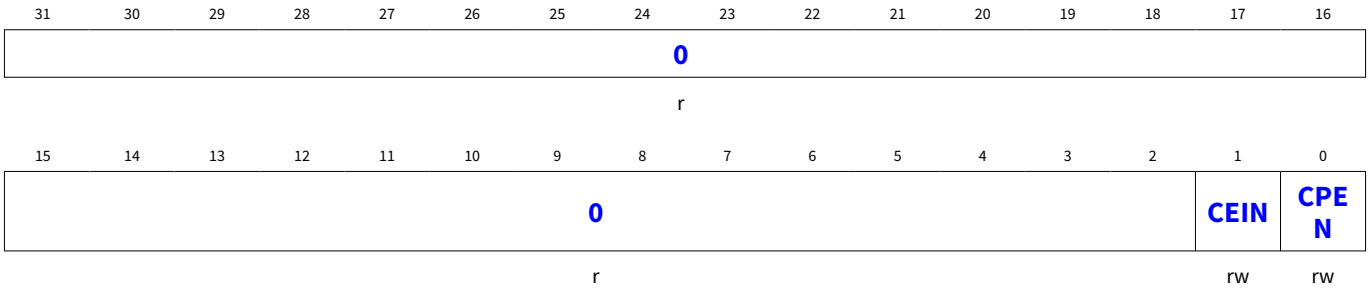
Field	Bits	Type	Description
MAC_SCS1	31:0	rw	**Synopsys Reserved** All the bits must be set to 0. This field is reserved for Synopsys Internal use, and must always be set to 0 unless instructed by Synopsys. Setting any bit to 1 might cause unexpected behavior in the IP.

14.6.93 Port j MAC SCSR Control

This register is used to control safety protected CSR parity generation.

Portj_MAC_SCSR_Control (j=0-1) Offset address: $10164_H+j*2000_H$

Port j MAC SCSR Control Kernel Reset value: $0000\ 0000_H$



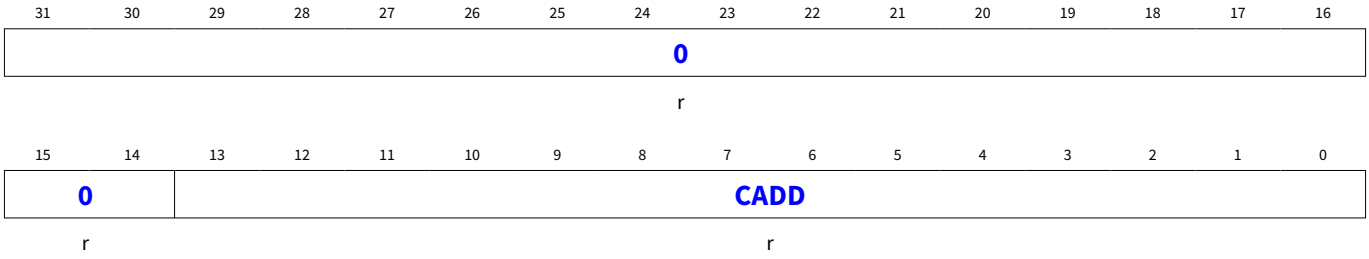
Field	Bits	Type	Description
CPEN	0	rw	**CSR Parity Enable** When this bit is set, CSR parity checking is enabled.
CEIN	1	rw	**CSR Error Injection** When this bit is set, CSR error injection is enabled. Error is injected in the subsequent write operation where at the least one of the byte enable matches with byte enable indicated in CEBE. After error injection, this bit is cleared by the design.
0	31:2	r	**Reserved** Read as 0; should be written with 0.

14.6.94 Port j MAC SCSR Parity Status

This register indicates the offset address of the register where parity_error occurred.

Portj_MAC_SCSR_Parity_Status (j=0-1) Offset address: $10168_H+j*2000_H$

Port j MAC SCSR Parity Status Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
CADD	13:0	r	**CSR Address with error** This field indicates the offset address of last CSR register for which read data parity mismatched with reference parity. For example, when parity mismatch occurred when MAC_Rx_Configuration register is read, this field will contain value of 16'h0004.
0	31:14	r	**Reserved** Read as 0; should be written with 0.

14.6.95 Port j MAC SCSR Parity Err Count

This register indicates error count associated with parity check.

Portj_MAC_SCSR_Parity_Err_Count (j=0-1) Offset address: $1016C_H+j*2000_H$

Port j MAC SCSR Parity Err Count Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0						ECLL									
r						r									

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0						ECCL									
r						r									

Field	Bits	Type	Description
ECCL	9:0	r	**Count of parity errors in current safety loop** This field indicates the Count of parity errors in the current loop of all safety register parity check. Application read related errors are also included in the count. Value saturates at all "1"s and is cleared on read.
ECLL	25:16	r	**Count of parity errors in the previous safety loop** This field indicates the Count of parity errors in previous loop of all safety register parity check. Any application read related errors are also included in the count. Value saturates at all "1"s and is cleared on read.
0	15:10, 31:26	r	**Reserved** Read as 0; should be written with 0.

14.6.96 Port j MAC Flex Hdr Cfg

This register contains flexible header enable, length and start position for Flexible Header Feature.

Portj_MAC_Flex_Hdr_Cfg (j=0-1) Offset address: $10194_H+j*2000_H$

Port j MAC Flex Hdr Cfg Kernel Reset value: $0300\ 0C00_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0					FHL			0							
r					rw			r							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			FHSP					0					EFLL	FHTX	FHRX
r			rw					r					rw	rw	rw

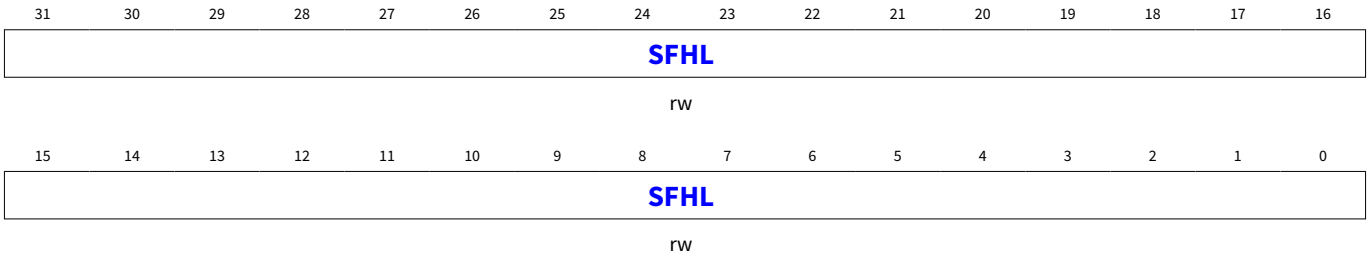
Field	Bits	Type	Description
FHRX	0	rw	**Flexible Header Enable for Rx path** When this is set, all the incoming packets from the Line is expected to have FHL bytes Flexible Header in the Rx path. When set to 0, normal/standard Ethernet packets are expected to be received and transmitted. This bit programming must be not changed when there are packets present in the RX path.
FHTX	1	rw	**Flexible Header Enable for Tx path** When this is set, all the packets from the Software is expected to have FHL bytes Flexible Header in the Tx path. When set to 0, normal/standard Ethernet packets are expected to be received and transmitted. This bit programming must be not changed when there are packets present in the TX path.
EFLL	2	rw	**Insert Flexible Header for Pause frames - When this bit is set along with FHTX, static flexible header of the programmed length is inserted in the MAC-generated pause frames at the programmed position - When this bit is set along with FHRX, in Rx path, pause frame must have FH programmed in the SFH (static flexible header) field** - When this bit is reset, pause frames do not have flexible header inserted in them, in both Tx and Rx paths.
FHSP	12:8	rw	**Flexible Header Start Position** When FHRX/FHTX bit is set, FH is present in Tx/Rx packets at one of the following encoded positions - 5'b00000: Before DA field (Byte 0) - 5'b00001 - 5'b01011: RSVD - 5'b01100: After SA field (Byte 12) - 5'b01101: Reserved - 5'b01110 - 5'11101: Byte 14 to Byte 29
FHL	26:24	rw	**Flexible Header Length** When FHRX/FHTX bit is set, all the packets coming on the line in the Rx path and all the packets transmitted from the application on the Tx path must have Flexible Header of the following size, in bytes: - 3'b000 - 3'b111: 1 Byte to 8 Bytes
0	7:3, 23:13, 31:27	r	**Reserved** Read as 0; should be written with 0.

14.6.97 Port j MAC Flex Hdr Low

This register contains programmable Lower Static Flexible Header for Mac-gen pause/hardware offload engines generated packets (such as PTP) Flexible Header Feature.

Portj_MAC_Flex_Hdr_Low (j=0-1) Offset address: $10198_H+j*2000_H$

Port j MAC Flex Hdr Low Kernel Reset value: $0000\ 0000_H$



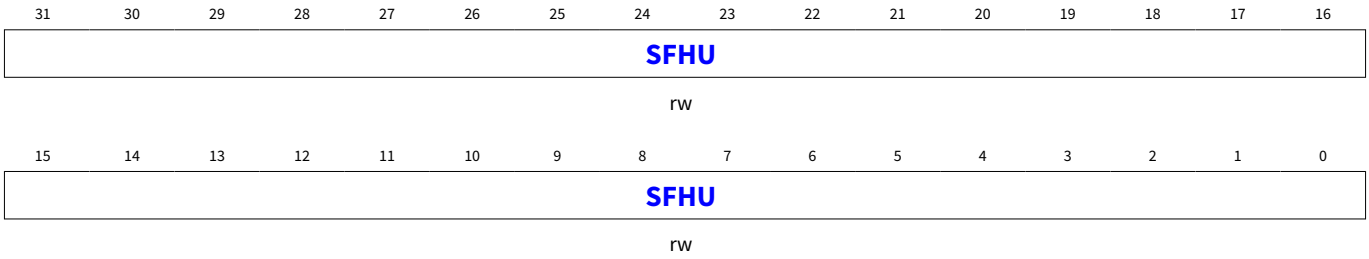
Field	Bits	Type	Description
SFHL	31:0	rw	**Static Flexible Header - Lower** When Flexible header feature is enabled in the Tx path, all the MAC generated Tx packets such as PTP packets and Link Layer packets (such as pause and verify-response if EFLL is set) have FH inserted from this programmable field, at the programmed FH position. This field must be used for FH length of upto 4 bytes

14.6.98 Port j MAC Flex Hdr High

This register contains programmable Upper Static Flexible Header for Mac-gen (pause)/hardware offload engines generated packets(such as PTP) Flexible Header Feature.

Portj_MAC_Flex_Hdr_High (j=0-1) Offset address: $1019C_H+j*2000_H$

Port j MAC Flex Hdr High Kernel Reset value: $0000\ 0000_H$



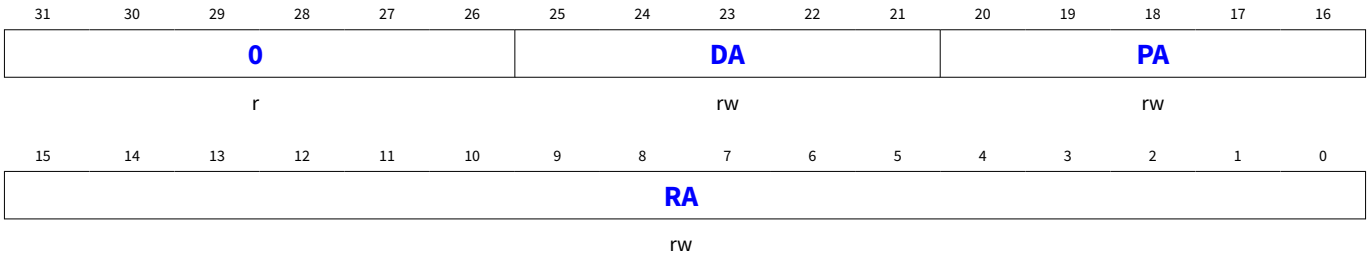
Field	Bits	Type	Description
SFHU	31:0	rw	**Static Flexible Header - Upper** This field is used for programming FH's upper 4 bytes when FH length is from 5 bytes to 8 bytes for Link Layer/ MAC auto-generated packets

14.6.99 Port j MDIO Single Command Address

This register contains the addresses of the ports, devices, and registers accessed by the SMA during single read or write transfers.

Portj_MDIO_Single_Command_Address (j=0-1) Offset address: $10200_H+j*2000_H$

Port j MDIO Single Command Address Kernel Reset value: $0000\ 0000_H$



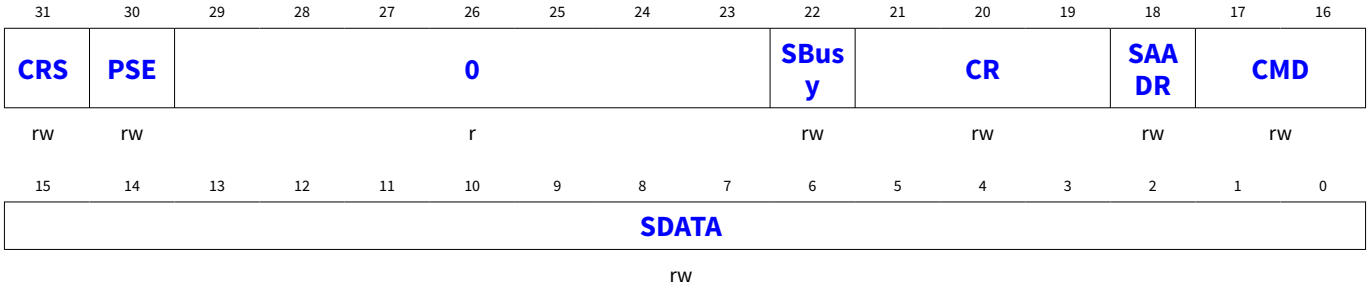
Field	Bits	Type	Description
RA	15:0	rw	**Register Address** Register Address for single read/write. When using Clause 22 frames, only the lower 5 bits of the field are used.
PA	20:16	rw	**Port Address** Port Address used for indirect single read/write access.
DA	25:21	rw	**Device Address** Device Address for indirect addressing of device for single read/write.
0	31:26	r	**Reserved** Read as 0; should be written with 0.

14.6.100 Port j MDIO Single Command Control Data

This register controls single read/write operations, and contains the data bits read on a single-read operation and the data to be written during single-write operations.

Portj_MDIO_Single_Command_Control_Data (j=0-1) Offset address: $10204_H+j*2000_H$

Port j MDIO Single Command Control Data Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
SDATA	15:0	rw	**Single Write Data** For single writes, this field contains data to be written. For single reads, this field contains data that has been read.
CMD	17:16	rw	**Control Command** SMA commands are written to these bits. The command codes are: - 2'b11: SINGLE_READ - 2'b01: SINGLE_WRITE - 2'b10: POST INCREMENT READ OPERATION - 2'b00: Reserved (Results are undefined)

(table continues...)

(continued)

Field	Bits	Type	Description
SAADR	18	rw	**Skip Address Frame** The application sets this bit to prevent the SMA from sending address frames before read, write, or post-increment read frames.
CR	21:19	rw	**Application Clock Range** The Application Clock Range selection determines the relationship between the CSR and MDC clocks. Do not change the value of CR bits unless there is no operation (continuous write or scan) in progress on the SMA interface. When CRS bit is set to 0: - 3'b000 -- clk_csr_i: 100-150 MHz -- MDC clock: clk_csr_i/62 - 3'b001 -- clk_csr_i: 150-250 MHz -- MDC clock: clk_csr_i/102 - 3'b010 -- clk_csr_i: 250-300 MHz -- MDC clock: clk_csr_i/122 - 3'b011 -- clk_csr_i: 300-350 MHz -- MDC clock: clk_csr_i/142 - 3'b100 -- clk_csr_i: 350-400 MHz -- MDC clock: clk_csr_i/162 - 3'b101 -- clk_csr_i: 400-500 MHz -- MDC clock: clk_csr_i/202 - 3'b110 -- clk_csr_i: Reserved -- MDC clock: Reserved - 3'b111 -- clk_csr_i: Reserved -- MDC clock: Reserved When CRS bit is set to 1, MDIO clock is derived based on CSR clock as follows: - 3'b000: MDC clock: clk_csr_i/4 - 3'b001: MDC clock: clk_csr_i/6 - 3'b010: MDC clock: clk_csr_i/8 - 3'b011: MDC clock: clk_csr_i/10 - 3'b100: MDC clock: clk_csr_i/12 - 3'b101: MDC clock: clk_csr_i/14 - 3'b110: MDC clock: clk_csr_i/16 - 3'b111: MDC clock: clk_csr_i/18

(table continues...)

(continued)

Field	Bits	Type	Description
SBusy	22	rw	**Busy** When you set this bit, it triggers a single read, single write, or post-increment read operation. DWC_xgmac clears this bit when the triggered operation is completed. During the MDIO transfer cycle, this bit remains high to indicate the cycle in progress. When the Busy bit is high, do not modify any bits in SMA registers 1 or 2. When this bit is set high, bits[15:0] of MDIO_Single_Command_Control_Data must be kept valid for MDIO write access or bits[15:0] of MDIO_Single_Command_Control_Data are invalid (for MDIO read access).
PSE	30	rw	**Preamble Suppression Enable** When this bit is set, the SMA suppresses the 32-bit preamble and transmits MDIO packets with only 1 preamble bit. When this bit is 0, the MDIO packet always has 32 bits of preamble as defined in the IEEE specifications.
CRS	31	rw	**Clock Range Select** When this set overrides the default MDIO clock generation based on CSR clock. When this bit is set MDIO Clock generation using CSR clock such that MDIO frequency is equals to CSR clock divided by 4, 6, 8, 10, 12, 14, 16, 18. Therefore, this mode enables faster clock on MDIO.
0	29:23	r	**Reserved** Read as 0; should be written with 0.

14.6.101 Port j MDIO Continuous Write Address

You can program this register to start continuous write operation.

Portj_MDIO_Continuous_Write_Address (j=0-1) Offset address: $10208_H+j*2000_H$

Port j MDIO Continuous Write Address Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0									CBUSY	CPRT	CADDR				
r									rw	rw	rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CREGADDR															
rw															

Field	Bits	Type	Description
CREGADDR	15:0	rw	**Continuous Register Address** Register address to which continuous writes are performed. For Clause 22 ports, only the five least significant bits are valid.

(table continues...)

(continued)

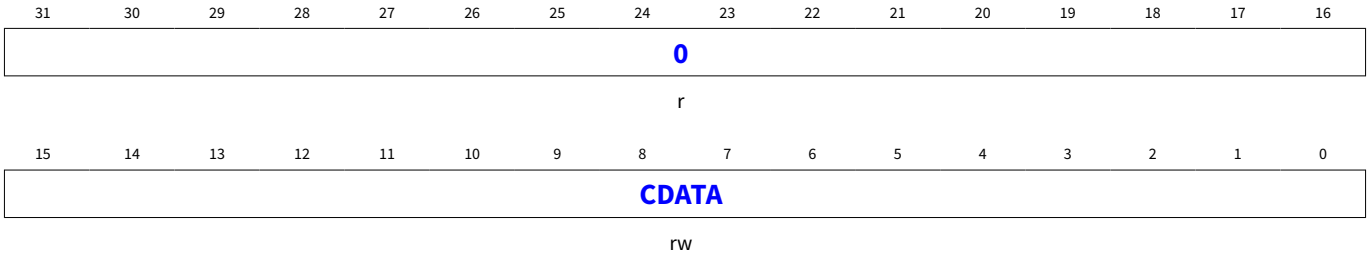
Field	Bits	Type	Description
CADDR	20:16	rw	**Continuous Port or Device Address** Common address to which continuous writes are applied. Used either as a device address or a port address, depending on the CPRT setting. As a port address, the valid range is 0-3 only.
CPRT	21	rw	**Continuous Port or Device Address Select** When this bit is set, the CADDR is used as the port address and continuous writes are performed on all port devices. When this bit is unset, CADDR is used as the device address and continuous writes are performed on all ports.
CBUSY	22	rw	**Continuous Write Busy** When this bit is set, continuous write operation starts. When continuous write operation is complete, this bit is reset. This bit must read as 0 before the next continuous write command is issued.
0	31:23	r	**Reserved** Read as 0; should be written with 0.

14.6.102 Port j MDIO Continuous Write Data

This register contains the data to be written during continuous write operation. Do not change the register's contents until the Continuous Write Busy bit (CBUSY) is cleared.

Portj_MDIO_Continuous_Write_Data (j=0-1) Offset address: $1020C_H+j*2000_H$

Port j MDIO Continuous Write Data Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
CDATA	15:0	rw	**Continuous Write Data** Data to be used in the MDIO frame during a continuous write. Do not modify the register's contents unless the Continuous Write Busy bit (CBUSY) is cleared.
0	31:16	r	**Reserved** Read as 0; should be written with 0.

14.6.103 Port j MDIO Continuous Scan Port Enable

This register controls the PHY ports and corresponding devices (enabled by the corresponding Device In Use register) to be accessed during the SMA's continuous scan operation. If you change any bit in this register during a continuous scan, it takes effect at the end of the current scan read cycle. In other words, if Bit 2 (Port 2) is enabled while Port 3, 0, or 1 is being scanned, port 2 scanning is scheduled at the end of the Port 1 scan. If any bit is cleared by software, it takes effect at the end of the reading the last register of last enabled device.

Portj_MDIO_Continuous_Scan_Port_Enable (j=0-1) Offset address: $10210_H+j*2000_H$

Port j MDIO Continuous Scan Port Enable Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PORT31SCE	PORT30SCE	PORT29SCE	PORT28SCE	PORT27SCE	PORT26SCE	PORT25SCE	PORT24SCE	PORT23SCE	PORT22SCE	PORT21SCE	PORT20SCE	PORT19SCE	PORT18SCE	PORT17SCE	PORT16SCE
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PORT15SCE	PORT14SCE	PORT13SCE	PORT12SCE	PORT11SCE	PORT10SCE	PORT9SCE	PORT8SCE	PORT7SCE	PORT6SCE	PORT5SCE	PORT4SCE	PORT3SCE	PORT2SCE	PORT1SCE	PORT0SCE
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
PORT0SCE	0	rw	**Port 0 Continuous Scan Enable** This bit indicates that Port 0 is enabled for continuous scan. Software can disable the port for continuous scan by setting this bit to 0.
PORT1SCE	1	rw	**Port 1 Continuous Scan Enable** This bit indicates that Port 1 is enabled for continuous scan. Software can disable the port for continuous scan by setting this bit to 0.
PORT2SCE	2	rw	**Port 2 Continuous Scan Enable** This bit indicates that Port 2 is enabled for continuous scan. Software can disable the port for continuous scan by setting this bit to 0.
PORT3SCE	3	rw	**Port 3 Continuous Scan Enable** This bit indicates that Port 3 is enabled for continuous scan. Software can disable the port for continuous scan by setting this bit to 0.
PORT4SCE	4	rw	**Port 4 Continuous Scan Enable** This bit indicates that Port 4 is enabled for continuous scan. Software can disable the port for continuous scan by setting this bit to 0.
PORT5SCE	5	rw	**Port 5 Continuous Scan Enable** This bit indicates that Port 5 is enabled for continuous scan. Software can disable the port for continuous scan by setting this bit to 0.
PORT6SCE	6	rw	**Port 6 Continuous Scan Enable** This bit indicates that Port 6 is enabled for continuous scan. Software can disable the port for continuous scan by setting this bit to 0.
PORT7SCE	7	rw	**Port 7 Continuous Scan Enable** This bit indicates that Port 7 is enabled for continuous scan. Software can disable the port for continuous scan by setting this bit to 0.

(table continues...)

(continued)

Field	Bits	Type	Description
PORT8SCE	8	rw	**Port 8 Continuous Scan Enable** This bit indicates that Port 8 is enabled for continuous scan. Software can disable the port for continuous scan by setting this bit to 0.
PORT9SCE	9	rw	**Port 9 Continuous Scan Enable** This bit indicates that Port 9 is enabled for continuous scan. Software can disable the port for continuous scan by setting this bit to 0.
PORT10SCE	10	rw	**Port 10 Continuous Scan Enable** This bit indicates that Port 10 is enabled for continuous scan. Software can disable the port for continuous scan by setting this bit to 0.
PORT11SCE	11	rw	**Port 11 Continuous Scan Enable** This bit indicates that Port 11 is enabled for continuous scan. Software can disable the port for continuous scan by setting this bit to 0.
PORT12SCE	12	rw	**Port 12 Continuous Scan Enable** This bit indicates that Port 12 is enabled for continuous scan. Software can disable the port for continuous scan by setting this bit to 0.
PORT13SCE	13	rw	**Port 13 Continuous Scan Enable** This bit indicates that Port 13 is enabled for continuous scan. Software can disable the port for continuous scan by setting this bit to 0.
PORT14SCE	14	rw	**Port 14 Continuous Scan Enable** This bit indicates that Port 14 is enabled for continuous scan. Software can disable the port for continuous scan by setting this bit to 0.
PORT15SCE	15	rw	**Port 15 Continuous Scan Enable** This bit indicates that Port 15 is enabled for continuous scan. Software can disable the port for continuous scan by setting this bit to 0.
PORT16SCE	16	rw	**Port 16 Continuous Scan Enable** This bit indicates that Port 16 is enabled for continuous scan. Software can disable the port for continuous scan by setting this bit to 0.
PORT17SCE	17	rw	**Port 17 Continuous Scan Enable** This bit indicates that Port 17 is enabled for continuous scan. Software can disable the port for continuous scan by setting this bit to 0.
PORT18SCE	18	rw	**Port 18 Continuous Scan Enable** This bit indicates that Port 18 is enabled for continuous scan. Software can disable the port for continuous scan by setting this bit to 0.
PORT19SCE	19	rw	**Port 19 Continuous Scan Enable** This bit indicates that Port 19 is enabled for continuous scan. Software can disable the port for continuous scan by setting this bit to 0.
PORT20SCE	20	rw	**Port 20 Continuous Scan Enable** This bit indicates that Port 20 is enabled for continuous scan. Software can disable the port for continuous scan by setting this bit to 0.
PORT21SCE	21	rw	**Port 21 Continuous Scan Enable** This bit indicates that Port 21 is enabled for continuous scan. Software can disable the port for continuous scan by setting this bit to 0.

(table continues...)

(continued)

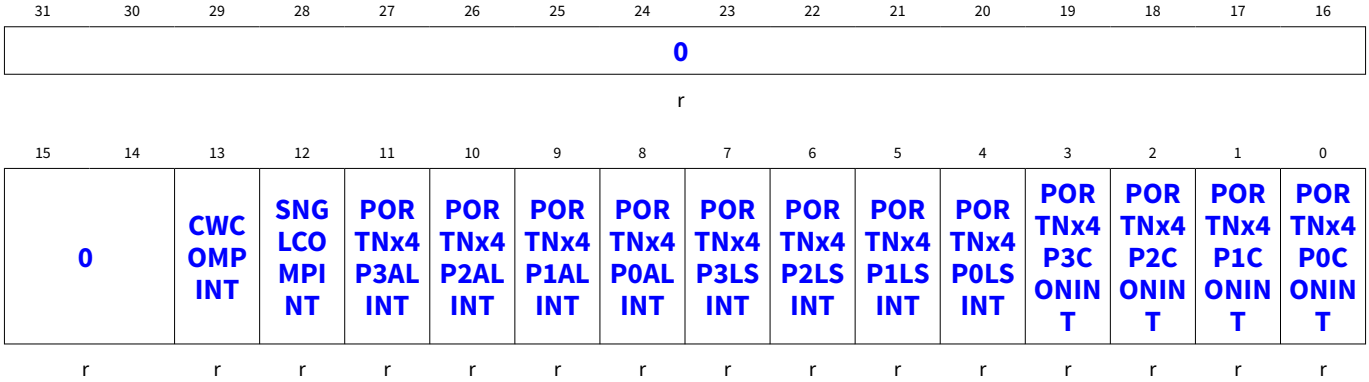
Field	Bits	Type	Description
PORT22SCE	22	rw	**Port 22 Continuous Scan Enable** This bit indicates that Port 22 is enabled for continuous scan. Software can disable the port for continuous scan by setting this bit to 0.
PORT23SCE	23	rw	**Port 23 Continuous Scan Enable** This bit indicates that Port 23 is enabled for continuous scan. Software can disable the port for continuous scan by setting this bit to 0.
PORT24SCE	24	rw	**Port 24 Continuous Scan Enable** This bit indicates that Port 24 is enabled for continuous scan. Software can disable the port for continuous scan by setting this bit to 0.
PORT25SCE	25	rw	**Port 25 Continuous Scan Enable** This bit indicates that Port 25 is enabled for continuous scan. Software can disable the port for continuous scan by setting this bit to 0.
PORT26SCE	26	rw	**Port 26 Continuous Scan Enable** This bit indicates that Port 26 is enabled for continuous scan. Software can disable the port for continuous scan by setting this bit to 0.
PORT27SCE	27	rw	**Port 27 Continuous Scan Enable** This bit indicates that Port 27 is enabled for continuous scan. Software can disable the port for continuous scan by setting this bit to 0.
PORT28SCE	28	rw	**Port 28 Continuous Scan Enable** This bit indicates that Port 28 is enabled for continuous scan. Software can disable the port for continuous scan by setting this bit to 0.
PORT29SCE	29	rw	**Port 29 Continuous Scan Enable** This bit indicates that Port 29 is enabled for continuous scan. Software can disable the port for continuous scan by setting this bit to 0.
PORT30SCE	30	rw	**Port 30 Continuous Scan Enable** This bit indicates that Port 30 is enabled for continuous scan. Software can disable the port for continuous scan by setting this bit to 0.
PORT31SCE	31	rw	**Port 31 Continuous Scan Enable** This bit indicates that Port 31 is enabled for continuous scan. Software can disable the port for continuous scan by setting this bit to 0.

14.6.104 Port j MDIO Interrupt Status

This register gives the source of the interrupt raised by the SMA module on the sbd_intr_o output line. The status bits are cleared when the software reads the corresponding bytes.

Portj_MDIO_Interrupt_Status (j=0-1) Offset address: $10214_H+j*2000_H$

Port j MDIO Interrupt Status Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
PORTNx4P0CONINT	0	r	**Connect/Disconnect Event Interrupt (Port Nx4 Plus0)** When set, indicates that Port Nx4 Plus0 caused an interrupt due to a connection or disconnection. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register Access restriction applies. Clears on reading byte-0 (or when byte-0 is written when RCWE bit in MAC_CSR_SW_Ctrl register is set)
PORTNx4P1CONINT	1	r	**Connect/Disconnect Event Interrupt (Port Nx4 Plus1)** When set, indicates that Port Nx4 Plus1 caused an interrupt due to a connection or disconnection. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register Access restriction applies. Clears on reading byte-0 (or when byte-0 is written when RCWE bit in MAC_CSR_SW_Ctrl register is set)
PORTNx4P2CONINT	2	r	**Connect/Disconnect Event Interrupt (Port Nx4 Plus2)** When set, indicates that Port Nx4 Plus2 caused an interrupt due to a connection or disconnection. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register Access restriction applies. Clears on reading byte-0 (or when byte-0 is written when RCWE bit in MAC_CSR_SW_Ctrl register is set)
PORTNx4P3CONINT	3	r	**Connect/Disconnect Event Interrupt (Port Nx4 Plus3)** When set, indicates that Port Nx4 Plus3 caused an interrupt due to a connection or disconnection. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register Access restriction applies. Clears on reading byte-0 (or when byte-0 is written when RCWE bit in MAC_CSR_SW_Ctrl register is set)
PORTNx4P0LSINT	4	r	**Link Status Change Interrupt (Port Nx4 Plus0)** When set, indicates that the link status changed on at least one of the Port Nx4 Plus0 devices. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register Access restriction applies. Clears on reading byte-0 (or when byte-0 is written when RCWE bit in MAC_CSR_SW_Ctrl register is set)

(table continues...)

(continued)

Field	Bits	Type	Description
PORTNx4P1LSINT	5	r	**Link Status Change Interrupt (Port Nx4 Plus1)** When set, indicates that the link status changed on at least one of the Port Nx4 Plus1 devices. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register Access restriction applies. Clears on reading byte-0 (or when byte-0 is written when RCWE bit in MAC_CSR_SW_Ctrl register is set)
PORTNx4P2LSINT	6	r	**Link Status Change Interrupt (Port Nx4 Plus2)** When set, indicates that the link status changed on at least one of the Port Nx4 Plus2 devices. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register Access restriction applies. Clears on reading byte-0 (or when byte-0 is written when RCWE bit in MAC_CSR_SW_Ctrl register is set)
PORTNx4P3LSINT	7	r	**Link Status Change Interrupt (Port Nx4 Plus3)** When set, indicates that the link status changed on at least one of the Port Nx4 Plus3 devices. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register Access restriction applies. Clears on reading byte-0 (or when byte-0 is written when RCWE bit in MAC_CSR_SW_Ctrl register is set)
PORTNx4P0ALINT	8	r	**Device Present Status Change Interrupt (Port Nx4 Plus0)** When set, indicates that an interrupt occurred due to a change in one or more of the Port Nx4 Plus0 device's Alive status. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register Access restriction applies. Clears on reading byte-1 (or when byte-1 is written when RCWE bit in MAC_CSR_SW_Ctrl register is set)
PORTNx4P1ALINT	9	r	**Device Present Status Change Interrupt (Port Nx4 Plus1)** When set, indicates that an interrupt occurred due to a change in one or more of the Port Nx4 Plus1 device's Alive status. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register Access restriction applies. Clears on reading byte-1 (or when byte-1 is written when RCWE bit in MAC_CSR_SW_Ctrl register is set)
PORTNx4P2ALINT	10	r	**Device Present Status Change Interrupt (Port Nx4 Plus2)** When set, indicates that an interrupt occurred due to a change in one or more of the Port Nx4 Plus2 device's Alive status. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register Access restriction applies. Clears on reading byte-1 (or when byte-1 is written when RCWE bit in MAC_CSR_SW_Ctrl register is set)
PORTNx4P3ALINT	11	r	**Device Present Status Change Interrupt (Port Nx4 Plus3)** When set, indicates that an interrupt occurred due to a change in one or more of the Port Nx4 Plus3 device's Alive status. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register Access restriction applies. Clears on reading byte-1 (or when byte-1 is written when RCWE bit in MAC_CSR_SW_Ctrl register is set)

(table continues...)

(continued)

Field	Bits	Type	Description
SNGLCOMPINT	12	r	**Single Command Completion Interrupt** Set when a single command completes. Note: This bit is available only when PRS field of MDIO Port Nx4 Indirect Control register is 0. Access restriction applies. Clears on read (or write of 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event.
CWCOMPINT	13	r	**Continuous Write Completion Interrupt** Set when a continuous write completes. Note: This bit is valid only when PRS field of MDIO Port Nx4 Indirect Control register is 0. Access restriction applies. Clears on read (or write of 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event.
0	31:14	r	**Reserved** Read as 0; should be written with 0.

14.6.105 Port j MDIO Interrupt Enable

This register controls the enabling of interrupt sources in the MDIO Interrupt Status register.

Portj_MDIO_Interrupt_Enable (j=0-1) Offset address: $10218_H+j*2000_H$

Port j MDIO Interrupt Enable Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		CWCOMPIE	SNGLCOMPIE	PTRNx4P3ALIE	PTRNx4P2ALIE	PTRNx4P1ALIE	PTRNx4P0ALIE	PTRNx4P3LSIE	PTRNx4P2LSIE	PTRNx4P1LSIE	PTRNx4P0LSIE	PTRNx4P3CONIE	PTRNx4P2CONIE	PTRNx4P1CONIE	PTRNx4P0CONIE
r		rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
PTRNx4P0CONIE	0	rw	**Connect/Disconnect Event Interrupt (Port Nx4 Plus0) Enable** When set, this bit enables Connect/Disconnect Event interrupts on Port Nx4 Plus0. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register
PTRNx4P1CONIE	1	rw	**Connect/Disconnect Event Interrupt (Port Nx4 Plus1) Enable** When set, this bit enables Connect/Disconnect Event interrupts on Port Nx4 Plus1. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register

(table continues...)

(continued)

Field	Bits	Type	Description
PTRNx4P2CONIE	2	rw	**Connect/Disconnect Event Interrupt (Port Nx4 Plus2) Enable** When set, this bit enables Connect/Disconnect Event interrupts on Port Nx4 Plus2. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register
PTRNx4P3CONIE	3	rw	**Connect/Disconnect Event Interrupt (Port Nx4 Plus3) Enable** When set, this bit enables Connect/Disconnect Event interrupts on Port Nx4 Plus3. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register
PTRNx4P0LSIE	4	rw	**Link Status Change Interrupt (Port Nx4 Plus0) Enable** When set, this bit enables Link Status Change interrupts on Port Nx4 Plus0. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register
PTRNx4P1LSIE	5	rw	**Link Status Change Interrupt (Port Nx4 Plus1) Enable** When set, this bit enables Link Status Change interrupts on Port Nx4 Plus1. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register
PTRNx4P2LSIE	6	rw	**Link Status Change Interrupt (Port Nx4 Plus2) Enable** When set, this bit enables Link Status Change interrupts on Port Nx4 Plus2. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register
PTRNx4P3LSIE	7	rw	**Link Status Change Interrupt (Port Nx4 Plus3) Enable** When set, this bit enables Link Status Change interrupts on Port Nx4 Plus3. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register
PTRNx4P0ALIE	8	rw	**Device Present Status Change Interrupt (Port Nx4 Plus0) Enable** When set, this bit enables Device Present Status Change interrupts on Port Nx4 Plus0. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register
PTRNx4P1ALIE	9	rw	**Device Present Status Change Interrupt (Port Nx4 Plus1) Enable** When set, this bit enables Device Present Status Change interrupts on Port Nx4 Plus1. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register
PTRNx4P2ALIE	10	rw	**Device Present Status Change Interrupt (Port Nx4 Plus2) Enable** When set, this bit enables Device Present Status Change interrupts on Port Nx4 Plus2. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register
PTRNx4P3ALIE	11	rw	**Device Present Status Change Interrupt (Port Nx4 Plus3) Enable** When set, this bit enables Device Present Status Change interrupts on Port Nx4 Plus3. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register

(table continues...)

(continued)

Field	Bits	Type	Description
SNGLCOMPIE	12	rw	**Single Command Completion Interrupt Enable** When set, this bit enables Single Command Completion interrupts. Note: This bit is available only when the PRS field of MDIO Port Nx4 Indirect Control register is 0.
CWCOMPIE	13	rw	**Continuous Write Completion Interrupt Enable** When set, this bit enables the Continuous Write Completion interrupts. Note: This bit is available only when the PRS field of MDIO Port Nx4 Indirect Control register is 0
0	31:14	r	**Reserved** Read as 0; should be written with 0.

14.6.106 Port j MDIO Port Connect Disconnect Status

This register gives the hot plug-in status for Ports 0-31. This register is updated at the end of any read operation to any register of the corresponding PHY port. It is updated during single-read transfers or during continuous scan operations. During continuous scan operation, a PHY port is only accessed when the corresponding MDIO Continuous Scan Port Enable register bit is set.

Portj_MDIO_Port_Connect_Disconnect_Status (j=0-1) Offset address: $1021C_H+j*2000_H$

Port j MDIO Port Connect Disconnect Status Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PORT31CON	PORT30CON	PORT29CON	PORT28CON	PORT27CON	PORT26CON	PORT25CON	PORT24CON	PORT23CON	PORT22CON	PORT21CON	PORT20CON	PORT19CON	PORT18CON	PORT17CON	PORT16CON
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PORT15CON	PORT14CON	PORT13CON	PORT12CON	PORT11CON	PORT10CON	PORT9CON	PORT8CON	PORT7CON	PORT6CON	PORT5CON	PORT4CON	PORT3CON	PORT2CON	PORT1CON	PORT0CON
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

Field	Bits	Type	Description
PORT0CON	0	r	**Port 0 Connect/Disconnect** The hardware sets this bit when a connection event is detected on Port 0. It is cleared when a disconnection event is detected.
PORT1CON	1	r	**Port 1 Connect/Disconnect** The hardware sets this bit when a connection event is detected on Port 1. It is cleared when a disconnection event is detected.
PORT2CON	2	r	**Port 2 Connect/Disconnect** The hardware sets this bit when a connection event is detected on Port 2. It is cleared when a disconnection event is detected.

(table continues...)

(continued)

Field	Bits	Type	Description
PORT3CON	3	r	**Port 3 Connect/Disconnect** The hardware sets this bit when a connection event is detected on Port 3. It is cleared when a disconnection event is detected.
PORT4CON	4	r	**Port 4 Connect/Disconnect** The hardware sets this bit when a connection event is detected on Port 4. It is cleared when a disconnection event is detected.
PORT5CON	5	r	**Port 5 Connect/Disconnect** The hardware sets this bit when a connection event is detected on Port 5. It is cleared when a disconnection event is detected.
PORT6CON	6	r	**Port 6 Connect/Disconnect** The hardware sets this bit when a connection event is detected on Port 6. It is cleared when a disconnection event is detected.
PORT7CON	7	r	**Port 7 Connect/Disconnect** The hardware sets this bit when a connection event is detected on Port 7. It is cleared when a disconnection event is detected.
PORT8CON	8	r	**Port 8 Connect/Disconnect** The hardware sets this bit when a connection event is detected on Port 8. It is cleared when a disconnection event is detected.
PORT9CON	9	r	**Port 9 Connect/Disconnect** The hardware sets this bit when a connection event is detected on Port 9. It is cleared when a disconnection event is detected.
PORT10CON	10	r	**Port 10 Connect/Disconnect** The hardware sets this bit when a connection event is detected on Port 10. It is cleared when a disconnection event is detected.
PORT11CON	11	r	**Port 11 Connect/Disconnect** The hardware sets this bit when a connection event is detected on Port 11. It is cleared when a disconnection event is detected.
PORT12CON	12	r	**Port 12 Connect/Disconnect** The hardware sets this bit when a connection event is detected on Port 12. It is cleared when a disconnection event is detected.
PORT13CON	13	r	**Port 13 Connect/Disconnect** The hardware sets this bit when a connection event is detected on Port 13. It is cleared when a disconnection event is detected.
PORT14CON	14	r	**Port 14 Connect/Disconnect** The hardware sets this bit when a connection event is detected on Port 14. It is cleared when a disconnection event is detected.
PORT15CON	15	r	**Port 15 Connect/Disconnect** The hardware sets this bit when a connection event is detected on Port 15. It is cleared when a disconnection event is detected.
PORT16CON	16	r	**Port 16 Connect/Disconnect** The hardware sets this bit when a connection event is detected on Port 16. It is cleared when a disconnection event is detected.

(table continues...)

(continued)

Field	Bits	Type	Description
PORT17CON	17	r	**Port 17 Connect/Disconnect** The hardware sets this bit when a connection event is detected on Port 17. It is cleared when a disconnection event is detected.
PORT18CON	18	r	**Port 18 Connect/Disconnect** The hardware sets this bit when a connection event is detected on Port 18. It is cleared when a disconnection event is detected.
PORT19CON	19	r	**Port 19 Connect/Disconnect** The hardware sets this bit when a connection event is detected on Port 19. It is cleared when a disconnection event is detected.
PORT20CON	20	r	**Port 20 Connect/Disconnect** The hardware sets this bit when a connection event is detected on Port 20. It is cleared when a disconnection event is detected.
PORT21CON	21	r	**Port 21 Connect/Disconnect** The hardware sets this bit when a connection event is detected on Port 21. It is cleared when a disconnection event is detected.
PORT22CON	22	r	**Port 22 Connect/Disconnect** The hardware sets this bit when a connection event is detected on Port 22. It is cleared when a disconnection event is detected.
PORT23CON	23	r	**Port 23 Connect/Disconnect** The hardware sets this bit when a connection event is detected on Port 23. It is cleared when a disconnection event is detected.
PORT24CON	24	r	**Port 24 Connect/Disconnect** The hardware sets this bit when a connection event is detected on Port 24. It is cleared when a disconnection event is detected.
PORT25CON	25	r	**Port 25 Connect/Disconnect** The hardware sets this bit when a connection event is detected on Port 25. It is cleared when a disconnection event is detected.
PORT26CON	26	r	**Port 26 Connect/Disconnect** The hardware sets this bit when a connection event is detected on Port 26. It is cleared when a disconnection event is detected.
PORT27CON	27	r	**Port 27 Connect/Disconnect** The hardware sets this bit when a connection event is detected on Port 27. It is cleared when a disconnection event is detected.
PORT28CON	28	r	**Port 28 Connect/Disconnect** The hardware sets this bit when a connection event is detected on Port 28. It is cleared when a disconnection event is detected.
PORT29CON	29	r	**Port 29 Connect/Disconnect** The hardware sets this bit when a connection event is detected on Port 29. It is cleared when a disconnection event is detected.
PORT30CON	30	r	**Port 30 Connect/Disconnect** The hardware sets this bit when a connection event is detected on Port 30. It is cleared when a disconnection event is detected.

(table continues...)

(continued)

Field	Bits	Type	Description
PORT31CON	31	r	**Port 31 Connect/Disconnect** The hardware sets this bit when a connection event is detected on Port 31. It is cleared when a disconnection event is detected.

14.6.107 Port j MDIO Clause 22 Port

This register configures the SMA to access the PHY ports using either the Clause 22 or Clause 45 packet format. All SMA operations (single or continuous) depend on setting these bits correctly.

Portj_MDIO_Clause_22_Port (j=0-1) Offset address: $10220_H+j*2000_H$

Port j MDIO Clause 22 Port Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PTR31CL22	PTR30CL22	PTR29CL22	PTR28CL22	PTR27CL22	PTR26CL22	PTR25CL22	PTR24CL22	PTR23CL22	PTR22CL22	PTR21CL22	PTR20CL22	PTR19CL22	PTR18CL22	PTR17CL22	PTR16CL22
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PTR15CL22	PTR14CL22	PTR13CL22	PTR12CL22	PTR11CL22	PTR10CL22	PTR9CL22	PTR8CL22	PTR7CL22	PTR6CL22	PTR5CL22	PTR4CL22	PTR3CL22	PTR2CL22	PTR1CL22	PTR0CL22
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
PTR0CL22	0	rw	**Port0CL22 Enable** Indicates that Port 0 is connected to a PHY that only supports Clause 22. When this bit is reset, the PHY supports Clause 45.
PTR1CL22	1	rw	**Port1CL22 Enable** Indicates that Port 1 is connected to a PHY that only supports Clause 22. When this bit is reset, the PHY supports Clause 45.
PTR2CL22	2	rw	**Port2CL22 Enable** Indicates that Port 2 is connected to a PHY that only supports Clause 22. When this bit is reset, the PHY supports Clause 45.
PTR3CL22	3	rw	**Port3CL22 Enable** Indicates that Port 3 is connected to a PHY that only supports Clause 22. When this bit is reset, the PHY supports Clause 45.
PTR4CL22	4	rw	**Port4CL22 Enable** Indicates that Port 4 is connected to a PHY that only supports Clause 22. When this bit is reset, the PHY supports Clause 45.
PTR5CL22	5	rw	**Port5CL22 Enable** Indicates that Port 5 is connected to a PHY that only supports Clause 22. When this bit is reset, the PHY supports Clause 45.

(table continues...)

(continued)

Field	Bits	Type	Description
PTR6CL22	6	rw	**Port6CL22 Enable** Indicates that Port 6 is connected to a PHY that only supports Clause 22. When this bit is reset, the PHY supports Clause 45.
PTR7CL22	7	rw	**Port7CL22 Enable** Indicates that Port 7 is connected to a PHY that only supports Clause 22. When this bit is reset, the PHY supports Clause 45.
PTR8CL22	8	rw	**Port8CL22 Enable** Indicates that Port 8 is connected to a PHY that only supports Clause 22. When this bit is reset, the PHY supports Clause 45.
PTR9CL22	9	rw	**Port9CL22 Enable** Indicates that Port 9 is connected to a PHY that only supports Clause 22. When this bit is reset, the PHY supports Clause 45.
PTR10CL22	10	rw	**Port10CL22 Enable** Indicates that Port 10 is connected to a PHY that only supports Clause 22. When this bit is reset, the PHY supports Clause 45.
PTR11CL22	11	rw	**Port11CL22 Enable** Indicates that Port 11 is connected to a PHY that only supports Clause 22. When this bit is reset, the PHY supports Clause 45.
PTR12CL22	12	rw	**Port12CL22 Enable** Indicates that Port 12 is connected to a PHY that only supports Clause 22. When this bit is reset, the PHY supports Clause 45.
PTR13CL22	13	rw	**Port13CL22 Enable** Indicates that Port 13 is connected to a PHY that only supports Clause 22. When this bit is reset, the PHY supports Clause 45.
PTR14CL22	14	rw	**Port14CL22 Enable** Indicates that Port 14 is connected to a PHY that only supports Clause 22. When this bit is reset, the PHY supports Clause 45.
PTR15CL22	15	rw	**Port15CL22 Enable** Indicates that Port 15 is connected to a PHY that only supports Clause 22. When this bit is reset, the PHY supports Clause 45.
PTR16CL22	16	rw	**Port16CL22 Enable** Indicates that Port 16 is connected to a PHY that only supports Clause 22. When this bit is reset, the PHY supports Clause 45.
PTR17CL22	17	rw	**Port17CL22 Enable** Indicates that Port 17 is connected to a PHY that only supports Clause 22. When this bit is reset, the PHY supports Clause 45.
PTR18CL22	18	rw	**Port18CL22 Enable** Indicates that Port 18 is connected to a PHY that only supports Clause 22. When this bit is reset, the PHY supports Clause 45.
PTR19CL22	19	rw	**Port19CL22 Enable** Indicates that Port 19 is connected to a PHY that only supports Clause 22. When this bit is reset, the PHY supports Clause 45.

(table continues...)

(continued)

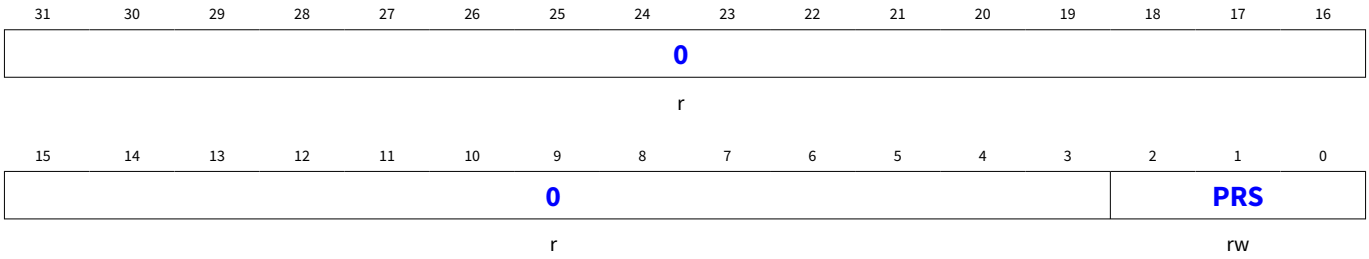
Field	Bits	Type	Description
PTR20CL22	20	rw	**Port20CL22 Enable** Indicates that Port 20 is connected to a PHY that only supports Clause 22. When this bit is reset, the PHY supports Clause 45.
PTR21CL22	21	rw	**Port21CL22 Enable** Indicates that Port 21 is connected to a PHY that only supports Clause 22. When this bit is reset, the PHY supports Clause 45.
PTR22CL22	22	rw	**Port22CL22 Enable** Indicates that Port 22 is connected to a PHY that only supports Clause 22. When this bit is reset, the PHY supports Clause 45.
PTR23CL22	23	rw	**Port23CL22 Enable** Indicates that Port 23 is connected to a PHY that only supports Clause 22. When this bit is reset, the PHY supports Clause 45.
PTR24CL22	24	rw	**Port24CL22 Enable** Indicates that Port 24 is connected to a PHY that only supports Clause 22. When this bit is reset, the PHY supports Clause 45.
PTR25CL22	25	rw	**Port25CL22 Enable** Indicates that Port 25 is connected to a PHY that only supports Clause 22. When this bit is reset, the PHY supports Clause 45.
PTR26CL22	26	rw	**Port26CL22 Enable** Indicates that Port 26 is connected to a PHY that only supports Clause 22. When this bit is reset, the PHY supports Clause 45.
PTR27CL22	27	rw	**Port27CL22 Enable** Indicates that Port 27 is connected to a PHY that only supports Clause 22. When this bit is reset, the PHY supports Clause 45.
PTR28CL22	28	rw	**Port28CL22 Enable** Indicates that Port 28 is connected to a PHY that only supports Clause 22. When this bit is reset, the PHY supports Clause 45.
PTR29CL22	29	rw	**Port29CL22 Enable** Indicates that Port 29 is connected to a PHY that only supports Clause 22. When this bit is reset, the PHY supports Clause 45.
PTR30CL22	30	rw	**Port30CL22 Enable** Indicates that Port 30 is connected to a PHY that only supports Clause 22. When this bit is reset, the PHY supports Clause 45.
PTR31CL22	31	rw	**Port31CL22 Enable** Indicates that Port 31 is connected to a PHY that only supports Clause 22. When this bit is reset, the PHY supports Clause 45.

14.6.108 Port j MDIO Port Nx4 Indirect Control

This register contain fields that control the Port range select for accessing register sets.

Portj_MDIO_Port_Nx4_Indirect_Control (j=0-1) Offset address: $10224_H+j*2000_H$

Port j MDIO Port Nx4 Indirect Control Kernel Reset value: $0000\ 0000_H$



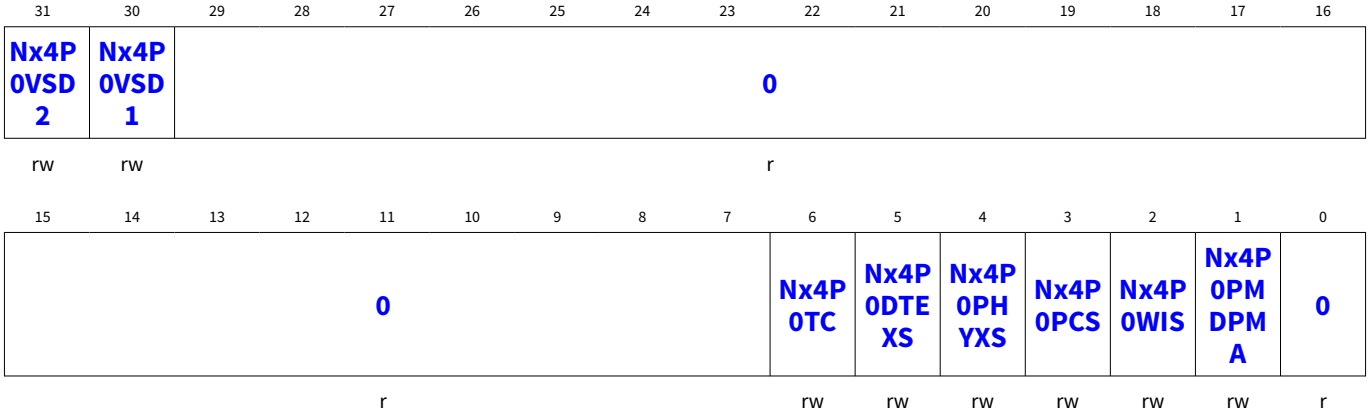
Field	Bits	Type	Description
PRS	2:0	rw	**Port Range Select** This register field controls the Port range select for accessing all the following register sets - MDIO Port Nx4P0/1/2/3 Device In Use register - MDIO Port Nx4P0/1/2/3 Link Status register - MDIO Port Nx4P0/1/2/3 Alive Status register - MDIO Interrupt Status register - MDIO Interrupt Enable register -- 0: Selects the Port address range of 0 to 3 -- 1: Selects the Port Address range of 4 to 7 -- 2: Selects the Port Address range of 8 to 11 -- 3: Selects the Port Address range of 12 to 15 -- 4: Selects the Port Address range of 16 to 19 -- 5: Selects the Port Address range of 20 to 23 -- 6: Selects the Port Address range of 24 to 27 -- 7: Selects the Port Address range of 28 to 31
0	31:3	r	**Reserved** Read as 0; should be written with 0.

14.6.109 Port j MDIO PortNx4P0 Device In Use

This register gives the status of each device on Port Nx4 Plus0 (Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register). Set these bits to indicate that a device is present and that the SMA must access it during continuous scan/write operations. If any bit is cleared, the corresponding device is not scanned when the Continuous Scan command is issued nor is it written when the Continuous Write command is issued. This register setting does not affect single read or write commands.

Portj_MDIO_PortNx4P0_Device_In_Use (j=0-1) Offset address: $10230_H+j*2000_H$

Port j MDIO PortNx4P0 Device In Use Kernel Reset value: $C000\ 007E_H$



Field	Bits	Type	Description
Nx4P0PMDPMA	1	rw	**PMA Device is in Use** When PMD/PMA is present, this bit is set to 1. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register
Nx4P0WIS	2	rw	**WIS Device is in Use** When WIS is present, this bit is set to 1. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register
Nx4P0PCS	3	rw	**PCS Device is in Use** When PCS is present, this bit is set to 1. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register
Nx4P0PHYXS	4	rw	**PHYXS Device is in Use** When PHY XS is present, this bit is set to 1. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register
Nx4P0DTEXS	5	rw	**DTEXS Device is in Use** When DTE XS is present, this bit is set to 1. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register
Nx4P0TC	6	rw	**TC Device is in Use** When TC is present, this bit is set to 1. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register
Nx4P0VSD1	30	rw	**VSD1 Device is in Use** When Vendor-Specific Device 1 is present, this bit is set to 1. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register
Nx4P0VSD2	31	rw	**VSD2 Device is in Use** When Vendor-Specific Device 2 is present, this bit is set to 1. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register
0	0, 29:7	r	**Reserved** Read as 0; should be written with 0.

14.6.110 Port j MDIO PortNx4P0 Link Status

This register gives the link status of the devices in PHY Port Nx4Plus0 ((Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register) at the end of each continuous scan cycle. Up to eight devices are supported, as specified in IEEE 802.3-2018. You can disable the polling of any device by clearing its corresponding MDIO Port Nx4Plus0 Device In Use register bit.

Note: All the register fields are renamed from P0 to Nx4P0 (Nx4 Plus 0)

Portj_MDIO_PortNx4P0_Link_Status (j=0-1) Offset address: $10234_H+j*2000_H$

Port j MDIO PortNx4P0 Link Status Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Nx4P0VSD2LS	Nx4P0VSD1LS	0													
r	r	r													

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0									Nx4P0TCLS	Nx4P0DTEXSLS	Nx4P0PHYXSLS	Nx4P0PCSLS	Nx4P0WISLS	Nx4P0PMDPMALS	0
r									r	r	r	r	r	r	r

Field	Bits	Type	Description
Nx4P0PMDPMALS	1	r	**Link Status of PMA Device** When the PMD/PMA link is up, this bit is set to 1. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register
Nx4P0WISLS	2	r	**Link Status of WIS Device** When the WIS link is up, this bit is set to 1. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register
Nx4P0PCSLS	3	r	**Link Status of PCS Device** When the PCS link is up, this bit is set to 1. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register
Nx4P0PHYXSLS	4	r	**Link Status of PHYXS Device** When the PHY XS link is up, this bit is set to 1. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register
Nx4P0DTEXSLS	5	r	**Link Status of DTEXS Device** When the DTE XS link is up, this bit is set to 1. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register
Nx4P0TCLS	6	r	**Link Status of TC Device** When the TC link is up, this bit is set to 1. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register
Nx4P0VSD1LS	30	r	**Link Status of VCD1 Device** When the Vendor-Specific Device 1 link is up, this bit is set to 1. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register

(table continues...)

(continued)

Field	Bits	Type	Description
Nx4P0VSD2LS	31	r	**Link Status of VSD2 Device** When the Vendor-Specific Device 2 link is up, this bit is set to 1. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register
0	0, 29:7	r	**Reserved** Read as 0; should be written with 0.

14.6.111 Port j MDIO PortNx4P0 Alive Status

This register gives the device status in PHY Port Nx4Plus0 ((Here N is as per PRS field of "MDIO Port Nx4 Indirect Control" register) at the end of each continuous scan cycle. Up to eight devices are supported, as specified in IEEE 802.3-2018. You can disable the polling of any device by clearing its corresponding MDIO Port Nx4Plus0 Device In Use register bit.

Note: All the register fields are renamed from P0 to Nx4P0 (Nx4 Plus 0)

Portj_MDIO_PortNx4P0_Alive_Status (j=0-1) Offset address: $10238_H+j*2000_H$

Port j MDIO PortNx4P0 Alive Status Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Nx4P0VSD2LS	Nx4P0VSD1LS	0													
r	r	r													

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0									Nx4P0TCLS	Nx4P0DTEXSLS	Nx4P0PHYXSLS	Nx4P0PCSLS	Nx4P0WISLS	Nx4P0PMDPMALS	0
r									r	r	r	r	r	r	r

Field	Bits	Type	Description
Nx4P0PMDPMALS	1	r	**Alive Status of PMA Device** When PMD/PMA is present and functioning properly, this bit is set to 1. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register
Nx4P0WISLS	2	r	**Alive Status of WIS Device** When WIS is present and functioning properly, this bit is set to 1. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register
Nx4P0PCSLS	3	r	**Alive Status of PCS Device** When PCS is present and functioning properly, this bit is set to 1. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register
Nx4P0PHYXSLS	4	r	**Alive Status of PHYXS Device** When PHY XS is present and functioning properly, this bit is set to 1. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register

(table continues...)

(continued)

Field	Bits	Type	Description
Nx4P0DTEXSLS	5	r	**Alive Status of DTEXS Device** When DTE XS is present and functioning properly, this bit is set to 1. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register
Nx4P0TCLS	6	r	**Alive Status of TC Device** When TC is present and functioning properly, this bit is set to 1. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register
Nx4P0VSD1LS	30	r	**Alive Status of VSD1 Device** When Vendor-Specific Device 1 is present and functioning properly, this bit is set to 1. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register
Nx4P0VSD2LS	31	r	**Alive Status of VSD2 Device** When Vendor-Specific Device 2 is present and functioning properly, this bit is set to 1. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register
0	0, 29:7	r	**Reserved** Read as 0; should be written with 0.

14.6.112 Port j MDIO PortNx4P1 Device In Use

This register gives the status of each device on Port Nx4 Plus1 (Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register). Set these bits to indicate that a device is present and that the SMA must access it during continuous scan/write operations. If any bit is cleared, the corresponding device is not scanned when the Continuous Scan command is issued nor is it written when the Continuous Write command is issued. This register setting does not affect single read or write commands.

Note: All the register fields are renamed from P1 to Nx4P1 (Nx4 Plus 1)

Portj_MDIO_PortNx4P1_Device_In_Use (j=0-1) Offset address: $10240_H+j*2000_H$

Port j MDIO PortNx4P1 Device In Use Kernel Reset value: C000 007E$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Nx4P1VSD2	Nx4P1VSD1	0													
rw	rw	r													

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0									Nx4P1TC	Nx4P1DTEXS	Nx4P1PHYXS	Nx4P1PCS	Nx4P1WIS	Nx4P1PMDPMA	0
r									rw	rw	rw	rw	rw	rw	r

Field	Bits	Type	Description
Nx4P1PMDPMA	1	rw	**PMA Device is in Use** When PMD/PMA is present, this bit is set to 1. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register
Nx4P1WIS	2	rw	**WIS Device is in Use** When WIS is present, this bit is set to 1. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register
Nx4P1PCS	3	rw	**PCS Device is in Use** When PCS is present, this bit is set to 1. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register
Nx4P1PHYXS	4	rw	**PHYXS Device is in Use** When PHY XS is present, this bit is set to 1. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register
Nx4P1DTEXS	5	rw	**DTEXS Device is in Use** When DTE XS is present, this bit is set to 1. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register
Nx4P1TC	6	rw	**TC Device is in Use** When TC is present, this bit is set to 1. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register
Nx4P1VSD1	30	rw	**VSD1 Device is in Use** When Vendor-Specific Device 1 is present, this bit is set to 1. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register
Nx4P1VSD2	31	rw	**VSD2 Device is in Use** When Vendor-Specific Device 2 is present, this bit is set to 1. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register
0	0, 29:7	r	**Reserved** Read as 0; should be written with 0.

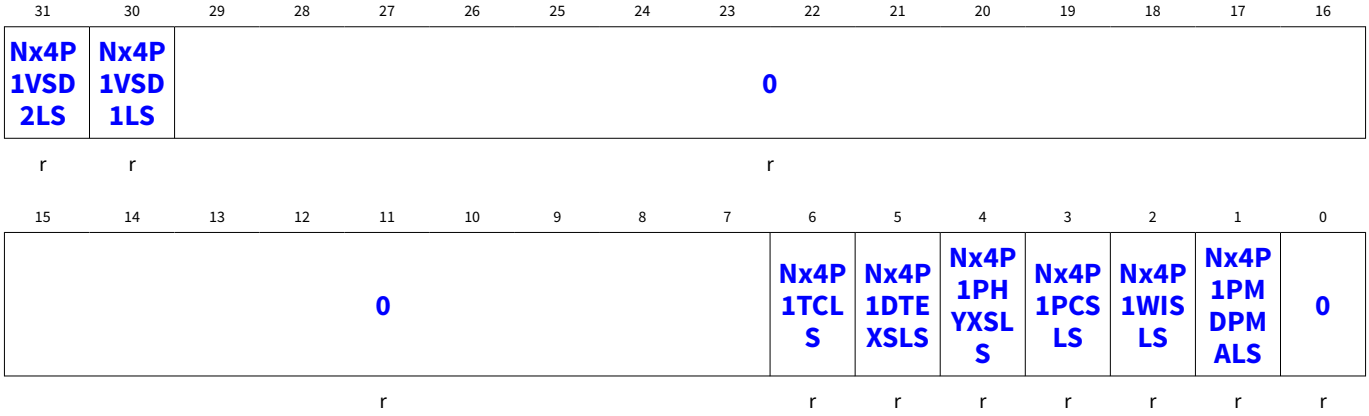
14.6.113 Port j MDIO PortNx4P1 Link Status

This register gives the link status of the devices in PHY Port Nx4Plus1 ((Here N is as per PRS field of MDIO Port Nx4 Indirect Control register) at the end of each continuous scan cycle. Up to eight devices are supported, as specified in IEEE 802.3-2018. You can disable the polling of any device by clearing its corresponding MDIO Port Nx4Plus1 Device In Use register bit.

Note: All the register fields are renamed from P1 to Nx4P1 (Nx4 Plus 1)

Portj_MDIO_PortNx4P1_Link_Status (j=0-1) Offset address: $10244_H+j*2000_H$

Port j MDIO PortNx4P1 Link Status Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
Nx4P1PMDPMALS	1	r	**Link Status of PMA Device** When the PMD/PMA link is up, this bit is set to 1. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register
Nx4P1WISLS	2	r	**Link Status of WIS Device** When the WIS link is up, this bit is set to 1. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register
Nx4P1PCSLS	3	r	**Link Status of PCS Device** When the PCS link is up, this bit is set to 1. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register
Nx4P1PHYXSLS	4	r	**Link Status of PHYXS Device** When the PHY XS link is up, this bit is set to 1. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register
Nx4P1DTEXSLS	5	r	**Link Status of DTEXS Device** When the DTE XS link is up, this bit is set to 1. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register
Nx4P1TCLS	6	r	**Link Status of TC Device** When the TC link is up, this bit is set to 1. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register
Nx4P1VSD1LS	30	r	**Link Status of VSD1 Device** When the Vendor-Specific Device 1 link is up, this bit is set to 1. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register
Nx4P1VSD2LS	31	r	**Link Status of VSD2 Device** When the Vendor-Specific Device 2 link is up, this bit is set to 1. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register
0	0, 29:7	r	**Reserved** Read as 0; should be written with 0.

14.6.114 Port j MDIO PortNx4P1 Alive Status

This register gives the device status in PHY Port Nx4Plus1 ((Here N is as per PRS field of MDIO Port Nx4 Indirect Control register) at the end of each continuous scan cycle. Up to eight devices are supported, as specified in IEEE 802.3-2018. You can disable the polling of any device by clearing its corresponding MDIO Port Nx4Plus1 Device In Use register bit.

Note: All the register fields are renamed from P1 to Nx4P1 (Nx4 Plus 1)

Portj_MDIO_PortNx4P1_Alive_Status (j=0-1) Offset address: $10248_H+j*2000_H$

Port j MDIO PortNx4P1 Alive Status Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Nx4P1VSD2LS	Nx4P1VSD1LS	0													
r	r	r													

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0									Nx4P1TCLS	Nx4P1DTEXSLS	Nx4P1PHYXSLS	Nx4P1PCSLS	Nx4P1WISLS	Nx4P1PMDPMALS	0
r									r	r	r	r	r	r	r

Field	Bits	Type	Description
Nx4P1PMDPMALS	1	r	**Alive Status of PMA Device** When PMD/PMA is present and functioning properly, this bit is set to 1. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register
Nx4P1WISLS	2	r	**Alive Status of WIS Device** When WIS is present and functioning properly, this bit is set to 1. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register
Nx4P1PCSLS	3	r	**Alive Status of PCS Device** When PCS is present and functioning properly, this bit is set to 1. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register
Nx4P1PHYXSLS	4	r	**Alive Status of PHYXS Device** When PHY XS is present and functioning properly, this bit is set to 1. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register
Nx4P1DTEXSLS	5	r	**Alive Status of DTEXS Device** When DTE XS is present and functioning properly, this bit is set to 1. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register
Nx4P1TCLS	6	r	**Alive Status of TC Device** When TC is present and functioning properly, this bit is set to 1. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register
Nx4P1VSD1LS	30	r	**Alive Status of VSD1 Device** When Vendor-Specific Device 1 is present and functioning properly, this bit is set to 1. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register

(table continues...)

(continued)

Field	Bits	Type	Description
Nx4P1VSD2LS	31	r	**Alive Status of VSD2 Device** When Vendor-Specific Device 2 is present and functioning properly, this bit is set to 1. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register
0	0, 29:7	r	**Reserved** Read as 0; should be written with 0.

14.6.115 Port j MDIO PortNx4P2 Device In Use

This register gives the status of each device on Port Nx4 Plus2 (Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register). Set these bits to indicate that a device is present and that the SMA must access it during continuous scan/write operations. If any bit is cleared, the corresponding device is not scanned when the Continuous Scan command is issued nor is it written when the Continuous Write command is issued. This register setting does not affect single read or write commands.

Note: All the register fields are renamed from P2 to Nx4P2 (Nx4 Plus 2)

Portj_MDIO_PortNx4P2_Device_In_Use (j=0-1) Offset address: $10250_H+j*2000_H$

Port j MDIO PortNx4P2 Device In Use Kernel Reset value: $C000\ 007E_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Nx4P2VSD2	Nx4P2VSD1	0													
rw	rw	r													

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0									Nx4P2TC	Nx4P2DTEXS	Nx4P2PHYXS	Nx4P2PCS	Nx4P2WIS	Nx4P2PMDPMA	0
r									rw	rw	rw	rw	rw	rw	r

Field	Bits	Type	Description
Nx4P2PMDPMA	1	rw	**PMA Device is in Use** When PMD/PMA is present, this bit is set to 1. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register
Nx4P2WIS	2	rw	**WIS Device is in Use** When WIS is present, this bit is set to 1. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register
Nx4P2PCS	3	rw	**PCS Device is in Use** When PCS is present, this bit is set to 1. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register
Nx4P2PHYXS	4	rw	**PHYXS Device is in Use** When PHY XS is present, this bit is set to 1. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register

(table continues...)

(continued)

Field	Bits	Type	Description
Nx4P2DTEXS	5	rw	**DTEXS Device is in Use** When DTE XS is present, this bit is set to 1. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register
Nx4P2TC	6	rw	**TC Device is in Use** When TC is present, this bit is set to 1. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register
Nx4P2VSD1	30	rw	**VSD1 Device is in Use** When Vendor-Specific Device 1 is present, this bit is set to 1. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register
Nx4P2VSD2	31	rw	**VSD2 Device is in Use** When Vendor-Specific Device 2 is present, this bit is set to 1. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register
0	0, 29:7	r	**Reserved** Read as 0; should be written with 0.

14.6.116 Port j MDIO PortNx4P2 Link Status

This register gives the link status of the devices in PHY Port Nx4Plus2 ((Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register) at the end of each continuous scan cycle. Up to eight devices are supported, as specified in IEEE 802.3-2018. You can disable the polling of any device by clearing its corresponding MDIO Port Nx4Plus1 Device In Use register bit.

Note: All the register fields are renamed from P2 to Nx4P2 (Nx4 Plus 2)

Portj_MDIO_PortNx4P2_Link_Status (j=0-1) Offset address: $10254_H+j*2000_H$

Port j MDIO PortNx4P2 Link Status Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Nx4P2VSD2LS	Nx4P2VSD1LS	0													
r	r	r													

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0									Nx4P2TCLS	Nx4P2DTEXSLS	Nx4P2PHYXSLS	Nx4P2PCSLS	Nx4P2WISLS	Nx4P2PMDPMALS	0
r				r					r	r	r	r	r	r	r

Field	Bits	Type	Description
Nx4P2PMDPMALS	1	r	**Link Status of PMA Device** When the PMD/PMA link is up, this bit is set to 1. Here N is as per PRS field of MDIO Port Nx4 Indirect Control register

(table continues...)

(continued)

Field	Bits	Type	Description
Nx4P2WISLS	2	r	**Link Status of WIS Device** When the WIS link is up, this bit is set to 1. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register
Nx4P2PCSLS	3	r	**Link Status of PCS Device** When the PCS link is up, this bit is set to 1. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register
Nx4P2PHYXSLS	4	r	**Link Status of PHYXS Device** When the PHY XS link is up, this bit is set to 1. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register
Nx4P2DTEXSLS	5	r	**Link Status of DTEXS Device** When the DTE XS link is up, this bit is set to 1. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register
Nx4P2TCLS	6	r	**Link Status of TC Device** When the TC link is up, this bit is set to 1. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register
Nx4P2VSD1LS	30	r	**Link Status of VSD1 Device** When the Vendor-Specific Device 1 link is up, this bit is set to 1. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register
Nx4P2VSD2LS	31	r	**Link Status of VSD2 Device** When the Vendor-Specific Device 2 link is up, this bit is set to 1. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register
0	0, 29:7	r	**Reserved** Read as 0; should be written with 0.

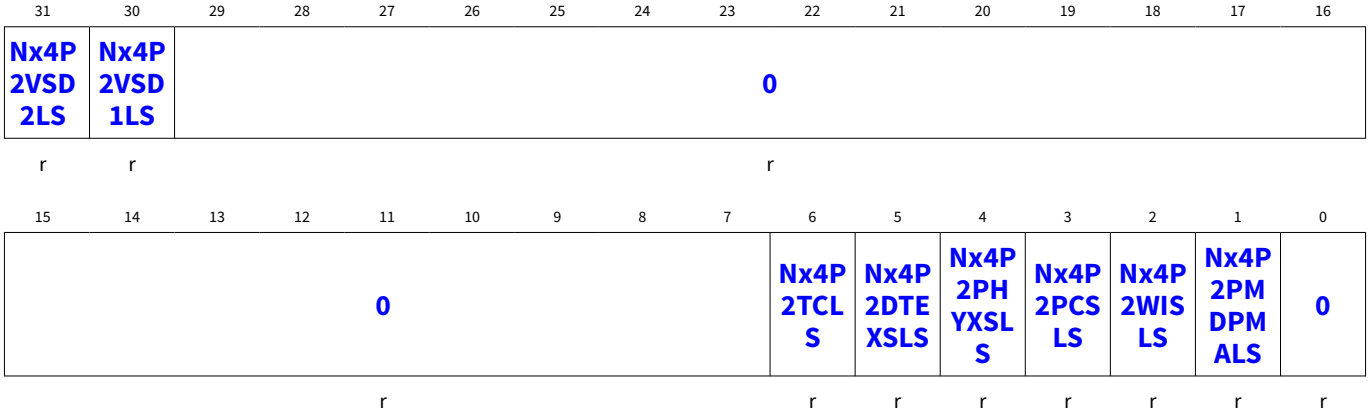
14.6.117 Port j MDIO PortNx4P2 Alive Status

This register gives the device status in PHY Port Nx4Plus2 ((Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register) at the end of each continuous scan cycle. Up to eight devices are supported, as specified in IEEE 802.3-2018. You can disable the polling of any device by clearing its corresponding MDIO Port Nx4Plus2 Device In Use register bit.

Note: All the register fields are renamed from P2 to Nx4P2 (Nx4 Plus 2)

Portj_MDIO_PortNx4P2_Alive_Status (j=0-1) Offset address: $10258_H+j*2000_H$

Port j MDIO PortNx4P2 Alive Status Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
Nx4P2PMDPMALS	1	r	**Alive Status of PMA Device** When PMD/PMA is present and functioning properly, this bit is set to 1. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register
Nx4P2WISLS	2	r	**Alive Status of WIS Device** When WIS is present and functioning properly, this bit is set to 1. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register
Nx4P2PCSLS	3	r	**Alive Status of PCS Device** When PCS is present and functioning properly, this bit is set to 1. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register
Nx4P2PHYXSLS	4	r	**Alive Status of PHYXS Device** When PHY XS is present and functioning properly, this bit is set to 1. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register
Nx4P2DTEXSLS	5	r	**Alive Status of DTEXS Device** When DTE XS is present and functioning properly, this bit is set to 1. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register
Nx4P2TCLS	6	r	**Alive Status of TC Device** When TC is present and functioning properly, this bit is set to 1. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register
Nx4P2VSD1LS	30	r	**Alive Status of VSD1 Device** When Vendor-Specific Device 1 is present and functioning properly, this bit is set to 1. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register
Nx4P2VSD2LS	31	r	**Alive Status of VSD2 Device** When Vendor-Specific Device 2 is present and functioning properly, this bit is set to 1. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register
0	0, 29:7	r	**Reserved** Read as 0; should be written with 0.

14.6.118 Port j MDIO PortNx4P3 Device In Use

This register gives the status of each device on Port Nx4 Plus3 (Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register). Set these bits to indicate that a device is present and that the SMA must access it during continuous scan/write operations. If any bit is cleared, the corresponding device is not scanned when the Continuous Scan command is issued nor is it written when the Continuous Write command is issued. This register setting does not affect single read or write commands.

Note: All the register fields are renamed from P3 to Nx4P3 (Nx4 Plus 3)

Portj_MDIO_PortNx4P3_Device_In_Use (j=0-1) Offset address: $10260_H+j*2000_H$

Port j MDIO PortNx4P3 Device In Use Kernel Reset value: $C000\ 007E_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Nx4P3VSD2	Nx4P3VSD1	0													
rw	rw	r													

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0									Nx4P3TC	Nx4P3DTEXS	Nx4P3PHYXS	Nx4P3PCS	Nx4P3WIS	Nx4P3PMDPMA	0
r									rw	rw	rw	rw	rw	rw	r

Field	Bits	Type	Description
Nx4P3PMDPMA	1	rw	**PMA Device is in Use** When PMD/PMA is present, this bit is set to 1. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register
Nx4P3WIS	2	rw	**WIS Device is in Use** When WIS is present, this bit is set to 1. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register
Nx4P3PCS	3	rw	**PCS Device is in Use** When PCS is present, this bit is set to 1. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register
Nx4P3PHYXS	4	rw	**PHYXS Device is in Use** When PHY XS is present, this bit is set to 1. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register
Nx4P3DTEXS	5	rw	**DTEXS Device is in Use** When DTE XS is present, this bit is set to 1. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register
Nx4P3TC	6	rw	**TC Device is in Use** When TC is present, this bit is set to 1. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register
Nx4P3VSD1	30	rw	**VSD1 Device is in Use** When Vendor-Specific Device 1 is present, this bit is set to 1. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register

(table continues...)

(continued)

Field	Bits	Type	Description
Nx4P3VSD2	31	rw	**VSD2 Device is in Use** When Vendor-Specific Device 2 is present, this bit is set to 1. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register
0	0, 29:7	r	**Reserved** Read as 0; should be written with 0.

14.6.119 Port j MDIO PortNx4P3 Link Status

This register gives the link status of the devices in PHY Port Nx4Plus3 ((Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register) at the end of each continuous scan cycle. Up to eight devices are supported, as specified in IEEE 802.3-2018. You can disable the polling of any device by clearing its corresponding MDIO Port Nx4Plus3 Device In Use register bit.

Note: All the register fields are renamed from P3 to Nx4P3 (Nx4 Plus 3)

Portj_MDIO_PortNx4P3_Link_Status (j=0-1) Offset address: $10264_H+j*2000_H$

Port j MDIO PortNx4P3 Link Status Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Nx4P3VSD2LS	Nx4P3VSD1LS	0													
r	r	r													

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0									Nx4P3TCLS	Nx4P3DTEXSLS	Nx4P3PHYXSLS	Nx4P3PCSLS	Nx4P3WISLS	Nx4P3PMDPMALS	0
r									r	r	r	r	r	r	r

Field	Bits	Type	Description
Nx4P3PMDPMALS	1	r	**Link Status of PMA Device** When the PMD/PMA link is up, this bit is set to 1. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register
Nx4P3WISLS	2	r	**Link Status of WIS Device** When the WIS link is up, this bit is set to 1. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register
Nx4P3PCSLS	3	r	**Link Status of PCS Device** When the PCS link is up, this bit is set to 1. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register
Nx4P3PHYXSLS	4	r	**Link Status of PHYXS Device** When the PHY XS link is up, this bit is set to 1. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register

(table continues...)

(continued)

Field	Bits	Type	Description
Nx4P3DTEXSLS	5	r	**Link Status of DTEXS Device** When the DTE XS link is up, this bit is set to 1. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register
Nx4P3TCLS	6	r	**Link Status of TC Device** When the TC link is up, this bit is set to 1. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register
Nx4P3VSD1LS	30	r	**Link Status of VSD1 Device** When the Vendor-Specific Device 1 link is up, this bit is set to 1. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register
Nx4P3VSD2LS	31	r	**Link Status of VSD2 Device** When the Vendor-Specific Device 2 link is up, this bit is set to 1. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register
0	0, 29:7	r	**Reserved** Read as 0; should be written with 0.

14.6.120 Port j MDIO PortNx4P3 Alive Status

This register gives the device status in PHY Port Nx4Plus3 ((Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register) at the end of each continuous scan cycle. Up to eight devices are supported, as specified in IEEE 802.3-2018. You can disable the polling of any device by clearing its corresponding MDIO Port Nx4Plus0 Device In Use register bit.

Note: All the register fields are renamed from P3 to Nx4P3 (Nx4 Plus 3)

Portj_MDIO_PortNx4P3_Alive_Status (j=0-1) Offset address: $10268_H+j*2000_H$

Port j MDIO PortNx4P3 Alive Status Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Nx4P3VSD2LS	Nx4P3VSD1LS	0													
r	r								r						

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0									Nx4P3TCLS	Nx4P3DTEXSLS	Nx4P3PHYXSLS	Nx4P3PCSLS	Nx4P3WISLS	Nx4P3PMDPMALS	0
				r					r	r	r	r	r	r	r

Field	Bits	Type	Description
Nx4P3PMDPMALS	1	r	**Alive Status of PMA Device** When PMD/PMA is present and functioning properly, this bit is set to 1. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register

(table continues...)

(continued)

Field	Bits	Type	Description
Nx4P3WISLS	2	r	**Alive Status of WIS Device** When WIS is present and functioning properly, this bit is set to 1. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register
Nx4P3PCSLS	3	r	**Alive Status of PCS Device** When PCS is present and functioning properly, this bit is set to 1. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register
Nx4P3PHYXSLS	4	r	**Alive Status of PHYXS Device** When PHY XS is present and functioning properly, this bit is set to 1. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register
Nx4P3DTEXSLS	5	r	**Alive Status of DTEXS Device** When DTE XS is present and functioning properly, this bit is set to 1. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register
Nx4P3TCLS	6	r	**Alive Status of TC Device** When TC is present and functioning properly, this bit is set to 1. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register
Nx4P3VSD1LS	30	r	**Alive Status of VSD1 Device** When Vendor-Specific Device 1 is present and functioning properly, this bit is set to 1. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register
Nx4P3VSD2LS	31	r	**Alive Status of VSD2 Device** When Vendor-Specific Device 2 is present and functioning properly, this bit is set to 1. Here N is as per the PRS field of MDIO Port Nx4 Indirect Control register
0	0, 29:7	r	**Reserved** Read as 0; should be written with 0.

14.6.121 Port j MAC FPE CTRL STS

This register controls the operation of Frame Preemption.

Portj_MAC_FPE_CTRL_STS (j=0-1) Offset address: $10280_H+j*2000_H$

Port j MAC FPE CTRL STS Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0												TRSP	TVER	RRSP	RVER
r												r	r	r	r

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												ARV	SRSP	SVER	EFPE
r												rw	rw	rw	rw

Field	Bits	Type	Description
EFPE	0	rw	**Enable Tx Frame Preemption** When set, Frame Preemption Tx functionality is enabled. 0_B **DISABLE**: Tx Frame Preemption is disabled 1_B **ENABLE**: Tx Frame Preemption is enabled
SVER	1	rw	**Send Verify mPacket** When set, this bit enables hardware to send a verify mPacket. Reset by hardware after sending the Verify mPacket. Access restriction applies. Writing 1 sets the bit. Self-cleared. Writing 0 has no effect. 0_B **DISABLE**: Send Verify mPacket is disabled 1_B **ENABLE**: Send Verify mPacket is enabled
SRSP	2	rw	**Send Respond mPacket** When set, this bit enabled hardware to send a Respond mPacket. Reset by hardware after sending the Respond mPacket. Access restriction applies. Can be set by software or by hardware when ARV bit is set and a verify mPacket is received. Self-cleared. Writing 0 has no effect. 0_B **DISABLE**: Send Respond mPacket is disabled 1_B **ENABLE**: Send Respond mPacket is enabled
ARV	3	rw	**Autogenerate Respond mPacket on receiving Verify mPacket** When set, function to autogenerate a Respond mPacket on receiving Verify mPacket is enabled. 0_B **DISABLE**: Autogenerate Respond mPacket on receiving Verify mPakcet is disabled 1_B **ENABLE**: Autogenerate Respond mPacket on receiving Verify mPakcet is enabled
RVER	16	r	**Received Verify Frame** This field is set when a Verify mPacket is received. An interrupt can be generated for this event if FPEIE bit of MAC_Interrupt_Enable is set. Access restriction applies. Clears on read (or write of 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event. 0_B **INACTIVE**: Not received Verify Frame 1_B **ACTIVE**: Received Verify Frame
RRSP	17	r	**Received Respond Frame** This field is set when a Respond mPacket is received. An interrupt can be generated for this event if FPEIE bit of MAC_Interrupt_Enable is set. Access restriction applies. Clears on read (or write of 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event. 0_B **INACTIVE**: Not received Respond Frame 1_B **ACTIVE**: Received Respond Frame

(table continues...)

(continued)

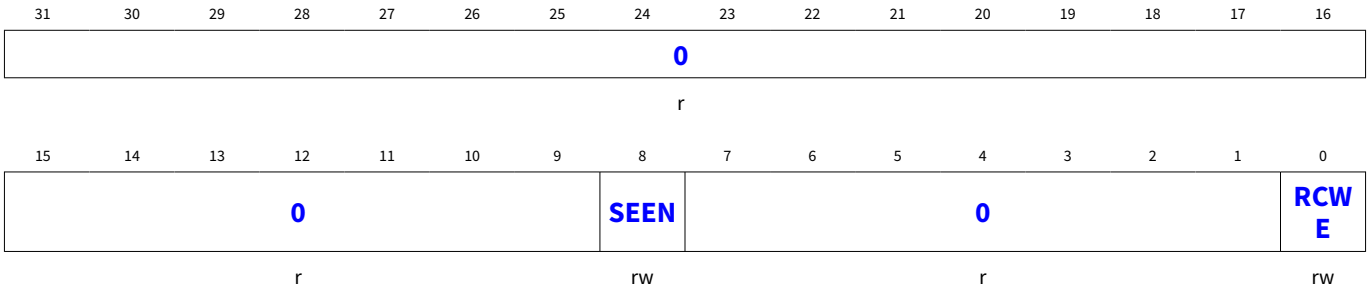
Field	Bits	Type	Description
TVER	18	r	**Transmitted Verify Frame** This field is set when a Verify mPacket is transmitted (triggered by setting SVER field). An interrupt can be generated for this event if FPEIE bit of MAC_Interrupt_Enable is set. Access restriction applies. Clears on read (or write of 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event. 0_B **INACTIVE**: Not transmitted Verify Frame 1_B **ACTIVE**: transmitted Verify Frame
TRSP	19	r	**Transmitted Respond Frame** This field is set when a Respond mPacket is transmitted (triggered by setting SRSP field). An interrupt can be generated for this event if FPEIE bit of MAC_Interrupt_Enable is set. Access restriction applies. Clears on read (or write of 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event. 0_B **INACTIVE**: Not transmitted Respond Frame 1_B **ACTIVE**: transmitted Respond Frame
0	15:4, 31:20	r	**Reserved** Read as 0; should be written with 0.

14.6.122 Port j MAC CSR SW Ctrl

This register contains software programmable controls for changing the CSR access response and status bits clearing.

Portj_MAC_CSR_SW_Ctrl (j=0-1) Offset address: $10290_H+j*2000_H$

Port j MAC CSR SW Ctrl Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
RCWE	0	rw	**Register Clear on Write 1 Enable** When this bit is set, the access mode of list of register fields provided in Appendix F changes to Clear on Write 1, the application needs to set that respective bit to 1 to clear it. When this bit is reset, the access mode of list of register fields provided in Appendix F remain as Clear on Read. 0_B **DISABLE**: Register Clear on Write 1 Disable 1_B **ENABLE**: Register Clear on Write 1 Enable

(table continues...)

(continued)

Field	Bits	Type	Description
SEEN	8	rw	**Slave Error Response Enable - When this bit is set, the MAC responds with Slave Error for accesses to reserved registers in CSR space** - When this bit is reset, the MAC responds with Okay response to any register accessed from CSR space. 0_B **DISABLE**: Slave Error response Disabled 1_B **ENABLE**: Slave Error response Enabled
0	7:1, 31:9	r	**Reserved** Read as 0; should be written with 0.

14.6.123 Port j MAC Address0 High

The MAC_Address0_High register holds the upper 16 bits of the first 6-byte MAC address of the station. The first DA byte that is received on the (X)GMII interface corresponds to the LS byte (Bits [7:0]) of the MAC_Address0_Low register. For example, if 0x112233445566 is received (0x11 in lane 0 of the first column) on the (X)GMII as the destination address, then the MAC_Address0 register [47:0] is compared with 0x665544332211.

If the MAC address registers are configured to be double-synchronized to the (X)GMII clock domain or Power Management feature is selected, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC_Address0_Low register are written, so MAC_Address0_High register must be written prior to MAC_Address0_Low register. For proper synchronization updates, the consecutive writes to MAC_Address0_Low register must be performed after at least four clock cycles of the slowest clock between the fGETH and (X)GMII clock domains.

Portj_MAC_Address0_High (j=0-1) Offset address: $10300_H+j*2000_H$

Port j MAC Address0 High Kernel Reset value: 8000 FFFF$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AE	0												DCS		
r	r												rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRHI															
rw															

Field	Bits	Type	Description
ADDRHI	15:0	rw	**MAC Address0[47:32]** This field contains the upper 16 bits[47:32] of the first 6-byte MAC address. The MAC uses this field for filtering the received packets and inserting the MAC address in the Transmit Flow Control (Pause) Packets.

(table continues...)

(continued)

Field	Bits	Type	Description
DCS	18:16	rw	**DMA Channel Select** If the DDS bit of MAC_Extended_Configuration register is not set: This field contains the binary representation of the DMA Channel number to which an Rx packet whose DA matches the MAC Address0 content is routed. If the DDS bit of MAC_Extended_Configuration register is set: XDCS field of MAC_DChSel_IndReg register contains the one-hot representation of one or more DMA Channel numbers to which an Rx packet whose DA matches the MAC Address0 content is routed.
AE	31	r	**Address Enable** This bit is always set to 1.
0	30:19	r	**Reserved** Read as 0; should be written with 0.

14.6.124 Port j MAC Address0 Low

The MAC_Address0_Low register holds the lower 32 bits of the 0th 6-byte MAC address of the station.

Portj_MAC_Address0_Low (j=0-1) Offset address: $10304_H+j*2000_H$

Port j MAC Address0 Low Kernel Reset value: FFFF FFFF$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADDRLO															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRLO															
rw															

Field	Bits	Type	Description
ADDRLO	31:0	rw	**MAC Address0 [31:0]** This field contains the lower 32 bits of 0th 6-byte MAC address. The MAC uses this field for filtering the received packets and inserting the MAC address in the Transmit Flow Control (Pause) Packets.

14.6.125 Port j MAC Address1 High

The MAC_Address1_High register holds the upper 16 bits of the 1th 6-byte MAC address of the station. The first DA byte that is received on the (X)GMII interface corresponds to the LS byte (Bits [7:0]) of the MAC_Address1_Low register. For example, if 0x112233445566 is received (0x11 in lane 0 of the first column) on the (X)GMII as the destination address, then the MAC_Address1 register [47:0] is compared with 0x665544332211.

If the MAC address registers are configured to be double-synchronized to the (X)GMII clock domain, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the MAC_Address1_Low register are written, so MAC_Address1_High register must be written prior to MAC_Address1_Low register. For proper synchronization updates, the consecutive writes to MAC_Address1_Low register must be performed after at least four clock cycles of the slowest clock between the source and the destination clock domains.

Portj_MAC_Address1_High (j=0-1) Offset address: $10308_H+j*2000_H$

Port j MAC Address1 High Kernel Reset value: $0000\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AE	SA	MBC						0					DCS		
rw	rw	rw						r					rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRHI															
rw															

Field	Bits	Type	Description
ADDRHI	15:0	rw	**MAC Address1 [47:32]** This field contains the upper 16 bits[47:32] of the 1th 6-byte MAC address.
DCS	18:16	rw	**DMA Channel Select** If the DDS bit of MAC_Extended_Configuration register is not set: This field contains the binary representation of the DMA Channel number to which an Rx packet whose DA matches the MAC Address0 content is routed. If the DDS bit of MAC_Extended_Configuration register is set: The XDCS field of MAC_DChSel_IndReg register contains the one-hot representation of one or more DMA Channel numbers to which an Rx packet whose DA matches the MAC Address1 content is routed.

(table continues...)

(continued)

Field	Bits	Type	Description
MBC	29:24	rw	**Mask Byte Control** These bits are mask control bits for comparing each of the MAC Address bytes. When set high, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address1 registers. Each bit controls the masking of the bytes as follows: - Bit 29: MAD Address High [15:8] - Bit 28: MAD Address High [7:0] - Bit 27: MAD Address Low[31:24] - ... - Bit 24: MAD Address Low[7:0] You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address.
SA	30	rw	**Source Address** - 1: The MAC Address1[47:0] is used to compare with the SA fields of the received packet. - 0: The MAC Address1[47:0] is used to compare with the DA fields of the received packet.
AE	31	rw	**Address Enable** - 1: The address filter module uses the 1th MAC address for perfect filtering. - 0: The address filter module ignores the address for filtering.
0	23:19	r	**Reserved** Read as 0; should be written with 0.

14.6.126 Port j MAC Address1 Low

The MAC_Address1_Low register holds the lower 32 bits of the 1th 6-byte MAC address of the station.

Portj_MAC_Address1_Low (j=0-1) Offset address: $1030C_H+j*2000_H$

Port j MAC Address1 Low Kernel Reset value: $FFFF\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADDRLO															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRLO															
rw															

Field	Bits	Type	Description
ADDRLO	31:0	rw	**MAC Address1 [31:0]** This field contains the lower 32 bits of 1th 6-byte MAC address. The MAC uses this field for filtering the received packets and inserting the MAC address in the Transmit Flow Control (Pause) Packets.

14.6.127 Port j MAC Address2 High

The MAC_Address2_High register holds the upper 16 bits of the 2th 6-byte MAC address of the station. The first DA byte that is received on the (X)GMII interface corresponds to the LS byte (Bits [7:0]) of the MAC_Address2_Low register. For example, if 0x112233445566 is received (0x11 in lane 0 of the first column) on the (X)GMII as the destination address, then the MAC_Address2 register [47:0] is compared with 0x665544332211.

If the MAC address registers are configured to be double-synchronized to the (X)GMII clock domain, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the MAC_Address2_Low register are written, so MAC_Address2_High register must be written prior to MAC_Address2_Low register. For proper synchronization updates, the consecutive writes to MAC_Address2_Low register must be performed after at least four clock cycles of the slowest clock between the source and the destination clock domains.

Portj_MAC_Address2_High (j=0-1) Offset address: $10310_H+j*2000_H$

Port j MAC Address2 High Kernel Reset value: $0000\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AE	SA	MBC						0					DCS		
rw	rw	rw						r					rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRHI															
rw															

Field	Bits	Type	Description
ADDRHI	15:0	rw	**MAC Address2 [47:32]** This field contains the upper 16 bits[47:32] of the 2th 6-byte MAC address.
DCS	18:16	rw	**DMA Channel Select** If the DDS bit of MAC_Extended_Configuration register is not set: This field contains the binary representation of the DMA Channel number to which an Rx packet whose DA matches the MAC Address0 content is routed. If the DDS bit of MAC_Extended_Configuration register is set: The XDCS field of MAC_DChSel_IndReg register contains the one-hot representation of one or more DMA Channel numbers to which an Rx packet whose DA matches the MAC Address2 content is routed.

(table continues...)

(continued)

Field	Bits	Type	Description
MBC	29:24	rw	**Mask Byte Control** These bits are mask control bits for comparing each of the MAC Address bytes. When set high, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address2 registers. Each bit controls the masking of the bytes as follows: - Bit 29: MAD Address High [15:8] - Bit 28: MAD Address High [7:0] - Bit 27: MAD Address Low[31:24] - ... - Bit 24: MAD Address Low[7:0] You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address.
SA	30	rw	**Source Address** - 1: The MAC Address2[47:0] is used to compare with the SA fields of the received packet. - 0: The MAC Address2[47:0] is used to compare with the DA fields of the received packet.
AE	31	rw	**Address Enable** - 1: The address filter module uses the 2th MAC address for perfect filtering. - 0: The address filter module ignores the address for filtering.
0	23:19	r	**Reserved** Read as 0; should be written with 0.

14.6.128 Port j MAC Address2 Low

The MAC_Address2_Low register holds the lower 32 bits of the 2th 6-byte MAC address of the station.

Portj_MAC_Address2_Low (j=0-1) Offset address: $10314_H+j*2000_H$

Port j MAC Address2 Low Kernel Reset value: FFFF FFFF$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADDRLO															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRLO															
rw															

Field	Bits	Type	Description
ADDRLO	31:0	rw	**MAC Address2 [31:0]** This field contains the lower 32 bits of 2th 6-byte MAC address. The MAC uses this field for filtering the received packets and inserting the MAC address in the Transmit Flow Control (Pause) Packets.

14.6.129 Port j MAC Address3 High

The MAC_Address3_High register holds the upper 16 bits of the 3th 6-byte MAC address of the station. The first DA byte that is received on the (X)GMII interface corresponds to the LS byte (Bits [7:0]) of the MAC_Address3_Low register. For example, if 0x112233445566 is received (0x11 in lane 0 of the first column) on the (X)GMII as the destination address, then the MAC_Address3 register [47:0] is compared with 0x665544332211.

If the MAC address registers are configured to be double-synchronized to the (X)GMII clock domain, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the MAC_Address3_Low register are written, so MAC_Address3_High register must be written prior to MAC_Address3_Low register. For proper synchronization updates, the consecutive writes to MAC_Address3_Low register must be performed after at least four clock cycles of the slowest clock between the source and the destination clock domains.

Portj_MAC_Address3_High (j=0-1) Offset address: $10318_H+j*2000_H$

Port j MAC Address3 High Kernel Reset value: $0000\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AE	SA	MBC						0					DCS		
rw	rw	rw						r					rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRHI															
rw															

Field	Bits	Type	Description
ADDRHI	15:0	rw	**MAC Address3 [47:32]** This field contains the upper 16 bits[47:32] of the 3th 6-byte MAC address.
DCS	18:16	rw	**DMA Channel Select** If the DDS bit of MAC_Extended_Configuration register is not set: This field contains the binary representation of the DMA Channel number to which an Rx packet whose DA matches the MAC Address0 content is routed. If the DDS bit of MAC_Extended_Configuration register is set: The XDCS field of MAC_DChSel_IndReg register contains the one-hot representation of one or more DMA Channel numbers to which an Rx packet whose DA matches the MAC Address3 content is routed.

(table continues...)

(continued)

Field	Bits	Type	Description
MBC	29:24	rw	**Mask Byte Control** These bits are mask control bits for comparing each of the MAC Address bytes. When set high, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address3 registers. Each bit controls the masking of the bytes as follows: - Bit 29: MAD Address High [15:8] - Bit 28: MAD Address High [7:0] - Bit 27: MAD Address Low[31:24] - ... - Bit 24: MAD Address Low[7:0] You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address.
SA	30	rw	**Source Address** - 1: The MAC Address3[47:0] is used to compare with the SA fields of the received packet. - 0: The MAC Address3[47:0] is used to compare with the DA fields of the received packet.
AE	31	rw	**Address Enable** - 1: The address filter module uses the 3th MAC address for perfect filtering. - 0: The address filter module ignores the address for filtering.
0	23:19	r	**Reserved** Read as 0; should be written with 0.

14.6.130 Port j MAC Address3 Low

The MAC_Address3_Low register holds the lower 32 bits of the 3th 6-byte MAC address of the station.

Portj_MAC_Address3_Low (j=0-1) Offset address: $1031C_H+j*2000_H$

Port j MAC Address3 Low Kernel Reset value: $FFFF\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADDRLO															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRLO															
rw															

Field	Bits	Type	Description
ADDRLO	31:0	rw	**MAC Address3 [31:0]** This field contains the lower 32 bits of 3th 6-byte MAC address. The MAC uses this field for filtering the received packets and inserting the MAC address in the Transmit Flow Control (Pause) Packets.

14.6.131 Port j MAC Address4 High

The MAC_Address4_High register holds the upper 16 bits of the 4th 6-byte MAC address of the station. The first DA byte that is received on the (X)GMII interface corresponds to the LS byte (Bits [7:0]) of the MAC_Address4_Low register. For example, if 0x112233445566 is received (0x11 in lane 0 of the first column) on the (X)GMII as the destination address, then the MAC_Address4 register [47:0] is compared with 0x665544332211.

If the MAC address registers are configured to be double-synchronized to the (X)GMII clock domain, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the MAC_Address4_Low register are written, so MAC_Address4_High register must be written prior to MAC_Address4_Low register. For proper synchronization updates, the consecutive writes to MAC_Address4_Low register must be performed after at least four clock cycles of the slowest clock between the source and the destination clock domains.

Portj_MAC_Address4_High (j=0-1) Offset address: $10320_H+j*2000_H$

Port j MAC Address4 High Kernel Reset value: $0000\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AE	SA	MBC						0					DCS		
rw	rw	rw						r					rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRHI															
rw															

Field	Bits	Type	Description
ADDRHI	15:0	rw	**MAC Address4 [47:32]** This field contains the upper 16 bits[47:32] of the 4th 6-byte MAC address.
DCS	18:16	rw	**DMA Channel Select** If the DDS bit of MAC_Extended_Configuration register is not set: This field contains the binary representation of the DMA Channel number to which an Rx packet whose DA matches the MAC Address0 content is routed. If the DDS bit of MAC_Extended_Configuration register is set: The XDCS field of MAC_DChSel_IndReg register contains the one-hot representation of one or more DMA Channel numbers to which an Rx packet whose DA matches the MAC Address4 content is routed.

(table continues...)

(continued)

Field	Bits	Type	Description
MBC	29:24	rw	**Mask Byte Control** These bits are mask control bits for comparing each of the MAC Address bytes. When set high, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address4 registers. Each bit controls the masking of the bytes as follows: - Bit 29: MAD Address High [15:8] - Bit 28: MAD Address High [7:0] - Bit 27: MAD Address Low[31:24] - ... - Bit 24: MAD Address Low[7:0] You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address.
SA	30	rw	**Source Address** - 1: The MAC Address4[47:0] is used to compare with the SA fields of the received packet. - 0: The MAC Address4[47:0] is used to compare with the DA fields of the received packet.
AE	31	rw	**Address Enable** - 1: The address filter module uses the 4th MAC address for perfect filtering. - 0: The address filter module ignores the address for filtering.
0	23:19	r	**Reserved** Read as 0; should be written with 0.

14.6.132 Port j MAC Address4 Low

The MAC_Address4_Low register holds the lower 32 bits of the 4th 6-byte MAC address of the station.

Portj_MAC_Address4_Low (j=0-1) Offset address: $10324_H+j*2000_H$

Port j MAC Address4 Low Kernel Reset value: FFFF FFFF$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADDRLO															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRLO															
rw															

Field	Bits	Type	Description
ADDRLO	31:0	rw	**MAC Address4 [31:0]** This field contains the lower 32 bits of 4th 6-byte MAC address. The MAC uses this field for filtering the received packets and inserting the MAC address in the Transmit Flow Control (Pause) Packets.

14.6.133 Port j MAC Address5 High

The MAC_Address5_High register holds the upper 16 bits of the 5th 6-byte MAC address of the station. The first DA byte that is received on the (X)GMII interface corresponds to the LS byte (Bits [7:0]) of the MAC_Address5_Low register. For example, if 0x112233445566 is received (0x11 in lane 0 of the first column) on the (X)GMII as the destination address, then the MAC_Address5 register [47:0] is compared with 0x665544332211.

If the MAC address registers are configured to be double-synchronized to the (X)GMII clock domain, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the MAC_Address5_Low register are written, so MAC_Address5_High register must be written prior to MAC_Address5_Low register. For proper synchronization updates, the consecutive writes to MAC_Address5_Low register must be performed after at least four clock cycles of the slowest clock between the source and the destination clock domains.

Portj_MAC_Address5_High (j=0-1) Offset address: $10328_H+j*2000_H$

Port j MAC Address5 High Kernel Reset value: $0000\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AE	SA	MBC						0					DCS		
rw	rw	rw						r					rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRHI															
rw															

Field	Bits	Type	Description
ADDRHI	15:0	rw	**MAC Address5 [47:32]** This field contains the upper 16 bits[47:32] of the 5th 6-byte MAC address.
DCS	18:16	rw	**DMA Channel Select** If the DDS bit of MAC_Extended_Configuration register is not set: This field contains the binary representation of the DMA Channel number to which an Rx packet whose DA matches the MAC Address0 content is routed. If the DDS bit of MAC_Extended_Configuration register is set: The XDCS field of MAC_DChSel_IndReg register contains the one-hot representation of one or more DMA Channel numbers to which an Rx packet whose DA matches the MAC Address5 content is routed.

(table continues...)

(continued)

Field	Bits	Type	Description
MBC	29:24	rw	**Mask Byte Control** These bits are mask control bits for comparing each of the MAC Address bytes. When set high, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address5 registers. Each bit controls the masking of the bytes as follows: - Bit 29: MAD Address High [15:8] - Bit 28: MAD Address High [7:0] - Bit 27: MAD Address Low[31:24] - ... - Bit 24: MAD Address Low[7:0] You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address.
SA	30	rw	**Source Address** - 1: The MAC Address5[47:0] is used to compare with the SA fields of the received packet. - 0: The MAC Address5[47:0] is used to compare with the DA fields of the received packet.
AE	31	rw	**Address Enable** - 1: The address filter module uses the 5th MAC address for perfect filtering. - 0: The address filter module ignores the address for filtering.
0	23:19	r	**Reserved** Read as 0; should be written with 0.

14.6.134 Port j MAC Address5 Low

The MAC_Address5_Low register holds the lower 32 bits of the 5th 6-byte MAC address of the station.

Portj_MAC_Address5_Low (j=0-1) Offset address: $1032C_H+j*2000_H$

Port j MAC Address5 Low Kernel Reset value: $FFFF\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADDRLO															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRLO															
rw															

Field	Bits	Type	Description
ADDRLO	31:0	rw	**MAC Address5 [31:0]** This field contains the lower 32 bits of 5th 6-byte MAC address. The MAC uses this field for filtering the received packets and inserting the MAC address in the Transmit Flow Control (Pause) Packets.

14.6.135 Port j MAC Address6 High

The MAC_Address6_High register holds the upper 16 bits of the 6th 6-byte MAC address of the station. The first DA byte that is received on the (X)GMII interface corresponds to the LS byte (Bits [7:0]) of the MAC_Address6_Low register. For example, if 0x112233445566 is received (0x11 in lane 0 of the first column) on the (X)GMII as the destination address, then the MAC_Address6 register [47:0] is compared with 0x665544332211.

If the MAC address registers are configured to be double-synchronized to the (X)GMII clock domain, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the MAC_Address6_Low register are written, so MAC_Address6_High register must be written prior to MAC_Address6_Low register. For proper synchronization updates, the consecutive writes to MAC_Address6_Low register must be performed after at least four clock cycles of the slowest clock between the source and the destination clock domains.

Portj_MAC_Address6_High (j=0-1) Offset address: $10330_H+j*2000_H$

Port j MAC Address6 High Kernel Reset value: $0000\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AE	SA	MBC						0					DCS		
rw	rw	rw						r					rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRHI															
rw															

Field	Bits	Type	Description
ADDRHI	15:0	rw	**MAC Address6 [47:32]** This field contains the upper 16 bits[47:32] of the 6th 6-byte MAC address.
DCS	18:16	rw	**DMA Channel Select** If the DDS bit of MAC_Extended_Configuration register is not set: This field contains the binary representation of the DMA Channel number to which an Rx packet whose DA matches the MAC Address0 content is routed. If the DDS bit of MAC_Extended_Configuration register is set: The XDCS field of MAC_DChSel_IndReg register contains the one-hot representation of one or more DMA Channel numbers to which an Rx packet whose DA matches the MAC Address6 content is routed.

(table continues...)

(continued)

Field	Bits	Type	Description
MBC	29:24	rw	**Mask Byte Control** These bits are mask control bits for comparing each of the MAC Address bytes. When set high, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address6 registers. Each bit controls the masking of the bytes as follows: - Bit 29: MAD Address High [15:8] - Bit 28: MAD Address High [7:0] - Bit 27: MAD Address Low[31:24] - ... - Bit 24: MAD Address Low[7:0] You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address.
SA	30	rw	**Source Address** - 1: The MAC Address6[47:0] is used to compare with the SA fields of the received packet. - 0: The MAC Address6[47:0] is used to compare with the DA fields of the received packet.
AE	31	rw	**Address Enable** - 1: The address filter module uses the 6th MAC address for perfect filtering. - 0: The address filter module ignores the address for filtering.
0	23:19	r	**Reserved** Read as 0; should be written with 0.

14.6.136 Port j MAC Address6 Low

The MAC_Address6_Low register holds the lower 32 bits of the 6th 6-byte MAC address of the station.

Portj_MAC_Address6_Low (j=0-1) Offset address: $10334_H+j*2000_H$

Port j MAC Address6 Low Kernel Reset value: FFFF FFFF$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADDRLO															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRLO															
rw															

Field	Bits	Type	Description
ADDRLO	31:0	rw	**MAC Address6 [31:0]** This field contains the lower 32 bits of 6th 6-byte MAC address. The MAC uses this field for filtering the received packets and inserting the MAC address in the Transmit Flow Control (Pause) Packets.

14.6.137 Port j MAC Address7 High

The MAC_Address7_High register holds the upper 16 bits of the 7th 6-byte MAC address of the station. The first DA byte that is received on the (X)GMII interface corresponds to the LS byte (Bits [7:0]) of the MAC_Address7_Low register. For example, if 0x112233445566 is received (0x11 in lane 0 of the first column) on the (X)GMII as the destination address, then the MAC_Address7 register [47:0] is compared with 0x665544332211.

If the MAC address registers are configured to be double-synchronized to the (X)GMII clock domain, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the MAC_Address7_Low register are written, so MAC_Address7_High register must be written prior to MAC_Address7_Low register. For proper synchronization updates, the consecutive writes to MAC_Address7_Low register must be performed after at least four clock cycles of the slowest clock between the source and the destination clock domains.

Portj_MAC_Address7_High (j=0-1) Offset address: $10338_H+j*2000_H$

Port j MAC Address7 High Kernel Reset value: $0000\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AE	SA	MBC						0					DCS		
rw	rw	rw						r					rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRHI															
rw															

Field	Bits	Type	Description
ADDRHI	15:0	rw	**MAC Address7 [47:32]** This field contains the upper 16 bits[47:32] of the 7th 6-byte MAC address.
DCS	18:16	rw	**DMA Channel Select** If the DDS bit of MAC_Extended_Configuration register is not set: This field contains the binary representation of the DMA Channel number to which an Rx packet whose DA matches the MAC Address0 content is routed. If the DDS bit of MAC_Extended_Configuration register is set: The XDCS field of MAC_DChSel_IndReg register contains the one-hot representation of one or more DMA Channel numbers to which an Rx packet whose DA matches the MAC Address7 content is routed.

(table continues...)

(continued)

Field	Bits	Type	Description
MBC	29:24	rw	**Mask Byte Control** These bits are mask control bits for comparing each of the MAC Address bytes. When set high, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address7 registers. Each bit controls the masking of the bytes as follows: - Bit 29: MAD Address High [15:8] - Bit 28: MAD Address High [7:0] - Bit 27: MAD Address Low[31:24] - ... - Bit 24: MAD Address Low[7:0] You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address.
SA	30	rw	**Source Address** - 1: The MAC Address7[47:0] is used to compare with the SA fields of the received packet. - 0: The MAC Address7[47:0] is used to compare with the DA fields of the received packet.
AE	31	rw	**Address Enable** - 1: The address filter module uses the 7th MAC address for perfect filtering. - 0: The address filter module ignores the address for filtering.
0	23:19	r	**Reserved** Read as 0; should be written with 0.

14.6.138 Port j MAC Address7 Low

The MAC_Address7_Low register holds the lower 32 bits of the 7th 6-byte MAC address of the station.

Portj_MAC_Address7_Low (j=0-1) Offset address: $1033C_H+j*2000_H$

Port j MAC Address7 Low Kernel Reset value: $FFFF\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADDRLO															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRLO															
rw															

Field	Bits	Type	Description
ADDRLO	31:0	rw	**MAC Address7 [31:0]** This field contains the lower 32 bits of 7th 6-byte MAC address. The MAC uses this field for filtering the received packets and inserting the MAC address in the Transmit Flow Control (Pause) Packets.

14.6.139 Port j MAC Address8 High

The MAC_Address8_High register holds the upper 16 bits of the 8th 6-byte MAC address of the station. The first DA byte that is received on the (X)GMII interface corresponds to the LS byte (Bits [7:0]) of the MAC_Address8_Low register. For example, if 0x112233445566 is received (0x11 in lane 0 of the first column) on the (X)GMII as the destination address, then the MAC_Address8 register [47:0] is compared with 0x665544332211.

If the MAC address registers are configured to be double-synchronized to the (X)GMII clock domain, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the MAC_Address8_Low register are written, so MAC_Address8_High register must be written prior to MAC_Address8_Low register. For proper synchronization updates, the consecutive writes to MAC_Address8_Low register must be performed after at least four clock cycles of the slowest clock between the source and the destination clock domains.

Portj_MAC_Address8_High (j=0-1) Offset address: $10340_H+j*2000_H$

Port j MAC Address8 High Kernel Reset value: $0000\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AE	SA	MBC						0					DCS		
rw	rw	rw						r					rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRHI															
rw															

Field	Bits	Type	Description
ADDRHI	15:0	rw	**MAC Address8 [47:32]** This field contains the upper 16 bits[47:32] of the 8th 6-byte MAC address.
DCS	18:16	rw	**DMA Channel Select** If the DDS bit of MAC_Extended_Configuration register is not set: This field contains the binary representation of the DMA Channel number to which an Rx packet whose DA matches the MAC Address0 content is routed. If the DDS bit of MAC_Extended_Configuration register is set: The XDCS field of MAC_DChSel_IndReg register contains the one-hot representation of one or more DMA Channel numbers to which an Rx packet whose DA matches the MAC Address8 content is routed.

(table continues...)

(continued)

Field	Bits	Type	Description
MBC	29:24	rw	**Mask Byte Control** These bits are mask control bits for comparing each of the MAC Address bytes. When set high, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address8 registers. Each bit controls the masking of the bytes as follows: - Bit 29: MAD Address High [15:8] - Bit 28: MAD Address High [7:0] - Bit 27: MAD Address Low[31:24] - ... - Bit 24: MAD Address Low[7:0] You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address.
SA	30	rw	**Source Address** - 1: The MAC Address8[47:0] is used to compare with the SA fields of the received packet. - 0: The MAC Address8[47:0] is used to compare with the DA fields of the received packet.
AE	31	rw	**Address Enable** - 1: The address filter module uses the 8th MAC address for perfect filtering. - 0: The address filter module ignores the address for filtering.
0	23:19	r	**Reserved** Read as 0; should be written with 0.

14.6.140 Port j MAC Address8 Low

The MAC_Address8_Low register holds the lower 32 bits of the 8th 6-byte MAC address of the station.

Portj_MAC_Address8_Low (j=0-1) Offset address: $10344_H+j*2000_H$

Port j MAC Address8 Low Kernel Reset value: FFFF FFFF$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADDRLO															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRLO															
rw															

Field	Bits	Type	Description
ADDRLO	31:0	rw	**MAC Address8 [31:0]** This field contains the lower 32 bits of 8th 6-byte MAC address. The MAC uses this field for filtering the received packets and inserting the MAC address in the Transmit Flow Control (Pause) Packets.

14.6.141 Port j MAC Address9 High

The MAC_Address9_High register holds the upper 16 bits of the 9th 6-byte MAC address of the station. The first DA byte that is received on the (X)GMII interface corresponds to the LS byte (Bits [7:0]) of the MAC_Address9_Low register. For example, if 0x112233445566 is received (0x11 in lane 0 of the first column) on the (X)GMII as the destination address, then the MAC_Address9 register [47:0] is compared with 0x665544332211.

If the MAC address registers are configured to be double-synchronized to the (X)GMII clock domain, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the MAC_Address9_Low register are written, so MAC_Address9_High register must be written prior to MAC_Address9_Low register. For proper synchronization updates, the consecutive writes to MAC_Address9_Low register must be performed after at least four clock cycles of the slowest clock between the source and the destination clock domains.

Portj_MAC_Address9_High (j=0-1) Offset address: $10348_H+j*2000_H$

Port j MAC Address9 High Kernel Reset value: $0000\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AE	SA	MBC						0					DCS		
rw	rw	rw						r					rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRHI															
rw															

Field	Bits	Type	Description
ADDRHI	15:0	rw	**MAC Address9 [47:32]** This field contains the upper 16 bits[47:32] of the 9th 6-byte MAC address.
DCS	18:16	rw	**DMA Channel Select** If the DDS bit of MAC_Extended_Configuration register is not set: This field contains the binary representation of the DMA Channel number to which an Rx packet whose DA matches the MAC Address0 content is routed. If the DDS bit of MAC_Extended_Configuration register is set: The XDCS field of MAC_DChSel_IndReg register contains the one-hot representation of one or more DMA Channel numbers to which an Rx packet whose DA matches the MAC Address9 content is routed.

(table continues...)

(continued)

Field	Bits	Type	Description
MBC	29:24	rw	**Mask Byte Control** These bits are mask control bits for comparing each of the MAC Address bytes. When set high, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address9 registers. Each bit controls the masking of the bytes as follows: - Bit 29: MAD Address High [15:8] - Bit 28: MAD Address High [7:0] - Bit 27: MAD Address Low[31:24] - ... - Bit 24: MAD Address Low[7:0] You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address.
SA	30	rw	**Source Address** - 1: The MAC Address9[47:0] is used to compare with the SA fields of the received packet. - 0: The MAC Address9[47:0] is used to compare with the DA fields of the received packet.
AE	31	rw	**Address Enable** - 1: The address filter module uses the 9th MAC address for perfect filtering. - 0: The address filter module ignores the address for filtering.
0	23:19	r	**Reserved** Read as 0; should be written with 0.

14.6.142 Port j MAC Address9 Low

The MAC_Address9_Low register holds the lower 32 bits of the 9th 6-byte MAC address of the station.

Portj_MAC_Address9_Low (j=0-1) Offset address: $1034C_H+j*2000_H$

Port j MAC Address9 Low Kernel Reset value: $FFFF\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADDRLO															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRLO															
rw															

Field	Bits	Type	Description
ADDRLO	31:0	rw	**MAC Address9 [31:0]** This field contains the lower 32 bits of 9th 6-byte MAC address. The MAC uses this field for filtering the received packets and inserting the MAC address in the Transmit Flow Control (Pause) Packets.

14.6.143 Port j MAC Address10 High

The MAC_Address10_High register holds the upper 16 bits of the 10th 6-byte MAC address of the station. The first DA byte that is received on the (X)GMII interface corresponds to the LS byte (Bits [7:0]) of the MAC_Address10_Low register. For example, if 0x112233445566 is received (0x11 in lane 0 of the first column) on the (X)GMII as the destination address, then the MAC_Address10 register [47:0] is compared with 0x665544332211.

If the MAC address registers are configured to be double-synchronized to the (X)GMII clock domain, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the MAC_Address10_Low register are written, so MAC_Address10_High register must be written prior to MAC_Address10_Low register. For proper synchronization updates, the consecutive writes to MAC_Address10_Low register must be performed after at least four clock cycles of the slowest clock between the source and the destination clock domains.

Portj_MAC_Address10_High (j=0-1) Offset address: $10350_H+j*2000_H$

Port j MAC Address10 High Kernel Reset value: $0000\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AE	SA	MBC						0					DCS		
rw	rw	rw						r					rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRHI															
rw															

Field	Bits	Type	Description
ADDRHI	15:0	rw	**MAC Address10 [47:32]** This field contains the upper 16 bits[47:32] of the 10th 6-byte MAC address.
DCS	18:16	rw	**DMA Channel Select** If the DDS bit of MAC_Extended_Configuration register is not set: This field contains the binary representation of the DMA Channel number to which an Rx packet whose DA matches the MAC Address0 content is routed. If the DDS bit of MAC_Extended_Configuration register is set: The XDCS field of MAC_DChSel_IndReg register contains the one-hot representation of one or more DMA Channel numbers to which an Rx packet whose DA matches the MAC Address10 content is routed.

(table continues...)

(continued)

Field	Bits	Type	Description
MBC	29:24	rw	**Mask Byte Control** These bits are mask control bits for comparing each of the MAC Address bytes. When set high, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address10 registers. Each bit controls the masking of the bytes as follows: - Bit 29: MAD Address High [15:8] - Bit 28: MAD Address High [7:0] - Bit 27: MAD Address Low[31:24] - ... - Bit 24: MAD Address Low[7:0] You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address.
SA	30	rw	**Source Address** - 1: The MAC Address10[47:0] is used to compare with the SA fields of the received packet. - 0: The MAC Address10[47:0] is used to compare with the DA fields of the received packet.
AE	31	rw	**Address Enable** - 1: The address filter module uses the 10th MAC address for perfect filtering. - 0: The address filter module ignores the address for filtering.
0	23:19	r	**Reserved** Read as 0; should be written with 0.

14.6.144 Port j MAC Address10 Low

The MAC_Address10_Low register holds the lower 32 bits of the 10th 6-byte MAC address of the station.

Portj_MAC_Address10_Low (j=0-1) Offset address: $10354_H+j*2000_H$

Port j MAC Address10 Low Kernel Reset value: FFFF FFFF$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADDRLO															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRLO															
rw															

Field	Bits	Type	Description
ADDRLO	31:0	rw	**MAC Address10 [31:0]** This field contains the lower 32 bits of 10th 6-byte MAC address. The MAC uses this field for filtering the received packets and inserting the MAC address in the Transmit Flow Control (Pause) Packets.

14.6.145 Port j MAC Address11 High

The MAC_Address11_High register holds the upper 16 bits of the 11th 6-byte MAC address of the station. The first DA byte that is received on the (X)GMII interface corresponds to the LS byte (Bits [7:0]) of the MAC_Address11_Low register. For example, if 0x112233445566 is received (0x11 in lane 0 of the first column) on the (X)GMII as the destination address, then the MAC_Address11 register [47:0] is compared with 0x665544332211.

If the MAC address registers are configured to be double-synchronized to the (X)GMII clock domain, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the MAC_Address11_Low register are written, so MAC_Address11_High register must be written prior to MAC_Address11_Low register. For proper synchronization updates, the consecutive writes to MAC_Address11_Low register must be performed after at least four clock cycles of the slowest clock between the source and the destination clock domains.

Portj_MAC_Address11_High (j=0-1) Offset address: $10358_H+j*2000_H$

Port j MAC Address11 High Kernel Reset value: $0000\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AE	SA	MBC						0					DCS		
rw	rw	rw						r					rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRHI															
rw															

Field	Bits	Type	Description
ADDRHI	15:0	rw	**MAC Address11 [47:32]** This field contains the upper 16 bits[47:32] of the 11th 6-byte MAC address.
DCS	18:16	rw	**DMA Channel Select** If the DDS bit of MAC_Extended_Configuration register is not set: This field contains the binary representation of the DMA Channel number to which an Rx packet whose DA matches the MAC Address0 content is routed. If the DDS bit of MAC_Extended_Configuration register is set: The XDCS field of MAC_DChSel_IndReg register contains the one-hot representation of one or more DMA Channel numbers to which an Rx packet whose DA matches the MAC Address11 content is routed.

(table continues...)

(continued)

Field	Bits	Type	Description
MBC	29:24	rw	**Mask Byte Control** These bits are mask control bits for comparing each of the MAC Address bytes. When set high, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address11 registers. Each bit controls the masking of the bytes as follows: - Bit 29: MAD Address High [15:8] - Bit 28: MAD Address High [7:0] - Bit 27: MAD Address Low[31:24] - ... - Bit 24: MAD Address Low[7:0] You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address.
SA	30	rw	**Source Address** - 1: The MAC Address11[47:0] is used to compare with the SA fields of the received packet. - 0: The MAC Address11[47:0] is used to compare with the DA fields of the received packet.
AE	31	rw	**Address Enable** - 1: The address filter module uses the 11th MAC address for perfect filtering. - 0: The address filter module ignores the address for filtering.
0	23:19	r	**Reserved** Read as 0; should be written with 0.

14.6.146 Port j MAC Address11 Low

The MAC_Address11_Low register holds the lower 32 bits of the 11th 6-byte MAC address of the station.

Portj_MAC_Address11_Low (j=0-1) Offset address: $1035C_H+j*2000_H$

Port j MAC Address11 Low Kernel Reset value: $FFFF\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADDRLO															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRLO															
rw															

Field	Bits	Type	Description
ADDRLO	31:0	rw	**MAC Address11 [31:0]** This field contains the lower 32 bits of 11th 6-byte MAC address. The MAC uses this field for filtering the received packets and inserting the MAC address in the Transmit Flow Control (Pause) Packets.

14.6.147 Port j MAC Address12 High

The MAC_Address12_High register holds the upper 16 bits of the 12th 6-byte MAC address of the station. The first DA byte that is received on the (X)GMII interface corresponds to the LS byte (Bits [7:0]) of the MAC_Address12_Low register. For example, if 0x112233445566 is received (0x11 in lane 0 of the first column) on the (X)GMII as the destination address, then the MAC_Address12 register [47:0] is compared with 0x665544332211.

If the MAC address registers are configured to be double-synchronized to the (X)GMII clock domain, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the MAC_Address12_Low register are written, so MAC_Address12_High register must be written prior to MAC_Address12_Low register. For proper synchronization updates, the consecutive writes to MAC_Address12_Low register must be performed after at least four clock cycles of the slowest clock between the source and the destination clock domains.

Portj_MAC_Address12_High (j=0-1) Offset address: $10360_H+j*2000_H$

Port j MAC Address12 High Kernel Reset value: $0000\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AE	SA	MBC						0					DCS		
rw	rw	rw						r					rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRHI															
rw															

Field	Bits	Type	Description
ADDRHI	15:0	rw	**MAC Address12 [47:32]** This field contains the upper 16 bits[47:32] of the 12th 6-byte MAC address.
DCS	18:16	rw	**DMA Channel Select** If the DDS bit of MAC_Extended_Configuration register is not set: This field contains the binary representation of the DMA Channel number to which an Rx packet whose DA matches the MAC Address0 content is routed. If the DDS bit of MAC_Extended_Configuration register is set: The XDCS field of MAC_DChSel_IndReg register contains the one-hot representation of one or more DMA Channel numbers to which an Rx packet whose DA matches the MAC Address12 content is routed.

(table continues...)

(continued)

Field	Bits	Type	Description
MBC	29:24	rw	**Mask Byte Control** These bits are mask control bits for comparing each of the MAC Address bytes. When set high, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address12 registers. Each bit controls the masking of the bytes as follows: - Bit 29: MAD Address High [15:8] - Bit 28: MAD Address High [7:0] - Bit 27: MAD Address Low[31:24] - ... - Bit 24: MAD Address Low[7:0] You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address.
SA	30	rw	**Source Address** - 1: The MAC Address12[47:0] is used to compare with the SA fields of the received packet. - 0: The MAC Address12[47:0] is used to compare with the DA fields of the received packet.
AE	31	rw	**Address Enable** - 1: The address filter module uses the 12th MAC address for perfect filtering. - 0: The address filter module ignores the address for filtering.
0	23:19	r	**Reserved** Read as 0; should be written with 0.

14.6.148 Port j MAC Address12 Low

The MAC_Address12_Low register holds the lower 32 bits of the 12th 6-byte MAC address of the station.

Portj_MAC_Address12_Low (j=0-1) Offset address: $10364_H+j*2000_H$

Port j MAC Address12 Low Kernel Reset value: FFFF FFFF$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADDRLO															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRLO															
rw															

Field	Bits	Type	Description
ADDRLO	31:0	rw	**MAC Address12 [31:0]** This field contains the lower 32 bits of 12th 6-byte MAC address. The MAC uses this field for filtering the received packets and inserting the MAC address in the Transmit Flow Control (Pause) Packets.

14.6.149 Port j MAC Address13 High

The MAC_Address13_High register holds the upper 16 bits of the 13th 6-byte MAC address of the station. The first DA byte that is received on the (X)GMII interface corresponds to the LS byte (Bits [7:0]) of the MAC_Address13_Low register. For example, if 0x112233445566 is received (0x11 in lane 0 of the first column) on the (X)GMII as the destination address, then the MAC_Address13 register [47:0] is compared with 0x665544332211.

If the MAC address registers are configured to be double-synchronized to the (X)GMII clock domain, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the MAC_Address13_Low register are written, so MAC_Address13_High register must be written prior to MAC_Address13_Low register. For proper synchronization updates, the consecutive writes to MAC_Address13_Low register must be performed after at least four clock cycles of the slowest clock between the source and the destination clock domains.

Portj_MAC_Address13_High (j=0-1) Offset address: $10368_H+j*2000_H$

Port j MAC Address13 High Kernel Reset value: 0000 $FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AE	SA	MBC						0					DCS		
rw	rw	rw						r					rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRHI															
rw															

Field	Bits	Type	Description
ADDRHI	15:0	rw	**MAC Address13 [47:32]** This field contains the upper 16 bits[47:32] of the 13th 6-byte MAC address.
DCS	18:16	rw	**DMA Channel Select** If the DDS bit of MAC_Extended_Configuration register is not set: This field contains the binary representation of the DMA Channel number to which an Rx packet whose DA matches the MAC Address0 content is routed. If the DDS bit of MAC_Extended_Configuration register is set: The XDCS field of MAC_DChSel_IndReg register contains the one-hot representation of one or more DMA Channel numbers to which an Rx packet whose DA matches the MAC Address13 content is routed.

(table continues...)

(continued)

Field	Bits	Type	Description
MBC	29:24	rw	**Mask Byte Control** These bits are mask control bits for comparing each of the MAC Address bytes. When set high, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address13 registers. Each bit controls the masking of the bytes as follows: - Bit 29: MAD Address High [15:8] - Bit 28: MAD Address High [7:0] - Bit 27: MAD Address Low[31:24] - ... - Bit 24: MAD Address Low[7:0] You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address.
SA	30	rw	**Source Address** - 1: The MAC Address13[47:0] is used to compare with the SA fields of the received packet. - 0: The MAC Address13[47:0] is used to compare with the DA fields of the received packet.
AE	31	rw	**Address Enable** - 1: The address filter module uses the 13th MAC address for perfect filtering. - 0: The address filter module ignores the address for filtering.
0	23:19	r	**Reserved** Read as 0; should be written with 0.

14.6.150 Port j MAC Address13 Low

The MAC_Address13_Low register holds the lower 32 bits of the 13th 6-byte MAC address of the station.

Portj_MAC_Address13_Low (j=0-1) Offset address: $1036C_H+j*2000_H$

Port j MAC Address13 Low Kernel Reset value: $FFFF\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADDRLO															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRLO															
rw															

Field	Bits	Type	Description
ADDRLO	31:0	rw	**MAC Address13 [31:0]** This field contains the lower 32 bits of 13th 6-byte MAC address. The MAC uses this field for filtering the received packets and inserting the MAC address in the Transmit Flow Control (Pause) Packets.

14.6.151 Port j MAC Address14 High

The MAC_Address14_High register holds the upper 16 bits of the 14th 6-byte MAC address of the station. The first DA byte that is received on the (X)GMII interface corresponds to the LS byte (Bits [7:0]) of the MAC_Address14_Low register. For example, if 0x112233445566 is received (0x11 in lane 0 of the first column) on the (X)GMII as the destination address, then the MAC_Address14 register [47:0] is compared with 0x665544332211.

If the MAC address registers are configured to be double-synchronized to the (X)GMII clock domain, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the MAC_Address14_Low register are written, so MAC_Address14_High register must be written prior to MAC_Address14_Low register. For proper synchronization updates, the consecutive writes to MAC_Address14_Low register must be performed after at least four clock cycles of the slowest clock between the source and the destination clock domains.

Portj_MAC_Address14_High (j=0-1) Offset address: $10370_H+j*2000_H$

Port j MAC Address14 High Kernel Reset value: $0000\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AE	SA	MBC						0					DCS		
rw	rw	rw						r					rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRHI															
rw															

Field	Bits	Type	Description
ADDRHI	15:0	rw	**MAC Address14 [47:32]** This field contains the upper 16 bits[47:32] of the 14th 6-byte MAC address.
DCS	18:16	rw	**DMA Channel Select** If the DDS bit of MAC_Extended_Configuration register is not set: This field contains the binary representation of the DMA Channel number to which an Rx packet whose DA matches the MAC Address0 content is routed. If the DDS bit of MAC_Extended_Configuration register is set: The XDCS field of MAC_DChSel_IndReg register contains the one-hot representation of one or more DMA Channel numbers to which an Rx packet whose DA matches the MAC Address14 content is routed.

(table continues...)

(continued)

Field	Bits	Type	Description
MBC	29:24	rw	**Mask Byte Control** These bits are mask control bits for comparing each of the MAC Address bytes. When set high, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address14 registers. Each bit controls the masking of the bytes as follows: - Bit 29: MAD Address High [15:8] - Bit 28: MAD Address High [7:0] - Bit 27: MAD Address Low[31:24] - ... - Bit 24: MAD Address Low[7:0] You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address.
SA	30	rw	**Source Address** - 1: The MAC Address14[47:0] is used to compare with the SA fields of the received packet. - 0: The MAC Address14[47:0] is used to compare with the DA fields of the received packet.
AE	31	rw	**Address Enable** - 1: The address filter module uses the 14th MAC address for perfect filtering. - 0: The address filter module ignores the address for filtering.
0	23:19	r	**Reserved** Read as 0; should be written with 0.

14.6.152 Port j MAC Address14 Low

The MAC_Address14_Low register holds the lower 32 bits of the 14th 6-byte MAC address of the station.

Portj_MAC_Address14_Low (j=0-1) Offset address: $10374_H+j*2000_H$

Port j MAC Address14 Low Kernel Reset value: FFFF FFFF$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADDRLO															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRLO															
rw															

Field	Bits	Type	Description
ADDRLO	31:0	rw	**MAC Address14 [31:0]** This field contains the lower 32 bits of 14th 6-byte MAC address. The MAC uses this field for filtering the received packets and inserting the MAC address in the Transmit Flow Control (Pause) Packets.

14.6.153 Port j MAC Address15 High

The MAC_Address15_High register holds the upper 16 bits of the 15th 6-byte MAC address of the station. The first DA byte that is received on the (X)GMII interface corresponds to the LS byte (Bits [7:0]) of the MAC_Address15_Low register. For example, if 0x112233445566 is received (0x11 in lane 0 of the first column) on the (X)GMII as the destination address, then the MAC_Address15 register [47:0] is compared with 0x665544332211.

If the MAC address registers are configured to be double-synchronized to the (X)GMII clock domain, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the MAC_Address15_Low register are written, so MAC_Address15_High register must be written prior to MAC_Address15_Low register. For proper synchronization updates, the consecutive writes to MAC_Address15_Low register must be performed after at least four clock cycles of the slowest clock between the source and the destination clock domains.

Portj_MAC_Address15_High (j=0-1) Offset address: $10378_H+j*2000_H$

Port j MAC Address15 High Kernel Reset value: $0000\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AE	SA	MBC						0					DCS		
rw	rw	rw						r					rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRHI															
rw															

Field	Bits	Type	Description
ADDRHI	15:0	rw	**MAC Address15 [47:32]** This field contains the upper 16 bits[47:32] of the 15th 6-byte MAC address.
DCS	18:16	rw	**DMA Channel Select** If the DDS bit of MAC_Extended_Configuration register is not set: This field contains the binary representation of the DMA Channel number to which an Rx packet whose DA matches the MAC Address0 content is routed. If the DDS bit of MAC_Extended_Configuration register is set: The XDCS field of MAC_DChSel_IndReg register contains the one-hot representation of one or more DMA Channel numbers to which an Rx packet whose DA matches the MAC Address15 content is routed.

(table continues...)

(continued)

Field	Bits	Type	Description
MBC	29:24	rw	**Mask Byte Control** These bits are mask control bits for comparing each of the MAC Address bytes. When set high, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address15 registers. Each bit controls the masking of the bytes as follows: - Bit 29: MAD Address High [15:8] - Bit 28: MAD Address High [7:0] - Bit 27: MAD Address Low[31:24] - ... - Bit 24: MAD Address Low[7:0] You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address.
SA	30	rw	**Source Address** - 1: The MAC Address15[47:0] is used to compare with the SA fields of the received packet. - 0: The MAC Address15[47:0] is used to compare with the DA fields of the received packet.
AE	31	rw	**Address Enable** - 1: The address filter module uses the 15th MAC address for perfect filtering. - 0: The address filter module ignores the address for filtering.
0	23:19	r	**Reserved** Read as 0; should be written with 0.

14.6.154 Port j MAC Address15 Low

The MAC_Address15_Low register holds the lower 32 bits of the 15th 6-byte MAC address of the station.

Portj_MAC_Address15_Low (j=0-1) Offset address: $1037C_H+j*2000_H$

Port j MAC Address15 Low Kernel Reset value: $FFFF\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADDRLO															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRLO															
rw															

Field	Bits	Type	Description
ADDRLO	31:0	rw	**MAC Address15 [31:0]** This field contains the lower 32 bits of 15th 6-byte MAC address. The MAC uses this field for filtering the received packets and inserting the MAC address in the Transmit Flow Control (Pause) Packets.

14.6.155 Port j MAC Address16 High

The MAC_Address16_High register holds the upper 16 bits of the 16th 6-byte MAC address of the station. The first DA byte that is received on the (X)GMII interface corresponds to the LS byte (Bits [7:0]) of the MAC_Address16_Low register. For example, if 0x112233445566 is received (0x11 in lane 0 of the first column) on the (X)GMII as the destination address, then the MAC_Address16 register [47:0] is compared with 0x665544332211.

If the MAC address registers are configured to be double-synchronized to the (X)GMII clock domain, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the MAC_Address16_Low register are written, so MAC_Address16_High register must be written prior to MAC_Address16_Low register. For proper synchronization updates, the consecutive writes to MAC_Address16_Low register must be performed after at least four clock cycles of the slowest clock between the source and the destination clock domains.

Portj_MAC_Address16_High (j=0-1) Offset address: $10380_H+j*2000_H$

Port j MAC Address16 High Kernel Reset value: $0000\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AE	SA	MBC						0					DCS		
rw	rw	rw						r					rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRHI															
rw															

Field	Bits	Type	Description
ADDRHI	15:0	rw	**MAC Address16 [47:32]** This field contains the upper 16 bits[47:32] of the 16th 6-byte MAC address.
DCS	18:16	rw	**DMA Channel Select** If the DDS bit of MAC_Extended_Configuration register is not set: This field contains the binary representation of the DMA Channel number to which an Rx packet whose DA matches the MAC Address0 content is routed. If the DDS bit of MAC_Extended_Configuration register is set: The XDCS field of MAC_DChSel_IndReg register contains the one-hot representation of one or more DMA Channel numbers to which an Rx packet whose DA matches the MAC Address16 content is routed.

(table continues...)

(continued)

Field	Bits	Type	Description
MBC	29:24	rw	**Mask Byte Control** These bits are mask control bits for comparing each of the MAC Address bytes. When set high, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address16 registers. Each bit controls the masking of the bytes as follows: - Bit 29: MAD Address High [15:8] - Bit 28: MAD Address High [7:0] - Bit 27: MAD Address Low[31:24] - ... - Bit 24: MAD Address Low[7:0] You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address.
SA	30	rw	**Source Address** - 1: The MAC Address16[47:0] is used to compare with the SA fields of the received packet. - 0: The MAC Address16[47:0] is used to compare with the DA fields of the received packet.
AE	31	rw	**Address Enable** - 1: The address filter module uses the 16th MAC address for perfect filtering. - 0: The address filter module ignores the address for filtering.
0	23:19	r	**Reserved** Read as 0; should be written with 0.

14.6.156 Port j MAC Address16 Low

The MAC_Address16_Low register holds the lower 32 bits of the 16th 6-byte MAC address of the station.

Portj_MAC_Address16_Low (j=0-1) Offset address: $10384_H+j*2000_H$

Port j MAC Address16 Low Kernel Reset value: FFFF FFFF$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADDRLO															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRLO															
rw															

Field	Bits	Type	Description
ADDRLO	31:0	rw	**MAC Address16 [31:0]** This field contains the lower 32 bits of 16th 6-byte MAC address. The MAC uses this field for filtering the received packets and inserting the MAC address in the Transmit Flow Control (Pause) Packets.

14.6.157 Port j MAC Address17 High

The MAC_Address17_High register holds the upper 16 bits of the 17th 6-byte MAC address of the station. The first DA byte that is received on the (X)GMII interface corresponds to the LS byte (Bits [7:0]) of the MAC_Address17_Low register. For example, if 0x112233445566 is received (0x11 in lane 0 of the first column) on the (X)GMII as the destination address, then the MAC_Address17 register [47:0] is compared with 0x665544332211.

If the MAC address registers are configured to be double-synchronized to the (X)GMII clock domain, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the MAC_Address17_Low register are written, so MAC_Address17_High register must be written prior to MAC_Address17_Low register. For proper synchronization updates, the consecutive writes to MAC_Address17_Low register must be performed after at least four clock cycles of the slowest clock between the source and the destination clock domains.

Portj_MAC_Address17_High (j=0-1) Offset address: $10388_H+j*2000_H$

Port j MAC Address17 High Kernel Reset value: $0000\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AE	SA	MBC						0					DCS		
rw	rw	rw						r					rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRHI															
rw															

Field	Bits	Type	Description
ADDRHI	15:0	rw	**MAC Address17 [47:32]** This field contains the upper 16 bits[47:32] of the 17th 6-byte MAC address.
DCS	18:16	rw	**DMA Channel Select** If the DDS bit of MAC_Extended_Configuration register is not set: This field contains the binary representation of the DMA Channel number to which an Rx packet whose DA matches the MAC Address0 content is routed. If the DDS bit of MAC_Extended_Configuration register is set: The XDCS field of MAC_DChSel_IndReg register contains the one-hot representation of one or more DMA Channel numbers to which an Rx packet whose DA matches the MAC Address17 content is routed.

(table continues...)

(continued)

Field	Bits	Type	Description
MBC	29:24	rw	**Mask Byte Control** These bits are mask control bits for comparing each of the MAC Address bytes. When set high, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address17 registers. Each bit controls the masking of the bytes as follows: - Bit 29: MAD Address High [15:8] - Bit 28: MAD Address High [7:0] - Bit 27: MAD Address Low[31:24] - ... - Bit 24: MAD Address Low[7:0] You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address.
SA	30	rw	**Source Address** - 1: The MAC Address17[47:0] is used to compare with the SA fields of the received packet. - 0: The MAC Address17[47:0] is used to compare with the DA fields of the received packet.
AE	31	rw	**Address Enable** - 1: The address filter module uses the 17th MAC address for perfect filtering. - 0: The address filter module ignores the address for filtering.
0	23:19	r	**Reserved** Read as 0; should be written with 0.

14.6.158 Port j MAC Address17 Low

The MAC_Address17_Low register holds the lower 32 bits of the 17th 6-byte MAC address of the station.

Portj_MAC_Address17_Low (j=0-1) Offset address: $1038C_H+j*2000_H$

Port j MAC Address17 Low Kernel Reset value: $FFFF\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADDRLO															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRLO															
rw															

Field	Bits	Type	Description
ADDRLO	31:0	rw	**MAC Address17 [31:0]** This field contains the lower 32 bits of 17th 6-byte MAC address. The MAC uses this field for filtering the received packets and inserting the MAC address in the Transmit Flow Control (Pause) Packets.

14.6.159 Port j MAC Address18 High

The MAC_Address18_High register holds the upper 16 bits of the 18th 6-byte MAC address of the station. The first DA byte that is received on the (X)GMII interface corresponds to the LS byte (Bits [7:0]) of the MAC_Address18_Low register. For example, if 0x112233445566 is received (0x11 in lane 0 of the first column) on the (X)GMII as the destination address, then the MAC_Address18 register [47:0] is compared with 0x665544332211.

If the MAC address registers are configured to be double-synchronized to the (X)GMII clock domain, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the MAC_Address18_Low register are written, so MAC_Address18_High register must be written prior to MAC_Address18_Low register. For proper synchronization updates, the consecutive writes to MAC_Address18_Low register must be performed after at least four clock cycles of the slowest clock between the source and the destination clock domains.

Portj_MAC_Address18_High (j=0-1) Offset address: $10390_H+j*2000_H$

Port j MAC Address18 High Kernel Reset value: $0000\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AE	SA	MBC						0					DCS		
rw	rw	rw						r					rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRHI															
rw															

Field	Bits	Type	Description
ADDRHI	15:0	rw	**MAC Address18 [47:32]** This field contains the upper 16 bits[47:32] of the 18th 6-byte MAC address.
DCS	18:16	rw	**DMA Channel Select** If the DDS bit of MAC_Extended_Configuration register is not set: This field contains the binary representation of the DMA Channel number to which an Rx packet whose DA matches the MAC Address0 content is routed. If the DDS bit of MAC_Extended_Configuration register is set: The XDCS field of MAC_DChSel_IndReg register contains the one-hot representation of one or more DMA Channel numbers to which an Rx packet whose DA matches the MAC Address18 content is routed.

(table continues...)

(continued)

Field	Bits	Type	Description
MBC	29:24	rw	**Mask Byte Control** These bits are mask control bits for comparing each of the MAC Address bytes. When set high, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address18 registers. Each bit controls the masking of the bytes as follows: - Bit 29: MAD Address High [15:8] - Bit 28: MAD Address High [7:0] - Bit 27: MAD Address Low[31:24] - ... - Bit 24: MAD Address Low[7:0] You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address.
SA	30	rw	**Source Address** - 1: The MAC Address18[47:0] is used to compare with the SA fields of the received packet. - 0: The MAC Address18[47:0] is used to compare with the DA fields of the received packet.
AE	31	rw	**Address Enable** - 1: The address filter module uses the 18th MAC address for perfect filtering. - 0: The address filter module ignores the address for filtering.
0	23:19	r	**Reserved** Read as 0; should be written with 0.

14.6.160 Port j MAC Address18 Low

The MAC_Address18_Low register holds the lower 32 bits of the 18th 6-byte MAC address of the station.

Portj_MAC_Address18_Low (j=0-1) Offset address: $10394_H+j*2000_H$

Port j MAC Address18 Low Kernel Reset value: FFFF FFFF$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADDRLO															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRLO															
rw															

Field	Bits	Type	Description
ADDRLO	31:0	rw	**MAC Address18 [31:0]** This field contains the lower 32 bits of 18th 6-byte MAC address. The MAC uses this field for filtering the received packets and inserting the MAC address in the Transmit Flow Control (Pause) Packets.

14.6.161 Port j MAC Address19 High

The MAC_Address19_High register holds the upper 16 bits of the 19th 6-byte MAC address of the station. The first DA byte that is received on the (X)GMII interface corresponds to the LS byte (Bits [7:0]) of the MAC_Address19_Low register. For example, if 0x112233445566 is received (0x11 in lane 0 of the first column) on the (X)GMII as the destination address, then the MAC_Address19 register [47:0] is compared with 0x665544332211.

If the MAC address registers are configured to be double-synchronized to the (X)GMII clock domain, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the MAC_Address19_Low register are written, so MAC_Address19_High register must be written prior to MAC_Address19_Low register. For proper synchronization updates, the consecutive writes to MAC_Address19_Low register must be performed after at least four clock cycles of the slowest clock between the source and the destination clock domains.

Portj_MAC_Address19_High (j=0-1) Offset address: $10398_H+j*2000_H$

Port j MAC Address19 High Kernel Reset value: $0000\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AE	SA	MBC						0					DCS		
rw	rw	rw						r					rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRHI															
rw															

Field	Bits	Type	Description
ADDRHI	15:0	rw	**MAC Address19 [47:32]** This field contains the upper 16 bits[47:32] of the 19th 6-byte MAC address.
DCS	18:16	rw	**DMA Channel Select** If the DDS bit of MAC_Extended_Configuration register is not set: This field contains the binary representation of the DMA Channel number to which an Rx packet whose DA matches the MAC Address0 content is routed. If the DDS bit of MAC_Extended_Configuration register is set: The XDCS field of MAC_DChSel_IndReg register contains the one-hot representation of one or more DMA Channel numbers to which an Rx packet whose DA matches the MAC Address19 content is routed.

(table continues...)

(continued)

Field	Bits	Type	Description
MBC	29:24	rw	**Mask Byte Control** These bits are mask control bits for comparing each of the MAC Address bytes. When set high, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address19 registers. Each bit controls the masking of the bytes as follows: - Bit 29: MAD Address High [15:8] - Bit 28: MAD Address High [7:0] - Bit 27: MAD Address Low[31:24] - ... - Bit 24: MAD Address Low[7:0] You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address.
SA	30	rw	**Source Address** - 1: The MAC Address19[47:0] is used to compare with the SA fields of the received packet. - 0: The MAC Address19[47:0] is used to compare with the DA fields of the received packet.
AE	31	rw	**Address Enable** - 1: The address filter module uses the 19th MAC address for perfect filtering. - 0: The address filter module ignores the address for filtering.
0	23:19	r	**Reserved** Read as 0; should be written with 0.

14.6.162 Port j MAC Address19 Low

The MAC_Address19_Low register holds the lower 32 bits of the 19th 6-byte MAC address of the station.

Portj_MAC_Address19_Low (j=0-1) Offset address: $1039C_H+j*2000_H$

Port j MAC Address19 Low Kernel Reset value: $FFFF\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADDRLO															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRLO															
rw															

Field	Bits	Type	Description
ADDRLO	31:0	rw	**MAC Address19 [31:0]** This field contains the lower 32 bits of 19th 6-byte MAC address. The MAC uses this field for filtering the received packets and inserting the MAC address in the Transmit Flow Control (Pause) Packets.

14.6.163 Port j MAC Address20 High

The MAC_Address20_High register holds the upper 16 bits of the 20th 6-byte MAC address of the station. The first DA byte that is received on the (X)GMII interface corresponds to the LS byte (Bits [7:0]) of the MAC_Address20_Low register. For example, if 0x112233445566 is received (0x11 in lane 0 of the first column) on the (X)GMII as the destination address, then the MAC_Address20 register [47:0] is compared with 0x665544332211.

If the MAC address registers are configured to be double-synchronized to the (X)GMII clock domain, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the MAC_Address20_Low register are written, so MAC_Address20_High register must be written prior to MAC_Address20_Low register. For proper synchronization updates, the consecutive writes to MAC_Address20_Low register must be performed after at least four clock cycles of the slowest clock between the source and the destination clock domains.

Portj_MAC_Address20_High (j=0-1) Offset address: $103A0_H+j*2000_H$

Port j MAC Address20 High Kernel Reset value: $0000\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AE	SA	MBC						0					DCS		
rw	rw	rw						r					rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRHI															
rw															

Field	Bits	Type	Description
ADDRHI	15:0	rw	**MAC Address20 [47:32]** This field contains the upper 16 bits[47:32] of the 20th 6-byte MAC address.
DCS	18:16	rw	**DMA Channel Select** If the DDS bit of MAC_Extended_Configuration register is not set: This field contains the binary representation of the DMA Channel number to which an Rx packet whose DA matches the MAC Address0 content is routed. If the DDS bit of MAC_Extended_Configuration register is set: The XDCS field of MAC_DChSel_IndReg register contains the one-hot representation of one or more DMA Channel numbers to which an Rx packet whose DA matches the MAC Address20 content is routed.

(table continues...)

(continued)

Field	Bits	Type	Description
MBC	29:24	rw	**Mask Byte Control** These bits are mask control bits for comparing each of the MAC Address bytes. When set high, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address20 registers. Each bit controls the masking of the bytes as follows: - Bit 29: MAD Address High [15:8] - Bit 28: MAD Address High [7:0] - Bit 27: MAD Address Low[31:24] - ... - Bit 24: MAD Address Low[7:0] You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address.
SA	30	rw	**Source Address** - 1: The MAC Address20[47:0] is used to compare with the SA fields of the received packet. - 0: The MAC Address20[47:0] is used to compare with the DA fields of the received packet.
AE	31	rw	**Address Enable** - 1: The address filter module uses the 20th MAC address for perfect filtering. - 0: The address filter module ignores the address for filtering.
0	23:19	r	**Reserved** Read as 0; should be written with 0.

14.6.164 Port j MAC Address20 Low

The MAC_Address20_Low register holds the lower 32 bits of the 20th 6-byte MAC address of the station.

Portj_MAC_Address20_Low (j=0-1) Offset address: $103A4_H+j*2000_H$

Port j MAC Address20 Low Kernel Reset value: $FFFF\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADDRLO															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRLO															
rw															

Field	Bits	Type	Description
ADDRLO	31:0	rw	**MAC Address20 [31:0]** This field contains the lower 32 bits of 20th 6-byte MAC address. The MAC uses this field for filtering the received packets and inserting the MAC address in the Transmit Flow Control (Pause) Packets.

14.6.165 Port j MAC Address21 High

The MAC_Address21_High register holds the upper 16 bits of the 21th 6-byte MAC address of the station. The first DA byte that is received on the (X)GMII interface corresponds to the LS byte (Bits [7:0]) of the MAC_Address21_Low register. For example, if 0x112233445566 is received (0x11 in lane 0 of the first column) on the (X)GMII as the destination address, then the MAC_Address21 register [47:0] is compared with 0x665544332211.

If the MAC address registers are configured to be double-synchronized to the (X)GMII clock domain, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the MAC_Address21_Low register are written, so MAC_Address21_High register must be written prior to MAC_Address21_Low register. For proper synchronization updates, the consecutive writes to MAC_Address21_Low register must be performed after at least four clock cycles of the slowest clock between the source and the destination clock domains.

Portj_MAC_Address21_High (j=0-1) Offset address: $103A8_H+j*2000_H$

Port j MAC Address21 High Kernel Reset value: $0000\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AE	SA	MBC						0					DCS		
rw	rw	rw						r					rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRHI															
rw															

Field	Bits	Type	Description
ADDRHI	15:0	rw	**MAC Address21 [47:32]** This field contains the upper 16 bits[47:32] of the 21th 6-byte MAC address.
DCS	18:16	rw	**DMA Channel Select** If the DDS bit of MAC_Extended_Configuration register is not set: This field contains the binary representation of the DMA Channel number to which an Rx packet whose DA matches the MAC Address0 content is routed. If the DDS bit of MAC_Extended_Configuration register is set: The XDCS field of MAC_DChSel_IndReg register contains the one-hot representation of one or more DMA Channel numbers to which an Rx packet whose DA matches the MAC Address21 content is routed.

(table continues...)

(continued)

Field	Bits	Type	Description
MBC	29:24	rw	**Mask Byte Control** These bits are mask control bits for comparing each of the MAC Address bytes. When set high, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address21 registers. Each bit controls the masking of the bytes as follows: - Bit 29: MAD Address High [15:8] - Bit 28: MAD Address High [7:0] - Bit 27: MAD Address Low[31:24] - ... - Bit 24: MAD Address Low[7:0] You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address.
SA	30	rw	**Source Address** - 1: The MAC Address21[47:0] is used to compare with the SA fields of the received packet. - 0: The MAC Address21[47:0] is used to compare with the DA fields of the received packet.
AE	31	rw	**Address Enable** - 1: The address filter module uses the 21th MAC address for perfect filtering. - 0: The address filter module ignores the address for filtering.
0	23:19	r	**Reserved** Read as 0; should be written with 0.

14.6.166 Port j MAC Address21 Low

The MAC_Address21_Low register holds the lower 32 bits of the 21th 6-byte MAC address of the station.

Portj_MAC_Address21_Low (j=0-1) Offset address: $103AC_H+j*2000_H$

Port j MAC Address21 Low Kernel Reset value: $FFFF\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADDRLO															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRLO															
rw															

Field	Bits	Type	Description
ADDRLO	31:0	rw	**MAC Address21 [31:0]** This field contains the lower 32 bits of 21th 6-byte MAC address. The MAC uses this field for filtering the received packets and inserting the MAC address in the Transmit Flow Control (Pause) Packets.

14.6.167 Port j MAC Address22 High

The MAC_Address22_High register holds the upper 16 bits of the 22th 6-byte MAC address of the station. The first DA byte that is received on the (X)GMII interface corresponds to the LS byte (Bits [7:0]) of the MAC_Address22_Low register. For example, if 0x112233445566 is received (0x11 in lane 0 of the first column) on the (X)GMII as the destination address, then the MAC_Address22 register [47:0] is compared with 0x665544332211.

If the MAC address registers are configured to be double-synchronized to the (X)GMII clock domain, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the MAC_Address22_Low register are written, so MAC_Address22_High register must be written prior to MAC_Address22_Low register. For proper synchronization updates, the consecutive writes to MAC_Address22_Low register must be performed after at least four clock cycles of the slowest clock between the source and the destination clock domains.

Portj_MAC_Address22_High (j=0-1) Offset address: $103B0_H+j*2000_H$

Port j MAC Address22 High Kernel Reset value: $0000\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AE	**SA**	**MBC**						**0**					**DCS**		
rw	rw	rw						r					rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRHI															
rw															

Field	Bits	Type	Description
ADDRHI	15:0	rw	**MAC Address22 [47:32]** This field contains the upper 16 bits[47:32] of the 22th 6-byte MAC address.
DCS	18:16	rw	**DMA Channel Select** If the DDS bit of MAC_Extended_Configuration register is not set: This field contains the binary representation of the DMA Channel number to which an Rx packet whose DA matches the MAC Address0 content is routed. If the DDS bit of MAC_Extended_Configuration register is set: The XDCS field of MAC_DChSel_IndReg register contains the one-hot representation of one or more DMA Channel numbers to which an Rx packet whose DA matches the MAC Address22 content is routed.

(table continues...)

(continued)

Field	Bits	Type	Description
MBC	29:24	rw	**Mask Byte Control** These bits are mask control bits for comparing each of the MAC Address bytes. When set high, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address22 registers. Each bit controls the masking of the bytes as follows: - Bit 29: MAD Address High [15:8] - Bit 28: MAD Address High [7:0] - Bit 27: MAD Address Low[31:24] - ... - Bit 24: MAD Address Low[7:0] You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address.
SA	30	rw	**Source Address** - 1: The MAC Address22[47:0] is used to compare with the SA fields of the received packet. - 0: The MAC Address22[47:0] is used to compare with the DA fields of the received packet.
AE	31	rw	**Address Enable** - 1: The address filter module uses the 22th MAC address for perfect filtering. - 0: The address filter module ignores the address for filtering.
0	23:19	r	**Reserved** Read as 0; should be written with 0.

14.6.168 Port j MAC Address22 Low

The MAC_Address22_Low register holds the lower 32 bits of the 22th 6-byte MAC address of the station.

Portj_MAC_Address22_Low (j=0-1) Offset address: $103B4_H+j*2000_H$

Port j MAC Address22 Low Kernel Reset value: $FFFF\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADDRLO															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRLO															
rw															

Field	Bits	Type	Description
ADDRLO	31:0	rw	**MAC Address22 [31:0]** This field contains the lower 32 bits of 22th 6-byte MAC address. The MAC uses this field for filtering the received packets and inserting the MAC address in the Transmit Flow Control (Pause) Packets.

14.6.169 Port j MAC Address23 High

The MAC_Address23_High register holds the upper 16 bits of the 23th 6-byte MAC address of the station. The first DA byte that is received on the (X)GMII interface corresponds to the LS byte (Bits [7:0]) of the MAC_Address23_Low register. For example, if 0x112233445566 is received (0x11 in lane 0 of the first column) on the (X)GMII as the destination address, then the MAC_Address23 register [47:0] is compared with 0x665544332211.

If the MAC address registers are configured to be double-synchronized to the (X)GMII clock domain, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the MAC_Address23_Low register are written, so MAC_Address23_High register must be written prior to MAC_Address23_Low register. For proper synchronization updates, the consecutive writes to MAC_Address23_Low register must be performed after at least four clock cycles of the slowest clock between the source and the destination clock domains.

Portj_MAC_Address23_High (j=0-1) Offset address: $103B8_H+j*2000_H$

Port j MAC Address23 High Kernel Reset value: $0000\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AE	SA	MBC						0					DCS		
rw	rw	rw						r					rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRHI															
rw															

Field	Bits	Type	Description
ADDRHI	15:0	rw	**MAC Address23 [47:32]** This field contains the upper 16 bits[47:32] of the 23th 6-byte MAC address.
DCS	18:16	rw	**DMA Channel Select** If the DDS bit of MAC_Extended_Configuration register is not set: This field contains the binary representation of the DMA Channel number to which an Rx packet whose DA matches the MAC Address0 content is routed. If the DDS bit of MAC_Extended_Configuration register is set: The XDCS field of MAC_DChSel_IndReg register contains the one-hot representation of one or more DMA Channel numbers to which an Rx packet whose DA matches the MAC Address23 content is routed.

(table continues...)

(continued)

Field	Bits	Type	Description
MBC	29:24	rw	**Mask Byte Control** These bits are mask control bits for comparing each of the MAC Address bytes. When set high, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address23 registers. Each bit controls the masking of the bytes as follows: - Bit 29: MAD Address High [15:8] - Bit 28: MAD Address High [7:0] - Bit 27: MAD Address Low[31:24] - ... - Bit 24: MAD Address Low[7:0] You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address.
SA	30	rw	**Source Address** - 1: The MAC Address23[47:0] is used to compare with the SA fields of the received packet. - 0: The MAC Address23[47:0] is used to compare with the DA fields of the received packet.
AE	31	rw	**Address Enable** - 1: The address filter module uses the 23th MAC address for perfect filtering. - 0: The address filter module ignores the address for filtering.
0	23:19	r	**Reserved** Read as 0; should be written with 0.

14.6.170 Port j MAC Address23 Low

The MAC_Address23_Low register holds the lower 32 bits of the 23th 6-byte MAC address of the station.

Portj_MAC_Address23_Low (j=0-1) Offset address: $103BC_H+j*2000_H$

Port j MAC Address23 Low Kernel Reset value: $FFFF\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADDRLO															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRLO															
rw															

Field	Bits	Type	Description
ADDRLO	31:0	rw	**MAC Address23 [31:0]** This field contains the lower 32 bits of 23th 6-byte MAC address. The MAC uses this field for filtering the received packets and inserting the MAC address in the Transmit Flow Control (Pause) Packets.

14.6.171 Port j MAC Address24 High

The MAC_Address24_High register holds the upper 16 bits of the 24th 6-byte MAC address of the station. The first DA byte that is received on the (X)GMII interface corresponds to the LS byte (Bits [7:0]) of the MAC_Address24_Low register. For example, if 0x112233445566 is received (0x11 in lane 0 of the first column) on the (X)GMII as the destination address, then the MAC_Address24 register [47:0] is compared with 0x665544332211.

If the MAC address registers are configured to be double-synchronized to the (X)GMII clock domain, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the MAC_Address24_Low register are written, so MAC_Address24_High register must be written prior to MAC_Address24_Low register. For proper synchronization updates, the consecutive writes to MAC_Address24_Low register must be performed after at least four clock cycles of the slowest clock between the source and the destination clock domains.

Portj_MAC_Address24_High (j=0-1) Offset address: $103C0_H+j*2000_H$

Port j MAC Address24 High Kernel Reset value: $0000\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AE	SA	MBC						0					DCS		
rw	rw	rw						r					rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRHI															
rw															

Field	Bits	Type	Description
ADDRHI	15:0	rw	**MAC Address24 [47:32]** This field contains the upper 16 bits[47:32] of the 24th 6-byte MAC address.
DCS	18:16	rw	**DMA Channel Select** If the DDS bit of MAC_Extended_Configuration register is not set: This field contains the binary representation of the DMA Channel number to which an Rx packet whose DA matches the MAC Address0 content is routed. If the DDS bit of MAC_Extended_Configuration register is set: The XDCS field of MAC_DChSel_IndReg register contains the one-hot representation of one or more DMA Channel numbers to which an Rx packet whose DA matches the MAC Address24 content is routed.

(table continues...)

(continued)

Field	Bits	Type	Description
MBC	29:24	rw	**Mask Byte Control** These bits are mask control bits for comparing each of the MAC Address bytes. When set high, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address24 registers. Each bit controls the masking of the bytes as follows: - Bit 29: MAD Address High [15:8] - Bit 28: MAD Address High [7:0] - Bit 27: MAD Address Low[31:24] - ... - Bit 24: MAD Address Low[7:0] You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address.
SA	30	rw	**Source Address** - 1: The MAC Address24[47:0] is used to compare with the SA fields of the received packet. - 0: The MAC Address24[47:0] is used to compare with the DA fields of the received packet.
AE	31	rw	**Address Enable** - 1: The address filter module uses the 24th MAC address for perfect filtering. - 0: The address filter module ignores the address for filtering.
0	23:19	r	**Reserved** Read as 0; should be written with 0.

14.6.172 Port j MAC Address24 Low

The MAC_Address24_Low register holds the lower 32 bits of the 24th 6-byte MAC address of the station.

Portj_MAC_Address24_Low (j=0-1) Offset address: $103C4_H+j*2000_H$

Port j MAC Address24 Low Kernel Reset value: $FFFF\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADDRLO															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRLO															
rw															

Field	Bits	Type	Description
ADDRLO	31:0	rw	**MAC Address24 [31:0]** This field contains the lower 32 bits of 24th 6-byte MAC address. The MAC uses this field for filtering the received packets and inserting the MAC address in the Transmit Flow Control (Pause) Packets.

14.6.173 Port j MAC Address25 High

The MAC_Address25_High register holds the upper 16 bits of the 25th 6-byte MAC address of the station. The first DA byte that is received on the (X)GMII interface corresponds to the LS byte (Bits [7:0]) of the MAC_Address25_Low register. For example, if 0x112233445566 is received (0x11 in lane 0 of the first column) on the (X)GMII as the destination address, then the MAC_Address25 register [47:0] is compared with 0x665544332211.

If the MAC address registers are configured to be double-synchronized to the (X)GMII clock domain, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the MAC_Address25_Low register are written, so MAC_Address25_High register must be written prior to MAC_Address25_Low register. For proper synchronization updates, the consecutive writes to MAC_Address25_Low register must be performed after at least four clock cycles of the slowest clock between the source and the destination clock domains.

Portj_MAC_Address25_High (j=0-1) Offset address: $103C8_H+j*2000_H$

Port j MAC Address25 High Kernel Reset value: $0000\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AE	SA	MBC						0					DCS		
rw	rw	rw						r					rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRHI															
rw															

Field	Bits	Type	Description
ADDRHI	15:0	rw	**MAC Address25 [47:32]** This field contains the upper 16 bits[47:32] of the 25th 6-byte MAC address.
DCS	18:16	rw	**DMA Channel Select** If the DDS bit of MAC_Extended_Configuration register is not set: This field contains the binary representation of the DMA Channel number to which an Rx packet whose DA matches the MAC Address0 content is routed. If the DDS bit of MAC_Extended_Configuration register is set: The XDCS field of MAC_DChSel_IndReg register contains the one-hot representation of one or more DMA Channel numbers to which an Rx packet whose DA matches the MAC Address25 content is routed.

(table continues...)

(continued)

Field	Bits	Type	Description
MBC	29:24	rw	**Mask Byte Control** These bits are mask control bits for comparing each of the MAC Address bytes. When set high, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address25 registers. Each bit controls the masking of the bytes as follows: - Bit 29: MAD Address High [15:8] - Bit 28: MAD Address High [7:0] - Bit 27: MAD Address Low[31:24] - ... - Bit 24: MAD Address Low[7:0] You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address.
SA	30	rw	**Source Address** - 1: The MAC Address25[47:0] is used to compare with the SA fields of the received packet. - 0: The MAC Address25[47:0] is used to compare with the DA fields of the received packet.
AE	31	rw	**Address Enable** - 1: The address filter module uses the 25th MAC address for perfect filtering. - 0: The address filter module ignores the address for filtering.
0	23:19	r	**Reserved** Read as 0; should be written with 0.

14.6.174 Port j MAC Address25 Low

The MAC_Address25_Low register holds the lower 32 bits of the 25th 6-byte MAC address of the station.

Portj_MAC_Address25_Low (j=0-1) Offset address: $103CC_H+j*2000_H$

Port j MAC Address25 Low Kernel Reset value: $FFFF\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADDRLO															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRLO															
rw															

Field	Bits	Type	Description
ADDRLO	31:0	rw	**MAC Address25 [31:0]** This field contains the lower 32 bits of 25th 6-byte MAC address. The MAC uses this field for filtering the received packets and inserting the MAC address in the Transmit Flow Control (Pause) Packets.

14.6.175 Port j MAC Address26 High

The MAC_Address26_High register holds the upper 16 bits of the 26th 6-byte MAC address of the station. The first DA byte that is received on the (X)GMII interface corresponds to the LS byte (Bits [7:0]) of the MAC_Address26_Low register. For example, if 0x112233445566 is received (0x11 in lane 0 of the first column) on the (X)GMII as the destination address, then the MAC_Address26 register [47:0] is compared with 0x665544332211.

If the MAC address registers are configured to be double-synchronized to the (X)GMII clock domain, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the MAC_Address26_Low register are written, so MAC_Address26_High register must be written prior to MAC_Address26_Low register. For proper synchronization updates, the consecutive writes to MAC_Address26_Low register must be performed after at least four clock cycles of the slowest clock between the source and the destination clock domains.

Portj_MAC_Address26_High (j=0-1) Offset address: $103D0_H+j*2000_H$

Port j MAC Address26 High Kernel Reset value: $0000\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AE	SA	MBC						0					DCS		
rw	rw	rw						r					rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRHI															
rw															

Field	Bits	Type	Description
ADDRHI	15:0	rw	**MAC Address26 [47:32]** This field contains the upper 16 bits[47:32] of the 26th 6-byte MAC address.
DCS	18:16	rw	**DMA Channel Select** If the DDS bit of MAC_Extended_Configuration register is not set: This field contains the binary representation of the DMA Channel number to which an Rx packet whose DA matches the MAC Address0 content is routed. If the DDS bit of MAC_Extended_Configuration register is set: The XDCS field of MAC_DChSel_IndReg register contains the one-hot representation of one or more DMA Channel numbers to which an Rx packet whose DA matches the MAC Address26 content is routed.

(table continues...)

(continued)

Field	Bits	Type	Description
MBC	29:24	rw	**Mask Byte Control** These bits are mask control bits for comparing each of the MAC Address bytes. When set high, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address26 registers. Each bit controls the masking of the bytes as follows: - Bit 29: MAD Address High [15:8] - Bit 28: MAD Address High [7:0] - Bit 27: MAD Address Low[31:24] - ... - Bit 24: MAD Address Low[7:0] You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address.
SA	30	rw	**Source Address** - 1: The MAC Address26[47:0] is used to compare with the SA fields of the received packet. - 0: The MAC Address26[47:0] is used to compare with the DA fields of the received packet.
AE	31	rw	**Address Enable** - 1: The address filter module uses the 26th MAC address for perfect filtering. - 0: The address filter module ignores the address for filtering.
0	23:19	r	**Reserved** Read as 0; should be written with 0.

14.6.176 Port j MAC Address26 Low

The MAC_Address26_Low register holds the lower 32 bits of the 26th 6-byte MAC address of the station.

Portj_MAC_Address26_Low (j=0-1) Offset address: $103D4_H+j*2000_H$

Port j MAC Address26 Low Kernel Reset value: $FFFF\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADDRLO															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRLO															
rw															

Field	Bits	Type	Description
ADDRLO	31:0	rw	**MAC Address26 [31:0]** This field contains the lower 32 bits of 26th 6-byte MAC address. The MAC uses this field for filtering the received packets and inserting the MAC address in the Transmit Flow Control (Pause) Packets.

14.6.177 Port j MAC Address27 High

The MAC_Address27_High register holds the upper 16 bits of the 27th 6-byte MAC address of the station. The first DA byte that is received on the (X)GMII interface corresponds to the LS byte (Bits [7:0]) of the MAC_Address27_Low register. For example, if 0x112233445566 is received (0x11 in lane 0 of the first column) on the (X)GMII as the destination address, then the MAC_Address27 register [47:0] is compared with 0x665544332211.

If the MAC address registers are configured to be double-synchronized to the (X)GMII clock domain, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the MAC_Address27_Low register are written, so MAC_Address27_High register must be written prior to MAC_Address27_Low register. For proper synchronization updates, the consecutive writes to MAC_Address27_Low register must be performed after at least four clock cycles of the slowest clock between the source and the destination clock domains.

Portj_MAC_Address27_High (j=0-1) Offset address: $103D8_H+j*2000_H$

Port j MAC Address27 High Kernel Reset value: $0000\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AE	SA	MBC						0					DCS		
rw	rw	rw						r					rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRHI															
rw															

Field	Bits	Type	Description
ADDRHI	15:0	rw	**MAC Address27 [47:32]** This field contains the upper 16 bits[47:32] of the 27th 6-byte MAC address.
DCS	18:16	rw	**DMA Channel Select** If the DDS bit of MAC_Extended_Configuration register is not set: This field contains the binary representation of the DMA Channel number to which an Rx packet whose DA matches the MAC Address0 content is routed. If the DDS bit of MAC_Extended_Configuration register is set: The XDCS field of MAC_DChSel_IndReg register contains the one-hot representation of one or more DMA Channel numbers to which an Rx packet whose DA matches the MAC Address27 content is routed.

(table continues...)

(continued)

Field	Bits	Type	Description
MBC	29:24	rw	**Mask Byte Control** These bits are mask control bits for comparing each of the MAC Address bytes. When set high, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address27 registers. Each bit controls the masking of the bytes as follows: - Bit 29: MAD Address High [15:8] - Bit 28: MAD Address High [7:0] - Bit 27: MAD Address Low[31:24] - ... - Bit 24: MAD Address Low[7:0] You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address.
SA	30	rw	**Source Address** - 1: The MAC Address27[47:0] is used to compare with the SA fields of the received packet. - 0: The MAC Address27[47:0] is used to compare with the DA fields of the received packet.
AE	31	rw	**Address Enable** - 1: The address filter module uses the 27th MAC address for perfect filtering. - 0: The address filter module ignores the address for filtering.
0	23:19	r	**Reserved** Read as 0; should be written with 0.

14.6.178 Port j MAC Address27 Low

The MAC_Address27_Low register holds the lower 32 bits of the 27th 6-byte MAC address of the station.

Portj_MAC_Address27_Low (j=0-1) Offset address: $103DC_H+j*2000_H$

Port j MAC Address27 Low Kernel Reset value: $FFFF\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADDRLO															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRLO															
rw															

Field	Bits	Type	Description
ADDRLO	31:0	rw	**MAC Address27 [31:0]** This field contains the lower 32 bits of 27th 6-byte MAC address. The MAC uses this field for filtering the received packets and inserting the MAC address in the Transmit Flow Control (Pause) Packets.

14.6.179 Port j MAC Address28 High

The MAC_Address28_High register holds the upper 16 bits of the 28th 6-byte MAC address of the station. The first DA byte that is received on the (X)GMII interface corresponds to the LS byte (Bits [7:0]) of the MAC_Address28_Low register. For example, if 0x112233445566 is received (0x11 in lane 0 of the first column) on the (X)GMII as the destination address, then the MAC_Address28 register [47:0] is compared with 0x665544332211.

If the MAC address registers are configured to be double-synchronized to the (X)GMII clock domain, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the MAC_Address28_Low register are written, so MAC_Address28_High register must be written prior to MAC_Address28_Low register. For proper synchronization updates, the consecutive writes to MAC_Address28_Low register must be performed after at least four clock cycles of the slowest clock between the source and the destination clock domains.

Portj_MAC_Address28_High (j=0-1) Offset address: $103E0_H+j*2000_H$

Port j MAC Address28 High Kernel Reset value: $0000\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AE	SA	MBC						0					DCS		
rw	rw	rw						r					rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRHI															
rw															

Field	Bits	Type	Description
ADDRHI	15:0	rw	**MAC Address28 [47:32]** This field contains the upper 16 bits[47:32] of the 28th 6-byte MAC address.
DCS	18:16	rw	**DMA Channel Select** If the DDS bit of MAC_Extended_Configuration register is not set: This field contains the binary representation of the DMA Channel number to which an Rx packet whose DA matches the MAC Address0 content is routed. If the DDS bit of MAC_Extended_Configuration register is set: The XDCS field of MAC_DChSel_IndReg register contains the one-hot representation of one or more DMA Channel numbers to which an Rx packet whose DA matches the MAC Address28 content is routed.

(table continues...)

(continued)

Field	Bits	Type	Description
MBC	29:24	rw	**Mask Byte Control** These bits are mask control bits for comparing each of the MAC Address bytes. When set high, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address28 registers. Each bit controls the masking of the bytes as follows: - Bit 29: MAD Address High [15:8] - Bit 28: MAD Address High [7:0] - Bit 27: MAD Address Low[31:24] - ... - Bit 24: MAD Address Low[7:0] You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address.
SA	30	rw	**Source Address** - 1: The MAC Address28[47:0] is used to compare with the SA fields of the received packet. - 0: The MAC Address28[47:0] is used to compare with the DA fields of the received packet.
AE	31	rw	**Address Enable** - 1: The address filter module uses the 28th MAC address for perfect filtering. - 0: The address filter module ignores the address for filtering.
0	23:19	r	**Reserved** Read as 0; should be written with 0.

14.6.180 Port j MAC Address28 Low

The MAC_Address28_Low register holds the lower 32 bits of the 28th 6-byte MAC address of the station.

Portj_MAC_Address28_Low (j=0-1) Offset address: $103E4_H+j*2000_H$

Port j MAC Address28 Low Kernel Reset value: $FFFF\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADDRLO															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRLO															
rw															

Field	Bits	Type	Description
ADDRLO	31:0	rw	**MAC Address28 [31:0]** This field contains the lower 32 bits of 28th 6-byte MAC address. The MAC uses this field for filtering the received packets and inserting the MAC address in the Transmit Flow Control (Pause) Packets.

14.6.181 Port j MAC Address29 High

The MAC_Address29_High register holds the upper 16 bits of the 29th 6-byte MAC address of the station. The first DA byte that is received on the (X)GMII interface corresponds to the LS byte (Bits [7:0]) of the MAC_Address29_Low register. For example, if 0x112233445566 is received (0x11 in lane 0 of the first column) on the (X)GMII as the destination address, then the MAC_Address29 register [47:0] is compared with 0x665544332211.

If the MAC address registers are configured to be double-synchronized to the (X)GMII clock domain, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the MAC_Address29_Low register are written, so MAC_Address29_High register must be written prior to MAC_Address29_Low register. For proper synchronization updates, the consecutive writes to MAC_Address29_Low register must be performed after at least four clock cycles of the slowest clock between the source and the destination clock domains.

Portj_MAC_Address29_High (j=0-1) Offset address: $103E8_H+j*2000_H$

Port j MAC Address29 High Kernel Reset value: $0000\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AE	SA	MBC						0					DCS		
rw	rw	rw						r					rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRHI															
rw															

Field	Bits	Type	Description
ADDRHI	15:0	rw	**MAC Address29 [47:32]** This field contains the upper 16 bits[47:32] of the 29th 6-byte MAC address.
DCS	18:16	rw	**DMA Channel Select** If the DDS bit of MAC_Extended_Configuration register is not set: This field contains the binary representation of the DMA Channel number to which an Rx packet whose DA matches the MAC Address0 content is routed. If the DDS bit of MAC_Extended_Configuration register is set: The XDCS field of MAC_DChSel_IndReg register contains the one-hot representation of one or more DMA Channel numbers to which an Rx packet whose DA matches the MAC Address29 content is routed.

(table continues...)

(continued)

Field	Bits	Type	Description
MBC	29:24	rw	**Mask Byte Control** These bits are mask control bits for comparing each of the MAC Address bytes. When set high, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address29 registers. Each bit controls the masking of the bytes as follows: - Bit 29: MAD Address High [15:8] - Bit 28: MAD Address High [7:0] - Bit 27: MAD Address Low[31:24] - ... - Bit 24: MAD Address Low[7:0] You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address.
SA	30	rw	**Source Address** - 1: The MAC Address29[47:0] is used to compare with the SA fields of the received packet. - 0: The MAC Address29[47:0] is used to compare with the DA fields of the received packet.
AE	31	rw	**Address Enable** - 1: The address filter module uses the 29th MAC address for perfect filtering. - 0: The address filter module ignores the address for filtering.
0	23:19	r	**Reserved** Read as 0; should be written with 0.

14.6.182 Port j MAC Address29 Low

The MAC_Address29_Low register holds the lower 32 bits of the 29th 6-byte MAC address of the station.

Portj_MAC_Address29_Low (j=0-1) Offset address: $103EC_H+j*2000_H$

Port j MAC Address29 Low Kernel Reset value: $FFFF\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADDRLO															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRLO															
rw															

Field	Bits	Type	Description
ADDRLO	31:0	rw	**MAC Address29 [31:0]** This field contains the lower 32 bits of 29th 6-byte MAC address. The MAC uses this field for filtering the received packets and inserting the MAC address in the Transmit Flow Control (Pause) Packets.

14.6.183 Port j MAC Address30 High

The MAC_Address30_High register holds the upper 16 bits of the 30th 6-byte MAC address of the station. The first DA byte that is received on the (X)GMII interface corresponds to the LS byte (Bits [7:0]) of the MAC_Address30_Low register. For example, if 0x112233445566 is received (0x11 in lane 0 of the first column) on the (X)GMII as the destination address, then the MAC_Address30 register [47:0] is compared with 0x665544332211.

If the MAC address registers are configured to be double-synchronized to the (X)GMII clock domain, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the MAC_Address30_Low register are written, so MAC_Address30_High register must be written prior to MAC_Address30_Low register. For proper synchronization updates, the consecutive writes to MAC_Address30_Low register must be performed after at least four clock cycles of the slowest clock between the source and the destination clock domains.

Portj_MAC_Address30_High (j=0-1) Offset address: $103F0_H+j*2000_H$

Port j MAC Address30 High Kernel Reset value: $0000\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AE	SA	MBC						0					DCS		
rw	rw	rw						r					rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRHI															
rw															

Field	Bits	Type	Description
ADDRHI	15:0	rw	**MAC Address30 [47:32]** This field contains the upper 16 bits[47:32] of the 30th 6-byte MAC address.
DCS	18:16	rw	**DMA Channel Select** If the DDS bit of MAC_Extended_Configuration register is not set: This field contains the binary representation of the DMA Channel number to which an Rx packet whose DA matches the MAC Address0 content is routed. If the DDS bit of MAC_Extended_Configuration register is set: The XDCS field of MAC_DChSel_IndReg register contains the one-hot representation of one or more DMA Channel numbers to which an Rx packet whose DA matches the MAC Address30 content is routed.

(table continues...)

(continued)

Field	Bits	Type	Description
MBC	29:24	rw	**Mask Byte Control** These bits are mask control bits for comparing each of the MAC Address bytes. When set high, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address30 registers. Each bit controls the masking of the bytes as follows: - Bit 29: MAD Address High [15:8] - Bit 28: MAD Address High [7:0] - Bit 27: MAD Address Low[31:24] - ... - Bit 24: MAD Address Low[7:0] You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address.
SA	30	rw	**Source Address** - 1: The MAC Address30[47:0] is used to compare with the SA fields of the received packet. - 0: The MAC Address30[47:0] is used to compare with the DA fields of the received packet.
AE	31	rw	**Address Enable** - 1: The address filter module uses the 30th MAC address for perfect filtering. - 0: The address filter module ignores the address for filtering.
0	23:19	r	**Reserved** Read as 0; should be written with 0.

14.6.184 Port j MAC Address30 Low

The MAC_Address30_Low register holds the lower 32 bits of the 30th 6-byte MAC address of the station.

Portj_MAC_Address30_Low (j=0-1) Offset address: $103F4_H+j*2000_H$

Port j MAC Address30 Low Kernel Reset value: $FFFF\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADDRLO															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRLO															
rw															

Field	Bits	Type	Description
ADDRLO	31:0	rw	**MAC Address30 [31:0]** This field contains the lower 32 bits of 30th 6-byte MAC address. The MAC uses this field for filtering the received packets and inserting the MAC address in the Transmit Flow Control (Pause) Packets.

14.6.185 Port j MAC Address31 High

The MAC_Address31_High register holds the upper 16 bits of the 31th 6-byte MAC address of the station. The first DA byte that is received on the (X)GMII interface corresponds to the LS byte (Bits [7:0]) of the MAC_Address31_Low register. For example, if 0x112233445566 is received (0x11 in lane 0 of the first column) on the (X)GMII as the destination address, then the MAC_Address31 register [47:0] is compared with 0x665544332211.

If the MAC address registers are configured to be double-synchronized to the (X)GMII clock domain, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the MAC_Address31_Low register are written, so MAC_Address31_High register must be written prior to MAC_Address31_Low register. For proper synchronization updates, the consecutive writes to MAC_Address31_Low register must be performed after at least four clock cycles of the slowest clock between the source and the destination clock domains.

Portj_MAC_Address31_High (j=0-1) Offset address: $103F8_H+j*2000_H$

Port j MAC Address31 High Kernel Reset value: $0000\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AE	SA	MBC						0					DCS		
rw	rw	rw						r					rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRHI															
rw															

Field	Bits	Type	Description
ADDRHI	15:0	rw	**MAC Address31 [47:32]** This field contains the upper 16 bits[47:32] of the 31th 6-byte MAC address.
DCS	18:16	rw	**DMA Channel Select** If the DDS bit of MAC_Extended_Configuration register is not set: This field contains the binary representation of the DMA Channel number to which an Rx packet whose DA matches the MAC Address0 content is routed. If the DDS bit of MAC_Extended_Configuration register is set: The XDCS field of MAC_DChSel_IndReg register contains the one-hot representation of one or more DMA Channel numbers to which an Rx packet whose DA matches the MAC Address31 content is routed.

(table continues...)

(continued)

Field	Bits	Type	Description
MBC	29:24	rw	**Mask Byte Control** These bits are mask control bits for comparing each of the MAC Address bytes. When set high, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address31 registers. Each bit controls the masking of the bytes as follows: - Bit 29: MAD Address High [15:8] - Bit 28: MAD Address High [7:0] - Bit 27: MAD Address Low[31:24] - ... - Bit 24: MAD Address Low[7:0] You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address.
SA	30	rw	**Source Address** - 1: The MAC Address31[47:0] is used to compare with the SA fields of the received packet. - 0: The MAC Address31[47:0] is used to compare with the DA fields of the received packet.
AE	31	rw	**Address Enable** - 1: The address filter module uses the 31th MAC address for perfect filtering. - 0: The address filter module ignores the address for filtering.
0	23:19	r	**Reserved** Read as 0; should be written with 0.

14.6.186 Port j MAC Address31 Low

The MAC_Address31_Low register holds the lower 32 bits of the 31th 6-byte MAC address of the station.

Portj_MAC_Address31_Low (j=0-1) Offset address: $103FC_H+j*2000_H$

Port j MAC Address31 Low Kernel Reset value: $FFFF\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADDRLO															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRLO															
rw															

Field	Bits	Type	Description
ADDRLO	31:0	rw	**MAC Address31 [31:0]** This field contains the lower 32 bits of 31th 6-byte MAC address. The MAC uses this field for filtering the received packets and inserting the MAC address in the Transmit Flow Control (Pause) Packets.

14.6.187 Port j MAC Indir Access Ctrl

This register provides the Indirect Access control and status for MAC_(#MSEL)_IndReg(#AOFF) registers.

Portj_MAC_Indir_Access_Ctrl (j=0-1) Offset address: $10700_H+j*2000_H$

Port j MAC Indir Access Ctrl Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SNPS_R	0	MSEL				0		0							
rw	r	rw				r		r							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AOFF								0		AUTO	0			COM	OB
rw								r		rw	r			rw	rw

Field	Bits	Type	Description
OB	0	rw	**Operation Busy** This bit is set along with a read or write command for initiating the indirect access to MAC_(#MSEL)_IndReg(#AOFF) registers. This bit is reset when the read or write command to MAC_(#MSEL)_IndReg(#AOFF) register is complete. The next indirect register access can be initiated only after this bit is reset. During a write operation, the bit is reset only after the data has been written into MAC_(#MSEL)_IndReg(#AOFF) register. During a read operation, the data must be read from MAC_Indir_Access_Data register only after this bit is reset. To initiate an indirect access, this bit must be set at last.
COM	1	rw	**Command type** This bit indicates if the current register access is a read or a write. When set, it indicates a read operation. When reset, it indicates a write operation. 0_B **WRITE**: Write operation 1_B **READ**: Read operation
AUTO	5	rw	**Auto increment - When Auto increment is set, AOFF is incremented by 1 and care must be taken by software not to cause a wrap condition** - Byte wise read/write is not supported when auto increment is enabled.

(table continues...)

(continued)

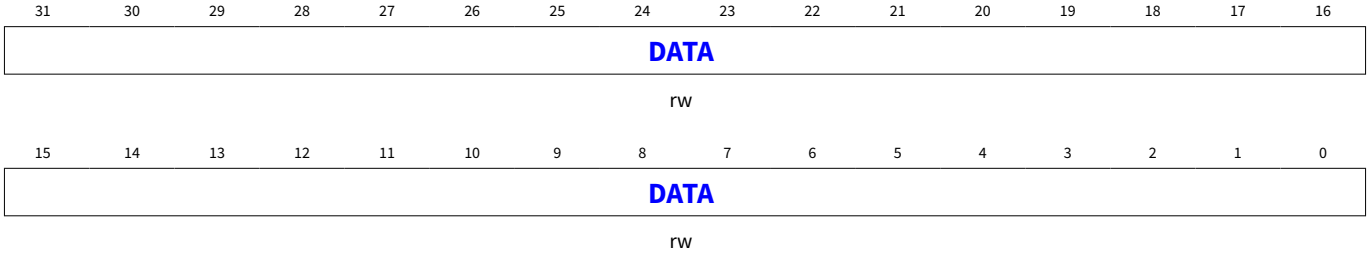
Field	Bits	Type	Description
AOFF	15:8	rw	**Address Offset** This field is used in indirect access of MAC_(#MSEL)_IndReg(#AOFF) registers. This field must be set along with initiation of read/write to MAC_(#MSEL)_IndReg(#AOFF) registers and must not be changed until the OB is reset. Note: Supported maximum value of AOFF are specific to MSEL. Based on maximum value remaining bits are not implemented and are reserved. AOFF Values: - 0x0 : IndReg0 (Indirect register 0) - ... - 0xFFFF : IndReg65535
MSEL	29:26	rw	**Mode Select** This field is used in indirect access of MAC_(#MSEL)_IndReg(#AOFF) registers. This field must be set along with initiation of read/write to MAC_(#MSEL)_IndReg(#AOFF) registers and must not be changed until the OB is reset. Values: - 0x0: DChSel (DMA_Channel_Select) - 0x1: PCCtrl (Policing_Counter_Control) - 0x2: PCntr (Policing_Counter) - 0x3: DPCSel (DA_Policing_Pounter_Select) - 0x4: VPCSel (VLAN_Policing_Counter_Select) - 0x5: LPCSel (L3L4_Policing_Counter_Select) - 0x6: FPCSel (Additional_Event_Policing_Counter_Select) - 0x7: PCStatus (Policing_Counter_Status for DA, VLAN, L3L4 and Additional_Event) - 0x8: EXT_DAH (External DA Hashtable) - 0x9: EXT_ROVTL (External Rx Outer VLAN Lookup) - 0xA: EXT_RIVTL (External Rx Inner VLAN Lookup) - 0xB - 0xF: Reserved for future use.
SNPS_R	31	rw	**Synopsys Reserved** This bit must be set to 0. This field is reserved for Synopsys Internal use, and must always be set to 0 unless instructed by Synopsys. Setting any bit to 1 might cause unexpected behavior in the IP.
0	4:2, 7:6, 23:16, 25:24, 30	r	**Reserved** Read as 0; should be written with 0.

14.6.188 Port j MAC Indir Access Data

This register holds the read/write data for Indirect Access of MAC_(#MSEL)_IndReg(#AOFF) registers.

Portj_MAC_Indir_Access_Data (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC Indir Access Data Kernel Reset value: $0000\ 0000_H$



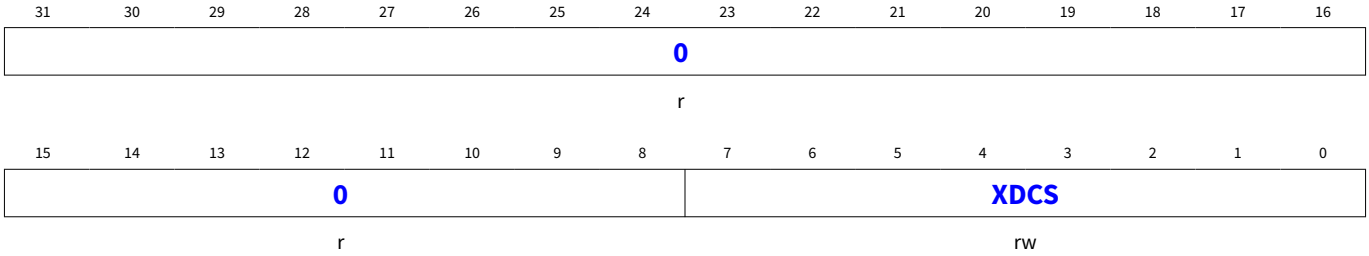
Field	Bits	Type	Description
DATA	31:0	rw	**This field contains data to read/write for Indirect address access associated with MAC_Indir_Access_Ctrl register**

14.6.189 Port j MAC DChSel IndReg0

This register contains DCS (XDCS) one hot encoded value of receive channel selection for multicast/broadcast.

Portj_MAC_DChSel_IndReg0 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC DChSel IndReg0 Kernel Reset value: $0000\ 0000_H$



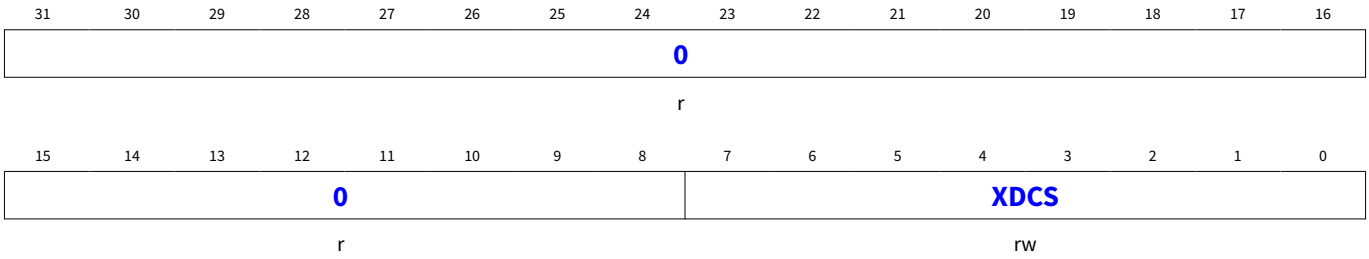
Field	Bits	Type	Description
XDCS	7:0	rw	**DMA Channel Select** This field contains one hot encoded DMA receive channel selection for MAC Address registers to forward the received MC/BC packet.
0	31:8	r	**Reserved** Read as 0; should be written with 0.

14.6.190 Port j MAC DChSel IndReg1

This register contains DCS (XDCS) one hot encoded value of receive channel selection for multicast/broadcast.

Portj_MAC_DChSel_IndReg1 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC DChSel IndReg1 Kernel Reset value: $0000\ 0000_H$



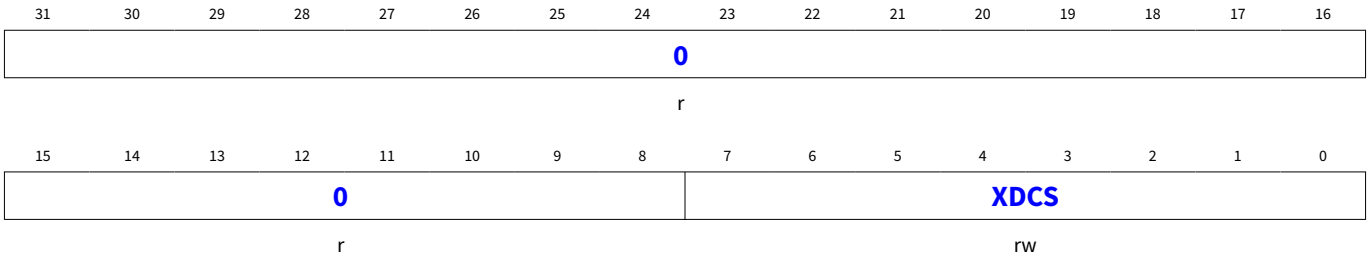
Field	Bits	Type	Description
XDCS	7:0	rw	**DMA Channel Select** This field contains one hot encoded DMA receive channel selection for MAC Address registers to forward the received MC/BC packet.
0	31:8	r	**Reserved** Read as 0; should be written with 0.

14.6.191 Port j MAC DChSel IndReg10

This register contains DCS (XDCS) one hot encoded value of receive channel selection for multicast/broadcast.

Portj_MAC_DChSel_IndReg10 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC DChSel IndReg10 Kernel Reset value: $0000\ 0000_H$



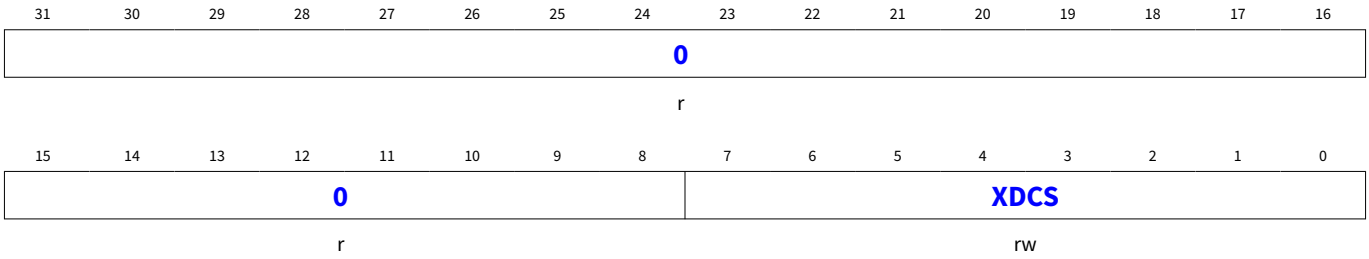
Field	Bits	Type	Description
XDCS	7:0	rw	**DMA Channel Select** This field contains one hot encoded DMA receive channel selection for MAC Address registers to forward the received MC/BC packet.
0	31:8	r	**Reserved** Read as 0; should be written with 0.

14.6.192 Port j MAC DChSel IndReg11

This register contains DCS (XDCS) one hot encoded value of receive channel selection for multicast/broadcast.

Portj_MAC_DChSel_IndReg11 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC DChSel IndReg11 Kernel Reset value: $0000\ 0000_H$



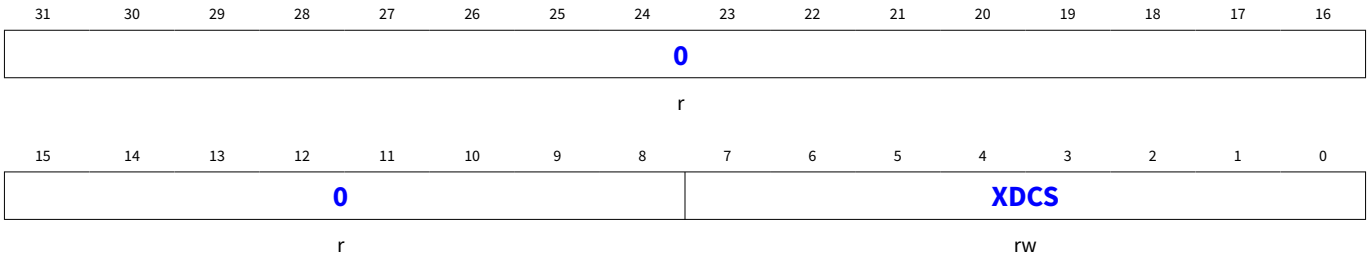
Field	Bits	Type	Description
XDCS	7:0	rw	**DMA Channel Select** This field contains one hot encoded DMA receive channel selection for MAC Address registers to forward the received MC/BC packet.
0	31:8	r	**Reserved** Read as 0; should be written with 0.

14.6.193 Port j MAC DChSel IndReg12

This register contains DCS (XDCS) one hot encoded value of receive channel selection for multicast/broadcast.

Portj_MAC_DChSel_IndReg12 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC DChSel IndReg12 Kernel Reset value: $0000\ 0000_H$



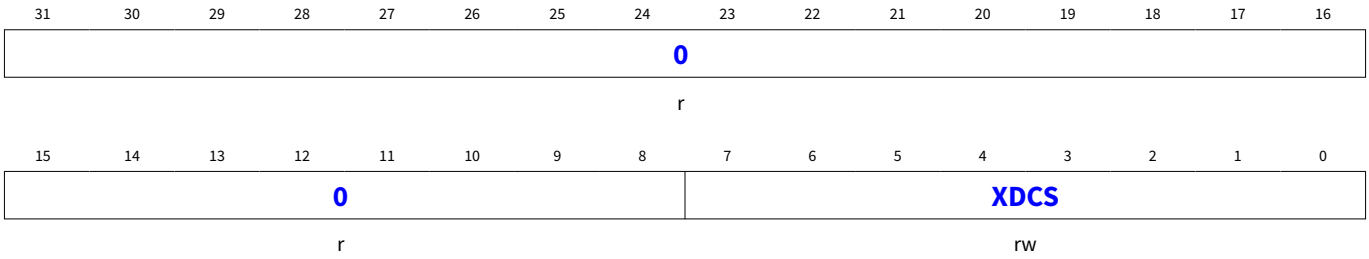
Field	Bits	Type	Description
XDCS	7:0	rw	**DMA Channel Select** This field contains one hot encoded DMA receive channel selection for MAC Address registers to forward the received MC/BC packet.
0	31:8	r	**Reserved** Read as 0; should be written with 0.

14.6.194 Port j MAC DChSel IndReg13

This register contains DCS (XDCS) one hot encoded value of receive channel selection for multicast/broadcast.

Portj_MAC_DChSel_IndReg13 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC DChSel IndReg13 Kernel Reset value: $0000\ 0000_H$



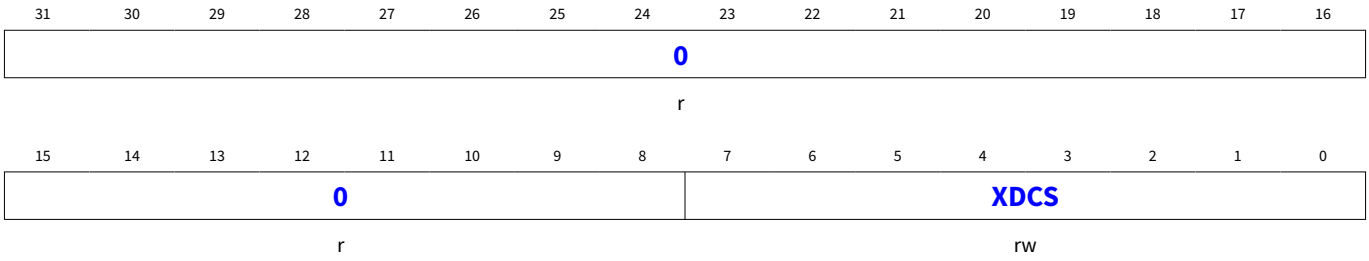
Field	Bits	Type	Description
XDCS	7:0	rw	**DMA Channel Select** This field contains one hot encoded DMA receive channel selection for MAC Address registers to forward the received MC/BC packet.
0	31:8	r	**Reserved** Read as 0; should be written with 0.

14.6.195 Port j MAC DChSel IndReg14

This register contains DCS (XDCS) one hot encoded value of receive channel selection for multicast/broadcast.

Portj_MAC_DChSel_IndReg14 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC DChSel IndReg14 Kernel Reset value: $0000\ 0000_H$



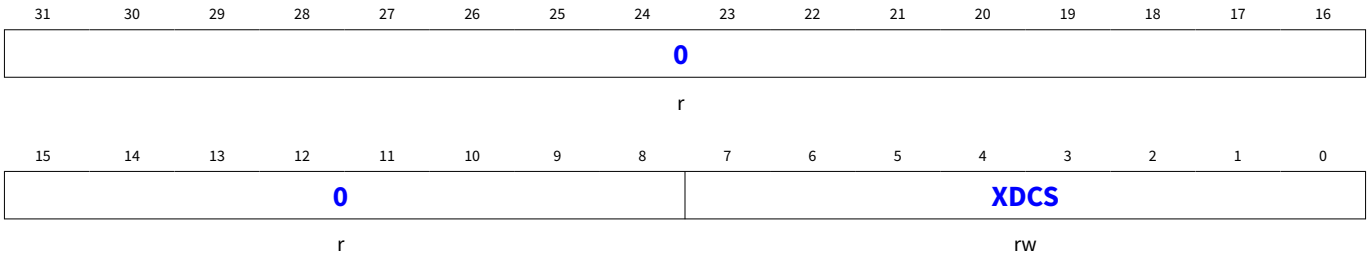
Field	Bits	Type	Description
XDCS	7:0	rw	**DMA Channel Select** This field contains one hot encoded DMA receive channel selection for MAC Address registers to forward the received MC/BC packet.
0	31:8	r	**Reserved** Read as 0; should be written with 0.

14.6.196 Port j MAC DChSel IndReg15

This register contains DCS (XDCS) one hot encoded value of receive channel selection for multicast/broadcast.

Portj_MAC_DChSel_IndReg15 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC DChSel IndReg15 Kernel Reset value: $0000\ 0000_H$



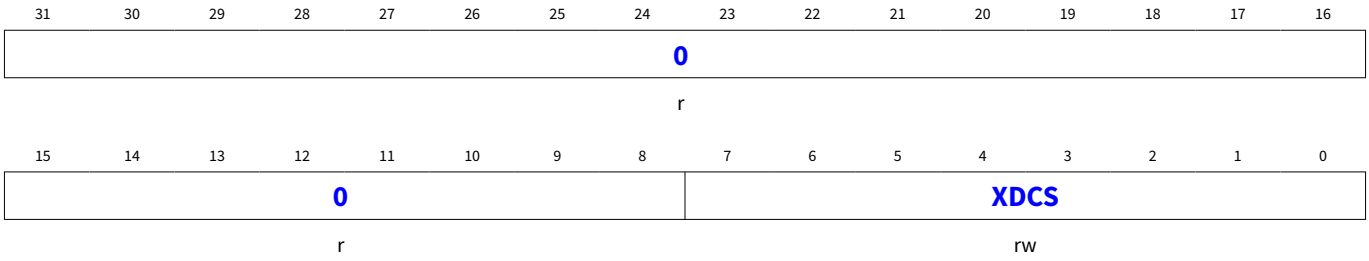
Field	Bits	Type	Description
XDCS	7:0	rw	**DMA Channel Select** This field contains one hot encoded DMA receive channel selection for MAC Address registers to forward the received MC/BC packet.
0	31:8	r	**Reserved** Read as 0; should be written with 0.

14.6.197 Port j MAC DChSel IndReg16

This register contains DCS (XDCS) one hot encoded value of receive channel selection for multicast/broadcast.

Portj_MAC_DChSel_IndReg16 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC DChSel IndReg16 Kernel Reset value: $0000\ 0000_H$



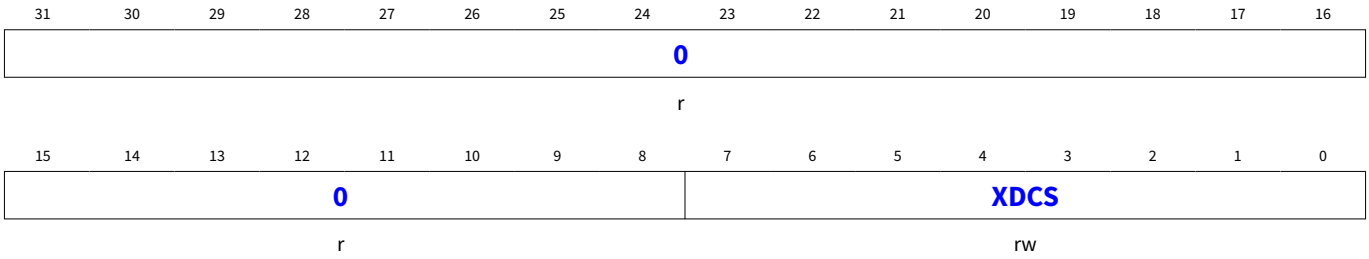
Field	Bits	Type	Description
XDCS	7:0	rw	**DMA Channel Select** This field contains one hot encoded DMA receive channel selection for MAC Address registers to forward the received MC/BC packet.
0	31:8	r	**Reserved** Read as 0; should be written with 0.

14.6.198 Port j MAC DChSel IndReg17

This register contains DCS (XDCS) one hot encoded value of receive channel selection for multicast/broadcast.

Portj_MAC_DChSel_IndReg17 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC DChSel IndReg17 Kernel Reset value: $0000\ 0000_H$



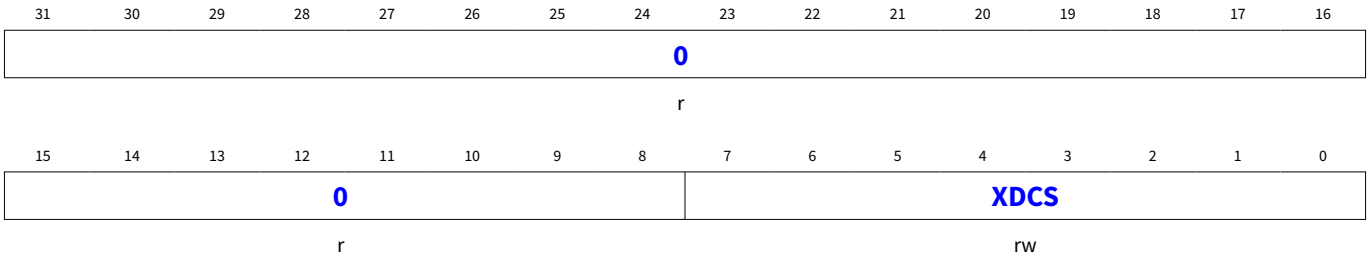
Field	Bits	Type	Description
XDCS	7:0	rw	**DMA Channel Select** This field contains one hot encoded DMA receive channel selection for MAC Address registers to forward the received MC/BC packet.
0	31:8	r	**Reserved** Read as 0; should be written with 0.

14.6.199 Port j MAC DChSel IndReg18

This register contains DCS (XDCS) one hot encoded value of receive channel selection for multicast/broadcast.

Portj_MAC_DChSel_IndReg18 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC DChSel IndReg18 Kernel Reset value: $0000\ 0000_H$



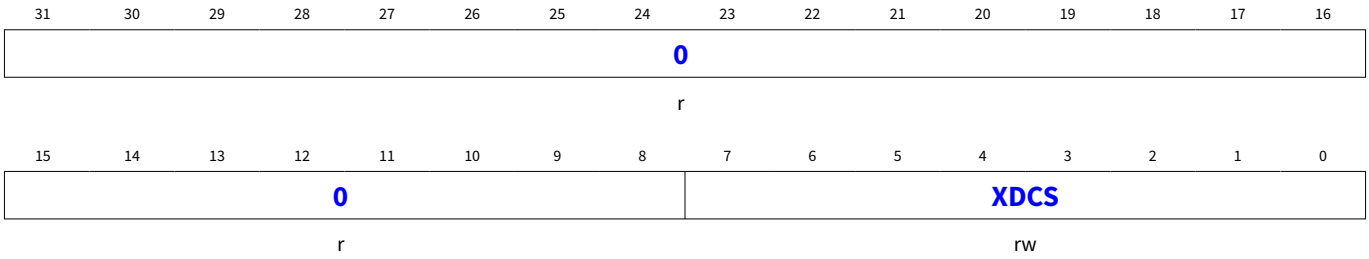
Field	Bits	Type	Description
XDCS	7:0	rw	**DMA Channel Select** This field contains one hot encoded DMA receive channel selection for MAC Address registers to forward the received MC/BC packet.
0	31:8	r	**Reserved** Read as 0; should be written with 0.

14.6.200 Port j MAC DChSel IndReg19

This register contains DCS (XDCS) one hot encoded value of receive channel selection for multicast/broadcast.

Portj_MAC_DChSel_IndReg19 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC DChSel IndReg19 Kernel Reset value: $0000\ 0000_H$



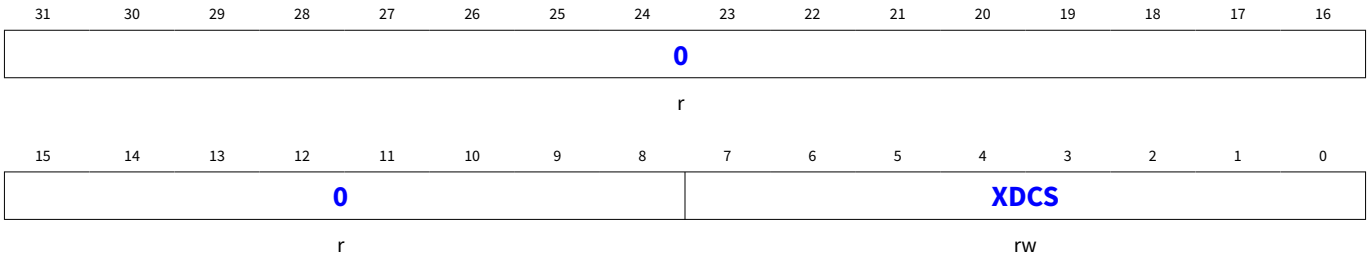
Field	Bits	Type	Description
XDCS	7:0	rw	**DMA Channel Select** This field contains one hot encoded DMA receive channel selection for MAC Address registers to forward the received MC/BC packet.
0	31:8	r	**Reserved** Read as 0; should be written with 0.

14.6.201 Port j MAC DChSel IndReg2

This register contains DCS (XDCS) one hot encoded value of receive channel selection for multicast/broadcast.

Portj_MAC_DChSel_IndReg2 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC DChSel IndReg2 Kernel Reset value: $0000\ 0000_H$



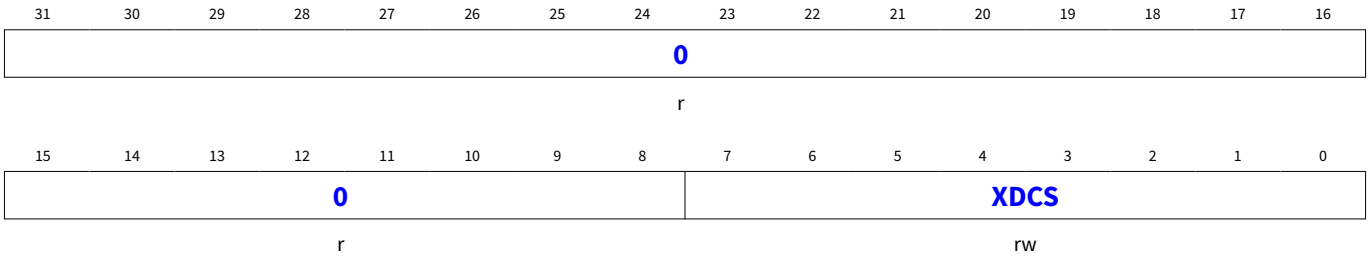
Field	Bits	Type	Description
XDCS	7:0	rw	**DMA Channel Select** This field contains one hot encoded DMA receive channel selection for MAC Address registers to forward the received MC/BC packet.
0	31:8	r	**Reserved** Read as 0; should be written with 0.

14.6.202 Port j MAC DChSel IndReg20

This register contains DCS (XDCS) one hot encoded value of receive channel selection for multicast/broadcast.

Portj_MAC_DChSel_IndReg20 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC DChSel IndReg20 Kernel Reset value: $0000\ 0000_H$



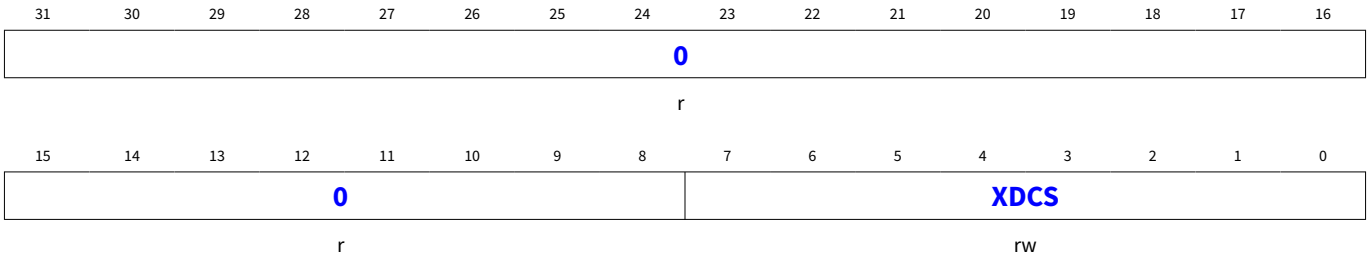
Field	Bits	Type	Description
XDCS	7:0	rw	**DMA Channel Select** This field contains one hot encoded DMA receive channel selection for MAC Address registers to forward the received MC/BC packet.
0	31:8	r	**Reserved** Read as 0; should be written with 0.

14.6.203 Port j MAC DChSel IndReg21

This register contains DCS (XDCS) one hot encoded value of receive channel selection for multicast/broadcast.

Portj_MAC_DChSel_IndReg21 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC DChSel IndReg21 Kernel Reset value: $0000\ 0000_H$



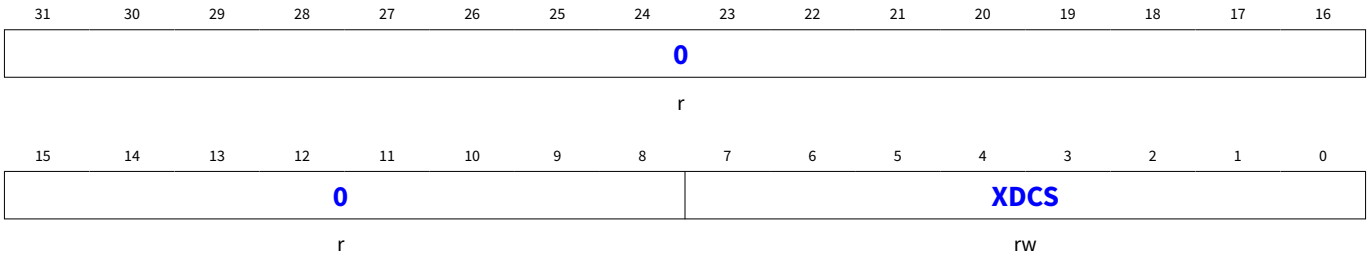
Field	Bits	Type	Description
XDCS	7:0	rw	**DMA Channel Select** This field contains one hot encoded DMA receive channel selection for MAC Address registers to forward the received MC/BC packet.
0	31:8	r	**Reserved** Read as 0; should be written with 0.

14.6.204 Port j MAC DChSel IndReg22

This register contains DCS (XDCS) one hot encoded value of receive channel selection for multicast/broadcast.

Portj_MAC_DChSel_IndReg22 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC DChSel IndReg22 Kernel Reset value: $0000\ 0000_H$



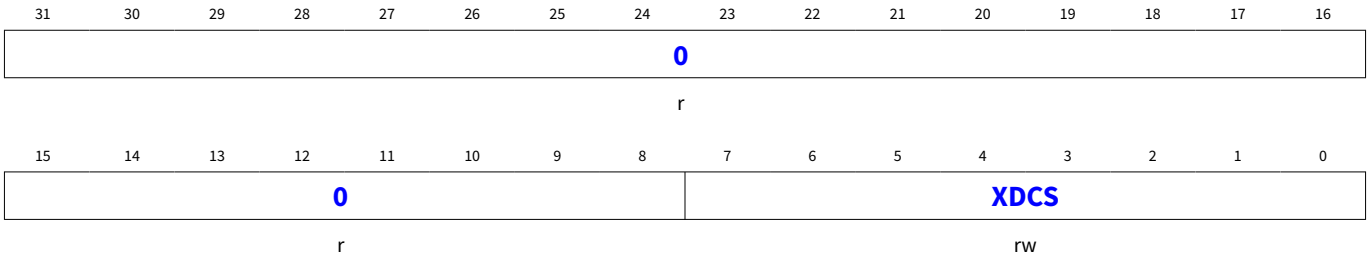
Field	Bits	Type	Description
XDCS	7:0	rw	**DMA Channel Select** This field contains one hot encoded DMA receive channel selection for MAC Address registers to forward the received MC/BC packet.
0	31:8	r	**Reserved** Read as 0; should be written with 0.

14.6.205 Port j MAC DChSel IndReg23

This register contains DCS (XDCS) one hot encoded value of receive channel selection for multicast/broadcast.

Portj_MAC_DChSel_IndReg23 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC DChSel IndReg23 Kernel Reset value: $0000\ 0000_H$



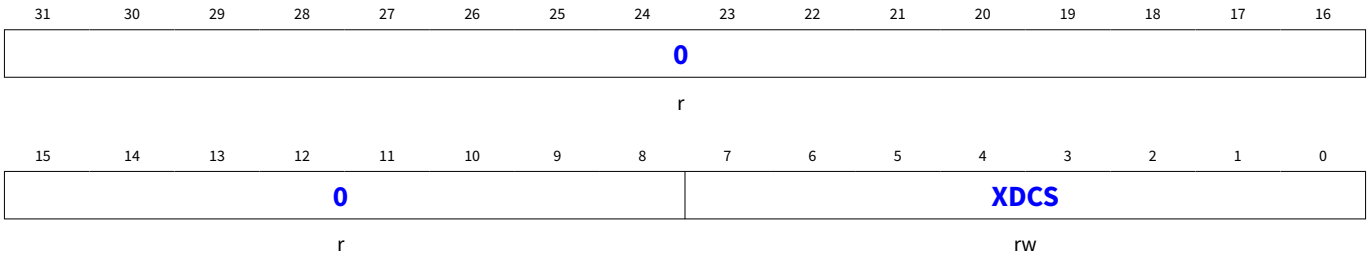
Field	Bits	Type	Description
XDCS	7:0	rw	**DMA Channel Select** This field contains one hot encoded DMA receive channel selection for MAC Address registers to forward the received MC/BC packet.
0	31:8	r	**Reserved** Read as 0; should be written with 0.

14.6.206 Port j MAC DChSel IndReg24

This register contains DCS (XDCS) one hot encoded value of receive channel selection for multicast/broadcast.

Portj_MAC_DChSel_IndReg24 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC DChSel IndReg24 Kernel Reset value: $0000\ 0000_H$



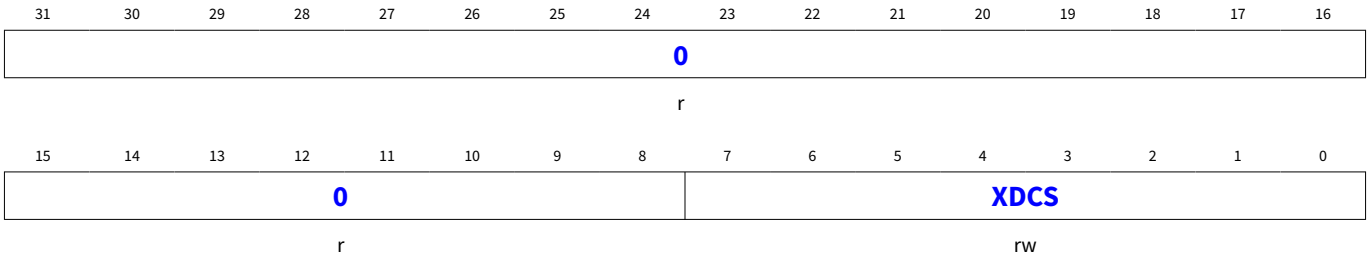
Field	Bits	Type	Description
XDCS	7:0	rw	**DMA Channel Select** This field contains one hot encoded DMA receive channel selection for MAC Address registers to forward the received MC/BC packet.
0	31:8	r	**Reserved** Read as 0; should be written with 0.

14.6.207 Port j MAC DChSel IndReg25

This register contains DCS (XDCS) one hot encoded value of receive channel selection for multicast/broadcast.

Portj_MAC_DChSel_IndReg25 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC DChSel IndReg25 Kernel Reset value: $0000\ 0000_H$



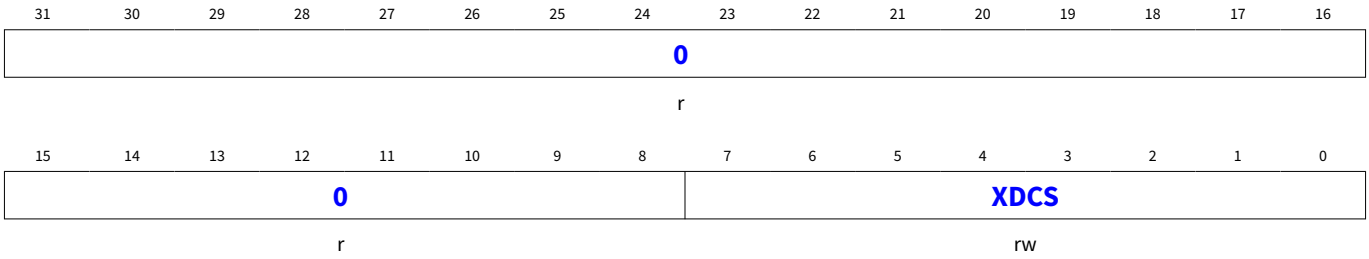
Field	Bits	Type	Description
XDCS	7:0	rw	**DMA Channel Select** This field contains one hot encoded DMA receive channel selection for MAC Address registers to forward the received MC/BC packet.
0	31:8	r	**Reserved** Read as 0; should be written with 0.

14.6.208 Port j MAC DChSel IndReg26

This register contains DCS (XDCS) one hot encoded value of receive channel selection for multicast/broadcast.

Portj_MAC_DChSel_IndReg26 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC DChSel IndReg26 Kernel Reset value: $0000\ 0000_H$



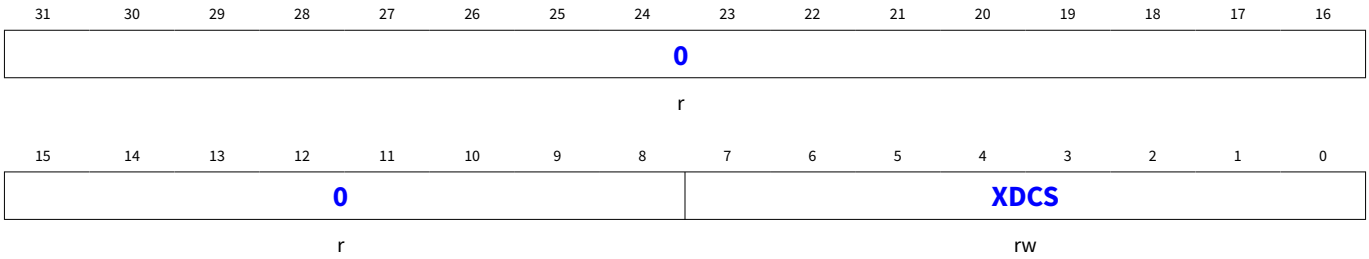
Field	Bits	Type	Description
XDCS	7:0	rw	**DMA Channel Select** This field contains one hot encoded DMA receive channel selection for MAC Address registers to forward the received MC/BC packet.
0	31:8	r	**Reserved** Read as 0; should be written with 0.

14.6.209 Port j MAC DChSel IndReg27

This register contains DCS (XDCS) one hot encoded value of receive channel selection for multicast/broadcast.

Portj_MAC_DChSel_IndReg27 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC DChSel IndReg27 Kernel Reset value: $0000\ 0000_H$



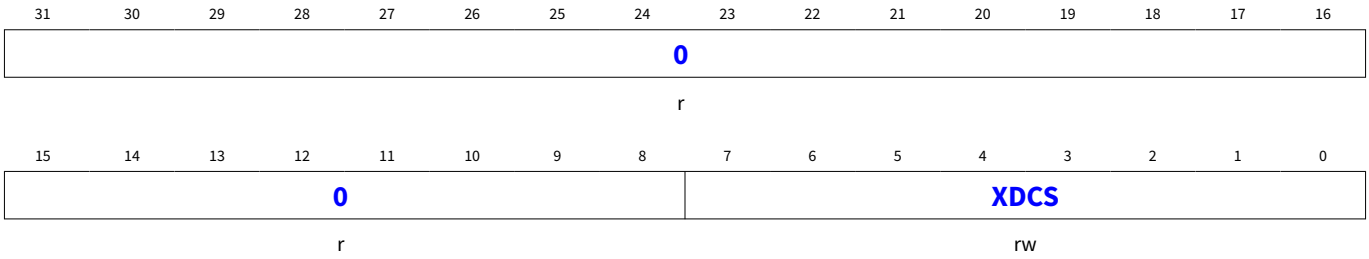
Field	Bits	Type	Description
XDCS	7:0	rw	**DMA Channel Select** This field contains one hot encoded DMA receive channel selection for MAC Address registers to forward the received MC/BC packet.
0	31:8	r	**Reserved** Read as 0; should be written with 0.

14.6.210 Port j MAC DChSel IndReg28

This register contains DCS (XDCS) one hot encoded value of receive channel selection for multicast/broadcast.

Portj_MAC_DChSel_IndReg28 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC DChSel IndReg28 Kernel Reset value: $0000\ 0000_H$



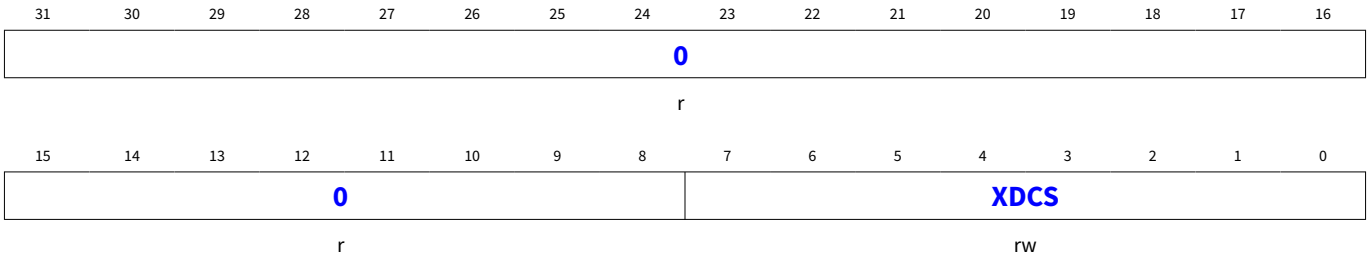
Field	Bits	Type	Description
XDCS	7:0	rw	**DMA Channel Select** This field contains one hot encoded DMA receive channel selection for MAC Address registers to forward the received MC/BC packet.
0	31:8	r	**Reserved** Read as 0; should be written with 0.

14.6.211 Port j MAC DChSel IndReg29

This register contains DCS (XDCS) one hot encoded value of receive channel selection for multicast/broadcast.

Portj_MAC_DChSel_IndReg29 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC DChSel IndReg29 Kernel Reset value: $0000\ 0000_H$



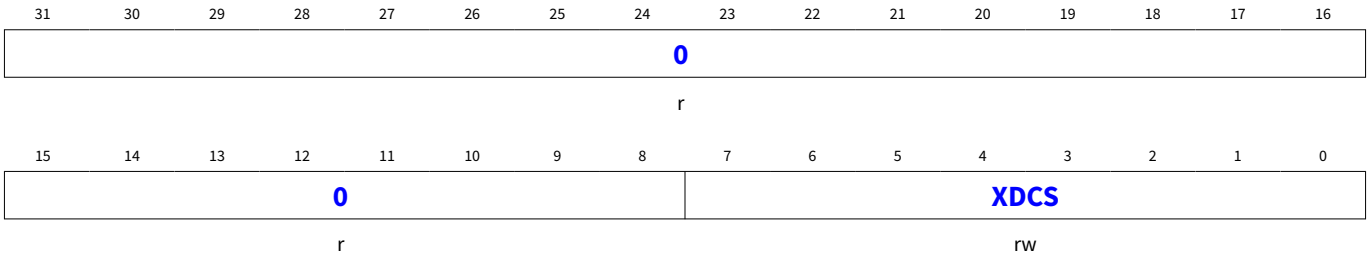
Field	Bits	Type	Description
XDCS	7:0	rw	**DMA Channel Select** This field contains one hot encoded DMA receive channel selection for MAC Address registers to forward the received MC/BC packet.
0	31:8	r	**Reserved** Read as 0; should be written with 0.

14.6.212 Port j MAC DChSel IndReg3

This register contains DCS (XDCS) one hot encoded value of receive channel selection for multicast/broadcast.

Portj_MAC_DChSel_IndReg3 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC DChSel IndReg3 Kernel Reset value: $0000\ 0000_H$



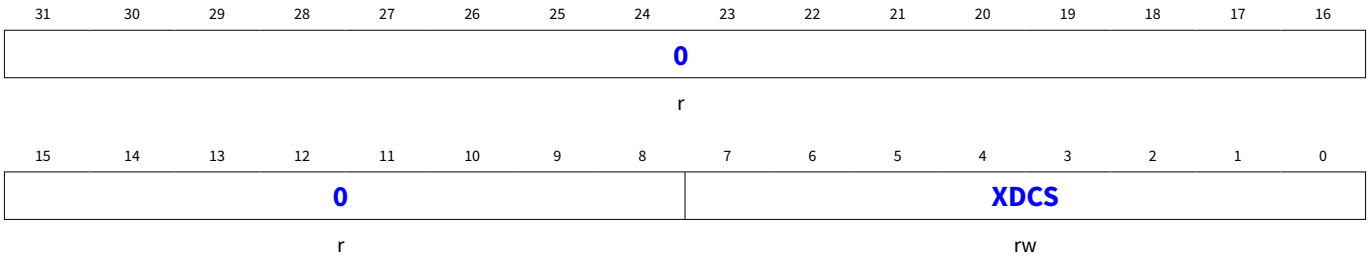
Field	Bits	Type	Description
XDCS	7:0	rw	**DMA Channel Select** This field contains one hot encoded DMA receive channel selection for MAC Address registers to forward the received MC/BC packet.
0	31:8	r	**Reserved** Read as 0; should be written with 0.

14.6.213 Port j MAC DChSel IndReg30

This register contains DCS (XDCS) one hot encoded value of receive channel selection for multicast/broadcast.

Portj_MAC_DChSel_IndReg30 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC DChSel IndReg30 Kernel Reset value: $0000\ 0000_H$



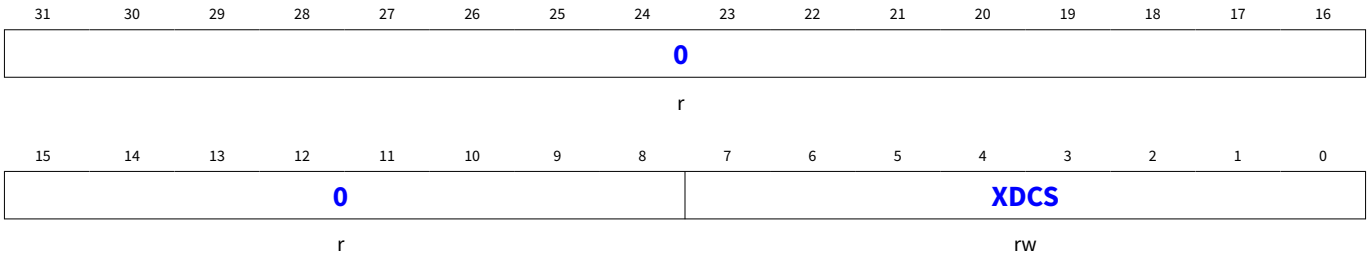
Field	Bits	Type	Description
XDCS	7:0	rw	**DMA Channel Select** This field contains one hot encoded DMA receive channel selection for MAC Address registers to forward the received MC/BC packet.
0	31:8	r	**Reserved** Read as 0; should be written with 0.

14.6.214 Port j MAC DChSel IndReg31

This register contains DCS (XDCS) one hot encoded value of receive channel selection for multicast/broadcast.

Portj_MAC_DChSel_IndReg31 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC DChSel IndReg31 Kernel Reset value: $0000\ 0000_H$



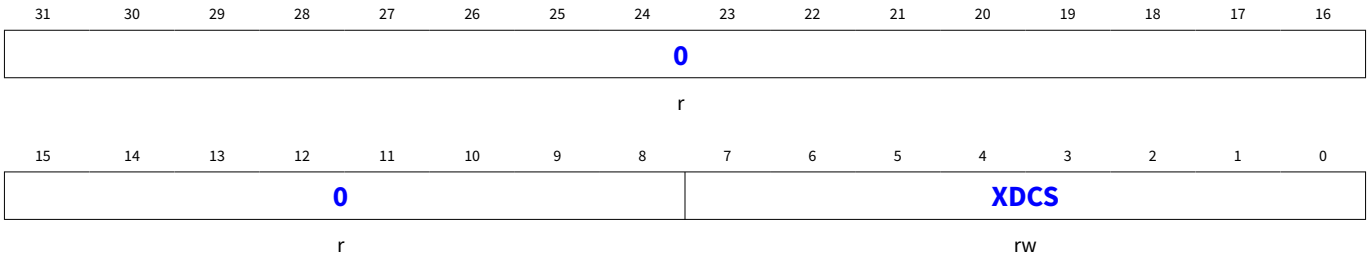
Field	Bits	Type	Description
XDCS	7:0	rw	**DMA Channel Select** This field contains one hot encoded DMA receive channel selection for MAC Address registers to forward the received MC/BC packet.
0	31:8	r	**Reserved** Read as 0; should be written with 0.

14.6.215 Port j MAC DChSel IndReg4

This register contains DCS (XDCS) one hot encoded value of receive channel selection for multicast/broadcast.

Portj_MAC_DChSel_IndReg4 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC DChSel IndReg4 Kernel Reset value: $0000\ 0000_H$



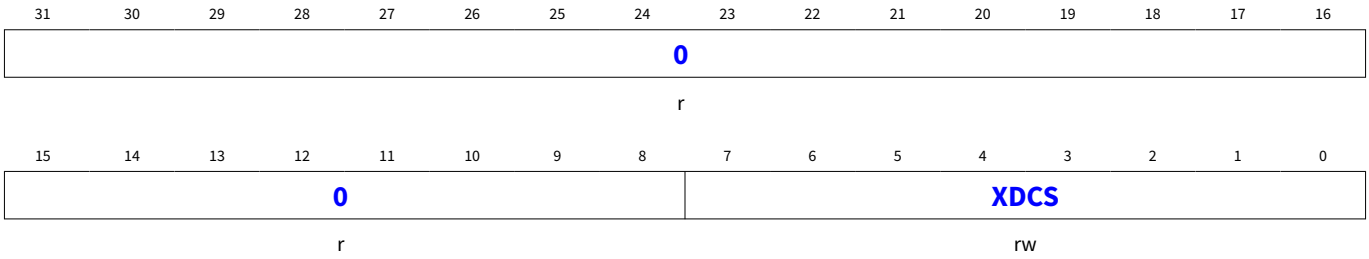
Field	Bits	Type	Description
XDCS	7:0	rw	**DMA Channel Select** This field contains one hot encoded DMA receive channel selection for MAC Address registers to forward the received MC/BC packet.
0	31:8	r	**Reserved** Read as 0; should be written with 0.

14.6.216 Port j MAC DChSel IndReg5

This register contains DCS (XDCS) one hot encoded value of receive channel selection for multicast/broadcast.

Portj_MAC_DChSel_IndReg5 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC DChSel IndReg5 Kernel Reset value: $0000\ 0000_H$



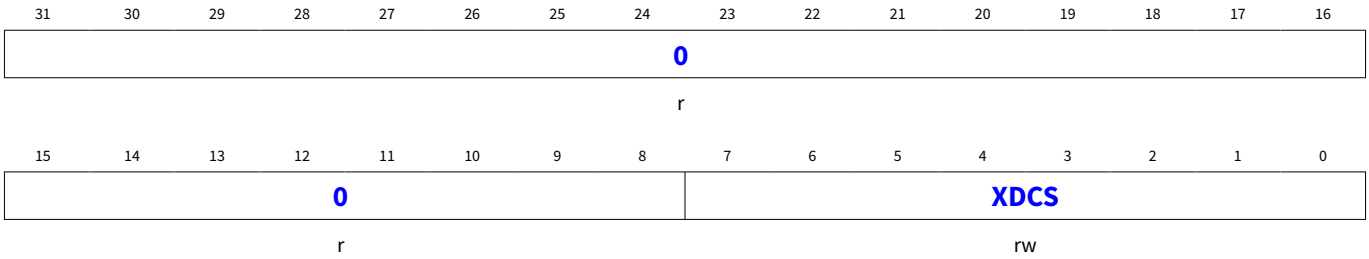
Field	Bits	Type	Description
XDCS	7:0	rw	**DMA Channel Select** This field contains one hot encoded DMA receive channel selection for MAC Address registers to forward the received MC/BC packet.
0	31:8	r	**Reserved** Read as 0; should be written with 0.

14.6.217 Port j MAC DChSel IndReg6

This register contains DCS (XDCS) one hot encoded value of receive channel selection for multicast/broadcast.

Portj_MAC_DChSel_IndReg6 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC DChSel IndReg6 Kernel Reset value: $0000\ 0000_H$



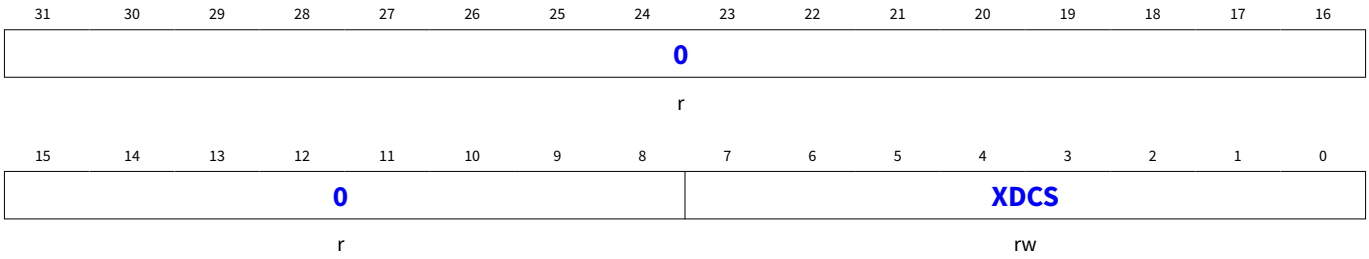
Field	Bits	Type	Description
XDCS	7:0	rw	**DMA Channel Select** This field contains one hot encoded DMA receive channel selection for MAC Address registers to forward the received MC/BC packet.
0	31:8	r	**Reserved** Read as 0; should be written with 0.

14.6.218 Port j MAC DChSel IndReg7

This register contains DCS (XDCS) one hot encoded value of receive channel selection for multicast/broadcast.

Portj_MAC_DChSel_IndReg7 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC DChSel IndReg7 Kernel Reset value: $0000\ 0000_H$



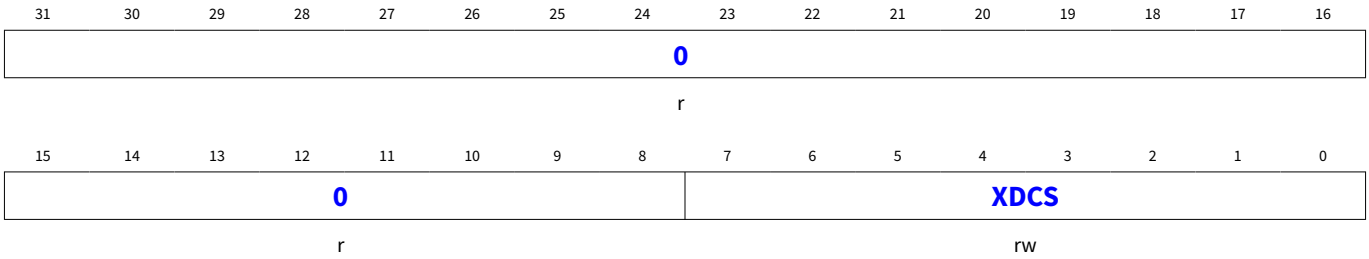
Field	Bits	Type	Description
XDCS	7:0	rw	**DMA Channel Select** This field contains one hot encoded DMA receive channel selection for MAC Address registers to forward the received MC/BC packet.
0	31:8	r	**Reserved** Read as 0; should be written with 0.

14.6.219 Port j MAC DChSel IndReg8

This register contains DCS (XDCS) one hot encoded value of receive channel selection for multicast/broadcast.

Portj_MAC_DChSel_IndReg8 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC DChSel IndReg8 Kernel Reset value: $0000\ 0000_H$



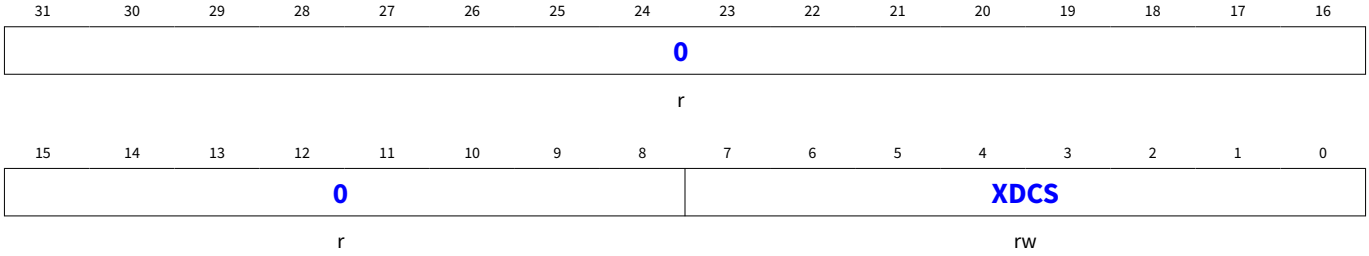
Field	Bits	Type	Description
XDCS	7:0	rw	**DMA Channel Select** This field contains one hot encoded DMA receive channel selection for MAC Address registers to forward the received MC/BC packet.
0	31:8	r	**Reserved** Read as 0; should be written with 0.

14.6.220 Port j MAC DChSel IndReg9

This register contains DCS (XDCS) one hot encoded value of receive channel selection for multicast/broadcast.

Portj_MAC_DChSel_IndReg9 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC DChSel IndReg9 Kernel Reset value: $0000\ 0000_H$



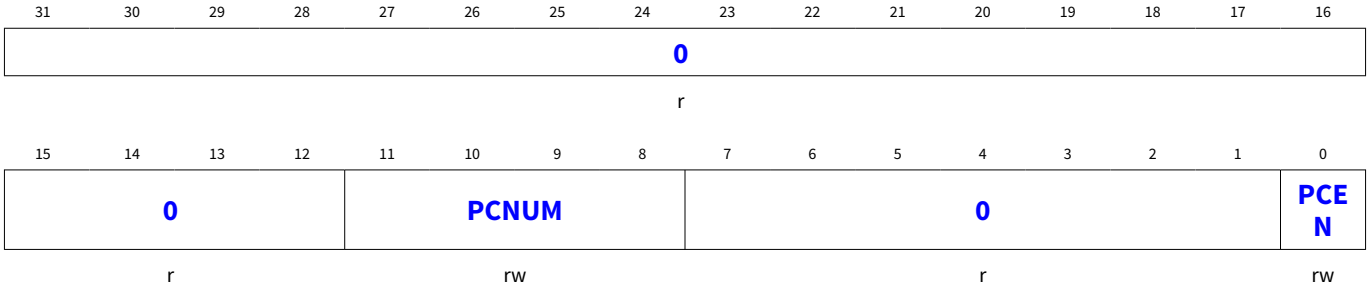
Field	Bits	Type	Description
XDCS	7:0	rw	**DMA Channel Select** This field contains one hot encoded DMA receive channel selection for MAC Address registers to forward the received MC/BC packet.
0	31:8	r	**Reserved** Read as 0; should be written with 0.

14.6.221 Port j MAC DPCSel IndReg0

This register contains policing counter number corresponding to each DA or SA filter event.

Portj_MAC_DPCSel_IndReg0 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC DPCSel IndReg0 Kernel Reset value: $0000\ 0000_H$



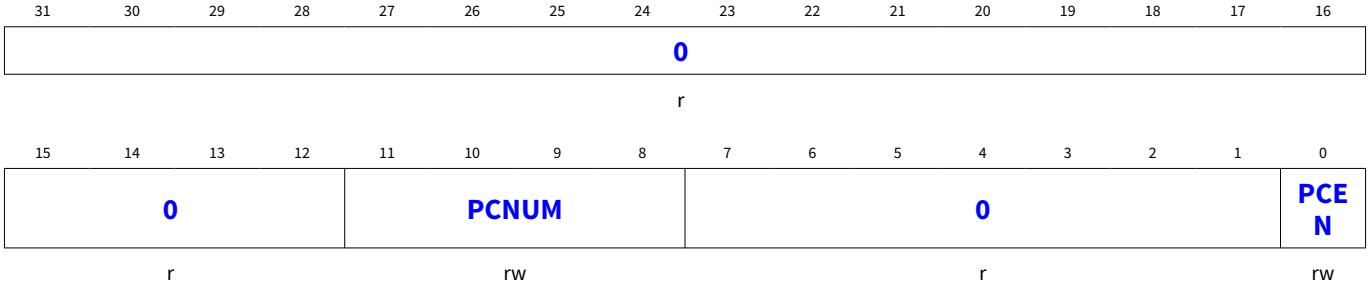
Field	Bits	Type	Description
PCEN	0	rw	**Policing Counter Enable** When set enables the policing counter indicated by the PCNUM field
PCNUM	11:8	rw	**Policing Counter Number** This field specifies the counter number selected for corresponding DA/SA address filter event.
0	7:1, 31:12	r	**Reserved** Read as 0; should be written with 0.

14.6.222 Port j MAC DPCSel IndReg1

This register contains policing counter number corresponding to each DA or SA filter event.

Portj_MAC_DPCSel_IndReg1 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC DPCSel IndReg1 Kernel Reset value: $0000\ 0000_H$



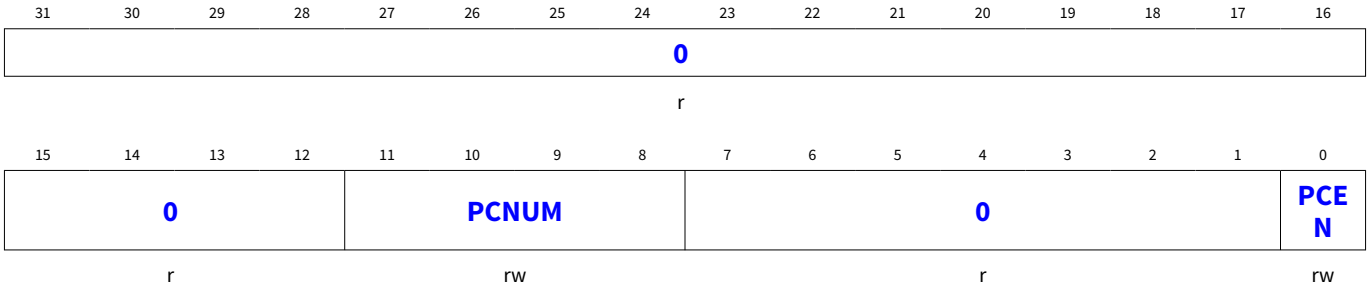
Field	Bits	Type	Description
PCEN	0	rw	**Policing Counter Enable** When set enables the policing counter indicated by the PCNUM field
PCNUM	11:8	rw	**Policing Counter Number** This field specifies the counter number selected for corresponding DA/SA address filter event.
0	7:1, 31:12	r	**Reserved** Read as 0; should be written with 0.

14.6.223 Port j MAC DPCSel IndReg10

This register contains policing counter number corresponding to each DA or SA filter event.

Portj_MAC_DPCSel_IndReg10 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC DPCSel IndReg10 Kernel Reset value: $0000\ 0000_H$



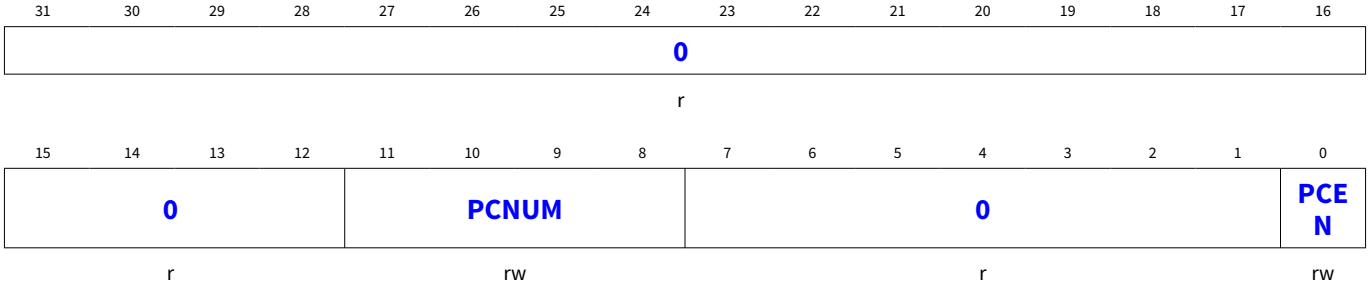
Field	Bits	Type	Description
PCEN	0	rw	**Policing Counter Enable** When set enables the policing counter indicated by the PCNUM field
PCNUM	11:8	rw	**Policing Counter Number** This field specifies the counter number selected for corresponding DA/SA address filter event.
0	7:1, 31:12	r	**Reserved** Read as 0; should be written with 0.

14.6.224 Port j MAC DPCSel IndReg11

This register contains policing counter number corresponding to each DA or SA filter event.

Portj_MAC_DPCSel_IndReg11 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC DPCSel IndReg11 Kernel Reset value: $0000\ 0000_H$



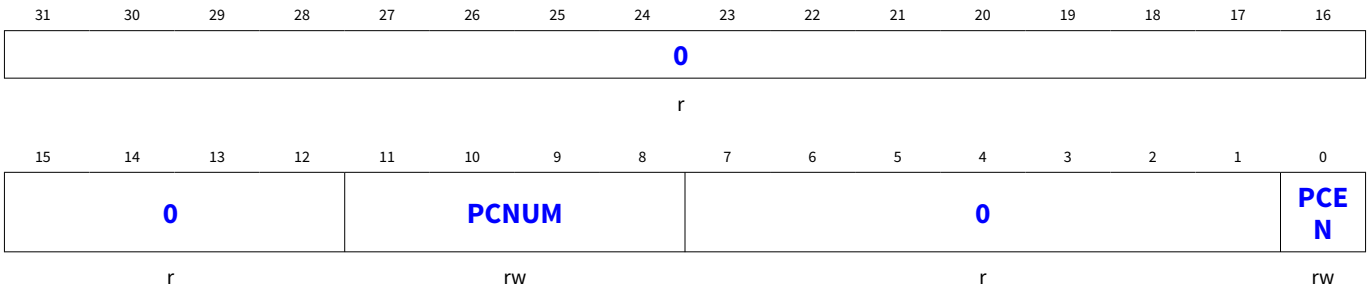
Field	Bits	Type	Description
PCEN	0	rw	**Policing Counter Enable** When set enables the policing counter indicated by the PCNUM field
PCNUM	11:8	rw	**Policing Counter Number** This field specifies the counter number selected for corresponding DA/SA address filter event.
0	7:1, 31:12	r	**Reserved** Read as 0; should be written with 0.

14.6.225 Port j MAC DPCSel IndReg12

This register contains policing counter number corresponding to each DA or SA filter event.

Portj_MAC_DPCSel_IndReg12 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC DPCSel IndReg12 Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
PCEN	0	rw	**Policing Counter Enable** When set enables the policing counter indicated by the PCNUM field
PCNUM	11:8	rw	**Policing Counter Number** This field specifies the counter number selected for corresponding DA/SA address filter event.

(table continues...)

(continued)

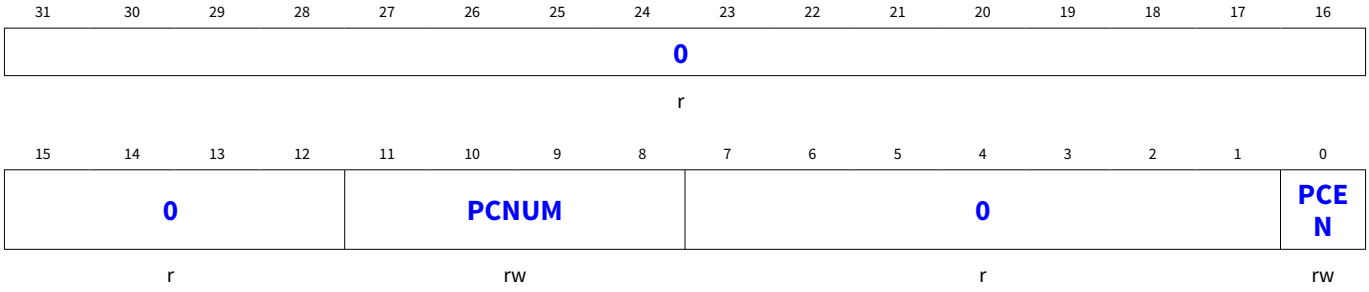
Field	Bits	Type	Description
0	7:1, 31:12	r	**Reserved** Read as 0; should be written with 0.

14.6.226 Port j MAC DPCSel IndReg13

This register contains policing counter number corresponding to each DA or SA filter event.

Portj_MAC_DPCSel_IndReg13 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC DPCSel IndReg13 Kernel Reset value: $0000\ 0000_H$



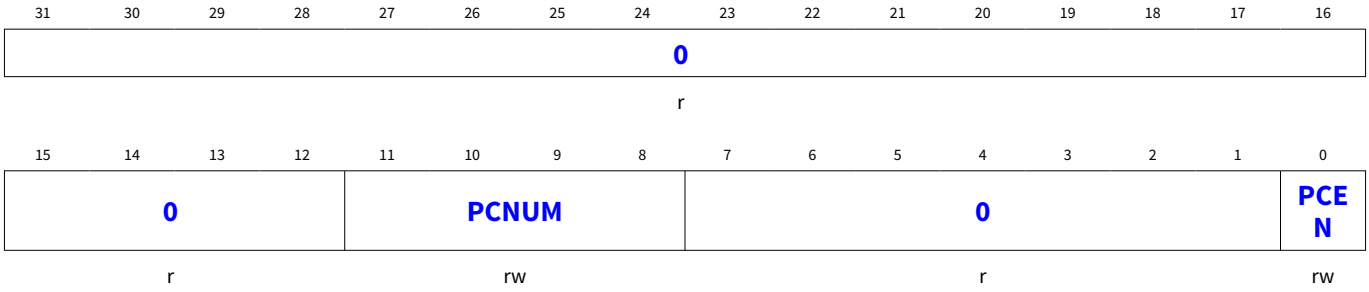
Field	Bits	Type	Description
PCEN	0	rw	**Policing Counter Enable** When set enables the policing counter indicated by the PCNUM field
PCNUM	11:8	rw	**Policing Counter Number** This field specifies the counter number selected for corresponding DA/SA address filter event.
0	7:1, 31:12	r	**Reserved** Read as 0; should be written with 0.

14.6.227 Port j MAC DPCSel IndReg14

This register contains policing counter number corresponding to each DA or SA filter event.

Portj_MAC_DPCSel_IndReg14 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC DPCSel IndReg14 Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
PCEN	0	rw	**Policing Counter Enable** When set enables the policing counter indicated by the PCNUM field
PCNUM	11:8	rw	**Policing Counter Number** This field specifies the counter number selected for corresponding DA/SA address filter event.
0	7:1, 31:12	r	**Reserved** Read as 0; should be written with 0.

14.6.228 Port j MAC DPCSel IndReg15

This register contains policing counter number corresponding to each DA or SA filter event.

Portj_MAC_DPCSel_IndReg15 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC DPCSel IndReg15 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				PCNUM				0							PCEN
r				rw				r							rw

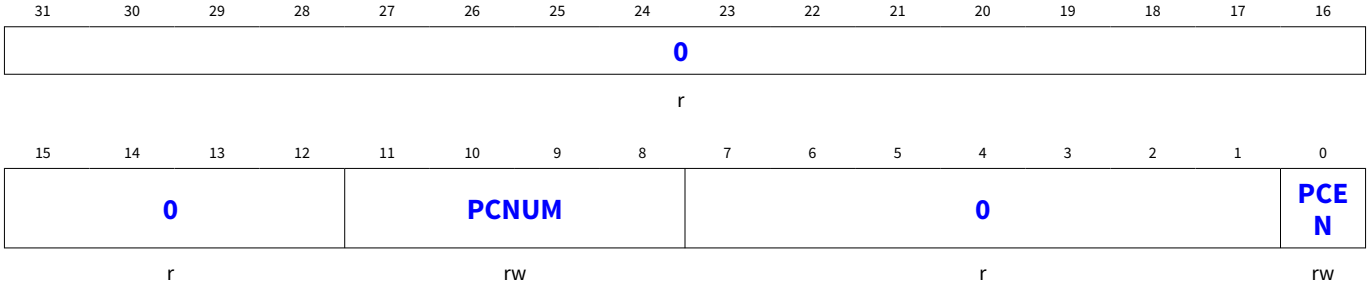
Field	Bits	Type	Description
PCEN	0	rw	**Policing Counter Enable** When set enables the policing counter indicated by the PCNUM field
PCNUM	11:8	rw	**Policing Counter Number** This field specifies the counter number selected for corresponding DA/SA address filter event.
0	7:1, 31:12	r	**Reserved** Read as 0; should be written with 0.

14.6.229 Port j MAC DPCSel IndReg16

This register contains policing counter number corresponding to each DA or SA filter event.

Portj_MAC_DPCSel_IndReg16 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC DPCSel IndReg16 Kernel Reset value: $0000\ 0000_H$



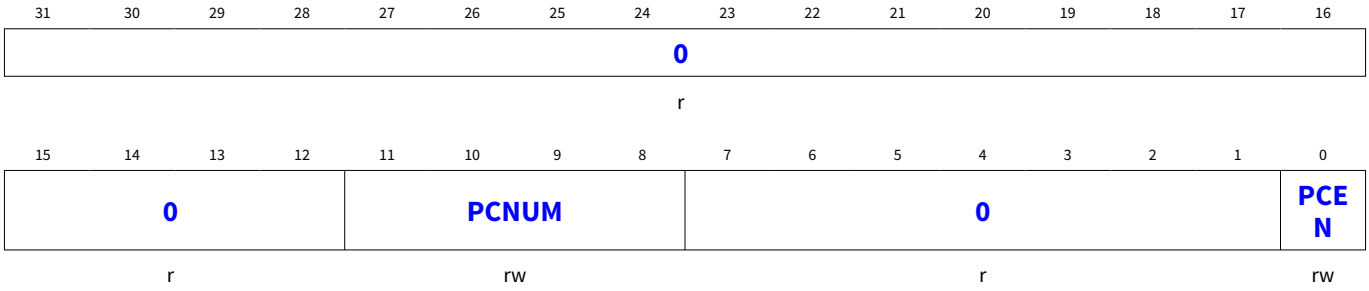
Field	Bits	Type	Description
PCEN	0	rw	**Policing Counter Enable** When set enables the policing counter indicated by the PCNUM field
PCNUM	11:8	rw	**Policing Counter Number** This field specifies the counter number selected for corresponding DA/SA address filter event.
0	7:1, 31:12	r	**Reserved** Read as 0; should be written with 0.

14.6.230 Port j MAC DPCSel IndReg17

This register contains policing counter number corresponding to each DA or SA filter event.

Portj_MAC_DPCSel_IndReg17 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC DPCSel IndReg17 Kernel Reset value: $0000\ 0000_H$



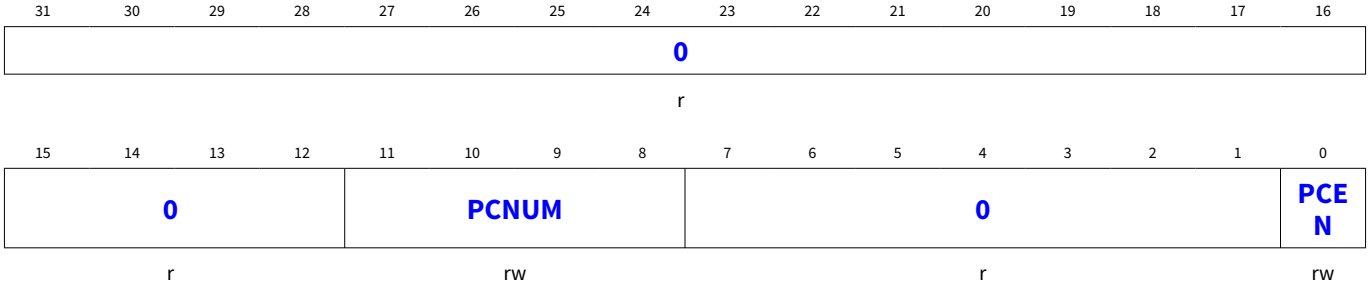
Field	Bits	Type	Description
PCEN	0	rw	**Policing Counter Enable** When set enables the policing counter indicated by the PCNUM field
PCNUM	11:8	rw	**Policing Counter Number** This field specifies the counter number selected for corresponding DA/SA address filter event.
0	7:1, 31:12	r	**Reserved** Read as 0; should be written with 0.

14.6.231 Port j MAC DPCSel IndReg18

This register contains policing counter number corresponding to each DA or SA filter event.

Portj_MAC_DPCSel_IndReg18 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC DPCSel IndReg18 Kernel Reset value: $0000\ 0000_H$



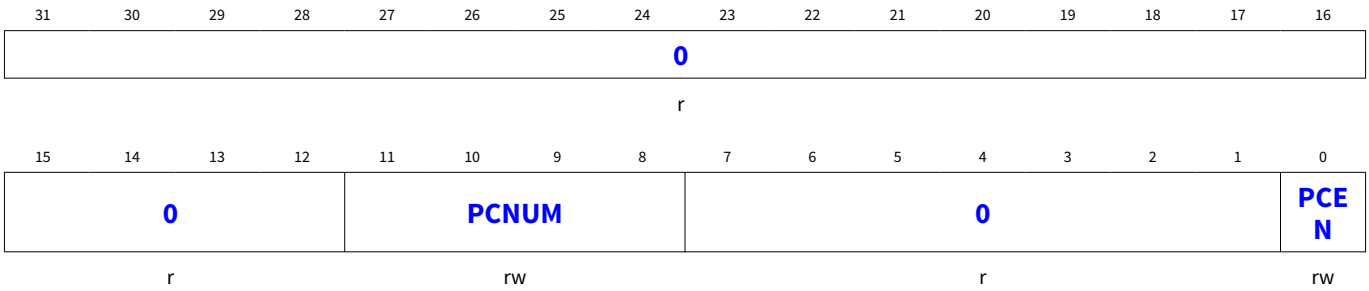
Field	Bits	Type	Description
PCEN	0	rw	**Policing Counter Enable** When set enables the policing counter indicated by the PCNUM field
PCNUM	11:8	rw	**Policing Counter Number** This field specifies the counter number selected for corresponding DA/SA address filter event.
0	7:1, 31:12	r	**Reserved** Read as 0; should be written with 0.

14.6.232 Port j MAC DPCSel IndReg19

This register contains policing counter number corresponding to each DA or SA filter event.

Portj_MAC_DPCSel_IndReg19 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC DPCSel IndReg19 Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
PCEN	0	rw	**Policing Counter Enable** When set enables the policing counter indicated by the PCNUM field
PCNUM	11:8	rw	**Policing Counter Number** This field specifies the counter number selected for corresponding DA/SA address filter event.

(table continues...)

(continued)

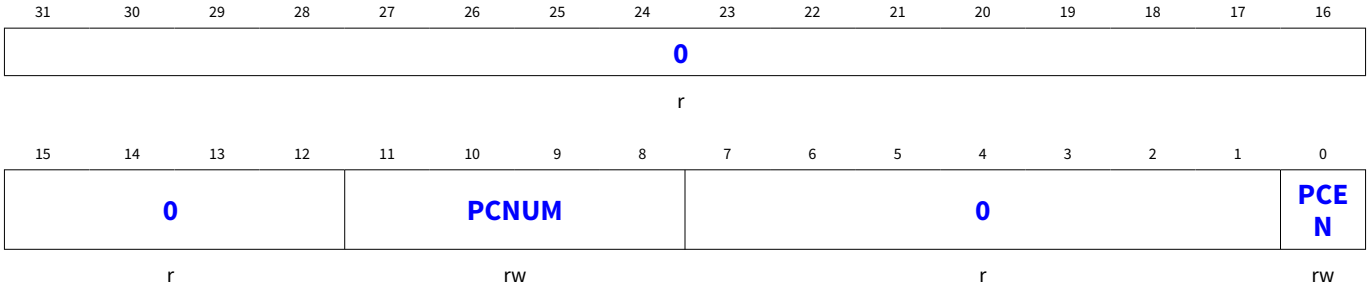
Field	Bits	Type	Description
0	7:1, 31:12	r	**Reserved** Read as 0; should be written with 0.

14.6.233 Port j MAC DPCSel IndReg2

This register contains policing counter number corresponding to each DA or SA filter event.

Portj_MAC_DPCSel_IndReg2 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC DPCSel IndReg2 Kernel Reset value: $0000\ 0000_H$



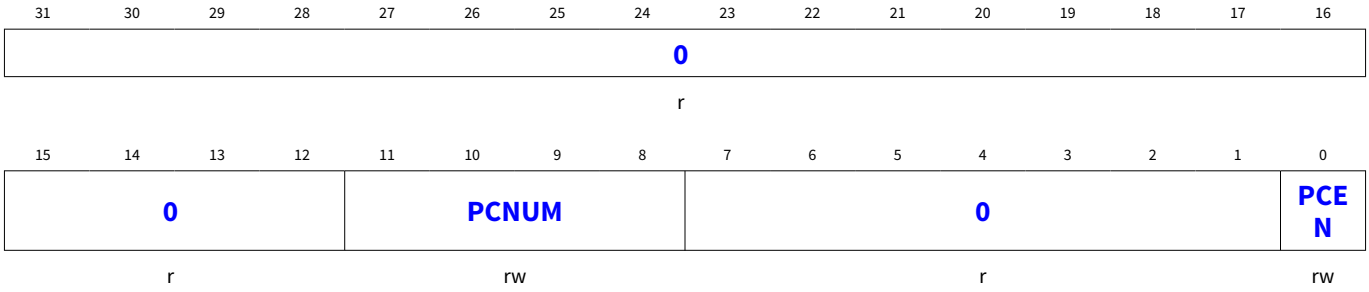
Field	Bits	Type	Description
PCEN	0	rw	**Policing Counter Enable** When set enables the policing counter indicated by the PCNUM field
PCNUM	11:8	rw	**Policing Counter Number** This field specifies the counter number selected for corresponding DA/SA address filter event.
0	7:1, 31:12	r	**Reserved** Read as 0; should be written with 0.

14.6.234 Port j MAC DPCSel IndReg20

This register contains policing counter number corresponding to each DA or SA filter event.

Portj_MAC_DPCSel_IndReg20 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC DPCSel IndReg20 Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
PCEN	0	rw	**Policing Counter Enable** When set enables the policing counter indicated by the PCNUM field
PCNUM	11:8	rw	**Policing Counter Number** This field specifies the counter number selected for corresponding DA/SA address filter event.
0	7:1, 31:12	r	**Reserved** Read as 0; should be written with 0.

14.6.235 Port j MAC DPCSel IndReg21

This register contains policing counter number corresponding to each DA or SA filter event.

Portj_MAC_DPCSel_IndReg21 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC DPCSel IndReg21 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				PCNUM				0							PCEN
r				rw				r							rw

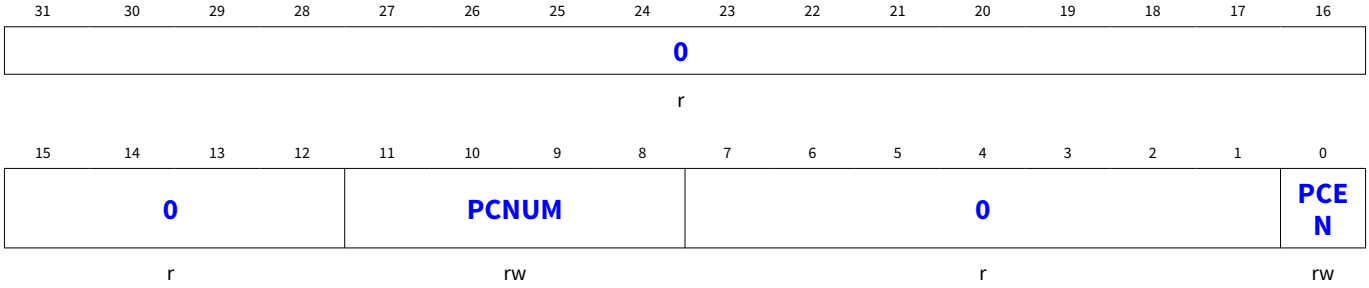
Field	Bits	Type	Description
PCEN	0	rw	**Policing Counter Enable** When set enables the policing counter indicated by the PCNUM field
PCNUM	11:8	rw	**Policing Counter Number** This field specifies the counter number selected for corresponding DA/SA address filter event.
0	7:1, 31:12	r	**Reserved** Read as 0; should be written with 0.

14.6.236 Port j MAC DPCSel IndReg22

This register contains policing counter number corresponding to each DA or SA filter event.

Portj_MAC_DPCSel_IndReg22 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC DPCSel IndReg22 Kernel Reset value: $0000\ 0000_H$



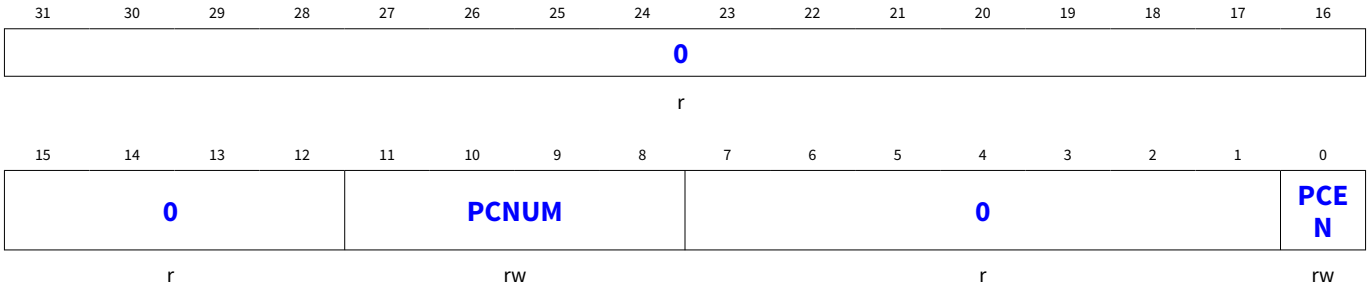
Field	Bits	Type	Description
PCEN	0	rw	**Policing Counter Enable** When set enables the policing counter indicated by the PCNUM field
PCNUM	11:8	rw	**Policing Counter Number** This field specifies the counter number selected for corresponding DA/SA address filter event.
0	7:1, 31:12	r	**Reserved** Read as 0; should be written with 0.

14.6.237 Port j MAC DPCSel IndReg23

This register contains policing counter number corresponding to each DA or SA filter event.

Portj_MAC_DPCSel_IndReg23 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC DPCSel IndReg23 Kernel Reset value: $0000\ 0000_H$



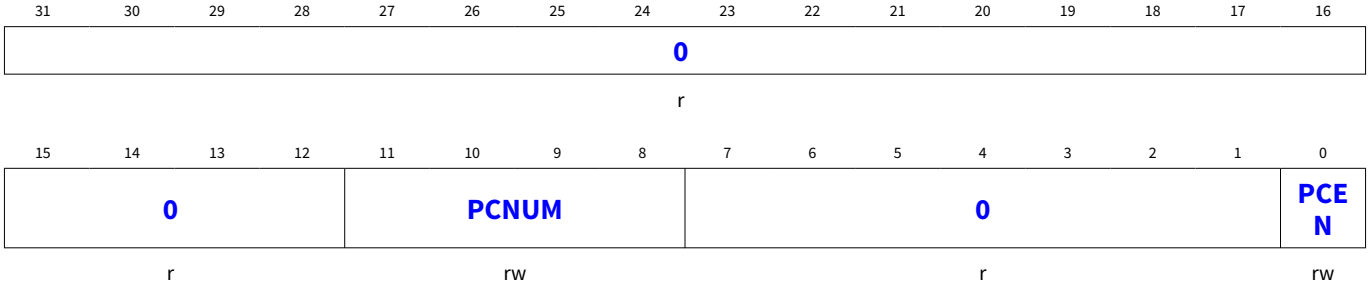
Field	Bits	Type	Description
PCEN	0	rw	**Policing Counter Enable** When set enables the policing counter indicated by the PCNUM field
PCNUM	11:8	rw	**Policing Counter Number** This field specifies the counter number selected for corresponding DA/SA address filter event.
0	7:1, 31:12	r	**Reserved** Read as 0; should be written with 0.

14.6.238 Port j MAC DPCSel IndReg24

This register contains policing counter number corresponding to each DA or SA filter event.

Portj_MAC_DPCSel_IndReg24 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC DPCSel IndReg24 Kernel Reset value: $0000\ 0000_H$



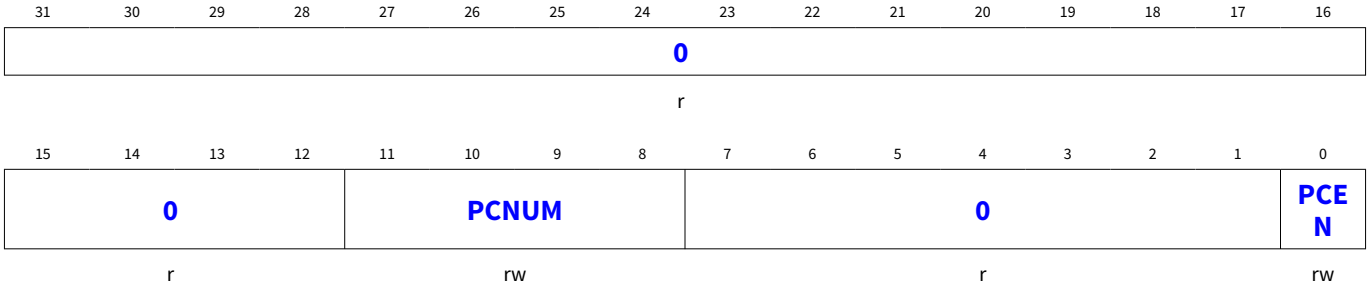
Field	Bits	Type	Description
PCEN	0	rw	**Policing Counter Enable** When set enables the policing counter indicated by the PCNUM field
PCNUM	11:8	rw	**Policing Counter Number** This field specifies the counter number selected for corresponding DA/SA address filter event.
0	7:1, 31:12	r	**Reserved** Read as 0; should be written with 0.

14.6.239 Port j MAC DPCSel IndReg25

This register contains policing counter number corresponding to each DA or SA filter event.

Portj_MAC_DPCSel_IndReg25 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC DPCSel IndReg25 Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
PCEN	0	rw	**Policing Counter Enable** When set enables the policing counter indicated by the PCNUM field
PCNUM	11:8	rw	**Policing Counter Number** This field specifies the counter number selected for corresponding DA/SA address filter event.

(table continues...)

(continued)

Field	Bits	Type	Description
0	7:1, 31:12	r	**Reserved** Read as 0; should be written with 0.

14.6.240 Port j MAC DPCSel IndReg26

This register contains policing counter number corresponding to each DA or SA filter event.

Portj_MAC_DPCSel_IndReg26 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC DPCSel IndReg26 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				PCNUM				0							PCEN
r				rw				r							rw

Field	Bits	Type	Description
PCEN	0	rw	**Policing Counter Enable** When set enables the policing counter indicated by the PCNUM field
PCNUM	11:8	rw	**Policing Counter Number** This field specifies the counter number selected for corresponding DA/SA address filter event.
0	7:1, 31:12	r	**Reserved** Read as 0; should be written with 0.

14.6.241 Port j MAC DPCSel IndReg27

This register contains policing counter number corresponding to each DA or SA filter event.

Portj_MAC_DPCSel_IndReg27 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC DPCSel IndReg27 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				PCNUM				0							PCEN
r				rw				r							rw

Field	Bits	Type	Description
PCEN	0	rw	**Policing Counter Enable** When set enables the policing counter indicated by the PCNUM field
PCNUM	11:8	rw	**Policing Counter Number** This field specifies the counter number selected for corresponding DA/SA address filter event.
0	7:1, 31:12	r	**Reserved** Read as 0; should be written with 0.

14.6.242 Port j MAC DPCSel IndReg28

This register contains policing counter number corresponding to each DA or SA filter event.

Portj_MAC_DPCSel_IndReg28 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC DPCSel IndReg28 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				PCNUM				0							PCEN
r				rw				r							rw

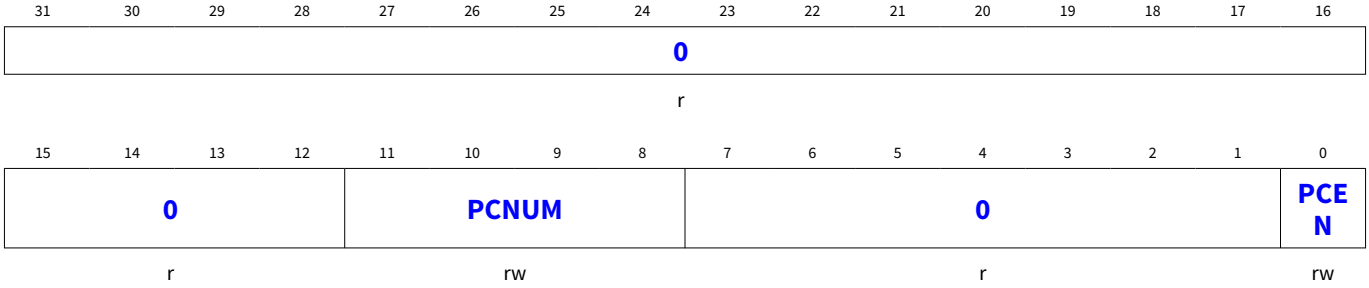
Field	Bits	Type	Description
PCEN	0	rw	**Policing Counter Enable** When set enables the policing counter indicated by the PCNUM field
PCNUM	11:8	rw	**Policing Counter Number** This field specifies the counter number selected for corresponding DA/SA address filter event.
0	7:1, 31:12	r	**Reserved** Read as 0; should be written with 0.

14.6.243 Port j MAC DPCSel IndReg29

This register contains policing counter number corresponding to each DA or SA filter event.

Portj_MAC_DPCSel_IndReg29 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC DPCSel IndReg29 Kernel Reset value: $0000\ 0000_H$



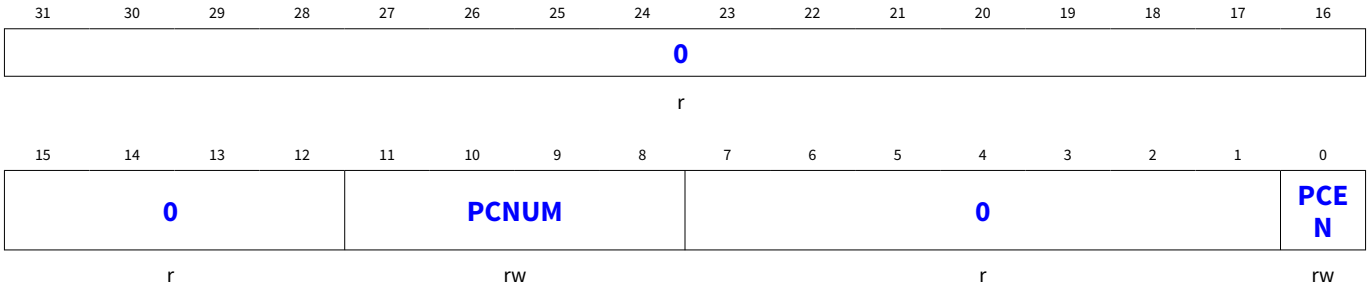
Field	Bits	Type	Description
PCEN	0	rw	**Policing Counter Enable** When set enables the policing counter indicated by the PCNUM field
PCNUM	11:8	rw	**Policing Counter Number** This field specifies the counter number selected for corresponding DA/SA address filter event.
0	7:1, 31:12	r	**Reserved** Read as 0; should be written with 0.

14.6.244 Port j MAC DPCSel IndReg3

This register contains policing counter number corresponding to each DA or SA filter event.

Portj_MAC_DPCSel_IndReg3 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC DPCSel IndReg3 Kernel Reset value: $0000\ 0000_H$



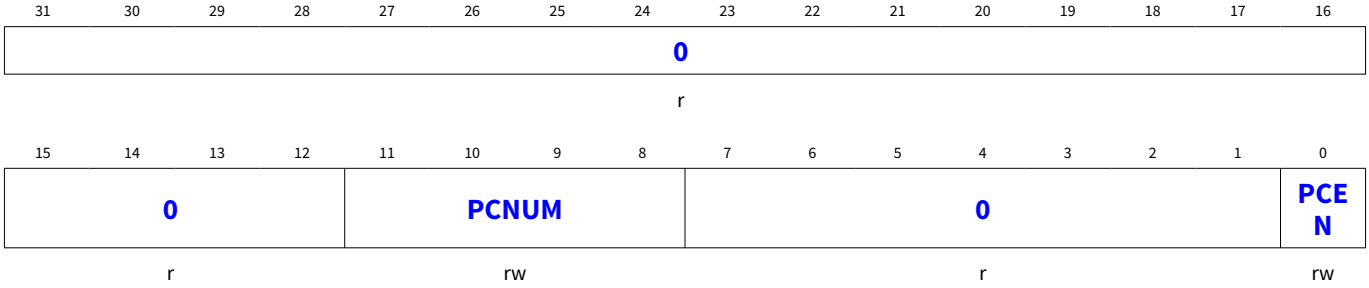
Field	Bits	Type	Description
PCEN	0	rw	**Policing Counter Enable** When set enables the policing counter indicated by the PCNUM field
PCNUM	11:8	rw	**Policing Counter Number** This field specifies the counter number selected for corresponding DA/SA address filter event.
0	7:1, 31:12	r	**Reserved** Read as 0; should be written with 0.

14.6.245 Port j MAC DPCSel IndReg30

This register contains policing counter number corresponding to each DA or SA filter event.

Portj_MAC_DPCSel_IndReg30 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC DPCSel IndReg30 Kernel Reset value: $0000\ 0000_H$



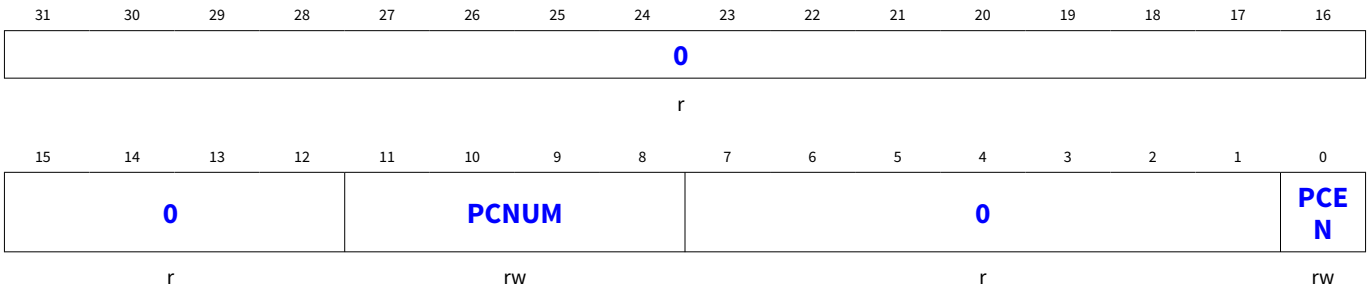
Field	Bits	Type	Description
PCEN	0	rw	**Policing Counter Enable** When set enables the policing counter indicated by the PCNUM field
PCNUM	11:8	rw	**Policing Counter Number** This field specifies the counter number selected for corresponding DA/SA address filter event.
0	7:1, 31:12	r	**Reserved** Read as 0; should be written with 0.

14.6.246 Port j MAC DPCSel IndReg31

This register contains policing counter number corresponding to each DA or SA filter event.

Portj_MAC_DPCSel_IndReg31 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC DPCSel IndReg31 Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
PCEN	0	rw	**Policing Counter Enable** When set enables the policing counter indicated by the PCNUM field
PCNUM	11:8	rw	**Policing Counter Number** This field specifies the counter number selected for corresponding DA/SA address filter event.

(table continues...)

(continued)

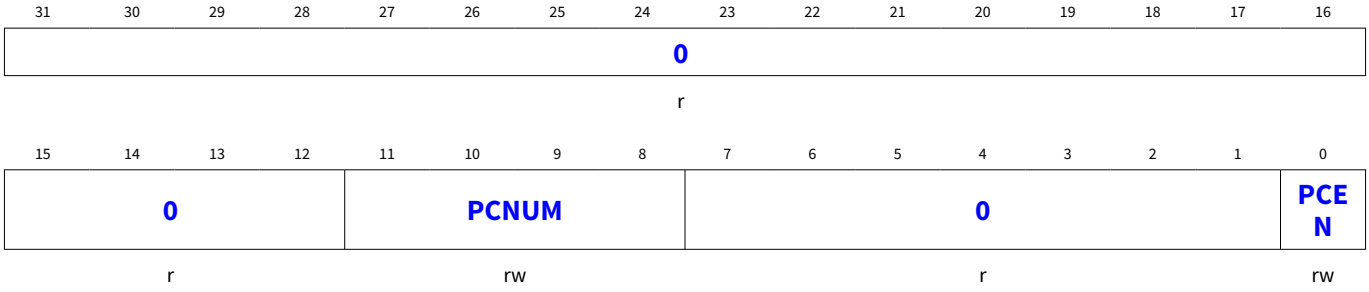
Field	Bits	Type	Description
0	7:1, 31:12	r	**Reserved** Read as 0; should be written with 0.

14.6.247 Port j MAC DPCSel IndReg4

This register contains policing counter number corresponding to each DA or SA filter event.

Portj_MAC_DPCSel_IndReg4 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC DPCSel IndReg4 Kernel Reset value: $0000\ 0000_H$



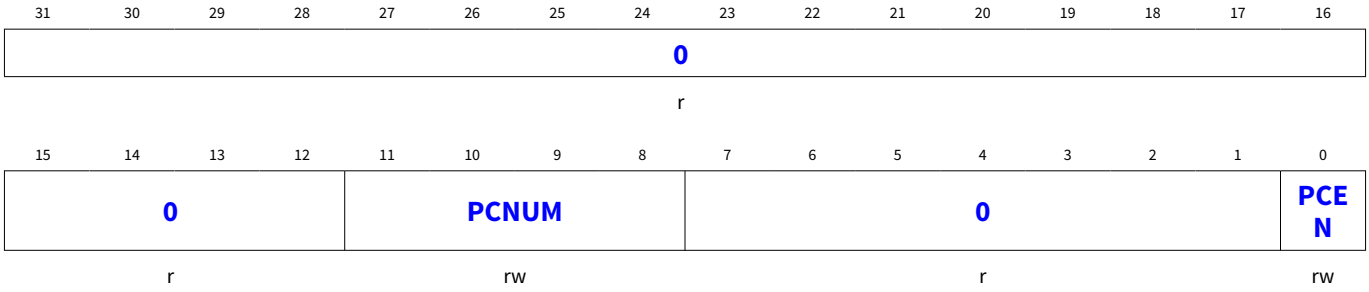
Field	Bits	Type	Description
PCEN	0	rw	**Policing Counter Enable** When set enables the policing counter indicated by the PCNUM field
PCNUM	11:8	rw	**Policing Counter Number** This field specifies the counter number selected for corresponding DA/SA address filter event.
0	7:1, 31:12	r	**Reserved** Read as 0; should be written with 0.

14.6.248 Port j MAC DPCSel IndReg5

This register contains policing counter number corresponding to each DA or SA filter event.

Portj_MAC_DPCSel_IndReg5 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC DPCSel IndReg5 Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
PCEN	0	rw	**Policing Counter Enable** When set enables the policing counter indicated by the PCNUM field
PCNUM	11:8	rw	**Policing Counter Number** This field specifies the counter number selected for corresponding DA/SA address filter event.
0	7:1, 31:12	r	**Reserved** Read as 0; should be written with 0.

14.6.249 Port j MAC DPCSel IndReg6

This register contains policing counter number corresponding to each DA or SA filter event.

Portj_MAC_DPCSel_IndReg6 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC DPCSel IndReg6 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				PCNUM				0							PCEN
r				rw				r							rw

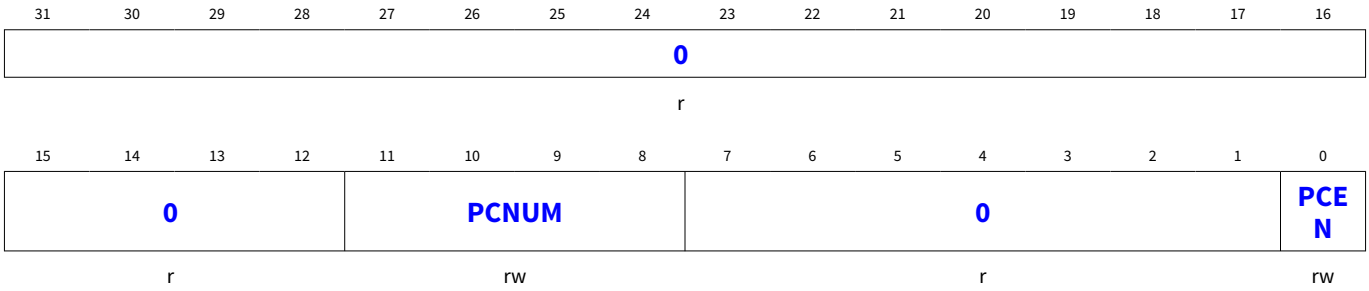
Field	Bits	Type	Description
PCEN	0	rw	**Policing Counter Enable** When set enables the policing counter indicated by the PCNUM field
PCNUM	11:8	rw	**Policing Counter Number** This field specifies the counter number selected for corresponding DA/SA address filter event.
0	7:1, 31:12	r	**Reserved** Read as 0; should be written with 0.

14.6.250 Port j MAC DPCSel IndReg7

This register contains policing counter number corresponding to each DA or SA filter event.

Portj_MAC_DPCSel_IndReg7 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC DPCSel IndReg7 Kernel Reset value: $0000\ 0000_H$



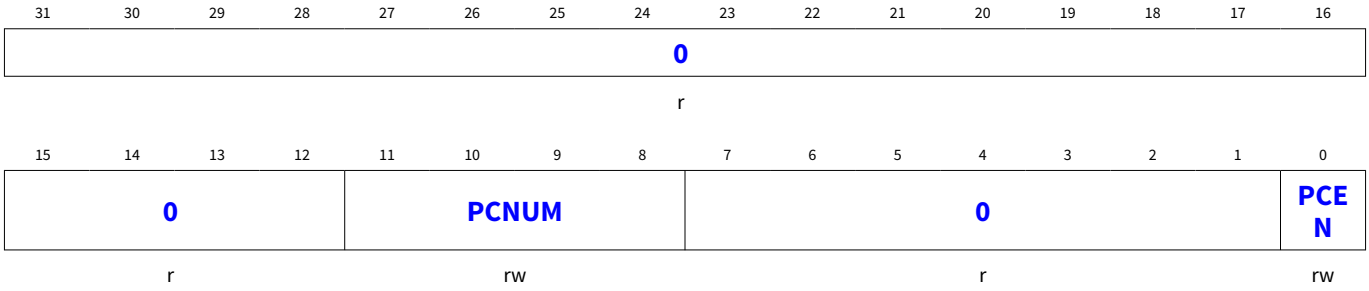
Field	Bits	Type	Description
PCEN	0	rw	**Policing Counter Enable** When set enables the policing counter indicated by the PCNUM field
PCNUM	11:8	rw	**Policing Counter Number** This field specifies the counter number selected for corresponding DA/SA address filter event.
0	7:1, 31:12	r	**Reserved** Read as 0; should be written with 0.

14.6.251 Port j MAC DPCSel IndReg8

This register contains policing counter number corresponding to each DA or SA filter event.

Portj_MAC_DPCSel_IndReg8 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC DPCSel IndReg8 Kernel Reset value: $0000\ 0000_H$



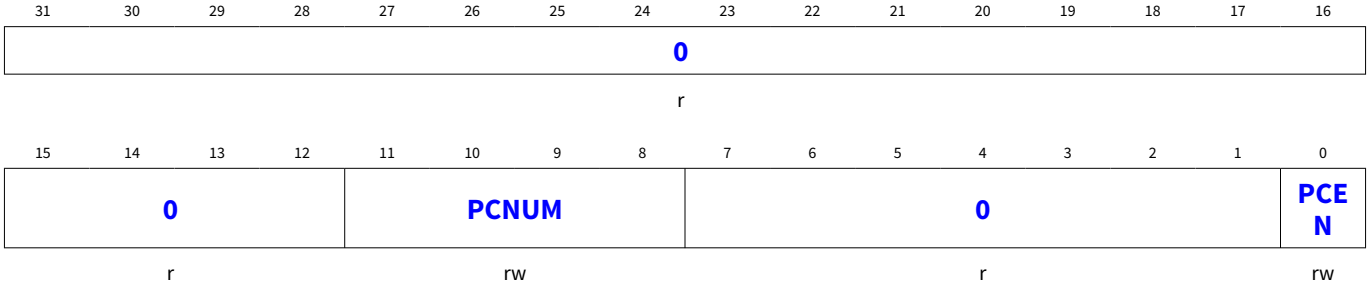
Field	Bits	Type	Description
PCEN	0	rw	**Policing Counter Enable** When set enables the policing counter indicated by the PCNUM field
PCNUM	11:8	rw	**Policing Counter Number** This field specifies the counter number selected for corresponding DA/SA address filter event.
0	7:1, 31:12	r	**Reserved** Read as 0; should be written with 0.

14.6.252 Port j MAC DPCSel IndReg9

This register contains policing counter number corresponding to each DA or SA filter event.

Portj_MAC_DPCSel_IndReg9 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC DPCSel IndReg9 Kernel Reset value: $0000\ 0000_H$



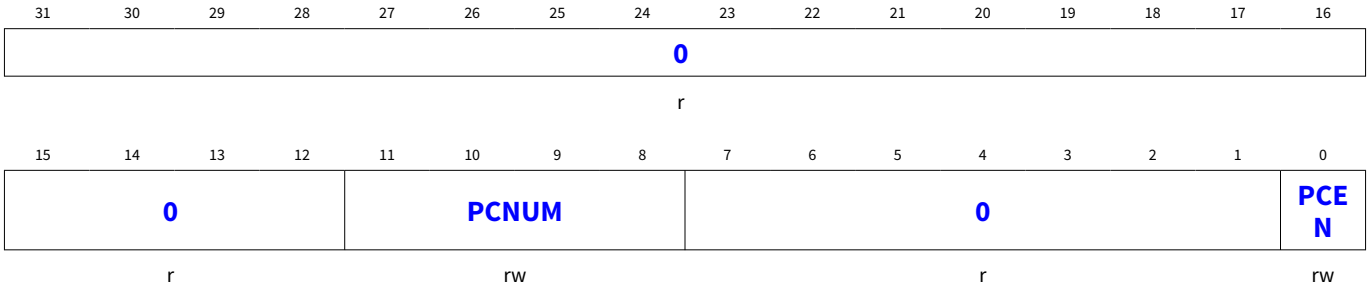
Field	Bits	Type	Description
PCEN	0	rw	**Policing Counter Enable** When set enables the policing counter indicated by the PCNUM field
PCNUM	11:8	rw	**Policing Counter Number** This field specifies the counter number selected for corresponding DA/SA address filter event.
0	7:1, 31:12	r	**Reserved** Read as 0; should be written with 0.

14.6.253 Port j MAC FPCSel IndReg0

This register contains policing counter number for events other than the Filter Pass events which is given below.
1. Filter Fail

Portj_MAC_FPCSel_IndReg0 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC FPCSel IndReg0 Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
PCEN	0	rw	**Policing Counter Enable** When set enables the policing counter indicated by the PCNUM field
PCNUM	11:8	rw	**Policing Counter Number** This field specifies the counter number selected for "Filter fail" event.

(table continues...)

(continued)

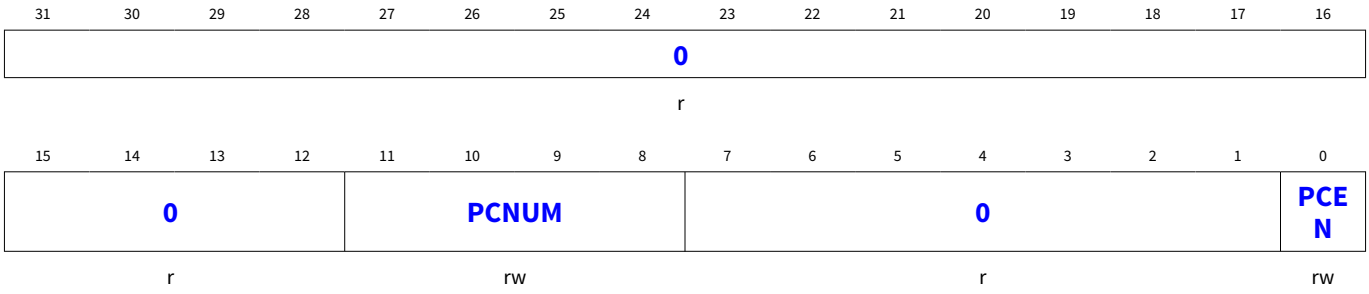
Field	Bits	Type	Description
0	7:1, 31:12	r	**Reserved** Read as 0; should be written with 0.

14.6.254 Port j MAC LPCSel IndReg0

This register contains policing counter number corresponding to each L3L4 filter event.

Portj_MAC_LPCSel_IndReg0 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC LPCSel IndReg0 Kernel Reset value: $0000\ 0000_H$



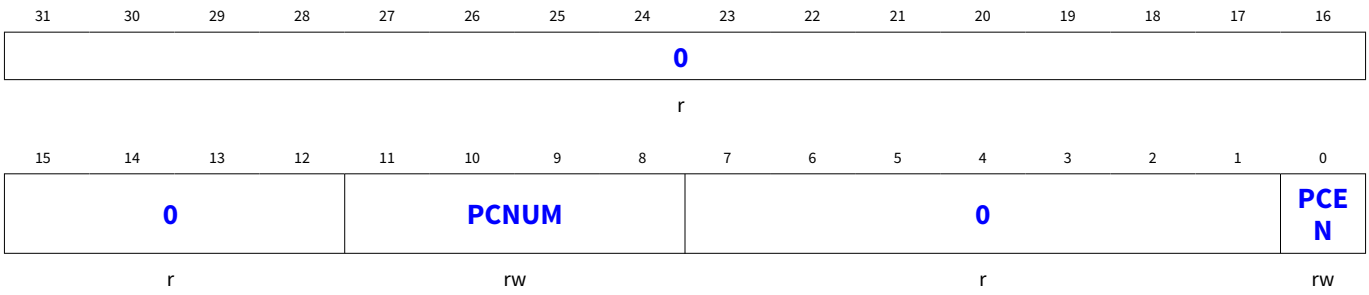
Field	Bits	Type	Description
PCEN	0	rw	**Policing Counter Enable** When set enables the policing counter indicated by the PCNUM field
PCNUM	11:8	rw	**Policing Counter Number** This field specifies the counter number selected for corresponding L3_L4 filter event.
0	7:1, 31:12	r	**Reserved** Read as 0; should be written with 0.

14.6.255 Port j MAC LPCSel IndReg1

This register contains policing counter number corresponding to each L3L4 filter event.

Portj_MAC_LPCSel_IndReg1 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC LPCSel IndReg1 Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
PCEN	0	rw	**Policing Counter Enable** When set enables the policing counter indicated by the PCNUM field
PCNUM	11:8	rw	**Policing Counter Number** This field specifies the counter number selected for corresponding L3_L4 filter event.
0	7:1, 31:12	r	**Reserved** Read as 0; should be written with 0.

14.6.256 Port j MAC LPCSel IndReg2

This register contains policing counter number corresponding to each L3L4 filter event.

Portj_MAC_LPCSel_IndReg2 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC LPCSel IndReg2 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				PCNUM				0							PCEN
r				rw				r							rw

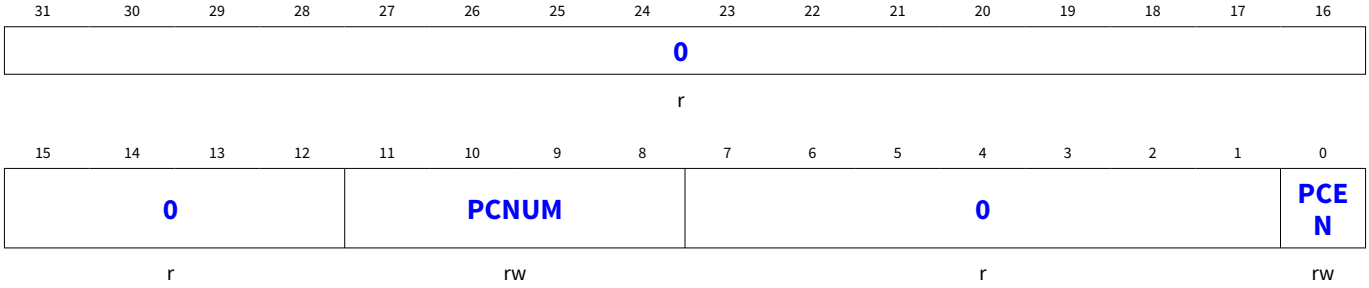
Field	Bits	Type	Description
PCEN	0	rw	**Policing Counter Enable** When set enables the policing counter indicated by the PCNUM field
PCNUM	11:8	rw	**Policing Counter Number** This field specifies the counter number selected for corresponding L3_L4 filter event.
0	7:1, 31:12	r	**Reserved** Read as 0; should be written with 0.

14.6.257 Port j MAC LPCSel IndReg3

This register contains policing counter number corresponding to each L3L4 filter event.

Portj_MAC_LPCSel_IndReg3 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC LPCSel IndReg3 Kernel Reset value: $0000\ 0000_H$



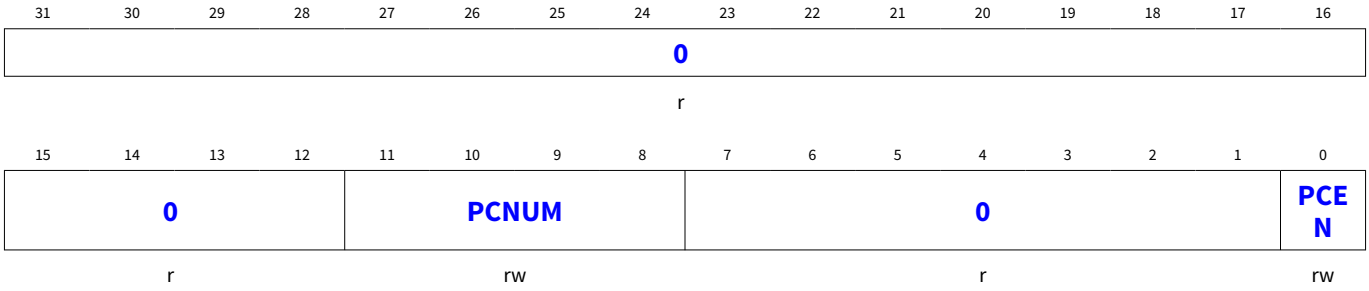
Field	Bits	Type	Description
PCEN	0	rw	**Policing Counter Enable** When set enables the policing counter indicated by the PCNUM field
PCNUM	11:8	rw	**Policing Counter Number** This field specifies the counter number selected for corresponding L3_L4 filter event.
0	7:1, 31:12	r	**Reserved** Read as 0; should be written with 0.

14.6.258 Port j MAC LPCSel IndReg4

This register contains policing counter number corresponding to each L3L4 filter event.

Portj_MAC_LPCSel_IndReg4 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC LPCSel IndReg4 Kernel Reset value: $0000\ 0000_H$



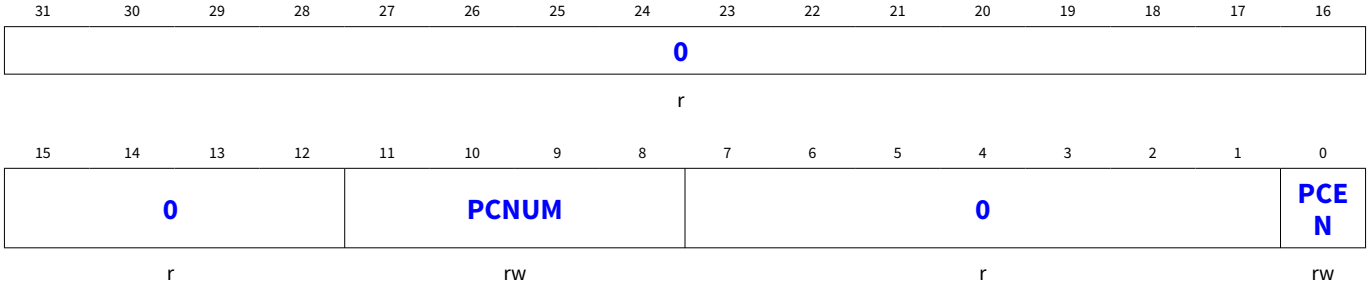
Field	Bits	Type	Description
PCEN	0	rw	**Policing Counter Enable** When set enables the policing counter indicated by the PCNUM field
PCNUM	11:8	rw	**Policing Counter Number** This field specifies the counter number selected for corresponding L3_L4 filter event.
0	7:1, 31:12	r	**Reserved** Read as 0; should be written with 0.

14.6.259 Port j MAC LPCSel IndReg5

This register contains policing counter number corresponding to each L3L4 filter event.

Portj_MAC_LPCSel_IndReg5 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC LPCSel IndReg5 Kernel Reset value: $0000\ 0000_H$



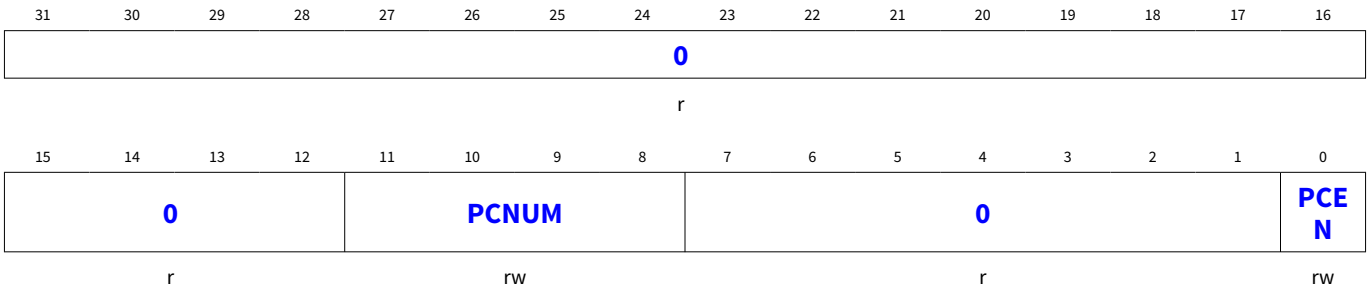
Field	Bits	Type	Description
PCEN	0	rw	**Policing Counter Enable** When set enables the policing counter indicated by the PCNUM field
PCNUM	11:8	rw	**Policing Counter Number** This field specifies the counter number selected for corresponding L3_L4 filter event.
0	7:1, 31:12	r	**Reserved** Read as 0; should be written with 0.

14.6.260 Port j MAC LPCSel IndReg6

This register contains policing counter number corresponding to each L3L4 filter event.

Portj_MAC_LPCSel_IndReg6 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC LPCSel IndReg6 Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
PCEN	0	rw	**Policing Counter Enable** When set enables the policing counter indicated by the PCNUM field
PCNUM	11:8	rw	**Policing Counter Number** This field specifies the counter number selected for corresponding L3_L4 filter event.

(table continues...)

(continued)

Field	Bits	Type	Description
0	7:1, 31:12	r	**Reserved** Read as 0; should be written with 0.

14.6.261 Port j MAC LPCSel IndReg7

This register contains policing counter number corresponding to each L3L4 filter event.

Portj_MAC_LPCSel_IndReg7 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC LPCSel IndReg7 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				PCNUM				0							PCEN
r				rw				r							rw

Field	Bits	Type	Description
PCEN	0	rw	**Policing Counter Enable** When set enables the policing counter indicated by the PCNUM field
PCNUM	11:8	rw	**Policing Counter Number** This field specifies the counter number selected for corresponding L3_L4 filter event.
0	7:1, 31:12	r	**Reserved** Read as 0; should be written with 0.

14.6.262 Port j MAC PCCtrl IndReg0

This register contains control information for policing counters associated with MAC_(#MSEL)_(#AOFF) registers.

Portj_MAC_PCCtrl_IndReg0 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC PCCtrl IndReg0 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			FFAT	FRZ	PCM	TWSEL		0			TRSLD				
r			rw	rw	rw	rw		r			rw				

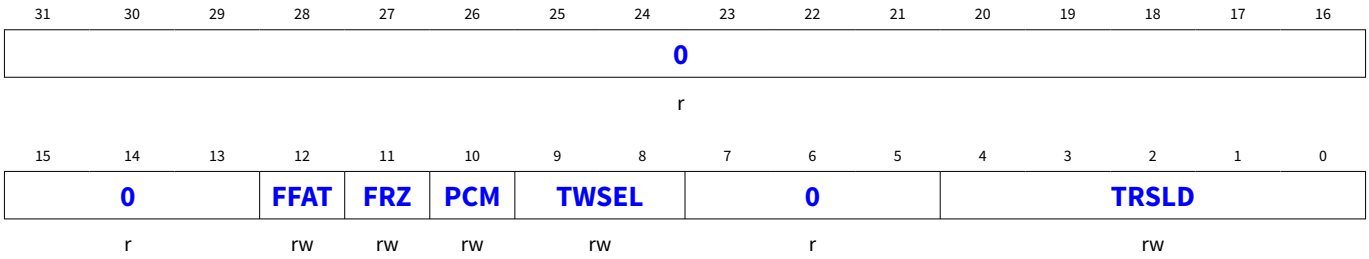
Field	Bits	Type	Description
TRSLD	4:0	rw	**Threshold set for Policing Counter** This field indicates programmed filter event count threshold set for the policing counter. - 00000: 1(2^0) - 00001: 2(2^1) - ..... - 11111: 32(2^31)
TWSEL	9:8	rw	**Time window Select** This field is used to select any one of the time window from sbd_pc_tim_wdw_i[3:0] input, for counter increment. - 00: Time window 0 - 01: Time window 1 - 10: Time window 2 - 11: Time window 3
PCM	10	rw	**Policing Counter Mode** Values: - 0x0: Free running counter. In this mode, counter is incremented irrespective of time window. - 0x1: Time window-based counter. In this mode, counter is incremented within the provided time window.
FRZ	11	rw	**Freeze Enable** This field indicates counter freeze when end of time window is encountered Values: - 0x0: Restarts the counter on next start of time window Free running counter - 0x1: Retains counter value till corresponding counter is read, counts restart from the start of the next time window.
FFAT	12	rw	**Forced Filter fail After Threshold** When this bit is set and the corresponding counter crosses the threshold, all the filter events mapped to that counter is treated as fail. Packets are treated as fail from the time threshold is crossed, until the counter is cleared.
0	7:5, 31:13	r	**Reserved** Read as 0; should be written with 0.

14.6.263 Port j MAC PCCtrl IndReg1

This register contains control information for policing counters associated with MAC_(#MSEL)_(#AOFF) registers.

Portj_MAC_PCCtrl_IndReg1 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC PCCtrl IndReg1 Kernel Reset value: $0000\ 0000_H$



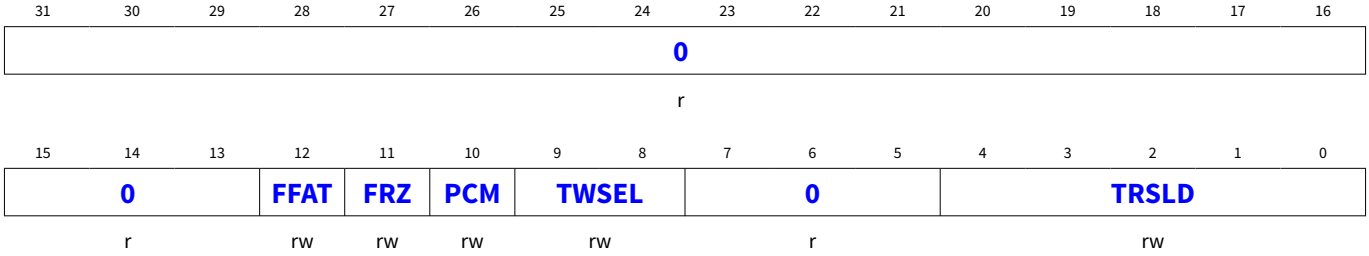
Field	Bits	Type	Description
TRSLD	4:0	rw	**Threshold set for Policing Counter** This field indicates programmed filter event count threshold set for the policing counter. - 00000: 1(2^0) - 00001: 2(2^1) - ..... - 11111: 32(2^31)
TWSEL	9:8	rw	**Time window Select** This field is used to select any one of the time window from sbd_pc_tim_wdw_i[3:0] input, for counter increment. - 00: Time window 0 - 01: Time window 1 - 10: Time window 2 - 11: Time window 3
PCM	10	rw	**Policing Counter Mode** Values: - 0x0: Free running counter. In this mode, counter is incremented irrespective of time window. - 0x1: Time window-based counter. In this mode, counter is incremented within the provided time window.
FRZ	11	rw	**Freeze Enable** This field indicates counter freeze when end of time window is encountered Values: - 0x0: Restarts the counter on next start of time window Free running counter - 0x1: Retains counter value till corresponding counter is read, counts restart from the start of the next time window.
FFAT	12	rw	**Forced Filter fail After Threshold** When this bit is set and the corresponding counter crosses the threshold, all the filter events mapped to that counter is treated as fail. Packets are treated as fail from the time threshold is crossed, until the counter is cleared.
0	7:5, 31:13	r	**Reserved** Read as 0; should be written with 0.

14.6.264 Port j MAC PCCtrl IndReg10

This register contains control information for policing counters associated with MAC_(#MSEL)_(#AOFF) registers.

Portj_MAC_PCCtrl_IndReg10 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC PCCtrl IndReg10 Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
TRSLD	4:0	rw	**Threshold set for Policing Counter** This field indicates programmed filter event count threshold set for the policing counter. - 00000: 1(2^0) - 00001: 2(2^1) - ..... - 11111: 32(2^31)
TWSEL	9:8	rw	**Time window Select** This field is used to select any one of the time window from sbd_pc_tim_wdw_i[3:0] input, for counter increment. - 00: Time window 0 - 01: Time window 1 - 10: Time window 2 - 11: Time window 3
PCM	10	rw	**Policing Counter Mode** Values: - 0x0: Free running counter. In this mode, counter is incremented irrespective of time window. - 0x1: Time window-based counter. In this mode, counter is incremented within the provided time window.
FRZ	11	rw	**Freeze Enable** This field indicates counter freeze when end of time window is encountered Values: - 0x0: Restarts the counter on next start of time window Free running counter - 0x1: Retains counter value till corresponding counter is read, counts restart from the start of the next time window.

(table continues...)

(continued)

Field	Bits	Type	Description
FFAT	12	rw	**Forced Filter fail After Threshold** When this bit is set and the corresponding counter crosses the threshold, all the filter events mapped to that counter is treated as fail. Packets are treated as fail from the time threshold is crossed, until the counter is cleared.
0	7:5, 31:13	r	**Reserved** Read as 0; should be written with 0.

14.6.265 Port j MAC PCCtrl IndReg11

This register contains control information for policing counters associated with MAC_(#MSEL)_(#AOFF) registers.

Portj_MAC_PCCtrl_IndReg11 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC PCCtrl IndReg11 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			FFAT	FRZ	PCM	TWSEL		0			TRSLD				
r			rw	rw	rw	rw		r			rw				

Field	Bits	Type	Description
TRSLD	4:0	rw	**Threshold set for Policing Counter** This field indicates programmed filter event count threshold set for the policing counter. - 00000: 1(2^0) - 00001: 2(2^1) - ..... - 11111: 32(2^31)
TWSEL	9:8	rw	**Time window Select** This field is used to select any one of the time window from sbd_pc_tim_wdw_i[3:0] input, for counter increment. - 00: Time window 0 - 01: Time window 1 - 10: Time window 2 - 11: Time window 3
PCM	10	rw	**Policing Counter Mode** Values: - 0x0: Free running counter. In this mode, counter is incremented irrespective of time window. - 0x1: Time window-based counter. In this mode, counter is incremented within the provided time window.

(table continues...)

(continued)

Field	Bits	Type	Description
FRZ	11	rw	**Freeze Enable** This field indicates counter freeze when end of time window is encountered Values: - 0x0: Restarts the counter on next start of time window Free running counter - 0x1: Retains counter value till corresponding counter is read, counts restart from the start of the next time window.
FFAT	12	rw	**Forced Filter fail After Threshold** When this bit is set and the corresponding counter crosses the threshold, all the filter events mapped to that counter is treated as fail. Packets are treated as fail from the time threshold is crossed, until the counter is cleared.
0	7:5, 31:13	r	**Reserved** Read as 0; should be written with 0.

14.6.266 Port j MAC PCCtrl IndReg12

This register contains control information for policing counters associated with MAC_(#MSEL)_(#AOFF) registers.

Portj_MAC_PCCtrl_IndReg12 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC PCCtrl IndReg12 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			FFAT	FRZ	PCM	TWSEL		0			TRSLD				
r			rw	rw	rw	rw		r			rw				

Field	Bits	Type	Description
TRSLD	4:0	rw	**Threshold set for Policing Counter** This field indicates programmed filter event count threshold set for the policing counter. - 00000: 1(2^0) - 00001: 2(2^1) - ..... - 11111: 32(2^31)

(table continues...)

(continued)

Field	Bits	Type	Description
TWSEL	9:8	rw	**Time window Select** This field is used to select any one of the time window from sbd_pc_tim_wdw_i[3:0] input, for counter increment. - 00: Time window 0 - 01: Time window 1 - 10: Time window 2 - 11: Time window 3
PCM	10	rw	**Policing Counter Mode** Values: - 0x0: Free running counter. In this mode, counter is incremented irrespective of time window. - 0x1: Time window-based counter. In this mode, counter is incremented within the provided time window.
FRZ	11	rw	**Freeze Enable** This field indicates counter freeze when end of time window is encountered Values: - 0x0: Restarts the counter on next start of time window Free running counter - 0x1: Retains counter value till corresponding counter is read, counts restart from the start of the next time window.
FFAT	12	rw	**Forced Filter fail After Threshold** When this bit is set and the corresponding counter crosses the threshold, all the filter events mapped to that counter is treated as fail. Packets are treated as fail from the time threshold is crossed, until the counter is cleared.
0	7:5, 31:13	r	**Reserved** Read as 0; should be written with 0.

14.6.267 Port j MAC PCCtrl IndReg13

This register contains control information for policing counters associated with MAC_(#MSEL)_(#AOFF) registers.

Portj_MAC_PCCtrl_IndReg13 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC PCCtrl IndReg13 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			FFAT	FRZ	PCM	TWSEL		0			TRSLD				
r			rw	rw	rw	rw		r			rw				

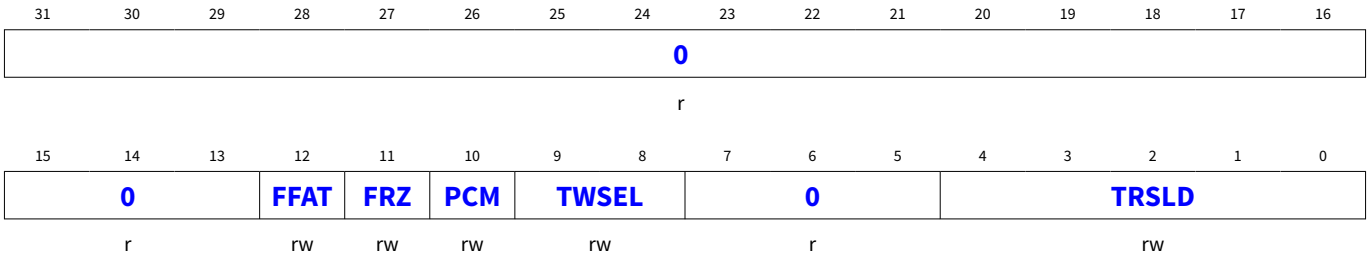
Field	Bits	Type	Description
TRSLD	4:0	rw	**Threshold set for Policing Counter** This field indicates programmed filter event count threshold set for the policing counter. - 00000: 1(2^0) - 00001: 2(2^1) - ..... - 11111: 32(2^31)
TWSEL	9:8	rw	**Time window Select** This field is used to select any one of the time window from sbd_pc_tim_wdw_i[3:0] input, for counter increment. - 00: Time window 0 - 01: Time window 1 - 10: Time window 2 - 11: Time window 3
PCM	10	rw	**Policing Counter Mode** Values: - 0x0: Free running counter. In this mode, counter is incremented irrespective of time window. - 0x1: Time window-based counter. In this mode, counter is incremented within the provided time window.
FRZ	11	rw	**Freeze Enable** This field indicates counter freeze when end of time window is encountered Values: - 0x0: Restarts the counter on next start of time window Free running counter - 0x1: Retains counter value till corresponding counter is read, counts restart from the start of the next time window.
FFAT	12	rw	**Forced Filter fail After Threshold** When this bit is set and the corresponding counter crosses the threshold, all the filter events mapped to that counter is treated as fail. Packets are treated as fail from the time threshold is crossed, until the counter is cleared.
0	7:5, 31:13	r	**Reserved** Read as 0; should be written with 0.

14.6.268 Port j MAC PCCtrl IndReg14

This register contains control information for policing counters associated with MAC_(#MSEL)_(#AOFF) registers.

Portj_MAC_PCCtrl_IndReg14 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC PCCtrl IndReg14 Kernel Reset value: $0000\ 0000_H$



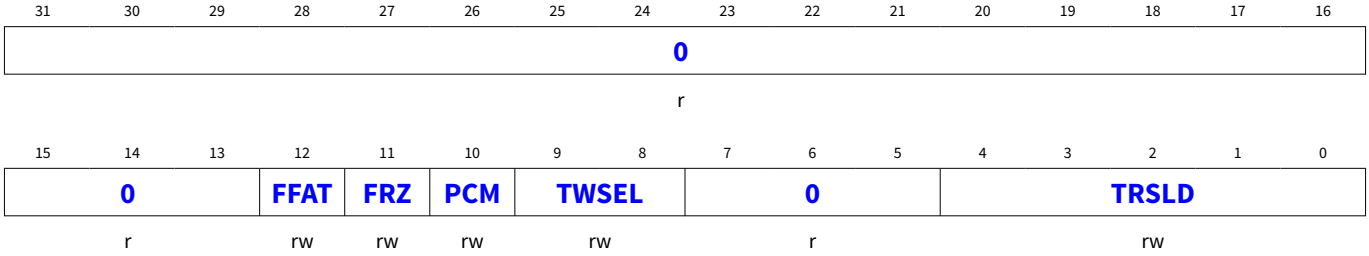
Field	Bits	Type	Description
TRSLD	4:0	rw	**Threshold set for Policing Counter** This field indicates programmed filter event count threshold set for the policing counter. - 00000: 1(2^0) - 00001: 2(2^1) - ..... - 11111: 32(2^31)
TWSEL	9:8	rw	**Time window Select** This field is used to select any one of the time window from sbd_pc_tim_wdw_i[3:0] input, for counter increment. - 00: Time window 0 - 01: Time window 1 - 10: Time window 2 - 11: Time window 3
PCM	10	rw	**Policing Counter Mode** Values: - 0x0: Free running counter. In this mode, counter is incremented irrespective of time window. - 0x1: Time window-based counter. In this mode, counter is incremented within the provided time window.
FRZ	11	rw	**Freeze Enable** This field indicates counter freeze when end of time window is encountered Values: - 0x0: Restarts the counter on next start of time window Free running counter - 0x1: Retains counter value till corresponding counter is read, counts restart from the start of the next time window.
FFAT	12	rw	**Forced Filter fail After Threshold** When this bit is set and the corresponding counter crosses the threshold, all the filter events mapped to that counter is treated as fail. Packets are treated as fail from the time threshold is crossed, until the counter is cleared.
0	7:5, 31:13	r	**Reserved** Read as 0; should be written with 0.

14.6.269 Port j MAC PCCtrl IndReg15

This register contains control information for policing counters associated with MAC_(#MSEL)_(#AOFF) registers.

Portj_MAC_PCCtrl_IndReg15 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC PCCtrl IndReg15 Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
TRSLD	4:0	rw	**Threshold set for Policing Counter** This field indicates programmed filter event count threshold set for the policing counter. - 00000: 1(2^0) - 00001: 2(2^1) - ..... - 11111: 32(2^31)
TWSEL	9:8	rw	**Time window Select** This field is used to select any one of the time window from sbd_pc_tim_wdw_i[3:0] input, for counter increment. - 00: Time window 0 - 01: Time window 1 - 10: Time window 2 - 11: Time window 3
PCM	10	rw	**Policing Counter Mode** Values: - 0x0: Free running counter. In this mode, counter is incremented irrespective of time window. - 0x1: Time window-based counter. In this mode, counter is incremented within the provided time window.
FRZ	11	rw	**Freeze Enable** This field indicates counter freeze when end of time window is encountered Values: - 0x0: Restarts the counter on next start of time window Free running counter - 0x1: Retains counter value till corresponding counter is read, counts restart from the start of the next time window.

(table continues...)

(continued)

Field	Bits	Type	Description
FFAT	12	rw	**Forced Filter fail After Threshold** When this bit is set and the corresponding counter crosses the threshold, all the filter events mapped to that counter is treated as fail. Packets are treated as fail from the time threshold is crossed, until the counter is cleared.
0	7:5, 31:13	r	**Reserved** Read as 0; should be written with 0.

14.6.270 Port j MAC PCCtrl IndReg2

This register contains control information for policing counters associated with MAC_(#MSEL)_(#AOFF) registers.

Portj_MAC_PCCtrl_IndReg2 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC PCCtrl IndReg2 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			FFAT	FRZ	PCM	TWSEL		0			TRSLD				
r			rw	rw	rw	rw		r			rw				

Field	Bits	Type	Description
TRSLD	4:0	rw	**Threshold set for Policing Counter** This field indicates programmed filter event count threshold set for the policing counter. - 00000: 1(2^0) - 00001: 2(2^1) - ..... - 11111: 32(2^31)
TWSEL	9:8	rw	**Time window Select** This field is used to select any one of the time window from sbd_pc_tim_wdw_i[3:0] input, for counter increment. - 00: Time window 0 - 01: Time window 1 - 10: Time window 2 - 11: Time window 3
PCM	10	rw	**Policing Counter Mode** Values: - 0x0: Free running counter. In this mode, counter is incremented irrespective of time window. - 0x1: Time window-based counter. In this mode, counter is incremented within the provided time window.

(table continues...)

(continued)

Field	Bits	Type	Description
FRZ	11	rw	**Freeze Enable** This field indicates counter freeze when end of time window is encountered Values: - 0x0: Restarts the counter on next start of time window Free running counter - 0x1: Retains counter value till corresponding counter is read, counts restart from the start of the next time window.
FFAT	12	rw	**Forced Filter fail After Threshold** When this bit is set and the corresponding counter crosses the threshold, all the filter events mapped to that counter is treated as fail. Packets are treated as fail from the time threshold is crossed, until the counter is cleared.
0	7:5, 31:13	r	**Reserved** Read as 0; should be written with 0.

14.6.271 Port j MAC PCCtrl IndReg3

This register contains control information for policing counters associated with MAC_(#MSEL)_(#AOFF) registers.

Portj_MAC_PCCtrl_IndReg3 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC PCCtrl IndReg3 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			FFAT	FRZ	PCM	TWSEL		0			TRSLD				
r			rw	rw	rw	rw		r			rw				

Field	Bits	Type	Description
TRSLD	4:0	rw	**Threshold set for Policing Counter** This field indicates programmed filter event count threshold set for the policing counter. - 00000: 1(2^0) - 00001: 2(2^1) - ..... - 11111: 32(2^31)

(table continues...)

(continued)

Field	Bits	Type	Description
TWSEL	9:8	rw	**Time window Select** This field is used to select any one of the time window from sbd_pc_tim_wdw_i[3:0] input, for counter increment. - 00: Time window 0 - 01: Time window 1 - 10: Time window 2 - 11: Time window 3
PCM	10	rw	**Policing Counter Mode** Values: - 0x0: Free running counter. In this mode, counter is incremented irrespective of time window. - 0x1: Time window-based counter. In this mode, counter is incremented within the provided time window.
FRZ	11	rw	**Freeze Enable** This field indicates counter freeze when end of time window is encountered Values: - 0x0: Restarts the counter on next start of time window Free running counter - 0x1: Retains counter value till corresponding counter is read, counts restart from the start of the next time window.
FFAT	12	rw	**Forced Filter fail After Threshold** When this bit is set and the corresponding counter crosses the threshold, all the filter events mapped to that counter is treated as fail. Packets are treated as fail from the time threshold is crossed, until the counter is cleared.
0	7:5, 31:13	r	**Reserved** Read as 0; should be written with 0.

14.6.272 Port j MAC PCCtrl IndReg4

This register contains control information for policing counters associated with MAC_(#MSEL)_(#AOFF) registers.

Portj_MAC_PCCtrl_IndReg4 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC PCCtrl IndReg4 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			FFAT	FRZ	PCM	TWSEL		0			TRSLD				
r			rw	rw	rw	rw		r			rw				

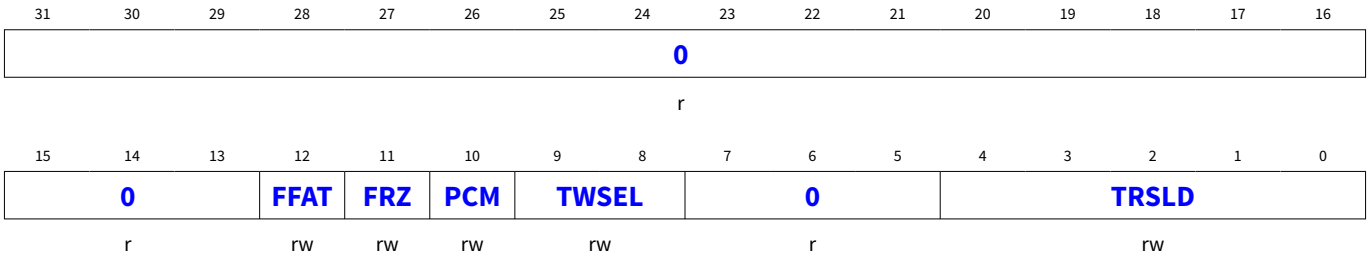
Field	Bits	Type	Description
TRSLD	4:0	rw	**Threshold set for Policing Counter** This field indicates programmed filter event count threshold set for the policing counter. - 00000: 1(2^0) - 00001: 2(2^1) - ..... - 11111: 32(2^31)
TWSEL	9:8	rw	**Time window Select** This field is used to select any one of the time window from sbd_pc_tim_wdw_i[3:0] input, for counter increment. - 00: Time window 0 - 01: Time window 1 - 10: Time window 2 - 11: Time window 3
PCM	10	rw	**Policing Counter Mode** Values: - 0x0: Free running counter. In this mode, counter is incremented irrespective of time window. - 0x1: Time window-based counter. In this mode, counter is incremented within the provided time window.
FRZ	11	rw	**Freeze Enable** This field indicates counter freeze when end of time window is encountered Values: - 0x0: Restarts the counter on next start of time window Free running counter - 0x1: Retains counter value till corresponding counter is read, counts restart from the start of the next time window.
FFAT	12	rw	**Forced Filter fail After Threshold** When this bit is set and the corresponding counter crosses the threshold, all the filter events mapped to that counter is treated as fail. Packets are treated as fail from the time threshold is crossed, until the counter is cleared.
0	7:5, 31:13	r	**Reserved** Read as 0; should be written with 0.

14.6.273 Port j MAC PCCtrl IndReg5

This register contains control information for policing counters associated with MAC_(#MSEL)_(#AOFF) registers.

Portj_MAC_PCCtrl_IndReg5 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC PCCtrl IndReg5 Kernel Reset value: $0000\ 0000_H$



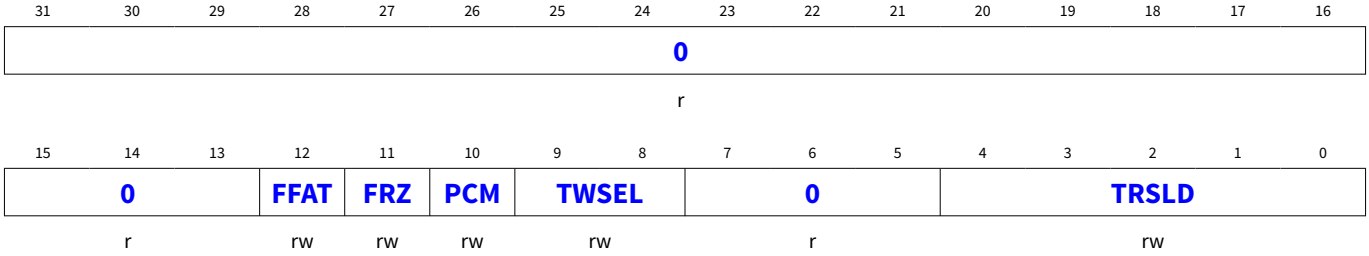
Field	Bits	Type	Description
TRSLD	4:0	rw	**Threshold set for Policing Counter** This field indicates programmed filter event count threshold set for the policing counter. - 00000: 1(2^0) - 00001: 2(2^1) - ..... - 11111: 32(2^31)
TWSEL	9:8	rw	**Time window Select** This field is used to select any one of the time window from sbd_pc_tim_wdw_i[3:0] input, for counter increment. - 00: Time window 0 - 01: Time window 1 - 10: Time window 2 - 11: Time window 3
PCM	10	rw	**Policing Counter Mode** Values: - 0x0: Free running counter. In this mode, counter is incremented irrespective of time window. - 0x1: Time window-based counter. In this mode, counter is incremented within the provided time window.
FRZ	11	rw	**Freeze Enable** This field indicates counter freeze when end of time window is encountered Values: - 0x0: Restarts the counter on next start of time window Free running counter - 0x1: Retains counter value till corresponding counter is read, counts restart from the start of the next time window.
FFAT	12	rw	**Forced Filter fail After Threshold** When this bit is set and the corresponding counter crosses the threshold, all the filter events mapped to that counter is treated as fail. Packets are treated as fail from the time threshold is crossed, until the counter is cleared.
0	7:5, 31:13	r	**Reserved** Read as 0; should be written with 0.

14.6.274 Port j MAC PCCtrl IndReg6

This register contains control information for policing counters associated with MAC_(#MSEL)_(#AOFF) registers.

Portj_MAC_PCCtrl_IndReg6 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC PCCtrl IndReg6 Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
TRSLD	4:0	rw	**Threshold set for Policing Counter** This field indicates programmed filter event count threshold set for the policing counter. - 00000: 1(2^0) - 00001: 2(2^1) - ..... - 11111: 32(2^31)
TWSEL	9:8	rw	**Time window Select** This field is used to select any one of the time window from sbd_pc_tim_wdw_i[3:0] input, for counter increment. - 00: Time window 0 - 01: Time window 1 - 10: Time window 2 - 11: Time window 3
PCM	10	rw	**Policing Counter Mode** Values: - 0x0: Free running counter. In this mode, counter is incremented irrespective of time window. - 0x1: Time window-based counter. In this mode, counter is incremented within the provided time window.
FRZ	11	rw	**Freeze Enable** This field indicates counter freeze when end of time window is encountered Values: - 0x0: Restarts the counter on next start of time window Free running counter - 0x1: Retains counter value till corresponding counter is read, counts restart from the start of the next time window.

(table continues...)

(continued)

Field	Bits	Type	Description
FFAT	12	rw	**Forced Filter fail After Threshold** When this bit is set and the corresponding counter crosses the threshold, all the filter events mapped to that counter is treated as fail. Packets are treated as fail from the time threshold is crossed, until the counter is cleared.
0	7:5, 31:13	r	**Reserved** Read as 0; should be written with 0.

14.6.275 Port j MAC PCCtrl IndReg7

This register contains control information for policing counters associated with MAC_(#MSEL)_(#AOFF) registers.

Portj_MAC_PCCtrl_IndReg7 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC PCCtrl IndReg7 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			FFAT	FRZ	PCM	TWSEL		0			TRSLD				
r			rw	rw	rw	rw		r			rw				

Field	Bits	Type	Description
TRSLD	4:0	rw	**Threshold set for Policing Counter** This field indicates programmed filter event count threshold set for the policing counter. - 00000: 1(2^0) - 00001: 2(2^1) - ..... - 11111: 32(2^31)
TWSEL	9:8	rw	**Time window Select** This field is used to select any one of the time window from sbd_pc_tim_wdw_i[3:0] input, for counter increment. - 00: Time window 0 - 01: Time window 1 - 10: Time window 2 - 11: Time window 3
PCM	10	rw	**Policing Counter Mode** Values: - 0x0: Free running counter. In this mode, counter is incremented irrespective of time window. - 0x1: Time window-based counter. In this mode, counter is incremented within the provided time window.

(table continues...)

(continued)

Field	Bits	Type	Description
FRZ	11	rw	**Freeze Enable** This field indicates counter freeze when end of time window is encountered Values: - 0x0: Restarts the counter on next start of time window Free running counter - 0x1: Retains counter value till corresponding counter is read, counts restart from the start of the next time window.
FFAT	12	rw	**Forced Filter fail After Threshold** When this bit is set and the corresponding counter crosses the threshold, all the filter events mapped to that counter is treated as fail. Packets are treated as fail from the time threshold is crossed, until the counter is cleared.
0	7:5, 31:13	r	**Reserved** Read as 0; should be written with 0.

14.6.276 Port j MAC PCCtrl IndReg8

This register contains control information for policing counters associated with MAC_(#MSEL)_(#AOFF) registers.

Portj_MAC_PCCtrl_IndReg8 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC PCCtrl IndReg8 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			FFAT	FRZ	PCM	TWSEL		0			TRSLD				
r			rw	rw	rw	rw		r			rw				

Field	Bits	Type	Description
TRSLD	4:0	rw	**Threshold set for Policing Counter** This field indicates programmed filter event count threshold set for the policing counter. - 00000: 1(2^0) - 00001: 2(2^1) - ..... - 11111: 32(2^31)

(table continues...)

(continued)

Field	Bits	Type	Description
TWSEL	9:8	rw	**Time window Select** This field is used to select any one of the time window from sbd_pc_tim_wdw_i[3:0] input, for counter increment. - 00: Time window 0 - 01: Time window 1 - 10: Time window 2 - 11: Time window 3
PCM	10	rw	**Policing Counter Mode** Values: - 0x0: Free running counter. In this mode, counter is incremented irrespective of time window. - 0x1: Time window-based counter. In this mode, counter is incremented within the provided time window.
FRZ	11	rw	**Freeze Enable** This field indicates counter freeze when end of time window is encountered Values: - 0x0: Restarts the counter on next start of time window Free running counter - 0x1: Retains counter value till corresponding counter is read, counts restart from the start of the next time window.
FFAT	12	rw	**Forced Filter fail After Threshold** When this bit is set and the corresponding counter crosses the threshold, all the filter events mapped to that counter is treated as fail. Packets are treated as fail from the time threshold is crossed, until the counter is cleared.
0	7:5, 31:13	r	**Reserved** Read as 0; should be written with 0.

14.6.277 Port j MAC PCCtrl IndReg9

This register contains control information for policing counters associated with MAC_(#MSEL)_(#AOFF) registers.

Portj_MAC_PCCtrl_IndReg9 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC PCCtrl IndReg9 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			FFAT	FRZ	PCM	TWSEL		0			TRSLD				
r			rw	rw	rw	rw		r			rw				

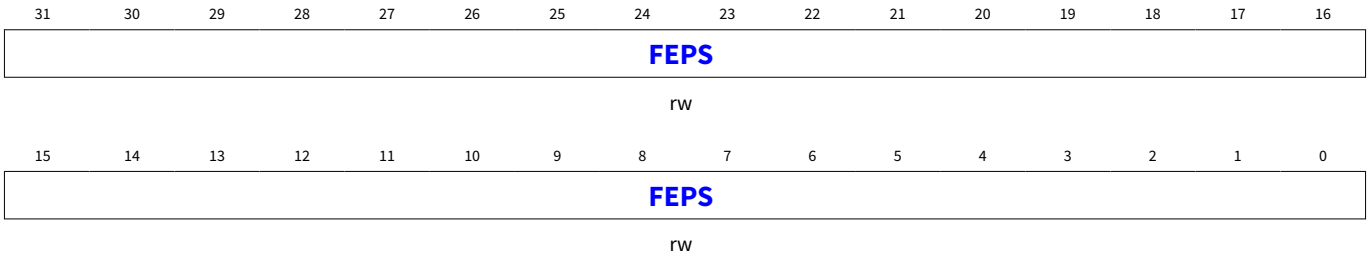
Field	Bits	Type	Description
TRSLD	4:0	rw	**Threshold set for Policing Counter** This field indicates programmed filter event count threshold set for the policing counter. - 00000: 1(2^0) - 00001: 2(2^1) - ..... - 11111: 32(2^31)
TWSEL	9:8	rw	**Time window Select** This field is used to select any one of the time window from sbd_pc_tim_wdw_i[3:0] input, for counter increment. - 00: Time window 0 - 01: Time window 1 - 10: Time window 2 - 11: Time window 3
PCM	10	rw	**Policing Counter Mode** Values: - 0x0: Free running counter. In this mode, counter is incremented irrespective of time window. - 0x1: Time window-based counter. In this mode, counter is incremented within the provided time window.
FRZ	11	rw	**Freeze Enable** This field indicates counter freeze when end of time window is encountered Values: - 0x0: Restarts the counter on next start of time window Free running counter - 0x1: Retains counter value till corresponding counter is read, counts restart from the start of the next time window.
FFAT	12	rw	**Forced Filter fail After Threshold** When this bit is set and the corresponding counter crosses the threshold, all the filter events mapped to that counter is treated as fail. Packets are treated as fail from the time threshold is crossed, until the counter is cleared.
0	7:5, 31:13	r	**Reserved** Read as 0; should be written with 0.

14.6.278 Port j MAC PCStatus DA IndReg

This register contains pass status for DA/SA filter where each bit corresponds to specific DA/SA filter event.

Portj_MAC_PCStatus_DA_IndReg (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC PCStatus DA IndReg Kernel Reset value: $0000\ 0000_H$



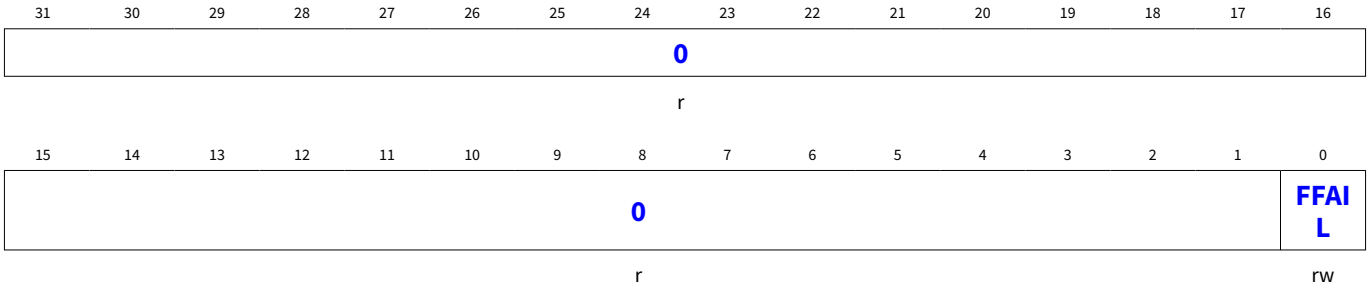
Field	Bits	Type	Description
FEPS	31:0	rw	**Filter Event Pass Status** When set, indicates at least one DA/SA filter event is passed. Each bit represents individual DA/SA filter event. Access restriction applies. This field is cleared when read (or written to 1 when RCWE bit of MAC_CSR_SW_Ctrl register is set).

14.6.279 Port j MAC PCStatus Filter Event IndReg

This register contains status for the events monitored using policing counters apart from Filter Pass Events which is given below. 1. Filter_Fail

Portj_MAC_PCStatus_Filter_Event_IndReg (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC PCStatus Filter Event IndReg Kernel Reset value: $0000\ 0000_H$



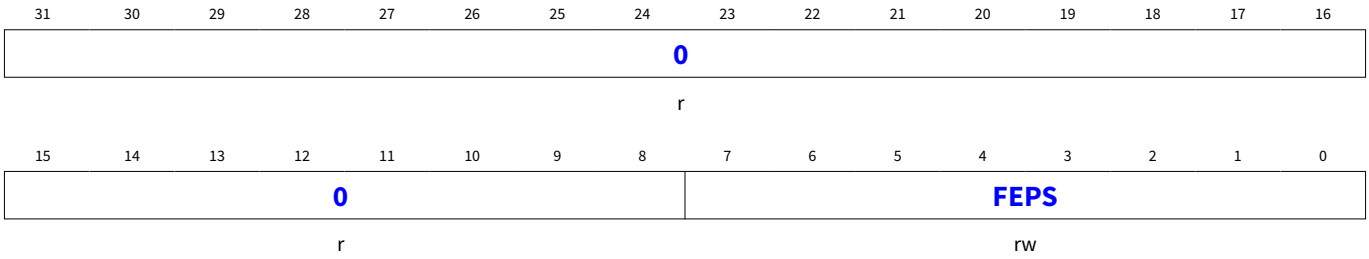
Field	Bits	Type	Description
FFAIL	0	rw	**Filter Fail** This bit is asserted when "Filter fail" event is set/triggered at least once. Access restriction applies. This bit is cleared when read (or written to 1 when RCWE bit of MAC_CSR_SW_Ctrl register is set).
0	31:1	r	**Reserved** Read as 0; should be written with 0.

14.6.280 Port j MAC PCStatus L3L4 IndReg

This register contains pass status for L3_L4 filter where each bit corresponds to specific L3_L4 address filter event.

Portj_MAC_PCStatus_L3L4_IndReg (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC PCStatus L3L4 IndReg Kernel Reset value: $0000\ 0000_H$



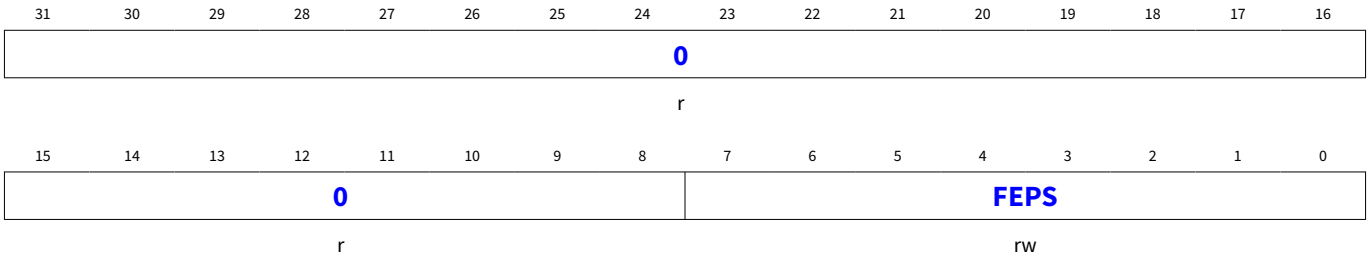
Field	Bits	Type	Description
FEPS	7:0	rw	**Filter Event Pass Status** When set, indicates at least one L3_L4 filter event is passed. Each bit represents individual L3_L4 filter. Access restriction applies. This field is cleared when read (or written to 1 when RCWE bit of MAC_CSR_SW_Ctrl register is set).
0	31:8	r	**Reserved** Read as 0; should be written with 0.

14.6.281 Port j MAC PCStatus VLAN IndReg

This register contains pass status for VLAN filter where each bit corresponds to specific VLAN filter event.

Portj_MAC_PCStatus_VLAN_IndReg (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC PCStatus VLAN IndReg Kernel Reset value: $0000\ 0000_H$



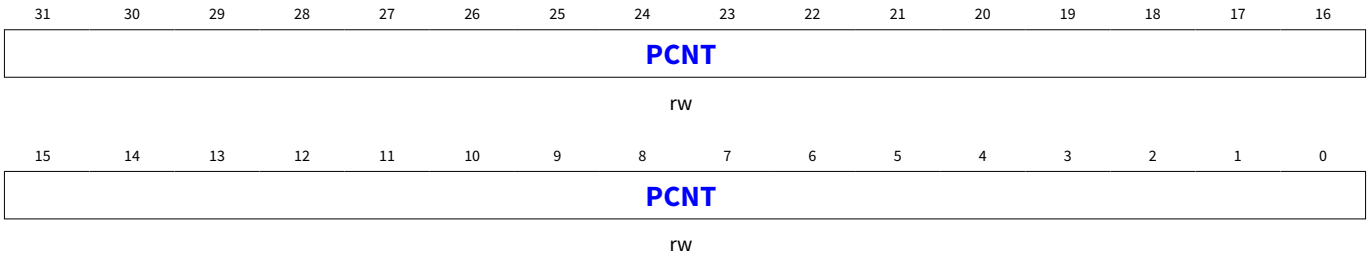
Field	Bits	Type	Description
FEPS	7:0	rw	**Filter Event Pass Status** When set, indicates at-least one VLAN filter event is passed. Each bit represents individual VLAN filters. Access restriction applies. This field is cleared when read (or written to 1 when RCWE bit of MAC_CSR_SW_Ctrl register is set).
0	31:8	r	**Reserved** Read as 0; should be written with 0.

14.6.282 Port j MAC PCntr IndReg0

This register contains count value of policing counters.

Portj_MAC_PCntr_IndReg0 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC PCntr IndReg0 Kernel Reset value: $0000\ 0000_H$



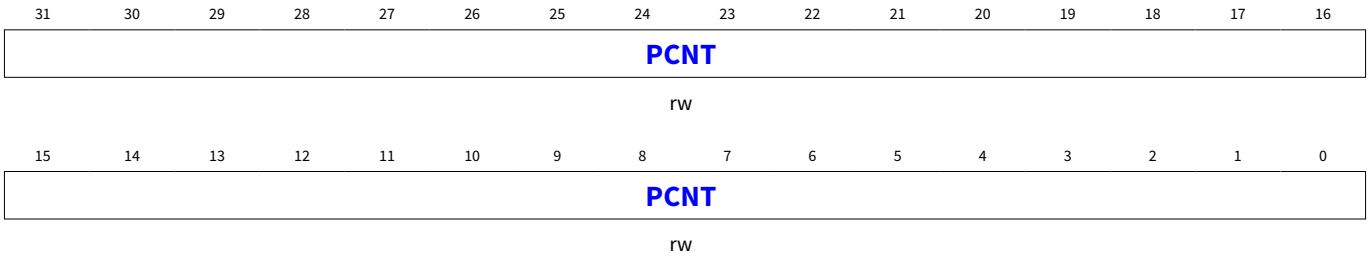
Field	Bits	Type	Description
PCNT	31:0	rw	**Policing Count Value** This field contains count value for policing counters. The counter is incremented within the start and end of the time window. Filter events results in increment of count. - Time window mode: When FRZ is not set, the counter value is cleared when start of the time window is detected. When FRZ is set, the counter is cleared on read. - Free running mode: Counter is cleared on read irrespective of the value of the FRZ bit

14.6.283 Port j MAC PCntr IndReg1

This register contains count value of policing counters.

Portj_MAC_PCntr_IndReg1 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC PCntr IndReg1 Kernel Reset value: $0000\ 0000_H$



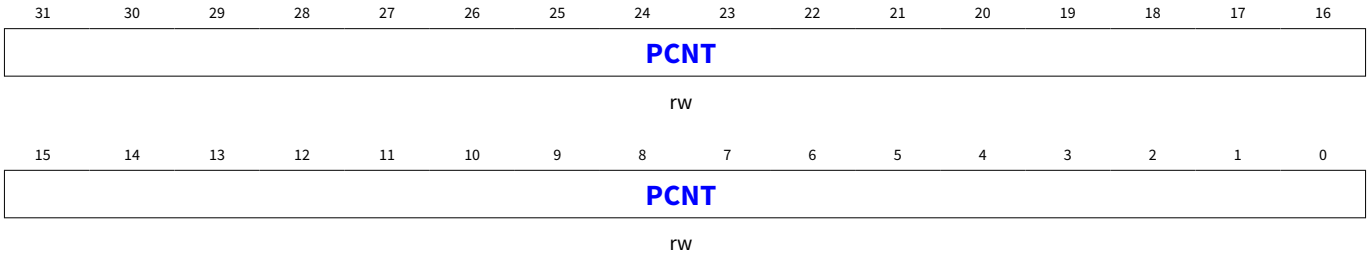
Field	Bits	Type	Description
PCNT	31:0	rw	**Policing Count Value** This field contains count value for policing counters. The counter is incremented within the start and end of the time window. Filter events results in increment of count. - Time window mode: When FRZ is not set, the counter value is cleared when start of the time window is detected. When FRZ is set, the counter is cleared on read. - Free running mode: Counter is cleared on read irrespective of the value of the FRZ bit

14.6.284 Port j MAC PCntr IndReg10

This register contains count value of policing counters.

Portj_MAC_PCntr_IndReg10 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC PCntr IndReg10 Kernel Reset value: $0000\ 0000_H$



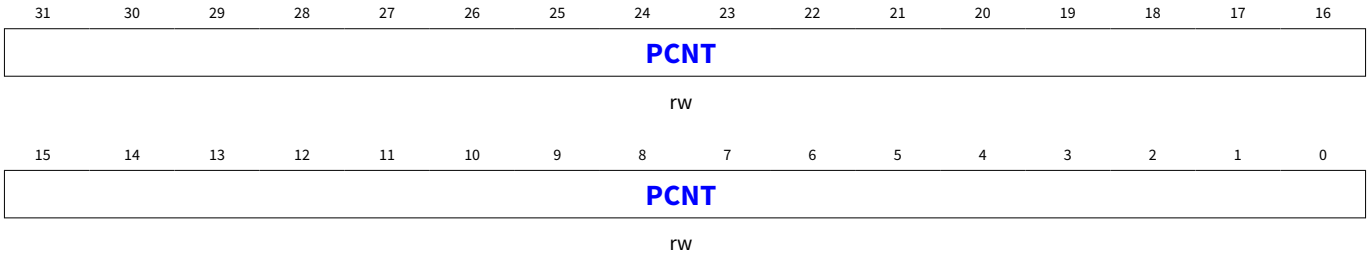
Field	Bits	Type	Description
PCNT	31:0	rw	**Policing Count Value** This field contains count value for policing counters. The counter is incremented within the start and end of the time window. Filter events results in increment of count. - Time window mode: When FRZ is not set, the counter value is cleared when start of the time window is detected. When FRZ is set, the counter is cleared on read. - Free running mode: Counter is cleared on read irrespective of the value of the FRZ bit

14.6.285 Port j MAC PCntr IndReg11

This register contains count value of policing counters.

Portj_MAC_PCntr_IndReg11 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC PCntr IndReg11 Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
PCNT	31:0	rw	**Policing Count Value** This field contains count value for policing counters. The counter is incremented within the start and end of the time window. Filter events results in increment of count. - Time window mode: When FRZ is not set, the counter value is cleared when start of the time window is detected. When FRZ is set, the counter is cleared on read. - Free running mode: Counter is cleared on read irrespective of the value of the FRZ bit

14.6.286 Port j MAC PCntr IndReg12

This register contains count value of policing counters.

Portj_MAC_PCntr_IndReg12 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC PCntr IndReg12 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PCNT															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PCNT															
rw															

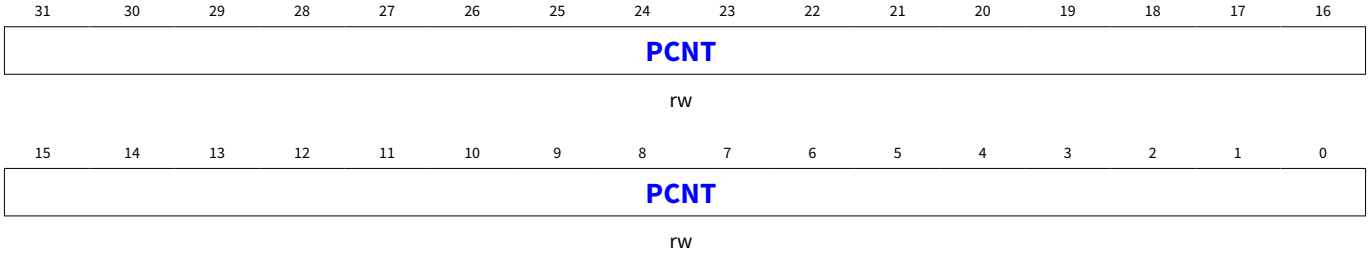
Field	Bits	Type	Description
PCNT	31:0	rw	**Policing Count Value** This field contains count value for policing counters. The counter is incremented within the start and end of the time window. Filter events results in increment of count. - Time window mode: When FRZ is not set, the counter value is cleared when start of the time window is detected. When FRZ is set, the counter is cleared on read. - Free running mode: Counter is cleared on read irrespective of the value of the FRZ bit

14.6.287 Port j MAC PCntr IndReg13

This register contains count value of policing counters.

Portj_MAC_PCntr_IndReg13 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC PCntr IndReg13 Kernel Reset value: $0000\ 0000_H$



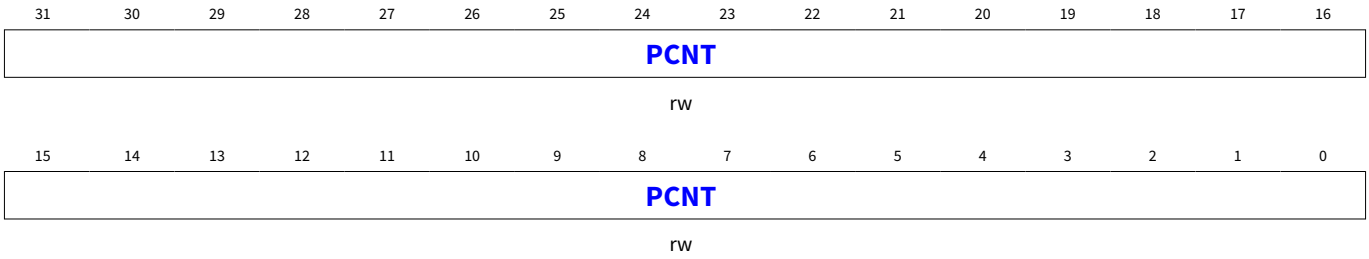
Field	Bits	Type	Description
PCNT	31:0	rw	**Policing Count Value** This field contains count value for policing counters. The counter is incremented within the start and end of the time window. Filter events results in increment of count. - Time window mode: When FRZ is not set, the counter value is cleared when start of the time window is detected. When FRZ is set, the counter is cleared on read. - Free running mode: Counter is cleared on read irrespective of the value of the FRZ bit

14.6.288 Port j MAC PCntr IndReg14

This register contains count value of policing counters.

Portj_MAC_PCntr_IndReg14 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC PCntr IndReg14 Kernel Reset value: $0000\ 0000_H$



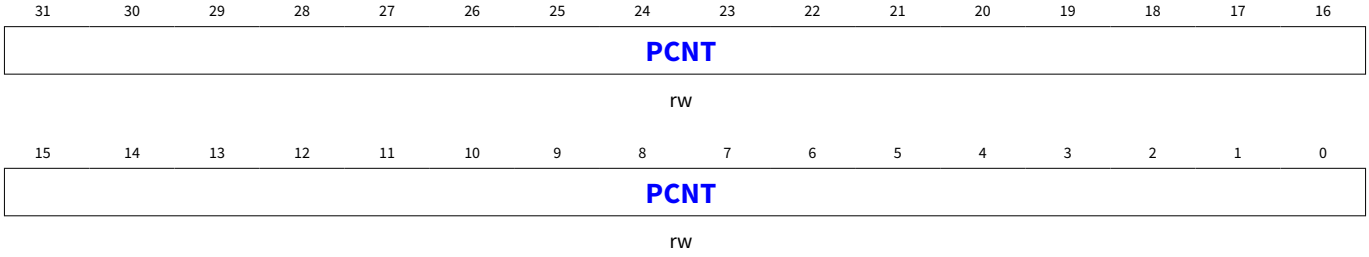
Field	Bits	Type	Description
PCNT	31:0	rw	**Policing Count Value** This field contains count value for policing counters. The counter is incremented within the start and end of the time window. Filter events results in increment of count. - Time window mode: When FRZ is not set, the counter value is cleared when start of the time window is detected. When FRZ is set, the counter is cleared on read. - Free running mode: Counter is cleared on read irrespective of the value of the FRZ bit

14.6.289 Port j MAC PCntr IndReg15

This register contains count value of policing counters.

Portj_MAC_PCntr_IndReg15 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC PCntr IndReg15 Kernel Reset value: $0000\ 0000_H$



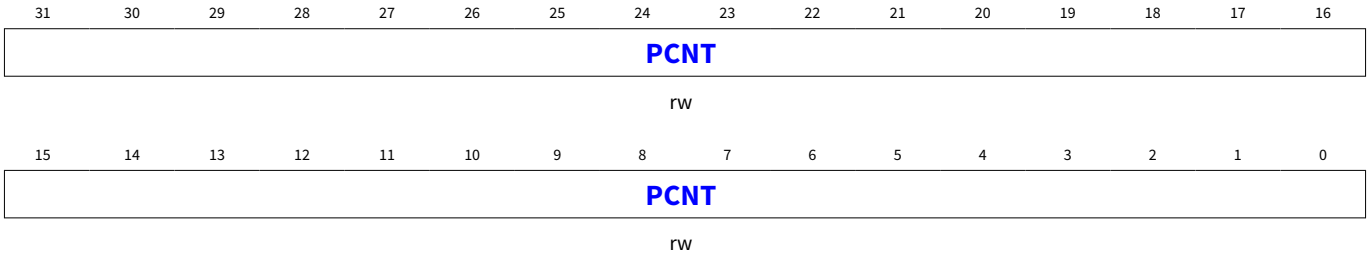
Field	Bits	Type	Description
PCNT	31:0	rw	**Policing Count Value** This field contains count value for policing counters. The counter is incremented within the start and end of the time window. Filter events results in increment of count. - Time window mode: When FRZ is not set, the counter value is cleared when start of the time window is detected. When FRZ is set, the counter is cleared on read. - Free running mode: Counter is cleared on read irrespective of the value of the FRZ bit

14.6.290 Port j MAC PCntr IndReg2

This register contains count value of policing counters.

Portj_MAC_PCntr_IndReg2 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC PCntr IndReg2 Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
PCNT	31:0	rw	**Policing Count Value** This field contains count value for policing counters. The counter is incremented within the start and end of the time window. Filter events results in increment of count. - Time window mode: When FRZ is not set, the counter value is cleared when start of the time window is detected. When FRZ is set, the counter is cleared on read. - Free running mode: Counter is cleared on read irrespective of the value of the FRZ bit

14.6.291 Port j MAC PCntr IndReg3

This register contains count value of policing counters.

Portj_MAC_PCntr_IndReg3 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC PCntr IndReg3 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PCNT															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PCNT															
rw															

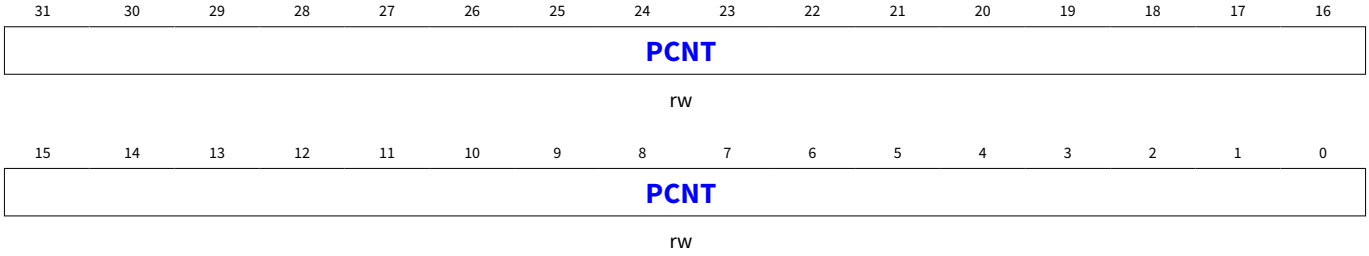
Field	Bits	Type	Description
PCNT	31:0	rw	**Policing Count Value** This field contains count value for policing counters. The counter is incremented within the start and end of the time window. Filter events results in increment of count. - Time window mode: When FRZ is not set, the counter value is cleared when start of the time window is detected. When FRZ is set, the counter is cleared on read. - Free running mode: Counter is cleared on read irrespective of the value of the FRZ bit

14.6.292 Port j MAC PCntr IndReg4

This register contains count value of policing counters.

Portj_MAC_PCntr_IndReg4 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC PCntr IndReg4 Kernel Reset value: $0000\ 0000_H$



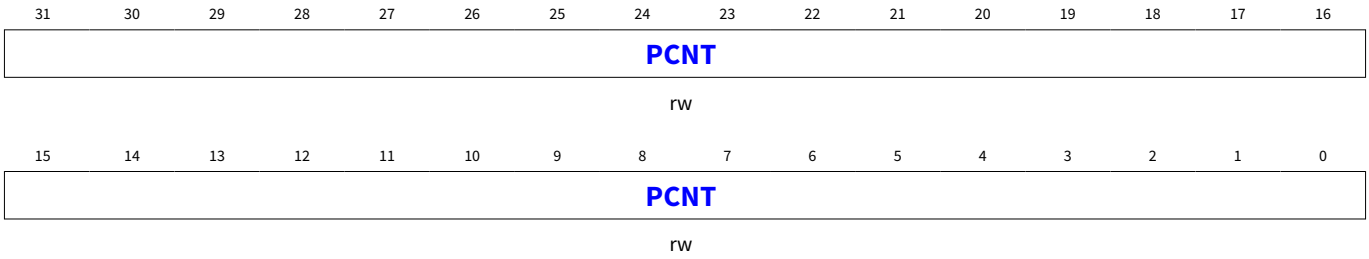
Field	Bits	Type	Description
PCNT	31:0	rw	**Policing Count Value** This field contains count value for policing counters. The counter is incremented within the start and end of the time window. Filter events results in increment of count. - Time window mode: When FRZ is not set, the counter value is cleared when start of the time window is detected. When FRZ is set, the counter is cleared on read. - Free running mode: Counter is cleared on read irrespective of the value of the FRZ bit

14.6.293 Port j MAC PCntr IndReg5

This register contains count value of policing counters.

Portj_MAC_PCntr_IndReg5 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC PCntr IndReg5 Kernel Reset value: $0000\ 0000_H$



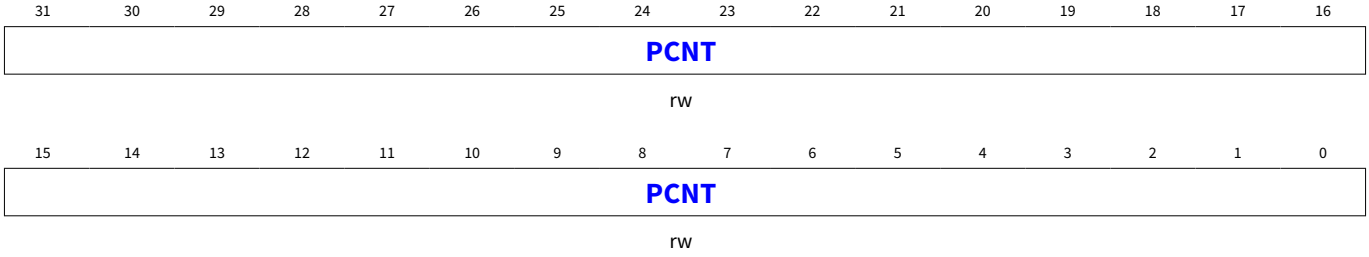
Field	Bits	Type	Description
PCNT	31:0	rw	**Policing Count Value** This field contains count value for policing counters. The counter is incremented within the start and end of the time window. Filter events results in increment of count. - Time window mode: When FRZ is not set, the counter value is cleared when start of the time window is detected. When FRZ is set, the counter is cleared on read. - Free running mode: Counter is cleared on read irrespective of the value of the FRZ bit

14.6.294 Port j MAC PCntr IndReg6

This register contains count value of policing counters.

Portj_MAC_PCntr_IndReg6 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC PCntr IndReg6 Kernel Reset value: $0000\ 0000_H$



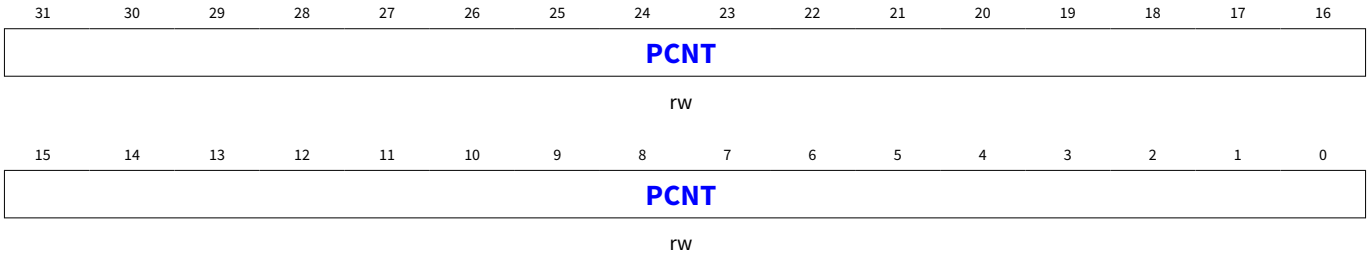
Field	Bits	Type	Description
PCNT	31:0	rw	**Policing Count Value** This field contains count value for policing counters. The counter is incremented within the start and end of the time window. Filter events results in increment of count. - Time window mode: When FRZ is not set, the counter value is cleared when start of the time window is detected. When FRZ is set, the counter is cleared on read. - Free running mode: Counter is cleared on read irrespective of the value of the FRZ bit

14.6.295 Port j MAC PCntr IndReg7

This register contains count value of policing counters.

Portj_MAC_PCntr_IndReg7 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC PCntr IndReg7 Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
PCNT	31:0	rw	**Policing Count Value** This field contains count value for policing counters. The counter is incremented within the start and end of the time window. Filter events results in increment of count. - Time window mode: When FRZ is not set, the counter value is cleared when start of the time window is detected. When FRZ is set, the counter is cleared on read. - Free running mode: Counter is cleared on read irrespective of the value of the FRZ bit

14.6.296 Port j MAC PCntr IndReg8

This register contains count value of policing counters.

Portj_MAC_PCntr_IndReg8 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC PCntr IndReg8 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PCNT															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PCNT															
rw															

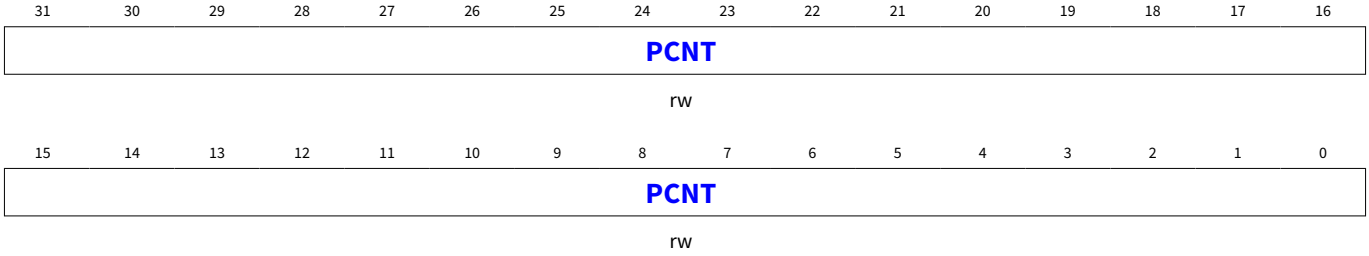
Field	Bits	Type	Description
PCNT	31:0	rw	**Policing Count Value** This field contains count value for policing counters. The counter is incremented within the start and end of the time window. Filter events results in increment of count. - Time window mode: When FRZ is not set, the counter value is cleared when start of the time window is detected. When FRZ is set, the counter is cleared on read. - Free running mode: Counter is cleared on read irrespective of the value of the FRZ bit

14.6.297 Port j MAC PCntr IndReg9

This register contains count value of policing counters.

Portj_MAC_PCntr_IndReg9 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC PCntr IndReg9 Kernel Reset value: $0000\ 0000_H$



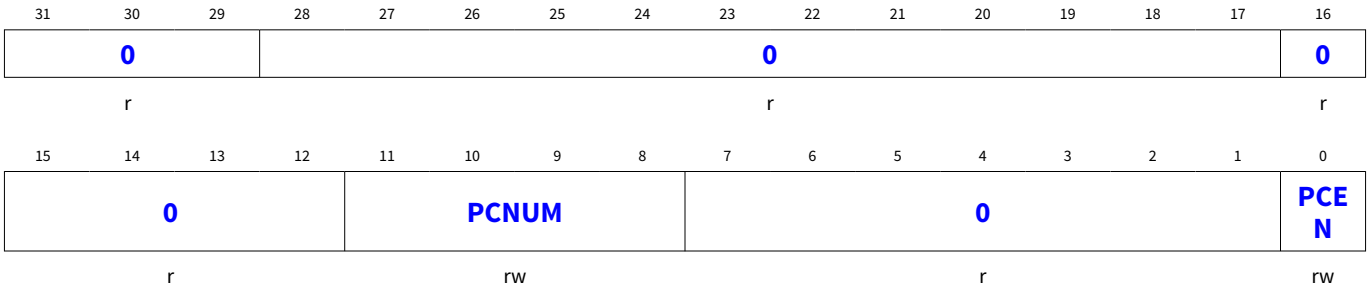
Field	Bits	Type	Description
PCNT	31:0	rw	**Policing Count Value** This field contains count value for policing counters. The counter is incremented within the start and end of the time window. Filter events results in increment of count. - Time window mode: When FRZ is not set, the counter value is cleared when start of the time window is detected. When FRZ is set, the counter is cleared on read. - Free running mode: Counter is cleared on read irrespective of the value of the FRZ bit

14.6.298 Port j MAC VPCSel IndReg0

This register contains policing counter number corresponding to each VLAN filter event.

Portj_MAC_VPCSel_IndReg0 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC VPCSel IndReg0 Kernel Reset value: $0000\ 0000_H$



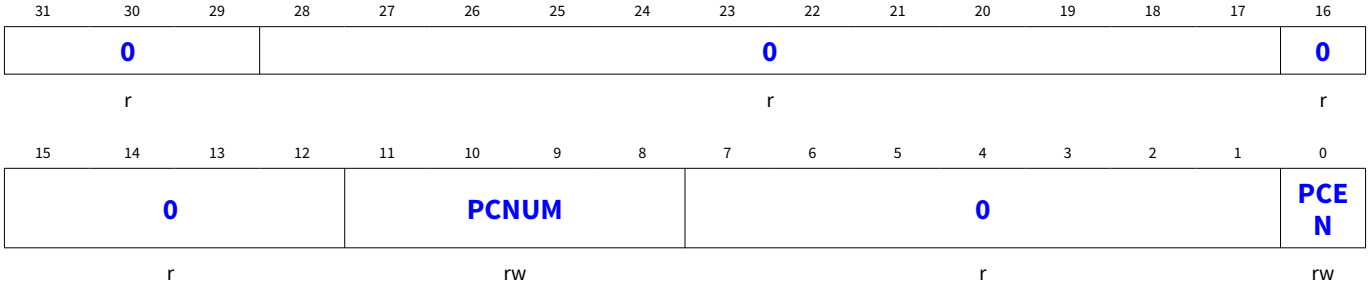
Field	Bits	Type	Description
PCEN	0	rw	**Policing Counter Enable** When set enables the policing counter indicated by the PCNUM field
PCNUM	11:8	rw	**Policing Counter Number** This field specifies the counter number selected for corresponding VLAN filter event.
0	7:1, 15:12, 16, 28:17, 31:29	r	**Reserved** Read as 0; should be written with 0.

14.6.299 Port j MAC VPCSel IndReg1

This register contains policing counter number corresponding to each VLAN filter event.

Portj_MAC_VPCSel_IndReg1 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC VPCSel IndReg1 Kernel Reset value: $0000\ 0000_H$



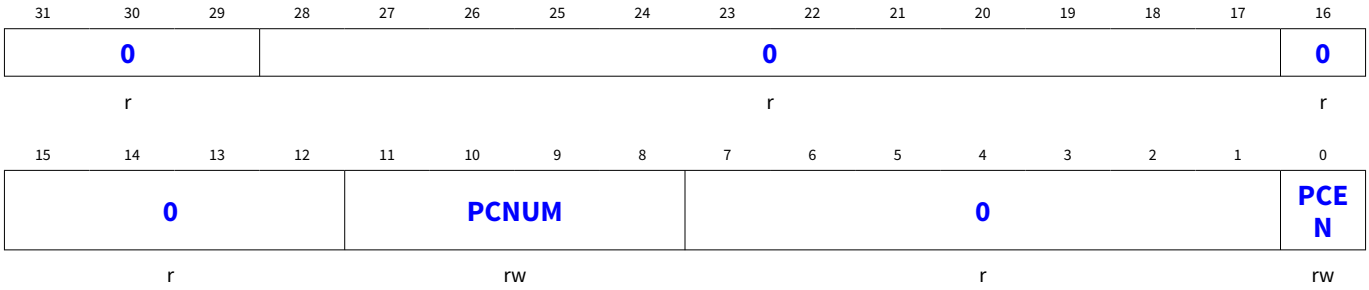
Field	Bits	Type	Description
PCEN	0	rw	**Policing Counter Enable** When set enables the policing counter indicated by the PCNUM field
PCNUM	11:8	rw	**Policing Counter Number** This field specifies the counter number selected for corresponding VLAN filter event.
0	7:1, 15:12, 16, 28:17, 31:29	r	**Reserved** Read as 0; should be written with 0.

14.6.300 Port j MAC VPCSel IndReg2

This register contains policing counter number corresponding to each VLAN filter event.

Portj_MAC_VPCSel_IndReg2 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC VPCSel IndReg2 Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
PCEN	0	rw	**Policing Counter Enable** When set enables the policing counter indicated by the PCNUM field

(table continues...)

(continued)

Field	Bits	Type	Description
PCNUM	11:8	rw	**Policing Counter Number** This field specifies the counter number selected for corresponding VLAN filter event.
0	7:1, 15:12, 16, 28:17, 31:29	r	**Reserved** Read as 0; should be written with 0.

14.6.301 Port j MAC VPCSel IndReg3

This register contains policing counter number corresponding to each VLAN filter event.

Portj_MAC_VPCSel_IndReg3 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC VPCSel IndReg3 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0			0												0
r			r												r

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				PCNUM				0							PCEN
r				rw				r							rw

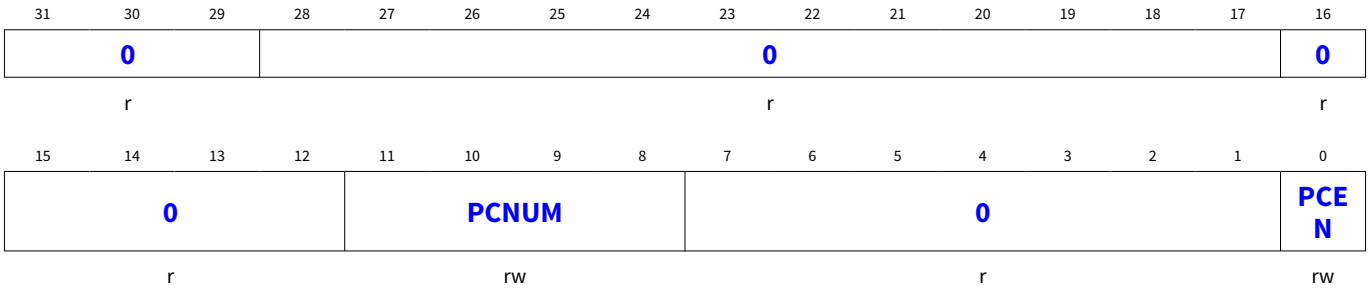
Field	Bits	Type	Description
PCEN	0	rw	**Policing Counter Enable** When set enables the policing counter indicated by the PCNUM field
PCNUM	11:8	rw	**Policing Counter Number** This field specifies the counter number selected for corresponding VLAN filter event.
0	7:1, 15:12, 16, 28:17, 31:29	r	**Reserved** Read as 0; should be written with 0.

14.6.302 Port j MAC VPCSel IndReg4

This register contains policing counter number corresponding to each VLAN filter event.

Portj_MAC_VPCSel_IndReg4 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC VPCSel IndReg4 Kernel Reset value: $0000\ 0000_H$



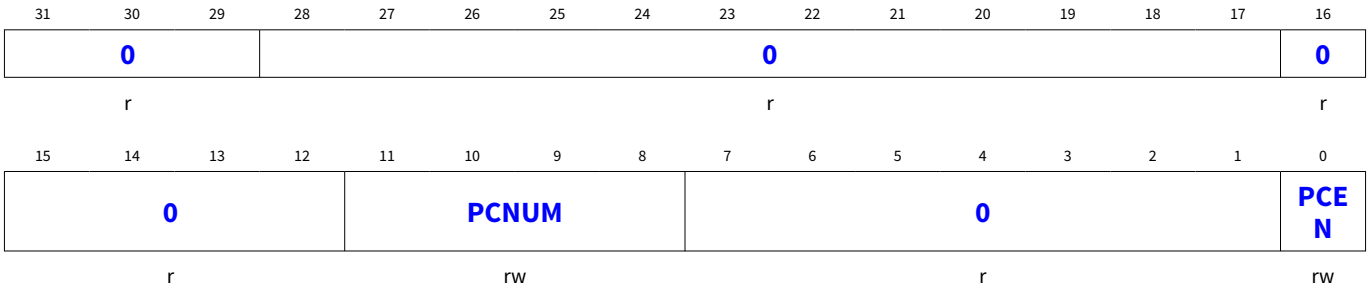
Field	Bits	Type	Description
PCEN	0	rw	**Policing Counter Enable** When set enables the policing counter indicated by the PCNUM field
PCNUM	11:8	rw	**Policing Counter Number** This field specifies the counter number selected for corresponding VLAN filter event.
0	7:1, 15:12, 16, 28:17, 31:29	r	**Reserved** Read as 0; should be written with 0.

14.6.303 Port j MAC VPCSel IndReg5

This register contains policing counter number corresponding to each VLAN filter event.

Portj_MAC_VPCSel_IndReg5 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC VPCSel IndReg5 Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
PCEN	0	rw	**Policing Counter Enable** When set enables the policing counter indicated by the PCNUM field
PCNUM	11:8	rw	**Policing Counter Number** This field specifies the counter number selected for corresponding VLAN filter event.

(table continues...)

(continued)

Field	Bits	Type	Description
0	7:1, 15:12, 16, 28:17, 31:29	r	**Reserved** Read as 0; should be written with 0.

14.6.304 Port j MAC VPCSel IndReg6

This register contains policing counter number corresponding to each VLAN filter event.

Portj_MAC_VPCSel_IndReg6 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC VPCSel IndReg6 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0			0												0
r			r												r

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				PCNUM				0							PCEN
r				rw				r							rw

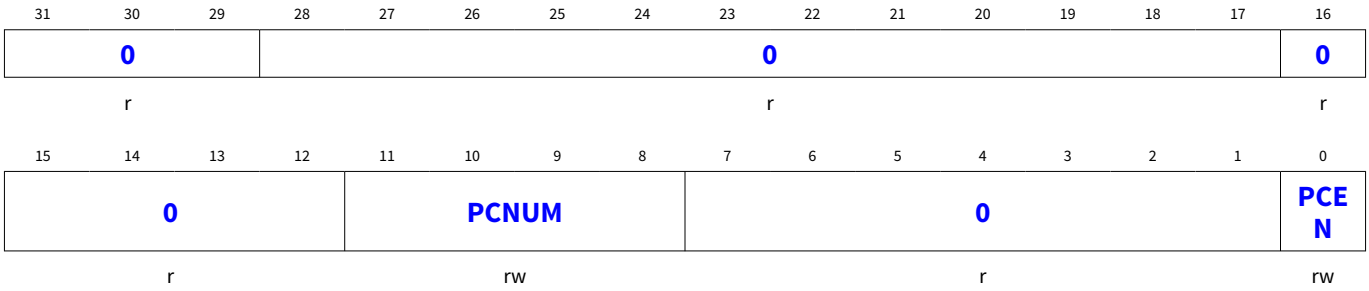
Field	Bits	Type	Description
PCEN	0	rw	**Policing Counter Enable** When set enables the policing counter indicated by the PCNUM field
PCNUM	11:8	rw	**Policing Counter Number** This field specifies the counter number selected for corresponding VLAN filter event.
0	7:1, 15:12, 16, 28:17, 31:29	r	**Reserved** Read as 0; should be written with 0.

14.6.305 Port j MAC VPCSel IndReg7

This register contains policing counter number corresponding to each VLAN filter event.

Portj_MAC_VPCSel_IndReg7 (j=0-1) Offset address: $10704_H+j*2000_H$

Port j MAC VPCSel IndReg7 Kernel Reset value: $0000\ 0000_H$



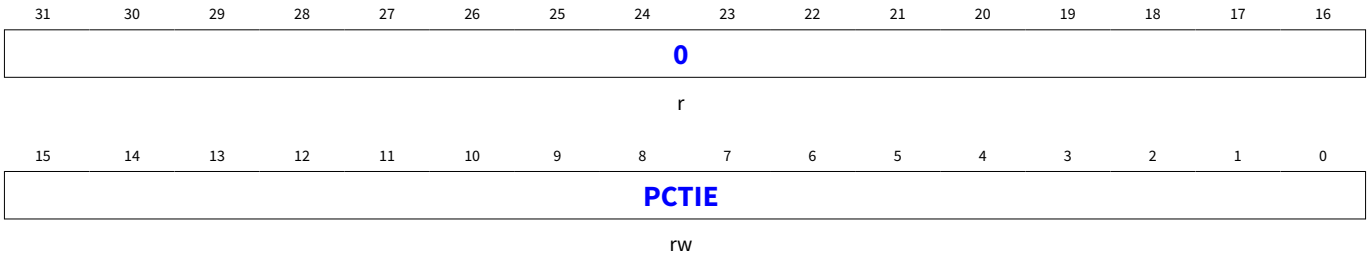
Field	Bits	Type	Description
PCEN	0	rw	**Policing Counter Enable** When set enables the policing counter indicated by the PCNUM field
PCNUM	11:8	rw	**Policing Counter Number** This field specifies the counter number selected for corresponding VLAN filter event.
0	7:1, 15:12, 16, 28:17, 31:29	r	**Reserved** Read as 0; should be written with 0.

14.6.306 Port j MAC PCTH Intr Enable

This register contains threshold based interrupt enable for corresponding policing counters.

Portj_MAC_PCTH_Intr_Enable (j=0-1) Offset address: $10708_H+j*2000_H$

Port j MAC PCTH Intr Enable Kernel Reset value: $0000\ 0000_H$



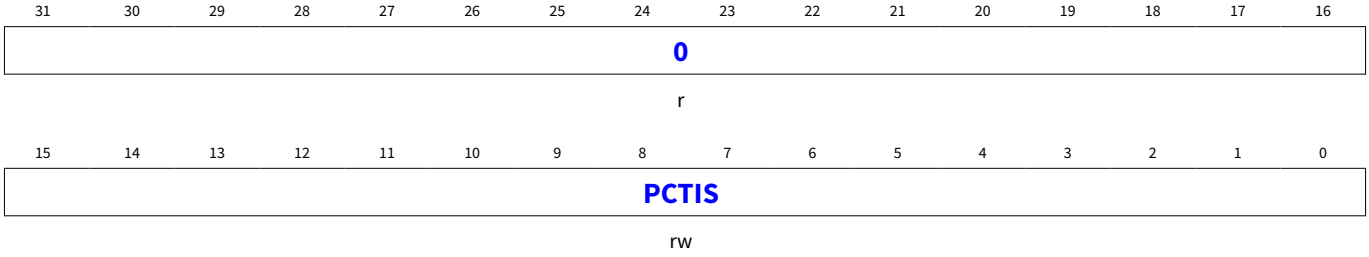
Field	Bits	Type	Description
PCTIE	15:0	rw	**Policing Counter Threshold Interrupt Enable** When set indicates the assertion of the interrupt is enabled for corresponding policing counters.
0	31:16	r	**Reserved** Read as 0; should be written with 0.

14.6.307 Port j MAC PCTH Intr Status

This register contains threshold based interrupts for corresponding policing counters.

Portj_MAC_PCTH_Intr_Status (j=0-1) Offset address: $1070C_H+j*2000_H$

Port j MAC PCTH Intr Status Kernel Reset value: $0000\ 0000_H$



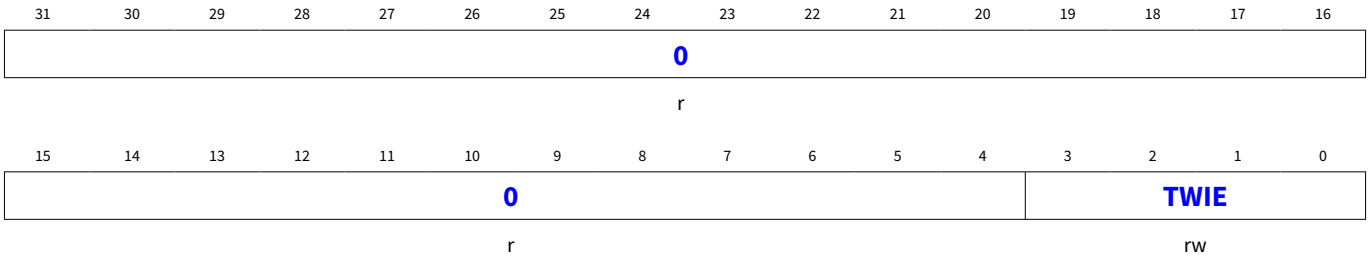
Field	Bits	Type	Description
PCTIS	15:0	rw	**Policing Counter Threshold Interrupt Status** This field is set when corresponding threshold-based interrupt is generated in corresponding policing counter Access restriction applies. This field is cleared when read (or written to 1 when RCWE bit of MAC_CSR_SW_Ctrl register is set).
0	31:16	r	**Reserved** Read as 0; should be written with 0.

14.6.308 Port j MAC PCTW Intr Enable

This register contains enable bits for Time based interrupt.

Portj_MAC_PCTW_Intr_Enable (j=0-1) Offset address: $10730_H+j*2000_H$

Port j MAC PCTW Intr Enable Kernel Reset value: $0000\ 0000_H$



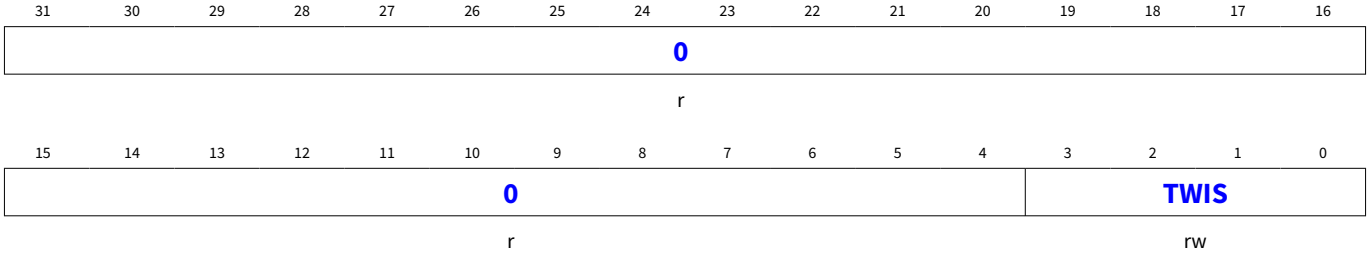
Field	Bits	Type	Description
TWIE	3:0	rw	**Time window Interrupt Enable** When set indicates the assertion of the interrupt is enabled for corresponding Time window.
0	31:4	r	**Reserved** Read as 0; should be written with 0.

14.6.309 Port j MAC PCTW Intr Status

This register contains Time window based interrupt status.

Portj_MAC_PCTW_Intr_Status (j=0-1) Offset address: $10734_H+j*2000_H$

Port j MAC PCTW Intr Status Kernel Reset value: $0000\ 0000_H$



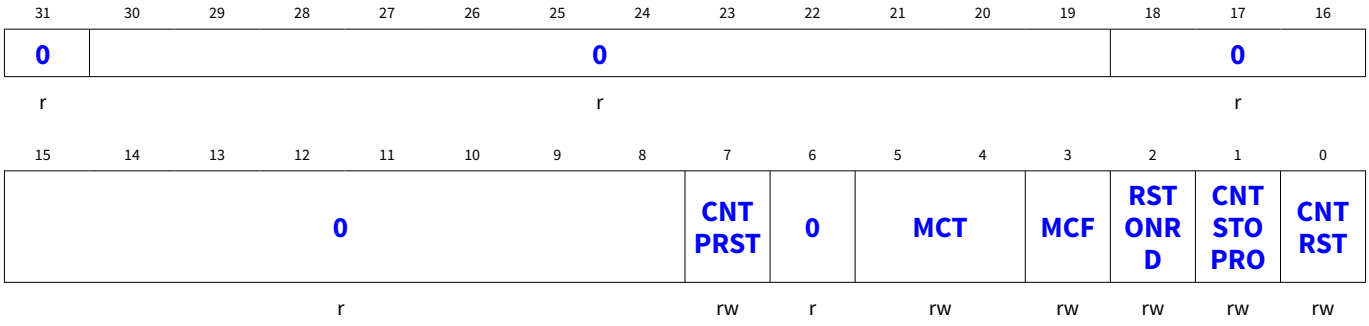
Field	Bits	Type	Description
TWIS	3:0	rw	**Time window Interrupt Status** This field is set when corresponding end of time window is detected. Access restriction applies. This field is cleared when read (or written to 1 when RCWE bit of MAC_CSR_SW_Ctrl register is set).
0	31:4	r	**Reserved** Read as 0; should be written with 0.

14.6.310 Port j MMC Control

The MMC Control register establishes the operating mode of the management counters.

Portj_MMC_Control (j=0-1) Offset address: $10800_H+j*2000_H$

Port j MMC Control Kernel Reset value: $0000\ 0020_H$



Field	Bits	Type	Description
CNTRST	0	rw	**Counters Reset** When this bit is set, all counters are reset. This bit is cleared automatically after 1 clock cycle.

(table continues...)

(continued)

Field	Bits	Type	Description
CNTSTOPRO	1	rw	**Counter Stop Rollover** When this bit is set, no MMC counter rolls over to zero after reaching the maximum count value. The counters are cleared when the least significant byte lane (bits[7:0]) is read after reaching the maximum count value. When the MMC module is enabled for the 64-bit counter mode, the lower and upper counters are cleared only when the least significant byte lane (bits[7:0]) of the upper counter is read after reaching the maximum count value.
RSTONRD	2	rw	**Reset on Read** When this bit is set, the MMC counters are reset to zero after a read. The counters are cleared when the least significant byte lane (bits[7:0]) is read. When the MMC module is enabled for the 64-bit counter mode, the lower and upper counters are cleared only when the least significant byte lane (bits[7:0]) of the corresponding counter is read. Note: When 64-bit counter mode is selected and RSTONRD is set, read the lower counter first followed by the corresponding upper counter. This is to ensure that only the lower 32-bit counter is updated after the first read operation and the software gets the original higher 32-bit counter and gets a consistent value.
MCF	3	rw	**MMC Counter Freeze** When set, this bit freezes the MMC counters to their current value. No MMC counter is updated for subsequent packet transmission or reception until this bit is reset to 0. If any MMC counter is read with the Reset on Read bit set, that counter is also cleared when this bit is set.
MCT	5:4	rw	**MMC Counter Threshold** Threshold value for the counter which, when crossed, causes the interrupt to be scheduled - 2'b00: Reserved - 2'b01: One-quarter the maximum value - X - 2'b10: One-half of the maximum value - X - 2'b11: Three-quarters of the maximum value - X Here, - X = 16 for all the Packet counters and - X = 2048 for all the Octet/Byte counters
CNTPRST	7	rw	**Counters Preset** When this bit is set, all counters are initialized or preset to a value depending on Bits [5:4]. This bit is cleared after 1 clock cycle. This bit, along with Bits [5:4], is useful for debugging and testing the assertion of interrupts because of MMC counter becoming one-quarter-full/half-full/three-quarter-full. If this bit is set along with CNTRST bit, CNTRST bit has precedence.
0	6, 15:8, 18:16, 30:19, 31	r	**Reserved** Read as 0; should be written with 0.

14.6.311 Port j MMC Rx Interrupt

The MMC Receive Interrupt Status register maintains the interrupts generated when receive statistic counters reach the threshold programmed in the MMC control register. When the MMC counter that caused the interrupt is read, the corresponding interrupt status bit is cleared. The least significant byte lane (bits[7:0]) of the counter must be read to clear the interrupt status bit. When the MMC module is enabled for 64-bit counter mode, the least significant byte lane (bits[7:0]) of the upper counter must be read to clear the interrupt status bit.

Portj_MMC_Rx_Interrupt (j=0-1) Offset address: $10804_H+j*2000_H$

Port j MMC Rx Interrupt Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RXPRMMCIS	0	SGFPIS	SGPPIS	RXALEPIS	RXLPITRCIS	RXLPIUSCIS	RXDISOCGBIS	RXDISPCGBIS	RXWDOGPIS	RXVLANGBPIS	RXFOVPIS	RXPAUSPIS	RXORANGEPIS	RXLENERPIS	RXUCGPIS
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RX1024TMAXOCTGBPIS	RX512T1023OCTGBPIS	RX256T511OCTGBPIS	RX128T255OCTGBPIS	RX65T127OCTGBPIS	RX64OCTGBPIS	RXOSIZEGPIS	RXUSIZEGPIS	RXJABERPIS	RXRUNTPIS	RXCRCERPIS	RXMCGPIS	RXBCGPIS	RXGOCTIS	RXGBOCTIS	RXGBPKTIS
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

Field	Bits	Type	Description
RXGBPKTIS	0	r	**Rx Good Bad Packet Counter Interrupt Status** The bit is set when the 64-bit Rx_Packet_Count_Good_Bad_High and Rx_Packet_Count_Good_Bad_Low counter or 32-bit Rx_Packet_Count_Good_Bad_Low counter reaches the threshold value programmed in the MMC control register.
RXGBOCTIS	1	r	**Rx Good Bad Octet Counter Interrupt Status** The bit is set when the Rx_Octet_Count_Good_Bad_High and Rx_Octet_Count_Good_Bad_Low counter reaches the threshold value programmed in the MMC control register.
RXGOCTIS	2	r	**Rx Good Octet Counter Interrupt Status** The bit is set when the Rx_Octet_Count_Good_High and Rx_Octet_Count_Good_Low counter reaches the threshold value programmed in the MMC control register.
RXBCGPIS	3	r	**Rx Good Broadcast Packet Counter Interrupt Status** The bit is set when the 64-bit Rx_Broadcast_Packets_Good_High and Rx_Broadcast_Packets_Good_Low counter or 32-bit Rx_Broadcast_Packets_Good_Low counter reaches the threshold value programmed in the MMC control register.
RXMCGPIS	4	r	**Rx Good Multicast Packet Counter Interrupt Status** The bit is set when the 64-bit Rx_Multicast_Packets_Good_High and Rx_Multicast_Packets_Good_Low counter or 32-bit Rx_Multicast_Packets_Good_Low counter reaches the threshold value programmed in the MMC control register.

(table continues...)

(continued)

Field	Bits	Type	Description
RXCRCERPIS	5	r	**Rx CRC Error Packet Counter Interrupt Status** The bit is set when the 64-bit Rx_CRC_Error_Packets_High and Rx_CRC_Error_Packets_Low counter or 32-bit Rx_CRC_Error_Packets_Low reaches the threshold value programmed in the MMC control register.
RXRUNTPIS	6	r	**Rx Runt Error Packet Counter Interrupt Status** This bit is set when the Rx_Runt_Error_Packets counter reaches the threshold value programmed in the MMC control register.
RXJABERPIS	7	r	**Rx Jabber Error Packet Counter Interrupt Status** The bit is set when the Rx_Jabber_Error_Packets counter reaches the threshold value programmed in the MMC control register.
RXUSIZEGPIS	8	r	**Rx Good Undersize Packet Counter Interrupt Status** The bit is set when the Rx_Undersize_Packets_Good counter reaches the threshold value programmed in the MMC control register.
RXOSIZEGPIS	9	r	**Rx Good Oversize Packet Counter Interrupt Status** The bit is set when the Rx_Oversize_Packets_Good counter reaches the threshold value programmed in the MMC control register.
RX64OCTGBPIS	10	r	**Rx Good Bad 64 Octet Size Packet Counter Interrupt Status** The bit is set when the 64-bit Rx_64Octets_Packets_Good_Bad_High and Rx_64Octets_Packets_Good_Bad_Low counter or 32-bit Rx_64Octets_Packets_Good_Bad_Low counter reaches the threshold value programmed in the MMC control register.
RX65T127OCTGBPIS	11	r	**Rx Good Bad 65-to-127 Octet Size Packet Counter Interrupt Status** The bit is set when the 64-bit Rx_65To127Octets_Packets_Good_Bad_High and Rx_65To127Octets_Packets_Good_Bad_Low counter or 32-bit Rx_65To127Octets_Packets_Good_Bad_Low counter reaches the threshold value programmed in the MMC control register.
RX128T255OCTGBPIS	12	r	**Rx Good Bad 128-to-255 Octet Size Packet Counter Interrupt Status** The bit is set when the 64-bit Rx_128To255Octets_Packets_Good_Bad_High and Rx_128To255Octets_Packets_Good_Bad_Low counter or 32-bit Rx_128To255Octets_Packets_Good_Bad_Low counter reaches the threshold value programmed in the MMC control register.
RX256T511OCTGBPIS	13	r	**Rx Good Bad 256-to-511 Octet Size Packet Counter Interrupt Status** The bit is set when the 64-bit Rx_256To511Octets_Packets_Good_Bad_High and Rx_256To511Octets_Packets_Good_Bad_Low counter or 32-bit Rx_256To511Octets_Packets_Good_Bad_Low counter reaches the threshold value programmed in the MMC control register.
RX512T1023OCTGBPIS	14	r	**Rx Good Bad 512-to-1023 Octet Size Packet Counter Interrupt Status** The bit is set when the 64-bit Rx_512To1023Octets_Packets_Good_Bad_High and Rx_512To1023Octets_Packets_Good_Bad_Low counter or 32-bit Rx_512To1023Octets_Packets_Good_Bad_Low counter reaches the threshold value programmed in the MMC control register.

(table continues...)

(continued)

Field	Bits	Type	Description
RX1024TMAXOCTGBPIS	15	r	**Rx Good Bad 1024-to-Max Octet Size Packet Counter Interrupt Status** The bit is set when the 64-bit Rx_1024ToMaxOctets_Packets_Good_Bad_High and Rx_1024ToMaxOctets_Packets_Good_Bad_Low counter or 32-bit Rx_1024ToMaxOctets_Packets_Good_Bad_Low counter reaches the threshold value programmed in the MMC control register.
RXUCGPIS	16	r	**Rx Good Unicast Packet Counter Interrupt Status** The bit is set when the 64-bit Rx_Unicast_Packets_Good_High and Rx_Unicast_Packets_Good_Low counter or 32-bit Rx_Unicast_Packets_Good_Low counter reaches the threshold value programmed in the MMC control register.
RXLENERPIS	17	r	**Rx Length Error Packet Counter Interrupt Status** The bit is set when the 64-bit Rx_Length_Error_Packets_High and Rx_Length_Error_Packets_Low counter or 32-bit Rx_Length_Error_Packets_Low counter reaches the threshold value programmed in the MMC control register.
RXORANGEPIS	18	r	**Rx Out-of-Range Packet Counter Interrupt Status** The bit is set when the 64-bit Rx_OutofRange_Packets_High and Rx_OutofRange_Packets_Low counter or 32-bit Rx_OutofRange_Packets_Low counter reaches the threshold value programmed in the MMC control register.
RXPAUSPIS	19	r	**Rx Pause Packet Counter Interrupt Status** The bit is set when the 64-bit Rx_Pause_Packets_High and Rx_Pause_Packets_Low counter or 32-bit Rx_Pause_Packets_Low counter reaches the threshold value programmed in the MMC control register.
RXFOVPIS	20	r	**Rx Queue Overflow Packet Counter Interrupt Status** The bit is set when the 64-bit Rx_FIFOOverflow_Packets_High and Rx_FIFOOverflow_Packets_Low counter or 32-bit Rx_FIFOOverflow_Packets_Low counter reaches the threshold value programmed in the MMC control register.
RXVLANGBPIS	21	r	**Rx Good Bad VLAN Packet Counter Interrupt Status** The bit is set when the 64-bit Rx_VLAN_Packets_Good_Bad_High and Rx_VLAN_Packets_Good_Bad_Low counter or 32-bit Rx_VLAN_Packets_Good_Bad_Low counter reaches the threshold value programmed in the MMC control register.
RXWDOGPIS	22	r	**Rx Watchdog Error Packet Counter Interrupt Status** The bit is set when the Rx_Watchdog_Error_Packets counter reaches the threshold value programmed in the MMC control register.
RXDISPCGBIS	23	r	**Rx Good Bad Discarded Packet Counter Interrupt Status** This bit is set when the 64-bit Rx_Discard_Packets_Good_Bad_High and Rx_Discard_Packets_Good_Bad_Low counter or 32-bit Rx_Discard_Packets_Good_Bad_Low counter reaches the threshold value programmed in the MMC control register.

(table continues...)

(continued)

Field	Bits	Type	Description
RXDISOCGBIS	24	r	**Rx Good Bad Discarded Octet Counter Interrupt Status** This bit is set when the Rx_Discard_Octets_Good_Bad_High and Rx_Discard_Octets_Good_Bad_Low counter reaches the threshold value programmed in the MMC control register.
RXLPIUSCIS	25	r	**Rx LPI microsecond Counter Interrupt Status** This bit is set when the Rx_LPI_USEC_Cntr reaches the threshold value programed in the MMC control register.
RXLPITRCIS	26	r	**Rx LPI Transition Counter Interrupt Status** This bit is set when the Rx_LPI_Tran_Cntr reaches the threshold value programed in the MMC control register.
RXALEPIS	27	r	**Rx Alignment Error Packet Counter Interrupt Status** This bit is set when the Rx_Alignment_Error_Packets counter reaches the threshold value programed in the MMC control register.
SGPPIS	28	r	**SGF Pass Packet Counter Interrupt Status** The bit is set when the SGF Pass Packet Counter reaches the threshold value programmed in the MMC control register.
SGFPIS	29	r	**SGF Fail Packet Counter Interrupt Status** The bit is set when the and when the SGF Fail Packet Counter reaches the threshold value programmed in the MMC control register.
RXPRMMCIS	31	r	**Rx Priority MMC Counter Interrupt Status** The bit is set when the any of the Rx Per Priority MMC counter (0x09D0 to 0x09FC) reaches the threshold value programmed in the MMC control register.
0	30	r	**Reserved** Read as 0; should be written with 0.

14.6.312 Port j MMC Tx Interrupt

The MMC Transmit Interrupt Status register maintains the interrupts generated when the transmit statistic counters reach the threshold value programmed in MMC control register (for example, the counter's MSB is set). This register is 32 bits wide. When the MMC counter that caused the interrupt is read, the corresponding interrupt status bit is cleared. The least significant byte lane (bits[7:0]) of the corresponding counter must be read to clear the interrupt status bit. When the MMC module is enabled for 64-bit counter mode, the least significant byte lane (bits[7:0]) of the upper counter must be read to clear the interrupt status bit.

Portj_MMC_Tx_Interrupt (j=0-1) Offset address: $10808_H+j*2000_H$

Port j MMC Tx Interrupt Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TXPRMMCIS	0				TXEXDEFPIS	TXCARERPIS	TXEXCOLPIS	TXLATCOLPIS	TXDEFPIS	TXMCOLGPIS	TXSCOLGPIS	TXLPITRCIS	TXLPIUSCIS	TXVLANGPIS	TXPAUSPIS
r	r				r	r	r	r	r	r	r	r	r	r	r

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXGPKTIS	TXGOCTIS	TXUFLOWERPIS	TXBCGBPIS	TXMCGBPIS	TXUCGBPIS	TX1024TMAXOCTGBPIS	TX512T1023OCTGBPIS	TX256T511OCTGBPIS	TX128T255OCTGBPIS	TX65T127OCTGBPIS	TX64OCTGBPIS	TXMCGPIS	TXBCGPIS	TXGBPKTIS	TXGBOCTIS
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

Field	Bits	Type	Description
TXGBOCTIS	0	r	**Tx Good Bad Octet Counter Interrupt Status** The bit is set when the Tx_Octet_Count_Good_Bad_High and Tx_Octet_Count_Good_Bad_Low counter reaches the threshold value programmed in the MMC control register.
TXGBPKTIS	1	r	**Tx Good Bad Packet Counter Interrupt Status** The bit is set when the 64-bit Tx_Packet_Count_Good_Bad_High and Tx_Packet_Count_Good_Bad_Low counter or 32-bit Tx_Packet_Count_Good_Bad_Low counter reaches the threshold value programmed in the MMC control register.
TXBCGPIS	2	r	**Tx Good Broadcast Packet Counter Interrupt Status** The bit is set when the 64-bit Tx_Broadcast_Packets_Good_High and Tx_Broadcast_Packets_Good_Low counter or 32-bit Tx_Broadcast_Packets_Good_Low counter reaches the threshold value programmed in the MMC control register.
TXMCGPIS	3	r	**Tx Good Multicast Packet Counter Interrupt Status** The bit is set when the 64-bit Tx_Multicast_Packets_Good_High and Tx_Multicast_Packets_Good_Low counter or 32-bit Tx_Multicast_Packets_Good_Low counter reaches the threshold value programmed in the MMC control register.
TX64OCTGBPIS	4	r	**Tx Good Bad 64 Octet Size Packet Counter Interrupt Status** The bit is set when the 64-bit Tx_64Octets_Packets_Good_Bad_High and Tx_64Octets_Packets_Good_Bad_Low counter or 32-bit Tx_64Octets_Packets_Good_Bad_Low counter reaches the threshold value programmed in the MMC control register.
TX65T127OCTGBPIS	5	r	**Tx Good Bad 65-to-127 Octet Size Packet Counter Interrupt Status** The bit is set when the 64-bit Tx_65To127Octets_Packets_Good_Bad_High and Tx_65To127Octets_Packets_Good_Bad_Low counter or 32-bit Tx_65To127Octets_Packets_Good_Bad_Low counter reaches the threshold value programmed in the MMC control register.

(table continues...)

(continued)

Field	Bits	Type	Description
TX128T255OCTGBPIS	6	r	**Tx Good Bad 128-to-255 Octet Size Packet Counter Interrupt Status** The bit is set when the 64-bit Tx_128To255Octets_Packets_Good_Bad_High and Tx_128To255Octets_Packets_Good_Bad_Low counter or 32-bit Tx_128To255Octets_Packets_Good_Bad_Low counter reaches the threshold value programmed in the MMC control register.
TX256T511OCTGBPIS	7	r	**Tx Good Bad 256-to-511 Octet Size Packet Counter Interrupt Status** The bit is set when the 64-bit Tx_256To511Octets_Packets_Good_Bad_High and Tx_256To511Octets_Packets_Good_Bad_Low counter or 32-bit Tx_256To511Octets_Packets_Good_Bad_Low counter reaches the threshold value programmed in the MMC control register.
TX512T1023OCTGBPIS	8	r	**Tx Good Bad 512-to-1023 Octet Size Packet Counter Interrupt Status** The bit is set when the 64-bit Tx_512To1023Octets_Packets_Good_Bad_High and Tx_512To1023Octets_Packets_Good_Bad_Low counter or 32-bit Tx_512To1023Octets_Packets_Good_Bad_Low counter reaches the threshold value programmed in the MMC control register.
TX1024TMAXOCTGBPIS	9	r	**Tx Good Bad 1024-to-Max Octet Size Packet Counter Interrupt Status** The bit is set when the 64-bit Tx_1024ToMaxOctets_Packets_Good_Bad_High and Tx_1024ToMaxOctets_Packets_Good_Bad_Low counter or 32-bit Tx_1024ToMaxOctets_Packets_Good_Bad_Low counter reaches the threshold value programmed in the MMC control register.
TXUCGBPIS	10	r	**Tx Good Bad Unicast Packet Counter Interrupt Status** The bit is set when the 64-bit Tx_Unicast_Packets_Good_Bad_High and Tx_Unicast_Packets_Good_Bad_Low counter or 32-bit Tx_Unicast_Packets_Good_Bad_Low counter reaches the threshold value programmed in the MMC control register.
TXMCGBPIS	11	r	**Tx Good Bad Multicast Packet Counter Interrupt Status** The bit is set when the 64-bit Tx_Multicast_Packets_Good_Bad_High and Tx_Multicast_Packets_Good_Bad_Low counter or 32-bit Tx_Multicast_Packets_Good_Bad_Low counter reaches the threshold value programmed in the MMC control register.
TXBCGBPIS	12	r	**Tx Good Bad Broadcast Packet Counter Interrupt Status** The bit is set when the 64-bit Tx_Broadcast_Packets_Good_Bad_High and Tx_Broadcast_Packets_Good_Bad_Low counter or 32-bit Tx_Broadcast_Packets_Good_Bad_Low counter reaches the threshold value programmed in the MMC control register.
TXUFLOWERPIS	13	r	**Tx Underflow Error Packet Counter Interrupt Status** The bit is set when the 64-bit Tx_Underflow_Error_Packets_High and Tx_Underflow_Error_Packets_Low counter or 32-bit Tx_Underflow_Error_Packets_Low counter reaches the threshold value programmed in the MMC control register.

(table continues...)

(continued)

Field	Bits	Type	Description
TXGOCTIS	14	r	**Tx Good Octet Counter Interrupt Status** The bit is set when the Tx_Octet_Count_Good_High and Tx_Octet_Count_Good_Low counter reaches the threshold value programmed in the MMC control register.
TXGPKTIS	15	r	**Tx Good Packet Counter Interrupt Status** The bit is set when the 64-bit Tx_Packet_Count_Good_High and Tx_Packet_Count_Good_Low counter or 32-bit Tx_Packet_Count_Good_Low counter reaches the threshold value programmed in the MMC control register.
TXPAUSPIS	16	r	**Tx Pause Packet Counter Interrupt Status** The bit is set when the 64-bit Tx_Pause_Packets_High and Tx_Pause_Packets_Low counter or 32-bit Tx_Pause_Packets_Low counter reaches the threshold value programmed in the MMC control register.
TXVLANGPIS	17	r	**Tx Good VLAN Packet Counter Interrupt Status** The bit is set when the 64-bit Tx_VLAN_Packets_Good_High and Tx_VLAN_Packets_Good_Low counter or 32-bit Tx_VLAN_Packets_Good_Low counter reaches the threshold value programmed in the MMC control register.
TXLPIUSCIS	18	r	**Tx LPI microsecond Counter Interrupt Status** This bit is set when the Tx_LPI_USEC_Cntr counter reaches the threshold value programmed in the MMC counter register.
TXLPITRCIS	19	r	**Tx LPI Transition Counter Interrupt Status** This bit is set when the Tx_LPI_Tran_Cntr counter reaches the threshold value programmed in the MMC counter register.
TXSCOLGPIS	20	r	**MMC Transmit Single Collision Good Packet Counter Interrupt Status** This bit is set when the txsinglecol_g counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0_B **INACTIVE**: MMC Transmit Single Collision Good Packet Counter Interrupt Status not detected 1_B **ACTIVE**: MMC Transmit Single Collision Good Packet Counter Interrupt Status detected
TXMCOLGPIS	21	r	**MMC Transmit Multiple Collision Good Packet Counter Interrupt Status** This bit is set when the txmulticol_g counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0_B **INACTIVE**: MMC Transmit Multiple Collision Good Packet Counter Interrupt Status not detected 1_B **ACTIVE**: MMC Transmit Multiple Collision Good Packet Counter Interrupt Status detected

(table continues...)

(continued)

Field	Bits	Type	Description
TXDEFPIS	22	r	**MMC Transmit Deferred Packet Counter Interrupt Status** This bit is set when the txdeferred counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0_B **INACTIVE**: MMC Transmit Deferred Packet Counter Interrupt Status not detected 1_B **ACTIVE**: MMC Transmit Deferred Packet Counter Interrupt Status detected
TXLATCOLPIS	23	r	**MMC Transmit Late Collision Packet Counter Interrupt Status** This bit is set when the txlatecol counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0_B **INACTIVE**: MMC Transmit Late Collision Packet Counter Interrupt Status not detected 1_B **ACTIVE**: MMC Transmit Late Collision Packet Counter Interrupt Status detected
TXEXCOLPIS	24	r	**MMC Transmit Excessive Collision Packet Counter Interrupt Status** This bit is set when the txexesscol counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0_B **INACTIVE**: MMC Transmit Excessive Collision Packet Counter Interrupt Status not detected 1_B **ACTIVE**: MMC Transmit Excessive Collision Packet Counter Interrupt Status detected
TXCARERPIS	25	r	**MMC Transmit Carrier Error Packet Counter Interrupt Status** This bit is set when the txcarriererror counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0_B **INACTIVE**: MMC Transmit Carrier Error Packet Counter Interrupt Status not detected 1_B **ACTIVE**: MMC Transmit Carrier Error Packet Counter Interrupt Status detected
TXEXDEFPIS	26	r	**MMC Transmit Excessive Deferral Packet Counter Interrupt Status** This bit is set when the txexcessdef counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0_B **INACTIVE**: MMC Transmit Excessive Deferral Packet Counter Interrupt Status not detected 1_B **ACTIVE**: MMC Transmit Excessive Deferral Packet Counter Interrupt Status detected

(table continues...)

(continued)

Field	Bits	Type	Description
TXPRMMCIS	31	r	**Tx Priority MMC Counter Interrupt Status** The bit is set when the any of the Tx Per Priority MMC counter (0x08D0 to 0x08FC) reaches the threshold value programmed in the MMC control register.
0	30:27	r	**Reserved** Read as 0; should be written with 0.

14.6.313 Port j MMC Receive Interrupt Enable

The MMC Receive Interrupt Enable register maintains the enables for the interrupts generated when receive statistic counters reach the threshold value programmed in the MMC Control register. This register is 32 bits wide.

Portj_MMC_Receive_Interrupt_Enable (j=0-1) Offset address: $1080C_H+j*2000_H$

Port j MMC Receive Interrupt Enable Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	SGFPIE	SGPPIE	RXALEPIE	RXLPITRCIE	RXLPIUSCIE	RXDISOCGBIE	RXDISPCGBIE	RXWDOGPIE	RXVLANGBPIE	RXFOVPIE	RXPAUSPIE	RXORANGEPIE	RXLENERPIE	RXUCGPIE
r	r	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RX1024TMAXOCTGBPIE	RX512T1023OCTGBPIE	RX256T511OCTGBPIE	RX128T255OCTGBPIE	RX65T127OCTGBPIE	RX64OCTGBPIE	RXOSIZEGPIE	RXUSIZEGPIE	RXJABERPIE	RXRUNTPIE	RXCRCERPIE	RXMCGPIE	RXBCGPIE	RXGOCTIE	RXGBOCTIE	RXGBPKTIE
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RXGBPKTIE	0	rw	**Rx Good Bad Packet Counter Interrupt Enable** Setting this bit enables the interrupt when the 64-bit Rx_Packet_Count_Good_Bad_High and Rx_Packet_Count_Good_Bad_Low counter or 32-bit Rx_Packet_Count_Good_Bad_Low counter reaches the threshold value programmed in the MMC control register.
RXGBOCTIE	1	rw	**Rx Good Bad Octet Counter Interrupt Enable** Setting this bit enables the interrupt when the Rx_Octet_Count_Good_Bad_High and Rx_Octet_Count_Good_Bad_Low counter reaches the threshold value programmed in the MMC control register.
RXGOCTIE	2	rw	**Rx Good Octet Counter Interrupt Enable** Setting this bit enables the interrupt when the Rx_Octet_Count_Good_High and Rx_Octet_Count_Good_Lo counter reaches the threshold value programmed in the MMC control register.

(table continues...)

(continued)

Field	Bits	Type	Description
RXBCGPIE	3	rw	**Rx Good Broadcast Packet Counter Interrupt Enable** Setting this bit enables the interrupt when the 64-bit Rx_Broadcast_Packets_Good_High and Rx_Broadcast_Packets_Good_Low counter or 32-bit Rx_Broadcast_Packets_Good_Low counter reaches the threshold value programmed in the MMC control register.
RXMCGPIE	4	rw	**Rx Good Multicast Packet Counter Interrupt Enable** Setting this bit enables the interrupt when the 64-bit Rx_Multicast_Packets_Good_High and Rx_Multicast_Packets_Good_Low counter or 32-bit Rx_Multicast_Packets_Good_Low counter reaches the threshold value programmed in the MMC control register.
RXCRCERPIE	5	rw	**Rx CRC Error Packet Counter Interrupt Enable** Setting this bit enables the interrupt when the 64-bit Rx_CRC_Error_Packets_High and Rx_CRC_Error_Packets_Low counter or 32-bit Rx_CRC_Error_Packets_Low counter reaches the threshold value programmed in the MMC control register.
RXRUNTPIE	6	rw	**Rx Runt Error Packet Counter Interrupt Enable** Setting this bit enables the interrupt when the Rx_Runt_Error_Packets counter reaches the threshold value programmed in the MMC control register.
RXJABERPIE	7	rw	**Rx Jabber Error Packet Counter Interrupt Enable** Setting this bit enables the interrupt when the Rx_Jabber_Error_Packets counter reaches the threshold value programmed in the MMC control register.
RXUSIZEGPIE	8	rw	**Rx Good Undersize Packet Counter Interrupt Enable** Setting this bit enables the interrupt when the Rx_Undersize_Packets_Good counter reaches the threshold value programmed in the MMC control register.
RXOSIZEGPIE	9	rw	**Rx Good Oversize Packet Counter Interrupt Enable** Setting this bit enables the interrupt when the Rx_Oversize_Packets_Good counter reaches the threshold value programmed in the MMC control register.
RX64OCTGBPIE	10	rw	**Rx Good Bad 64 Octet Size Packet Counter Interrupt Enable** Setting this bit enables the interrupt when the 64-bit Rx_64Octets_Packets_Good_Bad_High and Rx_64Octets_Packets_Good_Bad_Low counter or 32-bit Rx_64Octets_Packets_Good_Bad_Low counter reaches the threshold value programmed in the MMC control register.
RX65T127OCTGBPIE	11	rw	**Rx Good Bad 65-to-127 Octet Size Packet Counter Interrupt Enable** Setting this bit enables the interrupt when the 64-bit Rx_65To127Octets_Packets_Good_Bad_High and Rx_65To127Octets_Packets_Good_Bad_Low counter or 32-bit Rx_65To127Octets_Packets_Good_Bad_Low counter reaches the threshold value programmed in the MMC control register.

(table continues...)

(continued)

Field	Bits	Type	Description
RX128T255OCTGBPIE	12	rw	**Rx Good Bad 128-to-255 Octet Size Packet Counter Interrupt Enable** Setting this bit enables the interrupt when the 64-bit Rx_128To255Octets_Packets_Good_Bad_High and Rx_128To255Octets_Packets_Good_Bad_Low counter or 32-bit Rx_128To255Octets_Packets_Good_Bad_Low counter reaches the threshold value programmed in the MMC control register.
RX256T511OCTGBPIE	13	rw	**Rx Good Bad 256-to-511 Octet Size Packet Counter Interrupt Enable** Setting this bit enables the interrupt when the 64-bit Rx_256To511Octets_Packets_Good_Bad_High and Rx_256To511Octets_Packets_Good_Bad_Low counter or 32-bit Rx_256To511Octets_Packets_Good_Bad_Low counter reaches the threshold value programmed in the MMC control register.
RX512T1023OCTGBPIE	14	rw	**Rx Good Bad 512-to-1023 Octet Size Packet Counter Interrupt Enable** Setting this bit enables the interrupt when the 64-bit Rx_512To1023Octets_Packets_Good_Bad_High and Rx_512To1023Octets_Packets_Good_Bad_Low counter or 32-bit Rx_512To1023Octets_Packets_Good_Bad_Low counter reaches the threshold value programmed in the MMC control register.
RX1024TMAXOCTGBPIE	15	rw	**Rx Good Bad 1024-to-Max Octet Size Packet Counter Interrupt Enable** Setting this bit enables the interrupt when the 64-bit Rx_1024ToMaxOctets_Packets_Good_Bad_High and Rx_1024ToMaxOctets_Packets_Good_Bad_Low counter or 32-bit Rx_1024ToMaxOctets_Packets_Good_Bad_Low counter reaches the threshold value programmed in the MMC control register.
RXUCGPIE	16	rw	**Rx Good Unicast Packet Counter Interrupt Enable** Setting this bit enables the interrupt when the 64-bit Rx_Unicast_Packets_Good_High and Rx_Unicast_Packets_Good_Low counter or 32-bit Rx_Unicast_Packets_Good_Low counter reaches the threshold value programmed in the MMC control register.
RXLENERPIE	17	rw	**Rx Length Error Packet Counter Interrupt Enable** Setting this bit enables the interrupt when the 64-bit Rx_Length_Error_Packets_High and Rx_Length_Error_Packets_Low counter or 32-bit Rx_Length_Error_Packets_Low counter reaches the threshold value programmed in the MMC control register.
RXORANGEPIE	18	rw	**Rx Out-of-Range Packet Counter Interrupt Enable** Setting this bit enables the interrupt when the 64-bit Rx_OutofRange_Packets_High and Rx_OutofRange_Packets_Low counter or 32-bit Rx_OutofRange_Packets_Low counter reaches the threshold value programmed in the MMC control register.
RXPAUSPIE	19	rw	**Rx Pause Packet Counter Interrupt Enable** Setting this bit enables the interrupt when the 64-bit Rx_Pause_Packets_High and Rx_Pause_Packets_Low counter or 32-bit Rx_Pause_Packets_Low counter reaches the threshold value programmed in the MMC control register.

(table continues...)

(continued)

Field	Bits	Type	Description
RXFOVPIE	20	rw	**Rx Queue Overflow Packet Counter Interrupt Enable** Setting this bit enables the interrupt when the 64-bit Rx_FIFOOverflow_Packets_High and Rx_FIFOOverflow_Packets_Low counter or 32-bit Rx_FIFOOverflow_Packets_Low counter reaches the threshold value programmed in the MMC control register.
RXVLANGBPIE	21	rw	**Rx Good Bad VLAN Packet Counter Interrupt Enable** Setting this bit enables the interrupt when the 64-bit Rx_VLAN_Packets_Good_Bad_High and Rx_VLAN_Packets_Good_Bad_Low counter or 32-bit Rx_VLAN_Packets_Good_Bad_Low counter reaches the threshold value programmed in the MMC control register.
RXWDOGPIE	22	rw	**Rx Watchdog Error Packet Counter Interrupt Enable** Setting this bit enables the interrupt when the Rx_Watchdog_Error_Packets counter reaches the threshold value programmed in the MMC control register.
RXDISPCGBIE	23	rw	**Rx Discard Packet Counter Interrupt Enable** Setting this bit enables the interrupt when the 64-bit Rx_Discard_Packets_Good_Bad_High and Rx_Discard_Packets_Good_Bad_Low counter or 32-bit Rx_Discard_Packets_Good_Bad_Low counter reaches the threshold value programmed in the MMC control register.
RXDISOCGBIE	24	rw	**Rx Discard Octet Counter Interrupt Enable** Setting this bit enables the interrupt when the Rx_Discard_Octets_Good_Bad_High and Rx_Discard_Octets_Good_Bad_Low counter reaches the threshold value programmed in the MMC control register.
RXLPIUSCIE	25	rw	**Rx LPI microsecond Counter Interrupt Enable** Setting this bit enables the interrupt when the Rx_LPI_USEC_Cntr reaches the threshold value programed in the MMC control register.
RXLPITRCIE	26	rw	**Rx LPI Transition Counter Interrupt Enable** Setting this bit enables the interrupt when Rx_LPI_Tran_Cntr reaches the threshold value programed in the MMC control register.
RXALEPIE	27	rw	**Rx Alignment Error Packet Counter Interrupt Enable** Setting this bit enables the interrupt when Rx_Alignment_Error_Packets counter reaches the threshold value programmed in the MMC control register.
SGPPIE	28	rw	**SGF Pass Packet Counter Interrupt Enable** Setting this bit enables the interrupt when SGF Pass Packet Counter counter reaches the threshold value programmed in the MMC control register.
SGFPIE	29	rw	**SGF Fail Packet Counter Interrupt Enable** Setting this bit enables the interrupt when SGF Fail Packet Counter counter reaches the threshold value programmed in the MMC control register.
0	30, 31	r	**Reserved** Read as 0; should be written with 0.

14.6.314 Port j MMC Transmit Interrupt Enable

The MMC Transmit Interrupt Enable register maintains the enables for the interrupts generated when transmit statistic counters reach the threshold value programmed in the MMC Control register. This register is 32 bits wide.

Portj_MMC_Transmit_Interrupt_Enable (j=0-1) Offset address: $10810_H+j*2000_H$

Port j MMC Transmit Interrupt Enable Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0				TXEXDEFPIE	TXCARERPIE	TXEXCOLPIE	TXLATCOLPIE	TXDEFPIE	TXMCOLGPIE	TXSCOLGPIE	TXLPITRCIE	TXLPIUSCIE	TXVLANGPIE	TXPAUSPIE
r	r				rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXGPKTIE	TXGOCTIE	TXUFLOWERPIE	TXBCGBPIE	TXMCGBPIE	TXUCGBPIE	TX1024TMAXOCTGBPIE	TX512T1023OCTGBPIE	TX256T511OCTGBPIE	TX128T255OCTGBPIE	TX65T127OCTGBPIE	TX64OCTGBPIE	TXMCGPIE	TXBCGPIE	TXGBPKTIE	TXGBOCTIE
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
TXGBOCTIE	0	rw	**Tx Good Bad Octet Counter Interrupt Enable** Setting this bit enables the interrupt when the Tx_Octet_Count_Good_Bad_High and Tx_Octet_Count_Good_Bad_Low counter reaches the threshold value programmed in the MMC control register.
TXGBPKTIE	1	rw	**Tx Good Bad Packet Counter Interrupt Enable** Setting this bit enables the interrupt when the 64-bit Tx_Packet_Count_Good_Bad_High and Tx_Packet_Count_Good_Bad_Low counter or 32-bit Tx_Packet_Count_Good_Bad_Low counter reaches the threshold value programmed in the MMC control register.
TXBCGPIE	2	rw	**Tx Good Broadcast Packet Counter Interrupt Enable** Setting this bit enables the interrupt when the 64-bit Tx_Broadcast_Packets_Good_High and Tx_Broadcast_Packets_Good_Low counter or 32-bit Tx_Broadcast_Packets_Good_Low counter reaches the threshold value programmed in the MMC control register.
TXMCGPIE	3	rw	**Tx Good Multicast Packet Counter Interrupt Enable** Setting this bit enables the interrupt when the 64-bit Tx_Multicast_Packets_Good_High and Tx_Multicast_Packets_Good_Low counter or 32-bit Tx_Multicast_Packets_Good_Low counter reaches the threshold value programmed in the MMC control register.

(table continues...)

(continued)

Field	Bits	Type	Description
TX64OCTGBPIE	4	rw	**Tx Good Bad 64 Octet Size Packet Counter Interrupt Enable** Setting this bit enables the interrupt when the 64-bit Tx_64Octets_Packets_Good_Bad_High and Tx_64Octets_Packets_Good_Bad_Low counter or 32-bit Tx_64Octets_Packets_Good_Bad_Low counter reaches the threshold value programmed in the MMC control register.
TX65T127OCTGBPIE	5	rw	**Tx Good Bad 65-to-127 Octet Size Packet Counter Interrupt Enable** Setting this bit enables the interrupt when the 64-bit Tx_65To127Octets_Packets_Good_Bad_High and Tx_65To127Octets_Packets_Good_Bad_Low counter or 32-bit Tx_65To127Octets_Packets_Good_Bad_Low counter reaches the threshold value programmed in the MMC control register.
TX128T255OCTGBPIE	6	rw	**Tx Good Bad 128-to-255 Octet Size Packet Counter Interrupt Enable** Setting this bit enables the interrupt when the 64-bit Tx_128To255Octets_Packets_Good_Bad_High and Tx_128To255Octets_Packets_Good_Bad_Low counter or 32-bit Tx_128To255Octets_Packets_Good_Bad_Low counter reaches the threshold value programmed in the MMC control register.
TX256T511OCTGBPIE	7	rw	**Tx Good Bad 256-to-511 Octet Size Packet Counter Interrupt Enable** Setting this bit enables the interrupt when the 64-bit Tx_256To511Octets_Packets_Good_Bad_High and Tx_256To511Octets_Packets_Good_Bad_Low counter or 32-bit Tx_256To511Octets_Packets_Good_Bad_Low counter reaches the threshold value programmed in the MMC control register.
TX512T1023OCTGBPIE	8	rw	**Tx Good Bad 512-to-1023 Octet Size Packet Counter Interrupt Enable** Setting this bit enables the interrupt when the 64-bit Tx_512To1023Octets_Packets_Good_Bad_High and Tx_512To1023Octets_Packets_Good_Bad_Low counter or 32-bit Tx_512To1023Octets_Packets_Good_Bad_Low counter reaches the threshold value programmed in the MMC control register.
TX1024TMAXOCTGBPIE	9	rw	**Tx Good Bad 1024-to-Max Octet Size Packet Counter Interrupt Enable** Setting this bit enables the interrupt when the 64-bit Tx_1024ToMaxOctets_Packets_Good_Bad_High and Tx_1024ToMaxOctets_Packets_Good_Bad_Low counter or 32-bit Tx_1024ToMaxOctets_Packets_Good_Bad_Low counter reaches the threshold value programmed in the MMC control register.
TXUCGBPIE	10	rw	**Tx Good Bad Unicast Packet Counter Interrupt Enable** Setting this bit enables the interrupt when the 64-bit Tx_Unicast_Packets_Good_Bad_High and Tx_Unicast_Packets_Good_Bad_Low counter or 32-bit Tx_Unicast_Packets_Good_Bad_Low counter reaches the threshold value programmed in the MMC control register.

(table continues...)

(continued)

Field	Bits	Type	Description
TXMCGBPIE	11	rw	**Tx Good Bad Multicast Packet Counter Interrupt Enable** Setting this bit enables the interrupt when the 64-bit Tx_Multicast_Packets_Good_Bad_High and Tx_Multicast_Packets_Good_Bad_Low counter or 32-bit Tx_Multicast_Packets_Good_Bad_Low counter reaches the threshold value programmed in the MMC control register.
TXBCGBPIE	12	rw	**Tx Good Bad Broadcast Packet Counter Interrupt Enable** Setting this bit enables the interrupt when the 64-bit Tx_Broadcast_Packets_Good_Bad_High and Tx_Broadcast_Packets_Good_Bad_Low counter or 32-bit Tx_Broadcast_Packets_Good_Bad_Low counter reaches the threshold value programmed in the MMC control register.
TXUFLOWERPIE	13	rw	**Tx Underflow Error Packet Counter Interrupt Enable** Setting this bit enables the interrupt when the 64-bit Tx_Underflow_Error_Packets_High and Tx_Underflow_Error_Packets_Low counter or 32-bit Tx_Underflow_Error_Packets_Low counter reaches the threshold value programmed in the MMC control register.
TXGOCTIE	14	rw	**Tx Good Octet Counter Interrupt Enable** Setting this bit enables the interrupt when the Tx_Octet_Count_Good_High and Tx_Octet_Count_Good_Low counter reaches the threshold value programmed in the MMC control register.
TXGPKTIE	15	rw	**Tx Good Packet Counter Interrupt Enable** Setting this bit enables the interrupt when the 64-bit Tx_Packet_Count_Good_High and Tx_Packet_Count_Good_Low counter or 32-bit Tx_Packet_Count_Good_Low counter reaches the threshold value programmed in the MMC control register.
TXPAUSPIE	16	rw	**Tx Pause Packet Counter Interrupt Enable** Setting this bit enables the interrupt when the 64-bit Tx_Pause_Packets_High and Tx_Pause_Packets_Low counter or 32-bit Tx_Pause_Packets_Low counter reaches the threshold value programmed in the MMC control register.
TXVLANGPIE	17	rw	**Tx Good VLAN Packet Counter Interrupt Enable** Setting this bit enables the interrupt when the 64-bit Tx_VLAN_Packets_Good_High and Tx_VLAN_Packets_Good_Low counter or 32-bit Tx_VLAN_Packets_Good_Low reaches the threshold value programmed in the MMC control register.
TXLPIUSCIE	18	rw	**Tx LPI microsecond Counter Interrupt Enable** Setting this bit enables the interrupt when Tx_LPI_USEC_Cntr counter reaches the threshold value programmed in the MMC counter register.
TXLPITRCIE	19	rw	**Tx LPI Transition Counter Interrupt Enable** Setting this bit enables the interrupt when Tx_LPI_Tran_Cntr counter reaches the threshold value programmed in the MMC counter register.

(table continues...)

(continued)

Field	Bits	Type	Description
TXSCOLGPIE	20	rw	**MMC Transmit Single Collision Good Packet Counter Interrupt Enable** Setting this bit masks the interrupt when the txsinglecol_g counter reaches the threshold value programmed in the MMC control register. 0_B **DISABLE**: MMC Transmit Single Collision Good Packet Counter Interrupt Enable is disabled 1_B **ENABLE**: MMC Transmit Single Collision Good Packet Counter Interrupt Enable is enabled
TXMCOLGPIE	21	rw	**MMC Transmit Multiple Collision Good Packet Counter Interrupt Enable** Setting this bit masks the interrupt when the txmulticol_g counter reaches the threshold value programmed in the MMC control register. 0_B **DISABLE**: MMC Transmit Multiple Collision Good Packet Counter Interrupt Enable is disabled 1_B **ENABLE**: MMC Transmit Multiple Collision Good Packet Counter Interrupt Enable is enabled
TXDEFPIE	22	rw	**MMC Transmit Deferred Packet Counter Interrupt Enable** Setting this bit masks the interrupt when the txdeferred counter reaches the threshold value programmed in the MMC control register. 0_B **DISABLE**: MMC Transmit Deferred Packet Counter Interrupt Enable is disabled 1_B **ENABLE**: MMC Transmit Deferred Packet Counter Interrupt Enable is enabled
TXLATCOLPIE	23	rw	**MMC Transmit Late Collision Packet Counter Interrupt Enable** Setting this bit masks the interrupt when the txlatecol counter reaches the threshold value programmed in the MMC control register. 0_B **DISABLE**: MMC Transmit Late Collision Packet Counter Interrupt Enable is disabled 1_B **ENABLE**: MMC Transmit Late Collision Packet Counter Interrupt Enable is enabled
TXEXCOLPIE	24	rw	**MMC Transmit Excessive Collision Packet Counter Interrupt Enable** Setting this bit masks the interrupt when the txexcesscol counter reaches the threshold value programmed in the MMC control register. 0_B **DISABLE**: MMC Transmit Excessive Collision Packet Counter Interrupt Enable is disabled 1_B **ENABLE**: MMC Transmit Excessive Collision Packet Counter Interrupt Enable is enabled
TXCARERPIE	25	rw	**MMC Transmit Carrier Error Packet Counter Interrupt Enable** Setting this bit masks the interrupt when the txcarriererror counter reaches the threshold value programmed in the MMC control register. 0_B **DISABLE**: MMC Transmit Carrier Error Packet Counter Interrupt Enable is disabled 1_B **ENABLE**: MMC Transmit Carrier Error Packet Counter Interrupt Enable is enabled

(table continues...)

(continued)

Field	Bits	Type	Description
TXEXDEFPIE	26	rw	**MMC Transmit Excessive Deferral Packet Counter Interrupt Enable** Setting this bit masks the interrupt when the txexcessdef counter reaches the threshold value programmed in the MMC control register. 0_B **DISABLE**: MMC Transmit Excessive Deferral Packet Counter Interrupt Enable is disabled 1_B **ENABLE**: MMC Transmit Excessive Deferral Packet Counter Interrupt Enable is enabled
0	30:27, 31	r	**Reserved** Read as 0; should be written with 0.

14.6.315 Port j Tx Octet Count Good Bad Low

This register provides lower 32 bits of transmitted octet count, excluding preamble, in good and bad packets.

Portj_Tx_Octet_Count_Good_Bad_Low (j=0-1) Offset address: $10814_H+j*2000_H$

Port j Tx Octet Count Good Bad Low Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TXOCTGBLO															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXOCTGBLO															
r															

Field	Bits	Type	Description
TXOCTGBLO	31:0	r	**Tx Good Bad Octet Count Low** This field indicates the lower 32 bits of transmitted octet count, where octet count is the number of bytes transmitted, excluding preamble, in good and bad packets.

14.6.316 Port j Tx Octet Count Good Bad High

This register provides upper 32 bits of transmitted octet count, excluding preamble, in good and bad packets.

Portj_Tx_Octet_Count_Good_Bad_High (j=0-1) Offset address: $10818_H+j*2000_H$

Port j Tx Octet Count Good Bad High Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TXOCTGBHI															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXOCTGBHI															
r															

Field	Bits	Type	Description
TXOCTGBHI	31:0	r	**Tx Good Bad Octet Count High** This field indicates upper 32 bits of transmitted octet count, where octet count is the number of bytes transmitted, excluding preamble, in good and bad packets (optional).

14.6.317 Port j Tx Packet Count Good Bad Low

This register provides lower 32 bits of transmitted good and bad packet count.

Portj_Tx_Packet_Count_Good_Bad_Low (j=0-1) Offset address: $1081C_H+j*2000_H$

Port j Tx Packet Count Good Bad Low Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TXPKTGBLO															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXPKTGBLO															
r															

Field	Bits	Type	Description
TXPKTGBLO	31:0	r	**Tx Good Bad Packet Count Low** This field indicates lower 32 bits of the counter that counts the number of good and bad packets transmitted.

14.6.318 Port j Tx Broadcast Packets Good Low

This register provides lower 32 bits of transmitted good broadcast packets count.

Portj_Tx_Broadcast_Packets_Good_Low (j=0-1) Offset address: $10824_H+j*2000_H$

Port j Tx Broadcast Packets Good Low Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TXBCASTGLO															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXBCASTGLO															
r															

Field	Bits	Type	Description
TXBCASTGLO	31:0	r	**Tx Good Broadcast Packet Count Low** This field indicates lower 32 bits of the counter that counts the number of good broadcast packets transmitted.

14.6.319 Port j Tx Multicast Packets Good Low

This register provides lower 32 bits of transmitted good multicast packets count.

Portj_Tx_Multicast_Packets_Good_Low (j=0-1) Offset address: $1082C_H+j*2000_H$

Port j Tx Multicast Packets Good Low Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TXMCASTGLO															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXMCASTGLO															
r															

Field	Bits	Type	Description
TXMCASTGLO	31:0	r	**Tx Good Multicast Packet Count Low** This field indicates lower 32 bits of the counter that counts the number of good multicast packets transmitted.

14.6.320 Port j Tx 64Octets Packets Good Bad Low

This register provides lower 32 bits of transmitted 64 octet size good and bad packets count.

Portj_Tx_64Octets_Packets_Good_Bad_Low (j=0-1) Offset address: $10834_H+j*2000_H$

Port j Tx 64Octets Packets Good Bad Low Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TX64OCTGBLO															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TX64OCTGBLO															
r															

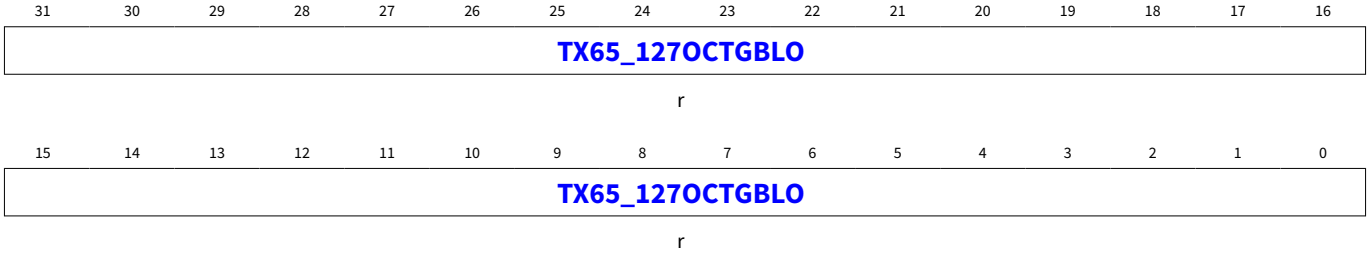
Field	Bits	Type	Description
TX64OCTGBLO	31:0	r	**Tx Good Bad 64 Octet Size Packet Count Low** This field indicates lower 32 bits of the counter that counts the number of good and bad packets transmitted that are of size 64 bytes, excluding preamble.

14.6.321 Port j Tx 65To127Octets Packets Good Bad Low

This register provides lower 32 bits of transmitted 65-to-127 octet size good and bad packets count.

Portj_Tx_65To127Octets_Packets_Good_Bad_Low (j=0-1) Offset address: $1083C_H+j*2000_H$

Port j Tx 65To127Octets Packets Good Bad Low Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
TX65_127OCTGBLO	31:0	r	**Tx Good Bad 65-to-127 Octet Size Packet Count Low** This field indicates lower 32 bits of the counter that counts the number of 65-to-127-byte good and bad packets transmitted, excluding preamble.

14.6.322 Port j Tx 128To255Octets Packets Good Bad Low

This register provides lower 32 bits of transmitted 128-to-255 octet size good and bad packets count.

Portj_Tx_128To255Octets_Packets_Good_Bad_Low (j=0-1) Offset address: $10844_H+j*2000_H$

Port j Tx 128To255Octets Packets Good Bad Low Kernel Reset value: $0000\ 0000_H$

Field	Bits	Type	Description
TX128_255OCTGBLO	31:0	r	**Tx Good Bad 128-to-255 Octet Size Packet Count Low** This field indicates lower 32 bits of the counter that counts the number of 128-to-255-byte good and bad packets transmitted, excluding preamble.

14.6.323 Port j Tx 256To511Octets Packets Good Bad Low

This register provides lower 32 bits of transmitted 256-to-511 octet size good and bad packets count.

Portj_Tx_256To511Octets_Packets_Good_Bad_Low (j=0-1) Offset address: $1084C_H+j*2000_H$

Port j Tx 256To511Octets Packets Good Bad Low Kernel Reset value: $0000\ 0000_H$

Field	Bits	Type	Description
TX256_511OCTGBLO	31:0	r	**Tx Good Bad 256-to-511 Octet Size Packet Count Low** This field indicates lower 32 bits of the counter that counts the number of 256-to-511-byte good and bad packets transmitted, excluding preamble.

14.6.324 Port j Tx 512To1023Octets Packets Good Bad Low

This register provides lower 32 bits of transmitted 512-to-1023 octet size good and bad packets count.

Portj_Tx_512To1023Octets_Packets_Good_Bad_Low (j=0-1) Offset address: $10854_H+j*2000_H$

Port j Tx 512To1023Octets Packets Good Bad Low Kernel Reset value: $0000\ 0000_H$

Field	Bits	Type	Description
TX512_1023OCTGBLO	31:0	r	**Tx Good Bad 512-to-1023 Octet Size Packet Count Low** This field indicates lower 32 bits of the counter that counts the number of 512-to-1023-byte good and bad packets transmitted, excluding preamble.

14.6.325 Port j Tx 1024ToMaxOctets Packets Good Bad Low

This register provides lower 32 bits of transmitted 1024-tomaxsize octet size good and bad packets count.

Portj_Tx_1024ToMaxOctets_Packets_Good_Bad_Low (j=0-1) Offset address: $1085C_H+j*2000_H$

Port j Tx 1024ToMaxOctets Packets Good Bad Low Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TX1024_MAXOCTGBLO															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TX1024_MAXOCTGBLO															
r															

Field	Bits	Type	Description
TX1024_MAXOCTGBLO	31:0	r	**Tx Good Bad 1024-tomaxsize Octet Size Packet Count Low** This field indicates lower 32 bits of the counter that counts the number of 1024-tomaxsize-byte (inclusive) good and bad packets transmitted, excluding preamble.

14.6.326 Port j Tx Unicast Packets Good Bad Low

This register provides lower 32 bits of transmitted good bad unicast packets count.

Portj_Tx_Unicast_Packets_Good_Bad_Low (j=0-1) Offset address: $10864_H+j*2000_H$

Port j Tx Unicast Packets Good Bad Low Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TXUCASTGBLO															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXUCASTGBLO															
r															

Field	Bits	Type	Description
TXUCASTGBLO	31:0	r	**Tx Good Bad Unicast Packet Count Low** This field indicates lower 32 bits of the counter that counts the number of good and bad unicast packets transmitted.

14.6.327 Port j Tx Multicast Packets Good Bad Low

This register provides lower 32 bits of transmitted good bad multicast packets count.

Portj_Tx_Multicast_Packets_Good_Bad_Low (j=0-1) Offset address: $1086C_H+j*2000_H$

Port j Tx Multicast Packets Good Bad Low Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TXMCASTGBLO															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXMCASTGBLO															
r															

Field	Bits	Type	Description
TXMCASTGBLO	31:0	r	**Tx Good Bad Multicast Packet Count Low** This field indicates lower 32 bits of the counter that counts the number of good and bad multicast packets transmitted.

14.6.328 Port j Tx Broadcast Packets Good Bad Low

This register provides lower 32 bits of transmitted good bad broadcast packets count.

Portj_Tx_Broadcast_Packets_Good_Bad_Low (j=0-1) Offset address: $10874_H+j*2000_H$

Port j Tx Broadcast Packets Good Bad Low Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TXBCASTGBLO															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXBCASTGBLO															
r															

Field	Bits	Type	Description
TXBCASTGBLO	31:0	r	**Tx Good Bad Broadcast Packet Count Low** This field indicates lower 32 bits of the counter that counts the number of good and bad broadcast packets transmitted.

14.6.329 Port j Tx Underflow Error Packets Low

This register provides lower 32 bits of transmitted packets aborted due to packet underflow error count.

Portj_Tx_Underflow_Error_Packets_Low (j=0-1) Offset address: $1087C_H+j*2000_H$

Port j Tx Underflow Error Packets Low Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TXUNDRFLWLO															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXUNDRFLWLO															
r															

Field	Bits	Type	Description
TXUNDRFLWLO	31:0	r	**Tx Underflow Error Packet Count Low** This field indicates lower 32 bits of the counter that counts the number of packets aborted due to packet underflow error.

14.6.330 Port j Tx Octet Count Good Low

This register provides lower 32 bits of transmitted good octets count.

Portj_Tx_Octet_Count_Good_Low (j=0-1) Offset address: $10884_H+j*2000_H$

Port j Tx Octet Count Good Low Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TXOCTGLO															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXOCTGLO															
r															

Field	Bits	Type	Description
TXOCTGLO	31:0	r	**Tx Good Octet Count Low** This field indicates lower 32 bits of the transmitted octet count, where the octet count is the number of bytes transmitted, excluding preamble, in good packets only.

14.6.331 Port j Tx Octet Count Good High

This register provides upper 32 bits of transmitted good octets count.

Portj_Tx_Octet_Count_Good_High (j=0-1) Offset address: $10888_H+j*2000_H$

Port j Tx Octet Count Good High Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TXOCTGHI															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXOCTGHI															
r															

Field	Bits	Type	Description
TXOCTGHI	31:0	r	**Tx Good Octet Count High** This field indicates upper 32 bits of the transmitted octet count, where the octet count is the number of bytes transmitted, excluding preamble, in good packets only (optional).

14.6.332 Port j Tx Packet Count Good Low

This register provides lower 32 bits of transmitted good packets count.

Portj_Tx_Packet_Count_Good_Low (j=0-1) Offset address: $1088C_H+j*2000_H$

Port j Tx Packet Count Good Low Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TXPKTGLO															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXPKTGLO															
r															

Field	Bits	Type	Description
TXPKTGLO	31:0	r	**Tx Good Packet Count Low** This field indicates lower 32 bits of the counter that counts the number of good packets transmitted.

14.6.333 Port j Tx Pause Packets Low

This register provides lower 32 bits of transmitted Pause packets count.

Portj_Tx_Pause_Packets_Low (j=0-1) Offset address: $10894_H+j*2000_H$

Port j Tx Pause Packets Low Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TXPAUSEGLO															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXPAUSEGLO															
r															

Field	Bits	Type	Description
TXPAUSEGLO	31:0	r	**Tx Pause Packet Count Low** This field indicates lower 32 bits of the counter that counts the number of good PAUSE packets transmitted.

14.6.334 Port j Tx VLAN Packets Good Low

This register provides lower 32 bits of transmitted good VLAN packets count.

Portj_Tx_VLAN_Packets_Good_Low (j=0-1) Offset address: $1089C_H+j*2000_H$

Port j Tx VLAN Packets Good Low Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TXVLANGLO															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXVLANGLO															
r															

Field	Bits	Type	Description
TXVLANGLO	31:0	r	**Tx Good VLAN Packet Count Low** This field indicates lower 32 bits of the counter that counts the number of good VLAN packets transmitted.

14.6.335 Port j Tx LPI USEC Cntr

This register provides duration in microsecond for which MAC was in Tx LPI.

Portj_Tx_LPI_USEC_Cntr (j=0-1) Offset address: $108A4_H+j*2000_H$

Port j Tx LPI USEC Cntr Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TXLPIUSC															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXLPIUSC															
r															

Field	Bits	Type	Description
TXLPIUSC	31:0	r	**Tx LPI microsecond Count** This field indicates the number of microseconds DWC_xgmac is in Tx LPI state.

14.6.336 Port j Tx LPI Tran Cntr

This register provides number of times MAC transitioned to Tx LPI state.

Portj_Tx_LPI_Tran_Cntr (j=0-1) Offset address: $108A8_H+j*2000_H$

Port j Tx LPI Tran Cntr Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TXLPITRC															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXLPITRC															
r															

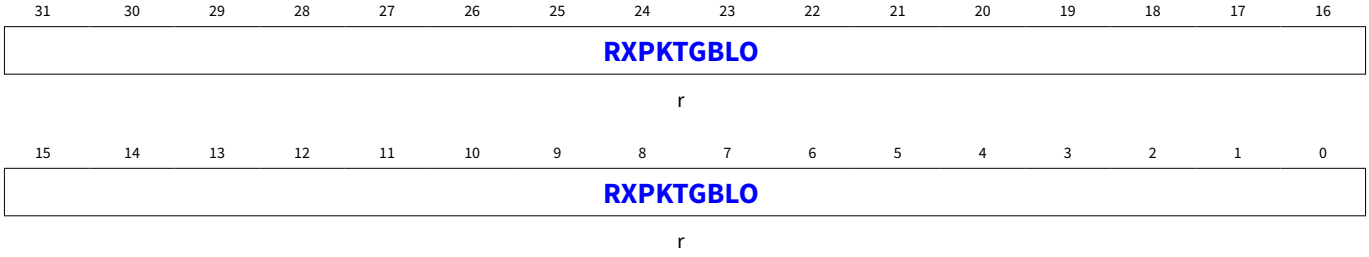
Field	Bits	Type	Description
TXLPITRC	31:0	r	**Tx LPI Transition Count** This field indicates the number of times DWC_xgmac entered to Tx LPI state.

14.6.337 Port j Rx Packet Count Good Bad Low

This register provides lower 32 bits of received good and bad packets count.

Portj_Rx_Packet_Count_Good_Bad_Low (j=0-1) Offset address: $10900_H+j*2000_H$

Port j Rx Packet Count Good Bad Low Kernel Reset value: $0000\ 0000_H$



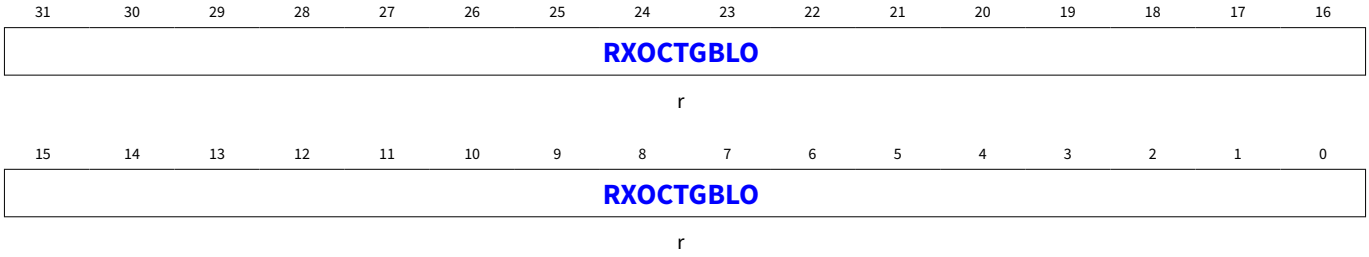
Field	Bits	Type	Description
RXPKTGBLO	31:0	r	**Rx Good Bad Packet Count Low** This field indicates lower 32 bits of the counter that counts the number of good and bad packets received.

14.6.338 Port j Rx Octet Count Good Bad Low

This register provides lower 32 bits of received octet count, excluding preamble, in good and bad packets.

Portj_Rx_Octet_Count_Good_Bad_Low (j=0-1) Offset address: $10908_H+j*2000_H$

Port j Rx Octet Count Good Bad Low Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
RXOCTGBLO	31:0	r	**Rx Good Bad Octet Count Low** This field indicates the lower 32 bits of received octet count, where octet count is the number of bytes received, excluding preamble, in good and bad packets.

14.6.339 Port j Rx Octet Count Good Bad High

This register provides upper 32 bits of received octet count, excluding preamble, in good and bad packets.

Portj_Rx_Octet_Count_Good_Bad_High (j=0-1) Offset address: $1090C_H+j*2000_H$

Port j Rx Octet Count Good Bad High Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RXOCTGBHI															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXOCTGBHI															
r															

Field	Bits	Type	Description
RXOCTGBHI	31:0	r	**Rx Good Bad Octet Count High** This field indicates upper 32 bits of received octet count, where octet count is the number of bytes received, excluding preamble, in good and bad packets (optional).

14.6.340 Port j Rx Octet Count Good Low

This register provides lower 32 bits of received good octets count.

Portj_Rx_Octet_Count_Good_Low (j=0-1) Offset address: $10910_H+j*2000_H$

Port j Rx Octet Count Good Low Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RXOCTGLO															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXOCTGLO															
r															

Field	Bits	Type	Description
RXOCTGLO	31:0	r	**Rx Good Octet Count Low** This field indicates lower 32 bits of the received octet count, where the octet count is the number of bytes received, excluding preamble, in good packets only.

14.6.341 Port j Rx Octet Count Good High

This register provides upper 32 bits of received good octets count.

Portj_Rx_Octet_Count_Good_High (j=0-1) Offset address: $10914_H+j*2000_H$

Port j Rx Octet Count Good High Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RXOCTGHI															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXOCTGHI															
r															

Field	Bits	Type	Description
RXOCTGHI	31:0	r	**Rx Good Octet Count High** This field indicates upper 32 bits of the received octet count, where the octet count is the number of bytes received, excluding preamble, in good packets only (optional).

14.6.342 Port j Rx Broadcast Packets Good Low

This register provides lower 32 bits of received good broadcast packets count.

Portj_Rx_Broadcast_Packets_Good_Low (j=0-1) Offset address: $10918_H+j*2000_H$

Port j Rx Broadcast Packets Good Low Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RXBCASTGLO															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXBCASTGLO															
r															

Field	Bits	Type	Description
RXBCASTGLO	31:0	r	**Rx Good Broadcast Packet Count Low** This field indicates lower 32 bits of the counter that counts the number of good broadcast packets received.

14.6.343 Port j Rx Multicast Packets Good Low

This register provides lower 32 bits of received good multicast packets count.

Portj_Rx_Multicast_Packets_Good_Low (j=0-1) Offset address: $10920_H+j*2000_H$

Port j Rx Multicast Packets Good Low Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RXMCASTGLO															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXMCASTGLO															
r															

Field	Bits	Type	Description
RXMCASTGLO	31:0	r	**Rx Good Multicast Packet Count Low** This field indicates lower 32 bits of the counter that counts the number of good multicast packets received.

14.6.344 Port j Rx CRC Error Packets Low

This register provides lower 32 bits of received packets with CRC error count.

Portj_Rx_CRC_Error_Packets_Low (j=0-1) Offset address: $10928_H+j*2000_H$

Port j Rx CRC Error Packets Low Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RXCRCERLO															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXCRCERLO															
r															

Field	Bits	Type	Description
RXCRCERLO	31:0	r	**Rx CRC Error Packet Count Low** This field indicates lower 32 bits of the counter that counts the number of packets received with a CRC error.

14.6.345 Port j Rx Runt Error Packets

This register provides received packets with Runt error count.

Portj_Rx_Runt_Error_Packets (j=0-1) Offset address: $10930_H+j*2000_H$

Port j Rx Runt Error Packets Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RXRUNTER															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXRUNTER															
r															

Field	Bits	Type	Description
RXRUNTER	31:0	r	**Rx Runt Error Packet Count** This field indicates number of shorter-than-64-byte packets received with a CRC error.

14.6.346 Port j Rx Jabber Error Packets

This register provides received packets with Jabber error count.

Portj_Rx_Jabber_Error_Packets (j=0-1) Offset address: $10934_H+j*2000_H$

Port j Rx Jabber Error Packets Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RXJABERER															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXJABERER															
r															

Field	Bits	Type	Description
RXJABERER	31:0	r	**Rx Jabber Error Packet Count** This field indicates number of giant (including CRC) packets received with a CRC error. The giant packet status is generated as per the programming described in MAC_Rx_Configuration.

14.6.347 Port j Rx Undersize Packets Good

This register provides count of received good undersize packets.

Portj_Rx_Undersize_Packets_Good (j=0-1) Offset address: $10938_H+j*2000_H$

Port j Rx Undersize Packets Good Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RXUSIZEG															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXUSIZEG															
r															

Field	Bits	Type	Description
RXUSIZEG	31:0	r	**Rx Good Undersize Packet Count** This field indicates number of shorter-than-64-byte packets received without any errors.

14.6.348 Port j Rx Oversize Packets Good

This register provides count of received good oversize packets.

Portj_Rx_Oversize_Packets_Good (j=0-1) Offset address: $1093C_H+j*2000_H$

Port j Rx Oversize Packets Good Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RXOSIZEG															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXOSIZEG															
r															

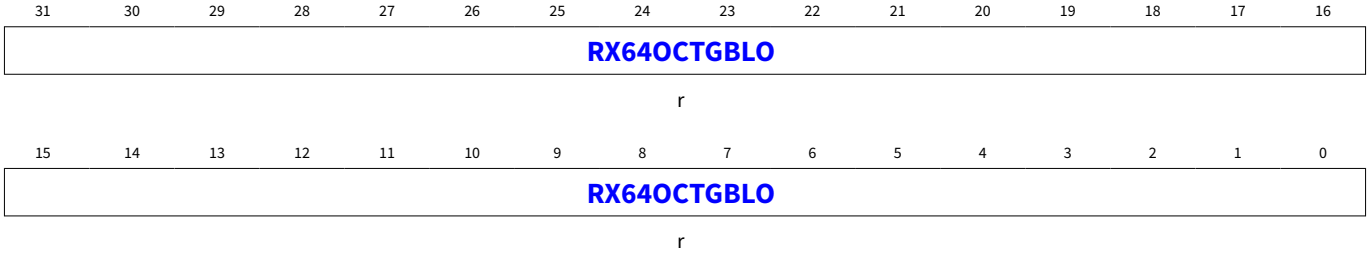
Field	Bits	Type	Description
RXOSIZEG	31:0	r	**Rx Good Oversize Packet Count** This field indicates number of greater than the maxsize packets received without errors. In other words, the number of good giant packets.

14.6.349 Port j Rx 64Octets Packets Good Bad Low

This register provides lower 32 bits of received 64 octet size good and bad packets count.

Portj_Rx_64Octets_Packets_Good_Bad_Low (j=0-1) Offset address: $10940_H+j*2000_H$

Port j Rx 64Octets Packets Good Bad Low Kernel Reset value: $0000\ 0000_H$



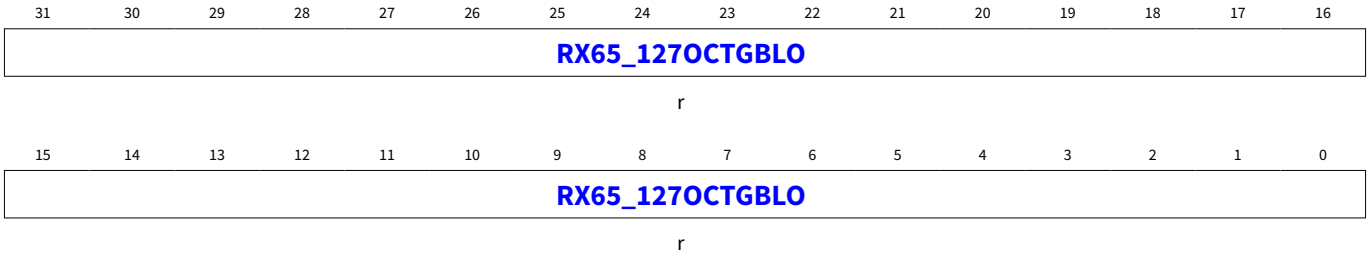
Field	Bits	Type	Description
RX64OCTGBLO	31:0	r	**Rx Good Bad 64 Octet Size Packet Count Low** This field indicates lower 32 bits of the counter that counts the number of good and bad packets received that are of size 64 bytes, excluding preamble.

14.6.350 Port j Rx 65To127Octets Packets Good Bad Low

This register provides lower 32 bits of received 65-to-127 octet size good and bad packets count.

Portj_Rx_65To127Octets_Packets_Good_Bad_Low (j=0-1) Offset address: $10948_H+j*2000_H$

Port j Rx 65To127Octets Packets Good Bad Low Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
RX65_127OCTGBLO	31:0	r	**Rx Good Bad 65-to-127 Octet Size Packet Count Low** This field indicates lower 32 bits of the counter that counts the number of 65-to-127-byte good and bad packets received, excluding preamble.

14.6.351 Port j Rx 128To255Octets Packets Good Bad Low

This register provides lower 32 bits of received 128-to-255 octet size good and bad packets count.

Portj_Rx_128To255Octets_Packets_Good_Bad_Low (j=0-1) Offset address: $10950_H+j*2000_H$

Port j Rx 128To255Octets Packets Good Bad Low Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RX128_255OCTGBLO															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RX128_255OCTGBLO															
r															

Field	Bits	Type	Description
RX128_255OCTGBLO	31:0	r	**Rx Good Bad 128-to-255 Octet Size Packet Count Low** This field indicates lower 32 bits of the counter that counts the number of 128-to-255-byte good and bad packets received, excluding preamble.

14.6.352 Port j Rx 256To511Octets Packets Good Bad Low

This register provides lower 32 bits of received 256-to-511 octet size good and bad packets count.

Portj_Rx_256To511Octets_Packets_Good_Bad_Low (j=0-1) Offset address: $10958_H+j*2000_H$

Port j Rx 256To511Octets Packets Good Bad Low Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RX256_511OCTGBLO															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RX256_511OCTGBLO															
r															

Field	Bits	Type	Description
RX256_511OCTGBLO	31:0	r	**Rx Good Bad 256-to-511 Octet Size Packet Count Low** This field indicates lower 32 bits of the counter that counts the number of 256-to-511-byte good and bad packets received, excluding preamble.

14.6.353 Port j Rx 512To1023Octets Packets Good Bad Low

This register provides lower 32 bits of received 512-to-1023 octet size good and bad packets count.

Portj_Rx_512To1023Octets_Packets_Good_Bad_Low (j=0-1) Offset address: $10960_H+j*2000_H$

Port j Rx 512To1023Octets Packets Good Bad Low Kernel Reset value: $0000\ 0000_H$

Field	Bits	Type	Description
RX512_1023OCTGBLO	31:0	r	**Rx Good Bad 512-to-1023 Octet Size Packet Count Low** This field indicates lower 32 bits of the counter that counts the number of 512-to-1023-byte good and bad packets received, excluding preamble.

14.6.354 Port j Rx 1024ToMaxOctets Packets Good Bad Low

This register provides lower 32 bits of received 1024-tomaxsize octet size good and bad packets count.

Portj_Rx_1024ToMaxOctets_Packets_Good_Bad_Low (j=0-1) Offset address: $10968_H+j*2000_H$

Port j Rx 1024ToMaxOctets Packets Good Bad Low Kernel Reset value: $0000\ 0000_H$

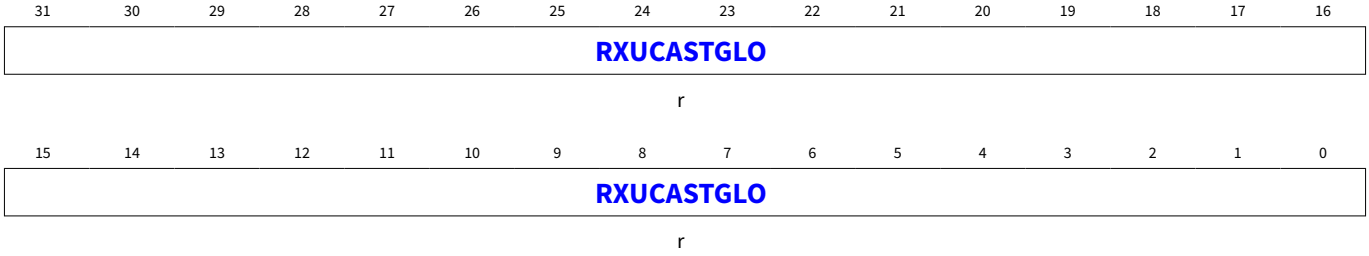
Field	Bits	Type	Description
RX1024_MAXOCTGBLO	31:0	r	**Rx Good Bad 1024-tomaxsize Octet Size Packet Count Low** This field indicates lower 32 bits of the counter that counts the number of 1024-tomaxsize-byte (inclusive) good and bad packets received, excluding preamble.

14.6.355 Port j Rx Unicast Packets Good Low

This register provides lower 32 bits of received good unicast packets count.

Portj_Rx_Unicast_Packets_Good_Low (j=0-1) Offset address: $10970_H+j*2000_H$

Port j Rx Unicast Packets Good Low Kernel Reset value: $0000\ 0000_H$



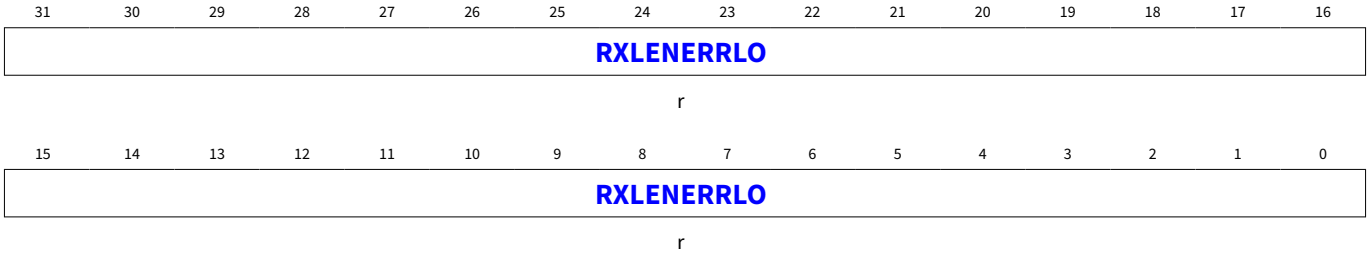
Field	Bits	Type	Description
RXUCASTGLO	31:0	r	**Rx Good Unicast Packet Count Low** This field indicates lower 32 bits of the counter that counts the number of good unicast packets received.

14.6.356 Port j Rx Length Error Packets Low

This register provides lower 32 bits of received packets with Length error count.

Portj_Rx_Length_Error_Packets_Low (j=0-1) Offset address: $10978_H+j*2000_H$

Port j Rx Length Error Packets Low Kernel Reset value: $0000\ 0000_H$



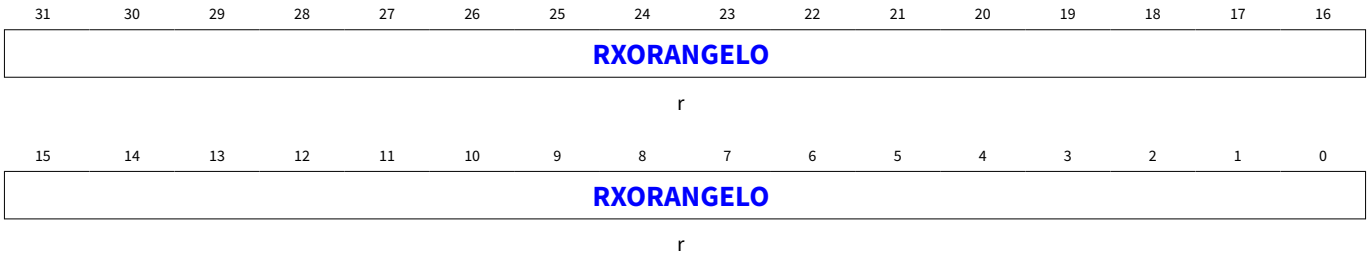
Field	Bits	Type	Description
RXLENERRLO	31:0	r	**Rx Length Error Packet Count Low** This field indicates lower 32 bits of the counter that counts the number of packets received with a Length error (Length field not equal to packet size), for all packets with valid Length field.

14.6.357 Port j Rx OutofRange Packets Low

This register provides lower 32 bits of received packets with out of range length field count.

Portj_Rx_OutofRange_Packets_Low (j=0-1) Offset address: $10980_H+j*2000_H$

Port j Rx OutofRange Packets Low Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
RXORANGELO	31:0	r	**Rx Out of Range Packet Count Low** This field indicates lower 32 bits of the counter that counts the number of packets received with Length field not equal to the valid packet size (greater than 1500).

14.6.358 Port j Rx Pause Packets Low

This register provides lower 32 bits of received Pause packets count.

Portj_Rx_Pause_Packets_Low (j=0-1) Offset address: $10988_H+j*2000_H$

Port j Rx Pause Packets Low Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RXPAUSELO															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXPAUSELO															
r															

Field	Bits	Type	Description
RXPAUSELO	31:0	r	**Rx Pause Packet Count Low** This field indicates lower 32 bits of the counter that counts the number of good and valid PAUSE packets received.

14.6.359 Port j Rx FIFOOverflow Packets Low

This register provides lower 32 bits of received packets count that had Rx Queue overflow.

Portj_Rx_FIFOOverflow_Packets_Low (j=0-1) Offset address: $10990_H+j*2000_H$

Port j Rx FIFOOverflow Packets Low Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RXFOVFLO															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXFOVFLO															
r															

Field	Bits	Type	Description
RXFOVFLO	31:0	r	**Rx FIFO Overflow Packet Count Low** This field indicates lower 32 bits of the counter that counts the number of receive packets dropped due to FIFO overflow.

14.6.360 Port j Rx VLAN Packets Good Bad Low

This register provides lower 32 bits of received good bad VLAN packets count.

Portj_Rx_VLAN_Packets_Good_Bad_Low (j=0-1) Offset address: $10998_H+j*2000_H$

Port j Rx VLAN Packets Good Bad Low Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RXVLANGBLO															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXVLANGBLO															
r															

Field	Bits	Type	Description
RXVLANGBLO	31:0	r	**Rx Good Bad VLAN Packet Count Low** This field indicates lower 32 bits of the counter that counts the number of good and bad VLAN packets received.

14.6.361 Port j Rx Watchdog Error Packets

This register has a count of packets that are received with watchdog error.

Portj_Rx_Watchdog_Error_Packets (j=0-1) Offset address: $109A0_H+j*2000_H$

Port j Rx Watchdog Error Packets Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RXWDOGERR															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXWDOGERR															
r															

Field	Bits	Type	Description
RXWDOGERR	31:0	r	**Rx Watchdog Error Packet Count** This field indicates number of packets received with a watchdog time-out error (packets of length greater than the watchdog time-out limit as described in MAC_WD_JB_Timeout).

14.6.362 Port j Rx LPI USEC Cntr

This register provides duration in microsecond for which MAC was in Rx LPI.

Portj_Rx_LPI_USEC_Cntr (j=0-1) Offset address: $109A4_H+j*2000_H$

Port j Rx LPI USEC Cntr Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RXLPIUSC															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXLPIUSC															
r															

Field	Bits	Type	Description
RXLPIUSC	31:0	r	**Rx LPI microsecond Count** This field indicates the number of microseconds DWC_xgmac is in Rx LPI state.

14.6.363 Port j Rx LPI Tran Cntr

This register provides number of times MAC transitioned to Rx LPI state.

Portj_Rx_LPI_Tran_Cntr (j=0-1) Offset address: $109A8_H+j*2000_H$

Port j Rx LPI Tran Cntr Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RXLPITRC															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXLPITRC															
r															

Field	Bits	Type	Description
RXLPITRC	31:0	r	**Rx LPI Transition Count** This field indicates the number of times DWC_xgmac entered the Rx LPI state.

14.6.364 Port j Rx Discard Packets Good Bad Low

This register provides lower 32 bits of received good bad packets count that were discarded due to filter fail.

Portj_Rx_Discard_Packets_Good_Bad_Low (j=0-1) Offset address: $109AC_H+j*2000_H$

Port j Rx Discard Packets Good Bad Low Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RXDPCNTGBLO															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXDPCNTGBLO															
r															

Field	Bits	Type	Description
RXDPCNTGBLO	31:0	r	**Rx Good Bad Discard Packet Count Low** This field indicates the number of packets received and discarded due to Filter Fail. This represents the lower 32 bits of the counter. When a packet is dropped due to filtering and not counted in the normal RMON counters, this counter is incremented. This counter is incremented irrespective of whether it is a good or bad packet.

14.6.365 Port j Rx Discard Octets Good Bad Low

This register provides lower 32 bits of received good bad octets count that were discarded due to filter fail.

Portj_Rx_Discard_Octets_Good_Bad_Low (j=0-1) Offset address: $109B4_H+j*2000_H$

Port j Rx Discard Octets Good Bad Low Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RXDOCNTGBLO															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXDOCNTGBLO															
r															

Field	Bits	Type	Description
RXDOCNTGBLO	31:0	r	**Rx Good Bad Discard Octet Count Low** This field indicates the number of octets received and discarded due to Filter Fail. This represents the lower 32 bits of the counter. When a packet is dropped due to filtering and not counted in the normal RMON counters, this octet counter is incremented by packet size. This counter is incremented irrespective of whether it is a good or bad packet.

14.6.366 Port j Rx Discard Octets Good Bad High

This register provides upper 32 bits of received good bad octets count that were discarded due to filter fail.

Portj_Rx_Discard_Octets_Good_Bad_High (j=0-1) Offset address: $109B8_H+j*2000_H$

Port j Rx Discard Octets Good Bad High Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RXDOCNTGBHI															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXDOCNTGBHI															
r															

Field	Bits	Type	Description
RXDOCNTGBHI	31:0	r	**Rx Good Bad Discard Octet Count High** This field indicates the number of octets received and discarded due to Filter Fail. This represents the upper 32 bits of the counter. Any packet dropped due to filtering and not counted in the normal RMON counters then this octet counter is incremented by packet size. This counter is incremented irrespective of whether it is a good or bad packet.

14.6.367 Port j Rx Alignment Error Packets

This register provides the number of packets received by DWC_xgmac with alignment (dribble) error. It is valid only in 10/100 mode.

Portj_Rx_Alignment_Error_Packets (j=0-1) Offset address: $109BC_H+j*2000_H$

Port j Rx Alignment Error Packets Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RXALIGNERR															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXALIGNERR															
r															

Field	Bits	Type	Description
RXALIGNERR	31:0	r	**Rx Alignment Error Packet Count** This field indicates the number of packets received with alignment (dribble) error. It is valid only in 10/100 mode.

14.6.368 Port j SGF Pass PktCnt

This register provides the number of Stream-Gate Filter (SGF) Passed Packets.

Portj_SGF_Pass_PktCnt (j=0-1) Offset address: $109F0_H+j*2000_H$

Port j SGF Pass PktCnt Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SGPP															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SGPP															
r															

Field	Bits	Type	Description
SGPP	31:0	r	**Stream-Gate Filter Pass Packets Count** This field indicates the number of SGF Passed Packets.

14.6.369 Port j SGF Fail PktCnt

This register provides the number of SGF Failed Packets.

Portj_SGF_Fail_PktCnt (j=0-1) Offset address: $109F4_H+j*2000_H$

Port j SGF Fail PktCnt Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SGFP															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SGFP															
r															

Field	Bits	Type	Description
SGFP	31:0	r	**Stream-Gate Filter Failed Packets Count** This field indicates the number of SGF Failed Packets.

14.6.370 Port j MMC FPE Tx Interrupt

This register maintains the interrupts generated from all FPE related Transmit statistics counters. The MMC FPE Transmit Interrupt register maintains the interrupts generated when the transmit statistic counters reach the threshold value programmed in MMC control register (for example, the counter's MSB is set). This register is 32 bits wide. An interrupt bit is cleared when the respective MMC counter that caused the interrupt is read. The least significant byte lane (bits[7:0]) of the corresponding counter must be read to clear the interrupt status bit.

Portj_MMC_FPE_Tx_Interrupt (j=0-1) Offset address: $10A00_H+j*2000_H$

Port j MMC FPE Tx Interrupt Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0													GOCIS	HRCIS	FCIS
r													r	r	r

Field	Bits	Type	Description
FCIS	0	r	**MMC Tx FPE Fragment Counter Interrupt status** This bit is set when the Tx_FPE_Fragment_Cntr counter reaches the threshold value programmed in MMC control register. Access restriction applies. Clears on read. Self-set to 1 on internal event. Exists when any one of the Rx/Tx MMC counters are enabled during FPE Enabled configuration. 0_B **INACTIVE**: MMC Tx FPE Fragment Counter Interrupt status not detected 1_B **ACTIVE**: MMC Tx FPE Fragment Counter Interrupt status detected
HRCIS	1	r	**MMC Tx Hold Request Counter Interrupt Status** This bit is set when the Tx_Hold_Req_Cntr counter reaches the threshold value programmed in MMC control register. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0_B **INACTIVE**: MMC Tx Hold Request Counter Interrupt Status not detected 1_B **ACTIVE**: MMC Tx Hold Request Counter Interrupt Status detected
GOCIS	2	r	**MMC Tx Gate/Window Overrun Counter Interrupt Status** This bit is set when the Tx_Gate_Orun_Cntr_Low/High counter reaches the threshold value programmed in MMC control register. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0_B **INACTIVE**: MMC Tx Gate/Window Overrun Counter Interrupt Status not detected 1_B **ACTIVE**: MMC Tx Gate/Window Overrun Counter Interrupt Status detected
0	31:3	r	**Reserved** Read as 0; should be written with 0.

14.6.371 Port j MMC FPE Tx Interrupt Mask

This register maintains the masks for interrupts generated from all FPE related Transmit statistics counters. The MMC Receive Interrupt Mask register maintains the masks for the interrupts generated when FPE related receive statistic counters reach the threshold value programmed in MMC control register. This register is 32 bit wide.

Portj_MMC_FPE_Tx_Interrupt_Mask (j=0-1) Offset address: $10A04_H+j*2000_H$

Port j MMC FPE Tx Interrupt Mask Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0													GOCIM	HRCIM	FCIM
r													rw	rw	rw

Field	Bits	Type	Description
FCIM	0	rw	**MMC Transmit Fragment Counter Interrupt Mask** Setting this bit masks the interrupt when the Tx_FPE_Fragment_Cntr counter reaches the threshold value programmed in MMC control register. Exists when any one of the Rx/Tx MMC counters are enabled during FPE Enabled configuration. 0_B **DISABLE**: MMC Transmit Fragment Counter Interrupt Mask is disabled 1_B **ENABLE**: MMC Transmit Fragment Counter Interrupt Mask is enabled
HRCIM	1	rw	**MMC Transmit Hold Request Counter Interrupt Mask** Setting this bit masks the interrupt when the Tx_Hold_Req_Cntr counter the threshold value programmed in MMC control register. 0_B **DISABLE**: MMC Transmit Hold Request Counter Interrupt Mask is disabled 1_B **ENABLE**: MMC Transmit Hold Request Counter Interrupt Mask is enabled
GOCIM	2	rw	**MMC Transmit Gate/Window Overrun Counter Interrupt Mask** Setting this bit masks the interrupt when the Tx_Gate_Orun_Cntr_Low/High counter the threshold value programmed in MMC control register. 0_B **DISABLE**: MMC Transmit Gate/Window Overrun Counter Interrupt Mask is disabled 1_B **ENABLE**: MMC Transmit Gate/Window Overrun Counter Interrupt Mask is enabled
0	31:3	r	**Reserved** Read as 0; should be written with 0.

14.6.372 Port j MMC Tx FPE Fragment Cntr

This register provides the number of additional mPackets transmitted due to preemption.

Portj_MMC_Tx_FPE_Fragment_Cntr (j=0-1) Offset address: $10A08_H+j*2000_H$

Port j MMC Tx FPE Fragment Cntr Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TXFFC															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXFFC															
r															

Field	Bits	Type	Description
TXFFC	31:0	r	**Tx FPE Fragment counter** This field indicates the number of additional mPackets transmitted due to preemption Exists when any one of the Rx/Tx MMC counters are enabled during FPE Enabled configuration.

14.6.373 Port j MMC Tx Hold Req Cntr

This register provides the count of number of times a hold request is given to MAC.

Portj_MMC_Tx_Hold_Req_Cntr (j=0-1) Offset address: $10A0C_H+j*2000_H$

Port j MMC Tx Hold Req Cntr Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TXHRC															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXHRC															
r															

Field	Bits	Type	Description
TXHRC	31:0	r	**Tx Hold Request Counter** This field indicates count of number of a hold request is given to MAC.

14.6.374 Port j MMC Tx Gate Orun Cntr Low

This register provides the lower 32-bits of counter that counts the number of times packets overrun the transmission gate/window.

Portj_MMC_Tx_Gate_Orun_Cntr_Low (j=0-1) Offset address: $10A10_H+j*2000_H$

Port j MMC Tx Gate Orun Cntr Low Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TXGOCL															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXGOCL															
r															

Field	Bits	Type	Description
TXGOCL	31:0	r	**Tx Gate/Window Overrun Counter Low** This field indicates the lower 32-bits of counter that counts the number of of times packets overrun the transmission gate/window. In FPE mode, the counter is updated based on pre-empt packet overrunning in Hold window. In EST mode, the counter is updated based on any packet of the queue overrunning after gate closed for the Tx Queue. When both enabled together, the counter is updated based on both the conditions.

14.6.375 Port j MMC Tx Gate Orun Cntr High

This register provides the upper 32-bits of counter that counts the number of times packets overrun the transmission gate/window.

Portj_MMC_Tx_Gate_Orun_Cntr_High (j=0-1) Offset address: $10A14_H+j*2000_H$

Port j MMC Tx Gate Orun Cntr High Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TXGOCH															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXGOCH															
r															

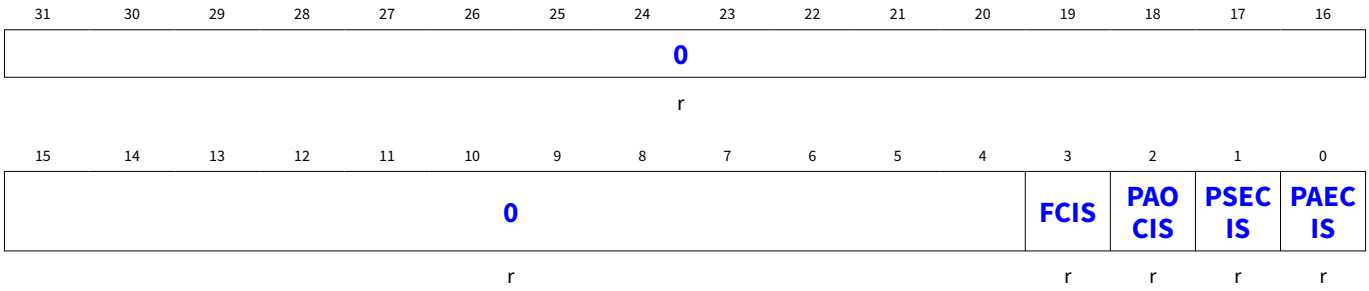
Field	Bits	Type	Description
TXGOCH	31:0	r	**Tx Gate/Window Overrun Counter High** This field indicates the upper 32-bits of counter that counts the number of of times packets overrun the transmission gate/window. In FPE mode, the counter is updated based on pre-empt packet overrunning in Hold window. In EST mode, the counter is updated based on any packet of the queue overrunning after gate closed for the Tx Queue. When both enabled together, the counter is updated based on both the conditions.

14.6.376 Port j MMC FPE Rx Interrupt

This register maintains the interrupts generated from all FPE related Receive statistics counters. The MMC FPE Receive Interrupt register maintains the interrupts generated when receive statistic counters reach the threshold value programmed in MMC control register (for example, the counter's MSB is set). The MMC FPE Receive Interrupt register is a 32 bit register. An interrupt bit is cleared when the respective MMC counter that caused the interrupt is read. The least significant byte lane (Bits[7:0]) of the respective counter must be read to clear the interrupt bit.

Portj_MMC_FPE_Rx_Interrupt (j=0-1) Offset address: $10A20_H+j*2000_H$

Port j MMC FPE Rx Interrupt Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
PAECIS	0	r	**MMC Rx Packet Assembly Error Counter Interrupt Status** This bit is set when the Rx_Packet_Assemble_Err_Cntr counter reaches the threshold value programmed in MMC control register. Access restriction applies. Clears on read. Self-set to 1 on internal event. Exists when any one of the Rx/Tx MMC counters are enabled during FPE Enabled configuration. 0_B **INACTIVE**: MMC Rx Packet Assembly Error Counter Interrupt Status not detected 1_B **ACTIVE**: MMC Rx Packet Assembly Error Counter Interrupt Status detected
PSECIS	1	r	**MMC Rx Packet SMD Error Counter Interrupt Status** This bit is set when the Rx_Packet_SMD_Err_Cntr counter reaches the threshold value programmed in MMC control register. Access restriction applies. Clears on read. Self-set to 1 on internal event. Exists when any one of the Rx/Tx MMC counters are enabled during FPE Enabled configuration. 0_B **INACTIVE**: MMC Rx Packet SMD Error Counter Interrupt Status not detected 1_B **ACTIVE**: MMC Rx Packet SMD Error Counter Interrupt Status detected
PAOCIS	2	r	**MMC Rx Packet Assembly OK Counter Interrupt Status** This bit is set when the Rx_Packet_Assemble_Ok_Cntr counter reach the threshold value programmed in MMC control register. Access restriction applies. Clears on read. Self-set to 1 on internal event. Exists when any one of the Rx/Tx MMC counters are enabled during FPE Enabled configuration. 0_B **INACTIVE**: MMC Rx Packet Assembly OK Counter Interrupt Status not detected 1_B **ACTIVE**: MMC Rx Packet Assembly OK Counter Interrupt Status detected

(table continues...)

(continued)

Field	Bits	Type	Description
FCIS	3	r	**MMC Rx FPE Fragment Counter Interrupt Status** This bit is set when the Rx_FPE_Fragment_Cntr counter reach the threshold value programmed in MMC control register. Access restriction applies. Clears on read. Self-set to 1 on internal event. Exists when any one of the Rx/Tx MMC counters are enabled during FPE Enabled configuration. 0_B **INACTIVE**: MMC Rx FPE Fragment Counter Interrupt Status not detected 1_B **ACTIVE**: MMC Rx FPE Fragment Counter Interrupt Status detected
0	31:4	r	**Reserved** Read as 0; should be written with 0.

14.6.377 Port j MMC FPE Rx Interrupt Mask

This register maintains the masks for interrupts generated from all FPE related Receive statistics counters. The MMC Receive Interrupt Mask register maintains the masks for the interrupts generated when FPE related receive statistic counters reach the threshold value programmed in MMC control register. This register is 32 bit wide.

Portj_MMC_FPE_Rx_Interrupt_Mask (j=0-1) Offset address: $10A24_H+j*2000_H$

Port j MMC FPE Rx Interrupt Mask Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												FCIM	PAOCIM	PSECIM	PAECIM
r												rw	rw	rw	rw

Field	Bits	Type	Description
PAECIM	0	rw	**MMC Rx Packet Assembly Error Counter Interrupt Mask** Setting this bit masks the interrupt when the Rx_Packet_Assemble_Err_Cntr counter reaches the threshold value programmed in MMC control register. Exists when any one of the Rx/Tx MMC counters are enabled during FPE Enabled configuration. 0_B **DISABLE**: MMC Rx Packet Assembly Error Counter Interrupt Mask is disabled 1_B **ENABLE**: MMC Rx Packet Assembly Error Counter Interrupt Mask is enabled

(table continues...)

(continued)

Field	Bits	Type	Description
PSECIM	1	rw	**MMC Rx Packet SMD Error Counter Interrupt Mask** Setting this bit masks the interrupt when the Rx_Packet_SMD_Err_Cntr counter reaches the threshold value programmed in MMC control register. Exists when any one of the Rx/Tx MMC counters are enabled during FPE Enabled configuration. 0_B **DISABLE**: MMC Rx Packet SMD Error Counter Interrupt Mask is disabled 1_B **ENABLE**: MMC Rx Packet SMD Error Counter Interrupt Mask is enabled
PAOCIM	2	rw	**MMC Rx Packet Assembly OK Counter Interrupt Mask** Setting this bit masks the interrupt when the Rx_Packet_Assemble_Ok_Cntr counter reaches the threshold value programmed in MMC control register. Exists when any one of the Rx/Tx MMC counters are enabled during FPE Enabled configuration. 0_B **DISABLE**: MMC Rx Packet Assembly OK Counter Interrupt Mask is disabled 1_B **ENABLE**: MMC Rx Packet Assembly OK Counter Interrupt Mask is enabled
FCIM	3	rw	**MMC Rx FPE Fragment Counter Interrupt Mask** Setting this bit masks the interrupt when the Tx_FPE_Fragment_Cntr counter reaches the threshold value programmed in MMC control register. Exists when any one of the Rx/Tx MMC counters are enabled during FPE Enabled configuration. 0_B **DISABLE**: MMC Rx FPE Fragment Counter Interrupt Mask is disabled 1_B **ENABLE**: MMC Rx FPE Fragment Counter Interrupt Mask is enabled
0	31:4	r	**Reserved** Read as 0; should be written with 0.

14.6.378 Port j MMC Rx Packet Assembly Err Cntr

This register provides the number of MAC frames with reassembly errors on the Receiver, due to mismatch in the Fragment Count value.

Portj_MMC_Rx_Packet_Assembly_Err_Cntr (j=0-1) Offset address: $10A28_H+j*2000_H$

Port j MMC Rx Packet Assembly Err Cntr Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PAEC															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PAEC															
r															

Field	Bits	Type	Description
PAEC	31:0	r	**Rx Packet Assembly Error Counter** This field indicates the number of MAC frames with reassembly errors on the Receiver, due to mismatch in the Fragment Count value. Exists when any one of the Rx/Tx MMC counters are enabled during FPE Enabled configuration.

14.6.379 Port j MMC Rx Packet SMD Err Cntr

This register provides the number of received MAC frames rejected due to unknown SMD value and MAC frame fragments rejected due to arriving with an SMD-C when there was no preceding preempted frame.

Portj_MMC_Rx_Packet_SMD_Err_Cntr (j=0-1) Offset address: $10A2C_H+j*2000_H$

Port j MMC Rx Packet SMD Err Cntr Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PSEC															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PSEC															
r															

Field	Bits	Type	Description
PSEC	31:0	r	**Rx Packet SMD Error Counter** This field indicates the number of MAC frames rejected due to unknown SMD value and MAC frame fragments rejected due to arriving with an SMD-C when there was no preceding preempted frame. Exists when at least one of the Rx/Tx MMC counters are enabled during FPE Enabled configuration.

14.6.380 Port j MMC Rx Packet Assembly OK Cntr

This register provides the number of MAC frames that were successfully reassembled and delivered to MAC.

Portj_MMC_Rx_Packet_Assembly_OK_Cntr (j=0-1) Offset address: $10A30_H+j*2000_H$

Port j MMC Rx Packet Assembly OK Cntr Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PAOC															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PAOC															
r															

Field	Bits	Type	Description
PAOC	31:0	r	**Rx Packet Assembly OK Counter** This field indicates the number of MAC frames that were successfully reassembled and delivered to MAC. Exists when at least one of the Rx/Tx MMC counters are enabled during FPE Enabled configuration.

14.6.381 Port j MMC Rx FPE Fragment Cntr

This register provides the number of additional mPackets received due to preemption.

Portj_MMC_Rx_FPE_Fragment_Cntr (j=0-1) Offset address: $10A34_H+j*2000_H$

Port j MMC Rx FPE Fragment Cntr Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FFC															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FFC															
r															

Field	Bits	Type	Description
FFC	31:0	r	**Rx FPE Fragment Counter** This field indicates the number of additional mPackets received due to preemption Exists when at least one of the Rx/Tx MMC counters are enabled during FPE Enabled configuration.

14.6.382 Port j Tx Single Collision Good Packets

This register provides the number of successfully transmitted packets by DWC_xgmac after a single collision in the half-duplex mode.

Portj_Tx_Single_Collision_Good_Packets (j=0-1) Offset address: $10A40_H+j*2000_H$

Port j Tx Single Collision Good Packets Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TXSNGLCOLG															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXSNGLCOLG															
r															

Field	Bits	Type	Description
TXSNGLCOLG	31:0	r	**Tx Single Collision Good Packets** This field indicates the number of successfully transmitted packets after a single collision in the half-duplex mode.

14.6.383 Port j Tx Multiple Collision Good Packets

This register provides the number of successfully transmitted packets by DWC_xgmac after multiple collisions in the half-duplex mode.

Portj_Tx_Multiple_Collision_Good_Packets (j=0-1) Offset address: $10A44_H+j*2000_H$

Port j Tx Multiple Collision Good Packets Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TXMULTCOLG															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXMULTCOLG															
r															

Field	Bits	Type	Description
TXMULTCOLG	31:0	r	**Tx Multiple Collision Good Packets** This field indicates the number of successfully transmitted packets after multiple collisions in the half-duplex mode.

14.6.384 Port j Tx Deferred Packets

This register provides the number of successfully transmitted by DWC_xgmac after a deferral in the half-duplex mode.

Portj_Tx_Deferred_Packets (j=0-1) Offset address: $10A48_H+j*2000_H$

Port j Tx Deferred Packets Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TXDEFRD															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXDEFRD															
r															

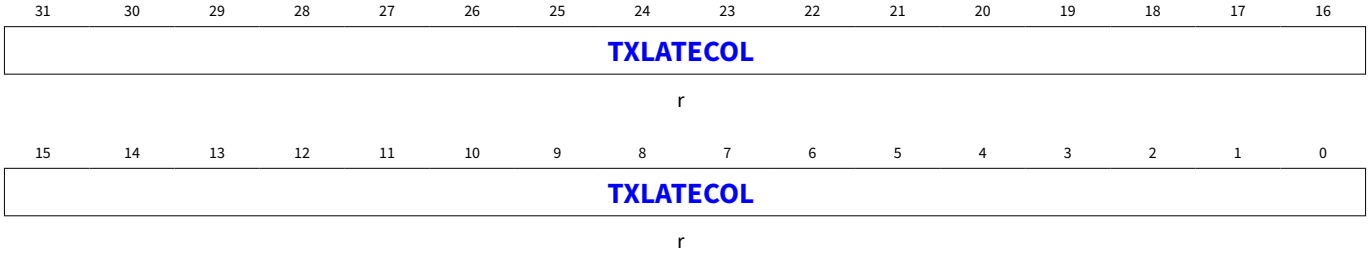
Field	Bits	Type	Description
TXDEFRD	31:0	r	**Tx Deferred Packets** This field indicates the number of successfully transmitted packets after a deferral, in the half-duplex mode.

14.6.385 Port j Tx Late Collision Packets

This register provides the number of packets aborted by DWC_xgmac because of late collision error.

Portj_Tx_Late_Collision_Packets (j=0-1) Offset address: $10A4C_H+j*2000_H$

Port j Tx Late Collision Packets Kernel Reset value: $0000\ 0000_H$



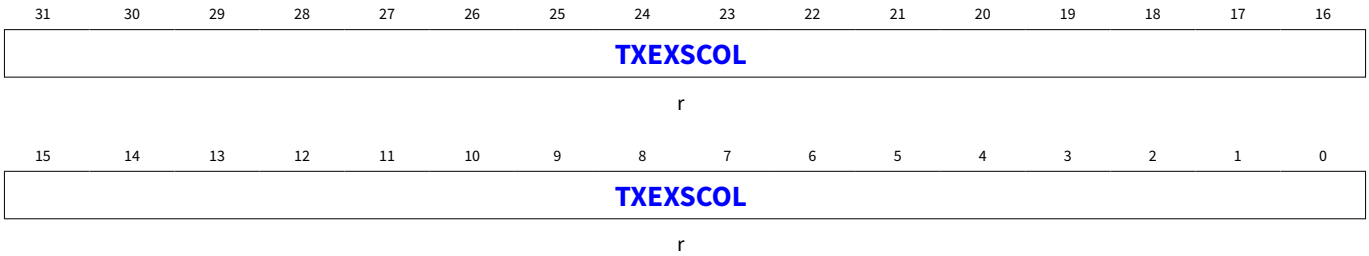
Field	Bits	Type	Description
TXLATECOL	31:0	r	**Tx Late Collision Packets** This field indicates the number of packets aborted because of late collision error.

14.6.386 Port j Tx Excessive Collision Packets

This register provides the number of packets aborted by DWC_xgmac because of excessive (16) collision errors.

Portj_Tx_Excessive_Collision_Packets (j=0-1) Offset address: $10A50_H+j*2000_H$

Port j Tx Excessive Collision Packets Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
TXEXSCOL	31:0	r	**Tx Excessive Collision Packets** This field indicates the number of packets aborted because of excessive (16) collision errors.

14.6.387 Port j Tx Carrier Error Packets

This register provides the number of packets aborted by DWC_xgmac because of carrier sense error (no carrier or loss of carrier).

Portj_Tx_Carrier_Error_Packets (j=0-1) Offset address: $10A54_H+j*2000_H$

Port j Tx Carrier Error Packets Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TXCARR															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXCARR															
r															

Field	Bits	Type	Description
TXCARR	31:0	r	**Tx Carrier Error Packets** This field indicates the number of packets aborted because of carrier sense error (no carrier or loss of carrier).

14.6.388 Port j Tx Excessive Deferral Error

This register provides the number of packets aborted by DWC_xgmac because of excessive deferral error (deferred for more than two max-size packet times).

Portj_Tx_Excessive_Deferral_Error (j=0-1) Offset address: $10A58_H+j*2000_H$

Port j Tx Excessive Deferral Error Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TXEXSDEF															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXEXSDEF															
r															

Field	Bits	Type	Description
TXEXSDEF	31:0	r	**Tx Excessive Deferral Error** This field indicates the number of packets aborted because of excessive deferral error (deferred for more than two max-sized packet times).

14.6.389 Port j MMC IPC Rx Interrupt Mask

This register maintains the mask for the interrupt generated from the receive IPC statistic counters. The MMC Receive Checksum Off load Interrupt Mask register maintains the masks for the interrupts generated when the receive IPC (Checksum Off load) statistic counters reach half their maximum value, and when they reach their maximum values. This register is 32 bits wide.

Portj_MMC_IPC_Rx_Interrupt_Mask (j=0-1) Offset address: $10A5C_H+j*2000_H$

Port j MMC IPC Rx Interrupt Mask Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		RXICMPEROIM	RXICMPGOIM	RXTCPEROIM	RXTCPGOIM	RXUDPEROIM	RXUDPGOIM	RXIPV6NOPAYOIM	RXIPV6HEROIM	RXIPV6GOIM	RXIPV4UDSBLOIM	RXIPV4FRAGOIM	RXIPV4NOPAYOIM	RXIPV4HEROIM	RXIPV4GOIM
r		rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	RXICMPERPIM	RXICMPGPIM	RXTCPERPIM	RXTCPGPIM	RXUDPERPIM	RXUDPGPIM	RXIPV6NOPAYPIM	RXIPV6HERPIM	RXIPV6GPIM	RXIPV4UDSBLPIM	RXIPV4FRAGPIM	RXIPV4NOPAYPIM	RXIPV4HERPIM	RXIPV4GPIM
r	r	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RXIPV4GPIM	0	rw	**MMC Receive IPv4 Good Packet Counter Interrupt Mask** Setting this bit masks the interrupt when the RxIPv4_Good_Packets counter reaches half of the maximum value or the maximum value. 0_B **DISABLE**: MMC Receive IPV4 Good Packet Counter Interrupt Mask is disabled 1_B **ENABLE**: MMC Receive IPV4 Good Packet Counter Interrupt Mask is enabled
RXIPV4HERPIM	1	rw	**MMC Receive IPv4 Header Error Packet Counter Interrupt Mask** Setting this bit masks the interrupt when the RxIPv4_Header_Error_Packets counter reaches half of the maximum value or the maximum value. 0_B **DISABLE**: MMC Receive IPV4 Header Error Packet Counter Interrupt Mask is disabled 1_B **ENABLE**: MMC Receive IPV4 Header Error Packet Counter Interrupt Mask is enabled
RXIPV4NOPAYPIM	2	rw	**MMC Receive IPv4 No Payload Packet Counter Interrupt Mask** Setting this bit masks the interrupt when the RxIPv4_No_Payload_Packets counter reaches half of the maximum value or the maximum value. 0_B **DISABLE**: MMC Receive IPV4 No Payload Packet Counter Interrupt Mask is disabled 1_B **ENABLE**: MMC Receive IPV4 No Payload Packet Counter Interrupt Mask is enabled
RXIPV4FRAGPIM	3	rw	**MMC Receive IPv4 Fragmented Packet Counter Interrupt Mask** Setting this bit masks the interrupt when the RxIPv4_Fragmented_Packets counter reaches half of the maximum value or the maximum value. 0_B **DISABLE**: MMC Receive IPV4 Fragmented Packet Counter Interrupt Mask is disabled 1_B **ENABLE**: MMC Receive IPV4 Fragmented Packet Counter Interrupt Mask is enabled

(table continues...)

(continued)

Field	Bits	Type	Description
RXIPV4UDSBLPIM	4	rw	**MMC Receive IPv4 UDP Checksum Disabled Packet Counter Interrupt Mask** Setting this bit masks the interrupt when the RxIPv4_UDP_Checksum_Disabled_Packets counter reaches half of the maximum value or the maximum value. 0_B **DISABLE**: MMC Receive IPV4 UDP Checksum Disabled Packet Counter Interrupt Mask is disabled 1_B **ENABLE**: MMC Receive IPV4 UDP Checksum Disabled Packet Counter Interrupt Mask is enabled
RXIPV6GPIM	5	rw	**MMC Receive IPv6 Good Packet Counter Interrupt Mask** Setting this bit masks the interrupt when the RxIPv6_Good_Packets counter reaches half of the maximum value or the maximum value. 0_B **DISABLE**: MMC Receive IPV6 Good Packet Counter Interrupt Mask is disabled 1_B **ENABLE**: MMC Receive IPV6 Good Packet Counter Interrupt Mask is enabled
RXIPV6HERPIM	6	rw	**MMC Receive IPv6 Header Error Packet Counter Interrupt Mask** Setting this bit masks the interrupt when the RxIPv6_Header_Error_Packets counter reaches half of the maximum value or the maximum value. 0_B **DISABLE**: MMC Receive IPV6 Header Error Packet Counter Interrupt Mask is disabled 1_B **ENABLE**: MMC Receive IPV6 Header Error Packet Counter Interrupt Mask is enabled
RXIPV6NOPAYPIM	7	rw	**MMC Receive IPv6 No Payload Packet Counter Interrupt Mask** Setting this bit masks the interrupt when the RxIPv6_No_Payload_Packets counter reaches half of the maximum value or the maximum value. 0_B **DISABLE**: MMC Receive IPV6 No Payload Packet Counter Interrupt Mask is disabled 1_B **ENABLE**: MMC Receive IPV6 No Payload Packet Counter Interrupt Mask is enabled
RXUDPGPIM	8	rw	**MMC Receive UDP Good Packet Counter Interrupt Mask** Setting this bit masks the interrupt when the RxUDP_Good_Packets counter reaches half of the maximum value or the maximum value. 0_B **DISABLE**: MMC Receive UDP Good Packet Counter Interrupt Mask is disabled 1_B **ENABLE**: MMC Receive UDP Good Packet Counter Interrupt Mask is enabled
RXUDPERPIM	9	rw	**MMC Receive UDP Error Packet Counter Interrupt Mask** Setting this bit masks the interrupt when the RxUDP_Error_Packets counter reaches half of the maximum value or the maximum value. 0_B **DISABLE**: MMC Receive UDP Error Packet Counter Interrupt Mask is disabled 1_B **ENABLE**: MMC Receive UDP Error Packet Counter Interrupt Mask is enabled

(table continues...)

(continued)

Field	Bits	Type	Description
RXTCPGPIM	10	rw	**MMC Receive TCP Good Packet Counter Interrupt Mask** Setting this bit masks the interrupt when the RxTCP_Good_Packets counter reaches half of the maximum value or the maximum value. 0_B **DISABLE**: MMC Receive TCP Good Packet Counter Interrupt Mask is disabled 1_B **ENABLE**: MMC Receive TCP Good Packet Counter Interrupt Mask is enabled
RXTCPERPIM	11	rw	**MMC Receive TCP Error Packet Counter Interrupt Mask** Setting this bit masks the interrupt when the RxTCP_Error_Packets counter reaches half of the maximum value or the maximum value. 0_B **DISABLE**: MMC Receive TCP Error Packet Counter Interrupt Mask is disabled 1_B **ENABLE**: MMC Receive TCP Error Packet Counter Interrupt Mask is enabled
RXICMPGPIM	12	rw	**MMC Receive ICMP Good Packet Counter Interrupt Mask** Setting this bit masks the interrupt when the RxICMP_Good_Packets counter reaches half of the maximum value or the maximum value. 0_B **DISABLE**: MMC Receive ICMP Good Packet Counter Interrupt Mask is disabled 1_B **ENABLE**: MMC Receive ICMP Good Packet Counter Interrupt Mask is enabled
RXICMPERPIM	13	rw	**MMC Receive ICMP Error Packet Counter Interrupt Mask** Setting this bit masks the interrupt when the RxICMP_Error_Packets counter reaches half of the maximum value or the maximum value. 0_B **DISABLE**: MMC Receive ICMP Error Packet Counter Interrupt Mask is disabled 1_B **ENABLE**: MMC Receive ICMP Error Packet Counter Interrupt Mask is enabled
RXIPV4GOIM	16	rw	**MMC Receive IPv4 Good Octet Counter Interrupt Mask** Setting this bit masks the interrupt when the RxIPv4_Good_Octets counter reaches half of the maximum value or the maximum value. 0_B **DISABLE**: MMC Receive IPV4 Good Octet Counter Interrupt Mask is disabled 1_B **ENABLE**: MMC Receive IPV4 Good Octet Counter Interrupt Mask is enabled
RXIPV4HEROIM	17	rw	**MMC Receive IPv4 Header Error Octet Counter Interrupt Mask** Setting this bit masks the interrupt when the RxIPv4_Header_Error_Octets counter reaches half of the maximum value or the maximum value. 0_B **DISABLE**: MMC Receive IPV4 Header Error Octet Counter Interrupt Mask is disabled 1_B **ENABLE**: MMC Receive IPV4 Header Error Octet Counter Interrupt Mask is enabled

(table continues...)

(continued)

Field	Bits	Type	Description
RXIPV4NOPAYOIM	18	rw	**MMC Receive IPv4 No Payload Octet Counter Interrupt Mask** Setting this bit masks the interrupt when the RxIPv4_No_Payload_Octets counter reaches half of the maximum value or the maximum value. 0_B **DISABLE**: MMC Receive IPV4 No Payload Octet Counter Interrupt Mask is disabled 1_B **ENABLE**: MMC Receive IPV4 No Payload Octet Counter Interrupt Mask is enabled
RXIPV4FRAGOIM	19	rw	**MMC Receive IPv4 Fragmented Octet Counter Interrupt Mask** Setting this bit masks the interrupt when the RxIPv4_Fragmented_Octets counter reaches half of the maximum value or the maximum value. 0_B **DISABLE**: MMC Receive IPV4 Fragmented Octet Counter Interrupt Mask is disabled 1_B **ENABLE**: MMC Receive IPV4 Fragmented Octet Counter Interrupt Mask is enabled
RXIPV4UDSBLOIM	20	rw	**MMC Receive IPv4 UDP Checksum Disabled Octet Counter Interrupt Mask** Setting this bit masks the interrupt when the RxIPv4_UDP_Checksum_Disable_Octets counter reaches half of the maximum value or the maximum value. 0_B **DISABLE**: MMC Receive IPV4 UDP Checksum Disabled Octet Counter Interrupt Mask is disabled 1_B **ENABLE**: MMC Receive IPV4 UDP Checksum Disabled Octet Counter Interrupt Mask is enabled
RXIPV6GOIM	21	rw	**MMC Receive IPv6 Good Octet Counter Interrupt Mask** Setting this bit masks the interrupt when the RxIPv6_Good_Octets counter reaches half of the maximum value or the maximum value. 0_B **DISABLE**: MMC Receive IPV6 Good Octet Counter Interrupt Mask is disabled 1_B **ENABLE**: MMC Receive IPV6 Good Octet Counter Interrupt Mask is enabled
RXIPV6HEROIM	22	rw	**MMC Receive IPv6 Good Octet Counter Interrupt Mask** Setting this bit masks the interrupt when the RxIPv6_Header_Error_Octets counter reaches half of the maximum value or the maximum value. 0_B **DISABLE**: MMC Receive IPV6 Good Octet Counter Interrupt Mask is disabled 1_B **ENABLE**: MMC Receive IPV6 Good Octet Counter Interrupt Mask is enabled

(table continues...)

(continued)

Field	Bits	Type	Description
RXIPV6NOPAYOIM	23	rw	**MMC Receive IPv6 Header Error Octet Counter Interrupt Mask** Setting this bit masks the interrupt when the RxIPv6_No_Payload_Octets counter reaches half of the maximum value or the maximum value. 0_B **DISABLE**: MMC Receive IPV6 Header Error Octet Counter Interrupt Mask is disabled 1_B **ENABLE**: MMC Receive IPV6 Header Error Octet Counter Interrupt Mask is enabled
RXUDPGOIM	24	rw	**MMC Receive IPv6 No Payload Octet Counter Interrupt Mask** Setting this bit masks the interrupt when the RxUDP_Good_Octets counter reaches half of the maximum value or the maximum value. 0_B **DISABLE**: MMC Receive IPV6 No Payload Octet Counter Interrupt Mask is disabled 1_B **ENABLE**: MMC Receive IPV6 No Payload Octet Counter Interrupt Mask is enabled
RXUDPEROIM	25	rw	**MMC Receive UDP Good Octet Counter Interrupt Mask** Setting this bit masks the interrupt when the RxUDP_Error_Octets counter reaches half of the maximum value or the maximum value. 0_B **DISABLE**: MMC Receive UDP Good Octet Counter Interrupt Mask is disabled 1_B **ENABLE**: MMC Receive UDP Good Octet Counter Interrupt Mask is enabled
RXTCPGOIM	26	rw	**MMC Receive TCP Good Octet Counter Interrupt Mask** Setting this bit masks the interrupt when the RxTCP_Good_Octets counter reaches half of the maximum value or the maximum value. 0_B **DISABLE**: MMC Receive TCP Good Octet Counter Interrupt Mask is disabled 1_B **ENABLE**: MMC Receive TCP Good Octet Counter Interrupt Mask is enabled
RXTCPEROIM	27	rw	**MMC Receive TCP Error Octet Counter Interrupt Mask** Setting this bit masks the interrupt when the RxTCP_Error_Octets counter reaches half of the maximum value or the maximum value. 0_B **DISABLE**: MMC Receive TCP Error Octet Counter Interrupt Mask is disabled 1_B **ENABLE**: MMC Receive TCP Error Octet Counter Interrupt Mask is enabled
RXICMPGOIM	28	rw	**MMC Receive ICMP Good Octet Counter Interrupt Mask** Setting this bit masks the interrupt when the RxICMP_Good_Octets counter reaches half of the maximum value or the maximum value. 0_B **DISABLE**: MMC Receive ICMP Good Octet Counter Interrupt Mask is disabled 1_B **ENABLE**: MMC Receive ICMP Good Octet Counter Interrupt Mask is enabled

(table continues...)

(continued)

Field	Bits	Type	Description
RXICMPEROIM	29	rw	**MMC Receive ICMP Error Octet Counter Interrupt Mask** Setting this bit masks the interrupt when the RxICMP_Error_Octets counter reaches half of the maximum value or the maximum value. 0_B **DISABLE**: MMC Receive ICMP Error Octet Counter Interrupt Mask is disabled 1_B **ENABLE**: MMC Receive ICMP Error Octet Counter Interrupt Mask is enabled
0	14, 15, 31:30	r	**Reserved** Read as 0; should be written with 0.

14.6.390 Port j MMC IPC Rx Interrupt

This register maintains the interrupt that the receive IPC statistic counters generate. The MMC Receive Checksum Offload Interrupt register maintains the interrupts generated when receive IPC statistic counters reach half their maximum values (0x8000_0000_0000_0000 for 64 bit counter and 0x8000_0000 for 32 bit counter), and when they cross their maximum values (0xFFFF_FFFF_FFFF_FFFF for 64 bit counter and 0xFFFF_FFFF for 32 bit counter). When Counter Stop Rollover is set, the interrupts are set but the counter remains at all-ones.

The MMC Receive Checksum Offload Interrupt register is 32 bit wide. When the MMC IPC counter that caused the interrupt is read, its corresponding interrupt bit is cleared. The counter's least-significant byte lane (Bits[7:0]) must be read to clear the interrupt bit.

Portj_MMC_IPC_Rx_Interrupt (j=0-1) Offset address: $10A60_H+j*2000_H$

Port j MMC IPC Rx Interrupt Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		RXICMPEROIS	RXICMPGOIS	RXTCPEROIS	RXTCPGOIS	RXUDPEROIS	RXUDPGOIS	RXIPV6NOPAYOIS	RXIPV6HEROIS	RXIPV6GOIS	RXIPV4UDSBLOIS	RXIPV4FRAGOIS	RXIPV4NOPAYOIS	RXIPV4HEROIS	RXIPV4GOIS
r		r	r	r	r	r	r	r	r	r	r	r	r	r	r

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	RXICMPERPIS	RXICMPGPIS	RXTCPERPIS	RXTCPGPIS	RXUDPERPIS	RXUDPGPIS	RXIPV6NOPAYPIS	RXIPV6HERPIS	RXIPV6GPIS	RXIPV4UDSBLPIS	RXIPV4FRAGPIS	RXIPV4NOPAYPIS	RXIPV4HERPIS	RXIPV4GPIS
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

Field	Bits	Type	Description
RXIPV4GPIS	0	r	**MMC Receive IPv4 Good Packet Counter Interrupt Status** This bit is set when the RxIPv4_Good_Packets counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0_B **INACTIVE**: MMC Receive IPV4 Good Packet Counter Interrupt Status not detected 1_B **ACTIVE**: MMC Receive IPV4 Good Packet Counter Interrupt Status detected
RXIPV4HERPIS	1	r	**MMC Receive IPv4 Header Error Packet Counter Interrupt Status** This bit is set when the RxIPv4_Header_Error_Packets counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0_B **INACTIVE**: MMC Receive IPV4 Header Error Packet Counter Interrupt Status not detected 1_B **ACTIVE**: MMC Receive IPV4 Header Error Packet Counter Interrupt Status detected
RXIPV4NOPAYPIS	2	r	**MMC Receive IPv4 No Payload Packet Counter Interrupt Status** This bit is set when the RxIPv4_No_Payload_Packets counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0_B **INACTIVE**: MMC Receive IPV4 No Payload Packet Counter Interrupt Status not detected 1_B **ACTIVE**: MMC Receive IPV4 No Payload Packet Counter Interrupt Status detected
RXIPV4FRAGPIS	3	r	**MMC Receive IPv4 Fragmented Packet Counter Interrupt Status** This bit is set when the RxIPv4_Fragmented_Packets counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0_B **INACTIVE**: MMC Receive IPV4 Fragmented Packet Counter Interrupt Status not detected 1_B **ACTIVE**: MMC Receive IPV4 Fragmented Packet Counter Interrupt Status detected
RXIPV4UDSBLPIS	4	r	**MMC Receive IPv4 UDP Checksum Disabled Packet Counter Interrupt Status** This bit is set when the RxIPv4_UDP_Checksum_Disabled_Packets counter reaches half of the maximum value or the maximum value. 0_B **INACTIVE**: MMC Receive IPV4 UDP Checksum Disabled Packet Counter Interrupt Status not detected 1_B **ACTIVE**: MMC Receive IPV4 UDP Checksum Disabled Packet Counter Interrupt Status detected

(table continues...)

(continued)

Field	Bits	Type	Description
RXIPV6GPIS	5	r	**MMC Receive IPV6 Good Packet Counter Interrupt Status** This bit is set when the RxIPv6_Good_Packets counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0_B **INACTIVE**: MMC Receive IPV6 Good Packet Counter Interrupt Status not detected 1_B **ACTIVE**: MMC Receive IPV6 Good Packet Counter Interrupt Status detected
RXIPV6HERPIS	6	r	**MMC Receive IPv6 Header Error Packet Counter Interrupt Status** This bit is set when the RxIPv6_Header_Error_Packets counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0_B **INACTIVE**: MMC Receive IPV6 Header Error Packet Counter Interrupt Status not detected 1_B **ACTIVE**: MMC Receive IPV6 Header Error Packet Counter Interrupt Status detected
RXIPV6NOPAYPIS	7	r	**MMC Receive IPv6 No Payload Packet Counter Interrupt Status** This bit is set when the RxIPv6_No_Payload_Packets counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0_B **INACTIVE**: MMC Receive IPV6 No Payload Packet Counter Interrupt Status not detected 1_B **ACTIVE**: MMC Receive IPV6 No Payload Packet Counter Interrupt Status detected
RXUDPGPIS	8	r	**MC Receive UDP Good Packet Counter Interrupt Status** This bit is set when the RxUDP_Good_Packets counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0_B **INACTIVE**: MMC Receive UDP Good Packet Counter Interrupt Status not detected 1_B **ACTIVE**: MMC Receive UDP Good Packet Counter Interrupt Status detected
RXUDPERPIS	9	r	**MMC Receive UDP Error Packet Counter Interrupt Status** This bit is set when the RxUDP_Error_Packets counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0_B **INACTIVE**: MMC Receive UDP Error Packet Counter Interrupt Status not detected 1_B **ACTIVE**: MMC Receive UDP Error Packet Counter Interrupt Status detected

(table continues...)

(continued)

Field	Bits	Type	Description
RXTCPGPIS	10	r	**MMC Receive TCP Good Packet Counter Interrupt Status** This bit is set when the RxTCP_Good_Packets counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0_B **INACTIVE**: MMC Receive TCP Good Packet Counter Interrupt Status not detected 1_B **ACTIVE**: MMC Receive TCP Good Packet Counter Interrupt Status detected
RXTCPERPIS	11	r	**MMC Receive TCP Error Packet Counter Interrupt Status** This bit is set when the RxTCP_Error_Packets counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0_B **INACTIVE**: MMC Receive TCP Error Packet Counter Interrupt Status not detected 1_B **ACTIVE**: MMC Receive TCP Error Packet Counter Interrupt Status detected
RXICMPGPIS	12	r	**MMC Receive ICMP Good Packet Counter Interrupt Status** This bit is set when the RxICMP_Good_Packets counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0_B **INACTIVE**: MMC Receive ICMP Good Packet Counter Interrupt Status not detected 1_B **ACTIVE**: MMC Receive ICMP Good Packet Counter Interrupt Status detected
RXICMPERPIS	13	r	**MMC Receive ICMP Error Packet Counter Interrupt Status** This bit is set when the RxICMP_Error_Packets counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0_B **INACTIVE**: MMC Receive ICMP Error Packet Counter Interrupt Status not detected 1_B **ACTIVE**: MMC Receive ICMP Error Packet Counter Interrupt Status detected
RXIPV4GOIS	16	r	**MMC Receive IPv4 Good Octet Counter Interrupt Status** This bit is set when the RxIPv4_Good_Octets counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0_B **INACTIVE**: MMC Receive IPV4 Good Octet Counter Interrupt Status not detected 1_B **ACTIVE**: MMC Receive IPV4 Good Octet Counter Interrupt Status detected

(table continues...)

(continued)

Field	Bits	Type	Description
RXIPV4HEROIS	17	r	**MMC Receive IPv4 Header Error Octet Counter Interrupt Status** This bit is set when the RxIPv4_Header_Error_Octets counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0_B **INACTIVE**: MMC Receive IPV4 Header Error Octet Counter Interrupt Status not detected 1_B **ACTIVE**: MMC Receive IPV4 Header Error Octet Counter Interrupt Status detected
RXIPV4NOPAYOIS	18	r	**MMC Receive IPv4 No Payload Octet Counter Interrupt Status** This bit is set when the RxIPv4_No_Payload_Octets counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0_B **INACTIVE**: MMC Receive IPV4 No Payload Octet Counter Interrupt Status not detected 1_B **ACTIVE**: MMC Receive IPV4 No Payload Octet Counter Interrupt Status detected
RXIPV4FRAGOIS	19	r	**MMC Receive IPv4 Fragmented Octet Counter Interrupt Status** This bit is set when the RxIPv4_Fragmented_Octets counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0_B **INACTIVE**: MMC Receive IPV4 Fragmented Octet Counter Interrupt Status not detected 1_B **ACTIVE**: MMC Receive IPV4 Fragmented Octet Counter Interrupt Status detected
RXIPV4UDSBLOIS	20	r	**MMC Receive IPv4 UDP Checksum Disabled Octet Counter Interrupt Status** This bit is set when the RxIPv4_UDP_Checksum_Disable_Octets counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0_B **INACTIVE**: MMC Receive IPV4 UDP Checksum Disabled Octet Counter Interrupt Status not detected 1_B **ACTIVE**: MMC Receive IPV4 UDP Checksum Disabled Octet Counter Interrupt Status detected
RXIPV6GOIS	21	r	**MMC Receive IPv6 Good Octet Counter Interrupt Status** This bit is set when the RxIPv6_Good_Octets counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0_B **INACTIVE**: MMC Receive IPV6 Good Octet Counter Interrupt Status not detected 1_B **ACTIVE**: MMC Receive IPV6 Good Octet Counter Interrupt Status detected

(table continues...)

(continued)

Field	Bits	Type	Description
RXIPV6HEROIS	22	r	**MMC Receive IPv6 Header Error Octet Counter Interrupt Status** This bit is set when the RxIPv6_Header_Error_Octets counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0_B **INACTIVE**: MMC Receive IPV6 Header Error Octet Counter Interrupt Status not detected 1_B **ACTIVE**: MMC Receive IPV6 Header Error Octet Counter Interrupt Status detected
RXIPV6NOPAYOIS	23	r	**MMC Receive IPv6 No Payload Octet Counter Interrupt Status** This bit is set when the RxIPv6_No_Payload_Octets counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0_B **INACTIVE**: MMC Receive IPV6 No Payload Octet Counter Interrupt Status not detected 1_B **ACTIVE**: MMC Receive IPV6 No Payload Octet Counter Interrupt Status detected
RXUDPGOIS	24	r	**MMC Receive UDP Good Octet Counter Interrupt Status** This bit is set when the RxUDP_Good_Octets counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0_B **INACTIVE**: MMC Receive UDP Good Octet Counter Interrupt Status not detected 1_B **ACTIVE**: MMC Receive UDP Good Octet Counter Interrupt Status detected
RXUDPEROIS	25	r	**MMC Receive UDP Error Octet Counter Interrupt Status** This bit is set when the RxUDP_Error_Octets counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0_B **INACTIVE**: MMC Receive UDP Error Octet Counter Interrupt Status not detected 1_B **ACTIVE**: MMC Receive UDP Error Octet Counter Interrupt Status detected
RXTCPGOIS	26	r	**MMC Receive TCP Good Octet Counter Interrupt Status** This bit is set when the RxTCP_Good_Octets counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0_B **INACTIVE**: MMC Receive TCP Good Octet Counter Interrupt Status not detected 1_B **ACTIVE**: MMC Receive TCP Good Octet Counter Interrupt Status detected

(table continues...)

(continued)

Field	Bits	Type	Description
RXTCPEROIS	27	r	**MMC Receive TCP Error Octet Counter Interrupt Status** This bit is set when the RxTCP_Error_Octets counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0_B **INACTIVE**: MMC Receive TCP Error Octet Counter Interrupt Status not detected 1_B **ACTIVE**: MMC Receive TCP Error Octet Counter Interrupt Status detected
RXICMPGOIS	28	r	**MMC Receive ICMP Good Octet Counter Interrupt Status** This bit is set when the RxICMP_Good_Octets reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0_B **INACTIVE**: MMC Receive ICMP Good Octet Counter Interrupt Status not detected 1_B **ACTIVE**: MMC Receive ICMP Good Octet Counter Interrupt Status detected
RXICMPEROIS	29	r	**MMC Receive ICMP Error Octet Counter Interrupt Status** This bit is set when the RxICMP_Error_Octets counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0_B **INACTIVE**: MMC Receive ICMP Error Octet Counter Interrupt Status not detected 1_B **ACTIVE**: MMC Receive ICMP Error Octet Counter Interrupt Status detected
0	14, 15, 31:30	r	**Reserved** Read as 0; should be written with 0.

14.6.391 Port j RxIPv4 Good Packets Low

This register provides lower 32 bits of good IPv4 datagrams received by DWC_xgmac with the TCP, UDP, or ICMP payload.

Portj_RxIPv4_Good_Packets_Low (j=0-1) Offset address: $10A64_H+j*2000_H$

Port j RxIPv4 Good Packets Low Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RXIPV4GDPKTLO															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXIPV4GDPKTLO															
r															

Field	Bits	Type	Description
RXIPV4GDPKTLO	31:0	r	**RxIPv4 Good Packets** This field indicates lower 32 bits of good IPv4 datagrams received with the TCP, UDP, or ICMP payload.

14.6.392 Port j RxIPv4 Header Error Packets Low

RxIPv4 Header Error Packets. This register provides lower 32 bits of IPv4 datagrams received by DWC_xgmac with header (checksum, length, or version mismatch) errors.

Portj_RxIPv4_Header_Error_Packets_Low (j=0-1) Offset address: $10A6C_H+j*2000_H$

Port j RxIPv4 Header Error Packets Low Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RXIPV4HDRERRPKTLO															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXIPV4HDRERRPKTLO															
r															

Field	Bits	Type	Description
RXIPV4HDRERRPKTLO	31:0	r	**RxIPv4 Header Error Packets** This field indicates lower 32 bits of IPv4 datagrams received with header (checksum, length, or version mismatch) errors.

14.6.393 Port j RxIPv4 No Payload Packets Low

This register provides lower 32 bits of IPv4 datagram packets received by DWC_xgmac that did not have a TCP, UDP, or ICMP payload.

Portj_RxIPv4_No_Payload_Packets_Low (j=0-1) Offset address: $10A74_H+j*2000_H$

Port j RxIPv4 No Payload Packets Low Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RXIPV4NOPAYPKTLO															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXIPV4NOPAYPKTLO															
r															

Field	Bits	Type	Description
RXIPV4NOPAYPKTLO	31:0	r	**RxIPv4 Payload Packets** This field indicates lower 32 bits of IPv4 datagram packets received that did not have a TCP, UDP, or ICMP payload.

14.6.394 Port j RxIPv4 Fragmented Packets Low

This register provides lower 32 bits of good IPv4 datagrams received by DWC_xgmac with fragmentation.

Portj_RxIPv4_Fragmented_Packets_Low (j=0-1) Offset address: $10A7C_H+j*2000_H$

Port j RxIPv4 Fragmented Packets Low Kernel Reset value: $0000\ 0000_H$

Field	Bits	Type	Description
RXIPV4FRAGPKTLO	31:0	r	**RxIPv4 Fragmented Packets** This field indicates lower 32 bits of good IPv4 datagrams received with fragmentation.

14.6.395 Port j RxIPv4 UDP Checksum Disabled Packets Low

This register provides lower 32 bits of good IPv4 datagrams received by DWC_xgmac that had a UDP payload with checksum disabled.

Portj_RxIPv4_UDP_Checksum_Disabled_Packets_Low (j=0-1) Offset address: $10A84_H+j*2000_H$

Port j RxIPv4 UDP Checksum Disabled Packets Low Kernel Reset value: $0000\ 0000_H$

Field	Bits	Type	Description
RXIPV4UDSBLPKTLO	31:0	r	**RxIPv4 UDP Checksum Disabled Packets** This field indicates lower 32 bits of good IPv4 datagrams received that had a UDP payload with checksum disabled.

14.6.396 Port j RxIPv6 Good Packets Low

This register provides lower 32 bits of good IPv6 datagrams received by DWC_xgmac with the TCP, UDP, or ICMP payload.

Portj_RxIPv6_Good_Packets_Low (j=0-1) Offset address: $10A8C_H+j*2000_H$

Port j RxIPv6 Good Packets Low Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RXIPV6GDPKTLO															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXIPV6GDPKTLO															
r															

Field	Bits	Type	Description
RXIPV6GDPKTLO	31:0	r	**RxIPv6 Good Packets** This field indicates lower 32 bits of good IPv6 datagrams received with the TCP, UDP, or ICMP payload.

14.6.397 Port j RxIPv6 Header Error Packets Low

This register provides lower 32 bits of IPv6 datagrams received by DWC_xgmac with header (length or version mismatch) errors.

Portj_RxIPv6_Header_Error_Packets_Low (j=0-1) Offset address: $10A94_H+j*2000_H$

Port j RxIPv6 Header Error Packets Low Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RXIPV6HDRERRPKTLO															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXIPV6HDRERRPKTLO															
r															

Field	Bits	Type	Description
RXIPV6HDRERRPKTLO	31:0	r	**RxIPv6 Header Error Packets** This field indicates lower 32 bits of IPv6 datagrams received with header (length or version mismatch) errors.

14.6.398 Port j RxIPv6 No Payload Packets Low

This register provides lower 32 bits of IPv6 datagram packets received by DWC_xgmac that did not have a TCP, UDP, or ICMP payload. This includes all IPv6 datagrams with fragmentation or security extension headers.

Portj_RxIPv6_No_Payload_Packets_Low (j=0-1) Offset address: $10A9C_H+j*2000_H$

Port j RxIPv6 No Payload Packets Low Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RXIPV6NOPAYPKTLO															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXIPV6NOPAYPKTLO															
r															

Field	Bits	Type	Description
RXIPV6NOPAYPKTLO	31:0	r	**RxIPv6 Payload Packets** This field indicates lower 32 bits of IPv6 datagram packets received that did not have a TCP, UDP, or ICMP payload. This includes all IPv6 datagrams with fragmentation or security extension headers.

14.6.399 Port j RxUDP Good Packets Low

This register provides lower 32 bits of good IP datagrams received by DWC_xgmac with a good UDP payload. This counter is not updated when the RxIPv4_UDP_Checksum_Disabled_Packets counter is incremented.

Portj_RxUDP_Good_Packets_Low (j=0-1) Offset address: $10AA4_H+j*2000_H$

Port j RxUDP Good Packets Low Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RXUDPGDPKTLO															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXUDPGDPKTLO															
r															

Field	Bits	Type	Description
RXUDPGDPKTLO	31:0	r	**RxUDP Good Packets** This field indicates lower 32 bits of good IP datagrams received with a good UDP payload. This counter is not updated when the RxIPv4_UDP_Checksum_Disabled_Packets counter is incremented.

14.6.400 Port j RxUDP Error Packets Low

This register provides lower 32 bits of good IP datagrams received by DWC_xgmac whose UDP payload has a checksum error.

Portj_RxUDP_Error_Packets_Low (j=0-1) Offset address: $10AAC_H+j*2000_H$

Port j RxUDP Error Packets Low Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RXUDPERRPKTLO															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXUDPERRPKTLO															
r															

Field	Bits	Type	Description
RXUDPERRPKTLO	31:0	r	**RxUDP Error Packets** This field indicates lower 32 bits of good IP datagrams received whose UDP payload has a checksum error.

14.6.401 Port j RxTCP Good Packets Low

This register provides lower 32 bits of good IP datagrams received by DWC_xgmac with a good TCP payload.

Portj_RxTCP_Good_Packets_Low (j=0-1) Offset address: $10AB4_H+j*2000_H$

Port j RxTCP Good Packets Low Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RXTCPGDPKTLO															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXTCPGDPKTLO															
r															

Field	Bits	Type	Description
RXTCPGDPKTLO	31:0	r	**RxTCP Good Packets** This field indicates lower 32 bits of good IP datagrams received with a good TCP payload.

14.6.402 Port j RxTCP Error Packets Low

This register provides lower 32 bits of good IP datagrams received by DWC_xgmac whose TCP payload has a checksum error.

Portj_RxTCP_Error_Packets_Low (j=0-1) Offset address: $10ABC_H+j*2000_H$

Port j RxTCP Error Packets Low Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RXTCPERRPKTLO															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXTCPERRPKTLO															
r															

Field	Bits	Type	Description
RXTCPERRPKTLO	31:0	r	**RxTCP Error Packets** This field indicates lower 32 bits of good IP datagrams received whose TCP payload has a checksum error.

14.6.403 Port j RxICMP Good Packets Low

This register provides lower 32 bits of good IP datagrams received by DWC_xgmac with a good ICMP payload.

Portj_RxICMP_Good_Packets_Low (j=0-1) Offset address: $10AC4_H+j*2000_H$

Port j RxICMP Good Packets Low Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RXICMPGDPKTLO															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXICMPGDPKTLO															
r															

Field	Bits	Type	Description
RXICMPGDPKTLO	31:0	r	**RxICMP Good Packets** This field indicates lower 32 bits of good IP datagrams received with a good ICMP payload.

14.6.404 Port j RxICMP Error Packets Low

This register provides lower 32 bits of good IP datagrams received by DWC_xgmac whose ICMP payload has a checksum error.

Portj_RxICMP_Error_Packets_Low (j=0-1) Offset address: $10ACC_H+j*2000_H$

Port j RxICMP Error Packets Low Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RXICMPERRPKTLO															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXICMPERRPKTLO															
r															

Field	Bits	Type	Description
RXICMPERRPKTLO	31:0	r	**RxICMP Error Packets** This field indicates lower 32 bits of good IP datagrams received whose ICMP payload has a checksum error.

14.6.405 Port j RxIPv4 Good Octets Low

This register provides lower 32 bits of bytes received by DWC_xgmac in good IPv4 datagrams encapsulating TCP, UDP, or ICMP data. (Ethernet header, FCS, pad, or IP pad bytes are not included in this counter.

Portj_RxIPv4_Good_Octets_Low (j=0-1) Offset address: $10AD4_H+j*2000_H$

Port j RxIPv4 Good Octets Low Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RXIPV4GDOCTLO															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXIPV4GDOCTLO															
r															

Field	Bits	Type	Description
RXIPV4GDOCTLO	31:0	r	**RxIPv4 Good Octets** This field indicates lower 32 bits of bytes received in good IPv4 datagrams encapsulating TCP, UDP, or ICMP data. (Ethernet header, FCS, pad, or IP pad bytes are not included in this counter.

14.6.406 Port j RxIPv4 Header Error Octets Low

This register provides lower 32 bits of bytes received by DWC_xgmac in IPv4 datagrams with header errors (checksum, length, version mismatch). The value in the Length field of IPv4 header is used to update this counter. (Ethernet header, FCS, pad, or IP pad bytes are not included in this counter.

Portj_RxIPv4_Header_Error_Octets_Low (j=0-1) Offset address: $10ADC_H+j*2000_H$

Port j RxIPv4 Header Error Octets Low Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RXIPV4HDRERROCTLO															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXIPV4HDRERROCTLO															
r															

Field	Bits	Type	Description
RXIPV4HDRERROCTLO	31:0	r	**RxIPv4 Header Error Octets** This field indicates lower 32 bits of bytes received in IPv4 datagrams with header errors (checksum, length, version mismatch). The value in the Length field of IPv4 header is used to update this counter. (Ethernet header, FCS, pad, or IP pad bytes are not included in this counter.

14.6.407 Port j RxIPv4 No Payload Octets Low

This register provides lower 32 bits of bytes received by DWC_xgmac in IPv4 datagrams that did not have a TCP, UDP, or ICMP payload. The value in the Length field of IPv4 header is used to update this counter. (Ethernet header, FCS, pad, or IP pad bytes are not included in this counter.

Portj_RxIPv4_No_Payload_Octets_Low (j=0-1) Offset address: $10AE4_H+j*2000_H$

Port j RxIPv4 No Payload Octets Low Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RXIPV4NOPAYOCTLO															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXIPV4NOPAYOCTLO															
r															

Field	Bits	Type	Description
RXIPV4NOPAYOCTLO	31:0	r	**RxIPv4 Payload Octets** This field indicates lower 32 bits of bytes received in IPv4 datagrams that did not have a TCP, UDP, or ICMP payload. The value in the Length field of IPv4 header is used to update this counter. (Ethernet header, FCS, pad, or IP pad bytes are not included in this counter.

14.6.408 Port j RxIPv4 Fragmented Octets Low

This register provides lower 32 bits of bytes received by DWC_xgmac in fragmented IPv4 datagrams. The value in the Length field of IPv4 header is used to update this counter. (Ethernet header, FCS, pad, or IP pad bytes are not included in this counter.

Portj_RxIPv4_Fragmented_Octets_Low (j=0-1) Offset address: $10AEC_H+j*2000_H$

Port j RxIPv4 Fragmented Octets Low Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RXIPV4FRAGOCTLO															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXIPV4FRAGOCTLO															
r															

Field	Bits	Type	Description
RXIPV4FRAGOCTLO	31:0	r	**RxIPv4 Fragmented Octets** This field indicates lower 32 bits of bytes received in fragmented IPv4 datagrams. The value in the Length field of IPv4 header is used to update this counter. (Ethernet header, FCS, pad, or IP pad bytes are not included in this counter.

14.6.409 Port j RxIPv4 UDP Checksum Disable Octets Low

This register provides lower 32 bits of bytes received by DWC_xgmac in a UDP segment that had the UDP checksum disabled. This counter does not count IP Header bytes. (Ethernet header, FCS, pad, or IP pad bytes are not included in this counter.

Portj_RxIPv4_UDP_Checksum_Disable_Octets_Low (j=0-1) Offset address: $10AF4_H+j*2000_H$

Port j RxIPv4 UDP Checksum Disable Octets Low Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RXIPV4UDSBLOCTLO															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXIPV4UDSBLOCTLO															
r															

Field	Bits	Type	Description
RXIPV4UDSBLOCTLO	31:0	r	**RxIPv4 UDP Checksum Disable Octets** This field indicates lower 32 bits of bytes received in a UDP segment that had the UDP checksum disabled. This counter does not count IP Header bytes. (Ethernet header, FCS, pad, or IP pad bytes are not included in this counter.

14.6.410 Port j RxIPv6 Good Octets Low

This register provides lower 32 bits of bytes received by DWC_xgmac in good IPv6 datagrams encapsulating TCP, UDP, or ICMP data. (Ethernet header, FCS, pad, or IP pad bytes are not included in this counter.

Portj_RxIPv6_Good_Octets_Low (j=0-1) Offset address: $10AFC_H+j*2000_H$

Port j RxIPv6 Good Octets Low Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RXIPV6GDOCTLO															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXIPV6GDOCTLO															
r															

Field	Bits	Type	Description
RXIPV6GDOCTLO	31:0	r	**RxIPv6 Good Octets** This field indicates lower 32 bits of bytes received in good IPv6 datagrams encapsulating TCP, UDP, or ICMP data. (Ethernet header, FCS, pad, or IP pad bytes are not included in this counter.

14.6.411 Port j RxIPv6 Header Error Octets Low

This register provides lower 32 bits of bytes received by DWC_xgmac in IPv6 datagrams with header errors (length, version mismatch). The value in the Length field of IPv6 header is used to update this counter. (Ethernet header, FCS, pad, or IP pad bytes are not included in this counter.

Portj_RxIPv6_Header_Error_Octets_Low (j=0-1) Offset address: $10B04_H+j*2000_H$

Port j RxIPv6 Header Error Octets Low Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RXIPV6HDRERROCTLO															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXIPV6HDRERROCTLO															
r															

Field	Bits	Type	Description
RXIPV6HDRERROCTLO	31:0	r	**RxIPv6 Header Error Octets** This field indicates lower 32 bits of bytes received in IPv6 datagrams with header errors (length, version mismatch). The value in the Length field of IPv6 header is used to update this counter. (Ethernet header, FCS, pad, or IP pad bytes are not included in this counter.

14.6.412 Port j RxIPv6 No Payload Octets Low

This register provides lower 32 bits of bytes received by DWC_xgmac in IPv6 datagrams that did not have a TCP, UDP, or ICMP payload. The value in the Length field of IPv6 header is used to update this counter. (Ethernet header, FCS, pad, or IP pad bytes are not included in this counter.

Portj_RxIPv6_No_Payload_Octets_Low (j=0-1) Offset address: $10B0C_H+j*2000_H$

Port j RxIPv6 No Payload Octets Low Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RXIPV6NOPAYOCTLO															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXIPV6NOPAYOCTLO															
r															

Field	Bits	Type	Description
RXIPV6NOPAYOCTLO	31:0	r	**RxIPv6 Payload Octets** This field indicates lower 32 bits of bytes received in IPv6 datagrams that did not have a TCP, UDP, or ICMP payload. The value in the Length field of IPv6 header is used to update this counter. (Ethernet header, FCS, pad, or IP pad bytes are not included in this counter.

14.6.413 Port j RxUDP Good Octets Low

This register provides lower 32 bits of bytes received by DWC_xgmac in a good UDP segment. This counter does not count IP header bytes.

Portj_RxUDP_Good_Octets_Low (j=0-1) Offset address: $10B14_H+j*2000_H$

Port j RxUDP Good Octets Low Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RXUDPGDOCTLO															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXUDPGDOCTLO															
r															

Field	Bits	Type	Description
RXUDPGDOCTLO	31:0	r	**RxUDP Good Octets** This field indicates lower 32 bits of bytes received in a good UDP segment. This counter does not count IP header bytes.

14.6.414 Port j RxUDP Error Octets Low

This register provides lower 32 bits of bytes received by DWC_xgmac in a UDP segment that had checksum errors. This counter does not count IP header bytes.

Portj_RxUDP_Error_Octets_Low (j=0-1) Offset address: $10B1C_H+j*2000_H$

Port j RxUDP Error Octets Low Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RXUDPERROCTLO															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXUDPERROCTLO															
r															

Field	Bits	Type	Description
RXUDPERROCTLO	31:0	r	**RxUDP Error Octets** This field indicates lower 32 bits of bytes received in a UDP segment that had checksum errors. This counter does not count IP header bytes.

14.6.415 Port j RxTCP Good Octets Low

This register provides lower 32 bits of bytes received by DWC_xgmac in a good TCP segment. This counter does not count IP header bytes.

Portj_RxTCP_Good_Octets_Low (j=0-1) Offset address: $10B24_H+j*2000_H$

Port j RxTCP Good Octets Low Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RXTCPGDOCTLO															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXTCPGDOCTLO															
r															

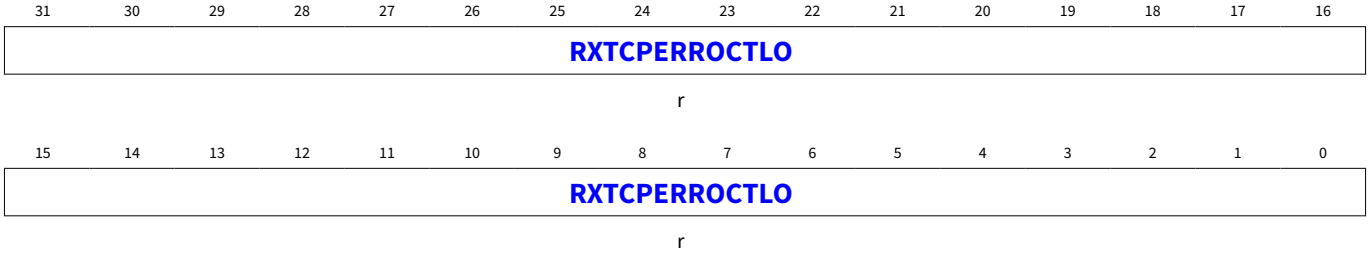
Field	Bits	Type	Description
RXTCPGDOCTLO	31:0	r	**RxTCP Good Octets** This field indicates lower 32 bits of bytes received in a good TCP segment. This counter does not count IP header bytes.

14.6.416 Port j RxTCP Error Octets Low

This register provides lower 32 bits of bytes received by DWC_xgmac in a TCP segment that had checksum errors. This counter does not count IP header bytes.

Portj_RxTCP_Error_Octets_Low (j=0-1) Offset address: $10B2C_H+j*2000_H$

Port j RxTCP Error Octets Low Kernel Reset value: $0000\ 0000_H$



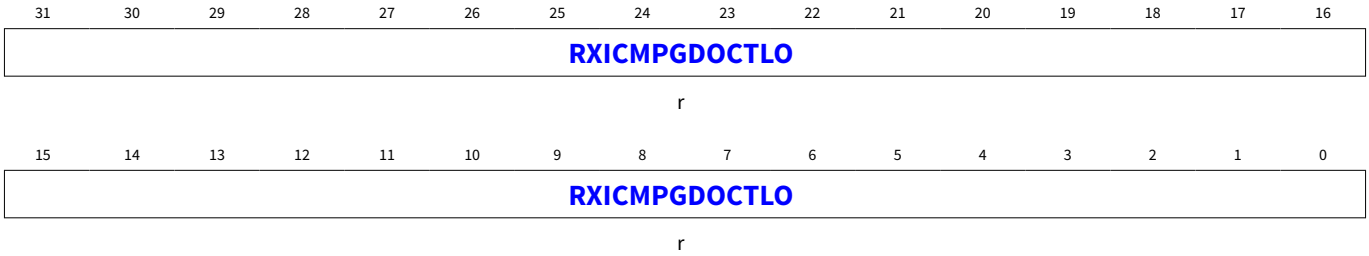
Field	Bits	Type	Description
RXTCPERROCTLO	31:0	r	**RxTCP Error Octets** This field indicates lower 32 bits of bytes received in a TCP segment that had checksum errors. This counter does not count IP header bytes.

14.6.417 Port j RxICMP Good Octets Low

This register provides lower 32 bits of bytes received by DWC_xgmac in a good ICMP segment. This counter does not count IP header bytes.

Portj_RxICMP_Good_Octets_Low (j=0-1) Offset address: $10B34_H+j*2000_H$

Port j RxICMP Good Octets Low Kernel Reset value: $0000\ 0000_H$



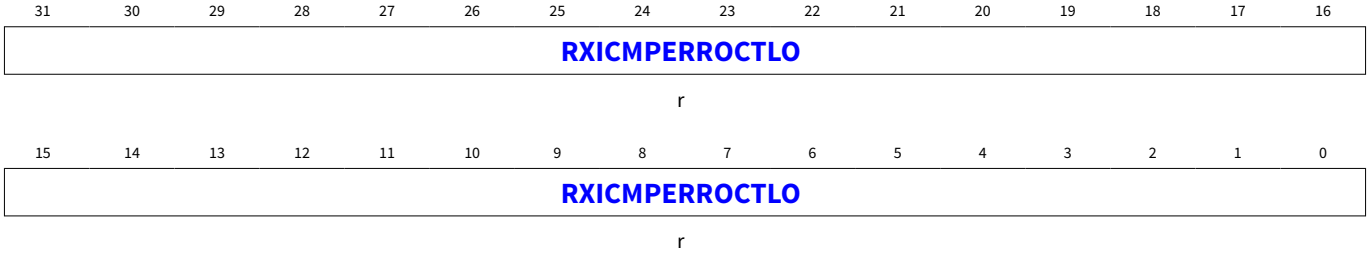
Field	Bits	Type	Description
RXICMPGDOCTLO	31:0	r	**RxICMP Good Octets** This field indicates lower 32 bits of bytes received in a good ICMP segment. This counter does not count IP header bytes.

14.6.418 Port j RxICMP Error Octets Low

This register provides lower 32 bits of bytes received by DWC_xgmac in a ICMP segment that had checksum errors. This counter does not count IP header bytes.

Portj_RxICMP_Error_Octets_Low (j=0-1) Offset address: $10B3C_H+j*2000_H$

Port j RxICMP Error Octets Low Kernel Reset value: $0000\ 0000_H$



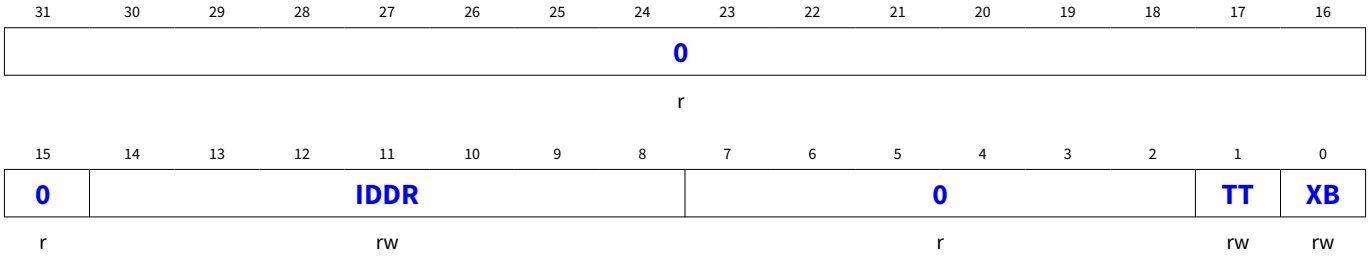
Field	Bits	Type	Description
RXICMPERROCTLO	31:0	r	**RxICMP Error Octets** This field indicates lower 32 bits of bytes received in a ICMP segment that had checksum errors. This counter does not count IP header bytes.

14.6.419 Port j MAC L3 L4 Address Control

This register has the indirect address and control for accessing the Layer 3 and Layer 4 Filter registers.

Portj_MAC_L3_L4_Address_Control (j=0-1) Offset address: $10C00_H+j*2000_H$

Port j MAC L3 L4 Address Control Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
XB	0	rw	**Transfer Busy** When set, initiates a read or write transaction (as indicated by TT) is initiated to the L3_L4 Filter register. When reset, indicates that read or write transaction to L3-L4 Filter register is complete. You must read and confirm that this bit is 0 (previous transfer is completed) before setting this bit again.
TT	1	rw	**Transfer Type** When set, a read transfer from the L3_L4 Filter register addressed by IDDR is executed. The data read from the register is available in the MAC_L3_L4_Data register. When reset, a write transfer to the L3_L4 Filter register addressed by IDDR is executed. The data is transferred from the MAC_L3_L4_Data register and written to the respective L3_L4 Filter register. You must write the MAC_L3_L4_Data register before initiating the write transaction.

(table continues...)

(continued)

Field	Bits	Type	Description
IDDR	14:8	rw	**L3_L4 Filter Address** This field contains the indirect address to the L3_L4 Filter registers. The IDDR[3:0] indicates the type of L3_L4 Filter register while bits[8:4] indicate the Filter number (0 to 7). IDDR[3:0] Values: - 0: MAC_L3_L4_Control - 1: MAC_Layer4_Address - 2-3: Reserved - 4: MAC_Layer3_Addr0_Reg - 5: MAC_Layer3_Addr1_Reg - 6: MAC_Layer3_Addr2_Reg - 7: MAC_Layer3_Addr3_Reg - 8 to 15: Reserved
0	7:2, 31:15	r	**Reserved** Read as 0; should be written with 0.

14.6.420 Port j MAC L3 L4 Data

This register has the indirect data and control for accessing the Layer 3 and Layer 4 Filter registers.

Portj_MAC_L3_L4_Data (j=0-1) Offset address: $10C04_H+j*2000_H$

Port j MAC L3 L4 Data Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
IDATA															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IDATA															
rw															

Field	Bits	Type	Description
IDATA	31:0	rw	**Indirect Data** The data written to or read from the L3_L4 Filter register.

14.6.421 Port j MAC L3 L4 Control0

The Layer 3 and Layer 4 Control register controls the operations of filter 0 of Layer 3 and Layer 4.

Portj_MAC_L3_L4_Control0 (j=0-1) Offset address: $10C04_H+j*2000_H$

Port j MAC L3 L4 Control0 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMCHEN	0	0			DMCHN			0	0	L4DPIM0	L4DPM0	L4SPIM0	L4SPM0	0	L4PEN0
rw	r	r			rw			r	r	rw	rw	rw	rw	r	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
L3HDBM0					L3HSBM0					L3DAIM0	L3DAM0	L3SAIM0	L3SAM0	0	L3PEN0
rw					rw					rw	rw	rw	rw	r	rw

Field	Bits	Type	Description
L3PEN0	0	rw	**Layer 3 Protocol Enable** When this bit is set, the Layer 3 IP Source or Destination Address matching is enabled for IPv6 packets. When this bit is reset, the Layer 3 IP Source or Destination Address matching is enabled for IPv4 packets. The Layer 3 matching is done only when the L3SAM0 or L3DAM0 bit is set.
L3SAM0	2	rw	**Layer 3 IP SA Match Enable** When this bit is set, the Layer 3 IP Source Address field is enabled for matching. When this bit is reset, the MAC ignores the Layer 3 IP Source Address field for matching. Note: When the L3PEN0 bit is set, you must set either this bit or the L3DAM0 bit because either IPv6 SA or DA can be checked for filtering.
L3SAIM0	3	rw	**Layer 3 IP SA Inverse Match Enable** When this bit is set, the Layer 3 IP Source Address field is enabled for inverse matching. When this bit reset, the Layer 3 IP Source Address field is enabled for perfect matching. This bit is valid and applicable only when the L3SAM0 bit is set.
L3DAM0	4	rw	**Layer 3 IP DA Match Enable** When this bit is set, the Layer 3 IP Destination Address field is enabled for matching. When this bit is reset, the MAC ignores the Layer 3 IP Destination Address field for matching. Note: When the L3PEN0 bit is set, you must set either this bit or the L3SAM0 bit because either IPv6 DA or SA can be checked for filtering.
L3DAIM0	5	rw	**Layer 3 IP DA Inverse Match Enable** When this bit is set, the Layer 3 IP Destination Address field is enabled for inverse matching. When this bit is reset, the Layer 3 IP Destination Address field is enabled for perfect matching. This bit is valid and applicable only when the L3DAM0 bit is set high.

(table continues...)

(continued)

Field	Bits	Type	Description
L3HSBM0	10:6	rw	**Layer 3 IP SA Higher Bits Match** IPv4 Packets: This field contains the number of lower bits of IP Source Address that are masked for matching in the IPv4 packets. The following list describes the values of this field: - 0: No bits are masked. - 1: Lsb[0] is masked - 2: Two Lsbs [1:0] are masked - ... - 31: All bits except Msb are masked. IPv6 Packets: This field contains Bits[4:0] of L3HSBM0. These bits indicate the number of higher bits of IP Source or Destination Address matched in the IPv6 packets. This field is valid and applicable only when the L3DAM0 or L3SAM0 bit is set high.
L3HDBM0	15:11	rw	**Layer 3 IP DA Higher Bits Match** IPv4 Packets: This field contains the number of higher bits of IP Destination Address that are matched in the IPv4 packets. The following list describes the values of this field: - 0: No bits are masked. - 1: LSB[0] is masked - 2: Two Lsbs [1:0] are masked - ... - 31: All bits except Msb are masked. IPv6 Packets: Bits[12:11] of this field correspond to Bits[6:5] of L3HSBM0 which indicate the number of lower bits of IP Source or Destination Address that are masked in the IPv6 packets. The following list describes the concatenated values of the L3HDBM0[1:0] and L3HSBM0 bits: - 0: No bits are masked. - 1: LSB[0] is masked. - 2: Two Lsbs [1:0] are masked - ... - 127: All bits except Msb are masked. This field is valid and applicable only when the L3DAM0 or L3SAM0 bit is set.
L4PEN0	16	rw	**Layer 4 Protocol Enable** When this bit is set, the Source and Destination Port number fields of UDP packets are used for matching. When this bit is reset, the Source and Destination Port number fields of TCP packets are used for matching. The Layer 4 matching is done only when the L4SPM0 or L4DPM0 bit is set.

(table continues...)

(continued)

Field	Bits	Type	Description
L4SPM0	18	rw	**Layer 4 Source Port Match Enable** When this bit is set, the Layer 4 Source Port number field is enabled for matching. When this bit is reset, the MAC ignores the Layer 4 Source Port number field for matching.
L4SPIM0	19	rw	**Layer 4 Source Port Inverse Match Enable** When this bit is set, the Layer 4 Source Port number field is enabled for inverse matching. When this bit is reset, the Layer 4 Source Port number field is enabled for perfect matching. This bit is valid and applicable only when the L4SPM0 bit is set high.
L4DPM0	20	rw	**Layer 4 Destination Port Match Enable** When this bit is set, the Layer 4 Destination Port number field is enabled for matching. When this bit is reset, the MAC ignores the Layer 4 Destination Port number field for matching.
L4DPIM0	21	rw	**Layer 4 Destination Port Inverse Match Enable** When this bit is set, the Layer 4 Destination Port number field is enabled for inverse matching. When this bit is reset, the Layer 4 Destination Port number field is enabled for perfect matching. This bit is valid and applicable only when the L4DPM0 bit is set high.
DMCHN	26:24	rw	**DMA Channel Number** When DMCHEN is set high, this field selects the DMA Channel number to which the packet passed by this filter is routed. The width of this field depends on the number of the DMA channels present in your configuration.
DMCHEN	31	rw	**DMA Channel Select Enable** When set, this bit enables the selection of the DMA channel number for the packet that is passed by this L3_L4 filter. The DMA channel is indicated by the DMCHN bits. When the bit is reset, the DMA channel is not decided by the filter.
0	1, 17, 22, 23, 29:27, 30	r	**Reserved** Read as 0; should be written with 0.

14.6.422 Port j MAC L3 L4 Control1

The Layer 3 and Layer 4 Control register controls the operations of filter 0 of Layer 3 and Layer 4.

Portj_MAC_L3_L4_Control1 (j=0-1) Offset address: $10C04_H+j*2000_H$

Port j MAC L3 L4 Control1 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMCHEN	0	0			DMCHN			0	0	L4DPIM1	L4DPM1	L4SPIM1	L4SPM1	0	L4PEN1
rw	r	r			rw			r	r	rw	rw	rw	rw	r	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
L3HDBM1					L3HSBM1					L3DAIM1	L3DAM1	L3SAIM1	L3SAM1	0	L3PEN1
rw					rw					rw	rw	rw	rw	r	rw

Field	Bits	Type	Description
L3PEN1	0	rw	**Layer 3 Protocol Enable** When this bit is set, the Layer 3 IP Source or Destination Address matching is enabled for IPv6 packets. When this bit is reset, the Layer 3 IP Source or Destination Address matching is enabled for IPv4 packets. The Layer 3 matching is done only when the L3SAM0 or L3DAM0 bit is set.
L3SAM1	2	rw	**Layer 3 IP SA Match Enable** When this bit is set, the Layer 3 IP Source Address field is enabled for matching. When this bit is reset, the MAC ignores the Layer 3 IP Source Address field for matching. Note: When the L3PEN0 bit is set, you must set either this bit or the L3DAM0 bit because either IPv6 SA or DA can be checked for filtering.
L3SAIM1	3	rw	**Layer 3 IP SA Inverse Match Enable** When this bit is set, the Layer 3 IP Source Address field is enabled for inverse matching. When this bit reset, the Layer 3 IP Source Address field is enabled for perfect matching. This bit is valid and applicable only when the L3SAM0 bit is set.
L3DAM1	4	rw	**Layer 3 IP DA Match Enable** When this bit is set, the Layer 3 IP Destination Address field is enabled for matching. When this bit is reset, the MAC ignores the Layer 3 IP Destination Address field for matching. Note: When the L3PEN0 bit is set, you must set either this bit or the L3SAM0 bit because either IPv6 DA or SA can be checked for filtering.
L3DAIM1	5	rw	**Layer 3 IP DA Inverse Match Enable** When this bit is set, the Layer 3 IP Destination Address field is enabled for inverse matching. When this bit is reset, the Layer 3 IP Destination Address field is enabled for perfect matching. This bit is valid and applicable only when the L3DAM0 bit is set high.

(table continues...)

(continued)

Field	Bits	Type	Description
L3HSBM1	10:6	rw	**Layer 3 IP SA Higher Bits Match** IPv4 Packets: This field contains the number of lower bits of IP Source Address that are masked for matching in the IPv4 packets. The following list describes the values of this field: - 0: No bits are masked. - 1: Lsb[0] is masked - 2: Two Lsbs [1:0] are masked - ... - 31: All bits except Msb are masked. IPv6 Packets: This field contains Bits[4:0] of L3HSBM0. These bits indicate the number of higher bits of IP Source or Destination Address matched in the IPv6 packets. This field is valid and applicable only when the L3DAM0 or L3SAM0 bit is set high.
L3HDBM1	15:11	rw	**Layer 3 IP DA Higher Bits Match** IPv4 Packets: This field contains the number of higher bits of IP Destination Address that are matched in the IPv4 packets. The following list describes the values of this field: - 0: No bits are masked. - 1: LSB[0] is masked - 2: Two Lsbs [1:0] are masked - ... - 31: All bits except Msb are masked. IPv6 Packets: Bits[12:11] of this field correspond to Bits[6:5] of L3HSBM0 which indicate the number of lower bits of IP Source or Destination Address that are masked in the IPv6 packets. The following list describes the concatenated values of the L3HDBM0[1:0] and L3HSBM0 bits: - 0: No bits are masked. - 1: LSB[0] is masked. - 2: Two Lsbs [1:0] are masked - ... - 127: All bits except Msb are masked. This field is valid and applicable only when the L3DAM0 or L3SAM0 bit is set.
L4PEN1	16	rw	**Layer 4 Protocol Enable** When this bit is set, the Source and Destination Port number fields of UDP packets are used for matching. When this bit is reset, the Source and Destination Port number fields of TCP packets are used for matching. The Layer 4 matching is done only when the L4SPM0 or L4DPM0 bit is set.

(table continues...)

(continued)

Field	Bits	Type	Description
L4SPM1	18	rw	**Layer 4 Source Port Match Enable** When this bit is set, the Layer 4 Source Port number field is enabled for matching. When this bit is reset, the MAC ignores the Layer 4 Source Port number field for matching.
L4SPIM1	19	rw	**Layer 4 Source Port Inverse Match Enable** When this bit is set, the Layer 4 Source Port number field is enabled for inverse matching. When this bit is reset, the Layer 4 Source Port number field is enabled for perfect matching. This bit is valid and applicable only when the L4SPM0 bit is set high.
L4DPM1	20	rw	**Layer 4 Destination Port Match Enable** When this bit is set, the Layer 4 Destination Port number field is enabled for matching. When this bit is reset, the MAC ignores the Layer 4 Destination Port number field for matching.
L4DPIM1	21	rw	**Layer 4 Destination Port Inverse Match Enable** When this bit is set, the Layer 4 Destination Port number field is enabled for inverse matching. When this bit is reset, the Layer 4 Destination Port number field is enabled for perfect matching. This bit is valid and applicable only when the L4DPM0 bit is set high.
DMCHN	26:24	rw	**DMA Channel Number** When DMCHEN is set high, this field selects the DMA Channel number to which the packet passed by this filter is routed. The width of this field depends on the number of the DMA channels present in your configuration.
DMCHEN	31	rw	**DMA Channel Select Enable** When set, this bit enables the selection of the DMA channel number for the packet that is passed by this L3_L4 filter. The DMA channel is indicated by the DMCHN bits. When the bit is reset, the DMA channel is not decided by the filter.
0	1, 17, 22, 23, 29:27, 30	r	**Reserved** Read as 0; should be written with 0.

14.6.423 Port j MAC L3 L4 Control2

The Layer 3 and Layer 4 Control register controls the operations of filter 0 of Layer 3 and Layer 4.

Portj_MAC_L3_L4_Control2 (j=0-1) Offset address: $10C04_H+j*2000_H$

Port j MAC L3 L4 Control2 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMCHEN	0	0			DMCHN			0	0	L4DPIM2	L4DPM2	L4SPIM2	L4SPM2	0	L4PEN2
rw	r	r			rw			r	r	rw	rw	rw	rw	r	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
L3HDBM2					L3HSBM2					L3DAIM2	L3DAM2	L3SAIM2	L3SAM2	0	L3PEN2
rw					rw					rw	rw	rw	rw	r	rw

Field	Bits	Type	Description
L3PEN2	0	rw	**Layer 3 Protocol Enable** When this bit is set, the Layer 3 IP Source or Destination Address matching is enabled for IPv6 packets. When this bit is reset, the Layer 3 IP Source or Destination Address matching is enabled for IPv4 packets. The Layer 3 matching is done only when the L3SAM0 or L3DAM0 bit is set.
L3SAM2	2	rw	**Layer 3 IP SA Match Enable** When this bit is set, the Layer 3 IP Source Address field is enabled for matching. When this bit is reset, the MAC ignores the Layer 3 IP Source Address field for matching. Note: When the L3PEN0 bit is set, you must set either this bit or the L3DAM0 bit because either IPv6 SA or DA can be checked for filtering.
L3SAIM2	3	rw	**Layer 3 IP SA Inverse Match Enable** When this bit is set, the Layer 3 IP Source Address field is enabled for inverse matching. When this bit reset, the Layer 3 IP Source Address field is enabled for perfect matching. This bit is valid and applicable only when the L3SAM0 bit is set.
L3DAM2	4	rw	**Layer 3 IP DA Match Enable** When this bit is set, the Layer 3 IP Destination Address field is enabled for matching. When this bit is reset, the MAC ignores the Layer 3 IP Destination Address field for matching. Note: When the L3PEN0 bit is set, you must set either this bit or the L3SAM0 bit because either IPv6 DA or SA can be checked for filtering.
L3DAIM2	5	rw	**Layer 3 IP DA Inverse Match Enable** When this bit is set, the Layer 3 IP Destination Address field is enabled for inverse matching. When this bit is reset, the Layer 3 IP Destination Address field is enabled for perfect matching. This bit is valid and applicable only when the L3DAM0 bit is set high.

(table continues...)

(continued)

Field	Bits	Type	Description
L3HSBM2	10:6	rw	**Layer 3 IP SA Higher Bits Match** IPv4 Packets: This field contains the number of lower bits of IP Source Address that are masked for matching in the IPv4 packets. The following list describes the values of this field: - 0: No bits are masked. - 1: Lsb[0] is masked - 2: Two Lsbs [1:0] are masked - ... - 31: All bits except Msb are masked. IPv6 Packets: This field contains Bits[4:0] of L3HSBM0. These bits indicate the number of higher bits of IP Source or Destination Address matched in the IPv6 packets. This field is valid and applicable only when the L3DAM0 or L3SAM0 bit is set high.
L3HDBM2	15:11	rw	**Layer 3 IP DA Higher Bits Match** IPv4 Packets: This field contains the number of higher bits of IP Destination Address that are matched in the IPv4 packets. The following list describes the values of this field: - 0: No bits are masked. - 1: LSB[0] is masked - 2: Two Lsbs [1:0] are masked - ... - 31: All bits except Msb are masked. IPv6 Packets: Bits[12:11] of this field correspond to Bits[6:5] of L3HSBM0 which indicate the number of lower bits of IP Source or Destination Address that are masked in the IPv6 packets. The following list describes the concatenated values of the L3HDBM0[1:0] and L3HSBM0 bits: - 0: No bits are masked. - 1: LSB[0] is masked. - 2: Two Lsbs [1:0] are masked - ... - 127: All bits except Msb are masked. This field is valid and applicable only when the L3DAM0 or L3SAM0 bit is set.
L4PEN2	16	rw	**Layer 4 Protocol Enable** When this bit is set, the Source and Destination Port number fields of UDP packets are used for matching. When this bit is reset, the Source and Destination Port number fields of TCP packets are used for matching. The Layer 4 matching is done only when the L4SPM0 or L4DPM0 bit is set.

(table continues...)

(continued)

Field	Bits	Type	Description
L4SPM2	18	rw	**Layer 4 Source Port Match Enable** When this bit is set, the Layer 4 Source Port number field is enabled for matching. When this bit is reset, the MAC ignores the Layer 4 Source Port number field for matching.
L4SPIM2	19	rw	**Layer 4 Source Port Inverse Match Enable** When this bit is set, the Layer 4 Source Port number field is enabled for inverse matching. When this bit is reset, the Layer 4 Source Port number field is enabled for perfect matching. This bit is valid and applicable only when the L4SPM0 bit is set high.
L4DPM2	20	rw	**Layer 4 Destination Port Match Enable** When this bit is set, the Layer 4 Destination Port number field is enabled for matching. When this bit is reset, the MAC ignores the Layer 4 Destination Port number field for matching.
L4DPIM2	21	rw	**Layer 4 Destination Port Inverse Match Enable** When this bit is set, the Layer 4 Destination Port number field is enabled for inverse matching. When this bit is reset, the Layer 4 Destination Port number field is enabled for perfect matching. This bit is valid and applicable only when the L4DPM0 bit is set high.
DMCHN	26:24	rw	**DMA Channel Number** When DMCHEN is set high, this field selects the DMA Channel number to which the packet passed by this filter is routed. The width of this field depends on the number of the DMA channels present in your configuration.
DMCHEN	31	rw	**DMA Channel Select Enable** When set, this bit enables the selection of the DMA channel number for the packet that is passed by this L3_L4 filter. The DMA channel is indicated by the DMCHN bits. When the bit is reset, the DMA channel is not decided by the filter.
0	1, 17, 22, 23, 29:27, 30	r	**Reserved** Read as 0; should be written with 0.

14.6.424 Port j MAC L3 L4 Control3

The Layer 3 and Layer 4 Control register controls the operations of filter 0 of Layer 3 and Layer 4.

Portj_MAC_L3_L4_Control3 (j=0-1) Offset address: $10C04_H+j*2000_H$

Port j MAC L3 L4 Control3 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMCHEN	0	0			DMCHN			0	0	L4DPIM3	L4DPM3	L4SPIM3	L4SPM3	0	L4PEN3
rw	r	r			rw			r	r	rw	rw	rw	rw	r	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
L3HDBM3					L3HSBM3					L3DAIM3	L3DAM3	L3SAIM3	L3SAM3	0	L3PEN3
rw					rw					rw	rw	rw	rw	r	rw

Field	Bits	Type	Description
L3PEN3	0	rw	**Layer 3 Protocol Enable** When this bit is set, the Layer 3 IP Source or Destination Address matching is enabled for IPv6 packets. When this bit is reset, the Layer 3 IP Source or Destination Address matching is enabled for IPv4 packets. The Layer 3 matching is done only when the L3SAM0 or L3DAM0 bit is set.
L3SAM3	2	rw	**Layer 3 IP SA Match Enable** When this bit is set, the Layer 3 IP Source Address field is enabled for matching. When this bit is reset, the MAC ignores the Layer 3 IP Source Address field for matching. Note: When the L3PEN0 bit is set, you must set either this bit or the L3DAM0 bit because either IPv6 SA or DA can be checked for filtering.
L3SAIM3	3	rw	**Layer 3 IP SA Inverse Match Enable** When this bit is set, the Layer 3 IP Source Address field is enabled for inverse matching. When this bit reset, the Layer 3 IP Source Address field is enabled for perfect matching. This bit is valid and applicable only when the L3SAM0 bit is set.
L3DAM3	4	rw	**Layer 3 IP DA Match Enable** When this bit is set, the Layer 3 IP Destination Address field is enabled for matching. When this bit is reset, the MAC ignores the Layer 3 IP Destination Address field for matching. Note: When the L3PEN0 bit is set, you must set either this bit or the L3SAM0 bit because either IPv6 DA or SA can be checked for filtering.
L3DAIM3	5	rw	**Layer 3 IP DA Inverse Match Enable** When this bit is set, the Layer 3 IP Destination Address field is enabled for inverse matching. When this bit is reset, the Layer 3 IP Destination Address field is enabled for perfect matching. This bit is valid and applicable only when the L3DAM0 bit is set high.

(table continues...)

(continued)

Field	Bits	Type	Description
L3HSBM3	10:6	rw	**Layer 3 IP SA Higher Bits Match** IPv4 Packets: This field contains the number of lower bits of IP Source Address that are masked for matching in the IPv4 packets. The following list describes the values of this field: - 0: No bits are masked. - 1: Lsb[0] is masked - 2: Two Lsbs [1:0] are masked - ... - 31: All bits except Msb are masked. IPv6 Packets: This field contains Bits[4:0] of L3HSBM0. These bits indicate the number of higher bits of IP Source or Destination Address matched in the IPv6 packets. This field is valid and applicable only when the L3DAM0 or L3SAM0 bit is set high.
L3HDBM3	15:11	rw	**Layer 3 IP DA Higher Bits Match** IPv4 Packets: This field contains the number of higher bits of IP Destination Address that are matched in the IPv4 packets. The following list describes the values of this field: - 0: No bits are masked. - 1: LSB[0] is masked - 2: Two Lsbs [1:0] are masked - ... - 31: All bits except Msb are masked. IPv6 Packets: Bits[12:11] of this field correspond to Bits[6:5] of L3HSBM0 which indicate the number of lower bits of IP Source or Destination Address that are masked in the IPv6 packets. The following list describes the concatenated values of the L3HDBM0[1:0] and L3HSBM0 bits: - 0: No bits are masked. - 1: LSB[0] is masked. - 2: Two Lsbs [1:0] are masked - ... - 127: All bits except Msb are masked. This field is valid and applicable only when the L3DAM0 or L3SAM0 bit is set.
L4PEN3	16	rw	**Layer 4 Protocol Enable** When this bit is set, the Source and Destination Port number fields of UDP packets are used for matching. When this bit is reset, the Source and Destination Port number fields of TCP packets are used for matching. The Layer 4 matching is done only when the L4SPM0 or L4DPM0 bit is set.

(table continues...)

(continued)

Field	Bits	Type	Description
L4SPM3	18	rw	**Layer 4 Source Port Match Enable** When this bit is set, the Layer 4 Source Port number field is enabled for matching. When this bit is reset, the MAC ignores the Layer 4 Source Port number field for matching.
L4SPIM3	19	rw	**Layer 4 Source Port Inverse Match Enable** When this bit is set, the Layer 4 Source Port number field is enabled for inverse matching. When this bit is reset, the Layer 4 Source Port number field is enabled for perfect matching. This bit is valid and applicable only when the L4SPM0 bit is set high.
L4DPM3	20	rw	**Layer 4 Destination Port Match Enable** When this bit is set, the Layer 4 Destination Port number field is enabled for matching. When this bit is reset, the MAC ignores the Layer 4 Destination Port number field for matching.
L4DPIM3	21	rw	**Layer 4 Destination Port Inverse Match Enable** When this bit is set, the Layer 4 Destination Port number field is enabled for inverse matching. When this bit is reset, the Layer 4 Destination Port number field is enabled for perfect matching. This bit is valid and applicable only when the L4DPM0 bit is set high.
DMCHN	26:24	rw	**DMA Channel Number** When DMCHEN is set high, this field selects the DMA Channel number to which the packet passed by this filter is routed. The width of this field depends on the number of the DMA channels present in your configuration.
DMCHEN	31	rw	**DMA Channel Select Enable** When set, this bit enables the selection of the DMA channel number for the packet that is passed by this L3_L4 filter. The DMA channel is indicated by the DMCHN bits. When the bit is reset, the DMA channel is not decided by the filter.
0	1, 17, 22, 23, 29:27, 30	r	**Reserved** Read as 0; should be written with 0.

14.6.425 Port j MAC L3 L4 Control4

The Layer 3 and Layer 4 Control register controls the operations of filter 0 of Layer 3 and Layer 4.

Portj_MAC_L3_L4_Control4 (j=0-1) Offset address: $10C04_H+j*2000_H$

Port j MAC L3 L4 Control4 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMCHEN	0	0			DMCHN			0	0	L4DPIM4	L4DPM4	L4SPIM4	L4SPM4	0	L4PEN4
rw	r	r			rw			r	r	rw	rw	rw	rw	r	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
L3HDBM4					L3HSBM4					L3DAIM4	L3DAM4	L3SAIM4	L3SAM4	0	L3PEN4
rw					rw					rw	rw	rw	rw	r	rw

Field	Bits	Type	Description
L3PEN4	0	rw	**Layer 3 Protocol Enable** When this bit is set, the Layer 3 IP Source or Destination Address matching is enabled for IPv6 packets. When this bit is reset, the Layer 3 IP Source or Destination Address matching is enabled for IPv4 packets. The Layer 3 matching is done only when the L3SAM0 or L3DAM0 bit is set.
L3SAM4	2	rw	**Layer 3 IP SA Match Enable** When this bit is set, the Layer 3 IP Source Address field is enabled for matching. When this bit is reset, the MAC ignores the Layer 3 IP Source Address field for matching. Note: When the L3PEN0 bit is set, you must set either this bit or the L3DAM0 bit because either IPv6 SA or DA can be checked for filtering.
L3SAIM4	3	rw	**Layer 3 IP SA Inverse Match Enable** When this bit is set, the Layer 3 IP Source Address field is enabled for inverse matching. When this bit reset, the Layer 3 IP Source Address field is enabled for perfect matching. This bit is valid and applicable only when the L3SAM0 bit is set.
L3DAM4	4	rw	**Layer 3 IP DA Match Enable** When this bit is set, the Layer 3 IP Destination Address field is enabled for matching. When this bit is reset, the MAC ignores the Layer 3 IP Destination Address field for matching. Note: When the L3PEN0 bit is set, you must set either this bit or the L3SAM0 bit because either IPv6 DA or SA can be checked for filtering.
L3DAIM4	5	rw	**Layer 3 IP DA Inverse Match Enable** When this bit is set, the Layer 3 IP Destination Address field is enabled for inverse matching. When this bit is reset, the Layer 3 IP Destination Address field is enabled for perfect matching. This bit is valid and applicable only when the L3DAM0 bit is set high.

(table continues...)

(continued)

Field	Bits	Type	Description
L3HSBM4	10:6	rw	**Layer 3 IP SA Higher Bits Match** IPv4 Packets: This field contains the number of lower bits of IP Source Address that are masked for matching in the IPv4 packets. The following list describes the values of this field: - 0: No bits are masked. - 1: Lsb[0] is masked - 2: Two Lsbs [1:0] are masked - ... - 31: All bits except Msb are masked. IPv6 Packets: This field contains Bits[4:0] of L3HSBM0. These bits indicate the number of higher bits of IP Source or Destination Address matched in the IPv6 packets. This field is valid and applicable only when the L3DAM0 or L3SAM0 bit is set high.
L3HDBM4	15:11	rw	**Layer 3 IP DA Higher Bits Match** IPv4 Packets: This field contains the number of higher bits of IP Destination Address that are matched in the IPv4 packets. The following list describes the values of this field: - 0: No bits are masked. - 1: LSB[0] is masked - 2: Two Lsbs [1:0] are masked - ... - 31: All bits except Msb are masked. IPv6 Packets: Bits[12:11] of this field correspond to Bits[6:5] of L3HSBM0 which indicate the number of lower bits of IP Source or Destination Address that are masked in the IPv6 packets. The following list describes the concatenated values of the L3HDBM0[1:0] and L3HSBM0 bits: - 0: No bits are masked. - 1: LSB[0] is masked. - 2: Two Lsbs [1:0] are masked - ... - 127: All bits except Msb are masked. This field is valid and applicable only when the L3DAM0 or L3SAM0 bit is set.
L4PEN4	16	rw	**Layer 4 Protocol Enable** When this bit is set, the Source and Destination Port number fields of UDP packets are used for matching. When this bit is reset, the Source and Destination Port number fields of TCP packets are used for matching. The Layer 4 matching is done only when the L4SPM0 or L4DPM0 bit is set.

(table continues...)

(continued)

Field	Bits	Type	Description
L4SPM4	18	rw	**Layer 4 Source Port Match Enable** When this bit is set, the Layer 4 Source Port number field is enabled for matching. When this bit is reset, the MAC ignores the Layer 4 Source Port number field for matching.
L4SPIM4	19	rw	**Layer 4 Source Port Inverse Match Enable** When this bit is set, the Layer 4 Source Port number field is enabled for inverse matching. When this bit is reset, the Layer 4 Source Port number field is enabled for perfect matching. This bit is valid and applicable only when the L4SPM0 bit is set high.
L4DPM4	20	rw	**Layer 4 Destination Port Match Enable** When this bit is set, the Layer 4 Destination Port number field is enabled for matching. When this bit is reset, the MAC ignores the Layer 4 Destination Port number field for matching.
L4DPIM4	21	rw	**Layer 4 Destination Port Inverse Match Enable** When this bit is set, the Layer 4 Destination Port number field is enabled for inverse matching. When this bit is reset, the Layer 4 Destination Port number field is enabled for perfect matching. This bit is valid and applicable only when the L4DPM0 bit is set high.
DMCHN	26:24	rw	**DMA Channel Number** When DMCHEN is set high, this field selects the DMA Channel number to which the packet passed by this filter is routed. The width of this field depends on the number of the DMA channels present in your configuration.
DMCHEN	31	rw	**DMA Channel Select Enable** When set, this bit enables the selection of the DMA channel number for the packet that is passed by this L3_L4 filter. The DMA channel is indicated by the DMCHN bits. When the bit is reset, the DMA channel is not decided by the filter.
0	1, 17, 22, 23, 29:27, 30	r	**Reserved** Read as 0; should be written with 0.

14.6.426 Port j MAC L3 L4 Control5

The Layer 3 and Layer 4 Control register controls the operations of filter 0 of Layer 3 and Layer 4.

Portj_MAC_L3_L4_Control5 (j=0-1) Offset address: $10C04_H+j*2000_H$

Port j MAC L3 L4 Control5 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMCHEN	0	0			DMCHN			0	0	L4DPIM5	L4DPM5	L4SPIM5	L4SPM5	0	L4PEN5
rw	r	r			rw			r	r	rw	rw	rw	rw	r	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
L3HDBM5					L3HSBM5					L3DAIM5	L3DAM5	L3SAIM5	L3SAM5	0	L3PEN5
rw					rw					rw	rw	rw	rw	r	rw

Field	Bits	Type	Description
L3PEN5	0	rw	**Layer 3 Protocol Enable** When this bit is set, the Layer 3 IP Source or Destination Address matching is enabled for IPv6 packets. When this bit is reset, the Layer 3 IP Source or Destination Address matching is enabled for IPv4 packets. The Layer 3 matching is done only when the L3SAM0 or L3DAM0 bit is set.
L3SAM5	2	rw	**Layer 3 IP SA Match Enable** When this bit is set, the Layer 3 IP Source Address field is enabled for matching. When this bit is reset, the MAC ignores the Layer 3 IP Source Address field for matching. Note: When the L3PEN0 bit is set, you must set either this bit or the L3DAM0 bit because either IPv6 SA or DA can be checked for filtering.
L3SAIM5	3	rw	**Layer 3 IP SA Inverse Match Enable** When this bit is set, the Layer 3 IP Source Address field is enabled for inverse matching. When this bit reset, the Layer 3 IP Source Address field is enabled for perfect matching. This bit is valid and applicable only when the L3SAM0 bit is set.
L3DAM5	4	rw	**Layer 3 IP DA Match Enable** When this bit is set, the Layer 3 IP Destination Address field is enabled for matching. When this bit is reset, the MAC ignores the Layer 3 IP Destination Address field for matching. Note: When the L3PEN0 bit is set, you must set either this bit or the L3SAM0 bit because either IPv6 DA or SA can be checked for filtering.
L3DAIM5	5	rw	**Layer 3 IP DA Inverse Match Enable** When this bit is set, the Layer 3 IP Destination Address field is enabled for inverse matching. When this bit is reset, the Layer 3 IP Destination Address field is enabled for perfect matching. This bit is valid and applicable only when the L3DAM0 bit is set high.

(table continues...)

(continued)

Field	Bits	Type	Description
L3HSBM5	10:6	rw	**Layer 3 IP SA Higher Bits Match** IPv4 Packets: This field contains the number of lower bits of IP Source Address that are masked for matching in the IPv4 packets. The following list describes the values of this field: - 0: No bits are masked. - 1: Lsb[0] is masked - 2: Two Lsbs [1:0] are masked - ... - 31: All bits except Msb are masked. IPv6 Packets: This field contains Bits[4:0] of L3HSBM0. These bits indicate the number of higher bits of IP Source or Destination Address matched in the IPv6 packets. This field is valid and applicable only when the L3DAM0 or L3SAM0 bit is set high.
L3HDBM5	15:11	rw	**Layer 3 IP DA Higher Bits Match** IPv4 Packets: This field contains the number of higher bits of IP Destination Address that are matched in the IPv4 packets. The following list describes the values of this field: - 0: No bits are masked. - 1: LSB[0] is masked - 2: Two Lsbs [1:0] are masked - ... - 31: All bits except Msb are masked. IPv6 Packets: Bits[12:11] of this field correspond to Bits[6:5] of L3HSBM0 which indicate the number of lower bits of IP Source or Destination Address that are masked in the IPv6 packets. The following list describes the concatenated values of the L3HDBM0[1:0] and L3HSBM0 bits: - 0: No bits are masked. - 1: LSB[0] is masked. - 2: Two Lsbs [1:0] are masked - ... - 127: All bits except Msb are masked. This field is valid and applicable only when the L3DAM0 or L3SAM0 bit is set.
L4PEN5	16	rw	**Layer 4 Protocol Enable** When this bit is set, the Source and Destination Port number fields of UDP packets are used for matching. When this bit is reset, the Source and Destination Port number fields of TCP packets are used for matching. The Layer 4 matching is done only when the L4SPM0 or L4DPM0 bit is set.

(table continues...)

(continued)

Field	Bits	Type	Description
L4SPM5	18	rw	**Layer 4 Source Port Match Enable** When this bit is set, the Layer 4 Source Port number field is enabled for matching. When this bit is reset, the MAC ignores the Layer 4 Source Port number field for matching.
L4SPIM5	19	rw	**Layer 4 Source Port Inverse Match Enable** When this bit is set, the Layer 4 Source Port number field is enabled for inverse matching. When this bit is reset, the Layer 4 Source Port number field is enabled for perfect matching. This bit is valid and applicable only when the L4SPM0 bit is set high.
L4DPM5	20	rw	**Layer 4 Destination Port Match Enable** When this bit is set, the Layer 4 Destination Port number field is enabled for matching. When this bit is reset, the MAC ignores the Layer 4 Destination Port number field for matching.
L4DPIM5	21	rw	**Layer 4 Destination Port Inverse Match Enable** When this bit is set, the Layer 4 Destination Port number field is enabled for inverse matching. When this bit is reset, the Layer 4 Destination Port number field is enabled for perfect matching. This bit is valid and applicable only when the L4DPM0 bit is set high.
DMCHN	26:24	rw	**DMA Channel Number** When DMCHEN is set high, this field selects the DMA Channel number to which the packet passed by this filter is routed. The width of this field depends on the number of the DMA channels present in your configuration.
DMCHEN	31	rw	**DMA Channel Select Enable** When set, this bit enables the selection of the DMA channel number for the packet that is passed by this L3_L4 filter. The DMA channel is indicated by the DMCHN bits. When the bit is reset, the DMA channel is not decided by the filter.
0	1, 17, 22, 23, 29:27, 30	r	**Reserved** Read as 0; should be written with 0.

14.6.427 Port j MAC L3 L4 Control6

The Layer 3 and Layer 4 Control register controls the operations of filter 0 of Layer 3 and Layer 4.

Portj_MAC_L3_L4_Control6 (j=0-1) Offset address: $10C04_H+j*2000_H$

Port j MAC L3 L4 Control6 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMCHEN	0	0			DMCHN			0	0	L4DPIM6	L4DPM6	L4SPIM6	L4SPM6	0	L4PEN6
rw	r	r			rw			r	r	rw	rw	rw	rw	r	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
L3HDBM6					L3HSBM6					L3DAIM6	L3DAM6	L3SAIM6	L3SAM6	0	L3PEN6
rw					rw					rw	rw	rw	rw	r	rw

Field	Bits	Type	Description
L3PEN6	0	rw	**Layer 3 Protocol Enable** When this bit is set, the Layer 3 IP Source or Destination Address matching is enabled for IPv6 packets. When this bit is reset, the Layer 3 IP Source or Destination Address matching is enabled for IPv4 packets. The Layer 3 matching is done only when the L3SAM0 or L3DAM0 bit is set.
L3SAM6	2	rw	**Layer 3 IP SA Match Enable** When this bit is set, the Layer 3 IP Source Address field is enabled for matching. When this bit is reset, the MAC ignores the Layer 3 IP Source Address field for matching. Note: When the L3PEN0 bit is set, you must set either this bit or the L3DAM0 bit because either IPv6 SA or DA can be checked for filtering.
L3SAIM6	3	rw	**Layer 3 IP SA Inverse Match Enable** When this bit is set, the Layer 3 IP Source Address field is enabled for inverse matching. When this bit reset, the Layer 3 IP Source Address field is enabled for perfect matching. This bit is valid and applicable only when the L3SAM0 bit is set.
L3DAM6	4	rw	**Layer 3 IP DA Match Enable** When this bit is set, the Layer 3 IP Destination Address field is enabled for matching. When this bit is reset, the MAC ignores the Layer 3 IP Destination Address field for matching. Note: When the L3PEN0 bit is set, you must set either this bit or the L3SAM0 bit because either IPv6 DA or SA can be checked for filtering.
L3DAIM6	5	rw	**Layer 3 IP DA Inverse Match Enable** When this bit is set, the Layer 3 IP Destination Address field is enabled for inverse matching. When this bit is reset, the Layer 3 IP Destination Address field is enabled for perfect matching. This bit is valid and applicable only when the L3DAM0 bit is set high.

(table continues...)

(continued)

Field	Bits	Type	Description
L3HSBM6	10:6	rw	**Layer 3 IP SA Higher Bits Match** IPv4 Packets: This field contains the number of lower bits of IP Source Address that are masked for matching in the IPv4 packets. The following list describes the values of this field: - 0: No bits are masked. - 1: Lsb[0] is masked - 2: Two Lsbs [1:0] are masked - ... - 31: All bits except Msb are masked. IPv6 Packets: This field contains Bits[4:0] of L3HSBM0. These bits indicate the number of higher bits of IP Source or Destination Address matched in the IPv6 packets. This field is valid and applicable only when the L3DAM0 or L3SAM0 bit is set high.
L3HDBM6	15:11	rw	**Layer 3 IP DA Higher Bits Match** IPv4 Packets: This field contains the number of higher bits of IP Destination Address that are matched in the IPv4 packets. The following list describes the values of this field: - 0: No bits are masked. - 1: LSB[0] is masked - 2: Two Lsbs [1:0] are masked - ... - 31: All bits except Msb are masked. IPv6 Packets: Bits[12:11] of this field correspond to Bits[6:5] of L3HSBM0 which indicate the number of lower bits of IP Source or Destination Address that are masked in the IPv6 packets. The following list describes the concatenated values of the L3HDBM0[1:0] and L3HSBM0 bits: - 0: No bits are masked. - 1: LSB[0] is masked. - 2: Two Lsbs [1:0] are masked - ... - 127: All bits except Msb are masked. This field is valid and applicable only when the L3DAM0 or L3SAM0 bit is set.
L4PEN6	16	rw	**Layer 4 Protocol Enable** When this bit is set, the Source and Destination Port number fields of UDP packets are used for matching. When this bit is reset, the Source and Destination Port number fields of TCP packets are used for matching. The Layer 4 matching is done only when the L4SPM0 or L4DPM0 bit is set.

(table continues...)

(continued)

Field	Bits	Type	Description
L4SPM6	18	rw	**Layer 4 Source Port Match Enable** When this bit is set, the Layer 4 Source Port number field is enabled for matching. When this bit is reset, the MAC ignores the Layer 4 Source Port number field for matching.
L4SPIM6	19	rw	**Layer 4 Source Port Inverse Match Enable** When this bit is set, the Layer 4 Source Port number field is enabled for inverse matching. When this bit is reset, the Layer 4 Source Port number field is enabled for perfect matching. This bit is valid and applicable only when the L4SPM0 bit is set high.
L4DPM6	20	rw	**Layer 4 Destination Port Match Enable** When this bit is set, the Layer 4 Destination Port number field is enabled for matching. When this bit is reset, the MAC ignores the Layer 4 Destination Port number field for matching.
L4DPIM6	21	rw	**Layer 4 Destination Port Inverse Match Enable** When this bit is set, the Layer 4 Destination Port number field is enabled for inverse matching. When this bit is reset, the Layer 4 Destination Port number field is enabled for perfect matching. This bit is valid and applicable only when the L4DPM0 bit is set high.
DMCHN	26:24	rw	**DMA Channel Number** When DMCHEN is set high, this field selects the DMA Channel number to which the packet passed by this filter is routed. The width of this field depends on the number of the DMA channels present in your configuration.
DMCHEN	31	rw	**DMA Channel Select Enable** When set, this bit enables the selection of the DMA channel number for the packet that is passed by this L3_L4 filter. The DMA channel is indicated by the DMCHN bits. When the bit is reset, the DMA channel is not decided by the filter.
0	1, 17, 22, 23, 29:27, 30	r	**Reserved** Read as 0; should be written with 0.

14.6.428 Port j MAC L3 L4 Control7

The Layer 3 and Layer 4 Control register controls the operations of filter 0 of Layer 3 and Layer 4.

Portj_MAC_L3_L4_Control7 (j=0-1) Offset address: $10C04_H+j*2000_H$

Port j MAC L3 L4 Control7 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMCHEN	0	0			DMCHN			0	0	L4DPIM7	L4DPM7	L4SPIM7	L4SPM7	0	L4PEN7
rw	r	r			rw			r	r	rw	rw	rw	rw	r	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
L3HDBM7					L3HSBM7					L3DAIM7	L3DAM7	L3SAIM7	L3SAM7	0	L3PEN7
rw					rw					rw	rw	rw	rw	r	rw

Field	Bits	Type	Description
L3PEN7	0	rw	**Layer 3 Protocol Enable** When this bit is set, the Layer 3 IP Source or Destination Address matching is enabled for IPv6 packets. When this bit is reset, the Layer 3 IP Source or Destination Address matching is enabled for IPv4 packets. The Layer 3 matching is done only when the L3SAM0 or L3DAM0 bit is set.
L3SAM7	2	rw	**Layer 3 IP SA Match Enable** When this bit is set, the Layer 3 IP Source Address field is enabled for matching. When this bit is reset, the MAC ignores the Layer 3 IP Source Address field for matching. Note: When the L3PEN0 bit is set, you must set either this bit or the L3DAM0 bit because either IPv6 SA or DA can be checked for filtering.
L3SAIM7	3	rw	**Layer 3 IP SA Inverse Match Enable** When this bit is set, the Layer 3 IP Source Address field is enabled for inverse matching. When this bit reset, the Layer 3 IP Source Address field is enabled for perfect matching. This bit is valid and applicable only when the L3SAM0 bit is set.
L3DAM7	4	rw	**Layer 3 IP DA Match Enable** When this bit is set, the Layer 3 IP Destination Address field is enabled for matching. When this bit is reset, the MAC ignores the Layer 3 IP Destination Address field for matching. Note: When the L3PEN0 bit is set, you must set either this bit or the L3SAM0 bit because either IPv6 DA or SA can be checked for filtering.
L3DAIM7	5	rw	**Layer 3 IP DA Inverse Match Enable** When this bit is set, the Layer 3 IP Destination Address field is enabled for inverse matching. When this bit is reset, the Layer 3 IP Destination Address field is enabled for perfect matching. This bit is valid and applicable only when the L3DAM0 bit is set high.

(table continues...)

(continued)

Field	Bits	Type	Description
L3HSBM7	10:6	rw	**Layer 3 IP SA Higher Bits Match** IPv4 Packets: This field contains the number of lower bits of IP Source Address that are masked for matching in the IPv4 packets. The following list describes the values of this field: - 0: No bits are masked. - 1: Lsb[0] is masked - 2: Two Lsbs [1:0] are masked - ... - 31: All bits except Msb are masked. IPv6 Packets: This field contains Bits[4:0] of L3HSBM0. These bits indicate the number of higher bits of IP Source or Destination Address matched in the IPv6 packets. This field is valid and applicable only when the L3DAM0 or L3SAM0 bit is set high.
L3HDBM7	15:11	rw	**Layer 3 IP DA Higher Bits Match** IPv4 Packets: This field contains the number of higher bits of IP Destination Address that are matched in the IPv4 packets. The following list describes the values of this field: - 0: No bits are masked. - 1: LSB[0] is masked - 2: Two Lsbs [1:0] are masked - ... - 31: All bits except Msb are masked. IPv6 Packets: Bits[12:11] of this field correspond to Bits[6:5] of L3HSBM0 which indicate the number of lower bits of IP Source or Destination Address that are masked in the IPv6 packets. The following list describes the concatenated values of the L3HDBM0[1:0] and L3HSBM0 bits: - 0: No bits are masked. - 1: LSB[0] is masked. - 2: Two Lsbs [1:0] are masked - ... - 127: All bits except Msb are masked. This field is valid and applicable only when the L3DAM0 or L3SAM0 bit is set.
L4PEN7	16	rw	**Layer 4 Protocol Enable** When this bit is set, the Source and Destination Port number fields of UDP packets are used for matching. When this bit is reset, the Source and Destination Port number fields of TCP packets are used for matching. The Layer 4 matching is done only when the L4SPM0 or L4DPM0 bit is set.

(table continues...)

(continued)

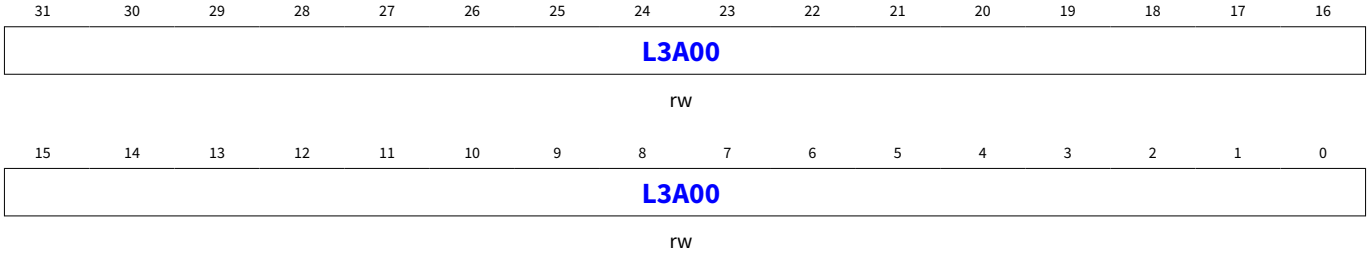
Field	Bits	Type	Description
L4SPM7	18	rw	**Layer 4 Source Port Match Enable** When this bit is set, the Layer 4 Source Port number field is enabled for matching. When this bit is reset, the MAC ignores the Layer 4 Source Port number field for matching.
L4SPIM7	19	rw	**Layer 4 Source Port Inverse Match Enable** When this bit is set, the Layer 4 Source Port number field is enabled for inverse matching. When this bit is reset, the Layer 4 Source Port number field is enabled for perfect matching. This bit is valid and applicable only when the L4SPM0 bit is set high.
L4DPM7	20	rw	**Layer 4 Destination Port Match Enable** When this bit is set, the Layer 4 Destination Port number field is enabled for matching. When this bit is reset, the MAC ignores the Layer 4 Destination Port number field for matching.
L4DPIM7	21	rw	**Layer 4 Destination Port Inverse Match Enable** When this bit is set, the Layer 4 Destination Port number field is enabled for inverse matching. When this bit is reset, the Layer 4 Destination Port number field is enabled for perfect matching. This bit is valid and applicable only when the L4DPM0 bit is set high.
DMCHN	26:24	rw	**DMA Channel Number** When DMCHEN is set high, this field selects the DMA Channel number to which the packet passed by this filter is routed. The width of this field depends on the number of the DMA channels present in your configuration.
DMCHEN	31	rw	**DMA Channel Select Enable** When set, this bit enables the selection of the DMA channel number for the packet that is passed by this L3_L4 filter. The DMA channel is indicated by the DMCHN bits. When the bit is reset, the DMA channel is not decided by the filter.
0	1, 17, 22, 23, 29:27, 30	r	**Reserved** Read as 0; should be written with 0.

14.6.429 Port j MAC Layer3 Addr0 Reg0

For IPv4 packets, the Layer 3 Address 0 Register 0 register contains the 32-bit IP Source Address field. For IPv6 packets, it contains Bits[31:0] of the 128-bit IP Source Address or Destination Address field.

Portj_MAC_Layer3_Addr0_Reg0 (j=0-1) Offset address: $10C04_H+j*2000_H$

Port j MAC Layer3 Addr0 Reg0 Kernel Reset value: $0000\ 0000_H$



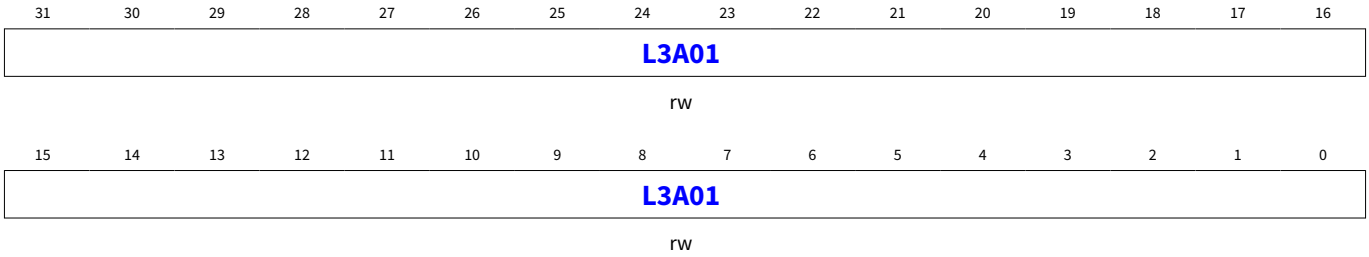
Field	Bits	Type	Description
L3A00	31:0	rw	**Layer 3 Address 0 Field** When the L3PEN0 and L3SAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[31:0] of the IP Source Address field in the IPv6 packets. When the L3PEN0 and L3DAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[31:0] of the IP Destination Address field in the IPv6 packets. When the L3PEN0 bit is reset and the L3SAM0 bit is set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with the IP Source Address field in the IPv4 packets.

14.6.430 Port j MAC Layer3 Addr0 Reg1

For IPv4 packets, the Layer 3 Address 0 Register 1 register contains the 32-bit IP Source Address field. For IPv6 packets, it contains Bits[31:0] of the 128-bit IP Source Address or Destination Address field.

Portj_MAC_Layer3_Addr0_Reg1 (j=0-1) Offset address: $10C04_H+j*2000_H$

Port j MAC Layer3 Addr0 Reg1 Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
L3A01	31:0	rw	**Layer 3 Address 0 Field** When the L3PEN0 and L3SAM0 bits are set in the MAC_L3_L4_Control1 register, this field contains the value to be matched with Bits[31:0] of the IP Source Address field in the IPv6 packets. When the L3PEN0 and L3DAM0 bits are set in the MAC_L3_L4_Control1 register, this field contains the value to be matched with Bits[31:0] of the IP Destination Address field in the IPv6 packets. When the L3PEN0 bit is reset and the L3SAM0 bit is set in the MAC_L3_L4_Control1 register, this field contains the value to be matched with the IP Source Address field in the IPv4 packets.

14.6.431 Port j MAC Layer3 Addr0 Reg2

For IPv4 packets, the Layer 3 Address 0 Register 2 register contains the 32-bit IP Source Address field. For IPv6 packets, it contains Bits[31:0] of the 128-bit IP Source Address or Destination Address field.

Portj_MAC_Layer3_Addr0_Reg2 (j=0-1) Offset address: $10C04_H+j*2000_H$

Port j MAC Layer3 Addr0 Reg2 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
L3A02															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
L3A02															
rw															

Field	Bits	Type	Description
L3A02	31:0	rw	**Layer 3 Address 0 Field** When the L3PEN0 and L3SAM0 bits are set in the MAC_L3_L4_Control2 register, this field contains the value to be matched with Bits[31:0] of the IP Source Address field in the IPv6 packets. When the L3PEN0 and L3DAM0 bits are set in the MAC_L3_L4_Control2 register, this field contains the value to be matched with Bits[31:0] of the IP Destination Address field in the IPv6 packets. When the L3PEN0 bit is reset and the L3SAM0 bit is set in the MAC_L3_L4_Control2 register, this field contains the value to be matched with the IP Source Address field in the IPv4 packets.

14.6.432 Port j MAC Layer3 Addr0 Reg3

For IPv4 packets, the Layer 3 Address 0 Register 3 register contains the 32-bit IP Source Address field. For IPv6 packets, it contains Bits[31:0] of the 128-bit IP Source Address or Destination Address field.

Portj_MAC_Layer3_Addr0_Reg3 (j=0-1) Offset address: $10C04_H+j*2000_H$

Port j MAC Layer3 Addr0 Reg3 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
L3A03															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
L3A03															
rw															

Field	Bits	Type	Description
L3A03	31:0	rw	**Layer 3 Address 0 Field** When the L3PEN0 and L3SAM0 bits are set in the MAC_L3_L4_Control3 register, this field contains the value to be matched with Bits[31:0] of the IP Source Address field in the IPv6 packets. When the L3PEN0 and L3DAM0 bits are set in the MAC_L3_L4_Control3 register, this field contains the value to be matched with Bits[31:0] of the IP Destination Address field in the IPv6 packets. When the L3PEN0 bit is reset and the L3SAM0 bit is set in the MAC_L3_L4_Control3 register, this field contains the value to be matched with the IP Source Address field in the IPv4 packets.

14.6.433 Port j MAC Layer3 Addr0 Reg4

For IPv4 packets, the Layer 3 Address 0 Register 4 register contains the 32-bit IP Source Address field. For IPv6 packets, it contains Bits[31:0] of the 128-bit IP Source Address or Destination Address field.

Portj_MAC_Layer3_Addr0_Reg4 (j=0-1) Offset address: $10C04_H+j*2000_H$

Port j MAC Layer3 Addr0 Reg4 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
L3A04															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
L3A04															
rw															

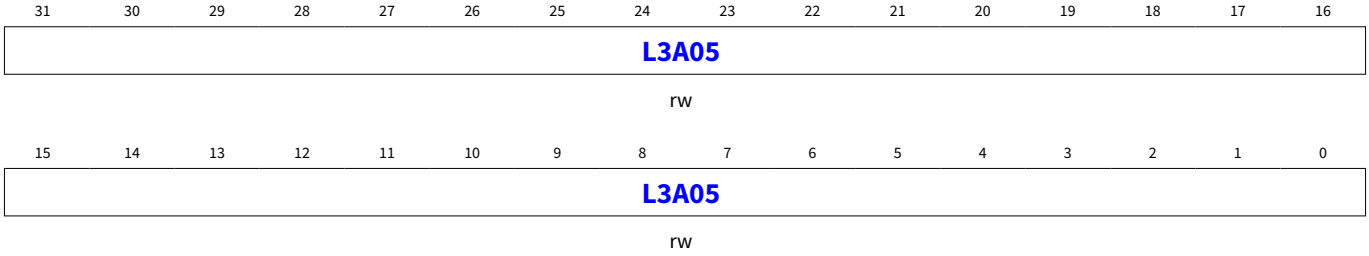
Field	Bits	Type	Description
L3A04	31:0	rw	**Layer 3 Address 0 Field** When the L3PEN0 and L3SAM0 bits are set in the MAC_L3_L4_Control4 register, this field contains the value to be matched with Bits[31:0] of the IP Source Address field in the IPv6 packets. When the L3PEN0 and L3DAM0 bits are set in the MAC_L3_L4_Control4 register, this field contains the value to be matched with Bits[31:0] of the IP Destination Address field in the IPv6 packets. When the L3PEN0 bit is reset and the L3SAM0 bit is set in the MAC_L3_L4_Control4 register, this field contains the value to be matched with the IP Source Address field in the IPv4 packets.

14.6.434 Port j MAC Layer3 Addr0 Reg5

For IPv4 packets, the Layer 3 Address 0 Register 5 register contains the 32-bit IP Source Address field. For IPv6 packets, it contains Bits[31:0] of the 128-bit IP Source Address or Destination Address field.

Portj_MAC_Layer3_Addr0_Reg5 (j=0-1) Offset address: $10C04_H+j*2000_H$

Port j MAC Layer3 Addr0 Reg5 Kernel Reset value: $0000\ 0000_H$



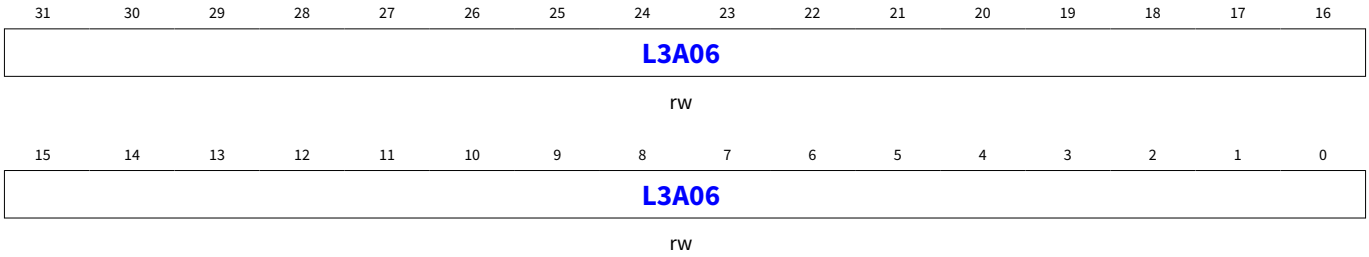
Field	Bits	Type	Description
L3A05	31:0	rw	**Layer 3 Address 0 Field** When the L3PEN0 and L3SAM0 bits are set in the MAC_L3_L4_Control5 register, this field contains the value to be matched with Bits[31:0] of the IP Source Address field in the IPv6 packets. When the L3PEN0 and L3DAM0 bits are set in the MAC_L3_L4_Control5 register, this field contains the value to be matched with Bits[31:0] of the IP Destination Address field in the IPv6 packets. When the L3PEN0 bit is reset and the L3SAM0 bit is set in the MAC_L3_L4_Control5 register, this field contains the value to be matched with the IP Source Address field in the IPv4 packets.

14.6.435 Port j MAC Layer3 Addr0 Reg6

For IPv4 packets, the Layer 3 Address 0 Register 6 register contains the 32-bit IP Source Address field. For IPv6 packets, it contains Bits[31:0] of the 128-bit IP Source Address or Destination Address field.

Portj_MAC_Layer3_Addr0_Reg6 (j=0-1) Offset address: $10C04_H+j*2000_H$

Port j MAC Layer3 Addr0 Reg6 Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
L3A06	31:0	rw	**Layer 3 Address 0 Field** When the L3PEN0 and L3SAM0 bits are set in the MAC_L3_L4_Control6 register, this field contains the value to be matched with Bits[31:0] of the IP Source Address field in the IPv6 packets. When the L3PEN0 and L3DAM0 bits are set in the MAC_L3_L4_Control6 register, this field contains the value to be matched with Bits[31:0] of the IP Destination Address field in the IPv6 packets. When the L3PEN0 bit is reset and the L3SAM0 bit is set in the MAC_L3_L4_Control6 register, this field contains the value to be matched with the IP Source Address field in the IPv4 packets.

14.6.436 Port j MAC Layer3 Addr0 Reg7

For IPv4 packets, the Layer 3 Address 0 Register 7 register contains the 32-bit IP Source Address field. For IPv6 packets, it contains Bits[31:0] of the 128-bit IP Source Address or Destination Address field.

Portj_MAC_Layer3_Addr0_Reg7 (j=0-1) Offset address: $10C04_H+j*2000_H$

Port j MAC Layer3 Addr0 Reg7 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
L3A07															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
L3A07															
rw															

Field	Bits	Type	Description
L3A07	31:0	rw	**Layer 3 Address 0 Field** When the L3PEN0 and L3SAM0 bits are set in the MAC_L3_L4_Control7 register, this field contains the value to be matched with Bits[31:0] of the IP Source Address field in the IPv6 packets. When the L3PEN0 and L3DAM0 bits are set in the MAC_L3_L4_Control7 register, this field contains the value to be matched with Bits[31:0] of the IP Destination Address field in the IPv6 packets. When the L3PEN0 bit is reset and the L3SAM0 bit is set in the MAC_L3_L4_Control7 register, this field contains the value to be matched with the IP Source Address field in the IPv4 packets.

14.6.437 Port j MAC Layer3 Addr1 Reg0

For IPv4 packets, the Layer 3 Address 1 Register 0 register contains the 32-bit IP Destination Address field. For IPv6 packets, it contains Bits[63:32] of the 128-bit IP Source Address or Destination Address field.

Portj_MAC_Layer3_Addr1_Reg0 (j=0-1) Offset address: $10C04_H+j*2000_H$

Port j MAC Layer3 Addr1 Reg0 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
L3A10															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
L3A10															
rw															

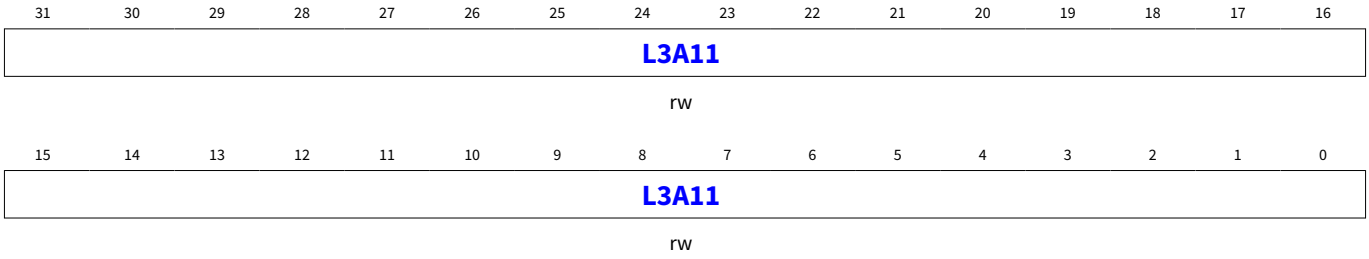
Field	Bits	Type	Description
L3A10	31:0	rw	**Layer 3 Address 1 Field** When the L3PEN0 and L3SAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[63:32] of the IP Source Address field in the IPv6 packets. When the L3PEN0 and L3DAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[63:32] of the IP Destination Address field in the IPv6 packets. When the L3PEN0 bit is reset and the L3DAM0 bit is set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with the IP Destination Address field in the IPv4 packets.

14.6.438 Port j MAC Layer3 Addr1 Reg1

For IPv4 packets, the Layer 3 Address 1 Register 1 register contains the 32-bit IP Destination Address field. For IPv6 packets, it contains Bits[63:32] of the 128-bit IP Source Address or Destination Address field.

Portj_MAC_Layer3_Addr1_Reg1 (j=0-1) Offset address: $10C04_H+j*2000_H$

Port j MAC Layer3 Addr1 Reg1 Kernel Reset value: $0000\ 0000_H$



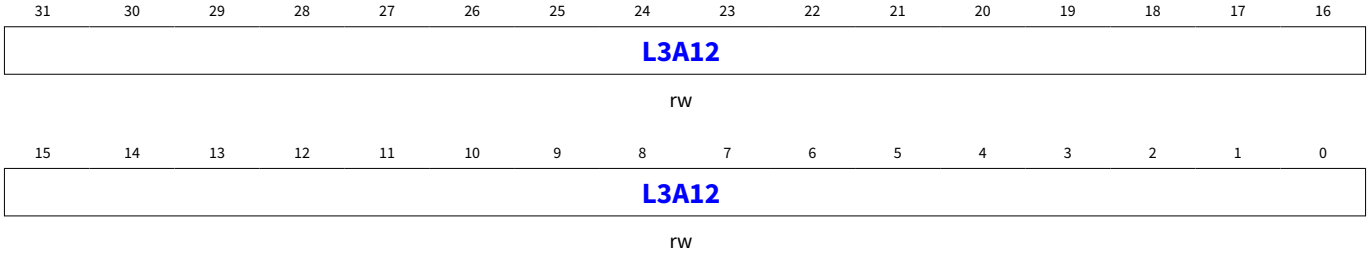
Field	Bits	Type	Description
L3A11	31:0	rw	**Layer 3 Address 1 Field** When the L3PEN0 and L3SAM0 bits are set in the MAC_L3_L4_Control1 register, this field contains the value to be matched with Bits[63:32] of the IP Source Address field in the IPv6 packets. When the L3PEN0 and L3DAM0 bits are set in the MAC_L3_L4_Control1 register, this field contains the value to be matched with Bits[63:32] of the IP Destination Address field in the IPv6 packets. When the L3PEN0 bit is reset and the L3DAM0 bit is set in the MAC_L3_L4_Control1 register, this field contains the value to be matched with the IP Destination Address field in the IPv4 packets.

14.6.439 Port j MAC Layer3 Addr1 Reg2

For IPv4 packets, the Layer 3 Address 1 Register 2 register contains the 32-bit IP Destination Address field. For IPv6 packets, it contains Bits[63:32] of the 128-bit IP Source Address or Destination Address field.

Portj_MAC_Layer3_Addr1_Reg2 (j=0-1) Offset address: $10C04_H+j*2000_H$

Port j MAC Layer3 Addr1 Reg2 Kernel Reset value: $0000\ 0000_H$



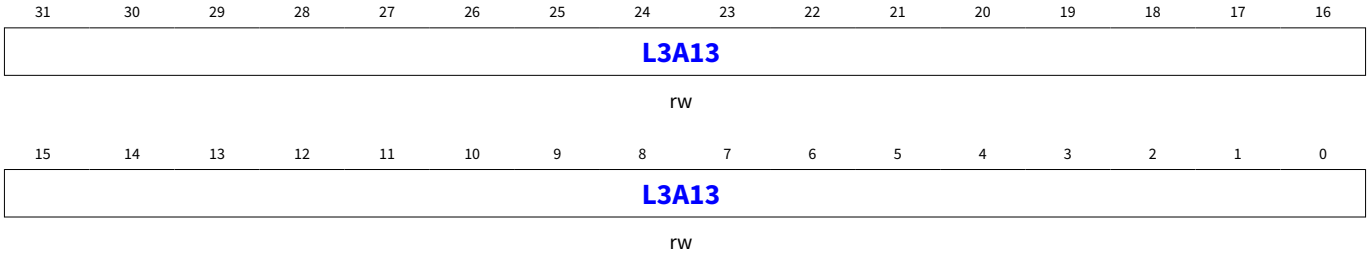
Field	Bits	Type	Description
L3A12	31:0	rw	**Layer 3 Address 1 Field** When the L3PEN0 and L3SAM0 bits are set in the MAC_L3_L4_Control2 register, this field contains the value to be matched with Bits[63:32] of the IP Source Address field in the IPv6 packets. When the L3PEN0 and L3DAM0 bits are set in the MAC_L3_L4_Control2 register, this field contains the value to be matched with Bits[63:32] of the IP Destination Address field in the IPv6 packets. When the L3PEN0 bit is reset and the L3DAM0 bit is set in the MAC_L3_L4_Control2 register, this field contains the value to be matched with the IP Destination Address field in the IPv4 packets.

14.6.440 Port j MAC Layer3 Addr1 Reg3

For IPv4 packets, the Layer 3 Address 1 Register 3 register contains the 32-bit IP Destination Address field. For IPv6 packets, it contains Bits[63:32] of the 128-bit IP Source Address or Destination Address field.

Portj_MAC_Layer3_Addr1_Reg3 (j=0-1) Offset address: $10C04_H+j*2000_H$

Port j MAC Layer3 Addr1 Reg3 Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
L3A13	31:0	rw	**Layer 3 Address 1 Field** When the L3PEN0 and L3SAM0 bits are set in the MAC_L3_L4_Control3 register, this field contains the value to be matched with Bits[63:32] of the IP Source Address field in the IPv6 packets. When the L3PEN0 and L3DAM0 bits are set in the MAC_L3_L4_Control3 register, this field contains the value to be matched with Bits[63:32] of the IP Destination Address field in the IPv6 packets. When the L3PEN0 bit is reset and the L3SAM0 bit is set in the MAC_L3_L4_Control3 register, this field contains the value to be matched with the IP Destination Address field in the IPv4 packets.

14.6.441 Port j MAC Layer3 Addr1 Reg4

For IPv4 packets, the Layer 3 Address 1 Register 4 register contains the 32-bit IP Destination Address field. For IPv6 packets, it contains Bits[63:32] of the 128-bit IP Source Address or Destination Address field.

Portj_MAC_Layer3_Addr1_Reg4 (j=0-1) Offset address: $10C04_H+j*2000_H$

Port j MAC Layer3 Addr1 Reg4 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
L3A14															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
L3A14															
rw															

Field	Bits	Type	Description
L3A14	31:0	rw	**Layer 3 Address 1 Field** When the L3PEN0 and L3SAM0 bits are set in the MAC_L3_L4_Control4 register, this field contains the value to be matched with Bits[63:32] of the IP Source Address field in the IPv6 packets. When the L3PEN0 and L3DAM0 bits are set in the MAC_L3_L4_Control4 register, this field contains the value to be matched with Bits[63:32] of the IP Destination Address field in the IPv6 packets. When the L3PEN0 bit is reset and the L3SAM0 bit is set in the MAC_L3_L4_Control4 register, this field contains the value to be matched with the IP Destination Address field in the IPv4 packets.

14.6.442 Port j MAC Layer3 Addr1 Reg5

For IPv4 packets, the Layer 3 Address 1 Register 5 register contains the 32-bit IP Destination Address field. For IPv6 packets, it contains Bits[63:32] of the 128-bit IP Source Address or Destination Address field.

Portj_MAC_Layer3_Addr1_Reg5 (j=0-1) Offset address: $10C04_H+j*2000_H$

Port j MAC Layer3 Addr1 Reg5 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
L3A15															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
L3A15															
rw															

Field	Bits	Type	Description
L3A15	31:0	rw	**Layer 3 Address 1 Field** When the L3PEN0 and L3SAM0 bits are set in the MAC_L3_L4_Control5 register, this field contains the value to be matched with Bits[63:32] of the IP Source Address field in the IPv6 packets. When the L3PEN0 and L3DAM0 bits are set in the MAC_L3_L4_Control5 register, this field contains the value to be matched with Bits[63:32] of the IP Destination Address field in the IPv6 packets. When the L3PEN0 bit is reset and the L3SAM0 bit is set in the MAC_L3_L4_Control5 register, this field contains the value to be matched with the IP Destination Address field in the IPv4 packets.

14.6.443 Port j MAC Layer3 Addr1 Reg6

For IPv4 packets, the Layer 3 Address 1 Register 6 register contains the 32-bit IP Destination Address field. For IPv6 packets, it contains Bits[63:32] of the 128-bit IP Source Address or Destination Address field.

Portj_MAC_Layer3_Addr1_Reg6 (j=0-1) Offset address: $10C04_H+j*2000_H$

Port j MAC Layer3 Addr1 Reg6 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
L3A16															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
L3A16															
rw															

Field	Bits	Type	Description
L3A16	31:0	rw	**Layer 3 Address 1 Field** When the L3PEN0 and L3SAM0 bits are set in the MAC_L3_L4_Control6 register, this field contains the value to be matched with Bits[63:32] of the IP Source Address field in the IPv6 packets. When the L3PEN0 and L3DAM0 bits are set in the MAC_L3_L4_Control6 register, this field contains the value to be matched with Bits[63:32] of the IP Destination Address field in the IPv6 packets. When the L3PEN0 bit is reset and the L3SAM0 bit is set in the MAC_L3_L4_Control6 register, this field contains the value to be matched with the IP Destination Address field in the IPv4 packets.

14.6.444 Port j MAC Layer3 Addr1 Reg7

For IPv4 packets, the Layer 3 Address 1 Register 7 register contains the 32-bit IP Destination Address field. For IPv6 packets, it contains Bits[63:32] of the 128-bit IP Source Address or Destination Address field.

Portj_MAC_Layer3_Addr1_Reg7 (j=0-1) Offset address: $10C04_H+j*2000_H$

Port j MAC Layer3 Addr1 Reg7 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
L3A17															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
L3A17															
rw															

Field	Bits	Type	Description
L3A17	31:0	rw	**Layer 3 Address 1 Field** When the L3PEN0 and L3SAM0 bits are set in the MAC_L3_L4_Control7 register, this field contains the value to be matched with Bits[63:32] of the IP Source Address field in the IPv6 packets. When the L3PEN0 and L3DAM0 bits are set in the MAC_L3_L4_Control7 register, this field contains the value to be matched with Bits[63:32] of the IP Destination Address field in the IPv6 packets. When the L3PEN0 bit is reset and the L3SAM0 bit is set in the MAC_L3_L4_Control7 register, this field contains the value to be matched with the IP Destination Address field in the IPv4 packets.

14.6.445 Port j MAC Layer3 Addr2 Reg0

The Layer 3 Address 2 Register 0 register is reserved for IPv4 packets. For IPv6 packets, it contains Bits[95:64] of 128-bit IP Source Address or Destination Address field.

Portj_MAC_Layer3_Addr2_Reg0 (j=0-1) Offset address: $10C04_H+j*2000_H$

Port j MAC Layer3 Addr2 Reg0 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
L3A20															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
L3A20															
rw															

Field	Bits	Type	Description
L3A20	31:0	rw	**Layer 3 Address 2 Field** When the L3PEN0 and L3SAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[95:64] of the IP Source Address field in the IPv6 packets. When the L3PEN0 and L3DAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[95:64] of the IP Destination Address field in the IPv6 packets. When the L3PEN0 bit is reset in the MAC_L3_L4_Control0 register, this field is not used.

14.6.446 Port j MAC Layer3 Addr2 Reg1

The Layer 3 Address 2 Register 1 register is reserved for IPv4 packets. For IPv6 packets, it contains Bits[95:64] of 128-bit IP Source Address or Destination Address field.

Portj_MAC_Layer3_Addr2_Reg1 (j=0-1) Offset address: $10C04_H+j*2000_H$

Port j MAC Layer3 Addr2 Reg1 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
L3A21															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
L3A21															
rw															

Field	Bits	Type	Description
L3A21	31:0	rw	**Layer 3 Address 2 Field** When the L3PEN0 and L3SAM0 bits are set in the MAC_L3_L4_Control1 register, this field contains the value to be matched with Bits[95:64] of the IP Source Address field in the IPv6 packets. When the L3PEN0 and L3DAM0 bits are set in the MAC_L3_L4_Control1 register, this field contains the value to be matched with Bits[95:64] of the IP Destination Address field in the IPv6 packets. When the L3PEN0 bit is reset in the MAC_L3_L4_Control1 register, this field is not used.

14.6.447 Port j MAC Layer3 Addr2 Reg2

The Layer 3 Address 2 Register 2 register is reserved for IPv4 packets. For IPv6 packets, it contains Bits[95:64] of 128-bit IP Source Address or Destination Address field.

Portj_MAC_Layer3_Addr2_Reg2 (j=0-1) Offset address: $10C04_H+j*2000_H$

Port j MAC Layer3 Addr2 Reg2 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
L3A22															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
L3A22															
rw															

Field	Bits	Type	Description
L3A22	31:0	rw	**Layer 3 Address 2 Field** When the L3PEN0 and L3SAM0 bits are set in the MAC_L3_L4_Control2 register, this field contains the value to be matched with Bits[95:64] of the IP Source Address field in the IPv6 packets. When the L3PEN0 and L3DAM0 bits are set in the MAC_L3_L4_Control2 register, this field contains the value to be matched with Bits[95:64] of the IP Destination Address field in the IPv6 packets. When the L3PEN0 bit is reset in the MAC_L3_L4_Control2 register, this field is not used.

14.6.448 Port j MAC Layer3 Addr2 Reg3

The Layer 3 Address 2 Register 3 register is reserved for IPv4 packets. For IPv6 packets, it contains Bits[95:64] of 128-bit IP Source Address or Destination Address field.

Portj_MAC_Layer3_Addr2_Reg3 (j=0-1) Offset address: $10C04_H+j*2000_H$

Port j MAC Layer3 Addr2 Reg3 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
L3A23															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
L3A23															
rw															

Field	Bits	Type	Description
L3A23	31:0	rw	**Layer 3 Address 2 Field** When the L3PEN0 and L3SAM0 bits are set in the MAC_L3_L4_Control3 register, this field contains the value to be matched with Bits[95:64] of the IP Source Address field in the IPv6 packets. When the L3PEN0 and L3DAM0 bits are set in the MAC_L3_L4_Control3 register, this field contains the value to be matched with Bits[95:64] of the IP Destination Address field in the IPv6 packets. When the L3PEN0 bit is reset in the MAC_L3_L4_Control3 register, this field is not used.

14.6.449 Port j MAC Layer3 Addr2 Reg4

The Layer 3 Address 2 Register 4 register is reserved for IPv4 packets. For IPv6 packets, it contains Bits[95:64] of 128-bit IP Source Address or Destination Address field.

Portj_MAC_Layer3_Addr2_Reg4 (j=0-1) Offset address: $10C04_H+j*2000_H$

Port j MAC Layer3 Addr2 Reg4 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
L3A24															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
L3A24															
rw															

Field	Bits	Type	Description
L3A24	31:0	rw	**Layer 3 Address 2 Field** When the L3PEN0 and L3SAM0 bits are set in the MAC_L3_L4_Control4 register, this field contains the value to be matched with Bits[95:64] of the IP Source Address field in the IPv6 packets. When the L3PEN0 and L3DAM0 bits are set in the MAC_L3_L4_Control4 register, this field contains the value to be matched with Bits[95:64] of the IP Destination Address field in the IPv6 packets. When the L3PEN0 bit is reset in the MAC_L3_L4_Control4 register, this field is not used.

14.6.450 Port j MAC Layer3 Addr2 Reg5

The Layer 3 Address 2 Register 5 register is reserved for IPv4 packets. For IPv6 packets, it contains Bits[95:64] of 128-bit IP Source Address or Destination Address field.

Portj_MAC_Layer3_Addr2_Reg5 (j=0-1) Offset address: $10C04_H+j*2000_H$

Port j MAC Layer3 Addr2 Reg5 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
L3A25															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
L3A25															
rw															

Field	Bits	Type	Description
L3A25	31:0	rw	**Layer 3 Address 2 Field** When the L3PEN0 and L3SAM0 bits are set in the MAC_L3_L4_Control5 register, this field contains the value to be matched with Bits[95:64] of the IP Source Address field in the IPv6 packets. When the L3PEN0 and L3DAM0 bits are set in the MAC_L3_L4_Control5 register, this field contains the value to be matched with Bits[95:64] of the IP Destination Address field in the IPv6 packets. When the L3PEN0 bit is reset in the MAC_L3_L4_Control5 register, this field is not used.

14.6.451 Port j MAC Layer3 Addr2 Reg6

The Layer 3 Address 2 Register 6 register is reserved for IPv4 packets. For IPv6 packets, it contains Bits[95:64] of 128-bit IP Source Address or Destination Address field.

Portj_MAC_Layer3_Addr2_Reg6 (j=0-1) Offset address: $10C04_H+j*2000_H$

Port j MAC Layer3 Addr2 Reg6 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
L3A26															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
L3A26															
rw															

Field	Bits	Type	Description
L3A26	31:0	rw	**Layer 3 Address 2 Field** When the L3PEN0 and L3SAM0 bits are set in the MAC_L3_L4_Control6 register, this field contains the value to be matched with Bits[95:64] of the IP Source Address field in the IPv6 packets. When the L3PEN0 and L3DAM0 bits are set in the MAC_L3_L4_Control6 register, this field contains the value to be matched with Bits[95:64] of the IP Destination Address field in the IPv6 packets. When the L3PEN0 bit is reset in the MAC_L3_L4_Control6 register, this field is not used.

14.6.452 Port j MAC Layer3 Addr2 Reg7

The Layer 3 Address 2 Register 7 register is reserved for IPv4 packets. For IPv6 packets, it contains Bits[95:64] of 128-bit IP Source Address or Destination Address field.

Portj_MAC_Layer3_Addr2_Reg7 (j=0-1) Offset address: $10C04_H+j*2000_H$

Port j MAC Layer3 Addr2 Reg7 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
L3A27															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
L3A27															
rw															

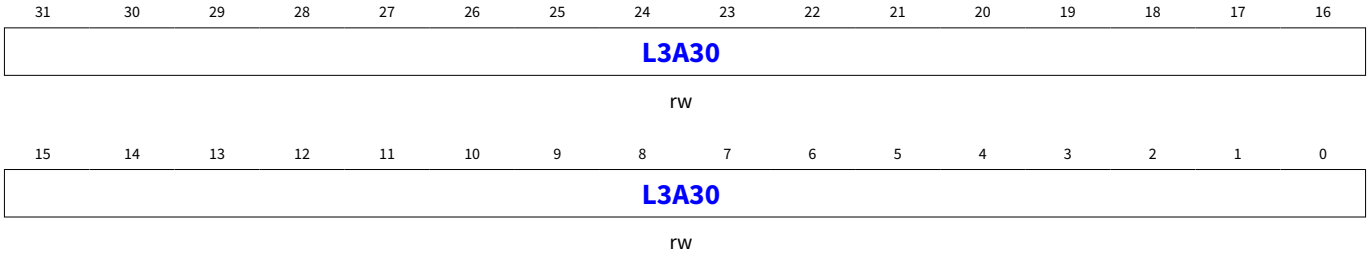
Field	Bits	Type	Description
L3A27	31:0	rw	**Layer 3 Address 2 Field** When the L3PEN0 and L3SAM0 bits are set in the MAC_L3_L4_Control7 register, this field contains the value to be matched with Bits[95:64] of the IP Source Address field in the IPv6 packets. When the L3PEN0 and L3DAM0 bits are set in the MAC_L3_L4_Control7 register, this field contains the value to be matched with Bits[95:64] of the IP Destination Address field in the IPv6 packets. When the L3PEN0 bit is reset in the MAC_L3_L4_Control7 register, this field is not used.

14.6.453 Port j MAC Layer3 Addr3 Reg0

The Layer 3 Address 3 Register 0 register is reserved for IPv4 packets. For IPv6 packets, it contains Bits[127:96] of 128-bit IP Source Address or Destination Address field.

Portj_MAC_Layer3_Addr3_Reg0 (j=0-1) Offset address: $10C04_H+j*2000_H$

Port j MAC Layer3 Addr3 Reg0 Kernel Reset value: $0000\ 0000_H$



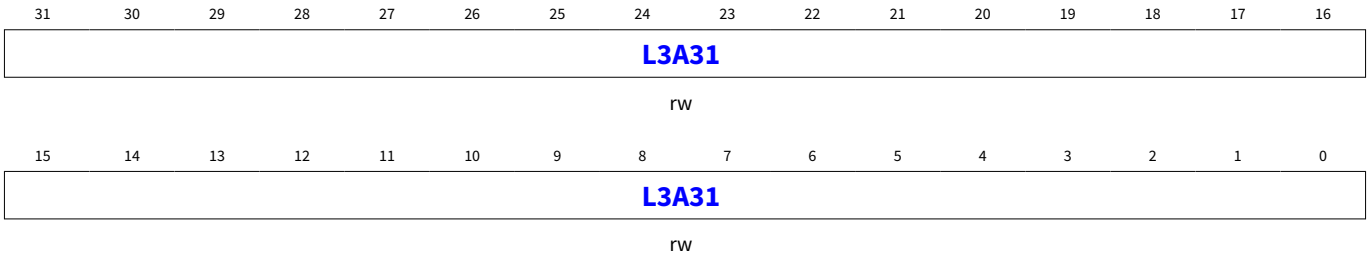
Field	Bits	Type	Description
L3A30	31:0	rw	**Layer 3 Address 3 Field** When the L3PEN0 and L3SAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[127:96] of the IP Source Address field in the IPv6 packets. When the L3PEN0 and L3DAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[127:96] of the IP Destination Address field in the IPv6 packets. When the L3PEN0 bit is reset in the MAC_L3_L4_Control0 register, this field is not used.

14.6.454 Port j MAC Layer3 Addr3 Reg1

The Layer 3 Address 3 Register 1 register is reserved for IPv4 packets. For IPv6 packets, it contains Bits[127:96] of 128-bit IP Source Address or Destination Address field.

Portj_MAC_Layer3_Addr3_Reg1 (j=0-1) Offset address: $10C04_H+j*2000_H$

Port j MAC Layer3 Addr3 Reg1 Kernel Reset value: $0000\ 0000_H$



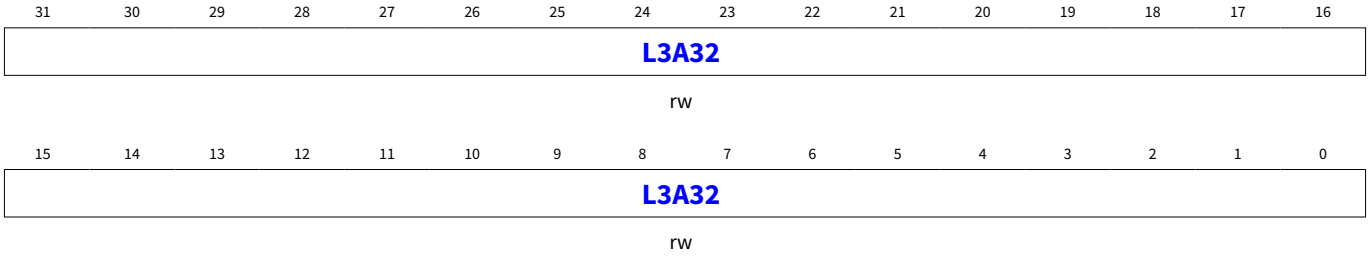
Field	Bits	Type	Description
L3A31	31:0	rw	**Layer 3 Address 3 Field** When the L3PEN0 and L3SAM0 bits are set in the MAC_L3_L4_Control1 register, this field contains the value to be matched with Bits[127:96] of the IP Source Address field in the IPv6 packets. When the L3PEN0 and L3DAM0 bits are set in the MAC_L3_L4_Control1 register, this field contains the value to be matched with Bits[127:96] of the IP Destination Address field in the IPv6 packets. When the L3PEN0 bit is reset in the MAC_L3_L4_Control1 register, this field is not used.

14.6.455 Port j MAC Layer3 Addr3 Reg2

The Layer 3 Address 3 Register 2 register is reserved for IPv4 packets. For IPv6 packets, it contains Bits[127:96] of 128-bit IP Source Address or Destination Address field.

Portj_MAC_Layer3_Addr3_Reg2 (j=0-1) Offset address: $10C04_H+j*2000_H$

Port j MAC Layer3 Addr3 Reg2 Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
L3A32	31:0	rw	**Layer 3 Address 3 Field** When the L3PEN0 and L3SAM0 bits are set in the MAC_L3_L4_Control2 register, this field contains the value to be matched with Bits[127:96] of the IP Source Address field in the IPv6 packets. When the L3PEN0 and L3DAM0 bits are set in the MAC_L3_L4_Control2 register, this field contains the value to be matched with Bits[127:96] of the IP Destination Address field in the IPv6 packets. When the L3PEN0 bit is reset in the MAC_L3_L4_Control2 register, this field is not used.

14.6.456 Port j MAC Layer3 Addr3 Reg3

The Layer 3 Address 3 Register 3 register is reserved for IPv4 packets. For IPv6 packets, it contains Bits[127:96] of 128-bit IP Source Address or Destination Address field.

Portj_MAC_Layer3_Addr3_Reg3 (j=0-1) Offset address: $10C04_H+j*2000_H$

Port j MAC Layer3 Addr3 Reg3 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
L3A33															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
L3A33															
rw															

Field	Bits	Type	Description
L3A33	31:0	rw	**Layer 3 Address 3 Field** When the L3PEN0 and L3SAM0 bits are set in the MAC_L3_L4_Control3 register, this field contains the value to be matched with Bits[127:96] of the IP Source Address field in the IPv6 packets. When the L3PEN0 and L3DAM0 bits are set in the MAC_L3_L4_Control3 register, this field contains the value to be matched with Bits[127:96] of the IP Destination Address field in the IPv6 packets. When the L3PEN0 bit is reset in the MAC_L3_L4_Control3 register, this field is not used.

14.6.457 Port j MAC Layer3 Addr3 Reg4

The Layer 3 Address 3 Register 4 register is reserved for IPv4 packets. For IPv6 packets, it contains Bits[127:96] of 128-bit IP Source Address or Destination Address field.

Portj_MAC_Layer3_Addr3_Reg4 (j=0-1) Offset address: $10C04_H+j*2000_H$

Port j MAC Layer3 Addr3 Reg4 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
L3A34															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
L3A34															
rw															

Field	Bits	Type	Description
L3A34	31:0	rw	**Layer 3 Address 3 Field** When the L3PEN0 and L3SAM0 bits are set in the MAC_L3_L4_Control4 register, this field contains the value to be matched with Bits[127:96] of the IP Source Address field in the IPv6 packets. When the L3PEN0 and L3DAM0 bits are set in the MAC_L3_L4_Control4 register, this field contains the value to be matched with Bits[127:96] of the IP Destination Address field in the IPv6 packets. When the L3PEN0 bit is reset in the MAC_L3_L4_Control4 register, this field is not used.

14.6.458 Port j MAC Layer3 Addr3 Reg5

The Layer 3 Address 3 Register 5 register is reserved for IPv4 packets. For IPv6 packets, it contains Bits[127:96] of 128-bit IP Source Address or Destination Address field.

Portj_MAC_Layer3_Addr3_Reg5 (j=0-1) Offset address: $10C04_H+j*2000_H$

Port j MAC Layer3 Addr3 Reg5 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
L3A35															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
L3A35															
rw															

Field	Bits	Type	Description
L3A35	31:0	rw	**Layer 3 Address 3 Field** When the L3PEN0 and L3SAM0 bits are set in the MAC_L3_L4_Control5 register, this field contains the value to be matched with Bits[127:96] of the IP Source Address field in the IPv6 packets. When the L3PEN0 and L3DAM0 bits are set in the MAC_L3_L4_Control5 register, this field contains the value to be matched with Bits[127:96] of the IP Destination Address field in the IPv6 packets. When the L3PEN0 bit is reset in the MAC_L3_L4_Control5 register, this field is not used.

14.6.459 Port j MAC Layer3 Addr3 Reg6

The Layer 3 Address 3 Register 6 register is reserved for IPv4 packets. For IPv6 packets, it contains Bits[127:96] of 128-bit IP Source Address or Destination Address field.

Portj_MAC_Layer3_Addr3_Reg6 (j=0-1) Offset address: $10C04_H+j*2000_H$

Port j MAC Layer3 Addr3 Reg6 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
L3A36															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
L3A36															
rw															

Field	Bits	Type	Description
L3A36	31:0	rw	**Layer 3 Address 3 Field** When the L3PEN0 and L3SAM0 bits are set in the MAC_L3_L4_Control6 register, this field contains the value to be matched with Bits[127:96] of the IP Source Address field in the IPv6 packets. When the L3PEN0 and L3DAM0 bits are set in the MAC_L3_L4_Control6 register, this field contains the value to be matched with Bits[127:96] of the IP Destination Address field in the IPv6 packets. When the L3PEN0 bit is reset in the MAC_L3_L4_Control6 register, this field is not used.

14.6.460 Port j MAC Layer3 Addr3 Reg7

The Layer 3 Address 3 Register 7 register is reserved for IPv4 packets. For IPv6 packets, it contains Bits[127:96] of 128-bit IP Source Address or Destination Address field.

Portj_MAC_Layer3_Addr3_Reg7 (j=0-1) Offset address: $10C04_H+j*2000_H$

Port j MAC Layer3 Addr3 Reg7 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
L3A37															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
L3A37															
rw															

Field	Bits	Type	Description
L3A37	31:0	rw	**Layer 3 Address 3 Field** When the L3PEN0 and L3SAM0 bits are set in the MAC_L3_L4_Control7 register, this field contains the value to be matched with Bits[127:96] of the IP Source Address field in the IPv6 packets. When the L3PEN0 and L3DAM0 bits are set in the MAC_L3_L4_Control7 register, this field contains the value to be matched with Bits[127:96] of the IP Destination Address field in the IPv6 packets. When the L3PEN0 bit is reset in the MAC_L3_L4_Control7 register, this field is not used.

14.6.461 Port j MAC Layer4 Address0

Layer 4 Address 0 register has the Layer 4 (TCP, UDP) Source and Destination Port fields checked against the received packet.

Portj_MAC_Layer4_Address0 (j=0-1) Offset address: $10C04_H+j*2000_H$

Port j MAC Layer4 Address0 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
L4DP0															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
L4SP0															
rw															

Field	Bits	Type	Description
L4SP0	15:0	rw	**Layer 4 Source Port Number Field** When the L4PEN0 bit is reset and the L4DPM0 bit is set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with the TCP Source Port Number field in the IPv4 or IPv6 packets. When the L4PEN0 and L4DPM0 bits are set in MAC_L3_L4_Control0 register, this field contains the value to be matched with the UDP Source Port Number field in the IPv4 or IPv6 packets.
L4DP0	31:16	rw	**Layer 4 Destination Port Number Field** When the L4PEN0 bit is reset and the L4DPM0 bit is set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with the TCP Destination Port Number field in the IPv4 or IPv6 packets. When the L4PEN0 and L4DPM0 bits are set in MAC_L3_L4_Control0 register, this field contains the value to be matched with the UDP Destination Port Number field in the IPv4 or IPv6 packets.

14.6.462 Port j MAC Layer4 Address1

Layer 4 Address 1 register has the Layer 4 (TCP, UDP) Source and Destination Port fields checked against the received packet.

Portj_MAC_Layer4_Address1 (j=0-1) Offset address: $10C04_H+j*2000_H$

Port j MAC Layer4 Address1 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
L4DP1															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
L4SP1															
rw															

Field	Bits	Type	Description
L4SP1	15:0	rw	**Layer 4 Source Port Number Field** When the L4PEN0 bit is reset and the L4DPM0 bit is set in the MAC_L3_L4_Control1 register, this field contains the value to be matched with the TCP Source Port Number field in the IPv4 or IPv6 packets. When the L4PEN0 and L4DPM0 bits are set in MAC_L3_L4_Control1 register, this field contains the value to be matched with the UDP Source Port Number field in the IPv4 or IPv6 packets.
L4DP1	31:16	rw	**Layer 4 Destination Port Number Field** When the L4PEN0 bit is reset and the L4DPM0 bit is set in the MAC_L3_L4_Control1 register, this field contains the value to be matched with the TCP Destination Port Number field in the IPv4 or IPv6 packets. When the L4PEN0 and L4DPM0 bits are set in MAC_L3_L4_Control1 register, this field contains the value to be matched with the UDP Destination Port Number field in the IPv4 or IPv6 packets.

14.6.463 Port j MAC Layer4 Address2

Layer 4 Address 2 register has the Layer 4 (TCP, UDP) Source and Destination Port fields checked against the received packet.

Portj_MAC_Layer4_Address2 (j=0-1) Offset address: $10C04_H+j*2000_H$

Port j MAC Layer4 Address2 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
L4DP2															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
L4SP2															
rw															

Field	Bits	Type	Description
L4SP2	15:0	rw	**Layer 4 Source Port Number Field** When the L4PEN0 bit is reset and the L4DPM0 bit is set in the MAC_L3_L4_Control2 register, this field contains the value to be matched with the TCP Source Port Number field in the IPv4 or IPv6 packets. When the L4PEN0 and L4DPM0 bits are set in MAC_L3_L4_Control2 register, this field contains the value to be matched with the UDP Source Port Number field in the IPv4 or IPv6 packets.

(table continues...)

(continued)

Field	Bits	Type	Description
L4DP2	31:16	rw	**Layer 4 Destination Port Number Field** When the L4PEN0 bit is reset and the L4DPM0 bit is set in the MAC_L3_L4_Control2 register, this field contains the value to be matched with the TCP Destination Port Number field in the IPv4 or IPv6 packets. When the L4PEN0 and L4DPM0 bits are set in MAC_L3_L4_Control2 register, this field contains the value to be matched with the UDP Destination Port Number field in the IPv4 or IPv6 packets.

14.6.464 Port j MAC Layer4 Address3

Layer 4 Address 3 register has the Layer 4 (TCP, UDP) Source and Destination Port fields checked against the received packet.

Portj_MAC_Layer4_Address3 (j=0-1) Offset address: $10C04_H+j*2000_H$

Port j MAC Layer4 Address3 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
L4DP3															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
L4SP3															
rw															

Field	Bits	Type	Description
L4SP3	15:0	rw	**Layer 4 Source Port Number Field** When the L4PEN0 bit is reset and the L4DPM0 bit is set in the MAC_L3_L4_Control3 register, this field contains the value to be matched with the TCP Source Port Number field in the IPv4 or IPv6 packets. When the L4PEN0 and L4DPM0 bits are set in MAC_L3_L4_Control3 register, this field contains the value to be matched with the UDP Source Port Number field in the IPv4 or IPv6 packets.
L4DP3	31:16	rw	**Layer 4 Destination Port Number Field** When the L4PEN0 bit is reset and the L4DPM0 bit is set in the MAC_L3_L4_Control3 register, this field contains the value to be matched with the TCP Destination Port Number field in the IPv4 or IPv6 packets. When the L4PEN0 and L4DPM0 bits are set in MAC_L3_L4_Control3 register, this field contains the value to be matched with the UDP Destination Port Number field in the IPv4 or IPv6 packets.

14.6.465 Port j MAC Layer4 Address4

Layer 4 Address 4 register has the Layer 4 (TCP, UDP) Source and Destination Port fields checked against the received packet.

Portj_MAC_Layer4_Address4 (j=0-1) Offset address: $10C04_H+j*2000_H$

Port j MAC Layer4 Address4 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
L4DP4															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
L4SP4															
rw															

Field	Bits	Type	Description
L4SP4	15:0	rw	**Layer 4 Source Port Number Field** When the L4PEN0 bit is reset and the L4DPM0 bit is set in the MAC_L3_L4_Control4 register, this field contains the value to be matched with the TCP Source Port Number field in the IPv4 or IPv6 packets. When the L4PEN0 and L4DPM0 bits are set in MAC_L3_L4_Control4 register, this field contains the value to be matched with the UDP Source Port Number field in the IPv4 or IPv6 packets.
L4DP4	31:16	rw	**Layer 4 Destination Port Number Field** When the L4PEN0 bit is reset and the L4DPM0 bit is set in the MAC_L3_L4_Control4 register, this field contains the value to be matched with the TCP Destination Port Number field in the IPv4 or IPv6 packets. When the L4PEN0 and L4DPM0 bits are set in MAC_L3_L4_Control4 register, this field contains the value to be matched with the UDP Destination Port Number field in the IPv4 or IPv6 packets.

14.6.466 Port j MAC Layer4 Address5

Layer 4 Address 5 register has the Layer 4 (TCP, UDP) Source and Destination Port fields checked against the received packet.

Portj_MAC_Layer4_Address5 (j=0-1) Offset address: $10C04_H+j*2000_H$

Port j MAC Layer4 Address5 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
L4DP5															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
L4SP5															
rw															

Field	Bits	Type	Description
L4SP5	15:0	rw	**Layer 4 Source Port Number Field** When the L4PEN0 bit is reset and the L4DPM0 bit is set in the MAC_L3_L4_Control5 register, this field contains the value to be matched with the TCP Source Port Number field in the IPv4 or IPv6 packets. When the L4PEN0 and L4DPM0 bits are set in MAC_L3_L4_Control5 register, this field contains the value to be matched with the UDP Source Port Number field in the IPv4 or IPv6 packets.
L4DP5	31:16	rw	**Layer 4 Destination Port Number Field** When the L4PEN0 bit is reset and the L4DPM0 bit is set in the MAC_L3_L4_Control5 register, this field contains the value to be matched with the TCP Destination Port Number field in the IPv4 or IPv6 packets. When the L4PEN0 and L4DPM0 bits are set in MAC_L3_L4_Control5 register, this field contains the value to be matched with the UDP Destination Port Number field in the IPv4 or IPv6 packets.

14.6.467 Port j MAC Layer4 Address6

Layer 4 Address 6 register has the Layer 4 (TCP, UDP) Source and Destination Port fields checked against the received packet.

Portj_MAC_Layer4_Address6 (j=0-1) Offset address: $10C04_H+j*2000_H$

Port j MAC Layer4 Address6 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
L4DP6															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
L4SP6															
rw															

Field	Bits	Type	Description
L4SP6	15:0	rw	**Layer 4 Source Port Number Field** When the L4PEN0 bit is reset and the L4DPM0 bit is set in the MAC_L3_L4_Control6 register, this field contains the value to be matched with the TCP Source Port Number field in the IPv4 or IPv6 packets. When the L4PEN0 and L4DPM0 bits are set in MAC_L3_L4_Control6 register, this field contains the value to be matched with the UDP Source Port Number field in the IPv4 or IPv6 packets.

(table continues...)

(continued)

Field	Bits	Type	Description
L4DP6	31:16	rw	**Layer 4 Destination Port Number Field** When the L4PEN0 bit is reset and the L4DPM0 bit is set in the MAC_L3_L4_Control6 register, this field contains the value to be matched with the TCP Destination Port Number field in the IPv4 or IPv6 packets. When the L4PEN0 and L4DPM0 bits are set in MAC_L3_L4_Control6 register, this field contains the value to be matched with the UDP Destination Port Number field in the IPv4 or IPv6 packets.

14.6.468 Port j MAC Layer4 Address7

Layer 4 Address 7 register has the Layer 4 (TCP, UDP) Source and Destination Port fields checked against the received packet.

Portj_MAC_Layer4_Address7 (j=0-1) Offset address: $10C04_H+j*2000_H$

Port j MAC Layer4 Address7 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
L4DP7															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
L4SP7															
rw															

Field	Bits	Type	Description
L4SP7	15:0	rw	**Layer 4 Source Port Number Field** When the L4PEN0 bit is reset and the L4DPM0 bit is set in the MAC_L3_L4_Control7 register, this field contains the value to be matched with the TCP Source Port Number field in the IPv4 or IPv6 packets. When the L4PEN0 and L4DPM0 bits are set in MAC_L3_L4_Control7 register, this field contains the value to be matched with the UDP Source Port Number field in the IPv4 or IPv6 packets.
L4DP7	31:16	rw	**Layer 4 Destination Port Number Field** When the L4PEN0 bit is reset and the L4DPM0 bit is set in the MAC_L3_L4_Control7 register, this field contains the value to be matched with the TCP Destination Port Number field in the IPv4 or IPv6 packets. When the L4PEN0 and L4DPM0 bits are set in MAC_L3_L4_Control7 register, this field contains the value to be matched with the UDP Destination Port Number field in the IPv4 or IPv6 packets.

14.6.469 Port j MAC Timestamp Control

This register controls the operation of the System Time generator and processing of PTP packets for timestamping in the Receiver.

Portj_MAC_Timestamp_Control (j=0-1) Offset address: $10D00_H+j*2000_H$

Port j MAC Timestamp Control Kernel Reset value: $0000\ 2000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		0	AV8021ASMEN	0		0	TXTSSTSM	0			ESTI	CSC	TSENMACADDR	SNAPTYPSEL	
r		r	rw	r		r	rw	r			rw	rw	rw	rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TSMSTRENA	TSEVNTENA	TSIPV4ENA	TSIPV6ENA	TSIPENA	TSVER2ENA	TSCTRLSSR	TSENALL	0	PTGE	TSADDREG	0	TSUPDT	TSINIT	TSCFUPDT	TSENA
rw	rw	rw	rw	rw	rw	rw	rw	r	rw	rw	r	rw	rw	rw	rw

Field	Bits	Type	Description
TSENA	0	rw	**Enable Timestamp** When this bit is set, the timestamp is added for Transmit and Receive packets. When disabled, timestamp is not added for transmit and receive packets and the Timestamp Generator is also suspended. You need to initialize the Timestamp (system time) after enabling this mode. On the Receive side, the MAC processes the 1588 packets only if this bit is set.
TSCFUPDT	1	rw	**Fine or Coarse Timestamp Update** When this bit is set, the Fine method is used to update system timestamp. When this bit is reset, Coarse method is used to update the system timestamp.
TSINIT	2	rw	**Initialize Timestamp** When this bit is set, the system time is initialized (overwritten) with the value specified in the MAC_System_Time_Seconds_Update and MAC_System_Time_Nanoseconds_Update registers. This bit must be zero before it is updated. This bit is reset when the initialization is complete. The Timestamp Higher Word register (if enabled during controller configuration) can only be initialized. When Media Clock Generation and Recovery is configured (DWCXG_FLEXI_PPS_OUT_EN) and enabled MAC_Presn_Time_Updt should also be updated before setting this field. Access restriction applies. Setting 1 sets. Self-cleared. Setting 0 has no effect.

(table continues...)

(continued)

Field	Bits	Type	Description
TSUPDT	3	rw	**Update Timestamp** When this bit is set, the system time is updated (added or subtracted) with the value specified in MAC_System_Time_Seconds_Update and MAC_System_Time_Nanoseconds_Update. This bit must be zero before updating it. This bit is reset when the update is complete in hardware. The Timestamp Higher Word register (if enabled during controller configuration) is not updated. When Media Clock Generation and Recovery is configured (DWCXG_FLEXI_PPS_OUT_EN) and enabled MAC_Presn_Time_Updt should also be updated before setting this field. Access restriction applies. Setting 1 sets. Self-cleared. Setting 0 has no effect.
TSADDREG	5	rw	**Update Addend Register** When this bit is set, the content of the Timestamp Addend register is updated in the PTP block for fine correction. This bit is cleared when the update is complete. This bit must be zero before it is set.
PTGE	6	rw	**Presentation Time Generation Enable** When this bit is set the Presentation Time generation is enabled.
TSENALL	8	rw	**Enable Timestamp for All Packets** When this bit is set, the timestamp snapshot is enabled for all packets received by the MAC.
TSCTRLSSR	9	rw	**Timestamp Digital or Binary Rollover Control** When this bit is set, the Timestamp Low register rolls over after 0x3B9A_C9FF value (that is, 1 nanosecond accuracy) and increments the timestamp (High) seconds. When this bit is reset, the rollover value of sub-second register is 0x7FFF_FFFF. The sub-second increment must be programmed correctly depending on the PTP reference clock frequency and the value of this bit.
TSVER2ENA	10	rw	**Enable PTP Packet Processing for Version 2 Format** When this bit is set, the IEEE 1588 version 2 format is used to process the PTP packets. When this bit is reset, the IEEE 1588 version 1 format is used to process the PTP packets. The IEEE 1588 formats are described in "PTP Processing and Control" section.
TSIPENA	11	rw	**Enable Processing of PTP over Ethernet Packets** When this bit is set, the MAC receiver processes the PTP packets encapsulated directly in the Ethernet packets. When this bit is reset, the MAC ignores the PTP over Ethernet packets.
TSIPV6ENA	12	rw	**Enable Processing of PTP Packets Sent over IPv6-UDP** When this bit is set, the MAC receiver processes the PTP packets encapsulated in IPv6-UDP packets. When this bit is clear, the MAC ignores the PTP transported over IPv6-UDP packets.
TSIPV4ENA	13	rw	**Enable Processing of PTP Packets Sent over IPv4-UDP** When this bit is set, the MAC receiver processes the PTP packets encapsulated in IPv4-UDP packets. When this bit is reset, the MAC ignores the PTP transported over IPv4-UDP packets. This bit is set by default.

(table continues...)

(continued)

Field	Bits	Type	Description
TSEVNTENA	14	rw	**Enable Timestamp Snapshot for Event Messages** When this bit is set, the timestamp snapshot is taken only for event messages (SYNC, Delay_Req, Pdelay_Req, or Pdelay_Resp). When this bit is reset, the snapshot is taken for all messages except Announce, Management, and Signaling.
TSMSTRENA	15	rw	**Enable Snapshot for Messages Relevant to Master** When this bit is set, the snapshot is taken only for the messages that are relevant to the master node. Otherwise, the snapshot is taken for the messages relevant to the slave node.
SNAPTYPSEL	17:16	rw	**Select PTP packets for Taking Snapshots** These bits, along with Bits 15 and 14, decide the set of PTP packet types for which snapshot needs to be taken.
TSENMACADDR	18	rw	**Enable MAC Address for PTP Packet Filtering** When this bit is set, the DA MAC address (that matches any MAC Address register) is used to filter the PTP packets when PTP is directly sent over Ethernet. Note: When this bit is set, the PTP Offload function qualifies received PTP packet with unicast DA matching MAC Address 0 register (in addition to received PTP packet with unicast DA matching PTP special multicast addresses) for PTP response generation.
CSC	19	rw	**Enable checksum correction during OST for PTP over UDP/IPv4 packets** When this bit is set, the last two bytes of PTP message sent over UDP/IPv4 is updated to keep the UDP checksum correct, for changes made to origin timestamp and/or correction field as part of one step timestamp operation. The application must form the packet with these two dummy bytes. When reset, no updates are done to keep the UDP checksum correct. The application must form the packet with UDP checksum set to 0.
ESTI	20	rw	**External System Time Input** When this bit is set, the MAC uses the external 64-bit reference System Time input for the following: - To take the timestamp provided as status - To insert the timestamp in transmit PTP packets When this bit is reset, the MAC uses the internal reference System Time.
TXTSSTSM	24	rw	**Transmit Timestamp Status Mode** - When this bit is set, the MAC overwrites the earlier transmit timestamp status even if it is not read by the software. The MAC indicates this by setting the TXTSSTSMIS bit of the MAC_TxTimestamp_Status_Nanoseconds register. - When this bit is reset, the MAC ignores the timestamp status of current packet if the timestamp status of previous packet is not read by the software. The MAC indicates this by setting the TXTSSTSHI bit of the MAC_TxTimestamp_Status_Seconds register.

(table continues...)

(continued)

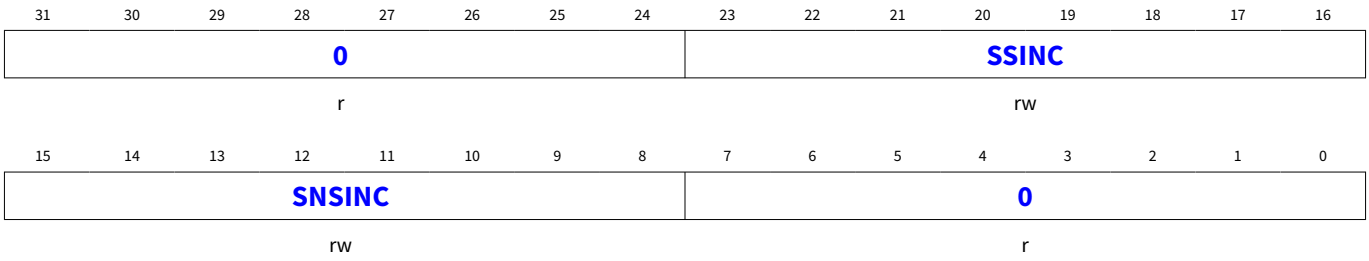
Field	Bits	Type	Description
AV8021ASMEN	28	rw	**AV IEEE 802.1AS Mode Enable** When this bit is set, the MAC processes only untagged PTP over Ethernet packets for providing PTP status and capturing timestamp snapshots, that is, IEEE 802.1AS mode of operation.
0	4, 7, 23:21, 25, 27:26, 29, 31:30	r	**Reserved** Read as 0; should be written with 0.

14.6.470 Port j MAC Sub Second Increment

In Coarse Update mode [Bit 1 in MAC_Timestamp_Control register], the value in this register is added to the system time every clock cycle of clk_ptp_ref_i. In Fine Update mode, the value in this register is added to the system time when the Accumulator gets an overflow.

Portj_MAC_Sub_Second_Increment (j=0-1) Offset address: $10D04_H+j*2000_H$

Port j MAC Sub Second Increment Kernel Reset value: $0000\ 0000_H$



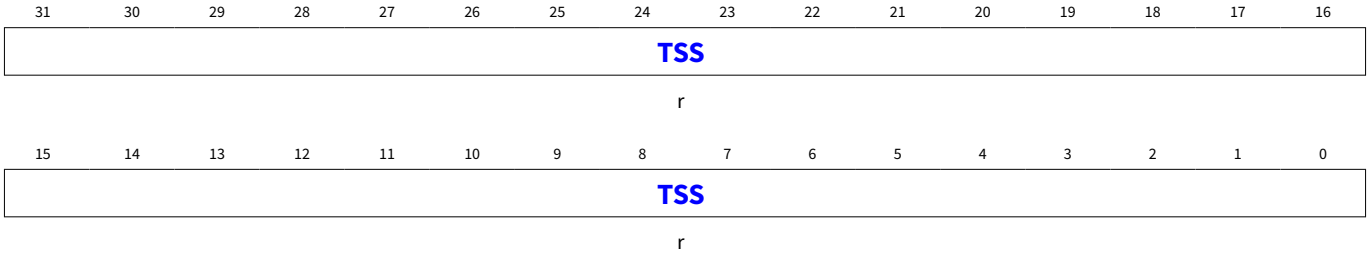
Field	Bits	Type	Description
SNSINC	15:8	rw	**Sub-nanosecond Increment Value** This field contains the sub-nanosecond increment value, represented in nanoseconds multiplied by 28. This value is accumulated with the sub-nanoseconds field of the sub-second register. For example, when TSCTRLSSR field in the MAC_Timestamp_Control register is set. and if the required increment is 5.3ns, then SSINC must be 0x05 and SNSINC must be 0x4C.
SSINC	23:16	rw	**Sub-second Increment Value** The value programmed in this field is accumulated with the contents of the sub-second register. For example, when the PTP clock is 50 MHz (period is 20 ns), you must program 20 (0x14) when the System Time-Nanoseconds register has an accuracy of 1 ns [Bit 9 (TSCTRLSSR) is set in MAC_Timestamp_Control]. When TSCTRLSSR is clear, the Nanoseconds register has a resolution of ~0.465 ns. In this case, you must program a value of 43 (0x2B) which is derived by 20 ns/0.465.
0	7:0, 31:24	r	**Reserved** Read as 0; should be written with 0.

14.6.471 Port j MAC System Time Seconds

The System Time-Seconds register, along with System Time-Nanoseconds register, indicates the current value of the system time maintained by the MAC. Though it is updated on a continuous basis, there is some delay from the actual time because of clock domain transfer latencies (from clk_ptp_ref_i to CSR clock).

Portj_MAC_System_Time_Seconds (j=0-1) Offset address: $10D08_H+j*2000_H$

Port j MAC System Time Seconds Kernel Reset value: $0000\ 0000_H$



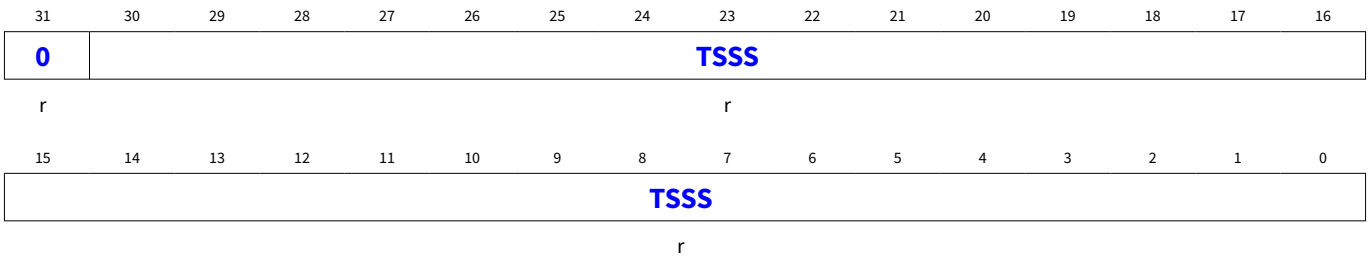
Field	Bits	Type	Description
TSS	31:0	r	**Timestamp Second** The value in this field indicates the current value in seconds of the System Time maintained by the MAC.

14.6.472 Port j MAC System Time Nanoseconds

The System Time-Nanoseconds register, along with System Time-Seconds register, indicates the current value of the system time maintained by the MAC.

Portj_MAC_System_Time_Nanoseconds (j=0-1) Offset address: $10D0C_H+j*2000_H$

Port j MAC System Time Nanoseconds Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
TSSS	30:0	r	**Timestamp Sub Seconds** The value in this field has the sub-second representation of time, with an accuracy of 0.46 ns. When Bit 9 is set in MAC_Timestamp_Control, each bit represents 1 ns. The maximum value is 0x3B9A_C9FF after which it rolls-over to zero.
0	31	r	**Reserved** Read as 0; should be written with 0.

14.6.473 Port j MAC System Time Seconds Update

The System Time-Seconds Update register, along with the System Time-Nanoseconds Update register, initializes or updates the system time maintained by the MAC. You must write both registers before setting the TSINIT or TSUPDT bits in MAC_Timestamp_Control register.

Portj_MAC_System_Time_Seconds_Update (j=0-1) Offset address: $10D10_H+j*2000_H$

Port j MAC System Time Seconds Update Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TSS															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TSS															
rw															

Field	Bits	Type	Description
TSS	31:0	rw	**Timestamp Second** Timestamp Seconds The value in this field is the seconds part of the update. When ADDSUB is reset, this field must be programmed with the seconds part of the update value. When ADDSUB is set, this field must be programmed with the complement of the seconds part of the update value. For example, if 2.000000001 seconds need to be subtracted from the system time, the TSS field in the MAC_Timestamp_Seconds_Update register must be 0xFFFF_FFFE (that is, 2^32 - 2).

14.6.474 Port j MAC System Time Nanoseconds Update

The System Time-Nanoseconds Update register, along with the System Time-Seconds Update register, initializes or updates the system time maintained by the MAC. You must write both registers before setting the TSINIT or TSUPDT bits in MAC_Timestamp_Control register

Portj_MAC_System_Time_Nanoseconds_Update (j=0-1) Offset address: $10D14_H+j*2000_H$

Port j MAC System Time Nanoseconds Update Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADDSUB	TSSS														
rw	rw														

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TSSS															
rw															

Field	Bits	Type	Description
TSSS	30:0	rw	**Timestamp Sub Seconds** The value in this field is the sub-second part of the update. - When ADDSUB is reset, this field must be programmed with the sub-second part of the update value, with an accuracy based on the TSCTRLSSR bit of the MAC Timestamp Control register. - When ADDSUB is set, this field must be programmed with the complement of the sub-second part of the update value, as follows. - When TSCTRLSSR is set, the programmed value must be 10^9 - (sub-second value). When TSCTRLSSR is reset, the programmed value must be 2^31 - (sub-second_value). For example, when TSCTRLSSR bit is set and if 2.000000001 second needs to be subtracted with the system time, the TSS field in the MAC Timestamp Seconds update register must be 0xFFFF_FFFE (for instance, 2^32 - 2), ADDSUB bit in this register must be set, and the TSSS field must be 0x3B9A_C9FF (for instance, 10^9 - 1).
ADDSUB	31	rw	**Add or Subtract Time** When this bit is set, the time value is subtracted with the contents of the update register. When this bit is reset, the time value is added with the contents of the update register.

14.6.475 Port j MAC Timestamp Addend

This register value is used only when the system time is configured for Fine Update mode (TSCFUPDT bit in the MAC_Timestamp_Control register). The content of this register is added to a 32-bit accumulator in every clock cycle (of clk_ptp_ref_i) and the system time is updated when the accumulator overflows.

Portj_MAC_Timestamp_Addend (j=0-1) Offset address: $10D18_H+j*2000_H$

Port j MAC Timestamp Addend Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TSAR															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TSAR															
rw															

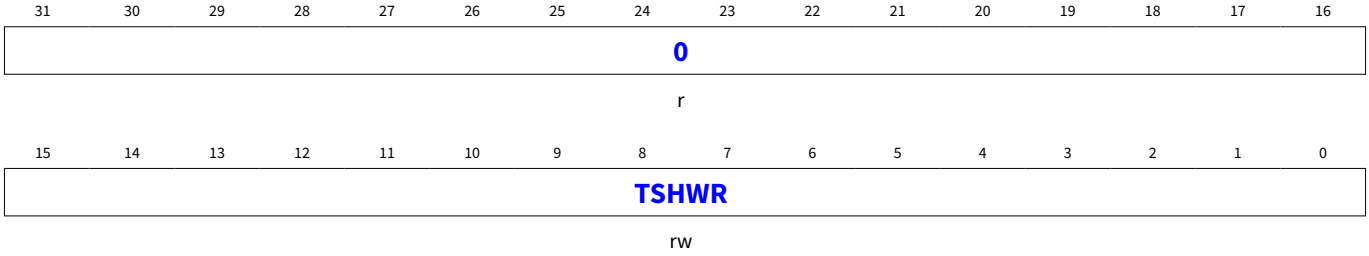
Field	Bits	Type	Description
TSAR	31:0	rw	**Timestamp Addend Register** This field indicates the 32-bit time value to be added to the Accumulator register to achieve time synchronization.

14.6.476 Port j MAC System Time Higher Word Seconds

This register contains the most-significant 16-bits of timestamp seconds value.

Portj_MAC_System_Time_Higher_Word_Seconds (j=0-1) Offset address: $10D1C_H+j*2000_H$

Port j MAC System Time Higher Word Seconds Kernel Reset value: $0000\ 0000_H$



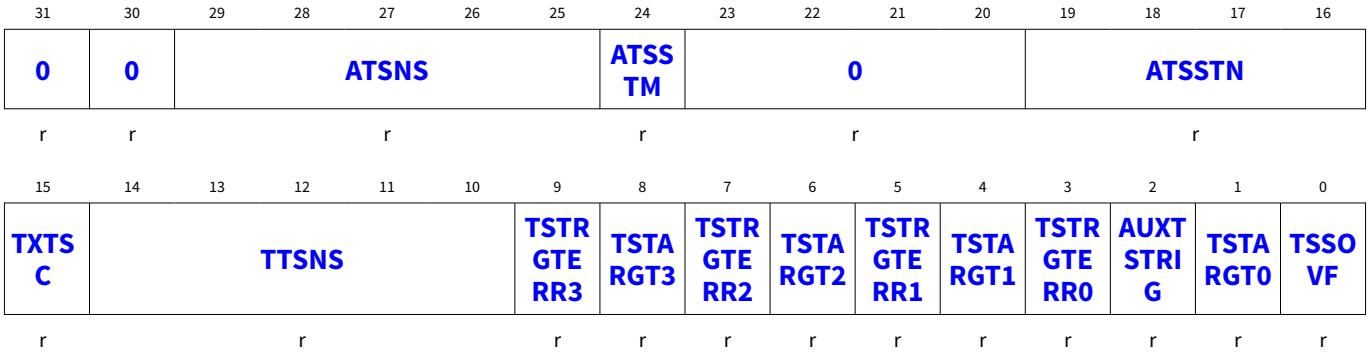
Field	Bits	Type	Description
TSHWR	15:0	rw	**Timestamp Higher Word Register** This field contains the most-significant 16-bits of timestamp seconds value. This register is optional. This register is directly written to initialize the value and it is incremented when there is an overflow from 32-bits of the System Time - Seconds register.
0	31:16	r	**Reserved** Read as 0; should be written with 0.

14.6.477 Port j MAC Timestamp Status

The Timestamp Status register All bits except Bits[29:25] gets cleared when the application reads this register.

Portj_MAC_Timestamp_Status (j=0-1) Offset address: $10D20_H+j*2000_H$

Port j MAC Timestamp Status Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
TSSOVF	0	r	**Timestamp Seconds Overflow** When this bit is set, it indicates that the seconds value of the timestamp (when supporting version 2 format) has overflowed beyond 32'hFFFF_FFFF. Access restriction applies. Clears on read (or write of 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event.

(table continues...)

(continued)

Field	Bits	Type	Description
TSTARGT0	1	r	**Timestamp Target Time Reached** When this bit is set and MCGREN0 of MAC_PPS_Control register is reset, it indicates that the value of system time is greater than or equal to the value specified in the MAC_PPS0_Target_Time_Seconds and MAC_PPS0_Target_Time_Nanoseconds registers. When this bit is set and MCGREN0 of MAC_PPS_Control register is set, it indicates that mcgr_dma_req_o[0] is asserted. Access restriction applies. Clears on read (or write of 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event.
AUXTSTRIG	2	r	**Reserved**
TSTRGTERR0	3	r	**Timestamp Target Time Error** This bit is set when the latest target time programmed in the MAC_PPS0_Target_Time_Seconds and MAC_PPS0_Target_Time_Nanoseconds registers has already elapsed. The bit is also set when the PPS command programmed cannot be executed as the earlier programmed target time has already elapsed. This bit is cleared when the application reads this bit. Access restriction applies. Clears on read (or write of 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event.
TSTARGT1	4	r	**Timestamp Target Time Reached for Target Time PPS1** When this bit is set and MCGREN1 of MAC_PPS_Control register is reset, it indicates that the value of system time is greater than or equal to the value specified in the MAC_PPS1_Target_Time_Seconds and MAC_PPS1_Target_Time_Nanoseconds registers. When this bit is set and MCGREN1 of MAC_PPS_Control register is set, it indicates that mcgr_dma_req_o[1] is asserted. Access restriction applies. Clears on read (or write of 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event.
TSTRGTERR1	5	r	**Timestamp Target Time Error** This bit is set when the latest target time programmed in the MAC_PPS1_Target_Time_Seconds and MAC_PPS1_Target_Time_Nanoseconds registers has already elapsed. The bit is also set when the PPS command programmed cannot be executed as the earlier programmed target time has already elapsed. This bit is cleared when the application reads this bit. Access restriction applies. Clears on read (or write of 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event.
TSTARGT2	6	r	**Timestamp Target Time Reached for Target Time PPS2** When this bit is set and MCGREN2 of MAC_PPS_Control register is reset, it indicates that the value of system time is greater than or equal to the value specified in the MAC_PPS2_Target_Time_Seconds and MAC_PPS2_Target_Time_Nanoseconds registers. When this bit is set and MCGREN2 of MAC_PPS_Control register is set, it indicates that mcgr_dma_req_o[2] is asserted. Access restriction applies. Clears on read (or write of 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event.

(table continues...)

(continued)

Field	Bits	Type	Description
TSTRGTERR2	7	r	**Timestamp Target Time Error** This bit is set when the latest target time programmed in the MAC_PPS2_Target_Time_Seconds and MAC_PPS2_Target_Time_Nanoseconds registers has already elapsed. The bit is also set when the PPS command programmed cannot be executed as the earlier programmed target time has already elapsed. This bit is cleared when the application reads this bit. Access restriction applies. Clears on read (or write of 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event.
TSTARGT3	8	r	**Timestamp Target Time Reached for Target Time PPS3** When this bit is set and MCGREN3 of MAC_PPS_Control register is reset, it indicates that the value of system time is greater than or equal to the value specified in the MAC_PPS3_Target_Time_Seconds and MAC_PPS3_Target_Time_Nanoseconds registers. When this bit is set and MCGREN3 of MAC_PPS_Control register is set, it indicates that mcgr_dma_req_o[3] is asserted. Access restriction applies. Clears on read (or write of 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event.
TSTRGTERR3	9	r	**Timestamp Target Time Error** This bit is set when the latest target time programmed in the MAC_PPS3_Target_Time_Seconds and MAC_PPS3_Target_Time_Nanoseconds registers has already elapsed. The bit is also set when the PPS command programmed cannot be executed as the earlier programmed target time has already elapsed. This bit is cleared when the application reads this bit. Access restriction applies. Clears on read (or write of 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event.
TTSNS	14:10	r	**Number of Tx Timestamp Snapshots** This field indicates the number of Tx Timestamp Snapshots available in the FIFO. A value equal to the selected depth of FIFO (2, 4, 8, or 16) indicates that the Tx Timestamp Snapshot FIFO is full. The Tx Timestamp Snapshot FIFO is implemented as synchronous FIFO in the CSR clock domain. This field is incremented along with the Tx Timestamp Snapshot write command to the FIFO. This field is decremented along with the Tx Timestamp Snapshot read command (issued when byte 3 or byte 0 of MAC_TxTimestamp_Status_Seconds register is read in little or big endian mode respectively) to the FIFO. If both write and read command is issued simultaneously, this field is not changed. This field is valid only if the "Enable IEEE 1588 Timestamp Support" option is selected.

(table continues...)

(continued)

Field	Bits	Type	Description
TXTSC	15	r	**Tx Timestamp Captured** When set, this bit indicates that a timestamp of the current packet being transmitted by the MAC is available in MAC_TxTimestamp_Status_Seconds and MAC_TxTimestamp_Status_Nanoseconds registers. This bit is cleared on reading (byte 3 or byte 0 in little or big endian mode respectively) the MAC_TxTimestamp_Status_Seconds register. When Tx Timestamp Status FIFO depth selected is more than 1, this bit is set as long as Tx Timestamp status FIFO has Tx Timestamp status entries. It is cleared when Seconds field corresponding to last entry is read (byte 3 or byte 0 in little or big endian mode respectively) from MAC_TxTimestamp_Status_Seconds register. Access restriction applies.This bit is cleared on reading (byte 3 or byte 0 in little or big endian mode respectively) the MAC_TxTimestamp_Status_Seconds register (or write to corresponding byte of MAC_Tx_Timestamp_Status_Seconds register when RCWE bit of MAC_CSR_SW_Ctrl register is set).
ATSSTN	19:16	r	**Reserved**
ATSSTM	24	r	**Reserved**
ATSNS	29:25	r	**Reserved**
0	23:20, 30, 31	r	**Reserved** Read as 0; should be written with 0.

14.6.478 Port j MAC Rx domain Time Incr

This register indicates the increment value of the PTP time maintained in Receive clock domain that is used for timestamping.

Portj_MAC_Rx_Domain_Time_Incr (j=0-1) Offset address: $10D24_H+j*2000_H$

Port j MAC Rx Domain Time Incr Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RXNS															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXSNS								0							
rw								r							

Field	Bits	Type	Description
RXSNS	15:8	rw	**Receive Domain Time Increment Value in Sub-Nanoseconds** This field indicates the Sub-Nanosecond part of the increment value of the PTP time maintained in Receive clock domain that is used for timestamping. This field has granularity of 1/256 ns. When both RXSNS and RXNS fields are all-zeros, the fixed value based on operating speed is used for PTP time increment in Receive clock domain.
RXNS	31:16	rw	**Receive Domain Time Increment Value in Nanoseconds** This field indicates the Nanosecond part of the increment value of the PTP time maintained in Receive clock domain that is used for timestamping. For example, when operating in 10G speed the clock period is 3.2ns, so for Digital Rollover mode the RXNS and RXSNS fields must be programmed with 3 and (0.2*256)=51 respectively. In Binary Rollover mode, 3/0.466=6.44 and 0.2+0.44=0.64, so RXNS and RXSNS fields must be programmed with 6 and (0.64*256)=164 respectively. When both RXSNS and RXNS fields are all-zeros, the fixed value based on operating speed is used for PTP time increment in Receive clock domain.
0	7:0	r	**Reserved** Read as 0; should be written with 0.

14.6.479 Port j MAC Tx domain Time Incr

This register indicates the increment value of the PTP time maintained in Transmit clock domain that is used for timestamping.

Portj_MAC_Tx_Domain_Time_Incr (j=0-1) Offset address: $10D28_H+j*2000_H$

Port j MAC Tx Domain Time Incr Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TXNS															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXSNS								0							
rw								r							

Field	Bits	Type	Description
TXSNS	15:8	rw	**Transmit Domain Time Increment Value in Sub-Nanoseconds** This field indicates the Sub-Nanosecond part of the increment value of the PTP time maintained in Transmit clock domain that is used for timestamping. This field has granularity of 1/256 ns. When both TXSNS and TXNS fields are all-zeros, the fixed value based on operating speed is used for PTP time increment in Transmit clock domain.

(table continues...)

(continued)

Field	Bits	Type	Description
TXNS	31:16	rw	**Transmit Domain Time Increment Value in Nanoseconds** This field indicates the Nanosecond part of the increment value of the PTP time maintained in Transmit clock domain that is used for timestamping. For example, when operating in 2.5G XGMII speed the clock period is 12.8ns, so for Digital Rollover mode the TXNS and TXSNS fields must be programmed with 12 and (0.8*256)=205 respectively. In Binary Rollover mode, 12/0.466=25.75 and 0.8+0.75=1.55, so TXNS and TXSNS fields must be programmed with (25+1)=26 and (0.55*256)=141 respectively. When both TXSNS and TXNS fields are all-zeros, the fixed value based on operating speed is used for PTP time increment in Transmit clock domain.
0	7:0	r	**Reserved** Read as 0; should be written with 0.

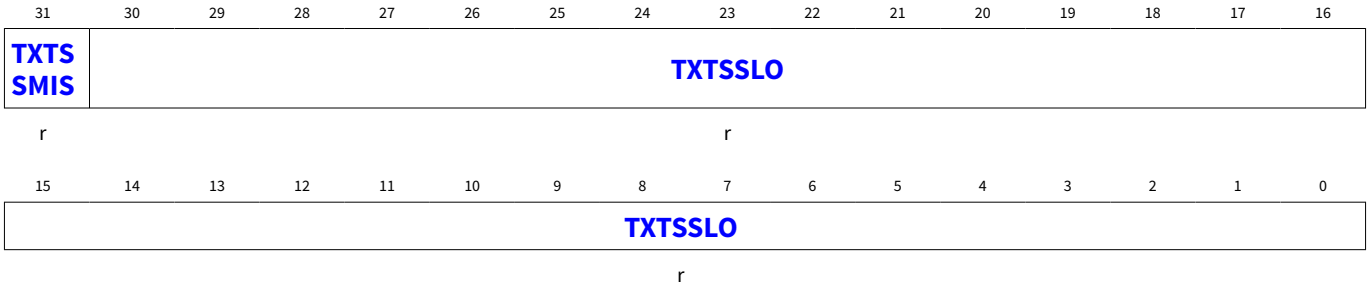
14.6.480 Port j MAC Tx Timestamp Status Nanoseconds

The MAC_TxTimestamp_Status_Nanoseconds register, along with MAC_TxTimestamp_Status_Seconds, gives the 64-bit timestamp captured for the PTP packet successfully transmitted by the MAC. This value is considered to be read by the application when the last byte of MAC_Tx_Timestamp_Status_Seconds is read. In the little-endian mode, this means bits[31:24] are read; in big-endian mode, bits[7:0] are read.

If the application does not read these registers and timestamp of another packet is captured, then either the current timestamp is lost (overwritten) or the new timestamp is lost (dropped), depending on the setting of the TXTSSTSM bit of the MAC_Timestamp_Control register. The status bit TXTSC bit [15] in MAC_Timestamp_Status register is set when the MAC transmitter captures the timestamp.

Portj_MAC_Tx_Timestamp_Status_Nanoseconds (j=0-1) Offset address: $10D30_H+j*2000_H$

Port j MAC Tx Timestamp Status Nanoseconds Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
TXTSSLO	30:0	r	**Transmit Timestamp Status Low** This field contains the 31 bits of the Nanoseconds field of the Transmit packet's captured timestamp.

(table continues...)

(continued)

Field	Bits	Type	Description
TXTSSMIS	31	r	**Transmit Timestamp Status Missed** When this bit is set, it indicates one of the following: - The timestamp of the current packet is ignored if TXTSSTSM bit of the MAC_Timestamp_Control register is reset - The timestamp of the previous packet is overwritten with timestamp of the current packet if TXTSSTSM bit of the MAC_Timestamp_Control register is set.

14.6.481 Port j MAC Tx Timestamp Status Seconds

The register contains the higher 32 bits of the timestamp (in seconds) captured when a PTP packet is transmitted.

Portj_MAC_Tx_Timestamp_Status_Seconds (j=0-1) Offset address: $10D34_H+j*2000_H$

Port j MAC Tx Timestamp Status Seconds Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TXTSSTSHI															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXTSSTSHI															
r															

Field	Bits	Type	Description
TXTSSTSHI	31:0	r	**Transmit Timestamp Status High** This field contains the lower 32 bits of the Seconds field of Transmit packet's captured timestamp.

14.6.482 Port j MAC Tx Timestamp Status PktID

This register contains the PTP packet ID corresponding to the Tx Timestamp Status in MAC_TxTimestamp_Status_Nanoseconds and MAC_TxTimestamp_Status_Seconds registers. This register is available when IEEE 1588 Tx Timestamp Status FIFO Depth is greater than 1.

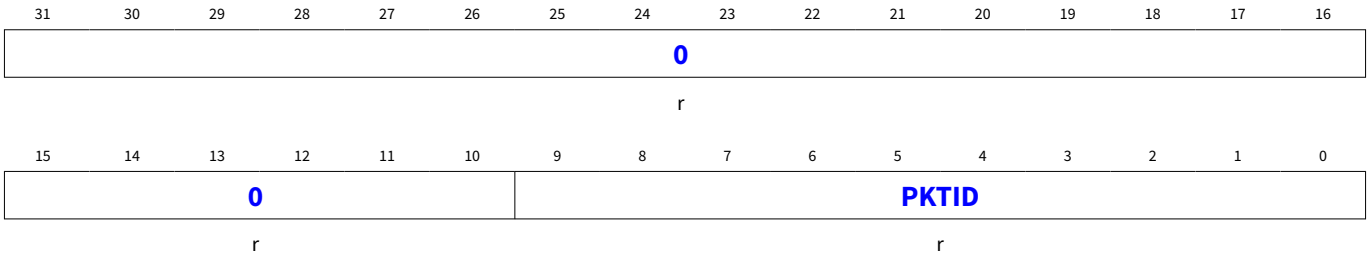
The correct reading sequence of these registers is as follows:

- MAC_TxTimestamp_Status_PktID register
- MAC_TxTimestamp_Status_Nanoseconds register
- MAC_TxTimestamp_Status_Seconds register

This is because, all the 3 registers are updated to a new value when the application reads the last byte of the MAC_TxTimestamp_Status_Seconds register.

Portj_MAC_Tx_Timestamp_Status_PktID (j=0-1) Offset address: $10D38_H+j*2000_H$

Port j MAC Tx Timestamp Status PktID Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
PKTID	9:0	r	**Packet ID** The field contains the packet ID corresponding to the Tx Timestamp status in MAC_TxTimestamp_Status_Nanoseconds and MAC_TxTimestamp_Status_Seconds registers. Note: The value of 0 for Packet ID is reserved, it is used by DWC_xgmac for indicating timestamp status of packets for which Packet ID was not provided by Software or for PTP response packets generated by the hardware PTP Offload engine.
0	31:10	r	**Reserved** Read as 0; should be written with 0.

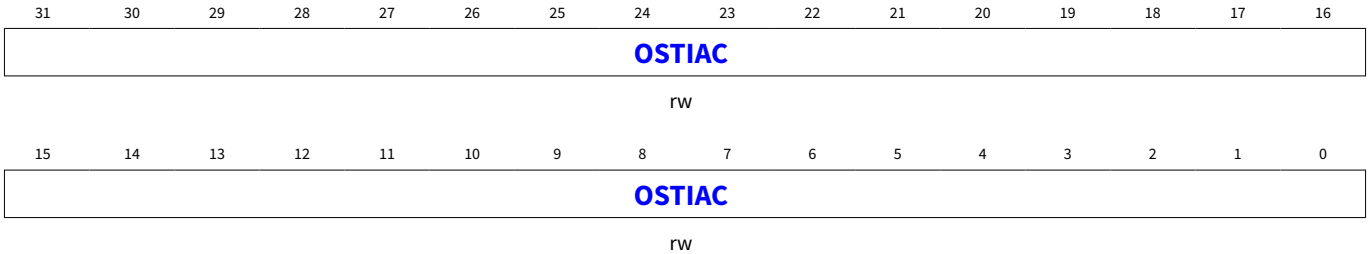
14.6.483 Port j MAC Timestamp Ingress Asym Corr

The MAC Timestamp Ingress Asymmetry Correction register contains the Ingress Asymmetry Correction value to be used while updating correction field in some PTP messages. Ideally, Ingress Asymmetry correction needs to be done in the receive side, when the corresponding packet is received. But when one-step time stamping is enabled, the controller performs ingress asymmetry correction on the transmit (egress) side, which still holds good for packets that are received and transmitted on the same port (for example, Pdelay Resp, which is sent for a Pdelay Request received in the same port).

In other cases (for example, transparent bridge), this feature can still be used if all receive ports have the same ingress correction value. If not, then OSTIAC must be programmed to 0x0.

Portj_MAC_Timestamp_Ingress_Asym_Corr (j=0-1) Offset address: $10D50_H+j*2000_H$

Port j MAC Timestamp Ingress Asym Corr Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
OSTIAC	31:0	rw	**One-Step Timestamp Ingress Asymmetry Correction** This field contains the ingress path asymmetry value to be added to correctionField of Sync and Pdelay_Resp PTP packet. The programmed value must be in units of nanoseconds and multiplied by 2^16. For example, 2.5 ns is represented as 0x00028000. The value can also be negative and then is represented in twos complement form with bit 31 representing the sign bit.

14.6.484 Port j MAC Timestamp Egress Asym Corr

The MAC Timestamp Egress Asymmetry Correction register contains the Egress Asymmetry Correction value to be used while updating the correction field in for some PTP message types.

Portj_MAC_Timestamp_Egress_Asym_Corr (j=0-1) Offset address: $10D54_H+j*2000_H$

Port j MAC Timestamp Egress Asym Corr Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OSTEAC															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OSTEAC															
rw															

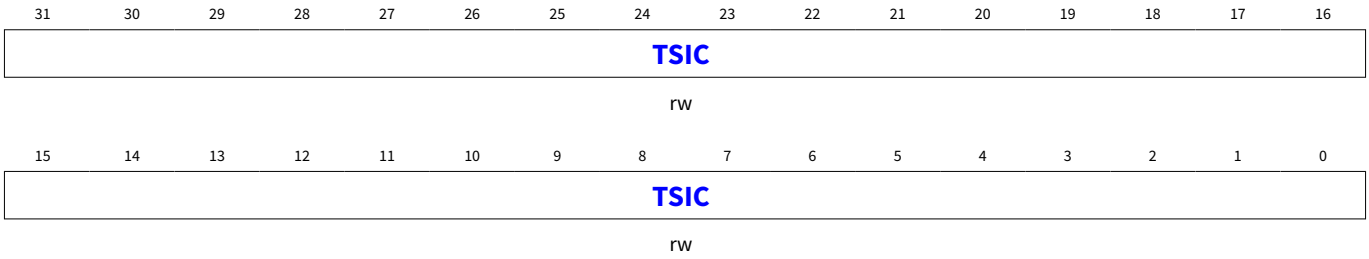
Field	Bits	Type	Description
OSTEAC	31:0	rw	**One-Step Timestamp Egress Asymmetry Correction** This field contains the egress path asymmetry value to be subtracted from correctionField of Delay_Req and Pdelay_Req PTP packet. The programmed value must be the negated value in units of nanoseconds multiplied by 2^16. For example, if the required correction is +2.5 ns, then the programmed value must be 0xFFFD_8000, which is twos complement of 0x0002_8000 (2.5 * 2^16). Similarly, if the required correction is -3.3 ns, then the programmed value is 0x0003_4CCC (3.3 * 2^16).

14.6.485 Port j MAC Timestamp Ingress Corr Nanosecond

The MAC Timestamp Ingress Correction Nanosecond register contains the correction value to be used with the captured timestamp value in the ingress path. This is used to compensate the errors due to timestamp synchronization from the PTP clock domain to the MAC Rx clock domain, and the latency incurred between the message timestamp point and the point at which the timestamp snapshot is taken.

Portj_MAC_Timestamp_Ingress_Corr_Nanosecond (j=0-1) Offset address: $10D58_H+j*2000_H$

Port j MAC Timestamp Ingress Corr Nanosecond Kernel Reset value: $0000\ 0000_H$



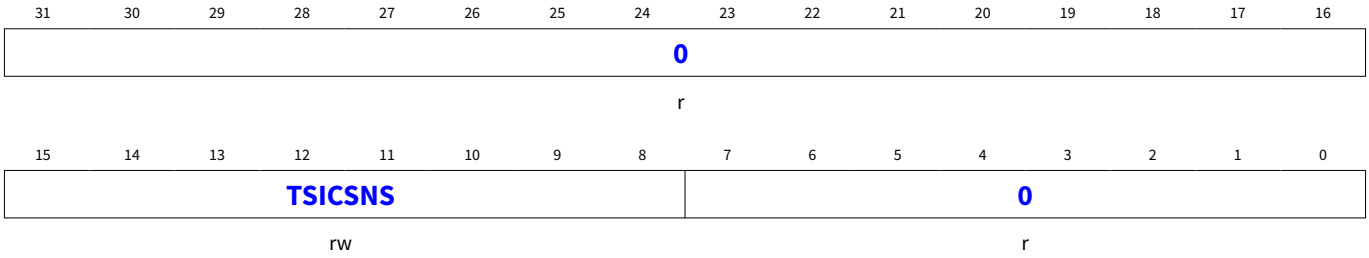
Field	Bits	Type	Description
TSIC	31:0	rw	**Timestamp Ingress Correction** This field contains the ingress path correction value in nanoseconds as defined by the "Ingress Correction" expression.

14.6.486 Port j MAC Timestamp Ingress Corr Subnanosecond

The MAC Timestamp Ingress Correction Sub-Nanosecond register contains the sub-nanosecond part of the correction value to be used with the captured timestamp value, for ingress directions.

Portj_MAC_Timestamp_Ingress_Corr_Subnanosecond (j=0-1) Offset address: $10D5C_H+j*2000_H$

Port j MAC Timestamp Ingress Corr Subnanosecond Kernel Reset value: $0000\ 0000_H$



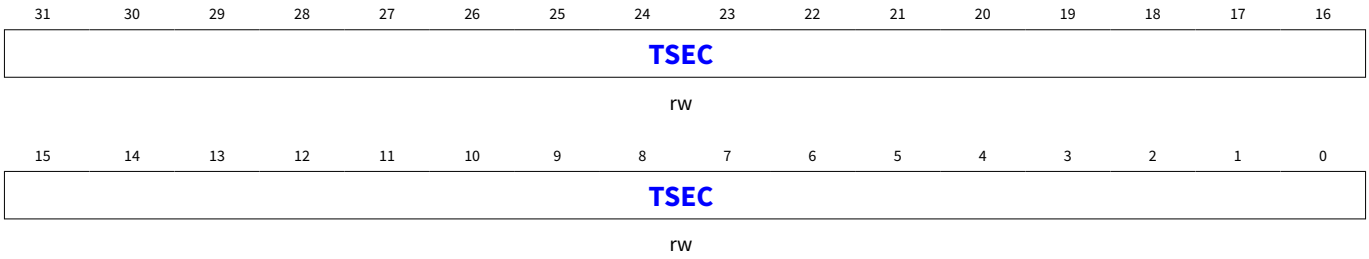
Field	Bits	Type	Description
TSICSNS	15:8	rw	**Timestamp Ingress Correction, sub-nanoseconds** This field contains the sub-nanoseconds part of the ingress path correction value as defined by the "Ingress Correction" expression.
0	7:0, 31:16	r	**Reserved** Read as 0; should be written with 0.

14.6.487 Port j MAC Timestamp Egress Corr Nanosecond

The MAC Egress Timestamp Correction Nanosecond register contains the correction value to be used with the captured timestamp value in the egress path. This is used to compensate the errors due to timestamp synchronization from PTP clock domain to MAC Tx clock domain, and the latency incurred between the point at which the timestamp snapshot is taken and the message timestamp point.

Portj_MAC_Timestamp_Egress_Corr_Nanosecond (j=0-1) Offset address: $10D60_H+j*2000_H$

Port j MAC Timestamp Egress Corr Nanosecond Kernel Reset value: $0000\ 0000_H$



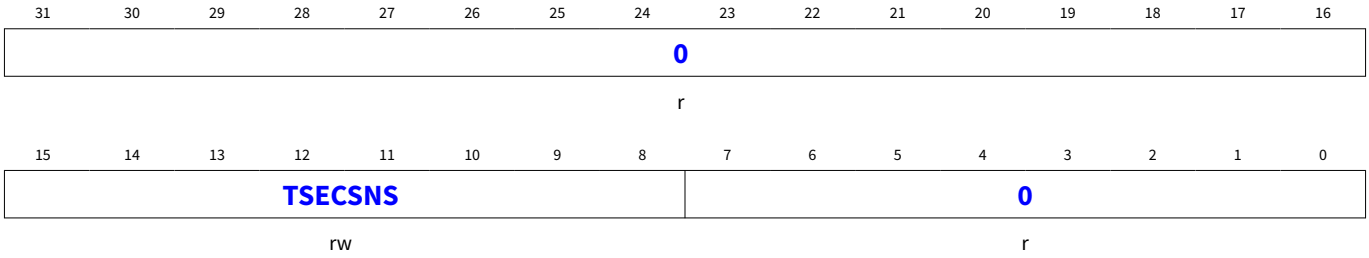
Field	Bits	Type	Description
TSEC	31:0	rw	**Timestamp Egress Correction** This field contains the nanoseconds part of the egress path correction value as defined by the "Egress Correction" expression,

14.6.488 Port j MAC Timestamp Egress Corr Subnanosecond

The MAC Timestamp Correction Egress Sub-nanosecond register contains the sub-nanosecond part of the correction value to be used with the captured timestamp value, for egress directions.

Portj_MAC_Timestamp_Egress_Corr_Subnanosecond (j=0-1) Offset address: $10D64_H+j*2000_H$

Port j MAC Timestamp Egress Corr Subnanosecond Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
TSECSNS	15:8	rw	**Timestamp Egress Correction, sub-nanoseconds** This field contains the sub-nanoseconds part of the egress path correction value as defined by the "Egress Correction" expression.
0	7:0, 31:16	r	**Reserved** Read as 0; should be written with 0.

14.6.489 Port j MAC PPS Control

. . Bits[22:16] are valid only when three or more Flexible PPS outputs are selected. Bits[14:8] are valid only when two or more Flexible PPS outputs are selected. Bits[6:4] are valid.

Portj_MAC_PPS_Control (j=0-1) Offset address: $10D70_H+j*2000_H$

Port j MAC PPS Control Kernel Reset value: $0000\ 0000_H$

14 Gigabit Ethernet (GETH)

31	30 29	28	27	26 25 24	23	22 21	20	19	18 17 16
0	0	0	0	0	0	0	0	0	0
r	r	r	r	r	r	r	r	r	r

15	14 13	12	11 10 9 8	7	6 5	4	3 2 1 0
MCGREN1	TRGTMODSEL1	PPSEN1	PPSCMD1	MCGREN0	TRGTMODSEL0	PPSEN0	PPSCTRL0_PPSCMD0
rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
PPSCTRL0_PPSCMD0	3:0	rw	**PPSCMD0: Flexible PPS Output (ptp_pps_o[0]) Control** Programming these bits with a non-zero value instructs the MAC to initiate an event. When the command is transferred or synchronized to the PTP clock domain, these bits get cleared automatically. The software must ensure that these bits are programmed only when they are all-zero. The following list describes the values of PPSCMD0: - 0000: No Command - 0001: START Single Pulse This command generates a single pulse rising at the start point defined in Target Time registers and of a duration defined in the PPS0 Width register. - 0010: START Pulse Train This command generates the train of pulses rising at the start point defined in the Target Time registers and of a duration defined in the PPS0 Width register and repeated at interval defined in the PPS Interval register. By default, the PPS pulse train is free-running unless stopped by the "Stop Pulse train at time" or "Stop Pulse Train immediately" commands. - 0011: Cancel START This command cancels the START Single Pulse and START Pulse Train commands if the system time has not crossed the programmed start time. - 0100: STOP Pulse train at time This command stops the train of pulses initiated by the START Pulse Train command (PPSCMD = 0010) after the time programmed in the Target Time registers elapses. - 0101: STOP Pulse Train immediately This command immediately stops the train of pulses initiated by the START Pulse Train command (PPSCMD = 0010). - 0110: Cancel STOP Pulse train This command cancels the STOP pulse train at time command if the programmed stop time has not elapsed. The PPS pulse train becomes free-running on the successful execution of this command. - 0111-1111: Reserved or Presentation Time Control When MCGREN0 is 1, these bits are treated as Presentation time control bits. The following list describes the values of PPSCMD0: - 0000: MCGR operation is not carried out. If set to this value in the mid of clock recovery or generation, all the processing inputs are flushed - 0001: Capture the Presentation time at the rising edge of mcg_pst_trig_i[0] into the MAC_PPS0_Target_Time_Seconds register - 0010: Capture the Presentation time at the falling edge of mcg_pst_trig_i[0] into the MAC_PPS0_Target_Time_Seconds register - 0011: Capture the Presentation time at both edges of mcg_pst_trig_i[0] into the MAC_PPS0_Target_Time_Seconds register - 0100-1000: Reserved - 1001: Toggle output on compare

(table continues...)

(continued)

Field	Bits	Type	Description
			- 1010: Pulse output low on compare for one PTP-clock cycle - 1011: Pulse output high on compare for one PTP-clock cycle - 1100-1111: Reserved This field can be the PPS Output Frequency Control (PPSCTRL0) or the Flexible PPS Output (ptp_pps_o[0]) Control (PPSCMD0) PPSCTRL0: PPS Output Frequency Control This field controls the frequency of the PPS0 output (ptp_pps_o[0]) signal. The default value of PPSCTRL is 0000, and the PPS output is 1 pulse (of width clk_ptp_ref_i) every second. For other values of PPSCTRL, the PPS output becomes a generated clock of following frequencies: - 0001: The binary rollover is 2 Hz, and the digital rollover is 1 Hz. - 0010: The binary rollover is 4 Hz, and the digital rollover is 2 Hz. - 0011: The binary rollover is 8 Hz, and the digital rollover is 4 Hz. - 0100: The binary rollover is 16 Hz, and the digital rollover is 8 Hz. - ... - 1111: The binary rollover is 32.768 KHz and the digital rollover is 16.384 KHz. In the binary rollover mode, the PPS output (ptp_pps_o) has a duty cycle of 50 percent with these frequencies. In the digital rollover mode, the PPS output frequency is an average number. The actual clock is of different frequency that gets synchronized every second. For example: When PPSCTRL = 0001, the PPS (1 Hz) has a low period of 537 ms and a high period of 463 ms When PPSCTRL = 0010, the PPS (2 Hz) is a sequence of: - One clock of 50 percent duty cycle and 537 ms period - Second clock of 463 ms period (268 ms low and 195 ms high) When PPSCTRL = 0011, the PPS (4 Hz) is a sequence of: - Three clocks of 50 percent duty cycle and 268 ms period - Fourth clock of 195 ms period (134 ms low and 61 ms high) This behavior is because of the non-linear toggling of bits in the digital rollover mode in the MAC_System_Time_Nanoseconds register.
PPSEN0	4	rw	**Flexible PPS Output Mode Enable** When this bit is set, Bits[3:0] function as PPSCMD. When this bit is reset, Bits[3:0] function as PPSCTRL (Fixed PPS mode).

(table continues...)

(continued)

Field	Bits	Type	Description
TRGTMODSEL0	6:5	rw	**Target Time Register Mode for PPS0 Output** This field indicates the Target Time registers mode for PPS0 output signal: - 00: Target Time registers are programmed only for generating the Target Time Reached interrupt event. The Flexible PPS function must not be enabled in this mode, otherwise spurious transitions might be observed on the corresponding ptp_pps_o output port - 01: Reserved - 10: Target Time registers are programmed for generating the Target Time Reached interrupt event and starting or stopping the PPS0 output signal generation. - 11: Target Time registers are programmed only for starting or stopping the PPS0 output signal generation. No Target Time Reached interrupt is asserted.
MCGREN0	7	rw	**MCGR Mode Enable for PPS0 Output** This field enables the 0th PPS instance to operate in PPS or MCGR mode. When set it operates in MCGR mode and on reset it operates in PPS mode.
PPSCMD1	11:8	rw	**Flexible PPS1 Output Control** This field controls the flexible PPS1 output (ptp_pps_o[1]) signal. This field is similar to the PPSCMD0 field. - When MCGREN1 is 1, PPSCMD1 indicated by the 4 bits [11:8] are taken as Presentation Time Control bits for media clock generation and recovery for comparator instance 1. This field is similar to the PPSCMD0 Presentation Time Control bits. - When MCGREN1 is 0, only 3 bits from [10:8] are used as PPSCMD1 and the 4th bit must be set as 0. Access restriction applies. Setting 1 sets. Self-cleared. Setting 0 has no effect.
PPSEN1	12	rw	
TRGTMODSEL1	14:13	rw	**Target Time Register Mode for PPS1 Output** This field indicates the Target Time registers mode for PPS1 output signal. This field is similar to the TRGTMODSEL0 field.
MCGREN1	15	rw	**MCGR Mode Enable for PPS1 Output** This field enables the 1st PPS instance to operate in PPS or MCGR mode. - 1: Operates in MCGR mode - 0: Operates in PPS mode

(table continues...)

(continued)

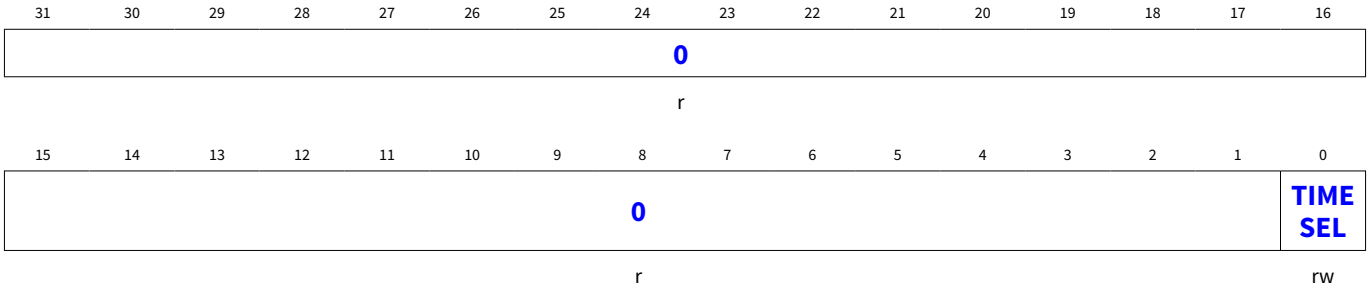
Field	Bits	Type	Description
0	18:16, 19, 20, 22:21, 23, 26:24, 27, 28, 30:29, 31	r	**Reserved** Read as 0; should be written with 0.

14.6.490 Port j MAC PPS Extnd Ctrl

.

Portj_MAC_PPS_Extnd_Ctrl (j=0-1) Offset address: $10D74_H+j*2000_H$

Port j MAC PPS Extnd Ctrl Kernel Reset value: $0000\ 0000_H$



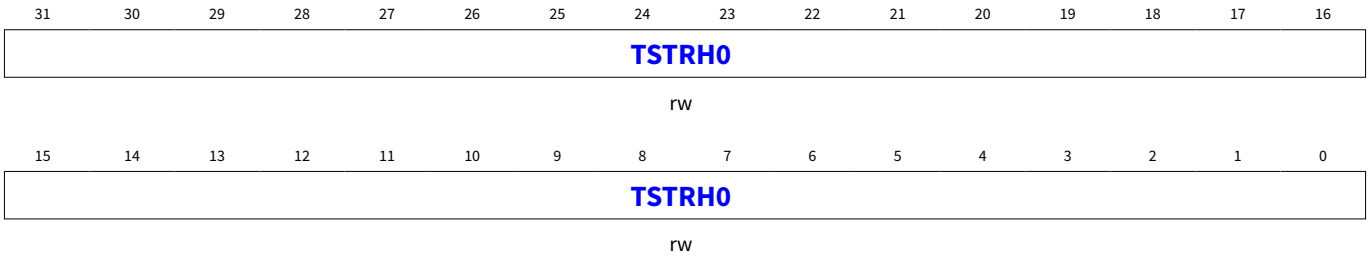
Field	Bits	Type	Description
TIMESEL	0	rw	
0	31:1	r	**Reserved** Read as 0; should be written with 0.

14.6.491 Port j MAC PPS0 Target Time Seconds

The PPS0 Target Time Seconds register, along with PPS0 Target Time Nanoseconds register, is used to schedule an interrupt event [Bit 1 of MAC_Timestamp_Status] when the system time exceeds the value programmed in these registers.

Portj_MAC_PPS0_Target_Time_Seconds (j=0-1) Offset address: $10D80_H+j*2000_H$

Port j MAC PPS0 Target Time Seconds Kernel Reset value: $0000\ 0000_H$



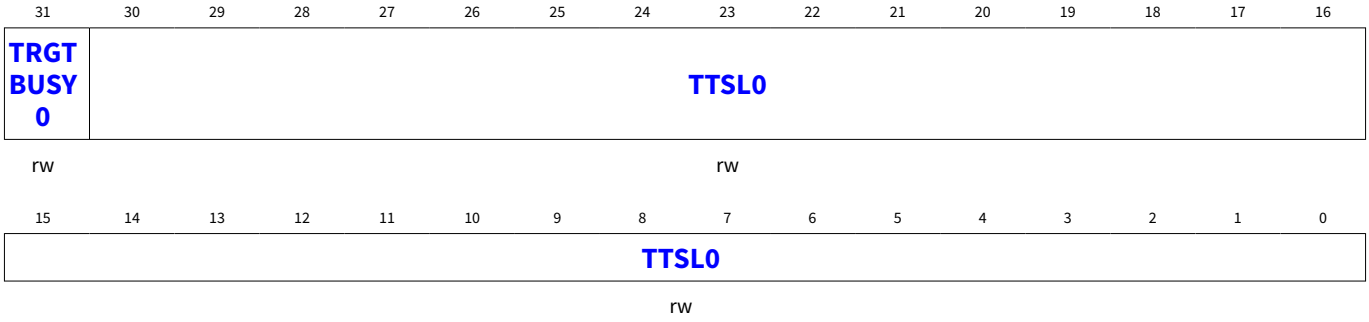
Field	Bits	Type	Description
TSTRH0	31:0	rw	**PPS0 Target Time Seconds Register** This field stores the time in seconds. When the timestamp value matches or exceeds both Target Timestamp registers, the MAC starts or stops the PPS signal output and generates an interrupt (if enabled) based on Bits[6:5] of MAC_PPS_Control register. If DWCXG_FLEXI_PPS_OUT_EN is enabled in the configuration and PTGE field of MAC_Timestamp_Control Register is set with Presentation time control set in recovery mode, then these bits indicate the TPT being programmed by the application and in generation mode it indicates the CPT generated at the sampled trigger.

14.6.492 Port j MAC PPS0 Target Time Nanoseconds

The PPS0 Target Time Nanoseconds register is present only when more than one Flexible PPS output is selected.

Portj_MAC_PPS0_Target_Time_Nanoseconds (j=0-1) Offset address: $10D84_H+j*2000_H$

Port j MAC PPS0 Target Time Nanoseconds Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
TTSL0	30:0	rw	**Target Time Low for PPS0 Register** This register stores the time in (signed) nanoseconds. When the value of the timestamp matches the value in both Target Timestamp registers, the MAC starts or stops the PPS signal output or generates a Target Time Reached interrupt (if enabled) based on the TRGTMODSEL0 field (Bits [6:5]) in MAC_PPS_Control. When the TSCTRLSSR bit is set in the MAC_Timestamp_Control register, this value must not exceed 0x3B9A_C9FF. The actual start or stop time of the PPS signal output might have an error margin up to one unit of sub-second increment value.
TRGTBUSY0	31	rw	**PPS0 Target Time Register Busy** The MAC sets this bit when the PPSCMD0 field in the MAC_PPS_Control register is programmed to 010 or 011. Programming the PPSCMD0 field to 010 or 011 instructs the MAC to synchronize the Target Time registers to the PTP clock domain. The MAC clears this bit after synchronizing the Target Time registers to the PTP clock domain The application must not update the Target Time registers when this bit is read as 1. Otherwise, the synchronization of the previous programmed time gets corrupted.

14.6.493 Port j MAC PPS0 Interval

The PPS0 Interval register contains the number of units of sub-second increment value between the rising edges of PPS0 signal output (ptp_pps_o[0]).

Portj_MAC_PPS0_Interval (j=0-1) Offset address: $10D88_H+j*2000_H$

Port j MAC PPS0 Interval Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PPSINT0															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PPSINT0															
rw															

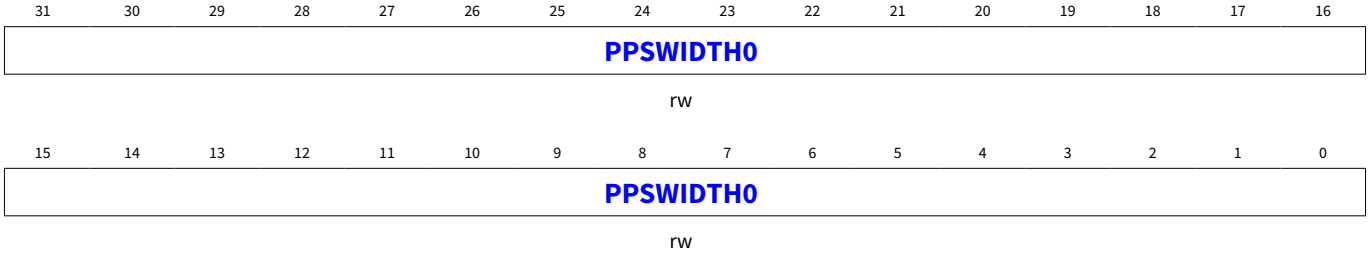
Field	Bits	Type	Description
PPSINT0	31:0	rw	**PPS0 Output Signal Interval** These bits store the interval between the rising edges of PPS0 signal output. The interval is stored in terms of number of units of sub-second increment value. You need to program one value less than the required interval. For example, if the PTP reference clock is 50 MHz (period of 20 ns), and desired interval between the rising edges of PPS0 signal output is 100 ns (that is, five units of sub-second increment value), you must program value 4 (5 - 1) in this register.

14.6.494 Port j MAC PPS0 Width

The PPS0 Width register contains the number of units of sub-second increment value between the rising and corresponding falling edges of PPS0 signal output (ptp_pps_o[0]).

Portj_MAC_PPS0_Width (j=0-1) Offset address: $10D8C_H+j*2000_H$

Port j MAC PPS0 Width Kernel Reset value: $0000\ 0000_H$



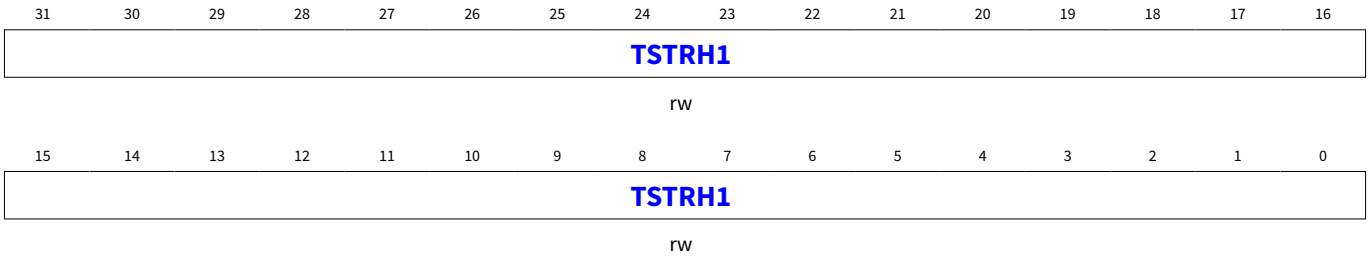
Field	Bits	Type	Description
PPSWIDTH0	31:0	rw	**PPS0 Output Signal Width** These bits store the width between the rising edge and corresponding falling edge of PPS0 signal output. The width is stored in terms of number of units of sub-second increment value. You need to program one value less than the required interval. For example, if PTP reference clock is 50 MHz (period of 20 ns), and width between the rising and corresponding falling edges of PPS0 signal output is 80 ns (that is, four units of sub-second increment value), you must program value 3 (4 - 1) in this register. Note: The value programmed in this register must be lesser than the value programmed in MAC_PPS0_Interval.

14.6.495 Port j MAC PPS1 Target Time Seconds

The PPS1 Target Time Seconds register, along with PPS1 Target Time Nanoseconds register, is used to schedule an interrupt event [Bit 1 of MAC_Timestamp_Status] when the system time exceeds the value programmed in these registers.

Portj_MAC_PPS1_Target_Time_Seconds (j=0-1) Offset address: $10D90_H+j*2000_H$

Port j MAC PPS1 Target Time Seconds Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
TSTRH1	31:0	rw	**PPS1 Target Time Seconds** This field stores the time in seconds. When the timestamp value matches or exceeds both Target Timestamp registers, the MAC starts or stops the PPS signal output and generates an interrupt (if enabled) based on Bits[14:13] of MAC_PPS_Control register. If DWCXG_FLEXI_PPS_OUT_EN is enabled in the configuration and PTGE field of MAC_Timestamp_Control Register is set with Presentation time control set in recovery mode, then these bits indicate the TPT being programmed by the application and in generation mode it indicates the CPT generated at the sampled trigger.

14.6.496 Port j MAC PPS1 Target Time Nanoseconds

The PPS1 Target Time Nanoseconds register, along with PPS1 Target Time Seconds register, is used to schedule an interrupt event [Bit 1 of MAC_Timestamp_Status] when the system time exceeds the value programmed in these registers.

Portj_MAC_PPS1_Target_Time_Nanoseconds (j=0-1) Offset address: $10D94_H+j*2000_H$

Port j MAC PPS1 Target Time Nanoseconds Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TRGTBUSY1								TTSL1							
rw								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TTSL1															
rw															

Field	Bits	Type	Description
TTSL1	30:0	rw	**Target Time Low for PPS1 Register** This register stores the time in (signed) nanoseconds. When the value of the timestamp matches the both Target Timestamp registers, then based on the TRGTMODSEL1 field in the MAC_PPS_Control register, the MAC starts or stops the PPS signal output or generates a Target Time Reached interrupt (if enabled). This value must not exceed 0x3B9A_C9FF when the TSCTRLSSR field is set in the MAC_Timestamp_Control register. The actual start or stop time of the PPS signal output might have an error margin up to one unit of sub-second increment value.

(table continues...)

(continued)

Field	Bits	Type	Description
TRGTBUSY1	31	rw	**PPS1 Target Time Register Busy** The MAC sets this bit when the PPSCMD1 field in MAC_PPS_Control register is programmed to 010 or 011. Programming the PPSCMD1 field to 010 or 011 instructs the MAC to synchronize the Target Time registers to the PTP clock domain. The MAC clears this bit after synchronizing the Target Time registers to the PTP clock domain The application must not update the Target Time registers when this bit is read as 1. Otherwise, the synchronization of the previous programmed time gets corrupted.

14.6.497 Port j MAC PPS1 Interval

The PPS1 Interval register contains the number of units of sub-second increment value between the rising edges of PPS1 signal output (ptp_pps_o[1]).

Portj_MAC_PPS1_Interval (j=0-1) Offset address: $10D98_H+j*2000_H$

Port j MAC PPS1 Interval Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PPSINT1															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PPSINT1															
rw															

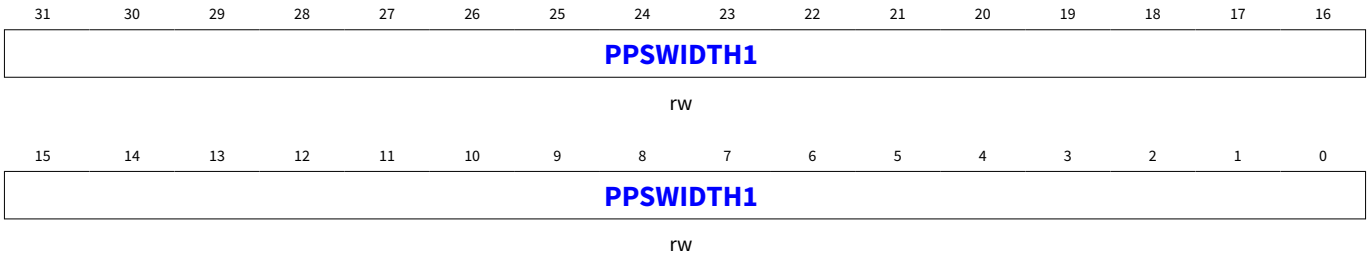
Field	Bits	Type	Description
PPSINT1	31:0	rw	**PPS1 Output Signal Interval** These bits store the interval between the rising edges of PPS1 signal output. The interval is stored in terms of number of units of sub-second increment value. You need to program one value less than the required interval. For example, if the PTP reference clock is 50 MHz (period of 20 ns), and desired interval between the rising edges of PPS1 signal output is 100 ns (that is, five units of sub-second increment value), you must program value 4 (5 - 1) in this register.

14.6.498 Port j MAC PPS1 Width

The PPS1 Width register contains the number of units of sub-second increment value between the rising and corresponding falling edges of PPS1 signal output (ptp_pps_o[1]).

Portj_MAC_PPS1_Width (j=0-1) Offset address: $10D9C_H+j*2000_H$

Port j MAC PPS1 Width Kernel Reset value: $0000\ 0000_H$



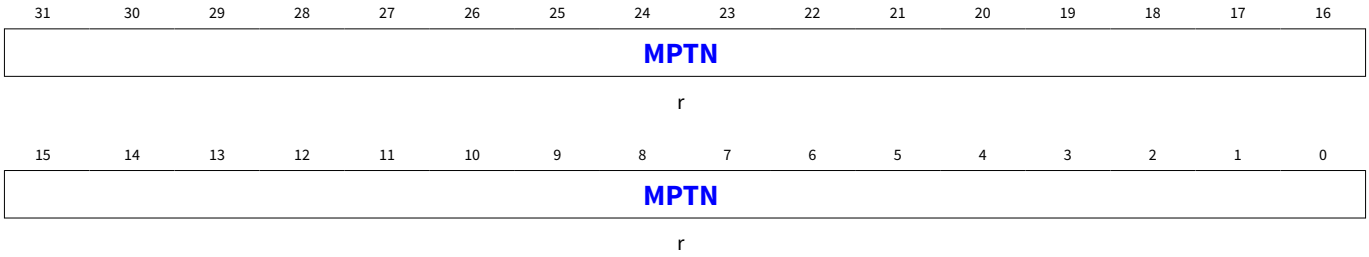
Field	Bits	Type	Description
PPSWIDTH1	31:0	rw	**PPS1 Output Signal Width** These bits store the width between the rising edge and corresponding falling edge of PPS1 signal output. The width is stored in terms of number of units of sub-second increment value. You need to program one value less than the required interval. For example, if PTP reference clock is 50 MHz (period of 20 ns), and width between the rising and corresponding falling edges of PPS1 signal output is 80 ns (that is, four units of sub-second increment value), you must program value 3 (4 - 1) in this register. Note: The value programmed in this register must be lesser than the value programmed in MAC_PPS1_Interval.

14.6.499 Port j MAC Presn Time ns

This register contains the 32-bit binary rollover equivalent time of the PTP System Time in ns.

Portj_MAC_Presn_Time_ns (j=0-1) Offset address: $10DE0_H+j*2000_H$

Port j MAC Presn Time ns Kernel Reset value: $0000\ 0000_H$



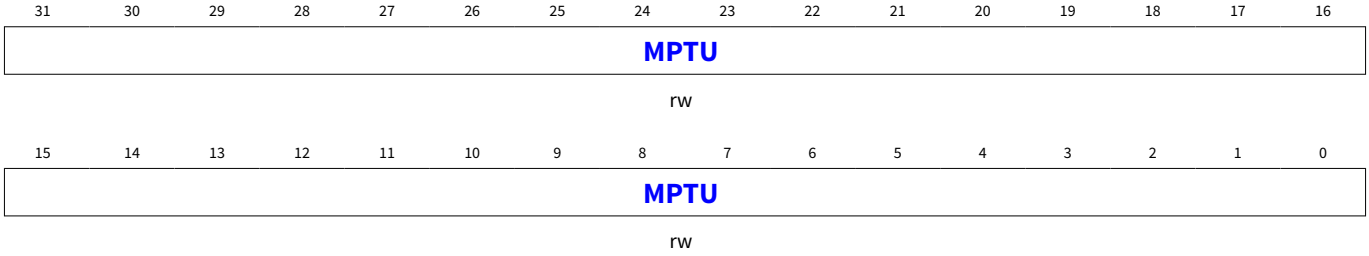
Field	Bits	Type	Description
MPTN	31:0	r	**MAC 1722 Presentation Time in ns** These bits indicate the value of the 32-bit binary rollover equivalent time of the PTP System Time in ns.

14.6.500 Port j MAC Presn Time Updt

This field holds the 32-bit value of MAC 1722 Presentation Time in ns, that should be added to the Current Presentation Time Counter value. Init happens when TSINIT is set, and update happens when the TSUPDT bit is set (TSINIT and TSINIT defined in MAC_Timestamp_ Control register).

Portj_MAC_Presn_Time_Updt (j=0-1) Offset address: $10DE4_H+j*2000_H$

Port j MAC Presn Time Updt Kernel Reset value: $0000\ 0000_H$



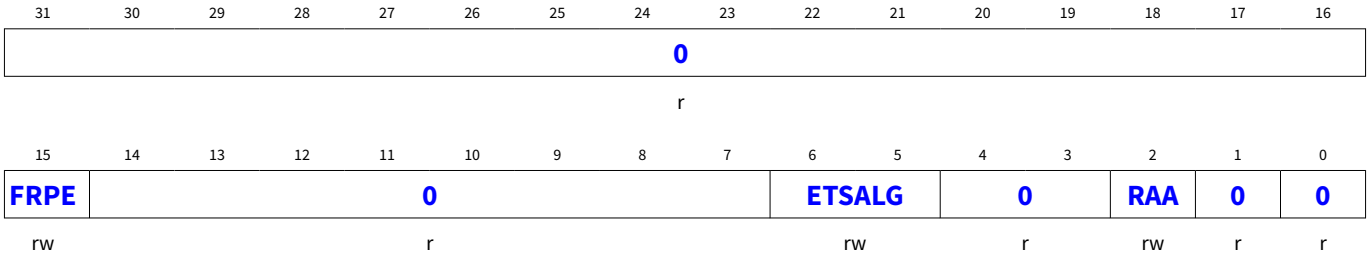
Field	Bits	Type	Description
MPTU	31:0	rw	**MAC 1722 Presentation Time Update** This field holds the init value or the update value for the presentation time. When used for update, this field holds the 32-bit value in ns, that should be added to the Current Presentation Time Counter value. Init happens when TSINIT is set, and update happens when the TSUPDT bit is set (TSINIT and TSINIT defined in MAC_Timestamp_ Control register). When ADDSUB field of MAC_System_Time_Nanoseconds_Update is set, this value is directly used for subtraction.

14.6.501 Port j MTL Operation Mode

The Operation Mode register establishes the Transmit and Receive operating modes and commands.

Portj_MTL_Operation_Mode (j=0-1) Offset address: $11000_H+j*2000_H$

Port j MTL Operation Mode Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
RAA	2	rw	**Receive Arbitration Algorithm** This field is used to select the arbitration algorithm for the Rx side. - 0: Strict Priority (SP) - 1: Weighted Strict Priority (WSP) Queue 0 has the lowest priority and the last queue has the highest priority. RXQ_WEGT field of MTL_RxQ(#i)_Control register controls the weight assigned to receive queue.
ETSALG	6:5	rw	**ETS Algorithm** This field selects the type of ETS algorithm to be applied for traffic classes whose transmission selection algorithm (TSA) is set to ETS: - 2'b00: WRR algorithm - 2'b11: Reserved

(table continues...)

(continued)

Field	Bits	Type	Description
FRPE	15	rw	**Flexible Receive Parser Enable** When this bit is set, Flexible Receive Parser is enabled for packet filtering and classification. When this bit is reset, Flexible Receive Parser is disabled for packet filtering and classification. Note: If the flexible receive parser is in the middle of the parsing then, FRPE programming change is applicable from next immediate packet. 0_B **DISABLE**: Flexible Rx parser is disabled 1_B **ENABLE**: Flexible Rx parser is enabled
0	0, 1, 4:3, 14:7, 31:16	r	**Reserved** Read as 0; should be written with 0.

14.6.502 Port j MTL Debug Control

The MTL Debug Control register controls the operation mode of debug access to the TXFIFO, RXFIFO and Descriptor Cache memories connected to the DWC_xgmac. Note: Consecutive write or read to this register must be performed after at least 16 clock cycles of the slowest clock among MAC Transmitter/Receiver clock and Application/CSR clock.

Portj_MTL_Debug_Control (j=0-1) Offset address: $11008_H+j*2000_H$

Port j MTL Debug Control Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0													EIEC	EIAE E	EIEE
r													rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	PKTI E	FIFOSEL		FIFO WRE N	FIFO RDE N	RSTS EL	RSTA LL	0	PKTSTATE		0	BYTEEN		DBG MOD	FDB GEN
r	rw	rw		rw	rw	rw	rw	r	rw		r	rw		rw	rw

Field	Bits	Type	Description
FDBGEN	0	rw	**FIFO Debug Access Enable** When this bit is set, access is available to the memories selected by FIFOSEL. The type of access (Debug mode or Slave mode) is determined by DBGMOD. When this bit is reset, the memories are in mission mode and direct access to these memories are disabled..

(table continues...)

(continued)

Field	Bits	Type	Description
DBGMOD	1	rw	**Debug Mode Access to FIFO** When this bit is set and FDBGEN is set, direct access is enabled to the memories selected by FIFOSEL. The transfer of packets between the host interface and the MAC through the TXFIFO & RXFIFO memory are disabled. The CPU/software can perform direct write/read operations to the memories as determined by programming of FIFOWREN/ FIFORDEN bits. When this bit is reset and FDBGEN is set, the TXFIFO and RXFIFO memory are put in Slave mode. In this mode, packets can be directly written into the TXFIFO memory for transmission by MAC as well as the packets received by the MAC can be read directly. The transfer of packets between the FIFO memories and the host interface is stopped. In this mode, only the following access are allowed: Packet Write access to the Tx FIFO Packet Read access to the Rx FIFO
BYTEEN	3:2	rw	**Write Data Byte Enable** This field indicates the number of data bytes valid in the MTL_Debug_Data register during Write operation. This is valid only when PKTSTATE is 2'b11 (EOP) and Tx FIFO or Rx FIFO is selected. - 00: Byte 0 valid - 01: Byte 0 and Byte 1 are valid - 10: Byte 0, Byte 1, and Byte 2 are valid - 11: All four bytes are valid
PKTSTATE	6:5	rw	**Write Control Type** This field is used to decide the tag bits along with the data being written into the FIFO. When the TxFIFO is written in Slave mode or Debug mode, this field marks the type of data written as follows: - 00: Packet Data - 01: Control Word - 10: SOP Data - 11: EOP Data When the Rx FIFO is being written in Slave mode, this field marks the type of data written as follows: - 00: Packet Data - 01: Normal Status - 10: Last Status - 11: EOP
RSTALL	8	rw	**Reset All Pointers** When this bit is set, the address pointers to all FIFO memory in btoh Debug and Slave modes are reset. This bit is self cleared.
RSTSEL	9	rw	**Reset Pointers of Selected FIFO** When this bit is set, the address pointer of the memory selected by FIFOSEL is reset in both Debug and Slave modes. This bit is self cleared.

(table continues...)

(continued)

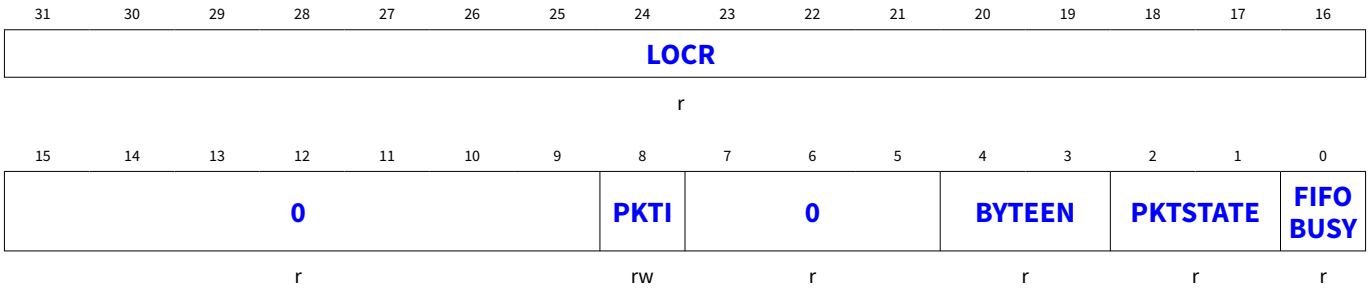
Field	Bits	Type	Description
FIFORDEN	10	rw	**FIFO Read Enable** When this bit is set, it enables the Read operation on the memory selected by FIFOSEL. If both FIFOWREN and FIFORDEN are set to 1 together, the behaviour is unpredictable.
FIFOWREN	11	rw	**FIFO Write Enable** When this bit is set, it enables the Write operation on the memory selected by FIFOSEL. If both FIFOWREN and FIFORDEN are set to 1 together, the behaviour is unpredictable.
FIFOSEL	13:12	rw	**FIFO Selected for Access** This field indicates the FIFO selected for debug access: - 00: Tx FIFO - 01: Descriptor cache (only when DBGMOD = 1) - 11: Rx FIFO
PKTIE	14	rw	**Receive Packet Available Interrupt Status Enable** When this bit is set, an interrupt is generated when EOP of received packet is written by the MAC to the Rx FIFO. This bit is applicable only when Slave mode is active.
EIEE	16	rw	**ECC Inject Error Enable** When set, enables the ECC error injection function in Debug mode. If EIAEE is set, enables the ECC Address Error Injection, otherwise, enables the ECC Data/Check Bits Error Injection (See EIM and BLEI field in MTL_DPP_ECC_EIC Register). When reset, disables the ECC error injection function.
EIAEE	17	rw	**ECC Inject Address Error Enable** When set, indicates that the ECC Debug error should be injected in Address Bits. When reset, indicates that the ECC Debug error should be injected in Data/Check Bits.
EIEC	18	rw	**ECC Inject Error Control** When EIEE bit of this register is set, errors are inserted in databus to memory based on the value encoded in this field. 0_B **M_1BIT**: Insert 1 bit error 1_B **M_2BIT**: Insert 2 bit errors
0	4, 7, 15, 31:19	r	**Reserved** Read as 0; should be written with 0.

14.6.503 Port j MTL Debug Status

The Debug Status register contains the state of the transactions to FIFO memory in Debug or Slave modes.
Note: Consecutive write or read to this register must be performed after at least 16 clock cycles of the slowest clock among MAC Transmitter/Receiver clock and Application/CSR clock.

Portj_MTL_Debug_Status (j=0-1) Offset address: $1100C_H+j*2000_H$

Port j MTL Debug Status Kernel Reset value: $0000\ 0018_H$



Field	Bits	Type	Description
FIFOBUSY	0	r	**FIFO Busy** When set, this bit indicates that a FIFO operation is in progress in the MAC and content of the following fields is not valid: - All other fields of this register - All fields of the MTL_FIFO_Debug_Data register
PKTSTATE	2:1	r	**Read Control Type** This field indicates the tag bits read along with the data being read from the selected FIFO. When the TxFIFO is read in Debug mode, this field marks the type of data read as follows: - 00: Packet Data - 01: Control Word - 10: SOP Data - 11: EOP Data When the RxFIFO is read in Debug or Slave mode, this field marks the type of data read as listed as follows: - 00: Packet Data - 01: Normal Status - 10: Last Status - 11: EOP
BYTEEN	4:3	r	**Read Data Byte Enable** This field indicates the number of data bytes valid in the MTL_FIFO_Debug_Data register during Read operation. This is valid only when PKTSTATE is 2'b11 (EOP). - 00: Byte 0 valid - 01: Byte 0 and Byte 1 are valid - 10: Byte 0, Byte 1, and Byte 2 are valid - 11: All four bytes are valid
PKTI	8	rw	**Receive Packet Available Interrupt Status** When set, this bit indicates that MAC layer has written the EOP of received packet to the Rx FIFO. This bit is reset when 1 is written to this bit.

(table continues...)

(continued)

Field	Bits	Type	Description
LOCR	31:16	r	**Remaining Locations in the FIFO** Slave Access Mode: This field indicates the space available in selected FIFO. Debug Access Mode: This field contains the Write or Read pointer value of the selected FIFO during Write or Read operation, respectively.
0	7:5, 15:9	r	**Reserved** Read as 0; should be written with 0.

14.6.504 Port j MTL FIFO Debug Data

The FIFO Debug Data register contains the data to be written to or read from the FIFOs. Note: Consecutive write or read to this register must be performed after at least 16 clock cycles of the slowest clock among MAC Transmitter/Receiver clock and Application/CSR clock.

Portj_MTL_FIFO_Debug_Data (j=0-1) Offset address: $11010_H+j*2000_H$

Port j MTL FIFO Debug Data Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FDBGDATA															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FDBGDATA															
rw															

Field	Bits	Type	Description
FDBGDATA	31:0	rw	**FIFO Debug Data** During debug or slave access write operation, this field contains the data to be written to the Tx FIFO, Rx FIFO, or Descriptor Cache memories. During debug or slave access read operation, this field contains the data read from the Tx FIFO, Rx FIFO, or Descriptor Cache memories.

14.6.505 Port j MTL Interrupt Status

The software driver (application) reads this register during interrupt service routine or polling to determine the interrupt status of MTL queues and the MAC.

Portj_MTL_Interrupt_Status (j=0-1) Offset address: $11020_H+j*2000_H$

Port j MTL Interrupt Status Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								MTLPIS	0		SGFIS	TINS	ESTIS	DBGIS	MACIS
r								r	r		r	r	r	r	r

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	Q7IS	Q6IS	Q5IS	Q4IS	Q3IS	Q2IS	Q1IS	Q0IS
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

Field	Bits	Type	Description
Q0IS	0	r	**Traffic Class/Queue 0 Interrupt Status** This bit indicates that there is an interrupt from Traffic Class/Queue 0. To reset this bit, the software must read Traffic Class/Queue 0 Interrupt Control and Status register to get the exact cause of the interrupt and clear its source.
Q1IS	1	r	**Traffic Class/Queue 1 Interrupt Status** This bit indicates that there is an interrupt from Traffic Class/Queue 1. To reset this bit, the software must read Traffic Class/Queue 1 Interrupt Control and Status register to get the exact cause of the interrupt and clear its source.
Q2IS	2	r	**Traffic Class/Queue 2 Interrupt Status** This bit indicates that there is an interrupt from Traffic Class/Queue 2. To reset this bit, the software must read Traffic Class/Queue 2 Interrupt Control and Status register to get the exact cause of the interrupt and clear its source.
Q3IS	3	r	**Traffic Class/Queue 3 Interrupt Status** This bit indicates that there is an interrupt from Traffic Class/Queue 3. To reset this bit, the software must read Traffic Class/Queue 3 Interrupt Control and Status register to get the exact cause of the interrupt and clear its source.
Q4IS	4	r	**Traffic Class/Queue 4 Interrupt Status** This bit indicates that there is an interrupt from Traffic Class/Queue 4. To reset this bit, the software must read Traffic Class/Queue 4 Interrupt Control and Status register to get the exact cause of the interrupt and clear its source.
Q5IS	5	r	**Traffic Class/Queue 5 Interrupt Status** This bit indicates that there is an interrupt from traffic Class/Queue 5. To reset this bit, the software must read traffic Class/Queue 5 Interrupt Control and Status register to get the exact cause of the interrupt and clear its source.
Q6IS	6	r	**Traffic Class/Queue 6 Interrupt Status** This bit indicates that there is an interrupt from Traffic Class/Queue 6. To reset this bit, the software must read Traffic Class/Queue 6 Interrupt Control and Status register to get the exact cause of the interrupt and clear its source.

(table continues...)

(continued)

Field	Bits	Type	Description
Q7IS	7	r	**Traffic Class/Queue 7 Interrupt Status** This bit indicates that there is an interrupt from Traffic Class/Queue 7. To reset this bit, the software must read Traffic Class/Queue 7 Interrupt Control and Status register to get the exact cause of the interrupt and clear its source.
MACIS	16	r	**MAC Interrupt Status** This bit indicates an interrupt event in the MAC. To reset this bit to 1'b0, the software must read the corresponding MAC_Interrupt_Status in the MAC to get the exact cause of the interrupt and clear its source.
DBGIS	17	r	**Debug Interrupt Status** This bit indicates an interrupt event during the slave access. To reset this bit, the software must read the MTL_Debug_Status register to get the exact cause of the interrupt and clear its source.
ESTIS	18	r	**EST (TAS- IEEE 802.1Qbv) Interrupt Status** This bit indicates an interrupt event during the operation of Time Aware Shaper for transmit traffic. To reset this bit, the application must clear the error/event that has caused the interrupt. The status bits that are set in MTL_EST_Status registers should be written to 1, to clear the ESTIS bit 0_B **INACTIVE**: EST (TAS- 802.1Qbv) Interrupt status not activated 1_B **ACTIVE**: EST (TAS- 802.1Qbv) Interrupt event activated
TINS	19	r	**TBS Interrupt Status** This bit is set when a frame is dropped in any of the TX queues in TBS mode due to the expiry of Launch time or transmission window. Access restriction applies. Clears on write of 1. 0_B **INACTIVE**: TBS Interrupt status not activated 1_B **ACTIVE**: TBS Interrupt event activated
SGFIS	20	r	**SGF Interrupt Status** This bit indicates an interrupt event during the operation of SGF. To reset this bit, the application must clear the source of the error/event that has caused the interrupt. The status bits that are set in MTL_SGF_Status registers should be written to 1, to clear the SGFIS bit. 0_B **INACTIVE**: SGF Interrupt status not activated 1_B **ACTIVE**: SGF Interrupt event activated
MTLPIS	23	r	**MTL Rx-Parser Interrupt Status** This bit indicates that there is an interrupt from Rx Parser Block. To reset this bit, the application must read the MTL_RXP_Interrupt_Control_Status register to get the exact cause of the interrupt and clear its source. 0_B **INACTIVE**: MTL Rx Parser Interrupt status not activated 1_B **ACTIVE**: MTL Rx Parser Interrupt event activated

(table continues...)

(continued)

Field	Bits	Type	Description
0	8, 9, 10, 11, 12, 13, 14, 15, 22:21, 31:24	r	**Reserved** Read as 0; should be written with 0.

14.6.506 Port j MTL RxQ DMA Map0

The Receive Queue and DMA Channel Mapping 0 register controls the static or dynamic mapping of Receive Queues 0-3 to Receive DMA Channels.

Portj_MTL_RxQ_DMA_Map0 (j=0-1) Offset address: $11030_H+j*2000_H$

Port j MTL RxQ DMA Map0 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Q3DDMACH	0				Q3MDMACH			Q2DDMACH	0				Q2MDMACH		
rw	r				rw			rw	r				rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Q1DDMACH	0				Q1MDMACH			Q0DDMACH	0				Q0MDMACH		
rw	r				rw			rw	r				rw		

Field	Bits	Type	Description
Q0MDMACH	2:0	rw	**Receive Queue 0 Mapped to DMA Channel** This field controls the routing of the packet received in Receive Queue 0 to the DMA channel. This field is valid when the Q0DDMACH field is reset. The encoding is the same as Q3MDMACH of this register.
Q0DDMACH	7	rw	**Receive Queue 0 Enabled for Dynamic DMA Channel Selection** When set, this bit indicates that each packet received in Receive Queue 0 is routed to a DMA channel as decided in the MAC Receiver based on the DMA channel number programmed in the L3-L4 filter registers, the Ethernet DA address or VLAN filter registers. When reset, this bit indicates that all packets received in Receive Queue 0 are routed to the DMA Channel programmed in the Q0MDMACH field.

(table continues...)

(continued)

Field	Bits	Type	Description
Q1MDMACH	10:8	rw	**Receive Queue 1 Mapped to DMA Channel** This field controls the routing of the received packet in Receive Queue 1 to the DMA channel. This field is valid when the Q1DDMACH field is reset. The encoding is the same as Q3MDMACH of this register.
Q1DDMACH	15	rw	**Receive Queue 1 Enabled for Dynamic DMA Channel Selection** When set, this bit indicates that each packet received in Receive Queue 1 is routed to a DMA channel as decided in the MAC Receiver based on the DMA channel number programmed in the L3-L4 filter registers, the Ethernet DA address registers or VLAN filter registers. When reset, this bit indicates that all packets received in Receive Queue 1 are routed to the DMA Channel programmed in the Q1MDMACH field.
Q2MDMACH	18:16	rw	**Receive Queue 2 Mapped to DMA Channel** This field controls the routing of the received packet in Receive Queue 2 to the DMA channel. This field is valid when the Q2DDMACH field is reset. The encoding is the same as Q3MDMACH of this register.
Q2DDMACH	23	rw	**Receive Queue 2 Enabled for Dynamic MA Channel Selection** When set, this bit indicates that each packet received in Receive Queue 2 is routed to a DMA channel as decided in the MAC Receiver based on the DMA channel number programmed in the L3-L4 filter registers, the Ethernet DA address registers or VLAN filter registers. When reset, this bit indicates that all packets received in Receive Queue 2 are routed to the DMA Channel programmed in the Q2MDMACH field.
Q3MDMACH	26:24	rw	**Receive Queue 3 Mapped to DMA Channel** This field controls the routing of the received packet in Receive Queue 3 to the DMA channel. This field is valid when the Q3DDMACH field is reset. - 4'b0000: DMA Channel 0 - 4'b0001: DMA Channel 1 - 4'b0010: DMA Channel 2 - 4'b0011: DMA Channel 3 - 4'b0100: DMA Channel 4 - 4'b0101: DMA Channel 5 - 4'b0110: DMA Channel 6 - 4'b0111: DMA Channel 7
Q3DDMACH	31	rw	**Receive Queue 3 Enabled for Dynamic DMA Channel Selection** When set, each packet received in Receive Queue 3 is routed to a DMA channel as decided in the MAC Receiver based on the DMA channel number programmed in the L3-L4 filter registers, the Ethernet DA address registers or VLAN filter registers. When reset, this bit indicates that all packets received in Receive Queue 3 are routed to the DMA Channel programmed in the Q3MDMACH field.
0	6:3, 14:11, 22:19, 30:27	r	**Reserved** Read as 0; should be written with 0.

14.6.507 Port j MTL RxQ DMA Map1

The Receive Queue and DMA Channel Mapping 1 register controls the static or dynamic mapping of Receive Queues 4-7 to Receive DMA Channels.

Portj_MTL_RxQ_DMA_Map1 (j=0-1) Offset address: $11034_H+j*2000_H$

Port j MTL RxQ DMA Map1 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Q7DDMACH	0				Q7MDMACH			Q6DDMACH	0				Q6MDMACH		
rw	r				rw			rw	r				rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Q5DDMACH	0				Q5MDMACH			Q4DDMACH	0				Q4MDMACH		
rw	r				rw			rw	r				rw		

Field	Bits	Type	Description
Q4MDMACH	2:0	rw	**Receive Queue 4 Mapped to DMA Channel** This field controls the routing of the packet received in Receive Queue 4 to the DMA channel. This field is valid when the Q4DDMACH field is reset. The encoding is the same as Q3MDMACH of MTL_RxQ_DMA_Map0 register.
Q4DDMACH	7	rw	**Receive Queue 4 Enabled for Dynamic DMA Channel Selection** When set, this bit indicates that each packet received in Receive Queue 4 is routed to a DMA channel as decided in the MAC Receiver based on the DMA channel number programmed in the L3-L4 filter registers, the Ethernet DA address or VLAN filter registers. When reset, this bit indicates that all packets received in Receive Queue 4 are routed to the DMA Channel programmed in the Q4MDMACH field.
Q5MDMACH	10:8	rw	**Receive Queue 5 Mapped to DMA Channel** This field controls the routing of the received packet in Receive Queue 5 to the DMA channel. This field is valid when the Q5DDMACH field is reset. The encoding is the same as Q3MDMACH of MTL_RxQ_DMA_Map0 register.
Q5DDMACH	15	rw	**Receive Queue 5 Enabled for Dynamic DMA Channel Selection** When set, this bit indicates that each packet received in Receive Queue 5 is routed to a DMA channel as decided in the MAC Receiver based on the DMA channel number programmed in the L3-L4 filter registers, the Ethernet DA address registers or VLAN filter registers. When reset, this bit indicates that all packets received in Receive Queue 5 are routed to the DMA Channel programmed in the Q5MDMACH field.
Q6MDMACH	18:16	rw	**Receive Queue 6 Mapped to DMA Channel** This field controls the routing of the received packet in Receive Queue 6 to the DMA channel. This field is valid when the Q6DDMACH field is reset. The encoding is the same as Q3MDMACH of MTL_RxQ_DMA_Map0 register.

(table continues...)

(continued)

Field	Bits	Type	Description
Q6DDMACH	23	rw	**Receive Queue 6 Enabled for Dynamic MA Channel Selection** When set, this bit indicates that each packet received in Receive Queue 6 is routed to a DMA channel as decided in the MAC Receiver based on the DMA channel number programmed in the L3-L4 filter registers, the Ethernet DA address registers or VLAN filter registers. When reset, this bit indicates that all packets received in Receive Queue 6 are routed to the DMA Channel programmed in the Q6MDMACH field.
Q7MDMACH	26:24	rw	**Receive Queue 7 Mapped to DMA Channel** This field controls the routing of the received packet in Receive Queue 7 to the DMA channel. This field is valid when the Q7DDMACH field is reset. The encoding is the same as Q3MDMACH of MTL_RxQ_DMA_Map0 register. The width of this field depends on the number of Receive DMA channels, therefore all values might not be valid in certain configurations. For example, if the number of Rx DMA channels selected is 2, the only valid values are 0 and 1. Other values are reserved.
Q7DDMACH	31	rw	**Receive Queue 7 Enabled for Dynamic DMA Channel Selection** When set, this bit indicates that each packet received in Receive Queue 7 is routed to a DMA channel as decided in the MAC Receiver based on the DMA channel number programmed in the L3-L4 filter registers, the Ethernet DA address registers or VLAN filter registers. When reset, this bit indicates that all packets received in Receive Queue 7 are routed to the DMA Channel programmed in the Q7MDMACH field.
0	6:3, 14:11, 22:19, 30:27	r	**Reserved** Read as 0; should be written with 0.

14.6.508 Port j MTL TBS CTRL

This register controls the operation of Time Based Scheduling.

Portj_MTL_TBS_CTRL (j=0-1) Offset address: $11048_H+j*2000_H$

Port j MTL TBS CTRL Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LEOS															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LEOS								0	LEGOS			0	TIEN	LEOV	ESTM
rw								r	rw			r	rw	rw	rw

Field	Bits	Type	Description
ESTM	0	rw	**EST offset Mode** When this bit is set, the Launch Time value used in Time Based Scheduling is interpreted as an EST offset value and is added to the Base Time Register (BTR) of the current list. When reset, the Launch Time value is used as an absolute value that must be compared with the System time [39:8]. 0_B **DISABLE**: EST offset Mode is disabled 1_B **ENABLE**: EST offset Mode is enabled
LEOV	1	rw	**Launch Expiry Offset Valid** When set indicates the LEOS field is valid. When not set, indicates the Launch Expiry Offset is not valid and the MTL must not check for Launch expiry time. 0_B **INVALID**: LEOS field is invalid 1_B **VALID**: LEOS field is valid
TIEN	2	rw	**TBS Interrupt Enable** When set indicates that TBS interrupt (occurs due to frame drop) is enabled. Value valid only when TIEN is set. 0_B **INVALID**: TBS interrupt is enabled 1_B **VALID**: TBS interrupt is disabled
LEGOS	6:4	rw	**Launch Expiry GSN Offset** The number of GSN slots that has to be added to the Launch GSN to compute the Launch Expiry time. Value valid only when LEOV is set.
LEOS	31:8	rw	**Launch Expiry Offset** The value in units of 256 nanoseconds that has to be added to the Launch time to compute the Launch Expiry time. A value of 0 means 256ns, 1 means 512ns and so on. Value valid only when LEOV is set. Max value: 999,999,999 ns, additionally must be smaller than CTR-1 value when ESTM mode is set, as this value is a modulo CTR value.
0	3, 7	r	**Reserved** Read as 0; should be written with 0.

14.6.509 Port j MTL TBS STATS

This register provides the One Hot encoded Queue numbers that have the frame drop related error.

Portj_MTL_TBS_STATS (j=0-1) Offset address: $1104C_H+j*2000_H$

Port j MTL TBS STATS Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EDQN							
r								rw							

Field	Bits	Type	Description
EDQN	7:0	rw	**Expiry drop Queue number** EDQN can have multiple bits set corresponding to the Queue numbers that have experienced error described in TINS field of MTL_Status register. Access restriction applies. Self-set to 1 on internal event. Clears on write of 1.
0	31:8	r	**Reserved** Read as 0; should be written with 0.

14.6.510 Port j MTL EST Control

This register controls the operation of Time Aware Shaper conforming to Enhancements to Scheduled Transmission (IEEE 802.1Qbv) standard.

Portj_MTL_EST_Control (j=0-1) Offset address: $11050_H+j*2000_H$

Port j MTL EST Control Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PTOV									CTOV						
rw									rw						

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CTOV					TILS			LCSE		DFBS	DDBF	QHLBF	0	SSWL	EEST
rw					rw			rw		rw	rw	rw	r	rw	rw

Field	Bits	Type	Description
EEST	0	rw	**Enable EST** When this bit is set, the Time Aware Shaper algorithm gets activated in the Transmit scheduler. During the toggle from 0 to 1, the gate control list processing starts only after the SSWL bit is set. When this bit is reset, the EST function is disabled and all gates of teh GCL table are considered to be in Open state. If an uncorrectable error is detected in the GCL memory, DWC_xgmac resets this bit and disables the EST function. 0_B **DISABLE**: EST is disabled 1_B **ENABLE**: EST is enabled

(table continues...)

(continued)

Field	Bits	Type	Description
SSWL	1	rw	**Switch to software owned list** When this bit is set, the DWC_xgmac is instructed to switch to the new GCL list (SWOL) programmed in the GCL memory, at the start of new BTR. Hardware clears this bit when the switch to the SWOL is completed or an non-recoverable BTR error (BTRL = "11111111" indicates the iterations exceeded the value of 512 and DWC_xgmac was not able to update New BTR to be equal to or greater than Current Time.) is set. When BTRE is set this bit is cleared but SWOL is not updated as the switch was not successful. The write to this bit is effective only when EEST bit is set to 1 along with it or EEST bit is previously set to 1. Access restriction applies. Writing 1 sets the bit. Self-cleared. Writing 0 has no effect. 0_B **DISABLE**: Switch to S/W owned list is disabled 1_B **ENABLE**: Switch to S/W owned list is enabled
QHLBF	3	rw	**Quick Assertion of HLBF Error** When set, HLBF error is reported after the packet is not transmitted for 1 to 2 iterations of GCL list. When this bit is reset, the HLBF error is reported after the packet is not transmitted for 2 to 3 iterations of GCL list. 0_B **DISABLE**: Disable Quick assertion of HLBF error 1_B **ENABLE**: Quick Assertion of HLBF Error
DDBF	4	rw	**Do not Drop frames due to HLBF error** When this bit is set, frames that are not scheduled for transmission because its size is larger than the Gate Open time window, are not dropped. This can result in head-of-line blocking in the corresponding TX Queue. When this bit is reset, frames that do not get scheduled for transmission due to HLBF error after the completion of the number of GCL Table iterations as set in QHLBF. 0_B **DROP**: Drop frames due to HLBF error 1_B **DONT_DROP**: Do not Drop frames due to HLBF error
DFBS	5	rw	**Drop Frames causing Scheduling Error** When this bit is set, frames that do not get scheduled for transmission after the number of GCL iterations as set in LCSE, are dropped. When this bit is reset, frames are not dropped due to any TAS scheduling conflicts or delays (HLBS). This can create HOL (head-of-line) blocking in the transmit queue. 0_B **DONT_DROP**: Do not Drop Frames causing Scheduling Error 1_B **DROP**: Drop Frames causing Scheduling Error

(table continues...)

(continued)

Field	Bits	Type	Description
LCSE	7:6	rw	**Loop Count to report Scheduling Error** Programmable number of GCL list iterations before reporting an HLBS error defined in EST_Status register. 00_B **M_4_ITERNS**: 4 iterations 01_B **M_8_ITERNS**: 8 iterations 10_B **M_16_ITERNS**: 16 iterations 11_B **M_32_ITERNS**: 32 iterations
TILS	10:8	rw	**Time Interval Left Shift Amount** This field provides the left shift amount for the programmed Time Interval values used in the Gate Control Lists. - 000: No left shift needed (equal to x1ns) - 001: Left shift TI by 1 bit (equal to x2ns) - 010: Left shift TI by 2 bits (equal to x4ns) - ... - 111: Left shift TI by 7 bits (equal to x128ns) Based on the configuration, one or more bits of this field must be treated as Reserved/Read-Only.
CTOV	22:11	rw	**Current Time Offset Value** Provides a 12 bit time offset value in nanosecond that is added to the current time to compensate for all the implementation pipeline delays such as the CDC sync delay, buffering delays, data path delays, and so on. This offset helps to ensure that the impact of gate controls is visible on the line exactly at the pre-determined schedule (or as close to the schedule as possible).
PTOV	31:23	rw	**PTP Time Offset Value** The value of PTP Clock period multiplied by 9 in nanoseconds. This value is needed to avoid transmission overruns at the beginning of the installation of a new GCL.
0	2	r	**Reserved** Read as 0; should be written with 0.

14.6.511 Port j MTL EST Overhead

This register indicates the value of Overhead in bytes to be considered for EST Scheduling.

Portj_MTL_EST_Overhead (j=0-1) Offset address: $11054_H+j*2000_H$

Port j MTL EST Overhead Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0											ABYT				
r											rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								OVHD							
r								rw							

Field	Bits	Type	Description
OVHD	7:0	rw	**Overhead Bytes Value** This field should be programmed with the delay in bytes between the point at which a packet is selected for transmission by the EST scheduler and time at which the Start of Packet appears on the XGMII/ GMII TX interface. This value differs based on the operating speed and configuration. The value to be programmed is provided in Table "Overheads to be Accounted in the EST Overhead Field".
ABYT	20:16	rw	**Bytes transmitted in 1 application clock** This field should be programmed with the number of bytes transmitted on XGMII/GMII interface in the time window corresponding to 1 application clock period. For Example: if Time period of Application clock is 6.4 ns and speed mode is XGMII 10G, the programmed value must be, ((6.4)/(3.2)* (Number of Bytes in 1 Tx clock =4) = 8. (for values greater 0.5, round it off to 1)
0	15:8, 31:21	r	**Reserved** Read as 0; should be written with 0.

14.6.512 Port j MTL EST Status

This register reports the events related to Enhancements to Scheduled Transmission (IEEE 802.1Qbv).

Portj_MTL_EST_Status (j=0-1) Offset address: $11058_H+j*2000_H$

Port j MTL EST Status Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0												CGSN			
r												r			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BTRL								SWOL	0		CGCE	HLBS	HLBF	BTRE	SWLC
r								r	r		rw	r	r	rw	rw

Field	Bits	Type	Description
SWLC	0	rw	**Switch to Software Owned List Complete** When this field is set, indicates that DWC_xgmac has successfully switched to the SWOL, and the SWOL bit is updated to that effect. This field is cleared when the SSWL field of EST_Control register transitions from 0 to 1, or when a 1 is written to this bit. Access restriction applies. Self-set to 1 on internal event. Clears on write of 1. Writing 0 has no effect. 0_B **INACTIVE**: Switch to S/W owned list Complete not detected 1_B **ACTIVE**: Switch to S/W owned list Complete detected

(table continues...)

(continued)

Field	Bits	Type	Description
BTRE	1	rw	**BTR Error** When this field is set, indicates a programming error in the BTR of SWOL where the programmed value is less than current time. If the BTRL = "11111111", SWOL is not updated and software must reprogram the BTR to a value greater than current time and set the SSWL bit to re-initiate the switch to SWOL. Access restriction applies. Self-set to 1 on internal event. Self clear or clears on write of 1. Writing 0 has no effect. 0_B **INACTIVE**: BTR Error not detected 1_B **ACTIVE**: BTR Error detected
HLBF	2	r	**Head-Of-Line Blocking due to Frame Size** This field is set when HOL Blocking is noticed on one or more Queues when none of the Time Intervals of open-gates in the GCL is greater than or equal to the duration needed for frame (or frame fragment size when preemption is enabled) transmission. The one-hot encoded Queue numbers that are experiencing HLBF are indicated in the MTL_EST_Frm_Size_Error register. Additionally, the first Queue number that experienced HLBF along with the frame size is captured in MTL_EST_Frm_Size_Capture register. Bit cleared when MTL_EST_Frame_Size_Error register is all zeros. 0_B **INACTIVE**: Head-Of-Line Blocking due to Frame Size not detected 1_B **ACTIVE**: Head-Of-Line Blocking due to Frame Size detected
HLBS	3	r	**Head-Of-Line Blocking due to Scheduling** When set, it indicates that a packet is not able to get scheduled by EST even after LCSE number of GCL iterations has elapsed. Indicates to software a potential programming error. The one-hot encoded values of the Queue Numbers that have such packets blocking the queue, are indicated in the MTL_EST_Sch_Error register. Bit cleared when MTL_EST_Sch_Error register is all zeros. Note: A single fixed reference or tic is generated at integral multiple of MTL_EST_Control.LCSE loops/iterations. The HLBS is generated after (MTL_EST_Control.LCSE + N) loops/iterations, where N is non-integral number of MTL_EST_Control.LCSE loops/iterations the packet could not be scheduled before the last 2 tics. 0_B **INACTIVE**: Head-Of-Line Blocking due to Scheduling not detected 1_B **ACTIVE**: Head-Of-Line Blocking due to Scheduling detected
CGCE	4	rw	**Constant Gate Control Error** This error occurs when the programmed Time Interval (TI) value after the optional Left Shifting is more than or equal to the Cycle Time (CTR). Programming a GCL row's time interval TI greater than CTR is equivalent of always closing or opening of certain Transmit queues which can be done by disabling or enabling the required open or closed queues without enabling EST. Due to this reason, such a programming is treated as Error. Access restriction applies. Self-set to 1 on internal event. Clears on write of 1. Writing 0 has no effect. 0_B **INACTIVE**: Constant Gate Control Error not detected 1_B **ACTIVE**: Constant Gate Control Error detected

(table continues...)

(continued)

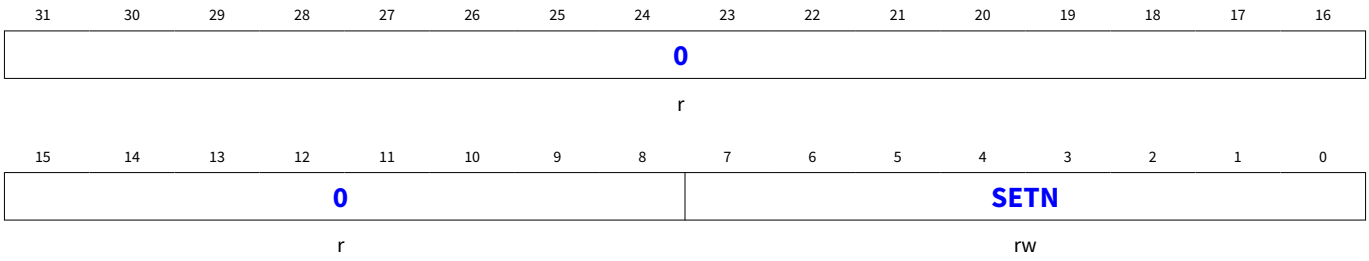
Field	Bits	Type	Description
SWOL	7	r	**Software Owned List** When this field is 0, indicates that the Gate control list number 0 is owned by software. When this field is 1, indicates that the Gate Control list 1 is owned by the software. Any reads/writes by the software (using indirect access through GCL_Control) is directed to the list indicated by this value. The inverse of this value is treated as HWOL. R/W operations performed by DWC_xgmac are directed to the list pointed by HWOL. Values: - 0x0: Gate control list number "0" is owned by software - 0x1: Gate control list number "1" is owned by software
BTRL	15:8	r	**BTR Loop Count** This field provides the number of GCL cycle times after which the switch to new SWOL GCL occurs after SSWL is set. It is determined by the value N which satisfies the equation "Current Time is less than or equal to New BTR + (N * 4 * New Cycle Time)". N = "11111111" indicates that the iterations exceeded the value of 512 and DWC_xgmac is unable to update New BTR to be equal to or greater than Current Time. Software intervention is needed to update the New BTR. This field is cleared when BTRE field of this register is cleared.
CGSN	19:16	r	**Current GCL Slot Number** Indicates the slot number of the GCL list. Slot number is a modulo 16 count of the GCL List loops executed so far. Even if a new GCL list is installed, the count is incremental.
0	6:5, 31:20	r	**Reserved** Read as 0; should be written with 0.

14.6.513 Port j MTL EST Sch Error

This register provides the Traffic Class numbers that have the GCL scheduling related error.

Portj_MTL_EST_Sch_Error (j=0-1) Offset address: $11060_H+j*2000_H$

Port j MTL EST Sch Error Kernel Reset value: $0000\ 0000_H$



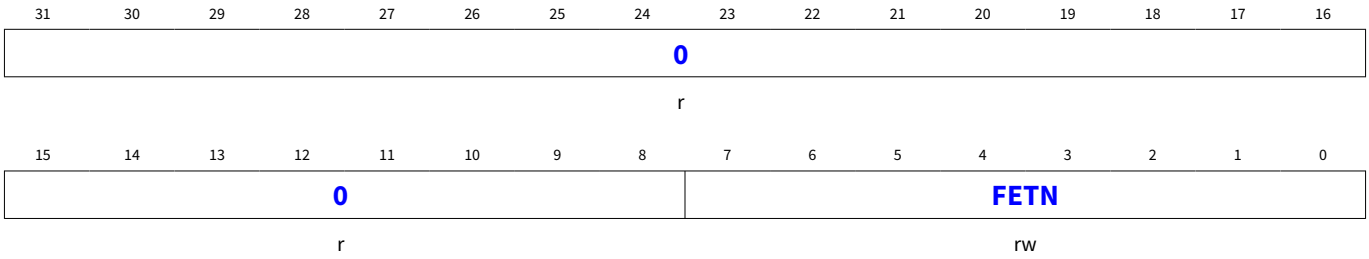
Field	Bits	Type	Description
SETN	7:0	rw	**Schedule Error TC Number** SETN can have multiple bits set corresponding to the TC numbers that have experienced error described in HLBS field of status register. Bit 0 corresponds to TC0, Bit 1 corresponds to TC1, and so on. Access restriction applies. Self-set to 1 on internal event. Clears on write of 1. Writing 0 has no effect.
0	31:8	r	**Reserved** Read as 0; should be written with 0.

14.6.514 Port j MTL EST Frm Size Error

This register provides the Traffic Class numbers that have the Frame Size related scheduling error.

Portj_MTL_EST_Frm_Size_Error (j=0-1) Offset address: $11064_H+j*2000_H$

Port j MTL EST Frm Size Error Kernel Reset value: $0000\ 0000_H$



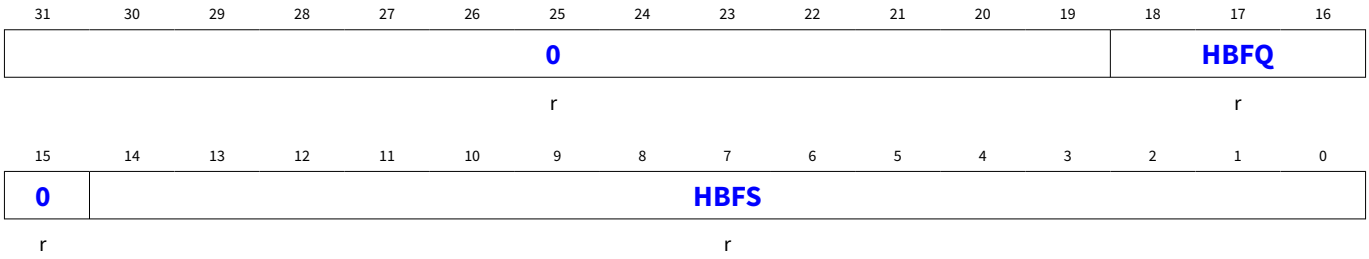
Field	Bits	Type	Description
FETN	7:0	rw	**Frame Size Error TC Number** FETN can have multiple bits set corresponding to the TC numbers that have experienced error described in HLBF field of status register. Bit 0 corresponds to TC0, Bit 1 corresponds to TC1, and so on. Access restriction applies. Self-set to 1 on internal event. Clears on write of 1. Writing 0 has no effect.
0	31:8	r	**Reserved** Read as 0; should be written with 0.

14.6.515 Port j MTL EST Frm Size Capture

This register captures the Frame Size and Traffic Class Number of the first occurrence of the Frame Size related error. Upon clearing, this register captures the data of the next occurrence of a similar error.

Portj_MTL_EST_Frm_Size_Capture (j=0-1) Offset address: $11068_H+j*2000_H$

Port j MTL EST Frm Size Capture Kernel Reset value: $0000\ 0000_H$



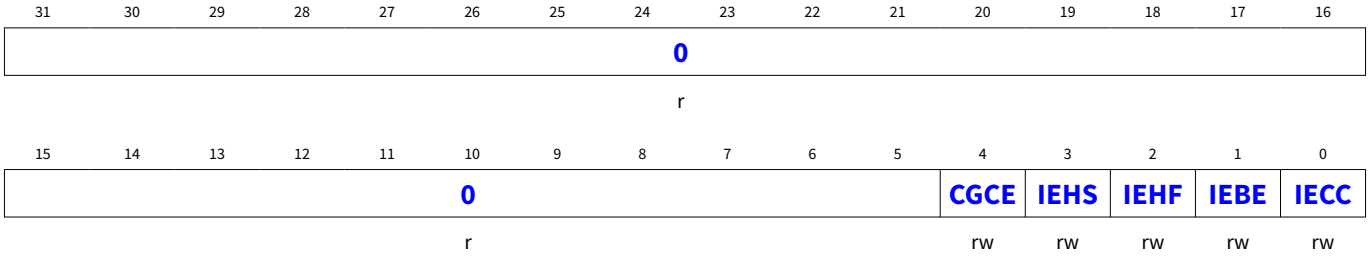
Field	Bits	Type	Description
HBFS	14:0	r	**Frame Size of HLBF** Captures the Frame Size of the frame present in the queue number indicated by HBFQ field of this register. Contents of this register must be considered invalid, if this field is zero. Cleared when MTL_EST_Frm_Size_Error register is all zeros.
HBFQ	18:16	r	**Transmit Queue Number of HLBF** Captures the binary value of the first Transmit Queue experiencing HLBF error (see HLBF field of MTL_EST_status register). Value once written is not altered by any subsequent queue errors of similar nature. Once cleared, the queue number of the next occurring HLBF error is captured. Cleared when MTL_EST_Frm_Size_Error register is all zeros.
0	15, 31:19	r	**Reserved** Read as 0; should be written with 0.

14.6.516 Port j MTL EST Intr Enable

This register implements the Interrupt Enable bits for the various events in the EST scheduler. Bit positions have a 1- to-1 correspondence with the status bit positions in MTL_EST_Status register.

Portj_MTL_EST_Intr_Enable (j=0-1) Offset address: $11070_H+j*2000_H$

Port j MTL EST Intr Enable Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
IECC	0	rw	**Interrupt Enable for Switch List** When set, generates interrupt when the GCL Table switch is successful. When reset, this event does not generate an interrupt. 0_B **DISABLE**: Interrupt for Switch List is disabled 1_B **ENABLE**: Interrupt for Switch List is enabled

(table continues...)

(continued)

Field	Bits	Type	Description
IEBE	1	rw	**Interrupt Enable for BTR Error** When set, generates interrupt when the BTR Error occurs and is indicated in the status. When reset, this event does not generate an interrupt. 0_B **DISABLE**: Interrupt for BTR Error is disabled 1_B **ENABLE**: Interrupt for BTR Error is enabled
IEHF	2	rw	**Interrupt Enable for HLBF** When set, generates interrupt when HLBS status bit is set. When reset, this event does not generate an interrupt. 0_B **DISABLE**: Interrupt for HLBF is disabled 1_B **ENABLE**: Interrupt for HLBF is enabled
IEHS	3	rw	**Interrupt Enable for HLBS** When set, generates interrupt when HLBS status bit is set. When reset, this event does not generate an interrupt. 0_B **DISABLE**: Interrupt for HLBS is disabled 1_B **ENABLE**: Interrupt for HLBS is enabled
CGCE	4	rw	**Interrupt Enable for CGCE** When set, generates interrupt when the Constant Gate Control Error occurs and is indicated in the status. When reset, this event does not generate an interrupt. 0_B **DISABLE**: Interrupt for CGCE is disabled 1_B **ENABLE**: Interrupt for CGCE is enabled
0	31:5	r	**Reserved** Read as 0; should be written with 0.

14.6.517 Port j MTL GCL Control

This register provides the control for reading/writing to the Gate Control lists.

Portj_MTL_GCL_Control (j=0-1) Offset address: $11080_H+j*2000_H$

Port j MTL GCL Control Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								GCLEIEC	GCLEIAEE	GCLEIEE	0				
r								rw	rw	rw	r				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR								0		DBGB	DBGM	GCLS	GCRR	R1W0	SRWO
rw								r		rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
SRWO	0	rw	**Start Read/Write Op - When this field is set, indicates that a Read/ Write operation has started and is in progress** - When this field is reset by DWC_xgmac, indicates that the R/W operation is complete or an error has occurred. - Reads: Data can be read from MTL_GCL_Data register after this bit is reset. - Writes: MTL_GCL_Data must be programmed with write data before setting SRWO bit. Access restriction applies. Writing 1 sets the bit. Self-cleared. Writing 0 has no effect. 0_B **DISABLE**: Start Read/Write Op disabled 1_B **ENABLE**: Start Read/Write Op enabled
R1W0	1	rw	**Read 1, Write 0: - When set to 1: Read Operation - When set to 0: Write Operation** 0_B **WRITE**: Write Operation 1_B **READ**: Read Operation
GCRR	2	rw	**Gate Control Related Registers** When this field is set to 1, indicates that the R/W access is for the GCL related registers (BTR, CTR, TER, LLR) whose address is provided by GCRA. When 0, indicates that R/W must be directed to GCL from the address provided by GCLA.
GCLS	3	rw	**Gate Control/Address Select** When set to "1" indicates the SGF R/W access for the GCL related registers/GCL memory. When set to "0" indicates the EST R/W access for the GCL related registers/GCL memory.
DBGM	4	rw	**Debug Mode** When this field is set to 1, indicates R/W in debug mode where the memory bank (for GCL and Time related registers) is explicitly provided by DBGB value. When set to 0, SWOL bit is used to determine which bank to access. 0_B **DISABLE**: Debug Mode is disabled 1_B **ENABLE**: Debug Mode is enabled
DBGB	5	rw	**Debug Mode Bank Select** When this field is set to 0, indicates R/W in debug mode must be directed to Bank 0 (GCL0 and corresponding Time related registers). When set to 1, indicates R/W in debug mode must be directed to Bank 1 (GCL1 and corresponding Time related registers). This value is used when DBGM is set and overrides the value of SWOL, which is normally used. 0_B **BANK0**: R/W in debug mode should be directed to Bank 0 1_B **BANK1**: R/W in debug mode should be directed to Bank 1

(table continues...)

(continued)

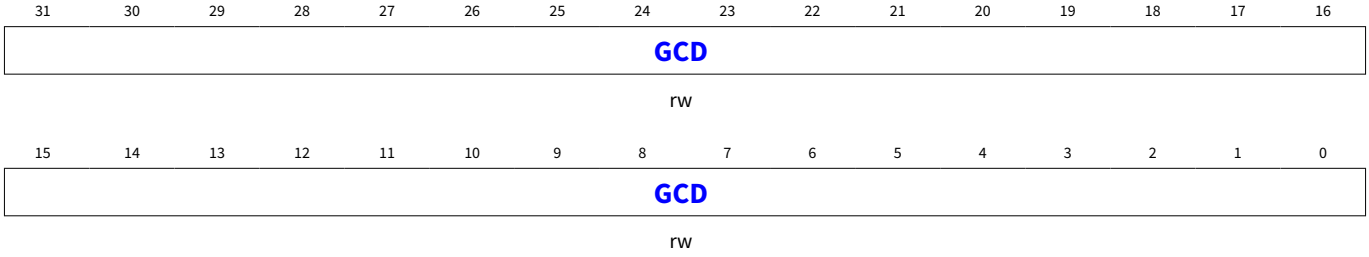
Field	Bits	Type	Description
ADDR	15:8	rw	**Gate Control List Address: (GCLA when GCRR is 0)** Provides the address (row number) of the Gate Control List at which the R/W operation has to be performed. By default the Gate Control List pointed by SWOL of MTL_EST_Status is selected for R/W. However, if the DBGM bit of this register is set, a debug mode access is given to R/W from DBGB. Gate Control list Related Registers Address: (GCRA when GCRR is 1). By default, the GCL related register set pointed by SWOL of MTL_EST_Status is selected for R/W. However, if the DBGM bit of this register is set, a debug mode access is given to R/W from DBGB. Only the lower 3 bits are used in this mode. Higher order bits are treated as dont care values. - 000: BTR Low (31:0) - 001: BTR High (63:31) - 010: CTR Low (31:0) - 011: CTR High (39:32) - 100: TER (30:0) - 101: LLR (8:0) - Others: Reserved Range Variable[x]:15
GCLEIEE	21	rw	**GCL ECC Inject Error Enable** When set along with EEST bit of MTL_EST_Control register, enables the ECC error injection feature. If GCLEIAEE is set, enables the ECC Address Error Injection, otherwise, error is injected in address bus, otherwise error is injected in Data/ECC bits to teh GCL memory. When reset, disables the ECC error injection feature. 0_B **DISABLE**: GCL ECC Inject Error is disabled 1_B **ENABLE**: GCL ECC Inject Error is enabled
GCLEIAEE	22	rw	**GCL ECC Inject Address Error** When this bit is set, error injection (when triggered by GCLEIEE) is done in Address bus to GCL memory. When this bit is reset, error is injected in the data bus to GCL memory.
GCLEIEC	23	rw	**ECC Inject Error Control for GCL Memory** When GCLEIEE bit of this register is set, ECC error is injected based on the value encoded in this field. 0_B **M_1BIT**: Insert 1 bit error 1_B **M_2BIT**: Insert 2 bit errors
0	7:6, 20:16, 31:24	r	**Reserved** Read as 0; should be written with 0.

14.6.518 Port j MTL GCL Data

This register holds the read data or write data in case of reads and writes respectively.

Portj_MTL_GCL_Data (j=0-1) Offset address: $11084_H+j*2000_H$

Port j MTL GCL Data Kernel Reset value: $0000\ 0000_H$



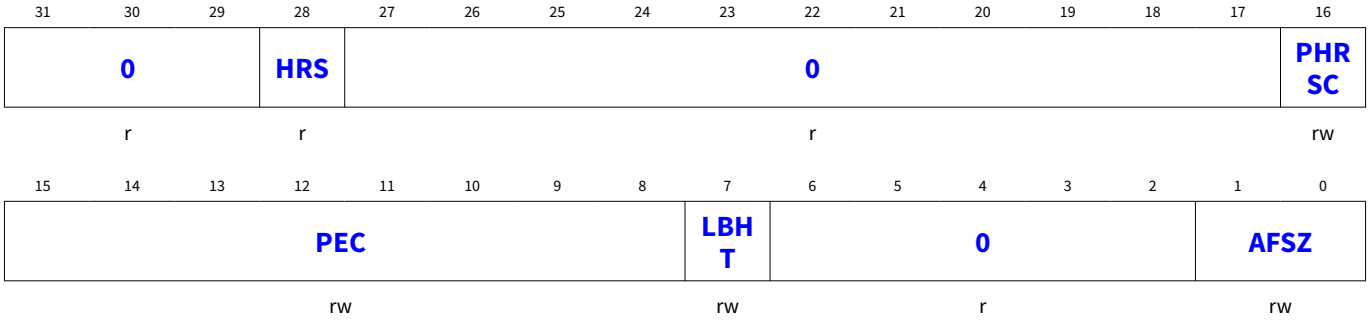
Field	Bits	Type	Description
GCD	31:0	rw	**Gate Control Data** The data corresponding to the address selected in the GCL_Control register. Used for both Read and Write operations.

14.6.519 Port j MTL FPE CTRL STS

This register controls the operation of, and provides status for Frame Preemption (IEEE 802.1Qbu/802.3br).

Portj_MTL_FPE_CTRL_STS (j=0-1) Offset address: $11090_H+j*2000_H$

Port j MTL FPE CTRL STS Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
AFSZ	1:0	rw	**Additional Fragment Size** Used to indicate, in units of 64 bytes, the minimum number of bytes over 64 bytes required in non-final fragments of preempted frames. The minimum non-final fragment size is (AFSZ +1) * 64 bytes
LBHT	7	rw	**Level Based Hold Transition** When set to 1, enables hold operation in second row of EST GCL during non-first loop even when hold request is set in first row, that is there is no release request to hold request transition. When set to 0, enables hold operation in second row of EST GCL only on transition from release request to hold request.

(table continues...)

(continued)

Field	Bits	Type	Description
PEC	15:8	rw	**Preemption Classification** This controls whether a traffic class is marked as preemptable or express. Each bit corresponds to a TC (Bit8 is TC0, Bit9 is TC1, and so on). When a bit is set, the corresponding TC is marked as preemptable. When bit is 0, that traffic class is marked as express packets. When both EST (Qbv) and Preemption are enabled, TC0 is always assumed to be preemptable. When EST (Qbv) is enabled, traffic class categorized as preemptable here are always assumed to be in "Open" state in the Gate Control List.
PHRSC	16	rw	**Preemption MAC Hold/Release Sideband Control Select - 1: Indicates that Set-Hold-and-Release-MAC operation is controlled by Sideband Signal sbd_pmac_hold_i** - 0: Indicates that Set-Hold-and-Release-MAC operation is controlled by IEEE 802.1Qbv Enhancement to Scheduled Traffic (EST) Gate Control List (GCL) entries.
HRS	28	r	**Hold/Release Status - 1: Indicates a Set-and-Hold-MAC operation was executed and the pMAC is in Hold State** - 0: Indicates a Set-and-Release-MAC operation was executed and the pMAC is in Release State. 0_B **SET_REL**: Indicates a Set-and-Release-MAC operation was executed and the pMAC is in Release State 1_B **SET_HOLD**: Indicates a Set-and-Hold-MAC operation was executed and the pMAC is in Hold State
0	6:2, 27:17, 31:29	r	**Reserved** Read as 0; should be written with 0.

14.6.520 Port j MTL FPE Advance

This register holds the Hold and Release Advance time.

Portj_MTL_FPE_Advance (j=0-1) Offset address: $11094_H+j*2000_H$

Port j MTL FPE Advance Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RADV															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HADV															
rw															

Field	Bits	Type	Description
HADV	15:0	rw	**Hold Advance** The maximum time in nanoseconds that can elapse between issuing a HOLD to the MAC and the MAC ceasing to transmit any preemptable frame that is in the process of transmission or any preemptable frames that are queued for transmission.
RADV	31:16	rw	**Release Advance** The maximum time in nanoseconds that can elapse between issuing a RELEASE to the MAC and the MAC being ready to resume transmission of preemptable frames, in the absence of there being any express frames available for transmission.

14.6.521 Port j MTL RXP Control Status

This is the Control and Status register for Flexible Receive Parser.

Portj_MTL_RXP_Control_Status (j=0-1) Offset address: $110A0_H+j*2000_H$

Port j MTL RXP Control Status Kernel Reset value: $40FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RXPI	PIPE			0			0	NPE							
r	r			r			r	rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	FPE_RCH			OKI_DME	0	0	0	NVE							
r	rw			rw	r	r	r	rw							

Field	Bits	Type	Description
NVE	7:0	rw	**Number Valid Entries in the Instruction table** This field indicates the last valid entry in the Instruction Table. This is used by the Rx-Parser logic to detect programming errors. While parsing the table, if the address becomes greater than this entry, NVEOVIS bit in the MTL_RXP_Interrupt_Control_Status register is set. Note: - Value of 0 implies, 1st Instruction; a value of 255 implies 256th instruction. - Program this field with a value between 0 and 255. Read the value after programming this field.
OKI_DME	11	rw	**Instruction's OK Index Dual Meaning Enable** When this bit is set to 1, the OKI field in the Instruction can be interpreted also as Policing Counter Number/Valid or Gate ID/Valid when SGF or PC functions are enabled. For more detail refer respective feature section. When this bit is reset, the OKI field interpretation is always "OK Index".

(table continues...)

(continued)

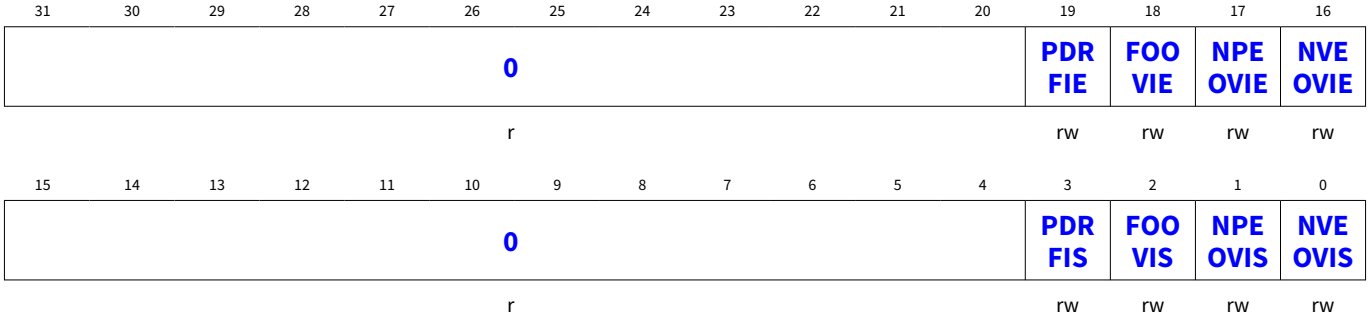
Field	Bits	Type	Description
FPE_RCH	14:12	rw	**Frame Preemption Residue Receive DMA Channel** This field holds the Rx Channel number to which the residual preemption frames must be forwarded. Preemption frames which pass Rx Parser filtering, are routed based on Instruction Table. All other frames are treated as residual frames and are routed to the channel number mentioned in this field. Channel-0 is used as a default Channel for express residual frames, so this field must not be programmed to a value 0.
NPE	23:16	rw	**Number of Parsable Entries in the Instruction Table** This field indicates the number of parsable entries in the Instruction Table. This field is used in Rx-Parser logic to detect programming error. If the number of parsed entries for a packet is more than this value, NPEOVIS bit in the MTL_RXP_Interrupt_Control_Status register is set. Note: - A value of 0 value implies 1 Instruction. Value of 255 implies 256 instructions. - Program this field with a value between 0 and 255.Read the value after programming this field.
PIPE	30:28	r	**Number of Parallel Instruction Processor Engines** This field is read-only.
RXPI	31	r	**Rx Parser is Active** This status bit is set to 1 when the Rx parser is in Active state. This bit can be used as a handshake with the software when the parser gets disabled/enabled. 0_B **INACTIVE**: Rx Parser is not Active 1_B **ACTIVE**: Rx Parser is Active
0	8, 9, 10, 15, 24, 27:25	r	**Reserved** Read as 0; should be written with 0.

14.6.522 Port j MTL RXP Interrupt Control Status

This is Interrupt Control and Status register for Flexible Receive Parser.

Portj_MTL_RXP_Interrupt_Control_Status (j=0-1) Offset address: $110A4_H+j*2000_H$

Port j MTL RXP Interrupt Control Status Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
NVEOVIS	0	rw	**Number of Valid Entries Overflow Interrupt Status** While parsing, if the instruction address is found to be more than NVE (Number of Valid Entries in MTL_RXP_Control register), this bit is set to 1. This bit is cleared when the software writes 1 to this bit. Access restriction applies. Self-set to 1 on internal event. Clears on write of 1. Writing 0 has no effect. 0_B **INACTIVE**: Number of Valid Entries Overflow Interrupt Status not detected 1_B **ACTIVE**: Number of Valid Entries Overflow Interrupt Status detected
NPEOVIS	1	rw	**Number of Parsable Entries Overflow Interrupt Status** While parsing, if the number of parsed entries is found to be more than NPE[] (Number of Parsable Entries in MTL_RXP_Control register), this bit is set to 1. This bit is cleared when the application writes 1 to this bit. Access restriction applies. Self-set to 1 on internal event. Clears on write of 1. Writing 0 has no effect. 0_B **INACTIVE**: Number of Parsable Entries Overflow Interrupt Status not detected 1_B **ACTIVE**: Number of Parsable Entries Overflow Interrupt Status detected
FOOVIS	2	rw	**Frame Offset Overflow Interrupt Status** During parsing, if the Instruction table entry's 'Frame Offset' is found to be more than EOF offset, this bit is set. This bit is cleared when the software writes 1 to this bit. Access restriction applies. Self-set to 1 on internal event. Clears on write of 1. Writing 0 has no effect. 0_B **INACTIVE**: Frame Offset Overflow Interrupt Status not detected 1_B **ACTIVE**: Frame Offset Overflow Interrupt Status detected
PDRFIS	3	rw	**Packet Dropped due to RF Interrupt Status** When the Rx-Parser requests a packet-drop (by setting RF=1 in the instruction table), this bit is set to 1. This bit is cleared when the software writes 1 to this bit. Access restriction applies. Self-set to 1 on internal event. Clears on write of 1. Writing 0 has no effect. 0_B **INACTIVE**: Packet Dropped due to RF Interrupt Status not detected 1_B **ACTIVE**: Packet Dropped due to RF Interrupt Status detected

(table continues...)

(continued)

Field	Bits	Type	Description
NVEOVIE	16	rw	**Number of Valid Entries Overflow Interrupt Enable** When this bit is set, the interrupt is asserted when NVEOVIS bit is set. When this bit is reset, the interrupt is not asserted when NVEOVIS is set. 0_B **DISABLE**: Last valid instruction Overflow Interrupt is disabled 1_B **ENABLE**: Last valid instruction Overflow Interrupt is enabled
NPEOVIE	17	rw	**Number of Parsable Entries Overflow Interrupt Enable** When this bit is set, the interrupt is asserted when NPEOVIS bit is set. When this bit is reset, the interrupt is not asserted when NPEOVIS is set. 0_B **DISABLE**: Number of Parsable Instructions Overflow Interrupt is disabled 1_B **ENABLE**: Number of Parsable Instructions Overflow Interrupt is enabled
FOOVIE	18	rw	**Frame Offset Overflow Interrupt Enable** When this bit is set, the interrupt is asserted when FOOVIS bit is set. When this bit is reset, the interrupt is not asserted when FOOVIS is set. 0_B **DISABLE**: Frame Offset Overflow Interrupt is disabled 1_B **ENABLE**: Frame Offset Overflow Interrupt is enabled
PDRFIE	19	rw	**Packet Drop due to RF Interrupt Enable** When this bit is set, the interrupt is asserted when PDRFIS bit is set. When this bit is reset, the interrupt is not asserted when PDRFIS is set. 0_B **DISABLE**: Packet Drop due to RF Interrupt is disabled 1_B **ENABLE**: Packet Drop due to RF Interrupt is enabled
0	15:4, 31:20	r	**Reserved** Read as 0; should be written with 0.

14.6.523 Port j MTL Indirect Acc Control Status

This is Control register to access the MTL indirect registers/tables.

Portj_MTL_Indirect_Acc_Control_Status (j=0-1) Offset address: $110B0_H+j*2000_H$

Port j MTL Indirect Acc Control Status Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
STARTBUSY	0					ACCSEL		0	RXPEIEC		RXPEIEE	0	CRWEN	CRWSEL	WRRDN
rw	r					rw		r	rw		rw	r	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0						ADDR									
r						rw									

Field	Bits	Type	Description
ADDR	9:0	rw	**Indirect Access Write/Read Address** This field indicates the address of each 32-bit data/register of the selected ACCSEL. ACCSEL Value Usage of this Field - ACC_IT: used for Instruction Table's 32 bits data word address. - ACC_IFR: used for Indirect FRP Register Block. Indirect FRP Register Block Address Map. 1) 0x0 : For MTL_RXP_Drop_Cnt. 2) 0x1 : For MTL_RXP_Error_Cnt. 3) 0x2 : For MTL_RXP_Bypass_Cnt. 4) 0x40 + (0x10 * i + 0x0): For MTL_FRP_ACNTR, where i = 0 to DWCXG_FRP_NUM_AC-1. 5) 0x40 + (0x10 * i + 0x2): For MTL_FRP_ACNTR_CNFG, where i = 0 to DWCXG_FRP_NUM_AC-1. Note: Use valid address according to the ACCSEL selection. Software shall write this before issuing any Read/Write command.
WRRDN	16	rw	**Read Write Control** When this bit is set to 1, indicates the write operation is selected for ACCSEL. When this bit is set to 0, indicates the read operation is selected for ACCSEL. 0_B **READ**: Read operation to ACCSEL 1_B **WRITE**: Write operation to ACCSEL
CRWSEL	17	rw	**Continuous Read/Write Mode Select** This field is valid only when Instruction Table (ACC_IT), ACC_RCHLIST or ACC_MMC is selected in ACCSEL field. When CRWEN is set, CRWSEL is valid. Software can select which continuous mode to use, either Write or Read. 0_B **READ**: Continuous Read 1_B **WRITE**: Continuous Write
CRWEN	18	rw	**Continuous Read/Write Mode Enable** This field is valid only when Instruction Table (ACC_IT) or ACC_RCHLIST is selected in the ACCSEL field. When this bit is set, continuous read/write mode is enabled. CRWSEL selects either Read or Write continuous mode. In this mode, software has to set ADDR, STARTBUSY once. For subsequent write/read operations, it is not required to set ADDR and STARTBUSY. The hardware automatically triggers the read or write access to the next address when the MTL_Indirect_Acc_Data register is read or written. To disable this mode, set this bit to 0. Note: Byte-wise Write/Read is not supported. 0_B **DISABLE**: Disabled Continuous Read/Write Mode 1_B **ENABLE**: Enabled Continuous Read/Write Mode

(table continues...)

(continued)

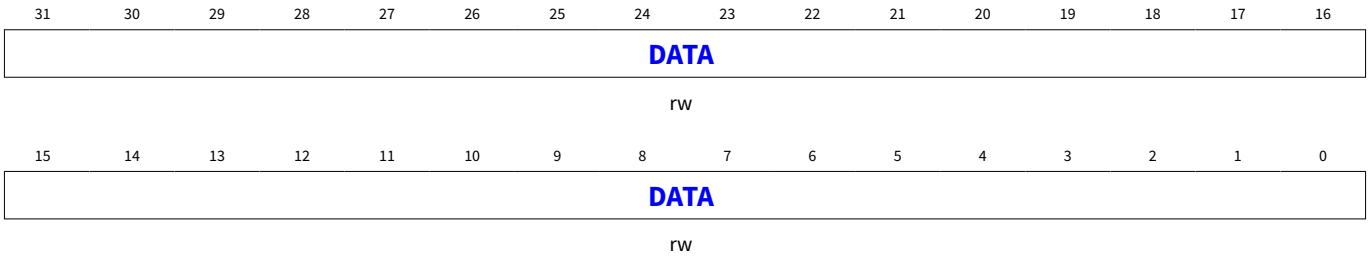
Field	Bits	Type	Description
RXPEIEE	20	rw	**ECC Inject Error Enable** This field is valid only when Instruction Table (ACC_IT) is selected in ACCSEL field. When set, enables the ECC error injection feature. When reset, disables the ECC error injection feature. Note: Hardware injects error when this bit is set and software initiates read access to Rx Parser Instruction Table. 0_B **DISABLE**: ECC Inject Error for ACCSEL is disabled 1_B **ENABLE**: ECC Inject Error for ACCSEL is enabled
RXPEIEC	22:21	rw	**ECC Inject Error Control** This field is valid only when Instruction Table (ACC_IT) is selected in the ACCSEL field. When RXPEIEE bit of this register is set, following errors are inserted based on the value encoded in this field. 00_B **D_1BIT**: Insert 1 bit ECC error in data or parity 01_B **D_2BIT**: Insert 2 bit ECC error in data or parity 10_B **A_1BIT**: Insert 1 bit ECC error in address 11_B **RSVD**: Reserved
ACCSEL	25:24	rw	**Indirect Access Selection** This field is used to select the indirect table/register block to access. Software can select what to access from following selection option: - 0x0: ACC_IT, Access FRP Instruction Table. - 0x1: ACC_IFR, Access Indirect FRP Register block. - 0x2: Reserved. - 0x3: Reserved.
STARTBUSY	31	rw	**Indirect Access Busy** When this bit is set to 1 by the software, it indicates to start the Read/ Write operation from/to selected ACCSEL. Software must read this bit as 0 before issuing read or write request to access the selected ACCSEL. This bit is cleared by DWC_xgmac when the the read/write transaction is completed. Software must not issue the next read or write operation, until this bit is cleared. 0_B **INACTIVE**: Read/Write transaction complete and Idle 1_B **ACTIVE**: Read/Write transaction is active
0	15:10, 19, 23, 30:26	r	**Reserved** Read as 0; should be written with 0.

14.6.524 Port j MTL Indirect Acc Data

This is the data register to Access MTL indirect registers/tables.

Portj_MTL_Indirect_Acc_Data (j=0-1) Offset address: $110B4_H+j*2000_H$

Port j MTL Indirect Acc Data Kernel Reset value: $0000\ 0000_H$



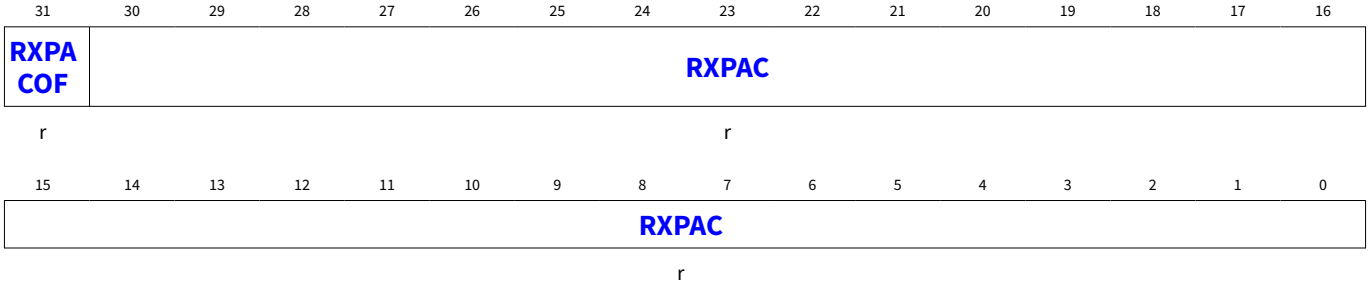
Field	Bits	Type	Description
DATA	31:0	rw	**Indirect Access Write/Read Data** Software must write this register before issuing any write command. DWC_xgmac provides the read data from selected ACCSEL for read operation when STARTBUSY =0 after read command.

14.6.525 Port j MTL FRP ACNTR0

The MTL_FRP_ACNTR(#i) registers are indirect and provide the count of the number of frames accepted by FRP particular to channel. Such accept counters mapping with receive channel is static, one to one, if supported number of receive channels are less than or equal to 16. Otherwise if it is more than 16 then, FRP implements fix number of accept counters and provides programmable option to application to map it to application-specific receive channel via corresponding MTL_FRP_ACNTR_CNFG(#i) indirect registers. These indirect registers can be accessible through MTL_Indirect_Acc_Control_Status and MTL_Indirect_Acc_Data registers.

Portj_MTL_FRP_ACNTR0 (j=0-1) Offset address: $110B4_H+j*2000_H$

Port j MTL FRP ACNTR0 Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
RXPAC	30:0	r	**Rx-Parser Accept Counter** This 31-bit counter is incremented whenever the Rx Parser accepts a packet due to AF = 1 and RF = 0. The packet must match the Rx Parser acceptance criteria irrespective of its type (length or type) and size (undersize/runt or oversize). The counter is cleared when the register is read.
RXPACOF	31	r	**Rx-Parser Accept Counter Overflow Bit** When set, this bit indicates that the RXPAC Counter field crossed the maximum limit. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0_B **INACTIVE**: Rx Parser Accept Counter overflow not occurred 1_B **ACTIVE**: Rx Parser Accept Counter overflow occurred

14.6.526 Port j MTL FRP ACNTR1

The MTL_FRP_ACNTR(#i) registers are indirect and provide the count of the number of frames accepted by FRP particular to channel. Such accept counters mapping with receive channel is static, one to one, if supported number of receive channels are less than or equal to 16. Otherwise if it is more than 16 then, FRP implements fix number of accept counters and provides programmable option to application to map it to application-specific receive channel via corresponding MTL_FRP_ACNTR_CNFG(#i) indirect registers. These indirect registers can be accessible through MTL_Indirect_Acc_Control_Status and MTL_Indirect_Acc_Data registers.

Portj_MTL_FRP_ACNTR1 (j=0-1) Offset address: $110B4_H+j*2000_H$

Port j MTL FRP ACNTR1 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RXPACOF	RXPAC														
r	r														

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXPAC															
r															

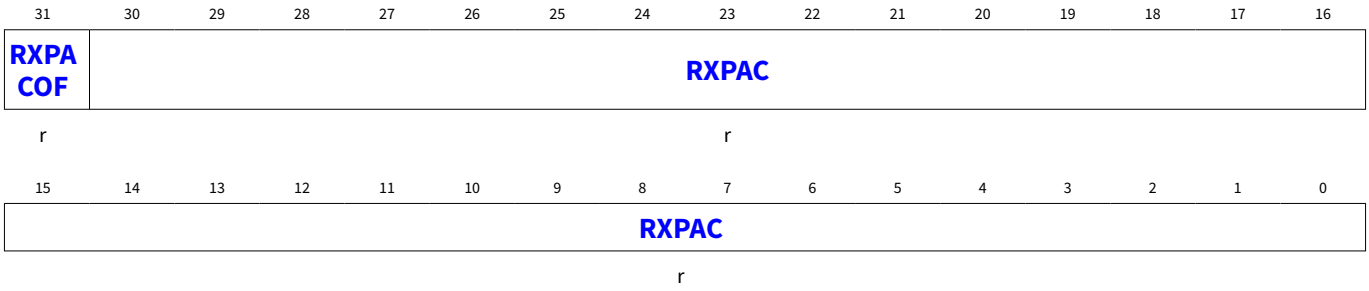
Field	Bits	Type	Description
RXPAC	30:0	r	**Rx-Parser Accept Counter** This 31-bit counter is incremented whenever the Rx Parser accepts a packet due to AF = 1 and RF = 0. The packet must match the Rx Parser acceptance criteria irrespective of its type (length or type) and size (undersize/runt or oversize). The counter is cleared when the register is read.
RXPACOF	31	r	**Rx-Parser Accept Counter Overflow Bit** When set, this bit indicates that the RXPAC Counter field crossed the maximum limit. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0_B **INACTIVE**: Rx Parser Accept Counter overflow not occurred 1_B **ACTIVE**: Rx Parser Accept Counter overflow occurred

14.6.527 Port j MTL FRP ACNTR2

The MTL_FRP_ACNTR(#i) registers are indirect and provide the count of the number of frames accepted by FRP particular to channel. Such accept counters mapping with receive channel is static, one to one, if supported number of receive channels are less than or equal to 16. Otherwise if it is more than 16 then, FRP implements fix number of accept counters and provides programmable option to application to map it to application-specific receive channel via corresponding MTL_FRP_ACNTR_CNFG(#i) indirect registers. These indirect registers can be accessible through MTL_Indirect_Acc_Control_Status and MTL_Indirect_Acc_Data registers.

Portj_MTL_FRP_ACNTR2 (j=0-1) Offset address: $110B4_H+j*2000_H$

Port j MTL FRP ACNTR2 Kernel Reset value: $0000\ 0000_H$



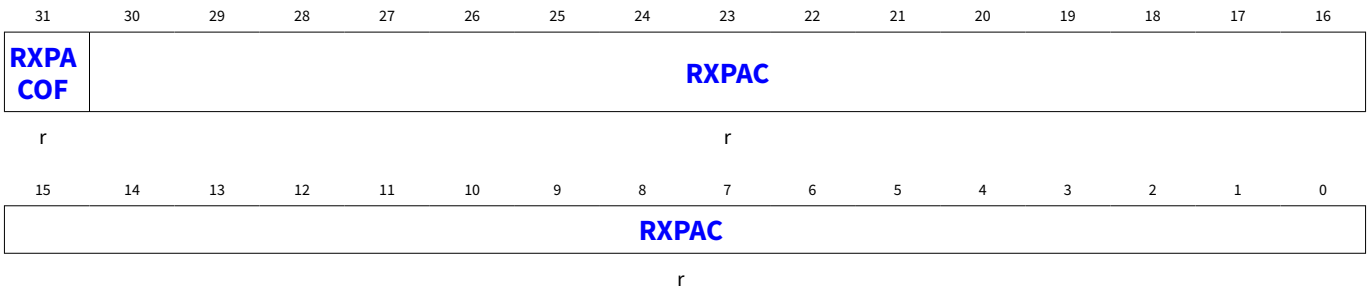
Field	Bits	Type	Description
RXPAC	30:0	r	**Rx-Parser Accept Counter** This 31-bit counter is incremented whenever the Rx Parser accepts a packet due to AF = 1 and RF = 0. The packet must match the Rx Parser acceptance criteria irrespective of its type (length or type) and size (undersize/runt or oversize). The counter is cleared when the register is read.
RXPACOF	31	r	**Rx-Parser Accept Counter Overflow Bit** When set, this bit indicates that the RXPAC Counter field crossed the maximum limit. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0_B **INACTIVE**: Rx Parser Accept Counter overflow not occurred 1_B **ACTIVE**: Rx Parser Accept Counter overflow occurred

14.6.528 Port j MTL FRP ACNTR3

The MTL_FRP_ACNTR(#i) registers are indirect and provide the count of the number of frames accepted by FRP particular to channel. Such accept counters mapping with receive channel is static, one to one, if supported number of receive channels are less than or equal to 16. Otherwise if it is more than 16 then, FRP implements fix number of accept counters and provides programmable option to application to map it to application-specific receive channel via corresponding MTL_FRP_ACNTR_CNFG(#i) indirect registers. These indirect registers can be accessible through MTL_Indirect_Acc_Control_Status and MTL_Indirect_Acc_Data registers.

Portj_MTL_FRP_ACNTR3 (j=0-1) Offset address: $110B4_H+j*2000_H$

Port j MTL FRP ACNTR3 Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
RXPAC	30:0	r	**Rx-Parser Accept Counter** This 31-bit counter is incremented whenever the Rx Parser accepts a packet due to AF = 1 and RF = 0. The packet must match the Rx Parser acceptance criteria irrespective of its type (length or type) and size (undersize/runt or oversize). The counter is cleared when the register is read.
RXPACOF	31	r	**Rx-Parser Accept Counter Overflow Bit** When set, this bit indicates that the RXPAC Counter field crossed the maximum limit. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0_B **INACTIVE**: Rx Parser Accept Counter overflow not occurred 1_B **ACTIVE**: Rx Parser Accept Counter overflow occurred

14.6.529 Port j MTL FRP ACNTR4

The MTL_FRP_ACNTR(#i) registers are indirect and provide the count of the number of frames accepted by FRP particular to channel. Such accept counters mapping with receive channel is static, one to one, if supported number of receive channels are less than or equal to 16. Otherwise if it is more than 16 then, FRP implements fix number of accept counters and provides programmable option to application to map it to application-specific receive channel via corresponding MTL_FRP_ACNTR_CNFG(#i) indirect registers. These indirect registers can be accessible through MTL_Indirect_Acc_Control_Status and MTL_Indirect_Acc_Data registers.

Portj_MTL_FRP_ACNTR4 (j=0-1) Offset address: $110B4_H+j*2000_H$

Port j MTL FRP ACNTR4 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RXPACOF	RXPAC														
r	r														

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXPAC															
r															

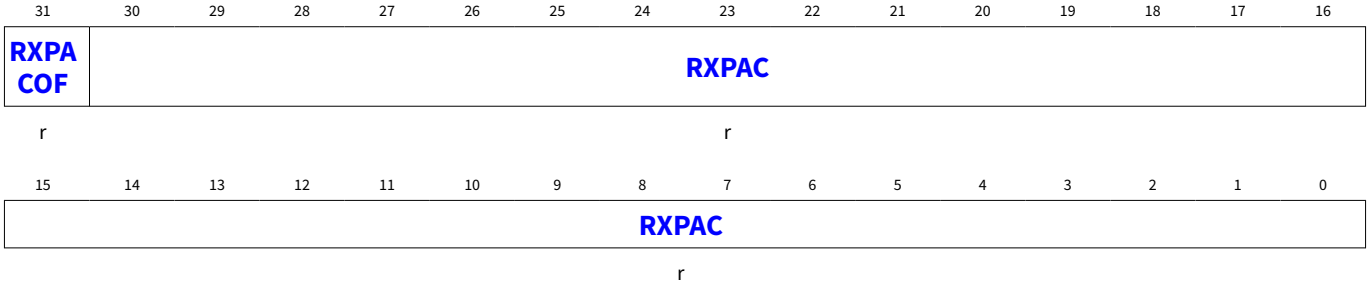
Field	Bits	Type	Description
RXPAC	30:0	r	**Rx-Parser Accept Counter** This 31-bit counter is incremented whenever the Rx Parser accepts a packet due to AF = 1 and RF = 0. The packet must match the Rx Parser acceptance criteria irrespective of its type (length or type) and size (undersize/runt or oversize). The counter is cleared when the register is read.
RXPACOF	31	r	**Rx-Parser Accept Counter Overflow Bit** When set, this bit indicates that the RXPAC Counter field crossed the maximum limit. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0_B **INACTIVE**: Rx Parser Accept Counter overflow not occurred 1_B **ACTIVE**: Rx Parser Accept Counter overflow occurred

14.6.530 Port j MTL FRP ACNTR5

The MTL_FRP_ACNTR(#i) registers are indirect and provide the count of the number of frames accepted by FRP particular to channel. Such accept counters mapping with receive channel is static, one to one, if supported number of receive channels are less than or equal to 16. Otherwise if it is more than 16 then, FRP implements fix number of accept counters and provides programmable option to application to map it to application-specific receive channel via corresponding MTL_FRP_ACNTR_CNFG(#i) indirect registers. These indirect registers can be accessible through MTL_Indirect_Acc_Control_Status and MTL_Indirect_Acc_Data registers.

Portj_MTL_FRP_ACNTR5 (j=0-1) Offset address: $110B4_H+j*2000_H$

Port j MTL FRP ACNTR5 Kernel Reset value: $0000\ 0000_H$



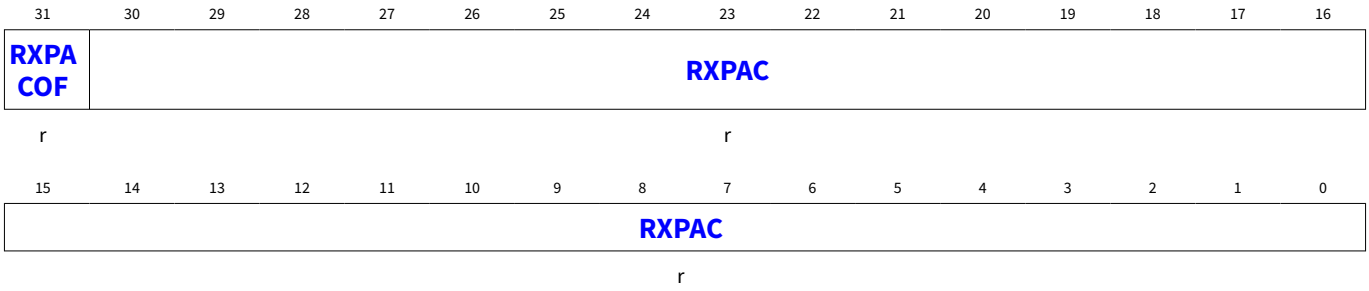
Field	Bits	Type	Description
RXPAC	30:0	r	**Rx-Parser Accept Counter** This 31-bit counter is incremented whenever the Rx Parser accepts a packet due to AF = 1 and RF = 0. The packet must match the Rx Parser acceptance criteria irrespective of its type (length or type) and size (undersize/runt or oversize). The counter is cleared when the register is read.
RXPACOF	31	r	**Rx-Parser Accept Counter Overflow Bit** When set, this bit indicates that the RXPAC Counter field crossed the maximum limit. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0_B **INACTIVE**: Rx Parser Accept Counter overflow not occurred 1_B **ACTIVE**: Rx Parser Accept Counter overflow occurred

14.6.531 Port j MTL FRP ACNTR6

The MTL_FRP_ACNTR(#i) registers are indirect and provide the count of the number of frames accepted by FRP particular to channel. Such accept counters mapping with receive channel is static, one to one, if supported number of receive channels are less than or equal to 16. Otherwise if it is more than 16 then, FRP implements fix number of accept counters and provides programmable option to application to map it to application-specific receive channel via corresponding MTL_FRP_ACNTR_CNFG(#i) indirect registers. These indirect registers can be accessible through MTL_Indirect_Acc_Control_Status and MTL_Indirect_Acc_Data registers.

Portj_MTL_FRP_ACNTR6 (j=0-1) Offset address: $110B4_H+j*2000_H$

Port j MTL FRP ACNTR6 Kernel Reset value: $0000\ 0000_H$



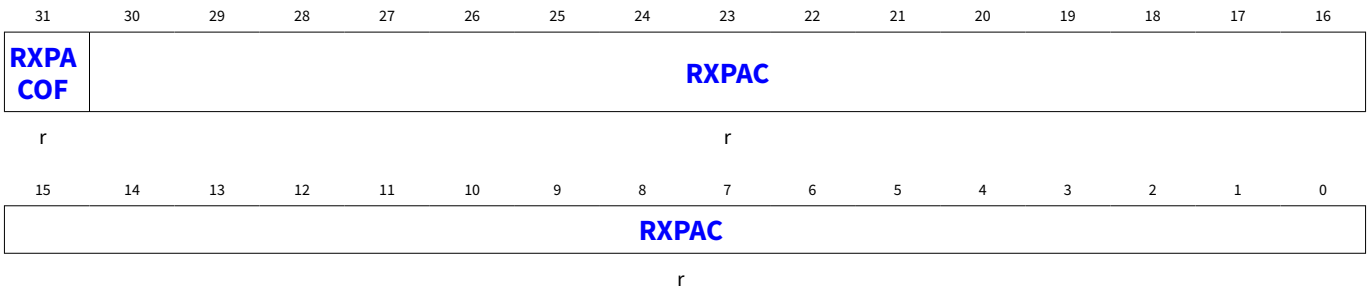
Field	Bits	Type	Description
RXPAC	30:0	r	**Rx-Parser Accept Counter** This 31-bit counter is incremented whenever the Rx Parser accepts a packet due to AF = 1 and RF = 0. The packet must match the Rx Parser acceptance criteria irrespective of its type (length or type) and size (undersize/runt or oversize). The counter is cleared when the register is read.
RXPACOF	31	r	**Rx-Parser Accept Counter Overflow Bit** When set, this bit indicates that the RXPAC Counter field crossed the maximum limit. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0_B **INACTIVE**: Rx Parser Accept Counter overflow not occurred 1_B **ACTIVE**: Rx Parser Accept Counter overflow occurred

14.6.532 Port j MTL FRP ACNTR7

The MTL_FRP_ACNTR(#i) registers are indirect and provide the count of the number of frames accepted by FRP particular to channel. Such accept counters mapping with receive channel is static, one to one, if supported number of receive channels are less than or equal to 16. Otherwise if it is more than 16 then, FRP implements fix number of accept counters and provides programmable option to application to map it to application-specific receive channel via corresponding MTL_FRP_ACNTR_CNFG(#i) indirect registers. These indirect registers can be accessible through MTL_Indirect_Acc_Control_Status and MTL_Indirect_Acc_Data registers.

Portj_MTL_FRP_ACNTR7 (j=0-1) Offset address: $110B4_H+j*2000_H$

Port j MTL FRP ACNTR7 Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
RXPAC	30:0	r	**Rx-Parser Accept Counter** This 31-bit counter is incremented whenever the Rx Parser accepts a packet due to AF = 1 and RF = 0. The packet must match the Rx Parser acceptance criteria irrespective of its type (length or type) and size (undersize/runt or oversize). The counter is cleared when the register is read.
RXPACOF	31	r	**Rx-Parser Accept Counter Overflow Bit** When set, this bit indicates that the RXPAC Counter field crossed the maximum limit. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0_B **INACTIVE**: Rx Parser Accept Counter overflow not occurred 1_B **ACTIVE**: Rx Parser Accept Counter overflow occurred

14.6.533 Port j MTL RXP Bypass Cnt

The MTL_RXP_Bypass_Cnt registers provides the count of the number of frames bypassed by Rx Parser. This is an indirect register which can be access through MTL_Indirect_Acc_Control_Status and MTL_Indirect_Acc_Data registers.

Portj_MTL_RXP_Bypass_Cnt (j=0-1) Offset address: $110B4_H+j*2000_H$

Port j MTL RXP Bypass Cnt Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RXPBCOF	RXPBC														
r	r														

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXPBC															
r															

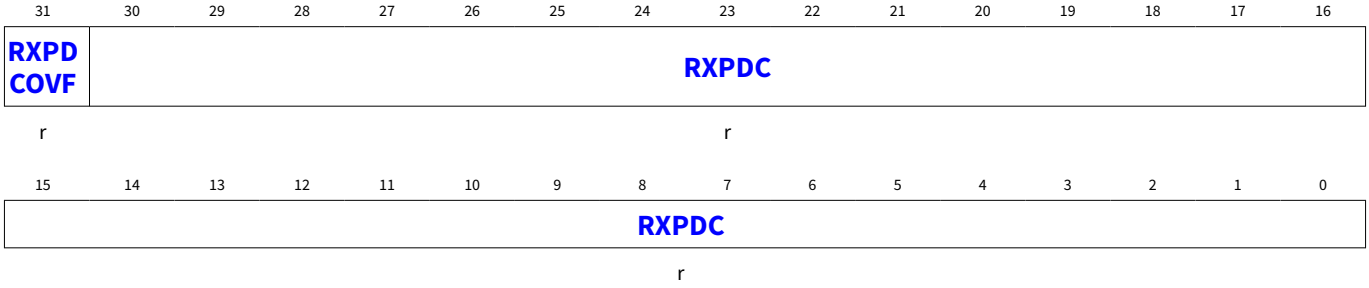
Field	Bits	Type	Description
RXPBC	30:0	r	**Rx-Parser Bypass Counter** This 31-bit counter is implemented whenever a Rx Parser Bypass a packet due to AF = 1 and RF = 1. The counter is cleared when the register is read.
RXPBCOF	31	r	**Rx-Parser Bypass Counter Overflow Bit** When set, this bit indicates that the RXPBC Counter field crossed the maximum limit. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0_B **INACTIVE**: Rx Parser Bypass Counter overflow not occurred 1_B **ACTIVE**: Rx Parser Bypass Counter overflow occurred

14.6.534 Port j MTL RXP Drop Cnt

This register provides the number of packets dropped by the Flexible Receive Parser. This is an indirect register which can be accessed through MTL_Indirect_Acc_Control_Status and MTL_Indirect_Acc_Data registers.

Portj_MTL_RXP_Drop_Cnt (j=0-1) Offset address: $110B4_H+j*2000_H$

Port j MTL RXP Drop Cnt Kernel Reset value: $0000\ 0000_H$



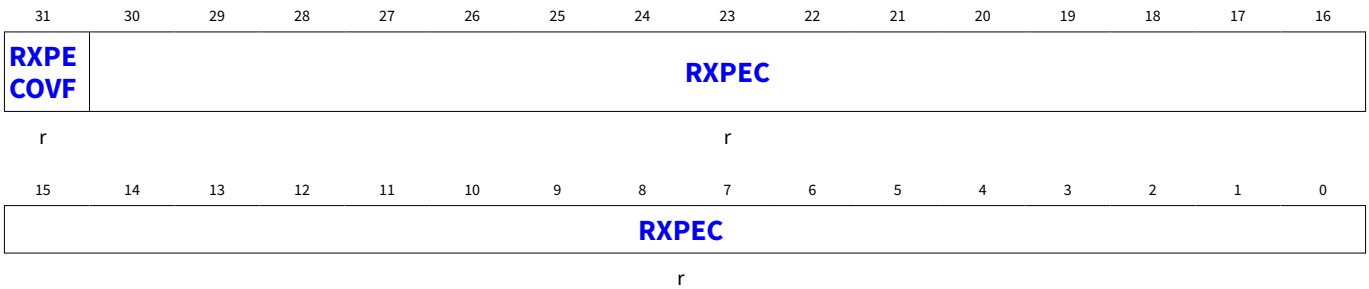
Field	Bits	Type	Description
RXPDC	30:0	r	**Rx Parser Packet Drop Count** This 31-bit counter is incremented when a Rx Parser drops a packet due to AF = 0 and RF = 1. The counter is cleared when the register is read.
RXPDCOVF	31	r	**Rx Parser Packet Drop Counter Overflow** When set, this bit indicates that the MTL_RXP_Drop_cnt (RXPDC) counter field has crossed the maximum limit. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0_B **INACTIVE**: Rx Parser Drop count overflow not occurred 1_B **ACTIVE**: Rx Parser Drop count overflow occurred

14.6.535 Port j MTL RXP Error Cnt

This register provides Error Counts of the Flexible Receive Parser. This is an indirect register which can be access through MTL_Indirect_Acc_Control_Status and MTL_Indirect_Acc_Data registers.

Portj_MTL_RXP_Error_Cnt (j=0-1) Offset address: $110B4_H+j*2000_H$

Port j MTL RXP Error Cnt Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
RXPEC	30:0	r	**Rx Parser Error count** This 31-bit counter is incremented when the Rx Parser encounters the following error scenarios: - Current Instruction Address is greater than NVE[] - Number of Parsed Instructions is greater than (NPE + 1) - Current Instruction's Frame offset is greater than EOF frame offset or Last Parasable Byte - Parsing is not complete even after receiving the next to next packet SOF The counter is cleared when the register is read.
RXPECOVF	31	r	**Rx Parser Error Counter Overflow Bit** When set, this bit indicates that the MTL_RXP_Error_cnt (RXPEC) counter field has crossed the maximum limit. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0_B **INACTIVE**: Rx Parser Error count overflow not occurred 1_B **ACTIVE**: Rx Parser Error count overflow occurred

14.6.536 Port j MTL ECC Control

The MTL_ECC_Control register establishes the operating mode of ECC related to MAC and MTL memories.

Portj_MTL_ECC_Control (j=0-1) Offset address: $110C0_H+j*2000_H$

Port j MTL ECC Control Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															0
r															r

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0							MEEAO	0		DESCED	0	MRXPED	MGCLED	MRXED	MTXED
r							rw	r		rw	r	rw	rw	rw	rw

Field	Bits	Type	Description
MTXED	0	rw	**MTL Tx FIFO ECC Checker Disable - 1: Disables the ECC Checker for MTL Tx FIFO memory** - 0: Enables the ECC Checker for MTL Tx FIFO memory.
MRXED	1	rw	**MTL Rx FIFO ECC Checker Disable - 1: Disables the ECC Checker for MTL Rx FIFO memory** - 0: Enables the ECC Checker for MTL Rx FIFO memory.
MGCLED	2	rw	**MTL GCL ECC Checker Disable - 1: Disables the ECC Checker for GCL memory** - 0: Enables the ECC Checker for GCL memory.

(table continues...)

(continued)

Field	Bits	Type	Description
MRXPED	3	rw	**MTL Rx Parser ECC Checker Disable - 1: Disables the ECC Checker for Rx Parser memory** - 0: Enables the ECC Checker for Rx Parser memory.
DESCED	5	rw	**DCACHE memory ECC Checker Disable - 1: Disables the ECC Checker for DCACHE memory in DMA** - 0: Enables the ECC Checker for DCACHE memory in DMA.
MEEAO	8	rw	**MAC/MTL ECC Error Address Status Override** When set to 1, the address of location in which the latest ECC error is detected, is stored in the MTL_ECC_Err_Addr_Status register. When reset to 0, the address of the location in which the ECC error is reported for the first time is stored in this register.
0	4, 7:6, 15:9, 16, 31:17	r	**Reserved** Read as 0; should be written with 0.

14.6.537 Port j MTL Safety Interrupt Status

The MTL_Safety_Interrupt_Status registers indicates the occurrence of various safety related events.

Portj_MTL_Safety_Interrupt_Status (j=0-1) Offset address: $110C4_H+j*2000_H$

Port j MTL Safety Interrupt Status Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MCSIS	0														
r	r														

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0														MEUIS	MECIS
r														r	r

Field	Bits	Type	Description
MECIS	0	r	**MAC/MTL ECC Correctable error Interrupt Status** This bit is set when the interrupt due to Correctable Status is set in the MTL_ECC_Interrupt_Status register. To get the exact cause of the interrupt, the application must read the MTL_ECC_Interrupt_Status register.
MEUIS	1	r	**MAC/MTL ECC Uncorrectable error Interrupt Status** This bit is set when the interrupt due to Unrecoverable Status is set in the MTL_ECC_Interrupt_Status register. To get the exact cause of the interrupt, the application must read the MTL_ECC_Interrupt_Status register.

(table continues...)

(continued)

Field	Bits	Type	Description
MCSIS	31	r	**MAC Safety Unrecoverable Interrupt Status** This bit is set when the interrupt due to Unrecoverable Status is set in the MAC_DPP_FSM_Interrupt_Status register. MAC_DPP_FSM_Interrupt_Status register must be read when this bit is set, to get the cause of the safety interrupt in MAC.
0	30:2	r	**Reserved** Read as 0; should be written with 0.

14.6.538 Port j MTL ECC Interrupt Enable

The MTL_ECC_Interrupt_Enable register provides enable bits for the ECC interrupts.

Portj_MTL_ECC_Interrupt_Enable (j=0-1) Offset address: $110C8_H+j*2000_H$

Port j MTL ECC Interrupt Enable Kernel Reset value: $0000\ 1111_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															0
r															r

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			RPCEIE	0			GCEIE	0			RXCEIE	0			TXCEIE
r			rw	r			rw	r			rw	r			rw

Field	Bits	Type	Description
TXCEIE	0	rw	**Tx memory Correctable Error Interrupt Enable** When set, generates an interrupt when a correctable error is detected at the MTL Tx memory interface. It is indicated in the TXCES bit of MTL_ECC_Interrupt_Status register. When reset, this event does not generate an interrupt. 0_B **DISABLE**: Tx memory Correctable Error Interrupt is disabled 1_B **ENABLE**: Tx memory Correctable Error Interrupt is enabled
RXCEIE	4	rw	**Rx memory Correctable Error Interrupt Enable** When set, generates an interrupt when a correctable error is detected at the MTL Rx memory interface. It is indicated in the RXCES bit of MTL_ECC_Interrupt_Status register. When reset, this event does not generate an interrupt. 0_B **DISABLE**: Rx memory Correctable Error Interrupt is disabled 1_B **ENABLE**: Rx memory Correctable Error Interrupt is enabled

(table continues...)

(continued)

Field	Bits	Type	Description
GCEIE	8	rw	**GCL memory Correctable Error Interrupt Enable** When set, generates an interrupt when a correctable error is detected at the MTL GCL memory interface(EST/SGF). It is indicated in the GCES and SCES bits of MTL_ECC_Interrupt_Status register. When reset, this event does not generate an interrupt. 0_B **DISABLE**: GCL memory Correctable Error Interrupt is disabled 1_B **ENABLE**: GCL memory Correctable Error Interrupt is enabled
RPCEIE	12	rw	**Rx Parser memory Correctable Error Interrupt Enable** When set, generates an interrupt when an correctable error is detected at the Rx Parser memory interface. It is indicated in RPCES status bit of MTL_ECC_Interrupt_Status register. When reset, this event does not generate an interrupt. 0_B **DISABLE**: Rx Parser memory Correctable Error Interrupt is disabled 1_B **ENABLE**: Rx Parser memory Correctable Error Interrupt is enabled
0	3:1, 7:5, 11:9, 15:13, 16, 31:17	r	**Reserved** Read as 0; should be written with 0.

14.6.539 Port j MTL ECC Interrupt Status

The MTL_ECC_Interrupt_Status register provides MAC/MTL ECC Interrupt Status.

Portj_MTL_ECC_Interrupt_Status (j=0-1) Offset address: $110CC_H+j*2000_H$

Port j MTL ECC Interrupt Status Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0									0	0	0	0	SUES	SAMS	SCES
r									r	r	r	r	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	RPUES	RPAMS	RPCES	0	GUES	GAMS	GCES	0	RXUES	RXAMS	RXCES	0	TXUES	TXAMS	TXCES
r	rw	rw	rw	r	rw	rw	rw	r	rw	rw	rw	r	rw	rw	rw

Field	Bits	Type	Description
TXCES	0	rw	**MTL Tx memory Correctable Error Status** When this bit is set, it indicates that a correctable error is detected at the MTL Tx memory. 0_B **INACTIVE**: MTL Tx memory Correctable Error Status not detected 1_B **ACTIVE**: MTL Tx memory Correctable Error Status detected

(table continues...)

(continued)

Field	Bits	Type	Description
TXAMS	1	rw	**MTL Tx memory Address Mismatch Status** When this bit is set, it indicates that address mismatch is found for address bus of the MTL Tx memory. 0_B **INACTIVE**: MTL Tx memory Address Mismatch Status not detected 1_B **ACTIVE**: MTL Tx memory Address Mismatch Status detected
TXUES	2	rw	**MTL Tx memory Uncorrectable Error Status** When set, indicates that an uncorrectable error is detected at the MTL Tx memory interface. 0_B **INACTIVE**: MTL Tx memory Uncorrectable Error Status not detected 1_B **ACTIVE**: MTL Tx memory Uncorrectable Error Status detected
RXCES	4	rw	**MTL Rx memory Correctable Error Status** When this bit is set, it indicates that correctable error is detected at the MTL Rx memory. 0_B **INACTIVE**: MTL Rx memory correctable Error Status not detected 1_B **ACTIVE**: MTL Rx memory correctable Error Status detected
RXAMS	5	rw	**MTL Rx memory Address Mismatch Status** When this bit is set, it indicates that address mismatch is found for address bus of the MTL Rx memory. 0_B **INACTIVE**: MTL Rx memory Address Mismatch Status not detected 1_B **ACTIVE**: MTL Rx memory Address Mismatch Status detected
RXUES	6	rw	**MTL Rx memory Uncorrectable Error Status** When set, indicates that an uncorrectable error is detected at the MTL Rx memory interface. 0_B **INACTIVE**: MTL Rx memory Uncorrectable Error Status not detected 1_B **ACTIVE**: MTL Rx memory Uncorrectable Error Status detected
GCES	8	rw	**MTL EST GCL memory Correctable Error Status** When this bit is set, it indicates that correctable error is detected at the MTL EST GCL memory. The Correctable safety interrupt is generated when this bit and GCEIE bit of MTL_ECC_Interrupt_Enable register are set. 0_B **INACTIVE**: MTL EST memory Correctable Error Status not detected 1_B **ACTIVE**: MTL EST memory Correctable Error Status detected
GAMS	9	rw	**MTL EST GCL memory Address Mismatch Status** When this bit is set, it indicates that address mismatch is found for address bus of MTL EST GCL memory. 0_B **INACTIVE**: MTL EST memory Address Mismatch Status not detected 1_B **ACTIVE**: MTL EST memory Address Mismatch Status detected

(table continues...)

(continued)

Field	Bits	Type	Description
GUES	10	rw	**MTL EST GCL memory Uncorrectable Error Status** When this bit is set, it indicates that an uncorrectable error is detected at MTL EST GCL memory interface. 0_B **INACTIVE**: MTL EST memory Uncorrectable Error Status not detected 1_B **ACTIVE**: MTL EST memory Uncorrectable Error Status detected
RPCES	12	rw	**MTL Rx Parser memory Correctable Error Status** When this bit is set, it indicates that correctable error is detected at Rx Parser memory interface. 0_B **INACTIVE**: MTL Rx Parser memory Correctable Error Status not detected 1_B **ACTIVE**: MTL Rx Parser memory Correctable Error Status detected
RPAMS	13	rw	**MTL Rx Parser memory Address Mismatch Status** When this bit is set, it indicates that address mismatch is found for address bus of Rx Parser memory. 0_B **INACTIVE**: MTL Rx Parser memory Address Mismatch Status not detected 1_B **ACTIVE**: MTL Rx Parser memory Address Mismatch Status detected
RPUES	14	rw	**Rx Parser memory Uncorrectable Error Status - 1: Indicates that an uncorrectable error is detected at Rx Parser memory interface** 0_B **INACTIVE**: Rx Parser memory Uncorrectable Error Status not detected 1_B **ACTIVE**: Rx Parser memory Uncorrectable Error Status detected
SCES	16	rw	**MTL SGF GCL memory Correctable Error Status** When this bit is set, it indicates that correctable error is detected at the MTL SGF GCL memory. The Correctable safety interrupt is generated when this bit and GCEIE bit of MTL_ECC_Interrupt_Enable register are set. 0_B **INACTIVE**: MTL SGF memory Correctable Error Status not detected 1_B **ACTIVE**: MTL SGF memory Correctable Error Status detected
SAMS	17	rw	**MTL SGF GCL memory Address Mismatch Status** When this bit is set, it indicates that address mismatch is found for address bus of MTL SGF GCL memory. 0_B **INACTIVE**: MTL SGF memory Address Mismatch Status not detected 1_B **ACTIVE**: MTL SGF memory Address Mismatch Status detected
SUES	18	rw	**MTL SGF GCL memory Uncorrectable Error Status** When this bit is set, it indicates that an uncorrectable error is detected at MTL SGF GCL memory interface. 0_B **INACTIVE**: MTL SGF memory Uncorrectable Error Status not detected 1_B **ACTIVE**: MTL SGF memory Uncorrectable Error Status detected

(table continues...)

(continued)

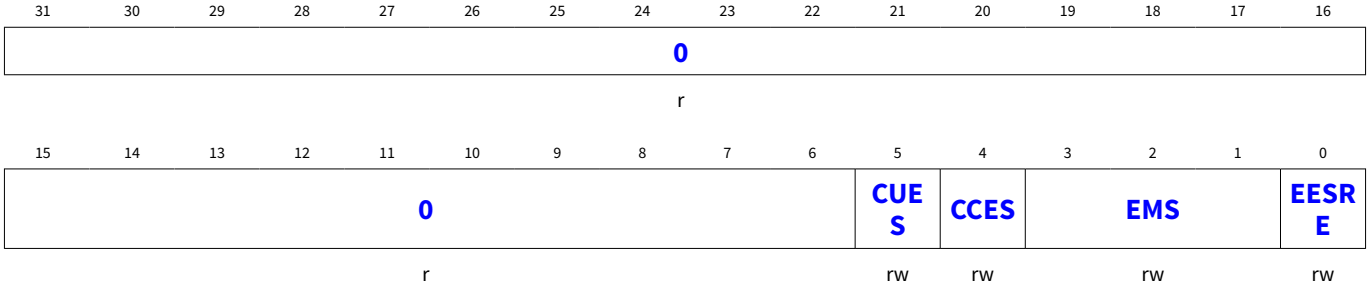
Field	Bits	Type	Description
0	3, 7, 11, 15, 19, 20, 21, 22, 31:23	r	**Reserved** Read as 0; should be written with 0.

14.6.540 Port j MTL ECC Err Sts Rctl

The MTL_ECC_Err_Sts_Rctl register establishes the control for ECC Error status capture.

Portj_MTL_ECC_Err_Sts_Rctl (j=0-1) Offset address: $110D0_H+j*2000_H$

Port j MTL ECC Err Sts Rctl Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
EESRE	0	rw	**MAC/MTL ECC Error Status Read Enable** When this bit is set, the error status of the memory selected by EMS is captured as follows: - The correctable and uncorrectable error count values are captured in the MTL_ECC_Err_Cnt_Status register. - The address locations of correctable and uncorrectable errors are captured in the MTL_ECC_Err_Addr_Status register. Hardware resets this bit when all the status values are captured in the MTL_ECC_Err_Cnt_Status and MTL_ECC_Err_Addr_Status registers. 0_B **DISABLE**: MAC/MTL ECC Error Status Read is disabled 1_B **ENABLE**: MAC/MTL ECC Error Status Read is enabled

(table continues...)

(continued)

Field	Bits	Type	Description
EMS	3:1	rw	**MAC/MTL ECC Memory Selection** When EESRE bit of this register is set, this field indicates which memory's error status value to be read. Values: - 0x0 (TX_MEM): MTL Tx Memory - 0x1 (RX_MEM): MTL Rx Memory - 0x2 (EST_MEM): MTL GCL Memory - 0x3 (RXP_MEM): MTL Rx Parser Memory - 0x5 (DCACHE_MEM): DMA DCACHE Memory - 0x6 (SGF_MEM): MTL SGE
CCES	4	rw	**Clear Correctable Error Status** When this bit is set along with EESRE bit, correctable error address and correctable error count of the memory indicated by EMS field are cleared upon reading. Hardware resets this bit when all the error status values are cleared.
CUES	5	rw	**Clear Uncorrectable Error Status** When this bit is set along with EESRE bit, the uncorrectable error address and uncorrectable error count of the memory indicated by EMS field are cleared upon reading. Hardware resets this bit when all the error status values are cleared.
0	31:6	r	**Reserved** Read as 0; should be written with 0.

14.6.541 Port j MTL ECC Err Addr Status

The MTL_ECC_Err_Addr_Status register provides the memory addresses for the correctable and uncorrectable errors.

Portj_MTL_ECC_Err_Addr_Status (j=0-1) Offset address: $110D4_H+j*2000_H$

Port j MTL ECC Err Addr Status Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EUEAS															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECEAS															
r															

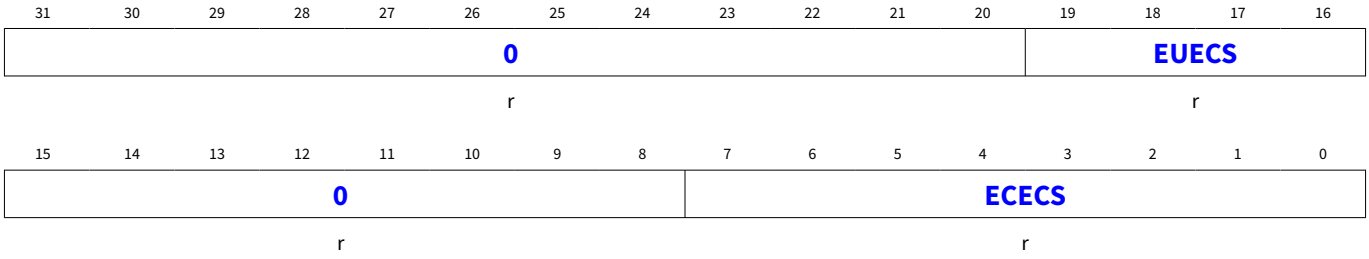
Field	Bits	Type	Description
ECEAS	15:0	r	**MAC/MTL ECC Correctable Error Address Status** Based on the EMS field of MTL_ECC_Err_Sts_Rctl register, this field holds the respective memory's address locations for which a correctable error is detected. When MEEAO bit of MTL_ECC_Control register is set, this field holds the address of memory for which correctable error or address mismatch is detected most recently. When MEEAO bit of MTL_ECC_Control register is reset, this field holds the address of the memory for which correctable error is detected for the first time.
EUEAS	31:16	r	**MAC/MTL ECC Uncorrectable Error Address Status** Based on the EMS field of MTL_ECC_Err_Sts_Rctl register, this field holds the respective memory's address locations for which an uncorrectable error or address mismatch is detected. When MEEAO bit of MTL_ECC_Control register is set, this field holds the address of memory for which either an uncorrectable error or an address mismatch is detected most recently. When MEEAO bit of MTL_ECC_Control register is reset, this field holds the address of the memory for which either an uncorrectable error or address mismatch is detected for the first time.

14.6.542 Port j MTL ECC Err Cntr Status

The MTL_ECC_Err_Cntr_Status register provides Error count for Correctable and uncorrectable ECC errors.

Portj_MTL_ECC_Err_Cntr_Status (j=0-1) Offset address: $110D8_H+j*2000_H$

Port j MTL ECC Err Cntr Status Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
ECECS	7:0	r	**MAC/MTL ECC Correctable Error Counter Status** Based on the EMS field of MTL_ECC_Err_Cntr_Rctl register, this field holds the respective memory's correctable error count value.
EUECS	19:16	r	**MAC/MTL ECC Uncorrectable Error Counter Status** Based on the EMS field of MTL_ECC_Err_Cntr_Rctl register, this field holds the respective memory's uncorrectable error count value.
0	15:8, 31:20	r	**Reserved** Read as 0; should be written with 0.

14.6.543 Port j MTL DPP Control

The MTL_DPP_Control establishes the operating mode of Data Parity protection and error injection.

Portj_MTL_DPP_Control (j=0-1) Offset address: $110E0_H+j*2000_H$

Port j MTL DPP Control Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															IPEASR
r															rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IPECW	IPEMRRC	IPEMRWC	IPEAR	IPEMTFC	IPEMTBU	0	IPEMTD	0	0	IPEDC	0	0	0	OPE	DDPP
rw	rw	rw	rw	rw	rw	r	rw	r	r	rw	r	r	r	rw	rw

Field	Bits	Type	Description
DDPP	0	rw	**Disable Data path Parity Protection** When set to 0, enables the parity protection for XGMAC datapath by generating and checking the parity on XGMAC datapath. When set to 1, disables the parity protection for XGMAC datapath. 0_B **DISABLE**: Data path Parity Protection is enabled 1_B **ENABLE**: Data path Parity Protection is disabled
OPE	1	rw	**Odd Parity Enable - 1: Enables odd parity protection on all the external interfaces** - 0: Enables even parity protection on all the external interfaces. 0_B **DISABLE**: Odd Parity is disabled 1_B **ENABLE**: Odd Parity is enabled
IPEDC	5	rw	**Insert Parity Error in Descriptor Cache data path Parity checker** When set to 1, parity/data bit of first valid input parity/data of the Descriptor Cache parity checker (or at PC-desc) is flipped. Based on the EIM/BLEI field of MTL_DPP_ECC_EIC register, software can corrupt any one bit of parity/data. Hardware clears this bit after the respective parity bit is flipped.
IPEMTD	8	rw	**Insert Parity Error in MTL data path Parity checker** When set to 1, parity/data bit of first valid input parity/data of the MTL transmit write controller parity checker (or at PC2) is flipped. Based on the EIM/BLEI field of MTL_DPP_ECC_EIC register, software can corrupt any one bit of parity/data. Hardware clears this bit after the respective parity bit is flipped. 0_B **DISABLE**: Insert Parity Error in MTL data path Parity checker is disabled 1_B **ENABLE**: Insert Parity Error in MTL data path Parity checker is enabled

(table continues...)

(continued)

Field	Bits	Type	Description
IPEMTBU	10	rw	**Insert Parity error in MAC TBU data path Parity checker** When set to 1, parity/data bit of first valid input parity/data of the MAC TBU input data on mti interface (or at PC4) is flipped. Based on the EIM/BLEI field of MTL_DPP_ECC_EIC register, software can corrupt any one bit of parity/data. Hardware clears this bit after the respective parity bit is flipped. 0_B **DISABLE**: Insert Parity Error in MAC TBU data path Parity Checker is disabled 1_B **ENABLE**: Insert Parity Error in MAC TBU data path Parity Checker is enabled
IPEMTFC	11	rw	**Insert Parity error in MAC TFC data path Parity checker** When set to 1, parity/data bit of first valid input parity/data of the MAC TFC input data (or at PC5) is flipped. Based on the EIM/BLEI field of MTL_DPP_ECC_EIC register, software can corrupt any one bit of parity/data. Hardware clears this bit after the respective parity bit is flipped. 0_B **DISABLE**: Insert Parity error in MAC TFC data path Parity Checker is disabled 1_B **ENABLE**: Insert Parity error in MAC TFC data path Parity Checker is enabled
IPEAR	12	rw	**Insert Parity error in Insert Parity Error in Application Receive interface data path Parity Checker** When set to 1, parity/data bit of first valid input parity/data of Application Receive Interface Parity Checker is flipped. Following checkers based on the system configuration as described below - In MTL configuration (DWCXG = 1), parity checker (PC-b) at ARI interface. - In AXI configuration (DWCXG = 2), parity checker (PC-a) at AXI master interface. Based on the EIM/BLEI field of MTL_DPP_ECC_EIC register, software can corrupt any one bit of parity/data. Hardware clears this bit after the respective parity bit is flipped. 0_B **DISABLE**: Insert Parity error in Application Receive interface data path Parity checker is disabled 1_B **ENABLE**: Insert Parity error in Application Receive interface data path Parity checker is enabled

(table continues...)

(continued)

Field	Bits	Type	Description
IPEMRWC	13	rw	**Insert Parity error in MTL RWC data path Parity Checker** When set to 1, parity/data bit of first valid input parity/data of the MTL RWC input data (or at PC6) is flipped. Based on the EIM/BLEI field of MTL_DPP_ECC_EIC register, software can corrupt any one bit of parity/data. Hardware clears this bit after the respective parity bit is flipped. 0_B **DISABLE**: Insert Parity error in MTL RWC data path Parity checker is disabled 1_B **ENABLE**: Insert Parity error in MTL RWC data path Parity checker is enabled
IPEMRRC	14	rw	**Insert Parity Error in MTL RRC data path Parity Checker** When set to 1, parity/data bit of first valid input parity/data of the MAC RRC output data (or at PC7) is flipped. Based on the EIM/BLEI field of MTL_DPP_ECC_EIC register, software can corrupt any one bit of parity/data. Hardware clears this bit after the respective parity bit is flipped. 0_B **DISABLE**: Insert Parity error in MTL RRC data path Parity checker is disabled 1_B **ENABLE**: Insert Parity error in MTL RRC data path Parity checker is enabled
IPECW	15	rw	**Insert Parity Error in CSR Write data path Parity Checker** When set to 1, parity/data bit of first valid input parity/data of the CSR write data interface is flipped. When EPSI bit of MTL_DPP_Control register is set and this bit is set, parity/data bit of first valid input parity/data ofthe input slave parity ports(or at PC-axi) is flipped. Based on the EIM/BLEI field of MTL_DPP_ECC_EIC register, software can corrupt any one bit of parity/data. Hardware clears this bit after the respective parity bit is flipped. 0_B **DISABLE**: Insert Parity error in CSR Write data path Parity checker is disabled 1_B **ENABLE**: Insert Parity error in CSR Write data path Parity checker is enabled
IPEASR	16	rw	**Insert Parity Error in AXI Slave Read data path Parity Checker** When set to 1, parity/data bit of first valid input parity/data of the AXI Slave read data interface is flipped. Based on the EIM/BLEI field of MTL_DPP_ECC_EIC register, software can corrupt any one bit of parity/data. Hardware clears this bit after the respective parity bit is flipped. 0_B **DISABLE**: Insert Parity error in AXI Slave Read data path Parity checker is disabled 1_B **ENABLE**: Insert Parity error in AXI Slave Read data path Parity checker is enabled

(table continues...)

(continued)

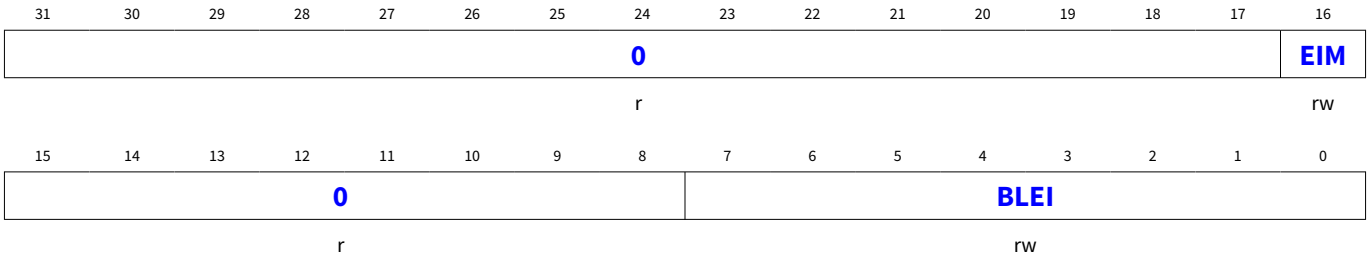
Field	Bits	Type	Description
0	2, 3, 4, 6, 7, 9, 31:17	r	**Reserved** Read as 0; should be written with 0.

14.6.544 Port j MTL DPP ECC EIC

The MTL_DPP_ECC_EIC establishes the operating mode of ECC/DPP error injection.

Portj_MTL_DPP_ECC_EIC (j=0-1) Offset address: $110E4_H+j*2000_H$

Port j MTL DPP ECC EIC Kernel Reset value: $0000\ 0000_H$



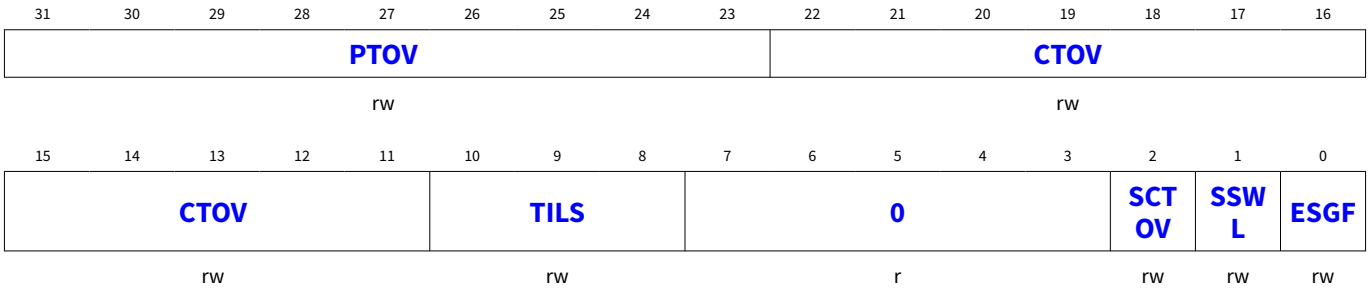
Field	Bits	Type	Description
BLEI	7:0	rw	**Bit Location of error injection** This field indicates the bit location of DPP/ECC error injection, determination of error in Parity/ECC bits or Data (being protected) depends on the Error Injection Mode (EIM field). Depending on the interface being used for error injection not all bits of this field are valid. Example, for error injection on a 64 bit data interface this field must be programmed to a value between 63 and 0. In case of 2-bit error injection bit 0 is always included in error injection and this field must represent the second bit selection for error injection.If the second bit is programed at bit 0 when 2-bit error injection is enabled, DUT inserts only a 1-bit error at bit 0.
EIM	16	rw	**Error Injection Mode - 0: Indicates error injection on data - 1: Indicates error injection on ECC/Parity bits (Check that the address error injection mode is disabled)**
0	15:8, 31:17	r	**Reserved** Read as 0; should be written with 0.

14.6.545 Port j MTL SGF Control

This register controls the operation of Stream-Gate Filtering.

Portj_MTL_SGF_Control (j=0-1) Offset address: $110E8_H+j*2000_H$

Port j MTL SGF Control Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
ESGF	0	rw	**Enable Stream-Gate Filtering - 0: Indicates that the gate control list processing is halted and all gates are assumed to be in Open state** - 1: DWC_xgmac starts processing the gate control lists, only when this bit is 1. During the toggle from 0 to 1, the gate control list processing starts only after the SSWL bit is set. When ECC is selected during the configuration, if an uncorrectable error is detected in the SGF memory, DWC_xgmac resets this bit and disables the SGF function. 0_B **DISABLE**: SGF is disabled 1_B **ENABLE**: SGF is enabled
SSWL	1	rw	**Switch to software owned list - 1: Indicates that the software has programmed the list that it currently owns (SWOL) and the DWC_xgmac must switch to the new list based on the new BTR** DWC_xgmac clears this bit when it switches the SWOL, to indicate the completion of the switch or when a non-recoverable BTR error occurs (BTRL = "11111111" indicates the iterations exceeded the value of 512 and DWC_xgmac is unable to update New BTR to be equal to or greater than Current Time). When BTRE == 1, this bit is cleared. However, SWOL is not updated as the switch is unsuccessful. The write to this bit is effective only when ESGF bit is set to 1 along with it or ESGF bit is previously set to 1. Access restriction applies. Setting 1 sets. Self-cleared. Setting 0 has no effect. 0_B **DISABLE**: Switch to S/W owned list is disabled 1_B **ENABLE**: Switch to S/W owned list is triggered
SCTOV	2	rw	**Sign bit of SGF CTOV** When reset indicates that the SGF CTOV is a negative value. When set that the SGF CTOV is a positive value. 0_B **DISABLE**: Indicates that the SGF CTOV is a negative value 1_B **ENABLE**: Indicates that the SGF CTOV is a positive value

(table continues...)

(continued)

Field	Bits	Type	Description
TILS	10:8	rw	**Time Interval Left Shift Amount** This field provides the left shift amount for the programmed Time Interval values used in the Gate Control Lists. - 000: No left shift needed (equal to x1ns) - 001: Left shift TI by 1 bit (equal to x2ns) - 010: Left shift TI by 2 bits (equal to x4ns) - ... - 111: Left shift TI by 7 bits (equal to x128ns)
CTOV	22:11	rw	**Current Time Offset Value** Provides a 12 bit time offset value in nanoseconds that is added to the current time, to compensate for all the implementation pipeline delays such as the CDC sync delay, buffering delays, data path delays, and so on.
PTOV	31:23	rw	**PTP Time Offset Value** This field must be programmed with a value of PTP Clock period multiplied by 9 in nanoseconds. This value is needed to avoid transmission overruns at the beginning of the installation of a new GCL due to sync delays between PTP clock and MAC transmitter or Application clock.
0	7:3	r	**Reserved** Read as 0; should be written with 0.

14.6.546 Port j MTL SGF Extended Control

This register controls the extended operation of Stream-Gate Filtering.

Portj_MTL_SGF_Extended_Control (j=0-1) Offset address: $110EC_H+j*2000_H$

Port j MTL SGF Extended Control Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												0		ESPF	FFPF F
r												r		rw	rw

Field	Bits	Type	Description
FFPFF	0	rw	**Forward Frame due to SGF filter failed - 1: Flexible Receive Parser forwards the frame to the residual DMA Channel if stream-gate filter fails** - 0: Flexible Receive Parser drops the frame if stream-gate filter fails. 0_B **DISABLE**: When reset, Rx Parser drops the frame directly if Stream-Gate filter fails 1_B **ENABLE**: When set, Rx Parser forwards the frame to residual channel if Stream-Gate filter fails
ESPF	1	rw	**Enable Stream-Gate filtering of Preemption Frames - 1: Stream-Gate filtering is enabled for preemption frames** - 0: Stream-Gate filtering is disabled for preemption frames. 0_B **DISABLE**: Preemption frame of SGF filter is disabled 1_B **ENABLE**: Preemption frame of SGF filter is enabled
0	3:2, 31:4	r	**Reserved** Read as 0; should be written with 0.

14.6.547 Port j MTL SGF Status

This register provides Status related to Stream-Gate Filtering.

Portj_MTL_SGF_Status (j=0-1) Offset address: $110F0_H+j*2000_H$

Port j MTL SGF Status Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BTRL								SWOL	0		CGCE	0		BTRE	SWLC
r								r	r		rw	r		rw	rw

Field	Bits	Type	Description
SWLC	0	rw	**Switch to Software Owned List Complete** When this field is set, indicates that DWC_xgmac has successfully switched to the SWOL, and the SWOL bit is updated to that effect. This field is cleared when the SSWL field of SGF_Control register transitions from 0 to 1, or on a software write. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears this field. Setting 0 has no effect. 0_B **INACTIVE**: Switch to S/W owned list not Complete 1_B **ACTIVE**: Switch to S/W owned list Complete

(table continues...)

(continued)

Field	Bits	Type	Description
BTRE	1	rw	**BTR Error** When this field is set, indicates a programming error in the BTR of SWOL where the programmed value is less than current time. If the BTRL = "11111111", SWOL is not updated and software must reprogram the BTR to a value greater than current time and set the SSWL bit to re-initiate the switch to SWOL. Otherwise, if the value of BTRL is less than "11111111", SWOL is updated and this field indicates the number of iterations (of + CycleTime) taken by DWC_xgmac to update the BTR to a value greater than Current Time. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. 0_B **INACTIVE**: BTR Error not detected 1_B **ACTIVE**: BTR Error detected
CGCE	4	rw	**Constant Gate Control Error** This error occurs when the list length (LLR) is 1 and the programmed Time Interval (TI) value after the optional Left Shifting is more than or equal to the Cycle Time (CTR). These programming guidelines imply that Gates are either always Closed or always Open based on the Gate Control values; the same effect can be achieved by other simpler (non TSN) programming mechanisms. As the implementation does not support such a programming mechanism, an error is reported. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. 0_B **INACTIVE**: Constant Gate Control Error not detected 1_B **ACTIVE**: Constant Gate Control Error detected
SWOL	7	r	**Software Owned List** When this field is 0, indicates that the Gate control list number 0 is owned by software. When this field is 1, indicates that the Gate Control list 1 is owned by the software. Any reads/writes by the software (using indirect access through GCL_Control) is directed to the list indicated by this value. The inverse of this value is treated as HWOL. R/W operations performed by DWC_xgmac are directed to the list pointed by HWOL. Values: - 0x0: Gate control list number "0" is owned by software - 0x1: Gate control list number "1" is owned by software
BTRL	15:8	r	**BTR Loop Count** This field provides the number of GCL cycle times after which the switch to new SWOL GCL occurs after SSWL is set. It is determined by the value N which satisfies the equation "Current Time less than or equal to New BTR + (N * 4 * New Cycle Time)". N = "11111111" indicates that the iterations exceeded the value of 512 and DWC_xgmac is unable to update New BTR to be equal to or greater than Current Time. Software intervention is needed to update the New BTR. This field is cleared when BTRE field of this register is cleared.

(table continues...)

(continued)

Field	Bits	Type	Description
0	3:2, 6:5, 31:16	r	**Reserved** Read as 0; should be written with 0.

14.6.548 Port j MTL SGF Interrupt Enable

This register implements the Interrupt Enable bits for the various events that generate an interrupt. Bit positions have a 1- to-1 correspondence with the status bit positions in MTL_SGF_Status register.

Portj_MTL_SGF_Intr_Enable (j=0-1) Offset address: $110F4_H+j*2000_H$

Port j MTL SGF Interrupt Enable Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0											CGCE	0		IEBE	IECC
r											rw	r		rw	rw

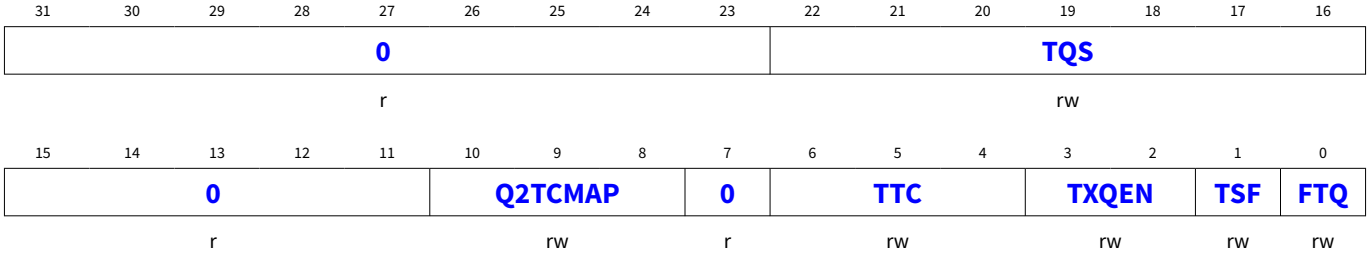
Field	Bits	Type	Description
IECC	0	rw	**Interrupt Enable for Switch List - 1: Generates interrupt when the configuration change is successful and DWC_xgmac has switched to the new list** - 0: This event does not generate an interrupt. 0_B **DISABLE**: Interrupt for Switch List is disabled 1_B **ENABLE**: Interrupt for Switch List is enabled
IEBE	1	rw	**Interrupt Enable for BTR Error - 1: Generates interrupt when the BTR Error occurs** Error is indicated in the status. - 0: This event does not generate an interrupt. 0_B **DISABLE**: Interrupt for BTR Error is disabled 1_B **ENABLE**: Interrupt for BTR Error is enabled
CGCE	4	rw	**Interrupt Enable for CGCE - 1: Generates interrupt when the Constant Gate Control Error occurs** Error is indicated in the status. - 0: This event does not generate an interrupt. 0_B **DISABLE**: Interrupt for CGCE is disabled 1_B **ENABLE**: Interrupt for CGCE is enabled
0	3:2, 31:5	r	**Reserved** Read as 0; should be written with 0.

14.6.549 Port j MTL TxQ0 Operation Mode

The Queue 0 Transmit Operation Mode register establishes the Transmit queue operating modes and commands.

Portj_MTL_TxQ0_Operation_Mode (j=0-1) Offset address: $11100_H+j*2000_H$

Port j MTL TxQ0 Operation Mode Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
FTQ	0	rw	**Flush Transmit Queue** When this bit is set, the Tx Queue Controller logic is reset to its default values. Therefore, all the data in the Tx queue is lost or flushed. This bit is internally reset when the flushing operation is complete. You must not write to this register until this bit is reset. The data which is already accepted by the MAC transmitter is not flushed. It is scheduled for transmission and results in underflow and runt packet transmission. Note: The flush operation is complete only when the Tx queue is empty. To complete this flush operation, the PHY Tx clock must be active.
TSF	1	rw	**Transmit Store and Forward** When this bit is set, the transmission starts when a full packet resides in the MTL Tx Queue. When this bit is set, the values specified in the TTC field are ignored. This bit must be changed only when the transmission is stopped.
TXQEN	3:2	rw	**Transmit Queue Enable** This field is used to enable/disable the transmit queue. - 2'b00: Not enabled - 2'b01: Audio Video Bridging - 2'b10: Enabled - 2'b11: Reserved In multiple Tx queues configuration, as the queue is disabled by default, enable the Tx Queue by programming this field in the AVB mode or default mode as desired.

(table continues...)

(continued)

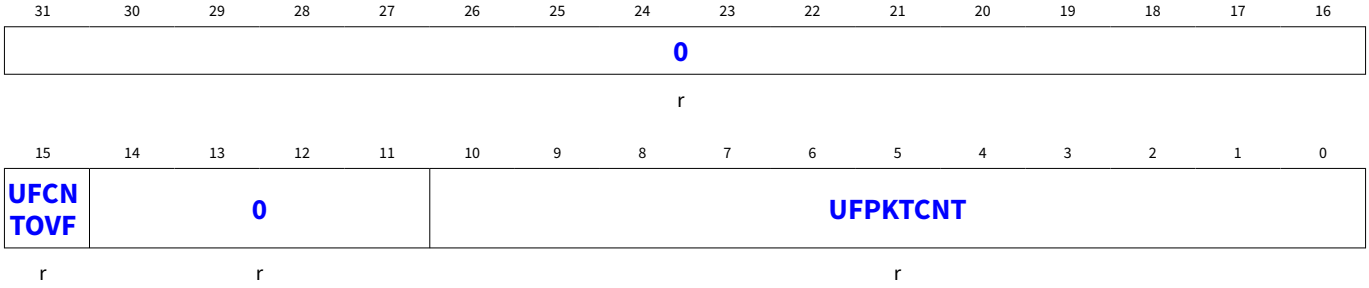
Field	Bits	Type	Description
TTC	6:4	rw	**Transmit Threshold Control** These bits control the threshold level of the MTL Tx Queue. Transmission starts when the packet size within the MTL Tx Queue is larger than the threshold. In addition, full packets with length less than the threshold are also transmitted. These bits are used only when the TSF bit is reset. - 3'b000: 64 - 3'b001: Reserved - 3'b010: 96 - 3'b011: 128 - 3'b100: 192 - 3'b101: 256 - 3'b110: 384 - 3'b111: 512
Q2TCMAP	10:8	rw	**Queue to Traffic Class Mapping** This field is used to map this Tx Queue to a traffic class (TC). - 3'b000: TC0 - 3'b001: TC1 - 3'b010: TC2 - 3'b011: TC3 - 3'b100: TC4 - 3'b101: TC5 - 3'b110: TC6 - 3'b111: TC7
TQS	22:16	rw	**Transmit Queue Size** This field indicates the size of the allocated Transmit queues in blocks of 256 bytes. The sixteenth bit is the starting bit of this field. A value of 0 indicates 256 bytes. When this queue is enabled (TXQEN=10), a minimum value of 7 (size = 2048) is recommended so that the queue can store more than a complete max-sized Ethernet packet.
0	7, 15:11, 31:23	r	**Reserved** Read as 0; should be written with 0.

14.6.550 Port j MTL TxQ0 Underflow

The Queue 0 Underflow Counter register contains the number of packets aborted because of Transmit queue underflow.

Portj_MTL_TxQ0_Underflow (j=0-1) Offset address: $11104_H+j*2000_H$

Port j MTL TxQ0 Underflow Kernel Reset value: $0000\ 0000_H$



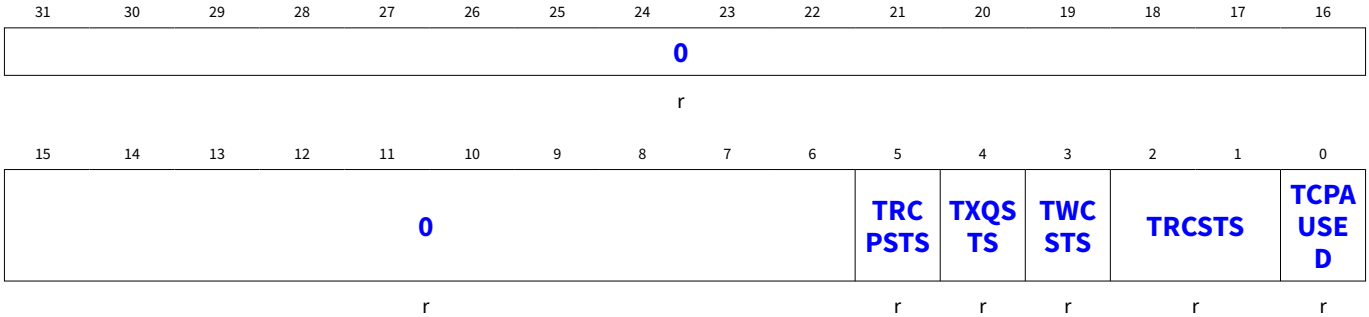
Field	Bits	Type	Description
UFPKTCNT	10:0	r	**Underflow Packet Counter** This field indicates the number of packets aborted by the controller because of Transmit Queue Underflow. This counter is incremented each time the MAC aborts an outgoing packet because of underflow. The counter is reset when this register is read with LSB of the Byte Enable for CSR access signal at 1'b1 .
UFCNTOVF	15	r	**Overflow Bit for Underflow Packet Counter** This bit is set every time the Tx Queue Underflow Packet Counter field overflows, that is, it has crossed the maximum count. In such a scenario, the overflow packet counter is reset to all-zeros and this bit indicates that the rollover happened.
0	14:11, 31:16	r	**Reserved** Read as 0; should be written with 0.

14.6.551 Port j MTL TxQ0 Debug

The Queue 0 Transmit Debug register gives the debug status of various blocks related to the Transmit Queue 0 and Traffic Class 0.

Portj_MTL_TxQ0_Debug (j=0-1) Offset address: $11108_H+j*2000_H$

Port j MTL TxQ0 Debug Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
TCPAUSED	0	r	**Traffic Class in Pause** - Reception of IEEE 802.3-2018 Pause packet

(table continues...)

(continued)

Field	Bits	Type	Description
TRCSTS	2:1	r	**MTL Tx Queue Read Controller Status** This field indicates the state of the Tx Queue Read Controller: - 2'b00: Idle state - 2'b01: Read state (transferring data to the MAC transmitter) - 2'b10: Waiting for pending Tx Status from the MAC transmitter - 2'b11: Flushing the Tx queue because of the Packet Abort request from the MAC
TWCSTS	3	r	**MTL Tx Queue Write Controller Status** When high, this bit indicates that the MTL Tx Queue Write Controller is active, and it is transferring the data to the Tx Queue.
TXQSTS	4	r	**MTL Tx Queue Not Empty Status** When this bit is high, it indicates that the MTL Tx Queue is not empty and some data is left for transmission.
TRCPSTS	5	r	**MTL TRC Pending Status** When this bit is high, it indicates that the MTL Tx Read Controller (TRC) has a pending status from MAC.
0	31:6	r	**Reserved** Read as 0; should be written with 0.

14.6.552 Port j MTL TC0 ETS Control

This register configures the transmission selection algorithm, for Traffic Class 0.

Portj_MTL_TC0_ETS_Control (j=0-1) Offset address: $11110_H+j*2000_H$

Port j MTL TC0 ETS Control Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0									0			0	CBSEN	TSA	
r									r			r	r	rw	

Field	Bits	Type	Description
TSA	1:0	rw	**Transmission Selection Algorithm** This field is used to assign a transmission selection algorithm for this traffic class. - 2'b00: Strict priority - 2'b01: CBS - 2'b10: ETS - 2'b11: Reserved

(table continues...)

(continued)

Field	Bits	Type	Description
CBSEN	2	r	**Credit Based Shaper Enable** This field indicates that TSA field of this register is enabled for CBS
0	3, 6:4, 31:7	r	**Reserved** Read as 0; should be written with 0.

14.6.553 Port j MTL TC0 Quantum Weight

The Quantum or Weights register controls the average traffic transmitted in Traffic Class 0.

Portj_MTL_TC0_Quantum_Weight (j=0-1) Offset address: $11118_H+j*2000_H$

Port j MTL TC0 Quantum Weight Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0											QW				
r											rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
QW															
rw															

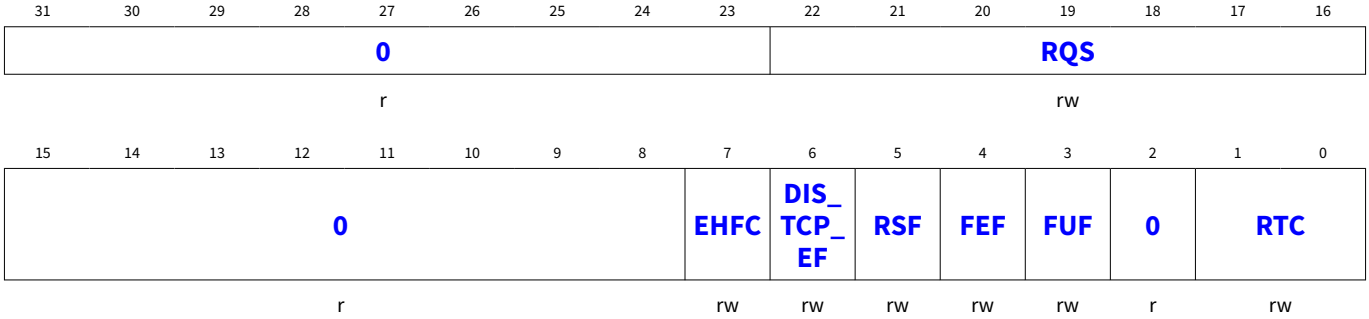
Field	Bits	Type	Description
QW	20:0	rw	**idleSlopeCredit, Quantum or Weights** - idleSlopeCredit -- When this TC is in AVB mode, this field contains the idleSlopeCredit value required for the credit-based shaper algorithm for Traffic Class 0. This is the rate of change of credit in bits per cycle (3.2 ns for 2.5/10 Gbps; 8 ns for 1000 Mbps) when the credit is increasing. The software must program this field with computed credit in bits per cycle scaled by 1024. The maximum value is portTransmitRate, that is, 0x2000 in 1000/2500 Mbps mode and 0x8000 in 10 Gbps mode. Bits[20:16] must be written to zero. This field is not valid for Traffic Class 0. - Weights -- When TSA is set to ETS and ETSALG is set to WRR algorithm for Traffic Class 1 traffic, this field contains the weight for this queue. The maximum value is 0x64. Bits[20:7] must be written to zero. This field is reserved and RO when the number of traffic classes selected is less than two. This field is reserved in corresponding Queues 8-15 register.
0	31:21	r	**Reserved** Read as 0; should be written with 0.

14.6.554 Port j MTL RxQ0 Operation Mode

The Queue 0 Receive Operation Mode register establishes the Receive queue 0 operating modes.

Portj_MTL_RxQ0_Operation_Mode (j=0-1) Offset address: $11140_H+j*2000_H$

Port j MTL RxQ0 Operation Mode Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
RTC	1:0	rw	**Receive Queue Threshold Control** These bits control the threshold level of the MTL Rx queue (in bytes): - 2'b00: 64 - 2'b01: Reserved - 2'b10: 96 - 2'b11: 128 The received packet is transferred to the application or DMA when the packet size within the MTL Rx queue is larger than the threshold. In addition, full packets with length less than the threshold are automatically transferred. The value of 11 is not applicable if the size of the configured Rx Queue is 128 bytes. This field is valid only when the RSF bit is zero. This field is ignored when the RSF bit is set to 1.
FUF	3	rw	**Forward Undersized Good Packets** - 1: Rx queue forwards the undersized good packets (with no Invalid-code/GMII/CRC/Length errors and length less than 64 bytes at Rx (X)GMII interface), including pad-bytes and CRC. - 0: Rx queue drops all packets of length less than 64 bytes.
FEF	4	rw	**Forward Error Packets** - 0: Rx queue drops packets with error status (CRC error, GMII_ER, watchdog timeout, or overflow). However, if the start byte (write) pointer of a packet is already transferred to the read controller side (in Threshold mode), the packet is not dropped. - 1: All packets except the runt error packets are forwarded to the application or DMA. If the RSF bit is set and the Rx queue overflows when a partial packet is written, the packet is dropped irrespective of the setting of this bit. However, if the RSF bit is reset and the Rx queue overflows when a partial packet is written, a partial packet might be forwarded to the application or DMA.
RSF	5	rw	**Receive Queue Store and Forward** - 1: DWC_xgmac reads a packet from the Rx queue only after the complete packet has been written to it, ignoring the RTC field of this register. - 0: Rx queue operates in the Threshold (cut-through) mode, subject to the threshold specified by the RTC field of this register.

(table continues...)

(continued)

Field	Bits	Type	Description
DIS_TCP_EF	6	rw	**Disable Dropping of TCP/IP Checksum Error Packets** - When this bit is set, the MAC does not drop the packets which only have the errors detected by the Receive Checksum Offload engine. Such packets have errors only in the encapsulated payload. There are no errors (including FCS error) in the Ethernet packet received by the MAC. - When this bit is reset, all error packets are dropped if the FEF bit is reset.
EHFC	7	rw	**Enable Hardware Flow Control** When this bit is set, the flow control signal operation, based on the fill-level of Rx queue, is enabled. When reset, the flow control operation is disabled.
RQS	22:16	rw	**Receive Queue Size** This field indicates the size of the allocated Receive queues in blocks of 256 bytes. The sixteenth bit is the starting bit of this field. A value of 0 indicates 256 bytes.When this queue is used, a minimum value of 7 (size = 2048) is recommended so that the queue can store more than a complete max-sized Ethernet packet.
0	2, 15:8, 31:23	r	**Reserved** Read as 0; should be written with 0.

14.6.555 Port j MTL RxQ0 Missed Pkt Overflow Cnt

The Queue 0 Overflow Counter register contains the counter for packets missed because of Receive queue packet flush and packets discarded because of Receive queue overflow.

Portj_MTL_RxQ0_Missed_Pkt_Overflow_Cnt (j=0-1) Offset address: $11144_H+j*2000_H$

Port j MTL RxQ0 Missed Pkt Overflow Cnt Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MISCNTOVF	0				MISPKTCNT										
r	r				r										

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OVFCNTOVF	0				OVFPKTCNT										
r	r				r										

Field	Bits	Type	Description
OVFPKTCNT	10:0	r	**Overflow Packet Counter** This field indicates the number of packets discarded by DWC_xgmac because of Receive queue overflow. This counter is incremented each time DWC_xgmac discards an incoming packet because of overflow. This counter is reset when this register bits[7:0] is read by software.

(table continues...)

(continued)

Field	Bits	Type	Description
OVFCNTOVF	15	r	**Overflow Counter Overflow Bit** When set, this bit indicates that OVFPKTCNT field crossed the maximum limit.
MISPKTCNT	26:16	r	**Missed Packet Counter** This field indicates the number of packets missed by DWC_xgmac because the host asserted packet-flush for this queue. This counter is incremented each time a packet is flushed for this queue. This counter is reset when this register's bits[23:16] is read by the software.
MISCNTOVF	31	r	**Missed Packet Counter Overflow Bit** When set, this bit indicates that the MISPKTCNT field crossed the maximum limit.
0	14:11, 30:27	r	**Reserved** Read as 0; should be written with 0.

14.6.556 Port j MTL RxQ0 Debug

The Queue 0 Receive Debug register gives the debug status of various blocks related to the Receive queue.

Portj_MTL_RxQ0_Debug (j=0-1) Offset address: $11148_H+j*2000_H$

Port j MTL RxQ0 Debug Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		PRXQ													
r		r													

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0										RXQSTS		0	RRCSTS		RWCSTS
r										r		r	r		r

Field	Bits	Type	Description
RWCSTS	0	r	**MTL Rx Queue Write Controller Active Status** When high, this bit indicates that the MTL Rx queue Write controller is active, and it is transferring a received packet to the Rx Queue.
RRCSTS	2:1	r	**MTL Rx Queue Read Controller State** This field gives the state of the Rx queue Read controller: - 2'b00: Idle state - 2'b01: Reading packet data - 2'b10: Reading packet status (or timestamp) - 2'b11: Flushing the packet data and status

(table continues...)

(continued)

Field	Bits	Type	Description
RXQSTS	5:4	r	**MTL Rx Queue Fill-Level Status** This field gives the status of the fill-level of the Rx Queue: - 2'b00: Rx Queue empty - 2'b01: Rx Queue fill-level below flow-control deactivate threshold - 2'b10: Rx Queue fill-level above flow-control activate threshold - 2'b11: Rx Queue full
PRXQ	29:16	r	**Number of Packets in Receive Queue** This field indicates the current number of packets in the Rx Queue. The theoretical maximum value for this field is 256KB/16B = 16K Packets, that is, Max_Queue_Size/Min_Packet_Size.
0	3, 15:6, 31:30	r	**Reserved** Read as 0; should be written with 0.

14.6.557 Port j MTL RxQ0 Control

The Queue 0 Receive Control register controls the receive arbitration and passing of received packets to the application.

Portj_MTL_RxQ0_Control (j=0-1) Offset address: $1114C_H+j*2000_H$

Port j MTL RxQ0 Control Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												RXQ_PKT_ARBIT	RXQ_WEGT		
r												rw	rw		

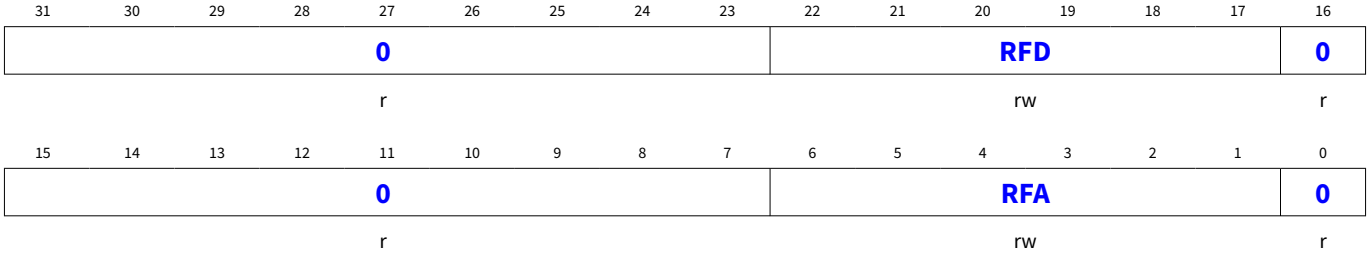
Field	Bits	Type	Description
RXQ_WEGT	2:0	rw	**Receive Queue Weight** This field indicates the weight assigned to the Rx Queue 0. The weight is used as the number of continuous PBL requests or contiguous packets (depending on the RXQ_PKT_ARBIT) allocated to the queue in one arbitration cycle.
RXQ_PKT_ARBIT	3	rw	**Receive Queue Packet Arbitration Control** - 1: MTL enables the packet-level arbitration. - 0: MTL enables the PBL based arbitration.
0	31:4	r	**Reserved** Read as 0; should be written with 0.

14.6.558 Port j MTL RxQ0 Flow Control

This register controls the activation and de-activation threshold in the queue for triggering the generation of Flow Control packets by MAC transmitter.

Portj_MTL_RxQ0_Flow_Control (j=0-1) Offset address: $11150_H+j*2000_H$

Port j MTL RxQ0 Flow Control Kernel Reset value: $0000\ 0000_H$



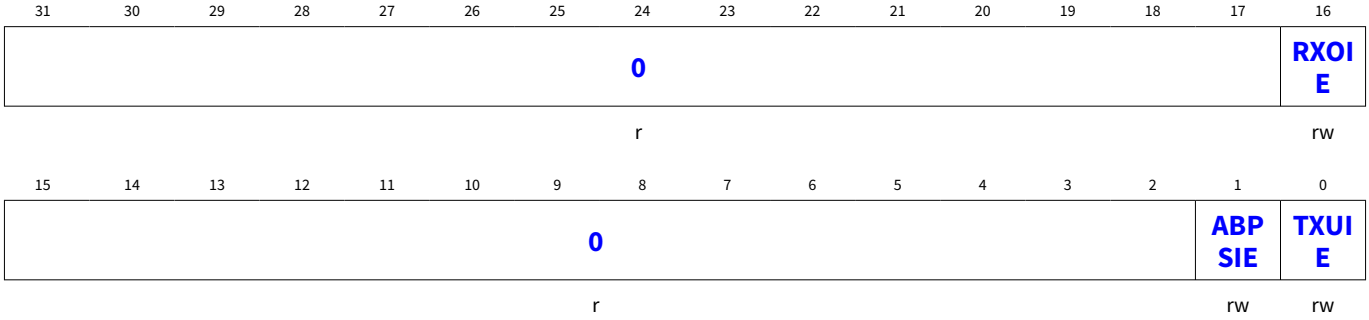
Field	Bits	Type	Description
RFA	6:1	rw	**Threshold for activating Flow Control** These bits control the threshold (fill-level of RxQ0) at which the flow control is activated - 6'b000000: RQS value minus 1 KB, that is, FULL - 1024 bytes - 6'b000001: RQS value minus 1.5 KB, that is, FULL - 1536 bytes - 6'b000010: RQS value minus 2 KB, that is, FULL - 2048 bytes. - ... - 6'b111111: RQS value minus 32.5 KB, that is, FULL - 33280 bytes The flow control is triggered when EHFC=1 and the queue size (RQS value in MTL_RxQ0_Operation Mode register) is equal to or more than 4KB. Note that the value to be programmed in RFA must be such that the threshold is positive number.
RFD	22:17	rw	**Threshold for deactivating Flow Control** These bits control the threshold (fill-level of RxQ0) at which the flow control is de-asserted once it is activated. - 6'b000000: RQS value minus 1 KB, that is, FULL - 1024 bytes - 6'b000001: RQS value minus 1.5 KB, that is, FULL - 1536 bytes - 6'b000010: RQS value minus 2 KB, that is, FULL - 2048 bytes. - ... - 6'b111111: RQS value minus 32.5 KB, that is, FULL - 33280 bytes The de-assertion is effective only after flow control is asserted as per the setting of RFA field. Note that the value to be programmed must be such that the threshold is positive number.
0	0, 16:7, 31:23	r	**Reserved** Read as 0; should be written with 0.

14.6.559 Port j MTL Q0 Interrupt Enable

This register contains the interrupt enable bits for Traffic class/Queue 0.

Portj_MTL_Q0_Interrupt_Enable (j=0-1) Offset address: $11170_H+j*2000_H$

Port j MTL Q0 Interrupt Enable Kernel Reset value: $0000\ 0000_H$



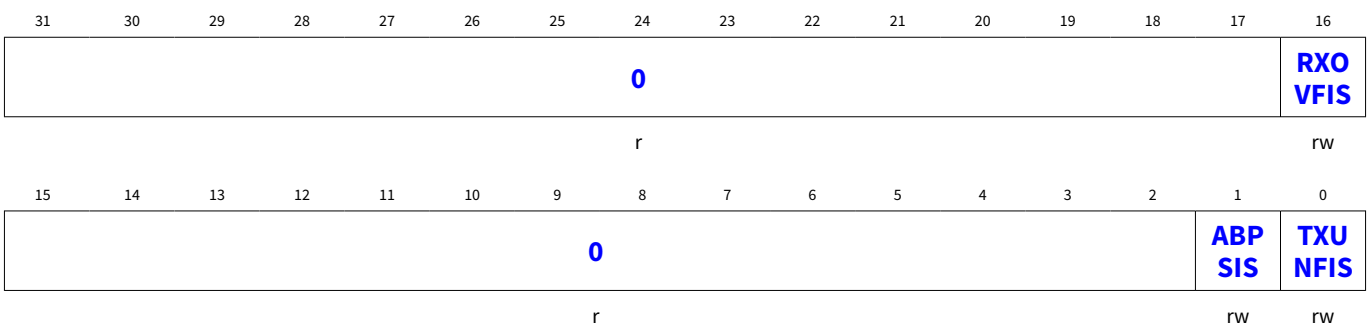
Field	Bits	Type	Description
TXUIE	0	rw	**Transmit Queue Underflow Interrupt Enable** - 1: Transmit Queue Underflow interrupt is enabled. - 0: Transmit Queue Underflow interrupt is disabled.
ABPSIE	1	rw	**Average Bits Per Slot Interrupt Enable** When this bit is set, average bits per slot interrupt is enabled for Traffic Class 0. An interrupt is generated when ABPSIS is set for this traffic class. This bit is reserved and RO when the number of traffic classes selected is less than 0.
RXOIE	16	rw	**Receive Queue Overflow Interrupt Enable** - 1: Receive Queue Overflow interrupt is enabled. - 0: Receive Queue Overflow interrupt is disabled.
0	15:2, 31:17	r	**Reserved** Read as 0; should be written with 0.

14.6.560 Port j MTL Q0 Interrupt Status

This register contains the interrupt status bits for Traffic class/Queue 0.

Portj_MTL_Q0_Interrupt_Status (j=0-1) Offset address: $11174_H+j*2000_H$

Port j MTL Q0 Interrupt Status Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
TXUNFIS	0	rw	**Transmit Queue Underflow Interrupt Status** This bit indicates that the Transmit Queue had an Underflow during packet transmission. Transmission is suspended and an Underflow Error TDES3[2] is set. This bit is cleared when the application writes a 1 to this bit.
ABPSIS	1	rw	**Average Bits Per Slot Interrupt Status** When set, this bit indicates that the MAC has updated the ABS value for this traffic class. This bit is cleared when the application writes a 1 to this bit.
RXOVFIS	16	rw	**Receive Queue Overflow Interrupt Status** This bit indicates that the Receive Queue had an Overflow during packet reception. If partial packet is transferred to the application, the overflow status is set in the corresponding Rx status. This bit is cleared when the application writes a 1 to this bit.
0	15:2, 31:17	r	**Reserved** Read as 0; should be written with 0.

14.6.561 Port j MTL TxQ1 Operation Mode

The Queue 1 Transmit Operation Mode register establishes the Transmit queue operating modes and commands.

Portj_MTL_TxQ1_Operation_Mode (j=0-1) Offset address: $11180_H+j*2000_H$

Port j MTL TxQ1 Operation Mode Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0									TQS						
r									rw						

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0					Q2TCMAP			0	TTC			TXQEN		TSF	FTQ
r					rw			r	rw			rw		rw	rw

Field	Bits	Type	Description
FTQ	0	rw	**Flush Transmit Queue** When this bit is set, the Tx Queue Controller logic is reset to its default values. Therefore, all the data in the Tx queue is lost or flushed. This bit is internally reset when the flushing operation is complete. You must not write to this register until this bit is reset. The data which is already accepted by the MAC transmitter is not flushed. It is scheduled for transmission and results in underflow and runt packet transmission. Note: The flush operation is complete only when the Tx queue is empty. To complete this flush operation, the PHY Tx clock must be active.

(table continues...)

(continued)

Field	Bits	Type	Description
TSF	1	rw	**Transmit Store and Forward** When this bit is set, the transmission starts when a full packet resides in the MTL Tx Queue. When this bit is set, the values specified in the TTC field are ignored. This bit must be changed only when the transmission is stopped.
TXQEN	3:2	rw	**Transmit Queue Enable** This field is used to enable/disable the transmit queue. - 2'b00: Not enabled - 2'b01: Audio Video Bridging - 2'b10: Enabled - 2'b11: Reserved In multiple Tx queues configuration, as the queue is disabled by default, enable the Tx Queue by programming this field in the AVB mode or default mode as desired.
TTC	6:4	rw	**Transmit Threshold Control** These bits control the threshold level of the MTL Tx Queue. Transmission starts when the packet size within the MTL Tx Queue is larger than the threshold. In addition, full packets with length less than the threshold are also transmitted. These bits are used only when the TSF bit is reset. - 3'b000: 64 - 3'b001: Reserved - 3'b010: 96 - 3'b011: 128 - 3'b100: 192 - 3'b101: 256 - 3'b110: 384 - 3'b111: 512
Q2TCMAP	10:8	rw	**Queue to Traffic Class Mapping** This field is used to map this Tx Queue to a traffic class (TC). - 3'b000: TC0 - 3'b001: TC1 - 3'b010: TC2 - 3'b011: TC3 - 3'b100: TC4 - 3'b101: TC5 - 3'b110: TC6 - 3'b111: TC7
TQS	22:16	rw	**Transmit Queue Size** This field indicates the size of the allocated Transmit queues in blocks of 256 bytes. The sixteenth bit is the starting bit of this field. A value of 0 indicates 256 bytes. When this queue is enabled (TXQEN=10), a minimum value of 7 (size = 2048) is recommended so that the queue can store more than a complete max-sized Ethernet packet.

(table continues...)

(continued)

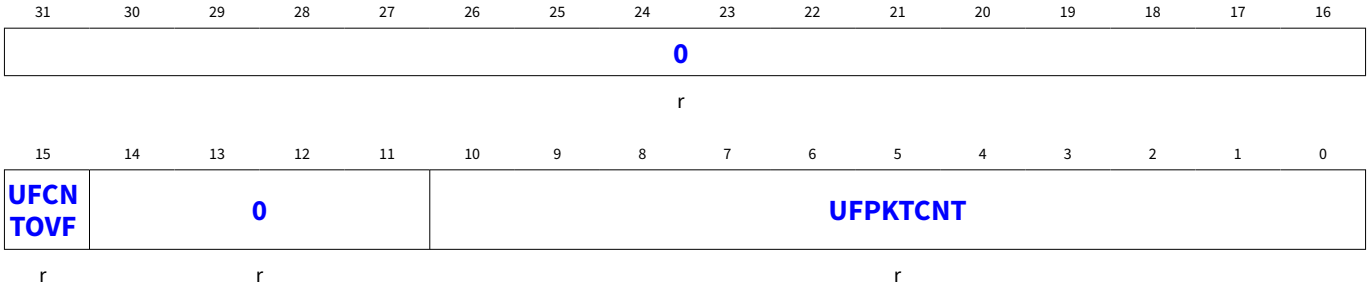
Field	Bits	Type	Description
0	7, 15:11, 31:23	r	**Reserved** Read as 0; should be written with 0.

14.6.562 Port j MTL TxQ1 Underflow

The Queue 1 Underflow Counter register contains the number of packets aborted because of Transmit queue underflow.

Portj_MTL_TxQ1_Underflow (j=0-1) Offset address: $11184_H+j*2000_H$

Port j MTL TxQ1 Underflow Kernel Reset value: $0000\ 0000_H$



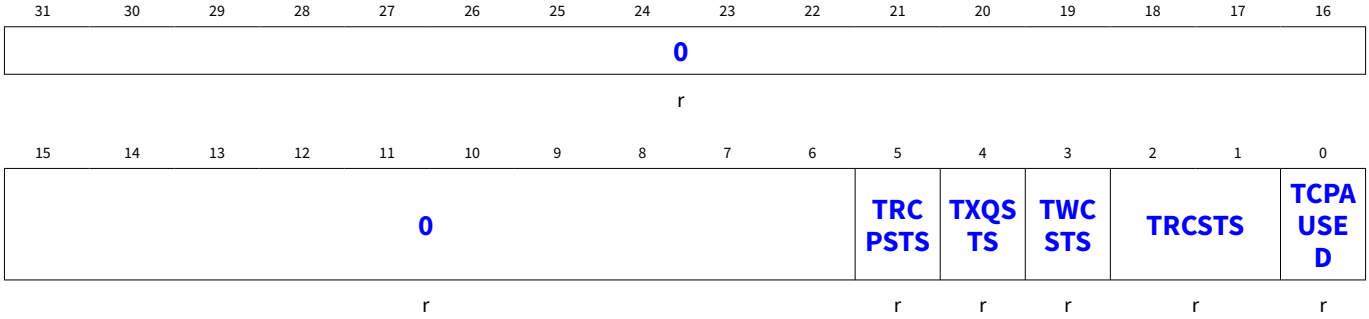
Field	Bits	Type	Description
UFPKTCNT	10:0	r	**Underflow Packet Counter** This field indicates the number of packets aborted by the controller because of Transmit Queue Underflow. This counter is incremented each time the MAC aborts an outgoing packet because of underflow. The counter is reset when this register is read with LSB of the Byte Enable for CSR access signal at 1'b1 .
UFCNTOVF	15	r	**Overflow Bit for Underflow Packet Counter** This bit is set every time the Tx Queue Underflow Packet Counter field overflows, that is, it has crossed the maximum count. In such a scenario, the overflow packet counter is reset to all-zeros and this bit indicates that the rollover happened.
0	14:11, 31:16	r	**Reserved** Read as 0; should be written with 0.

14.6.563 Port j MTL TxQ1 Debug

The Queue 1 Transmit Debug register gives the debug status of various blocks related to the Transmit Queue 1 and Traffic Class 1.

Portj_MTL_TxQ1_Debug (j=0-1) Offset address: $11188_H+j*2000_H$

Port j MTL TxQ1 Debug Kernel Reset value: $0000\ 0000_H$



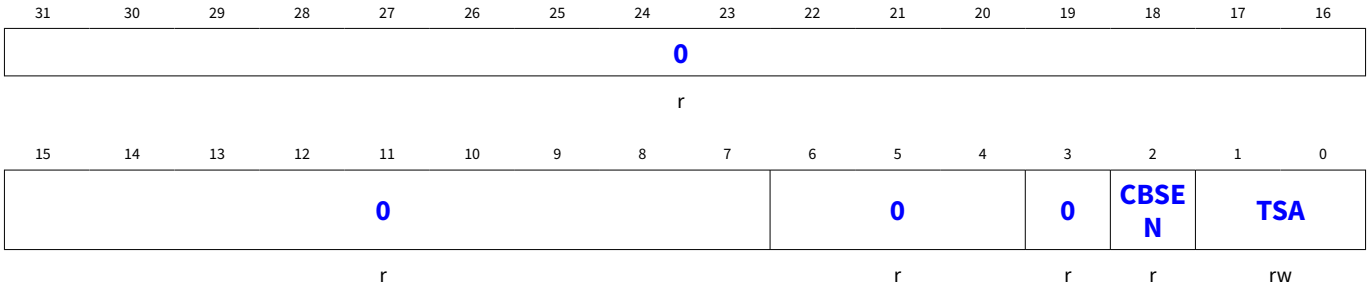
Field	Bits	Type	Description
TCPAUSED	0	r	**Traffic Class in Pause** - Reception of IEEE 802.3-2018 Pause packet
TRCSTS	2:1	r	**MTL Tx Queue Read Controller Status** This field indicates the state of the Tx Queue Read Controller: - 2'b00: Idle state - 2'b01: Read state (transferring data to the MAC transmitter) - 2'b10: Waiting for pending Tx Status from the MAC transmitter - 2'b11: Flushing the Tx queue because of the Packet Abort request from the MAC
TWCSTS	3	r	**MTL Tx Queue Write Controller Status** When high, this bit indicates that the MTL Tx Queue Write Controller is active, and it is transferring the data to the Tx Queue.
TXQSTS	4	r	**MTL Tx Queue Not Empty Status** When this bit is high, it indicates that the MTL Tx Queue is not empty and some data is left for transmission.
TRCPSTS	5	r	**MTL TRC Pending Status** When this bit is high, it indicates that the MTL Tx Read Controller (TRC) has a pending status from MAC.
0	31:6	r	**Reserved** Read as 0; should be written with 0.

14.6.564 Port j MTL TC1 ETS Control

This register configures the transmission selection algorithm, for Traffic Class 1.

Portj_MTL_TC1_ETS_Control (j=0-1) Offset address: $11190_H+j*2000_H$

Port j MTL TC1 ETS Control Kernel Reset value: $0000\ 0000_H$



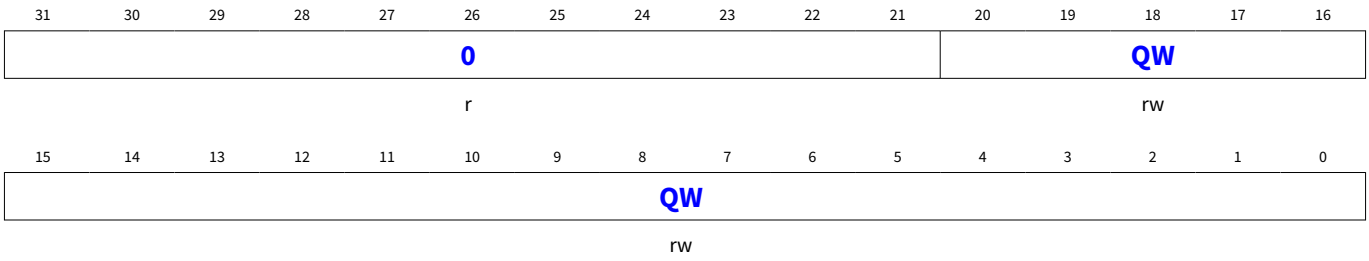
Field	Bits	Type	Description
TSA	1:0	rw	**Transmission Selection Algorithm** This field is used to assign a transmission selection algorithm for this traffic class. - 2'b00: Strict priority - 2'b01: CBS - 2'b10: ETS - 2'b11: Reserved
CBSEN	2	r	**Credit Based Shaper Enable** This field indicates that TSA field of this register is enabled for CBS
0	3, 6:4, 31:7	r	**Reserved** Read as 0; should be written with 0.

14.6.565 Port j MTL TC1 Quantum Weight

The Quantum or Weights register controls the average traffic transmitted in Traffic Class 1.

Portj_MTL_TC1_Quantum_Weight (j=0-1) Offset address: $11198_H+j*2000_H$

Port j MTL TC1 Quantum Weight Kernel Reset value: $0000\ 0000_H$



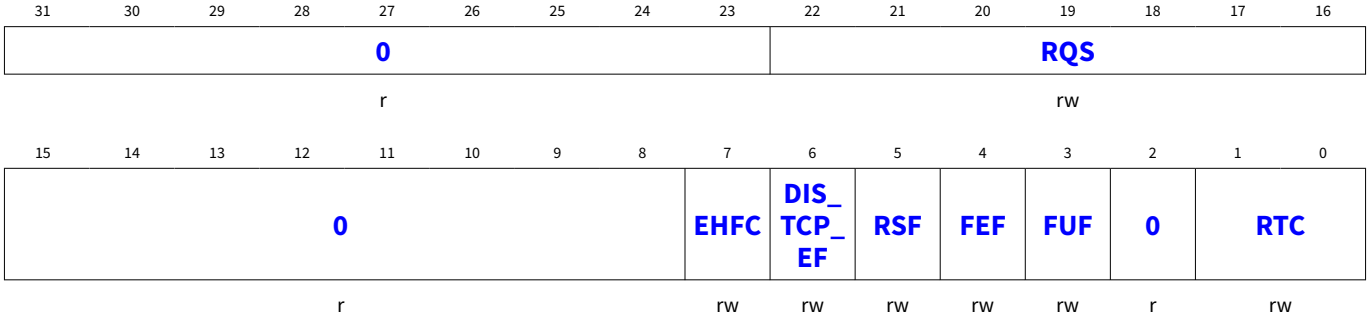
Field	Bits	Type	Description
QW	20:0	rw	**idleSlopeCredit, Quantum or Weights** - idleSlopeCredit -- When this TC is in AVB mode, this field contains the idleSlopeCredit value required for the credit-based shaper algorithm for Traffic Class 1. This is the rate of change of credit in bits per cycle (3.2 ns for 2.5/10 Gbps; 8 ns for 1000 Mbps) when the credit is increasing. The software must program this field with computed credit in bits per cycle scaled by 1024. The maximum value is portTransmitRate, that is, 0x2000 in 1000/2500 Mbps mode and 0x8000 in 10 Gbps mode. Bits[20:16] must be written to zero. This field is not valid for Traffic Class 0. - Weights -- When TSA is set to ETS and ETSALG is set to WRR algorithm for Traffic Class 1 traffic, this field contains the weight for this queue. The maximum value is 0x64. Bits[20:7] must be written to zero. This field is reserved and RO when the number of traffic classes selected is less than two. This field is reserved in corresponding Queues 8-15 register.
0	31:21	r	**Reserved** Read as 0; should be written with 0.

14.6.566 Port j MTL RxQ1 Operation Mode

The Queue 1 Receive Operation Mode register establishes the Receive queue 1 operating modes.

Portj_MTL_RxQ1_Operation_Mode (j=0-1) Offset address: $111C0_H+j*2000_H$

Port j MTL RxQ1 Operation Mode Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
RTC	1:0	rw	**Receive Queue Threshold Control** These bits control the threshold level of the MTL Rx queue (in bytes): - 2'b00: 64 - 2'b01: Reserved - 2'b10: 96 - 2'b11: 128 The received packet is transferred to the application or DMA when the packet size within the MTL Rx queue is larger than the threshold. In addition, full packets with length less than the threshold are automatically transferred. The value of 11 is not applicable if the size of the configured Rx Queue is 128 bytes. This field is valid only when the RSF bit is zero. This field is ignored when the RSF bit is set to 1.
FUF	3	rw	**Forward Undersized Good Packets** - 1: Rx queue forwards the undersized good packets (with no Invalid-code/GMII/CRC/Length errors and length less than 64 bytes at Rx (X)GMII interface), including pad-bytes and CRC. - 0: Rx queue drops all packets of length less than 64 bytes.
FEF	4	rw	**Forward Error Packets** - 0: Rx queue drops packets with error status (CRC error, GMII_ER, watchdog timeout, or overflow). However, if the start byte (write) pointer of a packet is already transferred to the read controller side (in Threshold mode), the packet is not dropped. - 1: All packets except the runt error packets are forwarded to the application or DMA. If the RSF bit is set and the Rx queue overflows when a partial packet is written, the packet is dropped irrespective of the setting of this bit. However, if the RSF bit is reset and the Rx queue overflows when a partial packet is written, a partial packet might be forwarded to the application or DMA.

(table continues...)

(continued)

Field	Bits	Type	Description
RSF	5	rw	**Receive Queue Store and Forward** - 1: DWC_xgmac reads a packet from the Rx queue only after the complete packet has been written to it, ignoring the RTC field of this register. - 0: Rx queue operates in the Threshold (cut-through) mode, subject to the threshold specified by the RTC field of this register.
DIS_TCP_EF	6	rw	**Disable Dropping of TCP/IP Checksum Error Packets** - When this bit is set, the MAC does not drop the packets which only have the errors detected by the Receive Checksum Offload engine. Such packets have errors only in the encapsulated payload. There are no errors (including FCS error) in the Ethernet packet received by the MAC. - When this bit is reset, all error packets are dropped if the FEF bit is reset.
EHFC	7	rw	**Enable Hardware Flow Control** When this bit is set, the flow control signal operation, based on the fill-level of Rx queue, is enabled. When reset, the flow control operation is disabled.
RQS	22:16	rw	**Receive Queue Size** This field indicates the size of the allocated Receive queues in blocks of 256 bytes. The sixteenth bit is the starting bit of this field. A value of 0 indicates 256 bytes.When this queue is used, a minimum value of 7 (size = 2048) is recommended so that the queue can store more than a complete max-sized Ethernet packet.
0	2, 15:8, 31:23	r	**Reserved** Read as 0; should be written with 0.

14.6.567 Port j MTL RxQ1 Missed Pkt Overflow Cnt

The Queue 1 Overflow Counter register contains the counter for packets missed because of Receive queue packet flush and packets discarded because of Receive queue overflow.

Portj_MTL_RxQ1_Missed_Pkt_Overflow_Cnt (j=0-1) Offset address: $111C4_H+j*2000_H$

Port j MTL RxQ1 Missed Pkt Overflow Cnt Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MISCNTOVF		0			MISPKTCNT										
r		r			r										

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OVFCNTOVF		0			OVFPKTCNT										
r		r			r										

Field	Bits	Type	Description
OVFPKTCNT	10:0	r	**Overflow Packet Counter** This field indicates the number of packets discarded by DWC_xgmac because of Receive queue overflow. This counter is incremented each time DWC_xgmac discards an incoming packet because of overflow. This counter is reset when this register bits[7:0] is read by software.
OVFCNTOVF	15	r	**Overflow Counter Overflow Bit** When set, this bit indicates that OVFPKTCNT field crossed the maximum limit.
MISPKTCNT	26:16	r	**Missed Packet Counter** This field indicates the number of packets missed by DWC_xgmac because the host asserted packet-flush for this queue. This counter is incremented each time a packet is flushed for this queue. This counter is reset when this register's bits[23:16] is read by the software.
MISCNTOVF	31	r	**Missed Packet Counter Overflow Bit** When set, this bit indicates that the MISPKTCNT field crossed the maximum limit.
0	14:11, 30:27	r	**Reserved** Read as 0; should be written with 0.

14.6.568 Port j MTL RxQ1 Debug

The Queue 1 Receive Debug register gives the debug status of various blocks related to the Receive queue.

Portj_MTL_RxQ1_Debug (j=0-1) Offset address: $111C8_H+j*2000_H$

Port j MTL RxQ1 Debug Kernel Reset value: $0000\ 0000_H$

Field	Bits	Type	Description
RWCSTS	0	r	**MTL Rx Queue Write Controller Active Status** When high, this bit indicates that the MTL Rx queue Write controller is active, and it is transferring a received packet to the Rx Queue.
RRCSTS	2:1	r	**MTL Rx Queue Read Controller State** This field gives the state of the Rx queue Read controller: - 2'b00: Idle state - 2'b01: Reading packet data - 2'b10: Reading packet status (or timestamp) - 2'b11: Flushing the packet data and status

(table continues...)

(continued)

Field	Bits	Type	Description
RXQSTS	5:4	r	**MTL Rx Queue Fill-Level Status** This field gives the status of the fill-level of the Rx Queue: - 2'b00: Rx Queue empty - 2'b01: Rx Queue fill-level below flow-control deactivate threshold - 2'b10: Rx Queue fill-level above flow-control activate threshold - 2'b11: Rx Queue full
PRXQ	29:16	r	**Number of Packets in Receive Queue** This field indicates the current number of packets in the Rx Queue. The theoretical maximum value for this field is 256KB/16B = 16K Packets, that is, Max_Queue_Size/Min_Packet_Size.
0	3, 15:6, 31:30	r	**Reserved** Read as 0; should be written with 0.

14.6.569 Port j MTL RxQ1 Control

The Queue 1 Receive Control register controls the receive arbitration and passing of received packets to the application.

Portj_MTL_RxQ1_Control (j=0-1) Offset address: $111CC_H+j*2000_H$

Port j MTL RxQ1 Control Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												RXQ_PKT_ARBIT	RXQ_WEGT		
r												rw	rw		

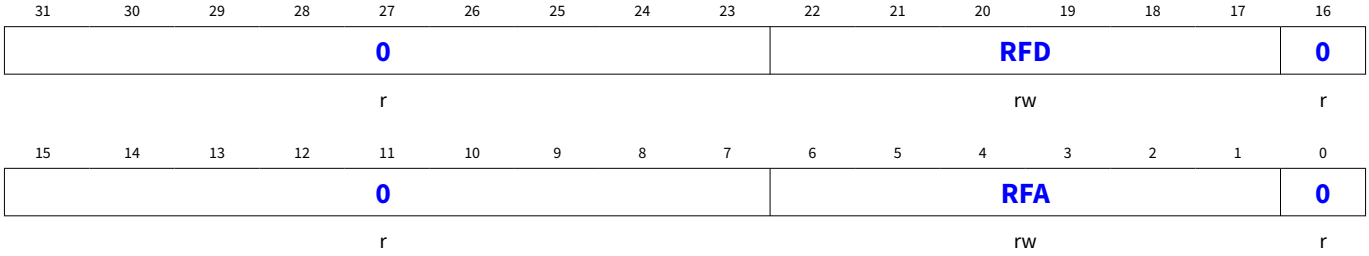
Field	Bits	Type	Description
RXQ_WEGT	2:0	rw	**Receive Queue Weight** This field indicates the weight assigned to the Rx Queue 1. The weight is used as the number of continuous PBL requests or contiguous packets (depending on the RXQ_PKT_ARBIT) allocated to the queue in one arbitration cycle.
RXQ_PKT_ARBIT	3	rw	**Receive Queue Packet Arbitration Control** - 1: MTL enables the packet-level arbitration. - 0: MTL enables the PBL based arbitration.
0	31:4	r	**Reserved** Read as 0; should be written with 0.

14.6.570 Port j MTL RxQ1 Flow Control

This register controls the activation and de-activation threshold in the queue for triggering the generation of Flow Control packets by MAC transmitter.

Portj_MTL_RxQ1_Flow_Control (j=0-1) Offset address: $111D0_H+j*2000_H$

Port j MTL RxQ1 Flow Control Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
RFA	6:1	rw	**Threshold for activating Flow Control** These bits control the threshold (fill-level of RxQ0) at which the flow control is activated - 6'b000000: RQS value minus 1 KB, that is, FULL - 1024 bytes - 6'b000001: RQS value minus 1.5 KB, that is, FULL - 1536 bytes - 6'b000010: RQS value minus 2 KB, that is, FULL - 2048 bytes. - ... - 6'b111111: RQS value minus 32.5 KB, that is, FULL - 33280 bytes The flow control is triggered when EHFC=1 and the queue size (RQS value in MTL_RxQ1_Operation Mode register) is equal to or more than 4KB. Note that the value to be programmed in RFA must be such that the threshold is positive number.
RFD	22:17	rw	**Threshold for deactivating Flow Control** These bits control the threshold (fill-level of RxQ0) at which the flow control is de-asserted once it is activated. - 6'b000000: RQS value minus 1 KB, that is, FULL - 1024 bytes - 6'b000001: RQS value minus 1.5 KB, that is, FULL - 1536 bytes - 6'b000010: RQS value minus 2 KB, that is, FULL - 2048 bytes. - ... - 6'b111111: RQS value minus 32.5 KB, that is, FULL - 33280 bytes The de-assertion is effective only after flow control is asserted as per the setting of RFA field. Note that the value to be programmed must be such that the threshold is positive number.
0	0, 16:7, 31:23	r	**Reserved** Read as 0; should be written with 0.

14.6.571 Port j MTL Q1 Interrupt Enable

This register contains the interrupt enable bits for Traffic class/Queue 1.

Portj_MTL_Q1_Interrupt_Enable (j=0-1) Offset address: $111F0_H+j*2000_H$

Port j MTL Q1 Interrupt Enable Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															RXOIE
r															rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0														ABPSIE	TXUIE
r														rw	rw

Field	Bits	Type	Description
TXUIE	0	rw	**Transmit Queue Underflow Interrupt Enable** - 1: Transmit Queue Underflow interrupt is enabled. - 0: Transmit Queue Underflow interrupt is disabled.
ABPSIE	1	rw	**Average Bits Per Slot Interrupt Enable** When this bit is set, average bits per slot interrupt is enabled for Traffic Class 1. An interrupt is generated when ABPSIS is set for this traffic class. This bit is reserved and RO when the number of traffic classes selected is less than 1.
RXOIE	16	rw	**Receive Queue Overflow Interrupt Enable** - 1: Receive Queue Overflow interrupt is enabled. - 0: Receive Queue Overflow interrupt is disabled.
0	15:2, 31:17	r	**Reserved** Read as 0; should be written with 0.

14.6.572 Port j MTL Q1 Interrupt Status

This register contains the interrupt status bits for Traffic class/Queue 1.

Portj_MTL_Q1_Interrupt_Status (j=0-1) Offset address: $111F4_H+j*2000_H$

Port j MTL Q1 Interrupt Status Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															RXOVFIS
r															rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0														ABPSIS	TXUNFIS
r														rw	rw

Field	Bits	Type	Description
TXUNFIS	0	rw	**Transmit Queue Underflow Interrupt Status** This bit indicates that the Transmit Queue had an Underflow during packet transmission. Transmission is suspended and an Underflow Error TDES3[2] is set. This bit is cleared when the application writes a 1 to this bit.
ABPSIS	1	rw	**Average Bits Per Slot Interrupt Status** When set, this bit indicates that the MAC has updated the ABS value for this traffic class. This bit is cleared when the application writes a 1 to this bit. This bit is reserved and RO when the number of traffic classes selected is less than 1.
RXOVFIS	16	rw	**Receive Queue Overflow Interrupt Status** This bit indicates that the Receive Queue had an Overflow during packet reception. If partial packet is transferred to the application, the overflow status is set in the corresponding Rx status. This bit is cleared when the application writes a 1 to this bit.
0	15:2, 31:17	r	**Reserved** Read as 0; should be written with 0.

14.6.573 Port j MTL TxQ2 Operation Mode

The Queue 2 Transmit Operation Mode register establishes the Transmit queue operating modes and commands.

Portj_MTL_TxQ2_Operation_Mode (j=0-1) Offset address: $11200_H+j*2000_H$

Port j MTL TxQ2 Operation Mode Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0									TQS						
r									rw						

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0					Q2TCMAP			0	TTC			TXQEN		TSF	FTQ
r					rw			r	rw			rw		rw	rw

Field	Bits	Type	Description
FTQ	0	rw	**Flush Transmit Queue** When this bit is set, the Tx Queue Controller logic is reset to its default values. Therefore, all the data in the Tx queue is lost or flushed. This bit is internally reset when the flushing operation is complete. You must not write to this register until this bit is reset. The data which is already accepted by the MAC transmitter is not flushed. It is scheduled for transmission and results in underflow and runt packet transmission. Note: The flush operation is complete only when the Tx queue is empty. To complete this flush operation, the PHY Tx clock must be active.

(table continues...)

(continued)

Field	Bits	Type	Description
TSF	1	rw	**Transmit Store and Forward** When this bit is set, the transmission starts when a full packet resides in the MTL Tx Queue. When this bit is set, the values specified in the TTC field are ignored. This bit must be changed only when the transmission is stopped.
TXQEN	3:2	rw	**Transmit Queue Enable** This field is used to enable/disable the transmit queue. - 2'b00: Not enabled - 2'b01: Audio Video Bridging - 2'b10: Enabled - 2'b11: Reserved In multiple Tx queues configuration, as the queue is disabled by default, enable the Tx Queue by programming this field in the AVB mode or default mode as desired.
TTC	6:4	rw	**Transmit Threshold Control** These bits control the threshold level of the MTL Tx Queue. Transmission starts when the packet size within the MTL Tx Queue is larger than the threshold. In addition, full packets with length less than the threshold are also transmitted. These bits are used only when the TSF bit is reset. - 3'b000: 64 - 3'b001: Reserved - 3'b010: 96 - 3'b011: 128 - 3'b100: 192 - 3'b101: 256 - 3'b110: 384 - 3'b111: 512
Q2TCMAP	10:8	rw	**Queue to Traffic Class Mapping** This field is used to map this Tx Queue to a traffic class (TC). - 3'b000: TC0 - 3'b001: TC1 - 3'b010: TC2 - 3'b011: TC3 - 3'b100: TC4 - 3'b101: TC5 - 3'b110: TC6 - 3'b111: TC7
TQS	22:16	rw	**Transmit Queue Size** This field indicates the size of the allocated Transmit queues in blocks of 256 bytes. The sixteenth bit is the starting bit of this field. A value of 0 indicates 256 bytes. When this queue is enabled (TXQEN=10), a minimum value of 7 (size = 2048) is recommended so that the queue can store more than a complete max-sized Ethernet packet.

(table continues...)

(continued)

Field	Bits	Type	Description
0	7, 15:11, 31:23	r	**Reserved** Read as 0; should be written with 0.

14.6.574 Port j MTL TxQ2 Underflow

The Queue 2 Underflow Counter register contains the number of packets aborted because of Transmit queue underflow.

Portj_MTL_TxQ2_Underflow (j=0-1) Offset address: $11204_H+j*2000_H$

Port j MTL TxQ2 Underflow Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UFCNTOVF	0				UFPKTCNT										
r	r				r										

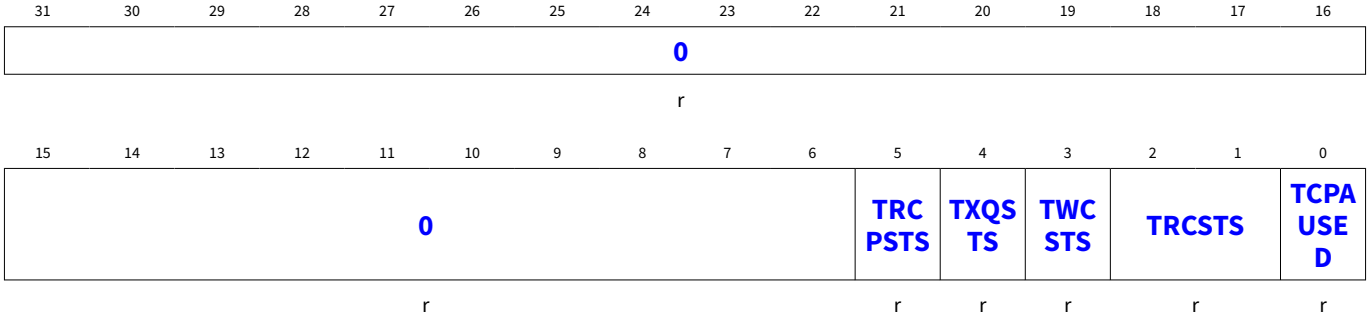
Field	Bits	Type	Description
UFPKTCNT	10:0	r	**Underflow Packet Counter** This field indicates the number of packets aborted by the controller because of Transmit Queue Underflow. This counter is incremented each time the MAC aborts an outgoing packet because of underflow. The counter is reset when this register is read with LSB of the Byte Enable for CSR access signal at 1'b1 .
UFCNTOVF	15	r	**Overflow Bit for Underflow Packet Counter** This bit is set every time the Tx Queue Underflow Packet Counter field overflows, that is, it has crossed the maximum count. In such a scenario, the overflow packet counter is reset to all-zeros and this bit indicates that the rollover happened.
0	14:11, 31:16	r	**Reserved** Read as 0; should be written with 0.

14.6.575 Port j MTL TxQ2 Debug

The Queue 2 Transmit Debug register gives the debug status of various blocks related to the Transmit Queue 2 and Traffic Class 2.

Portj_MTL_TxQ2_Debug (j=0-1) Offset address: $11208_H+j*2000_H$

Port j MTL TxQ2 Debug Kernel Reset value: $0000\ 0000_H$



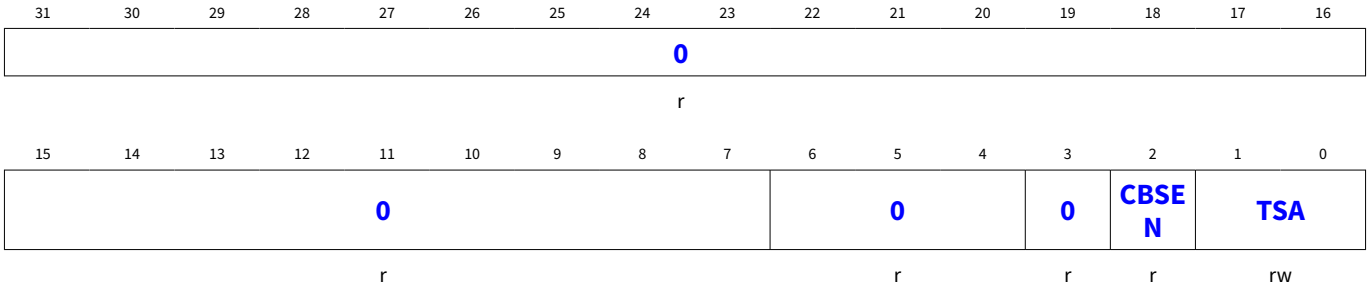
Field	Bits	Type	Description
TCPAUSED	0	r	**Traffic Class in Pause** - Reception of IEEE 802.3-2018 Pause packet
TRCSTS	2:1	r	**MTL Tx Queue Read Controller Status** This field indicates the state of the Tx Queue Read Controller: - 2'b00: Idle state - 2'b01: Read state (transferring data to the MAC transmitter) - 2'b10: Waiting for pending Tx Status from the MAC transmitter - 2'b11: Flushing the Tx queue because of the Packet Abort request from the MAC
TWCSTS	3	r	**MTL Tx Queue Write Controller Status** When high, this bit indicates that the MTL Tx Queue Write Controller is active, and it is transferring the data to the Tx Queue.
TXQSTS	4	r	**MTL Tx Queue Not Empty Status** When this bit is high, it indicates that the MTL Tx Queue is not empty and some data is left for transmission.
TRCPSTS	5	r	**MTL TRC Pending Status** When this bit is high, it indicates that the MTL Tx Read Controller (TRC) has a pending status from MAC.
0	31:6	r	**Reserved** Read as 0; should be written with 0.

14.6.576 Port j MTL TC2 ETS Control

This register configures the transmission selection algorithm, for Traffic Class 2.

Portj_MTL_TC2_ETS_Control (j=0-1) Offset address: $11210_H+j*2000_H$

Port j MTL TC2 ETS Control Kernel Reset value: $0000\ 0000_H$



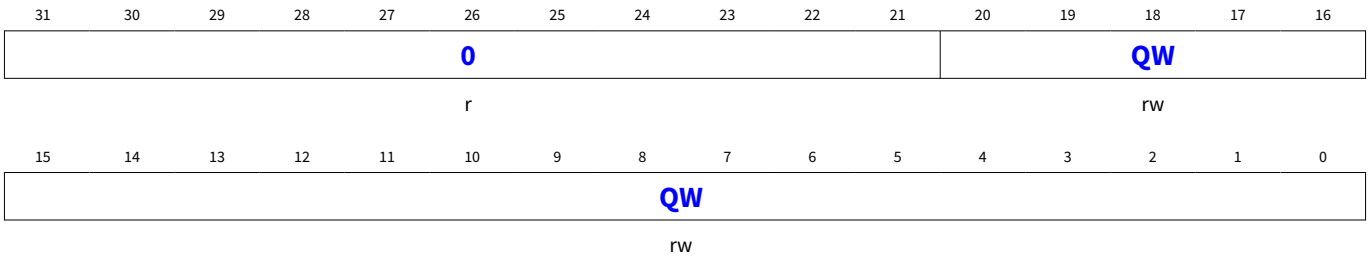
Field	Bits	Type	Description
TSA	1:0	rw	**Transmission Selection Algorithm** This field is used to assign a transmission selection algorithm for this traffic class. - 2'b00: Strict priority - 2'b01: CBS - 2'b10: ETS - 2'b11: Reserved
CBSEN	2	r	**Credit Based Shaper Enable** This field indicates that TSA field of this register is enabled for CBS
0	3, 6:4, 31:7	r	**Reserved** Read as 0; should be written with 0.

14.6.577 Port j MTL TC2 Quantum Weight

The Quantum or Weights register controls the average traffic transmitted in Traffic Class 2.

Portj_MTL_TC2_Quantum_Weight (j=0-1) Offset address: $11218_H+j*2000_H$

Port j MTL TC2 Quantum Weight Kernel Reset value: $0000\ 0000_H$



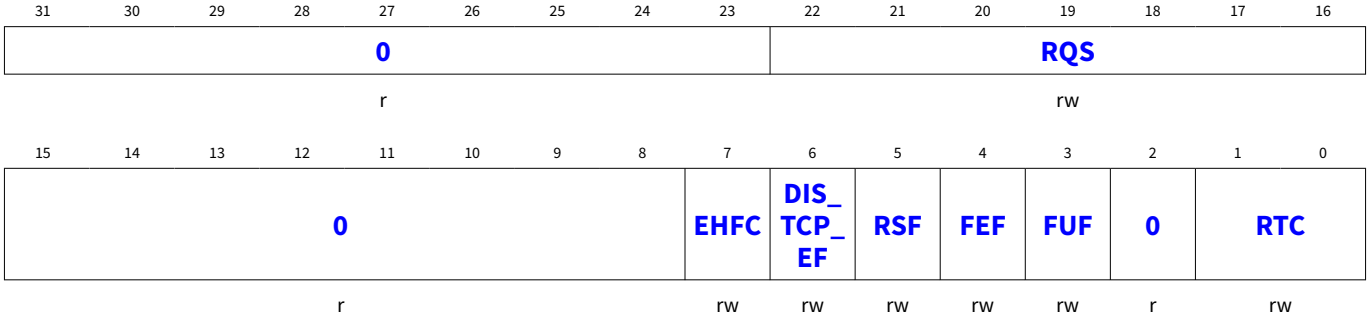
Field	Bits	Type	Description
QW	20:0	rw	**idleSlopeCredit, Quantum or Weights** - idleSlopeCredit -- When this TC is in AVB mode, this field contains the idleSlopeCredit value required for the credit-based shaper algorithm for Traffic Class 2. This is the rate of change of credit in bits per cycle (3.2 ns for 2.5/10 Gbps; 8 ns for 1000 Mbps) when the credit is increasing. The software must program this field with computed credit in bits per cycle scaled by 1024. The maximum value is portTransmitRate, that is, 0x2000 in 1000/2500 Mbps mode and 0x8000 in 10 Gbps mode. Bits[20:16] must be written to zero. This field is not valid for Traffic Class 0. - Weights -- When TSA is set to ETS and ETSALG is set to WRR algorithm for Traffic Class 1 traffic, this field contains the weight for this queue. The maximum value is 0x64. Bits[20:7] must be written to zero. This field is reserved and RO when the number of traffic classes selected is less than two. This field is reserved in corresponding Queues 8-15 register.
0	31:21	r	**Reserved** Read as 0; should be written with 0.

14.6.578 Port j MTL RxQ2 Operation Mode

The Queue 2 Receive Operation Mode register establishes the Receive queue 2 operating modes.

Portj_MTL_RxQ2_Operation_Mode (j=0-1) Offset address: $11240_H+j*2000_H$

Port j MTL RxQ2 Operation Mode Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
RTC	1:0	rw	**Receive Queue Threshold Control** These bits control the threshold level of the MTL Rx queue (in bytes): - 2'b00: 64 - 2'b01: Reserved - 2'b10: 96 - 2'b11: 128 The received packet is transferred to the application or DMA when the packet size within the MTL Rx queue is larger than the threshold. In addition, full packets with length less than the threshold are automatically transferred. The value of 11 is not applicable if the size of the configured Rx Queue is 128 bytes. This field is valid only when the RSF bit is zero. This field is ignored when the RSF bit is set to 1.
FUF	3	rw	**Forward Undersized Good Packets** - 1: Rx queue forwards the undersized good packets (with no Invalid-code/GMII/CRC/Length errors and length less than 64 bytes at Rx (X)GMII interface), including pad-bytes and CRC. - 0: Rx queue drops all packets of length less than 64 bytes.
FEF	4	rw	**Forward Error Packets** - 0: Rx queue drops packets with error status (CRC error, GMII_ER, watchdog timeout, or overflow). However, if the start byte (write) pointer of a packet is already transferred to the read controller side (in Threshold mode), the packet is not dropped. - 1: All packets except the runt error packets are forwarded to the application or DMA. If the RSF bit is set and the Rx queue overflows when a partial packet is written, the packet is dropped irrespective of the setting of this bit. However, if the RSF bit is reset and the Rx queue overflows when a partial packet is written, a partial packet might be forwarded to the application or DMA.

(table continues...)

(continued)

Field	Bits	Type	Description
RSF	5	rw	**Receive Queue Store and Forward** - 1: DWC_xgmac reads a packet from the Rx queue only after the complete packet has been written to it, ignoring the RTC field of this register. - 0: Rx queue operates in the Threshold (cut-through) mode, subject to the threshold specified by the RTC field of this register.
DIS_TCP_EF	6	rw	**Disable Dropping of TCP/IP Checksum Error Packets** - When this bit is set, the MAC does not drop the packets which only have the errors detected by the Receive Checksum Offload engine. Such packets have errors only in the encapsulated payload. There are no errors (including FCS error) in the Ethernet packet received by the MAC. - When this bit is reset, all error packets are dropped if the FEF bit is reset.
EHFC	7	rw	**Enable Hardware Flow Control** When this bit is set, the flow control signal operation, based on the fill-level of Rx queue, is enabled. When reset, the flow control operation is disabled.
RQS	22:16	rw	**Receive Queue Size** This field indicates the size of the allocated Receive queues in blocks of 256 bytes. The sixteenth bit is the starting bit of this field. A value of 0 indicates 256 bytes.When this queue is used, a minimum value of 7 (size = 2048) is recommended so that the queue can store more than a complete max-sized Ethernet packet.
0	2, 15:8, 31:23	r	**Reserved** Read as 0; should be written with 0.

14.6.579 Port j MTL RxQ2 Missed Pkt Overflow Cnt

The Queue 2 Overflow Counter register contains the counter for packets missed because of Receive queue packet flush and packets discarded because of Receive queue overflow.

Portj_MTL_RxQ2_Missed_Pkt_Overflow_Cnt (j=0-1) Offset address: $11244_H+j*2000_H$

Port j MTL RxQ2 Missed Pkt Overflow Cnt Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MISCNTOVF		0			MISPKTCNT										
r		r			r										

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OVFCNTOVF		0			OVFPKTCNT										
r		r			r										

Field	Bits	Type	Description
OVFPKTCNT	10:0	r	**Overflow Packet Counter** This field indicates the number of packets discarded by DWC_xgmac because of Receive queue overflow. This counter is incremented each time DWC_xgmac discards an incoming packet because of overflow. This counter is reset when this register bits[7:0] is read by software.
OVFCNTOVF	15	r	**Overflow Counter Overflow Bit** When set, this bit indicates that OVFPKTCNT field crossed the maximum limit.
MISPKTCNT	26:16	r	**Missed Packet Counter** This field indicates the number of packets missed by DWC_xgmac because the host asserted packet-flush for this queue. This counter is incremented each time a packet is flushed for this queue. This counter is reset when this register's bits[23:16] is read by the software.
MISCNTOVF	31	r	**Missed Packet Counter Overflow Bit** When set, this bit indicates that the MISPKTCNT field crossed the maximum limit.
0	14:11, 30:27	r	**Reserved** Read as 0; should be written with 0.

14.6.580 Port j MTL RxQ2 Debug

The Queue 2 Receive Debug register gives the debug status of various blocks related to the Receive queue.

Portj_MTL_RxQ2_Debug (j=0-1) Offset address: $11248_H+j*2000_H$

Port j MTL RxQ2 Debug Kernel Reset value: $0000\ 0000_H$

Field	Bits	Type	Description
RWCSTS	0	r	**MTL Rx Queue Write Controller Active Status** When high, this bit indicates that the MTL Rx queue Write controller is active, and it is transferring a received packet to the Rx Queue.
RRCSTS	2:1	r	**MTL Rx Queue Read Controller State** This field gives the state of the Rx queue Read controller: - 2'b00: Idle state - 2'b01: Reading packet data - 2'b10: Reading packet status (or timestamp) - 2'b11: Flushing the packet data and status

(table continues...)

(continued)

Field	Bits	Type	Description
RXQSTS	5:4	r	**MTL Rx Queue Fill-Level Status** This field gives the status of the fill-level of the Rx Queue: - 2'b00: Rx Queue empty - 2'b01: Rx Queue fill-level below flow-control deactivate threshold - 2'b10: Rx Queue fill-level above flow-control activate threshold - 2'b11: Rx Queue full
PRXQ	29:16	r	**Number of Packets in Receive Queue** This field indicates the current number of packets in the Rx Queue. The theoretical maximum value for this field is 256KB/16B = 16K Packets, that is, Max_Queue_Size/Min_Packet_Size.
0	3, 15:6, 31:30	r	**Reserved** Read as 0; should be written with 0.

14.6.581 Port j MTL RxQ2 Control

The Queue 2 Receive Control register controls the receive arbitration and passing of received packets to the application.

Portj_MTL_RxQ2_Control (j=0-1) Offset address: $1124C_H+j*2000_H$

Port j MTL RxQ2 Control Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												RXQ_PKT_ARBIT	RXQ_WEGT		
r												rw	rw		

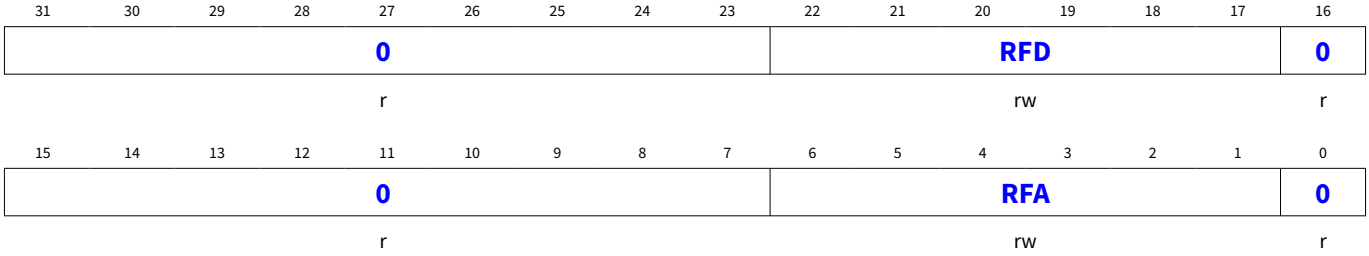
Field	Bits	Type	Description
RXQ_WEGT	2:0	rw	**Receive Queue Weight** This field indicates the weight assigned to the Rx Queue 2. The weight is used as the number of continuous PBL requests or contiguous packets (depending on the RXQ_PKT_ARBIT) allocated to the queue in one arbitration cycle.
RXQ_PKT_ARBIT	3	rw	**Receive Queue Packet Arbitration Control** - 1: MTL enables the packet-level arbitration. - 0: MTL enables the PBL based arbitration.
0	31:4	r	**Reserved** Read as 0; should be written with 0.

14.6.582 Port j MTL RxQ2 Flow Control

This register controls the activation and de-activation threshold in the queue for triggering the generation of Flow Control packets by MAC transmitter.

Portj_MTL_RxQ2_Flow_Control (j=0-1) Offset address: $11250_H+j*2000_H$

Port j MTL RxQ2 Flow Control Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
RFA	6:1	rw	**Threshold for activating Flow Control** These bits control the threshold (fill-level of RxQ0) at which the flow control is activated - 6'b000000: RQS value minus 1 KB, that is, FULL - 1024 bytes - 6'b000001: RQS value minus 1.5 KB, that is, FULL - 1536 bytes - 6'b000010: RQS value minus 2 KB, that is, FULL - 2048 bytes. - ... - 6'b111111: RQS value minus 32.5 KB, that is, FULL - 33280 bytes The flow control is triggered when EHFC=1 and the queue size (RQS value in MTL_RxQ2_Operation Mode register) is equal to or more than 4KB. Note that the value to be programmed in RFA must be such that the threshold is positive number.
RFD	22:17	rw	**Threshold for deactivating Flow Control** These bits control the threshold (fill-level of RxQ0) at which the flow control is de-asserted once it is activated. - 6'b000000: RQS value minus 1 KB, that is, FULL - 1024 bytes - 6'b000001: RQS value minus 1.5 KB, that is, FULL - 1536 bytes - 6'b000010: RQS value minus 2 KB, that is, FULL - 2048 bytes. - ... - 6'b111111: RQS value minus 32.5 KB, that is, FULL - 33280 bytes The de-assertion is effective only after flow control is asserted as per the setting of RFA field. Note that the value to be programmed must be such that the threshold is positive number.
0	0, 16:7, 31:23	r	**Reserved** Read as 0; should be written with 0.

14.6.583 Port j MTL Q2 Interrupt Enable

This register contains the interrupt enable bits for Traffic class/Queue 2.

Portj_MTL_Q2_Interrupt_Enable (j=0-1) Offset address: $11270_H+j*2000_H$

Port j MTL Q2 Interrupt Enable Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															RXOIE
r															rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0														ABPSIE	TXUIE
r														rw	rw

Field	Bits	Type	Description
TXUIE	0	rw	**Transmit Queue Underflow Interrupt Enable** - 1: Transmit Queue Underflow interrupt is enabled. - 0: Transmit Queue Underflow interrupt is disabled.
ABPSIE	1	rw	**Average Bits Per Slot Interrupt Enable** When this bit is set, average bits per slot interrupt is enabled for Traffic Class 2. An interrupt is generated when ABPSIS is set for this traffic class. This bit is reserved and RO when the number of traffic classes selected is less than 2.
RXOIE	16	rw	**Receive Queue Overflow Interrupt Enable** - 1: Receive Queue Overflow interrupt is enabled. - 0: Receive Queue Overflow interrupt is disabled.
0	15:2, 31:17	r	**Reserved** Read as 0; should be written with 0.

14.6.584 Port j MTL Q2 Interrupt Status

This register contains the interrupt status bits for Traffic class/Queue 2.

Portj_MTL_Q2_Interrupt_Status (j=0-1) Offset address: $11274_H+j*2000_H$

Port j MTL Q2 Interrupt Status Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															RXOVFIS
r															rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0														ABPSIS	TXUNFIS
r														rw	rw

Field	Bits	Type	Description
TXUNFIS	0	rw	**Transmit Queue Underflow Interrupt Status** This bit indicates that the Transmit Queue had an Underflow during packet transmission. Transmission is suspended and an Underflow Error TDES3[2] is set. This bit is cleared when the application writes a 1 to this bit.
ABPSIS	1	rw	**Average Bits Per Slot Interrupt Status** When set, this bit indicates that the MAC has updated the ABS value for this traffic class. This bit is cleared when the application writes a 1 to this bit. This bit is reserved and RO when the number of traffic classes selected is less than 2.
RXOVFIS	16	rw	**Receive Queue Overflow Interrupt Status** This bit indicates that the Receive Queue had an Overflow during packet reception. If partial packet is transferred to the application, the overflow status is set in the corresponding Rx status. This bit is cleared when the application writes a 1 to this bit.
0	15:2, 31:17	r	**Reserved** Read as 0; should be written with 0.

14.6.585 Port j MTL TxQ3 Operation Mode

The Queue 3 Transmit Operation Mode register establishes the Transmit queue operating modes and commands.

Portj_MTL_TxQ3_Operation_Mode (j=0-1) Offset address: $11280_H+j*2000_H$

Port j MTL TxQ3 Operation Mode Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0									TQS						
r									rw						

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0					Q2TCMAP			0	TTC			TXQEN		TSF	FTQ
r					rw			r	rw			rw		rw	rw

Field	Bits	Type	Description
FTQ	0	rw	**Flush Transmit Queue** When this bit is set, the Tx Queue Controller logic is reset to its default values. Therefore, all the data in the Tx queue is lost or flushed. This bit is internally reset when the flushing operation is complete. You must not write to this register until this bit is reset. The data which is already accepted by the MAC transmitter is not flushed. It is scheduled for transmission and results in underflow and runt packet transmission. Note: The flush operation is complete only when the Tx queue is empty. To complete this flush operation, the PHY Tx clock must be active.

(table continues...)

(continued)

Field	Bits	Type	Description
TSF	1	rw	**Transmit Store and Forward** When this bit is set, the transmission starts when a full packet resides in the MTL Tx Queue. When this bit is set, the values specified in the TTC field are ignored. This bit must be changed only when the transmission is stopped.
TXQEN	3:2	rw	**Transmit Queue Enable** This field is used to enable/disable the transmit queue. - 2'b00: Not enabled - 2'b01: Audio Video Bridging - 2'b10: Enabled - 2'b11: Reserved In multiple Tx queues configuration, as the queue is disabled by default, enable the Tx Queue by programming this field in the AVB mode or default mode as desired.
TTC	6:4	rw	**Transmit Threshold Control** These bits control the threshold level of the MTL Tx Queue. Transmission starts when the packet size within the MTL Tx Queue is larger than the threshold. In addition, full packets with length less than the threshold are also transmitted. These bits are used only when the TSF bit is reset. - 3'b000: 64 - 3'b001: Reserved - 3'b010: 96 - 3'b011: 128 - 3'b100: 192 - 3'b101: 256 - 3'b110: 384 - 3'b111: 512
Q2TCMAP	10:8	rw	**Queue to Traffic Class Mapping** This field is used to map this Tx Queue to a traffic class (TC). - 3'b000: TC0 - 3'b001: TC1 - 3'b010: TC2 - 3'b011: TC3 - 3'b100: TC4 - 3'b101: TC5 - 3'b110: TC6 - 3'b111: TC7
TQS	22:16	rw	**Transmit Queue Size** This field indicates the size of the allocated Transmit queues in blocks of 256 bytes. The sixteenth bit is the starting bit of this field. A value of 0 indicates 256 bytes. When this queue is enabled (TXQEN=10), a minimum value of 7 (size = 2048) is recommended so that the queue can store more than a complete max-sized Ethernet packet.

(table continues...)

(continued)

Field	Bits	Type	Description
0	7, 15:11, 31:23	r	**Reserved** Read as 0; should be written with 0.

14.6.586 Port j MTL TxQ3 Underflow

The Queue 3 Underflow Counter register contains the number of packets aborted because of Transmit queue underflow.

Portj_MTL_TxQ3_Underflow (j=0-1) Offset address: $11284_H+j*2000_H$

Port j MTL TxQ3 Underflow Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UFCNTOVF	0				UFPKTCNT										
r	r				r										

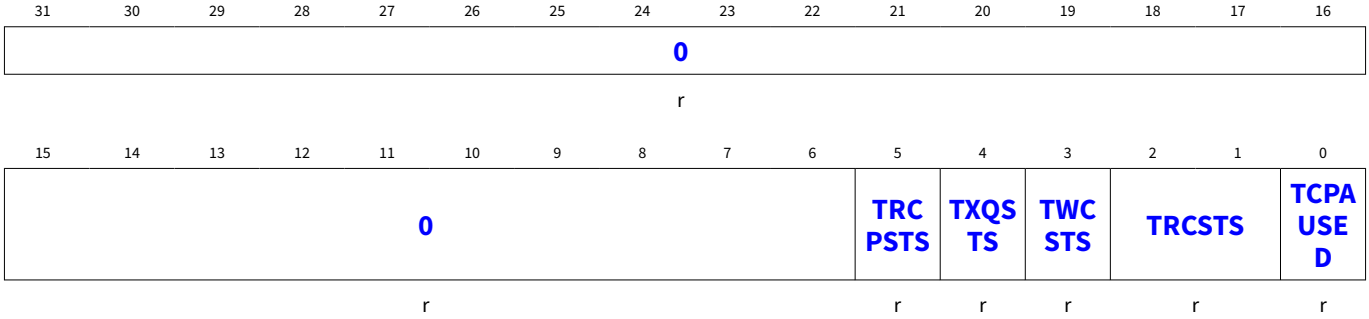
Field	Bits	Type	Description
UFPKTCNT	10:0	r	**Underflow Packet Counter** This field indicates the number of packets aborted by the controller because of Transmit Queue Underflow. This counter is incremented each time the MAC aborts an outgoing packet because of underflow. The counter is reset when this register is read with LSB of the Byte Enable for CSR access signal at 1'b1 .
UFCNTOVF	15	r	**Overflow Bit for Underflow Packet Counter** This bit is set every time the Tx Queue Underflow Packet Counter field overflows, that is, it has crossed the maximum count. In such a scenario, the overflow packet counter is reset to all-zeros and this bit indicates that the rollover happened.
0	14:11, 31:16	r	**Reserved** Read as 0; should be written with 0.

14.6.587 Port j MTL TxQ3 Debug

The Queue 3 Transmit Debug register gives the debug status of various blocks related to the Transmit Queue 3 and Traffic Class 3.

Portj_MTL_TxQ3_Debug (j=0-1) Offset address: $11288_H+j*2000_H$

Port j MTL TxQ3 Debug Kernel Reset value: $0000\ 0000_H$



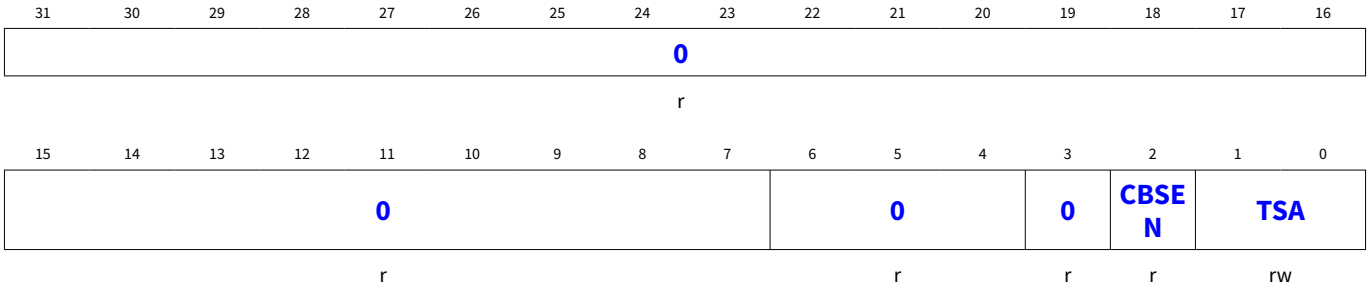
Field	Bits	Type	Description
TCPAUSED	0	r	**Traffic Class in Pause** - Reception of IEEE 802.3-2018 Pause packet
TRCSTS	2:1	r	**MTL Tx Queue Read Controller Status** This field indicates the state of the Tx Queue Read Controller: - 2'b00: Idle state - 2'b01: Read state (transferring data to the MAC transmitter) - 2'b10: Waiting for pending Tx Status from the MAC transmitter - 2'b11: Flushing the Tx queue because of the Packet Abort request from the MAC
TWCSTS	3	r	**MTL Tx Queue Write Controller Status** When high, this bit indicates that the MTL Tx Queue Write Controller is active, and it is transferring the data to the Tx Queue.
TXQSTS	4	r	**MTL Tx Queue Not Empty Status** When this bit is high, it indicates that the MTL Tx Queue is not empty and some data is left for transmission.
TRCPSTS	5	r	**MTL TRC Pending Status** When this bit is high, it indicates that the MTL Tx Read Controller (TRC) has a pending status from MAC.
0	31:6	r	**Reserved** Read as 0; should be written with 0.

14.6.588 Port j MTL TC3 ETS Control

This register configures the transmission selection algorithm, for Traffic Class 3.

Portj_MTL_TC3_ETS_Control (j=0-1) Offset address: $11290_H+j*2000_H$

Port j MTL TC3 ETS Control Kernel Reset value: $0000\ 0000_H$



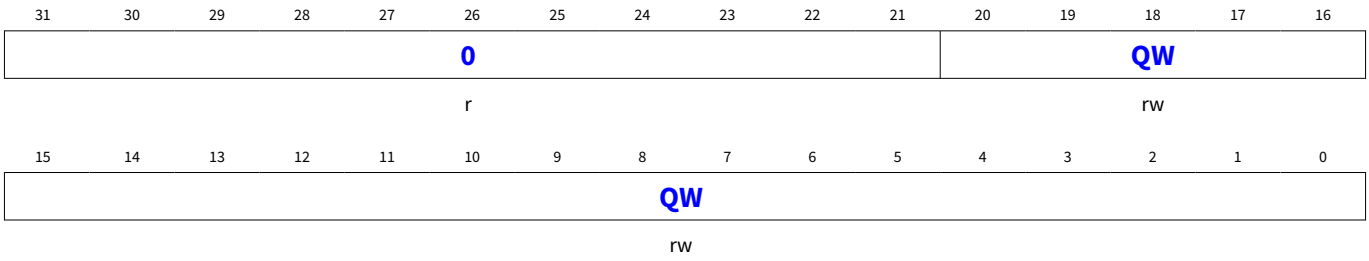
Field	Bits	Type	Description
TSA	1:0	rw	**Transmission Selection Algorithm** This field is used to assign a transmission selection algorithm for this traffic class. - 2'b00: Strict priority - 2'b01: CBS - 2'b10: ETS - 2'b11: Reserved
CBSEN	2	r	**Credit Based Shaper Enable** This field indicates that TSA field of this register is enabled for CBS
0	3, 6:4, 31:7	r	**Reserved** Read as 0; should be written with 0.

14.6.589 Port j MTL TC3 Quantum Weight

The Quantum or Weights register controls the average traffic transmitted in Traffic Class 3.

Portj_MTL_TC3_Quantum_Weight (j=0-1) Offset address: $11298_H+j*2000_H$

Port j MTL TC3 Quantum Weight Kernel Reset value: $0000\ 0000_H$



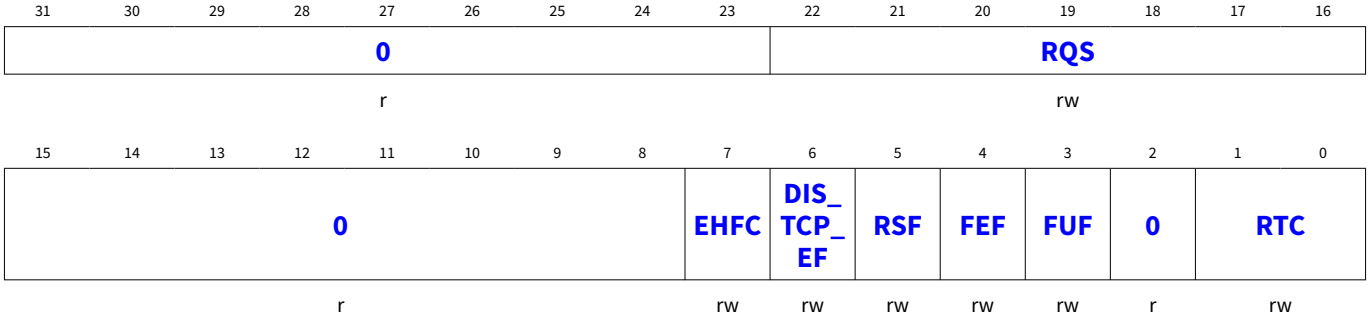
Field	Bits	Type	Description
QW	20:0	rw	**idleSlopeCredit, Quantum or Weights** - idleSlopeCredit -- When this TC is in AVB mode, this field contains the idleSlopeCredit value required for the credit-based shaper algorithm for Traffic Class 3. This is the rate of change of credit in bits per cycle (3.2 ns for 2.5/10 Gbps; 8 ns for 1000 Mbps) when the credit is increasing. The software must program this field with computed credit in bits per cycle scaled by 1024. The maximum value is portTransmitRate, that is, 0x2000 in 1000/2500 Mbps mode and 0x8000 in 10 Gbps mode. Bits[20:16] must be written to zero. This field is not valid for Traffic Class 0. - Weights -- When TSA is set to ETS and ETSALG is set to WRR algorithm for Traffic Class 1 traffic, this field contains the weight for this queue. The maximum value is 0x64. Bits[20:7] must be written to zero. This field is reserved and RO when the number of traffic classes selected is less than two. This field is reserved in corresponding Queues 8-15 register.
0	31:21	r	**Reserved** Read as 0; should be written with 0.

14.6.590 Port j MTL RxQ3 Operation Mode

The Queue 3 Receive Operation Mode register establishes the Receive queue 3 operating modes.

Portj_MTL_RxQ3_Operation_Mode (j=0-1) Offset address: $112C0_H+j*2000_H$

Port j MTL RxQ3 Operation Mode Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
RTC	1:0	rw	**Receive Queue Threshold Control** These bits control the threshold level of the MTL Rx queue (in bytes): - 2'b00: 64 - 2'b01: Reserved - 2'b10: 96 - 2'b11: 128 The received packet is transferred to the application or DMA when the packet size within the MTL Rx queue is larger than the threshold. In addition, full packets with length less than the threshold are automatically transferred. The value of 11 is not applicable if the size of the configured Rx Queue is 128 bytes. This field is valid only when the RSF bit is zero. This field is ignored when the RSF bit is set to 1.
FUF	3	rw	**Forward Undersized Good Packets** - 1: Rx queue forwards the undersized good packets (with no Invalid-code/GMII/CRC/Length errors and length less than 64 bytes at Rx (X)GMII interface), including pad-bytes and CRC. - 0: Rx queue drops all packets of length less than 64 bytes.
FEF	4	rw	**Forward Error Packets** - 0: Rx queue drops packets with error status (CRC error, GMII_ER, watchdog timeout, or overflow). However, if the start byte (write) pointer of a packet is already transferred to the read controller side (in Threshold mode), the packet is not dropped. - 1: All packets except the runt error packets are forwarded to the application or DMA. If the RSF bit is set and the Rx queue overflows when a partial packet is written, the packet is dropped irrespective of the setting of this bit. However, if the RSF bit is reset and the Rx queue overflows when a partial packet is written, a partial packet might be forwarded to the application or DMA.

(table continues...)

(continued)

Field	Bits	Type	Description
RSF	5	rw	**Receive Queue Store and Forward** - 1: DWC_xgmac reads a packet from the Rx queue only after the complete packet has been written to it, ignoring the RTC field of this register. - 0: Rx queue operates in the Threshold (cut-through) mode, subject to the threshold specified by the RTC field of this register.
DIS_TCP_EF	6	rw	**Disable Dropping of TCP/IP Checksum Error Packets** - When this bit is set, the MAC does not drop the packets which only have the errors detected by the Receive Checksum Offload engine. Such packets have errors only in the encapsulated payload. There are no errors (including FCS error) in the Ethernet packet received by the MAC. - When this bit is reset, all error packets are dropped if the FEF bit is reset.
EHFC	7	rw	**Enable Hardware Flow Control** When this bit is set, the flow control signal operation, based on the fill-level of Rx queue, is enabled. When reset, the flow control operation is disabled.
RQS	22:16	rw	**Receive Queue Size** This field indicates the size of the allocated Receive queues in blocks of 256 bytes. The sixteenth bit is the starting bit of this field. A value of 0 indicates 256 bytes.When this queue is used, a minimum value of 7 (size = 2048) is recommended so that the queue can store more than a complete max-sized Ethernet packet.
0	2, 15:8, 31:23	r	**Reserved** Read as 0; should be written with 0.

14.6.591 Port j MTL RxQ3 Missed Pkt Overflow Cnt

The Queue 3 Overflow Counter register contains the counter for packets missed because of Receive queue packet flush and packets discarded because of Receive queue overflow.

Portj_MTL_RxQ3_Missed_Pkt_Overflow_Cnt (j=0-1) Offset address: $112C4_H+j*2000_H$

Port j MTL RxQ3 Missed Pkt Overflow Cnt Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MISCNTOVF	0				MISPKTCNT										
r	r				r										

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OVFCNTOVF	0				OVFPKTCNT										
r	r				r										

Field	Bits	Type	Description
OVFPKTCNT	10:0	r	**Overflow Packet Counter** This field indicates the number of packets discarded by DWC_xgmac because of Receive queue overflow. This counter is incremented each time DWC_xgmac discards an incoming packet because of overflow. This counter is reset when this register bits[7:0] is read by software.
OVFCNTOVF	15	r	**Overflow Counter Overflow Bit** When set, this bit indicates that OVFPKTCNT field crossed the maximum limit.
MISPKTCNT	26:16	r	**Missed Packet Counter** This field indicates the number of packets missed by DWC_xgmac because the host asserted packet-flush for this queue. This counter is incremented each time a packet is flushed for this queue. This counter is reset when this register's bits[23:16] is read by the software.
MISCNTOVF	31	r	**Missed Packet Counter Overflow Bit** When set, this bit indicates that the MISPKTCNT field crossed the maximum limit.
0	14:11, 30:27	r	**Reserved** Read as 0; should be written with 0.

14.6.592 Port j MTL RxQ3 Debug

The Queue 3 Receive Debug register gives the debug status of various blocks related to the Receive queue.

Portj_MTL_RxQ3_Debug (j=0-1) Offset address: $112C8_H+j*2000_H$

Port j MTL RxQ3 Debug Kernel Reset value: $0000\ 0000_H$

Field	Bits	Type	Description
RWCSTS	0	r	**MTL Rx Queue Write Controller Active Status** When high, this bit indicates that the MTL Rx queue Write controller is active, and it is transferring a received packet to the Rx Queue.
RRCSTS	2:1	r	**MTL Rx Queue Read Controller State** This field gives the state of the Rx queue Read controller: - 2'b00: Idle state - 2'b01: Reading packet data - 2'b10: Reading packet status (or timestamp) - 2'b11: Flushing the packet data and status

(table continues...)

(continued)

Field	Bits	Type	Description
RXQSTS	5:4	r	**MTL Rx Queue Fill-Level Status** This field gives the status of the fill-level of the Rx Queue: - 2'b00: Rx Queue empty - 2'b01: Rx Queue fill-level below flow-control deactivate threshold - 2'b10: Rx Queue fill-level above flow-control activate threshold - 2'b11: Rx Queue full
PRXQ	29:16	r	**Number of Packets in Receive Queue** This field indicates the current number of packets in the Rx Queue. The theoretical maximum value for this field is 256KB/16B = 16K Packets, that is, Max_Queue_Size/Min_Packet_Size.
0	3, 15:6, 31:30	r	**Reserved** Read as 0; should be written with 0.

14.6.593 Port j MTL RxQ3 Control

The Queue 3 Receive Control register controls the receive arbitration and passing of received packets to the application.

Portj_MTL_RxQ3_Control (j=0-1) Offset address: $112CC_H+j*2000_H$

Port j MTL RxQ3 Control Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												RXQ_PKT_ARBIT	RXQ_WEGT		
r												rw	rw		

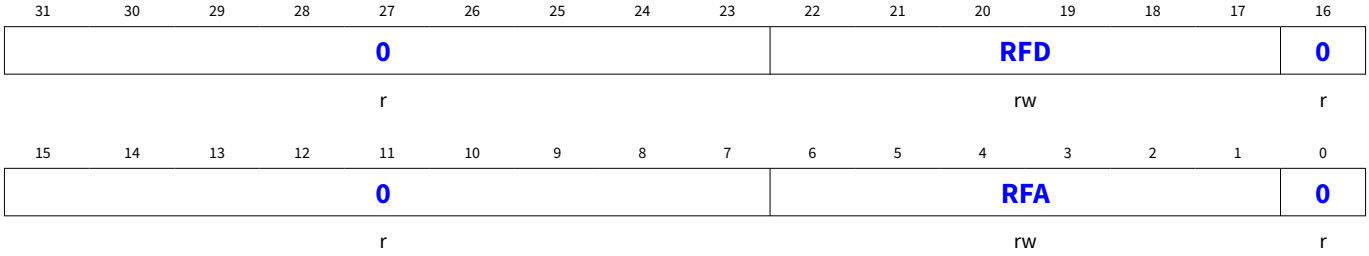
Field	Bits	Type	Description
RXQ_WEGT	2:0	rw	**Receive Queue Weight** This field indicates the weight assigned to the Rx Queue 3. The weight is used as the number of continuous PBL requests or contiguous packets (depending on the RXQ_PKT_ARBIT) allocated to the queue in one arbitration cycle.
RXQ_PKT_ARBIT	3	rw	**Receive Queue Packet Arbitration Control** - 1: MTL enables the packet-level arbitration. - 0: MTL enables the PBL based arbitration.
0	31:4	r	**Reserved** Read as 0; should be written with 0.

14.6.594 Port j MTL RxQ3 Flow Control

This register controls the activation and de-activation threshold in the queue for triggering the generation of Flow Control packets by MAC transmitter.

Portj_MTL_RxQ3_Flow_Control (j=0-1) Offset address: $112D0_H+j*2000_H$

Port j MTL RxQ3 Flow Control Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
RFA	6:1	rw	**Threshold for activating Flow Control** These bits control the threshold (fill-level of RxQ0) at which the flow control is activated - 6'b000000: RQS value minus 1 KB, that is, FULL - 1024 bytes - 6'b000001: RQS value minus 1.5 KB, that is, FULL - 1536 bytes - 6'b000010: RQS value minus 2 KB, that is, FULL - 2048 bytes. - ... - 6'b111111: RQS value minus 32.5 KB, that is, FULL - 33280 bytes The flow control is triggered when EHFC=1 and the queue size (RQS value in MTL_RxQ3_Operation Mode register) is equal to or more than 4KB. Note that the value to be programmed in RFA must be such that the threshold is positive number.
RFD	22:17	rw	**Threshold for deactivating Flow Control** These bits control the threshold (fill-level of RxQ0) at which the flow control is de-asserted once it is activated. - 6'b000000: RQS value minus 1 KB, that is, FULL - 1024 bytes - 6'b000001: RQS value minus 1.5 KB, that is, FULL - 1536 bytes - 6'b000010: RQS value minus 2 KB, that is, FULL - 2048 bytes. - ... - 6'b111111: RQS value minus 32.5 KB, that is, FULL - 33280 bytes The de-assertion is effective only after flow control is asserted as per the setting of RFA field. Note that the value to be programmed must be such that the threshold is positive number.
0	0, 16:7, 31:23	r	**Reserved** Read as 0; should be written with 0.

14.6.595 Port j MTL Q3 Interrupt Enable

This register contains the interrupt enable bits for Traffic class/Queue 3.

Portj_MTL_Q3_Interrupt_Enable (j=0-1) Offset address: $112F0_H+j*2000_H$

Port j MTL Q3 Interrupt Enable Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															RXOIE
r															rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0														ABPSIE	TXUIE
r														rw	rw

Field	Bits	Type	Description
TXUIE	0	rw	**Transmit Queue Underflow Interrupt Enable** - 1: Transmit Queue Underflow interrupt is enabled. - 0: Transmit Queue Underflow interrupt is disabled.
ABPSIE	1	rw	**Average Bits Per Slot Interrupt Enable** When this bit is set, average bits per slot interrupt is enabled for Traffic Class 3. An interrupt is generated when ABPSIS is set for this traffic class. This bit is reserved and RO when the number of traffic classes selected is less than 3.
RXOIE	16	rw	**Receive Queue Overflow Interrupt Enable** - 1: Receive Queue Overflow interrupt is enabled. - 0: Receive Queue Overflow interrupt is disabled.
0	15:2, 31:17	r	**Reserved** Read as 0; should be written with 0.

14.6.596 Port j MTL Q3 Interrupt Status

This register contains the interrupt status bits for Traffic class/Queue 3.

Portj_MTL_Q3_Interrupt_Status (j=0-1) Offset address: $112F4_H+j*2000_H$

Port j MTL Q3 Interrupt Status Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															RXOVFIS
r															rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0														ABPSIS	TXUNFIS
r														rw	rw

Field	Bits	Type	Description
TXUNFIS	0	rw	**Transmit Queue Underflow Interrupt Status** This bit indicates that the Transmit Queue had an Underflow during packet transmission. Transmission is suspended and an Underflow Error TDES3[2] is set. This bit is cleared when the application writes a 1 to this bit.
ABPSIS	1	rw	**Average Bits Per Slot Interrupt Status** When set, this bit indicates that the MAC has updated the ABS value for this traffic class. This bit is cleared when the application writes a 1 to this bit. This bit is reserved and RO when the number of traffic classes selected is less than 3.
RXOVFIS	16	rw	**Receive Queue Overflow Interrupt Status** This bit indicates that the Receive Queue had an Overflow during packet reception. If partial packet is transferred to the application, the overflow status is set in the corresponding Rx status. This bit is cleared when the application writes a 1 to this bit.
0	15:2, 31:17	r	**Reserved** Read as 0; should be written with 0.

14.6.597 Port j MTL TxQ4 Operation Mode

The Queue 4 Transmit Operation Mode register establishes the Transmit queue operating modes and commands.

Portj_MTL_TxQ4_Operation_Mode (j=0-1) Offset address: $11300_H+j*2000_H$

Port j MTL TxQ4 Operation Mode Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0									TQS						
r									rw						

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0					Q2TCMAP			0	TTC			TXQEN		TSF	FTQ
r					rw			r	rw			rw		rw	rw

Field	Bits	Type	Description
FTQ	0	rw	**Flush Transmit Queue** When this bit is set, the Tx Queue Controller logic is reset to its default values. Therefore, all the data in the Tx queue is lost or flushed. This bit is internally reset when the flushing operation is complete. You must not write to this register until this bit is reset. The data which is already accepted by the MAC transmitter is not flushed. It is scheduled for transmission and results in underflow and runt packet transmission. Note: The flush operation is complete only when the Tx queue is empty. To complete this flush operation, the PHY Tx clock must be active.

(table continues...)

(continued)

Field	Bits	Type	Description
TSF	1	rw	**Transmit Store and Forward** When this bit is set, the transmission starts when a full packet resides in the MTL Tx Queue. When this bit is set, the values specified in the TTC field are ignored. This bit must be changed only when the transmission is stopped.
TXQEN	3:2	rw	**Transmit Queue Enable** This field is used to enable/disable the transmit queue. - 2'b00: Not enabled - 2'b01: Audio Video Bridging - 2'b10: Enabled - 2'b11: Reserved In multiple Tx queues configuration, as the queue is disabled by default, enable the Tx Queue by programming this field in the AVB mode or default mode as desired.
TTC	6:4	rw	**Transmit Threshold Control** These bits control the threshold level of the MTL Tx Queue. Transmission starts when the packet size within the MTL Tx Queue is larger than the threshold. In addition, full packets with length less than the threshold are also transmitted. These bits are used only when the TSF bit is reset. - 3'b000: 64 - 3'b001: Reserved - 3'b010: 96 - 3'b011: 128 - 3'b100: 192 - 3'b101: 256 - 3'b110: 384 - 3'b111: 512
Q2TCMAP	10:8	rw	**Queue to Traffic Class Mapping** This field is used to map this Tx Queue to a traffic class (TC). - 3'b000: TC0 - 3'b001: TC1 - 3'b010: TC2 - 3'b011: TC3 - 3'b100: TC4 - 3'b101: TC5 - 3'b110: TC6 - 3'b111: TC7
TQS	22:16	rw	**Transmit Queue Size** This field indicates the size of the allocated Transmit queues in blocks of 256 bytes. The sixteenth bit is the starting bit of this field. A value of 0 indicates 256 bytes. When this queue is enabled (TXQEN=10), a minimum value of 7 (size = 2048) is recommended so that the queue can store more than a complete max-sized Ethernet packet.

(table continues...)

(continued)

Field	Bits	Type	Description
0	7, 15:11, 31:23	r	**Reserved** Read as 0; should be written with 0.

14.6.598 Port j MTL TxQ4 Underflow

The Queue 4 Underflow Counter register contains the number of packets aborted because of Transmit queue underflow.

Portj_MTL_TxQ4_Underflow (j=0-1) Offset address: $11304_H+j*2000_H$

Port j MTL TxQ4 Underflow Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UFCNTOVF	0				UFPKTCNT										
r	r				r										

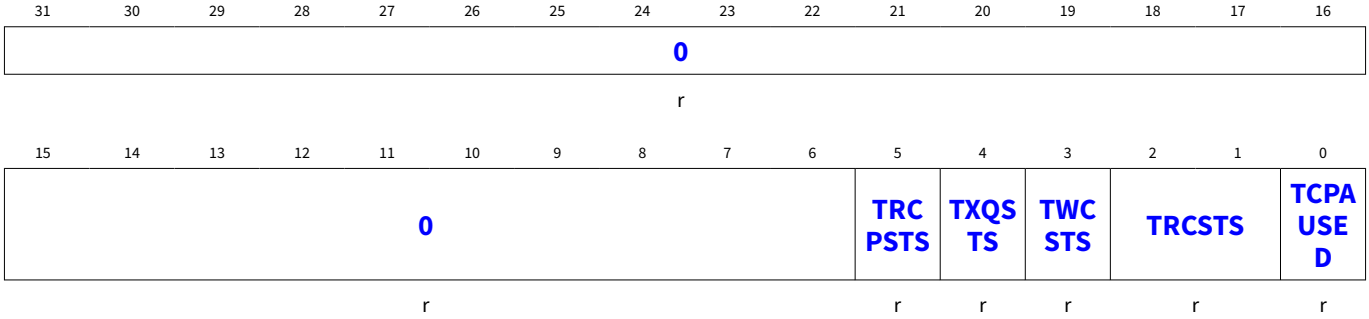
Field	Bits	Type	Description
UFPKTCNT	10:0	r	**Underflow Packet Counter** This field indicates the number of packets aborted by the controller because of Transmit Queue Underflow. This counter is incremented each time the MAC aborts an outgoing packet because of underflow. The counter is reset when this register is read with LSB of the Byte Enable for CSR access signal at 1'b1 .
UFCNTOVF	15	r	**Overflow Bit for Underflow Packet Counter** This bit is set every time the Tx Queue Underflow Packet Counter field overflows, that is, it has crossed the maximum count. In such a scenario, the overflow packet counter is reset to all-zeros and this bit indicates that the rollover happened.
0	14:11, 31:16	r	**Reserved** Read as 0; should be written with 0.

14.6.599 Port j MTL TxQ4 Debug

The Queue 4 Transmit Debug register gives the debug status of various blocks related to the Transmit Queue 4 and Traffic Class 4.

Portj_MTL_TxQ4_Debug (j=0-1) Offset address: $11308_H+j*2000_H$

Port j MTL TxQ4 Debug Kernel Reset value: $0000\ 0000_H$



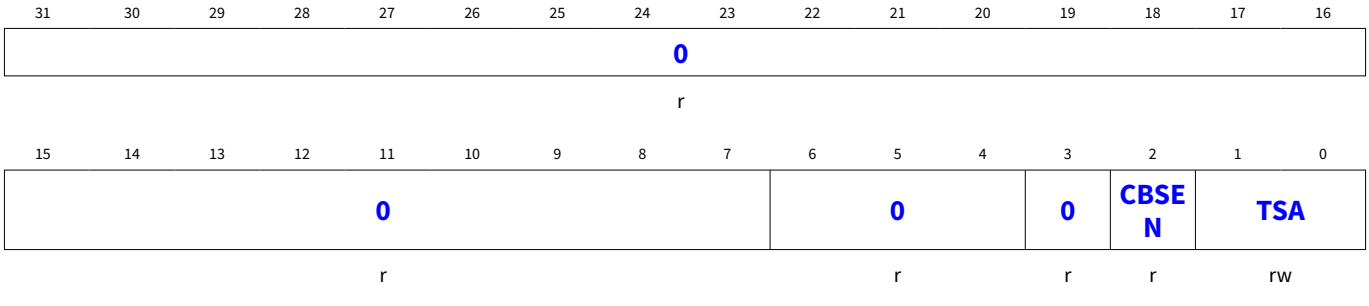
Field	Bits	Type	Description
TCPAUSED	0	r	**Traffic Class in Pause** - Reception of IEEE 802.3-2018 Pause packet
TRCSTS	2:1	r	**MTL Tx Queue Read Controller Status** This field indicates the state of the Tx Queue Read Controller: - 2'b00: Idle state - 2'b01: Read state (transferring data to the MAC transmitter) - 2'b10: Waiting for pending Tx Status from the MAC transmitter - 2'b11: Flushing the Tx queue because of the Packet Abort request from the MAC
TWCSTS	3	r	**MTL Tx Queue Write Controller Status** When high, this bit indicates that the MTL Tx Queue Write Controller is active, and it is transferring the data to the Tx Queue.
TXQSTS	4	r	**MTL Tx Queue Not Empty Status** When this bit is high, it indicates that the MTL Tx Queue is not empty and some data is left for transmission.
TRCPSTS	5	r	**MTL TRC Pending Status** When this bit is high, it indicates that the MTL Tx Read Controller (TRC) has a pending status from MAC.
0	31:6	r	**Reserved** Read as 0; should be written with 0.

14.6.600 Port j MTL TC4 ETS Control

This register configures the transmission selection algorithm, for Traffic Class 4.

Portj_MTL_TC4_ETS_Control (j=0-1) Offset address: $11310_H+j*2000_H$

Port j MTL TC4 ETS Control Kernel Reset value: $0000\ 0000_H$



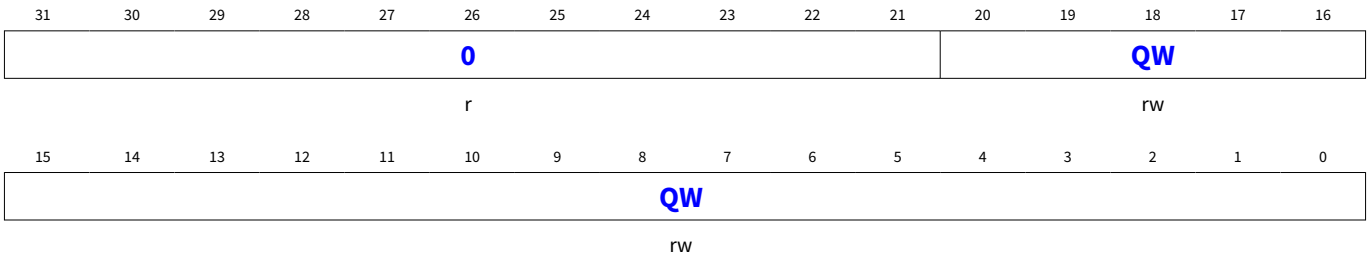
Field	Bits	Type	Description
TSA	1:0	rw	**Transmission Selection Algorithm** This field is used to assign a transmission selection algorithm for this traffic class. - 2'b00: Strict priority - 2'b01: CBS - 2'b10: ETS - 2'b11: Reserved
CBSEN	2	r	**Credit Based Shaper Enable** This field indicates that TSA field of this register is enabled for CBS
0	3, 6:4, 31:7	r	**Reserved** Read as 0; should be written with 0.

14.6.601 Port j MTL TC4 Quantum Weight

The Quantum or Weights register controls the average traffic transmitted in Traffic Class 4.

Portj_MTL_TC4_Quantum_Weight (j=0-1) Offset address: $11318_H+j*2000_H$

Port j MTL TC4 Quantum Weight Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
QW	20:0	rw	**idleSlopeCredit, Quantum or Weights** - idleSlopeCredit -- When this TC is in AVB mode, this field contains the idleSlopeCredit value required for the credit-based shaper algorithm for Traffic Class 4. This is the rate of change of credit in bits per cycle (3.2 ns for 2.5/10 Gbps; 8 ns for 1000 Mbps) when the credit is increasing. The software must program this field with computed credit in bits per cycle scaled by 1024. The maximum value is portTransmitRate, that is, 0x2000 in 1000/2500 Mbps mode and 0x8000 in 10 Gbps mode. Bits[20:16] must be written to zero. This field is not valid for Traffic Class 0. - Weights -- When TSA is set to ETS and ETSALG is set to WRR algorithm for Traffic Class 1 traffic, this field contains the weight for this queue. The maximum value is 0x64. Bits[20:7] must be written to zero. This field is reserved and RO when the number of traffic classes selected is less than two. This field is reserved in corresponding Queues 8-15 register.
0	31:21	r	**Reserved** Read as 0; should be written with 0.

14.6.602 Port j MTL RxQ4 Operation Mode

The Queue 4 Receive Operation Mode register establishes the Receive queue 4 operating modes.

Portj_MTL_RxQ4_Operation_Mode (j=0-1) Offset address: $11340_H+j*2000_H$

Port j MTL RxQ4 Operation Mode Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0									RQS						
r									rw						

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EHFC	DIS_TCP_EF	RSF	FEF	FUF	0	RTC	
r								rw	rw	rw	rw	rw	r	rw	

Field	Bits	Type	Description
RTC	1:0	rw	**Receive Queue Threshold Control** These bits control the threshold level of the MTL Rx queue (in bytes): - 2'b00: 64 - 2'b01: Reserved - 2'b10: 96 - 2'b11: 128 The received packet is transferred to the application or DMA when the packet size within the MTL Rx queue is larger than the threshold. In addition, full packets with length less than the threshold are automatically transferred. The value of 11 is not applicable if the size of the configured Rx Queue is 128 bytes. This field is valid only when the RSF bit is zero. This field is ignored when the RSF bit is set to 1.
FUF	3	rw	**Forward Undersized Good Packets** - 1: Rx queue forwards the undersized good packets (with no Invalid-code/GMII/CRC/Length errors and length less than 64 bytes at Rx (X)GMII interface), including pad-bytes and CRC. - 0: Rx queue drops all packets of length less than 64 bytes.
FEF	4	rw	**Forward Error Packets** - 0: Rx queue drops packets with error status (CRC error, GMII_ER, watchdog timeout, or overflow). However, if the start byte (write) pointer of a packet is already transferred to the read controller side (in Threshold mode), the packet is not dropped. - 1: All packets except the runt error packets are forwarded to the application or DMA. If the RSF bit is set and the Rx queue overflows when a partial packet is written, the packet is dropped irrespective of the setting of this bit. However, if the RSF bit is reset and the Rx queue overflows when a partial packet is written, a partial packet might be forwarded to the application or DMA.

(table continues...)

(continued)

Field	Bits	Type	Description
RSF	5	rw	**Receive Queue Store and Forward** - 1: DWC_xgmac reads a packet from the Rx queue only after the complete packet has been written to it, ignoring the RTC field of this register. - 0: Rx queue operates in the Threshold (cut-through) mode, subject to the threshold specified by the RTC field of this register.
DIS_TCP_EF	6	rw	**Disable Dropping of TCP/IP Checksum Error Packets** - When this bit is set, the MAC does not drop the packets which only have the errors detected by the Receive Checksum Offload engine. Such packets have errors only in the encapsulated payload. There are no errors (including FCS error) in the Ethernet packet received by the MAC. - When this bit is reset, all error packets are dropped if the FEF bit is reset.
EHFC	7	rw	**Enable Hardware Flow Control** When this bit is set, the flow control signal operation, based on the fill-level of Rx queue, is enabled. When reset, the flow control operation is disabled.
RQS	22:16	rw	**Receive Queue Size** This field indicates the size of the allocated Receive queues in blocks of 256 bytes. The sixteenth bit is the starting bit of this field. A value of 0 indicates 256 bytes.When this queue is used, a minimum value of 7 (size = 2048) is recommended so that the queue can store more than a complete max-sized Ethernet packet.
0	2, 15:8, 31:23	r	**Reserved** Read as 0; should be written with 0.

14.6.603 Port j MTL RxQ4 Missed Pkt Overflow Cnt

The Queue 4 Overflow Counter register contains the counter for packets missed because of Receive queue packet flush and packets discarded because of Receive queue overflow.

Portj_MTL_RxQ4_Missed_Pkt_Overflow_Cnt (j=0-1) Offset address: $11344_H+j*2000_H$

Port j MTL RxQ4 Missed Pkt Overflow Cnt Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MISCNTOVF		0			MISPKTCNT										
r		r			r										

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OVFCNTOVF		0			OVFPKTCNT										
r		r			r										

Field	Bits	Type	Description
OVFPKTCNT	10:0	r	**Overflow Packet Counter** This field indicates the number of packets discarded by DWC_xgmac because of Receive queue overflow. This counter is incremented each time DWC_xgmac discards an incoming packet because of overflow. This counter is reset when this register bits[7:0] is read by software.
OVFCNTOVF	15	r	**Overflow Counter Overflow Bit** When set, this bit indicates that OVFPKTCNT field crossed the maximum limit.
MISPKTCNT	26:16	r	**Missed Packet Counter** This field indicates the number of packets missed by DWC_xgmac because the host asserted packet-flush for this queue. This counter is incremented each time a packet is flushed for this queue. This counter is reset when this register's bits[23:16] is read by the software.
MISCNTOVF	31	r	**Missed Packet Counter Overflow Bit** When set, this bit indicates that the MISPKTCNT field crossed the maximum limit.
0	14:11, 30:27	r	**Reserved** Read as 0; should be written with 0.

14.6.604 Port j MTL RxQ4 Debug

The Queue 4 Receive Debug register gives the debug status of various blocks related to the Receive queue.

Portj_MTL_RxQ4_Debug (j=0-1) Offset address: $11348_H+j*2000_H$

Port j MTL RxQ4 Debug Kernel Reset value: $0000\ 0000_H$

Field	Bits	Type	Description
RWCSTS	0	r	**MTL Rx Queue Write Controller Active Status** When high, this bit indicates that the MTL Rx queue Write controller is active, and it is transferring a received packet to the Rx Queue.
RRCSTS	2:1	r	**MTL Rx Queue Read Controller State** This field gives the state of the Rx queue Read controller: - 2'b00: Idle state - 2'b01: Reading packet data - 2'b10: Reading packet status (or timestamp) - 2'b11: Flushing the packet data and status

(table continues...)

(continued)

Field	Bits	Type	Description
RXQSTS	5:4	r	**MTL Rx Queue Fill-Level Status** This field gives the status of the fill-level of the Rx Queue: - 2'b00: Rx Queue empty - 2'b01: Rx Queue fill-level below flow-control deactivate threshold - 2'b10: Rx Queue fill-level above flow-control activate threshold - 2'b11: Rx Queue full
PRXQ	29:16	r	**Number of Packets in Receive Queue** This field indicates the current number of packets in the Rx Queue. The theoretical maximum value for this field is 256KB/16B = 16K Packets, that is, Max_Queue_Size/Min_Packet_Size.
0	3, 15:6, 31:30	r	**Reserved** Read as 0; should be written with 0.

14.6.605 Port j MTL RxQ4 Control

The Queue 4 Receive Control register controls the receive arbitration and passing of received packets to the application.

Portj_MTL_RxQ4_Control (j=0-1) Offset address: $1134C_H+j*2000_H$

Port j MTL RxQ4 Control Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												RXQ_PKT_ARBIT	RXQ_WEGT		
r												rw	rw		

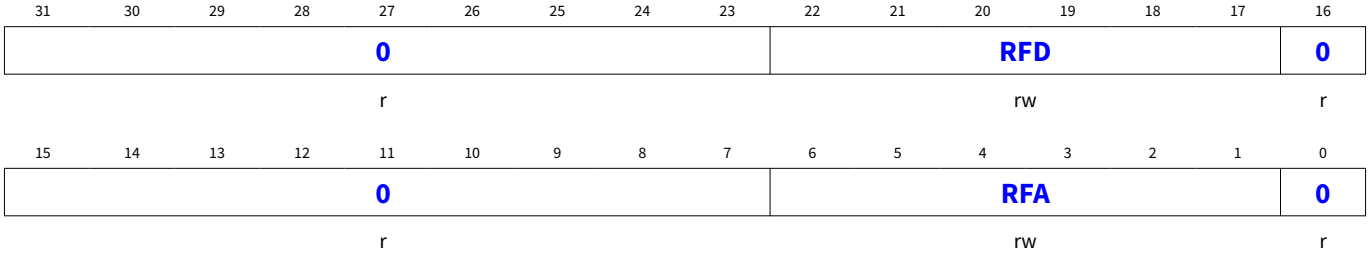
Field	Bits	Type	Description
RXQ_WEGT	2:0	rw	**Receive Queue Weight** This field indicates the weight assigned to the Rx Queue 4. The weight is used as the number of continuous PBL requests or contiguous packets (depending on the RXQ_PKT_ARBIT) allocated to the queue in one arbitration cycle.
RXQ_PKT_ARBIT	3	rw	**Receive Queue Packet Arbitration Control** - 1: MTL enables the packet-level arbitration. - 0: MTL enables the PBL based arbitration.
0	31:4	r	**Reserved** Read as 0; should be written with 0.

14.6.606 Port j MTL RxQ4 Flow Control

This register controls the activation and de-activation threshold in the queue for triggering the generation of Flow Control packets by MAC transmitter.

Portj_MTL_RxQ4_Flow_Control (j=0-1) Offset address: $11350_H+j*2000_H$

Port j MTL RxQ4 Flow Control Kernel Reset value: $0000\ 0000_H$



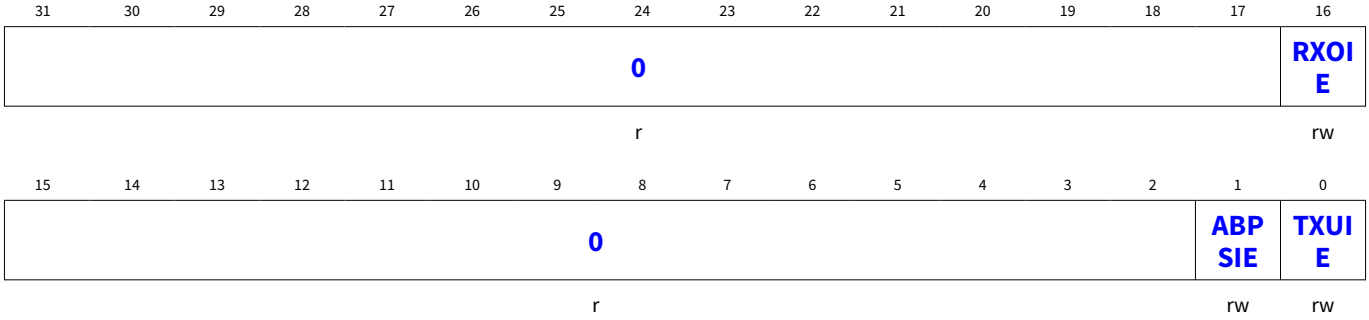
Field	Bits	Type	Description
RFA	6:1	rw	**Threshold for activating Flow Control** These bits control the threshold (fill-level of RxQ0) at which the flow control is activated - 6'b000000: RQS value minus 1 KB, that is, FULL - 1024 bytes - 6'b000001: RQS value minus 1.5 KB, that is, FULL - 1536 bytes - 6'b000010: RQS value minus 2 KB, that is, FULL - 2048 bytes. - ... - 6'b111111: RQS value minus 32.5 KB, that is, FULL - 33280 bytes The flow control is triggered when EHFC=1 and the queue size (RQS value in MTL_RxQ4_Operation Mode register) is equal to or more than 4KB. Note that the value to be programmed in RFA must be such that the threshold is positive number.
RFD	22:17	rw	**Threshold for deactivating Flow Control** These bits control the threshold (fill-level of RxQ0) at which the flow control is de-asserted once it is activated. - 6'b000000: RQS value minus 1 KB, that is, FULL - 1024 bytes - 6'b000001: RQS value minus 1.5 KB, that is, FULL - 1536 bytes - 6'b000010: RQS value minus 2 KB, that is, FULL - 2048 bytes. - ... - 6'b111111: RQS value minus 32.5 KB, that is, FULL - 33280 bytes The de-assertion is effective only after flow control is asserted as per the setting of RFA field. Note that the value to be programmed must be such that the threshold is positive number.
0	0, 16:7, 31:23	r	**Reserved** Read as 0; should be written with 0.

14.6.607 Port j MTL Q4 Interrupt Enable

This register contains the interrupt enable bits for Traffic class/Queue 4.

Portj_MTL_Q4_Interrupt_Enable (j=0-1) Offset address: $11370_H+j*2000_H$

Port j MTL Q4 Interrupt Enable Kernel Reset value: $0000\ 0000_H$



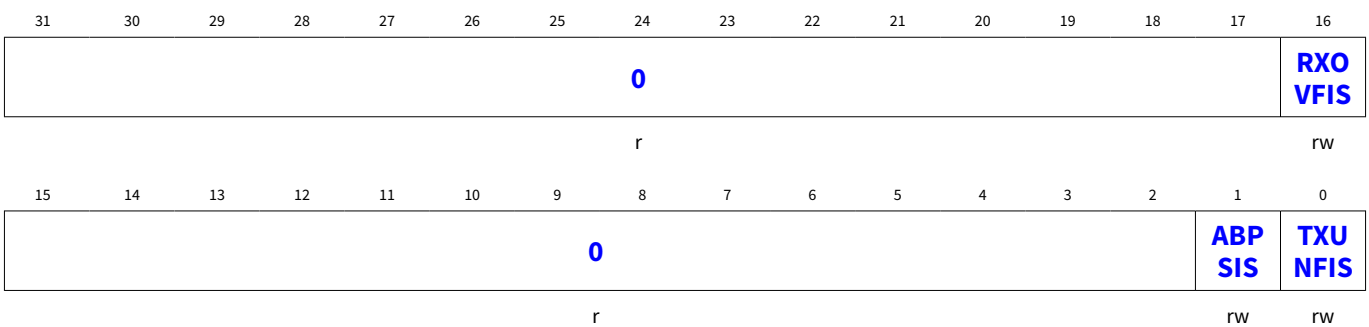
Field	Bits	Type	Description
TXUIE	0	rw	**Transmit Queue Underflow Interrupt Enable** - 1: Transmit Queue Underflow interrupt is enabled. - 0: Transmit Queue Underflow interrupt is disabled.
ABPSIE	1	rw	**Average Bits Per Slot Interrupt Enable** When this bit is set, average bits per slot interrupt is enabled for Traffic Class 4. An interrupt is generated when ABPSIS is set for this traffic class. This bit is reserved and RO when the number of traffic classes selected is less than 4.
RXOIE	16	rw	**Receive Queue Overflow Interrupt Enable** - 1: Receive Queue Overflow interrupt is enabled. - 0: Receive Queue Overflow interrupt is disabled.
0	15:2, 31:17	r	**Reserved** Read as 0; should be written with 0.

14.6.608 Port j MTL Q4 Interrupt Status

This register contains the interrupt status bits for Traffic class/Queue 4.

Portj_MTL_Q4_Interrupt_Status (j=0-1) Offset address: $11374_H+j*2000_H$

Port j MTL Q4 Interrupt Status Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
TXUNFIS	0	rw	**Transmit Queue Underflow Interrupt Status** This bit indicates that the Transmit Queue had an Underflow during packet transmission. Transmission is suspended and an Underflow Error TDES3[2] is set. This bit is cleared when the application writes a 1 to this bit.
ABPSIS	1	rw	**Average Bits Per Slot Interrupt Status** When set, this bit indicates that the MAC has updated the ABS value for this traffic class. This bit is cleared when the application writes a 1 to this bit. This bit is reserved and RO when the number of traffic classes selected is less than 4.
RXOVFIS	16	rw	**Receive Queue Overflow Interrupt Status** This bit indicates that the Receive Queue had an Overflow during packet reception. If partial packet is transferred to the application, the overflow status is set in the corresponding Rx status. This bit is cleared when the application writes a 1 to this bit.
0	15:2, 31:17	r	**Reserved** Read as 0; should be written with 0.

14.6.609 Port j MTL TxQ5 Operation Mode

The Queue 5 Transmit Operation Mode register establishes the Transmit queue operating modes and commands.

Portj_MTL_TxQ5_Operation_Mode (j=0-1) Offset address: $11380_H+j*2000_H$

Port j MTL TxQ5 Operation Mode Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0									TQS						
r									rw						

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0					Q2TCMAP			0	TTC			TXQEN		TSF	FTQ
r					rw			r	rw			rw		rw	rw

Field	Bits	Type	Description
FTQ	0	rw	**Flush Transmit Queue** When this bit is set, the Tx Queue Controller logic is reset to its default values. Therefore, all the data in the Tx queue is lost or flushed. This bit is internally reset when the flushing operation is complete. You must not write to this register until this bit is reset. The data which is already accepted by the MAC transmitter is not flushed. It is scheduled for transmission and results in underflow and runt packet transmission. Note: The flush operation is complete only when the Tx queue is empty. To complete this flush operation, the PHY Tx clock must be active.

(table continues...)

(continued)

Field	Bits	Type	Description
TSF	1	rw	**Transmit Store and Forward** When this bit is set, the transmission starts when a full packet resides in the MTL Tx Queue. When this bit is set, the values specified in the TTC field are ignored. This bit must be changed only when the transmission is stopped.
TXQEN	3:2	rw	**Transmit Queue Enable** This field is used to enable/disable the transmit queue. - 2'b00: Not enabled - 2'b01: Audio Video Bridging - 2'b10: Enabled - 2'b11: Reserved In multiple Tx queues configuration, as the queue is disabled by default, enable the Tx Queue by programming this field in the AVB mode or default mode as desired.
TTC	6:4	rw	**Transmit Threshold Control** These bits control the threshold level of the MTL Tx Queue. Transmission starts when the packet size within the MTL Tx Queue is larger than the threshold. In addition, full packets with length less than the threshold are also transmitted. These bits are used only when the TSF bit is reset. - 3'b000: 64 - 3'b001: Reserved - 3'b010: 96 - 3'b011: 128 - 3'b100: 192 - 3'b101: 256 - 3'b110: 384 - 3'b111: 512
Q2TCMAP	10:8	rw	**Queue to Traffic Class Mapping** This field is used to map this Tx Queue to a traffic class (TC). - 3'b000: TC0 - 3'b001: TC1 - 3'b010: TC2 - 3'b011: TC3 - 3'b100: TC4 - 3'b101: TC5 - 3'b110: TC6 - 3'b111: TC7
TQS	22:16	rw	**Transmit Queue Size** This field indicates the size of the allocated Transmit queues in blocks of 256 bytes. The sixteenth bit is the starting bit of this field. A value of 0 indicates 256 bytes. When this queue is enabled (TXQEN=10), a minimum value of 7 (size = 2048) is recommended so that the queue can store more than a complete max-sized Ethernet packet.

(table continues...)

(continued)

Field	Bits	Type	Description
0	7, 15:11, 31:23	r	**Reserved** Read as 0; should be written with 0.

14.6.610 Port j MTL TxQ5 Underflow

The Queue 5 Underflow Counter register contains the number of packets aborted because of Transmit queue underflow.

Portj_MTL_TxQ5_Underflow (j=0-1) Offset address: $11384_H+j*2000_H$

Port j MTL TxQ5 Underflow Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UFCNTOVF	0				UFPKTCNT										
r	r				r										

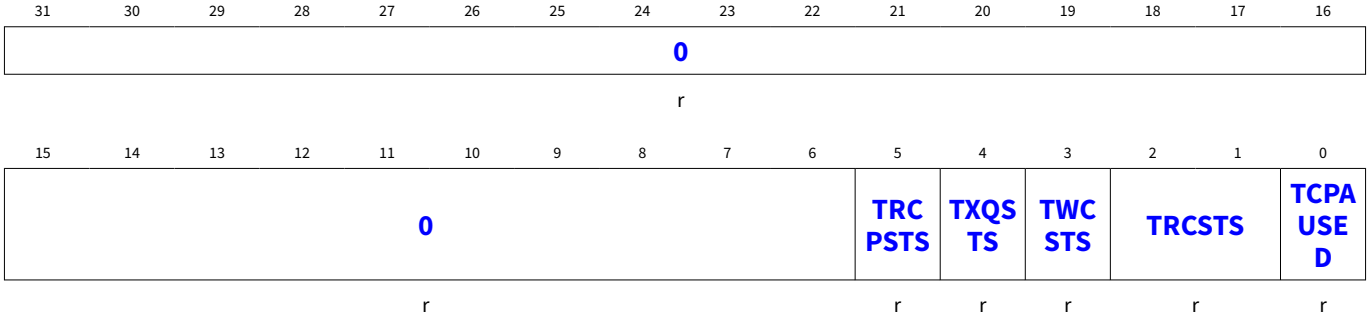
Field	Bits	Type	Description
UFPKTCNT	10:0	r	**Underflow Packet Counter** This field indicates the number of packets aborted by the controller because of Transmit Queue Underflow. This counter is incremented each time the MAC aborts an outgoing packet because of underflow. The counter is reset when this register is read with LSB of the Byte Enable for CSR access signal at 1'b1 .
UFCNTOVF	15	r	**Overflow Bit for Underflow Packet Counter** This bit is set every time the Tx Queue Underflow Packet Counter field overflows, that is, it has crossed the maximum count. In such a scenario, the overflow packet counter is reset to all-zeros and this bit indicates that the rollover happened.
0	14:11, 31:16	r	**Reserved** Read as 0; should be written with 0.

14.6.611 Port j MTL TxQ5 Debug

The Queue 5 Transmit Debug register gives the debug status of various blocks related to the Transmit Queue 5 and Traffic Class 5.

Portj_MTL_TxQ5_Debug (j=0-1) Offset address: $11388_H+j*2000_H$

Port j MTL TxQ5 Debug Kernel Reset value: $0000\ 0000_H$



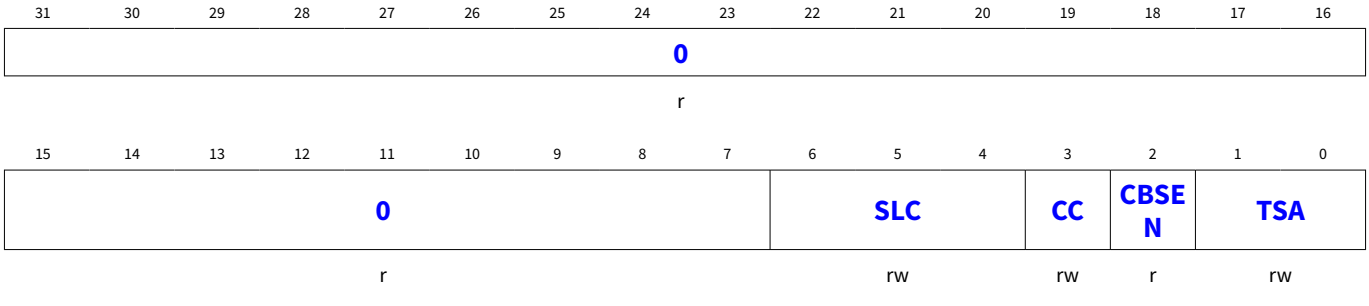
Field	Bits	Type	Description
TCPAUSED	0	r	**Traffic Class in Pause** - Reception of IEEE 802.3-2018 Pause packet
TRCSTS	2:1	r	**MTL Tx Queue Read Controller Status** This field indicates the state of the Tx Queue Read Controller: - 2'b00: Idle state - 2'b01: Read state (transferring data to the MAC transmitter) - 2'b10: Waiting for pending Tx Status from the MAC transmitter - 2'b11: Flushing the Tx queue because of the Packet Abort request from the MAC
TWCSTS	3	r	**MTL Tx Queue Write Controller Status** When high, this bit indicates that the MTL Tx Queue Write Controller is active, and it is transferring the data to the Tx Queue.
TXQSTS	4	r	**MTL Tx Queue Not Empty Status** When this bit is high, it indicates that the MTL Tx Queue is not empty and some data is left for transmission.
TRCPSTS	5	r	**MTL TRC Pending Status** When this bit is high, it indicates that the MTL Tx Read Controller (TRC) has a pending status from MAC.
0	31:6	r	**Reserved** Read as 0; should be written with 0.

14.6.612 Port j MTL TC5 ETS Control

This register configures the transmission selection algorithm, for Traffic Class 5.

Portj_MTL_TC5_ETS_Control (j=0-1) Offset address: $11390_H+j*2000_H$

Port j MTL TC5 ETS Control Kernel Reset value: $0000\ 0000_H$



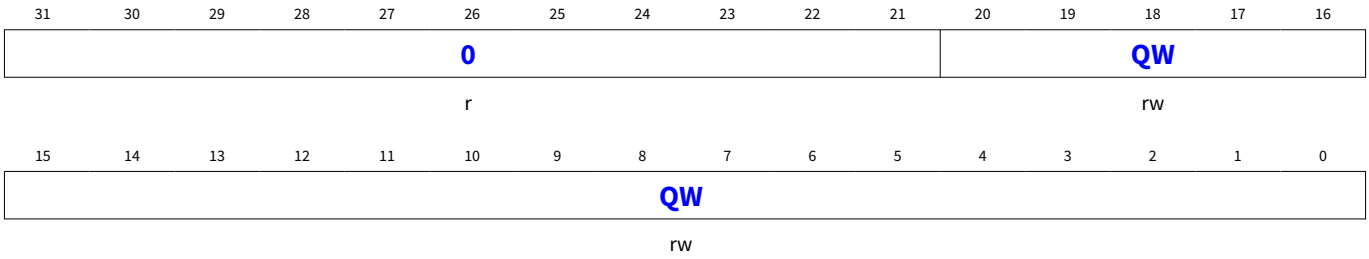
Field	Bits	Type	Description
TSA	1:0	rw	**Transmission Selection Algorithm** This field is used to assign a transmission selection algorithm for this traffic class. - 2'b00: Strict priority - 2'b01: CBS - 2'b10: ETS - 2'b11: Reserved
CBSEN	2	r	**Credit Based Shaper Enable** This field indicates that TSA field of this register is enabled for CBS
CC	3	rw	**Credit Control** When this bit is set, the accumulated credit parameter in the credit-based shaper algorithm logic is not reset to zero when there is positive credit and no packet to transmit in TC(#n). The credit accumulates even when there is no packet waiting in TC(#n) and another TC is transmitting. When this bit is reset, the accumulated credit parameter in the credit-based shaper algorithm logic is set to zero when there is positive credit and no packet to transmit in TC(#n). When there is no packet waiting in TC(#n) and other TC is transmitting, no credit is accumulated.
SLC	6:4	rw	**Slot Count** If the credit-based shaper algorithm is enabled, the software can program the number of slots (of 125 us duration) over which the average transmitted bits per slot, provided in the MTL_TC(#n)_ETS_Status register, need to be computed for TC(#n). The encoding is as follows: - 3'b000: 1 Slot - 3'b001: 2 Slots - 3'b010: 4 Slots - 3'b011: 8 Slots - 3'b100: 16 Slots - 3'b101-3'b111: Reserved
0	31:7	r	**Reserved** Read as 0; should be written with 0.

14.6.613 Port j MTL TC5 Quantum Weight

The Quantum or Weights register controls the average traffic transmitted in Traffic Class 5.

Portj_MTL_TC5_Quantum_Weight (j=0-1) Offset address: $11398_H+j*2000_H$

Port j MTL TC5 Quantum Weight Kernel Reset value: $0000\ 0000_H$



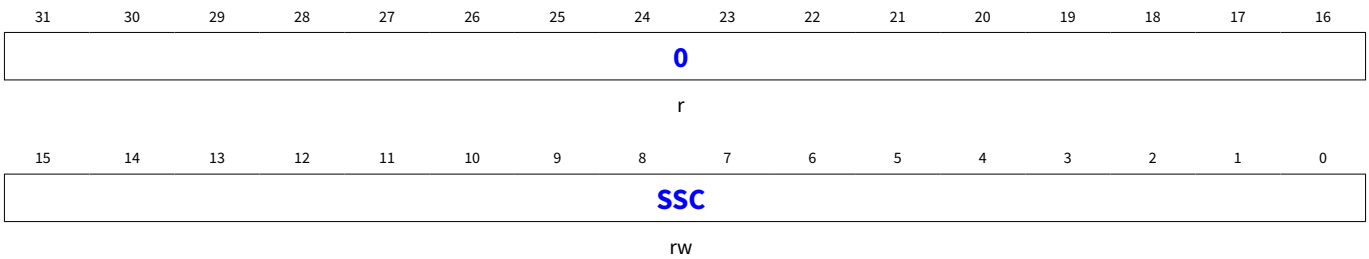
Field	Bits	Type	Description
QW	20:0	rw	**idleSlopeCredit, Quantum or Weights** - idleSlopeCredit -- When this TC is in AVB mode, this field contains the idleSlopeCredit value required for the credit-based shaper algorithm for Traffic Class 5. This is the rate of change of credit in bits per cycle (3.2 ns for 2.5/10 Gbps; 8 ns for 1000 Mbps) when the credit is increasing. The software must program this field with computed credit in bits per cycle scaled by 1024. The maximum value is portTransmitRate, that is, 0x2000 in 1000/2500 Mbps mode and 0x8000 in 10 Gbps mode. Bits[20:16] must be written to zero. This field is not valid for Traffic Class 0. - Weights -- When TSA is set to ETS and ETSALG is set to WRR algorithm for Traffic Class 1 traffic, this field contains the weight for this queue. The maximum value is 0x64. Bits[20:7] must be written to zero. This field is reserved and RO when the number of traffic classes selected is less than two. This field is reserved in corresponding Queues 8-15 register.
0	31:21	r	**Reserved** Read as 0; should be written with 0.

14.6.614 Port j MTL TC5 SendSlopeCredit

MTL registers for traffic classes and queues 1 registers.

Portj_MTL_TC5_SendSlopeCredit (j=0-1) Offset address: $1139C_H+j*2000_H$

Port j MTL TC5 SendSlopeCredit Kernel Reset value: $0000\ 0000_H$



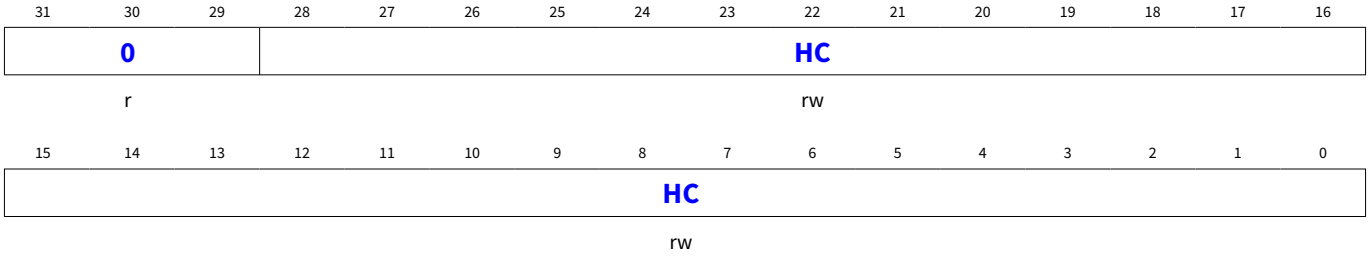
Field	Bits	Type	Description
SSC	15:0	rw	**sendSlopeCredit Value** - When Audio Video Bridging feature is enabled, this field contains the sendSlopeCredit value required for credit-based shaper algorithm for TC(#n). The software must program this field with computed credit in bits per cycle scaled by 1024. This field must be programmed with absolute sendSlopeCredit value. The credit-based shaper logic subtracts it from the accumulated credit when TC(#n) is selected for transmission. See the application note for more details on programming this value.
0	31:16	r	**Reserved** Read as 0; should be written with 0.

14.6.615 Port j MTL TC5 HiCredit

This register contains the hiCredit value required for the credit-based shaper algorithm for TC1.

Portj_MTL_TC5_HiCredit (j=0-1) Offset address: $113A0_H+j*2000_H$

Port j MTL TC5 HiCredit Kernel Reset value: $0000\ 0000_H$



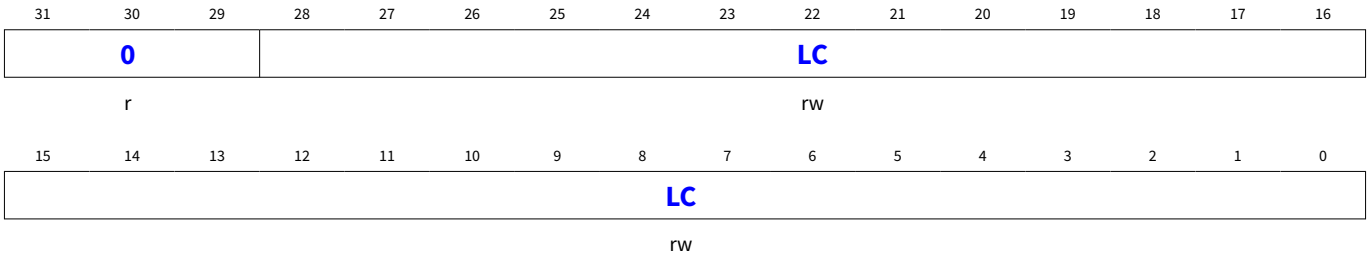
Field	Bits	Type	Description
HC	28:0	rw	**hiCredit Value** - When the Audio Video Bridging feature is enabled, this field contains the hiCredit value required for the credit-based shaper algorithm. This is the maximum value that can be accumulated in the credit parameter. This is specified in bits scaled by 1024. The maximum value is maxInterferenceSize, that is, best-effort maximum packet size (16384 bytes or 131072 bits). The value to be specified is 131072 * 1024 = 134217728 or 0x0800_0000
0	31:29	r	**Reserved** Read as 0; should be written with 0.

14.6.616 Port j MTL TC5 LoCredit

This register contains the loCredit value required for the credit-based shaper algorithm for TC1.

Portj_MTL_TC5_LoCredit (j=0-1) Offset address: $113A4_H+j*2000_H$

Port j MTL TC5 LoCredit Kernel Reset value: $0000\ 0000_H$



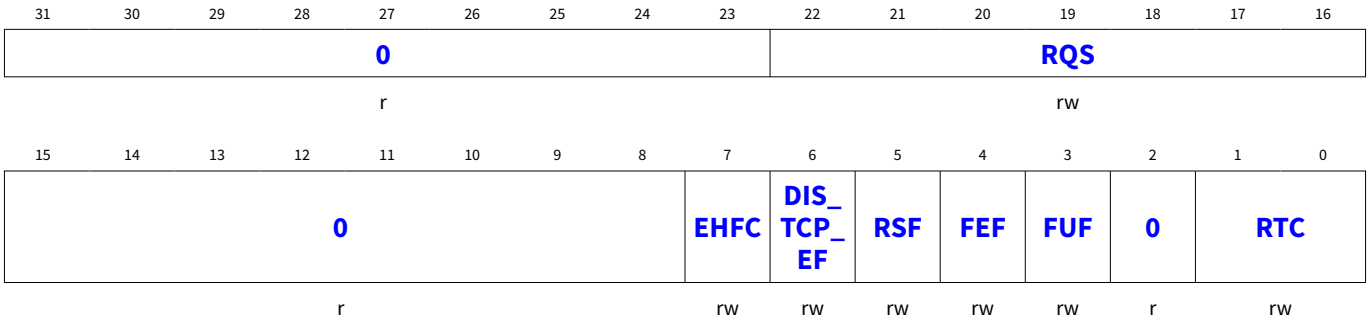
Field	Bits	Type	Description
LC	28:0	rw	**loCredit Value** - When Audio Video Bridging feature is enabled, this field contains the loCredit value required for the credit-based shaper algorithm. This is the minimum value that can be accumulated in the credit parameter. This is specified in bits scaled by 1024. The maximum value is maxPacketSize transmitted from this queue, that is, 16384 * 8 * 1024 = 134217728 or 0x0800_0000. Because it is a negative value, the programmed value is twos complement of the value, that is, 0x1800_0000.
0	31:29	r	**Reserved** Read as 0; should be written with 0.

14.6.617 Port j MTL RxQ5 Operation Mode

The Queue 5 Receive Operation Mode register establishes the Receive queue 5 operating modes.

Portj_MTL_RxQ5_Operation_Mode (j=0-1) Offset address: $113C0_H+j*2000_H$

Port j MTL RxQ5 Operation Mode Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
RTC	1:0	rw	**Receive Queue Threshold Control** These bits control the threshold level of the MTL Rx queue (in bytes): - 2'b00: 64 - 2'b01: Reserved - 2'b10: 96 - 2'b11: 128 The received packet is transferred to the application or DMA when the packet size within the MTL Rx queue is larger than the threshold. In addition, full packets with length less than the threshold are automatically transferred. The value of 11 is not applicable if the size of the configured Rx Queue is 128 bytes. This field is valid only when the RSF bit is zero. This field is ignored when the RSF bit is set to 1.
FUF	3	rw	**Forward Undersized Good Packets** - 1: Rx queue forwards the undersized good packets (with no Invalid-code/GMII/CRC/Length errors and length less than 64 bytes at Rx (X)GMII interface), including pad-bytes and CRC. - 0: Rx queue drops all packets of length less than 64 bytes.
FEF	4	rw	**Forward Error Packets** - 0: Rx queue drops packets with error status (CRC error, GMII_ER, watchdog timeout, or overflow). However, if the start byte (write) pointer of a packet is already transferred to the read controller side (in Threshold mode), the packet is not dropped. - 1: All packets except the runt error packets are forwarded to the application or DMA. If the RSF bit is set and the Rx queue overflows when a partial packet is written, the packet is dropped irrespective of the setting of this bit. However, if the RSF bit is reset and the Rx queue overflows when a partial packet is written, a partial packet might be forwarded to the application or DMA.
RSF	5	rw	**Receive Queue Store and Forward** - 1: DWC_xgmac reads a packet from the Rx queue only after the complete packet has been written to it, ignoring the RTC field of this register. - 0: Rx queue operates in the Threshold (cut-through) mode, subject to the threshold specified by the RTC field of this register.
DIS_TCP_EF	6	rw	**Disable Dropping of TCP/IP Checksum Error Packets** - When this bit is set, the MAC does not drop the packets which only have the errors detected by the Receive Checksum Offload engine. Such packets have errors only in the encapsulated payload. There are no errors (including FCS error) in the Ethernet packet received by the MAC. - When this bit is reset, all error packets are dropped if the FEF bit is reset.
EHFC	7	rw	**Enable Hardware Flow Control** When this bit is set, the flow control signal operation, based on the fill-level of Rx queue, is enabled. When reset, the flow control operation is disabled.

(table continues...)

(continued)

Field	Bits	Type	Description
RQS	22:16	rw	**Receive Queue Size** This field indicates the size of the allocated Receive queues in blocks of 256 bytes. The sixteenth bit is the starting bit of this field. A value of 0 indicates 256 bytes.When this queue is used, a minimum value of 7 (size = 2048) is recommended so that the queue can store more than a complete max-sized Ethernet packet.
0	2, 15:8, 31:23	r	**Reserved** Read as 0; should be written with 0.

14.6.618 Port j MTL RxQ5 Missed Pkt Overflow Cnt

The Queue 5 Overflow Counter register contains the counter for packets missed because of Receive queue packet flush and packets discarded because of Receive queue overflow.

Portj_MTL_RxQ5_Missed_Pkt_Overflow_Cnt (j=0-1) Offset address: $113C4_H+j*2000_H$

Port j MTL RxQ5 Missed Pkt Overflow Cnt Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MISCNTOVF	0				MISPKTCNT										
r	r				r										

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OVFCNTOVF	0				OVFPKTCNT										
r	r				r										

Field	Bits	Type	Description
OVFPKTCNT	10:0	r	**Overflow Packet Counter** This field indicates the number of packets discarded by DWC_xgmac because of Receive queue overflow. This counter is incremented each time DWC_xgmac discards an incoming packet because of overflow. This counter is reset when this register bits[7:0] is read by software.
OVFCNTOVF	15	r	**Overflow Counter Overflow Bit** When set, this bit indicates that OVFPKTCNT field crossed the maximum limit.
MISPKTCNT	26:16	r	**Missed Packet Counter** This field indicates the number of packets missed by DWC_xgmac because the host asserted packet-flush for this queue. This counter is incremented each time a packet is flushed for this queue. This counter is reset when this register's bits[23:16] is read by the software.

(table continues...)

(continued)

Field	Bits	Type	Description
MISCNTOVF	31	r	**Missed Packet Counter Overflow Bit** When set, this bit indicates that the MISPKTCNT field crossed the maximum limit.
0	14:11, 30:27	r	**Reserved** Read as 0; should be written with 0.

14.6.619 Port j MTL RxQ5 Debug

The Queue 5 Receive Debug register gives the debug status of various blocks related to the Receive queue.

Portj_MTL_RxQ5_Debug (j=0-1) Offset address: $113C8_H+j*2000_H$

Port j MTL RxQ5 Debug Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		PRXQ													
r		r													

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0										RXQSTS		0	RRCSTS		RWCSTS
r										r		r	r		r

Field	Bits	Type	Description
RWCSTS	0	r	**MTL Rx Queue Write Controller Active Status** When high, this bit indicates that the MTL Rx queue Write controller is active, and it is transferring a received packet to the Rx Queue.
RRCSTS	2:1	r	**MTL Rx Queue Read Controller State** This field gives the state of the Rx queue Read controller: - 2'b00: Idle state - 2'b01: Reading packet data - 2'b10: Reading packet status (or timestamp) - 2'b11: Flushing the packet data and status
RXQSTS	5:4	r	**MTL Rx Queue Fill-Level Status** This field gives the status of the fill-level of the Rx Queue: - 2'b00: Rx Queue empty - 2'b01: Rx Queue fill-level below flow-control deactivate threshold - 2'b10: Rx Queue fill-level above flow-control activate threshold - 2'b11: Rx Queue full
PRXQ	29:16	r	**Number of Packets in Receive Queue** This field indicates the current number of packets in the Rx Queue. The theoretical maximum value for this field is 256KB/16B = 16K Packets, that is, Max_Queue_Size/Min_Packet_Size.
0	3, 15:6, 31:30	r	**Reserved** Read as 0; should be written with 0.

14.6.620 Port j MTL RxQ5 Control

The Queue 5 Receive Control register controls the receive arbitration and passing of received packets to the application.

Portj_MTL_RxQ5_Control (j=0-1) Offset address: $113CC_H+j*2000_H$

Port j MTL RxQ5 Control Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												RXQ_PKT_ARBIT	RXQ_WEGT		
r												rw	rw		

Field	Bits	Type	Description
RXQ_WEGT	2:0	rw	**Receive Queue Weight** This field indicates the weight assigned to the Rx Queue 5. The weight is used as the number of continuous PBL requests or contiguous packets (depending on the RXQ_PKT_ARBIT) allocated to the queue in one arbitration cycle.
RXQ_PKT_ARBIT	3	rw	**Receive Queue Packet Arbitration Control** - 1: MTL enables the packet-level arbitration. - 0: MTL enables the PBL based arbitration.
0	31:4	r	**Reserved** Read as 0; should be written with 0.

14.6.621 Port j MTL RxQ5 Flow Control

This register controls the activation and de-activation threshold in the queue for triggering the generation of Flow Control packets by MAC transmitter.

Portj_MTL_RxQ5_Flow_Control (j=0-1) Offset address: $113D0_H+j*2000_H$

Port j MTL RxQ5 Flow Control Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0									RFD						0
r									rw						r

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0									RFA						0
r									rw						r

Field	Bits	Type	Description
RFA	6:1	rw	**Threshold for activating Flow Control** These bits control the threshold (fill-level of RxQ0) at which the flow control is activated - 6'b000000: RQS value minus 1 KB, that is, FULL - 1024 bytes - 6'b000001: RQS value minus 1.5 KB, that is, FULL - 1536 bytes - 6'b000010: RQS value minus 2 KB, that is, FULL - 2048 bytes. - ... - 6'b111111: RQS value minus 32.5 KB, that is, FULL - 33280 bytes The flow control is triggered when EHFC=1 and the queue size (RQS value in MTL_RxQ5_Operation Mode register) is equal to or more than 4KB. Note that the value to be programmed in RFA must be such that the threshold is positive number.
RFD	22:17	rw	**Threshold for deactivating Flow Control** These bits control the threshold (fill-level of RxQ0) at which the flow control is de-asserted once it is activated. - 6'b000000: RQS value minus 1 KB, that is, FULL - 1024 bytes - 6'b000001: RQS value minus 1.5 KB, that is, FULL - 1536 bytes - 6'b000010: RQS value minus 2 KB, that is, FULL - 2048 bytes. - ... - 6'b111111: RQS value minus 32.5 KB, that is, FULL - 33280 bytes The de-assertion is effective only after flow control is asserted as per the setting of RFA field. Note that the value to be programmed must be such that the threshold is positive number.
0	0, 16:7, 31:23	r	**Reserved** Read as 0; should be written with 0.

14.6.622 Port j MTL Q5 Interrupt Enable

This register contains the interrupt enable bits for Traffic class/Queue 5.

Portj_MTL_Q5_Interrupt_Enable (j=0-1) Offset address: $113F0_H+j*2000_H$

Port j MTL Q5 Interrupt Enable Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															RXOIE
r															rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0														ABPSIE	TXUIE
r														rw	rw

Field	Bits	Type	Description
TXUIE	0	rw	**Transmit Queue Underflow Interrupt Enable** - 1: Transmit Queue Underflow interrupt is enabled. - 0: Transmit Queue Underflow interrupt is disabled.
ABPSIE	1	rw	**Average Bits Per Slot Interrupt Enable** When this bit is set, average bits per slot interrupt is enabled for Traffic Class 5. An interrupt is generated when ABPSIS is set for this traffic class. This bit is reserved and RO when the number of traffic classes selected is less than 5.
RXOIE	16	rw	**Receive Queue Overflow Interrupt Enable** - 1: Receive Queue Overflow interrupt is enabled. - 0: Receive Queue Overflow interrupt is disabled.
0	15:2, 31:17	r	**Reserved** Read as 0; should be written with 0.

14.6.623 Port j MTL Q5 Interrupt Status

This register contains the interrupt status bits for Traffic class/Queue 5.

Portj_MTL_Q5_Interrupt_Status (j=0-1) Offset address: $113F4_H+j*2000_H$

Port j MTL Q5 Interrupt Status Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															RXOVFIS
r															rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0														ABPSIS	TXUNFIS
r														rw	rw

Field	Bits	Type	Description
TXUNFIS	0	rw	**Transmit Queue Underflow Interrupt Status** This bit indicates that the Transmit Queue had an Underflow during packet transmission. Transmission is suspended and an Underflow Error TDES3[2] is set. This bit is cleared when the application writes a 1 to this bit.
ABPSIS	1	rw	**Average Bits Per Slot Interrupt Status** When set, this bit indicates that the MAC has updated the ABS value for this traffic class. This bit is cleared when the application writes a 1 to this bit. This bit is reserved and RO when the number of traffic classes selected is less than 5.

(table continues...)

(continued)

Field	Bits	Type	Description
RXOVFIS	16	rw	**Receive Queue Overflow Interrupt Status** This bit indicates that the Receive Queue had an Overflow during packet reception. If partial packet is transferred to the application, the overflow status is set in the corresponding Rx status. This bit is cleared when the application writes a 1 to this bit.
0	15:2, 31:17	r	**Reserved** Read as 0; should be written with 0.

14.6.624 Port j MTL TxQ6 Operation Mode

The Queue 6 Transmit Operation Mode register establishes the Transmit queue operating modes and commands.

Portj_MTL_TxQ6_Operation_Mode (j=0-1) Offset address: $11400_H+j*2000_H$

Port j MTL TxQ6 Operation Mode Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0									TQS						
r									rw						

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0					Q2TCMAP			0	TTC			TXQEN		TSF	FTQ
r					rw			r	rw			rw		rw	rw

Field	Bits	Type	Description
FTQ	0	rw	**Flush Transmit Queue** When this bit is set, the Tx Queue Controller logic is reset to its default values. Therefore, all the data in the Tx queue is lost or flushed. This bit is internally reset when the flushing operation is complete. You must not write to this register until this bit is reset. The data which is already accepted by the MAC transmitter is not flushed. It is scheduled for transmission and results in underflow and runt packet transmission. Note: The flush operation is complete only when the Tx queue is empty. To complete this flush operation, the PHY Tx clock must be active.
TSF	1	rw	**Transmit Store and Forward** When this bit is set, the transmission starts when a full packet resides in the MTL Tx Queue. When this bit is set, the values specified in the TTC field are ignored. This bit must be changed only when the transmission is stopped.

(table continues...)

(continued)

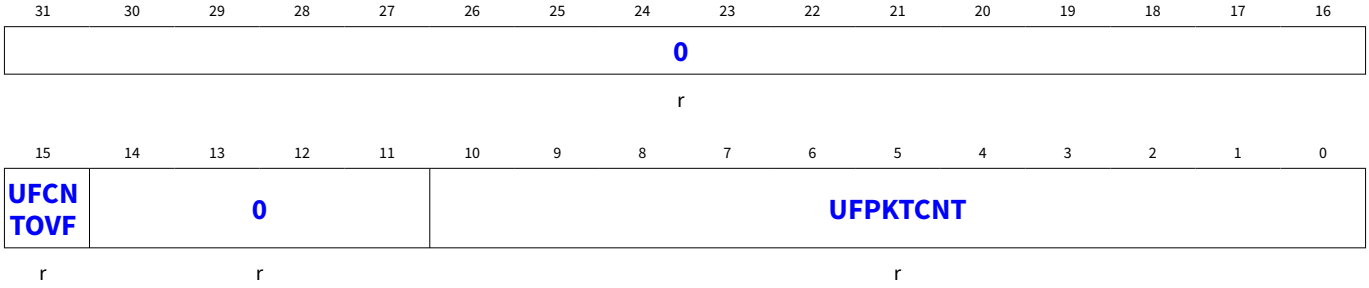
Field	Bits	Type	Description
TXQEN	3:2	rw	**Transmit Queue Enable** This field is used to enable/disable the transmit queue. - 2'b00: Not enabled - 2'b01: Audio Video Bridging - 2'b10: Enabled - 2'b11: Reserved In multiple Tx queues configuration, as the queue is disabled by default, enable the Tx Queue by programming this field in the AVB mode or default mode as desired.
TTC	6:4	rw	**Transmit Threshold Control** These bits control the threshold level of the MTL Tx Queue. Transmission starts when the packet size within the MTL Tx Queue is larger than the threshold. In addition, full packets with length less than the threshold are also transmitted. These bits are used only when the TSF bit is reset. - 3'b000: 64 - 3'b001: Reserved - 3'b010: 96 - 3'b011: 128 - 3'b100: 192 - 3'b101: 256 - 3'b110: 384 - 3'b111: 512
Q2TCMAP	10:8	rw	**Queue to Traffic Class Mapping** This field is used to map this Tx Queue to a traffic class (TC). - 3'b000: TC0 - 3'b001: TC1 - 3'b010: TC2 - 3'b011: TC3 - 3'b100: TC4 - 3'b101: TC5 - 3'b110: TC6 - 3'b111: TC7
TQS	22:16	rw	**Transmit Queue Size** This field indicates the size of the allocated Transmit queues in blocks of 256 bytes. The sixteenth bit is the starting bit of this field. A value of 0 indicates 256 bytes. When this queue is enabled (TXQEN=10), a minimum value of 7 (size = 2048) is recommended so that the queue can store more than a complete max-sized Ethernet packet.
0	7, 15:11, 31:23	r	**Reserved** Read as 0; should be written with 0.

14.6.625 Port j MTL TxQ6 Underflow

The Queue 6 Underflow Counter register contains the number of packets aborted because of Transmit queue underflow.

Portj_MTL_TxQ6_Underflow (j=0-1) Offset address: $11404_H+j*2000_H$

Port j MTL TxQ6 Underflow Kernel Reset value: $0000\ 0000_H$



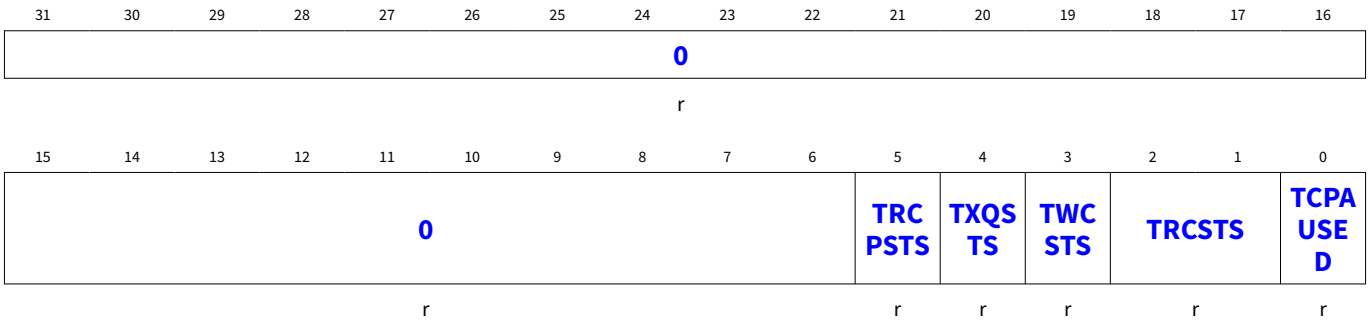
Field	Bits	Type	Description
UFPKTCNT	10:0	r	**Underflow Packet Counter** This field indicates the number of packets aborted by the controller because of Transmit Queue Underflow. This counter is incremented each time the MAC aborts an outgoing packet because of underflow. The counter is reset when this register is read with LSB of the Byte Enable for CSR access signal at 1'b1 .
UFCNTOVF	15	r	**Overflow Bit for Underflow Packet Counter** This bit is set every time the Tx Queue Underflow Packet Counter field overflows, that is, it has crossed the maximum count. In such a scenario, the overflow packet counter is reset to all-zeros and this bit indicates that the rollover happened.
0	14:11, 31:16	r	**Reserved** Read as 0; should be written with 0.

14.6.626 Port j MTL TxQ6 Debug

The Queue 6 Transmit Debug register gives the debug status of various blocks related to the Transmit Queue 6 and Traffic Class 6.

Portj_MTL_TxQ6_Debug (j=0-1) Offset address: $11408_H+j*2000_H$

Port j MTL TxQ6 Debug Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
TCPAUSED	0	r	**Traffic Class in Pause** - Reception of IEEE 802.3-2018 Pause packet
TRCSTS	2:1	r	**MTL Tx Queue Read Controller Status** This field indicates the state of the Tx Queue Read Controller: - 2'b00: Idle state - 2'b01: Read state (transferring data to the MAC transmitter) - 2'b10: Waiting for pending Tx Status from the MAC transmitter - 2'b11: Flushing the Tx queue because of the Packet Abort request from the MAC
TWCSTS	3	r	**MTL Tx Queue Write Controller Status** When high, this bit indicates that the MTL Tx Queue Write Controller is active, and it is transferring the data to the Tx Queue.
TXQSTS	4	r	**MTL Tx Queue Not Empty Status** When this bit is high, it indicates that the MTL Tx Queue is not empty and some data is left for transmission.
TRCPSTS	5	r	**MTL TRC Pending Status** When this bit is high, it indicates that the MTL Tx Read Controller (TRC) has a pending status from MAC.
0	31:6	r	**Reserved** Read as 0; should be written with 0.

14.6.627 Port j MTL TC6 ETS Control

This register configures the transmission selection algorithm, for Traffic Class 6.

Portj_MTL_TC6_ETS_Control (j=0-1) Offset address: $11410_H+j*2000_H$

Port j MTL TC6 ETS Control Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0									SLC			CC	CBSEN	TSA	
r									rw			rw	r	rw	

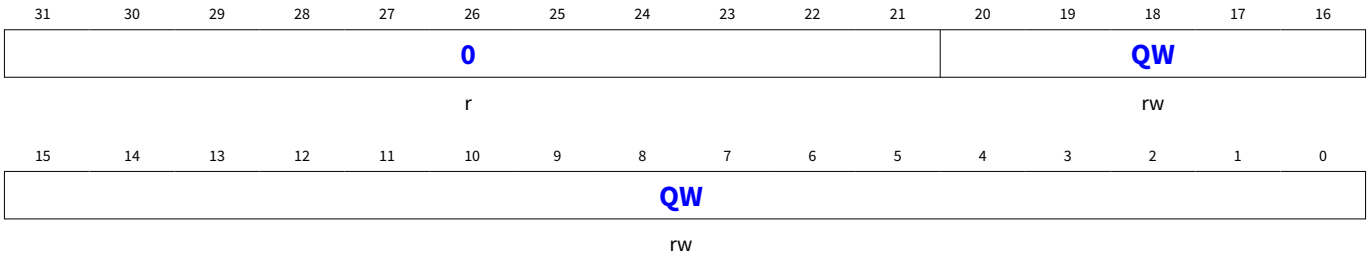
Field	Bits	Type	Description
TSA	1:0	rw	**Transmission Selection Algorithm** This field is used to assign a transmission selection algorithm for this traffic class. - 2'b00: Strict priority - 2'b01: CBS - 2'b10: ETS - 2'b11: Reserved
CBSEN	2	r	**Credit Based Shaper Enable** This field indicates that TSA field of this register is enabled for CBS
CC	3	rw	**Credit Control** When this bit is set, the accumulated credit parameter in the credit-based shaper algorithm logic is not reset to zero when there is positive credit and no packet to transmit in TC(#n). The credit accumulates even when there is no packet waiting in TC(#n) and another TC is transmitting. When this bit is reset, the accumulated credit parameter in the credit-based shaper algorithm logic is set to zero when there is positive credit and no packet to transmit in TC(#n). When there is no packet waiting in TC(#n) and other TC is transmitting, no credit is accumulated.
SLC	6:4	rw	**Slot Count** If the credit-based shaper algorithm is enabled, the software can program the number of slots (of 125 us duration) over which the average transmitted bits per slot, provided in the MTL_TC(#n)_ETS_Status register, need to be computed for TC(#n). The encoding is as follows: - 3'b000: 1 Slot - 3'b001: 2 Slots - 3'b010: 4 Slots - 3'b011: 8 Slots - 3'b100: 16 Slots - 3'b101-3'b111: Reserved
0	31:7	r	**Reserved** Read as 0; should be written with 0.

14.6.628 Port j MTL TC6 Quantum Weight

The Quantum or Weights register controls the average traffic transmitted in Traffic Class 6.

Portj_MTL_TC6_Quantum_Weight (j=0-1) Offset address: $11418_H+j*2000_H$

Port j MTL TC6 Quantum Weight Kernel Reset value: $0000\ 0000_H$



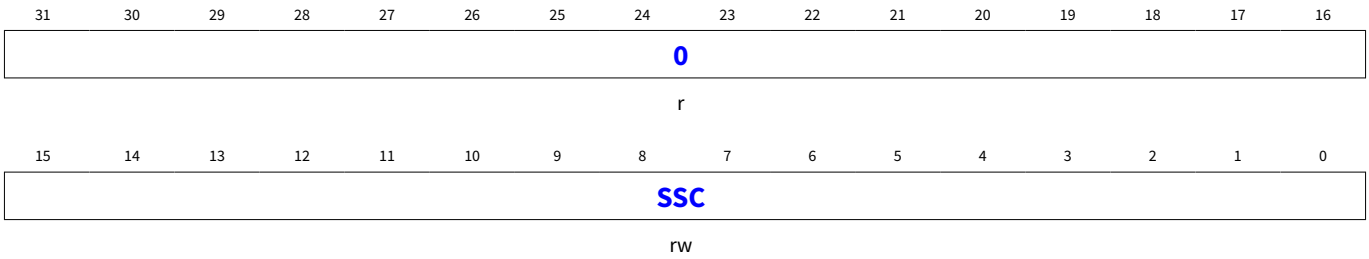
Field	Bits	Type	Description
QW	20:0	rw	**idleSlopeCredit, Quantum or Weights** - idleSlopeCredit -- When this TC is in AVB mode, this field contains the idleSlopeCredit value required for the credit-based shaper algorithm for Traffic Class 6. This is the rate of change of credit in bits per cycle (3.2 ns for 2.5/10 Gbps; 8 ns for 1000 Mbps) when the credit is increasing. The software must program this field with computed credit in bits per cycle scaled by 1024. The maximum value is portTransmitRate, that is, 0x2000 in 1000/2500 Mbps mode and 0x8000 in 10 Gbps mode. Bits[20:16] must be written to zero. This field is not valid for Traffic Class 0. - Weights -- When TSA is set to ETS and ETSALG is set to WRR algorithm for Traffic Class 1 traffic, this field contains the weight for this queue. The maximum value is 0x64. Bits[20:7] must be written to zero. This field is reserved and RO when the number of traffic classes selected is less than two. This field is reserved in corresponding Queues 8-15 register.
0	31:21	r	**Reserved** Read as 0; should be written with 0.

14.6.629 Port j MTL TC6 SendSlopeCredit

MTL registers for traffic classes and queues 1 registers.

Portj_MTL_TC6_SendSlopeCredit (j=0-1) Offset address: $1141C_H+j*2000_H$

Port j MTL TC6 SendSlopeCredit Kernel Reset value: $0000\ 0000_H$



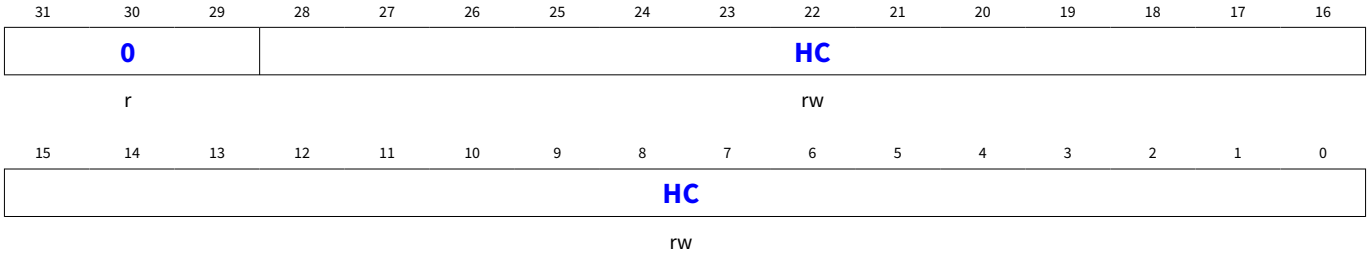
Field	Bits	Type	Description
SSC	15:0	rw	**sendSlopeCredit Value** - When Audio Video Bridging feature is enabled, this field contains the sendSlopeCredit value required for credit-based shaper algorithm for TC(#n). The software must program this field with computed credit in bits per cycle scaled by 1024. This field must be programmed with absolute sendSlopeCredit value. The credit-based shaper logic subtracts it from the accumulated credit when TC(#n) is selected for transmission. See the application note for more details on programming this value.
0	31:16	r	**Reserved** Read as 0; should be written with 0.

14.6.630 Port j MTL TC6 HiCredit

This register contains the hiCredit value required for the credit-based shaper algorithm for TC1.

Portj_MTL_TC6_HiCredit (j=0-1) Offset address: $11420_H+j*2000_H$

Port j MTL TC6 HiCredit Kernel Reset value: $0000\ 0000_H$



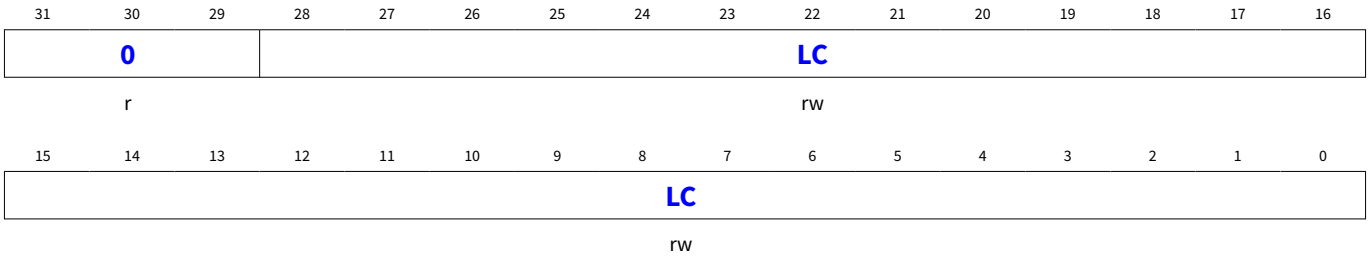
Field	Bits	Type	Description
HC	28:0	rw	**hiCredit Value** - When the Audio Video Bridging feature is enabled, this field contains the hiCredit value required for the credit-based shaper algorithm. This is the maximum value that can be accumulated in the credit parameter. This is specified in bits scaled by 1024. The maximum value is maxInterferenceSize, that is, best-effort maximum packet size (16384 bytes or 131072 bits). The value to be specified is 131072 * 1024 = 134217728 or 0x0800_0000
0	31:29	r	**Reserved** Read as 0; should be written with 0.

14.6.631 Port j MTL TC6 LoCredit

This register contains the loCredit value required for the credit-based shaper algorithm for TC1.

Portj_MTL_TC6_LoCredit (j=0-1) Offset address: $11424_H+j*2000_H$

Port j MTL TC6 LoCredit Kernel Reset value: $0000\ 0000_H$



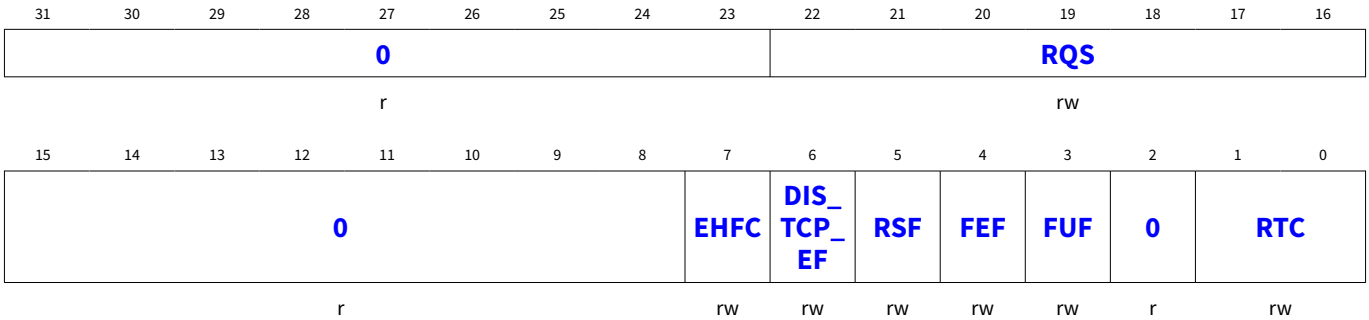
Field	Bits	Type	Description
LC	28:0	rw	**loCredit Value** - When Audio Video Bridging feature is enabled, this field contains the loCredit value required for the credit-based shaper algorithm. This is the minimum value that can be accumulated in the credit parameter. This is specified in bits scaled by 1024. The maximum value is maxPacketSize transmitted from this queue, that is, 16384 * 8 * 1024 = 134217728 or 0x0800_0000. Because it is a negative value, the programmed value is twos complement of the value, that is, 0x1800_0000.
0	31:29	r	**Reserved** Read as 0; should be written with 0.

14.6.632 Port j MTL RxQ6 Operation Mode

The Queue 6 Receive Operation Mode register establishes the Receive queue 6 operating modes.

Portj_MTL_RxQ6_Operation_Mode (j=0-1) Offset address: $11440_H+j*2000_H$

Port j MTL RxQ6 Operation Mode Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
RTC	1:0	rw	**Receive Queue Threshold Control** These bits control the threshold level of the MTL Rx queue (in bytes): - 2'b00: 64 - 2'b01: Reserved - 2'b10: 96 - 2'b11: 128 The received packet is transferred to the application or DMA when the packet size within the MTL Rx queue is larger than the threshold. In addition, full packets with length less than the threshold are automatically transferred. The value of 11 is not applicable if the size of the configured Rx Queue is 128 bytes. This field is valid only when the RSF bit is zero. This field is ignored when the RSF bit is set to 1.
FUF	3	rw	**Forward Undersized Good Packets** - 1: Rx queue forwards the undersized good packets (with no Invalid-code/GMII/CRC/Length errors and length less than 64 bytes at Rx (X)GMII interface), including pad-bytes and CRC. - 0: Rx queue drops all packets of length less than 64 bytes.
FEF	4	rw	**Forward Error Packets** - 0: Rx queue drops packets with error status (CRC error, GMII_ER, watchdog timeout, or overflow). However, if the start byte (write) pointer of a packet is already transferred to the read controller side (in Threshold mode), the packet is not dropped. - 1: All packets except the runt error packets are forwarded to the application or DMA. If the RSF bit is set and the Rx queue overflows when a partial packet is written, the packet is dropped irrespective of the setting of this bit. However, if the RSF bit is reset and the Rx queue overflows when a partial packet is written, a partial packet might be forwarded to the application or DMA.
RSF	5	rw	**Receive Queue Store and Forward** - 1: DWC_xgmac reads a packet from the Rx queue only after the complete packet has been written to it, ignoring the RTC field of this register. - 0: Rx queue operates in the Threshold (cut-through) mode, subject to the threshold specified by the RTC field of this register.
DIS_TCP_EF	6	rw	**Disable Dropping of TCP/IP Checksum Error Packets** - When this bit is set, the MAC does not drop the packets which only have the errors detected by the Receive Checksum Offload engine. Such packets have errors only in the encapsulated payload. There are no errors (including FCS error) in the Ethernet packet received by the MAC. - When this bit is reset, all error packets are dropped if the FEF bit is reset.
EHFC	7	rw	**Enable Hardware Flow Control** When this bit is set, the flow control signal operation, based on the fill-level of Rx queue, is enabled. When reset, the flow control operation is disabled.

(table continues...)

(continued)

Field	Bits	Type	Description
RQS	22:16	rw	**Receive Queue Size** This field indicates the size of the allocated Receive queues in blocks of 256 bytes. The sixteenth bit is the starting bit of this field. A value of 0 indicates 256 bytes.When this queue is used, a minimum value of 7 (size = 2048) is recommended so that the queue can store more than a complete max-sized Ethernet packet.
0	2, 15:8, 31:23	r	**Reserved** Read as 0; should be written with 0.

14.6.633 Port j MTL RxQ6 Missed Pkt Overflow Cnt

The Queue 6 Overflow Counter register contains the counter for packets missed because of Receive queue packet flush and packets discarded because of Receive queue overflow.

Portj_MTL_RxQ6_Missed_Pkt_Overflow_Cnt (j=0-1) Offset address: $11444_H+j*2000_H$

Port j MTL RxQ6 Missed Pkt Overflow Cnt Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MISCNTOVF	0				MISPKTCNT										
r	r				r										

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OVFCNTOVF	0				OVFPKTCNT										
r	r				r										

Field	Bits	Type	Description
OVFPKTCNT	10:0	r	**Overflow Packet Counter** This field indicates the number of packets discarded by DWC_xgmac because of Receive queue overflow. This counter is incremented each time DWC_xgmac discards an incoming packet because of overflow. This counter is reset when this register bits[7:0] is read by software.
OVFCNTOVF	15	r	**Overflow Counter Overflow Bit** When set, this bit indicates that OVFPKTCNT field crossed the maximum limit.
MISPKTCNT	26:16	r	**Missed Packet Counter** This field indicates the number of packets missed by DWC_xgmac because the host asserted packet-flush for this queue. This counter is incremented each time a packet is flushed for this queue. This counter is reset when this register's bits[23:16] is read by the software.

(table continues...)

(continued)

Field	Bits	Type	Description
MISCNTOVF	31	r	**Missed Packet Counter Overflow Bit** When set, this bit indicates that the MISPKTCNT field crossed the maximum limit.
0	14:11, 30:27	r	**Reserved** Read as 0; should be written with 0.

14.6.634 Port j MTL RxQ6 Debug

The Queue 6 Receive Debug register gives the debug status of various blocks related to the Receive queue.

Portj_MTL_RxQ6_Debug (j=0-1) Offset address: $11448_H+j*2000_H$

Port j MTL RxQ6 Debug Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		PRXQ													
r		r													

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0										RXQSTS		0	RRCSTS		RWCSTS
r										r		r	r		r

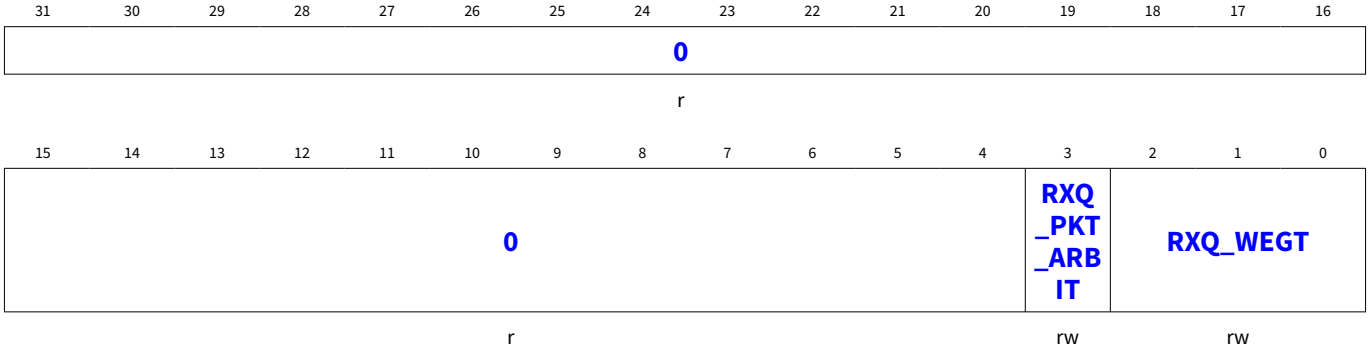
Field	Bits	Type	Description
RWCSTS	0	r	**MTL Rx Queue Write Controller Active Status** When high, this bit indicates that the MTL Rx queue Write controller is active, and it is transferring a received packet to the Rx Queue.
RRCSTS	2:1	r	**MTL Rx Queue Read Controller State** This field gives the state of the Rx queue Read controller: - 2'b00: Idle state - 2'b01: Reading packet data - 2'b10: Reading packet status (or timestamp) - 2'b11: Flushing the packet data and status
RXQSTS	5:4	r	**MTL Rx Queue Fill-Level Status** This field gives the status of the fill-level of the Rx Queue: - 2'b00: Rx Queue empty - 2'b01: Rx Queue fill-level below flow-control deactivate threshold - 2'b10: Rx Queue fill-level above flow-control activate threshold - 2'b11: Rx Queue full
PRXQ	29:16	r	**Number of Packets in Receive Queue** This field indicates the current number of packets in the Rx Queue. The theoretical maximum value for this field is 256KB/16B = 16K Packets, that is, Max_Queue_Size/Min_Packet_Size.
0	3, 15:6, 31:30	r	**Reserved** Read as 0; should be written with 0.

14.6.635 Port j MTL RxQ6 Control

The Queue 6 Receive Control register controls the receive arbitration and passing of received packets to the application.

Portj_MTL_RxQ6_Control (j=0-1) Offset address: $1144C_H+j*2000_H$

Port j MTL RxQ6 Control Kernel Reset value: $0000\ 0000_H$



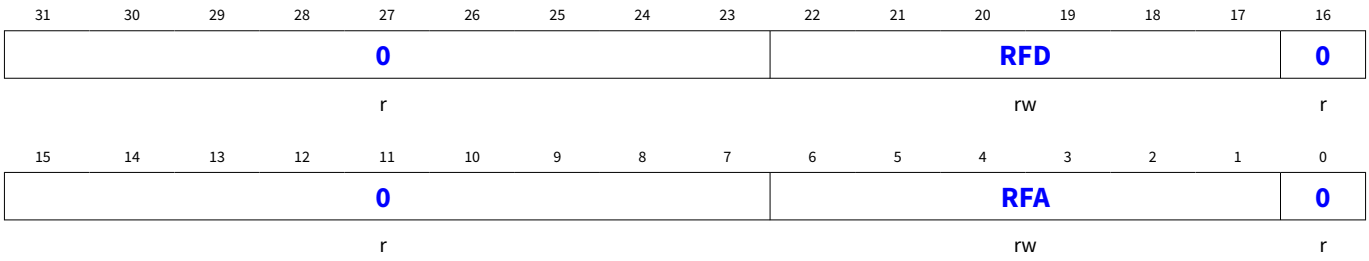
Field	Bits	Type	Description
RXQ_WEGT	2:0	rw	**Receive Queue Weight** This field indicates the weight assigned to the Rx Queue 6. The weight is used as the number of continuous PBL requests or contiguous packets (depending on the RXQ_PKT_ARBIT) allocated to the queue in one arbitration cycle.
RXQ_PKT_ARBIT	3	rw	**Receive Queue Packet Arbitration Control** - 1: MTL enables the packet-level arbitration. - 0: MTL enables the PBL based arbitration.
0	31:4	r	**Reserved** Read as 0; should be written with 0.

14.6.636 Port j MTL RxQ6 Flow Control

This register controls the activation and de-activation threshold in the queue for triggering the generation of Flow Control packets by MAC transmitter.

Portj_MTL_RxQ6_Flow_Control (j=0-1) Offset address: $11450_H+j*2000_H$

Port j MTL RxQ6 Flow Control Kernel Reset value: $0000\ 0000_H$



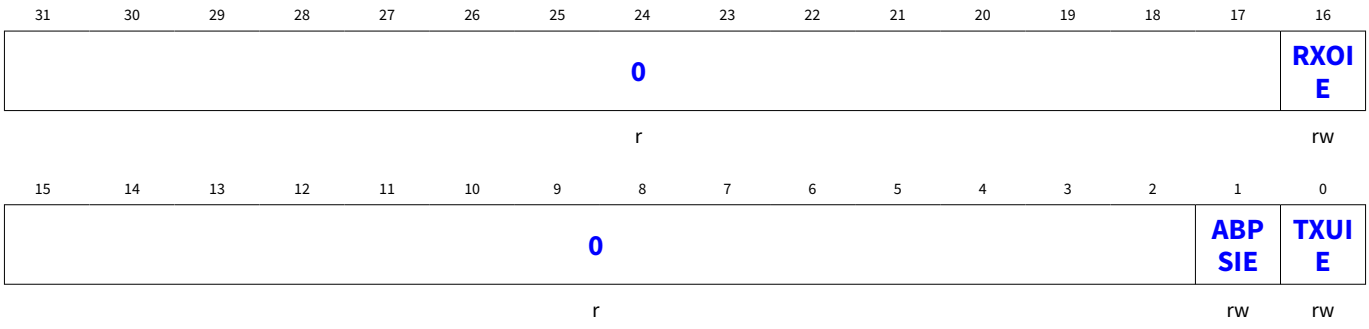
Field	Bits	Type	Description
RFA	6:1	rw	**Threshold for activating Flow Control** These bits control the threshold (fill-level of RxQ0) at which the flow control is activated - 6'b000000: RQS value minus 1 KB, that is, FULL - 1024 bytes - 6'b000001: RQS value minus 1.5 KB, that is, FULL - 1536 bytes - 6'b000010: RQS value minus 2 KB, that is, FULL - 2048 bytes. - ... - 6'b111111: RQS value minus 32.5 KB, that is, FULL - 33280 bytes The flow control is triggered when EHFC=1 and the queue size (RQS value in MTL_RxQ6_Operation Mode register) is equal to or more than 4KB. Note that the value to be programmed in RFA must be such that the threshold is positive number.
RFD	22:17	rw	**Threshold for deactivating Flow Control** These bits control the threshold (fill-level of RxQ0) at which the flow control is de-asserted once it is activated. - 6'b000000: RQS value minus 1 KB, that is, FULL - 1024 bytes - 6'b000001: RQS value minus 1.5 KB, that is, FULL - 1536 bytes - 6'b000010: RQS value minus 2 KB, that is, FULL - 2048 bytes. - ... - 6'b111111: RQS value minus 32.5 KB, that is, FULL - 33280 bytes The de-assertion is effective only after flow control is asserted as per the setting of RFA field. Note that the value to be programmed must be such that the threshold is positive number.
0	0, 16:7, 31:23	r	**Reserved** Read as 0; should be written with 0.

14.6.637 Port j MTL Q6 Interrupt Enable

This register contains the interrupt enable bits for Traffic class/Queue 6.

Portj_MTL_Q6_Interrupt_Enable (j=0-1) Offset address: $11470_H+j*2000_H$

Port j MTL Q6 Interrupt Enable Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
TXUIE	0	rw	**Transmit Queue Underflow Interrupt Enable** - 1: Transmit Queue Underflow interrupt is enabled. - 0: Transmit Queue Underflow interrupt is disabled.
ABPSIE	1	rw	**Average Bits Per Slot Interrupt Enable** When this bit is set, average bits per slot interrupt is enabled for Traffic Class 6. An interrupt is generated when ABPSIS is set for this traffic class. This bit is reserved and RO when the number of traffic classes selected is less than 6.
RXOIE	16	rw	**Receive Queue Overflow Interrupt Enable** - 1: Receive Queue Overflow interrupt is enabled. - 0: Receive Queue Overflow interrupt is disabled.
0	15:2, 31:17	r	**Reserved** Read as 0; should be written with 0.

14.6.638 Port j MTL Q6 Interrupt Status

This register contains the interrupt status bits for Traffic class/Queue 6.

Portj_MTL_Q6_Interrupt_Status (j=0-1) Offset address: $11474_H+j*2000_H$

Port j MTL Q6 Interrupt Status Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															RXOVFIS
r															rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0														ABPSIS	TXUNFIS
r														rw	rw

Field	Bits	Type	Description
TXUNFIS	0	rw	**Transmit Queue Underflow Interrupt Status** This bit indicates that the Transmit Queue had an Underflow during packet transmission. Transmission is suspended and an Underflow Error TDES3[2] is set. This bit is cleared when the application writes a 1 to this bit.
ABPSIS	1	rw	**Average Bits Per Slot Interrupt Status** When set, this bit indicates that the MAC has updated the ABS value for this traffic class. This bit is cleared when the application writes a 1 to this bit. This bit is reserved and RO when the number of traffic classes selected is less than 6.

(table continues...)

(continued)

Field	Bits	Type	Description
RXOVFIS	16	rw	**Receive Queue Overflow Interrupt Status** This bit indicates that the Receive Queue had an Overflow during packet reception. If partial packet is transferred to the application, the overflow status is set in the corresponding Rx status. This bit is cleared when the application writes a 1 to this bit.
0	15:2, 31:17	r	**Reserved** Read as 0; should be written with 0.

14.6.639 Port j MTL TxQ7 Operation Mode

The Queue 7 Transmit Operation Mode register establishes the Transmit queue operating modes and commands.

Portj_MTL_TxQ7_Operation_Mode (j=0-1) Offset address: $11480_H+j*2000_H$

Port j MTL TxQ7 Operation Mode Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0									TQS						
r									rw						

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0					Q2TCMAP			0	TTC			TXQEN		TSF	FTQ
r					rw			r	rw			rw		rw	rw

Field	Bits	Type	Description
FTQ	0	rw	**Flush Transmit Queue** When this bit is set, the Tx Queue Controller logic is reset to its default values. Therefore, all the data in the Tx queue is lost or flushed. This bit is internally reset when the flushing operation is complete. You must not write to this register until this bit is reset. The data which is already accepted by the MAC transmitter is not flushed. It is scheduled for transmission and results in underflow and runt packet transmission. Note: The flush operation is complete only when the Tx queue is empty. To complete this flush operation, the PHY Tx clock must be active.
TSF	1	rw	**Transmit Store and Forward** When this bit is set, the transmission starts when a full packet resides in the MTL Tx Queue. When this bit is set, the values specified in the TTC field are ignored. This bit must be changed only when the transmission is stopped.

(table continues...)

(continued)

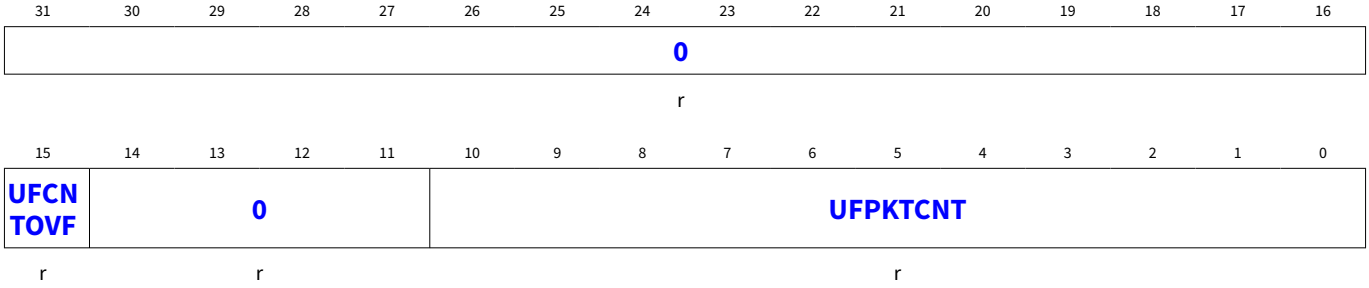
Field	Bits	Type	Description
TXQEN	3:2	rw	**Transmit Queue Enable** This field is used to enable/disable the transmit queue. - 2'b00: Not enabled - 2'b01: Audio Video Bridging - 2'b10: Enabled - 2'b11: Reserved In multiple Tx queues configuration, as the queue is disabled by default, enable the Tx Queue by programming this field in the AVB mode or default mode as desired.
TTC	6:4	rw	**Transmit Threshold Control** These bits control the threshold level of the MTL Tx Queue. Transmission starts when the packet size within the MTL Tx Queue is larger than the threshold. In addition, full packets with length less than the threshold are also transmitted. These bits are used only when the TSF bit is reset. - 3'b000: 64 - 3'b001: Reserved - 3'b010: 96 - 3'b011: 128 - 3'b100: 192 - 3'b101: 256 - 3'b110: 384 - 3'b111: 512
Q2TCMAP	10:8	rw	**Queue to Traffic Class Mapping** This field is used to map this Tx Queue to a traffic class (TC). - 3'b000: TC0 - 3'b001: TC1 - 3'b010: TC2 - 3'b011: TC3 - 3'b100: TC4 - 3'b101: TC5 - 3'b110: TC6 - 3'b111: TC7
TQS	22:16	rw	**Transmit Queue Size** This field indicates the size of the allocated Transmit queues in blocks of 256 bytes. The sixteenth bit is the starting bit of this field. A value of 0 indicates 256 bytes. When this queue is enabled (TXQEN=10), a minimum value of 7 (size = 2048) is recommended so that the queue can store more than a complete max-sized Ethernet packet.
0	7, 15:11, 31:23	r	**Reserved** Read as 0; should be written with 0.

14.6.640 Port j MTL TxQ7 Underflow

The Queue 7 Underflow Counter register contains the number of packets aborted because of Transmit queue underflow.

Portj_MTL_TxQ7_Underflow (j=0-1) Offset address: $11484_H+j*2000_H$

Port j MTL TxQ7 Underflow Kernel Reset value: $0000\ 0000_H$



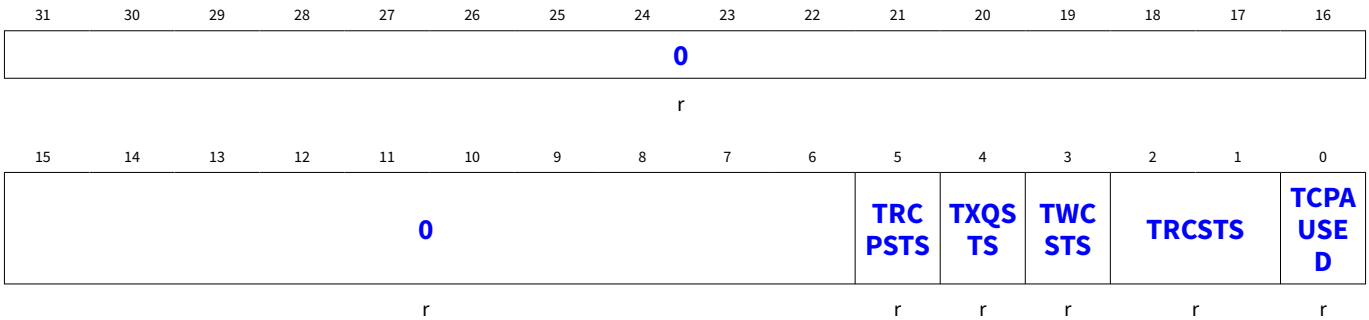
Field	Bits	Type	Description
UFPKTCNT	10:0	r	**Underflow Packet Counter** This field indicates the number of packets aborted by the controller because of Transmit Queue Underflow. This counter is incremented each time the MAC aborts an outgoing packet because of underflow. The counter is reset when this register is read with LSB of the Byte Enable for CSR access signal at 1'b1 .
UFCNTOVF	15	r	**Overflow Bit for Underflow Packet Counter** This bit is set every time the Tx Queue Underflow Packet Counter field overflows, that is, it has crossed the maximum count. In such a scenario, the overflow packet counter is reset to all-zeros and this bit indicates that the rollover happened.
0	14:11, 31:16	r	**Reserved** Read as 0; should be written with 0.

14.6.641 Port j MTL TxQ7 Debug

The Queue 7 Transmit Debug register gives the debug status of various blocks related to the Transmit Queue 7 and Traffic Class 7.

Portj_MTL_TxQ7_Debug (j=0-1) Offset address: $11488_H+j*2000_H$

Port j MTL TxQ7 Debug Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
TCPAUSED	0	r	**Traffic Class in Pause** - Reception of IEEE 802.3-2018 Pause packet
TRCSTS	2:1	r	**MTL Tx Queue Read Controller Status** This field indicates the state of the Tx Queue Read Controller: - 2'b00: Idle state - 2'b01: Read state (transferring data to the MAC transmitter) - 2'b10: Waiting for pending Tx Status from the MAC transmitter - 2'b11: Flushing the Tx queue because of the Packet Abort request from the MAC
TWCSTS	3	r	**MTL Tx Queue Write Controller Status** When high, this bit indicates that the MTL Tx Queue Write Controller is active, and it is transferring the data to the Tx Queue.
TXQSTS	4	r	**MTL Tx Queue Not Empty Status** When this bit is high, it indicates that the MTL Tx Queue is not empty and some data is left for transmission.
TRCPSTS	5	r	**MTL TRC Pending Status** When this bit is high, it indicates that the MTL Tx Read Controller (TRC) has a pending status from MAC.
0	31:6	r	**Reserved** Read as 0; should be written with 0.

14.6.642 Port j MTL TC7 ETS Control

This register configures the transmission selection algorithm, for Traffic Class 7.

Portj_MTL_TC7_ETS_Control (j=0-1) Offset address: $11490_H+j*2000_H$

Port j MTL TC7 ETS Control Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0									SLC			CC	CBSEN	TSA	
r									rw			rw	r	rw	

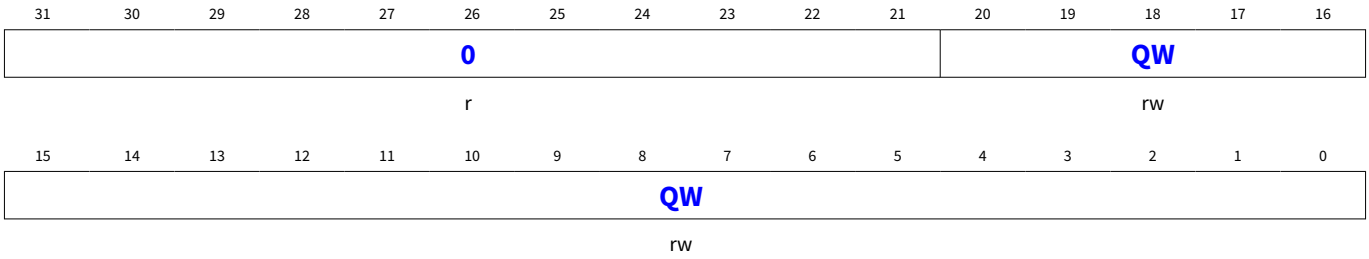
Field	Bits	Type	Description
TSA	1:0	rw	**Transmission Selection Algorithm** This field is used to assign a transmission selection algorithm for this traffic class. - 2'b00: Strict priority - 2'b01: CBS - 2'b10: ETS - 2'b11: Reserved
CBSEN	2	r	**Credit Based Shaper Enable** This field indicates that TSA field of this register is enabled for CBS
CC	3	rw	**Credit Control** When this bit is set, the accumulated credit parameter in the credit-based shaper algorithm logic is not reset to zero when there is positive credit and no packet to transmit in TC(#n). The credit accumulates even when there is no packet waiting in TC(#n) and another TC is transmitting. When this bit is reset, the accumulated credit parameter in the credit-based shaper algorithm logic is set to zero when there is positive credit and no packet to transmit in TC(#n). When there is no packet waiting in TC(#n) and other TC is transmitting, no credit is accumulated.
SLC	6:4	rw	**Slot Count** If the credit-based shaper algorithm is enabled, the software can program the number of slots (of 125 us duration) over which the average transmitted bits per slot, provided in the MTL_TC(#n)_ETS_Status register, need to be computed for TC(#n). The encoding is as follows: - 3'b000: 1 Slot - 3'b001: 2 Slots - 3'b010: 4 Slots - 3'b011: 8 Slots - 3'b100: 16 Slots - 3'b101-3'b111: Reserved
0	31:7	r	**Reserved** Read as 0; should be written with 0.

14.6.643 Port j MTL TC7 Quantum Weight

The Quantum or Weights register controls the average traffic transmitted in Traffic Class 7.

Portj_MTL_TC7_Quantum_Weight (j=0-1) Offset address: $11498_H+j*2000_H$

Port j MTL TC7 Quantum Weight Kernel Reset value: $0000\ 0000_H$



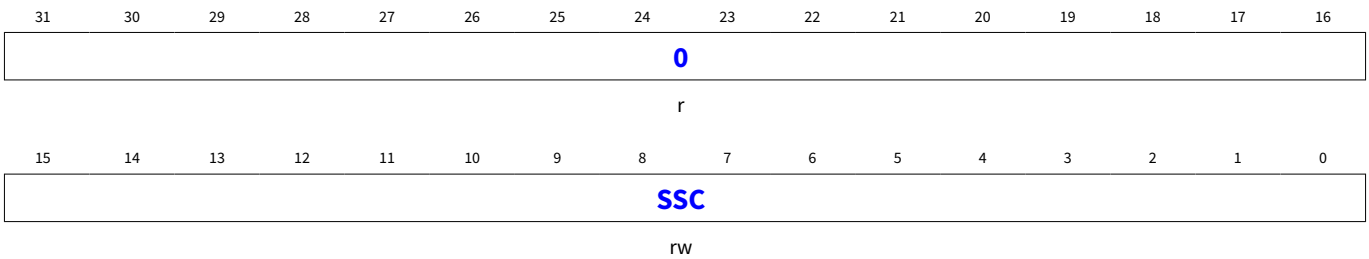
Field	Bits	Type	Description
QW	20:0	rw	**idleSlopeCredit, Quantum or Weights** - idleSlopeCredit -- When this TC is in AVB mode, this field contains the idleSlopeCredit value required for the credit-based shaper algorithm for Traffic Class 7. This is the rate of change of credit in bits per cycle (3.2 ns for 2.5/10 Gbps; 8 ns for 1000 Mbps) when the credit is increasing. The software must program this field with computed credit in bits per cycle scaled by 1024. The maximum value is portTransmitRate, that is, 0x2000 in 1000/2500 Mbps mode and 0x8000 in 10 Gbps mode. Bits[20:16] must be written to zero. This field is not valid for Traffic Class 0. - Weights -- When TSA is set to ETS and ETSALG is set to WRR algorithm for Traffic Class 1 traffic, this field contains the weight for this queue. The maximum value is 0x64. Bits[20:7] must be written to zero. This field is reserved and RO when the number of traffic classes selected is less than two. This field is reserved in corresponding Queues 8-15 register.
0	31:21	r	**Reserved** Read as 0; should be written with 0.

14.6.644 Port j MTL TC7 SendSlopeCredit

MTL registers for traffic classes and queues 1 registers.

Portj_MTL_TC7_SendSlopeCredit (j=0-1) Offset address: $1149C_H+j*2000_H$

Port j MTL TC7 SendSlopeCredit Kernel Reset value: $0000\ 0000_H$



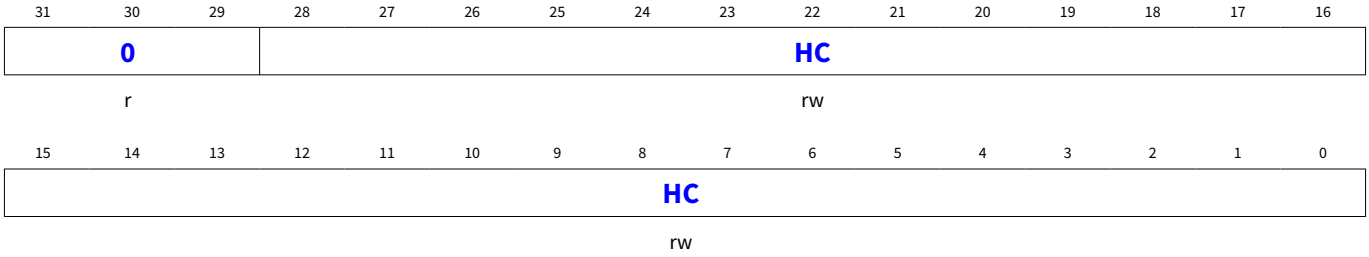
Field	Bits	Type	Description
SSC	15:0	rw	**sendSlopeCredit Value** - When Audio Video Bridging feature is enabled, this field contains the sendSlopeCredit value required for credit-based shaper algorithm for TC(#n). The software must program this field with computed credit in bits per cycle scaled by 1024. This field must be programmed with absolute sendSlopeCredit value. The credit-based shaper logic subtracts it from the accumulated credit when TC(#n) is selected for transmission. See the application note for more details on programming this value.
0	31:16	r	**Reserved** Read as 0; should be written with 0.

14.6.645 Port j MTL TC7 HiCredit

This register contains the hiCredit value required for the credit-based shaper algorithm for TC1.

Portj_MTL_TC7_HiCredit (j=0-1) Offset address: $114A0_H+j*2000_H$

Port j MTL TC7 HiCredit Kernel Reset value: $0000\ 0000_H$



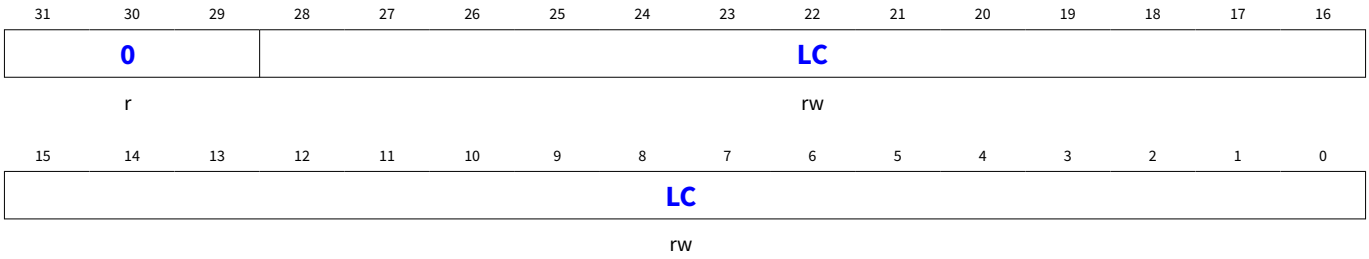
Field	Bits	Type	Description
HC	28:0	rw	**hiCredit Value** - When the Audio Video Bridging feature is enabled, this field contains the hiCredit value required for the credit-based shaper algorithm. This is the maximum value that can be accumulated in the credit parameter. This is specified in bits scaled by 1024. The maximum value is maxInterferenceSize, that is, best-effort maximum packet size (16384 bytes or 131072 bits). The value to be specified is 131072 * 1024 = 134217728 or 0x0800_0000
0	31:29	r	**Reserved** Read as 0; should be written with 0.

14.6.646 Port j MTL TC7 LoCredit

This register contains the loCredit value required for the credit-based shaper algorithm for TC1.

Portj_MTL_TC7_LoCredit (j=0-1) Offset address: $114A4_H+j*2000_H$

Port j MTL TC7 LoCredit Kernel Reset value: $0000\ 0000_H$



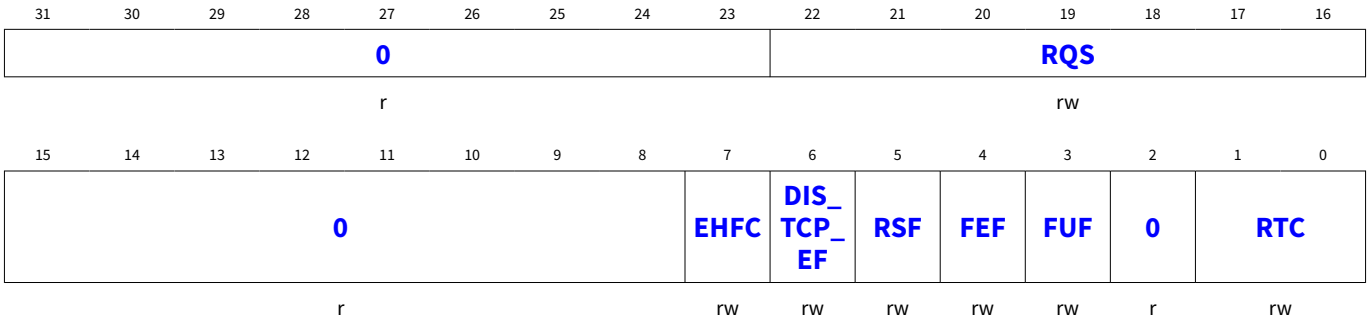
Field	Bits	Type	Description
LC	28:0	rw	**loCredit Value** - When Audio Video Bridging feature is enabled, this field contains the loCredit value required for the credit-based shaper algorithm. This is the minimum value that can be accumulated in the credit parameter. This is specified in bits scaled by 1024. The maximum value is maxPacketSize transmitted from this queue, that is, 16384 * 8 * 1024 = 134217728 or 0x0800_0000. Because it is a negative value, the programmed value is twos complement of the value, that is, 0x1800_0000.
0	31:29	r	**Reserved** Read as 0; should be written with 0.

14.6.647 Port j MTL RxQ7 Operation Mode

The Queue 7 Receive Operation Mode register establishes the Receive queue 7 operating modes.

Portj_MTL_RxQ7_Operation_Mode (j=0-1) Offset address: $114C0_H+j*2000_H$

Port j MTL RxQ7 Operation Mode Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
RTC	1:0	rw	**Receive Queue Threshold Control** These bits control the threshold level of the MTL Rx queue (in bytes): - 2'b00: 64 - 2'b01: Reserved - 2'b10: 96 - 2'b11: 128 The received packet is transferred to the application or DMA when the packet size within the MTL Rx queue is larger than the threshold. In addition, full packets with length less than the threshold are automatically transferred. The value of 11 is not applicable if the size of the configured Rx Queue is 128 bytes. This field is valid only when the RSF bit is zero. This field is ignored when the RSF bit is set to 1.
FUF	3	rw	**Forward Undersized Good Packets** - 1: Rx queue forwards the undersized good packets (with no Invalid-code/GMII/CRC/Length errors and length less than 64 bytes at Rx (X)GMII interface), including pad-bytes and CRC. - 0: Rx queue drops all packets of length less than 64 bytes.
FEF	4	rw	**Forward Error Packets** - 0: Rx queue drops packets with error status (CRC error, GMII_ER, watchdog timeout, or overflow). However, if the start byte (write) pointer of a packet is already transferred to the read controller side (in Threshold mode), the packet is not dropped. - 1: All packets except the runt error packets are forwarded to the application or DMA. If the RSF bit is set and the Rx queue overflows when a partial packet is written, the packet is dropped irrespective of the setting of this bit. However, if the RSF bit is reset and the Rx queue overflows when a partial packet is written, a partial packet might be forwarded to the application or DMA.
RSF	5	rw	**Receive Queue Store and Forward** - 1: DWC_xgmac reads a packet from the Rx queue only after the complete packet has been written to it, ignoring the RTC field of this register. - 0: Rx queue operates in the Threshold (cut-through) mode, subject to the threshold specified by the RTC field of this register.
DIS_TCP_EF	6	rw	**Disable Dropping of TCP/IP Checksum Error Packets** - When this bit is set, the MAC does not drop the packets which only have the errors detected by the Receive Checksum Offload engine. Such packets have errors only in the encapsulated payload. There are no errors (including FCS error) in the Ethernet packet received by the MAC. - When this bit is reset, all error packets are dropped if the FEF bit is reset.
EHFC	7	rw	**Enable Hardware Flow Control** When this bit is set, the flow control signal operation, based on the fill-level of Rx queue, is enabled. When reset, the flow control operation is disabled.

(table continues...)

(continued)

Field	Bits	Type	Description
RQS	22:16	rw	**Receive Queue Size** This field indicates the size of the allocated Receive queues in blocks of 256 bytes. The sixteenth bit is the starting bit of this field. A value of 0 indicates 256 bytes.When this queue is used, a minimum value of 7 (size = 2048) is recommended so that the queue can store more than a complete max-sized Ethernet packet.
0	2, 15:8, 31:23	r	**Reserved** Read as 0; should be written with 0.

14.6.648 Port j MTL RxQ7 Missed Pkt Overflow Cnt

The Queue 7 Overflow Counter register contains the counter for packets missed because of Receive queue packet flush and packets discarded because of Receive queue overflow.

Portj_MTL_RxQ7_Missed_Pkt_Overflow_Cnt (j=0-1) Offset address: $114C4_H+j*2000_H$

Port j MTL RxQ7 Missed Pkt Overflow Cnt Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MISCNTOVF	0				MISPKTCNT										
r	r				r										

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OVFCNTOVF	0				OVFPKTCNT										
r	r				r										

Field	Bits	Type	Description
OVFPKTCNT	10:0	r	**Overflow Packet Counter** This field indicates the number of packets discarded by DWC_xgmac because of Receive queue overflow. This counter is incremented each time DWC_xgmac discards an incoming packet because of overflow. This counter is reset when this register bits[7:0] is read by software.
OVFCNTOVF	15	r	**Overflow Counter Overflow Bit** When set, this bit indicates that OVFPKTCNT field crossed the maximum limit.
MISPKTCNT	26:16	r	**Missed Packet Counter** This field indicates the number of packets missed by DWC_xgmac because the host asserted packet-flush for this queue. This counter is incremented each time a packet is flushed for this queue. This counter is reset when this register's bits[23:16] is read by the software.

(table continues...)

(continued)

Field	Bits	Type	Description
MISCNTOVF	31	r	**Missed Packet Counter Overflow Bit** When set, this bit indicates that the MISPKTCNT field crossed the maximum limit.
0	14:11, 30:27	r	**Reserved** Read as 0; should be written with 0.

14.6.649 Port j MTL RxQ7 Debug

The Queue 7 Receive Debug register gives the debug status of various blocks related to the Receive queue.

Portj_MTL_RxQ7_Debug (j=0-1) Offset address: $114C8_H+j*2000_H$

Port j MTL RxQ7 Debug Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		PRXQ													
r		r													

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0										RXQSTS		0	RRCSTS		RWCSTS
r										r		r	r		r

Field	Bits	Type	Description
RWCSTS	0	r	**MTL Rx Queue Write Controller Active Status** When high, this bit indicates that the MTL Rx queue Write controller is active, and it is transferring a received packet to the Rx Queue.
RRCSTS	2:1	r	**MTL Rx Queue Read Controller State** This field gives the state of the Rx queue Read controller: - 2'b00: Idle state - 2'b01: Reading packet data - 2'b10: Reading packet status (or timestamp) - 2'b11: Flushing the packet data and status
RXQSTS	5:4	r	**MTL Rx Queue Fill-Level Status** This field gives the status of the fill-level of the Rx Queue: - 2'b00: Rx Queue empty - 2'b01: Rx Queue fill-level below flow-control deactivate threshold - 2'b10: Rx Queue fill-level above flow-control activate threshold - 2'b11: Rx Queue full
PRXQ	29:16	r	**Number of Packets in Receive Queue** This field indicates the current number of packets in the Rx Queue. The theoretical maximum value for this field is 256KB/16B = 16K Packets, that is, Max_Queue_Size/Min_Packet_Size.
0	3, 15:6, 31:30	r	**Reserved** Read as 0; should be written with 0.

14.6.650 Port j MTL RxQ7 Control

The Queue 7 Receive Control register controls the receive arbitration and passing of received packets to the application.

Portj_MTL_RxQ7_Control (j=0-1) Offset address: $114CC_H+j*2000_H$

Port j MTL RxQ7 Control Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												RXQ_PKT_ARBIT	RXQ_WEGT		
r												rw	rw		

Field	Bits	Type	Description
RXQ_WEGT	2:0	rw	**Receive Queue Weight** This field indicates the weight assigned to the Rx Queue 7. The weight is used as the number of continuous PBL requests or contiguous packets (depending on the RXQ_PKT_ARBIT) allocated to the queue in one arbitration cycle.
RXQ_PKT_ARBIT	3	rw	**Receive Queue Packet Arbitration Control** - 1: MTL enables the packet-level arbitration. - 0: MTL enables the PBL based arbitration.
0	31:4	r	**Reserved** Read as 0; should be written with 0.

14.6.651 Port j MTL RxQ7 Flow Control

This register controls the activation and de-activation threshold in the queue for triggering the generation of Flow Control packets by MAC transmitter.

Portj_MTL_RxQ7_Flow_Control (j=0-1) Offset address: $114D0_H+j*2000_H$

Port j MTL RxQ7 Flow Control Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0									RFD						0
r									rw						r

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0									RFA						0
r									rw						r

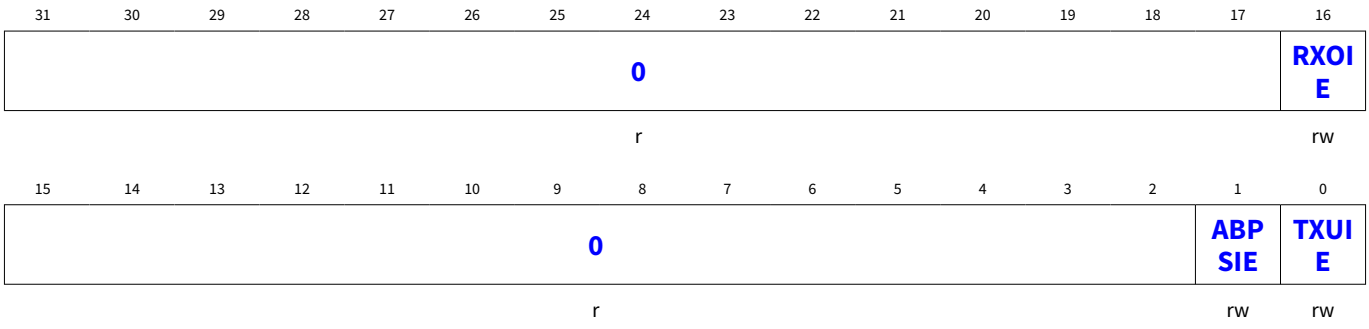
Field	Bits	Type	Description
RFA	6:1	rw	**Threshold for activating Flow Control** These bits control the threshold (fill-level of RxQ0) at which the flow control is activated - 6'b000000: RQS value minus 1 KB, that is, FULL - 1024 bytes - 6'b000001: RQS value minus 1.5 KB, that is, FULL - 1536 bytes - 6'b000010: RQS value minus 2 KB, that is, FULL - 2048 bytes. - ... - 6'b111111: RQS value minus 32.5 KB, that is, FULL - 33280 bytes The flow control is triggered when EHFC=1 and the queue size (RQS value in MTL_RxQ7_Operation Mode register) is equal to or more than 4KB. Note that the value to be programmed in RFA must be such that the threshold is positive number.
RFD	22:17	rw	**Threshold for deactivating Flow Control** These bits control the threshold (fill-level of RxQ0) at which the flow control is de-asserted once it is activated. - 6'b000000: RQS value minus 1 KB, that is, FULL - 1024 bytes - 6'b000001: RQS value minus 1.5 KB, that is, FULL - 1536 bytes - 6'b000010: RQS value minus 2 KB, that is, FULL - 2048 bytes. - ... - 6'b111111: RQS value minus 32.5 KB, that is, FULL - 33280 bytes The de-assertion is effective only after flow control is asserted as per the setting of RFA field. Note that the value to be programmed must be such that the threshold is positive number.
0	0, 16:7, 31:23	r	**Reserved** Read as 0; should be written with 0.

14.6.652 Port j MTL Q7 Interrupt Enable

This register contains the interrupt enable bits for Traffic class/Queue 7.

Portj_MTL_Q7_Interrupt_Enable (j=0-1) Offset address: $114F0_H+j*2000_H$

Port j MTL Q7 Interrupt Enable Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
TXUIE	0	rw	**Transmit Queue Underflow Interrupt Enable** - 1: Transmit Queue Underflow interrupt is enabled. - 0: Transmit Queue Underflow interrupt is disabled.
ABPSIE	1	rw	**Average Bits Per Slot Interrupt Enable** When this bit is set, average bits per slot interrupt is enabled for Traffic Class 7. An interrupt is generated when ABPSIS is set for this traffic class. This bit is reserved and RO when the number of traffic classes selected is less than 7.
RXOIE	16	rw	**Receive Queue Overflow Interrupt Enable** - 1: Receive Queue Overflow interrupt is enabled. - 0: Receive Queue Overflow interrupt is disabled.
0	15:2, 31:17	r	**Reserved** Read as 0; should be written with 0.

14.6.653 Port j MTL Q7 Interrupt Status

This register contains the interrupt status bits for Traffic class/Queue 7.

Portj_MTL_Q7_Interrupt_Status (j=0-1) Offset address: $114F4_H+j*2000_H$

Port j MTL Q7 Interrupt Status Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															RXOVFIS
r															rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0														ABPSIS	TXUNFIS
r														rw	rw

Field	Bits	Type	Description
TXUNFIS	0	rw	**Transmit Queue Underflow Interrupt Status** This bit indicates that the Transmit Queue had an Underflow during packet transmission. Transmission is suspended and an Underflow Error TDES3[2] is set. This bit is cleared when the application writes a 1 to this bit.
ABPSIS	1	rw	**Average Bits Per Slot Interrupt Status** When set, this bit indicates that the MAC has updated the ABS value for this traffic class. This bit is cleared when the application writes a 1 to this bit. This bit is reserved and RO when the number of traffic classes selected is less than 7.

(table continues...)

(continued)

Field	Bits	Type	Description
RXOVFIS	16	rw	**Receive Queue Overflow Interrupt Status** This bit indicates that the Receive Queue had an Overflow during packet reception. If partial packet is transferred to the application, the overflow status is set in the corresponding Rx status. This bit is cleared when the application writes a 1 to this bit.
0	15:2, 31:17	r	**Reserved** Read as 0; should be written with 0.

14.6.654 Forward Control

This register comprises of Forward Control settings applicable for all Bridge Ports except Host port.

Forward_Control Offset address: $1E000_H$

Forward Control Kernel Reset value: $0000\ 0040_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Q_CH_MAPPING_EN	PORT_SEL	PBL								0	P2P_RUNT_PKT_FWEI	GIANT_PKT_FWEI	IP_HDRERR_FWEI	PYLD_CHKERR_FWEI	FW_DIS_EIN
rw	rw	rw								r	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MAX_PKT_LENGTH															
rw															

Field	Bits	Type	Description
MAX_PKT_LENGTH	15:0	rw	**Maximum Packet Length for the Port to Port Forwarding Traffic** The maximum value of this field is 0x7FFF. Not used for 1-port and 2-port modes. Reserved for future multiple-port modes.
FW_DIS_EIN	16	rw	**Disable ATI Error Injection in the Forwarding Port for all Rx Packet Error types in the ingress port Values: - 0(ENABLE): ATI Error Injection Enabled - 1(DISABLE): ATI Error Injection Disabled**
PYLD_CHKERR_FWEI	17	rw	**Enable ATI Error Injection for the port forwarded packets which are received with IP Payload Checksum Error Status in the ingress port Values: - 0(DISABLE): ATI Error Injection Disabled for Rx packets with Payload Checksum Error - 1(ENABLE): ATI Error Injection Enabled for Rx packets with Payload Checksum Error**
IP_HDRERR_FWEI	18	rw	**Enable ATI Error Injection for the port forwarded packets which are received with IP Header Error Status in the ingress port Values: - 0(DISABLE): ATI Error Injection Disabled for Rx packets with IP Header Error - 1(ENABLE): ATI Error Injection Enabled for Rx packets with IP Header Error**

(table continues...)

(continued)

Field	Bits	Type	Description
GIANT_PKT_FWEI	19	rw	**Enable ATI Error Injection for the port forwarded packets which are received with Giant Packet Status in the ingress port** Values: - 0(DISABLE): ATI Error Injection Disabled for Rx packets with Giant Packet Status - 1(ENABLE): ATI Error Injection Enabled for Rx packets with Giant Packet Status
P2P_RUNT_PKT_FWEI	20	rw	**Enable ATI Error Injection for the port forwarded packets which are received with Good Runt Packet Status in the ingress port** Values: - 0(DISABLE): ATI Error Injection Disabled for Rx packets with Good Runt Packet Status - 1(ENABLE): ATI Error Injection Enabled for Rx packets with Good Runt Packet Status
PBL	29:22	rw	**Transmit Programmable Burst Length** These bits indicate the maximum number of beats to be transferred in one ATI/ARI data transfer in the forwarding path. This is the maximum value that is used in a single block ARI/ATI transfer. You can program PBL with any of the following values: 1, 2, 4, 8, 16, or 32. Any other value results in undefined behavior. Note: The maximum limit is equal to half of the Tx Queue depth (Queue Depth = Queue Size / Datawidth).
PORT_SEL	30	rw	**Set to 0x1 to enable Port 1 in one-port mode Set to 0x0 to enable Port 0 in one-port mode**
Q_CH_MAPPING_EN	31	rw	**Set to 0x1 to enable TxQ/RxC mapping in two-port mode** In one-port mode, no mapping is required. This bit enables 2-port mode.
0	21	r	**Reserved Fields** Read as 0; should be written with 0.

14.6.655 Port Flush and Loopback Control

This register provides RXQ flush enable and Port loopback mode enable.

Port_Flush_and_Loopback_Control Offset address: $1E004_H$

Port Flush and Loopback Control Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0							PTP_PORT_NUM	0						PORT_LPBK_EN	
r							rw	r						rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXQ_FLUSH_PORT1								RXQ_FLUSH_PORT0							
rw								rw							

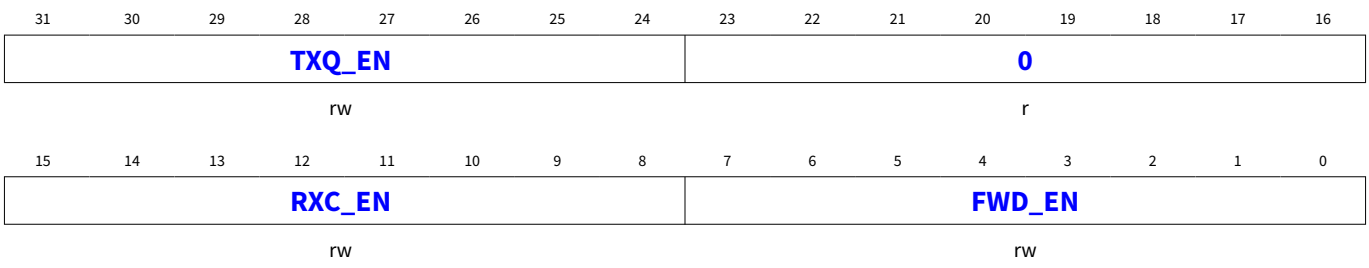
Field	Bits	Type	Description
RXQ_FLUSH_PORT0	7:0	rw	**To flush the Rx Queue corresponding to the selected RxC in Port 0 when the RxC is mapped to forwarding path** - Bit[0] : Flush the Rx queue corresponding to RxC0 of Port 0 - Bit[1] : Flush the Rx queue corresponding to RxC1 of Port 0 - Bit[2] : Flush the Rx queue corresponding to RxC2 of Port 0 - Bit[3] : Flush the Rx queue corresponding to RxC3 of Port 0 - Bit[4] : Flush the Rx queue corresponding to RxC4 of Port 0 - Bit[5] : Flush the Rx queue corresponding to RxC5 of Port 0 - Bit[6] : Flush the Rx queue corresponding to RxC6 of Port 0 - Bit[7] : Flush the Rx queue corresponding to RxC7 of Port 0
RXQ_FLUSH_PORT1	15:8	rw	**To flush the Rx Queue corresponding to the selected RxC in Port 1 when the RxC is mapped to forwarding path** - Bit[8] : Flush the Rx queue corresponding to RxC0 of Port 1 - Bit[9] : Flush the Rx queue corresponding to RxC1 of Port 1 - Bit[10] : Flush the Rx queue corresponding to RxC2 of Port 1 - Bit[11] : Flush the Rx queue corresponding to RxC3 of Port 1 - Bit[12] : Flush the Rx queue corresponding to RxC4 of Port 1 - Bit[13] : Flush the Rx queue corresponding to RxC5 of Port 1 - Bit[14] : Flush the Rx queue corresponding to RxC6 of Port 1 - Bit[15] : Flush the Rx queue corresponding to RxC7 of Port 1
PORT_LPBK_EN	17:16	rw	**To enable same-port loopback mode for port 1 (bit[17]) and port 0 (bit[16])**
PTP_PORT_NUM	24	rw	**Specifies the Port number for the PTP Timestamp Selection from Port 0/ Port 1** Values: - 0x0: Use Port 0 PTP Timestamp - 0x1: Use Port 1 PTP Timestamp Note: Software must program this field based on XGMAC's MAC_Timestamp_Control register.
0	23:18, 31:25	r	**Reserved Fields** Read as 0; should be written with 0.

14.6.656 Port 0 Control

This register enables the queues and channels and indicate whether it is a forwarding path.

Port_0_Control Offset address: $1E00C_H$

Port 0 Control Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
FWD_EN	7:0	rw	**Indicate whether any of the TxQ[7:0] is a forwarding path**
RXC_EN	15:8	rw	**Enable any of the RxC[7:0]**
TXQ_EN	31:24	rw	**Enable any of the TxQ[7:0]**
0	23:16	r	**Reserved Fields** Read as 0; should be written with 0.

14.6.657 TXQ Map Port 0

TxQ Map for Port 0. This register provides Port 0 TxQ connection mappings to either Tx EDMA or RxC of other Ethernet line ports in the forwarding path. When the TxQ of the Ethernet line Port is assigned to forwarding path (when FWD_EN bit of the TxQ in Port_0_Control register is set), the value of TXQ_Map_Port_0 provides the mapped RxC number of the source Ethernet line Port. When the TxQ of the Ethernet Line Port is mapped to Host traffic (when FWD_EN bit of the TxQ in Port_0_Control is zero), the value of TXQ_Map_Port_0 provides the mapped TxDMA channel number of the Host path.

TXQ_Map_Port_0 Offset address: $1E010_H$

TXQ Map Port 0 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TXQ7				TXQ6				TXQ5				TXQ4			
rw				rw				rw				rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXQ3				TXQ2				TXQ1				TXQ0			
rw				rw				rw				rw			

Field	Bits	Type	Description
TXQ0	3:0	rw	**TxQ0 connection mappings to either Tx EDMA channel of the Host port or Rx EDMA Channels of the other Ethernet line ports** Values: - 0x0: Mapped to RxC0 (Forward path) of Port 1 or TxEDMA0 (Host traffic) - 0x1: Mapped to RxC1 (Forward path) of Port 1 or TxEDMA1 (Host traffic) - 0x2: Mapped to RxC2 (Forward path) of Port 1 or TxEDMA2 (Host traffic) - 0x3: Mapped to RxC3 (Forward path) of Port 1 or TxEDMA3 (Host traffic) - 0x4: Mapped to RxC4 (Forward path) of Port 1 or TxEDMA4 (Host traffic) - 0x5: Mapped to RxC5 (Forward path) of Port 1 or TxEDMA5 (Host traffic) - 0x6: Mapped to RxC6 (Forward path) of Port 1 or TxEDMA6 (Host traffic) - 0x7: Mapped to RxC7 (Forward path) of Port 1 or TxEDMA7 (Host traffic) - 0x8 - 0xf: Reserved
TXQ1	7:4	rw	**TxQ1 connection mappings to either Tx EDMA channel of the Host port or Rx EDMA Channels of the other Ethernet line ports** Values: - 0x0: Mapped to RxC0 (Forward path) of Port 1 or TxEDMA0 (Host traffic) - 0x1: Mapped to RxC1 (Forward path) of Port 1 or TxEDMA1 (Host traffic) - 0x2: Mapped to RxC2 (Forward path) of Port 1 or TxEDMA2 (Host traffic) - 0x3: Mapped to RxC3 (Forward path) of Port 1 or TxEDMA3 (Host traffic) - 0x4: Mapped to RxC4 (Forward path) of Port 1 or TxEDMA4 (Host traffic) - 0x5: Mapped to RxC5 (Forward path) of Port 1 or TxEDMA5 (Host traffic) - 0x6: Mapped to RxC6 (Forward path) of Port 1 or TxEDMA6 (Host traffic) - 0x7: Mapped to RxC7 (Forward path) of Port 1 or TxEDMA7 (Host traffic) - 0x8 - 0xf: Reserved

(table continues...)

(continued)

Field	Bits	Type	Description
TXQ2	11:8	rw	**TxQ2 connection mappings to either Tx EDMA channel of the Host port or Rx EDMA Channels of the other Ethernet line ports** Values: - 0x0: Mapped to RxC0 (Forward path) of Port 1 or TxEDMA0 (Host traffic) - 0x1: Mapped to RxC1 (Forward path) of Port 1 or TxEDMA1 (Host traffic) - 0x2: Mapped to RxC2 (Forward path) of Port 1 or TxEDMA2 (Host traffic) - 0x3: Mapped to RxC3 (Forward path) of Port 1 or TxEDMA3 (Host traffic) - 0x4: Mapped to RxC4 (Forward path) of Port 1 or TxEDMA4 (Host traffic) - 0x5: Mapped to RxC5 (Forward path) of Port 1 or TxEDMA5 (Host traffic) - 0x6: Mapped to RxC6 (Forward path) of Port 1 or TxEDMA6 (Host traffic) - 0x7: Mapped to RxC7 (Forward path) of Port 1 or TxEDMA7 (Host traffic) - 0x8 - 0xf: Reserved
TXQ3	15:12	rw	**TxQ3 connection mappings to either Tx EDMA channel of the Host port or Rx EDMA Channels of the other Ethernet line ports** Values: - 0x0: Mapped to RxC0 (Forward path) of Port 1 or TxEDMA0 (Host traffic) - 0x1: Mapped to RxC1 (Forward path) of Port 1 or TxEDMA1 (Host traffic) - 0x2: Mapped to RxC2 (Forward path) of Port 1 or TxEDMA2 (Host traffic) - 0x3: Mapped to RxC3 (Forward path) of Port 1 or TxEDMA3 (Host traffic) - 0x4: Mapped to RxC4 (Forward path) of Port 1 or TxEDMA4 (Host traffic) - 0x5: Mapped to RxC5 (Forward path) of Port 1 or TxEDMA5 (Host traffic) - 0x6: Mapped to RxC6 (Forward path) of Port 1 or TxEDMA6 (Host traffic) - 0x7: Mapped to RxC7 (Forward path) of Port 1 or TxEDMA7 (Host traffic) - 0x8 - 0xf: Reserved

(table continues...)

(continued)

Field	Bits	Type	Description
TXQ4	19:16	rw	**TxQ4 connection mappings to either Tx EDMA channel of the Host port or Rx EDMA Channels of the other Ethernet line ports** Values: - 0x0: Mapped to RxC0 (Forward path) of Port 1 or TxEDMA0 (Host traffic) - 0x1: Mapped to RxC1 (Forward path) of Port 1 or TxEDMA1 (Host traffic) - 0x2: Mapped to RxC2 (Forward path) of Port 1 or TxEDMA2 (Host traffic) - 0x3: Mapped to RxC3 (Forward path) of Port 1 or TxEDMA3 (Host traffic) - 0x4: Mapped to RxC4 (Forward path) of Port 1 or TxEDMA4 (Host traffic) - 0x5: Mapped to RxC5 (Forward path) of Port 1 or TxEDMA5 (Host traffic) - 0x6: Mapped to RxC6 (Forward path) of Port 1 or TxEDMA6 (Host traffic) - 0x7: Mapped to RxC7 (Forward path) of Port 1 or TxEDMA7 (Host traffic) - 0x8 - 0xf: Reserved
TXQ5	23:20	rw	**TxQ5 connection mappings to either Tx EDMA channel of the Host port or Rx EDMA Channels of the other Ethernet line ports** Values: - 0x0: Mapped to RxC0 (Forward path) of Port 1 or TxEDMA0 (Host traffic) - 0x1: Mapped to RxC1 (Forward path) of Port 1 or TxEDMA1 (Host traffic) - 0x2: Mapped to RxC2 (Forward path) of Port 1 or TxEDMA2 (Host traffic) - 0x3: Mapped to RxC3 (Forward path) of Port 1 or TxEDMA3 (Host traffic) - 0x4: Mapped to RxC4 (Forward path) of Port 1 or TxEDMA4 (Host traffic) - 0x5: Mapped to RxC5 (Forward path) of Port 1 or TxEDMA5 (Host traffic) - 0x6: Mapped to RxC6 (Forward path) of Port 1 or TxEDMA6 (Host traffic) - 0x7: Mapped to RxC7 (Forward path) of Port 1 or TxEDMA7 (Host traffic) - 0x8 - 0xf: Reserved

(table continues...)

(continued)

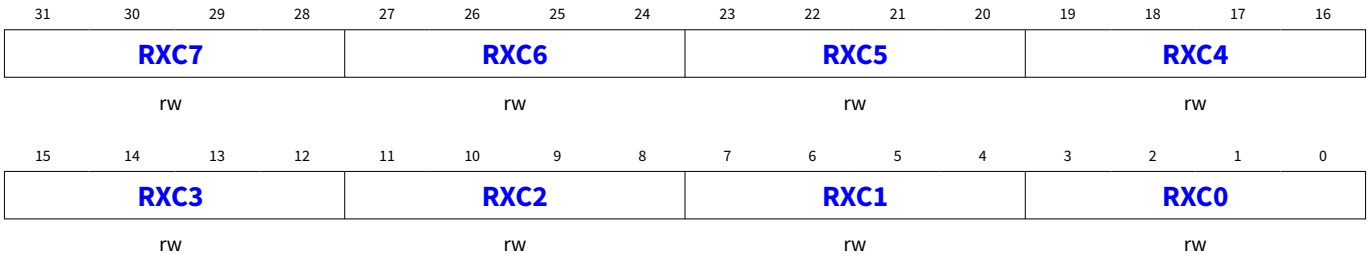
Field	Bits	Type	Description
TXQ6	27:24	rw	**TxQ6 connection mappings to either Tx EDMA channel of the Host port or Rx EDMA Channels of the other Ethernet line ports** Values: - 0x0: Mapped to RxC0 (Forward path) of Port 1 or TxEDMA0 (Host traffic) - 0x1: Mapped to RxC1 (Forward path) of Port 1 or TxEDMA1 (Host traffic) - 0x2: Mapped to RxC2 (Forward path) of Port 1 or TxEDMA2 (Host traffic) - 0x3: Mapped to RxC3 (Forward path) of Port 1 or TxEDMA3 (Host traffic) - 0x4: Mapped to RxC4 (Forward path) of Port 1 or TxEDMA4 (Host traffic) - 0x5: Mapped to RxC5 (Forward path) of Port 1 or TxEDMA5 (Host traffic) - 0x6: Mapped to RxC6 (Forward path) of Port 1 or TxEDMA6 (Host traffic) - 0x7: Mapped to RxC7 (Forward path) of Port 1 or TxEDMA7 (Host traffic) - 0x8 - 0xf: Reserved
TXQ7	31:28	rw	**TxQ7 connection mappings to either Tx EDMA channel of the Host port or Rx EDMA Channels of the other Ethernet line ports** Values: - 0x0: Mapped to RxC0 (Forward path) of Port 1 or TxEDMA0 (Host traffic) - 0x1: Mapped to RxC1 (Forward path) of Port 1 or TxEDMA1 (Host traffic) - 0x2: Mapped to RxC2 (Forward path) of Port 1 or TxEDMA2 (Host traffic) - 0x3: Mapped to RxC3 (Forward path) of Port 1 or TxEDMA3 (Host traffic) - 0x4: Mapped to RxC4 (Forward path) of Port 1 or TxEDMA4 (Host traffic) - 0x5: Mapped to RxC5 (Forward path) of Port 1 or TxEDMA5 (Host traffic) - 0x6: Mapped to RxC6 (Forward path) of Port 1 or TxEDMA6 (Host traffic) - 0x7: Mapped to RxC7 (Forward path) of Port 1 or TxEDMA7 (Host traffic) - 0x8 - 0xf: Reserved

14.6.658 RXC Map Port 0

This register provides Port 0 RxC connection mappings to Rx EDMA Channels. When the RxC of Ethernet Line Port is not participating in the forwarding traffic i.e the RxC number is not mapped to the TXQ of other Ethernet Line Ports, the value programmed in RXC field corresponds to the Rx EDMA of the Host path.

RXC_Map_Port_0 Offset address: $1E014_H$

RXC Map Port 0 Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
RXC0	3:0	rw	**RxC0 connection mappings to Rx EDMA Channel of Host port** Values: - 0x0: Mapped to RxEDMA0. - 0x1: Mapped to RxEDMA1. - 0x2: Mapped to RxEDMA2. - 0x3: Mapped to RxEDMA3. - 0x4: Mapped to RxEDMA4. - 0x5: Mapped to RxEDMA5. - 0x6: Mapped to RxEDMA6. - 0x7: Mapped to RxEDMA7. - 0x8 - 0xf: Reserved
RXC1	7:4	rw	**RxC1 connection mappings to Rx EDMA Channel of Host port** Values: - 0x0: Mapped to RxEDMA0. - 0x1: Mapped to RxEDMA1. - 0x2: Mapped to RxEDMA2. - 0x3: Mapped to RxEDMA3. - 0x4: Mapped to RxEDMA4. - 0x5: Mapped to RxEDMA5. - 0x6: Mapped to RxEDMA6. - 0x7: Mapped to RxEDMA7. - 0x8 - 0xf: Reserved
RXC2	11:8	rw	**RxC2 connection mappings to Rx EDMA Channel of Host port** Values: - 0x0: Mapped to RxEDMA0. - 0x1: Mapped to RxEDMA1. - 0x2: Mapped to RxEDMA2. - 0x3: Mapped to RxEDMA3. - 0x4: Mapped to RxEDMA4. - 0x5: Mapped to RxEDMA5. - 0x6: Mapped to RxEDMA6. - 0x7: Mapped to RxEDMA7. - 0x8 - 0xf: Reserved

(table continues...)

(continued)

Field	Bits	Type	Description
RXC3	15:12	rw	**RxC3 connection mappings to Rx EDMA Channel of Host port** Values: - 0x0: Mapped to RxEDMA0. - 0x1: Mapped to RxEDMA1. - 0x2: Mapped to RxEDMA2. - 0x3: Mapped to RxEDMA3. - 0x4: Mapped to RxEDMA4. - 0x5: Mapped to RxEDMA5. - 0x6: Mapped to RxEDMA6. - 0x7: Mapped to RxEDMA7. - 0x8 - 0xf: Reserved
RXC4	19:16	rw	**RxC4 connection mappings to Rx EDMA Channel of Host port** Values: - 0x0: Mapped to RxEDMA0. - 0x1: Mapped to RxEDMA1. - 0x2: Mapped to RxEDMA2. - 0x3: Mapped to RxEDMA3. - 0x4: Mapped to RxEDMA4. - 0x5: Mapped to RxEDMA5. - 0x6: Mapped to RxEDMA6. - 0x7: Mapped to RxEDMA7. - 0x8 - 0xf: Reserved
RXC5	23:20	rw	**RxC5 connection mappings to Rx EDMA Channel of Host port** Values: - 0x0: Mapped to RxEDMA0. - 0x1: Mapped to RxEDMA1. - 0x2: Mapped to RxEDMA2. - 0x3: Mapped to RxEDMA3. - 0x4: Mapped to RxEDMA4. - 0x5: Mapped to RxEDMA5. - 0x6: Mapped to RxEDMA6. - 0x7: Mapped to RxEDMA7. - 0x8 - 0xf: Reserved

(table continues...)

(continued)

Field	Bits	Type	Description
RXC6	27:24	rw	**RxC6 connection mappings to Rx EDMA Channel of Host port** Values: - 0x0: Mapped to RxEDMA0. - 0x1: Mapped to RxEDMA1. - 0x2: Mapped to RxEDMA2. - 0x3: Mapped to RxEDMA3. - 0x4: Mapped to RxEDMA4. - 0x5: Mapped to RxEDMA5. - 0x6: Mapped to RxEDMA6. - 0x7: Mapped to RxEDMA7. - 0x8 - 0xf: Reserved
RXC7	31:28	rw	**RxC7 connection mappings to Rx EDMA Channel of Host port** Values: - 0x0: Mapped to RxEDMA0. - 0x1: Mapped to RxEDMA1. - 0x2: Mapped to RxEDMA2. - 0x3: Mapped to RxEDMA3. - 0x4: Mapped to RxEDMA4. - 0x5: Mapped to RxEDMA5. - 0x6: Mapped to RxEDMA6. - 0x7: Mapped to RxEDMA7. - 0x8 - 0xf: Reserved

14.6.659 Port 1 Control

This register enables the queues and channels and indicate whether it is a forwarding path.

Port_1_Control Offset address: $1E018_H$

Port 1 Control Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TXQ_EN								0							
rw								r							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXC_EN								FWD_EN							
rw								rw							

Field	Bits	Type	Description
FWD_EN	7:0	rw	**Indicate whether any of the TxQ[7:0] is a forwarding path**
RXC_EN	15:8	rw	**Enable any of the RxC[7:0]**
TXQ_EN	31:24	rw	**Enable any of the TxQ[7:0]**
0	23:16	r	**Reserved Fields** Read as 0; should be written with 0.

14.6.660 TXQ Map Port 1

TxQ Map for Port 1. This register provides Port 1 TxQ connection mappings to either Tx EDMA or RxC of other Ethernet line ports in the forwarding path. When the TxQ of the Ethernet line Port is assigned to forwarding path (when FWD_EN bit of the TxQ in Port_1_Control register is set), the value of TXQ_Map_Port_1 provides the mapped RxC number of the source Ethernet line Port. When the TxQ of the Ethernet Line Port is mapped to Host traffic (when FWD_EN bit of the TxQ in Port_1_Control is zero), the value of TXQ_Map_Port_1 provides the mapped TxDMA channel number of the Host path.

TXQ_Map_Port_1 Offset address: $1E01C_H$

TXQ Map Port 1 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TXQ7				TXQ6				TXQ5				TXQ4			
rw				rw				rw				rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXQ3				TXQ2				TXQ1				TXQ0			
rw				rw				rw				rw			

Field	Bits	Type	Description
TXQ0	3:0	rw	**TxQ0 connection mappings to either Tx EDMA channel of the Host port or Rx EDMA Channels of the other Ethernet line ports** Values: - 0x0: Mapped to RxC0 (Forward path) of Port 0 or TxEDMA0 (Host traffic) - 0x1: Mapped to RxC1 (Forward path) of Port 0 or TxEDMA1 (Host traffic) - 0x2: Mapped to RxC2 (Forward path) of Port 0 or TxEDMA2 (Host traffic) - 0x3: Mapped to RxC3 (Forward path) of Port 0 or TxEDMA3 (Host traffic) - 0x4: Mapped to RxC4 (Forward path) of Port 0 or TxEDMA4 (Host traffic) - 0x5: Mapped to RxC5 (Forward path) of Port 0 or TxEDMA5 (Host traffic) - 0x6: Mapped to RxC6 (Forward path) of Port 0 or TxEDMA6 (Host traffic) - 0x7: Mapped to RxC7 (Forward path) of Port 0 or TxEDMA7 (Host traffic) - 0x8 - 0xf: Reserved

(table continues...)

(continued)

Field	Bits	Type	Description
TXQ1	7:4	rw	**TxQ1 connection mappings to either Tx EDMA channel of the Host port or Rx EDMA Channels of the other Ethernet line ports** Values: - 0x0: Mapped to RxC0 (Forward path) of Port 0 or TxEDMA0 (Host traffic) - 0x1: Mapped to RxC1 (Forward path) of Port 0 or TxEDMA1 (Host traffic) - 0x2: Mapped to RxC2 (Forward path) of Port 0 or TxEDMA2 (Host traffic) - 0x3: Mapped to RxC3 (Forward path) of Port 0 or TxEDMA3 (Host traffic) - 0x4: Mapped to RxC4 (Forward path) of Port 0 or TxEDMA4 (Host traffic) - 0x5: Mapped to RxC5 (Forward path) of Port 0 or TxEDMA5 (Host traffic) - 0x6: Mapped to RxC6 (Forward path) of Port 0 or TxEDMA6 (Host traffic) - 0x7: Mapped to RxC7 (Forward path) of Port 0 or TxEDMA7 (Host traffic) - 0x8 - 0xf: Reserved
TXQ2	11:8	rw	**TxQ2 connection mappings to either Tx EDMA channel of the Host port or Rx EDMA Channels of the other Ethernet line ports** Values: - 0x0: Mapped to RxC0 (Forward path) of Port 0 or TxEDMA0 (Host traffic) - 0x1: Mapped to RxC1 (Forward path) of Port 0 or TxEDMA1 (Host traffic) - 0x2: Mapped to RxC2 (Forward path) of Port 0 or TxEDMA2 (Host traffic) - 0x3: Mapped to RxC3 (Forward path) of Port 0 or TxEDMA3 (Host traffic) - 0x4: Mapped to RxC4 (Forward path) of Port 0 or TxEDMA4 (Host traffic) - 0x5: Mapped to RxC5 (Forward path) of Port 0 or TxEDMA5 (Host traffic) - 0x6: Mapped to RxC6 (Forward path) of Port 0 or TxEDMA6 (Host traffic) - 0x7: Mapped to RxC7 (Forward path) of Port 0 or TxEDMA7 (Host traffic) - 0x8 - 0xf: Reserved

(table continues...)

(continued)

Field	Bits	Type	Description
TXQ3	15:12	rw	**TxQ3 connection mappings to either Tx EDMA channel of the Host port or Rx EDMA Channels of the other Ethernet line ports** Values: - 0x0: Mapped to RxC0 (Forward path) of Port 0 or TxEDMA0 (Host traffic) - 0x1: Mapped to RxC1 (Forward path) of Port 0 or TxEDMA1 (Host traffic) - 0x2: Mapped to RxC2 (Forward path) of Port 0 or TxEDMA2 (Host traffic) - 0x3: Mapped to RxC3 (Forward path) of Port 0 or TxEDMA3 (Host traffic) - 0x4: Mapped to RxC4 (Forward path) of Port 0 or TxEDMA4 (Host traffic) - 0x5: Mapped to RxC5 (Forward path) of Port 0 or TxEDMA5 (Host traffic) - 0x6: Mapped to RxC6 (Forward path) of Port 0 or TxEDMA6 (Host traffic) - 0x7: Mapped to RxC7 (Forward path) of Port 0 or TxEDMA7 (Host traffic) - 0x8 - 0xf: Reserved
TXQ4	19:16	rw	**TxQ4 connection mappings to either Tx EDMA channel of the Host port or Rx EDMA Channels of the other Ethernet line ports** Values: - 0x0: Mapped to RxC0 (Forward path) of Port 0 or TxEDMA0 (Host traffic) - 0x1: Mapped to RxC1 (Forward path) of Port 0 or TxEDMA1 (Host traffic) - 0x2: Mapped to RxC2 (Forward path) of Port 0 or TxEDMA2 (Host traffic) - 0x3: Mapped to RxC3 (Forward path) of Port 0 or TxEDMA3 (Host traffic) - 0x4: Mapped to RxC4 (Forward path) of Port 0 or TxEDMA4 (Host traffic) - 0x5: Mapped to RxC5 (Forward path) of Port 0 or TxEDMA5 (Host traffic) - 0x6: Mapped to RxC6 (Forward path) of Port 0 or TxEDMA6 (Host traffic) - 0x7: Mapped to RxC7 (Forward path) of Port 0 or TxEDMA7 (Host traffic) - 0x8 - 0xf: Reserved

(table continues...)

(continued)

Field	Bits	Type	Description
TXQ5	23:20	rw	**TxQ5 connection mappings to either Tx EDMA channel of the Host port or Rx EDMA Channels of the other Ethernet line ports** Values: - 0x0: Mapped to RxC0 (Forward path) of Port 0 or TxEDMA0 (Host traffic) - 0x1: Mapped to RxC1 (Forward path) of Port 0 or TxEDMA1 (Host traffic) - 0x2: Mapped to RxC2 (Forward path) of Port 0 or TxEDMA2 (Host traffic) - 0x3: Mapped to RxC3 (Forward path) of Port 0 or TxEDMA3 (Host traffic) - 0x4: Mapped to RxC4 (Forward path) of Port 0 or TxEDMA4 (Host traffic) - 0x5: Mapped to RxC5 (Forward path) of Port 0 or TxEDMA5 (Host traffic) - 0x6: Mapped to RxC6 (Forward path) of Port 0 or TxEDMA6 (Host traffic) - 0x7: Mapped to RxC7 (Forward path) of Port 0 or TxEDMA7 (Host traffic) - 0x8 - 0xf: Reserved
TXQ6	27:24	rw	**TxQ6 connection mappings to either Tx EDMA channel of the Host port or Rx EDMA Channels of the other Ethernet line ports** Values: - 0x0: Mapped to RxC0 (Forward path) of Port 0 or TxEDMA0 (Host traffic) - 0x1: Mapped to RxC1 (Forward path) of Port 0 or TxEDMA1 (Host traffic) - 0x2: Mapped to RxC2 (Forward path) of Port 0 or TxEDMA2 (Host traffic) - 0x3: Mapped to RxC3 (Forward path) of Port 0 or TxEDMA3 (Host traffic) - 0x4: Mapped to RxC4 (Forward path) of Port 0 or TxEDMA4 (Host traffic) - 0x5: Mapped to RxC5 (Forward path) of Port 0 or TxEDMA5 (Host traffic) - 0x6: Mapped to RxC6 (Forward path) of Port 0 or TxEDMA6 (Host traffic) - 0x7: Mapped to RxC7 (Forward path) of Port 0 or TxEDMA7 (Host traffic) - 0x8 - 0xf: Reserved

(table continues...)

(continued)

Field	Bits	Type	Description
TXQ7	31:28	rw	**TxQ7 connection mappings to either Tx EDMA channel of the Host port or Rx EDMA Channels of the other Ethernet line ports** Values: - 0x0: Mapped to RxC0 (Forward path) of Port 0 or TxEDMA0 (Host traffic) - 0x1: Mapped to RxC1 (Forward path) of Port 0 or TxEDMA1 (Host traffic) - 0x2: Mapped to RxC2 (Forward path) of Port 0 or TxEDMA2 (Host traffic) - 0x3: Mapped to RxC3 (Forward path) of Port 0 or TxEDMA3 (Host traffic) - 0x4: Mapped to RxC4 (Forward path) of Port 0 or TxEDMA4 (Host traffic) - 0x5: Mapped to RxC5 (Forward path) of Port 0 or TxEDMA5 (Host traffic) - 0x6: Mapped to RxC6 (Forward path) of Port 0 or TxEDMA6 (Host traffic) - 0x7: Mapped to RxC7 (Forward path) of Port 0 or TxEDMA7 (Host traffic) - 0x8 - 0xf: Reserved

14.6.661 RXC Map Port 1

This register provides Port 1 RxC connection mappings to Rx EDMA Channels. When the RxC of Ethernet Line Port is not participating in the forwarding traffic i.e the RxC number is not mapped to the TXQ of other Ethernet Line Ports, the value programmed in RXC field corresponds to the Rx EDMA of the Host path.

RXC_Map_Port_1 Offset address: $1E020_H$

RXC Map Port 1 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RXC7				RXC6				RXC5				RXC4			
rw				rw				rw				rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXC3				RXC2				RXC1				RXC0			
rw				rw				rw				rw			

Field	Bits	Type	Description
RXC0	3:0	rw	**RxC0 connection mappings to Rx EDMA Channel of Host port** Values: - 0x0: Mapped to RxEDMA0. - 0x1: Mapped to RxEDMA1. - 0x2: Mapped to RxEDMA2. - 0x3: Mapped to RxEDMA3. - 0x4: Mapped to RxEDMA4. - 0x5: Mapped to RxEDMA5. - 0x6: Mapped to RxEDMA6. - 0x7: Mapped to RxEDMA7. - 0x8 - 0xf: Reserved
RXC1	7:4	rw	**RxC1 connection mappings to Rx EDMA Channel of Host port** Values: - 0x0: Mapped to RxEDMA0. - 0x1: Mapped to RxEDMA1. - 0x2: Mapped to RxEDMA2. - 0x3: Mapped to RxEDMA3. - 0x4: Mapped to RxEDMA4. - 0x5: Mapped to RxEDMA5. - 0x6: Mapped to RxEDMA6. - 0x7: Mapped to RxEDMA7. - 0x8 - 0xf: Reserved
RXC2	11:8	rw	**RxC2 connection mappings to Rx EDMA Channel of Host port** Values: - 0x0: Mapped to RxEDMA0. - 0x1: Mapped to RxEDMA1. - 0x2: Mapped to RxEDMA2. - 0x3: Mapped to RxEDMA3. - 0x4: Mapped to RxEDMA4. - 0x5: Mapped to RxEDMA5. - 0x6: Mapped to RxEDMA6. - 0x7: Mapped to RxEDMA7. - 0x8 - 0xf: Reserved

(table continues...)

(continued)

Field	Bits	Type	Description
RXC3	15:12	rw	**RxC3 connection mappings to Rx EDMA Channel of Host port** Values: - 0x0: Mapped to RxEDMA0. - 0x1: Mapped to RxEDMA1. - 0x2: Mapped to RxEDMA2. - 0x3: Mapped to RxEDMA3. - 0x4: Mapped to RxEDMA4. - 0x5: Mapped to RxEDMA5. - 0x6: Mapped to RxEDMA6. - 0x7: Mapped to RxEDMA7. - 0x8 - 0xf: Reserved
RXC4	19:16	rw	**RxC4 connection mappings to Rx EDMA Channel of Host port** Values: - 0x0: Mapped to RxEDMA0. - 0x1: Mapped to RxEDMA1. - 0x2: Mapped to RxEDMA2. - 0x3: Mapped to RxEDMA3. - 0x4: Mapped to RxEDMA4. - 0x5: Mapped to RxEDMA5. - 0x6: Mapped to RxEDMA6. - 0x7: Mapped to RxEDMA7. - 0x8 - 0xf: Reserved
RXC5	23:20	rw	**RxC5 connection mappings to Rx EDMA Channel of Host port** Values: - 0x0: Mapped to RxEDMA0. - 0x1: Mapped to RxEDMA1. - 0x2: Mapped to RxEDMA2. - 0x3: Mapped to RxEDMA3. - 0x4: Mapped to RxEDMA4. - 0x5: Mapped to RxEDMA5. - 0x6: Mapped to RxEDMA6. - 0x7: Mapped to RxEDMA7. - 0x8 - 0xf: Reserved

(table continues...)

(continued)

Field	Bits	Type	Description
RXC6	27:24	rw	**RxC6 connection mappings to Rx EDMA Channel of Host port** Values: - 0x0: Mapped to RxEDMA0. - 0x1: Mapped to RxEDMA1. - 0x2: Mapped to RxEDMA2. - 0x3: Mapped to RxEDMA3. - 0x4: Mapped to RxEDMA4. - 0x5: Mapped to RxEDMA5. - 0x6: Mapped to RxEDMA6. - 0x7: Mapped to RxEDMA7. - 0x8 - 0xf: Reserved
RXC7	31:28	rw	**RxC7 connection mappings to Rx EDMA Channel of Host port** Values: - 0x0: Mapped to RxEDMA0. - 0x1: Mapped to RxEDMA1. - 0x2: Mapped to RxEDMA2. - 0x3: Mapped to RxEDMA3. - 0x4: Mapped to RxEDMA4. - 0x5: Mapped to RxEDMA5. - 0x6: Mapped to RxEDMA6. - 0x7: Mapped to RxEDMA7. - 0x8 - 0xf: Reserved

14.6.662 CSR SW Control

This register contains software programmable controls for changing the CSR access response and status for the Ethernet Bridge Registers.

CSR_SW_Control Offset address: $1E100_H$

CSR SW Control Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0							SEEN	0							RCWE
r							rw	r							rw

Field	Bits	Type	Description
RCWE	0	rw	**Register Clear on Write 1 Enable - When this bit is set, the access mode of some Subsystem status register fields changes to Clear on Write 1** The application needs to set the respective bit to 1 to clear it. - When this bit is reset, the access mode of these register fields remain as Clear on Read. Values: - 0x0 (DISABLE): Register Clear on Write 1 Disabled - 0x1 (ENABLE): Register Clear on Write 1 Enabled
SEEN	8	rw	**Slave Error Response Enable - When this bit is set, the MAC responds with Slave Error for accesses to reserved registers in CSR space** - When this bit is reset, the MAC responds with Okay response to any register accessed from CSR space. Values: - 0x0 (DISABLE): Slave Error response Disabled - 0x1 (ENABLE): Slave Error response Enabled
0	7:1, 31:9	r	**Reserved** Read as 0; should be written with 0.

14.6.663 Interrupt Status

The application reads this Interrupt Status register during interrupt service routine or polling to determine the interrupt status of Port0 and Port1 (MTL and MAC), EDMA and the Ethernet Bridge.

Interrupt_Status Offset address: $1E104_H$

Interrupt Status Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															DMAIS
r															r

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												Port_1_MTLIS	Port_1_MACIS	Port_0_MTLIS	Port_0_MACIS
r												r	r	r	r

Field	Bits	Type	Description
Port_0_MACIS	0	r	**Port 0 MAC Interrupt Status** This bit indicates an interrupt event in the Port 0 MAC. To reset this bit to 1'b0, the software must read the corresponding register in the Port 0 MAC to get the exact cause of the interrupt and clear its source.

(table continues...)

(continued)

Field	Bits	Type	Description
Port_0_MTLIS	1	r	**Port 0 MTL Interrupt Status** This bit indicates an interrupt event in the Port 0 MTL. To reset this bit to 1'b0, the software must read the corresponding register in the Port 0 MTL to get the exact cause of the interrupt and clear its source.
Port_1_MACIS	2	r	**Port 1 MAC Interrupt Status** This bit indicates an interrupt event in the Port 1 MAC. To reset this bit to 1'b0, the software must read the corresponding register in the Port 1 MAC to get the exact cause of the interrupt and clear its source.
Port_1_MTLIS	3	r	**Port 1 MTL Interrupt Status** This bit indicates an interrupt event in the Port 1 MTL. To reset this bit to 1'b0, the software must read the corresponding register in the Port 1 MTL to get the exact cause of the interrupt and clear its source.
DMAIS	16	r	**DMA Interrupt Status** This bit indicates an interrupt event in DMA. To reset this bit to 1'b0, the software must read DMA_Interrupt_Status register to identify the DMA channel and read the corresponding status register in DMA to get the exact cause of the interrupt and clear its source.
0	15:4, 31:17	r	**Reserved Fields** Read as 0; should be written with 0.

14.6.664 Safety Interrupt Status

This register indicates summary (whether error occurred in EDMA/Ethernet Bridge/XGMAC Port 0/ XGMAC Port 1 and correctable/uncorrectable) of the Automotive Safety related error interrupts.

Safety_Interrupt_Status Offset address: $1E108_H$

Safety Interrupt Status Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0						Port_1_MCSIS	Port_1_MSUIS	Port_1_MSCIS	Port_0_MCSIS	Port_0_MSUIS	Port_0_MSCIS	0	DUIS	DCIS	0
r						r	r	r	r	r	r	r	r	r	r

Field	Bits	Type	Description
DCIS	1	r	**DMA Safety Correctable Error Interrupt Status This bit indicates an interrupt event in the DMA safety feature** To get the exact cause of the interrupt, the application should read the DMA_Safety_Interrupt_Status register. Values: - 0x0 (INACTIVE): DMA ECC Correctable error Interrupt Status not detected - 0x1 (ACTIVE): DMA ECC Correctable error Interrupt Status detected
DUIS	2	r	**DMA Safety Unrecoverable Error Interrupt Status This bit indicates an uncorrectable error interrupt event in DMA Safety feature** To get exact cause of the interrupt, the software should read the DMA_Safety_Interrupt_Status and DMA_DPP_Interrupt_Status registers. Values: - 0x0 (INACTIVE): Ethernet Bridge Safety Uncorrectable Error Interrupt Status not detected - 0x1 (ACTIVE): Ethernet Bridge Safety Uncorrectable Error Interrupt Status detected
Port_0_MSCIS	4	r	**Port 0 MTL Safety Correctable error Interrupt Status This bit indicates a correctable error interrupt event in Port 0 MTL** To get exact cause of the interrupt, the software should read the Port 0 MTL_Safety_Interrupt_Status register. Values: - 0x0 (INACTIVE): Port 0 MTL Safety Correctable error Interrupt Status not detected - 0x1 (ACTIVE): Port 0 MTL Safety Correctable error Interrupt Status detected
Port_0_MSUIS	5	r	**Port 0 MTL Safety Unrecoverable error Interrupt Status This bit indicates an uncorrectable error interrupt event in Port 0 MTL** To get exact cause of the interrupt, the software should read the Port 0 MTL_Safety_Interrupt_Status register. Values: - 0x0 (INACTIVE): Port 0 MTL Safety Uncorrectable error Interrupt Status not detected - 0x1 (ACTIVE): Port 0 MTL Safety Uncorrectable error Interrupt Status detected

(table continues...)

(continued)

Field	Bits	Type	Description
Port_0_MCSIS	6	r	**Port 0 MAC Safety Unrecoverable Interrupt Status Indicates an unrecoverable Safety related Interrupt is set in the Port 0 MAC module** Port 0 MAC_DPP_FSM_Interrupt_Status register should be read when this bit is set, to get the cause of the Safety Interrupt in Port 0 MAC. Values: - 0x0 (INACTIVE): Port 0 MAC Safety Uncorrectable Interrupt Status not detected - 0x1 (ACTIVE): Port 0 MAC Safety Uncorrectable Interrupt Status detected
Port_1_MSCIS	7	r	**Port 1 MTL Safety Correctable error Interrupt Status This bit indicates a correctable error interrupt event in Port 1 MTL** To get exact cause of the interrupt, the software should read the Port 1 MTL_Safety_Interrupt_Status register. Values: - 0x0 (INACTIVE): Port 1 MTL Safety Correctable error Interrupt Status not detected - 0x1 (ACTIVE): Port 1 MTL Safety Correctable error Interrupt Status detected
Port_1_MSUIS	8	r	**Port 1 MTL Safety Unrecoverable error Interrupt Status This bit indicates an uncorrectable error interrupt event in Port 1 MTL** To get exact cause of the interrupt, the software should read the Port 1 MTL_Safety_Interrupt_Status register. Values: - 0x0 (INACTIVE): Port 1 MTL Safety Uncorrectable error Interrupt Status not detected - 0x1 (ACTIVE): Port 1 MTL Safety Uncorrectable error Interrupt Status detected
Port_1_MCSIS	9	r	**Port 1 MAC Safety Unrecoverable Interrupt Status Indicates an unrecoverable Safety related Interrupt is set in the Port 1 MAC module** Port 1 MAC_DPP_FSM_Interrupt_Status register should be read when this bit is set, to get the cause of the Safety Interrupt in Port 1 MAC. Values: - 0x0 (INACTIVE): Port 1 MAC Safety Uncorrectable Interrupt Status not detected - 0x1 (ACTIVE): Port 1 MAC Safety Uncorrectable Interrupt Status detected

(table continues...)

(continued)

Field	Bits	Type	Description
0	0, 3, 31:10	r	**Reserved Fields** Read as 0; should be written with 0.

14.6.665 DMA Port Selection

This register provides the Port 0 or Port 1 selection for each EDMA channel for Time Based Scheduling (TBS-EST) control signals. It needs to be coherent with the Bridge core mapping register i.e., TXQ_Map_Port_1 / TXQ_Map_Port_0 / RXC_Map_Port_1 / RXC_Map_Port_0.

DMA_Port_Selection Offset address: $1E110_H$

DMA Port Selection Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								DMAPORTSEL							
r								rw							

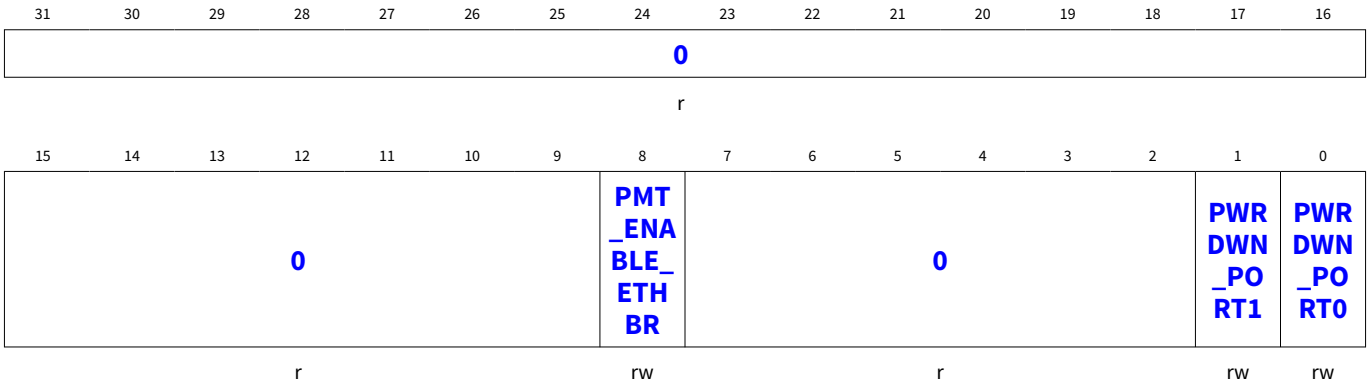
Field	Bits	Type	Description
DMAPORTSEL	7:0	rw	**DMA Channel Port Selection** One bit per DMA channel where bit[0] represents Channel 0, bit[1] represents Channel 1 and so on. Bit[0] = 0, DMA Channel-0 is allocated to Port-0. 1, DMA channel 0 is allocated to Port 1 Bit[1] = 0, DMA Channel-1 is allocated to Port-0. 1, DMA channel 1 is allocated to Port 1 - - Bit[7] = 0, DMA Channel-7 is allocated to Port-0. 1, DMA channel 7 is allocated to Port 1 Please refer to 2.2.8 DMA Port Selection.
0	31:8	r	**Reserved Fields** Read as 0; should be written with 0.

14.6.666 PMT Control

This register provides the power-up/power-down control of the Ethernet Bridge Port MTL/MAC switchable and always-ON logic when the port is not active.

PMT_Control Offset address: $1E114_H$

PMT Control Kernel Reset value: $0000\ 0100_H$



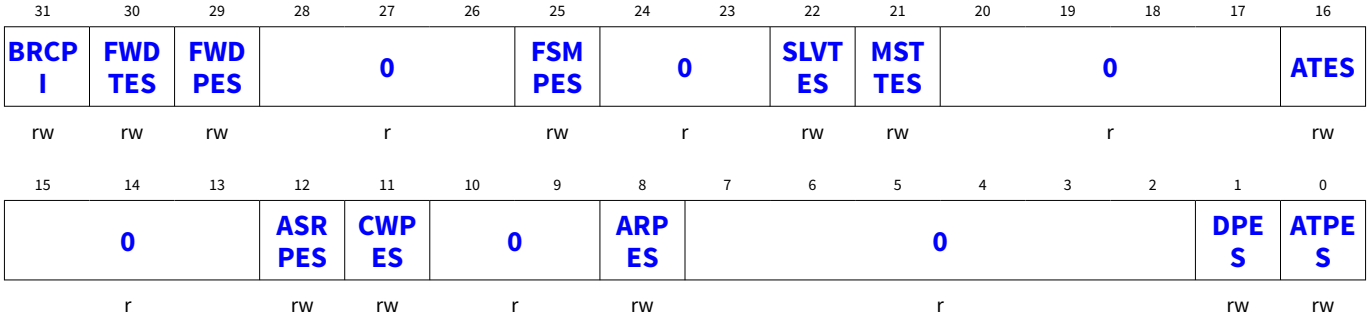
Field	Bits	Type	Description
PWRDWN_PORT0	0	rw	**Port-0 Power Down Enable When software sets this bit, both switchable and always-on logic of Port-0 is placed in the Power OFF state** When software clears this bit, both switchable and always-on logic of Port-0 is placed in the Power ON state. Values: - 0x0: Power-down Disable for Port-0 - 0x1: Power-down Enable for Port-0
PWRDWN_PORT1	1	rw	**Port 1 Power Down Enable When software sets this bit, both switchable and always-on logic of Port 1 is placed in the Power OFF state** When software clears this bit, both switchable and always-on logic of Port 1 is placed in the Power ON state. Values: - 0x0: Power-down Disable for Port 1 - 0x1: Power-down Enable for Port 1
PMT_ENABLE_ETHBR	8	rw	**PMT Enable for Ethernet Bridge switchable Power Domain** Values: - 0x0: Disable PMT for Ethernet Bridge Subsystem - 0x1: Enable PMT for Ethernet Bridge Subsystem -down Enable for Port 1
0	7:2, 31:9	r	**Reserved Fields** Read as 0; should be written with 0.

14.6.667 DPP FSM Interrupt Status

This register contains the status of Automotive Safety related Data Path Parity Errors, Interface Timeout Errors, FSM State Parity Errors, and FSM State Timeout Errors for the Ethernet Bridge, EDMA, CSR and AXI Slave/Master Interfaces.

DPP_FSM_Interrupt_Status Offset address: $1E150_H$

DPP FSM Interrupt Status Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
ATPES	0	rw	**Application Transmit Interface Parity Checker Error Status** This field, when set, indicates that a parity error is detected on the AXI Master read data parity checker (PC-e). This field, when set, indicates that a parity error is detected on the interface port parity checker. Access restriction applies. Clears on read (or write of 1 when RCWE bit in CSR_SW_Ctrl register is set). Self-set to 1 on internal event. Values: - 0x0 (INACTIVE): Application Transmit Interface Parity Checker Error Status not detected - 0x1 (ACTIVE): Application Transmit Interface Parity Checker Error Status detected
DPES	1	rw	**Descriptor Cache write data path Parity Checker Error Status This field, when set, indicates that a parity error is detected at the Descriptor Cache parity checker (or at PC-desc)** Access restriction applies. Clears on read (or write of 1 when RCWE bit in CSR_SW_Ctrl register is set). Self-set to 1 on internal event. Values: - 0x0 (INACTIVE): EDMA Descriptor Cache write data path Parity Checker Error Status not detected - 0x1 (ACTIVE): EDMA Descriptor Cache write data path Parity Checker Error Status detected
ARPES	8	rw	**Application Receive interface data path Parity Error Status This field, when set, indicates that a parity error is detected at parity checker at AXI Master Interface (PC-a)** Access restriction applies. Clears on read (or write of 1 when RCWE bit in CSR_SW_Ctrl register is set). Self-set to 1 on internal event. Values: - 0x0 (INACTIVE): Application Receive interface data path Parity Checker Error Status not detected - 0x1 (ACTIVE): Application Receive interface data path Parity Checker Error Status detected

(table continues...)

(continued)

Field	Bits	Type	Description
CWPES	11	rw	**CSR Write data path Parity Checker Error Status This field, when set, indicates that parity error is detected at the CSR write data interface on mci_wdata_i (or at PC8 checker)** This bit is set to 1 when EPSI bit of DPP_Control register is set and a parity mismatch is detected on the input slave parity ports (or at PC-axi/apb/mci slave checker). Access restriction applies. Clears on read (or write of 1 when RCWE bit in CSR_SW_Ctrl register is set). Self-set to 1 on internal event. Values: - 0x0 (INACTIVE): CSR Write data path Parity Checker Error Status not detected - 0x1 (ACTIVE): CSR Write data path Parity Checker Error Status detected
ASRPES	12	rw	**AXI Slave Read data path Parity Checker Error Status This field, when set, indicates that parity error is detected at the AXI Slave read data interface on rdata_s_o or at PC9 checker** Access restriction applies. Clears on read (or write of 1 when RCWE bit in CSR_SW_Ctrl register is set). Self-set to 1 on internal event. Values: - 0x0 (INACTIVE): AXI Slave Read data path Parity Checker Error Status not detected - 0x1 (ACTIVE): AXI Slave Read data path Parity Checker Error Status detected
ATES	16	rw	**APP FSM Timeout Error Status This field, when set, indicates that one of the APP FSM timeout has occurred** Access restriction applies. Clears on read (or write of 1 when RCWE bit in CSR_SW_Ctrl register is set). Self-set to 1 on internal event. Values: - 0x0 (INACTIVE): APP FSM Timeout Error Status not detected - 0x1 (ACTIVE): APP FSM Timeout Error Status detected
MSTTES	21	rw	**Master Read/Write Timeout Error Status This field, when set, indicates that an Application/CSR timeout has occurred on the master (AXI) interface** Access restriction applies. Clears on read (or write of 1 when RCWE bit in CSR_SW_Ctrl register is set). Self-set to 1 on internal event. Values: - 0x0 (INACTIVE): Master Read/Write Timeout Error Status not detected - 0x1 (ACTIVE): Master Read/Write Timeout Error Status detected

(table continues...)

(continued)

Field	Bits	Type	Description
SLVTES	22	rw	**Slave Read/Write Timeout Error Status This field, when set, indicates that an Application/CSR timeout has occurred on the AXI slave interface** Access restriction applies. Clears on read (or write of 1 when RCWE bit in CSR_SW_Ctrl register is set). Self-set to 1 on internal event. Values: - 0x0 (INACTIVE): Slave Read/Write Timeout Error Status not detected - 0x1 (ACTIVE): Slave Read/Write Timeout Error Status detected
FSMPES	25	rw	**FSM State Parity Error Status Indicates that parity error is detected in one of the FSM State registers** Access restriction applies. Clears on read (or write of 1 when RCWE bit in CSR_SW_Ctrl register is set). Self-set to 1 on internal event. Values: - 0x0 (INACTIVE): Slave Read/Write Timeout Error Status not detected - 0x1 (ACTIVE): Slave Read/Write Timeout Error Status detected
FWDPES	29	rw	**Ethernet Bridge Forward FSM State Parity Error Status Indicates that parity error is detected in one of the Ethernet Bridge Forward FSM State registers** Access restriction applies. Clears on read (or write of 1 when RCWE bit in CSR_SW_Ctrl register is set). Self-set to 1 on internal event. Values: - 0x0 (INACTIVE): Ethernet Bridge FSM Parity Error Status not detected - 0x1 (ACTIVE): Ethernet Bridge FSM Parity Error Status detected
FWDTES	30	rw	**Ethernet Bridge Forward FSM Timeout Error Status This field, when set, indicates that an FSM timeout has occurred on the Ethernet Bridge Forwarding Path** Access restriction applies. Clears on read (or write of 1 when RCWE bit in CSR_SW_Ctrl register is set). Self-set to 1 on internal event. Values: - 0x0 (INACTIVE): Slave Read/Write Timeout Error Status not detected - 0x1 (ACTIVE): Slave Read/Write Timeout Error Status detected
BRCPI	31	rw	**Ethernet Bridge Control Register Parity Check Error This field, when set, indicates a parity check error for Ethernet Bridge and EDMA CSR registers** Access restriction applies. Clears on read (or write of 1 when RCWE bit in CSR_SW_Ctrl register is set). Self-set to 1 on internal event.

(table continues...)

(continued)

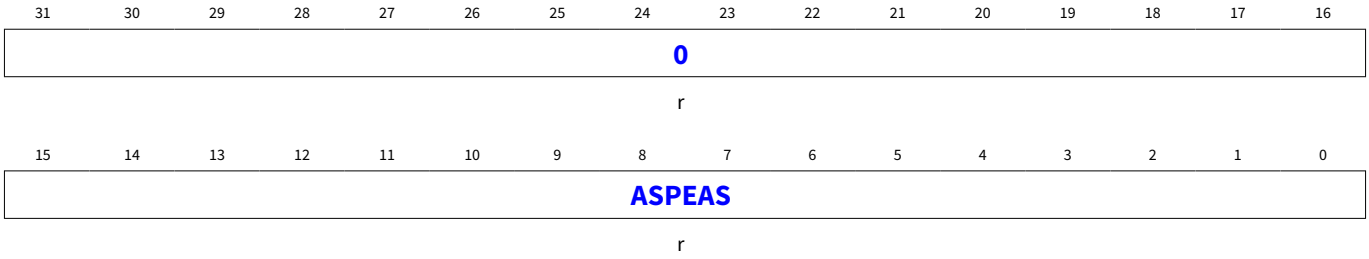
Field	Bits	Type	Description
0	7:2, 10:9, 15:13, 20:17, 24:23, 28:26	r	**Reserved Fields** Read as 0; should be written with 0.

14.6.668 AXI Slave DPP Error Address Status

This register indicates the CSR address corresponding to the CSR write data on which parity error occurred in the Ethernet Bridge and EDMA register space.

AXI_Slave_DPP_Error_Address_Status Offset address: $1E154_H$

AXI Slave DPP Error Address Status Kernel Reset value: $0000\ 0000_H$



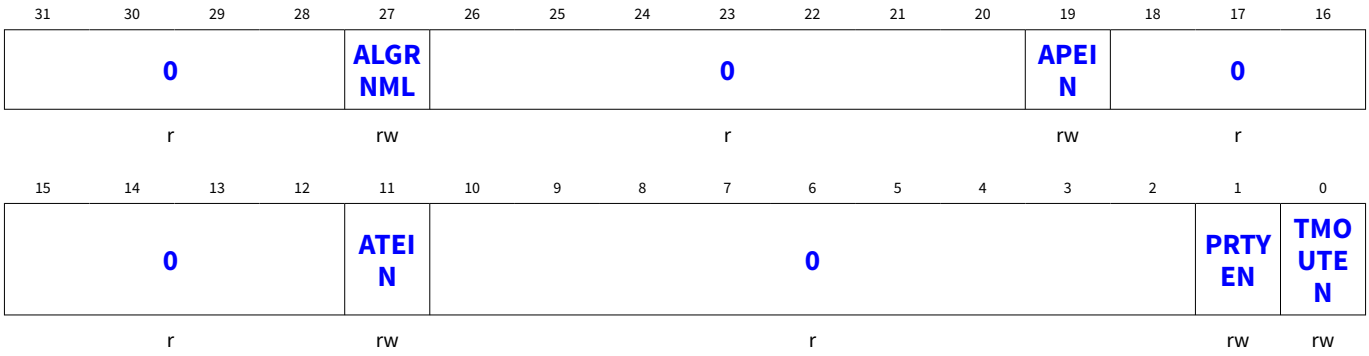
Field	Bits	Type	Description
ASPEAS	15:0	r	**AXI Slave data path Parity Error Address Status This field holds the CSR address for which parity error is detected on the CSR write data** This field holds the first address for which parity error is detected on the write data which is cleared on read.
0	31:16	r	**Reserved Fields** Read as 0; should be written with 0.

14.6.669 FSM Control

This register controls FSM Error.

FSM_Control Offset address: $1E158_H$

FSM Control Kernel Reset value: $0000\ 0000_H$



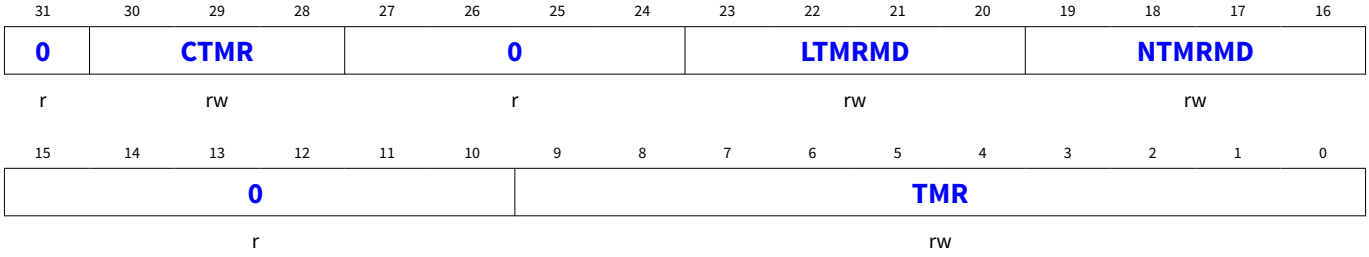
Field	Bits	Type	Description
TMOUTEN	0	rw	**This field, when set, indicates that the FSM timeout feature is enabled** Values: - 0x0 (DISABLE): FSM timeout feature is disabled - 0x1 (ENABLE): FSM timeout feature is enabled
PRTYEN	1	rw	**This field, when set, indicates that the FSM parity feature is enabled** Values: - 0x0 (DISABLE): FSM Parity feature is disabled - 0x1 (ENABLE): FSM Parity feature is enabled
ATEIN	11	rw	**APP FSM Timeout Error Injection This field, when set, indicates that Error Injection for APP FSM timeout is enabled** Values: - 0x0 (DISABLE): APP FSM Timeout Error Injection is disabled - 0x1 (ENABLE): APP FSM Timeout Error Injection is enabled
APEIN	19	rw	**APP FSM Parity Error Injection This field, when set, indicates that Error Injection for APP FSM Parity is enabled** Values: - 0x0 (DISABLE): APP FSM Parity Error Injection is disabled - 0x1 (ENABLE): APP FSM Parity Error Injection is enabled
ALGRNML	27	rw	**APP Large/Normal Mode Select This field, when set, indicates that large mode tic generation is used for APP domain; otherwise, normal mode tic generation is used** Values: - 0x0 (DISABLE): normal mode tic generation is used for APP domain - 0x1 (ENABLE): large mode tic generation is used for APP domain
0	10:2, 18:12, 26:20, 31:28	r	**Reserved Fields** Read as 0; should be written with 0.

14.6.670 FSM ACT Timer

This register is used to select the FSM and Interface Timeout values for Ethernet Bridge.

FSM_ACT_Timer Offset address: $1E15C_H$

FSM ACT Timer Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
TMR	9:0	rw	**CSR Clocks for 1us Tic This field indicates the number of CSR clocks required to generate 1us tic**
NTMRMD	19:16	rw	**Normal Mode Timeout Value This field provides the value to be used for normal mode FSM and other interface time outs** The timeout duration based on the mode value is as follows: Values: - 0x0 (DISABLE): Timer disabled - 0x1 (M_1MICRO_SEC): 1us - 0x2 (M_4MILLI_SEC): 1.024ms (~4ms) - 0x3 (M_16MILLI_SEC): 16.384ms (~16ms) - 0x4 (M_64MILLI_SEC): 65.536ms (~64ms) - 0x5 (M_256MILLI_SEC): 262.144ms (~256ms) - 0x6 (M_1SEC): 1.048sec (~1sec) - 0x7 (M_4SEC): 4.194sec (~4sec) - 0x8 (M_16SEC): 16.777sec (~16sec) - 0x9 (M_32SEC): 33.554sec (~32sec) - 0xa (M_64SEC): 67.108sec (~64sec) - 0xb (RSVD): Reserved

(table continues...)

(continued)

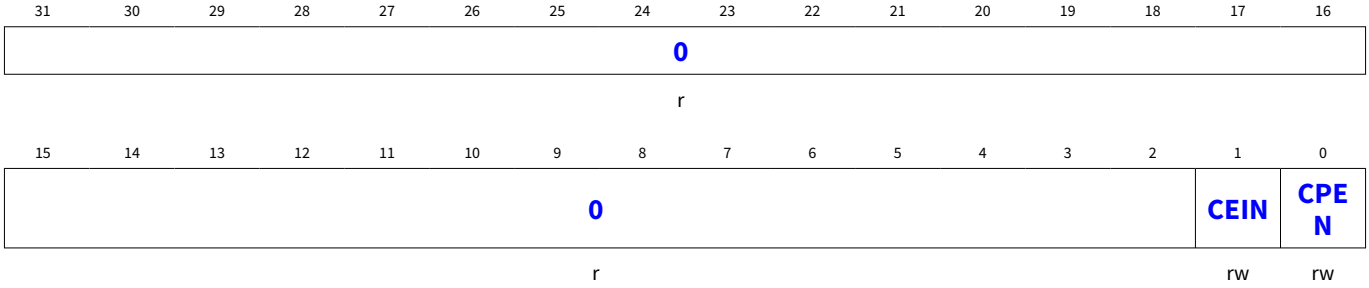
Field	Bits	Type	Description
LTMRMD	23:20	rw	**Large Mode Timeout Value This field provides the mode value to be used for large mode FSM and other interface time outs** The timeout duration based on the mode value is as follows: Values: - 0x0 (DISABLE): Timer disabled - 0x1 (M_1MICRO_SEC): 1us - 0x2 (M_4MILLI_SEC): 1.024ms (~4ms) - 0x3 (M_16MILLI_SEC): 16.384ms (~16ms) - 0x4 (M_64MILLI_SEC): 65.536ms (~64ms) - 0x5 (M_256MILLI_SEC): 262.144ms (~256ms) - 0x6 (M_1SEC): 1.048sec (~1sec) - 0x7 (M_4SEC): 4.194sec (~4sec) - 0x8 (M_16SEC): 16.777sec (~16sec) - 0x9 (M_32SEC): 33.554sec (~32sec) - 0xa (M_64SEC): 67.108sec (~64sec) - 0xb (RSVD): Reserved
CTMR	30:28	rw	**CSR Parity Check Interval Value This register is used for setting timer value for generating timer tic for CSR parity check** Values: - 0x0 (M_1MILLI_SEC): 1ms - 0x1 (M_1MICRO_SEC_CSR): 1us - 0x2 (M_4MILLI_SEC_CSR): 4.096ms (~4ms) - 0x3 (M_16MILLI_SEC_CSR): 16.384ms (~16ms) - 0x4 (M_64MILLI_SEC_CSR): 65.536ms (~64ms) - 0x5 (M_256MILLI_SEC_CSR): 262.144ms (~256ms) - 0x6 (M_1SEC_CSR): 1.048sec (~1sec) - 0x7 (ALWAYS_EN): Always Enabled
0	15:10, 27:24, 31	r	**Reserved Fields** Read as 0; should be written with 0.

14.6.671 SCSR Control

This register is used to control safety protected CSR parity generation for Ethernet Bridge and EDMA register space.

SCSR_Control Offset address: $1E164_H$

SCSR Control Kernel Reset value: $0000\ 0000_H$



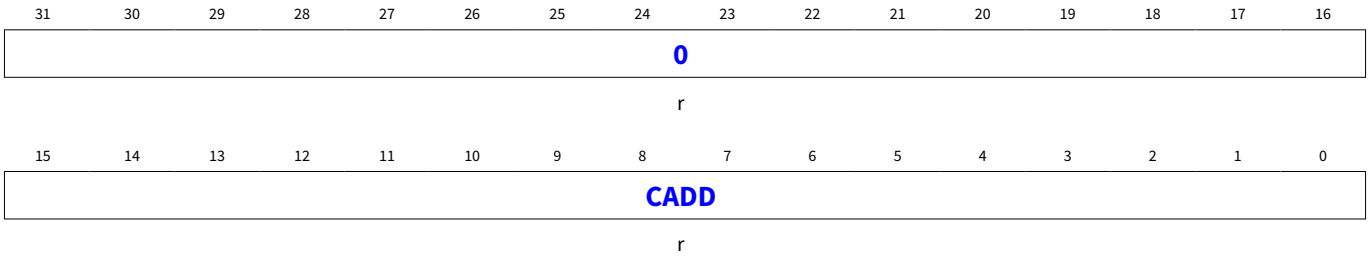
Field	Bits	Type	Description
CPEN	0	rw	**Ethernet Bridge/EDMA CSR Parity Enable This field, when set, indicates that CSR parity checking is enabled**
CEIN	1	rw	**Ethernet Bridge /EDMA CSR Error Injection Enable This field, when set, indicates that CSR error injection is enabled for the protected Ethernet Bridge and EDMA registers** Error is injected in the subsequent write operation where at the least one of the byte enable matches with byte enable indicated in CEBE. After error injection, this bit is cleared by the design.
0	31:2	r	**Reserved Fields** Read as 0; should be written with 0.

14.6.672 SCSR Parity Status

This register indicates the offset address of the register where parity_error occurred in the Ethernet Bridge/ EDMA register space.

SCSR_Parity_Status Offset address: $1E168_H$

SCSR Parity Status Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
CADD	15:0	r	**CSR Address with error This field indicates the offset address of the last register in which parity mismatch occurred** Self clears on read.
0	31:16	r	**Reserved Fields** Read as 0; should be written with 0.

14.6.673 SCSR Parity Err Count

This register indicates error count associated with partity check in the Ethernet Bridge/EDMA parity protected registers.

SCSR_Parity_Err_Count Offset address: $1E16C_H$

SCSR Parity Err Count Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0						ECLL									
r						r									

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0						ECCL									
r						r									

Field	Bits	Type	Description
ECCL	9:0	r	**Count of parity errors in current safety loop This field indicates the Count of parity errors in the current loop of all safety register parity check** Application read related errors are also included in the count. Value saturates at all "1"s and is cleared on read.
ECLL	25:16	r	**Count of parity errors in the previous safety loop This field indicates the Count of parity errors in previous loop of all safety register parity check** Any application read related errors are also included in the count. Value saturates at all "1"s and is cleared on read.
0	15:10, 31:26	r	**Reserved Fields** Read as 0; should be written with 0.

14.6.674 Debug Control

This register is used to control the ECC error injection for the Descriptor Cache memory in Debug mode.

Debug_Control Offset address: $1E208_H$

Debug Control Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0													EIEC	EIAEE	EIEE
r													rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															
r															

Field	Bits	Type	Description
EIEE	16	rw	**ECC Inject Error Enable for Descriptor Cache memories When set, enables the ECC error injection feature** If EIAEE is set, enables the ECC Address Error Injection, otherwise, enables the ECC Data/Check Bits Error Injection (See EIM and BLEI field in DPP_ECC_EIC Register). When reset, disables the ECC error injection feature.
EIAEE	17	rw	**ECC Inject Address Error Enable for Descriptor Cache memories When set, enables the ECC Address error injection feature** When reset, disables the ECC Address error injection feature. Values: - 0x0 (DISABLE): ECC Inject Address Error for Descriptor Cache memories is disabled - 0x1 (ENABLE): ECC Inject Address Error for Descriptor Cache memories is enabled
EIEC	18	rw	**ECC Inject Error Control for Descriptor Cache memories When EIEE bit of this register is set, following errors are inserted based on the value encoded in this field** Values: - 0x0 (M_1BIT): Insert 1-bit error - 0x1 (M_2BIT): Insert 2-bit errors
0	15:0, 31:19	r	**Reserved Field** Read as 0; should be written with 0.

14.6.675 ECC Control

The ECC_Control register establishes the operating mode of ECC related to Descriptor Cache memory.

ECC_Control Offset address: $1E2C0_H$

ECC Control Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0							EEAO	0		DESCED	0				
r							rw	r		rw	r				

Field	Bits	Type	Description
DESCED	5	rw	**DCACHE memory ECC Checker Disable When set to 1, disables the ECC Checker for DCACHE memory in DMA** When set to 0, enables the ECC Checker for DCACHE memory in DMA. Values: - 0x0 (ENABLE): Descriptor Cache memory ECC Checker is enabled - 0x1 (DISABLE): Descriptor Cache memory ECC Checker is disabled
EEAO	8	rw	**Descriptor Cache ECC Error Address Status Override When set, the following error address fields hold the last valid address where the error is detected** When reset, the following error address fields hold the first address where the error is detected. EUEAS/ECEAS of ECC_Err_Addr_Status register. Values: - 0x0 (DISABLE): ECC Error Address Status Over-ride is disabled - 0x1 (ENABLE): ECC Error Address Status Over-ride is enabled
0	4:0, 7:6, 31:9	r	**Reserved Fields** Read as 0; should be written with 0.

14.6.676 ECC Err Sts RCtl

The ECC_Err_Sts_Rctl register establishes the control for Descriptor Cache ECC Error status capture.

ECC_Error_Status_Capture_Control Offset address: $1E2D0_H$

ECC Err Sts RCtl Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0										CUES	CCES	EMS			EESRE
r										rw	rw	rw			rw

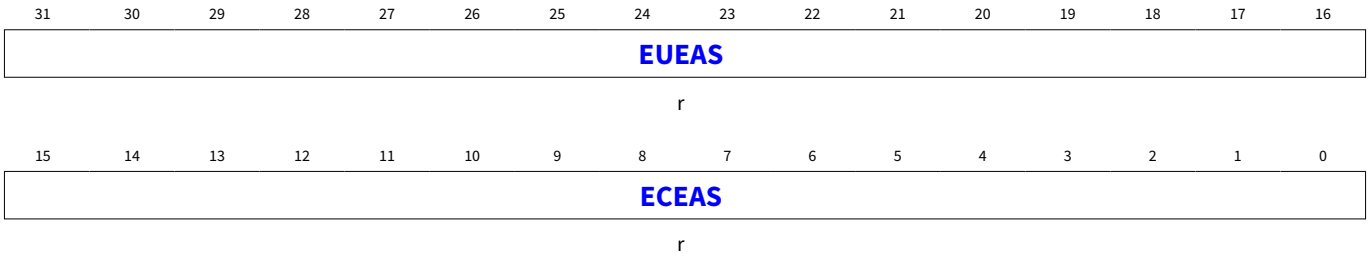
Field	Bits	Type	Description
EESRE	0	rw	**ECC Error Status Read Enable When this bit is set, based on the EMS field of this register, the Descriptor Cache memory's error status values is captured as follows: - The correctable and uncorrectable error count values are captured in the ECC_Err_Cntr_Status register** - The address locations of correctable and uncorrectable errors are captured in the ECC_Err_Addr_Status register. Hardware resets this bit when all the status values are captured in the ECC_Err_Cntr_Status and ECC_Err_Addr_Status registers. Values: - 0x0 (DISABLE): Descriptor Cache ECC Error Status Read is disabled - 0x1 (ENABLE): Descriptor Cache ECC Error Status Read is enabled
EMS	3:1	rw	**ECC Memory Selection When EESRE bit of this register is set, this field indicates which memory's error status value to be read** The memory selection encoding is as follows: Values: - 0x5 (DCACHE_MEM): DMA DCACHE memory
CCES	4	rw	**Clear Correctable Error Status When this bit is set along with EESRE bit of this register, based on the EMS field of this register, the Descriptor Cache memory's correctable error address and correctable error count values are cleared upon reading** Hardware resets this bit when all the error status values are cleared. Values: - 0x0 (INACTIVE): Clear Correctable Error Status not detected - 0x1 (ACTIVE): Clear Correctable Error Status detected
CUES	5	rw	**Clear Uncorrectable Error Status When this bit is set along with EESRE bit of this register, based on the EMS field of this register, the Descriptor Cache memory's uncorrectable error address and uncorrectable error count values are cleared upon reading** Hardware resets this bit when all the error status values are cleared. Values: - 0x0 (INACTIVE): Clear Uncorrectable Error Status not detected - 0x1 (ACTIVE): Clear Uncorrectable Error Status detected
0	31:6	r	**Reserved Fields** Read as 0; should be written with 0.

14.6.677 ECC Err Addr Status

This register provides the Descriptor Cache memory addresses for the correctable and uncorrectable errors.

ECC_Error_Address_Status — Offset address: $1E2D4_H$

ECC Err Addr Status — Kernel Reset value: $0000\ 0000_H$



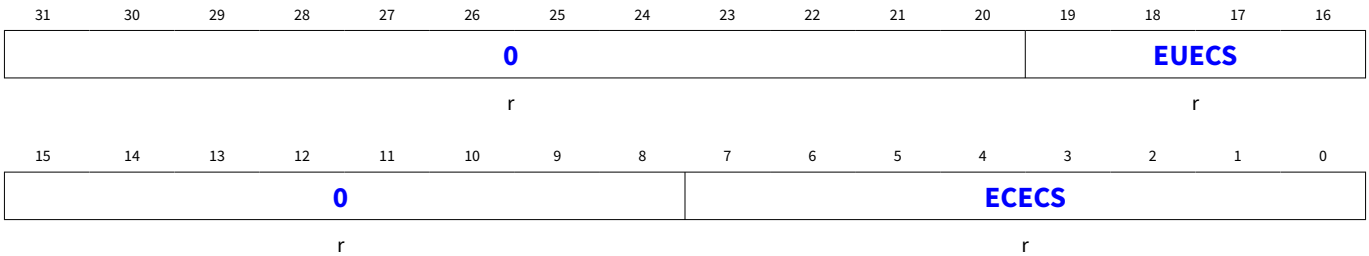
Field	Bits	Type	Description
ECEAS	15:0	r	**DCACHE Memory ECC Correctable Error Address Status Based on the EMS field of ECC_Err_Sts_Rctl register, this field holds the Descriptor Memory’s address locations for which a correctable error is detected** When EEAO bit of ECC_Control register is set, this field holds the last valid address of memory for which correctable error or address mismatch is detected. When EEAO bit of ECC_Control register is reset, this field holds the first address of the memory for which correctable error is detected.
EUEAS	31:16	r	**DCACHE Memory ECC Uncorrectable Error Address Status** Based on the EMS field of ECC_Err_Sts_Rctl register, this field holds the Descriptor Cache memory's address locations for which an uncorrectable error or address mismatch is detected. When EEAO bit of ECC_Control register is set, this field holds the last valid address of memory for which either an uncorrectable error or an address mismatch is detected. When EEAO bit of ECC_Control register is reset, this field holds the first address of the memory for which either an uncorrectable error or address mismatch is detected.

14.6.678 ECC Err Cntr Status

This register provides the Descriptor Cache memory ECC Error Count for the correctable and uncorrectable errors.

ECC_Error_Count_Status Offset address: $1E2D8_H$

ECC Err Cntr Status Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
ECECS	7:0	r	**DCACHE Memory ECC Correctable Error Counter Status Based on the EMS field of ECC_Err_Sts_Rctl register, this field holds the Descriptor Cache memory's correctable error count value**

(table continues...)

(continued)

Field	Bits	Type	Description
EUECS	19:16	r	**DCACHE Memory ECC Uncorrectable Error Counter Status Based on the EMS field of ECC_Err_Sts_Rctl register, this field holds the Descriptor Cache memory's uncorrectable error count value**
0	15:8, 31:20	r	**Reserved Fields** Read as 0; should be written with 0.

14.6.679 DPP Control

The DPP_Control establishes the operating mode of Data Parity protection and error injection for Ethernet Bridge, AXI Master and AXI Slave blocks.

DPP_Control Offset address: $1E2E0_H$

DPP Control Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															IPEASR
r															rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IPECW	0		IPEAR	0						IPEDC	IPEAT	0	ESPI	OPE	DDPP
rw	r		rw	r						rw	rw	r	rw	rw	rw

Field	Bits	Type	Description
DDPP	0	rw	**Disable Data path Parity Protection for Ethernet Bridge When set to 0, enables the parity protection for XGMAC datapath by generating and checking the parity on XGMAC datapath** When set to 1, disables the parity protection for XGMAC datapath. Values: - 0x0 (DISABLE): Data path Parity Protection is enabled - 0x1 (ENABLE): Data path Parity Protection is disabled
OPE	1	rw	**Odd Parity Enable for Ethernet Bridge When set to 1, enables odd parity protection on all the external interfaces** When set to 0, enables even parity protection on all the external interfaces. OPE default value (up on reset) is based on the DWCXG_ASP_PPE_ODD parameter, which checks if external interface parity should be odd or even). Values: - 0x0 (DISABLE): Odd Parity is disabled - 0x1 (ENABLE): Odd Parity is enabled

(table continues...)

(continued)

Field	Bits	Type	Description
ESPI	2	rw	**Enable Parity on Slave Interface port When set to 1, enables the parity check for the slave interface ports and disables the internal parity generation for the input slave data port (axi/apb/mci slave port)** When set to 0, disables the parity check for the slave interface ports and enables the internal parity generation for the input slave data port. Values: - 0x0 (DISABLE): Parity on Slave Interface port is disabled - 0x1 (ENABLE): Parity on Slave Interface port is enabled
IPEAT	4	rw	**Insert Parity Error in Application Transmit Interface Parity checker When set to 1, parity/data bit of first valid input parity/data of the Interface data parity checker is flipped** Based on the EIM/BLEI field of DPP_ECC_EIC register, software can corrupt any one bit of parity/data at Parity Checker (PC-e) at AXI Master interface. Hardware clears this bit after the respective parity bit is flipped. Values: - 0x0 (DISABLE): Insert Parity Error in Application Transmit Interface Parity checker is disabled - 0x1 (ENABLE): Insert Parity Error in Application Transmit Interface Parity checker is enabled
IPEDC	5	rw	**Insert Parity Error in Descriptor Cache data path Parity checker When set to 1, parity/data bit of first valid input parity/data of the Descriptor Cache parity checker (or at PC-desc) is flipped** Based on the EIM/BLEI field of DPP_ECC_EIC register, software can corrupt any one bit of parity/data. Hardware clears this bit after the respective parity bit is flipped. Values: - 0x0 (DISABLE): Insert Parity Error in EDMA Descriptor Cache (use different memory with TSO or TSO feature is disabled) write data path Parity checker is disabled - 0x1 (ENABLE): Insert Parity Error in EDMA Descriptor Cache (use different memory with TSO or TSO feature is disabled) write data path Parity Checker is enabled.
IPEAR	12	rw	**Insert Parity error in Insert Parity Error in Application Receive interface data path Parity Checker When set to 1, parity/data bit of first valid input parity/data of Application Receive Interface Parity Checker (PC-a) at AXI Master interface is flipped** Based on the EIM/BLEI field of DPP_ECC_EIC register, software can corrupt any one bit of parity/data. Hardware clears this bit after the respective parity bit is flipped. Values: - 0x0 (DISABLE): Insert Parity error in Application Receive interface data path Parity checker is disabled - 0x1 (ENABLE): Insert Parity error in Application Receive interface data path Parity checker is enabled

(table continues...)

(continued)

Field	Bits	Type	Description
IPECW	15	rw	**Insert Parity Error in CSR Write data path Parity Checker When set to 1, parity/data bit of first valid input parity/data of the CSR write data interface is flipped** When EPSI bit of DPP_Control register is set and this bit is set, parity/data bit of first valid input parity/data of the input slave parity ports (or at PC-axi) is flipped. Based on the EIM/BLEI field of DPP_ECC_EIC register, software can corrupt any one bit of parity/data. Hardware clears this bit after the respective parity bit is flipped. Values: - 0x0 (DISABLE): Insert Parity error in CSR Write data path Parity checker is disabled - 0x1 (ENABLE): Insert Parity error in CSR Write data path Parity checker is enabled
IPEASR	16	rw	**Insert Parity Error in AXI Slave Read data path Parity Checker When set to 1, parity/data bit of first valid input parity/data of the AXI Slave read data interface is flipped** Based on the EIM/BLEI field of DPP_ECC_EIC register, software can corrupt any one bit of parity/data. Hardware clears this bit after the respective parity bit is flipped. Values: - 0x0 (DISABLE): Insert Parity error in AXI Slave Read data path Parity checker is disabled - 0x1 (ENABLE): Insert Parity error in AXI Slave Read data path Parity checker is enabled
0	3, 11:6, 14:13, 31:17	r	**Reserved Field** Read as 0; should be written with 0.

14.6.680 DPP ECC EIC

The DPP_ECC_EIC establishes the operating mode of ETHBR ECC/DPP error injection.

DPP_ECC_Error_Injection_Control Offset address: $1E2E4_H$

DPP ECC EIC Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															EIM
r															rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								BLEI							
r								rw							

Field	Bits	Type	Description
BLEI	7:0	rw	**Bit Location of error injection This field indicates the bit location of DPP/ECC error injection, determination of error in Parity/ECC bits or Data (being protected) depends on the Error Injection Mode (EIM field)** Depending on the interface being used for error injection not all bits of this field are valid. Example, for error injection on a 64-bit data interface this field should be programmed to a value between 63 and 0. In case of 2-bit error injection bit 0 is always included in error injection and this field should represent the second bit selection for error injection. If the second bit is programed at bit 0 when 2-bit error injection is enabled, DUT only insert 1-bit error at bit 0.
EIM	16	rw	**Error Injection Mode When it set to 0, indicate error injection on data; When it set to 1, indicate error injection on ECC/Parity bits (need to check the address error injection mode is disabled)**
0	15:8, 31:17	r	**Reserved Fields** Read as 0; should be written with 0.

14.6.681 DMA Mode

The Bus Mode register establishes the bus operating modes for the DMA.

DMA_Mode Offset address: $1F000_H$

DMA Mode Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		INTM		0			DSPW	0		TMRP	TDRP	0			SWR
r		rw		r			rw	r		rw	rw	r			rw

Field	Bits	Type	Description
SWR	0	rw	**Software Reset** When this bit is set, the MAC and the DMA controller reset the logic and all internal registers of the DMA, MTL, and MAC. This bit is automatically cleared after the reset operation is complete in all active clock domains. Before reprogramming any DWC_xgmac register, read a value of 0 from this bit. Note: The reset operation is complete only when all resets in all active clock domains are de-asserted. Therefore, it is essential that all PHY inputs clocks (applicable for the selected PHY interface) are present for software reset completion. The time to complete the software reset operation depends on the frequency of the slowest active clock.

(table continues...)

(continued)

Field	Bits	Type	Description
TDRP	4	rw	**Tx Descriptor Read Priority** This bit controls the arbitration priority of descriptor fetches from the local cache when simultaneous requests are generated from RxDMA and TxDMA channels. By default, when this bit is 0, the RxDMA requests get higher priority. When this bit is set to 1, TxDMA descriptor fetches are serviced first by the arbiter before the RxDMA requests.
TMRP	5	rw	**Reserved**
DSPW	8	rw	**Descriptor Posted Write** - 0: By default, all Rx DMA descriptor writes are non-posted and any Rx interrupt generation mechanism can be used, that is, Descriptor IOC, Watchdog timer, Byte Counter based or any combination of these. - 1: All Rx DMA data and descriptor writes are posted and Descriptor IOC based Rx interrupt generation does not function properly and must not be used. Only Watchdog timer, Byte Counter based or any combination of these Rx interrupt generation mechanisms must be used. The Tx/Rx DMA Descriptor IOC based Tx/Rx interrupt generation mechanism does not function properly and must not be used.
INTM	13:12	rw	**Interrupt Mode** This field controls the behavior of the interrupt signal (sbd_intr_o, sbd_perch_tx_intr_o, sbd_perch_rx_intr_o) outputs for packet transfer completion events in TxDMA and RxDMA. - 00: sbd_perch_* are pulse signals for each completion events. sbd_intr_o is also asserted and cleared only when software clears the corresponding RI/TI status bits. - 01: sbd_perch_* are level signals asserted on the corresponding event and de-asserted when the software clears the corresponding RI/TI status bits. The sbd_intr_o is not asserted for these packet transfer completion events. - 10: sbd_perch_* are level signals asserted on the corresponding event and de-asserted when the software clears the corresponding RI/TI status bits. However, the signal is asserted again if the same event occurred again before it was cleared. The sbd_intr_o is not asserted for these packet transfer completion events. - 11: Reserved For more details see Table, "DWC_xgmac Transfer Complete Interrupt Behavior"
0	3:1, 7:6, 11:9, 31:14	r	**Reserved** Read as 0; should be written with 0.

14.6.682 DMA SysBus Mode

The System Bus mode register controls the behavior of the AXI master. It mainly controls burst splitting and number of outstanding requests.

DMA_SysBus_Mode Offset address: $1F004_H$

DMA SysBus Mode Kernel Reset value: $0101\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0			WR_OSR_LMT					0			RD_OSR_LMT				
r			rw					r			rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN_LPI	LPI_XIT_PKT	ONEKBBE	AAL	0	AALE	0		BLEN256	BLEN128	BLEN64	BLEN32	BLEN16	BLEN8	BLEN4	UBL
rw	rw	rw	rw	r	rw	r		rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
UBL	0	rw	**AXI Undefined Burst Length** - 1: The AXI master can perform burst transfers that are equal to or less than the maximum allowed burst length programmed in Bits[7:4]. - 0: The AXI master performs one of the following burst transfers: -- Burst transfers of fixed burst lengths as indicated by the BLEN256, BLEN128, BLEN64, BLEN32, BLEN16, BLEN8, or BLEN4 field -- Burst transfers of length 1 When this bit is set and no BLEN bit is set, the AXI master performs bursts transfers of any length upto 16.
BLEN4	1	rw	**AXI Burst Length 4** When this bit and the UBL bit is set to 0, the AXI master can select a burst length of 4 on the AXI interface. When the UBL bit is set to 1, setting this bit has no effect.
BLEN8	2	rw	**AXI Burst Length 8** When this bit and the UBL bit is set to 0, the AXI master can select a burst length of 8 on the AXI interface. When the UBL bit is set to 1, setting this bit has no effect.
BLEN16	3	rw	**AXI Burst Length 16** When this bit and the UBL bit is set to 0, the AXI master can select a burst length of 16 on the AXI interface. When UBL is set to 1, this bit has no effect as the AXI master can always select a burst length up to 16 on the AXI interface.
BLEN32	4	rw	**AXI Burst Length 32** When this bit is set to 1, the AXI master can select a burst length of 32 on the AXI interface.
BLEN64	5	rw	**AXI Burst Length 64** When this bit is set to 1, the AXI master can select a burst length of 64 on the AXI interface.
BLEN128	6	rw	**AXI Burst Length 128** When this bit is set to 1, the AXI master can select a burst length of 128 on the AXI interface.
BLEN256	7	rw	**AXI Burst Length 256** When this bit is set to 1, the AXI master can select a burst length of 256 on the AXI interface.

(table continues...)

(continued)

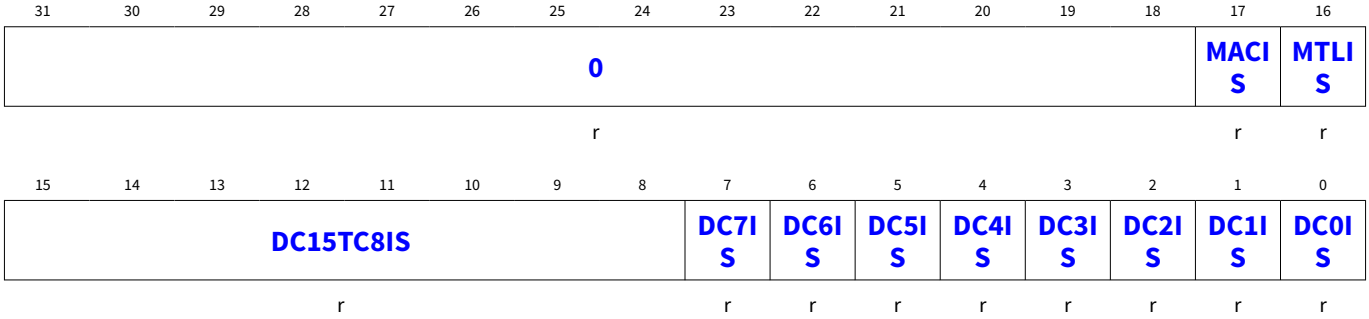
Field	Bits	Type	Description
AALE	10	rw	**Automatic AXI LPI enable** - 1: Enables the AXI master to enter into LPI state when there is no activity in DWC_xgmac for the number of system clock cycles programmed in the LPIEI field of AXI_LPI_Entry_Interval register.
AAL	12	rw	**Address-Aligned Beats** When this bit is set to 1, the AXI master performs address-aligned burst transfers on Read and Write channels.
ONEKBBE	13	rw	**1 KB Boundary Crossing Enable for the AXI Master** - 1: The burst transfers performed by the AXI master do not cross 1 KB boundary. - 0: The burst transfers performed by the AXI master do not cross 4 KB boundary.
LPI_XIT_PKT	14	rw	**Unlock on Magic Packet or Remote Wake-Up Packet** - 1: Enables the AXI master to come out of the LPI mode only when the magic packet or remote wake-up packet is received. - 0: Enables the AXI master to come out of the LPI mode when any packet is received.
EN_LPI	15	rw	**Enable Low Power Interface (LPI)** - 1: Enables the LPI mode supported by the AXI interface and accepts the LPI request from the AXI System Clock controller. - 0: Disables the LPI mode and always denies the LPI request from the AXI System Clock controller.
RD_OSR_LMT	20:16	rw	**AXI Maximum Read Outstanding Request Limit** This value limits the maximum outstanding request on the AXI read interface. Maximum outstanding requests = RD_OSR_LMT + 1
WR_OSR_LMT	28:24	rw	**AXI Maximum Write Outstanding Request Limit** This value limits the maximum outstanding request on the AXI write interface. Maximum outstanding requests = WR_OSR_LMT + 1
0	9:8, 11, 23:21, 31:29	r	**Reserved** Read as 0; should be written with 0.

14.6.683 DMA Interrupt Status

The application reads this Interrupt Status register during interrupt service routine or polling to determine the interrupt status of DMA channels, MTL queues, and the MAC.

DMA_Interrupt_Status Offset address: $1F008_H$

DMA Interrupt Status Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
DC0IS	0	r	**DMA Channel 0 Interrupt Status** This bit indicates an interrupt event in DMA Channel 0. To reset this bit to 1'b0, the software must read the corresponding register in DMA Channel 0 to get the exact cause of the interrupt and clear its source.
DC1IS	1	r	**DMA Channel 1 Interrupt Status** This bit indicates an interrupt event in DMA Channel 1. To reset this bit to 1'b0, the software must read the corresponding register in DMA Channel 1 to get the exact cause of the interrupt and clear its source.
DC2IS	2	r	**DMA Channel 2 Interrupt Status** This bit indicates an interrupt event in DMA Channel 2. To reset this bit to 1'b0, the software must read the corresponding register in DMA Channel 2 to get the exact cause of the interrupt and clear its source.
DC3IS	3	r	**DMA Channel 3 Interrupt Status** This bit indicates an interrupt event in DMA Channel 3. To reset this bit to 1'b0, the software must read the corresponding register in DMA Channel 3 to get the exact cause of the interrupt and clear its source.
DC4IS	4	r	**DMA Channel 4 Interrupt Status** This bit indicates an interrupt event in DMA Channel 4. To reset this bit to 1'b0, the software must read the corresponding register in DMA Channel 4 to get the exact cause of the interrupt and clear its source.
DC5IS	5	r	**DMA Channel 5 Interrupt Status** This bit indicates an interrupt event in DMA Channel 5. To reset this bit to 1'b0, the software must read the corresponding register in DMA Channel 5 to get the exact cause of the interrupt and clear its source.
DC6IS	6	r	**DMA Channel 6 Interrupt Status** This bit indicates an interrupt event in DMA Channel 6. To reset this bit to 1'b0, the software must read the corresponding register in DMA Channel 6 to get the exact cause of the interrupt and clear its source.
DC7IS	7	r	**DMA Channel 7 Interrupt Status** This bit indicates an interrupt event in DMA Channel 7. To reset this bit to 1'b0, the software must read the corresponding register in DMA Channel 7 to get the exact cause of the interrupt and clear its source.
DC15TC8IS	15:8	r	**DMA Channel 15 Interrupt Status to DMA Channel 8 Interrupt Status** To reset a bit to 1'b0, the software must read the Channel Status register in the corresponding DMA Channel (15 to 8) to get the exact cause of the interrupt and clear its source.

(table continues...)

(continued)

Field	Bits	Type	Description
MTLIS	16	r	**MTL Interrupt Status** This bit indicates an interrupt event in the MTL. To reset this bit to 1'b0, the software must read the corresponding register in the MTL to get the exact cause of the interrupt and clear its source.
MACIS	17	r	**MAC Interrupt Status** This bit indicates an interrupt event in the MAC. To reset this bit to 1'b0, the software must read the corresponding register in the MAC to get the exact cause of the interrupt and clear its source.
0	31:18	r	**Reserved** Read as 0; should be written with 0.

14.6.684 AXI Tx AR ACE Control

This register is used to control the AXI4 Cache Coherency Signals for read transactions by all the Transmit DMA channels. The following signals of the AXI4 interface are driven with different values as programmed for corresponding type (descriptor, buffer1, buffer2) of access.

- AXI Master read control channel's cache type
- AXI Master read control channel's Domain indicator signal

AXI_Tx_AR_ACE_Control Offset address: $1F010_H$

AXI Tx AR ACE Control Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0										THD		THC			
r										rw		rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		TED		TEC				0		TDRD		TDRC			
r		rw		rw				r		rw		rw			

Field	Bits	Type	Description
TDRC	3:0	rw	**Transmit DMA Read Descriptor Cache Control** This field is used to drive arcache_o[3:0] signal when Transmit DMA engines access the Descriptor.
TDRD	5:4	rw	**Transmit DMA Read Descriptor Domain Control** This field is used to drive ardomain_o[1:0] signal when Transmit DMA engines access the Descriptor.
TEC	11:8	rw	**Transmit DMA Extended Packet Buffer** This field is used to drive arcache o[3:0] signal when Transmit DMA is accessing the extended buffers (when packet is distributed across multiple buffers).
TED	13:12	rw	**Transmit DMA Extended Packet Buffer** This field is used to drive ardomain_o[1:0] signal when Transmit DMA is accessing the extended buffers (when packet is distributed across multiple buffers).

(table continues...)

(continued)

Field	Bits	Type	Description
THC	19:16	rw	**Transmit DMA First Packet Buffer** This field is used to drive arcache_o[3:0] signal when Transmit DMA is accessing First Buffer of the Packet (First valid buffer with FD being set in the TDES3 of the Descriptor.)
THD	21:20	rw	**Transmit DMA First Packet Buffer** This field is used to drive ardomain_o[1:0] signal when Transmit DMA is accessing First Buffer of the Packet (First valid buffer with FD being set in the TDES3 of the Descriptor).
0	7:6, 15:14, 31:22	r	**Reserved** Read as 0; should be written with 0.

14.6.685 AXI Rx AW ACE Control

This register is used to control the Cache Coherency Signals of the AXI4-ACE Lite on the Receive DMA. The programmable controllability is supported on the following signals of the AXI Write channel.

- AXI Master write control channel's cache type
- AXI Master write control channel's Domain indicator signal

AXI_Rx_AW_ACE_Control Offset address: $1F018_H$

AXI Rx AW ACE Control Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		RDD		RDC				0		RHD		RHC			
r		rw		rw				r		rw		rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		RPD		RPC				0		RDWD		RDWC			
r		rw		rw				r		rw		rw			

Field	Bits	Type	Description
RDWC	3:0	rw	**Receive DMA Write Descriptor Cache Control** This field is used to drive awcache_o[3:0] signal when Receive DMA access the Descriptor.
RDWD	5:4	rw	**Receive DMA Write Descriptor Domain Control** This field is used to drive awdomain_o[1:0] signal when Receive DMA access the Descriptor.
RPC	11:8	rw	**Receive DMA Payload Cache Control** This field is used to drive awcache_o[3:0] signal when Receive DMA is accessing the Payload Buffer when Header and payload are separated.
RPD	13:12	rw	**Receive DMA Payload Domain Control** This field is used to drive awdomain_o[1:0] signal when Receive DMA is accessing the Payload Buffer when Header and payload are separated.

(table continues...)

(continued)

Field	Bits	Type	Description
RHC	19:16	rw	**Receive DMA Header Cache Control** This field is used to drive awcache_o[3:0] and signal when Receive DMA is accessing the header Buffer when Header and payload are separated.
RHD	21:20	rw	**Receive DMA Header Domain Control** This field is used to drive awdomain_o[1:0] and signal when Receive DMA is accessing the header Buffer when Header and payload are separated.
RDC	27:24	rw	**Receive DMA Buffer Cache Control** This field is used to drive awcache_o[3:0] signal when Receive DMA is accessing the Buffer when Header and payload are not separated.
RDD	29:28	rw	**Receive DMA Buffer Domain Control** This field is used to drive the awdomain_o[1:0] signal when Receive DMA is accessing the Buffer when Header and payload are not separated.
0	7:6, 15:14, 23:22, 31:30	r	**Reserved** Read as 0; should be written with 0.

14.6.686 AXI TxRx AWAR ACE Control

This register is used to control the AXI4 Cache Coherency Signals for Descriptor write transactions by all the TxDMA channels and Descriptor read transactions by all the RxDMA channels. It also controls the values to be driven on AXI Master write control channel's protection control signal and AXI Master read control channel's protection control signal .

AXI_TxRx_AWAR_ACE_Control Offset address: $1F01C_H$

AXI TxRx AWAR ACE Control Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0									WRP			0	RDP		
r									rw			r	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		RDRD		RDRC				0		TDWD		TDWC			
r		rw		rw				r		rw		rw			

Field	Bits	Type	Description
TDWC	3:0	rw	**Transmit DMA Write Descriptor Cache control** This field is used to drive awcache_o[3:0] signal when Transmit DMA is writing the Descriptor.
TDWD	5:4	rw	**Transmit DMA Write Descriptor Domain control** This field is used to drive awdomain_o[1:0] signal when Transmit DMA is writing the Descriptor.

(table continues...)

(continued)

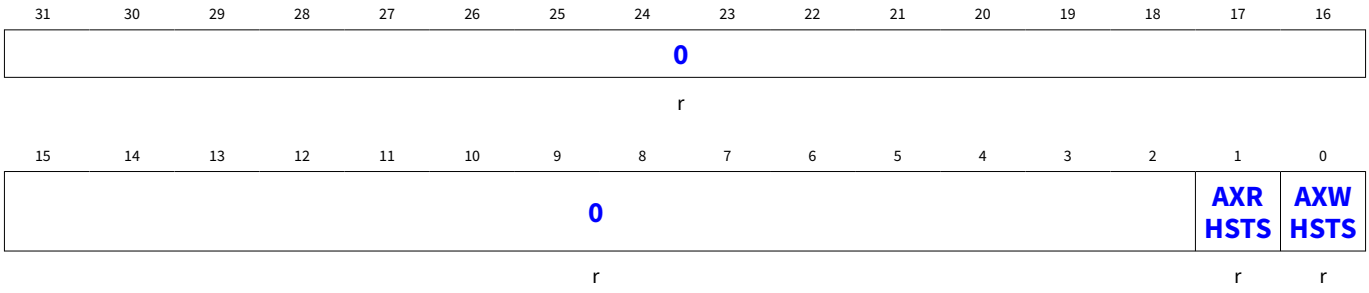
Field	Bits	Type	Description
RDRC	11:8	rw	**Receive DMA Read Descriptor Cache control** This field is used to drive arcache_o[3:0] signal when Receive DMA engine is reading the Descriptor.
RDRD	13:12	rw	**Receive DMA Read Descriptor Domain control** This field is used to drive ardomain_o[1:0] signal when Receive DMA engine is reading the Descriptor.
RDP	18:16	rw	**DMA Read Protection control** This field is used to drive the AXI Master read control channel's protection control signal during all read requests.
WRP	22:20	rw	**DMA Write Protection control** This field is used to drive the AXI Master write control channel's protection control signal during all write Requests.
0	7:6, 15:14, 19, 31:23	r	**Reserved** Read as 0; should be written with 0.

14.6.687 DMA Debug Status0

The Debug Status 0 register gives the Receive and Transmit process status for DMA Channel 0 - Channel 2 for debugging purpose.

DMA_Debug_Status0 Offset address: $1F020_H$

DMA Debug Status0 Kernel Reset value: $0000\ 0000_H$



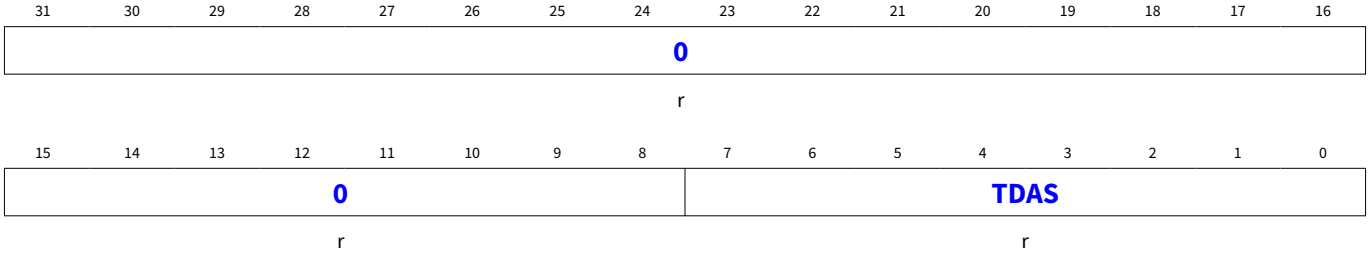
Field	Bits	Type	Description
AXWHSTS	0	r	**AXI Master Write Channel Status** - 1: Indicates that the write channel of the AXI master is active, and it is transferring data.
AXRHSTS	1	r	**AXI Master Read Channel Status** When high, this bit indicates that the read channel of the AXI master is active, and it is transferring the data.
0	31:2	r	**Reserved** Read as 0; should be written with 0.

14.6.688 DMA Debug Status1

The Debug Status1 register gives the per Transmit DMA active status.

DMA_Debug_Status1 Offset address: $1F024_H$

DMA Debug Status1 Kernel Reset value: $0000\ 0000_H$



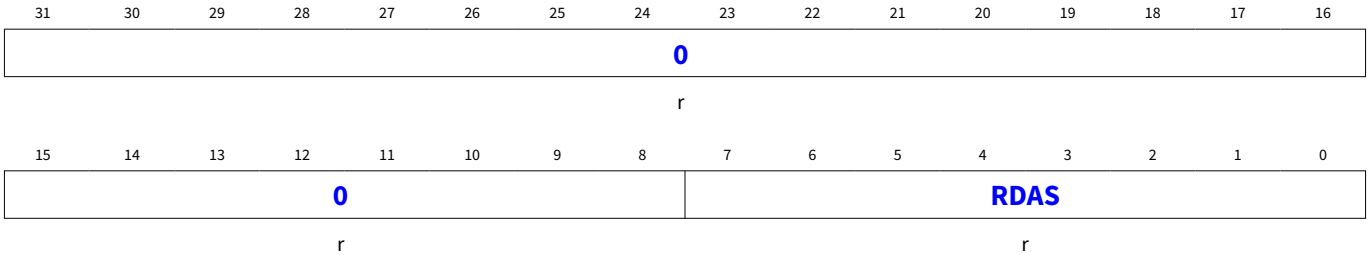
Field	Bits	Type	Description
TDAS	7:0	r	**Transmit DMA Active Status** - 1: The corresponding Transmit DMA FSMs are not idle, that is, the FSMs are actively processing the descriptors or packet data. - 0: The corresponding Transmit DMA FSMs are in idle state.
0	31:8	r	**Reserved** Read as 0; should be written with 0.

14.6.689 DMA Debug Status3

The Debug Status3 register gives the per Receive DMA active status.

DMA_Debug_Status3 Offset address: $1F02C_H$

DMA Debug Status3 Kernel Reset value: $0000\ 0000_H$



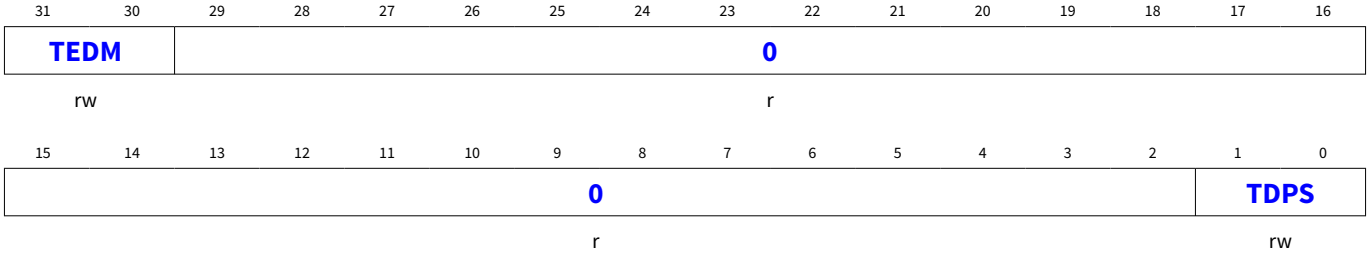
Field	Bits	Type	Description
RDAS	7:0	r	**Receive DMA Active Status** When the bit is set to 1, the corresponding Receive DMA FSMs are not idle, that is, actively processing the descriptors or packet data. When the bit is set to 0, the corresponding Receive DMA FSMs are in idle state.
0	31:8	r	**Reserved** Read as 0; should be written with 0.

14.6.690 DMA Tx EDMA Control

The DMA_Tx_EDMA_Control register controls the sequence of all TxDMA operations with the DMA engine.

DMA_Tx_EDMA_Control Offset address: $1F040_H$

DMA Tx EDMA Control Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
TDPS	1:0	rw	**Tx Descriptor Pre-fetch threshold Size** This field controls the threshold in the Descriptor cache after which the DMA starts pre-fetching the TxDMA descriptors. The DMA engine for all TxDMA channels initiate requests for the descriptor fetches as soon as the number of descriptors in the cache memory for that DMA channel, falls below or equal to the programmed threshold (each descriptor is 16 bytes). - 3'b000: Threshold of 0 descriptors. Descriptor pre-fetch is triggered only when cache is empty. - 3'b001: Threshold is 4 descriptors. Descriptor pre-fetch is triggered when 4 descriptors are left in the cache. - 3'b010: Threshold is 8 descriptors. Descriptor pre-fetch is triggered when 8 descriptors are left in the cache. - 3'b011: Threshold is 12 descriptors. Descriptor pre-fetch is triggered when 12 descriptors are left in the cache. - 3'b100: Threshold is 16 descriptors. Descriptor pre-fetch is triggered when 16 descriptors are left in the cache. - 3'b101: Threshold is 24 descriptors. Descriptor pre-fetch is triggered when 24 descriptors are left in the cache. - 3'b110 - 3'b111: Reserved. Note: Bit 2 is reserved and RO, that is TDPS value of 100 and 101 is not valid.

(table continues...)

(continued)

Field	Bits	Type	Description
TEDM	31:30	rw	**Tx DMA Mode** This field controls the sequence of descriptor and data transfers in the DMA engine for all TxDMA channels is as follows: - 2'b00: Default mode of DMA, in which all transfers are independently triggered and can occur in parallel. This gives you the best performance. Use the other modes only for debug purposes. - 2'b01: The value of TDPS field is ignored and only one Descriptor is fetched at a time from the System Memory. The next descriptor is fetched only after the current descriptor is accepted by Data Transfer Engine. The Data transfer, Descriptor close and the next descriptor fetch can still occur in parallel. - 2'b10: The value of TDPS field is ignored and only one Descriptor is fetched at a time from the System Memory. The next descriptor is fetched only after the data transfer corresponding to current descriptor is completed. The Descriptor close and the next descriptor fetch can still happen in parallel. - 2'b11: All operations are sequential. Descriptor is fetched, data transfer is completed, descriptor is closed and then only the next descriptor is fetched. The value of TDPS is ignored in this mode.
0	29:2	r	**Reserved** Read as 0; should be written with 0.

14.6.691 DMA Rx EDMA Control

The DMA_Rx_EDMA_Control register controls the sequence of all RxDMA operations with the DMA engine.

DMA_Rx_EDMA_Control Offset address: $1F044_H$

DMA Rx EDMA Control Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
REDM		0													
rw		r													

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0														RDPS	
r														rw	

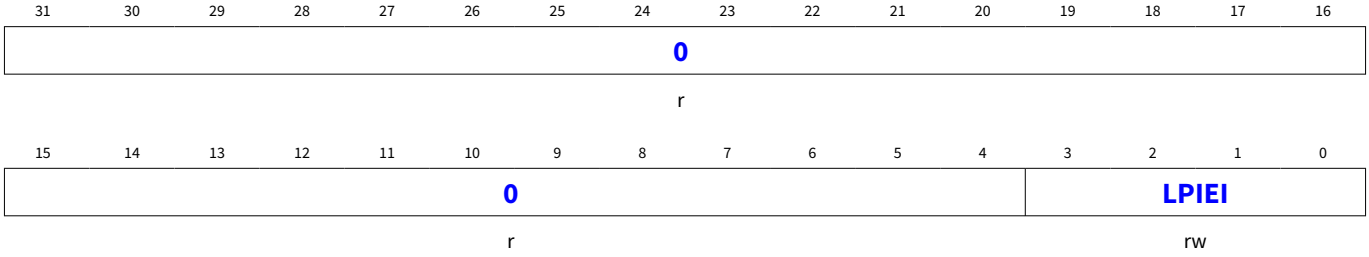
Field	Bits	Type	Description
RDPS	1:0	rw	**Rx Descriptor Pre-Fetch Threshold Size** This field controls the threshold in the Descriptor cache after which the DMA starts pre-fetching the RxDMA descriptors. The DMA engine for all RxDMA channels initiate requests for the descriptor fetches as soon as the number of descriptors in the cache memory for that DMA channel, falls lower than or equal to the programmed threshold (each descriptor is 16 bytes). - 3'b000: Threshold of 0 descriptors. Descriptor pre-fetch is triggered only when cache is empty. - 3'b001: Threshold is 4 descriptors. Descriptor pre-fetch is triggered when 4 descriptors are left in the cache. - 3'b010: Threshold is 8 descriptors. Descriptor pre-fetch is triggered when 8 descriptors are left in the cache. - 3'b011: Threshold is 12 descriptors. Descriptor pre-fetch is triggered when 12 descriptors are left in the cache. - 3'b100: Threshold is 16 descriptors. Descriptor pre-fetch is triggered when 16 descriptors are left in the cache. - 3'b101: Threshold is 24 descriptors. Descriptor pre-fetch is triggered when 24 descriptors are left in the cache. - 3'b110-3'b111: Reserved. Note: Bit 2 is reserved and RO , that is RDPS value of 100 and 101 is not valid.
REDM	31:30	rw	**Rx DMA Mode** This field controls the sequence of descriptor and data transfers in the DMA engine for all RxDMA channels: - 2'b00: Default mode of DMA as explained in section 2.4.8 in which all transfers are independently triggered and can occur in parallel. This gives you the best performance. Use the other modes only for debug purposes. - 2'b01: The value of RDPS field is ignored and only 1 Descriptor is fetched at a time from the System Memory. The next descriptor is fetched only after the current descriptor is accepted by Data Transfer Engine. The Data transfer, Descriptor close and the next descriptor fetch can still happen in parallel. - 2'b10: The value of RDPS field is ignored and only 1 Descriptor is fetched at a time from the System Memory. The next descriptor is fetched only after the data transfer corresponding to current descriptor is completed. The Descriptor close and the next descriptor fetch can still happen in parallel. - 2'b11: All operations are sequential. Descriptor is fetched, data transfer is completed, descriptor is closed; then the next descriptor is fetched. The value of RDPS is ignored in this mode.
0	29:2	r	**Reserved** Read as 0; should be written with 0.

14.6.692 AXI LPI Entry Interval

This register is used to control the AXI LPI entry interval.

AXI_LPI_Entry_Interval Offset address: $1F050_H$

AXI LPI Entry Interval Kernel Reset value: $0000\ 0000_H$



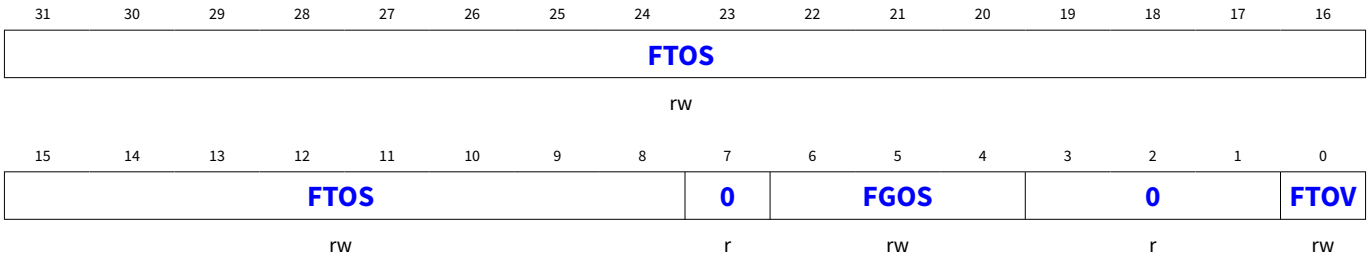
Field	Bits	Type	Description
LPIEI	3:0	rw	**LPI Entry Interval** This field contains the number of system clock cycles, multiplied by 256, to wait for an activity in DWC_xgmac to enter into the AXI low-power state, 0 indicates 256 clock cycles.
0	31:4	r	**Reserved** Read as 0; should be written with 0.

14.6.693 DMA TBS CTRL0

This register is used to control the TBS0 attributes.

DMA_TBS_CTRL0 Offset address: $1F054_H$

DMA TBS CTRL0 Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
FTOV	0	rw	**Fetch Time Offset 0 Valid** When set indicates the FTOS field is valid. When not set, indicates the Fetch Offset is not valid and the DMA engine can fetch the frames from host memory without any time restrictions. 0_B **INVALID**: Fetch Time Offset is invalid 1_B **VALID**: Fetch Time Offset is valid
FGOS	6:4	rw	**Fetch GSN Offset 0** The number of GSN slots that must be deducted from the Launch GSN to compute the Fetch GSN. Value valid only when FTOV is set.

(table continues...)

(continued)

Field	Bits	Type	Description
FTOS	31:8	rw	**Fetch Time Offset 0 - The value in units of 256 ns must be deducted from the Launch time to compute the Fetch Time** - Max value: 999,999,999 ns. Also, this must be smaller than CTR-1 value when ESTM mode is set, as this value is a modulo CTR value.
0	3:1, 7	r	**Reserved** Read as 0; should be written with 0.

14.6.694 DMA TBS CTRL1

This register is used to control the TBS1 attributes.

DMA_TBS_CTRL1 Offset address: $1F058_H$

DMA TBS CTRL1 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FTOS															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FTOS								0	FGOS			0			FTOV
rw								r	rw			r			rw

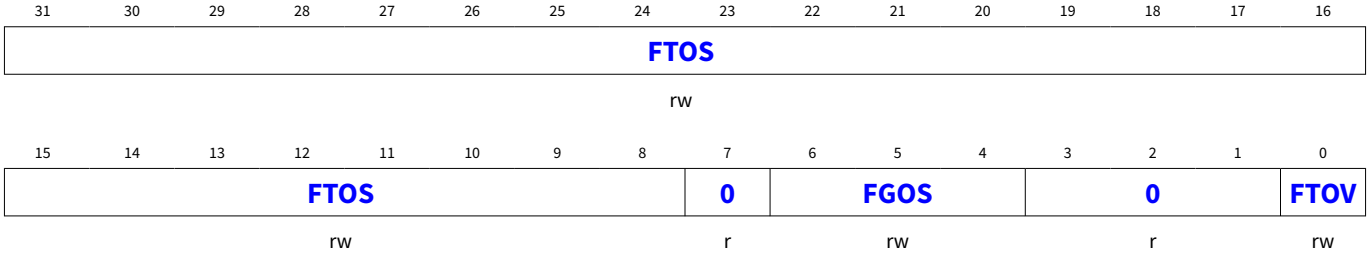
Field	Bits	Type	Description
FTOV	0	rw	**Fetch Time Offset 1 Valid** When set indicates the FTOS field is valid. When not set, indicates the Fetch Offset is not valid and the DMA engine can fetch the frames from host memory without any time restrictions. 0_B **INVALID**: Fetch Time Offset is invalid 1_B **VALID**: Fetch Time Offset is valid
FGOS	6:4	rw	**Fetch GSN Offset 1** The number of GSN slots that must be deducted from the Launch GSN to compute the Fetch GSN. Value valid only when FTOV is set.
FTOS	31:8	rw	**Fetch Time Offset 1 - The value in units of 256 ns, that must be deducted from the Launch time to compute the Fetch Time** - Max value: 999,999,999 ns. Also, this must be smaller than CTR-1 value when ESTM mode is set, as this value is a modulo CTR value.
0	3:1, 7	r	**Reserved** Read as 0; should be written with 0.

14.6.695 DMA TBS CTRL2

This register is used to control the TBS2 attributes.

DMA_TBS_CTRL2 Offset address: $1F05C_H$

DMA TBS CTRL2 Kernel Reset value: $0000\ 0000_H$



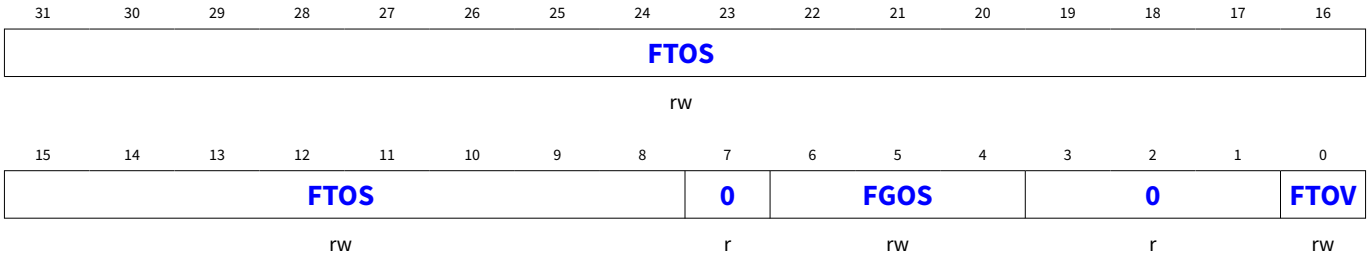
Field	Bits	Type	Description
FTOV	0	rw	**Fetch Time Offset 2 Valid** When set indicates the FTOS field is valid. When not set, indicates the Fetch Offset is not valid and the DMA engine can fetch the frames from host memory without any time restrictions. 0_B **INVALID**: Fetch Time Offset is invalid 1_B **VALID**: Fetch Time Offset is valid
FGOS	6:4	rw	**Fetch GSN Offset 2** The number of GSN slots that must be deducted from the Launch GSN to compute the Fetch GSN. Value valid only when FTOV is set.
FTOS	31:8	rw	**Fetch Time Offset 2** The value in units of 256 ns, that must be deducted from the Launch time to compute the Fetch Time. Max value: 999,999,999 ns. This must be smaller than CTR-1 value when ESTM mode is set, as this value is a modulo CTR value.
0	3:1, 7	r	**Reserved** Read as 0; should be written with 0.

14.6.696 DMA TBS CTRL3

This register is used to control the TBS3 attributes.

DMA_TBS_CTRL3 Offset address: $1F060_H$

DMA TBS CTRL3 Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
FTOV	0	rw	**Fetch Time Offset 3 Valid** When set indicates the FTOS field is valid. When not set, indicates the Fetch Offset is not valid and the DMA engine can fetch the frames from host memory without any time restrictions. 0_B **INVALID**: Fetch Time Offset is invalid 1_B **VALID**: Fetch Time Offset is valid
FGOS	6:4	rw	**Fetch GSN Offset 3** The number GSN slots that must be deducted from the Launch GSN to compute the Fetch GSN. Value valid only when FTOV is set.
FTOS	31:8	rw	**Fetch Time Offset 3** The value in units of 256 ns, that has to be deducted from the Launch time to compute the Fetch Time. Max value: 999,999,999 ns. Also, this must be smaller than CTR-1 value when ESTM mode is set, as this value is a modulo CTR value.
0	3:1, 7	r	**Reserved** Read as 0; should be written with 0.

14.6.697 DMA Safety Interrupt Status

This register indicates summary (whether error occurred in DMA/MTL/MAC and correctable/uncorrectable) of the Automotive Safety related error interrupts.

DMA_Safety_Interrupt_Status Offset address: $1F064_H$

DMA Safety Interrupt Status Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MCSIS	0	MSUIS	MSCIS	0											
r	r	r	r	r											

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0										ASUIS	ASCIS	AMUIS	AMCIS	DEUIS	DECIS
r										r	r	r	r	r	r

Field	Bits	Type	Description
DECIS	0	r	**DMA ECC Correctable error Interrupt Status** This bit indicates an interrupt event in the DMA ECC safety feature. To get the exact cause of the interrupt, the application must read the DMA_ECC_Interrupt_Status register. 0_B **INACTIVE**: DMA ECC Correctable error Interrupt Status not detected 1_B **ACTIVE**: DMA ECC Correctable error Interrupt Status detected

(table continues...)

(continued)

Field	Bits	Type	Description
DEUIS	1	r	**DMA ECC Uncorrectable error Interrupt Status** This bit indicates an interrupt event in the DMA ECC safety feature. To get the exact cause of the interrupt, the application must read the DMA_ECC_Interrupt_Status register. 0_B **INACTIVE**: DMA ECC Uncorrectable error Interrupt Status not detected 1_B **ACTIVE**: DMA ECC Uncorrectable error Interrupt Status detected
AMCIS	2	r	**AXI Master interface Safety Correctable error Interrupt Status This bit indicates an interrupt event in the AXI safety feature** To get the exact cause of the interrupt, the application must read DMA_AM_Sfty_ECC_Intr_Sts. 0_B **INACTIVE**: AXI Master Interface Safety Correctable error Interrupt Status not detected 1_B **ACTIVE**: AXI Master Interface Safety Correctable error Interrupt Status detected
AMUIS	3	r	**AXI Master interface Safety Uncorrectable error Interrupt Status This bit indicates an interrupt event in the AXI safety feature** To get the exact cause of the interrupt, the application must read DMA_AM_Sfty_ECC_Intr_Sts and DMA_AM_Sfty_DPP_Intr_Sts. 0_B **INACTIVE**: AXI Master Interface Safety Uncorrectable error Interrupt Status not detected 1_B **ACTIVE**: AXI Master Interface Safety Uncorrectable error Interrupt Status detected
ASCIS	4	r	**AXI Slave interface Safety Correctable error Interrupt Status Indicates an interrupt event in the AXI safety feature** To get the cause of the interrupt, read the DMA_AS_Sfty_ECC_Intr_Sts register. 0_B **INACTIVE**: AXI Master Interface Safety Correctable error Interrupt Status not detected 1_B **ACTIVE**: AXI Master Interface Safety Correctable error Interrupt Status detected
ASUIS	5	r	**AXI Slave interface Safety Uncorrectable error Interrupt Status Indicates an interrupt event in the AXI safety feature** To get the cause of the interrupt, read the DMA_AS_Sfty_ECC_Intr_Sts and DMA_AS_Sfty_DPP_Intr_Sts. 0_B **INACTIVE**: AXI Master Interface Safety Uncorrectable error Interrupt Status not detected 1_B **ACTIVE**: AXI Master Interface Safety Uncorrectable error Interrupt Status detected
MSCIS	28	r	**MTL Safety Correctable error Interrupt Status** This bit indicates a correctable error interrupt event in MTL. To get exact cause of the interrupt, the software must read the MTL_Safety_Interrupt_Status register. 0_B **INACTIVE**: MTL Safety Correctable error Interrupt Status not detected 1_B **ACTIVE**: MTL Safety Correctable error Interrupt Status detected

(table continues...)

(continued)

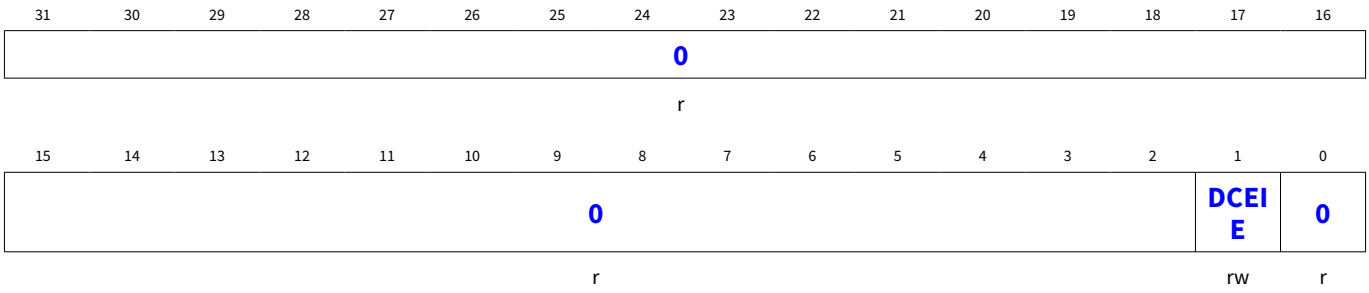
Field	Bits	Type	Description
MSUIS	29	r	**MTL Safety Unrecoverable error Interrupt Status** This bit indicates an uncorrectable error interrupt event in MTL. To get exact cause of the interrupt, the software must read the MTL_Safety_Interrupt_Status register. 0_B **INACTIVE**: MTL Safety Uncorrectable error Interrupt Status not detected 1_B **ACTIVE**: MTL Safety Uncorrectable error Interrupt Status detected
MCSIS	31	r	**MAC Safety Unrecoverable Interrupt Status** Indicates an unrecoverable Safety related Interrupt is set in the MAC module. When this bit is set, to get the cause of the safety interrupt in the MAC, read the MAC_DPP_FSM_Interrupt_Status and DMA_DPP_Interrupt_Status registers. 0_B **INACTIVE**: MAC Safety Uncorrectable Interrupt Status not detected 1_B **ACTIVE**: MAC Safety Uncorrectable Interrupt Status detected
0	27:6, 30	r	**Reserved** Read as 0; should be written with 0.

14.6.698 DMA ECC Interrupt enable

This register is used to enable the Automotive Safety related ECC error interrupt.

DMA_ECC_Interrupt_enable Offset address: $1F068_H$

DMA ECC Interrupt enable Kernel Reset value: $0000\ 0002_H$



Field	Bits	Type	Description
DCEIE	1	rw	**DCACHE memory Correctable Error Interrupt Enable** When set, generates an interrupt when a correctable error is detected at the DMA DCACHE memory interface. It is indicated in the DCES bit of DMA_ECC_Interrupt_Status register. When reset, this event does not generate an interrupt. 0_B **DISABLE**: Descriptor Cache Memory Correctable Error Interrupt is disabled 1_B **ENABLE**: Descriptor Cache Memory Correctable Error Interrupt is enabled
0	0, 31:2	r	**Reserved** Read as 0; should be written with 0.

14.6.699 DMA ECC Interrupt Status

This register indicates the Automotive Safety related ECC error interrupt status.

DMA_ECC_Interrupt_Status Offset address: $1F06C_H$

DMA ECC Interrupt Status Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0									DUES	DAMS	DCES	0	0	0	0
r									rw	rw	rw	r	r	r	r

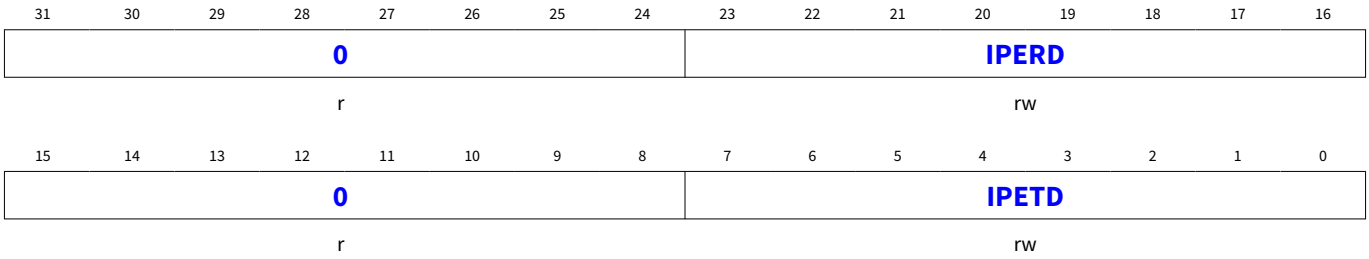
Field	Bits	Type	Description
DCES	4	rw	**DMA DCACHE memory Correctable Error Status** When this bit is set, it indicates that correctable error is detected at DMA DCACHE memory interface. 0_B **INACTIVE**: DMA DCACHE memory Correctable Error status not detected 1_B **ACTIVE**: DMA DCACHE memory Correctable Error status detected
DAMS	5	rw	**DMA DCACHE memory Address Mismatch Status - 1: Indicates that the address mismatch is found for address bus of DMA DCACHE memory** 0_B **INACTIVE**: DMA DCACHE memory Address Mismatch status not detected 1_B **ACTIVE**: DMA DCACHE memory Address Mismatch status detected
DUES	6	rw	**DMA DCACHE memory Uncorrectable Error Status - 1: Indicates that an uncorrectable error is detected at DMA DCACHE memory interface** 0_B **INACTIVE**: DMA DCACHE memory Uncorrectable Error status not detected 1_B **ACTIVE**: DMA DCACHE memory Uncorrectable Error status detected
0	0, 1, 2, 3, 31:7	r	**Read as 0; should be written with 0.**

14.6.700 DMA DPP Control

The DMA_DPP_Control establishes the operating mode of Data Parity protection and error injection per channel in DMA.

DMA_DPP_Control Offset address: $1F070_H$

DMA DPP Control Kernel Reset value: $0000\ 0000_H$



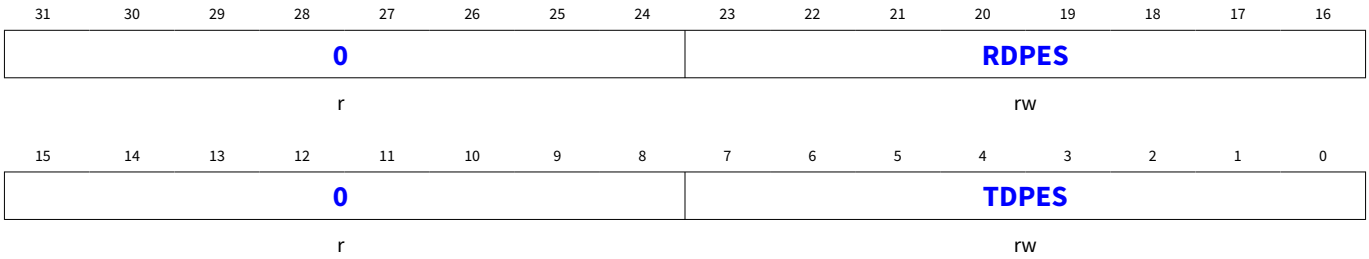
Field	Bits	Type	Description
IPETD	7:0	rw	**Insert Parity Error in DMA TD parity checker - 1: Parity/data bit of the first valid input parity/data of DMA Read Tx descriptor parity checker (or at PC1#td) is flipped** Based on the EIM/BLEI field of MTL_DPP_ECC_EIC register, software can corrupt any one bit of parity/data. DWC_xgmac clears this bit after the respective parity bit is flipped. 00_H **DISABLE**: Insert Parity Error in DMA TD Parity Checker is disabled 01_H **ENABLE**: Insert Parity Error in DMA TD Parity Checker is enabled **others,** Reserved
IPERD	23:16	rw	**Insert Parity Error in DMA RD parity checker - 1: Parity/data bit of the first valid input parity/data of DMA Read Rx descriptor parity checker (or at PC1#rd) is flipped** Based on the EIM/BLEI field of MTL_DPP_ECC_EIC register, software can corrupt any one bit of parity/data. DWC_xgmac clears this bit after the respective parity bit is flipped. 00_H **DISABLE**: Insert Parity Error in DMA RD Parity Checker is disabled 01_H **ENABLE**: Insert Parity Error in DMA RD Parity Checker is enabled **others,** Reserved
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

14.6.701 DMA DPP Interrupt Status

This register contains the status of Automotive Safety related Data Path Parity Errors per channel in DMA.

DMA_DPP_Interrupt_Status Offset address: $1F074_H$

DMA DPP Interrupt Status Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
TDPES	7:0	rw	**Read Tx Descriptor Parity checker Error Status - 1: Indicates that a parity error is detected at the DMA Read Tx descriptor per channel parity checker (or at PC2#td) is flipped** It is cleared when written to 1. 00_H **INACTIVE**: Read Transmit Descriptor(Per Channel) Parity checker Error Status not detected 01_H **ACTIVE**: Read Transmit Descriptor(Per Channel) Parity checker Error Status detected **others,** Reserved
RDPES	23:16	rw	**Read Rx Descriptor Parity checker Error Status - 1: Indicates that a parity error is detected at the DMA Read Rx descriptor per channel parity checker (or at PC2#rd) is flipped** It is cleared when written to 1. 00_H **INACTIVE**: Read Receive Descriptor(Per Channel) Parity checker Error Status not detected 01_H **ACTIVE**: Read Receive Descriptor(Per Channel) Parity checker Error Status detected **others,** Reserved
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

14.6.702 DMA CH Ind Ctrl

This register is used to program Indirect Access DMA configuration registers.

DMA_CH_Ind_Ctrl Offset address: $1F080_H$

DMA CH Ind Ctrl Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0												MSEL			
r												rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				AOFF				0		AUTO		0		CT	OB
r				rw				r		rw		r		rw	rw

Field	Bits	Type	Description
OB	0	rw	**Operation Busy** This bit is set along with a read or write command for initiating the indirect access to DMA_(#MSEL)_(#AOFF) register. This bit is reset when the read or write command to DMA_(#MSEL)_(#AOFF) register is complete. The next indirect register access can be initiated only after this bit is reset. During a write operation, the bit is reset only after the data has been written into the DMA_(#MSEL)_(#AOFF) register. During a read operation, the data must be read from the DMA_CH_Ind_Data register, only after this bit is reset.

(table continues...)

(continued)

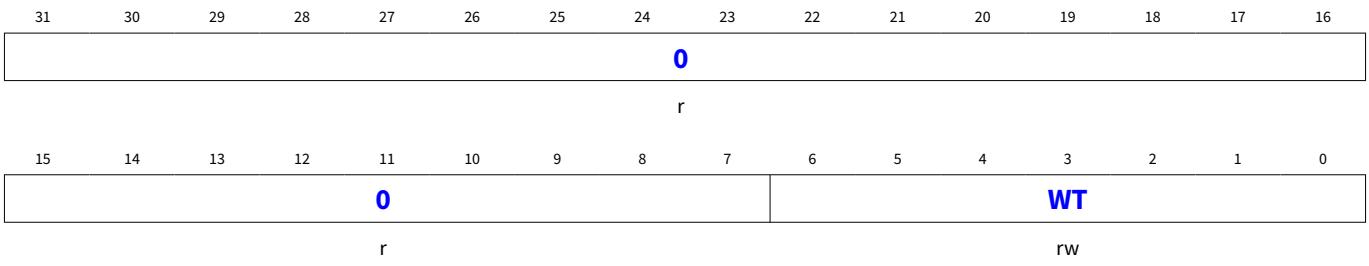
Field	Bits	Type	Description
CT	1	rw	**Command Type** This bit indicates if the current register access is a read or a write operation. - 1: Indicates a read operation. - 0: Indicates a write operation.
AUTO	5:4	rw	**Auto Increment - 00: No auto increment - 01: Auto increment MSEL only - 10: Auto increment AOFF only - 11: No auto increment**
AOFF	11:8	rw	**Address Offset This field selects the offset address of per DMA Channel registers specified by MSEL field** - 0000: CH0 - 0001: CH1 - ---- - 1111 : CH15 AOFF field is used in the indirect access of the DMA_(#MSEL)_(#AOFF) This field must be set along with initiation of read/write to the DMA_(#MSEL)_(#AOFF) register and must not be changed until the OB is reset.
MSEL	19:16	rw	**Mode Select** This field selects a group of identical registers. - 0000: DMA_CH(#i)_TxExtCfg registers - 0001-1111: reserved for future use MSEL field is used in the indirect access of the DMA_(#MSEL)_(#AOFF) This field must be set along with initiation of read/write to the DMA_(#MSEL)_(#AOFF) register and must not be changed until the OB is reset.
0	3:2, 7:6, 15:12, 31:20	r	**Reserved** Read as 0; should be written with 0.

14.6.703 DMA CH Ind Data

This register holds the read/write data for Indirect Access of the registers indicated by DMA_(#MSEL)_(#AOFF). During the read access, this field contains valid read data only after the OB bit is reset. During the write access, this field must be valid prior setting the OB bit in the DMA_CH_Ind_Data register. Once auto increment is set, application is allowed to write without setting OB bit of DMA_CH_Ind_Ctrl register.

DMA_CH_Ind_Data Offset address: $1F084_H$

DMA CH Ind Data Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
WT	6:0	rw	**Weights** This field contains the weights of the corresponding Tx DMA used for Weighted Round Robin Arbitration of Tx DMA Data Reads. The maximum value is 0x64.
0	31:7	r	**Reserved** Read as 0; should be written with 0.

14.6.704 DMA Channel j TxExtCfg

This register contains data related to per channel Tx weights used for Weighted Round Robin Arbitration of Tx DMA Data Reads.

DMA_CH0_TxExtCfg Offset address: $1F084_H$

DMA Channel j TxExtCfg Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0									WT						
r									rw						

Field	Bits	Type	Description
WT	6:0	rw	**Weights** This field contains the weights of corresponding Tx DMA used for Weighted Round Robin Arbitration of Tx DMA Data Reads. The maximum value is 0x64.
0	31:7	r	**Reserved** Read as 0; should be written with 0.

14.6.705 DMA Channel j TxExtCfg

This register contains data related to per channel Tx weights used for Weighted Round Robin Arbitration of Tx DMA Data Reads.

DMA_CH1_TxExtCfg Offset address: $1F084_H$

DMA Channel j TxExtCfg Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0									WT						
r									rw						

Field	Bits	Type	Description
WT	6:0	rw	**Weights** This field contains the weights of corresponding Tx DMA used for Weighted Round Robin Arbitration of Tx DMA Data Reads. The maximum value is 0x64.
0	31:7	r	**Reserved** Read as 0; should be written with 0.

14.6.706 DMA Channel j TxExtCfg

This register contains data related to per channel Tx weights used for Weighted Round Robin Arbitration of Tx DMA Data Reads.

DMA_CH2_TxExtCfg Offset address: $1F084_H$

DMA Channel j TxExtCfg Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0									WT						
r									rw						

Field	Bits	Type	Description
WT	6:0	rw	**Weights** This field contains the weights of corresponding Tx DMA used for Weighted Round Robin Arbitration of Tx DMA Data Reads. The maximum value is 0x64.
0	31:7	r	**Reserved** Read as 0; should be written with 0.

14.6.707 DMA Channel j TxExtCfg

This register contains data related to per channel Tx weights used for Weighted Round Robin Arbitration of Tx DMA Data Reads.

DMA_CH3_TxExtCfg Offset address: $1F084_H$

DMA Channel j TxExtCfg Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0									WT						
r									rw						

Field	Bits	Type	Description
WT	6:0	rw	**Weights** This field contains the weights of corresponding Tx DMA used for Weighted Round Robin Arbitration of Tx DMA Data Reads. The maximum value is 0x64.
0	31:7	r	**Reserved** Read as 0; should be written with 0.

14.6.708 DMA Channel j TxExtCfg

This register contains data related to per channel Tx weights used for Weighted Round Robin Arbitration of Tx DMA Data Reads.

DMA_CH4_TxExtCfg Offset address: $1F084_H$

DMA Channel j TxExtCfg Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0									WT						
r									rw						

Field	Bits	Type	Description
WT	6:0	rw	**Weights** This field contains the weights of corresponding Tx DMA used for Weighted Round Robin Arbitration of Tx DMA Data Reads. The maximum value is 0x64.
0	31:7	r	**Reserved** Read as 0; should be written with 0.

14.6.709 DMA Channel j TxExtCfg

This register contains data related to per channel Tx weights used for Weighted Round Robin Arbitration of Tx DMA Data Reads.

DMA_CH5_TxExtCfg Offset address: $1F084_H$

DMA Channel j TxExtCfg Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0									WT						
r									rw						

Field	Bits	Type	Description
WT	6:0	rw	**Weights** This field contains the weights of corresponding Tx DMA used for Weighted Round Robin Arbitration of Tx DMA Data Reads. The maximum value is 0x64.
0	31:7	r	**Reserved** Read as 0; should be written with 0.

14.6.710 DMA Channel j TxExtCfg

This register contains data related to per channel Tx weights used for Weighted Round Robin Arbitration of Tx DMA Data Reads.

DMA_CH6_TxExtCfg Offset address: $1F084_H$

DMA Channel j TxExtCfg Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0									WT						
r									rw						

Field	Bits	Type	Description
WT	6:0	rw	**Weights** This field contains the weights of corresponding Tx DMA used for Weighted Round Robin Arbitration of Tx DMA Data Reads. The maximum value is 0x64.
0	31:7	r	**Reserved** Read as 0; should be written with 0.

14.6.711 DMA Channel j TxExtCfg

This register contains data related to per channel Tx weights used for Weighted Round Robin Arbitration of Tx DMA Data Reads.

DMA_CH7_TxExtCfg Offset address: $1F084_H$

DMA Channel j TxExtCfg Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0									WT						
r									rw						

Field	Bits	Type	Description
WT	6:0	rw	**Weights** This field contains the weights of corresponding Tx DMA used for Weighted Round Robin Arbitration of Tx DMA Data Reads. The maximum value is 0x64.
0	31:7	r	**Reserved** Read as 0; should be written with 0.

14.6.712 DMA Sfty Ind Ctrl

This a control register to access DMA Safety related indirect registers.

DMA_Sfty_Ind_Ctrl Offset address: $1F088_H$

DMA Sfty Ind Ctrl Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								DAPS		0		MSEL			
r								rw		r		rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				AOFF				0		AUTO		0		CT	OB
r				rw				r		rw		r		rw	rw

Field	Bits	Type	Description
OB	0	rw	**Operation Busy - 1: Indicates the initiation of access to DMA Safety indirect register space** The DWC_xgmac clears this bit when the access to DMA Safety indirect register is complete. The requested data from indirect register is available in DMA_Sfty_Ind_Data register for read operation. The application must not change content of this register while this bit is set.
CT	1	rw	**Command Type - 0: Indicates a read operation** - 1: Indicates a write operation.
AUTO	5:4	rw	**Auto Increment** This field indicates that the next access is for incremental MSEL or AOFF, the application need not program the MSEL or AOFF for subsequent accesses. - 00: No Auto increment - 01: Auto increment MSEL only - 10: Auto increment AOFF only - 11: No Auto increment

(table continues...)

(continued)

Field	Bits	Type	Description
AOFF	11:8	rw	**Address Offset** This field indicates offset address of DMA Safety indirect registers within the set of registers. If MSEL is 0000: - 0000: DMA_Sfty_Dbg_EIC - 0101 to 1111 - reserved If MSEL is 0001: - 0000: DMA_AM_Sfty_Ctrl - 0001: DMA_AM_Sfty_Intr_En - 0010: DMA_AM_Sfty_ECC_Intr_Sts - 0011: DMA_AM_Sfty_DPP_Intr_Sts - 0100: DMA_AM_Sfty_RDC_ECC_Err_Cntr_Sts - 0101 to 1111 - reserved If MSEL is 0010: - 0000: DMA_AM_Sfty_RDC_Dbg_Data - 0001: DMA_AM_Sfty_RDC_Dbg_Ctrl - 0010:DMA_AM_Sfty_WRC_Dbg_Ctrl If MSEL is 0011: - 0000: DMA_AS_Sfty_Ctrl - 0001: DMA_AS_Sfty_Intr_En - 0010: DMA_AS_Sfty_ECC_Intr_Sts - 0011: DMA_AS_Sfty_DPP_Intr_Sts - 0100: DMA_AS_Sfty_WDC_ECC_Err_Cntr_Sts If MSEL is 0100: - 0000: DMA_AS_Sfty_WDC_Dbg_Data - 0001: DMA_AS_Sfty_WDC_Dbg_Ctrl - 0010: DMA_AS_Sfty_RAC_Dbg_Addr - 0011: DMA_AS_Sfty_RAC_Dbg_Ctrl - 0100: DMA_AS_Sfty_WAC_Dbg_Addr - 0101: DMA_AS_Sfty_WAC_Dbg_Ctrl - 0110-1111: Reserved
MSEL	19:16	rw	**Mode Select** This field indicates which set of DMA Safety indirect registers is accessed. - 0000: Common Safety Registers - 0001: AXI Master Interface Safety Registers - 0010: AXI Master Interface Safety Debug Error Injection Registers - 0011: AXI Slave Interface Safety Registers - 0100: AXI Slave Interface Safety Debug Error Injection Registers - 0101-1111: reserved for future use

(table continues...)

(continued)

Field	Bits	Type	Description
DAPS	23:22	rw	**Data or Address Part Select** This field indicates which 32-bit data or address word is written to AXI Master Read Data channel data, AXI Slave Write Data channel data, AXI Slave Read Address channel address or AXI Slave Write Address channel address, in the debug mode. - 00: data[31:0] - 01: data[63:32] - 10: data[95:64] - 11: data[127:96]
0	3:2, 7:6, 15:12, 21:20, 31:24	r	**Reserved** Read as 0; should be written with 0.

14.6.713 DMA Sfty Ind Data

This field contains data to read/write from DMA Safety indirect registers.

DMA_Sfty_Ind_Data Offset address: $1F08C_H$

DMA Sfty Ind Data Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DATA															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA															
rw															

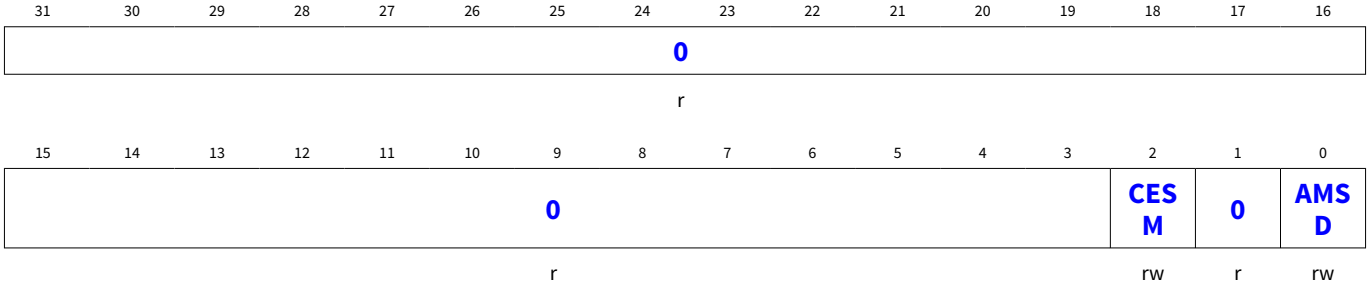
Field	Bits	Type	Description
DATA	31:0	rw	**Read or Write Data** This field contains data to read/write from DMA Safety indirect register. - Read access: This field contains valid read data after the OB bit in the DMA_Sfty_Ind_Data register is reset. - Write access: This field must be valid prior to setting the OB bit in the DMA_Sfty_Ind_Data register. When Auto Increment bit is set, the application can write new data without setting OB bit of DMA_Sfty_Ind_Data register.

14.6.714 DMA AM Sfty Ctrl

This register contains control signal for AXI Master interface safety.

DMA_AM_Sfty_Ctrl Offset address: $1F08C_H$

DMA AM Sfty Ctrl Kernel Reset value: $0000\ 0000_H$



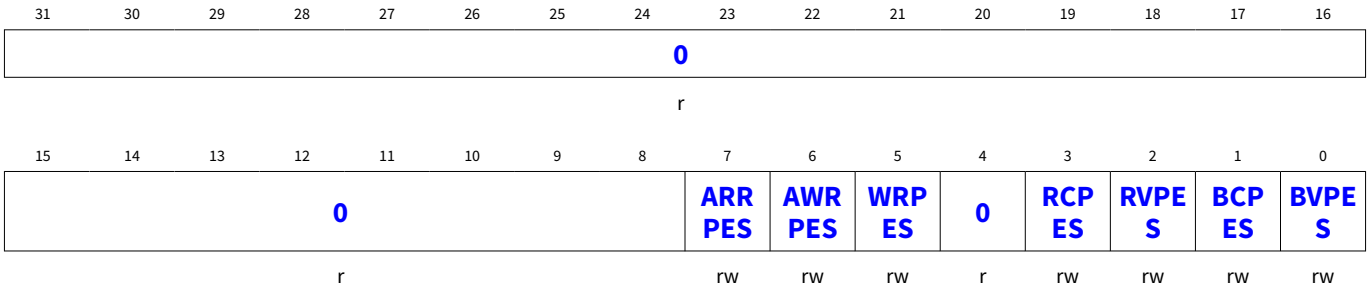
Field	Bits	Type	Description
AMSD	0	rw	**AXI Master Safety Disable - 1: Disables the safety features for all the channels of AXI Master Interface** - 0: Enables the safety features for all the channels of AXI Master Interface.
CESM	2	rw	**Clear Error Status for AXI Master Interface Safety - 1: AXI Master Interface Safety ECC uncorrectable and correctable error counters are cleared upon reading by application** The DWC_xgmac clears this bit when all the error counters are cleared.
0	1, 31:3	r	**Reserved** Read as 0; should be written with 0.

14.6.715 DMA AM Sfty DPP Intr Sts

This register contains the DPP interrupt status for AXI Master Interface safety.

DMA_AM_Sfty_DPP_Intr_Sts Offset address: $1F08C_H$

DMA AM Sfty DPP Intr Sts Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
BVPES	0	rw	**AXI Master Write Response Channel Valid Parity Error Status - 1: Indicates that a parity error is detected at the AXI Master Write Response channel valid signal**
BCPES	1	rw	**AXI Master Write Response channel Control Parity Error Status - 1: Indicates that a parity error is detected at the AXI Master Write Response channel control signals**
RVPES	2	rw	**AXI Master Read Data channel Valid Parity Error Status - 1: Indicates that a parity error is detected on the AXI Master Read Data channel valid signal**

(table continues...)

(continued)

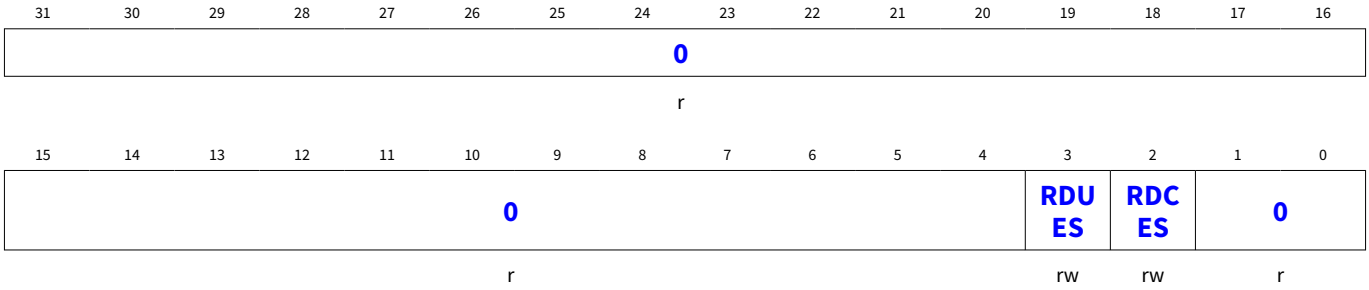
Field	Bits	Type	Description
RCPES	3	rw	**AXI Master Read Data channel Control Parity Error Status - 1: Indicates that a parity error is detected on the AXI Master Read Data channel control signals**
WRPES	5	rw	**AXI Master Write Data channel Ready Parity Error Status - 1: Indicates that a parity error is detected on AXI Master W channel Ready signal**
AWRPES	6	rw	**AXI Master Write Address channel Ready Parity Error Status - 1: Indicates that a parity error is detected on the AXI Master AW channel Ready signal**
ARRPES	7	rw	**AXI Master Read Address channel Ready Parity Error Status - 1: Indicates that a parity error is detected on the AXI Master Read Address channel Ready signal**
0	4, 31:8	r	**Reserved** Read as 0; should be written with 0.

14.6.716 DMA AM Sfty ECC Intr Sts

This register contains the ECC interrupt status for the AXI Master Interface Safety.

DMA_AM_Sfty_ECC_Intr_Sts Offset address: $1F08C_H$

DMA AM Sfty ECC Intr Sts Kernel Reset value: $0000\ 0000_H$



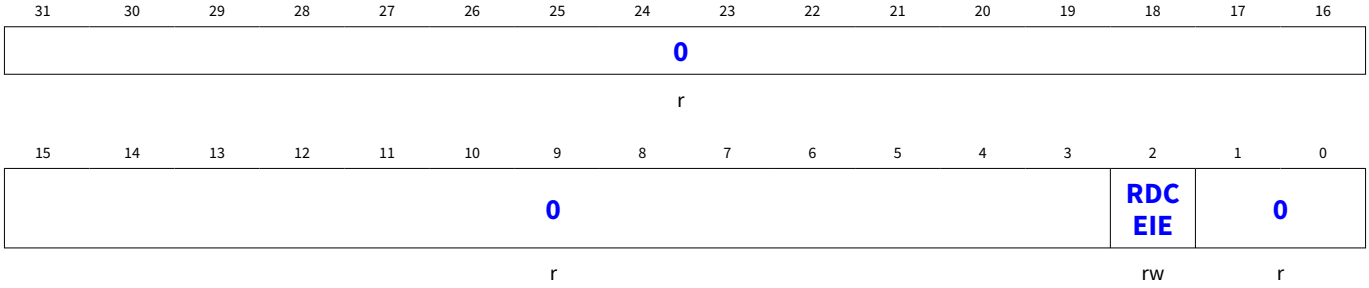
Field	Bits	Type	Description
RDCES	2	rw	**AXI Master Read Data channel data Correctable Error Status - 1: Indicates that a correctable error is detected at AXI Master Read Data channel data**
RDUES	3	rw	**AXI Master Read Data channel data Uncorrectable Error Status - 1: Indicates that an uncorrectable error is detected at AXI Master Read Data channel data**
0	1:0, 31:4	r	**Reserved** Read as 0; should be written with 0.

14.6.717 DMA AM Sfty Intr En

This register contains the interrupt enables for AXI Master Interface Safety.

DMA_AM_Sfty_Intr_En Offset address: $1F08C_H$

DMA AM Sfty Intr En Kernel Reset value: $0000\ 0000_H$



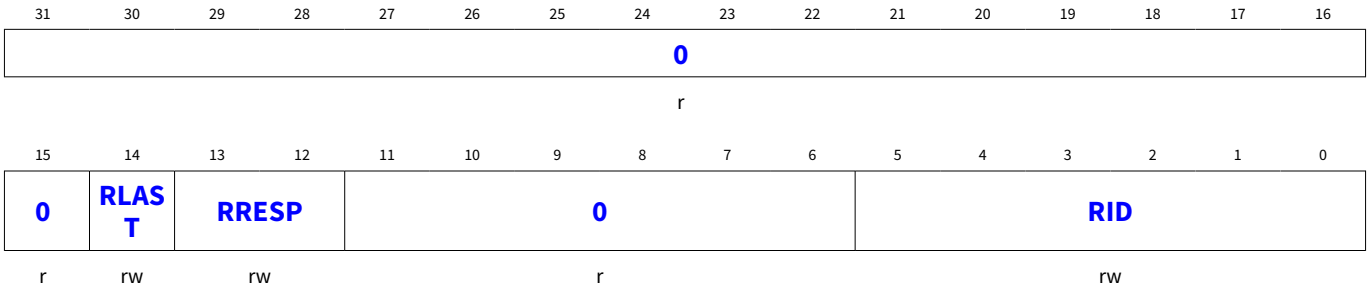
Field	Bits	Type	Description
RDCEIE	2	rw	**AXI Master Read Data channel Data Correctable Error Interrupt Enable - 1: DWC_xgmac generates an interrupt when a correctable error is detected in the AXI Master Read Data channel data and RDCES bit of DMA_AM_Sfty_ECC_Intr_Sts register is set** - 0: DWC_xgmac does not generate an interrupt when a correctable error is detected in the AXI Master Read Data channel data and RDCES bit of DMA_AM_Sfty_ECC_Intr_Sts register is set.
0	1:0, 31:3	r	**Reserved** Read as 0; should be written with 0.

14.6.718 DMA AM Sfty RDC Dbg Ctrl

This register contains the data to be driven on AXI Master Read Data channel control signals in Debug Error Injection mode.

DMA_AM_Sfty_RDC_Dbg_Ctrl Offset address: $1F08C_H$

DMA AM Sfty RDC Dbg Ctrl Kernel Reset value: $0000\ 0000_H$



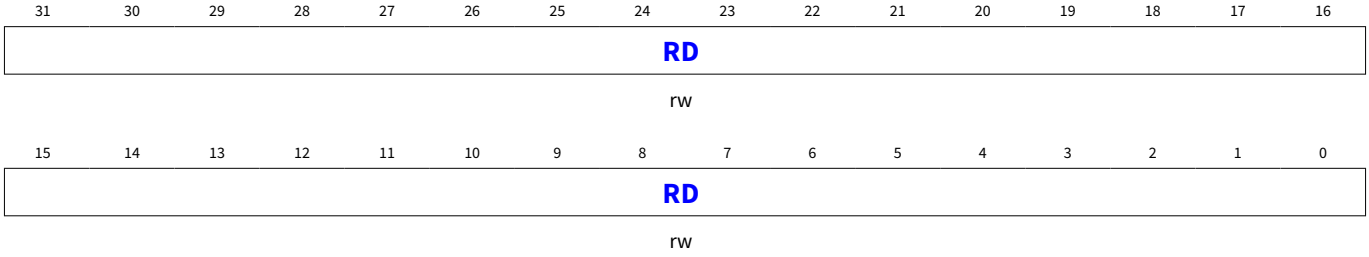
Field	Bits	Type	Description
RID	5:0	rw	**AXI Master Read Data channel RID Control Signal** This field contains the debug data to be driven on AXI Master RID signal.
RRESP	13:12	rw	**AXI Master Read Data channel RRESP Control Signal** This field contains the debug data to be driven on AXI Master RRESP signal.
RLAST	14	rw	**AXI Master Read Data channel RLAST Control Signal** This bit contains the debug data to be driven on AXI Master RLAST signal.
0	11:6, 31:15	r	**Reserved** Read as 0; should be written with 0.

14.6.719 DMA AM Sfty RDC Dbg Data

This register contains the data to be driven on AXI Master read data in Debug Error Injection mode.

DMA_AM_Sfty_RDC_Dbg_Data Offset address: $1F08C_H$

DMA AM Sfty RDC Dbg Data Kernel Reset value: $0000\ 0000_H$



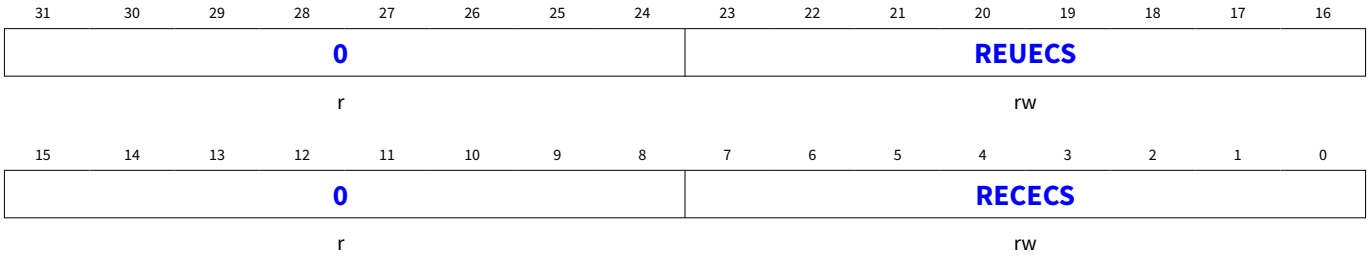
Field	Bits	Type	Description
RD	31:0	rw	**AXI Master Read Data channel Debug Data** This field contains the 32-bit debug data to be driven on AXI Master Read Data bus. The DAPS field of DMA_Sfty_Ind_Ctl register indicates which 32-bit word of read data bus this correspond to.

14.6.720 DMA AM Sfty RDC ECC Err Cntr Sts

This register contains the ECC Error counters for read data of AXI Master Interface safety.

DMA_AM_Sfty_RDC_ECC_Err_Cntr_Sts Offset address: $1F08C_H$

DMA AM Sfty RDC ECC Err Cntr Sts Kernel Reset value: $0000\ 0000_H$



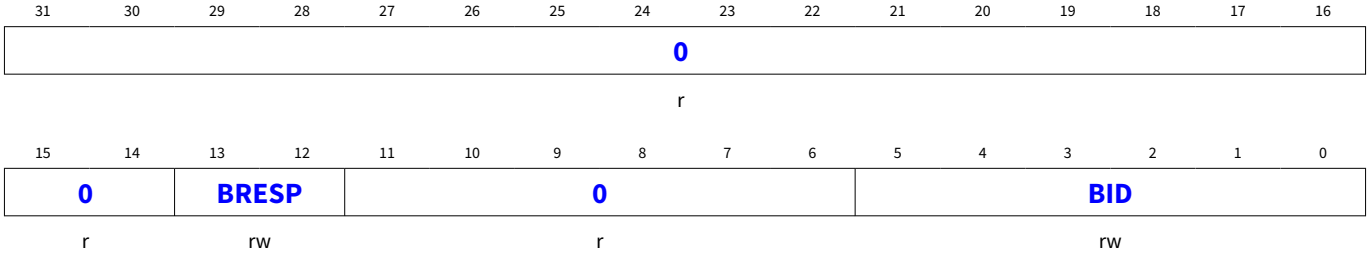
Field	Bits	Type	Description
RECECS	7:0	rw	**AXI Master Read Data channel ECC Correctable Error Counter** This field provides the number of ECC correctable errors detected in AXI Master Read Data channel. This field is cleared when the CESM bit is set to 1 and the application reads this field.
REUECS	23:16	rw	**AXI Master Read Data channel ECC Uncorrectable Error Counter** This field provides the number of ECC uncorrectable errors detected in AXI Master Read Data channel. This field is cleared when the CESM bit is set to 1 and the application reads this field.
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

14.6.721 DMA AM Sfty WRC Dbg Ctrl

This register contains the data to be driven on AXI Master write response channel control signals in Debug Error Injection mode.

DMA_AM_Sfty_WRC_Dbg_Ctrl Offset address: $1F08C_H$

DMA AM Sfty WRC Dbg Ctrl Kernel Reset value: $0000\ 0000_H$



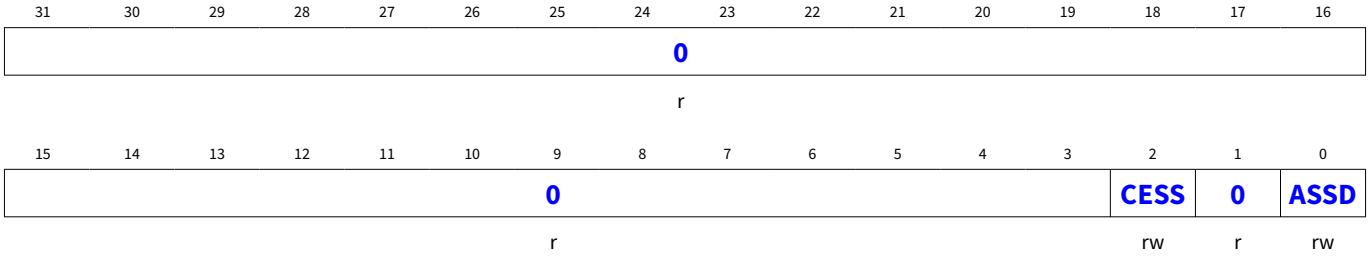
Field	Bits	Type	Description
BID	5:0	rw	**AXI Master Response Channel BID Control Signal** This field contains the debug data to be driven on AXI Master BID signal.
BRESP	13:12	rw	**AXI Master Response Channel BRESP Control Signal** This field contains the debug data to be driven on AXI Master BRESP signal.
0	11:6, 31:14	r	**Reserved** Read as 0; should be written with 0.

14.6.722 DMA AS Sfty Ctrl

This register contains control signal for AXI Slave interface safety module.

DMA_AS_Sfty_Ctrl Offset address: $1F08C_H$

DMA AS Sfty Ctrl Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
ASSD	0	rw	**AXI Slave Safety Disable - 1: Disables the safety features for all the channels of AXI Slave Interface** - 0: Enables the safety features for all the channels of AXI Slave Interface.

(table continues...)

(continued)

Field	Bits	Type	Description
CESS	2	rw	**Clear Error Status for AXI Slave Interface Safety - 1: AXI Slave Interface Safety ECC uncorrectable and correctable error counters are cleared upon reading by application** The DWC_xgmac clears this bit when all the error counters are cleared.
0	1, 31:3	r	**Reserved** Read as 0; should be written with 0.

14.6.723 DMA AS Sfty DPP Intr Sts

This register contains the DPP interrupt status for AXI Slave Interface safety.

DMA_AS_Sfty_DPP_Intr_Sts Offset address: $1F08C_H$

DMA AS Sfty DPP Intr Sts Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0					ARAPES	ARCPES	ARVPES	AWAPES	AWCPES	AWVPES	0	WCPES	WVPES	RRPES	BRPES
r					rw	rw	rw	rw	rw	rw	r	rw	rw	rw	rw

Field	Bits	Type	Description
BRPES	0	rw	**AXI Slave Write Response channel Ready Parity Error Status - 1: Indicates that a parity error is detected on the AXI Slave Write Response channel ready signal**
RRPES	1	rw	**AXI Slave Read Data channel Ready Parity Error Status - 1: Indicates that a parity error is detected on the AXI Slave Read Data channel ready signal**
WVPES	2	rw	**AXI Slave Write Data channel Valid Parity Error Status - 1: Indicates that a parity error is detected on the AXI Slave Write Data channel valid signal**
WCPES	3	rw	**AXI Slave Write Data channel Control Parity Error Status - 1: Indicates that a parity error is detected on the AXI Slave Write Data channel control signal**
AWVPES	5	rw	**AXI Slave Write Address channel Valid Parity Error Status** When this bit is set to 1, it indicates that a parity error is detected on the AXI Slave Write Address channel valid signal.
AWCPES	6	rw	**AXI Slave Write Address channel Control Parity Error Status** When this bit is set to 1, it indicates that a parity error is detected on the AXI Slave Write Address channel control signal.
AWAPES	7	rw	**AXI Slave Write Address channel Address Parity Error Status** When this bit is set to 1, it indicates that a parity error is detected on the AXI Slave Write Address channel address signal.

(table continues...)

(continued)

Field	Bits	Type	Description
ARVPES	8	rw	**AXI Slave Read Address channel Valid Parity Error Status** When this bit is set to 1, it indicates that a parity error is detected on the AXI Slave Read Address channel valid signal.
ARCPES	9	rw	**AXI Slave Read Address channel Control Parity Error Status** When this bit is set to 1, it indicates that a parity error is detected on the AXI Slave Read Address channel control signal.
ARAPES	10	rw	**AXI Slave Read Address channel Address Parity Error Status** When this bit is set to 1, it indicates that a parity error is detected on the AXI Slave Read Address channel address signal.
0	4, 31:11	r	**Reserved** Read as 0; should be written with 0.

14.6.724 DMA AS Sfty ECC Intr Sts

This register contains the ECC interrupt status for the AXI Slave Interface Safety.

DMA_AS_Sfty_ECC_Intr_Sts Offset address: $1F08C_H$

DMA AS Sfty ECC Intr Sts Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												WDUES	WDCES	0	
r												rw	rw	r	

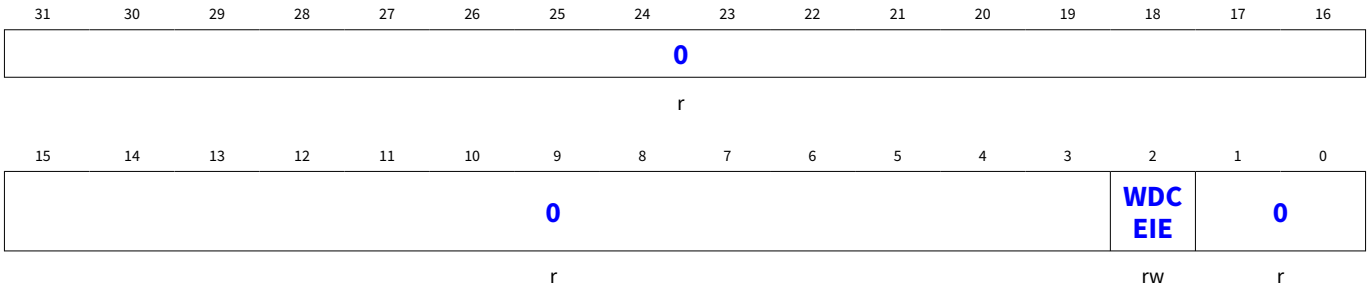
Field	Bits	Type	Description
WDCES	2	rw	**AXI Slave Write Data channel Data Correctable Error Status** When this bit is set to 1, it indicates that a correctable error is detected at AXI Slave Write Data channel data.
WDUES	3	rw	**AXI Slave Write Data channel Data Uncorrectable Error Status - 1: Indicates that an uncorrectable error is detected at AXI Slave Write Data channel data**
0	1:0, 31:4	r	**Reserved** Read as 0; should be written with 0.

14.6.725 DMA AS Sfty Intr En

This register contains the interrupt enables for AXI Slave Interface Safety.

DMA_AS_Sfty_Intr_En Offset address: $1F08C_H$

DMA AS Sfty Intr En Kernel Reset value: $0000\ 0000_H$



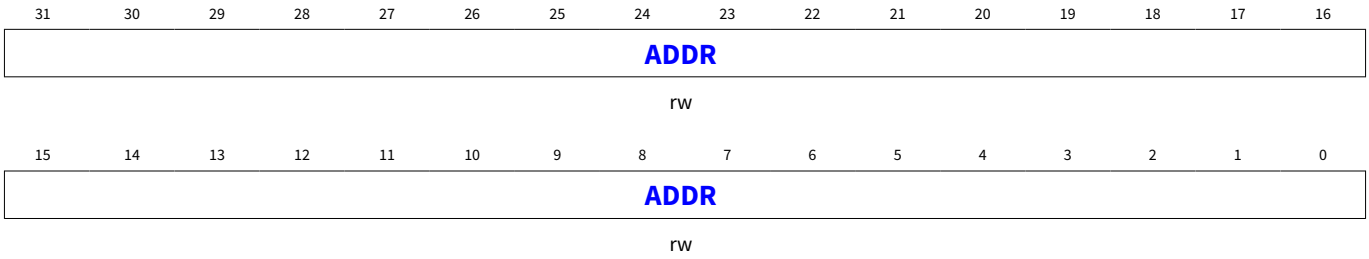
Field	Bits	Type	Description
WDCEIE	2	rw	**AXI Slave Write Data channel Data Correctable Error Interrupt Enable - 1: DWC_xgmac generates an interrupt when a correctable error is detected in the AXI Slave Write Data channel data and WDCES bit of DMA_AS_Sfty_ECC_Intr_Sts register is set** - 0: DWC_xgmac does not generate an interrupt when a correctable error is detected in the AXI Slave Write Data channel data and WDCES bit of DMA_AS_Sfty_ECC_Intr_Sts register is set.
0	1:0, 31:3	r	**Reserved** Read as 0; should be written with 0.

14.6.726 DMA AS Sfty RAC Dbg Addr

This register contains the data to be driven on address bus of AXI Slave read address channel in Debug Error Injection mode.

DMA_AS_Sfty_RAC_Dbg_Addr Offset address: $1F08C_H$

DMA AS Sfty RAC Dbg Addr Kernel Reset value: $0000\ 0000_H$



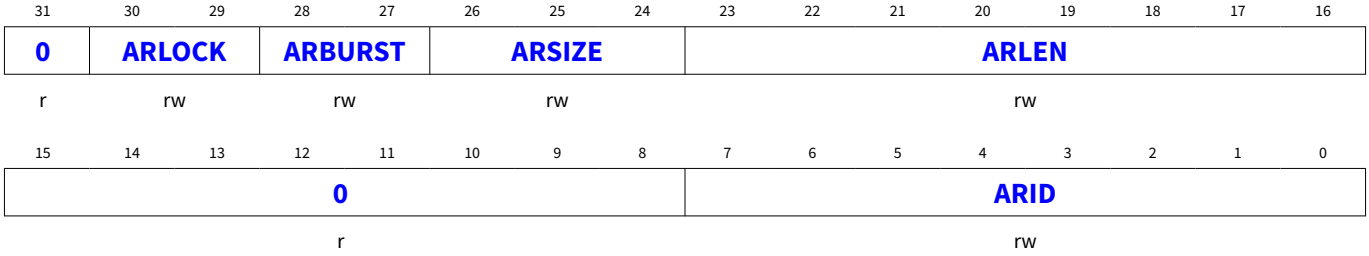
Field	Bits	Type	Description
ADDR	31:0	rw	**AXI Slave Read Address Channel Address** This field contains the 32-bit debug data to be driven on AXI Slave Read address bus. The DAPS field of DMA_Sfty_Ind_Ctl register indicates which 32-bit word of Read address bus this correspond to.

14.6.727 DMA AS Sfty RAC Dbg Ctrl

This register contains the data to be driven on control signals of AXI Slave read address channel in Debug Error Injection mode.

DMA_AS_Sfty_RAC_Dbg_Ctrl Offset address: $1F08C_H$

DMA AS Sfty RAC Dbg Ctrl Kernel Reset value: $0000\ 0000_H$



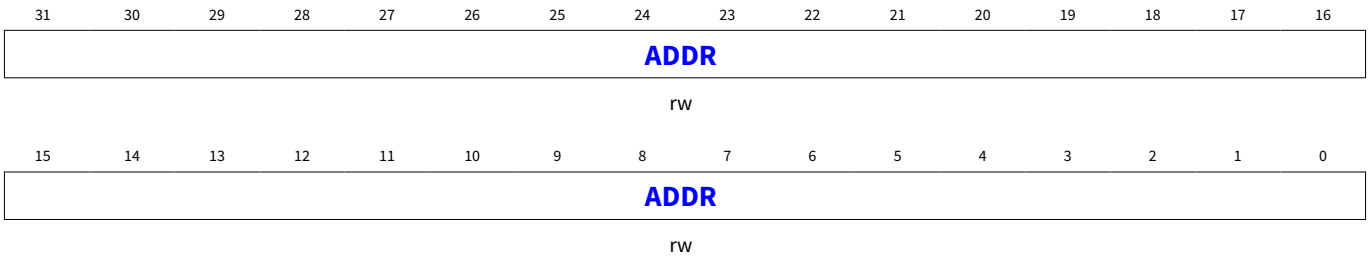
Field	Bits	Type	Description
ARID	7:0	rw	**AXI Slave Address Read Address channel ARID Control Signal** This field contains the debug data to be driven on AXI Slave ARID signal.
ARLEN	23:16	rw	**AXI Slave Address Read Address channel ARLEN Control Signal** This field contains the debug data to be driven on AXI Slave ARLEN signal.
ARSIZE	26:24	rw	**AXI Slave Address Read Address channel ARSIZE Control Signal** This field contains the debug data to be driven on AXI Slave ARSIZE signal.
ARBURST	28:27	rw	**AXI Slave Address Read Address channel ARBURST Control Signal** This field contains the debug data to be driven on AXI Slave ARBURST signal.
ARLOCK	30:29	rw	**AXI Slave Address Read Address channel ARLOCK Control Signal** This field contains the debug data to be driven on AXI Slave ARLOCK signal.
0	15:8, 31	r	**Reserved** Read as 0; should be written with 0.

14.6.728 DMA AS Sfty WAC Dbg Addr

This register contains the data to be driven on address bus of AXI Slave write address channel in Debug Error Injection mode.

DMA_AS_Sfty_WAC_Dbg_Addr Offset address: $1F08C_H$

DMA AS Sfty WAC Dbg Addr Kernel Reset value: $0000\ 0000_H$



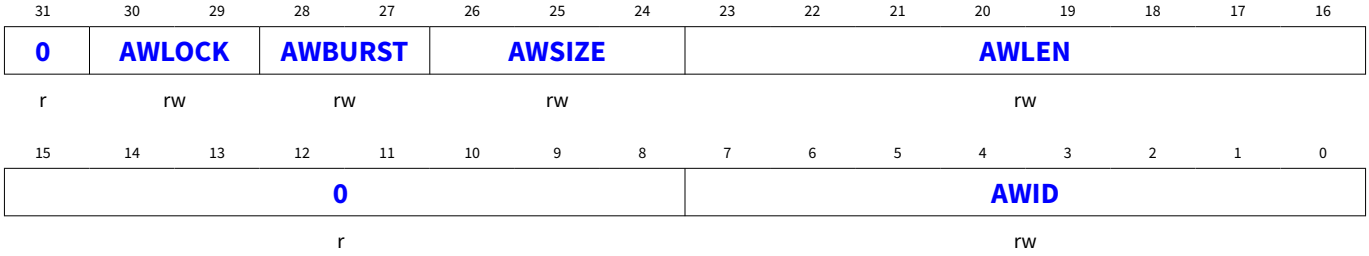
Field	Bits	Type	Description
ADDR	31:0	rw	**AXI Slave Address Write Address channel address** This field contains the 32-bit debug data to be driven on AXI Slave Write address bus. The DAPS field of DMA_Sfty_Ind_Ctl register indicates which 32-bit word of Write address bus this correspond to.

14.6.729 DMA AS Sfty WAC Dbg Ctrl

This register contains the data to be driven on control signals of AXI Slave write address channel in Debug Error Injection mode.

DMA_AS_Sfty_WAC_Dbg_Ctrl Offset address: $1F08C_H$

DMA AS Sfty WAC Dbg Ctrl Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
AWID	7:0	rw	**AXI Slave Address Write Address channel AWID Control Signal** This field contains the debug data to be driven on AXI Slave AWID signal.
AWLEN	23:16	rw	**AXI Slave Address Write Address channel AWLEN Control Signal** This field contains the debug data to be driven on AXI Slave AWLEN signal.
AWSIZE	26:24	rw	**AXI Slave Address Write Address channel AWSIZE Control Signal** This field contains the debug data to be driven on AXI Slave AWSIZE signal.
AWBURST	28:27	rw	**AXI Slave Address Write Address channel AWBURST Control Signal** This field contains the debug data to be driven on AXI Slave AWBURST signal.
AWLOCK	30:29	rw	**AXI Slave Address Write Address channel AWLOCK Control Signal** This field contains the debug data to be driven on AXI Slave AWLOCK signal.
0	15:8, 31	r	**Reserved** Read as 0; should be written with 0.

14.6.730 DMA AS Sfty WDC Dbg Ctrl

This register contains the data to be driven on AXI Slave Write Data channel control signals in Debug Error Injection mode.

DMA_AS_Sfty_WDC_Dbg_Ctrl Offset address: $1F08C_H$

DMA AS Sfty WDC Dbg Ctrl Kernel Reset value: $0000\ 0000_H$

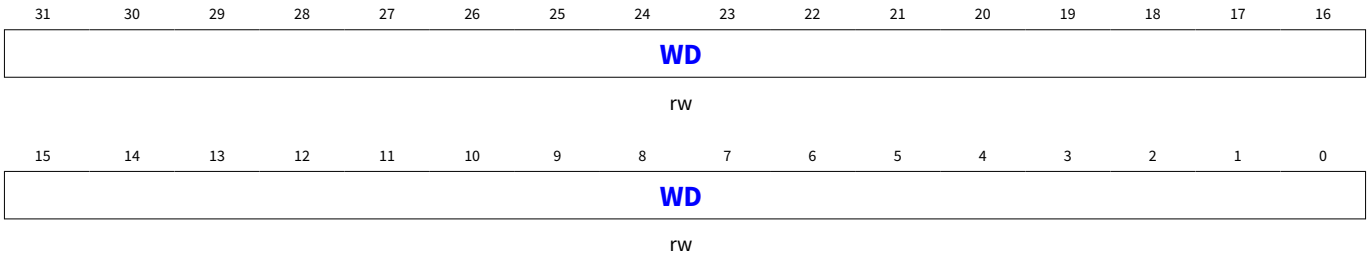
Field	Bits	Type	Description
WID	7:0	rw	**AXI Slave Write Data channel WID Control Signal** This field contains the debug data to be driven on AXI Slave WID signal.
WSTRB	19:16	rw	**AXI Slave Write Data channel WSTRB Control Signal** This field contains the debug data to be driven on AXI Slave WSTRB signal.
0	15:8, 31:20	r	**Reserved** Read as 0; should be written with 0.

14.6.731 DMA AS Sfty WDC Dbg Data

This register contains the data to be driven on AXI Slave write data in Debug Error Injection mode.

DMA_AS_Sfty_WDC_Dbg_Data Offset address: $1F08C_H$

DMA AS Sfty WDC Dbg Data Kernel Reset value: $0000\ 0000_H$



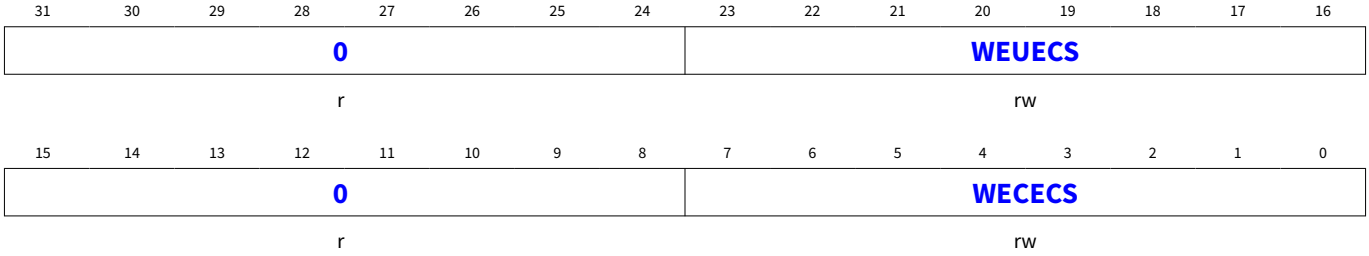
Field	Bits	Type	Description
WD	31:0	rw	**AXI Slave Write Data Channel Debug Data** This field contains the 32-bit debug data to be driven on AXI Slave write Data bus. The DAPS field of DMA_Sfty_Ind_Ctl register indicates which 32-bit word of write data bus this correspond to.

14.6.732 DMA AS Sfty WDC ECC Err Cntr Sts

This register contains the ECC Error counters for write data of AXI Slave Interface safety.

DMA_AS_Sfty_WDC_ECC_Err_Cntr_Sts Offset address: $1F08C_H$

DMA AS Sfty WDC ECC Err Cntr Sts Kernel Reset value: $0000\ 0000_H$



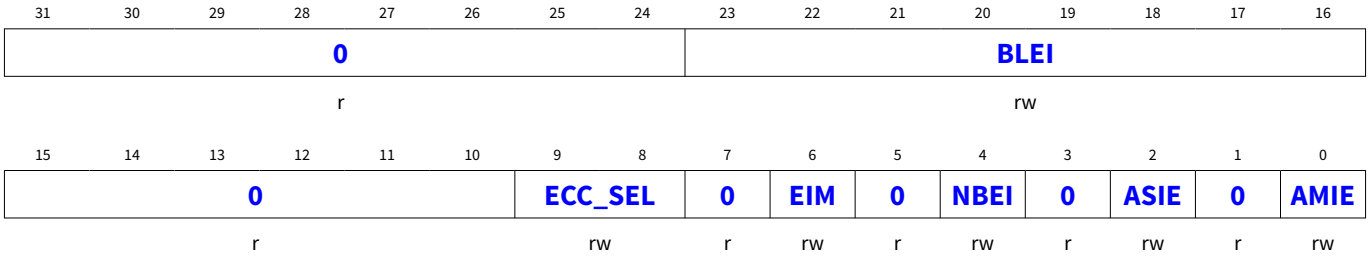
Field	Bits	Type	Description
WECECS	7:0	rw	**AXI Slave Write Data channel ECC Correctable Error Counter** This field provides the number of ECC correctable errors detected in AXI Slave Write Data channel. This field is cleared when the CESS bit is set to 1 and the application reads this field.
WEUECS	23:16	rw	**AXI Slave Write Data channel ECC Uncorrectable Error Counter** This field provides the number of ECC uncorrectable errors detected in AXI Slave Write Data channel. This field is cleared when the CESS bit is set to 1 and the application reads this field.
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

14.6.733 DMA Sfty Dbg EIC

This register contains control signals for Debug Error Injection.

DMA_Sfty_Dbg_EIC Offset address: $1F08C_H$

DMA Sfty Dbg EIC Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
AMIE	0	rw	**AXI Master Insert Error** When this bit is set to 1, parity/ECC/data bit at input of selected AXI Master channel is flipped. The error type and location is based on the EIM/BLEI field of this register. DWC_xgmac clears this bit after the respective bit is flipped.

(table continues...)

(continued)

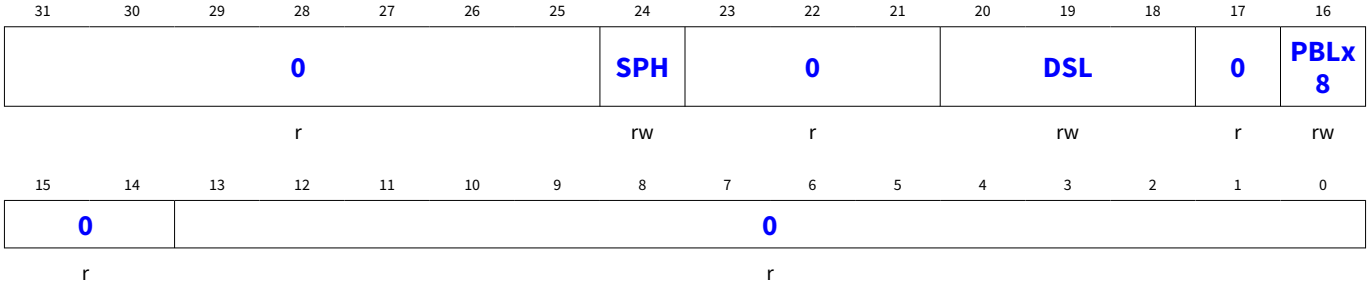
Field	Bits	Type	Description
ASIE	2	rw	**AXI Slave Insert Error - 1: Parity/ECC/data bit at the input of selected AXI Slave channel is flipped** The error type and location is based on the EIM/BLEI field of this register. DWC_xgmac clears this bit after the respective bit is flipped.
NBEI	4	rw	**Number of Bit Error Injection in ECC for AXI Interface** When ASIE or AMIE bit is set, following errors are inserted based on the value encoded in this field: - 0: Inserts 1-bit error - 1: Inserts 2-bit error
EIM	6	rw	**Debug Error Injection Mode - 0: Indicates debug error injected on data - 1: Indicates debug error injected on ECC/Parity Check-bits**
ECC_SEL	9:8	rw	**ECC Select This field selects the ECC to insert error** Depending on the configuration (input data width, ECC granularity) there can be maximum of 4 granules available. Values: - 2'b00: Insert ECC Error in 0th granule - 2'b01: Insert ECC Error in 1st granule - 2'b10: Insert ECC Error in 2nd granule - 2'b11: Insert ECC Error in 3rd granule
BLEI	23:16	rw	**Bit Location of error injection** This field indicates the bit location of DPP/ECC error injection. The EIM bit of DMA_Sfty_Dbg_EIC register indicates whether the error is injected in Data or Parity/ECC Check-bits. The channel/interface selected for debug error injection determines the number of valid bits in this field, for example, for debug error injection on a 64 bit data bus this field can be programmed to a value between 0 and 63, so only bits[21:16] are valid. When 2-bit error injection is enabled, the error is always injected in bit 0 and second error bit is indicated by this field. If the field is also programed to 0 when 2-bit error injection is enabled, DWC_xgmac inserts single bit error at bit 0.
0	1, 3, 5, 7, 15:10, 31:24	r	**Reserved** Read as 0; should be written with 0.

14.6.734 DMA Channel j Control

The DMA Channel Control register controls the common settings of both TX DMA and RX DMA channels.

DMA_CHj_Control (j=0-7) Offset address: $1F100_H+j*80_H$

DMA Channel j Control Kernel Reset value: $0000\ 0000_H$



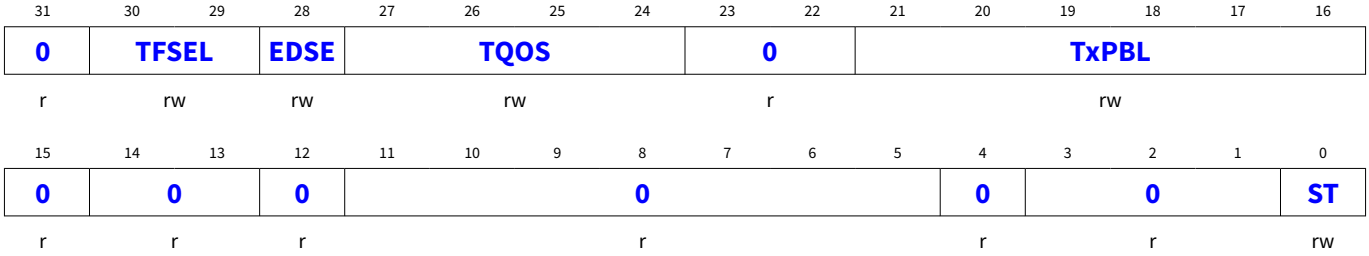
Field	Bits	Type	Description
PBLx8	16	rw	**8xPBL mode** When this bit is set, the PBL value programmed in TxPBL field of DMA_CH0_TX_Control register and RxPBL field of DMA_CH0_RX_Control is multiplied eight times. Therefore, the DMA transfers the data in 8, 16, 32, 64, 128, and 256 beats depending on the PBL value.
DSL	20:18	rw	**Descriptor Skip Length** This bit specifies the number of DWord to skip between two descriptors. The address skipping starts from the end of the current descriptor to the start of the next descriptor. When the DSL value is equal to zero, the DMA takes the descriptor list as contiguous.
SPH	24	rw	**Header-Payload Split** When this bit is set, the DMA splits the header and payload in the Receive path. The DMA writes the header to the Buffer Address1. The DMA writes the payload to the buffer to which the Buffer Address2 is pointing. The software must ensure that the header fits into the Receive buffers. If the header length exceeds the receive buffer size, the DMA does not split the header and payload.
0	13:0, 15:14, 17, 23:21, 31:25	r	**Reserved** Read as 0; should be written with 0.

14.6.735 DMA Channel j Tx Control

The DMA Channel Transmit Control register controls the functions of the TX DMA channel.

DMA_CHj_Tx_Control (j=0-7) Offset address: $1F104_H+j*80_H$

DMA Channel j Tx Control Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
ST	0	rw	**Start or Stop Transmission Command** When this bit is set,the TX DMA channel is placed in the Running state. The DMA starts fetching the transmit descriptor from either of the following locations. - The address pointed by the DMA_CHi_TxDesc_List_LAddress register if this register was updated before the ST bit is set. - The position in the Transmit List at which the DMA channel stopped last when ST bit was changed from 1 to 0. The Start Transmit command is effective only if the TX DMA is already in STOP state. The DMA checks the Transmit list at the current position for a packet to be transmitted. When this bit is reset, the transmission process is placed in the Stopped state after completing the transmission of the current packet. The Next Descriptor position in the Transmit list is saved, and it becomes the current position when the transmission is restarted. To change the list address, program DMA_CHi_TxDesc_List_LAddress register with a new value when this bit is reset. The new value is considered when this bit is set again. The stop transmission command is effective only when the transmission of the current packet is complete or the transmission is in the Suspended state.
TxPBL	21:16	rw	**Transmit Programmable Burst Length** This field controls the maximum size of burst transaction that can be initiated by the TX DMA channel for packet data transfers. The TX DMA channel attempts to fetch the data of this burst size if available, from the Transmit Buffer pointed by the descriptor. You can program PBL with any of the following values: 1, 2, 4, 8, 16, or 32. Any other value results in undefined behavior. To transfer more than 32 beats, perform the following steps: - Set the PBLx8 mode in DMA_CH0_Control register. - Set the PBL. Note-1: This PBL controls the maximum burst size of the requests from this internal DMA engine. The AXI Interface block can split this PBL requests from DMA engine into multiple AXI requests based on the programmed values in DMA_SysBus_Mode register. Note-2: The maximum limit programmed must be less than or equal to half the size of the corresponding Tx Queue depth

(table continues...)

(continued)

Field	Bits	Type	Description
TQOS	27:24	rw	**Transmit QOS** This field is used to drive the values on AXI Master read control channel's QoS indicator or AXI Master write control channel's QoS indicator output signals of the AXI bus for all transactions initiated by this Tx DMA channel
EDSE	28	rw	**Enhanced Descriptor Enable** When this bit is set, the corresponding channel uses enhanced Descriptors with increased size of 32 Bytes for both Normal and Context Descriptors. When reset, the corresponding channel uses the descriptors with default size of 16 Bytes. 0_B **DISABLE**: Enhanced Descriptor is disabled 1_B **ENABLE**: Enhanced Descriptor is enabled
TFSEL	30:29	rw	**TBS Fetch Time Select** Selects the DMA_TBS_CTRLn register used to determine the settings of the TBS function applicable for the packets transferred by this TX DMA channel 00_B **DMA_TBS_CTRL0**: 01_B **DMA_TBS_CTRL1**: 10_B **DMA_TBS_CTRL2**: 11_B **DMA_TBS_CTRL3**:
0	3:1, 4, 11:5, 12, 14:13, 15, 23:22, 31	r	**Reserved** Read as 0; should be written with 0.

14.6.736 DMA Channel j Rx Control

The DMA Channel Receive Control register controls the operating modes and parameters of this RX DMA channel.

DMA_CHj_Rx_Control (j=0-7) Offset address: $1F108_H+j*80_H$

DMA Channel j Rx Control Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RPF	0			RQOS				0		RxPBL					
rw	r			rw				r		rw					

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	RBSZ												0		SR
r	rw												r		rw

Field	Bits	Type	Description
SR	0	rw	**Start or Stop Receive** When this bit is set, the RX DMA moves to the running state and starts to transfer packets from the MTL Rx Queues to the host memory. The RX DMA tries to first acquire RX descriptor from either of the following locations. -The address pointed by the DMA_CH0_RxDesc_List_LAddress register if this register was updated before the SR bit is set. -The position in the Receive List at which the DMA channel stopped last when SR bit was changed from 1 to 0. Rx DMA operation is stopped after the transfer of the current packet. The next descriptor position in the Receive list is saved, and it becomes the current position after the Rx process is restarted. The Stop Receive command is effective only when the Rx process is in the Running (waiting for Rx packet) or Suspended state. Similarly, Start Receive command is effective only when the Receive process is in the Stopped state. So, read the corresponding channel's DMA_CH0_Debug Status register to get the current state of the RxDMA before writing the SR bit.
RBSZ	14:4	rw	**Receive Buffer size** This field indicates the size of the Rx buffers specified in bytes allocated by the software to store the packets the Rx DMA transfers to the host memory. The maximum buffer size is limited to 16K bytes. The buffer size is applicable to payload buffers when split headers are enabled. Note-1: The buffer size must be a multiple of 8 . This is required even if the value of buffer address pointer is not aligned to bus width. If the buffer size is not a multiple of 8, it might result into undefined behavior. The LSB bits (2:0) are ignored and the DMA internally takes the LSB bits as all-zero. Therefore, these LSB bits are read-only (RO). Therefore, the maximum value that can be programmed into this field is 0x3FF8 (16376) . Note-2: It is recommended to program a minimum size of 256 bytes for better performance.

(table continues...)

(continued)

Field	Bits	Type	Description
RxPBL	21:16	rw	**Receive Programmable Burst Length** This field controls the maximum size of burst transaction that can be initiated by the RX DMA channel for packet data transfers. The RX DMA channel attempts to transfer the data of this burst size if available, to the Receive Buffer pointed by the descriptor.You can program PBL with any of the following values: 1, 2, 4, 8, 16, or 32. Any other value results in undefined behavior. It is recommended to use higher values (8 and higher) for better performance. A value of 0 might result in undefined behavior. To transfer more than 32 beats, perform the following steps: - Set the PBLx8 mode in the DMA_CH0_Control register. - Set the PBL. Note-1: This PBL controls the maximum burst size of the requests from this internal DMA engine. The AXI Interface block can split this PBL requests from DMA engine into multiple AXI requests based on the programmed values in DMA_SysBus_Mode register. Note-2: The maximum limit programmed must be less than or equal to half the size of the corresponding Rx Queue Depth of all the Rx Queues from which packets can be services by this RX DMA . For example, in 64-bit configuration, there are 8 bytes per location. So, when Rx Queue Size is 256 bytes, the Rx Queue Depth = 256/8 = 32. The RxPBL must be programmed is less that or equal to total no. of locations in Rx Queue/2, so RxPBL must be programmed less than 32.
RQOS	27:24	rw	**Receive QOS** This field is used to drive values on AXI Master read control channel's QoS indicator or AXI Master write control channel's QoS indicator output signals of AXI bus for all transactions inititated by this Rx DMA Channel.
RPF	31	rw	**Rx DMA Packet Flush** When this bit is set to 1, then DWC_xgmac automatically flushes the packet from the Rx Queues destined to this DMA Rx Channel, when it is stopped or suspended due to unavailability of descriptors. When this bit remains set and the DMA is re-started by the software driver the packets residing in the Rx Queues that were received when the DMA is stopped, still gets flushed out. The packets received by the MAC only after the DMA is re-started is routed to the Rx DMA. The flushing is done on the Read side of the Rx Queue. When the packet is duplicated to multiple Rx DMA, it is flushed by the Rx DMA to which it is routed only when it is stopped or suspended due to unavailability of descriptors to prevent head-of-line blocking. The packet duplicated to multiple Rx DMA is not flushed when the Rx DMA is re-started, irrespective of whether the packet was received during one or more Rx DMAs were stopped. When this bit is set to 0, DWC_xgmac does not flush the packet in the Rx Queue destined to this RxDMA Channel when it is in STOP state. This might cause head-of-line blocking in the corresponding RxQueue.

(table continues...)

(continued)

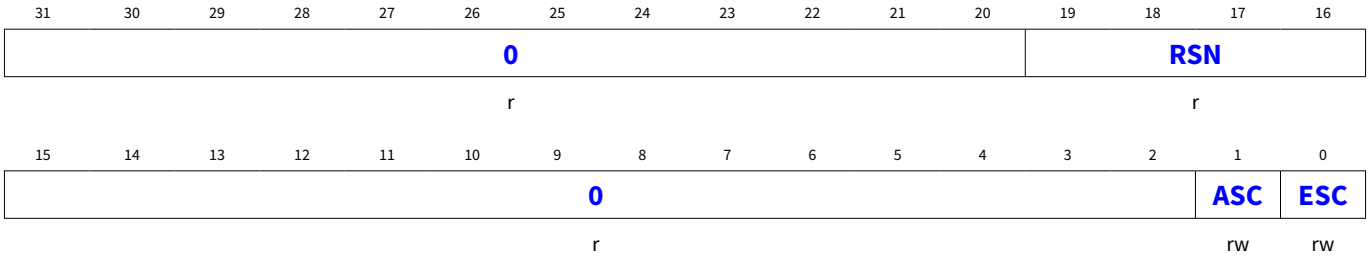
Field	Bits	Type	Description
0	3:1, 15, 23:22, 30:28	r	**Reserved** Read as 0; should be written with 0.

14.6.737 DMA Channel j Slot Function Control Status

This register contains the control and status bits of the AVB slot function in Transmit path.

DMA_CHj_Slot_Function_Control_Status (j=0-7) Offset address: $1F10C_H+j*80_H$

DMA Channel j Slot Function Control Status Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
ESC	0	rw	**Enable Slot Comparison** When set, this bit enables the checking of the slot numbers programmed in the Tx descriptor with the current reference given in the RSN field. The DMA fetches the data from the corresponding buffer only when the RSN value is equal to or less the SLOTNUM value programmed in the descriptor as determined by ASC. When reset, this bit disables the checking of the slot numbers. The DMA fetches the data immediately after the descriptor is processed. 0_B **DISABLE**: Slot Comparison is disabled 1_B **ENABLE**: Slot Comparison is enabled
ASC	1	rw	**Advance Slot Check** When this bit is set, the TX DMA fetches data from the buffer when the RSN value is equal to or less than the SLOTNUM value programmed in that descriptor by up to 2 slots. When this bit is reset, the DMA will start fetching the the data when the RSN value is equal to or less the SLOTNUM value by one slot. This bit is applicable only when the ESC bit is set.
RSN	19:16	r	**Reference Slot Number** This field gives the current value of the reference slot number active in the DMA. It is used for slot comparison with the value programmed in SLOTNUM field of TX Descriptor before the data is fetched.
0	15:2, 31:20	r	**Reserved** Read as 0; should be written with 0.

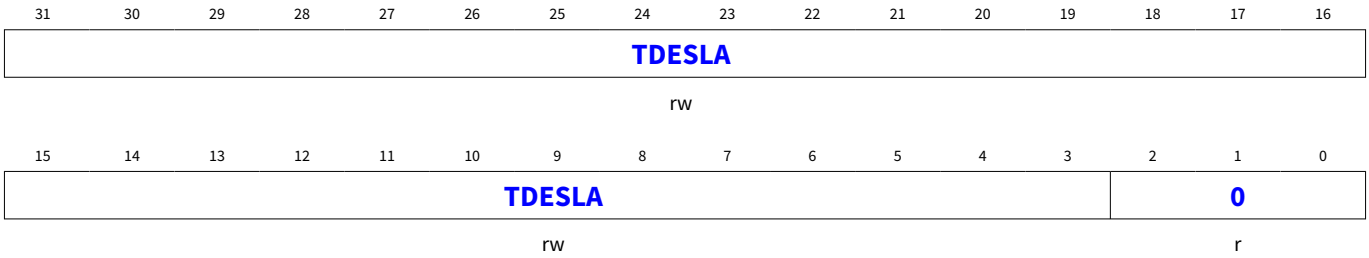
14.6.738 DMA Channel j TxDesc List LAddress

The Tx Descriptor List LAddress register has the lower 32 bits of the start address of the Transmit descriptor list. The descriptor lists reside in the physical memory space of the application and must be aligned to the databus width(Dword -aligned).The DMA internally converts it to bus width aligned address by making the corresponding LSB to low.

You can write to this register only when the Tx DMA has stopped, that is, the ST bit is set to zero in the corresponding DMA_CH0_Tx_Control register. When stopped, this register can be written with a new descriptor list address. When you set the ST bit to 1, the DMA takes the newly-programmed descriptor base address. If this register is not changed when the ST bit is set to 0, the DMA on restart takes the descriptor address where it was stopped earlier.

DMA_CHj_TxDesc_List_LAddress (j=0-7) Offset address: $1F114_H+j*80_H$

DMA Channel j TxDesc List LAddress Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
TDESLA	31:3	rw	**Start of Transmit List** This field contains the base address of the first descriptor in the Transmit descriptor list. The 32b address is formed by extending the TDESLA field with all-zero in the LSB bits below which are RO.
0	2:0	r	**Reserved** Read as 0; should be written with 0.

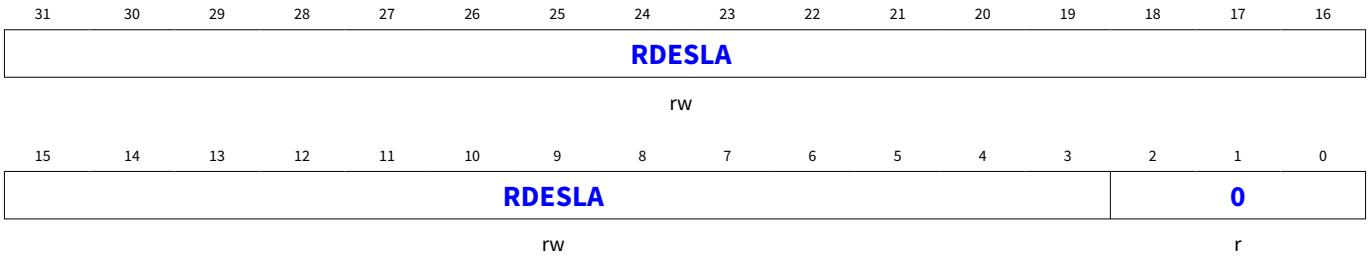
14.6.739 DMA Channel j RxDesc List LAddress

The Rx Descriptor List LAddress register has the lower 32 bits of the start address of the Receive descriptor list. This register points to the start of the Receive Descriptor List. The descriptor lists reside in the physical memory space of the application and must be aligned to the databus width(Dword -aligned). The DMA internally converts it to bus width aligned address by making the corresponding LS bits low. Writing to this register is permitted only when reception is stopped. When stopped, this register must be written to before the receive Start command is given. You can write to this register only when Rx DMA has stopped, that is, SR bit is set to zero in the corresponding DMA_CH0_Rx_Control register. When stopped, this register can be written with a new descriptor list address.

When you set the SR bit to 1, the DMA takes the newly programmed descriptor base address.

DMA_CHj_RxDesc_List_LAddress (j=0-7) Offset address: $1F11C_H+j*80_H$

DMA Channel j RxDesc List LAddress Kernel Reset value: $0000\ 0000_H$



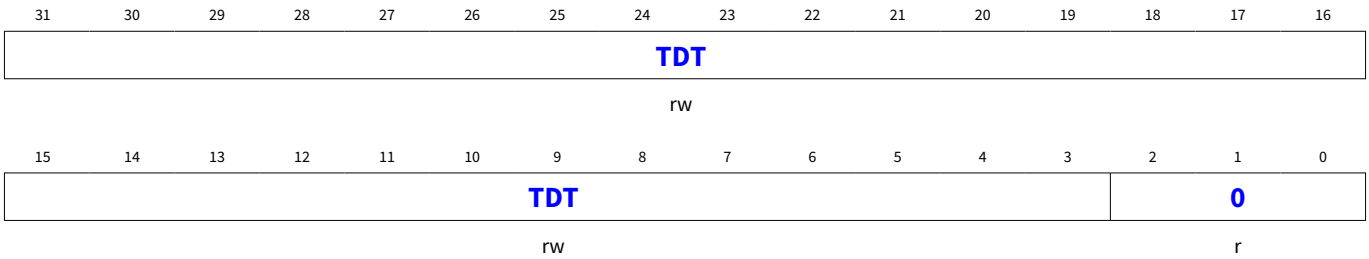
Field	Bits	Type	Description
RDESLA	31:3	rw	**Start of Receive List** This field contains the base address of the first descriptor in the Rx Descriptor list.The 32b address is formed by extending the RDESLA field with all-zero in the LSB bits below which are RO.
0	2:0	r	**Reserved** Read as 0; should be written with 0.

14.6.740 DMA Channel j TxDesc Tail LPointer

The Tx Descriptor Tail LPointer register has the 32 bits end address of the Transmit descriptor list.

DMA_CHj_TxDesc_Tail_LPointer (j=0-7) Offset address: $1F124_H+j*80_H$

DMA Channel j TxDesc Tail LPointer Kernel Reset value: $0000\ 0000_H$



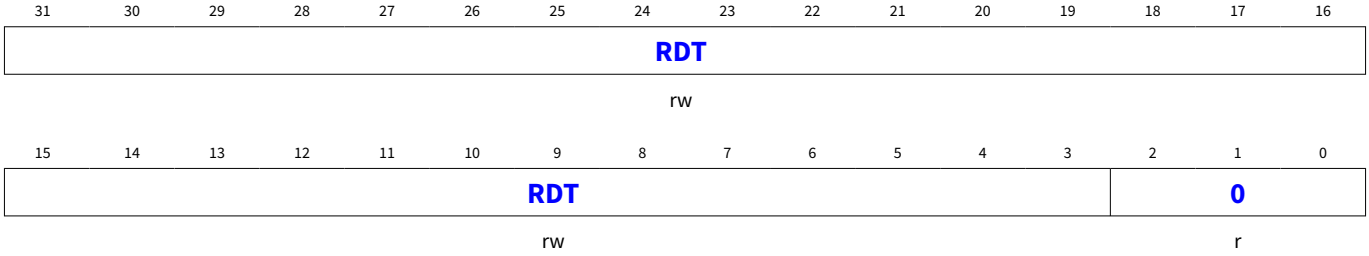
Field	Bits	Type	Description
TDT	31:3	rw	**Transmit Descriptor Tail Pointer** This field contains the tail pointer for the Tx descriptor ring. The software writes the tail pointer to add more descriptors to the Transmit List. DWC_xgmac tries to transmit all packets referenced by the descriptors between the head and the tail pointer registers.The 32b address is formed by extending the TDT field with all-zero in the LSB bits below which are RO.
0	2:0	r	**Reserved** Read as 0; should be written with 0.

14.6.741 DMA Channel j RxDesc Tail LPointer

The Rx Descriptor Tail LPointer register has the 32 bits end address of the Receive descriptor list.

DMA_CHj_RxDesc_Tail_LPointer (j=0-7) Offset address: $1F12C_H+j*80_H$

DMA Channel j RxDesc Tail LPointer Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
RDT	31:3	rw	**Receive Descriptor Tail Pointer** This field contains the tail pointer for the Rx descriptor ring. The software writes the tail pointer to add more descriptors to the Rx channel. DWC_xgmac tries to write all the received packets to the descriptors referenced between the head and the tail pointer registers.The 32b address is formed by extending the RDT field with all-zero in the LSB bits below which are RO.
0	2:0	r	**Reserved** Read as 0; should be written with 0.

14.6.742 DMA Channel j Tx Control2

The Tx_Control2 register contains programmable control to Outstanding read requests per DMA and the length of the Transmit descriptor ring.

DMA_CHj_Tx_Control2 (j=0-7) Offset address: $1F130_H+j*80_H$

DMA Channel j Tx Control2 Kernel Reset value: $0000\ 0000_H$

Field	Bits	Type	Description
TDRL	15:0	rw	**Transmit Descriptor Ring Length** This field sets the maximum number of Tx descriptors in the circular descriptor ring. . - 0 implies ring length of 1 descriptor. - 0xFFFF implies a ring length of 65536 descriptors.

(table continues...)

(continued)

Field	Bits	Type	Description
ORRQ	24	rw	**Read Outstanding Requests** When this is set to 0, the DMA can initiate the maximum number of outstanding read transactions of packet data up to 2 for each Transmit DMA channel. When this field is set to a value greater than zero, the DMA limits the number of outstanding read transactions to the value of ORRQ.
0	23:16, 31:25	r	**Reserved** Read as 0; should be written with 0.

14.6.743 DMA Channel j Rx Control2

The Channel0 Rx Control2 register contains programmable control to Outstanding Write requests and the length of the Receive descriptor circular ring.

DMA_CHj_Rx_Control2 (j=0-7) Offset address: $1F134_H+j*80_H$

DMA Channel j Rx Control2 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0							OWRQ	ARBS							0
r							rw	rw							r

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RDRL															
rw															

Field	Bits	Type	Description
RDRL	15:0	rw	**Receive Descriptor Ring Length** This field sets the maximum number of Rx descriptors in the circular descriptor ring. - 0 implies ring length of 1 descriptor. - 0xFFFF implies a ring length of 65536 descriptors.
ARBS	23:17	rw	**Alternate Receive Buffer Size** Indicates size in bytes for Buffer 1 when ARBS is programmed to a non-zero value (when split header feature is not enabled). When split header feature is enabled, ARBS indicates the buffer size for header data. The maximum alternate buffer is limited to 1016 -bytes . When ARBS=0, Rx Buffer1 and Rx Buffer2 sizes are based on RBSZ field of DMA_CH0_Rx_Control. Width of ARBS field is 7 -bits

(table continues...)

(continued)

Field	Bits	Type	Description
OWRQ	24	rw	**Write Outstanding Requests** When this is set to 0, the DMA can initiate the maximum number of outstanding write transactions of packet data, up to 2 for each Rx DMA channel. When this field is set to a value greater than zero, the DMA limits the number of outstanding write transactions to the value of OWRQ.
0	16, 31:25	r	**Reserved** Read as 0; should be written with 0.

14.6.744 DMA Channel j Interrupt Enable

The Channel0 Interrupt Enable register enables the interrupts reported by the Status register.

DMA_CHj_Interrupt_Enable (j=0-7) Offset address: $1F138_H+j*80_H$

DMA Channel j Interrupt Enable Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NIE	AIE	CDEE	FBEE	0		DDEE	RSE	RBUE	RIE	0			TBUE	TXSE	TIE
rw	rw	rw	rw	r		rw	rw	rw	rw	r			rw	rw	rw

Field	Bits	Type	Description
TIE	0	rw	**Transmit Interrupt Enable** When this bit is set, NIS bit in DMA_CH0_Status register is set to high when TI bit becomes high. When this bit is reset, setting of CDE bit does not affect the TI bit
TXSE	1	rw	**Transmit Stopped Enable** When this bit is set, AIS bit in DMA_CH0_Status register is set to high when TPS bit becomes high. When this bit is reset, setting of TPS bit does not affect the AIS bit.
TBUE	2	rw	**Transmit Buffer Unavailable Enable** When this bit is set, NIS bit in DMA_CH0_Status register is set to high when TBU bit becomes high. When this bit is reset, setting of TBU bit does not affect the NIS bit.
RIE	6	rw	**Receive Interrupt Enable** When this bit is set, NIS bit in DMA_CH0_Status register is set to high when RI bit becomes high. When this bit is reset, setting of RI bit does not affect the NIS bit.

(table continues...)

(continued)

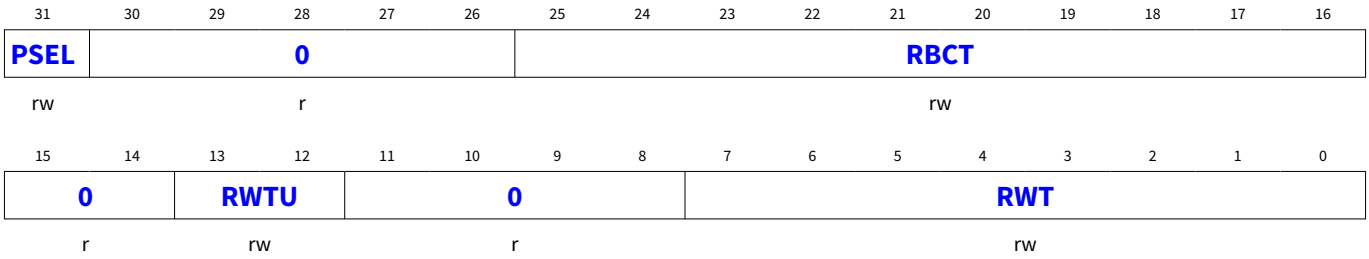
Field	Bits	Type	Description
RBUE	7	rw	**Receive Buffer Unavailable Enable** When this bit is set, AIS bit in DMA_CH0_Status register is set to high when RPS bit becomes high. When this bit is reset, setting of RPS bit does not affect the AIS bit.
RSE	8	rw	**Receive Stopped Enable** When this bit is set along with the AIE bit, the Receive Stopped Interrupt is enabled. When this bit is reset, the Receive Stopped interrupt is disabled.
DDEE	9	rw	**Descriptor Definition Error Enable** When this bit is set, AIS bit in DMA_CH0_Status register is set to high when DDE bit becomes high. When this bit is reset, setting of DDE bit does not affect the AIS bit.
FBEE	12	rw	**Fatal Bus Error Enable** When this bit is set, AIS bit in DMA_CH0_Status register is set to high when FBE bit becomes high. When this bit is reset, setting of FBE bit does not affect the AIS bit.
CDEE	13	rw	**Context Descriptor Error Enable** When this bit is set, AIS bit in DMA_CH0_Status register is set to high when CDE bit becomes high. When this bit is reset, setting of CDE bit does not affect the AIS bit.
AIE	14	rw	**Abnormal Interrupt Summary Enable** When this bit is set,interrupt is activated by XGMAC when AIS bit in DMA_CH0_Status register is set. When this bit is reset, interrupt is not activated due to setting of AIS bit.
NIE	15	rw	**Normal Interrupt Summary Enable** When this bit is set,interrupt is activated by XGMAC when NIS bit in DMA_CH0_Status register is set. When this bit is reset, interrupt is not activated due to setting of NIS bit.
0	5:3, 11:10, 31:16	r	**Reserved** Read as 0; should be written with 0.

14.6.745 DMA Channel j Rx Interrupt Watchdog Timer

The Receive Interrupt Watchdog Timer register controls the generation of Receive Interrupt (RI) from the DMA with respect to a time window. This time-window can be in terms of number of clocks (timer of RWT) or number of transferred bytes or number of transferred packets (counter of RBCT) or a combination of both.

DMA_CHj_Rx_Interrupt_Watchdog_Timer (j=0-7) Offset address: $1F13C_H+j*80_H$

DMA Channel j Rx Interrupt Watchdog Timer Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
RWT	7:0	rw	**Receive Interrupt Watchdog Timer Count** This field control the number of AXI clock cycles, multiplied by factor indicated in RWTU field, for which the watchdog timer is set. The watchdog timer is triggered with the programmed value after the Rx DMA completes the transfer of a packet for which the RI bit is not set in the DMA_CH0_Status register, because the Interrupt Enable bit in the corresponding descriptor RDES3[30] is reset. When the watchdog timer runs out, the RI bit is set and the timer is stopped. The watchdog timer is reset when the RI bit is set high whenever the Interrupt Enable bit (RDES3[30]) is set in descriptor, corresponding to the RX packet transfer or due to the triggering of RBCT counter threshold. When both RWTU and RWT are set to 0, then the Rx Watchdog timer is disabled.
RWTU	13:12	rw	**Receive Interrupt Watchdog Timer Count Units** This fields indicates the number of system clock cycles corresponding to one unit in RWT field. - 2'b00: 256 - 2'b01: 512 - 2'b10: 1024 - 2'b11: 2048 For example, when RWT=2 and RWTU=1, the watchdog timer is set for 2*512=1024 system clock cycles.
RBCT	25:16	rw	**Receive Byte Count Threshold** This field sets the limit in number of bytes or number of RX packets (based on PSEL) transferred without the assertion of Receive Interrupt. Once this limit is crossed, Receive interrupt is asserted, and the watchdog timer (corresponding to RWT) is also reset. This function is valid only when set to a non-zero value. When RBCT = 0, RX Packet/Byte counter is inactive. When PSEL = 0, this field is multiplied by 1024, to get the actual number of byte threshold. When PSEL =1, RBCT value is directly used. Thus the maximum limit is 511 packets or 511 x 1024 bytes respectively.

(table continues...)

(continued)

Field	Bits	Type	Description
PSEL	31	rw	**Packet Count Interrupt Select** When this bit is set, RBCT indicates the number of packet threshold at which the Receive Interrupt (RI) is activated and the watchdog timer reset. When this bit is low, RBCT indicates the number of data byte threshold at which the Receive Interrupt (RI) is activated and the watchdog timer reset. - 1: RBCT field is used to generate interrupt based on number of packets received. - 0: RBCT field is used to generate interrupt based on number of bytes received.
0	11:8, 15:14, 30:26	r	**Reserved** Read as 0; should be written with 0.

14.6.746 DMA Channel j Current App TxDesc L

The Channel Current Application Transmit Descriptor Low register has the lower 32 bits of the address of the Transmit descriptor to be read by the DMA. The upper 32 address bits of the Current Transmit Descriptor are equal to the value written in the DMA_CHi_TxDesc_List_LAddress. Therefore, the upper address bits are not given in the register map.

DMA_CHj_Current_App_TxDesc_L (j=0-7) Offset address: $1F144_H+j*80_H$

DMA Channel j Current App TxDesc L Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CURTDESAPTR															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CURTDESAPTR															
r															

Field	Bits	Type	Description
CURTDESAPTR	31:0	r	**Application Transmit Descriptor Address Pointer** This gives the address of the Descriptor of the next descriptor that is going to be fetched from system memory.

14.6.747 DMA Channel j Current App RxDesc L

The Channel Current Application Receive Descriptor Low register has the lower 32 bits of the address of the Receive descriptor to be read by the DMA. The upper 32 address bits of the Current Receive Descriptor are equal to the value written in the DMA_CH0_RxDesc_List_LAddress. Therefore, the upper address bits are not given in the register map.

DMA_CHj_Current_App_RxDesc_L (j=0-7) Offset address: $1F14C_H+j*80_H$

DMA Channel j Current App RxDesc L Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CURRDESAPTR															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CURRDESAPTR															
r															

Field	Bits	Type	Description
CURRDESAPTR	31:0	r	**Application Receive Descriptor Address Pointer** This gives the address of the next descriptor that is going to be fetched from system memory.

14.6.748 DMA Channel j Current App TxBuffer L

The Channel Current Application Transmit Buffer Address Low register has the lower 32 bits of the current address of the Transmit buffer address read by the DMA.

DMA_CHj_Current_App_TxBuffer_L (j=0-7) Offset address: $1F154_H+j*80_H$

DMA Channel j Current App TxBuffer L Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CURTBUFAPTR															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CURTBUFAPTR															
r															

Field	Bits	Type	Description
CURTBUFAPTR	31:0	r	**Application Transmit Buffer Address Pointer** This field is set to the value of the last data buffer address specified in Transmit Descriptor that DMA put read request on to the AXI interface.

14.6.749 DMA Channel j Current App RxBuffer L

The Channel0 Current Application Receive Buffer Address Low register has the lower 32 bits of the current address of the Receive buffer address read by the DMA.

DMA_CHj_Current_App_RxBuffer_L (j=0-7) Offset address: $1F15C_H+j*80_H$

DMA Channel j Current App RxBuffer L Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CURRBUFAPTR															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CURRBUFAPTR															
r															

Field	Bits	Type	Description
CURRBUFAPTR	31:0	r	**Application Receive Buffer Address Pointer** This field is set to the value of the last data buffer address specified in Receive Descriptor that DMA put write request on to the AXI interface.

14.6.750 DMA Channel j Status

The software driver (application) reads the Status register during interrupt service routine or polling to determine the status of the DMA.

DMA_CHj_Status (j=0-7) Offset address: $1F160_H+j*80_H$

DMA Channel j Status Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0										REB			TEB		
r										rw			rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NIS	AIS	CDE	FBE	0		DDE	RPS	RBU	RI	0			TBU	TPS	TI
rw	rw	rw	rw	r		rw	rw	rw	rw	r			rw	rw	rw

Field	Bits	Type	Description
TI	0	rw	**Transmit Interrupt** This bit when set, indicates that transfer of packet to the TxQueue is completed. This is set only when the corresponding Tx packet descriptor's IOC bit (TDES2[31]) is set. This is a sticky bit and is cleared only when a "1" is written to this bit
TPS	1	rw	**Transmit Process Stopped** This bit when set, indicates that the Transmit DMA process has entered STOP state. This is a sticky bit and is cleared only when a "1" is written to this bit
TBU	2	rw	**Transmit Buffer Unavailable** This bit when set, indicates that the application owns the next descriptor in the Transmit Descriptor list, and the DMA cannot acquire it and hence is suspended. To resume the transmission, the application must do the following: - Change the ownership of the descriptor by setting Bit 31 of TDES0. - Issue a Transmit Poll Demand command by performing a write to the Transmit Descriptor Tail Pointer register. This is a sticky bit and is cleared only when a '1' is written to this bit.

(table continues...)

(continued)

Field	Bits	Type	Description
RI	6	rw	**Receive Interrupt** This bit when set, indicates that the transfer of Rx Packet to host memory is complete. This bit is set only if either of the following is true: - Corresponding Rx descriptor's IOC bit (RDES3[30]) is set - The Rx Watchdog timer or counter reaches limit set in the RWT or RBCT fields of DMA_CH0_Rx_Interrupt_Watchdog_Timer register. This is a sticky bit and is cleared only when a '1' is written to this bit.
RBU	7	rw	**Receive Buffer Unavailable** This bit when set, indicates that the software application owns the next descriptor in the Receive Descriptor list, and the DMA cannot acquire it. The Rx process is suspended. To resume the Rx process, the application must change the ownership of the descriptor and advance the DMA_CH0_RxDesc_Tail_LPointer register. If this is not done, the RxDMA remains in SUSPEND state and results in overflow of the corresponding RxQueue.This is a sticky bit and is cleared only when a 1 is written to this bit
RPS	8	rw	**Receive Process Stopped** This bit is asserted when the Rx DMA process enters the Stopped state.This is a sticky bit and is cleared only when a "1" is written to this bit.
DDE	9	rw	**Descriptor Definition Error** This bit when set, indicates that a Descriptor Definition error (Descriptor content is all-ones) occurred. When this bit is set, the corresponding DMA channel engine stops Descriptor Fetch. This is a sticky bit and is cleared only when a "1" is written to this bit. The recovery is only by issuing a Software Reset, that is setting SWR bit of DMA_Mode register, followed by reconfiguration and restarting.
FBE	12	rw	**Fatal Bus Error** This bit when set, indicates that a bus error occurred (as described in the REB and TEB fields). When this bit is set, the corresponding DMA channel engine disables all subsequent transactions and proceeds to STOP state. This is a sticky bit and is cleared only when a "1" is written to this bit.
CDE	13	rw	**Context Descriptor Error** This bit when set, indicates that the DMA Tx engine received a context descriptor in the middle of a packet (in an intermediate descriptor), and the DMA Tx engine ignored it.This is a sticky bit and is cleared only when a "1" is written to this bit.

(table continues...)

(continued)

Field	Bits	Type	Description
AIS	14	rw	**Abnormal Interrupt Summary** Abnormal Interrupt Summary bit value is the logical OR of the following when the corresponding interrupt enable bits are set in the DMA_CH0_Interrupt_Enable register: - Bit 1: Transmit Process Stopped - Bit 7: Receive Buffer Unavailable - Bit 8: Receive Process Stopped - Bit 9: Descriptor Definition Error - Bit 12: Fatal Bus Error - Bit 13: Context Descriptor Error Only unmasked bits affect the Abnormal Interrupt Summary bit. This is a sticky bit. You must clear this bit (by writing 1 to this bit) each time a corresponding bit, which causes AIS to be set, is cleared.
NIS	15	rw	**Normal Interrupt Summary** Normal Interrupt Summary bit value is the logical OR of the following bits when the corresponding interrupt enable bits are set in the DMA_CH0_Interrupt_Enable register: - Bit 0: Transmit Interrupt - Bit 2: Transmit Buffer Unavailable - Bit 6: Receive Interrupt Only unmasked bits (interrupts for which interrupt enable is set in DMA_CH0_Interrupt_Enable register) affect the Normal Interrupt Summary bit. This is a sticky bit. You must clear this bit (by writing 1 to this bit) each time a corresponding bit which causes NIS to be set is cleared.
TEB	18:16	rw	**Tx DMA Bus Error** This field indicates the type of Tx DMA transaction that resulted in Bus error response on the AXI interface. - 000 : No Bus Error response - 100 : Bus Error during data buffer write - 110 : Bus Error during Descriptor write - 111 : Bus Error during Descriptor read - others : Reserved This field is valid only when the FBE bit is set. This field does not generate an interrupt.
REB	21:19	rw	**Rx DMA Bus Error** This field indicates the type of Rx DMA transaction that resulted in a Bus error response on the AXI interface. - 000 : No Bus Error response - 100 : Bus Error during data buffer write - 110 : Bus Error during Descriptor write - 111 : Bus Error during Descriptor read - others : Reserved This field is valid only when the FBE bit is set. This field does not generate an interrupt.

(table continues...)

(continued)

Field	Bits	Type	Description
0	5:3, 11:10, 31:22	r	**Reserved** Read as 0; should be written with 0.

14.6.751 DMA Channel j Debug Status

The DMA Channel Debug Status register gives the current status of the various state machines in the TX and RX DMA channel.

DMA_CHj_Debug_Status (j=0-7) Offset address: $1F164_H+j*80_H$

DMA Channel j Debug Status Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	RDWS			RDTS									RDFS		
r	r			r									r		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	TDWS			TDTS				TDRS		TDXS			TDFS		
r	r			r				r		r			r		

Field	Bits	Type	Description
TDFS	2:0	r	**TxDMA Descriptor Fetch FSM State** The encoding for this fields is as below: - 'x0: STOP/IDLE State - 'x1: Requesting for Descriptor Read state - 'x2: Reading the Descriptor and writing into Descriptor Cache state - 'x3: SUSPEND state - 'x4: ERROR state - 'x5: Waiting for Descriptor data transfer complete after OWN bit error - 'x6: Start Read descriptor with OWN bit not set again - 'x7: Reserved
TDXS	5:3	r	**TxDMA Data Transfer FSM State** The encoding for this fields is as below: - 'x0: IDLE state - 'x1: Control word writing state - 'x2: Data transferring state - 'x3: Descriptor Writing Back state - 'x4: Writing Timestamp correction field state - 'x5-'x7: Reserved

(table continues...)

(continued)

Field	Bits	Type	Description
TDRS	7:6	r	**TxDMA Data Request FSM State** The encoding for this fields is as below: - 'x0: IDLE state - 'x1: Checking Tx FIFO space state - 'x2: AXI cache request state - 'x3: Internal Request information updating state
TDTS	11:8	r	**TxDMA Data Transfer Control Module FSM State** The encoding for this fields is as below: - 'x0: STOP state - 'x1: IDLE state - 'x2: Buffer Reading request - 'x3: Internal normal control word writing - 'x4: ERROR state - 'x5: Internal context control word writing - 'x6: Internal Timestamp correction updating - 'x7-'xF: Others
TDWS	14:12	r	**TxDMA Descriptor Write FSM state** The encoding for this fields is as below: - 'x0: STOP/IDLE state - 'x1: Request to Arbiter for Descriptor Write state - 'x2: Descriptor Write Transfer state - 'x3: Waiting for AXI Write Response state - 'x4: ERROR state - 'x5-'x7: Reserved
RDFS	18:16	r	**RxDMA Descriptor Fetch FSM State** The encoding for this fields is as below: - 'x0: STOP/IDLE State - 'x1: Requesting for Descriptor Read - 'x2: Reading the Descriptor and writing into Descriptor Cache - 'x3: SUSPEND state - 'x4: ERROR state - 'x5: Waiting for Descriptor data transfer complete after OWN bit error - 'x6: Start Read descriptor with OWN bit not set again - 'x7: Reserved

(table continues...)

(continued)

Field	Bits	Type	Description
RDTS	27:19	r	**RxDMA Data Transfer FSM State** The encoding for this fields is as below: - 'x0: STOP state - 'x1: waiting for the New Descriptor from RDF state - 'x2: Waiting for Packet Header Status from ARI state - 'x3: Waiting for new intermediate Descriptor from RDF state - 'x4: Waiting for Watermark on ARI state - 'x5: Request for Data Transfer on AXI state - 'x6: Write Data Transfer on AXI state - 'x7: Write Dummy Data after receiving AXI Error state - 'x8: Waiting for previous descriptor to closed by RDW state - 'x9: Error state - 'xA-'xF: Reserved
RDWS	30:28	r	**RxDMA Descriptor Write FSM state** The encoding for this fields is as below: - 'x0: STOP/IDLE state - 'x1: Request to Arbiter for Descriptor Write state - 'x2: Descriptor Write Transfer state - 'x3: Waiting for AXI Write Response state - 'x4: ERROR state - 'x5-'x7: Reserved
0	15, 31	r	**Reserved** Read as 0; should be written with 0.

14.6.752 DMA Channel j Desc Mem Cache Fill Level

The DMA Channel Descriptor Cache Fill Level Status register gives the fill level of the descriptor prefetch cache of the TX and RX DMA engines.

DMA_CHj_Desc_Mem_Cache_Fill_Level (j=0-7) Offset address: $1F168_H+j*80_H$

DMA Channel j Desc Mem Cache Fill Level Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0									RX_FILL_LVL						
r									r						

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0									TX_FILL_LVL						
r									r						

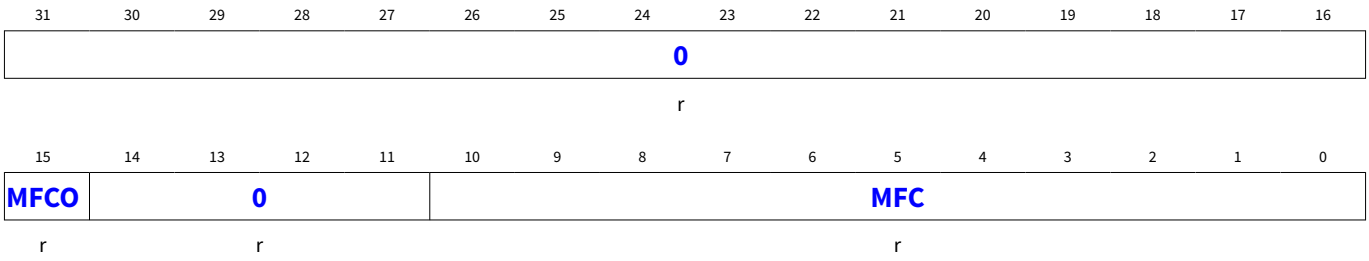
Field	Bits	Type	Description
TX_FILL_LVL	6:0	r	**TxDMA Descriptor Cache Fill Level** This field gives the fill level of the Descriptor Cache for TxDMA channel. A value of 0 implies empty and a value equal to size of Descriptor Cache implies it is full.
RX_FILL_LVL	22:16	r	**RxDMA Descriptor Cache Fill Level** This field gives the fill level of the Descriptor Cache for RxDMA Channel. A value of 0 implies empty and a value equal to size of Descriptor Cache implies it is full.
0	15:7, 31:23	r	**Reserved** Read as 0; should be written with 0.

14.6.753 DMA Channel j Miss Packet Cnt

This register has the number of the packets that are dropped by the Receive DMA either due to Bus Error or due to programming the RPF field in the DMA_CH0_Rx_Control register.

DMA_CHj_Miss_Packet_Cnt (j=0-7) Offset address: $1F16C_H+j*80_H$

DMA Channel j Miss Packet Cnt Kernel Reset value: $0000\ 0000_H$



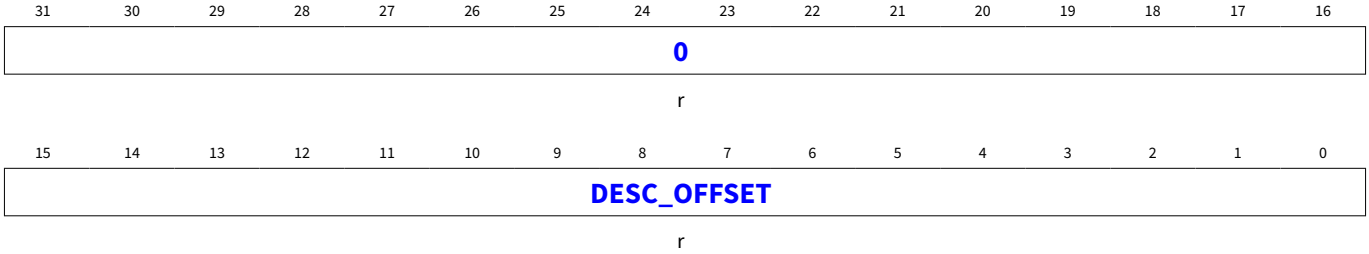
Field	Bits	Type	Description
MFC	10:0	r	**Missed Packet Counter** This counter indicates the number of packet dropped by the DMA either due to Bus Error or due to programming RPF field in DMA_CH0_Rx_Control register. (Refer RPF field of DMA_CH0_Rx_Control register for more details.) The counter gets cleared when this register is read.
MFCO	15	r	**Overflow status of MFC Counter** When this bit is set, the MFC counter reached the maximum value and wrapped around to 0. The MFC counter does not get incremented further. This bit gets cleared when this register is read.
0	14:11, 31:16	r	**Reserved** Read as 0; should be written with 0.

14.6.754 DMA Channel j Tx Data Xfer Ring Offset

This register indicates the Offset of the Tx Descriptor List, for which the DMA engine is fetching the data.

DMA_CHj_Tx_Data_Xfer_Ring_Offset (j=0-7) Offset address: $1F170_H+j*80_H$

DMA Channel j Tx Data Xfer Ring Offset Kernel Reset value: $0000\ 0000_H$



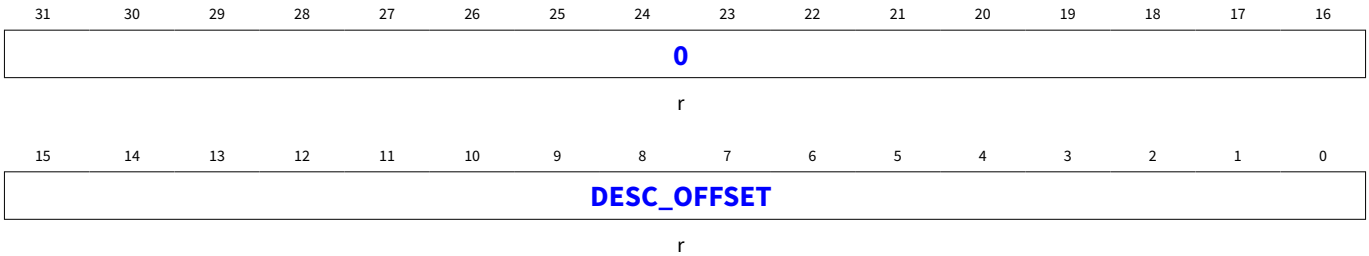
Field	Bits	Type	Description
DESC_OFFSET	15:0	r	**TxDMA channel Data Transfer ring Offset** In DMA, this field indicates the ring offset of the Descriptor DMA currently working on data transmission.
0	31:16	r	**Reserved** Read as 0; should be written with 0.

14.6.755 DMA Channel j Rx Data Xfer Ring Offset

The DMA Channel Receive Data Transfer Ring Offset register indicates the Offset of the Rx Descriptor List, for which the DMA engine is transferring the data.

DMA_CHj_Rx_Data_Xfer_Ring_Offset (j=0-7) Offset address: $1F174_H+j*80_H$

DMA Channel j Rx Data Xfer Ring Offset Kernel Reset value: $0000\ 0000_H$



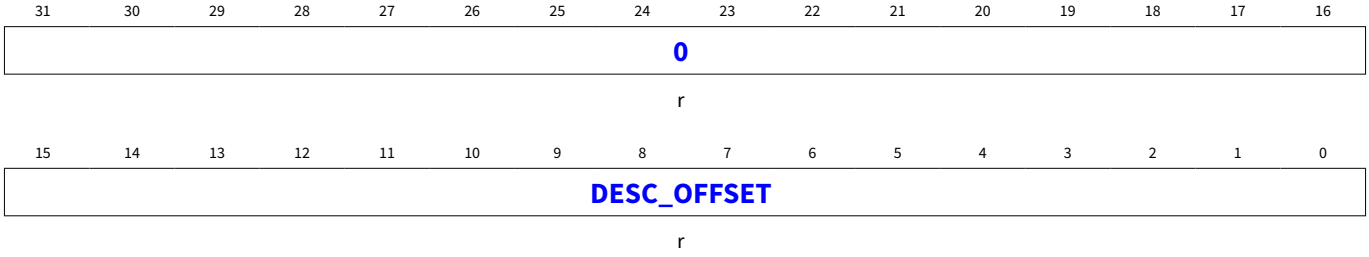
Field	Bits	Type	Description
DESC_OFFSET	15:0	r	**RxDMA Channel Data Transfer Ring Offset** In DMA, this field indicates the ring offset of the Descriptor DMA currently working on data reception.
0	31:16	r	**Reserved** Read as 0; should be written with 0.

14.6.756 DMA Channel j Tx Desc Write Ring Offset

This register indicates the Offset of the Tx Descriptor List, for which the DMA engine is closing the descriptor.

DMA_CHj_Tx_Desc_Write_Ring_Offset (j=0-7) Offset address: $1F178_H+j*80_H$

DMA Channel j Tx Desc Write Ring Offset Kernel Reset value: $0000\ 0000_H$



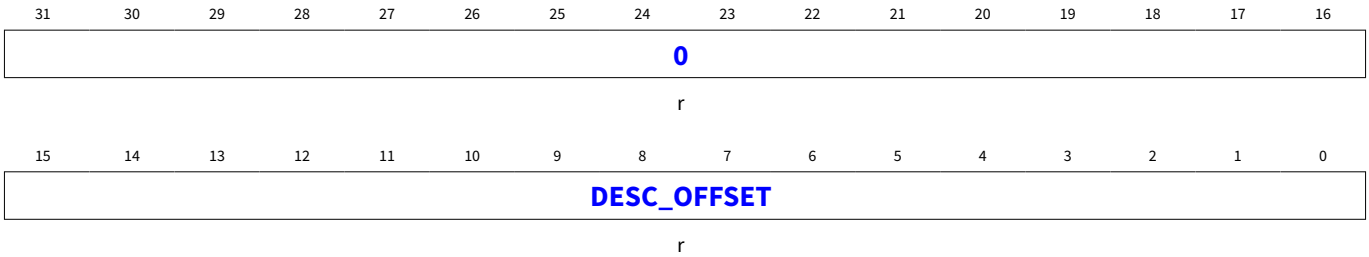
Field	Bits	Type	Description
DESC_OFFSET	15:0	r	**TxDMA Channel Descriptor Write Ring Offset** This field indicates the ring offset of the descriptor DMA posted write-back request.
0	31:16	r	**Reserved** Read as 0; should be written with 0.

14.6.757 DMA Channel j Rx Desc Write Ring Offset

This register indicates the Offset of the Rx Descriptor List, for which the DMA engine is closing the descriptor.

DMA_CHj_Rx_Desc_Write_Ring_Offset (j=0-7) Offset address: $1F17C_H+j*80_H$

DMA Channel j Rx Desc Write Ring Offset Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
DESC_OFFSET	15:0	r	**RxDMA Channel Descriptor Write Ring Offset** This field indicates the ring offset of the descriptor DMA posted write-back request.
0	31:16	r	**Reserved** Read as 0; should be written with 0.

14.7 Debug information

This section is not applicable for the module being described.

14.8 References

1. IEEE802.3-2012 IEEE Standard for Ethernet
2. IEEE 802.1Qbu-2016 Frame Preemption
3. IEEE 802.1Qbv-2015 Enhancements for Scheduled Traffic
4. IEEE 802.3br-2016 Specification and Management Parameters for Interspersing Express Traffic
5. IEEE 802.1Qat Stream Reservation Protocol
6. IEEE 802.1Qci – Per-Stream Filtering and Policing
7. IEEE 802.1Qav-2009, 802.1AS-2020 Standards for Audio Video (AV) traffic
8. IEEE Standard 802.3az-2010 Standard for Energy Efficient Ethernet (EEE)
9. AMBA4 AXI and ACE protocol specification, February 2013, ARM Ltd for AXI interface
10. AMBA3 APB Protocol Specification, v1.0, ARM Ltd for APB3 interface
11. RFC 768, User Datagram Protocol (UDP), DARPA Internet Program, August 1980
12. RFC 791, Protocol Specification (Internet Protocol, Version 4 (IPv4)
13. IEEE 802.1Qaz EEE Virtual Bridged Local Area

14.9 GETH revision history

Initial release of the chapter.

14.10 TC4Dx GETH information

14.10.1 TC4Dx GETH configuration

Table 777 Device specific parameters

Parameter	GETH module
Number of MACs	2
Data rate	10M/100M - Full duplex and Half duplex 1G/2.5G (by GMII only) /5G - Full duplex only
Bridge	Yes
Number of TxQ per MAC	8
TxQ memory size per MAC	32 KBytes
Number of RxQ per MAC	8
Rx memory size per MAC	32 KBytes
Number of Rx DMA Channels	8
Number of Tx DMA Channels	8
Number of MAC filters per MAC	32
Number of VLAN filters per MAC	8
Number of Layer 3 and Layer 4 filters	8
Number of Traffic Classes (TC)	8 (TC0-7)
CBS Enabled Traffic Classes	TC5, TC6 and TC7
PHY interfaces	MII, RMII, RGMII and SGMII (UXSGMII for 5G mode)
Maximum Entries of Flexible Receive Parser (FRP) Instruction Table	256
Maximum FRP Parsable Bytes from Start of Packet	256 Bytes

14.10.2 TC4Dx GETH features

There are no deviations from the generic specification.

14.10.3 TC4Dx GETH functional description

There are no deviations from the generic specification.

14.10.4 TC4Dx GETH registers

14.10.4.1 Memory overview tables of GETH

Table 778 Memory overview - GETH (ascending address)

Short name	Long name	Address
GETH0_Memory	(10000_H Byte)	$F9020000_H$

14.10.4.2 Register address space - GETH

Table 779 Registers address space - GETH

Module	Base address	End address	Note
GETH0	$F9000000_H$	$F902FFFF_H$	

14.10.4.3 Register overview - access mode glossary

Table 780 Register overview - access mode glossary

Keyword	Description
SE	Access protection using PROT register GETH0_PROTSE .
APU-PCHy (y=0-7)	Protection group consisting of registers GETH0_ACCENCHy_WRA , GETH0_ACCENCHy_WRB , GETH0_ACCENCHy_RDA , GETH0_ACCENCHy_RDB , GETH0_ACCENCHy_VM , GETH0_ACCENCHy_PRS .
PCHy	Access protection using *APU-PCHy* registers.
APU-PG	Protection group consisting of registers GETH0_ACCENGLOBAL_WRA , GETH0_ACCENGLOBAL_WRB , GETH0_ACCENGLOBAL_RDA , GETH0_ACCENGLOBAL_RDB , GETH0_ACCENGLOBAL_VM , GETH0_ACCENGLOBAL_PRS .
PG	Access protection using *APU-PG* registers.
APU-PMy (y=0-1)	Protection group consisting of registers GETH0_ACCENMACy_WRA , GETH0_ACCENMACy_WRB , GETH0_ACCENMACy_RDA , GETH0_ACCENMACy_RDB , GETH0_ACCENMACy_VM , GETH0_ACCENMACy_PRS .
PMy	Access protection using *APU-PMy* registers.
BE	Always returns a Bus Error.
32	Access only when using 32-bit width.
PROT	Access restrictions as defined in the PROT register access rules.

14.10.4.4 Registers overview - GETH0 (ascending offset address)

Table 781 Registers overview - GETH0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
GETH0_CLC	Clock control register	00000_H	PG	PG, SV	Application Reset	1642

(table continues...)

Table 781 (continued) Registers overview - GETH0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
GETH0_ID	Module identification register	00008_H	PG	BE	PowerOn Reset	1642
GETH0_RST_CTRLA	Reset control register A	$0000C_H$	PG	PG, SV	Application Reset	1643
GETH0_RST_CTRLB	Reset Control Register B	00010_H	PG	PG, SV	Application Reset	1644
GETH0_RST_STAT	Reset status register	00014_H	PG	BE	Application Reset	1644
GETH0_MACEN	Clock control register for MAC	00018_H	PG	PG, SV	Application Reset	1645
GETH0_PROTSE	PROT register safe endinit	00024_H	32	SV, 32, PROT	Application Reset	1646
GETH0_ACCENGLOBAL_WRA	Write access enable register A	00040_H	32	SE, SV, 32	Application Reset	1647
GETH0_ACCENGLOBAL_WRB	Write access enable register B	00044_H	32	SE, SV, 32	Application Reset	1648
GETH0_ACCENGLOBAL_RDA	Read access enable register A	00048_H	32	SE, SV, 32	Application Reset	1648
GETH0_ACCENGLOBAL_RDB	Read access enable register B	$0004C_H$	32	SE, SV, 32	Application Reset	1649
GETH0_ACCENGLOBAL_VM	VM access enable register	00050_H	32	SE, SV, 32	Application Reset	1649
GETH0_ACCENGLOBAL_PRS	PRS access enable register	00054_H	32	SE, SV, 32	Application Reset	1650
GETH0_ACCENMACy_WRA (y=0-1)	MAC y Write access enable register A	00060_H+ y*20_H	32	SE, SV, 32	Application Reset	1651
GETH0_ACCENMACy_WRB (y=0-1)	MAC y Write access enable register B	00064_H+ y*20_H	32	SE, SV, 32	Application Reset	1651
GETH0_ACCENMACy_RDA (y=0-1)	MAC y Read access enable register A	00068_H+ y*20_H	32	SE, SV, 32	Application Reset	1652
GETH0_ACCENMACy_RDB (y=0-1)	MAC y Read access enable register B	$0006C_H$+ y*20_H	32	SE, SV, 32	Application Reset	1652
GETH0_ACCENMACy_VM (y=0-1)	MAC y VM access enable register	00070_H+ y*20_H	32	SE, SV, 32	Application Reset	1653
GETH0_ACCENMACy_PRS (y=0-1)	MAC y PRS access enable register	00074_H+ y*20_H	32	SE, SV, 32	Application Reset	1653

(table continues...)

Table 781 (continued) Registers overview - GETH0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
GETH0_ACCENCHy_WRA (y=0-7)	CH y Write access enable register A	$000A0_H$+y*20_H	32	SE, SV, 32	Application Reset	1654
GETH0_ACCENCHy_WRB (y=0-7)	CH y Write access enable register B	$000A4_H$+y*20_H	32	SE, SV, 32	Application Reset	1654
GETH0_ACCENCHy_RDA (y=0-7)	CH y Read access enable register A	$000A8_H$+y*20_H	32	SE, SV, 32	Application Reset	1655
GETH0_ACCENCHy_RDB (y=0-7)	CH y Read access enable register B	$000AC_H$+y*20_H	32	SE, SV, 32	Application Reset	1655
GETH0_ACCENCHy_VM (y=0-7)	CH y VM access enable register	$000B0_H$+y*20_H	32	SE, SV, 32	Application Reset	1656
GETH0_ACCENCHy_PRS (y=0-7)	CH y PRS access enable register	$000B4_H$+y*20_H	32	SE, SV, 32	Application Reset	1657
GETH0_Portj_MAC_Tx_Configuration (j=0-1)	Port j MAC Tx Configuration	10000_H+j*2000_H	PMj	PMj	Kernel Reset	1657
GETH0_Portj_MAC_Rx_Configuration (j=0-1)	Port j MAC Rx Configuration	10004_H+j*2000_H	PMj	PMj	Kernel Reset	1661
GETH0_Portj_MAC_Packet_Filter (j=0-1)	Port j MAC Packet Filter	10008_H+j*2000_H	PMj	PMj	Kernel Reset	1664
GETH0_Portj_MAC_WD_JB_Timeout (j=0-1)	Port j MAC WD JB Timeout	$1000C_H$+j*2000_H	PMj	PMj	Kernel Reset	1667
GETH0_Portj_MAC_VLAN_Tag_Ctrl (j=0-1)	Port j MAC VLAN Tag Ctrl	10050_H+j*2000_H	PMj	PMj	Kernel Reset	1668
GETH0_Portj_MAC_VLAN_Tag_Data (j=0-1)	Port j MAC VLAN Tag Data	10054_H+j*2000_H	PMj	PMj	Kernel Reset	1671

(table continues...)

Table 781 (continued) Registers overview - GETH0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
GETH0_Portj_MAC_VLAN_Tag_Filter0 (j=0-1)	Port j MAC VLAN Tag Filter0	10054_H+j *2000_H	PMj	PMj	Kernel Reset	1672
GETH0_Portj_MAC_VLAN_Tag_Filter1 (j=0-1)	Port j MAC VLAN Tag Filter1	10054_H+j *2000_H	PMj	PMj	Kernel Reset	1673
GETH0_Portj_MAC_VLAN_Tag_Filter2 (j=0-1)	Port j MAC VLAN Tag Filter2	10054_H+j *2000_H	PMj	PMj	Kernel Reset	1675
GETH0_Portj_MAC_VLAN_Tag_Filter3 (j=0-1)	Port j MAC VLAN Tag Filter3	10054_H+j *2000_H	PMj	PMj	Kernel Reset	1676
GETH0_Portj_MAC_VLAN_Tag_Filter4 (j=0-1)	Port j MAC VLAN Tag Filter4	10054_H+j *2000_H	PMj	PMj	Kernel Reset	1678
GETH0_Portj_MAC_VLAN_Tag_Filter5 (j=0-1)	Port j MAC VLAN Tag Filter5	10054_H+j *2000_H	PMj	PMj	Kernel Reset	1679
GETH0_Portj_MAC_VLAN_Tag_Filter6 (j=0-1)	Port j MAC VLAN Tag Filter6	10054_H+j *2000_H	PMj	PMj	Kernel Reset	1681
GETH0_Portj_MAC_VLAN_Tag_Filter7 (j=0-1)	Port j MAC VLAN Tag Filter7	10054_H+j *2000_H	PMj	PMj	Kernel Reset	1682
GETH0_Portj_MAC_VLAN_Hash_Table (j=0-1)	Port j MAC VLAN Hash Table	10058_H+j *2000_H	PMj	PMj	Kernel Reset	1684
GETH0_Portj_MAC_VLAN_Incl (j=0-1)	Port j MAC VLAN Incl	10060_H+j *2000_H	PMj	PMj	Kernel Reset	1684
GETH0_Portj_MAC_VLAN_Incl0 (j=0-1)	Port j MAC VLAN Incl0	10060_H+j *2000_H	PMj	PMj	Kernel Reset	1686

(table continues...)

Table 781 (continued) Registers overview - GETH0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
GETH0_Portj_MAC_VLAN_Incl1 (j=0-1)	Port j MAC VLAN Incl1	10060_H+j *2000_H	PMj	PMj	Kernel Reset	1687
GETH0_Portj_MAC_VLAN_Incl2 (j=0-1)	Port j MAC VLAN Incl2	10060_H+j *2000_H	PMj	PMj	Kernel Reset	1688
GETH0_Portj_MAC_VLAN_Incl3 (j=0-1)	Port j MAC VLAN Incl3	10060_H+j *2000_H	PMj	PMj	Kernel Reset	1689
GETH0_Portj_MAC_VLAN_Incl4 (j=0-1)	Port j MAC VLAN Incl4	10060_H+j *2000_H	PMj	PMj	Kernel Reset	1689
GETH0_Portj_MAC_VLAN_Incl5 (j=0-1)	Port j MAC VLAN Incl5	10060_H+j *2000_H	PMj	PMj	Kernel Reset	1690
GETH0_Portj_MAC_VLAN_Incl6 (j=0-1)	Port j MAC VLAN Incl6	10060_H+j *2000_H	PMj	PMj	Kernel Reset	1691
GETH0_Portj_MAC_VLAN_Incl7 (j=0-1)	Port j MAC VLAN Incl7	10060_H+j *2000_H	PMj	PMj	Kernel Reset	1692
GETH0_Portj_MAC_Inner_VLAN_Incl (j=0-1)	Port j MAC Inner VLAN Incl	10064_H+j *2000_H	PMj	PMj	Kernel Reset	1692
GETH0_Portj_MAC_Rx_Eth_Type_Match (j=0-1)	Port j MAC Rx Eth Type Match	$1006C_H$+ j*2000_H	PMj	PMj	Kernel Reset	1694
GETH0_Portj_MAC_Q0_Tx_Flow_Ctrl (j=0-1)	Port j MAC Q0 Tx Flow Ctrl	10070_H+j *2000_H	PMj	PMj	Kernel Reset	1694
GETH0_Portj_MAC_Rx_Flow_Ctrl (j=0-1)	Port j MAC Rx Flow Ctrl	10090_H+j *2000_H	PMj	PMj	Kernel Reset	1696
GETH0_Portj_MAC_RxQ_Ctrl4 (j=0-1)	Port j MAC RxQ Ctrl4	10094_H+j *2000_H	PMj	PMj	Kernel Reset	1697
GETH0_Portj_MAC_RxQ_Ctrl5 (j=0-1)	Port j MAC RxQ Ctrl5	10098_H+j *2000_H	PMj	PMj	Kernel Reset	1699

(table continues...)

Table 781 (continued) Registers overview - GETH0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
GETH0_Portj_MAC_RxQ_Ctrl0 (j=0-1)	Port j MAC RxQ Ctrl0	$100A0_H$+ j*2000_H	PMj	PMj	Kernel Reset	1700
GETH0_Portj_MAC_RxQ_Ctrl1 (j=0-1)	Port j MAC RxQ Ctrl1	$100A4_H$+ j*2000_H	PMj	PMj	Kernel Reset	1701
GETH0_Portj_MAC_RxQ_Ctrl2 (j=0-1)	Port j MAC RxQ Ctrl2	$100A8_H$+ j*2000_H	PMj	PMj	Kernel Reset	1703
GETH0_Portj_MAC_RxQ_Ctrl3 (j=0-1)	Port j MAC RxQ Ctrl3	$100AC_H$+ j*2000_H	PMj	PMj	Kernel Reset	1704
GETH0_Portj_MAC_Interrupt_Status (j=0-1)	Port j MAC Interrupt Status	$100B0_H$+ j*2000_H	PMj	PMj	Kernel Reset	1705
GETH0_Portj_MAC_Interrupt_Enable (j=0-1)	Port j MAC Interrupt Enable	$100B4_H$+ j*2000_H	PMj	PMj	Kernel Reset	1709
GETH0_Portj_MAC_Rx_Tx_Status (j=0-1)	Port j MAC Rx Tx Status	$100B8_H$+ j*2000_H	PMj	PMj	Kernel Reset	1710
GETH0_Portj_MAC_PMT_Control_Status (j=0-1)	Port j MAC PMT Control Status	$100C0_H$+ j*2000_H	PMj	PMj	Kernel Reset	1713
GETH0_Portj_MAC_RWK_Packet_Filter (j=0-1)	Port j MAC RWK Packet Filter	$100C4_H$+ j*2000_H	PMj	PMj	Kernel Reset	1715
GETH0_Portj_RWK_Filter0_1_CRC (j=0-1)	Port j RWK Filter0 1 CRC	$100C4_H$+ j*2000_H	PMj	PMj	Kernel Reset	1716
GETH0_Portj_RWK_Filter0_3_Command (j=0-1)	Port j RWK Filter0 3 Command	$100C4_H$+ j*2000_H	PMj	PMj	Kernel Reset	1716
GETH0_Portj_RWK_Filter0_3_Offset (j=0-1)	Port j RWK Filter0 3 Offset	$100C4_H$+ j*2000_H	PMj	PMj	Kernel Reset	1718

(table continues...)

Table 781 (continued) Registers overview - GETH0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
GETH0_Portj_RWK_Filter0_Byte_Mask (j=0-1)	Port j RWK Filter0 Byte Mask	$100C4_H$+ j*2000_H	PMj	PMj	Kernel Reset	1719
GETH0_Portj_RWK_Filter1_Byte_Mask (j=0-1)	Port j RWK Filter1 Byte Mask	$100C4_H$+ j*2000_H	PMj	PMj	Kernel Reset	1720
GETH0_Portj_RWK_Filter2_3_CRC (j=0-1)	Port j RWK Filter2 3 CRC	$100C4_H$+ j*2000_H	PMj	PMj	Kernel Reset	1720
GETH0_Portj_RWK_Filter2_Byte_Mask (j=0-1)	Port j RWK Filter2 Byte Mask	$100C4_H$+ j*2000_H	PMj	PMj	Kernel Reset	1721
GETH0_Portj_RWK_Filter3_Byte_Mask (j=0-1)	Port j RWK Filter3 Byte Mask	$100C4_H$+ j*2000_H	PMj	PMj	Kernel Reset	1722
GETH0_Portj_MAC_LPI_Control_Status (j=0-1)	Port j MAC LPI Control Status	$100D0_H$+ j*2000_H	PMj	PMj	Kernel Reset	1722
GETH0_Portj_MAC_LPI_Timers_Control (j=0-1)	Port j MAC LPI Timers Control	$100D4_H$+ j*2000_H	PMj	PMj	Kernel Reset	1725
GETH0_Portj_MAC_LPI_Auto_Entry_Timer (j=0-1)	Port j MAC LPI Auto Entry Timer	$100D8_H$+ j*2000_H	PMj	PMj	Kernel Reset	1725
GETH0_Portj_MAC_1US_Tic_Counter (j=0-1)	Port j MAC 1US Tic Counter	$100DC_H$+ j*2000_H	PMj	PMj	Kernel Reset	1726
GETH0_Portj_MAC_Version (j=0-1)	Port j MAC Version	10110_H+j *2000_H	PMj	PMj	Kernel Reset	1726
GETH0_Portj_MAC_Debug (j=0-1)	Port j MAC Debug	10114_H+j *2000_H	PMj	PMj	Kernel Reset	1727

(table continues...)

Table 781 (continued) Registers overview - GETH0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
GETH0_Portj_MAC_HW_Feature0 (j=0-1)	Port j MAC HW Feature0	$1011C_H$+j*2000_H	PMj	PMj	Kernel Reset	1728
GETH0_Portj_MAC_HW_Feature1 (j=0-1)	Port j MAC HW Feature1	10120_H+j*2000_H	PMj	PMj	Kernel Reset	1730
GETH0_Portj_MAC_HW_Feature2 (j=0-1)	Port j MAC HW Feature2	10124_H+j*2000_H	PMj	PMj	Kernel Reset	1733
GETH0_Portj_MAC_HW_Feature3 (j=0-1)	Port j MAC HW Feature3	10128_H+j*2000_H	PMj	PMj	Kernel Reset	1734
GETH0_Portj_MAC_HW_Feature4 (j=0-1)	Port j MAC HW Feature4	$1012C_H$+j*2000_H	PMj	PMj	Kernel Reset	1738
GETH0_Portj_MAC_Extended_Configuration (j=0-1)	Port j MAC Extended Configuration	10140_H+j*2000_H	PMj	PMj	Kernel Reset	1738
GETH0_Portj_MAC_Ext_Cfg1 (j=0-1)	Port j MAC Ext Cfg1	10144_H+j*2000_H	PMj	PMj	Kernel Reset	1741
GETH0_Portj_MAC_DPP_FSM_Interrupt_Status (j=0-1)	Port j MAC DPP FSM Interrupt Status	10150_H+j*2000_H	PMj	PMj	Kernel Reset	1742
GETH0_Portj_MAC_AXI_SLV_DPE_Addr_Status (j=0-1)	Port j MAC AXI SLV DPE Addr Status	10154_H+j*2000_H	PMj	PMj	Kernel Reset	1746
GETH0_Portj_MAC_FSM_Control (j=0-1)	Port j MAC FSM Control	10158_H+j*2000_H	PMj	PMj	Kernel Reset	1747
GETH0_Portj_MAC_FSM_ACT_Timer (j=0-1)	Port j MAC FSM ACT Timer	$1015C_H$+j*2000_H	PMj	PMj	Kernel Reset	1749
GETH0_Portj_SNPS_SCS_REG1 (j=0-1)	Port j SNPS SCS REG1	10160_H+j*2000_H	PMj	PMj	Kernel Reset	1751

(table continues...)

Table 781 (continued) Registers overview - GETH0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
GETH0_Portj_MAC_SCSR_Control (j=0-1)	Port j MAC SCSR Control	10164_H+j *2000_H	PMj	PMj	Kernel Reset	1751
GETH0_Portj_MAC_SCSR_Parity_Status (j=0-1)	Port j MAC SCSR Parity Status	10168_H+j *2000_H	PMj	PMj	Kernel Reset	1752
GETH0_Portj_MAC_SCSR_Parity_Err_Count (j=0-1)	Port j MAC SCSR Parity Err Count	$1016C_H$+ j*2000_H	PMj	PMj	Kernel Reset	1753
GETH0_Portj_MAC_Flex_Hdr_Cfg (j=0-1)	Port j MAC Flex Hdr Cfg	10194_H+j *2000_H	PMj	PMj	Kernel Reset	1753
GETH0_Portj_MAC_Flex_Hdr_Low (j=0-1)	Port j MAC Flex Hdr Low	10198_H+j *2000_H	PMj	PMj	Kernel Reset	1754
GETH0_Portj_MAC_Flex_Hdr_High (j=0-1)	Port j MAC Flex Hdr High	$1019C_H$+ j*2000_H	PMj	PMj	Kernel Reset	1755
GETH0_Portj_MDIO_Single_Command_Address (j=0-1)	Port j MDIO Single Command Address	10200_H+j *2000_H	PMj	PMj	Kernel Reset	1755
GETH0_Portj_MDIO_Single_Command_Control_Data (j=0-1)	Port j MDIO Single Command Control Data	10204_H+j *2000_H	PMj	PMj	Kernel Reset	1756
GETH0_Portj_MDIO_Continuous_Write_Address (j=0-1)	Port j MDIO Continuous Write Address	10208_H+j *2000_H	PMj	PMj	Kernel Reset	1758
GETH0_Portj_MDIO_Continuous_Write_Data (j=0-1)	Port j MDIO Continuous Write Data	$1020C_H$+ j*2000_H	PMj	PMj	Kernel Reset	1759
GETH0_Portj_MDIO_Continuous_Scan_Port_Enable (j=0-1)	Port j MDIO Continuous Scan Port Enable	10210_H+j *2000_H	PMj	PMj	Kernel Reset	1760

(table continues...)

Table 781 (continued) Registers overview - GETH0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
GETH0_Portj_MDIO_Interrupt_Status (j=0-1)	Port j MDIO Interrupt Status	10214_H+j*2000_H	PMj	PMj	Kernel Reset	1762
GETH0_Portj_MDIO_Interrupt_Enable (j=0-1)	Port j MDIO Interrupt Enable	10218_H+j*2000_H	PMj	PMj	Kernel Reset	1765
GETH0_Portj_MDIO_Port_Connect_Disconnect_Status (j=0-1)	Port j MDIO Port Connect Disconnect Status	$1021C_H$+j*2000_H	PMj	PMj	Kernel Reset	1767
GETH0_Portj_MDIO_Clause_22_Port (j=0-1)	Port j MDIO Clause 22 Port	10220_H+j*2000_H	PMj	PMj	Kernel Reset	1770
GETH0_Portj_MDIO_Port_Nx4_Indirect_Control (j=0-1)	Port j MDIO Port Nx4 Indirect Control	10224_H+j*2000_H	PMj	PMj	Kernel Reset	1772
GETH0_Portj_MDIO_PortNx4P0_Device_In_Use (j=0-1)	Port j MDIO PortNx4P0 Device In Use	10230_H+j*2000_H	PMj	PMj	Kernel Reset	1773
GETH0_Portj_MDIO_PortNx4P0_Link_Status (j=0-1)	Port j MDIO PortNx4P0 Link Status	10234_H+j*2000_H	PMj	PMj	Kernel Reset	1775
GETH0_Portj_MDIO_PortNx4P0_Alive_Status (j=0-1)	Port j MDIO PortNx4P0 Alive Status	10238_H+j*2000_H	PMj	PMj	Kernel Reset	1776
GETH0_Portj_MDIO_PortNx4P1_Device_In_Use (j=0-1)	Port j MDIO PortNx4P1 Device In Use	10240_H+j*2000_H	PMj	PMj	Kernel Reset	1777
GETH0_Portj_MDIO_PortNx4P1_Link_Status (j=0-1)	Port j MDIO PortNx4P1 Link Status	10244_H+j*2000_H	PMj	PMj	Kernel Reset	1778

(table continues...)

Table 781 (continued) Registers overview - GETH0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
GETH0_Portj_MDIO_PortNx4P1_Alive_Status (j=0-1)	Port j MDIO PortNx4P1 Alive Status	10248_H+j *2000_H	PMj	PMj	Kernel Reset	1780
GETH0_Portj_MDIO_PortNx4P2_Device_In_Use (j=0-1)	Port j MDIO PortNx4P2 Device In Use	10250_H+j *2000_H	PMj	PMj	Kernel Reset	1781
GETH0_Portj_MDIO_PortNx4P2_Link_Status (j=0-1)	Port j MDIO PortNx4P2 Link Status	10254_H+j *2000_H	PMj	PMj	Kernel Reset	1782
GETH0_Portj_MDIO_PortNx4P2_Alive_Status (j=0-1)	Port j MDIO PortNx4P2 Alive Status	10258_H+j *2000_H	PMj	PMj	Kernel Reset	1783
GETH0_Portj_MDIO_PortNx4P3_Device_In_Use (j=0-1)	Port j MDIO PortNx4P3 Device In Use	10260_H+j *2000_H	PMj	PMj	Kernel Reset	1785
GETH0_Portj_MDIO_PortNx4P3_Link_Status (j=0-1)	Port j MDIO PortNx4P3 Link Status	10264_H+j *2000_H	PMj	PMj	Kernel Reset	1786
GETH0_Portj_MDIO_PortNx4P3_Alive_Status (j=0-1)	Port j MDIO PortNx4P3 Alive Status	10268_H+j *2000_H	PMj	PMj	Kernel Reset	1787
GETH0_Portj_MAC_FPE_CTRL_STS (j=0-1)	Port j MAC FPE CTRL STS	10280_H+j *2000_H	PMj	PMj	Kernel Reset	1788
GETH0_Portj_MAC_CSR_SW_Ctrl (j=0-1)	Port j MAC CSR SW Ctrl	10290_H+j *2000_H	PMj	PMj	Kernel Reset	1790
GETH0_Portj_MAC_Address0_High (j=0-1)	Port j MAC Address0 High	10300_H+j *2000_H	PMj	PMj	Kernel Reset	1791
GETH0_Portj_MAC_Address0_Low (j=0-1)	Port j MAC Address0 Low	10304_H+j *2000_H	PMj	PMj	Kernel Reset	1792

(table continues...)

Table 781 (continued) Registers overview - GETH0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
GETH0_Portj_MAC_Address1_High (j=0-1)	Port j MAC Address1 High	10308_H+j*2000_H	PMj	PMj	Kernel Reset	1793
GETH0_Portj_MAC_Address1_Low (j=0-1)	Port j MAC Address1 Low	$1030C_H$+j*2000_H	PMj	PMj	Kernel Reset	1794
GETH0_Portj_MAC_Address2_High (j=0-1)	Port j MAC Address2 High	10310_H+j*2000_H	PMj	PMj	Kernel Reset	1795
GETH0_Portj_MAC_Address2_Low (j=0-1)	Port j MAC Address2 Low	10314_H+j*2000_H	PMj	PMj	Kernel Reset	1796
GETH0_Portj_MAC_Address3_High (j=0-1)	Port j MAC Address3 High	10318_H+j*2000_H	PMj	PMj	Kernel Reset	1797
GETH0_Portj_MAC_Address3_Low (j=0-1)	Port j MAC Address3 Low	$1031C_H$+j*2000_H	PMj	PMj	Kernel Reset	1798
GETH0_Portj_MAC_Address4_High (j=0-1)	Port j MAC Address4 High	10320_H+j*2000_H	PMj	PMj	Kernel Reset	1799
GETH0_Portj_MAC_Address4_Low (j=0-1)	Port j MAC Address4 Low	10324_H+j*2000_H	PMj	PMj	Kernel Reset	1800
GETH0_Portj_MAC_Address5_High (j=0-1)	Port j MAC Address5 High	10328_H+j*2000_H	PMj	PMj	Kernel Reset	1801
GETH0_Portj_MAC_Address5_Low (j=0-1)	Port j MAC Address5 Low	$1032C_H$+j*2000_H	PMj	PMj	Kernel Reset	1802
GETH0_Portj_MAC_Address6_High (j=0-1)	Port j MAC Address6 High	10330_H+j*2000_H	PMj	PMj	Kernel Reset	1803
GETH0_Portj_MAC_Address6_Low (j=0-1)	Port j MAC Address6 Low	10334_H+j*2000_H	PMj	PMj	Kernel Reset	1804

(table continues...)

Table 781 (continued) Registers overview - GETH0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
GETH0_Portj_MAC_Address7_High (j=0-1)	Port j MAC Address7 High	10338_H+j*2000_H	PMj	PMj	Kernel Reset	1805
GETH0_Portj_MAC_Address7_Low (j=0-1)	Port j MAC Address7 Low	$1033C_H$+j*2000_H	PMj	PMj	Kernel Reset	1806
GETH0_Portj_MAC_Address8_High (j=0-1)	Port j MAC Address8 High	10340_H+j*2000_H	PMj	PMj	Kernel Reset	1807
GETH0_Portj_MAC_Address8_Low (j=0-1)	Port j MAC Address8 Low	10344_H+j*2000_H	PMj	PMj	Kernel Reset	1808
GETH0_Portj_MAC_Address9_High (j=0-1)	Port j MAC Address9 High	10348_H+j*2000_H	PMj	PMj	Kernel Reset	1809
GETH0_Portj_MAC_Address9_Low (j=0-1)	Port j MAC Address9 Low	$1034C_H$+j*2000_H	PMj	PMj	Kernel Reset	1810
GETH0_Portj_MAC_Address10_High (j=0-1)	Port j MAC Address10 High	10350_H+j*2000_H	PMj	PMj	Kernel Reset	1811
GETH0_Portj_MAC_Address10_Low (j=0-1)	Port j MAC Address10 Low	10354_H+j*2000_H	PMj	PMj	Kernel Reset	1812
GETH0_Portj_MAC_Address11_High (j=0-1)	Port j MAC Address11 High	10358_H+j*2000_H	PMj	PMj	Kernel Reset	1813
GETH0_Portj_MAC_Address11_Low (j=0-1)	Port j MAC Address11 Low	$1035C_H$+j*2000_H	PMj	PMj	Kernel Reset	1814
GETH0_Portj_MAC_Address12_High (j=0-1)	Port j MAC Address12 High	10360_H+j*2000_H	PMj	PMj	Kernel Reset	1815

(table continues...)

Table 781 (continued) Registers overview - GETH0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
GETH0_Portj_MAC_Address12_Low (j=0-1)	Port j MAC Address12 Low	10364_H+j*2000_H	PMj	PMj	Kernel Reset	1816
GETH0_Portj_MAC_Address13_High (j=0-1)	Port j MAC Address13 High	10368_H+j*2000_H	PMj	PMj	Kernel Reset	1817
GETH0_Portj_MAC_Address13_Low (j=0-1)	Port j MAC Address13 Low	$1036C_H$+j*2000_H	PMj	PMj	Kernel Reset	1818
GETH0_Portj_MAC_Address14_High (j=0-1)	Port j MAC Address14 High	10370_H+j*2000_H	PMj	PMj	Kernel Reset	1819
GETH0_Portj_MAC_Address14_Low (j=0-1)	Port j MAC Address14 Low	10374_H+j*2000_H	PMj	PMj	Kernel Reset	1820
GETH0_Portj_MAC_Address15_High (j=0-1)	Port j MAC Address15 High	10378_H+j*2000_H	PMj	PMj	Kernel Reset	1821
GETH0_Portj_MAC_Address15_Low (j=0-1)	Port j MAC Address15 Low	$1037C_H$+j*2000_H	PMj	PMj	Kernel Reset	1822
GETH0_Portj_MAC_Address16_High (j=0-1)	Port j MAC Address16 High	10380_H+j*2000_H	PMj	PMj	Kernel Reset	1823
GETH0_Portj_MAC_Address16_Low (j=0-1)	Port j MAC Address16 Low	10384_H+j*2000_H	PMj	PMj	Kernel Reset	1824
GETH0_Portj_MAC_Address17_High (j=0-1)	Port j MAC Address17 High	10388_H+j*2000_H	PMj	PMj	Kernel Reset	1825
GETH0_Portj_MAC_Address17_Low (j=0-1)	Port j MAC Address17 Low	$1038C_H$+j*2000_H	PMj	PMj	Kernel Reset	1826

(table continues...)

Table 781 (continued) Registers overview - GETH0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
GETH0_Portj_MAC_Address18_High (j=0-1)	Port j MAC Address18 High	10390_H+j*2000_H	PMj	PMj	Kernel Reset	1827
GETH0_Portj_MAC_Address18_Low (j=0-1)	Port j MAC Address18 Low	10394_H+j*2000_H	PMj	PMj	Kernel Reset	1828
GETH0_Portj_MAC_Address19_High (j=0-1)	Port j MAC Address19 High	10398_H+j*2000_H	PMj	PMj	Kernel Reset	1829
GETH0_Portj_MAC_Address19_Low (j=0-1)	Port j MAC Address19 Low	$1039C_H$+j*2000_H	PMj	PMj	Kernel Reset	1830
GETH0_Portj_MAC_Address20_High (j=0-1)	Port j MAC Address20 High	$103A0_H$+j*2000_H	PMj	PMj	Kernel Reset	1831
GETH0_Portj_MAC_Address20_Low (j=0-1)	Port j MAC Address20 Low	$103A4_H$+j*2000_H	PMj	PMj	Kernel Reset	1832
GETH0_Portj_MAC_Address21_High (j=0-1)	Port j MAC Address21 High	$103A8_H$+j*2000_H	PMj	PMj	Kernel Reset	1833
GETH0_Portj_MAC_Address21_Low (j=0-1)	Port j MAC Address21 Low	$103AC_H$+j*2000_H	PMj	PMj	Kernel Reset	1834
GETH0_Portj_MAC_Address22_High (j=0-1)	Port j MAC Address22 High	$103B0_H$+j*2000_H	PMj	PMj	Kernel Reset	1835
GETH0_Portj_MAC_Address22_Low (j=0-1)	Port j MAC Address22 Low	$103B4_H$+j*2000_H	PMj	PMj	Kernel Reset	1836
GETH0_Portj_MAC_Address23_High (j=0-1)	Port j MAC Address23 High	$103B8_H$+j*2000_H	PMj	PMj	Kernel Reset	1837

(table continues...)

Table 781 (continued) Registers overview - GETH0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
GETH0_Portj_MAC_Address23_Low (j=0-1)	Port j MAC Address23 Low	$103BC_H$+ j*2000_H	PMj	PMj	Kernel Reset	1838
GETH0_Portj_MAC_Address24_High (j=0-1)	Port j MAC Address24 High	$103C0_H$+ j*2000_H	PMj	PMj	Kernel Reset	1839
GETH0_Portj_MAC_Address24_Low (j=0-1)	Port j MAC Address24 Low	$103C4_H$+ j*2000_H	PMj	PMj	Kernel Reset	1840
GETH0_Portj_MAC_Address25_High (j=0-1)	Port j MAC Address25 High	$103C8_H$+ j*2000_H	PMj	PMj	Kernel Reset	1841
GETH0_Portj_MAC_Address25_Low (j=0-1)	Port j MAC Address25 Low	$103CC_H$+ j*2000_H	PMj	PMj	Kernel Reset	1842
GETH0_Portj_MAC_Address26_High (j=0-1)	Port j MAC Address26 High	$103D0_H$+ j*2000_H	PMj	PMj	Kernel Reset	1843
GETH0_Portj_MAC_Address26_Low (j=0-1)	Port j MAC Address26 Low	$103D4_H$+ j*2000_H	PMj	PMj	Kernel Reset	1844
GETH0_Portj_MAC_Address27_High (j=0-1)	Port j MAC Address27 High	$103D8_H$+ j*2000_H	PMj	PMj	Kernel Reset	1845
GETH0_Portj_MAC_Address27_Low (j=0-1)	Port j MAC Address27 Low	$103DC_H$+ j*2000_H	PMj	PMj	Kernel Reset	1846
GETH0_Portj_MAC_Address28_High (j=0-1)	Port j MAC Address28 High	$103E0_H$+j *2000_H	PMj	PMj	Kernel Reset	1847
GETH0_Portj_MAC_Address28_Low (j=0-1)	Port j MAC Address28 Low	$103E4_H$+j *2000_H	PMj	PMj	Kernel Reset	1848

(table continues...)

Table 781 (continued) Registers overview - GETH0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
GETH0_Portj_MAC_Address29_High (j=0-1)	Port j MAC Address29 High	$103E8_H$+j*2000_H	PMj	PMj	Kernel Reset	1849
GETH0_Portj_MAC_Address29_Low (j=0-1)	Port j MAC Address29 Low	$103EC_H$+j*2000_H	PMj	PMj	Kernel Reset	1850
GETH0_Portj_MAC_Address30_High (j=0-1)	Port j MAC Address30 High	$103F0_H$+j*2000_H	PMj	PMj	Kernel Reset	1851
GETH0_Portj_MAC_Address30_Low (j=0-1)	Port j MAC Address30 Low	$103F4_H$+j*2000_H	PMj	PMj	Kernel Reset	1852
GETH0_Portj_MAC_Address31_High (j=0-1)	Port j MAC Address31 High	$103F8_H$+j*2000_H	PMj	PMj	Kernel Reset	1853
GETH0_Portj_MAC_Address31_Low (j=0-1)	Port j MAC Address31 Low	$103FC_H$+j*2000_H	PMj	PMj	Kernel Reset	1854
GETH0_Portj_MAC_Indir_Access_Ctrl (j=0-1)	Port j MAC Indir Access Ctrl	10700_H+j*2000_H	PMj	PMj	Kernel Reset	1855
GETH0_Portj_MAC_Indir_Access_Data (j=0-1)	Port j MAC Indir Access Data	10704_H+j*2000_H	PMj	PMj	Kernel Reset	1857
GETH0_Portj_MAC_DChSel_IndReg0 (j=0-1)	Port j MAC DChSel IndReg0	10704_H+j*2000_H	PMj	PMj	Kernel Reset	1857
GETH0_Portj_MAC_DChSel_IndReg1 (j=0-1)	Port j MAC DChSel IndReg1	10704_H+j*2000_H	PMj	PMj	Kernel Reset	1857
GETH0_Portj_MAC_DChSel_IndReg10 (j=0-1)	Port j MAC DChSel IndReg10	10704_H+j*2000_H	PMj	PMj	Kernel Reset	1858

(table continues...)

Table 781 (continued) Registers overview - GETH0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
GETH0_Portj_MAC_DChSel_IndReg11 (j=0-1)	Port j MAC DChSel IndReg11	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1858
GETH0_Portj_MAC_DChSel_IndReg12 (j=0-1)	Port j MAC DChSel IndReg12	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1859
GETH0_Portj_MAC_DChSel_IndReg13 (j=0-1)	Port j MAC DChSel IndReg13	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1859
GETH0_Portj_MAC_DChSel_IndReg14 (j=0-1)	Port j MAC DChSel IndReg14	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1860
GETH0_Portj_MAC_DChSel_IndReg15 (j=0-1)	Port j MAC DChSel IndReg15	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1860
GETH0_Portj_MAC_DChSel_IndReg16 (j=0-1)	Port j MAC DChSel IndReg16	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1861
GETH0_Portj_MAC_DChSel_IndReg17 (j=0-1)	Port j MAC DChSel IndReg17	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1861
GETH0_Portj_MAC_DChSel_IndReg18 (j=0-1)	Port j MAC DChSel IndReg18	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1862
GETH0_Portj_MAC_DChSel_IndReg19 (j=0-1)	Port j MAC DChSel IndReg19	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1862
GETH0_Portj_MAC_DChSel_IndReg2 (j=0-1)	Port j MAC DChSel IndReg2	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1863
GETH0_Portj_MAC_DChSel_IndReg20 (j=0-1)	Port j MAC DChSel IndReg20	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1863

(table continues...)

Table 781 (continued) Registers overview - GETH0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
GETH0_Portj_MAC_DChSel_IndReg21 (j=0-1)	Port j MAC DChSel IndReg21	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1864
GETH0_Portj_MAC_DChSel_IndReg22 (j=0-1)	Port j MAC DChSel IndReg22	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1864
GETH0_Portj_MAC_DChSel_IndReg23 (j=0-1)	Port j MAC DChSel IndReg23	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1865
GETH0_Portj_MAC_DChSel_IndReg24 (j=0-1)	Port j MAC DChSel IndReg24	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1865
GETH0_Portj_MAC_DChSel_IndReg25 (j=0-1)	Port j MAC DChSel IndReg25	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1866
GETH0_Portj_MAC_DChSel_IndReg26 (j=0-1)	Port j MAC DChSel IndReg26	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1866
GETH0_Portj_MAC_DChSel_IndReg27 (j=0-1)	Port j MAC DChSel IndReg27	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1867
GETH0_Portj_MAC_DChSel_IndReg28 (j=0-1)	Port j MAC DChSel IndReg28	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1867
GETH0_Portj_MAC_DChSel_IndReg29 (j=0-1)	Port j MAC DChSel IndReg29	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1868
GETH0_Portj_MAC_DChSel_IndReg3 (j=0-1)	Port j MAC DChSel IndReg3	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1868
GETH0_Portj_MAC_DChSel_IndReg30 (j=0-1)	Port j MAC DChSel IndReg30	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1869

(table continues...)

Table 781 (continued) Registers overview - GETH0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
GETH0_Portj_MAC_DChSel_IndReg31 (j=0-1)	Port j MAC DChSel IndReg31	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1869
GETH0_Portj_MAC_DChSel_IndReg4 (j=0-1)	Port j MAC DChSel IndReg4	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1870
GETH0_Portj_MAC_DChSel_IndReg5 (j=0-1)	Port j MAC DChSel IndReg5	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1870
GETH0_Portj_MAC_DChSel_IndReg6 (j=0-1)	Port j MAC DChSel IndReg6	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1871
GETH0_Portj_MAC_DChSel_IndReg7 (j=0-1)	Port j MAC DChSel IndReg7	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1871
GETH0_Portj_MAC_DChSel_IndReg8 (j=0-1)	Port j MAC DChSel IndReg8	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1872
GETH0_Portj_MAC_DChSel_IndReg9 (j=0-1)	Port j MAC DChSel IndReg9	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1872
GETH0_Portj_MAC_DPCSel_IndReg0 (j=0-1)	Port j MAC DPCSel IndReg0	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1873
GETH0_Portj_MAC_DPCSel_IndReg1 (j=0-1)	Port j MAC DPCSel IndReg1	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1873
GETH0_Portj_MAC_DPCSel_IndReg10 (j=0-1)	Port j MAC DPCSel IndReg10	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1874
GETH0_Portj_MAC_DPCSel_IndReg11 (j=0-1)	Port j MAC DPCSel IndReg11	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1875

(table continues...)

Table 781 (continued) Registers overview - GETH0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
GETH0_Portj_MAC_DPCSel_IndReg12 (j=0-1)	Port j MAC DPCSel IndReg12	10704_H+j*2000_H	PMj	PMj	Kernel Reset	1875
GETH0_Portj_MAC_DPCSel_IndReg13 (j=0-1)	Port j MAC DPCSel IndReg13	10704_H+j*2000_H	PMj	PMj	Kernel Reset	1876
GETH0_Portj_MAC_DPCSel_IndReg14 (j=0-1)	Port j MAC DPCSel IndReg14	10704_H+j*2000_H	PMj	PMj	Kernel Reset	1876
GETH0_Portj_MAC_DPCSel_IndReg15 (j=0-1)	Port j MAC DPCSel IndReg15	10704_H+j*2000_H	PMj	PMj	Kernel Reset	1877
GETH0_Portj_MAC_DPCSel_IndReg16 (j=0-1)	Port j MAC DPCSel IndReg16	10704_H+j*2000_H	PMj	PMj	Kernel Reset	1877
GETH0_Portj_MAC_DPCSel_IndReg17 (j=0-1)	Port j MAC DPCSel IndReg17	10704_H+j*2000_H	PMj	PMj	Kernel Reset	1878
GETH0_Portj_MAC_DPCSel_IndReg18 (j=0-1)	Port j MAC DPCSel IndReg18	10704_H+j*2000_H	PMj	PMj	Kernel Reset	1879
GETH0_Portj_MAC_DPCSel_IndReg19 (j=0-1)	Port j MAC DPCSel IndReg19	10704_H+j*2000_H	PMj	PMj	Kernel Reset	1879
GETH0_Portj_MAC_DPCSel_IndReg2 (j=0-1)	Port j MAC DPCSel IndReg2	10704_H+j*2000_H	PMj	PMj	Kernel Reset	1880
GETH0_Portj_MAC_DPCSel_IndReg20 (j=0-1)	Port j MAC DPCSel IndReg20	10704_H+j*2000_H	PMj	PMj	Kernel Reset	1880
GETH0_Portj_MAC_DPCSel_IndReg21 (j=0-1)	Port j MAC DPCSel IndReg21	10704_H+j*2000_H	PMj	PMj	Kernel Reset	1881

(table continues...)

Table 781 (continued) Registers overview - GETH0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
GETH0_Portj_MAC_DPCSel_IndReg22 (j=0-1)	Port j MAC DPCSel IndReg22	10704_H+j*2000_H	PMj	PMj	Kernel Reset	1881
GETH0_Portj_MAC_DPCSel_IndReg23 (j=0-1)	Port j MAC DPCSel IndReg23	10704_H+j*2000_H	PMj	PMj	Kernel Reset	1882
GETH0_Portj_MAC_DPCSel_IndReg24 (j=0-1)	Port j MAC DPCSel IndReg24	10704_H+j*2000_H	PMj	PMj	Kernel Reset	1883
GETH0_Portj_MAC_DPCSel_IndReg25 (j=0-1)	Port j MAC DPCSel IndReg25	10704_H+j*2000_H	PMj	PMj	Kernel Reset	1883
GETH0_Portj_MAC_DPCSel_IndReg26 (j=0-1)	Port j MAC DPCSel IndReg26	10704_H+j*2000_H	PMj	PMj	Kernel Reset	1884
GETH0_Portj_MAC_DPCSel_IndReg27 (j=0-1)	Port j MAC DPCSel IndReg27	10704_H+j*2000_H	PMj	PMj	Kernel Reset	1884
GETH0_Portj_MAC_DPCSel_IndReg28 (j=0-1)	Port j MAC DPCSel IndReg28	10704_H+j*2000_H	PMj	PMj	Kernel Reset	1885
GETH0_Portj_MAC_DPCSel_IndReg29 (j=0-1)	Port j MAC DPCSel IndReg29	10704_H+j*2000_H	PMj	PMj	Kernel Reset	1885
GETH0_Portj_MAC_DPCSel_IndReg3 (j=0-1)	Port j MAC DPCSel IndReg3	10704_H+j*2000_H	PMj	PMj	Kernel Reset	1886
GETH0_Portj_MAC_DPCSel_IndReg30 (j=0-1)	Port j MAC DPCSel IndReg30	10704_H+j*2000_H	PMj	PMj	Kernel Reset	1887
GETH0_Portj_MAC_DPCSel_IndReg31 (j=0-1)	Port j MAC DPCSel IndReg31	10704_H+j*2000_H	PMj	PMj	Kernel Reset	1887

(table continues...)

Table 781 (continued) Registers overview - GETH0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
GETH0_Portj_MAC_DPCSel_IndReg4 (j=0-1)	Port j MAC DPCSel IndReg4	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1888
GETH0_Portj_MAC_DPCSel_IndReg5 (j=0-1)	Port j MAC DPCSel IndReg5	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1888
GETH0_Portj_MAC_DPCSel_IndReg6 (j=0-1)	Port j MAC DPCSel IndReg6	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1889
GETH0_Portj_MAC_DPCSel_IndReg7 (j=0-1)	Port j MAC DPCSel IndReg7	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1889
GETH0_Portj_MAC_DPCSel_IndReg8 (j=0-1)	Port j MAC DPCSel IndReg8	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1890
GETH0_Portj_MAC_DPCSel_IndReg9 (j=0-1)	Port j MAC DPCSel IndReg9	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1891
GETH0_Portj_MAC_FPCSel_IndReg0 (j=0-1)	Port j MAC FPCSel IndReg0	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1891
GETH0_Portj_MAC_LPCSel_IndReg0 (j=0-1)	Port j MAC LPCSel IndReg0	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1892
GETH0_Portj_MAC_LPCSel_IndReg1 (j=0-1)	Port j MAC LPCSel IndReg1	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1892
GETH0_Portj_MAC_LPCSel_IndReg2 (j=0-1)	Port j MAC LPCSel IndReg2	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1893
GETH0_Portj_MAC_LPCSel_IndReg3 (j=0-1)	Port j MAC LPCSel IndReg3	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1893

(table continues...)

Table 781 (continued) Registers overview - GETH0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
GETH0_Portj_MAC_LPCSel_IndReg4 (j=0-1)	Port j MAC LPCSel IndReg4	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1894
GETH0_Portj_MAC_LPCSel_IndReg5 (j=0-1)	Port j MAC LPCSel IndReg5	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1895
GETH0_Portj_MAC_LPCSel_IndReg6 (j=0-1)	Port j MAC LPCSel IndReg6	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1895
GETH0_Portj_MAC_LPCSel_IndReg7 (j=0-1)	Port j MAC LPCSel IndReg7	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1896
GETH0_Portj_MAC_PCCtrl_IndReg0 (j=0-1)	Port j MAC PCCtrl IndReg0	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1896
GETH0_Portj_MAC_PCCtrl_IndReg1 (j=0-1)	Port j MAC PCCtrl IndReg1	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1897
GETH0_Portj_MAC_PCCtrl_IndReg10 (j=0-1)	Port j MAC PCCtrl IndReg10	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1899
GETH0_Portj_MAC_PCCtrl_IndReg11 (j=0-1)	Port j MAC PCCtrl IndReg11	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1900
GETH0_Portj_MAC_PCCtrl_IndReg12 (j=0-1)	Port j MAC PCCtrl IndReg12	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1901
GETH0_Portj_MAC_PCCtrl_IndReg13 (j=0-1)	Port j MAC PCCtrl IndReg13	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1902
GETH0_Portj_MAC_PCCtrl_IndReg14 (j=0-1)	Port j MAC PCCtrl IndReg14	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1903

(table continues...)

Table 781 (continued) Registers overview - GETH0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
GETH0_Portj_MAC_PCCtrl_IndReg15 (j=0-1)	Port j MAC PCCtrl IndReg15	10704_H+j*2000_H	PMj	PMj	Kernel Reset	1905
GETH0_Portj_MAC_PCCtrl_IndReg2 (j=0-1)	Port j MAC PCCtrl IndReg2	10704_H+j*2000_H	PMj	PMj	Kernel Reset	1906
GETH0_Portj_MAC_PCCtrl_IndReg3 (j=0-1)	Port j MAC PCCtrl IndReg3	10704_H+j*2000_H	PMj	PMj	Kernel Reset	1907
GETH0_Portj_MAC_PCCtrl_IndReg4 (j=0-1)	Port j MAC PCCtrl IndReg4	10704_H+j*2000_H	PMj	PMj	Kernel Reset	1908
GETH0_Portj_MAC_PCCtrl_IndReg5 (j=0-1)	Port j MAC PCCtrl IndReg5	10704_H+j*2000_H	PMj	PMj	Kernel Reset	1909
GETH0_Portj_MAC_PCCtrl_IndReg6 (j=0-1)	Port j MAC PCCtrl IndReg6	10704_H+j*2000_H	PMj	PMj	Kernel Reset	1911
GETH0_Portj_MAC_PCCtrl_IndReg7 (j=0-1)	Port j MAC PCCtrl IndReg7	10704_H+j*2000_H	PMj	PMj	Kernel Reset	1912
GETH0_Portj_MAC_PCCtrl_IndReg8 (j=0-1)	Port j MAC PCCtrl IndReg8	10704_H+j*2000_H	PMj	PMj	Kernel Reset	1913
GETH0_Portj_MAC_PCCtrl_IndReg9 (j=0-1)	Port j MAC PCCtrl IndReg9	10704_H+j*2000_H	PMj	PMj	Kernel Reset	1914
GETH0_Portj_MAC_PCStatus_DA_IndReg (j=0-1)	Port j MAC PCStatus DA IndReg	10704_H+j*2000_H	PMj	PMj	Kernel Reset	1915
GETH0_Portj_MAC_PCStatus_Filter_Event_IndReg (j=0-1)	Port j MAC PCStatus Filter Event IndReg	10704_H+j*2000_H	PMj	PMj	Kernel Reset	1916

(table continues...)

Table 781 (continued) Registers overview - GETH0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
GETH0_Portj_MAC_PCStatus_L3L4_IndReg (j=0-1)	Port j MAC PCStatus L3L4 IndReg	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1916
GETH0_Portj_MAC_PCStatus_VLAN_IndReg (j=0-1)	Port j MAC PCStatus VLAN IndReg	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1917
GETH0_Portj_MAC_PCntr_IndReg0 (j=0-1)	Port j MAC PCntr IndReg0	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1917
GETH0_Portj_MAC_PCntr_IndReg1 (j=0-1)	Port j MAC PCntr IndReg1	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1918
GETH0_Portj_MAC_PCntr_IndReg10 (j=0-1)	Port j MAC PCntr IndReg10	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1919
GETH0_Portj_MAC_PCntr_IndReg11 (j=0-1)	Port j MAC PCntr IndReg11	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1919
GETH0_Portj_MAC_PCntr_IndReg12 (j=0-1)	Port j MAC PCntr IndReg12	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1920
GETH0_Portj_MAC_PCntr_IndReg13 (j=0-1)	Port j MAC PCntr IndReg13	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1920
GETH0_Portj_MAC_PCntr_IndReg14 (j=0-1)	Port j MAC PCntr IndReg14	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1921
GETH0_Portj_MAC_PCntr_IndReg15 (j=0-1)	Port j MAC PCntr IndReg15	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1922
GETH0_Portj_MAC_PCntr_IndReg2 (j=0-1)	Port j MAC PCntr IndReg2	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1922

(table continues...)

Table 781 (continued) Registers overview - GETH0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
GETH0_Portj_MAC_PCntr_IndReg3 (j=0-1)	Port j MAC PCntr IndReg3	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1923
GETH0_Portj_MAC_PCntr_IndReg4 (j=0-1)	Port j MAC PCntr IndReg4	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1923
GETH0_Portj_MAC_PCntr_IndReg5 (j=0-1)	Port j MAC PCntr IndReg5	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1924
GETH0_Portj_MAC_PCntr_IndReg6 (j=0-1)	Port j MAC PCntr IndReg6	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1925
GETH0_Portj_MAC_PCntr_IndReg7 (j=0-1)	Port j MAC PCntr IndReg7	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1925
GETH0_Portj_MAC_PCntr_IndReg8 (j=0-1)	Port j MAC PCntr IndReg8	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1926
GETH0_Portj_MAC_PCntr_IndReg9 (j=0-1)	Port j MAC PCntr IndReg9	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1926
GETH0_Portj_MAC_VPCSel_IndReg0 (j=0-1)	Port j MAC VPCSel IndReg0	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1927
GETH0_Portj_MAC_VPCSel_IndReg1 (j=0-1)	Port j MAC VPCSel IndReg1	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1928
GETH0_Portj_MAC_VPCSel_IndReg2 (j=0-1)	Port j MAC VPCSel IndReg2	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1928
GETH0_Portj_MAC_VPCSel_IndReg3 (j=0-1)	Port j MAC VPCSel IndReg3	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1929

(table continues...)

Table 781 (continued) Registers overview - GETH0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
GETH0_Portj_MAC_VPCSel_IndReg4 (j=0-1)	Port j MAC VPCSel IndReg4	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1929
GETH0_Portj_MAC_VPCSel_IndReg5 (j=0-1)	Port j MAC VPCSel IndReg5	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1930
GETH0_Portj_MAC_VPCSel_IndReg6 (j=0-1)	Port j MAC VPCSel IndReg6	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1931
GETH0_Portj_MAC_VPCSel_IndReg7 (j=0-1)	Port j MAC VPCSel IndReg7	10704_H+j *2000_H	PMj	PMj	Kernel Reset	1931
GETH0_Portj_MAC_PCTH_Intr_Enable (j=0-1)	Port j MAC PCTH Intr Enable	10708_H+j *2000_H	PMj	PMj	Kernel Reset	1932
GETH0_Portj_MAC_PCTH_Intr_Status (j=0-1)	Port j MAC PCTH Intr Status	$1070C_H$+ j*2000_H	PMj	PMj	Kernel Reset	1933
GETH0_Portj_MAC_PCTW_Intr_Enable (j=0-1)	Port j MAC PCTW Intr Enable	10730_H+j *2000_H	PMj	PMj	Kernel Reset	1933
GETH0_Portj_MAC_PCTW_Intr_Status (j=0-1)	Port j MAC PCTW Intr Status	10734_H+j *2000_H	PMj	PMj	Kernel Reset	1934
GETH0_Portj_MMC_Control (j=0-1)	Port j MMC Control	10800_H+j *2000_H	PMj	PMj	Kernel Reset	1934
GETH0_Portj_MMC_Rx_Interrupt (j=0-1)	Port j MMC Rx Interrupt	10804_H+j *2000_H	PMj	PMj	Kernel Reset	1936
GETH0_Portj_MMC_Tx_Interrupt (j=0-1)	Port j MMC Tx Interrupt	10808_H+j *2000_H	PMj	PMj	Kernel Reset	1939

(table continues...)

Table 781 (continued) Registers overview - GETH0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
GETH0_Portj_MMC_Receive_Interrupt_Enable (j=0-1)	Port j MMC Receive Interrupt Enable	$1080C_H$+j*2000_H	PMj	PMj	Kernel Reset	1944
GETH0_Portj_MMC_Transmit_Interrupt_Enable (j=0-1)	Port j MMC Transmit Interrupt Enable	10810_H+j*2000_H	PMj	PMj	Kernel Reset	1948
GETH0_Portj_Tx_Octet_Count_Good_Bad_Low (j=0-1)	Port j Tx Octet Count Good Bad Low	10814_H+j*2000_H	PMj	PMj	Kernel Reset	1952
GETH0_Portj_Tx_Octet_Count_Good_Bad_High (j=0-1)	Port j Tx Octet Count Good Bad High	10818_H+j*2000_H	PMj	PMj	Kernel Reset	1952
GETH0_Portj_Tx_Packet_Count_Good_Bad_Low (j=0-1)	Port j Tx Packet Count Good Bad Low	$1081C_H$+j*2000_H	PMj	PMj	Kernel Reset	1953
GETH0_Portj_Tx_Broadcast_Packets_Good_Low (j=0-1)	Port j Tx Broadcast Packets Good Low	10824_H+j*2000_H	PMj	PMj	Kernel Reset	1953
GETH0_Portj_Tx_Multicast_Packets_Good_Low (j=0-1)	Port j Tx Multicast Packets Good Low	$1082C_H$+j*2000_H	PMj	PMj	Kernel Reset	1954
GETH0_Portj_Tx_64Octets_Packets_Good_Bad_Low (j=0-1)	Port j Tx 64Octets Packets Good Bad Low	10834_H+j*2000_H	PMj	PMj	Kernel Reset	1954
GETH0_Portj_Tx_65To127Octets_Packets_Good_Bad_Low (j=0-1)	Port j Tx 65To127Octets Packets Good Bad Low	$1083C_H$+j*2000_H	PMj	PMj	Kernel Reset	1954
GETH0_Portj_Tx_128To255Octets_Packets_Good_Bad_Low (j=0-1)	Port j Tx 128To255Octets Packets Good Bad Low	10844_H+j*2000_H	PMj	PMj	Kernel Reset	1955

(table continues...)

Table 781 (continued) Registers overview - GETH0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
GETH0_Portj_Tx_256To511Octets_Packets_Good_Bad_Low (j=0-1)	Port j Tx 256To511Octets Packets Good Bad Low	$1084C_H$+j*2000_H	PMj	PMj	Kernel Reset	1955
GETH0_Portj_Tx_512To1023Octets_Packets_Good_Bad_Low (j=0-1)	Port j Tx 512To1023Octets Packets Good Bad Low	10854_H+j*2000_H	PMj	PMj	Kernel Reset	1956
GETH0_Portj_Tx_1024ToMaxOctets_Packets_Good_Bad_Low (j=0-1)	Port j Tx 1024ToMaxOctets Packets Good Bad Low	$1085C_H$+j*2000_H	PMj	PMj	Kernel Reset	1956
GETH0_Portj_Tx_Unicast_Packets_Good_Bad_Low (j=0-1)	Port j Tx Unicast Packets Good Bad Low	10864_H+j*2000_H	PMj	PMj	Kernel Reset	1957
GETH0_Portj_Tx_Multicast_Packets_Good_Bad_Low (j=0-1)	Port j Tx Multicast Packets Good Bad Low	$1086C_H$+j*2000_H	PMj	PMj	Kernel Reset	1957
GETH0_Portj_Tx_Broadcast_Packets_Good_Bad_Low (j=0-1)	Port j Tx Broadcast Packets Good Bad Low	10874_H+j*2000_H	PMj	PMj	Kernel Reset	1958
GETH0_Portj_Tx_Underflow_Error_Packets_Low (j=0-1)	Port j Tx Underflow Error Packets Low	$1087C_H$+j*2000_H	PMj	PMj	Kernel Reset	1958
GETH0_Portj_Tx_Octet_Count_Good_Low (j=0-1)	Port j Tx Octet Count Good Low	10884_H+j*2000_H	PMj	PMj	Kernel Reset	1959
GETH0_Portj_Tx_Octet_Count_Good_High (j=0-1)	Port j Tx Octet Count Good High	10888_H+j*2000_H	PMj	PMj	Kernel Reset	1959

(table continues...)

Table 781 (continued) Registers overview - GETH0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
GETH0_Portj_Tx_Packet_Count_Good_Low (j=0-1)	Port j Tx Packet Count Good Low	$1088C_H$+j*2000_H	PMj	PMj	Kernel Reset	1959
GETH0_Portj_Tx_Pause_Packets_Low (j=0-1)	Port j Tx Pause Packets Low	10894_H+j*2000_H	PMj	PMj	Kernel Reset	1960
GETH0_Portj_Tx_VLAN_Packets_Good_Low (j=0-1)	Port j Tx VLAN Packets Good Low	$1089C_H$+j*2000_H	PMj	PMj	Kernel Reset	1960
GETH0_Portj_Tx_LPI_USEC_Cntr (j=0-1)	Port j Tx LPI USEC Cntr	$108A4_H$+j*2000_H	PMj	PMj	Kernel Reset	1961
GETH0_Portj_Tx_LPI_Tran_Cntr (j=0-1)	Port j Tx LPI Tran Cntr	$108A8_H$+j*2000_H	PMj	PMj	Kernel Reset	1961
GETH0_Portj_Rx_Packet_Count_Good_Bad_Low (j=0-1)	Port j Rx Packet Count Good Bad Low	10900_H+j*2000_H	PMj	PMj	Kernel Reset	1962
GETH0_Portj_Rx_Octet_Count_Good_Bad_Low (j=0-1)	Port j Rx Octet Count Good Bad Low	10908_H+j*2000_H	PMj	PMj	Kernel Reset	1962
GETH0_Portj_Rx_Octet_Count_Good_Bad_High (j=0-1)	Port j Rx Octet Count Good Bad High	$1090C_H$+j*2000_H	PMj	PMj	Kernel Reset	1962
GETH0_Portj_Rx_Octet_Count_Good_Low (j=0-1)	Port j Rx Octet Count Good Low	10910_H+j*2000_H	PMj	PMj	Kernel Reset	1963
GETH0_Portj_Rx_Octet_Count_Good_High (j=0-1)	Port j Rx Octet Count Good High	10914_H+j*2000_H	PMj	PMj	Kernel Reset	1963
GETH0_Portj_Rx_Broadcast_Packets_Good_Low (j=0-1)	Port j Rx Broadcast Packets Good Low	10918_H+j*2000_H	PMj	PMj	Kernel Reset	1964

(table continues...)

Table 781 (continued) Registers overview - GETH0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
GETH0_Portj_Rx_Multicast_Packets_Good_Low (j=0-1)	Port j Rx Multicast Packets Good Low	10920_H+j*2000_H	PMj	PMj	Kernel Reset	1964
GETH0_Portj_Rx_CRC_Error_Packets_Low (j=0-1)	Port j Rx CRC Error Packets Low	10928_H+j*2000_H	PMj	PMj	Kernel Reset	1965
GETH0_Portj_Rx_Runt_Error_Packets (j=0-1)	Port j Rx Runt Error Packets	10930_H+j*2000_H	PMj	PMj	Kernel Reset	1965
GETH0_Portj_Rx_Jabber_Error_Packets (j=0-1)	Port j Rx Jabber Error Packets	10934_H+j*2000_H	PMj	PMj	Kernel Reset	1965
GETH0_Portj_Rx_Undersize_Packets_Good (j=0-1)	Port j Rx Undersize Packets Good	10938_H+j*2000_H	PMj	PMj	Kernel Reset	1966
GETH0_Portj_Rx_Oversize_Packets_Good (j=0-1)	Port j Rx Oversize Packets Good	$1093C_H$+j*2000_H	PMj	PMj	Kernel Reset	1966
GETH0_Portj_Rx_64Octets_Packets_Good_Bad_Low (j=0-1)	Port j Rx 64Octets Packets Good Bad Low	10940_H+j*2000_H	PMj	PMj	Kernel Reset	1967
GETH0_Portj_Rx_65To127Octets_Packets_Good_Bad_Low (j=0-1)	Port j Rx 65To127Octets Packets Good Bad Low	10948_H+j*2000_H	PMj	PMj	Kernel Reset	1967
GETH0_Portj_Rx_128To255Octets_Packets_Good_Bad_Low (j=0-1)	Port j Rx 128To255Octets Packets Good Bad Low	10950_H+j*2000_H	PMj	PMj	Kernel Reset	1968
GETH0_Portj_Rx_256To511Octets_Packets_Good_Bad_Low (j=0-1)	Port j Rx 256To511Octets Packets Good Bad Low	10958_H+j*2000_H	PMj	PMj	Kernel Reset	1968

(table continues...)

Table 781 (continued) Registers overview - GETH0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
GETH0_Portj_Rx_512To1023Octets_Packets_Good_Bad_Low (j=0-1)	Port j Rx 512To1023Octets Packets Good Bad Low	10960_H+j *2000_H	PMj	PMj	Kernel Reset	1968
GETH0_Portj_Rx_1024ToMaxOctets_Packets_Good_Bad_Low (j=0-1)	Port j Rx 1024ToMaxOctets Packets Good Bad Low	10968_H+j *2000_H	PMj	PMj	Kernel Reset	1969
GETH0_Portj_Rx_Unicast_Packets_Good_Low (j=0-1)	Port j Rx Unicast Packets Good Low	10970_H+j *2000_H	PMj	PMj	Kernel Reset	1969
GETH0_Portj_Rx_Length_Error_Packets_Low (j=0-1)	Port j Rx Length Error Packets Low	10978_H+j *2000_H	PMj	PMj	Kernel Reset	1970
GETH0_Portj_Rx_OutofRange_Packets_Low (j=0-1)	Port j Rx OutofRange Packets Low	10980_H+j *2000_H	PMj	PMj	Kernel Reset	1970
GETH0_Portj_Rx_Pause_Packets_Low (j=0-1)	Port j Rx Pause Packets Low	10988_H+j *2000_H	PMj	PMj	Kernel Reset	1971
GETH0_Portj_Rx_FIFOOverflow_Packets_Low (j=0-1)	Port j Rx FIFOOverflow Packets Low	10990_H+j *2000_H	PMj	PMj	Kernel Reset	1971
GETH0_Portj_Rx_VLAN_Packets_Good_Bad_Low (j=0-1)	Port j Rx VLAN Packets Good Bad Low	10998_H+j *2000_H	PMj	PMj	Kernel Reset	1972
GETH0_Portj_Rx_Watchdog_Error_Packets (j=0-1)	Port j Rx Watchdog Error Packets	$109A0_H$+ j*2000_H	PMj	PMj	Kernel Reset	1972
GETH0_Portj_Rx_LPI_USEC_Cntr (j=0-1)	Port j Rx LPI USEC Cntr	$109A4_H$+ j*2000_H	PMj	PMj	Kernel Reset	1972
GETH0_Portj_Rx_LPI_Tran_Cntr (j=0-1)	Port j Rx LPI Tran Cntr	$109A8_H$+ j*2000_H	PMj	PMj	Kernel Reset	1973

(table continues...)

Table 781 (continued) Registers overview - GETH0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
GETH0_Portj_Rx_Discard_Packets_Good_Bad_Low (j=0-1)	Port j Rx Discard Packets Good Bad Low	$109AC_H$+j*2000_H	PMj	PMj	Kernel Reset	1973
GETH0_Portj_Rx_Discard_Octets_Good_Bad_Low (j=0-1)	Port j Rx Discard Octets Good Bad Low	$109B4_H$+j*2000_H	PMj	PMj	Kernel Reset	1974
GETH0_Portj_Rx_Discard_Octets_Good_Bad_High (j=0-1)	Port j Rx Discard Octets Good Bad High	$109B8_H$+j*2000_H	PMj	PMj	Kernel Reset	1974
GETH0_Portj_Rx_Alignment_Error_Packets (j=0-1)	Port j Rx Alignment Error Packets	$109BC_H$+j*2000_H	PMj	PMj	Kernel Reset	1975
GETH0_Portj_SGF_Pass_PktCnt (j=0-1)	Port j SGF Pass PktCnt	$109F0_H$+j*2000_H	PMj	PMj	Kernel Reset	1975
GETH0_Portj_SGF_Fail_PktCnt (j=0-1)	Port j SGF Fail PktCnt	$109F4_H$+j*2000_H	PMj	PMj	Kernel Reset	1976
GETH0_Portj_MMC_FPE_Tx_Interrupt (j=0-1)	Port j MMC FPE Tx Interrupt	$10A00_H$+j*2000_H	PMj	PMj	Kernel Reset	1976
GETH0_Portj_MMC_FPE_Tx_Interrupt_Mask (j=0-1)	Port j MMC FPE Tx Interrupt Mask	$10A04_H$+j*2000_H	PMj	PMj	Kernel Reset	1977
GETH0_Portj_MMC_Tx_FPE_Fragment_Cntr (j=0-1)	Port j MMC Tx FPE Fragment Cntr	$10A08_H$+j*2000_H	PMj	PMj	Kernel Reset	1978
GETH0_Portj_MMC_Tx_Hold_Req_Cntr (j=0-1)	Port j MMC Tx Hold Req Cntr	$10A0C_H$+j*2000_H	PMj	PMj	Kernel Reset	1979
GETH0_Portj_MMC_Tx_Gate_Orun_Cntr_Low (j=0-1)	Port j MMC Tx Gate Orun Cntr Low	$10A10_H$+j*2000_H	PMj	PMj	Kernel Reset	1979

(table continues...)

Table 781 (continued) Registers overview - GETH0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
GETH0_Portj_MMC_Tx_Gate_Orun_Cntr_High (j=0-1)	Port j MMC Tx Gate Orun Cntr High	$10A14_H$+ j*2000_H	PMj	PMj	Kernel Reset	1980
GETH0_Portj_MMC_FPE_Rx_Interrupt (j=0-1)	Port j MMC FPE Rx Interrupt	$10A20_H$+ j*2000_H	PMj	PMj	Kernel Reset	1980
GETH0_Portj_MMC_FPE_Rx_Interrupt_Mask (j=0-1)	Port j MMC FPE Rx Interrupt Mask	$10A24_H$+ j*2000_H	PMj	PMj	Kernel Reset	1982
GETH0_Portj_MMC_Rx_Packet_Assembly_Err_Cntr (j=0-1)	Port j MMC Rx Packet Assembly Err Cntr	$10A28_H$+ j*2000_H	PMj	PMj	Kernel Reset	1983
GETH0_Portj_MMC_Rx_Packet_SMD_Err_Cntr (j=0-1)	Port j MMC Rx Packet SMD Err Cntr	$10A2C_H$+ j*2000_H	PMj	PMj	Kernel Reset	1984
GETH0_Portj_MMC_Rx_Packet_Assembly_OK_Cntr (j=0-1)	Port j MMC Rx Packet Assembly OK Cntr	$10A30_H$+ j*2000_H	PMj	PMj	Kernel Reset	1984
GETH0_Portj_MMC_Rx_FPE_Fragment_Cntr (j=0-1)	Port j MMC Rx FPE Fragment Cntr	$10A34_H$+ j*2000_H	PMj	PMj	Kernel Reset	1985
GETH0_Portj_Tx_Single_Collision_Good_Packets (j=0-1)	Port j Tx Single Collision Good Packets	$10A40_H$+ j*2000_H	PMj	PMj	Kernel Reset	1985
GETH0_Portj_Tx_Multiple_Collision_Good_Packets (j=0-1)	Port j Tx Multiple Collision Good Packets	$10A44_H$+ j*2000_H	PMj	PMj	Kernel Reset	1986
GETH0_Portj_Tx_Deferred_Packets (j=0-1)	Port j Tx Deferred Packets	$10A48_H$+ j*2000_H	PMj	PMj	Kernel Reset	1986

(table continues...)

Table 781 (continued) Registers overview - GETH0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
GETH0_Portj_Tx_Late_Collision_Packets (j=0-1)	Port j Tx Late Collision Packets	$10A4C_H$+ j*2000_H	PMj	PMj	Kernel Reset	1987
GETH0_Portj_Tx_Excessive_Collision_Packets (j=0-1)	Port j Tx Excessive Collision Packets	$10A50_H$+ j*2000_H	PMj	PMj	Kernel Reset	1987
GETH0_Portj_Tx_Carrier_Error_Packets (j=0-1)	Port j Tx Carrier Error Packets	$10A54_H$+ j*2000_H	PMj	PMj	Kernel Reset	1987
GETH0_Portj_Tx_Excessive_Deferral_Error (j=0-1)	Port j Tx Excessive Deferral Error	$10A58_H$+ j*2000_H	PMj	PMj	Kernel Reset	1988
GETH0_Portj_MMC_IPC_Rx_Interrupt_Mask (j=0-1)	Port j MMC IPC Rx Interrupt Mask	$10A5C_H$+ j*2000_H	PMj	PMj	Kernel Reset	1988
GETH0_Portj_MMC_IPC_Rx_Interrupt (j=0-1)	Port j MMC IPC Rx Interrupt	$10A60_H$+ j*2000_H	PMj	PMj	Kernel Reset	1994
GETH0_Portj_RxIPv4_Good_Packets_Low (j=0-1)	Port j RxIPv4 Good Packets Low	$10A64_H$+ j*2000_H	PMj	PMj	Kernel Reset	2000
GETH0_Portj_RxIPv4_Header_Error_Packets_Low (j=0-1)	Port j RxIPv4 Header Error Packets Low	$10A6C_H$+ j*2000_H	PMj	PMj	Kernel Reset	2001
GETH0_Portj_RxIPv4_No_Payload_Packets_Low (j=0-1)	Port j RxIPv4 No Payload Packets Low	$10A74_H$+ j*2000_H	PMj	PMj	Kernel Reset	2001
GETH0_Portj_RxIPv4_Fragmented_Packets_Low (j=0-1)	Port j RxIPv4 Fragmented Packets Low	$10A7C_H$+ j*2000_H	PMj	PMj	Kernel Reset	2002

(table continues...)

Table 781 (continued) Registers overview - GETH0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
GETH0_Portj_RxIPv4_UDP_Checksum_Disabled_Packets_Low (j=0-1)	Port j RxIPv4 UDP Checksum Disabled Packets Low	$10A84_H$+ j*2000_H	PMj	PMj	Kernel Reset	2002
GETH0_Portj_RxIPv6_Good_Packets_Low (j=0-1)	Port j RxIPv6 Good Packets Low	$10A8C_H$+ j*2000_H	PMj	PMj	Kernel Reset	2002
GETH0_Portj_RxIPv6_Header_Error_Packets_Low (j=0-1)	Port j RxIPv6 Header Error Packets Low	$10A94_H$+ j*2000_H	PMj	PMj	Kernel Reset	2003
GETH0_Portj_RxIPv6_No_Payload_Packets_Low (j=0-1)	Port j RxIPv6 No Payload Packets Low	$10A9C_H$+ j*2000_H	PMj	PMj	Kernel Reset	2003
GETH0_Portj_RxUDP_Good_Packets_Low (j=0-1)	Port j RxUDP Good Packets Low	$10AA4_H$+ j*2000_H	PMj	PMj	Kernel Reset	2004
GETH0_Portj_RxUDP_Error_Packets_Low (j=0-1)	Port j RxUDP Error Packets Low	$10AAC_H$+ j*2000_H	PMj	PMj	Kernel Reset	2004
GETH0_Portj_RxTCP_Good_Packets_Low (j=0-1)	Port j RxTCP Good Packets Low	$10AB4_H$+ j*2000_H	PMj	PMj	Kernel Reset	2005
GETH0_Portj_RxTCP_Error_Packets_Low (j=0-1)	Port j RxTCP Error Packets Low	$10ABC_H$+ j*2000_H	PMj	PMj	Kernel Reset	2005
GETH0_Portj_RxICMP_Good_Packets_Low (j=0-1)	Port j RxICMP Good Packets Low	$10AC4_H$+ j*2000_H	PMj	PMj	Kernel Reset	2005
GETH0_Portj_RxICMP_Error_Packets_Low (j=0-1)	Port j RxICMP Error Packets Low	$10ACC_H$+ j*2000_H	PMj	PMj	Kernel Reset	2006

(table continues...)

Table 781 (continued) Registers overview - GETH0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
GETH0_Portj_RxIPv4_Good_Octets_Low (j=0-1)	Port j RxIPv4 Good Octets Low	$10AD4_H$+ j*2000_H	PMj	PMj	Kernel Reset	2006
GETH0_Portj_RxIPv4_Header_Error_Octets_Low (j=0-1)	Port j RxIPv4 Header Error Octets Low	$10ADC_H$+ j*2000_H	PMj	PMj	Kernel Reset	2007
GETH0_Portj_RxIPv4_No_Payload_Octets_Low (j=0-1)	Port j RxIPv4 No Payload Octets Low	$10AE4_H$+ j*2000_H	PMj	PMj	Kernel Reset	2007
GETH0_Portj_RxIPv4_Fragmented_Octets_Low (j=0-1)	Port j RxIPv4 Fragmented Octets Low	$10AEC_H$+ j*2000_H	PMj	PMj	Kernel Reset	2008
GETH0_Portj_RxIPv4_UDP_Checksum_Disable_Octets_Low (j=0-1)	Port j RxIPv4 UDP Checksum Disable Octets Low	$10AF4_H$+ j*2000_H	PMj	PMj	Kernel Reset	2008
GETH0_Portj_RxIPv6_Good_Octets_Low (j=0-1)	Port j RxIPv6 Good Octets Low	$10AFC_H$+ j*2000_H	PMj	PMj	Kernel Reset	2009
GETH0_Portj_RxIPv6_Header_Error_Octets_Low (j=0-1)	Port j RxIPv6 Header Error Octets Low	$10B04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2009
GETH0_Portj_RxIPv6_No_Payload_Octets_Low (j=0-1)	Port j RxIPv6 No Payload Octets Low	$10B0C_H$+ j*2000_H	PMj	PMj	Kernel Reset	2010
GETH0_Portj_RxUDP_Good_Octets_Low (j=0-1)	Port j RxUDP Good Octets Low	$10B14_H$+ j*2000_H	PMj	PMj	Kernel Reset	2010
GETH0_Portj_RxUDP_Error_Octets_Low (j=0-1)	Port j RxUDP Error Octets Low	$10B1C_H$+ j*2000_H	PMj	PMj	Kernel Reset	2011

(table continues...)

Table 781 (continued) Registers overview - GETH0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
GETH0_Portj_RxTCP_Good_Octets_Low (j=0-1)	Port j RxTCP Good Octets Low	$10B24_H$+ j*2000_H	PMj	PMj	Kernel Reset	2011
GETH0_Portj_RxTCP_Error_Octets_Low (j=0-1)	Port j RxTCP Error Octets Low	$10B2C_H$+ j*2000_H	PMj	PMj	Kernel Reset	2012
GETH0_Portj_RxICMP_Good_Octets_Low (j=0-1)	Port j RxICMP Good Octets Low	$10B34_H$+ j*2000_H	PMj	PMj	Kernel Reset	2012
GETH0_Portj_RxICMP_Error_Octets_Low (j=0-1)	Port j RxICMP Error Octets Low	$10B3C_H$+ j*2000_H	PMj	PMj	Kernel Reset	2012
GETH0_Portj_MAC_L3_L4_Address_Control (j=0-1)	Port j MAC L3 L4 Address Control	$10C00_H$+ j*2000_H	PMj	PMj	Kernel Reset	2013
GETH0_Portj_MAC_L3_L4_Data (j=0-1)	Port j MAC L3 L4 Data	$10C04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2014
GETH0_Portj_MAC_L3_L4_Control0 (j=0-1)	Port j MAC L3 L4 Control0	$10C04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2014
GETH0_Portj_MAC_L3_L4_Control1 (j=0-1)	Port j MAC L3 L4 Control1	$10C04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2017
GETH0_Portj_MAC_L3_L4_Control2 (j=0-1)	Port j MAC L3 L4 Control2	$10C04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2020
GETH0_Portj_MAC_L3_L4_Control3 (j=0-1)	Port j MAC L3 L4 Control3	$10C04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2023
GETH0_Portj_MAC_L3_L4_Control4 (j=0-1)	Port j MAC L3 L4 Control4	$10C04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2026

(table continues...)

Table 781 (continued) Registers overview - GETH0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
GETH0_Portj_MAC_L3_L4_Control5 (j=0-1)	Port j MAC L3 L4 Control5	$10C04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2029
GETH0_Portj_MAC_L3_L4_Control6 (j=0-1)	Port j MAC L3 L4 Control6	$10C04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2032
GETH0_Portj_MAC_L3_L4_Control7 (j=0-1)	Port j MAC L3 L4 Control7	$10C04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2035
GETH0_Portj_MAC_Layer3_Addr0_Reg0 (j=0-1)	Port j MAC Layer3 Addr0 Reg0	$10C04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2038
GETH0_Portj_MAC_Layer3_Addr0_Reg1 (j=0-1)	Port j MAC Layer3 Addr0 Reg1	$10C04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2039
GETH0_Portj_MAC_Layer3_Addr0_Reg2 (j=0-1)	Port j MAC Layer3 Addr0 Reg2	$10C04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2040
GETH0_Portj_MAC_Layer3_Addr0_Reg3 (j=0-1)	Port j MAC Layer3 Addr0 Reg3	$10C04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2040
GETH0_Portj_MAC_Layer3_Addr0_Reg4 (j=0-1)	Port j MAC Layer3 Addr0 Reg4	$10C04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2041
GETH0_Portj_MAC_Layer3_Addr0_Reg5 (j=0-1)	Port j MAC Layer3 Addr0 Reg5	$10C04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2041
GETH0_Portj_MAC_Layer3_Addr0_Reg6 (j=0-1)	Port j MAC Layer3 Addr0 Reg6	$10C04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2042
GETH0_Portj_MAC_Layer3_Addr0_Reg7 (j=0-1)	Port j MAC Layer3 Addr0 Reg7	$10C04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2043

(table continues...)

Table 781 (continued) Registers overview - GETH0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
GETH0_Portj_MAC_Layer3_Addr1_Reg0 (j=0-1)	Port j MAC Layer3 Addr1 Reg0	$10C04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2043
GETH0_Portj_MAC_Layer3_Addr1_Reg1 (j=0-1)	Port j MAC Layer3 Addr1 Reg1	$10C04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2044
GETH0_Portj_MAC_Layer3_Addr1_Reg2 (j=0-1)	Port j MAC Layer3 Addr1 Reg2	$10C04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2044
GETH0_Portj_MAC_Layer3_Addr1_Reg3 (j=0-1)	Port j MAC Layer3 Addr1 Reg3	$10C04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2045
GETH0_Portj_MAC_Layer3_Addr1_Reg4 (j=0-1)	Port j MAC Layer3 Addr1 Reg4	$10C04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2046
GETH0_Portj_MAC_Layer3_Addr1_Reg5 (j=0-1)	Port j MAC Layer3 Addr1 Reg5	$10C04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2046
GETH0_Portj_MAC_Layer3_Addr1_Reg6 (j=0-1)	Port j MAC Layer3 Addr1 Reg6	$10C04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2047
GETH0_Portj_MAC_Layer3_Addr1_Reg7 (j=0-1)	Port j MAC Layer3 Addr1 Reg7	$10C04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2047
GETH0_Portj_MAC_Layer3_Addr2_Reg0 (j=0-1)	Port j MAC Layer3 Addr2 Reg0	$10C04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2048
GETH0_Portj_MAC_Layer3_Addr2_Reg1 (j=0-1)	Port j MAC Layer3 Addr2 Reg1	$10C04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2049
GETH0_Portj_MAC_Layer3_Addr2_Reg2 (j=0-1)	Port j MAC Layer3 Addr2 Reg2	$10C04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2049

(table continues...)

Table 781 (continued) Registers overview - GETH0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
GETH0_Portj_MAC_Layer3_Addr2_Reg3 (j=0-1)	Port j MAC Layer3 Addr2 Reg3	$10C04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2050
GETH0_Portj_MAC_Layer3_Addr2_Reg4 (j=0-1)	Port j MAC Layer3 Addr2 Reg4	$10C04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2050
GETH0_Portj_MAC_Layer3_Addr2_Reg5 (j=0-1)	Port j MAC Layer3 Addr2 Reg5	$10C04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2051
GETH0_Portj_MAC_Layer3_Addr2_Reg6 (j=0-1)	Port j MAC Layer3 Addr2 Reg6	$10C04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2052
GETH0_Portj_MAC_Layer3_Addr2_Reg7 (j=0-1)	Port j MAC Layer3 Addr2 Reg7	$10C04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2052
GETH0_Portj_MAC_Layer3_Addr3_Reg0 (j=0-1)	Port j MAC Layer3 Addr3 Reg0	$10C04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2053
GETH0_Portj_MAC_Layer3_Addr3_Reg1 (j=0-1)	Port j MAC Layer3 Addr3 Reg1	$10C04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2053
GETH0_Portj_MAC_Layer3_Addr3_Reg2 (j=0-1)	Port j MAC Layer3 Addr3 Reg2	$10C04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2054
GETH0_Portj_MAC_Layer3_Addr3_Reg3 (j=0-1)	Port j MAC Layer3 Addr3 Reg3	$10C04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2055
GETH0_Portj_MAC_Layer3_Addr3_Reg4 (j=0-1)	Port j MAC Layer3 Addr3 Reg4	$10C04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2055
GETH0_Portj_MAC_Layer3_Addr3_Reg5 (j=0-1)	Port j MAC Layer3 Addr3 Reg5	$10C04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2056

(table continues...)

Table 781 (continued) Registers overview - GETH0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
GETH0_Portj_MAC_Layer3_Addr3_Reg6 (j=0-1)	Port j MAC Layer3 Addr3 Reg6	$10C04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2056
GETH0_Portj_MAC_Layer3_Addr3_Reg7 (j=0-1)	Port j MAC Layer3 Addr3 Reg7	$10C04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2057
GETH0_Portj_MAC_Layer4_Address0 (j=0-1)	Port j MAC Layer4 Address0	$10C04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2058
GETH0_Portj_MAC_Layer4_Address1 (j=0-1)	Port j MAC Layer4 Address1	$10C04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2058
GETH0_Portj_MAC_Layer4_Address2 (j=0-1)	Port j MAC Layer4 Address2	$10C04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2059
GETH0_Portj_MAC_Layer4_Address3 (j=0-1)	Port j MAC Layer4 Address3	$10C04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2060
GETH0_Portj_MAC_Layer4_Address4 (j=0-1)	Port j MAC Layer4 Address4	$10C04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2061
GETH0_Portj_MAC_Layer4_Address5 (j=0-1)	Port j MAC Layer4 Address5	$10C04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2061
GETH0_Portj_MAC_Layer4_Address6 (j=0-1)	Port j MAC Layer4 Address6	$10C04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2062
GETH0_Portj_MAC_Layer4_Address7 (j=0-1)	Port j MAC Layer4 Address7	$10C04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2063
GETH0_Portj_MAC_Timestamp_Control (j=0-1)	Port j MAC Timestamp Control	$10D00_H$+ j*2000_H	PMj	PMj	Kernel Reset	2064

(table continues...)

Table 781 (continued) Registers overview - GETH0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
GETH0_Portj_MAC_Sub_Second_Increment (j=0-1)	Port j MAC Sub Second Increment	$10D04_H$+ j*2000_H	PMj	PMj	Kernel Reset	2067
GETH0_Portj_MAC_System_Time_Seconds (j=0-1)	Port j MAC System Time Seconds	$10D08_H$+ j*2000_H	PMj	PMj	Kernel Reset	2068
GETH0_Portj_MAC_System_Time_Nanoseconds (j=0-1)	Port j MAC System Time Nanoseconds	$10D0C_H$+ j*2000_H	PMj	PMj	Kernel Reset	2068
GETH0_Portj_MAC_System_Time_Seconds_Update (j=0-1)	Port j MAC System Time Seconds Update	$10D10_H$+ j*2000_H	PMj	PMj	Kernel Reset	2069
GETH0_Portj_MAC_System_Time_Nanoseconds_Update (j=0-1)	Port j MAC System Time Nanoseconds Update	$10D14_H$+ j*2000_H	PMj	PMj	Kernel Reset	2069
GETH0_Portj_MAC_Timestamp_Addend (j=0-1)	Port j MAC Timestamp Addend	$10D18_H$+ j*2000_H	PMj	PMj	Kernel Reset	2070
GETH0_Portj_MAC_System_Time_Higher_Word_Seconds (j=0-1)	Port j MAC System Time Higher Word Seconds	$10D1C_H$+ j*2000_H	PMj	PMj	Kernel Reset	2070
GETH0_Portj_MAC_Timestamp_Status (j=0-1)	Port j MAC Timestamp Status	$10D20_H$+ j*2000_H	PMj	PMj	Kernel Reset	2071
GETH0_Portj_MAC_Rx_Domain_Time_Incr (j=0-1)	Port j MAC Rx Domain Time Incr	$10D24_H$+ j*2000_H	PMj	PMj	Kernel Reset	2074
GETH0_Portj_MAC_Tx_Domain_Time_Incr (j=0-1)	Port j MAC Tx Domain Time Incr	$10D28_H$+ j*2000_H	PMj	PMj	Kernel Reset	2075

(table continues...)

Table 781 (continued) Registers overview - GETH0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
GETH0_Portj_MAC_Tx_Timestamp_Status_Nanoseconds (j=0-1)	Port j MAC Tx Timestamp Status Nanoseconds	$10D30_H$+ j*2000_H	PMj	PMj	Kernel Reset	2076
GETH0_Portj_MAC_Tx_Timestamp_Status_Seconds (j=0-1)	Port j MAC Tx Timestamp Status Seconds	$10D34_H$+ j*2000_H	PMj	PMj	Kernel Reset	2077
GETH0_Portj_MAC_Tx_Timestamp_Status_PktID (j=0-1)	Port j MAC Tx Timestamp Status PktID	$10D38_H$+ j*2000_H	PMj	PMj	Kernel Reset	2077
GETH0_Portj_MAC_Timestamp_Ingress_Asym_Corr (j=0-1)	Port j MAC Timestamp Ingress Asym Corr	$10D50_H$+ j*2000_H	PMj	PMj	Kernel Reset	2078
GETH0_Portj_MAC_Timestamp_Egress_Asym_Corr (j=0-1)	Port j MAC Timestamp Egress Asym Corr	$10D54_H$+ j*2000_H	PMj	PMj	Kernel Reset	2079
GETH0_Portj_MAC_Timestamp_Ingress_Corr_Nanosecond (j=0-1)	Port j MAC Timestamp Ingress Corr Nanosecond	$10D58_H$+ j*2000_H	PMj	PMj	Kernel Reset	2079
GETH0_Portj_MAC_Timestamp_Ingress_Corr_Subnanosecond (j=0-1)	Port j MAC Timestamp Ingress Corr Subnanosecond	$10D5C_H$+ j*2000_H	PMj	PMj	Kernel Reset	2080
GETH0_Portj_MAC_Timestamp_Egress_Corr_Nanosecond (j=0-1)	Port j MAC Timestamp Egress Corr Nanosecond	$10D60_H$+ j*2000_H	PMj	PMj	Kernel Reset	2080
GETH0_Portj_MAC_Timestamp_Egress_Corr_Subnanosecond (j=0-1)	Port j MAC Timestamp Egress Corr Subnanosecond	$10D64_H$+ j*2000_H	PMj	PMj	Kernel Reset	2081
GETH0_Portj_MAC_PPS_Control (j=0-1)	Port j MAC PPS Control	$10D70_H$+ j*2000_H	PMj	PMj	Kernel Reset	2081

(table continues...)

Table 781 (continued) Registers overview - GETH0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
GETH0_Portj_MAC_PPS_Extnd_Ctrl (j=0-1)	Port j MAC PPS Extnd Ctrl	$10D74_H$+ j*2000_H	PMj	PMj	Kernel Reset	2086
GETH0_Portj_MAC_PPS0_Target_Time_Seconds (j=0-1)	Port j MAC PPS0 Target Time Seconds	$10D80_H$+ j*2000_H	PMj	PMj	Kernel Reset	2086
GETH0_Portj_MAC_PPS0_Target_Time_Nanoseconds (j=0-1)	Port j MAC PPS0 Target Time Nanoseconds	$10D84_H$+ j*2000_H	PMj	PMj	Kernel Reset	2087
GETH0_Portj_MAC_PPS0_Interval (j=0-1)	Port j MAC PPS0 Interval	$10D88_H$+ j*2000_H	PMj	PMj	Kernel Reset	2088
GETH0_Portj_MAC_PPS0_Width (j=0-1)	Port j MAC PPS0 Width	$10D8C_H$+ j*2000_H	PMj	PMj	Kernel Reset	2089
GETH0_Portj_MAC_PPS1_Target_Time_Seconds (j=0-1)	Port j MAC PPS1 Target Time Seconds	$10D90_H$+ j*2000_H	PMj	PMj	Kernel Reset	2089
GETH0_Portj_MAC_PPS1_Target_Time_Nanoseconds (j=0-1)	Port j MAC PPS1 Target Time Nanoseconds	$10D94_H$+ j*2000_H	PMj	PMj	Kernel Reset	2090
GETH0_Portj_MAC_PPS1_Interval (j=0-1)	Port j MAC PPS1 Interval	$10D98_H$+ j*2000_H	PMj	PMj	Kernel Reset	2091
GETH0_Portj_MAC_PPS1_Width (j=0-1)	Port j MAC PPS1 Width	$10D9C_H$+ j*2000_H	PMj	PMj	Kernel Reset	2091
GETH0_Portj_MAC_Presn_Time_ns (j=0-1)	Port j MAC Presn Time ns	$10DE0_H$+ j*2000_H	PMj	PMj	Kernel Reset	2092
GETH0_Portj_MAC_Presn_Time_Updt (j=0-1)	Port j MAC Presn Time Updt	$10DE4_H$+ j*2000_H	PMj	PMj	Kernel Reset	2092

(table continues...)

Table 781 (continued) Registers overview - GETH0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
GETH0_Portj_MTL_Operation_Mode (j=0-1)	Port j MTL Operation Mode	11000_H+j*2000_H	PMj	PMj	Kernel Reset	2093
GETH0_Portj_MTL_Debug_Control (j=0-1)	Port j MTL Debug Control	11008_H+j*2000_H	PMj	PMj	Kernel Reset	2094
GETH0_Portj_MTL_Debug_Status (j=0-1)	Port j MTL Debug Status	$1100C_H$+j*2000_H	PMj	PMj	Kernel Reset	2096
GETH0_Portj_MTL_FIFO_Debug_Data (j=0-1)	Port j MTL FIFO Debug Data	11010_H+j*2000_H	PMj	PMj	Kernel Reset	2098
GETH0_Portj_MTL_Interrupt_Status (j=0-1)	Port j MTL Interrupt Status	11020_H+j*2000_H	PMj	PMj	Kernel Reset	2098
GETH0_Portj_MTL_RxQ_DMA_Map0 (j=0-1)	Port j MTL RxQ DMA Map0	11030_H+j*2000_H	PMj	PMj	Kernel Reset	2101
GETH0_Portj_MTL_RxQ_DMA_Map1 (j=0-1)	Port j MTL RxQ DMA Map1	11034_H+j*2000_H	PMj	PMj	Kernel Reset	2103
GETH0_Portj_MTL_TBS_CTRL (j=0-1)	Port j MTL TBS CTRL	11048_H+j*2000_H	PMj	PMj	Kernel Reset	2104
GETH0_Portj_MTL_TBS_STATS (j=0-1)	Port j MTL TBS STATS	$1104C_H$+j*2000_H	PMj	PMj	Kernel Reset	2105
GETH0_Portj_MTL_EST_Control (j=0-1)	Port j MTL EST Control	11050_H+j*2000_H	PMj	PMj	Kernel Reset	2106
GETH0_Portj_MTL_EST_Overhead (j=0-1)	Port j MTL EST Overhead	11054_H+j*2000_H	PMj	PMj	Kernel Reset	2108
GETH0_Portj_MTL_EST_Status (j=0-1)	Port j MTL EST Status	11058_H+j*2000_H	PMj	PMj	Kernel Reset	2109

(table continues...)

Table 781 (continued) Registers overview - GETH0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
GETH0_Portj_MTL_EST_Sch_Error (j=0-1)	Port j MTL EST Sch Error	11060_H+j*2000_H	PMj	PMj	Kernel Reset	2111
GETH0_Portj_MTL_EST_Frm_Size_Error (j=0-1)	Port j MTL EST Frm Size Error	11064_H+j*2000_H	PMj	PMj	Kernel Reset	2112
GETH0_Portj_MTL_EST_Frm_Size_Capture (j=0-1)	Port j MTL EST Frm Size Capture	11068_H+j*2000_H	PMj	PMj	Kernel Reset	2112
GETH0_Portj_MTL_EST_Intr_Enable (j=0-1)	Port j MTL EST Intr Enable	11070_H+j*2000_H	PMj	PMj	Kernel Reset	2113
GETH0_Portj_MTL_GCL_Control (j=0-1)	Port j MTL GCL Control	11080_H+j*2000_H	PMj	PMj	Kernel Reset	2114
GETH0_Portj_MTL_GCL_Data (j=0-1)	Port j MTL GCL Data	11084_H+j*2000_H	PMj	PMj	Kernel Reset	2117
GETH0_Portj_MTL_FPE_CTRL_STS (j=0-1)	Port j MTL FPE CTRL STS	11090_H+j*2000_H	PMj	PMj	Kernel Reset	2117
GETH0_Portj_MTL_FPE_Advance (j=0-1)	Port j MTL FPE Advance	11094_H+j*2000_H	PMj	PMj	Kernel Reset	2118
GETH0_Portj_MTL_RXP_Control_Status (j=0-1)	Port j MTL RXP Control Status	$110A0_H$+j*2000_H	PMj	PMj	Kernel Reset	2119
GETH0_Portj_MTL_RXP_Interrupt_Control_Status (j=0-1)	Port j MTL RXP Interrupt Control Status	$110A4_H$+j*2000_H	PMj	PMj	Kernel Reset	2120
GETH0_Portj_MTL_Indirect_Acc_Control_Status (j=0-1)	Port j MTL Indirect Acc Control Status	$110B0_H$+j*2000_H	PMj	PMj	Kernel Reset	2122
GETH0_Portj_MTL_Indirect_Acc_Data (j=0-1)	Port j MTL Indirect Acc Data	$110B4_H$+j*2000_H	PMj	PMj	Kernel Reset	2124

(table continues...)

Table 781 (continued) Registers overview - GETH0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
GETH0_Portj_MTL_FRP_ACNTR0 (j=0-1)	Port j MTL FRP ACNTR0	$110B4_H$+ j*2000_H	PMj	PMj	Kernel Reset	2125
GETH0_Portj_MTL_FRP_ACNTR1 (j=0-1)	Port j MTL FRP ACNTR1	$110B4_H$+ j*2000_H	PMj	PMj	Kernel Reset	2126
GETH0_Portj_MTL_FRP_ACNTR2 (j=0-1)	Port j MTL FRP ACNTR2	$110B4_H$+ j*2000_H	PMj	PMj	Kernel Reset	2126
GETH0_Portj_MTL_FRP_ACNTR3 (j=0-1)	Port j MTL FRP ACNTR3	$110B4_H$+ j*2000_H	PMj	PMj	Kernel Reset	2127
GETH0_Portj_MTL_FRP_ACNTR4 (j=0-1)	Port j MTL FRP ACNTR4	$110B4_H$+ j*2000_H	PMj	PMj	Kernel Reset	2128
GETH0_Portj_MTL_FRP_ACNTR5 (j=0-1)	Port j MTL FRP ACNTR5	$110B4_H$+ j*2000_H	PMj	PMj	Kernel Reset	2129
GETH0_Portj_MTL_FRP_ACNTR6 (j=0-1)	Port j MTL FRP ACNTR6	$110B4_H$+ j*2000_H	PMj	PMj	Kernel Reset	2129
GETH0_Portj_MTL_FRP_ACNTR7 (j=0-1)	Port j MTL FRP ACNTR7	$110B4_H$+ j*2000_H	PMj	PMj	Kernel Reset	2130
GETH0_Portj_MTL_RXP_Bypass_Cnt (j=0-1)	Port j MTL RXP Bypass Cnt	$110B4_H$+ j*2000_H	PMj	PMj	Kernel Reset	2131
GETH0_Portj_MTL_RXP_Drop_Cnt (j=0-1)	Port j MTL RXP Drop Cnt	$110B4_H$+ j*2000_H	PMj	PMj	Kernel Reset	2132
GETH0_Portj_MTL_RXP_Error_Cnt (j=0-1)	Port j MTL RXP Error Cnt	$110B4_H$+ j*2000_H	PMj	PMj	Kernel Reset	2132
GETH0_Portj_MTL_ECC_Control (j=0-1)	Port j MTL ECC Control	$110C0_H$+ j*2000_H	PMj	PMj	Kernel Reset	2133
GETH0_Portj_MTL_Safety_Interrupt_Status (j=0-1)	Port j MTL Safety Interrupt Status	$110C4_H$+ j*2000_H	PMj	PMj	Kernel Reset	2134

(table continues...)

Table 781 (continued) Registers overview - GETH0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
GETH0_Portj_MTL_ECC_Interrupt_Enable (j=0-1)	Port j MTL ECC Interrupt Enable	$110C8_H$+ j*2000_H	PMj	PMj	Kernel Reset	2135
GETH0_Portj_MTL_ECC_Interrupt_Status (j=0-1)	Port j MTL ECC Interrupt Status	$110CC_H$+ j*2000_H	PMj	PMj	Kernel Reset	2136
GETH0_Portj_MTL_ECC_Err_Sts_Rctl (j=0-1)	Port j MTL ECC Err Sts Rctl	$110D0_H$+ j*2000_H	PMj	PMj	Kernel Reset	2139
GETH0_Portj_MTL_ECC_Err_Addr_Status (j=0-1)	Port j MTL ECC Err Addr Status	$110D4_H$+ j*2000_H	PMj	PMj	Kernel Reset	2140
GETH0_Portj_MTL_ECC_Err_Cntr_Status (j=0-1)	Port j MTL ECC Err Cntr Status	$110D8_H$+ j*2000_H	PMj	PMj	Kernel Reset	2141
GETH0_Portj_MTL_DPP_Control (j=0-1)	Port j MTL DPP Control	$110E0_H$+j *2000_H	PMj	PMj	Kernel Reset	2142
GETH0_Portj_MTL_DPP_ECC_EIC (j=0-1)	Port j MTL DPP ECC EIC	$110E4_H$+j *2000_H	PMj	PMj	Kernel Reset	2145
GETH0_Portj_MTL_SGF_Control (j=0-1)	Port j MTL SGF Control	$110E8_H$+j *2000_H	PMj	PMj	Kernel Reset	2145
GETH0_Portj_MTL_SGF_Extended_Control (j=0-1)	Port j MTL SGF Extended Control	$110EC_H$+ j*2000_H	PMj	PMj	Kernel Reset	2147
GETH0_Portj_MTL_SGF_Status (j=0-1)	Port j MTL SGF Status	$110F0_H$+j *2000_H	PMj	PMj	Kernel Reset	2148
GETH0_Portj_MTL_SGF_Intr_Enable (j=0-1)	Port j MTL SGF Interrupt Enable	$110F4_H$+j *2000_H	PMj	PMj	Kernel Reset	2150

(table continues...)

Table 781 (continued) Registers overview - GETH0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
GETH0_Portj_MTL_TxQ0_Operation_Mode (j=0-1)	Port j MTL TxQ0 Operation Mode	11100_H+j*2000_H	PMj	PMj	Kernel Reset	2151
GETH0_Portj_MTL_TxQ0_Underflow (j=0-1)	Port j MTL TxQ0 Underflow	11104_H+j*2000_H	PMj	PMj	Kernel Reset	2152
GETH0_Portj_MTL_TxQ0_Debug (j=0-1)	Port j MTL TxQ0 Debug	11108_H+j*2000_H	PMj	PMj	Kernel Reset	2153
GETH0_Portj_MTL_TC0_ETS_Control (j=0-1)	Port j MTL TC0 ETS Control	11110_H+j*2000_H	PMj	PMj	Kernel Reset	2154
GETH0_Portj_MTL_TC0_Quantum_Weight (j=0-1)	Port j MTL TC0 Quantum Weight	11118_H+j*2000_H	PMj	PMj	Kernel Reset	2155
GETH0_Portj_MTL_RxQ0_Operation_Mode (j=0-1)	Port j MTL RxQ0 Operation Mode	11140_H+j*2000_H	PMj	PMj	Kernel Reset	2155
GETH0_Portj_MTL_RxQ0_Missed_Pkt_Overflow_Cnt (j=0-1)	Port j MTL RxQ0 Missed Pkt Overflow Cnt	11144_H+j*2000_H	PMj	PMj	Kernel Reset	2157
GETH0_Portj_MTL_RxQ0_Debug (j=0-1)	Port j MTL RxQ0 Debug	11148_H+j*2000_H	PMj	PMj	Kernel Reset	2158
GETH0_Portj_MTL_RxQ0_Control (j=0-1)	Port j MTL RxQ0 Control	$1114C_H$+j*2000_H	PMj	PMj	Kernel Reset	2159
GETH0_Portj_MTL_RxQ0_Flow_Control (j=0-1)	Port j MTL RxQ0 Flow Control	11150_H+j*2000_H	PMj	PMj	Kernel Reset	2160
GETH0_Portj_MTL_Q0_Interrupt_Enable (j=0-1)	Port j MTL Q0 Interrupt Enable	11170_H+j*2000_H	PMj	PMj	Kernel Reset	2161

(table continues...)

Table 781 (continued) Registers overview - GETH0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
GETH0_Portj_MTL_Q0_Interrupt_Status (j=0-1)	Port j MTL Q0 Interrupt Status	11174_H+j *2000_H	PMj	PMj	Kernel Reset	2161
GETH0_Portj_MTL_TxQ1_Operation_Mode (j=0-1)	Port j MTL TxQ1 Operation Mode	11180_H+j *2000_H	PMj	PMj	Kernel Reset	2162
GETH0_Portj_MTL_TxQ1_Underflow (j=0-1)	Port j MTL TxQ1 Underflow	11184_H+j *2000_H	PMj	PMj	Kernel Reset	2164
GETH0_Portj_MTL_TxQ1_Debug (j=0-1)	Port j MTL TxQ1 Debug	11188_H+j *2000_H	PMj	PMj	Kernel Reset	2164
GETH0_Portj_MTL_TC1_ETS_Control (j=0-1)	Port j MTL TC1 ETS Control	11190_H+j *2000_H	PMj	PMj	Kernel Reset	2165
GETH0_Portj_MTL_TC1_Quantum_Weight (j=0-1)	Port j MTL TC1 Quantum Weight	11198_H+j *2000_H	PMj	PMj	Kernel Reset	2166
GETH0_Portj_MTL_RxQ1_Operation_Mode (j=0-1)	Port j MTL RxQ1 Operation Mode	$111C0_H$+ j*2000_H	PMj	PMj	Kernel Reset	2167
GETH0_Portj_MTL_RxQ1_Missed_Pkt_Overflow_Cnt (j=0-1)	Port j MTL RxQ1 Missed Pkt Overflow Cnt	$111C4_H$+ j*2000_H	PMj	PMj	Kernel Reset	2168
GETH0_Portj_MTL_RxQ1_Debug (j=0-1)	Port j MTL RxQ1 Debug	$111C8_H$+ j*2000_H	PMj	PMj	Kernel Reset	2169
GETH0_Portj_MTL_RxQ1_Control (j=0-1)	Port j MTL RxQ1 Control	$111CC_H$+ j*2000_H	PMj	PMj	Kernel Reset	2170
GETH0_Portj_MTL_RxQ1_Flow_Control (j=0-1)	Port j MTL RxQ1 Flow Control	$111D0_H$+ j*2000_H	PMj	PMj	Kernel Reset	2171

(table continues...)

Table 781 (continued) Registers overview - GETH0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
GETH0_Portj_MTL_Q1_Interrupt_Enable (j=0-1)	Port j MTL Q1 Interrupt Enable	$111F0_H$+j *2000_H	PMj	PMj	Kernel Reset	2172
GETH0_Portj_MTL_Q1_Interrupt_Status (j=0-1)	Port j MTL Q1 Interrupt Status	$111F4_H$+j *2000_H	PMj	PMj	Kernel Reset	2172
GETH0_Portj_MTL_TxQ2_Operation_Mode (j=0-1)	Port j MTL TxQ2 Operation Mode	11200_H+j *2000_H	PMj	PMj	Kernel Reset	2173
GETH0_Portj_MTL_TxQ2_Underflow (j=0-1)	Port j MTL TxQ2 Underflow	11204_H+j *2000_H	PMj	PMj	Kernel Reset	2175
GETH0_Portj_MTL_TxQ2_Debug (j=0-1)	Port j MTL TxQ2 Debug	11208_H+j *2000_H	PMj	PMj	Kernel Reset	2175
GETH0_Portj_MTL_TC2_ETS_Control (j=0-1)	Port j MTL TC2 ETS Control	11210_H+j *2000_H	PMj	PMj	Kernel Reset	2176
GETH0_Portj_MTL_TC2_Quantum_Weight (j=0-1)	Port j MTL TC2 Quantum Weight	11218_H+j *2000_H	PMj	PMj	Kernel Reset	2177
GETH0_Portj_MTL_RxQ2_Operation_Mode (j=0-1)	Port j MTL RxQ2 Operation Mode	11240_H+j *2000_H	PMj	PMj	Kernel Reset	2178
GETH0_Portj_MTL_RxQ2_Missed_Pkt_Overflow_Cnt (j=0-1)	Port j MTL RxQ2 Missed Pkt Overflow Cnt	11244_H+j *2000_H	PMj	PMj	Kernel Reset	2179
GETH0_Portj_MTL_RxQ2_Debug (j=0-1)	Port j MTL RxQ2 Debug	11248_H+j *2000_H	PMj	PMj	Kernel Reset	2180
GETH0_Portj_MTL_RxQ2_Control (j=0-1)	Port j MTL RxQ2 Control	$1124C_H$+ j*2000_H	PMj	PMj	Kernel Reset	2181

(table continues...)

Table 781 (continued) Registers overview - GETH0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
GETH0_Portj_MTL_RxQ2_Flow_Control (j=0-1)	Port j MTL RxQ2 Flow Control	11250_H+j*2000_H	PMj	PMj	Kernel Reset	2182
GETH0_Portj_MTL_Q2_Interrupt_Enable (j=0-1)	Port j MTL Q2 Interrupt Enable	11270_H+j*2000_H	PMj	PMj	Kernel Reset	2183
GETH0_Portj_MTL_Q2_Interrupt_Status (j=0-1)	Port j MTL Q2 Interrupt Status	11274_H+j*2000_H	PMj	PMj	Kernel Reset	2183
GETH0_Portj_MTL_TxQ3_Operation_Mode (j=0-1)	Port j MTL TxQ3 Operation Mode	11280_H+j*2000_H	PMj	PMj	Kernel Reset	2184
GETH0_Portj_MTL_TxQ3_Underflow (j=0-1)	Port j MTL TxQ3 Underflow	11284_H+j*2000_H	PMj	PMj	Kernel Reset	2186
GETH0_Portj_MTL_TxQ3_Debug (j=0-1)	Port j MTL TxQ3 Debug	11288_H+j*2000_H	PMj	PMj	Kernel Reset	2186
GETH0_Portj_MTL_TC3_ETS_Control (j=0-1)	Port j MTL TC3 ETS Control	11290_H+j*2000_H	PMj	PMj	Kernel Reset	2187
GETH0_Portj_MTL_TC3_Quantum_Weight (j=0-1)	Port j MTL TC3 Quantum Weight	11298_H+j*2000_H	PMj	PMj	Kernel Reset	2188
GETH0_Portj_MTL_RxQ3_Operation_Mode (j=0-1)	Port j MTL RxQ3 Operation Mode	$112C0_H$+j*2000_H	PMj	PMj	Kernel Reset	2189
GETH0_Portj_MTL_RxQ3_Missed_Pkt_Overflow_Cnt (j=0-1)	Port j MTL RxQ3 Missed Pkt Overflow Cnt	$112C4_H$+j*2000_H	PMj	PMj	Kernel Reset	2190
GETH0_Portj_MTL_RxQ3_Debug (j=0-1)	Port j MTL RxQ3 Debug	$112C8_H$+j*2000_H	PMj	PMj	Kernel Reset	2191

(table continues...)

Table 781 (continued) Registers overview - GETH0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
GETH0_Portj_MTL_RxQ3_Control (j=0-1)	Port j MTL RxQ3 Control	$112CC_H$+j*2000_H	PMj	PMj	Kernel Reset	2192
GETH0_Portj_MTL_RxQ3_Flow_Control (j=0-1)	Port j MTL RxQ3 Flow Control	$112D0_H$+j*2000_H	PMj	PMj	Kernel Reset	2193
GETH0_Portj_MTL_Q3_Interrupt_Enable (j=0-1)	Port j MTL Q3 Interrupt Enable	$112F0_H$+j*2000_H	PMj	PMj	Kernel Reset	2194
GETH0_Portj_MTL_Q3_Interrupt_Status (j=0-1)	Port j MTL Q3 Interrupt Status	$112F4_H$+j*2000_H	PMj	PMj	Kernel Reset	2194
GETH0_Portj_MTL_TxQ4_Operation_Mode (j=0-1)	Port j MTL TxQ4 Operation Mode	11300_H+j*2000_H	PMj	PMj	Kernel Reset	2195
GETH0_Portj_MTL_TxQ4_Underflow (j=0-1)	Port j MTL TxQ4 Underflow	11304_H+j*2000_H	PMj	PMj	Kernel Reset	2197
GETH0_Portj_MTL_TxQ4_Debug (j=0-1)	Port j MTL TxQ4 Debug	11308_H+j*2000_H	PMj	PMj	Kernel Reset	2197
GETH0_Portj_MTL_TC4_ETS_Control (j=0-1)	Port j MTL TC4 ETS Control	11310_H+j*2000_H	PMj	PMj	Kernel Reset	2198
GETH0_Portj_MTL_TC4_Quantum_Weight (j=0-1)	Port j MTL TC4 Quantum Weight	11318_H+j*2000_H	PMj	PMj	Kernel Reset	2199
GETH0_Portj_MTL_RxQ4_Operation_Mode (j=0-1)	Port j MTL RxQ4 Operation Mode	11340_H+j*2000_H	PMj	PMj	Kernel Reset	2200
GETH0_Portj_MTL_RxQ4_Missed_Pkt_Overflow_Cnt (j=0-1)	Port j MTL RxQ4 Missed Pkt Overflow Cnt	11344_H+j*2000_H	PMj	PMj	Kernel Reset	2201

(table continues...)

Table 781 (continued) Registers overview - GETH0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
GETH0_Portj_MTL_RxQ4_Debug (j=0-1)	Port j MTL RxQ4 Debug	11348_H+j*2000_H	PMj	PMj	Kernel Reset	2202
GETH0_Portj_MTL_RxQ4_Control (j=0-1)	Port j MTL RxQ4 Control	$1134C_H$+j*2000_H	PMj	PMj	Kernel Reset	2203
GETH0_Portj_MTL_RxQ4_Flow_Control (j=0-1)	Port j MTL RxQ4 Flow Control	11350_H+j*2000_H	PMj	PMj	Kernel Reset	2204
GETH0_Portj_MTL_Q4_Interrupt_Enable (j=0-1)	Port j MTL Q4 Interrupt Enable	11370_H+j*2000_H	PMj	PMj	Kernel Reset	2205
GETH0_Portj_MTL_Q4_Interrupt_Status (j=0-1)	Port j MTL Q4 Interrupt Status	11374_H+j*2000_H	PMj	PMj	Kernel Reset	2205
GETH0_Portj_MTL_TxQ5_Operation_Mode (j=0-1)	Port j MTL TxQ5 Operation Mode	11380_H+j*2000_H	PMj	PMj	Kernel Reset	2206
GETH0_Portj_MTL_TxQ5_Underflow (j=0-1)	Port j MTL TxQ5 Underflow	11384_H+j*2000_H	PMj	PMj	Kernel Reset	2208
GETH0_Portj_MTL_TxQ5_Debug (j=0-1)	Port j MTL TxQ5 Debug	11388_H+j*2000_H	PMj	PMj	Kernel Reset	2208
GETH0_Portj_MTL_TC5_ETS_Control (j=0-1)	Port j MTL TC5 ETS Control	11390_H+j*2000_H	PMj	PMj	Kernel Reset	2209
GETH0_Portj_MTL_TC5_Quantum_Weight (j=0-1)	Port j MTL TC5 Quantum Weight	11398_H+j*2000_H	PMj	PMj	Kernel Reset	2210
GETH0_Portj_MTL_TC5_SendSlopeCredit (j=0-1)	Port j MTL TC5 SendSlopeCredit	$1139C_H$+j*2000_H	PMj	PMj	Kernel Reset	2211

(table continues...)

Table 781 (continued) Registers overview - GETH0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
GETH0_Portj_MTL_TC5_HiCredit (j=0-1)	Port j MTL TC5 HiCredit	$113A0_H$+ j*2000_H	PMj	PMj	Kernel Reset	2212
GETH0_Portj_MTL_TC5_LoCredit (j=0-1)	Port j MTL TC5 LoCredit	$113A4_H$+ j*2000_H	PMj	PMj	Kernel Reset	2212
GETH0_Portj_MTL_RxQ5_Operation_Mode (j=0-1)	Port j MTL RxQ5 Operation Mode	$113C0_H$+ j*2000_H	PMj	PMj	Kernel Reset	2213
GETH0_Portj_MTL_RxQ5_Missed_Pkt_Overflow_Cnt (j=0-1)	Port j MTL RxQ5 Missed Pkt Overflow Cnt	$113C4_H$+ j*2000_H	PMj	PMj	Kernel Reset	2215
GETH0_Portj_MTL_RxQ5_Debug (j=0-1)	Port j MTL RxQ5 Debug	$113C8_H$+ j*2000_H	PMj	PMj	Kernel Reset	2216
GETH0_Portj_MTL_RxQ5_Control (j=0-1)	Port j MTL RxQ5 Control	$113CC_H$+ j*2000_H	PMj	PMj	Kernel Reset	2217
GETH0_Portj_MTL_RxQ5_Flow_Control (j=0-1)	Port j MTL RxQ5 Flow Control	$113D0_H$+ j*2000_H	PMj	PMj	Kernel Reset	2217
GETH0_Portj_MTL_Q5_Interrupt_Enable (j=0-1)	Port j MTL Q5 Interrupt Enable	$113F0_H$+j *2000_H	PMj	PMj	Kernel Reset	2218
GETH0_Portj_MTL_Q5_Interrupt_Status (j=0-1)	Port j MTL Q5 Interrupt Status	$113F4_H$+j *2000_H	PMj	PMj	Kernel Reset	2219
GETH0_Portj_MTL_TxQ6_Operation_Mode (j=0-1)	Port j MTL TxQ6 Operation Mode	11400_H+j *2000_H	PMj	PMj	Kernel Reset	2220
GETH0_Portj_MTL_TxQ6_Underflow (j=0-1)	Port j MTL TxQ6 Underflow	11404_H+j *2000_H	PMj	PMj	Kernel Reset	2222

(table continues...)

Table 781 (continued) Registers overview - GETH0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
GETH0_Portj_MTL_TxQ6_Debug (j=0-1)	Port j MTL TxQ6 Debug	11408_H+j*2000_H	PMj	PMj	Kernel Reset	2222
GETH0_Portj_MTL_TC6_ETS_Control (j=0-1)	Port j MTL TC6 ETS Control	11410_H+j*2000_H	PMj	PMj	Kernel Reset	2223
GETH0_Portj_MTL_TC6_Quantum_Weight (j=0-1)	Port j MTL TC6 Quantum Weight	11418_H+j*2000_H	PMj	PMj	Kernel Reset	2224
GETH0_Portj_MTL_TC6_SendSlopeCredit (j=0-1)	Port j MTL TC6 SendSlopeCredit	$1141C_H$+j*2000_H	PMj	PMj	Kernel Reset	2225
GETH0_Portj_MTL_TC6_HiCredit (j=0-1)	Port j MTL TC6 HiCredit	11420_H+j*2000_H	PMj	PMj	Kernel Reset	2226
GETH0_Portj_MTL_TC6_LoCredit (j=0-1)	Port j MTL TC6 LoCredit	11424_H+j*2000_H	PMj	PMj	Kernel Reset	2226
GETH0_Portj_MTL_RxQ6_Operation_Mode (j=0-1)	Port j MTL RxQ6 Operation Mode	11440_H+j*2000_H	PMj	PMj	Kernel Reset	2227
GETH0_Portj_MTL_RxQ6_Missed_Pkt_Overflow_Cnt (j=0-1)	Port j MTL RxQ6 Missed Pkt Overflow Cnt	11444_H+j*2000_H	PMj	PMj	Kernel Reset	2229
GETH0_Portj_MTL_RxQ6_Debug (j=0-1)	Port j MTL RxQ6 Debug	11448_H+j*2000_H	PMj	PMj	Kernel Reset	2230
GETH0_Portj_MTL_RxQ6_Control (j=0-1)	Port j MTL RxQ6 Control	$1144C_H$+j*2000_H	PMj	PMj	Kernel Reset	2231
GETH0_Portj_MTL_RxQ6_Flow_Control (j=0-1)	Port j MTL RxQ6 Flow Control	11450_H+j*2000_H	PMj	PMj	Kernel Reset	2231

(table continues...)

Table 781 (continued) Registers overview - GETH0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
GETH0_Portj_MTL_Q6_Interrupt_Enable (j=0-1)	Port j MTL Q6 Interrupt Enable	11470_H+j *2000_H	PMj	PMj	Kernel Reset	2232
GETH0_Portj_MTL_Q6_Interrupt_Status (j=0-1)	Port j MTL Q6 Interrupt Status	11474_H+j *2000_H	PMj	PMj	Kernel Reset	2233
GETH0_Portj_MTL_TxQ7_Operation_Mode (j=0-1)	Port j MTL TxQ7 Operation Mode	11480_H+j *2000_H	PMj	PMj	Kernel Reset	2234
GETH0_Portj_MTL_TxQ7_Underflow (j=0-1)	Port j MTL TxQ7 Underflow	11484_H+j *2000_H	PMj	PMj	Kernel Reset	2236
GETH0_Portj_MTL_TxQ7_Debug (j=0-1)	Port j MTL TxQ7 Debug	11488_H+j *2000_H	PMj	PMj	Kernel Reset	2236
GETH0_Portj_MTL_TC7_ETS_Control (j=0-1)	Port j MTL TC7 ETS Control	11490_H+j *2000_H	PMj	PMj	Kernel Reset	2237
GETH0_Portj_MTL_TC7_Quantum_Weight (j=0-1)	Port j MTL TC7 Quantum Weight	11498_H+j *2000_H	PMj	PMj	Kernel Reset	2238
GETH0_Portj_MTL_TC7_SendSlopeCredit (j=0-1)	Port j MTL TC7 SendSlopeCredit	$1149C_H$+ j*2000_H	PMj	PMj	Kernel Reset	2239
GETH0_Portj_MTL_TC7_HiCredit (j=0-1)	Port j MTL TC7 HiCredit	$114A0_H$+ j*2000_H	PMj	PMj	Kernel Reset	2240
GETH0_Portj_MTL_TC7_LoCredit (j=0-1)	Port j MTL TC7 LoCredit	$114A4_H$+ j*2000_H	PMj	PMj	Kernel Reset	2240
GETH0_Portj_MTL_RxQ7_Operation_Mode (j=0-1)	Port j MTL RxQ7 Operation Mode	$114C0_H$+ j*2000_H	PMj	PMj	Kernel Reset	2241

(table continues...)

Table 781 (continued) Registers overview - GETH0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
GETH0_Portj_MTL_RxQ7_Missed_Pkt_Overflow_Cnt (j=0-1)	Port j MTL RxQ7 Missed Pkt Overflow Cnt	$114C4_H$+j*2000_H	PMj	PMj	Kernel Reset	2243
GETH0_Portj_MTL_RxQ7_Debug (j=0-1)	Port j MTL RxQ7 Debug	$114C8_H$+j*2000_H	PMj	PMj	Kernel Reset	2244
GETH0_Portj_MTL_RxQ7_Control (j=0-1)	Port j MTL RxQ7 Control	$114CC_H$+j*2000_H	PMj	PMj	Kernel Reset	2245
GETH0_Portj_MTL_RxQ7_Flow_Control (j=0-1)	Port j MTL RxQ7 Flow Control	$114D0_H$+j*2000_H	PMj	PMj	Kernel Reset	2245
GETH0_Portj_MTL_Q7_Interrupt_Enable (j=0-1)	Port j MTL Q7 Interrupt Enable	$114F0_H$+j*2000_H	PMj	PMj	Kernel Reset	2246
GETH0_Portj_MTL_Q7_Interrupt_Status (j=0-1)	Port j MTL Q7 Interrupt Status	$114F4_H$+j*2000_H	PMj	PMj	Kernel Reset	2247
GETH0_Forward_Control	Forward Control	$1E000_H$	PG	PG	Kernel Reset	2248
GETH0_Port_Flush_and_Loopback_Control	Port Flush and Loopback Control	$1E004_H$	PG	PG	Kernel Reset	2249
GETH0_Port_0_Control	Port 0 Control	$1E00C_H$	PG	PG	Kernel Reset	2250
GETH0_TXQ_Map_Port_0	TXQ Map Port 0	$1E010_H$	PG	PG	Kernel Reset	2251
GETH0_RXC_Map_Port_0	RXC Map Port 0	$1E014_H$	PG	PG	Kernel Reset	2255
GETH0_Port_1_Control	Port 1 Control	$1E018_H$	PG	PG	Kernel Reset	2258
GETH0_TXQ_Map_Port_1	TXQ Map Port 1	$1E01C_H$	PG	PG	Kernel Reset	2259
GETH0_RXC_Map_Port_1	RXC Map Port 1	$1E020_H$	PG	PG	Kernel Reset	2263
GETH0_CSR_SW_Control	CSR SW Control	$1E100_H$	PG	PG	Kernel Reset	2266

(table continues...)

Table 781 (continued) Registers overview - GETH0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
GETH0_Interrupt_Status	Interrupt Status	$1E104_H$	PG	PG	Kernel Reset	2267
GETH0_Safety_Interrupt_Status	Safety Interrupt Status	$1E108_H$	PG	PG	Kernel Reset	2268
GETH0_DMA_Port_Selection	DMA Port Selection	$1E110_H$	PG	PG	Kernel Reset	2271
GETH0_PMT_Control	PMT Control	$1E114_H$	PG	PG	Kernel Reset	2271
GETH0_DPP_FSM_Interrupt_Status	DPP FSM Interrupt Status	$1E150_H$	PG	PG	Kernel Reset	2272
GETH0_AXI_Slave_DPP_Error_Address_Status	AXI Slave DPP Error Address Status	$1E154_H$	PG	PG	Kernel Reset	2276
GETH0_FSM_Control	FSM Control	$1E158_H$	PG	PG	Kernel Reset	2276
GETH0_FSM_ACT_Timer	FSM ACT Timer	$1E15C_H$	PG	PG	Kernel Reset	2278
GETH0_SCSR_Control	SCSR Control	$1E164_H$	PG	PG	Kernel Reset	2279
GETH0_SCSR_Parity_Status	SCSR Parity Status	$1E168_H$	PG	PG	Kernel Reset	2280
GETH0_SCSR_Parity_Err_Count	SCSR Parity Err Count	$1E16C_H$	PG	PG	Kernel Reset	2281
GETH0_Debug_Control	Debug Control	$1E208_H$	PG	PG	Kernel Reset	2281
GETH0_ECC_Control	ECC Control	$1E2C0_H$	PG	PG	Kernel Reset	2282
GETH0_ECC_Error_Status_Capture_Control	ECC Err Sts RCtl	$1E2D0_H$	PG	PG	Kernel Reset	2283
GETH0_ECC_Error_Address_Status	ECC Err Addr Status	$1E2D4_H$	PG	PG	Kernel Reset	2284
GETH0_ECC_Error_Count_Status	ECC Err Cntr Status	$1E2D8_H$	PG	PG	Kernel Reset	2285
GETH0_DPP_Control	DPP Control	$1E2E0_H$	PG	PG	Kernel Reset	2286
GETH0_DPP_ECC_Error_Injection_Control	DPP ECC EIC	$1E2E4_H$	PG	PG	Kernel Reset	2288
GETH0_DMA_Mode	DMA Mode	$1F000_H$	PG	PG	Kernel Reset	2289

(table continues...)

Table 781 (continued) Registers overview - GETH0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
GETH0_DMA_SysBus_Mode	DMA SysBus Mode	$1F004_H$	PG	PG	Kernel Reset	2290
GETH0_DMA_Interrupt_Status	DMA Interrupt Status	$1F008_H$	PG	PG	Kernel Reset	2292
GETH0_AXI_Tx_AR_ACE_Control	AXI Tx AR ACE Control	$1F010_H$	PG	PG	Kernel Reset	2294
GETH0_AXI_Rx_AW_ACE_Control	AXI Rx AW ACE Control	$1F018_H$	PG	PG	Kernel Reset	2295
GETH0_AXI_TxRx_AWAR_ACE_Control	AXI TxRx AWAR ACE Control	$1F01C_H$	PG	PG	Kernel Reset	2296
GETH0_DMA_Debug_Status0	DMA Debug Status0	$1F020_H$	PG	PG	Kernel Reset	2297
GETH0_DMA_Debug_Status1	DMA Debug Status1	$1F024_H$	PG	PG	Kernel Reset	2298
GETH0_DMA_Debug_Status3	DMA Debug Status3	$1F02C_H$	PG	PG	Kernel Reset	2298
GETH0_DMA_Tx_EDMA_Control	DMA Tx EDMA Control	$1F040_H$	PG	PG	Kernel Reset	2299
GETH0_DMA_Rx_EDMA_Control	DMA Rx EDMA Control	$1F044_H$	PG	PG	Kernel Reset	2300
GETH0_AXI_LPI_Entry_Interval	AXI LPI Entry Interval	$1F050_H$	PG	PG	Kernel Reset	2302
GETH0_DMA_TBS_CTRL0	DMA TBS CTRL0	$1F054_H$	PG	PG	Kernel Reset	2302
GETH0_DMA_TBS_CTRL1	DMA TBS CTRL1	$1F058_H$	PG	PG	Kernel Reset	2303
GETH0_DMA_TBS_CTRL2	DMA TBS CTRL2	$1F05C_H$	PG	PG	Kernel Reset	2303
GETH0_DMA_TBS_CTRL3	DMA TBS CTRL3	$1F060_H$	PG	PG	Kernel Reset	2304
GETH0_DMA_Safety_Interrupt_Status	DMA Safety Interrupt Status	$1F064_H$	PG	PG	Kernel Reset	2305
GETH0_DMA_ECC_Interrupt_enable	DMA ECC Interrupt enable	$1F068_H$	PG	PG	Kernel Reset	2307
GETH0_DMA_ECC_Interrupt_Status	DMA ECC Interrupt Status	$1F06C_H$	PG	PG	Kernel Reset	2308
GETH0_DMA_DPP_Control	DMA DPP Control	$1F070_H$	PG	PG	Kernel Reset	2308

(table continues...)

Table 781 (continued) Registers overview - GETH0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
GETH0_DMA_DPP_Interrupt_Status	DMA DPP Interrupt Status	$1F074_H$	PG	PG	Kernel Reset	2309
GETH0_DMA_CH_Ind_Ctrl	DMA CH Ind Ctrl	$1F080_H$	PG	PG	Kernel Reset	2310
GETH0_DMA_CH_Ind_Data	DMA CH Ind Data	$1F084_H$	PG	PG	Kernel Reset	2311
GETH0_DMA_CH0_TxExtCfg	DMA Channel j TxExtCfg	$1F084_H$	PG	PG	Kernel Reset	2312
GETH0_DMA_CH1_TxExtCfg	DMA Channel j TxExtCfg	$1F084_H$	PG	PG	Kernel Reset	2312
GETH0_DMA_CH2_TxExtCfg	DMA Channel j TxExtCfg	$1F084_H$	PG	PG	Kernel Reset	2313
GETH0_DMA_CH3_TxExtCfg	DMA Channel j TxExtCfg	$1F084_H$	PG	PG	Kernel Reset	2313
GETH0_DMA_CH4_TxExtCfg	DMA Channel j TxExtCfg	$1F084_H$	PG	PG	Kernel Reset	2314
GETH0_DMA_CH5_TxExtCfg	DMA Channel j TxExtCfg	$1F084_H$	PG	PG	Kernel Reset	2314
GETH0_DMA_CH6_TxExtCfg	DMA Channel j TxExtCfg	$1F084_H$	PG	PG	Kernel Reset	2315
GETH0_DMA_CH7_TxExtCfg	DMA Channel j TxExtCfg	$1F084_H$	PG	PG	Kernel Reset	2315
GETH0_DMA_Sfty_Ind_Ctrl	DMA Sfty Ind Ctrl	$1F088_H$	PG	PG	Kernel Reset	2316
GETH0_DMA_Sfty_Ind_Data	DMA Sfty Ind Data	$1F08C_H$	PG	PG	Kernel Reset	2318
GETH0_DMA_AM_Sfty_Ctrl	DMA AM Sfty Ctrl	$1F08C_H$	PG	PG	Kernel Reset	2318
GETH0_DMA_AM_Sfty_DPP_Intr_Sts	DMA AM Sfty DPP Intr Sts	$1F08C_H$	PG	PG	Kernel Reset	2319
GETH0_DMA_AM_Sfty_ECC_Intr_Sts	DMA AM Sfty ECC Intr Sts	$1F08C_H$	PG	PG	Kernel Reset	2320
GETH0_DMA_AM_Sfty_Intr_En	DMA AM Sfty Intr En	$1F08C_H$	PG	PG	Kernel Reset	2320
GETH0_DMA_AM_Sfty_RDC_Dbg_Ctrl	DMA AM Sfty RDC Dbg Ctrl	$1F08C_H$	PG	PG	Kernel Reset	2321
GETH0_DMA_AM_Sfty_RDC_Dbg_Data	DMA AM Sfty RDC Dbg Data	$1F08C_H$	PG	PG	Kernel Reset	2322

(table continues...)

Table 781 (continued) Registers overview - GETH0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
GETH0_DMA_AM_Sfty_RDC_ECC_Err_Cntr_Sts	DMA AM Sfty RDC ECC Err Cntr Sts	$1F08C_H$	PG	PG	Kernel Reset	2322
GETH0_DMA_AM_Sfty_WRC_Dbg_Ctrl	DMA AM Sfty WRC Dbg Ctrl	$1F08C_H$	PG	PG	Kernel Reset	2323
GETH0_DMA_AS_Sfty_Ctrl	DMA AS Sfty Ctrl	$1F08C_H$	PG	PG	Kernel Reset	2323
GETH0_DMA_AS_Sfty_DPP_Intr_Sts	DMA AS Sfty DPP Intr Sts	$1F08C_H$	PG	PG	Kernel Reset	2324
GETH0_DMA_AS_Sfty_ECC_Intr_Sts	DMA AS Sfty ECC Intr Sts	$1F08C_H$	PG	PG	Kernel Reset	2325
GETH0_DMA_AS_Sfty_Intr_En	DMA AS Sfty Intr En	$1F08C_H$	PG	PG	Kernel Reset	2325
GETH0_DMA_AS_Sfty_RAC_Dbg_Addr	DMA AS Sfty RAC Dbg Addr	$1F08C_H$	PG	PG	Kernel Reset	2326
GETH0_DMA_AS_Sfty_RAC_Dbg_Ctrl	DMA AS Sfty RAC Dbg Ctrl	$1F08C_H$	PG	PG	Kernel Reset	2326
GETH0_DMA_AS_Sfty_WAC_Dbg_Addr	DMA AS Sfty WAC Dbg Addr	$1F08C_H$	PG	PG	Kernel Reset	2327
GETH0_DMA_AS_Sfty_WAC_Dbg_Ctrl	DMA AS Sfty WAC Dbg Ctrl	$1F08C_H$	PG	PG	Kernel Reset	2328
GETH0_DMA_AS_Sfty_WDC_Dbg_Ctrl	DMA AS Sfty WDC Dbg Ctrl	$1F08C_H$	PG	PG	Kernel Reset	2328
GETH0_DMA_AS_Sfty_WDC_Dbg_Data	DMA AS Sfty WDC Dbg Data	$1F08C_H$	PG	PG	Kernel Reset	2329
GETH0_DMA_AS_Sfty_WDC_ECC_Err_Cntr_Sts	DMA AS Sfty WDC ECC Err Cntr Sts	$1F08C_H$	PG	PG	Kernel Reset	2329
GETH0_DMA_Sfty_Dbg_EIC	DMA Sfty Dbg EIC	$1F08C_H$	PG	PG	Kernel Reset	2330
GETH0_DMA_CHj_Control (j=0-7)	DMA Channel j Control	$1F100_H$+j *80_H	PCHj	PCHj	Kernel Reset	2331

(table continues...)

Table 781 (continued) Registers overview - GETH0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
GETH0_DMA_CHj_Tx_Control (j=0-7)	DMA Channel j Tx Control	$1F104_H$+j*80_H	PCHj	PCHj	Kernel Reset	2332
GETH0_DMA_CHj_Rx_Control (j=0-7)	DMA Channel j Rx Control	$1F108_H$+j*80_H	PCHj	PCHj	Kernel Reset	2334
GETH0_DMA_CHj_Slot_Function_Control_Status (j=0-7)	DMA Channel j Slot Function Control Status	$1F10C_H$+j*80_H	PCHj	PCHj	Kernel Reset	2337
GETH0_DMA_CHj_TxDesc_List_LAddress (j=0-7)	DMA Channel j TxDesc List LAddress	$1F114_H$+j*80_H	PCHj	PCHj	Kernel Reset	2338
GETH0_DMA_CHj_RxDesc_List_LAddress (j=0-7)	DMA Channel j RxDesc List LAddress	$1F11C_H$+j*80_H	PCHj	PCHj	Kernel Reset	2338
GETH0_DMA_CHj_TxDesc_Tail_LPointer (j=0-7)	DMA Channel j TxDesc Tail LPointer	$1F124_H$+j*80_H	PCHj	PCHj	Kernel Reset	2339
GETH0_DMA_CHj_RxDesc_Tail_LPointer (j=0-7)	DMA Channel j RxDesc Tail LPointer	$1F12C_H$+j*80_H	PCHj	PCHj	Kernel Reset	2339
GETH0_DMA_CHj_Tx_Control2 (j=0-7)	DMA Channel j Tx Control2	$1F130_H$+j*80_H	PCHj	PCHj	Kernel Reset	2340
GETH0_DMA_CHj_Rx_Control2 (j=0-7)	DMA Channel j Rx Control2	$1F134_H$+j*80_H	PCHj	PCHj	Kernel Reset	2341
GETH0_DMA_CHj_Interrupt_Enable (j=0-7)	DMA Channel j Interrupt Enable	$1F138_H$+j*80_H	PCHj	PCHj	Kernel Reset	2342
GETH0_DMA_CHj_Rx_Interrupt_Watchdog_Timer (j=0-7)	DMA Channel j Rx Interrupt Watchdog Timer	$1F13C_H$+j*80_H	PCHj	PCHj	Kernel Reset	2343

(table continues...)

Table 781 (continued) Registers overview - GETH0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
GETH0_DMA_CHj_Current_App_TxDesc_L (j=0-7)	DMA Channel j Current App TxDesc L	$1F144_H$+j*80_H	PCHj	PCHj	Kernel Reset	2345
GETH0_DMA_CHj_Current_App_RxDesc_L (j=0-7)	DMA Channel j Current App RxDesc L	$1F14C_H$+j*80_H	PCHj	PCHj	Kernel Reset	2345
GETH0_DMA_CHj_Current_App_TxBuffer_L (j=0-7)	DMA Channel j Current App TxBuffer L	$1F154_H$+j*80_H	PCHj	PCHj	Kernel Reset	2346
GETH0_DMA_CHj_Current_App_RxBuffer_L (j=0-7)	DMA Channel j Current App RxBuffer L	$1F15C_H$+j*80_H	PCHj	PCHj	Kernel Reset	2346
GETH0_DMA_CHj_Status (j=0-7)	DMA Channel j Status	$1F160_H$+j*80_H	PCHj	PCHj	Kernel Reset	2347
GETH0_DMA_CHj_Debug_Status (j=0-7)	DMA Channel j Debug Status	$1F164_H$+j*80_H	PCHj	PCHj	Kernel Reset	2350
GETH0_DMA_CHj_Desc_Mem_Cache_Fill_Level (j=0-7)	DMA Channel j Desc Mem Cache Fill Level	$1F168_H$+j*80_H	PCHj	PCHj	Kernel Reset	2352
GETH0_DMA_CHj_Miss_Packet_Cnt (j=0-7)	DMA Channel j Miss Packet Cnt	$1F16C_H$+j*80_H	PCHj	PCHj	Kernel Reset	2353
GETH0_DMA_CHj_Tx_Data_Xfer_Ring_Offset (j=0-7)	DMA Channel j Tx Data Xfer Ring Offset	$1F170_H$+j*80_H	PCHj	PCHj	Kernel Reset	2354
GETH0_DMA_CHj_Rx_Data_Xfer_Ring_Offset (j=0-7)	DMA Channel j Rx Data Xfer Ring Offset	$1F174_H$+j*80_H	PCHj	PCHj	Kernel Reset	2354
GETH0_DMA_CHj_Tx_Desc_Write_Ring_Offset (j=0-7)	DMA Channel j Tx Desc Write Ring Offset	$1F178_H$+j*80_H	PCHj	PCHj	Kernel Reset	2355

(table continues...)

Table 781 (continued) Registers overview - GETH0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
GETH0_DMA_CHj_Rx_Desc_Write_Ring_Offset (j=0-7)	DMA Channel j Rx Desc Write Ring Offset	$1F17C_H$+ $j*80_H$	PCHj	PCHj	Kernel Reset	2355

14.10.4.5 Device specific registers

There are no device specific register changes.

14.10.5 TC4Dx GETH connectivity

The table below lists interfaces to and from this functional block to other blocks in the device.

Table 782 List of GETH interface signals

Interface signals	I/O	Description
SRI_GETH_SIF	In	[SRI] slave interface, that provides software access to Configuration and Monitoring function.
GETH_SRI_MIF	Out	[SRI] Master Interface
CLOCK_GETH_fSRI	In	Clock for [GETH] configuration
CLOCK_GETH_fGETH	In	Clock (aclk_i) for [GETH] Receive and Transmit function (MTL + DMA) and PTP (clk_ptp_ref_i)
SMM_GETH_ApplicationReset	In	Application Reset
SMM_GETH_DebugReset	In	Debug Reset
GETH0_IR_SRC_GETHDMA[15:0]	Out	DMA TX and Rx Interrupt event to [IR]
GETH0_IR_SRC_GETHPPS[3:0]	Out	MAC[1:0]_PPS[1:0] Interrupt event to [IR]
GETH0_IR_SRC_GETHINTR	Out	Combined MAC0 or MAC1 Side Band Interrupt event to [IR]
GETH0_EGTM_SRC_GETHPPS[3:0]	Out	MAC[1:0]_PPS[1:0] Trigger event to [eGTM]
GETH0_PORTS_PPS[3:0]	Out	MAC[1:0]_PPS[1:0] output to [PORTS]
GETH0_SMU_GETHCE_ALARM	Out	Safety alarm signals to [SMU]
GETH0_SMU_GETHNCE_ALARM	Out	Safety alarm signals to [SMU]
GETH0_SMU_GETHSE_ALARM	Out	Safety alarm signals to [SMU]
GETH0_HSPHY_P0_TXEN	Out	MAC0 Transmit Enable of MII and RMII
GETH0_HSPHY_P0_GMIITXD[7:0]	Out	MAC0 Transmit Data, GMIITXD[3:0] for MII interface and GMIITXD[7:0] for GMII interface
GETH0_HSPHY_P0_TXER	Out	MAC0 Transmit Error of MII
HSPHY_GETH0_P0_TXCLK	In	MAC0 MII transmit clock Input (clk_tx_i) from PHY (10/100 Mbps). The external PHY or oscillator provides this transmission clock. TXD[3:0] is synchronous to this clock. This is 25 MHz in 100 Mbps mode and 2.5 MHz in 10 Mbps mode.
HSPHY_GETH0_P0_COL	In	MAC0 Collision MII
HSPHY_GETH0_P0_CRS	In	MAC0 Carrier Sense MII
HSPHY_GETH0_P0_RXDV	In	MAC0 Receive Data Valid MII
HSPHY_GETH0_P0_GMIIRXD[7:0]	In	MAC0 Receive Data, GMIIRXD[3:0] for MII interface and GMIIRXD[7:0] for GMII interface

(table continues...)

Table 782 (continued) List of GETH interface signals

Interface signals	I/O	Description
HSPHY_GETH0_P0_RXER	In	MAC0 Receive Error MII
HSPHY_GETH0_P0_RXCLK	In	MAC0 Receive clock (clk_rx_i) from PHY. The external PHY provides this receive clock for MII interfaces. This clock is 25 MHz in 100 Mbps mode, 2.5 MHz in 10 Mbps mode. Needs a clock input during SW reset (GETH0_DMA_MODE.SWR).
GETH0_HSPHY_P0_TXD[3:0]	Out	MAC0 Transmit Data, TXD[3:0] for RGMII. TXD[1:0] for RMII
HSPHY_GETH0_P0_RXD[3:0]	In	MAC0 Receive Data, RXD[3:0] for RGMII. RXD[1:0] for RMII
HSPHY_GETH0_P0_CRSDV	In	MAC0 Carrier Sense / Data Valid combi-signal for RMII
HSPHY_GETH0_P0_REFCLK	In	MAC0 50-MHz clock (clk_rmii_i) used by the RMII from PHY. In 100 Mbps Mode, it is divided internally to 25 MHz and provided to the internal MII interface. If RMII is downgraded to 10 Mbps it is divided internally to 2.5 MHz. This division is controlled automatically by evaluation of mac_speed_o[1:0] and phy_intf_sel_i[2:0]. Needs a clock input during SW reset (GETH0_DMA_MODE.SWR).
GETH0_HSPHY_P0_TCTL	Out	MAC0 Transmit Control for RGMII
HSPHY_GETH0_P0_GETH_TXCLK	In	MAC0 RGMII transmission clock input (clk_tx_i) from HSPHY. TXD[3:0] is synchronous (DDR) to this 125 MHz clock to the PHY. If RGMII is downgraded to 100 Mbps it is divided internally to 25 MHz and for 10 Mbps to 2.5 MHz. In each case it is generated from clk_gref_i by respective division. TXCLK has the same on chip delay and output pad delay as the TXD signals and TCTL unless clock skew is selected in SKEW.
HSPHY_GETH0_P0_GETH_RXCLK	In	MAC0 Receive clock (clk_rx_i) from PHY. The external PHY provides this receive clock for RGMII This clock is 125 MHz in 1 Gbps mode, 25 MHz in 100 Mbps mode. Needs a clock input during SW reset (GETH0_DMA_MODE.SWR).
HSPHY_GETH0_P0_RCTL	In	MAC0 Receive Control for RGMII
HSPHY_GETH0_P0_XGMIIRX[31:0]	In	MAC0 RXD interface of XGMII
GETH0_HSPHY_P0_XGMIITX[31:0]	Out	MAC0 TXD interface of XGMII
GETH0_HSPHY_P0_MDC	Out	MAC0 MDIO clock
GETH0_HSPHY_P0_MDIO	Out	MAC0 MDIO data output for bidirectional pin
HSPHY_GETH0_P0_MDIO	In	MAC0 MDIO data input for bidirectional pin
GETH0_HSPHY_P1_TXEN	Out	MAC1 Transmit Enable of MII and RMII
GETH0_HSPHY_P1_GMIITXD[7:0]	Out	MAC1 Transmit Data, GMIITXD[3:0] for MII interface and GMIITXD[7:0] for GMII interface
GETH0_HSPHY_P1_TXER	Out	MAC1 Transmit Error of MII

(table continues...)

Table 782 (continued) List of GETH interface signals

Interface signals	I/O	Description
HSPHY_GETH0_P1_TXCLK	In	MAC1 MII transmit clock Input (clk_tx_i) from PHY (10/100 Mbps). The external PHY or oscillator provides this transmission clock. TXD[3:0] is synchronous to this clock. This is 25 MHz in 100 Mbps mode and 2.5 MHz in 10 Mbps mode.
HSPHY_GETH0_P1_COL	In	MAC1 Collision MII
HSPHY_GETH0_P1_CRS	In	MAC1 Carrier Sense MII
HSPHY_GETH0_P1_RXDV	In	MAC1 Receive Data Valid MII
HSPHY_GETH0_P1_GMIIRXD[7:0]	In	MAC1 Receive Data, GMIIRXD[3:0] for MII interface and GMIIRXD[7:0] for GMII interface
HSPHY_GETH0_P1_RXER	In	MAC1 Receive Error MII
HSPHY_GETH0_P1_RXCLK	In	MAC1 Receive clock (clk_rx_i) from PHY. The external PHY provides this receive clock for MII interfaces. This clock is 25 MHz in 100 Mbps mode, 2.5 MHz in 10 Mbps mode. Needs a clock input during SW reset (GETH0_DMA_MODE.SWR).
GETH0_HSPHY_P1_TXD[3:0]	Out	MAC1 Transmit Data, TXD[1:0] for RMII
HSPHY_GETH0_P1_RXD[3:0]	In	MAC1 Receive Data, RXD[1:0] for RMII
HSPHY_GETH0_P1_CRSDV	In	MAC1 Carrier Sense / Data Valid combi-signal for RMII
HSPHY_GETH0_P1_REFCLK	In	MAC1 50-MHz clock (clk_rmii_i) used by the RMII from PHY. In 100 Mbps Mode, it is divided internally to 25 MHz and provided to the internal MII interface. If RMII is downgraded to 10 Mbps it is divided internally to 2.5 MHz. This division is controlled automatically by evaluation of mac_speed_o[1:0] and phy_intf_sel_i[2:0]. Needs a clock input during SW reset (GETH0_DMA_MODE.SWR).
HSPHY_GETH0_P1_XGMIIRX[31:0]	In	MAC1 RXD interface of XGMII
GETH0_HSPHY_P1_XGMIITX[31:0]	Out	MAC1 TXD interface of XGMII
GETH0_HSPHY_P1_MDC	Out	MAC1 MDIO clock
GETH0_HSPHY_P1_MDIO	Out	MAC1 MDIO data output for bidirectional pin
HSPHY_GETH0_P1_MDIO	In	MAC1 MDIO data input for bidirectional pin

14.10.6 TC4Dx GETH revision history

Initial release of the chapter.

15 LETH - NDA only

Information for this functional block is only available under NDA.

16 HSPHY - NDA only

Information for this functional block is only available under NDA.

17 High Speed Serial Link (HSSL)

The HSSL module provides point-to-point communication of both single data values and of large data blocks, called streams. The communicating partner can be complex microcontrollers, or a microcontroller and a device with only basic execution capabilities.

The HSSL module supports single block 8, 16, or 32-bit data write or read transfers either to or from the target device's register. Using a streaming channel it also supports transfers of long 256-bit data blocks.

The HSSL module consists of two main sub-modules:

- High Speed Serial Link (HSSL)
- High Speed Communication Tunnel (HSCT)

The HSSL sub-module implements Transport Layer tasks and hands over the data to the HSCT sub-module which provides Data Link Layer and Physical Layer services, data serialization, and transmission. All transfers are protected by safety features such as Cyclic Redundancy Check(CRC) and timeout.

17.1 Feature list

- Flexible functional configuration by application software
  - Frame formatting configuration
  - Various operation modes
  - TX-link baudrate support (10 MHz/20 MHz SysClk(System Clock)): 5 MBaud (low speed) and 320 MBaud (high speed)
  - TX-link baudrate support (25 MHz SysClk): 6.25 MBaud (low speed) and 320 MBaud (high speed)
  - RX-link baudrate support (20 MHz SysClk): 5 MBaud (low speed), 20 MBaud (medium speed) and 320 MBaud (high speed)
  - RX-link baudrate support (10 MHz SysClk): 5 MBaud (low speed) and 320 MBaud (high speed)
  - RX-link baudrate support (25 MHz SysClk): 6.25 MBaud (low speed) and 320 MBaud (high speed
- Transmit HSSL frames to external devices and handle all protocol relevant topics
  - Writing a single 8, 16, or 32-bit data value into the register of a target device
  - Supports transfers of a long 256-bit streaming data block
  - Automatic frame transfer ID (IDentity) generation for detection of dropped frames
  - Acknowledge for command and stream frames
- Receive HSSL frames from external devices and handle all protocol relevant topics
  - Reading single data from an 8, 16, or 32-bit register of a target device
  - Supports reception of a long 256-bit streaming data block
- Transfers protected by CRC16 (16 bit Cyclic Redundancy Check)
- Support of 32-bit address range
- Read and write HSSL frame data either to or from system address space
  - Support of Direct Memory Access(DMA) driven multiple register write or read transfers
  - Two stage FIFOs (First in, First Outs) for transmitting and receiving streaming data
- Flexible event signaling and interrupt structure
  - Programmable timeouts for detection of blocked answer transfers
  - Remote trigger of an event or an interrupt in the target device by the initiator
  - Total 17 interrupt lines, four types of channel specific interrupt per channel and one module interrupt
- Automatic FIFO flush when entering the run mode, for error handling
- Read and write access protection by an external Memory Protection Unit (MPU)
- Identification of the target by the JTAG ID (Joint Test Action Group Identification) number

- Multi-slave connection supports up to three slaves
- The communication interface is based on IEEE 1596.3 LVDS IOs (Low Voltage differential Signaling Input Outputs)and provides driver swing amplitude configuration

17.2 Functional overview

The block diagram of the HSSL module is shown in the following figure:

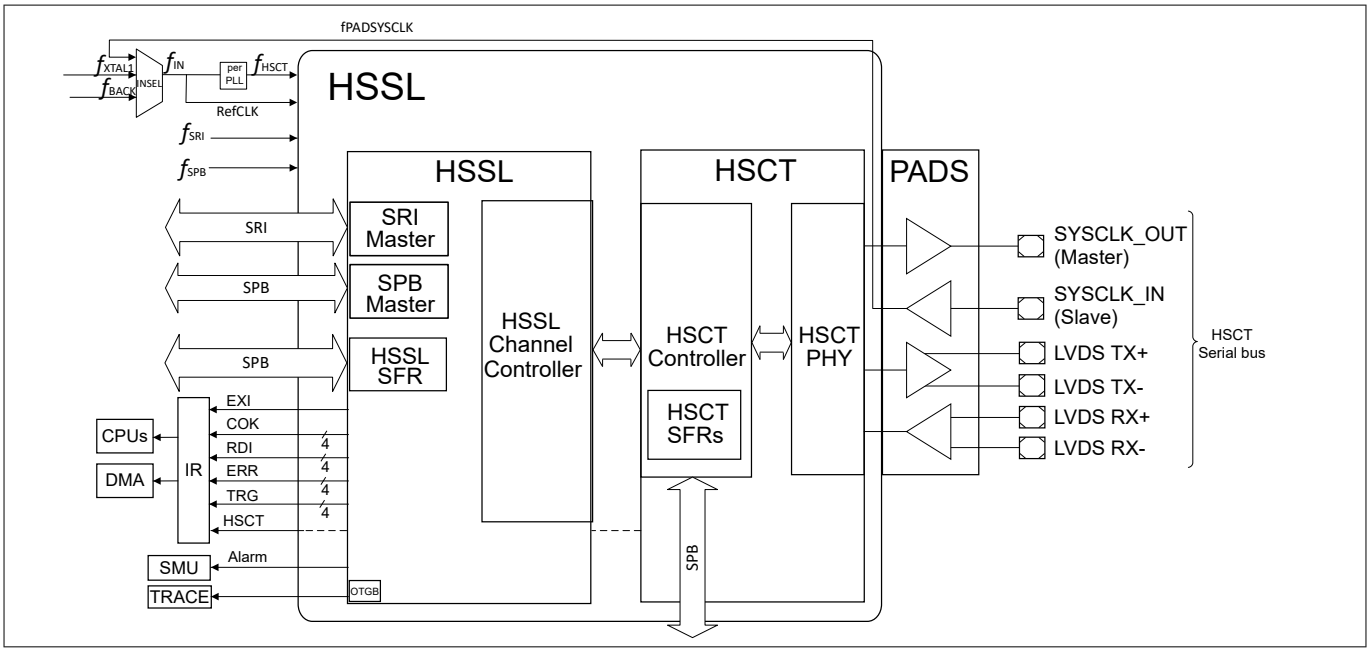


Figure 190 HSSL block diagram

***Note**: The bus interface used by the HSSL module is defined in the device specific chapter.*

The HSSL has one alarm, called 'HSSL SRI Read Data EDC Error' interfacing to the SMU (EDC stands for Error Detection and Correction). For the alarm description please refer to the alarm mapping tables in the SMU functional block and the safety manual. A set of 17 interrupts are mapped to IR(Interrupt Router), four types of channel specific interrupts per channel and one module interrupt. In order to support the debugging activities, the HSSL provides a set of internal signals to the on-chip debug system OCDS (On Chip Debug System) through OTGB (OnChip Trigger Generation Bus) trigger buses.

High Speed Serial Link (HSSL)

The HSSL sub-module provides point-to-point communication for both single data values and large data blocks, called streams. The functionality of each of these channels is further sub divided as 'transmitter and receiver' and 'Initiator and target' controllers by the HSSL channel controller. The communicating devices can be complex micro controllers, or a microcontroller and a device with only basic execution capabilities. There are four channels to transfer single values to/from target. They support direct writing of 8/16/32 bit data from the initiator into a target's local address map (e.g., registers and memory), as well as reading a value from a target, performed by a modules internal SPB/SRI bus master on the target side. For transferring large data blocks there is a channel containing FIFOs. The HSSL module implements transport layer tasks and hands over the data to another module which provides data link layer and physical layer services, data serialization and transmission. All transfers are protected by safety features like CRC and timeout. The SRI and SPB master interface on the target HSSL helps in accessing the peripherals on the corresponding bus. The SPB slave interface helps in accessing the special function registers of HSSL.

The SRI master interface is capable of performing both single and burst reads and writes on the SRI bus. The SPB slave interface is used by an SPB master (DMA, CPU, HSSL SPB Master) for writing the HSSL SFRs, so configuring the module and performing single read and write operations.

High Speed Communication Tunnel (HSCT)

The lower layer companion module and logic control the lower communication layers positioned between two devices, the data, and the physical layer. They consist of the High Speed Communication Tunnel (HSCT) and pads. For more information on the lower communication layers, see the HSCT chapter. The SPB slave interface helps in accessing the special function registers of HSCT.

The chip-to-chip interface employs a digital interface for inter chip communication between a master or a slave HSCT instance. The interface is capable of running in a master or in a slave HSCT mode. During configuration phase the role of the Interface (master or slave) has to be defined. It is not intended to change the system role after initialization and start-up. The HSCT controller consists of framer, deframer and the corresponding FSMs(finite state machines) along with the SFRs controlled via SPB slave interface.

The interface consists of a full-duplex RX and TX high-speed data interface based on double ended differential signals (in total 4 lines) and a master clock interface (SYSCLK_OUT). The master IC owns the crystal and provides the clock to the slave. The interface reset is derived from the System reset and provided by chip internal reset signaling.

Related information

TC4Dx SMU alarm mapping tables on page 7233

17.3 Functional description

The functional description is divided into the blocks sub-functions that follow.

17.4 High Speed Serial Link (HSSL)

This chapter explains the detailed functional behavior on transport layer tasks of the HSSL sub-module.

17.4.1 Feature list

- Connecting microcontroller to microcontroller or to any digital device
- Writing a single 8, 16, or 32-bit data value into the register of a target device
- Reading single data from an 8, 16, or 32-bit register of a target device
- Supporting 32-bit address range
- Transfers protected by CRC16 (16 bit Cyclic Redundancy Check)
- Programmable time outs for detection of blocked answer transfers
- Automatic frame transfer ID (IDentity) generation for detection of dropped frames
- Supporting DMA (Direct Memory Access) driven multiple register write and read transfers
- Efficient transmission and reception of large data blocks and streams
- Acknowledge for command and stream frames
- Two stage FIFOs(First in, First Outs) for transmitting and receiving streaming data
- Automatic FIFO flush when entering the run mode, for error handling
- Write protection by an external Memory Protection Unit (MPU)
- Remote trigger of event and interrupt in the target device by the initiator
- Identification of the target by the JTAG ID(Joint Test Action Group Identification)number
- Feature set identification of the HSSL module possible by using the JTAG ID number
- Access protection from an external master

17.4.2 Functional overview

The HSSL sub-module generates all necessary protocol formats in order to enable two devices to communicate to each other. Two basic modes are supported. In register transfer mode the transmitter (sending device) provides an address and a data value by writing them into dedicated registers of the HSSL sub-module. The HSSL sub-module puts it into an envelop (called a frame) and forwards it through the HSCT protocol and physical layer to the target device. The target takes the information out of the frame and writes the received data into the address which came in the same frame as the data. If the sender wants to read the content of a register or a memory location, it simply sends only the address. The target device reads the value from the specified location and replies it to the sender by transferring it through an answer frame.

The transfer of register or memory contents is protected by several security levels. A 16-bit CRC value (called CRC16) is attached to the end of each frame. Each transfer process triggers a watchdog, which makes sure that the communication happens within a deterministic schedule. The target device itself offers the capability to define access windows, which allow reading or writing only from or into certain address ranges.

Each frame must be acknowledged by the target by sending an appropriate response frame. If the response frame does not come inside a predefined time window, the initiator module triggers a timeout interrupt.

In data stream mode data blocks of user defined size can be moved from the memory of the sending device into the memory of the target device. A DMA master, which can be configured with any source address and source length (in multiple of 256-bit frame), autonomously fetches data, breaks it into smaller blocks of programmable size, and puts it into a FIFO. The lower level companion module of the HSSL module empties the FIFO from the 'other side' to the physical lines, which connect two devices. The target device's structure is the reverse of the sender system's, in which the HSSL module's lower-level companion module , puts received data blocks into a FIFO, which is in turn emptied by a DMA master into the target memory.

All streaming data transfers are subject to the protection features which are available for the register transfer mode.

The finite state machine is taking care of the protocol's request/response pair arbitration and administration of the four virtual HSSL channels.

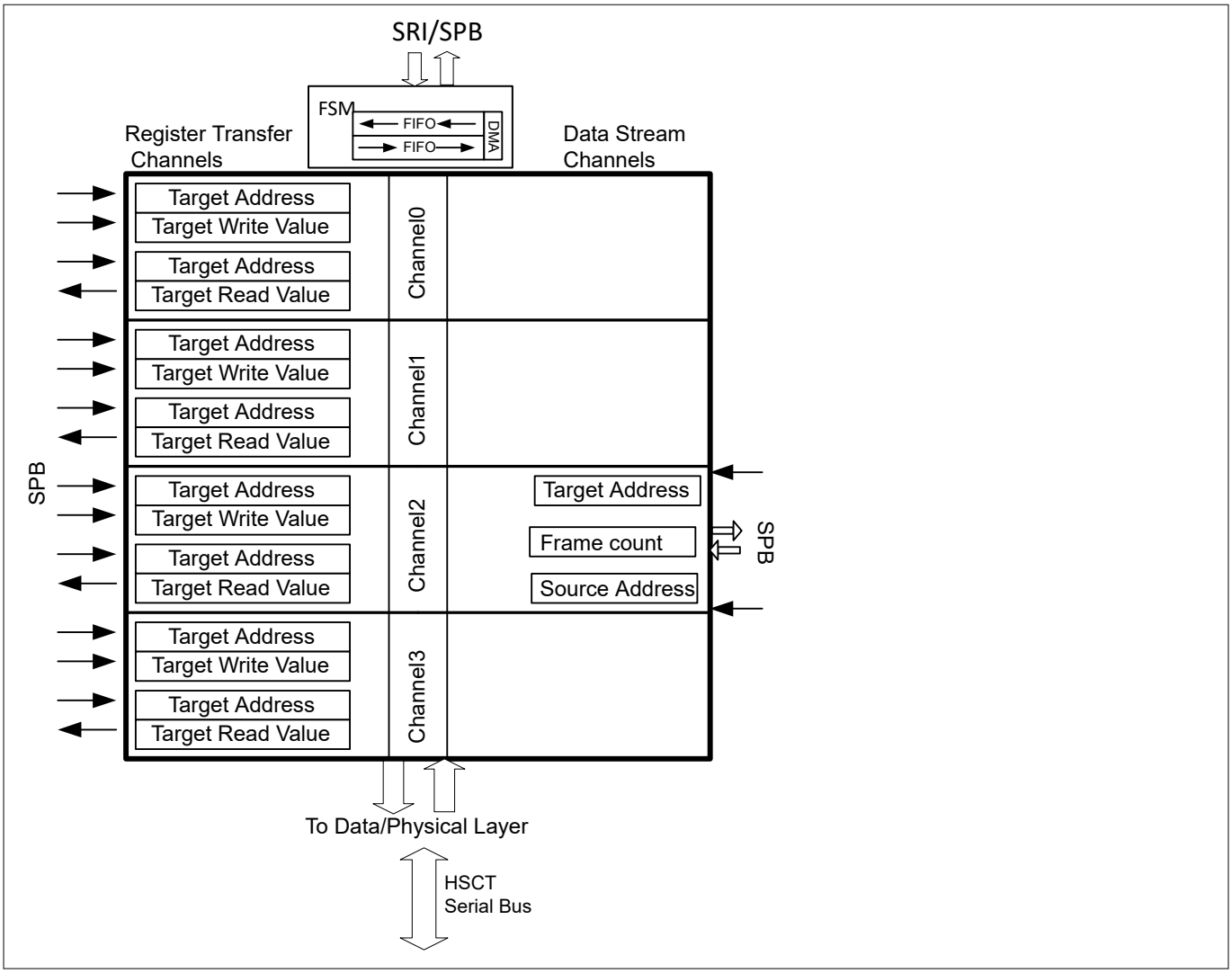


Figure 191 HSSL_IP block diagram

Related information

Security Control Register on page 2543
Multi-slave operation on page 2476

17.4.3 Functional description

This chapter describes in detail the HSSL sub-module functional operation.

17.4.3.1 HSSL protocol definition

The HSSL communication normally consists of two stage transactions; sending a command and receiving a response.

The communication involves two participants:

- An initiator
  - Starts a transaction by sending a command
- A target
  - Responds to the command

17.4.3.1.1 Terms and descriptions

Table 783 Common HSSL terms

Term	Description
ACK	Acknowledge frame
CRC	Cyclic redundancy check
HSSL	High speed serial link
Initiator	Device that starts a HSSL transaction by sending a command frame
lsb	Least significant bit
msb	Most significant bit
NACK	Not acknowledge frame (target error frame)
SoC	System on chip
SPB	Serial peripheral bus
Target	Device that responds to a command frame sent by an initiator
TO	Time out
TT	Transaction tag
TTERR	Transaction tag error

17.4.3.1.2 Frame types

The HSSL module generates a serial frame by:

- Taking the payload delivered by either a CPU or a DMA
- First wrapping it in a HSSL frame by adding header and CRC fields
- Wrapping the HSSL frame in a HSCT frame by adding Sync, header, and End bit-fields

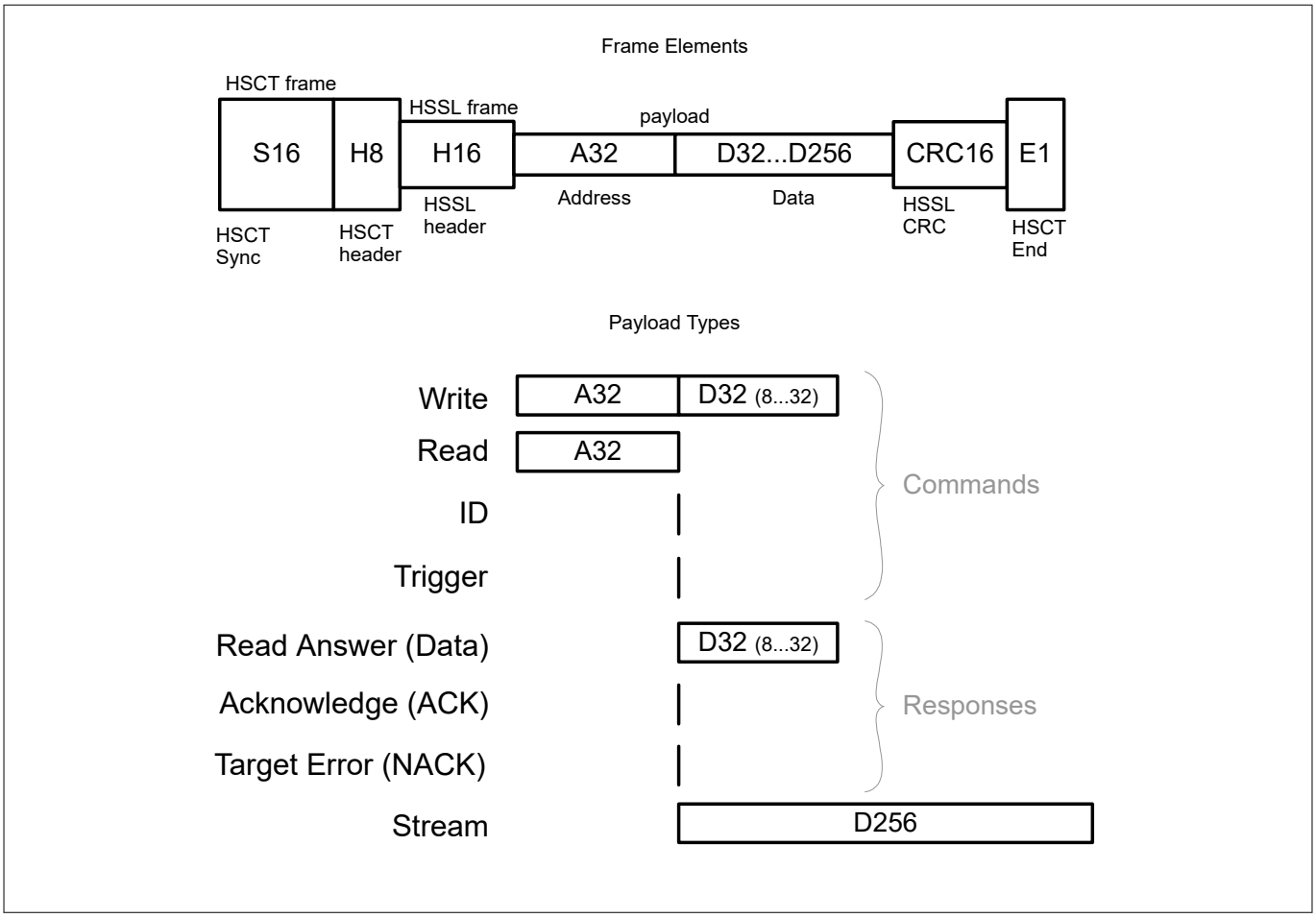


Figure 192 HSSL frame types

There are several types of frames:

- Command
- Response
- Stream

Command frames

Command frames are sent by the initiator to request an action by the target. There are four types of command frames:

- Write frame
  - Used by the initiator to request the target controller to write the data to the address, both provided by the initiator
- Read frame
  - Used by the initiator to request the target controller to read and deliver a content from an address space of target
- Read ID frame
  - Used by the initiator to request the target controller to read and deliver a content uniquely defining the target device. The answer is a standard read data answer frame
- Trigger frame
  - Used by the initiator to trigger the trigger interrupt in the target module

In this document, the command frames are sometimes referred to as WRT frames

Response frames

Response frames are sent by the target either to deliver the requested data or to report a successful or unsuccessful completion of a request. There are three types of response frames:

- Read Data Answer frame
  - Used by the target to deliver the data, requested by the initiator with a read command frame. It is frequently referred to as a Data frame
- Acknowledge frame
  - Used by the target to confirm the completion of a write or a trigger command. It is frequently referred to as an ACK frame
- Target Error frame
  - Used by the target to report the unsuccessful attempt to complete a write or a read command. It is frequently referred to as a NACK frame

Stream frame

Used by the initiator controller for sending data streams. It behaves similar to a write command frame which contains long 256-bit data block. It is acknowledged in the same way, but the acknowledge uses sliding window with depth of 2, instead of 1 as for the simple command frames.

Frame and payload lengths

Each type of frame has a single data length associated with it. The length of the frame depends on the length of the payload. Different frame types have different payload lengths, but the payload length for one type is constant for all variations of the frame type.

For example all data frames have the same length, although they can carry 8, 16, or 32-bit data. This is due to the fact that the data field in the frame is always 32 bit long, and if the data itself is shorter, it is copied several times to fill in the whole field. The same behavior is valid for the read answer frames.

The rest of the frame, consisting of sync and header fields, has always the same length.

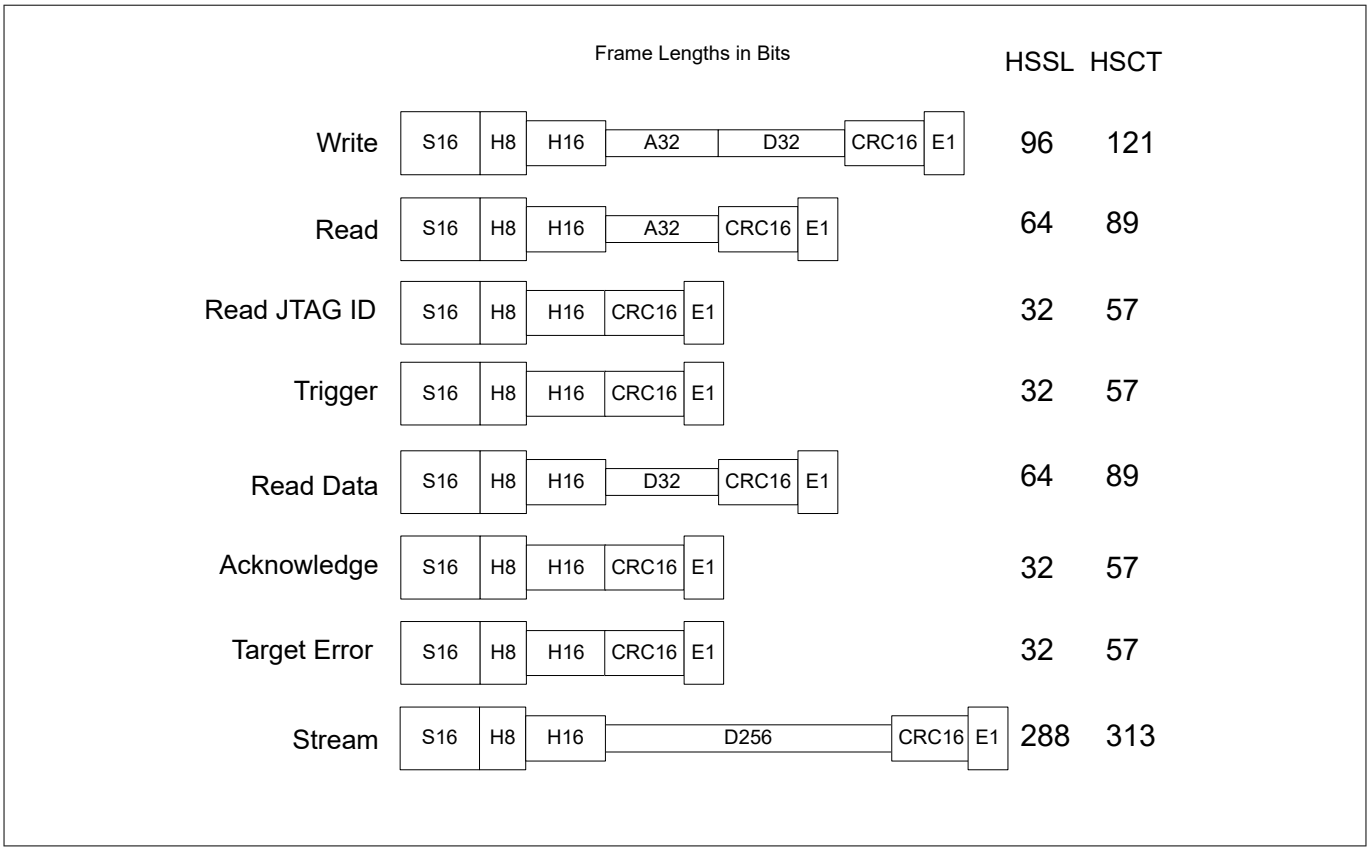


Figure 193 Frame lengths

Data types

There are four types of HSSL data:

- 8-bit
- 16-bit
- 32-bit
- 256-bit stream block

In the case of 8, 16, or 32-bit data types, the transmitted number of bits is always 32, where the shorter data types are copied several times to fill up this length. In case of 256 bit stream, each data length transmitted is of 8 bit.

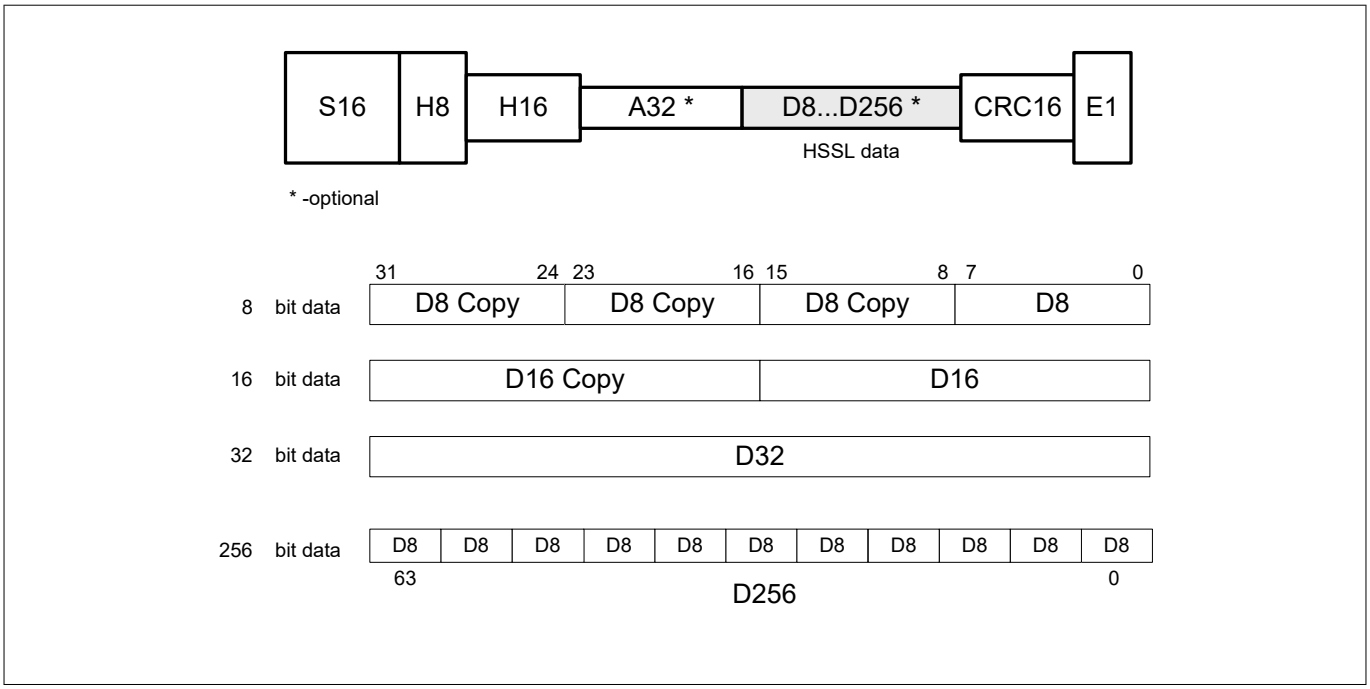


Figure 194 Data types

Cyclic redundancy check (CRC) field

The HSSL protocol uses the CRC-16-CCITT polynomial: $x^{16} + x^{12} + x^5 + 1$ with the following standard properties:

- Seed: $FFFF_H$
- Calculation direction: Msb first (header msb to lsb, then payload msb to lsb)
- CRC result direction: Msb first

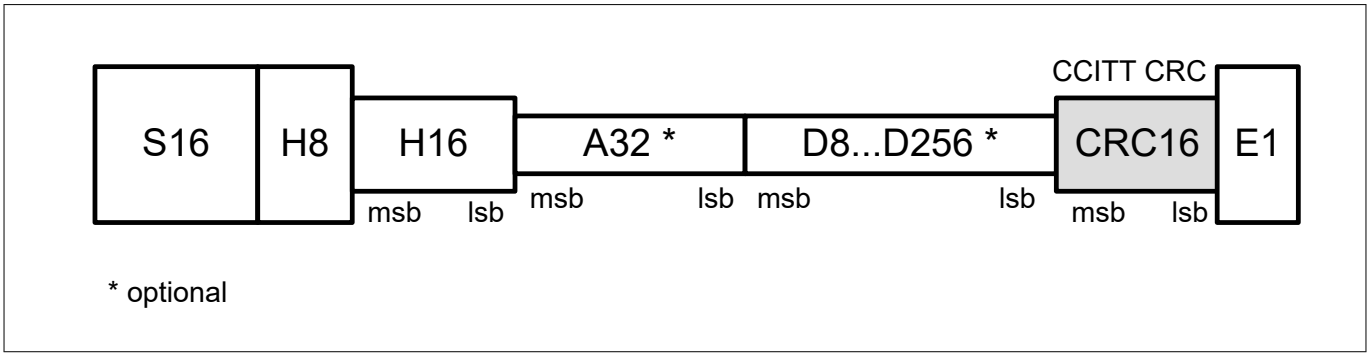


Figure 195 CRC field

Header structure

The HSSL header contains the protocol information that is required in addition to the raw payload data. The HSSL header describes the payload data in terms of the length and the type, and carries additional information such as the channel number code and transaction tag.

The header also carries stand-alone information in case of frames without payload, such as acknowledge and trigger frames.

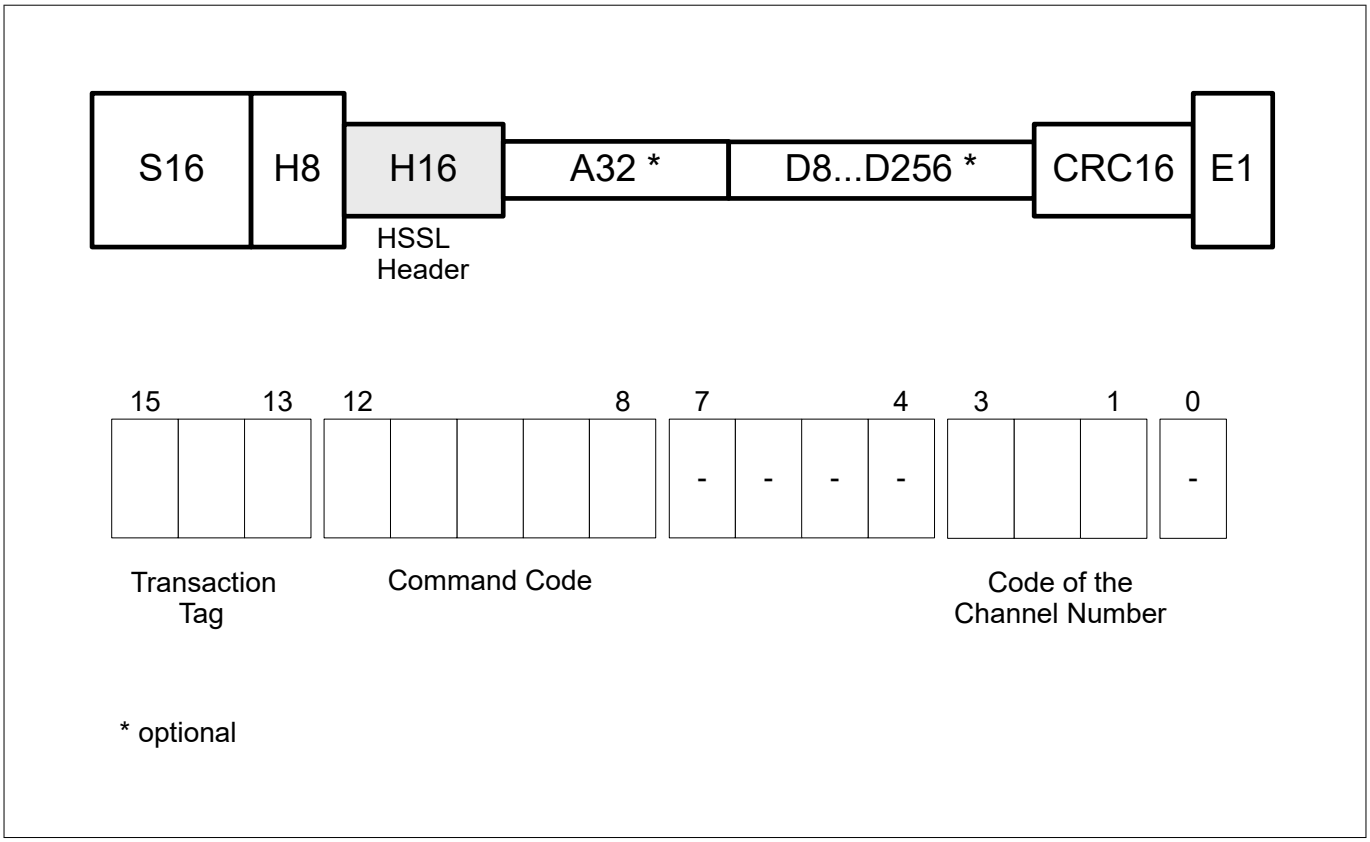


Figure 196 Header types

***Note**: The unused bits in the header are padded with 0.*

Table 784 Mapping of HSSL to HSCT channel codes

HSSL channel number	HSSL channel number, binary code	HSSL channel number, special code	HSCT channel	HSCT channel code
0	000	100	A	0100
1	001	101	B	0101
2	010	110	C	0110
3	011	111	D	0111
4 (reserved)	100	000	E	1000
5 (reserved)	101	001	F	1001
6 (reserved)	110	010	G	1010
7 (reserved)	111	011	H	1011

***Note**: The HSSL module supports four channels, 0 to 3.*

Table 785 List of command codes

Command code	Command description	HSSL frame size
00000	Read 8 bit	64

(table continues...)

Table 785 (continued) List of command codes

Command code	Command description	HSSL frame size
00001	Read 16 bit	64
00010	Read 32 bit	64
00011	RFU (Reserved for future use)	-
00100	Write 8 bit with ACK (Acknowledge)	96
00101	Write 16 bit with ACK	96
00110	Write 32 bit with ACK	96
00111	RFU	-
01000	ACK OK	32
01001	ACK target error	32
01010	Read answer OK	64
01011	RFU	-
01100	Trigger with ACK	32
01101	RFU	-
01110	RFU	-
01111	RFU	-
10000	RFU	-
10001	RFU	-
10010	Read JTAG ID 32-bit	32
10011	RFU	-
10100	RFU	-
10101	RFU	-
10110	RFU	-
10111	Stream data with ACK (32 Byte Data)	288
11000	RFU	-
11001	RFU	-
11010	RFU	-
11011	RFU	-
11100	RFU	-
11101	RFU	-
11110	RFU	-
11111	RFU	-

17.4.3.1.3 Single and block transfers

Single frame transfers are always performed expecting an acknowledgment.

17.4.3.1.4 Streaming interface

Stream frame transfers are always performed with an acknowledgment for each transfer.

There are only write streams possible. Read streams are not possible.

17.4.3.1.5 Sliding window protocol

For flow control, the HSSL module implements two variants of the 'Sliding Window' protocol depending on the sliding window depth:

- Depth of one used for command frames
  - The depth of one means that a single timer tracks the arrival of the corresponding response frame. A new command can be sent only after the response to the previous command has been received, or a timeout has occurred
- Depth of two used for streaming frames
  - The depth of two means that there are two timers tracking the acknowledgment times for the two latest streaming frames. In this case, a second stream frame can be sent although an acknowledgment or timeout to the previous stream frame has not been received yet. After the second stream frame, no third stream frame can be sent before the first one has been acknowledged

An example of communication using sliding window with depth of one protocol is shown in the next figure. Here, the initiator sends commands originating from one particular channel, for example channel 0 (labeled with the capital letter “A”), and the corresponding acknowledge frames from the target are labeled with the small letter “a”.

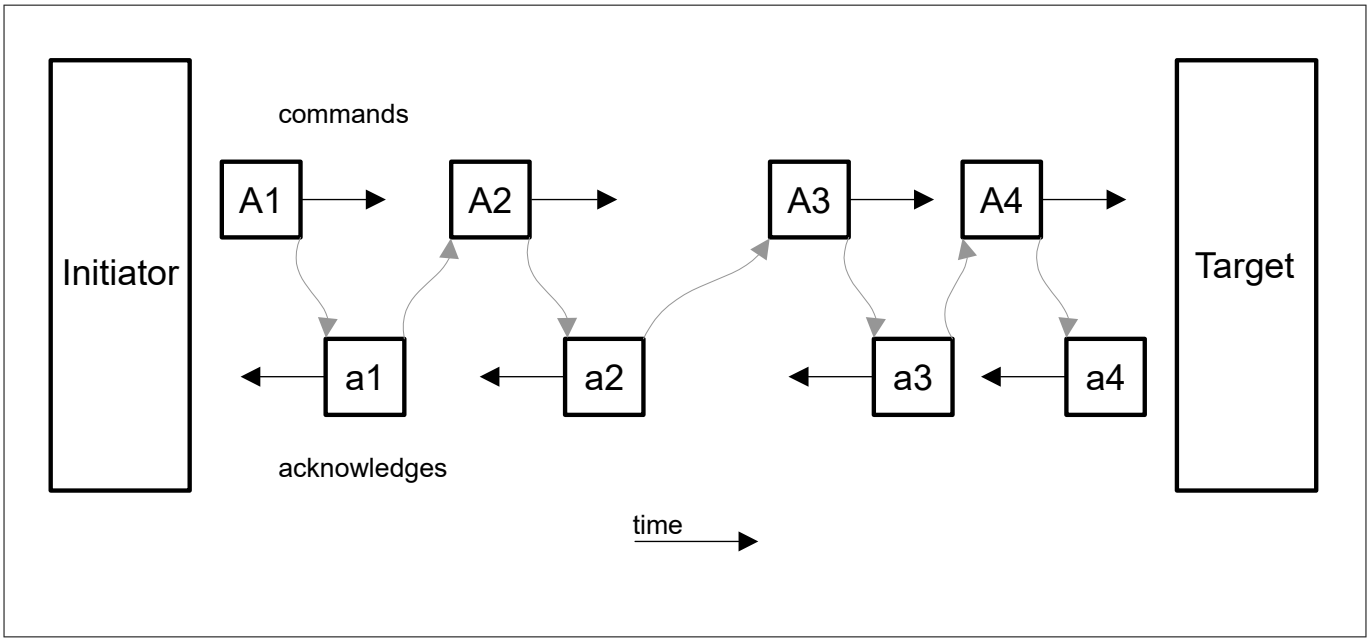


Figure 197 Flow control, single channel, sliding window with depth of one

The sliding window protocol requires the capability of sending acknowledge frames in the target, and timeout timers in the initiator.

Every time the target receives a frame without detecting an error, it sends either a response or an acknowledgment to the initiator.

If the target detects a frame with a CRC error, it does not send an acknowledgment. The missing acknowledgment causes a timeout event in the initiator channel.

If the command frames are longer than the acknowledge frames plus the reaction time, for example in the case of streaming, back-to-back transfers are easily possible, as shown in the figure below.

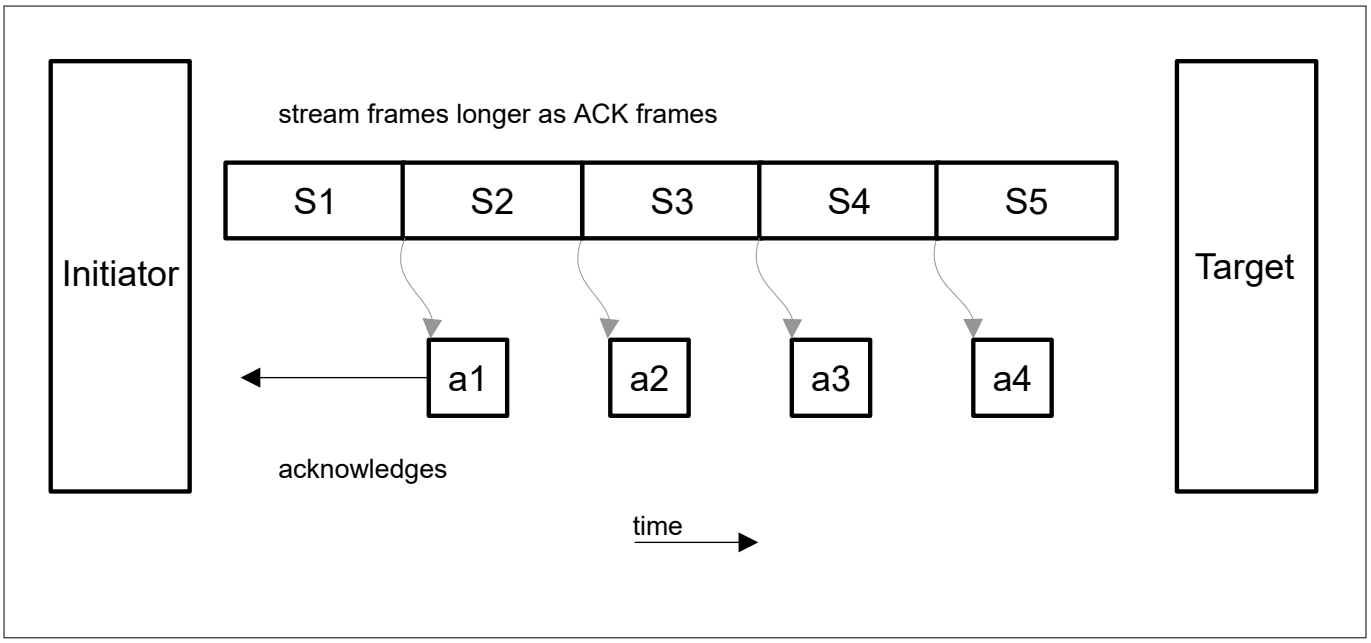


Figure 198 Flow control, streaming frames

17.4.3.1.6 Error management

The HSSL protocol defines four types of errors whose statuses can be checked in the MFLAGS register:

- Time out
- Transaction tag error
- CRC error
- Target error

Time out error

A time out error is detected at the initiator side, if the expected ACK frame was not received within the expected time window. This can occur either if a frame had been sent by the initiator, and the target detected an CRC error and did not answer with an acknowledgment, or the connection between the initiator and the target is physically damaged in one or the other direction.

Transaction tag error

A transaction tag error occurs at the initiator side, if instead the expected ACK frame with the expected TAG number, an acknowledgment with an unexpected transaction tag was received. This would indicate a missing frame or a missing acknowledgment. Transaction tag error generate frames that pass the CRC checking stage.

CRC error

A CRC error can occur:

- At the target side, in which case:
  - The CRC error flag is set
  - The received command frame with a CRC error is discarded

  - No acknowledge frame is sent
  - A channel non-specific EXI error interrupt is generated, if enabled
- At the initiator side, in which case:
  - The CRC error flag is set
  - The received response frame with a CRC error is discarded
  - A channel non-specific EXI error interrupt is generated, if enabled

Both scenarios lead to a time out at the initiator side.

In both cases the CRC error flag is set at the side receiving the erroneous frame, either initiator or target, and an interrupt is generated, if enabled.

For the purpose of CRC error injection, a bit field with an XOR mask used for toggling the bits of the calculated CRC is provided, see CRC.XORMASK. In order to switch on this feature, the XEN bit must be set by the application software. The error injection only influences the generation of the transmitted CRC.

Target error

A target error can occur while accessing the target memory. It can be a bus error or a memory protection error. In such a case, the target responds with an NACK frame indicating the error.

17.4.3.1.7 Shift direction

In general, the HSSL interface makes a copy of a memory location or block from the address space of one microcontroller into the address space of the other microcontroller.

The HSSL module delivers 32-bit parallel data.

The transfer over HSCT serial link is done by using msb first shifting at 32-bit level.

The overview of the path from the memory to the serial bus is shown in the next figure.

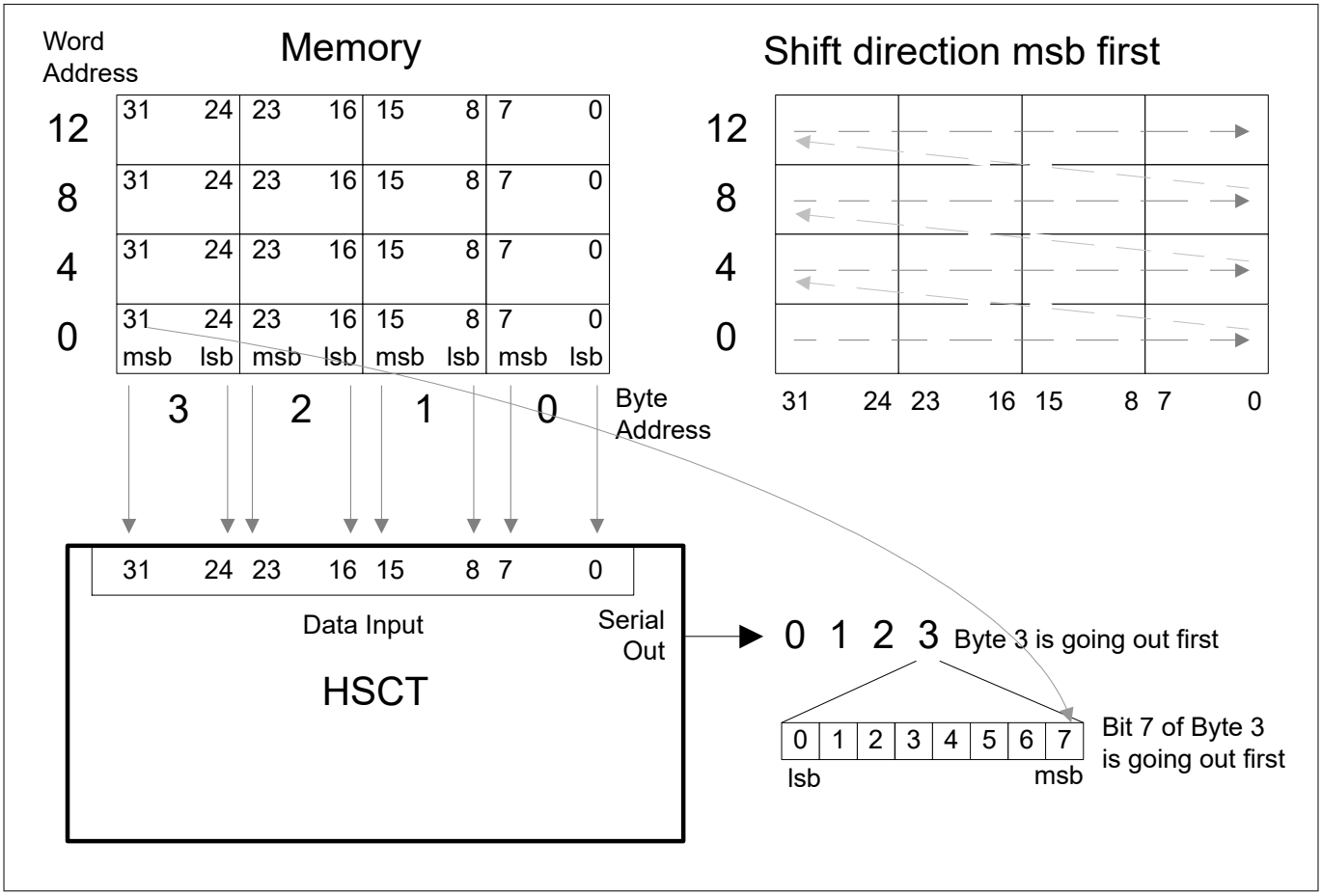


Figure 199 Shift direction in case of HSCT layer

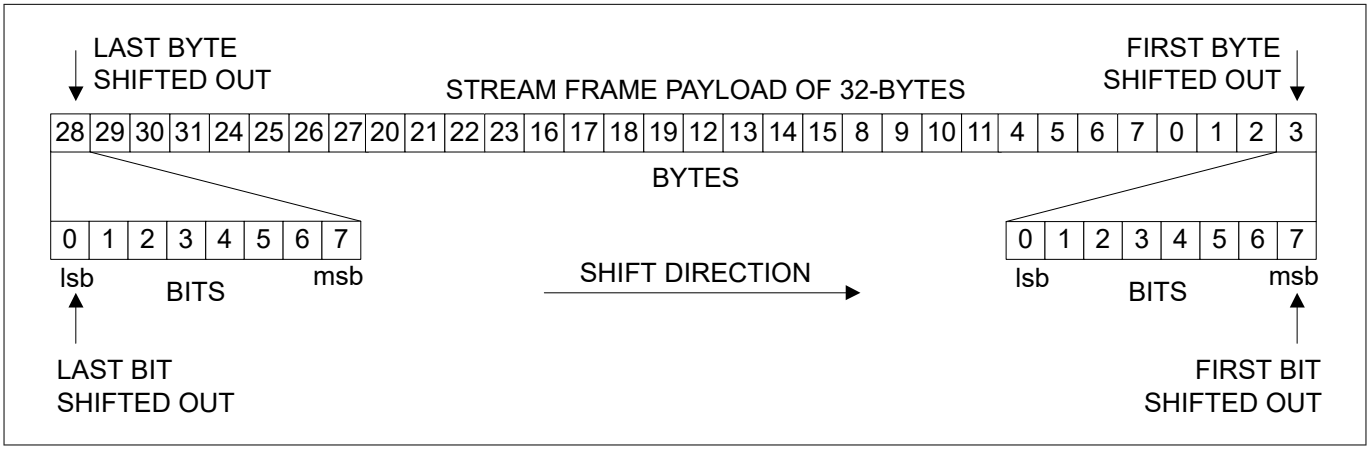


Figure 200 Stream frame in case of HSCT layer

17.4.3.2 HSSL implementation

This section describes the implementation of the HSSL protocol.

17.4.3.3 Overview

The HSSL module is connected to the SRI bus using a master interface. Additionally, the HSSL module is connected to the SPB bus with a slave and a master interface. Please refer to the System Interconnect for more details.

The SRI master interface is capable of performing both single and burst reads and writes on the SRI bus. The HSSL kernel provides logic for performing streaming communication. The write and read accesses are always performed in User Mode.

The SPB slave interface is used by an SPB master (DMA, CPU, HSSL SPB Master) for writing the HSSL registers, to configure the module and initiate the execution of streaming, read and write operations.

The SPB master interface is capable of performing single read operations, write operations, and streaming communication using BTR4 (Block Transfer Request) burst accesses on the SPB bus. The SPB master accesses are always performed in User Mode.

The HSSL module uses either the SRI master or the SPB master depending on the accessed address. The SPB address space (F000 0000_H – F3FF $FFFF_H$) is used as SPB master address window.

Accesses inside this SPB master address window are automatically routed to the SPB master, while accesses outside this SPB master address window are automatically routed to the SRI master.

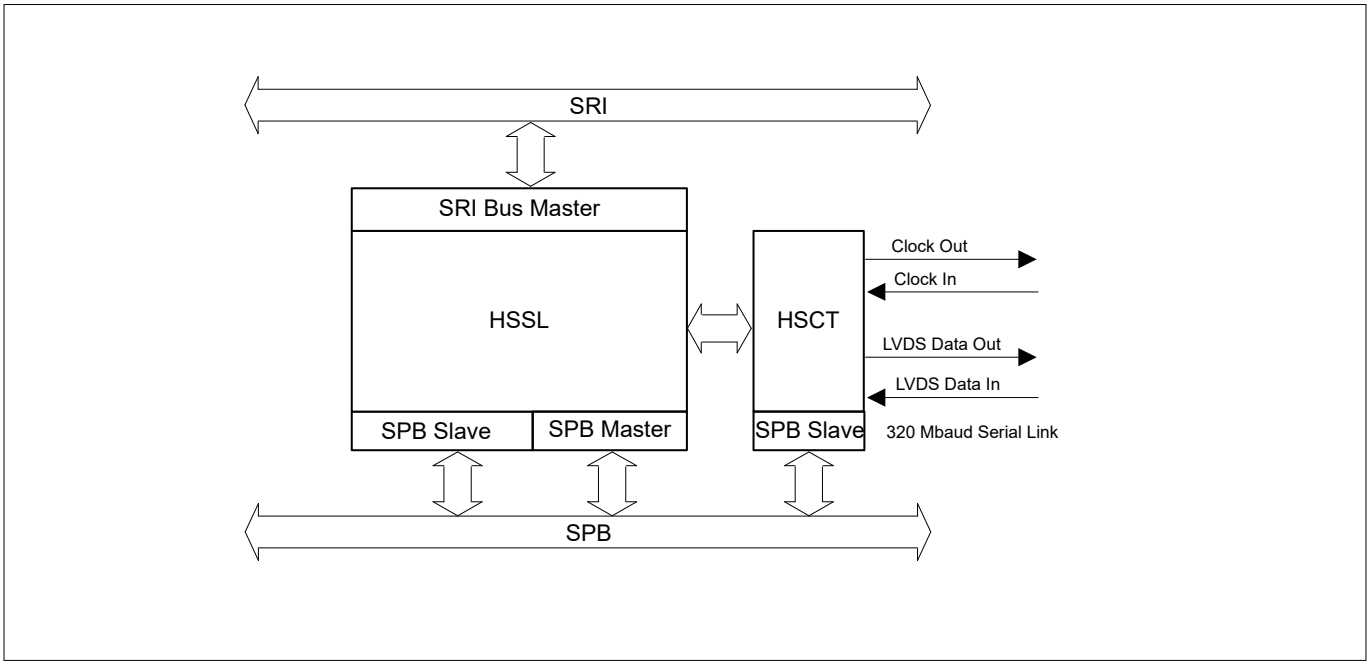


Figure 201 HSSL sub-system overview

***Note**: Any deviation from this generic description is described in the device specific chapter.*

17.4.3.3.1 HSSL operation

The HSSL module consists of separate transmitter and receiver, each containing four channels. The transmitter contains an arbiter block and a wrapper block, which are common for all channels.

17 High Speed Serial Link (HSSL)

The arbiter provides fixed arbitration between the channels which are ready to send when the serial link is available, with channel 0 having the highest priority, and channel 3 having the lowest priority. The wrapper block generates the CRC.

The receiver unwraps the HSSL frame, discards the frames with a CRC error, and distributes the error free commands and acknowledges to the channels.

A common prescaler generates the time basis for all channels, in particular for their timeout timers.

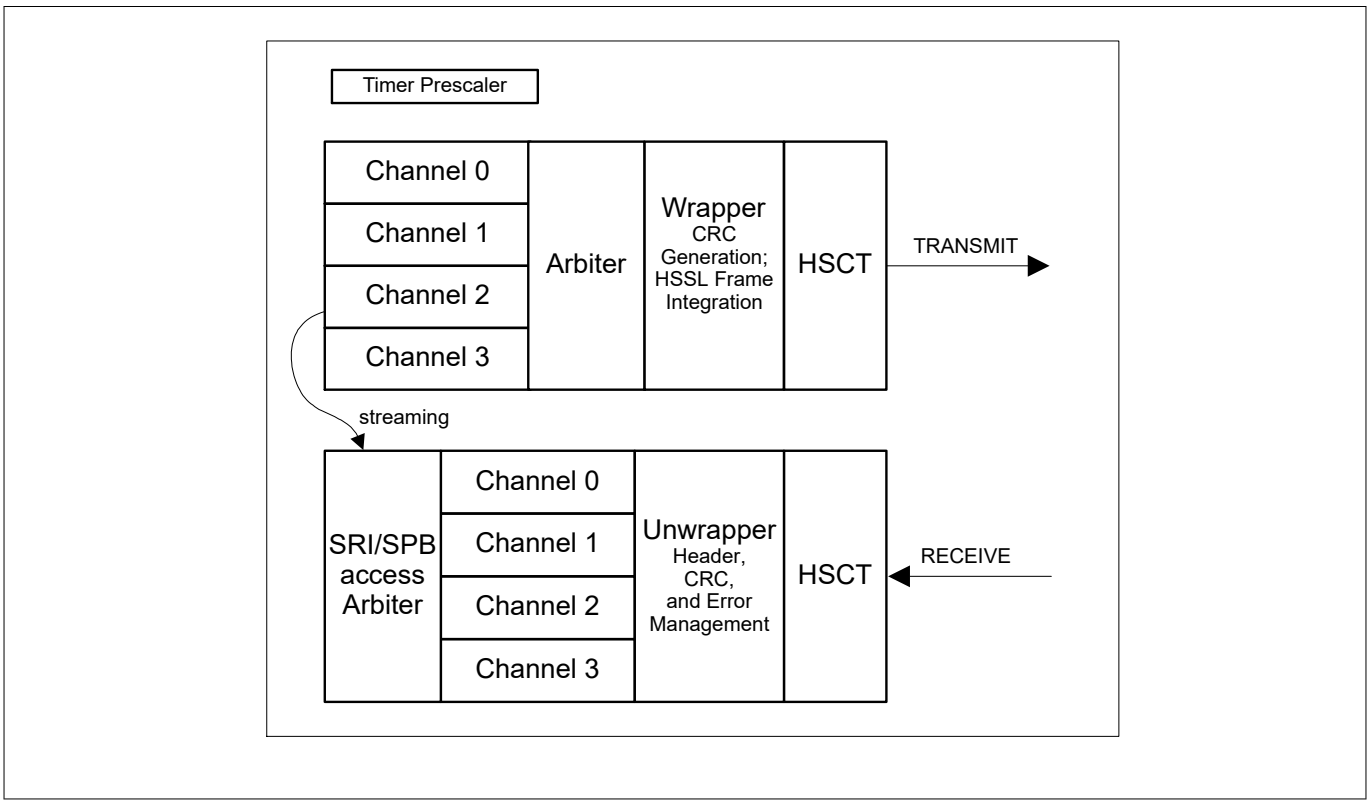


Figure 202 HSSL layer architecture

Frame transmission priorisation

When more than one frame is pending, an arbitration is performed between them. Each channel activates two flags: initiator command flag and response flag: channel 0 I0 and R0, channel 1 I1 and R1, and so on.

Between the response and command frame request types, the response request type has higher priority. Among the requests of a same type, a lower channel number has higher priority (see the figure that follows).

A pending frame is indicated with a corresponding request flag in the register QFLAGS.

The SRI/SPB bus master sets the response requests (“R” flags).

The software write accesses through the SPB bus set the initiator requests (“I” flags).

The transmit arbiter clears the “I” and “R” flags of the arbitration winner.

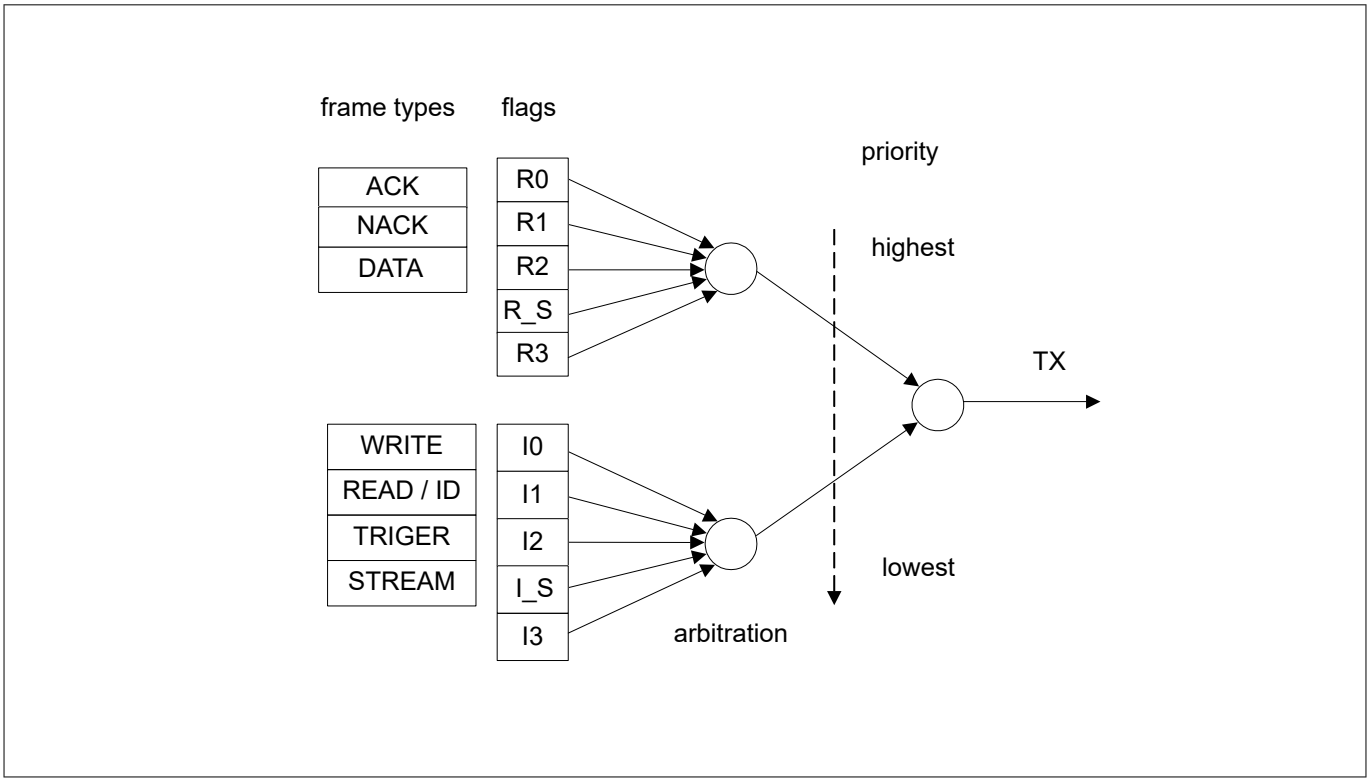


Figure 203 Frame priorities in the HSSL

Received frame management and command execution

After a frame passes the CRC check, it is checked for its type. A received command frame sets a request flag at Target for the SRI or SPB master, a received response frame resets the expect flag and stops the appropriate channel timeout,

The target flags “T” are set by the frame distributor, and cleared by the arbiter of the SRI or SPB master.

Target flags are set by the hardware according to the header information of a frame arrived at the target without a CRC error. They are used by the arbiter of the SRI/SPB master and cleared when the appropriate command is executed.

The expect flags “E” are set by the Tx arbiter, and reset by the Rx distributor.

An appropriate flag is set by the hardware when a timeout timer for a channel is started. The hardware stops the appropriate channel timeout when any response frame received resets the expect flag.

The ISB and ISF flags operate together to control the streaming process. The ISB flag (Initiator stream block flag) is set by the software to enable the streaming. The ISF flag (Initiator stream FIFO flag) is set or cleared by the TXFIFO according to its filling level.

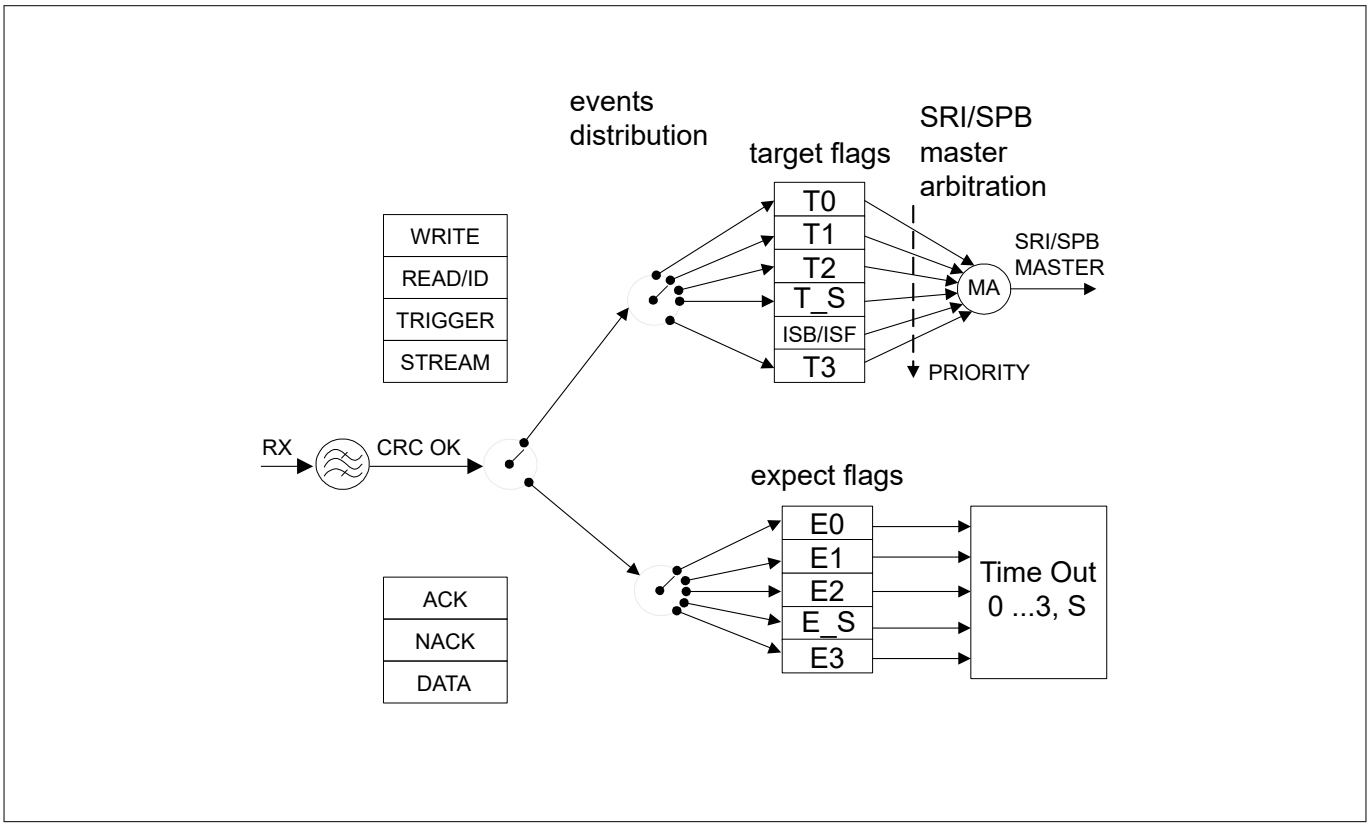


Figure 204 Received frames management

The CRC and Transaction Tag error flags are located in the MFLAGS register.

An overview of a full communication cycle over all sections of a channel (initiator, target or receive, transmit) is shown in the figure that follows, but see also the QFLAGS register description.

If the channel does not have a command pending and does not expect a response, it is ready.

***Note**: In case of at trigger command frame, the target command distributor sets the corresponding Rx flag, not the corresponding Tx flag, because the service request is triggered directly and not by the SRI or SPB master.*

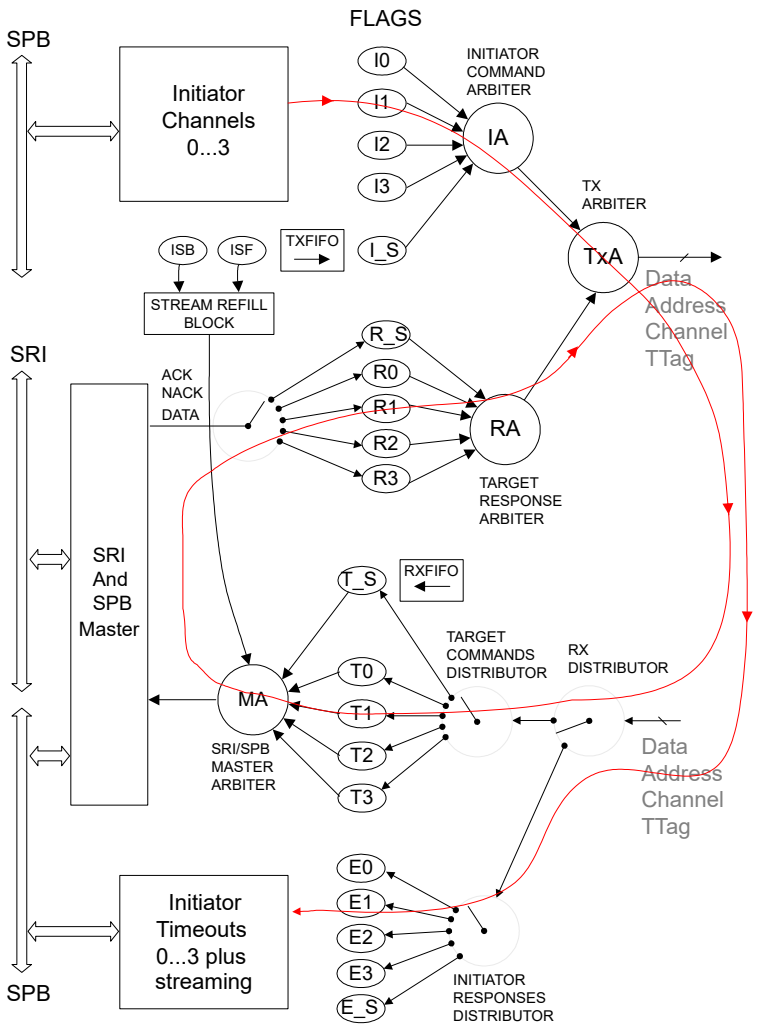


Figure 205 Request flags overview and a communication cycle example

Table 786 Set and clear of request flags

	I	R	T	E	ISB	ISF
SPB software write	set	-	-	-	set	-
Tx arbiter	clear	clear	-	set	-	-
Rx distributor	-	-[1)]	set	clear	-	-
SRI or SPB master	-	set	clear	-	-	-
Stream refill block	-	-	-	-	clear	clear
TXFIFO	-	-	-	-	-	set

1) In case of a trigger command, an R flag is set by the Rx Distributor instead of a T Flag, because a trigger command is not executed by the SRI/SPB master.

17.4.3.3.2 HSSL channel architecture

The HSSL module contains four channels. The functionality of each channel can be subdivided into four roles:

- Initiator Transmitter functionality: the channel is transmitting write, read or trigger command frames based on the arbitration described in Frame transmission priorisation, after the previous command has been responded to. It also generates an appropriate header for each frame including a transaction tag per channel to keep track of the expected response
- Initiator Receiver functionality: the channel does the comparison of the received response´s transaction tag with the one expected and do the interrupt handling

  ***Note**: The timeout management is a pure initiator functionality*

- Target Receiver functionality: the channel´s functionality is to receive commands and executes them. Interrupt handling is also a target receiver´s functionality
- Target Transmitter functionality: depending on the command type, the channel generates and transmits responses (read response, ACK, NACK) with the appropriate header for each frame. Since the reads and writes are performed by the SRI or SPB master, the module can send back a target error frame to the transmitter if an SRI or SPB bus error occurs and the transfer on the bus cannot be executed

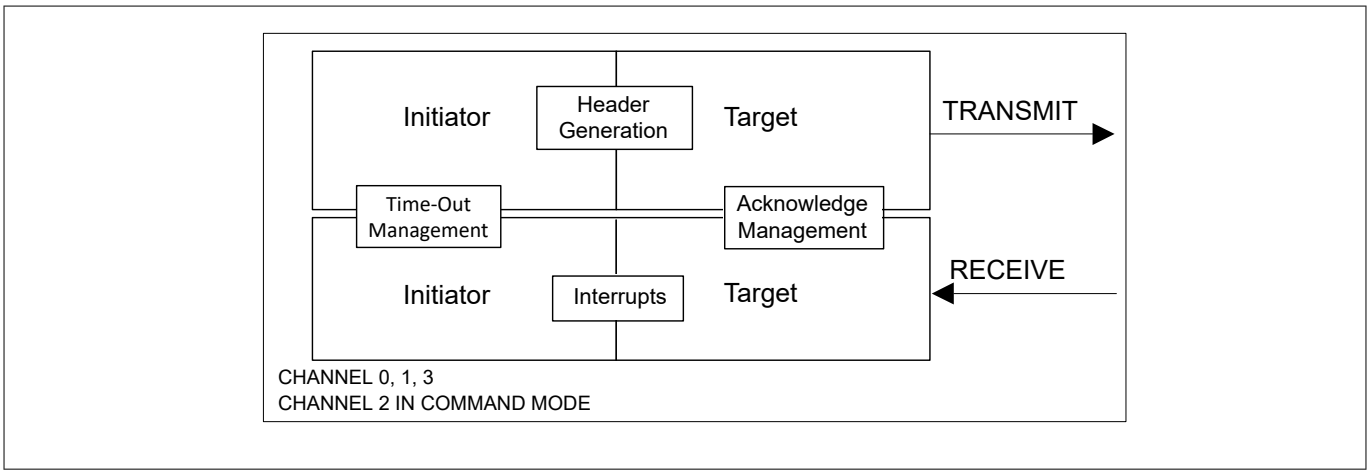


Figure 206 Architecture of channels 0,1, 3 and channel 2 in command mode

While all channels can operate in command mode using Read, Write and Trigger commands, only channel 2 can also be switched to streaming mode. Streaming mode allows to transfer a range of data from initiator to target device. In command mode a single data register is used and in streaming mode a FIFO is used.

The channel 2 contains one initiator and one target address to define the respective start address of streaming. A stream frame count register defines the size of data for the transfer.

The granularity of the start address is 256-bit.

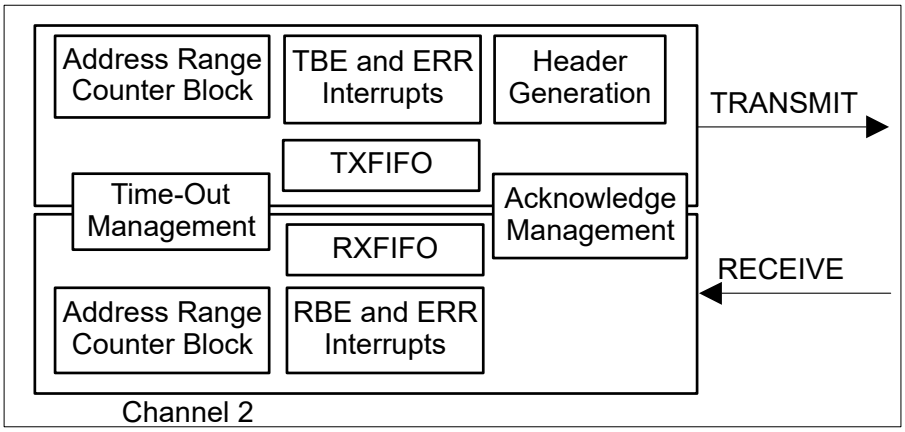


Figure 207 HSSL architecture of channel 2

	Initiator	Target
TX	Tx Commands SRI/SPB Master at Streaming	Tx Acknowledge Tx Transaction Tag Error Tx Read Answer
Rx	Rx Acknowledge Rx Transaction Tag Error Rx Read Answer	Rx Commands

Transmit / Receive related functions

	Initiator	Target
TX	Start Time-Out Timer Header Generation Command Frames Stream Frames	Header Generation Acknowledge Frames Target Error Frames
Rx	Stop Time-Out Timer Transaction Tag check	SRI/SPB Master Functions

Miscelaneous functions

Figure 208 Overview of channel functions

17.4.3.3.3 Acknowledge responses

A read or write command is executed by SRI or SPB master. The SRI or SPB master either reads form or writes to the SRI or SPB bus and signals either a successful transaction or an error. There are the following cases:

- When the SRI master either writes to or reads from an SRI memory space, in both cases the HSSL module receives an acknowledgment as feedback
- When the SPB master either writes to or reads from an SPB memory space, in both cases the HSSL module receives an acknowledgment as feedback

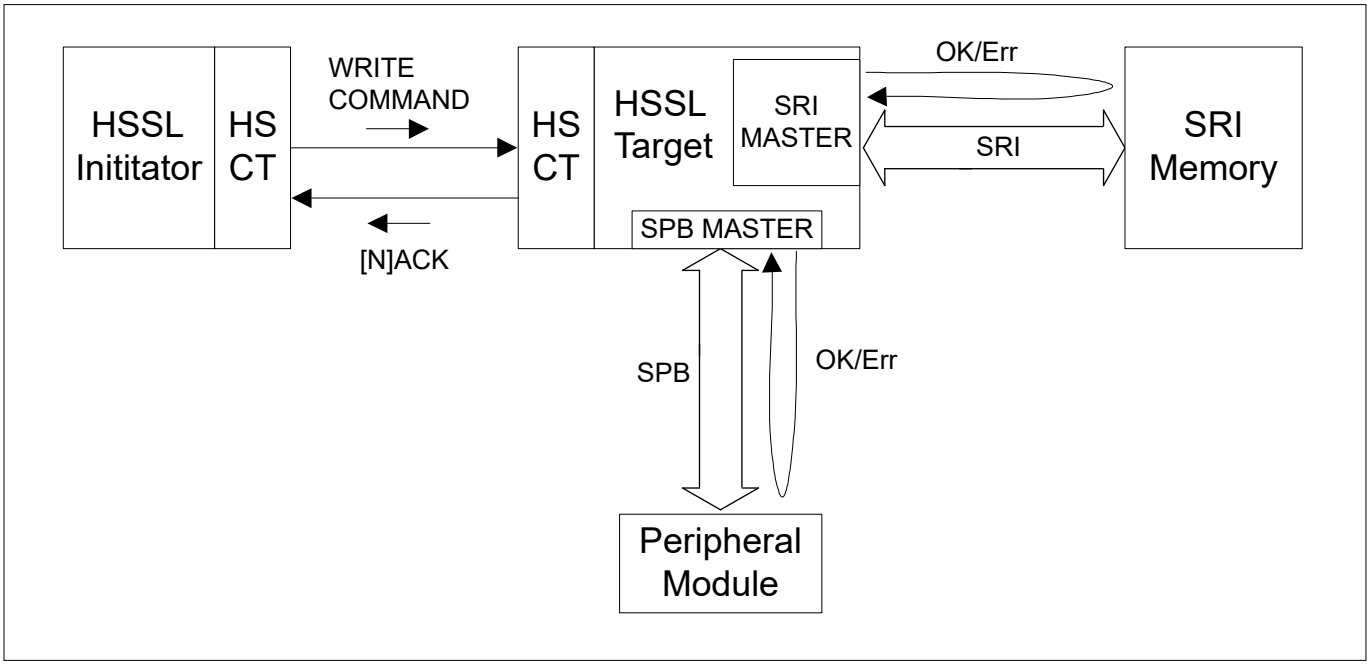


Figure 209 Bus error feedback paths

17.4.3.3.4 Cross-dependencies between the frame types

For a channel in a target role, there are cross-dependencies between a command frame arrival and triggering of an ACK or NACK frame for a channel. Each command frame received with a correct CRC of types Write, Stream or Trigger frame will result in either an ACK or a NACK (target error) response frame.

Each Read frame received with a correct CRC will result either with Data frame (Read Response frame) or with NACK (Target Error frame).

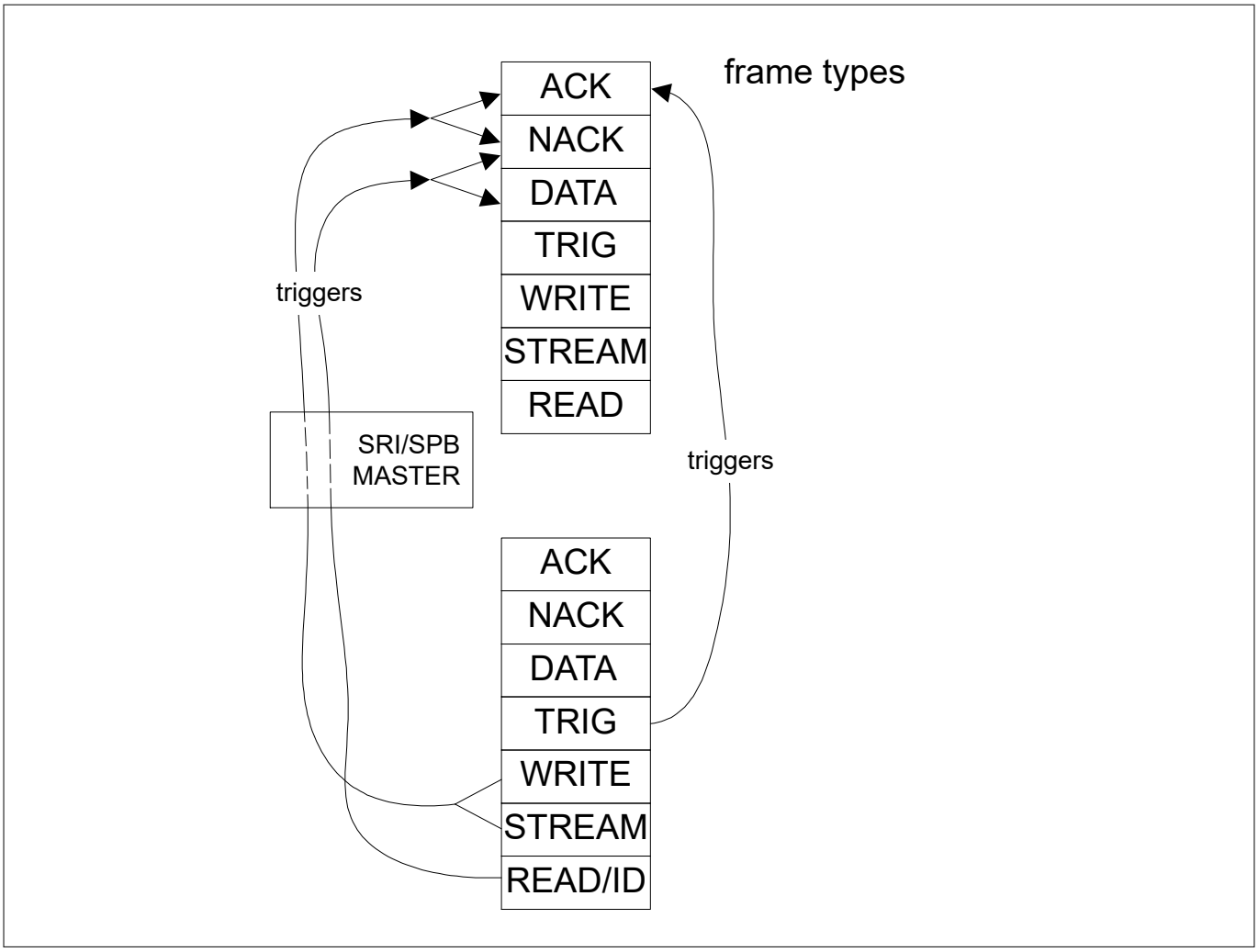


Figure 210 Channel 2 Cross-dependencies between stream and command frames and their responses in a target role

Generating a response frame involves copying the transaction tag of the command frame into the response frame. The response frame is triggered by a SRI or SPB master signalling either a successful SRI or SPB bus transaction or a SRI or SPB bus error. Only a Trigger Command frame triggers immediate acknowledgment because it results in an interrupt flag being set, and however it does not trigger a bus transaction.

On the initiator side, there is a forward looking cross-dependency between a command frame and the expected response. Setting a transmission request flag for a certain command is accompanied in parallel with:

- Capturing the current transaction tag in the bit-field ICONx.CETT
- Generating the next transaction tag by incrementing a three bit counter per channel, the ICONx.ITTAG
- Starting the timeout counter at the moment when the command wins the arbitration
- Setting an expect tag indicating that the timeout is running and a response frame is expected. See MFLAGS register

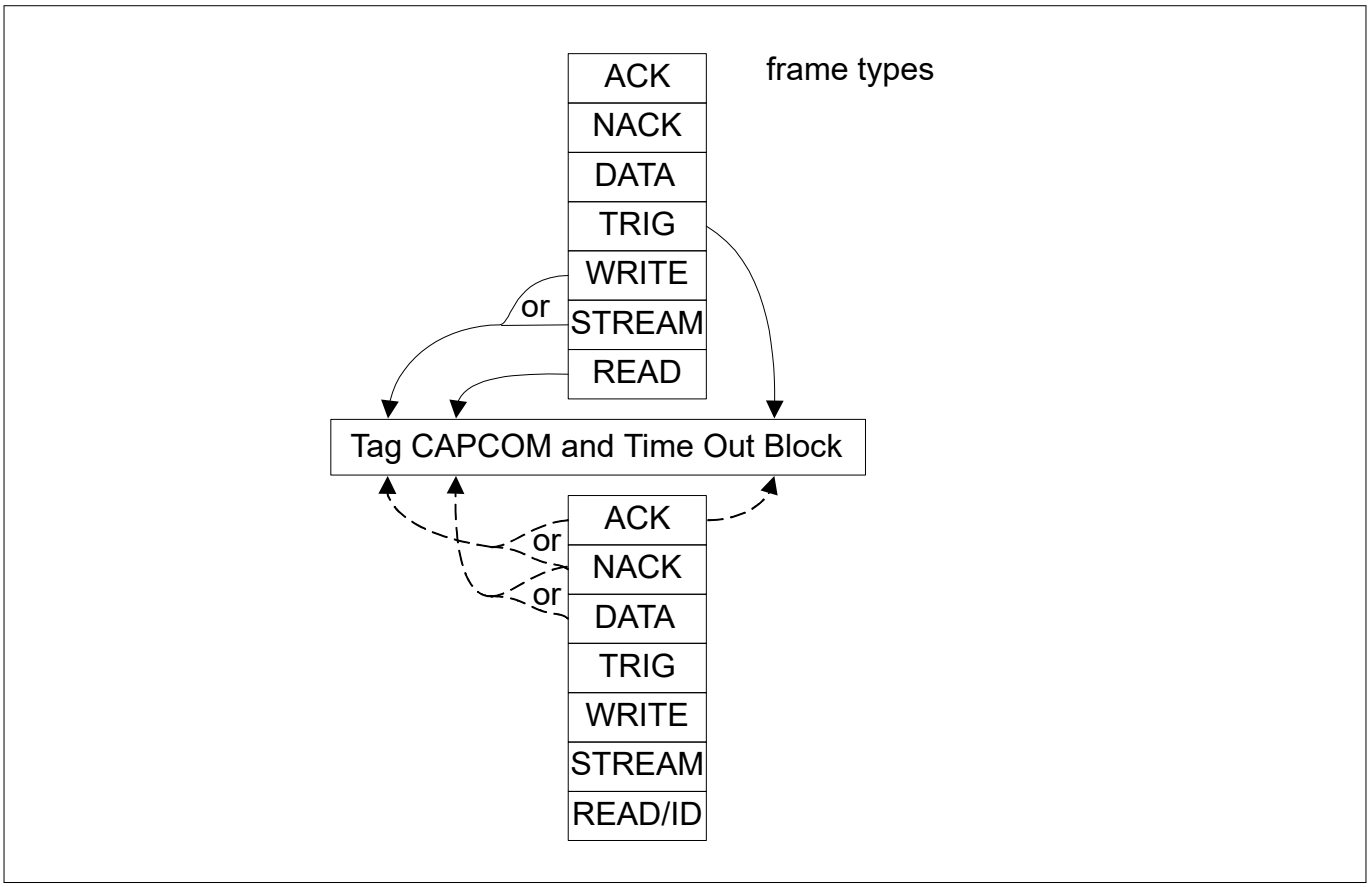


Figure 211 Channel 2 Cross-Dependencies between stream and command frames and their responses in an initiator role

17.4.3.3.5 Command timeout

A timeout timer is started when a frame wins the arbitration and is delivered by the channel for transmission. The timeout timer is stopped if an appropriate response frame without a CRC error is received in time.

If an appropriate response is not received, and the timeout timer reaches zero, then timer is stopped, a timeout error is triggered, and the expect flag is cleared.

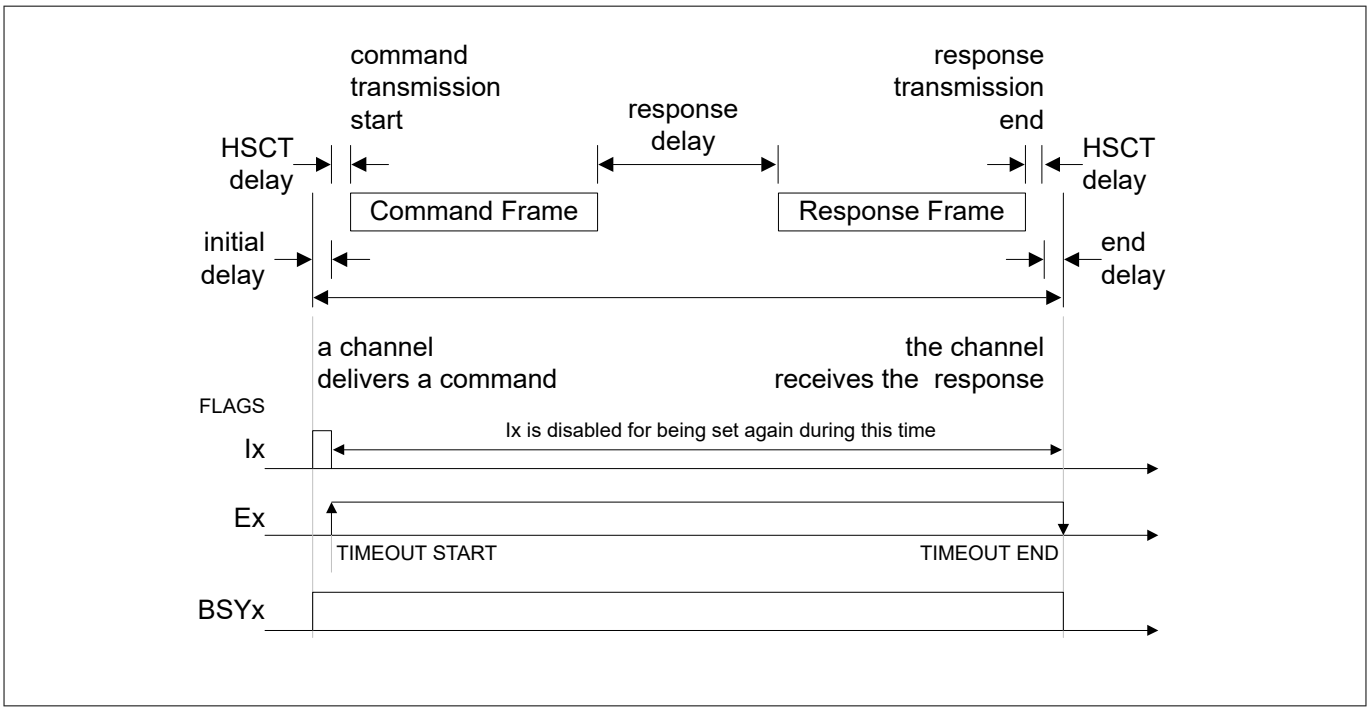


Figure 212 Command timeout measurement

Command timeout operation

The HSSL contains one prescaler common to all channels, and a channel specific timer block per channel. The timer block generates the transaction tag for the channel by incrementing a three bit counter, captures the current transaction tag, and compares it with the received response.

In case of an error sequence CRC and timeout error, the timeout block generates a timeout signal. The timeout timer is reloaded and starts down-counting when the command transmission process starts.

The current timeout timer value is indicated in the bit-field ICONx.TOCV.

The timeout reload value is configured in the bit-field ICONx.TOREL.

The current expected transaction tag is indicated in the bit-field ICONx.CETT.

The captured value of the latest erroneous transaction tag is indicated in the bit-field ICONx.LETT.

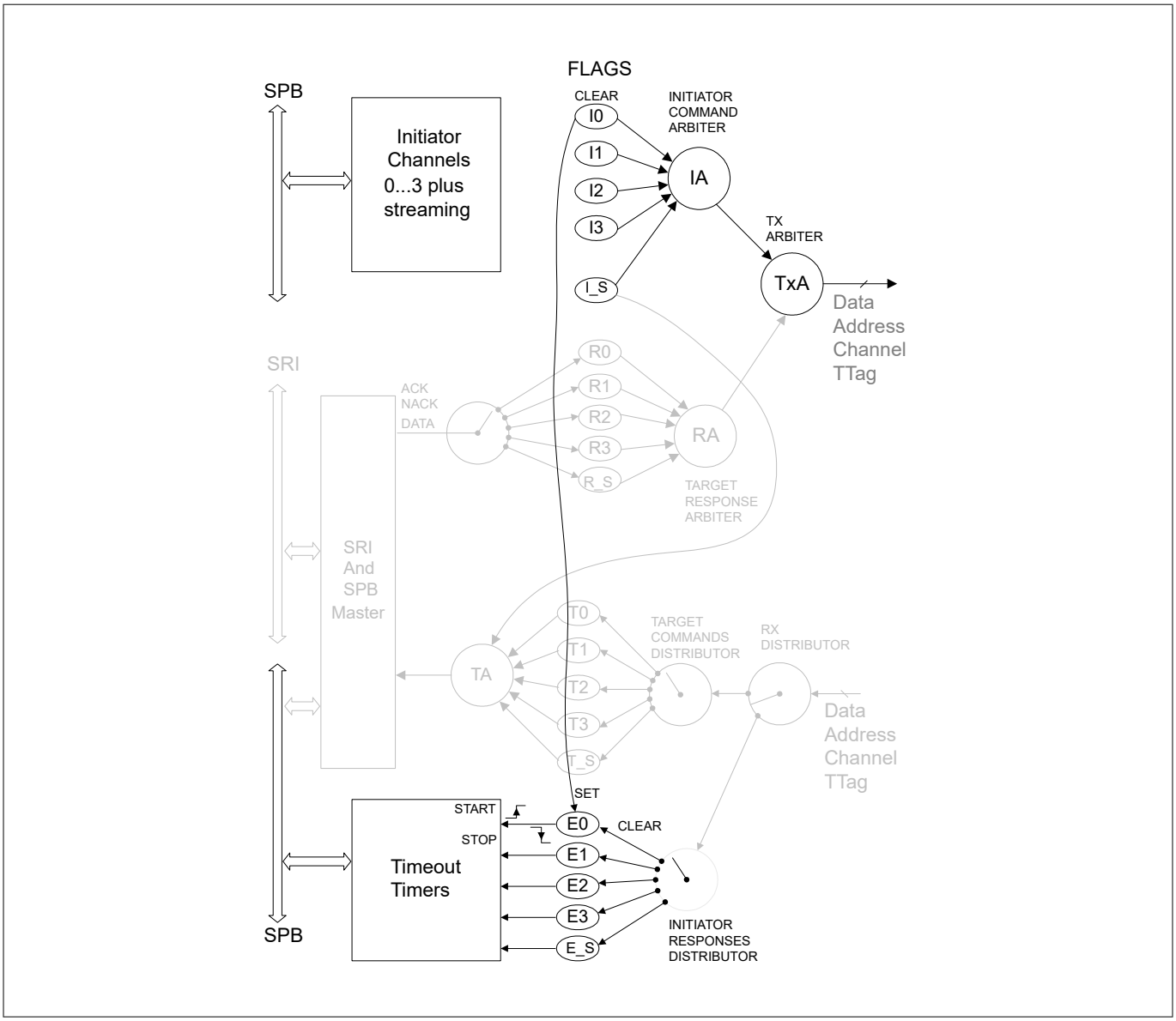


Figure 213 Request flags overview and a communication cycle example

From the flags point of view, the timeout starts with the rising edge of the QFLAGS.Ex flag and ends with its falling edge.

The Ex flag is set by the falling edge of the Ix flag, which is reset when its command request wins the transmit arbitration.

Related information

Cross-dependencies between the frame types on page 2454

17.4.3.3.6 Stream timeout

A free timeout timer block is started when a frame wins an arbitration round and is delivered by the channel for transmission. The timeout timer block is reset to zero and stopped if an appropriate response frame without a CRC error is not received in time.

If an appropriate response is not received, a timeout error is triggered and the timeout timer block is reset.

The stream initiator keeps track of the expected acknowledgements using a virtual expect FIFO having a virtual filling level .EXFL. Each new stream frame transmission increments the EXFL. Receiving a correct

acknowledgment decreases the .EXFL. As long as the 'Expect FIFO' is full (filling level 2) no new transmit request can be issued.

In order for a new stream frame request to be issued, which means the flag .I_S to be set by the module, three conditions must be met:

- EXFL<2 - the expect level must be below 2
- TXFL>0 - the TXFIFO must be not-empty
- ISB = 1 - the bit ISB must have been set by the software

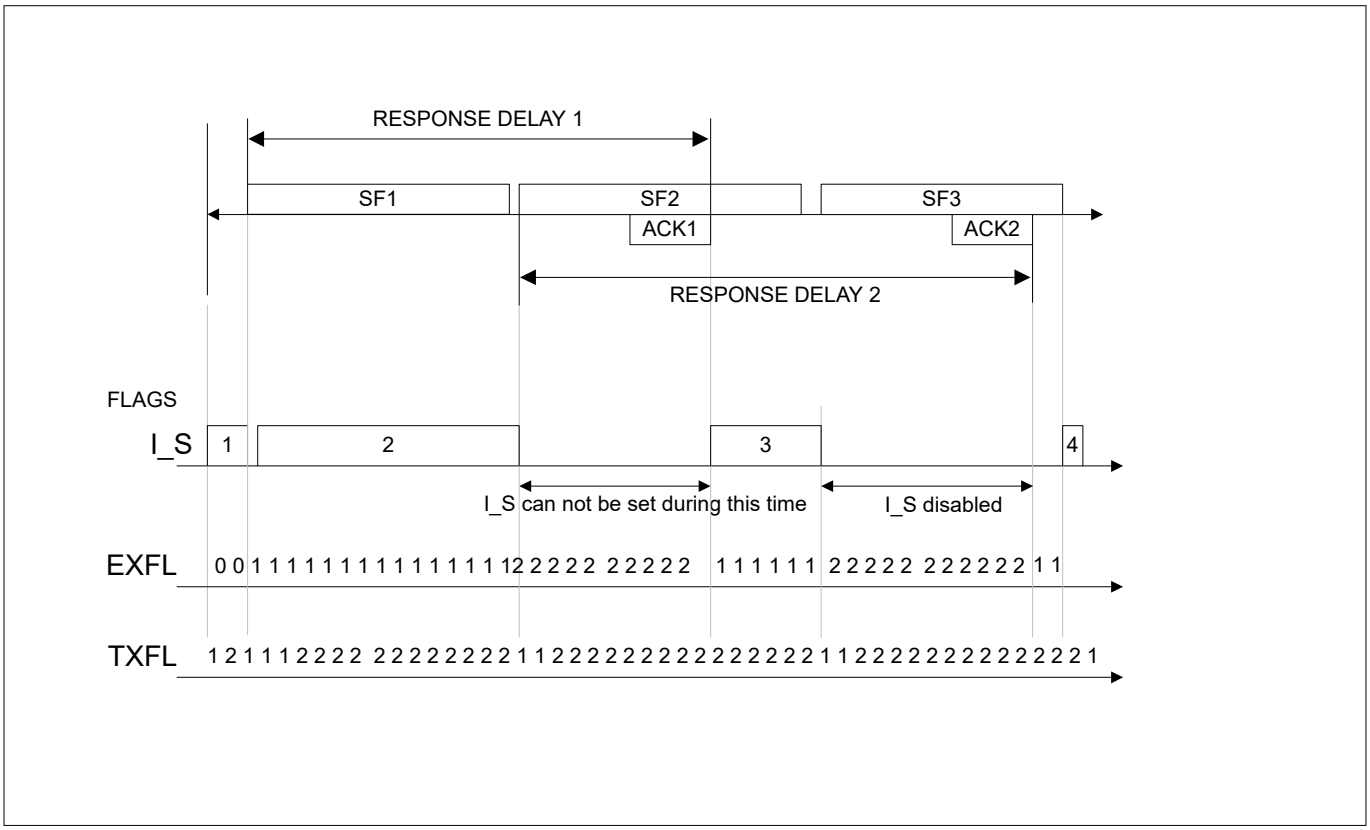


Figure 214 Stream timeout measurement

Stream timeout operation

The stream channel monitors the arrivals of the stream frame responses by using two timeout timer blocks. A single timer block is shown in the next figure.

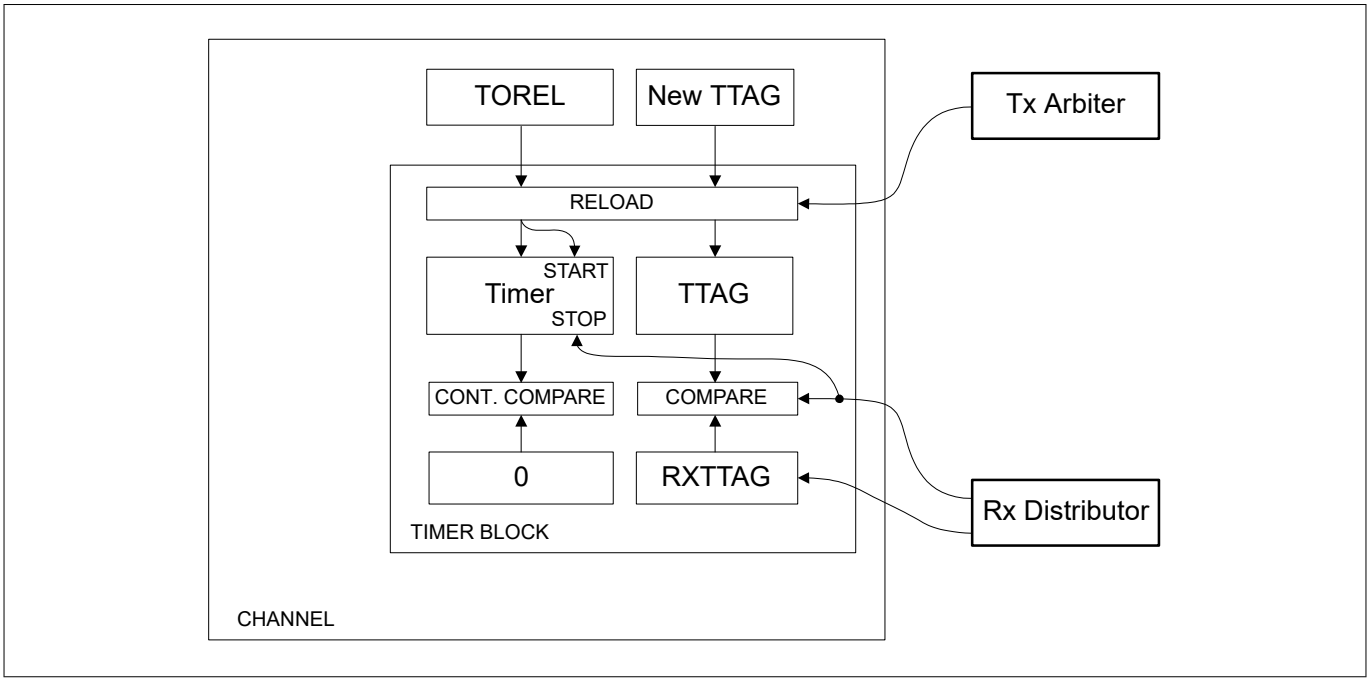


Figure 215 Single timeout block

The new initiator TTAG value which will be used for the next frame is visible in the bit-field ICONx.ITTAG. The Initiator Control Data Register x register also contains the TOREL and the TOCV bit-fields.

The streaming functionality uses two timers sharing the same reload value, as defined by the ICONx .TOREL, i=2. The timeout timer blocks simulate the behavior of a FIFO (named 'Expect FIFO'), by having additionally a “write” and “read” pointer:

- The “write” or transmit pointer points to the timer block where the next command frame will trigger the writing of the timer reload value and the transaction tag
- The “read” or receive pointer points to the block which performs the transaction tag comparison with the currently arrived stream acknowledge TTAG
- The EXFL or virtual filling level indicating how many timers are active 0, 1, or 2

The EXFL is visible in the register SFSFLAGS.

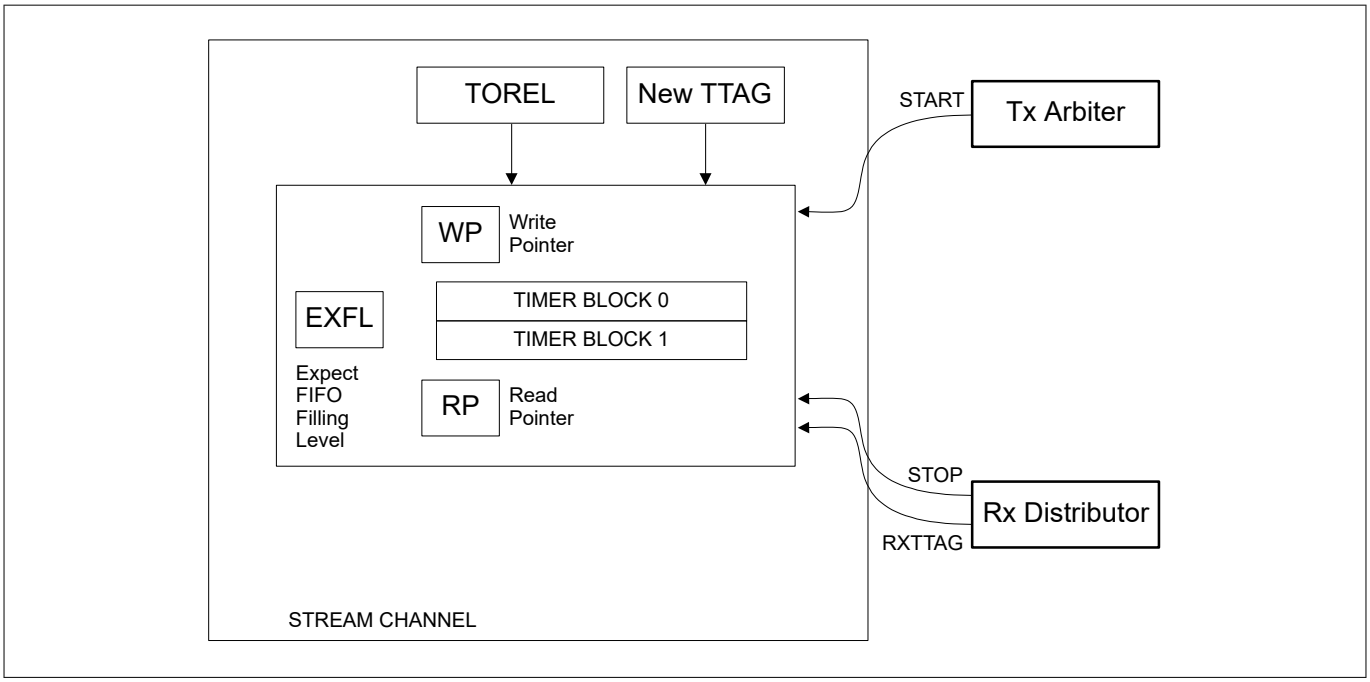


Figure 216 Stream timeout block

17.4.3.3.7 Data FIFOs of the streaming channel 2

Each stream frame delivers 256-bit data. This data is transferred by the SRI/SPB bus master in a memory by using BTR4 bursts. The start address and the end address is aligned at 256-bit block border.

Refer to:

- ISSAx
- ISFC
- TSSAx
- TSFC

The channel expects one full streaming frame of 256 bit to be delivered before triggering either the SRI or SPB master or the HSCT module for fetching the data. The streaming is full duplex capable, which means one FIFO for each direction is available. It also supports prioritization for the requests to the SRI or SPB master like emptying the RXFIFO has higher priority than filling the TXFIFO.

A service request to the SRI or SPB master is generated when the complete payload of a streaming frame is available in a FIFO.

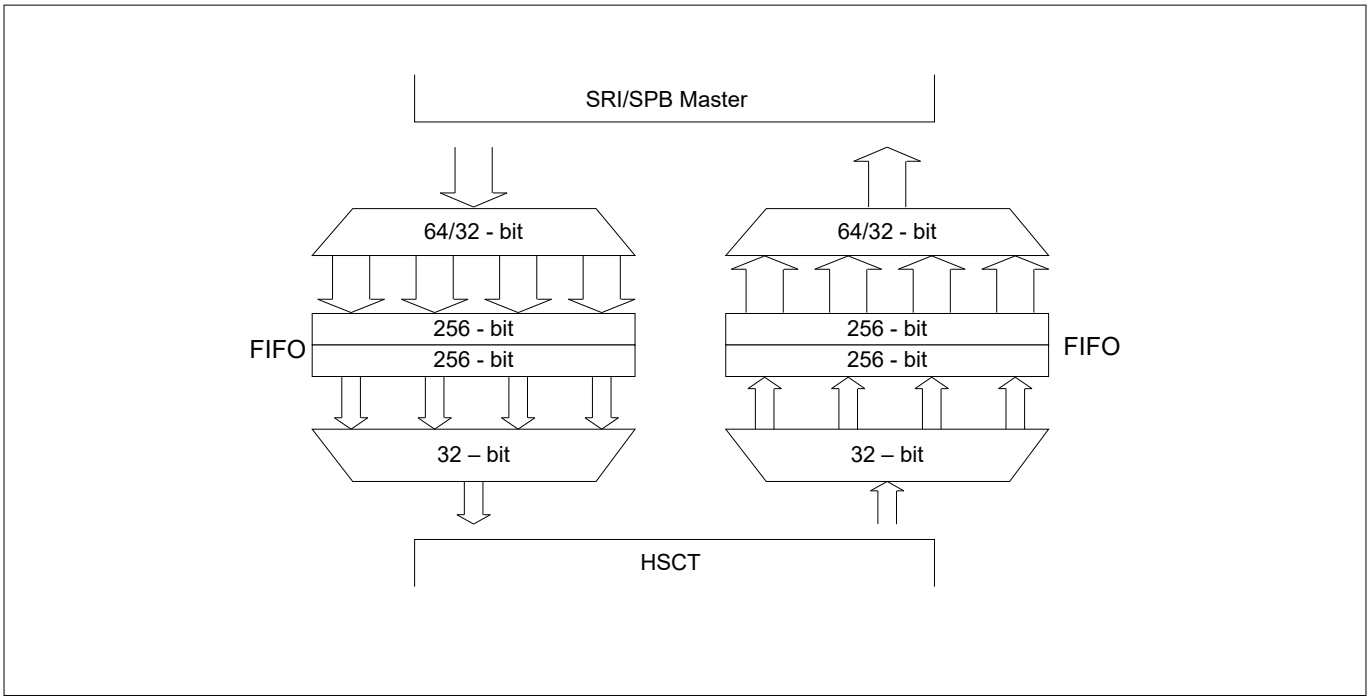


Figure 217 Streaming FIFO

17.4.3.4 Modes of operation

The HSSL module has the following modes:

- Disabled mode (software controlled through CLC.DISR bit)
- Initialize mode (software through MFLAGS.INI bit, sleep and suspend signals)
- Run mode (MFLAGS.INI bit, sleep and suspend signals)
- OCDS soft suspend mode (OCDS suspend signal)
- Sleep mode (sleep signal)

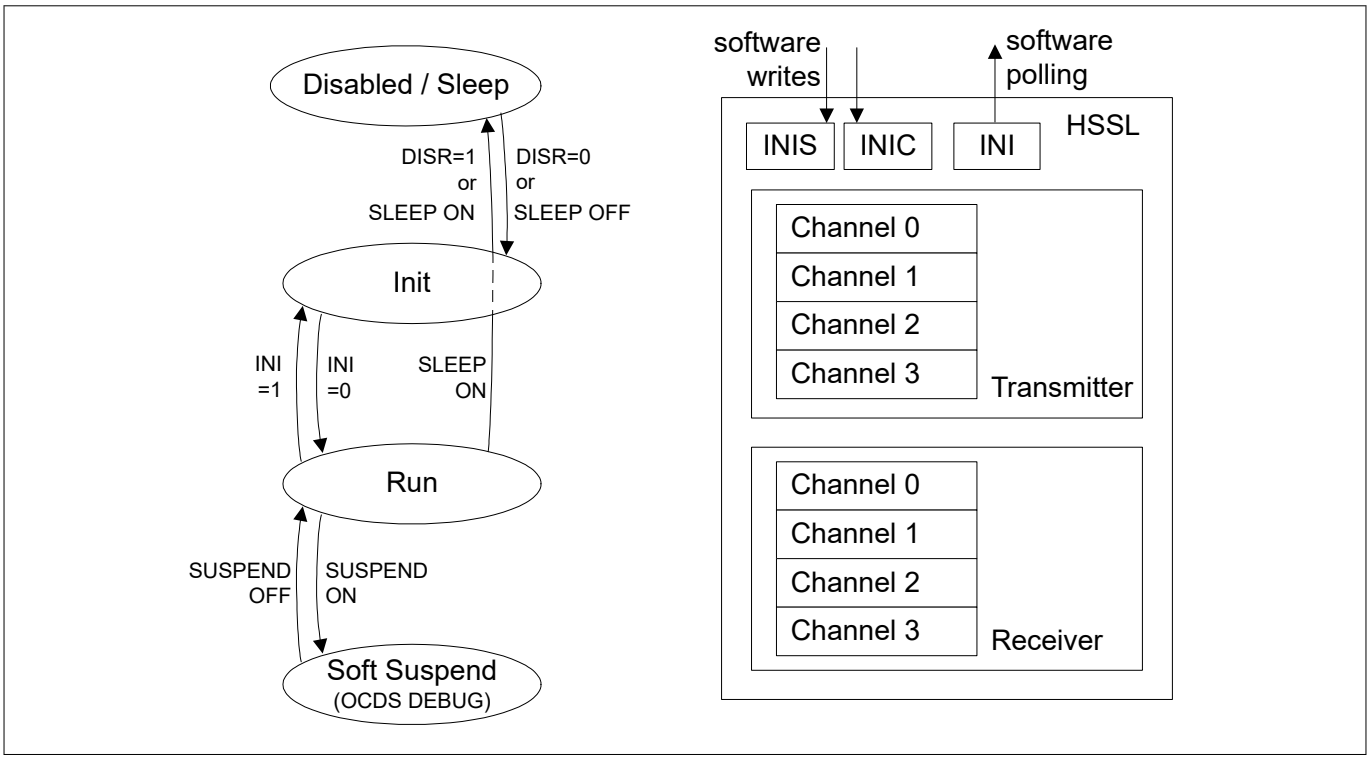


Figure 218 Modes of operation

After Init state has been requested by using the MFLAGSSET.INIS bit, the HSSL module immediately stops starting new SRI or SPB transactions and new command, response, and stream frame transmissions by resetting all request bits (I, T, and R) in the QFLAGS register, disabling these bits from being set again. Any transactions or transmissions that have already started will be finished. Afterwards, it waits for all pending response frames to arrive, or the corresponding timeout events to be raised. This condition is achieved when all 'expect bits' in the QFLAGS register has been cleared by the hardware, and there are no ongoing transmission and SRI or SPB master activities. Subsequently the module enters the Init state and the MFLAGS.INI flag is set. The Clear To Send(CTS) signal is deactivated to inform the peer module about the Init state.

During the transition from run to sleep state, triggered by the sleep signal, the MFLAGS.INI bit is automatically set by the hardware (and behaves as if the INI bit was set by software). When the sleep period is over and the sleep signal is deactivated, the module goes into the Init state and must be set to Run state by software, by writing MFLAGSCL.INIC=1.

At the state transition of Init to Run, the timeout timers and all the flags are reset. The RUN state starts with all channels inactive.

At the state transition Soft Suspend to Run, the timers are not reset but continue to run and the flags states are preserved, because the module continues operating from the point at which it was suspended.

When leaving the Run state towards Sleep, the CTS signal is deactivated and the transmission of the pending commands is stopped.

When a soft suspend is requested, the module stops the transmission of the pending command and stream frames, waits until all 'expect' flags are cleared, then deactivates the CTS signal and at the end acknowledges the soft suspend request and sets the bit OCS.SUSSTA. Incoming target frames are served until the CTS signal stops the other side from sending new command and stream frames.

A Hard Suspend request causes the module clocks to switch off immediately. After completion of the Hard Suspend state, both HSSL and HSCT modules must be reset by the application software, and communication restarted from reset state.

***Note**: The Kernel clock f_{CLC} is enabled for a few cycles, where the read from and write to registers is possible*

Attention**: **Register accesses with clocking in Hard Suspend mode can have unintended side effects such as signals becoming and staying active. This can also affect other modules so a HSSL kernel reset might not be sufficient to bring the system into a defined state.

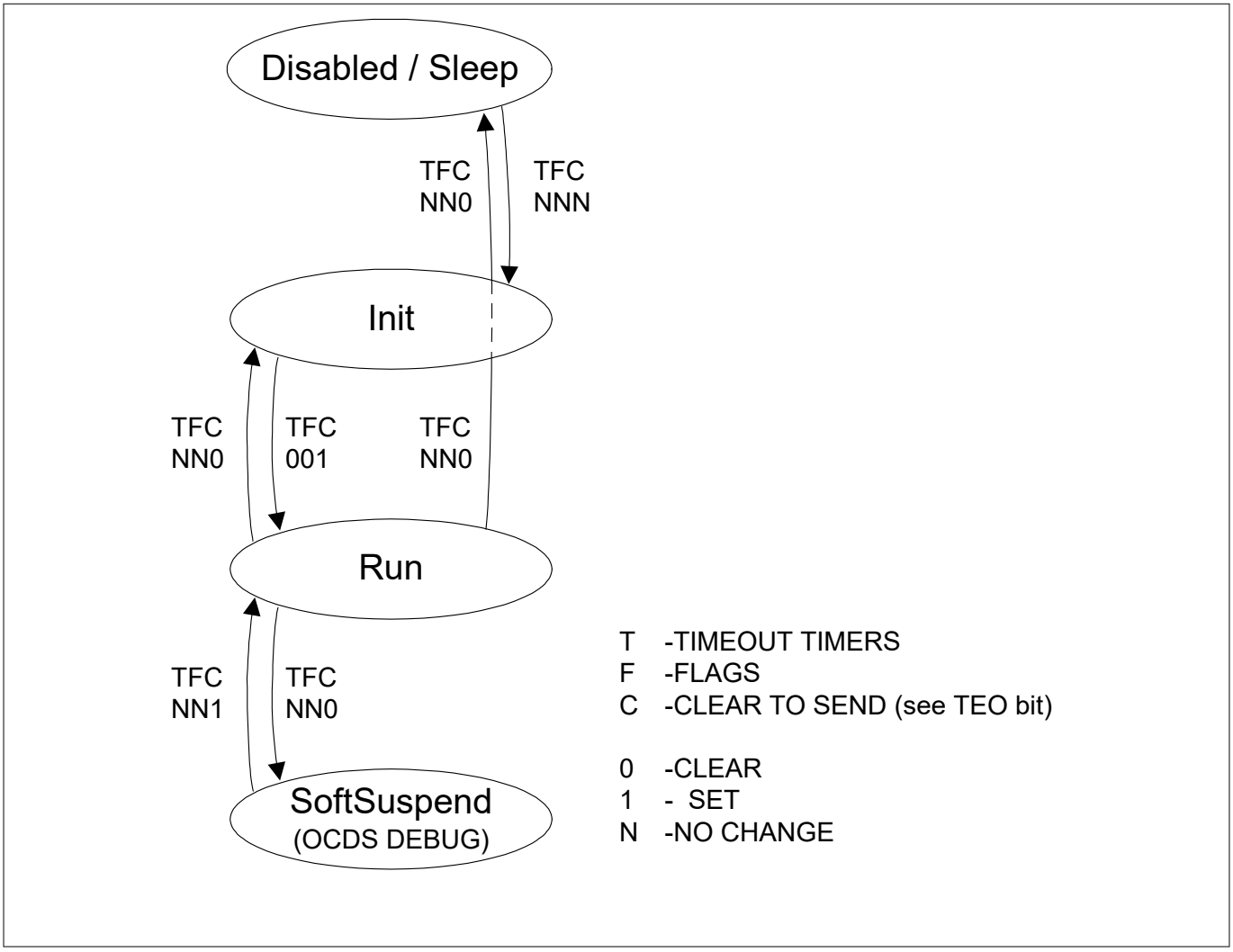


Figure 219 Actions on state transitions

17.4.3.5 Interrupts

The HSSL module generates four types of interrupts per channel and one channel unspecific module interrupt.

Command channel interrupts

Each HSSL command channel generates four interrupts:

- Command OK interrupt COK
- Read Data interrupt RDI
- Error interrupt ERR
- Trigger interrupt, triggered by a trigger command frame TRG

The names of the corresponding interrupt signals are:

- COK
- RDI
- ERR
- TRG

The arrival of an error free frame response (ACK frame) triggers a COK interrupt, otherwise an ERR interrupt is triggered. The arrival of a read response frame triggers at the initiator side, in addition to the COK interrupt, an RDI interrupt.

The arrival of a trigger frame at the target side triggers a TRG interrupt there.

ERR interrupt is disjunct to COK and RDI, which means that for one frame, either ERR or COK and optionally RDI can occur. After an ERR interrupt, normal transmission must be resumed by software, because an optional DMA would remain and not be re-triggered, and would wait for COK indefinitely. The events that trigger an ERR interrupt are indicated by the bit fields UNEXPECTED, TIMEOUT, TTE and NACK corresponding to Miscellaneous Flags Register. These bits can be enabled or masked and cleared by the registers Flags Enable Register and Miscellaneous Flags Clear Register respectively.

Stream channel interrupts

The HSSL stream transmit sub-channel generates three interrupts:

- TBE
  - Transmit Block End interrupt, shared with the COK interrupt of the command mode of the channel 2
- Error interrupt ERR
  - Shared with the ERR interrupt of the command mode of the channel 2
- RBE
  - Receive Block End interrupt, shared with the TRG interrupt of the command mode of the channel 2

Exception interrupt EXI

If the receive stage of the HSSL module detects a CRC error or any inconsistency in the received data, the global EXI Interrupt is activated, which are not channel specific. The events that trigger an EXI interrupt are indicated by the bitfields CRCE, PIE2, PIE1, SRIE and MAV corresponding to Miscellaneous Flags Register. These bits can be enabled or masked and cleared by the registers Flags Enable Register and Miscellaneous Flags Clear Register respectively.

***Note**: The HSCT module uses channels A to D for the HSSL channels 0 to 3. So, HSCT channel 0100_B corresponds to 000_B (binary code) or 100_B (special code) in the HSSL header.*

The EXI interrupt is also raised on an edge of the TEI signal (HSCT CTS output).

In total, the HSSL module provides $4 \times 4 + 1 = 17$ interrupt lines That is: number of channel(4) * channel specific interrupts(4) + module interrupt(1)

17.4.3.6 Operating a command channel

All HSSL command channel provide identical functionality.

17.4.3.6.1 Initiating a single write command

In order to initiate a write command, the software must:

- Configure the data width in ICONx.DATLEN and set the command type to write request in ICONx.RWT
- Provide the new data (often but not necessary) in IWDx register
- Provide new address in IRWAx register

Write to the address register requests the predefined type of the transfer.

17.4.3.6.2 Initiating a single read command

In order to initiate a read command, the software must:

- Configure the data width in ICONx.DATLEN and set the command type to read request in ICONx.RWT
- Skip providing the data step or provide dummy data
- Provide new address in IRWAx register

Write to the address register requests the predefined type of the transfer.

17.4.3.6.3 Initiating a single trigger command

In order to initiate a trigger command, the software can either:

- Request a trigger type of frame in ICONx.RWT and provide dummy data (optional) and address respectively in IWDx and IRWAx registers
- Directly request a trigger frame by writing to ICONx.TQ

17.4.3.6.4 DMA operated command queues

It is possible to use DMA to initiate lists of commands, by moving memory blocks containing the command lists to the HSSL module. In order to support such operation, each channel provides three registers arranged in a particular way.

The three registers are in the following order IWDx, ICONx, and IRWAx.

To execute a list of write commands all three registers must be written for each command.

To execute a list of read commands the first register IWDx is optional (dummy write possible) but the last two, ICONx and IRWAx, are mandatory.

To execute a trigger command only by writing to ICONx.TQ is mandatory, but optionally all three registers can be written: dummy write to IWDx, write to ICONx.RWT and write to IRWAx to issue the request.

The Answer data to a read command is available in the IRDx register.

***Note**: This approach relies on external peripherals like system DMA for command queuing without the intervention of the CPU. In case of a failed transaction, CPU has to be involved for recovery purposes.*

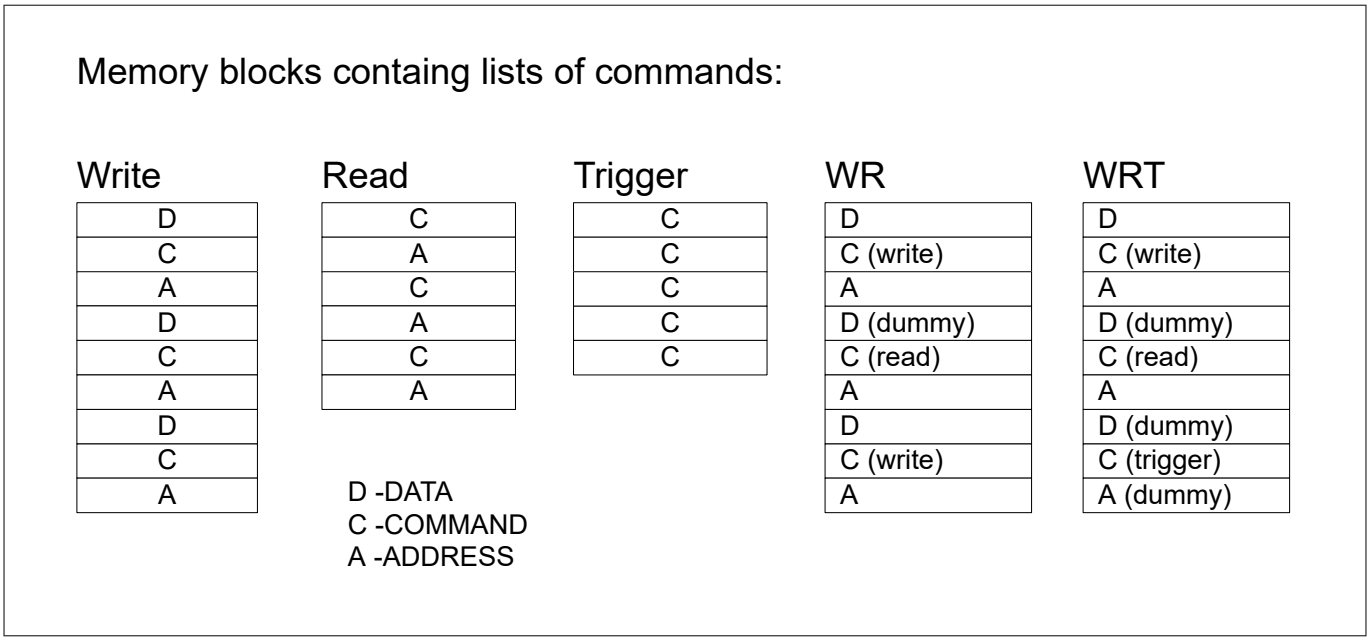


Figure 220 Command queues

17.4.3.6.5 Receiver error handling

The HSSL receiver generates three error signals, two channel specific and one general.

- Channel specific timeout and transaction tag errors
- Channel non-specific CRC error

Receiver errors are ignored in the following cases:

- Command frames, which pass the CRC check, containing an RFU command code are ignored
- Command frames, which pass the CRC check, containing invalid (greater than 3) channel number are ignored
- If a command frame is received and the acknowledgment of the previous command has not been sent yet, the new command frame is discarded
- If a response frame comes and the channel does not expect a response (the E flag is 0), an incoming response is discarded without any error interrupts in addition to the timeout interrupt. This scenario can occur if a response is received after a timeout and before the next command has been sent
- If a channel in command mode receives a stream frame, the stream frame will be ignored
- if a channel in streaming mode receives a command frame, the command frame will be ignored

Timeout error

Error triggered by the initiator.

Transaction tag error

Error is triggered by the initiator.

17.4.3.6.6 Global error

A global, non-channel specific error is triggered if a CRC error, or an inconsistency between the HSCT and HSSL header, or an SRI or SPB error occurs.

A CRC error is triggered:

- By the target when erroneous command frames are received
- By the initiator when erroneous response frames are received

There is a dedicated interrupt named EXI which signals the CRC error. There is no indication if the CRC error was caused by a command or response frame. The CRC error caused by a response frame will be followed by a timeout interrupt. The received data is discarded and it is not available for the software in any register(s).

Additionally, the CRC error interrupt signals an inconsistency between the data length and channel number code delivered by the HSCT module and the information contained in the HSSL header. The differentiation between the two types of error can be made by reading the MFLAGS.CRCE, PIE1, and PIE2 flags. All three error types can occur and are detected independent of each other, so that more than one flag can be set at the same time.

An SRI error interrupt (EXI) is triggered if there was a transaction ID error, ECC error, or SRI error acknowledgment on the SRI bus. The flag MFLAGS.SRIE indicates this type of error.

In case of an SRI read data ECC error, the HSSL SRI Read Data EDC Error signal reports this event to the SMU module.

An SPB error also triggers the global error interrupt (EXI) and sets the MFLAGS.SRIE flag.

***Note**: For one single 256-bit streaming data block, since BTR4 bursts are being used by the SRI master to transfer data, the HSSL SRI Read Data EDC Error signal could be asserted up to 4 times (once on every SRI read data ECC error occurrence in the 64-bit data transfer).*

17.4.3.7 Memory block transfer modes of the stream channel

The HSSL streaming channel is capable of transmitting one stream and receiving one stream in parallel. Streaming operates either in single block mode or continuous mode.

Single block mode

- In single block mode, after triggering or enabling the streaming by using the MFLAGS.ISB/TSE bit, the SRI or SPB master of the HSSL module transmits or receives the pre-configured memory block, generates an interrupt signal, and waits for the next trigger
- In single block mode, the hardware resets this bit after performing a block transfer

Continuous mode

- In continuous mode, after triggering or enabling the streaming by using the MFLAGS.ISB/TSE bit, the SRI or SPB master of the HSSL module transmits or receives two memory blocks of the same size in a round-robin fashion indefinitely, or after being stopped by the software
- In continuous mode, the transfer is ongoing continuously (MFLAGS.ISB or TSE remains set) and a stop of the transfer must be requested by writing one to MFLAGSCL.ISBC or TSEC bit.

Initiator and Target configuration:

- The streaming mode of the initiator or transmitter is configured by using the CFG.SMT bit
- The streaming mode of the target or receiver is configured by using the CFG.SMR bit
- Triggering or enabling a block or continuous stream is done by using the Stream Block Request bit MFLAGS.ISB or TSE bit
- On the initiator side, after completing the transfer of the current memory block, that is currently read from the memory, the transmitter will be stopped and the MFLAGS.ISB will be cleared
- On the target side, after completing the transfer of the current memory block, that is currently written to the memory, the receiver will be stopped and the MFLAGS.TSE will be cleared

Error handling:

- A SRI or SPB error on the target triggers an error interrupt and the hardware stops. The TSE bit (and the RXFIFO) will be cleared by the hardware and the incoming frames are ignored
- Any error on the initiator side, such as an SRI or SPB error, a target error, a timeout, or a transaction tag error, triggers an error interrupt and the hardware stops streaming. The bit ISB (and the TXFIFO) will be cleared by hardware. All the incoming frames will be ignored, including the acknowledge frames of the previous frames
- Error can also occur at the beginning of a new memory block, while the last frames of the previous memory block are still in the TXFIFO. These frames are also lost, and the previous memory block remains partially received
- After an error, while streaming in one direction, the software must restart the streaming in that direction at both sides. The streaming will restart with a new block
- An error while streaming in most cases leaves the target waiting in the middle of a block. The target receiver can be stopped immediately, and afterwards restarted, by clearing the MFLAGS.TSE only when TSFC.CURCOUNT equals TSFC.RELCOUNT.
- Normally, TSFC.CURCOUNT equals TSFC.RELCOUNT before the start of a transfer of a block or between block transfers. Otherwise, while waiting in the middle of a block transfer, TSFC.RELCOUNT can be made equal to TSFC.CURCOUNT by writing the TSFC.CURCOUNT value to TSFC.RELCOUNT. This can be done either locally, by the target itself, or remotely, by the initiator using HSSL command frames

Initiator configuration for stream channel:

- The two initiator start addresses are configured in ISSAx.
- The single block size is configured in the ISFC register.
- The current address being transferred is indicated in the read only (rh) ISCA register. The bit MFLAGS.IMB selects the corresponding ISSAx register. This bit is not modified by the hardware in single block mode, and is toggled by the hardware in the continuous mode. At the start of the transfer, the selected start address ISSAx is copied to the ISCA register

Target configuration for stream channel:

- The two target start addresses are configured in TSSAx
- The single block size is configured in the TSFC register
- The current address being transferred is indicated in the read only (rh) TSCA register. The bit MFLAGS.TMB selects the corresponding TSSAx register. This bit is not modified by the hardware in single block mode, and is toggled by the hardware in the continuous mode. At the start of the transfer, the selected start address TSSAx is copied to the TSCA register

Stream channel interrupts:

The HSSL stream transmitter generates one dedicated interrupt and several error events:

- Transmit block end (TBE) interrupt, generated after the acknowledgment of the last frame has been received
- Error events that can trigger a channel error interrupt: target, timeout, transaction tag, or unexpected error
- A global SRI or SPB error interrupt (check if the ISB bit is cleared by hardware)

The HSSL stream receiver generates one dedicated interrupt and several error events:

- Receive block end (RBE) interrupt event, generated after the last stream frame in a block has been written to the memory and the SRI or SPB master has received a confirmation (an acknowledgment, but not an error)
- Error event (target error that triggers an NACK frame)
- A SRI or SPB global error interrupt (check if the TSE bit is cleared by hardware)

As long as the ISB bit is low, the TXFIFO and the expect FIFO are cleared and kept in the empty state. As long as the TSE bit is low, the RXFIFO is cleared and kept in the empty state.

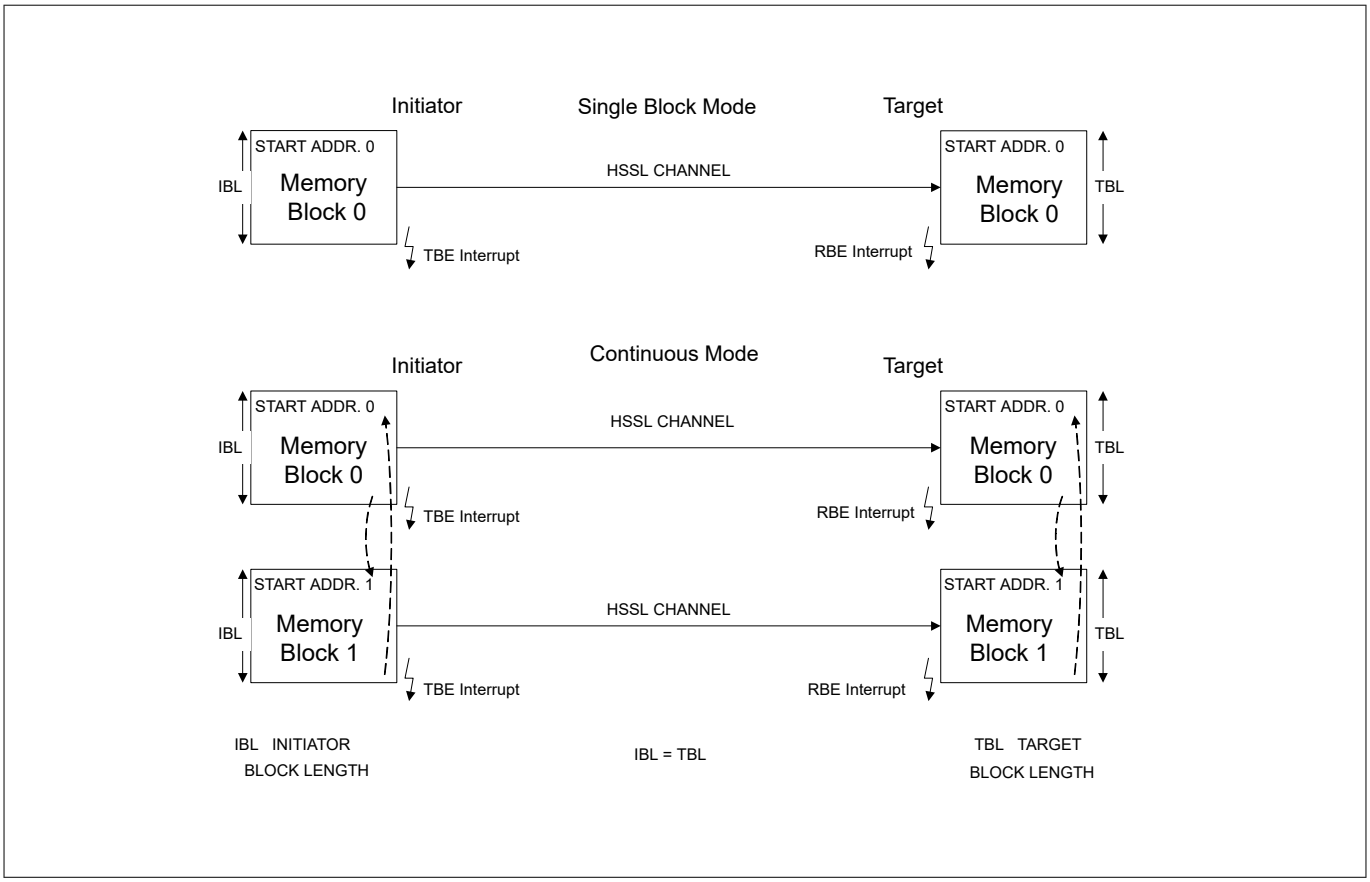


Figure 221 Stream modes of operation

17.4.3.8 HSSL reset

Both the HSCT and the HSSL modules must be reset together.

17.4.3.9 OCDS SRI or SPB master suspend

The SRI or SPB master access on the bus can be suspended by using the OCDS suspend request. When the master reaches idle state the suspend acknowledge signal is activated.

17.4.3.10 OCDS trigger sets

In order to support the debugging activities, the HSSL provides a set of internal signals to the on-chip debug system OCDS.

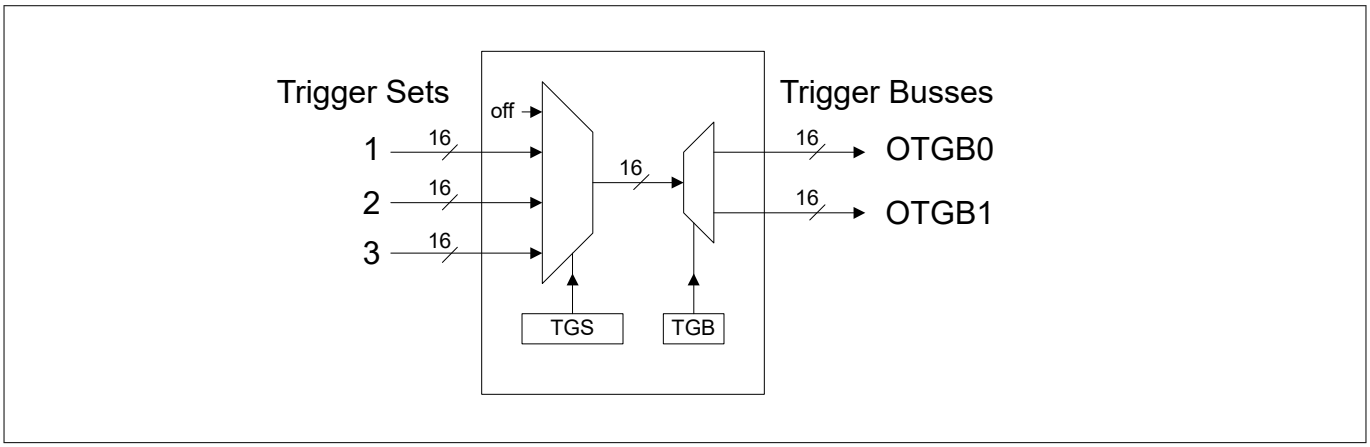


Figure 222 Overview of the trigger sets and busses

Configuration is done by using the bits OCS.TGS and TGB.

An edge on any signal, belonging to the selected set, going out on one of the two OTGB buses, triggers an action of the OCDS system. Only trigger set 1 and 2 are used and trigger set 3 is not used.

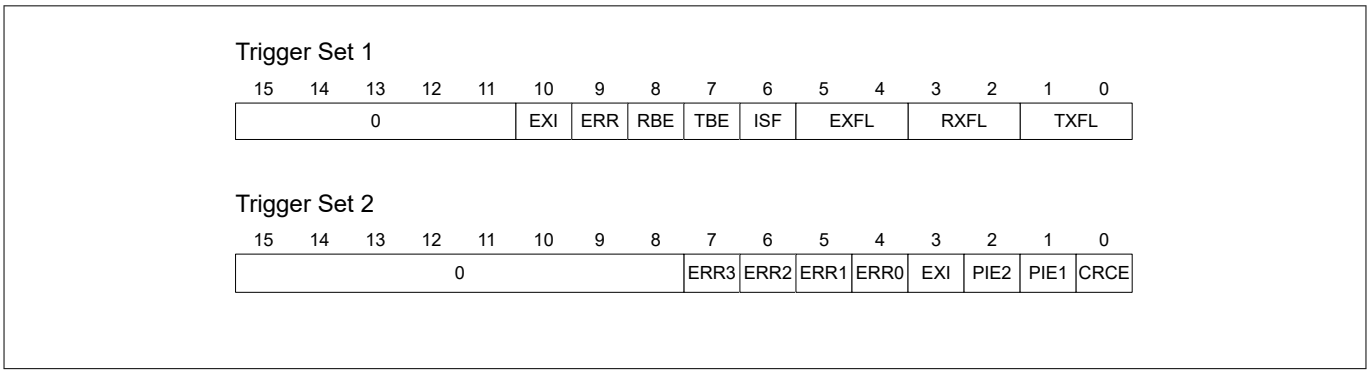


Figure 223 Overview of the trigger signal sets

Table 787 HSSL trigger sets

Name	Description
TS16_STR	Streaming channel trigger set (trigger set 1)
TS16_ERR	Error trigger set (trigger set 2)

The trigger sets 1 and 2 are used, the set 3 is not used (padded with 0).

Set 1, streaming channel debugging:

- 2 bits: TXFIFO filling level SFSFLAGS.TXFL
- 2 bits: RXFIFO filling level SFSFLAGS.RXFL
- 2 bits: Expect FIFO filling level SFSFLAGS.EXFL
- 1 bit: Initiator stream frame request SFSFLAGS.ISF
- 4 bits: streaming channel interrupt signals TBE, RBE, ERR, EXI, see the "Interrupts" section
- Others: not used

Set 2, timeout errors caused by non-specific errors debugging:

- 1 bit: CRC error MFLAGS .CRCE
- 1 bit: Phy Inconsistency error, inconsistent channel number code, MFLAGS.PIE1
- 1 bit: Phy Inconsistency error, inconsistent data length, MFLAGS.PIE2
- 5 bits: error interrupt signals 1 x EXI $1 \times \mathrm{EXI}$ and 4 x ERR $4 \times \mathrm{ERR}$. See the "Interrupts" section
- Others: not used

The signal lists are mapped to the trigger sets in the following way: First item from the list is mapped to the trigger signal 0, Second item in the list is mapped to the trigger signal 1, and so on upto 15.

Related information

Interrupts on page 2464

17.4.3.11 Access protection

The HSSL provides memory access protection in form of four memory access windows. It is the protection from external access to the address space of HSSL. Each window can be located anywhere within the address space, having an arbitrary size with the granularity of 256 bytes.

Each window defines a memory range where an access is allowed. All the four windows create a sort of access filter that protects the memory from external writing or reading. A window can be a 'read-only' (r), 'write-only'(w), or 'read and write' (rw).

Each window is defined with:

- Window start address. See register AWSTARTi
- Window end address. See register AWENDi
- Access rule: Read only, write only, or read and write. See register AR

If two access windows overlap, one read-only(r) and one write-only(w), the common address range is read and write(rw) accessible. If no access window overlaps with r, w, or rw window, the r, w, or rw window wins. No access window means the window is disabled.

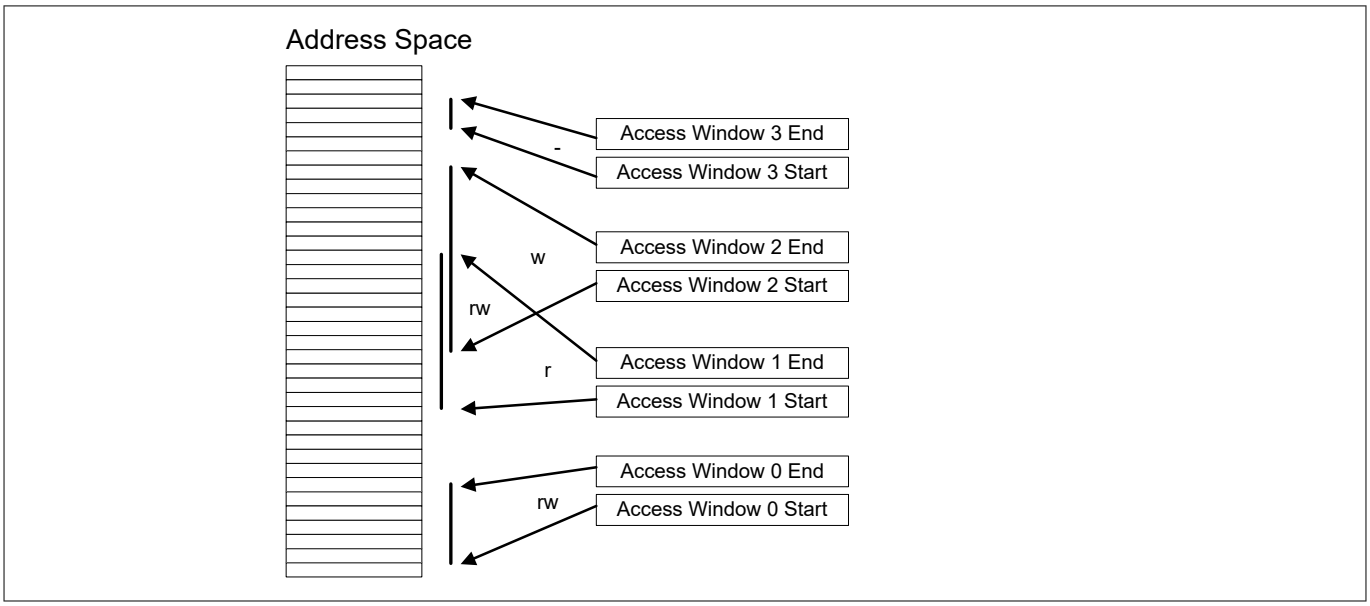


Figure 224 HSSL access windows

Normally, it is expected that the End Address is higher than or equal to the Start Address, otherwise the window is invalid and no commands can pass through the up-side-down range.

The AWSTARTi registers define the first address of the first 256-byte block of an address window.

The AWENDi registers define the first address of the last 256-byte block of an address window.

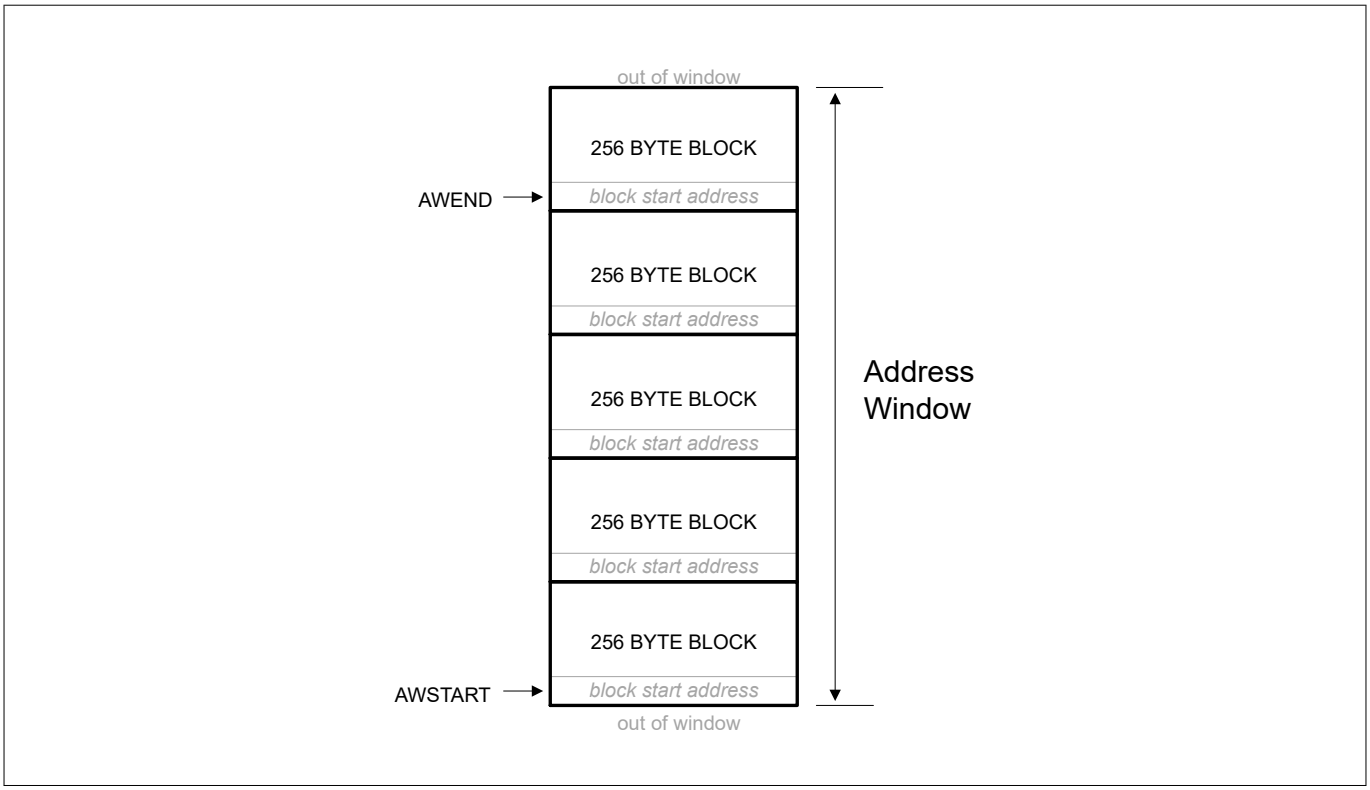


Figure 225 Access windows definition

These command frames are filtered based on the frame type:

- Register read
- Register write
- Stream Frame

These command always pass through:

- Read answer
- ACK
- NACK
- Trigger
- ID Request

17.4.3.11.1 Command frames access protection

If a HSSL target receives a read or write command frame with an address which passes through one of the windows, the command is executed and a response frame is sent back.

If a HSSL target receives a command frame with an address which does not pass through any windows, the command is not executed, a NACK response is sent back. The memory access violation flag MFLAGS.MAV is set, the access violating channel is captured in the bit field AR.MAVCH, and the EXI interrupt is triggered in the target.

17.4.3.11.2 Streaming access protection

Stream frames do not contain address information. The two distinct cases are, target side and initiator side.

Target side

At the target side, when the streaming starts, the initiator has already programmed the target addresses in the TSSAx registers by using write frames, which pass through and carry the stream address in their data field. Therefore, the memory filtering uses the current access address, visible in the TSCA register and if the current streaming address passes through the access protection filter, the read or write is executed, else a NACK frame is sent as a response.

If a HSSL target receives a stream frame and the current access address does not pass through any windows, the access is not executed, a NACK response is sent back. The MFLAGS.TSE bit will be cleared by hardware and the incoming frames will be ignored. The memory access violation flag MFLAGS.MAV is set, the access violating channel is captured in the bit-field AR.MAVCH, and the EXI interrupt is triggered in the target.

Initiator side

At the initiator side, the ISSAx registers, which are externally accessible, define the memory range to be transmitted. Therefore the access filtering at the initiator side covers the streaming read accesses by using the current access address, visible in the ISCA register.

At the initiator side, access violation stops the streaming, and the MFLAGS.ISB bit is cleared by hardware. The memory access violation flag Miscellaneous Flags Register.MAV is set, the access violating channel is captured in the bit-field AR.MAVCH, and the EXI interrupt is triggered.

17.4.3.11.3 Public and private registers

The HSSL module contains two types of registers, public and private.

- Public registers, which can be read and written by the HSSL module
- Private registers, which can only be read by the HSSL module

This behavior of the HSSL module registers is implemented by using the User Mode (U) and Supervisor Mode (SV) for writing registers. The private registers are writable in SV mode, the public registers are writable in both U and SV mode.

The memory access protection registers AWSTARTi, Access Window End Register i and AR including the TIDADD register are all writable only in Supervisor Mode, and are consequently private. The HSSL itself makes only User Mode accesses, so that it can not write these registers.

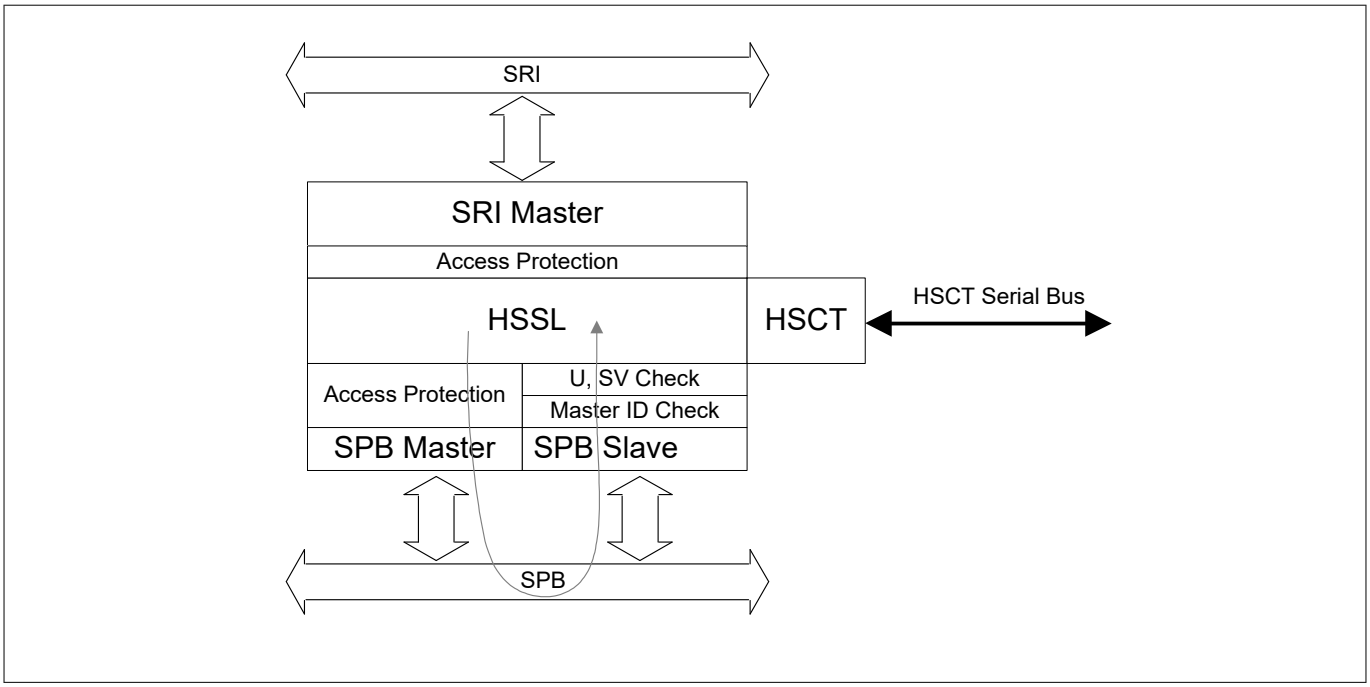


Figure 226 HSSL access to its own registers

17.4.3.12 Multi-slave operation

In multi slave use-cases, one HSSL master can communicate with up to three slaves. From the HSCT module's point of view, the master generates the reference clock and selects the active counterpart for the connection. All the slaves receive the reference clock, but the master selects one slave at a time.

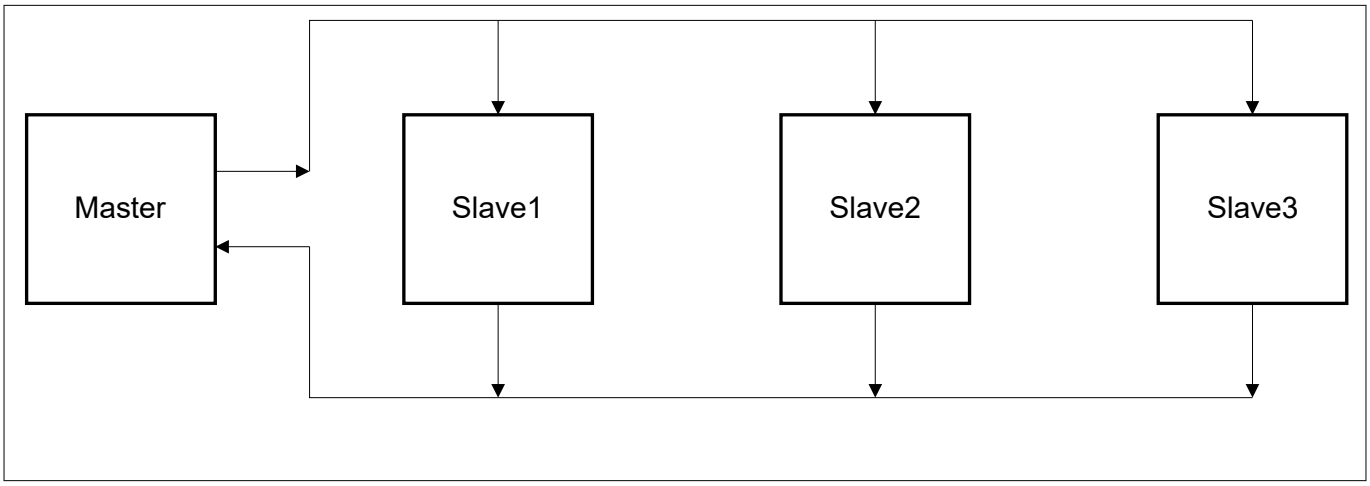


Figure 227 Three slave HSSL connection

The communication is connection based. At any point in time, there can be only one active connection (or zero) between the master and one slave. There are no broadcast or multi-cast messages on the HSSL level. There is a defined procedure for terminating a connection with one slave and activating a connection with another slave. During an active connection both master and slave can be initiators and targets.

After power-on reset, the transmit outputs of the slaves and the transmit output of the master are high impedance. Up to three slaves can be enabled for reception by the application software.

17.4.3.12.1 Slave tag and slave control

The multi-slave operation uses the identical set of command and stream frames as the standard HSSL two-device operation. The slaves are selected with a two bit slave tag in the HSSL header.

These bits are used to filter the frames which are relevant for a slave by comparing them with the two bit slave tag of the respective slave. See MSCR.SLAVETAG.

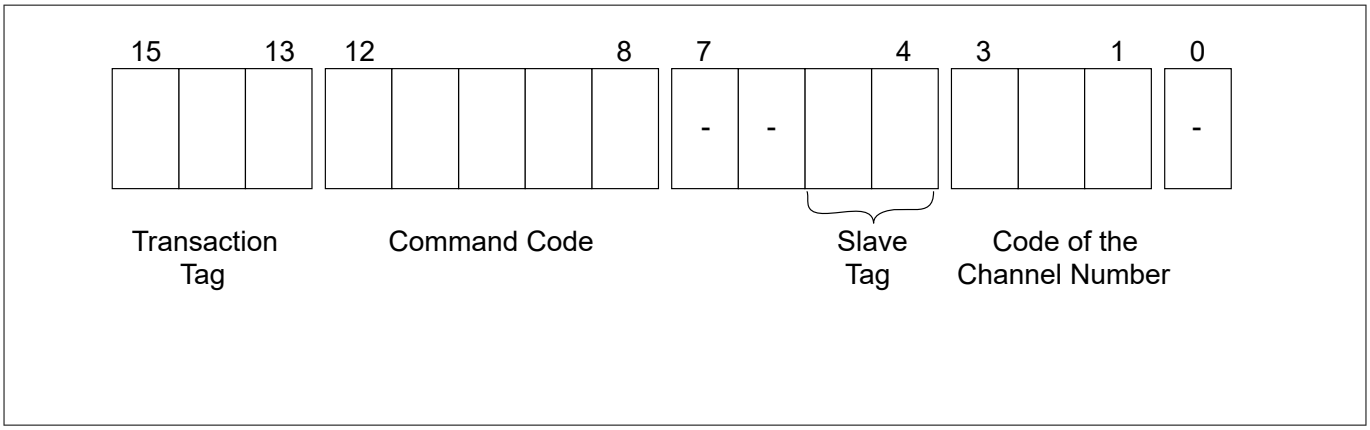


Figure 228 HSSL header with slave tag

17.4.3.12.2 Slave tag frame filter and translator

The filter is located after the CRC block. In receive direction, if there is a match, the frame is passed through and handled in a standard HSSL two-device-connection way. If there is no match, the frames are discarded.

In transmit direction, the headers of transmitted frames get the appropriate tag inserted. In case of a CRC error in a frame sent by the master, all slaves may detect the CRC error. The CRC checker discards the frames with CRC error. Even the addressed slave does not get the frame with a CRC error delivered. Frames with PIE1 and PIE2 error are also discarded. The corresponding error interrupts will be triggered if enabled.

The slaves have their tags fixed in the initialization phase and are never changed. The software of the master changes its slave tag when establishing connection with a different slave. Initially the HSCT link must be established, then the master must have its MSCR.SLAVETAG configured before starting to send HSSL commands or stream frames to the appropriate slave.

The MSCR.EN and MSCR.SLAVETAG bit fields are propagated to the HSCT module in order to ensure consistent operation of the whole communication channel.

If the multi-slave operation is disabled, by setting MSCR.EN=0, then neither the injection of the slave tag in transmit direction nor the filtering of the slave frames in the receive direction are performed. The header remains unchanged and all the incoming frames pass through.

It is not intended to set the MSCR.ITXSTOP bit in the master.

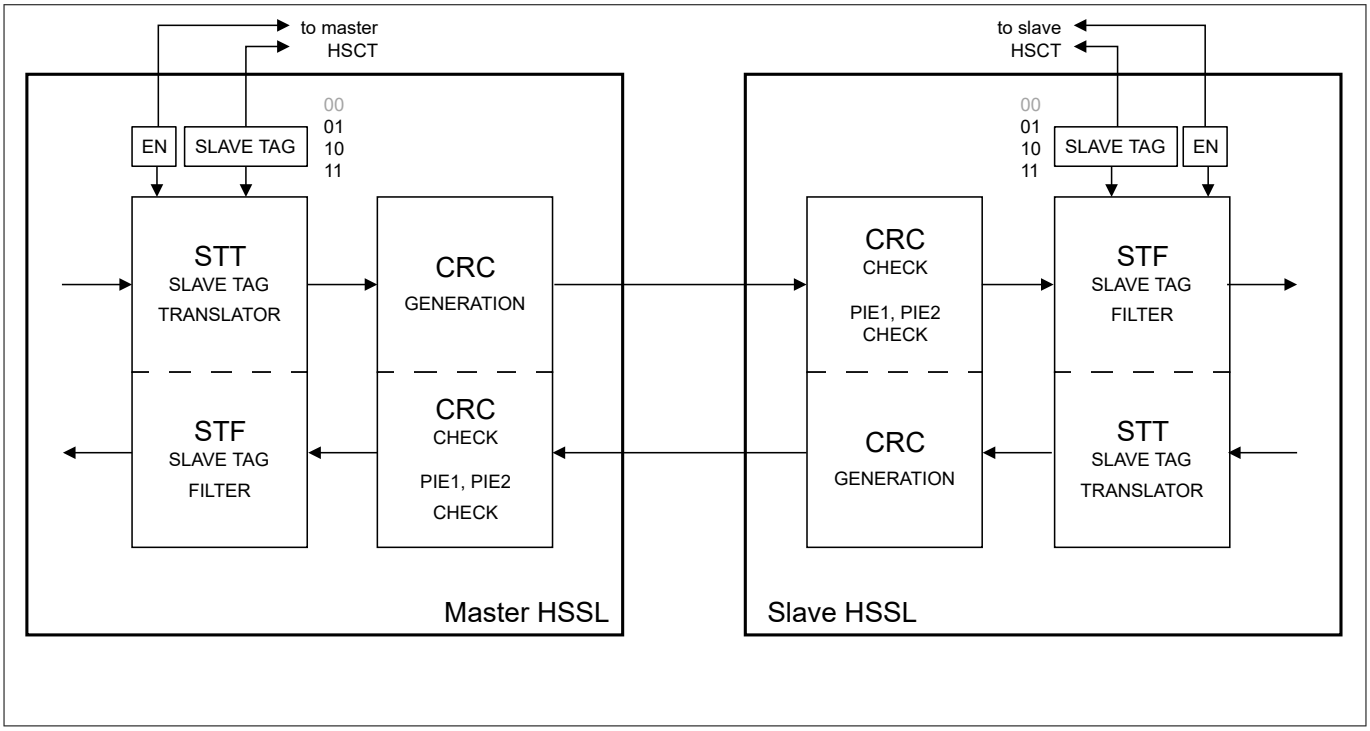


Figure 229 Slave frame filter

17.4.3.12.3 Activating a slave

A slave is activated in two steps:

- The master enables the LVDS transmitter pads of one slave by using a slave specific HSCT command. The Tx enable HSCT commands for the slaves 1, 2, and 3 have payloads 51_H, 52_H, and 53_H respectively
- The second step is either soft or hard HSSL activation:

Soft activation

The master notifies the slave by using standard HSSL commands, according to a predefined procedure. For example, the master can issue a write command to a certain location, send an interrupt trigger command frame, and a slave with a CPU can start transmitting using the HSSL module

Hard activation

The master clears the MSCR.ITXSTOP bit by using a standard HSSL 16-bit write command frame, assuming an appropriate address window for the HSSL registers has already been set up. In the next step, the master notifies the slave, for example with a trigger command, or enables the operation of the slave by setting appropriate bits. The slave continues the operation from the point its initiator transmission has been stopped

17.4.3.12.4 Deactivating a slave

The Master can deactivate a slave in two ways: soft and hard.

Soft deactivation

Soft deactivation procedure uses a software handshake procedure to bring the slave in a state where all ongoing slave initiator transactions has been completed, including streams, no new slave transactions are initiated, and the slave notifies the master that the safe deactivation state has been reached. Subsequently, the master disables the LVDS pad.

The soft deactivation sequence consists of:

- Soft HSSL deactivation:Using a software handshake with standard HSSL commands, such as writing to a memory location and triggering interrupts
- Switching off the LVDS Tx pads with a non-specific slave (broadcast) HSCT command with a payload 32_H

Hard deactivation

Hard deactivation is intended to provide faster deactivation with bounded reaction time, primarily for slaves without CPU and use-cases without streaming.

The hard deactivation sequence consists of:

- Hard HSSL deactivation
  - The master sets the MSCR.ITXSTOP bit by using a standard HSSL 16-bit write command frame. An appropriate address window for the HSSL registers must have been already set up
  - The master must assure that slave initiator commands have been processed by itself and that the acknowledge frames have been sent to the slave. For example, the master polls the slave E (expect) flags by using standard HSSL 16-bit read commands until it reads back 0
  - The master must assure that its command frames have been processed and acknowledged by the slave. For example, the master can poll its own expect flags until it reads 0
- Switching off the LVDS Tx pads with a non-specific slave (broadcast) HSCT command with a payload 32_H

The slave HSSL module initiator transmit path remains inactive and later, after being re-activated by the master, continues the operation from the point where it has been stopped. The master is not allowed to send commands to a deactivated slave.

17.4.3.12.5 MSCR HSSL to HSCT connections

The HSSL module provides the MSCR.EN and MSCR.SLAVETAG bit fields at the module border as outputs. These signals are connected to the HSCT module and provide a single point to configure the behavior of both HSSL and HSCT regarding multi-slave behavior.

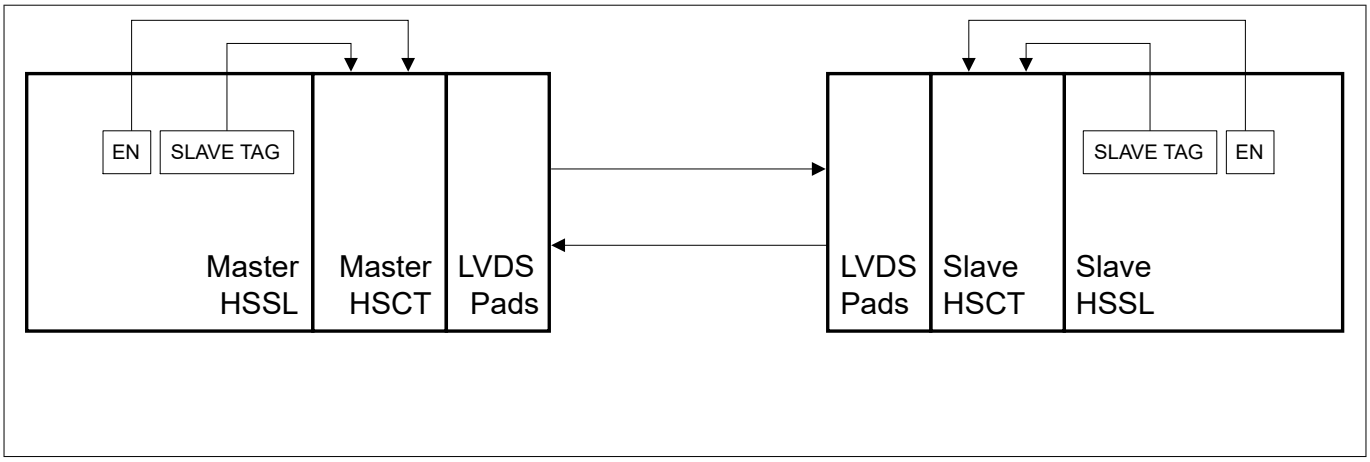


Figure 230 HSSL to HSCT connections

17.5 High Speed Communication Tunnel (HSCT)

A description of the High Speed Communication Tunnel interface for the microcontroller high speed data communication, including the physical layer and the control (link) layer.

17.5.1 Feature list

HSCT common interface basic features

- TX-link: transmit sample information from the master interface to the slave interface
- RX-link: receive sample information from the slave interface to the master interface
- System clock provided by the master interface (crystal owner) to the slave interface
- Fexible configuration of operating modes:
  - TX-link baudrate support (10 MHz/20 MHz SysClk): 5 MBaud (low speed) and 320 MBaud (high speed)
  - TX-link baudrate support (25 MHz SysClk):6.25 MBaud (low speed) and 320 MBaud (high speed)
  - RX-link data transfer speed based on f_{SYSCLK} = 20 MHz: 5 MBaud (low speed) 20 MBaud (medium speed) 320 MBaud (high speed)
  - RX-link data transfer speed based on 10 MHz SysClk: 5 MBaud (low speed) 320 MBaud (high speed)
  - RX-link baudrate support (25 MHz SysClk): 6.25 MBaud (low speed) and 320 MBaud (high speed)
- High speed data transfer rate can be reduced by implemented dividers (2, 4, 8, 16)
- The interface is based on IEEE 1596.3 LVDS IOs (Low Voltage differential Signaling Input Outputs) and provides driver swing amplitude configuration

Single slave connection features

- Control information on TX link and RX link
- Clear To Send(CTS) indication in both directions
- Unsolicited status in both directions
- Regular data transfer in both directions based on data channels

Multi-slave (maximum of 3) connection features and restrictions

- Control information on TX link only
- Multi-drop configuration (Master is addressing all slaves connected to the trace line)
- Clear to send indication not supported
- Unsolicited status not supported
- Regular data transfer in both directions based on data channel only (derived by Higher layer protocol)
- Multiple sender active or driving the RX link at the same time is prohibited (ensured by system master software)
- Slave communication channel to a selected slave by higher layer software; Only one slave can be addressed and communication must be finished before the next slave can be addressed
- Slaves driving the LVDS trace line simultaneously are prohibited
- Connection of 2 slaves to the LVDS trace line limits the maximum baud rate to 160 MBaud
- Connection of 3 slaves to the LVDS trace line limits the maximum baud rate to 80 MBaud
- Discrete termination resistor for differential receiver trace line impedance matching is required on PCB with 100 Ohm +/-5% (check PCB layout guidelines from Infineon); internal resistor is not supported

17.5.2 Functional overview

The HSCT employs an interface for inter-chip communication between a Master interface and a Slave interface. The interface is based on LVDS IO technology and developed to function as a Master or a Slave interface. During

the initialization phase the role of the Interface (Master or Slave) must to be defined. It is not intended to change the system role after initialization phase.

The interface is a full-duplex interface for TX and RX data transfer. The data transfer speed can be configured to function in low or high speed and allows to configure in each transfer direction an independent transfer speed in order to reduce power and EMI, based on transfer speed requirements.

The Master interface is defined to own the crystal and provides the clock (HSCT SysClk) to the Slave interface. The HSCT Master/Slave interface allows from system point of view one crystal only. In this scenario the HSCT SysClk input at Slave Side is used as reference clock for other PLL's like SysPLL. The interface reset is derived from System reset and supports common reset behavior, including module reset. The name of the SysClk signal driven by the Master is SYSCLK_OUT. The name of the SysClk signal received by the Slave is SYSCLK_IN.

In the next figure a system level view of the interface and interface connection is shown:

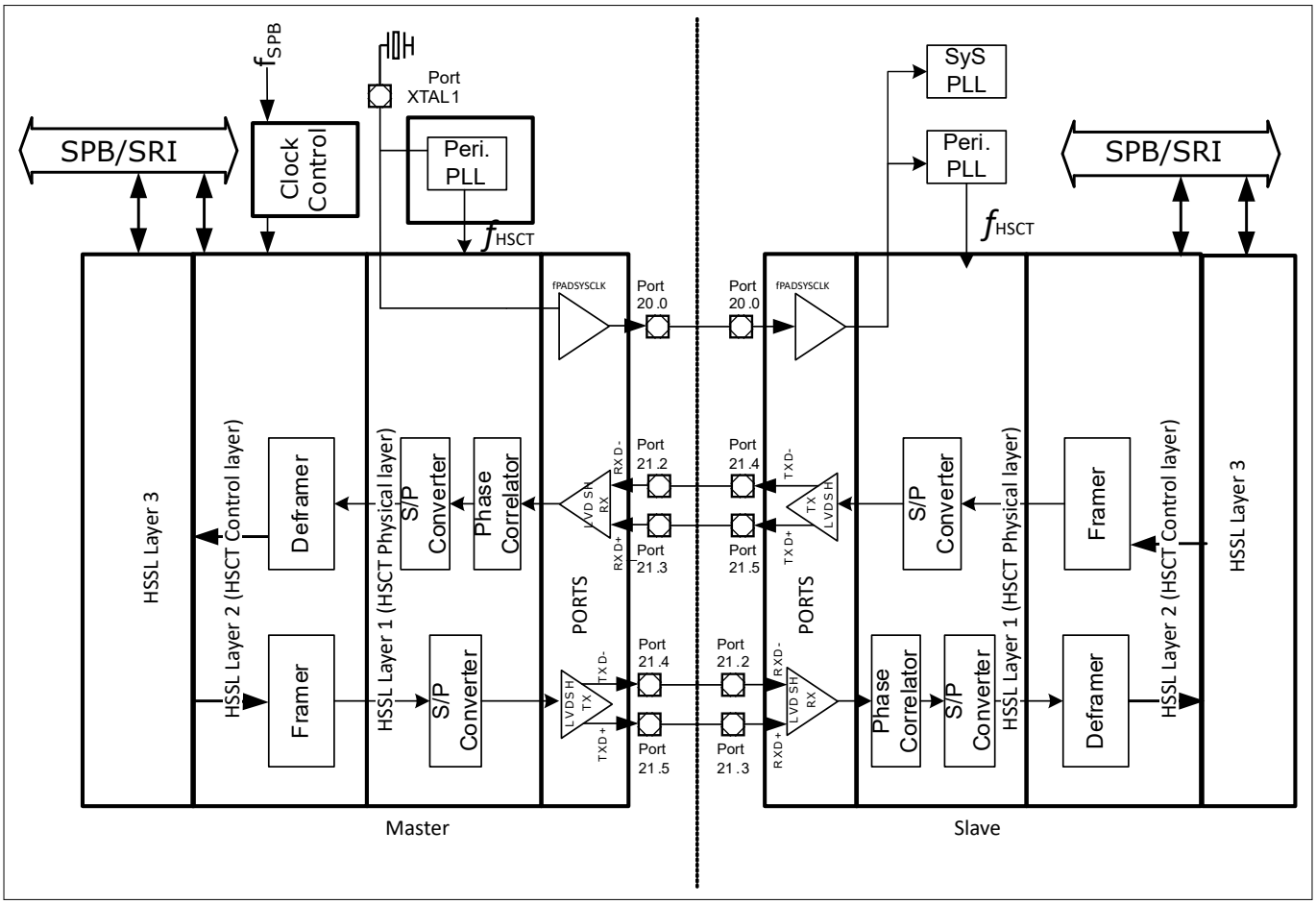


Figure 231 HSCT block diagram

17.5.3 Functional description

This chapter explains the detailed functional behavior on data-link and physical layer tasks of the HSCT sub-module.

17.5.3.1 Introduction

The initial HSCT communication is always established from master interface. The LVDS IOs are based on standard IO technology and follow the IEEE 1596.3 specification with minor limitations (see the corresponding datasheet for further information). The LVDS IO common voltage is defined to be a 1.2 V. Compared to the IEEE 1596.3 specification, the differential voltage level can be reduced through a configuration option to reduce EMI.

The intention is to implement the interface either as a master interface or as a slave interface. So the interface is able to handle both roles (master or slave). The description and definition of both the roles are as follows:

- TX line or TxData link (TX_DAT, TX_DATX) defines the data transfer from the master interface to the slave interface
- RX line or RxData link (RX_DAT, RX_DATX) defines the data transfer from the slave interface to the master interface
- Transmit direction, outbound data (or transmission) describes sending out data independent of the role of the interface
- Receive direction, inbound data (or receiving) describes receiving data independent of the role of the interface

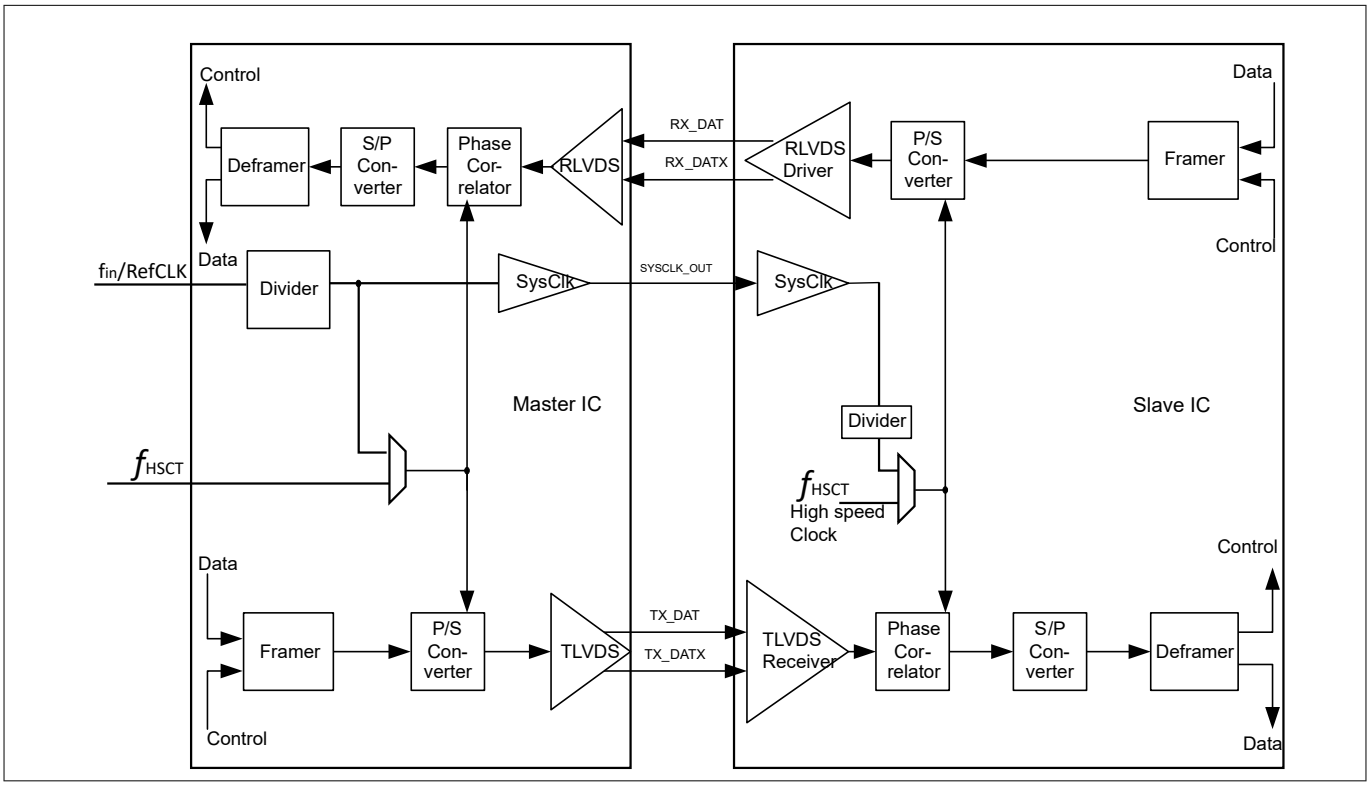


Figure 232 Internal block structure

17.5.3.2 Physical layer

This chapter explains in detail the physical layer composition of the HSCT sub-module.

17.5.3.2.1 RX/TX data communication

During initialization the associated TxData link starts in low-speed mode, with the high-speed clock not enabled or the clock generators turned off, and the RxData Link disabled. An interface control sequence, initiated by software, allows for switching the interface in high-speed mode.

The data interface can transmit with a maximum Baud rate of 320 Mbit/s in the RX and TX direction (full-duplex). The data is recovered from RX data stream (RX_DAT, RX_DATX) in the master interface and from TX data stream (TX_DAT,TX_DATX) in the slave interface. The table below shows the interconnects of the differential data lines for control and data transfers between the master and slave interface, and also shows the limitations for multi-slave connections.

Table 788 Physical interconnects

Direction	Name	Description
Transmit (single slave)	TX_DAT, TX_DATX	Channel TX data and control information at a maximum rate of 320 Mbit/s from the master interface to the slave
Receive (single slave)	RX_DAT, RX_DATX	Channel RX data and control information at a maximum rate of 320 Mbit/s from the slave interface to the master
Transmit (two slaves)	TX_DAT, TX_DATX	Channel TX data and control information at a maximum rate of 160 Mbit/s from the master interface to the slave
Receive (two slaves)	RX_DAT, RX_DATX	Channel RX data and control information at a maximum rate of 160 Mbit/s from the slave interface to the master
Transmit (three slaves)	TX_DAT, TX_DATX	Channel TX data and control information at a maximum rate of 80 Mbit/s from the master interface to the slave
Receive (three slaves)	RX_DAT, RX_DATX	Channel RX data and control information at a maximum rate of 80 Mbit/s from the slave interface to the master

The TX link differential data with the configured baud rate is transmitted from the master interface to the slave. Proper phase selection is processed at the receiver side in five different phases.

The RX link differential data is transferred from the slave interface to the master interface. A proper phase selection is required for data recovery at the master receiver path in five different phases derived from the clock generator (Peripheral PLL). The slave interface reference clock must be the HSCT SysClk, driven by the interface master. The slave must receive the SysClk from the master interface to enable reliable data transfer. A clock at the slave side derived from a crystal will lead to improper data communication, as frequency drift variation is uncontrollable and larger than if derived from a single clock source. The slave interface maximum frequency (320 MHz data clock) is generated internally from the HSCT SysClk.

The HSCT SysClk at the slave interface is also the reference clock for other PLLs on the chip.

***Note**: A local crystal at the slave interface taken as reference clock for the interface PLL is not allowed, as the data recovery based on clock phase shift works with a common reference clock source only!*

The RxDAT link (slave TX) enters shutdown mode by interface control logical channel command and exits shutdown mode again through a control command. The slave transmission line is disabled by default and must be enabled by the master. The master interface control commands are used to set the link in operation. TX or RX enable/disable shall not be changed during an active frame transmission.

Payload content is transferred most significant bit first. For example, if a 32-bit payload contains a message derived from a 32-bit register at TX side, bit31 of the register (the MSb) shall be the first bit of payload transmitted and bit0 (LSb) shall be the last. Larger payloads should be built up according to the same principle.

17.5.3.2.2 Differential signaling principle based on LVDS

The voltage at the termination resistor R_T of the receiver is either positive or negative, which depends on the direction of the current flowing through it. This is determined by the state of the switches at the transmit stage. The voltage V_{OD} should be evaluated by a comparator with hysteresis.

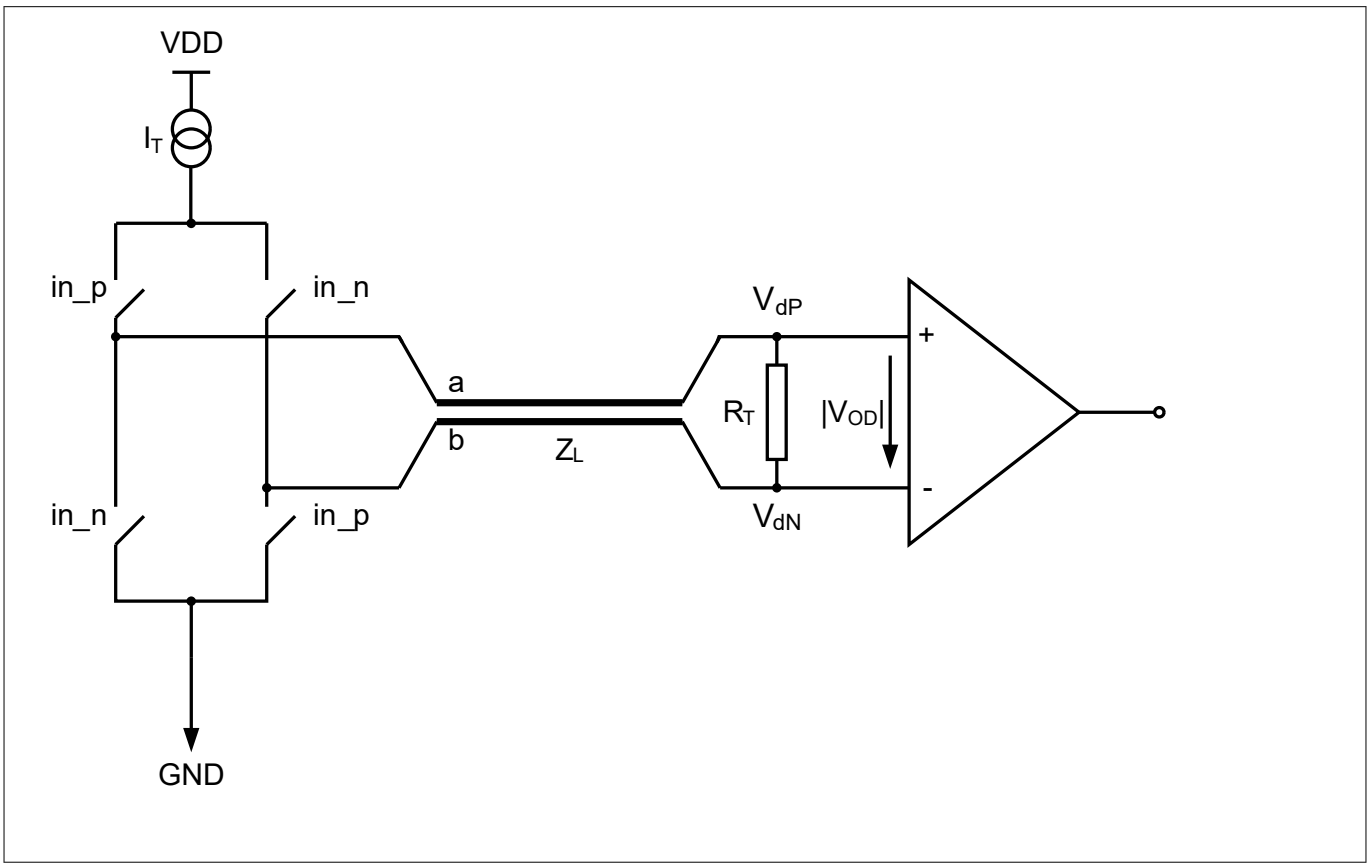


Figure 233 Pad driver and pad receive

The HSCT interface does not define any specific test for an error-free transmission, although the signal quality of course suffers from attenuation and noise, and particularly depends on proper wave propagation. Therefore the interconnect lines need to have a characteristic wave impedance Z_L equal to 100 Ω differential, corresponding to the termination resistor R_T at the receiver side.

***Note**: Due to the nature of wave propagation, a termination other than the characteristic line impedance leads to inevitable signal degradations. Therefore any operation mode other than 100 Ω termination is only supported from a functional point of view (that is the corresponding electrical configuration is possible) without any guarantee of an error-free transmission.*

***Note**: The interface receiver provides an input termination impedance of 100 Ω +/- 20% at the IO. If a more precise termination is required the design offers to disable the internal termination impedance and allow an external termination on the PCB. In a multi-slave approach the internal termination impedance must be disabled and a resistor with 100 Ω +/- 5% must be placed on the PCB instead, as close as possible to the receiver IO (exact placement can be identified based on the Infineon PCB layout guidelines).*

17 High Speed Serial Link (HSSL)

Some AURIX™ TC4xx devices may not include an internal termination resistor in the LVDS RX I/O cells. Refer to device specific chapter and Ports functional block content for related information.

The differential output voltage V_{OD} is defined as the difference of the voltages V_{dP} and V_{dN} at the receiver input pins.

$$V_{OD} = V_{dP} - V_{dN} \quad (1)$$

The common-mode voltage V_{OS} is defined as the arithmetic mean value of the voltages at the receiver input pins:

$$V_{0S} = \frac{V_{dP} + V_{dN}}{2} \quad (2)$$

V_{OD}, V_{OS}, and the corresponding minimum differential eye opening V_{eye} considering LVDS transmitter jitter are shown in the following two figures for ideal signals.

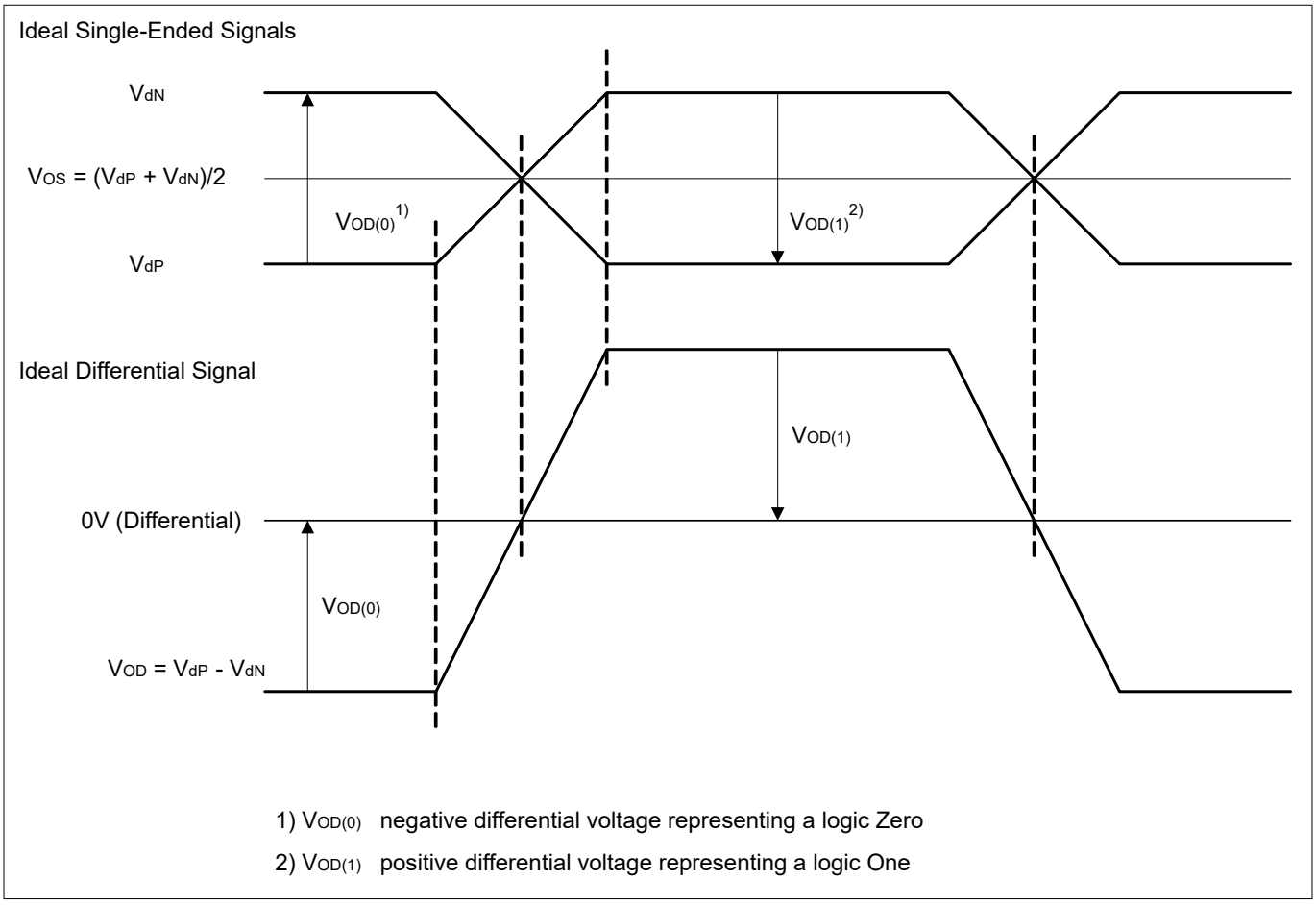


Figure 234 Ideal single-ended and ideal differential signals

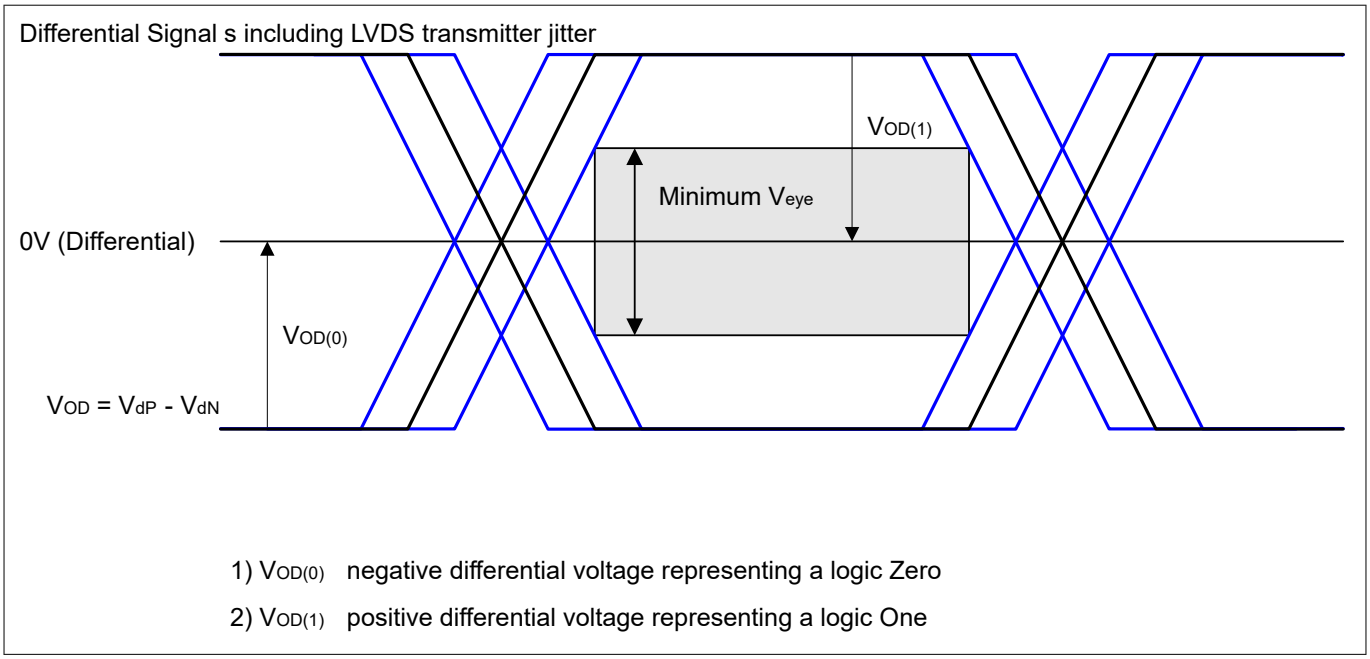


Figure 235 Differential eye opening

LVDS pads in multi-slave mode

In multi-slave mode, the procedure of changing the active slave is:

- Disabling the currently active slave LVDS transmitter
- Waiting for the LVDS pads to stabilize into Hi-Z state
- Enabling the LVDS transmitter of the next slave
- Waiting for the LVDS bus to stabilize into active state

After this, the first HSSL command can be sent to the newly activated slave.

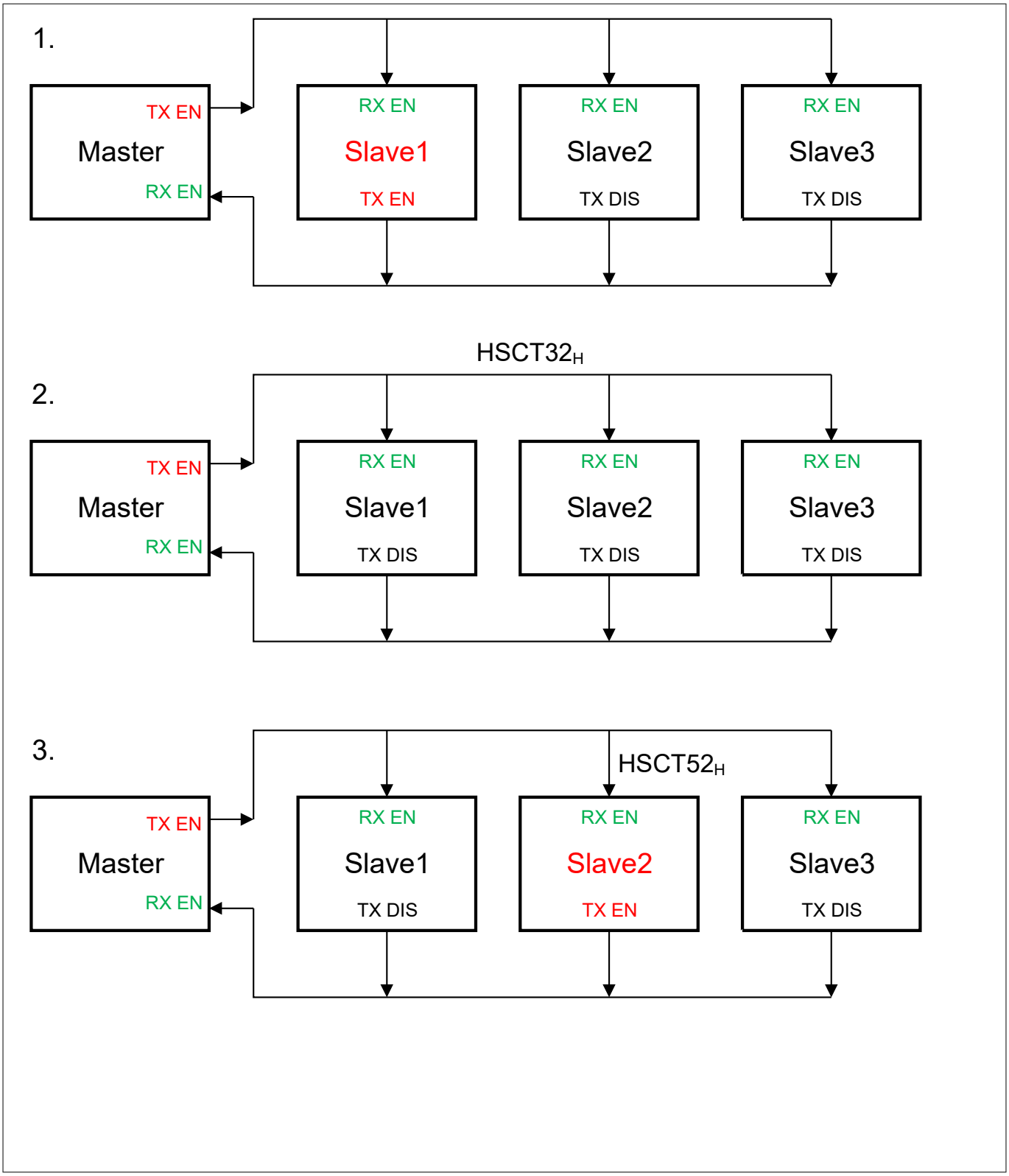


Figure 236 Changing the active slave

In step 2 of Figure 48, the output bus of the slaves is not driven by any slave and all LVDS Tx drivers are disabled. The differential voltage is equal to zero, and the offset voltage can take any value between the ground and power supply level.

The enable and disable time of the LVDS pads shown in the figure below are defined in the datasheet of the corresponding device as parameters t_{tx} (LVDS TX) and t_{trxi} (LVDS RX) $t_{LVDSDIS}$.

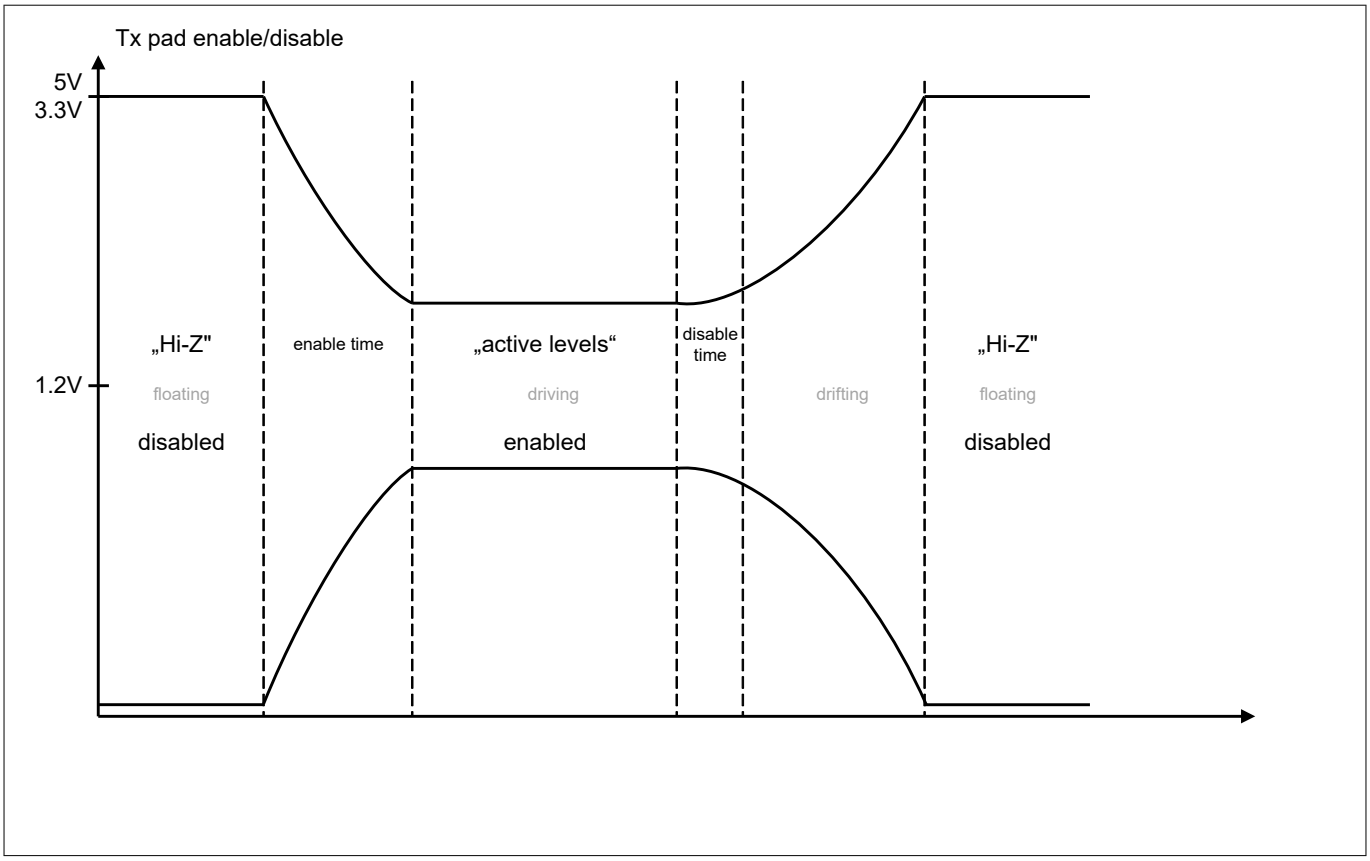


Figure 237 Voltage levels on the slave driven bus

17.5.3.2.3 Electrical characteristics based on LVDS for reduced link trace length

The characteristics are based on the LVDS standard IEEE P1596.3-1995 with some limitations as defined in the Datasheet. It is adapted for low power consumption and lower supply voltages for short ranges. It is defined that the interface supply voltages meet the common mode voltage for the differential signaling based on 1.2 V.

Operating modes

The HSCT interface comprises the following operating modes:

- Terminated Mode: The data transmitter requires a 100 Ω external termination between the pins TX_DAT and TX_DATX. The termination is given by a 100 Ω resistor at the receiver input, of the opposite LVDS device, for matching the differential characteristic impedance of the line with the impedance of the receiver. PCB design has to follow guidelines on differential signal routing. The PCB placed termination resistor can be chosen very precisely. Through configuration option, an 100 Ω LVDS receiver internal resistor can be enabled with a +/- 20% tolerance and shall not be enabled for a multi-slave scenario (PCB placed and +/- 5% tolerance according to IFX PCB guidelines)
- Power up: The line receiver is always enabled after power up and SysClk is active. The data decoder behind the receiver is not enabled at this point. It is enabled when the first transition on the data lines is detected by the line receiver (8 zeros at configured Baud rate). The line driver must be enabled on the Master side through software control. On the Slave side, the line driver must be enabled by sending interface message control commands from the Master. After power-up, the link starts in low speed mode. After line driver

enable, some time is required to power up the LVDS transmitter line. The data link in RX and TX direction is ready to transmit and receive data after the defined setup time

- Power down: When any interface driver supply voltage is removed, the driver shall provide a weak pull-down to 0 V to ensure that the signals are held in a well-defined state. It is a requirement to be free of cross current and bias current during power-down. This can be configured in the LVDS port control.

***Note**: The line driver in the Master interface shall be turned on before the line driver in the Slave interface is activated, that is before sending the transmitter activation command to the Slave interface, in order to avoid receiving undefined data in the Slave interface.*

Due to the strong dependency on link length and layout, as well as the susceptibility to noise and interference, it is recommended, to have the termination available on the PCB:

- Receiver internal termination mode not active (PCB termination): Can be programmed in register Init. The data transmitter is operated without a termination inside the differential receiver pair
- Receiver internal termination mode active: allows to have a 100 Ω resistor internally with a +/-20% tolerance (prohibited use in multi-slave scenario)

***Note**: Refer to device specific chapter to check the availability of the internal termination resistor in this particular device. In the absence of explicit reference, consider that it is available for usage.*

17.5.3.2.4 Data rates

The Slave interface shall be ready to receive frames whenever the HSCT SysClk is running. The link by default is set to low-speed mode after reset. Subsequent speed and mode changes shall be commanded by the Master interface.

Interface data rate change on both the TxData and the RxData link shall be possible without losing data. It is highly recommended to keep the link quiet during the time the interface transfer speed is changed. Changing transfer data rate on an active link does not guarantee stable data transfer.

The HSCT module provides three different data speed modes

- Low speed mode
- Medium speed mode (only available on RX link)
- High speed mode

Additionally, the HSCT module provides different data speed modes, derived from the crystal frequency. Taking the Automotive common crystal frequency of 20 MHz the following data rates can be produced based on the available multiplication factors.

SysClk; if RefClk frequency = 20 MHz

- 5 MHz = RefClk frequency/4 (low speed)
- 20 MHz = RefClk frequency (medium speed)
- 320 MHz = RefClk frequency * 16 (high speed)

Low speed mode and medium speed mode frequencies are generated directly from the refclk frequency. High speed mode frequency requires a PLL, which takes the crystal clock as reference clock. The following figure shows a frequency Scheme based on 20 MHz (refclk)crystal clock / SysClk as an example.

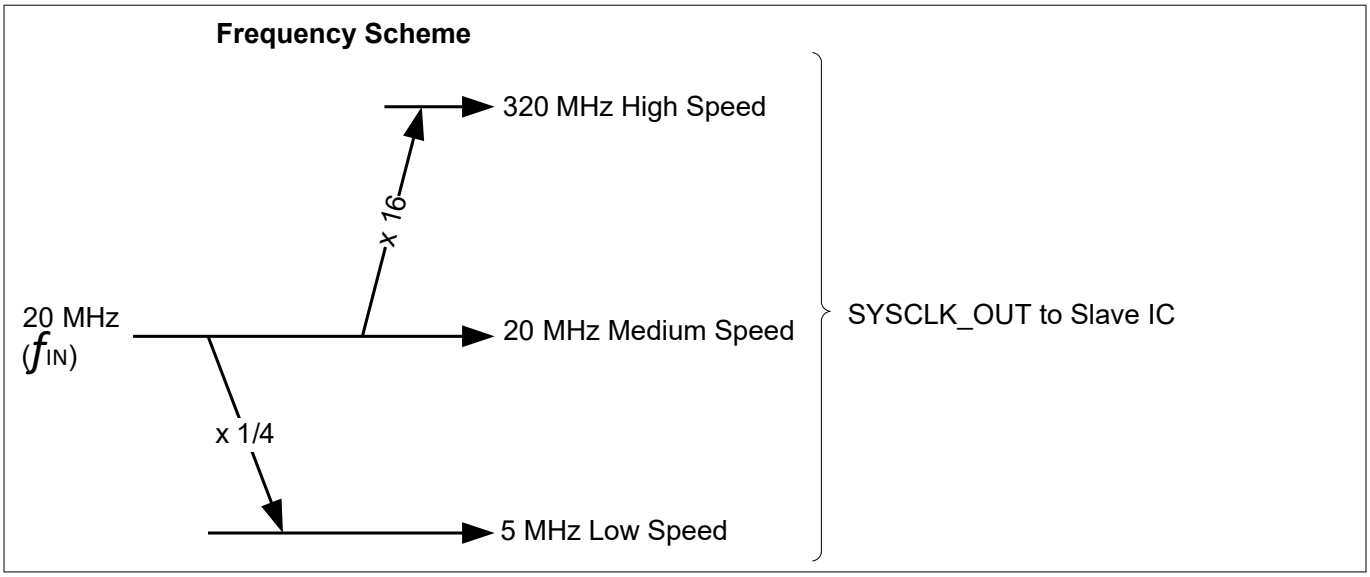


Figure 238 Frequency scheme

Connecting a 20 MHz or a 40 MHz crystal to the system, allows to generate the automotive frequency scheme.

The following table shows the data transfer direction and the available automotive transfer speed based on crystal clock frequency.

Table 789 Data rates

Direction	Data rate	Operating mode
Master interface to Slave interface	5 Mbit/s	“low speed mode” for startup/fallback
Master interface to Slave interface	320 Mbit/s	“high speed mode” (Single Slave only); Multi Slave divide
Slave interface to Master interface	5 Mbit/s	“low speed mode” for startup/fallback
Slave interface to Master interface	20 Mbit/s	“medium speed mode” ***Note**: Only available if reference clock SysClk is at 20 MHz (Crystal)*
Slave interface to Master interface	320 Mbit/s	“high speed mode” (Single Slave only); Multi Slave divide

PLL configuration and baud rates

The PLL supports different speed modes and frequency modes, which will be described in this section. Each mode and transfer direction can be configured separately.

Starting from a reference clock(REFCLK) selection in the CCU it is possible to configure a REFCLK divider (Init.SRCF) for the internal clock switches and correlator circuit as well as configure the SYSCLK divider (Init.SSCF), which divides the SysClk provided by the Master interface. The REFCLK DIV generated PHY_CLK is considered in low speed and medium speed mode. During configuration of the reference clock path the role of the interface (Master or Slave) needs to be considered.

The HSCT as Master Interface requires to enable the crystal(XTAL) path. The preferred XTAL frequency for HSCT is 20 MHz. Based on the XTAL frequency a PHY_CLK of 20 MHz shall be configured using the REFCLK DIV. In addition the Master Interface provides the SysClk having f_{SYSCLK}, which is configured using SYSCLK DIV based on the reference clock frequency. Recommended SysClk out frequency is 20 MHz

The HSCT as Slave Interface requires to enable the SYSCLK PAD path. The preferred SysCLK frequency for HSCT is 20 MHz. Based on the SysClk frequency the PHY_CLK shall be configured using REFCLK DIV. Also in this case 20 MHz is recommended. A 10 MHz PHY_CLK leads to not availability of medium speed (20 MBaud).

The data path baud rate can be configured independent per link direction using the Init.TXHD and Init.RXHD.

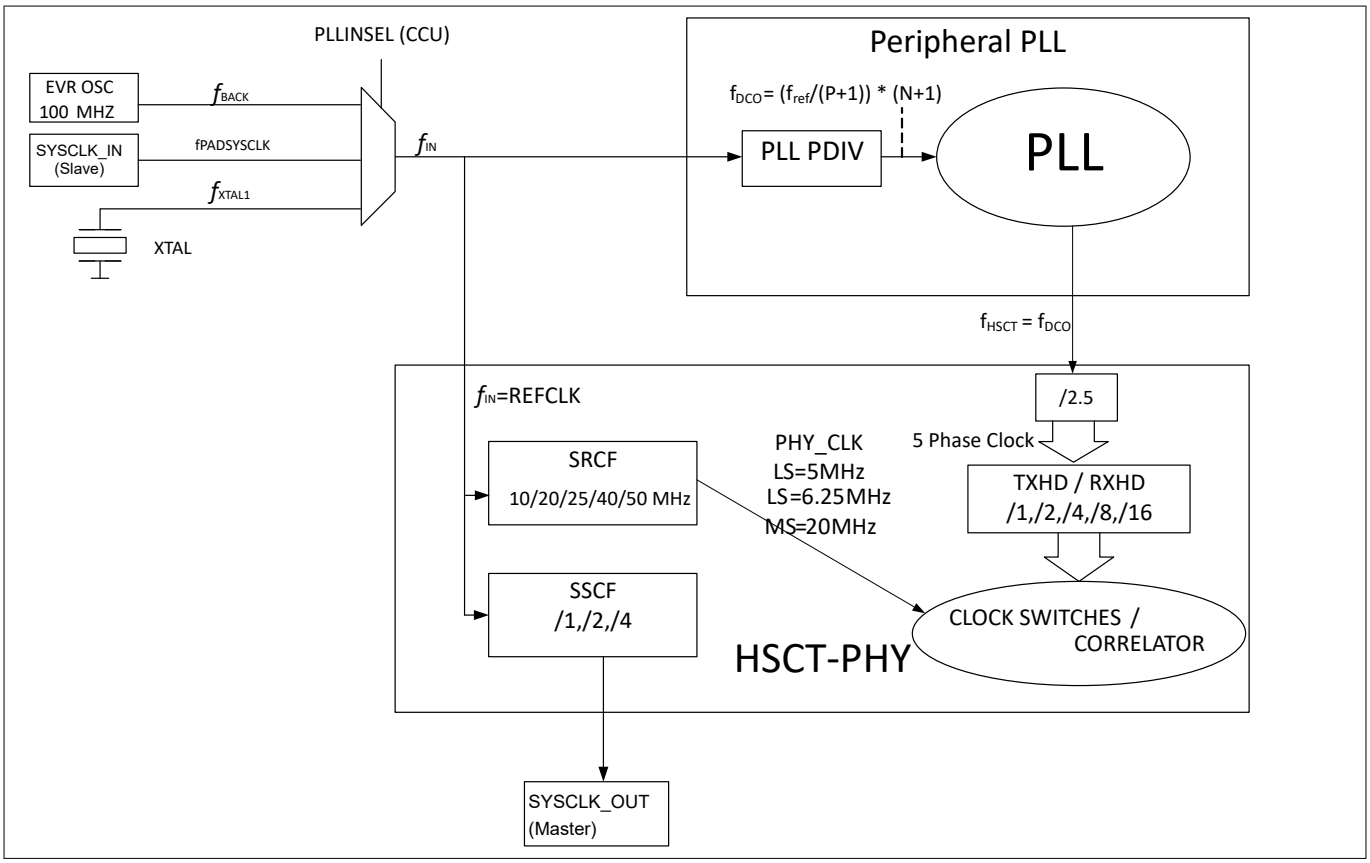


Figure 239 Frequency generation and clock divider

The following table gives more detailed information about the Baud rate configuration and the configuration bit's involved. The peripheral PLL reference frequency and their PLL frequency control word Multiplication factor (Peripheral PLL PDIV and NDIV) defines the PLL speed and are used to configure the output frequency of the PLL. The interface baud rate can be configured separately in TX direction (TXHD) and RX direction (RXHD). The table below shows the sample configuration in the TX direction.

Table 790 High speed mode selection

PLL speed fHSCT [MHz]	**800**	**800**	**800**	**800**	**800**
PLL reference frequency [MHz]	50	40	25	20	10
PLL PDIV (Peripheral PLL) [hex]	0x1	0x1	0x0	0x0	0x0
PLL NDIV (Peripheral PLL) [hex]	0x1F	0x27	0x1F	0x27	0x4F
TXHD $\frac{1}{1}$ [MBaud] (max Baud single Slave)	320	320	320	320	320
TXHD $\frac{1}{2}$ [MBaud] (max Baud two Slave)	160	160	160	160	160
TXHD $\frac{1}{4}$ [MBaud] (max Baud three Slave)	80	80	80	80	80

(table continues...)

Table 790 (continued) High speed mode selection

PLL speed fHSCT [MHz]	800	800	800	800	800
TXHD $\frac{1}{8}$ [MBaud]	40	40	40	40	40
TXHD $\frac{1}{16}$ [MBaud]	20	20	20	20	20

17.5.3.2.5 Correlator

Data transfer from the Master interface to the Slave interface can either be in “high speed mode” or “low speed mode” (for example $\frac{\text{Crystal}}{4}$ at 20 MHz HSCT SysClk or $\frac{\text{SysClk}}{2}$ at 10 MHz SysClk). A correlator is used to select the optimum sample phase for the frame based data transfer.

Phase selection

Every frame starts with a 16-bit sync field (see figure below) containing a synchronization pattern. Note that the sync pattern starts with 1, so a new frame always starts on the rising edge. In between frames, there is always at least one bit 0 in normal mode. Neither the Master interface nor the Slave interface may assume that the other side will maintain data phase relationship between frames. The receiving interface shall use the sync pattern to determine the start time of the frame relative to the local clock.

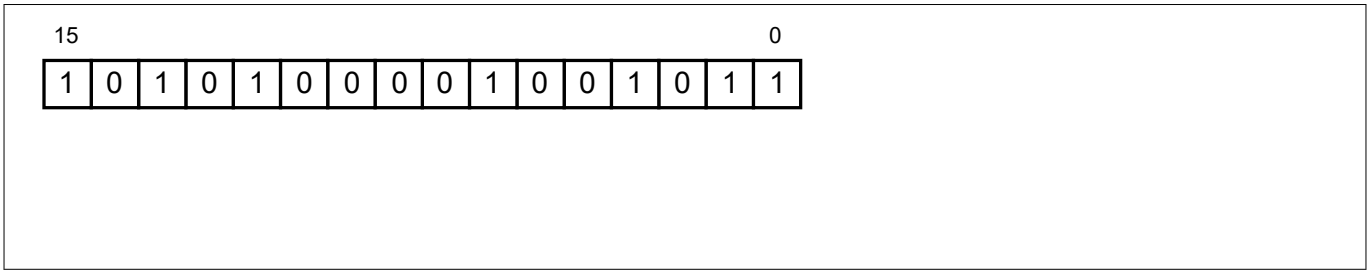


Figure 240 Synchronization pattern

The synchronization pattern is only needed for selecting the optimum sample phase, so it is not feed through to the output register of the correlator.

Payload size decoding

Every frame consists of three parts. A 16-bit synchronization pattern, an 8-bit header, and the payload field which can be 8, 32, 64, 96, 128, 256, or 288 bits. The header contains information about payload size and logical channel type. Although decoding the header is the task of the deframer, the payload size shall also be evaluated inside the correlator for finding the end of a frame.

The header is sent with MSB first. The first 3 bits (bit5, bit6, and bit7) contain the payload size of this frame.

Table 791 Payload size coding

b7:b5	Payload size (bits)	Total frame size (bits)
000	8	32
001	32	56
010	64	88

(table continues...)

Table 791 (continued) Payload size coding

b7:b5	Payload size (bits)	Total frame size (bits)
011	96	120
100	128	152
101	256	280
110	512	536 (not supported)
111	288	312

***Note**: In a harsh Automotive environment, a payload of 512 bit may cause increased error bit streams due to the long distance from one synchronization pattern to the next. This is the reason why this data transfer length is not supported.*

The correlator shall decode the payload size field and count the number of received bytes. After the last byte in a frame has been fed through to the correlator output register, the tuning of the multiplexer unit becomes invalid. In between frames at least one bit of value 0 shall be transmitted. The comparison procedure is started again as soon as there is a 1 at the MSB position in the shift registers.

The 8-bit header and the payload are used as input for the deframer block.

***Note**: The sync pattern and inter-frame bits are not passed to the deframer. An error in the sync pattern generates an error and is detected by the Physical layer.*

17.5.3.2.6 Jitter budget

The system clock is distributed directly from the Master interface to the Slave interface (see figure 56).

Subsequently, all jitter investigations considered in the following sections are referenced to the system clock f_{SYSCLK}.

Each system clock distribution has an associated filter function that incorporates the worst case combination of PLL bandwidth/peaking and equivalent jitter. The effective jitter seen at the LVDS receivers clock-data recovery inputs is a function of the difference in LVDS receiver PLL bandwidth and peaking and LVDS transmitter PLL bandwidth and peaking convolved with the jitter spectrum of the system clock.

System Clock variations contain jitter over a wide range of frequencies, some of which will be tracked by the LVDS receiver or otherwise removed by the combination of the LVDS transmitter and the LVDS receiver PLLs. Consequently, the nature of the filter functions is in part dependent on how the system clock is distributed. In general, the system clock jitter is filtered by the PLL difference function which comprehends the “mismatch” between LVDS transmitter and LVDS receiver PLL and also accounts for the impact of the transport delay.

Direct system clock distribution

This architecture provides direct distribution of the crystal clock as system clock from the Master interface to the Slave interface. Hence, the system clock is directly supplied to both LVDS transmitter PLL and LVDS receiver PLL. Hence, a major part of the clock jitter is sourced equally through LVDS transmitter and LVDS receiver PLLs. The figure below illustrates the direct system clock distribution, depicting noise sources, delay, and PLL transform sources. The amount of jitter appearing at the data recovery block is then defined by the transfer function difference of the LVDS transmitter and LVDS receiver PLLs, and by the sum of the clock jitter of the LVDS transmitter PLL, $X_{clkMaster(s)}$, and the LVDS receiver PLL, $X_{clkSlave(s)}$.

Based on the above clock architecture, a difference function may be defined that describes the “mismatch” between LVDS transmitter and LVDS receiver PLLs. As a first approximation, second order transfer functions are assumed.

The jitter of the direct system clock distribution is described by the phase noise power spectral density function as the absolute square of $X_{d(s)}$, as follows.

$$\left|X_d(s)\right|^2 = \left|X(s)\right|^2 \left|H_1(s)e^{-sT} - H_2(s)\right|^2 + \left|X_{clkMaster}(s)\right|^2 + \left|X_{clkSlave}(s)\right|^2 \quad (3)$$

where

$$H_{1(s)} = \frac{2s\,\zeta_1\omega_{n1} + \omega_{n1}^2}{s^2 + 2s\,\zeta_1\,\omega_{n1} + \omega_{n1}^2} \quad (4)$$

and

$$H_{2(s)} = \frac{2s\,\zeta_2\omega_{n2} + \omega_{n2}^2}{s^2 + 2s\,\zeta_2\,\omega_{n2} + \omega_{n2}^2} \quad (5)$$

Conversion between the natural PLL frequency ω_n and the -3 dB point is given by the following expression.

$$\omega_{3dB} = \omega_n\sqrt{1 + 2\zeta^2 + \sqrt{\left(1 + 2\,\zeta^2\right)^2 + 1}} \quad (6)$$

Under the assumption that crystal clock jitter can be ignored, that is,

$$X(s) \approx 0 \quad (7)$$

and LVDS transmitter to LVDS receiver transport delay can be neglected, that is,

$$e^{-sT} \approx 1 \quad (8)$$

the phase noise power spectral density of the direct system clock distribution can be simplified, as follows.

$$\left|X_d(s)\right|^2 = \left|X_{clkMaster}(s)\right|^2 + \left|X_{clkSlave}(s)\right|^2 \quad (9)$$

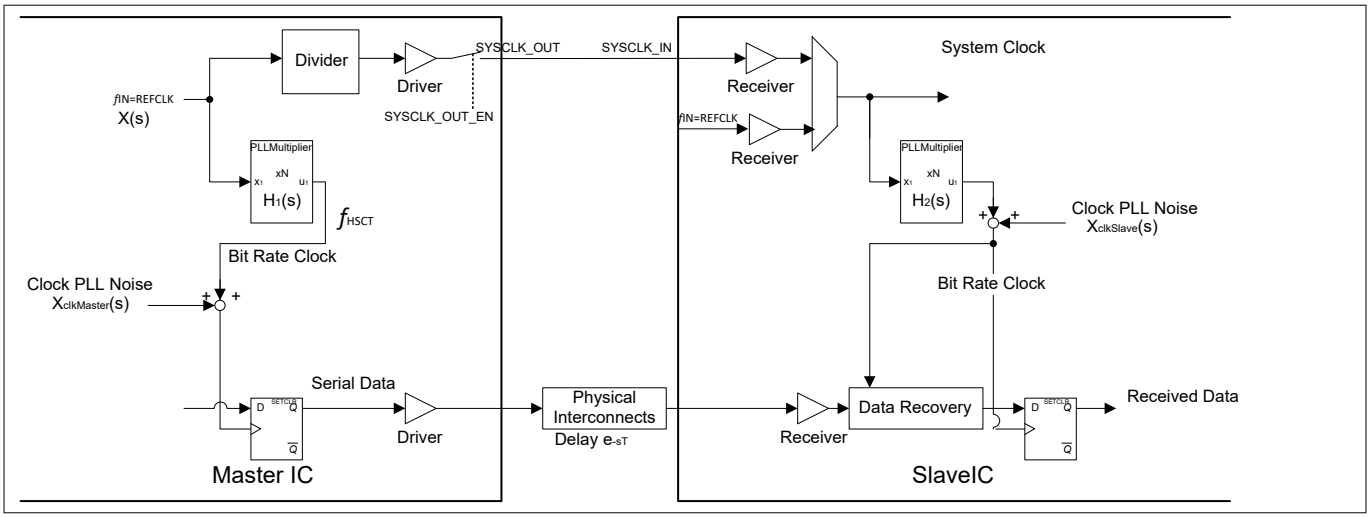


Figure 241 Direct system clock distribution

In the direct system clock distribution, the overall jitter is given by the sum of both phase noise spectral densities from the LVDS transmitter PLL, $X_{clkMaster(s)}$, and the LVDS receiver PLL, $X_{clkSlave(s)}$.

***Note**: The Slave interface requires a SysClk from the Master always as reference clock. It is not supported by the interface to have a separate crystal as reference clock at the Slave interface. The Master interface generated SysClk needs to be enabled.*

Jitter budgeting (direct system clock distribution)

To estimate the total rms jitter of LVDS transmitter PLL and LVDS receiver PLL for data recovery the following procedure has to be followed.

- Piecewise linear integration of the additive SSB (Single Sideband) phase noise for direct clock distribution to obtain the total phase noise power of the LVDS transmitter PLL
- Calculation of the rms jitter of the LVDS transmitter PLL as shown in the figure below
- Calculation of the total rms jitter *RJ* as the square root of the sum of the squares of the rms jitter from the LVDS transmitter PLL and the LVDS receiver PLL
- Calculation of total jitter that is random and deterministic, considering duty-cycle distortion and other effects to guarantee minimum eye opening at the interface

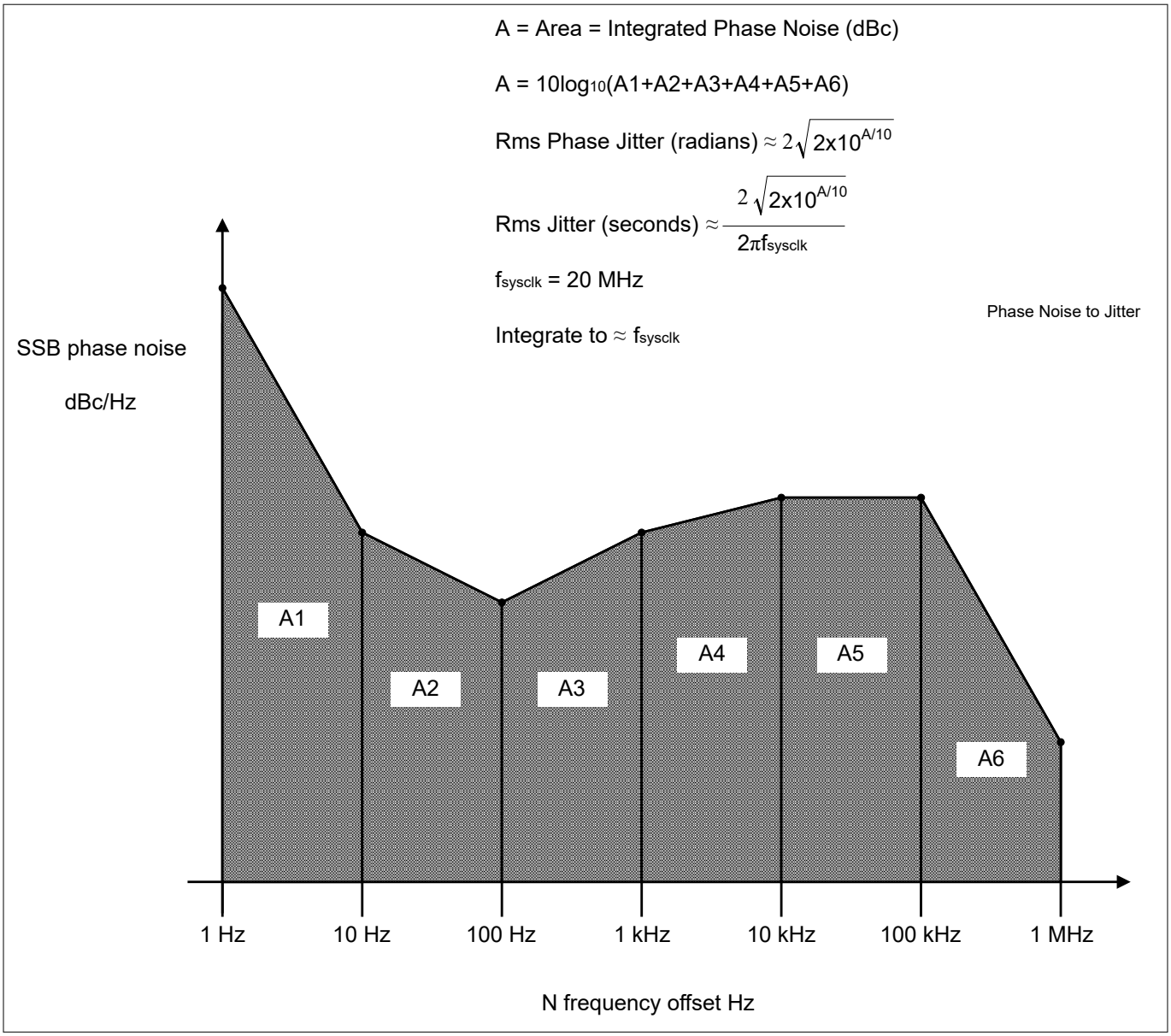


Figure 242 Calculation of jitter from SSB phase noise

Calculation of total jitter budget (direct system clock distribution)

To estimate the total jitter budget allowed for the random rms jitter *(RJ)* as described in the sections before, the following procedure for direct clock distribution has to be followed:

- Calculation of the jitter headroom available for the ideal jitter-free case given by the minimum eye opening, that is, 55% of a 320 MHz period
- Calculation of the total jitter headroom required under the assumption that the total long-term jitter *TJ* will not exceed the maximum peak-to-peak jitter specified
- Subtraction of two phases of the oversampling frequency used for data recovery taking into account the data-recovery uncertainty in multiple phases of the oversampling frequency. The data-recovery uncertainty is implementation specific and determined by the data-recovery oversampling factor DR_{OS}. The figure depicts the case where the selected sampling phase of the data recovery is almost one sample

phase away from the ideal sampling point. So the budget left for the peak jitter is reduced by almost one sample phase and the peak-to-peak jitter is reduced by almost two phases

- The remaining sum represents the jitter budget which is the upper limit for the peak-to-peak jitter calculated from the random rms jitter distribution *RJ* described in the sections before. Given a required bit error rate BER = 10^{-12} for SysClk 20 MHz and BER = 10^{-9} for SysClk 10 MHz

The jitter budget is given by the following equation.

$$TJ_{pp} = DJ_{pp} + RJ_{pp} = (DJ_{TX} + DJ_{RX}) + 2 \times Q \times \sqrt{J_{ABS}^2 + J_{ACC}^2} \quad (10)$$

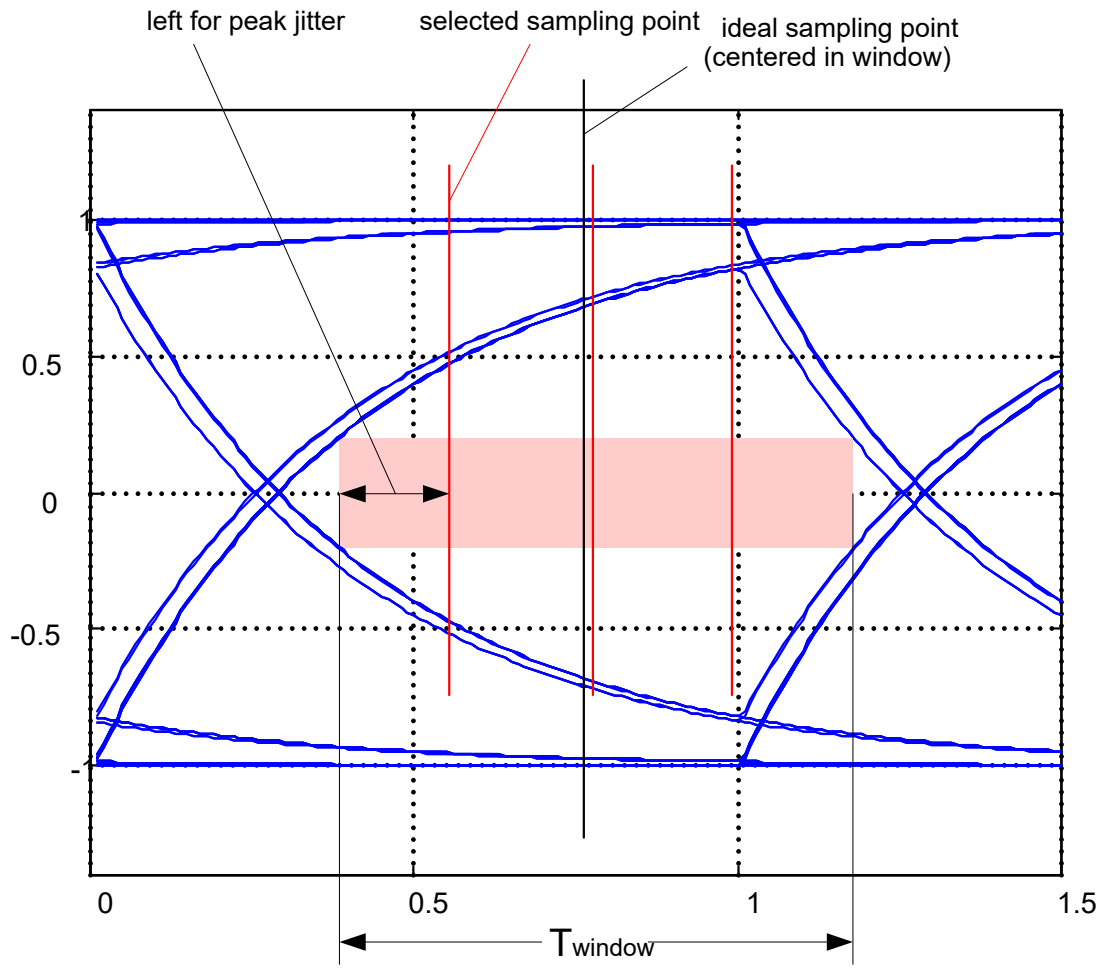


Figure 243 Sampling point selection

17.5.3.3 Protocol layer

The protocol layer is between the higher layer protocol (HSSL) and the physical layer. The protocol Layer is performing the frame generation in transmit direction and extracts the payload out of a frame in receive direction, called deframing.

17.5.3.3.1 Deframer

The deframer's tasks are, decoding the 8-bit header and separating the data stream according to the logical channel type.

During reset, the deframer shall ignore all data coming in from the correlator. This is a precaution measure to eliminate possible malfunctions due to floating HSCT lines.

Additionally the deframer checks the header for potential errors, for example, size is matching received data or size is allowed and received type is allowed. An error in the header discards the received data.

Header decoding

After the sync pattern has been filtered out by the correlator in a frame, the first 8 bits is the header or the frame type field. It contains information about the payload size and logical channel type coding. Please note again that all transfers are MSB first.

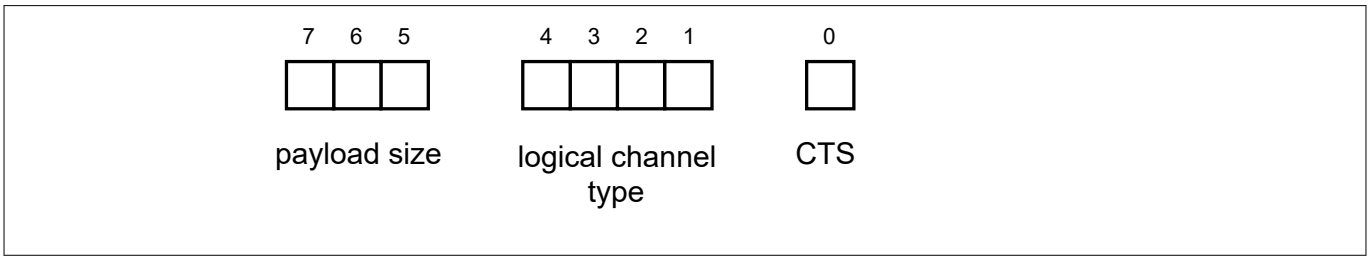


Figure 244 Header (frame type field)

- For payload size coding, refer to the Payload size decoding topic
- Bits 1 to 4 define the logical channel type. The table which follows will provide details about the logical channel type coding. Note that some kinds of transfer are exclusively used from the Master interface to the Slave interface or vice versa
- Bit 0 is used as Clear To Send bit (CTS) for transfers from the Slave interface to the Master interface (requires an enable through configuration). For transfers from the Master interface to the Slave interface also the Master interface can send CTS, which must also be enabled through configuration before. Also CTS from Master interface indicates the Master to the Slave that no more data can be accepted. This feature requires both for the Master and the Slave interface that higher layer indicates early enough in hardware that no more data can be accepted with the next frame. This needs to be send out with the frame based CTS signaling (Consider the transferred payload size for early enough CTS signaling through a frame)

Table 792 Logical channel type coding

b4:b1	Logical channel type master interface to slave interface (TX link)	Logical channel type slave interface to master interface (RX link)
0000	Interface Control; restrictions in multi-slave Mode	Interface Control (PING answer, 32-bit value; not supported in multi-slave mode)
0001	Master interface Unsolicited Status (32-bit only; not supported in multi-slave mode)	Slave interface Unsolicited Status (32-bit only; not supported in multi-slave mode)
0010	Reserved - receiver error interrupt	Reserved - receiver error interrupt
0011	CTS Transfers; not supported in multis-slave mode	CTS Transfer; not supported in multi-slave mode
0100	Data Channel A	Data Channel A
0101	Data Channel B	Data Channel B

(table continues...)

Table 792 (continued) Logical channel type coding

b4:b1	Logical channel type master interface to slave interface (TX link)	Logical channel type slave interface to master interface (RX link)
0110	Data Channel C	Data Channel C
0111	Data Channel D	Data Channel D
1000	Data Channel E (not supported, receiver generates an error interrupt)	Data Channel E (receiver error interrupt)
1001	Data Channel F (not supported, receiver error interrupt)	Data Channel F (receiver error interrupt)
1010	Data Channel G (not supported, receiver error interrupt)	Data Channel G (receiver error interrupt)
1011	Data Channel H (not supported, receiver generates an error interrupt)	Data Channel H (receiver error interrupt)
1100	Reserved - receiver error interrupt	Reserved - receiver error interrupt
1101	Reserved - receiver error interrupt	Reserved - receiver error interrupt
1110	Reserved - receiver error interrupt	Reserved - receiver error interrupt
1111	Reserved - receiver error interrupt	Reserved - receiver error interrupt

***Note**: A logical channel type TX transfer request of not supported or reserved types is not stopped or prevented to be sent out. Receiver side is expected to react with an error indication.*

***Note**: The received header information can be checked by software reading the Header status register (STAT). An error in the header is a severe error in a running system. Application software must handle the situation by interface link checks (PING. but it is not supported in multi-slave scenario). Additionally it is suggested to restart a running stream, if data loss cannot be tolerated. Received header frames are discarded, the same as the CTS information inside this error frame.*

Payload size for transmission data

For a transmission data transfer it is recommended to use a maximum 288-bit payload size.

Related information

Payload size decoding on page 2492

Logical channel separation

Reflecting the logical channel type to the higher level interface, the deframer sends the frame payload. The payload, as all other parts in a frame, is sent MSB first. The in the multi-slave scenario, it is used as a data protocol tunnel engine. Any addressing of the connected Slaves is done at a higher layer in hardware.

- All data received on and decoded as channel A to D data are sent through a 32-bit data interface to the higher layer hardware level
- Data can be sent using the 32-bit, 64-bit, 96-bit or 288-bit payload size

***Note**: Receiving a 8-bit header size on a logical channel results in an header error.*

17.5.3.3.2 Framer

The framer combines data and control information from the higher layer protocol to the frame payload. The payload is sent MSB first.

The framer codes the logical channel type and the payload size in the header according to the HSCT header format. Then a complete frame consisting of a 16-bit synchronization pattern, an 8-bit header, and a payload field, is put together.

The following logical channels are used for transmissions between Master interface and Slave interface:

- Data from the transmit unit: The frame will be sent to the receiving interface side as soon as the higher hardware layer transmit path contains enough data for the required payload size. The appropriate values will be signaled with the payload size signal and in the logical channel type field in the header. For data transfer, only the logical channel types Data Channel A to D are used. Receiving a header type reflecting the type Data Channel E to H generates an error
- Unsolicited status information can be sent by the Master interface and the Slave interface based on a 32-bit payload size. An unsolicited status can be used to send an unexpected and urgent message to the other side. Receiving an unsolicited message results in an interrupt at the receiving side. The Unsolicited status information can only be sent in a single slave scenario. In a multi-slave scenario this feature is not available

***Note**: The Slave interface in a single slave scenario (RX link) can send out a control command (channel type encoding 0h), which is the predefined PING answer with a defined 32-bit payload. No other control commands are available for the RX link to send. Receiving a header type not supported on RX link or TX link generates an error.*

The framer handles all frame parts. To determine the end of a frame, the framer counts the number of payload bits sent to the receiver side. In between frames bits 0 are transmitted.

Related information

Header decoding on page 2498

Phase selection on page 2492

Clear to send (CTS) for single slave scenario

The HSCT interface is able to use “data pull” in both directions (Master interface to Slave interface and Slave interface to Master interface) to perform flow control on the transmission data path.

The CTS bit in the header is asserted when the other side is allowed to send more data, and it is negated when the other side should stop transferring data.

A CTS deactivation can be sent out in a “normal” transmit frame, indicated by the header bit. The following payload is not discarded at the receiving interface side and is processed normally. CTS deactivation can be sent out in a CTS type transmit frame, if no actual transmit data is available. The clear to send information is indicated by the header CTS bit deactivation. The following payload is discarded at the receiving side. A received clear to send deactivation requires transmission to stop immediately after the actual frame that is in transmission, or the prevention from sending a new frame if no transmission is currently active. Control frames can still be sent.

The clear to send is independent of the channel having caused the back pressure (when the transmit side has sent the CTS, signaling the remote side/receiver, but the remote side is not clear to send, then this causes back pressure at the remote side/receiver). It simply stops the data channel transfer. As soon as the traffic jam is resolved the CTS bit in the next transmitted frame is activated again, or a separate CTS frame is sent with the CTS bit activated.

Higher layer software must analyze which channel caused the traffic jam and can send the information to the other side.

The use of the CTS signaling is optional. CTS signaling can be disabled by disabling the CTSCTRL.CTS_FRAME

By default the CTS bit in the header is set to 1. When the situation in the receive data path is reached, where no more data can be accepted , the CTS bit shall be reset to 0 in the header of the next frame sent. If there is currently no data in the higher layer module transmission data path to be sent in a new frame, a dedicated CTS transfer according to the logical channel type is generated. This transfer always uses the 8-bit payload size containing dummy bits. The payload in a CTS transfer will be discarded by the receiving side. Generation of dedicated CTS frames can be disabled by clearing CTSCTRL .CTS_FRAME.

The CTS bit is set to 1 again when the receive data path at the higher layer hardware protocol allows new data to be accepted. This is reflected by activating the CTS signal again.

The CTS signals are full duplex signals and are side information in the transferred data frame. The deframer filters the TXCTS information from the received frame header. The TXCTS is presented to the higher layer TX interface frame based. Changes within a frame are not intended. At the higher layer receiver side the higher layer activates the RXCTS signal if no more data can be received. The framer sends the RXCTS information with the header of the next frame.

Related information

See the Logical Channel Type Coding table on page 2498

17.5.3.3.3 Interface control

The Master interface is configured through a CPU based configuration. The Slave interface can be configured by sending an interface control command from the Master interface. The interface control command is used to configure the Slave interface into the defined mode.

The receiving Slave interface shall take no action if a reserved payload value is received. Instead it generates an error interrupt and hardware reacts on the received control command.

After start-up the interface is enabled to work at low speed (based on 20 MHz HSCT SysClk) and the PLL can be or might be off (if not used by other peripherals), regardless of the mode it had been in before. Shutting down the AURIX™ TC4xx device causes the interface to leave loopback-mode or clock test mode, if it was previously in any of these modes, and forces the receiver and transmitter into power-down mode.

The interface transmitter path is disabled by default. It is enabled by using the appropriate interface control, which is a payload command for the Slave interface and a control through software for the Master interface.

***Note**: No data can be transferred to the other side as long as the transmitter is disabled. In this case all data in the TX direction is discarded.*

The Interface control logical channel transfer command always uses the 8-bit payload size and is sent only by the Master interface. The payload is sent as MSB first. The interface control payload values are shown in the next table. All other payload values are reserved for future use. Whenever a reserved payload value is received it shall be ignored and reported by an error interrupt.

Table 793 Interface control payload values

***Note**: In multi-slave mode, it is not allowed to use the commands marked as unavailable.*

Value (hex)	Function	Multi-slave availability
00_H	“ping”. Sent by the Master interface. The Slave interface sends back a fixed 32-bit payload result.	No
01_H	Reserved	No
02_H	Slave interface clock multiplier start (in preparation for high-speed mode)[1)]	Yes

(table continues...)

Table 793 (continued) Interface control payload values

***Note**: In multi-slave mode, it is not allowed to use the commands marked as unavailable.*

Value (hex)	Function	Multi-slave availability
04_H	Slave interface clock multiplier stop[1)]	No
08_H	Select low-speed mode for transfers from the Master interface to the Slave interface	Yes
10_H	Select high-speed mode for transfers from the Master interface to the Slave interface	Yes
20_H	Select low-speed mode for transfers from the Slave interface to the Master interface	Yes
40_H	Select medium-speed mode for transfers from the Slave interface to the Master interface ***Note**: This is not allowed if SysClk = Crystal/2 (10 MHz) at Slave interface.*	Yes
80_H	Select high-speed mode for transfers from the Slave interface to the Master interface	Yes
31_H	Enable Slave interface transmitter[2)]	No
32_H	Disable Slave interface LVDS transmitter	Yes
34_H	Turn on clock test mode Send 101010… on Rx line continuously using currently configured Rx line rate; this is canceled by issuing the 'test mode off' command. Internal transmit and receive path is disabled. Received data is discarded. The only exception to this is the 'test mode off' function, which turns off the clock test mode and re-initializes for normal operation, retaining whichever interface speed mode is currently configured	No
38_H	Turn off test mode. Cancel clock test mode and payload loopback	No
51_H	Enable Slave interface LVDS transmitter of Slave 1 (only available in multi-slave mode)	Yes
52_H	Enable Slave interface LVDS transmitter of Slave 2 (only available in multi-slave mode)	Yes
53_H	Enable Slave interface LVDS transmitter of Slave 3 (only available in multi-slave mode)	Yes
FF_H	Turn on payload loopback Incoming frames at Slave interface are looped back until canceled by a Frame containing 'test mode off'. Internal transmit and receive path is disabled. Received data is discarded. The only exception to this is the 'test mode off' function, which turns off the loopback mode and re-initializes for normal operation, retaining whichever interface speed mode is currently configured ***Note**: This requires the same speed configuration for the RX and TX-link.*	No

1) The Slave PLL (Peripheral PLL) on a TC4xx device is also clock source for other modules on the SoC, and can only be switched on from the Master side using higher layer protocol commands, or with initialization software running on the Slave side. The command shows no effect on a TC4xx device at the Slave interface side. For ASICs and backward compatibility it is still supported.

2) For more details refer to the RX link state machine

***Note**: Before changing the speed mode, software has to take care that data Transmission on both links is stopped and the interface remains quiet.*

As a general principle, after a mode change the Master interface should request a data transfer from the Slave interface to stimulate a frame, which confirms the correct reception and indicates the mode change was successful. The PING function, available in the interface control logical channel in the single Slave scenario, is used to verify a successful speed change.

The Master interface generated PING has to be sent in system defined time intervals, to verify the link is still alive. Sending out a ping requires system software to activate a system timer. If the PING answer is not received within the system defined time in the single Slave scenario, then the link cannot be considered as stable and a new initialization is required. In single Slave scenario the Slave interface sends back only a fixed, known response based on a control command, at the currently configured link rate on the Rx data line.

Related information

Slow to fast data baud rate configuration on page 2503
Fast to slow data baud rate configuration on page 2504
TX link state machine on page 2504
Loopback mode on page 2505
RX link state machine on page 2506
Ping function on page 2507
Configuration on page 2507
Interface control processing times on page 2508

Slow to fast data baud rate configuration

The slow to fast interface speed change can be applied to either just one of the interface's data transfer directions or to both at the same time. At both sides, clock multipliers are required when either interface is to run at high speed.

If the RXData link is not enabled, but needs to be, the Master interface sends an 'Enable Slave Interface transmitter' command.

If the Slave interface clock multiplier is not already running, the Master interface sends a 'Slave interface clock multiplier start' command to the Slave interface at low speed. The peripheral PLL register needs to be directly configured to activate the PLL through higher layer protocol commands.

The Master interface enables its own clock multiplier, if it is not already running. After a proper settling time, the Master interface sends 'select high speed' to the Slave interface at the currently configured speed for the TXData link.

- The Slave interface switches the requested interface to high speed
- The Master interface switches the requested interface to high speed
- The Master interface sends a frame to the Slave interface and expects a response at the currently configured speed for the TxData link
- The Slave interface responds with a frame to confirm the mode change has worked

In order to select high speed for both RXData and TXData links the Master interface sends 'select high speed' for the RxData link first and then for the TxData link.

In the signle slave scenario it is recommended to send the PING for verifying a successful speed change.

For the multi-slave scenario the validity of the speed transition is performed using the higher layer protocol.

Related information

Interface control on page 2501

Fast to slow data baud rate configuration

The fast to slow interface speed change is the reverse of the slow-to-fast transition. Both clock multipliers must be running for either of the interfaces to operate in high-speed mode.

Before entering high speed mode it shall be assured that the high speed clock is available. This can be achieved by reading back Peripheral PLL lock information from PLL status register through higher layer protocol register access mechanism. If a command for entering high speed mode is received and the PLL is not locked, the command is ignored and lost. Interface control command value 02_H (PLL start) or 04_H (PLL stop), are not supported as the high speed reference clock comes from a common PLL also used by many other peripherals. Switching the PLL on or off through commands might lead to safety issues. Instead, the PLL switch on or switch off must be performed by register configuration through the higher layer protocol or through software routines established in the application software. Nevertheless the command is produced and sent out if required for backward compatibility.

Related information

Interface control on page 2501

TX link state machine

The TX transmission state machine controls transfers from the Master interface to Slave interface. The interface control value (int_ctrl) represents the received interface control payload value.

***Note**: Application software has to ensure, that no data communication is active on RX-link or TX-link during the time the interface speed is changed, otherwise a safe speed change without data loss can not be guaranteed interface.*

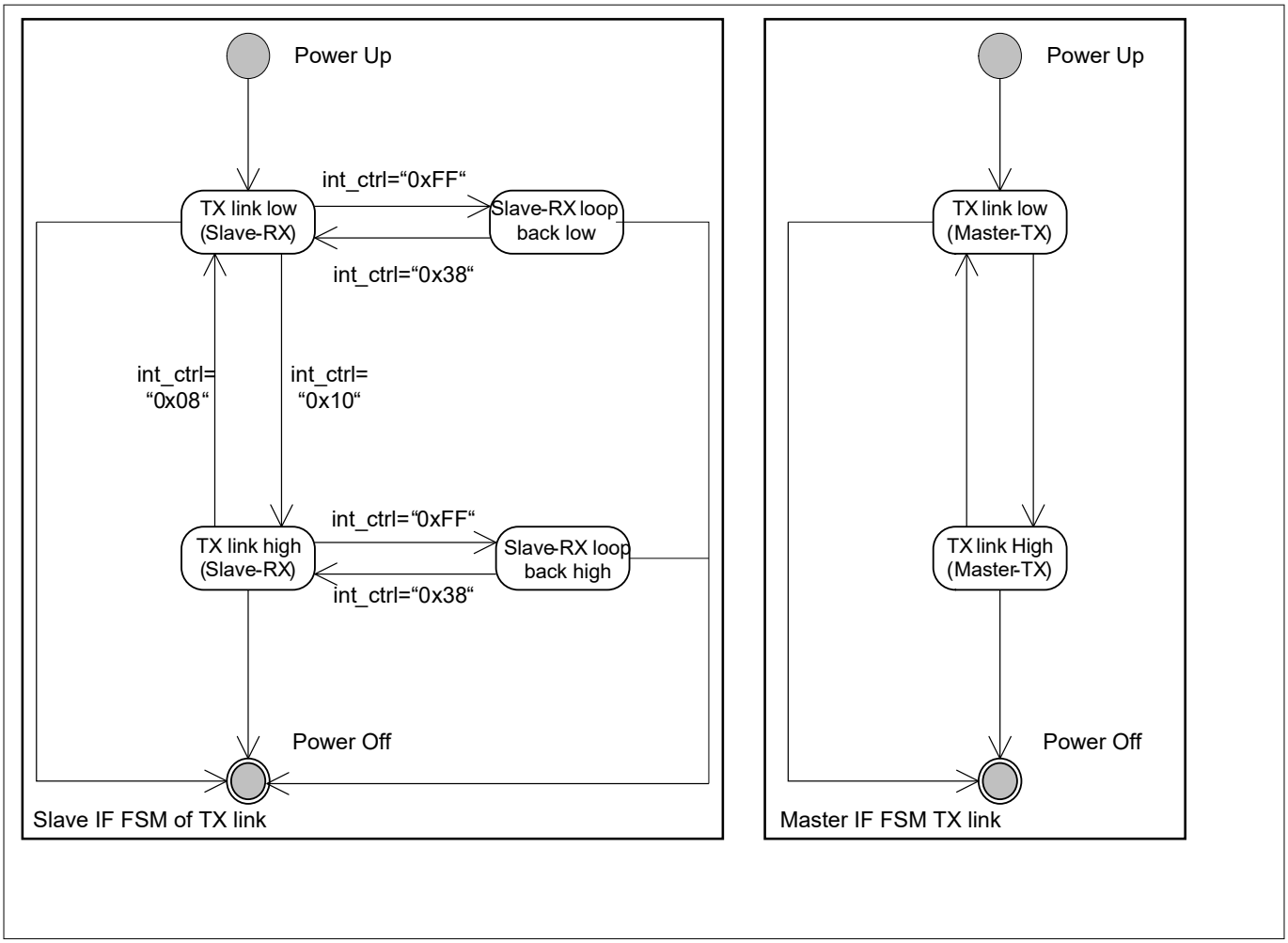


Figure 245 TX link state machine

Related information

Interface control on page 2501

Loopback mode

In the single slave scenario, loopback mode is used for test and verification to identify BER. When the Master interface demands loopback mode, the Slave interface internally loops back all received payload values in the following frames using the currently configured clock speed on the receive data link, without applying further processing internally.

Internally, transmit and receive chains shall be disabled. The header configuration is the same as in the incoming frame and the CTS bit in the header shall be asserted. Payload contents are ignored at Slave interface in loopback mode, except for the 'turn off test mode' interface control value, which ends loopback mode. The 'turn off test mode' frame is not looped back to the Master interface.

After leaving loopback mode, the interface is prepared for normal operation at the previously used clock speed.

***Note**: The 'loopback on' function has been intentionally coded at a large Hamming distance from the other codes. A minimum of 5, but usually 7, therefore requiring at least five bit-errors in the Header field for another function to be corrupted into 'loopback on'.*

Related information

Interface control on page 2501

RX link state machine

The RX link state machine is responsible for controlling transfers from a TC4xx device to the Master interface. The state machine is implemented on the Slave interface at the transmitting path.

The int_ctrl represents the received interface control payload value according to the interface control payload values.

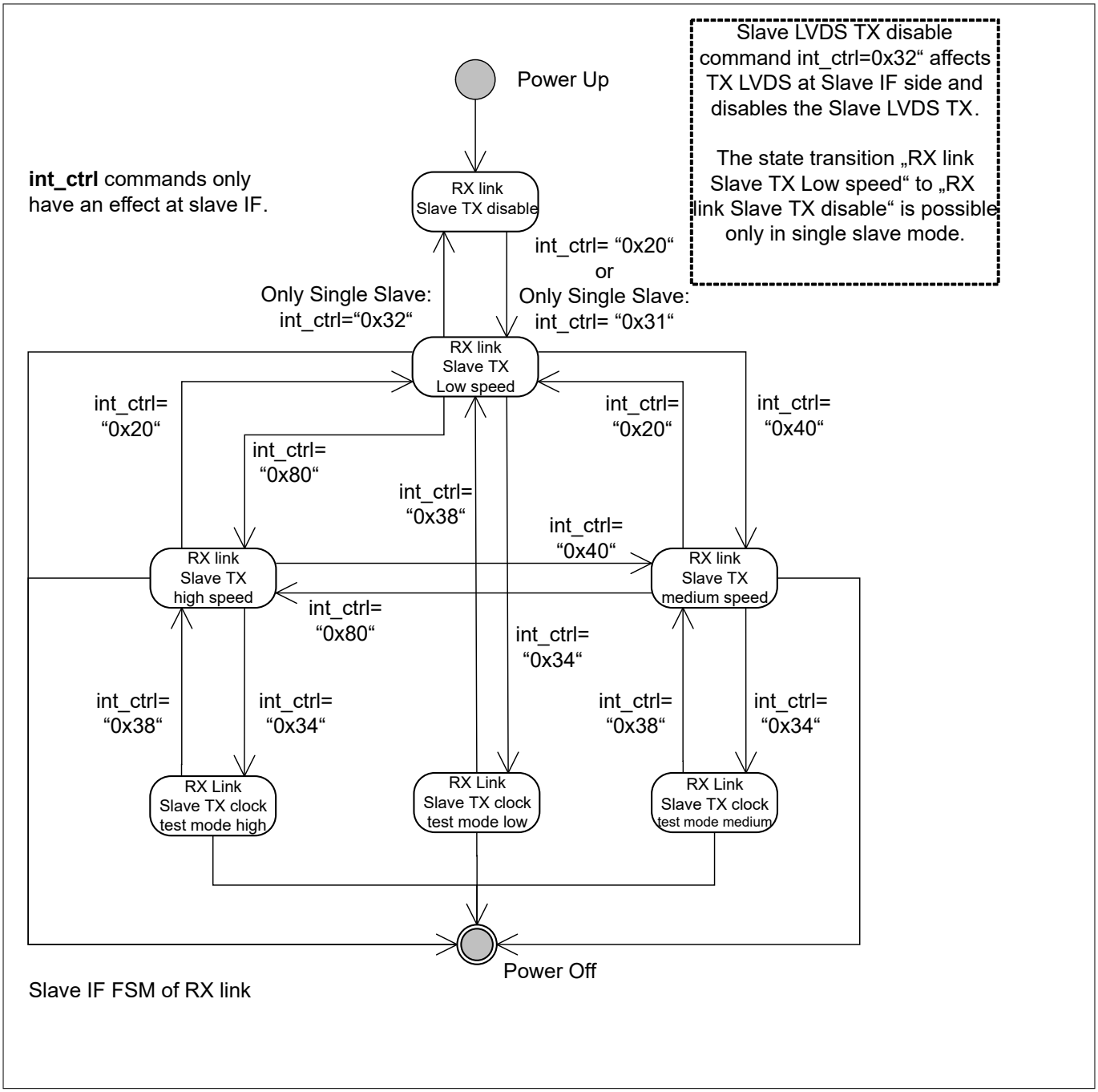


Figure 246 RX link state machine

The Slave interface TX, which is the RX link of the HSCT interfacing, can be activated from a disabled state into low-speed mode using the command "Select low-speed mode for transfers from the Slave interface to the Master interface" (20_H).

In single slave mode the command "Enable Slave interface LVDS transmitter" allows a change into "RX link Slave TX Low speed" mode.

Receiving the command "Disable Slave interface LVDS transmitter" in any of the other states disables the LVDS TX at slave side, but there is no state transition. This means for example being in high-speed mode and receiving the "Disable Slave interface LVDS transmitter" command, only the TX LVDS is disabled, but the state remains in high speed.

***Note**: The medium speed mode is only available, if the reference clock (SysClk) is running at 20 MHz.*

In clock test mode (only in single slave scenario) the transmitter sends '101010 …' continuously using the currently configured clock speed. Clock test mode can be left by sending the interface control value "turn off test mode".

Related information

Interface control on page 2501

Ping function

The ping function in TC4xx always uses the 32-bit payload size, sending back the value $ABCDEF01_H$ using the logical channel type Hex 0 (PING Status on RX link).

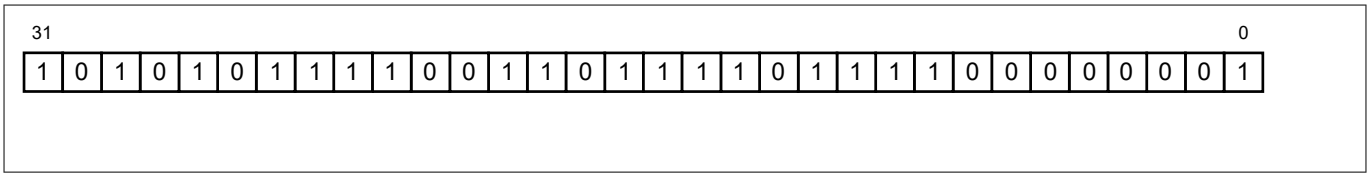


Figure 247 The Slave interface 'ping' response

Related information

Interface control on page 2501

Configuration

To configure the Slave interface, the Interface control commands have to be written into the Master interface register bits IFCTRL.IFCVS.

To send the control frame, 1b needs to be written to the register bit Interface Control Register.SIFCV to trigger the frame transfer.

The 8 register bits IFCTRL.IFCVS represent the same functionality as defined for the interface control payload values. Bit encoding is as shown in the Interface control payload values table.

The register field IFCTRL.IFCVS shall be written with the appropriate interface control payload value for every state transition shown in the TX state machine and Rx state machine figures.

The interface control in the Slave interface can alternatively be performed by programming the IFCTRL.IFCVS. This allows control of the Slave interface as well as the frame based configuration. The Master interface is only controlled by register configuration of the physical layer.

Related information

Interface control on page 2501

Interface control processing times

In the following table the processing times for the time consuming interface control transitions (as defined in the Interface control payload values) are listed.

Table 794 Interface control processing times

IFCVS (Hex)	Transition description	Timing parameter reference (datasheet)
02_H	Slave interface clock multiplier off to Slave interface high speed clock available[1)]	Not applicable
04_H	Slave interface clock multiplier on to Slave interface clock multiplier off[2)]	Not applicable
08_H	TX high speed mode to TX low speed mode	t_{HSLS}
10_H	TX low speed mode to TX high speed mode	t_{LSHS}
20_H	RX high or medium speed to RX low speed mode	t_{HMSLS}
40_H	RX high or low speed to RX medium speed mode	t_{HLSMS}
80_H	RX low or medium speed to RX high speed mode	t_{MLSHS}
31_H	Rx-link TX disabled to TX low speed mode	t_{tx}

1) TC4xx Slave interface shows no action if this command is received. PLL is automatically activated after power on.
2) TC4xx Slave interface shows no action if this command is received. PLL switch off is to be done using a higher layer protocol command, if required.

Related information

Interface control on page 2501

17.5.3.3.4 Power-up sequence

- The power up sequence of the HSCT Physical Layer and the LVDS IO requires to have the peripheral PLL in a locked state, a stable bandgap and bias distributor voltages before they can be enabled
  - It must also be ensured by the user that a correct clock is provided

- PLL Locked status can be read via STATPHY.PLOCK bit
- Bandgap enabling and bias distributor stabilization is automatically executed by HW after power on reset and takes t_{SET_LVDS} time

- For HSCT master, SYSCLK_OUT clock signal towards the slave is a direct output in the corresponding port. So, setting the bit PCSRx in the Selection Register for Module Port Control (PCSRSEL) is necessary for successful clock toggling (Please refer to Ports chapter for the register)
- For HSCT master, LVDS TX/RX PADS can be enabled in two possible ways depending on the value of bit TEN_CTRL/REN_CTRL set in the LVDS PAD Control register (LPCR)(Please refer to Ports chapter for these registers):
  - If TEN_CTRL/REN_CTRL =0 (port controlled condition), the LVDS TX/RX PADS are enabled in PORTS control by enabling the TX_EN/RX_EN bit in the LPCR register
  - If TEN_CTRL/REN_CTRL =1 (HSCT controlled condition), the LVDS TX/RX PADS are enabled via bit TX_DIS/RX_DIS in transmission disable register (DISABLE)
- The required time to power up LVDS TX PADS is about t_{tx} and LVDS RX PADS is t_{rxi}
- For HSCT Slave interface mode, the slave LVDS TX PADS are enabled/disabled with interface commands from master and LVDS RX PADS should always be enabled. Therefore, TEN_CTRL and REN_CTRL bits in the LVDS PAD Control register needs to be enabled by start-up software during initialization process
- After the peripheral PLL is in a locked state and LVDS is ready, the physical layer can be powered by enabling CONFIGPHY.PON
- After enabling the bit CONFIGPHY.PON, the module is ready to transfer data after t_{PON}time

***Note**: For actual values of the parameters please check the HSCT parameters section in the product specific datasheet.*

17.5.3.4 Use cases

The HSCT is as the name says a High Speed Communication Tunnel, which is used after initialization phase as high speed data transfer tunnel by higher layer protocol engines like HSSL. The interface is intended for chip to chip communication. The Micro Controller (MC) is from a system point of view intended to be the Master interface, if it owns the crystal. The connected Slave interface get's the clock (HSCT SysClk) provided by the master and can be operated without an additional reference clock for the internal PLL's. The reference clock at the Slave interface clocking the PLL is used to clock the data recovery path. The interface role definition (Master or Slave) is not allowed to change after the interface initialization phase.

There are two major UseCase scenarios in TC4xx the first one is the LVDS based point to point connection and the second is a point to multi point connection.

In both cases it is highly recommended to follow IFX PCB layout guidelines, which can be take as a reference to design a reliable PCB and meet the defined BER of 10^{-12}.

17.5.3.4.1 HSCT point-to-point LVDS based communication

The HSCT point to point LVDS based communication is a full-duplex link in TX and RX direction, which allows independent data transfer per link direction. It allows max baudrate of 320 MBaud with continuous data transfer in both directions. Subsequently some use cases are listed, which are an example of potential system scenarios.

MC to ASIC or FPGA

The MC (Master) is connected to the ASIC (Slave) and allows data streaming in both directions derived from higher layer hardware protocol (for example HSSL). In addition, the HSCT is used by the higher layer hardware

protocol to access Slave control register (read/write) and have therefore full controllability. This allows the MC to enhance it's system functions with the functions available in the connected ASIC or FPGA.

MC to MC

This use case allows to connect two MC's together. From the interface point of view, one of them is the Master function and other is the Slave function. Potential application would be register control or data streaming via higher layer hardware protocol (for example HSSL).

17.5.3.4.2 HSCT point to multi-point LVDS based communication

The is enhanced with the capability to have multiple Salves (max 3 Slaves) connected to one Master. Scenarios like MC connected to three ASICs or MC connected to two ASICs and one MC are supported. The initial Master configuration is fix and is not allowed to change during operation.

The connection of two ASICs (or ASIC and 1 MC) through LVDS allows max Baud rate of 160 MBaud.

The connection of three ASICs (or 2 ASICs and 1 MC) through LVDS additional reduces the max Baud rate to 80 MBaud.

The communication with each of the connected Slave devices is time sequential, derived by the MC. Only one Slave can be addressed (selection by Higher layer hardware protocol) and communication must be finished before the next Slave can be addressed. Rest of the Slaves stay quiet during the time one slave is addressed (selected). Higher Layer SW running on HSSL ensures this behavior and also addresses the respective slave device. Several transmitter driving the same LVDS link at once is prohibited. Electrical collisions are not allowed.

During Slave configuration phase commands are used and the Master is addressing all Slaves based on a Multi-Drop communication.

To achieve this scenario's careful high speed PCB design based on MicroStrip trace lines is required with well routed differential trace lines on PCB and good termination at the receiver input. In the multi-slave scenario the termination resistor needs to be a discrete 100 Ohm +/-5% or with less tolerance. An internal LVDS resistor is not supported in multiple Slave scenario. The location of the discrete resistor is suggested to be placed as close as possible to the Slave with a max geometrical distance from master defined (check IFX PCB layout guidelines for more details).

The LVDS IO (based on IEEE1596.3) offers several driver swing configurations, which allow application based adaptation.

17.5.3.5 Suspend and power-off behavior

Generally, the HSCT module transmits and receives data and control information through the differential signal interface.

A power down of the module can be done in any situation of the transmission. It is strongly recommended to have the transfers stopped from the higher layer protocol before sending the HSCT layer into power down state using the user software in a well defined way. Issuing an power down within a data transfer will result in data lose.

17.5.3.5.1 OCDS suspend

OCDS suspend request, for debugging purposes, simply freezes the module in the current state. This corresponds to the hard suspend of the whole module, which includes both the transmitter and the receiver. After the end of suspend state, the receive data will be most probably corrupted. Therefore, the whole interface must be reset, re-initialized and reactivated.

17.5.3.5.2 Disable request (power-off)

Going into power off requires that the module has finished all activities, which needs to be ensured by SW. It is also required that the higher level HSSL protocol has finished the transfers. It is the software responsibility to change to low speed mode and disable the transmitter and the receiver according to the application by writing the appropriate registers, as the change in speed mode and disabling of transmitter and receiver do not occur automatically. The HSCT high speed dividers need to be disabled (Init.RXHD = 0 and Init.TXHD = 0), before the physical layer is set to Power-Off mode. Power-Off mode of Physical Layer can be achieved by disabling the 5 phases with CONFIGPHY.PON. The clocks of the other parts of the HSCT module can be switched off by the CLC.DISR request bit or by the chip system sleep mode control (see CLC.EDIS).

17.5.3.6 Multi-HSCT support and restrictions

For family derivatives with two HSSL instances, the respective HSCT modules use the same peripheral PLL clock and also share the SysClk pin. The common SYSCLK output functionality is controlled by the HSCT0 register Init. Therefore, if both HSCT modules are active at the same time, they can either operate as two masters or two slaves. If both modules on one device are used as slaves, the corresponding master modules connected to these slaves must be from one device and use the same clock. For an overview of the HSSL/HSCT subsystem, see figure below.

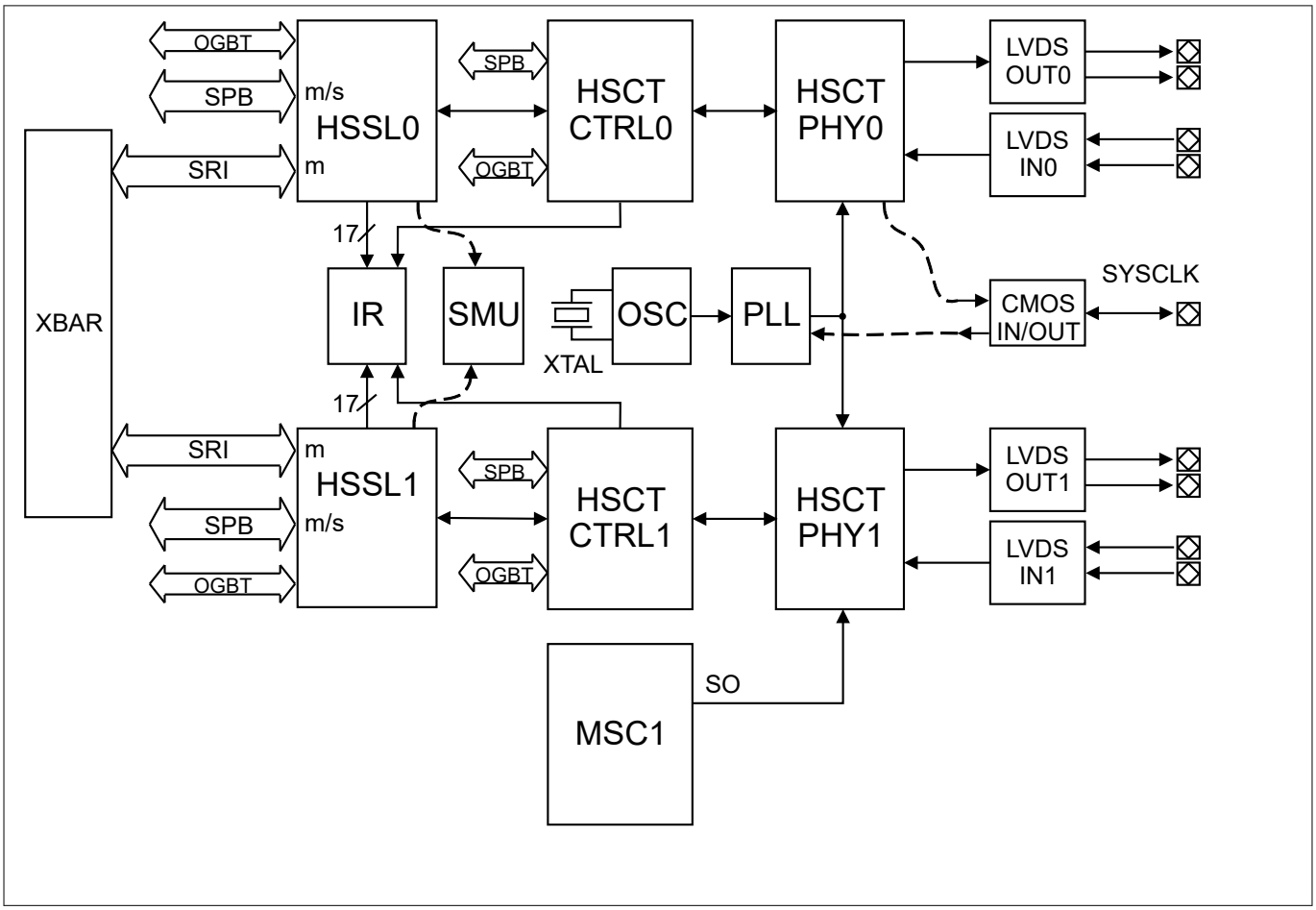


Figure 248 HSCT subsystem overview

The output LVDS Pads of the HSCT0 are shared with port signal. The port signal is either system PLL clock output or the port multiplexer output, which is selected by configuring port register PCSRSEL. The output LVDS pads of the HSCT1 are shared with an MSC and a QSPI module . The input LVDS pads are shared with QSPI. The software can select which module drives the LVDS output pad by configuring the port register LPCRx.TEN_CTRL, see figure below .

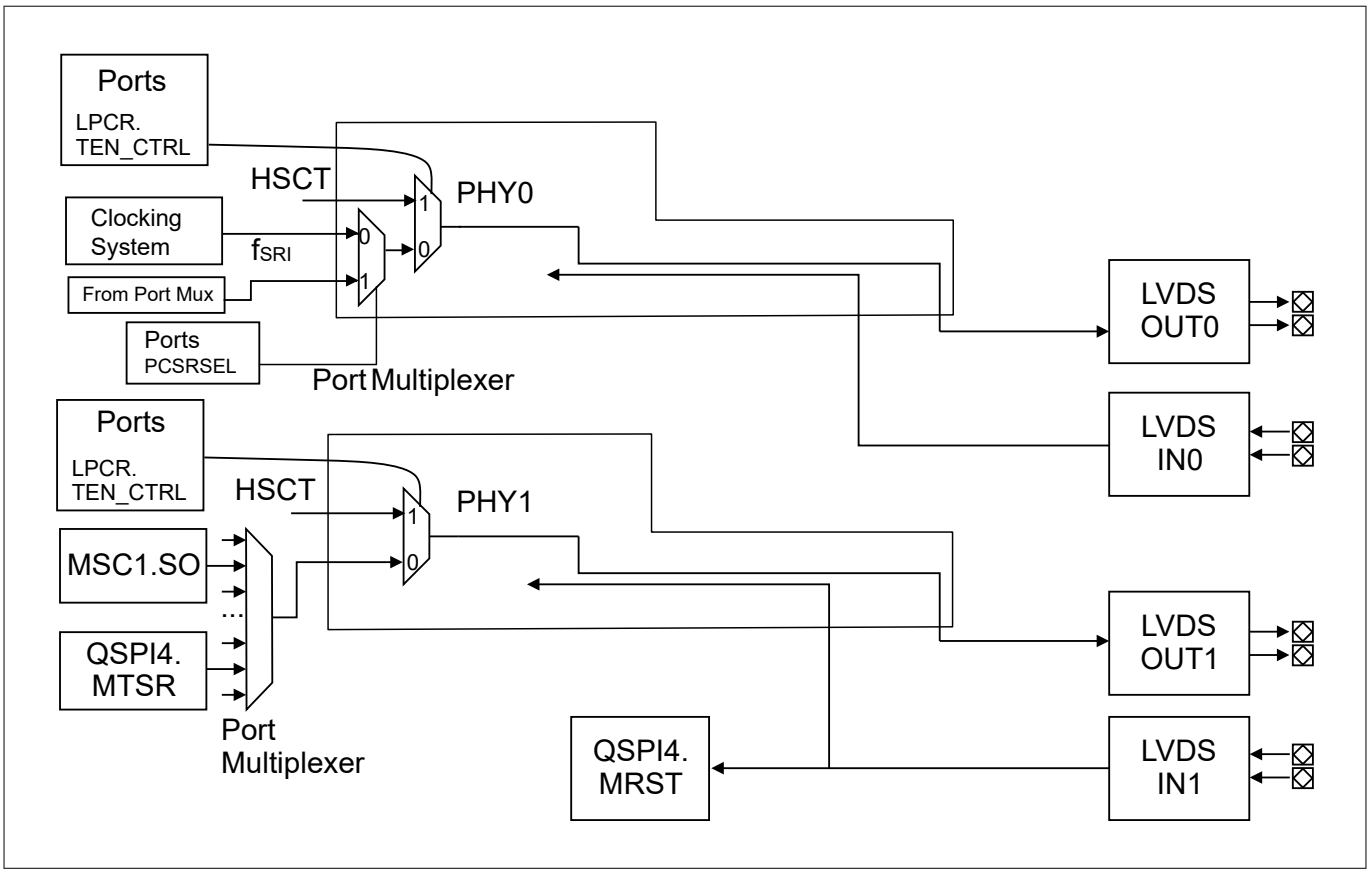


Figure 249 Pin sharing overview

17.5.3.7 Interrupts

The HSCT module generates one interrupt, called 'HSCT' towards the IR module.

The HSCT interrupt is triggered when any event represented in the register Interrupt register is triggered. These events can be masked or enabled with the help of register Interrupt Enable Register and can be cleared using the register Interrupt Clear Register.

17.6 Registers

17.6.1 HSSL registers

This section describes the registers of the HSSL module.

***Note**: The HSSL module occupies 1KByte of address space, although only the first 256 bytes are used. Accessing an address location that is not used generates a bus error.*

There are no registers in the HSSL with the destructive read property, where a read access would generate some action in the hardware.

17.6.1.1 Register overview - access mode glossary

Table 795 Register overview - access mode glossary

Keyword	Description
E	Access protection using PROT register PROTE.
SE	Access protection using PROT register PROTSE.
APU-P	Protection group consisting of registers ACCEN_WRA, ACCEN_WRB, ACCEN_RDA, ACCEN_RDB, ACCEN_VM, ACCEN_PRS.
P	Access protection using *APU-P* registers.
BE	Always returns a Bus Error.
U	No access restrictions.
PROT	Access restrictions as defined in the PROT register access rules.
SECURE_MASTER	Description can be found in global access mode definition.

17.6.1.2 Registers overview - HSSL (ascending offset address)

Table 796 Registers overview - HSSL (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CLC	Clock Control Register	000_H	P	P, SV, E	Application Reset	2545
OCS	OCDS Control and Status Register	004_H	P	SV, P	Debug Reset	2545
ID	Module Identification Register	008_H	P	BE	PowerOn Reset	2516
RST_CTRLA	Reset Control Register A	$00C_H$	P	P, SV, E	Application Reset	2547
RST_CTRLB	Reset Control Register B	010_H	P	P, SV, E	Application Reset	2547
RST_STAT	Reset Status Register	014_H	P	BE	Application Reset	2548

(table continues...)

Table 796 (continued) Registers overview - HSSL (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PROTE	PROT Register Endinit	020_H	U	SV, PROT	Application Reset	2549
PROTSE	PROT Register Safe Endinit	024_H	U	SV, PROT	Application Reset	2550
ACCEN_WRA	Write access enable register A	040_H	U	SE, SV	Application Reset	2552
ACCEN_WRB	Write access enable register B	044_H	U	SE, SV	Application Reset	2552
ACCEN_RDA	Read access enable register A	048_H	U	SE, SV	Application Reset	2553
ACCEN_RDB	Read access enable register B	$04C_H$	U	SE, SV	Application Reset	2553
ACCEN_VM	VM access enable register	050_H	U	SE, SV	Application Reset	2554
ACCEN_PRS	PRS access enable register	054_H	U	SE, SV	Application Reset	2554
CRC	CRC Control Register	060_H	P	SV, E, P	Kernel Reset	2516
CFG	Configuration Register	064_H	P	P	Kernel Reset	2517
QFLAGS	Request Flags Register	068_H	P	P	Kernel Reset	2518
MFLAGS	Miscellaneous Flags Register	$06C_H$	P	P	Kernel Reset	2520
MFLAGSSET	Miscellaneous Flags Set Register	070_H	P	P	Kernel Reset	2522
MFLAGSCL	Miscellaneous Flags Clear Register	074_H	P	P	Kernel Reset	2525
MFLAGSEN	Flags Enable Register	078_H	P	P	Kernel Reset	2528
SFSFLAGS	Stream FIFOs Status Flags Register	$07C_H$	P	P	Kernel Reset	2530
IWDx	Initiator Write Data Register x	080_H+x* 10_H	P	P	Kernel Reset	2532
ICONx	Initiator Control Data Register x	084_H+x* 10_H	P	P	Kernel Reset	2532
IRWAx	Initiator Read Write Address Register x	088_H+x* 10_H	P	P	Kernel Reset	2534
IRDx	Initiator Read Data Register x	$08C_H$+x* 10_H	P	P	Kernel Reset	2534
TCDi	Target Current Data Register i	$0C0_H$+i*8	P	P	Kernel Reset	2535
TCAi	Target Current Address Register i	$0C4_H$+i*8	P	P	Kernel Reset	2535
TSTAT	Target Status Register	$0E0_H$	P	P	Kernel Reset	2536

(table continues...)

Table 796 (continued) Registers overview - HSSL (ascending offset address)

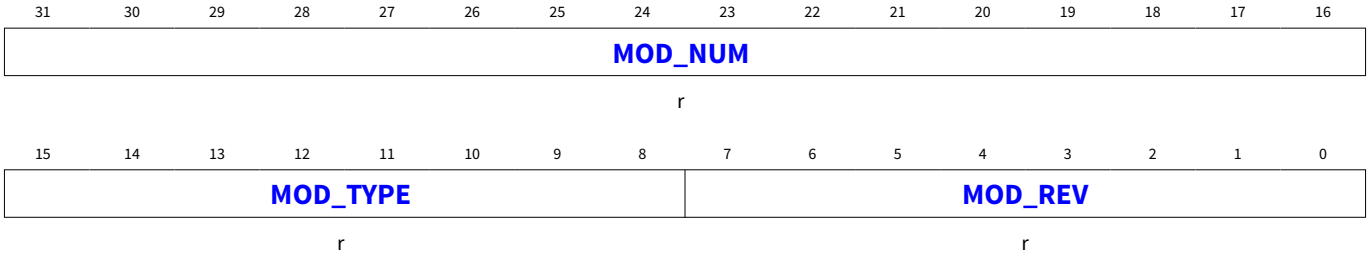
Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
TIDADD	Target ID Address Register	$0E4_H$	P	SV, P	Kernel Reset	2536
SEC	Security Control Register	$0E8_H$	SECURE_MASTER|P	SECURE_MASTER	Kernel Reset	2543
MSCR	Multi Slave Control Register	$0EC_H$	P	P	Kernel Reset	2543
ISSAx	Initiator Stream Start Address Register x	$0F0_H$+x*4	P	P	Kernel Reset	2537
ISCA	Initiator Stream Current Address Register	$0F8_H$	P	P	Kernel Reset	2537
ISFC	Initiator Stream Frame Count Register	$0FC_H$	P	P	Kernel Reset	2538
TSSAx	Target Stream Start Address Register x	100_H+x*4	P	P	Kernel Reset	2539
TSCA	Target Stream Current Address Register	108_H	P	P	Kernel Reset	2539
TSFC	Target Stream Frame Count Register	$10C_H$	P	P	Kernel Reset	2540
AWSTARTi	Access Window Start Register i	110_H+i*8	SV, P	SV, P	Kernel Reset	2541
AWENDi	Access Window End Register i	114_H+i*8	SV, P	SV, P	Kernel Reset	2541
AR	Access Rules Register	130_H	SV, P	SV, P	Kernel Reset	2542

17.6.1.3 Global Registers

17.6.1.3.1 Module Identification Register

ID Offset address: 008_H

Module Identification Register PowerOn Reset value: $00CB\ C004_H$



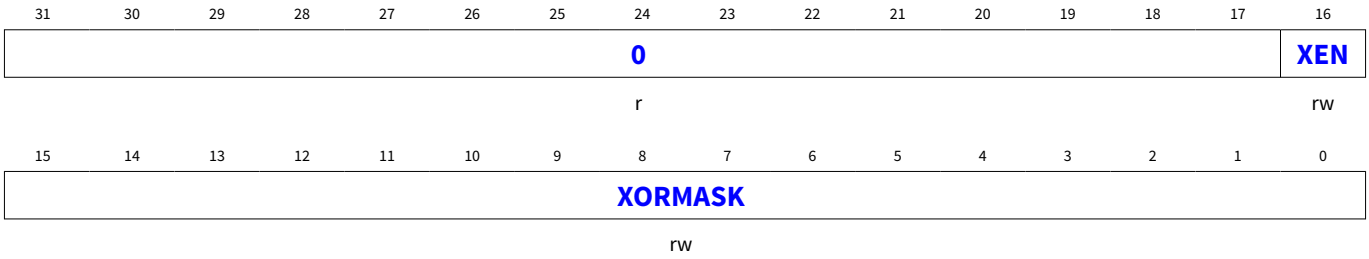
Field	Bits	Type	Description
MOD_REV	7:0	r	**Module Revision** Indicates the revision number of the implementation.
MOD_TYPE	15:8	r	**Module Type** This internal marker is fixed to $C0_H$.
MOD_NUM	31:16	r	**Module Number** Indicates the module identification number

17.6.1.3.2 CRC Control Register

This register only influences the generation of the CRC in the transmit direction.

CRC Offset address: 060_H

CRC Control Register Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
XORMASK	15:0	rw	**Value to be XORed with the Calculated CRC** Used for error injection / test purposes.
XEN	16	rw	**Enable the Error Injection via XORMASK** 0_B disabled 1_B enabled
0	31:17	r	**Reserved** Read as 0; should be written with 0

17.6.1.3.3 Configuration Register

The CFG register is used to configure the HSSL module. It shall only be written in the Initialize Mode.

***Note**: The duration of a timeout interval may vary for an amount of one predivider clock period. Therefore the predivider should be set as low as possible, and the timeout length as many predivider periods as possible.*

CFG Offset address: 064_H

Configuration Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0												CCC	SCM	SMR	SMT
r												rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		PREDIV													
r		rw													

Field	Bits	Type	Description
PREDIV	13:0	rw	**Global Predivider** Defines the down-scaled module clock to be used by all channel timeout timers. 0000_H 1 0001_H 2 0002_H 3 **others,** corresponding down-scale factor
SMT	16	rw	**Streaming Mode Transmitter** 0_B Continuous 1_B Single
SMR	17	rw	**Streaming Mode Receiver** 0_B Continuous 1_B Single
SCM	18	rw	**Streaming Channel Mode** Defines if the channel 2 is used in a streaming or command mode. 0_B Command 1_B Streaming
CCC	19	rw	**Channel Code Control** Defines the coding of the channel number in the HSSL header. 0_B Binary 1_B Special
0	15:14, 31:20	r	**Reserved** Read as 0; should be written with 0

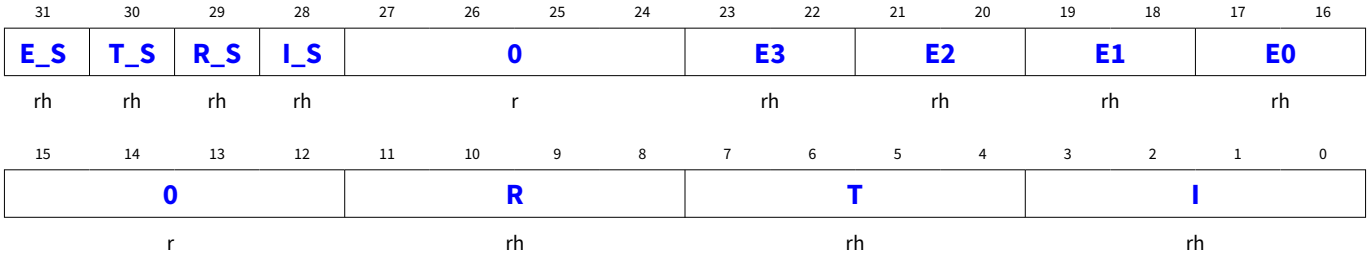
17.6.1.4 Channel Flags Registers

17.6.1.4.1 Request Flags Register

This register contains flags indicating if an appropriate action request is pending or not.

QFLAGS Offset address: 068_H

Request Flags Register Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
I	3:0	rh	**Request Flags for Initiated Commands** These flags are set by the corresponding channel when a WRT command is initiated. The WRT commands are initiated via the SPB bus. The S(tream) commands are initiated by the module internally, by the TXFIFO, except for the first stream frame start, which is done via the SPB bus. See MFLAGS.ISB and ISF flags. 0_H No request pending 1_H Channel 0 request pending 2_H Channel 1 request pending 4_H Channel 2 request pending 8_H Channel 3 request pending **others,** corresponding combination of requests
T	7:4	rh	**Request Flags for Commands Arrived at Target** These flags are set by the hardware according to the header information of a frame arrived at the target without a CRC error. They are used by the arbiter of the SRI/SPB master and cleared when the appropriate command is executed. 0_H No request pending 1_H Channel 0 request pending 2_H Channel 1 request pending 4_H Channel 2 request pending 8_H Channel 3 request pending **others,** corresponding combination of requests

(table continues...)

(continued)

Field	Bits	Type	Description
R	11:8	rh	**Request Flags for Response Frames at the Target** After a command has been executed by the SRI/SPB master, an appropriate flag is being set which indicates that an ACK/NACK/DATA is pending. 0_H No request pending 1_H Channel 0 request pending 2_H Channel 1 request pending 4_H Channel 2 request pending 8_H Channel 3 request pending **others,** corresponding combination of requests
E0	17:16	rh	**Expect Flags for Activated Timeout Timer 0** An appropriate two bit flag is set by the hardware when a timeout timer for a channel is started. The hardware clears the appropriate flag when any response frame arrives at the initiator. In case of an unexpected response the flag UNEXPECTED is additionally set. 00_B No request pending 01_B write request pending 10_B read request pending 11_B trigger request pending
E1	19:18	rh	**Expect Flags for Activated Timeout Timer 1**
E2	21:20	rh	**Expect Flags for Activated Timeout Timer 2**
E3	23:22	rh	**Expect Flags for Activated Timeout Timer 3**
I_S	28	rh	**I Flag for Stream Frames** See the “I” flag description above. 0_B No request pending 1_B Request pending
R_S	29	rh	**R Flag for Stream Frames** See the “R” flag description above. 0_B No request pending 1_B Request pending
T_S	30	rh	**T Flag for Stream Frames** See the “T” flag description above. 0_B No request pending 1_B Request pending
E_S	31	rh	**E Flag for Stream Frames** See the “E” flag description above. 0_B No request pending 1_B Request pending
0	15:12, 27:24	r	**Reserved** Read as 0; should be written with 0

17.6.1.4.2 Miscellaneous Flags Register

MFLAGS Offset address: $06C_H$

Miscellaneous Flags Register Kernel Reset value: $8000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
INI	TEO	TEI	TSE	0		CRCE	PIE2	PIE1	SRIE	MAV	ISB	IMB	TMB	0	
rh	rh	rh	rh	r		rh	rh	rh	rh	rh	rh	rh	rh	r	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UNEXPECTED				TIMEOUT				TTE				NACK			
rh				rh				rh				rh			

Field	Bits	Type	Description
NACK	3:0	rh	**Not Acknowledge Error - Target Error** Indicates for each channel that a target error frame has been received.
TTE	7:4	rh	**Transaction Tag Error** Indicates for each channel if a CRC correct acknowledge frame with an unexpected transaction tag number has been received.
TIMEOUT	11:8	rh	**Timeout Error** Indicates for each channel if an timeout event has occurred.
UNEXPECTED	15:12	rh	**Unexpected Type of Frame Error** Indicates for each channel if an unexpected or inappropriate response is received. For example a NACK for a Trigger frame or DATA for WRITE frame.
TMB	18	rh	**Target Memory Block** Selects the currently active memory block used by the target as a target for the streaming data, with its start address and frame counter. Switching the active block in the middle of a block transfer is not allowed. 0_B Memory block 0 1_B Memory block 1
IMB	19	rh	**Initiator Memory Block** Selects the currently active memory block used by the initiator as a source for the streaming data, with its start address and frame counter. Switching the active block in the middle of a block transfer is not allowed. 0_B Memory block 0 1_B Memory block 1

(table continues...)

(continued)

Field	Bits	Type	Description
ISB	20	rh	**Initiator Stream Block Request** Indicates if stream block request is pending. Set by the software to start a stream block transfer by using the MFLAGSSET.ISBS bit; clear by the software possible (if needed) by using the MFLAGSCL.ISBC bit; cleared by hardware at the end of the current block transfer in single mode, but not in continuous mode. 0_B No request or streaming ongoing 1_B Streaming ongoing
MAV	21	rh	**Memory Access Violation** Indicates a memory access violation. 0_B No violation 1_B Violation
SRIE	22	rh	**SRI/SPB Bus Access Error** Indicates an error on the SRI bus - transaction ID, ECC error or error acknowledge. Indicates an error on the SPB bus, error acknowledge or timeout. 0_B No error 1_B Error
PIE1	23	rh	**PHY Inconsistency Error 1(Channel Number Code Error)** Indicates if HSCT to HSSL channel number code comparator has detected an inconsistency error. 0_B No error 1_B Error
PIE2	24	rh	**PHY Inconsistency Error 2(Data Length Error)** Indicates if HSCT to HSSL data length comparator has detected an inconsistency error. 0_B No error 1_B Error
CRCE	25	rh	**CRC Error** Indicates if CRC checker has detected a CRC error. 0_B No error 1_B Error
TSE	28	rh	**Target Stream Enable** Used by the hardware to handle the single and continuous streaming. In single mode, cleared by hardware after the current block transfer ends. The module ignores afterwards the incoming steam frames. 0_B Disabled 1_B Enabled

(table continues...)

(continued)

Field	Bits	Type	Description
TEI	29	rh	**Transmit Enable Input** Indicates the state of the TEI input signal of the HSSL module, which is driven by the CTS output signal of the HSCT module. Any edge on this signal triggers an EXI interrupt. This low level signal stops the transmission of both command and response frames. Note: After kernel reset, when the HSSL module clock is enabled the TEI bit is updated with the current input value from the HSCT module.
TEO	30	rh	**Transmit Enable Output** Indicates the state of the TEO output signal of the HSSL module, which drives thee CTS input signal of the HSCT module. This bit is cleared by hardware at entering the INIT and Soft Suspend state.
INI	31	rh	**Initialize Mode** Indicates if the module is in the Initialize or Run mode. 0_B Run mode 1_B Initialize mode
0	17:16, 27:26	r	**Reserved** Read as 0; should be written with 0

17.6.1.4.3 Miscellaneous Flags Set Register

***Note**: Read access to this register returns all zero.*

MFLAGSSET Offset address: 070_H

Miscellaneous Flags Set Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
INIS	TEOS	0	TSES	0		CRCES	PIE2S	PIE1S	SRIES	MAVS	ISBS	IMBS	TMBS	0	
w	w	r	w	r		w	w	w	w	w	w	w	w	r	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UNEXPECTEDS				TIMEOUTS				TTES				NACKS			
w				w				w				w			

Field	Bits	Type	Description
NACKS	3:0	w	**NACK Flags Set** Writing 1 sets the corresponding bit in the MFLAGS register and triggers the ERR interrupt for the corresponding channel, if enabled in the MFLAGSEN register. Writing 0 has no effect. 0_H no action 1_H set channel 0 2_H set channel 1 4_H set channel 2 8_H set channel 3 **others,** set the corresponding combination of channels
TTES	7:4	w	**Transaction Tag Error Flags Set** Writing 1 sets the corresponding bit in the MFLAGS register and triggers the ERR interrupt for the corresponding channel, if enabled in the MFLAGSEN register. Writing 0 has no effect. 0_H no action 1_H set channel 0 2_H set channel 1 4_H set channel 2 8_H set channel 3 **others,** set the corresponding combination of channels
TIMEOUTS	11:8	w	**Timeout Error Flags Set** Writing 1 sets the corresponding bit in the MFLAGS register and triggers the ERR interrupt for the corresponding channel, if enabled in the MFLAGSEN register. Writing 0 has no effect. 0_H no action 1_H set channel 0 2_H set channel 1 4_H set channel 2 8_H set channel 3 **others,** set the corresponding combination of channels
UNEXPECTEDS	15:12	w	**Unexpected Error Flags Set** Writing 1 sets the corresponding bit in the MFLAGS register and triggers the ERR interrupt for the corresponding channel, if enabled in the MFLAGSEN register. Writing 0 has no effect. 0_H no action 1_H set channel 0 2_H set channel 1 4_H set channel 2 8_H set channel 3 **others,** set the corresponding combination of channels

(table continues...)

(continued)

Field	Bits	Type	Description
TMBS	18	w	**Target Memory Block Flag Set** Writing 1 sets the corresponding bit in the MFLAGS register. Writing 0 has no effect. 0_B No action 1_B Set
IMBS	19	w	**Initiator Memory Block Flag Set** Writing 1 sets the corresponding bit in the MFLAGS register. Writing 0 has no effect. 0_B No action 1_B Set
ISBS	20	w	**Initiator Stream Block Request Set** Writing 1 sets the corresponding bit in the MFLAGS register. Writing 0 has no effect. 0_B No action 1_B Set
MAVS	21	w	**MAV Flag Set** Writing 1 sets the corresponding bit in the MFLAGS register, and generates an EXI interrupt if enabled in the corresponding bit of the MFLAGSEN register. Writing 0 has no effect 0_B No action 1_B Set
SRIES	22	w	**SRI/SPB Bus Access Error Flag Set** Writing 1 sets the corresponding bit in the MFLAGS register, and generates an EXI interrupt if enabled in the corresponding bit of the MFLAGSEN register. Writing 0 has no effect 0_B No action 1_B Set
PIE1S	23	w	**PIE1 Error Flag Set** Writing 1 sets the corresponding bit in the MFLAGS register, and generates an EXI interrupt if enabled in the corresponding bit of the MFLAGSEN register. Writing 0 has no effect 0_B No action 1_B Set
PIE2S	24	w	**PIE2 Error Flag Set** Writing 1 sets the corresponding bit in the MFLAGS register, and generates an EXI interrupt if enabled in the corresponding bit of the MFLAGSEN register. Writing 0 has no effect. 0_B No action 1_B Set

(table continues...)

(continued)

Field	Bits	Type	Description
CRCES	25	w	**CRC Error Flag Set** Writing 1 sets the corresponding bit in the MFLAGS register, and generates an EXI interrupt if enabled in the corresponding bit of the MFLAGSEN register. Writing 0 has no effect. 0_B No action 1_B Set
TSES	28	w	**Target Stream Enable Flag Set** Writing 1 sets the corresponding bit in the MFLAGS register. Writing 0 has no effect. 0_B No action 1_B Set
TEOS	30	w	**Transmit Enable Flag Set** Writing 1 sets the corresponding bit in the MFLAGS register. Writing 0 has no effect. 0_B No action 1_B Set
INIS	31	w	**Initialize Mode Flag Set** Writing 1 sets the corresponding bit in the MFLAGS register. Writing 0 has no effect. 0_B No action 1_B Set
0	17:16, 27:26, 29	r	**Reserved** Read as 0; should be written with 0

17.6.1.4.4 Miscellaneous Flags Clear Register

***Note**: Read access to this register returns all zero.*

MFLAGSCL Offset address: 074_H

Miscellaneous Flags Clear Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
INIC	TEOC	0	TSEC	0		CRCEC	PIE2C	PIE1C	SRIEC	MAVC	ISBC	IMBC	TMBC	0	
w	w	r	w	r		w	w	w	w	w	w	w	w	r	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UNEXPECTEDC				TIMEOUTC				TTEC				NACKC			
w				w				w				w			

Field	Bits	Type	Description
NACKC	3:0	w	**NACK Flags Clear** Writing 1 clears the corresponding bit in the MFLAGS register. Writing 0 has no effect. 0_H no action 1_H clear channel 0 2_H clear channel 1 4_H clear channel 2 8_H clear channel 3 **others,** clear the corresponding combination of channels
TTEC	7:4	w	**Transaction Tag Error Flags Clear** Writing 1 clears the corresponding bit in the MFLAGS register. Writing 0 has no effect. 0_H no action 1_H clear channel 0 2_H clear channel 1 4_H clear channel 2 8_H clear channel 3 **others,** clear the corresponding combination of channels
TIMEOUTC	11:8	w	**Timeout Error Flags Clear** Writing 1 clears the corresponding bit in the MFLAGS register. Writing 0 has no effect. 0_H no action 1_H clear channel 0 2_H clear channel 1 4_H clear channel 2 8_H clear channel 3 **others,** clear the corresponding combination of channels
UNEXPECTEDC	15:12	w	**Unexpected Error Flags Clear** Writing 1 clears the corresponding bit in the MFLAGS register. Writing 0 has no effect. 0_H no action 1_H clear channel 0 2_H clear channel 1 4_H clear channel 2 8_H clear channel 3 **others,** clear the corresponding combination of channels
TMBC	18	w	**Target Memory Block Flag Clear** Writing 1 clears the corresponding bit in the MFLAGS register. Writing 0 has no effect. 0_B No action 1_B Clear

(table continues...)

(continued)

Field	Bits	Type	Description
IMBC	19	w	**Initiator Memory Block Flag Clear** Writing 1 clears the corresponding bit in the MFLAGS register. Writing 0 has no effect. 0_B No action 1_B Clear
ISBC	20	w	**Initiator Stream Block Request Clear** Writing 1 clears the corresponding bit in the MFLAGS register. Writing 0 has no effect. 0_B No action 1_B Clear
MAVC	21	w	**MAV Flag Clear** Writing 1 clears the corresponding bit in the MFLAGS register. Writing 0 has no effect 0_B No action 1_B Clear
SRIEC	22	w	**SRI/SPB Bus Access Error Flag Clear** Writing 1 clears the corresponding bit in the MFLAGS register. Writing 0 has no effect 0_B No action 1_B Clear
PIE1C	23	w	**PIE1 Error Flag Clear** Writing 1 clears the corresponding bit in the MFLAGS register. Writing 0 has no effect 0_B No action 1_B Clear
PIE2C	24	w	**PIE2 Error Flag Clear** Writing 1 clears the corresponding bit in the MFLAGS register. Writing 0 has no effect. 0_B No action 1_B Clear
CRCEC	25	w	**CRC Error Flag Clear** Writing 1 clears the corresponding bit in the MFLAGS register. Writing 0 has no effect. 0_B No action 1_B Clear
TSEC	28	w	**Target Stream Enable Flag Clear** Writing 1 sets the corresponding bit in the MFLAGS register. Writing 0 has no effect. 0_B No action 1_B Clear

(table continues...)

(continued)

Field	Bits	Type	Description
TEOC	30	w	**Transmit Enable Flag Clear** Writing 1 clears the corresponding bit in the MFLAGS register. Writing 0 has no effect. 0_B No action 1_B Clear
INIC	31	w	**Initialize Mode Flag Clear** Writing 1 clears the corresponding bit in the MFLAGS register. Writing 0 has no effect. 0_B No action 1_B Clear
0	17:16, 27:26, 29	r	**Reserved** Read as 0; should be written with 0

17.6.1.4.5 Flags Enable Register

MFLAGSEN Offset address: 078_H

Flags Enable Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		TEIEN	0			CRCEEN	PIE2EN	PIE1EN	SRIEEN	MAVEN	0				
r		rw	r			rw	rw	rw	rw	rw	r				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UNEXPECTEDEN				TIMEOUTEN				TTEEN				NACKEN			
rw				rw				rw				rw			

Field	Bits	Type	Description
NACKEN	3:0	rw	**Not Acknowledge Error Enable Bits** Used to enable the error interrupt associated to the corresponding bit in the MFLAGS register. 0_H all disabled 1_H enable channel 0 2_H enable channel 1 4_H enable channel 2 8_H enable channel 3 **others,** enable the corresponding combination of channels

(table continues...)

(continued)

Field	Bits	Type	Description
TTEEN	7:4	rw	**Transaction Tag Error Enable Bits** Used to enable the error interrupt associated to the corresponding bit in the MFLAGS register. 0_H all disabled 1_H enable channel 0 2_H enable channel 1 4_H enable channel 2 8_H enable channel 3 **others,** enable the corresponding combination of channels
TIMEOUTEN	11:8	rw	**Timeout Error Enable Bits** Used to enable the error interrupt associated to the corresponding bit in the MFLAGS register. 0_H all disabled 1_H enable channel 0 2_H enable channel 1 4_H enable channel 2 8_H enable channel 3 **others,** enable the corresponding combination of channels
UNEXPECTEDEN	15:12	rw	**Unexpected Error Enable Bits** Used to enable the error interrupt associated to the corresponding bit in the MFLAGS register. 0_H all disabled 1_H enable channel 0 2_H enable channel 1 4_H enable channel 2 8_H enable channel 3 **others,** enable the corresponding combination of channels
MAVEN	21	rw	**MAV Enable Bit** Used to enable the interrupt associated to the corresponding bit in the MFLAGS register. 0_B Disabled 1_B Enabled
SRIEEN	22	rw	**SRI/SPB Bus Access Error Enable Bit** Used to enable the error interrupt associated to the corresponding bit in the MFLAGS register. 0_B Disabled 1_B Enabled
PIE1EN	23	rw	**PIE1 Error Enable Bit** Used to enable the error interrupt associated to the corresponding bit in the MFLAGS register. 0_B Disabled 1_B Enabled

(table continues...)

(continued)

Field	Bits	Type	Description
PIE2EN	24	rw	**PIE2 Error Enable Bit** Used to enable the error interrupt associated to the corresponding bit in the MFLAGS register. 0_B Disabled 1_B Enabled
CRCEEN	25	rw	**CRC Error Enable Bit** Used to enable the error interrupt associated to the corresponding bit in the MFLAGS register. 0_B Disabled 1_B Enabled
TEIEN	29	rw	**TEI Enable Bit** Used to enable the interrupt associated to the corresponding bit in the MFLAGS register. 0_B Disabled 1_B Enabled
0	20:16, 28:26, 31:30	r	**Reserved** Read as 0; should be written with 0

17.6.1.4.6 Stream FIFOs Status Flags Register

SFSFLAGS Offset address: $07C_H$

Stream FIFOs Status Flags Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ISF	0									EXFL		TXFL		RXFL	
rh	r									rh		rh		rh	

Field	Bits	Type	Description
RXFL	1:0	rh	**Stream RxFIFO Filling Level** Indicates the filling level of the FIFO with granularity of 32 bytes (one stream frame payload size). 00_B 0 01_B 1 10_B 2 11_B Reserved (not possible)

(table continues...)

(continued)

Field	Bits	Type	Description
TXFL	3:2	rh	**Stream TxFIFO Filling Level** Indicates the filling level of the FIFO with granularity of 32 bytes (one stream frame payload size). 00_B 0 01_B 1 10_B 2 11_B Reserved (not possible)
EXFL	5:4	rh	**Stream Expect FIFO Filling Level** Indicates the filling level of the FIFO with granularity of 32 bytes (one stream frame payload size). 00_B 0 01_B 1 10_B 2 11_B Reserved (not possible)
ISF	15	rh	**Initiator Stream Frame Request** Indicates if stream TXFIFO request is pending. Set and cleared by the TXFIFO. 0_B TXFIFO fill request not pending 1_B TXFIFO fill request pending
0	14:6, 31:16	r	**Reserved** Read as 0; should be written with 0

17.6.1.5 Channel Initiator Registers

This section describes the channel specific registers of the HSSL module, associated to both the initiator and the target functions. In order to support command queues, the addresses of the IWDx, ICONx, IRWAx and IRDx registers are ordered in a sequence of IWD0, ICON0, IRWA0, IRD0, IWD1, ICON1, IRWA1, IRD1, IWD2, ICON2, IRWA2, IRD2, IWD3, ICON3, IRWA3, IRD3, see chapter "DMA Operated Command Queues".

For the timeout handling description, see chapter "Command Timeout Operation".

Related information

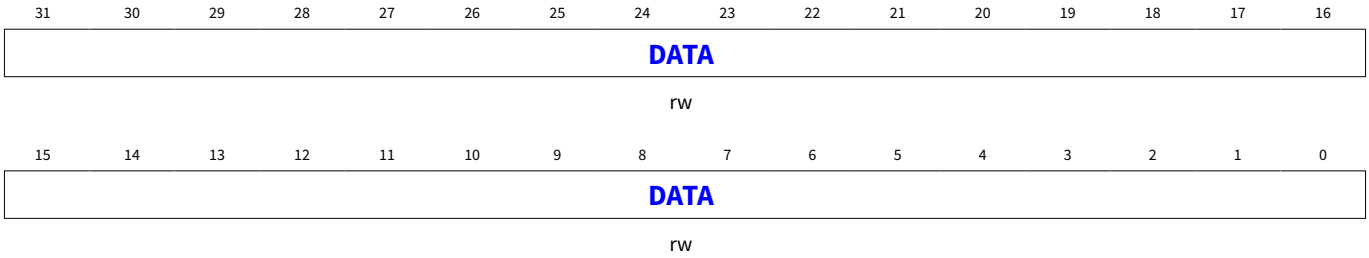
DMA operated command queues on page 2465
Command timeout operation on page 2456

17.6.1.5.1 Initiator Write Data Register x

The IWD register contains the data part of a write command.

IWDx (x=0-3) Offset address: 080_H+x*10_H

Initiator Write Data Register x Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
DATA	31:0	rw	**Data Part of the Payload of a Write Frame** For 8-bit and16-bit write command frames, the whole frame payload width of 32-bit is automatically filled with copies of the lower 8-bits or 16-bits of the register.

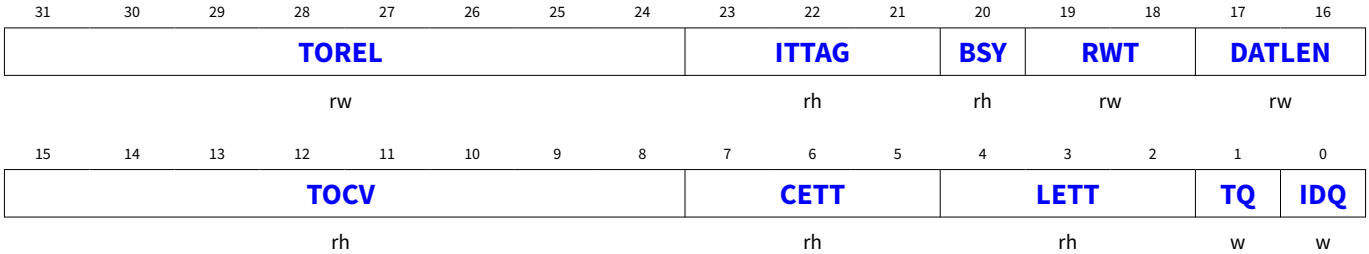
17.6.1.5.2 Initiator Control Data Register x

The ICON register contains the control and the configuration bits of a channel.

Bitfields TOREL and ITTAG are used for command frames, and also reused for streaming frames in channel 2. All other bit fields are used only for command frames.

ICONx (x=0-3) Offset address: 084_H+x*10_H

Initiator Control Data Register x Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
IDQ	0	w	**Read ID Request** This bit provides the only way to request a read ID frame. Reads always 0. Write of 1 commences a request. In case of parallel write of 1 to TQ and IDQ, the IDQ request has higher priority. 0_B No read ID request 1_B Read ID frame request
TQ	1	w	**Trigger Request** This bit provides an alternative way to request a trigger frame, without having to write to the channel address register. Reads always 0. Write of 1 commences a request. In case of parallel write of 1 to TQ and IDQ, the IDQ request has higher priority. 0_B No trigger request 1_B Trigger frame request
LETT	4:2	rh	**Last Error Transaction Tag** This bitfield indicates the latest erroneous transaction tage value.
CETT	7:5	rh	**Currently Expected Transaction Tag** This bitfield indicates the currently expected transaction tag value.
TOCV	15:8	rh	**Time Out Current Value** This bitfield indicates the timeout timer current value.
DATLEN	17:16	rw	**Data Length** Defines the length of the data in bits of the write and read command. 00_B 8-bit 01_B 16-bit 10_B 32-bit 11_B Reserved (implemented as 32-bit)
RWT	19:18	rw	**Read Write Trigger Command Type** Defines if the write to the IRWA register will trigger read, write or trigger request. 00_B No action 01_B Read Frame 10_B Write Frame 11_B Trigger Frame
BSY	20	rh	**Channel Busy** This bit indicates if the channel is busy or not.
ITTAG	23:21	rh	**Initiator Transaction Tag** This bit displays the current value of the three bit counter generating the new transaction tag value.
TOREL	31:24	rw	**Time Out Reload Value** Defines the duration of the timeout in units of prescaled clock periods. This parameter is valid both in command and stream mode of the channel 2.

17.6.1.5.3 Initiator Read Write Address Register x

The IRWA register contains the address part of a write command. Writing the IRWA register triggers a transmit request for the appropriate channel.

IRWAx (x=0-3) Offset address: 088_H+x*10_H

Initiator Read Write Address Register x Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADDRESS															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRESS															
rw															

Field	Bits	Type	Description
ADDRESS	31:0	rw	**Address Part of the Payload of a Write Frame** Writing to this registers triggers transmission of a Write Frame. The address must be aligned according to the data width: byte addresses for 8-bit data, half-word addresses for 16-bit data, word addresses for 32-bit data.

17.6.1.5.4 Initiator Read Data Register x

The IRD register contains the data read from the target, which has been fed back as a response to a read command.

IRDx (x=0-3) Offset address: $08C_H+x*10_H$

Initiator Read Data Register x Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DATA															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA															
rh															

Field	Bits	Type	Description
DATA	31:0	rh	**Data Delivered by a Read Response Frame** This bitfield indicates the data delivered by a read response frame.

17.6.1.6 Channel Target Registers

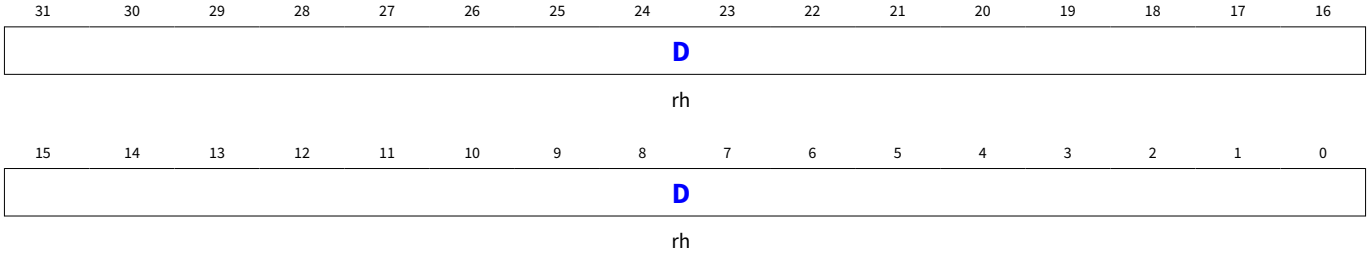
This section describes the channel specific registers of the HSSL module, associated to both the initiator and the target functions.

17.6.1.6.1 Target Current Data Register i

The TCD register contains the data part of a write command which is currently being processed by the channel on the target side, or the latest data which has been read by the SRI/SPB master in case of a read command.

TCDi (i=0-3) Offset address: $0C0_H+i*8$

Target Current Data Register i Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
D	31:0	rh	**Data Part of a WR Command Frame or RD Data of a RD Command Frame** This bitfield indicates the data part of the payload of a write command frame or read data of a read command frame.

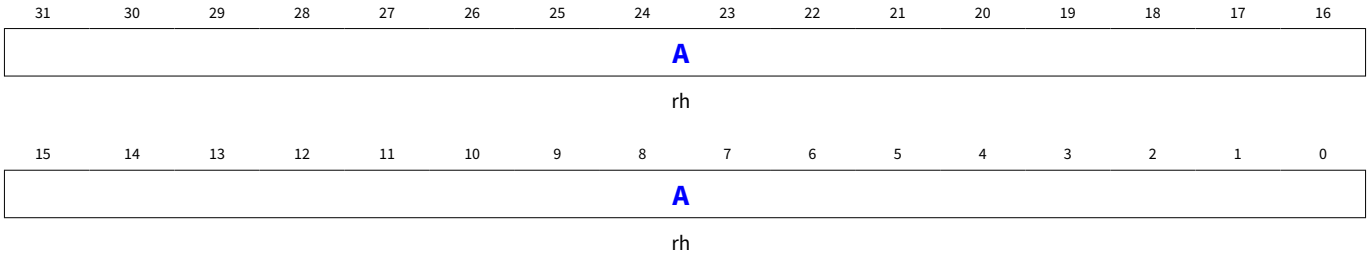
17.6.1.6.2 Target Current Address Register i

The TCA register contains the address part of a read or write command which is currently being processed by the channel on the target side.

In case of Read ID frame, the address is copied from the register TIDADD.

TCAi (i=0-3) Offset address: $0C4_H+i*8$

Target Current Address Register i Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
A	31:0	rh	**Address Part of a Write or a Read Command or ID Frame - of payload** This bitfield indicates the address part of the payload of a write command frame or a read command frame or ID frame.

17.6.1.6.3 Target Status Register

The TSTAT register contains the status bits of the common target functionality.

TSTAT Offset address: $0E0_H$

Target Status Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LASTTT3			LASTCC3					LASTTT2			LASTCC2				
rh			rh					rh			rh				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LASTTT1			LASTCC1					LASTTT0			LASTCC0				
rh			rh					rh			rh				

Field	Bits	Type	Description
LASTCCx (x=0-3)	8*x+4:8*x	rh	**Last Command Code** Indicates the latest command code from the header for the corresponding channel.
LASTTTx (x=0-3)	8*x+7:8*x+5	rh	**Last Transaction Tag** Indicates the transaction tag of the latest command or stream frame for the corresponding channel.

17.6.1.6.4 Target ID Address Register

The TIDADD register contains the address from which the target JTAG ID is fetched. The application software should write this register after any reset with this address of CBS_JTAGID register or some other customer defined address value.

TIDADD Offset address: $0E4_H$

Target ID Address Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
A															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
A															
rw															

Field	Bits	Type	Description
A	31:0	rw	**Address Pointer** Address pointer containing the address of the memory location containing the unique ID data.

17.6.1.7 Initiator Stream Registers

Related information

Memory block transfer modes of the stream channel on page 2467

17.6.1.7.1 Initiator Stream Start Address Register x

ISSAx (x=0-1) Offset address: $0F0_H$+x*4

Initiator Stream Start Address Register x Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
START															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
START											0				
rw											r				

Field	Bits	Type	Description
START	31:5	rw	**Start Address for the Memory Range** Aligned on 256-bit limit (stream frame payload size).
0	4:0	r	**Reserved** Read as 0; should be written with 0

17.6.1.7.2 Initiator Stream Current Address Register

ISCA Offset address: $0F8_H$

Initiator Stream Current Address Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CURR															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CURR											0				
rh											r				

Field	Bits	Type	Description
CURR	31:5	rh	**Address of the Memory Location for the Current Transfer** Aligned on 256-bit limit (stream frame payload size).
0	4:0	r	**Reserved** Read as 0; should be written with 0

17.6.1.7.3 Initiator Stream Frame Count Register

ISFC Offset address: $0FC_H$

Initiator Stream Frame Count Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CURCOUNT															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RELCOUNT															
rw															

Field	Bits	Type	Description
RELCOUNT	15:0	rw	**Reload Count Number** Contains the number of frames to transfer per memory block. Bit field length depends on application requirements. 0000_H 1 0001_H 1 0002_H 2 0003_H 3 **others,** corresponding number of frames
CURCOUNT	31:16	rh	**Current Count Number** Displays the current count number, which is generated by down-counting from the RELCOUNT value. Bit field length depends on application requirements.

17.6.1.8 Target Stream Registers

Related information

Memory block transfer modes of the stream channel on page 2467

17.6.1.8.1 Target Stream Start Address Register x

TSSAx (x=0-1) Offset address: 100_H+x*4

Target Stream Start Address Register x Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADDR															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR											0				
rw											r				

Field	Bits	Type	Description
ADDR	31:5	rw	**Start Address for the Memory Range** Aligned on 256-bit (or 32 byte) limit (stream frame payload size).
0	4:0	r	**Reserved** Read as 0; should be written with 0

17.6.1.8.2 Target Stream Current Address Register

TSCA Offset address: 108_H

Target Stream Current Address Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CURR															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CURR											0				
rh											r				

Field	Bits	Type	Description
CURR	31:5	rh	**Address of the Memory Location for the Current Transfer** Aligned on 256-bit (or 32 byte) limit (stream frame payload size).
0	4:0	r	**Reserved** Read as 0; should be written with 0

17.6.1.8.3 Target Stream Frame Count Register

TSFC Offset address: $10C_H$

Target Stream Frame Count Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CURCOUNT															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RELCOUNT															
rw															

Field	Bits	Type	Description
RELCOUNT	15:0	rw	**Reload Count Number** Contains the number of frames to transfer per memory block. Bit field length depends on application requirements. 0000_H 1 0001_H 1 0002_H 2 0003_H 3 **others,** corresponding number of frames
CURCOUNT	31:16	rh	**Current Count Number** Displays the current count number, which is generated by down-counting from the RELCOUNT value. Bit field length depends on application requirements.

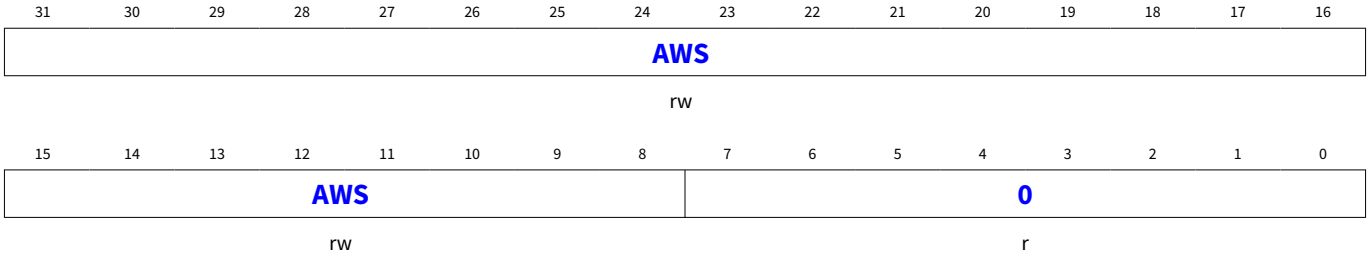
17.6.1.9 Access Protection Registers

This sub section describes the registers defining the access windows where the HSSL accesses are allowed. These registers are writable only in supervisor mode, which makes them not writable by the HSSL itself.

17.6.1.9.1 Access Window Start Register i

AWSTARTi (i=0-3) Offset address: 110_H+i*8

Access Window Start Register i Kernel Reset value: $0000\ 0000_H$

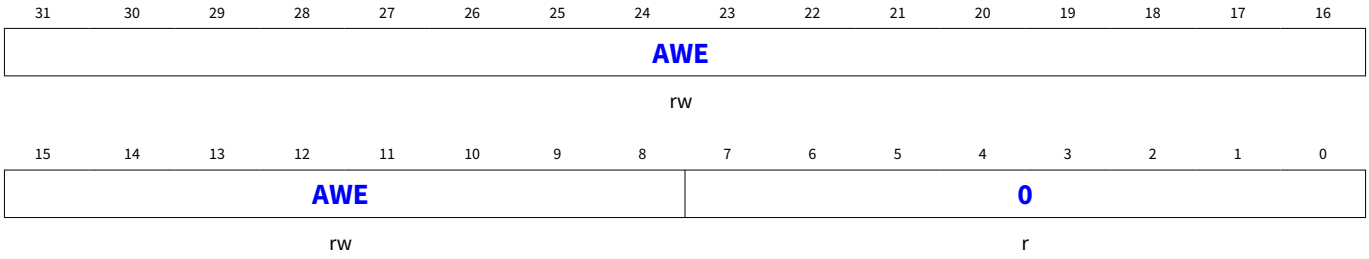


Field	Bits	Type	Description
AWS	31:8	rw	**Access Window Start Address** This bit field defines the upper 24 bits of the start address of the corresponding access window. This results in a granularity of 256 bytes for the start address.
0	7:0	r	**Reserved** Read as 0; should be written with 0

17.6.1.9.2 Access Window End Register i

AWENDi (i=0-3) Offset address: 114_H+i*8

Access Window End Register i Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
AWE	31:8	rw	**Access Window End Address** This bit field defines the upper 24 bits of the end address of the corresponding access window. This results in a granularity of 256 bytes for the end address.
0	7:0	r	**Reserved** Read as 0; should be written with 0

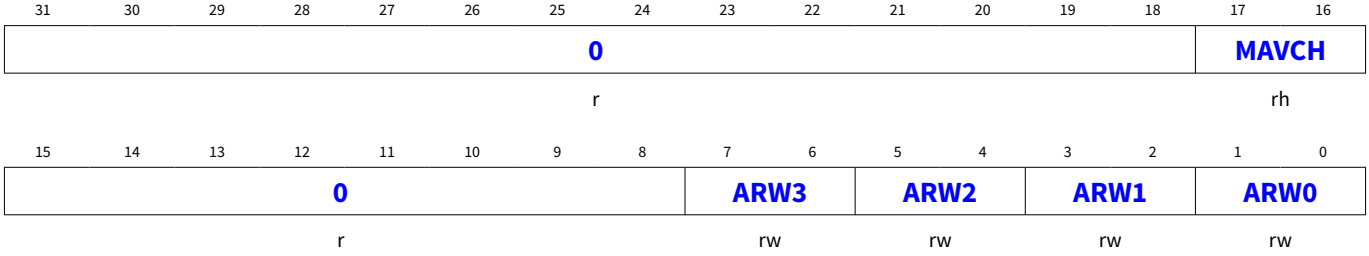
17.6.1.9.3 Access Rules Register

AR

Access Rules Register

Offset address: 130_H

Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
ARWx (x=0-3)	2*x+1:2*x	rw	**Access Rule for Window x** 00_B No access (window disabled) 01_B Read access allowed 10_B Write access allowed 11_B Read and write access allowed
MAVCH	17:16	rh	**Memory Access Violation Channel** This bit field shows the number of the latest channel that attempted a not allowed access. 00_B Channel 0 01_B Channel 1 10_B Channel 2 11_B Channel 3
0	15:8, 31:18	r	**Reserved** Read as 0; should be written with 0

17.6.1.10 Security and Multi Slave Registers

This sub section describes two registers: one handling the security aspects regarding the HSSL module operation and one configuring the multi slave mode of operation.

17.6.1.10.1 Security Control Register

SEC Offset address: $0E8_H$

Security Control Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0														LAW	LCK
r														rw	rw

Field	Bits	Type	Description
LCK	0	rw	**Lock the HSSL Module** Setting this bit field prevents the MFLAGS.INI bit to be cleared, that is prevents to leave the INIT state and go to RUN mode. If INIT bit has already been cleared, it is possible to set it, but not to clear it afterwards. 0_B Unlocked 1_B Locked
LAW	1	rw	**Lock the Address Windows Registers** This bit field shows if the Address Window Registers AWSTARTx, AWENDx, AR and TIDADD are locked or not. If locked, the properties of the address windows cannot be changed any more. 0_B Unlocked 1_B Locked
0	31:2	r	**Reserved** Read as 0; should be written with 0

17.6.1.10.2 Multi Slave Control Register

MSCR Offset address: $0EC_H$

Multi Slave Control Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															ITXSTOP
r															rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0													SLAVETAG		EN
r													rw		rw

Field	Bits	Type	Description
EN	0	rw	**Multi Slave Mode Enable** This bit enables the multi slave mode of operation of the HSSL module. 0_B Disabled 1_B Enabled
SLAVETAG	2:1	rw	**Slave Tag** The master uses this bit field to define the current slave it sends to and receives from. The slave uses this bit field to pass through only the relevant incoming frames and to tag its own transmit frames. This tag is injected in the header by the Slave Tag Translator block and used for filtering in Slave Tag Filter Block. 00_B Reserved 01_B Slave 1 10_B Slave 2 11_B Slave 3
ITXSTOP	16	rw	**Initiator Transmission Stop** This bit is intended to be set or cleared by the master in a multi-slave operation by using 16-bit write command frames. MSCR.ITXSTOP functionality can be activated only if MSCR.EN = 1, otherwise it is ignored. Setting this bit stops the arbitration and transmission of new command and stream frames of the initiator but does not stop the ongoing frames. 0_B Initiator transmission active 1_B Initiator transmission stopped
0	15:3, 31:17	r	**Reserved** Read as 0; should be written with 0

17.6.1.11 UBS Registers

17.6.1.11.1 Clock Control Register

The Clock Control Register CLC allows the programmer to adapt the functionality and power consumption of the module to the requirements of the application.

Register CLC controls the module clock signal and the reactivity to the sleep signal.

CLC Offset address: 000_H

Clock Control Register Application Reset value: $0000\ 0003_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												EDIS	0	DISS	DISR
r												rw	r	rh	rw

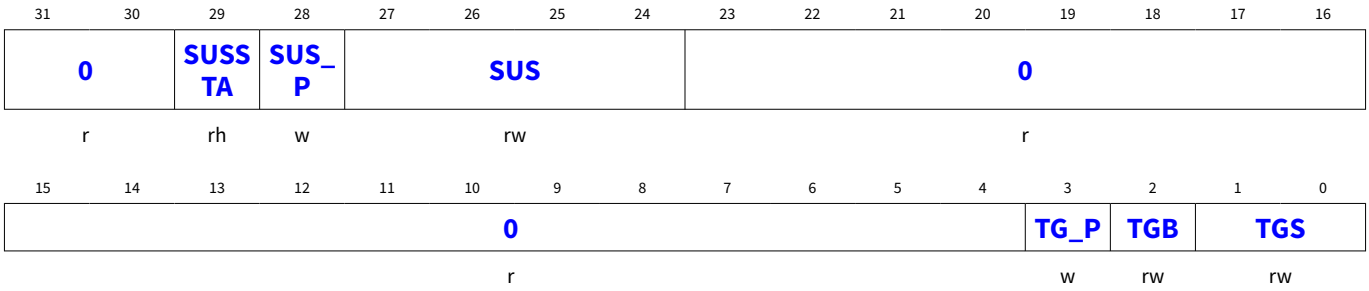
Field	Bits	Type	Description
DISR	0	rw	**Module Disable Request Bit** Used for enable/disable control of the module. 0_B On request: enable the module clock 1_B Off request: stop the module clock
DISS	1	rh	**Module Disable Status Bit** 0_B Module clock is enabled 1_B Off: module is not clocked
EDIS	3	rw	**Sleep Mode Enable Control** Used to control the module's reaction to sleep mode. 0_B Sleep mode request is enabled and functional 1_B Module disregards the sleep mode control signal
0	2, 31:4	r	**Reserved** Read as 0; should be written with 0

17.6.1.11.2 OCDS Control and Status Register

The OCDS Control and Status register OCS controls the debug and trace behavior by selecting suspend modes and OTGB trigger sets. When OCDS is disabled the suspend control is ineffective.

OCS Offset address: 004_H

OCDS Control and Status Register Debug Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
TGS	1:0	rw	**Trigger Set for OTGB0/1** 00_B No Trigger Set output 01_B TS16_STR, Streaming Channel Trigger Set 10_B TS16_ERR, Errors Trigger Set 11_B Reserved
TGB	2	rw	**OTGB0/1 Bus Select** 0_B Trigger Set is output on OTGB0 1_B Trigger Set is output on OTGB1
TG_P	3	w	**TGS, TGB Write Protection** TGS and TGB are only written when TG_P is 1, otherwise unchanged. Read as 0.
SUS	27:24	rw	**OCDS Suspend Control** Controls the sensitivity to the suspend signal coming from the OCDS Trigger Switch (OTGS) Not listed combinations are reserved. 0_H Will not suspend 1_H Hard suspend: Clocks will be disabled immediately 2_H reserved 3_H Soft suspend mode
SUS_P	28	w	**SUS Write Protection** SUS is only written when SUS_P is 1, otherwise unchanged. Read as 0.
SUSSTA	29	rh	**Suspend State** 0_B Module is not (yet) suspended 1_B Module is suspended
0	23:4, 31:30	r	**Reserved** Read as 0; should be written with 0

Table 797 Access mode restrictions of OCS sorted by descending priority

Mode name	Access mode		Description
write 1 to .**TG_P**	rw	TGB, TGS	Set TG_P during write access
write 1 to .**SUS_P**	rw	SUS	Set SUS_P during write access
(default)	r	SUS, TGB, TGS	

17.6.1.11.3 Reset Control Register A

RST_CTRLA Offset address: $00C_H$

Reset Control Register A Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				GRSTEN3	GRSTEN2	GRSTEN1	GRSTEN0	0							KRST
r				rw	rw	rw	rw	r							rwh

Field	Bits	Type	Description
KRST	0	rwh	**Kernel Reset** Request a kernel reset. The requested reset is executed if CTRLA.KRST=1 and CTRLB.KRST=1. KRST is cleared after the kernel reset was executed. 0_B No action 1_B A kernel reset was requested
GRSTENx (x=0-3)	x+8	rw	**Enable for Global Module Reset Group x** 0_B global module reset group x does not have any effect 1_B global module reset group x results in a kernel reset
0	7:1, 31:12	r	**Reserved** Read as 0; should be written with 0

17.6.1.11.4 Reset Control Register B

RST_CTRLB Offset address: 010_H

Reset Control Register B Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
STATCLR								0							
w								r							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															KRST
r															rwh

Field	Bits	Type	Description
KRST	0	rwh	**Kernel Reset** Request a kernel reset. The requested reset is executed if CTRLA.KRST=1 and CTRLB.KRST=1. KRST is cleared after the kernel reset was executed. 0_B No action 1_B A kernel reset was requested
STATCLR	31	w	**Kernel Reset Status Clear** Clears all status bits in RST_STAT when 1 is written. Read will return 0. 0_B No action 1_B Write with ´1´ clears bits STAT.GRSTx and bit STAT.KRST
0	30:1	r	**Reserved** Read as 0; should be written with 0

17.6.1.11.5 Reset Status Register

The reset status register contains the status bits for kernel reset (KRST) and the global module reset groups (GRSTx).

RST_STAT Offset address: 014_H

Reset Status Register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				GRST3	GRST2	GRST1	GRST0	0							KRST
r				rh	rh	rh	rh	r							rh

Field	Bits	Type	Description
KRST	0	rh	**Kernel Reset Status** Request a kernel reset. The requested reset is executed if CTRLA.KRST=1 and CTRLB.KRST=1. KRST is cleared after the kernel reset was executed.
GRSTx (x=0-3)	x+8	rh	**Status for Global Module Reset Group x** 0_B Reset was not triggered by global reset group x 1_B Reset was triggered by global reset group x
0	7:1, 31:12	r	**Reserved** Read as 0; should be written with 0

17.6.1.11.6 PROT Register Endinit

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner and in some PROT states, the secure master to update the PROT state as described in the PROT mechanism.

The PROTE register controls lock / unlock of the local Endinit (E) protected control registers.

PROTE Offset address: 020_H

PROT Register Endinit Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OWEN	ODEF	TAGID						PRSEN	PRS			VMEN	VM		
w	rw	rw						rw	rw			rw	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												SWEN	STATE		
r												w	rwh		

Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
VM	18:16	rw	**Virtual machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual machine definition enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.

(table continues...)

(continued)

Field	Bits	Type	Description
PRSEN	23	rw	**Protection set definition enable for PROT owner** 0_B **Disable**: PRS is not part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG-ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; should be written with 0.

Table 798 Access mode restrictions of PROTE sorted by descending priority

Mode name	Access mode		Description
write 1 to .**SWEN**	rwh	STATE	
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

17.6.1.11.7 PROT Register Safe Endinit

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner and in some PROT states, the secure master to update the PROT state as described in the PROT mechanism.

The PROTSE register controls lock / unlock of the local Safe Endinit (SE) protected control registers.

PROTSE Offset address: 024_H

PROT Register Safe Endinit Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OWEN	**ODEF**	**TAGID**						**PRSEN**	**PRS**			**VMEN**	**VM**		
w	rw	rw						rw	rw			rw	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												**SWEN**	**STATE**		
r												w	rwh		

Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
VM	18:16	rw	**Virtual machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual machine definition enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.
PRSEN	23	rw	**Protection set definition enable for PROT owner** 0_B **Disable**: PRS is not part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG-ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; should be written with 0.

Table 799 Access mode restrictions of PROTSE sorted by descending priority

Mode name	Access mode		Description
write 1 to .**SWEN**	rwh	STATE	
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

17.6.1.11.8 Write access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

ACCEN_WRA Offset address: 040_H

Write access enable register A Application Reset value: $1000\ 0003_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	**EN30**	**EN29**	**EN28**	**EN27**	**EN26**	**EN25**	**EN24**	**EN23**	**EN22**	**EN21**	**EN20**	**EN19**	**EN18**	**EN17**	**EN16**
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	**EN14**	**EN13**	**EN12**	**EN11**	**EN10**	**EN09**	**EN08**	**EN07**	**EN06**	**EN05**	**EN04**	**EN03**	**EN02**	**EN01**	**EN00**
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access

17.6.1.11.9 Write access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

ACCEN_WRB Offset address: 044_H

Write access enable register B Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								**EN39**	**EN38**	**EN37**	**EN36**	**EN35**	**EN34**	**EN33**	**EN32**
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access
0	31:8	r	**Reserved** Read as 0; should be written with 0.

17.6.1.11.10 Read access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

ACCEN_RDA Offset address: 048_H

Read access enable register A Application Reset value: FFFF $FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access

17.6.1.11.11 Read access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

ACCEN_RDB Offset address: $04C_H$

Read access enable register B Application Reset value: 0000 $00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access
0	31:8	r	**Reserved** Read as 0; should be written with 0.

17.6.1.11.12 VM access enable register

This register defines which virtual machine encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each VMx.

ACCEN_VM Offset address: 050_H

VM access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for VM ID q** This bit enables read access to the access protected region for transactions with the VM ID q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for VM ID q** This bit enables write access to the access protected region for transactions with the VM ID q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

17.6.1.11.13 PRS access enable register

This register defines which protection register set encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each PRSx.

ACCEN_PRS Offset address: 054_H

PRS access enable register Application Reset value: $00FF\ 00FF_H$

17 High Speed Serial Link (HSSL)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for PRS q** This bit enables read access to the access protected region for transactions with the PRS q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for PRS q** This bit enables write access to the access protected region for transactions with the PRS q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

17.6.2 HSCT registers

17.6.2.1 Register overview - access mode glossary

Table 800 Register overview - access mode glossary

Keyword	Description
E	Access protection using PROT register PROTE.
SE	Access protection using PROT register PROTSE.
APU-P	Protection group consisting of registers ACCEN_WRA, ACCEN_WRB, ACCEN_RDA, ACCEN_RDB, ACCEN_VM, ACCEN_PRS.
P	Access protection using *APU-P* registers.
BE	Always returns a Bus Error.
U	No access restrictions.
PROT	Access restrictions as defined in the PROT register access rules.

17.6.2.2 Registers overview - HSCT (ascending offset address)

Table 801 Registers overview - HSCT (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CLC	Clock Control Register	00000_H	P	P, SV, E	Application Reset	2574
OCS	OCDS Control and Status Register	00004_H	P	SV, P	Debug Reset	2574
ID	Module Identification Register	00008_H	P	BE	PowerOn Reset	2581
RST_CTRLA	Reset Control Register A	$0000C_H$	P	P, SV, E	Application Reset	2576
RST_CTRLB	Reset Control Register B	00010_H	P	P, SV, E	Application Reset	2576
RST_STAT	Reset Status Register	00014_H	P	BE	Application Reset	2577
PROTE	Register Endinit	00020_H	U	SV, PROT	Application Reset	2578
PROTSE	PROT Register Safe Endinit	00024_H	U	SV, PROT	Application Reset	2579
ACCEN_WRA	Write access enable register A	00040_H	U	SE, SV	Application Reset	2581
ACCEN_WRB	Write access enable register B	00044_H	U	SE, SV	Application Reset	2582
ACCEN_RDA	Read access enable register A	00048_H	U	SE, SV	Application Reset	2582

(table continues...)

Table 801 (continued) Registers overview - HSCT (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ACCEN_RDB	Read access enable register B	$0004C_H$	U	SE, SV	Application Reset	2583
ACCEN_VM	VM access enable register	00050_H	U	SE, SV	Application Reset	2583
ACCEN_PRS	PRS access enable register	00054_H	U	SE, SV	Application Reset	2584
INIT	Initialization Register	00060_H	P	P	Kernel Reset	2557
IFCTRL	Interface Control Register	00064_H	P	P	Kernel Reset	2559
CTSCTRL	Clear To Send Control Register	$0006C_H$	P	P	Kernel Reset	2561
DISABLE	Transmission Disable Register	00070_H	P	P	Kernel Reset	2562
STAT	Status Register	00074_H	P	P	Kernel Reset	2563
IFSTAT	Interface Status Register	00078_H	P	P	Kernel Reset	2564
CONFIGPHY	Configuration Physical Layer Register	$0007C_H$	P	P	Kernel Reset	2564
STATPHY	STATPHY	00080_H	P	P	Kernel Reset	2565
IRQ	Interrupt register	00084_H	P	P	Kernel Reset	2566
IRQEN	Interrupt Enable Register	00088_H	P	P	Kernel Reset	2568
IRQCLR	Interrupt Clear Register	$0008C_H$	P	P	Kernel Reset	2570
USMR	Unsolicited Status Message Received	00090_H	P	P	Kernel Reset	2571
USMS	Unsolicited Status Message Send	00094_H	P	P	Kernel Reset	2572
TESTCTRL	Test Control Register	00098_H	P	SV, P	Kernel Reset	2572

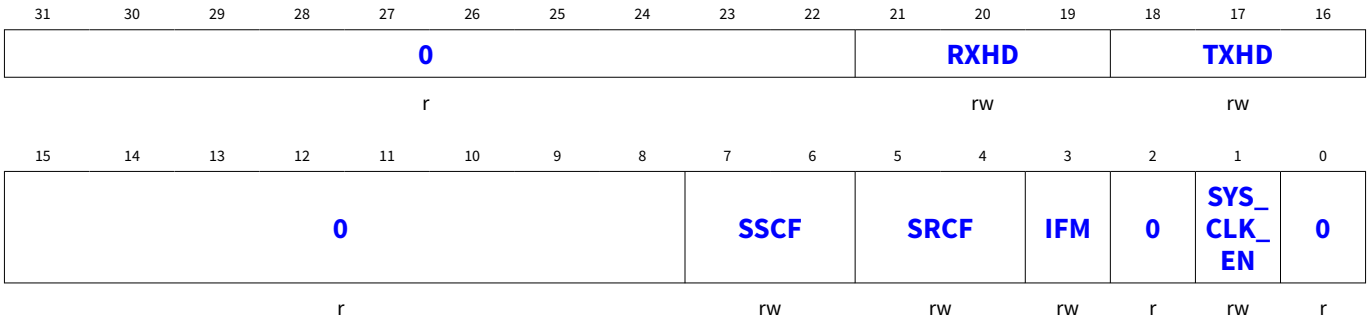
17.6.2.3 HSCT Kernel Register Definitions

17.6.2.3.1 Initialization Register

INIT Offset address: 00060_H

Initialization Register Kernel Reset value: $0000\ 0000_H$

17 High Speed Serial Link (HSSL)



Field	Bits	Type	Description
SYS_CLK_EN	1	rw	**Enable HSCT SysClk in Master interface** SysClk enable activates the SysClk. This feature is only available in the Master interface. In Slave interface mode the register setting does not have an effect. 0_B Disable (default) 1_B Enabled
IFM	3	rw	**Select Interface Mode** Select the Interface Mode (Master IF or Slave IF). 0_B Master IF 1_B Slave IF
SRCF	5:4	rw	**Select Reference Clock Frequency Divider** Defines physical layer reference frequency (PHY_CLK) and respective input frequency divider. The configuration is valid for Low speed and Medium Speed mode. 00_B REFCLK 10MHz, LS 5MBaud, MS n.a. 01_B REFCLK 20MHz, LS 5MBaud, MS=20MBaud 01_B REFCLK 25MHz, LS 6.25MBaud, MS=n.a. 10_B REFCLK 40MHz, LS 5MBaud, MS=20MBaud 10_B REFCLK 50MHz, LS 6.25MBaud, MS=n.a. 11_B Reserved; For future use
SSCF	7:6	rw	**Select SysClk Frequency Divider** For master interface defines SysClk pad output frequency. The allowed SysClk frequency is 10MHz or 20MHz or 25MHz. 00_B SYSCLK DIV 1/1 01_B SYSCLK DIV 1/2 10_B SYSCLK DIV 1/4 11_B For future use

(table continues...)

(continued)

Field	Bits	Type	Description
TXHD	18:16	rw	**Transmit High Speed Divider.** The Transmit High Speed data rate can be reduced by dividing factors. The Transmit High Speed data rate is separated from the Receive High Speed data rate. 000_B Divider 1/1 001_B Divider 1/2 010_B Divider 1/4 011_B Divider 1/8 100_B Divider 1/16 101_B For future use. … 111_B For future use.
RXHD	21:19	rw	**Receive High Speed Divider.** The Receive High Speed data rate can be reduced by dividing factors. The Receive High Speed data rate is separated from the Transmit High Speed data rate. ***Note**: For future use configuration leads to no output clock delivered to the correlator.* 000_B Divider 1/1 001_B Divider 1/2 010_B Divider 1/4 011_B Divider 1/8 100_B Divider 1/16 101_B For future use. … 111_B For future use.
0	0, 2, 15:8, 31:22	r	**Reserved** Read as 0; should be written with 0

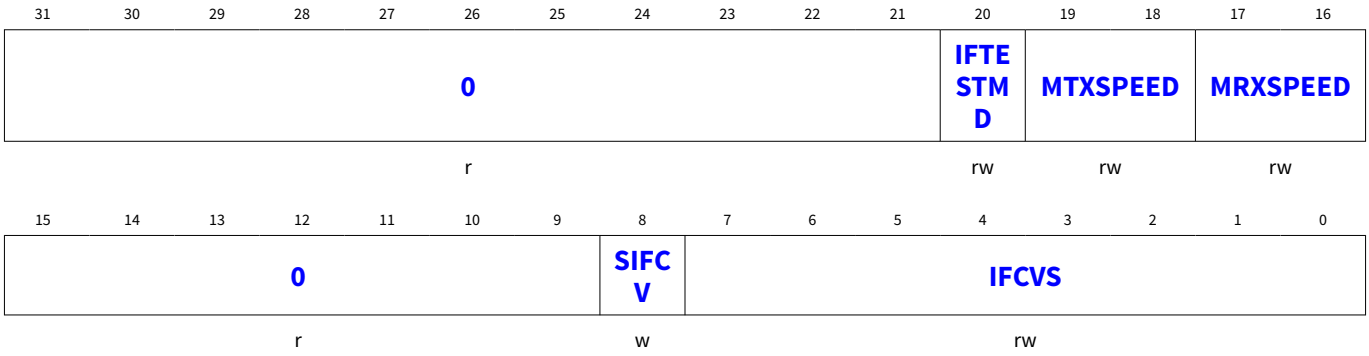
17.6.2.3.2 Interface Control Register

This register allows to configure the transfer Speed of the Master Interface and is also used to send the command frame to the Slave.

***Note**: The Master interface link speed configuration immediately takes place. Therefore the Master interface speed transition must not be changed with a Slave interface command send activity.*

IFCTRL Offset address: 00064_H

Interface Control Register Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
IFCVS	7:0	rw	**Master Mode - Trigger for Interface Control Value to be send to Slave interface** See the table "Interface Control Payload Values". Master IF Mode: The value is taken as control frame value send as payload to the Slave IF. Slave IF Mode: The value is a new configuration of the Slave IF (not recommended flow!!!).
SIFCV	8	w	**Master Mode - Slave IF control frame trigger** Changing the interface configuration, software must guarantee not having transfers active on the interface. This register bit always reads back zero. 0_B The IFCVS field configured value has no effect (default). 1_B Writing a one to the register bit sends the control frame configured in register field IFCVS, if the interface is configured as Master. In Slave Mode the trigger has an effect and takes the IFCTRL.IFCVS value. In Slave mode this is not the recommended control flow. A frame based configuration shall be used instead. A control frame has higher priority then a register control. At a simultaneous occurrence of both configuration source the register configuration in Slave mode is lost.
MRXSPEED	17:16	rw	**Master Mode RX speed** ***Note**: Register setting only valid in interface Master mode.* 00_B Low Speed 01_B Medium Speed 10_B High Speed 11_B For future use
MTXSPEED	19:18	rw	**Master Mode TX speed** ***Note**: Register setting only valid in interface Master mode.* 00_B Low Speed 01_B for future use 10_B High Speed 11_B For future use

(table continues...)

(continued)

Field	Bits	Type	Description
IFTESTMD	20	rw	**Interface TX Test Mode** 0_B Test Mode disabled 1_B Test Mode enabled - send out 101010101… test pattern continuously.
0	15:9, 31:21	r	**Reserved** Read as 0; should be written with 0

17.6.2.3.3 Clear To Send Control Register

The Clear To Send function is not available in multi Slave scenario. This function is an automatic HW controlled function and in multi Slave mode not easy to control. Therefore it is removed in this mode.

CTSCTRL Offset address: $0006C_H$

Clear To Send Control Register Kernel Reset value: $0000\ 0001_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												HSSL_CTS_FBD	CTS_RXD	CTS_TXD	CTS_FRAME
r												rw	rw	rw	rw

Field	Bits	Type	Description
CTS_FRAME	0	rw	**Transmit CTS Frame Generation** Dedicated CTS frames are generated after the receive data path is not able to accept more data. The situation is indicated by the CTS header bit in case there is currently no data to be transferred. 0_B Generation of dedicated CTS frames disabled. 1_B Generation of dedicated CTS frames enabled.
CTS_TXD	1	rw	**Disable TX CTS signaling** If this bit is set to 1, CTS signaling is not performed at the interface and the status remains at the clear to send for every frame send. 0_B Enable CTS signaling (default). 1_B Disable CTS signaling.
CTS_RXD	2	rw	**Disable RX CTS detection** If this bit is set to 1, CTS detection is not performed at the receiver and the status remains internally at clear to send for every frame received. 0_B Enable CTS detection (default). 1_B Disable CTS detection.

(table continues...)

(continued)

Field	Bits	Type	Description
HSSL_CTS_FBD	3	rw	**Disable HSSL interface CTS Frame Blocking** If this bit is set to 1, CTS signaling is not performed at the interface and the status remains at the clear to send for every frame send. 0_B Enable CTS frame blocking (default). 1_B Disable CTS frame blocking.
0	31:4	r	**Reserved** Read as 0; should be written with 0

17.6.2.3.4 Transmission Disable Register

DISABLE Offset address: 00070_H

Transmission Disable Register Kernel Reset value: $0000\ 0003_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0													RX_HEPD	RX_DIS	TX_DIS
r													rw	rw	rw

Field	Bits	Type	Description
TX_DIS	0	rw	**Disable HSCT Transmit path in Master interface** Disable the transmit path of the HSCT interface. If this bit is set to 1 - no transfer can be initiated and the LVDS driver is disabled. 0_B Enable 1_B Disable
RX_DIS	1	rw	**Disable HSCT Receive path in Master interface** Disable the receive line path of the HSCT interface. If this bit is set to 1 - no transfer from the other side can be received and the Master RX path is in a low power state. This feature is only available in the Master interface. Slave interface RX path can not be disabled and a write to the register has no effect. 0_B Enable 1_B Disable
RX_HEPD	2	rw	**Disable RX Header Error Discard Payload data.** Instead of discarding the Payload data at a header error the payload data is passed to the higher layer (HSSL). Only channel data to HSSL is affected. This function is available in Master and in Slave mode. 0_B Header error received data is discarded 1_B Header error received data is passed to the higher hardware layer.
0	31:3	r	**Reserved** Read as 0; should be written with 0

17.6.2.3.5 Status Register

STAT Offset address: 00074_H

Status Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LIFCCMDR								0				TX_CHANNEL_TYPE			
rh								r				rh			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	TX_PSIZE			0			TX_SLEEP	RX_SLEEP	RX_CHANNEL				RX_PSIZE		
r	rh			r			rh	rh	rh				rh		

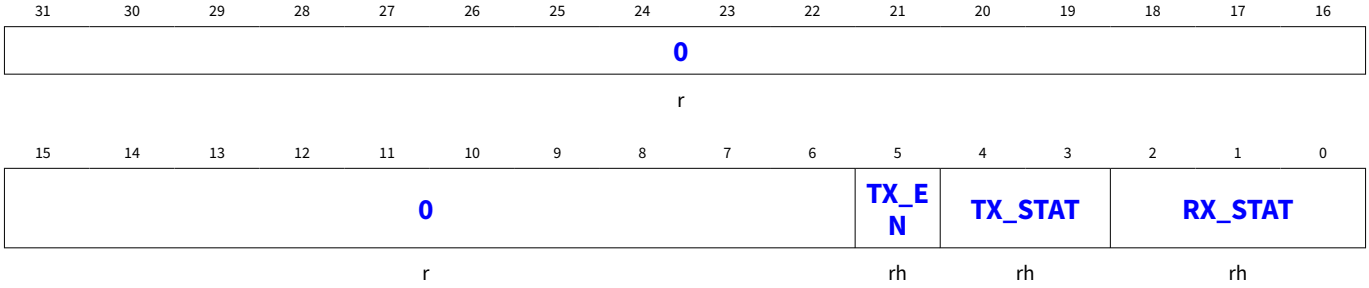
Field	Bits	Type	Description
RX_PSIZE	2:0	rh	**RX (Receiving) Payload Size** Contains the payload size of the last received frame.
RX_CHANNEL	6:3	rh	**RX (Receiving) Logical Channel Type** Contains the logical channel type of the last received frame. See Table "Logical Channel Type Coding".
RX_SLEEP	7	rh	**RX (Receiving) Sleep Mode Status** 0_B HSCT is in receive direction active 1_B HSCT is in receive direction in sleep mode
TX_SLEEP	8	rh	**TX (Transmission) Sleep Mode Status** 0_B HSCT is in transmit direction active 1_B HSCT is in transmit direction in sleep mode
TX_PSIZE	14:12	rh	**Transmission Payload Size** Coding of the logical channel type is according to the Table: Payload Size Coding. This value was used in the logical channel type field in the header for the actual transfer in transmit direction.
TX_CHANNEL_TYPE	19:16	rh	**Transmission Logical Channel Type** Coding of the logical channel type is according to the Table: Logical Channel Type Coding. This value was used in the logical channel type field in the header for the actual transfer in transmit direction.
LIFCCMDR	31:24	rh	**Last Interface Control Command Received** The bit value reflects the last control command received. The bit is active in Slave interface mode only. In Master mode it reflects logic 0 always.
0	11:9, 15, 23:20	r	**Reserved** Read as 0; should be written with 0

17.6.2.3.6 Interface Status Register

Clock test mode and Loopback mode are only available in single Slave scenario. In multi Slave scenario are not available.

IFSTAT Offset address: 00078_H

Interface Status Register Kernel Reset value: $0000\ 0000_H$

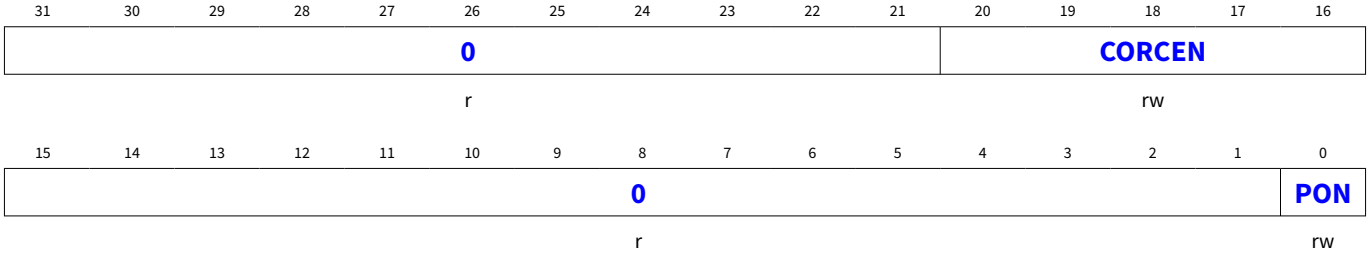


Field	Bits	Type	Description
RX_STAT	2:0	rh	**HSCT Slave interface Status for RX link** ***Note**: Slave interface transmitter only* 000_B Interface is disabled in RX link direction. 001_B Interface runs at low speed on RX link. 010_B Interface runs at medium speed on RX link. 011_B Interface runs at high speed on RX link direction. 100_B Reserved 101_B Clock test mode and low speed on RX ink 110_B Clock test mode and medium speed on RX link 111_B Clock test mode and high speed on RX link.
TX_STAT	4:3	rh	**HSCT Slave interface Status for TX link** ***Note**: Slave interface receiver only* 00_B Interface runs at low speed on TX link. 01_B Interface runs at high speed on TX link. 10_B Loopback mode low speed. 11_B Loopback mode high speed.
TX_EN	5	rh	**HSCT LVDS Slave interface TX enable** ***Note**: Slave interface only* 0_B LVDS TX disabled. 1_B LVDS TX enabled.
0	31:6	r	**Reserved** Read as 0; should be written with 0

17.6.2.3.7 Configuration Physical Layer Register

CONFIGPHY Offset address: $0007C_H$

Configuration Physical Layer Register Kernel Reset value: $001F\ 0000_H$

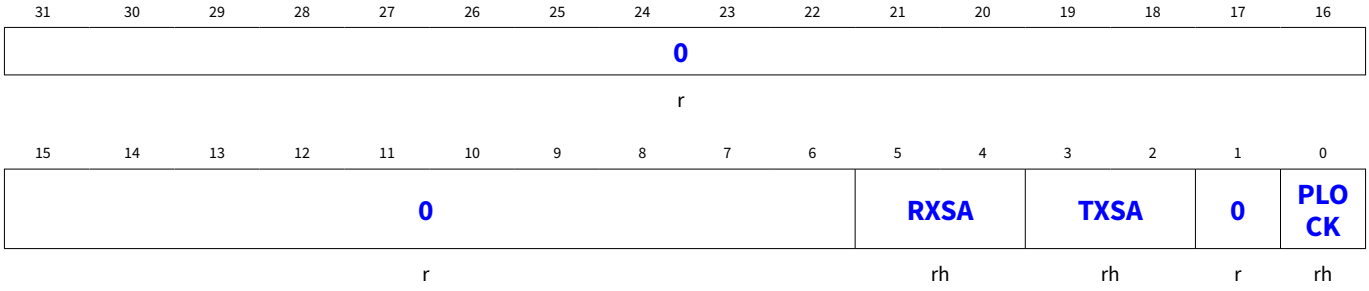


Field	Bits	Type	Description
PON	0	rw	**Physical Layer Power On.** This bit if set enables the physical layer.
CORCEN	20:16	rw	**Correlator phase enable - allows to en/dis each of the 5 Phase o/p separately.** This bit allows to enable/disable each of the 5 Phase outputs separately.
0	15:1, 31:21	r	**Reserved** Read as 0; should be written with 0

17.6.2.3.8 STATPHY

STATPHY Offset address: 00080_H

STATPHY Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
PLOCK	0	rh	**PLL locked** 0_B PLL out of lock (default) 1_B PLL locked
TXSA	3:2	rh	**Transmitter speed** 00_B Transmitter in Low speed 01_B Transmitter in Medium speed 10_B Transmitter in High speed 11_B reserved

(table continues...)

(continued)

Field	Bits	Type	Description
RXSA	5:4	rh	**Receiver speed** 00_B Receiver in Low speed 01_B Receiver in Medium speed 10_B Receiver in High speed 11_B reserved
0	1, 31:6	r	**Reserved** Read as 0; should be written with 0

17.6.2.3.9 Interrupt register

Interrupt status register. Read only and updated by HW. The bit's remain active until cleared. The clear pulse is coming from the IRQCLR register. Only an interrupt source not being already active in the interrupt status register generates an interrupt pulse to the SoC interrupt controller.

IRQ Offset address: 00084_H

Interrupt register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		MSCE	SFU	SFO	TXTE	PAR	USM	PLER	USMSF	SMER	IFCFS	CER	PYER	HER	0
r		rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	r

Field	Bits	Type	Description
HER	1	rh	**Header error detected** Not supported size: • Received command at Slave interface 8-bit size only - other command sizes generate an error. • Received command ping answer at Master interface 32-bit size only - other command sizes generate an error • Unsolicited data 32-bit only • Logical data channel size 8-bit Not supported logical channel type: • 0b1xxx • 0010 (Slave interface control and Slave interface read) 0_B Header OK 1_B Header issue detected.
PYER	2	rh	**Payload error detected** Payload does not fit the header size 0_B Payload size OK 1_B Received payload size wrong

(table continues...)

(continued)

Field	Bits	Type	Description
CER	3	rh	**HSCT command error** Control command not valid or control payload size bigger than 8-bit. Single Slave specific commands received in Multi Slave Mode do not trigger CER error, but only MSCE error. 0_B No command error 1_B HSCT command error
IFCFS	4	rh	**HSCT interface control frame send** The scheduled interface control command is send. 0_B No interface control command send 1_B Interface control command send.
SMER	5	rh	**Speed Mode Switch Error (Master Mode only)** Speed mode change did not work. Received PING payload not valid. 0_B Interfaces runs at defined speed 1_B Ping payload error
USMSF	6	rh	**Unsolicited message frame send finished** Interrupt is indicated after the unsolicited message send is finished. 0_B No unsolicited message send. 1_B Unsolicited message send finished.
PLER	7	rh	**PLL lost lock error** After the PLL locked, the PLL may loose lock, which is reflected by the error 0_B PLL lock 1_B PLL lock loss
USM	8	rh	**Unsolicited Message Received** Unsolicited message received indication. Unsolicited message indicates a system event to the other interface side. 0_B no unsolicited message available 1_B unsolicited message available
PAR	9	rh	**PING Answer Received** The received message was identified as PING. 0_B No PING message available 1_B PING message received
TXTE	10	rh	**TX transfer error occurred on a disabled TX channel.** A disabled TX triggers an error interrupt, if: • TX disabled on a pending or active data transfer. • TX CTS configuration change on a active CTS frame. 0_B No error situation occurred 1_B Error situation occurred

(table continues...)

(continued)

Field	Bits	Type	Description
SFO	11	rh	**Synchronization FIFO overflow (in RX direction)** Physical layer to Controller data synchronization FIFO in RX transfer direction hit an overflow situation. ***Note**: This interrupt is an indication about a to slow SRI clock compared to Physical layer clock, which results in a overflow situation.* 0_B RX synchronization is running well. 1_B RX FIFO overflow
SFU	12	rh	**Synchronization FIFO underflow (in TX direction)** Controller to Physical layer data synchronization FIFO in TX transfer direction hit an underflow situation. ***Note**: This interrupt is an indication about a to slow SRI clock compared to Physical layer clock, which results in a underflow situation.* 0_B TX synchronization is running well. 1_B TX FIFO underflow
MSCE	13	rh	**Multi Slave scenario Command Error** This interrupt indicates a control command which is not allowed in multi Slave scenario. In Master and Slave mode a not allowed command results to an error. 0_B No multi Slave scenario command error. 1_B Multi Slave scenario command error (not allowed command used)
0	0, 31:14	r	**Reserved** Read as 0; should be written with 0

17.6.2.3.10 Interrupt Enable Register

Interrupt enable register. An enabled register generates a pulsed interrupt. On a disabled interrupt the interrupt pulse does not come through on the interrupt line.

IRQEN Offset address: 00088_H

Interrupt Enable Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		MSCEEN	SFUEN	SFOEN	TXTEEN	PAREN	USMEN	PLEREN	USMSFEN	SMEREN	IFCFSEN	CEREN	PYEREN	HEREN	0
r		rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	r

Field	Bits	Type	Description
HEREN	1	rw	**Header error detected interrupt enable** 0_B Interrupt disabled 1_B Interrupt enabled
PYEREN	2	rw	**Payload error detected interrupt enable** 0_B Interrupt disabled 1_B Interrupt enabled
CEREN	3	rw	**HSCT command error interrupt enable** 0_B Interrupt disabled 1_B Interrupt enabled
IFCFSEN	4	rw	**HSCT interface control command send enable** 0_B Interrupt disabled 1_B Interrupt enabled
SMEREN	5	rw	**Speed Mode Switch Error interrupt enable** 0_B Interrupt disabled 1_B Interrupt enabled
USMSFEN	6	rw	**Unsolicited message frame send finished** 0_B Interrupt disabled 1_B Interrupt enabled
PLEREN	7	rw	**PLL lost lock error interrupt enable** 0_B Interrupt disabled 1_B Interrupt enabled
USMEN	8	rw	**Unsolicited Message received enable** 0_B Interrupt disabled 1_B Interrupt enabled
PAREN	9	rw	**PING Answer Received enable** 0_B Interrupt disabled 1_B Interrupt enabled
TXTEEN	10	rw	**TX disable error interrupt enable** 0_B Interrupt disabled 1_B Interrupt enabled
SFOEN	11	rw	**Synchronization FIFO overflow (in RX direction) interrupt enable** 0_B Interrupt disabled 1_B Interrupt enabled
SFUEN	12	rw	**Synchronization FIFO underflow (in TX direction) interrupt enable** 0_B Interrupt disabled 1_B Interrupt enabled
MSCEEN	13	rw	**Multi Slave scenario Command Error interrupt enable** 0_B Interrupt disabled 1_B Interrupt enabled

(table continues...)

(continued)

Field	Bits	Type	Description
0	0, 31:14	r	**Reserved** Read as 0; should be written with 0

17.6.2.3.11 Interrupt Clear Register

This register is the Interrupt clear register. By writing a logic 1 to the register the interrupt in the IRQ register is cleared by a clear pulse. This means the interrupt clear methodology is write one to clear. Reading from this register presents the logic value 0 to software.

IRQCLR Offset address: $0008C_H$

Interrupt Clear Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		MSCELRCLR	SFUCLR	SFOCLR	TXTECLR	PARCLR	USMCLR	PLERCLR	USMSFCLR	SMERCLR	IFCFSCLR	CERCLR	PYERCLR	HERCLR	0
r		w	w	w	w	w	w	w	w	w	w	w	w	w	r

Field	Bits	Type	Description
HERCLR	1	w	**Header error detected interrupt clear** 0_B No action 1_B Clear HER
PYERCLR	2	w	**Payload error detected interrupt clear** 0_B No action 1_B Clear PYER
CERCLR	3	w	**HSCT command error interrupt clear** 0_B No action 1_B Clear CER
IFCFSCLR	4	w	**HSCT interface control command send interrupt clear** 0_B No action 1_B Clear IFCFS
SMERCLR	5	w	**Speed Mode Switch Error interrupt clear** 0_B No action 1_B Clear SMER
USMSFCLR	6	w	**Unsolicited message frame send finished interrupt clear** 0_B No action 1_B Clear USMSF

(table continues...)

(continued)

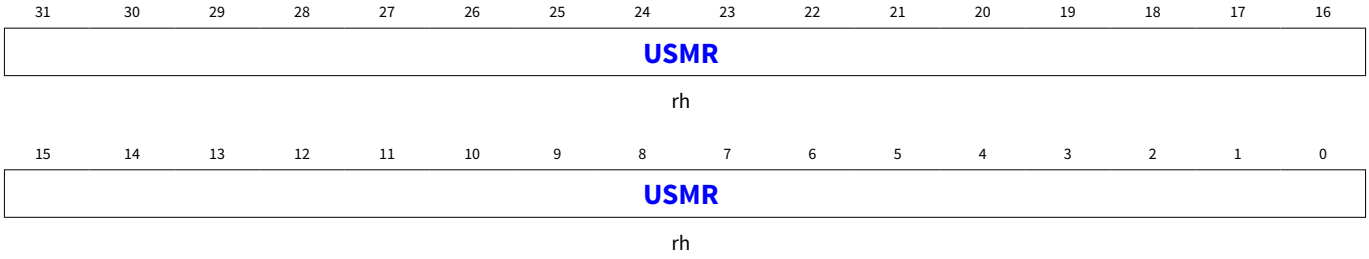
Field	Bits	Type	Description
PLERCLR	7	w	**PLL lost lock error interrupt clear** 0_B No action 1_B Clear PLER
USMCLR	8	w	**Unsolicited Message received clear** 0_B No action 1_B Clear USM
PARCLR	9	w	**PING Answer received clear** 0_B No action 1_B Clear PAR
TXTECLR	10	w	**TX disable error interrupt clear** 0_B No action 1_B Clear TXTE
SFOCLR	11	w	**Synchronization FIFO overflow (in RX direction) interrupt clear** 0_B No action 1_B Clear SFO
SFUCLR	12	w	**Synchronization FIFO underflow (in TX direction) interrupt clear** 0_B No action 1_B Clear SFU
MSCELR	13	w	**Multi Slave scenario Command Error interrupt clear** 0_B No action 1_B Clear MSCE
0	0, 31:14	r	**Reserved** Read as 0; should be written with 0

17.6.2.3.12 Unsolicited Status Message Received

The Unsolicited Status message register is available to capture unsolicited messages received with the logical channel command encoding in the header of the frame (Logical Channel Type Hex 1). This function is not available in multi Slave scenario.

USMR Offset address: 00090_H

Unsolicited Status Message Received Kernel Reset value: $0000\ 0000_H$



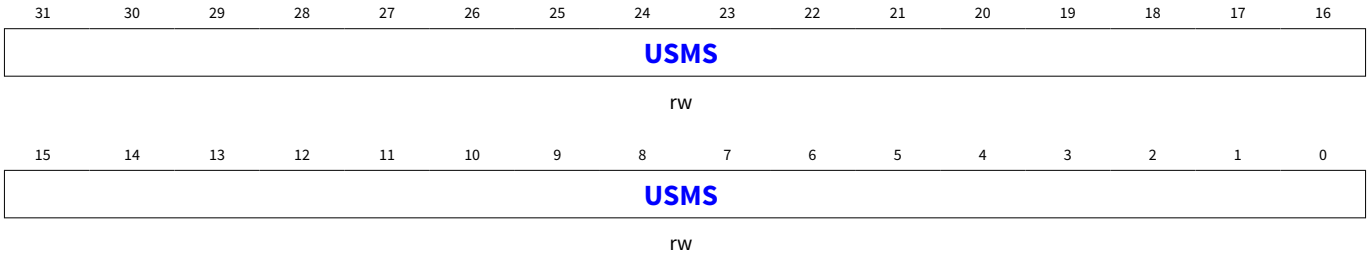
Field	Bits	Type	Description
USMR	31:0	rh	**Unsolicited status message received** The register contains the last received unsolicited status message.

17.6.2.3.13 Unsolicited Status Message Send

A write to the register activates the interface to send out an unsolicited status message to the other interface side. The Logical Channel Type reflected in the header is Hex 1 and the Payload is 32-bit always. An unsolicited frame has a higher priority then a Frame based on a data channel. In between frames the USMS gets higher priority then the data transfer requested by the HSSL. This function is not available in multi Slave scenario.

USMS Offset address: 00094_H

Unsolicited Status Message Send Kernel Reset value: $0000\ 0000_H$



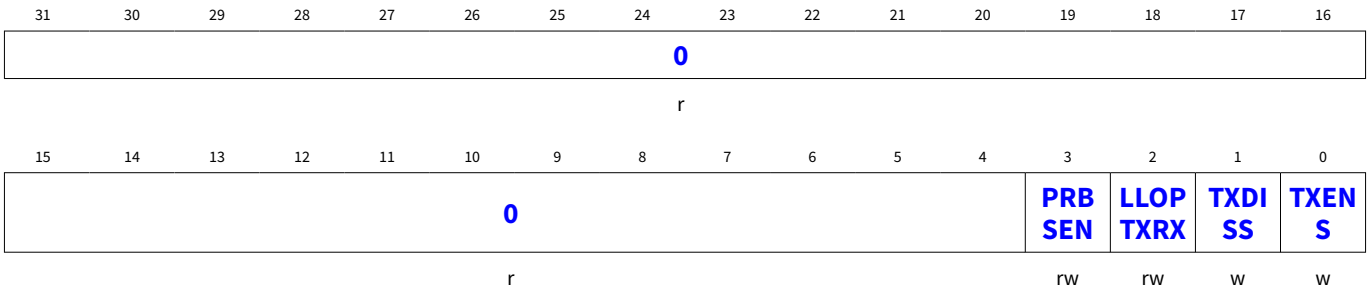
Field	Bits	Type	Description
USMS	31:0	rw	**Unsolicited status message send** Writing to the register triggers an unsolicited status message to be send to the other interface side.

17.6.2.3.14 Test Control Register

***Note**: The TESTCTRL.LLOPTXRX bit is not working with IFCTRL.IFTESTMD. Data is not transported to HSCT controller part (link layer).*

TESTCTRL Offset address: 00098_H

Test Control Register Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
TXENS	0	w	**Enable Slave TX path (Slave interface mode only)** ***Note**: This function should be only used during interface testing the mode and SW development. In functional mode the Slave interface should only be controlled via transfer commands received from Master interface. (This trigger register reads back 0 always.) If TXENS and TXDISS are written one, TXDISS has higher priority. The status is visible in IFSTAT.TX_EN.* 0_B Writing logic 0 has no effect 1_B Triggers the Slave interface TX enable.
TXDISS	1	w	**Disable Slave TX path (Slave Interface mode only)** ***Note**: This function should be only used during interface testing mode and SW development. In functional mode the Slave interface should only be controlled via transfer commands received from Master interface. (This trigger register reads back 0 always.) If TXENS and TXDISS are written one, TXDISS has higher priority. The status is visible in IFSTAT.TX_EN.* 0_B Writing logic 0 has no effect 1_B Triggers the Slave interface TX disable.
LLOPTXRX	2	rw	**LVDS loop back TX to RX enable** Transmit data at LVDS is directly looped back to the LVDS RX. Data transfer speed is defined by the TX speed configuration. (The data path in the SoC is using the complete Transmit path through all functional layers and is looped back at LVDS from TX to RX. Also at RX data path all SoC data layers are active.) ***Note**: Requires same speed configuration for RX- and TX-link before enabled.* 0_B Disabled LVDS TX to RX data Loop back test mode 1_B Enabled TX to RX data loop back test mode.
PRBSEN	3	rw	**PRBS Pattern enable** Enable of the PRBSEN bit allows a continuos PRBS stream with the configured transfer speed Baud rate. This feature is available to measure ISI during the time other SoC functions are running in an applicative mode. This feature is for measurement purpose only. 0_B Disabled PRBS pattern generation on TX 1_B Enabled PRBS pattern generation on TX.
0	31:4	r	**Reserved** Read as 0; should be written with 0

17.6.2.4 UBS Registers

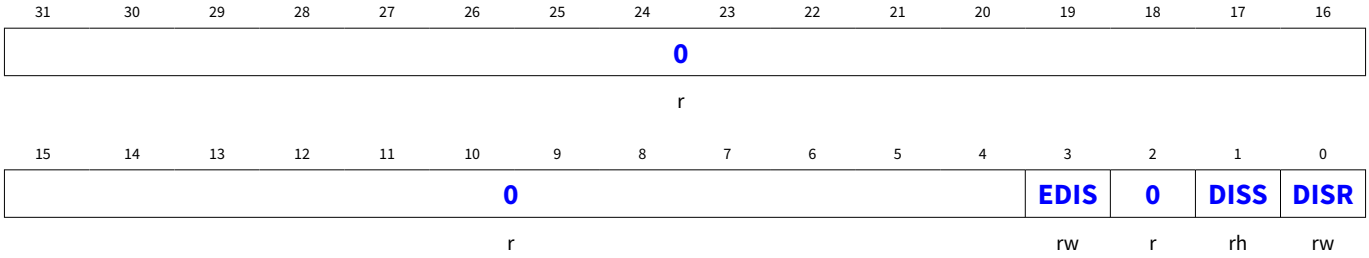
17.6.2.4.1 Clock Control Register

The Clock Control Register CLC allows the programmer to adapt the functionality and power consumption of the module to the requirements of the application.

Register CLC controls the module clock signal and the reactivity to the sleep signal.

CLC Offset address: 00000_H

Clock Control Register Application Reset value: $0000\ 0003_H$



Field	Bits	Type	Description
DISR	0	rw	**Module Disable Request Bit** Used for enable/disable control of the module. 0_B On request: enable the module clock 1_B Off request: stop the module clock
DISS	1	rh	**Module Disable Status Bit** 0_B Module clock is enabled 1_B Off: module is not clocked
EDIS	3	rw	**Sleep Mode Enable Control** Used to control the module's reaction to sleep mode. 0_B Sleep mode request is enabled and functional 1_B Module disregards the sleep mode control signal
0	2, 31:4	r	**Reserved** Read as 0; should be written with 0

17.6.2.4.2 OCDS Control and Status Register

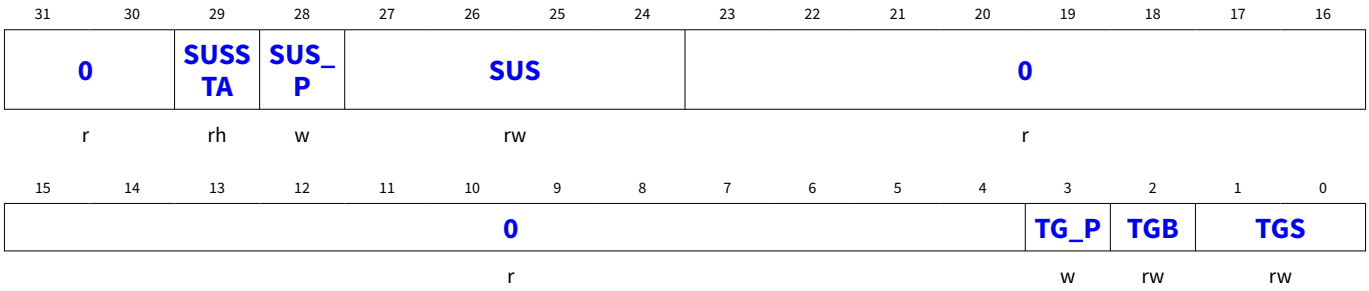
The OCDS Control and Status (OCS) register is cleared by Debug Reset.

The OCDS Control and Status register OCS controls the debug and trace behavior by selecting suspend modes and OTGB trigger sets. When OCDS is disabled the suspend control is ineffective.

OCS Offset address: 00004_H

OCDS Control and Status Register Debug Reset value: $0000\ 0000_H$

17 High Speed Serial Link (HSSL)



Field	Bits	Type	Description
TGS	1:0	rw	**Trigger Set for OTGB0/1** 00_B No Trigger Set output 01_B TS16_HSCT **others,** reserved (no Trigger Set selected)
TGB	2	rw	**OTGB0/1 Bus Select** 0_B Trigger Set is output on OTGB0 1_B Trigger Set is output on OTGB1
TG_P	3	w	**TGS, TGB Write Protection** TGS and TGB are only written when TG_P is 1, otherwise unchanged. Read as 0.
SUS	27:24	rw	**OCDS Suspend Control** Controls the sensitivity to the suspend signal coming from the OCDS Trigger Switch (OTGS) Not listed combinations are reserved. ***Note**: After a Hard suspend the HSCT and higher layer protocol module must be reset. A a new initialization sequence is required afterwards* 0_H Will not suspend 1_H Hard suspend: Clocks will be disabled immediately **others,** reserved
SUS_P	28	w	**SUS Write Protection** SUS is only written when SUS_P is 1, otherwise unchanged. Read as 0.
SUSSTA	29	rh	**Suspend State** 0_B Module is not (yet) suspended 1_B Module is suspended
0	23:4, 31:30	r	**Reserved** Read as 0; should be written with 0

Table 802 Access mode restrictions of OCS sorted by descending priority

Mode name	Access mode		Description
write 1 to .**TG_P**	rw	TGB, TGS	Set TG_P during write access
write 1 to .**SUS_P**	rw	SUS	Set SUS_P during write access
(default)	r	SUS, TGB, TGS	

17.6.2.4.3 Reset Control Register A

RST_CTRLA Offset address: $0000C_H$

Reset Control Register A Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				GRSTEN3	GRSTEN2	GRSTEN1	GRSTEN0	0							KRST
r				rw	rw	rw	rw	r							rwh

Field	Bits	Type	Description
KRST	0	rwh	**Kernel Reset** Request a kernel reset. The requested reset is executed if CTRLA.KRST=1 and CTRLB.KRST=1. KRST is cleared after the kernel reset was executed. . 0_B No action 1_B A kernel reset was requested
GRSTENx (x=0-3)	x+8	rw	**Enable for Global Module Reset Group x** 0_B global module reset group x does not have any effect 1_B global module reset group x results in a kernel reset
0	7:1, 31:12	r	**Reserved** Read as 0; should be written with 0

17.6.2.4.4 Reset Control Register B

RST_CTRLB Offset address: 00010_H

Reset Control Register B Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
STATCLR								0							
w								r							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															KRST
r															rwh

Field	Bits	Type	Description
KRST	0	rwh	**Kernel Reset** Request a kernel reset. The requested reset is executed if CTRLA.KRST=1 and CTRLB.KRST=1. KRST is cleared after the kernel reset was executed. 0_B No action 1_B A kernel reset was requested
STATCLR	31	w	**Kernel Reset Status Clear** Clears all status bits in RST_STAT when 1 is written. Read will return 0. 0_B No action 1_B Write with ´1´ clears bits STAT.GRSTx and bit STAT.KRST .
0	30:1	r	**Reserved** Read as 0; should be written with 0

17.6.2.4.5 Reset Status Register

The reset status register contains the status bits for kernel reset (KRST) and the global module reset groups (GRSTx).

RST_STAT Offset address: 00014_H

Reset Status Register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				GRST3	GRST2	GRST1	GRST0	0							KRST
r				rh	rh	rh	rh	r							rh

Field	Bits	Type	Description
KRST	0	rh	**Kernel Reset Status** Indicates an executed kernel reset. RSTSTAT.KRST is set after the execution of a kernel reset in the same clock cycle in which the reset bits are cleared. Clear KRST by setting bit STATCLR in register RST_CTLB.
GRSTx (x=0-3)	x+8	rh	**Status for Global Module Reset Group x** 0_B Reset was not triggered by global reset group x 1_B Reset was triggered by global reset group x
0	7:1, 31:12	r	**Reserved** Read as 0; should be written with 0

17.6.2.4.6 Register Endinit

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner and in some PROT states, the secure master to update the PROT state as described in the PROT mechanism.

The PROTE register controls lock / unlock of the local Endinit (E) protected control registers.

PROTE Offset address: 00020_H

Register Endinit Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OWEN	ODEF	TAGID						PRSEN	PRS			VMEN	VM		
w	rw	rw						rw	rw			rw	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												SWEN	STATE		
r												w	rwh		

Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
VM	18:16	rw	**Virtual machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual machine definition enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.

(table continues...)

(continued)

Field	Bits	Type	Description
PRSEN	23	rw	**Protection set definition enable for PROT owner** 0_B **Disable**: PRS is not part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG-ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; should be written with 0.

Table 803 Access mode restrictions of PROTE sorted by descending priority

Mode name	Access mode		Description
write 1 to .**SWEN**	rwh	STATE	
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

17.6.2.4.7 PROT Register Safe Endinit

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner and in some PROT states, the secure master to update the PROT state as described in the PROT mechanism.

The PROTSE register controls lock / unlock of the local Safe Endinit (SE) protected control registers.

PROTSE Offset address: 00024_H

PROT Register Safe Endinit Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OWEN	**ODEF**	**TAGID**						**PRSEN**	**PRS**			**VMEN**	**VM**		
w	rw	rw						rw	rw			rw	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												**SWEN**	**STATE**		
r												w	rwh		

Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
VM	18:16	rw	**Virtual machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual machine definition enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.
PRSEN	23	rw	**Protection set definition enable for PROT owner** 0_B **Disable**: PRS is not part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG-ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; should be written with 0.

Table 804 Access mode restrictions of PROTSE sorted by descending priority

Mode name	Access mode		Description
write 1 to .**SWEN**	rwh	STATE	
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

17.6.2.4.8 Module Identification Register

ID Offset address: 00008_H

Module Identification Register PowerOn Reset value: $00B4\ C003_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MOD_NUM															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MOD_TYPE								MOD_REV							
r								r							

Field	Bits	Type	Description
MOD_REV	7:0	r	**Module Revision** Indicates the revision number of the implementation.
MOD_TYPE	15:8	r	**Module Type** This internal marker is fixed to $C0_H$.
MOD_NUM	31:16	r	**Module Number** Indicates the module identification number

17.6.2.4.9 Write access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

ACCEN_WRA Offset address: 00040_H

Write access enable register A Application Reset value: $1000\ 0003_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access

17.6.2.4.10 Write access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

ACCEN_WRB Offset address: 00044_H

Write access enable register B Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access
0	31:8	r	**Reserved** Read as 0; should be written with 0.

17.6.2.4.11 Read access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

ACCEN_RDA Offset address: 00048_H

Read access enable register A Application Reset value: $FFFF\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access

17.6.2.4.12 Read access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

ACCEN_RDB Offset address: $0004C_H$

Read access enable register B Application Reset value: $0000\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access
0	31:8	r	**Reserved** Read as 0; should be written with 0.

17.6.2.4.13 VM access enable register

This register defines which virtual machine encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each VMx.

ACCEN_VM Offset address: 00050_H

VM access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for VM ID q** This bit enables read access to the access protected region for transactions with the VM ID q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for VM ID q** This bit enables write access to the access protected region for transactions with the VM ID q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

17.6.2.4.14 PRS access enable register

This register defines which protection register set encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each PRSx.

ACCEN_PRS Offset address: 00054_H

PRS access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for PRS q** This bit enables read access to the access protected region for transactions with the PRS q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for PRS q** This bit enables write access to the access protected region for transactions with the PRS q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

17.7 Debug information

HSSL module debug information on suspend can be found in section "Modes of Operation". HSSL trigger set debug information is available in section "OCDS Trigger Sets". HSCT module suspend mode information can be found in section "Suspend and power-off behavior".

The HSCT has one OCDS Trigger set available which gives a maximum of 16 triggers. The next table describes the available triggers.

Table 805 TS16_HSCT OCDS trigger set

Bits	Name	Description
0	HRXRQ	HSSL Interface RX transfer direction request
1	HRXCTS	HSSL Interface RX transfer direction CTS
2	HTXCTS	HSSL Interface TX transfer direction CTS
3	HTXRQ	HSSL Interface TX transfer direction request
4	TXF	First Byte transfer of transfer frame in TX transfer direction at controller to physical layer interface
5	TXL	Last Byte transfer of transfer frame in TX transfer direction at controller to physical layer interface
6	TXW	Transmit direction wake-up at controller to physical layer interface
7	TXS	Transmit direction sleep indication at controller to physical layer interface
8	RXFS	Receive direction frame start at physical layer to controller interface
9	DFSMI	Deframer state machine not in idle
10	IFCT	Slave interface frame command trigger
[15:11]		Reserved (value is 0)

Related information

Modes of operation on page 2462
OCDS trigger sets on page 2470
Suspend and power-off behavior on page 2510

17.8 References

[1] IEEE Std 1596.3-1995/1996 - IEEE Standard for Low-Voltage Differential Signals (LVDS) for Scalable Coherent Interface (SCI)

[2] C40FLA LVDS Development Spec from Infineon.

[3] IFX PCB layout guidelines from Infineon.

[4] HSCT_PHY Specification document: Hardware Detailed Design Description 1.15

17.9 HSSL revision history

Reference	Description of change(s)
Date range: 2024-08-17 to 2024-09-26	
Interrupt register	• Removing the sentence of Minimum SRI frequency sentence as not applicable

17.10 TC4Dx HSSL information

17.10.1 TC4Dx HSSL configuration

In TC4D there are two instances of HSSL sub system being implemented.

17.10.2 TC4Dx HSSL features

There are no deviations from the generic specification.

17.10.3 TC4Dx HSSL functional description

In TC4Dx, the HSSL module is connected to the COMPB bus with a slave and a master interface. The HSSL module uses the SRI master or the COMPB master depending on the accessed address. The COMPB address space (F400 0000_H - F5FF $FFFF_H$) is used as COMPB master address window in TC4Dx.

***Note**: In the HSSL generic content, the description for SPB also applies to COMPB, except where a COMPB difference is explicitly mentioned.*

17.10.4 TC4Dx HSSL registers

There are no deviations from the generic specification.

17.10.4.1 Register address space - HSSL

Table 806 Registers address space - HSSL

Module	Base address	End address	Note
HSSL0	$F4480000_H$	$F44803FF_H$	
HSSL1	$F44A0000_H$	$F44A03FF_H$	

17.10.4.2 Register overview - access mode glossary

Table 807 Register overview - access mode glossary

Keyword	Description
E	Access protection using PROT register HSSL0_PROTE or HSSL1_PROTE .
SE	Access protection using PROT register HSSL0_PROTSE or HSSL1_PROTSE .
APU-P	Protection group consisting of registers HSSL0_ACCEN_WRA , HSSL0_ACCEN_WRB , HSSL0_ACCEN_RDA , HSSL0_ACCEN_RDB , HSSL0_ACCEN_VM , HSSL0_ACCEN_PRS or HSSL1_ACCEN_WRA , HSSL1_ACCEN_WRB , HSSL1_ACCEN_RDA , HSSL1_ACCEN_RDB , HSSL1_ACCEN_VM , HSSL1_ACCEN_PRS .
P	Access protection using *APU-P* registers.
BE	Always returns a Bus Error.
U	No access restrictions.
PROT	Access restrictions as defined in the PROT register access rules.
SECURE_MASTER	Description can be found in global access mode definition.

17.10.4.3 Register overview - HSSL0 (ascending offset address)

Table 808 Register overview - HSSL0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
HSSL0_CLC	Clock Control Register	000_H	P	P, SV, E	Application Reset	2545
HSSL0_OCS	OCDS Control and Status Register	004_H	P	SV, P	Debug Reset	2545
HSSL0_ID	Module Identification Register	008_H	P	BE	PowerOn Reset	2516
HSSL0_RST_CTRLA	Reset Control Register A	$00C_H$	P	P, SV, E	Application Reset	2547
HSSL0_RST_CTRLB	Reset Control Register B	010_H	P	P, SV, E	Application Reset	2547

(table continues...)

Table 808 (continued) Register overview - HSSL0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
HSSL0_RST_STAT	Reset Status Register	014_H	P	BE	Application Reset	2548
HSSL0_PROTE	PROT Register Endinit	020_H	U	SV, PROT	Application Reset	2549
HSSL0_PROTSE	PROT Register Safe Endinit	024_H	U	SV, PROT	Application Reset	2550
HSSL0_ACCEN_WRA	Write access enable register A	040_H	U	SE, SV	Application Reset	2552
HSSL0_ACCEN_WRB	Write access enable register B	044_H	U	SE, SV	Application Reset	2552
HSSL0_ACCEN_RDA	Read access enable register A	048_H	U	SE, SV	Application Reset	2553
HSSL0_ACCEN_RDB	Read access enable register B	$04C_H$	U	SE, SV	Application Reset	2553
HSSL0_ACCEN_VM	VM access enable register	050_H	U	SE, SV	Application Reset	2554
HSSL0_ACCEN_PRS	PRS access enable register	054_H	U	SE, SV	Application Reset	2554
HSSL0_CRC	CRC Control Register	060_H	P	SV, E, P	Kernel Reset	2516
HSSL0_CFG	Configuration Register	064_H	P	P	Kernel Reset	2517
HSSL0_QFLAGS	Request Flags Register	068_H	P	P	Kernel Reset	2518
HSSL0_MFLAGS	Miscellaneous Flags Register	$06C_H$	P	P	Kernel Reset	2520
HSSL0_MFLAGSSET	Miscellaneous Flags Set Register	070_H	P	P	Kernel Reset	2522
HSSL0_MFLAGSCL	Miscellaneous Flags Clear Register	074_H	P	P	Kernel Reset	2525
HSSL0_MFLAGSEN	Flags Enable Register	078_H	P	P	Kernel Reset	2528
HSSL0_SFSFLAGS	Stream FIFOs Status Flags Register	$07C_H$	P	P	Kernel Reset	2530
HSSL0_IWDx (x=0-3)	Initiator Write Data Register x	080_H+x* 10_H	P	P	Kernel Reset	2532
HSSL0_ICONx (x=0-3)	Initiator Control Data Register x	084_H+x* 10_H	P	P	Kernel Reset	2532
HSSL0_IRWAx (x=0-3)	Initiator Read Write Address Register x	088_H+x* 10_H	P	P	Kernel Reset	2534
HSSL0_IRDx (x=0-3)	Initiator Read Data Register x	$08C_H$+x* 10_H	P	P	Kernel Reset	2534

(table continues...)

Table 808 (continued) Register overview - HSSL0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
HSSL0_TCDi (i=0-3)	Target Current Data Register i	$0C0_H$+i*8	P	P	Kernel Reset	2535
HSSL0_TCAi (i=0-3)	Target Current Address Register i	$0C4_H$+i*8	P	P	Kernel Reset	2535
HSSL0_TSTAT	Target Status Register	$0E0_H$	P	P	Kernel Reset	2536
HSSL0_TIDADD	Target ID Address Register	$0E4_H$	P	SV, P	Kernel Reset	2536
HSSL0_SEC	Security Control Register	$0E8_H$	SECURE_MASTER|P	SECURE_MASTER	Kernel Reset	2543
HSSL0_MSCR	Multi Slave Control Register	$0EC_H$	P	P	Kernel Reset	2543
HSSL0_ISSAx (x=0-1)	Initiator Stream Start Address Register x	$0F0_H$+x*4	P	P	Kernel Reset	2537
HSSL0_ISCA	Initiator Stream Current Address Register	$0F8_H$	P	P	Kernel Reset	2537
HSSL0_ISFC	Initiator Stream Frame Count Register	$0FC_H$	P	P	Kernel Reset	2538
HSSL0_TSSAx (x=0-1)	Target Stream Start Address Register x	100_H+x*4	P	P	Kernel Reset	2539
HSSL0_TSCA	Target Stream Current Address Register	108_H	P	P	Kernel Reset	2539
HSSL0_TSFC	Target Stream Frame Count Register	$10C_H$	P	P	Kernel Reset	2540
HSSL0_AWSTARTi (i=0-3)	Access Window Start Register i	110_H+i*8	SV, P	SV, P	Kernel Reset	2541
HSSL0_AWENDi (i=0-3)	Access Window End Register i	114_H+i*8	SV, P	SV, P	Kernel Reset	2541
HSSL0_AR	Access Rules Register	130_H	SV, P	SV, P	Kernel Reset	2542

17.10.4.4 Register overview - HSSL1 (ascending offset address)

Table 809 Register overview - HSSL1 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
HSSL1_CLC	Clock Control Register	000_H	P	P, SV, E	Application Reset	2545
HSSL1_OCS	OCDS Control and Status Register	004_H	P	SV, P	Debug Reset	2545
HSSL1_ID	Module Identification Register	008_H	P	BE	PowerOn Reset	2516

(table continues...)

Table 809 (continued) Register overview - HSSL1 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
HSSL1_RST_CTRLA	Reset Control Register A	$00C_H$	P	P, SV, E	Application Reset	2547
HSSL1_RST_CTRLB	Reset Control Register B	010_H	P	P, SV, E	Application Reset	2547
HSSL1_RST_STAT	Reset Status Register	014_H	P	BE	Application Reset	2548
HSSL1_PROTE	PROT Register Endinit	020_H	U	SV, PROT	Application Reset	2549
HSSL1_PROTSE	PROT Register Safe Endinit	024_H	U	SV, PROT	Application Reset	2550
HSSL1_ACCEN_WRA	Write access enable register A	040_H	U	SE, SV	Application Reset	2552
HSSL1_ACCEN_WRB	Write access enable register B	044_H	U	SE, SV	Application Reset	2552
HSSL1_ACCEN_RDA	Read access enable register A	048_H	U	SE, SV	Application Reset	2553
HSSL1_ACCEN_RDB	Read access enable register B	$04C_H$	U	SE, SV	Application Reset	2553
HSSL1_ACCEN_VM	VM access enable register	050_H	U	SE, SV	Application Reset	2554
HSSL1_ACCEN_PRS	PRS access enable register	054_H	U	SE, SV	Application Reset	2554
HSSL1_CRC	CRC Control Register	060_H	P	SV, E, P	Kernel Reset	2516
HSSL1_CFG	Configuration Register	064_H	P	P	Kernel Reset	2517
HSSL1_QFLAGS	Request Flags Register	068_H	P	P	Kernel Reset	2518
HSSL1_MFLAGS	Miscellaneous Flags Register	$06C_H$	P	P	Kernel Reset	2520
HSSL1_MFLAGSSET	Miscellaneous Flags Set Register	070_H	P	P	Kernel Reset	2522
HSSL1_MFLAGSCL	Miscellaneous Flags Clear Register	074_H	P	P	Kernel Reset	2525
HSSL1_MFLAGSEN	Flags Enable Register	078_H	P	P	Kernel Reset	2528
HSSL1_SFSFLAGS	Stream FIFOs Status Flags Register	$07C_H$	P	P	Kernel Reset	2530
HSSL1_IWDx (x=0-3)	Initiator Write Data Register x	080_H+x* 10_H	P	P	Kernel Reset	2532
HSSL1_ICONx (x=0-3)	Initiator Control Data Register x	084_H+x* 10_H	P	P	Kernel Reset	2532
HSSL1_IRWAx (x=0-3)	Initiator Read Write Address Register x	088_H+x* 10_H	P	P	Kernel Reset	2534

(table continues...)

Table 809 (continued) Register overview - HSSL1 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
HSSL1_IRDx (x=0-3)	Initiator Read Data Register x	$08C_H$+x*10_H	P	P	Kernel Reset	2534
HSSL1_TCDi (i=0-3)	Target Current Data Register i	$0C0_H$+i*8	P	P	Kernel Reset	2535
HSSL1_TCAi (i=0-3)	Target Current Address Register i	$0C4_H$+i*8	P	P	Kernel Reset	2535
HSSL1_TSTAT	Target Status Register	$0E0_H$	P	P	Kernel Reset	2536
HSSL1_TIDADD	Target ID Address Register	$0E4_H$	P	SV, P	Kernel Reset	2536
HSSL1_SEC	Security Control Register	$0E8_H$	SECURE_MASTER|P	SECURE_MASTER	Kernel Reset	2543
HSSL1_MSCR	Multi Slave Control Register	$0EC_H$	P	P	Kernel Reset	2543
HSSL1_ISSAx (x=0-1)	Initiator Stream Start Address Register x	$0F0_H$+x*4	P	P	Kernel Reset	2537
HSSL1_ISCA	Initiator Stream Current Address Register	$0F8_H$	P	P	Kernel Reset	2537
HSSL1_ISFC	Initiator Stream Frame Count Register	$0FC_H$	P	P	Kernel Reset	2538
HSSL1_TSSAx (x=0-1)	Target Stream Start Address Register x	100_H+x*4	P	P	Kernel Reset	2539
HSSL1_TSCA	Target Stream Current Address Register	108_H	P	P	Kernel Reset	2539
HSSL1_TSFC	Target Stream Frame Count Register	$10C_H$	P	P	Kernel Reset	2540
HSSL1_AWSTARTi (i=0-3)	Access Window Start Register i	110_H+i*8	SV, P	SV, P	Kernel Reset	2541
HSSL1_AWENDi (i=0-3)	Access Window End Register i	114_H+i*8	SV, P	SV, P	Kernel Reset	2541
HSSL1_AR	Access Rules Register	130_H	SV, P	SV, P	Kernel Reset	2542

17.10.4.5 Device specific registers

There are no device specific register changes.

17.10.4.6 Register address space - HSCT

Table 810 Registers address space - HSCT

Module	Base address	End address	Note
HSCT0	$F4490000_H$	$F449FFFF_H$	
HSCT1	$F44B0000_H$	$F44BFFFF_H$	

17.10.4.7 Register overview - access mode glossary

Table 811 Register overview - access mode glossary

Keyword	Description
E	Access protection using PROT register HSCT0_PROTE or HSCT1_PROTE .
SE	Access protection using PROT register HSCT0_PROTSE or HSCT1_PROTSE .
APU-P	Protection group consisting of registers HSCT0_ACCEN_WRA , HSCT0_ACCEN_WRB , HSCT0_ACCEN_RDA , HSCT0_ACCEN_RDB , HSCT0_ACCEN_VM , HSCT0_ACCEN_PRS or HSCT1_ACCEN_WRA , HSCT1_ACCEN_WRB , HSCT1_ACCEN_RDA , HSCT1_ACCEN_RDB , HSCT1_ACCEN_VM , HSCT1_ACCEN_PRS .
P	Access protection using *APU-P* registers.
BE	Always returns a Bus Error.
U	No access restrictions.
PROT	Access restrictions as defined in the PROT register access rules.

17.10.4.8 Register overview - HSCT0 (ascending offset address)

Table 812 Register overview - HSCT0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
HSCT0_CLC	Clock Control Register	00000_H	P	P, SV, E	Application Reset	2574
HSCT0_OCS	OCDS Control and Status Register	00004_H	P	SV, P	Debug Reset	2574
HSCT0_ID	Module Identification Register	00008_H	P	BE	PowerOn Reset	2581
HSCT0_RST_CTRLA	Reset Control Register A	$0000C_H$	P	P, SV, E	Application Reset	2576
HSCT0_RST_CTRLB	Reset Control Register B	00010_H	P	P, SV, E	Application Reset	2576
HSCT0_RST_STAT	Reset Status Register	00014_H	P	BE	Application Reset	2577

(table continues...)

Table 812 (continued) Register overview - HSCT0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
HSCT0_PROTE	Register Endinit	00020_H	U	SV, PROT	Application Reset	2578
HSCT0_PROTSE	PROT Register Safe Endinit	00024_H	U	SV, PROT	Application Reset	2579
HSCT0_ACCEN_WRA	Write access enable register A	00040_H	U	SE, SV	Application Reset	2581
HSCT0_ACCEN_WRB	Write access enable register B	00044_H	U	SE, SV	Application Reset	2582
HSCT0_ACCEN_RDA	Read access enable register A	00048_H	U	SE, SV	Application Reset	2582
HSCT0_ACCEN_RDB	Read access enable register B	$0004C_H$	U	SE, SV	Application Reset	2583
HSCT0_ACCEN_VM	VM access enable register	00050_H	U	SE, SV	Application Reset	2583
HSCT0_ACCEN_PRS	PRS access enable register	00054_H	U	SE, SV	Application Reset	2584
HSCT0_INIT	Initialization Register	00060_H	P	P	Kernel Reset	2557
HSCT0_IFCTRL	Interface Control Register	00064_H	P	P	Kernel Reset	2559
HSCT0_CTSCTRL	Clear To Send Control Register	$0006C_H$	P	P	Kernel Reset	2561
HSCT0_DISABLE	Transmission Disable Register	00070_H	P	P	Kernel Reset	2562
HSCT0_STAT	Status Register	00074_H	P	P	Kernel Reset	2563
HSCT0_IFSTAT	Interface Status Register	00078_H	P	P	Kernel Reset	2564
HSCT0_CONFIGPHY	Configuration Physical Layer Register	$0007C_H$	P	P	Kernel Reset	2564
HSCT0_STATPHY	STATPHY	00080_H	P	P	Kernel Reset	2565
HSCT0_IRQ	Interrupt register	00084_H	P	P	Kernel Reset	2566
HSCT0_IRQEN	Interrupt Enable Register	00088_H	P	P	Kernel Reset	2568
HSCT0_IRQCLR	Interrupt Clear Register	$0008C_H$	P	P	Kernel Reset	2570
HSCT0_USMR	Unsolicited Status Message Received	00090_H	P	P	Kernel Reset	2571
HSCT0_USMS	Unsolicited Status Message Send	00094_H	P	P	Kernel Reset	2572
HSCT0_TESTCTRL	Test Control Register	00098_H	P	SV, P	Kernel Reset	2572

17.10.4.9 Register overview - HSCT1 (ascending offset address)

Table 813 Register overview - HSCT1 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
HSCT1_CLC	Clock Control Register	00000_H	P	P, SV, E	Application Reset	2574
HSCT1_OCS	OCDS Control and Status Register	00004_H	P	SV, P	Debug Reset	2574
HSCT1_ID	Module Identification Register	00008_H	P	BE	PowerOn Reset	2581
HSCT1_RST_CTRLA	Reset Control Register A	$0000C_H$	P	P, SV, E	Application Reset	2576
HSCT1_RST_CTRLB	Reset Control Register B	00010_H	P	P, SV, E	Application Reset	2576
HSCT1_RST_STAT	Reset Status Register	00014_H	P	BE	Application Reset	2577
HSCT1_PROTE	Register Endinit	00020_H	U	SV, PROT	Application Reset	2578
HSCT1_PROTSE	PROT Register Safe Endinit	00024_H	U	SV, PROT	Application Reset	2579
HSCT1_ACCEN_WRA	Write access enable register A	00040_H	U	SE, SV	Application Reset	2581
HSCT1_ACCEN_WRB	Write access enable register B	00044_H	U	SE, SV	Application Reset	2582
HSCT1_ACCEN_RDA	Read access enable register A	00048_H	U	SE, SV	Application Reset	2582
HSCT1_ACCEN_RDB	Read access enable register B	$0004C_H$	U	SE, SV	Application Reset	2583
HSCT1_ACCEN_VM	VM access enable register	00050_H	U	SE, SV	Application Reset	2583
HSCT1_ACCEN_PRS	PRS access enable register	00054_H	U	SE, SV	Application Reset	2584
HSCT1_INIT	Initialization Register	00060_H	P	P	Kernel Reset	2557
HSCT1_IFCTRL	Interface Control Register	00064_H	P	P	Kernel Reset	2559
HSCT1_CTSCTRL	Clear To Send Control Register	$0006C_H$	P	P	Kernel Reset	2561
HSCT1_DISABLE	Transmission Disable Register	00070_H	P	P	Kernel Reset	2562
HSCT1_STAT	Status Register	00074_H	P	P	Kernel Reset	2563
HSCT1_IFSTAT	Interface Status Register	00078_H	P	P	Kernel Reset	2564
HSCT1_CONFIGPHY	Configuration Physical Layer Register	$0007C_H$	P	P	Kernel Reset	2564
HSCT1_STATPHY	STATPHY	00080_H	P	P	Kernel Reset	2565

(table continues...)

Table 813 (continued) Register overview - HSCT1 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
HSCT1_IRQ	Interrupt register	00084_H	P	P	Kernel Reset	2566
HSCT1_IRQEN	Interrupt Enable Register	00088_H	P	P	Kernel Reset	2568
HSCT1_IRQCLR	Interrupt Clear Register	$0008C_H$	P	P	Kernel Reset	2570
HSCT1_USMR	Unsolicited Status Message Received	00090_H	P	P	Kernel Reset	2571
HSCT1_USMS	Unsolicited Status Message Send	00094_H	P	P	Kernel Reset	2572
HSCT1_TESTCTRL	Test Control Register	00098_H	P	SV, P	Kernel Reset	2572

17.10.4.10 Device specific registers

There are no device specific register changes.

17.10.5 TC4Dx HSSL connectivity

The table below lists interfaces to and from this functional block to other blocks in the device.

Table 814 List of HSSL interface signals

Interface signals	I/O	Description
CLOCK_HSSL_fSRI	In	SRI clock
CLOCK_HSSL_fSPB	In	SPB clock
HSSL_FPI_MIF	Out	HSSL FPI master interface
HSSL_SRI_MIF	Out	HSSL SRI master interface
FPI_HSSL_SIF	In	HSSL slave interface
HSSL_IR_COK[3:0]	Out	Channel OK service request
HSSL_IR_RDI[3:0]	Out	Channel read data service request
HSSL_IR_ERR[3:0]	Out	Channel error service request
HSSL_IR_TRG[3:0]	Out	Channel trigger interrupt service request
HSSL_IR_EXI	Out	HSSL exception service request
HSSL_IR_HSCT_TRIG	Out	Interrupt trigger from HSCT sub FB
FPI_HSSL_HSCT_SIF	In	HSCT slave interface
CLOCK_HSSL_fHSCT	In	HSCT clock
CLOCK_HSSL_REFCLK	In	HSCT reference clock
PORTS_HSSL_RXD[N,P]	In	Rx data
HSSL_PORTS_TXD[N,P]	Out	Tx data
HSSL_PORTS_SYSCLK_OUT	Out	System clock output when HSCT is Master
PORTS_HSSL_SYSCLK_IN	In	System clock input when HSCT is slave

17.10.6 TC4Dx HSSL revision history

Initial release of the chapter.

18 eXpanded Serial Peripheral Interface (xSPI)

The xSPI functional block is a controller interface to enable communication with SPI slave devices compliant to the JEDEC JESD251 standard.

This master interface provides a low pin count connection to off-chip single, dual, quad, or octal SPI devices in SDR and DDR mode. The partner devices include SEMPER™ flash from Infineon (formerly Cypress), or Xccela™ from Micron and HyperBus devices such as HyperFlash (NOR flash) and HyperRAM to name a few.

18.1 Feature list

- Single, Dual, Quad, or Octal SPI and Hyperbus support over port 16
- JESD251 protocol support for Dual Data Rate (DDR) transfers with read data strobe up to 200 MHz over port 16
- Support of Single Data Rate (SDR) transfers up to 133 MHz over port 16
- Support of Quad SPI interface in SDR mode up to 25 MHz for 3 V slave devices with GPIO pads
- Option to connect two slaves in parallel using chip select
- Execute in place (XIP) support
- Internal direct memory access (DMA) support for data transfer to or from an internal memory to an external slave device
- Dynamic control of the serial bit rate
- Interrupt and alarm generation from xSPI
- Automotive safety features for register, memory, data path, and finite state machine (FSM) protection against hardware faults

18.2 Functional overview

This following figure provides a simplified system overview in the form of a block diagram.

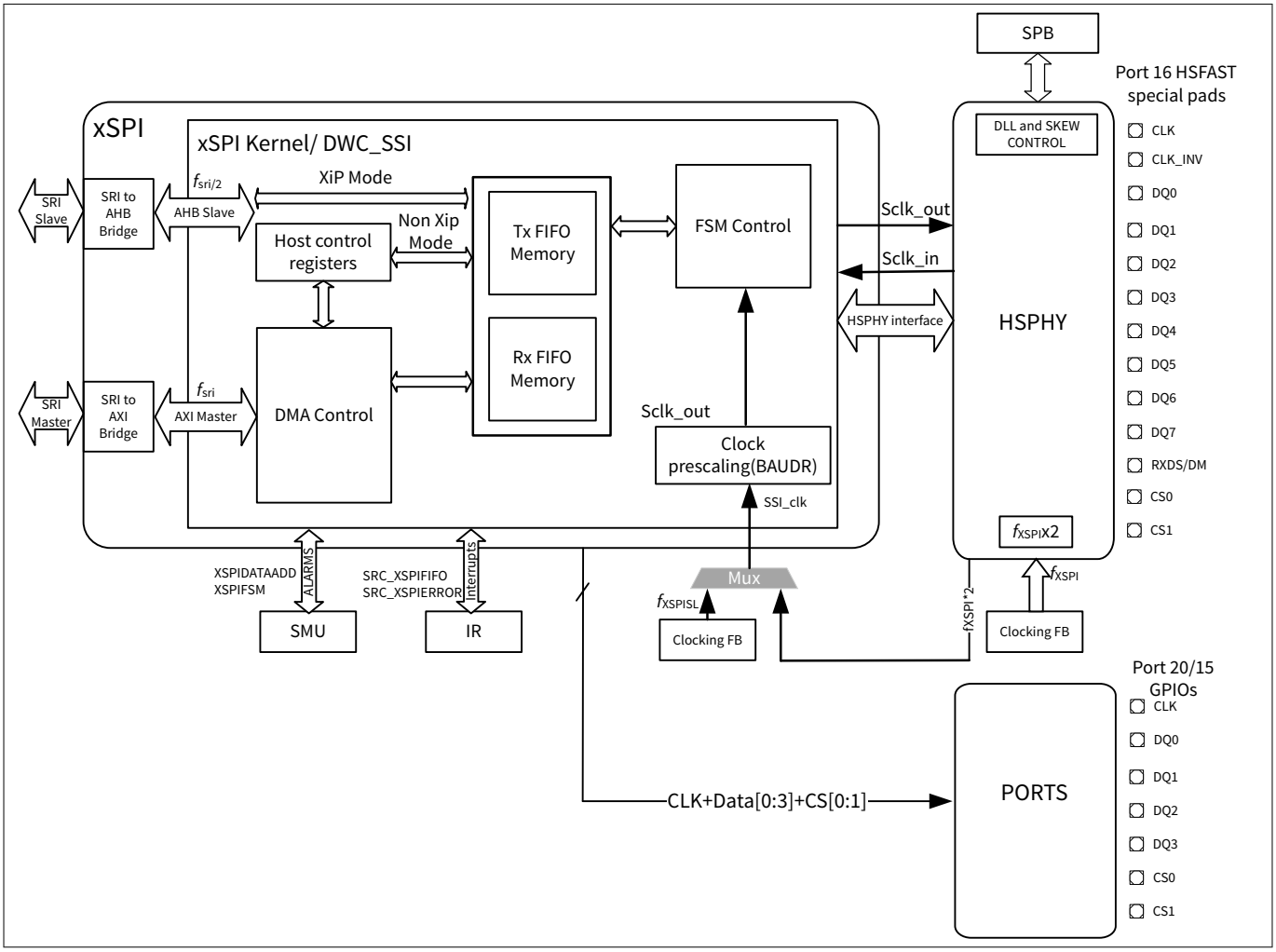


Figure 250 xSPI block diagram

For the alarm description please refer to the alarm mapping tables in the SMU functional block and the Safety Manual.

The xSPI kernel is connected to the SRI slave via the SRI to AHB bridge and SRI master via SRI to AXI bridge. Access to host control registers and memory mapped register space via XIP takes place through SRI slave interface. Data transfers from/to external memory to/from internal memory space takes place through an internal Direct Memory Access(DMA) engine via the SRI master interface. Any type of non DMA mode data transfers takes place through the SRI slave using the internal tx and rx FIFO memory.

All the safety related interrupts are grouped under two alarms XSPIDATAADD (XSPI data address error alarm) and XSPIFSM(XSPI FSM alarm) that interfaces to SMU (Safety and security alarm Management Unit). Similarly, the non safety related interrupts are grouped under two interrupts XSPIFIFO and XSPIERROR that interface to IR (Interrupt Router).

The HSPHY functional block contains the DLL (Delay Locked Loop) along with Port 16 IO (Input Output) pad related SFRs that can be accessed via an SPB interface. The xSPI receives two times the f_{xSPI} clock called f_{XSPI*2} from the HSPHY. The SSI_clk is the internal peripheral kernel clock that is used to generate the Sclk_out. The BAUDR register helps in scaling the SSI_clk if needed, to obtain the desired output clock Sclk_out. The Sclk_in is used to sample the incoming data during data reads.

The xSPI can be used either on :

- Port 16 through the HSPHY and HSFAST pads (pads specialized for xSPI performance) at maximum performance using f_{xspi*2}
- Or over GPIO pads through the PORTS in a reduced speed mode in quad SDR mode only, using f_{XSPISL} clock

Note:
- *One of the signal (TXD3/RXD3) belonging to port 20 Quad mode implementation is routed to port 15.*
- *Since there is only one instance of xSPI in any device, the xSPI cannot be used in parallel, on both port 16 and over other ports with GPIO pads*

Related information

TC4Dx SMU alarm mapping tables on page 7233

18.3 Functional description

This chapter describes the xSPI system overview, architecture, and various features supported in xSPI.

18.3.1 System overview

The figure below shows an application use case when two xSPI slave devices are connected to the interface.

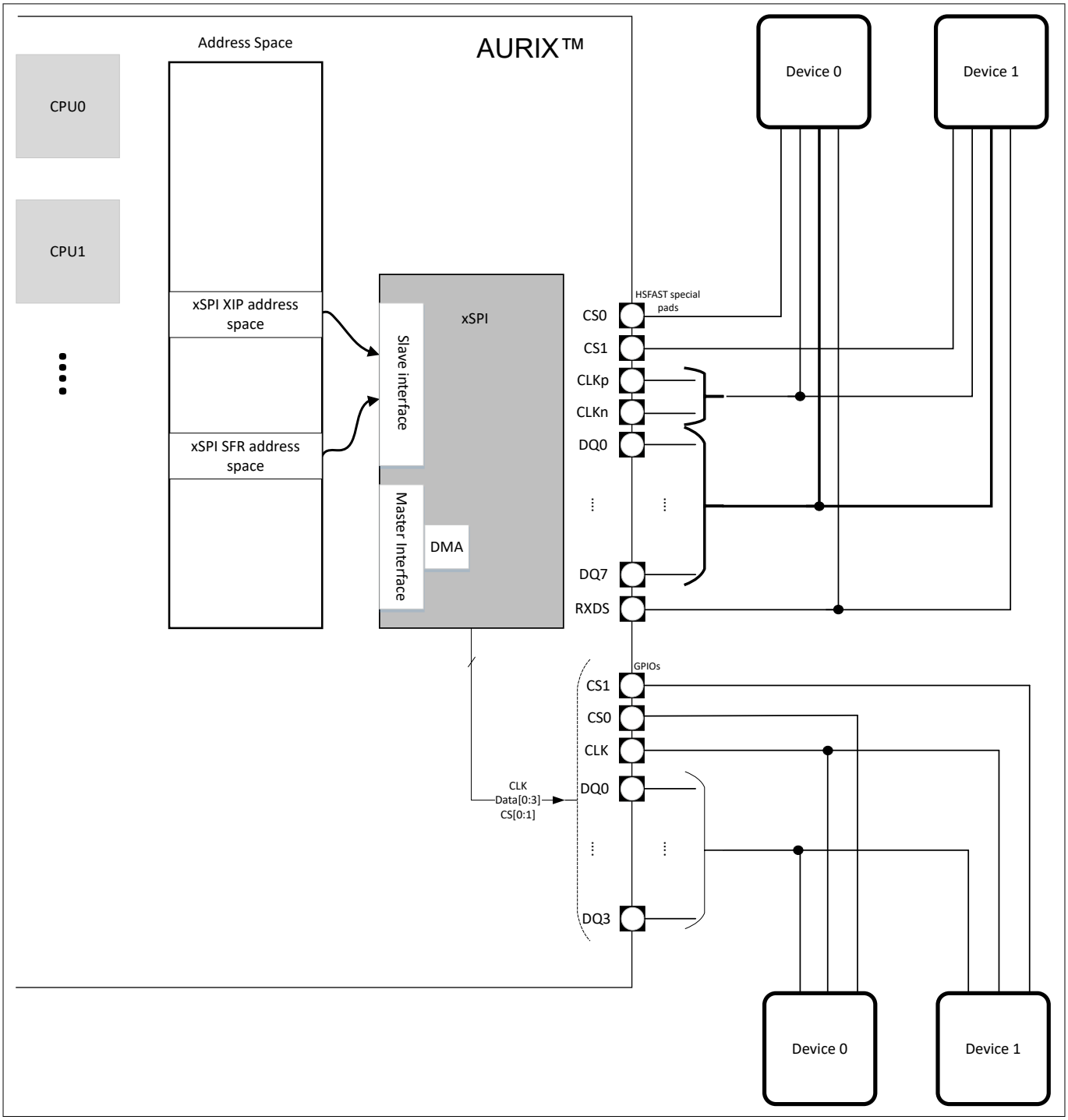


Figure 251 xSPI system overview

18.3.2 Serial Peripheral Interface (SPI)

The octal or Hyperbus, half duplex, high throughput and low pin count xSPI protocol is based on the legacy four wire, full duplex, low bandwidth SPI protocol.

xSPI is backward compatible to legacy SPI protocol along with dual and quad SPI interface protocols, which were intermediate development steps in the evolution of xSPI from the SPI protocol. The figure below shows the SPI interface and DDR octal example transaction at the xSPI interface.

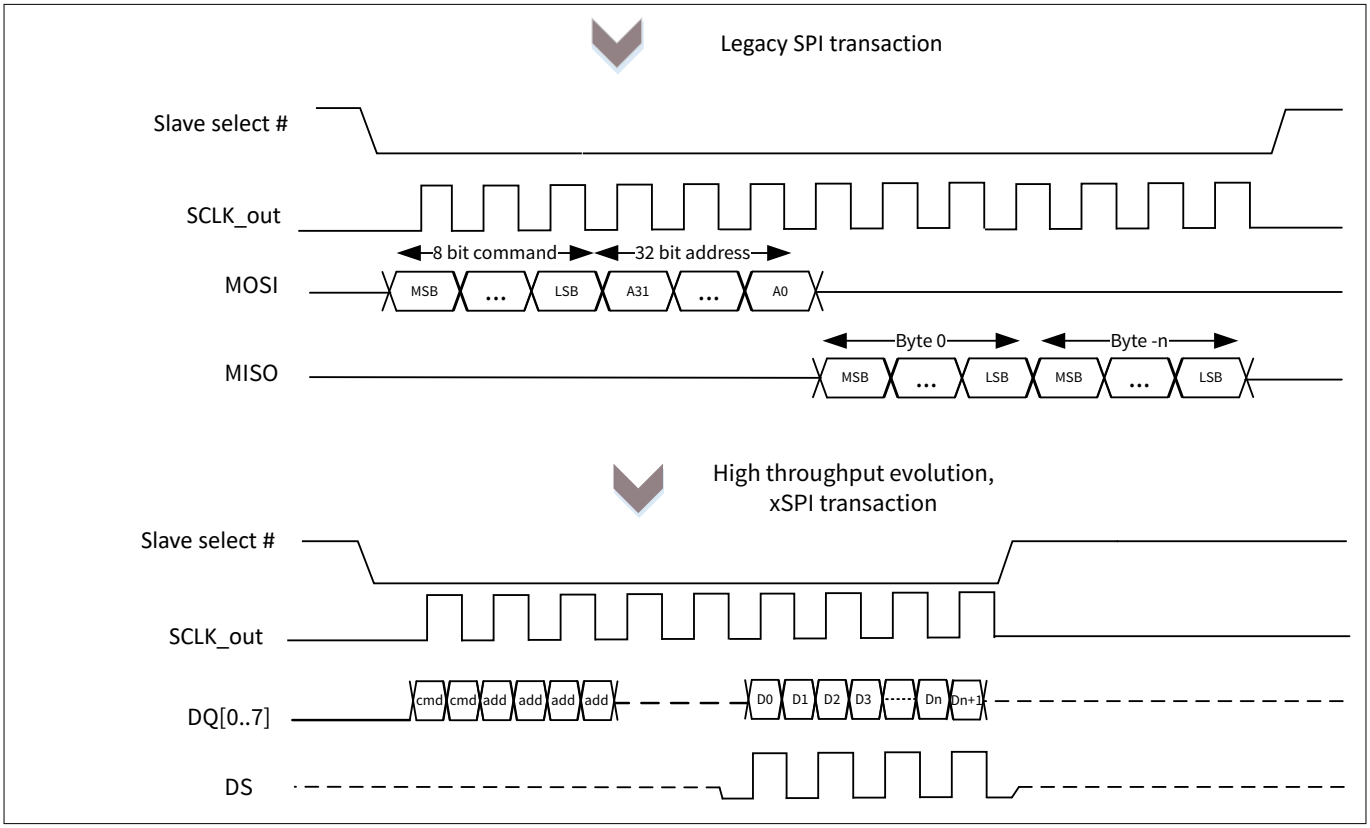


Figure 252 SPI and xSPI transaction

The following figure shows the TXD, RXD and OE signals, coming from the xSPI and HSPHY blocks, building the DQ signal. TXD is the transmit data towards the SPI slave and RXD is the receive data from the slave to the xSPI. The OE is an output enable signal which controls the IO output driver of data since this is half duplex interface.

The DLL is used to adjust the phase of the clock signal with respect to receive and transmit data signal, to generate the right set-up and hold timings for the xSPI interface. The selection of receive and transmit skew, with respect to the clock is done using the corresponding register bits TXCFG and RXCFG respectively in the HSPHY functional block

***Note**: TXD, RXD and, OE are shown here only for better understanding and are already taken care of by the host controller.*

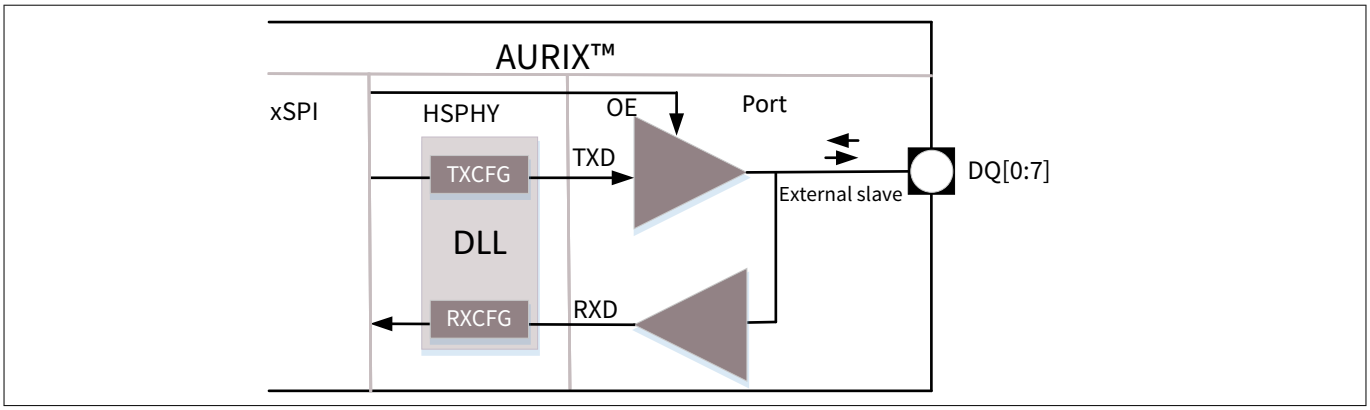


Figure 253 xSPI IO signaling

The xSPI interface provides a low pin count master-slave interface for high speed communication with JESD251 compliant non volatile memory devices, hyperbus compliant memories and is backward compliant to serial peripheral interface (SPI) devices.

In order to cater to different slave devices from different vendors available in the market, xSPI has the following configurable parameters:

- SPI Clock polarity (CTRLR0.SCPOL): Parameter determines whether the inactive state of the serial clock is high or low.
- SPI Clock Phase (CTRLR0.SCPH):
  - When CTRLR0.SCPH=0, data transmission begins on the falling edge of the slave select signal. The first data bit is captured by the master and slave peripherals on the first edge of the serial clock; therefore, valid data must be present on the TXD and RXD lines prior to the first serial clock edge. Data is driven by the master and slave peripherals on the trailing edge of the serial clock
  - When the configuration parameter CTRLR0.SCPH=1, both master and slave peripherals begin transmitting data on the first serial clock edge after the slave select line is activated. The first data bit is captured on the second (trailing) serial clock edge. Data is driven by the master and slave peripherals on the leading edge of the serial clock

The figure below shows a timing diagram for a single SPI data transfer. In the upper part of the diagram CTRLR0.SCPH = 0 (clock phase). In the lower part of the diagram CTRLR0.SCPH = 1 (clock phase). The serial clock polarity is shown for configuration parameters CTRLR0.SCPOL= 0 and CTRLR0.SCPOL= 1. To transmit data, both controller and communication partner or slave must have identical serial clock phase (CTRLR0.SCPH) and clock polarity (CTRLR0.SCPOL) values as the host. The data width can be 4 to 32 bits.

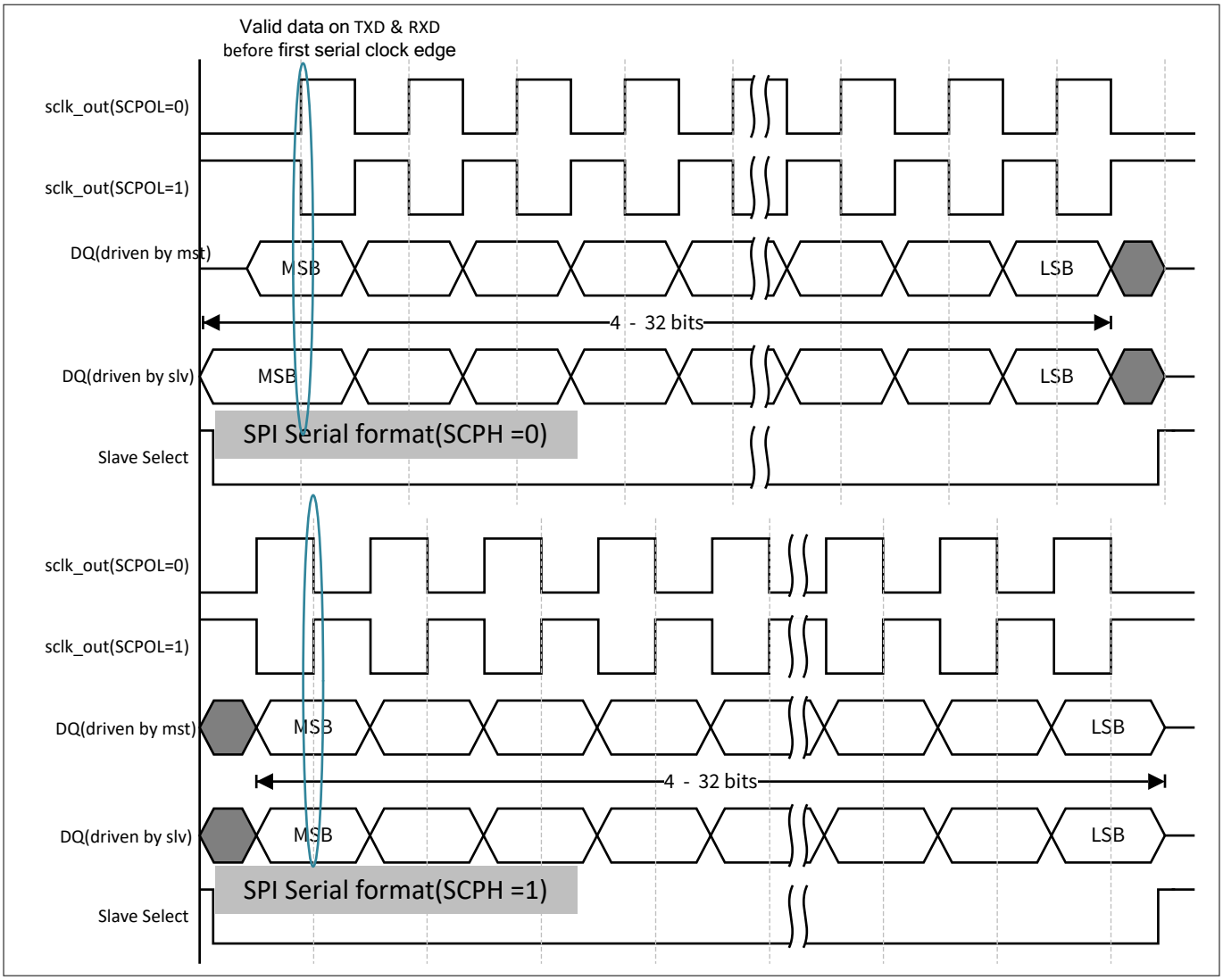


Figure 254 SPI serial format with SCPH = 0 (above) and SCPH = 1 (below)

Continuous data frames are transferred in the same way as single frames, with the MSB of the next frame following directly after the LSB of the current frame. The slave select signal is held active low for the duration of the transfer.

18.3.3 Transfer modes

This section discusses the different modes in which xSPI operates when transferring data on the serial bus. The transfer mode (TMOD) is set by writing to control register 0 (CTRLR0).

Transmit and receive

When TMOD = 00_B, both transmit and receive logic are valid. The data transfer occurs as normal, according to the selected frame format (serial protocol). Transmit data are popped from the transmit FIFO and sent through the TXD line to the target device, which replies with data on the RXD line. The receive data from the target device is moved from the receive shift register into the receive FIFO at the end of each data frame.

Transmit only

When TMOD = 01_B, the receive data is invalid and is not stored in the receive FIFO. The data transfer occurs as normal, according to the selected frame format (serial protocol). Transmit data are popped from the transmit FIFO and sent through the TXD line to the target device, which may reply with data on the RXD line. At the end of

the data frame, the receive shift register does not load its newly received data into the receive FIFO. The data in the receive shift register is overwritten by the next transfer. Interrupts originating from the receive logic should be masked when this mode is entered.

Receive only

When TMOD = 10_B, receive only mode is selected and the transmit data is invalid. The TXD outputs remain at a constant logic level during the transmission. The data transfer occurs as normal, according to the selected frame format (serial protocol). The receive data from the target device is moved from the receive shift register into the receive FIFO at the end of each data frame. Interrupts originating from the transmit logic should be masked when this mode is entered.

EEPROM read

When TMOD = 11_B, the transmit data is used to transmit an opcode and/or an address to the EEPROM device. Typically this takes three data frames (8-bit opcode followed by 8-bit upper address and 8-bit lower address). During the transmission of the opcode and address, no data is captured by the receive logic (as long as the xSPI master is transmitting data on its TXD line, data on the RXD line is ignored). The xSPI master continues to transmit data until the transmit FIFO is empty. Therefore, you should have enough data frames in the transmit FIFO to supply the opcode and address to the EEPROM. If more data frames are in the transmit FIFO than are needed, then read data is lost.

When the transmit FIFO becomes empty (all control information has been sent), data on the receive line (RXD) is valid and is stored in the receive FIFO; the TXD output is held at a constant logic level. The serial transfer continues until the number of data frames received by the xSPI master matches the value of the NDF field in the CTRLR1 register + 1.

18.3.4 Data transfers

Data transfers are started by the serial-master device when:

- xSPI is enabled (SSIENR.SSIC_EN = 1) and
- Serial-slave device is enabled (SER.SER)
- Number of data entries in transmit FIFO is greater than TXFTHR field of the TXFTLR register

While actively transferring the data, the busy flag (BUSY) in the status register (SR) is set. You must wait until the busy flag is cleared before attempting a new serial transfer.

***Note**:*

- *The SR.BUSY status is not set when the data is written into the transmit FIFO. This bit gets set only when the target slave has been selected and the transfer is underway. After writing data into the transmit FIFO, the shift logic does not begin the serial transfer until a positive edge of the `sclk_out` signal is present. The delay in waiting for this positive edge depends on the baud rate of the serial transfer. Before polling the SR.BUSY status, you should first poll the SR.TFE status (waiting for 1) or wait for `BAUDR * ssi_clk` clock cycles*
- *When DFS is greater than 8-bit, the data frame is sent and received in big endian format*

18.3.5 Enhanced SPI modes

The xSPI module supports the dual, quad, and octal modes of SPI. All four combinations of the serial clock's polarity and phase are valid in this mode and it works same as in normal SPI mode.

The mode of operation (write/read) can be selected using the CTRLR0.TMOD field.

Registers related to enhanced SPI modes

- CTRLR0
- SPI_CTRLR0

18.3.5.1 Read operation in enhanced SPI modes

A Dual, Quad, or Octal SPI read operation can be divided into four phases:

- Instruction phase
- Address phase
- Wait cycles
- Data phase

The following register fields are used for a read operation:

- CTRLR0.SPI_FRF - Specifies the format in which the transmission happens for the frame
- CTRLR1.NDF to indicate the number of data frames
- SPI_CTRLR0.TRANS_TYPE specifies the address and instruction transfer format
- SPI_CTRLR0.INST_L – Specifies length of an instruction (possible values for instruction length are 0, 4, 8, or 16 bits)
- SPI_CTRLR0.ADDR_L – Specifies address length
- CTRLR0.DFS – Specifies data length
- SPI_CTRLR0.WAIT_CYCLES field for programming the number of wait cycles. The wait cycles are introduced to provide time for the target slave device, to change its mode from input to output. The number of wait cycles can vary for different devices

For a READ-operation, xSPI sends instruction and control data once and waits until it receives NDF (CTRLR1 register) number of data frames and then de-asserts the slave select signal.

The following figure shows a typical read operation with value of N = 7 when in octal mode, N = 3 when in quad mode, and N = 1 when in dual mode.

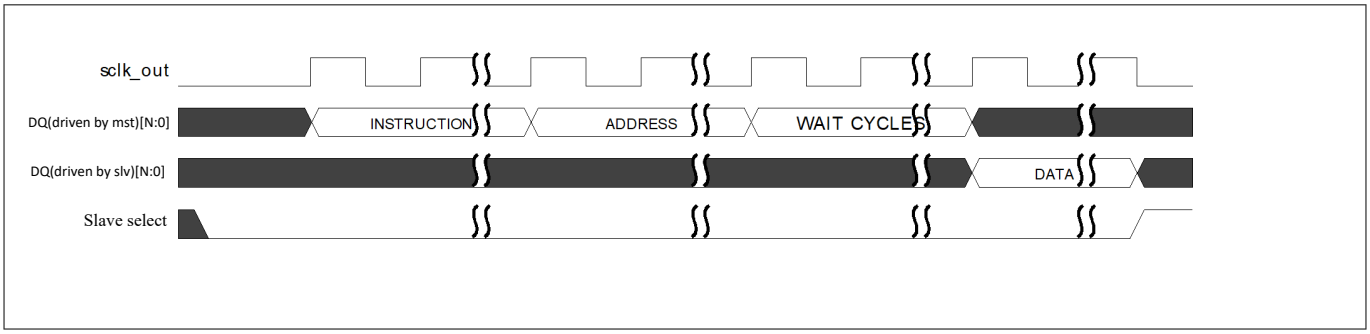


Figure 255 Typical read operation in enhanced SPI mode

Following are the possible cases of read operation in enhanced SPI modes:

1. Instruction and address both transmitted in standard SPI format: For this, SPI_CTRLR0.TRANS_TYPE field should be set to 00_B
2. Instruction transmitted in standard SPI format and address transmitted in enhanced SPI format: For this, SPI_CTRLR0.TRANS_TYPE field should be set to 01_B
3. Instruction and address both transmitted in Enhanced SPI format: For this, SPI_CTRLR0.TRANS_TYPE field must be set to 10_B
4. No Instruction, No Address READ transfer: For this, SPI_CTRLR0.ADDR_L and SPI_CTRLR0.INST_L must be set to 0 and SPI_CTRLR0.WAIT_CYCLES must be set to a non-zero value. To initiate this transfer, the software has to perform a dummy write in the data register (DRi (i=0-35).DR), xSPI waits for programmed wait cycles and fetches the amount of data specified in CTRLR1.NDF field

18.3.5.2 Write operation in enhanced SPI modes

Dual, Quad, or Octal SPI write operations can be divided into three parts:

- Instruction phase
- Address phase
- Data phase

The following register fields are used for a write operation:

- CTRLR0.SPI_FRF - Specifies the format in which the transmission happens for the frame
- SPI_CTRLR0 (SPI Control Register 0 register) – Specifies length of instruction and address
- SPI_CTRLR0.INST_L – Specifies length of an instruction (possible values for instruction length are 0, 4, 8, or 16 bits)
- SPI_CTRLR0.ADDR_L – Specifies address length
- CTRLR0.DFS – Specifies data length

An instruction takes one FIFO location and an address can take more than one FIFO location. Both the instruction and address must be programmed in the data register (DRi (i=0-35)). xSPI waits until both have been programmed to start the write operation.

The CTRLR0 and SPI_CTRLR0.TRANS_TYPE bit fields are used to control whether the instruction, address and data are transmitted in standard, dual, quad or octal mode.

***Notes**:*

1. *If CTRLR0.SPI_FRF is selected to be Standard SPI Format, everything is transmitted in Standard SPI mode and SPI_CTRLR0.TRANS_TYPE field is ignored*

2. *CTRLR0.SPI_FRF is only applicable if CTRLR0.FRF is programmed to 00_B*

The following figure shows a typical write operation with value of N = 7 when in octal mode, N = 3 when in quad mode and N = 1 when in dual mode.

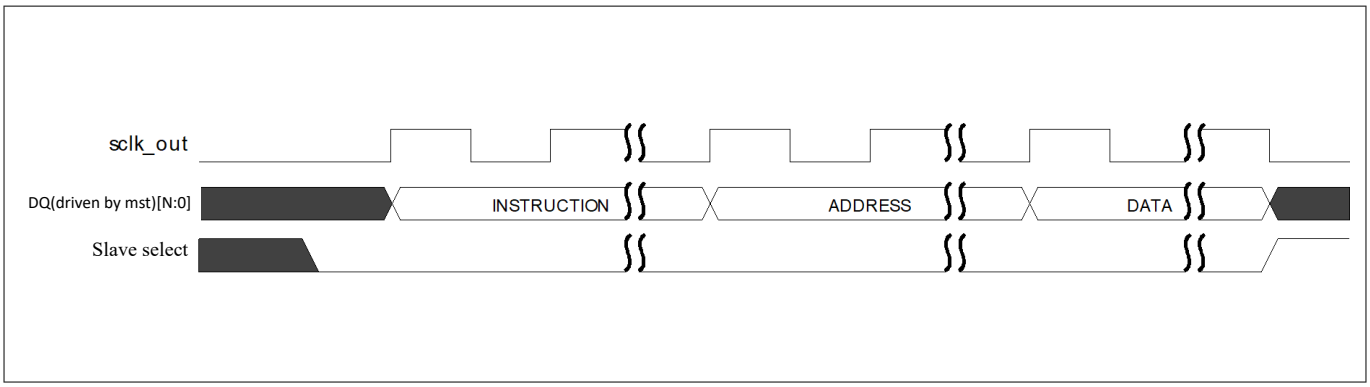


Figure 256 Typical write operation dual/quad/octal SPI mode

Following are the possible cases of write operation in enhanced SPI modes:

1. Instruction and address both transmitted in standard SPI format: For this, SPI_CTRLR0.TRANS_TYPE field must be set to 00_B

2. Instruction transmitted in standard SPI format and address transmitted in enhanced SPI format: For this, SPI_CTRLR0.TRANS_TYPE field must be set to 01_B

3. Instruction and Address both transmitted in enhanced SPI format: For this, SPI_CTRLR0.TRANS_TYPE field must be set to 10_B

4. Instruction only transfer in enhanced SPI format

***Note**:* *Some of the devices require instruction and address only transfer (without data phase). This could be achieved by skipping the address phase, and transmitting the address as data to the device. For this, SPI_CTRLR0.ADDR_L should be programmed to 0, and CTRLR0.DFS should be programmed with the required length.*

18.3.6 Clock stretching in enhanced SPI mode

xSPI includes clock stretching feature in enhanced SPI modes which can be used to prevent FIFO underflow and overflow conditions while transmitting or receiving the data respectively.

While xSPI is transmitting or receiving the data from SPI device, the software may not be able to keep up with the transfer rate due to the bandwidth issues on the slave interface. In such cases, TX FIFO becomes empty or RX FIFO overflows while transmitting and receiving the data. To avoid data corruption, CPU must discard the current operation and start a new transfer. This process is time consuming and inefficient.

To handle such scenarios, clock stretching support has been added in xSPI.

- For write transactions, whenever transmit TX FIFO becomes empty, xSPI does not de-select the slave. Instead, it masks the clock until the new data is pushed into the TX FIFO. Hence, the transfer is not broken, and CPU intervention is not required.
- For read transactions, similar to the write transaction, whenever xSPI detects that RX FIFO is full, the clock is masked until the data is read from FIFO.

The masking in this context means keeping the sclk_out at the default value, the sclk_out will remain at HIGH or LOW state based on the CTRLR0.SCPOL value. If clock stretching feature is enabled for all the write transactions, you should program the number of data frames in the CTRLR1 register. If RX_SAMPLE_DELAY register is enabled for receive transaction, then xSPI estimates when RX FIFO may become full and masks `sclk_out` accordingly.

***Note**: Clock stretching feature is applicable only for enhanced SPI transfer mode except XIP transfers.*

18.3.7 RXD sample delay

The xSPI has additional logic in order to delay the default sample time of the RXD signal(s). This additional logic can help to increase the maximum achievable frequency on the serial bus. This feature is particularly useful while working with a slave that does not support Data strobe.

Round trip routing delays on the `sclk_out` signal from the master and the RXD signal from the slave can mean that the timing of the RXD signal—as seen by the master—has moved away from the normal sampling time. The following figure illustrates this situation.

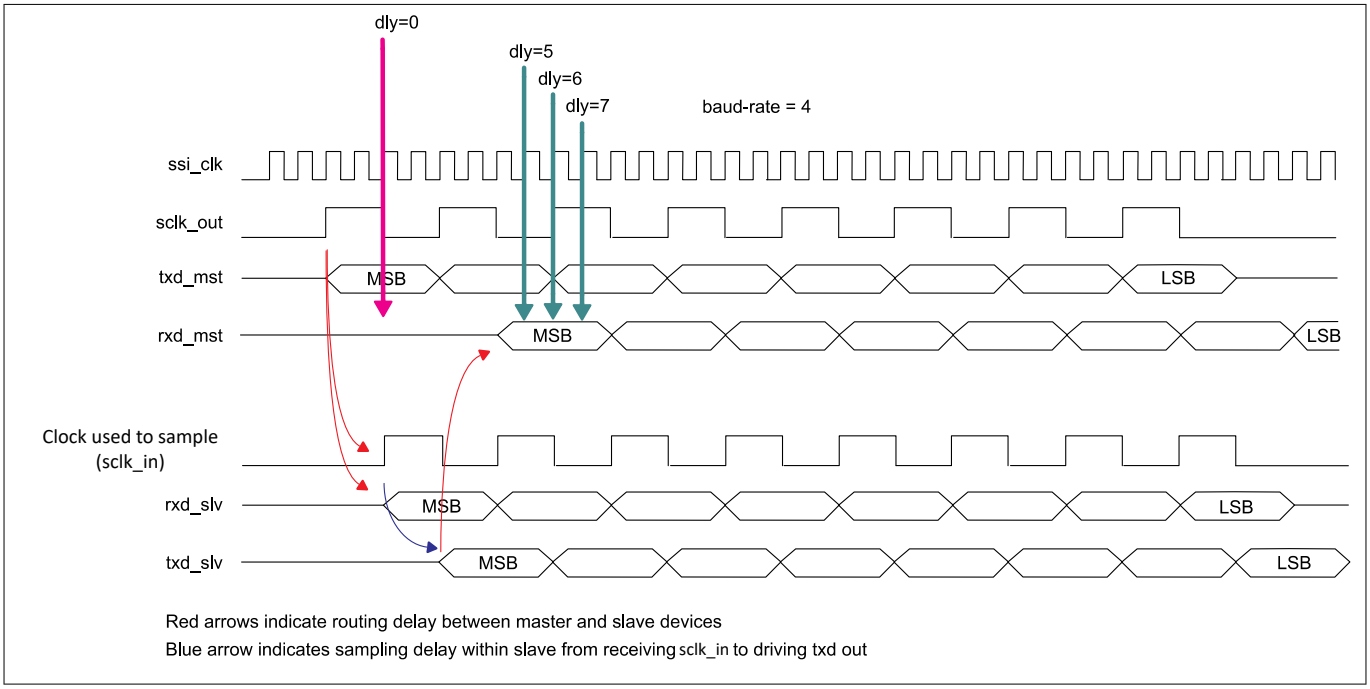


Figure 257 Effects of round-trip routing delays on the `sclk_out` signal

The slave uses the sclk_out signal from the master as a strobe in order to drive RXD signal data onto the serial bus. Routing and sampling delays on the sclk_out signal by the slave device can mean that the RXD bit has not stabilized to the correct value before the master samples the RXD signal. Figure above shows an example of how a routing delay on the RXD signal can result in an incorrect RXD value at the default time when the master samples the port.

Without the RXD Sample Delay logic, you must increase the BAUDR register value , that is increasing the dividing factor for the transfer in order to ensure that the setup times on the RXD signal are within range; this results in reducing the frequency of the serial interface.

When the RXD Sample Delay logic is used, you can dynamically program a delay value in order to move the sampling time of the RXD signal equal to a number of ssi_clk cycles from the default. The RXD sample delay logic can be configured to use positive edge of ssi_clk to sample RXD signal. This could increase the number of sampling points within a sclk_out period and help in meeting timings in higher frequencies.

xSPI uses RX_SAMPLE_DELAY register to change the sampling point of RXD signal.

- When the RX_SAMPLE_DELAY.SE is set to 0, xSPI delays the sampling point by the programmed number of ssi_clk cycles i.e, RX_SAMPLE_DELAY.RSD. Sampling is performed on the rising edge of the ssi_clk
- If RX_SAMPLE_DELAY.SE is set to 1, then xSPI delays sampling point by programmed number of ssi_clk cycles + 0.5 * (ssi_clk period). Thus, the sampling is done on negative edge of ssi_clk signal as described in the figure below. This gives user more sampling points within single sclk_out clock period.

***Note**: Using Rx sample delay it is important to arrive at a delay that covers all PVT variations. Data training during boot is also another option.*

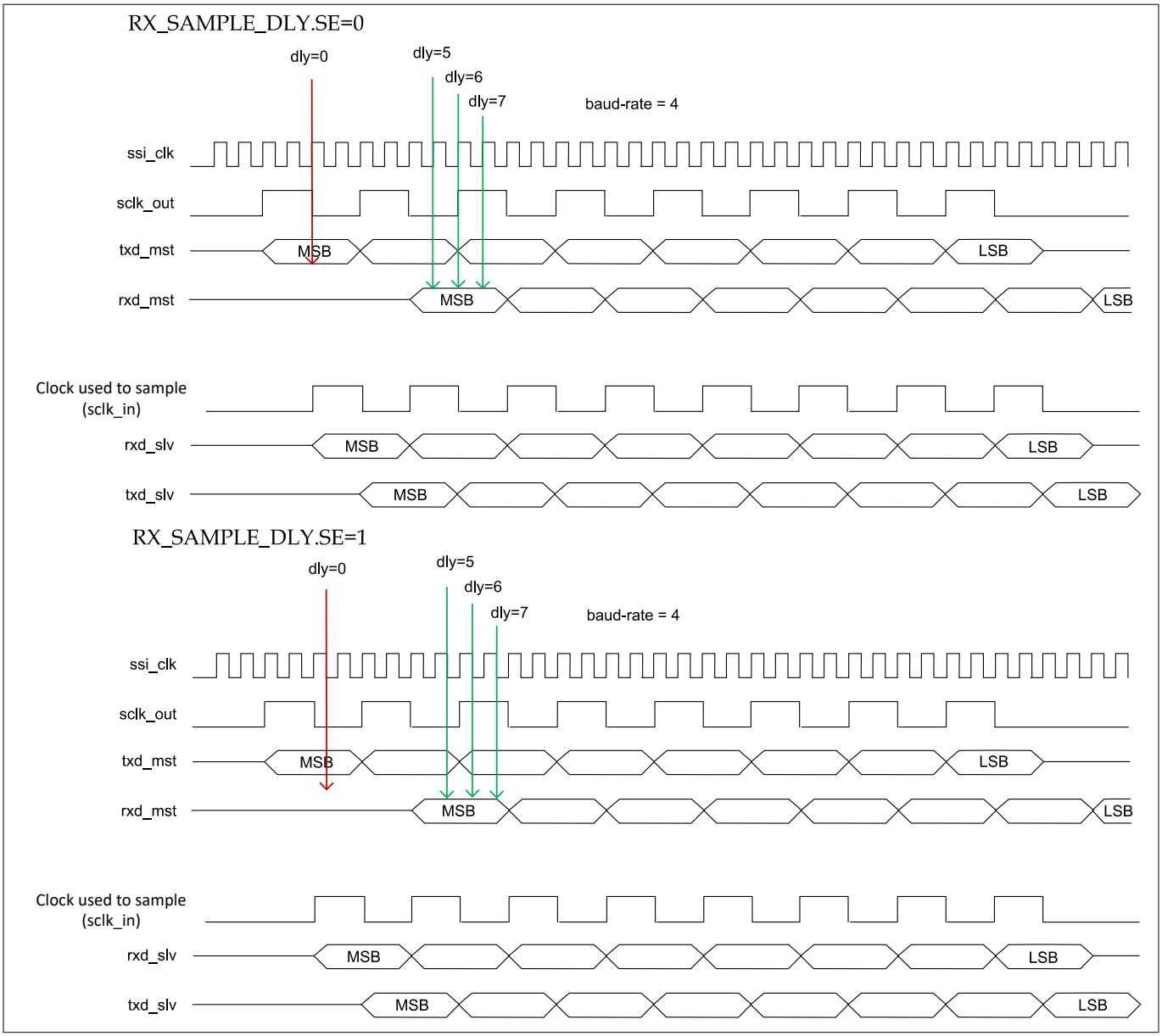


Figure 258 Sampling of RXD signal by xSPI

Registers related to RXD sample delay

- RX_SAMPLE_DELAY

18.3.8 Loopback clock sampling support

For the quad SPI mode of operation, there is no one standard that is guiding the behavior and signaling between the host and slave devices. A few slave devices support data strobe during quad mode and some, not. xSPI interface supports slaves with data strobe in quad mode. For the devices that support quad SPI mode with data strobe, achieving higher speeds (clock speeds of 133 MHz and more) is less of a challenge compared to devices that do not.

In cases where no data strobe support can be expected from the slave, it is already a challenge to reach frequencies above 50 MHz while reading a data. In order to stretch the achievable frequency, the incoming receive data could be sampled by two sample clocks as shown in figure below:

1. The transmit clock SCLK_out delayed by a configurable number of clock periods using the feature RXD sample delay available in xSPI kernel. The maximum frequency reachable in this case is limited to 66 MHz for 1.8 V devices and 83 MHz for 3 V devices
2. The looped back transmit clock , as shown in the figure as 'clock feedback'. This feature compensates the on-chip variations, on-chip delays, pad variations, pad delays and non-linearities introduced by Delay Lock Loop(DLL). This does not compensate non-linearities and delays due routing lines and slave device. This mode of compensation helps us achieve up to 133 MHz in SDR modes of operation when no data strobe is present

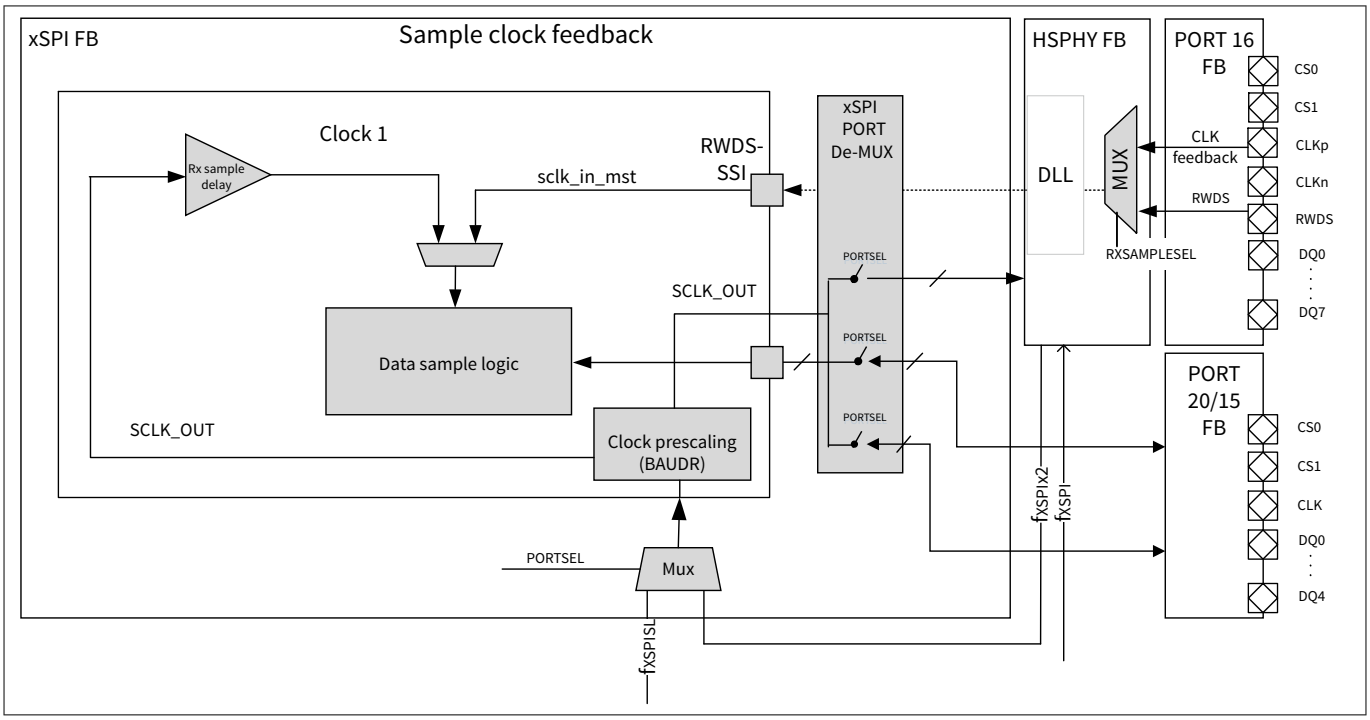


Figure 259 xSPI top level view

***Note**: When a slave with receive strobe is present, the multiplexer selection (bit RXSAMPLESEL in HSPHY register DLL_CFG) must be deactivated to avoid non-deterministic behavior*

18.3.8.1 Sampling in loopback mode

There are some slave devices that have an access time of more than half of the sclk_out period. In these cases, the user might not get an appropriate sampling window to sample the data on the positive edge of the incoming looped-back clock, Figure below shows an example for the same while using default setting, sampling with the rising edge of looped back clock, which is CTRLR0.SCPH=0.

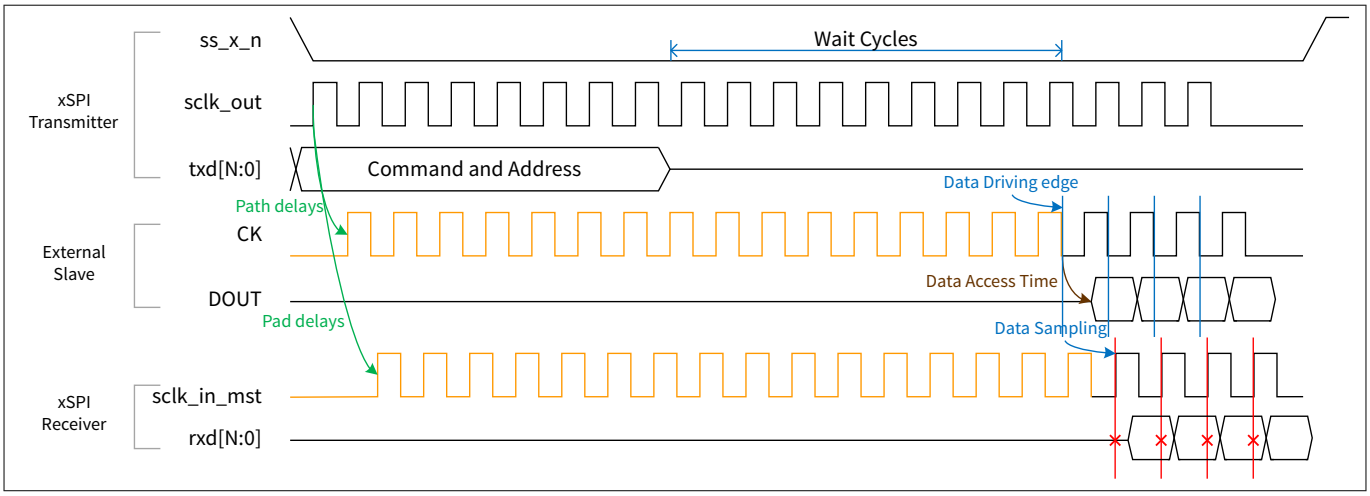


Figure 260 Clock loop back with higher access time

xSPI provides a programming option to change the sampling edge of the data in clock loopback mode as well. This enables xSPI to sample the data correctly. Figure below shows the example of the same. CTRLR0.SCPH=1 shifts the sampling to the falling edge of the looped back clock. This facilitates correct sampling of the incoming data in cases where the data access time from the slave device is more than 50% of sclk_out period. Please choose the correct sampling edge based on the access times of the slave devices.

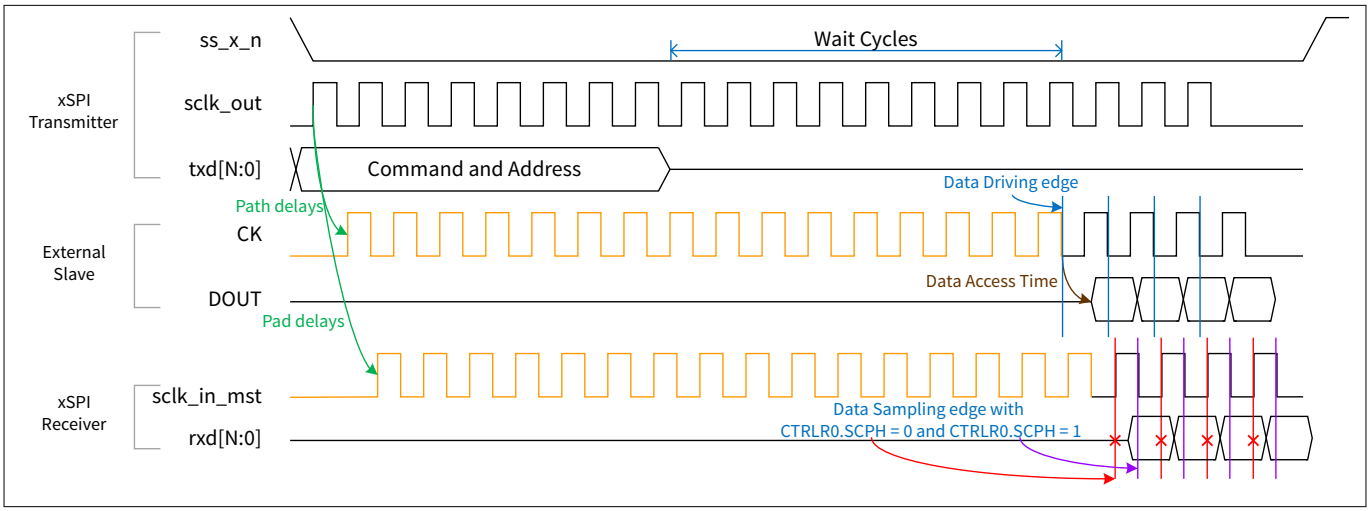


Figure 261 Clock loop back with higher access time with sampling edge variation

***Note**:*

- *The data is still transmitted in the SPI 0 format. When CTRLR0.CLK_LOOP_EN =1, CTRLR0.SCPH = 0 represents that data will be sampled with the rising edge and CTRLR0.SCPH = 1 represents that data will be sampled with the falling edge*

18.3.9 Extended quad SPI support over GPIO pads

In order to make xSPI available when high speed PORT16 is occupied by another source or FB, the xSPI functionality is extended to other standard GPIO pads as well. However, this extended xSPI support over GPIO pads has only a reduced functionality as listed below:

- support upto 25 MHz of clock
- SDR mode of operation only
- support for only 3 V external flash devices
- Only a maximum of 4 data lines are available. Octal mode of operation is not supported

In order to select the reduced functionality xSPI operation, CTRLR2.PORTSEL bit must be enabled. Since PORT16 and the DLL would be potentially occupied by RGMII, ssi_clock source must be selected to F_{XSPISL}.

***Note**:*
- *Special care must be taken to select the right clock for the right port selection*
- *xSPI cannot run in parallel on different ports. At a given time only one port can be used as we have only one instance of xSPI*

18.3.10 Dual data-rate (DDR) support

In standard operations, data transfer in SPI modes occurs on either the positive or negative edge of the clock. For improved throughput, the dual data-rate transfer can be used for reading or writing to the memories.

The DDR mode supports the following modes of the SPI protocol:

- Mode 0 – when the value of serial clock phase and serial clock polarity is 0_B (CTRLR0.SCPH = 0 and CTRLR0.SCPOL = 0)
- Mode 3 – when the value of serial clock phase and serial clock polarity is 1_B (CTRLR0.SCPH = 1 and CTRLR0.SCPOL = 1)

Description of DDR support

DDR commands enable data to be transferred on both the edges of clock. Following are the different types of DDR commands:

- Address and data are transmitted (or received in case of data) in DDR mode, while instruction is transmitted in standard format
- Instruction, address, and data are all transmitted or received in DDR mode

The SPI_CTRLR0.SPI_DDR_EN bit is used to determine if the address and data are transferred in DDR mode and the SPI_CTRLR0.INST_DDR_EN (SPI_CTRLR0[17]) bit is used to determine if instruction is transferred in DDR mode. These bits are only valid when the CTRLR0.SPI_FRF bits are set to Dual, Quad or Octal mode.

The following figure describes a DDR write transfer where instruction is transmitted in standard format. In the figure below, the value of N is 7 when CTRLR0.SPI_FRF is set to 11_B.

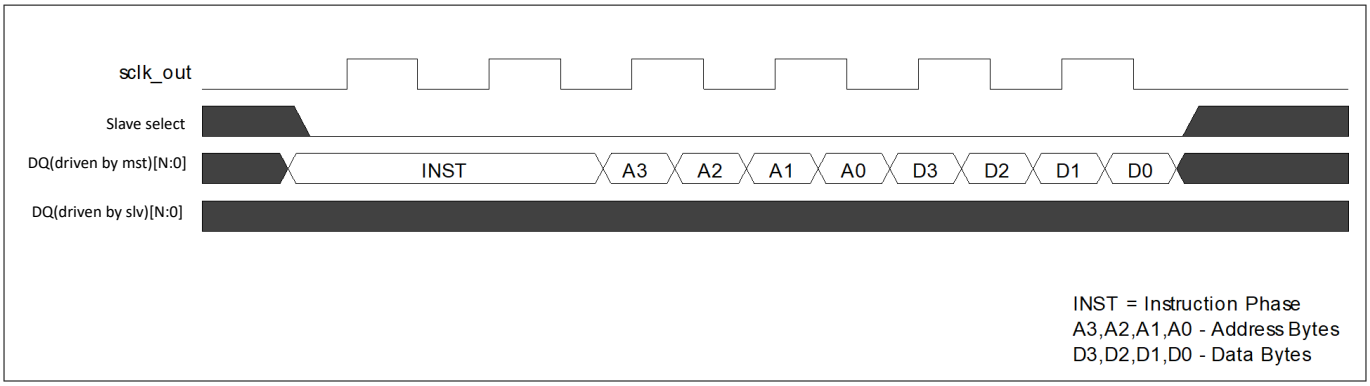


Figure 262 DDR transfer with instruction in standard format

The Figure below describes a DDR write transfer where instruction, address and data all are transferred in the DDR format.

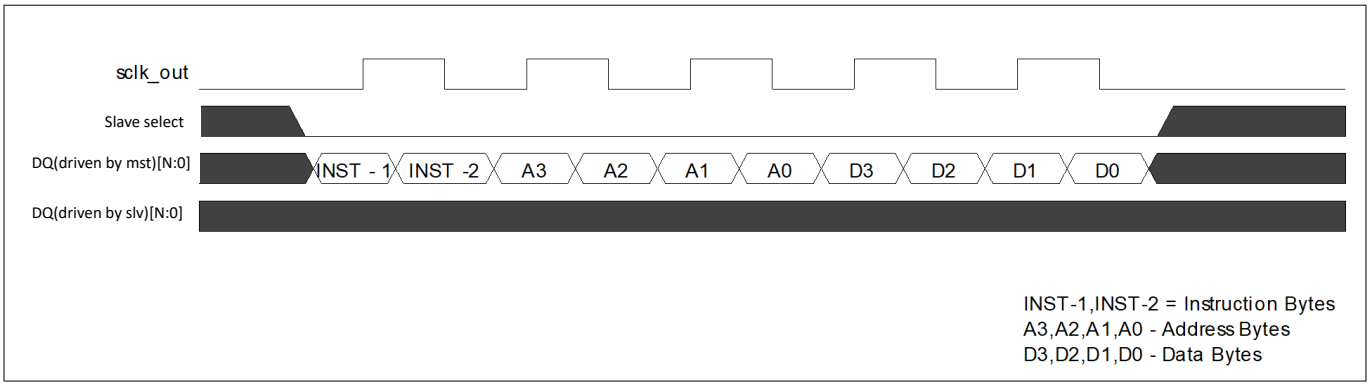


Figure 263 DDR transfer with instruction, address and data transmitted in DDR format

***Note**: In the DDR transfer, address and instruction cannot be programmed to a value of 0. The number of wait cycles cannot be 0 in a DDR read operation.*

Transmitting data in DDR mode

In DDR mode, data is transmitted on both edges. This makes it difficult to sample data correctly as the data with respect to sclk_out would have moved. xSPI uses an internal register to determine the ssi_clk edge on which the data should be transmitted. This ensures that the receiver is able to get stable data with respect to sclk_out while sampling. The internal register (DDR_DRIVE_EDGE) determines the edge on which the data is transmitted. The internal clock is divided by an integral number (ssi_clk / BAUDR) to derive the baud clock (sclk_out) which xSPI uses to send data.. The data needs to be transmitted within half clock cycle (BAUDR/2), therefore the maximum value for the DDR_DRIVE_EDGE.TDE bitfield is equal to [(BAUDR/2)-1]. Data is driven ((BAUD/2)-DDR_DRIVE_EDGE.TDE) ssi_clk before the sampling edge. If the programmed value of the DDR_DRIVE_EDGE.TDE register is set to 0 then the data is transmitted two ssi_clk before the edge of sclk_out.

The following figures show examples of how data is transmitted using different values of the DDR_DRIVE_EDGE.TDE bitfield. The green arrows in these examples represent the points where data is driven. BAUDR used in all these examples is 6.

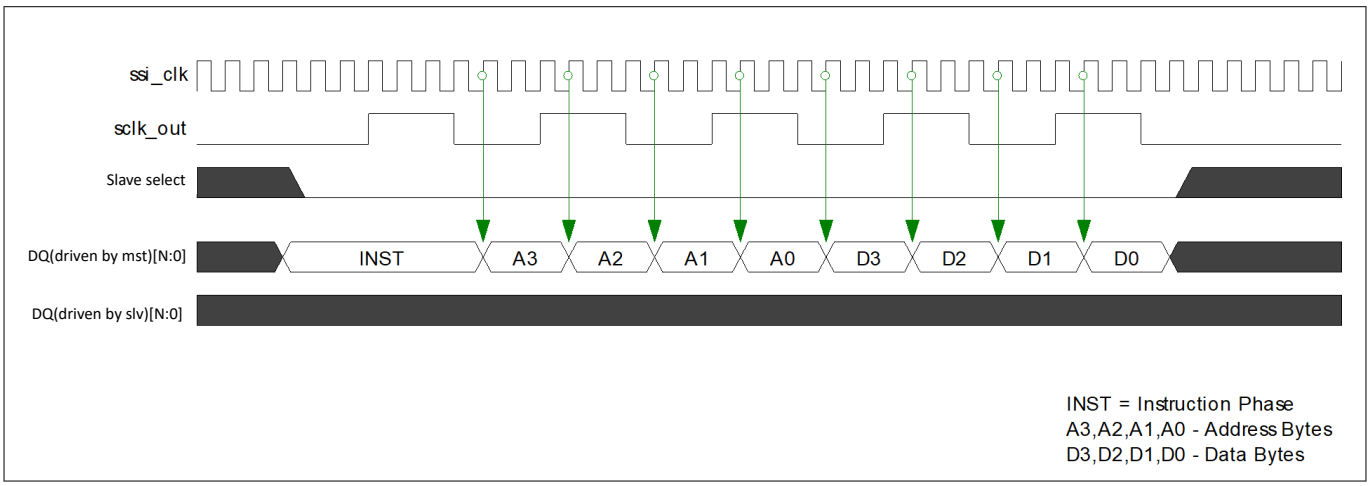


Figure 264 Transmit data with DDR_DRIVE_EDGE.TDE = 0

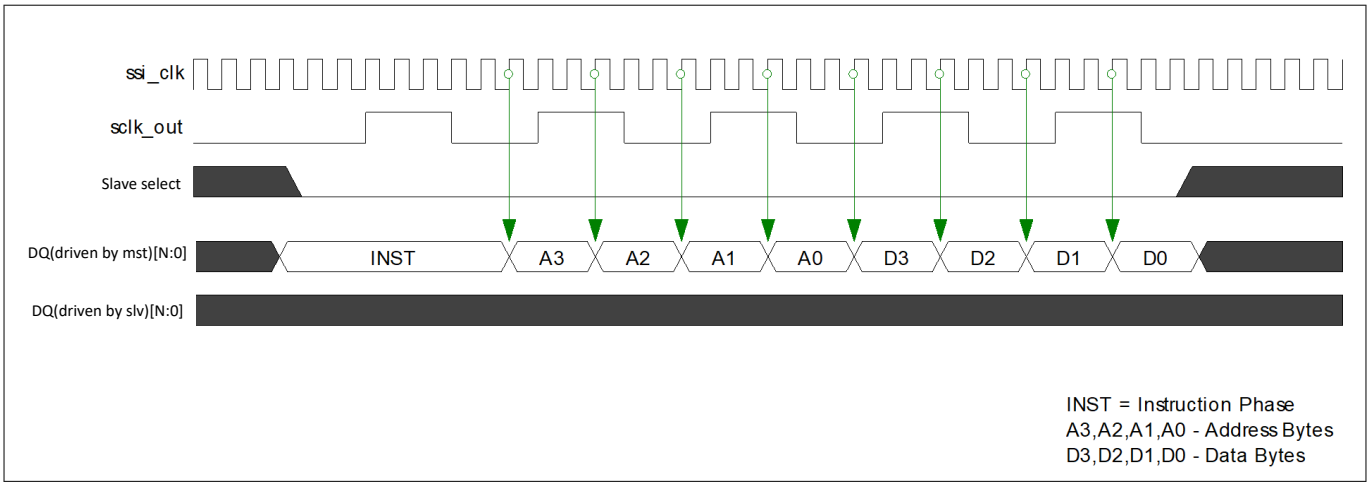


Figure 265 Transmit data with `DDR_DRIVE_EDGE.TDE = 1`

Registers related to dual data-rate (DDR)

- CTRLR0
- SPI_CTRLR0
- DDR_DRIVE_EDGE
- BAUDR

18.3.11 HyperBus protocol

A read or write transaction on a HyperBus interface consists of a sequential series of 16-bit, one-clock cycle, data transfers. Each 16-bit one-clock transfer consists of two 8-bit half-clock transfers, one on each clock edge.

xSPI supports the HyperBus protocol that works on the principle of the SPI DDR interface. The DDR protocol transfers two data bytes per clock cycle on the input/output (I/O) signals.

In the HyperBus protocol, read and write transactions transfer complete 16-bit words of data. Read data words always contain two valid bytes. Write data words may have one or both bytes masked to prevent writing of individual bytes within a write burst.

Command, address, and data information are transferred over the eight HyperBus (TXD[7:0]/RXD[7:0]) signals. The HyperBus slave device uses the clock to capture information while receiving the command, address, or data on the DQ signals. The Command or Address values are center aligned with clock transitions. The HyperBus uses a 48-bit command-address (CA) to convey the instruction and address information. The CA frame format is as shown in the table below.

Table 815 Command-address bit assignments

CA bit	Bit name	Bit function
47	R/W#	Identifies the transaction as a read or write. • 1 – Indicates a read transaction • 0 – Indicates a write transaction
46	Address space (AS)	Indicates whether the read or write transaction accesses the memory or register space. • 0 – Indicates the memory space • 1 – Indicates the register space

(table continues...)

Table 815 (continued) Command-address bit assignments

CA bit	Bit name	Bit function
45	Burst type	Indicates whether the burst is linear or wrapped. • 0 – Indicates a wrapped burst • 1 – Indicates a linear burst
44–16	Row and upper column address	Row and upper column component of the target address: System word address bits A31-A3 If a particular device density does not use upper row address bits, the host controller master interface must set it to 0. The size of Rows and therefore the address bit boundary between row and column address is dependent on the slave device
15–3	Reserved	Reserved for future column address expansion. Reserved bits are do not care in current HyperBus devices but the host controller must set this to 0 for future compatibility
2–0	Lower column address	Lower column component of the target address: System word address bits A2-A0 selecting the starting word within a half-page

Registers related to HyperBus protocol

- CTRLR0
- SPI_CTRLR0

18.3.11.1 HyperBus read

The HyperBus master begins a transaction by driving the slave select signal to low while the clock is idle. Then, the clock starts toggling while CA words are transferred. The HyperBus master then continues clocking for a number of cycles defined by the latency count setting in SPI_CTRLR0.WAIT_CYCLES. If the Read-Write Data Strobe (RWDS) signal, that is the slave counterpart of rxds signal for HyperBus devices, is low during the CA cycles, one latency count is inserted. If RWDS is high during the CA cycles, an additional latency count is inserted. After these latency clocks are completed, the slave transitions the RWDS and outputs the target data.

***Note**: Certain HyperBus devices can remove read data strobe signal in between transfers if a page boundary is reached. In such a situation, the master must send the clock continuously until the transfer resumes. xSPI does not support such transfer and the software must ensure that such situations do not occur before initiating the transfer by xSPI.*

18.3.11.2 HyperBus write

The HyperBus master begins a transaction by driving the slave select signal to low while the clock is idle. Then, the clock starts toggling while CA words are transferred. Further, the HyperBus master continues clocking for a number of cycles defined by the latency count setting in a configuration register SPI_CTRLR0. If RWDS is low during the CA cycles, one latency count is inserted. If RWDS is High during the CA cycles, an additional latency count is inserted. After these latency clocks are completed, the HyperBus master starts to output the target data. The Write data is center aligned with the clock edges. The first byte of data in each word is captured by the memory on the rising edge of `sclk_out` signal and the second byte is captured on the falling edge of `sclk_out` signal. The latency count can also be zero for a write transaction that is device dependent. For a HyperBus device, the read data strobe signal is driven by the slave device during the CA phase .

18.3.11.3 HyperBus enable

The CTRLR0.SPI_HyperBus_EN register bit is set to program xSPI as a HyperBus master. As the HyperBus interface has a fixed frame format, the following fields are fixed when xSPI is programmed into HyperBus mode:

- CTRLR0.SPI_FRF – 11_B (octal frame format)
- SPI_CTRLR0.TRANS_TYPE – 10_B (Address to be sent in octal format)
- SPI_CTRLR0.ADDR_L – 1100_B (address length of 48 bits)
- SPI_CTRLR0.INST_L – 00_B (no instruction phase)
- SPI_CTRLR0.SPI_DDR_EN – 1 (Address to be sent on octal-DDR mode)
- SPI_CTRLR0.SPI_RXDS_EN – 1 (enable read data strobe to sample incoming data)

Wait cycles are configured using the SPI_CTRLR0.WAIT_CYCLES field, depending on memory requirement. As the HyperBus protocol counts the wait cycles from the third CA cycle, xSPI sends SPI_CTRLR0.WAIT_CYCLES-1 wait cycles after the CA phase is complete.

If the SPI_CTRLR0.SPI_RXDS_SIG_EN bit is set to 0 or if the rxds level is 0 during CA phase, the number of wait cycles is fixed to the programmed value in the SPI_CTRLR0.WAIT_CYCLES bit-field.

You can set the type of transaction (read/write) and data frame size in CTRLR0.TMOD and CTRLR0.DFS fields, respectively. The read back value of fixed fields listed above is same as the register value but internally the hardware ignores those when CTRLR0.SPI_HyperBus_EN bit is set to 1.

After CTRLR0.SPI_HyperBus_EN is set to 1, xSPI expects a 48-bit address to start the transaction. The 48-bit data must be packed according to the HyperBus specification.

18.3.11.4 RXDS signaling

The slave HyperBus device can use for its transactions either a fixed or variable Latency.

- Variable latency: In case of variable latencies, xSPI provides an option for RXDS signaling in the CA phase as required by the HyperBus slave. One can enable this feature by programming the SPI_CTRLR0.RXDS_SIG_EN bit to 1. In this mode, if the RXDS signal is detected as high during command and address cycles, then xSPI sends (2*SPI_CTRLR0.WAIT_CYCLES-1) wait cycles after the address phase is complete. If the RXDS signal is detected low, then xSPI sends (1*SPI_CTRLR0.WAIT_CYCLES-1) wait cycles after the address phase is complete. Hence ideally, SPI_CTRLR0.WAIT_CYCLE should be programmed to the minimum latency cycles needed by slave
- Fixed latency: If the SPI_CTRLR0.RXDS_SIG_EN bit is set to 0 or if the RXDS level is 0 during CA phase, the number of wait cycles is fixed to the programmed value in the SPI_CTRLR0.WAIT_CYCLES-1) bit-field. This value should ideally be two times accounting also for the time needed for refresh latency for slaves that need them.

***Note**: The slave device has to be respectively configured in fixed or variable latency mode based on host configuration and vice versa*

18.3.12 Execute in place (XIP) mode

xSPI provides a function to directly perform memory read operation from AHB transaction. This is called execute in place mode, in which xSPI acts as memory mapped interface to an SPI memory.

XIP usage model

XIP operations are supported only in Dual, Quad, or Octal enhanced SPI modes of operation (or in hyperbus mode as described in the following section), and hence, the CTRLR0.SPI_FRF bit must not be programmed to 0. Typically, an XIP operation consists of an address phase and a data phase. The programming flow to set-up an XIP transfer is as follows:

1. Set the SPI frame format value in CTRLR0 register
2. Set the address length, wait cycles, and transaction type in SPI_CTRLR0

The instruction phase can also be included in XIP transfer by using XIP_INST_EN field of the SPI_CTRLR0 register. In this case, the following registers must be set:

1. Set length of instruction in the SPI_CTRLR0 register
2. Write the instruction opcodes in the XIP_INCR_INST and XIP_WRAP_INST registers

After the programming is completed, an XIP read transaction can be initiated by reading the xSPI memory mapped region. A read transaction is then completed using the programmed values.

When the XIP_DFS_HC field of the SPI_CTRLR0 register is set to 0, SRI control signals are used to derive the values of the data frame size and the number of data frames to be fetched by the device. Once the CTRLR0 and SPI_CTRLR0 registers are programmed with correct values for the transfer, then XIP transfers can be initiated by reading from the memory mapped xSPI address region.

Note:
- *Only a maximum of SRI BTR4 is supported at the AHB slave*
- *Memory Address regions for XIP access can be found in the device specific Memory overview table*

XIP transfers in HyperBus mode

For XIP transfers, the 32-bit address is obtained from the AHB interface and then packed to derive the 48-bit CA frame format required by the Hyperbus device. Because Hyperbus memories are word (16-bit) addressable, the least significant bit is not considered while packing the address in CA frame format.

Mode bits support

During a serial transfer, certain devices have the capability to send additional bits that are referred to as Mode bits. In xSPI, the mode bit support can be enabled by setting the XIP_MD_BIT_EN field of the SPI_CTRLR0 register. The actual value of the mode bits to be driven can be programmed by writing into the XIP_MODE_BITS register. The length of the mode bits can be programmed using XIP_MBL field of the SPI_CTRLR0 register. Mode bits occur immediately after the address phase is completed and follow the same rules as for the address bits. Therefore, serial transfers consist of following phases when mode bit support is enabled:

Instruction Phase > Address Phase > Mode Bits > Wait cycles > Data Phase

Based on the CTRLR0.SPI_FRF value, mode bits are driven as follows:

- If CTRLR0.SPI_FRF is dual, the mode bits are sent on two lanes
- If CTRLR0.SPI_FRF is quad, the mode bits are sent on four lanes
- If CTRLR0.SPI_FRF is octal, the mode bits are sent on eight lanes

XIP mode with instruction phase

Certain devices expect the instruction phase to exist during XIP transfers. xSPI supports inclusion of some fixed instruction sets during the XIP mode of operation. By default, the instruction phase feature is not enabled in XIP mode. To enable this feature, set the XIP_INST_EN bit of the SPI_CTRLR0 register. The length of instruction phase is controlled by the INST_L field of the SPI_CTRLR0 register. When the XIP_INST_EN bit of the SPI_CTRLR0 register is set to 1 and xSPI receives an XIP transfer, then the instruction opcode is also sent to the device followed by the address.

Registers related to XIP mode

- CTRLR0
- SPI_CTRLR0
- XIP_WRAP_INST
- XIP_INCR_INST
- XIP_MODE_BITS

18.3.12.1 Continuous transfer mode in XIP

When xSPI receives a memory mapped address read request on the slave, the address is transmitted onto the SPI interface directly. Each new transfer (XIP Read) is treated in the same manner. Therefore, for every request, a new address must be sent to the device thereby contributing to the latency of the system.

If a memory device allows the stretching of slave select signal across the XIP read transfers, then xSPI can be programmed for continuous XIP mode to achieve higher performance. In this mode, the host combines two or more burst requests into a single SPI command by ensuring that the command and address are not retransmitted and the host controller need not wait for any dummy cycles in between these bursts.

xSPI can be enabled for continuous XIP mode by programming the SPI_CTRLR0.XIP_CONT_XFER_EN bit to 1. When this bit is set to 1, then xSPI functions in continuous XIP mode as soon as first XIP command is received. For the first XIP transfer, the address (and instruction if SPI_CTRLR0.XIP_INST_EN = 1) is sent on the SPI interface. After reception of requested data, xSPI continues to keep the slave selected and the clock (sclk_out) remains in the default state. For subsequent XIP transfers , xSPI resumes the clock (sclk_out) and neither the command nor the address is retransmitted onto the SPI interface. The data is fetched from the device immediately (no dummy cycles).

xSPI moves out of the continuous mode due to either of the following conditions:

- A non-XIP command (example, an SFR register space access) is received on an XIP interface
- When non-consecutive addresses are issued, the slave select is removed and xSPI initiates a new XIP request
- When the xSPI does not detect any XIP transfer for the time-period specified in XIP_CNT_TIME_OUT register the slave is de-selected

***Notes**:*

1. *In Continuous transfer mode, if SPI_CTRLR0.XIP_DFS_HC is set to 1, the XIP address must always be aligned with the DFS value programmed in the CTRLR0.DFS bit*
2. *Continuous XIP mode will not work when clock stretching is enabled*

xSPI supports the following continuous transfers:

- Continuous incremental transfer
- Continuous wrap transfer

Continuous incremental transfer

A transfer is defined as continuous incremental when the first command received in the continuous XIP (SPI_CTRLR0.SSIC_XIP_CONT_XFER_EN = 1) mode is INCR . In this mode, when the transfer is completed, the next command must also be incremental, and the address must be a contiguous address of the last burst.

Continuous wrap transfer

A transfer is defined as continuous WRAP when the first command received in continuous XIP mode is WRAP. In this mode, when the transfer is completed, the next command must start from the address of last-burst address boundary.

A continuous XIP transfer can be ended by issuing non-XIP transfer, or when a XIP read is issued to a non-consecutive address. In this case, the slave select is removed, and then xSPI initiates a new XIP request.

Data sampling in continuous transfer mode

The data sampling in continuous transfer mode depends on the SPI protocol type (SCPH and SCPOL values). The different cases for all the combinations are listed as follows:

1. Case A: CTRLR0.SCPH = 0.

In this case, when the last continuous transfer completes, the next data is sampled on the first edge of clock. The polarity of the clock during the IDLE period is dependent on the CTRLR0.SCPOL value

2. Case B: CTRLR0.SCPH = 1.

   In this case, when the last continuous transfer completes, the next data is driven at the first edge and sampled on the second edge of the clock. The polarity of the clock during the IDLE period is dependent on the CTRLR0.SCPOL value

***Note**: For dual data-rate transfers (SPI_CTRLR0.SPI_DDR_EN = 1), the continuous transfer mode is supported only for CTRLR0.SCPH = 0 and CTRLR0.SCPOL = 0 configurations, and the remaining combinations are not supported.*

Slave deselect in continuous transfer mode

During the continuous transfer, a lot of power is dissipated on the slave device since the slave is selected all the time. To reduce the slave power consumption, xSPI provides a configuration option to enable a watchdog timer to de-select the slave after the counter runs out. The counter is based on the AHB slave clock frequency, which is $f_{SRI/2}$.

When the xSPI does not detect any XIP transfer for the time-period specified in XIP_CNT_TIME_OUT register the slave is de-selected. The counter starts its operation after the last XIP transfer is completed, and then waits for a new request on AHB slave interface. Register XIP_CNT_TIME_OUT is used to set the count-down value for the IDLE period. When set to 0 the counter is disabled, and xSPI continues to wait for the next XIP request without de-selecting the slave.

Registers related to continuous transfer mode in XIP

- CTRLR0
- SPI_CTRLR0
- XIP_CNT_TIME_OUT

18.3.13 Read data strobe signal

To achieve high frequency, data strobe signaling is used in read operations. The slave device provides a data strobe known as read data strobe (RXDS), signal along with read data, which determines the output data valid window. The frequency of this signal follows the frequency of the peripheral clock driven by xSPI, sclk_out. xSPI must receive the read data, which is center-aligned to the edge of data strobe. The slave device normally provides an edge-aligned data with respect to the strobe. This means that the DLL in the HSPHY FB phase delays the data strobe so that read data is center aligned as shown in the following figure. This delay is introduced and controlled by the register DLL_CFG.RXCFG in the HSPHY functional block.

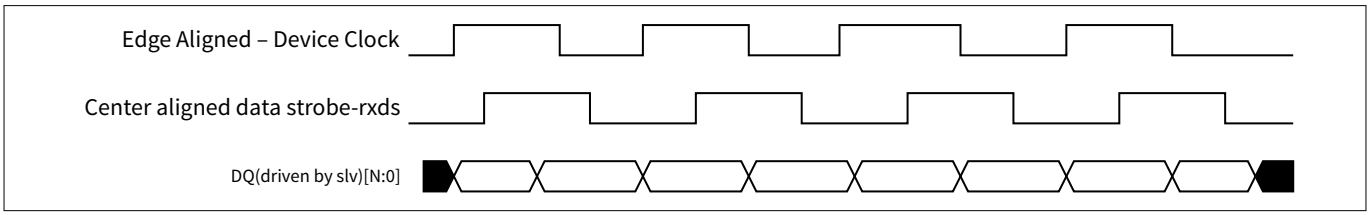


Figure 266 Center aligned data strobe for xSPI interface

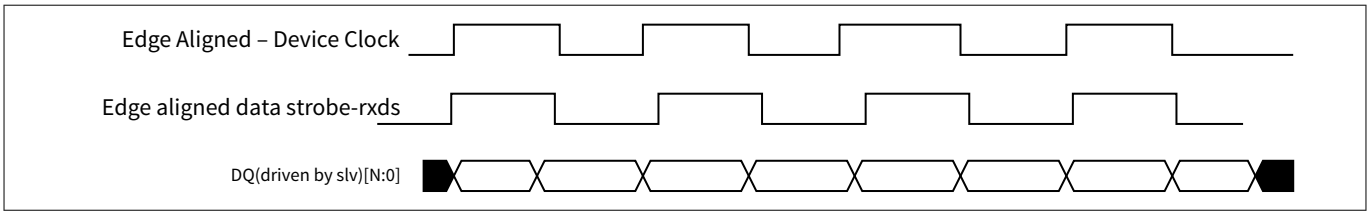


Figure 267 Edge aligned data strobe before entering DLL interface

SPI_RXDS_EN field of the SPI_CTRLR0 register is set to 1 to sample incoming data with respect to the read data strobe signal, as shown in the following figure. It is expected that the slave device uses the incoming clock, sclk_out to generate the data strobe. xSPI transmits only the minimum number of clock cycles required to complete the transfer. If the number of data frames expected is four, xSPI transmits only four clocks as shown in the following figure.

Note:

- The data must be stable around both clock edges of the data strobe. Data is sampled on both edges
- Only the SPI mode 0 is supported in this mode, that is when CTRLR0.SCPH = 0 and CTRLR0.SCPOL = 0

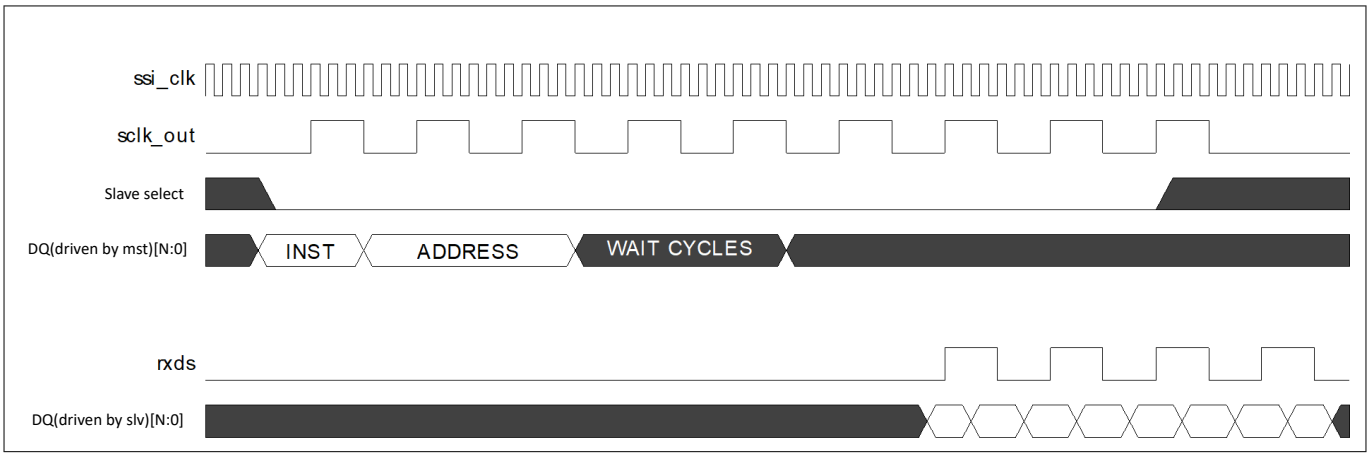


Figure 268 Example of a transfer using read data strobe signal

Registers related to read data strobe signal

- CTRLR0
- SPI_CTRLR0

18.3.14 Slave interface

The slave bus interface module or the slave bus, as referenced everywhere else in the document, implements the logic to access the internal registers of xSPI as well as perform XIP operations to the end device. This module supports only the little-endian scheme for data transfer.

***Note**: There is only one slave interface where both register accesses and XIP operations take place. Based on the addressed memory space, whether it is memory mapped external slave area or SFR register space, the transaction is either an XIP or a register access respectively.*

xSPI provides a slave error response from slave when any of the following conditions are met:

- If transmit FIFO is full and a write operation is attempted in the data register
- If receive FIFO is empty and a read operation is attempted in the data register
- If read or write operation is attempted in the data register when xSPI is not enabled (SSIENR register set to 0)
- Under XIP operation the following condition result in error response:
  - If any register write operation occurs during XIP operation

18.3.15 Data mask

xSPI supports the data mask feature that is used to mask the write data on an SPI interface. xSPI supports this feature on enhanced SPI modes. When configured in data mask mode, xSPI provides the TXD_dm signal to mask the data on the TXD line. This signal is used only during WRITE transactions in enhanced SPI DDR operations.

The SPI_DM_EN field in the register SPI_CTRLR0 is set to enable the data mask feature. The value of data mask is derived from incoming transactions on the AHB interface to data register. For example, if you want to transmit 4 bytes {D3,D2,D1,D0} (with D3 as MSB) by masking the D2 byte, you must divide the transfer into two transfers such that 16-bit transfer containing {D1,D0} is transferred and then, one 8-bit transfer with D3 being written in the data register. xSPI enables the data mask signal while transmitting the third byte in transfer.

While sending a 4 byte transfer {D3,D2,D1,D0} , that is CTRLR0=0x1f, and an AHB transfer of only one byte is received by the controller {D0} , then the remaining 3 bytes will be masked.

While sending a 2 byte transfer {D1,D0} , that is CTRLR0=0xf, when an AHB transfer of only one byte is received by the controller {D0}, then the remaining 1 byte will be masked.

Figure 269 shows a data mask used for Octal SPI write transfer. The data mask signal is active while transmitting the third byte of the data that is not written in the SPI device.

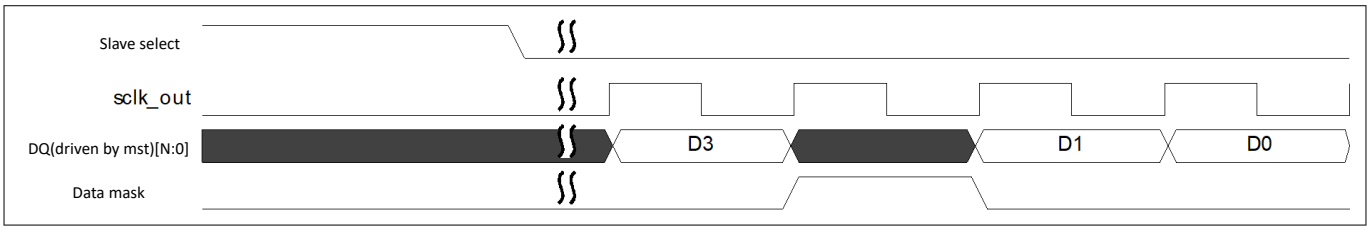


Figure 269 Octal SPI write transfer with data mask enabled

***Note**:*

- *Each FIFO location in the transmit data FIFO represents a data frame size, which can go from 4 to 32 bits. In case where the data frame size is 32, all the bits of FIFO data are valid and are sent to the SPI interface. But, if the data frame size is smaller than 32, for example 16, then, only 16 bits of each FIFO location are valid. Since the data register is 32 bits wide, all the 32 bits must be written. Among these 32 bits only 16 are valid and the data mask processing happens only for those16 bits. The reserved 16 bits must also be written and must not be skipped*
- *The data mask is only valid for the data phase of the enhanced SPI transfers. The address and instruction phases do not use the data mask. Since the address/command and the data register use the same location, you must write the address/command into the data register before starting to write the data*
- *In order to write smaller than 32 bits of data, please refer to description in CTRLR0.DFS.*

Registers related to data mask

- SPI_CTRLR0
- XIP_CTRL

18.3.16 Internal DMA feature

xSPI internal DMA feature is responsible for data transfer from an internal memory (or on chip memory) to an external slave SPI device and vice versa. xSPI uses the master interface on one side and the SPI interface on the other, acting as source or destination depending on the programming.

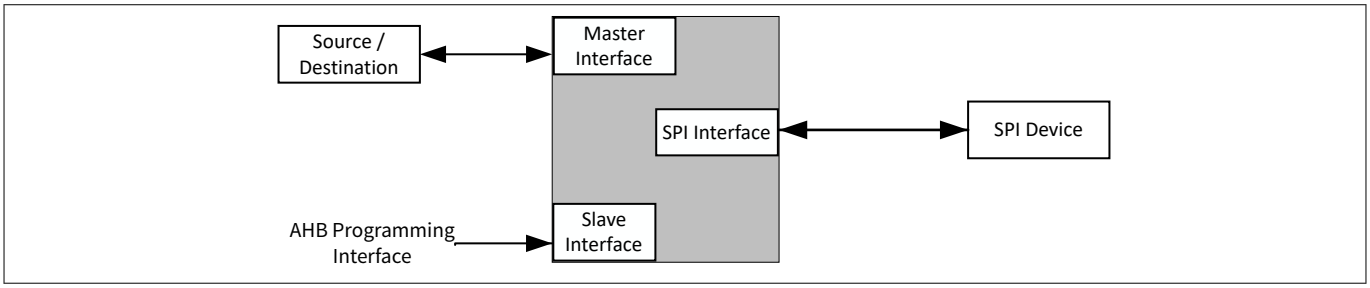


Figure 270 Block diagram for internal DMA feature for SPI operation

The programming sequence for DMA transfers can be found under the topic xSPI programming sequences. After xSPI has been programmed, it takes care of completing the transfer between the internal memory and external SPI device.

***Notes**:*

1. *The internal DMA operation is supported only in enhanced SPI (Dual/Quad/Octal and Hyperbus) mode of operation*
2. *When internal DMA is enabled, any read or write operation to Data Register (DR) results in error response from the AHB slave interface*
3. *Data Mask feature is not supported in internal DMA mode*
4. *The address length on SPI interface can be maximum of 32 bits in internal DMA mode*

Registers related to internal DMA feature

- CTRLR0
- CTRL1
- BAUDR
- SSIENR
- SER
- DMACR
- AXIAWLEN
- AXIARLEN
- SPI_CTRLR0
- SPIDR
- SPIAR
- AXIAR0

18.3.16.1 AXI master interface

xSPI uses an AXI interface to perform internal DMA operations.

The AXI interface can receive an error response from a write response or a read data channel. In such a situation, xSPI immediately aborts the transfer by de-asserting the slave-select signal and asserting the `ssi_axie_intr` signal. The data present in the internal FIFO is flushed and xSPI is disabled (`SSIC_EN = 0`). Also, xSPI updates the SR.CMPLTD_DF field with the amount of data that was successfully transferred in the last DMA

transfer. This information can be used to determine where exactly the error occurred, and the amount of data transferred successfully.

***Note**: The CMPLTD_DF of status register (SR) may not represent the correct value if the transfer is aborted by the user by programming SSIENR register to 0.*

18.3.17 Receive and transmit FIFO buffers

The transmit FIFO is loaded by writing to the xSPI data register (DRi (i=0-35)). Data is popped (removed) from the transmit FIFO by the shift control logic into the transmit shift register. The transmit FIFO generates a FIFO empty interrupt request (through the `ssi_txe_intr` signal) when the number of entries in the FIFO is less than or equal to the FIFO threshold value. The threshold value, set through the programmable register TXFTLR, determines the level of FIFO entries at which an interrupt is generated. The threshold value allows you to provide an early indication to the processor that the transmit FIFO is nearly empty. A transmit FIFO overflow interrupt request (through the `ssi_txo_intr` signal) is generated if you attempt to write data into an already full transmit FIFO.

Data is popped from the receive FIFO by reading the xSPI data register (DRi (i=0-35)). The receive FIFO is loaded from the receive shift register by the shift control logic. The receive FIFO generates a FIFO-full interrupt request (through the `ssi_rxf_intr` signal) when the number of entries in the FIFO is greater than or equal to the FIFO threshold value plus 1. The threshold value, set through programmable register RXFTLR, determines the level of FIFO entries at which an interrupt is generated. The threshold value provides an early indication to the processor that the receive FIFO is nearly full.

A receive FIFO overrun interrupt request (through the`ssi_rxo_intr` signal) is generated when the receive shift logic attempts to load data into a completely full receive FIFO. This newly received data is lost. A receive FIFO underflow interrupt request (through the `sssi_rxu_intr` signal) is generated if you attempt to read from an empty receive FIFO. This alerts the processor that the read data is invalid.

***Note**: The transmit and receive FIFO buffers are cleared when the xSPI is disabled (SSIENR.SSIC_EN = 0) or when it is reset .*

Registers related to receive and transmit FIFO buffers

- RXFTLR
- TXFTLR
- TXFLR
- RXFLR

18.3.18 Interrupts and alarms

This section describes the interrupts and alarms in the xSPI.

18.3.18.1 Interrupts

This section describes the interrupt grouping for xSPI.

xSPI has 2 interrupts that are routed on to the interrupt router:

- SRC_XSPIFIFO: xSPI FIFO interrupts are grouped into SRC_XSPIFIFO interrupt
- SRC_XSPIERROR: xSPI ERROR interrupts are routed to SRC_XSPIERROR interrupt

The table below shows the grouping of the corresponding interrupts.

Table 816 Interrupt multiplexing

Merged interrupt	Interrupt	Description
SRC_XSPIFIFO	ssi_txe_intr	Transmit FIFO empty active high interrupt. Active when the transmit FIFO is equal to or below the threshold value (TFT) and requires service to prevent an under-run. The threshold value, set through Transmit FIFO Threshold Level register, which determines the level of transmit FIFO entries at which an interrupt is generated. This interrupt is cleared by hardware when data is written into the transmit FIFO buffer, bringing it over the threshold level
	ssi_txo_intr	Transmit FIFO overflow active high interrupt. Set when there is an attempt to write into the transmit FIFO after it has been completely filled. When set, the data written is discarded
	ssi_txu_intr	Transmit FIFO underflow active high interrupt. Active when the transmit logic attempts to read from empty transmit FIFO
	ssi_rxf_intr	Receive FIFO full active high interrupt. Set when the receive FIFO is equal to or above its threshold value plus 1 and requires service to prevent an overflow. The threshold value, set through Receive FIFO Threshold Level register, which determines the level of receive FIFO entries at which an interrupt is generated. This interrupt is cleared by hardware when data is read from the receive FIFO buffer, bringing it below the threshold level
	ssi_rxo_intr	Receive FIFO overflow active high interrupt. Set when the receive logic attempts to place data into the receive FIFO after it has been completely filled. When set, the newly received data is discarded. This interrupt remains set until you read the receive FIFO overflow interrupt clear register (RXOICR)
	ssi_rxu_intr	Receive FIFO underflow active high interrupt. Set when an AHB access attempts to read from the receive FIFO when it is empty. When set, zeros are read back from the receive FIFO. This interrupt remains set until you read the receive FIFO underflow interrupt clear register (RXUICR)
	ssi_done_intr	Transfer done interrupt After xSPI transmits or receives the required amount of data to or from the slave respectively, the xSPI clears the SSIENR register, and interrupt ssi_done_intr is provided to software
SRC_XSPIERROR	ssi_axie_intr	AXI error interrupt

Registers related to SSI interrupts

- RXOICR
- RXUICR
- MSTICR
- ISR , RISR, ICR
- IMR

18.3.18.2 Alarms

This section describes the alarm grouping for xSPI.

xSPI has 2 Alarms that are routed to the SMU:

- XSIPDATAADD: xSPI data address error alarm
- XSPIFSM: xSPI FSM alarm

The table below shows the grouping of the corresponding alarm.

Table 817 Alarm multiplexing

Alarm name	Interrupt	Description
XSPIDATAADD	ssi_sfty_ap_intr	AXI interface parity interrupt.
	ssi_sfty_dpe_intr	SSI data path parity interrupt. This interrupt will be asserted when xSPI encounters a parity error on internal data path
	ssi_sfty_rs_intr	SSI data register space parity interrupt. This interrupt will be asserted when xSPI encounters an inconsistency in register space programming. For more details see Register space protection.
	ssi_sfty_addr_sp_intr	SSI AHB slave address interface parity interrupt. This signal will be asserted when xSPI encounters a address parity error on AHB slave interface
	ssi_sfty_data_sp_intr	SSI AHB slave data interface parity interrupt. This signal will be asserted when xSPI encounters a data parity error on AHB slave interface
	ssi_sfty_axi_ecc_ce_intr	AXI master interface ECC correctable error
	ssi_sfty_axi_ecc_ue_intr	AXI master interface ECC uncorrectable error
	sx_alarm_s2f_aph_err	SRI to FPI bridge address phase error
	sx_alarm_s2f_dph_err	SRI to FPI bridge data phase error
	sx_alarm_a2s_aph_err	AXI to SRI bridge address phase error
	sx_alarm_a2s_dph_err	AXI to SRI bridge data phase error
	ssi_sfty_axi_vrp_intr	AXI valid/ready parity error
	sx_alarm_fpi_mst	FPI master alarm. Error related to Universal Bus Slave (UBS) in FPI master from S2F bridge. This signal is triggered when there is inconsistencies of data phase signals (ready, ack and dp_inv).
	sx_alarm_fpi_slv	FPI slave alarm. Error related to Universal Bus Slave (UBS) in FPI master from S2F bridge. This signal is triggered when there is inconsistencies of data phase signals (ready, ack and dp_inv), inconsistencies in abort signal.

(table continues...)

Table 817 (continued) Alarm multiplexing

Alarm name	Interrupt	Description
	sx_alarm_fpi_wrerr	Write error on slave interface xSPI write errors, such as, access by masters who are not enabled for writing to xSPI address or SFR space, are flagged by this alarm
XSPIFSM	ssi_sfty_fp_intr	FSM parity interrupt
	ssi_sfty_fto_intr	FSM timeout interrupt

18.3.19 Hardware safety mechanisms

This section describes the safety mechanisms provided in the xSPI kernel.

18.3.19.1 Parity protection on AHB slave interface

The parity protection feature is implemented to provide high level data integrity checks for all data transfers at the AHB slave interface.

xSPI supports parity for both address and data signals.

Address parity check

The locally computed parity for the received address is compared with the parity received from the bridges. In case of an address parity check, a failure is indicated by the `ssi_sfty_addr_sp_intr` interrupt output. The actions taken by the xSPI core in case of an address parity protection failure are explained here for the different scenarios:

- Address Parity failure in register access (read and write)
  - The current register access is ignored. In case of a write transfer, the register is not updated. In case of read transfer, the data returned is all 0's
  - `ssi_sfty_addr_sp_intr` interrupt is generated
  - The subsequent register accesses proceed normally
  - `ssi_sfty_addr_sp_intr` interrupt is deasserted when the user clears it
- Address Parity failure in data register (DRi (i=0-35)) access
  - The interrupt is generated
  - The subsequent accesses to data registers are not allowed until the interrupt `ssi_sfty_addr_sp_intr` is cleared
- Address Parity failure in XIP transfers
  - `ssi_sfty_addr_sp_intr` interrupt is generated
  - If the interrupt `ssi_sfty_addr_sp_intr` is asserted due to address parity failure, the subsequent XIP transfers goes through even if the earlier interrupt is not cleared
  - Single transfers – transfer is not started on the SPI side
  - Burst transfers:
    - Address failure in the first beat – transfer is not started on the SPI side
    - If the first beat has no address parity failure and the second beat generates an address parity failure, the xSPI slave is deselected. The xSPI internally flushes all data

The SLVIFAPES field of the Safety Interrupt Status register (SAFETYISR) indicates the address parity failure status.

Write data parity check

The locally computed parity for the received write data is compared with the parity received. The actions taken by xSPI core in case of write data parity protection failure is explained here for the different scenarios.

- Register access
  - The current register write is ignored
  - The interrupt `ssi_sfty_data_sp_intr` is asserted to indicate the write data parity failure
  - Subsequent register accesses proceed normally
- Data Register access
  - The interrupt `ssi_sfty_data_sp_intr` is asserted to indicate the write data parity failure. Subsequent data register accesses are not allowed

The SLVIFDPES field of the Safety Interrupt Status register (SAFETYISR) indicates when write data parity failure status.

Read data parity checker

xSPI provides the parity per byte during read transfers along with the read data. The receiving bridge use this read parity to check the data integrity of the read data and flag (`sx_alarm_s2f_dph_err`) safety interrupt on parity mismatch.

Registers related to parity protection on AHB slave interface

- SAFETYCR
- SAFETYISR
- SAFETYICR

18.3.19.2 ECC and parity protection on master interface

The master interface is protected with ECC and Parity protection.

ECC protection on AXI master interface

The AXI master interface monitors and validates the ECC. If ECC monitor detects a single bit error, the received payload is corrected and a correctable error interrupt is generated. If a multi-bit error is detected, then an uncorrectable error interrupt is generated; however, the received payload cannot be corrected.

xSPI provides a total of 14 ECC Check bits for a payload width of 64 bits on the AXI interface.

The ECC Monitor generates the ECC Protection interrupts to indicate the correctable and uncorrectable error.

- AXI Channel ECC Correctable Error Interrupt: `ssi_sfty_axi_ce_intr`, if single bit error is detected, ECC Monitor generates correctable error interrupt. A single bit error is detected on the payload, and it is corrected by the ECC monitor
- AXI Channel ECC Uncorrectable Error Interrupt: `ssi_sfty_axi_ue_intr`, if multi-bit error is detected, ECC Monitor generates uncorrectable error interrupt. A multi bit error is detected on the payload, and it cannot be corrected by the ECC Monitor

Parity protection on AXI master interface

The AXI Master Interface Parity Protection implements the logic to generate and check the Parity for the AXI Channels that are being driven by the AXI Master. The xSPI supports even parity. xSPI receives parity per byte from AXI read data and AXI write response channels using side-band signals. The parity is used to protect all the control signals and data signals with separate parity bits for control fields and data fields. xSPI calculates an internal parity which is compared with the received parity bits. Along with this, a separate interrupt `ssi_sfty_ap_intr` is asserted when parity failure occurs. The AXI parity error is indicated by the AXIPES field of the Safety Interrupt Status (SAFETYISR) register.

Registers related to Parity protection on AXI master interface

- SAFETYCR
- SAFETYISR
- SAFETYICR

18.3.19.3 FSM protection

The following sections describe the two mechanisms available in xSPI to protect FSMs.

FSM parity protection

xSPI implements one-hot encoding scheme for holding the state register. Odd Parity is implemented on all the FSM state register bits. Parity is monitored on every clock, after the reset is de-asserted. When a bit flips due to transient errors or permanent faults, the erroneous FSM is set to its default state and an uncorrectable error is detected.

The FSM state parity protection is enabled by setting the FSM_PAR_EN field to 1, and the error insertion is enabled by setting the field FSM_EI_TYPE to 1 in the register SAFETYCR. The FSM in which error injection should happen is selected using SAFETYCR.FSM_SEL register bitfield. The FSM state error is indicated by (SAFETYISR).FSMPES register. Also, the safety interrupt `ssi_sfty_fp_intr` is asserted when an FSM error is detected. The FSM in which the error occurred can be determined by reading the Safety Interrupt register.

Error injection mode is also supported for the FSM parity error check for all the clock domains. This can be set using dedicated register bits SAFETYCR.FSM_SEL register.

FSM timeout protection

The Finite State Machine (FSM) time out feature provides the mechanism to ensure that all FSMs in xSPI can complete transaction, and return to a known completion state within a programmed time out value.

xSPI contains FSMs in slave, SSI and master clock domains. Each clock domain has one time out register. This can be programmed using the FSM time out registers for their respective clock domains. The timer ticks are generated based on the programmable time-out value. At every timer tick, all the FSMs are monitored for being in the active state (non-IDLE state, which indicates FSM is actively processing transactions or hand-shakes) and an internal FSM active flag field is set. The internal active flag bit is implemented for every FSM that is monitored for timeout. When the FSM reaches an IDLE or transaction completion state, the internal active flag field of the FSM is reset. At subsequent timer ticks, all FSM's active flag bits are monitored and any flag that is set indicates a time out for that FSM. This process of setting the flag, and checking at subsequent timer tick is repeated at every tick.

The FSM Time out error injection can be enabled by setting the FSM_EI_EN field to 1, and the FSM_EI_TYPE field to 0 in the SAFETYCR register. The FSM in which error injection should happen is selected using the FSM_SEL programming field of the SAFETYCR register. When time out error injection is enabled for an FSM, the FSM automatically times out and generates interrupt even without traffic.

The FSM time out state error is indicated by AXIFSMTOS/SSIFSMTOS/AHBFSMTOS fields of the xSPI Safety Interrupt Status (SAFETYISR) register for master, SSI and slave clock domains respectively. Also, the safety interrupt `ssi_sfty_fto_intr` is asserted when an FSM error is detected in any of AXI, SSI or AHB clock domains.

FSM timeouts provide a mechanism to ensure all the FSMs complete transactions, and returns to a known completion state. If an FSM is hung in a state for more than a programmed amount of time, then it is indicated to the application layer through the safety interrupt `ssi_sfty_fto_intr`.

Registers related to FSM protection

- SAFETYCR
- SAFETYISR
- SAFETYICR
- SLVIFFSMTOCR

- SSIFSMTOCR
- AXIFSMTOCR

18.3.19.4 On-chip data protection with parity

The On-Chip Data Protection with Parity feature provides parity protection for the transmit and receive data paths, wherever feasible. For every byte of data, one parity bit is generated.

At the transmit side, the data from master/slave interface is protected until it reaches the shift control logic. At the receive side, the data from serial interface is protected from the serial shift register until it reaches the slave interface or master interface.

***Note**: The master interface is used in the internal DMA mode of operation, otherwise the slave interface is used for data transmission or reception. Only one of them can be active at the same time.*

Transmit data path with parity protection

For data transmission the data can come from master or the slave interface based on internal programming. Figure below describes the locations where parity generators and parity checkers are placed in the transmit data path.

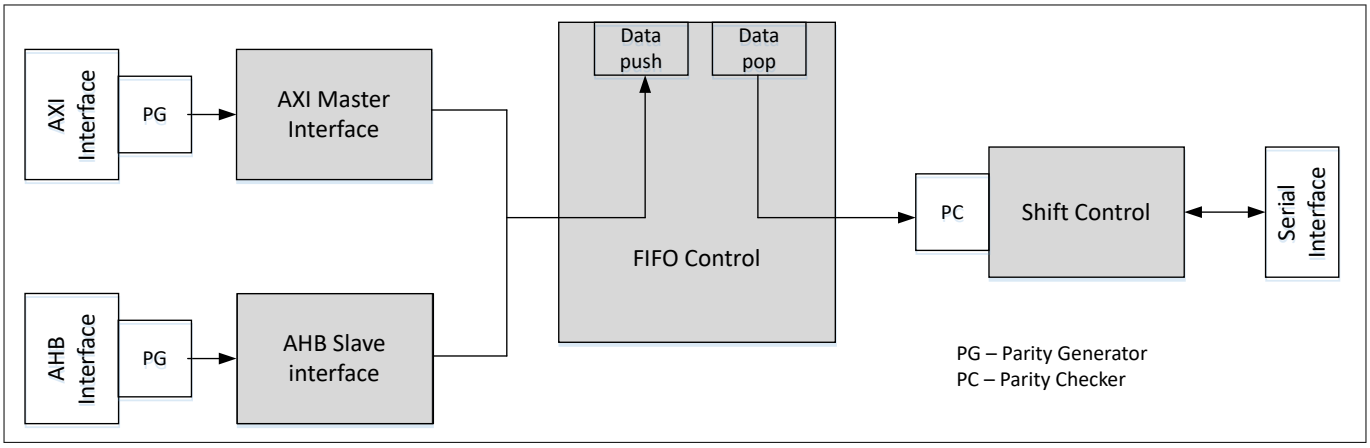


Figure 271 Data path parity protection for transmit data path

Following are the places where parity generators and checkers are present in the transmit data path:

- Parity is generated when master or slave interface has data ready to shift to the FIFO
- Parity is checked when the data is loaded into the internal shift register

Receive data path with parity protection

Figure below describes the locations where parity generators and parity checkers are placed in the receive data path.

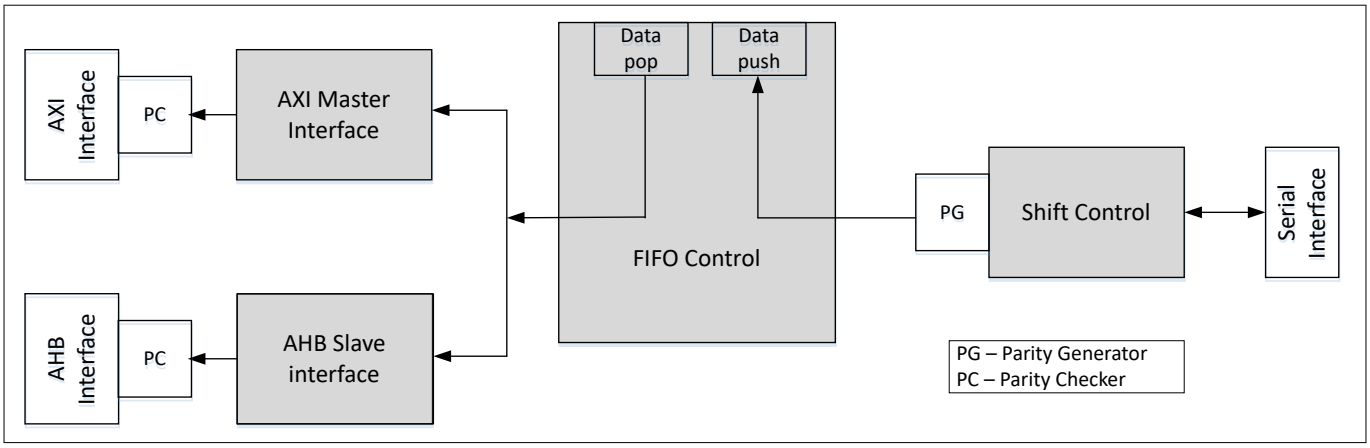


Figure 272 Data path parity protection for receive data path

Following are the places where parity generators and checkers are available in the receive data path:

- Parity is generated when the serial interface has data ready to shift to the FIFO
- Parity is checked when the data is about to be sent into the master or slave interface

Error injection in data path with parity protection

Each Parity generator supports control to inject parity errors. When enabled, the parity of the incoming data is flipped until the parity error is detected by the parity checker. The user must set the DPPG_EI_EN field of the SSI Safety Control (SAFETYCR) register. After the data path parity interrupt is detected, the hardware clears the DPPG_EI_EN. There are four possible parity error injection methods for each path. The following describes how to inject parity errors in each path:

For DMA mode:

- If the transfer mode is set to Write, an Error is injected at master interface parity generator and the error is detected at parity checker present before shift control logic
- If the transfer mode is set to Read, an Error is injected at parity generator present after shift control logic and the error is detected at the parity checker present in the master interface

For normal operation mode:

- If the transfer mode is set to Transmit Only, an Error is injected at slave interface parity generator and the error is detetced at parity checker present before shift control logic
- If the transfer mode is set to Receive Only, an Error is injected at parity generator present after shift control logic and the error is detected at the parity checker present at slave interface

After the generator inserts an error in the generated parity data, the subsequent parity checker should detect the mismatch and report the error status in SSI Safety Interrupt status (SAFETYISR) register.

For example, if xSPI needs to insert an error in the input data at parity generator, the sequence of operation is as follows:

1. Application sets the DPPG_EI_EN field of the SSI Safety Control (SAFETYCR) register
2. Application enables xSPI by setting SSIENR register to 1
3. The parity generator inserts an error in the parity generated for the data
4. As the data traverses through multiple paths, the subsequent checkers detect the parity mismatch and report the error status in the SSI Safety Interrupt (SAFETYISR) register and generate the `ssi_sfty_dpe_intr` interrupt
5. After the interrupt is detected, xSPI clears the DPPG_EI_EN bit of the SAFETYCR register

***Notes**:*

1. *For Error insertion the transfer mode field (CTRLR0.TMOD) should not be set to 00_B (transmit and receive)*
2. *Error insertion is not applicable for XIP transfers*

Registers related to on-chip data protection with parity

- SAFETYCR
- SAFETYISR
- SAFETYICR

18.3.19.5 Register space protection

The Register Space Protection feature is based on a simple form of hardware redundancy, where a set of configuration registers are duplicated. The original set drives the logic, whereas the copy is used for consistency and it is not connected to any logic as shown in the figure below.

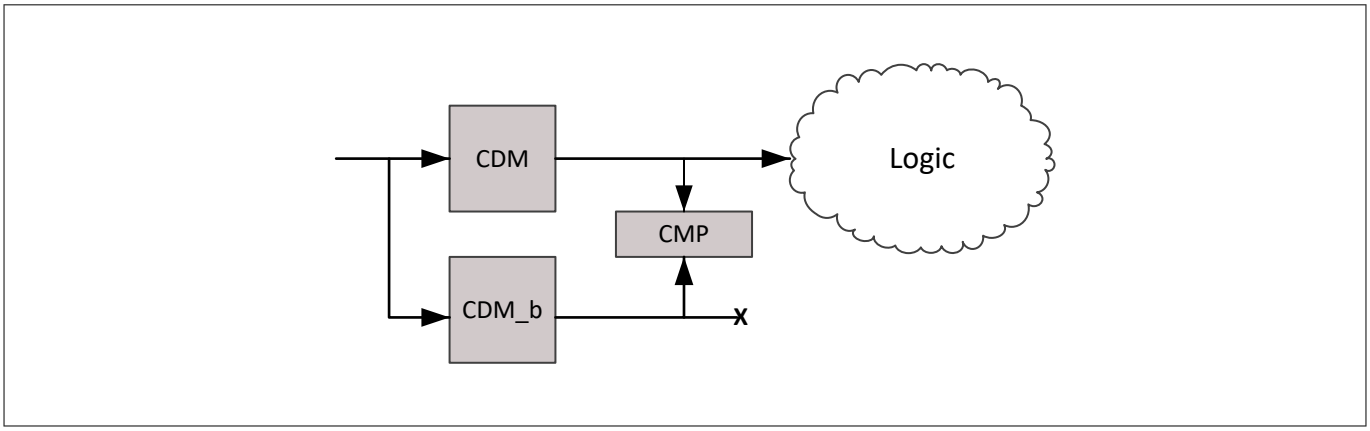


Figure 273 Register space protection architecture

Both register sets are written with the same values when any one of the registers is programmed. A state machine runs periodically traversing the register space, and comparing the values from both register sets. If a difference is detected, it is signaled to the application layer using the dedicated interrupt `ssi_sfty_rs_intr`.

Error injection capability is available for the register duplication logic. This can be enabled with the bit REGP_EI_EN of the SSI Safety Control (SAFETYCR) register. Errors can be injected in each of the registers, which are protected through this logic. The SAFETYCR.REG_SEL field is used to select the register in which error should be injected. Table below lists all the registers that are protected using this feature along with the bitfield setting for SAFETYCR.REG_SEL.

Table 818 Register protected with duplication and field decode values

Register code	Register name
0	CTRLR0: Control 0 Register
1	CTRLR1: Control 1 Register
2	SER: Slave Enable Register
3	BAUDR: Baud Rate Register
4	DMACR: DMA Control Register
5	RX_SAMPLE_DELAY: RX Sample Delay Register
6	SPI_CTRLR0: SPI Control 0 register
9	SAFETYCR: Safety Control Register

Registers related to Register space protection

- SAFETYCR

- SAFETYISR
- SAFETYICR

18.3.20 Memory and SFR protection

The access protection unit (APU) provides granular control of the protection settings for a group of registers or memory regions. The APU consists of a set of configurable registers and access checking logic which either allows or disallows each access from the bus interface to the intended target. Disallowed accesses are terminated with an alarm or bus error.

Memory protection

In xSPI the memory mapped region is protected by a maximum of eight APUs, which are represented by set of eight configuration registers RGNx_ACCEN where x= 0 to 7. The region for each APUs is configured by writing the lower and upper addresses of the desired region into the RGNx_ACCEN_RGNLA and RGNx_ACCEN_RGNUA registers respectively. The APU allows configuration software to enable or disable access to each master agent function in the MCU through RGNx_ACCEN_WRA or RGNx_ACCEN_WRB and RGNx_ACCEN_RDA or RGNx_ACCEN_RDB registers. Each master agent function can be configured to have read-only access, write-only access, read and write access or can be disallowed access altogether.

The reset state of the APU is set to enable all master agent functions for read access, and enable only CPU0 and Cerberus for write access. The user has to configure the desired master access for XIP operation after every application reset. If overlapping regions are defined, then an access only needs to be permitted by one of the overlapping regions for it to succeed. Figure below shows APU protection scheme for memory mapped regions.

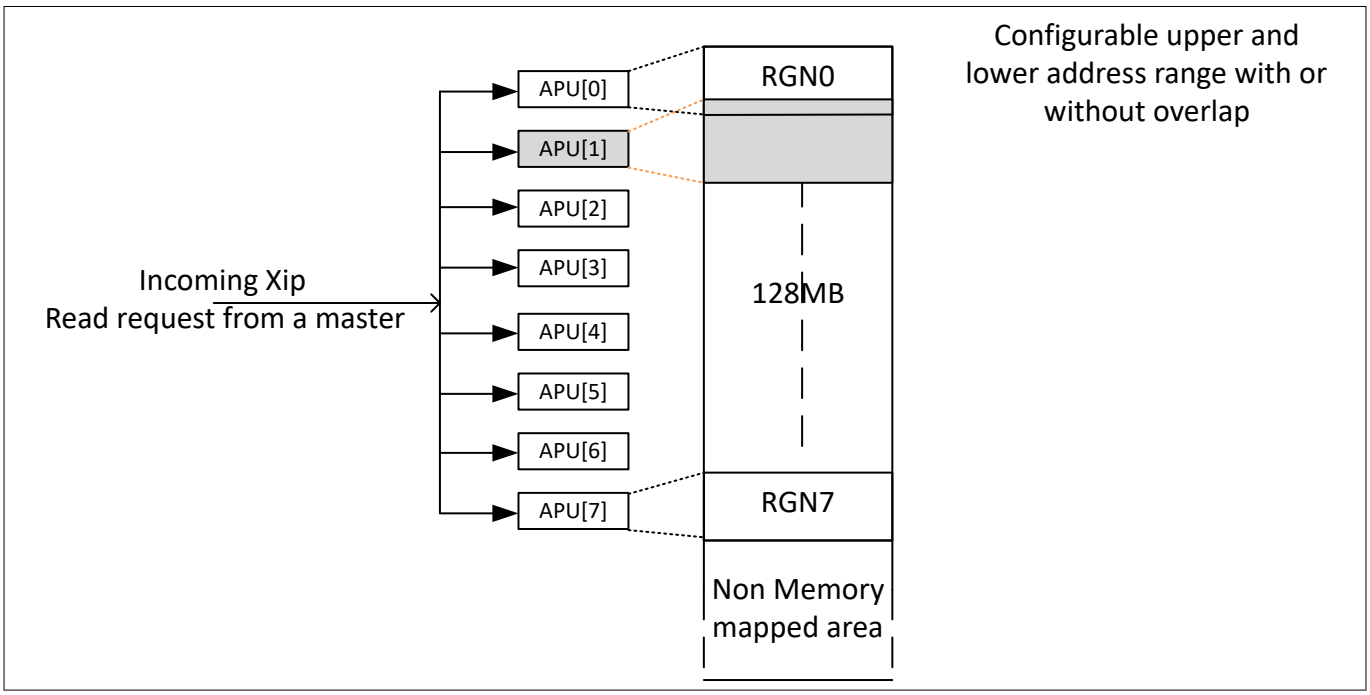


Figure 274 APU for memory mapped region

Register protection

The xSPI Special Function Registers (SFRs) are protected by one APU configuration register set ACCEN. The reset state of the APU is set to enable all master agent functions for read access, and enable only CPU0 and Cerberus for write access.

Related information

APU registers on page 78

18 eXpanded Serial Peripheral Interface (xSPI)

APU arrays to protect system memories on page 77
Access protection on page 63

18.4 Registers

18.4.1 Register overview - access mode glossary

Table 819 Register overview - access mode glossary

Keyword	Description
E	Access protection using PROT register PROTE.
SE	Access protection using PROT register PROTSE.
APU-P	Protection group consisting of registers ACCEN_WRA, ACCEN_WRB, ACCEN_RDA, ACCEN_RDB, ACCEN_VM, ACCEN_PRS.
P	Access protection using *APU-P* registers.
APU-PRGNx (x=0-7)	Protection group consisting of registers RGNx_ACCEN_WRA, RGNx_ACCEN_WRB, RGNx_ACCEN_RDA, RGNx_ACCEN_RDB, RGNx_ACCEN_VM, RGNx_ACCEN_PRS, RGNx_ACCEN_RGNLA, RGNx_ACCEN_RGNUA.
PRGNx	Access protection using *APU-PRGNx* registers.
BE	Always returns a Bus Error.
U	No access restrictions.
PROT	Access restrictions as defined in the PROT register access rules.
nBE	Indicates that no Bus Error is generated when accessing this address range, even though it is either an access to an undefined address or the access does not follow the given rules.

18.4.2 Register overview - XSPI domain SRI (ascending offset address)

Table 820 Register overview - XSPI domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CLC	Clock Control Register	00000000$_H$	P	P, SV, E	Application Reset	2639
OCS	OCDS Control and Status Register	00000004$_H$	P	SV, P	Debug Reset	2639
ID	Module Identification Register	00000008$_H$	P	BE	PowerOn Reset	2640
RST_CTRLA	Reset Control Register A	0000000C$_H$	P	P, SV, E	Application Reset	2640
RST_CTRLB	Reset Control Register B	00000010$_H$	P	P, SV, E	Application Reset	2641
RST_STAT	Reset Status Register	00000014$_H$	P	BE	Application Reset	2642
PROTE	PROT Register Endinit	00000020$_H$	U	SV, PROT	Application Reset	2642
PROTSE	PROT Register Safe Endinit	00000024$_H$	U	SV, PROT	Application Reset	2644

(table continues...)

Table 820 (continued) Register overview - XSPI domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ACCEN_WRA	Write access enable register A	00000040$_H$	U	SE, SV	Application Reset	2645
ACCEN_WRB	Write access enable register B	00000044$_H$	U	SE, SV	Application Reset	2646
ACCEN_RDA	Read access enable register A	00000048$_H$	U	SE, SV	Application Reset	2646
ACCEN_RDB	Read access enable register B	0000004C$_H$	U	SE, SV	Application Reset	2647
ACCEN_VM	VM access enable register	00000050$_H$	U	SE, SV	Application Reset	2647
ACCEN_PRS	PRS access enable register	00000054$_H$	U	SE, SV	Application Reset	2648
RGNx_ACCEN_WRA	CHx Write access enable register A	00000060$_H$+x*20$_H$	U	SE, SV	Application Reset	2649
RGNx_ACCEN_WRB	CHx Write access enable register B	00000064$_H$+x*20$_H$	U	SE, SV	Application Reset	2649
RGNx_ACCEN_RDA	CHx Read access enable register A	00000068$_H$+x*20$_H$	U	SE, SV	Application Reset	2650
RGNx_ACCEN_RDB	CHx Read access enable register B	0000006C$_H$+x*20$_H$	U	SE, SV	Application Reset	2650
RGNx_ACCEN_VM	CHx VM access enable register	00000070$_H$+x*20$_H$	U	SE, SV	Application Reset	2651
RGNx_ACCEN_PRS	CHx PRS access enable register	00000074$_H$+x*20$_H$	U	SE, SV	Application Reset	2651
RGNx_ACCEN_RGNLA	CHx RGN Lower address	00000078$_H$+x*20$_H$	U	SE, SV	Application Reset	2652
RGNx_ACCEN_RGNUA	CHx RGN upper address	0000007C$_H$+x*20$_H$	U	SE, SV	Application Reset	2653
INTSTAT	Error handling control	00000160$_H$	P	P	Kernel Reset	2653
INTCLR	Error handling interrupt/ alarm clear	00000164$_H$	P	P	Kernel Reset	2654
INTMSK	Error handling interrupt/ alarm mask	00000168$_H$	P	P	Kernel Reset	2655

(table continues...)

Table 820 (continued) Register overview - XSPI domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CTRLR2	Control Register 2	0000016 C_H	P	P	Kernel Reset	2657
CTRLR0	Control Register 0	0000020 0_H	P	P	Kernel Reset	2658
CTRLR1	Control Register 1	0000020 4_H	P	P	Kernel Reset	2661
SSIENR	xSPI kernel Enable Register	0000020 8_H	P	P	Kernel Reset	2662
SER	Slave Enable Register	0000021 0_H	P	P	Kernel Reset	2663
BAUDR	Baud Rate Select	0000021 4_H	P	P	Kernel Reset	2663
TXFTLR	Transmit FIFO Threshold Level	0000021 8_H	P	P	Kernel Reset	2664
RXFTLR	Receive FIFO Threshold Level	0000021 C_H	P	P	Kernel Reset	2665
TXFLR	Transmit FIFO Level Register	0000022 0_H	P	P	Kernel Reset	2665
RXFLR	Receive FIFO Level Register	0000022 4_H	P	P	Kernel Reset	2666
SR	Status Register	0000022 8_H	P	P	Kernel Reset	2666
IMR	Interrupt Mask Register	0000022 C_H	P	P	Kernel Reset	2668
ISR	Interrupt Status Register	0000023 0_H	P	P	Kernel Reset	2669
RISR	Raw Interrupt Status Register	0000023 4_H	P	P	Kernel Reset	2670
TXEICR	Transmit FIFO Error Interrupt Clear Register Register	0000023 8_H	P	P	Kernel Reset	2672
RXOICR	Receive FIFO Overflow Interrupt Clear Register	0000023 C_H	P	P	Kernel Reset	2672
RXUICR	Receive FIFO Underflow Interrupt Clear Register	0000024 0_H	P	P	Kernel Reset	2672
MSTICR	Multi-Master Interrupt Clear Register	0000024 4_H	P	P	Kernel Reset	2673
ICR	Interrupt Clear Register	0000024 8_H	P	P	Kernel Reset	2673
DMACR	DMA Control Register	0000024 C_H	P	P	Kernel Reset	2674

(table continues...)

Table 820 (continued) Register overview - XSPI domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
AXIAWLEN	Destination Burst Length	00000250$_H$	P	P	Kernel Reset	2675
AXIARLEN	Source Burst Length	00000254$_H$	P	P	Kernel Reset	2675
SSIC_VERSION_ID	xSPI kernel component version	0000025C$_H$	P	P	Kernel Reset	2676
DRi	xSPI Data Register i	00000260$_H$+i*4	P	P	Kernel Reset	2676
RX_SAMPLE_DELAY	RX Sample Delay Register	000002F0$_H$	P	P	Kernel Reset	2677
SPI_CTRLR0	SPI_CTRLR0 - SPI Control Register	000002F4$_H$	P	P	Kernel Reset	2677
DDR_DRIVE_EDGE	DDR_DRIVE_EDGE - Transmit Drive Edge Register	000002F8$_H$	P	P	Kernel Reset	2680
XIP_MODE_BITS	eXecute in Place - Mode bits	000002FC$_H$	P	P	Kernel Reset	2681
XIP_INCR_INST	XIP_INCR_INST - XIP INCR transfer opcode	00000300$_H$	P	P	Kernel Reset	2681
XIP_WRAP_INST	XIP WRAP transfer opcode	00000304$_H$	P	P	Kernel Reset	2682
XIP_CNT_TIME_OUT	XIP time out register for continuous transfers	00000314$_H$	P	P	Kernel Reset	2682
SPIDR	SPI Device Register	00000320$_H$	P	P	Kernel Reset	2683
SPIAR	SPI Device Address Register	00000324$_H$	P	P	Kernel Reset	2683
AXIAR0	AXI Address Register 0	00000328$_H$	P	P	Kernel Reset	2683
AXIECR	AXI Master Error Interrupt Clear Register	00000330$_H$	P	P	Kernel Reset	2684
DONECR	Transfer Done Clear Interrupt Clear Register	00000334$_H$	P	P	Kernel Reset	2684
SAFETYCR	Safety Control register	00000380$_H$	P	P	Kernel Reset	2685
SLVIFFSMTOCR	Slave interface Clock domain FSM time out register	00000384$_H$	P	P	Kernel Reset	2687
SSIFSMTOCR	xSPI kernel Clock domain FSM time out register	00000388$_H$	P	P	Kernel Reset	2687
AXIFSMTOCR	AXI Clock domain FSM time out register	0000038C$_H$	P	P	Kernel Reset	2688

(table continues...)

Table 820 (continued) Register overview - XSPI domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
SAFETYISR	Safety Interrupt Status Register	000003A 0_H	P	P	Kernel Reset	2688
SAFETYICR	Safety Interrupt Clear Register	000003A 4_H	P	P	Kernel Reset	2689

18.4.3 Clock Control Register

The Clock Control Register CLC allows the programmer to adapt the functionality and power consumption of the module to the requirements of the application.

Register CLC controls the module clock signal and the reactivity to the sleep signal.

CLC Offset address: 00000000_H

Clock Control Register Application Reset value: $0000\ 0003_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

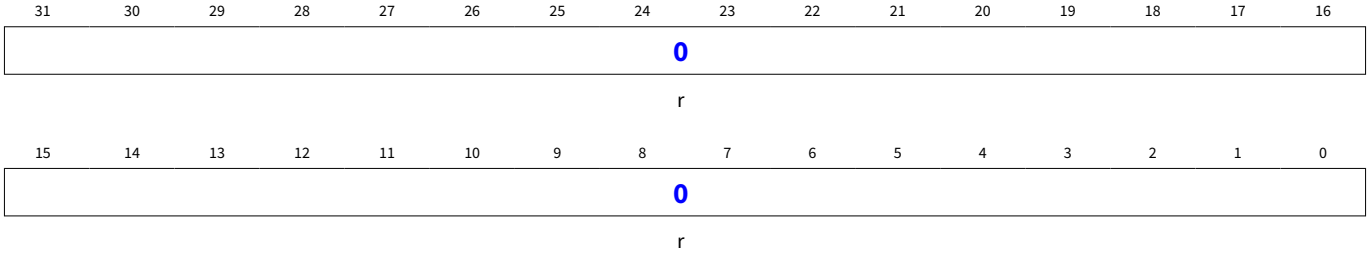
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												EDIS	0	DISS	DISR
r												rw	r	rh	rw

Field	Bits	Type	Description
DISR	0	rw	**Module Disable Request Bit** Used for enable/disable control of the module. 0_B On request: enable the module clock 1_B Off request: stop the module clock
DISS	1	rh	**Module Disable Status Bit** 0_B Module clock is enabled 1_B Off: module is not clocked
EDIS	3	rw	**Sleep Mode Enable Control** Used to control the module's reaction to sleep mode. 0_B Sleep mode request is enabled and functional 1_B Module disregards the sleep mode control signal
0	2, 31:4	r	**Reserved** Read as 0; should be written with 0.

18.4.4 OCDS Control and Status Register

OCS Offset address: 00000004_H

OCDS Control and Status Register Debug Reset value: $0000\ 0000_H$

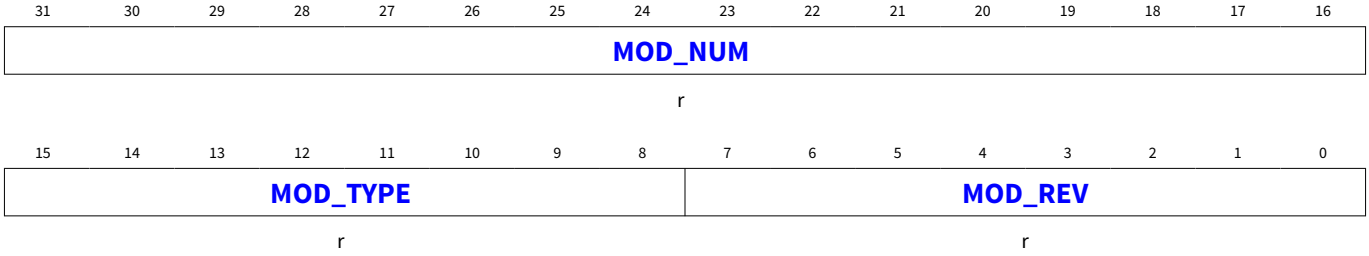


Field	Bits	Type	Description
0	31:0	r	**Reserved** Read as 0; should be written with 0.

18.4.5 Module Identification Register

ID Offset address: 00000008_H

Module Identification Register PowerOn Reset value: $0101\ C002_H$

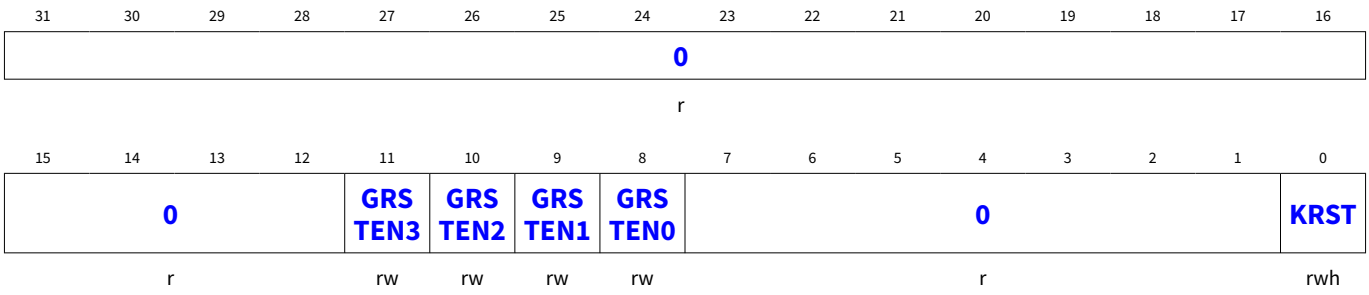


Field	Bits	Type	Description
MOD_REV	7:0	r	**Module Revision** Indicates the revision number of the implementation.
MOD_TYPE	15:8	r	**Module Type** This internal marker is fixed to $C0_H$.
MOD_NUM	31:16	r	**Module Number** Indicates the module identification number

18.4.6 Reset Control Register A

RST_CTRLA Offset address: $0000000C_H$

Reset Control Register A Application Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
KRST	0	rwh	**Kernel Reset** Request a kernel reset. The requested reset is executed if CTRLA.KRST=1 and CTRLB.KRST=1. KRST is cleared after the kernel reset was executed. 0_B No action 1_B A kernel reset was requested
GRSTENx (x=0-3)	x+8	rw	**Enable for Global Module Reset Group x** 0_B Global module reset group x does not have any effect 1_B Global module reset group x results in a kernel reset
0	7:1, 31:12	r	**Reserved** Read as 0; should be written with 0.

18.4.7 Reset Control Register B

RST_CTRLB Offset address: 00000010_H

Reset Control Register B Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
STATCLR								0							
w								r							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															KRST
r															rwh

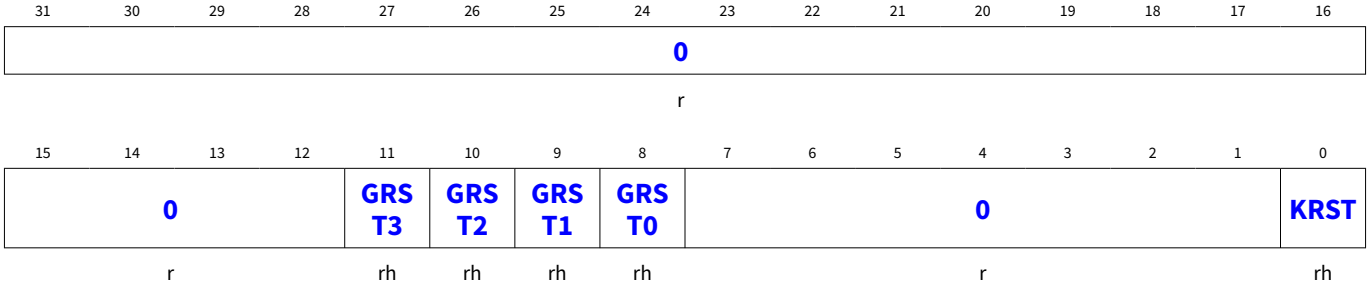
Field	Bits	Type	Description
KRST	0	rwh	**Kernel Reset** Request a kernel reset. The requested reset is executed if CTRLA.KRST=1 and CTRLB.KRST=1. KRST is cleared after the kernel reset was executed. 0_B No action 1_B A kernel reset was requested
STATCLR	31	w	**Kernel Reset Status Clear** Clears all status bits in RST_STAT when 1 is written. Read will return 0. 0_B No action 1_B Write with ´1´ clears bits STAT.GRSTx and bit STAT.KRST .
0	30:1	r	**Reserved** Read as 0; should be written with 0.

18.4.8 Reset Status Register

The reset status register contains the status bits for kernel reset (KRST) and the global module reset groups (GRSTx).

RST_STAT Offset address: 00000014_H

Reset Status Register Application Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
KRST	0	rh	**Kernel reset status** Indicates an executed kernel reset. RST_STAT.KRST is set after the execution of a kernel reset in the same clock cycle in which the reset bits are cleared. Clear KRST by setting bit STATCLR in register RST_CTRLB.
GRSTx (x=0-3)	x+8	rh	**Status for global module reset group x** 0_B Reset was not triggered by global reset group x 1_B Reset was triggered by global reset group x
0	7:1, 31:12	r	**Reserved** Read as 0.

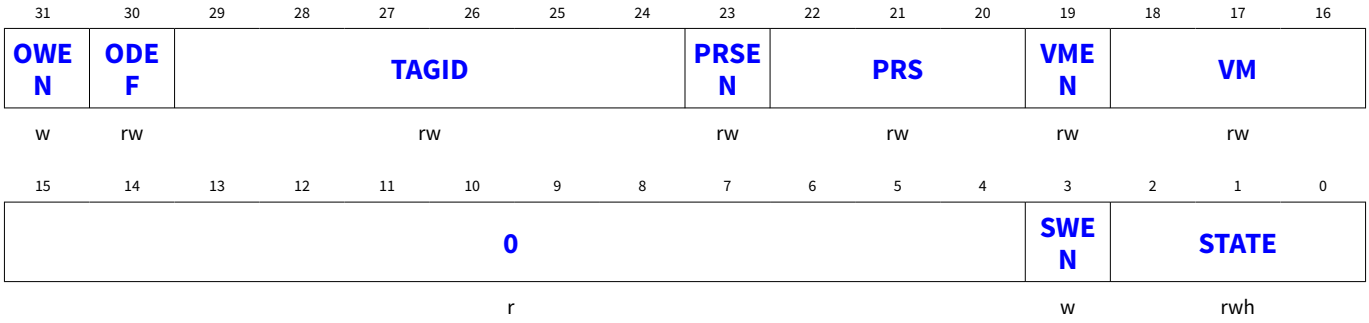
18.4.9 PROT Register Endinit

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner or CS master to update the PROT state as described in the PROT mechanism.

The PROTE register controls lock / unlock of the local Endinit (E) protected control registers.

PROTE Offset address: 00000020_H

PROT Register Endinit Application Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
VM	18:16	rw	**Virtual machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual machine definition enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.
PRSEN	23	rw	**Protection set definition enable for PROT owner** 0_B **Disable**: PRS is not part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG-ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; should be written with 0.

Table 821 Access mode restrictions of PROTE sorted by descending priority

Mode name	Access mode		Description
write 1 to .**SWEN**	rwh	STATE	
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

18.4.10 PROT Register Safe Endinit

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner or CS master to update the PROT state as described in the PROT mechanism.

The PROTSE register controls lock / unlock of the local Safe Endinit (SE) protected control registers.

PROTSE Offset address: 00000024_H

PROT Register Safe Endinit Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OWEN	**ODEF**	**TAGID**						**PRSEN**	**PRS**			**VMEN**	**VM**		
w	rw	rw						rw	rw			rw	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												**SWEN**	**STATE**		
r												w	rwh		

Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write

(table continues...)

(continued)

Field	Bits	Type	Description
VM	18:16	rw	**Virtual machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual machine definition enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.
PRSEN	23	rw	**Protection set definition enable for PROT owner** 0_B **Disable**: PRS is not part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG-ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; should be written with 0.

Table 822 Access mode restrictions of PROTSE sorted by descending priority

Mode name	Access mode		Description
write 1 to .**SWEN**	rwh	STATE	
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

18.4.11 Write access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

ACCEN_WRA Offset address: 00000040_H

Write access enable register A Application Reset value: $1000\ 0003_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access

18.4.12 Write access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

ACCEN_WRB Offset address: 00000044_H

Write access enable register B Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN63	EN62	EN61	EN60	EN59	EN58	EN57	EN56	EN55	EN54	EN53	EN52	EN51	EN50	EN49	EN48
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN47	EN46	EN45	EN44	EN43	EN42	EN41	EN40	EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-63)	q-32	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access

18.4.13 Read access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

ACCEN_RDA Offset address: 00000048_H

Read access enable register A Application Reset value: $FFFF\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access

18.4.14 Read access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

ACCEN_RDB Offset address: $0000004C_H$

Read access enable register B Application Reset value: $0000\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN63	EN62	EN61	EN60	EN59	EN58	EN57	EN56	EN55	EN54	EN53	EN52	EN51	EN50	EN49	EN48
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN47	EN46	EN45	EN44	EN43	EN42	EN41	EN40	EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-63)	q-32	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access

18.4.15 VM access enable register

This register defines which virtual machine encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each VMx.

ACCEN_VM Offset address: 00000050_H

VM access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for VM ID q** This bit enables read access to the access protected region for transactions with the VM ID q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for VM ID q** This bit enables write access to the access protected region for transactions with the VM ID q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

18.4.16 PRS access enable register

This register defines which protection register set encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each PRSx.

ACCEN_PRS Offset address: 00000054_H

PRS access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for PRS q** This bit enables read access to the access protected region for transactions with the PRS q. 0_B Disabled for read access 1_B Enabled for read access

(table continues...)

(continued)

Field	Bits	Type	Description
WRq (q=00-07)	q+16	rw	**Write access enable for PRS q** This bit enables write access to the access protected region for transactions with the PRS q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

18.4.17 CHx Write access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

RGNx_ACCEN_WRA (x=0-7) Offset address: 00000060_H+x*20_H

CHx Write access enable register A Application Reset value: $1000\ 0003_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access

18.4.18 CHx Write access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

RGNx_ACCEN_WRB (x=0-7) Offset address: 00000064_H+x*20_H

CHx Write access enable register B Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN63	EN62	EN61	EN60	EN59	EN58	EN57	EN56	EN55	EN54	EN53	EN52	EN51	EN50	EN49	EN48
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN47	EN46	EN45	EN44	EN43	EN42	EN41	EN40	EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-63)	q-32	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access

18.4.19 CHx Read access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

RGNx_ACCEN_RDA (x=0-7) Offset address: 00000068_H+x*20_H

CHx Read access enable register A Application Reset value: FFFF $FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access

18.4.20 CHx Read access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

RGNx_ACCEN_RDB (x=0-7) Offset address: $0000006C_H+x*20_H$

CHx Read access enable register B Application Reset value: 0000 $00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN63	EN62	EN61	EN60	EN59	EN58	EN57	EN56	EN55	EN54	EN53	EN52	EN51	EN50	EN49	EN48
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN47	EN46	EN45	EN44	EN43	EN42	EN41	EN40	EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-63)	q-32	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access

18.4.21 CHx VM access enable register

This register defines which virtual machine encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each VMx.

RGNx_ACCEN_VM (x=0-7) Offset address: 00000070_H+x*20_H

CHx VM access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for VM ID q** This bit enables read access to the access protected region for transactions with the VM ID q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for VM ID q** This bit enables write access to the access protected region for transactions with the VM ID q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

18.4.22 CHx PRS access enable register

This register defines which protection register set encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each PRSx.

RGNx_ACCEN_PRS (x=0-7) Offset address: 00000074_H+x*20_H

CHx PRS access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for PRS q** This bit enables read access to the access protected region for transactions with the PRS q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for PRS q** This bit enables write access to the access protected region for transactions with the PRS q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

18.4.23 CHx RGN Lower address

This register sets the lower bound of the access protected region. Addresses greater than or equal to the lower bound are considered part of the access protected region.

Lower Address of the memory mapped region corresponding to the respective APU array

RGNx_ACCEN_RGNLA (x=0-7) Offset address: 00000078_H+x*20_H

CHx RGN Lower address Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADDR															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR										0					
rw										r					

Field	Bits	Type	Description
ADDR	31:6	rw	**Bits 31 to 6 of the lower bound of the access protected region** Defines bits 31:6 of the lower bound of the access protected region.
0	5:0	r	**Reserved** Read as 0; should be written with 0.

18.4.24 CHx RGN upper address

This register sets the upper bound of the access protected region. Addresses stricly less than the upper bound are considered part of the access protected region.

Upper Address of the memory mapped region corresponding to the respective APU array

RGNx_ACCEN_RGNUA (x=0-7) Offset address: $0000007C_H+x*20_H$

CHx RGN upper address Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADDR															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR										0					
rw										r					

Field	Bits	Type	Description
ADDR	31:6	rw	**Bits 31 to 6 of the upper bound of the access protected region** Defines bits 31:6 of the upper bound of the access protected region.
0	5:0	r	**Reserved** Read as 0; should be written with 0.

18.4.25 Error handling control

INTSTAT Offset address: 00000160_H

Error handling control Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0									WRERR	FPISLV	FPIMST	S2FADDR	S2FDAT	A2SADDR	A2SDAT
r									rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
A2SDAT	0	rh	**Error detected during the data phase in the A2S bridge** 0_B **INACTIVE**: A2SDAT is not active prior masking 1_B **ACTIVE**: A2SDAT is active prior masking
A2SADDR	1	rh	**Error detected during the address phase in the A2S bridge** 0_B **INACTIVE**: A2SADDR is not active prior masking 1_B **ACTIVE**: A2SADDR is active prior masking

(table continues...)

(continued)

Field	Bits	Type	Description
S2FDAT	2	rh	**Error detected during the data phase in the S2F bridge.** 0_B **INACTIVE**: S2FDAT is not active prior masking 1_B **ACTIVE**: S2FDAT is active prior masking
S2FADDR	3	rh	**Error detected during the address phase in the S2F bridge.** 0_B **INACTIVE**: S2FADDR is not active prior masking 1_B **ACTIVE**: S2FADDR is active prior masking
FPIMST	4	rh	**Error related to UBS issues in FPI master from S2F bridge** Address and read data EDCs, inconsistancies of data phase signals (ready, ack and dp_inv) 0_B **INACTIVE**: FPIMST is not active prior masking 1_B **ACTIVE**: FPIMST is active prior masking
FPISLV	5	rh	**Error signal generated by UBS** due to: Address and write data EDCs , inconsistancies of data phase signals (ready, ack and dp_inv), inconsistencies in abort signal 0_B **INACTIVE**: FPISLV is not active prior masking 1_B **ACTIVE**: FPISLV is active prior masking
WRERR	6	rh	**Write error on slave interface** xSPI write errors, such as, access by masters who are not enabled for writing to xSPI address or SFR space, are flagged by this bit 0_B **INACTIVE**: WRERR is not active prior masking 1_B **ACTIVE**: WRERR is active prior masking
0	31:7	r	**Reserved** Read as 0; should be written with 0.

18.4.26 Error handling interrupt/alarm clear

INTCLR Offset address: 00000164_H

Error handling interrupt/alarm clear Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0									WRERRCLR	FPISCLR	FPIMCLR	S2FACLR	S2FDCLR	A2SACLR	A2SDCLR
r									w	w	w	w	w	w	w

Field	Bits	Type	Description
A2SDCLR	0	w	**Clear the A2SDAT bit.** 0_B Keep the status 1_B Clear the status
A2SACLR	1	w	**Clear the A2SADDR bit.** 0_B Keep the status 1_B Clear the status
S2FDCLR	2	w	**Clear the S2FDAT bit.** 0_B Keep the status 1_B Clear the status
S2FACLR	3	w	**Clear the S2FADDR bit.** 0_B Keep the status 1_B Clear the status
FPIMCLR	4	w	**Clear the FPIMST bit.** 0_B Keep the status 1_B Clear the status
FPISCLR	5	w	**Clear the FPISLV bit.** 0_B Keep the status 1_B Clear the status
WRERRCLR	6	w	**Write error on slave interface Clear** 0_B **INACTIVE**: Clear the status 1_B **ACTIVE**: Keep the status
0	31:7	r	**Reserved** Read as 0; should be written with 0.

18.4.27 Error handling interrupt/alarm mask

INTMSK Offset address: 00000168_H

Error handling interrupt/alarm mask Kernel Reset value: $0000\ 3FFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															WRERRMSK
r															rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FPISMSK	FPIMMSK	AXIVRPECMSK	AXIFSMTOMSK	FSMPEMSK	SLVIFDPEMSK	AXIUEMSK	AXICEMSK	SLVIFAPEMSK	REGERRMSK	DPPECMSK	AXIPECMSK	S2FAMSK	S2FDMSK	A2SAMSK	A2SDMSK
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
A2SDMSK	0	rw	**A2SDAT interrupt mask** 0_B **MASKED**: Masked 1_B **UNMASKED**: Unmasked
A2SAMSK	1	rw	**A2SADDR interrupt mask** 0_B **MASKED**: Masked 1_B **UNMASKED**: Unmasked
S2FDMSK	2	rw	**S2FDAT interrupt mask** 0_B **MASKED**: Masked 1_B **UNMASKED**: Unmasked
S2FAMSK	3	rw	**S2FADDR interrupt mask** 0_B **MASKED**: Masked 1_B **UNMASKED**: Unmasked
AXIPECMSK	4	rw	**AXI parity failure interrupt mask register** 0_B **MASKED**: Masked 1_B **UNMASKED**: Unmasked
DPPECMSK	5	rw	**Data path Parity checker error mask** 0_B **MASKED**: Masked 1_B **UNMASKED**: Unmasked
REGERRMSK	6	rw	**Register space error mask** 0_B **MASKED**: Masked 1_B **UNMASKED**: Unmasked
SLVIFAPEMSK	7	rw	**AHB address parity failure interrupt mask** 0_B **MASKED**: Masked 1_B **UNMASKED**: Unmasked
AXICEMSK	8	rw	**AXI Master Interface ECC correctable error mask** 0_B **MASKED**: Masked 1_B **UNMASKED**: Unmasked
AXIUEMSK	9	rw	**AXI Master Interface ECC uncorrectable error mask** 0_B **MASKED**: Masked 1_B **UNMASKED**: Unmasked
SLVIFDPEMSK	10	rw	**AHB write data parity failure interrupt mask** 0_B **MASKED**: Masked 1_B **UNMASKED**: Unmasked
FSMPEMSK	11	rw	**FSM Parity Error mask** 0_B **MASKED**: Masked 1_B **UNMASKED**: Unmasked
AXIFSMTOMSK	12	rw	**AXI clock domain FSM time out error mask** 0_B **MASKED**: Masked 1_B **UNMASKED**: Unmasked

(table continues...)

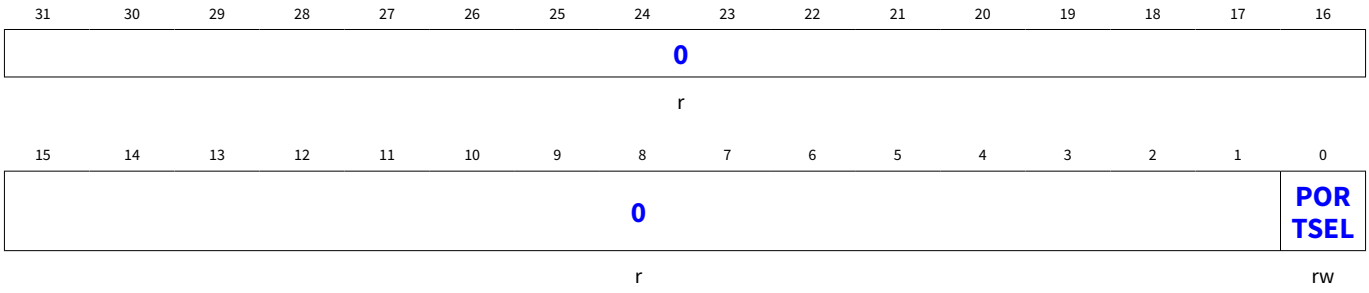
(continued)

Field	Bits	Type	Description
AXIVRPECMSK	13	rw	**AXI Valid/Ready Parity Error status mask** 0_B **MASKED**: Masked 1_B **UNMASKED**: Unmasked
FPIMMSK	14	rw	**FPIMST interrupt mask** 0_B **MASKED**: Masked 1_B **UNMASKED**: Unmasked
FPISMSK	15	rw	**FPISLV interrupt mask** 0_B **MASKED**: Masked 1_B **UNMASKED**: Unmasked
WRERRMSK	16	rw	**Write error on slave interface mask** 0_B **INACTIVE**: Unmasking 1_B **ACTIVE**: Masked
0	31:17	r	**Reserved** Read as 0; should be written with 0.

18.4.28 Control Register 2

CTRLR2 Offset address: $0000016C_H$

Control Register 2 Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
PORTSEL	0	rw	**Port selection** 0_B **Port16**: Port 16 selected High performant xSPI path over HSFAST pads on Port 16 is selected. Supports 1.8 V and 3 V devices in Octal, Quad, Dual and legacy SPI mode. Supports upto 200 MHz DDR/SDR in presence of data strobe and 133 MHz SDR when no data strobe is present. 1_B **GPIO**: Port with GPIO pads selected Selection of port 20. A low performant path over GPIO pads is selected. Supports 3 V devices in Quad, Dual and legacy SPI mode. Supports upto 25 MHz SDR and no data strobe. Note: RXD3/TXD3 signal is routed to port 15. Please see data sheet for exact mapping of all signals generalised as belonging to Port 20
0	31:1	r	**Reserved** Read as 0; should be written with 0.

18.4.29 Control Register 0

This register controls the serial data transfer. It is impossible to write to this register when the xSPI is enabled.

CTRLR0 Offset address: 00000200_H

Control Register 0 Kernel Reset value: $0000\ 481F_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SSI_IS_MST	RSVD_CTRLR0_27_31				CLK_LOOP_EN	Res	SPI_HYPERBUS_EN	SPI_FRF		RSVD_CTRLR0_20_21		Res			
r	r				rw	r	rw	rw		r		r			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD_CTRLR0_15	SSTE	SRL	Res	TMOD		SCPOL	SCPH	FRF		RSVD_CTRLR0_5	DFS				
r	rw	rw	r	rw		rw	rw	rw		r	rw				

Field	Bits	Type	Description
DFS	4:0	rw	**Data Frame Size** Selects the data frame length. When the data frame size is programmed to be less than 32 bits, the receive data is automatically right-justified by the receive logic, with the upper bits of the receive FIFO zero-padded. You must right-justify transmit data before writing into the transmit FIFO. The transmit logic ignores the upper unused bits when transmitting the data. 00_H **DFS_01_BIT**: Reserved 01_H **DFS_02_BIT**: Reserved 02_H **DFS_03_BIT**: Reserved 03_H **DFS_04_BIT**: 04-bit serial data transfer 04_H **DFS_05_BIT**: 05-bit serial data transfer 05_H **DFS_06_BIT**: 06-bit serial data transfer 06_H **DFS_07_BIT**: 07-bit serial data transfer 07_H **DFS_08_BIT**: 08-bit serial data transfer 08_H **DFS_09_BIT**: 09-bit serial data transfer 09_H **DFS_10_BIT**: 10-bit serial data transfer $0A_H$ **DFS_11_BIT**: 11-bit serial data transfer $0B_H$ **DFS_12_BIT**: 12-bit serial data transfer $0C_H$ **DFS_13_BIT**: 13-bit serial data transfer $0D_H$ **DFS_14_BIT**: 14-bit serial data transfer $0E_H$ **DFS_15_BIT**: 15-bit serial data transfer $0F_H$ **DFS_16_BIT**: 16-bit serial data transfer 10_H **DFS_17_BIT**: 17-bit serial data transfer 11_H **DFS_18_BIT**: 18-bit serial data transfer 12_H **DFS_19_BIT**: 19-bit serial data transfer 13_H **DFS_20_BIT**: 20-bit serial data transfer 14_H **DFS_21_BIT**: 21-bit serial data transfer 15_H **DFS_22_BIT**: 22-bit serial data transfer 16_H **DFS_23_BIT**: 23-bit serial data transfer 17_H **DFS_24_BIT**: 24-bit serial data transfer 18_H **DFS_25_BIT**: 25-bit serial data transfer 19_H **DFS_26_BIT**: 26-bit serial data transfer $1A_H$ **DFS_27_BIT**: 27-bit serial data transfer $1B_H$ **DFS_28_BIT**: 28-bit serial data transfer $1C_H$ **DFS_29_BIT**: 29-bit serial data transfer $1D_H$ **DFS_30_BIT**: 30-bit serial data transfer $1E_H$ **DFS_31_BIT**: 31-bit serial data transfer $1F_H$ **DFS_32_BIT**: 32-bit serial data transfer
RSVD_CTRLR0_5	5	r	**Reserved bits - read as zero**
FRF	7:6	rw	**Frame Format** Selects which serial protocol transfers the data. 00_B **SPI**: Motorola SPI Frame Format 11_B **RESERVED**: Reserved

(table continues...)

(continued)

Field	Bits	Type	Description
SCPH	8	rw	**Serial Clock Phase** Valid when the frame format (FRF) is set to Motorola SPI. The serial clock phase selects the relationship of the serial clock with the slave select signal. When SCPH = 0, data are captured on the first edge of the serial clock. When SCPH = 1, the serial clock starts toggling one cycle after the slave select line is activated, and data are captured on the second edge of the serial clock. 0_B **MIDDLE_BIT**: Serial clock toggles in middle of first bit 1_B **START_BIT**: Serial clock toggles at start of first bit
SCPOL	9	rw	**Serial Clock Polarity** Valid when the frame format (FRF) is set to Motorola SPI. Used to select the polarity of the inactive serial clock, which is held inactive when the DWC_ssi master is not actively transferring data on the serial bus. 0_B **INACTIVE_LOW**: Inactive state of serial clock is low 1_B **INACTIVE_HIGH**: Inactive state of serial clock is high
TMOD	11:10	rw	**Transfer Mode** Selects the mode of transfer for serial communication. This field does not affect the transfer duplicity. Only indicates whether the receive or transmit data are valid. In transmit-only mode, data received from the external device is not valid and is not stored in the receive FIFO memory; it is overwritten on the next transfer. In receive-only mode, transmitted data are not valid. After the first write to the transmit FIFO, the same word is retransmitted for the duration of the transfer. In transmit-and-receive mode, both transmit and receive data are valid. The transfer continues until the transmit FIFO is empty. Data received from the external device are stored into the receive FIFO memory, where it can be accessed by the host processor. 00_B **TX_AND_RX**: Transmit & Receive; Not Applicable in enhanced SPI operating mode 01_B **TX_ONLY**: Transmit only mode; Or Write in enhanced SPI operating mode 10_B **RX_ONLY**: Receive only mode; Or Read in enhanced SPI operating mode 11_B **EEPROM_READ**: EEPROM Read mode; Not Applicable in enhanced SPI operating mode
SRL	13	rw	**Shift Register Loop** Used for testing purposes only. When active, connects the transmit shift register output to the receive shift register input. Can be used in both serial-slave and serial-master modes. 0_B **NORMAL_MODE**: Normal mode operation 1_B **TESTING_MODE**: Test Mode Operation

(table continues...)

(continued)

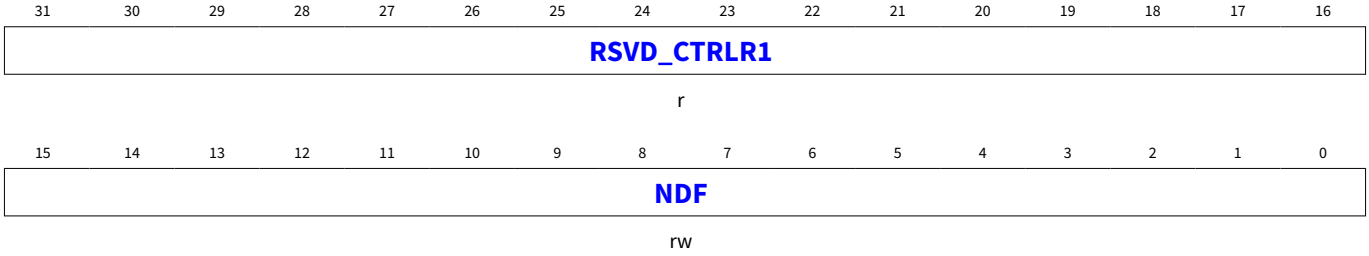
Field	Bits	Type	Description
SSTE	14	rw	**Slave Select Toggle Enable** While operating in SPI mode with clock phase (SCPH) set to 0, this register controls the behavior of the slave select line (ss_o_n) between data frames. 0_B **TOGGLE_DISABLE**: ss_*_n will stay low and sclk will run continuously for the duration of the transfer 1_B **TOGGLE_EN**: ss_*_n line will toggle between consecutive data frames, with the serial clock (sclk) being held to its default value while ss_*_n is high
RSVD_CTRLR0_15	15	r	**Reserved bits - read as zero**
RSVD_CTRLR0_20_21	21:20	r	**Reserved bits - read as zero**
SPI_FRF	23:22	rw	**SPI Frame Format** Selects data frame format for Transmitting or Receiving the data 00_B **SPI_STANDARD**: Standard SPI Format 01_B **SPI_DUAL**: Dual SPI Format 10_B **SPI_QUAD**: Quad SPI Format 11_B **SPI_OCTAL**: Octal SPI Format
SPI_HYPERBUS_EN	24	rw	**SPI Hyperbus Frame format enable** Selects if data frame format for Transmitting/Receiving the data is in Hyperbus mode. 0_B **DISABLE**: Disable Hyperbus Format 1_B **ENABLE**: Enable Hyperbus Format
CLK_LOOP_EN	26	rw	**Clock loop back enable bit** Once this bit is set to 1, xSPI will use looped back clock (mst_sclk_in) to capture read data 0_B **DISABLE**: Disable Clock Loop Back 1_B **ENABLE**: Enable Clock Loop Back
RSVD_CTRLR0_27_31	30:27	r	**Reserved bits - read as zero**
SSI_IS_MST	31	r	**Reserved** Read as 0. xSPI is always master. xSPI in slave mode is not supported.

18.4.30 Control Register 1

This register exists only when the DWC_ssi is configured as a master device. When the DWC_ssi is configured as a serial slave, writing to this location has no effect; reading from this location returns 0. Control register 1 controls the end of serial transfers when in receive-only mode. It is impossible to write to this register when the DWC_ssi is enabled.

CTRLR1 Offset address: 00000204_H

Control Register 1 Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
NDF	15:0	rw	**Number of Data Frames** When TMOD = 10_B or TMOD = 11_B , this register field sets the number of data frames to be continuously received by the DWC_ssi. The DWC_ssi continues to receive serial data until the number of data frames received is equal to this register value plus 1, which enables you to receive up to 64 KB of data in a continuous transfer. When SPI_CTRLR0.CLK_STRETCH_EN=1 and TMOD = 01_B, this register field sets the number of data frames to be continously transmitted by DWC_ssi. If the Transmit FIFO goes empty in-between, DWC_ssi masks the serial clock (sclk_out), i.e, keep the sclk_out HIGH or LOW state based on the SCPOL settings and wait for rest of the data until the programmed amount of frames are transferred successfully.
RSVD_CTRLR1	31:16	r	**Reserved bits - read as zero**

18.4.31 xSPI kernel Enable Register

This register enables and disables the DWC_ssi.

SSIENR Offset address: 00000208_H

xSPI kernel Enable Register Kernel Reset value: $0000\ 0001_H$

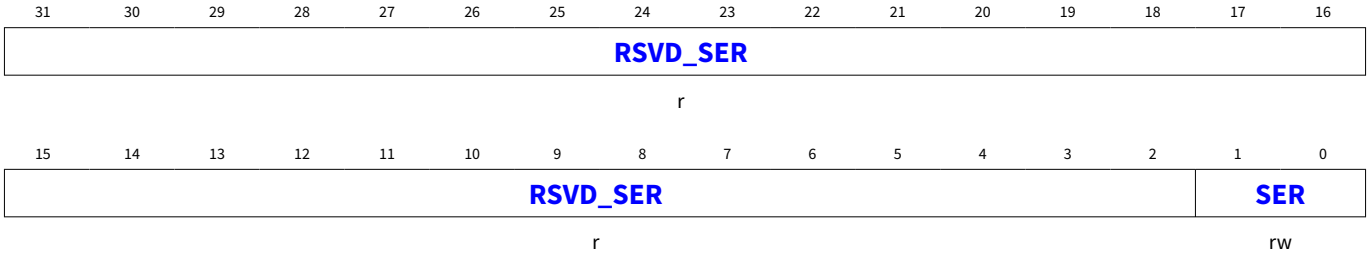
Field	Bits	Type	Description
SSIC_EN	0	rwh	**SSI Enable** Enables and disables all DWC_ssi operations. When disabled, all serial transfers are halted immediately. Transmit and receive FIFO buffers are cleared when the device is disabled. It is impossible to program some of the DWC_ssi control registers when enabled. 0_B **DISABLE**: Disables DWC_ssi 1_B **ENABLED**: Enables DWC_ssi
RSVD_SSIENR	31:1	r	**Reserved bits - read as zero**

18.4.32 Slave Enable Register

The register enables the individual slave select output lines from the DWC_ssi master. Up to 2 slave-select output pins are available on the xSPI master. You cannot write to this register when DWC_ssi is busy and when SSIC_EN = 1.

SER Offset address: 00000210_H

Slave Enable Register Kernel Reset value: $0000\ 0001_H$



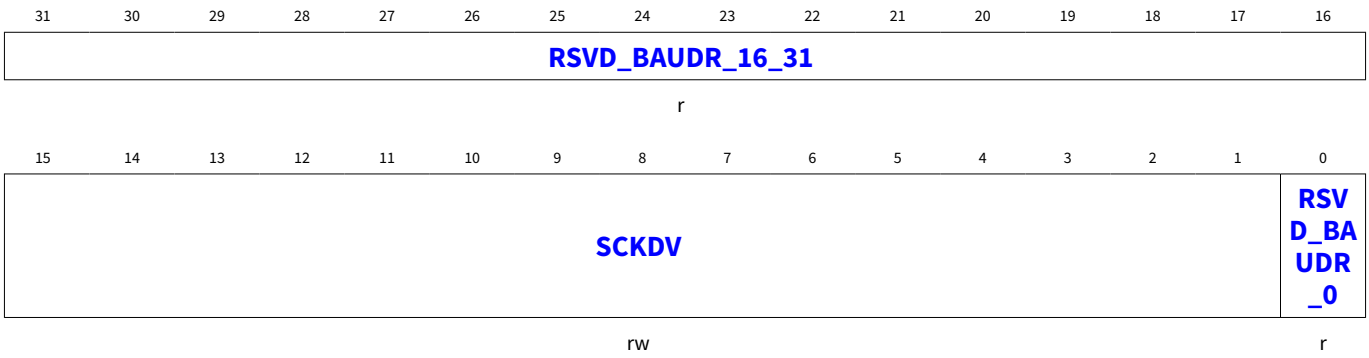
Field	Bits	Type	Description
SER	1:0	rw	**Slave Select Enable Flag** Each bit in this register corresponds to a slave select line (ss_x_n) from the DWC_ssi master. When a bit in this register is set (1), the corresponding slave select line from the master is activated when a serial transfer begins. It should be noted that setting or clearing bits in this register have no effect on the corresponding slave select outputs until a transfer is started. Before beginning a transfer, you should enable the bit in this register that corresponds to the slave device with which the master wants to communicate. When not operating in broadcast mode, only one bit in this field should be set.
RSVD_SER	31:2	r	**Reserved bits - read as zero**

18.4.33 Baud Rate Select

The register derives the frequency of the serial clock that regulates the data transfer. The 15-bit wide bit-field along with bit 0 which is always set to 0, defines the ssi_clk divider value. It is impossible to write to this register when the xSPI is enabled.

BAUDR Offset address: 00000214_H

Baud Rate Select Kernel Reset value: $0000\ 0004_H$



Field	Bits	Type	Description
RSVD_BAUDR_0	0	r	**Reserved bits - read as zero**
SCKDV	15:1	rw	**SSI Clock Divider** The LSB of the BAUDR register is always a 0, which ensures that an even value is held in this register only. If the value of this bitfield is 0, the serial output clock (sclk_out) is disabled. The frequency of the sclk_out is derived from the following equation: Fsclk_out = Fssi_clk/BAUDR where BAUDR is any even value between 2 and 65534 (BAUDR = SCKDV*2). For example: for Fssi_clk = 3.6864MHz and BAUDR =2 Fsclk_out = 3.6864/2 = 1.8432MHz
RSVD_BAUDR_16_31	31:16	r	**Reserved bits - read as zero**

18.4.34 Transmit FIFO Threshold Level

This register controls the threshold value for the transmit FIFO memory..

TXFTLR Offset address: 00000218_H

Transmit FIFO Threshold Level Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVD_TXFTHR									TXFTHR						
r									rw						

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD_TXFTLR												TFT			
r												rw			

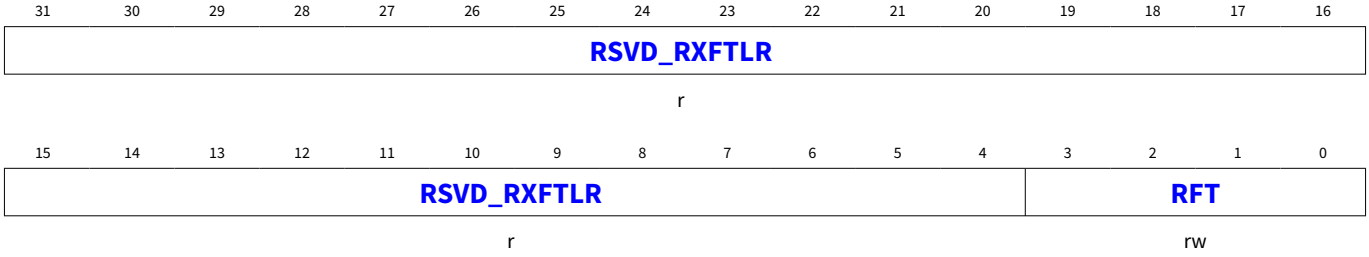
Field	Bits	Type	Description
TFT	3:0	rw	**Transmit FIFO Threshold** Controls the level of entries (or below) at which the transmit FIFO controller triggers an interrupt. The FIFO depth is 16; this register is sized to the number of address bits needed to access the FIFO. If you attempt to set this value greater than or equal to the depth of the FIFO, this field is not written and retains its current value. When the number of transmit FIFO entries is less than or equal to this value, the transmit FIFO empty interrupt is triggered.
RSVD_TXFTLR	15:4	r	**Reserved bits - read as zero**
TXFTHR	22:16	rw	**Transfer start FIFO level** Used to control the level of entries in transmit FIFO above which transfer will start on serial line. This register can be used to ensure that sufficient data is present in transmit FIFO before starting a write operation on serial line. In Internal DMA mode, this field sets the minimum amount of data entries present in the FIFO after which DWC_ssi starts the transfer. This field is valid only for Master mode of operation.
RSVD_TXFTHR	31:23	r	**Reserved bits - read as zero**

18.4.35 Receive FIFO Threshold Level

This register controls the threshold value for the receive FIFO memory..

RXFTLR Offset address: $0000021C_H$

Receive FIFO Threshold Level Kernel Reset value: $0000\ 0000_H$



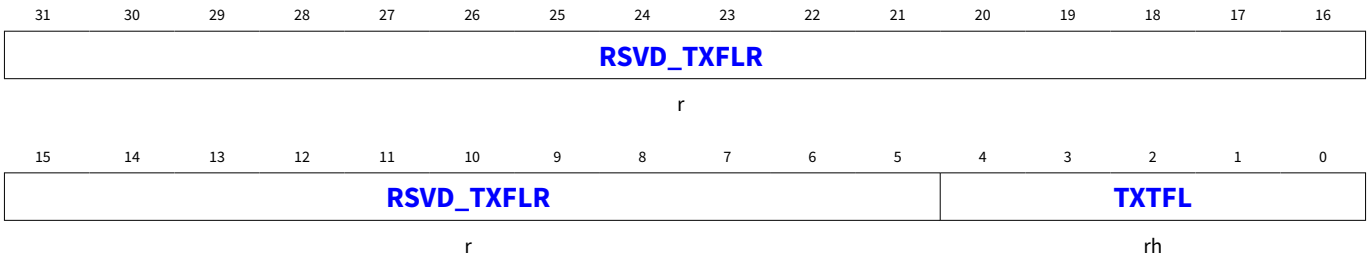
Field	Bits	Type	Description
RFT	3:0	rw	**Receive FIFO Threshold** Controls the level of entries (or above) at which the receive FIFO controller triggers an interrupt. The FIFO depth is 16. This register is sized to the number of address bits needed to access the FIFO. If you attempt to set this value greater than the depth of the FIFO, this field is not written and retains its current value. When the number of receive FIFO entries is greater than or equal to this value + 1, the receive FIFO full interrupt is triggered.
RSVD_RXFTLR	31:4	r	**Reserved bits - read as zero**

18.4.36 Transmit FIFO Level Register

This register contains the number of valid data entries in the transmit FIFO memory.

TXFLR Offset address: 00000220_H

Transmit FIFO Level Register Kernel Reset value: $0000\ 0000_H$



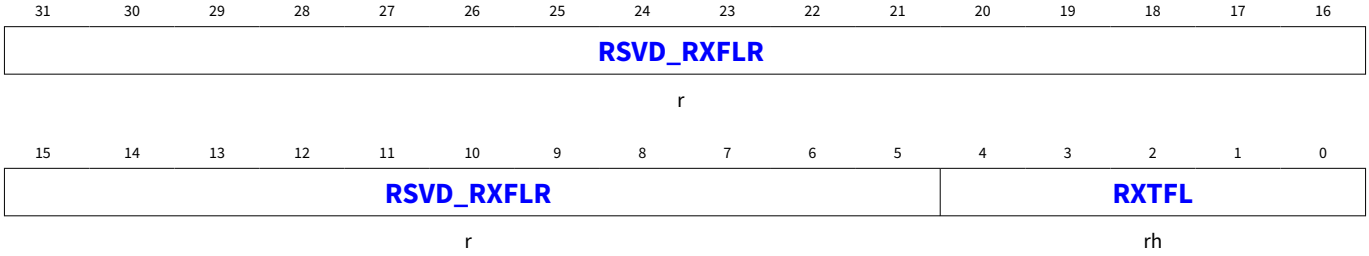
Field	Bits	Type	Description
TXTFL	4:0	rh	**Transmit FIFO Level** Contains the number of valid data entries in the transmit FIFO.
RSVD_TXFLR	31:5	r	**Reserved bits - read as zero**

18.4.37 Receive FIFO Level Register

This register contains the number of valid data entries in the receive FIFO memory. This register can be read at any time.

RXFLR Offset address: 00000224_H

Receive FIFO Level Register Kernel Reset value: $0000\ 0000_H$



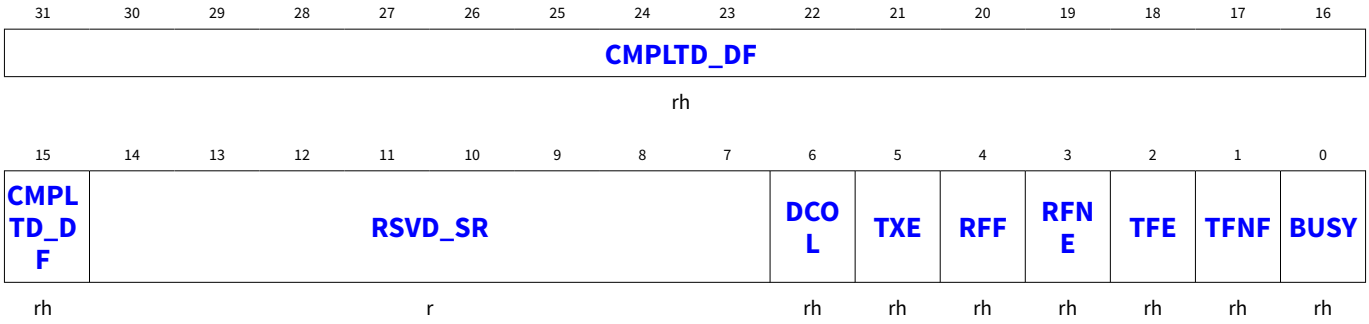
Field	Bits	Type	Description
RXTFL	4:0	rh	**Receive FIFO Level** Contains the number of valid data entries in the receive FIFO.
RSVD_RXFLR	31:5	r	**Reserved bits - read as zero**

18.4.38 Status Register

This is a read-only register used to indicate the current transfer status, FIFO status, and any transmission/ reception errors that may have occurred. The status register may be read at any time. None of the bits in this register request an interrupt.

SR Offset address: 00000228_H

Status Register Kernel Reset value: $0000\ 0006_H$



Field	Bits	Type	Description
BUSY	0	rh	**SSI Busy Flag** When set, indicates that a serial transfer is in progress; when cleared indicates that the DWC_ssi is idle or disabled. 0_B **INACTIVE**: DWC_ssi is idle or disabled 1_B **ACTIVE**: DWC_ssi is actively transferring data

(table continues...)

(continued)

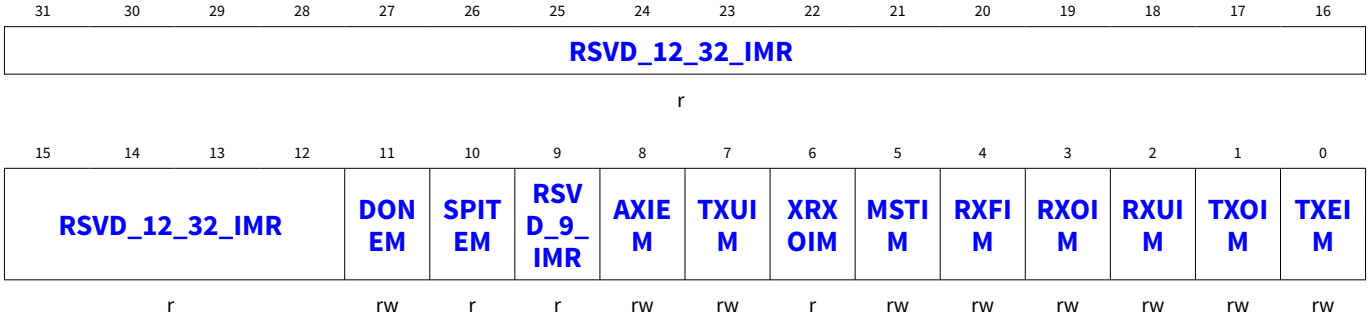
Field	Bits	Type	Description
TFNF	1	rh	**Transmit FIFO Not Full** Set when the transmit FIFO contains one or more empty locations, and is cleared when the FIFO is full. 0_B **FULL**: Tx FIFO is full 1_B **NOT_FULL**: Tx FIFO is not Full
TFE	2	rh	**Transmit FIFO Empty** When the transmit FIFO is completely empty, this bit is set. When the transmit FIFO contains one or more valid entries, this bit is cleared. This bit field does not request an interrupt. 0_B **NOT_EMPTY**: Transmit FIFO is not empty 1_B **EMPTY**: Transmit FIFO is empty
RFNE	3	rh	**Receive FIFO Not Empty** Set when the receive FIFO contains one or more entries and is cleared when the receive FIFO is empty. This bit can be polled by software to completely empty the receive FIFO. 0_B **EMPTY**: Receive FIFO is empty 1_B **NOT_EMPTY**: Receive FIFO is not empty
RFF	4	rh	**Receive FIFO Full** When the receive FIFO is completely full, this bit is set. When the receive FIFO contains one or more empty location, this bit is cleared. 0_B **NOT_FULL**: Receive FIFO is not full 1_B **FULL**: Receive FIFO is full
TXE	5	rh	**Transmission Error** Set if the transmit FIFO is empty when a transfer is started. This bit can be set only when the DWC_ssi is configured as a slave device. Data from the previous transmission is resent on the txd line. This bit is cleared when read. 0_B **NO_ERROR**: No Error 1_B **TX_ERROR**: Transmission Error
DCOL	6	rh	**Data Collision Error** Relevant only when the DWC_ssi is configured as a master device. This bit will be set if ss_in_n input is asserted by other master, when the DWC_ssi master is in the middle of the transfer. This informs the processor that the last data transfer was halted before completion. This bit is cleared when read. 0_B **NO_ERROR_CONDITION**: No Error 1_B **TX_COLLISION_ERROR**: Transmit Data Collision Error
RSVD_SR	14:7	r	**Reserved bits - read as zero**
CMPLTD_DF	31:15	rh	**Completed Data frames** This field indicates total data frames transferred in the previous internal DMA transfer

18.4.39 Interrupt Mask Register

This register masks or enables all interrupts generated by the xSPI.

IMR Offset address: 0000022C$_H$

Interrupt Mask Register Kernel Reset value: 0000 09BE$_H$



Field	Bits	Type	Description
TXEIM	0	rw	**Transmit FIFO Empty Interrupt Mask** 0_B **MASKED**: ssi_txe_intr interrupt is masked 1_B **UNMASKED**: ssi_txe_intr interrupt is not masked
TXOIM	1	rw	**Transmit FIFO Overflow Interrupt Mask** 0_B **MASKED**: ssi_txo_intr interrupt is masked 1_B **UNMASKED**: ssi_txo_intr interrupt is not masked
RXUIM	2	rw	**Receive FIFO Underflow Interrupt Mask** 0_B **MASKED**: ssi_rxu_intr interrupt is masked 1_B **UNMASKED**: ssi_rxu_intr interrupt is not masked
RXOIM	3	rw	**Receive FIFO Overflow Interrupt Mask** 0_B **MASKED**: ssi_rxo_intr interrupt is masked 1_B **UNMASKED**: ssi_rxo_intr interrupt is not masked
RXFIM	4	rw	**Receive FIFO Full Interrupt Mask0** - ssi_rxf_intr interrupt is masked 1 - ssi_rxf_intr interrupt is not masked 0_B **MASKED**: ssi_rxf_intr interrupt is masked 1_B **UNMASKED**: ssi_rxf_intr interrupt is not masked
MSTIM	5	rw	**Multi-Master Contention Interrupt Mask** 0_B **MASKED**: ssi_mst_intr interrupt is masked 1_B **UNMASKED**: ssi_mst_intr interrupt is not masked
XRXOIM	6	r	**XIP Receive FIFO Overflow Interrupt Mask** 0_B **MASKED**: ssi_xrxo_intr interrupt is masked 1_B **UNMASKED**: ssi_xrxo_intr interrupt is not masked
TXUIM	7	rw	**Transmit FIFO Underflow Interrupt Mask** 0_B **MASKED**: ssi_txu_intr interrupt is masked 1_B **UNMASKED**: ssi_txu_intr interrupt is not masked

(table continues...)

(continued)

Field	Bits	Type	Description
AXIEM	8	rw	**AXI Error Interrupt Mask** 0_B **MASKED**: ssi_axie_intr interrupt is masked 1_B **UNMASKED**: ssi_axie_intr interrupt is not masked
RSVD_9_IMR	9	r	**Reserved bits - read as zero**
SPITEM	10	r	**SPI Transmit Error Interrupt Mask** 0_B **MASKED**: ssi_spite_intr interrupt is masked 1_B **UNMASKED**: ssi_spite_intr interrupt is not masked
DONEM	11	rw	**SSI Done Interrupt Mask** 0_B **MASKED**: ssi_done_intr interrupt is masked 1_B **UNMASKED**: ssi_done_intr interrupt is not masked
RSVD_12_32_IMR	31:12	r	**Reserved bits - read as zero**

18.4.40 Interrupt Status Register

This register reports the status of the DWC_ssi interrupts after they have been masked.

ISR Offset address: 00000230_H

Interrupt Status Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVD_12_32_RISR															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD_12_32_RISR				DONES	SPITES	RSVD_9_RISR	AXIES	TXUIS	XRXOIS	MSTIS	RXFIS	RXOIS	RXUIS	TXOIS	TXEIS
r				rh	rh	r	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
TXEIS	0	rh	**Transmit FIFO Empty Interrupt Status** 0_B **INACTIVE**: ssi_txe_intr interrupt is not active after masking 1_B **ACTIVE**: ssi_txe_intr interrupt is active after masking
TXOIS	1	rh	**Transmit FIFO Overflow Interrupt Status** 0_B **INACTIVE**: ssi_txo_intr interrupt is not active after masking 1_B **ACTIVE**: ssi_txo_intr interrupt is active after masking
RXUIS	2	rh	**Receive FIFO Underflow Interrupt Status** 0_B **INACTIVE**: ssi_rxu_intr interrupt is not active after masking 1_B **ACTIVE**: ssi_rxu_intr interrupt is active after masking
RXOIS	3	rh	**Receive FIFO Overflow Interrupt Status** 0_B **INACTIVE**: ssi_rxo_intr interrupt is not active after masking 1_B **ACTIVE**: ssi_rxo_intr interrupt is active after masking

(table continues...)

(continued)

Field	Bits	Type	Description
RXFIS	4	rh	**Receive FIFO Full Interrupt Status** 0_B **INACTIVE**: ssi_rxf_intr interrupt is not active after masking 1_B **ACTIVE**: ssi_rxf_intr interrupt is active after masking
MSTIS	5	rh	**Multi-Master Contention Interrupt Status** 0_B **INACTIVE**: ssi_mst_intr interrupt is not active after masking 1_B **ACTIVE**: ssi_mst_intr interrupt is active after masking
XRXOIS	6	rh	**XIP Receive FIFO Overflow Interrupt Status** 0_B **INACTIVE**: ssi_xrxo_intr interrupt is not active after masking 1_B **ACTIVE**: ssi_xrxo_intr interrupt is active after masking
TXUIS	7	rh	**Transmit FIFO Underflow Interrupt Status** 0_B **INACTIVE**: ssi_txu_intr interrupt is not active after masking 1_B **ACTIVE**: ssi_txu_intr interrupt is active after masking
AXIES	8	rh	**AXI Error Interrupt Status** 0_B **INACTIVE**: ssi_axie_intr interrupt is not active after masking 1_B **ACTIVE**: ssi_axie_intr interrupt is active after masking
RSVD_9_RISR	9	r	**Reserved bits - read as zero**
SPITES	10	rh	**SPI Transmit Error Interrupt** 0_B **INACTIVE**: ssi_spite_intr interrupt is not active after masking 1_B **ACTIVE**: ssi_spite_intr interrupt is active after masking
DONES	11	rh	**SSI Done Interrupt Status** 0_B **INACTIVE**: ssi_done_intr interrupt is not active after masking 1_B **ACTIVE**: ssi_done_intr interrupt is active after masking
RSVD_12_32_RISR	31:12	r	**Reserved bits - read as zero**

18.4.41 Raw Interrupt Status Register

RISR Offset address: 00000234_H

Raw Interrupt Status Register Kernel Reset value: $0000\ 0001_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVD_12_32_RISR															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD_12_32_RISR				DONER	SPITER	RSVD_9_RISR	AXIER	TXUIR	XRXOIR	MSTIR	RXFIR	RXOIR	RXUIR	TXOIR	TXEIR
r				rh	rh	r	rh	rh	rh	rh	rh	rh	rh	rh	rh

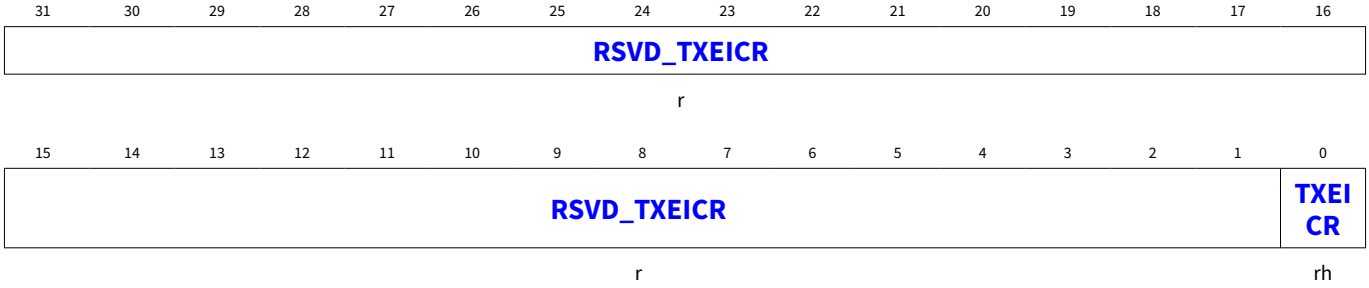
Field	Bits	Type	Description
TXEIR	0	rh	**Transmit FIFO Empty Raw Interrupt Status** 0_B **INACTIVE**: ssi_txe_intr interrupt is not active prior masking 1_B **ACTIVE**: ssi_txe_intr interrupt is active prior to masking
TXOIR	1	rh	**Transmit FIFO Overflow Raw Interrupt Status** 0_B **INACTIVE**: ssi_txo_intr interrupt is not active prior masking 1_B **ACTIVE**: ssi_txo_intr interrupt is active prior to masking
RXUIR	2	rh	**Receive FIFO Underflow Raw Interrupt Status** 0_B **INACTIVE**: ssi_rxu_intr interrupt is not active prior masking 1_B **ACTIVE**: ssi_rxu_intr interrupt is active prior to masking
RXOIR	3	rh	**Receive FIFO Overflow Raw Interrupt Status** 0_B **INACTIVE**: ssi_rxo_intr interrupt is not active prior masking 1_B **ACTIVE**: ssi_rxo_intr interrupt is active prior to masking
RXFIR	4	rh	**Receive FIFO Full Raw Interrupt Status** 0_B **INACTIVE**: ssi_rxf_intr interrupt is not active prior masking 1_B **ACTIVE**: ssi_rxf_intr interrupt is active prior to masking
MSTIR	5	rh	**Multi-Master Contention Raw Interrupt Status** 0_B **INACTIVE**: ssi_mst_intr interrupt is not active prior masking 1_B **ACTIVE**: ssi_mst_intr interrupt is active prior to masking
XRXOIR	6	rh	**XIP Receive FIFO Overflow Raw Interrupt Status** 0_B **INACTIVE**: ssi_xrxo_intr interrupt is not active prior masking 1_B **ACTIVE**: ssi_xrxo_intr interrupt is active prior to masking
TXUIR	7	rh	**Transmit FIFO Underflow Interrupt Raw Status** 0_B **INACTIVE**: ssi_txu_intr interrupt is not active prior masking 1_B **ACTIVE**: ssi_txu_intr interrupt is active prior to masking
AXIER	8	rh	**AXI Error Interrupt Raw Status** 0_B **INACTIVE**: ssi_axie_intr interrupt is not active prior masking 1_B **ACTIVE**: ssi_axie_intr interrupt is active prior to masking
RSVD_9_RISR	9	r	**Reserved bits - read as zero**
SPITER	10	rh	**SPI Transmit Error Interrupt status** This bit gets set, If SPI Master fails to get a READY status from the slave until the amount of time defined in SPI_CTRLR1.MAX_WS field, then it will stop the SPI transfer and the FIFO is flushed (in case of write operation). 0_B **INACTIVE**: ssi_spite_intr interrupt is not active prior to masking 1_B **ACTIVE**: ssi_spite_intr interrupt is active prior to masking
DONER	11	rh	**SSI Done Interrupt Raw Status** 0_B **INACTIVE**: ssi_done_intr interrupt is not active prior masking 1_B **ACTIVE**: ssi_done_intr interrupt is active prior to masking
RSVD_12_32_RISR	31:12	r	**Reserved bits - read as zero**

18.4.42 Transmit FIFO Error Interrupt Clear Register Register

Transmit FIFO Error Interrupt Clear Register

TXEICR Offset address: 00000238_H

Transmit FIFO Error Interrupt Clear Register Register Kernel Reset value: $0000\ 0000_H$

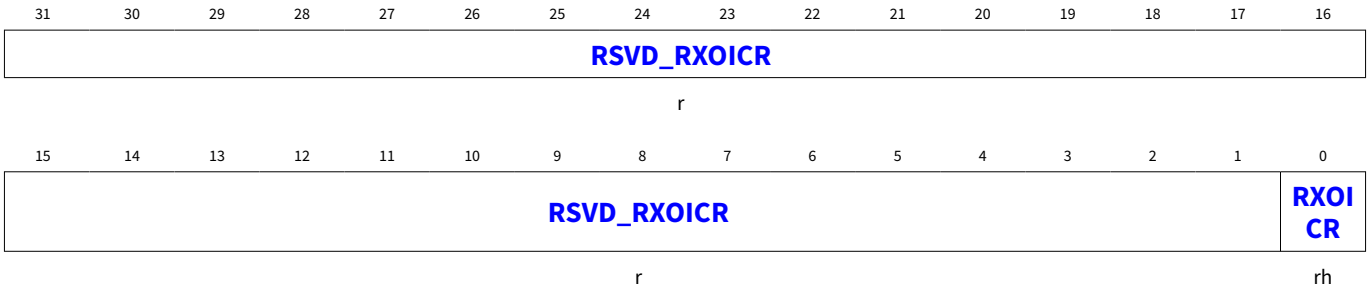


Field	Bits	Type	Description
TXEICR	0	rh	**Clear Transmit FIFO Overflow/Underflow Interrupt** This register reflects the status of the interrupt. A read from this register clears the ssi_txo_intr/ssi_txu_intr interrupt; writing has no effect.
RSVD_TXEICR	31:1	r	**Reserved bits - read as zero**

18.4.43 Receive FIFO Overflow Interrupt Clear Register

RXOICR Offset address: $0000023C_H$

Receive FIFO Overflow Interrupt Clear Register Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
RXOICR	0	rh	**Clear Receive FIFO Overflow Interrupt** This register reflects the status of the interrupt. A read from this register clears the ssi_rxo_intr interrupt; writing has no effect.
RSVD_RXOICR	31:1	r	**Reserved bits - read as zero**

18.4.44 Receive FIFO Underflow Interrupt Clear Register

RXUICR Offset address: 00000240_H

Receive FIFO Underflow Interrupt Clear Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVD_RXUICR															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD_RXUICR															RXUICR
r															rh

Field	Bits	Type	Description
RXUICR	0	rh	**Clear Receive FIFO Underflow Interrupt** This register reflects the status of the interrupt. A read from this register clears the ssi_rxu_intr interrupt; writing has no effect.
RSVD_RXUICR	31:1	r	**Reserved bits - read as zero**

18.4.45 Multi-Master Interrupt Clear Register

MSTICR Offset address: 00000244_H

Multi-Master Interrupt Clear Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVD_MSTICR															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD_MSTICR															MSTICR
r															rh

Field	Bits	Type	Description
MSTICR	0	rh	**Clear Multi-Master Contention Interrupt**
RSVD_MSTICR	31:1	r	**Reserved bits - read as zero**

18.4.46 Interrupt Clear Register

ICR Offset address: 00000248_H

Interrupt Clear Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVD_ICR															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD_ICR															ICR
r															rh

Field	Bits	Type	Description
ICR	0	rh	**Clear Interrupts** This register is set if any of the interrupts below are active. A read clears the ssi_txo_intr, ssi_rxu_intr, ssi_rxo_intr, and the ssi_mst_intr interrupts. Writing to this register has no effect.
RSVD_ICR	31:1	r	**Reserved bits - read as zero**

18.4.47 DMA Control Register

DMA Control Register.

DMACR Offset address: $0000024C_H$

DMA Control Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVD_DMACR													AID		
r													rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AID	APROT			ACACHE				RSVD_DMACR7	AINC	RSVD_DMACR5	ATW		IDMAE	TDMAE	RDMAE
rw	rw			rw				r	rw	r	rw		rw	r	r

Field	Bits	Type	Description
RDMAE	0	r	**Receive DMA Enable** This bit enables/disables the receive FIFO DMA channel. 0_B **DISABLE**: Receive DMA disabled 1_B **ENABLED**: Receive DMA enabled
TDMAE	1	r	**Transmit DMA Enable** This bit enables/disables the transmit FIFO DMA channel. 0_B **DISABLE**: Transmit DMA disabled 1_B **ENABLED**: Transmit DMA enabled
IDMAE	2	rw	**Internal DMA Enable** This bit should be enabled only when CTRLR0.FRF = 0 (Motorola SPI) and CTRLR0.SPI_FRF > 0.
ATW	4:3	rw	**AXI transfer width for DMA transfers mapped to arsize/awsize** Values: 0x0: 1 byte 0x1: 2 bytes 0x2: 4 bytes 0x3: 8 bytes.
RSVD_DMACR5	5	r	**Reserved bits - read as zero**
AINC	6	rw	**Address Increment** Indicates whether to increment the AXI address on every transfer. 1 = Increment 0 = No Change Note: Increment aligns the address to the next DMACR.ATW boundary

(table continues...)

(continued)

Field	Bits	Type	Description
RSVD_DMACR7	7	r	**Reserved bits - read as zero**
ACACHE	11:8	rw	**AXI arcache/awcache signal value** This signal is tied to 0. Setting this bit will not have any impact
APROT	14:12	rw	**AXI arprot/awprot signal value** 1: access to supervisor mode protected register allowed 0: access to supervisor protected register not allowed
AID	18:15	rw	**AXI awid/arid signal value** This signal is tied to 0. Setting this bit will not have any impact
RSVD_DMACR	31:19	r	**Reserved bits - read as zero**

18.4.48 Destination Burst Length

AXIAWLEN Offset address: 00000250_H

Destination Burst Length Kernel Reset value: $0000\ 0700_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVD_AXIAWLEN2															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AWLEN								RSVD_AXIAWLEN							
rw								r							

Field	Bits	Type	Description
RSVD_AXIAWLEN	7:0	r	**Reserved bits - read as zero**
AWLEN	15:8	rw	**Destination Burst Length** AXI Master Interface Write Burst Length in bytes
RSVD_AXIAWLEN2	31:16	r	**Reserved bits - read as zero**

18.4.49 Source Burst Length

AXIARLEN Offset address: 00000254_H

Source Burst Length Kernel Reset value: $0000\ 0700_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVD_AXIARLEN2															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ARLEN								RSVD_AXIARLEN							
rw								r							

Field	Bits	Type	Description
RSVD_AXIARLEN	7:0	r	**Reserved bits - read as zero**
ARLEN	15:8	rw	**Source Burst Length** AXI Master Interface Read Burst Length in bytes
RSVD_AXIARLEN2	31:16	r	**Reserved bits - read as zero**

18.4.50 xSPI kernel component version

This read-only register stores the specific DWC_ssi component version.

SSIC_VERSION_ID Offset address: $0000025C_H$

xSPI kernel component version Kernel Reset value: $3130\ 332A_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SSIC_COMP_VERSION															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SSIC_COMP_VERSION															
r															

Field	Bits	Type	Description
SSIC_COMP_VERSION	31:0	r	**Contains the hex representation of the Synopsys component version** Consists of ASCII value for each number in the version, followed by *. For example 31_30_33_2A represents the version 1.03*.

18.4.51 xSPI Data Register i

The DWC_ssi data register is a 32-bit read/write buffer for the transmit/receive FIFOs. When the register is read, data in the receive FIFO buffer is accessed. When it is written to, data are moved into the transmit FIFO buffer; a write can occur only when SSIC_EN = 1. FIFOs are reset when SSIC_EN = 0. Note The DR register in the xSPI occupies thirty-six 32-bit address locations of the memory map to facilitate AHB burst transfers. Writing to any of these address locations has the same effect as pushing the data from the AHB bus into the transmit FIFO. Reading from any of these locations has the same effect as popping data from the receive FIFO onto the AHB bus. The FIFO buffers on the DWC_ssi are not addressable.

DRi (i=0-35) Offset address: 00000260_H+i*4

xSPI Data Register i Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DR															
rwh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DR															
rwh															

Field	Bits	Type	Description
DR	31:0	rwh	**Data Register** When writing to this register, you must right-justify the data. Read data are automatically right-justified. Read = Receive FIFO buffer Write = Transmit FIFO buffer.

18.4.52 RX Sample Delay Register

This register control the number of ssi_clk cycles that are delayed (from the default sample time) before the actual sample of the rxd input occurs. It is impossible to write to this register when the DWC_ssi is enabled.

RX_SAMPLE_DELAY Offset address: 000002F0$_H$

RX Sample Delay Register Kernel Reset value: 0000 0000$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVD1_RX_SAMPLE_DLY															SE
r															rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD0_RX_SAMPLE_DLY								RSD							
r								rw							

Field	Bits	Type	Description
RSD	7:0	rw	**Receive Data (rxd) Sample Delay** This register is used to delay the sample of the rxd input port. Each value represents a single ssi_clk delay on the sample of rxd. Note; If this register is programmed with a value exceeding 4 (which is the depth of the internal shift registers), zero delay will be applied to the rxd sample.
RSVD0_RX_SAMPLE_DLY	15:8	r	**Reserved bits - read as zero**
SE	16	rw	**Receive Data (rxd) Sampling Edge** This register is used to decide the sampling edge for RXD signal with ssi_clk. This bit is set to 1 then negative edge of ssi_clk will be used to sample the incoming data, otherwise positive edge will be used for sampling.
RSVD1_RX_SAMPLE_DLY	31:17	r	**Reserved bits - read as zero**

18.4.53 SPI_CTRLR0 - SPI Control Register

This register is used to control the serial data transfer in enhanced SPI mode of operation. The register is relevant only when either CTRLR0.SPI_DWS_EN is set to 1 or CTRLR0.SPI_FRF is set to either 01_B or 10_B or 11_B. It is not possible to write to this register when the DWC_ssi is enabled (SSIC_EN=1).

SPI_CTRLR0 Offset address: 000002F4$_H$

SPI_CTRLR0 - SPI Control Register Kernel Reset value: 0800 0A10$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVD_SPI_CTRLR0	CLK_STRETCH_EN	XIP_PREFETCH_EN	RSVD_SPI_CTRLR0_28	XIP_MBL		SPI_RXDS_SIG_EN	SPI_DM_EN	RSVD_SPI_CTRLR0_22_23		SSIC_XIP_CONT_XFER_EN	XIP_INST_EN	XIP_DFS_HC	SPI_RXDS_EN	INST_DDR_EN	SPI_DDR_EN
r	rw	r	r	rw		rw	rw	r		rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WAIT_CYCLES					RSVD_SPI_CTRLR0_10	INST_L		XIP_MD_BIT_EN	RSVD_SPI_CTRLR0_6	ADDR_L				TRANS_TYPE	
rw					r	rw		rw	r	rw				rw	

Field	Bits	Type	Description
TRANS_TYPE	1:0	rw	**Address and instruction transfer format** Selects whether DWC_ssi will transmit instruction/address either in Standard SPI mode or the SPI mode selected in CTRLR0.SPI_FRF field. 00_B **TT0**: Instruction and Address will be sent in Standard SPI Mode 01_B **TT1**: Instruction will be sent in Standard SPI Mode and Address will be sent in the mode specified by CTRLR0.SPI_FRF 10_B **TT2**: Both Instruction and Address will be sent in the mode specified by SPI_FRF 11_B **TT3**: Reserved
ADDR_L	5:2	rw	**This bit defines Length of Address to be transmitted** Only after this much bits are programmed in to the FIFO the transfer can begin. 0_H **ADDR_L0**: No Address 1_H **ADDR_L4**: 4 bit Address length 2_H **ADDR_L8**: 8 bit Address length 3_H **ADDR_L12**: 12 bit Address length 4_H **ADDR_L16**: 16 bit Address length 5_H **ADDR_L20**: 20 bit Address length 6_H **ADDR_L24**: 24 bit Address length 7_H **ADDR_L28**: 28 bit Address length 8_H **ADDR_L32**: 32 bit Address length 9_H **ADDR_L36**: 36 bit Address length A_H **ADDR_L40**: 40 bit Address length B_H **ADDR_L44**: 44 bit Address length C_H **ADDR_L48**: 48 bit Address length D_H **ADDR_L52**: 52 bit Address length E_H **ADDR_L56**: 56 bit Address length F_H **ADDR_L60**: 60 bit Address length
RSVD_SPI_CTRLR0_6	6	r	**Reserved bits - read as zero**

(table continues...)

(continued)

Field	Bits	Type	Description
XIP_MD_BIT_EN	7	rw	**Mode bits enable in XIP mode** If this bit is set to 1, then in XIP mode of operation DWC_ssi will insert mode bits after the address phase. These bits are set in register XIP_MODE_BITS register. The length of mode bits is always set to 8 bits.
INST_L	9:8	rw	**Dual/Quad/Octal mode instruction length in bits** 00_B **INST_L0**: No Instruction 01_B **INST_L4**: 4 bit Instruction length 10_B **INST_L8**: 8 bit Instruction length 11_B **INST_L16**: 16 bit Instruction length
RSVD_SPI_CTRLR0_10	10	r	**Reserved bits - read as zero**
WAIT_CYCLES	15:11	rw	**Wait cycles** Wait cycles in Dual/Quad/Octal mode between control frames transmit and data reception. Specified as number of SPI clock cycles.
SPI_DDR_EN	16	rw	**SPI DDR Enable bit** This will enable Dual-data rate transfers in Dual/Quad/Octal frame formats of SPI.
INST_DDR_EN	17	rw	**Instruction DDR Enable bit** This will enable Dual-data rate transfer for Instruction phase.
SPI_RXDS_EN	18	rw	**Read data strobe enable bit** Once this bit is set to 1 DWC_ssi will use Read data strobe (rxds) to capture read data.
XIP_DFS_HC	19	rw	**Fix DFS for XIP transfers** If this bit is set to 1 then data frame size for XIP transfers will be fixed to the programmed value in CTRLR0.DFS, the number of data frames to fetch will be determined by SIZE and BURST signals received on the AHB bus. If this bit is set to 0 then data frame size and number of data frames to fetch will be determined by the SIZE and BURST signals on the AHB bus
XIP_INST_EN	20	rw	**XIP instruction enable bit** If this bit is set to 1 then XIP transfers will also have instruction phase. The instruction op-codes will be chosen from XIP_INCR_INST or XIP_WRAP_INST registers bases on AHB transfer type.
SSIC_XIP_CONT_XFER_EN	21	rw	**Enable continuous transfer in XIP mode** If this bit is set to 1 then continuous transfer mode in XIP will be enabled, in this mode DWC_ssi will keep slave selected until a non-XIP transfer is detected on the AHB interface.
RSVD_SPI_CTRLR0_22_23	23:22	r	**Reserved bits - read as zero**
SPI_DM_EN	24	rw	**SPI data mask enable bit** When this bit is enabled, the txd_dm signal is used to mask the data on the txd data line.

(table continues...)

(continued)

Field	Bits	Type	Description
SPI_RXDS_SIG_EN	25	rw	**Enable rxds signaling during address and command phase of Hypebus transfer** This bit enables RXDS signaling by Hyperbus slave devices during Command-Address (CA) phase. If the rxds signal is set to 1 during the CA phase of transfer, DWC_ssi transmits (2*SPI_CTRLR0.WAIT_CYCLES-1) wait cycles after the address phase is complete.
XIP_MBL	27:26	rw	**XIP Mode bits length** Sets the length of mode bits in XIP mode of operation. These bits are valid only when SPI_CTRLR0.XIP_MD_BIT_EN is set to 1. 00_B **MBL_2**: Mode bits length equal to 2 01_B **MBL_4**: Mode bits length equal to 4 10_B **MBL_8**: Mode bits length equal to 8 11_B **MBL_16**: Mode bits length equal to 16
RSVD_SPI_CTRLR0_28	28	r	**Reserved bits - read as zero**
XIP_PREFETCH_EN	29	r	**Reserved bits - read as zero**
CLK_STRETCH_EN	30	rw	**Enables clock stretching capability in SPI transfers** In case of write, if the FIFO becomes empty DWC_ssi will stretch the clock until FIFO has enough data to continue the transfer. In case of read, if the receive FIFO becomes full DWC_ssi will stop the clock until data has been read from the FIFO.
RSVD_SPI_CTRLR0	31	r	**Reserved bits - read as zero**

18.4.54 DDR_DRIVE_EDGE - Transmit Drive Edge Register

This register is used to control the driving edge in DDR mode. It is not possible to write to this register when the DWC_ssi is enabled (SSIC_EN=1).

DDR_DRIVE_EDGE Offset address: $000002F8_H$

DDR_DRIVE_EDGE - Transmit Drive Edge Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVD_DDR_DRIVE_EDGE															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD_DDR_DRIVE_EDGE								TDE							
r								rw							

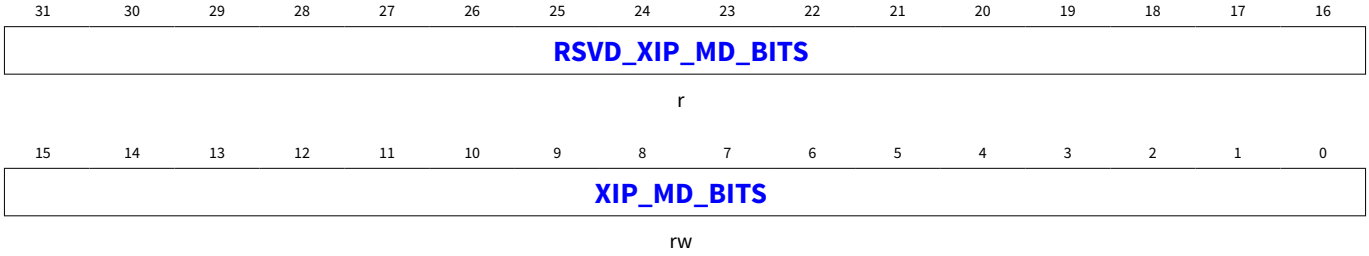
Field	Bits	Type	Description
TDE	7:0	rw	**TXD Drive edge register which decided the driving edge of transmit data** The maximum value of this register is = (BAUDR/2) -1.
RSVD_DDR_DRIVE_EDGE	31:8	r	**Reserved bits - read as zero**

18.4.55 eXecute in Place - Mode bits

This register carries the mode bits which are sent in the XIP mode of operation after address phase. This register cannot be written when xSPI is enabled, i.e, when SSIENR=1.

XIP_MODE_BITS Offset address: 000002FC$_H$

eXecute in Place - Mode bits Kernel Reset value: 0000 0000$_H$



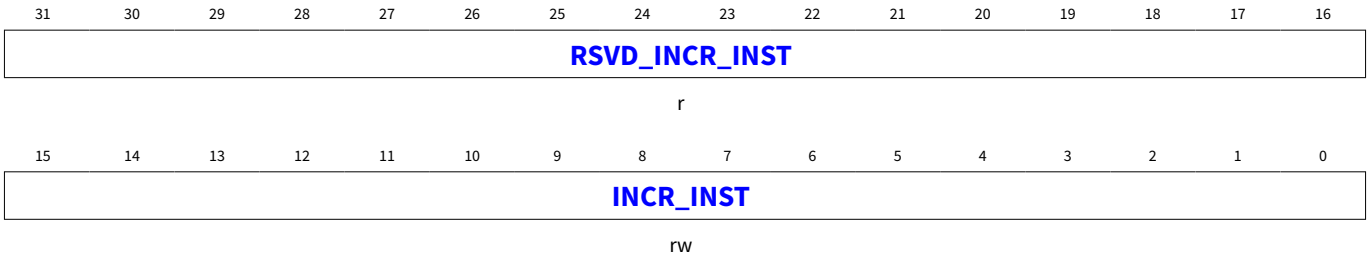
Field	Bits	Type	Description
XIP_MD_BITS	15:0	rw	**XIP mode bits to be sent after address phase of XIP transfer**
RSVD_XIP_MD_BITS	31:16	r	**Reserved bits - read as zero**

18.4.56 XIP_INCR_INST - XIP INCR transfer opcode

This register is used to store the instruction op-code to be used in INCR transactions when the same is requested on the slave interface. It is not possible to write to this register when the DWC_ssi is enabled (SSIC_EN=1).

XIP_INCR_INST Offset address: 00000300$_H$

XIP_INCR_INST - XIP INCR transfer opcode Kernel Reset value: 0000 0001$_H$



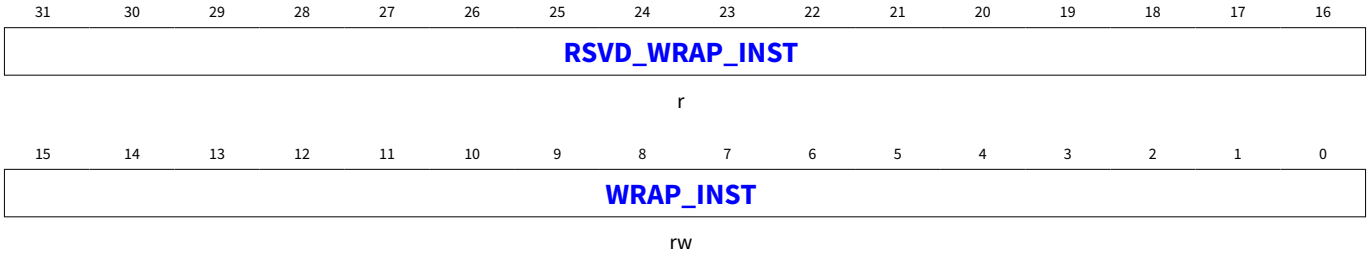
Field	Bits	Type	Description
INCR_INST	15:0	rw	**XIP INCR transfer opcode** When SPI_CTRLR0.XIP_INST_EN bit is set to 1, DWC_ssi sends instruction for all XIP transfers, this register field stores the instruction op-code to be sent when an INCR type transfer is requested on AHB bus. The number of bits to be send in instruction phase is determined by SPI_CTRL0.INST_L field.
RSVD_INCR_INST	31:16	r	**Reserved bits - read as zero**

18.4.57 XIP WRAP transfer opcode

This Register is valid only when SSIC_XIP_EN is equal to 1. This register is used to store the instruction op-code to be used in WRAP transactions when the same is requested on AHB interface. It is not possible to write to this register when the DWC_ssi is enabled (SSIC_EN=1).

XIP_WRAP_INST Offset address: 00000304_H

XIP WRAP transfer opcode Kernel Reset value: $0000\ 0002_H$



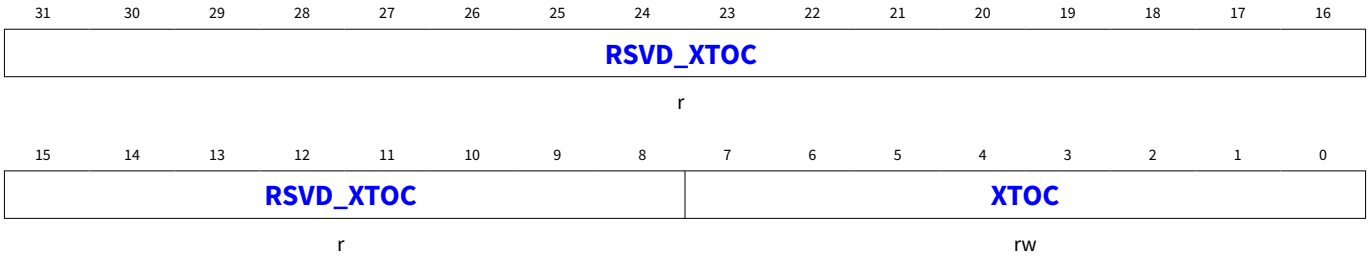
Field	Bits	Type	Description
WRAP_INST	15:0	rw	**XIP WRAP transfer opcode** When SPI_CTRLR0.XIP_INST_EN bit is set to 1, DWC_ssi sends instruction for all XIP transfers, this register field stores the instruction op-code to be sent when an WRAP type transfer is requested at the slave interface. The number of bits to be send in instruction phase is determined by SPI_CTRL0.INST_L field.
RSVD_WRAP_INST	31:16	r	**Reserved bits - read as zero**

18.4.58 XIP time out register for continuous transfers

XIP count down register for continuous mode. The counter is used to de-select the slave during continuous transfer mode. It is not possible to write to this register when the DWC_ssi is enabled (SSIC_EN=1).

XIP_CNT_TIME_OUT Offset address: 00000314_H

XIP time out register for continuous transfers Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
XTOC	7:0	rw	**XIP time out value in terms of Fsri/2** Once slave is selected in continuous XIP mode this counter will be used to de-select the slave, if there is no request for the time specified in the counter.
RSVD_XTOC	31:8	r	**Reserved bits - read as zero**

18.4.59 SPI Device Register

SPIDR Offset address: 00000320_H

SPI Device Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVD_SPIDR															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SPI_INST															
rw															

Field	Bits	Type	Description
SPI_INST	15:0	rw	**SPI Instruction code** This instruction code will be used for SPI operation
RSVD_SPIDR	31:16	r	**Reserved bits - read as zero**

18.4.60 SPI Device Address Register

SPIAR Offset address: 00000324_H

SPI Device Address Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SDAR															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SDAR															
rw															

Field	Bits	Type	Description
SDAR	31:0	rw	**SPI Device Address Register.** This address will be used during read/write on SPI interface for DMA transfer

18.4.61 AXI Address Register 0

AXIAR0 Offset address: 00000328_H

AXI Address Register 0 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AXIAR_0_31															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AXIAR_0_31															
rw															

Field	Bits	Type	Description
AXIAR_0_31	31:0	rw	**LSB for AXI address for DMA operation**

18.4.62 AXI Master Error Interrupt Clear Register

AXIECR Offset address: 00000330_H

AXI Master Error Interrupt Clear Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVD_AXIECR															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD_AXIECR															AXIECR
r															rh

Field	Bits	Type	Description
AXIECR	0	rh	**Clear AXI Error Interrupt** This register will reflect the status of the interrupt. A read from this register clears the ssi_axie_intr interrupt. Writing to this register has no effect.
RSVD_AXIECR	31:1	r	**Reserved bit**

18.4.63 Transfer Done Clear Interrupt Clear Register

DONECR Offset address: 00000334_H

Transfer Done Clear Interrupt Clear Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVD_DONECR															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD_DONECR															DONECR
r															rh

Field	Bits	Type	Description
DONECR	0	rh	**Clear Transfer Done Interrupt** This register will reflect the status of the interrupt. A read from this register clears the ssi_done_intr interrupt. Writing to this register has no effect.
RSVD_DONECR	31:1	r	**Reserved bit**

18.4.64 Safety Control register

Safety Control Register

SAFETYCR Offset address: 00000380_H

Safety Control register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVD_SAFETYCR_27_31					REG_SEL				REGP_EI_EN	DPPG_EI_EN	RSVD_SAFETYCR_20	FSM_SEL		FSM_EI_TYPE	FSM_EI_EN
r					rw				rwh	rwh	r	rw		rw	rwh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD_SAFETYCR_3_16													FSM_PAR_EN	PAR_MODE	RSVD_SAFETYCR_0
r													rw	r	r

Field	Bits	Type	Description
RSVD_SAFETYCR_0	0	r	**Reserved bit - read as zero**
PAR_MODE	1	r	**Parity Mode** This bit selects the even or odd parity scheme for the Parity Protection feature. This covers data path, AXI Master interface and AHB slave interface parity feature. 0_B **EVEN**: Even Parity Check for Parity Protection feature 1_B **ODD**: Odd Parity Check for Parity Protection feature
FSM_PAR_EN	2	rw	**FSM Parity Enable** This bit enables parity checker for all FSM state variables in DWC_ssi. 0_B **DISABLE**: Disable FSM Parity Protection 1_B **ENABLE**: Enable FSM Parity Protection
RSVD_SAFETYCR_3_16	15:3	r	**Reserved bits - read as zero**

(table continues...)

(continued)

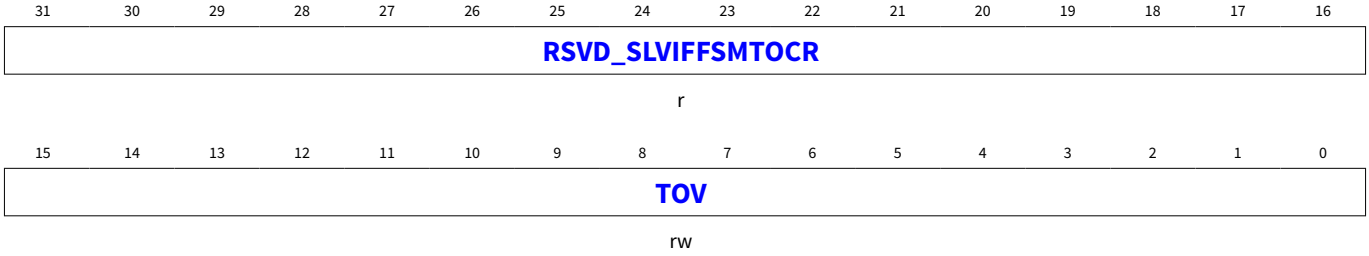
Field	Bits	Type	Description
FSM_EI_EN	16	rwh	**FSM Error Insertion Enable** This bit enables error injection in FSM selected using FSM_SEL programming field. Once this bit is enabled DWC_ssi injects error in DWC_ssi FSM safety mechanism. This bit will be cleared by hardware once the error interrupt is generated. 0_B **DISABLE**: Disable FSM Error Insertion 1_B **ENABLE**: Enable FSM Error Insertion
FSM_EI_TYPE	17	rw	**FSM Error Insertion Type** This bit selects the type of FSM safety feature should be used for error injection. 0_B **TIMEOUT**: FSM Timeout Error Insertion 1_B **PARITY**: FSM Parity Error Insertion
FSM_SEL	19:18	rw	**Select FSM in which the error injection should be enabled** 00_B **SSIM**: DWC_ssi Master FSM 01_B **SLVIF**: Slave Interface FSM 10_B **AXIM**: AXI Master FSM 11_B **DMA**: DMA FSM
RSVD_SAFETYCR_20	20	r	**Reserved bits - read as zero**
DPPG_EI_EN	21	rwh	**Data path parity Error Insertion Enable** Insert data path parity error in 1st parity generator. This bit will be cleared by hardware once the error interrupt is generated. 0_B **DISABLE**: Disable Data Path Parity Error Insertion 1_B **ENABLE**: Enable Data Path Parity Error Insertion
REGP_EI_EN	22	rwh	**Register space protection Error insertion Enable** Insert error in register space protection logic. The error will be injected in the register copy of the selected register using REG_SEL field. This bit will be cleared by hardware once the error interrupt is generated. 0_B **DISABLE**: Disable Register space Error Insertion 1_B **ENABLE**: Enable Register space Error Insertion
REG_SEL	26:23	rw	**Register select for error injection** This bit will select the register in which the error injection should be done. 0_H **CTRLR0**: Control 0 Register 1_H **CTRLR1**: Control 1 Register 2_H **SER**: Slave Enable Register 3_H **BAUDR**: Baud Rate Register 4_H **DMACR**: DMA Control Register 5_H **RX_SAMPLE_DLY**: RX Sample Delay Register 6_H **SPI_CTRLR0**: SPI Control Register 9_H **SAFETYCR**: Safety Control Register **others,** Reserved
RSVD_SAFETYCR_27_31	31:27	r	**Reserved bits - read as zero**

18.4.65 Slave interface Clock domain FSM time out register

This register contains control bits to enable AHB slave clock domain timeout features. The AHB clock domain runs on $F_{SRI/2}$. Please refer to section FSM protection for more details

SLVIFFSMTOCR Offset address: 00000384_H

Slave interface Clock domain FSM time out register Kernel Reset value: $0000\ 0000_H$



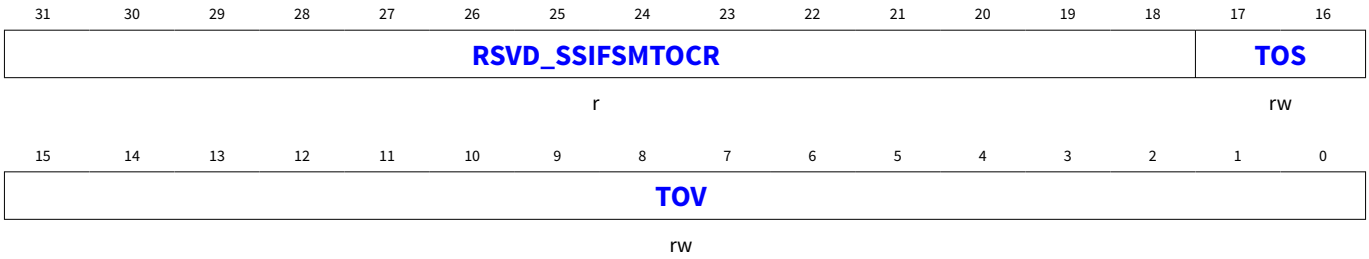
Field	Bits	Type	Description
TOV	15:0	rw	**Time Out Value** This bit sets the time out value for AHB slave clock domain FSM. If FSM state does not reach an IDLE state in given amount of AHB clock cycles, the corresponding error interrupt is asserted.
RSVD_SLVIFFSMTOCR	31:16	r	**Reserved bits - read as zero**

18.4.66 xSPI kernel Clock domain FSM time out register

This register controls the core clock domain FSM timeout features. The SSI clock domain runs on SSI_clk. Please refer to section FSM protection for more details

SSIFSMTOCR Offset address: 00000388_H

xSPI kernel Clock domain FSM time out register Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
TOV	15:0	rw	**Time Out Value** This bit sets the time out value for DWC_ssi core clock domain FSMs. If FSM state does not reach an IDLE state in the given time period, the corresponding error interrupt is asserted.

(table continues...)

(continued)

Field	Bits	Type	Description
TOS	17:16	rw	**Time out scale** This bit is used to scale the time-out value for SSI clock domain. 00_B **DFLT**: Each count tick happens after BAUDR*DFS ssi_clk cycles 01_B **MUL2**: Each count tick happens after 2*BAUDR*DFS ssi_clk cycles 10_B **MUL4**: Each count tick happens after 4*BAUDR*DFS ssi_clk cycles 11_B **MUL8**: Each count tick happens after 8*BAUDR*DFS ssi_clk cycles
RSVD_SSIFSMTOCR	31:18	r	**Reserved bits - read as zero**

18.4.67 AXI Clock domain FSM time out register

This register contains control bits to enable automotive safety features in xSPI. The AXI clock domain runs on F_{SRI}. Please refer to section FSM protection for more details

AXIFSMTOCR Offset address: $0000038C_H$

AXI Clock domain FSM time out register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVD_AXIFSMTOCR															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TOV															
rw															

Field	Bits	Type	Description
TOV	15:0	rw	**Time Out Value** This bit sets the time out value for AXI clock domain FSMs. If FSM state does not reach an IDLE state in given amount of AXI clock cycles, the corresponding error interrupt is asserted.
RSVD_AXIFSMTOCR	31:16	r	**Reserved bits - read as zero**

18.4.68 Safety Interrupt Status Register

SAFETYISR Offset address: $000003A0_H$

Safety Interrupt Status Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVD_13_32_SAFETYISR															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD_13_32_SAFETYISR			AXIUES	AXICES	AXIVRPES	SLVIFAPES	SLVIFDPES	REGERRS	DPPES	AXIPES	RSVD_4_SAFETYISR	SSIFSMTOS	SLVIFSMTOS	AXIFSMTOS	FSMPES
r			rh	rh	rh	rh	rh	rh	rh	rh	r	rh	rh	rh	rh

Field	Bits	Type	Description
FSMPES	0	rh	**FSM Parity Error status** This register bit is set when FSM parity failure occurs.
AXIFSMTOS	1	rh	**AXI clock domain FSM time out error status** This register bit is set when AXI clock domain FSM time out occurs.
SLVIFSMTOS	2	rh	**AHB Slave Interface clock domain FSM time out error status** This register bit is set when AHB Slave Interface clock domain FSM time out occurs.
SSIFSMTOS	3	rh	**SSI Core clock domain FSM time out error status** This register bit is set when SSI Core clock domain FSM time out occurs.
RSVD_4_SAFETYISR	4	r	**Reserved bit - Read as zero**
AXIPES	5	rh	**AXI parity failure interrupt status**
DPPES	6	rh	**Data path Parity checker error**
REGERRS	7	rh	**Register space error status** This bit will be asserted when error is detected in register space protection logic.
SLVIFDPES	8	rh	**AHB write data parity failure interrupt status**
SLVIFAPES	9	rh	**AHB address parity failure interrupt status**
AXIVRPES	10	rh	**AXI Valid/Ready Parity Error status register** This bit is set when AXI Interface Valid/Ready parity failure is detected.
AXICES	11	rh	**AXI Master Interface ECC correctable error status register**
AXIUES	12	rh	**AXI Master Interface ECC uncorrectable error status register**
RSVD_13_32_SAFETYISR	31:13	r	**RSVD2_SAFETYISR Reserved bits - Read as zero**

18.4.69 Safety Interrupt Clear Register

SAFETYICR

Safety Interrupt Clear Register

Offset address: $000003A4_H$

Kernel Reset value: $0000\ 0000_H$

18 eXpanded Serial Peripheral Interface (xSPI)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVD_13_32_SAFETYICR															
w															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD_13_32_SAFETYICR			AXIUEC	AXICEC	AXIVRPEC	SLVIFAPEC	SLVIFDPEC	REGERRC	DPPEC	AXIPEC	RSVD_4_SAFETYICR	SSIFSMTOC	SLVIFSMTOC	AXIFSMTOC	FSMPEC
w			w	w	w	w	w	w	w	w	w	w	w	w	w

Field	Bits	Type	Description
FSMPEC	0	w	**FSM Parity Error status clear register**
AXIFSMTOC	1	w	**AXI clock domain FSM time out error clear register**
SLVIFSMTOC	2	w	**AHB Slave Interface clock domain FSM time out error clear register**
SSIFSMTOC	3	w	**SSI Core clock domain FSM time out error clear register**
RSVD_4_SAFETYICR	4	w	**Reserved bit**
AXIPEC	5	w	**AXI parity failure interrupt clear register**
DPPEC	6	w	**Data path Parity checker error clear register**
REGERRC	7	w	**Register space error clear register**
SLVIFDPEC	8	w	**AHB write data parity failure interrupt clear**
SLVIFAPEC	9	w	**AHB address parity failure interrupt clear register**
AXIVRPEC	10	w	**AXI Valid/Ready Parity Error status clear register**
AXICEC	11	w	**AXI Master Interface ECC correctable error status register**
AXIUEC	12	w	**AXI Master Interface ECC uncorrectable error status clear register**
RSVD_13_32_SAFETYICR	31:13	w	**Reserved bits - read as zero**

18.5 Debug information

This section is not applicable for the module being described.

18.6 References

1. JEDEC Expanded Serial Peripheral Interface (xSPI) for Non Volatile Memory Devices, JESD251C, May 2022

18.7 xSPI revision history

Reference	Description of change(s)
Date range: 2024-08-17 to 2025-01-31	
Data transfers	• Added note related to data frames with DFS greater than 8-bit
Registers	• Corrected bit-field access type to rh or rwh for all hardware writable register bit-fields
Safety Control register	• Removed enumerations XIP_CTRL and XIP_SER from REG_SEL bit-field description for error injection as these registers do not exist
SPI_CTRLR0 - SPI Control Register	• Set the description of XIP_PREFETCH_EN bit-field as this feature is not available

18.8 TC4Dx xSPI information

18.8.1 TC4Dx xSPI configuration

There are no deviations from the generic specification.

18.8.2 TC4Dx xSPI features

There are no deviations from the generic specification.

18.8.3 TC4Dx xSPI functional description

The TC4D device does not support XIP write feature.

Functional restrictions for back-to-back transfers

When XSPI does back-to-back transfers without disabling the IP (by programming SSIENR register to 0). Serial FSM timeout counter does not reset to value of 0 between the two transfers. Serial FSM timeout counter resumes from the previous timeout value. Due to this a spurious FSM timeout interrupt, which left unmasked, a spurious XSPIFSM alarm may occur for the serial FSM timeout if the IDLE period across multiple transfers combined goes beyond the programmed timeout period.

Consequences:

- Serial FSM timeout behavior during IDLE will not reset to 0
- System will recover itself from this interrupt by following the standard IP disable and enable procedure
- No other functional impact expected due to this behavior

Workaround:

- Program the Serial FSM controller timeout value to be more than the duration of back-to-back transfers without disabling the xSPI
- Disable the xSPI in between two transfers
- Regular software flow to disable the IP can be performed whenever the user sees an interrupt and resume the transfers

18.8.4 TC4Dx xSPI registers

There are no deviations from the generic specification.

18.8.4.1 Memory overview tables of XSPI domain SRI

Table 823 Memory overview - XSPI domain SRI (ascending address)

Short name	Long name	Address
XSPI0__CXIP	Memory mapped area (08000000_H Byte)	98000000_H
XSPI0__NCXIP	Memory mapped area (08000000_H Byte)	$B8000000_H$

18.8.4.2 Register address space - XSPI

Table 824 Registers address space - XSPI

Key: The module name in brackets () indicates a memory section name. Without brackets the reference is to a functional block.

Module	Base address	End address	Note
(XSPI0)	98000000_H	$9FFFFFFF_H$	Slave interface for XIP cached address range
	$B8000000_H$	$BFFFFFFF_H$	Slave interface for XIP non-cached address range
XSPI0	$F9040000_H$	$F904FFFF_H$	Slave interface for SFR registers

18.8.4.3 Register overview - access mode glossary

Table 825 Register overview - access mode glossary

Keyword	Description
E	Access protection using PROT register XSPI0_PROTE .
SE	Access protection using PROT register XSPI0_PROTSE .
APU-P	Protection group consisting of registers XSPI0_ACCEN_WRA , XSPI0_ACCEN_WRB , XSPI0_ACCEN_RDA , XSPI0_ACCEN_RDB , XSPI0_ACCEN_VM , XSPI0_ACCEN_PRS .
P	Access protection using *APU-P* registers.
APU-PRGNx (x=0-7)	Protection group consisting of registers XSPI0_RGNx_ACCEN_WRA , XSPI0_RGNx_ACCEN_WRB , XSPI0_RGNx_ACCEN_RDA , XSPI0_RGNx_ACCEN_RDB , XSPI0_RGNx_ACCEN_VM , XSPI0_RGNx_ACCEN_PRS , XSPI0_RGNx_ACCEN_RGNLA , XSPI0_RGNx_ACCEN_RGNUA .
PRGNx	Access protection using *APU-PRGNx* registers.
BE	Always returns a Bus Error.
U	No access restrictions.
PROT	Access restrictions as defined in the PROT register access rules.
nBE	Indicates that no Bus Error is generated when accessing this address range, even though it is either an access to an undefined address or the access does not follow the given rules.

18.8.4.4 Register overview - XSPI0 domain SRI (ascending offset address)

Table 826 Register overview - XSPI0 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
XSPI0_CLC	Clock Control Register	00000000$_H$	P	P, SV, E	Application Reset	2639
XSPI0_OCS	OCDS Control and Status Register	00000004$_H$	P	SV, P	Debug Reset	2639
XSPI0_ID	Module Identification Register	00000008$_H$	P	BE	PowerOn Reset	2640
XSPI0_RST_CTRLA	Reset Control Register A	0000000C$_H$	P	P, SV, E	Application Reset	2640
XSPI0_RST_CTRLB	Reset Control Register B	00000010$_H$	P	P, SV, E	Application Reset	2641
XSPI0_RST_STAT	Reset Status Register	00000014$_H$	P	BE	Application Reset	2642
XSPI0_PROTE	PROT Register Endinit	00000020$_H$	U	SV, PROT	Application Reset	2642
XSPI0_PROTSE	PROT Register Safe Endinit	00000024$_H$	U	SV, PROT	Application Reset	2644
XSPI0_ACCEN_WRA	Write access enable register A	00000040$_H$	U	SE, SV	Application Reset	2645
XSPI0_ACCEN_WRB	Write access enable register B	00000044$_H$	U	SE, SV	Application Reset	2646
XSPI0_ACCEN_RDA	Read access enable register A	00000048$_H$	U	SE, SV	Application Reset	2646
XSPI0_ACCEN_RDB	Read access enable register B	0000004C$_H$	U	SE, SV	Application Reset	2647
XSPI0_ACCEN_VM	VM access enable register	00000050$_H$	U	SE, SV	Application Reset	2647
XSPI0_ACCEN_PRS	PRS access enable register	00000054$_H$	U	SE, SV	Application Reset	2648
XSPI0_RGNx_ACCEN_WRA (x=0-7)	CHx Write access enable register A	00000060$_H$+x*20$_H$	U	SE, SV	Application Reset	2649
XSPI0_RGNx_ACCEN_WRB (x=0-7)	CHx Write access enable register B	00000064$_H$+x*20$_H$	U	SE, SV	Application Reset	2649
XSPI0_RGNx_ACCEN_RDA (x=0-7)	CHx Read access enable register A	00000068$_H$+x*20$_H$	U	SE, SV	Application Reset	2650
XSPI0_RGNx_ACCEN_RDB (x=0-7)	CHx Read access enable register B	0000006C$_H$+x*20$_H$	U	SE, SV	Application Reset	2650

(table continues...)

Table 826 (continued) Register overview - XSPI0 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
XSPI0_RGNx_ACCEN_VM (x=0-7)	CHx VM access enable register	00000070$_H$+x*20$_H$	U	SE, SV	Application Reset	2651
XSPI0_RGNx_ACCEN_PRS (x=0-7)	CHx PRS access enable register	00000074$_H$+x*20$_H$	U	SE, SV	Application Reset	2651
XSPI0_RGNx_ACCEN_RGNLA (x=0-7)	CHx RGN Lower address	00000078$_H$+x*20$_H$	U	SE, SV	Application Reset	2652
XSPI0_RGNx_ACCEN_RGNUA (x=0-7)	CHx RGN upper address	0000007C$_H$+x*20$_H$	U	SE, SV	Application Reset	2653
XSPI0_INTSTAT	Error handling control	00000160$_H$	P	P	Kernel Reset	2653
XSPI0_INTCLR	Error handling interrupt/ alarm clear	00000164$_H$	P	P	Kernel Reset	2654
XSPI0_INTMSK	Error handling interrupt/ alarm mask	00000168$_H$	P	P	Kernel Reset	2655
XSPI0_CTRLR2	Control Register 2	0000016C$_H$	P	P	Kernel Reset	2657
XSPI0_CTRLR0	Control Register 0	00000200$_H$	P	P	Kernel Reset	2658
XSPI0_CTRLR1	Control Register 1	00000204$_H$	P	P	Kernel Reset	2661
XSPI0_SSIENR	xSPI kernel Enable Register	00000208$_H$	P	P	Kernel Reset	2662
XSPI0_SER	Slave Enable Register	00000210$_H$	P	P	Kernel Reset	2663
XSPI0_BAUDR	Baud Rate Select	00000214$_H$	P	P	Kernel Reset	2663
XSPI0_TXFTLR	Transmit FIFO Threshold Level	00000218$_H$	P	P	Kernel Reset	2664
XSPI0_RXFTLR	Receive FIFO Threshold Level	0000021C$_H$	P	P	Kernel Reset	2665
XSPI0_TXFLR	Transmit FIFO Level Register	00000220$_H$	P	P	Kernel Reset	2665
XSPI0_RXFLR	Receive FIFO Level Register	00000224$_H$	P	P	Kernel Reset	2666
XSPI0_SR	Status Register	00000228$_H$	P	P	Kernel Reset	2666
XSPI0_IMR	Interrupt Mask Register	0000022C$_H$	P	P	Kernel Reset	2668

(table continues...)

Table 826 (continued) Register overview - XSPI0 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
XSPI0_ISR	Interrupt Status Register	0000023 0_H	P	P	Kernel Reset	2669
XSPI0_RISR	Raw Interrupt Status Register	0000023 4_H	P	P	Kernel Reset	2670
XSPI0_TXEICR	Transmit FIFO Error Interrupt Clear Register Register	0000023 8_H	P	P	Kernel Reset	2672
XSPI0_RXOICR	Receive FIFO Overflow Interrupt Clear Register	0000023 C_H	P	P	Kernel Reset	2672
XSPI0_RXUICR	Receive FIFO Underflow Interrupt Clear Register	0000024 0_H	P	P	Kernel Reset	2672
XSPI0_MSTICR	Multi-Master Interrupt Clear Register	0000024 4_H	P	P	Kernel Reset	2673
XSPI0_ICR	Interrupt Clear Register	0000024 8_H	P	P	Kernel Reset	2673
XSPI0_DMACR	DMA Control Register	0000024 C_H	P	P	Kernel Reset	2674
XSPI0_AXIAWLEN	Destination Burst Length	0000025 0_H	P	P	Kernel Reset	2675
XSPI0_AXIARLEN	Source Burst Length	0000025 4_H	P	P	Kernel Reset	2675
XSPI0_SSIC_VERSION_ID	xSPI kernel component version	0000025 C_H	P	P	Kernel Reset	2676
XSPI0_DRi (i=0-35)	xSPI Data Register i	0000026 0_H+i*4	P	P	Kernel Reset	2676
XSPI0_RX_SAMPLE_DELAY	RX Sample Delay Register	000002F 0_H	P	P	Kernel Reset	2677
XSPI0_SPI_CTRLR0	SPI_CTRLR0 - SPI Control Register	000002F 4_H	P	P	Kernel Reset	2677
XSPI0_DDR_DRIVE_EDGE	DDR_DRIVE_EDGE - Transmit Drive Edge Register	000002F 8_H	P	P	Kernel Reset	2680
XSPI0_XIP_MODE_BITS	eXecute in Place - Mode bits	000002F C_H	P	P	Kernel Reset	2681
XSPI0_XIP_INCR_INST	XIP_INCR_INST - XIP INCR transfer opcode	0000030 0_H	P	P	Kernel Reset	2681
XSPI0_XIP_WRAP_INST	XIP WRAP transfer opcode	0000030 4_H	P	P	Kernel Reset	2682
XSPI0_XIP_CNT_TIME_OUT	XIP time out register for continuous transfers	0000031 4_H	P	P	Kernel Reset	2682
XSPI0_SPIDR	SPI Device Register	0000032 0_H	P	P	Kernel Reset	2683

(table continues...)

Table 826 (continued) Register overview - XSPI0 domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
XSPI0_SPIAR	SPI Device Address Register	00000324H	P	P	Kernel Reset	2683
XSPI0_AXIAR0	AXI Address Register 0	00000328H	P	P	Kernel Reset	2683
XSPI0_AXIECR	AXI Master Error Interrupt Clear Register	00000330H	P	P	Kernel Reset	2684
XSPI0_DONECR	Transfer Done Clear Interrupt Clear Register	00000334H	P	P	Kernel Reset	2684
XSPI0_SAFETYCR	Safety Control register	00000380H	P	P	Kernel Reset	2685
XSPI0_SLVIFFSMTOCR	Slave interface Clock domain FSM time out register	00000384H	P	P	Kernel Reset	2687
XSPI0_SSIFSMTOCR	xSPI kernel Clock domain FSM time out register	00000388H	P	P	Kernel Reset	2687
XSPI0_AXIFSMTOCR	AXI Clock domain FSM time out register	0000038CH	P	P	Kernel Reset	2688
XSPI0_SAFETYISR	Safety Interrupt Status Register	000003A0H	P	P	Kernel Reset	2688
XSPI0_SAFETYICR	Safety Interrupt Clear Register	000003A4H	P	P	Kernel Reset	2689

18.8.4.5 Device specific registers

There are no device specific register changes.

18.8.5 TC4Dx xSPI connectivity

The table below lists interfaces to and from this functional block to other blocks in the device.

Table 827 List of xSPI interface signals

Interface signals	I/O	Description
HSPHY_XSPI_fxSPI*2	In	Peripheral clock
SRI_XSPI_SIF	In	Slave interface
XSPI_SRI_MIF	Out	Master interface
XSPI_IR_xSPIFIFO	Out	xSPI FIFO interrupts
XSPI_IR_xSPIERROR	Out	xSPI ERROR intterupts
XSPI_SMU_ALARM[1:0]	Out	xSPI Alarms
XSPI_HSPHY_CS0_N	Out	Chipselect 0
XSPI_HSPHY_CS1_N	Out	Chipselect 1
XSPI_HSPHY_xSPI_TXD[0:7]	Out	xSPI Transmit data
HSPHY_XSPI_xSPI_RXD[0:7]	In	xSPI Receive data
XSPI_HSPHY_DM	Out	Data mask
XSPI_HSPHY_CLK	Out	Clock out (sclk_out)
HSPHY_XSPI_RWDS	In	Read write data strobe
XSPI_PORTS_P20_CS0_N	Out	Chipselect 0
XSPI_PORTS_P20_CS1_N	Out	Chipselect 1
XSPI_PORTS_P20_TXD[0:2]	Out	xSPI Transmit data
PORTS_XSPI_P20_RXD[0:2]	In	xSPI Receive data
XSPI_PORTS_P15_TXD3	Out	xSPI Transmit data
PORTS_XSPI_P15_RXD3	In	xSPI Receive data
XSPI_PORTS_P20_CLK	Out	Clock out (sclk_out)
CLOCK_XSPI_fxSPISL	In	Peripheral clock option for extended slow speed functionality

18.8.6 TC4Dx xSPI revision history

Reference	Description of change(s)
Date range: 2024-08-17 to 2024-09-30	
TC4Dx xSPI connectivity	• Corrected the direction of RXD signal as input for port20/15

19 Peripheral Component Interconnect Express (PCIe)

The PCIe is a high-speed communication interface. It is a full-duplex connection capable of sending and receiving data at the same time. The interface is compliant with PCIe Gen 5 and the data rate is up to 8 GT/s.

19.1 Feature list

- Compliant with the PCI Express™ specification, revision 5.0 Version 1.0
- Gen3 data rate, x1 lane up to 8 GT/s (giga-transfers per second)
- Dual mode: either root complex or endpoint
- Programmable Base Address Register (BAR) addressing: 32 bit or 64 bit
- Programmable BAR sizing
- 256 byte maximum payload size (MAX_PAYLOAD_SIZE)
- Requester and completer function support for PCIe transactions
- Interrupt generation and reception
  - INTx support
  - MSI support, up to 32 interrupts
  - MSI-X support, up to 64 interrupts
  - Additional support for non-standard interrupt reception in endpoint mode
- Power management
  - D0, $D3_{hot}$ and $D3_{cold}$ support
  - L0, L0s, L1, L1 sub-states, L2, and L3 support
  - Reference clock power management with CLKREQ#
  - Active-State Power Management (ASPM) support
  - PCIe beacon and wake-up (WAKE#)
- Embedded Direct Memory Access (DMA) support
  - 4 read and 4 write DMA channels
  - Linked list support
  - Remote configurable DMA channels (EP only)
- Address translation unit
  - 8 translation regions for inbound transactions
  - 8 translation regions for outbound transactions
  - Transaction type conversion
- Baseline and Advanced Error reporting (AER) support
- One physical function implemented in endpoint mode
- Optional end-to-end cyclic redundancy (ECRC) support
- Accesses to the System-on-Chip (SoC) memory and registers from wire
- Spread Spectrum Clocking (SSC) support
- Optimized Buffer Flush/Fill (OBFF) support
- Precision Time Measurement (PTM) support
- Latency Tolerance Reporting (LTR) support
- Readiness notification support
- Advanced on-chip debug support (OCDS) trigger bus (OTGB) trace support
- Safety measures for registers, memory, and data path against random hardware faults

19.2 Functional overview

PCI Express is a high performance, general purpose I/O interconnect for a wide range of applications. The interface is compatible with the PCI Express™ Base Specification, Revision 3.1a. The target of this manual is not to document the PCIe protocol. This manual describes mainly how to use the PCIe interface and only basic concept of the PCIe protocol.

The PCIe interface can be configured as a root complex or an endpoint after the power-on. Afterwards, the work mode cannot be changed without resetting the PCIe interface. The speed of the PCIe interface can be Gen1, Gen2 and Gen3. An embedded DMA is designed to offload the CPU by using DMA transfers. In addition, an address translation unit provides flexible mapping between the PCIe wire address and the SoC internal address.

The PCIe interface consists of two sub-blocks, namely the PCIe controller and the HSPHY (PCIe PHY), as shown in the following figure. An SRI master interface and an SRI slave interface are used for data transfer while two SPB slave interfaces are only for the configuration purpose. The interface connects to interrupt router (IR), safety and security alarm management unit (SMU) and TRACE for the common infrastructure functionalities.

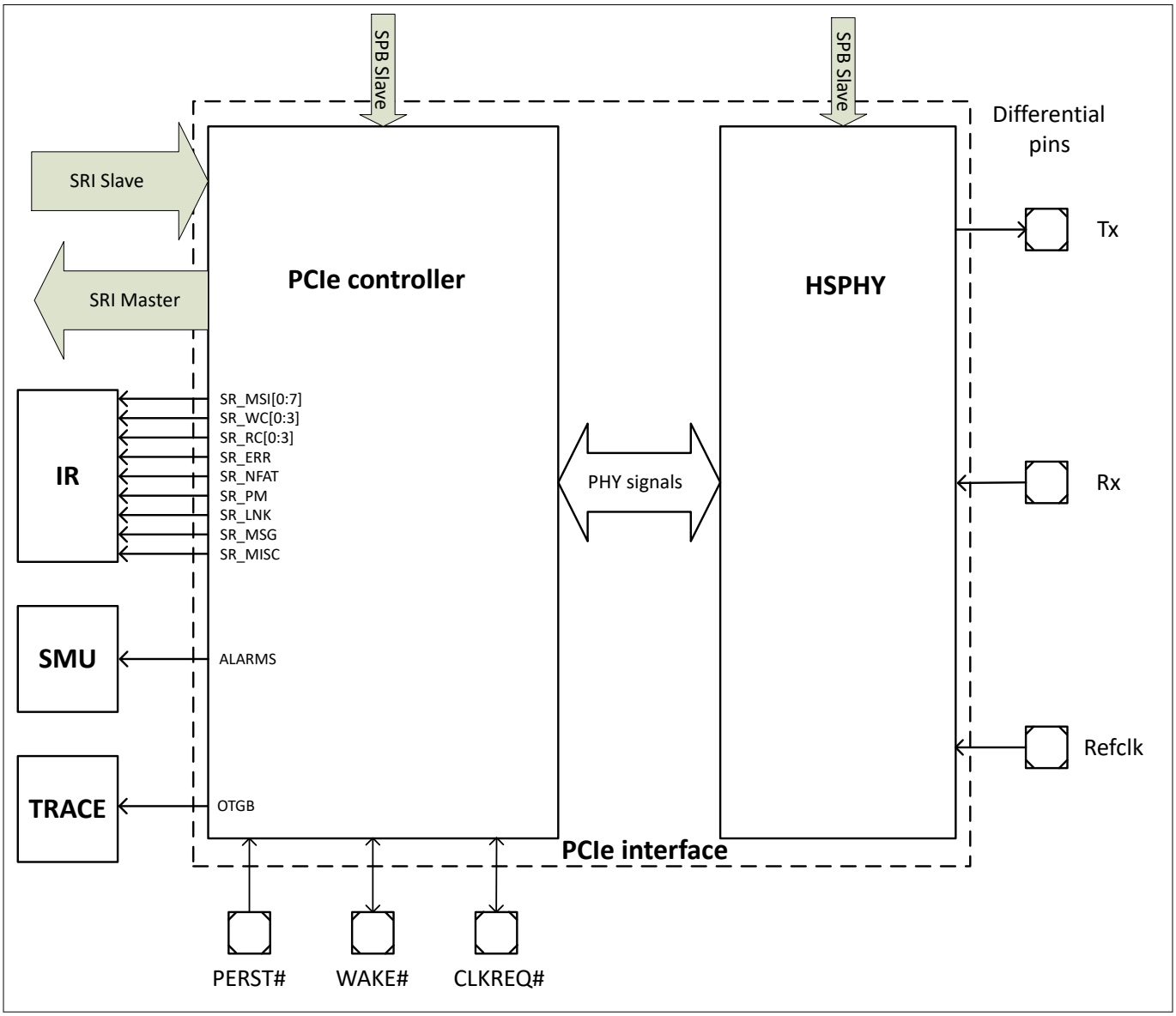


Figure 275 PCIe block diagram

For the alarm description please refer to the alarm mapping tables in the SMU functional block and the Safety Manual.

19 Peripheral Component Interconnect Express (PCIe)

Related information

TC4Dx SMU alarm mapping tables on page 7233

19.3 Functional description

The PCIe module described in this chapter contains both the PCIe controller and the UPCS layer, as introduced in the following figure. The blocks that are related to data stream and configuration stream are depicted. The arrows show how inbound and outbound requests are routed. The posted and the non-posted requests use different data paths.

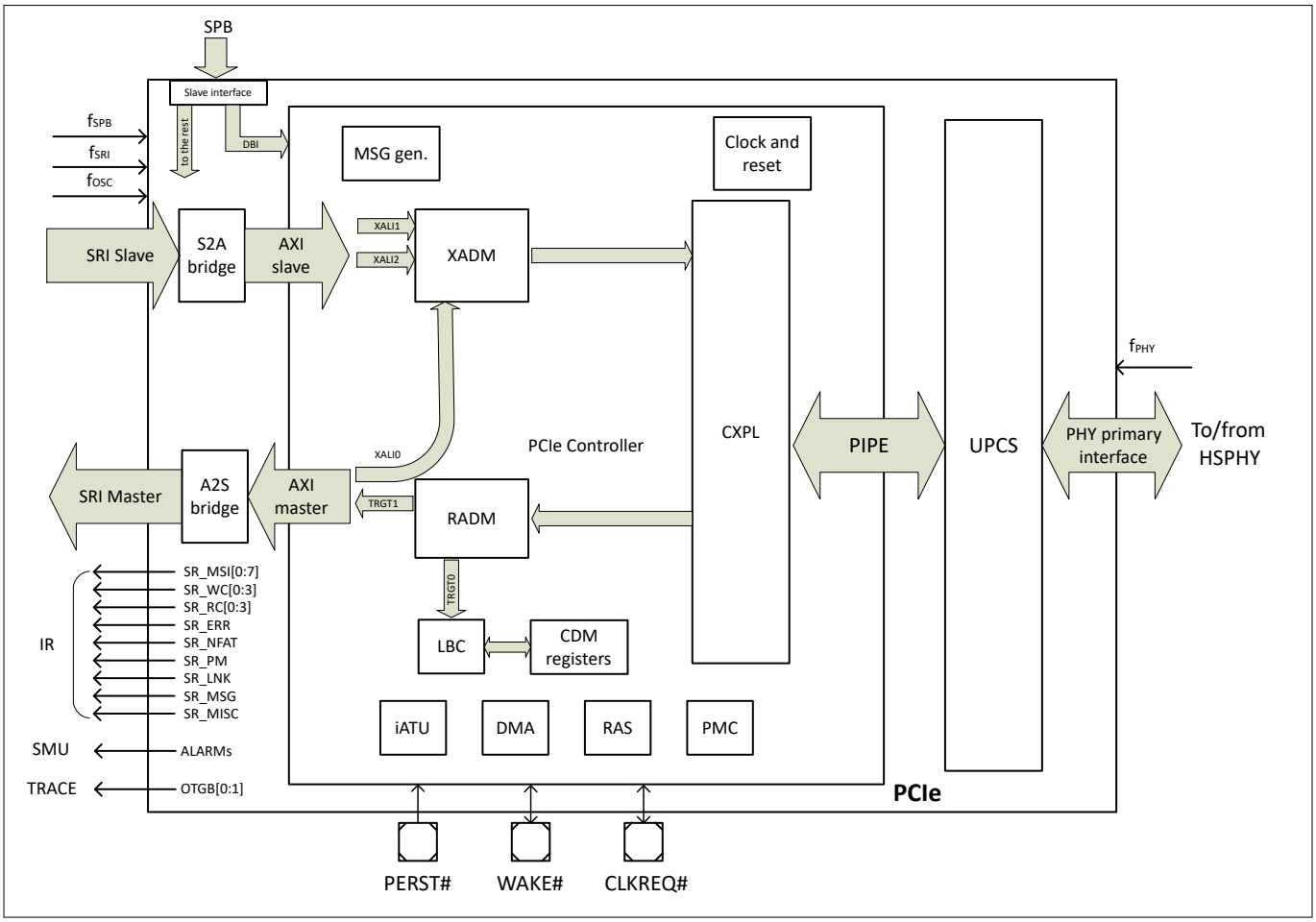


Figure 276 The overview of the PCIe module

Common Express Port Logic (CXPL) Module implements the basic functionality for the PCI Express physical, link, and transaction layers. This module implements a large part of the transaction layer logic, all of the data link layer logic, and the controller portion of the physical layer, including the link training and status state machine (LTSSM). The CXPL connects to the universal physical coding sublayer (UPCS) through the PIPE (The PHY Interface for the PCIe Express, SATA, USB, DisplayPort and Converged IO architectures), a standard PHY interface for PCIe. The UPCS provides many of the functions outlined in the PIPE 4 specification and comprises the PCS as referenced in the PIPE 4 specification.

Transmit Application-Dependent Module (XADM) implements the application-specific functionality of the PCI Express transaction layer for packet transmission. Its functions include:

- TLP Arbitration
- TLP Formation
- Flow Control (FC) Credit checking

The transmit path uses a cut-through architecture. It does not implement transmit buffering/queues (other than the retry buffer). The controller maintains an internal Target Completion Lookup Table to store certain TLP header information from the Rx request.

Receive Application-Dependent Module (RADM) implements application-specific functionality of the PCI Express transaction layer for packet reception. Its functions include:

- Sorting/filtering of received TLPs. The filtering rules and routing are configurable
- Buffering and queuing of the received TLPs
- Routing of received TLP to the controller's receive interfaces

Configuration-Dependent Module (CDM) implements:

- Standard PCI Express configuration space
- Controller-specific register space

Local Bus Controller (LBC) provides a mechanism for a link partner or a local CPU (through the SPB) to access the controller internal registers. The access to the internal registers from wire is only possible when AURIX™ is operating in EP mode.

Message Generation Module (MSG gen.) transmits messages generated by the controller.

Clock and reset block is responsible for distributing clocks to different blocks in the controller and controlling the reset of the controller.

Power management controller (PMC) implements the power management features of the controller.

PCIe system Reliability, Availability, and Serviceability (RAS) ensures that failures in the underlying processes and hardware components do not cause any interruptions in the overall system operation.

Embedded direct memory access (DMA) is designed to offload the transferring of large blocks of data and leave the CPU free to perform other tasks.

The controller uses the internal address translation unit (iATU) to implement a local address translation scheme that replaces the TLP address and TLP header fields in the current TLP request header.

The PCIe controller has the following data interfaces for various traffic.

- Receive:
  - TRGT0: the internal receive interface accesses the CDM registers and some internal registers
  - TRGT1: the internal receive interface access BAR-matched memory requests
- Transmit:
  - XALI0: the internal transmit interface for the completions
  - XALI1: the internal transmit interface for the outbound non-posted requests
  - XALI2: the internal transmit interface for the outbound posted requests

The SRI interfaces are mainly for transferring data. The SRI slave is connected to the AXI slave via the S2A bridge while the SRI master is connected to the AXI master via the A2S bridge. The PCIe controller is accessed via SRI by the CPU. The AXI interface is internal and cannot be accessed directly by software.

The SPB interface is for register configuration purpose. The SPB is connected to the DBI to access the registers located inside of the PCIe kernel. The DBI is transparent from the software point of view.

The SR_* signals are the service request signals to the IR. The details are described in the "Local interrupts" section. The ALARMS is the alarm signal to the SMU to detect errors. The OTGB signals provide diagnostic information to the TRACE for debug.

19.3.1 Link initialization

The initialization process of PCIe follows the sequence defined in the following figure.

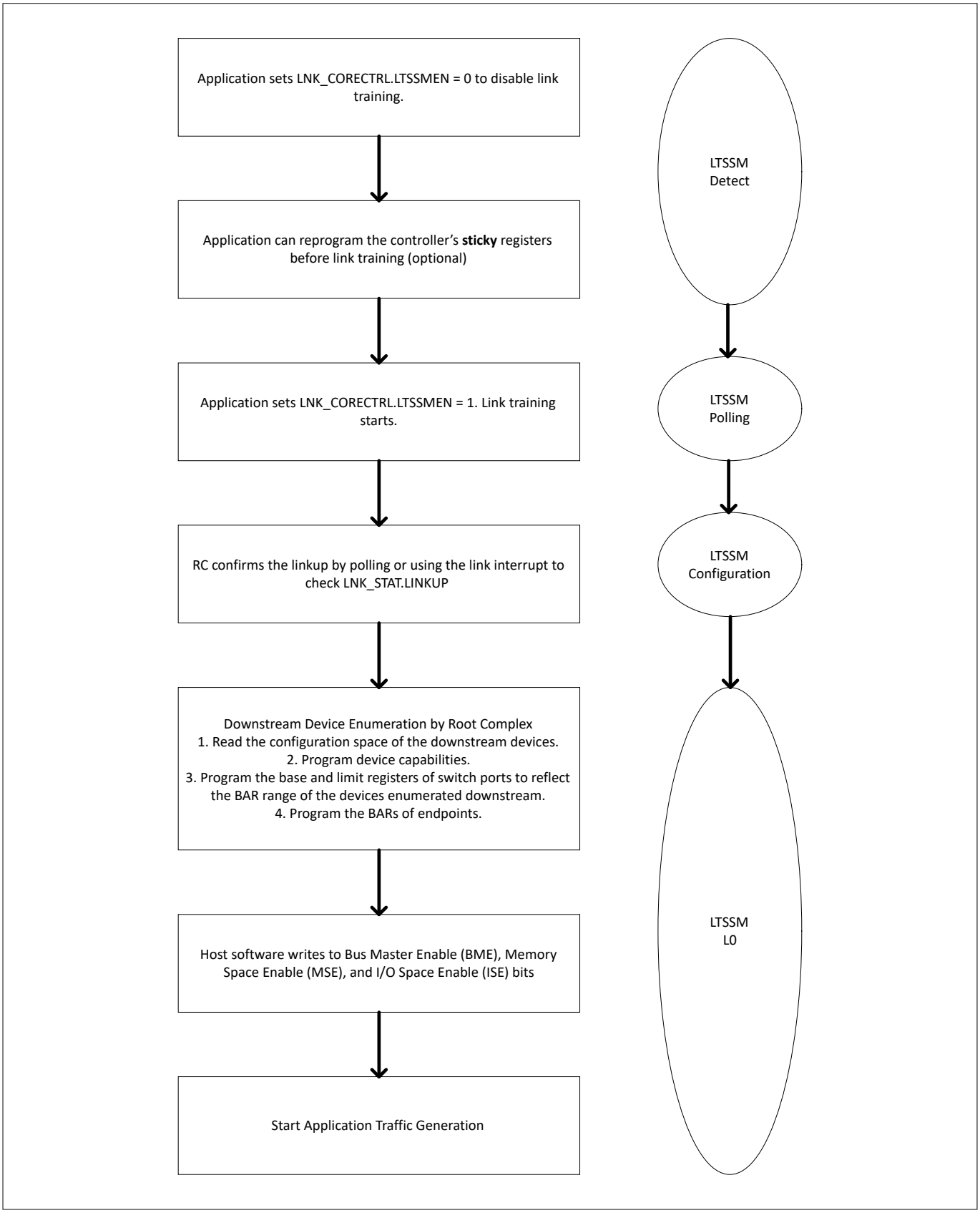


Figure 277 The steps of link establishment

Related information

iATU on page 2752

19.3.1.1 Minimum working example of RC

1. HSPHY: The HSPHY module clock shall be enabled by asserting CLC.DISR. Set the PHY_CTRL1.FSP to 1 to select the PCIe to the corresponding MP8G PHY. Select the reference clock (100 MHz refclk) from the pad by setting the PHY_CTRL1.PRS to $xx1_B$
2. After power-up, the module clock shall be enabled by using CLC.DISR. The device type shall be configured to be root complex by setting LNK_DVCTYP.TYPE = 0100_B. A PCIe module reset is needed then by applying the module reset sequences that will reset the MP8G PHY and PCIe controller together
3. After the PCIe module reset is applied, if a common reference clock is used, the PHY control signals must be overridden according to TC4Dx specific UPCS override settings for common refclk PHY settings
4. HSPHY: After the MP8G PHY reset triggered by the PCIe, there is an option to load the firmware that is executed in the MP8G PHY by the application. The PHY_CTRL1.INITDONE indicates that the MP8G PHY is ready and the application can load the firmware into the SRAM (optional). The application should assert the PHY_CTRL1.EXTLDDONE after the loading or without the loading so the MP8GPHY can continue to initialization process. During the initialization of the MP8G PHY, the clock from the MP8G PHY is not provided to the PCIe controller and no access to the controller registers should be initiated. The end of the initialization phase is indicated by the RAWCMN_DIG_AON_CMNCAL_STATUS register in the HSPHY
5. The link training can be started by asserting the LNK_CORECTRL.LTSSMEN bit. The link state is indicated in the LNK_STAT register. The software can check the current link speed by checking the LINK_CONTROL_LINK_STATUS_REG.PCIE_CAP_LINK_SPEED field, that indicates the negotiated link speed of the given PCIe link
6. The enumeration is performed with Cfg TLPs. To initiate Cfg TLPs with normal bus transfers by CPU, ATU is applied to convert the outbound accesses into Cfg TLPs. The details of how to configure ATU is described in the section iATU. The address registers of an ATU region shall follow the enhanced configuration access mechanism (ECAM) defined in the PCIe specification and be properly configured to target at the right device. The IATU_REGION_CTRL_1_OFF_OUTBOUND.TYPE must be the right TLP type
7. After the configuration of end points, the memory space or the IO space can be enabled. Please note that RC application must not generate any MEM or I/O requests until the host has enabled the PCI_TYPE0_MEM_SPACE_EN and the PCI_TYPE0_IO_EN bits in the PCIe command register (STATUS_COMMAND_REG)
8. Memory TLPs can be initiated to access EPs after the proper configuration of BAR registers in EPs

***Note**: for re-initialising the link, it has to be disabled first using the register LINK_CONTROL_LINK_STATUS_REG.PCIE_CAP_LINK_DISABLE*

19.3.1.2 Minimum working example of EP

1. HSPHY: The HSPHY module clock shall be enabled by asserting CLC.DISR. Set the PHY_CTRL1.FSP to 1 to select the PCIe to the corresponding MP8G PHY. Select the reference clock (100 MHz refclk) from the pad by setting the PHY_CTRL1.PRS to $xx1_B$
2. After power-up, the module clock shall be enabled by asserting CLC.DISR. The device type is set to endpoint by default. A PCIe module reset is needed then by applying the module sequences that will reset the MP8G PHY and PCIe controller together
3. After the PCIe module reset is applied, if a common reference clock is used, the PHY control signals must be overridden according to TC4Dx specific UPCS override settings for common refclk PHY settings
4. HSPHY: After the MP8GPHY reset triggered by the PCIe, there is an option to load the firmware that is executed in the MP8G PHY by the application. The PHY_CTRL1.INITDONE0 indicates that the MP8G PHY is ready and the application can load the firmware into the SRAM (optional). The application should assert

the PHY_CTRL1.EXTLDDONE after the loading or without the loading so the MP8G PHY can continue to initialization process. During the initialization of the MP8G PHY, the clock from the MP8G PHY is not provided to the PCIe controller and no access to the controller registers should be initiated. The end of the initialization phase is indicated by the RAWCMN_DIG_AON_CMNCAL_STATU register in the HSPHY

5. The link training can be started by asserting the LNK_CORECTRL.LTSSMEN bit. The link state is indicated in the LNK_STAT register. The software can check the current link speed by checking the LINK_CONTROL_LINK_STATUS_REG.PCIE_CAP_LINK_SPEED field, that indicates the negotiated link speed of the given PCIe link
6. After the configuration of end points by RC, the memory space or the IO space is supposed to be enabled. Memory TLPs can be initiated to access memory address space

19.3.1.3 Optional IO control

The optional pins including PERST#, WAKE# and CLKREQ# are supported by the PCIe controller. Additional software configuration is necessary to connect the signal to the right pin.

The PERST# is an input to the PCIe controller and controls the reset of the PCIe interface. One alternative pin input can be selected via LNK_CORECTRL.PERST from the possible alternatives. For more details regarding the pin assignment, refer to the corresponding data sheet.

The WAKE# acts as both input and output to the PCIe controller and supports the low power state. One alternative pin input/output can be selected via LNK_CORECTRL.WAKE from possible alternatives. For more details regarding the pin assignment, refer to the corresponding data sheet.

The CLKREQ# acts as both input and output to the PCIe controller and supports the L1 sub-states. One alternative pin input/output can be selected via LNK_CORECTRL.CLKREQ from possible alternatives. For more details regarding the pin assignment, refer to the corresponding data sheet.

19.3.2 Address spaces

PCIe supports four address spaces: configuration, memory, IO and message. Configuration address space allows software to setup and check devices in a standard way. Memory address space is the space in CPU memory address range consumed by devices for non-configuration purposes. IO address space is for data at the IO-mapped location. It is legacy from PCI and not recommended to be used according to the PCIe specification. Message address space covers event signal mechanism to general purpose messaging.

The four address spaces can be accessed from wire, meaning from a remote PCIe device. There are several restrictions regarding address space access defined in the PCIe specification. For example, only RC can access the configuration address space. The selection and switching of different address spaces mainly rely on ATU, more details introduced in the chapter iATU.

All registers in the controller can be accessed from the AURIX™ side. Part of the registers can also be accessed from wire.

Related information

iATU on page 2752

19.3.2.1 Address spaces from wire

The following table explains how memory and configuration accesses from the wire map to the local PCIe module registers.

The protocol registers including PCIe configuration header, standard and extended capability registers are registers specified by the PCIe specification while the implementation-specific registers are registers in this module but not specified by the PCIe specification.

Table 828 Register region access

Group	Register type	Local address offset[1]	Configuration space address offset	Memory space address offset (BAR0)
1	Application registers: registers related to PCIe and interactions with other modules	0_H - $2FFF_H$ (except for the locations belonging to group 2 and 3)	N.A.	N.A.
2	Request trigger registers (INT_REQTRIG and INT_TRIGSTAT)	364_H and 358_H	N.A.	000_H and 004_H
3	Mailbox registers	$3FC_H$ - $41B_H$	N.A.	008_H - 027_H
4	ATU registers	3000_H - $3FFF_H$	N.A.	N.A.
5	Protocol registers: PCIe configuration header, standard and extended capability registers	4000_H - $433F_H$	0_H - $33F_H$	N.A.
6	PCIe implementation specific registers	4700_H - $4B8F_H$	700_H - $B8F_H$	N.A.
7	MSI-X table	$5A00_H$ - $5DFF_H$	N.A.[2]	$A00_H$ - DFF_H
8	PBA table	$5E10_H$ - $5E17_H$	N.A.[2]	$E10_H$ - $E17_H$
9	Shadow registers including BAR mask registers and MSIX registers	6000_H - $6FFF_H$	N.A.	N.A.
10	DMA registers	7000_H - $791F_H$	N.A.	4000_H - $491F_H$[3]
11	Data access control registers	$3FFFE00_H$ - $3FFFF07_H$(SRI base address)	N.A.	N.A.

1) More details in Register Overview - PCIE_USP Domain FPI
2) Can be accessed depending on the value of TARGET_ABOVE_CONFIG_LIMIT_REG and CONFIG_LIMIT_REG. Possible but not recommended.
3) By default, they are not accessible. The size of BAR0 should be increased to at least 20 KB to include the DMA registers.

The register ranges in the table above are not continuous.

According to the specification, the configuration space is up to 4 KB for a function. Depending on the value of TARGET_ABOVE_CONFIG_LIMIT_REG in the MISC_CONTROL_1_OFF, the Cfg TLP with an address over CONFIG_LIMIT_REG can be routed to the TRGT1 or the internal register groups 7 and 8 that are located in the External Local Bus Interface (ELBI). Normally, CONFIG_LIMIT_REG cannot be set to less than 340_H DWORDs (equivalent to $D00_H$ bytes) which is the top of the registers. The routed TLP to TRGT1 or ELBI inherits the address offset of the Cfg TLP. For TRGT1, it uses the address as bus address. For ELBI, it uses it as the offset in the memory space referring to BAR0.

In the MEM/IO address space, the PCIe module registers and even the AURIX™ address space can be accessible, as explained in the following table. In the PCIe module, an embedded DMA is designed to facilitate data transfers between remote and local memory. The configuration registers of the DMA are addressable via BAR0 in the MEM/IO address space. The EP interrupt trigger registers can be also read/written in the same way. BAR1 to BAR5 are designed to access the internal AURIX™ address space.

The columns for the configuration space and the memory space are not applicable for RC.

Table 829 MEM/IO address space as EP

Base address register	Address offset	Access region	Note
BAR0	$A00_H \leq$ offset < DFF_H	MSI-X table (group 7)	The register layout is identical to the one from local address in the previous table.
	$E10_H \leq$ offset < $E17_H$	PBA table (group 8)	
	000_H	INT_REQTRIG (group 2)	
	004_H	INT_TRIGSTAT (group 2)	
	$008_H \leq$ offset < 027_H	mailbox registers (group 3)	
	$4000_H \leq$ offset < $491F_H$ [1)]	DMA registers (group 10)	
BAR1-5	-	AURIX™ internal address space	The addresses can be remapped by the ATU.

1) The size of BAR0 should be at least 20 KB.

19.3.2.2 Local bus accesses

The configuration of PCIe controller from the local bus is introduced in the first part of the following table. Different regions are addressed depending on the offsets. To transmit TLPs in outbound direction, the slave interface of SRI and the iATU are utilized to convert SRI transactions into PCIe TLPs. The address ranges for MEM/IO, message and configuration are configurable in the translation regions of iATU, that has eight outbound and eight inbound translation regions respectively. More details regarding addressing at address spaces can be found in the chapter iATU.

Table 830 Address map from local bus including the memory and the registers

Type	Base address	Groups	Address offset
Configuration of PCIe controller	PCIe SPB base address	1 - 10	0_H - $791F_H$
Outbound window to generate TLPs: Memory read, I/O, configuration and message	PCIe SRI base address	N.A.	0_H - $3FFFDFF_H$ (64 MB address window can be shared . The splitting is dependent on the APU and the ATU)
Configuration of data access control registers	PCIe SRI base address	11	$3FFFE00_H$ - $3FFFF07_H$

Access protection unit

The above accesses are subject to the restrictions of Access Protection Unit (APU), as shown in the below figure.

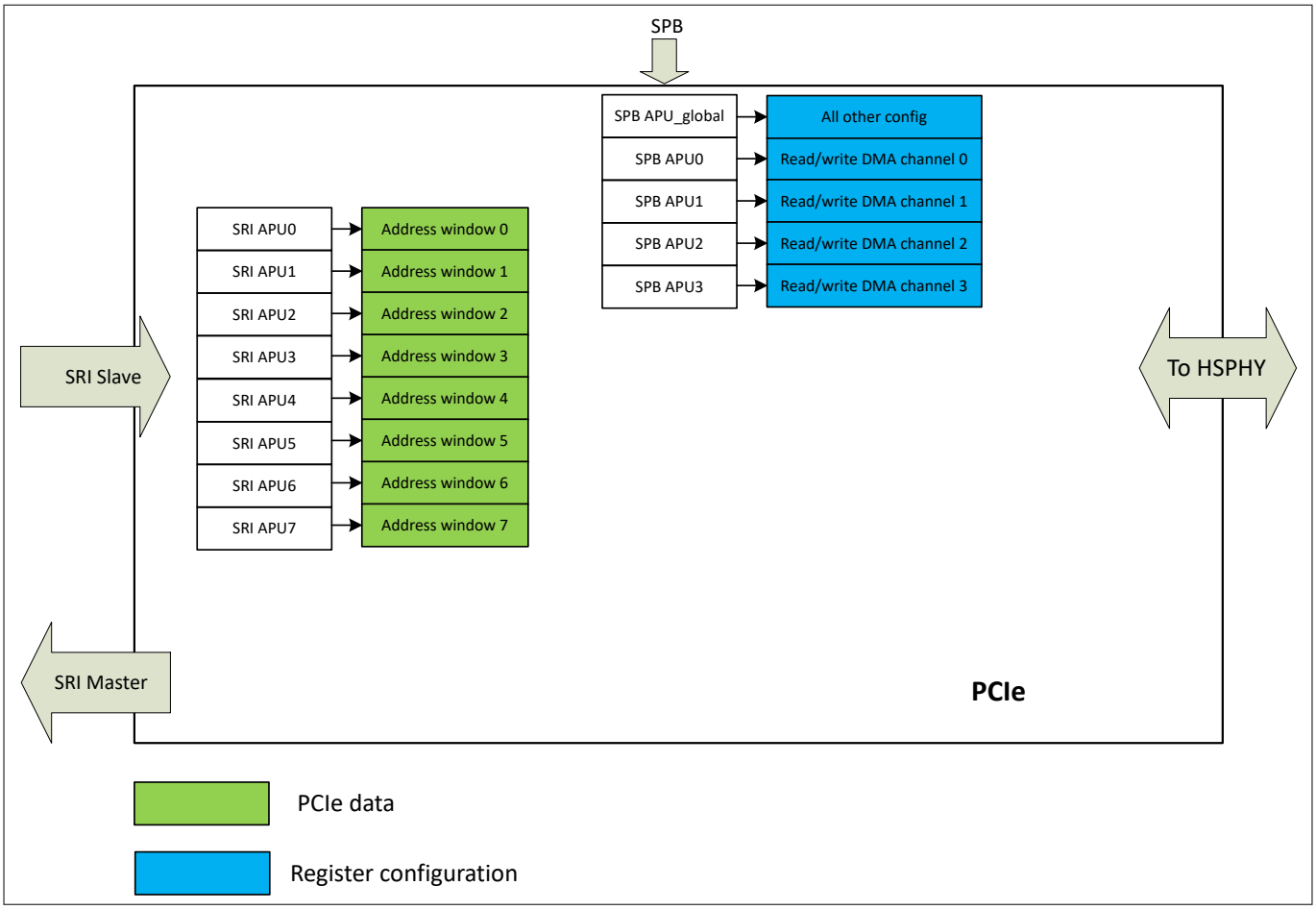


Figure 278 Access protection to the PCIe controller

For the configuration of PCIe controller, one global APU instance (APU_global) controls which bus masters are allowed to configure the PCIe controller. Four additional APUs (DMA APU0-3) protect the access to the configuration registers in the four DMA channels (including both read and write channels with the same DMA index) respectively. The configuration registers specific to DMA channels are protected by these APUs for example transfer size, doorbell and addresses. The common DMA configurations are covered by the global APU. In this way, specific virtual machines can use the assigned DMA channels without interfering each other. The details of which registers are protected by which specific APU are described in the register chapter.

To allow both dedicated control of a specific DMA channel and global control of all DMA channels, several functionality control registers appear twice, once in the common part and once in the channel specific part. For example, a DMA transfer from read channel 0 can be started by using the DMA_READ_DOORBELL_OFF that starts all read channels. It can also be started by using the DMA_CHN0_RD_DOOR that triggers only the DMA read channel 0. The DMA_READ_DOORBELL_OFF is covered by the global APU while the DMA_CHN0_RD_DOOR is protected by the DMA APU0.

The SRI bus transactions initiated by the DMA channels may have additional tag information to indicate which DMA channel is the initiator. This additional information is reflected in the VM or the PRS in an SRI transaction. The selection of the VM or the PRS is controlled by the CFG_INFO.SRITAG. Depending on the enumeration of the SRITAG, the channel ID is assigned to the corresponding VM and PRS tag. This additional tag information can help to have a better granularity of controlling the DMA accesses to the other modules.

For the region related to SRI data transactions (the second row of the previous table), eight APU instances (APU0-7) are available and each one defines an address window for data accesses. When the address of a local access falls into the address window of an APU, then this access is by this APU configuration. The APUaddress windows shall be 64 byte aligned. Every APU has independent configurations to permit accesses fromspecific b us masters. For example, each address window can be 10 MB and each window is only allowed to beaccessed b y a VMx on CPU0. More details about APU are available in the SRI chapter.

When iATU is not applied, the bus transactions in AURIX™ to the SRI slave interface of PCIe will create memory TLP for example MRd or MWr and access the memory space of remote devices.

Related information

iATU on page 2752

19.3.3 Transmit/receive TLP

This section describes how TLPs are transmitted and received. The routing and the arbitration are also introduced in this section. These functionalities are performed by the XADM and the RADM.

19.3.3.1 Transmit TLP

Transmit TLP

TLPs and DLLPs have equal priorities during transmit arbitration as shown in the following figure. The priority of TLP types and DLLP types are shown in the following two tables, respectively.

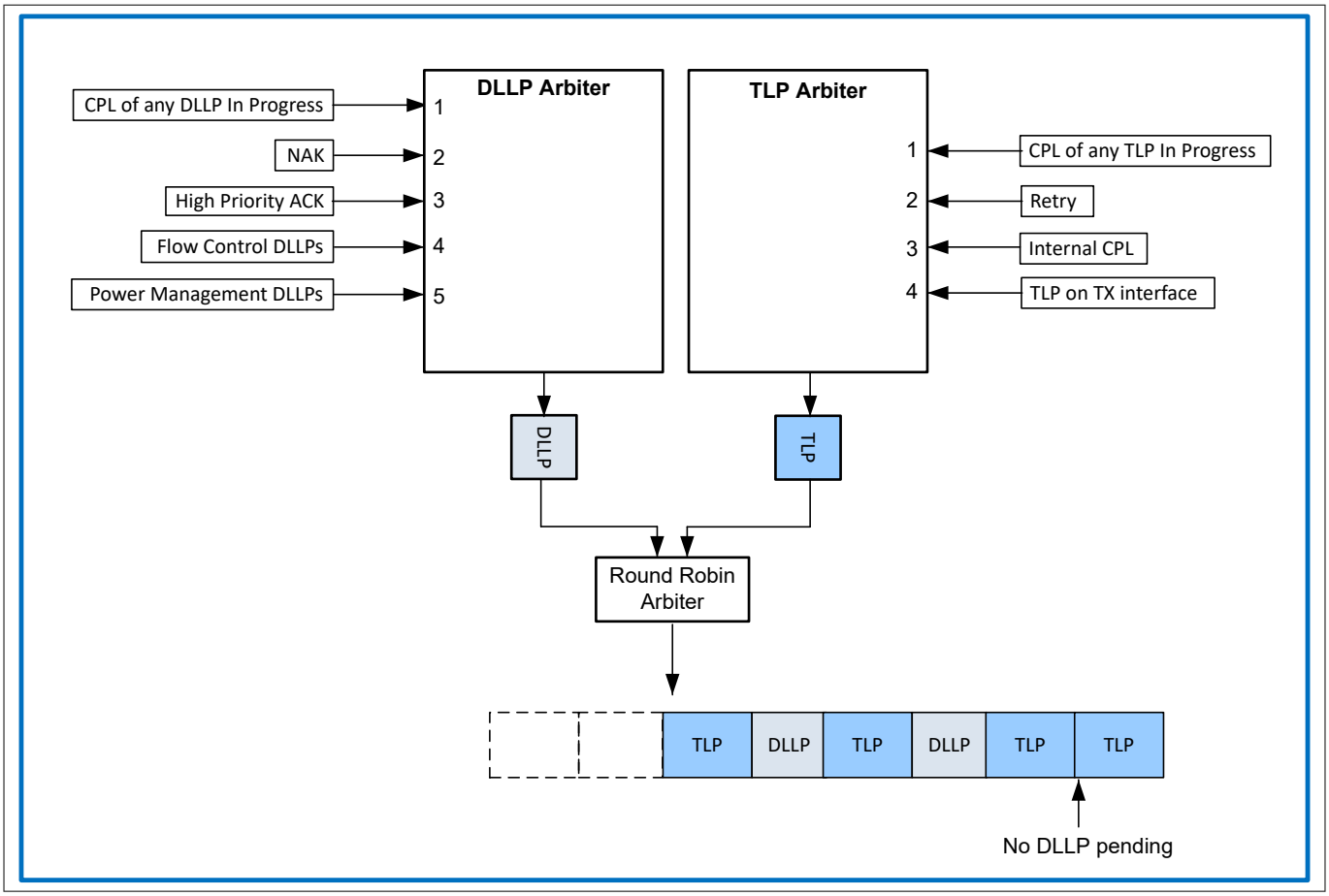


Figure 279 Transmit TLP/DLLP arbitration

Table 831 TLP arbitration priority (1 = highest, 4 = lowest)

Priority	TLP type
1	Completion (Cpl) of any TLP currently in progress
2	Retry buffer retransmissions

(table continues...)

Table 831 (continued) TLP arbitration priority (1 = highest, 4 = lowest)

Priority	TLP type
3	TLPs from the transaction layer (in the following order of priority): • Messages generated by the controller • Upstream ports: Completions generated by the controller (including memory or I/O mapped application register space) for Type 0 configuration read and write requests, or responses to error conditions (unsupported requests) Downstream ports: Completions generated by the controller for unsupported requests or completer aborts
4	TLPs on the bus interface

Table 832 DLLP arbitration priority (1= highest, 5 = lowest)

Priority	DLLP type
1	Completion of any DLLP currently in progress
2	NAK DLLP
3	High Priority ACK DLLP
4	Flow Control DLLP
5	Power management or any other Low Priority DLLP

Selecting transmit client arbitration scheme

All XALI transmit interfaces (XALI0, XALI1 and XALI2) are using round robin arbitration, when credit is available, regardless of the type of transaction. The XALI0 is responsible transmitting completions TLPs. The XALI1 transmits NP requests while the XALI2 handles P requests and messages.

Effects of flow control credits on transmit client arbitration

The controller checks that enough flow control (FC) credits are available in the remote device for the specific type of transaction (posted, non-posted, completion) before allowing a transmission of a TLP. TLPs that passed the credit check are arbitrated according to the supported arbitration method. Internally generated completions and messages are also gated by the arbitration logic, though at highest priority, and must also pass the FC credit test before they are accepted for transmission.

ACK/NAK scheduling

The following figure shows how you can reprogram the controller to influence the sending of ACK and NAK DLLPs. The registers that can be used for the same are Lane Skew Register.ACK_NAK_DISABLE, Timer Control and Max Function Number Register.TIMER_MOD_ACK_NAK, Ack Latency Timer and Replay Timer Register.ROUND_TRIP_LATENCY_TIME_LIMIT and Ack Frequency and L0-L1 ASPM Control Register.ACK_FREQ . When the link is idle, the controller sends an ACK DLLP every time it receives a good TLP. The default values are a tradeoff for performance vs. latency tolerance. If those do not fit the special custom requirements, they can be adapted.

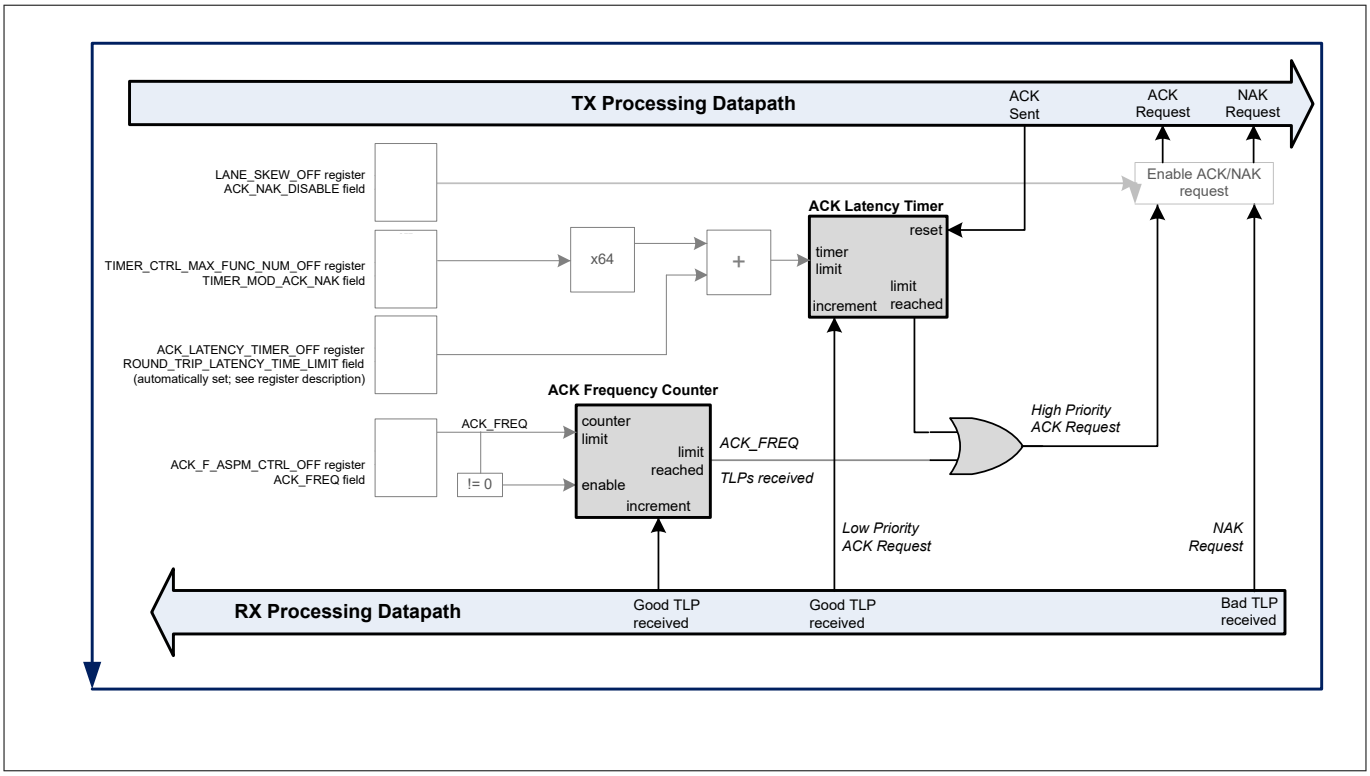


Figure 280 Ack frequency and latency timers operation (Representative)

When a low priority ACK request is scheduled for transmission to the remote link partner, the controller waits until either the ACK Latency Timer or the ACK Frequency Counter expires, then converts the low priority ACK request to a high priority ACK request, and schedules it for transmission to the remote link partner.

Transmit replay

You can modify the time-out value of the replay buffer as shown in the following figure. The default values are a tradeoff for performance vs. latency tolerance. If those do not fit your special custom requirements you can adapt them.

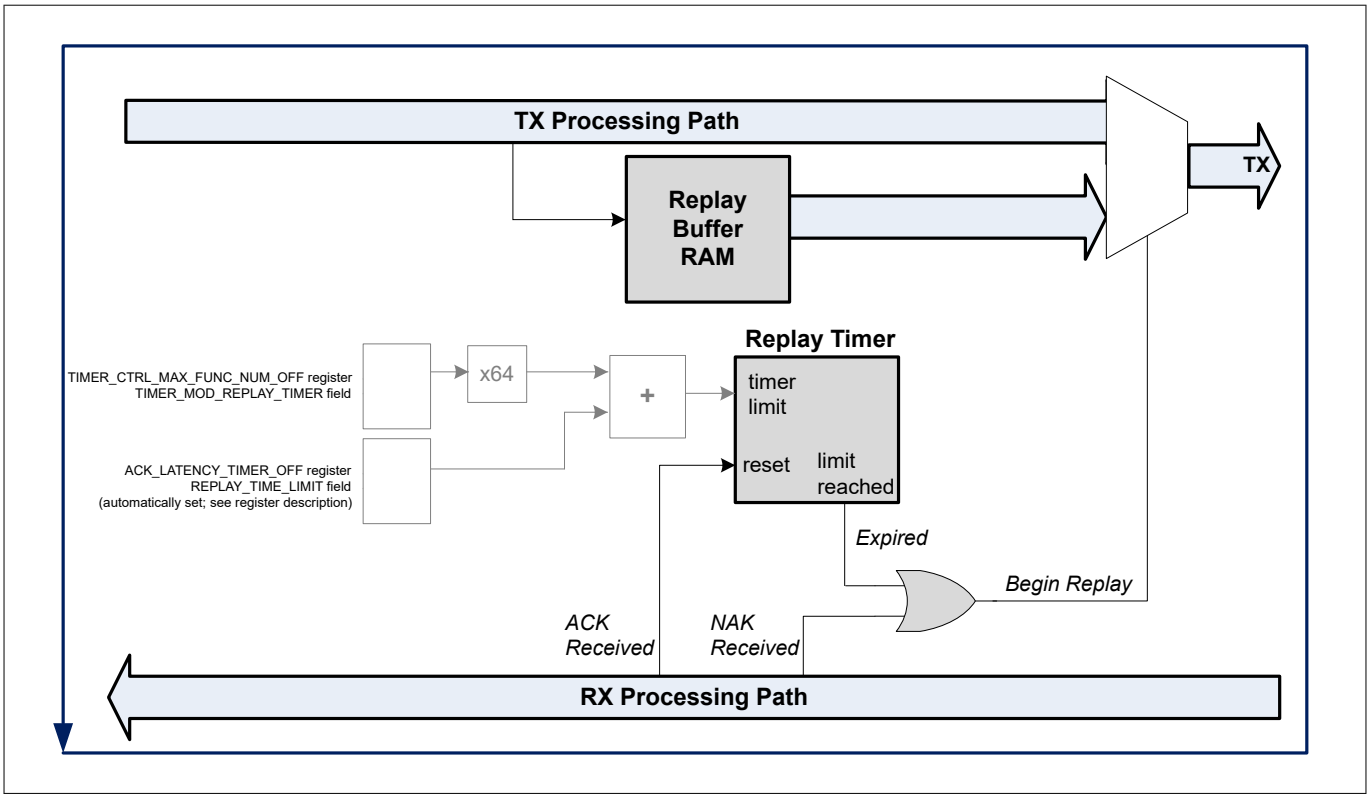


Figure 281 Replay buffer and time (representative)

Interface progression detection

This safety feature provides a mechanism for the controller to inform the application about the lack of progression on SRI/AXI slave and master interfaces. It also monitors the internal message generation interface and the configuration bus completion interface. Progression on the data transmission interfaces is not expected to halt under normal operating circumstances; it might halt if soft errors or silicon failures occur.

Tx interface progression detection features

- Timeout detection unit sets a flag (INTERFACE_TIMER_STATUS_OFF) when interface is stalling/waiting. It resets the flag when interface is not stalling/waiting
- Watchdog timer clocks the timeout detection units
- Controller automatically adjusts watchdog timer's incrementing interval when clock frequency changes after Gen1/2/3 speed change
- Controller automatically adjusts watchdog timer's incrementing interval in L1 and L2 low-power states
- Controller output if_timeout_status signal that is merged into SMU alarms to indicates a time-out event
- Programmable generation of AER message after time-out event (INTERFACE_TIMER_CONTROL_OFF.INTERFACE_TIMER_AER_EN)

The interface progression timer module instantiates a watchdog timer and timeout detection units in the interfaces to be monitored. The application software sets a window for monitoring transfer progression by programming the watchdog timer period through INTERFACE_TIMER_TARGET_OFF register. The watchdog timer signal is connected to the timeout detection units. When a change occurs in the watchdog timer signal, the timeout detection unit checks the interface signals to determine if the interface is halted, and sets a flag. This flag is reset if the interface is progressing. If the flag is found to be set on the next change to the watchdog timer signal, interface timeout status signal (if_timeout_status) is asserted. This uses an SMU alarm (PCIe uncorrectable data alarm) to indicate to the application that a progression timeout has occurred on one of the interfaces. The application can then read the INTERFACE_TIMER_STATUS_OFF register to determine which interface is causing the timeout.

19.3.3.2 Receive TLP

Receive filtering

The controller contains a filter module that is responsible for the following tasks:

- Determine the status of a received TLP using filtering rules (SYMBOL_TIMER_FILTER_1_OFF and FILTER_MASK_2_OFF)
- Determine the destination of a received TLP based on the filtering status
- Indicate the status of the received TLP
- Report Errors to AER registers in RC via error messages based on filter results. If more than one type of error is detected, Section 6.2.3.2.3 Error Pollution, of the PCI Express Base Specification, Revision 5.0, Version 1.0 is followed

The controller filters and routes the received TLPs according to a set of rules determined by the TLP type based on the PCI Express Base Specification, Revision 5.0, Version 1.0 and user-configurable filtering options. The filtering rules for a received TLP are affected by the register values (run-time options).

The following general rules apply to all incoming TLPs that are not malformed. By default:

- If the device is in power state D3hot, the controller only accepts CFG and MSG requests TLPs for that function. All other incoming request types for that function are treated as unsupported requests (UR)
- When the controller detects an error in a received TLP, it normally performs the following:
  - Discarding the TLP
  - Generating a completion (for non-posted requests) with the completion status set to CA or UR
  - Setting the status in the PCI-compatible Status register
  - Seting the status in the AER registers (when the user enables AER)
  - Generating an error message (upstream port only)
- All error-free MSG requests are decoded internally, signaled on the message interface (MSG_PAYLOADx) and then dropped

Receive routing

This section discusses how the controller routes different TLP types to different receive interfaces depending on TLP type and filter result.

EP mode routing overview

The possible destinations of a posted or non-posted request TLP are bus master interface (TRGT1), controller internal space (TRGT0) or discard[15]. By default:

- CFG requests are routed to controller internal space
- BAR-matched MEM and I/O requests are routed to the bus master interface
- MSG requests are decoded internally, signaled on the registers, and then terminated

The possible destinations of a completion TLP are bus interface and discard.

RC mode routing overview

The possible destinations of a posted or non-posted request are bus master interface (TRGT1) or discard. By default:

- MEM requests outside of the memory range and prefetchable memory range as determined by the corresponding Base and Limit fields in the Type-1 header are routed to the bus master interface
- MSG requests are decoded internally, signaled on the registers, and then terminated
- An RC does not expect to receive CFG or I/O requests
- BARs should be disabled and not used

15 Serviced internally, and then not passed to the application but dropped.

The possible destinations of a completion TLP are the bus interface or discard

19.3.4 Ordering

Inbound order enforcement

The controller only enforces ordering of TLPs that are delivered to the primary receive application interfaces TRGT1. It does not enforce ordering between these TLPs and the TLPs that are delivered to the TRGT0 or SII.

The system information interface (SII) exchanges system information between the controller and the application via the registers. It is connected to the output of the receive filter and is not subject to receive queuing. Therefore, TLPs that are routed to the SII may overtake the TLPs that are destined for the application memory. Typically, the controller sends all message TLPs including Assert_INTx/Deassert_INTx messages to the SII. An MSI interrupt received by the controller is signaled on the SII may overtake the received data destined for the application memory.

PCIe ordering :

Summary: This scheme fully obeys the producer-consumer model and deadlock is avoided. All of the PCI Express ordering rules in Table 2-34 (“2.4.1 Transaction Ordering Rules” section of the PCI Express Base Specification, Revision 5.0, Version 1.0) are obeyed.

Method: TLPs are filtered into three separate FIFO queues: posted, non-posted, and completion. The ordering controller determines which of the three queues have a TLP that is allowed to proceed.

When more than one queue has a TLP that is allowed to proceed, it uses a priority arbitration algorithm to select a transaction. This prevents starvation, which occurs when delivery rate for a particular TLP type to your application is unfairly reduced.

Rules: This scheme implements all of the PCI Express ordering rules.

Relaxed Ordering (RELAXED_ORDER_SUPPORT =1): A completion or a posted transaction with its RO header bit set to 1, can pass a previously issued posted transaction that is blocked. RO is not implemented for non-posted writes.

Outbound order enforcement

The controller does not enforce ordering between the transmit interfaces, DBI (SPB), SII, XALI0, XALI1, XALI2, and so on. Furthermore, it does not enforce ordering between transactions on any of the transmit interfaces, and internally generated messages or completions (for example, for a CFG access to internal registers).

The request arbitration priority (highest first) for transmission of TLPs at the transmit interfaces is:

1. Internally generated messages
2. Internally generated completions for received CFG requests
3. Transmit TLPs from XALI0/1/2 according to the arbitration scheme in the chapter Transmit TLP

Related information

Transmit TLP on page 2711

19.3.5 Interrupts

In this section, both the local interrupts and the PCIe interrupts (interrupt generation and reception) are introduced. The local interrupts are the interrupt signals to the system interrupt router. They include 8 PCIe interrupts (converted from PCIe interrupts namely INTX, MSI and MSI-X), 8 DMA interrupts, error interrupt, power management interrupt, link status interrupt, PCIe message interrupt and others.

All three PCIe interrupt mechanisms namely INTx, MSI and MSI-X are supported. INTx, MSI and MSI-X are mutually exclusive. Only one interrupt mechanism can be applied at a time. UpStream Port (USP) generates interrupt while DownStream Port (DSP) receives interrupts, as specified in the PCIe specification.

Message signaled interrupts (MSI or MSI-X) are the preferred interrupt mechanism in PCIe. PCI Express defines one legacy interrupt (INTA) message for a single function device and up to four legacy interrupt (INTA, INTB,

INTC and INTD) messages for a multi-Function device. Since PCIe is a single function device, only INTA can be generated in EP mode, however, can receive all the 4 interrupts, INTA, INTB, INTC and INTD in RC mode.

The following section describes the processing of interrupts in the controller. The following descriptions are based on the different types of interrupts as specified in Sections 6.1.1, “Rationale for PCI Express Interrupt Model”,6.1.4, “Message Signaled Interrupt (MSI or MSI-X) Support”, 6.1.2, “PCI Compatible INTx Emulation”, and 6.1.3, “INTx Emulation Software Model” of the PCI Express Base Specification, Revision 5.0, Version 1.0.

MSI or MSI-X uses MWr to represent interrupt Messages. The Request format used for MSI or MSI-X transactions is identical to the MWr, and MSI or MSI-X Requests are indistinguishable from memory writes with regard to ordering, flow control, and data integrity.

19.3.5.1 Interrupt reception

When used as an RC, AURIX™ is intended to service the interrupts received from USPs. The configuration of USPs must be set properly by RC.

The inbound PCIe interrupts are propagated to the IR when AURIX™ is used as an RC, as depicted in the next figure. Interrupts received by the PCIe RC are forwarded directly to the IR that is able to trigger interrupts of CPUs. All three interrupts namely INTx, MSI and MSI-X utilize this mechanism. The RC has to configure the EP properly according to the PCIe specification in order to map the interrupts correctly.

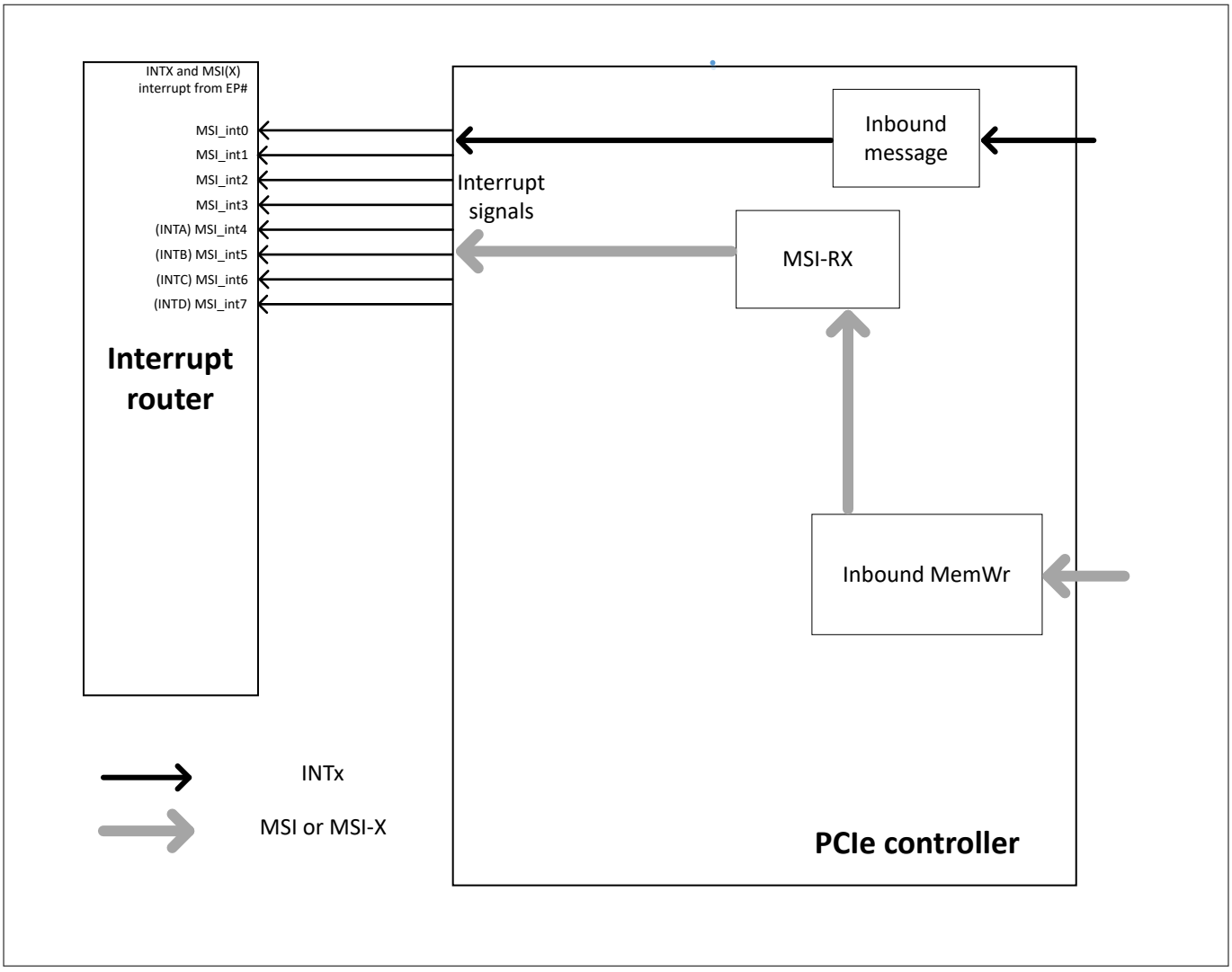


Figure 282 Interrupt reception options

PCIe controller interrupt reception

The controller decodes received Assert_INTx or Deassert_INTx messages and pulses the interrupt signals to generate a level-based interrupt to IR.

The inbound interrupts including MSI and MSI-X in the form of MWr TLP received by the controller are terminated, rather than propagating MSI MWr TLPs onto the bus through the master interface. The MSI packets are captured by the MSI-RX logic and interrupts are signaled directly to the IR. The MSI reception logic (MSI-RX) is programmed with an address (MSI_CTRL_ADDR_OFF and MSI_CTRL_UPPER_ADDR_OFF) that is used as the system MSI address. When an inbound MWr request matches this address for an MSI memory write request, an MSI interrupt is detected. Afterwards, the inbound MWr request is dropped and never appears on the bus.

The following conditions have to be fulfilled for the MSI-RX detection.

- Header attributes bits are 0_H. No snoop (NS) and relaxed ordering (RO) fields must be 0_H
- Length field is 01_H to indicate a payload of one DWORD
- First byte enable (FBE) is $xx11_B$ (enabling the first two bytes of the payload)
- Last byte enable (LBE) is 0000_B
- TLP address corresponds to systems chosen MSI address as programmed in the MSI_CTRL_ADDR_OFF and MSI_CTRL_UPPER_ADDR_OFF port logic registers. This register is a vendor specific register and is not the MSI Lower 32 Bits Address Register which is part of the PCIe MSI capability register

The MSI-RX decodes the MSI MWr data payload to determine which endpoint device sent the MSI and which interrupt vector the MSI corresponds to. When a valid interrupt has been decoded, the corresponding signal to IR is asserted. This output remains asserted when any MSI interrupt is pending. It is only de-asserted when there is no MSI interrupt pending. The detailed process is explained in the following figure.

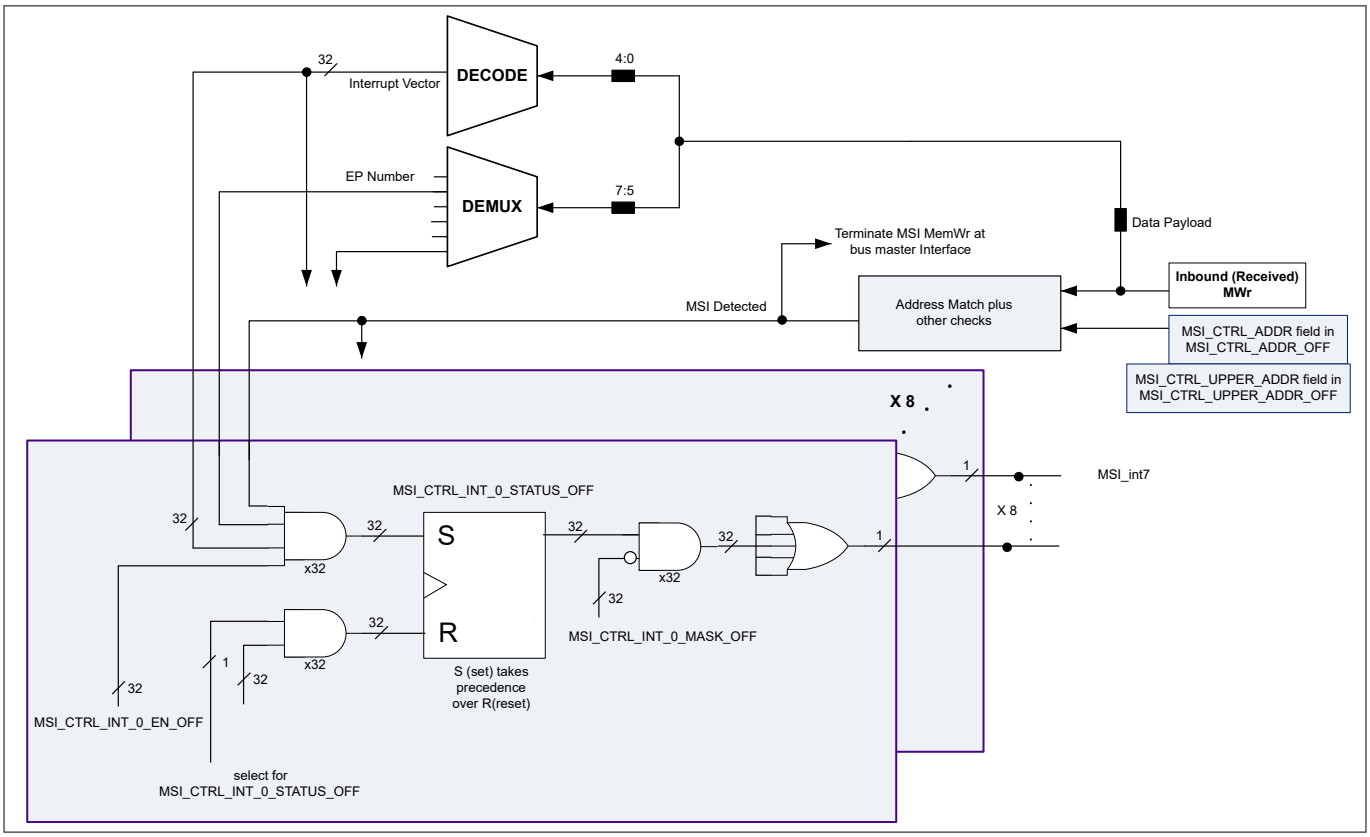


Figure 283 MSI-RX interrupt detection process

When the decoded interrupt vector is enabled (MSI_CTRL_INT_i_EN_OFF) and not masked (MSI_CTRL_INT_i_MASK_OFF), then the controller sets the corresponding bit in the MSI_CTRL_INT_i_STATUS_OFF. This bit remains asserted until the application clears the status bit by writing a 1 to the status bit. Writing a 0 has no effect.

When the decoded interrupt vector is enabled but is masked, then the controller sets the corresponding bit in the status register but it does not assert the signal to IR. When an MSI interrupt vector is received from an endpoint but that vector has not been enabled in the corresponding MSI_CTRL_INT_i_EN_OFF, then the controller does not set any bit in the interrupt status register. In addition, when no interrupts have been enabled in any of the eight interrupt enable registers, then all MSI detection logic is disabled and valid MSI MWr request TLPs are not terminated in the bridge but are passed by the bus master interface to the SRI bus.

19.3.5.1.1 Programming and usage

In order for the MSI interrupt system to work properly, the host has to configure the EP and also its own MSI-RX logic. The MSI-RX logic is responsible for the reception of MSI or MSI-X interrupts.

- The host CPU configures the MSI capabilities of all endpoints through CFG requests. Program the MSI Data Register (MSI_CAP_OFF_08H_REG or MSI_CAP_OFF_0CH_REG in the MSI capability) as introduced in the following table to allow the MSI-RX to decode the interrupt source
- Program the MSI Capability Lower 32 Bits Address Register (MSI_CAP_OFF_04H_REG) of every EP with one common MSI address
- The host CPU configures the MSI-RX through the local bus
  - It programs the MSI-RX's MSI_CTRL_ADDR_OFF and MSI_CTRL_UPPER_ADDR_OFF registers with the same value as in MSI_CAP_OFF_04H_REG
  - It reads the MSI capability of each EP to determine the number of vectors enabled in each EP and uses this information to program the MSI-RX's interrupt enable register (MSI_CTRL_INT_i_EN_OFF). This register allows up to 32 MSI interrupt vectors to be enabled within the MSI-RX for a given EP. It is the responsibility of the host CPU to read the contents of the Multiple Message Enable field in an EPs MSI capability structure and program that EPs Interrupt Enable register in the MSI-RX appropriately. For example, if the Multiple Message Enable is 100_B for endpoint #0 (which corresponds to 16 enabled interrupt vectors), the host CPU should program MSI_CTRL_INT_i_EN_OFF with $0000FFFF_H$

Table 833 MSI data register (MSI_CAP_OFF_08H_REG or MSI_CAP_OFF_0CH_REG) programming

15:8	7:5	4:0
Not used	EP number: Allows each EP to be identified within the system, For example, EP#5 is programmed with 101_B	Interrupt vector: Identifies the interrupt source within each EP. Programmed to 00000_B. Supports up to 32 Vectors

The example above shows only a possibility in a topology. The upper three bits are treated as the indication of 8 different EPs while the lower five bits are for 32 interrupts. The interpretation of data is flexible. For example, the upper 3 bits can also be used for a single EP when MSI-X is applied as MSI-X has flexible configuration of data and address.

For MSI and MSI-X reception, the data registers configuration of EPs shall follow the above rules in order to decode interrupts correctly. A software interrupt handler can read which interrupt is triggered by reading the MSI_CTRL_INT_i_STATUS register.

Related information

iATU on page 2752

19.3.5.1.2 Request trigger for EP

According to the PCIe specification, all types of interrupts, INTx, MSI and MSI-X are supposed to be received only by RC. The reception of interrupt by EP is not defined by the protocol. However, a special mechanism named Request Trigger is implemented to support interrupt reception even when AURIX™ works as an EP. The Request Trigger mechanism is not limited from RC to EP but also EP to EP. This mechanism can only be used when the device works as EP.

A special register named INT_REQTRIG is designed and it can be accessible from local bus or from wire, but can only written by wire. It is one of the EP interrupt trigger registers mentioned before and can be accessed through MEM or IO TLPs from wire. The MEM/IO TLPs shall target in the range of BAR0. The lower 5 bits of INT_REQTRIG are used by the decode logic to map them to 32 interrupt requests, as depicted in the following figure. The remaining 27 bits of the 32 bits in INT_REQTRIG are not functionally connected to hardware and can be optionally used by the application. The INT_TRIGSTAT is also readable from wire to facilitate the checking of interrupt service status. This register can only be cleared by the local software writing the INT_TRIGCLR register.

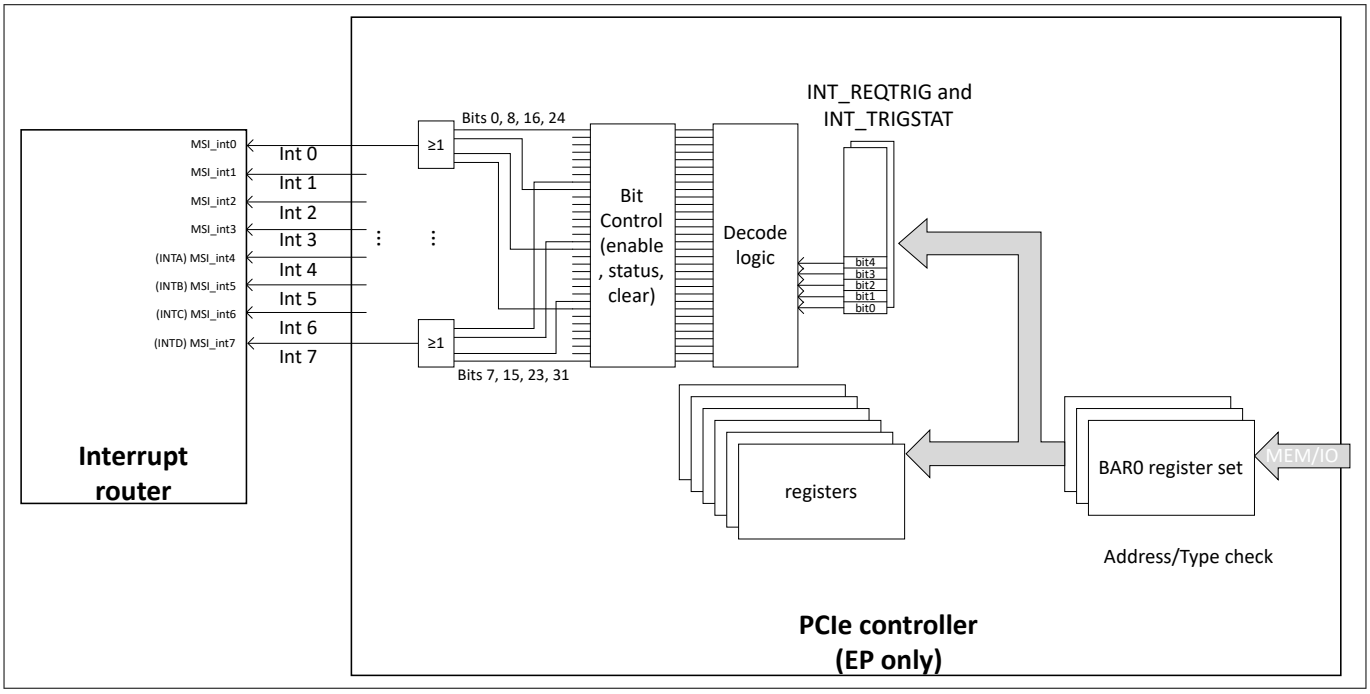


Figure 284 The request trigger mechanism for both RC and EP

After the decoding of the 5 bits, 32 interrupt requests are passed to the bit control block that consist of three different registers namely INT_REQTRIG, INT_TRIGSTAT and INT_TRIGCLR. The bit control block decides which of the 32 interrupts that will be masked, indicates the current interrupt status and clears the interrupts after service. The 32 interrupt requests will then be merged into 8 interrupts that will be handed over to IR and software interrupts triggered. The merging of interrupt requests is logical OR and is according to the following table.

Table 834 Request merging table

Interrupt	Merged request
int 0	0, 8, 16, 24
int 1	1, 9, 17, 25
int 2	2, 10, 18, 26
int 3	3, 11, 19, 27
int 4	4, 12, 20, 28
int 5	5, 13, 21, 29
int 6	6, 14, 22, 30
int 7	7, 15, 23, 31

Once a specific interrupt signal is triggered and interrupt handler starts to execute, the service handler software shall check the status of request (INT_TRIGSTAT) to see which request of the four is asserted. After which the service software shall clear the request bit by asserting the corresponding bit in INT_TRIGCLR.

If a request is not yet serviced while the same request comes again, then the second one will be missed and will not be seen by software.

19.3.5.2 Interrupt generation

As an EP, AURIX™ is supposed to trigger interrupts to RC. RC needs to configure the EP using the configuration space for matching interrupt generation on EP to interrupt reception on RC. Only one type of interrupt is allowed to be used at a time as MSI, MSI-X, and INTx are mutually exclusive.

19.3.5.2.1 INTx

To generate INTx interrupts, software needs to write INT_INTXCTRL.INTX bit-field to notify the controller that it should send an interrupt message. The controller generates two message TLPs, Assert_INTx and Deassert_INTx, in response to the assertion and the de-assertion of this bit.

Your application needs to de-assert the virtual interrupt bit-field when the link has been placed in a low power state. The controller does not automatically send a Deassert_INTx interrupt message when the power state changes.

19.3.5.2.2 MSI

An EP creates MSI interrupts using MWr requests. The application issues MSI requests through the local MSI registers; the controller then generates the corresponding memory write. Alternatively, your application can create the MSI MWr by writing to the SRI slave interface or using DMA.

The interrupts can be triggered by asserting the INTi (0 to 31) field in INT_MSICTRL register. Each bit in the INT_MSICTRL is corresponding to one of the 32 interrupts that can be supported by EP. If more than one bit is asserted at a time, the generation of MSI interrupt is processed one by one depending on the priority. The interrupt 0 has the highest priority while the interrupt 31 has the lowest. The status as to whether an interrupt has been sent is available in the INT_MSISTAT register.

INT_MSISTAT indicates whether the previous interrupt request is pending or not. A Non-pending interrupt does not mean that the interrupt is serviced but the interrupt message has been sent. If the application requests an interrupt that is still pending, this request will be lost.

MSI capability registers shall be configured by RC via Cfg TLPs. The details of these registers are specified by the PCIe protocol and not the scope of this manual.

19.3.5.2.3 MSI-X

MSI-X is an extension to basic MSI. It supports more vectors (2048) per function and independent address and data values for each interrupt compared to MSI.

Each entry in the MSI-X table corresponds to an interrupt. Each interrupt could have different message data and message address, as depicted in the following figure. The maximum number of interrupts supported by MSI-X is 64 (N=64). The 64 bits inside in the PBA table corresponds to 64 interrupts or 64 MSI-X table entries, as shown in the figure MSI-X PBA structure. When an MSI-X packet is generated, the data and the address (upper and lower addresses) in the MSI-X table are used to form the packet.

The detailed configuration process of MSI-X is described in the PCIe specification.

DWORD 3	DWORD 2	DWORD 1	DWORD 0	
Vector control	Msg data	Msg upper address	Msg lower address	entry 0
Vector control	Msg data	Msg upper address	Msg lower address	entry 1
Vector control	Msg data	Msg upper address	Msg lower address	entry 2
...	...	...	...	...
Vector control	Msg data	Msg upper address	Msg lower address	entry N-2
Vector control	Msg data	Msg upper address	Msg lower address	entry N-1

Figure 285 MSI-X table structure

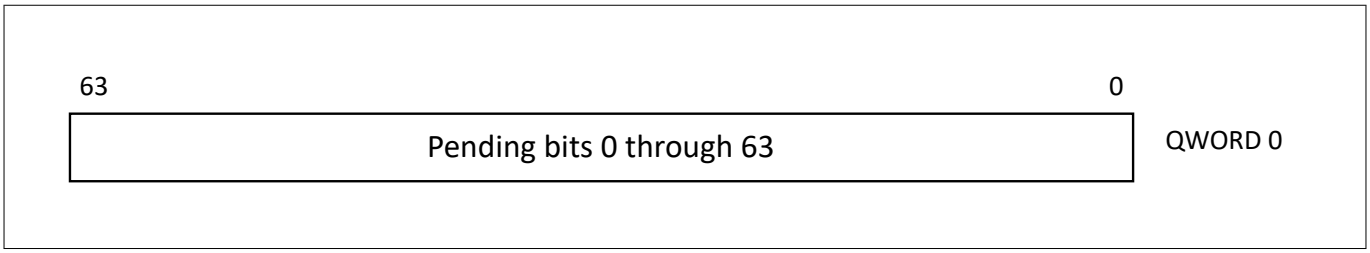


Figure 286 MSI-X PBA structure

The local application needs to configure the MSI-X logic before triggering any MSI-X interrupt. The detailed sequence of configuration as follows:

- Read the MSI-X Capability Structure registers through the local bus to get the value of MSI-X Table Size, Table BIR, Table Offset, PBA Offset, and PBA BIR. Get the starting address of the MSI-X Table as follows: MSI-X Table Start address = Start Address of BARn + Table Offset; where n = Table BIR (n = 0 in this case)
- Populate the MSI-X Table (by writing over the wire or local bus) with the required Message Address, Message Data, and Vector Control information for each entry, as explained in the figure MSI-X table structure. For example, to populate entry 0 of the MSI-X Table, perform four consecutive 32-bit writes to MSI-X Table Start Address + 0_H, 4_H,8_H and C_H with Data0 = Msg lower address, Data1 = Mag upper address, Data2 = Msg data, and Data3 = Vector control. Similarly to populate entry 2 write to "MSI-X Table Start Address + 2"
- Automatic triggering of MSI-X generation when application writes to the MSIX_DOORBELL_OFF register

MSI-X interrupt generation can be achieved by directly writing MSIX_DOORBELL_OFF by software. If the software needs to trigger a specific interrupt 'i', the index 'i' needs to be written to the vector field of MSIX_DOORBELL_OFF. Then the corresponding MSI-X table entry that contains the message data and message address is used to generate a MSI-X message to RC. The pending bits in the PBA structure indicate if a function has a pending message for the associated MSI-X table entry

MSIX_ADDRESS_MATCH_LOW_OFF and MSIX_ADDRESS_MATCH_HIGH_OFF are not supposed to be used and they shall not be modified after reset.

19.3.5.3 Local interrupts

This part describes the interrupts to the IR. It is not limited to the PCIe interrupts received but also includes local interrupts for example DMA interrupt, link state change, power management event, and errors.

Table 835 Summary of service requests to IR

Index	Service request name	Description	SRC register in IR	Relevant global interrupt control	Relevant local interrupt control
1	Remote int0	An interrupt requested from wire in RC role. It can be generated from MSI or MSI-X. In EP role int0 request trigger	SRC_PCIEwMSI0		MSI_CTRL_INT_0_STATUS_OFF[1], MSI_CTRL_INT_0_EN_OFF, MSI_CTRL_INT_0_MASK_OFF INT_TRIGSTAT, INT_TRIGCLR, INT_TRIGEN
2	Remote int1	An interrupt requested from wire in RC role. It can be generated from MSI or MSI-X. In EP role int1 request trigger	SRC_PCIEwMSI1		MSI_CTRL_INT_1_STATUS_OFF[1], MSI_CTRL_INT_1_EN_OFF, MSI_CTRL_INT_1_MASK_OFF INT_TRIGSTAT, INT_TRIGCLR, INT_TRIGEN
3	Remote int2	An interrupt requested from wire in RC role. It can be generated from MSI or MSI-X. In EP role int2 request trigger	SRC_PCIEwMSI2		MSI_CTRL_INT_2_STATUS_OFF[1], MSI_CTRL_INT_2_EN_OFF, MSI_CTRL_INT_2_MASK_OFF INT_TRIGSTAT, INT_TRIGCLR, INT_TRIGEN
4	Remote int3	An interrupt requested from wire in RC role. It can be generated from MSI or MSI-X. In EP role int3 request trigger	SRC_PCIEwMSI3		MSI_CTRL_INT_3_STATUS_OFF[1], MSI_CTRL_INT_3_EN_OFF, MSI_CTRL_INT_3_MASK_OFF INT_TRIGSTAT, INT_TRIGCLR, INT_TRIGEN

(table continues...)

Table 835 (continued) Summary of service requests to IR

Index	Service request name	Description	SRC register in IR	Relevant global interrupt control	Relevant local interrupt control
5	Remote int4	An interrupt requested from wire in RC role. It can be generated from INTX, MSI or MSI-X. In EP role int4 request trigger	SRC_PCIEwMSI4		MSI_CTRL_INT_4_STATUS_OFF[1], MSI_CTRL_INT_4_EN_OFF, MSI_CTRL_INT_4_MASK_OFF INT_TRIGSTAT, INT_TRIGCLR, INT_TRIGEN
6	Remote int5	An interrupt requested from wire in RC role. It can be generated from INTX, MSI or MSI-X. In EP role int5 request trigger	SRC_PCIEwMSI5		MSI_CTRL_INT_5_STATUS_OFF[1], MSI_CTRL_INT_5_EN_OFF, MSI_CTRL_INT_5_MASK_OFF INT_TRIGSTAT, INT_TRIGCLR, INT_TRIGEN
7	Remote int6	An interrupt requested from wire in RC role. It can be generated from INTX, MSI or MSI-X. In EP role int6 request trigger	SRC_PCIEwMSI6		MSI_CTRL_INT_6_STATUS_OFF[1], MSI_CTRL_INT_6_EN_OFF, MSI_CTRL_INT_6_MASK_OFF INT_TRIGSTAT, INT_TRIGCLR, INT_TRIGEN
8	Remote int7	An interrupt requested from wire in RC role. It can be generated from INTX, MSI or MSI-X. In EP role int7 request trigger	SRC_PCIEwMSI7		MSI_CTRL_INT_7_STATUS_OFF[1], MSI_CTRL_INT_7_EN_OFF, MSI_CTRL_INT_7_MASK_OFF INT_TRIGSTAT, INT_TRIGCLR, INT_TRIGEN
9	DMA int0	DMA write channel0 interrupt	SRC_PCIEwWC0		DMA_WRITE_INT_STATUS_OFF, DMA_WRITE_INT_MASK_OFF, DMA_WRITE_INT_CLEAR_OFF

(table continues...)

Table 835 (continued) Summary of service requests to IR

Index	Service request name	Description	SRC register in IR	Relevant global interrupt control	Relevant local interrupt control
10	DMA int1	DMA write channel1 interrupt	SRC_PCIEwWC1		DMA_WRITE_INT_STATUS_OFF, DMA_WRITE_INT_MASK_OFF, DMA_WRITE_INT_CLEAR_OFF
11	DMA int2	DMA write channel2 interrupt	SRC_PCIEwWC2		DMA_WRITE_INT_STATUS_OFF, DMA_WRITE_INT_MASK_OFF, DMA_WRITE_INT_CLEAR_OFF
12	DMA int3	DMA write channel3 interrupt	SRC_PCIEwWC3		DMA_WRITE_INT_STATUS_OFF, DMA_WRITE_INT_MASK_OFF, DMA_WRITE_INT_CLEAR_OFF
13	DMA int4	DMA read channel0 interrupt	SRC_PCIEwRC0		DMA_READ_INT_STATUS_OFF, DMA_READ_INT_MASK_OFF, DMA_READ_INT_CLEAR_OFF
14	DMA int5	DMA read channel1 interrupt	SRC_PCIEwRC1		DMA_READ_INT_STATUS_OFF, DMA_WRITE_INT_MASK_OFF, DMA_WRITE_INT_CLEAR_OFF
15	DMA int6	DMA read channel2 interrupt	SRC_PCIEwRC2		DMA_READ_INT_STATUS_OFF, DMA_WRITE_INT_MASK_OFF, DMA_WRITE_INT_CLEAR_OFF
16	DMA int7	DMA read channel3 interrupt	SRC_PCIEwRC3		DMA_READ_INT_STATUS_OFF DMA_WRITE_INT_MASK_OFF DMA_WRITE_INT_CLEAR_OFF

(table continues...)

Table 835 (continued) Summary of service requests to IR

Index	Service request name	Description	SRC register in IR	Relevant global interrupt control	Relevant local interrupt control
17	Error handling	General errors including PCIE errors and internal errors	SRC_PCIEwERR	ALARM_STAT.RCERRINT, ALARM_INTEN.RCERREN, ALARM_INTCLR.RCERRCLR	ROOT_ERR_STATUS_OFF.any bit[1)], ROOT_ERR_CMD_OFF. corresponding bit
				ALARM_STAT.SAFCORR, ALARM_INTEN.SAFCORREN, ALARM_INTCLR.SAFCORRCLR	SAFETY_STATUS_OFF.SAFETY_INT_STATUS_4[1)], SAFETY_STATUS_OFF.SAFETY_INT_STATUS_5[1)] SAFETY_MASK_OFF.SAFETY_INT_MASK_4, SAFETY_MASK_OFF.SAFETY_INT_MASK_5
				ALARM_STAT.SAFUNCORR, ALARM_INTEN.SAFUNCORREN, ALARM_INTCLR.SAFUNCORRCLR	SAFETY_STATUS_OFF.SAFETY_INT_STATUS_0[1)], SAFETY_STATUS_OFF.SAFETY_INT_STATUS_1[1)], SAFETY_STATUS_OFF.SAFETY_INT_STATUS_2[1)], SAFETY_STATUS_OFF.SAFETY_INT_STATUS_3[1)] SAFETY_MASK_OFF.SAFETY_INT_MASK_0, SAFETY_MASK_OFF.SAFETY_INT_MASK_1, SAFETY_MASK_OFF.SAFETY_INT_MASK_2, SAFETY_MASK_OFF.SAFETY_INT_MASK_3

(table continues...)

Table 835 (continued) Summary of service requests to IR

Index	Service request name	Description	SRC register in IR	Relevant global interrupt control	Relevant local interrupt control
				ALARM_STAT.SYSERR, ALARM_INTEN.SYSERREN, ALARM_INTCLR.SYSERRCLR	(ROOT_ERR_STATUS_OFF. ERR_COR_RX, NON_FATAL_ERR_MSG_RX, FATAL_ERR_MSG_RX)[1], (ROOT_CONTROL_ROOT_CAPABILITIES_REG.PCIE_CAP_SYS_ERR_ON_CORR_ERR_EN, PCIE_CAP_SYS_ERR_ON_NON_FATAL_ERR_EN, PCIE_CAP_SYS_ERR_ON_FATAL_ERR_EN)
18	Error messages related to AER or baseline received	Non-fatal, Correctable error and fatal error. RC only.	SRC_PCIEwNFAT	ERR_STAT.CORR, ERR_INTEN.CORREN, ERR_INTCLR.CORRCLR	ROOT_ERR_STATUS_OFF. ERR_COR_RX[1], ROOT_CONTROL_ROOT_CAPABILITIES_REG.PCIE_CAP_SYS_ERR_ON_CORR_ERR_EN
				ERR_STAT.NFAT, ERR_INTEN.NFATEN, ERR_INTCLR.NFATCLR	ROOT_ERR_STATUS_OFF.NON_FATAL_ERR_MSG_RX[1], ROOT_CONTROL_ROOT_CAPABILITIES_REG.PCIE_CAP_SYS_ERR_ON_NON_FATAL_ERR_EN
				ERR_STAT.FAT, ERR_INTEN.FATEN, ERR_INTCLR.FATCLR	ROOT_ERR_STATUS_OFF.FATAL_ERR_MSG_RX[1], ROOT_CONTROL_ROOT_CAPABILITIES_REG.PCIE_CAP_SYS_ERR_ON_FATAL_ERR_EN

(table continues...)

Table 835 (continued) Summary of service requests to IR

Index	Service request name	Description	SRC register in IR	Relevant global interrupt control	Relevant local interrupt control
19	PM request	Events related to power management	SRC_PCIEwPM	PM_CTRL.PMEINT, PM_INTEN.PMEINTEN, PM_INTCLR.PMEINTCLR	CON_STATUS_REG.PME_STATUS[1)], ROOT_STATUS_REGISTER..PCIE_CAP_PME_STATUS[1)], (CON_STATUS_REG.PME_ENABLE , ROOT_CONTROL_ROOT_CAPABILITIES_REG.PCIE_CAP_PME_INT_EN, TYPE1_STATUS_COMMAND_REG.INT_EN)
				PM_CTRL.PMEMSG, PM_INTEN.PMEMSGEN, PM_INTCLR.PMEMSGCLR	
				PM_CTRL.TRNACK, PM_INTCLR.TOACKCLR, PM_INTEN.TOACKEN	
				PM_CTRL.TRN, PM_INTEN.TOEN, PM_INTCLR.TOCLR	
				PM_STAT1.WAKINT, PM_INTEN.WAKINTEN, PM_INTCLR.WAKINTCLR	
20	Link int	Link state change	SRC_PCIEwLNK	LNK_STAT.LINKINT, LNK_INTEN.LINKINTEN	Link Control and Link Status Register.PCIE_CAP_LINK_AUTO_BW_STATUS[1)] , (LINK_CONTROL_LINK_STATUS_REG.PCIE_CAP_LINK_AUTO_BW_INT_EN, TYPE1_STATUS_COMMAND_REG.INT_EN)

(table continues...)

Table 835 (continued) Summary of service requests to IR

Index	Service request name	Description	SRC register in IR	Relevant global interrupt control	Relevant local interrupt control
				LNK_STAT.BWINT, LNK_INTEN.BWINTEN	Link Control and Link Status Register.PCIE_CAP_LINK_BW_MAN_STATUS[1)], LINK_CONTROL_LINK_STATUS_REG.PCIE_CAP_LINK_BW_MAN_INT_EN TYPE1_STATUS_COMMAND_REG.INT_EN
				LNK_STAT.EQINT, LNK_INTEN.EQINTEN	LINK_CONTROL2_LINK_STATUS2_REG.PCIE_CAP_LINK_EQ_REQ[1)], LINK_CONTROL3_REG.EQ_REQ_INT_EN
				OBFFMSG.IDLE, OBFFMSG.IDLEEN, OBFFMSG.IDLECLR	
				OBFFMSG.OBFF, OBFFMSG.OBFFEN, OBFFMSG.OBFFCLR	
				OBFFMSG.ACT, OBFFMSG.ACTEN, OBFFMSG.ACTCLR	
				OBFFWAKE.IDLE, OBFFWAKE.IDLEEN, OBFFWAKE.IDLECLR	
				OBFFWAKE.OBFF, OBFFWAKE.OBFFEN OBFFWAKE.OBFFCLR	
				OBFFWAKE.ACT, OBFFWAKE.ACTEN, OBFFWAKE.ACTCLR	
				OBFFWAKE.WAKERR, OBFFWAKE.ERREN, OBFFWAKE.ERRCLR	
				LNK_STAT.PERSTINT, LNK_INTEN.PERSTEN, LNK_CORECTRL.PERSTCLR	
		Link reset		LNK_STAT.LINKRQRST	

(table continues...)

Table 835 (continued) Summary of service requests to IR

Index	Service request name	Description	SRC register in IR	Relevant global interrupt control	Relevant local interrupt control
21	Message int	Message reception interrupts	SRC_PCIEwMSG	MSG_INFO.DRSSTAT, MSG_INFO.DRSEN, MSG_INFO.DRSCLR	LINK_CONTROL_LINK_STATUS_REG.PCIE_CAP_DRS_SIGNALING_CONTROL
				LTR_CTRL.RLTR, LTR_CTRL.RLTREN, LTR_CTRL.RLTRCLR	
22	Miscellaneous int	Miscellaneous interrupts	SRC_PCIEwMISC	PTM_CTRL.TRN, PTM_CTRL.TRNEN, PTM_CTRL.TRNCLR,	
				PTM_CTRL.DUP, PTM_CTRL.DUPEN, PTM_CTRL.DUPCLR	

***Notes**:*

1. *These status bits are write one to clear*

2. *Both the global and local interrupt mask or enable bits have to be enabled in-order for status change to trigger an interrupt to the application software. The global mask bits are the final control to trigger the corresponding interrupt*

***Note**:* *In-order to clear an interrupt status bit, both the global and local clear bits have to cleared after it is being serviced. Please clear all the bits in case of multiple clear bits*

19.3.6 DMA

The RC system CPU, or the EP application CPU, can offload the transferring of large blocks of data to the embedded DMA controller, leaving the CPU free to perform other tasks. You can use up to four read channels and four write channels. It can simultaneously perform the following types of memory transactions:

- DMA write:
  Transfer (copy) of a block of data from local (application) memory to remote (link partner) memory
- DMA read:
  Transfer (copy) of a block of data from remote (link partner) memory to local (application) memory

Therefore the DMA supports full duplex operation, processing read and write transfers at the same time, and in parallel with normal (non-DMA) traffic. Upon completion of a DMA transfer or an error, the DMA optionally interrupts the local CPU or sends an interrupt MWr (IMWr) to the remote CPU. The DMA is highly configurable and you can program it using the local configuration bus interface or over the PCIe wire[16)].

In linked list mode, the DMA fetches the transfer control information (called channel context) for each transfer (block), from a list of DMA elements that you have constructed in local memory.

16 In EP mode the DMA registers can be accessed by wire on the default mapping of BAR0 Address spaces from wire.

19.3.6.1 DMA architecture

A DMA write and read channel operate independently to maximize the performance of the DMA read and write data transfers over the PCIe link. When you configure the DMA with multiple read channels, then it uses a weighted round robin (WRR) arbitration scheme to select the next read channel to be serviced.

The same applies when you have multiple write channels. The WRR can be programmable through DMA_CTRL_DATA_ARB_PRIOR_OFF.

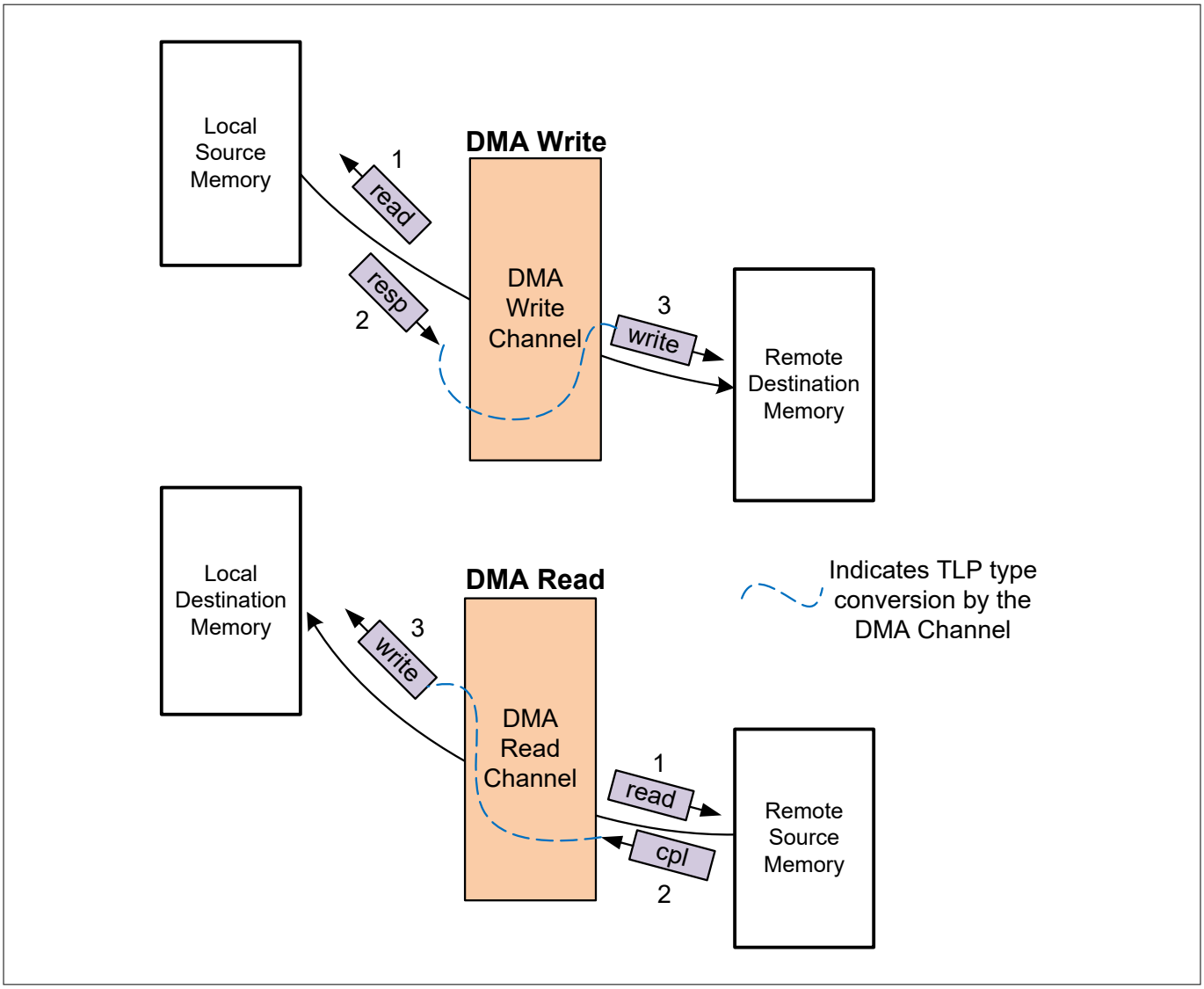


Figure 287 System level view of DMA

DMA write transfer

The DMA injects multiple MRd requests of size less than or equal to Max_Payload_Size into the inbound request path, directed toward the local application. The DMA converts the read responses into MWr TLPs, that it then transmits to the remote link partner. For a write transfer, the SAR is the address of the local memory, and the DAR is the address of the remote memory.

DMA read transfer

The DMA injects multiple MRd requests of size less than or equal to the minimum of {Max_Read_Request_Size, Max_Payload_Size} into the outbound request path, directed toward the remote link partner. The DMA converts the read responses into write requests, that it then transmits to the local SRI bus. When the DMA data transfer is

complete, the CPU is notified via interrupt. For a read transfer, the SAR is the address of the remote memory, and the DAR is the address of the local memory.

Registers and context memory

The registers that are specific to each channel are called the channel context registers. As the transfer progresses, the DMA updates these registers[17]. You program the channel context information and other DMA control registers directly with the local CPU through the configuration bus interface or remotely over the PCIe link.

19.3.6.1.1 Internal architecture

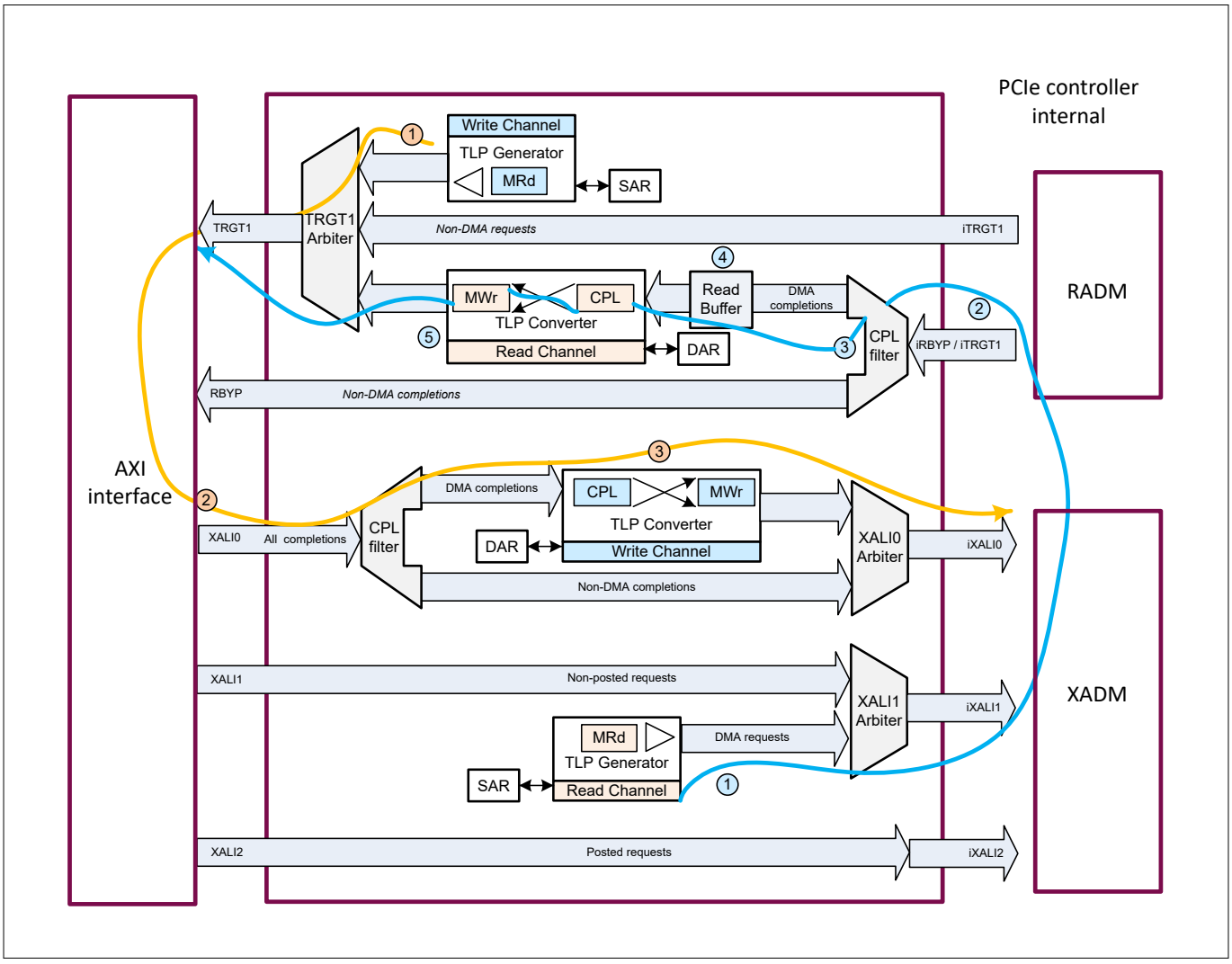


Figure 288 Operation of DMA read (blue) and write (orange) channels

DMA write transfer

The DMA injects multiple MRd requests of size less than or equal to Max_Payload_Size into the local bus. The TLP addresses are derived from the destination address. When the DMA data transfer is complete, the CPU is notified.

1. The DMA generates and sends READ requests to the local interface

17 The DMA hardware sometimes caches the context to increase performance. Therefore these registers might contain a delayed snapshot of the actual DMA channel state.

2. The local bus interface returns data on the XALI0 interface in response to these requests. Data must be returned in strict order within a single channel
3. The DMA converts the CPL TLPs to write requests and sends these to the XADM for transmission to the remote link partner

DMA read transfer

The DMA injects multiple MRd requests of a size less than or equal to the minimum of {Max_Read_Request_Size, Max_Payload_Size} into the outbound request path. The TLP addresses are derived from the source address. When the DMA data transfer is complete, the CPU is notified.

1. The DMA generates and sends read requests to the remote link partner. It uses a DMA tag
2. The link partner returns data in response to these requests on the iRBYP or iTRGT1 interface
3. The DMA module filters this data from non-DMA traffic on iRBYP or iTRGT1 and sends it to the Read Buffer. The DMA module identifies DMA traffic by looking for PCIe completion TLPs that have a DMA tag
4. The Read Buffer reorders completions that have been returned out-of-order. This is to ensure that the DMA writes data to the application in address-order
5. The DMA converts the completion TLPs to write requests and sends these to your application on TRGT1

In above figure, the XALI1 arbitration priority is Read Channel MSI (highest), Write Channel MSI, Non-DMA requests, Read Channel MRd. The XALI0 arbitration priority is Non-DMA requests (highest), Write Channel MWr. The TRGT1 arbitration priority is programmable (using the DMA_CTRL_DATA_ARB_PRIOR_OFF register) and defaults to Non-DMA requests (highest), Write Channel MRd (DMA data requests and LL element/descriptor access), Read Channel MRd (LL element/descriptor access), Read Channel MWr.

The arbitration among of DMA channels is configured through DMA_WRITE_CHANNEL_ARB_WEIGHT_LOW_OFF for write channels and DMA_READ_CHANNEL_ARB_WEIGHT_LOW_OFF for read channels. The 5-bit channel weight specifies the maximum number of TLP requests that the DMA can issue for that channel before it must return to the arbitration routine.

19.3.6.1.2 Interrupts and error handling

The DMA generates two interrupts per channel:

- Done: The DMA successfully completes the transfer
- Abort: The DMA fails to complete the transfer or an error occurs during the transfer

The interrupts are signaled to the software on your CPU, using one or both of the following mechanisms:

- Locally through the interrupt signals to IR. The two interrupts above are merged into one (SRC_PCIERCx for read channels or SRC_PCIEWCx for write channels)
- Remotely using a posted memory write (IMWr), which can be interpreted as an MSI or MSI-X when directed toward the RC
  - There are two programmable IMWr addresses, one each for the done (DMA_WRITE_DONE_IMWR_*) and abort (DMA_WRITE_ABORT_IMWR_*) interrupts. For MSI, you must program all IMWr address registers with the same MSI address, as PCIe only supports a single MSI address per function
  - A single IMWr data register is used for both types of interrupts, so you must read DMA_READ_INT_STATUS_OFF (or DMA_WRITE_INT_STATUS_OFF) to identify the interrupt type
  - The relaxed ordering (RO) bit is automatically set to zero in the TLP header of all IMWr's so that subsequent MWr TLPs do not arrive at the remote device before the IMWr

***Note**: The IMWr address and data do not come from the MSI PCIe capability registers but from DMA registers (DMA_WRITE_ABORT_IMWR_*) or (DMA_WRITE_DONE_IMWR_*). For an upstream port, your software must program these registers with the same information that it is using to program the MSI/MSI-X capability for the function that is associated by your software with that channel in a multifunction setup.*

The interrupt handling mechanism is different for linked list (LL) mode than non LL mode, and there are also some differences between the read and write channels.

Non linked list mode interrupt handling

You enable the local and remote interrupts through the local and remote interrupt enable (LIE and RIE) bits: DMA_CH_CONTROL1_OFF_WRCH_0.LIE and DMA_CH_CONTROL1_OFF_WRCH_0.RIE.

In the write channel, there is only one error condition that results in an abort interrupt. You mask, clear, and read the status of each of the two interrupts (done and abort) through the DMA interrupt register. You can read the status of the abort error (that contribute to the abort interrupt) through the APP_READ_ERR_DETECT field of the DMA_WRITE_ERR_STATUS_OFF register.

If you want a remote interrupt and not a local interrupt, you can set LIE and RIE and mask the local interrupt using the mask register. In this way, the local error status register can still reflect the error status but no interrupt will be propagated to the IR. This allows you to poll the status register to distinguish a DONE from an ABORT.

In the read channel, there are five error conditions that results in an abort interrupt. However, you can read the status of each of the five abort errors (that contribute to the abort interrupt) through DMA_READ_ERR_STATUS_LOW_OFFand DMA_READ_ERR_STATUS_HIGH_OFF.

Linked list mode interrupt handling

The LIE and RIE bits in the LL element enable the channel done interrupts (local and remote). The LLLAIE and LLRAIE bits of the DMA_WRITE_LINKED_LIST_ERR_EN_OFF/DMA_READ_LINKED_LIST_ERR_EN_OFF registers enable the channel abort interrupts (local and remote). In the write channel, there are two error conditions that results in an abort interrupt. You can read the status of each of the two abort errors (that contribute to the abort interrupt) through the DMA_WRITE_ERR_STATUS_OFF register.

In the read channel, there are six error conditions that results in an abort interrupt. You can read the status of each of the six abort errors (that contribute to the abort interrupt) through the DMA_READ_ERR_STATUS_LOW_OFF and DMA_READ_ERR_STATUS_HIGH_OFF registers.

In non-linked list mode, LIE acts as a global switch. However when in linked list mode, LIE is just local to the current linked list element and the global switch is LLLAIE.

Setting RIE and LIE in element A followed by RIE (only) in element B is not a verified usage scenario.

19.3.6.1.3 Linked list mode

The DMA provides a linked list (LL) mode to efficiently move data from source to destination with minimal intervention from the local CPU. This mode provides an alternative to programming the DMA multiple times to transfer multiple blocks of data.

The programming information (address, size, and so on) for each block of memory is pre-programmed by your software into a LL element (also known as a descriptor) in local memory. Each element (called a data element) in the LL structure (called a transfer list) can transfer up to 4 GB of data.

You enable LL operation for a channel, by setting the LLE field of the DMA_CH_CONTROL1_OFF_WRCH_i register to 1. You can enable LL mode independently for each channel. When you enable LL for more than one channel, then you must have a separate LL structure in local memory for each channel. Your application must produce the LL element structure in local memory as shown in the following figure. Normally, all of the elements are contiguous (one after the other) in memory, and each element has six DWORDs containing the information about the block of data to be transferred. You program the channel context registers (DMA_LLP_LOW_OFF_WRCH_i and DMA_LLP_HIGH_OFF_WRCH_i) with the location of where you have placed the LL element structure in local memory. The above description also applies to read channels with RDCH instead of WRCH.

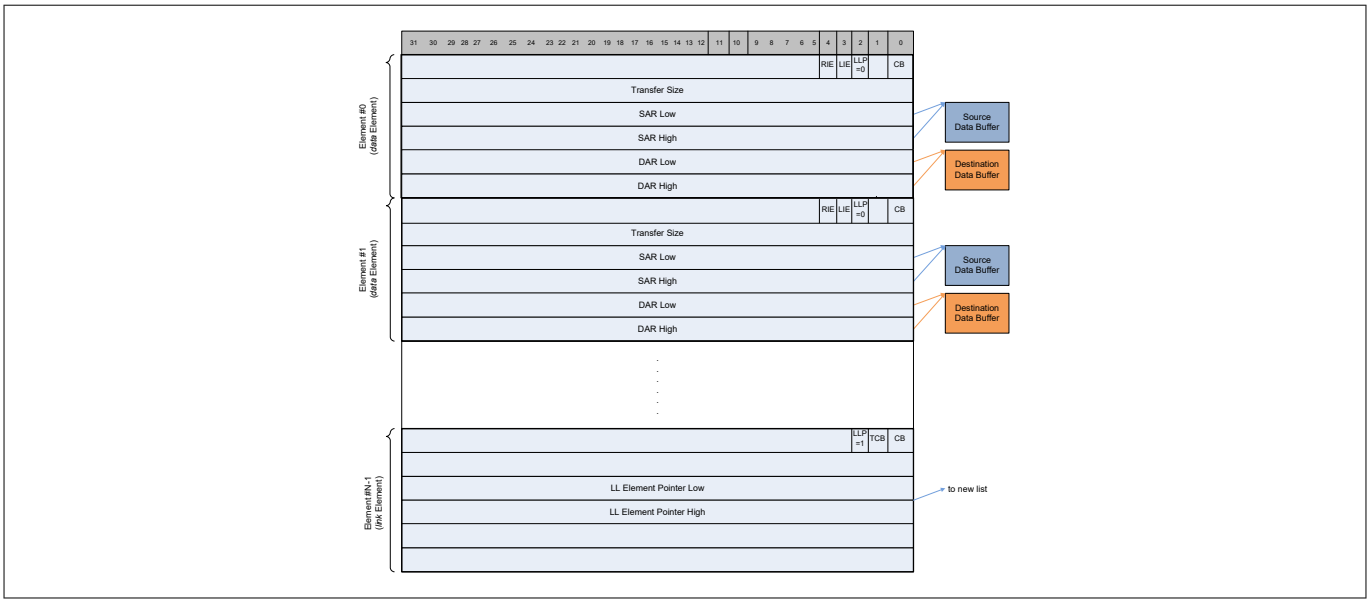


Figure 289 Linked list element/descriptor structure in local memory with N elements

When you start the DMA transfer (by writing to the DMA Write Doorbell Register DMA_WRITE_DOORBELL_OFF or DMA Read Doorbell Register DMA_READ_DOORBELL_OFF), the DMA reads (consumes) each element from local memory, and loads the information (SAR, DAR, size, and so on) from that element into the channel context registers in the DMA. These channel context registers determine the operation of the channel that the DMA controller is currently servicing. The DMA then proceeds to transfer the block of data (as defined by the element), and when it is finished, reads the next element from local memory. Normally, all of the elements are contiguous (one after the other) in memory, with the starting address defined in the channel context DMA Linked List Pointer Low Register DMA_LLP_LOW_OFF_WRCH_i. When you want to jump in local memory to another element list (or recycle the consumed elements), then you set the LLP bit in the element (for example, link element #N-1 in the previous figure), specify the location of the next element structure using the LL Element Pointer DWORDs (as indicated in the previous figure), and, set TCB to 1 (for recycling) or to 0 (to jump to another list).

Relationship Between Element DWORDs and Channel Context Registers

Notice the similarity between a data element and the DMA Channel Context registers for each channel. Each element has six DWORDs. There are eight channel context registers (DWORDs) for a channel. The DMA loads the six element DWORDs into the following channel context:

- CB, LLP, LIE, and RIE fields of the DMA Channel Control 1 register
- DMA Transfer Size
- DMA SAR Low and DMA SAR High
- DMA DAR Low and DMA DAR High

The definitions of the element DWORD bit fields are the same as the DMA Channel Context registers, with the exception of the LIE and RIE bits. The LIE and RIE bits in a LL element, only enable the done interrupt. In non-LL mode, the RIE and LIE bits (in the channel context registers) enable the done and abort interrupts.

When the LLP field of the first DWORD in the element is set to 1, the element is a link element. The DMA only loads the following information into the channel context registers:

- CB, TCB, and LLP bits of the first DWORD. The LIE and RIE bits are not defined in a link element
- LL Element Pointer (3rd and 4th DWORDs) into the DMA Linked List Pointer registers (DMA_LLP_LOW_OFF_WRCH_i and DMA_LLP_HIGH_OFF_WRCH_i)

Linked List Operation

- When enabling the linked list operation (DMA_CH_CONTROL1_OFF_WRCH_i.LLE = 1), the DMA overwrites the following DMA channel context registers with information from linked list data elements:
  - The CB, LLP, LIE, and RIE fields of the DMA Channel Control 1 register
  - DMA Transfer Size
  - DMA SAR Low and High
  - DMA DAR Low and High
- The structure of a link element is different to that of the data element. A data element has no TCB field. A link element has no LIE or RIE fields. It has no SAR, DAR, or Transfer Size DWORDs, but has LL Element Pointer DWORDs instead of the SAR DWORDs
- The LIE and RIE bits in an LL element, only enable the done interrupt. In non-LL mode, the RIE and LIE bits enable the done and abort interrupts

LL operation overview

In this section, a normal LL operation called recycling[18)] is described in detail. The process is described for a write channel, but it is possible to have duplicate processes running in parallel for other write or read channels. In this process interrupts are used to trigger the producer (software) to recycle elements. Typically a Watermark interrupt (positioned near the middle of the transfer list (TL)) and an Empty interrupt (in the last *data* element) are used by setting LIE in these two elements to '1'. The steps in this process are:

1. Software creates a LL element structure called a TL in local memory consisting of N-1 *data* elements and one *link* element:
   - The link elements "LL Element Pointer Low/High DWORDs" in the previous figure are programmed to point back to the beginning of the first data element
   - One of the data elements (near the middle of the TL) is programmed to generate a "Watermark" interrupt
   - The last data element is programmed to generate an "Empty" interrupt

2. Software programs the DMA with the location of the TL. This is done by writing to the DMA Linked List Pointer Low Register (DMA_LLP_LOW_OFF_WRCH_i) and DMA Linked List Pointer High Register (DMA_LLP_HIGH_OFF_WRCH_i)

3. Software starts the DMA process by ringing the write channel Doorbell. This is done by writing to the DMA Write Doorbell Register (DMA_WRITE_DOORBELL_OFF)

   The solid purple loop L1 in the following figure corresponds to steps 1-3

4. The DMA reads (consumes) each data element from local memory, and if the CB and CCS bits match it loads the information (SAR, DAR, size, and so on) from that data element into the channel context registers. These context registers determine the operation of the channel that the DMA is currently servicing. The DMA executes the check for the CB/CCS match after it has evaluated TCB and toggled CCS. The DMA always loads link elements into the channel context

   ***Note**: If the DMA stops on a link element, then your software must set the DMA Linked List Pointer registers, and restart the DMA process by ringing the doorbell.*

18 With recycling, the same LL structure is used over and over again. Another mode of operation would be to jump to a new LL element structure when the current one is consumed.

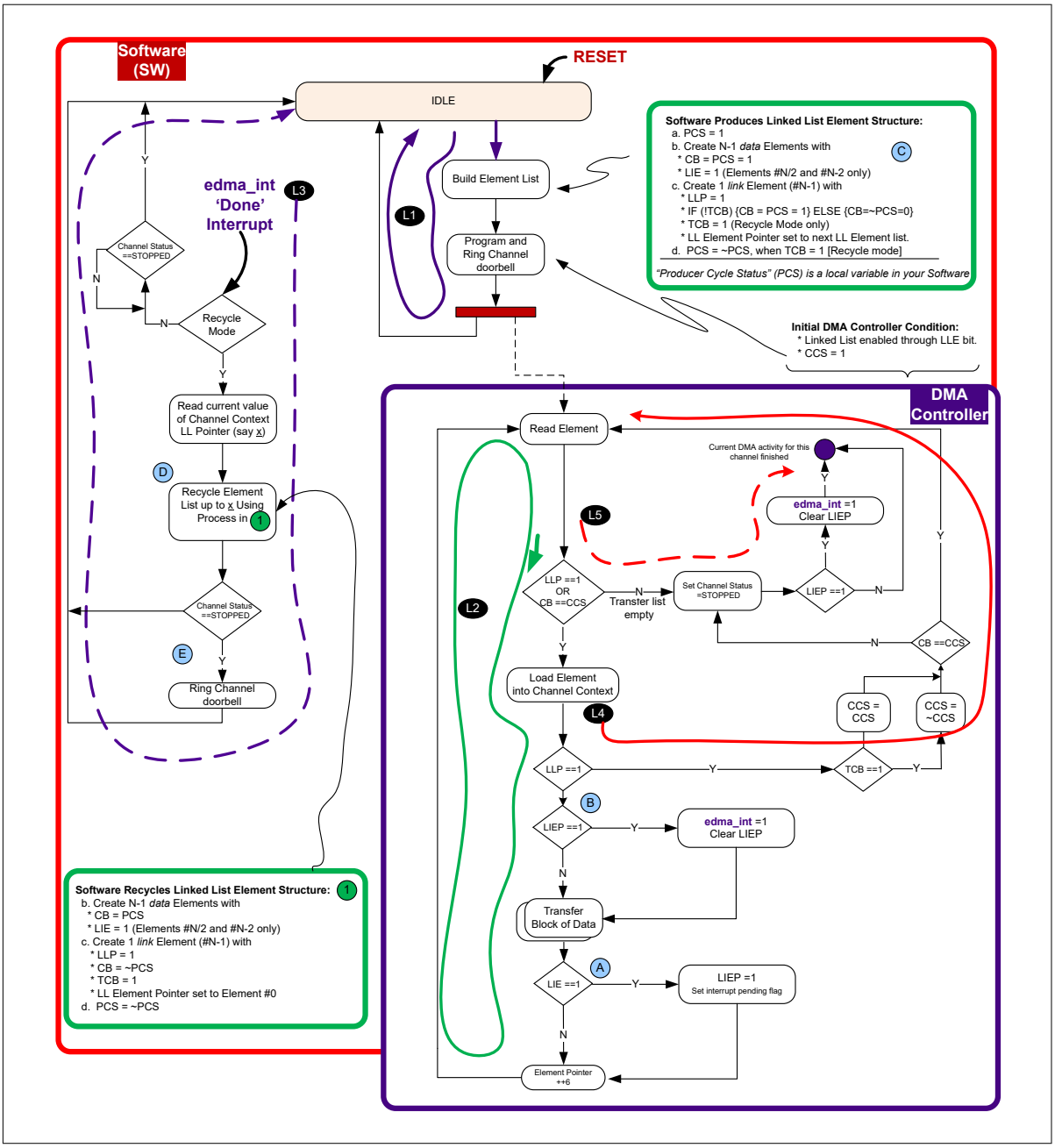


Figure 290 Linked list flow for producer and consumer

5. The DMA proceeds to transfer the block of data (as defined by the element), and when it is finished it reads the next element from local memory.

The solid green loop L2 in the figure above corresponds to steps 4-5

6. The last element in the list (called a link element, and indicated by having its LLP field set to 1) is not used to transfer a block of data.

- It causes the DMA to effectively repeat the task in step 4
- Its TCB is typically set to 1. This causes the DMA to toggle its CCS bit, and the software to toggle its PCS bit. The software and DMA use the "PCS-CCS-CB-TCB" producer-consumer synchronization mechanism to ensure that:
  - Software does not recycle elements that have not yet being consumed by the DMA
  - DMA correctly recognizes and consumes recycled elements

The solid red loop L4 in the figure above corresponds to step 6

7. Upon reception of the interrupt (mentioned in step 1), the software starts to recycle the TL
   - Software reprograms each data element with new DMA transfer information
   - The software and DMA use the “PCS-CCS-CB-TCB” producer-consumer synchronization mechanism described later in this section

   The dashed purple loop L3 in the figure above corresponds to step 7
8. At some point, the software wants to terminate the complete DMA process by not recycling any more elements. The DMA recognizes this condition when CB != CCS and sets the channel status to stopped

   The dashed red loop L5 in the figure above corresponds to step 8

Using interrupts for linked list producer-consumer synchronization

When the DMA is finished with an element, the DMA checks the LIE bit, before reading the next LL element. For more information, see circled step A in the figure above. When the LIE is set, then the DMA does not assert the “done” interrupt immediately, but sets an internal interrupt pending flag (LIEP). It asserts the actual done interrupt after the next data element has been read from system memory. For more information, see circled step B in the figure above. This automatic internal process of delaying the interrupt avoids a race condition between the DMA and software when the LIE bit is set in an element.

Your software should set the “Watermark” interrupt early to schedule the recycling of the consumed elements. Placing the Watermark interrupt too far down in the list, combined with a slow element recycling process in your application; ensures that the DMA returns to the start of the LL before your software has recycled it. The DMA channel STOPs, and you have to restart it by writing to its Doorbell. Typically a “Watermark interrupt” (positioned near the middle of the TL) and an Empty interrupt (in the last data element) are used by setting LIE in these two elements to 1. Upon reception of the interrupt, the software starts to recycle the TL.

PCS-CCS-CB-TCB producer-consumer synchronization

The software and DMA use the “PCS-CCS-CB-TCB” producer-consumer synchronization mechanism to ensure that software does not recycle elements that have not yet being consumed by the DMA, and that the DMA correctly recognizes and consumes recycled elements. This process, which is shared between the DMA and the software, is illustrated through an example producer-consumer flow in the figure below.

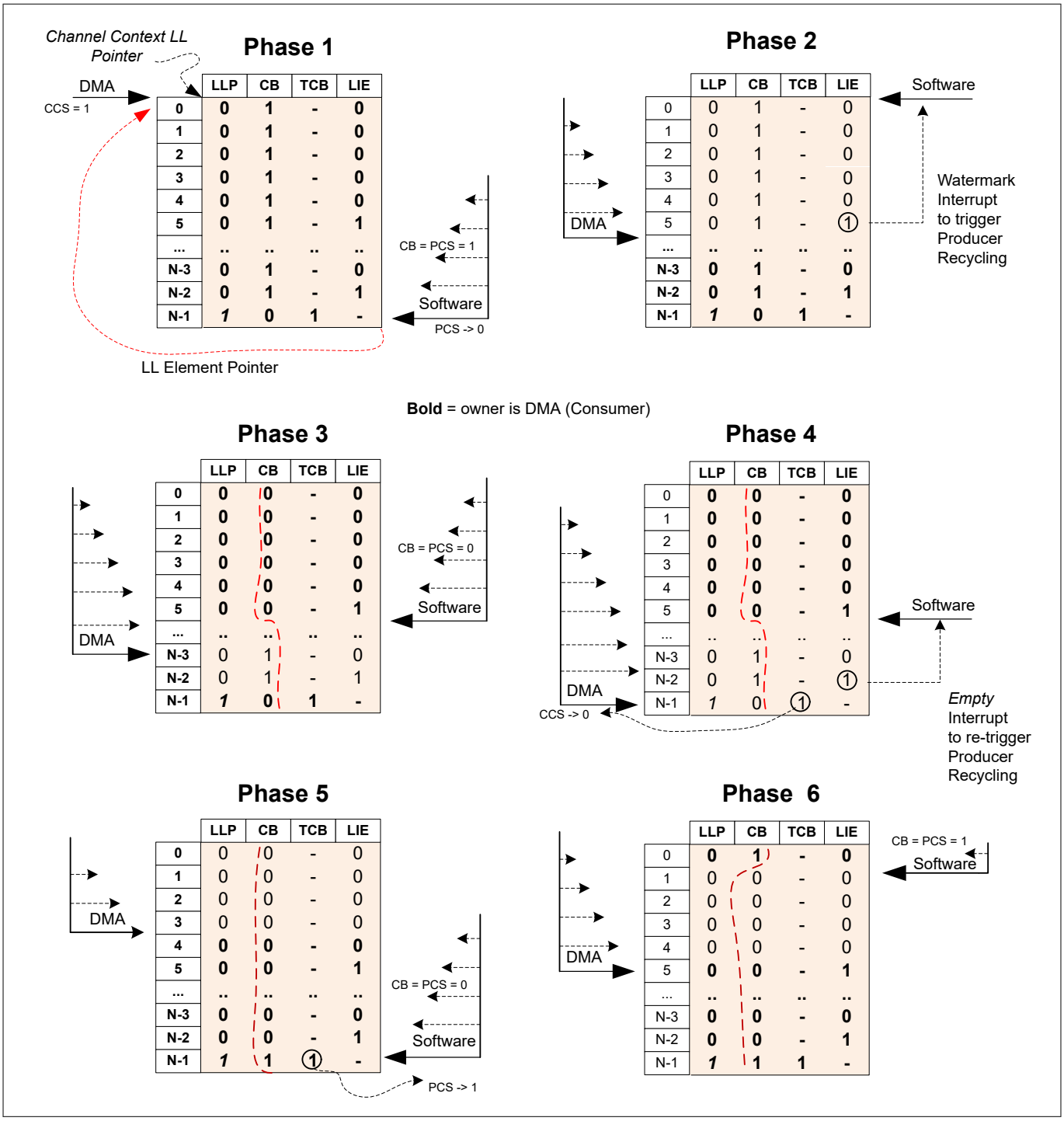


Figure 291 Example producer (software) - consumer (DMA) synchronization flow

Before looking at the example in detail, it is useful to note that the DMA performs the following two tests as part of this process:

- Consumer-Owned Element, or Transfer List (TL) Empty Test:

  The solid red loop L4 in the figure Linked List Flow for Producer and Consumer corresponds to this test. The Cycle Bit (CB) of DMA Channel Control 1 Register (DMA_CH_CONTROL1_OFF_WRCH_i) (which is loaded from the LL element) is tested against the DMA Consumer Cycle State (CCS) bit of the same register. When CB = CCS, the element is owned by the consumer (DMA) and the data transfer is executed. When CB != CCS, the TL is empty and the DMA sets channel Status (CS) in DMA Channel Control 1 Register (DMA_CH_CONTROL1_OFF_WRCH_i) to “Stopped”
- Toggle Cycle Bit (TCB) Test:

The Dashed Red Loop Marked L5 in the figure Linked List Flow for Producer and Consumer corresponds to this test. When the Toggle Cycle Bit (TCB) of the DMA Channel Control 1 Register (DMA_CH_CONTROL1_OFF_WRCH_i) (which is loaded from the LL element) is set to 1, the DMA toggles the CCS bit of the DMA Channel Control 1 Register (DMA_CH_CONTROL1_OFF_WRCH_i) to be ~ CB. The producer (software) also performs a similar action on its local variable “Producer Cycle Status” (PCS), as it sets PCS =~ PCS. The TCB field is only valid in the last LL element (the *link* element). Toggling the PCS and CCS flags in this manner synchronizes the consumer to the producer

The example in the previous figure has the following six phases:

1. The software builds the transfer list for the write (or read) channel, according to the steps outlined in the circled step C in the figure Linked List Flow for Producer and Consumer. For each element, CB = PCS = CCS = 1. The exception to this is the link element where CB is set to the next iteration value because the link element is always loaded into the channel context
   a. The link element points to the start of the TL and its TCB bit is 1. Software toggles its PCS variable
   b. Software writes the location of the TL into the channel context LL Pointer register
   c. Software rings the write channel Doorbell
2. The DMA starts processing elements. After processing element 5 (and fetching the next element), the DMA asserts a Watermark “done” interrupt to signal the software to start recycling elements
3. The software is recycling elements. It reads the channel context DMA Linked List Pointer Low Register (DMA_LLP_LOW_OFF_WRCH_i) and DMA Linked List Pointer High Register (DMA_LLP_HIGH_OFF_WRCH_i) and recycles elements up to this location[19] in the TL. It sets CB for each element to be PCS. The Dashed Purple Loop Marked L3 in figure 15 corresponds to this activity.

   The DMA continues processing (consuming) elements as before
4. The DMA then asserts an Empty “done” interrupt (from the LIE bit in the last data element at the end of TL that is set in the previous phase) to indicate to the software to start recycling elements again. The DMA reaches the end of the TL, and as the TCB is set to 1, it toggles its CCS bit to 0 (~ CB). For more information, see circled step D in the figure Linked List Flow for Producer and Consumer
5. The software reads the channel context DMA Linked List Pointer Low Register (DMA_LLP_LOW_OFF_WRCH_i) and DMA Linked List Pointer High Register (DMA_LLP_HIGH_OFF_WRCH_i) and recycles elements up to this location in the TL. It sets CB for each element to be PCS. As it recycles the link element, it detects that TCB is set to 1. It toggles its internal variable PCS to 0, and the initial TL has been fully recycled.

   The DMA starts processing the recycled elements
6. The DMA continues processing (consuming) elements as before. Software does not have to ring the Doorbell, as the first element is already recycled before the DMA started processing it.

   The software might continue recycling from the top of the list, if the DMA had already returned to the top of the list when the software read the channel context DMA Linked List Pointer Low Register (DMA_LLP_LOW_OFF_WRCH_i) and DMA Linked List Pointer High Register (DMA_LLP_HIGH_OFF_WRCH_i) in step 5. If not, it starts recycling again when the DMA processes element 4 and the DMA asserts a Watermark “done” interrupt

Element recycling

When your software is recycling elements, it should program the first DWORD (control bits) in the figure Linked List Element/Descriptor Structure in Local Memory with N Element only after the other DWORDs in the same element have been programmed. In addition, it should set the CB bit (to ~PCS) within this DWORD only after the other bits e.g. RIE in the same DWORD have been set. This programming sequence avoids the DMA incorrectly fetching an element that software is currently recycling.

When the software is finished recycling it must check the channel context status (CS). When CS is “Stopped”, the software must ring the channel Doorbell again. For more information, see circled step E in the figure Linked List Flow for Producer and Consumer. If your element recycling process is fast enough, the DMA seamlessly moves from processing the last old element to the first new element without the need for the software to ring the channel Doorbell again. To eliminate the possibility of a race condition between these two actions (read and

[19] If the consumer has moved on to another element, this is not a problem. These newly consumed elements are recycled in the next recycling phase.

write), you should first confirm the presence of the “done” interrupt bit in the DMA Write Interrupt Status Register (DMA_WRITE_INT_STATUS_OFF) or DMA Read Interrupt Status Register (DMA_READ_INT_STATUS_OFF) and also check the DMA Linked List Pointer Low Register (DMA_LLP_LOW_OFF_WRCH_i) to confirm that it is pointing to the correct location.

Link down recovery

During normal operation when the link goes down, the controller generates a “link reset“ interrupt (LNK_STAT.LINKRQRST) to IR. To determine the progress of the linked list operation when the link went down, you can read the contents of the channel context DMA Linked List Pointer Low Register (DMA_LLP_LOW_OFF_WRCH_i) and DMA Linked List Pointer High Register (DMA_LLP_HIGH_OFF_WRCH_i).

19.3.6.2 Using DMA

Source and destination address registers (SAR, DAR)

The DMA channel context SAR and DAR registers (DMA_SAR_LOW_OFF_WRCH_i[20] / DMA_SAR_HIGH_OFF_WRCH_i /DMA_DAR_LOW_OFF_WRCH_i / DMA_DAR_HIGH_OFF_WRCH_i) provide support for remote-to-local, and local-to-remote PCIe address mapping. You program the start of the local and remote data buffers using these registers, and the DMA increments the SAR and DAR as the DMA transfer progresses. For a write transfer, the SAR is the address of the local memory, and the DAR is the address of the remote memory, as shown in the following figure.

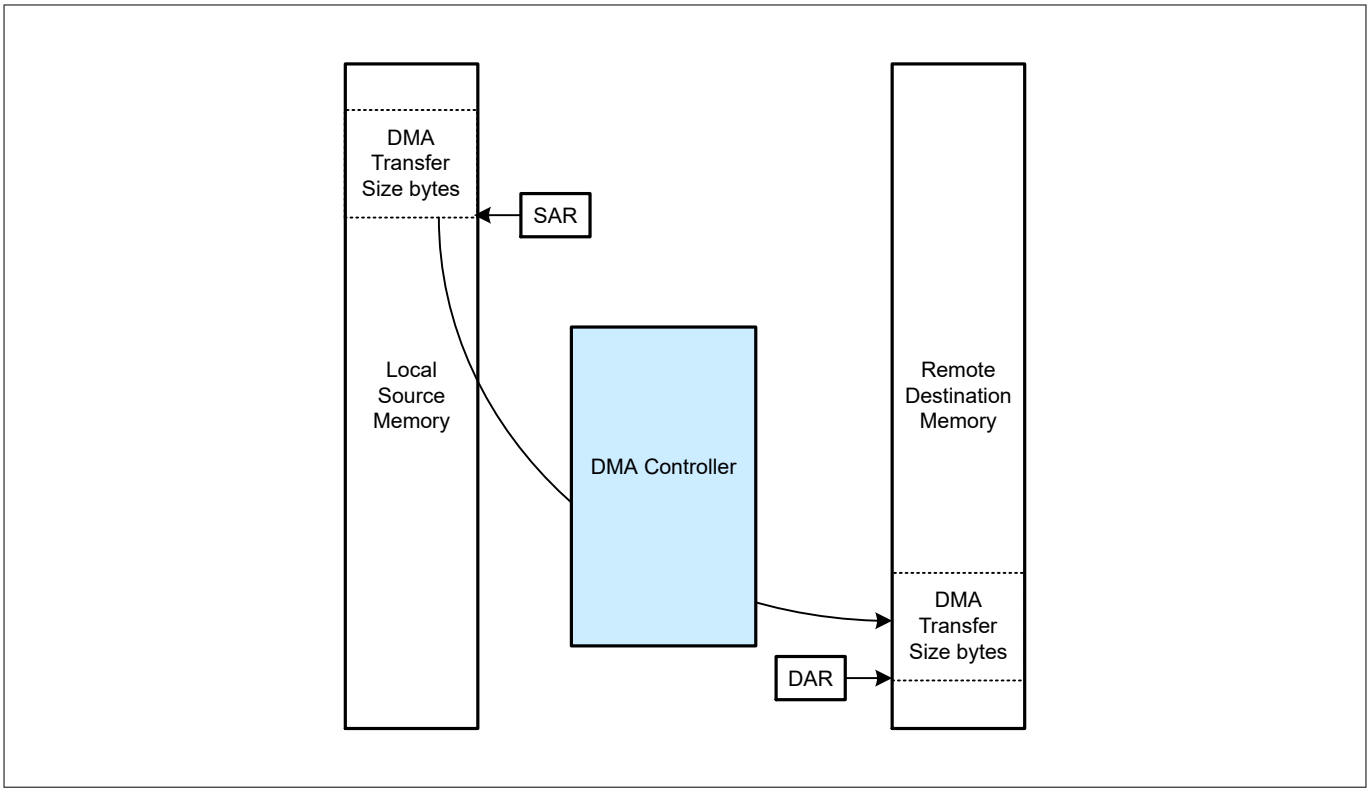


Figure 292 Write transfer: SAR and DAR for write channel 0

For a read transfer, the SAR is the address of the remote memory, and the DAR is the address of the local memory, as shown in the following figure.

[20] Replace WRCH with RDCH for a read channel.

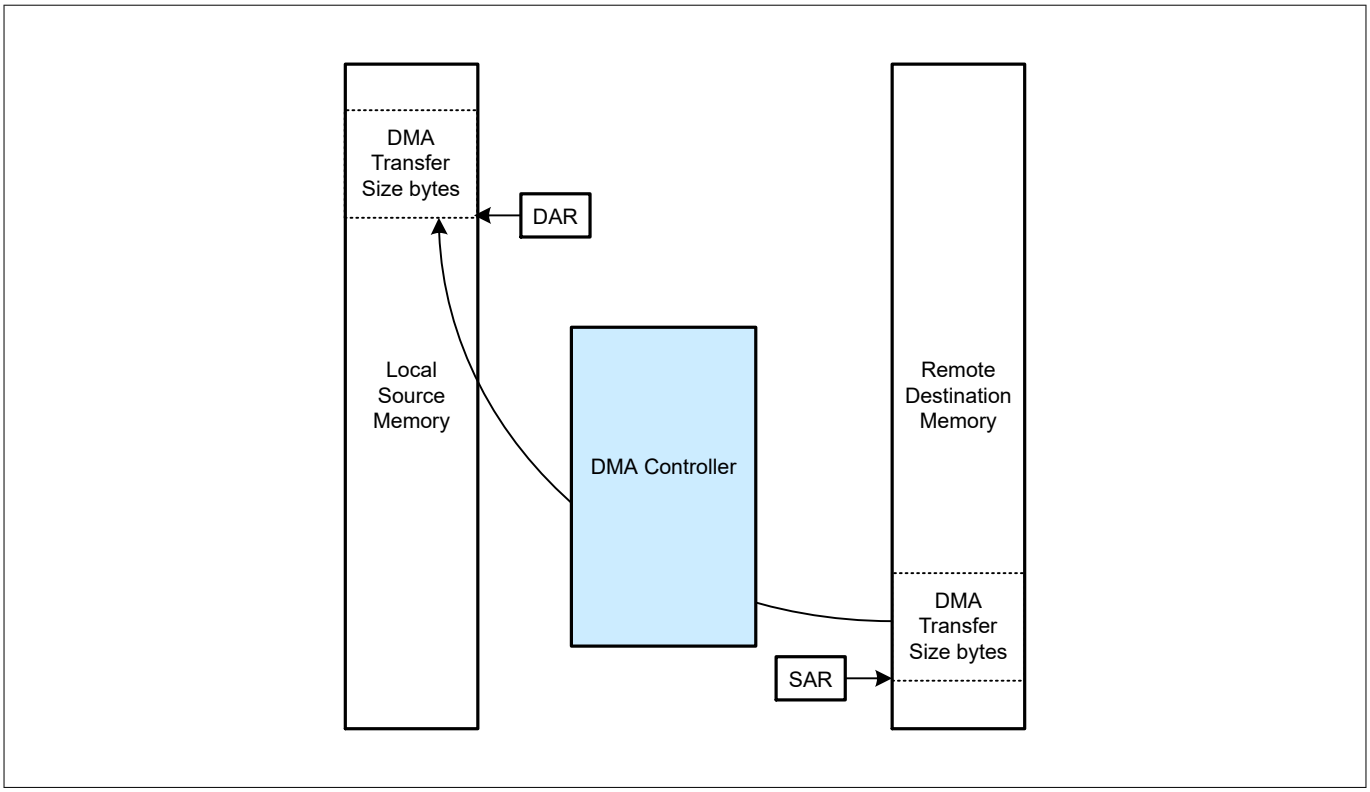


Figure 293 Read transfer: SAR and DAR for read channel 0

The presence of the DMA controller does not affect:

- Normal filtering rules for inbound TLPs (for example, BAR checking in an Endpoint)
- The operation of any internal or external address translation

When the iATU should not translate outbound requests that are generated by the internal DMA module, it is necessary to implement one of the following approaches:

- Ensuring that the combination of DMA channel programming and iATU control register programming causes no translation of DMA traffic to be done in the iATU by avoiding using the same address window
- Activating the DMA bypass mode to allow request TLPs which are initiated by the DMA controller to pass through the iATU untranslated. It is possible to activate DMA bypass mode by setting the DMA Bypass field of the iATU Control 2 Register (IATU_REGION_CTRL_2_OFF_OUTBOUND_i)

DMA transfer size register

You program the DMA transfer size using the DMA Transfer Size Register (DMA_TRANSFER_SIZE_OFF_WRCH_ior DMA_TRANSFER_SIZE_OFF_RDCH_i). The maximum DMA transfer size is 4 GB, and the minimum transfer size is one byte (1_H). The DMA decrements the value in this register as the DMA transfer progresses. When all bytes are successfully transferred, the value in this register is zero. In LL mode, the DMA overwrites this register with the corresponding dword of the LL element. You can read this register to monitor the transfer progress. However, in some scenarios there is a delay before the controller updates this register. For example, when less than three channels are doorbelled, this register is only updated after a descriptor finishes (LL mode), or the transfer ends (non-LL mode).

Starting the DMA transfer

After you program the DMA controller registers (including writing to the DMA Read Engine Enable or DMA Write Engine Enable register), you start a DMA transfer by writing the channel number to the Doorbell Number field of the DMA Write Doorbell Register (DMA_WRITE_DOORBELL_OFF) or DMA Read Doorbell Register (DMA_WRITE_DOORBELL_OFF). You can program and start both a read and a write transfer at the same time. The DMA supports full duplex operation, processing read and write transfers at the same time and in parallel

with normal (non-DMA) traffic. You can program and start any number of channels in the DMA controller sequentially or simultaneously.

- Sequentially: Configuring and starting a channel, then configuring and start another channel
- Simultaneously: Configuring multiple channels, then start multiple channels

***Note**: You must not write to any of the context registers for a particular channel after you start the channel by writing the channel number to the Doorbell Number field of the DMA Write Doorbell Register (DMA_WRITE_DOORBELL_OFF) or DMA Read Doorbell Register (DMA_READ_DOORBELL_OFF).*

Detecting the end of the DMA transfer

If a DMA transfer proceeds without any errors, it stops automatically when finished.

Detecting end of transfer without errors

The normal end of a DMA transfer is detected by any of the following methods:

- Done local interrupt asserted
- Done remote interrupt (IMWr) received
- Channel status field of the Channel Control 1 register is Stopped and the DMA Transfer Size register is 0_H
- Polling of the DMA Write Interrupt Status Register (DMA_WRITE_INT_STATUS_OFF) or DMA Read Interrupt Status Register (DMA_WRITE_INT_STATUS_OFF)

Detecting end of transfer with errors

The abnormal end of a DMA transfer is detected by any of the following methods:

- Abort local interrupt asserted
- Abort remote interrupt (IMWr) received
- Polling of the DMA Write Interrupt Status Register (DMA_WRITE_INT_STATUS_OFF) or DMA Read Interrupt Status Register (DMA_WRITE_INT_STATUS_OFF)
- Channel Status field of the Channel Control 1 register is halted. When the DMA controller detects an error, it forces the DMA to stop issuing requests for the channel. It also sets the channel status field of the Channel Control 1 register to halted, generates an abort interrupt (if enabled), and sends an abort IMWr (if enabled). The DMA Transfer Size register indicates the remaining number bytes to be transferred, except when there is an AXI write error during a DMA read transfer
- Channel Status field of the Channel Control 1 register is stopped and the DMA transfer size register is not 0_H. You have prematurely stopped this channel as described in the next section

Stopping the DMA transfer (software stop)

It is possible to manually abort (stop) the DMA transfer by writing the channel number to the Doorbell Number field and writing 1 to the Stop field in DMA_WRITE_DOORBELL_OFF. This causes the DMA to:

- Place the channel in a stopped state

  The channel Status field in DMA_CH_CONTROL1_OFF_WRCH_i is stopped and the value in DMA_TRANSFER_SIZE_OFF_WRCH_i is not 0_H
- Wait for all outstanding pending transactions
- Assert the abort interrupt (if it is enabled) in DMA_WRITE_INT_STATUS_OFF

You should not use the stop feature in operational mode. It is only a debug feature and has associated hazards

- Before setting the stop bit:
  - The user must read the channel status field (CS) of the DMA Channel Control 1 register to ensure that the corresponding channel is running (transferring data)
  - To eliminate the possibility of a race condition between these two actions (read and write), the user should confirm the presence of the abort bit in the DMA Write Interrupt Status Register

(DMA_WRITE_INT_STATUS_OFF) or DMA Read Interrupt Status Register (DMA_READ_INT_STATUS_OFF) and check the status of any fatal error if some other event has occurred

- After a Stop event, it is not possible to seamlessly resume the transfer again because the DMA does not resume operation exactly from the point it stopped. Therefore, it is necessary to set up and start the complete channel context again, including the:
  - Linked list pointer
  - Linked list structure in memory
  - Related PCS and CCS values
  - When operating in non-linked list mode, channel context registers mus be set up

TLP generator header control

You must program bits [31:12] of the DMA Channel Control 1 Register (DMA_CH_CONTROL1_OFF_WRCH_i/ DMA_CH_CONTROL1_OFF_RDCH_i), so that the DMA TLP generator correctly sets the TC, RO, NS, AT, and FN TLP header fields for any TLPs that it generates.

Programming examples

This section provides a programming and operation flow for each of the following scenarios:

- 1 MB Write Transfer On Write Channel #0 initiated by Local CPU (Non LL mode)
- 1 MB Read Transfer On Read Channel #0 initiated by Local CPU (Non LL mode)
- 1 MB Write Transfer On Write Channel #3 initiated by Remote CPU (Non LL mode)
- 1 MB Read Transfer On Read Channel #3 initiated by Remote CPU (Non LL mode)
- Linked List Mode Programming Example (Write Channel #0)

***Note**: You must enable the DMA before you start a DMA channel. Therefore you must write '1' to DMA Read Engine Enable or DMA Write Engine Enable register before you write '1' to the DMA Read Doorbell or DMA Write Doorbell register.*

1 MB write transfer on write channel #0 initiated by local CPU (non LL mode)

In this example, the IMWr generation is disabled, as the local CPU initiates the DMA transfer. The local CPU is interrupted using the corresponding interrupts. The SAR is the address of the local memory, and the DAR is the address of the remote memory, as shown in the following figure. The following table provides the programming information for this example transfer.

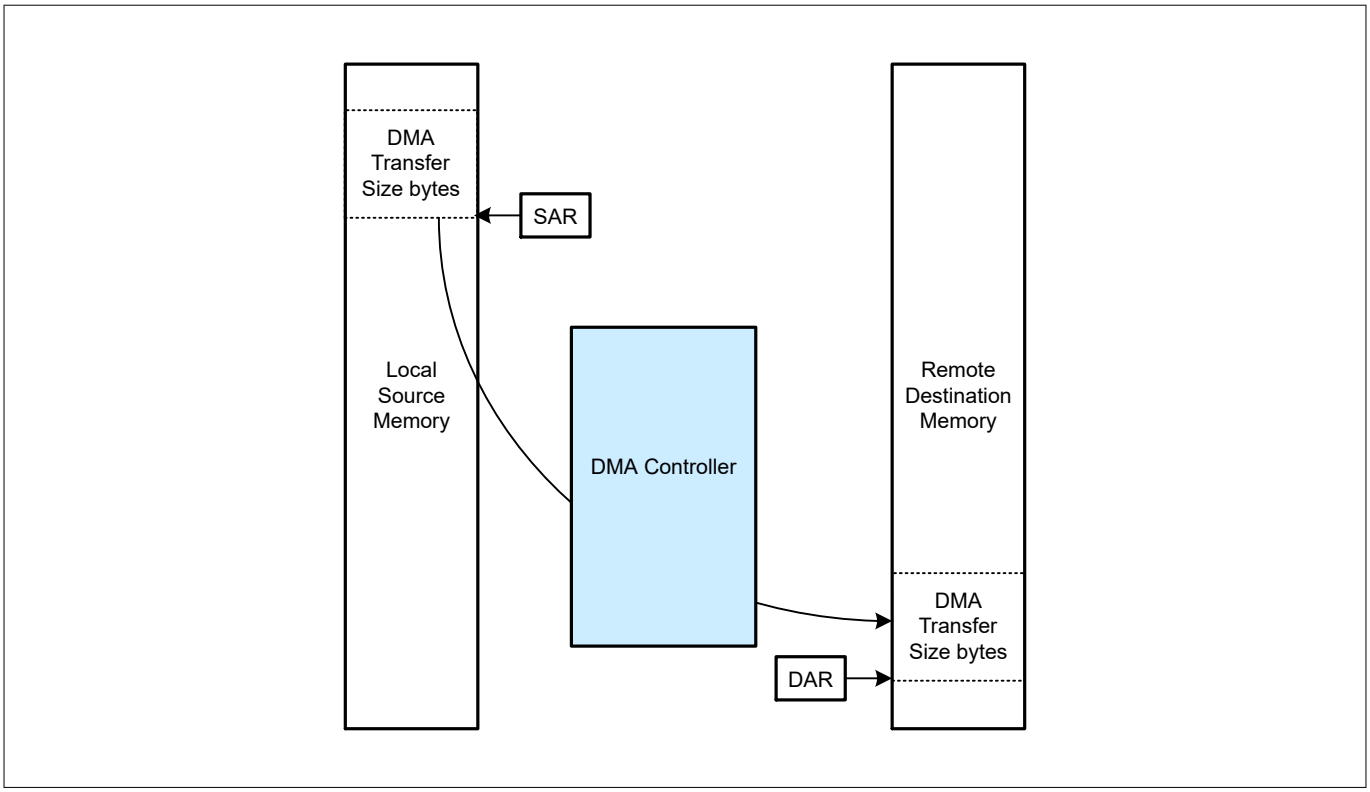


Figure 294 1 MB Write DMA Transfer from 0xBEEF_BEE0 to 0xCAFE_CAF0

Table 836 1 MB write DMA transfer from 0xBEEF_BEE0 to 0xCAFE_CAF0

Channel register	Name	Value
DMA_WRITE_ENGINE_EN_OFF	DMA Write Engine Enable You must not write 0 to this register. Even temporarily writing 0 to this register resets the DMA logic. For more information, see DMA Write Engine Enable Register (DMA_WRITE_ENGINE_EN_OFF).	1_H
DMA_WRITE_INT_MASK_OFF	DMA Write Interrupt Mask	0_H
DMA_CH_CONTROL1_OFF_WRCH_i	DMA Channel Control 1 register • Local Interrupt Enable (LIE) = 1 • Remote Interrupt Enable (RIE) = 0 • AT, RO, NS, TC, function number = 0	04000008_H
DMA_TRANSFER_SIZE_OFF_WRCH_i	DMA Transfer Size	00100000_H
DMA_SAR_LOW_OFF_WRCH_i	DMA SAR Low	$BEEF_BEE0_H$
DMA_SAR_HIGH_OFF_WRCH_i	DMA SAR High	0000_0000_H
DMA_DAR_LOW_OFF_WRCH_i	DMA DAR Low	$CAFE_CAF0_H$
DMA_DAR_HIGH_OFF_WRCH_i	DMA DAR High	0000_0000_H
DMA_WRITE_DOORBELL_OFF	DMA Write Doorbell	0_H

1 MB read transfer on read channel #0 initiated by local CPU (non LL mode)

In this example, the IMWr generation is disabled, as the local CPU initiates the DMA transfer. The local CPU is interrupted using the corresponding interrupts to IR. The SAR is the address of the remote memory, and the

DAR is the address of the local memory, as shown in the following figure. The following table provides the programming information for this example transfer.

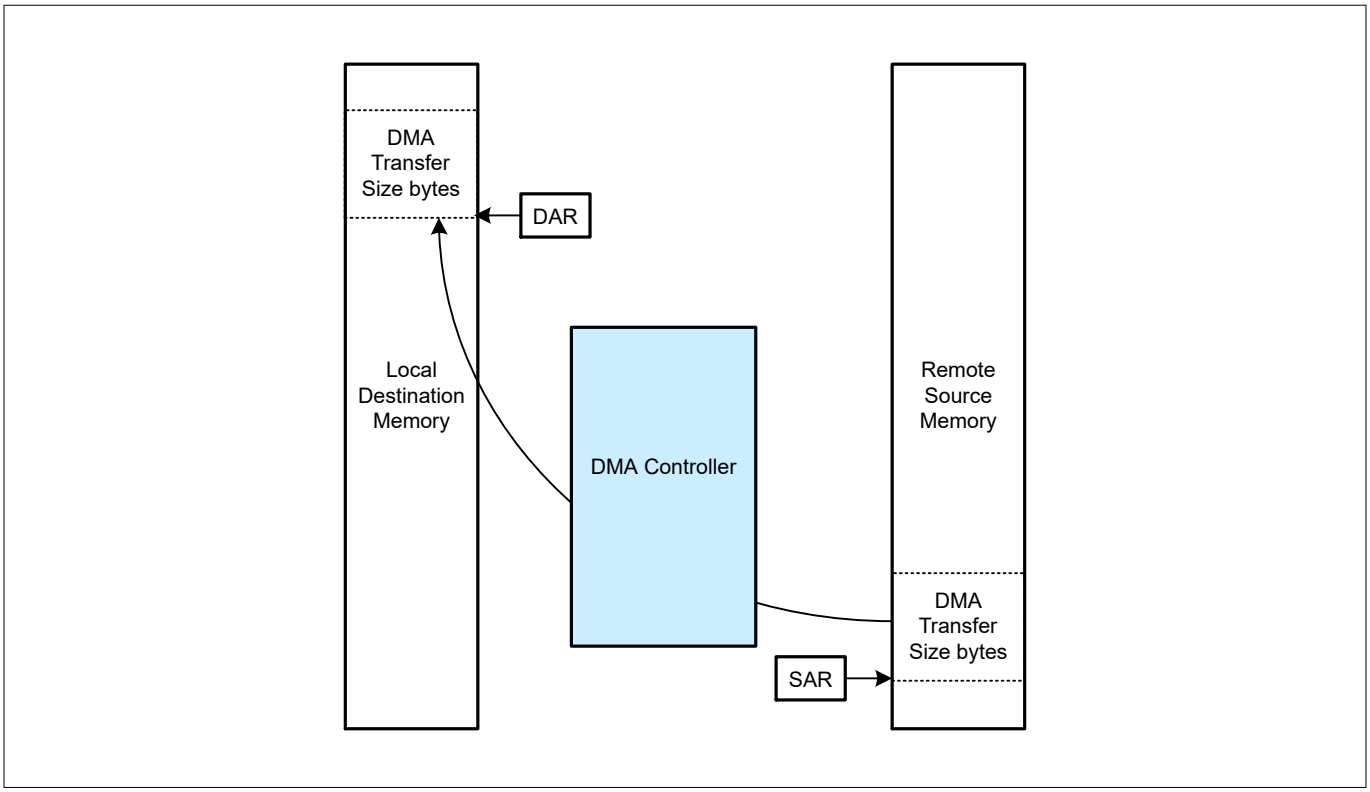


Figure 295 1 MB Read DMA Transfer from 0xBEEF_BEE0 to 0xCAFE_CAF0

Table 837 DMA register retup

Channel register	Name	Value
DMA_READ_ENGINE_EN_OFF	DMA Read Engine Enable You must not write 0 to this register. Even temporarily writing 0 to this register resets the DMA logic. For more information, see DMA Read Engine Enable Register (DMA_READ_ENGINE_EN_OFF).	1_H
DMA_READ_INT_MASK_OFF	DMA Read Interrupt Mask	0_H
DMA_CH_CONTROL1_OFF_RDCH_i	DMA Channel Control 1 register • Local Interrupt Enable (LIE) = 1 • Remote Interrupt Enable (RIE) = 0 • AT, RO, NS, TC, Function Number = 0	04000008_H
DMA_TRANSFER_SIZE_OFF_RDCH_i	DMA Transfer Size	00100000_H
DMA_SAR_LOW_OFF_RDCH_i	DMA SAR Low	$BEEF_BEE0_H$
DMA_SAR_HIGH_OFF_RDCH_i	DMA SAR High	0000_0000_H
DMA_DAR_LOW_OFF_RDCH_i	DMA DAR Low	$CAFE_CAF0_H$
DMA_DAR_HIGH_OFF_RDCH_i	DMA DAR High	0000_0000_H
DMA_READ_DOORBELL_OFF	DMA Read Doorbell	0_H

1 MB write transfer on write channel #3 initiated by remote CPU (non LL mode)

In this example, the local interrupt generation is disabled, as the remote CPU initiates the DMA transfer. The remote CPU is interrupted using an IMWr. The SAR is the address of the local memory, and the DAR is the address of the remote memory, as shown in the following figure. The following table provides the programming information for this example transfer.

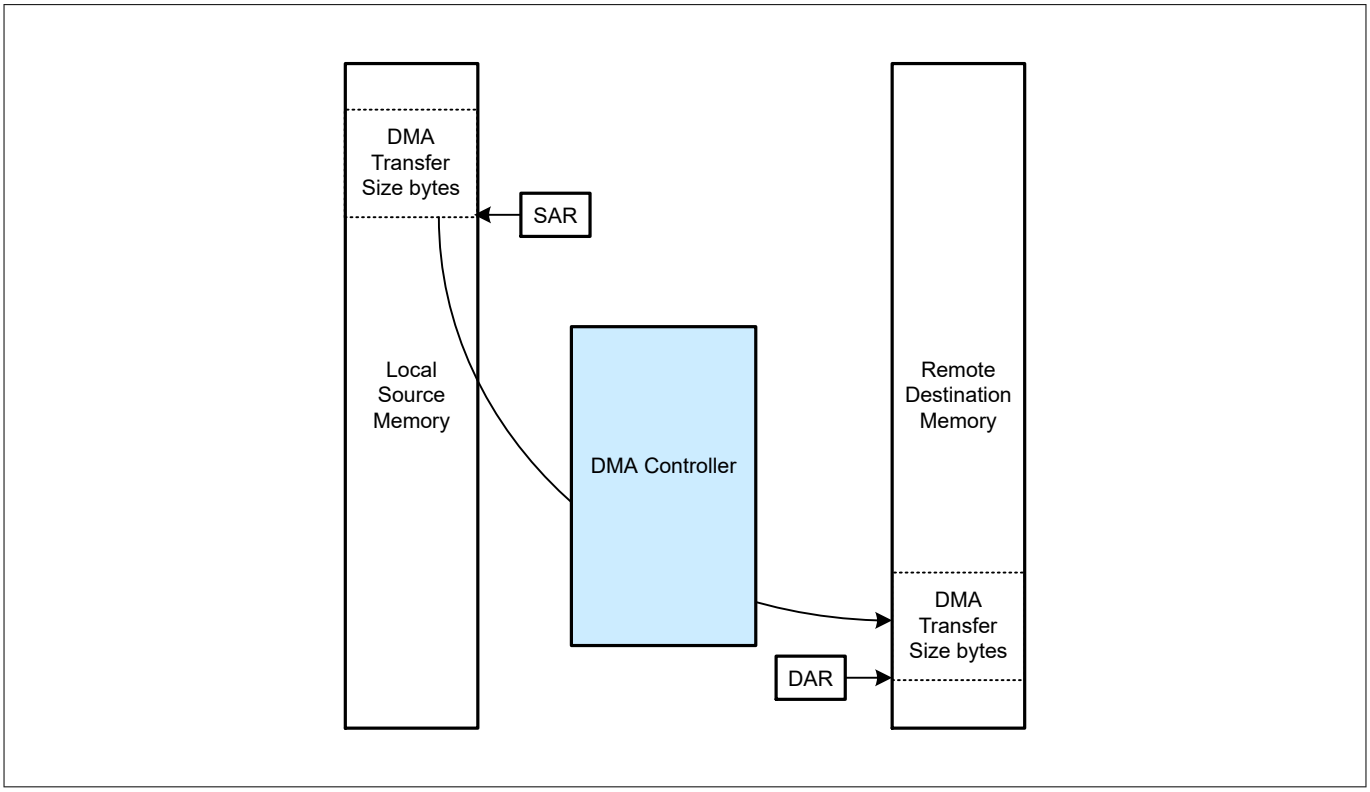


Figure 296 1 MB write DMA transfer from 0xBEEF_BEE0 to 0xCAFE_CAF0

Table 838 DMA register setup

Channel register	Name	Value
DMA_WRITE_ENGINE_EN_OFF	DMA Write Engine Enable You must not write 0 to this register. Even temporarily writing 0 to this register resets the DMA logic. For more information, see DMA Write Engine Enable Register (DMA_WRITE_ENGINE_EN_OFF).	1_H
DMA_WRITE_DONE_IMWR_LOW_OFF DMA_WRITE_DONE_IMWR_HIGH_OFF	DMA Write Done IMWr Address Low and High	your IMWr Address #1
DMA_WRITE_ABORT_IMWR_LOW_OFF DMA_WRITE_ABORT_IMWR_HIGH_OFF	DMA Write Abort IMWr Address Low and High	your IMWr Address #2
DMA_WRITE_CH23_IMWR_DATA_OFF	DMA Write Channel 3 IMWr Data	your IMWr Data
DMA_CH_CONTROL1_OFF_WRCH_i	DMA Channel Control 1 register • Local Interrupt Enable (LIE) = 0 • Remote Interrupt Enable (RIE) = 1 • AT, RO, NS, TC, Function Number = 0	04000010_H

(table continues...)

Table 838 (continued) DMA register setup

Channel register	Name	Value
DMA_TRANSFER_SIZE_OFF_WRCH_i	DMA Transfer Size	00100000_H
DMA_SAR_LOW_OFF_WRCH_i	DMA SAR Low	$BEEF_BEE0_H$
DMA_SAR_HIGH_OFF_WRCH_i	DMA SAR High	0000_0000_H
DMA_DAR_LOW_OFF_WRCH_i	DMA DAR Low	$CAFE_CAF0_H$
DMA_DAR_HIGH_OFF_WRCH_i	DMA DAR High	0000_0000_H
DMA_WRITE_DOORBELL_OFF	DMA Write Doorbell	3_H

1 MB read transfer on read channel #3 initiated by remote CPU (non LL mode)

In this example, the local interrupt generation is disabled, as the remote CPU initiates the DMA transfer. The remote CPU is interrupted using an IMWr. The SAR is the address of the remote memory, and the DAR is the address of the local memory, as shown in the following figure. The following table provides the programming information for this example transfer.

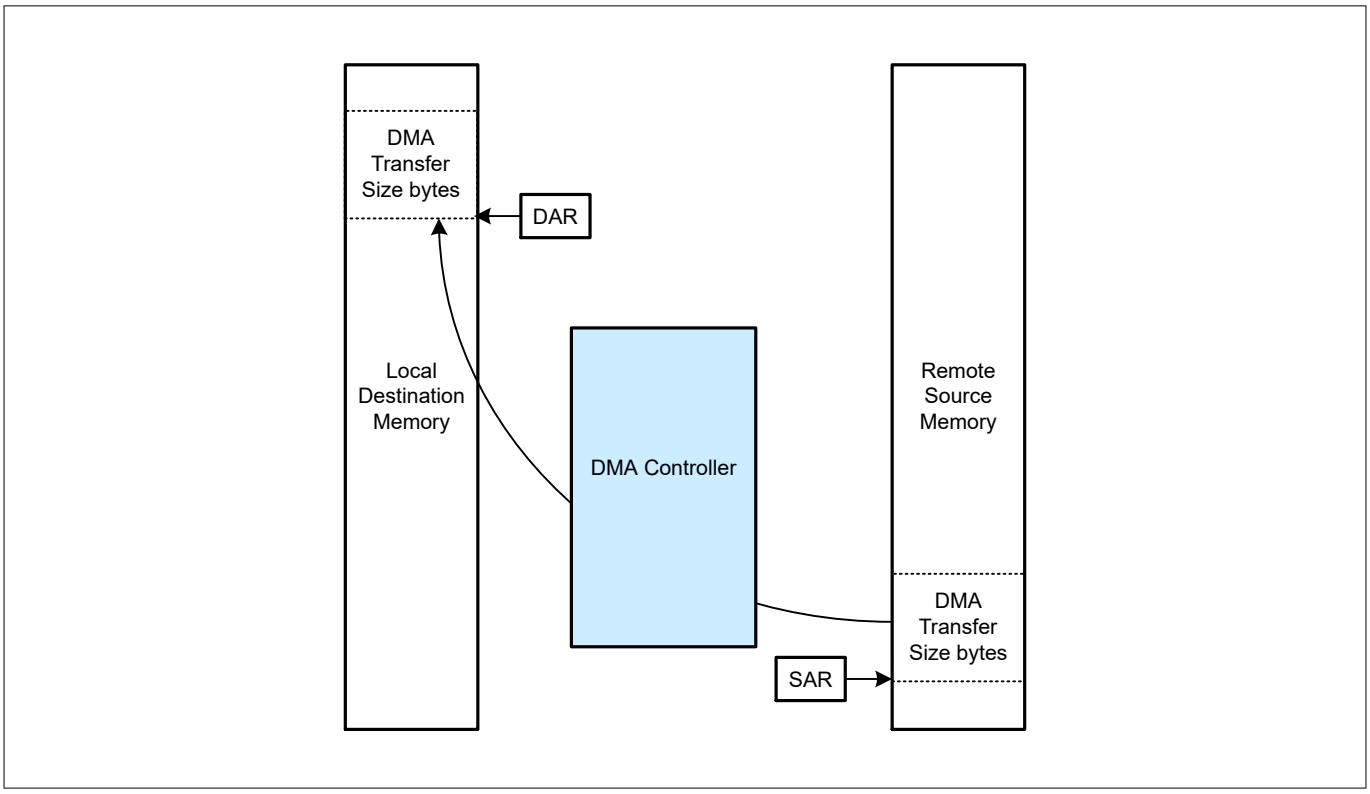


Figure 297 1 MB read DMA transfer from 0xBEEF_BEE0 to 0xCAFE_CAF0

Table 839 DMA register setup

Channel register	Name	Value
DMA_READ_ENGINE_EN_OFF	DMA Read Engine Enable You must not write 0 to this register. Even temporarily writing 0 to this register resets the DMA logic. For more information, see DMA Read Engine Enable Register (DMA_READ_ENGINE_EN_OFF).	1_H

(table continues...)

Table 839 (continued) DMA register setup

Channel register	Name	Value
DMA_READ_DONE_IMWR_LOW_OFF DMA_READ_DONE_IMWR_HIGH_OFF	DMA Read Done IMWr Address Low and high	your IMWr Address #1
DMA_READ_ABORT_IMWR_LOW_OFF DMA_READ_ABORT_IMWR_HIGH_OFF	DMA Read Abort IMWr Address Low and High	your IMWr Address #2
DMA_READ_CH23_IMWR_DATA_OFF	DMA Read Channel 3 IMWr Data	your IMWr Data
DMA_CH_CONTROL1_OFF_RDCH_i	DMA Channel Control 1 register • Local Interrupt Enable (LIE) = 0 • Remote Interrupt Enable (RIE) = 1 • AT, RO, NS, TC, Function Number = 0	04000010_H
DMA_TRANSFER_SIZE_OFF_RDCH_i	DMA Transfer Size	00100000_H
DMA_SAR_LOW_OFF_RDCH_i	DMA SAR Low	$BEEF_BEE0_H$
DMA_SAR_HIGH_OFF_RDCH_i	DMA SAR High	0000_0000_H
DMA_DAR_LOW_OFF_RDCH_i	DMA DAR Low	$CAFE_CAF0_H$
DMA_DAR_HIGH_OFF_RDCH_i	DMA DAR High	0000_0000_H
DMA_READ_DOORBELL_OFF	DMA Read Doorbell	3_H

Linked list mode programming example (write channel #0)

This write channel example has the following features:

- Recycled linked list (LL) of nine data elements[21] and one link element
- Transfer is initiated by local CPU
- Each data element corresponds to a 1 MB (0010_0000_H) write transfer
- The SAR for each element is separated by 2 MB (0020_0000_H). The DAR for each element is separated by 16 MB (0100_0000_H). The SAR is the address of the local memory, and the DAR is the address of the remote memory, as shown in the following figure
- The LL transfer list is located at address 0000_0200_H in local memory
- No remote interrupt is generated so the IMWr generation is disabled, as the local CPU initiates the DMA transfer. The local CPU is interrupted using the corresponding interrupts to IR
- There is a Watermark interrupt at element #5 and an Empty interrupt at element #8. Both of these interrupts are done interrupts (as opposed to abort interrupts)

You should familiarize yourself with the Linked List Flow for Producer and Consumer. These are the main steps for this process in this example:

- Step 1: Creating TL in local memory:
  - Creating the 10 elements according to the following table
    - For elements 0, 1, 2, 3, 4, 6, 7: RIE, LIE, LLP, CB = 0, 0, 0, 1
    - For element 5: RIE, LIE, LLP, CB = 0, 1, 0, 1
    - For element 8: RIE, LIE, LLP, CB = 0, 1, 0, 1
    - For element 9: LLP, TCB, CB = 1, 1, 0
- Step 2: Programming and starting DMA:
  - Configuring and starting the DMA registers as per DMA Register Setup
- Step 3: Waiting for done interrupt and recycle TL

[21] Also known as descriptors.

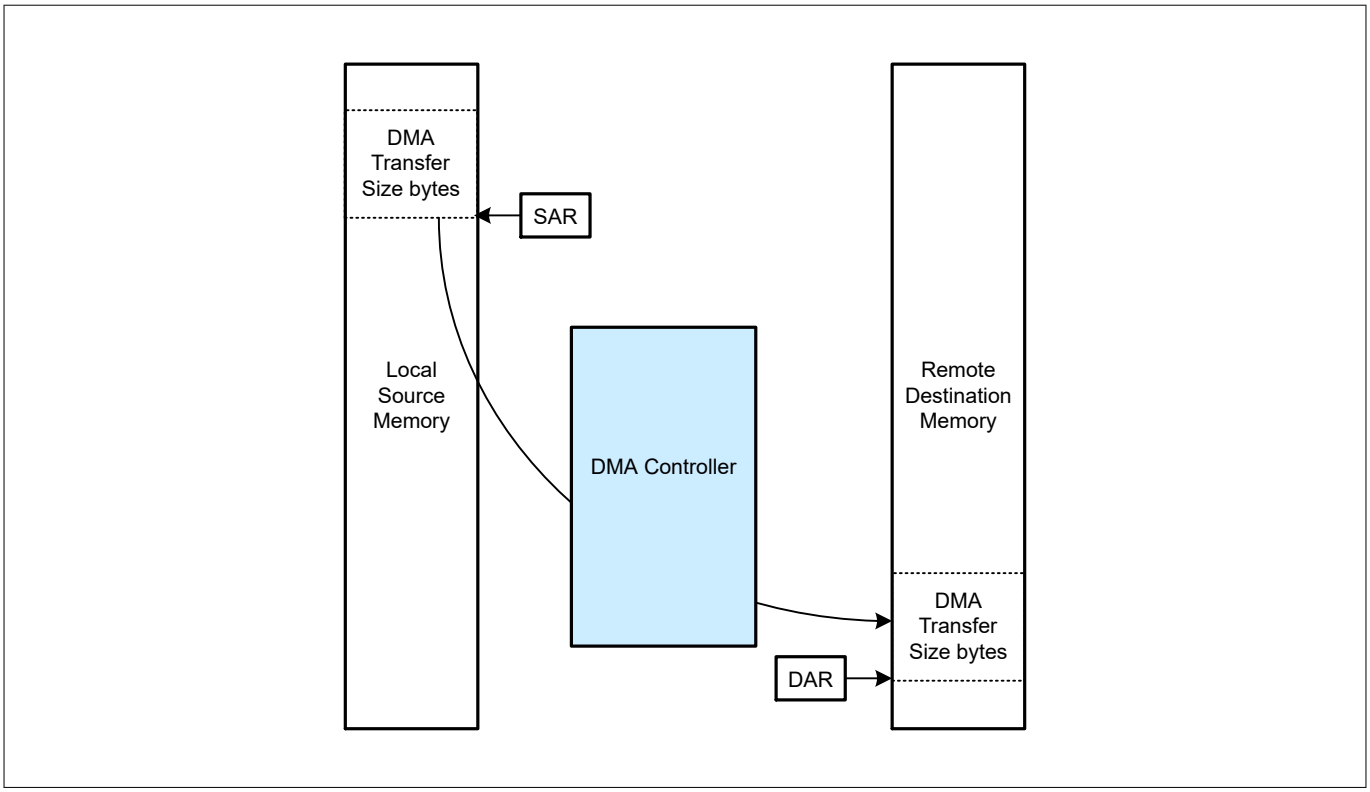


Figure 298 1 MB write DMA transfer from SAR to DAR

Table 840 Linked list element initial setup

Element #	Address	Data	Label
0	0000_0200_H	00000001_H	Channel Control
0	0000_0204_H	00100000_H	Transfer Size
0	0000_0208_H	$BEEF_BEE0_H$	SAR Low
0	0000_020C_H	0_H	SAR High
0	0000_0210_H	$CAFE_CAF0_H$	DAR Low
0	0000_0214_H	0_H	DAR High
1	0000_0218_H	00000001_H	Channel Control
1	0000_021C_H	00100000_H	Transfer Size
1	0000_0220_H	$BF0F_BEE0_H$	SAR Low
1	0000_0224_H	0_H	SAR High
1	0000_0228_H	$CBFE_CAF0_H$	DAR Low
1	0000_022C_H	0_H	DAR High
2	0000_0230_H	00000001_H	Channel Control
2	0000_0234_H	00100000_H	Transfer Size
2	0000_0238_H	$BF2F_BEE0_H$	SAR Low
2	0000_023C_H	0_H	SAR High
2	0000_0240_H	$CCFE_CAF0_H$	DAR Low
2	0000_0244_H	0_H	DAR High

(table continues...)

Table 840 (continued) Linked list element initial setup

Element #	Address	Data	Label
3	0000_0248_H	00000001_H	Channel Control
3	0000_024C_H	00100000_H	Transfer Size
3	0000_0250_H	$BF4F_BEE0_H$	SAR Low
3	0000_0254_H	0_H	SAR High
3	0000_0258_H	$CDFE_CAF0_H$	DAR Low
3	0000_025C_H	0_H	DAR High
4	0000_0260_H	00000001_H	Channel Control
4	0000_0264_H	00100000_H	Transfer Size
4	0000_0268_H	$BF6F_BEE0_H$	SAR Low
4	0000_026C_H	0_H	SAR High
4	0000_0270_H	$CEFE_CAF0_H$	DAR Low
4	0000_0274_H	0_H	DAR High
5	0000_0278_H	00000009_H	Channel Control
5	0000_027C_H	00100000_H	Transfer Size
5	0000_0280_H	$BF8F_BEE0_H$	SAR Low
5	0000_0284_H	0_H	SAR High
5	0000_0288_H	$CFFE_CAF0_H$	DAR Low
5	0000_028C_H	0_H	DAR High
6	0000_0290_H	00000001_H	Channel Control
6	0000_0294_H	00100000_H	Transfer Size
6	0000_0298_H	$BFAF_BEE0_H$	SAR Low
6	0000_029C_H	0_H	SAR High
6	0000_0290_H	$D0FE_CAF0_H$	DAR Low
6	0000_0294_H	0_H	DAR High
7	0000_0298_H	00000001_H	Channel Control
7	0000_029C_H	00100000_H	Transfer Size
7	0000_02A0_H	$BFCF_BEE0_H$	SAR Low
7	0000_02A4_H	0_H	SAR High
7	0000_02A8_H	$D1FE_CAF0_H$	DAR Low
7	0000_02AC_H	0_H	DAR High
8	0000_02B0_H	00000009_H	Channel Control
8	0000_02B4_H	00100000_H	Transfer Size
8	0000_02B8_H	$BFEF_BEE0_H$	SAR Low
8	0000_02BC_H	0_H	SAR High
8	0000_02C0_H	$D2FE_CAF0_H$	DAR Low
8	0000_02C4_H	0_H	DAR High

(table continues...)

Table 840 (continued) Linked list element initial setup

Element #	Address	Data	Label
9	0000_02C8_H	00000006_H	Channel Control
9	0000_02CC_H	0_H	Reserved
9	0000_02D0_H	0000_0200_H	Linked List Element Pointer Low
9	0000_02D4_H	0_H	Linked List Element Pointer High

Table 841 DMA register setup

Channel offset	Name	Value
DMA_WRITE_CHi_PWR_EN_OFF	DMA Write Engine Enable You must not write 0 to this register. Even temporarily writing 0 to this register resets the DMA logic. For more information, see DMA Write Engine Enable Register (DMA_WRITE_ENGINE_EN_OFF).	1_H
DMA_WRITE_INT_MASK_OFF	DMA Write Interrupt Mask	0_H
DMA_WRITE_LINKED_LIST_ERR_EN_OFF	DMA Write Linked List Error Enable • LLLAIE = 1	1_H
DMA_CH_CONTROL1_OFF_WRCH_i	DMA Channel Control 1 register • Linked List Enable (LLE) = 1 • Consumer Cycle Status (CCS) =PCS = 1 • AT, RO, NS, TC, Function Number = 0	04000300_H
DMA_LLP_LOW_OFF_WRCH_i	DMA Linked List Pointer Low	0000_0200_H
DMA_LLP_HIGH_OFF_WRCH_i	DMA Linked List Pointer High	0000_0000_H
DMA_WRITE_DOORBELL_OFF	DMA Write Doorbell	0_H

Related information

iATU on page 2752

Transmit/receive TLP on page 2711

19.3.7 iATU

Address translation is used for mapping different address ranges to different memory spaces supported by the system. A typical example maps your application memory space to PCIe memory space. The iATU also supports type translation. Without address translation, the bus address is passed unmodified to the TLPs directly through the bus interface. You can program the iATU to implement your own outbound address translation scheme.

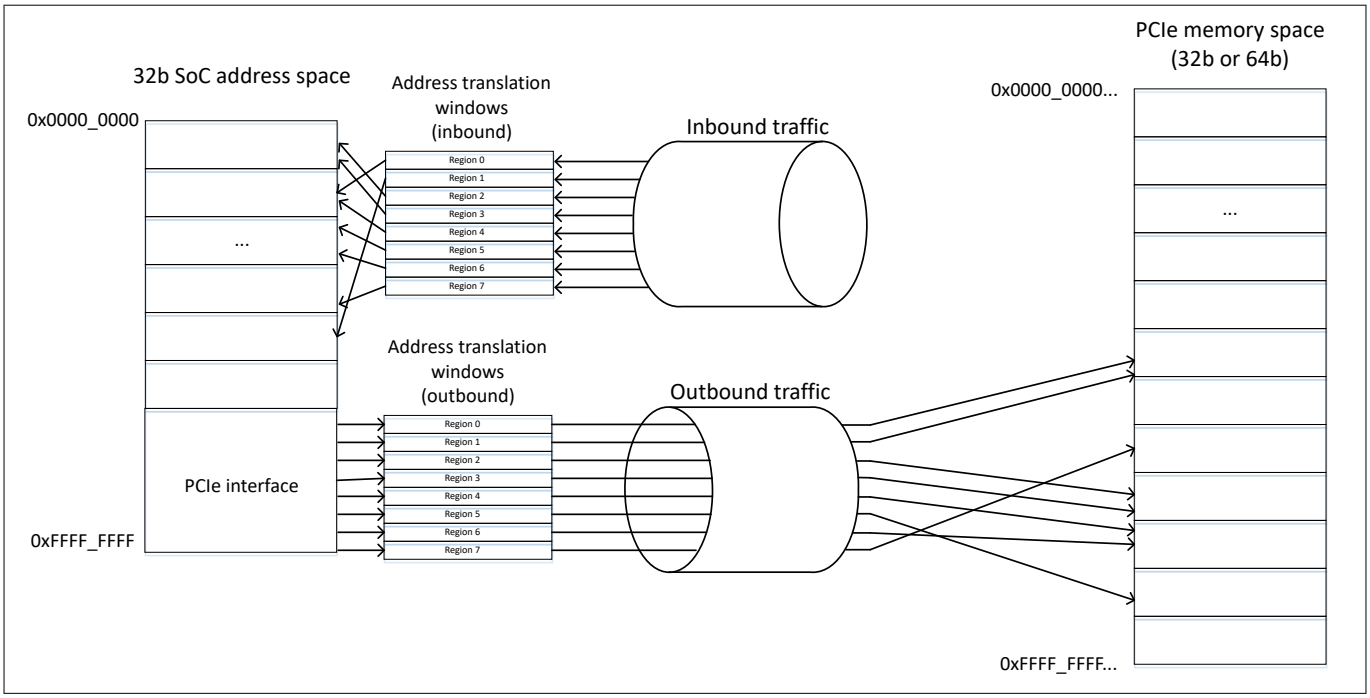


Figure 299 Memory map between PCIe memory space and SoC address (Cfg and Msg space not depicted)

19.3.7.1 Outbound features

- Address Match mode operation for MEM and I/O, CFG, and MSG TLPs
- Supports type translation through TLP type header field replacement for MEM or I/O types to MSG/CFG types
  - Includes posted to non-posted translation (for example, MWr to CfgWr0)
- Programmable TLP header field replacement
  - TYPE/TC/AT/ATTR/MSG-Code/TH/PH/ST
  - Function Number
- 8 address regions programmable for location and size
- Programmable enable/disable per region
- Automatic FMT field translation between three DWORDs and four DWORDs for 64-bit addresses
- Invert Address Matching mode to translate accesses outside of a successful address match
- Response code which defines the completion status to return for accesses matching a region
- Supports regions from 4 KB to 16 GB in size
- Payload Inhibit marks all TLPs as having no payload data
- Header Substitution replaces bytes 8 to 11 (for 3 DWORD header) or bytes 12 to 15 (for 4 DWORD header), inclusive, of the outbound TLP header
- Tag Substitution of the outbound TLP tag field
- DMA bypass mode to allow TLPs, which are initiated by the embedded DMA engine, to pass through the iATU untranslated
- TLP Header fields bypass mode to allow header information to be supplied from the bus interface

19.3.7.2 Outbound basic operation

The address field of each request MEM and I/O TLP is checked to see if it falls into any of the enabled address regions defined by the start and end addresses.

Note that when the Region Enable bit of the Region Control 2 register is '0', then that region is not used for address matching.

When an address match is found, then the TLP address field is modified as follows:

```
Translated Address = Original Address - Base Address + Target Address
```

and the TYPE, TC, AT, TH, PH, ST, Function Number, and ATTR header fields are replaced with the corresponding fields in the IATU_REGION_CTRL_1_OFF_OUTBOUND_i register. When the bus address field matches more than one of the 8 address regions, then the first (lowest of the numbers from 0 to 7) enabled region to be matched is used.

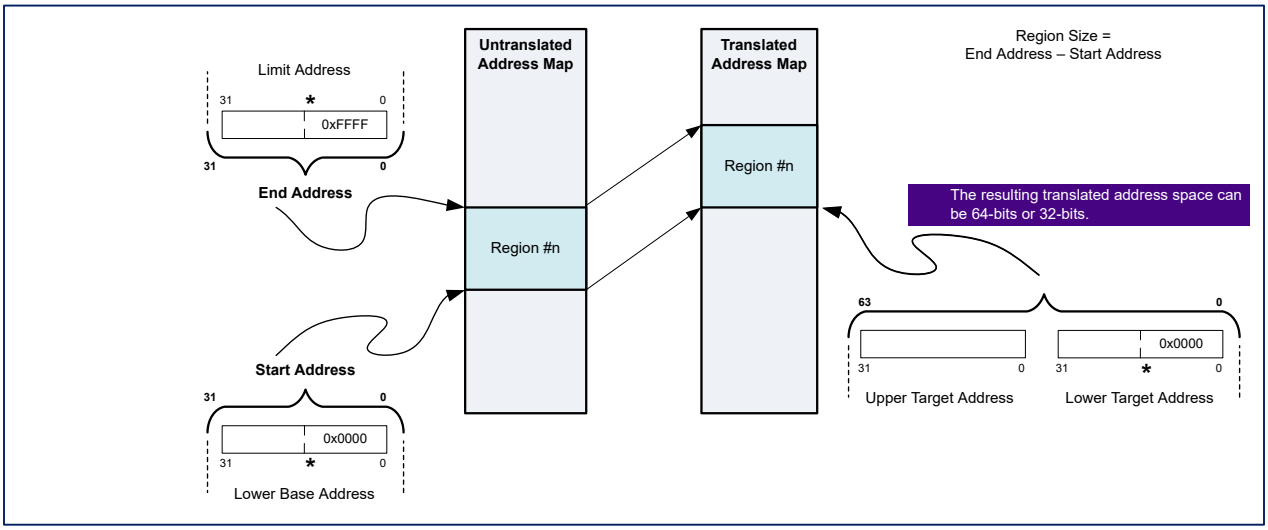


Figure 300 32-bit region mapping: outbound and inbound (address match mode)

19.3.7.3 Outbound detailed operation

Default traffic generated by the bus interface

If no regions matches, write or read accesses are converted to MWr or MRd TLPs. The generated TLP header fields such as TYPE, EP, TC and ATTR are all zeros. With ATU translation, the default memory TLPs can be translated into any TLP types.

RID BDF number replacement

Overview: When there is a successful address match on an outbound TLP, then the function number used in generating the function part of the requester ID[22)] field of the TLP is taken from the 3-bit Function Number field of the IATU_REGION_CTRL_1_OFF_OUTBOUND_i register. This feature can be bypassed by asserting the FUNC_BYPASS in IATU_REGION_CTRL_2_OFF_OUTBOUND_i.

22 Uses the 8:5:3 bit PCI Bus.Device.Function (BDF) format.

MEM to Msg/MsgD type translation

Overview: The iATU supports TYPE translation/conversion of MEM TLPs to Msg/MsgD TLPs. When there is a successful address match on an outbound MEM TLP, and the translated TLP type field is MSG (that is, the type field of the IATU_REGION_CTRL_1_OFF_OUTBOUND_i register is $10xxx_B$), then the message code field of the TLP is set to the value in the Message Code field of the IATU_REGION_CTRL_2_OFF_OUTBOUND_i register. A MWr with an effective length of '0' is converted to Msg and all other MWr TLPs are converted to MsgD. This is a way to generate messages.

MEM to CFG type translation

Overview: The iATU supports translation of MEM TLPs to CFG TLPs. The 16-bit BDF is located at bits [31:16] of the translated address where:

```
Translated Address = Original Address - Base Address + Target Address
```

As an example

```
Original Address[31:16] = {13h0,function_no[2:0]}
```

```
Base Address[31:16] = 16h0
```

```
Target Address[31:16] = {bus_no[7:0],device_no[4:0],3h0}
```

then

```
Translated Address[31:16] = BDF ={bus_no[7:0],device_no[4:0],function_no[2:0]}
```

To handle eight functions (as the previous example indicates), a 19-bit wide region size should be used. See CFG-shift feature how to limit the address range to 15 bit wide region size..

FMT translation

Overview: The iATU automatically sets the TLP format field for three DWORDs. When the original address and the translated address are of different format, the iATU ensures that the TLP header size matches the translated address format.

No address match result

Overview: When there is no address match then the address is untranslated but the TLP header information (for fields that are programmable) comes from the relevant fields on the bus interface, that is always MEM TLP.

Address match invert

Overview: In normal operation an address match on an outbound TLP occurs when the untranslated address is in the region bounded by the base address and limit address. When the invert feature is enabled, an address match occurs when the untranslated address is not in the region bounded by the base address and limit address.

Feature Configuration: To enable Invert feature, set the INVERT_MODE field of the IATU_REGION_CTRL_2_OFF_OUTBOUND_i register to '1'.

DMA bypass

Overview: When you do not want the iATU to translate outbound requests that are generated by the DMA, you must implement one of the following approaches:

- Ensure that the combination of DMA channel address programming and iATU control register programming causes no translation of DMA traffic to be done in the iATU, or
- Enable the DMA bypass mode to allow request TLPs which are initiated by the embedded DMA controller to pass through the iATU untranslated

Feature Configuration: To enable DMA bypass, set the DMA_BYPASS field of the IATU_REGION_CTRL_2_OFF_OUTBOUND_i register to '1'.

Payload inhibit

Overview: When payload inhibit is enabled and region address is matched, the iATU converts transactions with data payload to TLPs without payload data by forcing the TLP header Fmt[1] bit =0, regardless of the application inputs .

This feature must not be used for outbound transactions with data poisoning, because the EP bit is set for TLPs without data payload, this is a violation of data poisoning rules defined in section "2.7.2.2. Rules For Use of Data Poisoning" of the PCI Express Base Specification, Revision 5.0, Version 1.0.

Note: This feature is not supported for MemWr with data payload greater than 1DW.

Feature Configuration: To enable payload inhibit, set the INHIBIT_PAYLOAD field of IATU_REGION_CTRL_2_OFF_OUTBOUND_i to '1'.

Header substitution

Overview: When header substitution is enabled and region address is matched, the iATU fully substitutes bytes 8-11 (for 3 DWORD header) or bytes 12-15 (for 4 DWORD header) of the outbound TLP header with the contents of the LWR_TARGET_RW field in IATU_LWR_TARGET_ADDR_OFF_OUTBOUND_i.

Note: For a Msg/MsgD having 4 DWORD header, when HEADER_SUBSTITUTE_EN =1, the 4th DWORD (bytes 12-15) is replaced with the contents of LWR_TARGET_RW field in the IATU_LWR_TARGET_ADDR_OFF_OUTBOUND_i register. But regardless of the value of HEADER_SUBSTITUTE_EN, the 3rd DWORD is replaced with the contents of UPPER_TARGET_RW field in the IATU_UPPER_TARGET_ADDR_OFF_OUTBOUND_i register. So for a Msg, BDF (or reserved) and Vendor ID must be programmed in UPPER_TARGET_RW.

Usage Scenario: This is useful for Msg/MsgD and ATS (address translation service) transactions which is normally inefficient requiring a very large iATU region.

Feature Configuration: To enable header substitution, set the HEADER_SUBSTITUTE_EN field in IATU_REGION_CTRL_2_OFF_OUTBOUND_i register to '1'.

Tag substitution

Overview: When tag substitution is enabled and region address is matched, the iATU substitutes the TAG field of the outbound TLP header with the contents of the TAG field in IATU_REGION_CTRL_2_OFF_OUTBOUND_i.

Note: Your application must not attempt to perform TAG substitution for outgoing non-posted TLPs.

Usage Scenario: This is useful for translation from MWr to Msg/MsgD.

Feature Configuration: To enable tag substitution, set the TAG_SUBSTITUTE_EN field of the IATU_REGION_CTRL_2_OFF_OUTBOUND_i register to '1'.

19.3.7.4 Outbound programming

You can access the iATU registers through the configuration bus interface. The following registers are used for programming the iATU.

Table 842 iATU register map

Label (i = 0 to 7)	Description
IATU_REGION_CTRL_1_OFF_OUTBOUND_*i*	Region Control 1
IATU_REGION_CTRL_2_OFF_OUTBOUND_*i*	Region Control 2
ATU_LWR_BASE_ADDR_OFF_OUTBOUND_*i*	Lower Base Address
IATU_UPPER_BASE_ADDR_OFF_OUTBOUND_*i*	Upper Base Address
IATU_LIMIT_ADDR_OFF_OUTBOUND_*i*	Limit Address
IATU_LWR_TARGET_ADDR_OFF_OUTBOUND_*i*	Lower Target Address
IATU_UPPER_TARGET_ADDR_OFF_OUTBOUND_*i*	Upper Target Address

***Note**: You must not update the iATU registers while operations are in progress on the bus slave interface.*

Outbound programming examples

Define Outbound Region 1 as:

64 kB I/O region from 80000000_D000000$_H$ to 80000000_D000FFFF$_H$, to be mapped to 00010000$_H$ in the PCIe I/O space.

1. Setup the Region Base and Limit Address Registers.
   Write D0000000$_H$ to ATU_LWR_BASE_ADDR_OFF_OUTBOUND_*1* to set the Lower Base Address
   Write 80000000$_H$ to IATU_UPPER_BASE_ADDR_OFF_OUTBOUND_*1* to set the Upper Base Address
   Write D000FFFF$_H$ to IATU_LIMIT_ADDR_OFF_OUTBOUND_*1* to set the Limit Address
2. Setup the Target Address Registers.
   Write 00010000$_H$ to IATU_LWR_TARGET_ADDR_OFF_OUTBOUND_*1* to set the Lower Target Address
   Write 00000000$_H$ to IATU_UPPER_TARGET_ADDR_OFF_OUTBOUND_*i* to set the Upper Target Address
3. Configure the region through the Region Control 1 Register.
   Write 00000002$_H$ to IATU_REGION_CTRL_1_OFF_OUTBOUND_*1* to define the type of the region to be I/O
4. Enable the region.
   Write 80000000$_H$ to IATU_REGION_CTRL_2_OFF_OUTBOUND_*1* to enable the region.

Define Outbound Region 1 as 16 kB region from FC00_0000$_H$ to FC00_3FFF$_H$, to be mapped to 0113_0000$_H$ in the PCIe configuration space.

1. Setup the Region Base and Limit Address Registers.
   Write FC00_0000$_H$ to ATU_LWR_BASE_ADDR_OFF_OUTBOUND_*1* to set the Lower Base Address.
   Write FC00_3FFF$_H$ to IATU_LIMIT_ADDR_OFF_OUTBOUND_*1* to set the Limit Address
2. Setup the Target Address Registers.
   This is the lower 32 bits of the address of the start of the translated range. For configuration access translation bits 31:16 form the BDF (bus – 31:24, device – 23:19 and function – 18:16 numbers) in case the config_shift_mode bit is not set in IATU_REGION_CTRL_2_OFF_OUTBOUND_*1*. Write 0113_0000$_H$ to IATU_LWR_TARGET_ADDR_OFF_OUTBOUND_*1* to set the Lower Target Address for bus number 1, device number 2 and function number 3 when the config shift mode is not set
3. Configure the region through the Region Control 1 Register.
   Write 0000_0004$_H$ to IATU_REGION_CTRL_1_OFF_OUTBOUND_*1* to translate to type 0 configuration TLP
4. Enable the region.

Write 80000000_H to IATU_REGION_CTRL_2_OFF_OUTBOUND_*1* to enable the region

19.3.7.5 Inbound features

- Programmable Match mode operation for MEM, I/O, CFG, and MSG TLPs. No translation for completions
- Selectable BAR Match mode operation for I/O and MEM TLPs
  - TLPs destined for the TRGT0 interface are not translated
  - TLPs that are not error-free (ECRC, malformed and so on) are not translated
- Programmable TLP header field matching
  - TYPE/TD/TC/AT/ATTR/MSG-Code/TH/PH/ST
  - Function Number
- 8 address regions programmable for location and size
- Programmable enable or disable per region
- Automatic FMT field translation between three DWORDs and four DWORDs for 64-bit addresses
- Invert Address Matching mode to translate accesses outside of a successful address match
- ECAM Configuration Shift mode to allow a 256 MB CFG1 space to be located anywhere in the 64-bit address space
- Supports regions from 4 KB to 16 GB in size
- Single Address Location to allow all TLPs to be translated to a single address location
- Msg Type Match Mode to allow matching of any TLP of type Message

***Note**: The default behavior of the ATU when there is no address match in the outbound direction or no TLP attribute match in the inbound direction, is to pass the transaction through.*

19.3.7.6 Inbound basic operations

The setting of the MATCH_MODE field in IATU_REGION_CTRL_2_OFF_INBOUND_i determines how iATU inbound matching is done for each TLP type.

Table 843 Determination of match mode

TLP type	MATCH_MODE =0	MATCH_MODE =1
CFG0	Routing ID Match Mode	Accept Mode
MEM or I/O	Address Match Mode	BAR Match Mode
MSG/MSGD	Address Match Mode	Vendor ID Match Mode

The main difference between inbound and outbound iATU operation is that the TLP type is never changed in the inbound direction. Instead, the type field is used for more precise matching. Other fields can also be optionally used to further refine the matching process.

Another difference is that for MEM and I/O TLPs, you can select between address matching (as used in outbound operation) or BAR matching. Normally, an endpoint uses BAR match mode, and a root complex uses address mode as an root complex normally does not implement BARs.

By default, the MSG and the CFG TLPs are terminated and consumed internally by the controller. No iATU translation happens.

MEM or I/O match modes

Address match mode

The address field of each request TLP is checked to see if it falls into any of the enabled regions[23]. When an address match is found then the TLP address field is modified as follows:

```
Address = Address - Base Address + Target Address
```

When the TLP address field matches more than one of the eight address regions, then the first (lowest of the numbers from 0 to 7 enabled region to be matched is used.

In address match mode, when the address range does not match one of its BAR ranges in an upstream port, then the device rejects the request with unsupported request (UR) completion status and no translation occurs. When the PCIe controller is operating with 32-bit BARs, the operation is defined as in the following figure.

BAR match mode

Looking for an address match is a two-step process.

1. The standard internal PCI Express BAR Matching mechanism checks if the address field of any MEM and I/O request TLP falls into any address region defined by the enabled BAR addresses and masks
2. When a matched BAR is found, the iATU compares the BAR ID to the BAR Number (BAR_NUM) field in the IATU_REGION_CTRL_2_OFF_INBOUND_i register for all enabled regions. The following two figures provide more information on inbound translation in BAR Match Mode

When the PCIe controller is operating with 64-bit BARs, the operation is defined as in figure below where * is $\log_2$(BAR_MASK+1).

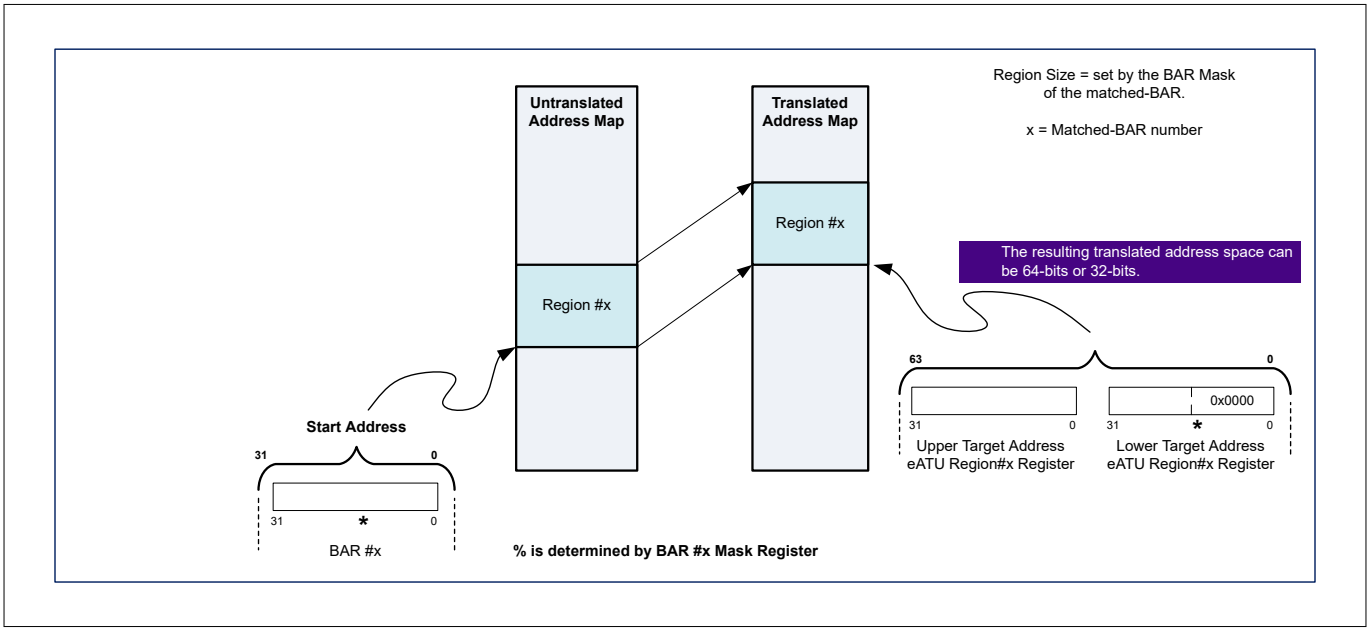


Figure 301 iATU address region mapping: inbound (BAR Match Mode), 32-bit BAR

23 When the Region Enable bit of the Region Control 2 register is 0, then that region is not used for address matching.

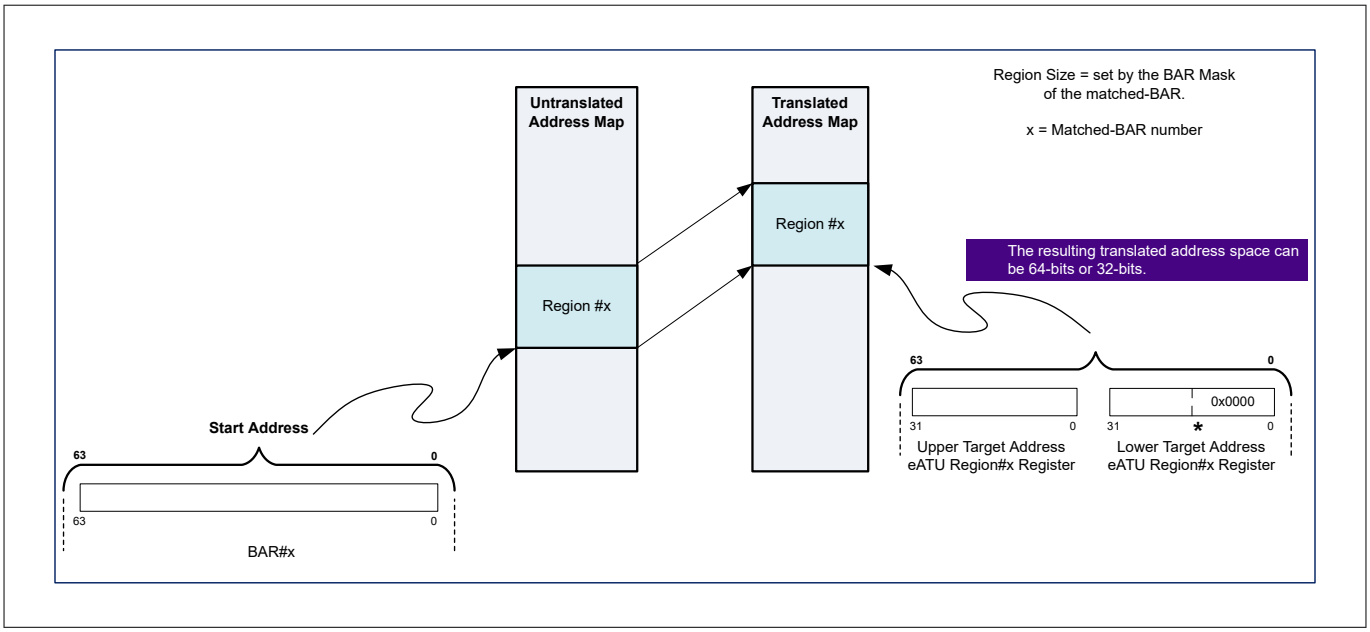


Figure 302 iATU address region mapping: inbound (BAR Match Mode), 64-bit BAR

CFG match mode

The controller normally routes CFG TLPs (to the controller internal registers) without translation. The iATU only translates CFG0 TLPs that the controller has routed to the bus master interface.

Accept mode

In Accept Mode, the controller always accepts CFG0 TLPs even when the CFG bus number does not match the current bus number of the device. The routing ID of received CFG0 TLPs are ignored when determining a match.

Routing ID match mode

The routing ID of the inbound CFG0 TLP must fall within the Base and Limit of the defined iATU region for matching to proceed. The iATU interprets the routing ID (Bytes[24)] 8-9 of TLP header) as an address. This corresponds to the upper 16 bits of the address in MEM and I/O transactions.

MSG/MSGD match mode

Address match mode

The third and fourth header DWORDs are treated as an address and are compared against the iATU Region Base and Limit Address registers.

Vendor ID match mode

The iATU ignores the routing ID (BDF) in bits [31:16] of the third DWORD of the TLP header[25)], but compares it against the vendor ID in bits [15:0] of the third DWORD of the TLP header (bytes[25)] 10 and 11). This allows messages to be filtered against specific vendor IDs without needing to know the BDF number which might vary

24 For more information, see Figure 2-18 Request Header Format for Configuration Transactions in Section 2.2.7 Memory, I/O, and Configuration Request Rules of the PCI Express Base Specification, Revision 5.0, Version 1.0.

25 In Figure 2-25 Header for Vendor_Defined Messages in Section 2.2.8.6, Vendor_Defined Message of the PCI Express Base Specification, Revision 5.0, Version 1.0.

depending on the PCI topology. Bits [15:0] of the Region Upper Base register should be programmed with the required vendor ID as follows:

```
Region Upper Base[15:8] =byte 10
```

```
Region Upper Base[7:0] =byte 11
```

The lower base and limit register should be programmed to translate TLPs based on vendor-specific information in the fourth DWORD of the TLP header.

19.3.7.7 Inbound detailed operations

FMT translation

The iATU automatically sets the TLP format field for three DWORDs when it detects all zeroed in the upper 32 bits of the translated address. When the original address and the translated address are of a different format then the iATU ensures that the TLP header size matches the translated address format.

Optional matching fields

Overview: In address and BAR match modes, a successful address/BAR match can be optionally gated by successful matching of the programmable TLP header fields (per region) mentioned in table 19. When the associated Match Enable bit (see column "Field" below) is set, the TLP header field of an inbound TLP is matched to the programmed TLP header field value.

Table 844 Registers to consider for TLP header field matching

TLP header field	Match enable		Match value	
	Field	Register	Field	Register
TC	TC_MATCH_EN	IATU_REGION_CTRL_2_OFF_INBOUND_i	TC	IATU_REGION_CTRL_1_OFF_INBOUND_i
TD	TD_MATCH_EN		TD	
ATTR	ATTR_MATCH_EN		ATTR	
TH	TH_MATCH_EN		TH	
AT	AT_MATCH_EN		AT	
PH	PH_MATCH_EN		PH	
Message Code	MSG_CODE_MATCH_EN		MSG_CODE	IATU_REGION_CTRL_2_OFF_INBOUND_i

***Note**: Address translation only proceeds when compares on all enabled field are successful.*

Single address location

Overview: Single address location feature has two modes of operation, fixed and circular buffer.

Fixed: When enabled and region address matches, the TLPs translates to a single address location as determined by the target address register of the iATU region.

Circular Buffer: In this mode, the first TLP translates to a single address location as determined by the target address register of the iATU region. But, for subsequent TLPs, the target address increments by a programmable increment size (CBUF_INCR), each time a region address matches. You can program the increment size using CBUF_INCR field of the IATU_LIMIT_ADDR_OFF_INBOUND_i register (the maximum

increment size is 8192). On reaching the circular buffer window size, the target address wraps around, and resets to single location base address.

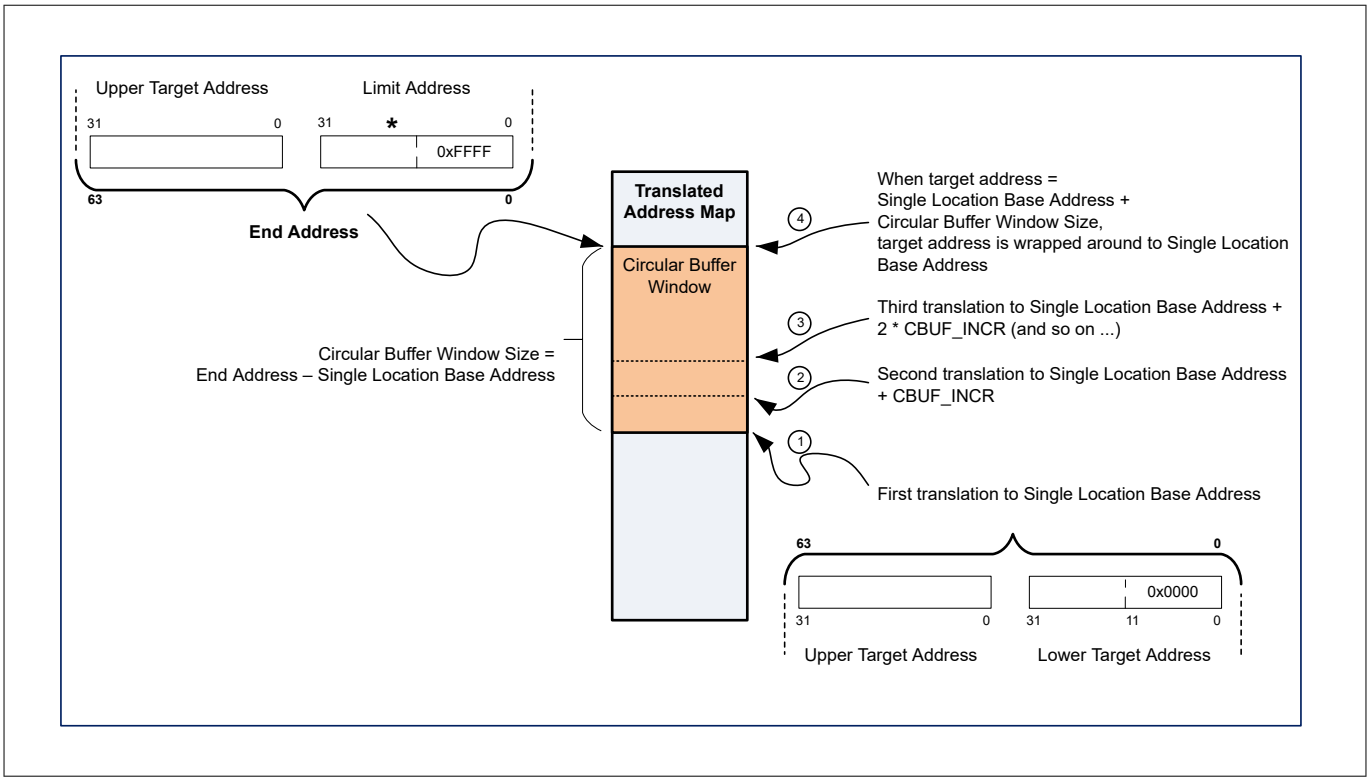


Figure 303 Circular buffer operation

Usage scenario: The main usage scenario is translation of Messages to MWr TLPs.

Feature configuration: To enable single address location feature, set the SINGLE_ADDR_LOC_TRANS_EN field of the IATU_REGION_CTRL_2_OFF_INBOUND_i register to '1'.

Steps to Program ATU Circular Buffer registers

1. Write to Upper Target Address, Lower Target Address, and Limit Address (all inbound) registers for start and end address values of Circular Buffer window for the ATU region
2. Set SINGLE_ADDR_LOC_TRANS_EN = 1 to enable single location address
3. Write the desired increment value to the CBUF_INCR field of the IATU_LIMIT_ADDR_OFF_INBOUND_i register. This enables the circular buffer feature

Table 845 Setting the single address location operation mode

Fixed	Circular buffer
IATU_LIMIT_ADDR_OFF_INBOUND_i.CBUF_INCR =0	IATU_LIMIT_ADDR_OFF_INBOUND_i.CBUF_INCR >0
	IATU_REGION_CTRL_2_OFF_INBOUND_i.SINGLE_ADDR_LOC_TRANS_EN =1

Response code

Overview: Based on the setting of the response code field of the IATU_REGION_CTRL_REG_2_INBOUND_i register (00_B: Normal RADM filter response, 01_B: UR, or 10_B: CA), the controller sets the completion status field of completion TLPs, sent in response to successfully matched non-posted TLPs. When the response code field is set to 2b00, then the normal receive filter response for this TLP is used.

***Note**: Response Code feature is not available in regions where Single Address Location Enable is set.*

Feature Configuration: To enable response code feature, set the RESPONSE_CODE field of the IATU_REGION_CTRL_2_OFF_INBOUND_i register!='00'.

Invert

Overview: Normally an address match on an inbound TLP occurs when the untranslated address is in the region bounded by the Base address and Limit address. When the invert feature is enabled, an address match occurs when the untranslated address is not in the region bounded by the Base address and Limit address.

This feature is supported only when all regions of that type are set to address match mode.

Feature Configuration: To enable invert feature, set the INVERT_MODE field of the IATU_REGION_CTRL_2_OFF_INBOUND_i register to ='1'.

19.3.7.8 Inbound programming

iATU inbound programming overview

This section discusses the register programming interface and provides details of programming examples.

You can access the iATU registers through the configuration bus interface. The following registers are used for programming the iATU.

Table 846 iATU register map

Label (i = 0 to CX_ATU_NUM_INBOUND_REGIONS-1)	Description
IATU_REGION_CTRL_1_OFF_INBOUND_*i*	Region Control 1
IATU_REGION_CTRL_2_OFF_INBOUND_*i*	Region Control 2
ATU_LWR_BASE_ADDR_OFF_INBOUND_*i*	Lower Base Address
IATU_UPPER_BASE_ADDR_OFF_INBOUND_*i*	Upper Base Address
IATU_LIMIT_ADDR_OFF_INBOUND_*i*	Limit Address
IATU_LWR_TARGET_ADDR_OFF_INBOUND_*i*	Lower Target Address

***Note**: You must not update the iATU registers while operations are in progress on the AXI bridge slave interface.*

Inbound programming examples

Address Match Mode Example

Define Inbound Region i as, MEM region matching TLPs with addresses in the range 00010000_H to $0003FFFF_H$ mapped to 90300000_H - $9032FFFF_H$ in your application memory space.

1. Setup the Region Base and Limit Address Registers.
   Write 00010000_H to ATU_LWR_BASE_ADDR_OFF_INBOUND_*i* to set the Lower Base Address.
   Write 00000000_H to IATU_UPPER_BASE_ADDR_OFF_INBOUND_*i* to set the Upper Base Address.
   Write $0003FFFF_H$ to IATU_LIMIT_ADDR_OFF_INBOUND_*i* to set the Limit Address
2. Setup the Target Address Registers.
   Write 90300000_H to IATU_LWR_TARGET_ADDR_OFF_INBOUND_i to set the Lower Target Address
3. Configure the region through the Region Control 1 Register.
   Write 00000000_H to IATU_REGION_CTRL_1_OFF_INBOUND_*i* to define the type of the region to be MEM
4. Enable the region.
   Write 80000000_H to IATU_REGION_CTRL_2_OFF_INBOUND_*i* to enable the region in address match mode

BAR Match Mode Example

Define Inbound Region i as MEM region matching BAR4 (BAR match mode) mapping to 90500000_H in your application memory space.

1. Setup the Target Address Registers.

Write 90500000_H to IATU_LWR_TARGET_ADDR_OFF_INBOUND_i to set the Lower Target Address

2. Configure the region through the Region Control 1 Register.
   Write 00000000_H to IATU_REGION_CTRL_1_OFF_INBOUND_*i* to define the type of the region to be MEM
3. Enable the region for BAR Match Mode.
   Write $C0000400_H$ to IATU_REGION_CTRL_2_OFF_INBOUND_*i* to enable the region for BAR match mode for BAR#4

***Note**: For an EP, the defined MEM or I/O regions must be inside an enabled BAR range. For a RC, the defined MEM or I/O regions must either match a BAR or be outside of the base and limit ranges defined for the port in the Type 1 configuration header.*

19.3.8 BAR details

As an EP, the PCIe controller provides six base address registers (BAR) to define the memory assignment. As a RC, two BARs are available according to the PCIe specification but they are not expected to be used and shall be disabled.

The controller provides three pairs of 32-bit BARs for each implemented function. Each pair (BARs 0 and 1, BARs 2 and 3, BARs 4 and 5) can be configured as follows:

- One 64-bit BAR: For example, BARs 0 and 1 are combined to form a single 64-bit BAR
- Two 32-bit BARs: For example, BARs 0 and 1 are two independent 32-bit BARs
- One 32-bit BAR: For example, BAR0 is a 32-bit BAR and BAR1 is either disabled

In AURIX™, BAR0 has different access regions than the other BARs, as described in table 2. When BAR0 is combined with BAR1 to have a 64-bit BAR, then this pair will inherit the access regions of BAR0.

The BAR masks (BARi_MASK_REG) are used for indicating the amount of address space that each BAR requests from host software. The BAR masks determine which bits in each BAR are non-writable by host software, which determines the size of the address space claimed by each BAR. The BAR mask values indicate the range of low-order bits, in each implemented BAR, not to use for address matching. The BAR mask value also indicates the range of low-order bits in the BAR that cannot be written from the host. The low-order bits in the BAR should be contiguous. The application can write to all BAR bits to allow setting of memory, I/O, and other standard BAR options.

You can change the mask at runtime and the BARs shall be configured again by RC. The BAR mask registers are only writable through the local bus, and not readable. They are not accessible from the PCIe wire by the remote link partner. When the BAR mask value for a BAR is less than that required for the BAR type, the controller automatically uses the minimum value for the BAR type

To disable a BAR, your application can write 0 to the LSB of the BAR mask register.

General rules for BAR setup

At run-time, software can overwrite the BAR contents to reconfigure the BARs. Software must observe the rules listed when writing to the BARs. The rules for BAR configuration are the same for all three pairs. Using BARs 0 and 1 as the example pair, the rules for BAR configuration are:

- Any pair (for example BARs 0 and 1) can be configured as one 64-bit BAR, two 32-bit BARs, or one 32-bit BAR
- BAR pairs cannot overlap to form a 64-bit BAR. For example, it is not possible to combine BARs 1 and 2 to form a 64-bit BAR
- An I/O BAR must be a 32-bit BAR and cannot be prefetchable
- When the device is configured as a PCI Express endpoint (not a Legacy endpoint), any memory that is configured as prefetchable must be assigned to a 64-bit memory BAR
- When BAR0 is configured as a 64-bit BAR:
  - BAR1 is the upper 32 bits of the combined 64-bit BAR formed by BAR0 and BAR1. Therefore, BAR1 must be disabled and cannot be configured independently

  - BAR0 must be a memory BAR and can be either prefetchable or non-prefetchable
  - The contents of the BAR0 mask register determine the number of writable bits in the 64-bit BAR, subject to the restrictions. The BAR1 mask register contains the upper 32 bits of the BAR0 mask value
  - BAR0 can be disabled by writing 0 to bit [0] of the BAR0 mask register
- When BAR0 is configured as a 32-bit BAR:
  - You can configure BAR1 as an independent 32-bit BAR
  - BAR0 can be configured as a memory BAR or an I/O BAR
  - The contents of the BAR0 mask register determine the number of writable bits in the 32-bit BAR0
  - BAR0 can be disabled by writing 0 to bit [0] of the BAR0 mask register
- When BAR0 is configured as a 32-bit BAR, BAR1 is available as an independent 32-bit BAR according to the following rules:
  - BAR1 can be configured as a memory BAR or an I/O BAR
  - The contents of the BAR1 mask register determine the number of writable bits in the 32-bit BAR1
  - BAR1 can be disabled by writing 0 to bit [0] of the BAR1 mask register

The same rules apply for pairs 2/3 and 4/5.

Memory BAR sizes

BAR bits [11:0] are always masked for a memory BAR. The controller requires each memory BAR to claim at least 4 KB. The PCI Express Base Specification, Revision 5.0, Version 1.0 states that the minimum memory address range requested by a BAR is 128 bytes. In the *PCI Local Bus Specification, rev 3.0*, it is recommended that devices that need less than 4 KB of address space should still consume 4 KB of address space in order to minimize the number of bits in the address decoder.

BAR Mask registers

Your local CPU can change the mask at runtime and the BARs shall be configured again by RC. The mask is a register that is invisible to the PCIe wire, but visible to local CPU. You cannot read the mask register but you can write to it.

Example BAR setup

The following figure shows an example configuration of the six BARs and their corresponding BAR mask registers. The example configuration includes:

- One 64-bit memory BAR (non-prefetchable)
- One 32-bit memory BAR (non-prefetchable)
- One 32-bit I/O BAR

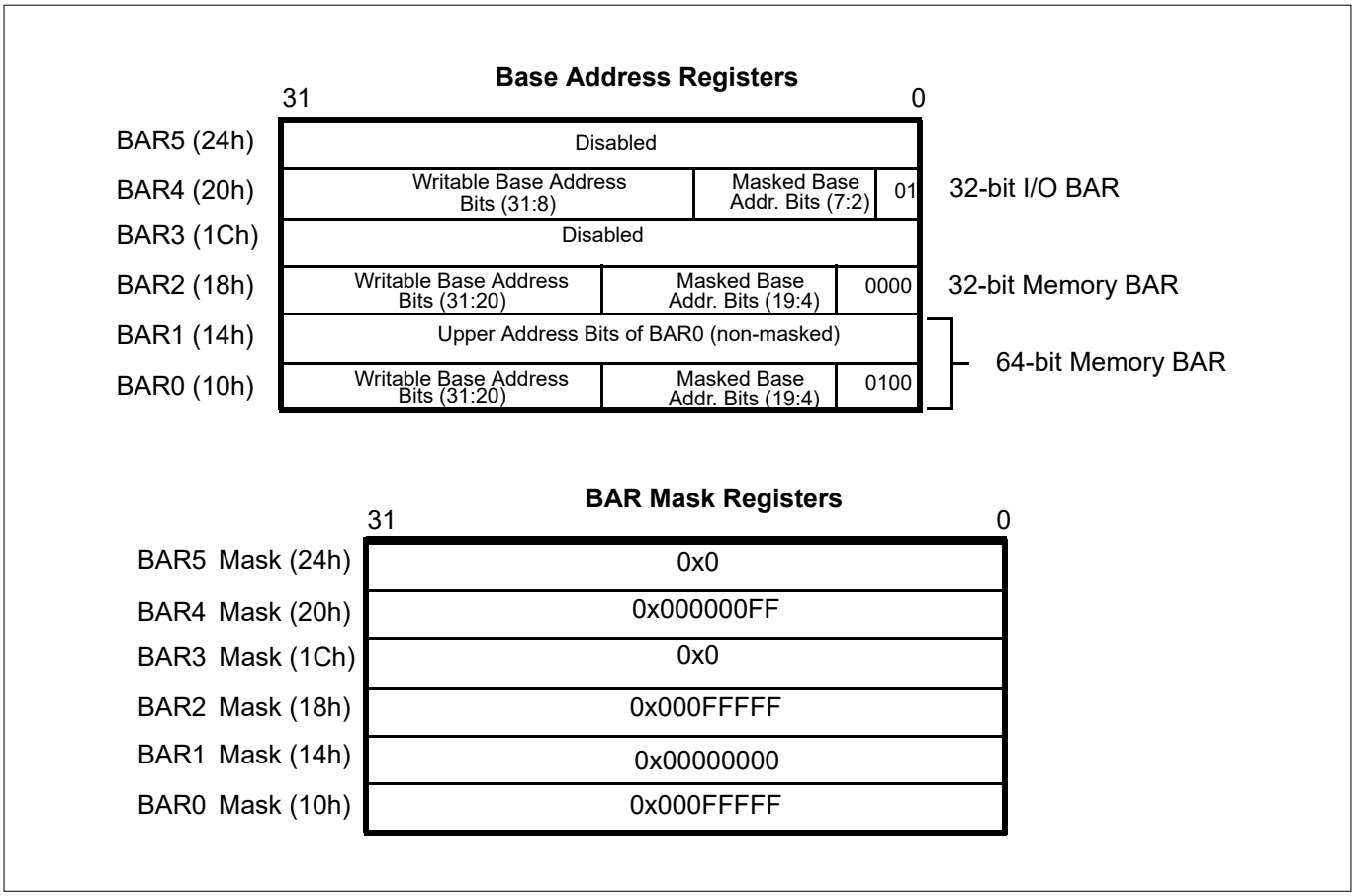


Figure 304 Example base address register configuration

Expansion ROM BAR mask register

The mask is a register that is invisible to the PCIe wire but visible to local CPU. You cannot read the mask register but you can write to it.

BARs in RC

Two BARs are present but are not expected to be used. You should disable them to avoid unnecessary memory assignment during device enumeration. If you do use a BAR, then you should program it to capture TLPs that are targeted to your local memory space residing on the bus. The BAR range must be outside of the three Base/Limit regions (three regions defined in type 1 configuration header). The controller provides one pair of 32-bit BARs (BAR0 and BAR1). The BARs can be configured as follows:

- One 64-bit BAR: BAR0 and BAR1 are combined to form a single 64-bit BAR
- Two 32-bit BARs: BAR0 and BAR1 are two independent 32-bit BARs
- One 32-bit BAR: BAR0 is a 32-bit BAR and BAR1 is either disabled or removed from controller altogether to reduce gate count

The sizing of BAR, the disabling and setup rules are similar to BARs in EP.

19.3.9 Messages

Message requests in PCIe replace many of the interrupt, error and power management sideband signals on PCI and PCI-X. They are handled as MSG TLPs and usually have fixed transmit directions.

A subset of PCIe messages is supported by AURIX™ . They are listed in the following table. The message propagation directions and brief descriptions are also introduced.

Table 847 Types of messages

Index	Message type	Message	EP mode	RC mode	Description
1	INTx	Assert_INTx, Deassert_INTx	T	R	legacy interrupt Messages
2	power management	PM_Active_State_Nak	R	T	indicates a rejection when entering L1
		PM_PME	T	R	requests a PM state change
		PME_Turn_Off	R	T	broadcasts the PME_Turn_Off Message in preparation for removing the main power source
		PME_TO_Ack	T	R	initiates the L2/L3 Ready transition protocol
3	error signaling	ERR_COR	T	R	detects a correctable error
		ERR_NONFATAL	T	R	detects a non-fatal, uncorrectable error
		ERR_FATAL	T	R	detects a fatal, uncorrectable error
4	Device readiness status	DRS or FRS	T	R	indicates that device has become configuration-ready
5	LTR	-	T	R	reports the tolerance of read or write service latencies
6	OBFF	-	R	T	reports platform central resource states to EP
7	PTM	PTM request	T	R	initiates PTM dialog
		PTM response	R	T	completes current PTM dialog – does not carry timing information
		PTM responseD	R	T	completes current PTM dialog - carries timing information

***Note**: R = Supports as Receiver; T= Supports as Transmitter*

Related information

Message generation on page 2767
Message reception on page 2768

19.3.9.1 Message generation

Various messages can be generated by AURIX™ depending on the work mode, either EP or RC. Some messages can only be generated under a specific work mode. For example, the INTx messages can be transmitted by an EP and received by a RC. These restrictions are defined by the PCIe specification.

The table below shows the message generation mechanisms under two modes. Some messages are triggered by hardware blocks autonomously. For example, the PTM messages. Some messages have to be triggered by application software directly. For example, the OBFF messages. The software uses OBFF messages to indicate the system status to other devices and provides hints how to save power. There are also messages that provide both possibilities.

Table 848 Generation sources of messages

Index	Message type	Message	EP mode	RC mode
1	INTx	Assert_INTx, Deassert_INTx	SW ()	-
2	Power management	PM_Active_State_Nak	-	HW
		PM_PME	SW ()	-
		PME_Turn_Off	-	SW ()
		PME_TO_Ack	HW	-
3	Error signaling	ERR_COR	HW;SW()	-
		ERR_NONFATAL	HW;SW()	-
		ERR_FATAL	HW;SW()	-
4	Device readiness status	DRS	HW	-
5	LTR	-	SW()	-
6	OBFF	-	-	SW()
7	PTM	PTM request	HW	-
		PTM response	-	HW
		PTM responseD	-	HW

***Note**: SW = software triggered message generation; HW = hardware autonomously triggered message generation*

There are two ways to generate a message by software. The straightforward way (Direct software triggering) is using the designated registers to trigger a specific message generation. Application software simply asserts a specific bit and then the desired message is created and transmitted. The second way (Indirect software triggering) is using the iATU to generate message TLPs. In this way, all types of messages can be created with the flexibility to control specific values in register fields. The details of how to use ATU to generate messages are described in the section iATU.

Indirect software triggering

Msg and MsgD can be generated with an indirect SW trigger. The iATU needs to be configured to translate MWr to Msg TLPs or MsgD TLPs. Msg has no payload, achieved by using the payload inhibit feature of iATU. The header and tag substitution are used to create the head for Msg by using IATU_LWR_TARGET_ADDR_OFF_OUTBOUND and IATU_UPPER_TARGET_ADDR_OFF_OUTBOUND. The address of a bus transaction triggered by application shall be DWORD aligned in order to be correctly translated to Msg.

Related information

INTx on page 2721

iATU on page 2752

19.3.9.2 Message reception

The reception of PCIe messages also follows rules defined by the PCIe specification. The table below indicates how various messages are handled by PCIe module under both work modes. Messages received can be handled by the hardware blocks automatically or by software interrupts.

Table 849 Reception handling of messages

Index	Message type	Message	EP mode	RC mode
1	INTx	Assert_INTx, Deassert_INTx	-	IR
2	Power management	PM_Active_State_Nak	PMC	-
		PM_PME	-	PMC
		PME_Turn_Off	PMC	-
		PME_TO_Ack	-	PMC
3	Error signaling	ERR_COR	-	SW()
		ERR_NONFATAL	-	SW()
		ERR_FATAL	-	SW()
4	Device readiness status	DRS	-	FRS queue
5	LTR	-	-	SW()
6	OBFF	-	SW()	-
7	PTM	PTM request	-	PTM
		PTM response	PTM	-
		PTM responseD	PTM	-

***Note**: IR = interrupt router; PMC = power management controller; SW = software handling via interrupt; FRSQ= FRS queue; PTM = precise time measurement block*

19.3.10 Power management

According to PCIe specification, D states and L states are specified. D states are associated with a particular function while L states are associated with a particular link. PM defines link power management states that a PCIe physical link is permitted to enter in response to either software driven D state transitions or active state link power management activities.

PCIe has several power management protocols. In this document, two relevant ones, Active State Power Management (ASPM) and software-driven PM, are briefly introduced and applied in AURIX™. ASPM is an autonomous hardware-based, active state mechanism that enables power savings. After a period of idle link time, an ASPM protocol places the idle link into a lower power state. Software-driven PM manipulates the D states and is a PCI-PM compatible mechanism. The link state are not visible to PCI-PM legacy compatible software, and are driven from the D state of the corresponding components connected to the link or by ASPM.

19.3.10.1 Device power management state

The PCIe protocol is compatible with all PCI power management functionality. Software can direct the function to enter any of the D1, D2 or D3 low-power states from D0. This device only supports D0, $D3_{hot}$ and $D3_{cold}$. This is achieved by writing to the PMCSR in the PCI-PM capability structure.

D states:

- D0: Normal power state. In this state, no power conservation is in effect and the device is fully operational

- D1 (not supported by this device): Light sleep. Before going into this state, software must ensure that all outstanding non-posted requests have received their associated completions. This state is not supported
- D2 (not supported by this device): Deep sleep. Similar to D1, software must ensure that all outstanding non-posted requests have received their associated completions. This state is not supported
- $D3_{hot}$: Very low power state; power supply available
- $D3_{cold}$: Power-off state; power supply removed; auxiliary power supply optionally available

The transitions between different D states are described in the following figure.

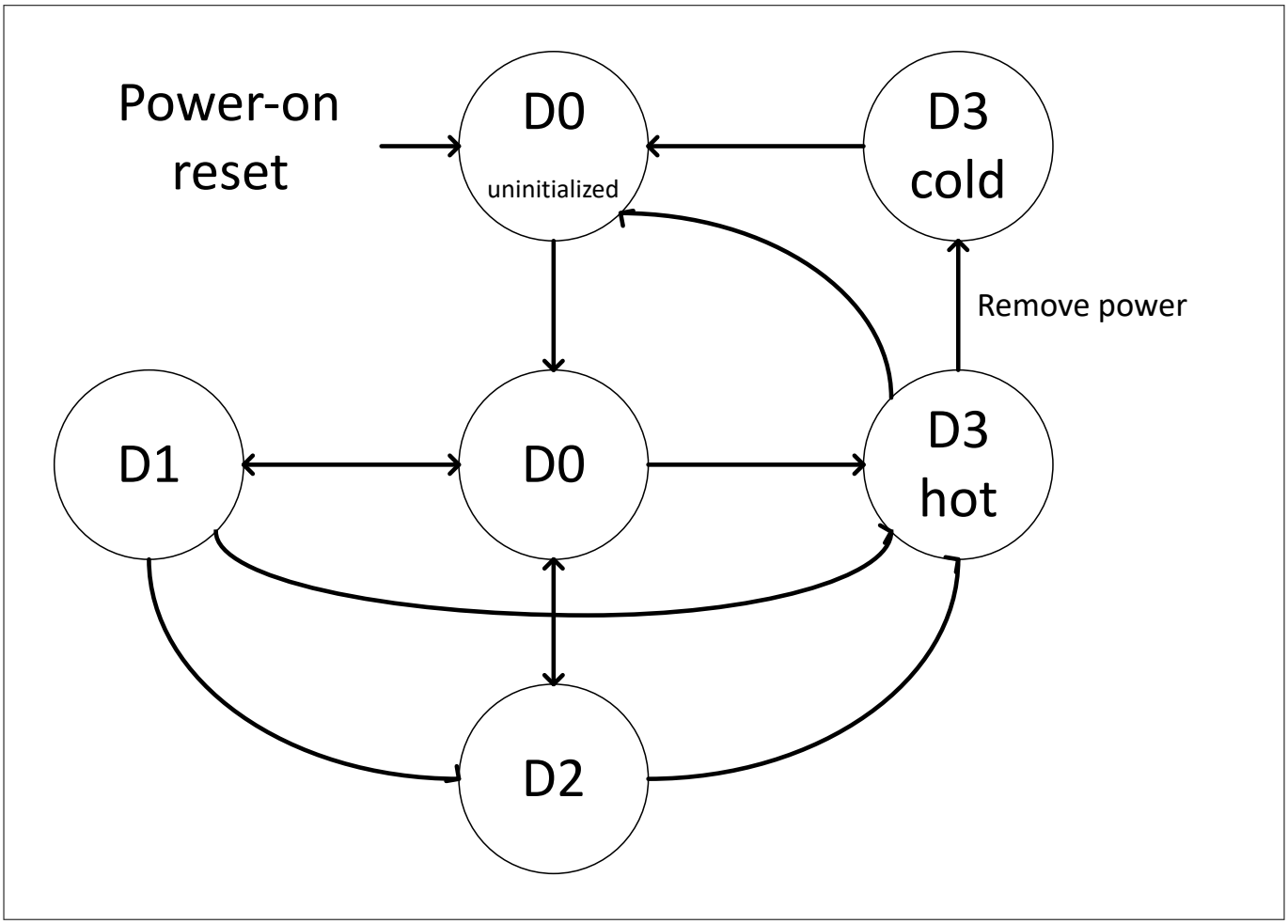


Figure 305 Device power management state flow diagram

Only the required D states: D0, $D3_{hot}$ and $D3_{cold}$ are supported in this module.

19.3.10.2 Link state power management

Several link power states are specified by the PCIe specification. PCIe controller supports all link states.

- L0s: a lower power state enabled by ASPM. The transmitters of two devices on a link are controlled separately. For example, a link works under L0s in the downstream direction while it works under L0 in the upstream direction
- L1: the L1 state is a higher latency, lower power "standby" state. L1 support is required for PCI-PM and optional for ASPM. Several substates of L1 are defined to further reduce the power consumption. L1 substates include L1.1 and L1.2
- L2/L3 ready: this is a staging point for L2 or L3
- L2: L2 is a auxiliary-powered link, deep-energy-saving state

- L3: it is a link-off state
- LDn: LDn is a transitional link down pseudo-state prior to L0

The transitions between different link states are explained by the diagram below .

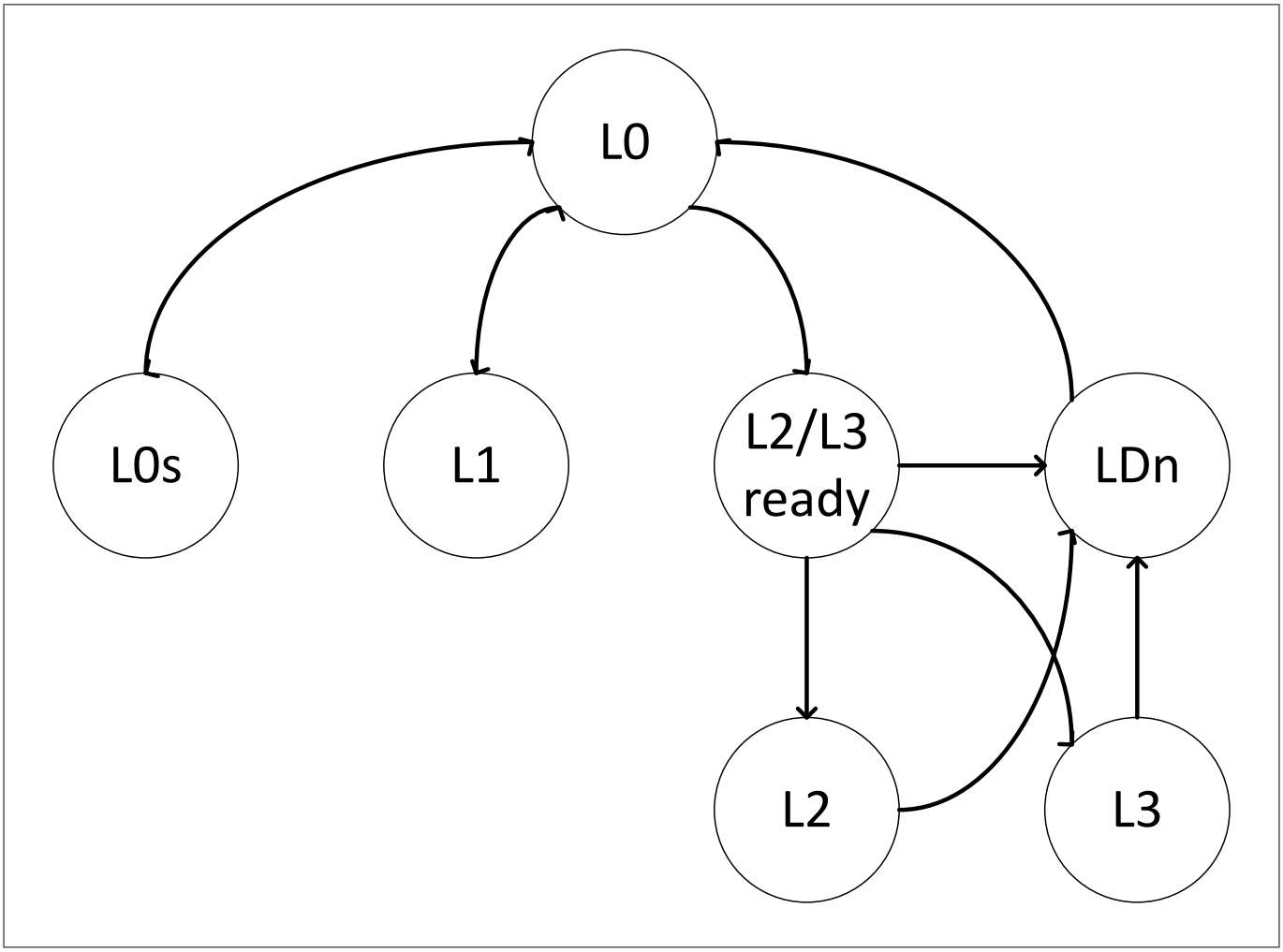


Figure 306 Link power management state flow diagram

ASPM is able to change link states autonomously.

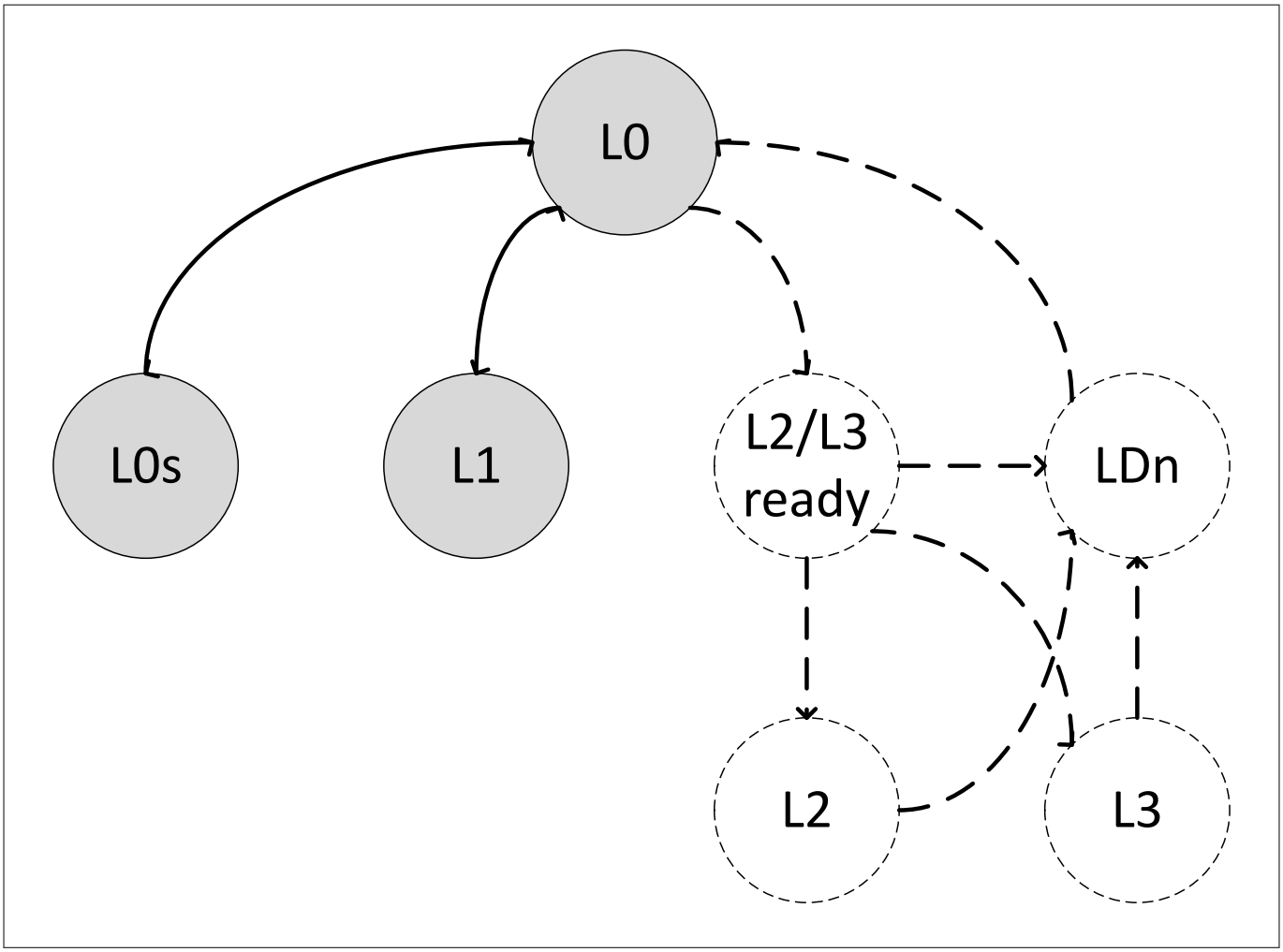


Figure 307 ASPM link state power management

The software driven mechanism can indirectly change the link state by setting the D states. The transition diagram by software driven mechanism is depicted in the diagram below.

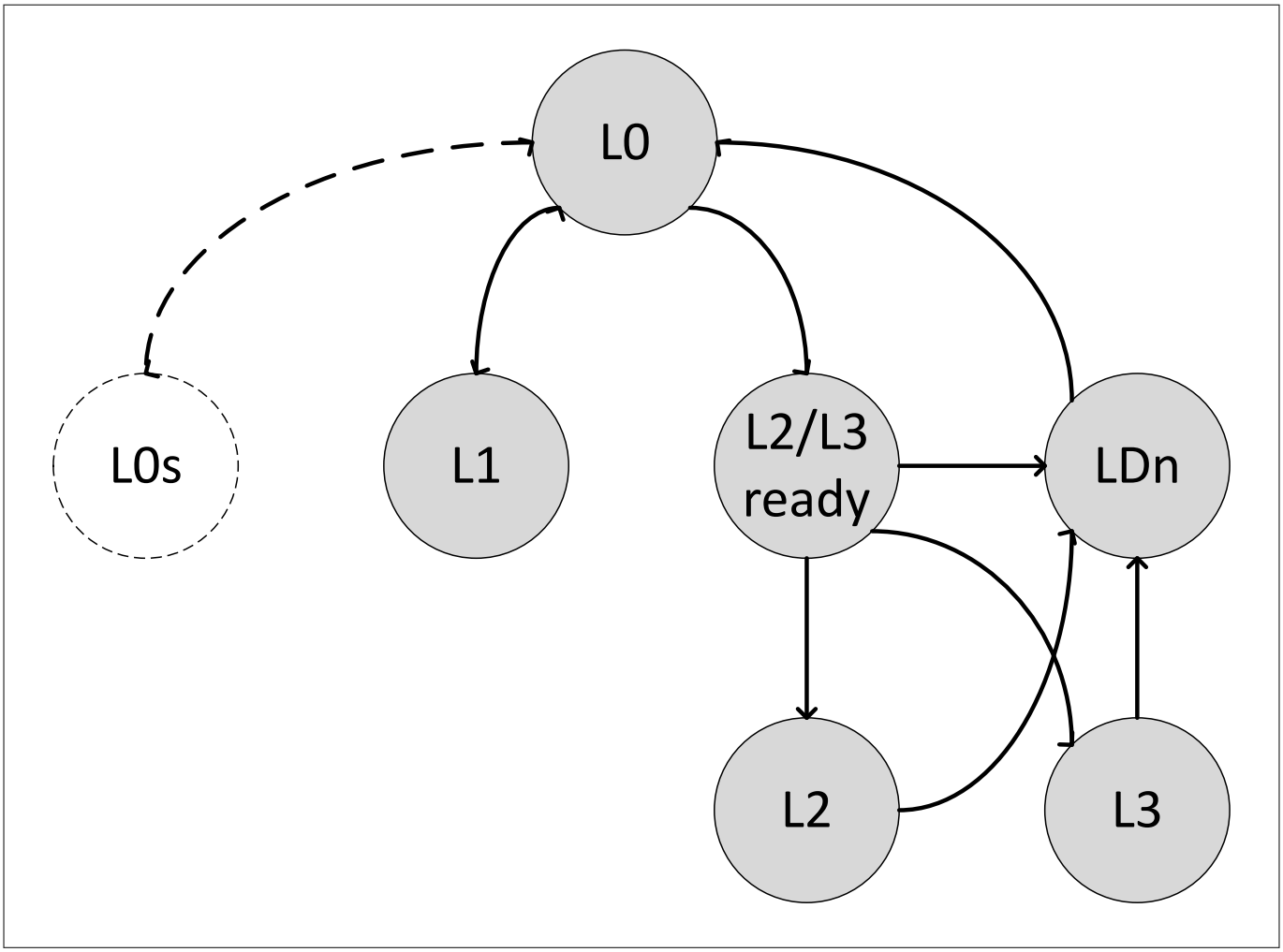


Figure 308 Software driven link state power management

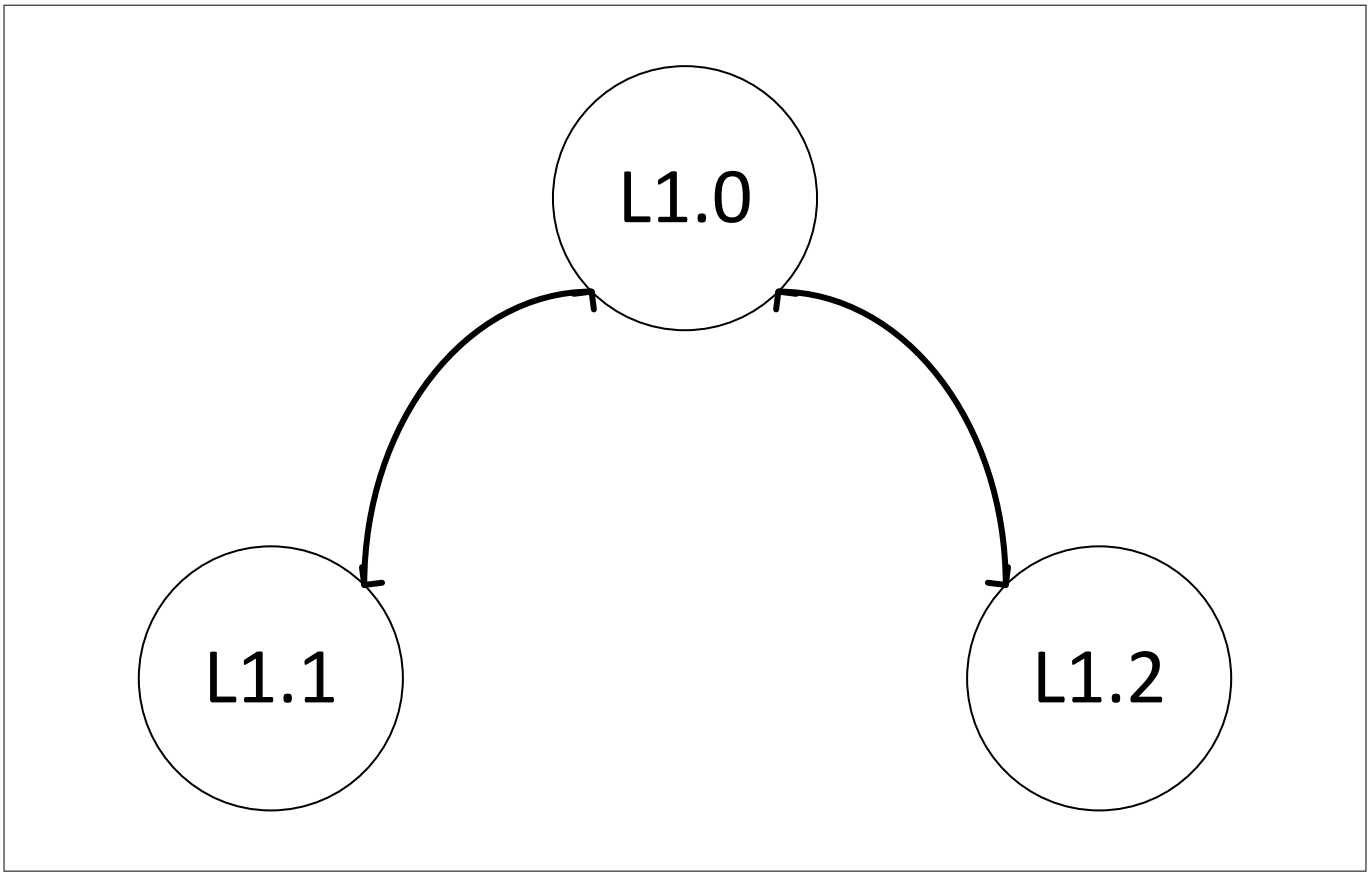


Figure 309 L1 substate power management

19.3.10.3 Impact of device states on link states

The power management state of a link is determined by the D state of it downstream component. These relationships are reflected in the following table. The link state change under D0 can only be triggered by ASPM while the link state change across D states are software driven. ASPM only applies to D0 state.

Table 850 Device and link state relationships

Downstream device state	Permissible upstream device state	Permissible link state	Link state change trigger
$D0_{uninitialized}$	-	L0	-
D0	D0	L0	ASPM
		L0s	ASPM
		L1/L1.x	ASPM
$D3_{hot}$	D0 - $D3_{hot}$	L1/L2	software-driven
$D3_{cold}$	D0 - $D3_{cold}$	L3	software-driven

According to the PCIe specification, link state transitions are allowed to be requested by certain roles, as explained in the following table.

Table 851 Link state change trigger capabilities for USP/DSP

Port type	Can trigger L0/L0s entry?	Can trigger L1 entry?	Can trigger L2/L3 entry?
USP	Yes	Yes, on any of these triggers: • Software programs non-D0 state • USPs idle timer expires (ASPM) • Application requests USP to enter L1	No
DSP	Yes	No	Yes

ASPM

When the USP is in L0 and detects idleness on the link for a specific amount of time (determined by the L0s Entrance Latency field in the “Ack Frequency and L0-L1 ASPM Control Register” ACK_F_ASPM_CTRL_OFF), it automatically transitions the link to the L0s power state. Each link direction (TX and RX) is processed separately. The controller exits L0s when there is traffic waiting to be sent.

When the USP is in L0 or L0s and it detects idleness on the link (both directions) for a specific amount of time (determined by the L1 Entrance Latency field in ACK_F_ASPM_CTRL_OFF), it automatically transitions the link to the L1 power state. The link partners must negotiate entry into the L1 link power state using PM TLPs and DLLPs. The L1 state achieves greater power savings but the exit latency from this state is higher than from L0s.

If the ASPM exit latencies from L0s or L1 are too high for the overall PCIe system data latency requirements, then the host software can disable ASPM using the ASPM Control Field of the Link Control register. The controller reports the exit latencies using the Link Capability registers.

L1 entry and exit

The power management state of a link is determined by the D-state of the USP. When the device state of the USP changes to D1, D2, or $D3_{hot}$ by writing to the PMCSR, the controller must initiate a link state transition to L1. L1 state is a bi-directional link low-power state. Both link partners must negotiate to enter this state. The L1-PM negotiation handshake uses the following DLLPs:

- PM_Enter_L1
- PM_Request_Ack

***Note**: asserting PM_CTRL.XFERPND doesn't prevent L1-PM entry, but causes immediate exit from L1-PM.*

The exit of L1-PM can be triggered by any of the below options:

- A DLLP or TLP is pending to be sent
- PM_CTRL.REQEXITL1 is asserted
- Link partner is requesting exit from L1
- PM_CTRL.XFERPND is asserted
- PM_CTRL.WKUPCMD is asserted to request the module to wake up, when the USP is programmed with capability to support PME. It can also be requested by the link partner to exit
- PM software (RC) requests a higher stage of power-down by writing to the PMCSR in the USP
- USP requests transmission of VDM, MSI/MSIX2, or LTR message
- The local software is requesting to send traffic
- The local software enables a DMA read or write channel or DMA controller is requesting data. Doorbelling any DMA read/write channel triggers L1 exit while writing to other DMA registers does not
- DSP initiates link disable or link retrain (by setting PCIE_CAP_LINK_DISABLE or PCIE_CAP_RETRAIN_LINK field in LINK_CONTROL_LINK_STATUS_REG to 1)

- DSP initiates a hot reset
- The local software requests a speed change (by setting DIRECT_SPEED_CHANGE field in GEN2_CTRL_OFF to 1)
- Clearing RADM_CLK_GATING_EN field in CLOCK_GATING_CTRL_OFF register
- Local software requests access to Configuration or Memory space mapped to the register group 6 and 7

L1 Substates

L1 Substates establish a Link power management regime that creates lower power substates of the L1 Link state.

- L1.0: The L1.0 substate corresponds to the conventional L1 Link state. This substate is entered whenever the Link enters L1
- L1.1: It uses a bidirectional open-drain clock request (CLKREQ#) signal for entry to and exit from this state. In the L1.1, the PLL and the Rx electrical idle detector are switched off
- L1.2: it uses a bidirectional open-drain clock request (CLKREQ#) signal for entry to and exit from this state. In the L1.2, the PLL, Rx electrical idle detector and Tx common mode are switched off

When the controller enters L1, the target L1 substate depends on several programming bits including PM Control/Status register bits and L1 Substates Control 1 register bits. The target L1 substate will be entered depending on the configuration according to "Entry conditions for L1 PM Substates and L1.0 Requirements" specified in the specification. A summary of these conditions are introduced in the following table. The target substate is always a deeper substate when the entry conditions of the deeper substate is fulfilled.

In order to enter the L1 substates, the application SW first sets PM_CTRL.CLKPMEN =1 and PCIE_CAP_EN_CLK_POWER_MAN field in LINK_CONTROL_LINK_STATUS_REG =1. Then it sets PM_CTRL.CLKRM =1, to indicate that the application is ready to have reference clock removed, and the controller can switch to aux_clk.

The Reported LTR is the maximum of the snoop/nosnoop latency values embedded in LTR messages transmitted by the upstream ports or received by the downstream port. The controller stores these values in the port logic LTR Latency Register (PL_LTR_LATENCY_OFF). When the requirement bit in the message is 0, the latency value is considered infinite (that is, the check with the threshold always pass).

Table 852 Target L1 sub-state

Transition trigger	ASPM PM L1.1 enabled	ASPM PM L1.2 enabled	LTR >= L1.2 threshold	PCI PM L1.1 enabled	PCI PM L1.2 enabled	Target L1 sub-state
PCI PM	-	-	-	0	0	L1
	-	-	-	1	0	L1.1
	-	-	-	-	1	L1.2
ASPM	0	0	-	-	-	L1
	1	0	-	-	-	L1.1
	0	1	0	-	-	L1
	1	1	0	-	-	L1.1
	-	1	1	-	-	L1.2

L2 and L3 entry and exit

L2/L3 entry is initiated after the RC calls power management software to initiate the removal of power and clocks. An EP must respond to the receipt of a PME_Turn_Off message by transmitting a PME_TO_Ack message. This is done by hardware automatically. Meanwhile, an interrupt (radm_pm_to_ack) is triggered when the PM_Turn_Off message is received. The application software needs to assert the PM_CTRL.RDYENTRL23 to allow the sending of PM_Enter_L23 DLLP, which will complete part of the negotiation handshake.

When the software is ready for transitioning to $D3_{cold}$, that is, loss of main power and reference clock, L2/L3_ready is entered and the downstream device begins preparation for the power and clock removal. After

main power has been removed, the link transitions to L2 if the auxiliary power is provided, or it transitions to L3 if no auxiliary power is provided.

The exit of L2/L3 is triggered by asserting the PM_CTRL.WKUPCMD bit to request the module to wakeup, when the USP is programmed with capability to support PME. It can also be requested by the link partner to exit.

Two wakeup mechanisms are implemented including Beacon (using in-band signaling) and WAKE# (using sideband signaling). Both will be triggered by the wakeup.

19.3.11 PCIe to AXI interface

There are one AXI/SRI master interface and one SRI/AXI slave interface. The AXI bus interface has the following basic properties:

- Address bus width: 32 bits
- Data bus width: 64 bits
- Maximum allowable master burst length: 16 beats

PCIe/AXI conversion

The PCIe controller provides an AXI bridging capability for directly adding a PCI Express link to an AXI system fabric. The bridge interconnects the AXI interfaces within an AXI-embedded system with a remote PCIe link. An AXI subsystem can only convey data, address and read/write information. It has no way to propagate PCIe concepts such as TLP type, posted/non-posted, function number, TLP attributes and so on.

The PCIe defines a flexible data transfer for example non-contiguous byte enable. It allows various sizes and transfers with special attributes. However, the bus protocol (AXI in this case) implemented on chip can only support a subset of possibilities. During the conversion from PCIe to AXI, several restrictions are applied. Therefore, the below section describes more about the transaction conversion from PCIe to AXI.

Inbound transfers

When the controller's AXI master wants to perform a transfer (whose length is greater than or equal to the databus width) it always uses beats of width equal to the databus size . Any unwanted data that is read is discarded. Therefore, only prefetchable[26)] memory is supported.

[26] For example, RAM where reading excess data does not prevent you from re-reading that data at a later time. Non-prefetchable memory would be a FIFO where data reads are destructive.

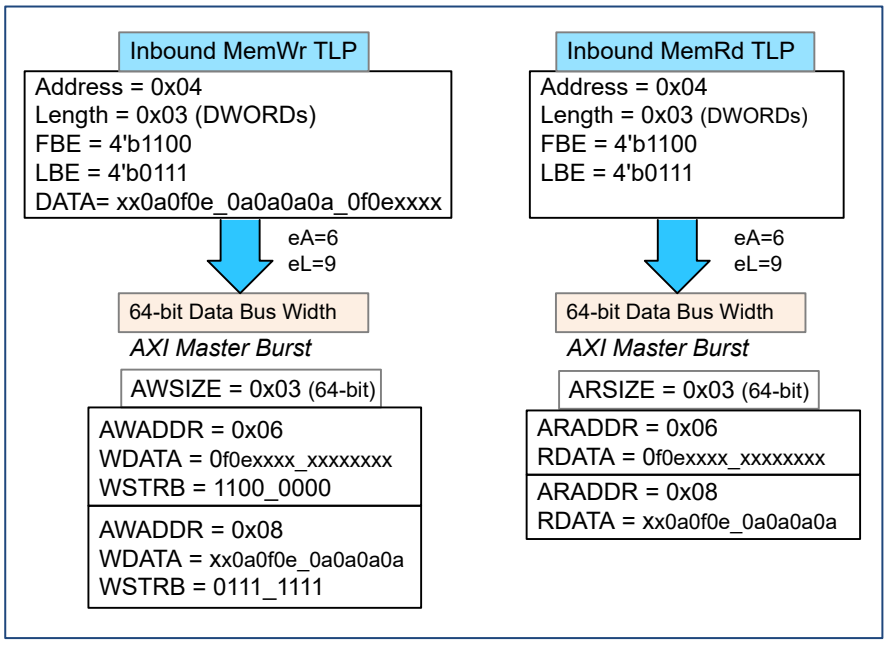


Figure 310 Examples of inbound AXI read and write transfers (eA = effective address; eL = effective length)

- Effective Length (eL): Effective PCIe TLP (byte) length excluding byte enable holes as per the first and last byte enable (FBE/LBE)
- Effective Address (eA): Effective PCIe TLP (byte) address after resolution of FBE/LBE and DWORD aligned TLP address becomes the AXI address

Inbound narrow reads

When the controllers AXI master executes a narrow (total number of read bytes < width of AXI databus) read transfer, it automatically selects one of two methods.

Table 853 Methods for executing narrow inbound reads at AXI master interface

Method and description	Automatically selected when
Approximate Full Size (Default Operation for Some Configurations) *Use for prefetchable memory.* Use an approximate byte count and perform one or more full-width accesses. With this method, the master uses the entire bus width . The controllers master reads in more bytes than the intended narrow transfer. Excess data is discarded and not forwarded to the link partner.	Single Beat Exact Size eA/eL criterion not satisfied
Single Beat Exact Size (Non-Default Operation) *Use for non-prefetchable memory (for example, FIFO).* Use an exact byte count and perform a single AXI access of width equal to the PCIe effective length (eL). A design complexity decision is taken to only support scenarios where the complete request can be completed within a single AXI beat.	Single Beat Exact Size eA/eL criterion[1)] is satisfied

1) The Single Beat Exact Size eA/eL criterion is: eL=2n and (eA is aligned to mstr_arsize).

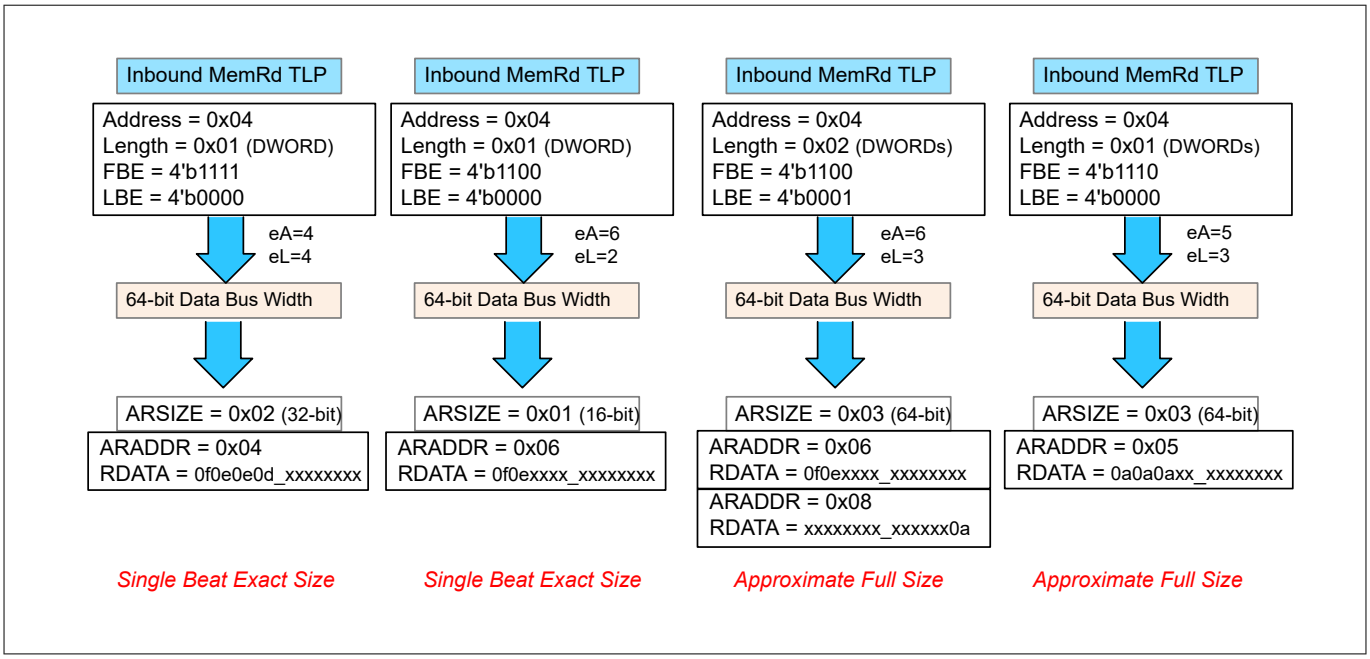


Figure 311 Examples of Inbound AXI Narrow Read Transfers (eA=effective Address; eL=effective Length)

Inbound narrow writes

When the controllers AXI master wants to perform a narrow WRITE transfer when the total number of bytes to be transferred is less than the width of the AXI data bus it uses always the full size method. It performs one or two accesses of width equal to AXI databus width. With this method, the controller master uses the entire bus width . The write strobes ensure a precise write to the intended write request address location.

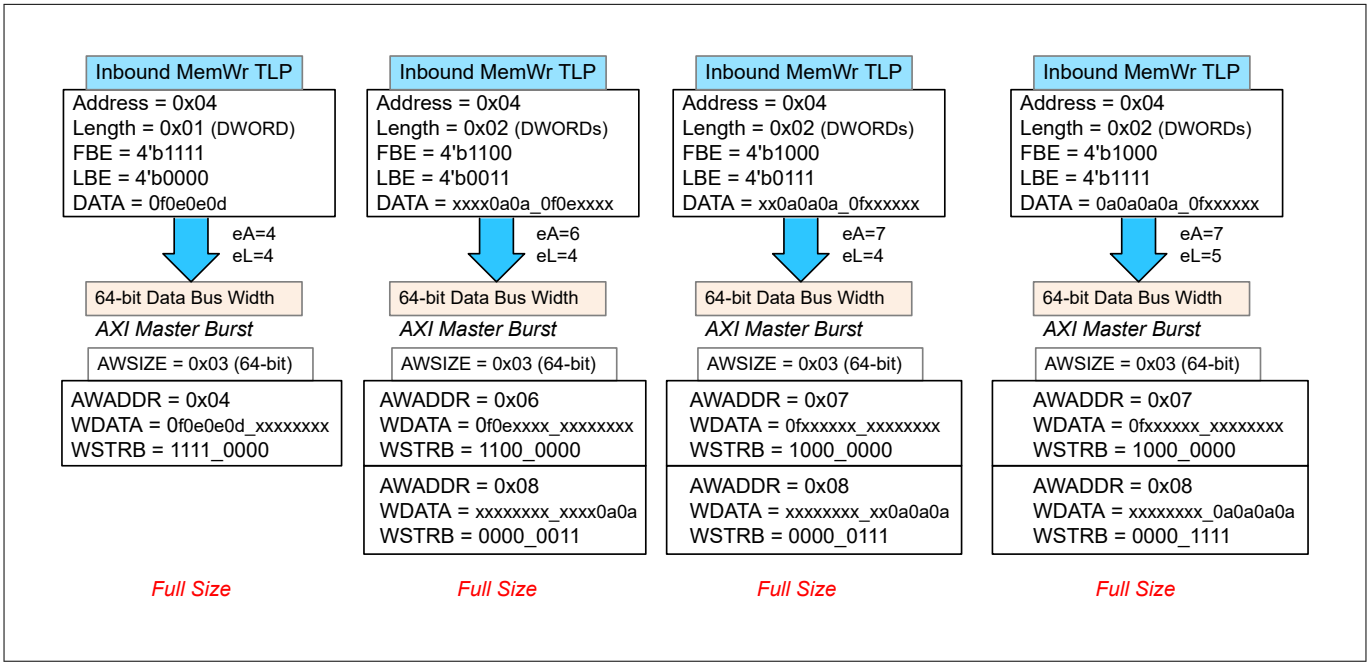


Figure 312 Examples of Inbound AXI Narrow Write Transfers (eA=effective Address; eL=effective Length)

Non-contiguous byte enable (NCBE)

The PCI Express Base Specification, Revision 5.0, Version 1.0 supports memory reads and writes with holes only when the read and write requests are one or two DWORDs long. The PCIe transaction header first and last byte

enable fields are used to signal which bytes are valid. A hole means that it is possible to have non-contiguous byte enables such as 0101_B. Non-contiguous byte enables occur when enabled bytes are separated by non-enabled bytes, for example, 0101_B. The rules for byte enables are given in Section 2.2.5 “First/Last DW Byte Enables Rules” of the PCI Express Base Specification, Revision 5.0, Version 1.0 Byte Enables are included with memory, I/O and configuration requests. The AXI protocol itself can support holes in write data as there are AXI write strobes.

Table 854 AXI inbound NCBE support for requests

	MemWr	MemRd
NCBE Supported	Yes	Yes
Size Restrictions	1,2 DWORD	1,2 DWORD

Table 855 Method and description

Method and description	
READS	The original PCIe request is first decomposed into a series of internal single byte request TLPs. These appear on the master interface according to the rules in table 28. Likewise, the responses to the individual decomposed requests are composed into a single (or double) DWORD of response data from that address. This method provides precision, but it sacrifices performance. To access n bytes, the master uses an n-beat byte-wise burst transfer.
WRITES	**Full Size** AXI write strobes are used. Perform accesses of width equal to AXI databus width. With this method, the controller master uses the entire bus width . The write strobes ensure a precise write to the intended write request address location.

Zero-length byte accesses

The PCI Express Base Specification, Revision 5.0, Version 1.0 supports zero-byte transactions as a means of flushing out previous transactions.

Rx Read/Write: Option 1 (DEFAULT_FILTER_MASK_2[3] =1)

The PCIe controller has a feature to terminate all zero-byte transactions when it is configured to do so. Enable the bit CX_FLT_MASK_HANDLE_FLUSH (default is 0) in the Filter Mask 2 register of the native controller to enable this feature. If the PCIe controller is configured to support termination of zero-byte transactions, it generates CPLs for zero-byte reads and return them automatically to the requester without sending the transaction to the AXI bridge. It also drops zero-byte writes. Transactions arriving at the PCIe receive queues are terminated with the flush function. Configuring the native PCIe controller to terminate zero-byte transactions is one way of avoiding handling zero-byte reads at the AXI interface. It has the disadvantage of not flushing out any previous transactions remaining in the AXI bridges queue buffer. In other words, the flush semantic terminates at the native PCIe controller’s receive queues.

Rx Read/Write: Option 2 (DEFAULT_FILTER_MASK_2[3] =0) - RECCOMENDED

A second possible means of handling zero-byte transactions is to let the AXI bridge receive the inbound zero-byte transaction from the native PCIe controller and present it to the AXI interface. The AXI specification (and the bridge) supports zero byte writes by setting all write strobes to 0. An inbound zero-byte read is handled as follows:

- If, AMBA_ORDERING_CTRL_OFF.AX_MSTR_ZeroLRead_FW = 0, an inbound zero-byte read is terminated at the AXI bridge master when any prior Posted transaction finishes implementing the PCIe flush semantics. When the zero-byte read terminates, it synthesizes a CPL transaction back to the requester
- If, AMBA_ORDERING_CTRL_OFF.AX_MSTR_ZeroLRead_FW = 1, an inbound zero-byte read is translated into a 1 byte read (from the original request’s address) on the AXI master and forwarded to the application. The resulting AXI response triggers the respective CPL back to the requester

19.3.12 Error handling

The controller supports the Baseline Capabilities, AER capabilities, and Advisory Non-Fatal Error Messaging as specified in Section 6.2, Error Signaling and Logging of the *PCI Express Base Specification, Revision 5.0, Version 1.0*. These include Correctable and Uncorrectable (Fatal and Non-Fatal) errors.

PCI Express errors can be classified as two types: Uncorrectable errors and Correctable errors. This classification separates those errors resulting in functional failure from those errors resulting in degraded performance. Uncorrectable errors can further be classified as Fatal or Non-Fatal.

For an RC port, it collects both the error messages from EP and the internal errors. The reporting of most errors is internal to the interrupt router or the SMU. No external error notifications are generated. One exception to this (for example) is unsupported request (UR) completion status.

EP reports error messages to the RC for reporting the detection of errors according to the severity of the error. It can also report the errors to the interrupt router or the SMU.

The error related interrupts are described in the below diagram. The additional information about the internally generated errors (SAFCORR and SAFUNCORR) can be found in the "Alarm" section.

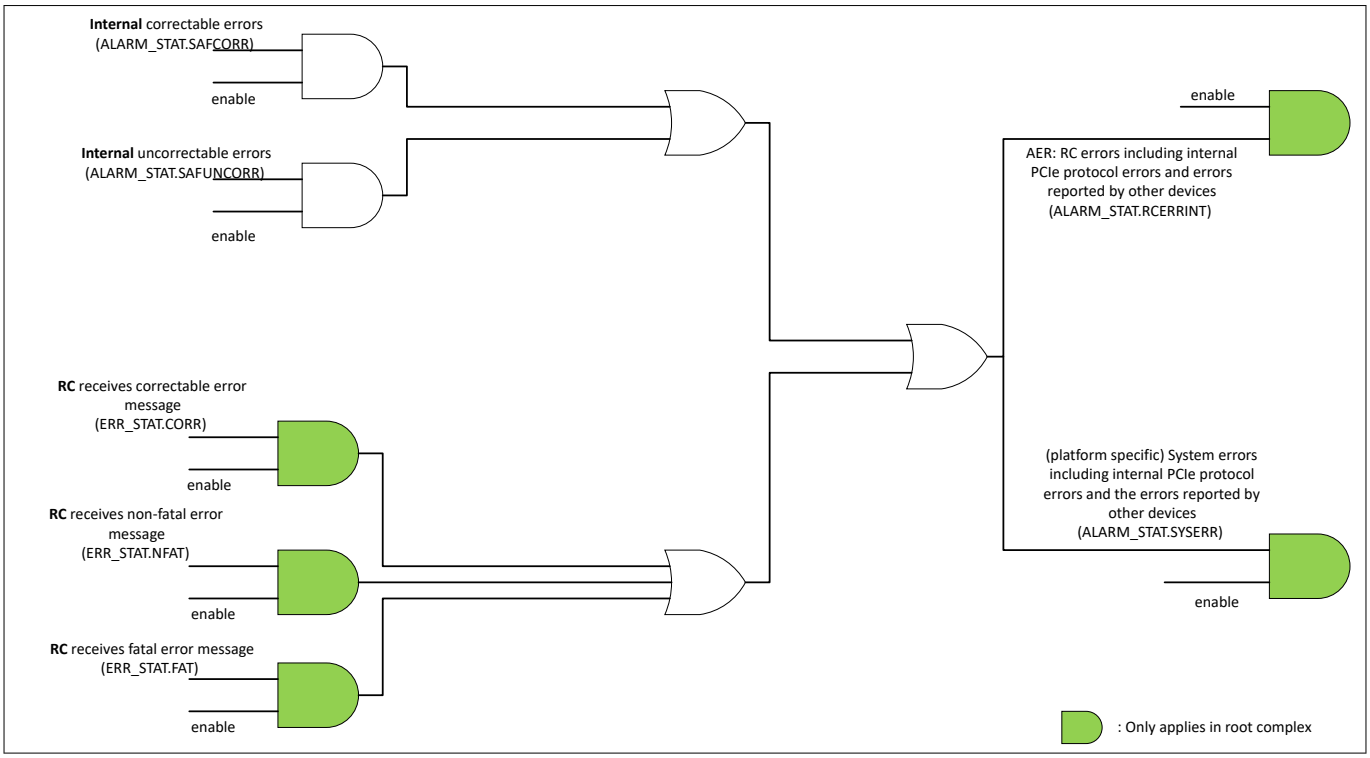


Figure 313 hierarchy of error interrupts

Error detection for received TLPs

The controller performs all mandatory error detections, the error reporting mechanism based on the *PCI Express Base Specification, Revision 5.0, Version 1.0*, and some optional error detections. When the controller detects an error in a received TLP, it performs the following by default:

- Discarding the TLP
- Generating a completion (for non-posted requests) with the completion status set to CA or UR
- Setting the status in the PCI-compatible status register
- Setting the status in the AER registers (when you enable AER)
- Generating an error MSG (upstream port only)
- For malformed TLPs credit is returned based on the buffer space which has been consumed by the TLP

AER multiple header logging

The controller supports the AER multiple header logging as specified in Section 6.2.4.2, “Multiple Error Handling (Advanced Error Reporting Capability)” of the PCI Express Base Specification, Revision 5.0, Version 1.0.

Physical Functions: by default, each physical function only logs the first header that caused an error. A valid queue entry is never overwritten. If software does not read the header log registers (HDR_LOGx) fast enough, you can miss the new header log information when the queue is full. The Completion Timeout Prefix/Header Log Capable bit is not implemented. Therefore no Header Log is stored when a Completion Timeout is received. Data Link Protocol, Surprise Link Down, Receiver Overflow, and Flow Control Protocol errors do not have TLP headers, so no Header Log is stored for those errors.

AXI and PCIe slave errors

Slave error mapping refers to what the controller (AXI) signals back to the S2A bridge that connects the SRI interconnect when it receives completion errors (CA, UR, CRS, and CPL-Timeout). The SLVERR and the DECERR are captured by the S2A bridge and will cause bus errors on SRI.

The AXI protocol provides response signaling for read transactions. The AXI responses are OKAY, SLVERR and DECERR. OKAY indicates that a normal access has been successful. SLVERR indicates slave error when the access has reached the slave successfully, but the slave wishes to return an error condition to the originating master. DECERR indicates decode error that there is no slave at the transaction address.

Table 856 AXI Slave Response for Erroneous Requests

Access Type	Transaction	Error Response to S2A
Wire	Read (Non-Posted) Write (Non-Posted)	OKAY (default) • The payload is set to 0xFFFF for reads SLVERR/DECERR • If your application sets the AMBA_ERROR_RESPONSE_GLOBAL field to ‘1’, and configures the AMBA_ERROR_RESPONSE_MAP field in the PCIEUSP_AMBA_ERROR_RESPONSE_DEFAULT_OFF register • SLVERR/DECERR will be signaled to the S2A bridge and result in a bus error
	Posted	OKAY

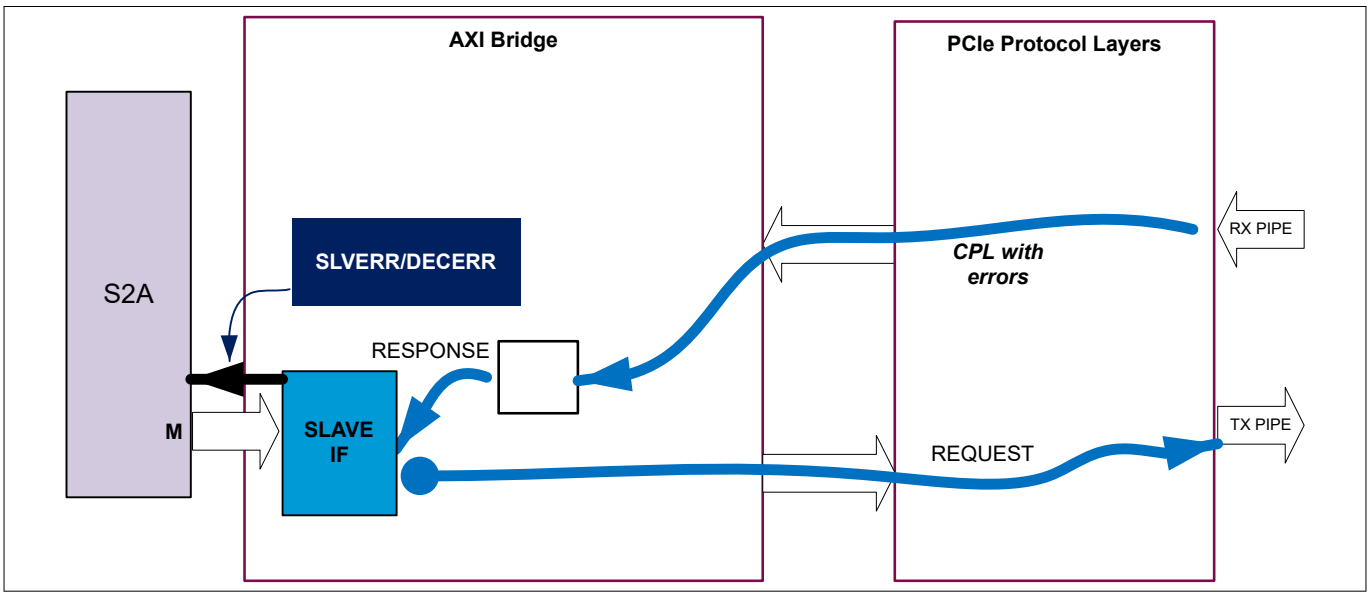


Figure 314 PCIe-to-AXI Slave Error Mapping

CRS CPL Error

A device in receipt of a Configuration Request following a valid reset condition may respond with a CRS Completion Status to terminate the Request, and thus effectively stall the Configuration Request until such time that the subsystem has completed local initialization and is ready to communicate with the host. Note that it is only legal to respond with a CRS Completion Status in response to a Configuration Request.

Receipt by the Requester of a Completion with CRS Completion Status terminates the Configuration Request on PCI Express. Root Complexes have the ability to report the receipt of CRS Completion Status to software, enabling software to attend to other tasks rather than being stalled while the device completes its self-initialization.

You can choose between different ways to handle a CRS completion error by programming AMBA_ERROR_RESPONSE_CRS field in the PCIEUSP_AMBA_ERROR_RESPONSE_DEFAULT_OFF register. The following options are available:

- 00: The AXI bridge always returns OKAY with non-FFFF data when a CRS completion is received.
- 01: The AXI bridge always returns OKAY with all-FFFF data when a CRS completion is received. This is useful when you want to detect that a completion is received with error.
- 10: The AXI bridge always returns OKAY with FFFF0001 data when a CRS completion is received for a Vendor ID read request, and OKAY with all-FFFF data for all other CRS completions. This is useful when you want to distinguish between CRS, and UR completions to Vendor ID read requests.
- 11: The AXI bridge always returns SLVERR or DECERR for all CRS completions, depending on the bit 1 value of AMBA_ERROR_RESPONSE_MAP field in the PCIEUSP_AMBA_ERROR_RESPONSE_DEFAULT_OFF register.

Reading Non-existent Vendor ID Register

The PCIe specification states that 0xFFFF is a reserved value for the Vendor ID register, to indicate that the function (containing Vendor ID register) is non-existent. An outbound CfgRd at the AXI bridge slave interface to a non-existent register (at the remote link partner) normally results in a UR response back from the PCIe wire without data. The AXI bridge slave automatically converts the completion with the UR response, to an OKAY response on the AXI slave interface, and also sets the completion data payload to 0xFFFF. This feature is supported for all CfgRd accesses (excluding byte-size accesses) to the Vendor ID register. You can modify this behavior using the AMBA_ERROR_RESPONSE_VENDORID bit of PCIEUSP_AMBA_ERROR_RESPONSE_DEFAULT_OFF.

AXI and PCIe master errors

Master error mapping refers to the AXI bus responses (for non-posted requests), and what the controller signals back to the remote link partner.

- Error-free responses (OKAY) are normally mapped to a CPL with Successful Completion (SC) status.
- Errored responses (SLVERR/DECERR) are always mapped to a CPL with CA status.

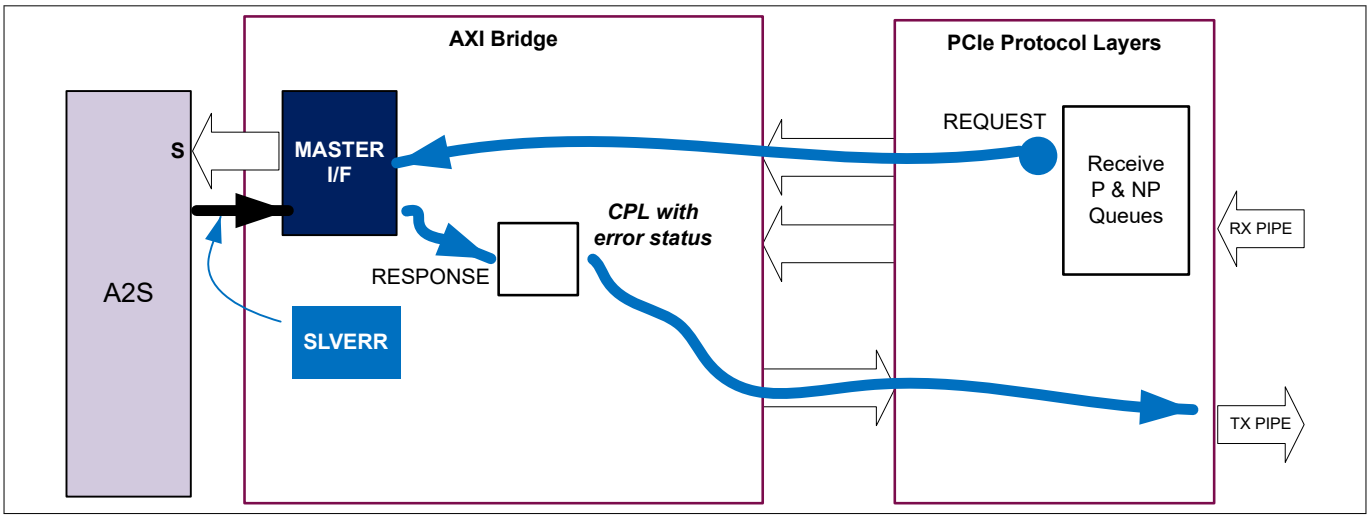


Figure 315 AXI-to-PCIe Master Error Mapping

For non-posted requests an errored master response results in the AXI bridge terminating the transmission of completions followed by a CA errored response completion. The AXI bridge master takes the following actions for each outstanding non-posted request when an errored master response data beat is detected:

1. Accepts the remaining master response beats without back pressuring the master response interface or canceling any of the remaining burst beats.
2. Sends a completion with CA status. No data is included in the completion.
3. Continues to re-format the master response packets until the last master response data beat for the last decomposed packet is detected.

19.3.13 Safety measures

This section describes the safety measures against random hardware faults for example soft errors.

19.3.13.1 Reliability, availability, and serviceability (RAS)

PCIe system Reliability, Availability, and Serviceability (RAS) ensures that failures in the underlying processes and hardware components do not cause any interruptions in the overall system operation. The PCIe RAS consists of two fundamental components:

- Reliable protocol architecture involving LCRC, ECRC, ACK and NAK handshake, and replay mechanisms
- Internal device-level Datapath and RAM protection (RAS DP) of data, header, and RAM control signals from the bus interface to LCRC extraction or generation points

19.3.13.1.1 ECRC

The controller uses the optional ECRC checksum, as defined in the PCIe specification, to protect the wire part of the link. The controller inserts the ECRC in the transmit logic and checks it in the receive logic.

Generation

The controller generates and inserts the ECRC by setting the ECRC_GEN_EN field of the ADV_ERR_CAP_CTRL_OFF register

Checking

By default, when the controller detects a TLP, with an ECRC error, it performs the following:

- Discards the TLP
- Generates a completion (for non-posted requests) with the completion status set to CA or UR
- Sets the status in the PCI-compatible status register
- Sets the status in the AER registers (when enabling AER):
  - When an ECRC error is detected by a PF, with ECRC checking enabled for any PF, ECRC error status is set for all PFs
- Generates an error message (upstream port only)

19.3.13.1.2 Data protection

For the parts of the controller outside of ECRC or LCRC protection, you can use the RAS DP feature.

Overview

- RAM Protection (ECC on data and Parity on address)

  An ECC checksum is calculated and added to data that is written to the RAM. When the data is read from the RAM, the controller recalculates the ECC and compares it against the value read from RAM. RAM ECC

protection is provided for all of the RAMs in the controller, except for the RAS-DES RAM. ECC corrects single-bit errors and reports (without correction) multi-bit errors

- Datapath Protection (ECC or Parity): for Tx traffic, the bus slave interface adds parity/ECC protection codes to traffic. Controller strips parity or ECC after LCRC Insertion on TX Path. For Rx traffic, controller inserts parity or ECC before LCRC extraction. The A2S or S2A bridges check the protection codes to check the data integrity
- AXI Interface (ECC): the standard AXI Interface Safety Block (ISB) in the AXI bridge module generates and checks ECC

Uncorrectable error handling

PCI Express defines two error reporting paradigms: the baseline capability and the Advanced Error Reporting Capability. Both are supported by this device.

PCI Express errors can be classified as two types: correctable errors and uncorrectable errors. Uncorrectable errors can further be classified as Fatal or Non-Fatal. This section describes an overview of errors. The details of standard PCIe error handling are not covered by this document.

Uncorrectable error handling

Upon detection of first Uncorrectable error (RAM and Datapath Protection), depending on value of AUTO_LINK_DOWN_EN field of the RASDP_ERROR_MODE_EN_OFF register, the controller takes one of the two actions:

- Forces the PCIe link to go down (AUTO_LINK_DOWN_EN = 1)
- Enters RASDP Error Mode (AUTO_LINK_DOWN_EN = 0)

PCIe link down

If the controller is configured to cause the link to go down upon detection of an uncorrectable error, the controller resets and resumes normal operation.

RASDP error mode

On entering RASDP error mode, the controller sets ERR_CTRL.MSTERRM or ERR_CTRL.SLVERRM. Wire access to controller registers is suspended to prevent corruption of the register configuration space.

TLP handling in RAS DP Error mode:

- Rx TLP Handling
  - Rx errored TLPs are forwarded to the bus
  - Subsequent Rx TLPs are not guaranteed to be correct; the user must discard all TLPs
- Tx TLP Handling
  - Bad TLP is nullified (EDB inserted and LCRC inverted) and transmitted
    - The controller might nullify TLPs that are received prior to the bad TLP
  - Subsequent TLPs are nullified and transmitted
  - In-progress replay is nullified and transmitted
  - Subsequent replays are silently dropped

The controller attempts a best-effort recovery in RASDP error mode (after detection of an uncorrectable error).

Credits accuracy is not retained under uncorrectable error conditions. The controller implements a best-effort approach in accounting for credits under uncorrectable error conditions. However, credit inaccuracies are unavoidable in such conditions. This might result in performance degradation or triggering of overflow protection mechanisms.

Error Logging:

- Uncorrectable error for example 2-bit ECC and all parity errors are logged
- Uncorrectable internal error reported (AER)
- The controller only reports uncorrectable errors when it exits RASDP error mode

Your application software can instruct the controller to exit RASDP error mode by writing to RASDP_ERROR_MODE_CLEAR_OFF register through SPB. Your application must perform any cleanup it may deem necessary prior to instructing the controller to exit RASDP error mode. In RASDP error mode, some uncorrectable errors can be severe enough to eventually cause the link to go down. If link down occurs, the controller resets and resumes normal operation.

Correctable error handling

The controller takes the following action to handle correctable (RAM and Datapath Protection) errors:

- Correctable[27] error is logged
- Correctable internal error reported (AER)
- TLP is modified or corrected
- Errors reported when detected (immediately)
- Error correction enable or disable control (per TX or RX direction and per-layer)[28]

19.3.13.2 CDM register check

The controller supports automatic CDM register checking to ensure that CDM registers are not corrupted. The iATU and DMA port logic registers are also checked.

The CDM registers start from 3000_H to 8000_H in the local address offset. All the physical CDM registers are checked.

CDM register check features and limitations

- The register check starts when PL_CHK_REG_CONTROL_STATUS_OFF.CHK_REG_START is set to 1
- Continuous or run once modes (PL_CHK_REG_CONTROL_STATUS_OFF.CHK_REG_CONTINUOUS)
- Programmable start and end check addresses through PL_CHK_REG_START_END_OFF register
- CHK_REG_COMPLETE field of PL_CHK_REG_CONTROL_STATUS_OFF indicates check has finished
- CHK_REG_COMPARISON_ERROR field of PL_CHK_REG_CONTROL_STATUS_OFF indicates comparison failure
- Failing address and PF of the first comparison error found is recorded in registers PL_CHK_REG_ERR_ADDR_OFF and PL_CHK_REG_ERR_PF_VF_OFF respectively

  ***Note**: Only the address of the first error found is recorded. The checking sequence proceeds until complete, unless software clears the CHK_REG_START field of PL_CHK_REG_CONTROL_STATUS_OFF register*

- The SPB or wire accesses to the CDM registers takes priority over any CDM register check sequence
- Built in self-test functionality checks the comparison logic

  CHK_REG_LOGIC_ERROR field of PL_CHK_REG_CONTROL_STATUS_OFF register indicates error in comparison logic
- The controller only reports the error (through AER: Uncorrectable Internal Error or upstream error message for EP product); it does not take any action to correct the error

CDM register check operational details

CDM register checking is controlled by a FSM which sequences checking of CDM register values against shadow registers, in CDM_B (*Register content check*). The CDM register checker's self-test functionality checks the comparison logic, by injection of an error at one particular address location (*Error Injection*). It also checks address increments during the comparison sequence (*Address Check*).

Register Content Check (CDM versus CDM_B: The CDM register checker compares CDM and CDM_B data. If a mismatch is found:

27 1-bit ECC errors.

28 If the controller is capable of single bit error correction, disabling ECC error correction does not change the nature of the errors and their reporting. A single-bit error is reported as a correctable error although correction might be disabled.

- Signal cdm_reg_chk_cmp_err is asserted, and it is connected to the SMU alarm
- CHK_REG_COMPARISON_ERROR field of PL_CHK_REG_CONTROL_STATUS_OFF register is set to 1 to indicate comparison error

The checking sequence as follows:

1. The address range specified in register PL_CHK_REG_START_END_OFF is checked for comparison errors. Check is done for each PF as well
2. If accesses to DMA registers are matched MEM/IO requests through BAR, rather than legacy indirect register addressing, DMA port logic registers are checked for all configured channels
3. If accesses to iATU registers are matched MEM/IO requests through BAR, rather than legacy indirect register addressing, iATU port logic registers are checked for all configured regions

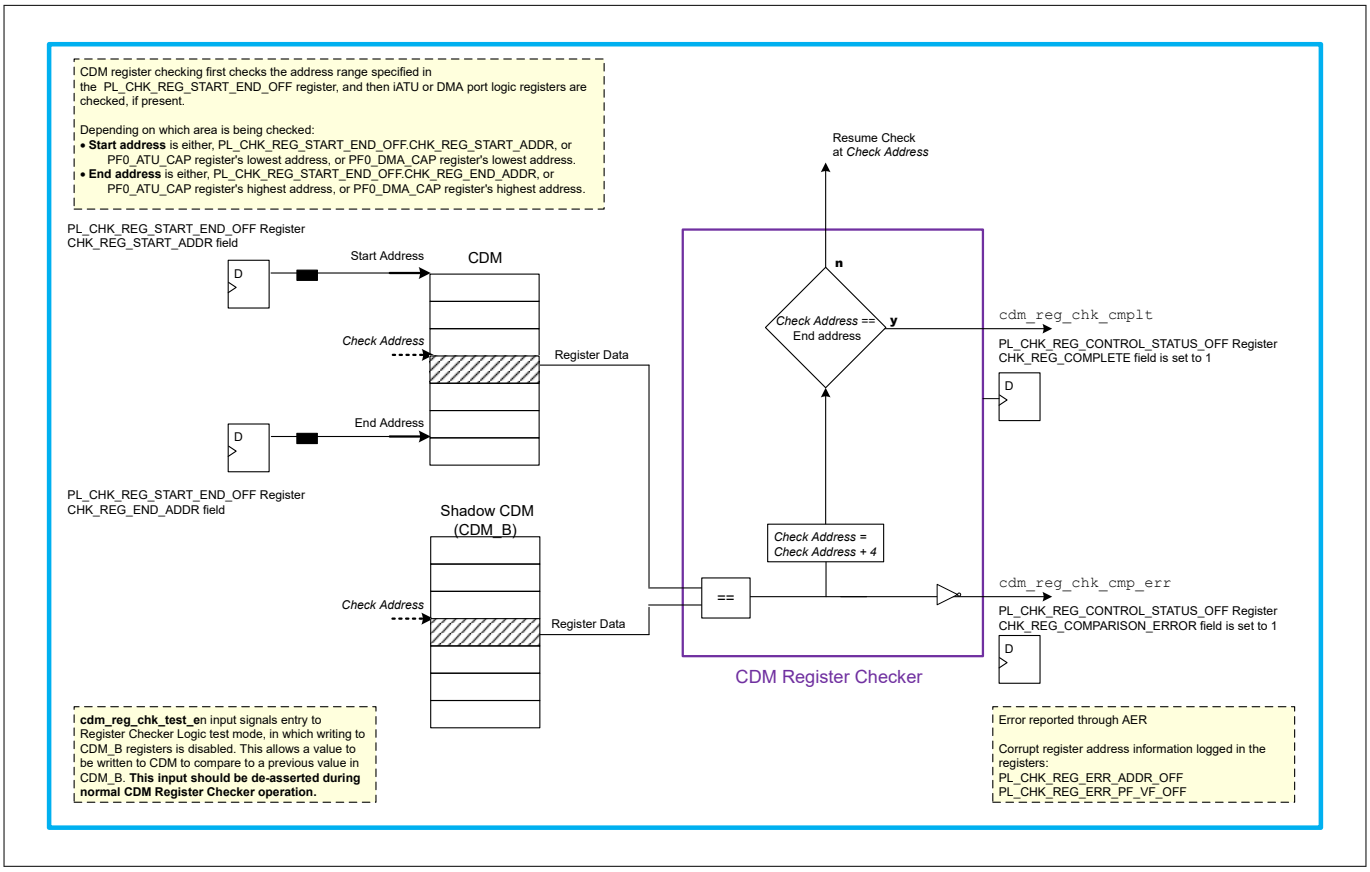


Figure 316 Register content check (CDM versus CDM_B)

Error Injection: At the end of *Register Content Check*, the comparison logic is checked by comparing data at address 100_H (offset from the beginning of CDM) in CDM against the inverted value of data in CDM_B at the same address. If the value and the inverted value do not match:

- Signal cdm_reg_chk_logic_err is asserted, and it is connected to the SMU alarm
- CHK_REG_LOGIC_ERROR field of PL_CHK_REG_CONTROL_STATUS_OFF register is set to 1 to indicate an error in comparison logic

Address Check: During *Register Content Check* and *Error Injection*, each time the address is incremented, it is checked. If the increment is not equal to 4 (corresponding to the next 32-bit register address) then:

- Signal cdm_reg_chk_logic_err is asserted, and
- CHK_REG_LOGIC_ERROR field of PL_CHK_REG_CONTROL_STATUS_OFF register is set to 1 to indicate an error in comparison logic

19.3.13.3 Tx interface progression detection

This feature provides a mechanism for the controller to inform the application about the lack of progression on AXI slave and master interfaces. It also monitors the internal MSG gen. interface and the LBC CFG completion interface. Progression on the data transmission interfaces is not expected to halt under normal operating circumstances; it might halt if soft errors or silicon failures occur, or if your application logic inadvertently hangs the TX or AXI application interfaces.

Related information

Transmit TLP on page 2711

19.3.13.4 Alarms

There are two SMU alarms, data uncorrectable alarm and configuration uncorrectable alarm. Please always refer to the device's safety manual to find out which safety mechanism of this functional block need to be considered for safety relevant applications.

PCIE data uncorrectable error alarm covers both the error signals cfg_safety_uncorr and cfg_safety_corr. The cfg_safety_uncorr includes RASDP, Tx progression detection, CDM register check and PCIe protocol uncorrectable errors. These error signals can be masked respectively and logically to cfg_safety_uncorr as shown in the following figure. The cfg_safety_corr includes both PCIe protocol correctable events and RSADP correctable events. Both signals are reported as the data uncorrectable alarm to SMU.

The controller provides dedicated functional safety interrupts (cfg_safety_uncorr and cfg_safety_corr) to route the safety related indications to your application. These interrupts report events of different severity; cfg_safety_uncorr reports uncorrectable errors and cfg_safety_corr reports correctable errors. These interrupts have independent status and mask bits in SAFETY_STATUS_OFF and SAFETY_MASK_OFF registers, respectively. The functional safety interrupt outputs provide indications from two different event categories:

- PCIe protocol defined events
- Internal safety mechanisms events
  - RAS Data Protection
  - CDM Register Check
  - Tx Interface Progression Detection

Events signaled from both the categories are recorded in the status register regardless of the specific event mask setting. The mask controls whether an event causes the safety interrupt (cfg_safety_uncorr or cfg_safety_corr) to be asserted or not.

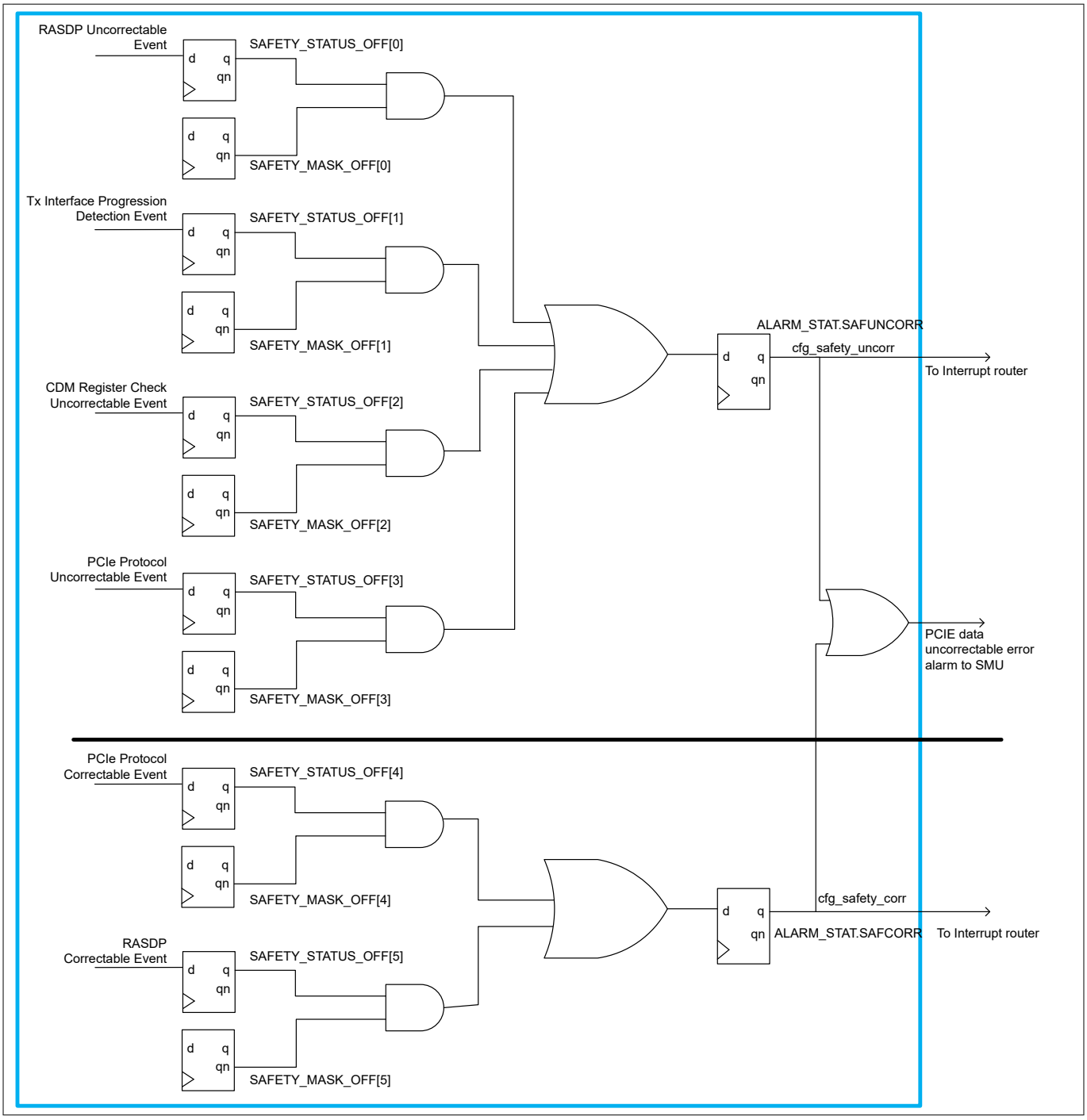


Figure 317 Events reported by cfg_safety_uncorr and cfg_safety_corr

The CDM register check feature is designed for the configuration monitoring. The errors (cdm_reg_chk_cmp_err or cdm_reg_chk_logic_err) in registers are reported as configuration uncorrectable alarm.

Additionally, errors from the S2A, A2S bridges including the data phase and address phase and FPI write error are also reported to the SMU as data uncorrectable alarm. The status and mask control are configured in ALARM_STAT, ALARM_INTCLR and ALARM_INTEN.

19.3.14 Clock and reset

The PCIe module can be reset either by local reset or PCIe reset. The local reset is the standard reset supported by AURIX™ and can only be triggered locally. The PCIe reset includes PERST# reset and hot reset.

19.3.14.1 Clocking

The clock reset module in the PCIe controller is responsible for the clock generation and distribution. It generates different reset signals for the reset of the PCIe controller.

The clock reset module receives f_{OSC} from CLOCKING and f_{PHY} from the HSPHY to generate several clocks including `core_clk`, `core_clk_ug`, `muxd_uax_clk`, and `aux_clk_g`. Normally, the controller uses the `core_clk` that is provided from the clock reset module (module reset) in normal mode. The switch selects the `aux_clk` in certain low power states, and power on reset or fundamental reset .

The clock reset module implements architectural clock gating on `core_clk`, which permits the disabling of `core_clk` in certain low power states, allowing the removal of REFCLK. In this case, the f_{PHY} disappears and the PCIe controllers uses only the f_{OSC} clock.

The HSPHY uses the standard PCIe reference clock (f_{REF_CLK}) 100Mhz to generate clock (f_{PHY}) to the PCIe controller.

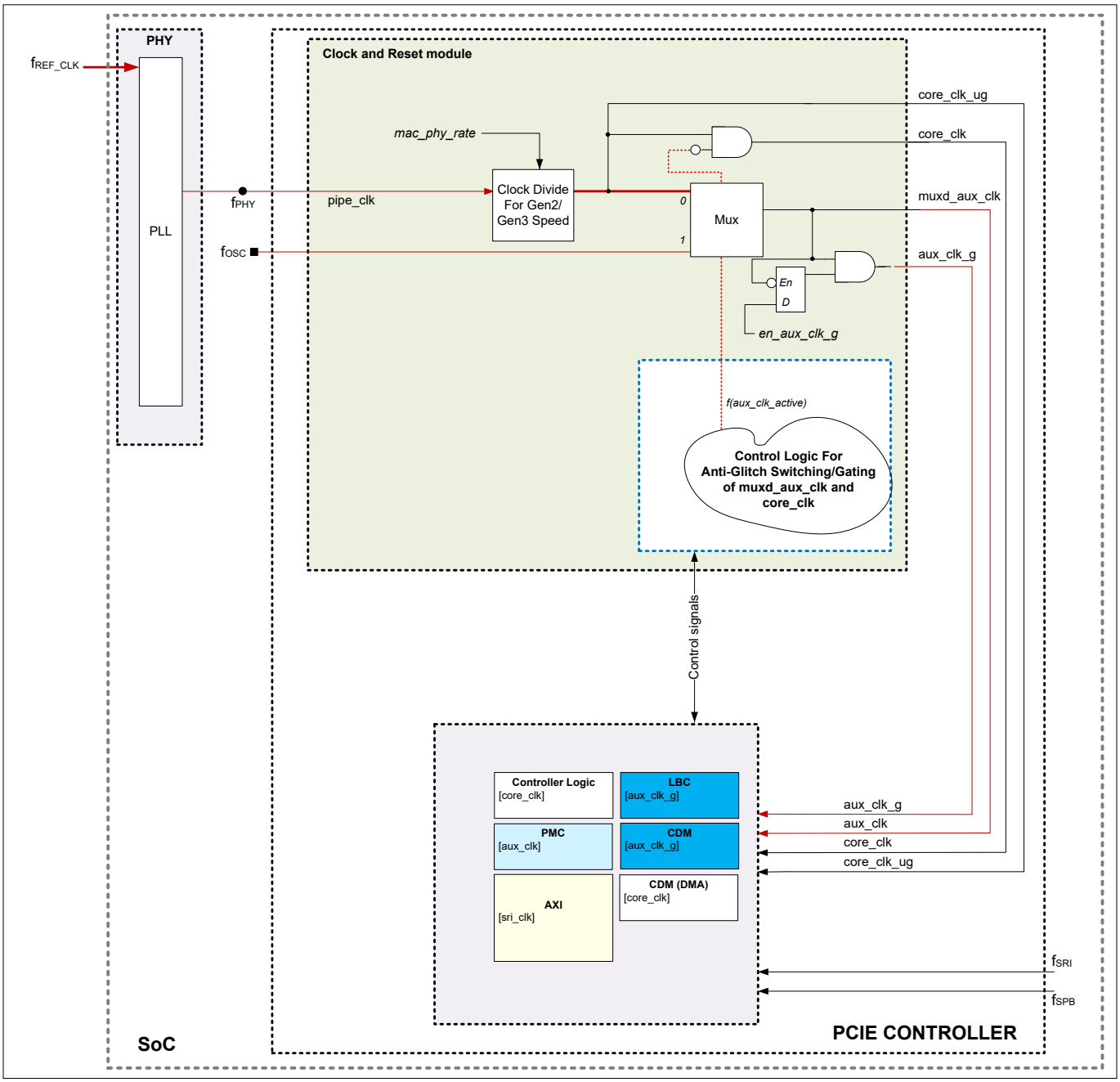


Figure 318 Clock overview of PCIe controller

Table 857 Generation of internal clock signals by the clock and reset module

Clock	Notes
core_clk	• This is the primary clock for the controller • It can have a frequency of 62.5, 125 or 250 MHz for Gen1, Gen2 and Gen3 respectively • The `core_clk` is derived from the f_{PHY} output of the HSPHY (whose TX PLL uses the platform reference clock REFCLK) • The `core_clk` signal must be gated-off when `muxd_aux_clk` is being switched to run at low frequency
core_clk_ug	• An ungated version of `core_clk` (called `core_clk_ug`) exists because a small number of controller registers must receive `core_clk` during switching to-and-from low-power states
aux_clk	• This clocks the PMC • It is derived from the f_{OSC} and it has the same frequency as the f_{OSC}
aux_clk_g	• This is a gated version of `aux_clk` that is used to clock the CDM registers and the LBC

19.3.14.2 Hot reset

A DSP can hot reset a USP controller by sending two consecutive TS1 ordered sets with the hot reset bit asserted, or by asserting the LNK_CORECTRL.HOTRST bit.

Alternatively, during normal operation, the link might fail and go down. After this link-down event, the controller requests the clock and the reset module to hot-reset the controller. There is no difference in the handling of a link-down reset or a hot reset.

To prevent application from hanging when the controller is reset, the controller must cleanly terminate all transactions that are still in-progress. The controller automatically executes the automatic flushing process before the hot reset. During this period, the access to the slave interfaces from the SRI and SPB will also be signaled as error to prevent potential bus hanging. To disable automatic flushing, set the AUTO_FLUSH_EN field in LINK_FLUSH_CONTROL_OFF to '0'.

A link reset interrupt to notify the application is available. The status is indicated by the LNK_STAT.LINKRQRST and controlled by the LNK_INTEN.LINKINTEN.

19.3.14.3 PERST# reset

The PERST# is the fundamental reset for PCIe controller but not for the SoC. The assertion of the PERST# will not lead to reset of SoC.

An unexpected assertion of the PESRT# resets the PCIe controller while other modules on the SoC run normally. This may lead to scenarios for example CPU accesses the resetting PCIe controller or on-going SRI transactions are interrupted by the reset and cannot be finished. These scenarios will cause hanging of the bus. In order to avoid this, the PERST# reset will not reset the PCIe controller immediately but be delayed until the current bus transaction completes. The bus accesses including SRI and SPB to a resetting PCIe controller will result in error signaling instead of hanging.

An interrupt is also possible to notify the SW that the PERST# is asserted.

19.3.14.4 Register accessibility during reset

On a reset event like hot reset, PERST or a link reset a link interrupt is triggered to notify the application. In such an event, PCIE controller registers are accessible until the reset sequence has completed. To check for the completion of the reset sequence, user can poll the following registers in the described order:

1. Wait until LNK_DVCTYP.PWD becomes a zero
2. Wait until PM_STAT0.LTSSMSTAT has a value greater than 0

   After the above two steps, the registers are accessible again

19.3.15 Precise time measurement

The controller supports the optional PTM feature as defined in Section 6.22, Precision Time Measurement (PTM) Mechanism of the PCI Express Base Specification, Revision 4.0, Version 1.0.

This feature enables the coordination of timing information across multiple devices with independent local time-bases. The PTM synchronization is initiated from a USP to a DSP. The controller handles the entire timestamp update process without software intervention. Each current ($t_{\#n}$) and previous timestamp ($t_{\#n-1}$) value as well as other context information can be read from the VSEC registers by software.

The controller implements 1 µs and 100 µs timers as per the requirements of the PCI Express Base Specification, Revision 5.0, Version 1.0. The controller also implements a 10 ms timer which is used for automatic update of PTM context by the Requester.

19.3.15.1 Transmit and receive latencies

A key operational goal of PTM is to stamp the time at which a message is transmitted from the physical boundary of the PCIe device.

The Tx latency is the time that it takes a message to travel from the boundary of the controller to the boundary of the PCIe device. It is added to the time at which the message left the controller.

The Rx latency is the time that it takes a message to travel from the boundary of the PCIe device to the boundary of the controller. It is subtracted from the time at which the message reaches the controller.

The Latency registers may be written to at any time with a more accurate value if required. For example:

- t1 = Local PTM Clock time when message leaves controller + PTM_REQ_TX_LATENCY_OFF.PTM_REQ_TX_LATENCY
- t3 = Local PTM Clock time when message leaves controller + PTM_RES_TX_LATENCY_OFF.PTM_RES_TX_LATENCY
- t2 = Local PTM Clock time when message received by controller - PTM_RES_RX_LATENCY_OFF.PTM_RES_RX_LATENCY
- t4 = Local PTM Clock time when message received by controller - PTM_REQ_RX_LATENCY_OFF.PTM_REQ_RX_LATENCY

***Note**:* *The Tx latency includes all delays from the moment timestamp is captured, to the time the message reaches the port pins. The Rx latency includes all delays from the time the message reaches the port pins, to the time the timestamp is captured. These delays depend on the characteristics of the PHY used, skew on the link, and deviation between the reference clocks of the two ports. There may be a dependency on link width and the current link speed also.*

For best accuracy the Tx and Rx latency values should be programmed during initialization, for each supported link speed. The controller automatically selects the Tx and Rx latency based on current link speed.

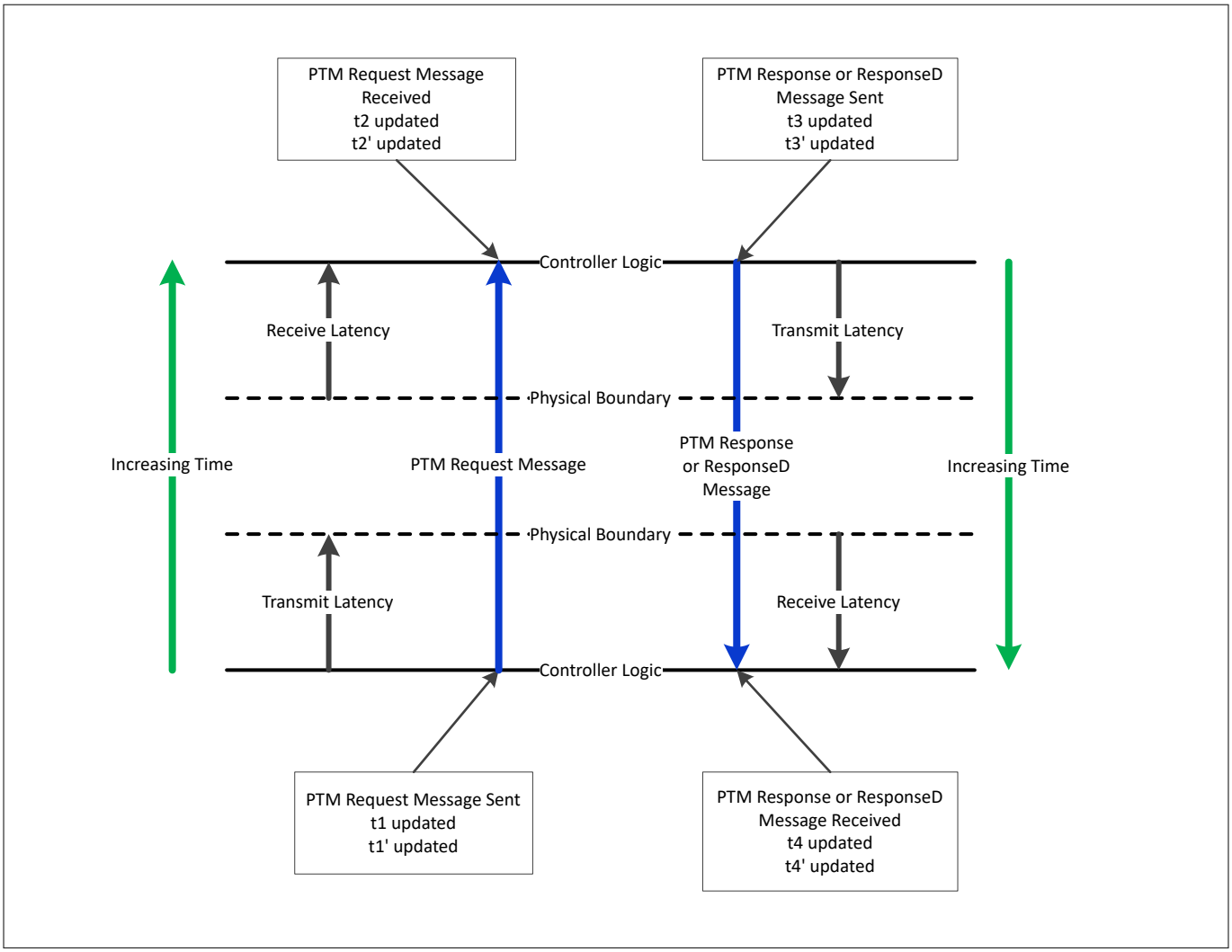


Figure 319 Tx and Rx latencies

Programming Tx and Rx latencies for each link speed

The Tx and Rx Latency registers are programmed through an indirect addressing scheme using an index register (PTM_[RES|REQ]_LATENCY_REG_SEL_OFF), to reduce the address footprint in the PCI Express extended configuration space. The index register has Responder/Requester Latency Register Write Select field (PTM_[RES|REQ]_LATENCY_REG_SEL), referred to as index, to determine which Tx or Rx Latency viewport register (PTM_RES_[TX|RX]_LATENCY_OFF) to program/read for each link speed. The following table provides the index to link speed mapping.

Table 858 Index to link speed mapping

PTM_[RES|REQ]_ESM_SEL	PTM_[RES|REQ]_LATENCY_REG_SEL	Link speed (GT/s)
PCI express speed mode		
0	0	2.5
0	1	5
0	2	8

If the Tx and Rx latency registers are not programmed, the default latency values are the same for all speeds, and are set by configuration parameters, as follows:

- Tx Latency = CX_PHY_TX_DELAY_PHY + CX_PHY_TX_DELAY_MAC
- Rx Latency = CX_PHY_RX_DELAY_PHY + CX_PHY_RX_DELAY_MAC

***Note**: The default values for Tx and Rx latency registers are defined in clock cycles. You must program the appropriate registers to set the latencies in ns.*

Programming examples

Example 1: To setup PTM Requester Tx Latency to 56 ns and Rx Latency to 334 ns for 2.5 GT/s

1. Setup speed mode

Write 0 to PTM_REQ_ESM_SEL, to set PCI Express speed mode

2. Setup the index

Write 0000_0000_H to PTM_REQ_LATENCY_REG_SEL to select 2.5 GT/s Latency viewport Tx and Rx registers

3. Write the Tx/Rx Latency values

Write 0000_0038_H to PTM_REQ_TX_LATENCY to set 2.5 GT/s Tx latency to 56 ns. Write 0000_014E_H to PTM_REQ_RX_LATENCY to set 2.5 GT/s Rx latency to 334 ns

Example 2: To setup PTM Responder Tx Latency to 24 ns and Rx Latency to 115 ns for 8 GT/s

1. Setup speed mode (PCI Express or CCIX ESM)

Write 0 to PTM_RES_ESM_SEL, to set PCI Express speed mode

2. Setup the index

Write 0000_0002_H to PTM_RES_LATENCY_REG_SEL to select 8 GT/s Latency viewport Tx and Rx registers

3. Write the Tx/Rx Latency value

Write 0000_0018_H to PTM_RES_TX_LATENCY to set 8 GT/s Tx latency to 24 ns. Write 0000_0073_H to PTM_RES_RX_LATENCY_OFF register to set 8 GT/s Rx latency to 115 ns

19.3.15.2 Requester features

- Integrated hardware for scheduling of requests to start PTM dialogs

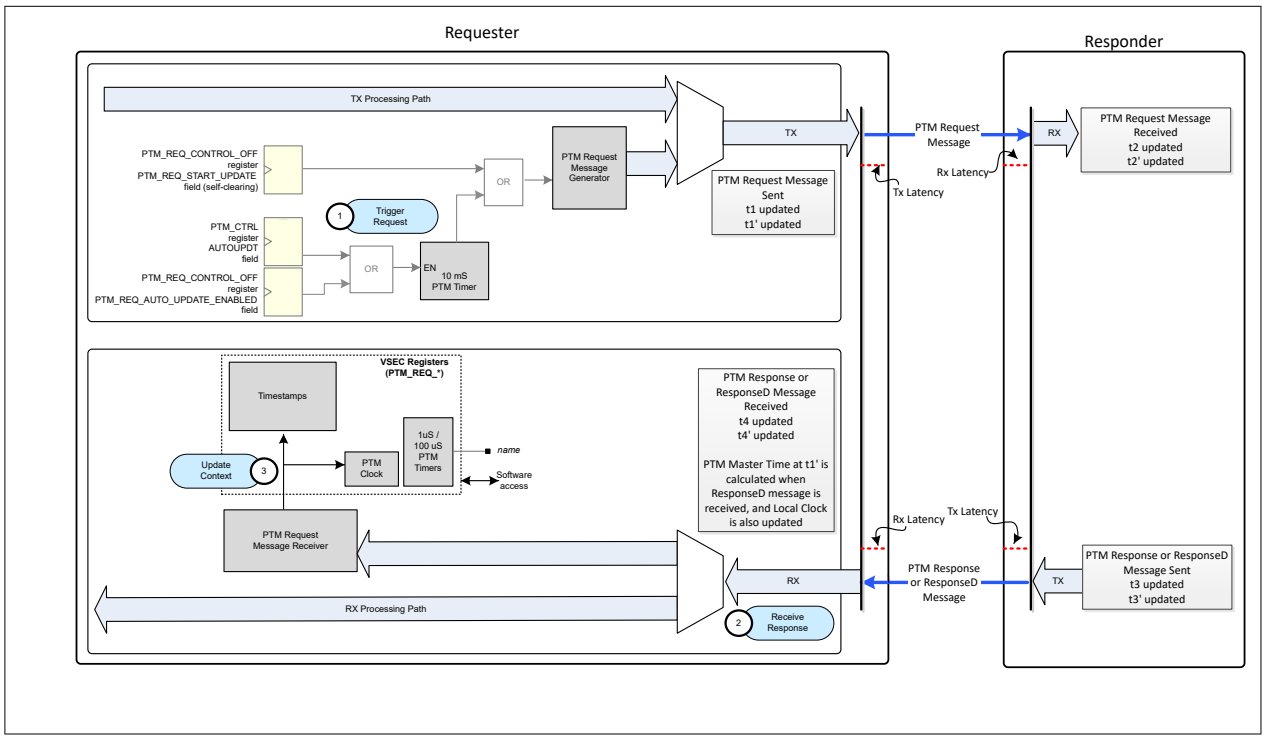


Figure 320 PTM requester (USP)

- Requester indicates PTM response timeout (through PTM_CTRL.TO interrupt) if no PTM response is received within 100 µs of sending a PTM request

  ***Note**:* *When PTM context is invalidated due to duplicates or replays, speed change, or L0 exit, the Requester waits for a PTM response or for 100 µs timeout since the previous request has been sent before allowing a new update cycle to start. If a timeout occurs while waiting, this interrupt is asserted.*

- Requester automatically updates PTM context (starting dialogs) when enabled using any or all of:
  - Automatic trigger every 10 ms (can be enabled through PTM_CTRL.AUTOUPDT)
  - Manual trigger through a register write: PTM_CTRL.TRGG is asserted or PTM_REQ_MANUAL_UPDATE_ALLOWED field in the PTM_REQ_STATUS_OFF register is set
- Requester indicates that a PTM update is in progress (through PTM_CTRL.UPDATING)
- Requester indicates reception of duplicate PTM TLPs (through PTM_CTRL.DUP interrupt) while PTM updates are in progress, or between updates when PTM context is valid
- Local PTM Clock
  - Runs on core_clk, counts time in non-integer nanoseconds, its minimum granularity is defined by the core_clk frequency
  - Auto-updated on PTM context refresh
  - Can be updated by software
  - Can be read by software
- Context automatically invalidated when:
  - core_clk stops or runs at the wrong frequency (for example, in L1 substates or when the link speed is changing), or
  - PTM is disabled by clearing the PTM_ENABLE bit in the PTM_CONTROL_OFF register, or

    ***Note**:* *If PTM is disabled by clearing the PTM_ENABLE bit, it is recommended to wait for some time to allow the PTM Responder in the link partner to complete sending a response before re-enabling PTM. The wait time depends on the maximum response time of the PTM Responder.*

  - PTM response timeouts (the requester restarts the PTM dialog when the auto update or manual update start conditions are met), or
  - A duplicate PTM TLP is received or a replay TLP is sent (if waiting for a response, the requester waits for 100 µs since the last non-duplicate request has been sent before allowing a new PTM dialog to be started)
- Context information stored in PTM Requester VSEC (PTM_REQ_*_OFF):
  - Provides access to timestamps of most recent context refresh
  - Provides notification when context is valid
  - Programmable Tx and Rx latency values for each link speed, automatically selected based on current link speed
- Requester prevents low power entry due to ASPM during PTM updates

19.3.15.3 Responder features

- Integrated hardware for scheduling of PTM responses to dialog requests when context is valid
- Timestamp generation when PTM Messages are detected

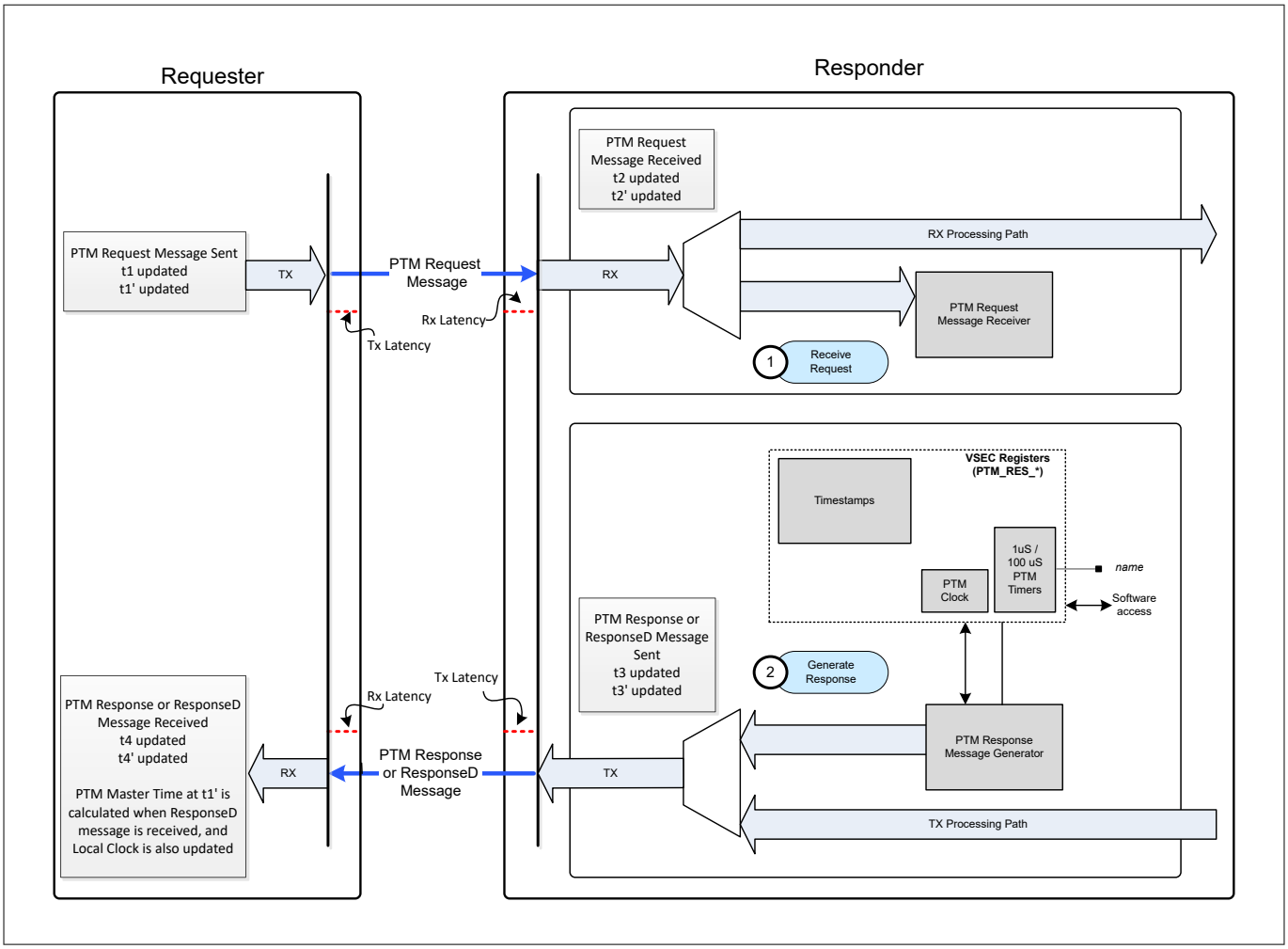


Figure 321 PTM responder (DSP)

- Local PTM Clock
  - Can also be updated by software
  - Can be read by software
- Context automatically invalidated when:
  - core_clk stops or runs at the wrong frequency (for example, in L1 substates or when the link speed is changing), or
  - When PTM is disabled by clearing the PTM_ENABLE bit in the PTM_CONTROL_OFF register

  When the controller asserts PTM_CTRL.RDYVLDT, the users application can re-validate the context by either:
  - Updating the local PTM clock using the PTM_MSKCLK.MSKCLK and PTM_CTRL.MSTSTRB bits, which automatically validates the context, or
  - Updating the local PTM clock by writing to PTM_RES_LOCAL_LSB_OFF and PTM_RES_LOCAL_MSB_OFF registers and then setting the PTM_RES_CCONTEXT_VALID field of PTM_RES_CONTROL_OFF to 1, which automatically validates the context

- Root Support
  - Ideally, a PTM root behaves like a responder with permanently valid context. However,
    - If the PTM root enters low power modes or changes link frequency, PTM context is invalidated (indicated by PTM_CTRL.CNTXTVLD)
    - Following PTM context invalidation, the application must re-validate PTM context, by using either the PTM master time interface signals or the PTM responder VSEC registers, as described in the previous list item
  - PTM messages must not contain TLP prefixes
- Context information stored in PTM Responder VSEC (PTM_RES_*_OFF):
  - Provides access to timestamps of most recent context refresh
  - Provides notification when context is valid
  - Programmable Tx and Rx latency values for each link speed, automatically selected based on current link speed
- When duplicate PTM TLP is received or a replay TLP is sent, the responder invalidates historical timestamps (t3 - t2)

19.3.15.4 Local PTM clock scaling

PTM clock scaling feature allows the local PTM clock to increment a programmable non-integer nanosecond amount in each controller clock period. This allows the PTM local time to remain coherent if there is a frequency deviation between PTM Requester and Responder.

PTM clock scaling also allows the controller to use PTM when the clock period is a non-integer nanosecond multiple.

PTM requester

To enable this feature set the PTM_REQ_SCALED_CLOCK_T_EN field of PTM_REQ_SCALED_CLOCK_T_OFF register to ‘1’ (this bit can be set only when PTM_CTRL.CNTXTVLD is set). You can set a 24-bit value consisting of an 8-bit integral part (nanoseconds) and 16-bit fractional part ($1/2^{16}$ nanoseconds) using PTM_REQ_SCALED_CLOCK_T_INT and PTM_REQ_SCALED_CLOCK_T_FRAC fields of the PTM_REQ_SCALED_CLOCK_T_OFF register, which represents the scaled local PTM clock period.

The clock scaling procedure is as follows:

- Following the second and subsequent assertions of the PTM_CTRL.UPDATED bit, while PTM_CTRL.CNTXTVLD remains asserted, your application may read the PTM Requester Clock Correction register (PTM_REQ_CLOCK_CORR_LSB_OFF)
- If the correction is below a threshold expected by your the application, your application may calculate a scaled local PTM clock period value, based on the current default clock period value, the programmed auto-update interval, and the contents of the PTM Clock Correction register (PTM_REQ_CLOCK_CORR_LSB_OFF)
- Your application may write the scaled PTM clock period to PTM Requester's Scaled PTM Clock Period register (PTM_REQ_SCALED_CLOCK_T_OFF), which represents the scaled local PTM clock period. This should happen as quickly as possible after the PTM_CTRL.UPDATEDpulse to avoid error accumulation

If the PTM context is invalidated, any pending scaled PTM clock period update should be aborted the clock scaling steps listed in this section to be restarted.

PTM responder

For the PTM Responder, your application fully controls PTM local clock updates. If the external master clock runs from the same source clock as the PCIe controller, clock scaling is not needed. However, for applications where the external master and PCIe controller run on unrelated clock sources, it may be necessary to scale the local PTM clock period.

To enable this feature set the PTM_RES_SCALED_CLOCK_T_EN field of PTM_RES_SCALED_CLOCK_T_OFF register to '1' (this bit can only be set when PTM_CTRL.CNTXTVLD is set). You can set a 24-bit value consisting of an 8-bit integral part (nanoseconds) and 16-bit fractional part ($1/2^{16}$ nanoseconds) using PTM_RES_SCALED_CLOCK_T_INT and PTM_RES_SCALED_CLOCK_T_FRAC fields of the PTM_RES_SCALED_CLOCK_T_OFF register, which represents the scaled local PTM clock period.

The clock scaling procedure is as follows:

- Your application initializes the PTM Responder with a known time from an external master
- The PTM Responder counts local PTM time using the Nominal PTM Clock Period, which your application can read from the PTM_RES_NOM_CLOCK_T_OFF register
- Your application waits for a known time and reads the local PTM clock value again
- Your application calculates the scaled PTM clock period using the deviation between the external master clock and the local PTM clock, and writes the scaled PTM clock period to PTM Responder's Scaled PTM Clock Period register (PTM_RES_SCALED_CLOCK_T_OFF), which represents the scaled local PTM clock period

If the PTM context becomes invalid your application must reiterate the steps listed in this section.

Use of scaled clock period and clock rate changes

The PTM Requester and PTM Responder scaled clock period enable bits are cleared when the core_clk rate changes, and can only be set when the clock rate change is complete. The core_clk rate is controlled by LTSSM.

19.3.16 OBFF

The optimized buffer flush/fill (OBFF) is a feature to improve overall power management. As a platform operates, each of its device does not know the power state of each of the resources in the system. Without coordination, each of the devices will go in and out of their low-power states as necessary to execute the tasks they are assigned to do.

As shown in the upper part of the following figure, asynchronous event including bus access and interrupts presents the RC going to a low power state, because the intermittent traffic keeps the system busy and unable to optimize the power management across the system.

With the help of OBFF, a host can provide information to all devices by broadcasting a message about the system power state. The devices can utilize this information to group a load of requests, wait until the system wakes up, and burst out all of the requests at the same time. By doing this, the idle windows and periods of system inactivity are prolonged, which saves overall system power, as depicted in the lower part of the following figure. More details about OBFF are described in PCIe specification.

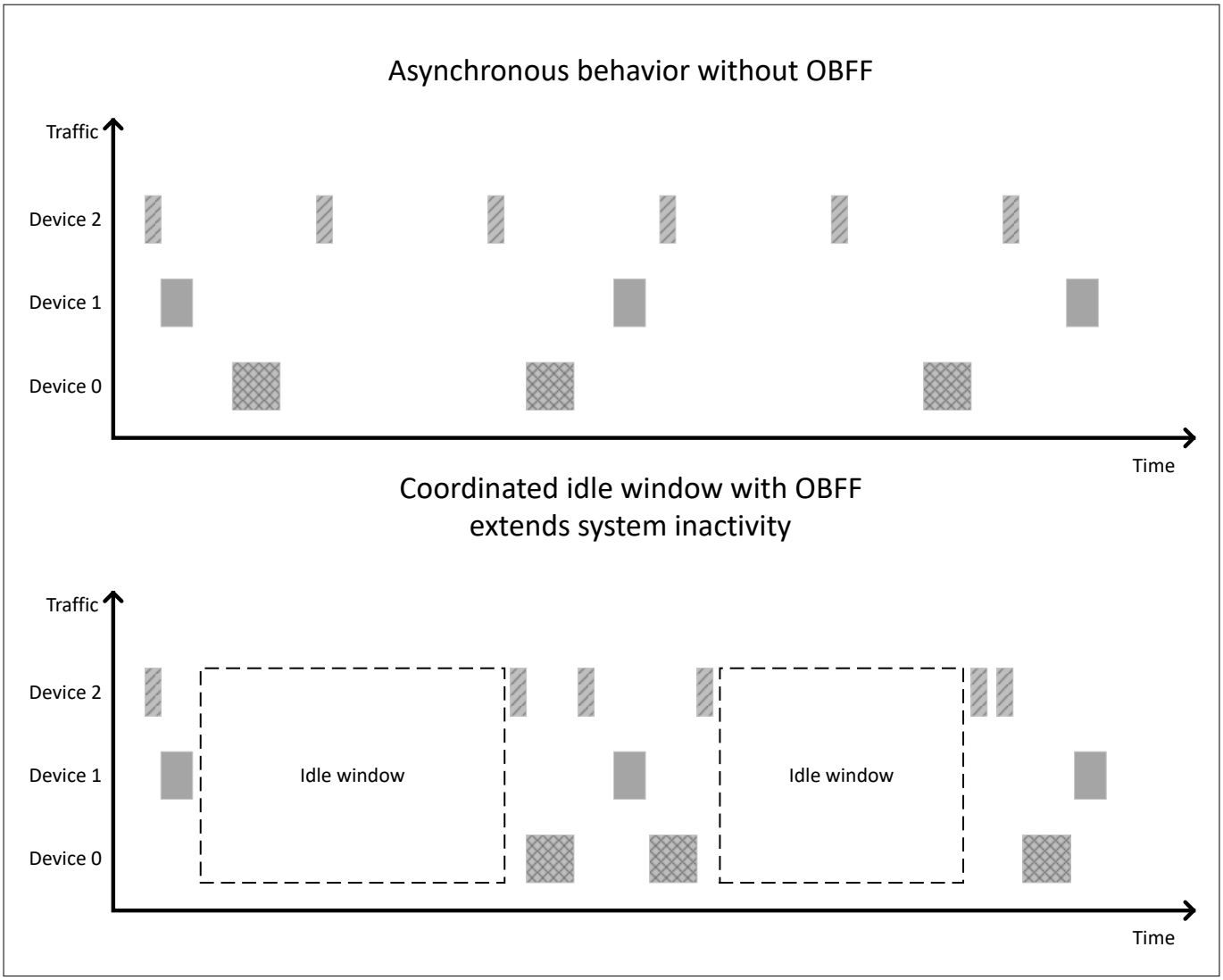


Figure 322 Optimized power consumption with OBFF

OBFF is indicated using either of the WAKE# signal or a message. The message is to be used exclusively on interconnects where the WAKE# signal is not available.

The OBFF messages shall be generated by RC and received by EP. The OBFF message generation is controlled by the OBFF register (OBFFMSG). The current OBFF hints broadcast by RC are indicated in the same register.

Another option of OBFF is using the WAKE# signal, as shown in the following figure. The original purpose of WAKE# was to wake up the system by an EP in lower power state. OBFF feature currently uses it in a compatible way such that this signal has now two functionalities. The usage of WAKE# signal is similar to the usage of messages and is controlled by the OBFFWAKE register.

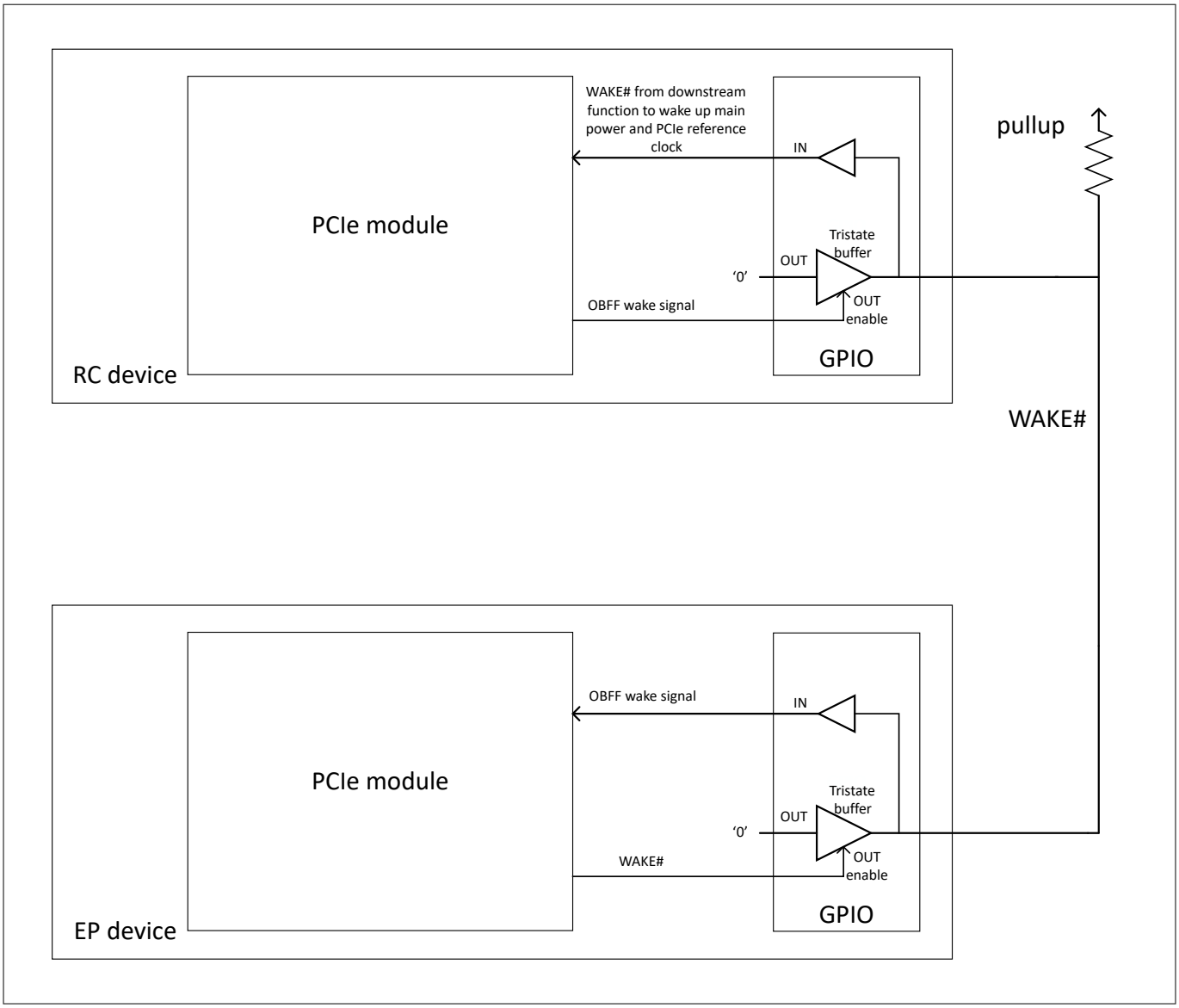


Figure 323 OBFF signaling using WAKE

In order to enable WAKE# input or output set the Controller control.WAKEN bit to 1. The generation of OBFF is triggered by asserting the corresponding OBFF registers. The reception of OBFF is indicated in the OBFF status registers and interrupt requests are also possible to notify software about the current system state. There are two OBFF registers, namely OBFFMSG and OBFFWAKE. Their functionalities are similar but one for message based OBFF and one for WAKE# pin based OBFF.

OBFF generation

You can generate OBFF messages using the following method:

- Direct software trigger: the controller generates OBFF messages in response to requests made by your application assertion of OBFFMSG.IDLEREQ, OBFFMSG.OBFFREQ and OBFFMSG.ACTREQ

You must not send an OBFF message when the OBFF Enable field of the Device Control 2 Register (DEVICE_CONTROL2_DEVICE_STATUS2_REG) is 0.

OBFF reception

When you have not enabled OBFF in the Device Control 2 register, or when the Traffic Class is non-zero, then the controller generates an UR completion when it receives an OBFF message. When the upstream port receives an OBFF message, it asserts one of the OBFFMSG.OBFF, OBFFMSG.ACT, or OBFFMSG.IDLE bit fields. Your application logic must use these outputs to control the traffic it generates. For more information, see *PCI Express Base Specification, Revision 5.0, Version 1.0*.

19.3.17 LTR

Similar to OBFF, latency tolerance reporting is another power-saving feature. The idea is to have each device in the system report its latency requirements to the host. LTR enables a device to communicate the necessary information back to the RC using an LTR message to report its required service latency.

The LTR values are used by the platform to implement an overall power management strategy that will reduce the power consumption while giving optimal performance. More details of LTR are described in the PCIe specification.

LTR generation

When you are using the direct SW triggering you should drive the contents of the LTR message header on the LTR_GNRTN.LTRLAT. It has the same format as LTR messages described in the PCIe specification. The LTR message generation is triggered by asserting LTR_CTRL.MSGREQ.

The actual values in the LTR message transmitted by this upstream port are captured in the port logic LTR Latency Register (PL_LTR_LATENCY_OFF).

The controller automatically generates a new LTR message with the Requirement bits set to 0 for both Snoop and No-Snoop latencies when:

- A function is directed to a non-D0 state through a write to the Power State field of the Power Management Control and Status register, and the LTR mechanism is active since the last DL_Down to DL_Up transition

  or
- The LTR mechanism is disabled by clearing the LTR Mechanism Enable field of the Device Control 2 register. This field is cleared through a CFG write.
- By setting DISABLE_AUTO_LTR_CLR_MSG field of the MISC_CONTROL_1_OFF register to 1, the automatic generation of the LTR clear message situations described in this section can be disabled

You must not send an LTR message when the LTR Mechanism Enable bit of the Device Control 2 Register is 0.

LTR reception

When the controller receives an LTR message, it asserts the LTR_CTRL.RLTR bit. You can access the LTR message information in the general message reception registers (PAYLOADz). The actual values in the LTR message received by this downstream port are captured in the port logic LTR Latency Register (PL_LTR_LATENCY_OFF). .

.

19.3.18 Readiness notification

The controller supports the optional Readiness Notification features. These features provide multiple mechanisms for a device or function to notify the host software that it is configuration-ready. A configuration-ready function always responds to a valid configuration request with a completion indicating successful completion status, and not CRS. For more information, see Readiness Notification in the PCI Express Base Specification, Revision 5.0, Version 1.0.

19.3.19 Limitations

- Locked transactions are not supported
- Atomic operations are not supported

Before the CLC.DISR is enabled, the SPB accesses to the PCIe module are responded with bus errors.

19.3.20 Advanced information

Advanced information that covers various topics is introduced in this part. It is not necessary for basic operations.

19.3.20.1 Separate refclk independent SSC (SRIS)

You can enable support for the Separate Refclk with Independent Spread Spectrum Clocking (SRIS) feature as defined by the PCI Express Base Specification, Revision 5.0, Version 1.0. When LNK_CORECTRL.SRISMD = 1 , then L0s is not supported and must not be advertised in the ASPM Support Capability Registers.

Table 859 SKP ordered set intervals in SRIS and non-SRIS modes

LNK_CORECTRL.SRISMD	SKP ordered set intervals (8b/10b encoding)	SKP ordered set intervals (128b/130b encoding)
0	Normal	Normal
1	Short (less than 154 Symbol times)	short (less than 38 Blocks)

19.3.21 UPCS

The PCIe 3.0 PCS block (also named universal PCS (UPCS)) provides many of the functions outlined in the PIPE 4 specification and comprises the PCS as referenced in the PIPE 4 specification. It can be used in conjunction with the HSPHY to create a PIPE4-compliant PHY solution.

The PCIe 3.0 PCS provides the following features:

- Standard PIPE 4 interface to upper layers of protocol stack (MAC layer and above)
- PCIe: 8b/10b (Gen 1/2) and 128b/130b (Gen3) encoding and decoding
- Translation of PIPE 4 power states (P0, P1, and so on) into the appropriate PHY controls, including any PHY-specific intermediate state sequencing
- Clock-rate compensation through SKP ordered sets and an elasticity buffer
- Error reporting

***Note**: Common refclk mode operation requires PHY controls to be overridden to align with PHY configuration given in the HSPHY device specific chapter.*

Related information

TC4Dx specific UPCS override settings for common refclk PHY settings on page 3617

19.4 Registers

19.4.1 Registers of PCIE_USP

19.4.1.1 Register overview - access mode glossary

Table 860 Register overview - access mode glossary

Keyword	Description
E	Access protection using PROT register PROTE.
SE	Access protection using PROT register PROTSE or S2A_PROTSE.
APU-P	Protection group consisting of registers ACCEN_WRA, ACCEN_WRB, ACCEN_RDA, ACCEN_RDB, ACCEN_VM, ACCEN_PRS.
P	Access protection using *APU-P* registers.
APU-Py (y=0-7)	Protection group consisting of registers S2A_ACCENy_WRA, S2A_ACCENy_WRB, S2A_ACCENy_RDA, S2A_ACCENy_RDB, S2A_ACCENy_VM, S2A_ACCENy_PRS, S2A_ACCENy_RGNLA, S2A_ACCENy_RGNUA.
Py	Access protection using *APU-Py* registers.
APU-P_DMAy (y=0-3)	Protection group consisting of registers DMA_ACCENy_WRA, DMA_ACCENy_WRB, DMA_ACCENy_RDA, DMA_ACCENy_RDB, DMA_ACCENy_VM, DMA_ACCENy_PRS.
P_DMAy	Access protection using *APU-P_DMAy* registers.
BE	Always returns a Bus Error.
U	No access restrictions.
PROT	Access restrictions as defined in the PROT register access rules.

19.4.1.2 Register overview - PCIE_USP domain FPI (ascending offset address)

Table 861 Register overview - PCIE_USP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CLC	Clock control register	0000000H	P	P, SV, E	Application Reset	2822
OCS	OCDS control and status register	0000004H	P	SV, P	Debug Reset	2823
ID	Module identification register	0000008H	P	BE	PowerOn Reset	2823
RST_CTRLA	Reset control register A	000000CH	P	P, SV, E	Application Reset	2824
RST_CTRLB	Reset control register B	0000010H	P	P, SV, E	Application Reset	2825
RST_STAT	Reset status register	0000014H	P	BE	Application Reset	2825

(table continues...)

Table 861 (continued) Register overview - PCIE_USP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PROTE	PROT register endinit	0000020$_H$	U	SV, PROT	Application Reset	2826
PROTSE	PROT register safe endinit	0000024$_H$	U	SV, PROT	Application Reset	2828
ACCEN_WRA	Write access enable register A	0000040$_H$	U	SE, SV	Application Reset	2829
ACCEN_WRB	Write access enable register B	0000044$_H$	U	SE, SV	Application Reset	2830
ACCEN_RDA	Read access enable register A	0000048$_H$	U	SE, SV	Application Reset	2830
ACCEN_RDB	Read access enable register B	000004C$_H$	U	SE, SV	Application Reset	2831
ACCEN_VM	VM access enable register	0000050$_H$	U	SE, SV	Application Reset	2831
ACCEN_PRS	PRS access enable register	0000054$_H$	U	SE, SV	Application Reset	2832
DMA_ACCENy_WRA	CHy Write access enable register A	0000060$_H$+y*20$_H$	U	SE, SV	Application Reset	2833
DMA_ACCENy_WRB	CHy Write access enable register B	0000064$_H$+y*20$_H$	U	SE, SV	Application Reset	2833
DMA_ACCENy_RDA	CHy Read access enable register A	0000068$_H$+y*20$_H$	U	SE, SV	Application Reset	2834
DMA_ACCENy_RDB	CHy Read access enable register B	000006C$_H$+y*20$_H$	U	SE, SV	Application Reset	2834
DMA_ACCENy_VM	CHy VM access enable register	0000070$_H$+y*20$_H$	U	SE, SV	Application Reset	2835
DMA_ACCENy_PRS	CHy PRS access enable register	0000074$_H$+y*20$_H$	U	SE, SV	Application Reset	2835
UPCS_OVRD1	UPCS layer override register 1	0000300$_H$	P	P	Kernel Reset	2836
UPCS_OVRD2	UPCS layer override register 2	0000304$_H$	P	P	Kernel Reset	2836
UPCS_OVRD3	UPCS layer override register 3	0000308$_H$	P	P	Kernel Reset	2837
UPCS_OVRD4	UPCS layer override register 4	000030C$_H$	P	P	Kernel Reset	2837
UPCS_OVRD5	UPCS layer override register 5	0000310$_H$	P	P	Kernel Reset	2838
UPCS_OVRD6	UPCS layer override register 6	0000314$_H$	P	P	Kernel Reset	2838
UPCS_OVRD7	UPCS layer override register 7	0000318$_H$	P	P	Kernel Reset	2839

(table continues...)

Table 861 (continued) Register overview - PCIE_USP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
UPCS_OVRD8	UPCS layer override register 8	000031C H	P	P	Kernel Reset	2839
UPCS_OVRD9	UPCS layer override register 9	0000320 H	P	P	Kernel Reset	2840
UPCS_OVRD10	UPCS layer override register 10	0000324 H	P	P	Kernel Reset	2840
UPCS_OVRD11	UPCS layer override register 11	0000328 H	P	P	Kernel Reset	2841
UPCS_OVRD12	UPCS layer override register 12	000032C H	P	P	Kernel Reset	2841
UPCS_OVRD13	UPCS layer override register 13	0000330 H	P	P	Kernel Reset	2841
UPCS_OVRD14	UPCS layer override register 14	0000334 H	P	P	Kernel Reset	2842
UPCS_OVRD15	UPCS layer override register 15	0000338 H	P	P	Kernel Reset	2842
UPCS_OVRD16	UPCS layer override register 16	000033C H	P	P	Kernel Reset	2843
UPCS_OVRD17	UPCS layer override register 17	0000340 H	P	P	Kernel Reset	2843
INT_MSICTRL	Request MSI interrupt, EP only.	0000348 H	P	P	Kernel Reset	2844
INT_MSISTAT	Pending status of MSI interrupt request, EP only.	000034C H	P	P	Kernel Reset	2844
INT_INTXCTRL	Trigger INTX interrupt, EP only.	0000350 H	P	P	Kernel Reset	2845
INT_TRIGSTAT	Trigger status	0000358 H	P	P	Kernel Reset	2845
INT_TRIGCLR	Interrupt clear, EP only.	000035C H	P	P	Kernel Reset	2846
INT_TRIGEN	Interrupt enable EP only.	0000360 H	P	P	Kernel Reset	2846
INT_REQTRIG	Receive incoming interrupt request, EP only.	0000364 H	P	P	Kernel Reset	2847
CFG_INFO	Configuration information	0000368 H	P	P	Kernel Reset	2847
CFG_BUSNUM	Bus/device number	000036C H	P	P	Kernel Reset	2849
LNK_DVCTYP	Select device/port type	0000370 H	P	P, E	PowerOn Reset	2849
LNK_STAT	Indicate link status	0000374 H	P	P	Kernel Reset	2850

(table continues...)

Table 861 (continued) Register overview - PCIE_USP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
LNK_CORECTRL	Controller control	0000378$_H$	P	P	Kernel Reset	2852
LNK_INTEN	Enable the interrupt requests to SRC_PCIELNK	000037C$_H$	P	P	Kernel Reset	2854
TXRX_RXCTRL	Rx control	0000380$_H$	P	P	Kernel Reset	2855
TXRX_TXCTRL0	Tx control 0	0000384$_H$	P	P	Kernel Reset	2856
PAYLOADz	Message payload z information	000038C$_H$+z*4	P	P	Kernel Reset	2857
MSG_INFO	Message information	0000394$_H$	P	P	Kernel Reset	2857
PM_CTRL	Power management control	0000398$_H$	P	P	Kernel Reset	2859
PM_STAT0	Power management status 0	000039C$_H$	P	P	Kernel Reset	2861
PM_STAT1	Power management status 1	00003A4$_H$	P	P	Kernel Reset	2863
PM_INTCLR	Power management interrupt clear	00003A8$_H$	P	P	Kernel Reset	2863
PM_INTEN	Power management interrupt enable	00003AC$_H$	P	P	Kernel Reset	2864
ERR_STAT	Error status	00003B0$_H$	P	P	Kernel Reset	2865
ERR_INTCLR	Error clear	00003B4$_H$	P	P	Kernel Reset	2865
ERR_INTEN	Error enable	00003B8$_H$	P	P	Kernel Reset	2866
DBG_STAT	Debug status	00003BC$_H$	P	P	Kernel Reset	2867
DBG_DATA	Debug information	00003C0$_H$	P	P	Kernel Reset	2868
LTR_GNRTN	LTR message	00003C4$_H$	P	P	Kernel Reset	2868
LTR_CTRL	LTR control	00003CC$_H$	P	P	Kernel Reset	2868
PTM_CTRL	PTM control	00003D0$_H$	P	P	Kernel Reset	2869
LCLCLKz	Local Clock z value	00003D4$_H$+z*4	P	P	Kernel Reset	2870
CLKCORRz	Clock z correction	00003DC$_H$+z*4	P	P	Kernel Reset	2870

(table continues...)

Table 861 (continued) Register overview - PCIE_USP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
MSKCLKz	PTM z External Master Time	00003E4$_H$+z*4	P	P	Kernel Reset	2871
OBFFMSG	OBFF message control	00003F0$_H$	P	P	Kernel Reset	2872
OBFFWAKE	OBFF WAKE control	00003F4$_H$	P	P	Kernel Reset	2874
CFGBD	PCIe configuration control	00003F8$_H$	P	P	Kernel Reset	2876
DATAx	ELBI x mailbox register	00003FC$_H$+x*4	P	P	Kernel Reset	2876
LLCTRL	DMA link list control register	000041C$_H$	P	P	Kernel Reset	2877
OTSS	PCIe OCDS Trigger Set Select	0000420$_H$	P	SV, P	Debug Reset	2877
ALARM_STAT	Interrupt/alarm status	0000424$_H$	P	P	Kernel Reset	2878
ALARM_INTCLR	Interrupt/alarm clear	0000428$_H$	P	P	Kernel Reset	2880
ALARM_INTEN	Interrupt/alarm enable	000042C$_H$	P	P	Kernel Reset	2881
IATU_REGION_CTRL_1_OFF_OUTBOUND_i	iATU Region i Control 1 Register Outbound	0003000$_H$+i*200$_H$	P	P	Kernel Reset	2882
IATU_REGION_CTRL_2_OFF_OUTBOUND_i	iATU Region i Control 2 Register Outbound	0003004$_H$+i*200$_H$	P	P	Kernel Reset	2883
IATU_LWR_BASE_ADDR_OFF_OUTBOUND_i	iATU i Lower Base Address Register Outbound	0003008$_H$+i*200$_H$	P	P	Kernel Reset	2885
IATU_UPPER_BASE_ADDR_OFF_OUTBOUND_i	iATU i Upper Base Address Register Outbound	000300C$_H$+i*200$_H$	P	P	Kernel Reset	2886
IATU_LIMIT_ADDR_OFF_OUTBOUND_i	iATU i Limit Address Register Outbound	0003010$_H$+i*200$_H$	P	P	Kernel Reset	2886
IATU_LWR_TARGET_ADDR_OFF_OUTBOUND_i	iATU i Lower Target Address Register Outbound	0003014$_H$+i*200$_H$	P	P	Kernel Reset	2887
IATU_UPPER_TARGET_ADDR_OFF_OUTBOUND_i	iATU i Upper Target Address Register Outbound	0003018$_H$+i*200$_H$	P	P	Kernel Reset	2887

(table continues...)

Table 861 (continued) Register overview - PCIE_USP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
IATU_REGION_CTRL_1_OFF_INBOUND_i	iATU Region i Control 1 Register Inbound	0003100$_H$+i*200$_H$	P	P	Kernel Reset	2888
IATU_REGION_CTRL_2_OFF_INBOUND_i	iATU Region i Control 2 Register Inbound	0003104$_H$+i*200$_H$	P	P	Kernel Reset	2889
IATU_LWR_BASE_ADDR_OFF_INBOUND_i	iATU i Lower Base Address Register Inbound	0003108$_H$+i*200$_H$	P	P	Kernel Reset	2893
IATU_UPPER_BASE_ADDR_OFF_INBOUND_i	iATU i Upper Base Address Register Inbound	000310C$_H$+i*200$_H$	P	P	Kernel Reset	2893
IATU_LIMIT_ADDR_OFF_INBOUND_i	iATU i Limit Address Register Inbound	0003110$_H$+i*200$_H$	P	P	Kernel Reset	2894
IATU_LWR_TARGET_ADDR_OFF_INBOUND_i	iATU i Lower Target Address Register Inbound	0003114$_H$+i*200$_H$	P	P	Kernel Reset	2895
DEVICE_ID_VENDOR_ID_REG	Device ID and Vendor ID Register	0004000$_H$	P	P	Kernel Reset	2895
STATUS_COMMAND_REG	Status and Command Register	0004004$_H$	P	P	Kernel Reset	2896
CLASS_CODE_REVISION_ID	Class Code and Revision ID Register	0004008$_H$	P	P	Kernel Reset	2900
BIST_HEADER_TYPE_LATENCY_CACHE_LINE_SIZE_REG	BIST, Header Type, Latency Timer, and Cache Line Size Register	000400C$_H$	P	P	Kernel Reset	2901
BARi_REG	BAR0 Register i	0004010$_H$+i*4	P	P	Kernel Reset	2902
CARDBUS_CIS_PTR_REG	CardBus CIS Pointer Register	0004028$_H$	P	P	Kernel Reset	2903
SUBSYSTEM_ID_SUBSYSTEM_VENDOR_ID_REG	Subsystem ID and Subsystem Vendor ID Register	000402C$_H$	P	P	Kernel Reset	2904
PCI_CAP_PTR_REG	Capabilities Pointer Register	0004034$_H$	P	P	Kernel Reset	2905
MAX_LATENCY_MIN_GRANT_INTERRUPT_PIN_INTERRUPT_LINE_REG	Max_Lat, Min_Gnt, Interrupt Pin, and Interrupt Line Register	000403C$_H$	P	P	Kernel Reset	2905
CAP_ID_NXT_PTR_REG	Power Management Capabilities Register	0004040$_H$	P	P	Kernel Reset	2906

(table continues...)

Table 861 (continued) Register overview - PCIE_USP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CON_STATUS_REG	Power Management Control and Status Register	0004044H	P	P	Kernel Reset	2909
PCI_MSI_CAP_ID_NEXT_CTRL_REG	MSI Capability Header and Message Control Register	0004050H	P	P	Kernel Reset	2912
MSI_CAP_OFF_04H_REG	Message Address Register for MSI (Offset 04h)	0004054H	P	P	Kernel Reset	2914
MSI_CAP_OFF_08H_REG	Message Address Register for MSI (Offset 08h)	0004058H	P	P	Kernel Reset	2915
MSI_CAP_OFF_0CH_REG	Message Address Register for MSI (Offset 0Ch)	000405CH	P	P	Kernel Reset	2916
MSI_CAP_OFF_10H_REG	Message Address Register for MSI (Offset 10h)	0004060H	P	P	Kernel Reset	2917
MSI_CAP_OFF_14H_REG	Message Address Register for MSI (Offset 14h)	0004064H	P	P	Kernel Reset	2917
PCIE_CAP_ID_PCIE_NEXT_CAP_PTR_PCIE_CAP_REG	PCI Express Capabilities, ID, Next Pointer Register	0004070H	P	P	Kernel Reset	2918
DEVICE_CAPABILITIES_REG	Device Capabilities Register	0004074H	P	P	Kernel Reset	2920
DEVICE_CONTROL_DEVICE_STATUS	Device Control and Device Status Register	0004078H	P	P	Kernel Reset	2924
LINK_CAPABILITIES_REG	Link Capabilities Register	000407CH	P	P	Kernel Reset	2929
LINK_CONTROL_LINK_STATUS_REG	Link Control and Link Status Register	0004080H	P	P	Kernel Reset	2934
DEVICE_CAPABILITIES2_REG	Device Capabilities 2 Register	0004094H	P	P	Kernel Reset	2941
DEVICE_CONTROL2_DEVICE_STATUS2_REG	Device Control 2 and Status 2 Register	0004098H	P	P	Kernel Reset	2945
LINK_CAPABILITIES2_REG	Link Capabilities 2 Register	000409CH	P	P	Kernel Reset	2947
LINK_CONTROL2_LINK_STATUS2_REG	Link Control 2 and Status 2 Register	00040A0H	P	P	Kernel Reset	2949
PCI_MSIX_CAP_ID_NEXT_CTRL_REG	MSI-X Capability ID, Next Pointer, Control Register	00040B0H	P	P	Kernel Reset	2955

(table continues...)

Table 861 (continued) Register overview - PCIE_USP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
MSIX_TABLE_OFFSET_REG	MSI-X Table Offset and BIR Register	00040B4$_H$	P	P	Kernel Reset	2957
MSIX_PBA_OFFSET_REG	MSI-X PBA Offset and BIR Register	00040B8$_H$	P	P	Kernel Reset	2957
AER_EXT_CAP_HDR_OFF	Advanced Error Reporting Extended Capability Header	0004100$_H$	P	P	Kernel Reset	2958
UNCORR_ERR_STATUS_OFF	Uncorrectable Error Status Register	0004104$_H$	P	P	Kernel Reset	2959
UNCORR_ERR_MASK_OFF	Uncorrectable Error Mask Register	0004108$_H$	P	P	Kernel Reset	2961
UNCORR_ERR_SEV_OFF	Uncorrectable Error Severity Register	000410C$_H$	P	P	Kernel Reset	2963
CORR_ERR_STATUS_OFF	Correctable Error Status Register	0004110$_H$	P	P	Kernel Reset	2965
CORR_ERR_MASK_OFF	Correctable Error Mask Register	0004114$_H$	P	P	Kernel Reset	2966
ADV_ERR_CAP_CTRL_OFF	Advanced Error Capabilities and Control Register	0004118$_H$	P	P	Kernel Reset	2967
HDR_LOG_i_OFF	Header i Log Register 0	000411C$_H$+i*4	P	P	Kernel Reset	2969
TLP_PREFIX_LOG_i_OFF	TLP Prefix i Log Register 1	0004138$_H$+i*4	P	P	Kernel Reset	2969
SN_BASE	Device Serial Number Extended Capability Header	0004148$_H$	P	P	Kernel Reset	2970
SER_NUM_REG_DW_1	Serial Number 1 Register	000414C$_H$	P	P	Kernel Reset	2971
SER_NUM_REG_DW_2	Serial Number 2 Register	0004150$_H$	P	P	Kernel Reset	2971
SPCIE_CAP_HEADER_REG	SPCIE Capability Header	0004158$_H$	P	P	Kernel Reset	2972
LINK_CONTROL3_REG	Link Control 3 Register	000415C$_H$	P	P	Kernel Reset	2973
LANE_ERR_STATUS_REG	Lane Error Status Register	0004160$_H$	P	P	Kernel Reset	2974
SPCIE_CAP_OFF_0CH_REG	Lane Equalization Control Register for lanes 1 and 0	0004164$_H$	P	P	Kernel Reset	2975
LTR_CAP_HDR_REG	LTR Extended Capability Header	0004168$_H$	P	P	Kernel Reset	2976
LTR_LATENCY_REG	LTR Max Snoop and No-Snoop Latency Register	000416C$_H$	P	P	Kernel Reset	2977
L1SUB_CAP_HEADER_REG	L1 Substates Extended Capability Header	0004170$_H$	P	P	Kernel Reset	2978

(table continues...)

Table 861 (continued) Register overview - PCIE_USP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
L1SUB_CAPABILITY_REG	L1 Substates Capability Register	0004174H	P	P	Kernel Reset	2979
L1SUB_CONTROL1_REG	L1 Substates Control 1 Register	0004178H	P	P	Kernel Reset	2981
L1SUB_CONTROL2_REG	L1 Substates Control 2 Register	000417CH	P	P	Kernel Reset	2982
RAS_DES_CAP_HEADER_REG	Vendor-Specific Extended Capability Header	0004180H	P	P	Kernel Reset	2984
VENDOR_SPECIFIC_HEADER_REG	Vendor-Specific Header	0004184H	P	P	Kernel Reset	2985
EVENT_COUNTER_CONTROL_REG	Event Counter Control	0004188H	P	P	Kernel Reset	2985
EVENT_COUNTER_DATA_REG	Event Counter Data	000418CH	P	P	Kernel Reset	2987
TIME_BASED_ANALYSIS_CONTROL_REG	Time-based Analysis Control	0004190H	P	P	Kernel Reset	2988
TIME_BASED_ANALYSIS_DATA_REG	Time-based Analysis Data	0004194H	P	P	Kernel Reset	2989
TIME_BASED_ANALYSIS_DATA_63_32_REG	Upper 32 bits of Time-based Analysis Data	0004198H	P	P	Kernel Reset	2990
EINJ_ENABLE_REG	Error Injection Enable	00041B0H	P	P	Kernel Reset	2990
EINJ0_CRC_REG	Error Injection Control 0 (CRC Error)	00041B4H	P	P	Kernel Reset	2992
EINJ1_SEQNUM_REG	Error Injection Control 1 (Sequence Number Error). 	00041B8H	P	P	Kernel Reset	2993
EINJ2_DLLP_REG	Error Injection Control 2 (DLLP Error)	00041BCH	P	P	Kernel Reset	2994
EINJ3_SYMBOL_REG	Error Injection Control 3 (Symbol Error)	00041C0H	P	P	Kernel Reset	2995
EINJ4_FC_REG	Error Injection Control 4 (FC Credit Error)	00041C4H	P	P	Kernel Reset	2996
EINJ5_SP_TLP_REG	Error Injection Control 5 (Specific TLP Error)	00041C8H	P	P	Kernel Reset	2997
EINJ6_COMPARE_POINT_Hi_REG	Error Injection Control 6 (Compare Point Header DWORD #0) i	00041CCH+i*4	P	P	Kernel Reset	2998

(table continues...)

Table 861 (continued) Register overview - PCIE_USP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
EINJ6_COMPARE_VALUE_Hi_REG	Error Injection Control 6 (Compare Value Header DWORD #0) i	00041DC$_H$+i*4	P	P	Kernel Reset	2999
EINJ6_CHANGE_POINT_Hi_REG	Error Injection Control 6 (Change Point Header DWORD #0) i	00041EC$_H$+i*4	P	P	Kernel Reset	2999
EINJ6_CHANGE_VALUE_Hi_REG	Error Injection Control 6 (Change Value Header DWORD #0) i	00041FC$_H$+i*4	P	P	Kernel Reset	3000
EINJ6_TLP_REG	Error Injection Control 6 (Packet Error)	000420C$_H$	P	P	Kernel Reset	3001
SD_CONTROL1_REG	Silicon Debug Control 1	0004220$_H$	P	P	Kernel Reset	3002
SD_CONTROL2_REG	Silicon Debug Control 2	0004224$_H$	P	P	Kernel Reset	3003
SD_STATUS_L1LANE_REG	Silicon Debug Status(Layer1 Per-lane)	0004230$_H$	P	P	Kernel Reset	3004
SD_STATUS_L1LTSSM_REG	Silicon Debug Status(Layer1 LTSSM)	0004234$_H$	P	P	Kernel Reset	3005
SD_STATUS_PM_REG	Silicon Debug Status(PM)	0004238$_H$	P	P	Kernel Reset	3008
SD_STATUS_L2_REG	Silicon Debug Status(Layer2)	000423C$_H$	P	P	Kernel Reset	3011
SD_STATUS_L3FC_REG	Silicon Debug Status(Layer3 FC)	0004240$_H$	P	P	Kernel Reset	3011
SD_STATUS_L3_REG	Silicon Debug Status(Layer3)	0004244$_H$	P	P	Kernel Reset	3013
SD_EQ_CONTROL1_REG	Silicon Debug EQ Control 1	0004250$_H$	P	P	Kernel Reset	3014
SD_EQ_CONTROL2_REG	Silicon Debug EQ Control 2	0004254$_H$	P	P	Kernel Reset	3016
SD_EQ_CONTROL3_REG	Silicon Debug EQ Control 3	0004258$_H$	P	P	Kernel Reset	3017
SD_EQ_STATUS1_REG	Silicon Debug EQ Status 1	0004260$_H$	P	P	Kernel Reset	3018
SD_EQ_STATUS2_REG	Silicon Debug EQ Status 2	0004264$_H$	P	P	Kernel Reset	3019
SD_EQ_STATUS3_REG	Silicon Debug EQ Status 3	0004268$_H$	P	P	Kernel Reset	3020
RASDP_EXT_CAP_HDR_OFF	PCIe Extended capability ID, Capability version and Next capability offset	0004280$_H$	P	P	Kernel Reset	3021

(table continues...)

Table 861 (continued) Register overview - PCIE_USP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
RASDP_VENDOR_SPECIFIC_HDR_OFF	Vendor Specific Header	0004284 H	P	P	Kernel Reset	3022
RASDP_ERROR_PROT_CTRL_OFF	ECC Error Correction and Control Register	0004288 H	P	P	Kernel Reset	3023
RASDP_ERROR_MODE_EN_OFF	RASDP error mode enable.	00042A8 H	P	P	Kernel Reset	3035
RASDP_ERROR_MODE_CLEAR_OFF	RASDP Error Mode Clear Register	00042AC H	P	P	Kernel Reset	3036
RASDP_RAM_ADDR_CORR_ERROR_OFF	RAM Address where a corrected error (1-bit ECC) is detected	00042B0 H	P	P	Kernel Reset	3037
RASDP_RAM_ADDR_UNCORR_ERROR_OFF	RAM Address where an uncorrected error (2-bit ECC) is detected	00042B4 H	P	P	Kernel Reset	3037
PTM_EXT_CAP_HDR_OFF	Precision Time Measurement Capability Header	00042B8 H	P	P	Kernel Reset	3038
PTM_CAP_OFF	PTM Capability Register	00042BC H	P	P	Kernel Reset	3039
PTM_CONTROL_OFF	PTM Control Register	00042C0 H	P	P	Kernel Reset	3040
PTM_REQ_CAP_HDR_OFF	Precision Time Measurement Requester Capability Header (VSEC)	00042C4 H	P	P	Kernel Reset	3041
PTM_REQ_HDR_OFF	Precision Time Measurement Requester Vendor Specific Header	00042C8 H	P	P	Kernel Reset	3042
PTM_REQ_CONTROL_OFF	PTM Requester Vendor Specific Control Register	00042CC H	P	P	Kernel Reset	3043
PTM_REQ_STATUS_OFF	PTM Requester Vendor Specific Status Register	00042D0 H	P	P	Kernel Reset	3044
PTM_REQ_LOCAL_LSB_OFF	PTM Requester Local Clock LSB	00042D4 H	P	P	Kernel Reset	3045
PTM_REQ_LOCAL_MSB_OFF	PTM Requester Local Clock MSB	00042D8 H	P	P	Kernel Reset	3046
PTM_REQ_T1_LSB_OFF	PTM Requester T1 Timestamp LSB	00042DC H	P	P	Kernel Reset	3046
PTM_REQ_T1_MSB_OFF	PTM Requester T1 Timestamp MSB	00042E0 H	P	P	Kernel Reset	3047
PTM_REQ_T1P_LSB_OFF	PTM Requester T1 Previous Timestamp LSB	00042E4 H	P	P	Kernel Reset	3047

(table continues...)

Table 861 (continued) Register overview - PCIE_USP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PTM_REQ_T1P_MSB_OFF	PTM Requester T1 Previous Timestamp MSB	00042E8H	P	P	Kernel Reset	3047
PTM_REQ_T4_LSB_OFF	PTM Requester T4 Timestamp LSB	00042ECH	P	P	Kernel Reset	3048
PTM_REQ_T4_MSB_OFF	PTM Requester T4 Timestamp MSB	00042F0H	P	P	Kernel Reset	3048
PTM_REQ_T4P_LSB_OFF	PTM Requester T4 Previous Timestamp LSB	00042F4H	P	P	Kernel Reset	3049
PTM_REQ_T4P_MSB_OFF	PTM Requester T4 Previous Timestamp MSB	00042F8H	P	P	Kernel Reset	3049
PTM_REQ_MASTER_LSB_OFF	PTM Requester Master Time LSB	00042FCH	P	P	Kernel Reset	3050
PTM_REQ_MASTER_MSB_OFF	PTM Requester Master Time MSB	0004300H	P	P	Kernel Reset	3050
PTM_REQ_PROP_DELAY_OFF	PTM Requester Propagation Delay	0004304H	P	P	Kernel Reset	3051
PTM_REQ_MASTERT1_LSB_OFF	PTM Requester Master Time at T1 LSB	0004308H	P	P	Kernel Reset	3051
PTM_REQ_MASTERT1_MSB_OFF	PTM Requester Master Time at T1 MSB	000430CH	P	P	Kernel Reset	3052
PTM_REQ_TX_LATENCY_OFF	PTM Requester TX Latency	0004310H	P	P	Kernel Reset	3052
PTM_REQ_RX_LATENCY_OFF	PTM Requester RX Latency	0004314H	P	P	Kernel Reset	3053
PTM_REQ_CLOCK_CORR_LSB_OFF	PTM Requester Clock Correction LSB	0004318H	P	P	Kernel Reset	3054
PTM_REQ_CLOCK_CORR_MSB_OFF	PTM Requester Clock Correction MSB	000431CH	P	P	Kernel Reset	3054
PTM_REQ_NOM_CLOCK_T_OFF	PTM Requester Nominal Clock Period	0004320H	P	P	Kernel Reset	3055
PTM_REQ_SCALED_CLOCK_T_OFF	PTM Requester Scaled Clock Period	0004324H	P	P	Kernel Reset	3055
PTM_REQ_LATENCY_REG_SEL_OFF	PTM Requester Latency Viewport Register Select	0004328H	P	P	Kernel Reset	3056
VSECDMA_EXT_CAP_HDR_OFF	PCIe Extended Capability ID, Capability Version, and Next Capability Offset Register	000432CH	P	P	Kernel Reset	3057

(table continues...)

Table 861 (continued) Register overview - PCIE_USP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
VSECDMA_VENDOR_SPECIFIC_HDR_OFF	Vendor Specific Header Register	0004330H	P	P	Kernel Reset	3058
VSECDMA_DEVICE_INFORMATION_OFF	DMA and related AXI Bridge Implementation Information	0004334H	P	P	Kernel Reset	3059
VSECDMA_NUM_CHAN_OFF	Number of Implemented Channels Register	0004338H	P	P	Kernel Reset	3060
VSECDMA_UNROLL_ADDR_OFFSET_LOW_OFF	DMA Register Map Start Address Offset Low Register	000433CH	P	P	Kernel Reset	3061
VSECDMA_UNROLL_ADDR_OFFSET_HIGH_OFF	DMA Register Map Start Address Offset High Register	0004340H	P	P	Kernel Reset	3061
ACK_LATENCY_TIMER_OFF	Ack Latency Timer and Replay Timer Register	0004700H	P	P	Kernel Reset	3062
VENDOR_SPEC_DLLP_OFF	Vendor Specific DLLP Register	0004704H	P	P	Kernel Reset	3063
PORT_FORCE_OFF	Port Force Link Register	0004708H	P	P	Kernel Reset	3063
ACK_F_ASPM_CTRL_OFF	Ack Frequency and L0-L1 ASPM Control Register	000470CH	P	P	Kernel Reset	3065
PORT_LINK_CTRL_OFF	Port Link Control Register	0004710H	P	P	Kernel Reset	3067
LANE_SKEW_OFF	Lane Skew Register	0004714H	P	P	Kernel Reset	3069
TIMER_CTRL_MAX_FUNC_NUM_OFF	Timer Control and Max Function Number Register	0004718H	P	P	Kernel Reset	3071
SYMBOL_TIMER_FILTER_1_OFF	Symbol Timer Register and Filter Mask 1 Register	000471CH	P	P	Kernel Reset	3072
FILTER_MASK_2_OFF	Filter Mask 2 Register	0004720H	P	P	Kernel Reset	3075
PL_DEBUG0_OFF	Debug Register 0	0004728H	P	P	Kernel Reset	3077
PL_DEBUG1_OFF	Debug Register 1	000472CH	P	P	Kernel Reset	3078
TX_P_FC_CREDIT_STATUS_OFF	Transmit Posted FC Credit Status	0004730H	P	P	Kernel Reset	3079
TX_NP_FC_CREDIT_STATUS_OFF	Transmit Non-Posted FC Credit Status	0004734H	P	P	Kernel Reset	3080
TX_CPL_FC_CREDIT_STATUS_OFF	Transmit Completion FC Credit Status	0004738H	P	P	Kernel Reset	3081

(table continues...)

Table 861 (continued) Register overview - PCIE_USP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
QUEUE_STATUS_OFF	Queue Status	000473C H	P	P	Kernel Reset	3082
VC0_P_RX_Q_CTRL_OFF	Segmented-Buffer VC0 Posted Receive Queue Control	0004748 H	P	P	Kernel Reset	3084
VC0_NP_RX_Q_CTRL_OFF	Segmented-Buffer VC0 Non-Posted Receive Queue Control	000474C H	P	P	Kernel Reset	3086
VC0_CPL_RX_Q_CTRL_OFF	Segmented-Buffer VC0 Completion Receive Queue Control	0004750 H	P	P	Kernel Reset	3087
GEN2_CTRL_OFF	Link Width and Speed Change Control Register	000480C H	P	P	Kernel Reset	3088
CLOCK_GATING_CTRL_OFF	Clock Gating Control Register	000488C H	P	P	Kernel Reset	3092
GEN3_RELATED_OFF	Gen3 Control Register	0004890 H	P	P	Kernel Reset	3093
GEN3_EQ_CONTROL_OFF	Gen3 EQ Control Register	00048A8 H	P	P	Kernel Reset	3096
GEN3_EQ_FB_MODE_DIR_CHANGE_OFF	Gen3 EQ Direction Change Feedback Mode Control Register	00048AC H	P	P	Kernel Reset	3099
ORDER_RULE_CTRL_OFF	Order Rule Control Register	00048B4 H	P	P	Kernel Reset	3100
PIPE_LOOPBACK_CONTROL_OFF	PIPE Loopback Control Register	00048B8 H	P	P	Kernel Reset	3101
MISC_CONTROL_1_OFF	DBI Read-Only Write Enable Register	00048BC H	P	P	Kernel Reset	3102
LINK_FLUSH_CONTROL_OFF	Link Reset Request Flush Control Register	00048CC H	P	P	Kernel Reset	3108
AMBA_ERROR_RESPONSE_DEFAULT_OFF	AXI Bridge Slave Error Response Register	00048D0 H	P	P	Kernel Reset	3109
AMBA_LINK_TIMEOUT_OFF	Link Down AXI Bridge Slave Timeout Register	00048D4 H	P	P	Kernel Reset	3110
AMBA_ORDERING_CTRL_OFF	AXI Bridge Ordering Control	00048D8 H	P	P	Kernel Reset	3111
PCIE_VERSION_NUMBER_OFF	PCIe Controller IIP Release Version Number	00048F8 H	P	P	Kernel Reset	3115
PCIE_VERSION_TYPE_OFF	PCIe Controller IIP Release Version Type	00048FC H	P	P	Kernel Reset	3116

(table continues...)

Table 861 (continued) Register overview - PCIE_USP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
INTERFACE_TIMER_CONTROL_OFF	Interface Timer Control Register	0004930H	P	P	Kernel Reset	3116
INTERFACE_TIMER_TARGET_OFF	Interface Timer Target Register	0004934H	P	P	Kernel Reset	3117
INTERFACE_TIMER_STATUS_OFF	Interface Timer Status Register	0004938H	P	P	Kernel Reset	3118
MSIX_ADDRESS_MATCH_LOW_OFF	MSI-X Address Match Low Register	0004940H	P	P	Kernel Reset	3119
MSIX_ADDRESS_MATCH_HIGH_OFF	MSIX Address Match High Register	0004944H	P	P	Kernel Reset	3120
MSIX_DOORBELL_OFF	MSI-X Doorbell Register	0004948H	P	P	Kernel Reset	3120
SAFETY_MASK_OFF	Functional Safety Interrupt Events Mask Register	0004960H	P	P	Kernel Reset	3123
SAFETY_STATUS_OFF	Status for functional safety interrupt events	0004964H	P	P	Kernel Reset	3124
PL_APP_BUS_DEV_NUM_STATUS_OFF	Application driven bus and device number register	0004B10H	P	P	Kernel Reset	3125
PL_CHK_REG_CONTROL_STATUS_OFF	CDM Register Checking Control and Status Register	0004B20H	P	P	Kernel Reset	3126
PL_CHK_REG_START_END_OFF	CDM Register Checking First and Last address to check	0004B24H	P	P	Kernel Reset	3128
PL_CHK_REG_ERR_ADDR_OFF	CDM Register Checking Error Address	0004B28H	P	P	Kernel Reset	3128
PL_CHK_REG_ERR_PF_VF_OFF	CDM Register Checking error PF Numbers	0004B2CH	P	P	Kernel Reset	3129
PL_LTR_LATENCY_OFF	LTR Latency Register	0004B30H	P	P	Kernel Reset	3129
AUX_CLK_FREQ_OFF	Auxiliary Clock Frequency Control Register	0004B40H	P	P	Kernel Reset	3130
POWERDOWN_CTRL_STATUS_OFF	Powerdown Control and Status Register	0004B48H	P	P	Kernel Reset	3132
MSIXTABLEx_LADDR	MSIX table x DWORD0	0005A00H+x*10H	P	P	Kernel Reset	3134
MSIXTABLEx_UADDR	MSIX table x DWORD1	0005A04H+x*10H	P	P	Kernel Reset	3134

(table continues...)

Table 861 (continued) Register overview - PCIE_USP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
MSIXTABLEx_DATA	MSIX table x DWORD2	0005A08$_H$+x*10$_H$	P	P	Kernel Reset	3135
MSIXTABLEx_VECTRL	MSIX table x DWORD3	0005A0C$_H$+x*10$_H$	P	P	Kernel Reset	3135
MSIXPBAx_DWORD	MSIX xtable DWORD	0005E10$_H$+x*4	P	P	Kernel Reset	3136
BARi_MASK_REG	BAR i Mask Register	0006010$_H$+i*4	P	P	Kernel Reset	3136
SHADOW_PCI_MSIX_CAP_ID_NEXT_CTRL_REG	MSI-X Capability ID, Next Pointer, Control Register	00060B0$_H$	P	P	Kernel Reset	3137
DMA_CTRL_DATA_ARB_PRIOR_OFF	DMA Arbitration Scheme for TRGT1 Interface	0007000$_H$	P	P	Kernel Reset	3139
DMA_CTRL_OFF	DMA Number of Channels Register	0007008$_H$	P	P	Kernel Reset	3140
DMA_WRITE_ENGINE_EN_OFF	DMA Write Engine Enable Register	000700C$_H$	P	P	Kernel Reset	3141
DMA_WRITE_DOORBELL_OFF	DMA Write Doorbell Register	0007010$_H$	P	P	Kernel Reset	3143
DMA_WRITE_CHANNEL_ARB_WEIGHT_LOW_OFF	DMA Write Engine Channel Arbitration Weight Low Register	0007018$_H$	P	P	Kernel Reset	3144
DMA_READ_ENGINE_EN_OFF	DMA Read Engine Enable Register	000702C$_H$	P	P	Kernel Reset	3146
DMA_READ_DOORBELL_OFF	DMA Read Doorbell Register	0007030$_H$	P	P	Kernel Reset	3147
DMA_READ_CHANNEL_ARB_WEIGHT_LOW_OFF	DMA Read Engine Channel Arbitration Weight Low Register	0007038$_H$	P	P	Kernel Reset	3148
DMA_WRITE_INT_STATUS_OFF	DMA Write Interrupt Status Register	000704C$_H$	P	P	Kernel Reset	3150
DMA_WRITE_INT_MASK_OFF	DMA Write Interrupt Mask Register	0007054$_H$	P	P	Kernel Reset	3151
DMA_WRITE_INT_CLEAR_OFF	DMA Write Interrupt Clear Register	0007058$_H$	P	P	Kernel Reset	3152
DMA_WRITE_ERR_STATUS_OFF	DMA Write Error Status Register	000705C$_H$	P	P	Kernel Reset	3153
DMA_WRITE_DONE_IMWR_LOW_OFF	DMA Write Done IMWr Address Low Register	0007060$_H$	P	P	Kernel Reset	3153

(table continues...)

Table 861 (continued) Register overview - PCIE_USP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
DMA_WRITE_DONE_IMWR_HIGH_OFF	DMA Write Done IMWr Interrupt Address High Register	0007064H	P	P	Kernel Reset	3154
DMA_WRITE_ABORT_IMWR_LOW_OFF	DMA Write Abort IMWr Address Low Register	0007068H	P	P	Kernel Reset	3154
DMA_WRITE_ABORT_IMWR_HIGH_OFF	DMA Write Abort IMWr Address High Register	000706CH	P	P	Kernel Reset	3155
DMA_WRITE_CH01_IMWR_DATA_OFF	DMA Write Channel 1 and 0 IMWr Data Register	0007070H	P	P	Kernel Reset	3155
DMA_WRITE_CH23_IMWR_DATA_OFF	DMA Write Channel 3 and 2 IMWr Data Register	0007074H	P	P	Kernel Reset	3156
DMA_WRITE_LINKED_LIST_ERR_EN_OFF	DMA Write Linked List Error Enable Register	0007090H	P	P	Kernel Reset	3157
DMA_READ_INT_STATUS_OFF	DMA Read Interrupt Status Register	00070A0H	P	P	Kernel Reset	3158
DMA_READ_INT_MASK_OFF	DMA Read Interrupt Mask Register	00070A8H	P	P	Kernel Reset	3159
DMA_READ_INT_CLEAR_OFF	DMA Read Interrupt Clear Register	00070ACH	P	P	Kernel Reset	3160
DMA_READ_ERR_STATUS_LOW_OFF	DMA Read Error Status Low Register	00070B4H	P	P	Kernel Reset	3161
DMA_READ_ERR_STATUS_HIGH_OFF	DMA Read Error Status High Register	00070B8H	P	P	Kernel Reset	3162
DMA_READ_LINKED_LIST_ERR_EN_OFF	DMA Read Linked List Error Enable Register	00070C4H	P	P	Kernel Reset	3163
DMA_READ_DONE_IMWR_LOW_OFF	DMA Read Done IMWr Address Low Register	00070CCH	P	P	Kernel Reset	3164
DMA_READ_DONE_IMWR_HIGH_OFF	DMA Read Done IMWr Address High Register	00070D0H	P	P	Kernel Reset	3164
DMA_READ_ABORT_IMWR_LOW_OFF	DMA Read Abort IMWr Address Low Register	00070D4H	P	P	Kernel Reset	3165

(table continues...)

Table 861 (continued) Register overview - PCIE_USP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
DMA_READ_ABORT_IMWR_HIGH_OFF	DMA Read Abort IMWr Address High Register	00070D8$_H$	P	P	Kernel Reset	3165
DMA_READ_CH01_IMWR_DATA_OFF	DMA Read Channel 1 and 0 IMWr Data Register	00070DC$_H$	P	P	Kernel Reset	3166
DMA_READ_CH23_IMWR_DATA_OFF	DMA Read Channel 3 and 2 IMWr Data Register	00070E0$_H$	P	P	Kernel Reset	3166
DMA_WRITE_ENGINE_HSHAKE_CNT_LOW_OFF	DMA Write Engine Handshake Counter Channel 0/1/2/3 Register	0007108$_H$	P	P	Kernel Reset	3168
DMA_READ_ENGINE_HSHAKE_CNT_LOW_OFF	DMA Read Engine Handshake Counter Channel 0/1/2/3 Register	0007118$_H$	P	P	Kernel Reset	3169
DMA_WRITE_CHi_PWR_EN_OFF	DMA Write Channel i Power Enable Register	0007128$_H$+i*4	P	P	Kernel Reset	3171
DMA_READ_CHi_PWR_EN_OFF	DMA Read Channel i Power Enable Register	0007168$_H$+i*4	P	P	Kernel Reset	3171
DMA_CH_CONTROL1_OFF_WRCH_i	DMA Write Channel i Control 1 Register	0007200$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3172
DMA_TRANSFER_SIZE_OFF_WRCH_i	DMA CH i Write Transfer Size Register	0007208$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3175
DMA_SAR_LOW_OFF_WRCH_i	DMA CH i Write SAR Low Register	000720C$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3175
DMA_SAR_HIGH_OFF_WRCH_i	DMA CH i Write SAR High Register	0007210$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3176
DMA_DAR_LOW_OFF_WRCH_i	DMA CH i Write DAR Low Register	0007214$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3176
DMA_DAR_HIGH_OFF_WRCH_i	DMA CH i Write DAR High Register	0007218$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3177
DMA_LLP_LOW_OFF_WRCH_i	DMA CH i Write Linked List Pointer Low Register	000721C$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3177
DMA_LLP_HIGH_OFF_WRCH_i	DMA CH i Write Linked List Pointer High Register	0007220$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3178
DMACAP_CHi_WRCHN_WR_INT_CLEAR	Interrupt clear for DMA write channel i	0007224$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3178
DMACAP_CHi_WRCHN_WR_INT_STATUS	Interrupt status for DMA write channel i	0007228$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3179

(table continues...)

Table 861 (continued) Register overview - PCIE_USP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
DMACAP_CHi_WRCHN_WR_ERR_STATUS	Error status for DMA write channel i	000722C$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3180
DMACAP_CHi_WRCHN_WR_DOORBELL	Doorbell for DMA write channel i	0007230$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3181
DMA_CH_CONTROL1_OFF_RDCH_i	DMA Read Channel i Control 1 Register	0007300$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3181
DMA_TRANSFER_SIZE_OFF_RDCH_i	DMA CH i Read Transfer Size Register	0007308$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3184
DMA_SAR_LOW_OFF_RDCH_i	DMA CH i Read SAR Low Register	000730C$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3185
DMA_SAR_HIGH_OFF_RDCH_i	DMA CH i Read SAR High Register	0007310$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3186
DMA_DAR_LOW_OFF_RDCH_i	DMA CH i Read DAR Low Register	0007314$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3186
DMA_DAR_HIGH_OFF_RDCH_i	DMA CH i Read DAR High Register	0007318$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3187
DMA_LLP_LOW_OFF_RDCH_i	DMA CH i Read Linked List Pointer Low Register	000731C$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3187
DMA_LLP_HIGH_OFF_RDCH_i	DMA CH i Read Linked List Pointer High Register	0007320$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3188
DMACAP_CHi_RDCHN_RD_INT_CLEAR	Interrupt clear for DMA read channel i	0007324$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3188
DMACAP_CHi_RDCHN_RD_INT_STATUS	Interrupt status for DMA read channel i	0007328$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3189
DMACAP_CHi_RDCHN_RD_ERR_STATUS	Error status for DMA read channel i	000732C$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3189
DMACAP_CHi_RDCHN_RD_ERR_STATUS_HIGH	Higher error status for DMA read channel i	0007330$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3190
DMACAP_CHi_RDCHN_RD_DOORBELL	Doorbell for DMA read channel i	0007334$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3191

19.4.1.3 Register overview - PCIE_USP_SRI domain SRI (ascending offset address)

Table 862 Register overview - PCIE_USP_SRI domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
S2A_ACCENy_WRA	Write access enable register A	3FFFE00$_H$+y*20$_H$	U	SE, SV	Application Reset	3192
S2A_ACCENy_WRB	Write access enable register B	3FFFE04$_H$+y*20$_H$	U	SE, SV	Application Reset	3193
S2A_ACCENy_RDA	Read access enable register A	3FFFE08$_H$+y*20$_H$	U	SE, SV	Application Reset	3193
S2A_ACCENy_RDB	Read access enable register B	3FFFE0C$_H$+y*20$_H$	U	SE, SV	Application Reset	3194
S2A_ACCENy_VM	VM access enable register	3FFFE10$_H$+y*20$_H$	U	SE, SV	Application Reset	3194
S2A_ACCENy_PRS	PRS access enable register	3FFFE14$_H$+y*20$_H$	U	SE, SV	Application Reset	3195
S2A_ACCENy_RGNLA	Region lower address register	3FFFE18$_H$+y*20$_H$	U	SE, SV	Application Reset	3195
S2A_ACCENy_RGNUA	Region upper address register	3FFFE1C$_H$+y*20$_H$	U	SE, SV	Application Reset	3196
S2A_PROTSE	PROT Register Safe Endinit	3FFFF04$_H$	U	SV	Application Reset	3196

19.4.1.4 Clock control register

The Clock Control Register CLC allows the programmer to adapt the functionality and power consumption of the module to the requirements of the application.

Register CLC controls the module clock signal and the reactivity to the sleep signal.

CLC Offset address: 0000000$_H$

Clock control register Application Reset value: 0000 0003$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												EDIS	0	DISS	DISR
r												rw	r	rh	rw

Field	Bits	Type	Description
DISR	0	rw	**Module Disable Request Bit** Used for enable/disable control of the module. 0_B On request: enable the module clock 1_B Off request: stop the module clock
DISS	1	rh	**Module Disable Status Bit** When CLC.DISS=1_B, registers CLC, OCS, ID, PROTE, PROTSE, ACCEN_* and DMA_ACCEN_* are still accessible. Accesses to other PCIe registers are not executed and will result in a bus error 0_B Module clock is enabled 1_B Off: module is not clocked
EDIS	3	rw	**Reserved** Used to control the module's reaction to sleep mode. Note: this bit field has no functionality implemented for PCIe. 0_B Sleep mode request is enabled and functional 1_B Module disregards the sleep mode control signal
0	2, 31:4	r	**Reserved** Read as 0; should be written with 0.

19.4.1.5 OCDS control and status register

OCS Offset address: 0000004_H

OCDS control and status register Debug Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

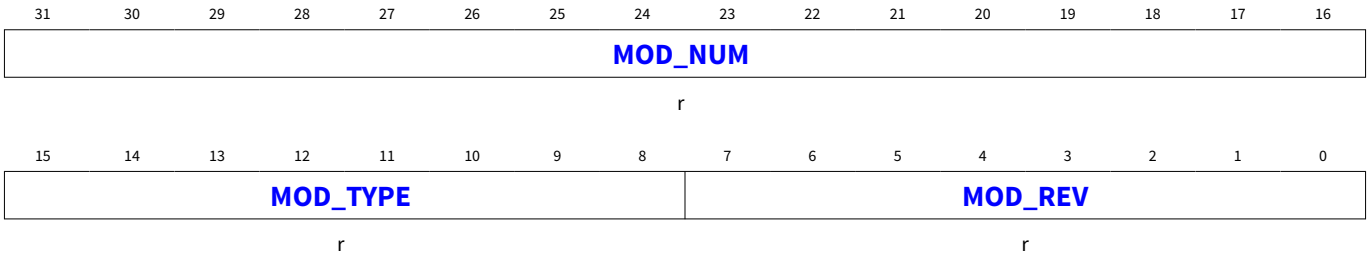
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															
r															

Field	Bits	Type	Description
0	31:0	r	**Reserved** Read as 0; should be written with 0.

19.4.1.6 Module identification register

ID Offset address: 0000008_H

Module identification register PowerOn Reset value: $0106\ C002_H$

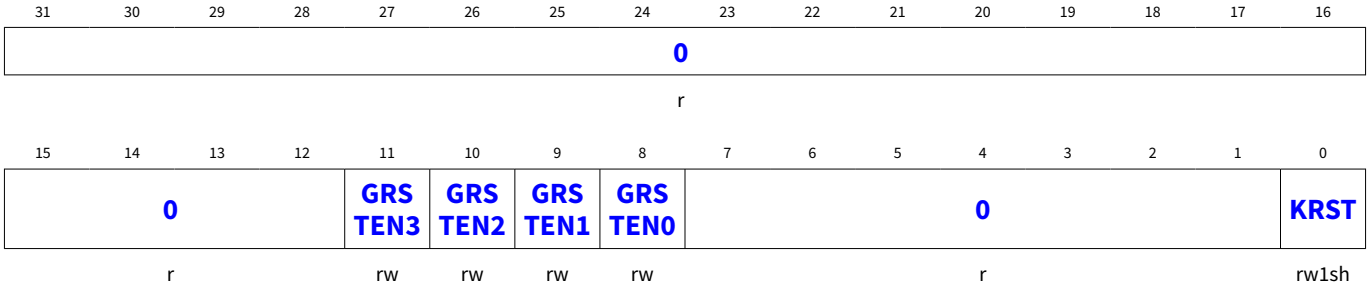


Field	Bits	Type	Description
MOD_REV	7:0	r	**Module Revision** Indicates the revision number of the implementation.
MOD_TYPE	15:8	r	**Module Type** This internal marker is fixed to $C0_H$.
MOD_NUM	31:16	r	**Module Number** Indicates the module identification number. 0x0106 for PCIe.

19.4.1.7 Reset control register A

RST_CTRLA Offset address: $000000C_H$

Reset control register A Application Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
KRST	0	rw1sh	**Kernel Reset** Request a kernel reset. The requested reset is executed if CTRLA.KRST=1 and CTRLB.KRST=1. KRST is cleared after the kernel reset was executed. 0_B No action 1_B A kernel reset was requested
GRSTENx (x=0-3)	x+8	rw	**Enable for Global Module Reset Group x** 0_B Global module reset group x does not have any effect 1_B Global module reset group x results in a kernel reset
0	7:1, 31:12	r	**Reserved** Read as 0; should be written with 0.

19.4.1.8 Reset control register B

RST_CTRLB Offset address: 0000010_H

Reset control register B Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
STATCLR								0							
w								r							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															KRST
r															rw1sh

Field	Bits	Type	Description
KRST	0	rw1sh	**Kernel Reset** Request a kernel reset. The requested reset is executed if CTRLA.KRST=1 and CTRLB.KRST=1. KRST is cleared after the kernel reset was executed. 0_B No action 1_B A kernel reset was requested
STATCLR	31	w	**Kernel Reset Status Clear** Clears all status bits in RST_STAT when 1 is written. Read will return 0. 0_B No action 1_B Write with ´1´ clears bits STAT.GRSTx and bit STAT.KRST .
0	30:1	r	**Reserved** Read as 0; should be written with 0.

19.4.1.9 Reset status register

The reset status register contains the status bits for kernel reset (KRST) and the global module reset groups (GRSTx).

RST_STAT Offset address: 0000014_H

Reset status register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				GRST3	GRST2	GRST1	GRST0	0							KRST
r				rh	rh	rh	rh	r							rh

Field	Bits	Type	Description
KRST	0	rh	**Kernel Reset Status** Indicates an executed kernel reset. RST_STAT.KRST is set after the execution of a kernel reset in the same clock cycle in which the reset bits are cleared. Clear KRST by setting bit STATCLR in register RST_CTRLB.
GRSTx (x=0-3)	x+8	rh	**Status for Global Module Reset Group x** 0_B Reset was not triggered by Global Reset Group x 1_B Reset was triggered by Global Reset Group x
0	7:1, 31:12	r	**Reserved** Read as 0.

19.4.1.10 PROT register endinit

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner and in some PROT states, the secure master to update the PROT state as described in the PROT mechanism.

The PROTE register controls lock / unlock of the local Endinit (E) protected control registers.

PROTE Offset address: 0000020_H

PROT register endinit Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OWEN	ODEF	TAGID						PRSEN	PRS			VMEN	VM		
w	rw	rw						rw	rw			rw	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												SWEN	STATE		
r												w	rwh		

Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)

(table continues...)

(continued)

Field	Bits	Type	Description
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
VM	18:16	rw	**Virtual machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual machine definition enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.
PRSEN	23	rw	**Protection set definition enable for PROT owner** 0_B **Disable**: PRS is not part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG-ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; should be written with 0.

Table 863 Access types of PROTE in order of evaluation

Condition	Type		Description
write 1 to .**SWEN**	rwh	STATE	
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
otherwise	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

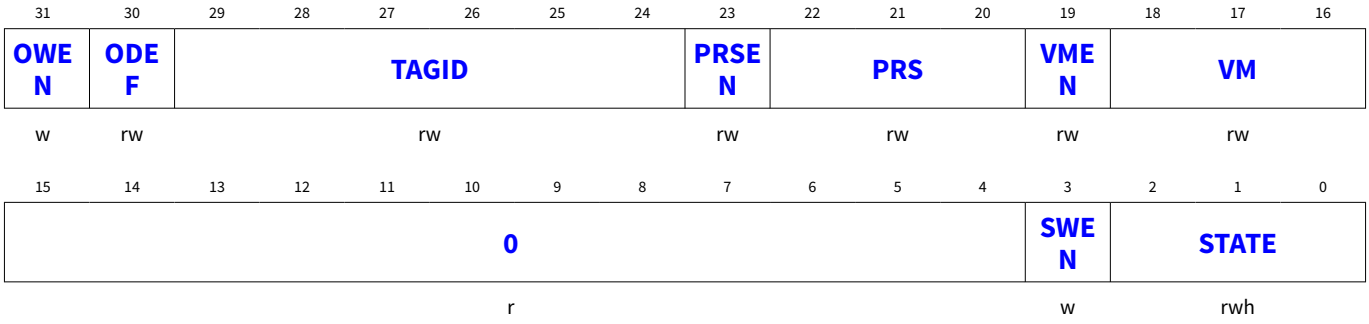
19.4.1.11 PROT register safe endinit

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner and in some PROT states, the secure master to update the PROT state as described in the PROT mechanism.

The PROTSE register controls lock / unlock of the local Safe Endinit (SE) protected control registers.

PROTSE Offset address: 0000024_H

PROT register safe endinit Application Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
VM	18:16	rw	**Virtual machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual machine definition enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.

(table continues...)

(continued)

Field	Bits	Type	Description
PRSEN	23	rw	**Protection set definition enable for PROT owner** 0_B **Disable**: PRS is not part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG-ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; should be written with 0.

Table 864 Access types of PROTSE in order of evaluation

Condition	Type		Description
write 1 to .**SWEN**	rwh	STATE	
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
otherwise	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

19.4.1.12 Write access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

ACCEN_WRA Offset address: 0000040_H

Write access enable register A Application Reset value: $1000\ 0003_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access

19.4.1.13 Write access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

ACCEN_WRB Offset address: 0000044_H

Write access enable register B Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access
0	31:8	r	**Reserved** Read as 0; should be written with 0.

19.4.1.14 Read access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

ACCEN_RDA Offset address: 0000048_H

Read access enable register A Application Reset value: $FFFF\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access

19.4.1.15 Read access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

ACCEN_RDB Offset address: 000004C$_H$

Read access enable register Application Reset value: 0000 00FF$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access
0	31:8	r	**Reserved** Read as 0; should be written with 0.

19.4.1.16 VM access enable register

This register defines which virtual machine encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each VMx.

ACCEN_VM Offset address: 0000050$_H$

VM access enable register Application Reset value: 00FF 00FF$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for VM ID q** This bit enables read access to the access protected region for transactions with the VM ID q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for VM ID q** This bit enables write access to the access protected region for transactions with the VM ID q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

19.4.1.17 PRS access enable register

This register defines which protection register set encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each PRSx.

ACCEN_PRS Offset address: 0000054_H

PRS access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for PRS q** This bit enables read access to the access protected region for transactions with the PRS q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for PRS q** This bit enables write access to the access protected region for transactions with the PRS q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

19.4.1.18 CHy Write access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

DMA_ACCENy_WRA (y=0-3) Offset address: 0000060_H+y*20_H

CHy Write access enable register A Application Reset value: $1000\ 0003_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access

19.4.1.19 CHy Write access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

DMA_ACCENy_WRB (y=0-3) Offset address: 0000064_H+y*20_H

CHy Write access enable register B Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access
0	31:8	r	**Reserved** Read as 0; should be written with 0.

19.4.1.20 CHy Read access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

DMA_ACCENy_RDA (y=0-3) Offset address: 0000068_H+y*20_H

CHy Read access enable register A Application Reset value: FFFF $FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access

19.4.1.21 CHy Read access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

DMA_ACCENy_RDB (y=0-3) Offset address: $000006C_H+y*20_H$

CHy Read access enable register B Application Reset value: 0000 $00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access
0	31:8	r	**Reserved** Read as 0; should be written with 0.

19.4.1.22 CHy VM access enable register

This register defines which virtual machine encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each VMx.

DMA_ACCENy_VM (y=0-3) Offset address: 0000070_H+y*20_H

CHy VM access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for VM ID q** This bit enables read access to the access protected region for transactions with the VM ID q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for VM ID q** This bit enables write access to the access protected region for transactions with the VM ID q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

19.4.1.23 CHy PRS access enable register

This register defines which protection register set encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each PRSx.

DMA_ACCENy_PRS (y=0-3) Offset address: 0000074_H+y*20_H

CHy PRS access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for PRS q** This bit enables read access to the access protected region for transactions with the PRS q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for PRS q** This bit enables write access to the access protected region for transactions with the PRS q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

19.4.1.24 UPCS layer override register 1

UPCS Override Settings

Note: the settings can be overwritten from these top-level when OVRD17.CTRLSEL is asserted.

UPCS_OVRD1 Offset address: 0000300_H

UPCS layer override register 1 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	**OVRD1**														
r	rw														

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OVRD1															
rw															

Field	Bits	Type	Description
OVRD1	30:0	rw	**UPCS override 1**
0	31	r	**Reserved** Read as 0; should be written with 0.

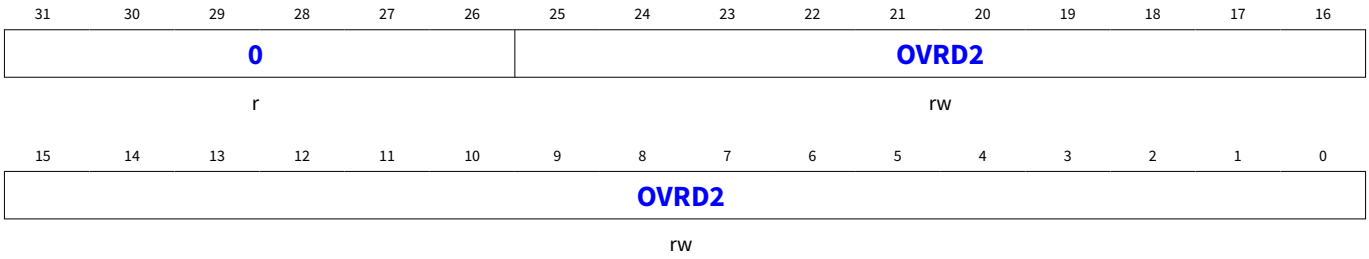
19.4.1.25 UPCS layer override register 2

UPCS Override Settings

Note: the settings can be overwritten from these top-level when OVRD17.CTRLSEL is asserted.

UPCS_OVRD2 Offset address: 0000304_H

UPCS layer override register 2 Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
OVRD2	25:0	rw	**UPCS override 2**
0	31:26	r	**Reserved** Read as 0; should be written with 0.

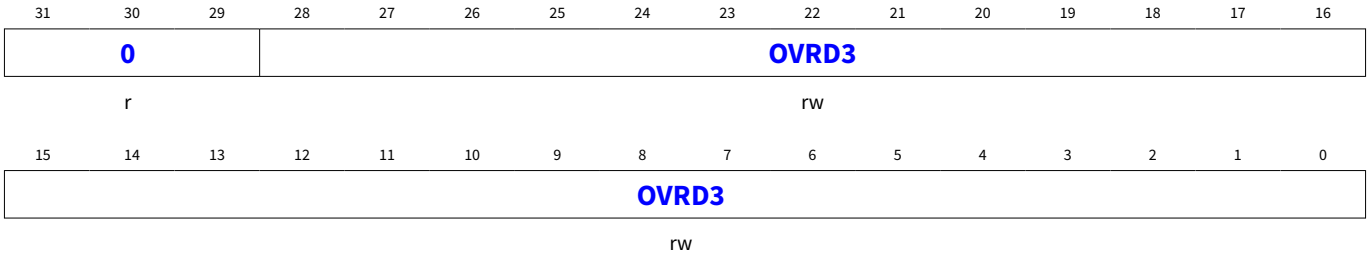
19.4.1.26 UPCS layer override register 3

UPCS Override Settings

Note: the settings can be overwritten from these top-level when OVRD17.CTRLSEL is asserted.

UPCS_OVRD3 Offset address: 0000308_H

UPCS layer override register 3 Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
OVRD3	28:0	rw	**UPCS override 3**
0	31:29	r	**Reserved** Read as 0; should be written with 0.

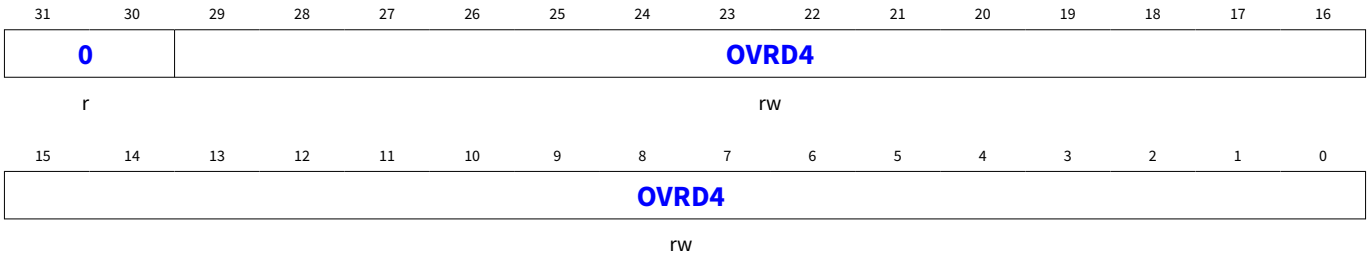
19.4.1.27 UPCS layer override register 4

UPCS Override Settings

Note: the settings can be overwritten from these top-level when OVRD17.CTRLSEL is asserted.

UPCS_OVRD4 Offset address: $000030C_H$

UPCS layer override register 4 Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
OVRD4	29:0	rw	**UPCS override 4**
0	31:30	r	**Reserved** Read as 0; should be written with 0.

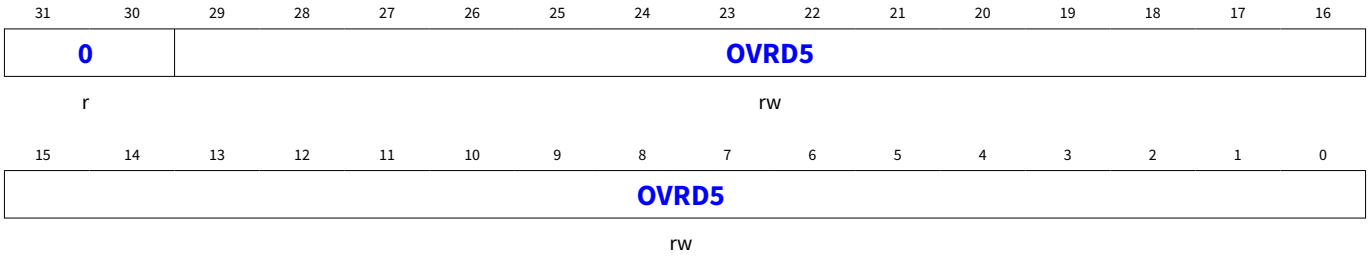
19.4.1.28 UPCS layer override register 5

UPCS Override Settings

Note: the settings can be overwritten from these top-level when OVRD17.CTRLSEL is asserted.

UPCS_OVRD5 Offset address: 0000310_H

UPCS layer override register 5 Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
OVRD5	29:0	rw	**UPCS override 5**
0	31:30	r	**Reserved** Read as 0; should be written with 0.

19.4.1.29 UPCS layer override register 6

UPCS Override Settings

Note: the settings can be overwritten from these top-level when OVRD17.CTRLSEL is asserted.

UPCS_OVRD6 Offset address: 0000314_H

UPCS layer override register 6 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0			OVRD6												
r			rw												

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OVRD6															
rw															

Field	Bits	Type	Description
OVRD6	28:0	rw	**UPCS override 6**
0	31:29	r	**Reserved** Read as 0; should be written with 0.

19.4.1.30 UPCS layer override register 7

UPCS Override Settings

Note: the settings can be overwritten from these top-level when OVRD17.CTRLSEL is asserted.

UPCS_OVRD7 Offset address: 0000318_H

UPCS layer override register 7 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OVRD7															
rwh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OVRD7															
rwh															

Field	Bits	Type	Description
OVRD7	31:0	rwh	**UPCS override 7**

19.4.1.31 UPCS layer override register 8

UPCS Override Settings

Note: the settings can be overwritten from these top-level when OVRD17.CTRLSEL is asserted.

UPCS_OVRD8 Offset address: $000031C_H$

UPCS layer override register 8 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OVRD8															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OVRD8															
rw															

Field	Bits	Type	Description
OVRD8	31:0	rw	**UPCS override 8**

19.4.1.32 UPCS layer override register 9

UPCS Override Settings

Note: the settings can be overwritten from these top-level when OVRD17.CTRLSEL is asserted.

UPCS_OVRD9 Offset address: 0000320_H

UPCS layer override register 9 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OVRD9															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OVRD9															
rw															

Field	Bits	Type	Description
OVRD9	31:0	rw	**UPCS override 9**

19.4.1.33 UPCS layer override register 10

UPCS Override Settings

Note: the settings can be overwritten from these top-level when OVRD17.CTRLSEL is asserted.

UPCS_OVRD10 Offset address: 0000324_H

UPCS layer override register 10 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0				OVRD10											
r				rw											

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OVRD10															
rw															

Field	Bits	Type	Description
OVRD10	27:0	rw	**UPCS override 10**
0	31:28	r	**Reserved** Read as 0; should be written with 0.

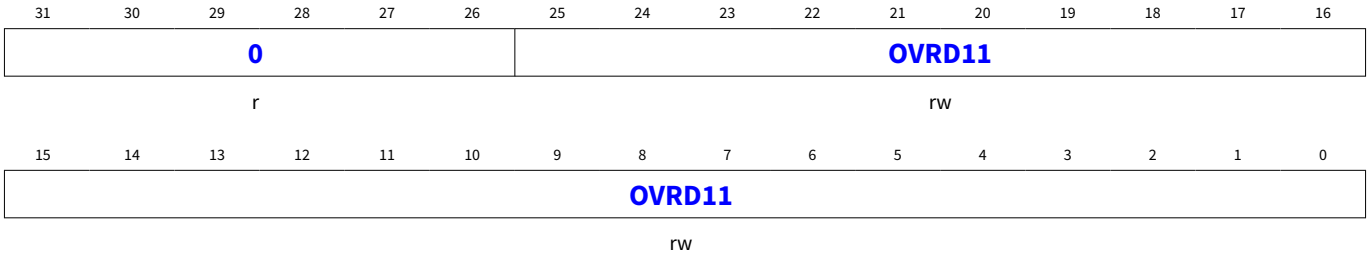
19.4.1.34 UPCS layer override register 11

UPCS Override Settings

Note: the settings can be overwritten from these top-level when OVRD17.CTRLSEL is asserted.

UPCS_OVRD11 Offset address: 0000328_H

UPCS layer override register 11 Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
OVRD11	25:0	rw	**UPCS override 11**
0	31:26	r	**Reserved** Read as 0; should be written with 0.

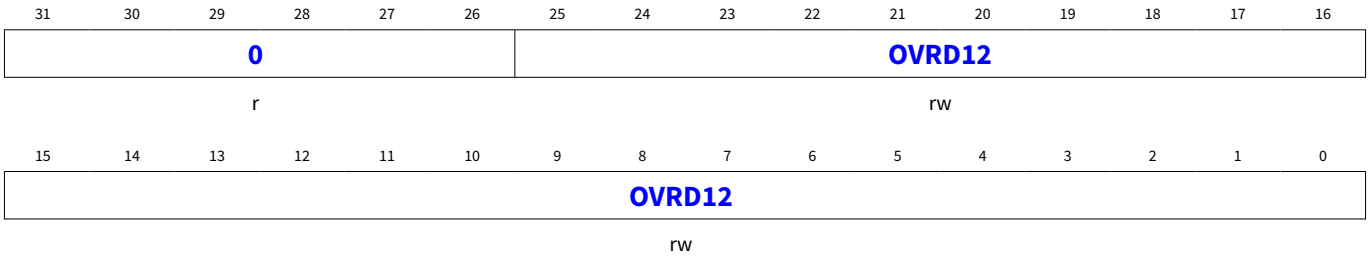
19.4.1.35 UPCS layer override register 12

UPCS Override Settings

Note: the settings can be overwritten from these top-level when OVRD17.CTRLSEL is asserted.

UPCS_OVRD12 Offset address: $000032C_H$

UPCS layer override register 12 Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
OVRD12	25:0	rw	**UPCS override 12**
0	31:26	r	**Reserved** Read as 0; should be written with 0.

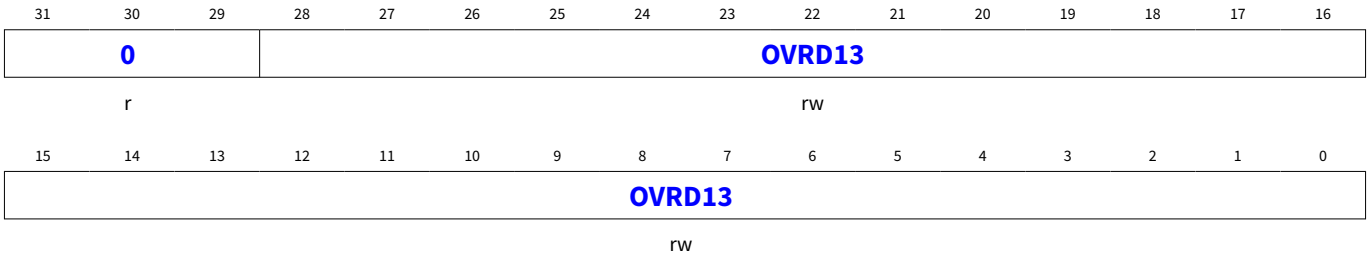
19.4.1.36 UPCS layer override register 13

UPCS Override Settings

Note: the settings can be overwritten from these top-level when OVRD17.CTRLSEL is asserted.

UPCS_OVRD13 Offset address: 0000330_H

UPCS layer override register 13 Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
OVRD13	28:0	rw	**UPCS override 13**
0	31:29	r	**Reserved** Read as 0; should be written with 0.

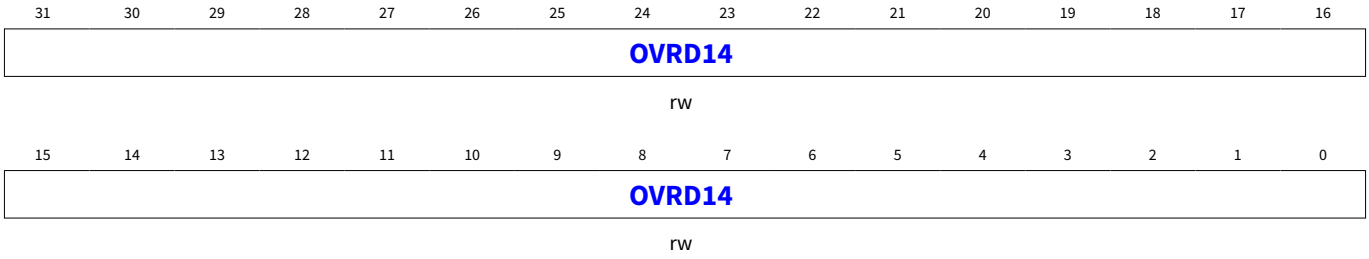
19.4.1.37 UPCS layer override register 14

UPCS Override Settings

Note: the settings can be overwritten from these top-level when OVRD17.CTRLSEL is asserted.

UPCS_OVRD14 Offset address: 0000334_H

UPCS layer override register 14 Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
OVRD14	31:0	rw	**UPCS override 14**

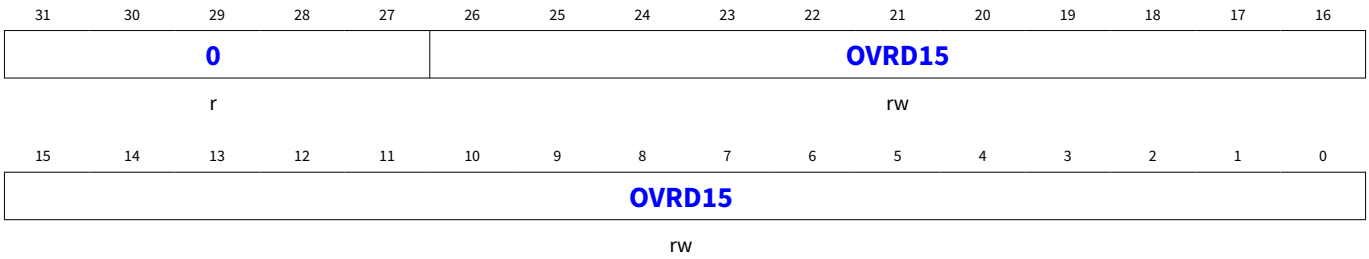
19.4.1.38 UPCS layer override register 15

UPCS Override Settings

Note: the settings can be overwritten from these top-level when OVRD17.CTRLSEL is asserted.

UPCS_OVRD15 Offset address: 0000338_H

UPCS layer override register 15 Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
OVRD15	26:0	rw	**UPCS override 15**
0	31:27	r	**Reserved** Read as 0; should be written with 0.

19.4.1.39 UPCS layer override register 16

UPCS Override Settings

Note: the settings can be overwritten from these top-level when OVRD17.CTRLSEL is asserted.

UPCS_OVRD16 Offset address: $000033C_H$

UPCS layer override register 16 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OVRD16															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OVRD16															
rw															

Field	Bits	Type	Description
OVRD16	31:0	rw	**UPCS override 16**

19.4.1.40 UPCS layer override register 17

UPCS Override Settings

Note: the settings can be overwritten from these top-level when OVRD17.CTRLSEL is asserted.

UPCS_OVRD17 Offset address: 0000340_H

UPCS layer override register 17 Kernel Reset value: $0004\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0													OVRD17		
r													rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OVRD17															
rw															

Field	Bits	Type	Description
OVRD17	18:0	rw	**UPCS override 17**
0	31:19	r	**Reserved** Read as 0; should be written with 0.

19.4.1.41 Request MSI interrupt, EP only.

INT_MSICTRL Offset address: 0000348_H

Request MSI interrupt, EP only. Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
INT31	INT30	INT29	INT28	INT27	INT26	INT25	INT24	INT23	INT22	INT21	INT20	INT19	INT18	INT17	INT16
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
INT15	INT14	INT13	INT12	INT11	INT10	INT9	INT8	INT7	INT6	INT5	INT4	INT3	INT2	INT1	INT0
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

Field	Bits	Type	Description
INTz (z=0-31)	z	w	**MSI interrupt request INT+ $z** Write 1 to generate an MSI interrupt $z MWr TLP to RC. This bit is write only. An interrupt request may be lost if the second request comes before the first request is processed by hardware. The status of request processing is indicated by MSISTAT.PD. 0_B Write access will not be executed 1_B Request an interrupt

19.4.1.42 Pending status of MSI interrupt request, EP only.

INT_MSISTAT Offset address: $000034C_H$

Pending status of MSI interrupt request, EP only. Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PD31	PD30	PD29	PD28	PD27	PD26	PD25	PD24	PD23	PD22	PD21	PD20	PD19	PD18	PD17	PD16
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PD15	PD14	PD13	PD12	PD11	PD10	PD9	PD8	PD7	PD6	PD5	PD4	PD3	PD2	PD1	PD0
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

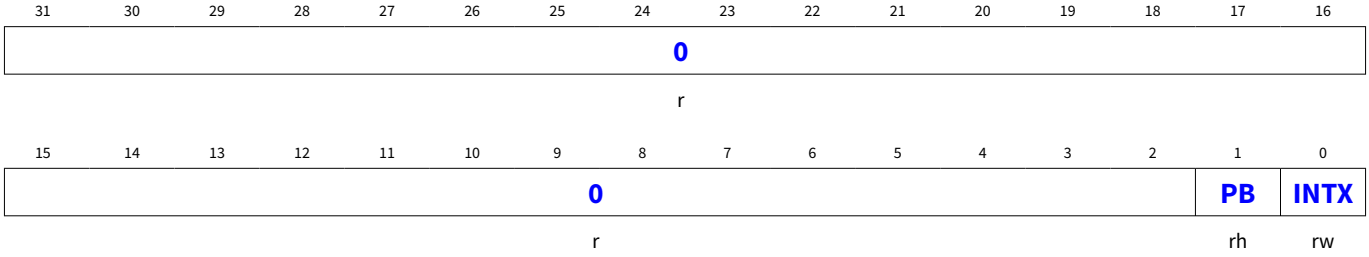
Field	Bits	Type	Description
PDz (z=0-31)	z	rh	**Pending status of interrupt request** Indicates if an MSI interrupt request is pending. 1 means pending and 0 means not pending. 0_B No interrupt request pending 1_B Interrupt pending

19.4.1.43 Trigger INTX interrupt, EP only.

Legay PCI interrupt INTx support, is supposed to be connected to sys_int

INT_INTXCTRL Offset address: 0000350_H

Trigger INTX interrupt, EP only. Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
INTX	0	rw	**INTX interrupt** When this bit goes from low to high, the controller generates an Assert_INTx Message. When it goes from high to low, the controller generates a Deassert_INTx Message. The Interrupt Pin register for the corresponding function determines which INTx Message the controller generates (INTA, INTB, INTC, or INTD). Legacy and native PCIe devices capable of generating an interrupt must support both Assert_INTx/ Deassert_INTx and MSI or MSI-X.
PB	1	rh	**Pending status of INTx interrupt request** Indicates if an INTx interrupt request is pending and the message is not yet sent. 1 means pending and 0 means not pending. 0_B No interrupt request pending 1_B Interrupt pending
0	31:2	r	**Reserved** Read as 0; should be written with 0.

19.4.1.44 Trigger status

After the decoding of the 5 bits, 32 interrupt requests are passed to the Bit control block that consist of three different registers namely EN, STAT and CLR. The Bit control block decides which interrupt is masked in the 32, indicates the current interrupt status and clears the interrupts after service. The above three registers are designed for these functionalities respectively. The 32 interrupt requests will then merged into 8 interrupts that will be handed over to IR and software interrupts are then triggered.

INT_TRIGSTAT Offset address: 0000358_H

Trigger status Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RQ31	RQ30	RQ29	RQ28	RQ27	RQ26	RQ25	RQ24	RQ23	RQ22	RQ21	RQ20	RQ19	RQ18	RQ17	RQ16
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RQ15	RQ14	RQ13	RQ12	RQ11	RQ10	RQ9	RQ8	RQ7	RQ6	RQ5	RQ4	RQ3	RQ2	RQ1	RQ0
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
RQy (y=0-31)	y	rh	**MSI request received** Indicate that interrupt request $y is received. This bit is read only. 0_B No interrupt received 1_B Interrupt received

19.4.1.45 Interrupt clear, EP only.

INT_TRIGCLR Offset address: $000035C_H$

Interrupt clear, EP only. Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RQCLR31	RQCLR30	RQCLR29	RQCLR28	RQCLR27	RQCLR26	RQCLR25	RQCLR24	RQCLR23	RQCLR22	RQCLR21	RQCLR20	RQCLR19	RQCLR18	RQCLR17	RQCLR16
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RQCLR15	RQCLR14	RQCLR13	RQCLR12	RQCLR11	RQCLR10	RQCLR9	RQCLR8	RQCLR7	RQCLR6	RQCLR5	RQCLR4	RQCLR3	RQCLR2	RQCLR1	RQCLR0
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

Field	Bits	Type	Description
RQCLRy (y=0-31)	y	w	**Clears the corresponding incoming interrupt request.** Clear the interrupt request status when the interrupt is serviced. This bit is write only. 0_B Keep the status 1_B Clear the interrupt status

19.4.1.46 Interrupt enable EP only.

INT_TRIGEN Offset address: 0000360_H

Interrupt enable EP only. Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RQEN31	RQEN30	RQEN29	RQEN28	RQEN27	RQEN26	RQEN25	RQEN24	RQEN23	RQEN22	RQEN21	RQEN20	RQEN19	RQEN18	RQEN17	RQEN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RQEN15	RQEN14	RQEN13	RQEN12	RQEN11	RQEN10	RQEN9	RQEN8	RQEN7	RQEN6	RQEN5	RQEN4	RQEN3	RQEN2	RQEN1	RQEN0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RQENy (y=0-31)	y	rw	**Enable bit to block the incoming interrupt to the local interrupt router.** 0_B Disable interrupt 1_B Enable interrupt

19.4.1.47 Receive incoming interrupt request, EP only.

INT_REQTRIG Offset address: 0000364_H

Receive incoming interrupt request, EP only. Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

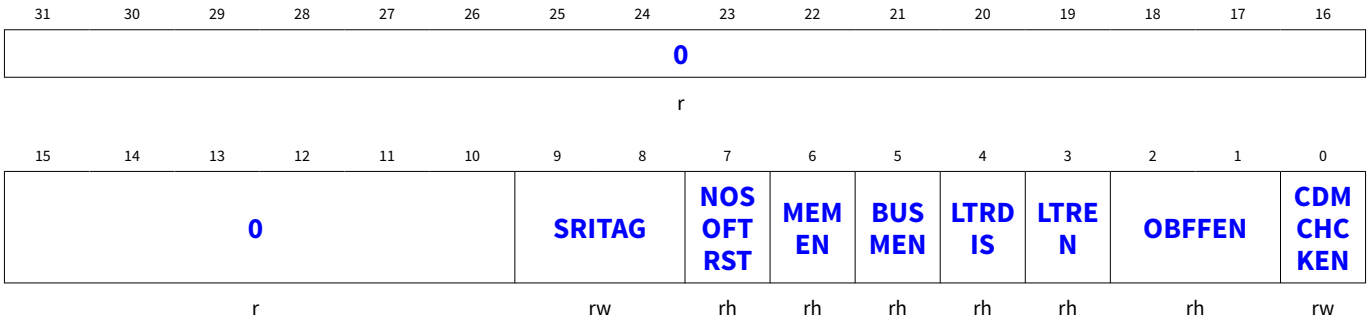
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0											REQTRIG				
r											rh				

Field	Bits	Type	Description
REQTRIG	4:0	rh	**Receive incoming interrupt request** This field can only be written from wire. The received value is parsed as 32 interrupt requests.
0	31:5	r	**Reserved** Read as 0; should be written with 0.

19.4.1.48 Configuration information

CFG_INFO Offset address: 0000368_H

Configuration information Kernel Reset value: $0000\ 0080_H$



Field	Bits	Type	Description
CDMCHCKEN	0	rw	**Signal to enter test mode in the Register Checking Logic.** Asserted to enter test mode in the Register Checking Logic and disable writes to shadow CDM registers. Allows a value to be written to CDM registers to compare against the previous value in shadow CDM registers. If the value written to CDM is different to the value held in shadow CDM registers, comparison error is generated when the address is checked, indicating the Register Content Check function is working correctly. Note: In normal operation mode this bit should be driven to 0.
OBFFEN	2:1	rh	**The OBFF Enable field of the Device Control 2 register of function 0** Indicates The OBFF Enable field of the Device Control 2 register of function 0.
LTREN	3	rh	**The LTR Mechanism Enable field of the Device Control 2 register of function 0.** Indicates the LTR Mechanism Enable field of the Device Control 2 register of function 0.
LTRDIS	4	rh	**Disable the autonomous generation of LTR clear message.**
BUSMEN	5	rh	**The state of the bus master enable bit in the PCI-compatible Command register.** Indicates the state of the bus master enable bit in the PCI-compatible Command register.
MEMEN	6	rh	**The state of the Memory Space Enable bit in the PCI-compatible** Indicates the state of the Memory Space Enable bit in the PCI-compatible Command register.
NOSOFTRST	7	rh	**No Soft Reset bit** This is the value of the No Soft Reset bit in the Power Management Control and Status Register. When set, you should not reset any controller registers when transitioning from D3hot to D0.
SRITAG	9:8	rw	**Selection of the VM or the PRS on the SRI** This bit selects which SRI tag (the VM or the PRS) is used to indicate the DMA index number, when an SRI bus transaction is initiated by the DMA. 00_B No VM/PRS tag used 01_B The VM tag is used in the SRI transaction 10_B The PRS tag is used in the SRI transaction 11_B Reserved, no VM/PRS tag used

(table continues...)

(continued)

Field	Bits	Type	Description
0	31:10	r	**Reserved** Read as 0; should be written with 0.

19.4.1.49 Bus/device number

CFG_BUSNUM Offset address: 000036C$_H$

Bus/device number Kernel Reset value: 0000 0000$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0			SUBBUSNUM								SECBUSNUM				
r			rh								rh				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SECBUSNUM			DEVNUM					PBUSNUM							
rh			rh					rh							

Field	Bits	Type	Description
PBUSNUM	7:0	rh	**The primary bus number assigned to the function.**
DEVNUM	12:8	rh	**The device number assigned to the function.**
SECBUSNUM	20:13	rh	**Configured Secondary Bus Number.** The secondary bus number assigned to the device. When the host software detects the upstream switch port, it assigns a secondary bus number. This bus number is then used in configuration requests to detect the downstream ports. The switch controller must see this so that it can route configuration requests.
SUBBUSNUM	28:21	rh	**Configured Subordinate Bus Number.** When the switch controller receives a configuration request from the upstream port, it must see the secondary and subordinate bus number fields from the downstream ports to route the request.
0	31:29	r	**Reserved** Read as 0; should be written with 0.

19.4.1.50 Select device/port type

LNK_DVCTYP Offset address: 0000370$_H$

Select device/port type PowerOn Reset value: 0000 0000$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0											PWD	TYP			
r											rh	rw			

Field	Bits	Type	Description
TYP	3:0	rw	**Device/port type** Indicates the specific type of this PCI Express function. It is also used to set the 'Device/Port Type' field of the 'PCI Express Capabilities Register'. The controller uses this input to determine the operating mode of the controller at run time. Encodings except for the below ones are not supported. 0_H PCIe endpoint 4_H PCIe root complex
PWD	4	rh	**Power down status of the PCIe controller** Indicate the power-down status of the PCIe controler. The power-down status can be triggered by PERST# reset, SW kernal reset, the hot reset and link reset. The S2A bridge and the SPB slave interface will refuse to accept new bus request when this bit is asserted.
0	31:5	r	**Reserved** Read as 0; should be written with 0.

19.4.1.51 Indicate link status

LNK_STAT Offset address: 0000374_H

Indicate link status Kernel Reset value: $0000\ 0004_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0									PERSTINT	EQINT	BWINT	LINKINT	LINKRQRST	Res	LINKUP
r									rh	rh	rh	rh	rh	r	rh

Field	Bits	Type	Description
LINKUP	0	rh	**PHY Link up/down indicator** 0_B Link is down 1_B Link is up
LINKRQRST	2	rh	**Reset request** Reset request because the link has gone down or the controller received a hot-reset request. A low level indicates that the controller is requesting external logic to reset the controller because the PHY link is down. The link reset request is delayed until all pending bus transfers have completed.

(table continues...)

(continued)

Field	Bits	Type	Description
LINKINT	3	rh	**Link Autonomous Bandwidth Interrupt. The interrupt can be triggered when the LNK_INTEN.LINKINTEN is asserted.** indicate that the Link Autonomous Bandwidth Status bit has been set. This bit is asserted when all of the following conditions are true When INTx is used • The INTx assertion disable bit in the Command register is 0, and • The Link Autonomous Bandwidth Interrupt Enable bit in the Link Control register is set to 1, and • The Link Autonomous Bandwidth Interrupt Status bit in the Link Status register is set to 1 Or when MSI is used: • MSI or MSI-X is enabled. • The Link Autonomous Bandwidth Status register (Link Status register bit 15) is updated. • The Link Autonomous Bandwidth Interrupt Enable (Link Control register bit 11) is set.
BWINT	4	rh	**Link Bandwidth Management Interrupt. The interrupt can be triggered when the LNK_INTEN.BWINTEN is asserted.** Indicate that the Link Bandwidth Management Status bit has been set. This bit is asserted when all of the following conditions are true: When INTx is used • The INTx assertion disable bit in the Command register is 0, and • The Bandwidth Management Interrupt Enable bit in the Link Control register is set to 1, and • The Bandwidth Management Interrupt Status bit in the Link Status register is set to 1. Or when MSI is used: • MSI or MSI-X is enabled. • The Link Bandwidth Management Status register (Link Control Status register bit 14) is updated • The Link Bandwidth Management Interrupt Enable (Link Control register bit 10) is set..
EQINT	5	rh	**Link Equalization Request. The interrupt can be triggered when the LNK_INTEN.EQINTEN is asserted.** Interrupt indicating to your application that the Link Equalization Request bit in the Link Status 2 Register has been set and the Link Equalization Request Interrupt Enable (Link Control 3 Register bit 1) is set.
PERSTINT	6	rh	**Interrupt status of the PERST#. The interrupt can be triggered when the LNK_INTEN.PERSTEN is asserted.** It indicates the PERST# interrupt (detecting PERST# become active 0) arrives.
0	31:7	r	**Reserved** Read as 0; should be written with 0.

19.4.1.52 Controller control

LNK_CORECTRL Offset address: 0000378_H

Controller control Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0														CLKREQEN	WAKEN
r														rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CLKREQ		WAKE		PERST		PERSTCLR	PERSTOVR	PERSTVAL	PERSTSTA	HOTRST	0	PFRQRTYEN	RQRTYEN	LTSSMEN	SRISMD
rw		rw		rw		w	rw	rw	rh	w	r	rw	rw	rw	rw

Field	Bits	Type	Description
SRISMD	0	rw	**SRIS operating mode** 0_B non-SRIS mode 1_B SRIS mode
LTSSMEN	1	rw	**Start link training.** Driven low default after cold, warm or hot reset to hold the LTSSM in the Detect state until your application is ready for the link training to begin. 0_B Hold LTSSM and delay link training 1_B Start link training
RQRTYEN	2	rw	**Defer incoming configuration requests** Provides a capability to defer incoming configuration requests until initialization is complete. When this bit is asserted, the controller completes incoming configuration requests with a configuration request retry status. Other incoming requests complete normally. When the Readiness Notification mechanism is supported, DRS messaging is blocked when RQRTYEN=1. When RQRTYEN=0 and MSG_INFO.DRSRDY=1, the controller autonomously transmits a DRS message when the link transitions from DL_Down to DL_Up.

(table continues...)

(continued)

Field	Bits	Type	Description
PFRQRTYEN	3	rw	**Per Physical Function (PF) to defer incoming configuration requests** Provides a per Physical Function (PF) capability to defer incoming configuration requests until initialization is complete. When PFRQRTYEN is asserted for a certain PF, the controller completes incoming configuration requests with a configuration request retry status only for that particular PF; other incoming requests complete normally. When RQRTYEN is also asserted, the controller completes incoming configuration requests with a configuration request retry status for ALL PFs and VFs, regardless of the status of PFRQRTYEN. When the Readiness Notification mechanism is supported, DRS messaging is blocked when PFRQRTYEN =1 for at least one PF. The same applies when PFRQRTYEN =0 for all PFs, but RQRTYEN is asserted. When PFRQRTYEN =0 for a certain PF, and RQRTYEN =0, and MSG_INFO.DRSRDY=1, the controller autonomously transmits a DRS message when the link transitions from DL_Down to DL_Up.
HOTRST	5	w	**Request from your application to send a hot reset to the upstream port, RC only.** A downstream port (DSP) can hot reset an upstream port (USP) by sending two consecutive TS1 ordered sets with the hot reset bit asserted. This field requests from your application to send a hot reset to the upstream port. In an upstream port, you should not use this bit. Note: This bit is not used by the controller to set the SBR field in the BRIDGE_CTRL_INT_PIN_INT_LINE_REG register. During the transition from DL_Active to DL_inactive, assertion of this bit indicates a Surprise Down Error, but setting of SBR field in the BRIDGE_CTRL_INT_PIN_INT_LINE_REG register through FPI does not trigger a Surprise Down Error.
PERSTSTA	6	rh	**PERST# pin status** Indicate the PERST# pin status. 0_B Asserted (active low) 1_B Deasserted (active low)
PERSTVAL	7	rw	**PERST# override value** This value decides the override value of the PERST# when the PERSTOVR is enabled. 0_B Asserted (active low) 1_B Deasserted (active low)
PERSTOVR	8	rw	**PERST# override enable** When this bit is enabled, the PERSTVAL bit will override the PERST# pin value in the controller.
PERSTCLR	9	w	**Clear the interrupt status of the PERSTINT.** Clear the interrupt status of the PERSTINT. 0_B Keep the status 1_B Clear the interrupt status

(table continues...)

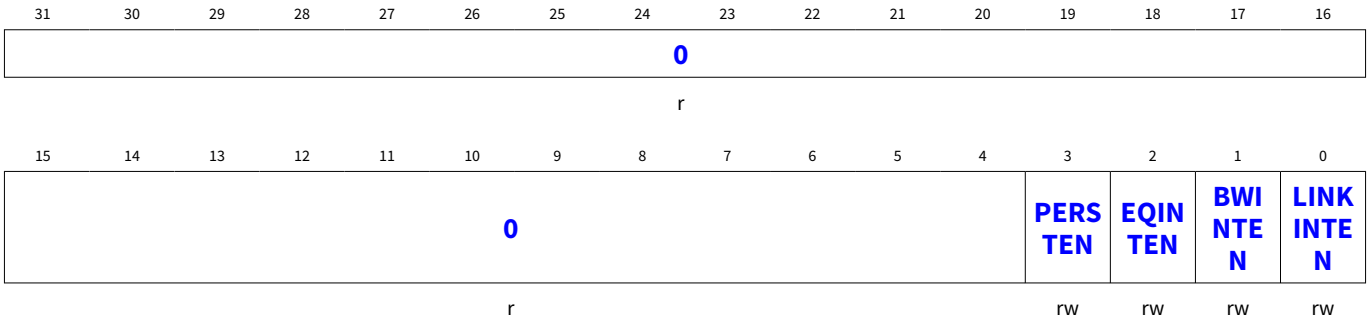
(continued)

Field	Bits	Type	Description
PERST	11:10	rw	**Alternate Input Select** Selects the alternate input for PERST#. 00_B Alternate Input A PERSTA_N 01_B Alternate Input B PERSTB_N 10_B Alternate Input A PERSTA_N 11_B Alternate Input A PERSTA_N
WAKE	13:12	rw	**Alternate input and output select** Selects the alternate input and output for WAKE#. 00_B Alternate input and output A WAKEA_IN_N 01_B Alternate input and output B WAKEB_IN_N 10_B Alternate input and output A WAKEA_IN_N 11_B Alternate input and output A WAKEA_IN_N
CLKREQ	15:14	rw	**Alternate input and output select** Selects the alternate input and output for CLKREQ#. 00_B Alternate input and output A CLKREQA_IN_N 01_B Alternate input and output B CLKREQB_IN_N 10_B Alternate input and output A CLKREQA_IN_N 11_B Alternate input and output A CLKREQA_IN_N
WAKEN	16	rw	**WAKE# enable control.** Enable the WAKE# input and output. If this bit is disabled, the alternative selection of the WAKE bit will have no effect 0_B WAKE# disable. None of the alternative pins is driven. 1_B WAKE# enable. One of the alternative pins is driven.
CLKREQEN	17	rw	**CLKREQ# enable control.** Enable the CLKREQ# input and output. If this bit is disabled, the alternative selection of the CLKREQ bit will have no effect 0_B CLKREQ# disable. None of the alternative pins is driven. 1_B CLKREQ# enable. One of the alternative pins is driven.
0	4, 31:18	r	**Reserved** Read as 0; should be written with 0.

19.4.1.53 Enable the interrupt requests to SRC_PCIELNK

LNK_INTEN Offset address: $000037C_H$

Enable the interrupt requests to SRC_PCIELNK Kernel Reset value: $0000\ 0000_H$

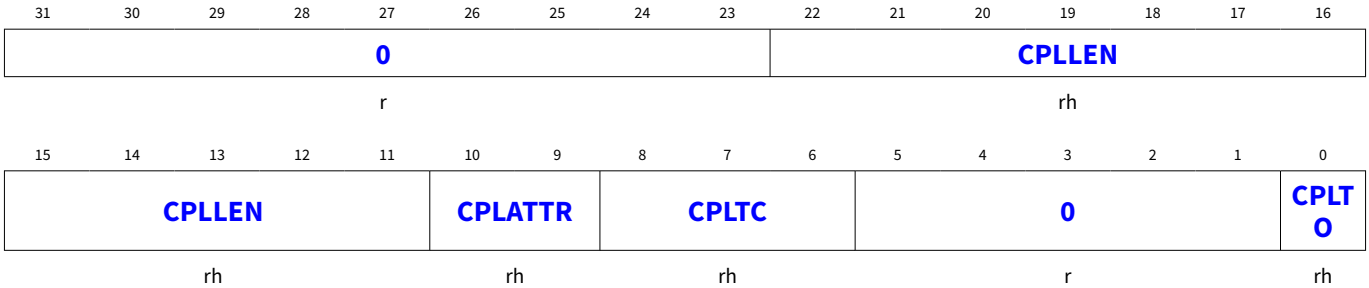


Field	Bits	Type	Description
LINKINTEN	0	rw	**Enable the LINKINT** 0_B Disable interrupt 1_B Enable interrupt
BWINTEN	1	rw	**Enable the BWINT bit.** 0_B Disable interrupt 1_B Enable interrupt
EQINTEN	2	rw	**Enable the EQINT bit.** 0_B Disable interrupt 1_B Enable interrupt
PERSTEN	3	rw	**Enable the PERSTINT bit.** 0_B Disable interrupt 1_B Enable interrupt
0	31:4	r	**Reserved** Read as 0; should be written with 0.

19.4.1.54 Rx control

TXRX_RXCTRL Offset address: 0000380_H

Rx control Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
CPLTO	0	rh	**Indicates that the completion TLP for a request has not been received**
CPLTC	8:6	rh	**The Traffic Class of the timed out completion**

(table continues...)

(continued)

Field	Bits	Type	Description
CPLATTR	10:9	rh	**The Attributes field of the timed out completion**
CPLLEN	22:11	rh	**Length (in bytes) of the timed out completion.** For a split completion, it indicates the number of bytes remaining to be delivered when the completion timed out.
0	5:1, 31:23	r	**Reserved** Read as 0; should be written with 0.

19.4.1.55 Tx control 0

TXRX_TXCTRL0 Offset address: 0000384_H

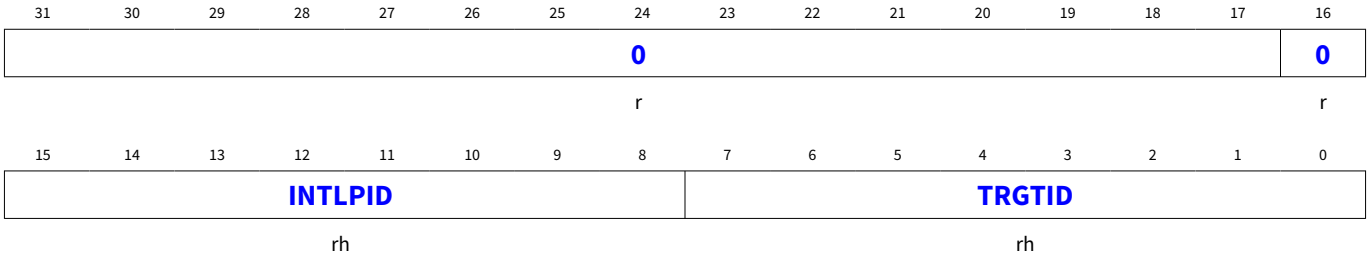
Tx control 0 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0							CPLLEN								
r							rh								

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CPLLEN			CPLATTR		CPLTC			CPLFNUM			CPLTO	PRTYERR			BLCKTLP
rh			rh		rh			rh			rh	rh			rh

Field	Bits	Type	Description
BLCKTLP	0	rh	**Block TLP generation** Indicates that your application must stop generating new outgoing request TLPs due to the current power management state. Your application can continue to generate completion TLPs. This output is not used in RC mode.
PRTYERR	3:1	rh	**Indicates that the controller detected a datapath parity error** The controller handles the parity error as an uncorrectable error. • PRTYERR[0]: Parity error at front end of the transmit datapath. • PRTYERR[1]: Parity error at back end of the transmit datapath. • PRTYERR[2]: Parity error the receive datapath. They do not indicate which packet contained the parity error.
CPLTO	4	rh	**No completion for an incoming request** Information about the timed out completion is available the fields listed later. When a completion timeout occurs, the controller does not clear the corresponding entry from the completion lookup table.
CPLFNUM	7:5	rh	**The function number of the timed out completion.**
CPLTC	10:8	rh	**The TC of the timed out completion.**
CPLATTR	12:11	rh	**The Attributes value of the timed out completion.**
CPLLEN	24:13	rh	**The Length of the timed out completion.**
0	31:25	r	**Reserved** Read as 0; should be written with 0.

19.4.1.56 Tx control 1

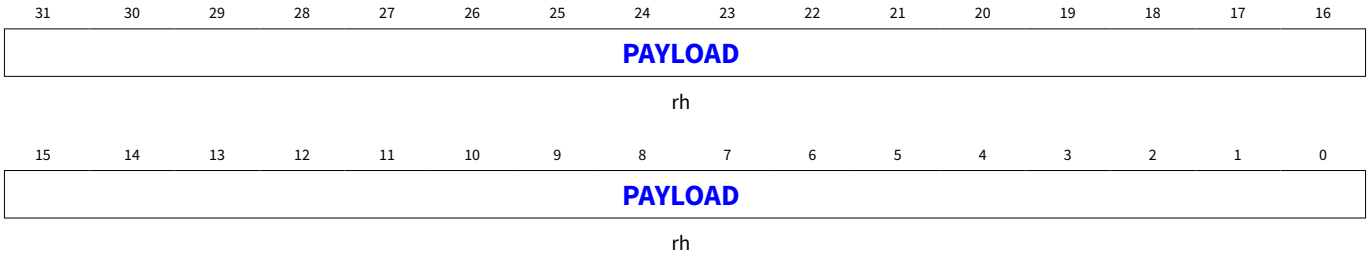


Field	Bits	Type	Description
TRGTID	7:0	rh	**The target completion LUT lookup ID of the timed out completion**
INTLPID	15:8	rh	**The target completion LUT lookup ID for the incoming request TLP.**
0	16, 31:17	r	**Reserved** Read as 0; should be written with 0.

19.4.1.57 Message payload z information

PAYLOADz (z=0-1) Offset address: $000038C_H+z*4$

Message payload z information Kernel Reset value: $0000\ 0000_H$



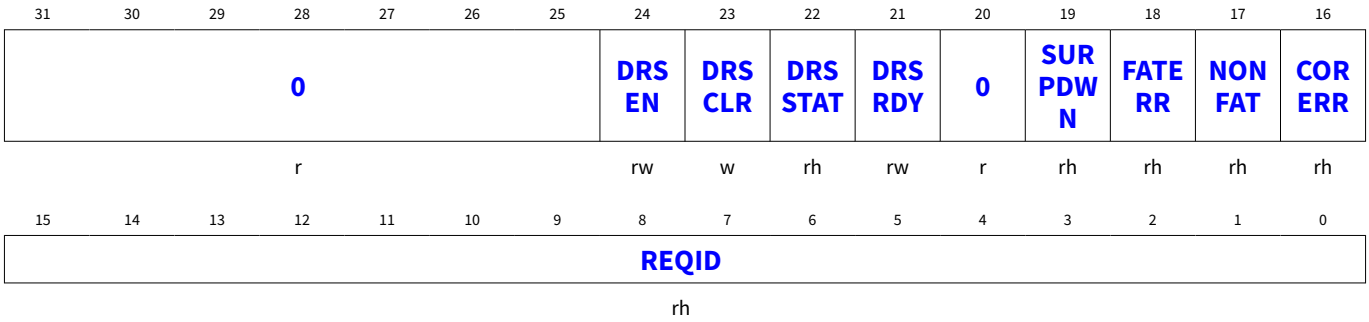
Field	Bits	Type	Description
PAYLOAD	31:0	rh	**Received message header information.** When a ltr message is received, the controller maps the message payload to the Rx TLP header dwords as follows: • PAYLOAD0 = bytes 12-15 (4th dword), where [7:0] =byte 15 • PAYLOAD1 = bytes 8-11 (3rd dword)

19.4.1.58 Message information

MSG_INFO Offset address: 0000394_H

Message information Kernel Reset value: $0000\ 0000_H$

19 Peripheral Component Interconnect Express (PCIe)



Field	Bits	Type	Description
REQID	15:0	rh	**The requester ID of the received Message.** • [2:0]: Function number • [7:3]: Device number • [15:8]: Bus number
CORERR	16	rh	**Sent Correctable Error, EP only.** Core has sent a message towards the Root Complex indicating that an Rx TLP that contained an error, and that can be corrected, was received by the EndPoint.
NONFAT	17	rh	**Sent Non-Fatal Error, EP only.** Controller has sent a message towards the Root Complex indicating that an Rx TLP that contained an non-fatal error, and that can not be corrected, was received by the EndPoint.
FATERR	18	rh	**Sent Fatal Error, EP only.** Controller has sent a message towards the Root Complex indicating that an Rx TLP that contained a fatal error, and that can not be corrected, was received by the EndPoint
SURPDWN	19	rh	**Indicates that a surprise down event is occurring in the controller.**
DRSRDY	21	rw	**Defers DRS messaging when set to '0', EP only.**
DRSSTAT	22	rh	**DRS Message Received Interrupt, RC only. The interrupt can be triggered when the DRSEN is asserted.** The DSP controller asserts this bit when all of the following are true: • It receives a DRS message • PCIE_CAP_DRS_SIGNALING_CONTROL in LINK_CONTROL_LINK_STATUS_REG is 2'b01 • MSI or MSI-X is enabled
DRSCLR	23	w	**Clear the DRS state, RC only.** 0_B Keep the status 1_B Clear the interrupt status
DRSEN	24	rw	**Enable the DRS interrupt, RC only.** 0_B Disable interrupt 1_B Enable interrupt
0	20, 31:25	r	**Reserved** Read as 0; should be written with 0.

19.4.1.59 Power management control

PM_CTRL Offset address: 0000398_H

Power management control Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0														TRN	TRNACK
r														rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PMEMSG	TURNOFF	0				CLKRM	PMEINT	CLKPMEN	CLKREQ	XFERPND	REQEXITL1	RDYENTRL23	REQENTRL1	AUXPWRDET	WKUPCMD
rh	w	r				rw	rh	rw	rh	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
WKUPCMD	0	rw	**Wake Up.** Wake Up. If PME is enabled and PME support is configured for current PMCSR D-state asserting this signal causes the controller to wake from either L1 or L2 state.
AUXPWRDET	1	rw	**Auxiliary Power Detected.** Used to report to the host software that auxiliary power (Vaux) is present.
REQENTRL1	2	rw	**Application request to Enter L1 ASPM state.** This bit is for use by applications that need to control L1 entry instead of using the L1 entry timer as defined in the PCI Express Specification. It is only effective when L1 is enabled. The controller latches this request when in L0 or L0s; to be acted upon later. Note: The controller ignores this input in RC mode
RDYENTRL23	3	rw	**Application Ready to Enter L23.** Indication from your application that it is ready to enter the L23 state. This bit is provided for applications that must control L23 entry (in case certain tasks must be performed before going into L23). The controller delays sending PM_Enter_L23 (in response to PM_Turn_Off) until this signal becomes active. When this signal has been asserted by the application, it must be kept asserted until L2 entry has completed.
REQEXITL1	4	rw	**Application request to Exit L1.** Request from your application to exit L1. It is only effective when L1 is enabled.

(table continues...)

(continued)

Field	Bits	Type	Description
XFERPND	5	rw	**Indicates that your application has transfers pending** If the entry into L1 is already in progress, assertion of app_xfer_pending causes an exit from L1. This is a level signal used to inform the controller about the state of external queues and pipeline stages that contain transactions to be transmitted by the controller. The controller uses this information to determine when to enter/exit L1. When this signal is asserted, it indicates that there are transactions outside the controller that the controller needs to transmit. When de-asserted, it indicates that there are no transactions outside the controller. The controller responds to an assertion on this signal as follows: • Upstream Ports: Prevents generation of requests to enter L1. Triggers exit if already in L1. • Downstream Ports: Triggers exit if already in L1.
CLKREQ	6	rh	**Status of the CLKREQ#** Status of the CLKREQ# bidirectional CMOS board-level signal. This should be connected to the GPIO. Used by the controller to determine when to enter and exit L1 Substates when using the CLKREQ#-based mechanism. For more details, see 'L1 Substates'.
CLKPMEN	7	rw	**Clock PM feature enabled by application.** Used to inhibit the programming of the Clock PM in Link Control Register.
PMEINT	8	rh	**PME interrupt received. The interrupt can be triggered when the PM_INTEN.PMEINTEN is asserted.** The controller asserts this bit when all of the following conditions are true: • The INTx Assertion Disable bit in the Command register is 0. • The PME Interrupt Enable bit in the Root Control register is set to 1. • The PME Status bit in the Root Status register is set to 1. Or • MSI or MSI-X is enabled. • The PME Interrupt Enable bit in the Root Control register is set to 1. • The PME Status bit in the Root Status register is set to 1.
CLKRM	9	rw	**Indicates that the application is ready to have reference clock removed.** The application should set this bit to 1'b1 if it supports reference clock removal. If the application does not want to remove reference clock it should set this bit to 1'b0. This bit should be asserted or de-asserted when L1 CPM or L1 Sub-states are being configured, it should not be changed dynamically during L1.
TURNOFF	14	w	**Request from your application to generate a PM_Turn_Off message.** You must assert this signal for one clock cycle. You must not assert the same bit again, until the previous message has been transmitted.
PMEMSG	15	rh	**The controller received a PM_PME message.The interrupt can be triggered when the PM_INTEN.PMEMSGEN is asserted.**

(table continues...)

(continued)

Field	Bits	Type	Description
TRNACK	16	rh	**The controller received a PME_TO_Ack message. Upstream port: Reserved. The interrupt can be triggered when the PM_INTEN.TOACKEN is asserted.**
TRN	17	rh	**The controller received a PME Turnoff message. The interrupt can be triggered when the PM_INTEN.TOEN is asserted.**
0	13:10, 31:18	r	**Reserved** Read as 0; should be written with 0.

19.4.1.60 Power management status 0

PM_STAT0 Offset address: 000039C$_H$

Power management status 0 Kernel Reset value: 0000 0000$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
L1SUBSTAT			SLVSTAT					MSTRSTAT					LINKSTL2E	LINKSTL2	LINKSTL1
rh			rh					rh					rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	DSTAT			0	WAKE	LTSSMSTAT						CURNTSTAT		
r	r	rh			r	rh	rh						rh		

Field	Bits	Type	Description
CURNTSTAT	2:0	rh	**Indicates the current power state.** 000_B L0 000_B L0s 000_B L1 000_B L2 (Aux Power Present) 000_B L3

(table continues...)

(continued)

Field	Bits	Type	Description
LTSSMSTAT	8:3	rh	**Current state of the LTSSM. Encoding is defined as follows:** 00_H S_DETECT_QUIET 01_H S_DETECT_ACT 02_H S_POLL_ACTIVE 03_H S_POLL_COMPLIANCE 04_H S_POLL_CONFIG 05_H S_PRE_DETECT_QUIET 06_H S_DETECT_WAIT 07_H S_CFG_LINKWD_START 08_H S_CFG_LINKWD_ACEPT 09_H S_CFG_LANENUM_WAI $0A_H$ S_CFG_LANENUM_ACEPT $0B_H$ S_CFG_COMPLETE $0C_H$ S_CFG_IDLE $0D_H$ S_RCVRY_LOCK $0E_H$ S_RCVRY_SPEED $0F_H$ S_RCVRY_RCVRCFG 10_H S_RCVRY_IDLE 11_H S_L0 12_H S_L0S 13_H S_L123_SEND_EIDLE 14_H S_L1_IDLE 15_H S_L2_IDLE 16_H S_L2_WAKE 17_H S_DISABLED_ENTRY 18_H S_DISABLED_IDLE 19_H S_DISABLED $1A_H$ S_LPBK_ENTRY $1B_H$ S_LPBK_ACTIVE $1C_H$ S_LPBK_EXIT $1D_H$ S_LPBK_EXIT_TIMEOUT $1E_H$ S_HOT_RESET_ENTRY $1F_H$ S_HOT_RESET 20_H S_RCVRY_EQ0 21_H S_RCVRY_EQ1 22_H S_RCVRY_EQ2 23_H S_RCVRY_EQ3
WAKE	9	rh	**Wake Up. Driving WAKE# for USP**
DSTAT	13:11	rh	**The current power management D-state of the function:** 000_B D0 001_B D1 010_B D2 011_B D3 100_B Uninitialized 101_B Reserved

(table continues...)

(continued)

Field	Bits	Type	Description
LINKSTL1	16	rh	**Power management is in L1 state.**
LINKSTL2	17	rh	**Power management is in L2 state.**
LINKSTL2E	18	rh	**Power management is exiting L2 state.** Not applicable for downstream port.
MSTRSTAT	23:19	rh	**Power management master FSM state.** OTGB trace
SLVSTAT	28:24	rh	**Power management slave FSM state.** OTGB trace
L1SUBSTAT	31:29	rh	**Power management L1 sub-states FSM state**
0	10, 14, 15	r	**Reserved** Read as 0; should be written with 0.

19.4.1.61 Power management status 1

PM_STAT1 Offset address: 00003A4$_H$

Power management status 1 Kernel Reset value: 0000 0000$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

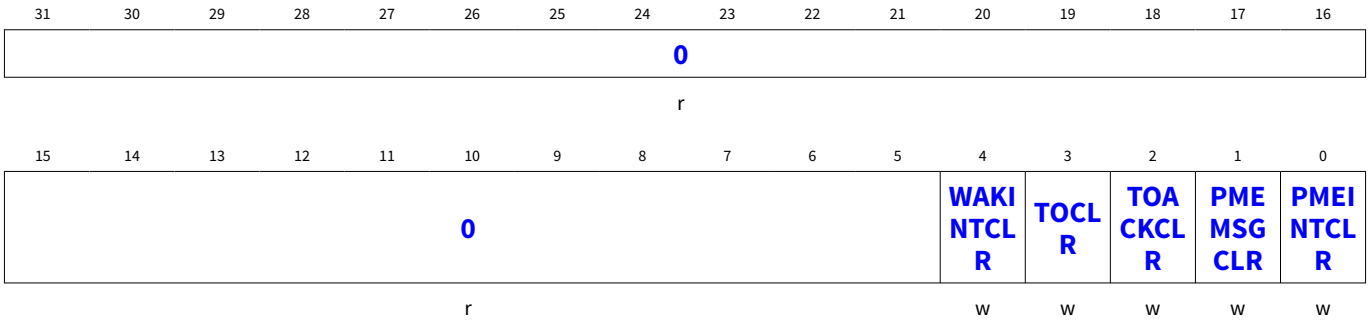
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												0	LINKSTL1SUB	0	WAKINT
r												r	rh	r	rh

Field	Bits	Type	Description
WAKINT	0	rh	**For DSP, indicate that the WAKE# is driven from high to low. The interrupt can be triggered when the PM_INTEN.WAKINTEN is asserted.** An interrupt can be triggered by this transition.
LINKSTL1SUB	2	rh	**Power management is in L1 substate.** Indicates when the link has entered L1 substates.
0	1, 3, 31:4	r	**Reserved** Read as 0; should be written with 0.

19.4.1.62 Power management interrupt clear

PM_INTCLR Offset address: 00003A8$_H$

Power management interrupt clear Kernel Reset value: 0000 0000$_H$



Field	Bits	Type	Description
PMEINTCLR	0	w	**Clear the PMEINT bit.**
PMEMSGCLR	1	w	**Clear the PMEMSG bit.**
TOACKCLR	2	w	**Clear the TRNACK bit.**
TOCLR	3	w	**Clear the TRN bit.**
WAKINTCLR	4	w	**Clear the WAKINT bit.**
0	31:5	r	**Reserved** Read as 0; should be written with 0.

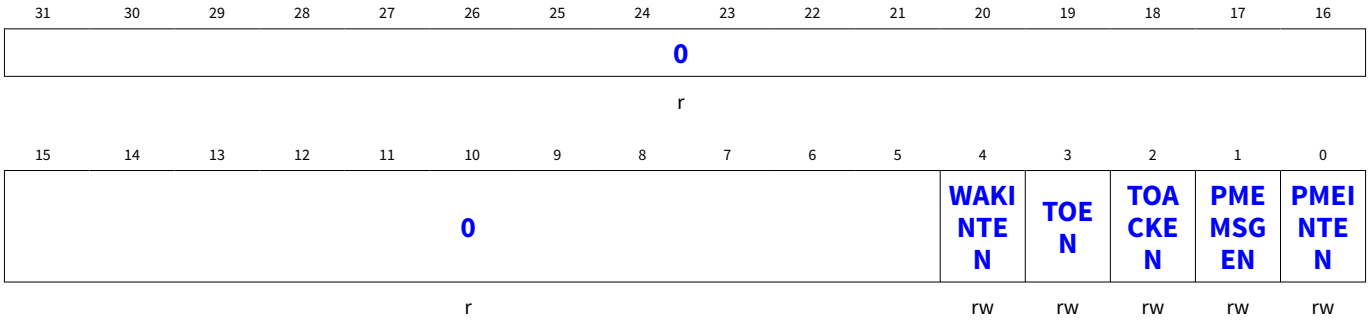
19.4.1.63 Power management interrupt enable

PM_INTEN

Offset address: $00003AC_H$

Power management interrupt enable

Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
PMEINTEN	0	rw	**Enable PME interrupt** 0_B Disable interrupt 1_B Enable interrupt
PMEMSGEN	1	rw	**Enable the PMEMSG interrupt** 0_B Disable interrupt 1_B Enable interrupt
TOACKEN	2	rw	**Enable the TRNACK interrupt** 0_B Disable interrupt 1_B Enable interrupt

(table continues...)

(continued)

Field	Bits	Type	Description
TOEN	3	rw	**Enable the TRN interrupt** 0_B Disable interrupt 1_B Enable interrupt
WAKINTEN	4	rw	**Enable the WAKINT interrupt** 0_B Disable interrupt 1_B Enable interrupt
0	31:5	r	**Reserved** Read as 0; should be written with 0.

19.4.1.64 Error status

ERR_STAT Offset address: $00003B0_H$

Error status Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

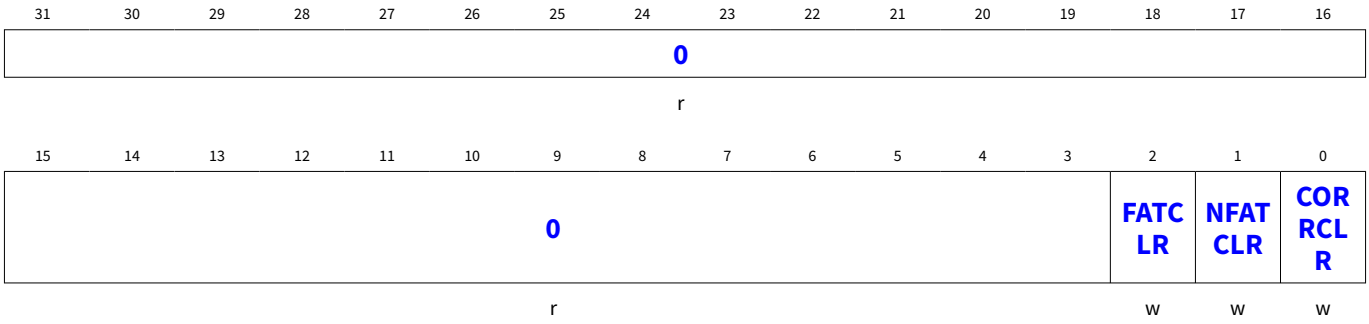
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0													FAT	NFAT	CORR
r													rh	rh	rh

Field	Bits	Type	Description
CORR	0	rh	**Received an ERR_COR message. The interrupt can be triggered when the ERR_INTEN.CORREN is asserted.** Indicates that the controller received an ERR_COR message. The controller makes the message header available in the MSG_PAYLOAD.
NFAT	1	rh	**Indicates that the controller received an ERR_NONFATAL message. The interrupt can be triggered when the ERR_INTEN.NFATEN is asserted.**
FAT	2	rh	**Indicates that the controller received an ERR_FATAL message. The interrupt can be triggered when the ERR_INTEN.FATEN is asserted.**
0	31:3	r	**Reserved** Read as 0; should be written with 0.

19.4.1.65 Error clear

ERR_INTCLR Offset address: $00003B4_H$

Error clear Kernel Reset value: $0000\ 0000_H$

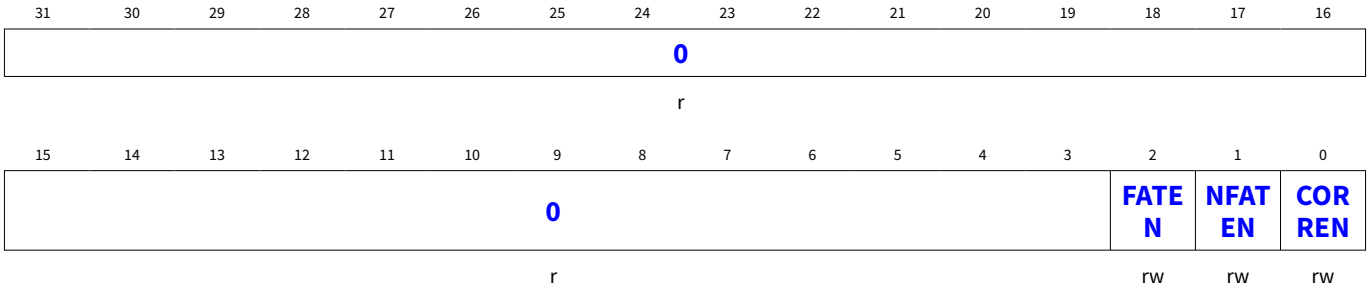


Field	Bits	Type	Description
CORRCLR	0	w	**Clear the CORR bit.** 0_B Keep the status 1_B Clear the interrupt status
NFATCLR	1	w	**Clear the NFAT bit.** 0_B Keep the status 1_B Clear the interrupt status
FATCLR	2	w	**Clear the FAT bit.** 0_B Keep the status 1_B Clear the interrupt status
0	31:3	r	**Reserved** Read as 0; should be written with 0.

19.4.1.66 Error enable

ERR_INTEN Offset address: $00003B8_H$

Error enable Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
CORREN	0	rw	**Enable the CORR bit.** 0_B Disable interrupt 1_B Enable interrupt
NFATEN	1	rw	**Enable the NFAT bit.** 0_B Disable interrupt 1_B Enable interrupt

(table continues...)

(continued)

Field	Bits	Type	Description
FATEN	2	rw	**Enable the FAT bit.** 0_B Disable interrupt 1_B Enable interrupt
0	31:3	r	**Reserved** Read as 0; should be written with 0.

19.4.1.67 Debug status

DBG_STAT Offset address: $00003BC_H$

Debug status Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0									Res	QNEMPTY	0	XFERPD	DMAPD	BRDGPD	LINKUP
r									r	rh	r	rh	rh	rh	rh

Field	Bits	Type	Description
LINKUP	0	rh	**Data link layer up/down indicator:** This status from the flow control initialization state machine indicates that flow control has been initiated and the Data link layer is ready to transmit and receive packets.
BRDGPD	1	rh	**AXI Slave non-DBI transfer pending status.** Indicates AXI Slave Read or Write transfers are pending, that is, AXI Slave transfers are awaiting a response from the controller. For debugging purposes.
DMAPD	2	rh	**eDMA transfer pending status.** Indicates eDMA Write or Read Channel transfers are pending, that is, DMA Write or Read Channels have not finished transferring data. For debugging purposes.
XFERPD	3	rh	**Receive request pending status.** Indicates Receive TLP requests are pending, that is, requests sent to the bus interfaces are awaiting a response from your application. For debugging purposes.
QNEMPTY	5	rh	**Level indicating that the receive queues contain TLP header/data**
0	4, 31:7	r	**Reserved** Read as 0; should be written with 0.

19.4.1.68 Debug information

DBG_DATA Offset address: $00003C0_H$

Debug information Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RFCDAT															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RFCDAT															
rh															

Field	Bits	Type	Description
RFCDAT	31:0	rh	**The data from the previously received flow control update DLLP.** .

19.4.1.69 LTR message

LTR_GNRTN Offset address: $00003C4_H$

LTR message Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LTRLAT															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LTRLAT															
rw															

Field	Bits	Type	Description
LTRLAT	31:0	rw	**LTR message that your application is requesting to send.**

19.4.1.70 LTR control

LTR_CTRL Offset address: $00003CC_H$

LTR control Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0						RLTREN	RLTRCLR	RLTR	0						MSGREQ
r						rw	w	rh	r						rwh

Field	Bits	Type	Description
MSGREQ	0	rwh	**Trigger to send an LTR message.** Once asserted, it must remain asserted until the controller grants the reqeust. It gets cleared when the request is granted. .
RLTR	7	rh	**Received an LTR message. The interrupt can be triggered when the RLTREN is asserted.** The controller makes the message header available the MSG.PAYLOAD. It is also available the PL_LTR_LATENCY_OFF.
RLTRCLR	8	w	**Clear the RLTR bit.** 0_B Keep the status 1_B Clear the interrupt status
RLTREN	9	rw	**Enable the RLTR bit.** 0_B Disable interrupt 1_B Enable interrupt
0	6:1, 31:10	r	**Reserved** Read as 0; should be written with 0.

19.4.1.71 PTM control

PTM_CTRL Offset address: $00003D0_H$

PTM control Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			DUPEN	DUPCLR	DUP	TRNEN	TRNCLR	TRN	UPDATED	UPDATING	TRGG	AUTOUPDT	MSTSTRB	RDYVLDT	CNTXTVLD
r			rw	w	rh	rw	w	rh	rh	rh	rh	rw	w	rh	rh

Field	Bits	Type	Description
CNTXTVLD	0	rh	**Context Valid.**
RDYVLDT	1	rh	**PTM Responder Ready to Validate.**
MSTSTRB	2	w	**PTM External Master Time Strobe.**
AUTOUPDT	3	rw	**Update the Context and Clock automatically** Configure the controller to update the PTM Requester Context and Clock automatically every 10ms.
TRGG	4	rh	**Indicates that a PTM Requester manual update trigger is allowed**
UPDATING	5	rh	**Indicates that a PTM update is in progress.**
UPDATED	6	rh	**Indicates that the controller has updated the Local Clock**

(table continues...)

(continued)

Field	Bits	Type	Description
TRN	7	rh	**PTM Requester Response Timeout status. The interrupt can be triggered when the TRNEN is asserted.** Indicating 100 us timeout occurred while waiting for a PTM Response or PTM ResponseD message.
TRNCLR	8	w	**Clear the TRN bit.** 0_B Keep the status 1_B Clear the interrupt status
TRNEN	9	rw	**Enable the TRN bit.** 0_B Disable interrupt 1_B Enable interrupt
DUP	10	rh	**PTM Requester Duplicate Received. The interrupt can be triggered when the DUPEN is asserted.** Single-cycle pulse indicating PTM Requester received a duplicate TLP while,
DUPCLR	11	w	**Clear the DUP bit.** 0_B Keep the status 1_B Clear the interrupt status
DUPEN	12	rw	**Enable the DUP bit.** 0_B Disable interrupt 1_B Enable interrupt
0	31:13	r	**Reserved** Read as 0; should be written with 0.

19.4.1.72 Local Clock z value

LCLCLKz (z=0-1) Offset address: $00003D4_H+z*4$

Local Clock z value Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LCLCLK															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LCLCLK															
rh															

Field	Bits	Type	Description
LCLCLK	31:0	rh	**Local Clock value.**

19.4.1.73 Clock z correction

CLKCORRz (z=0-1) Offset address: $00003DC_H+z*4$

Clock z correction Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CLKCORR															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CLKCORR															
rh															

Field	Bits	Type	Description
CLKCORR	31:0	rh	**Amount by which Local Clock has been corrected**

19.4.1.74 PTM z External Master Time

MSKCLKz (z=0-1) Offset address: $00003E4_H+z*4$

PTM z External Master Time Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MSKCLK															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MSKCLK															
rw															

Field	Bits	Type	Description
MSKCLK	31:0	rw	**PTM External Master Time.**

19.4.1.75 Hot-plug control

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												HPINTCLR	HPINTEN	HPINT	HPPME
r												w	rw	rh	rh

Field	Bits	Type	Description
HPPME	0	rh	**The controller asserts hp_pme when all of the following conditions are true:** • The PME Enable bit in the Power Management Control and Status register is set to 1. • Any bit in the Slot Status register transitions from 0 to 1 and the associated event notification is enabled in the Slot Control register. The controller does not check if the PM state is D1, D2, or D3hot. It is up to your application to check the value pm_dstate to make sure the device is in D1, D2, or D3hot. There is one bit of HPPME for each configured function. The controller pulses this bit output only when any hot plug status bit changes from 0 to 1 (as is HPINT). hp_int stays asserted as long as the status bit is set. In addition, it asserts HPPME only if PME is enabled, but it does not matter if hot-plug interrupts are enabled.
HPINT	1	rh	**The controller asserts hp_int when all of the following conditions are true:** • The INTx Assertion Disable bit in the Command register is 0. • Hot-Plug interrupts are enabled in the Slot Control register. • Any bit in the Slot Status register is equal to 1, and the associated event notification is enabled in the Slot Control register. Or • MSI or MSI-X is enabled. • Hot-Plug interrupts are enabled in the Slot Control register. • Any bit in the Slot Status register transitions from 0 to 1 and the associated event notification is enabled in the Slot Control register.
HPINTEN	2	rw	**Hot plug enable** 0_B Disable interrupt 1_B Enable interrupt
HPINTCLR	3	w	**Clear the HPINT bit.** 0_B Keep the status 1_B Clear the interrupt status
0	31:4	r	**Reserved** Read as 0; should be written with 0.

19.4.1.76 OBFF message control

OBFFMSG Offset address: $00003F0_H$

OBFF message control Kernel Reset value: $0000\ 0000_H$

19 Peripheral Component Interconnect Express (PCIe)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			ACTEN	ACTCLR	ACT	OBFFEN	OBFFCLR	OBFF	IDLEEN	IDLECLR	IDLE	ACTREQ	OBFFREQ	IDLEREQ	GRNT
r			rw	w	rh	rw	w	rh	rw	w	rh	w	w	w	rh

Field	Bits	Type	Description
GRNT	0	rh	**Request accepted to generate an OBFF message**
IDLEREQ	1	w	**To generate an 'IDLE' OBFF message.** Request from your application to generate an 'IDLE' OBFF message. Only usable in a downstream port.
OBFFREQ	2	w	**Request to generate an 'OBFF' OBFF message.** Request from your application to generate an 'OBFF' OBFF message. Only usable in a downstream port.
ACTREQ	3	w	**Request to generate a 'CPU Active' OBFF message.** Request from your application to generate a 'CPU Active' OBFF message. Only usable in a downstream port.
IDLE	4	rh	**The controller received an 'IDLE' OBFF message. The interrupt can be triggered when the IDLEEN is asserted.** Indicating that the controller received an 'IDLE' OBFF message. Only usable in an upstream port
IDLECLR	5	w	**Clear the IDLE bit** 0_B Keep the status 1_B Clear the interrupt status
IDLEEN	6	rw	**Enable the idle bit.** 0_B Disable interrupt 1_B Enable interrupt
OBFF	7	rh	**The controller received an 'OBFF' OBFF message. The interrupt can be triggered when the OBFFEN is asserted.** Indicating that the controller received an 'OBFF' OBFF message. Only usable in an upstream port.
OBFFCLR	8	w	**Clear the OBFFbit** 0_B Keep the status 1_B Clear the interrupt status
OBFFEN	9	rw	**Enable the OBFFbit.** 0_B Disable interrupt 1_B Enable interrupt
ACT	10	rh	**Received a 'CPU Active' OBFF message. The interrupt can be triggered when the ACTEN is asserted.** Indicating that the controller received a 'CPU Active' OBFF message. Only usable in an upstream port.

(table continues...)

(continued)

Field	Bits	Type	Description
ACTCLR	11	w	**Clear the ACT bit** 0_B Keep the status 1_B Clear the interrupt status
ACTEN	12	rw	**Enable the ACT bit.** 0_B Disable interrupt 1_B Enable interrupt
0	31:13	r	**Reserved** Read as 0; should be written with 0.

19.4.1.77 OBFF WAKE control

OBFFWAKE Offset address: 00003F4$_H$

OBFF WAKE control Kernel Reset value: 0000 0000$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0					EFSMSTAT				DFSMSTAT				INITSTAT		DEC RST
r					rh				rh				rw		w

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ERREN	ERRCLR	WAKERR	ACTEN	ACTCLR	ACT	OBFFEN	OBFFCLR	OBFF	IDLEEN	IDLECLR	IDLE	ACTREQ	OBFFREQ	IDLEREQ	GRNT
rw	w	rh	rw	w	rh	rw	w	rh	rw	w	rh	w	w	w	rh

Field	Bits	Type	Description
GRNT	0	rh	**Request received to generate a WAKE# signal.** Indicates that the controller has accepted your request to generate a WAKE# signal. Only usable in a downstream port
IDLEREQ	1	w	**Request to generate an 'IDLE' WAKE# signal** Request from your application to generate an 'IDLE' WAKE# signalOnly usable in a downstream port.
OBFFREQ	2	w	**Request to generate an 'OBFF' WAKE# signal.** Request from your application to generate an 'OBFF' WAKE# signal. Only usable in a downstream port.
ACTREQ	3	w	**Request to generate a 'CPU Active' WAKE# signal.** Request from your application to generate a 'CPU Active' WAKE# signal. Only usable in a downstream port.
IDLE	4	rh	**Received an 'IDLE' OBFF message. The interrupt can be triggered when the IDLEEN is asserted.** Indicating that the controller received an 'IDLE' OBFF message. Only usable in an upstream port

(table continues...)

(continued)

Field	Bits	Type	Description
IDLECLR	5	w	**Clear the IDLE bit** 0_B Keep the status 1_B Clear the interrupt status
IDLEEN	6	rw	**Enable the idle bit.** 0_B Disable interrupt 1_B Enable interrupt
OBFF	7	rh	**Received an 'OBFF' OBFF message. The interrupt can be triggered when the OBFFEN is asserted.** Indicating that the controller received an 'OBFF' OBFF message. Only usable in an upstream port.
OBFFCLR	8	w	**Clear the OBFFbit** 0_B Keep the status 1_B Clear the interrupt status
OBFFEN	9	rw	**Enable the OBFFbit.** 0_B Disable interrupt 1_B Enable interrupt
ACT	10	rh	**Received a 'CPU Active' OBFF message. The interrupt can be triggered when the ACTEN is asserted.** Indicating that the controller received a 'CPU Active' OBFF message. Only usable in an upstream port.
ACTCLR	11	w	**Clear the ACT bit** 0_B Keep the status 1_B Clear the interrupt status
ACTEN	12	rw	**Enable the ACT bit.** 0_B Disable interrupt 1_B Enable interrupt
WAKERR	13	rh	**The WAKE# signal decoder has received an invalid WAKE# pattern. The interrupt can be triggered when the ERREN is asserted.**
ERRCLR	14	w	**Clear the WAKERR bit** 0_B Keep the status 1_B Clear the interrupt status
ERREN	15	rw	**Enable the WAKERR bit.** 0_B Disable interrupt 1_B Enable interrupt
DECRST	16	w	**Reset the OBFF WAKE# signaling decoder.**
INITSTAT	18:17	rw	**The intial state after the OBFF WAKE# signaling decoder is reset by teh DECRST.** 00_B "IDLE" 01_B "OBFF" 10_B "CPU active" 11_B Reserved

(table continues...)

(continued)

Field	Bits	Type	Description
DFSMSTAT	22:19	rh	**Current decoder FSM state for debug. Not equal to OBFF code.**
EFSMSTAT	26:23	rh	**Current encoder FSM state for debug. Not equal to OBFF code.**
0	31:27	r	**Reserved** Read as 0; should be written with 0.

19.4.1.78 PCIe configuration control

app_dev_num_i and app_bus_num_i for ATU MEM to CFG translation

CFGBD Offset address: $00003F8_H$

PCIe configuration control Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			DEVNUM					BUSNUM							
r			rw					rw							

Field	Bits	Type	Description
BUSNUM	7:0	rw	**Bus number** Used to indicate the requester bus number information when a RC sends out CFG TLPs.
DEVNUM	12:8	rw	**Device number** Used to indicate the requester device number information when a RC sends out CFG TLPs.
0	31:13	r	**Reserved** Read as 0; should be written with 0.

19.4.1.79 ELBI x mailbox register

DATAx (x=0-7) Offset address: $00003FC_H+x*4$

ELBI x mailbox register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DATA															
rwh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA															
rwh															

Field	Bits	Type	Description
DATA	31:0	rwh	**ELBI mailbox registers.** These registers can be used as buffers to communicate information for debugging purpose.

19.4.1.80 DMA link list control register

LLCTRL Offset address: 000041C$_H$

DMA link list control register Kernel Reset value: 0000 0000$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

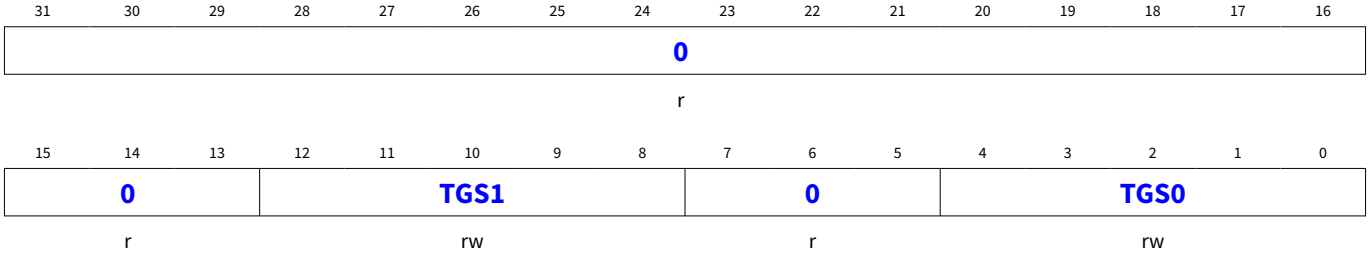
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RDDONE3	RDDONE2	RDDONE1	RDDONE0	WRDONE3	WRDONE2	WRDONE1	WRDONE0	RDTOGG3	RDTOGG2	RDTOGG1	RDTOGG0	WRTOGG3	WRTOGG2	WRTOGG1	WRTOGG0
rh	rh	rh	rh	rh	rh	rh	rh	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
WRTOGGz (z=0-3)	z	rw	**DMA Write engine descriptor transfer go toggle** DMA Write engine descriptor transfer go toggle signal. This signal is ignored whenever a DMA channel operates in non-linked mode.
RDTOGGz (z=0-3)	z+4	rw	**DMA Read engine descriptor transfer go toggle** DMA Read engine descriptor transfer go toggle signal. This signal is ignored whenever a DMA channel operates in non-linked mode.
WRDONEz (z=0-3)	z+8	rh	**DMA write engine descriptor transfer done toggle** DMA write engine descriptor transfer done toggle signal. After a doorbell command, this signal is de-asserted. When a channel is operating in non-linked list mode this signal does not toggles
RDDONEz (z=0-3)	z+12	rh	**DMA write engine descriptor transfer done toggle** DMA read engine descriptor transfer done toggle signal. After a doorbell command, this signal is de-asserted. When a channel is operating in non-linked list mode this signal does not toggles
0	31:16	r	**Reserved** Read as 0; should be written with 0.

19.4.1.81 PCIe OCDS Trigger Set Select

OTSS Offset address: 0000420$_H$

PCIe OCDS Trigger Set Select Debug Reset value: 0000 0000$_H$

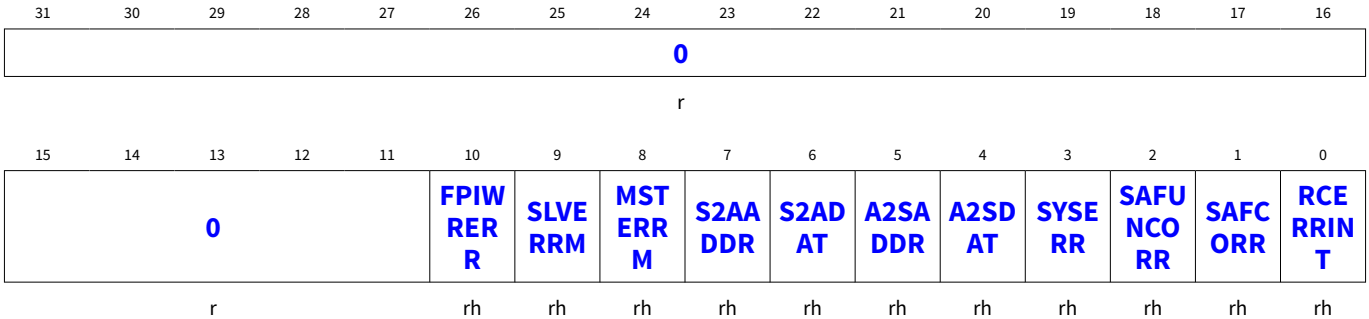


Field	Bits	Type	Description
TGS0	4:0	rw	**Trigger set select 0** 00_H No Trigger Set output 01_H Trigger Set 0 Trigger Set TS16_PCIe0 02_H Trigger Set 1 Trigger Set TS16_PCIe1 03_H Trigger Set 2 Trigger Set TS16_PCIe2 04_H Reserved
TGS1	12:8	rw	**Trigger Set for OTGB1** 00_H No Trigger Set output 01_H Trigger Set 0 Trigger Set TS16_PCIe0 02_H Trigger Set 1 Trigger Set TS16_PCIe1 03_H Trigger Set 2 Trigger Set TS16_PCIe2 04_H Reserved
0	7:5, 31:13	r	**Reserved** Read as 0; should be written with 0.

19.4.1.82 Interrupt/alarm status

ALARM_STAT Offset address: 0000424_H

Interrupt/alarm status Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
RCERRINT	0	rh	**Error reported. The interrupt can be triggered when the ALARM_INTEN.RCERREN is asserted.** Asserted when a reported error condition causes a bit to be set in the Root Error Status register and the associated error message reporting enable bit is set in the Root Error Command register. This bit is set when the RC internally generates an error or when an error message is received by the RC. Because the RC itself generates it, this needs to be propagated up to the system software which would then need to read the error registers to see which error occurred.
SAFCORR	1	rh	**Indicate correctable error. The interrupt can be triggered when the ALARM_INTEN.SAFCORREN is asserted.** The controller asserts this bit when a PCIe protocol correctable error or a Data Path protection correctable error is indicated and the associated mask for that event is not set.
SAFUNCORR	2	rh	**Indicate uncorrectable error. The interrupt can be triggered when the ALARM_INTEN.SAFUNCORR is asserted.** The controller asserts this bit when any of the internal safety mechanisms reports an uncorrectable error and the associated mask for that event is not set.
SYSERR	3	rh	**System error detected. The interrupt can be triggered when the ALARM_INTEN.SYSERREN is asserted.** Assertion of this bit indicates if any device in the hierarchy reports any of the following errors and the associated enable bit is set in the Root Control register: ERR_COR, ERR_FATAL, ERR_NONFATAL. Also asserted when an internal error is detected.
A2SDAT	4	rh	**Error detected during the data phase in the A2S bridge. The interrupt can be triggered when the ALARM_INTEN.A2SDEN is asserted.**
A2SADDR	5	rh	**Error detected during the address phase in the A2S bridge. The interrupt can be triggered when the ALARM_INTEN.A2SAEN is asserted.**
S2ADAT	6	rh	**Error detected during the data phase in the S2A bridge. The interrupt can be triggered when the ALARM_INTEN.S2ADEN is asserted.**
S2AADDR	7	rh	**Error detected during the address phase in the S2A bridge. The interrupt can be triggered when the ALARM_INTEN.S2AAEN is asserted.**
MSTERRM	8	rh	**The controller has entered RASDP error mode at the master interface of AXI.** The controller enters RASDP error mode (if the ERROR_MODE_EN register field =1) upon detection of the first uncorrectable error. During this mode: • Controller sets SLVERRM=1 • Controller sets MSTERRM=1 • Rx TLPs that are forwarded to your application are not guaranteed to be correct; you must discard them.

(table continues...)

(continued)

Field	Bits	Type	Description
SLVERRM	9	rh	**The controller has entered RASDP error mode at the slave interface of AXI.** The controller enters RASDP error mode (if the ERROR_MODE_EN register field =1) upon detection of the first uncorrectable error. During this mode: • Controller sets SLVERRM=1 • Controller sets MSTERRM=1 • Rx TLPs that are forwarded to your application are not guaranteed to be correct; you must discard them.
FPIWRERR	10	rh	**FPI write error status** Status to indicate write returned an error on the FPI interface
0	31:11	r	**Reserved** Read as 0; should be written with 0.

19.4.1.83 Interrupt/alarm clear

ALARM_INTCLR Offset address: 0000428_H

Interrupt/alarm clear Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res								0							
r								r							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0					FPIWRERRCLR	0		S2AACLR	S2ADCLR	A2SACLR	A2SDCLR	SYSERRCLR	SAFUNCORRCLR	SAFTCORRCLR	RCERRCLR
r					w	r		w	w	w	w	w	w	w	w

Field	Bits	Type	Description
RCERRCLR	0	w	**Clear the RCERRINT bit.** 0_B Keep the status 1_B Clear the interrupt status
SAFTCORRCLR	1	w	**Clear the SAFCORR bit.** 0_B Keep the status 1_B Clear the interrupt status
SAFUNCORRCLR	2	w	**Clear the SAFUNCORR bit.** 0_B Keep the status 1_B Clear the interrupt status
SYSERRCLR	3	w	**Clear the SYSERR bit.** 0_B Keep the status 1_B Clear the interrupt status

(table continues...)

(continued)

Field	Bits	Type	Description
A2SDCLR	4	w	**Clear the A2SDAT bit.** 0_B Keep the status 1_B Clear the interrupt status
A2SACLR	5	w	**Clear the A2SADDR bit.** 0_B Keep the status 1_B Clear the interrupt status
S2ADCLR	6	w	**Clear the S2ADAT bit.** 0_B Keep the status 1_B Clear the interrupt status
S2AACLR	7	w	**Clear the S2AADDR bit.** 0_B Keep the status 1_B Clear the interrupt status
FPIWRERRCLR	10	w	**clear FPIWRERR** clear FPIWRERR 0_B Keep the status 1_B Clear the interrupt status
0	9:8, 30:11	r	**Reserved** Read as 0; should be written with 0.

19.4.1.84 Interrupt/alarm enable

ALARM_INTEN Offset address: $000042C_H$

Interrupt/alarm enable Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res								0							
r								r							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0					FPIWRERREN	0		S2AAEN	S2ADEN	A2SAEN	A2SDEN	SYSERREN	SAFUNCORREN	SAFCORREN	RCERREN
r					rw	r		rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RCERREN	0	rw	**Enable the RCERRINT bit.** 0_B Disable interrupt 1_B Enable interrupt
SAFCORREN	1	rw	**Enable the SAFCORR bit.** 0_B Disable interrupt 1_B Enable interrupt

(table continues...)

(continued)

Field	Bits	Type	Description
SAFUNCORREN	2	rw	**Enable the SAFUNCORR bit.** 0_B Disable interrupt 1_B Enable interrupt
SYSERREN	3	rw	**Enable the SYSERR bit.** 0_B Disable interrupt 1_B Enable interrupt
A2SDEN	4	rw	**Enable the A2SDAT bit to generate alarm.** 0_B Disable interrupt 1_B Enable interrupt
A2SAEN	5	rw	**Enable the A2SADDR bit to generate alarm.** 0_B Disable interrupt 1_B Enable interrupt
S2ADEN	6	rw	**Enable the S2ADAT bit to generate alarm.** 0_B Disable interrupt 1_B Enable interrupt
S2AAEN	7	rw	**Enable the S2AADDR bit to generate alarm.** 0_B Disable interrupt 1_B Enable interrupt
FPIWRERREN	10	rw	**Enable FPIWRERR bit to generate alarm.** Enable FPIWRERR bit to generate alarm 0_B Disable 1_B Enable
0	9:8, 30:11	r	**Reserved** Read as 0; should be written with 0.

19.4.1.85 iATU Region i Control 1 Register Outbound

This register controls the iATU outbound region access based on the optional iATU outbound features enabled using iATU Region Control 2 Register.

IATU_REGION_CTRL_1_OFF_OUTBOUND_i (i=0-7) Offset address: $0003000_H+i*200_H$

iATU Region i Control 1 Register Outbound Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0									CTRL_1_FUNC_NUM			0			
r									rw			r			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		Res	0		ATTR		TD	TC			TYPE				
r		r	r		rw		rw	rw			rw				

Field	Bits	Type	Description
TYPE	4:0	rw	**When the address of an outbound TLP is matched to this region, then the TYPE field of the TLP is changed to the value in this register** Note: This register field is sticky.
TC	7:5	rw	**When the address of an outbound TLP is matched to this region, then the TC field of the TLP is changed to the value in this register** Note: This register field is sticky.
TD	8	rw	**This is a reserved field** Do not use. Note: This register field is sticky.
ATTR	10:9	rw	**When the address of an outbound TLP is matched to this region, then the ATTR field of the TLP is changed to the value in this register** Note: This register field is sticky.
CTRL_1_FUNC_NUM	22:20	rw	**Function Number** The value in this register must be 0x0 as only one function is supported by this controller. . When the address of an outbound TLP is matched to this region and the FUNC_BYPASS field in the "iATU Region Control 2 Register" is '0', then the function number used in generating the function part of the requester ID (RID) field of the TLP is taken from this 5-bit register. This field is swapped before AXI decomposition occurs so that the correct "Max_Read_Request_Size" and "Max_Payload_Size" values are used. Note: This register field is sticky
0	12:11, 19:14, 31:23	r	**Reserved** Read as all 0's; should be written with all 0's.

19.4.1.86 iATU Region i Control 2 Register Outbound

Using this register you can enable/disable the outbound iATU optional features.

IATU_REGION_CTRL_2_OFF_OUTBOUND_i (i=0-7) Offset address: $0003004_H+i*200_H$

iATU Region i Control 2 Register Outbound Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
REGION_EN	0	INVERT_MODE	Res	DMA_BYPASS	0			HEADER_SUBSTITUTE_EN	INHIBIT_PAYLOAD	Res		FUNC_BYPASS	0		TAG_SUBSTITUTE_EN
rw	r	rw	r	rw	r			rw	rw	r		rw	r		rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAG								MSG_CODE							
rw								rw							

Field	Bits	Type	Description
MSG_CODE	7:0	rw	**MSG TLPs (Message Code)** When the address of an outbound TLP is matched to this region, and the translated TLP TYPE field is Msg or MsgD; then the message field of the TLP is changed to the value in this register. Note: This register field is sticky.
TAG	15:8	rw	**TAG** The substituted TAG field (byte 6) in the outgoing TLP header when TAG_SUBSTITUTE_EN is set. Note: This register field is sticky.
TAG_SUBSTITUTE_EN	16	rw	**TAG Substitute Enable** When enabled and region address is matched, the iATU substitutes the TAG field of the outbound TLP header with the contents of the TAG field in this register. Note: This register field is sticky. 0_B **DISABLE**: Disable 1_B **ENABLE**: Enable
FUNC_BYPASS	19	rw	**Function Number Translation Bypass** In this mode, the function number of the translated TLP is taken from your application transmit interface and not from the CTRL_1_FUNC_NUM field of the "iATU Region Control 1 Register" Note: This register field is sticky. 0_B **DISABLE**: Disable 1_B **ENABLE**: Enable
INHIBIT_PAYLOAD	22	rw	**Inhibit TLP Payload Data for TLP's in Matched Region; assign iATU region to be TLP without data** When enabled and region address is matched, the iATU marks all TLPs as having no payload data by forcing the TLP header Fmt[1] bit =0b. Encoding are define as above. Note: This register field is sticky. 0_B **SUPPORT_DATA_TLP**: Fmt[1] =0/1 so that TLPs with or without data can be sent 1_B **NO_SUPPORT_FOR_DATA_TLP**: Fmt[1] =0 so that only TLP type without data is sent For example, a Msg instead of MsgD will be sent.
HEADER_SUBSTITUTE_EN	23	rw	**Header Substitute Enable** When enabled and region address is matched, the iATU fully substitutes bytes 8-11 (for 3 DWORD header) or bytes 12-15 (for 4 DWORD header) of the outbound TLP header with the contents of the LWR_TARGET_RW field in IATU_LWR_TARGET_ADDR_OFF_OUTBOUND_i. Encodings are as above. Note: This register field is sticky. 0_B **FORM_ADDR**: LWR_TARGET_RW in the iATU_LWR_TARGET_ADDR_OFF_OUTBOUND_i register forms the new address of the translated region 1_B **FILL_BYTES**: LWR_TARGET_RW in the iATU_LWR_TARGET_ADDR_OFF_OUTBOUND_i register is used to fill bytes 8-to-11 (for 3 DWORD header) or bytes 12-to-15 (for 4 DWORD header) of the translated TLP header

(table continues...)

(continued)

Field	Bits	Type	Description
DMA_BYPASS	27	rw	**DMA Bypass Mode** Allows request TLPs which are initiated by the DMA controller to pass through the iATU untranslated. Note: This register field is sticky. 0_B **DISABLE**: Disable 1_B **ENABLE**: Enable
INVERT_MODE	29	rw	**Invert Mode** When set the address matching region is inverted. Therefore, an address match occurs when the untranslated address is in the region outside the defined range (Base Address to Limit Address). Note: This register field is sticky. 0_B **DISABLE**: Disable 1_B **ENABLE**: Enable
REGION_EN	31	rw	**Region Enable** This bit must be set to '1' for address translation to take place. Note: This register field is sticky.
0	18:17, 26:24, 30	r	**Reserved** Read as all 0's; should be written with all 0's.

19.4.1.87 iATU i Lower Base Address Register Outbound

.

IATU_LWR_BASE_ADDR_OFF_OUTBOUND_i (i=0-7) Offset address: $0003008_H+i*200_H$

iATU i Lower Base Address Register Outbound Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LWR_BASE_RW															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LWR_BASE_RW				LWR_BASE_HW											
rw				r											

Field	Bits	Type	Description
LWR_BASE_HW	11:0	r	**Forms bits of the start address of the address region to be translated** The start address must be aligned to a 4 kB boundary, so these bits are always 0. A write to this location is ignored by the PCIe controller.
LWR_BASE_RW	31:12	rw	**Forms bits of the start address of the address region to be translated** Note: This register field is sticky.

19.4.1.88 iATU i Upper Base Address Register Outbound

This register holds the upper 32-bits of the start (and end) address of the address region to be translated.

IATU_UPPER_BASE_ADDR_OFF_OUTBOUND_i (i=0-7) Offset address: $000300C_H+i*200_H$

iATU i Upper Base Address Register Outbound Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UPPER_BASE_RW															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UPPER_BASE_RW															
rw															

Field	Bits	Type	Description
UPPER_BASE_RW	31:0	rw	**Forms bits [63:32] of the start (and end) address of the address region to be translated** In systems with a 32-bit address space, this register is not used and therefore writing to this register has no effect. Note: This register field is sticky.

19.4.1.89 iATU i Limit Address Register Outbound

This register holds the end address of the address region to be translated.

IATU_LIMIT_ADDR_OFF_OUTBOUND_i (i=0-7) Offset address: $0003010_H+i*200_H$

iATU i Limit Address Register Outbound Kernel Reset value: $0000\ 0FFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LIMIT_ADDR_RW															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LIMIT_ADDR_RW				LIMIT_ADDR_HW								CBUF_INCR			
rw				r								r			

Field	Bits	Type	Description
CBUF_INCR	3:0	r	**Circular Buffer** Note: This register field is sticky.
LIMIT_ADDR_HW	11:4	r	**Forms lower bits of the end address of the address region to be translated** The end address must be aligned to a 4 kB boundary, so these bits are always all ones. A write to this location is ignored by the PCIe controller. Note: This register field is sticky.
LIMIT_ADDR_RW	31:12	rw	**Forms upper bits of the end address of the address region to be translated** Note: This register field is sticky.

19.4.1.90 iATU i Lower Target Address Register Outbound

This register holds the Lower Target part of the new address of the translated region, or the outbound TLP header information, depending on the setting of HEADER_SUBSTITUTE_EN field of IATU_REGION_CTRL_2_VIEWPORT_OFF_OUTBOUND_i register.

IATU_LWR_TARGET_ADDR_OFF_OUTBOUND_i (i=0-7) Offset address: $0003014_H+i*200_H$

iATU i Lower Target Address Register Outbound Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LWR_TARGET_RW_OUTBOUND															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LWR_TARGET_RW_OUTBOUND															
rw															

Field	Bits	Type	Description
LWR_TARGET_RW_OUTBOUND	31:0	rw	**Containing the Lower Target part of the new address of the translated region, or the outbound TLP header information** When HEADER_SUBSTITUTE_EN is 0, for lower target address, the start address must be aligned to a 4 kB boundary, so the lower bits of the start address of the new address of the translated region (bits 11:0) are always '0'. When HEADER_SUBSTITUTE_EN in IATU_REGION_CTRL_2_OFF_OUTBOUND_i is '1': - this field forms bytes 8-11 (for 3 dword header) or bytes 12-15 (for 4 dword header) of the outbound TLP header. Note: This register field is sticky.

19.4.1.91 iATU i Upper Target Address Register Outbound

This register holds the upper 32 bits of the start address (Upper Target part) of the new address of the translated region.

IATU_UPPER_TARGET_ADDR_OFF_OUTBOUND_i (i=0-7) Offset address: $0003018_H+i*200_H$

iATU i Upper Target Address Register Outbound Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UPPER_TARGET_RW															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UPPER_TARGET_RW															
rw															

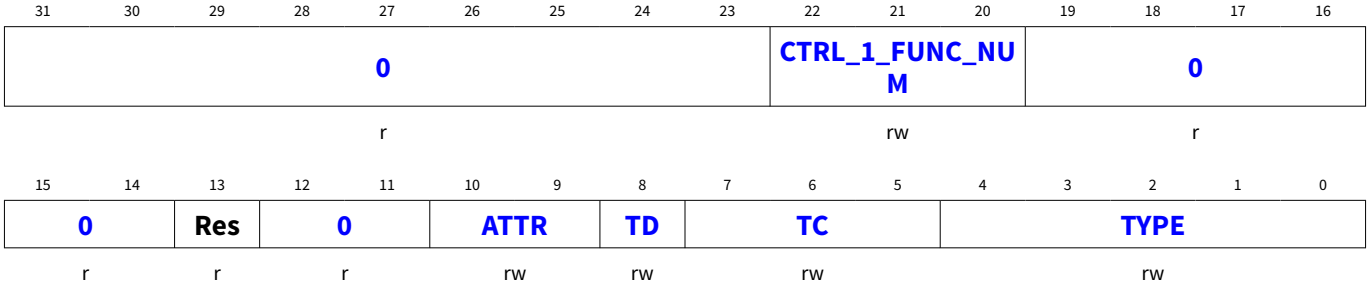
Field	Bits	Type	Description
UPPER_TARGET_RW	31:0	rw	**Forms bits [63:32] of the start address (Upper Target part) of the new address of the translated region** Note: This register field is sticky.

19.4.1.92 iATU Region i Control 1 Register Inbound

This register controls the iATU inbound region access based on the optional iATU inbound features enabled using iATU Region Control 2 Register.

IATU_REGION_CTRL_1_OFF_INBOUND_i (i=0-7) Offset address: $0003100_H+i*200_H$

iATU Region i Control 1 Register Inbound Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
TYPE	4:0	rw	**When the TYPE field of an inbound TLP is matched to this value, then address translation proceeds (when all other enabled field-matches are successful)** Note: This register field is sticky.
TC	7:5	rw	**When the TC field of an inbound TLP is matched to this value, then address translation proceeds (when all other enabled field-matches are successful)** This check is only performed if the "TC Match Enable" bit of the "iATU Region Control 2 Register" is set. Note: This register field is sticky.
TD	8	rw	**When the TD field of an inbound TLP is matched to this value, then address translation proceeds (when all other enabled field-matches are successful)** This check is only performed if the "TD Match Enable" bit of the "iATU Region Control 2 Register" is set. Note: This register field is sticky.
ATTR	10:9	rw	**When the ATTR field of an inbound TLP is matched to this value, then address translation proceeds (when all other enabled field-matches are successful)** This check is only performed if the "ATTR Match Enable" bit of the "iATU Region Control 2 Register" is set. Note: This register field is sticky.
CTRL_1_FUNC_NUM	22:20	rw	**Function Number** - MEM-I/O: When the Address and BAR matching logic in the controller indicate that a MEM-I/O transaction matches a BAR in the function corresponding to this value, then address translation proceeds. This check is only performed if the "Function Number Match Enable" bit of the "iATU Region Control 2 Register" is set. - CFG0/CFG1: When the destination function number as specified in the routing ID of the TLP header matches the function, then address translation proceeds. This check is only performed if the "Function Number Match Enable" bit of the "iATU Region Control 2 Register" is set. Note: This register field is sticky.

(table continues...)

(continued)

Field	Bits	Type	Description
0	12:11, 19:14, 31:23	r	**Reserved** Read as all 0's; should be written with all 0's.

19.4.1.93 iATU Region i Control 2 Register Inbound

Using this register you can enable/disable the inbound iATU optional features.

IATU_REGION_CTRL_2_OFF_INBOUND_i (i=0-7) Offset address: $0003104_H+i*200_H$

iATU Region i Control 2 Register Inbound Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
REGION_EN	MATCH_MODE	INVERT_MODE	Res		0	RESPONSE_CODE		SINGLE_ADDR_LOC_TRANS_EN	0	MSG_CODE_MATCH_EN	0	FUNC_NUM_MATCH_EN	0		ATTR_MATCH_EN
rw	rw	rw	r		r	rw		rw	r	rw	r	rw	r		rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TD_MATCH_EN	TC_MATCH_EN	MSG_TYPE_MATCH_MODE	0		BAR_NUM			MSG_CODE							
rw	rw	rw	r		rw			rw							

Field	Bits	Type	Description
MSG_CODE	7:0	rw	**MSG TLPs: (Message Code)** When the TYPE field of an inbound Msg/MsgD TLP is matched to this value, then address translation proceeds (when all other enabled field-matches are successful). This check is only performed if the "Message Code Match Enable" bit of the "iATU Region Control 2 Register" is set. Note: This register field is sticky.

(table continues...)

(continued)

Field	Bits	Type	Description
BAR_NUM	10:8	rw	**BAR Number** When the BAR number of an inbound MEM or IO TLP " that is matched by the normal internal BAR address matching mechanism " is the same as this field, address translation proceeds (when all other enabled field-matches are successful). This check is only performed if the "Match Mode" bit of the "iATU Region Control 2 Register" is set. IO translation would require either 00100b or 00101b in the inbound TLP TYPE; the BAR Number set in the range 000b - 101b and that BAR configured as an IO BAR. Note: This register field is sticky. 000_B **BAR0**: BAR0 001_B **BAR1**: BAR1 010_B **BAR2**: BAR2 011_B **BAR3**: BAR3 100_B **BAR4**: BAR4 101_B **BAR5**: BAR5 110_B **ROM**: ROM 111_B **RSVD**: reserved
MSG_TYPE_MATCH_MODE	13	rw	**Message Type Match Mode** When enabled, and if single address location translate enable is set, then inbound TLPs of type MSG/MSGd which match the type field of the IATU_REGION_CTRL_1_OFF_INBOUND_i register (TYPE[4:3]=2'b10) will be translated. Message type match mode overrides any value of MATCH_MODE field in this register. Usage scenarios for this are translation of VDM or ATS messages when AXI bridge is configured on client interface. Note: This register field is sticky. 0_B **DISABLE**: Disable 1_B **ENABLE**: Enable
TC_MATCH_EN	14	rw	**TC Match Enable** Ensures that a successful TC TLP field comparison match (see TC field of the "iATU Region Control 1 Register") occurs for address translation to proceed. Note: This register field is sticky. 0_B **DISABLE**: TC Match Disable 1_B **ENABLE**: TC Match Enable
TD_MATCH_EN	15	rw	**TD Match Enable** Ensures that a successful TD TLP field comparison match (see TD field of the "iATU Region Control 1 Register") occurs for address translation to proceed. Note: This register field is sticky. 0_B **DISABLE**: TD Match Disable 1_B **ENABLE**: TD Match Enable
ATTR_MATCH_EN	16	rw	**ATTR Match Enable** Ensures that a successful ATTR TLP field comparison match (see ATTR field of the "iATU Region Control 1 Register") occurs for address translation to proceed. Note: This register field is sticky. 0_B **DISABLE**: ATTR Match Disable 1_B **ENABLE**: ATTR Match Enable

(table continues...)

(continued)

Field	Bits	Type	Description
FUNC_NUM_MATCH_EN	19	rw	**Function Number Match Enable** Ensures that a successful Function Number TLP field comparison match (see Function Number field of the "iATU Region Control 1 Register") occurs (in MEM-I/O and CFG0/CFG1 transactions) for address translation to proceed. Note: This register field is sticky. 0_B **DISABLE**: Function Number Match Disable 1_B **ENABLE**: Function Number Match Enable
MSG_CODE_MATCH_EN	21	rw	**Message Code Match Enable (Msg TLPS)** Ensures that a successful message Code TLP field comparison match (see Message Code field of the "iATU Region Control 2 Register") occurs (in MSG transactions) for address translation to proceed. Note: This register field is sticky. 0_B **DISABLE**: Virtual Function Number Match Disable 1_B **ENABLE**: Message Code Match Enable (for Msg TLPS) or ST Match Enable (for Mem TLPs)
SINGLE_ADDR_LOC_TRANS_EN	23	rw	**Single Address Location Translate Enable** When enabled, Rx TLPs can be translated to a single address location as determined by the target address register of the iATU region. The main usage scenario is translation of Messages to MWr TLPs when the AXI bridge is enabled. Note: This register field is sticky. 0_B **DISABLE**: Disable 1_B **ENABLE**: Enable
RESPONSE_CODE	25:24	rw	**Response Code** Defines the type of response to give for accesses matching this region. This overrides the normal RADM filter response. Note that this feature is not available for any region where Single Address Location Translate is enabled. Note: This register field is sticky. 00_B **NORMAL_RADM**: Normal RADM filter response is used 01_B **UNSUP_REQ**: Unsupported request (UR) 10_B **COMPL_ABORT**: Completer abort (CA) 11_B **NOT_USED**: Not used / undefined / reserved
INVERT_MODE	29	rw	**Invert Mode Enable** When set the address matching region is inverted. Therefore, an address match occurs when the untranslated address is in the region outside the defined range (Base Address to Limit Address). When set all regions of that type must use address match mode. Note: This register field is sticky. 0_B **DISABLE**: Invert Mode Disable 1_B **ENABLE**: Invert Mode Enable

(table continues...)

(continued)

Field	Bits	Type	Description
MATCH_MODE	30	rw	**Match Mode** Determines Inbound matching mode for TLPs. The mode depends on the type of TLP that is received as follows: For MEM-I/O TLPs, this field is interpreted as follows: • 0: Address Match Mode. The iATU operates using addresses as in the outbound direction. The Region Base and Limit Registers must be setup. • 1: BAR Match Mode. BAR matching is used. The "BAR Number" field is relevant. Not used for RC. For CFG0 TLPs, this field is interpreted as follows: • 0: Routing ID Match Mode. The iATU interprets the Routing ID (Bytes 8 to 11 of TLP header) as an address. This corresponds to the upper 16 bits of the address in MEM-I/O transactions. The Routing ID of the TLP must be within the base and limit of the iATU region for matching to proceed • 1: Accept Mode. The iATU accepts all CFG0 transactions as address matches. The routing ID in the CFG0 TLP is ignored. This is useful as all received CFG0 TLPs should be processed regardless of the Bus number For MSG/MSGD TLPs, this field is interpreted as follows: • 0: Address Match Mode. The iATU treats the third dword and fourth dword of the inbound MSG/MSGD TLP as an address and it is matched against the Region Base and Limit Registers • 1: Vendor ID Match Mode. The iATU ignores the Routing ID (Bus, Device, Function) in bits [31:16] of the third dword of the TLP header, but matches against the Vendor ID in bits [15:0] of the third dword of the TLP header. Bits [15:0] of the Region Upper Base register should be programmed with the required Vendor ID. The lower Base and Limit Register should be programmed to translate TLPs based on vendor specific information in the fourth dword of the TLP header If SINGLE_ADDRESS_LOCATION_TRANSLATE_EN = 1 AND MSG_TYPE_MATCH_MODE =1, then Match Mode is ignored. Note: This register field is sticky. 0_B **ZERO**: The interpretation is dependent on TLP type, that is, MEM/IO, CFG0, or MSG/MSGD TLPs 1_B **ONE**: The interpretation is dependent on TLP type, that is, MEM/IO, CFG0, or MSG/MSGD TLPs
REGION_EN	31	rw	**Region Enable** This bit must be set to '1' for address translation to take place. Note: This register field is sticky. 0_B **DISABLE**: Disable 1_B **ENABLE**: Enable

(table continues...)

(continued)

Field	Bits	Type	Description
0	12:11, 18:17, 20, 22, 26	r	**Reserved** Read as all 0's; should be written with all 0's.

19.4.1.94 iATU i Lower Base Address Register Inbound

IATU_LWR_BASE_ADDR_OFF_INBOUND_i (i=0-7) Offset address: $0003108_H+i*200_H$

iATU i Lower Base Address Register Inbound Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LWR_BASE_RW															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LWR_BASE_RW				LWR_BASE_HW											
rw				r											

Field	Bits	Type	Description
LWR_BASE_HW	11:0	r	**Forms bits of the start address of the address region to be translated** The start address must be aligned to a 4 kB boundary, so these bits are always 0. A write to this location is ignored by the PCIe controller.
LWR_BASE_RW	31:12	rw	**Forms bits of the start address of the address region to be translated** Note: This register field is sticky.

19.4.1.95 iATU i Upper Base Address Register Inbound

This register holds the upper 32 bits of the start (and end) address of the address region to be translated.

IATU_UPPER_BASE_ADDR_OFF_INBOUND_i (i=0-7) Offset address: $000310C_H+i*200_H$

iATU i Upper Base Address Register Inbound Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UPPER_BASE_RW															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UPPER_BASE_RW															
rw															

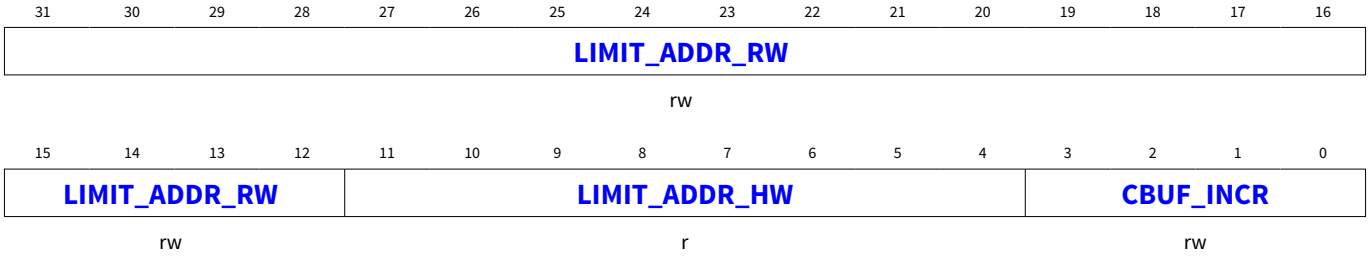
Field	Bits	Type	Description
UPPER_BASE_RW	31:0	rw	**Forms bits [63:32] of the start (and end) address of the address region to be translated** Note: This register field is sticky.

19.4.1.96 iATU i Limit Address Register Inbound

This register holds the end address of the address region to be translated.

IATU_LIMIT_ADDR_OFF_INBOUND_i (i=0-7) Offset address: $0003110_H+i*200_H$

iATU i Limit Address Register Inbound Kernel Reset value: $0000\ 0FF0_H$



Field	Bits	Type	Description
CBUF_INCR	3:0	rw	**Circular Buffer Increment** This field is R/W and forms the upper bits of the Circular Buffer Increment size (CBUF_INCR) field for Single Location Address translation. The increment value (in bytes) is decoded as follows: • 0000b: 0 (Default; legacy Single Address Location mode) • 0001b: 4 • 0010b: 8 • 0011b: 16 • 0100b: 32 • 0101b: 64 • 0110b: 128 • 0111b: 256 • 1000b: 512 • 1001b: 1024 • 1010b: 2048 • 1011b: 4096 • 1100b: 8192 • 1101b: rsvd • 1110b: rsvd • 1111b: rsvd Note: A write to any bit in the CBUF_INCR field resets the circular buffer pointer. This field must be written to AFTER the target and limit registers have been updated. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: R (sticky) Note: This register field is sticky.

(table continues...)

(continued)

Field	Bits	Type	Description
LIMIT_ADDR_HW	11:4	r	**Forms lower bits of the end address of the address region to be translated** The end address must be aligned to a 4 kB boundary, so these bits are always all ones. A write to this location is ignored by the PCIe controller. Note: This register field is sticky.
LIMIT_ADDR_RW	31:12	rw	**Forms upper bits of the end address of the address region to be translated** Note: This register field is sticky.

19.4.1.97 iATU i Lower Target Address Register Inbound

This register holds the Lower Target part of the new address of the translated region.

IATU_LWR_TARGET_ADDR_OFF_INBOUND_i (i=0-7) Offset address: $0003114_H+i*200_H$

iATU i Lower Target Address Register Inbound Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LWR_TARGET_RW															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LWR_TARGET_RW				LWR_TARGET_HW											
rw				r											

Field	Bits	Type	Description
LWR_TARGET_HW	11:0	r	**Forms the LSB's of the Lower Target part of the new address of the translated region** The start address must be aligned to a 4 kB boundary (in address match mode); and to the Bar size boundary (in BAR match mode) so that these bits are always '0'. If the BAR is smaller than the iATU region size, then the iATU target address must align to the iATU region size; otherwise it must align to the BAR size. A write to this location is ignored by the PCIe controller. .
LWR_TARGET_RW	31:12	rw	**Forms MSB's of the Lower Target part of the new address of the translated region** These bits are always '0'. Note: This register field is sticky.

19.4.1.98 Device ID and Vendor ID Register

This register holds the device ID and vendor ID.

DEVICE_ID_VENDOR_ID_REG Offset address: 0004000_H

Device ID and Vendor ID Register Kernel Reset value: $ABCD\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PCI_TYPE0_DEVICE_ID															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PCI_TYPE0_VENDOR_ID															
rw															

Field	Bits	Type	Description
PCI_TYPE0_VENDOR_ID	15:0	rw	**Vendor ID** The Vendor ID register identifies the manufacturer of the Function. Valid vendor identifiers are allocated by the PCI-SIG to ensure uniqueness. It is not permitted to keep this register populated with a value of FFFFh, which is an invalid value for Vendor ID. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky.
PCI_TYPE0_DEVICE_ID	31:16	rw	**Device ID** The Device ID register identifies the particular Function. This identifier is allocated by the vendor. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky.

19.4.1.99 Status and Command Register

This register provides the status and controls the behavior of a function.

STATUS_COMMAND_REG Offset address: 0004004_H

Status and Command Register Reset values see: Table 865

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DETECTED_PARITY_ERR	SIGNALED_SYS_ERR	RCVD_MASTER_ABORT	RCVD_TARGET_ABORT	SIGNALED_TARGET_ABORT	DEV_SEL_TIMING		MASTER_DPE	FAST_B2B_CAP	RSVDP_22	FAST_66MHZ_CAP	CAP_LIST	INT_STATUS	RSVDP_17		0
rw1ch	rw1ch	rw1ch	rw1ch	rw1ch	r		rw1ch	r	r	r	r	rh	r		r

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PCI_TYPE_RESERV					PCI_TYPE0_INT_EN	RSVDP_9	PCI_TYPE0_SERREN	PCI_TYPE_IDSEL_STEPPING	PCI_TYPE0_PARITY_ERR_EN	PCI_TYPE_VGA_PALETTE_SNOOP	PCI_TYPE_MWI_ENABLE	PCI_TYPE0_SPECIAL_CYCLE_OPERATION	PCI_TYPE0_BUS_MASTER_EN	PCI_TYPE0_MEM_SPACE_EN	PCI_TYPE0_IO_EN
r					rw	r	rw	r	rw	r	r	r	rw	rwh	rwh

Field	Bits	Type	Description
PCI_TYPE0_IO_EN	0	rwh	**IO Space Enable** Controls a Function's response to I/O Space accesses. • When this bit is set, the Function is enabled to decode the address and further process I/O Space accesses • When this bit is clear, all received I/O accesses are caused to be handled as Unsupported Requests. For a Function that does not support I/O Space accesses, the controller hardwires this bit to 0b. Note: The access attributes of this field are as follows: • Wire: !has_io_bar ? RO : RW • Dbi: !has_io_bar ? RO : RW
PCI_TYPE0_MEM_SPACE_EN	1	rwh	**Memory Space Enable** Controls a Function's response to Memory Space accesses. • When this bit is set, the Function is enabled to decode the address and further process Memory Space accesses • When this bit is clear, all received Memory Space accesses are caused to be handled as Unsupported Requests. For a Function does not support Memory Space accesses, the controller hardwires this bit to 0b. Note: The access attributes of this field are as follows: - Wire: RW - Dbi: RW
PCI_TYPE0_BUS_MASTER_EN	2	rw	**Bus Master Enable** Controls the ability of a Function to issue Memory and I/O Read/Write requests. • When this bit is set, the Function is allowed to issue Memory or I/O Requests • When this bit is clear, the Function is not allowed to issue any Memory or I/O Requests Requests other than Memory or I/O Requests are not controlled by this bit. Note: MSI/MSI-X interrupt Messages are in-band memory writes, setting the Bus Master Enable bit to 0b disables MSI/MSI-X interrupt Messages as well. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
PCI_TYPE0_SPECIAL_CYCLE_OPERATION	3	r	**Special Cycle Enable** This bit was originally described in the PCI Local Bus Specification. Its functionality does not apply to PCI Express. The controller hardwires this bit to 0b.
PCI_TYPE_MWI_ENABLE	4	r	**Memory Write and Invalidate** This bit was originally described in the PCI Local Bus Specification and the PCI-to-PCI Bridge architecture specification. Its functionality does not apply to PCI Express, the controller hardwires this bit to 0b.
PCI_TYPE_VGA_PALETTE_SNOOP	5	r	**VGA Palette Snoop** This bit was originally described in the PCI Local Bus Specification and the PCI-to-PCI Bridge architecture specification. Its functionality does not apply to PCI Express, the controller hardwires this bit to 0b.

(table continues...)

(continued)

Field	Bits	Type	Description
PCI_TYPE0_PARITY_ERR_EN	6	rw	**Parity Error Response** This bit controls the logging of poisoned TLPs in the Master Data Parity Error bit in the Status register.
PCI_TYPE_IDSEL_STEPPING	7	r	**IDSEL Stepping/Wait Cycle Control** This bit was originally described in the PCI Local Bus Specification. Its functionality does not apply to PCI Express. The controller hardwires this bit to 0b.
PCI_TYPE0_SERREN	8	rw	**SERR# Enable** When set, this bit enables reporting upstream of Non-fatal and Fatal errors detected by the Function. Note: The errors are reported if enabled either through this bit or through the PCI Express specific bits in the Device Control register.
RSVDP_9	9	r	**Reserved for future use**
PCI_TYPE0_INT_EN	10	rw	**Interrupt Disable** Controls the ability of a Function to generate INTx emulation interrupts. • When set, Functions are prevented from asserting INTx interrupts. • Any INTx emulation interrupts already asserted by the Function must be de-asserted when this bit is Set. INTx interrupts use virtual wires that must, if asserted, be de-asserted using the appropriate Deassert_INTx message(s) when this bit is set • Only the INTx virtual wire interrupt(s) associated with the Function(s) for which this bit is set are affected • For functions that generate INTx interrupts, this bit is required. For functions that do not generate INTx interrupts, this bit is optional
PCI_TYPE_RESERV	15:11	r	**Reserved**
RSVDP_17	18:17	r	**Reserved for future use**
INT_STATUS	19	rh	**Emulation interrupt pending** When set, indicates that an INTx emulation interrupt is pending internally in the Function. Setting the Interrupt Disable bit has no effect on the state of this bit. For Functions that do not generate INTx interrupts, the controller hardwires this bit to 0b.
CAP_LIST	20	r	**Capabilities List** Indicates the presence of an Extended Capability list item. Since all PCI Express device Functions are required to implement the PCI Express Capability structure, the controller hardwires this bit to 1b.
FAST_66MHZ_CAP	21	r	**66MHz Capable** This bit was originally described in the PCI Local Bus Specification. Its functionality does not apply to PCI Express. The controller hardwires this bit to 0b.
RSVDP_22	22	r	**Reserved for future use**
FAST_B2B_CAP	23	r	**Fast Back to Back Transaction Capable** This bit was originally described in the PCI Local Bus Specification. Its functionality does not apply to PCI Express. The controller hardwires this bit to 0b.

(table continues...)

(continued)

Field	Bits	Type	Description
MASTER_DPE	24	rw1ch	**Master Data Parity Error** This bit is set by a Function if the Parity Error Response bit in the Command register is 1b and either of the following two conditions occurs: • Function receives a Poisoned Completion • Function transmits a Poisoned Request If the Parity Error Response bit is 0b, this bit is never set
DEV_SEL_TIMING	26:25	r	**DEVSEL Timing** This field was originally described in the PCI Local Bus Specification. Its functionality does not apply to PCI Express. The controller hardwires this field to 00b.
SIGNALED_TARGET_ABORT	27	rw1ch	**Signaled Target Abort** Software write with 1 clears this bit-field.. 1_B **SET**: This bit is set when a Function completes a Posted or Non-Posted Request as a Completer Abort error The controller hardwires this bit to 0b for Functions that do not signal Completer Abort.
RCVD_TARGET_ABORT	28	rw1ch	**Received Target Abort** Software write with 1 clears this bit-field.. 1_B **SET**: This bit is set when a Requester receives a Completion with Completer Abort Completion Status For Functions that do not make Non-Posted Requests on their own behalf, the controller hardwires this bit to 0b.
RCVD_MASTER_ABORT	29	rw1ch	**Received Master Abort** Software write with 1 clears this bit-field.. 1_B **SET**: This bit is set when a Requester receives a Completion with Unsupported Request Completion Status For Functions that do not make Non-Posted Requests on their own behalf, the controller hardwires this bit to 0b.
SIGNALED_SYS_ERR	30	rw1ch	**Signaled System Error** Software write with 1 clears this bit-field.. 1_B **SET**: This bit is set when a Function sends an ERR_FATAL or ERR_NONFATAL message, and the SERR# Enable bit in the Command register is 1b For Functions that do not send ERR_FATAL or ERR_NONFATAL messages, the controller hardwires this bit to 0b.
DETECTED_PARITY_ERR	31	rw1ch	**Detected Parity Error** Software write with 1 clears this bit-field.. 1_B **SET**: This bit is set by a Function whenever it receives a Poisoned TLP, regardless of the state the Parity Error Response bit in the Command register
0	16	r	**Reserved** Read as 0; should be written with 0.

Table 865 Reset values of STATUS_COMMAND_REG

Reset	Reset value	Note
Kernel Reset	0000 0000 0001 X000 0000 0000 0000 0000$_B$	

19.4.1.100 Class Code and Revision ID Register

This register specifies the class code and revision ID of a function.

CLASS_CODE_REVISION_ID Offset address: 0004008$_H$

Class Code and Revision ID Register Kernel Reset value: 0000 0001$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BASE_CLASS_CODE								SUBCLASS_CODE							
rw								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PROGRAM_INTERFACE								REVISION_ID							
rw								rw							

Field	Bits	Type	Description
REVISION_ID	7:0	rw	**Revision ID** The value in this register specifies a Function specific revision identifier. The value is chosen by the vendor. Zero is an acceptable value. The Revision ID should be viewed as a vendor defined extension to the Device ID. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky.
PROGRAM_INTERFACE	15:8	rw	**Programming Interface** This field identifies a specific register-level programming interface (if any) so that device independent software can interact with the Function. Encodings for interface are provided in the PCI Code and ID Assignment Specification. All unspecified encodings are Reserved. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky.
SUBCLASS_CODE	23:16	rw	**Sub-Class Code** Specifies a base class sub-class, which identifies more specifically the operation of the Function. Encodings for sub-class are provided in the PCI Code and ID Assignment Specification. All unspecified encodings are Reserved. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky.
BASE_CLASS_CODE	31:24	rw	**Base Class Code** A code that broadly classifies the type of operation the Function performs. Encodings for base class, are provided in the PCI Code and ID Assignment Specification. All unspecified encodings are Reserved. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky.

19.4.1.101 BIST, Header Type, Latency Timer, and Cache Line Size Register

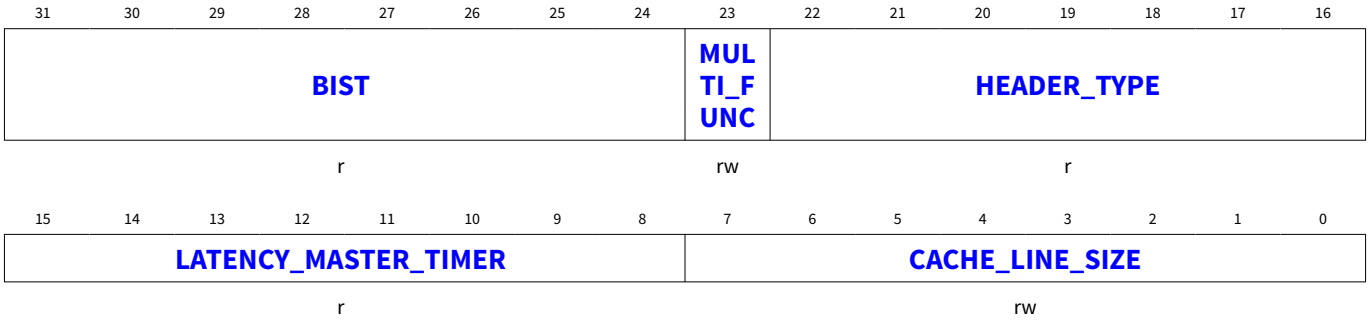
This register provides the status and controls BIST. It also holds information regarding the header layout, latency timer, and cache line size.

BIST_HEADER_TYPE_LATENCY_CACHE_LINE_SIZE_REG

Offset address: $000400C_H$

BIST, Header Type, Latency Timer, and Cache Line Size Register

Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
CACHE_LINE_SIZE	7:0	rw	**Cache Line Size** The Cache Line Size register is programmed by the system firmware or the operating system to system cache line size. However, legacy conventional PCI software may not always be able to program this register correctly especially in the case of Hot-Plug devices. This read-write register is implemented for legacy compatibility purposes but has no effect on any PCI Express device behavior.
LATENCY_MASTER_TIMER	15:8	r	**Latency Timer** The Latency Timer was originally described in the PCI Local Bus Specification and the PCI-to-PCI Bridge Architecture Specification. Its functionality does not apply to PCI Express. The controller hardwires this register to 00h.
HEADER_TYPE	22:16	r	**Header Layout** This field identifies the layout of the second part of the predefined header. The controller uses 000 0000b encoding.
MULTI_FUNC	23	rw	**Multi-Function Device** Except where stated otherwise, it is recommended that this bit be set if there are multiple Functions, and clear if there is only one Function. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky. 0_B **CLEAR**: Software must not probe for Functions other than Function 0 1_B **SET**: Indicates that the Device may contain multiple Functions, but not necessarily Software is permitted to probe for Functions other than Function 0

(table continues...)

(continued)

Field	Bits	Type	Description
BIST	31:24	r	**BIST** This register is used for control and status of BIST. For Functions that do not support BIST the controller hardwires the register to 00h. A Function whose BIST is invoked must not prevent normal operation of the PCI Express Link. Bit descriptions: • [31]: BIST Capable. When Set, this bit indicates that the Function supports BIST. When Clear, the Function does not support BIST • [30]: Start BIST. If BIST Capable is Set, Set this bit to invoke BIST. The Function resets the bit when BIST is complete. Software is permitted to fail the device if this bit is not Clear (BIST is not complete) 2 seconds after it had been Set. Writing this bit to 0b has no effect. This bit must be hardwired to 0b if BIST Capable is Clear • [29:28]: Reserved • [27:24]: Completion Code. This field encodes the status of the most recent test. A value of 0000b means that the Function has passed its test. Non-zero values mean the Function failed. Function-specific failure codes can be encoded in the non-zero values. This field's value is only meaningful when BIST Capable is Set and Start BIST is Clear. This field must be hardwired to 0000b if BIST Capable is clear

19.4.1.102 BAR0 Register i

System software must build a consistent address map before booting the machine to an operating system. This means it has to determine how much memory is in the system, and how much address space the Functions in the system require. After determining this information, system software can map the Functions into reasonable locations and proceed with system boot. In order to do this mapping in a device-independent manner, the base registers for this mapping are placed in the predefined header portion of Configuration Space. It is strongly recommended that power-up firmware/software also support the optional Enhanced Configuration Access Mechanism (ECAM).

BARi_REG (i=0-5) Offset address: 0004010_H+i*4

BAR0 Register i Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BAR0_START															
rwh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BAR0_START												BAR0_PREFETCH	BAR0_TYPE		BAR0_MEM_IO
rwh												rwh	rwh		rwh

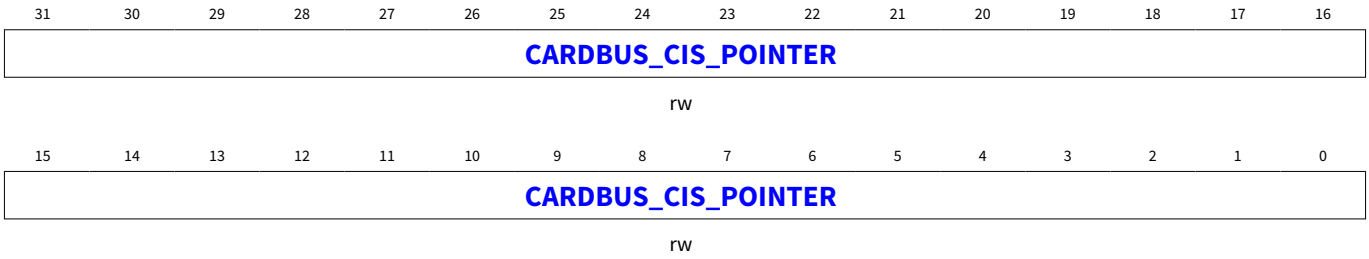
Field	Bits	Type	Description
BAR0_MEM_IO	0	rwh	**BAR0 Memory Space Indicator** This bit is used to determine whether the register maps into Memory or I/O Space. Base Address registers that map to Memory Space must return a 0b. Base Address registers that map to I/O Space must return a 1b. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (BAR_ENABLED == 1) then (if [DBI_RO_WR_EN == 1] then R(Sticky)/W(Sticky) else R(Sticky)) else RO(Sticky) Note: This register field is sticky.
BAR0_TYPE	2:1	rwh	**BAR0 Type** - Memory Space: Base Address registers that map into Memory Space can be 32 bits or 64 bits wide (to support mapping into a 64-bit address space). The encodings defined in Values: apply. - IO Space: Bit 1 is reserved and must return 0b on reads. Bits[31:2] are used to map the function into IO space. The encodings defined in Values: do not apply. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (BAR_ENABLED == 1) then (if [DBI_RO_WR_EN == 1] then R(Sticky)/W(Sticky) else R(Sticky)) else RO(Sticky) Note: This register field is sticky. 00_B **BAR_32**: Base register is 32 bits wide and can be mapped anywhere in the 32 address bit Memory Space 01_B **RSVD_1**: Reserved 10_B **BAR_64**: Base register is 64 bits wide and can be mapped anywhere in the 64 address bit Memory Space 11_B **RSVD_2**: Reserved
BAR0_PREFETCH	3	rwh	**BAR0 Prefetchable** - Memory Space: Set to one if data is prefetchable. A Function is permitted to mark a range as prefetchable. If there are no side effects on reads, the function returns all bytes on reads regardless of the byte enables, and host bridges can merge processor writes into this range without causing errors. Bit must me clear otherwise. - IO Space: Not applicable Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (BAR_ENABLED == 1) then (if [DBI_RO_WR_EN == 1] then R(Sticky)/W(Sticky) else R(Sticky)) else RO(Sticky) Note: This register field is sticky.
BAR0_START	31:4	rwh	**BAR0 Base Address** - Memory Space: Base Address. - IO Space: bits[31:2] are used to map the function into IO space/Base. Address. Note: The access attributes of this field are as follows: - Wire: R(Sticky)/W(Sticky) if (BAR_ENABLED == 1) then (if [DBI_RO_WR_EN == 1] enabled else R(Sticky) - Dbi: R(Sticky)/W(Sticky) if enabled else R(Sticky) Note: This register field is sticky.

19.4.1.103 CardBus CIS Pointer Register

This register holds the CardBus CIS pointer.

CARDBUS_CIS_PTR_REG Offset address: 0004028_H

CardBus CIS Pointer Register Kernel Reset value: $0000\ 0000_H$



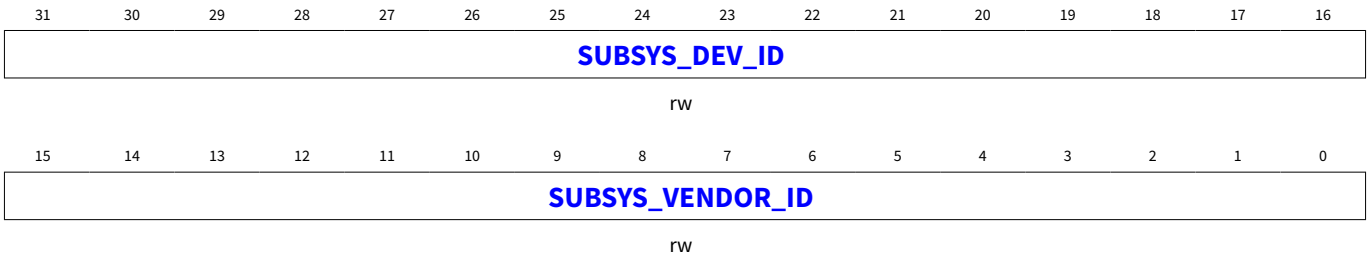
Field	Bits	Type	Description
CARDBUS_CIS_POINTER	31:0	rw	**CardBus CIS Pointer** Its functionality does not apply to PCI Express. It must be hardwired to 0000 0000h. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky.

19.4.1.104 Subsystem ID and Subsystem Vendor ID Register

These registers are used to uniquely identify the add-in card or subsystem where the PCI Express component resides. They provide a mechanism for vendors to distinguish their products from one another even though the assemblies may have the same PCI Express component on them (and, therefore, the same Vendor ID and Device ID).

SUBSYSTEM_ID_SUBSYSTEM_VENDOR_ID_REG Offset address: $000402C_H$

Subsystem ID and Subsystem Vendor ID Register Kernel Reset value: $0000\ 0000_H$



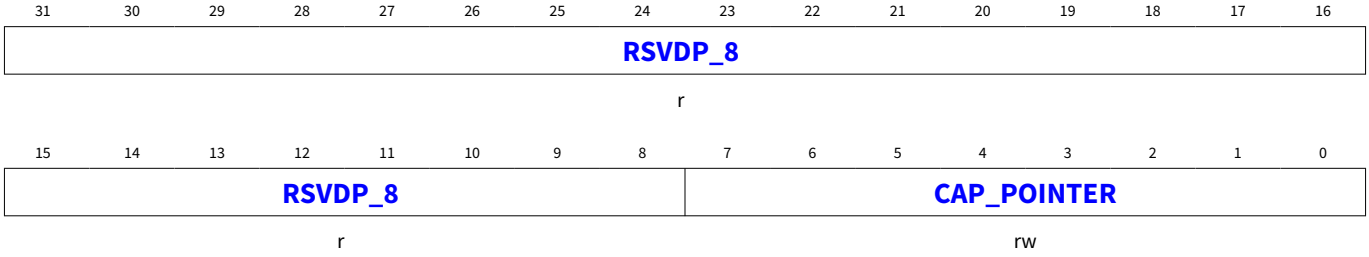
Field	Bits	Type	Description
SUBSYS_VENDOR_ID	15:0	rw	**Subsystem Vendor ID** Subsystem Vendor IDs can be obtained from the PCI SIG and are used to identify the vendor of the add-in card or subsystem. Values for the Subsystem ID are vendor-specific. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky.
SUBSYS_DEV_ID	31:16	rw	**Subsystem ID** Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky.

19.4.1.105 Capabilities Pointer Register

This register is used to point to a linked list of capabilities implemented by a Function.

PCI_CAP_PTR_REG Offset address: 0004034_H

Capabilities Pointer Register Kernel Reset value: $0000\ 0040_H$



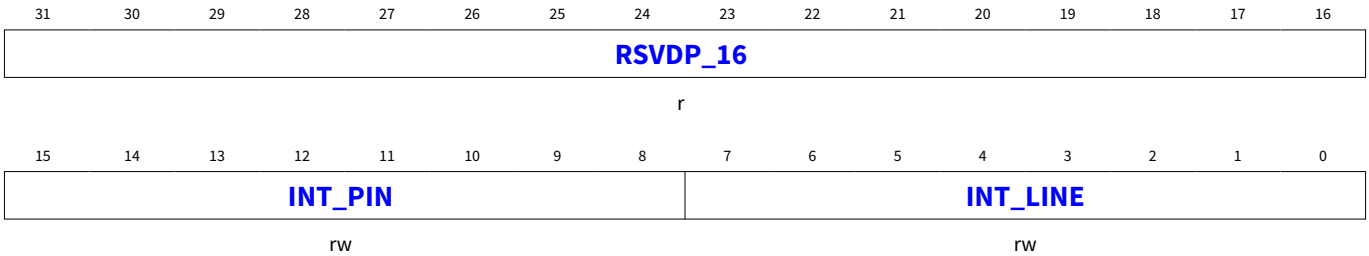
Field	Bits	Type	Description
CAP_POINTER	7:0	rw	**Capabilities Pointer** This register points to a valid capability structure. Either this structure is the PCI Express Capability structure, or a subsequent list item points to the PCI Express Capability structure. The bottom two bits are reserved, the controller sets it to 00b. Software must mask these bits off before using this register as a pointer in Configuration Space to the first entry of a linked list of new capabilities. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky.
RSVDP_8	31:8	r	**Reserved for future use**

19.4.1.106 Max_Lat, Min_Gnt, Interrupt Pin, and Interrupt Line Register

The Interrupt Line register communicates interrupt line routing information. The Interrupt Pin register identifies the legacy interrupt Message(s) the Function uses.

MAX_LATENCY_MIN_GRANT_INTERRUPT_PIN_INTERRUPT_LINE_REG Offset address: $000403C_H$

Max_Lat, Min_Gnt, Interrupt Pin, and Interrupt Line Register Kernel Reset value: $0000\ 01FF_H$



Field	Bits	Type	Description
INT_LINE	7:0	rw	**Interrupt Line** The Interrupt Line register communicates interrupt line routing information. The register must be implemented by any Function that uses an interrupt pin. Values in this register are programmed by system software and are system architecture specific. The Function itself does not use this value; rather the value in this register is used by device drivers and operating systems.
INT_PIN	15:8	rw	**Interrupt Pin** The Interrupt Pin register identifies the legacy interrupt Message(s) the Function uses. All encodings other than the defined encodings are reserved. PCI Express defines one legacy interrupt Message for a single Function device and up to four legacy interrupt Messages for a multi-Function device. For a single Function device, only INTA may be used. Any Function on a multi-Function device can use any of the INTx Messages. If a device implements a single legacy interrupt Message, it must be INTA; if it implements two legacy interrupt Messages, they must be INTA and INTB; and so forth. For a multi-Function device, all Functions may use the same INTx Message or each may have its own (up to a maximum of four Functions) or any combination thereof. A single Function can never generate an interrupt request on more than one INTx Message. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R Note: This register field is sticky. 00_H **NO_INT**: Indicates that the Function uses no legacy interrupt Message(s) 01_H **INTA**: Map to legacy interrupt Messages for INTA 02_H **INTB**: Map to legacy interrupt Messages for INTB 03_H **INTC**: Map to legacy interrupt Messages for INTC 04_H **INTD**: Map to legacy interrupt Messages for INTD **others,** Reserved
RSVDP_16	31:16	r	**Reserved for future use**

19.4.1.107 Power Management Capabilities Register

This register provides information refarding the Power Management Capabilities.

CAP_ID_NXT_PTR_REG Offset address: 0004040_H

Power Management Capabilities Register Reset values see: Table 866

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PME_SUPPORT					D2_SUPPORT	D1_SUPPORT	AUX_CURR			DSI	0	PME_CLK	PM_SPEC_VER		
rwh					rw	rw	rw			rw	r	r	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PM_NEXT_POINTER								PM_CAP_ID							
rw								r							

Field	Bits	Type	Description
PM_CAP_ID	7:0	r	**Capability ID** This field returns 01h to indicate that this is the PCI Power Management Capability. Each function may have only one item in its capability list with Capability ID set to 01h.
PM_NEXT_POINTER	15:8	rw	**Next Capability Pointer** This field provides an offset into the function's configuration space pointing to the location of next item in the capabilities list. If there are no additional items in the capabilities list, this field is set to 00h. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R Note: This register field is sticky.
PM_SPEC_VER	18:16	rw	**Version** This field provides the Power Management specification version. The controller hardwires this field to 011b for functions compliant to PCI Express Base Specification, Revision 4.0, Version 1.0>. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R Note: This register field is sticky.
PME_CLK	19	r	**PME Clock** Does not apply to PCI Express, the controller hardwires it to 0b. Note: This register field is sticky.
DSI	21	rw	**Device Specific Initialization** The DSI bit indicates whether special initialization of this function is required. When set, indicates that the function requires a device specific initialization sequence following a transition to the D0uninitialized state. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R Note: This register field is sticky. DSI capability is not supported in this implementation

(table continues...)

(continued)

Field	Bits	Type	Description
AUX_CURR	24:22	rw	**Aux_Current** This 3 bit field reports the Vaux auxiliary current requirements for the function. If this function implements the Data Register, the controller hardwires this field to 000b. If PME_Support is 0 xxxxb (PME assertion from D3cold is not supported), the controller hardwires this field to 0000b. For functions where PME_Support is 1 xxxxb (PME assertion from D3cold is supported), and which do not implement the Data field, the encodings defined in Values: apply: Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R Note: This register field is sticky. 000_B **SELF_**: 0 self powered 001_B **_055mA_**: 55mA Vaux Max Current Required 010_B **_100mA_**: 100mA Vaux Max Current Required 011_B **_160mA_**: 160mA Vaux Max Current Required 100_B **_220mA_**: 220mA Vaux Max Current Required 101_B **_270mA_**: 270mA Vaux Max Current Required 110_B **_320mA_**: 320mA Vaux Max Current Required 111_B **_370mA_**: 375mA Vaux Max Current Required
D1_SUPPORT	25	rw	**D1_Support** If this bit is set, this function supports the D1 Power Management state. Functions that do not support D1 must always return a value of 0b for this bit. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R Note: This register field is sticky. D1-support capability is not supported in this implementation
D2_SUPPORT	26	rw	**D2_Support** If this bit is set, this function supports the D2 Power Management state. Functions that do not support D2 must always return a value of 0b for this bit. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R Note: This register field is sticky. D2-support capability is not supported in this implementation

(table continues...)

(continued)

Field	Bits	Type	Description
PME_SUPPORT	31:27	rwh	**PME_Support** This 5-bit field indicates the power states in which the function may generate a PME and/or forward PME messages. A value of 0b for any bit indicates that the function is not capable of asserting PME while in that power state. • bit(27) X XXX1b - PME can be generated from D0 • bit(28) X XX1Xb - PME can be generated from D1 • bit(29) X X1XXb - PME can be generated from D2 • bit(30) X 1XXXb - PME can be generated from D3hot • bit(31) 1 XXXXb - PME can be generated from D3cold Bit 31 (PME can be asserted from D3cold) represents a special case. Functions that set this bit require some sort of auxiliary power source. Implementation specific mechanisms are recommended to validate that the power source is available before setting this bit. Each bit that corresponds to a supported D-state must be set for PCI-PCI Bridge structures representing Ports on Root Complexes/Switches to indicate that the Bridge will forward PME Messages. Bit 31 must only be set if the Port is still able to forward PME Messages when main power is not available. The read value from this field is the write value && (sys_aux_pwr_det, 1'b1, D2_SUPPORT, D1_SUPPORT, 1'b1), where D1_SUPPORT and D2_SUPPORT are fields in this register. The reset value PME_SUPPORT_n && (sys_aux_pwr_det, 1'b1, D2_SUPPORT, D1_SUPPORT, 1'b1), where PME_SUPPORT_n is a configuration parameter. Note: The access attributes of this field are as follows: - Wire: R - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R Note: This register field is sticky.
0	20	r	**Reserved** Read as 0; should be written with 0.

Table 866 Reset values of CAP_ID_NXT_PTR_REG

Reset	Reset value	Note
Kernel Reset	XXXX X001 1100 0011 0101 0000 0000 0001_B	

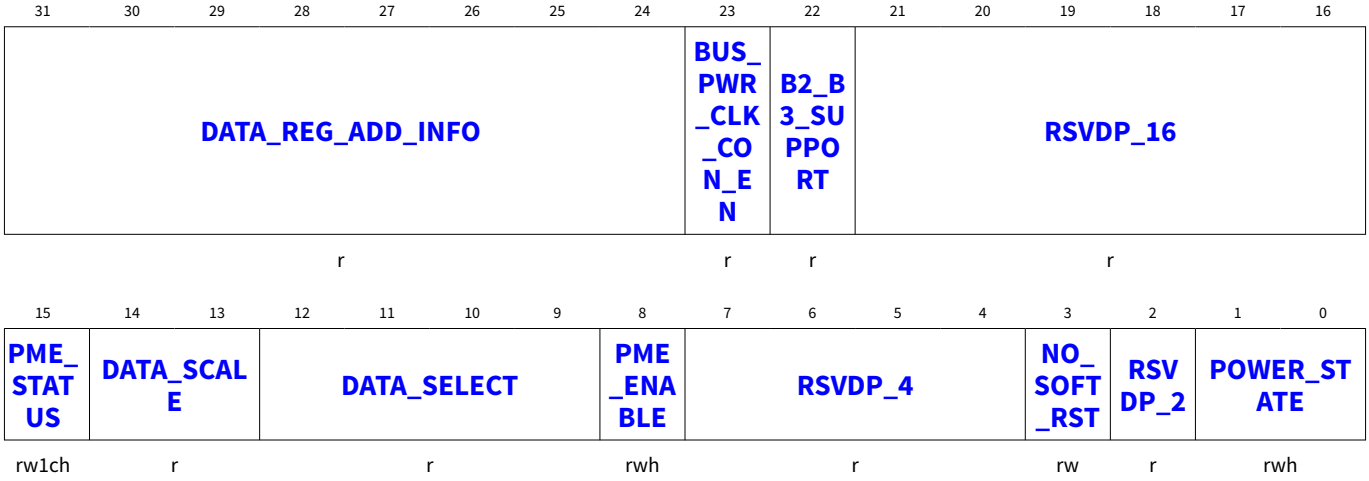
19.4.1.108 Power Management Control and Status Register

This register is used to manage the PCI function's power management state as well as to enable/monitor PMEs.

CON_STATUS_REG Offset address: 0004044_H

Power Management Control and Status Register Reset values see: Table 867

19 Peripheral Component Interconnect Express (PCIe)



Field	Bits	Type	Description
POWER_STATE	1:0	rwh	**PowerState** This 2-bit field is used both to determine the current power state of a function and to set the function into a new power state. You can write to this register; however, the read-back value is the actual power state, not the write value. If you attempt to write an unsupported, optional state to this field, the write operation completes normally; however, the data is discarded and no state change occurs. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W 00_B **D0**: D0 power state 01_B **D1**: D1 power state 10_B **D2**: D2 power state 11_B **D3hot**: D3hot D3hot power state
RSVDP_2	2	r	**Reserved for future use**
NO_SOFT_RST	3	rw	**No_Soft_Reset** This bit indicates the state of the function after writing the PowerState field to transition the function from D3hot to D0. • When set, this transition preserves internal function state. The function is in D0Active and no additional software intervention is required • When clear, this transition results in undefined internal function state Regardless of this bit, functions that transition from D3hot to D0 by Fundamental Reset will return to D0Uninitialized with only PME context preserved if PME is supported and enabled. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R Note: This register field is sticky.
RSVDP_4	7:4	r	**Reserved for future use**

(table continues...)

(continued)

Field	Bits	Type	Description
PME_ENABLE	8	rwh	**PME_En** • When set, the function is permitted to generate a PME • When clear, the function is not permitted to generate a PME. If PME_Support is 1 xxxxb (PME generation from D3cold) or the function consumes Aux power and Aux power is available this bit is RWS and the bit is not modified by Conventional Reset If PME_Support is 0 xxxxb, this field is not sticky (RW). If PME_Support is 0 0000b, the controller hardwires this bit to 0b. Note: This register field is sticky.
DATA_SELECT	12:9	r	**Data_Select** This 4-bit field is used to select which data is to be reported through the Data and Data_Scale field. If the Data field is not implemented, this field must be hardwired to 0000b.
DATA_SCALE	14:13	r	**Data_Scale** This field indicates the scaling factor to be used when interpreting the value of the Data field. The value and meaning of this field varies depending on which data value has been selected by the Data_Select field. For more details, see 7.5.2.3 section of PCI Express Base Specification.
PME_STATUS	15	rw1ch	**PME_Status** This bit is set when the function normally generates a PME signal. The value of this bit is not affected by the value of the PME_En bit. If PME_Support bit 31 of the Power Management Capabilities register is clear, this bit is permitted to be hardwired to 0b. Functions that consume Aux power must preserve the value of this sticky register when Aux power is available. In such functions, this register value is not modified by Conventional Reset Note: This register field is sticky. Software write with 1 clears this bit-field.
RSVDP_16	21:16	r	**Reserved for future use**
B2_B3_SUPPORT	22	r	**B2B3 Support for D3hot** If this field is set, B2B3 support for D3hot is available.
BUS_PWR_CLK_CON_EN	23	r	**Bus Power/Clock Control Enable** If this field is set, Bus Power/Clock Control is Enable.
DATA_REG_ADD_INFO	31:24	r	**Data** This field is used to report the state dependent data requested by the Data_Select field. The value of this field is scaled by the value reported by the Data_Scale field.

Table 867 Reset values of CON_STATUS_REG

Reset	Reset value	Note
Kernel Reset	0000 0000 0000 0000 0000 000X 0000 $10XX_B$	

19.4.1.109 MSI Capability Header and Message Control Register

This register holds MSI Capability Header information and controls the MSI behaviour.

PCI_MSI_CAP_ID_NEXT_CTRL_REG Offset address: 0004050_H

MSI Capability Header and Message Control Register Kernel Reset value: $018A\ 7005_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_27					PCI_MSI_EXT_DATA_EN	PCI_MSI_EXT_DATA_CAP	PCI_PVM_SUPPORT	PCI_MSI_64_BIT_ADDR_CAP	PCI_MSI_MULTIPLE_MSG_EN			PCI_MSI_MULTIPLE_MSG_CAP			PCI_MSI_ENABLE
r					rw	rw	r	rw	rw			rw			rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PCI_MSI_CAP_NEXT_OFFSET								PCI_MSI_CAP_ID							
rw								r							

Field	Bits	Type	Description
PCI_MSI_CAP_ID	7:0	r	**Capability ID** Indicates the MSI Capability structure. This field returns a Capability ID of 05h indicating that this is an MSI Capability structure.
PCI_MSI_CAP_NEXT_OFFSET	15:8	rw	**Next Capability Pointer** This field contains the offset to the next PCI Capability structure or 00h if no other items exist in the linked list of Capabilities. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R Note: This register field is sticky.
PCI_MSI_ENABLE	16	rw	**MSI Enable** • If set and the MSI-X Enable bit in the MSI-X Message Control register is clear, the function is permitted to use MSI to request service and is prohibited from using INTx interrupts. System configuration software sets this bit to enable MSI. A device driver is prohibited from writing this bit to mask a function's service request. For more details on control of INTx interrupts, see section 7.5.1.1 of PCI Express Base Specification. • If clear, the function is prohibited from using MSI to request service Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W

(table continues...)

(continued)

Field	Bits	Type	Description
PCI_MSI_MULTIPLE_MSG_CAP	19:17	rw	**Multiple Message Capable** System software reads this field to determine the number of requested vectors. The number of requested vectors must be aligned to a power of two (if a function requires three vectors, it requests four by initializing this field to 010b). All encodings other than the defined encodings are reserved. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R Note: This register field is sticky. 000_B **_1_VECTOR**: 1 vector requested 001_B **_2_VECTOR**: 2 vectors requested 010_B **_4_VECTOR**: 4 vectors requested 011_B **_8_VECTOR**: 8 vectors requested 100_B **_16_VECTOR**: 16 vectors requested 101_B **_32_VECTOR**: 32 vectors requested **others,** Reserved
PCI_MSI_MULTIPLE_MSG_EN	22:20	rw	**Multiple Message Enable** Software writes to this field to indicate the number of allocated vectors (equal to or less than the number of requested vectors). The number of allocated vectors is aligned to a power of two. If a function requests four vectors (indicated by a Multiple Message Capable encoding of 010b), system software can allocate either four, two, or one vector by writing a 010b, 001b, or 000b to this field, respectively. When MSI is enabled, a function will be allocated at least 1 vector. All encodings other than the defined encodings are reserved. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W 000_B **_1_VECTOR**: 1 vector allocated 001_B **_2_VECTOR**: 2 vectors allocated 010_B **_4_VECTOR**: 4 vectors allocated 011_B **_8_VECTOR**: 8 vectors allocated 100_B **_16_VECTOR**: 16 vectors allocated 101_B **_32_VECTOR**: 32 vectors allocated **others,** Reserved
PCI_MSI_64_BIT_ADDR_CAP	23	rw	**64 bit address capable** • If set, the function is capable of sending a 64-bit message address • If clear, the function is not capable of sending a 64-bit message address. This bit must be set if the function is a PCI Express Endpoint Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R Note: This register field is sticky.
PCI_PVM_SUPPORT	24	r	**Per-Vector Masking Capable** • If set, the function supports MSI Per-Vector Masking • If clear, the function does not support MSI Per-Vector Masking

(table continues...)

(continued)

Field	Bits	Type	Description
PCI_MSI_EXT_DATA_CAP	25	rw	**Extended Message Data Capable** • If set, the function is capable of providing Extended Message Data • If clear, the function does not support providing Extended Message Data Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R Note: This register field is sticky. Extended Message Data Capability is not supported in the current implementation
PCI_MSI_EXT_DATA_EN	26	rw	**Extended Message Data Enable** • If set, the function is enabled to provide Extended Message Data • If clear, the function is not enabled to provide Extended Message Data Note: The access attributes of this field are as follows: • Wire: PCI_MSI_CAP_ID_NEXT_CTRL_REG.PCI_MSI_EXT_DATA_CAP ? RW : RO • Dbi: PCI_MSI_CAP_ID_NEXT_CTRL_REG.PCI_MSI_EXT_DATA_CAP ? RW : RO
RSVDP_27	31:27	r	**Reserved for future use**

19.4.1.110 Message Address Register for MSI (Offset 04h)

This register holds the system specified message address for an MSI transaction.

MSI_CAP_OFF_04H_REG Offset address: 0004054_H

Message Address Register for MSI [Offset 04h) Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PCI_MSI_CAP_OFF_04H															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PCI_MSI_CAP_OFF_04H														RSVDP_0	
rw														r	

Field	Bits	Type	Description
RSVDP_0	1:0	r	**Reserved for future use**
PCI_MSI_CAP_OFF_04H	31:2	rw	**Message Address - System-specified message address** If the Message Enable bit (bit 16 of the PCI_MSI_CAP_ID_NEXT_CTRL_REG register) is set, the contents of this field specify the DWORD-aligned address (Address[31:02]) for the MSI transaction. Address[1:0] are set to 00b. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W

19.4.1.111 Message Address Register for MSI (Offset 08h)

This register represents the Message Upper Address Register for MSI (Offset 08h). It specifies the Message Upper Address (System-specified message upper address). This register is required for PCI Express Endpoints and is optional for other function types. If the Message Enable bit (bit 0 of the Message Control register) is set, the contents of this register (if non-zero) specify the upper 32-bits of a 64-bit message address (Address[63:32]). If the contents of this register are zero, the Function uses the 32 bit address specified by the Message Address register.

MSI_CAP_OFF_08H_REG Offset address: 0004058_H

Message Address Register for MSI [Offset 08h) Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PCI_MSI_CAP_OFF_0AH															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PCI_MSI_CAP_OFF_08H															
rw															

Field	Bits	Type	Description
PCI_MSI_CAP_OFF_08H	15:0	rw	**For a function that supports a 32-bit message address, this field contains Message Data (System-specified message data)** If the Message Enable bit (bit 16 of the PCI_MSI_CAP_ID_NEXT_CTRL_REG) is set, the function sends a DWORD Memory Write transaction using Message Data for the lower 16 bits. All 4 Byte Enables are set. The Multiple Message Enable field (bits 22:20 of the PCI_MSI_CAP_ID_NEXT_CTRL_REG) defines the number of low order message data bits the function is permitted to modify to generate its system software allocated vectors. For example, a Multiple Message Enable encoding of 010b indicates the function has been allocated four vectors and is permitted to modify message data bits 1 and 0 (a function modifies the lower message data bits to generate the allocated number of vectors). If the Multiple Message Enable field is 000b, the Function is not permitted to modify the message data. For a function that supports a 64-bit message address, it contains lower 16 bits of the Message Upper Address. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W

(table continues...)

(continued)

Field	Bits	Type	Description
PCI_MSI_CAP_OFF_0AH	31:16	rw	**For a function that supports a 32-bit message address, this field contains Extended Message Data (System-specified message data)** Extended Message Data Capability is not supported in the current implementation For the MSI Capability structures without per-vector masking, it must be implemented if the Extended Message Data Capable bit is set; otherwise, it is outside the MSI Capability structure and undefined. For the MSI Capability structures with Per-vector Masking, it must be implemented if the Extended Message Data Capable bit is set; otherwise, it is RsvdP. If the Extended Message Data Enable bit (bit 26 of the PCI_MSI_CAP_ID_NEXT_CTRL_REG) is set, the DWORD Memory Write transaction uses Extended Message Data for the upper 16 bits; otherwise, it uses 0000h for the upper 16 bits. For a function that supports a 64-bit message address, it contains upper 16 bits of the Message Upper Address. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W

19.4.1.112 Message Address Register for MSI (Offset 0Ch)

This register contains Message Data.

MSI_CAP_OFF_0CH_REG Offset address: $000405C_H$

Message Address Register for MSI [Offset 0Ch) Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PCI_MSI_CAP_OFF_0EH															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PCI_MSI_CAP_OFF_0CH															
rw															

Field	Bits	Type	Description
PCI_MSI_CAP_OFF_0CH	15:0	rw	**For a function that supports a 32-bit message address, this field contains the lower Mask Bits when the Per Vector Masking Capable bit (PCI_MSI_CAP_ID_NEXT_CTRL_REG.PCI_PVM_SUPPORT) is set** This field contains Message Data (System-specified message data). If the Message Enable bit (bit 16 of the PCI_MSI_CAP_ID_NEXT_CTRL_REG) is set, the function sends a DWORD Memory Write transaction using Message Data for the lower 16 bits. All 4 Byte Enables are set. The Multiple Message Enable field (bits 22:20 of the PCI_MSI_CAP_ID_NEXT_CTRL_REG) defines the number of low order message data bits the function is permitted to modify to generate its system software allocated vectors. For example, a Multiple Message Enable encoding of 010b indicates the function has been allocated four vectors and is permitted to modify message data bits 1 and 0 (a function modifies the lower message data bits to generate the allocated number of vectors). If the Multiple Message Enable field is 000b, the Function is not permitted to modify the message data. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W

(table continues...)

(continued)

Field	Bits	Type	Description
PCI_MSI_CAP_OFF_0EH	31:16	rw	**For a function that supports a 32-bit message address, this field contains the upper Mask Bits when the Per Vector Masking Capable bit (PCI_MSI_CAP_ID_NEXT_CTRL_REG.PCI_PVM_SUPPORT) is set** For a function that supports a 64-bit message address, this field contains Message Data (System-specified message data). Note: The access attributes of this field are as follows: - Wire: RW - Dbi: RW

19.4.1.113 Message Address Register for MSI (Offset 10h)

This register contains the Mask Bits.

MSI_CAP_OFF_10H_REG Offset address: 0004060_H

Message Address Register for MSI [Offset 10h) Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PCI_MSI_CAP_OFF_10H															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PCI_MSI_CAP_OFF_10H															
rw															

Field	Bits	Type	Description
PCI_MSI_CAP_OFF_10H	31:0	rw	**Used for MSI when the Per Vector Masking Capable bit (PCI_MSI_CAP_ID_NEXT_CTRL_REG.PCI_PVM_SUPPORT) is set** It contains Mask Bits. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W

19.4.1.114 Message Address Register for MSI (Offset 14h)

Pending Bits Register for MSI.

MSI_CAP_OFF_14H_REG Offset address: 0004064_H

Message Address Register for MSI [Offset 14h) Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PCI_MSI_CAP_OFF_14H															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PCI_MSI_CAP_OFF_14H															
r															

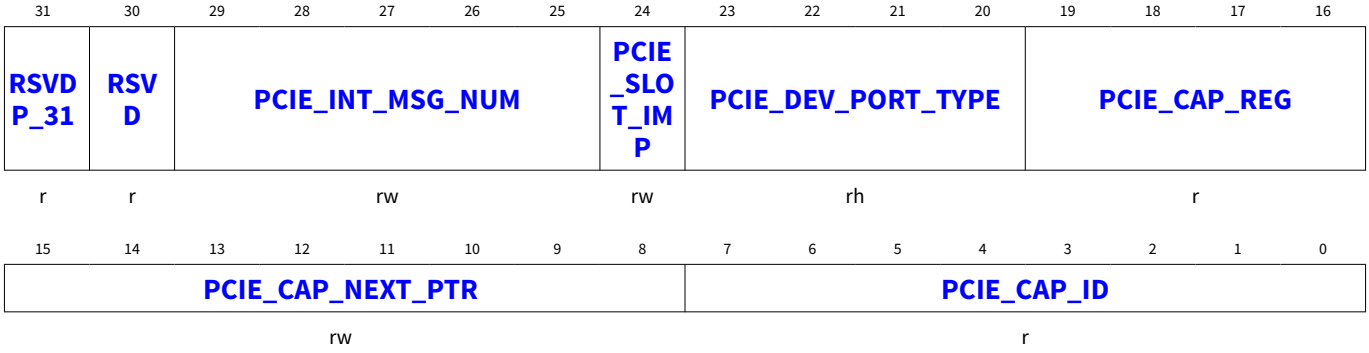
Field	Bits	Type	Description
PCI_MSI_CAP_OFF_14H	31:0	r	**Pending Bits** For each pending bit that is set, the function has a pending associated message.

19.4.1.115 PCI Express Capabilities, ID, Next Pointer Register

This is the PCI Express Capabilities, ID, and Next Pointer Register.

PCIE_CAP_ID_PCIE_NEXT_CAP_PTR_PCIE_CAP_REG Offset address: 0004070_H

PCI Express Capabilities, ID, Next Pointer Register Kernel Reset value: $0002\ B010_H$



Field	Bits	Type	Description
PCIE_CAP_ID	7:0	r	**Capability ID** Indicates the PCI Express Capability structure. This field must return a Capability ID of 10h indicating that this is a PCI Express Capability structure.
PCIE_CAP_NEXT_PTR	15:8	rw	**Next Capability Pointer** This field contains the offset to the next PCI Capability structure or 00h if no other items exist in the linked list of Capabilities. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R Note: This register field is sticky.
PCIE_CAP_REG	19:16	r	**Capability Version** Indicates PCI-SIG defined PCI Express Capability structure version number. A version of the specification that changes the PCI Express Capability structure in a way that is not otherwise identifiable (for example, through a new Capability field) is permitted to increment this field. All such changes to the PCI Express Capability structure must be software-compatible. Software must check for Capability Version numbers that are greater than or equal to the highest number defined when the software is written, as functions reporting any such Capability Version numbers will contain a PCI Express Capability structure that is compatible with that piece of software. The controller hardwires this field to 2h for functions compliant to PCI Express Base Specification, Revision 4.0, Version 1.0. Note: This register field is sticky.

(table continues...)

(continued)

Field	Bits	Type	Description
PCIE_DEV_PORT_TYPE	23:20	rh	**Device/Port Type** Indicates the specific type of this PCI Express function. Note: Different functions in a Multi-Function Device can generally be of different types. Defined encodings for functions that implement a Type 00h PCI Configuration Space header are: Defined encodings for functions that implement a Type 01h PCI Configuration Space header are: All other encodings are Reserved. Note: Different Endpoint types have notably different requirements in Section 1.3.2 of PCI Express Base Specification regarding I/O resources, Extended Configuration Space, and other capabilities. 0_H **PCIE_EP**: PCI Express Endpoint 1_H **PCIE_LEGACY_EP**: Legacy PCI Express Endpoint 4_H **ROOT_PORT_PCIE_RC**: Root Port of PCI Express Root Complex 5_H **USP_PCIE_SWITCH**: Upstream Port of PCI Express Switch 6_H **DSP_PCIE_SWITCH**: Downstream Port of PCI Express Switch **others,** Reserved
PCIE_SLOT_IMP	24	rw	**Slot Implemented** When set, this bit indicates that the Link associated with this Port is connected to a slot (as compared to being connected to a system-integrated device or being disabled). This bit is valid for Downstream Ports. This bit is undefined for Upstream Ports. Note: The access attributes of this field are as follows: - Wire: HWINIT - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R
PCIE_INT_MSG_NUM	29:25	rw	**PCIE Interrupt Message Number** Interrupt Message Number. This field indicates which MSI/MSI-X vector is used for the interrupt message generated in association with any of the status bits of this Capability structure. For MSI, the value in this field indicates the offset between the base Message Data and the interrupt message that is generated. Hardware is required to update this field so that it is correct if the number of MSI Messages assigned to the Function changes when software writes to the Multiple Message Enable field in the MSI Message Control register. For MSI-X, the value in this field indicates which MSI-X Table entry is used to generate the interrupt message. The entry must be one of the first 32 entries even if the Function implements more than 32 entries. For a given MSI-X implementation, the entry must remain constant. If both MSI and MSI-X are implemented, they are permitted to use different vectors, though software is permitted to enable only one mechanism at a time. If MSI-X is enabled, the value in this field must indicate the vector for MSI-X. If MSI is enabled or neither is enabled, the value in this field must indicate the vector for MSI. If software enables both MSI and MSI-X at the same time, the value in this field is undefined. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R Note: This register field is sticky.
RSVD	30	r	**Reserved**
RSVDP_31	31	r	**Reserved for future use**

19.4.1.116 Device Capabilities Register

The Device Capabilities register identifies PCI Express device function specific capabilities.

DEVICE_CAPABILITIES_REG Offset address: 0004074_H

Device Capabilities Register Reset values see: Table 868

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_29			PCIE_CAP_FLR_CAP	PCIE_CAP_CAP_SLOT_PWR_LMT_SCALE		PCIE_CAP_CAP_SLOT_PWR_LMT_VALUE								RSVDP_16	
r			rw	rh		rh								r	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PCIE_CAP_ROLE_BASED_ERR_REPORT	RSVDP_12			PCIE_CAP_EP_L1_ACCPT_LATENCY			PCIE_CAP_EP_L0S_ACCPT_LATENCY			PCIE_CAP_EXT_TAG_SUPP	PCIE_CAP_PHANTOM_FUNC_SUPPORT		PCIE_CAP_MAX_PAYLOAD_SIZE		
rw	r			rw			rw			rw	rw		rw		

Field	Bits	Type	Description
PCIE_CAP_MAX_PAYLOAD_SIZE	2:0	rw	**Max_Payload_Size Supported** This field indicates the maximum payload size that the function can support for TLPs. All encodings other than the defined encodings are reserved. The functions of a Multi-Function Device are permitted to report different values for this field. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R Note: This register field is sticky. 000_B **MAX_128B_SIZE**: 128 bytes max payload size 001_B **MAX_256B_SIZE**: 256 bytes max payload size 010_B **MAX_512B_SIZE**: 512 bytes max payload size 011_B **MAX_1024B_SIZE**: 1024 bytes max payload size 100_B **MAX_2048B_SIZE**: 2048 bytes max payload size 101_B **MAX_4096B_SIZE**: 4096 bytes max payload size **others,** Reserved

(table continues...)

(continued)

Field	Bits	Type	Description
PCIE_CAP_PHANTOM_FUNC_SUPPORT	4:3	rw	**Phantom Functions Supported** For functions that do not implement this capability (Phantom function capability is not available in this implementation), this bit will be hardwired to 0b. This field indicates the support for use of unclaimed function numbers to extend the number of outstanding transactions allowed by logically combining unclaimed function numbers (called Phantom Functions) with the Tag identifier (see Section 2.2.6.2 of PCI Express Base Specification for a description of Tag Extensions). With every Function in an ARI Device, the Phantom Functions Supported field must be set to 00b. The remainder of this field description applies only to non-ARI Multi-Function Devices. This field indicates the number of most significant bits of the Function Number portion of Requester ID that are logically combined with the Tag identifier. Note: Phantom Function support for the function must be enabled by the Phantom Functions Enable field in the Device Control register before the Function is permitted to use the Function Number field in the Requester ID for Phantom Functions. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R Note: This register field is sticky. 00_B **NO_PHNATOM_FUNC**: No Function Number bits are used for Phantom Functions Multi-Function Devices are permitted to implement up to 8 independent functions. 01_B **MAX_1_PHANTOM_PER_FUNC**: The most significant bit of the Function number in Requester ID is used for Phantom Functions; a Multi-Function Device is permitted to implement Functions 0-3 Functions 0, 1, 2, and 3 are permitted to use Function Numbers 4, 5, 6, and 7 respectively as Phantom Functions. 10_B **MAX_3_PHANTOM_PER_FUNC**: The two most significant bits of Function Number in Requester ID are used for Phantom Functions; a Multi-Function Device is permitted to implement Functions 0-1 Function 0 is permitted to use Function Numbers 2, 4, and 6 for Phantom Functions. Function 1 is permitted to use Function Numbers 3, 5, and 7 as Phantom Functions. 11_B **SINGLE_FUNC_MAX_7_PHANTOM_FUNC**: All 3 bits of Function Number in Requester ID used for Phantom Functions The device must have a single Function 0 that is permitted to use all other Function Numbers as Phantom Functions.

(table continues...)

(continued)

Field	Bits	Type	Description
PCIE_CAP_EXT_TAG_SUPP	5	rw	**Extended Tag Field Supported** This bit, in combination with the 10-Bit Tag Requester Supported bit in the Device Capabilities 2 register, indicates the maximum supported size of the Tag field as a Requester. This bit must be set if the 10-Bit Tag Requester Supported bit is set. Note: 8-bit Tag field generation must be enabled by the Extended Tag Field Enable bit in the Device Control register of the Requester Function before 8-bit Tags can be generated by the Requester. See Section 2.2.6.2 of PCI Express Base Specification for interactions with enabling the use of 10-Bit Tags. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R Note: This register field is sticky. 0_B **_5b_TAG**: 5-bit Tag field supported 1_B **_8b_TAG**: 8-bit Tag field supported
PCIE_CAP_EP_L0S_ACCPT_LATENCY	8:6	rw	**Endpoint L0s Acceptable Latency** This field indicates the acceptable total latency that an Endpoint can withstand due to the transition from L0s state to the L0 state. It is essentially an indirect measure of the Endpoint's internal buffering. Power management software uses the reported L0s Acceptable Latency number to compare against the L0s exit latencies reported by all components comprising the data path from this Endpoint to the Root Complex Root Port to determine whether ASPM L0s entry can be used with no loss of performance. For functions other than Endpoints, this field is Reserved and the controller hardwires it to 000b. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R Note: This register field is sticky. 000_B **MAX_64NS**: Maximum of 64 ns 001_B **MAX_128NS**: Maximum of 128 ns 010_B **MAX_256NS**: Maximum of 256 ns 011_B **MAX_512NS**: Maximum of 512 ns 100_B **MAX_1US**: Maximum of 1 µs 101_B **MAX_2US**: Maximum of 2 µs 110_B **MAX_4US**: Maximum of 4 µs 111_B **NO_LIMIT**: No limit

(table continues...)

(continued)

Field	Bits	Type	Description
PCIE_CAP_EP_L1_ACCPT_LATENCY	11:9	rw	**Endpoint L1 Acceptable Latency** This field indicates the acceptable latency that an Endpoint can withstand due to the transition from L1 state to the L0 state. It is essentially an indirect measure of the Endpoint's internal buffering. Power management software uses the reported L1 Acceptable Latency number to compare against the L1 Exit Latencies reported (see below) by all components comprising the data path from this Endpoint to the Root Complex Root Port to determine whether ASPM L1 entry can be used with no loss of performance. For functions other than Endpoints, this field is Reserved and the controller hardwires it to 000b. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W (Sticky) else R(Sticky) Note: This register field is sticky. 000_B **MAX_1US**: Maximum of 1 µs 001_B **MAX_2US**: Maximum of 2 µs 010_B **MAX_4US**: Maximum of 4 µs 011_B **MAX_8US**: Maximum of 8 µs 100_B **MAX_16US**: Maximum of 16 µs 101_B **MAX_32US**: Maximum of 32 µs 110_B **MAX_64US**: Maximum of 64 µs 111_B **NO_LIMIT**: No limit
RSVDP_12	14:12	r	**Reserved for future use**
PCIE_CAP_ROLE_BASED_ERR_REPORT	15	rw	**Role-Based Error Reporting** When set, this bit indicates that the function implements the functionality originally defined in the Error Reporting ECN for PCI Express Base Specification, Revision 1.0a, and later incorporated into PCI Express Base Specification, Revision 1.1. This bit must be set by all functions conforming to the ECN, PCI Express Base Specification, Revision 1.1., or subsequent PCI Express Base Specification revisions. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R Note: This register field is sticky.
RSVDP_16	17:16	r	**Reserved for future use**
PCIE_CAP_CAP_SLOT_PWR_LMT_VALUE	25:18	rh	**Captured Slot Power Limit Value** Captured Slot Power Limit Value (Upstream Ports only). In combination with the Captured Slot Power Limit Scale value, specifies the upper limit on power available to the adapter. Power limit (in Watts) is calculated by multiplying the value in this field by the value in the Captured Slot Power Limit Scale field except when the Captured Slot Power Limit Scale field equals 00b (1.0x) and the Captured Slot Power Limit Value exceeds EFh, then alternative encodings are used (for more details, see section 7.5.3.9 of PCI Express Base Specification). This value is set by the Set_Slot_Power_Limit Message or hardwired to 00h (for more details, see section 6.9 of PCI Express Base Specification).

(table continues...)

(continued)

Field	Bits	Type	Description
PCIE_CAP_CAP_SLOT_PWR_LMT_SCALE	27:26	rh	**Captured Slot Power Limit Scale** Captured Slot Power Limit Scale (Upstream Ports only). Specifies the scale used for the Slot Power Limit Value. This value is set by the Set_Slot_Power_Limit Message or hardwired to 00b (for more details, see section 6.9 of PCI Express Base Specification). 00_B **_1_0X**: 1.0x 01_B **_0_1X**: 0.1x 10_B **_0_01X**: 0.01x 11_B **_0_001X**: 0.001x
PCIE_CAP_FLR_CAP	28	rw	**Function Level Reset Capability** A value of 1b indicates the function supports the optional Function Level Reset mechanism described in section 6.6.2 of the PCI Express Base Specification. This bit applies to Endpoints only. For all other function types the controller hardwires this bit to 0b. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R Note: This register field is sticky. Function Level Reset mechanism is not supported in this implementation
RSVDP_29	31:29	r	**Reserved for future use**

Table 868 Reset values of DEVICE_CAPABILITIES_REG

Reset	Reset value	Note
Kernel Reset	0000 XXXX XXXX XX00 1000 1111 1100 0001_B	

19.4.1.117 Device Control and Device Status Register

This register controls PCI Express device specific parameters and provides information about PCI Express device (function) specific parameters.

DEVICE_CONTROL_DEVICE_STATUS Offset address: 0004078_H

Device Control and Device Status Register Reset values see: Table 869

19 Peripheral Component Interconnect Express (PCIe)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_22										PCIE_CAP_TRANS_PENDING	PCIE_CAP_AUX_POWER_DETECTED	PCIE_CAP_UNSUPORTED_REQ_DETECTED	PCIE_CAP_FATAL_ERR_DETECTED	PCIE_CAP_NON_FATAL_ERR_DETECTED	PCIE_CAP_CORR_ERR_DETECTED
r										rh	rh	rw1ch	rw1ch	rw1ch	rw1ch

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	PCIE_CAP_MAX_READ_REQ_SIZE			PCIE_CAP_EN_NO_SNOOP	PCIE_CAP_AUX_POWER_PM_EN	PCIE_CAP_PHANTOM_FUNC_EN	PCIE_CAP_EXT_TAG_EN	PCIE_CAP_MAX_PAYLOAD_SIZE_CS			PCIE_CAP_EN_REL_ORDER	PCIE_CAP_UNSUPORT_REQ_REP_EN	PCIE_CAP_FATAL_ERR_REPORT_EN	PCIE_CAP_NON_FATAL_ERR_REPORT_EN	PCIE_CAP_CORR_ERR_REPORT_EN
r	rw			rw	rwh	rh	rh	rw			rw	rw	rw	rw	rw

Field	Bits	Type	Description
PCIE_CAP_CORR_ERR_REPORT_EN	0	rw	**Correctable Error Reporting Enable** This bit, in conjunction with other bits, controls sending ERR_COR Messages (for more details, see section 6.2.5, section 6.2.6, and section 6.2.10.2 of PCI Express Base Specification). For a Multi-Function device, this bit controls error reporting for each function from point-of-view of the respective function. For a Root Port, the reporting of correctable errors is internal to the root. No external ERR_COR Message is generated.
PCIE_CAP_NON_FATAL_ERR_REPORT_EN	1	rw	**Non-Fatal Error Reporting Enable** This bit, in conjunction with other bits, controls sending ERR_NONFATAL Messages (for more details, see section 6.2.5 and Section 6.2.6 of PCI Express Base Specification). For a Multi-Function Device, this bit controls error reporting for each function from point-of-view of the respective Function. For a Root Port, the reporting of Non-fatal errors is internal to the root. No external ERR_NONFATAL Message is generated.
PCIE_CAP_FATAL_ERR_REPORT_EN	2	rw	**Fatal Error Reporting Enable** This bit, in conjunction with other bits, controls sending ERR_FATAL Messages (for more details, see section 6.2.5 and section 6.2.6 of of PCI Express Base Specification). For a Multi-Function device, this bit controls error reporting for each function from point-of-view of the respective function. For a Root Port, the reporting of Fatal errors is internal to the root. No external ERR_FATAL Message is generated.

(table continues...)

(continued)

Field	Bits	Type	Description
PCIE_CAP_UNSUPPORT_REQ_REP_EN	3	rw	**Unsupported Request Reporting Enable** This bit, in conjunction with other bits, controls the signaling of Unsupported Request Errors by sending error Messages (for more details, see section 6.2.5 and section 6.2.6 of PCI Express Base Specification). For a Multi-Function Device, this bit controls error reporting for each Function from point-of-view of the respective Function.
PCIE_CAP_EN_REL_ORDER	4	rw	**Enable Relaxed Ordering** If this bit is set, the function is permitted to set the Relaxed Ordering bit in the Attributes field of transactions it initiates that do not require strong write ordering (for more details, see section 2.2.6.4 and section 2.4 of PCI Express Base Specification). For a function that never sets the Relaxed Ordering attribute in transactions it initiates as a Requester, the controller hardwires this bit to 0b. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
PCIE_CAP_MAX_PAYLOAD_SIZE_CS	7:5	rw	**Max_Payload_Size** This field sets maximum TLP payload size for the Function. As a Receiver, the Function must handle TLPs as large as the set value. As a Transmitter, the Function must not generate TLPs exceeding the set value. Permissible values that can be programmed are indicated by the Max_Payload_Size Supported field (PCIE_CAP_MAX_PAYLOAD_SIZE) in the Device Capabilities (DEVICE_CAPABILITIES_REG) register (for more details, see section 7.5.3.3 of PCI Express Base Specification). This field sets the maximum Read Request size for the function as a Requester. The function must not generate Read Requests with a size exceeding the set value. For Functions that support only the 128-byte max payload size, the controller hardwires this field to 000b. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W 000_B **MAX_128B_SIZE**: 128 bytes maximum Read Request size 001_B **MAX_256B_SIZE**: 256 bytes maximum Read Request size 010_B **MAX_512B_SIZE**: 512 bytes maximum Read Request size 011_B **MAX_1024B_SIZE**: 1024 bytes maximum Read Request size 100_B **MAX_2048B_SIZE**: 2048 bytes maximum Read Request size 101_B **MAX_4096B_SIZE**: 4096 bytes maximum Read Request size 110_B **RESERVED1**: RESERVED 111_B **RESERVED2**: RESERVED

(table continues...)

(continued)

Field	Bits	Type	Description
PCIE_CAP_EXT_TAG_EN	8	rh	**Extended Tag Field Enable** This bit, in combination with the 10-Bit Tag Requester Enable bit in the Device Control 2 register, determines how many Tag field bits a Requester is permitted to use. When the 10-Bit Tag Requester Enable bit is clear, • If the Extended Tag Field Enable bit is set, the function is permitted to use an 8-bit Tag field as a Requester • If the Extended Tag Field Enable bit is clear, the Function is to a 5-bit Tag field See section 2.2.6.2 of PCI Express Base Specification for required behavior when the 10-Bit Tag Requester Enable bit is set. If software changes the value of the Extended Tag Field Enable bit while the function has outstanding Non-Posted Requests, the result is undefined. For functions that do not implement this capability, the controller hardwires this bit to 0b. Note: The access attributes of this field are as follows: - Wire: DEVICE_CAPABILITIES_REG.PCIE_CAP_EXT_TAG_SUPP ? RW : RO - Dbi: DEVICE_CAPABILITIES_REG.PCIE_CAP_EXT_TAG_SUPP ? RW : RO
PCIE_CAP_PHANTOM_FUNC_EN	9	rh	**Phantom Functions Enable** For functions that do not implement this capability (Phantom function capability is not available in this implementation), this bit will be hardwired to 0b. This bit, in combination with the 10-Bit Tag Requester Enable bit in the Device Control 2 register, determines how many Tag field bits a Requester is permitted to use. When the 10-Bit Tag Requester Enable bit is clear, • If this bit is set, it enables a function to use unclaimed functions as Phantom functions to extend the number of outstanding transaction identifiers • If this bit is clear, the function is not allowed to use Phantom functions For more details, see section 2.2.6.2 of PCI Express Base Specification. Software should not change the value of this bit while the function has outstanding Non-Posted Requests; otherwise, the result is undefined. For functions that do not implement this capability, the controller hardwires this bit to 0b. Note: The access attributes of this field are as follows: • Wire: DEVICE_CAPABILITIES_REG.PCIE_CAP_PHANTOM_FUNC_SUPPORT ? RW : RO • Dbi: DEVICE_CAPABILITIES_REG.PCIE_CAP_PHANTOM_FUNC_SUPPORT ? RW : R

(table continues...)

(continued)

Field	Bits	Type	Description
PCIE_CAP_AUX_POWER_PM_EN	10	rwh	**Aux Power PM Enable** This bit is derived by sampling the sys_aux_pwr_det input. When set this bit, enables a function to draw Aux power independent of PME Aux power. Functions that require Aux power on legacy operating systems should continue to indicate PME Aux power requirements. Aux power is allocated as requested in the Aux_Current field of the Power Management Capabilities register (PMC), independent of the PME_En bit in the Power Management Control/Status register (PMCSR). For Multi-Function devices, a component is allowed to draw Aux power if at least one of the functions has this bit set. Note: Functions that consume Aux power must preserve the value of this sticky register when Aux power is available. In such functions, this bit is not modified by Conventional Reset. For functions that do not implement this capability, the controller hardwires this bit to 0b. Note: This register field is sticky.
PCIE_CAP_EN_NO_SNOOP	11	rw	**Enable No Snoop** If this bit is set, the function is permitted to Set the No Snoop bit in the Requester Attributes of transactions it initiates that do not require hardware enforced cache coherency (see section 2.2.6.5 in PCI Express Base Specification). Note: Setting this bit to 1b should not cause a function to set the No Snoop attribute on all transactions that it initiates. Even when this bit is set, a function is only permitted to set the No Snoop attribute on a transaction when it can guarantee that the address of the transaction is not stored in any cache in the system. The controller hardwires this bit 0b if a function would never set the No Snoop attribute in transactions it initiates. Note: The access attributes of this field are as follows: - Wire: R - Dbi: R
PCIE_CAP_MAX_READ_REQ_SIZE	14:12	rw	**Max_Read_Request_Size** This field sets the maximum Read Request size for the function as a Requester. The function must not generate Read Requests with a size exceeding the set value. For functions that do not generate Read Requests larger than 128 bytes and functions that do not generate Read Requests on their own behalf, the controller implements this field as Read Only (RO) with a value of 000b. 000_B **MAX_128B_SIZE**: 128 bytes maximum Read Request size 001_B **MAX_256B_SIZE**: 256 bytes maximum Read Request size 010_B **MAX_512B_SIZE**: 512 bytes maximum Read Request size 011_B **MAX_1024B_SIZE**: 1024 bytes maximum Read Request size 100_B **MAX_2048B_SIZE**: 2048 bytes maximum Read Request size 101_B **MAX_4096B_SIZE**: 4096 bytes maximum Read Request size 110_B **RESERVED1**: RESERVED 111_B **RESERVED2**: RESERVED
PCIE_CAP_CORR_ERR_DETECTED	16	rw1ch	**Correctable Error Detected** This bit indicates status of correctable errors detected. Errors are logged in this register regardless of whether error reporting is enabled or not in the Device Control register. For functions supporting Advanced Error Handling, errors are logged in this register regardless of the settings of the Correctable Error Mask register. Software write with 1 clears this bit-field.

(table continues...)

(continued)

Field	Bits	Type	Description
PCIE_CAP_NON_FATAL_ERR_DETECTED	17	rw1ch	**Non-Fatal Error Detected** This bit indicates status of Non-fatal errors detected. Errors are logged in this register regardless of whether error reporting is enabled or not in the Device Control register. For functions supporting Advanced Error Handling, errors are logged in this register regardless of the settings of the Uncorrectable Error Mask register. Software write with 1 clears this bit-field.
PCIE_CAP_FATAL_ERR_DETECTED	18	rw1ch	**Fatal Error Detected** This bit indicates status of Fatal errors detected. Errors are logged in this register regardless of whether error reporting is enabled or not in the Device Control register. For Functions supporting Advanced Error Handling, errors are logged in this register regardless of the settings of the Uncorrectable Error Mask register. Software write with 1 clears this bit-field.
PCIE_CAP_UNSUPPORTED_REQ_DETECTED	19	rw1ch	**Unsupported Request Detected** This bit indicates that the function received an Unsupported Request. Errors are logged in this register regardless of whether error reporting is enabled or not in the Device Control register. Software write with 1 clears this bit-field.
PCIE_CAP_AUX_POWER_DETECTED	20	rh	**AUX Power Detected** Functions that require Aux power report this bit as set if Aux power is detected by the function. .
PCIE_CAP_TRANS_PENDING	21	rh	**Transactions Pending** Endpoints: When set, this bit indicates that the function has issued Non-Posted Requests that have not been completed. A Function reports this bit cleared only when all outstanding Non-Posted Requests have completed or have been terminated by the Completion Timeout mechanism. Root and Switch Ports: The controller hardwires this bit to 0b.
RSVDP_22	31:22	r	**Reserved for future use**
0	15	r	**Reserved** Read as 0; should be written with 0.

Table 869 Reset values of DEVICE_CONTROL_DEVICE_STATUS

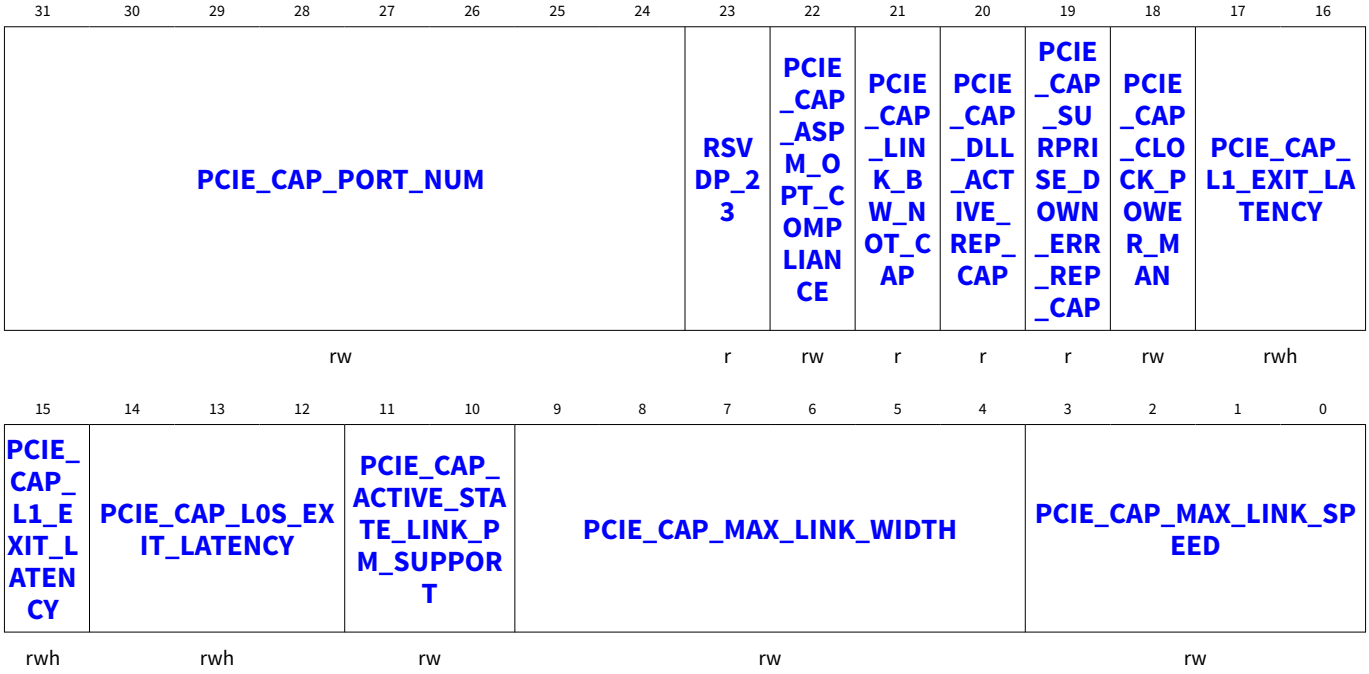
Reset	Reset value	Note
Kernel Reset	0000 0000 00XX 0000 0010 1XXX 0001 0000_B	

19.4.1.118 Link Capabilities Register

The Link Capabilities register identifies PCI Express Link specific capabilities.

LINK_CAPABILITIES_REG Offset address: $000407C_H$

Link Capabilities Register Kernel Reset value: $0047\ 6813_H$



Field	Bits	Type	Description
PCIE_CAP_MAX_LINK_SPEED	3:0	rw	**Max Link Speed** This field indicates the maximum Link speed of the associated Port. The encoded value specifies a bit location in the Supported Link Speeds Vector (in the Link Capabilities 2 register) that corresponds to the maximum Link speed. All encodings other than the defined encodings are reserved. Multi-Function Devices associated with an Upstream Port must report the same value in this field for all functions. In M-PCIe mode, the reset and dynamic values of this field are calculated by the controller. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R Note: This register field is sticky. 1_H **SUP_LINK_SPEED_FIELD_BIT_0**: Supported Link Speeds Vector field bit 0 2_H **SUP_LINK_SPEED_FIELD_BIT_1**: Supported Link Speeds Vector field bit 1 3_H **SUP_LINK_SPEED_FIELD_BIT_2**: Supported Link Speeds Vector field bit 2 4_H **SUP_LINK_SPEED_FIELD_BIT_3**: Supported Link Speeds Vector field bit 3 5_H **SUP_LINK_SPEED_FIELD_BIT_4**: Supported Link Speeds Vector field bit 4 6_H **SUP_LINK_SPEED_FIELD_BIT_5**: Supported Link Speeds Vector field bit 5 7_H **SUP_LINK_SPEED_FIELD_BIT_6**: Supported Link Speeds Vector field bit 6 **others,** Reserved

(table continues...)

(continued)

Field	Bits	Type	Description
PCIE_CAP_MAX_LINK_WIDTH	9:4	rw	**Maximum Link Width** This field indicates the maximum Link width (xN - corresponding to N Lanes) implemented by the component. This value is permitted to exceed the number of Lanes routed to the slot (Downstream Port), adapter connector (Upstream Port), or in the case of component-to-component connections, the actual wired connection width. All encodings other than the defined encodings are reserved. Multi-Function devices associated with an Upstream Port must report the same value in this field for all functions. In M-PCIe mode, the reset and dynamic values of this field are calculated by the controller. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R Note: This register field is sticky. 01_H **X1**: x1 02_H **X2**: x2 04_H **X4**: x4 08_H **X8**: x8 $0C_H$ **X12**: x12 10_H **X16**: x16 20_H **X32**: x32 **others,** Reserved
PCIE_CAP_ACTIVE_STATE_LINK_PM_SUPPORT	11:10	rw	**Active State Power Management (ASPM) Support** This field indicates the level of ASPM supported on the given PCI Express Link. For more details on ASPM support requirements, see section 5.4.1 of PCI Express Base Specification. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R Note: This register field is sticky. 00_B **NO_ASPM_SUP**: No ASPM Support 01_B **L0S_SUP**: L0s Supported 10_B **L1_SUP**: L1 Supported 11_B **L0S_L1_SUP**: L0s and L1 Supported

(table continues...)

(continued)

Field	Bits	Type	Description
PCIE_CAP_L0S_EXIT_LATENCY	14:12	rwh	**L0s Exit Latency** This field indicates the L0s exit latency for the given PCI Express Link. The value reported indicates the length of time this Port requires to complete transition from L0s to L0. If L0s is not supported, the value is undefined; however, see the Implementation Note "Potential Issues With Legacy Software When L0s is Not Supported" in section 5.4.1.1 of PCI Express Base Specification for the recommended value. Note: Exit latencies may be influenced by PCI Express reference clock configuration depending upon whether a component uses a common or separate reference clock. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R Note: This register field is sticky. 000_B **LESS_THAN_64NS**: Less than 64 ns 001_B **_64NS_TO_128NS**: 64 ns to less than 128 ns 010_B **_128NS_TO_256NS**: 128 ns to less than 256 ns 011_B **_256NS_TO_512NS**: 256 ns to less than 512 ns 100_B **_512NS_TO_1US**: 512 ns to less than 1 µs 101_B **_1US_TO_2US**: 1 µs to less than 2 µs 110_B **_2US_TO_4US**: 2 µs to 4 µs 111_B **GREATER_THAN_4US**: More than 4 µs
PCIE_CAP_L1_EXIT_LATENCY	17:15	rwh	**L1 Exit Latency** This field indicates the L1 exit latency for the given PCI Express Link. The value reported indicates the length of time this Port requires to complete transition from ASPM L1 to L0. If ASPM L1 is not supported, the value is undefined. Note: Exit latencies may be influenced by PCI Express reference clock configuration depending upon whether a component uses a common or separate reference clock. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R Note: This register field is sticky. 000_B **LESS_THAN_1US**: Less than 1 µs 001_B **_1US_TO_2US**: 1 µs to less than 2 µs 010_B **_2US_TO_4US**: 2 µs to less than 4 µs 011_B **_4US_TO_8US**: 4 µs to less than 8 µs 100_B **_8US_TO_16US**: 8 µs to less than 16 µs 101_B **_16US_TO_32US**: 16 µs to less than 32 µs 110_B **_32US_TO_64US**: 32 µs to 64 µs 111_B **GREATER_THAN_64US**: More than 64 µs

(table continues...)

(continued)

Field	Bits	Type	Description
PCIE_CAP_CLOCK_POWER_MAN	18	rw	**Clock Power Management** For Upstream Ports, a value of 1b in this bit indicates that the component tolerates the removal of any reference clock(s) via the "clock request" (CLKREQ#) mechanism when the Link is in the L1 and L2/L3 Ready Link states. A value of 0b indicates the component does not have this capability and that reference clock(s) must not be removed in these Link states. L1 PM Substates defines other semantics for the CLKREQ# signal, which are managed independently of Clock Power Management. This Capability is applicable only in form factors that support "clock request" (CLKREQ#) capability. The reset value of this bit field is decided by the value of PM_CTRL.CLKPMEN. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky.
PCIE_CAP_SURPRISE_DOWN_ERR_REP_CAP	19	r	**Surprise Down Error Reporting Capable** For a Downstream Port, this bit must be set if the component supports the optional capability of detecting and reporting a Surprise Down error condition. For Upstream Ports and components that do not support this optional capability, the controller hardwires this bit to 0b. Note: This register field is sticky.
PCIE_CAP_DLL_ACTIVE_REP_CAP	20	r	**Data Link Layer Link Active Reporting Capable** For a Downstream Port, the controller hardwires this bit to 1b if the component supports the optional capability of reporting the DL_Active state of the Data Link Control and Management State Machine. For a hot-plug capable Downstream Port (as indicated by the Hot-Plug Capable bit of the Slot Capabilities register) or a Downstream Port that supports Link speeds greater than 5.0 GT/s, the controller hardwires this bit to 1b. For Upstream Ports and components that do not support this optional capability, the controller hardwires this bit to 0b.
PCIE_CAP_LINK_BW_NOT_CAP	21	r	**Link Bandwidth Notification Capability** A value of 1b indicates support for the Link Bandwidth Notification status and interrupt mechanisms. This capability is required for all Root Ports and Switch Downstream Ports supporting Links wider than x1 and/or multiple Link speeds. This field is not applicable and is Reserved for Endpoints, PCI Express to PCI/PCI-X bridges, and Upstream Ports of Switches. For functions that do not implement the Link Bandwidth Notification Capability the controller hardwires this bit to 0b. Note: This register field is sticky.
PCIE_CAP_ASPM_OPT_COMPLIANCE	22	rw	**ASPM Optionality Compliance** This field must be set to 1b in all functions. Components implemented against certain earlier versions of this specification will have this bit set to 0b. Software is permitted to use the value of this bit to help determine whether to enable ASPM or whether to run ASPM compliance tests. Note: The access attributes of this field are as follows: - Wire: HWINIT (hardware initialized) - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R
RSVDP_23	23	r	**Reserved for future use**

(table continues...)

(continued)

Field	Bits	Type	Description
PCIE_CAP_PORT_NUM	31:24	rw	**Port Number** This field indicates the PCI Express Port number for the given PCI Express Link. Multi-Function Devices associated with an Upstream Port must report the same value in this field for all functions. Note: The access attributes of this field are as follows: - Wire: HWINIT - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R

19.4.1.119 Link Control and Link Status Register

This register controls and provides information about PCI Express Link specific parameters.

LINK_CONTROL_LINK_STATUS_REG Offset address: 0004080_H

Link Control and Link Status Register Reset values see: Table 870

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PCIE_CAP_LINK_AUTO_BW_STATUS	PCIE_CAP_LINK_BW_MAN_STATUS	PCIE_CAP_DLL_ACTIVE	PCIE_CAP_SLOT_CLK_CONFIG	PCIE_CAP_LINK_TRAINING	RSVDP_26	PCIE_CAP_NEGO_LINK_WIDTH						PCIE_CAP_LINK_SPEED			
r	r	r	rw	rh	r	rh						rh			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PCIE_CAP_DRS_SIGNALING_CONTROL		RSVDP_12		PCIE_CAP_LINK_AUTO_BW_INT_EN	PCIE_CAP_LINK_BW_MAN_INT_EN	PCIE_CAP_HW_AUTO_WIDTH_DISABLE	PCIE_CAP_EN_CLK_POWER_MAN	PCIE_CAP_EXTENDED_SYNCH	PCIE_CAP_COMMON_CLK_CONFIG	PCIE_CAP_RETRAIN_LINK	PCIE_CAP_LINK_DISABLE	PCIE_CAP_RCB	RSVDP_2	PCIE_CAP_ACTIVE_STATE_LINK_PM_CONTROL	
r		r		rwh	rwh	rw	rwh	rwh	rw	rwh	rwh	rw	r	rw	

Field	Bits	Type	Description
PCIE_CAP_ACTIVE_STATE_LINK_PM_CONTROL	1:0	rw	**Active State Power Management (ASPM) Control** This field controls the level of ASPM enabled on the given PCI Express Link. See section 5.4.1.3 of PCI Express Base Specification for requirements on when and how to enable ASPM. Note: "L0s Entry Enabled" enables the Transmitter to enter L0s. If L0s is supported, the Receiver must be capable of entering L0s even when the Transmitter is disabled from entering L0s (00b or 10b). ASPM L1 must be enabled by software in the Upstream component on a Link prior to enabling ASPM L1 in the Downstream component on that Link. When disabling ASPM L1, software must disable ASPM L1 in the Downstream component on a Link prior to disabling ASPM L1 in the Upstream component on that Link. ASPM L1 must only be enabled on the Downstream component if both components on a Link support ASPM L1. Software must not enable L0s in either direction on a given Link unless components on both sides of the Link each support L0s; otherwise, the result is undefined. 00_B **DISABLED**: Disabled 01_B **L0S_ENTRY_EN**: L0s Entry Enabled 10_B **L1_ENTRY_En**: L1 Entry Enabled 11_B **L0S_L1_ENTRY_EN**: L0s and L1 Entry Enabled
RSVDP_2	2	r	**Reserved for future use**
PCIE_CAP_RCB	3	rw	**Read Completion Boundary (RCB)** Root Ports: Indicates the RCB value for the Root Port. Refer to section 2.3.1.1 of PCI Express Base Specification for the definition of the parameter RCB. The controller hardwires this bit for a Root Port and returns its RCB support capabilities. Endpoints and Bridges: Optionally set by configuration software to indicate the RCB value of the Root Port Upstream from the Endpoint or Bridge. Refer to Section 2.3.1.1 of PCI Express Base Specification for the definition of the parameter RCB is same as Root Port. Configuration software must only set this bit if the Root Port Upstream from the Endpoint or Bridge reports an RCB value of 128 bytes (a value of 1b in the Read Completion Boundary bit). For functions that do not implement this feature, the controller hardwires this bit to 0b. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W 0_B **_64B**: 64 byte 1_B **_128B**: 128 byte
PCIE_CAP_LINK_DISABLE	4	rwh	**Link Disable** This bit disables the Link by directing the LTSSM to the Disabled state when set; this bit is Reserved on Endpoints, PCI Express to PCI/PCI-X bridges, and Upstream Ports of Switches. Writes to this bit are immediately reflected in the value read from the bit, regardless of actual Link state. After clearing this bit, software must honor timing requirements defined in Section 6.6.1 with respect to the first Configuration Read following a Conventional Reset. Note: The access attributes of this field are as follows: - Wire: dsp=1 ? RW : RO - Dbi: dsp=1? RW : RO

(table continues...)

(continued)

Field	Bits	Type	Description
PCIE_CAP_RETRAIN_LINK	5	rwh	**Retrain Link** A write of 1b to this bit initiates Link retraining by directing the Physical Layer LTSSM to the Recovery state. If the LTSSM is already in Recovery or Configuration, re-entering Recovery is permitted but not required. Reads of this bit always return 0b. It is permitted to write 1b to this bit while simultaneously writing modified values to other fields in this register. If the LTSSM is not already in Recovery or Configuration, the resulting Link training must use the modified values. If the LTSSM is already in Recovery or Configuration, the modified values are not required to affect the Link training that's already in progress. This bit is not applicable and is Reserved for Endpoints, PCI Express to PCI/PCI-X bridges, and Upstream Ports of Switches. This bit always returns 0b when read.
PCIE_CAP_COMMON_CLK_CONFIG	6	rw	**Common Clock Configuration** When set, this bit indicates that this component and the component at the opposite end of this Link are operating with a distributed common reference clock. A value of 0b indicates that this component and the component at the opposite end of this Link are operating with asynchronous reference clock. Components utilize this common clock configuration information to report the correct L0s and L1 Exit Latencies. After changing the value in this bit in both components on a Link, software must trigger the Link to retrain by writing a 1b to the Retrain Link bit of the Downstream Port.
PCIE_CAP_EXTENDED_SYNCH	7	rwh	**Extended Synch** When set, this bit forces the transmission of additional Ordered Sets when exiting the L0s state (see section 4.2.4.5 of PCI Express Base Specification) and when in the Recovery state (see section 4.2.6.4.1 of PCI Express Base Specification). This mode provides external devices (for example, logic analyzers) monitoring the Link time to achieve bit and Symbol lock before the Link enters the L0 state and resumes communication. For Multi-Function devices if any function has this bit set, then the component must transmit the additional Ordered Sets when exiting L0s or when in Recovery.

(table continues...)

(continued)

Field	Bits	Type	Description
PCIE_CAP_EN_CLK_POWER_MAN	8	rwh	**Enable Clock Power Management** Applicable only for Upstream Ports and with form factors that support a "Clock Request" (CLKREQ#) mechanism, this bit operates as follows: . For Downstream Ports and components that do not support Clock Power Management (as indicated by a 0b value in the Clock Power Management bit of the Link Capabilities register), the controller hardwires this bit to 0b. The write value is gated with the PCIE_CAP_CLOCK_POWER_MAN field in LINK_CAPABILITIES_REG. Note: The access attributes of this field are as follows: • Wire: LINK_CAPABILITIES_REG.PCIE_CAP_CLOCK_POWER_MAN ? RWS : ROS • Dbi: LINK_CAPABILITIES_REG.PCIE_CAP_CLOCK_POWER_MAN ? RWS : ROS Note: This register field is sticky 0_B **PM_DISABLE_LOW_CLKREQ**: Clock power management is disabled and device must hold CLKREQ# signal low 1_B **CLKREQ_IN_USE**: When this bit is set, the device is permitted to use CLKREQ# signal to power manage Link clock according to protocol defined in appropriate form factor specification
PCIE_CAP_HW_AUTO_WIDTH_DISABLE	9	rw	**Hardware Autonomous Width Disable** When set, this bit disables hardware from changing the Link width for reasons other than attempting to correct unreliable Link operation by reducing Link width. For components that do not implement the ability autonomously to change Link width, the ciontroller hardwires this bit to 0b. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
PCIE_CAP_LINK_BW_MAN_INT_EN	10	rwh	**Link Bandwidth Management Interrupt Enable** When set, this bit enables the generation of an interrupt to indicate that the Link Bandwidth Management Status bit has been set. The write value is gated with the PCIE_CAP_LINK_BW_NOT_CAP field in LINK_CAPABILITIES_REG. This bit is not applicable and is Reserved for Endpoints, PCI Express-to-PCI/PCI-X bridges, and Upstream Ports of Switches. For functions that do not implement the Link Bandwidth Notification Capability, the controller hardwires this bit to 0b. Note: The access attributes of this field are as follows: • Wire: LINK_CAPABILITIES_REG.PCIE_CAP_LINK_BW_NOT_CAP ? RW : RO • Dbi: LINK_CAPABILITIES_REG.PCIE_CAP_LINK_BW_NOT_CAP ? RW : RO

(table continues...)

(continued)

Field	Bits	Type	Description
PCIE_CAP_LINK_AUTO_BW_INT_EN	11	rwh	**Link Autonomous Bandwidth Management Interrupt Enable** When set, this bit enables the generation of an interrupt to indicate that the Link Autonomous Bandwidth Status bit has been set. The write value is gated with the PCIE_CAP_LINK_BW_NOT_CAP field in LINK_CAPABILITIES_REG. This bit is not applicable and is Reserved for Endpoints, PCI Express-to-PCI/PCI-X bridges, and Upstream Ports of Switches. For functions that do not implement the Link Bandwidth Notification Capability, the controller hardwires this bit to 0b. Note: The access attributes of this field are as follows: • Wire: LINK_CAPABILITIES_REG.PCIE_CAP_LINK_BW_NOT_CAP ? RW : RO • Dbi: LINK_CAPABILITIES_REG.PCIE_CAP_LINK_BW_NOT_CAP ? RW : RO
RSVDP_12	13:12	r	**Reserved for future use**
PCIE_CAP_DRS_SIGNALING_CONTROL	15:14	r	**DRS Signaling Control** Indicates the mechanism used to report reception of a DRS message. Must be implemented for Downstream Ports with the DRS Supported bit Set in the Link Capabilities 2 Register. Encodings are: • If DRS Supported is set, receiving a DRS Message will set DRS Message Received in the Link Control 2 and Status 2 Register Register but will otherwise have no effect • If the DRS Message Received bit in the Link Control 2 and Status 2 Register Register transitions from 0 to 1, and either MSI or MSI-X is enabled, an MSI or MSI-X interrupt is generated using the vector in Interrupt Message Number (section 7.5.3.2) Behavior is undefined if this field is set to 10b as the FRS capability is not Supported. Behavior is undefined if this field is set to 11b. For Downstream Ports with the DRS Supported bit clear in the Link Capabilities 2 register, the controller hardwires this field to 00b. This field is Reserved for Upstream Ports. 00_B **DRS_NOT_REPORTED**: DRS not Reported 01_B **DRS_INTRRUPT_EN**: DRS Interrupt Enabled 10_B **FRS_EN**: DRS to FRS Signaling Enabled **others,** Reserved

(table continues...)

(continued)

Field	Bits	Type	Description
PCIE_CAP_LINK_SPEED	19:16	rh	**Current Link Speed** This field indicates the negotiated Link speed of the given PCI Express Link. The encoded value specifies a bit location in the Supported Link Speeds Vector (in the Link Capabilities 2 register) that corresponds to the current Link speed. All encodings other than the defined encodings are reserved. The value in this field is undefined when the Link is not up. 1_H **SUP_LINK_SPEED_FIELD_BIT_0**: Supported Link Speeds Vector field bit 0 2_H **SUP_LINK_SPEED_FIELD_BIT_1**: Supported Link Speeds Vector field bit 1 3_H **SUP_LINK_SPEED_FIELD_BIT_2**: Supported Link Speeds Vector field bit 2 4_H **SUP_LINK_SPEED_FIELD_BIT_3**: Supported Link Speeds Vector field bit 3 5_H **SUP_LINK_SPEED_FIELD_BIT_4**: Supported Link Speeds Vector field bit 4 6_H **SUP_LINK_SPEED_FIELD_BIT_5**: Supported Link Speeds Vector field bit 5 7_H **SUP_LINK_SPEED_FIELD_BIT_6**: Supported Link Speeds Vector field bit 6 **others,** Reserved
PCIE_CAP_NEGO_LINK_WIDTH	25:20	rh	**Negotiated Link Width** This field indicates the negotiated width of the given PCI Express Link. All encodings other than the defined encodings are reserved. The value in this field is undefined when the Link is not up. 01_H **X1**: x1 02_H **X2**: x2 04_H **X4**: x4 08_H **X8**: x8 $0C_H$ **X12**: x12 10_H **X16**: x16 20_H **X32**: x32 **others,** Reserved
RSVDP_26	26	r	**Reserved for future use**
PCIE_CAP_LINK_TRAINING	27	rh	**Link Training** This read-only bit indicates that the Physical Layer LTSSM is in the Configuration or Recovery state, or that 1b was written to the Retrain Link bit but Link training has not yet begun. Hardware clears this bit when the LTSSM exits the Configuration/Recovery state. This bit is not applicable and Reserved for Endpoints, PCI Express to PCI/PCI-X bridges, and Upstream Ports of Switches, and the controller hardwires it to 0b. Note: The access attributes of this field are as follows: - Wire: RSVDP - Dbi: R

(table continues...)

(continued)

Field	Bits	Type	Description
PCIE_CAP_SLOT_CLK_CONFIG	28	rw	**Slot Clock Configuration** This bit indicates that the component uses the same physical reference clock that the platform provides on the connector. If the device uses an independent clock irrespective of the presence of a reference clock on the connector, this bit must be clear. For a Multi-Function Device, each Function must report the same value for this bit. Note: The access attributes of this field are as follows: - Wire: HWINIT - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R
PCIE_CAP_DLL_ACTIVE	29	r	**Data Link Layer Link Active** This bit indicates the status of the Data Link Control and Management State Machine. It returns a 1b to indicate the DL_Active state, 0b otherwise. This bit must be implemented if the Data Link Layer Link Active Reporting Capable bit is 1b. Otherwise, the controller hardwires it to 0b.
PCIE_CAP_LINK_BW_MAN_STATUS	30	r	**Link Bandwidth Management Status** This bit is set by hardware to indicate that either of the following has occurred without the Port transitioning through DL_Down status: • A Link retraining has completed following a write of 1b to the Retrain Link bit. Note: This bit is set following any write of 1b to the Retrain Link bit, including when the Link is in the process of retraining for some other reason • Hardware has changed Link speed or width to attempt to correct unreliable Link operation, either through an LTSSM timeout or a higher level process. This bit must be set if the Physical Layer reports a speed or width change was initiated by the Downstream component that was not indicated as an autonomous change. This bit is not applicable and is Reserved for Endpoints, PCI Express-to-PCI/PCI-X bridges, and Upstream Ports of Switches. The default value of this bit is 0b. The write value is gated with the PCIE_CAP_LINK_BW_NOT_CAP field in LINK_CAPABILITIES_REG. Note: The access attributes of this field are as follows: - Wire: RSVDP - Dbi: R
PCIE_CAP_LINK_AUTO_BW_STATUS	31	r	**Link Autonomous Bandwidth Status** This bit is set by hardware to indicate that hardware has autonomously changed Link speed or width, without the Port transitioning through DL_Down status, for reasons other than to attempt to correct unreliable Link operation. This bit must be set if the Physical Layer reports a speed or width change was initiated by the Downstream component that was indicated as an autonomous change. The write value is gated with the PCIE_CAP_LINK_BW_NOT_CAP field in LINK_CAPABILITIES_REG. This bit is not applicable and is Reserved for Endpoints, PCI Express-to-PCI/PCI-X bridges, and Upstream Ports of Switches. For functions that do not implement the Link Bandwidth Notification Capability, the controller hardwires this bit to 0b. Note: The access attributes of this field are as follows: - Wire: RSVDP - Dbi: R

Table 870 Reset values of LINK_CONTROL_LINK_STATUS_REG

Reset	Reset value	Note
Kernel Reset	0001 X0XX XXXX XXXX 0000 XX0X 00XX 0000_B	

19.4.1.120 Device Capabilities 2 Register

This register identifies PCI Express device specific capabilities; in addition to the Device Capabilities Register.

DEVICE_CAPABILITIES2_REG Offset address: 0004094_H

Device Capabilities 2 Register Kernel Reset value: $000C\ 081F_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	RSVDP_24							0				PCIE_CAP_OBFF_SUPPORT		PCIE_CAP2_10_BIT_TAG_REQ_SUPPORT	PCIE_CAP2_10_BIT_TAG_COMP_SUPPORT
r	r							r				rw		r	r

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PCIE_CAP2_LN_SYS_CLS		PCIE_CAP_TPH_CMPLT_SUPPORT_1	PCIE_CAP_TPH_CMPLT_SUPPORT_0	PCIE_CAP_LTR_SUPP	PCIE_CAP_NO_RO_EN_PR2PR_PAR	PCIE_CAP_128_CAS_CPL_SUPP	PCIE_CAP_64_ATOMIC_CPL_SUPP	PCIE_CAP_32_ATOMIC_CPL_SUPP	PCIE_CAP_ATOMIC_ROUTING_SUPP	PCIE_CAP_ARI_FORWARD_SUPPORT	PCIE_CAP_CPL_TIMEOUT_DISABLE_SUPPORT	PCIE_CAP_CPL_TIMEOUT_RANGE			
r		r	r	rw	r	r	r	r	r	r	r	r			

Field	Bits	Type	Description
PCIE_CAP_CPL_TIMEOUT_RANGE	3:0	r	**Completion Timeout Ranges Supported** This field indicates device Function support for the optional Completion Timeout programmability mechanism. This mechanism allows system software to modify the Completion Timeout value. This field is applicable only to Root Ports, Endpoints that issue Requests on their own behalf, and PCI Express to PCI/PCI-X Bridges that take ownership of Requests issued on PCI Express. For all other Functions this field is Reserved and must be hardwired to 0000b. Four time value ranges are defined: • Range A: 50 µs to 10 ms • Range B: 10 ms to 250 ms • Range C: 250 ms to 4 s • Range D: 4 s to 64 s Bits are set according to the list below to show timeout value ranges supported. All encodings other than the defined encodings are reserved. It is strongly recommended that the Completion Timeout mechanism not expire in less than 10 ms. 0_H **NOT_SUP**: Completion Timeout programming not supported, the Function must implement a timeout value in the range 50 µs to 50 ms 1_H **RANGE_A**: Range A 2_H **RANGE_B**: Range B 3_H **RANGE_A_B**: Ranges A and B 6_H **RANGE_B_C**: Ranges B and C 7_H **RANGE_A_B_C**: Ranges A, B, and C E_H **RANGE_B_C_D**: Ranges B, C, and D F_H **RANGE_A_B_C_D**: Ranges A, B, C, and D
PCIE_CAP_CPL_TIMEOUT_DISABLE_SUPPORT	4	r	**Completion Timeout Disable Supported** A value of 1b indicates support for the Completion Timeout Disable mechanism. The Completion Timeout Disable mechanism is required for Endpoints that issue Requests on their own behalf and PCI Express to PCI/PCI-X Bridges that take ownership of Requests issued on PCI Express. This mechanism is optional for Root Ports. For all other Functions this field is Reserved and the controller hardwires this bit to 0b.
PCIE_CAP_ARI_FORWARD_SUPPORT	5	r	**ARI Forwarding Supported** Alternate routing ID capability is not available in this implementation. Applicable only to Switch Downstream Ports and Root Ports; must be 0b for other Function types. This bit must be set to 1b if a Switch Downstream Port or Root Port supports this optional capability. For more details, see section 6.13 of PCI Express Base Specification.
PCIE_CAP_ATOMIC_ROUTING_SUPP	6	r	**AtomicOp Routing Supported** Applicable only to Switch Upstream Ports, Switch Downstream Ports, and Root Ports; must be 0b for other Function types. This bit must be set to 1b if the Port supports this optional capability. For more details, see section 6.15 of PCI Express Base Specification.

(table continues...)

(continued)

Field	Bits	Type	Description
PCIE_CAP_32_ATOMIC_CPL_SUPP	7	r	**32-bit AtomicOp Completer Supported** Applicable to Functions with Memory Space BARs as well as all Root Ports; must be 0b otherwise. Includes FetchAdd, Swap, and CAS AtomicOps. This bit must be set to 1b if the Function supports this optional capability. For more details on additional RC requirements, see section 6.15.3.1 of PCI Express Base Specification.
PCIE_CAP_64_ATOMIC_CPL_SUPP	8	r	**64-bit AtomicOp Completer Supported** Applicable to Functions with Memory Space BARs as well as all Root Ports; must be 0b otherwise. Includes FetchAdd, Swap, and CAS AtomicOps. This bit must be set to 1b if the Function supports this optional capability. For more details on additional RC requirements, see section 6.15.3.1 of PCI Express Base Specification.
PCIE_CAP_128_CAS_CPL_SUPP	9	r	**128-bit CAS Completer Supported** Applicable to Functions with Memory Space BARs as well as all Root Ports; must be 0b otherwise. This bit must be set to 1b if the Function supports this optional capability. For more details, see section 6.15 of PCI Express Base Specification.
PCIE_CAP_NO_RO_EN_PR2PR_PAR	10	r	**No RO-enabled PR-PR Passing** If this bit is set, the routing element never carries out the passing permitted by Table 2-39 of PCI Express Base Specification entry A2b that is associated with the Relaxed Ordering Attribute field being Set. This bit applies only for Switches and RCs that support peer-to-peer traffic between Root Ports. This bit applies only to Posted Requests being forwarded through the Switch or RC and does not apply to traffic originating or terminating within the Switch or RC itself. All Ports on a Switch or RC must report the same value for this bit. For all other functions, this bit must be 0b.
PCIE_CAP_LTR_SUPP	11	rw	**LTR Mechanism Supported** A value of 1b indicates support for the optional Latency Tolerance Reporting (LTR) mechanism. Root Ports, Switches and Endpoints are permitted to implement this capability. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W(Sticky) else R(Sticky) Note: This register field is sticky.
PCIE_CAP_TPH_CMPLT_SUPPORT_0	12	r	**TPH Completer Supported Bit 0** TLP Processing Hints(TPH) is not supported in this implementation. Value of this bit along with TPH Completer Supported Bit 1 indicates Completer support for TPH or Extended TPH. Applicable only to Root Ports and Endpoints. For all other Functions, this field is Reserved. For more details, see section 6.17 of PCI Express Base Specification. 0_B **TPH_EXT_TPH_CMPL_NOT_SUP_OR_RSVD**: TPH and Extended TPH Completer not supported (if TPH Completer Supported Bit is 0) or Reserved ((if TPH Completer Supported Bit is 1) 1_B **TPH_SUP_EXT_TPH_NOT_SUP_OR_BOTH_SUPP**: TPH Completer supported; Extended TPH Completer not supported(if TPH Completer Supported Bit is 0) or Both TPH and Extended TPH Completer supported ((if TPH Completer Supported Bit is 1)
PCIE_CAP_TPH_CMPLT_SUPPORT_1	13	r	**TPH Completer Supported Bit 1** TLP Processing Hints(TPH) is not supported in this implementation

(table continues...)

(continued)

Field	Bits	Type	Description
PCIE_CAP2_LN_SYS_CLS	15:14	r	**LN System CLS** Applicable only to Root Ports and RCRBs; must be 00b for all other Function types. This field indicates if the Root Port or RCRB supports LN protocol as an LN Completer, and if so, what cacheline size is in effect. All encodings other than the defined encodings are reserved. The above feature is not supported in this implementation Note: This register field is sticky. 00_B **LN_COMPLETER_NO_SUP**: LN Completer either not supported or not in effect 01_B **LN_COMPLETER_64B_CACHE**: LN Completer with 64-byte cachelines in effect 10_B **LN_COMPLETER_128B_CACHE**: LN Completer with 128-byte cachelines in effect **others,** Reserved
PCIE_CAP2_10_BIT_TAG_COMP_SUPPORT	16	r	**10-Bit Tag Completer Supported** If this bit is set, the Function supports 10-Bit Tag Completer capability; otherwise, the Function does not. For more details, see section 2.2.6.2. of PCI Express Base Specification.
PCIE_CAP2_10_BIT_TAG_REQ_SUPPORT	17	r	**10-Bit Tag Requester Supported** If this bit is set, the Function supports 10-Bit Tag Requester capability; otherwise, the Function does not. This bit must not be set if the 10-Bit Tag Completer Supported bit is clear. Note: 10-Bit Tag field generation must be enabled by the 10-Bit Tag Requester Enable bit in the Device Control 2 register of the Requester Function before 10-Bit Tags can be generated by the Requester. For more details, see section 2.2.6.2. of PCI Express Base Specification.
PCIE_CAP_OBFF_SUPPORT	19:18	rw	**OBFF Supported** This field indicates if OBFF is supported and, if so, what signaling mechanism is used. The value reported in this field must indicate support for WAKE# signaling only if: • for a Downstream Port, driving the WAKE# signal for OBFF is supported and the connector or component connected Downstream is known to receive that same WAKE# signal • for an Upstream Port, receiving the WAKE# signal for OBFF is supported and, if the component is on an add-in-card, that the component is connected to the WAKE# signal on the connector. Root Ports, Switch Ports, and Endpoints are permitted to implement this capability. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W(Sticky) else R(Sticky) Note: This register field is sticky. 00_B **NOT_SUP**: OBFF Not Supported 01_B **SUP_USING_MSG**: OBFF supported using Message signaling only 10_B **SUP_USING_WAKE**: OBFF supported using WAKE# signaling only 11_B **SUP_USING_MSG_AND_WAKE**: OBFF supported using WAKE# and Message signaling
RSVDP_24	30:24	r	**Reserved for future use**

(table continues...)

(continued)

Field	Bits	Type	Description
0	23:20, 31	r	**Reserved** Read as all 0's; should be written with all 0's.

19.4.1.121 Device Control 2 and Status 2 Register

This register controls PCI Express device specific parameters and provides information about PCI Express device (function) specific parameters; in addition to the Device Control and Device Status Register.

DEVICE_CONTROL2_DEVICE_STATUS2_REG Offset address: 0004098_H

Device Control 2 and Status 2 Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	PCIE_CAP_OBFF_EN		0		PCIE_CAP_LTR_EN	0				PCIE_CAP_ARI_FORWARD_SUPPORT_CS	PCIE_CAP_CPL_TIMEOUT_DISABLE	PCIE_CAP_CPL_TIMEOUT_VALUE			
r	rw		r		rw	r				r	rw	rw			

Field	Bits	Type	Description
PCIE_CAP_CPL_TIMEOUT_VALUE	3:0	rw	**Completion Timeout Value** In device Functions that support Completion Timeout programmability, this field allows system software to modify the Completion Timeout value. This field is applicable to Root Ports, Endpoints that issue Requests on their own behalf. Functions that support Completion Timeout programmability must support the values given below corresponding to the programmability ranges indicated in the Completion Timeout Ranges Supported field. All encodings other than the defined encodings are reserved. It is strongly recommended that the Completion Timeout mechanism not expire in less than 10 ms. Software is permitted to change the value in this field at any time. For Requests already pending when the Completion Timeout Value is changed, hardware is permitted to use either the new or the old value for the outstanding Requests, and is permitted to base the start time for each Request either on when this value was changed or on when each request was issued. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W 0_H **DEFAULT**: Default range: 50 µs to 50 ms 1_H **_50_TO_100_US**: 50 µs to 100 µs 2_H **_1_TO_10_MS**: 1 ms to 10 ms 5_H **_16_TO_55_MS**: 16 ms to 55 ms 6_H **_65_TO_210_MS**: 65 ms to 210 ms 9_H **_260_TO_900_MS**: 260 ms to 900 ms A_H **_1_TO_3_5_S**: 1 s to 3.5 s D_H **_4_TO_13_S**: 4 s to 13 s E_H **_17_TO_64_S**: 17 s to 64 s **others,** Reserved
PCIE_CAP_CPL_TIMEOUT_DISABLE	4	rw	**Completion Timeout Disable** When set, this bit disables the Completion Timeout mechanism. This bit is required for all Functions that support the Completion Timeout Disable Capability. Functions that do not support this optional capability are permitted to hardwire this bit to 0b Software is permitted to set or clear this bit at any time. When set, the Completion Timeout detection mechanism is disabled. If there are outstanding Requests when the bit is cleared, it is permitted but not required for hardware to apply the completion timeout mechanism to the outstanding Requests. If this is done, it is permitted to base the start time for each Request on either the time this bit was cleared or the time each Request was issued.
PCIE_CAP_ARI_FORWARD_SUPPORT_CS	5	r	**ARI Forwarding Enable** Alternate routing ID capability is not available in this implementation. When set, the Downstream Port disables its traditional Device Number field being 0 enforcement when turning a Type 1 Configuration Request into a Type 0 Configuration Request, permitting access to Extended Functions in an ARI Device immediately below the Port. For more details, see Section 6.13 of PCI Express Base Specification.

(table continues...)

(continued)

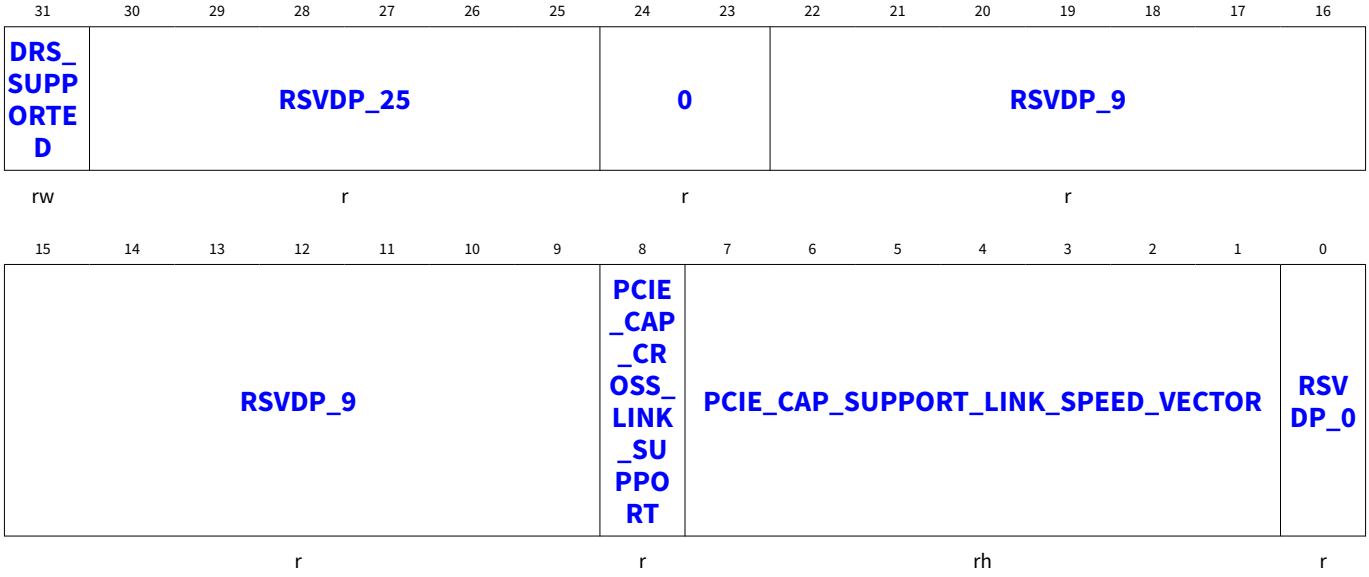
Field	Bits	Type	Description
PCIE_CAP_LTR_EN	10	rw	**LTR Mechanism Enable** When set to 1b, this bit enables Upstream Ports to send LTR messages and Downstream Ports to process LTR Messages. For Downstream Ports, this bit must be reset to the default value if the Port goes to DL_Down status. The write value is gated with the PCIE_CAP_LTR_SUPP field of DEVICE_CAPABILITIES2_REG. Note: RW for function #0 and RsdvP for all other functions. Note: The access attributes of this field are as follows: - Wire: if (pf=0 && DEVICE_CAPABILITIES2_REG.PCIE_CAP_LTR_SUPP) then R/W else R - Dbi: if (pf=0 && DEVICE_CAPABILITIES2_REG.PCIE_CAP_LTR_SUPP) then R/W else R
PCIE_CAP_OBFF_EN	14:13	rw	**OBFF Enable** This field enables the OBFF mechanism and selects the signaling method. For more details on these encodings, see section 6.19 of PCI Express Base Specification. This field is required for all Ports that support the OBFF Capability. Note: RW for function #0 and RsdvP for all other functions. Note: The access attributes of this field are as follows: • Wire: if (DEVICE_CAPABILITIES2_REG.PCIE_CAP_OBFF_SUPPORT) then R/W else R • Dbi: if (DEVICE_CAPABILITIES2_REG.PCIE_CAP_OBFF_SUPPORT) then R/W else R 00_B **DISABLED**: Disabled 01_B **EN_USING_MSG_SIG_A**: Enabled using Message signaling [Variation A] 10_B **EN_USING_MSG_SIG_B**: Enabled using Message signaling [Variation B] 11_B **EN_USING_WAKE_SIG**: Enabled using WAKE# signaling
0	9:6, 12:11, 31:15	r	**Reserved** Read as all 0's; should be written with all 0's.

19.4.1.122 Link Capabilities 2 Register

This register identifies PCI Express Link specific capabilities; in addition to the Link Capabilities Register.

LINK_CAPABILITIES2_REG Offset address: $000409C_H$

Link Capabilities 2 Register Reset values see: Table 871



Field	Bits	Type	Description
RSVDP_0	0	r	**Reserved for future use**
PCIE_CAP_SUPPORT_LINK_SPEED_VECTOR	7:1	rh	**Supported Link Speeds Vector** This field indicates the supported Link speed(s) of the associated Port. For each bit, a value of 1b indicates that the corresponding Link speed is supported; otherwise, the Link speed is not supported. For more details, see section 8.2.1 of PCI Express Base Specification. Bit definitions within this field are: • Bit 0 2.5 GT/s • Bit 1 5.0 GT/s • Bit 2 8.0 GT/s • Bit 3 16.0 GT/s • Bit 4 32.0 GT/s • Bits 6:5 RsvdP This field has a default of (PCIE_CAP_MAX_LINK_SPEED == 0101) ? 0011111 : (PCIE_CAP_MAX_LINK_SPEED == 0100) ? 0001111 : (PCIE_CAP_MAX_LINK_SPEED == 0011) ? 0000111 : (PCIE_CAP_MAX_LINK_SPEED == 0010) ? 0000011 : 0000001 where PCIE_CAP_MAX_LINK_SPEED is a field in the LINK_CAPABILITIES_REG register.

(table continues...)

(continued)

Field	Bits	Type	Description
PCIE_CAP_CROSS_LINK_SUPPORT	8	r	**Crosslink Supported** When set to 1b, this bit indicates that the associated Port supports crosslinks (for more details, see section 4.2.6.3.1 of PCI Express Base Specification). When set to 0b on a Port that supports Link speeds of 8.0 GT/s or higher, this bit indicates that the associated Port does not support crosslinks. When set to 0b on a Port that only supports Link speeds of 2.5 GT/s or 5.0 GT/s, this bit provides no information regarding the Port's level of crosslink support. It is recommended that this bit be Set in any Port that supports crosslinks even though doing so is only required for Ports that also support operating at 8.0 GT/s or higher Link speeds. Note: Software should use this bit when referencing fields whose definition depends on whether or not the Port supports crosslinks (for more details, see section 7.7.3.4 of PCI Express Base Specification). Multi-Function Devices associated with an Upstream Port must report the same value in this field for all Functions. No crosslink capability in this implementation
RSVDP_9	22:9	r	**Reserved for future use**
RSVDP_25	30:25	r	**Reserved for future use**
DRS_SUPPORTED	31	rw	**DRS Supported** When set, indicates support for the optional Device Readiness Status (DRS) capability. Must be Set in Downstream Ports that support DRS. Must be Set in Downstream Ports that support FRS. For Upstream Ports that support DRS, it is strongly recommended that this bit be Set in Function 0. For all other Functions associated with an Upstream Port, this bit must be Clear.127 Must be Clear in Functions that are not associated with a Port. RsvdP in all other Functions. Note: The access attributes of this field are as follows: - Wire: HWINIT - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R
0	24:23	r	**Reserved** Read as all 0's; should be written with all 0's.

Table 871 Reset values of LINK_CAPABILITIES2_REG

Reset	Reset value	Note
Kernel Reset	1000 0000 0000 0000 0000 0000 XXXX XXX0$_B$	

19.4.1.123 Link Control 2 and Status 2 Register

This register controls and provides information about PCI Express Link specific parameters; in addition to the Link Control and Link Status Register.

LINK_CONTROL2_LINK_STATUS2_REG Offset address: 00040A0$_H$

Link Control 2 and Status 2 Register Kernel Reset value: 0001 0003$_H$

19 Peripheral Component Interconnect Express (PCIe)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DRS_MESSAGE_RECEIVED	DOWNSTREAM_COMPO_PRESENCE			RSVDP_26		0				PCIE_CAP_LINK_EQ_REQ	PCIE_CAP_EQ_CPL_P3	PCIE_CAP_EQ_CPL_P2	PCIE_CAP_EQ_CPL_P1	PCIE_CAP_EQ_CPL	PCIE_CAP_CURR_DEEMPHASIS
r	r			r		r				rw1ch	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PCIE_CAP_COMPLIANCE_PRESET				PCIE_CAP_COMPLIANCE_SOS	PCIE_CAP_ENTER_MODIFIED_COMPLIANCE	PCIE_CAP_TX_MARGIN			PCIE_CAP_SEL_DEEMPHASIS	PCIE_CAP_HW_AUTO_SPEED_DISABLE	PCIE_CAP_ENTER_COMPLIANCE	PCIE_CAP_TARGET_LINK_SPEED			
rwh				rwh	rw	rwh			r	rw	rwh	rw			

Field	Bits	Type	Description
PCIE_CAP_TARGET_LINK_SPEED	3:0	rw	**Target Link Speed** For Downstream Ports, this field sets an upper limit on Link operational speed by restricting the values advertised by the Upstream component in its training sequences. The encoded value specifies a bit location in the Supported Link Speeds Vector (in the Link Capabilities 2 register) that corresponds to the desired target Link speed. All encodings other than the defined encodings are reserved. . If a value is written to this field that does not correspond to a supported speed (as indicated by the Supported Link Speeds Vector), the result is undefined. If either of the Enter Compliance or Enter Modified Compliance bits are implemented, then this field must also be implemented. The default value of this field is the highest Link speed supported by the component (as reported in the Max Link Speed field of the Link Capabilities register) unless the corresponding platform/form factor requires a different default value. For both Upstream and Downstream Ports, this field is used to set the target compliance mode speed when software is using the Enter Compliance bit to force a Link into compliance mode. For Upstream Ports, if the Enter Compliance bit is Clear, this field is permitted to have no effect. Note: The access attributes of this field are as follows: - Wire: R/W (sticky) - Dbi: R/W (sticky) Note: This register field is sticky 1_H **SUP_LINK_SPEED_VECTOR_BIT_0**: Supported Link Speeds Vector field bit 0 2_H **SUP_LINK_SPEED_VECTOR_BIT_1**: Supported Link Speeds Vector field bit 1 3_H **SUP_LINK_SPEED_VECTOR_BIT_2**: Supported Link Speeds Vector field bit 2 4_H **SUP_LINK_SPEED_VECTOR_BIT_3**: Supported Link Speeds Vector field bit 3 5_H **SUP_LINK_SPEED_VECTOR_BIT_4**: Supported Link Speeds Vector field bit 4 6_H **SUP_LINK_SPEED_VECTOR_BIT_5**: Supported Link Speeds Vector field bit 5 7_H **SUP_LINK_SPEED_VECTOR_BIT_6**: Supported Link Speeds Vector field bit 6 **others,** Reserved
PCIE_CAP_ENTER_COMPLIANCE	4	rwh	**Enter Compliance** Software is permitted to force a Link to enter Compliance mode (at the speed indicated in the Target Link Speed field and the de-emphasis/ preset level indicated by the Compliance Preset/De-emphasis field) by setting this bit to 1b in both components on a Link and then initiating a hot reset on the Link. Default value of this bit following Fundamental Reset is 0b. Components that support only the 2.5 GT/s speed are permitted to hardwire this bit to 0b. This bit is intended for debug, compliance testing purposes only. System firmware and software is allowed to modify this bit only during debug or compliance testing. In all other cases, the system must ensure that this bit is set to the default value. Note: The access attributes of this field are as follows: - Wire: R/W (sticky) - Dbi: R/W (sticky) Note: This register field is sticky.

(table continues...)

(continued)

Field	Bits	Type	Description
PCIE_CAP_HW_AUTO_SPEED_DISABLE	5	rw	**Hardware Autonomous Speed Disable** When set, this bit disables hardware from changing the Link speed for device-specific reasons other than attempting to correct unreliable Link operation by reducing Link speed. Initial transition to the highest supported common link speed is not blocked by this bit. Functions that do not implement the associated mechanism are permitted to hardwire this bit to 0b. Note: The access attributes of this field are as follows: - Wire: R/W (sticky) - Dbi: R/W (sticky) Note: This register field is sticky.
PCIE_CAP_SEL_DEEMPHASIS	6	r	**Selectable De-emphasis** When the Link is operating at 5.0 GT/s speed, this bit is used to control the transmit de-emphasis of the link in specific situations. For more details, see section 4.2.6 of PCI Express Base Specification. When the Link is not operating at 5.0 GT/s speed, the setting of this bit has no effect. Components that support only the 2.5 GT/s speed are permitted to hardwire this bit to 0b. This bit is not applicable and Reserved for Endpoints, PCI Express to PCI/PCI-X bridges, and Upstream Ports of Switches. Note: This register field is sticky. 0_B **_6_DB**: -6 dB 1_B **_3_5_DB**: -3.5 dB
PCIE_CAP_TX_MARGIN	9:7	rwh	**Transmit Margin, This field controls the value of the non-deemphasized voltage level at the Transmitter pins** This field is reset to 000b on entry to the LTSSM Polling.Configuration substate (see Chapter 4 of PCI Express Base Specification for details of how the Transmitter voltage level is determined in various states). - 001b-111b: As defined in Section 8.3.4 not all encodings are required to be implemented. For components that support only the 2.5 GT/s speed, the controller hardwires this bit to 000b. This field is intended for debug, compliance testing purposes only. System firmware and software is allowed to modify this field only during debug or compliance testing. In all other cases, the system must ensure that this field is set to the default value. Note: This register field is sticky. 000_B **NORMAL_RANGE**: Normal operating range **others,** Reserved
PCIE_CAP_ENTER_MODIFIED_COMPLIANCE	10	rw	**Enter Modified Compliance** When this bit is set to 1b, the device transmits Modified Compliance Pattern if the LTSSM enters Polling.Compliance substate. Components that support only the 2.5 GT/s speed are permitted to hardwire this bit to 0b. This bit is intended for debug, compliance testing purposes only. System firmware and software is allowed to modify this bit only during debug or compliance testing. In all other cases, the system must ensure that this bit is set to the default value. Note: The access attributes of this field are as follows: - Wire: R/W (sticky) - Dbi: R/W (sticky) Note: This register field is sticky.

(table continues...)

(continued)

Field	Bits	Type	Description
PCIE_CAP_COMPLIANCE_SOS	11	rwh	**Compliance SOS** When set to 1b, the LTSSM is required to send SKP Ordered Sets between sequences when sending the Compliance Pattern or Modified Compliance Pattern. This bit is applicable when the Link is operating at 2.5 GT/s or 5.0 GT/s data rates only. Note: The access attributes of this field are as follows: - Wire: R/W (sticky) - Dbi: R/W (sticky) Note: This register field is sticky.
PCIE_CAP_COMPLIANCE_PRESET	15:12	rwh	**Compliance Preset/De-emphasis** • For 8.0 GT/s and higher Data Rate: This field sets the Transmitter Preset in Polling.Compliance state if the entry occurred due to the Enter Compliance bit being 1b. The encodings are defined in section 4.2.3.2 of PCI Express Base Specification . Results are undefined if a reserved preset encoding is used when entering Polling.Compliance in this way. • For 5.0 GT/s Data Rate: This field sets the de-emphasis level in Polling.Compliance state if the entry occurred due to the Enter Compliance bit being 1b • When the Link is operating at 2.5 GT/s, the setting of this field has no effect. Components that support only 2.5 GT/s speed are permitted to hardwire this field to 0000b This field is intended for debug and compliance testing purposes. System firmware and software is allowed to modify this field only during debug or compliance testing. In all other cases, the system must ensure that this field is set to the default value. Note: The access attributes of this field are as follows: - Wire: R/W (sticky) - Dbi: R/W (sticky) Note: This register field is sticky. 0_H **_6_DB**: -6 dB (for 5.0 GT/s Data Rate) 1_H **_3_5_DB**: -3.5 dB (for 5.0 GT/s Data Rate) **others,** Reserved
PCIE_CAP_CURR_DEEMPHASIS	16	rh	**Current De-emphasis Level** When the Link is operating at 5.0 GT/s speed, this bit reflects the level of de-emphasis. The value in this bit is undefined when the Link is not operating at 5.0 GT/s speed. 0_B **_6DB**: -6 dB 1_B **_3_5DB**: -3.5 dB
PCIE_CAP_EQ_CPL	17	rh	**Equalization 8.0 GT/s Complete** When set to 1b, this bit indicates that the Transmitter Equalization procedure at the 8.0 GT/s data rate has completed. Details of the Transmitter Equalization process and when this bit needs to be set to 1b is provided in section 4.2.6.4.2 of PCI Express Base Specification. For Multi-Function Upstream Port, this bit must be implemented in Function 0 and RsvdZ in other Functions. For components that only support speeds below 8.0 GT/s, the controller hardwires this bit to 0b. Note: This register field is sticky.

(table continues...)

(continued)

Field	Bits	Type	Description
PCIE_CAP_EQ_CPL_P1	18	rh	**Equalization 8.0 GT/s Phase 1 Successful** When set to 1b, this bit indicates that Phase 1 of the 8.0 GT/s Transmitter Equalization procedure has successfully completed. Details of the Transmitter Equalization process and when this bit needs to be set to 1b is provided in section 4.2.6.4.2 of PCI Express Base Specification. Note: This register field is sticky.
PCIE_CAP_EQ_CPL_P2	19	rh	**Equalization 8.0 GT/s Phase 2 Successful** When set to 1b, this bit indicates that Phase 2 of the 8.0 GT/s Transmitter Equalization procedure has successfully completed. Details of the Transmitter Equalization process and when this bit needs to be set to 1b is provided in section 4.2.6.4.2 of PCI Express Base Specification. Note: This register field is sticky.
PCIE_CAP_EQ_CPL_P3	20	rh	**EEqualization 8.0 GT/s Phase 3 Successful** When set to 1b, this bit indicates that Phase 3 of the 8.0 GT/s Transmitter Equalization procedure has successfully completed. Details of the Transmitter Equalization process and when this bit needs to be set to 1b is provided in section 4.2.6.4.2 of PCI Express Base Specification. Note: This register field is sticky.
PCIE_CAP_LINK_EQ_REQ	21	rw1ch	**Link Equalization Request 8.0 GT/s** This bit is set by hardware to request the 8.0 GT/s Link equalization process to be performed on the Link. For more details, see sections 4.2.3 and 4.2.6.4.2 of PCI Express Base Specification. Note: This register field is sticky. Software write with 1 clears this bit-field.
RSVDP_26	27:26	r	**Reserved for future use**

(table continues...)

(continued)

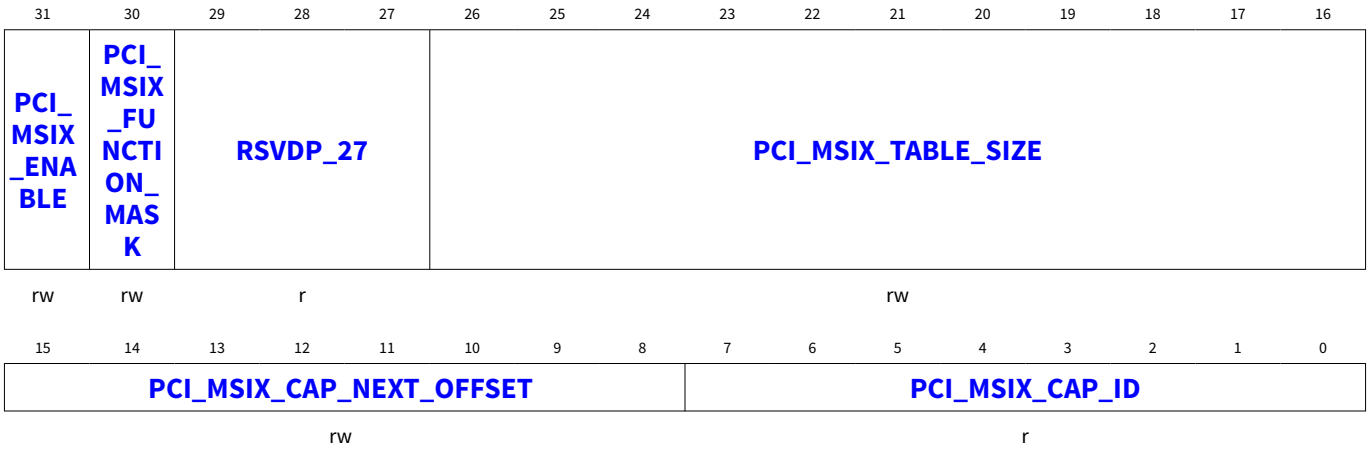
Field	Bits	Type	Description
DOWNSTREAM_COMPO_PRESENCE	30:28	r	**Downstream Component Presence** This field indicates the presence and DRS status for the Downstream Component, if any, connected to the Link. All encodings other than the defined encodings are reserved. Component Presence state must be determined by the logical "OR" of the Physical Layer in-band presence detect mechanism and, if present, any out-of-band presence detect mechanism implemented for the Link. If no out-of-band presence detect mechanism is implemented, then Component Presence state must be determined solely by the Physical Layer in-band presence detect mechanism. This field must be implemented in any Downstream Port where the DRS Supported bit is Set in the Link Capabilities 2 register. This field is RsvdZ for all other Functions. 000_B **LINK_DN_NOT_DETERMINED**: Link Down - Presence Not Determined 001_B **LINK_DN_DSP_NOT_PRESENT**: Link Down - Component Not Present indicates the Downstream Port (DP) has determined that a Downstream Component is not present 010_B **LINK_DN_DLL_NOT_ACTIVE**: Link Down - Component Present indicates the DP has determined that a Downstream Component is present, but the Data Link Layer is not active 100_B **LINK_UP_DRS_NOT_RECEIVED**: Link Up - Component Presentindicates the DP has determined that a Downstream Component is present, but no DRS Message has been received since the Data Link Layer became active 101_B **LINK_UP_DRS_RECEIVED**: Link Up - Component Present and DRS Received indicates the DP has received a DRS Message since the Data Link Layer became active **others,** Reserved
DRS_MESSAGE_RECEIVED	31	r	**DRS Message Received** This bit must be set whenever the Port receives a DRS Message. This bit must be cleared in DL_Down. This bit must be implemented in any Downstream Port where the DRS Supported bit is Set in the Link Capabilities 2 register. This bit is RsvdZ for all other Functions.
0	25:22	r	**Reserved** Read as all 0's; should be written with all 0's.

19.4.1.124 MSI-X Capability ID, Next Pointer, Control Register

This Register holds MSI-X Capability ID, Next Capability pointer. It also controls the MSI-X behaviour.

PCI_MSIX_CAP_ID_NEXT_CTRL_REG Offset address: $00040B0_H$

MSI-X Capability ID, Next Pointer, Control Register Kernel Reset value: $0040\ 0011_H$



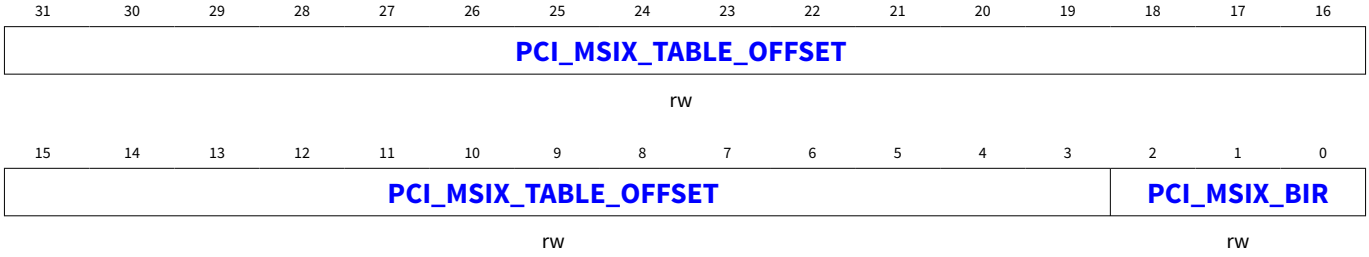
Field	Bits	Type	Description
PCI_MSIX_CAP_ID	7:0	r	**MSI-X Capability ID** This field indicates the MSI-X Capability structure. This field must return a Capability ID of 11h indicating that this is an MSI-X Capability structure.
PCI_MSIX_CAP_NEXT_OFFSET	15:8	rw	**MSI-X Next Capability Pointer** This field contains the offset to the next PCI Capability structure or 00h if no other items exist in the linked list of Capabilities. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R Note: This register field is sticky.
PCI_MSIX_TABLE_SIZE	26:16	rw	**MSI-X Table Size** System software reads this field to determine the MSI-X Table Size N, which is encoded as N-1. For example, a returned value of 000 0000 0011b indicates a table size of 4. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R Note: This register field is sticky.
RSVDP_27	29:27	r	**Reserved for future use**
PCI_MSIX_FUNCTION_MASK	30	rw	**Function Mask** If Set, all of the vectors associated with the Function are masked, regardless of their per-vector Mask bit values. If Clear, each vector's Mask bit determines whether the vector is masked or not. Setting or Clearing the MSI-X Function Mask bit has no effect on the value of the per-vector Mask bits. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
PCI_MSIX_ENABLE	31	rw	**MSI-X Enable** If Set and the MSI Enable bit in the MSI Message Control Register for MSI is Clear, the Function is permitted to use MSI-X to request service and is prohibited from using INTx interrupts (if implemented). System configuration software Sets this bit to enable MSI-X. A device driver is prohibited from writing this bit to mask a Function's service request. If Clear, the Function is prohibited from using MSI-X to request service.

19.4.1.125 MSI-X Table Offset and BIR Register

This register provides Table BIR and MSI-x Table offset select.

MSIX_TABLE_OFFSET_REG Offset address: $00040B4_H$

MSI-X Table Offset and BIR Register Kernel Reset value: $0000\ 0A00_H$



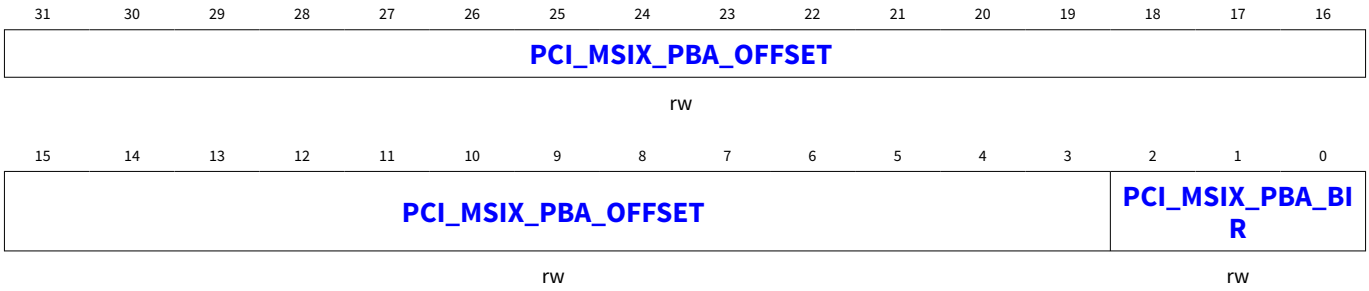
Field	Bits	Type	Description
PCI_MSIX_BIR	2:0	rw	**MSI-X Table BAR Indicator Register Field** This field indicates which one of a Function's Base Address Registers , located beginning at 10h in Configuration Space, or entry in the Enhanced Allocation capability with a matching BEI , is used to map the Function's MSI-X Table into Memory Space. All encodings other than the defined encodings are reserved. For a 64-bit Base Address Register , the Table BIR indicates the lower DWORD. For Functions with Type 1 Configuration Space headers, BIR values 2 through 5 are also Reserved. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R Note: This register field is sticky. 000_B **BAR_10**: Base Address Register 10h 001_B **BAR_14**: Base Address Register 14h 010_B **BAR_18**: Base Address Register 18h 011_B **BAR_1C**: Base Address Register 1Ch 100_B **BAR_20**: Base Address Register 20h 101_B **BAR_24**: Base Address Register 24h **others,** Reserved
PCI_MSIX_TABLE_OFFSET	31:3	rw	**MSI-X Table Offset** Used as an offset from the address contained by one of the Function's Base Address Registers to point to the base of the MSI-X Table. The lower 3 Table BIR bits are masked off (set to zero) by software to form a 32-bit QWORD-aligned offset. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R Note: This register field is sticky.

19.4.1.126 MSI-X PBA Offset and BIR Register

This register provides PBA Offset and PBA BIR value.

MSIX_PBA_OFFSET_REG Offset address: $00040B8_H$

MSI-X PBA Offset and BIR Register Kernel Reset value: $0000\ 0E10_H$



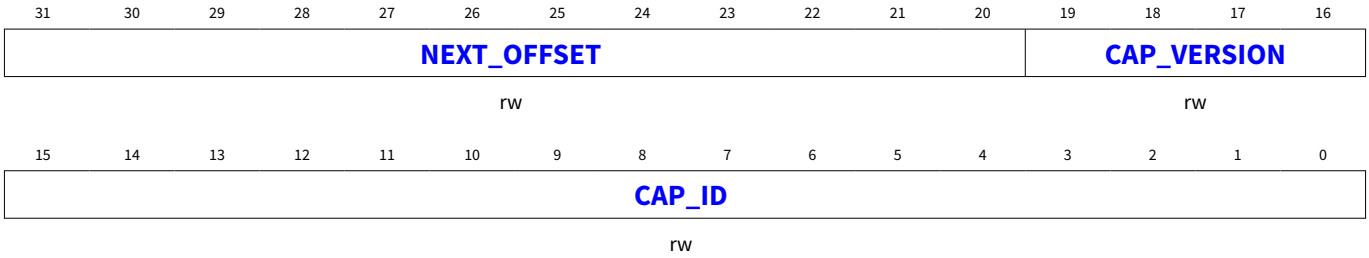
Field	Bits	Type	Description
PCI_MSIX_PBA_BIR	2:0	rw	**MSI-X PBA BIR** This field indicates which one of a Function's Base Address Registers , located beginning at 10h in Configuration Space, or entry in the Enhanced Allocation capability with a matching BEI, is used to map the Function's MSI-X PBA into Memory Space. The PBA BIR value definitions are identical to those for the Table BIR . Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R Note: This register field is sticky.
PCI_MSIX_PBA_OFFSET	31:3	rw	**MSI-X PBA Offset** Used as an offset from the address contained by one of the Function's Base Address Registers to point to the base of the MSI-X PBA. The lower 3 PBA BIR bits are masked off (set to zero) by software to form a 32-bit QWORD-aligned offset. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R Note: This register field is sticky.

19.4.1.127 Advanced Error Reporting Extended Capability Header

Advanced Error Reporting Extended Capability Header provides information about Capability ID, Version, and next offset.

AER_EXT_CAP_HDR_OFF Offset address: 0004100_H

Advanced Error Reporting Extended Capability Header Kernel Reset value: $1482\ 0001_H$



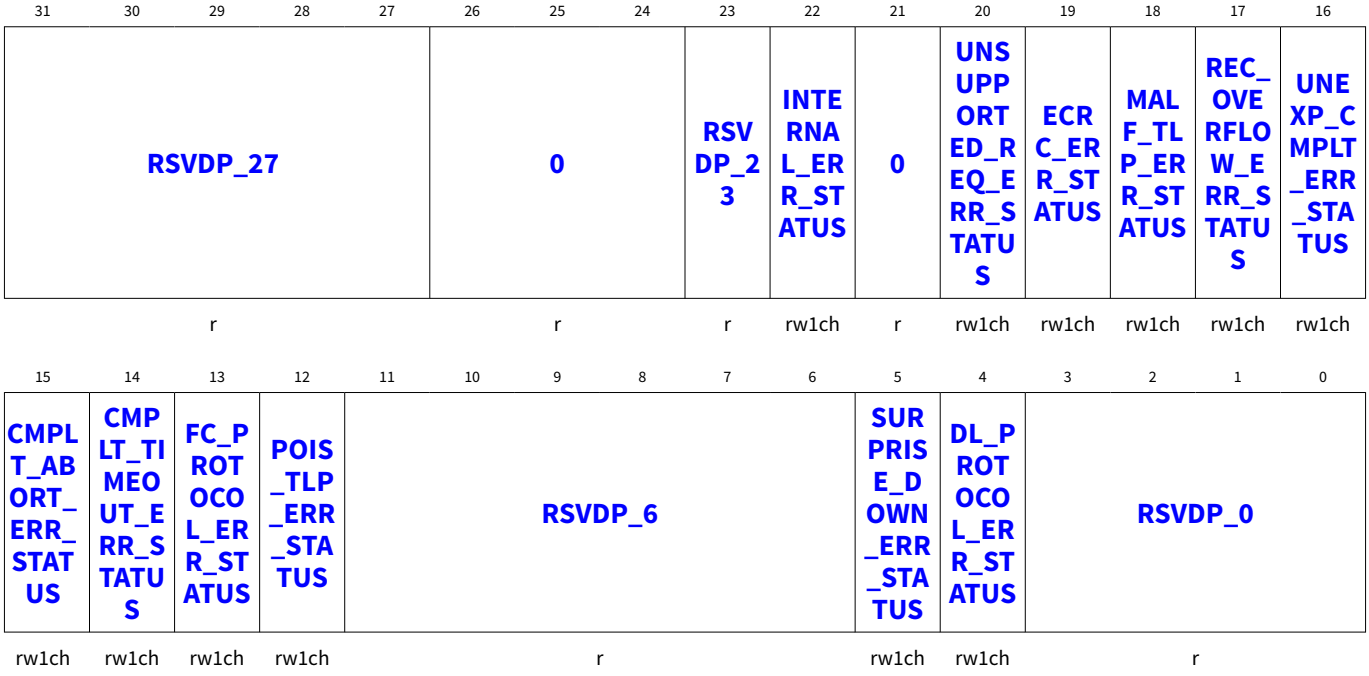
Field	Bits	Type	Description
CAP_ID	15:0	rw	**AER Extended Capability ID** This field is a PCI-SIG defined ID number that indicates the nature and format of the Extended Capability. The Extended Capability ID for the Advanced Error Reporting Capability is 0001h. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky.
CAP_VERSION	19:16	rw	**Capability Version** This field is a PCI-SIG defined version number that indicates the version of the Capability structure present. This field must be 2h if the End-End TLP Prefix Supported bit is set and must be 1h or 2h otherwise. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky.
NEXT_OFFSET	31:20	rw	**Next Capability Offset** This field contains the offset to the next PCI Express Capability structure or 000h if no other items exist in the linked list of Capabilities. For Extended Capabilities implemented in Configuration Space, this offset is relative to the beginning of PCI-compatible Configuration Space and thus must always be either 000h (for terminating list of Capabilities) or greater than 0FFh. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky.

19.4.1.128 Uncorrectable Error Status Register

The Uncorrectable Error Status Register (Offset 04h) indicates error detection status of individual errors on a PCI Express device Function. An individual error status bit that is Set indicates that a particular error was detected; software may clear an error status by writing a 1b to the respective bit. Register bits not implemented by the Function are hardwired to 0b.

UNCORR_ERR_STATUS_OFF Offset address: 0004104_H

Uncorrectable Error Status Register Reset values see: Table 872



Field	Bits	Type	Description
RSVDP_0	3:0	r	**Reserved for future use**
DL_PROTOCOL_ERR_STATUS	4	rw1ch	**Data Link Protocol Error Status** Status bit for Data Link Protocol Error. Note: This register field is sticky. Software write with 1 clears this bit-field.
SURPRISE_DOWN_ERR_STATUS	5	rw1ch	**Surprise Down Error Status (Optional)** Status bit for Surprise Down Error. Note: This register field is sticky. Software write with 1 clears this bit-field.
RSVDP_6	11:6	r	**Reserved for future use**
POIS_TLP_ERR_STATUS	12	rw1ch	**Poisoned TLP Status** Status bit for Poisoned TLP. Note: This register field is sticky. Software write with 1 clears this bit-field.
FC_PROTOCOL_ERR_STATUS	13	rw1ch	**Flow Control Protocol Error Status** Status bit for Flow Control Protocol Error. Note: This register field is sticky. Software write with 1 clears this bit-field.
CMPLT_TIMEOUT_ERR_STATUS	14	rw1ch	**Completion Timeout Status** Status for Completion Timeout. Note: This register field is sticky. Software write with 1 clears this bit-field.
CMPLT_ABORT_ERR_STATUS	15	rw1ch	**Completer Abort Status** Status bit for Completer Abort. Note: This register field is sticky. Software write with 1 clears this bit-field.
UNEXP_CMPLT_ERR_STATUS	16	rw1ch	**Unexpected Completion Status** Status bit for Unexpected Completion. Note: This register field is sticky. Software write with 1 clears this bit-field.

(table continues...)

(continued)

Field	Bits	Type	Description
REC_OVERFLOW_ERR_STATUS	17	rw1ch	**Receiver Overflow Status** Status bit for Receiver Overflow. Note: This register field is sticky. Software write with 1 clears this bit-field.
MALF_TLP_ERR_STATUS	18	rw1ch	**Malformed TLP Status** This field represents status of Malformed TLP. Note: This register field is sticky. Software write with 1 clears this bit-field.
ECRC_ERR_STATUS	19	rw1ch	**ECRC Error Status** This field represents status of ECRC Error. Note: This register field is sticky. Software write with 1 clears this bit-field.
UNSUPPORTED_REQ_ERR_STATUS	20	rw1ch	**Unsupported Request Error Status** This field represents status of Unsupported Request Error. Note: This register field is sticky. Software write with 1 clears this bit-field.
INTERNAL_ERR_STATUS	22	rw1ch	**Uncorrectable Internal Error Status** This field gives status of the Uncorrectable Internal Error.Note: This register field is sticky. Software write with 1 clears this bit-field.
RSVDP_23	23	r	**Reserved for future use**
RSVDP_27	31:27	r	**Reserved for future use**
0	21, 26:24	r	**Reserved** Read as 0; should be written with 0.

Table 872 Reset values of UNCORR_ERR_STATUS_OFF

Reset	Reset value	Note
Kernel Reset	0000 0000 0000 0XX0 00X0 0000 00XX 0000_B	

19.4.1.129 Uncorrectable Error Mask Register

The Uncorrectable Error Mask Register controls reporting of individual errors by the device Function to the PCI Express Root Complex through a PCI Express error Message. A masked error (respective bit Set in the mask register) is not recorded or reported in the Header Log, TLP Prefix Log, or First Error Pointer, and is not reported to the PCI Express Root Complex by this Function. There is a mask bit per error bit of the Uncorrectable Error Status register. Register fields for bits not implemented by the Function are hardwired to 0b.

UNCORR_ERR_MASK_OFF Offset address: 0004108_H

Uncorrectable Error Mask Register Kernel Reset value: $0040\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_26						TLP_PRFX_BLOCKED_ERR_MASK	ATOMIC_EGRESS_BLOCKED_ERR_MASK	RSVDP_23	INTERNAL_ERR_MASK	0	UNSUPPORTED_REQ_ERR_MASK	ECRC_ERR_MASK	MALF_TLP_ERR_MASK	REC_OVERFLOW_ERR_MASK	UNEXP_CMPLT_ERR_MASK
r						r	r	r	rw	r	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CMPLT_ABORT_ERR_MASK	CMPLT_TIMEOUT_ERR_MASK	FC_PROTOCOL_ERR_MASK	POIS_TLP_ERR_MASK	RSVDP_6						SURPRISE_DOWN_ERR_MASK	DL_PROTOCOL_ERR_MASK	RSVDP_0			
rw	rw	rw	rw	r						r	rw	r			

Field	Bits	Type	Description
RSVDP_0	3:0	r	**Reserved for future use**
DL_PROTOCOL_ERR_MASK	4	rw	**Data Link Protocol Error Mask** This field informs whether Data Link Protocol Error is masked or not. Note: This register field is sticky.
SURPRISE_DOWN_ERR_MASK	5	r	**Surprise Down Error Mask** Mask bit for Surprise Down Error. Note: This register field is sticky.
RSVDP_6	11:6	r	**Reserved for future use**
POIS_TLP_ERR_MASK	12	rw	**Poisoned TLP Error Mask** Mask bit for Poisoned TLP Error. Note: This register field is sticky.
FC_PROTOCOL_ERR_MASK	13	rw	**Flow Control Protocol Error Mask** Mask bit for Flow Control Protocol Error. Note: This register field is sticky.
CMPLT_TIMEOUT_ERR_MASK	14	rw	**Completion Timeout Error Mask** Mask bit for Completion Timeout Error. Note: This register field is sticky.
CMPLT_ABORT_ERR_MASK	15	rw	**Completer Abort Error Mask (Optional)** Mask bit for Completer Abort Error. Note: This register field is sticky.
UNEXP_CMPLT_ERR_MASK	16	rw	**Unexpected Completion Mask** Mask bit for Unexpected Completion Error. Note: This register field is sticky.
REC_OVERFLOW_ERR_MASK	17	rw	**Receiver Overflow Mask (Optional)** This field represents Receiver Overflow Mask. Note: This register field is sticky.

(table continues...)

(continued)

Field	Bits	Type	Description
MALF_TLP_ERR_MASK	18	rw	**Malformed TLP Mask** Mask bit for Malformed TLP. Note: This register field is sticky.
ECRC_ERR_MASK	19	rw	**ECRC Error Mask (Optional)** Mask bit for ECRC Error. Note: The access attributes of this field are as follows: - Wire: R/W (sticky) - Dbi: R/W (sticky) Note: This register field is sticky.
UNSUPPORTED_REQ_ERR_MASK	20	rw	**Unsupported Request Error Mask** Mask bit for Unsupported Request Error. Note: This register field is sticky.
INTERNAL_ERR_MASK	22	rw	**Uncorrectable Internal Error Mask (Optional)** Mask bit for Uncorrectable Internal Error. Note: This register field is sticky.
RSVDP_23	23	r	**Reserved for future use**
ATOMIC_EGRESS_BLOCKED_ERR_MASK	24	r	**AtomicOp Egress Block Mask (Optional)** Mask bit for AtomicOp Egress Block Error. Note: This register field is sticky.
TLP_PRFX_BLOCKED_ERR_MASK	25	r	**TLP Prefix Blocked Error Mask** Mask bit for TLP Prefix Blocked Error. Note: Not supported. Note: This register field is sticky.
RSVDP_26	31:26	r	**Reserved for future use**
0	21	r	**Reserved** Read as 0; should be written with 0.

19.4.1.130 Uncorrectable Error Severity Register

The Uncorrectable Error Severity Register controls whether an individual error is reported as a Non-fatal or Fatal error. An error is reported as fatal when the corresponding error bit in the severity register is Set. If the bit is Clear, the corresponding error is considered non-fatal.

UNCORR_ERR_SEV_OFF Offset address: $000410C_H$

Uncorrectable Error Severity Register Kernel Reset value: $0046\ 2030_H$

19 Peripheral Component Interconnect Express (PCIe)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_27					0	TLP_PRFX_BLOCKED_ERR_SEVERITY	0	RSVDP_23	INTERNAL_ERR_SEVERITY	0	UNSUPPORTED_REQ_ERR_SEVERITY	ECRC_ERR_SEVERITY	MALF_TLP_ERR_SEVERITY	REC_OVERFLOW_ERR_SEVERITY	UNEXP_CMPLT_ERR_SEVERITY
r					r	r	r	r	rw	r	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CMPLT_ABORT_ERR_SEVERITY	CMPLT_TIMEOUT_ERR_SEVERITY	FC_PROTOCOL_ERR_SEVERITY	POIS_TLP_ERR_SEVERITY	RSVDP_6						SURPRISE_DOWN_ERR_SVRITY	DL_PROTOCOL_ERR_SEVERITY	RSVDP_0			
rw	rw	rw	rw	r						r	rw	r			

Field	Bits	Type	Description
RSVDP_0	3:0	r	**Reserved for future use**
DL_PROTOCOL_ERR_SEVERITY	4	rw	**Data Link Protocol Error Severity** Severity bit for Data Link Protocol Error. Note: This register field is sticky.
SURPRISE_DOWN_ERR_SVRITY	5	r	**Surprise Down Error Severity (Optional)** Severity bit for Surprise Down Error. Note: This register field is sticky.
RSVDP_6	11:6	r	**Reserved for future use**
POIS_TLP_ERR_SEVERITY	12	rw	**Poisoned TLP Severity** Severity bit for Poisoned TLP. Note: This register field is sticky.
FC_PROTOCOL_ERR_SEVERITY	13	rw	**Flow Control Protocol Error Severity (Optional)** Severity bit for Flow Control Protocol Error. Note: This register field is sticky.
CMPLT_TIMEOUT_ERR_SEVERITY	14	rw	**Completion Timeout Error Severity** Severity bit for Completion Timeout Error. Note: This register field is sticky.
CMPLT_ABORT_ERR_SEVERITY	15	rw	**Completer Abort Error Severity (Optional)** Severity bit for Completer Abort Error. Note: This register field is sticky.
UNEXP_CMPLT_ERR_SEVERITY	16	rw	**Unexpected Completion Error Severity** Severity bit for Unexpected Completion Error. Note: This register field is sticky.
REC_OVERFLOW_ERR_SEVERITY	17	rw	**Receiver Overflow Error Severity (Optional)** Severity bit for Receiver Overflow Error. Note: This register field is sticky.

(table continues...)

(continued)

Field	Bits	Type	Description
MALF_TLP_ERR_SEVERITY	18	rw	**Malformed TLP Severity** Severity bit for Malformed TLP. Note: This register field is sticky.
ECRC_ERR_SEVERITY	19	rw	**ECRC Error Severity (Optional)** Severity bit for ECRC Error. Note: The access attributes of this field are as follows: - Wire: R/W (sticky) - Dbi: R/W (sticky) Note: This register field is sticky.
UNSUPPORTED_REQ_ERR_SEVERITY	20	rw	**Unsupported Request Error Severity** Severity bit for Unsupported Request Error. Note: This register field is sticky.
INTERNAL_ERR_SEVERITY	22	rw	**Uncorrectable Internal Error Severity (Optional)** Severity bit for Uncorrectable Internal Error. Note: This register field is sticky.
RSVDP_23	23	r	**Reserved for future use**
TLP_PRFX_BLOCKED_ERR_SEVERITY	25	r	**TLP Prefix Blocked Error Severity (Optional)** Severity bit for TLP Prefix Blocked Error. Note: Not supported. Note: This register field is sticky.
RSVDP_27	31:27	r	**Reserved for future use**
0	21, 24, 26	r	**Reserved** Read as 0; should be written with 0.

19.4.1.131 Correctable Error Status Register

The Correctable Error Status register reports error status of individual correctable error sources on a PCI Express device Function. When an individual error status bit is Set, it indicates that a particular error occurred; software may clear an error status by writing a 1b to the respective bit. Register bits not implemented by the Function are hardwired to 0b by the controller.

CORR_ERR_STATUS_OFF Offset address: 0004110_H

Correctable Error Status Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_16															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HEADER_LOG_OVERFLOW_STATUS	CORRECTED_INT_ERR_STATUS	ADVISORY_NON_FATAL_ERR_STATUS	RPL_TIMER_TIMEOUT_STATUS	RSVDP_9			REPLAY_NO_ROLEOVER_STATUS	BAD_DLLP_STATUS	BAD_TLP_STATUS	RSVDP_1					RX_ERR_STATUS
rw1ch	rw1ch	rw1ch	rw1ch	r			rw1ch	rw1ch	rw1ch	r					rw1ch

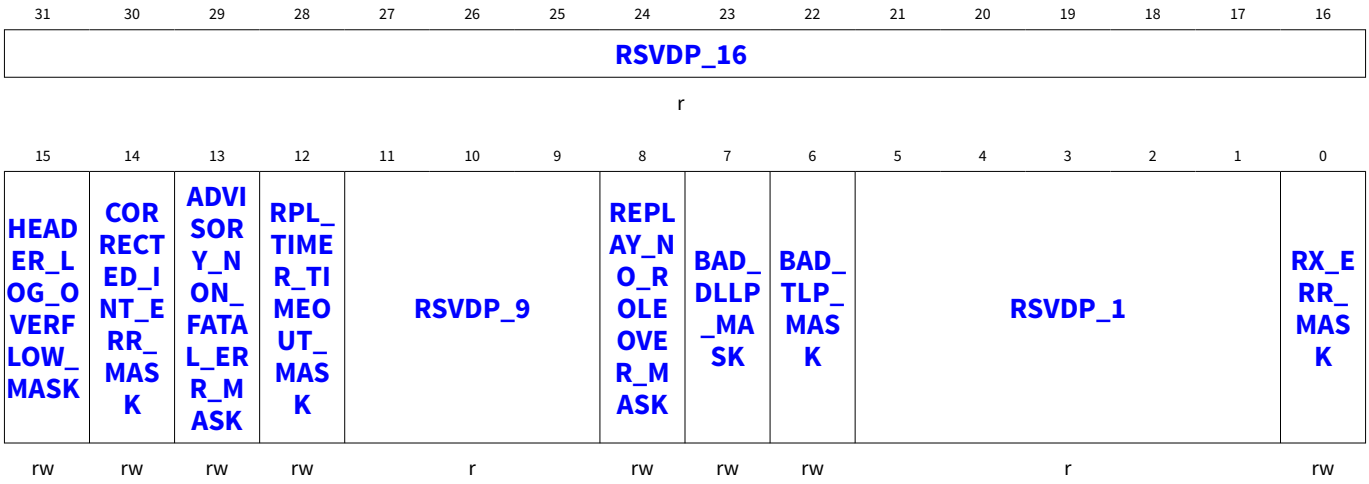
Field	Bits	Type	Description
RX_ERR_STATUS	0	rw1ch	**Receiver Error Status (Optional)** This field provides status of Receiver Error. Note: This register field is sticky. Software write with 1 clears this bit-field.
RSVDP_1	5:1	r	**Reserved for future use**
BAD_TLP_STATUS	6	rw1ch	**Bad TLP Status** Status bit for Bad TLP. Note: This register field is sticky. Software write with 1 clears this bit-field.
BAD_DLLP_STATUS	7	rw1ch	**Bad DLLP Status** Status bit for Bad DLLP. Note: This register field is sticky. Software write with 1 clears this bit-field.
REPLAY_NO_ROLEOVER_STATUS	8	rw1ch	**REPLAY_NUM Rollover Status** Status bit for REPLAY_NUM Rollover. Note: This register field is sticky. Software write with 1 clears this bit-field.
RSVDP_9	11:9	r	**Reserved for future use**
RPL_TIMER_TIMEOUT_STATUS	12	rw1ch	**Replay Timer Timeout Status** Status bit for Replay Timer Timeout. Note: This register field is sticky. Software write with 1 clears this bit-field.
ADVISORY_NON_FATAL_ERR_STATUS	13	rw1ch	**Advisory Non-Fatal Error Status** Status bit for Advisory Non-Fatal Error. Note: This register field is sticky. Software write with 1 clears this bit-field.
CORRECTED_INT_ERR_STATUS	14	rw1ch	**Corrected Internal Error Status (Optional)** This field provides status of Corrected Internal Error. Note: This register field is sticky. Software write with 1 clears this bit-field.
HEADER_LOG_OVERFLOW_STATUS	15	rw1ch	**Header Log Overflow Error Status (Optional)** This field provides status of Header Log Overflow Error. Note: This register field is sticky. Software write with 1 clears this bit-field.
RSVDP_16	31:16	r	**Reserved for future use**

19.4.1.132 Correctable Error Mask Register

The Correctable Error Mask Register controls reporting of individual correctable errors by this Function to the PCI Express Root Complex through a PCI Express error Message. A masked error (respective bit Set in the mask register) is not reported to the PCI Express Root Complex by this Function. There is a mask bit per error bit in the Correctable Error Status register. Register fields for bits not implemented by the Function are hardwired to 0b by the controller.

CORR_ERR_MASK_OFF Offset address: 0004114_H

Correctable Error Mask Register Kernel Reset value: $0000\ E000_H$



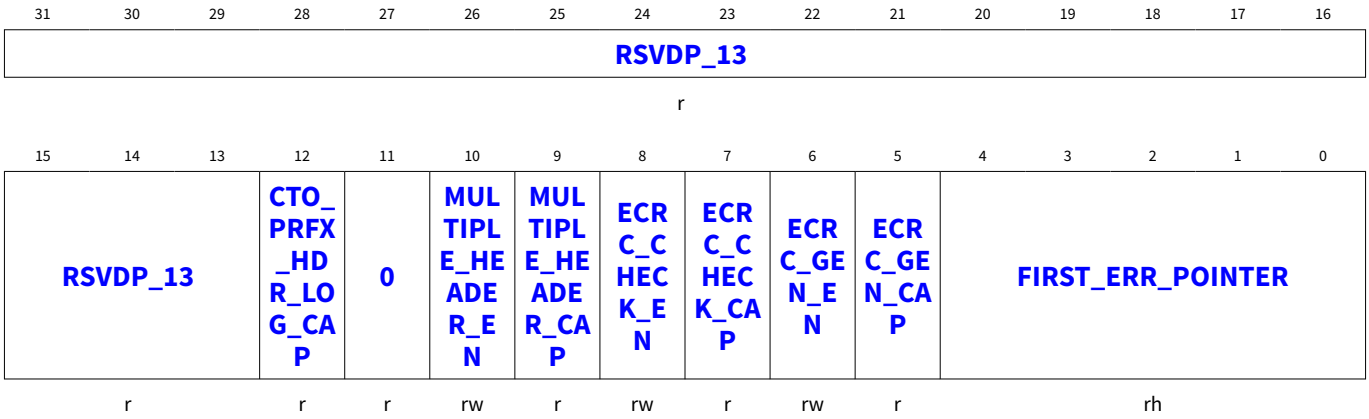
Field	Bits	Type	Description
RX_ERR_MASK	0	rw	**Receiver Error Mask (Optional)** Masking bit for Receiver Error. Note: This register field is sticky.
RSVDP_1	5:1	r	**Reserved for future use**
BAD_TLP_MASK	6	rw	**Bad TLP Mask** Masking bit for Bad TLP. Note: This register field is sticky.
BAD_DLLP_MASK	7	rw	**Bad DLLP Mask** Masking bit for Bad DLLP. Note: This register field is sticky.
REPLAY_NO_ROLEOVER_MASK	8	rw	**REPLAY_NUM Rollover Mask** Masking bit for REPLAY_NUM Rollover. Note: This register field is sticky.
RSVDP_9	11:9	r	**Reserved for future use**
RPL_TIMER_TIMEOUT_MASK	12	rw	**Replay Timer Timeout Mask** Masking bit for Replay Timer Timeout. Note: This register field is sticky.
ADVISORY_NON_FATAL_ERR_MASK	13	rw	**Advisory Non-Fatal Error Mask** Masking bit for Advisory Non-Fatal Error. Note: This register field is sticky.
CORRECTED_INT_ERR_MASK	14	rw	**Corrected Internal Error Mask (Optional)** Masking bit for Corrected Internal Error Mask. Note: This register field is sticky.
HEADER_LOG_OVERFLOW_MASK	15	rw	**Header Log Overflow Error Mask (Optional)** Masking bit for Header Log Overflow Error. Note: This register field is sticky.
RSVDP_16	31:16	r	**Reserved for future use**

19.4.1.133 Advanced Error Capabilities and Control Register

Advanced Error Capabilities and Control Register provides information whether the individual capability is supported or not. If the capability is supported then it is enabled or not.

ADV_ERR_CAP_CTRL_OFF Offset address: 0004118_H

Advanced Error Capabilities and Control Register Kernel Reset value: $0000\ 02A0_H$



Field	Bits	Type	Description
FIRST_ERR_POINTER	4:0	rh	**First Error Pointer** The First Error Pointer is a field that identifies the bit position of the first error reported in the Uncorrectable Error Status register. Note: This register field is sticky.
ECRC_GEN_CAP	5	r	**ECRC Generation Capable** If Set, this bit indicates that the Function is capable of generating ECRC. Note: This register field is sticky.
ECRC_GEN_EN	6	rw	**ECRC Generation Enable** When Set, ECRC generation is enabled. Functions that do not implement the associated mechanism are permitted to hardwire this bit to 0b. Default value of this bit is 0b. Note: This register field is sticky.
ECRC_CHECK_CAP	7	r	**ECRC Check Capable** If Set, this bit indicates that the Function is capable of checking ECRC. Note: This register field is sticky.
ECRC_CHECK_EN	8	rw	**ECRC Check Enable** When Set, ECRC checking is enabled. Functions that do not implement the associated mechanism are permitted to hardwire this bit to 0b. Default value of this bit is 0b. Note: This register field is sticky.
MULTIPLE_HEADER_CAP	9	r	**Multiple Header Recording Capable** If Set, this bit indicates that the Function is capable of recording more than one error header. Note: This register field is sticky.
MULTIPLE_HEADER_EN	10	rw	**Multiple Header Recording Enable** When Set, this bit enables the Function to record more than one error header. Functions that do not implement the associated mechanism are permitted to hardwire this bit to 0b. Default value of this bit is 0b. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: R (sticky) Note: This register field is sticky.
CTO_PRFX_HDR_LOG_CAP	12	r	**Completion Timeout Prefix/Header Log Capable** If Set, this bit indicates that the Function records the prefix/header of Request TLPs that experience a Completion Timeout error.
RSVDP_13	31:13	r	**Reserved for future use**
0	11	r	**Reserved** Read as 0; should be written with 0.

19.4.1.134 Header i Log Register 0

The Header Log Register 0 contains the header for the TLP corresponding to a detected error; The header is captured such that, when read using DW accesses, the fields of the header are laid out in the same way the headers are presented in the specification. Therefore, byte 0 of the header is located in byte 3 of the Header Log Register 0, byte 1 of the header is in byte 2 of the Header Log Register 0, and so forth. For 12-byte headers, only bytes 0 through 11 of the Header Log Register are used and values in bytes 12 through 15 are undefined. In certain cases where a Malformed TLP is reported, the Header Log Register may contain TLP Prefix information.

HDR_LOG_i_OFF (i=0-3) Offset address: $000411C_H+i*4$

Header i Log Register 0 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIRST_DWORD_FOURTH_BYTE								FIRST_DWORD_THIRD_BYTE							
rh								rh							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIRST_DWORD_SECOND_BYTE								FIRST_DWORD_FIRST_BYTE							
rh								rh							

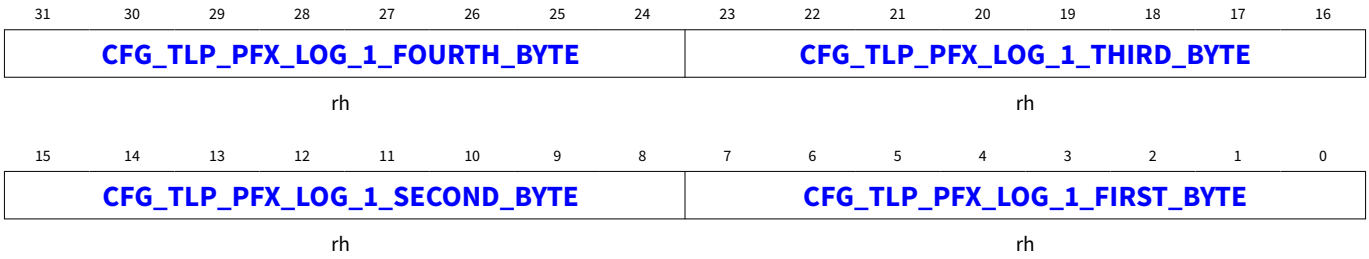
Field	Bits	Type	Description
FIRST_DWORD_FIRST_BYTE	7:0	rh	**Byte 0 of Header log register of First 32-bit Data Word** This field represents first byte of First DW of Header. Note: This register field is sticky.
FIRST_DWORD_SECOND_BYTE	15:8	rh	**Byte 1 of Header log register of First 32-bit Data Word** This field represents second byte of First DW of Header. Note: This register field is sticky.
FIRST_DWORD_THIRD_BYTE	23:16	rh	**Byte 2 of Header log register of First 32-bit Data Word** This field represents third byte of First DW of Header. Note: This register field is sticky.
FIRST_DWORD_FOURTH_BYTE	31:24	rh	**Byte 3 of Header log register of First 32-bit Data Word** This field represents fourth byte of First DW of Header. Note: This register field is sticky.

19.4.1.135 TLP Prefix i Log Register 1

The First TLP Prefix Log Register contains the first End-End TLP Prefix from the TLP corresponding to the detected error. The TLP Prefix Log Register is only meaningful when the TLP Prefix Log Present bit is Set. The TLP Prefixes are captured such that, when read using DW accesses, the fields of the TLP Prefix are laid out in the same way the fields of the TLP Prefix are described. Therefore, byte 0 of a TLP Prefix is located in byte 3 of the associated TLP Prefix Log Register; byte 1 of a TLP Prefix is located in byte 2; and so forth. The TLP Prefix Log Registers beyond the number supported by the Function are hardwired to zero by controller. If the End-End TLP Prefix Supported bit is Clear, the TLP Prefix Log Register is not required to be implemented.

TLP_PREFIX_LOG_i_OFF (i=0-3) Offset address: 0004138_H+i*4

TLP Prefix i Log Register 1 Kernel Reset value: $0000\ 0000_H$



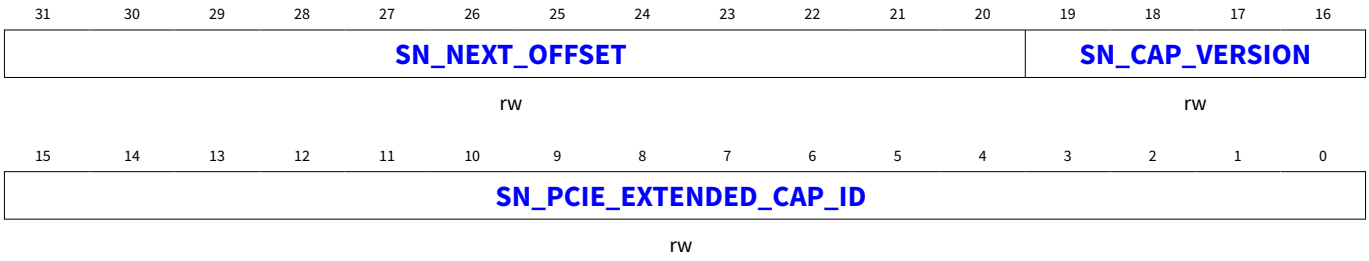
Field	Bits	Type	Description
CFG_TLP_PFX_LOG_1_FIRST_BYTE	7:0	rh	**Byte 0 of Error TLP Prefix Log 1** This field contains first byte of First DW of TLP Prefix. Note: This register field is sticky.
CFG_TLP_PFX_LOG_1_SECOND_BYTE	15:8	rh	**Byte 1 of Error TLP Prefix Log 1** This field contains second byte of First DW of TLP Prefix. Note: This register field is sticky.
CFG_TLP_PFX_LOG_1_THIRD_BYTE	23:16	rh	**Byte 2 of Error TLP Prefix Log 1** This field contains third byte of First DW of TLP Prefix. Note: This register field is sticky.
CFG_TLP_PFX_LOG_1_FOURTH_BYTE	31:24	rh	**Byte 3 of Error TLP Prefix Log 1** This field contains fourth byte of First DW of TLP Prefix. Note: This register field is sticky.

19.4.1.136 Device Serial Number Extended Capability Header

The Device Serial Number Extended Capability is an optional Extended Capability that may be implemented by any PCI Express device Function. This register provides Capability Id, Capability version, and Next offset of Capability structure.

SN_BASE Offset address: 0004148_H

Device Serial Number Extended Capability Header Kernel Reset value: $1581\ 0003_H$



Field	Bits	Type	Description
SN_PCIE_EXTENDED_CAP_ID	15:0	rw	**Serial Number Extended Capability ID** This field is a PCI-SIG defined ID number that indicates the nature and format of the Extended Capability. Extended Capability ID for the Device Serial Number Extended Capability is 0003h. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky.

(table continues...)

(continued)

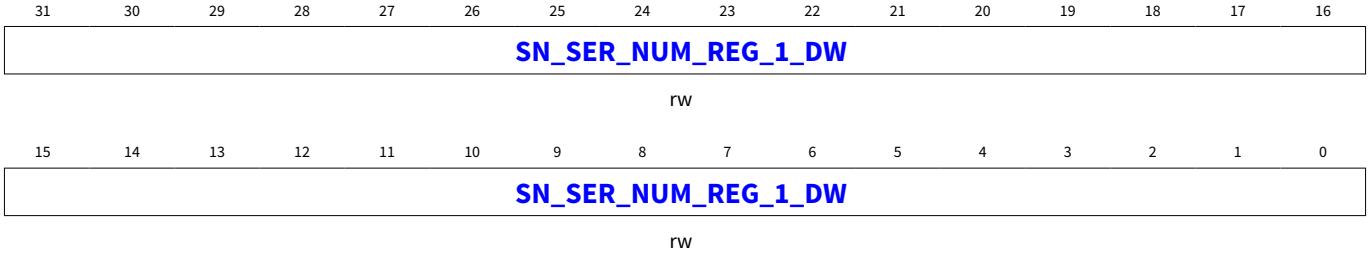
Field	Bits	Type	Description
SN_CAP_VERSION	19:16	rw	**Capability Version** This field is a PCI-SIG defined version number that indicates the version of the Capability structure present. This field is depends on the version of specification. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky.
SN_NEXT_OFFSET	31:20	rw	**Next Capability Offset** This field contains the offset to the next PCI Express Capability structure or 000h if no other items exist in the linked list of Capabilities. For Extended Capabilities implemented in Configuration Space, this offset is relative to the beginning of PCI-compatible Configuration Space and thus must always be either 000h (for terminating list of Capabilities) or greater than 0FFh. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky.

19.4.1.137 Serial Number 1 Register

The Serial Number register is a 64-bit field that contains the IEEE defined 64-bit extended unique identifier.

SER_NUM_REG_DW_1 Offset address: $000414C_H$

Serial Number 1 Register Kernel Reset value: $0000\ 0000_H$



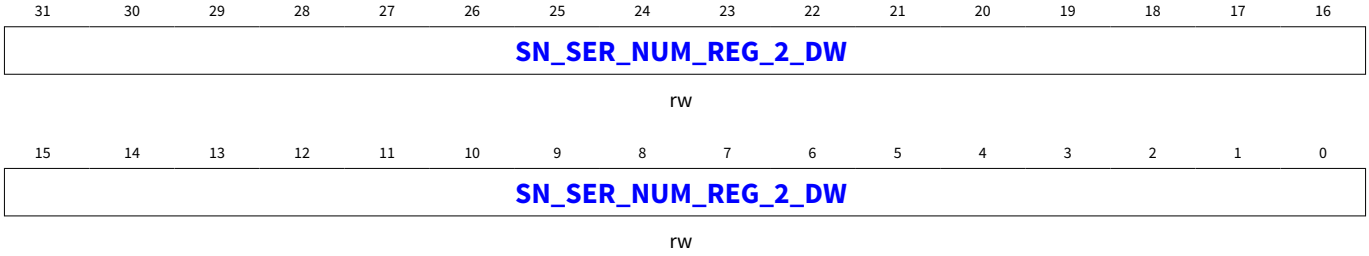
Field	Bits	Type	Description
SN_SER_NUM_REG_1_DW	31:0	rw	**IEEE 64 bit Device Serial Number (DW 1)** This field contains the IEEE defined 64-bit Extended Unique Identifier [EUI-64]. This identifier includes a 24-bit company id value assigned by IEEE registration authority and a 40-bit extension identifier assigned by the manufacturer. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky.

19.4.1.138 Serial Number 2 Register

The Serial Number register is a 64-bit field that contains the IEEE defined 64-bit extended unique identifier.

SER_NUM_REG_DW_2 Offset address: 0004150_H

Serial Number 2 Register Kernel Reset value: $0000\ 0000_H$



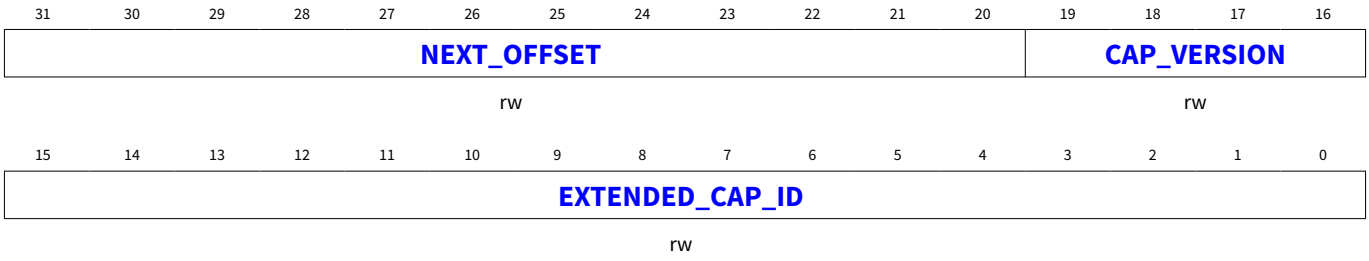
Field	Bits	Type	Description
SN_SER_NUM_REG_2_DW	31:0	rw	**IEEE 64 bit Device Serial Number (DW 2)** This field contains the IEEE defined 64-bit Extended Unique Identifier [EUI-64]. This identifier includes a 24-bit company id value assigned by IEEE registration authority and a 40-bit extension identifier assigned by the manufacturer. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky.

19.4.1.139 SPCIE Capability Header

This Register provides Capability Id, Capability Version, and next Offset of SPCIE Structure.

SPCIE_CAP_HEADER_REG Offset address: 0004158_H

SPCIE Capability Header Kernel Reset value: $1681\ 0019_H$



Field	Bits	Type	Description
EXTENDED_CAP_ID	15:0	rw	**Secondary PCI Express Extended Capability ID** This field is a PCI-SIG defined ID number that indicates the nature and format of the Extended Capability. PCI Express Extended Capability ID for the Secondary PCI Express Extended Capability is 0019h. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky. 0000_H **MIN_VAL**: Min value $FFFF_H$ **MAX_VAL**: Max value

(table continues...)

(continued)

Field	Bits	Type	Description
CAP_VERSION	19:16	rw	**Capability Version** This field is a PCI-SIG defined version number that indicates the version of the Capability structure present. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky. 0_H **MIN_VAL**: Min value F_H **MAX_VAL**: Max value
NEXT_OFFSET	31:20	rw	**Next Capability Offset** This field contains the offset to the next PCI Express Extended Capability structure or 000h if no other items exist in the linked list of Capabilities. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky. 000_H **MIN_VAL**: Min value FFF_H **MAX_VAL**: Max value

19.4.1.140 Link Control 3 Register

This Register controls equilization and equilization interrupt.

LINK_CONTROL3_REG Offset address: $000415C_H$

Link Control 3 Register Reset values see: Table 873

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_2															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVDP_2														EQ_REQ_INT_EN	PERFORM_EQ
r														r	rh

Field	Bits	Type	Description
PERFORM_EQ	0	rh	**Perform Equalization** This bit is RW for Downstream Ports and for Upstream Ports when Crosslink Supported is 1b. This bit is not applicable and is RsvdP for Upstream Ports when the Crosslink Supported bit is 0b. Note: The access attributes of this field are as follows: - Wire: RSVDP - Dbi: RSVDP 0_B **CLEAR**: Clear 1_B **SET**: When this bit is 1b and a 1b is written to the Retrain Link bit with the Target Link Speed field set to 8.0 GT/s or higher, the Downstream Port must perform Link Equalization

(table continues...)

(continued)

Field	Bits	Type	Description
EQ_REQ_INT_EN	1	r	**Link Equalization Request Interrupt Enable** This bit is RW for Downstream Ports and for Upstream Ports when Crosslink Supported is 1b. This bit is not applicable and is RsvdP for Upstream Ports when the Crosslink Supported bit is 0b. Note: The access attributes of this field are as follows: - Wire: RSVDP - Dbi: RSVDP 0_B **CLEAR**: Clear 1_B **SET**: When Set, this bit enables the generation of an interrupt to indicate that the Link Equalization Request 8.0 GT/s bit, the Link Equalization Request 16.0 GT/s bit, or the Link Equalization Request 32.0 GT/s bit has been set
RSVDP_2	31:2	r	**Reserved for future use**

Table 873 Reset values of LINK_CONTROL3_REG

Reset	Reset value	Note
Kernel Reset	0000 0000 0000 0000 0000 0000 0000 000X_B	

19.4.1.141 Lane Error Status Register

This Register contains Lane Error Status Bits per Lane.

LANE_ERR_STATUS_REG Offset address: 0004160_H

Lane Error Status Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_LANE_ERR_STATUS															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVDP_LANE_ERR_STATUS															LANE_ERR_STATUS
r															rw1ch

Field	Bits	Type	Description
LANE_ERR_STATUS	0	rw1ch	**Lane Error Status Bits per Lane** Each bit indicates if the corresponding Lane detected a Lane-based error. A value of 1b indicates that a Lane based-error was detected on the corresponding Lane Number. For Ports that are narrower than 32 Lanes, the unused upper bits [31: Maximum Link Width] are RsvdZ. Note: This register field is sticky. Software write with 1 clears this bit-field.
RSVDP_LANE_ERR_STATUS	31:1	r	**Reserved for future use**

19.4.1.142 Lane Equalization Control Register for lanes 1 and 0

This register provides Transmitter Preset and Receiver Preset Hint for Downstream Port and Upstream Port.

SPCIE_CAP_OFF_0CH_REG Offset address: 0004164_H

Lane Equalization Control Register for lanes 1 and 0 Kernel Reset value: $0000\ 7400_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_31	0							RSVDP_23	0						
r	r							r	r						

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVDP_15	USP_RX_PRESET_HINT0			USP_TX_PRESET0				RSVDP_7	DSP_RX_PRESET_HINT0			DSP_TX_PRESET0			
r	r			r				r	r			r			

Field	Bits	Type	Description
DSP_TX_PRESET0	3:0	r	**Downstream Port 8.0 GT/s Transmitter Preset 0** Transmitter preset 0 value used for 8.0 GT/s equalization by this Port when the Port is operating as a Downstream Port. This field is ignored when the Port is operating as an Upstream Port. For an Upstream Port if Crosslink Supported is 0b, this field is RsvdP. Otherwise, this field is HwInit. Note: The access attributes of this field are as follows: - Wire: RSVDP - Dbi: RSVDP
DSP_RX_PRESET_HINT0	6:4	r	**Downstream Port 8.0 GT/s Receiver Preset Hint 0** Receiver preset hint 0 value that may be used as a suggested setting for 8.0 GT/s receiver equalization by this Port when the Port is operating as a Downstream Port. This field is ignored when the Port is operating as an Upstream Port. For an Upstream Port if Crosslink Supported is 0b, this field is RsvdP. Otherwise, this field is HwInit. Note: The access attributes of this field are as follows: - Wire: RSVDP - Dbi: RSVDP
RSVDP_7	7	r	**Reserved for future use**

(table continues...)

(continued)

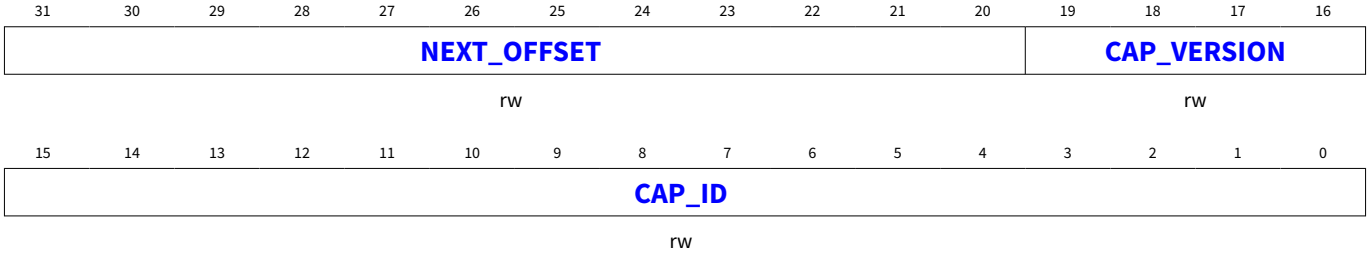
Field	Bits	Type	Description
USP_TX_PRESET0	11:8	r	**Upstream Port 8.0 GT/s Transmitter Preset 0** The write value is gated with the PCIE_CAP_CROSS_LINK_SUPPORT field of LINK_CAPABILITIES2_REG. Field contains the Transmitter preset 0 value sent or received during 8.0 GT/s Link Equalization. • Case A: When Operating port is Downstream port and whether the Crosslink is supported or not, this field representss the value sent on Lane 0 during 8.0 GT/s equalization • Case B: When Operating port is Upstream port and Crosslink Supported bit is 0, this field is intended for debug and diagnostics. It contains the value captured from Lane 0 during Link Equalization • Case C: When Operating port is Upstream port and Crosslink Supported bit is 1, Field is not used or affected by the current Link Equalization. Field value will be used if a future crosslink negotiation switches the Operating Port Direction so that case A applies For case A and C, Field is HwInit. For case B, Field is RO. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: R (sticky) Note: This register field is sticky.
USP_RX_PRESET_HINT0	14:12	r	**Upstream Port 8.0 GT/s Receiver Preset Hint 0** The write value is gated with the PCIE_CAP_CROSS_LINK_SUPPORT field of LINK_CAPABILITIES2_REG. Field contains the Receiver preset hint 0 value sent or received during 8.0 GT/s Link Equalization. • Case A: When Operating port is Downstream port and whether the Crosslink is supported or not, this field representss the value sent on Lane 0 during 8.0 GT/s equalization • Case B: When Operating port is Upstream port and Crosslink Supported bit is 0, this field is intended for debug and diagnostics. It contains the value captured from Lane 0 during Link Equalization • Case C: When Operating port is Upstream port and Crosslink Supported bit is 1, Field is not used or affected by the current Link Equalization. Field value will be used if a future crosslink negotiation switches the Operating Port Direction so that case A applies For case A and C, Field is HwInit. For case B, Field is RO. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: R (sticky) Note: This register field is sticky.
RSVDP_15	15	r	**Reserved for future use**
RSVDP_23	23	r	**Reserved for future use**
RSVDP_31	31	r	**Reserved for future use**
0	22:16, 30:24	r	**Reserved** Read as all 0's; should be written with all 0's.

19.4.1.143 LTR Extended Capability Header

This register provides capbility ID, capability version and next offset value for LTR(Latency Tolerance Reporting).

LTR_CAP_HDR_REG Offset address: 0004168_H

LTR Extended Capability Header Kernel Reset value: $1701\ 0018_H$



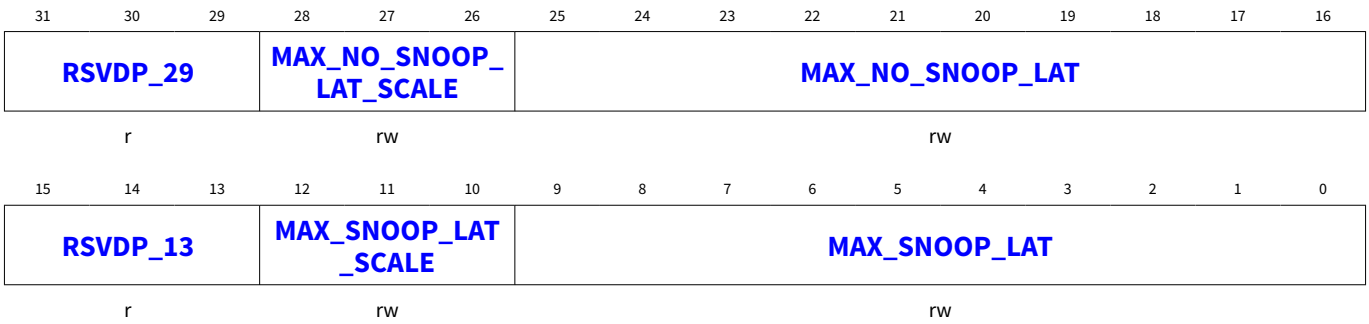
Field	Bits	Type	Description
CAP_ID	15:0	rw	**LTR Extended Capacity ID** This field is a PCI-SIG defined ID number that indicates the nature and format of the Extended Capability. PCI Express Extended Capability for the LTR Extended Capability is 0018h. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky.
CAP_VERSION	19:16	rw	**Capability Version** This field is a PCI-SIG defined version number that indicates the version of the Capability structure present. This field is depends on version of the specification. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky.
NEXT_OFFSET	31:20	rw	**Next Capability Offset** This field contains the offset to the next PCI Express Extended Capability structure or 000h if no other items exist in the linked list of Capabilities. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky.

19.4.1.144 LTR Max Snoop and No-Snoop Latency Register

This register indicates Latency scale and vlaue for Max Snoop and No-Snoop.

LTR_LATENCY_REG Offset address: $000416C_H$

LTR Max Snoop and No-Snoop Latency Register Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
MAX_SNOOP_LAT	9:0	rw	**Max Snoop Latency Value** Along with the Max Snoop LatencyScale field,this register specifies the maximum snoop latency that a device is permitted to request. Software should set this to the maximum supported latency or less. It is strongly recommended that any updates to this field are reflected in LTR Message(s) sent by the device within 1 ms.
MAX_SNOOP_LAT_SCALE	12:10	rw	**Max Snoop Latency Scale** This register provides a scale for the value contained within the Max Snoop LatencyValue field. Encoding is the same as the LatencyScale fields in the LTR Message. It is strongly recommended that any updates to this field are reflected in LTR Message(s) sent by the device within 1 ms. Hardware operation is undefined if software writes a Not Permitted value to this field.
RSVDP_13	15:13	r	**Reserved for future use**
MAX_NO_SNOOP_LAT	25:16	rw	**Max No-Snoop Latency Value** Along with the Max No-Snoop LatencyScale field, this register specifies the maximum no-snoop latency that a device is permitted to request. Software should set this to the maximum supported latency or less. It is strongly recommended that any updates to this field are reflected in LTR Message(s) sent by the device within 1 ms.
MAX_NO_SNOOP_LAT_SCALE	28:26	rw	**Max No-Snoop Latency Scale** This register provides a scale for the value contained within the Max No-Snoop LatencyValue field. Encoding is the same as the LatencyScale fields in the LTR Message. It is strongly recommended that any updates to this field are reflected in LTR Message(s) sent by the device within 1 ms. Hardware operation is undefined if software writes a Not Permitted value to this field.
RSVDP_29	31:29	r	**Reserved for future use**

19.4.1.145 L1 Substates Extended Capability Header

L1 Substates Extended Capability Header provides capability ID, capability version and next offset value for L1 Substates.

L1SUB_CAP_HEADER_REG Offset address: 0004170_H

L1 Substates Extended Capability Header Kernel Reset value: $1801\ 001E_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
NEXT_OFFSET												CAP_VERSION			
rw												rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EXTENDED_CAP_ID															
rw															

Field	Bits	Type	Description
EXTENDED_CAP_ID	15:0	rw	**L1SUB Extended Capability ID** This field is a PCI-SIG defined ID number that indicates the nature and format of the Extended Capability. Extended Capability ID for L1 PM Substates is 001Eh. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky.
CAP_VERSION	19:16	rw	**Capability Version** This field is a PCI-SIG defined version number that indicates the version of the Capability structure present. This field is depends on version of the specification. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky.
NEXT_OFFSET	31:20	rw	**Next Capability Offset** This field contains the offset to the next PCI Express Capability structure or 000h, if no other items exist in the linked list of Capabilities. For Extended Capabilities implemented in Configuration Space, this offset is relative to the beginning of PCI-compatible Configuration Space and thus must always be either 000h (for terminating list of Capabilities) or greater than 0FFh. The bottom 2 bits of this offset are Reserved and must be implemented as 00b although software must mask them to allow for future uses of these bits. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky.

19.4.1.146 L1 Substates Capability Register

This register provides extended capability of L1 Substates.

L1SUB_CAPABILITY_REG Offset address: 0004174_H

L1 Substates Capability Register Kernel Reset value: $0038\ 0A1F_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_24								PWR_ON_VALUE_SUPPORT					RSVDP_18	PWR_ON_SCALE_SUPPORT	
r								rw					r	rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
COMM_MODE_SUPPORT								RSVDP_5			L1_PMSUB_SUPPORT	L1_1_ASPM_SUPPORT	L1_2_ASPM_SUPPORT	L1_1_PCIPM_SUPPORT	L1_2_PCIPM_SUPPORT
rw								r			rw	rw	rw	rw	rw

Field	Bits	Type	Description
L1_2_PCIPM_SUPPORT	0	rw	**PCI-PM L12 Supported** When Set this field indicates that PCI-PM L1.2 is supported. Note: The access attributes of this field are as follows: - Wire: HWINIT - Dbi: R/W (sticky)
L1_1_PCIPM_SUPPORT	1	rw	**PCI-PM L11 Supported** When Set this field indicates that PCI-PM L1.1 is supported, and must be Set by all Ports implementing L1 PM Substates. Note: The access attributes of this field are as follows: - Wire: HWINIT - Dbi: R/W (sticky)
L1_2_ASPM_SUPPORT	2	rw	**ASPM L12 Supported** When Set this field indicates that ASPM L1.2 is supported. Note: The access attributes of this field are as follows: - Wire: HWINIT - Dbi: R/W (sticky)
L1_1_ASPM_SUPPORT	3	rw	**ASPM L11 Supported** When Set this field indicates that ASPM L1.1 is supported. Note: The access attributes of this field are as follows: - Wire: HWINIT - Dbi: R/W (sticky)
L1_PMSUB_SUPPORT	4	rw	**L1 PM Substates ECN Supported** When Set this field indicates that this Port supports L1 PM Substates. Note: The access attributes of this field are as follows: - Wire: HWINIT - Dbi: R/W (sticky)
RSVDP_5	7:5	r	**Reserved for future use**
COMM_MODE_SUPPORT	15:8	rw	**Port Common Mode Restore Time** Time (in µs) required for this Port to re-establish common mode. Required for all Ports for which either the PCI-PM L1.2 Supported bit is Set, ASPM L1.2 Supported bit is Set, or both are Set, otherwise this field is of type RsvdP. Note: The access attributes of this field are as follows: - Wire: HWINIT - Dbi: L1SUB_CAPABILITY_REG.L1_2_ASPM_SUPPORT|| L1SUB_CAPABILITY_REG.L1_2_PCIPM_SUPPORT ? RWS : RSVDP
PWR_ON_SCALE_SUPPORT	17:16	rw	**Port T Power On Scale** Specifies the scale used for the Port T_POWER_ON_VALUE field in the L1 PM Substates Capabilities register. Range of values are given below. Required for all Ports for which either the PCI-PM L1.2 Supported bit is Set, ASPM L1.2 Supported bit is Set, or both are Set, otherwise this field is of type RsvdP. Note: The access attributes of this field are as follows: - Wire: HWINIT - Dbi: L1SUB_CAPABILITY_REG.L1_2_ASPM_SUPPORT|| L1SUB_CAPABILITY_REG.L1_2_PCIPM_SUPPORT ? RWS : RSVDP 00_B **_2us**: Scale is 2 µs 01_B **_10us**: Scale is 10 µs 10_B **_100us**: Scale is 100 µs 11_B **Reserved**: Reserved
RSVDP_18	18	r	**Reserved for future use**

(table continues...)

(continued)

Field	Bits	Type	Description
PWR_ON_VALUE_SUPPORT	23:19	rw	**Port T Power On Value** Along with the Port T_POWER_ON_SCALE field in the L1 PM Substates Capabilities register sets the time (in µs) that this Port requires the port on the opposite side of Link to wait in L1.2.Exit after sampling CLKREQ# asserted before actively driving the interface. The value of Port T_POWER_ON is calculated by multiplying the value in this field by the scale value in the Port T_POWER_ON_SCALE field in the L1 PM Substates Capabilities register. Required for all Ports for which either the PCI-PM L1.2 Supported bit is Set, ASPM L1.2 Supported bit is Set, or both are Set, otherwise this field is of type RsvdP. Note: The access attributes of this field are as follows: • Wire: HWINIT • Dbi: L1SUB_CAPABILITY_REG.L1_2_ASPM_SUPPORT|| L1SUB_CAPABILITY_REG.L1_2_PCIPM_SUPPORT ? RWS : RSVDP
RSVDP_24	31:24	r	**Reserved for future use**

19.4.1.147 L1 Substates Control 1 Register

This register Controls that the individual extended capability is enabled ot not.

L1SUB_CONTROL1_REG Offset address: 0004178_H

L1 Substates Control 1 Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
L1_2_TH_SCA			RSVDP_26			L1_2_TH_VAL									
rw			r			rw									

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
T_COMMON_MODE								RSVDP_4				L1_1_ASPM_EN	L1_2_ASPM_EN	L1_1_PCIPM_EN	L1_2_PCIPM_EN
r								r				rw	rw	rw	rw

Field	Bits	Type	Description
L1_2_PCIPM_EN	0	rw	**PCI-PM L12 Enable** When Set this field, enables PCI-PM L1.2. For Ports for which the PCI-PM L1.2 Supported bit is clear, this bit is permitted to be hardwired to 0. Required for both Upstream and Downstream Ports. For compatibility with possible future extensions, software must not enable L1 PM Substates unless the L1 PM Substates Supported field in the L1 PM Substates Capabilities Register is Set.
L1_1_PCIPM_EN	1	rw	**PCI-PM L11 Enable** When Set this field, enables PCI-PM L1.1. Required for both Upstream and Downstream Ports. For compatibility with possible future extensions, software must not enable L1 PM Substates unless the L1 PM Substates Supported field in the L1 PM Substates Capabilities Register is Set.

(table continues...)

(continued)

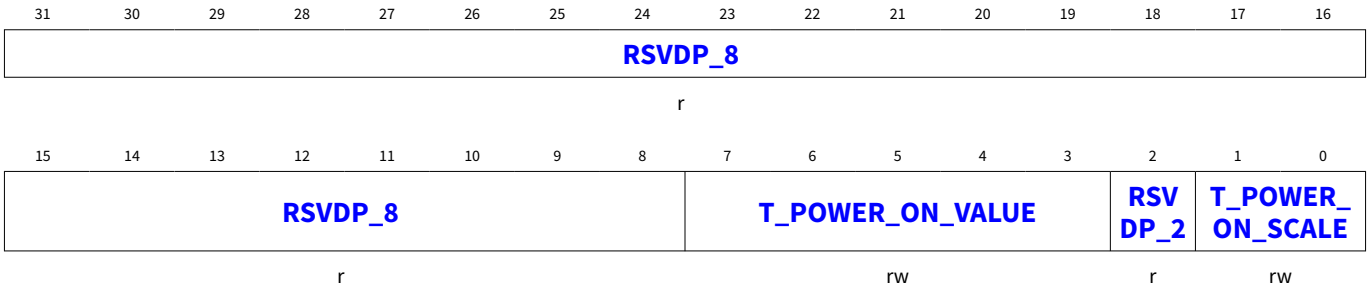
Field	Bits	Type	Description
L1_2_ASPM_EN	2	rw	**ASPM L12 Enable** When Set this field, enables ASPM L1.2. For Ports for which the ASPM L1.2 Supported bit is clear, this bit is permitted to be hardwired to 0. Required for both Upstream and Downstream Ports. For compatibility with possible future extensions, software must not enable L1 PM Substates unless the L1 PM Substates Supported field in the L1 PM Substates Capabilities Register is Set.
L1_1_ASPM_EN	3	rw	**ASPM L11 Enable** When Set this field, enables ASPM L1.1. For Ports for which the ASPM L1.1 Supported bit is clear, this bit is permitted to be hardwired to 0. Required for both Upstream and Downstream Ports. For compatibility with possible future extensions, software must not enable L1 PM Substates unless the L1 PM Substates Supported field in the L1 PM Substates Capabilities Register is Set.
RSVDP_4	7:4	r	**Reserved for future use**
T_COMMON_MODE	15:8	r	**Common Mode Restore Time** Sets value of TCOMMONMODE (in µs), which must be used by the Downstream Ports for timing the re-establishment of common mode. This field is of type RsvdP for Upstream Ports.
L1_2_TH_VAL	25:16	rw	**LTR L12 Threshold Value** Along with the LTR_L1.2_THRESHOLD_Scale, this field indicates the LTR threshold used to determine if entry into L1 results in L1.1 (if enabled) or L1.2 (if enabled). Required for all Ports for which the ASPM L12 Supported bit is Set, otherwise this field is of type RsvdP. Note: The access attributes of this field are as follows • Wire: L1SUB_CAPABILITY_REG.L1_2_ASPM_SUPPORT ? RW : RSVDP • Dbi: L1SUB_CAPABILITY_REG.L1_2_ASPM_SUPPORT ? RW : RSVDP
RSVDP_26	28:26	r	**Reserved for future use**
L1_2_TH_SCA	31:29	rw	**LTR L12 Threshold Scale** This field provides a scale for the value contained within the LTR_L1.2_THRESHOLD_Value. Hardware operation is undefined if software writes a Not-Permitted value to this field. Required for all Ports Ports for which the ASPM L12 Supported bit is Set, otherwise this field is of type RsvdP. Note: The access attributes of this field are as follows: • Wire: L1SUB_CAPABILITY_REG.L1_2_ASPM_SUPPORT ? RW : RSVDP • Dbi: L1SUB_CAPABILITY_REG.L1_2_ASPM_SUPPORT ? RW : RSVD

19.4.1.148 L1 Substates Control 2 Register

This register Controls that the individual extended capability is enabled ot not.

L1SUB_CONTROL2_REG Offset address: $000417C_H$

L1 Substates Control 2 Register Kernel Reset value: $0000\ 0028_H$



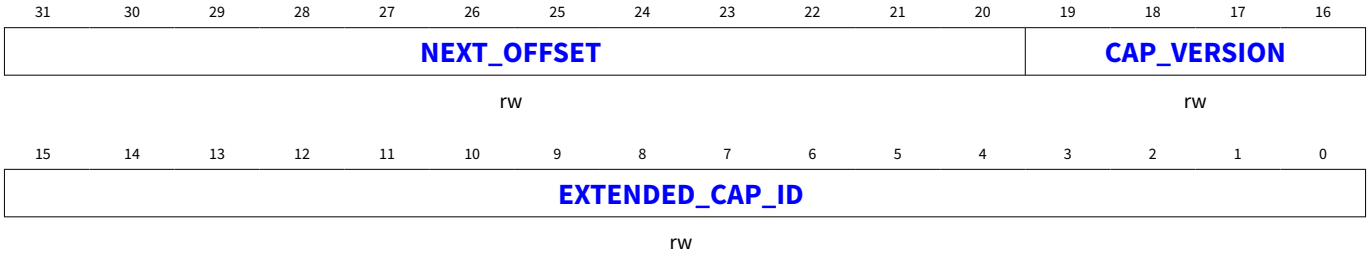
Field	Bits	Type	Description
T_POWER_ON_SCALE	1:0	rw	**T Power On Scale** Specifies the scale used for T_POWER_ON_VALUE. Range of values are given below. Required for all Ports that support L1.2, otherwise this field is of type RsvdP. This field must only be modified when the ASPM L1.2 Enable and PCI-PM L1.2 Enable bits are both Clear. The Port behavior is undefined if this field is modified when either the ASPM L1.2 Enable and/or PCI-PM L1.2 Enable bit(s) are set. Note: The access attributes of this field are as follows: • Wire: L1SUB_CAPABILITY_REG.L1_2_ASPM_SUPPORT|| L1SUB_CAPABILITY_REG.L1_2_PCIPM_SUPPORT ? RWS : RSVDP • Dbi: L1SUB_CAPABILITY_REG.L1_2_ASPM_SUPPORT|| L1SUB_CAPABILITY_REG.L1_2_PCIPM_SUPPORT ? RWS : RSVDP 00_B **_2us**: Scale is 2 µs 01_B **_10us**: Scale is 10 µs 10_B **_100us**: Scale is 100 µs 11_B **Reserved**: Reserved
RSVDP_2	2	r	**Reserved for future use**
T_POWER_ON_VALUE	7:3	rw	**T Power On Value** Along with the T_POWER_ON_SCALE sets the minimum amount of time (in µs) that the Port must wait in L1.2.Exit after sampling CLKREQ# asserted before actively driving the interface. T_POWER_ON is calculated by multiplying the value in this field by the value in the T_POWER_ON_SCALE field. Required for all Ports that support L1.2, otherwise this field is of type RsvdP.This field must only be modified when the ASPM L1.2 Enable and PCI-PM L1.2 Enable bits are both Clear. The Port behavior is undefined if this field is modified when either the ASPM L1.2 Enable and/or PCI-PM L1.2 Enable bit(s) are set. Note: The access attributes of this field are as follows: • Wire: L1SUB_CAPABILITY_REG.L1_2_ASPM_SUPPORT|| L1SUB_CAPABILITY_REG.L1_2_PCIPM_SUPPORT ? RWS : RSVDP • Dbi: L1SUB_CAPABILITY_REG.L1_2_ASPM_SUPPORT|| L1SUB_CAPABILITY_REG.L1_2_PCIPM_SUPPORT ? RWS : RSVDP
RSVDP_8	31:8	r	**Reserved for future use**

19.4.1.149 Vendor-Specific Extended Capability Header

The Vendor-Specific Extended Capability (VSEC Capability) is an optional Extended Capability that is permitted to be implemented by any PCI Express Function or RCRB. This Register contains Capability Id, Capability Version and Next Offset value for Vendor-Specific Extended Capability.

RAS_DES_CAP_HEADER_REG Offset address: 0004180_H

Vendor-Specific Extended Capability Header Kernel Reset value: $2801\ 000B_H$



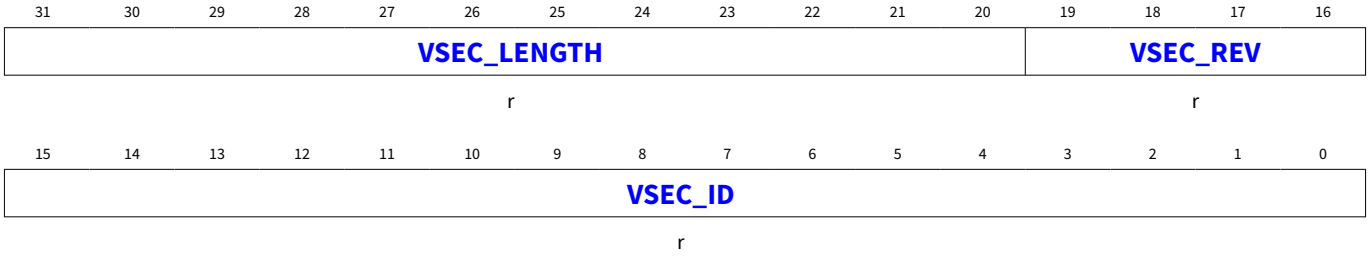
Field	Bits	Type	Description
EXTENDED_CAP_ID	15:0	rw	**PCI Express Extended Capability ID** This field is a PCI-SIG defined ID number that indicates the nature and format of the Extended Capability. Extended Capability ID for the Vendor-Specific Extended Capability is 000Bh. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky. 0000_H **MIN_VAL**: Min value $FFFF_H$ **MAX_VAL**: Max value
CAP_VERSION	19:16	rw	**Capability Version** This field is a PCI-SIG defined version number that indicates the version of the Capability structure present. Value of this field is depends on the version of the specification. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky. 0_H **MIN_VAL**: Min value F_H **MAX_VAL**: Max value
NEXT_OFFSET	31:20	rw	**Next Capability Offset** This field contains the offset to the next PCI Express Capability structure or 000h if no other items exist in the linked list of Capabilities. For Extended Capabilities implemented in Configuration Space, this offset is relative to the beginning of PCI-compatible Configuration Space and thus must always be either 000h (for terminating list of Capabilities) or greater than 0FFh. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky. 000_H **MIN_VAL**: Min value FFF_H **MAX_VAL**: Max value

19.4.1.150 Vendor-Specific Header

This Register field provides VSEC Length, VSEC ID and VSEC Rev(Version Number). Vendor-specific software must qualify the associated Vendor ID of the PCI Express Function or RCRB before attempting to interpret the values in the VSEC ID or VSEC Rev fields.

VENDOR_SPECIFIC_HEADER_REG Offset address: 0004184_H

Vendor-Specific Header Kernel Reset value: $1004\ 0002_H$



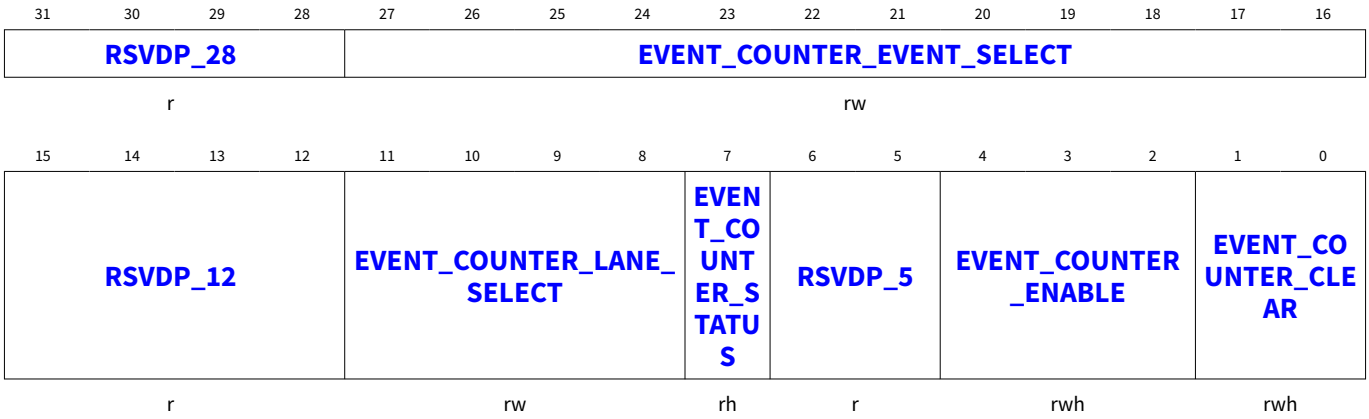
Field	Bits	Type	Description
VSEC_ID	15:0	r	**VSEC ID** This field is a vendor-defined ID number that indicates the nature and format of the VSEC structure. Software must qualify the Vendor ID before interpreting this field. 0000_H **MIN_VAL**: Min value $FFFF_H$ **MAX_VAL**: Max value
VSEC_REV	19:16	r	**VSEC Rev** This field is a vendor-defined version number that indicates the version of the VSEC structure. Software must qualify the Vendor ID and VSEC ID before interpreting this field. 0_H **MIN_VAL**: Min value F_H **MAX_VAL**: Max value
VSEC_LENGTH	31:20	r	**VSEC Length** This field indicates the number of bytes in the entire VSEC structure, including the Vendor-Specific Extended Capability Header, the Vendor-Specific Header, and the vendor-specific registers. 000_H **MIN_VAL**: Min value FFF_H **MAX_VAL**: Max value

19.4.1.151 Event Counter Control

This is a viewport control register. - Setting the EVENT_COUNTER_EVENT_SELECT and EVENT_COUNTER_LANE_SELECT fields in this register determine the Event Counter data returned by the EVENT_COUNTER_DATA_REG viewport register. - Setting the EVENT_COUNTER_ENABLE field in this register enables the Event Counter selected by the EVENT_COUNTER_EVENT_SELECT and EVENT_COUNTER_LANE_SELECT fields in this register. - Setting the EVENT_COUNTER_CLEAR field in this register clears the Event Counter selected by the EVENT_COUNTER_EVENT_SELECT and EVENT_COUNTER_LANE_SELECT fields in this register. - Reading the EVENT_COUNTER_STATUS field in this register returns the Enable status of the Event Counter selected by the EVENT_COUNTER_EVENT_SELECT and EVENT_COUNTER_LANE_SELECT fields in this register.

EVENT_COUNTER_CONTROL_REG Offset address: 0004188_H

Event Counter Control Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
EVENT_COUNTER_CLEAR	1:0	rwh	**Event Counter Clear** Clears the Event Counter selected by the EVENT_COUNTER_EVENT_SELECT and EVENT_COUNTER_LANE_SELECT fields in this register. You can clear the value of a specific Event Counter by writing the 'per clear' code and you can clear all event counters at once by writing the 'all clear' code. The read value is always '0'. Other values are reserved. 00_B **NO_CHANGE**: no change 01_B **PER_CLEAR**: per clear 10_B **NO_CHANGE_2**: no change 11_B **ALL_CLEAR**: all clear
EVENT_COUNTER_ENABLE	4:2	rwh	**Event Counter Enable** Enables/disables the Event Counter selected by the EVENT_COUNTER_EVENT_SELECT and EVENT_COUNTER_LANE_SELECT fields in this register. By default, all event counters are disabled. You can enable/disable a specific Event Counter by writing the 'per event off' or 'per event on' codes. You can enable/disable all event counters by writing the 'all on' or 'all off' codes. The read value is always '0'. For other values no change. 001_B **PER_EVENT_OFF**: per event off 011_B **PER_EVENT_ON**: per event on 101_B **ALL_OFF**: all off 111_B **ALL_ON**: all on
RSVDP_5	6:5	r	**Reserved for future use**
EVENT_COUNTER_STATUS	7	rh	**Event Counter Status** This register returns the current value of the Event Counter selected by the following fields: - EVENT_COUNTER_EVENT_SELECT - EVENT_COUNTER_LANE_SELECT Note: This register field is sticky. 0_B **CLEAR**: Clear 1_B **SET**: Set

(table continues...)

(continued)

Field	Bits	Type	Description
EVENT_COUNTER_LANE_SELECT	11:8	rw	**Event Counter Lane Select** This field in conjunction with EVENT_COUNTER_EVENT_SELECT indexes the Event Counter data returned by the EVENT_COUNTER_DATA_REG register. Note: This register field is sticky. 0_H **MIN_VAL**: Lane0 F_H **MAX_VAL**: Lane15
RSVDP_12	15:12	r	**Reserved for future use**
EVENT_COUNTER_EVENT_SELECT	27:16	rw	**Event Counter Data Select** This field in conjunction with the EVENT_COUNTER_LANE_SELECT field indexes the Event Counter data returned by the EVENT_COUNTER_DATA_REG register • 27-24: Group number(4-bit: 0..0x7) • 23-16: Event number(8-bit: 0..0x13) within the Group Note: This register field is sticky. 000_H **EBUF_OVERFLOW**: Ebuf Overflow 001_H **EBUF_UNDERRUN**: Ebuf Underrun 700_H **TX_MEM_WRITE**: Tx Memory Write 713_H **RX_MSG_TLP**: Rx Message TLP **others,** Reserved
RSVDP_28	31:28	r	**Reserved for future use**

19.4.1.152 Event Counter Data

This viewport register returns the data selected by the following fields: - EVENT_COUNTER_EVENT_SELECT in EVENT_COUNTER_CONTROL_REG - EVENT_COUNTER_LANE_SELECT in EVENT_COUNTER_CONTROL_REG

EVENT_COUNTER_DATA_REG Offset address: $000418C_H$

Event Counter Data Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EVENT_COUNTER_DATA															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EVENT_COUNTER_DATA															
rh															

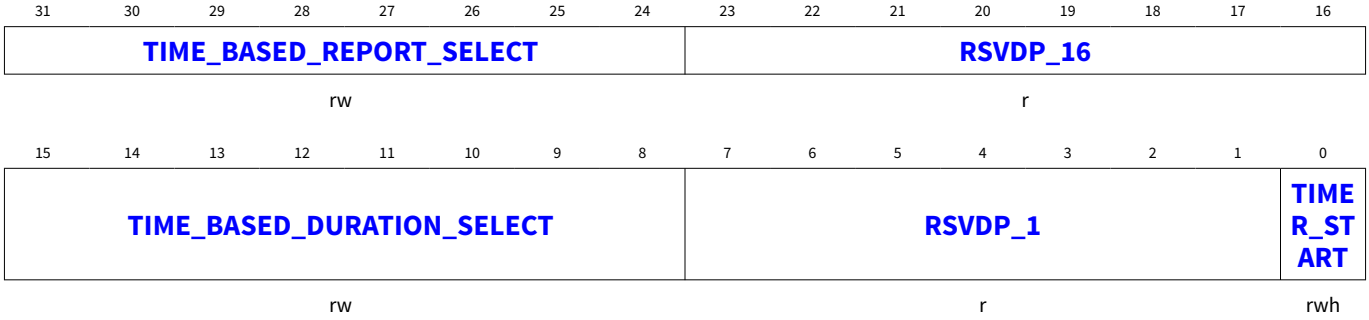
Field	Bits	Type	Description
EVENT_COUNTER_DATA	31:0	rh	**Event Counter Data** This register returns the data selected by the following fields: - EVENT_COUNTER_EVENT_SELECT in EVENT_COUNTER_CONTROL_REG Note: This register field is sticky. 00000000_H **MIN_VAL**: Min value $FFFFFFFF_H$ **MAX_VAL**: Max value

19.4.1.153 Time-based Analysis Control

Used for controlling the measurement of RX/TX data throughput and time spent in each low-power LTSSM state.

TIME_BASED_ANALYSIS_CONTROL_REG Offset address: 0004190_H

Time-based Analysis Control Kernel Reset value: $0000\ 0100_H$



Field	Bits	Type	Description
TIMER_START	0	rwh	**Timer Start** This bit will be cleared automatically when the measurement is finished. Note: This register field is sticky. 0_B **STOP**: Stop 1_B **START_RESTART**: Start/Restart
RSVDP_1	7:1	r	**Reserved for future use**
TIME_BASED_DURATION_SELECT	15:8	rw	**Time-based Duration Select** Selects the duration of time-based analysis. When "manual control" is selected and TIMER_START is set to '1', this analysis never stops until TIMER_STOP is set to '0'. All encodings other than the defined encodings are reserved. Note: This register field is sticky. 00_H **MANUAL**: Manual control 01_H **_1_MS**: 1ms 02_H **_2_MS**: 10ms 03_H **_100_MS**: 100ms 04_H **_1_S**: 1s 05_H **_2_S**: 2s 06_H **_4_S**: 4s FF_H **_4_US**: 4 µs (Debug purpose)
RSVDP_16	23:16	r	**Reserved for future use**

(table continues...)

(continued)

Field	Bits	Type	Description
TIME_BASED_REPORT_SELECT	31:24	rw	**Time-based Report Select** Selects what type of data is measured for the selected duration (TIME_BASED_DURATION_SELECT), and returned in TIME_BASED_ANALYSIS_DATA. Each type of data is measured using one of three types of units: • Core_clk Cycles for 2.5GT/s, 5.0GT/s, 8.0GT/s. Total time in ps is [Value of TIME_BASED_ANALYSIS_DATA returned when TIME_BASED_REPORT_SELECT=0x00] * TIME_BASED_ANALYSIS_DATA. Values 0x0-0x4 and 0x7-0x8 correspond to Core_clk Cycles for 2.5GT/s, 5.0GT/s, 8.0GT/s. • Aux_clk Cycles. Total time in ps is [Period of platform specific clock] * TIME_BASED_ANALYSIS_DATA. Values 0x5, 0x6, and 0x9 correspond to aux_clk Cycles • Data Bytes. Actual amount of bytes is 16 * TIME_BASED_ANALYSIS_DATA. Values 0x20-0x23 correspond to data bytes All encodings other than the defined encodings are reserved. Note: This register field is sticky. 00_H **ONE_CYCLE**: Duration of 1 cycle 01_H **TX_L0S**: TxL0s 02_H **RX_L0S**: RxL0s 03_H **L0**: L0 04_H **L1**: L1 05_H **L1_1**: L1.1 06_H **L1_2**: L1.2 07_H **CFG_RCVRY**: Configuration/Recovery 08_H **TX_RX_L0S**: TxL0s and RxL0s 09_H **L1_AUX**: L1 aux 20_H **TX_PCIE_TLP**: Tx PCIe TLP data payload Bytes 21_H **RX_PCIE_TLP**: Rx PCIe TLP data payload Bytes 22_H **TX_CCIX_TLP**: Tx CCIX TLP data payload Bytes 23_H **RX_CCIX_TLP**: Rx CCIX TLP data payload Bytes **others,** Reserved

19.4.1.154 Time-based Analysis Data

Contains the measurement results of RX/TX data throughput and time spent in each low-power LTSSM state. This viewport register returns the data selected by the TIME_BASED_REPORT_SELECT field in TIME_BASED_ANALYSIS_CONTROL_REG.

TIME_BASED_ANALYSIS_DATA_REG Offset address: 0004194_H

Time-based Analysis Data Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TIME_BASED_ANALYSIS_DATA															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TIME_BASED_ANALYSIS_DATA															
rh															

Field	Bits	Type	Description
TIME_BASED_ANALYSIS_DATA	31:0	rh	**Time Based Analysis Data** This register returns the data selected by the TIME_BASED_REPORT_SELECT field in TIME_BASED_ANALYSIS_CONTROL_REG. The results are cleared when next measurement starts. Note: This register field is sticky. 00000000_H **MIN_VAL**: Min value $FFFFFFFF_H$ **MAX_VAL**: Max value

19.4.1.155 Upper 32 bits of Time-based Analysis Data

This viewport register returns the data selected by the TIME_BASED_REPORT_SELECT field in TIME_BASED_ANALYSIS_CONTROL_REG.

TIME_BASED_ANALYSIS_DATA_63_32_REG Offset address: 0004198_H

Upper 32 bits of Time-based Analysis Data Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TIME_BASED_ANALYSIS_DATA_63_32															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TIME_BASED_ANALYSIS_DATA_63_32															
rh															

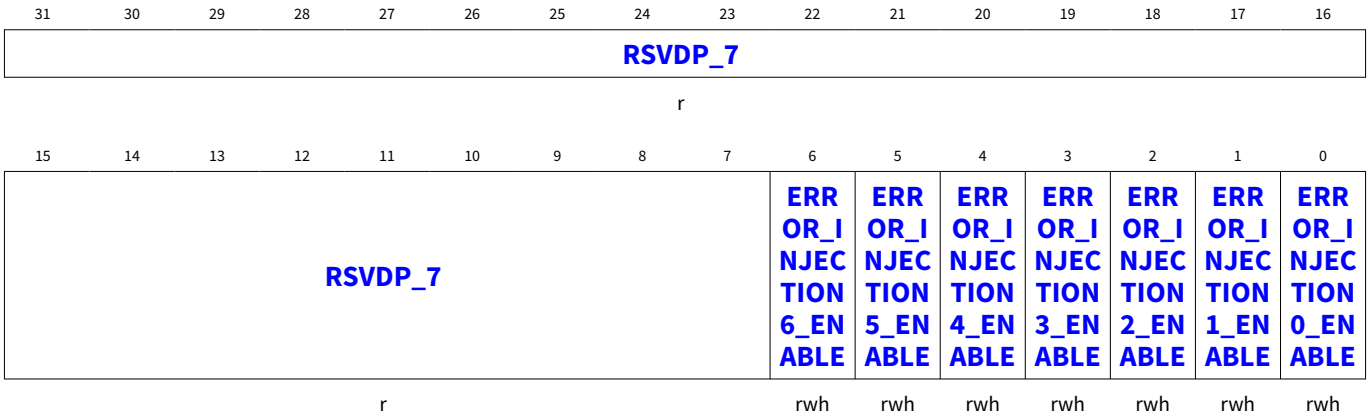
Field	Bits	Type	Description
TIME_BASED_ANALYSIS_DATA_63_32	31:0	rh	**Upper 32 bits of Time Based Analysis Data** Note: This register field is sticky. 00000000_H **MIN_VAL**: Min value $FFFFFFFF_H$ **MAX_VAL**: Max value

19.4.1.156 Error Injection Enable

Each type of error insertion is enabled by the corresponding bit in this register. The specific injection controls for each type of error are defined in the following registers: - 0: CRC Error: EINJ0_CRC_REG - 1: Sequence Number Error: EINJ1_SEQNUM_REG - 2: DLLP Error: EINJ2_DLLP_REG - 3: Symbol DataK Mask Error or Sync Header Error: EINJ3_SYMBOL_REG - 4: FC Credit Update Error: EINJ4_FC_REG - 5: TLP Duplicate/Nullify Error: EINJ5_SP_TLP_REG - 6: Specific TLP Error: EINJ6_COMPARE_*_REG/EINJ6_CHANGE_*_REG/EINJ6_TLP_REG After the errors have been inserted by controller, it will clear each bit here.

EINJ_ENABLE_REG Offset address: $00041B0_H$

Error Injection Enable Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
ERROR_INJECTION0_ENABLE	0	rwh	**Error Injection0 Enable (CRC Error)** Enables insertion of errors into various CRC. For more details, see the EINJ0_CRC_REG register. Note: This register field is sticky. 0_B **DISABLE**: Disable 1_B **ENABLE**: Enable
ERROR_INJECTION1_ENABLE	1	rwh	**Error Injection1 Enable (Sequence Number Error)** Enables insertion of errors into sequence numbers. For more details, see the EINJ1_SEQNUM_REG register. Note: This register field is sticky. 0_B **DISABLE**: Disable 1_B **ENABLE**: Enable
ERROR_INJECTION2_ENABLE	2	rwh	**Error Injection2 Enable (DLLP Error)** Enables insertion of DLLP errors. For more details, see the EINJ2_DLLP_REG register. Note: This register field is sticky. 0_B **DISABLE**: Disable 1_B **ENABLE**: Enable
ERROR_INJECTION3_ENABLE	3	rwh	**Error Injection3 Enable (Symbol DataK Mask Error or Sync Header Error)** Enables DataK masking of special symbols or the breaking of the sync header. For more details, see the EINJ3_SYMBOL_REG register. Note: This register field is sticky. 0_B **DISABLE**: Disable 1_B **ENABLE**: Enable
ERROR_INJECTION4_ENABLE	4	rwh	**Error Injection4 Enable (FC Credit Update Error)** Enables insertion of errors into UpdateFCs. For more details, see the EINJ4_FC_REG register. Note: This register field is sticky. 0_B **DISABLE**: Disable 1_B **ENABLE**: Enable
ERROR_INJECTION5_ENABLE	5	rwh	**Error Injection5 Enable (TLP Duplicate/Nullify Error)** Enables insertion of duplicate/nullified TLPs. For more details, see the EINJ5_SP_TLP_REG register. Note: This register field is sticky. 0_B **DISABLE**: Disable 1_B **ENABLE**: Enable

(table continues...)

(continued)

Field	Bits	Type	Description
ERROR_INJECTION6_ENABLE	6	rwh	**Error Injection6 Enable (Specific TLP Error)** Enables insertion of errors into the packets that you select. For more details, see the EINJ6_COMPARE_*_REG/EINJ6_CHANGE_*_REG/ EINJ6_TLP_REG registers. Note: This register field is sticky. 0_B **DISABLE**: Disable 1_B **ENABLE**: Enable
RSVDP_7	31:7	r	**Reserved for future use**

19.4.1.157 Error Injection Control 0 (CRC Error)

Controls the insertion of errors into the CRC, and parity of ordered sets for the selected type of the packets as follows: - LCRC. Bad TLP will be detected at the receiver side; receiver responds with NAK DLLP; Data Link Retry starts. - 16-bit CRC of ACK/NAK DLLPs. Bad DLLP occurs at the receiver side; Replay NUM Rollover occurs. - 16-bit CRC of UpdateFC DLLPs. Error insertion continues for the specific time; LTSSM transitions to the Recovery state because of the UpdateFC timeout (if the timeout is implemented at the receiver of the UpdateFCs). - ECRC. If ECRC check is enabled, ECRC error is detected at the receiver side. - FCRC. Framing error will be detected, TLP is discarded, and the LTSSM transitions to Recovery state. - Parity of TSOS. Error insertion continues for the specific time; LTSSM Recovery/Configuration timeout will occur. - Parity of SKPOS. Lane error will be detected at the receiver side.

EINJ0_CRC_REG Offset address: $00041B4_H$

Error Injection Control 0 [CRC Error) Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_12															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVDP_12				EINJ0_CRC_TYPE				EINJ0_COUNT							
r				rw				rwh							

Field	Bits	Type	Description
EINJ0_COUNT	7:0	rwh	**Error injection count** Indicates the number of errors. This register is decremented when the errors have been inserted. - If the counter value is 0x01 and error is inserted, ERROR_INJECTION0_ENABLE in EINJ_ENABLE_REG returns 0b. - If the counter value is 0x00 and ERROR_INJECTION0_ENABLE=1, the errors are inserted until ERROR_INJECTION0_ENABLE is set to '0'. Note: This register field is sticky.

(table continues...)

(continued)

Field	Bits	Type	Description
EINJ0_CRC_TYPE	11:8	rw	**Error injection type** Selects the type of CRC error to be inserted. All encodings other than the defined encodings are reserved. Note: This register field is sticky. 0_H **TLP_LCRC_ERR**: (TX Path) New TLP's LCRC error injection 1_H **_16b_CRC_ERR_ACK_NAK_DLLP**: (TX Path) 16bCRC error injection of ACK/NAK DLLP 2_H **_16b_CRC_ERR_UPD_FC**: (TX Path) 16bCRC error injection of Update-FC DLLP 3_H **TLP_ECRC_ERR**: (TX Path) New TLP's ECRC error injection 4_H **FCRC_ERR_TLP**: (TX Path) TLP's FCRC error injection (128b/130b) 5_H **PARITY_TSOS_ERR**: (TX Path) Parity error of TSOS (128b/130b) 6_H **PARITY_SKPOS_ERR**: (TX Path) Parity error of SKPOS (128b/130b) 8_H **LCRC_ERR**: (RX Path) LCRC error injection B_H **ECRC_ERR**: (RX Path) ECRC error injection **others,** Reserved
RSVDP_12	31:12	r	**Reserved for future use**

19.4.1.158 Error Injection Control 1 (Sequence Number Error).

Controls the sequence number of the specific TLPs and ACK/NAK DLLPs. Data Link Protocol Error will be detected at the Rx side of ACK/NAL DLLPs when one of these conditions is true: - ((NEXT_TRANSMIT_SEQ -1) - AckNak_Seq_Num) mod 4096 > 2048 - (AckNak_Seq_Num - ACKD_SEQ) mod 4096 >= 2048 TLP is treated as Duplicate TLP at the Rx side when all these conditions are true: - Sequence Number != NEXT_RCV_SEQ - (NEXT_RCV_SEQ - Sequence Number) mod 4096 <= 2048 TLP is treated as Bad TLP at the Rx side when all these conditions are true: - Sequence Number != NEXT_RCV_SEQ and - (NEXT_RCV_SEQ - Sequence Number) mod 4096 > 2048

EINJ1_SEQNUM_REG Offset address: $00041B8_H$

Error Injection Control 1 [Sequence Number Error).
 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_29			EINJ1_BAD_SEQNUM												
r			rw												

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVDP_9							EINJ1_SEQNUM_TYPE	EINJ1_COUNT							
r							rw	rwh							

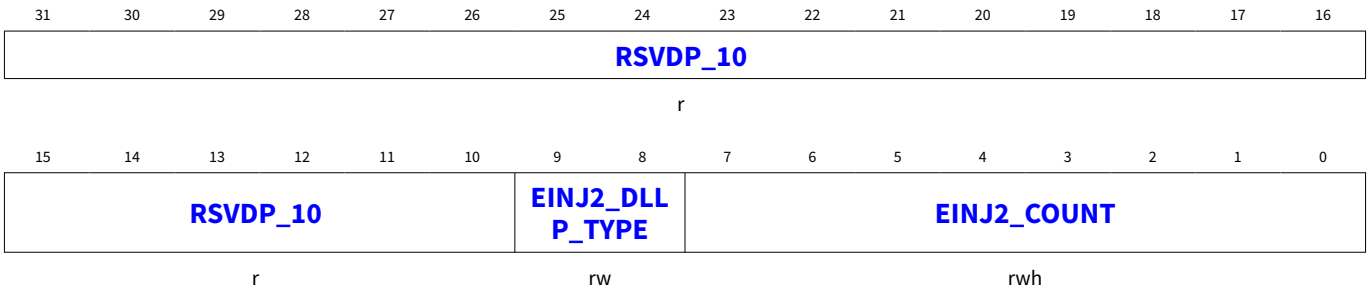
Field	Bits	Type	Description
EINJ1_COUNT	7:0	rwh	**Error injection count** Indicates the number of errors. This register is decremented as the errors are being inserted. - If the counter value is 0x01 and error is inserted, ERROR_INJECTION1_ENABLE in EINJ_ENABLE_REG returns '0'. - If the counter value is 0x00 and ERROR_INJECTION1_ENABLE=1, the errors are inserted until ERROR_INJECTION1_ENABLE is set to '0'. Note: This register field is sticky.
EINJ1_SEQNUM_TYPE	8	rw	**Sequence number type** Selects the type of sequence number. Note: This register field is sticky. 0_B **TLP_ERR**: Insertion of New TLP's SEQ# error 1_B **ACK_NAK_DLLP_ERR**: Insertion of ACK/NAK DLLP's SEQ# Error
RSVDP_9	15:9	r	**Reserved for future use**
EINJ1_BAD_SEQNUM	28:16	rw	**Bad sequence number** Indicates the value to add/subtract from the naturally-assigned sequence numbers. This value is represented by two's complement. For example: - Set Type, SEQ# and Count -- EINJ1_SEQNUM_TYPE =0 (Insert errors into new TLPs) -- EINJ1_BAD_SEQNUM =0x1FFD (represents -3) -- EINJ1_COUNT =1 - Enable Error Injection -- ERROR_INJECTION1_ENABLE =1 - Send a TLP From the Core's Application Interface -- Assume SEQ#5 is given to the TLP. - The SEQ# is Changed to #2 by the Error Injection Function in Layer2. -- 5 + (-3) = 2 - The TLP with SEQ#2 is Transmitted to PCIe Link. Note: This register field is sticky. $0FFF_H$ **MAX_VAL**: +4095 1001_H **MIN_VAL**: -4095 **others,** Reserved
RSVDP_29	31:29	r	**Reserved for future use**

19.4.1.159 Error Injection Control 2 (DLLP Error)

Controls the transmission of DLLPs and inserts the following errors: - If "ACK/NAK DLLP's transmission block" is selected, replay timeout error will occur at the transmitter of the TLPs and then Data Link Retry will occur. - If "Update FC DLLP's transmission block" is selected, LTSSM will transition to the Recovery state because of the UpdateFC timeout (if the timeout is implemented at the receiver of the UpdateFCs). - If "Always Transmission for NAK DLLP" is selected, Data Link Retry will occur at the transmitter of the TLPs. Furthermore, Replay NUM Rollover will occur when the transmitter has been requested four times to send the TLP with the same sequence number.

EINJ2_DLLP_REG Offset address: $00041BC_H$

Error Injection Control 2 [DLLP Error) Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
EINJ2_COUNT	7:0	rwh	**Error injection count** Indicates the number of errors. This register is decremented as the errors are being inserted. - If the counter value is 0x01 and the error is inserted, ERROR_INJECTION2_ENABLE in EINJ_ENABLE_REG returns '0'. - If the counter value is 0x00 and ERROR_INJECTION2_ENABLE =1, the errors are inserted until ERROR_INJECTION2_ENABLE is set to '0'. This register is affected only when EINJ2_DLLP_TYPE =2'10b. Note: This register field is sticky.
EINJ2_DLLP_TYPE	9:8	rw	**DLLP Type** Selects the type of DLLP errors to be inserted. Note: This register field is sticky. 00_B **ACK_NACK_DLLP**: ACK/NAK DLLP's transmission block 01_B **UPD_FC_DLLP**: Update FC DLLP's transmission block 10_B **NAK_DLLP**: Always Transmission for NAK DLLP 11_B **RSVD**: Reserved
RSVDP_10	31:10	r	**Reserved for future use**

19.4.1.160 Error Injection Control 3 (Symbol Error)

When 8b/10b encoding is used, this register controls error insertion into the special (K code) symbols. - If TS1/TS2/FTS/E-Idle/SKP is selected, it affects whole of the ordered set. It might cause timeout of the LTSSM. - If END/EDB/STP/SDP is selected, TLP/DLLP will be discarded at the receiver side. When 128b/130b encoding is used, this register controls error insertion into the sync-header.

EINJ3_SYMBOL_REG Offset address: $00041C0_H$

Error Injection Control 3 [Symbol Error) Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_11															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVDP_11					EINJ3_SYMBOL_TYPE			EINJ3_COUNT							
r					rw			rwh							

Field	Bits	Type	Description
EINJ3_COUNT	7:0	rwh	**Error injection count** Indicates the number of errors. This register is decremented as the errors are being inserted. - If the counter value is 0x01 and error is inserted, ERROR_INJECTION3_ENABLE in EINJ_ENABLE_REG returns '0'. - If the counter value is 0x00 and ERROR_INJECTION3_ENABLE =1, the errors are inserted until ERROR_INJECTION3_ENABLE is set to '0'. Note: This register field is sticky.

(table continues...)

(continued)

Field	Bits	Type	Description
EINJ3_SYMBOL_TYPE	10:8	rw	**Error Type** 8b/10b encoding - Mask K symbol. All encodings other than the defined encodings are reserved. Note: This register field is sticky. 000_B **RSVD_OR_INVRT_SYNC_HDR**: Invert sync header for 128b/130b encoding or this field is reserved for 8b/10b encoding 001_B **COM_PAD_TS1**: COM/PAD(TS1 Order set) 010_B **COM_PAD_TS2**: COM/PAD(TS2 Order set) 011_B **COM_FTS**: COM/FTS(FTS Order set) 100_B **COM_IDL**: COM/IDL(E-Idle Order set) 101_B **END_EDB**: END/EDB Symbol 110_B **STP_SDP**: STP/SDP Symbol 111_B **COM_SKP**: COM/SKP(SKP Order set)
RSVDP_11	31:11	r	**Reserved for future use**

19.4.1.161 Error Injection Control 4 (FC Credit Error)

Controls error insertion into the credit value in the UpdateFCs. It is possible to insert errors for any of the following types: - Posted TLP Header credit - Non-Posted TLP Header credit - Completion TLP Header credit - Posted TLP Data credit - Non-Posted TLP Data credit - Completion TLP Data credit These errors are not correctable while error insertion is enabled. Receiver buffer overflow error might occur.

EINJ4_FC_REG Offset address: $00041C4_H$

Error Injection Control 4 [FC Credit Error) Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_29			EINJ4_BAD_UPDFC_VALUE												
r			rw												

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVDP_15	EINJ4_VC_NUMBER			RSVDP_11	EINJ4_UPDFC_TYPE			EINJ4_COUNT							
r	rw			r	rw			rwh							

Field	Bits	Type	Description
EINJ4_COUNT	7:0	rwh	**Error injection count** Indicates the number of errors. This register is decremented as the errors are being inserted. - If the counter value is 0x01 and error is inserted, ERROR_INJECTION4_ENABLE in EINJ_ENABLE_REG returns '0'. - If the counter value is 0x00 and ERROR_INJECTION4_ENABLE =1, the errors are inserted until ERROR_INJECTION4_ENABLE is set to '0'. Note: This register field is sticky.

(table continues...)

(continued)

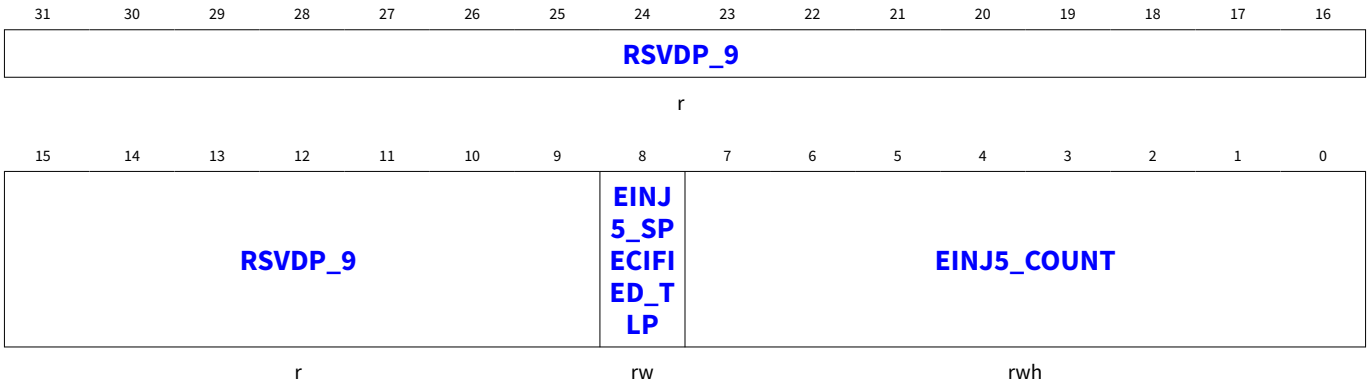
Field	Bits	Type	Description
EINJ4_UPDFC_TYPE	10:8	rw	**Update-FC type** Selects the credit type. Note: This register field is sticky. 000_B **POSTED_TLP_HDR**: Posted TLP Header Credit value control 001_B **NON_POSTED_TLP_HDR**: Non-Posted TLP Header Credit value control 010_B **CMPL_TLP_HDR**: Completion TLP Header Credit value control 011_B **RSERVED**: Reserved 100_B **POSTED_TLP_DATA**: Posted TLP Data Credit value control 101_B **NON_POSTED_TLP_DATA**: Non-Posted TLP Data Credit value control 110_B **CMPL_TLP_DATA**: Completion TLP Data Credit value control 111_B **RSVD**: Reserved
RSVDP_11	11	r	**Reserved for future use**
EINJ4_VC_NUMBER	14:12	rw	**VC Number** Indicates target VC Number. Note: This register field is sticky. 000_B **MIN_VAL**: Min Value 111_B **MAX_VAL**: Min Value
RSVDP_15	15	r	**Reserved for future use**
EINJ4_BAD_UPDFC_VALUE	28:16	rw	**Bad update-FC credit value** Indicates the value to add/subtract from the UpdateFC credit. This value is represented by two's complement. Note: This register field is sticky. $0FFF_H$ **MAX_VAL**: +4095 1001_H **MIN_VAL**: -4095 **others,** Reserved
RSVDP_29	31:29	r	**Reserved for future use**

19.4.1.162 Error Injection Control 5 (Specific TLP Error)

Controls the generation of specified TLPs. Correctable errors will occur which will be fixed by the PCIe protocol. - For Duplicate TLP, the controller initiates Data Link Retry by handling ACK DLLP as NAK DLLP. These TLPs will be duplicate TLPs at the receiver side. - For Nullified TLP, the TLPs that the controller transmits are changed into nullified TLPs and the original TLPs are stored in the retry buffer. The receiver of these TLPs will detect the lack of seq# and send NAK DLLP at the next TLP. Then the original TLPs are sent from retry buffer and the data controls are recovered. For 128 bit controller or more than 128 bit, the controller inserts errors the number of times of EINJ5_COUNT but doesn't ensure that the errors are continuously inserted into TLPs.

EINJ5_SP_TLP_REG Offset address: $00041C8_H$

Error Injection Control 5 [Specific TLP Error) Kernel Reset value: $0000\ 0000_H$



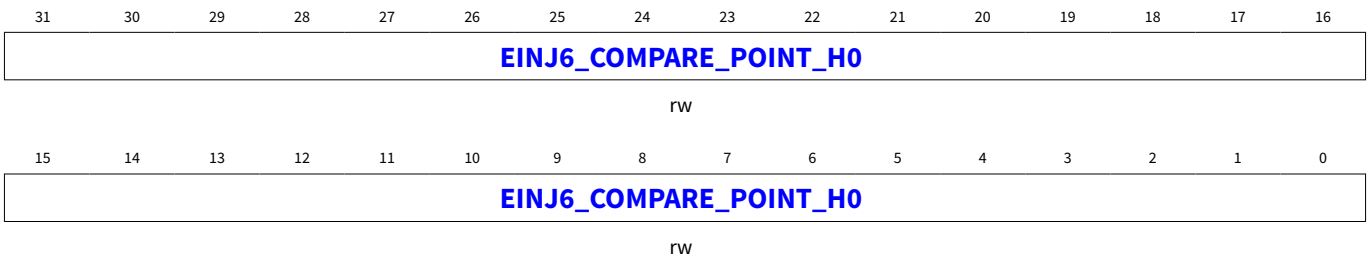
Field	Bits	Type	Description
EINJ5_COUNT	7:0	rwh	**Error injection count** Indicates the number of errors. This register is decremented as the errors are being inserted. - If the counter value is 0x01 and error is inserted, ERROR_INJECTION5_ENABLE in EINJ_ENABLE_REG returns '0'. - If the counter value is 0x00 and ERROR_INJECTION5_ENABLE =1, the errors are inserted until ERROR_INJECTION5_ENABLE is set to '0'. Note: This register field is sticky.
EINJ5_SPECIFIED_TLP	8	rw	**Specified TLP** Selects the specified TLP to be inserted. Note: This register field is sticky. 0_B **DUPLICATE_DLLP**: Generates duplicate TLPs by handling ACK DLLP as NAK DLLP 1_B **NULLIFIED_TLP**: Generates Nullified TLP (Original TLP will be stored in retry buffer)
RSVDP_9	31:9	r	**Reserved for future use**

19.4.1.163 Error Injection Control 6 (Compare Point Header DWORD #0) i

Program this register for the 1st DWORD of TLP header/prefix. It is necessary to carefully consider the endianness when you program this register. Bits [31:0] = TLP_DW0[7:0], TLP_DW0[15:8], TLP_DW0[23:16], TLP_DW0[31:24] The Packet Compare Point registers (EINJ6_COMPARE_POINT*) specify which Tx TLP header bits to compare with the corresponding bits in the Packet Compare Value registers (EINJ6_COMPARE_VALUE*). When all specified bits (in the Tx TLP header and EINJ6_COMPARE_VALUE*) match, the controller inserts errors into the TLP. The type and number of errors are specified by the EINJ6_TLP_REG register.

EINJ6_COMPARE_POINT_Hi_REG (i=0-3) Offset address: $00041CC_H+i*4$

Error Injection Control 6 [Compare Point Header DWORD #0) i — Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
EINJ6_COMPARE_POINT_H0	31:0	rw	**Packet Compare Point: 1st DWORD** Specifies which Tx TLP header DWORD#0 bits to compare with the corresponding bits in the Packet Compare Value registers (EINJ6_COMPARE_VALUE*). When all specified bits (in the Tx TLP header and EINJ6_COMPARE_VALUE*) match, the controller inserts errors into the TLP. Note: This register field is sticky.

19.4.1.164 Error Injection Control 6 (Compare Value Header DWORD #0) i

Program this register for the 1st DWORD of TLP header/prefix. It is necessary to carefully consider the endianness when you program this register. Bits [31:0] = TLP_DW0[7:0], TLP_DW0[15:8], TLP_DW0[23:16], TLP_DW0[31:24] The Packet Compare Point registers (EINJ6_COMPARE_POINT*) specify which Tx TLP header bits to compare with the corresponding bits in the Packet Compare Value registers (EINJ6_COMPARE_VALUE*). When all specified bits (in the Tx TLP header and EINJ6_COMPARE_VALUE*) match, the controller inserts errors into the TLP. The type and number of errors are specified by the EINJ6_TLP_REG register.

EINJ6_COMPARE_VALUE_Hi_REG (i=0-3) Offset address: $00041DC_H+i*4$

Error Injection Control 6 [Compare Value Header DWORD #0) i — Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EINJ6_COMPARE_VALUE_H0															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EINJ6_COMPARE_VALUE_H0															
rw															

Field	Bits	Type	Description
EINJ6_COMPARE_VALUE_H0	31:0	rw	**Packet Compare Value: 1st DWORD** Specifies the value to compare against Tx the TLP header DWORD#0 bits specified in the Packet Compare Point registers (EINJ6_COMPARE_POINT*). Note: This register field is sticky.

19.4.1.165 Error Injection Control 6 (Change Point Header DWORD #0) i

Program this register for the 1st DWORD of TLP header/prefix. It is necessary to carefully consider the endianness when you program this register. Bits [31:0] = TLP_DW0[7:0], TLP_DW0[15:8], TLP_DW0[23:16], TLP_DW0[31:24] The Packet Change Point registers (EINJ6_CHANGE_POINT*) specify which Tx TLP header bits to replace with the corresponding bits in the Packet Change Value registers (EINJ6_CHANGE_VALUE*). The type and number of errors are specified by the EINJ6_TLP_REG register. Only applies when EINJ6_INVERTED_CONTROL in EINJ6_TLP_REG =0.

EINJ6_CHANGE_POINT_Hi_REG (i=0-3) Offset address: $00041EC_H+i*4$

Error Injection Control 6 [Change Point Header DWORD #0) i — Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EINJ6_CHANGE_POINT_H0															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EINJ6_CHANGE_POINT_H0															
rw															

Field	Bits	Type	Description
EINJ6_CHANGE_POINT_H0	31:0	rw	**Packet Change Point: 1st DWORD** Specifies which Tx TLP header DWORD#0 bits to replace with the corresponding bits in the Packet Change Value registers (EINJ6_CHANGE_VALUE*). Note: This register field is sticky.

19.4.1.166 Error Injection Control 6 (Change Value Header DWORD #0) i

Program this register for the 1st DWORD of TLP header/prefix. It is necessary to carefully consider the endianness when you program this register. Bits [31:0] = TLP_DW0[7:0], TLP_DW0[15:8], TLP_DW0[23:16], TLP_DW0[31:24] The Packet Change Point registers (EINJ6_CHANGE_POINT*) specify which Tx TLP header bits to replace with the corresponding bits in the Packet Change Value registers (EINJ6_CHANGE_VALUE*). The type and number of errors are specified by the EINJ6_TLP_REG register. Only applies when EINJ6_INVERTED_CONTROL in EINJ6_TLP_REG =0.

EINJ6_CHANGE_VALUE_Hi_REG (i=0-3)

Error Injection Control 6 [Change Value Header DWORD #0) i

Offset address: $00041FC_H+i*4$

Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EINJ6_CHANGE_VALUE_H0															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EINJ6_CHANGE_VALUE_H0															
rw															

Field	Bits	Type	Description
EINJ6_CHANGE_VALUE_H0	31:0	rw	**Packet Change Value: 1st DWORD** Specifies replacement values for the Tx TLP header DWORD#0 bits defined in the Packet Change Point registers (EINJ6_CHANGE_POINT*). Only applies when the EINJ6_INVERTED_CONTROL field in EINJ6_TLP_REG is '0'. Note: This register field is sticky.

19.4.1.167 Error Injection Control 6 (Packet Error)

The Packet Compare Point registers (EINJ6_COMPARE_POINT*) specify which Tx TLP header bits to compare with the corresponding bits in the Packet Compare Value registers (EINJ6_COMPARE_VALUE*). When all specified bits (in the Tx TLP header and EINJ6_COMPARE_VALUE*) match, the controller inserts errors into the TLP. The type and number of errors are specified by the this register. The Packet Change Point registers (EINJ6_CHANGE_POINT*) specify which Tx TLP header bits to replace with the corresponding bits in the Packet Change Value registers (EINJ6_CHANGE_VALUE*). The type and number of errors are specified by the this register. Only applies when EINJ6_INVERTED_CONTROL in this register =0. The TLP into that errors are injected will not arrive at the transaction layer of the remote device when all of the following conditions are true. - Using 128b/130b encoding - Injecting errors into TLP Length field / TLP digest bit

EINJ6_TLP_REG Offset address: $000420C_H$

Error Injection Control 6 [Packet Error) Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_12															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVDP_12				EINJ6_PACKET_TYPE			EINJ6_INVERTED_CONTROL	EINJ6_COUNT							
r				rw			rw	rwh							

Field	Bits	Type	Description
EINJ6_COUNT	7:0	rwh	**Error Injection Count** Indicates the number of errors to insert. This counter is decremented while errors are been inserted. • If the counter value is 0x01 and error is inserted, ERROR_INJECTION6_ENABLE in EINJ_ENABLE_REG returns '0' • If the counter value is 0x00 and ERROR_INJECTION6_ENABLE=1, errors are inserted until ERROR_INJECTION6_ENABLE is set to '0' Note: This register field is sticky.
EINJ6_INVERTED_CONTROL	8	rw	**Inverted Error Injection Control** Encodded vlues given as above. Note: This register field is sticky. 0_B **REPLACE**: EINJ6_CHANGE_VALUE_H[0/1/2/3] is used to replace bits specified by EINJ6_CHANGE_POINT_H[0/1/2/3] 1_B **IGNORE**: EINJ6_CHANGE_VALUE_H[0/1/2/3] is ignored and inverts bits specified by EINJ6_CHANGE_POINT_H[0/1/2/3]
EINJ6_PACKET_TYPE	11:9	rw	**Packet type** Selects the TLP packets to inject errors into. All encodings other than the specified encodings are reserved. Note: This register field is sticky. 000_B **HEADER**: TLP Header 001_B **TLP_FIRST_4_DW**: TLP Prefix 1st 4-DWORDs 010_B **TLP_SECOND_DW**: TLP Prefix 2nd -DWORDs **others,** Reserved

(table continues...)

(continued)

Field	Bits	Type	Description
RSVDP_12	31:12	r	**Reserved for future use**

19.4.1.168 Silicon Debug Control 1

SD_CONTROL1_REG Offset address: 0004220_H

Silicon Debug Control 1 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_24								LOW_POWER_INTERVAL		TX_EIOS_NUM		RSVDP_17			FORCE_DETECT_LANE_EN
r								rw		rw		r			rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FORCE_DETECT_LANE															
rw															

Field	Bits	Type	Description
FORCE_DETECT_LANE	15:0	rw	**Force Detect Lane** When the FORCE_DETECT_LANE_EN field is set, the controller ignores receiver detection from PHY during LTSSM Detect state and uses this value instead. Value represents lane number. Note: This register field is sticky. 0000_H **MIN_VAL**: Lane0 $000F_H$ **MAX_VAL**: Lane15 **others,** Reserved
FORCE_DETECT_LANE_EN	16	rw	**Force Detect Lane Enable** Note: This register field is sticky. 0_B **CLEAR**: Clear 1_B **SET**: When this bit is set, the controller ignores receiver detection from PHY during LTSSM Detect state and uses FORCE_DETECT_LANE
RSVDP_17	19:17	r	**Reserved for future use**

(table continues...)

(continued)

Field	Bits	Type	Description
TX_EIOS_NUM	21:20	rw	**Number of Tx EIOS** This register sets the number of transmit EIOS for L0s/L1 entry and Disable/Loopback/Hot-reset exit. The controller selects the greater value between this register and the value defined by the PCI-SIG specification. Note: This register field is sticky. 00_B **G2_2_EIOS_OTHER_SPEED_1_EIOS**: (2.5GT/s, 8.0GT/s or higher) 1 or (5.0GT/s) 2 01_B **G2_8_EIOS_OTHER_SPEED_4_EIOS**: (2.5GT/s, 8.0GT/s or higher) 4 or (5.0GT/s) 8 10_B **G2_16_EIOS_OTHER_SPEED_8_EIOS**: (2.5GT/s, 8.0GT/s or higher) 8 or (5.0GT/s) 16 11_B **G2_32_EIOS_OTHER_SPEED_16_EIOS**: (2.5GT/s, 8.0GT/s or higher) 16 or (5.0GT/s) 32
LOW_POWER_INTERVAL	23:22	rw	**Low Power Entry Interval Time** Interval Time that the controller starts monitoring RXELECIDLE signal after L0s/L1/L2 entry. You should set the value according to the latency from receiving EIOS to, RXELECIDLE assertion at the PHY. Note: This register field is sticky. 00_B **_40NS**: 40ns 01_B **_160NS**: 160ns 10_B **_320NS**: 320ns 11_B **_640NS**: 640ns
RSVDP_24	31:24	r	**Reserved for future use**

19.4.1.169 Silicon Debug Control 2

SD_CONTROL2_REG Offset address: 0004224_H

Silicon Debug Control 2 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_17															FRAMING_ERR_RECOVERY_DISABLE
r															rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVDP_11					DIRECT_LPBKSLV_TO_EXIT	DIRECT_POLCOMP_TO_DETECT	DIRECT_RECIDLE_TO_CONFIG	RSVDP_3					NOACK_FORCE_LINKDOWN	RECOVERY_REQUEST	HOLD_LTSSM
r					rw	rw	rw	r					rw	rwh	rw

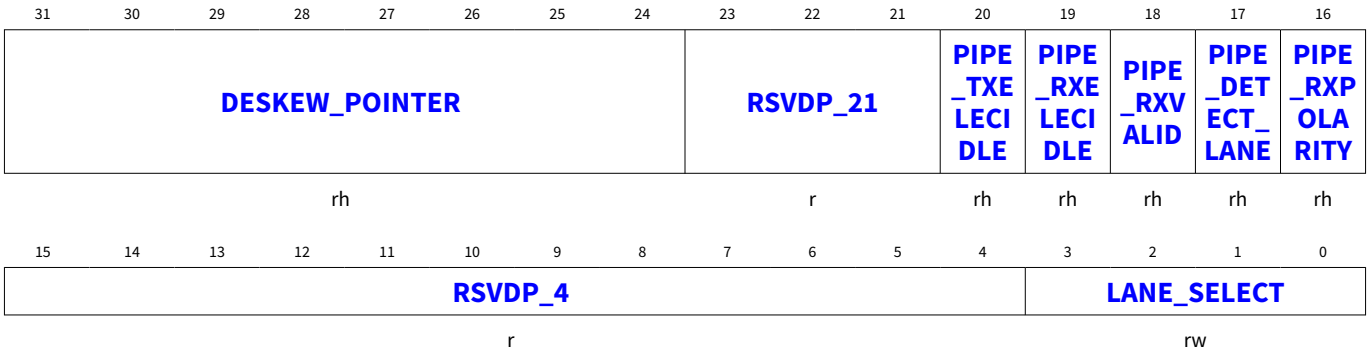
Field	Bits	Type	Description
HOLD_LTSSM	0	rw	**Hold and Release LTSSM** Note: This register field is sticky. 0_B **CLEAR**: Clear 1_B **SET**: As long as this field is '1', the controller stays in the current LTSSM
RECOVERY_REQUEST	1	rwh	**Recovery Request. Reading register field always returns a '0'.** 0_B **CLEAR**: Clear 1_B **SET**: When this bit is set to '1' in L0 or L0s, the LTSSM starts transitioning to Recovery State This request does not cause a speed change or re-equalization.
NOACK_FORCE_LINKDOWN	2	rw	**Force LinkDown** Note: This register field is sticky. 0_B **CLEAR**: Clear 1_B **SET**: When this bit is set and the controller detects REPLY_NUM rolling over 4 times, the LTSSM transitions to Detect State
RSVDP_3	7:3	r	**Reserved for future use**
DIRECT_RECIDLE_TO_CONFIG	8	rw	**Direct Recovery.Idle to Configuration** Note: This register field is sticky. 0_B **CLEAR**: Clear 1_B **SET**: When this bit is set and the LTSSM is in Recovery Idle State, the LTSSM transitions to Configuration state
DIRECT_POLCOMP_TO_DETECT	9	rw	**Direct Polling.Compliance to Detect** Note: This register field is sticky. 0_B **CLEAR**: Clear 1_B **SET**: When this bit is set and the LTSSM is in Polling Compliance State, the LTSSM transitions to Detect state
DIRECT_LPBKSLV_TO_EXIT	10	rw	**Direct Loopback Slave To Exit** Note: This register field is sticky. 0_B **CLEAR**: Clear 1_B **SET**: When this bit is set and the LTSSM is in Loopback Slave Active State, the LTSSM transitions to Loopback Slave Exit state
RSVDP_11	15:11	r	**Reserved for future use**
FRAMING_ERR_RECOVERY_DISABLE	16	rw	**Framing Error Recovery Disable** This bit disables a transition to Recovery state when a Framing Error is occurred. Note: This register field is sticky.
RSVDP_17	31:17	r	**Reserved for future use**

19.4.1.170 Silicon Debug Status(Layer1 Per-lane)

This viewport register returns the data selected by the following field: - LANE_SELECT in SD_STATUS_L1LANE_REG

SD_STATUS_L1LANE_REG Offset address: 0004230_H

Silicon Debug Status[Layer1 Per-lane) Kernel Reset value: $0018\ 0000_H$



Field	Bits	Type	Description
LANE_SELECT	3:0	rw	**Lane Select** Lane Select register for Silicon Debug Status Register of Layer1-PerLane. Note: This register field is sticky. 0_H **MIN_VAL**: Lane0 F_H **MAX_VAL**: Lane15
RSVDP_4	15:4	r	**Reserved for future use**
PIPE_RXPOLARITY	16	rh	**PIPE:RxPolarity** Indicates PIPE RXPOLARITY signal of selected lane number(LANE_SELECT). Note: This register field is sticky.
PIPE_DETECT_LANE	17	rh	**PIPE:Detect Lane** Indicates whether PHY indicates receiver detection or not on selected lane number(LANE_SELECT). Note: This register field is sticky.
PIPE_RXVALID	18	rh	**PIPE:RxValid** Indicates PIPE RXVALID signal of selected lane number(LANE_SELECT). Note: This register field is sticky.
PIPE_RXELECIDLE	19	rh	**PIPE:RxElecIdle** Indicates PIPE RXELECIDLE signal of selected lane number(LANE_SELECT). Note: This register field is sticky.
PIPE_TXELECIDLE	20	rh	**PIPE:TxElecIdle** Indicates PIPE TXELECIDLE signal of selected lane number(LANE_SELECT). Note: This register field is sticky.
RSVDP_21	23:21	r	**Reserved for future use**
DESKEW_POINTER	31:24	rh	**Deskew Pointer** Indicates Deskew pointer of internal Deskew buffer of selected lane number(LANE_SELECT). Note: This register field is sticky.

19.4.1.171 Silicon Debug Status(Layer1 LTSSM)

SD_STATUS_L1LTSSM_REG Offset address: 0004234_H

Silicon Debug Status[Layer1 LTSSM) Kernel Reset value: $0000\ 0200_H$

19 Peripheral Component Interconnect Express (PCIe)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LTSSM_VARIABLE															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LANE_REVERSAL	RSVDP_11				PIPE_POWER_DOWN			FRAMING_ERR	FRAMING_ERR_PTR						
rh	r				rh			rw1ch	rh						

Field	Bits	Type	Description
FRAMING_ERR_PTR	6:0	rh	**First Framing Error Pointer** Identifies the first Framing Error using the following encoding. The field contents are only valid value when FRAMING_ERR =1. • Received Unexpected Framing Token (Values 01h to 06h) - Received Unexpected STP Token (Values 11h to 13h) • Received Unexpected Block (Values 21h to 2Eh) All encodings other than the defined encodings are reserved. Note: This register field is sticky. 01_H **STP_OR_SDP_OR_IDL_RECEIVED_NO_TLP_OR_DLLP**: When non- STP/SDP/IDL Token was received and it was not in TLP/DLLP reception 02_H **CURRENT_NO_VALID_EDB_PREVIOUS_VALID_EDB**: When current token was not a valid EDB token and previous token was an EDB (128/256 bit controller only) 03_H **SDP_RECEIVED_NOT_EXPECTED**: When SDP token was received but not expected (128 bit & (x8 | x16) controller only) 04_H **STP_RECEIVED_NOT_EXPECTED**: When STP token was received but not expected (128 bit & (x8 | x16) controller only) 05_H **EDS_RECEIVED_NOT_EXPECTED**: When EDS token was expected but not received or whenever an EDS token was received but not expected 06_H **FRE_ERR_DESKEW_PKT_IN_PROGRESS**: When a framing error was detected in the deskew block while a packet has been in progress in token_finder 11_H **CRC_STP_NOT_MATCH**: When Framing CRC in STP token did not match 12_H **PARITY_STP_NOT_MATCH**: When Framing Parity in STP token did not match 13_H **TLP_LENGTH_SMALLER_THEN_5DW**: When Framing TLP Length in STP token was smaller than 5 DWORDs 21_H **RECEVING_OS_AFTER_SDS_IN_DATA_STREAM**: When Receiving an OS Block following SDS in Datastream state 22_H **AFTER_DATA_BLK_OS_BLK_NOT_SKP_EI_EIE**: When Data Block followed by OS Block different from SKP, EI, EIE in Datastream state 23_H **UNDEFINE_BLK_TYPE**: When Block with an undefined Block Type in Datastream state 24_H **DATA_STREAM_WITHOUT_3_CYCLE_DATA_STREAM_S**: When Data Stream without data over three cycles in Datastream state 25_H **OS_BLK_DURING_DATA_STREAM**: When OS Block during Data Stream in Datastream state 26_H **RXSTATUS_ERR_DATA_STREAM_STATE**: When RxStatus Error was detected in Datastream state 27_H **NOT_ALL_LANE_START_RECEVING_SKP_OS_SAME_T**: When Not all active lanes receiving SKP OS starting at same cycle time in SKPOS state

(table continues...)

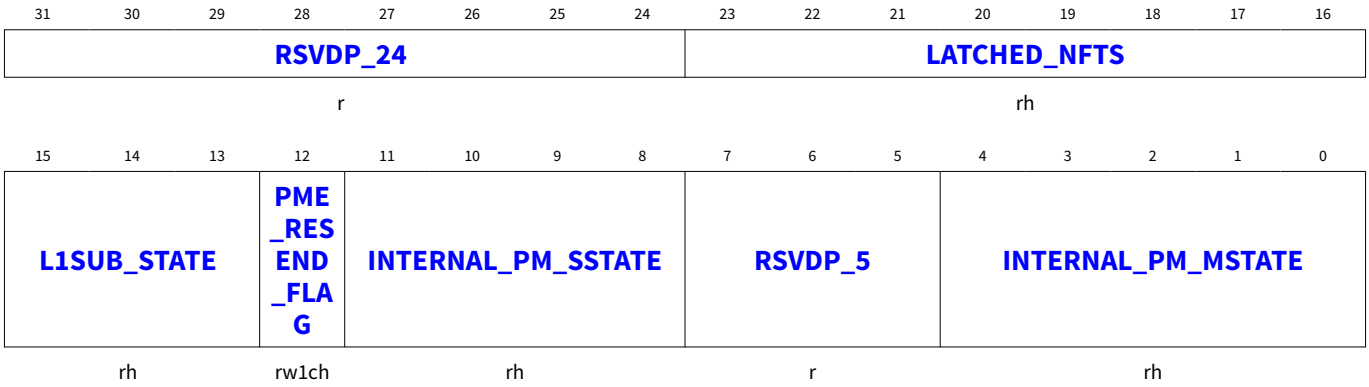
(continued)

Field	Bits	Type	Description
			28_H **_2_BLK_TIMEOUT_SKP_OS_SKPOS_STATE**: When a 2-Block timeout occurs for SKP OS in SKPOS state 29_H **CONSECUTIVE_OS_WITHIN_DATA_IN_SKPOS_STATE**: When Receiving consecutive OS Blocks within a Data Stream in SKPOS state $2A_H$ **PHYSTATUS_ERR_IN_SKPOS_STATE**: When Phy status error was detected in SKPOS state $2B_H$ **NOT_ALL_LANE_START_RECEVING_EIOS_SAME_T**: When Not all active lanes receiving EIOS starting at same cycle time in EIOS state $2C_H$ **AT_LEAST_1_SYM_IS_NOT_EIOS_FROM_4_SYM**: When At least one Symbol from the first 4 Symbols is not EIOS Symbol in EIOS state (CX_NB=2 only) $2D_H$ **NOT_ALL_LANE_START_RECEVING_EIEOS_SAME_T**: When Not all active lanes receiving EIEOS starting at same cycle time in EIEOS state $2E_H$ **NOT_16_EIEOS_SYM_RECEIVED**: When Not full 16 eieos symbols are received in EIEOS state **others,** Reserved
FRAMING_ERR	7	rw1ch	**Framing Error** Indicates Framing Error detection status. Note: This register field is sticky. Software write with 1 clears this bit-field.
PIPE_POWER_DOWN	10:8	rh	**PIPE:PowerDown** Indicates PIPE PowerDown signal. Note: This register field is sticky.
RSVDP_11	14:11	r	**Reserved for future use**
LANE_REVERSAL	15	rh	**Lane Reversal Operation** Receiver detected lane reversal. This field is only valid in the L0 LTSSM state. Note: This register field is sticky.
LTSSM_VARIABLE	31:16	rh	**LTSSM Variable** Indicates internal LTSSM variables defined in the PCI Express Base Specification. . Note: This register field is sticky. 0000_H **DIR_SPEED_CHANGE**: directed_speed_change 0001_H **CHANGED_SPEED_RCVRY**: changed_speed_recovery 0002_H **SUCCESSFUL_SPEED_NEGO**: successful_speed_negotiation 0003_H **UPCFG_CAPABLE**: upconfigure_capable; Set to '1' if both ports advertised the UpConfigure capability in the last Config.Complete 0004_H **SEL_DE_EMPHASIS**: select_deemphasis 0005_H **START_EQ_W_PRESET**: start_equalization_w_preset 0006_H **EQ_DONE_8GT**: equalization_done_8GT_data_rate 0007_H **EQ_DONE_16GT**: equalization_done_16GT_data_rate **others,** Reserved

19.4.1.172 Silicon Debug Status(PM)

SD_STATUS_PM_REG Offset address: 0004238_H

Silicon Debug Status[PM) Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
INTERNAL_PM_MSTATE	4:0	rh	**Internal PM State(Master)** Indicates internal state machine of Power Management Master controller. Note: This register field is sticky. 00_H **IDLE**: IDLE 01_H **L0**: L0 02_H **L0S**: L0S 03_H **ENTER_L0S**: ENTER_L0S 04_H **EXIT_L0S**: L0S_EXIT 05_H **WAIT_PMCSR_CPL_SENT**: WAIT_PMCSR_CPL_SENT 08_H **L1**: L1 09_H **L1_BLOCK_TLP**: L1_BLOCK_TLP $0A_H$ **L1_WAIT_LAST_TLP_ACK**: L1_WAIT_LAST_TLP_ACK $0B_H$ **L1_WAIT_PMDLLP_ACK**: L1_WAIT_PMDLLP_ACK $0C_H$ **L1_LINK_ENTR_L1**: L1_LINK_ENTR_L1 $0D_H$ **L1_EXIT**: L1_EXIT $0F_H$ **PREP_4L1**: PREP_4L1 10_H **L23_BLOCK_TLP**: L23_BLOCK_TLP 11_H **L23_WAIT_LAST_TLP_ACK**: L23_WAIT_LAST_TLP_ACK 12_H **L23_WAIT_PMDLLP_ACK**: L23_WAIT_PMDLLP_ACK 13_H **L23_ENTR_L23**: L23_ENTR_L23 14_H **L23RDY**: L23RDY 15_H **PREP_4L23**: PREP_4L23 16_H **L23RDY_WAIT4ALIVE**: L23RDY_WAIT4ALIVE 17_H **L0S_BLOCK_TLP**: L0S_BLOCK_TLP 18_H **WAIT_LAST_PMDLLP**: WAIT_LAST_PMDLLP 19_H **WAIT_DSTATE_UPDATE**: WAIT_DSTATE_UPDATE **others,** Reserved
RSVDP_5	7:5	r	**Reserved for future use**

(table continues...)

(continued)

Field	Bits	Type	Description
INTERNAL_PM_SSTATE	11:8	rh	**Internal PM State(Slave)** Indicates internal state machine of Power Management Slave controller. Note: This register field is sticky. 0_H **S_IDLE**: S_IDLE 1_H **S_RESPOND_NAK**: S_RESPOND_NAK 2_H **S_BLOCK_TLP**: S_BLOCK_TLP 3_H **S_WAIT_LAST_TLP_ACK**: S_WAIT_LAST_TLP_ACK 4_H **S_WAIT_EIDLE**: S_WAIT_EIDLE 5_H **S_LINK_ENTR_L1**: S_LINK_ENTR_L1 6_H **S_L1**: S_L1 7_H **S_L1_EXIT**: S_L1_EXIT 8_H **S_L23RDY**: S_L23RDY 9_H **S_LINK_ENTR_L23**: S_LINK_ENTR_L23 A_H **S_L23RDY_WAIT4ALIVE**: S_L23RDY_WAIT4ALIVE B_H **S_ACK_WAIT4IDLE**: S_ACK_WAIT4IDLE C_H **S_WAIT_LAST_PMDLLP**: S_WAIT_LAST_PMDLLP **others,** Reserved
PME_RESEND_FLAG	12	rw1ch	**PME Re-send flag** When the DUT sends a PM_PME message TLP, the DUT sets PME_Status bit. If host software does not clear PME_Status bit for 100ms(+50%/-5%), the DUT resends the PM_PME Message. This bit indicates that a PM_PME was resent. Note: This register field is sticky. Software write with 1 clears this bit-field.
L1SUB_STATE	15:13	rh	**L1Sub State** Indicates internal state machine of L1Sub state. Note: This register field is sticky. 000_B **S_L1_U**: S_L1_U : idle state 001_B **S_L1_0**: S_L1_0 : wait for aux_clk_active 010_B **S_L1_0_WAIT4_ACK**: S_L1_0_WAIT4_ACK : wait for pclkack 011_B **S_L1_0_WAIT4_CLKREQ**: S_L1_0_WAIT4_CLKREQ : wait for clkreq 100_B **S_L1_N_ENTRY**: S_L1_N_ENTRY : check clkreq_in_n is de-asserted for t_power_off time (only for L1.2, reduces to one cycle for L1.1) 101_B **S_L1_N**: S_L1_N : L1 substate, turn off txcommonmode circuits (L1.2 only) and rx electrical idle detection circuits 110_B **S_L1_N_EXIT**: S_L1_N_EXIT : locally/remotely initiated exit, assert pclkreq, wait for pclkack 111_B **S_L1_N_ABORT**: S_L1_N_ABORT : wait for pclkack when aborting an attempt to enter L1_N
LATCHED_NFTS	23:16	rh	**Latched N_FTS** Indicates the value of N_FTS in the received TS Ordered Sets from the Link Partner. Note: This register field is sticky.
RSVDP_24	31:24	r	**Reserved for future use**

19.4.1.173 Silicon Debug Status(Layer2)

SD_STATUS_L2_REG Offset address: 000423C$_H$

Silicon Debug Status[Layer2) Kernel Reset value: 00FF F000$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_28				FC_INIT2	FC_INIT1	DLCMSM		RX_ACK_SEQ_NO							
r				rh	rh	rh		rh							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RX_ACK_SEQ_NO				TX_TLP_SEQ_NO											
rh				rh											

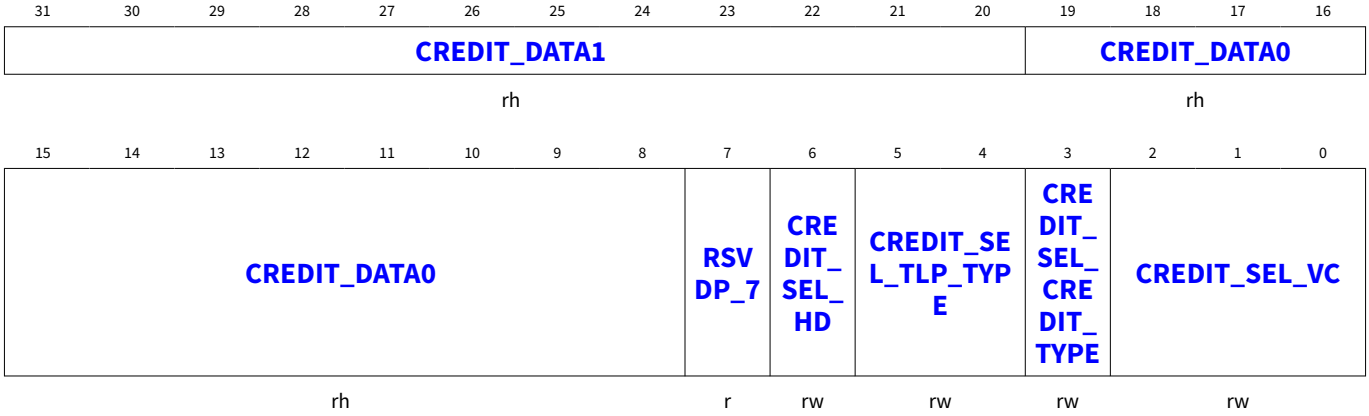
Field	Bits	Type	Description
TX_TLP_SEQ_NO	11:0	rh	**Tx Tlp Sequence Number** Indicates next transmit sequence number for transmit TLP. Note: This register field is sticky.
RX_ACK_SEQ_NO	23:12	rh	**Tx Ack Sequence Number** Indicates ACKD_SEQ which is updated by receiving ACK/NAK DLLP. Note: This register field is sticky.
DLCMSM	25:24	rh	**DLCMSM** Indicates the current DLCMSM. Note: This register field is sticky. 00_B **DL_INACTIVE**: DL_INACTIVE 01_B **DL_FC_INIT**: DL_FC_INIT 11_B **DL_ACTIVE**: DL_ACTIVE
FC_INIT1	26	rh	**FC_INIT1** Indicates the controller is in FC_INIT1(VC0) state. Note: This register field is sticky.
FC_INIT2	27	rh	**FC_INIT2** Indicates the controller is in FC_INIT2(VC0) state. Note: This register field is sticky.
RSVDP_28	31:28	r	**Reserved for future use**

19.4.1.174 Silicon Debug Status(Layer3 FC)

The CREDIT_DATA[0/1] fields in this viewport register return the data for the VC and TLP Type selected by the following fields: - CREDIT_SEL_VC - CREDIT_SEL_CREDIT_TYPE - CREDIT_SEL_TLP_TYPE - CREDIT_SEL_HD

SD_STATUS_L3FC_REG Offset address: 0004240$_H$

Silicon Debug Status[Layer3 FC) Kernel Reset value: 0000 0000$_H$



Field	Bits	Type	Description
CREDIT_SEL_VC	2:0	rw	**Credit Select(VC)** This field in conjunction with the CREDIT_SEL_CREDIT_TYPE, CREDIT_SEL_TLP_TYPE, and CREDIT_SEL_HD viewport-select fields determines that data that is returned by the CREDIT_DATA0 and CREDIT_DATA1 data fields. Note: This register field is sticky. 000_B **MIN_VAL**: VC0 111_B **MAX_VAL**: VC7
CREDIT_SEL_CREDIT_TYPE	3	rw	**Credit Select(Credit Type)** This field in conjunction with the CREDIT_SEL_VC, CREDIT_SEL_TLP_TYPE, and CREDIT_SEL_HD viewport-select fields determines that data that is returned by the CREDIT_DATA0 and CREDIT_DATA1 data fields. Note: This register field is sticky. 0_B **RX**: Rx 1_B **TX**: Tx
CREDIT_SEL_TLP_TYPE	5:4	rw	**Credit Select(TLP Type)** This field in conjunction with the CREDIT_SEL_VC, CREDIT_SEL_CREDIT_TYPE, and CREDIT_SEL_HD viewport-select fields determines that data that is returned by the CREDIT_DATA0 and CREDIT_DATA1 data fields. Note: This register field is sticky. 00_B **POSTED**: Posted 01_B **NON_POSTED**: Non-Posted 10_B **COMPLETION**: Completion **others,** Reserved
CREDIT_SEL_HD	6	rw	**Credit Select(HeaderData)** This field in conjunction with the CREDIT_SEL_VC, CREDIT_SEL_CREDIT_TYPE, and CREDIT_SEL_TLP_TYPE viewport-select fields determines that data that is returned by the CREDIT_DATA0 and CREDIT_DATA1 data fields. Note: This register field is sticky. 0_B **HEADER_CREDIT**: Header Credit 1_B **DATA_CREDIT**: Data Credit
RSVDP_7	7	r	**Reserved for future use**

(table continues...)

(continued)

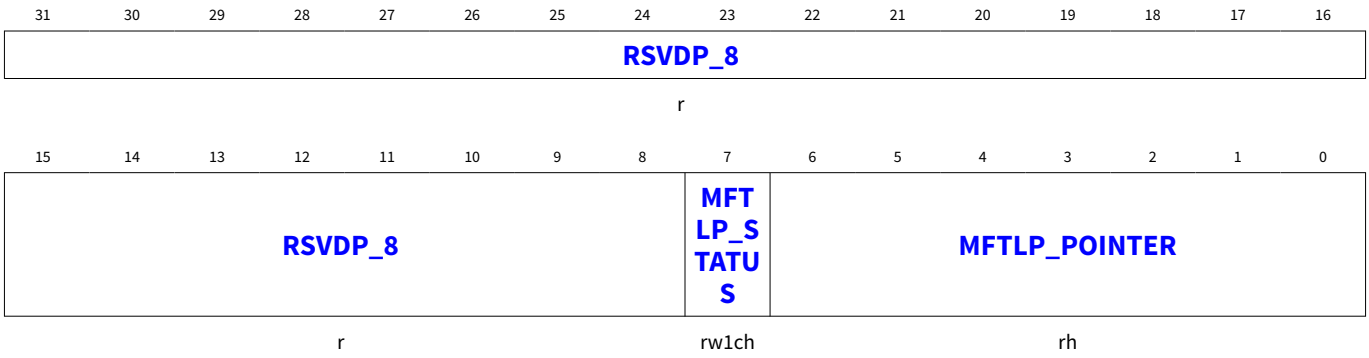
Field	Bits	Type	Description
CREDIT_DATA0	19:8	rh	**Credit Data0** Current FC credit data selected by the CREDIT_SEL_VC, CREDIT_SEL_CREDIT_TYPE, CREDIT_SEL_TLP_TYPE, and CREDIT_SEL_HD viewport-select fields. - Rx: Credit Received Value - Tx: Credit Consumed Value Note: This register field is sticky.
CREDIT_DATA1	31:20	rh	**Credit Data1** Current FC credit data selected by the CREDIT_SEL_VC, CREDIT_SEL_CREDIT_TYPE, CREDIT_SEL_TLP_TYPE, and CREDIT_SEL_HD viewport-select fields. - Rx: Credit Allocated Value - Tx: Credit Limit Value. This value is valid when DLCMSM=0x3(DL_ACTIVE). Note: This register field is sticky.

19.4.1.175 Silicon Debug Status(Layer3)

Silicon Debug Status(Layer3).

SD_STATUS_L3_REG Offset address: 0004244_H

Silicon Debug Status[Layer3] Kernel Reset value: $0000\ 0000_H$



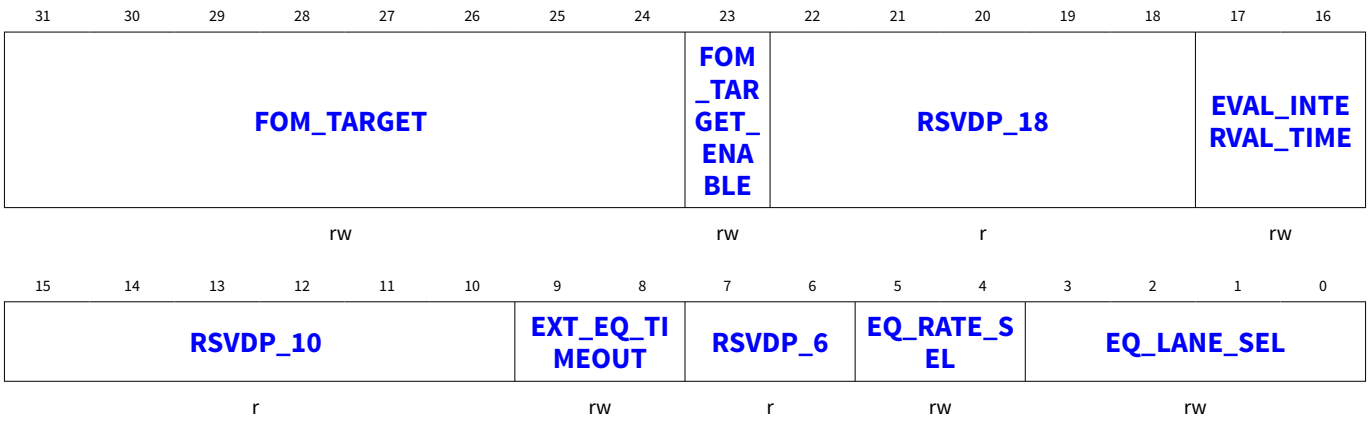
Field	Bits	Type	Description
MFTLP_POINTER	6:0	rh	**First Malformed TLP Error Pointer** Indicates the element of the received first malformed TLP. This pointer is validated by MFTLP_STATUS. All encodings other than the defined encodings are reserved. Note: This register field is sticky. 01_H **ATMC_OP_ALIGN**: AtomicOp address alignment 02_H **ATMC_OP_OPERAND**: AtomicOp operand 03_H **ATMC_OP_BYTE_EN**: AtomicOp byte enable 04_H **TLP_LENGTH_MISMATCH**: TLP length miss match 05_H **MAX_PAYLOAD_SIZE**: Max payload size 06_H **TLP_WITHOUT_TC0**: Message TLP without TC0 07_H **INVALID_TC**: Invalid TC 08_H **UNXPCTD_ROUTE_BIT_MSG_TLP**: Unexpected route bit in Message TLP 09_H **UNXPCTD_CRS_STATUS_CMPL_TLP**: Unexpected CRS status in Completion TLP $0A_H$ **BYTE_ENABLE**: Byte enable $0B_H$ **MEM_ADDR_4KB_BOUNDARY**: Memory Address 4KB boundary $0C_H$ **TLP_PREFIX_RULES**: TLP prefix rules $0D_H$ **TRANSLATION_RULES**: Translation request rules $0E_H$ **INVALID_TLP_TYPE**: Invalid TLP type $0F_H$ **CMPL_RULES**: Completion rules $7F_H$ **APPLICATION**: Application
MFTLP_STATUS	7	rw1ch	**Malformed TLP Status** Indicates malformed TLP has occurred. Note: This register field is sticky. Software write with 1 clears this bit-field.
RSVDP_8	31:8	r	**Reserved for future use**

19.4.1.176 Silicon Debug EQ Control 1

This is a viewport control register. Setting the EQ_RATE_SEL and EQ_LANE_SEL fields in this register determine the per-lane Silicon Debug EQ Status data returned by the SD_EQ_STATUS[1/2/3] viewport registers.

SD_EQ_CONTROL1_REG Offset address: 0004250_H

Silicon Debug EQ Control 1 Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
EQ_LANE_SEL	3:0	rw	**EQ Status Lane Select** Setting this field in conjunction with the EQ_RATE_SEL field determines the per-lane Silicon Debug EQ Status data returned by the SD_EQ_CONTROL[2/3] and SD_EQ_STATUS[1/2/3] viewport registers. Note: This register field is sticky. 0_H **MIN_VAL**: Lane0 F_H **MAX_VAL**: Lane15
EQ_RATE_SEL	5:4	rw	**EQ Status Rate Select** Setting this field in conjunction with the EQ_LANE_SEL field determines the per-lane Silicon Debug EQ Status data returned by the SD_EQ_CONTROL[2/3] and SD_EQ_STATUS[1/2/3] viewport registers. Note: This register field is sticky. 00_B **_8GT**: 8.0GT/s Speed (include ESM data rate) 01_B **_16GT**: 16.0GT/s Speed (include ESM data rate) 10_B **_32GT**: 32.0GT/s Speed **others,** Reserved
RSVDP_6	7:6	r	**Reserved for future use**
EXT_EQ_TIMEOUT	9:8	rw	**Extends EQ Phase2/3 Timeout** This field is used when the LTSSM is in Recovery.EQ2/3. When this field is set, the value of EQ2/3 timeout is extended. Note: This register field is sticky. 00_B **_24MS_OR_32MS**: [EQ Master(DSP in EQ Phase3/USP in EQ Phase2)] 24ms (default) or [EQ Slave(DSP in EQ Phase2/USP in EQ Phase3)] 32ms (default) 01_B **_48MS_OR_56MS**: [EQ Master(DSP in EQ Phase3/USP in EQ Phase2)] 48ms (x2) or [EQ Slave(DSP in EQ Phase2/USP in EQ Phase3)] 56ms (32ms+24ms) 10_B **_240MS_OR_248MS**: [EQ Master(DSP in EQ Phase3/USP in EQ Phase2)] 240ms (x10) or [EQ Slave(DSP in EQ Phase2/USP in EQ Phase3)] 248ms (32ms +9*24ms) 11_B **NO_TIMEOUT**: [EQ Master(DSP in EQ Phase3/USP in EQ Phase2)] No timeout or [EQ Slave(DSP in EQ Phase2/USP in EQ Phase3)] No timeout
RSVDP_10	15:10	r	**Reserved for future use**
EVAL_INTERVAL_TIME	17:16	rw	**Eval Interval Time** Indicates interval time of RxEqEval assertion. This field is used for EQ Master(DSP in EQ Phase3/USP in EQ Phase2). Note: This register field is sticky. 00_B **_500NS**: 500 ns 01_B **_1US**: 1 µs 10_B **_2US**: 2 µs 11_B **_4US**: 4 µs
RSVDP_18	22:18	r	**Reserved for future use**
FOM_TARGET_ENABLE	23	rw	**FOM Target Enable** Enables the FOM_TARGET fields. Note: This register field is sticky.

(table continues...)

(continued)

Field	Bits	Type	Description
FOM_TARGET	31:24	rw	**FOM Target** Indicates figure of merit target criteria value of EQ Master(DSP in EQ Phase3/USP in EQ Phase2). This field is only valid when GEN3_EQ_FB_MODE is 0001b(Figure Of Merit). Note: This register field is sticky.

19.4.1.177 Silicon Debug EQ Control 2

This viewport register returns the value for the rate and lane selected by the EQ_RATE_SEL and EQ_LANE_SEL fields in the SD_EQ_CONTROL1_REG register.

SD_EQ_CONTROL2_REG Offset address: 0004254_H

Silicon Debug EQ Control 2 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_31	FORCE_LOCAL_TX_PRESET_ENABLE	FORCE_LOCAL_RX_HINT_ENABLE	FORCE_LOCAL_TX_COEF_ENABLE	FORCE_LOCAL_TX_PRESET				RSVDP_21			FORCE_LOCAL_RX_HINT			FORCE_LOCAL_TX_POST_CURSOR	
r	rwh	rwh	rwh	rwh				r			rwh			rwh	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FORCE_LOCAL_TX_POST_CURSOR				FORCE_LOCAL_TX_CURSOR						FORCE_LOCAL_TX_PRE_CURSOR					
rwh				rwh						rwh					

Field	Bits	Type	Description
FORCE_LOCAL_TX_PRE_CURSOR	5:0	rwh	**Force Local Transmitter Pre-cursor** Indicates the coefficient value of EQ Slave(DSP in EQ Phase2/USP in EQ Phase3), instead of the value instructed from link partner. Note: This register field is sticky.
FORCE_LOCAL_TX_CURSOR	11:6	rwh	**Force Local Transmitter Cursor** Indicates the coefficient value of EQ Slave(DSP in EQ Phase2/USP in EQ Phase3), instead of the value instructed from link partner. Note: This register field is sticky.
FORCE_LOCAL_TX_POST_CURSOR	17:12	rwh	**Force Local Transmitter Post-Cursor** Indicates the coefficient value of EQ Slave(DSP in EQ Phase2/USP in EQ Phase3), instead of the value instructed from link partner. Note: This register field is sticky.
FORCE_LOCAL_RX_HINT	20:18	rwh	**Force Local Receiver Preset Hint** Indicates the RxPresetHint value of EQ Slave(DSP in EQ Phase2/USP in EQ Phase3), instead of received or set value. If select rate in the EQ_RATE_SEL field is other than 8.0GT/s Speed, this feature is not available. Note: This register field is sticky.

(table continues...)

(continued)

Field	Bits	Type	Description
RSVDP_21	23:21	r	**Reserved for future use**
FORCE_LOCAL_TX_PRESET	27:24	rwh	**Force Local Transmitter Preset** Indicates initial preset value of USP in EQ Slave(EQ Phase2) instead of receiving EQ TS2. If select rate in the EQ_RATE_SEL field is 32.0GT/s Speed, this feature is not available. Note: This register field is sticky.
FORCE_LOCAL_TX_COEF_ENABLE	28	rwh	**Force Local Transmitter Coefficient Enable** Enables the following fields: - FORCE_LOCAL_TX_PRE_CURSOR - FORCE_LOCAL_TX_CURSOR - FORCE_LOCAL_TX_POST_CURSOR Note: This register field is sticky.
FORCE_LOCAL_RX_HINT_ENABLE	29	rwh	**Force Local Receiver Preset Hint Enable** Enables the FORCE_LOCAL_RX_HINT field. If select rate in the EQ_RATE_SEL field is other than 8.0GT/s Speed, this feature is not available. Note: This register field is sticky.
FORCE_LOCAL_TX_PRESET_ENABLE	30	rwh	**Force Local Transmitter Preset Enable** Enables the FORCE_LOCAL_TX_PRESET field. If select rate in the EQ_RATE_SEL field is 32.0GT/s Speed, this feature is not available. Note: This register field is sticky.
RSVDP_31	31	r	**Reserved for future use**

19.4.1.178 Silicon Debug EQ Control 3

This viewport register returns the value for the rate and lane selected by the EQ_RATE_SEL and EQ_LANE_SEL fields in the SD_EQ_CONTROL1_REG register..

SD_EQ_CONTROL3_REG Offset address: 0004258_H

Silicon Debug EQ Control 3 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_29			FORCE_REMOTE_TX_COEF_ENABLE	RSVDP_18										FORCE_REMOTE_TX_POST_CURSOR	
r			rwh	r										rwh	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FORCE_REMOTE_TX_POST_CURSOR				FORCE_REMOTE_TX_CURSOR						FORCE_REMOTE_TX_PRE_CURSOR					
rwh				rwh						rwh					

Field	Bits	Type	Description
FORCE_REMOTE_TX_PRE_CURSOR	5:0	rwh	**Force Remote Transmitter Pre-Cursor** Indicates the coefficient value of EQ Master(DSP in EQ Phase3/USP in EQ Phase2), instead of the value instructed from local phy in dirchange mode. Note: This register field is sticky.
FORCE_REMOTE_TX_CURSOR	11:6	rwh	**Force Remote Transmitter Cursor** Indicates the coefficient value of EQ Master(DSP in EQ Phase3/USP in EQ Phase2), instead of the value instructed from local phy in dirchange mode. Note: This register field is sticky.
FORCE_REMOTE_TX_POST_CURSOR	17:12	rwh	**Force Remote Transmitter Post-Cursor** Indicates the coefficient value of EQ Master(DSP in EQ Phase3/USP in EQ Phase2), instead of the value instructed from local phy in dirchange mode. Note: This register field is sticky.
RSVDP_18	27:18	r	**Reserved for future use**
FORCE_REMOTE_TX_COEF_ENABLE	28	rwh	**Force Remote Transmitter Coefficient Enable** Enables the following fields: - FORCE_REMOTE_TX_PRE_CURSOR - FORCE_REMOTE_TX_CURSOR - FORCE_REMOTE_TX_POST_CURSOR This function can only be used when GEN3_EQ_FB_MODE = 0000b(Direction Change) Note: This register field is sticky.
RSVDP_29	31:29	r	**Reserved for future use**

19.4.1.179 Silicon Debug EQ Status 1

This viewport register returns the first of three words of Silicon Debug EQ Status data for the rate and lane selected by the EQ_RATE_SEL and EQ_LANE_SEL fields in the SD_EQ_CONTROL1_REG register. The following fields are available when Equalization finished unsuccessfully(EQ_CONVERGENCE_INFO=2). - EQ_RULEA_VIOLATION - EQ_RULEB_VIOLATION - EQ_RULEC_VIOLATION - EQ_REJECT_EVENT

SD_EQ_STATUS1_REG Offset address: 0004260_H

Silicon Debug EQ Status 1 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_8															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVDP_8								EQ_REJECT_EVENT	EQ_RULEC_VIOLATION	EQ_RULEB_VIOLATION	EQ_RULEA_VIOLATION	RSVDP_3	EQ_CONVERGENCE_INFO		EQ_SEQUENCE
r								rh	rh	rh	rh	r	rh		rh

Field	Bits	Type	Description
EQ_SEQUENCE	0	rh	**EQ Sequence** Indicates that the controller is starting the equalization sequence. Note: This register field is sticky.

(table continues...)

(continued)

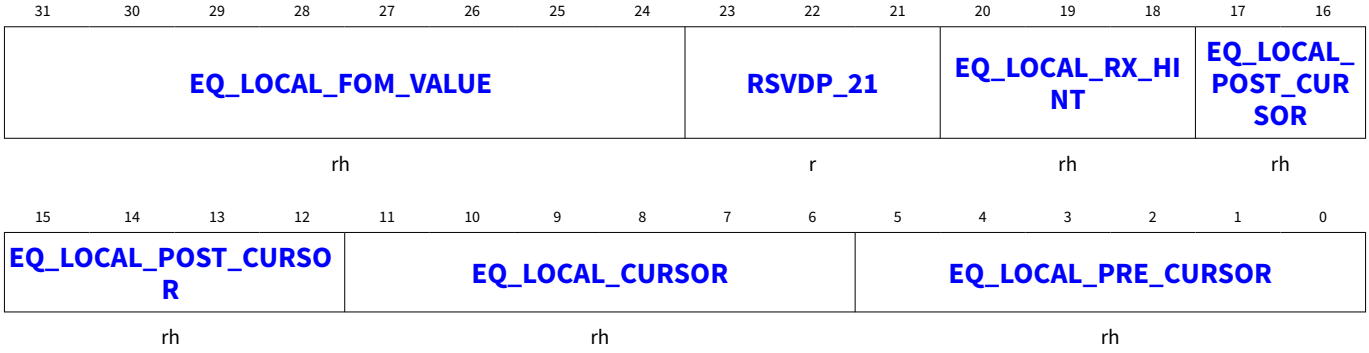
Field	Bits	Type	Description
EQ_CONVERGENCE_INFO	2:1	rh	**EQ Convergence Info** Indicates equalization convergence information. This bit is automatically cleared when the controller starts EQ Master phase again. Note: This register field is sticky. 00_B **EQ_NOT_ATTEMPTED**: Equalization is not attempted 01_B **EQ_SUCCESSFUL**: Equalization finished successfully 10_B **EQ_UNSUCCESSFUL**: Equalization finished unsuccessfully 11_B **RSVD**: Reserved
RSVDP_3	3	r	**Reserved for future use**
EQ_RULEA_VIOLATION	4	rh	**EQ Rule A Violation** Indicates that coefficients rule A violation is detected in the values provided by PHY using direction change method during EQ Master phase(DSP in EQ Phase3/USP in EQ Phase2). The coefficients rules A correspond to the rules a) from section "Rules for Transmitter Coefficents" in the PCI Express Base Specification. This bit is automatically cleared when the controller starts EQ Master phase again. Note: This register field is sticky.
EQ_RULEB_VIOLATION	5	rh	**EQ Rule B Violation** Indicates that coefficients rule B violation is detected in the values provided by PHY using direction change method during EQ Master phase(DSP in EQ Phase3/USP in EQ Phase2). The coefficients rules B correspond to the rules b) from section "Rules for Transmitter Coefficents" in the PCI Express Base Specification. This bit is automatically cleared when the controller starts EQ Master phase again. Note: This register field is sticky.
EQ_RULEC_VIOLATION	6	rh	**EQ Rule C Violation** Indicates that coefficients rule C violation is detected in the values provided by PHY using direction change method during EQ Master phase(DSP in EQ Phase3/USP in EQ Phase2). The coefficients rule C correspond to the rules c) from section "Rules for Transmitter Coefficents" in the PCI Express Base Specification. This bit is automatically cleared when the controller starts EQ Master phase again. Note: This register field is sticky.
EQ_REJECT_EVENT	7	rh	**EQ Reject Event** Indicates that the controller receives two consecutive TS1 OS w/ Reject=1b during EQ Master phase(DSP in EQ Phase3/USP in EQ Phase2). This bit is automatically cleared when the controller starts EQ Master phase again. Note: This register field is sticky.
RSVDP_8	31:8	r	**Reserved for future use**

19.4.1.180 Silicon Debug EQ Status 2

This viewport register returns the second of three words of Silicon Debug EQ Status data for the rate and lane selected by the EQ_RATE_SEL and EQ_LANE_SEL fields in the SD_EQ_CONTROL1_REG register. Each field is available when Equalization finished successfully(EQ_CONVERGENCE_INFO=1).

SD_EQ_STATUS2_REG Offset address: 0004264_H

Silicon Debug EQ Status 2 Kernel Reset value: $0000\ 0000_H$



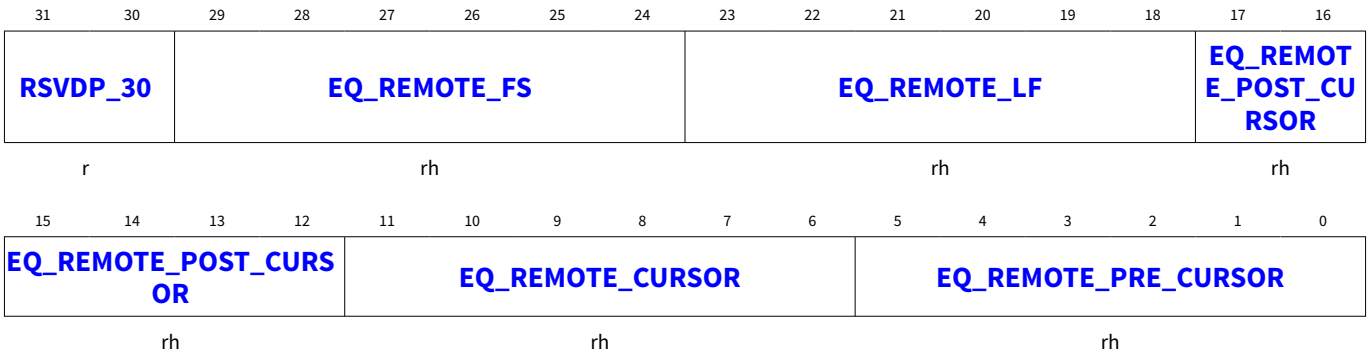
Field	Bits	Type	Description
EQ_LOCAL_PRE_CURSOR	5:0	rh	**EQ Local Pre-Cursor** Indicates Local pre cursor coefficient value. Note: This register field is sticky.
EQ_LOCAL_CURSOR	11:6	rh	**EQ Local Cursor** Indicates Local cursor coefficient value. Note: This register field is sticky.
EQ_LOCAL_POST_CURSOR	17:12	rh	**EQ Local Post-Cursor** Indicates Local post cursor coefficient value. Note: This register field is sticky.
EQ_LOCAL_RX_HINT	20:18	rh	**EQ Local Receiver Preset Hint** Indicates Local Receiver Preset Hint value. If select rate in the EQ_RATE_SEL field is other than 8.0GT/s Speed, this feature is not available. Note: This register field is sticky.
RSVDP_21	23:21	r	**Reserved for future use**
EQ_LOCAL_FOM_VALUE	31:24	rh	**EQ Local Figure of Merit** Indicates Local maximum Figure of Merit value. Note: This register field is sticky.

19.4.1.181 Silicon Debug EQ Status 3

This viewport register returns the third of three words of Silicon Debug EQ Status data for the rate and lane selected by the EQ_RATE_SEL and EQ_LANE_SEL fields in the SD_EQ_CONTROL1_REG register. Each field is available when Equalization finished successfully(EQ_CONVERGENCE_INFO=1).

SD_EQ_STATUS3_REG Offset address: 0004268_H

Silicon Debug EQ Status 3 Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
EQ_REMOTE_PRE_CURSOR	5:0	rh	**EQ Remote Pre-Cursor** Indicates Remote pre cursor coefficient value. Note: This register field is sticky.
EQ_REMOTE_CURSOR	11:6	rh	**EQ Remote Cursor** Indicates Remote cursor coefficient value. Note: This register field is sticky.
EQ_REMOTE_POST_CURSOR	17:12	rh	**EQ Remote Post-Cursor** Indicates Remote post cursor coefficient value. Note: This register field is sticky.
EQ_REMOTE_LF	23:18	rh	**EQ Remote LF** Indicates Remote LF value. Note: This register field is sticky.
EQ_REMOTE_FS	29:24	rh	**EQ Remote FS** Indicates Remote FS value. Note: This register field is sticky.
RSVDP_30	31:30	r	**Reserved for future use**

19.4.1.182 PCIe Extended capability ID, Capability version and Next capability offset

This Register provides capability ID, Capability version, and Next capability offset for PCIe Extended capability stucture.

RASDP_EXT_CAP_HDR_OFF

PCIe Extended capability ID, Capability version and Next capability offset

Offset address: 0004280_H

Kernel Reset value: $2B81\,000B_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
NEXT_OFFSET												CAP			
rw												rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ID															
rw															

Field	Bits	Type	Description
ID	15:0	rw	**PCI Express Extended Capability ID** This field is a PCI-SIG defined ID number that indicates the nature and format of the Extended Capability. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky. 0000_H **MIN_VAL**: Min value $FFFF_H$ **MAX_VAL**: Max value

(table continues...)

(continued)

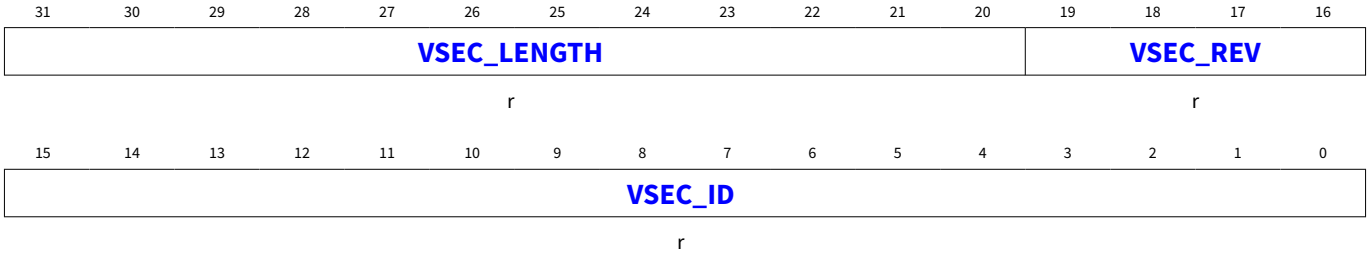
Field	Bits	Type	Description
CAP	19:16	rw	**Capability Version** This field is a PCI-SIG defined version number that indicates the version of the Capability structure present. A version of the specification that changes the Extended Capability in a way that is not otherwise identifiable (e.g., through a new Capability field) is permitted to increment this field. All such changes to the Capability structure must be software-compatible. Software must check for Capability Version numbers that are greater than or equal to the highest number defined when the software is written, as Functions reporting any such Capability Version numbers will contain a Capability structure that is compatible with that piece of software. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky. 0_H **MIN_VAL**: Min value F_H **MAX_VAL**: Max value
NEXT_OFFSET	31:20	rw	**Next Capability Offset** This field contains the offset to the next PCI Express Capability structure or 000h if no other items exist in the linked list of Capabilities. For Extended Capabilities implemented in Configuration Space, this offset is relative to the beginning of PCI-compatible Configuration Space and thus must always be either 000h (for terminating list of Capabilities) or greater than 0FFh. The bottom 2 bits of this offset are Reserved and must be implemented as 00b although software must mask them to allow for future uses of these bits. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky. 000_H **MIN_VAL**: Min value FFF_H **MAX_VAL**: Max value

19.4.1.183 Vendor Specific Header

This Register provides VSEC Length, VSEC ID and VSEC Rev (Version Number). Vendor-specific software must qualify the associated Vendor ID of the PCI Express Function or RCRB before attempting to interpret the values in the VSEC ID or VSEC Rev fields.

RASDP_VENDOR_SPECIFIC_HDR_OFF Offset address: 0004284_H

Vendor Specific Header Kernel Reset value: $0381\ 0001_H$



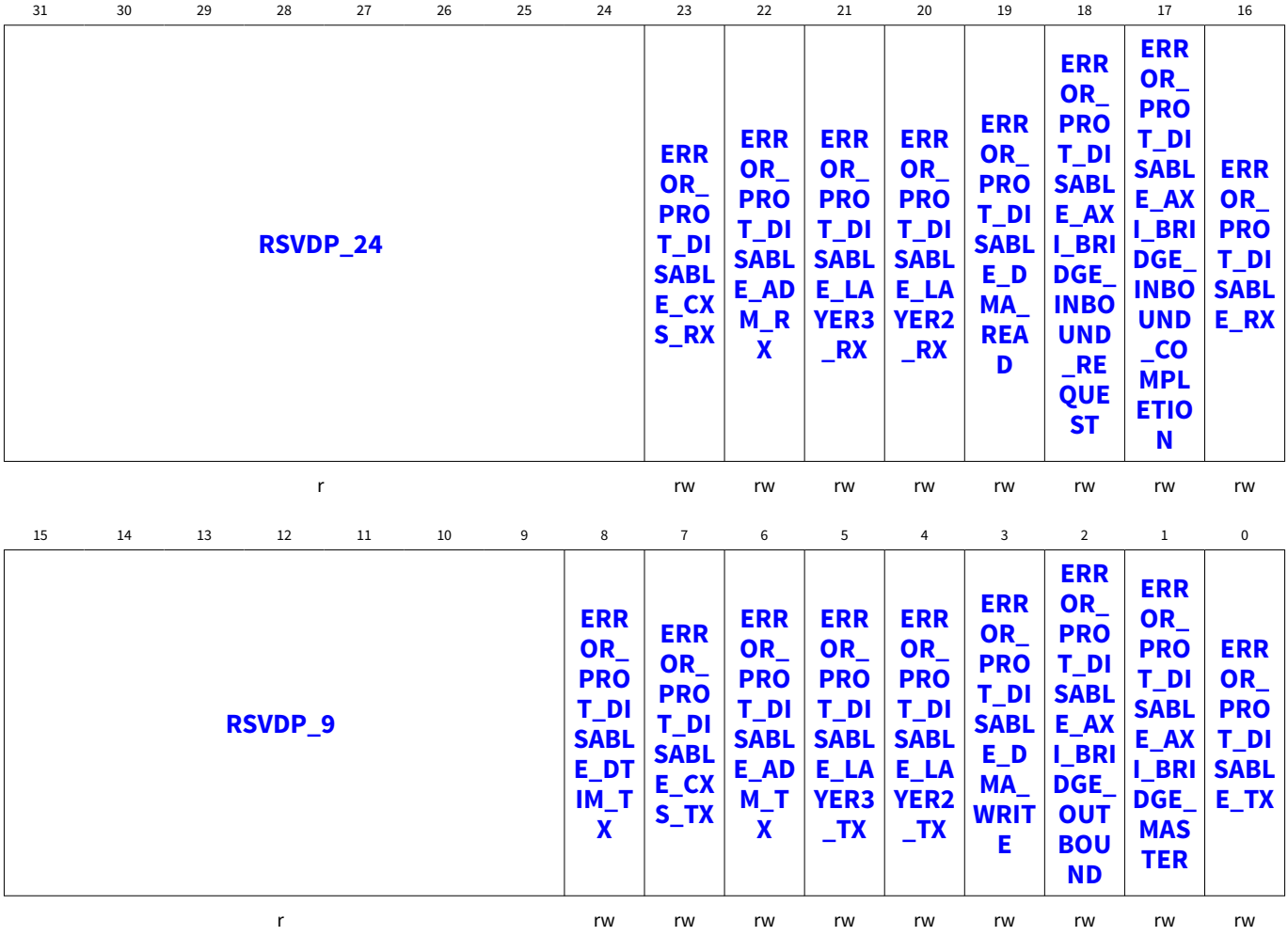
Field	Bits	Type	Description
VSEC_ID	15:0	r	**VSEC ID** This field is a vendor-defined ID number that indicates the nature and format of the VSEC structure. Software must qualify the Vendor ID before interpreting this field. Note: This register field is sticky. 0000_H **MIN_VAL**: Min value $FFFF_H$ **MAX_VAL**: Max value
VSEC_REV	19:16	r	**VSEC Rev** This field is a vendor-defined version number that indicates the version of the VSEC structure. Software must qualify the Vendor ID and VSEC ID before interpreting this field. Note: This register field is sticky. 0_H **MIN_VAL**: Min value F_H **MAX_VAL**: Max value
VSEC_LENGTH	31:20	r	**VSEC Length** This field indicates the number of bytes in the entire VSEC structure, including the Vendor-Specific Extended Capability Header, the Vendor-Specific Header, and the vendor-specific registers. Note: This register field is sticky. 000_H **MIN_VAL**: Min value FFF_H **MAX_VAL**: Max value

19.4.1.184 ECC Error Correction and Control Register

Allows you to disable ECC error correction for RAMs and datapath. When the AXI Bridge Module is implemented and the master / slave clocks are asynchronous to the PCIe native controller clock (core_clk), you must not write this register while operations are in progress in the AXI master / slave interface.

RASDP_ERROR_PROT_CTRL_OFF Offset address: 0004288_H

ECC Error Correction and Control Register Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
ERROR_PROT_DISABLE_TX	0	rw	**Global error correction disable for all Tx layers** Does not disable the error detection reporting for 1-bit and 2-bit ECC errors. Note: This register field is sticky.
ERROR_PROT_DISABLE_AXI_BRIDGE_MASTER	1	rw	**Error correction disable for AXI bridge master completion buffer** Note: This register field is sticky.
ERROR_PROT_DISABLE_AXI_BRIDGE_OUTBOUND	2	rw	**Error correction disable for AXI bridge outbound request path** Note: This register field is sticky.
ERROR_PROT_DISABLE_DMA_WRITE	3	rw	**Error correction disable for DMA write engine** Note: This register field is sticky.
ERROR_PROT_DISABLE_LAYER2_TX	4	rw	**Error correction disable for layer 2 Tx path** Note: This register field is sticky.

(table continues...)

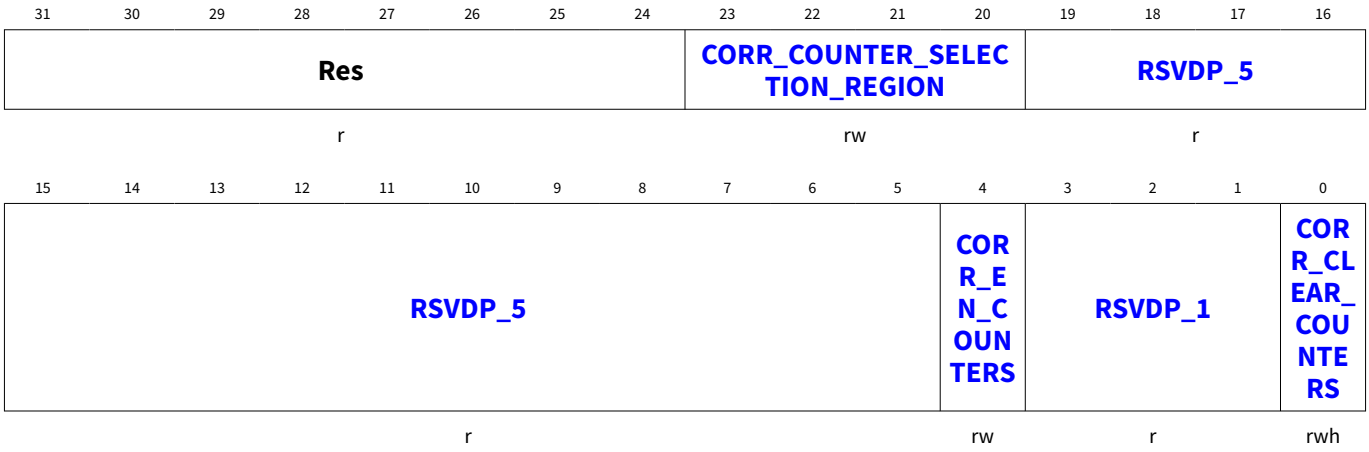
(continued)

Field	Bits	Type	Description
ERROR_PROT_DISABLE_LAYER3_TX	5	rw	**Error correction disable for layer 3 Tx path** Note: This register field is sticky.
ERROR_PROT_DISABLE_ADM_TX	6	rw	**Error correction disable for Adm Tx path** Note: This register field is sticky.
ERROR_PROT_DISABLE_CXS_TX	7	rw	**Error correction disable for CXS Rx path (PCIe Tx path)** Note: This register field is sticky.
ERROR_PROT_DISABLE_DTIM_TX	8	rw	**Error correction disable for DTIM Tx path** Note: This register field is sticky.
RSVDP_9	15:9	r	**Reserved for future use**
ERROR_PROT_DISABLE_RX	16	rw	**Global error correction disable for all Rx layers** Note: This register field is sticky.
ERROR_PROT_DISABLE_AXI_BRIDGE_INBOUND_COMPLETION	17	rw	**Error correction disable for AXI bridge inbound completion composer** Does not disable the error detection reporting for 1-bit and 2-bit ECC errors. Note: This register field is sticky.
ERROR_PROT_DISABLE_AXI_BRIDGE_INBOUND_REQUEST	18	rw	**Error correction disable for AXI bridge inbound request path** Note: This register field is sticky.
ERROR_PROT_DISABLE_DMA_READ	19	rw	**Error correction disable for DMA read engine** Note: This register field is sticky.
ERROR_PROT_DISABLE_LAYER2_RX	20	rw	**Error correction disable for layer 2 Rx path** Note: This register field is sticky.
ERROR_PROT_DISABLE_LAYER3_RX	21	rw	**Error correction disable for layer 3 Rx path** Note: This register field is sticky.
ERROR_PROT_DISABLE_ADM_RX	22	rw	**Error correction disable for ADM Rx path** Note: This register field is sticky.
ERROR_PROT_DISABLE_CXS_RX	23	rw	**Error correction disable for CXS Tx path (PCIe Rx path)** Note: This register field is sticky.
RSVDP_24	31:24	r	**Reserved for future use**

19.4.1.185 Corrected Error (1-bit ECC) Counter Selection and Control Register

This is a viewport control register. Setting the CORR_COUNTER_SELECTION_REGION and CORR_COUNTER_SELECTION fields in this register determine the counter data returned by the RASDP_CORR_COUNT_REPORT_OFF viewport data register.

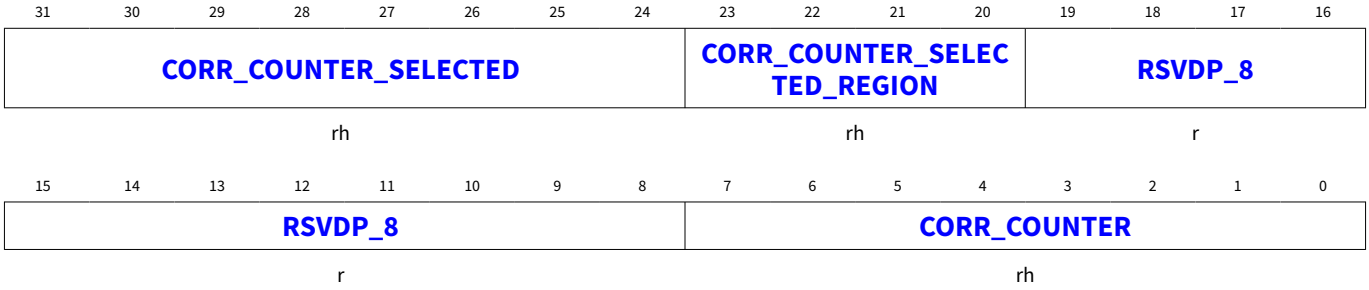
19 Peripheral Component Interconnect Express (PCIe)



Field	Bits	Type	Description
CORR_CLEAR_COUNTERS	0	rwh	**Clear all correctable error counters** Reading register field always returns a '0'.
RSVDP_1	3:1	r	**Reserved for future use**
CORR_EN_COUNTERS	4	rw	**Enable correctable errors counters** The counters are enabled by default. Note: This register field is sticky. 0_B **COUNTER_FROZEN**: Counters are frozen 1_B **COUNTER_EN**: Counters increment when the controller detects a correctable error
RSVDP_5	19:5	r	**Reserved for future use**
CORR_COUNTER_SELECTION_REGION	23:20	rw	**Select correctable counter region** All encodings other than the defined encodings are reserved. Note: This register field is sticky. 0_H **ADM_RX_PATH**: Region select for Adm Rx path 1_H **L3_RX_PATH**: Region select for layer 3 Rx path 2_H **L2_RX_PATH**: Region select for layer 2 Rx path 3_H **DMA_INBOUND_PATH**: Region select for DMA inbound path 4_H **AXI_INBOUND_REQ_PATH**: Region select for AXI bridge inbound request path 5_H **AXI_INBOUND_CMPL_PATH**: Region select for AXI bridge inbound completion composer path 6_H **ADM_TX_PATH**: Region select for Adm Tx path 7_H **L3_TX_PATH**: Region select for layer 3 Tx path 8_H **L2_TX_PATH**: Region select for layer 2 Tx path 9_H **DMA_OUTBOUND_PATH**: Region select for DMA outbound path A_H **AXI_OUTBOUND_REQ_PATH**: Region select for AXI bridge outbound request path B_H **AXI_OUTBOUND_CMPL_PATH**: Region select for AXI bridge outbound master completion buffer path C_H **CXS_TX_RX_PATH**: Region select for CXS Tx/Rx path D_H **DTIM_TX_RX_PATH**: Region select for DTIM Tx/Rx path **others,** Reserved

19.4.1.186 Corrected Error (1-bit ECC) Counter Data Register

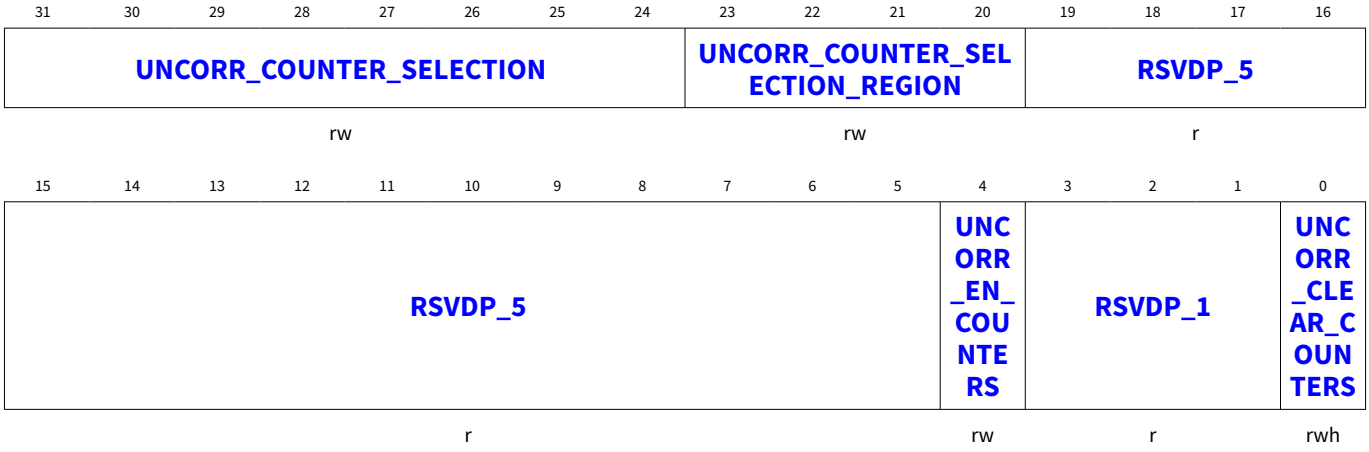
This viewport register returns the counter data selected by the CORR_COUNTER_SELECTION_REGION and CORR_COUNTER_SELECTION fields in the RASDP_CORR_COUNTER_CTRL_OFF register.



Field	Bits	Type	Description
CORR_COUNTER	7:0	rh	**Current corrected error count for the selected counter** Note: This register field is sticky.
RSVDP_8	19:8	r	**Reserved for future use**
CORR_COUNTER_SELECTED_REGION	23:20	rh	**Selected correctable counter region** All encodings other than the defined encodings are reserved. Note: This register field is sticky. 0_H **ADM_RX_PATH**: Region select for Adm Rx path 1_H **L3_RX_PATH**: Region select for layer 3 Rx path 2_H **L2_RX_PATH**: Region select for layer 2 Rx path 3_H **DMA_INBOUND_PATH**: Region select for DMA inbound path 4_H **AXI_INBOUND_REQ_PATH**: Region select for AXI bridge inbound request path 5_H **AXI_INBOUND_CMPL_PATH**: Region select for AXI bridge inbound completion composer path 6_H **ADM_TX_PATH**: Region select for Adm Tx path 7_H **L3_TX_PATH**: Region select for layer 3 Tx path 8_H **L2_TX_PATH**: Region select for layer 2 Tx path 9_H **DMA_OUTBOUND_PATH**: Region select for DMA outbound path A_H **AXI_OUTBOUND_REQ_PATH**: Region select for AXI bridge outbound request path B_H **AXI_OUTBOUND_CMPL_PATH**: Region select for AXI bridge outbound master completion buffer path C_H **CXS_TX_RX_PATH**: Region select for CXS Tx/Rx path D_H **DTIM_TX_RX_PATH**: Region select for DTIM Tx/Rx path **others,** Reserved
CORR_COUNTER_SELECTED	31:24	rh	**Counter selection** Returns the value set in the CORR_COUNTER_SELECTION field of the RASDP_CORR_COUNTER_CTRL_OFF register. Note: This register field is sticky.

19.4.1.187 Uncorrected Error (2-bit ECC and parity) Counter Selection and Control Register

This is a viewport control register. Setting the UNCORR_COUNTER_SELECTION_REGION and UNCORR_COUNTER_SELECTION fields in this register determine the counter data returned by the RASDP_UNCORR_COUNT_REPORT_OFF viewport data register.



Field	Bits	Type	Description
UNCORR_CLEAR_COUNTERS	0	rwh	**Clear uncorrectable errors counters** When asserted causes all counters tracking the uncorrectable errors to be cleared. Reading register field always returns a '0'.
RSVDP_1	3:1	r	**Reserved for future use**
UNCORR_EN_COUNTERS	4	rw	**Enable uncorrectable errors counters** The counters are enabled by default. Note: This register field is sticky. 0_B **COUNTER_FROZEN**: Counters are frozen 1_B **ENABLE_COUNTER**: Enables the counters to increment on detected correctable errors
RSVDP_5	19:5	r	**Reserved for future use**

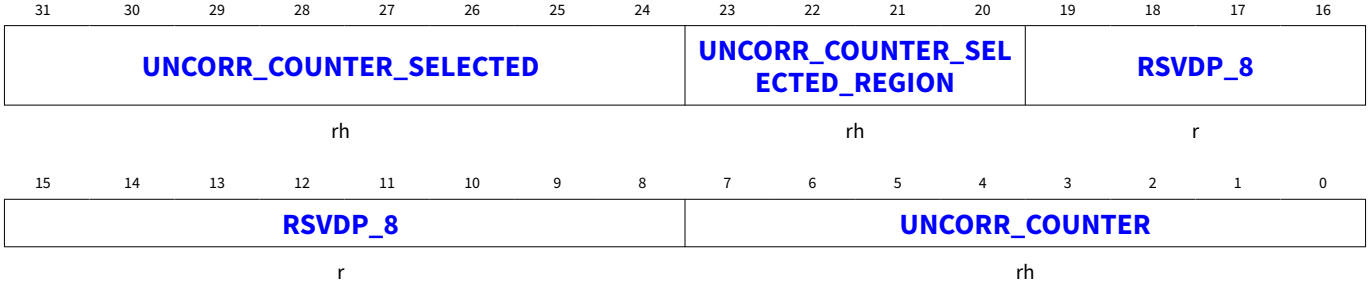
(table continues...)

(continued)

Field	Bits	Type	Description
UNCORR_COUNTER_SELECTION_REGION	23:20	rw	**Select uncorrectable counter region** All encodings other than the defined encodings are reserved. Note: This register field is sticky. 0_H **ADM_RX_PATH**: Region select for Adm Rx path 1_H **L3_RX_PATH**: Region select for layer 3 Rx path 2_H **L2_RX_PATH**: Region select for layer 2 Rx path 3_H **DMA_INBOUND_PATH**: Region select for DMA inbound path 4_H **AXI_INBOUND_REQ_PATH**: Region select for AXI bridge inbound request path 5_H **AXI_INBOUND_CMPL_PATH**: Region select for AXI bridge inbound completion composer path 6_H **ADM_TX_PATH**: Region select for Adm Tx path 7_H **L3_TX_PATH**: Region select for layer 3 Tx path 8_H **L2_TX_PATH**: Region select for layer 2 Tx path 9_H **DMA_OUTBOUND_PATH**: Region select for DMA outbound path A_H **AXI_OUTBOUND_REQ_PATH**: Region select for AXI bridge outbound request path B_H **AXI_OUTBOUND_CMPL_PATH**: Region select for AXI bridge outbound master completion buffer path C_H **CXS_TX_RX_PATH**: Region select for CXS Tx/Rx path D_H **DTIM_TX_RX_PATH**: Region select for DTIM Tx/Rx path **others,** Reserved
UNCORR_COUNTER_SELECTION	31:24	rw	**Counter selection** This field selects the counter ID (within the region defined by UNCORR_COUNTER_SELECTION_REGION) whose contents can be read from the RASDP_UNCORR_COUNT_REPORT_OFF register. You can cycle this field value from 0 to 255 to access all counters according to the detailed report of check points at http://www.synopsys.com/dw/doc.php/iip/DWC_pcie/latest/doc/RASDP_CheckPoints.pdf Note: This register field is sticky.

19.4.1.188 Uncorrected Error (2-bit ECC and parity) Counter Data Register

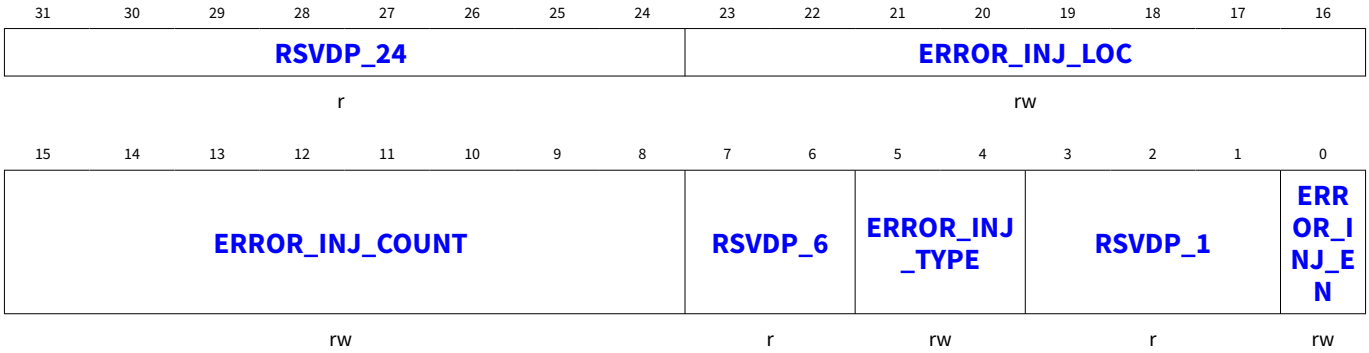
This viewport register returns the counter data selected by the UNCORR_COUNTER_SELECTION_REGION and UNCORR_COUNTER_SELECTION fields in the RASDP_UNCORR_COUNTER_CTRL_OFF register.



Field	Bits	Type	Description
UNCORR_COUNTER	7:0	rh	**Current uncorrected error count for the selected counter** Note: This register field is sticky.
RSVDP_8	19:8	r	**Reserved for future use**
UNCORR_COUNTER_SELECTED_REGION	23:20	rh	**Selected uncorrectable counter region** All encodings other than the defined encodings are reserved. Note: This register field is sticky. 0_H **ADM_RX_PATH**: Region select for Adm Rx path 1_H **L3_RX_PATH**: Region select for layer 3 Rx path 2_H **L2_RX_PATH**: Region select for layer 2 Rx path 3_H **DMA_INBOUND_PATH**: Region select for DMA inbound path 4_H **AXI_INBOUND_REQ_PATH**: Region select for AXI bridge inbound request path 5_H **AXI_INBOUND_CMPL_PATH**: Region select for AXI bridge inbound completion composer path 6_H **ADM_TX_PATH**: Region select for Adm Tx path 7_H **L3_TX_PATH**: Region select for layer 3 Tx path 8_H **L2_TX_PATH**: Region select for layer 2 Tx path 9_H **DMA_OUTBOUND_PATH**: Region select for DMA outbound path A_H **AXI_OUTBOUND_REQ_PATH**: Region select for AXI bridge outbound request path B_H **AXI_OUTBOUND_CMPL_PATH**: Region select for AXI bridge outbound master completion buffer path C_H **CXS_TX_RX_PATH**: Region select for CXS Tx/Rx path D_H **DTIM_TX_RX_PATH**: Region select for DTIM Tx/Rx path **others,** Reserved
UNCORR_COUNTER_SELECTED	31:24	rh	**Counter selection** Returns the value set in the UNCORR_COUNTER_SELECTION field of the RASDP_UNCORR_COUNTER_CTRL_OFF register. Note: This register field is sticky.

19.4.1.189 Error Injection Control Register

Error injection control for the following features: - 1-bit or 2-bit injection - Continuous or fixed-number (n) injection modes - Global enable/disable - Selectable location where injection occurs



Field	Bits	Type	Description
ERROR_INJ_EN	0	rw	**Error injection global enable** When set, enables the error insertion logic.
RSVDP_1	3:1	r	**Reserved for future use**
ERROR_INJ_TYPE	5:4	rw	**Error injection type** Note: This register field is sticky. 00_B **NONE**: none 01_B **_1_BIT**: 1-bit 10_B **_2_BIT**: 2-bit **others,** Reserved
RSVDP_6	7:6	r	**Reserved for future use**
ERROR_INJ_COUNT	15:8	rw	**Error injection count** If value is n, n amount of errors injected. Note: This register field is sticky. 00_H **ERR_INJ_EVERY_TLP**: Errors are inserted in every TLP until you clear ERROR_INJ_EN 01_H **ERR_INJ_1**: One errors injected 02_H **ERR_INJ_2**: Two errors injected **others,** Reserved
ERROR_INJ_LOC	23:16	rw	**Error injection location** Selects where error injection takes place. You can cycle this field value from 0 to 255 to access all locations according to the detailed report of check points at RASDP_CheckPoints.pdf Note: This register field is sticky.
RSVDP_24	31:24	r	**Reserved for future use**

19.4.1.190 Corrected Errors Location Register

Corrected errors locations.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LOC_LAST_CORR_ERROR								REG_LAST_CORR_ERROR				RSVDP_16			
rh								rh				r			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LOC_FIRST_CORR_ERROR								REG_FIRST_CORR_ERROR				RSVDP_0			
rh								rh				r			

Field	Bits	Type	Description
RSVDP_0	3:0	r	**Reserved for future use**

(table continues...)

(continued)

Field	Bits	Type	Description
REG_FIRST_CORR_ERROR	7:4	rh	**Region of the first corrected error** All encodings other than the defined encodings are reserved. Note: This register field is sticky. 0_H **ADM_RX_PATH**: Region select for Adm Rx path 1_H **L3_RX_PATH**: Region select for layer 3 Rx path 2_H **L2_RX_PATH**: Region select for layer 2 Rx path 3_H **DMA_INBOUND_PATH**: Region select for DMA inbound path 4_H **AXI_INBOUND_REQ_PATH**: Region select for AXI bridge inbound request path 5_H **AXI_INBOUND_CMPL_PATH**: Region select for AXI bridge inbound completion composer path 6_H **ADM_TX_PATH**: Region select for Adm Tx path 7_H **L3_TX_PATH**: Region select for layer 3 Tx path 8_H **L2_TX_PATH**: Region select for layer 2 Tx path 9_H **DMA_OUTBOUND_PATH**: Region select for DMA outbound path A_H **AXI_OUTBOUND_REQ_PATH**: Region select for AXI bridge outbound request path B_H **AXI_OUTBOUND_CMPL_PATH**: Region select for AXI bridge outbound master completion buffer path C_H **CXS_TX_RX_PATH**: Region select for CXS Tx/Rx path D_H **DTIM_TX_RX_PATH**: Region select for DTIM Tx/Rx path **others,** Reserved
LOC_FIRST_CORR_ERROR	15:8	rh	**Location/ID of the first corrected error within the region defined by REG_FIRST_CORR_ERROR** You can cycle this field value from 0 to 255 to access all counters according to the detailed report of check points RASDP_CheckPoints.pdf Note: This register field is sticky.
RSVDP_16	19:16	r	**Reserved for future use**

(table continues...)

(continued)

Field	Bits	Type	Description
REG_LAST_CORR_ERROR	23:20	rh	**Region of the last corrected error** All encodings other than the defined encodings are reserved. Note: This register field is sticky. 0_H **ADM_RX_PATH**: Region select for Adm Rx path 1_H **L3_RX_PATH**: Region select for layer 3 Rx path 2_H **L2_RX_PATH**: Region select for layer 2 Rx path 3_H **DMA_INBOUND_PATH**: Region select for DMA inbound path 4_H **AXI_INBOUND_REQ_PATH**: Region select for AXI bridge inbound request path 5_H **AXI_INBOUND_CMPL_PATH**: Region select for AXI bridge inbound completion composer path 6_H **ADM_TX_PATH**: Region select for Adm Tx path 7_H **L3_TX_PATH**: Region select for layer 3 Tx path 8_H **L2_TX_PATH**: Region select for layer 2 Tx path 9_H **DMA_OUTBOUND_PATH**: Region select for DMA outbound path A_H **AXI_OUTBOUND_REQ_PATH**: Region select for AXI bridge outbound request path B_H **AXI_OUTBOUND_CMPL_PATH**: Region select for AXI bridge outbound master completion buffer path C_H **CXS_TX_RX_PATH**: Region select for CXS Tx/Rx path D_H **DTIM_TX_RX_PATH**: Region select for DTIM Tx/Rx path **others,** Reserved
LOC_LAST_CORR_ERROR	31:24	rh	**Location/ID of the last corrected error within the region defined by REG_LAST_CORR_ERROR** You can cycle this field value from 0 to 255 to access all counters according to the detailed report of check points at http://www.synopsys.com/dw/doc.php/iip/DWC_pcie/latest/doc/RASDP_CheckPoints.pdf Note: This register field is sticky.

19.4.1.191 Uncorrected Errors Location Register

Uncorrected errors locations. For more details, see the RAS Data Protection (DP) section in the Core Operations chapter of the user manual.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LOC_LAST_UNCORR_ERROR								REG_LAST_UNCORR_ERROR				RSVDP_16			
rh								rh				r			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LOC_FIRST_UNCORR_ERROR								REG_FIRST_UNCORR_ERROR				RSVDP_0			
rh								rh				r			

Field	Bits	Type	Description
RSVDP_0	3:0	r	**Reserved for future use**

(table continues...)

(continued)

Field	Bits	Type	Description
REG_FIRST_UNCORR_ERROR	7:4	rh	**Region of the first uncorrected error** All encodings other than the defined encodings are reserved. Note: This register field is sticky. 0_H **ADM_RX_PATH**: Region select for Adm Rx path 1_H **L3_RX_PATH**: Region select for layer 3 Rx path 2_H **L2_RX_PATH**: Region select for layer 2 Rx path 3_H **DMA_INBOUND_PATH**: Region select for DMA inbound path 4_H **AXI_INBOUND_REQ_PATH**: Region select for AXI bridge inbound request path 5_H **AXI_INBOUND_CMPL_PATH**: Region select for AXI bridge inbound completion composer path 6_H **ADM_TX_PATH**: Region select for Adm Tx path 7_H **L3_TX_PATH**: Region select for layer 3 Tx path 8_H **L2_TX_PATH**: Region select for layer 2 Tx path 9_H **DMA_OUTBOUND_PATH**: Region select for DMA outbound path A_H **AXI_OUTBOUND_REQ_PATH**: Region select for AXI bridge outbound request path B_H **AXI_OUTBOUND_CMPL_PATH**: Region select for AXI bridge outbound master completion buffer path C_H **CXS_TX_RX_PATH**: Region select for CXS Tx/Rx path D_H **DTIM_TX_RX_PATH**: Region select for DTIM Tx/Rx path **others,** Reserved
LOC_FIRST_UNCORR_ERROR	15:8	rh	**Location/ID of the first uncorrected error within the region defined by REG_FIRST_UNCORR_ERROR** You can cycle this field value from 0 to 255 to access all counters according to the detailed report of check points at http://www.synopsys.com/dw/doc.php/iip/DWC_pcie/latest/doc/RASDP_CheckPoints.pdf Note: This register field is sticky.
RSVDP_16	19:16	r	**Reserved for future use**

(table continues...)

(continued)

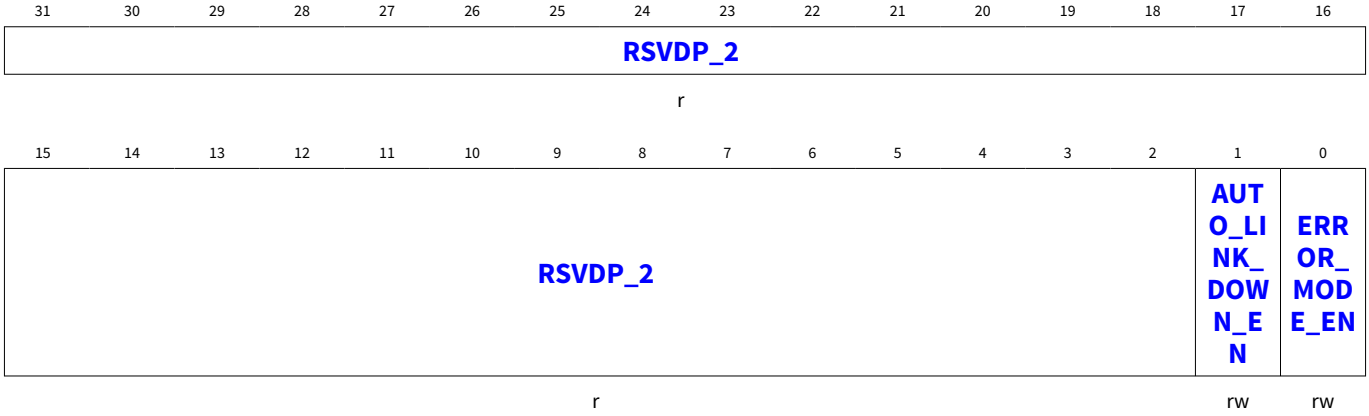
Field	Bits	Type	Description
REG_LAST_UNCORR_ERROR	23:20	rh	**Region of the last uncorrected error** All encodings other than the defined encodings are reserved. Note: This register field is sticky. 0_H **ADM_RX_PATH**: Region select for Adm Rx path 1_H **L3_RX_PATH**: Region select for layer 3 Rx path 2_H **L2_RX_PATH**: Region select for layer 2 Rx path 3_H **DMA_INBOUND_PATH**: Region select for DMA inbound path 4_H **AXI_INBOUND_REQ_PATH**: Region select for AXI bridge inbound request path 5_H **AXI_INBOUND_CMPL_PATH**: Region select for AXI bridge inbound completion composer path 6_H **ADM_TX_PATH**: Region select for Adm Tx path 7_H **L3_TX_PATH**: Region select for layer 3 Tx path 8_H **L2_TX_PATH**: Region select for layer 2 Tx path 9_H **DMA_OUTBOUND_PATH**: Region select for DMA outbound path A_H **AXI_OUTBOUND_REQ_PATH**: Region select for AXI bridge outbound request path B_H **AXI_OUTBOUND_CMPL_PATH**: Region select for AXI bridge outbound master completion buffer path C_H **CXS_TX_RX_PATH**: Region select for CXS Tx/Rx path D_H **DTIM_TX_RX_PATH**: Region select for DTIM Tx/Rx path **others,** Reserved
LOC_LAST_UNCORR_ERROR	31:24	rh	**Location/ID of the last uncorrected error within the region defined by REG_LAST_UNCORR_ERROR** You can cycle this field value from 0 to 255 to access all counters according to the detailed report of check points at http://www.synopsys.com/dw/doc.php/iip/DWC_pcie/latest/doc/RASDP_CheckPoints.pdf Note: This register field is sticky.

19.4.1.192 RASDP error mode enable.

RASDP error mode enable. The controller enters RASDP error mode (if ERROR_MODE_EN =1) upon detection of the first uncorrectable error. During this mode: - Rx TLPs that are forwarded to your application are not guaranteed to be correct; you must discard them. For more details, see the RAS Data Protection (DP) section in the Core Operations chapter of the user manual.

RASDP_ERROR_MODE_EN_OFF Offset address: $00042A8_H$

RASDP error mode enable. Kernel Reset value: $0000\ 0001_H$



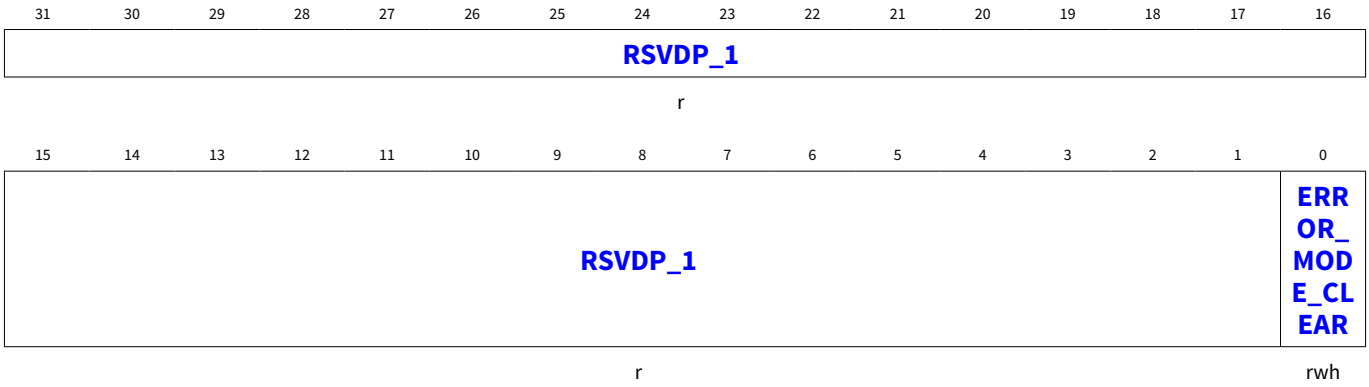
Field	Bits	Type	Description
ERROR_MODE_EN	0	rw	**Write '1' to enable the controller enter RASDP error mode when it detects an uncorrectable error** Note: This register field is sticky.
AUTO_LINK_DOWN_EN	1	rw	**Write '1' to enable the controller to bring the link down when the controller enters RASDP error mode** Note: This register field is sticky.
RSVDP_2	31:2	r	**Reserved for future use**

19.4.1.193 RASDP Error Mode Clear Register

Exit RASDP error mode. For more details, see the RAS Data Protection (DP) section in the Core Operations chapter of the user manual.

RASDP_ERROR_MODE_CLEAR_OFF Offset address: $00042AC_H$

RASDP Error Mode Clear Register Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
ERROR_MODE_CLEAR	0	rwh	**Write '1' to take the controller out of RASDP error mode** The controller will then report uncorrectable errors (through AER internal error reporting) and also stop nullifying/discarding TLPs. Reading register field always returns a '0'.
RSVDP_1	31:1	r	**Reserved for future use**

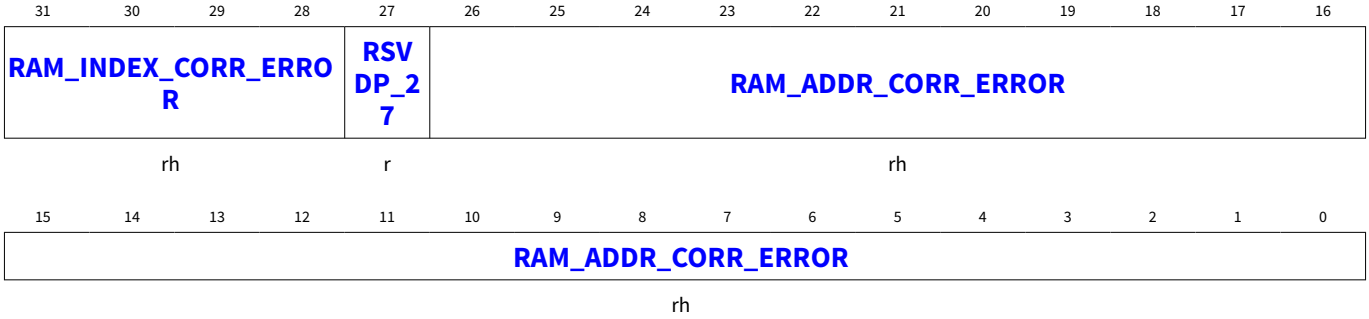
19.4.1.194 RAM Address where a corrected error (1-bit ECC) is detected

For more details, see the RAS Data Protection (DP) section in the Core Operations chapter of the user manual.

RASDP_RAM_ADDR_CORR_ERROR_OFF Offset address: $00042B0_H$

RAM Address where a corrected error [1-bit ECC) is detected

Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
RAM_ADDR_CORR_ERROR	26:0	rh	**RAM Address where a corrected error (1-bit ECC) is detected** Note: This register field is sticky.
RSVDP_27	27	r	**Reserved for future use**
RAM_INDEX_CORR_ERROR	31:28	rh	**RAM index where a corrected error (1-bit ECC) is detected** Note: This register field is sticky.

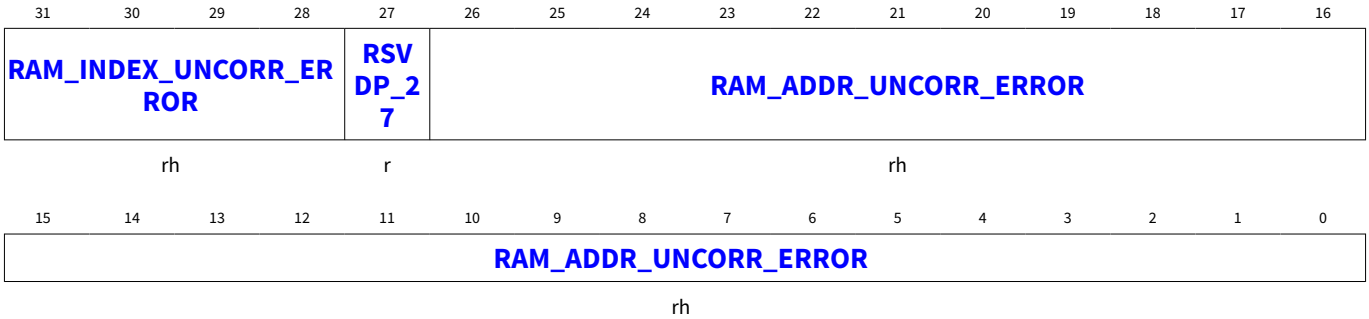
19.4.1.195 RAM Address where an uncorrected error (2-bit ECC) is detected

For more details, see the RAS Data Protection (DP) section in the Core Operations chapter of the user manual.

RASDP_RAM_ADDR_UNCORR_ERROR_OFF Offset address: $00042B4_H$

RAM Address where an uncorrected error [2-bit ECC) is detected

Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
RAM_ADDR_UNCORR_ERROR	26:0	rh	**RAM Address where an uncorrected error (2-bit ECC) is detected** Note: This register field is sticky.
RSVDP_27	27	r	**Reserved for future use**

(table continues...)

(continued)

Field	Bits	Type	Description
RAM_INDEX_UNCORR_ERROR	31:28	rh	**RAM index where an uncorrected error (2-bit ECC) is detected** Note: This register field is sticky.

19.4.1.196 Precision Time Measurement Capability Header

This Register provides Capability ID, Cabability Version, and Next Offset of Precision Time Measurement Capability structure.

PTM_EXT_CAP_HDR_OFF Offset address: $00042B8_H$

Precision Time Measurement Capability Header Kernel Reset value: $2C41\ 001F_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PTM_NEXT_OFFSET												PTM_CAP_VERSION			
rw												rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PTM_CAP_ID															
rw															

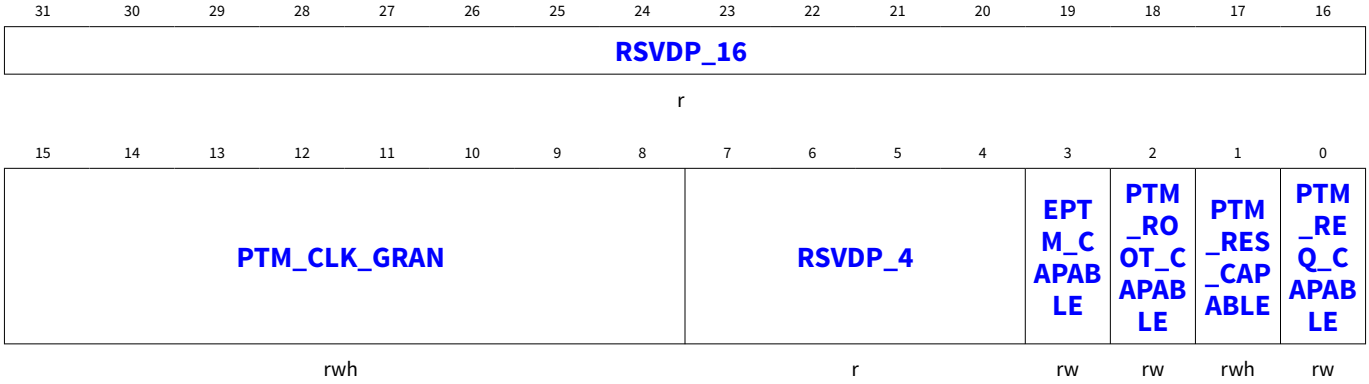
Field	Bits	Type	Description
PTM_CAP_ID	15:0	rw	**Precision Time Measurement PCI Express Extended Capability ID** This field is a PCI-SIG defined ID number that indicates the nature and format of the Extended Capability. PCI Express Extended Capability ID for the Precision Time Measurement Capability is 001Fh. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky. 0000_H **MIN_VAL**: Min value $FFFF_H$ **MAX_VAL**: Max value
PTM_CAP_VERSION	19:16	rw	**Precision Time Measurement PCI Express Extended Capability Version** This field is a PCI-SIG defined version number that indicates the version of the Capability structure present. This bit depends on the version of the specification. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky. 0_H **MIN_VAL**: Min value F_H **MAX_VAL**: Max value
PTM_NEXT_OFFSET	31:20	rw	**Precision Time Measurement PCI Express Extended Capability Next Offset** This field contains the offset to the next PCI Express Extended Capability structure or 000h if no other items exist in the linked list of Capabilities. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky. 000_H **MIN_VAL**: Min value FFF_H **MAX_VAL**: Max value

19.4.1.197 PTM Capability Register

This register describes a Function's support for Precision Time Measurement. Not all fields within this register apply to all Functions capable of implementing PTM.

PTM_CAP_OFF Offset address: $00042BC_H$

PTM Capability Register Kernel Reset value: $0000\ 040F_H$



Field	Bits	Type	Description
PTM_REQ_CAPABLE	0	rw	**PTM Requester Capable** Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky. 0_B **CLEAR**: Clear 1_B **SET**: Endpoints and RCiEPs are permitted to, and Switches supporting PTM must, set this bit to 1b to indicate they implement the PTM Requester role
PTM_RES_CAPABLE	1	rwh	**PTM Responder Capable** Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky. 0_B **CLEAR**: Clear 1_B **SET**: Root Ports and RCRB's are permitted to, and Switches supporting PTM must, set this bit to 1b to indicate they implement the PTM Responder role If PTM Root Capable is Set, this bit must be Set to 1b.
PTM_ROOT_CAPABLE	2	rw	**PTM Root Capable** Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky. 0_B **CLEAR**: Clear 1_B **SET**: Root Ports, RCRBs and Switches are permitted to set this bit to 1b, if they implement a PTM Time Source Role and are capable of serving as the PTM Root

(table continues...)

(continued)

Field	Bits	Type	Description
EPTM_CAPABLE	3	rw	**ePTM Capable** Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky. 0_B **CLEAR**: Clear 1_B **SET**: Indicates Enhanced Precision Time Management (ePTM) is supported Defaults to 0x1.
RSVDP_4	7:4	r	**Reserved for future use**
PTM_CLK_GRAN	15:8	rwh	**PTM Local Clock Granularity** For other than this value (between b00000001-b111111110) indicates the period of this Time Source's local clock in ns. If the PTM Root Select bit is Set, this local clock is used to provide PTM Master Time. Otherwise, the Time Source uses this local clock to locally track PTM Master Time received from further Upstream within a PTM Hierarchy. This field is reserved for Functions that do not implement the PTM Time Source role. Note: The access attributes of this field are as follows: - Wire: RSVDP - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) 00_H **No_local_clock**: Time Source does not implement a local clock It simply propagates timing information obtained from further Upstream in the PTM Hierarchy when responding to PTM Request messages. FF_H **Greater_than_254ns**: Indicates the period of this Time Source's local clock is greater than 254 ns
RSVDP_16	31:16	r	**Reserved for future use**

19.4.1.198 PTM Control Register

This register controls a Function's participation in the Precision Time Measurement mechanism. Not all fields within this register apply to all Functions capable of implementing PTM.

PTM_CONTROL_OFF Offset address: $00042C0_H$

PTM Control Register Kernel Reset value: $0000\ 0000_H$

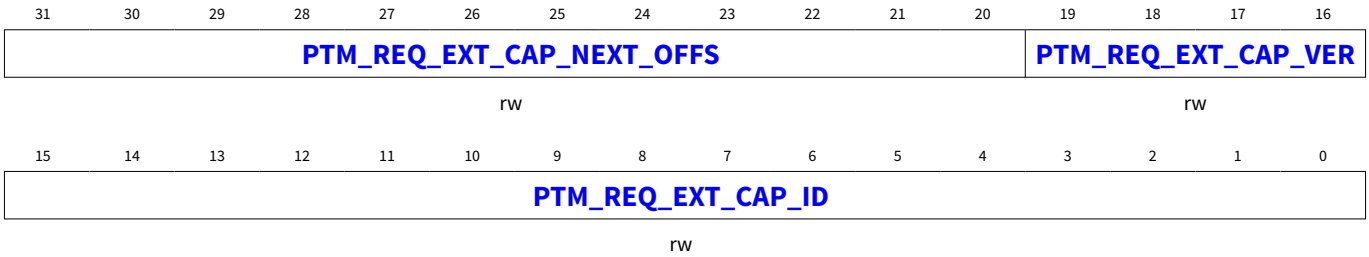
Field	Bits	Type	Description
PTM_ENABLE	0	rw	**PTM Enable** 0_B **CLEAR**: Clear 1_B **SET**: When Set, this Function is permitted to participate in the PTM mechanism according to its selected role(s)
ROOT_SELECT	1	rw	**PTM Root Select** If the value of the PTM Root Capable bit is 0b, this bit is permitted to be hardwired to 0b by the controller. Within each PTM Hierarchy, it is recommended that system software select only the furthest Upstream Time Source to be the PTM Root. Note: The access attributes of this field are as follows: - Wire: HWINIT - Dbi: HWINIT 0_B **CLEAR**: Clear 1_B **SET**: When Set, if the PTM Enable bit is also Set, this Time Source is the PTM Root
RSVDP_2	7:2	r	**Reserved for future use**
EFF_GRAN	15:8	rh	**PTM Effective Granularity** For Functions implementing the PTM Requester Role, this field provides information relating to the expected accuracy of the PTM clock, but does not otherwise affect the PTM mechanism. For Endpoints, system software must program this field to the value representing the maximum Local Clock Granularity reported by the PTM Root and all intervening PTM Time Sources. For RCiEPs, system software must set this field to the value reported in the Local Clock Granularity field by the associated PTM Time Source. For other than this value (between b00000001-b111111110) indicates the effective PTM granularity in ns. Note: The access attributes of this field are as follows: - Wire: HWINIT - Dbi: HWINIT 00_H **No_local_clock**: Unknown PTM granularity - one or more Switches between this Function and the PTM Root reported a Local Clock Granularity value of 0000 0000b FF_H **Greater_than_254ns**: Indicates the effective PTM granularity is greater than 254 ns
RSVDP_16	31:16	r	**Reserved for future use**

19.4.1.199 Precision Time Measurement Requester Capability Header (VSEC)

This Register provides Capability ID, Cabability Version, and Next Offset of Precision Time Measurement Capability structure.

PTM_REQ_CAP_HDR_OFF — Offset address: $00042C4_H$

Precision Time Measurement Requester Capability Header [VSEC) — Kernel Reset value: $32C1\ 000B_H$



Field	Bits	Type	Description
PTM_REQ_EXT_CAP_ID	15:0	rw	**Precision Time Measurement Requester VSEC ID** For more details, see the PTM section in the user manual. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky. 0000_H **MIN_VAL**: Min value $FFFF_H$ **MAX_VAL**: Max value
PTM_REQ_EXT_CAP_VER	19:16	rw	**Precision Time Measurement Requester VSEC Version** For more details, see the PTM section in the user manual. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky. 0_H **MIN_VAL**: Min value F_H **MAX_VAL**: Max value
PTM_REQ_EXT_CAP_NEXT_OFFS	31:20	rw	**Precision Time Measurement Requester VSEC Next Pointer** For more details, see the PTM section in the user manual. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky. 000_H **MIN_VAL**: Min value FFF_H **MAX_VAL**: Max value

19.4.1.200 Precision Time Measurement Requester Vendor Specific Header

For more details, see the PTM section in the user manual.

PTM_REQ_HDR_OFF — Offset address: $00042C8_H$

Precision Time Measurement Requester Vendor Specific Header — Kernel Reset value: $0541\ 0003_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PTM_REQ_VSEC_LENGTH												PTM_REQ_VSEC_REV			
rw												rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PTM_REQ_VSEC_ID															
rw															

Field	Bits	Type	Description
PTM_REQ_VSEC_ID	15:0	rw	**PTM Requester VSEC ID** For more details, see the PTM section in the user manual. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky. 0000_H **MIN_VAL**: Min value $FFFF_H$ **MAX_VAL**: Max value

(table continues...)

(continued)

Field	Bits	Type	Description
PTM_REQ_VSEC_REV	19:16	rw	**PTM Requester VSEC Revision** For more details, see the PTM section in the user manual. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky. 0_H **MIN_VAL**: Min value F_H **MAX_VAL**: Max value
PTM_REQ_VSEC_LENGTH	31:20	rw	**PTM Requester VSEC Length** For more details, see the PTM section in the user manual. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky. 000_H **MIN_VAL**: Min value FFF_H **MAX_VAL**: Max value

19.4.1.201 PTM Requester Vendor Specific Control Register

For more details, see the PTM section in the user manual.

PTM_REQ_CONTROL_OFF Offset address: $00042CC_H$

PTM Requester Vendor Specific Control Register Kernel Reset value: $0000\ 0900_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_17															PTM_REQ_PDEL_BYTE_REV
r															rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PTM_REQ_LONG_TIMER								RSVDP_3					PTM_REQ_FAST_TIMERS	PTM_REQ_START_UPDATE	PTM_REQ_AUTO_UPDATE_ENABLED
rw								r					rw	rw1ch	rw

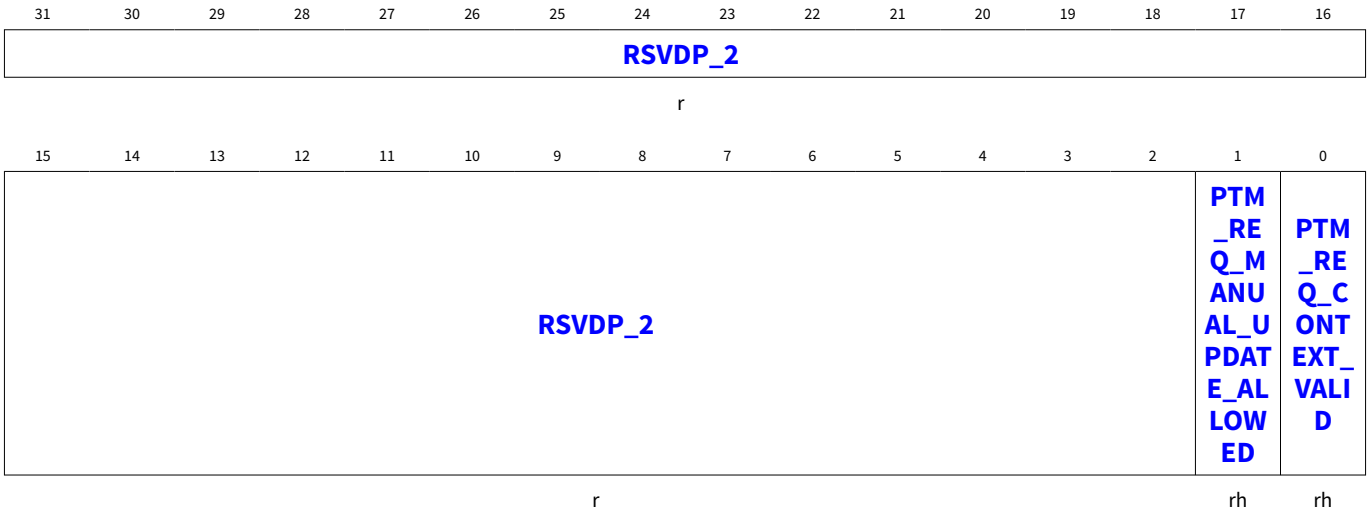
Field	Bits	Type	Description
PTM_REQ_AUTO_UPDATE_ENABLED	0	rw	**PTM Requester Auto Update Enabled** For more details, see the PTM section in the user manual. Note: This register field is sticky. 0_B **DISABLE**: Disable 1_B **ENABLE**: When enabled PTM Requester will automatically atempt to update its context every 10ms
PTM_REQ_START_UPDATE	1	rw1ch	**PTM Requester Start Update** For more details, see the PTM section in the user manual. Software write with 1 clears this bit-field. 0_B **CLEAR**: Clear 1_B **SET**: When set, the PTM Requester will attempt a PTM Dialogue to update its context; This bit is self clearing
PTM_REQ_FAST_TIMERS	2	rw	**PTM Fast Timers (Debug mode for PTM Timers)** The 100 µs timer output will go high at 30 µs and the 10 ms timer output will go high at 100 µs (the Long Timer Value is ignored). There is no change to the 1 µs timer. The requester operation will otherwise remain the same. For more details, see the PTM section in the user manual. Note: This register field is sticky.
RSVDP_3	7:3	r	**Reserved for future use**
PTM_REQ_LONG_TIMER	15:8	rw	**PTM Requester Long Timer** Determines the period between each auto update PTM Dialogue in miliseconds. Update period is the register value +1 milisecond. For the Switch product this value must not be set larger than 0x9 for spec compliance. For more details, see the PTM section in the user manual. Note: This register field is sticky.
PTM_REQ_PDEL_BYTE_REV	16	rw	**PTM Requester Propagation Delay Byte Reversal** Reverse the order of bytes in the PTM Propagation Delay data word of the PTM ResponseD Message, for compatibility with previous revisions. For more details, see the PTM section in the user manual. Note: This register field is sticky.
RSVDP_17	31:17	r	**Reserved for future use**

19.4.1.202 PTM Requester Vendor Specific Status Register

For more details, see the PTM section in the user manual.

PTM_REQ_STATUS_OFF Offset address: $00042D0_H$

PTM Requester Vendor Specific Status Register Kernel Reset value: $0000\ 0000_H$



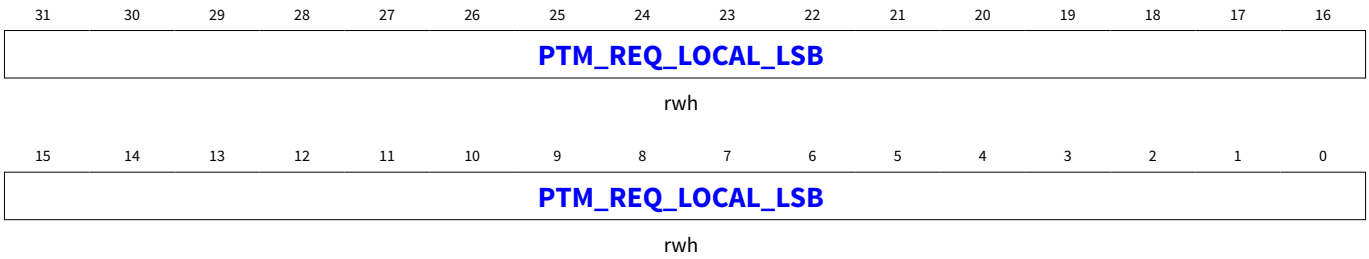
Field	Bits	Type	Description
PTM_REQ_CONTEXT_VALID	0	rh	**PTM Requester Context Valid** Indicate that the Timing Context is valid. For more details, see the PTM section in the user manual.
PTM_REQ_MANUAL_UPDATE_ALLOWED	1	rh	**PTM Requester Manual Update Allowed** Indicates whether or not a Manual Update can be signalled. For more details, see the PTM section in the user manual.
RSVDP_2	31:2	r	**Reserved for future use**

19.4.1.203 PTM Requester Local Clock LSB

For more details, see the PTM section in the user manual.

PTM_REQ_LOCAL_LSB_OFF Offset address: $00042D4_H$

PTM Requester Local Clock LSB Kernel Reset value: $0000\ 0000_H$



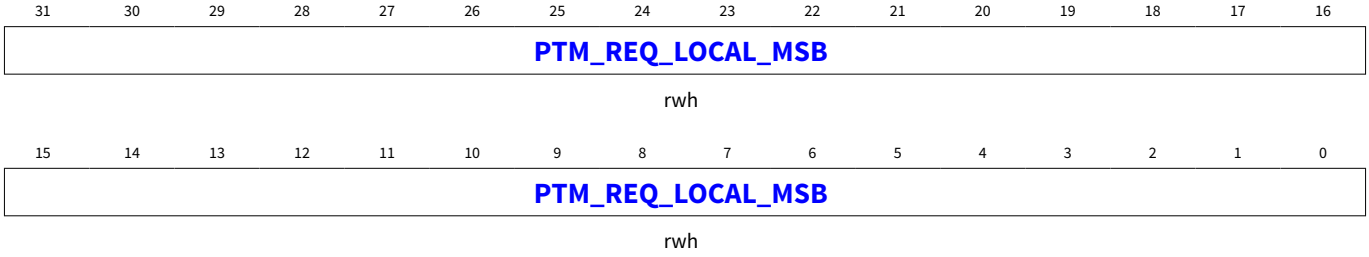
Field	Bits	Type	Description
PTM_REQ_LOCAL_LSB	31:0	rwh	**PTM Requester Local Clock LSB** Lower 32 bits of local timer value. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky. 00000000_H **MIN_VAL**: Min value $FFFFFFFF_H$ **MAX_VAL**: Max value

19.4.1.204 PTM Requester Local Clock MSB

For more details, see the PTM section in the user manual.

PTM_REQ_LOCAL_MSB_OFF Offset address: $00042D8_H$

PTM Requester Local Clock MSB Kernel Reset value: $0000\ 0000_H$



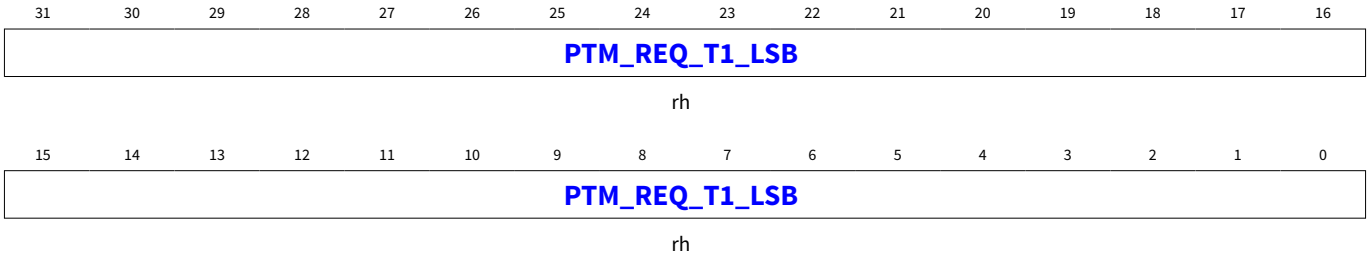
Field	Bits	Type	Description
PTM_REQ_LOCAL_MSB	31:0	rwh	**PTM Requester Local Clock MSB** Upper 32 bits of local timer value. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky. 00000000_H **MIN_VAL**: Min value $FFFFFFFF_H$ **MAX_VAL**: Max value

19.4.1.205 PTM Requester T1 Timestamp LSB

For more details, see the PTM section in the user manual.

PTM_REQ_T1_LSB_OFF Offset address: $00042DC_H$

PTM Requester T1 Timestamp LSB Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
PTM_REQ_T1_LSB	31:0	rh	**PTM Requester T1 Timestamp LSB** Lower 32 bits of the T1 Timestamp value. Note: This register field is sticky. 00000000_H **MIN_VAL**: Min value $FFFFFFFF_H$ **MAX_VAL**: Max value

19.4.1.206 PTM Requester T1 Timestamp MSB

For more details, see the PTM section in the user manual.

PTM_REQ_T1_MSB_OFF Offset address: $00042E0_H$

PTM Requester T1 Timestamp MSB Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PTM_REQ_T1_MSB															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PTM_REQ_T1_MSB															
rh															

Field	Bits	Type	Description
PTM_REQ_T1_MSB	31:0	rh	**PTM Requester T1 Timestamp MSB** Upper 32 bits of the T1 Timestamp value. Note: This register field is sticky. 00000000_H **MIN_VAL**: Min value $FFFFFFFF_H$ **MAX_VAL**: Max value

19.4.1.207 PTM Requester T1 Previous Timestamp LSB

For more details, see the PTM section in the user manual.

PTM_REQ_T1P_LSB_OFF Offset address: $00042E4_H$

PTM Requester T1 Previous Timestamp LSB Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PTM_REQ_T1P_LSB															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PTM_REQ_T1P_LSB															
rh															

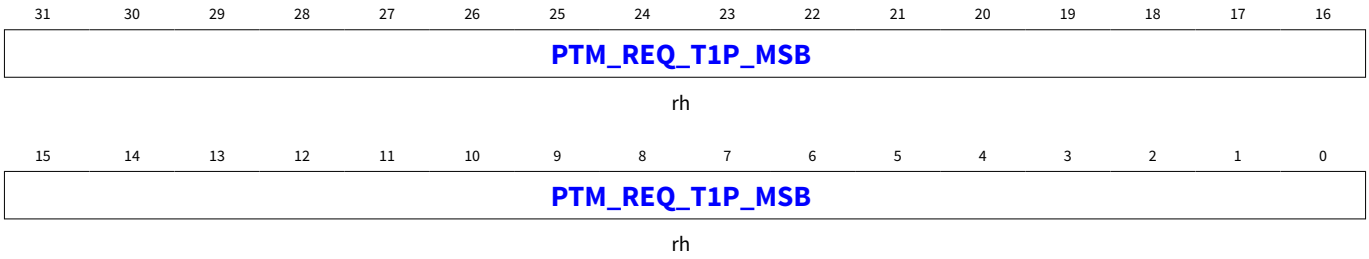
Field	Bits	Type	Description
PTM_REQ_T1P_LSB	31:0	rh	**PTM Requester T1 Previous Timestamp LSB** Lower 32 bits of the previously stored T1 Timestamp value. Note: This register field is sticky. 00000000_H **MIN_VAL**: Min value $FFFFFFFF_H$ **MAX_VAL**: Max value

19.4.1.208 PTM Requester T1 Previous Timestamp MSB

For more details, see the PTM section in the user manual.

PTM_REQ_T1P_MSB_OFF Offset address: $00042E8_H$

PTM Requester T1 Previous Timestamp MSB Kernel Reset value: $0000\ 0000_H$



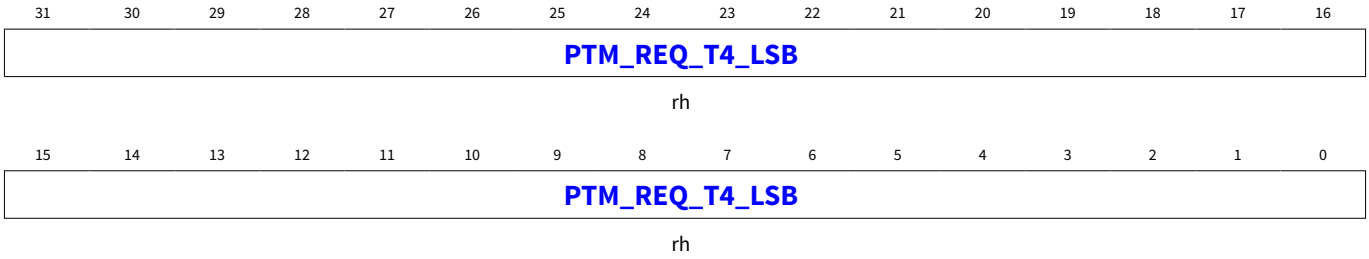
Field	Bits	Type	Description
PTM_REQ_T1P_MSB	31:0	rh	**PTM Requester T1 Previous Timestamp MSB** Upper 32 bits of the previously stored T1 Timestamp value. Note: This register field is sticky. 00000000_H **MIN_VAL**: Min value $FFFFFFFF_H$ **MAX_VAL**: Max value

19.4.1.209 PTM Requester T4 Timestamp LSB

For more details, see the PTM section in the user manual.

PTM_REQ_T4_LSB_OFF Offset address: $00042EC_H$

PTM Requester T4 Timestamp LSB Kernel Reset value: $0000\ 0000_H$



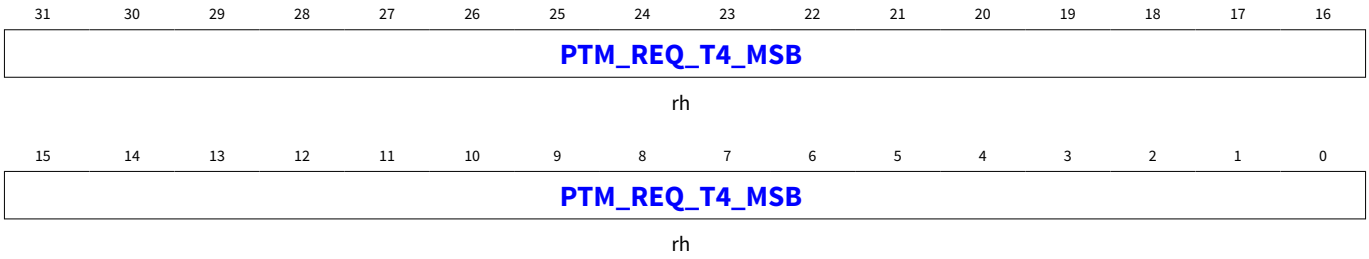
Field	Bits	Type	Description
PTM_REQ_T4_LSB	31:0	rh	**PTM Requester T4 Timestamp LSB** Lower 32 bits of the T4 Timestamp value. Note: This register field is sticky. 00000000_H **MIN_VAL**: Min value $FFFFFFFF_H$ **MAX_VAL**: Max value

19.4.1.210 PTM Requester T4 Timestamp MSB

For more details, see the PTM section in the user manual.

PTM_REQ_T4_MSB_OFF Offset address: $00042F0_H$

PTM Requester T4 Timestamp MSB Kernel Reset value: $0000\ 0000_H$



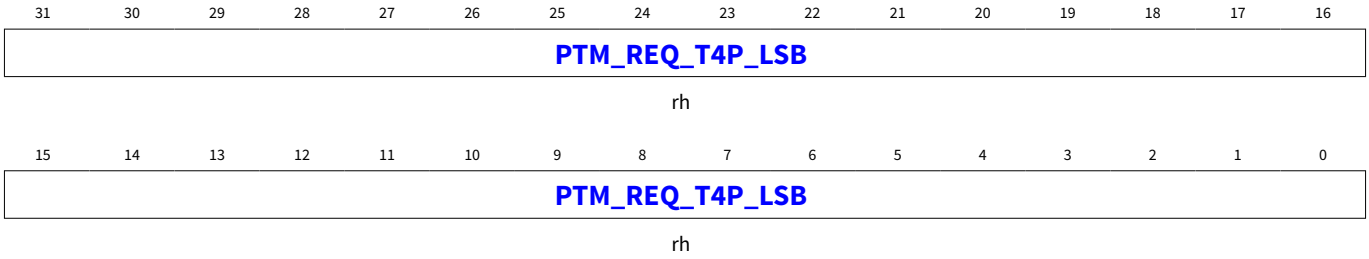
Field	Bits	Type	Description
PTM_REQ_T4_MSB	31:0	rh	**PTM Requester T4 Timestamp MSB** Upper 32 bits of the T4 Timestamp value. Note: This register field is sticky. 00000000_H **MIN_VAL**: Min value $FFFFFFFF_H$ **MAX_VAL**: Max value

19.4.1.211 PTM Requester T4 Previous Timestamp LSB

For more details, see the PTM section in the user manual.

PTM_REQ_T4P_LSB_OFF Offset address: $00042F4_H$

PTM Requester T4 Previous Timestamp LSB Kernel Reset value: $0000\ 0000_H$



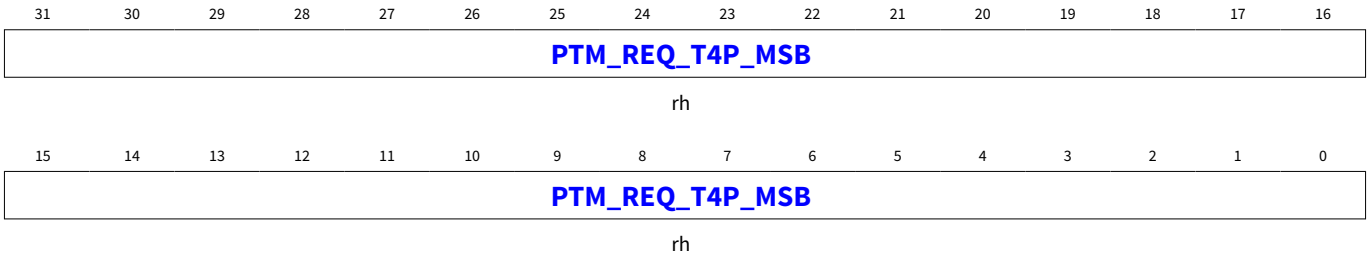
Field	Bits	Type	Description
PTM_REQ_T4P_LSB	31:0	rh	**PTM Requester T4 Previous Timestamp LSB** Lower 32 bits of the previously stored T4 Timestamp value. Note: This register field is sticky. 00000000_H **MIN_VAL**: Min value $FFFFFFFF_H$ **MAX_VAL**: Max value

19.4.1.212 PTM Requester T4 Previous Timestamp MSB

For more details, see the PTM section in the user manual.

PTM_REQ_T4P_MSB_OFF Offset address: $00042F8_H$

PTM Requester T4 Previous Timestamp MSB Kernel Reset value: $0000\ 0000_H$



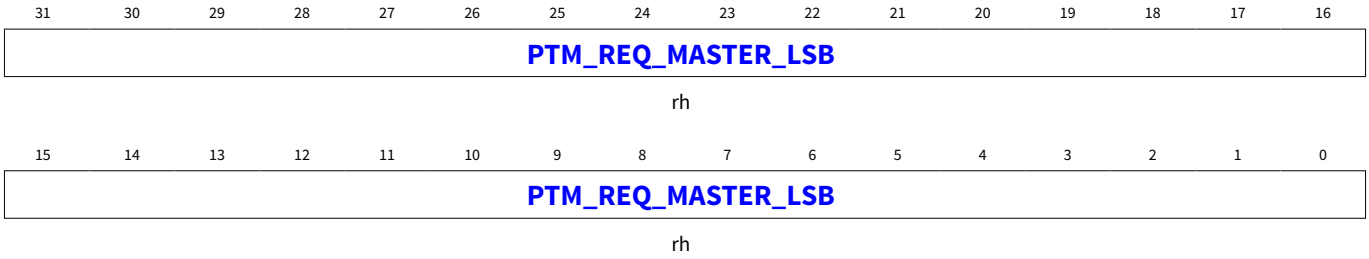
Field	Bits	Type	Description
PTM_REQ_T4P_MSB	31:0	rh	**PTM Requester T4 Previous Timestamp MSB** Upper 32 bits of the previously stored T4 Timestamp value. Note: This register field is sticky. 00000000_H **MIN_VAL**: Min value $FFFFFFFF_H$ **MAX_VAL**: Max value

19.4.1.213 PTM Requester Master Time LSB

For more details, see the PTM section in the user manual.

PTM_REQ_MASTER_LSB_OFF Offset address: $00042FC_H$

PTM Requester Master Time LSB Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
PTM_REQ_MASTER_LSB	31:0	rh	**PTM Requester Master Time LSB** Lower 32 bits of Master Time value sent from PTM Responder. Note: This register field is sticky. 00000000_H **MIN_VAL**: Min value $FFFFFFFF_H$ **MAX_VAL**: Max value

19.4.1.214 PTM Requester Master Time MSB

For more details, see the PTM section in the user manual.

PTM_REQ_MASTER_MSB_OFF Offset address: 0004300_H

PTM Requester Master Time MSB Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PTM_REQ_MASTER_MSB															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PTM_REQ_MASTER_MSB															
rh															

Field	Bits	Type	Description
PTM_REQ_MASTER_MSB	31:0	rh	**PTM Requester Master Time MSB** Upper 32 bits of Master Time value sent from PTM Responder. Note: This register field is sticky. 00000000_H **MIN_VAL**: Min value $FFFFFFFF_H$ **MAX_VAL**: Max value

19.4.1.215 PTM Requester Propagation Delay

For more details, see the PTM section in the user manual.

PTM_REQ_PROP_DELAY_OFF Offset address: 0004304_H

PTM Requester Propagation Delay Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PTM_REQ_PROP_DELAY															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PTM_REQ_PROP_DELAY															
rh															

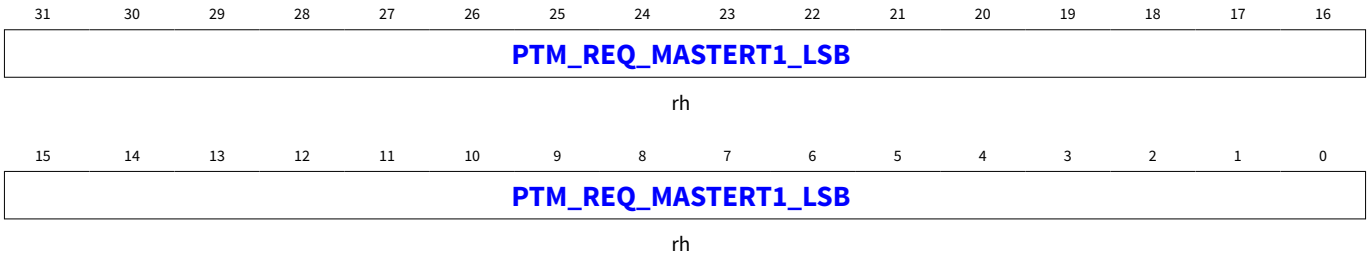
Field	Bits	Type	Description
PTM_REQ_PROP_DELAY	31:0	rh	**PTM Requester Propagation Delay** Propagation Delay sent from the PTM Responder. Note: This register field is sticky. 00000000_H **MIN_VAL**: Min value $FFFFFFFF_H$ **MAX_VAL**: Max value

19.4.1.216 PTM Requester Master Time at T1 LSB

For more details, see the PTM section in the user manual.

PTM_REQ_MASTERT1_LSB_OFF Offset address: 0004308_H

PTM Requester Master Time at T1 LSB Kernel Reset value: $0000\ 0000_H$



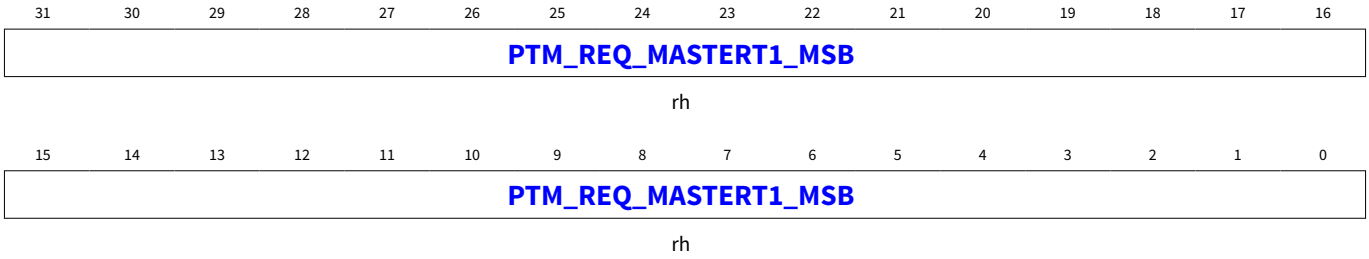
Field	Bits	Type	Description
PTM_REQ_MASTERT1_LSB	31:0	rh	**PTM Requester Master Time at T1 LSB** Lower 32 bits of the calculated Master Time at T1 prime. Note: This register field is sticky. 00000000_H **MIN_VAL**: Min value $FFFFFFFF_H$ **MAX_VAL**: Max value

19.4.1.217 PTM Requester Master Time at T1 MSB

For more details, see the PTM section in the user manual.

PTM_REQ_MASTERT1_MSB_OFF Offset address: $000430C_H$

PTM Requester Master Time at T1 MSB Kernel Reset value: $0000\ 0000_H$



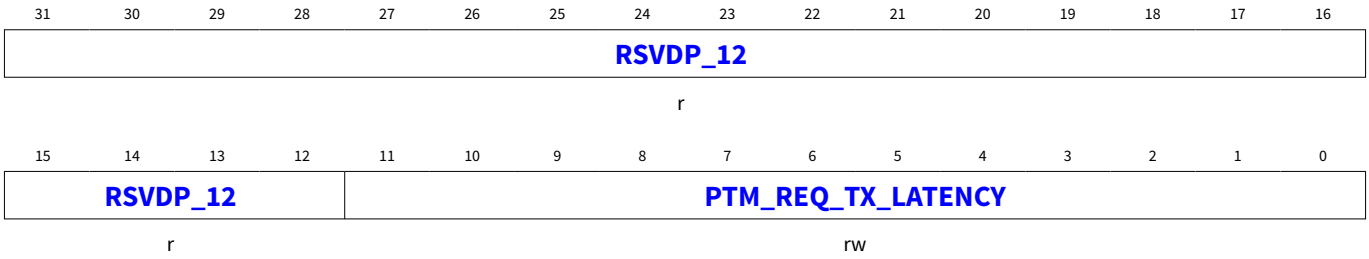
Field	Bits	Type	Description
PTM_REQ_MASTERT1_MSB	31:0	rh	**PTM Requester Master Time at T1 MSB** Upper 32 bits of the calculated Master Time at T1 prime. Note: This register field is sticky. 00000000_H **MIN_VAL**: Min value $FFFFFFFF_H$ **MAX_VAL**: Max value

19.4.1.218 PTM Requester TX Latency

For more details, see the PTM section in the user manual.

PTM_REQ_TX_LATENCY_OFF Offset address: 0004310_H

PTM Requester TX Latency Kernel Reset value: $0000\ 000E_H$



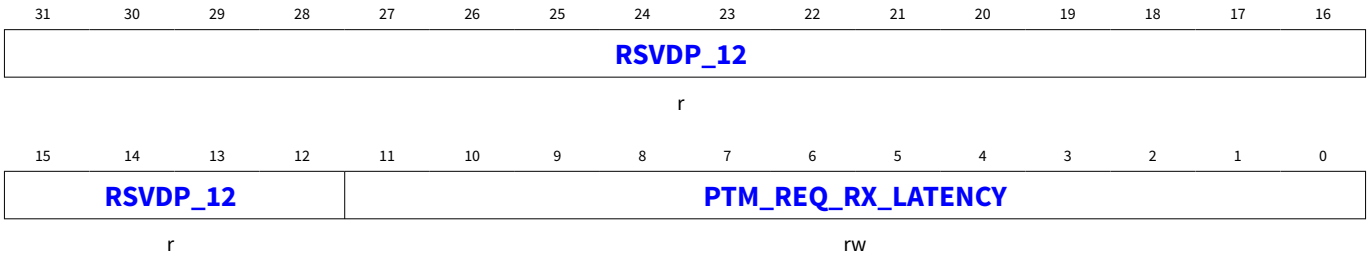
Field	Bits	Type	Description
PTM_REQ_TX_LATENCY	11:0	rw	**PTM Requester TX Latency** Requester Transmit path latency viewport register. A register is provided for each supported link speed, and the value used for timestamp adjustment is automatically selected based on the current speed. The PTM Requester Latency Register Select register specifies the register to be written or read. For more details, see the PTM section in the user manual. Note: This register field is sticky. 000_H **MIN_VAL**: Min value FFF_H **MAX_VAL**: Max value
RSVDP_12	31:12	r	**Reserved for future use**

19.4.1.219 PTM Requester RX Latency

For more details, see the PTM section in the user manual.

PTM_REQ_RX_LATENCY_OFF Offset address: 0004314_H

PTM Requester RX Latency Kernel Reset value: $0000\ 002A_H$



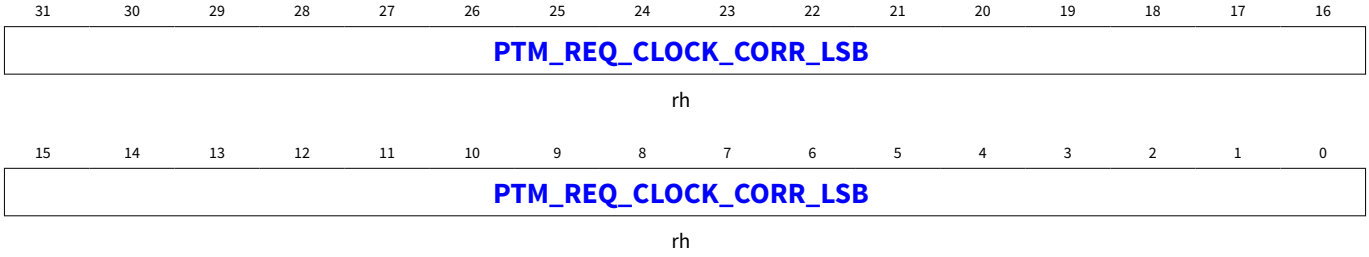
Field	Bits	Type	Description
PTM_REQ_RX_LATENCY	11:0	rw	**PTM Requester RX Latency - Requester Receive path latency viewport register** A register is provided for each supported link speed, and the value used for timestamp adjustment is automatically selected based on the current speed. The PTM Requester Latency Register Select register specifies the register to be written or read. For more details, see the PTM section in the user manual. Note: This register field is sticky. 000_H **MIN_VAL**: Min value FFF_H **MAX_VAL**: Max value
RSVDP_12	31:12	r	**Reserved for future use**

19.4.1.220 PTM Requester Clock Correction LSB

For more details, see the PTM section in the user manual.

PTM_REQ_CLOCK_CORR_LSB_OFF Offset address: 0004318_H

PTM Requester Clock Correction LSB Kernel Reset value: $0000\ 0000_H$



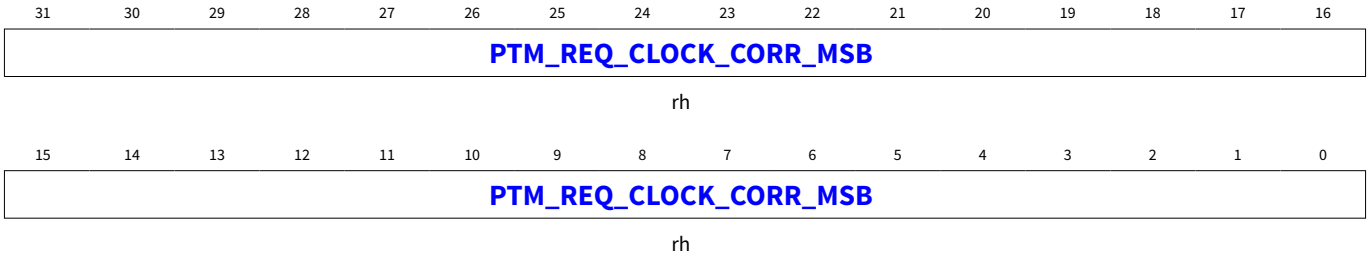
Field	Bits	Type	Description
PTM_REQ_CLOCK_CORR_LSB	31:0	rh	**PTM Requester Clock Correction LSB** Lower 32 bits of amount by which PTM Requester Local Clock has been corrected. For more details, see the PTM section in the user manual. Note: This register field is sticky. 00000000_H **MIN_VAL**: Min value $FFFFFFFF_H$ **MAX_VAL**: Max value

19.4.1.221 PTM Requester Clock Correction MSB

For more details, see the PTM section in the user manual.

PTM_REQ_CLOCK_CORR_MSB_OFF Offset address: $000431C_H$

PTM Requester Clock Correction MSB Kernel Reset value: $0000\ 0000_H$



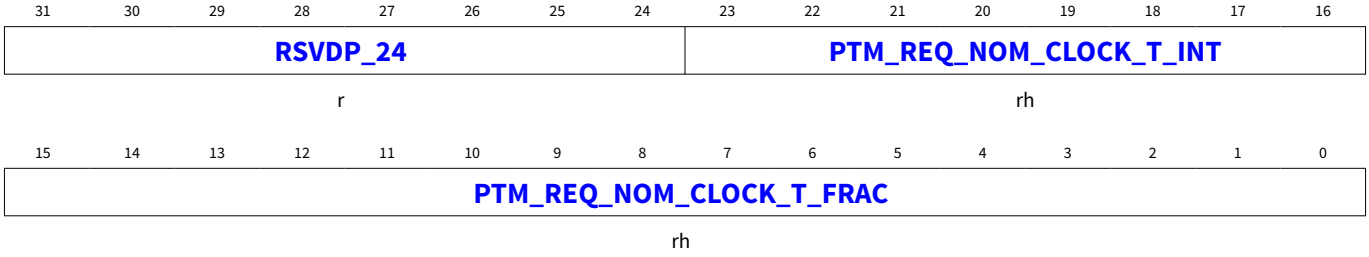
Field	Bits	Type	Description
PTM_REQ_CLOCK_CORR_MSB	31:0	rh	**PTM Requester Clock Correction MSB** Upper 32 bits of amount by which PTM Requester Local Clock has been corrected. For more details, see the PTM section in the user manual. Note: This register field is sticky. 00000000_H **MIN_VAL**: Min value $FFFFFFFF_H$ **MAX_VAL**: Max value

19.4.1.222 PTM Requester Nominal Clock Period

For more details, see the PTM section in the user manual.

PTM_REQ_NOM_CLOCK_T_OFF Offset address: 0004320_H

PTM Requester Nominal Clock Period Kernel Reset value: $0000\ 0000_H$



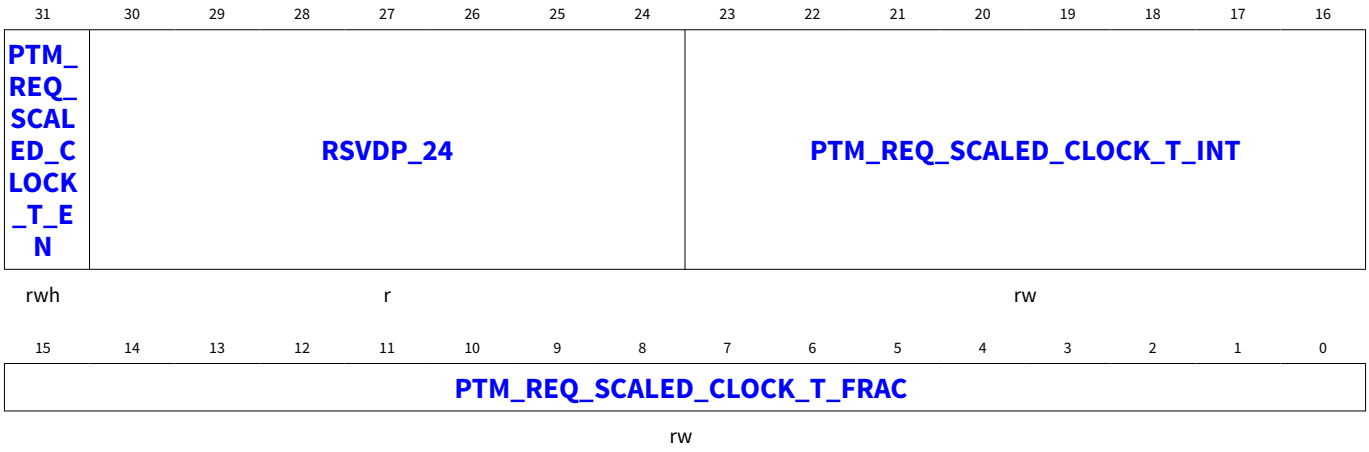
Field	Bits	Type	Description
PTM_REQ_NOM_CLOCK_T_FRAC	15:0	rh	**PTM Requester Nominal Clock Period Fractional** Fractional part of the nominal PTM local clock period, in nanoseconds. For more details, see the PTM section in the user manual. Note: This register field is sticky. 0000_H **MIN_VAL**: Min value $FFFF_H$ **MAX_VAL**: Max value
PTM_REQ_NOM_CLOCK_T_INT	23:16	rh	**PTM Requester Nominal Clock Period Integral** Integral part of the nominal PTM local clock period. LSB is 1/(2^16) nanoseconds. For more details, see the PTM section in the user manual. Note: This register field is sticky. 00_H **MIN_VAL**: Min value FF_H **MAX_VAL**: Max value
RSVDP_24	31:24	r	**Reserved for future use**

19.4.1.223 PTM Requester Scaled Clock Period

For more details, see the PTM section in the user manual.

PTM_REQ_SCALED_CLOCK_T_OFF Offset address: 0004324_H

PTM Requester Scaled Clock Period Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
PTM_REQ_SCALED_CLOCK_T_FRAC	15:0	rw	**PTM Requester Scaled Clock Period Fractional** Fractional part of the nominal PTM local clock period. LSB is 1/(2^16) nanoseconds. For more details, see the PTM section in the user manual. Note: This register field is sticky. 0000_H **MIN_VAL**: Min value $FFFF_H$ **MAX_VAL**: Max value
PTM_REQ_SCALED_CLOCK_T_INT	23:16	rw	**PTM Requester Scaled Clock Period Integral** Integral part of the nominal PTM local clock period, in nanoseconds. For more details, see the PTM section in the user manual. Note: This register field is sticky. 00_H **MIN_VAL**: Min value FF_H **MAX_VAL**: Max value
RSVDP_24	30:24	r	**Reserved for future use**
PTM_REQ_SCALED_CLOCK_T_EN	31	rwh	**PTM Requester Scaled Clock Period Enable** This bit is cleared when the core_clk rate starts to change and can only be set when the clock period change is complete. For more details, see the PTM section in the user manual. Note: This register field is sticky. 0_B **DISABLE**: Disable 1_B **ENABLE**: Use the programmed scaled PTM clock period rather than the nominal values

19.4.1.224 PTM Requester Latency Viewport Register Select

For more details, see the PTM section in the user manual.

PTM_REQ_LATENCY_REG_SEL_OFF Offset address: 0004328_H

PTM Requester Latency Viewport Register Select Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_8															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVDP_8								PTM_REQ_ESM_SEL	RSVDP_4			PTM_REQ_LATENCY_REG_SEL			
r								rw	r			rw			

Field	Bits	Type	Description
PTM_REQ_LATENCY_REG_SEL	3:0	rw	**Selects the PTM Requester Tx and Rx Latency Viewport register to be written or read, together with the PTM_REQ_ESM_SEL field** For more details, see the PTM section in the user manual. Note: This register field is sticky. 0_H **MIN_VAL**: Min value F_H **MAX_VAL**: Max value
RSVDP_4	6:4	r	**Reserved for future use**
PTM_REQ_ESM_SEL	7	rw	**Selection Mode for PTM Requester Tx and Rx Latency Viewport register to be written or read** For more details, see the PTM section in the user manual. Note: This register field is sticky. 0_B **SEL_PCIE_REGS**: write or read latency registers for PCIe rates 1_B **SEL_CCIX_ESM_REGS**: write or read latency registers for additional ESM rates
RSVDP_8	31:8	r	**Reserved for future use**

19.4.1.225 PCIe Extended Capability ID, Capability Version, and Next Capability Offset Register

This register provides capability ID, capability version, and next capability offset for PCIe extended capability structure.

VSECDMA_EXT_CAP_HDR_OFF Offset address: $000432C_H$

PCIe Extended Capability ID, Capability Version, and Next Capability Offset Register — Kernel Reset value: $0001\ 000B_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
NEXT_OFFSET												CAP			
rw												rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ID															
rw															

Field	Bits	Type	Description
ID	15:0	rw	**PCI Express Extended Capability ID** This field is a PCI-SIG defined ID number that indicates the nature and format of the Extended Capability. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky.

(table continues...)

(continued)

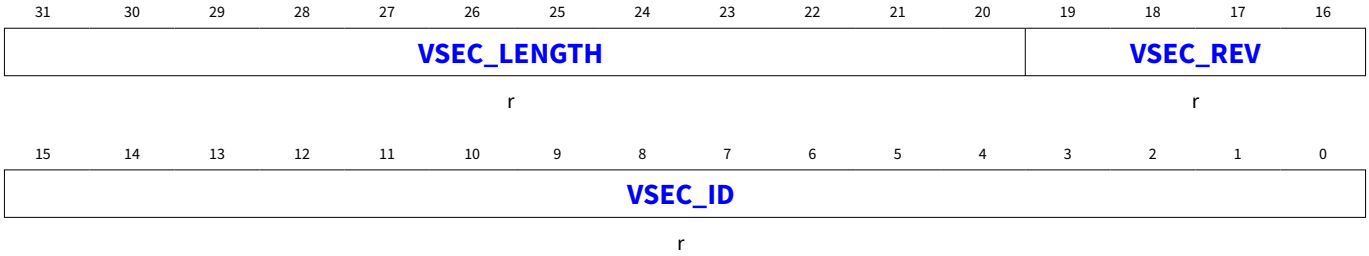
Field	Bits	Type	Description
CAP	19:16	rw	**Capability Version** This field is a PCI-SIG defined version number that indicates the version of the Capability structure present. A version of the specification that changes the Extended Capability in a way that is not otherwise identifiable (that is, through a new Capability field) is permitted to increment this field. All such changes to the Capability structure must be software-compatible. Software must check for Capability Version numbers that are greater than or equal to the highest number defined when the software is written, as Functions reporting any such Capability Version numbers will contain a Capability structure that is compatible with that piece of software. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky.
NEXT_OFFSET	31:20	rw	**Next Capability Offset** This field contains the offset to the next PCI Express Capability structure or 000h if no other items exist in the linked list of Capabilities. For Extended Capabilities implemented in Configuration Space, this offset is relative to the beginning of PCI-compatible Configuration Space and thus must always be either 000h (for terminating list of Capabilities) or greater than 0FFh. The bottom 2 bits of this offset are Reserved and must be implemented as 00b although software must mask them to allow for future uses of these bits. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky.

19.4.1.226 Vendor Specific Header Register

This Register provides VSEC Length, VSEC ID, and VSEC Rev(Version Number). Vendor-specific software must qualify the associated Vendor ID of the PCI Express Function or RCRB before attempting to interpret the values in the VSEC ID or VSEC Rev fields.

VSECDMA_VENDOR_SPECIFIC_HDR_OFF Offset address: 0004330_H

Vendor Specific Header Register Kernel Reset value: $0180\ 0006_H$



Field	Bits	Type	Description
VSEC_ID	15:0	r	**VSEC ID** This field is a vendor-defined ID number that indicates the nature and format of the VSEC structure. Software must qualify the Vendor ID before interpreting this field.

(table continues...)

(continued)

Field	Bits	Type	Description
VSEC_REV	19:16	r	**VSEC Rev** This field is a vendor-defined version number that indicates the version of the VSEC structure. Software must qualify the Vendor ID and VSEC ID before interpreting this field.
VSEC_LENGTH	31:20	r	**VSEC Length** This field indicates the number of bytes in the entire VSEC structure, including the Vendor-Specific Extended Capability Header, the Vendor-Specific Header, and the vendor-specific registers.

19.4.1.227 DMA and related AXI Bridge Implementation Information

This register provides DMA and AXI Bridge implementation-specific information: - DMA Architecture: Legacy DMA or Hyper DMA (HDMA). - Register Location: Port-logic or Mapped to a Function and BAR. - Register Map: Legacy DMA or HDMA. - Channel Separation: Address distance between read and write channels. - AXI Bridge: Used or Not Used - AXI Master Bus Specification: AXI Master Bus Width, Burst Length, and Boundary Pointer Width You must use this information along with the IP version registers: - PCIE_VERSION_NUMBER_OFF - PCIE_VERSION_TYPE_OFF

VSECDMA_DEVICE_INFORMATION_OFF Offset address: 0004334_H

DMA and related AXI Bridge Implementation Information Kernel Reset value: $3498\ 0001_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_30		MASTER_PAGE_BOUNDARY_POINTER_WIDTH				MASTER_BURST_LENGTH			MASTER_BUS_WIDTH			AXI	CHANNEL_SEPARATION		
r		r				r			r			r	r		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PFN					BARN			RSVDP_3					MAP_FORMAT		
r					r			r					r		

Field	Bits	Type	Description
MAP_FORMAT	2:0	r	**Defines the register map format and features to be one of the following values: Other values are reserved** 000_B **EDMA_LEGACY_PL**: Legacy DMA register map accessed by the port-logic registers 001_B **EDMA_LEGACY_UNROLL**: Legacy DMA register map, mapped to a PF/BAR 101_B **HDMA_COMPATIBILITY_MODE**: HDMA compatibility mode (CC_LEGACY_DMA_MAP =1) register map, mapped to a PF/BAR 111_B **HDMA_NATIVE**: HDMA native (CC_LEGACY_DMA_MAP =0) register map, mapped to a PF/BAR
RSVDP_3	7:3	r	**Reserved for future use**
BARN	10:8	r	**Bar Number** This field provides information regarding the DMA register and BAR number mapping.

(table continues...)

(continued)

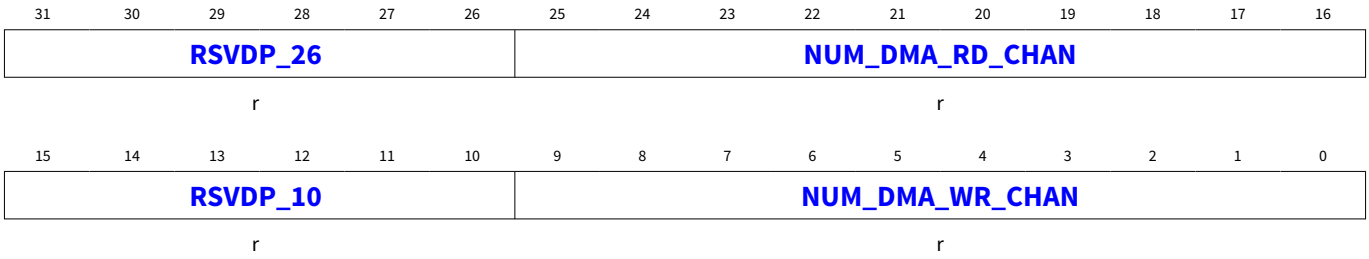
Field	Bits	Type	Description
PFN	15:11	r	**Physical Function Number** This field provides information regarding the DMA register and physical function mapping.
CHANNEL_SEPARATION	18:16	r	**If the MAP_FORMAT is set to HDMA_NATIVE, this field specifies the read write channel address separation** Other values are reserved. 000_B **CHSEP_256**: 256 separated 001_B **CHSEP_512**: 512 separated 010_B **CHSEP_1K**: 1k separated 011_B **CHSEP_2K**: 2k separated 100_B **CHSEP_4K**: 4k separated 101_B **CHSEP_8K**: 8k separated 110_B **CHSEP_16K**: 16k separated 111_B **CHSEP_32K**: 32k separated
AXI	19	r	**This field provides information about AXI interface usage** It reports the value of AXI_POPULATED configuration parameter.
MASTER_BUS_WIDTH	22:20	r	**This field provides information regarding the AXI master data bus width** It reports the value of width of the AXI master databus. 000_B **MSTR_BUSWD_32**: 32 bits 001_B **MSTR_BUSWD_64**: 64 bits 010_B **MSTR_BUSWD_128**: 128 bits 011_B **MSTR_BUSWD_256**: 256 bits 100_B **MSTR_BUSWD_512**: 512 bits **others,** Reserved
MASTER_BURST_LENGTH	25:23	r	**Reports the maximum burst length (in beats) of an AXI transfer at the master interface** 000_B **MSTR_BRSTLEN_8**: 8 bits 001_B **MSTR_BRSTLEN_16**: 16 bits 010_B **MSTR_BRSTLEN_32**: 32 bits 011_B **MSTR_BRSTLEN_64**: 64 bits 100_B **MSTR_BRSTLEN_128**: 128 bits 101_B **MSTR_BRSTLEN_256**: 256 bits **others,** Reserved
MASTER_PAGE_BOUNDARY_POINTER_WIDTH	29:26	r	**This field provides address page boundary information** It reports the value of the page boundary of 128, 256, 512, 1K, 2K or 4K (bytes). .
RSVDP_30	31:30	r	**Reserved for future use**

19.4.1.228 Number of Implemented Channels Register

This register specifies the number of read and write channels implemented.

VSECDMA_NUM_CHAN_OFF Offset address: 0004338_H

Number of Implemented Channels Register Kernel Reset value: $0004\ 0004_H$



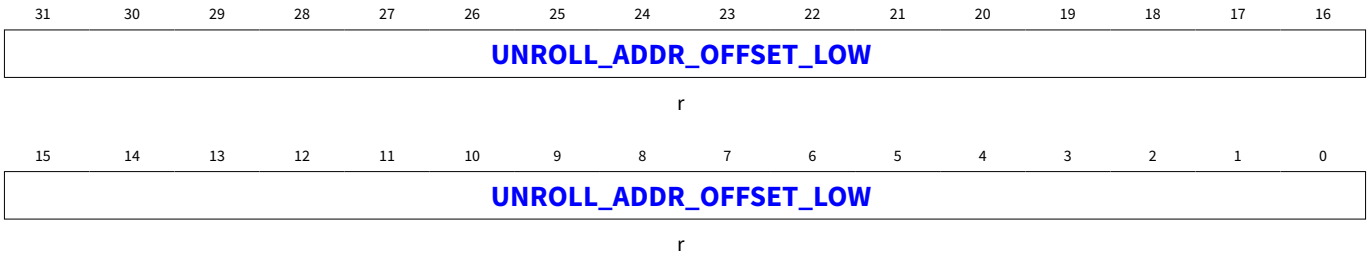
Field	Bits	Type	Description
NUM_DMA_WR_CHAN	9:0	r	**This field provides information regarding the number of implemented write channels** It reports the value of the number of write channels implemented in the DMA logic.
RSVDP_10	15:10	r	**Reserved for future use**
NUM_DMA_RD_CHAN	25:16	r	**This field provides information regarding the number of implemented read channels** It reports the value of the number of read channels implemented in the DMA logic.
RSVDP_26	31:26	r	**Reserved for future use**

19.4.1.229 DMA Register Map Start Address Offset Low Register

This register specifies the lower 32 bits of the offset of the start address of the DMA register map.

VSECDMA_UNROLL_ADDR_OFFSET_LOW_OFF Offset address: $000433C_H$

DMA Register Map Start Address Offset Low Register Kernel Reset value: $0000\ 4000_H$



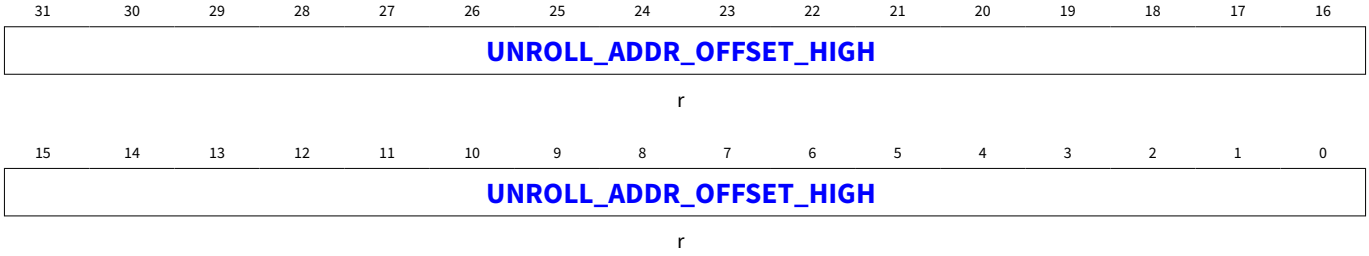
Field	Bits	Type	Description
UNROLL_ADDR_OFFSET_LOW	31:0	r	**BAR address offset, 32-bit LSB**

19.4.1.230 DMA Register Map Start Address Offset High Register

This register specifies the upper 32 bits of the offset of the start address of the DMA register map.

VSECDMA_UNROLL_ADDR_OFFSET_HIGH_OFF Offset address: 0004340_H

DMA Register Map Start Address Offset High Register Kernel Reset value: $0000\ 0000_H$



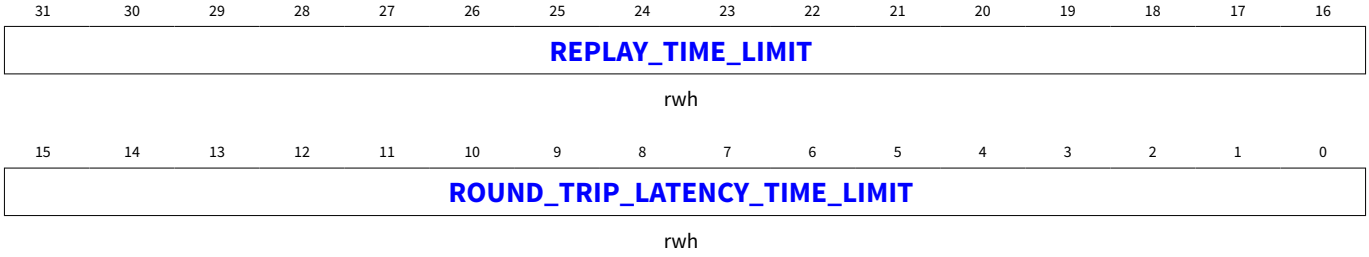
Field	Bits	Type	Description
UNROLL_ADDR_OFFSET_HIGH	31:0	r	**BAR address offset, 32-bit MSB**

19.4.1.231 Ack Latency Timer and Replay Timer Register

This register holds the ack latency timer limit and replay timer limit values.

ACK_LATENCY_TIMER_OFF Offset address: 0004700_H

Ack Latency Timer and Replay Timer Register Kernel Reset value: $0C23\ 040B_H$



Field	Bits	Type	Description
ROUND_TRIP_LATENCY_TIME_LIMIT	15:0	rwh	**Ack Latency Timer Limit** The Ack latency timer expires when it reaches this limit. - You can modify the effective timer limit through the TIMER_MOD_ACK_NAK field of the TIMER_CTRL_MAX_FUNC_NUM_OFF register. - After reset, the controller updates the default according to the Negotiated Link Width, Max_Payload_Size, and speed. The value is determined from Tables 3-7, 3-8, and 3-9 of the PCI Express Base Specification. - The limit must reflect the round trip latency from requester to completer. - If there is a change in the payload size or link width, the controller overrides any value that you have written to this register field, and resets the field back to the specification-defined value. The controller does not change the value in the TIMER_MOD_ACK_NAK field of the TIMER_CTRL_MAX_FUNC_NUM_OFF register.

(table continues...)

(continued)

Field	Bits	Type	Description
REPLAY_TIME_LIMIT	31:16	rwh	**Replay Timer Limit** The replay timer expires when it reaches this limit. The controller initiates a replay upon reception of a NAK or when the replay timer expires. - You can modify the effective timer limit through the TIMER_MOD_REPLAY_TIMER field of the TIMER_CTRL_MAX_FUNC_NUM_OFF register. - After reset, the controller updates the default according to the Negotiated Link Width, Max_Payload_Size, and speed. The value is determined from Tables 3-4, 3-5, and 3-6 of the PCI Express Base Specification. - If there is a change in the payload size or link speed, the controller overrides any value that you have written to this register field, and resets the field back to the specification-defined value. The controller does not change the value in the TIMER_MOD_REPLAY_TIMER field of the TIMER_CTRL_MAX_FUNC_NUM_OFF register.

19.4.1.232 Vendor Specific DLLP Register

This register holds the vendor specific DLLP.

VENDOR_SPEC_DLLP_OFF Offset address: 0004704_H

Vendor Specific DLLP Register Kernel Reset value: $FFFF\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
VENDOR_SPEC_DLLP															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VENDOR_SPEC_DLLP															
rw															

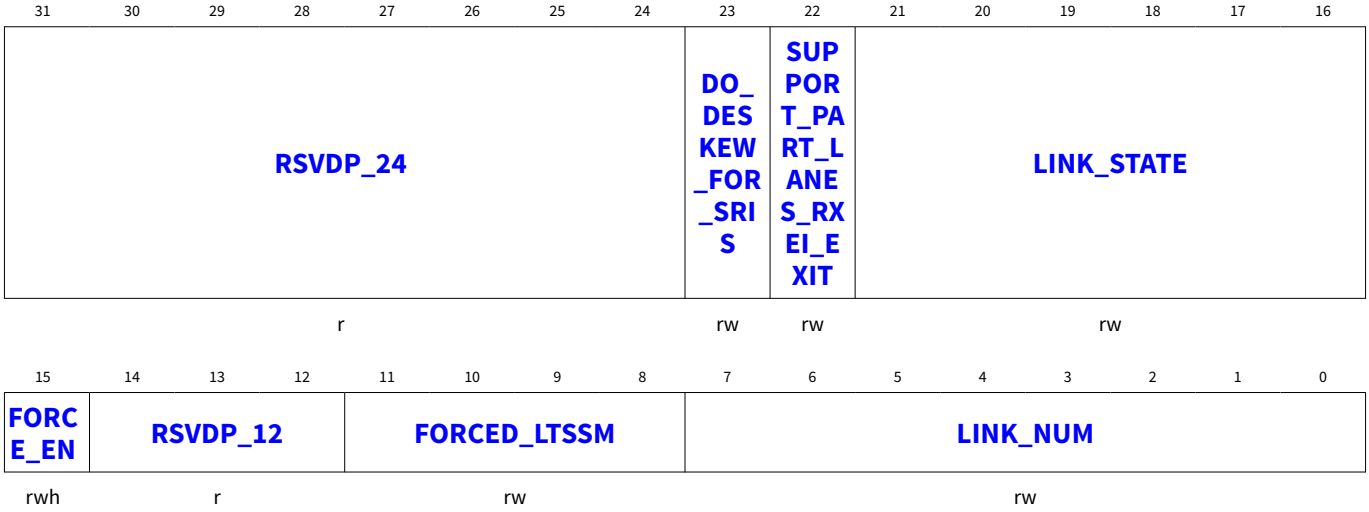
Field	Bits	Type	Description
VENDOR_SPEC_DLLP	31:0	rw	**Vendor Specific DLLP Register** You can use this register to send a specific PCI Express DLLP. Your application can write 8-bit DLLP Type and 24-bit Payload data into this register, and set the VENDOR_SPECIFIC_DLLP_REQ field of the PORT_LINK_CTRL_OFF, to send the DLLP. - Bits[7:0]: DLLP Type - Bits[31:8]: Vendor Defined Payload (24 bits) Note: This register field is sticky. 00000000_H **ZERO**: Zero value $FFFFFFFF_H$ **MAX**: Max value

19.4.1.233 Port Force Link Register

This register can be used for testing and debugging the link.

PORT_FORCE_OFF Offset address: 0004708_H

Port Force Link Register Kernel Reset value: $0080\ 0001_H$



Field	Bits	Type	Description
LINK_NUM	7:0	rw	**Link Number** Not used for endpoint. Note: This register field is sticky.
FORCED_LTSSM	11:8	rw	**Forced Link Command** The link command that the controller is forced to transmit when you set FORCE_EN bit (Force Link). Link command encoding is defined by the ltssm_cmd variable in workspace/src/Layer1/smlh_ltssm.v. Note: This register field is sticky.
RSVDP_12	14:12	r	**Reserved for future use**
FORCE_EN	15	rwh	**Force Link** The controller supports a testing and debug capability to allow your software to force the LTSSM state machine into a specific state, and to force the controller to transmit a specific Link Command. Asserting this bit triggers the following actions: • Forces the LTSSM to the state specified by the Forced LTSSM State field • Forces the controller to transmit the command specified by the Forced Link Command field This is a self-clearing register field. Reading from this register field always returns a '0'. 0_B **DISABLE**: Disable 1_B **ENABLE**: Enable
LINK_STATE	21:16	rw	**Forced LTSSM State** The LTSSM state that the controller is forced to when you set the FORCE_EN bit (Force Link). . Note: This register field is sticky.

(table continues...)

(continued)

Field	Bits	Type	Description
SUPPORT_PART_LANES_RXEI_EXIT	22	rw	**Support LTSSM transition from Polling.Active to Polling.Config based on Rx 8 TSs on any lanes which are Rx EI exit too from base spec after 24ms timeout** This prevents some lanes detected but not Rx EI exit and LTSSM cannot move to Polling.Config. Note: This register field is sticky. 0_B **UNSUPPORT**: Any lanes receives 8 consecutive TS OS and all predetermined lanes have Rx ElecIdle exit, LTSSM moves from Polling.Active to Polling.Config This is legacy mode from Base Spec. Any predetermined lanes are still on Rx ElecIdle, LTSSM moves from Polling.Active to Polling.Compliance. 1_B **SUPPORT**: Any lanes receives 8 consecutive TS OSs, LTSSM moves from Polling.Active to Polling.Config If all lanes do not receive 8 consecutive TS OSs and any predetermined lanes are still on Rx ElecIdle, LTSSM moves from Polling.Active to Polling.Compliance.
DO_DESKEW_FOR_SRIS	23	rw	**Use the transitions from TS2 to Logical Idle Symbol, SKP OS to Logical Idle Symbol, EIEOS to Logical Idle Symbol, and FTS Sequence to SKP OS to do deskew instead of using received SKP OS or TS1 to TS2 transition if DO_DESKEW_FOR_SRIS is set to '1'** Note: This register field is sticky.
RSVDP_24	31:24	r	**Reserved for future use**

19.4.1.234 Ack Frequency and L0-L1 ASPM Control Register

This register is used to control ack frequency and L0-L1 ASPM behaviour.

ACK_F_ASPM_CTRL_OFF Offset address: 000470C$_H$

Ack Frequency and L0-L1 ASPM Control Register Kernel Reset value: 09B4 B400$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_31	ENTER_ASPM	L1_ENTRANCE_LATENCY			L0S_ENTRANCE_LATENCY			COMMON_CLK_N_FTS							
r	rw	rw			rw			r							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ACK_N_FTS								ACK_FREQ							
rw								rw							

Field	Bits	Type	Description
ACK_FREQ	7:0	rw	**Ack Frequency** The controller accumulates the number of pending ACKs specified here (up to 255) before scheduling an ACK DLLP • 0: Indicates that this Ack Frequency Counter feature is turned off. The controller generates a low-priority ACK request for every TLP that it receives. The controller waits until the ACK Latency Timer expires, then converts the current low-priority ACK request to a high-priority ACK request and schedules the DLLP for transmission to the remote link partner • 1-255: Indicates that the controller will schedule a high-priority ACK after receiving this number of TLPs. It might schedule the ACK before receiving this number of TLPs if the ACK Latency Timer expires, but never later For a typical system, you do not have to modify the default setting. For more details, see "ACK/NAK Scheduling" in the user manual. Note: This register field is sticky. 00_H **MIN_VAL**: The value '0' indicates that this Ack Frequency Counter feature is turned off FF_H **MAX_VAL**: Any value between 1 and 255 indicates that the controller will schedule a high-priority ACK after receiving the specified number of TLPs
ACK_N_FTS	15:8	rw	**The number of Fast Training Sequence(N_FTS) ordered sets to be transmitted when transitioning from L0s to L0** Note: This register field is sticky. 00_H **ZERO**: The controller does not support a value of zero; a value of zero can cause the LTSSM to go into the recovery state when exiting from L0s FF_H **MAX_VAL**: The maximum number of FTS ordered-sets that a component can request is 255
COMMON_CLK_N_FTS	23:16	r	**Common Clock N_FTS** This is the N_FTS when common clock is used. The number of Fast Training Sequence ordered sets to be transmitted when transitioning from L0s to L0. Note: The access attributes of this field are as follows: - Wire: R - Dbi: R 00_H **ZERO**: The controller does not support a value of zero; a value of zero can cause the LTSSM to go into the recovery state when exiting from L0s FF_H **MAX_VAL**: The maximum number of FTS ordered-sets that a component can request is 255

(table continues...)

(continued)

Field	Bits	Type	Description
L0S_ENTRANCE_LATENCY	26:24	rw	**L0s Entrance Latency** Note: This register field is sticky. 000_B **_1_US**: 1 µs 001_B **_2_US**: 2 µs 010_B **_3_US**: 3 µs 011_B **_4_US**: 4 µs 100_B **_5_US**: 5 µs 101_B **_6_US**: 6 µs 110_B **_7_US**: 7 µs 111_B **_7US_**: 7 µs
L1_ENTRANCE_LATENCY	29:27	rw	**L1 Entrance Latency** Note: Programming this timer with a value greater that 32 µs has no effect unless extended sync is used, or all of the credits are infinite. Note: This register field is sticky. 000_B **_1_US**: 1 µs 001_B **_2_US**: 2 µs 010_B **_4_US**: 4 µs 011_B **_8_US**: 8 µs 100_B **_16_US**: 16 µs 101_B **_32_US**: 32 µs 110_B **_64_US**: 64 µs 111_B **_64US_**: 64 µs
ENTER_ASPM	30	rw	**ASPM L1 Entry Control** Note: This register field is sticky. 0_B **L0S_TO_L1**: Controller enters ASPM L1 only after idle period during which both receive and transmit are in L0s 1_B **IDLE_TO_L1**: Controller enters ASPM L1 after a period in which it has been idle
RSVDP_31	31	r	**Reserved for future use**

19.4.1.235 Port Link Control Register

Using this register you can control the port link behaviour.

PORT_LINK_CTRL_OFF Offset address: 0004710_H

Port Link Control Register Kernel Reset value: $0001\ 0120_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_28				TRANSMIT_LANE_REVERSALE_ENABLE	EXTENDED_SYNCH	CORRUPT_LCRC_ENABLE	BEACON_ENABLE	0		LINK_CAPABLE					
r				rw	rw	rw	rw	r		rw					

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVDP_12				LINK_RATE				Res	LINK_DISABLE	DLL_LINK_EN	RSVDP_4	RESET_ASSERT	LOOPBACK_ENABLE	SCRAMBLE_DISABLE	VENDOR_SPECIFIC_DLLP_REQ
r				rw				r	rw	rw	r	rw	rw	rw	rw1ch

Field	Bits	Type	Description
VENDOR_SPECIFIC_DLLP_REQ	0	rw1ch	**Vendor Specific DLLP Request** Reading from this self-clearing register field always returns a '0'. Software write with 1 clears this bit-field. 0_B **CLEAR**: This is a self clearing register 1_B **SET**: When software writes a '1' to this bit, the controller transmits the DLLP contained in the VENDOR_SPEC_DLLP field of VENDOR_SPEC_DLLP_OFF
SCRAMBLE_DISABLE	1	rw	**Scramble Disable** Turns off data scrambling. Note: The access attributes of this field are as follows: - Wire: R/W (sticky) - Dbi: R/W (sticky) Note: This register field is sticky. 0_B **DISABLE**: Disable 1_B **ENABLE**: Enable
LOOPBACK_ENABLE	2	rw	**Loopback Enable** Turns on loopback. For more details, see "Loopback" in the user manual. Note: The access attributes of this field are as follows: - Wire: R/W (sticky) - Dbi: R/W (sticky) Note: This register field is sticky. 0_B **DISABLE**: Disable 1_B **ENABLE**: Enable
RESET_ASSERT	3	rw	**Reset Assert** Triggers a recovery and forces the LTSSM to the hot reset state (downstream port only). Note: This register field is sticky. 0_B **CLEAR**: Clear 1_B **SET**: Set
RSVDP_4	4	r	**Reserved for future use**

(table continues...)

(continued)

Field	Bits	Type	Description
DLL_LINK_EN	5	rw	**DLL Link Enable** Note: This register field is sticky. 0_B **DISABLE**: The controller does not transmit InitFC DLLPs and does not establish a link 1_B **ENABLE**: Enables link initialization
LINK_DISABLE	6	rw	**LINK_DISABLE is an internally reserved field** Do not use. Note: This register field is sticky.
LINK_RATE	11:8	rw	**LINK_RATE is an internally reserved field** Do not use. Note: This register field is sticky.
RSVDP_12	15:12	r	**Reserved for future use**
LINK_CAPABLE	21:16	rw	**Link Mode Enable** Sets the number of lanes in the link that you want to connect to the link partner.Note: The access attributes of this field are as follows: - Wire: R/W (sticky) - Dbi: R/W (sticky) Note: This register field is sticky. 01_H **X1**: x1 03_H **X2**: x2 07_H **X4**: x4 $0F_H$ **X8**: x8 $1F_H$ **X16**: x16 $3F_H$ **X32**: x32 (not supported)
BEACON_ENABLE	24	rw	**BEACON_ENABLE is an internally reserved field** Do not use. Note: This register field is sticky.
CORRUPT_LCRC_ENABLE	25	rw	**CORRUPT_LCRC_ENABLE is an internally reserved field** Do not use. Note: This register field is sticky.
EXTENDED_SYNCH	26	rw	**EXTENDED_SYNCH is an internally reserved field** Do not use. Note: This register field is sticky.
TRANSMIT_LANE_REVERSALE_ENABLE	27	rw	**TRANSMIT_LANE_REVERSALE_ENABLE is an internally reserved field** Do not use. Note: This register field is sticky.
RSVDP_28	31:28	r	**Reserved for future use**
0	23:22	r	**Reserved** Read as all 0's; should be written with all 0's.

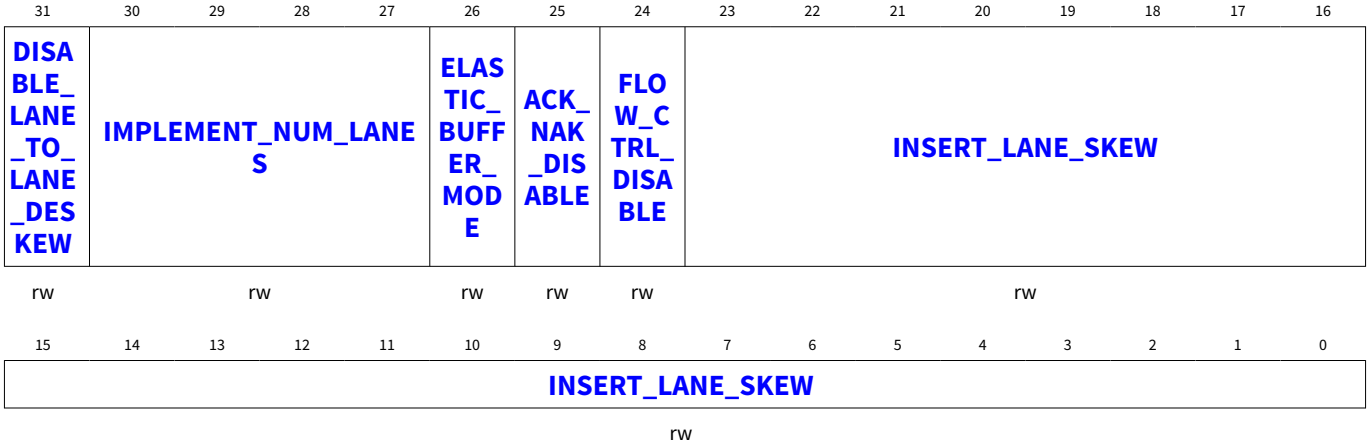
19.4.1.236 Lane Skew Register

This register is used to control the lane skew behaviour.

LANE_SKEW_OFF Offset address: 0004714_H

Lane Skew Register Kernel Reset value: $0400\ 0000_H$

19 Peripheral Component Interconnect Express (PCIe)



Field	Bits	Type	Description
INSERT_LANE_SKEW	23:0	rw	**INSERT_LANE_SKEW is an internally reserved field** Do not use. Note: This register field is sticky. 000000_H **MIN**: Zero value $0000FF_H$ **MAX**: Max value **others,** Reserved
FLOW_CTRL_DISABLE	24	rw	**Flow Control Disable** Prevents the controller from sending FC DLLPs. Note: This register field is sticky. 0_B **DISABLE**: Disable 1_B **ENABLE**: Enable
ACK_NAK_DISABLE	25	rw	**Ack/Nak Disable** Prevents the controller from sending ACK and NAK DLLPs. Note: This register field is sticky. 0_B **DISABLE**: Disable 1_B **ENABLE**: Enable
ELASTIC_BUFFER_MODE	26	rw	**Selects Elasticity Buffer operating mode:** Note: The access attributes of this field are as follows: - Wire: R/W (sticky) - Dbi: R/W (sticky) Note: This register field is sticky. 0_B **HALF_FULL**: Nominal Half Full Buffer mode 1_B **EMPTY**: Nominal Empty Buffer Mode

(table continues...)

(continued)

Field	Bits	Type	Description
IMPLEMENT_NUM_LANES	30:27	rw	**Implementation-specific Number of Lanes** Set the implementation-specific number of lanes. The number of lanes to be used when in Loopback Master. The number of lanes programmed must be equal to or less than the valid number of lanes set in LINK_CAPABLE field. You must configure this field before initiating Loopback by writing in the LOOPBACK_ENABLE field. The controller will transition from Loopback.Entry to Loopback.Active after receiving two consecutive TS1 Ordered Sets with the Loopback bit asserted on the implementation specific number of lanes configured in this field. Note: This register field is sticky. 0_H **_1_LANE**: 1 lane 1_H **_2_LANE**: 2 lanes 3_H **_4_LANE**: 4 lanes 7_H **_8_LANE**: 8 lanes F_H **_16_LANE**: 16 lanes
DISABLE_LANE_TO_LANE_DESKEW	31	rw	**Disable Lane-to-Lane Deskew** Causes the controller to disable the internal Lane-to-Lane deskew logic. Note: The access attributes of this field are as follows: - Wire: R/W (sticky) - Dbi: R/W (sticky) Note: This register field is sticky. 0_B **DISABLE**: Disable 1_B **ENABLE**: Enable

19.4.1.237 Timer Control and Max Function Number Register

This register holds the ack frequency, latency, replay, and max function number values.

TIMER_CTRL_MAX_FUNC_NUM_OFF Offset address: 0004718_H

Timer Control and Max Function Number Register Reset values see: Table 874

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_31	Res		UPDATE_FREQ_TIMER					TIMER_MOD_ACK_NAK					TIMER_MOD_REPLAY_TIMER		
r	r		rw					rw					rwh		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TIMER_MOD_REPLAY_TIMER			RSVDP_8					MAX_FUNC_NUM							
rwh			r					rw							

Field	Bits	Type	Description
MAX_FUNC_NUM	7:0	rw	**Maximum function number that can be used in a request** Configuration requests targeted at function numbers above this value are returned with UR (unsupported request). Note: This register field is sticky. 00_H **MIN**: Zero value FF_H **MAX**: Max value
RSVDP_8	13:8	r	**Reserved for future use**
TIMER_MOD_REPLAY_TIMER	18:14	rwh	**Replay Timer Limit Modifier** Increases the time-out value for the replay timer in increments of 64 clock cycles at Gen1 or Gen2 speed, and in increments of 256 clock cycles at Gen3 speed. A value of '0' represents no modification to the timer limit. For more details, see the REPLAY_TIME_LIMIT field of the ACK_LATENCY_TIMER_OFF register. At Gen3 speed, the controller automatically changes the value of this field to DEFAULT_GEN3_REPLAY_ADJ. Note: This register field is sticky.
TIMER_MOD_ACK_NAK	23:19	rw	**Ack Latency Timer Modifier** Increases the timer value for the Ack latency timer in increments of 64 clock cycles. A value of '0' represents no modification to the timer value. For more details, see the ROUND_TRIP_LATENCY_TIME_LIMIT field of the ACK_LATENCY_TIMER_OFF register. Note: This register field is sticky.
UPDATE_FREQ_TIMER	28:24	rw	**UPDATE_FREQ_TIMER is an internally reserved field** Do not use. Note: This register field is sticky.
RSVDP_31	31	r	**Reserved for future use**

Table 874 Reset values of TIMER_CTRL_MAX_FUNC_NUM_OFF

Reset	Reset value	Note
Kernel Reset	0100 0000 0000 0XXX XX00 0000 0000 0000_B	

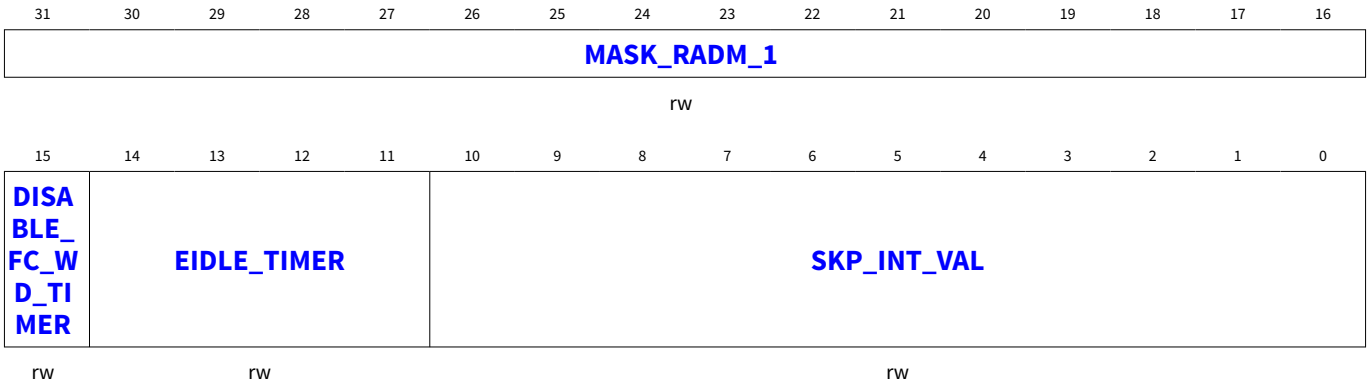
19.4.1.238 Symbol Timer Register and Filter Mask 1 Register

The Filter Mask 1 Register modifies the RADM filtering and error handling rules. For more details, see the "Receive Filtering" section. In each case, '0' applies the associated filtering rule and '1' masks the associated filtering rule.

SYMBOL_TIMER_FILTER_1_OFF Offset address: $000471C_H$

Symbol Timer Register and Filter Mask 1 Register Kernel Reset value: $0000\ 0140_H$

19 Peripheral Component Interconnect Express (PCIe)



Field	Bits	Type	Description
SKP_INT_VAL	10:0	rw	**SKP Interval Value** The number of symbol times to wait between transmitting SKP ordered sets. The controller waits the number of symbol times in this register plus 1, between transmitting SKP ordered sets. Your application must program this register accordingly. For example, if 384 were programmed into this register (62.5 MHz for GEN 1, 4 symbols/cycle), then the controller actually transmits SKP ordered sets once every 1537 (384*4+1) symbol times. The value programmed to this register is actually clock ticks and not symbol times. Note: This value is not used at Gen3 speed; the skip interval is hardcoded to 370 blocks. Note: This register field is sticky.
EIDLE_TIMER	14:11	rw	**EIDLE_TIMER is an internally reserved field** Do not use. Note: This register field is sticky.
DISABLE_FC_WD_TIMER	15	rw	**Disable FC Watchdog Timer** Note: This register field is sticky. 0_B **DISABLE**: Disable 1_B **ENABLE**: Enable

(table continues...)

(continued)

Field	Bits	Type	Description
MASK_RADM_1	31:16	rw	**Filter Mask 1** The Filter Mask 1 Register modifies the RADM filtering and error handling rules. For more details, see the "Receive Filtering" section. In each case, '0' applies the associated filtering rule and '1' masks the associated filtering rule. • [31]: CX_FLT_MASK_RC_CFG_DISCARD - 0: For RADM RC filter to not allow CFG transaction being received - 1: For RADM RC filter to allow CFG transaction being received • [30]: CX_FLT_MASK_RC_IO_DISCARD - 0: For RADM RC filter to not allow IO transaction being received - 1: For RADM RC filter to allow IO transaction being received • [29]: CX_FLT_MASK_MSG_DROP - 0: Drop MSG TLP (except for Vendor MSG). Send decoded message on the SII - 1: Do not Drop MSG (except for Vendor MSG). Send message TLPs to your application on TRGT1 and send decoded message on the SII - The default for this bit is the inverse of FLT_DROP_MSG. That is, if FLT_DROP_MSG =1, then the default of this bit is '0' (drop message TLPs). This bit only controls message TLPs other than Vendor MSGs. Vendor MSGs are controlled by Filter Mask Register 2, bits [1:0]. The controller never passes ATS Invalidate messages to the SII interface regardless of this filter rule setting. The controller passes all ATS Invalidate messages to TRGT1 (or AXI bridge master), as they are too big for the SII • [28]: CX_FLT_MASK_CPL_ECRC_DISCARD - Only used when completion queue is advertised with infinite credits and is in store-and-forward mode. - 0: Discard completions with ECRC errors - 1: Allow completions with ECRC errors to be passed up - Reserved field for SW • [27]: CX_FLT_MASK_ECRC_DISCARD - 0: Discard TLPs with ECRC errors - 1: Allow TLPs with ECRC errors to be passed up • [26]: CX_FLT_MASK_CPL_LEN_MATCH - 0: Enforce length match for completions; a violation results in cpl_abort, and possibly AER of unexp_cpl_err - 1: Mask length match for completions • [25]: CX_FLT_MASK_CPL_ATTR_MATCH - 0: Enforce attribute match for completions; a violation results in a malformed TLP error, and possibly AER of unexp_cpl_err, cpl_rcvd_ur, cpl_rcvd_ca - 1: Mask attribute match for completions • [24]: CX_FLT_MASK_CPL_TC_MATCH - 0: Enforce Traffic Class match for completions; a violation results in a malformed TLP error, and possibly AER of unexp_cpl_err, cpl_rcvd_ur, cpl_rcvd_ca - 1: Mask Traffic Class match for completions • [23]: CX_FLT_MASK_CPL_FUNC_MATCH - 0: Enforce function match for completions; a violation results in cpl_abort, and possibly AER of unexp_cpl_err, cpl_rcvd_ur, cpl_rcvd_ca - 1: Mask function match for completions • [22]: CX_FLT_MASK_CPL_REQID_MATCH - 0: Enforce Req. Id match for completions; a violation result in cpl_abort, and possibly AER of

(table continues...)

(continued)

Field	Bits	Type	Description
			unexp_cpl_err, cpl_rcvd_ur, cpl_rcvd_ca - 1: Mask Req. Id match for completions • [21]: CX_FLT_MASK_CPL_TAGERR_MATCH - 0: Enforce Tag Error Rules for completions; a violation result in cpl_abort, and possibly AER of unexp_cpl_err, cpl_rcvd_ur, cpl_rcvd_ca - 1: Mask Tag Error Rules for completions • [20]: CX_FLT_MASK_LOCKED_RD_AS_UR - 0: Treat locked Read TLPs as UR for EP; Supported for RC - 1: Treat locked Read TLPs as Supported for EP; UR for RC • [19]: CX_FLT_MASK_CFG_TYPE1_REQ_AS_UR - 0: Treat CFG type1 TLPs as UR for EP; Supported for RC - 1: Treat CFG type1 TLPs as Supported for EP; UR for RC • [18]: CX_FLT_MASK_UR_OUTSIDE_BAR - 0: Treat out-of-bar TLPs as UR - 1: Do not treat out-of-bar TLPs as UR • [17]: CX_FLT_MASK_UR_POIS - 0: Treat poisoned request TLPs as UR - 1: Do not treat poisoned request TLPs as UR - The native controller always passes poisoned completions to your application except when you are using the DMA read channel • [16]: CX_FLT_MASK_UR_FUNC_MISMATCH - 0: Treat Function MisMatched TLPs as UR - 1: Do not treat Function MisMatched TLPs as UR. CX_FLT_MASK_UR_FUNC_MISMATCH is not allowed to be set unless CONFIG_LIMIT_REG =0 and TARGET_ABOVE_CONFIG_LIMIT_REG =2 in MISC_CONTROL_1_OFF register or CFG_TLP_BYPASS_EN_REG =1 and TARGET_ABOVE_CONFIG_LIMIT_REG =2 in MISC_CONTROL_1_OFF register Note: This register field is sticky.

19.4.1.239 Filter Mask 2 Register

This register modifies the RADM filtering and error handling rules. For more details, see the "Receive Filtering" section. In each case, '0' applies the associated filtering rule and '1' masks the associated filtering rule.

FILTER_MASK_2_OFF Offset address: 0004720_H

Filter Mask 2 Register Kernel Reset value: $0000\ 0000_H$

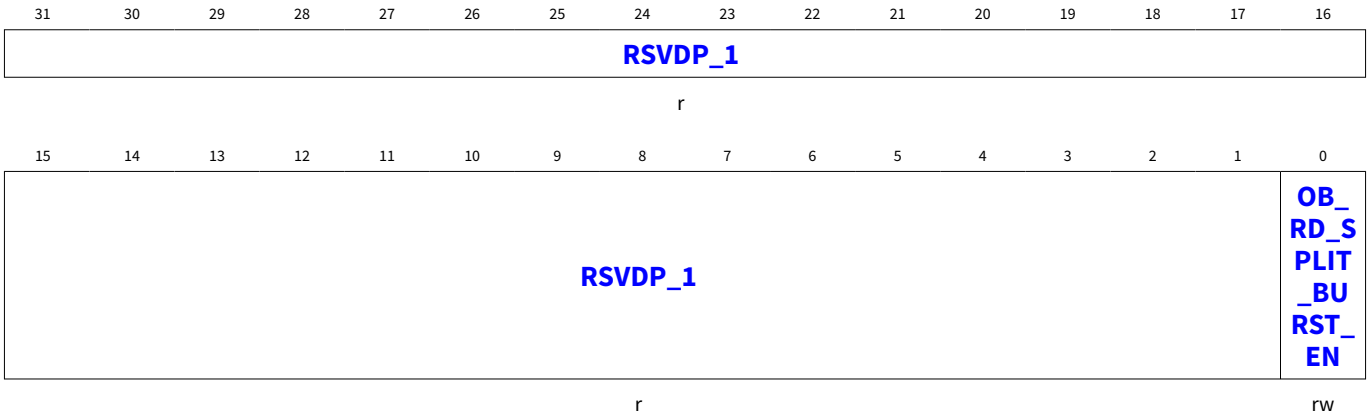
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MASK_RADM_2															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MASK_RADM_2															
rw															

Field	Bits	Type	Description
MASK_RADM_2	31:0	rw	**Filter Mask 2** This field modifies the RADM filtering and error handling rules. For more details, see the "Receive Filtering" in the user manual. In each case, '0' applies the associated filtering rule and '1' masks the associated filtering rule. • [31:10]: Reserved • [9]: CX_FLT_MASK_CPL_IN_LUT_CHECK - 0: Disable masking of checking if the tag of CPL is registered in LUT - 1: Enable masking of checking if the tag of CPL is registered in LUT • [8]: CX_FLT_MASK_POIS_ERROR_REPORTING - 0: Disable masking of error reporting for Poisoned TLPs - 1: Enable masking of error reporting for Poisoned TLPs • [7]: CX_FLT_MASK_PRS_DROP - 0: Allow PRS message to pass through - 1: Drop PRS Messages silently - This bit is ignored when the CX_FLT_MASK_MSG_DROP bit in the MASK_RADM_1 field of the SYMBOL_TIMER_FILTER_1_OFF register is set to '1' • [6]: CX_FLT_UNMASK_TD - 0: Disable unmask TD bit if CX_STRIP_ECRC_ENABLE - 1: Enable unmask TD bit if CX_STRIP_ECRC_ENABLE • [5]: CX_FLT_UNMASK_UR_POIS_TRGT0 - 0: Disable unmask CX_FLT_MASK_UR_POIS with TRGT0 destination - 1: Enable unmask CX_FLT_MASK_UR_POIS with TRGT0 destination • [4]: CX_FLT_MASK_LN_VENMSG1_DROP - 0: Allow LN message to pass through - 1: Drop LN Messages silently • [3]: CX_FLT_MASK_HANDLE_FLUSH - 0: Disable controller Filter to handle flush request - 1: Enable controller Filter to handle flush request • [2]: CX_FLT_MASK_DABORT_4UCPL - 0: Enable DLLP abort for unexpected completion - 1: Do not enable DLLP abort for unexpected completion • [1]: CX_FLT_MASK_VENMSG1_DROP - 0: Vendor MSG Type 1 dropped silently - 1: Vendor MSG Type 1 not dropped • [0]: CX_FLT_MASK_VENMSG0_DROP - 0: Vendor MSG Type 0 dropped with UR error reporting - 1: Vendor MSG Type 0 not dropped Note: This register field is sticky.

19.4.1.240 AMBA Multiple Outbound Decomposed NP SubRequests Control Register

This register controls multiple outbound decomposed NP subRequests operation.



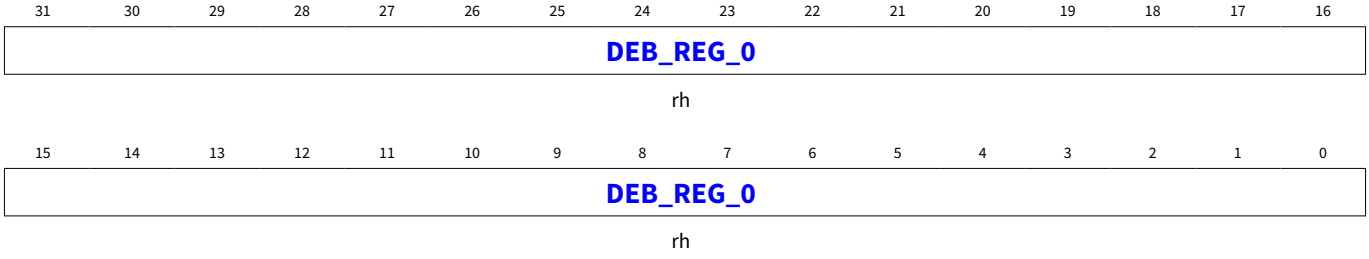
Field	Bits	Type	Description
OB_RD_SPLIT_BURST_EN	0	rw	**Enable AMBA Multiple Outbound Decomposed NP SubRequests** You should not clear this register unless your application master is requesting an amount of read data greater than Max_Read_Request_Size, and the remote device (or switch) is reordering completions that have different tags. Note: The access attributes of this field are as follows: - Wire: R/W (sticky) - Dbi: R/W (sticky) Note: This register field is sticky. 0_B **DISABLE**: This bit when set to '0' disables the possibility of having multiple outstanding non-posted requests that were derived from decomposition of an outbound AMBA request 1_B **ENABLE**: Enable
RSVDP_1	31:1	r	**Reserved for future use**

19.4.1.241 Debug Register 0

This register holds the debug info.

PL_DEBUG0_OFF Offset address: 0004728_H

Debug Register 0 Kernel Reset value: $XXXX\ XXXX_H$



Field	Bits	Type	Description
DEB_REG_0	31:0	rh	**The value on cxpl_debug_info[31:0]** [31:28]: Link control bits advertised by link partner [27]: Currently receiving k237 (PAD) in place of lane number [26]: Currently receiving k237 (PAD) in place of link number [25]: Receiver is receiving logical idle [24]: 2n symbol is also idle (16bit PHY interface only) [23:8]: PIPE(RMMI for M-PCIe) Transmit data [7:6]: PIPE(RMMI for M-PCIe) transmit K indication [5:0]: LTSSM current state. Encoding is same as the dedicated smlh_ltssm_state output. 00000000_H **ZERO**: Zero value $FFFFFFFF_H$ **MAX**: Max value

19.4.1.242 Debug Register 1

This register holds the debug info.

PL_DEBUG1_OFF Offset address: $000472C_H$

Debug Register 1 Kernel Reset value: $XXXX\ XXXX_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DEB_REG_1															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DEB_REG_1															
rh															

Field	Bits	Type	Description
DEB_REG_1	31:0	rh	**The value on cxpl_debug_info[63:32]** [63]: Scrambling disabled for the link [62]: LTSSM in DISABLE state. Link inoperable [61]: LTSSM performing link training [60]: LTSSM is in Polling.Configuration state [59]: LTSSM-negotiated link reset [58:55]: 0000b: Reserved [54]: PIPE receiver detect/loopback request [53]: PIPE transmit electrical idle request [52]: PIPE transmit compliance request [51]: Application request to initiate training reset [50:48]: 000b: Reserved [47:40]: rmlh_ts_link_num: Link number advertised/confirmed by link partner [39:38]: 00b: Reserved [37]: A skip ordered set has been transmitted [36]: LTSSM reports PHY link up or LTSSM is in Loopback. Active for Loopback Master [35]: Receiver reports skip reception [34]: TS1 training sequence received (pulse) [33]: TS2 training sequence received (pulse) [32]: Receiver detected lane reversal 00000000_H **ZERO**: Zero value $FFFFFFFF_H$ **MAX**: Max value

19.4.1.243 Transmit Posted FC Credit Status

This register provides transmit posted FC credit status.

TX_P_FC_CREDIT_STATUS_OFF Offset address: 0004730_H

Transmit Posted FC Credit Status Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_TX_P_FC_CREDIT_STATUS												TX_P_HEADER_FC_CREDIT			
r												rh			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TX_P_HEADER_FC_CREDIT				TX_P_DATA_FC_CREDIT											
rh				rh											

Field	Bits	Type	Description
TX_P_DATA_FC_CREDIT	11:0	rh	**Transmit Posted Data FC Credits** • The posted Data credits advertised by the receiver at the other end of the link, updated with each UpdateFC DLLP • Default value depends on the number of advertised credits for header and data • Scaled Flow Control: when the number of advertised posted credits (both header and data) are infinite • No Scaling: when the number of advertised posted credits (both header and data) are infinite
TX_P_HEADER_FC_CREDIT	19:12	rh	**Transmit Posted Header FC Credits** • The posted Header credits advertised by the receiver at the other end of the link, updated with each UpdateFC DLLP • Default value depends on the number of advertised credits for header and data • Scaled Flow Control: when the number of advertised posted credits (both header and data) are infinite • No Scaling: when the number of advertised posted credits (both header and data) are infinite
RSVDP_TX_P_FC_CREDIT_STATUS	31:20	r	**Reserved for future use**

19.4.1.244 Transmit Non-Posted FC Credit Status

This register provides the transmit Non-Posted FC credit status.

TX_NP_FC_CREDIT_STATUS_OFF Offset address: 0004734_H

Transmit Non-Posted FC Credit Status Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_TX_NP_FC_CREDIT_STATUS												TX_NP_HEADER_FC_CREDIT			
r												rh			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TX_NP_HEADER_FC_CREDIT				TX_NP_DATA_FC_CREDIT											
rh				rh											

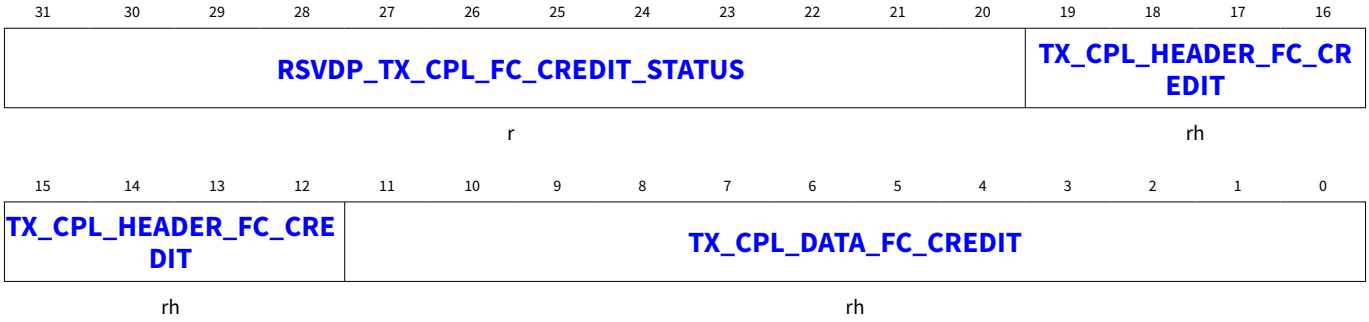
Field	Bits	Type	Description
TX_NP_DATA_FC_CREDIT	11:0	rh	**Transmit Non-Posted Data FC Credits** • The non-posted Data credits advertised by the receiver at the other end of the link, updated with each UpdateFC DLLP • Default value depends on the number of advertised credits for header and data • Scaled Flow Control: when the number of advertised non-posted credits (both header and data) are infinite • No Scaling: when the number of advertised non-posted credits (both header and data) are infinite
TX_NP_HEADER_FC_CREDIT	19:12	rh	**Transmit Non-Posted Header FC Credits** • The non-posted Header credits advertised by the receiver at the other end of the link, updated with each UpdateFC DLLP • Default value depends on the number of advertised credits for header and data
RSVDP_TX_NP_FC_CREDIT_STATUS	31:20	r	**Reserved for future use**

19.4.1.245 Transmit Completion FC Credit Status

This register provides transmit completion FC credit status.

TX_CPL_FC_CREDIT_STATUS_OFF Offset address: 0004738_H

Transmit Completion FC Credit Status Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
TX_CPL_DATA_FC_CREDIT	11:0	rh	**Transmit Completion Data FC Credits** • The Completion Data credits advertised by the receiver at the other end of the link, updated with each UpdateFC DLLP • Default value depends on the number of advertised credits for header and data
TX_CPL_HEADER_FC_CREDIT	19:12	rh	**Transmit Completion Header FC Credits** • The Completion Header credits advertised by the receiver at the other end of the link, updated with each UpdateFC DLLP • Default value depends on the number of advertised credits for header and data

(table continues...)

(continued)

Field	Bits	Type	Description
RSVDP_TX_CPL_FC_CREDIT_STATUS	31:20	r	**Reserved for future use**

19.4.1.246 Queue Status

This register provides the queue status.

QUEUE_STATUS_OFF Offset address: 000473C$_H$

Queue Status Kernel Reset value: 0000 0000$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TIMER_MOD_FLOW_CONTROL_EN		RSVDP_29							TIMER_MOD_FLOW_CONTROL						
rw		r							rw						

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		RX_SERIALIZATION_Q_NON_EMPTY	RSVDP_4									RX_QUEUE_OVERFLOW	RX_QUEUE_NON_EMPTY	TX_RETRY_BUFFER_NE	RX_TLP_FC_CREDIT_NON_RETURN
r		rh	r									rw1ch	rh	rh	rh

Field	Bits	Type	Description
RX_TLP_FC_CREDIT_NON_RETURN	0	rh	**Received TLP FC Credits Not Returned** 0_B **CLEAR**: Clear 1_B **SET**: Indicates that the controller has received a TLP but has not yet sent an UpdateFC DLLP indicating that the credits for that TLP have been restored by the receiver at the other end of the link
TX_RETRY_BUFFER_NE	1	rh	**Transmit Retry Buffer Not Empty** 0_B **CLEAR**: Clear 1_B **SET**: Indicates that there is data in the transmit retry buffer
RX_QUEUE_NON_EMPTY	2	rh	**Receive Credit Queue Not Empty** 0_B **CLEAR**: Clear 1_B **SET**: Indicates there is data in one or more of the receive buffers

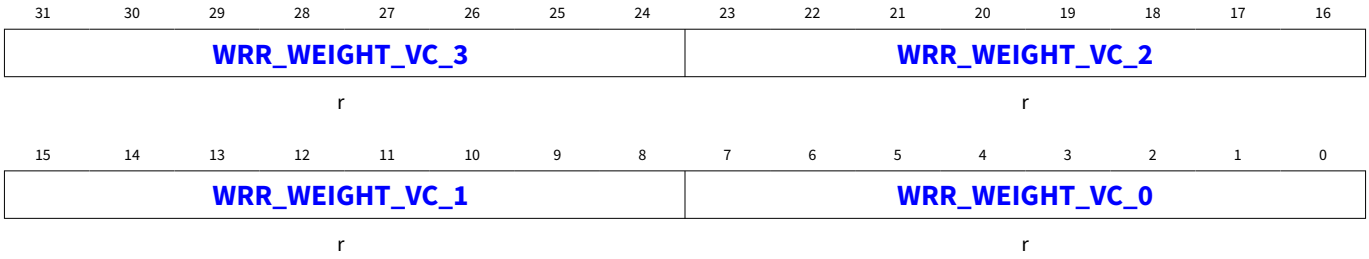
(table continues...)

(continued)

Field	Bits	Type	Description
RX_QUEUE_OVERFLOW	3	rw1ch	**Receive Credit Queue Overflow** Software write with 1 clears this bit-field.. 0_B **CLEAR**: Clear 1_B **SET**: Indicates insufficient buffer space available to write to the P/NP/CPL credit queue
RSVDP_4	12:4	r	**Reserved for future use**
RX_SERIALIZATION_Q_NON_EMPTY	13	rh	**Receive Serialization Queue Not Empty** 0_B **CLEAR**: Clear 1_B **SET**: Indicates there is data in the serialization queue
TIMER_MOD_FLOW_CONTROL	28:16	rw	**FC Latency Timer Override Value** When you set the "FC Latency Timer Override Enable" in this register, the value in this field will override the FC latency timer value that the controller calculates according to the PCIe specification. Note: This register field is sticky.
RSVDP_29	30:29	r	**Reserved for future use**
TIMER_MOD_FLOW_CONTROL_EN	31	rw	**FC Latency Timer Override Enable** Note: This register field is sticky. 0_B **CLEAR**: Clear 1_B **SET**: When this bit is set, the value from the "FC Latency Timer Override Value" field in this register will override the FC latency timer value that the controller calculates according to the PCIe specification
0	15:14	r	**Reserved** Read as all 0's; should be written with all 0's.

19.4.1.247 VC Transmit Arbitration Register 1

This register is used for setting the WRR weights for VC0 - VC3.



Field	Bits	Type	Description
WRR_WEIGHT_VC_0	7:0	r	**WRR Weight for VC0** Note: The access attributes of this field are as follows: - Wire: R - Dbi: R
WRR_WEIGHT_VC_1	15:8	r	**WRR Weight for VC1** Note: The access attributes of this field are as follows: - Wire: R - Dbi: R

(table continues...)

(continued)

Field	Bits	Type	Description
WRR_WEIGHT_VC_2	23:16	r	**WRR Weight for VC2** Note: The access attributes of this field are as follows: - Wire: R - Dbi: R
WRR_WEIGHT_VC_3	31:24	r	**WRR Weight for VC3** Note: The access attributes of this field are as follows: - Wire: R - Dbi: R

19.4.1.248 VC Transmit Arbitration Register 2

This register is used for setting the WRR weights for VC4 - VC7.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
WRR_WEIGHT_VC_7								WRR_WEIGHT_VC_6							
r								r							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WRR_WEIGHT_VC_5								WRR_WEIGHT_VC_4							
r								r							

Field	Bits	Type	Description
WRR_WEIGHT_VC_4	7:0	r	**WRR Weight for VC4** Note: The access attributes of this field are as follows: - Wire: R - Dbi: R
WRR_WEIGHT_VC_5	15:8	r	**WRR Weight for VC5** Note: The access attributes of this field are as follows: - Wire: R - Dbi: R
WRR_WEIGHT_VC_6	23:16	r	**WRR Weight for VC6** Note: The access attributes of this field are as follows: - Wire: R - Dbi: R
WRR_WEIGHT_VC_7	31:24	r	**WRR Weight for VC7** Note: The access attributes of this field are as follows: - Wire: R - Dbi: R

19.4.1.249 Segmented-Buffer VC0 Posted Receive Queue Control

This register controls segmented-buffer VC0 posted receive queue operation.

VC0_P_RX_Q_CTRL_OFF Offset address: 0004748_H

Segmented-Buffer VC0 Posted Receive Queue Control Kernel Reset value: $C522\ 2050_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
VC_ORDERING_RX_Q	TLP_TYPE_ORDERING_VC0	RESERVED5		VC0_P_DATA_SCALE		VC0_P_HDR_SCALE		VC0_P_TLP_Q_MODE			RESERVED4	VC0_P_HEADER_CREDIT			
rw	rw	rw		rw		rw		rw			rw	rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VC0_P_HEADER_CREDIT				VC0_P_DATA_CREDIT											
rw				rw											

Field	Bits	Type	Description
VC0_P_DATA_CREDIT	11:0	rw	**VC0 Posted Data Credits** The number of initial posted data credits for VC0, used only in the segmented-buffer configuration. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: R (sticky) Note: This register field is sticky. 000_H **MIN_VAL**: Zero value FFF_H **MAX_VAL**: Max value
VC0_P_HEADER_CREDIT	19:12	rw	**VC0 Posted Header Credits** The number of initial posted header credits for VC0, used only in the segmented-buffer configuration. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: R (sticky) Note: This register field is sticky. 00_H **MIN_VAL**: Zero value FF_H **MAX_VAL**: Max value
RESERVED4	20	rw	**Reserved** Note: This register field is sticky.
VC0_P_TLP_Q_MODE	23:21	rw	**Reserved** Note: This register field is sticky.
VC0_P_HDR_SCALE	25:24	rw	**VC0 Scale Posted Header Credits** Note: This register field is sticky. 00_B **MIN_VAL**: Zero value 11_B **MAX_VAL**: Max value
VC0_P_DATA_SCALE	27:26	rw	**VC0 Scale Posted Data Credits** Note: This register field is sticky. 00_B **MIN_VAL**: Zero value 11_B **MAX_VAL**: Max value
RESERVED5	29:28	rw	**Reserved** Note: This register field is sticky.
TLP_TYPE_ORDERING_VC0	30	rw	**TLP Type Ordering for VC0** Determines the TLP type ordering rule for VC0 receive queues, used only in the segmented-buffer configuration: Note: This register field is sticky. 0_B **STRICT_ORDERING**: Strict ordering: posted, completion, then non-posted 1_B **PCIE_ORDERING_RULE**: PCIe ordering rules (recommended)
VC_ORDERING_RX_Q	31	rw	**VC Ordering for Receive Queues** Determines the VC ordering rule for the receive queues, used only in the segmented-buffer configuration: Note: This register field is sticky. 0_B **ROUND_ROBIN**: Round robin 1_B **STRICT_ORDERING**: Strict ordering, higher numbered VCs have higher priority

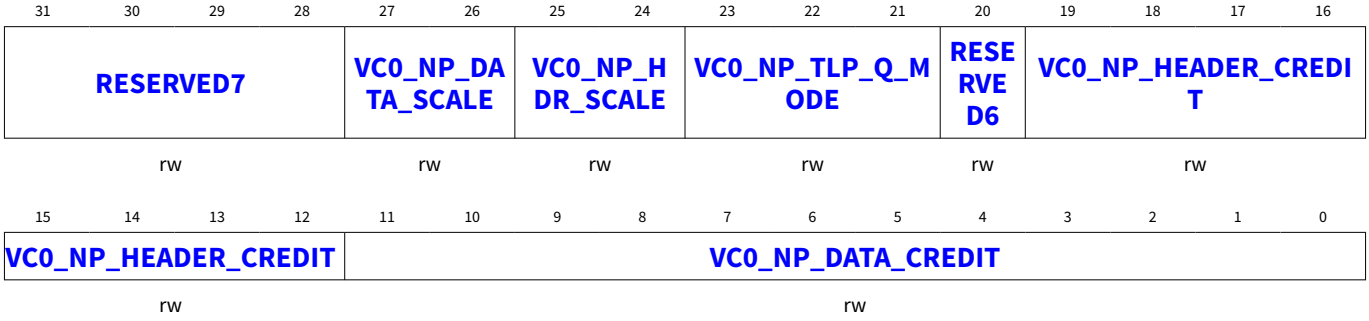
19.4.1.250 Segmented-Buffer VC0 Non-Posted Receive Queue Control

This register controls the segmented-buffer VC0 non-posted receive queue operation.

VC0_NP_RX_Q_CTRL_OFF Offset address: $000474C_H$

Segmented-Buffer VC0 Non-Posted Receive Queue Control

Kernel Reset value: $0522\ 2008_H$



Field	Bits	Type	Description
VC0_NP_DATA_CREDIT	11:0	rw	**VC0 Non-Posted Data Credits** The number of initial non-posted data credits for VC0, used only in the segmented-buffer configuration. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: R (sticky) Note: This register field is sticky. 000_H **MIN_VAL**: Zero value FFF_H **MAX_VAL**: Max value
VC0_NP_HEADER_CREDIT	19:12	rw	**VC0 Non-Posted Header Credits** The number of initial non-posted header credits for VC0, used only in the segmented-buffer configuration. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: R (sticky) Note: This register field is sticky. 00_H **MIN_VAL**: Zero value FF_H **MAX_VAL**: Max value
RESERVED6	20	rw	**Reserved** Note: This register field is sticky.
VC0_NP_TLP_Q_MODE	23:21	rw	**Reserved** Note: This register field is sticky.
VC0_NP_HDR_SCALE	25:24	rw	**VC0 Scale Non-Posted Header Credits** Note: This register field is sticky. 00_B **MIN_VAL**: Zero value 11_B **MAX_VAL**: Max value
VC0_NP_DATA_SCALE	27:26	rw	**VC0 Scale Non-Posted Data Credits** Note: This register field is sticky. 00_B **MIN_VAL**: Zero value 11_B **MAX_VAL**: Max value
RESERVED7	31:28	rw	**Reserved** Note: This register field is sticky.

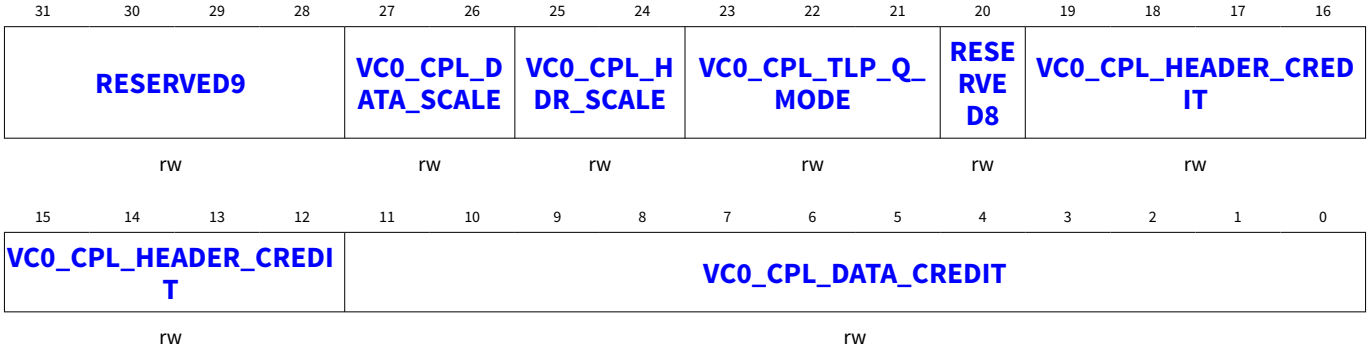
19.4.1.251 Segmented-Buffer VC0 Completion Receive Queue Control

This register controls the segmented-buffer VC0 completion receive queue operation.

VC0_CPL_RX_Q_CTRL_OFF Offset address: 0004750_H

Segmented-Buffer VC0 Completion Receive Queue Control

Kernel Reset value: $0520\ 0000_H$



Field	Bits	Type	Description
VC0_CPL_DATA_CREDIT	11:0	rw	**VC0 Completion Data Credits** The number of initial Completion data credits for VC0, used only in the segmented-buffer configuration. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: R (sticky) Note: This register field is sticky. 000_H **MIN_VAL**: Zero value FFF_H **MAX_VAL**: Max value
VC0_CPL_HEADER_CREDIT	19:12	rw	**VC0 Completion Header Credits** The number of initial Completion header credits for VC0, used only in the segmented-buffer configuration. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: R (sticky) Note: This register field is sticky. 00_H **MIN_VAL**: Zero value FF_H **MAX_VAL**: Max value
RESERVED8	20	rw	**Reserved** Note: This register field is sticky.
VC0_CPL_TLP_Q_MODE	23:21	rw	**Reserved** Note: This register field is sticky.
VC0_CPL_HDR_SCALE	25:24	rw	**VC0 Scale CPL Header Credits** Note: This register field is sticky. 00_B **MIN_VAL**: Zero value 11_B **MAX_VAL**: Max value
VC0_CPL_DATA_SCALE	27:26	rw	**VC0 Scale CPL Data Credits** Note: This register field is sticky. 00_B **MIN_VAL**: Zero value 11_B **MAX_VAL**: Max value
RESERVED9	31:28	rw	**Reserved** Note: This register field is sticky.

19.4.1.252 Link Width and Speed Change Control Register

This register controls various functions of the controller related to link training, lane reversal, and equalization.

GEN2_CTRL_OFF Offset address: 000480C$_H$

Link Width and Speed Change Control Register Kernel Reset value: 0002 01B4$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								SELECTABLE_DEEMPH_BIT_MUX	SELECT_DEEMPH_VAR_MUX	GEN1_EI_INFERENCE	SEL_DEEMPHASIS	CONFIG_TX_COMP_RX	Res	DIRECT_SPEED_CHANGE	AUTO_LANE_FLIP_CTRL_EN
r								rw	rw	rw	rw	rw	r	rwh	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PRE_DET_LANE			NUM_OF_LANES					FAST_TRAINING_SEQ							
rw			rw					rw							

Field	Bits	Type	Description
FAST_TRAINING_SEQ	7:0	rw	**Sets the Number of Fast Training Sequences (N_FTS) that the controller advertises as its N_FTS during Gen2 or Gen3 link training** This value is used to inform the link partner about the PHY's ability to recover synchronization after a low power state. Do not set N_FTS to zero; doing so can cause the LTSSM to go into the recovery state when exiting from L0s. Note: The access attributes of this field are as follows: - Wire: R/W (sticky) - Dbi: R/W (sticky) Note: This register field is sticky.
NUM_OF_LANES	12:8	rw	**Predetermined Number of Lanes** Defines the number of lanes which are connected and not bad. Used to limit the effective link width to ignore 'broken" or "unused" lanes that detect a receiver. Indicates the number of lanes to check for exit from Electrical Idle in Polling.Active and L2.Idle. It is possible that the LTSSM might detect a receiver on a bad or broken lane during the Detect Substate. However, it is also possible that such a lane might also fail to exit Electrical Idle and therefore prevent a valid link from being configured. This value is referred to as the "Predetermined Number of Lanes" in section 4.2.6.2.1 of the PCI Express Base Specification. Note: The access attributes of this field are as follows: - Wire: R/W (sticky) - Dbi: R/W (sticky) Note: This register field is sticky. 01_H **_1_LANE**: 1 lane 02_H **_2_LANE**: 2 lanes 03_H **_3_LANE**: 3 lanes **others,** Reserved

(table continues...)

(continued)

Field	Bits	Type	Description
PRE_DET_LANE	15:13	rw	**Predetermined Lane for Auto Flip** This field defines which physical lane is connected to logical Lane0 by the flip operation performed in Detect. This field is used to restrict the receiver detect procedure to a particular lane when the default detect and polling procedure performed on all lanes cannot be successful. A notable example of when it is useful to program this field to a value different from the default, is when a lane is asymmetrically broken, that is, it is detected in Detect LTSSM state but it cannot exit Electrical Idle in Polling LTSSM state. Note: The access attributes of this field are as follows: - Wire: R/W (sticky) - Dbi: R/W (sticky) Note: This register field is sticky. 000_B **LANE0**: Connect logical Lane0 to physical lane 0, depending on which lane is detected 001_B **LANE1**: Connect logical Lane0 to physical lane 1 010_B **LANE3**: Connect logical Lane0 to physical lane 3 011_B **LANE7**: connect logical lane0 to physical lane 7 100_B **LANE15**: Connect logical Lane0 to physical lane 15 **others,** Reserved
AUTO_LANE_FLIP_CTRL_EN	16	rw	**Enable Auto flipping of the lanes** This feature is not supported by hardware. Note: The access attributes of this field are as follows: - Wire: R/W (sticky) - Dbi: R/W (sticky) Note: This register field is sticky. 0_B **DISABLE**: Disable 1_B **ENABLE**: Enable
DIRECT_SPEED_CHANGE	17	rwh	**Directed Speed Change** When the speed change occurs, the controller will clear the contents of this field; and a read to this field by your software will return a '0'. To manually initiate the speed change: • Write to LINK_CONTROL2_LINK_STATUS2_REG.PCIE_CAP_TARGET_LINK_SPEED in the local device • Deassert this field • Assert this field If you set the default of this field using the DEFAULT_GEN2_SPEED_CHANGE configuration parameter to '1', then the speed change is initiated automatically after link up, and the controller clears the contents of this field. If you want to prevent this automatic speed change, then write a lower speed value to the Target Link Speed field of the Link Control 2 register (LINK_CONTROL2_LINK_STATUS2_OFF.PCIE_CAP_TARGET_LINK_SPEED) through the DBI before link up. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W 0_B **CLEAR**: Clear 1_B **SET**: Writing '1' to this field instructs the LTSSM to initiate a speed change to Gen2 or Gen3 after the link is initialized at Gen1 speed

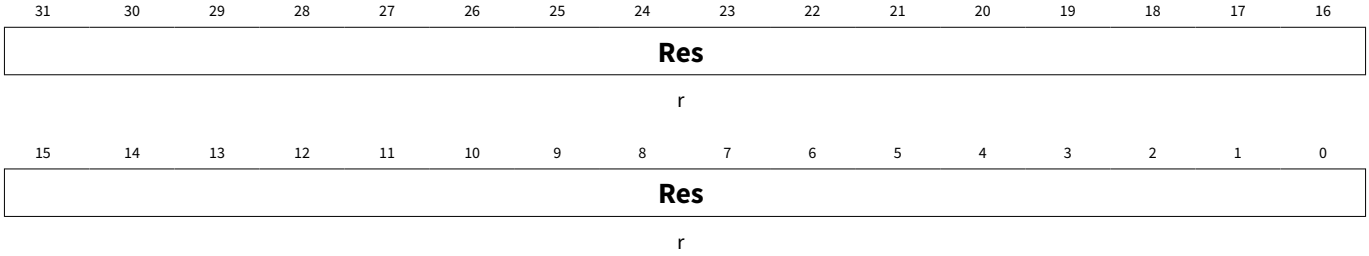
(table continues...)

(continued)

Field	Bits	Type	Description
CONFIG_TX_COMP_RX	19	rw	**Config Tx Compliance Receive Bit** Note: The access attributes of this field are as follows: - Wire: R/W (sticky) - Dbi: R/W (sticky) Note: This register field is sticky. 0_B **CLEAR**: Clear 1_B **SET**: When set to 1, signals LTSSM to transmit TS ordered sets with the compliance receive bit assert (equal to '1')
SEL_DEEMPHASIS	20	rw	**Used to set the de-emphasis level for upstream ports** This bit selects the level of de-emphasis the link operates at. Note: The access attributes of this field are as follows: - Wire: R/W (sticky) - Dbi: R/W (sticky) Note: This register field is sticky. 0_B **N_6DB**: -6 dB 1_B **N_3_5DB**: -3.5 dB
GEN1_EI_INFERENCE	21	rw	**Electrical Idle Inference Mode at Gen1 Rate** Programmable mode to determine inferred electrical idle (EI) in Recovery.Speed or Loopback.Active (as slave) state at Gen1 speed by looking for a '1' value on RxElecIdle instead of looking for a '0' on RxValid. If the PHY fails to deassert the RxValid signal in Recovery.Speed or Loopback.Active (because of corrupted EIOS for example), then EI cannot be inferred successfully in the controller by just detecting the condition RxValid=0. Note: This register field is sticky. 0_B **RXELECIDLE_TO_INFER**: Use RxElecIdle signal to infer Electrical Idle 1_B **RXVALID_TO_INFER**: Use RxValid signal to infer Electrical Idle
SELECT_DEEMPH_VAR_MUX	22	rw	**The select_deemphasis variable for DSP on entry to Recovery.RcvrCfg state is muxed between the Selectable De-emphasis field in the Link Control 2 register and the value requested by the Upstream Port in the eight consecutive TS1 Ordered Sets it received** Note: This register field is sticky. 0_B **REQUESTED_BY_USP**: The value requested by USP in Recovery.RcvrLock state through Tx TS1s from USP 1_B **FROM_LINK_CTRL_2_REG**: The value from the Selectable De-emphasis field in the Link Control 2 register
SELECTABLE_DEEMPH_BIT_MUX	23	rw	**The selectable deemphasis bit (Symbol 4 bit 6) of the transmitted TS2 Ordered Sets for DSP in Recovery.RcvrCfg state is muxed between the Selectable De-emphasis field in the Link Control 2 register and the value requested by USP in Recovery.RcvrLock state through Tx TS1s from USP** Note: This register field is sticky. 0_B **FROM_LINK_CTRL_2_REG**: The value from the Selectable De-emphasis field in the Link Control 2 register 1_B **REQUESTED_BY_USP**: The value requested by USP in Recovery.RcvrLock state through Tx TS1s from USP
0	31:24	r	**Reserved** Read as all 0's; should be written with all 0's.

19.4.1.253 PHY Status Register

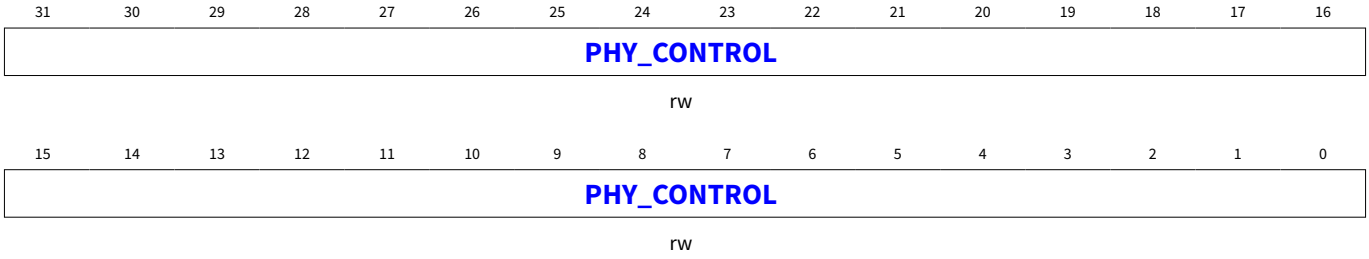
Memory mapped register from phy_cfg_status GPIO input pins.



Field	Bits	Type	Description

19.4.1.254 PHY Control Register

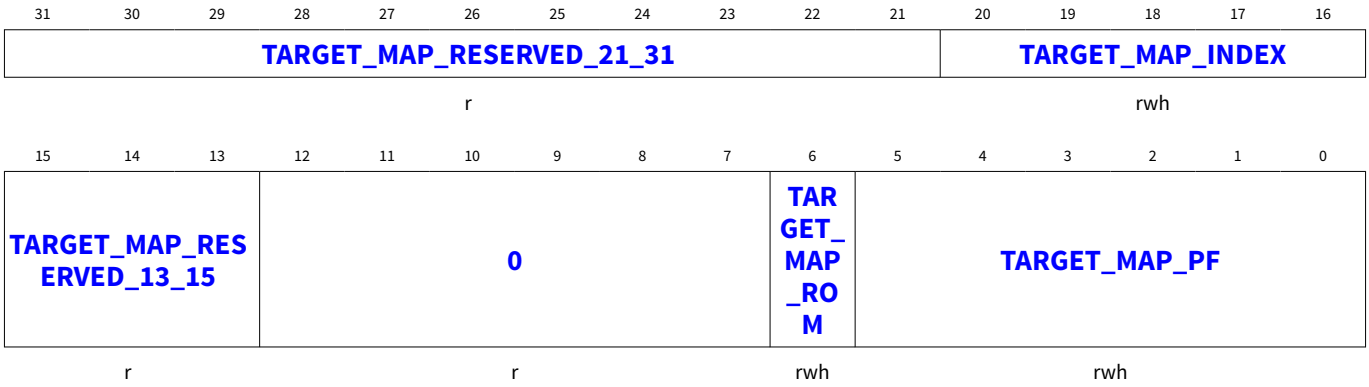
Memory mapped register to cfg_phy_control GPIO output pins.



Field	Bits	Type	Description
PHY_CONTROL	31:0	rw	**PHY Control** Data sent directly to the cfg_phy_control bus. This is a GPIO register driving the values on the static cfg_phy_control output signals, and does not in any way influence controller functionality. It can be used for any static sideband control signaling requirements that you have for your PHY. Usage of this register and the associated GPIO bus is RESERVED when the controller is connected to a Synopsys PHY. Note: This register field is sticky. 00000000_H **MIN_VAL**: Zero value $FFFFFFFF_H$ **MAX_VAL**: Max value

19.4.1.255 Programmable Target Map Control Register

This register controls the target map.



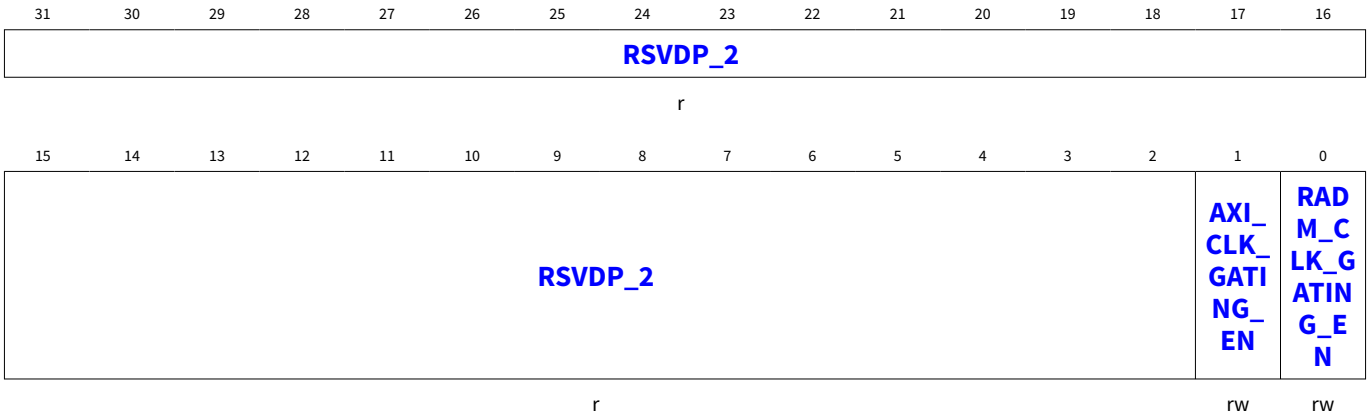
Field	Bits	Type	Description
TARGET_MAP_PF	5:0	rwh	**Target Values for each BAR on the PF Function selected by the index number** This register does not respect the Byte Enable setting, any write will affect all register bits.
TARGET_MAP_ROM	6	rwh	**Target Value for the ROM page of the PF Function selected by the index number** This register does not respect the Byte Enable setting, any write will affect all register bits.
TARGET_MAP_RESERVED_13_15	15:13	r	**Reserved** Note: The access attributes of this field are as follows: - Wire: RSVDP - Dbi: R (sticky)
TARGET_MAP_INDEX	20:16	rwh	**The number of the PF Function on which the Target Values are set** This register does not respect the Byte Enable setting, any write will affect all register bits.
TARGET_MAP_RESERVED_21_31	31:21	r	**Reserved** Note: The access attributes of this field are as follows: - Wire: RSVDP - Dbi: R (sticky)
0	12:7	r	**Reserved** Read as all 0's; should be written with all 0's.

19.4.1.256 Clock Gating Control Register

This register enables you to disable dynamic clock gating. By default dynamic clock gating is on, allowing the controller to autonomously enable and disable its clocks. The clock gating is performed in the clock and reset module, and is initiated by the controllers clock enable signals. The following modules support dynamic clock gating: - AXI Bridge - RADM

CLOCK_GATING_CTRL_OFF Offset address: $000488C_H$

Clock Gating Control Register Kernel Reset value: $0000\ 0003_H$



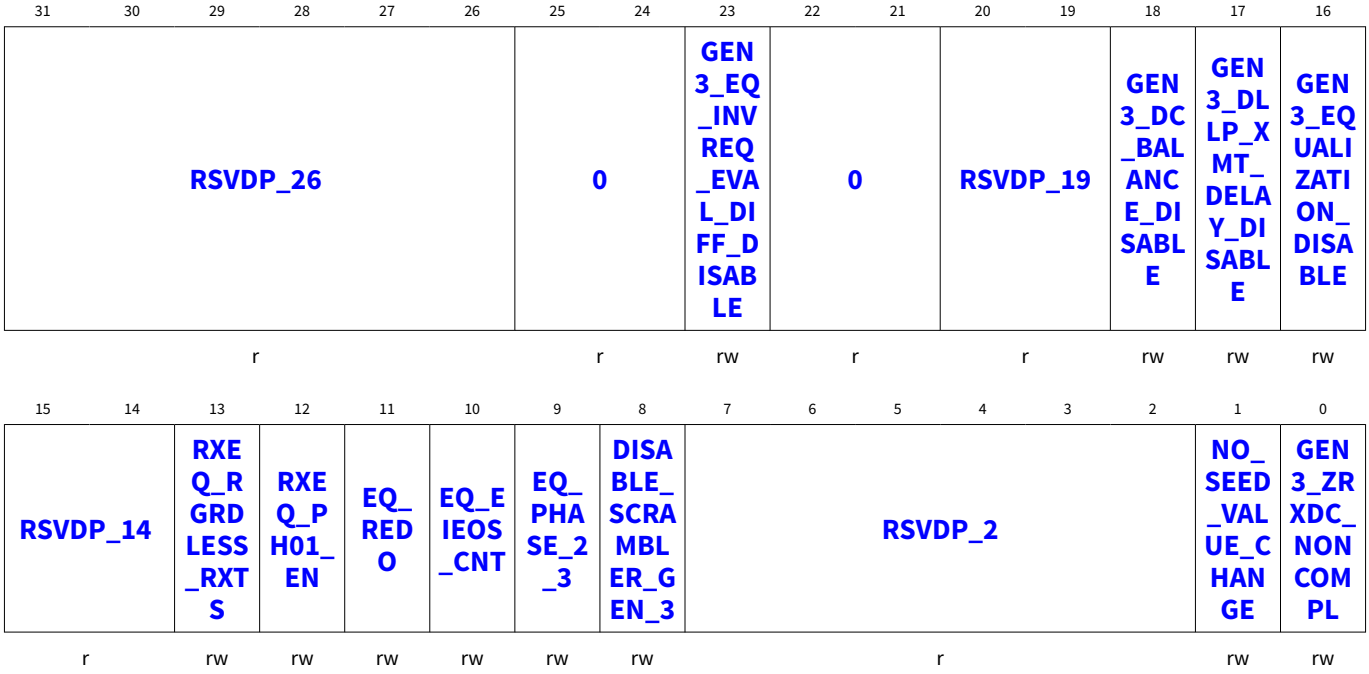
Field	Bits	Type	Description
RADM_CLK_GATING_EN	0	rw	**RADM Clock Gating Enable** This register, if set, enables the RADM to autonomously enable and disable its clock. Note: This register field is sticky. 0_B **DISABLE**: Disable 1_B **ENABLE**: Enable (default)
AXI_CLK_GATING_EN	1	rw	**AXI Clock Gating Enable** This register enables the AXI Bridge to autonomously enable and disable the AXI Master clock, the AXI Slave clock and the AXI DBI slave clock. Note: This register field is sticky. 0_B **DISABLE**: Disable 1_B **ENABLE**: Enable (default)
RSVDP_2	31:2	r	**Reserved for future use**

19.4.1.257 Gen3 Control Register

There is no Gen3-specific N_FTS field. The N_FTS field in the "Link Width and Speed Change Control Register" is used for both Gen2 and Gen3 speed modes. There is no Gen3-specific "Directed Speed Change" field. The "Directed Speed Change" field in the "Link Width and Speed Change Control Register" is used to change to Gen2 or Gen3 speed. A speed change to Gen3 occurs if (1) the "Directed Speed Change" field is set to '1' and (2) the "Target Link Speed" field in the Link Control 2 Register is set to Gen3. Gen3 support is advertised by both sides of the link during link training.

GEN3_RELATED_OFF Offset address: 0004890_H

Gen3 Control Register Kernel Reset value: $0000\ 2001_H$



Field	Bits	Type	Description
GEN3_ZRXDC_NONCOMPL	0	rw	**Gen3 Receiver Impedance ZRX-DC Not Compliant** Receivers that operate at 8.0 GT/s with an impedance other than the range defined by the ZRX-DC parameter for 2.5 GT/s (40-60 Ohms) must meet additional behavior requirements in the following LTSSM states: Polling, Rx_L0s, L1, L2, and Disabled. Note: This register field is sticky. 0_B **RCVR_COMPLIES**: The receiver complies with the ZRX-DC parameter for 2.5 GT/s when operating at 8 GT/s or higher 1_B **RCVR_NOT_COMPLIES**: The receiver does not comply with the ZRX-DC parameter for 2.5 GT/s when operating at 8 GT/s or higher
NO_SEED_VALUE_CHANGE	1	rw	**If this bit is set to 1, the seed value of LFSR for scrambler at Gen3 rate does not change after LinkUp = 1** This feature is not supported. Note: this register is shared for Gen3 and Gen4/Gen5 data rates. Note: This register field is sticky. 1_B **NO_CHANGE**: Not Change
RSVDP_2	7:2	r	**Reserved for future use**
DISABLE_SCRAMBLER_GEN_3	8	rw	**Disable Scrambler for Gen3 and Gen4 Data Rate** The Gen3 and Gen4 scrambler/descrambler within the controller needs to be disabled when the scrambling function is implemented outside of the controller (for example within the PHY). Note: This register field is sticky.
EQ_PHASE_2_3	9	rw	**Equalization Phase 2 and Phase 3 Disable** This applies to downstream ports only. Note: The access attributes of this field are as follows: - Wire: see description - Dbi: see description Note: This register field is sticky. 0_B **NO_TX_EQ_RX_EQ_PH01**: Rx equalization in phase 0/1 1_B **TX_EQ_23_RX_EQ_PH01**: No Rx equalization

(table continues...)

(continued)

Field	Bits	Type	Description
EQ_EIEOS_CNT	10	rw	**Equalization EIEOS Count Reset Disable** Disable requesting reset of EIEOS count during equalization. Note: This register field is sticky. 1_B **DISABLE**: Disable
EQ_REDO	11	rw	**Equalization Redo Disable** Disable autonomous mechanism for requesting to redo the equalization process. The received presets or coefficients mismatch in Recovery.RcvrLock after Recovery EQ phases causes the EQ redo requests. If the EQ redo is infinite or you do not want eq requests and redo, setting this bit to 1 will stop the EQ requests and EQ redo so that the link can go ahead to L0 state for packet trasmissions. Note: This register field is sticky.
RXEQ_PH01_EN	12	rw	**Rx Equalization Phase 0/Phase 1 Hold Enable** When this bit is set the upstream port holds phase 0 (the downstream port holds phase 1) for 10ms. Holding phase 0 or phase 1 can be used to allow sufficient time for Rx Equalization to be performed by the PHY. This bit is used during Virtex-7 Gen3 equalization. The programmable bits [RXEQ_PH01_EN, EQ_PHASE_2_3] can be used to obtain the following variations of the equalization procedure: Note: The access attributes of this field are as follows: - Wire: see description - Dbi: see description Note: This register field is sticky. 0_B **TX_EQ_23**: Tx equalization only in phase 2/3 1_B **NO_TX_EQ_NO_RX_EQ**: No Tx equalization
RXEQ_RGRDLESS_RXTS	13	rw	**When set to '1', the controller as Gen3 EQ master asserts RxEqEval to instruct the PHY to do Rx adaptation and evaluation after a 500ns timeout from a new preset request** . Note: The access attributes of this field are as follows: - Wire: see description - Dbi: see description Note: This register field is sticky. 0_B **ASSERT_1US**: mac_phy_rxeqeval asserts after 1 µs and 2 TS1 received from remote partner 1_B **ASSERT_500NS**: mac_phy_rxeqeval asserts after 500ns regardless of TS's received or not
RSVDP_14	15:14	r	**Reserved for future use**
GEN3_EQUALIZATION_DISABLE	16	rw	**Equalization Disable** Disable equalization feature. This bit cannot be changed once the LTSSM starts link training. Note: This register field is sticky.
GEN3_DLLP_XMT_DELAY_DISABLE	17	rw	**DLLP Transmission Delay Disable** Disable delay transmission of DLLPs before equalization. Note: This register field is sticky.
GEN3_DC_BALANCE_DISABLE	18	rw	**DC Balance Disable** Disable DC Balance feature. Note: This register field is sticky.
RSVDP_19	20:19	r	**Reserved for future use**
GEN3_EQ_INVREQ_EVAL_DIFF_DISABLE	23	rw	**Eq InvalidRequest and RxEqEval Different Time Assertion Disable** Disable the assertion of Eq InvalidRequest and RxEqEval at different time. Note: This register field is sticky.

(table continues...)

(continued)

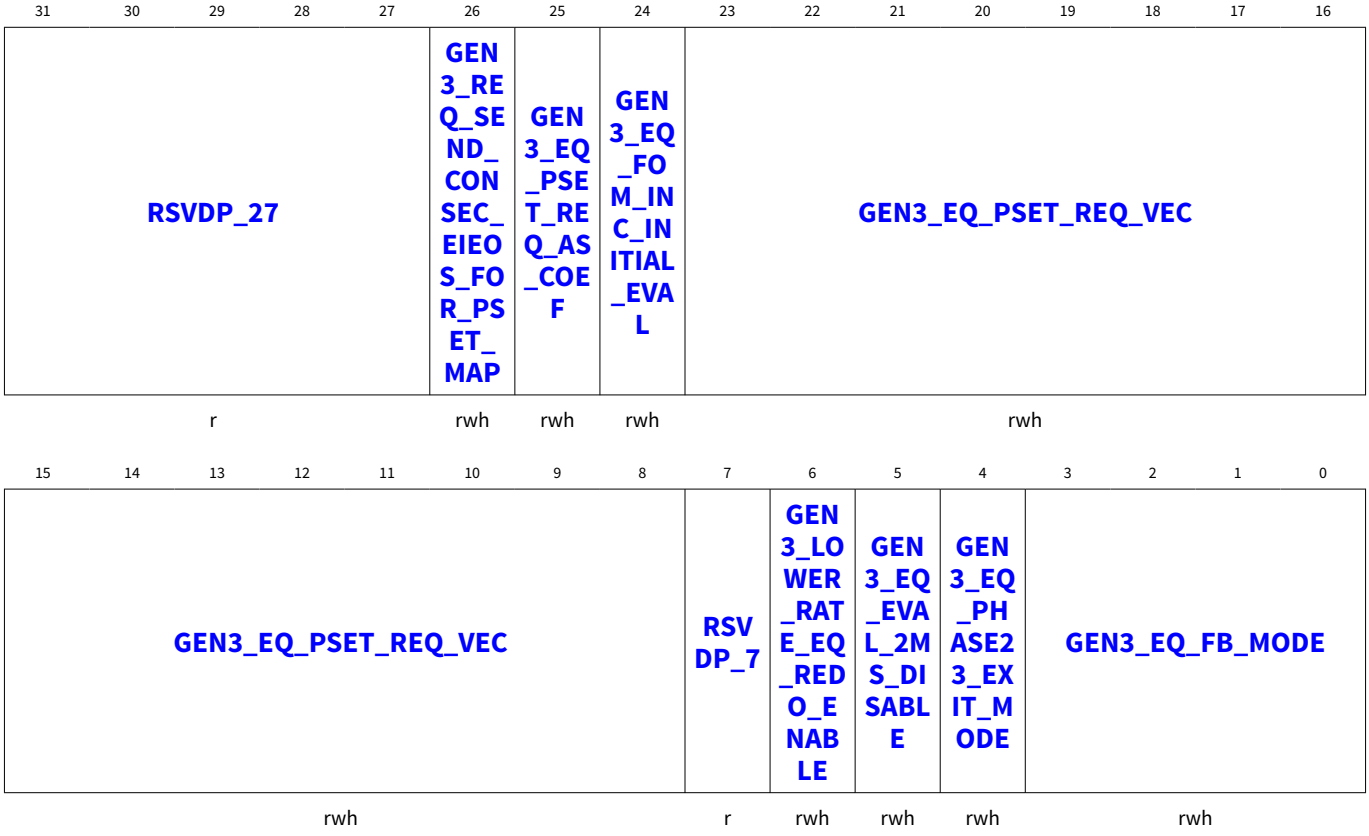
Field	Bits	Type	Description
RSVDP_26	31:26	r	**Reserved for future use**
0	22:21, 25:24	r	**Reserved** Read as all 0's; should be written with all 0's.

19.4.1.258 Gen3 EQ Control Register

This register controls equalization for Phase2 in an upstream port (USP), or Phase3 in a downstream port (DSP).

GEN3_EQ_CONTROL_OFF Offset address: $00048A8_H$

Gen3 EQ Control Register Kernel Reset value: $0501\ 7271_H$



Field	Bits	Type	Description
GEN3_EQ_FB_MODE	3:0	rwh	**Feedback Mode** Other values are reserved. Note: This register field is sticky. 0_H **DIR_CHG**: Direction Change 1_H **FOM**: Figure Of Merit **others,** Reserved

(table continues...)

(continued)

Field	Bits	Type	Description
GEN3_EQ_PHASE23_EXIT_MODE	4	rwh	**Behavior After 24 ms Timeout (when optimal settings are not found)** For a USP: Determine next LTSSM state from Phase2 after 24ms Timeout • 0: Recovery.Speed • 1: Recovery.Equalization.Phase3 When optimal settings are not found then: • Equalization Phase 2 Successful status bit is not set in the "Link Status Register 2" when GEN3_EQ_PHASE23_EXIT_MODE = 0 • Equalization Phase 2 Successful status bit is set in the "Link Status Register 2" when GEN3_EQ_PHASE23_EXIT_MODE = 1 • Equalization Phase 2 Complete status bit is set in the "Link Status Register 2" For a DSP: Determine next LTSSM state from Phase3 after 24ms Timeout • 0: Recovery.Speed • 1: Recovery.Equalization.RcvrLock When optimal settings are not found then: • Equalization Phase 3 Successful status bit is not set in the "Link Status Register 2" when GEN3_EQ_PHASE23_EXIT_MODE = 0 • Equalization Phase 3 Successful status bit is set in the "Link Status Register 2" when GEN3_EQ_PHASE23_EXIT_MODE = 1 • Equalization Phase 3 Complete status bit is set in the "Link Status Register 2" Note: GEN3_EQ_PHASE23_EXIT_MODE = 1 affects Direction Change feed back mode. EQ requests for Figure Of Merit mode complete before 24 ms timeout. Please see GEN3_EQ_PSET_REQ_VEC Register for more. Note: This register field is sticky. 0_B **RCVRY_SPEED**: Recovery.Speed 1_B **RCVRY_EQ**: USP: Recovery.Equalization.Phase3; DSP: Recovery.Equalization.RcvrLock
GEN3_EQ_EVAL_2MS_DISABLE	5	rwh	**Phase2_3 2 ms Timeout Disable** Determine behavior in Phase2 for USP (Phase3 if DSP) when the PHY does not respond within 2ms to the assertion of RxEqEval. Note: This register field is sticky. 0_B **ABORT_CURRENT_EVA**: Abort the current evaluation, stop any attempt to modify the remote transmitter settings, Phase2 is terminated by the 24ms timeout 1_B **IGNORE_2MS_TIMEOUT**: Ignore the 2ms timeout and continue as normal This is used to support PHYs that require more than 2ms to respond to the assertion of RxEqEval.
GEN3_LOWER_RATE_EQ_REDO_ENABLE	6	rwh	**Support EQ redo and lower rate change** To access this field, RATE_SHADOW_SEL should be set to 0. Note: This register field is sticky. 0_B **NO_SUP**: Not supported 1_B **SUP**: Supported
RSVDP_7	7	r	**Reserved for future use**

(table continues...)

(continued)

Field	Bits	Type	Description
GEN3_EQ_PSET_REQ_VEC	23:8	rwh	**Preset Request Vector** Requesting of Presets during the initial part of the EQ Master Phase. Encoding scheme is as follows: Bit [15:0] =0x0: No preset is requested and evaluated in EQ Master Phase. Bit [i] =1: "Preset=i" is requested and evaluated in EQ Master Phase. • 0000000000000000: No preset be requested and evaluated in EQ Master Phase • 000000xxxxxxxxx1: Preset 0 is requested and evaluated in EQ Master Phase • 000000xxxxxxxx1x: Preset 1 is requested and evaluated in EQ Master Phase • 000000xxxxxxx1xx: Preset 2 is requested and evaluated in EQ Master Phase • 000000xxxxxx1xxx: Preset 3 is requested and evaluated in EQ Master Phase • 000000xxxxx1xxxx: Preset 4 is requested and evaluated in EQ Master Phase • 000000xxxx1xxxxx: Preset 5 is requested and evaluated in EQ Master Phase • 000000xxx1xxxxxx: Preset 6 is requested and evaluated in EQ Master Phase • 000000xx1xxxxxxx: Preset 7 is requested and evaluated in EQ Master Phase • 000000x1xxxxxxxx: Preset 8 is requested and evaluated in EQ Master Phase • 00000x1xxxxxxxxx: Preset 9 is requested and evaluated in EQ Master Phase • 000001xxxxxxxxxx: Preset 10 is requested and evaluated in EQ Master Phase • All other encodings: Reserved Note: This register field is sticky.
GEN3_EQ_FOM_INC_INITIAL_EVAL	24	rwh	**Include Initial FOM** Include or not the FOM feedback from the initial preset evaluation performed in the EQ Master, when finding the highest FOM among all preset evaluations. Note: This register field is sticky. 0_B **DO_NOT_INCLUDE**: Do not include 1_B **INCLUDE**: Include
GEN3_EQ_PSET_REQ_AS_COEF	25	rwh	**GEN3_EQ_PSET_REQ_AS_COEF is an internally reserved field** Do not use. Note: This register field is sticky.

(table continues...)

(continued)

Field	Bits	Type	Description
GEN3_REQ_SEND_CONSEC_EIEOS_FOR_PSET_MAP	26	rwh	**Request controller to send back-to-back EIEOS in Recovery.RcvrLock state until presets to coefficients mapping is complete** Note: Gen3 and Gen4 share the same register bit and have the same feature. Note: This register field is sticky. 0_B **DO_NOT_REQ**: Do not request 1_B **REQ**: Request
RSVDP_27	31:27	r	**Reserved for future use**

19.4.1.259 Gen3 EQ Direction Change Feedback Mode Control Register

Equalization controls to be used in Phase2 (USP) or Phase 3 (DSP), when you set the Feedback Mode in "Gen3 EQ Control Register" to "Direction Change." These fields allow control over the initial starting point for the search of optimal coefficient settings, and allow control over the criteria used to determine when the optimal settings have been achieved. The values are applied to all the lanes.

GEN3_EQ_FB_MODE_DIR_CHANGE_OFF — Offset address: $00048AC_H$

Gen3 EQ Direction Change Feedback Mode Control Register — Kernel Reset value: $0000\ 0040_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_18														GEN3_EQ_FMDC_MAX_POST_CUSROR_DELTA	
r														rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GEN3_EQ_FMDC_MAX_POST_CUSROR_DELTA		GEN3_EQ_FMDC_MAX_PRE_CUSROR_DELTA				GEN3_EQ_FMDC_N_EVALS					GEN3_EQ_FMDC_T_MIN_PHASE23				
rw		rw				rw					rw				

Field	Bits	Type	Description
GEN3_EQ_FMDC_T_MIN_PHASE23	4:0	rw	**Minimum Time (in ms) To Remain in EQ Master Phase** The LTSSM stays in EQ Master phase for at least this amount of time, before starting to check for convergence of the coefficients. Allowed values 0,1,...,24. Note: This register field is sticky. 00_H **MIN_VAL**: Min value 18_H **MAX_VAL**: Max value **others,** Reserved

(table continues...)

(continued)

Field	Bits	Type	Description
GEN3_EQ_FMDC_N_EVALS	9:5	rw	**Convergence Window Depth** Number of consecutive evaluations considered in Phase 2/3 when determining if optimal coefficients have been found. Allowed range: 0,1,2,.. up to 8. When set to 0, EQ Master is performed without sending any requests to the remote partner in Phase 2 for USP and Phase 3 for DSP. Therefore, the remote partner will not change its transmitter coefficients and will move to the next state. Note: This register field is sticky. 00_H **MIN_VAL**: Min value $1F_H$ **MAX_VAL**: Maximum of CX_GEN3_EQ_COEFQ_DEPTH
GEN3_EQ_FMDC_MAX_PRE_CUSROR_DELTA	13:10	rw	**Convergence Window Aperture for C-1** Pre-cursor coefficients maximum delta within the convergence window depth. Allowed range: 0,1,2,..15. Note: This register field is sticky. 0_H **MIN_VAL**: Min value F_H **MAX_VAL**: Max value
GEN3_EQ_FMDC_MAX_POST_CUSROR_DELTA	17:14	rw	**Convergence Window Aperture for C+1** Post-cursor coefficients maximum delta within the convergence window depth. Allowed range: 0,1,2,..15. Note: This register field is sticky. 0_H **MIN_VAL**: Min value F_H **MAX_VAL**: Max value
RSVDP_18	31:18	r	**Reserved for future use**

19.4.1.260 Order Rule Control Register

This register controls the order rule.

ORDER_RULE_CTRL_OFF Offset address: $00048B4_H$

Order Rule Control Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_16															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CPL_PASS_P								NP_PASS_P							
rw								rw							

Field	Bits	Type	Description
NP_PASS_P	7:0	rw	**Non-Posted Passing Posted Ordering Rule Control** Determines if NP can pass halted P queue. 00_H **NP_CAN_NOT_PASS**: NP can not pass P (recommended) 01_H **NP_CAN_PASS**: NP can pass P **others,** Reserved

(table continues...)

(continued)

Field	Bits	Type	Description
CPL_PASS_P	15:8	rw	**Completion Passing Posted Ordering Rule Control** Determines if CPL can pass halted P queue. 00_H **CPL_CAN_NOT_PASS**: CPL can not pass P (recommended) 01_H **CPL_CAN_PASS**: CPL can pass P **others,** Reserved
RSVDP_16	31:16	r	**Reserved for future use**

19.4.1.261 PIPE Loopback Control Register

This register controls the PIPE Loopback.

PIPE_LOOPBACK_CONTROL_OFF Offset address: $00048B8_H$

PIPE Loopback Control Register Reset values see: Table 875

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PIPE_LOOPBACK	RSVDP_27				RXSTATUS_VALUE			RSVDP_22		RXSTATUS_LANE					
rw	r				rwh			r		rw					

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LPBK_RXVALID															
rw															

Field	Bits	Type	Description
LPBK_RXVALID	15:0	rw	**LPBK_RXVALID is an internally reserved field** Do not use. Note: This register field is sticky.
RXSTATUS_LANE	21:16	rw	**RXSTATUS_LANE is an internally reserved field** Do not use. Note: This register field is sticky.
RSVDP_22	23:22	r	**Reserved for future use**
RXSTATUS_VALUE	26:24	rwh	**RXSTATUS_VALUE is an internally reserved field** Do not use.
RSVDP_27	30:27	r	**Reserved for future use**
PIPE_LOOPBACK	31	rw	**PIPE Loopback Enable** Indicates RMMI Loopback if M-PCIe. Note: This register field is sticky.

Table 875 Reset values of PIPE_LOOPBACK_CONTROL_OFF

Reset	Reset value	Note
Kernel Reset	0000 0XXX 0000 0000 0000 0000 0000 0001_B	

19.4.1.262 DBI Read-Only Write Enable Register

This is the DBI Read-Only write enable register.

MISC_CONTROL_1_OFF Offset address: $00048BC_H$

DBI Read-Only Write Enable Register Kernel Reset value: $0047\ FF41_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_23									PORT_LOGIC_WR_DISABLE	P2P_ERR_RPT_CTRL	P2P_TRACK_CPL_TO_REG	TARGET_ABOVE_CONFIG_LIMIT_REG		CONFIG_LIMIT_REG	
r									rw	rw	rw	rw		rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CONFIG_LIMIT_REG								CFG_TLP_BYPASS_EN_REG	CPLQ_MNG_EN	ARI_DEVICE_NUMBER	DISABLE_AUTO_LTR_CLR_MSG	SIMPLIFIED_REPLAY_TIMER	UR_CA_MASK_4_TRGT1	DEFAULT_TARGET	DBI_RO_WR_EN
rw								rw	rw	rw	rw	rw	rw	rw	rwh

Field	Bits	Type	Description
DBI_RO_WR_EN	0	rwh	**Write to RO Registers Using DBI** This bit field protects the access to the registers to avoid intended modifications.. Note: This register field is sticky. 0_B **CLEAR**: Clear 1_B **SET**: Your application can write to some RO and HwInit register fields through the DBI when you set this field to '1'
DEFAULT_TARGET	1	rw	**Default target for an IO or MEM request with UR/CA/CRS received** Based on the value of this field the controller either drops or forwards these requests to your application. Note: This register field is sticky. 0_B **DROP_IO_MEM_REQ**: The controller drops all incoming I/O or MEM requests (after corresponding error reporting) A completion with UR status is generated for non-posted requests. 1_B **FWD_IO_MEM_UR_CA_CRS**: The controller forwards all incoming I/O or MEM requests with UR/CA/CRS status to your application
UR_CA_MASK_4_TRGT1	2	rw	**When this field is set to '1', the controller suppresses error logging, error message generation, and CPL generation for non-posted requests TLPs (with UR filtering status) forwarded to your application (that is, when DEFAULT_TARGET =1)** Note: This register field is sticky. 0_B **CLEAR**: Clear 1_B **SET**: Set

(table continues...)

(continued)

Field	Bits	Type	Description
SIMPLIFIED_REPLAY_TIMER	3	rw	**Enables Simplified Replay Timer (Gen4)** Simplified Replay Timer can have the following Values: • A value from 24,000 to 31,000 Symbol Times when Extended Synch is 0b • A value from 80,000 to 100,000 Symbol Times when Extended Synch is 1b The Simplified Replay Timer value must not be changed while the link is in use. Note: This register field is sticky. 0_B **DISABLE**: Disable 1_B **ENABLE**: Enable
DISABLE_AUTO_LTR_CLR_MSG	4	rw	**Disable the autonomous generation of LTR clear message in upstream port** This field can have the following values. Note: This register field is sticky. 0_B **ALLOW**: Allow the autonomous generation of LTR clear message 1_B **DISABLE**: Disable the autonomous generation of LTR clear message
ARI_DEVICE_NUMBER	5	rw	**When ARI is enabled, this field enables use of the device ID** Note: This register field is sticky. ARI capability is not available in this implementation 0_B **DISABLE**: Disable 1_B **ENABLE**: Enable
CPLQ_MNG_EN	6	rw	**This field enables the Completion Queue Management feature** Note: This register field is sticky. 0_B **DISABLE**: Disable 1_B **ENABLE**: Enable
CFG_TLP_BYPASS_EN_REG	7	rw	**Setting of this field defines how to decide the destination of Configuration requests** Note: When LNK_CORECTRL.RQRTYEN is asserted, the setting of this field is ignored. Note: This register field is sticky. 0_B **ZERO**: Configuration TLPs are routed according to the setting of TARGET_ABOVE_CONFIG_LIMIT_REG, depending on the setting of CONFIG_LIMIT_REG Refer to the definition of CONFIG_LIMIT_REG for details. 1_B **ONE**: Configuration TLPs are routed according to the setting of TARGET_ABOVE_CONFIG_LIMIT_REG, regardless the value of CONFIG_LIMIT_REG

(table continues...)

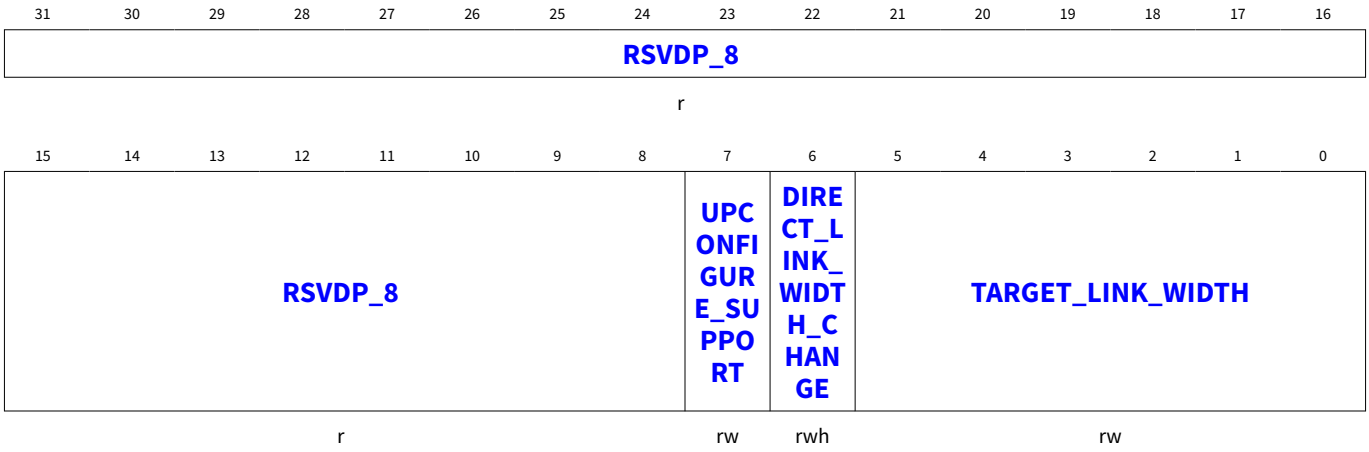
(continued)

Field	Bits	Type	Description
CONFIG_LIMIT_REG	17:8	rw	**Configuration requests are directed either to CDM or ELBI/RTRGT1 based on the value of this field** • Configuration requests with an address less CONFIG_LIMIT_REG are directed to the CDM • Configuration requests with an address greater than CONFIG_LIMIT_REG are directed to either ELBI or TRGT1 interface based on the setting of TARGET_ABOVE_CONFIG_LIMIT_REG field Your application must set a proper value for this field based on your extended configuration registers. Note: This register field is sticky.
TARGET_ABOVE_CONFIG_LIMIT_REG	19:18	rw	**Configuration requests with an address greater than CONFIG_LIMIT_REG are directed to either ELBI or TRGT1 interface based on the setting of this field** This field can have the following values: Note: This register field is sticky. 01_B **ELBI**: ELBI 10_B **TRGT1**: TRGT1 **others,** Reserved
P2P_TRACK_CPL_TO_REG	20	rw	**Determines whether to track completion of transmitted Non-Posted TLPs in P2P mode** Note: This register field is sticky. 0_B **DO_NOT_TRACk**: Do not track completion 1_B **TRACK**: Track completion
P2P_ERR_RPT_CTRL	21	rw	**Determines whether to enable Peer to Peer (P2P) error reporting** Note: This register field is sticky. 0_B **DISABLE**: Disable P2P error reporting 1_B **ENABLE**: Enable P2P error reporting
PORT_LOGIC_WR_DISABLE	22	rw	**Disable port logic register write from wire side** Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: R/W (sticky) Note: This register field is sticky.
RSVDP_23	31:23	r	**Reserved for future use**

19.4.1.263 UpConfigure Multi-lane Control Register

Used when upsizing or downsizing the link width through Configuration state without bringing the link down.

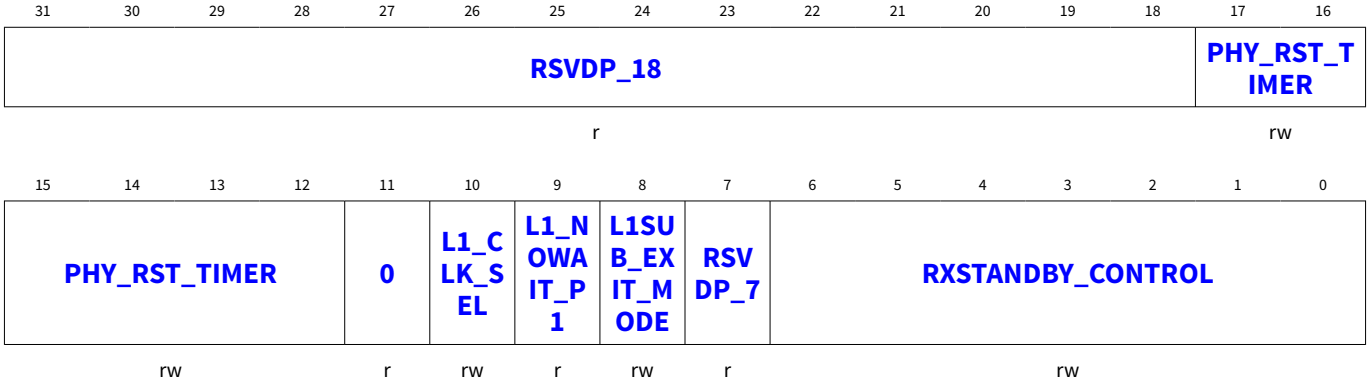
19 Peripheral Component Interconnect Express (PCIe)



Field	Bits	Type	Description
TARGET_LINK_WIDTH	5:0	rw	**Target Link Width** Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W 00_H **NOT_START**: Controller does not start upconfigure or autonomous width downsizing in the Configuration state 01_H **X1**: x1 02_H **X2**: x2 04_H **X4**: x4 08_H **X8**: x8 10_H **X16**: x16 20_H **X32**: x32 **others,** Reserved
DIRECT_LINK_WIDTH_CHANGE	6	rwh	**Directed Link Width Change** • If the upconfigure_capable variable is '1' and the PCIE_CAP_HW_AUTO_WIDTH_DISABLE bit in LINK_CONTROL_LINK_STATUS_REG is '0', the controller starts upconfigure or autonomous width downsizing (to the TARGET_LINK_WIDTH value) in the Configuration state • If TARGET_LINK_WIDTH value is 0x0, the controller does not start upconfigure or autonomous width downsizing in the Configuration state The controller self-clears this field when the controller accepts this request. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W 0_B **CLEAR**: Clear 1_B **SET**: The controller always moves to Configuration state through Recovery state when this bit is set to '1'
UPCONFIGURE_SUPPORT	7	rw	**Upconfigure Support** The controller sends this value as the Link Upconfigure Capability in TS2 Ordered Sets in Configuration.Complete state. Note: The access attributes of this field are as follows: - Wire: R/W (sticky) - Dbi: R/W (sticky) Note: This register field is sticky.
RSVDP_8	31:8	r	**Reserved for future use**

19.4.1.264 PHY Interoperability Control Register

This register controls the PHY interoperability.



Field	Bits	Type	Description
RXSTANDBY_CONTROL	6:0	rw	**Rxstandby Control** Bits 0..5 determine if the controller asserts the RxStandby signal (mac_phy_rxstandby) in the indicated condition. Bit 6 enables the controller to perform the RxStandby/RxStandbyStatus handshake. This field is reserved for internal use. You should not write to this field and change the default unless specifically instructed by Synopsys support. • [0]: Rx EIOS and subsequent T TX-IDLE-MIN • [1]: Rate Change • [2]: Inactive lane for upconfigure/downconfigure • [3]: PowerDown=P1orP2 • [4]: RxL0s.Idle • [5]: EI Infer in L0 • [6]: Execute RxStandby/RxStandbyStatus Handshake Note: This register field is sticky.
RSVDP_7	7	r	**Reserved for future use**
L1SUB_EXIT_MODE	8	rw	**L1 Exit Control Using phy_mac_pclkack_n** This field is reserved for internal use. You should not write to this field and change the default unless specifically instructed by Synopsys support. If PCLK as PHY input is selected, you should not write to this field and change the default value since PCLK as PHY input doesn't support any value other than the default. Note: This register field is sticky. 0_B **WAIT_FOR_PCLKACK**: Controller waits for the PHY to assert phy_mac_pclkack_n before exiting L1 1_B **NO_WAIT_FOR_PCLKACK**: Controller exits L1 without waiting for the PHY to assert phy_mac_pclkack_n

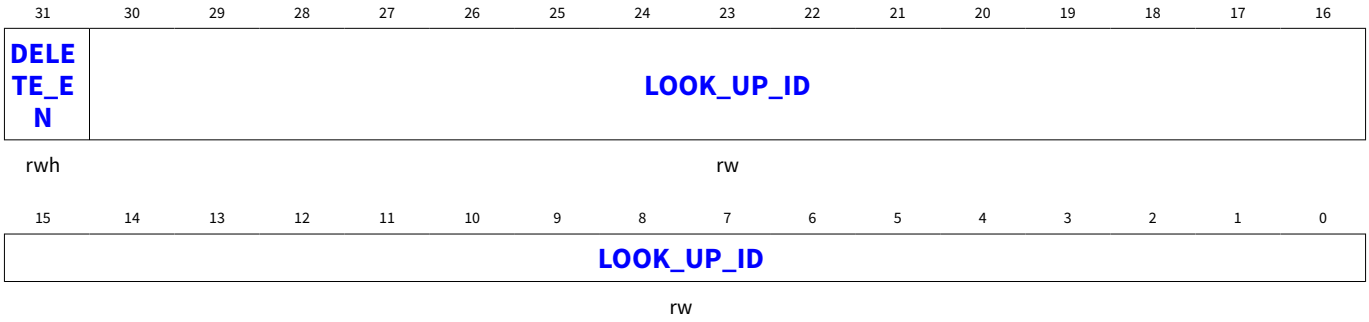
(table continues...)

(continued)

Field	Bits	Type	Description
L1_NOWAIT_P1	9	r	**L1 entry control bit** This field is reserved for internal use. You should not write to this field and change the default unless specifically instructed by Synopsys support. Note: The access attributes of this field are as follows: - Wire: R/W (sticky) - Dbi: R/W (sticky) Note: This register field is sticky. 0_B **WAIT_FOR_ACK**: Controller waits for the PHY to acknowledge transition to P1 before entering L1 1_B **NO_WAIT_FOR_ACK**: Controller does not wait for PHY to acknowledge transition to P1 before entering L1
L1_CLK_SEL	10	rw	**L1 Clock control bit** This field is reserved for internal use. You should not write to this field and change the default unless specifically instructed by Synopsys support. Note: This register field is sticky. 0_B **CAN_REQ**: Controller requests aux_clk switch and core_clk gating in L1 1_B **NO_REQ**: Controller does not request aux_clk switch and core_clk gating in L1
PHY_RST_TIMER	17:12	rw	**Decide how many aux clock cycles the PHY reset lasts (0 to 63 aux clock cycles)** Note: This register field is sticky. 00_H **MIN_VAL**: Zero value $3F_H$ **MAX_VAL**: Max value
RSVDP_18	31:18	r	**Reserved for future use**
0	11	r	**Reserved** Read as 0; should be written with 0.

19.4.1.265 TRGT_CPL_LUT Delete Entry Control register

Using this register you can delete one entry in the target completion LUT. You should only use this register when you know that your application will never send the completion because of an FLR or any other reason. Note:: The target completion LUT (and associated target completion timeout event) is watching for completions (from your application on XALI0/1/2 or AXI master read channel) corresponding to previously received non-posted requests from the PCIe wire.



Field	Bits	Type	Description
LOOK_UP_ID	30:0	rw	**This number selects one entry to delete of the TRGT_CPL_LUT**

(table continues...)

(continued)

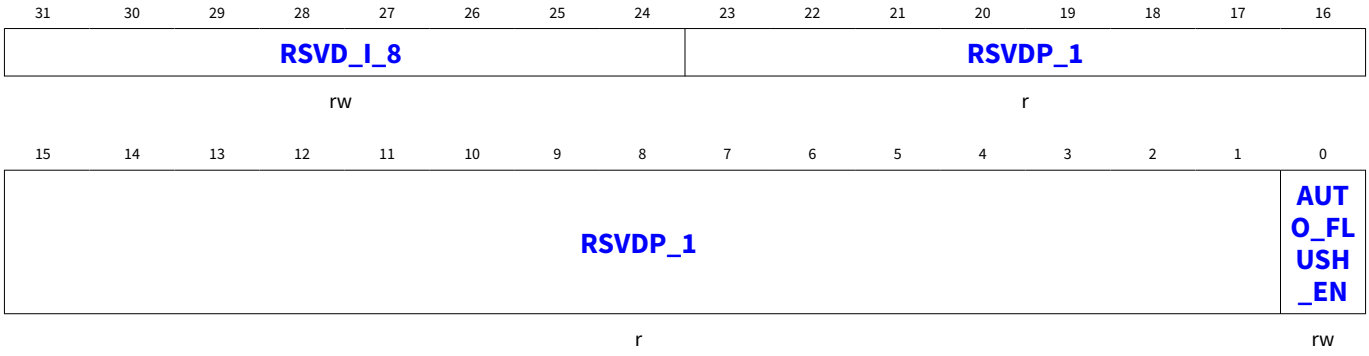
Field	Bits	Type	Description
DELETE_EN	31	rwh	**This is a one-shot bit** This is a self-clearing register field. Reading from this register field always returns a '0'. 0_B **CLEAR**: Clear 1_B **SET**: A '1' write to this bit triggers the deletion of the target completion LUT entry that is specified in the LOOK_UP_ID field

19.4.1.266 Link Reset Request Flush Control Register

This register controls link reset request flush behaviour.

LINK_FLUSH_CONTROL_OFF Offset address: $00048CC_H$

Link Reset Request Flush Control Register Kernel Reset value: $FF00\ 0001_H$



Field	Bits	Type	Description
AUTO_FLUSH_EN	0	rw	**Enables automatic flushing of pending requests before sending the reset request to the application logic to reset the PCIe controller and the AXI Bridge** The flushing process is initiated if any of the following events occur: • Hot reset request. A downstream port (DSP) can "hot reset" an upstream port (USP) by sending two consecutive TS1 ordered sets with the hot reset bit asserted • Warm (Soft) reset request. Generated when exiting from D3 to D0 and CFG_INFO.NOSOFTRST=0 • Link down reset request. A high to low transition on LNK_STAT.LINKRQRST indicates the link has gone down and the controller is requesting a reset If you disable automatic flushing, your application is responsible for resetting the PCIe controller and the AXI Bridge. Note: This register field is sticky. 0_B **DISABLE**: Disable 1_B **ENABLE**: Enable
RSVDP_1	23:1	r	**Reserved for future use**
RSVD_I_8	31:24	rw	**This is an internally reserved field** Do not use. Note: This register field is sticky.

19.4.1.267 AXI Bridge Slave Error Response Register

This register holds the AXI bridge slave error responses.

AMBA_ERROR_RESPONSE_DEFAULT_OFF Offset address: $00048D0_H$

AXI Bridge Slave Error Response Register Kernel Reset value: $0000\ 9C00_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_16															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AMBA_ERROR_RESPONSE_MAP						RSVDP_5					AMBA_ERROR_RESPONSE_CRS		AMBA_ERROR_RESPONSE_VENDORID	RSVDP_1	AMBA_ERROR_RESPONSE_GLOBAL
rw						r					rw		rw	r	rw

Field	Bits	Type	Description
AMBA_ERROR_RESPONSE_GLOBAL	0	rw	**Global Slave Error Response Mapping** Determines the AXI slave response for all error scenarios on non-posted requests. Note: This register field is sticky. 0_B **OK_**: OKAY (with FFFF data for non-posted requests) 1_B **ERR_**: SLVERR/DECERR (the AMBA_ERROR_RESPONSE_MAP field determines the PCIe-to-AXI Slave error response mapping)
RSVDP_1	1	r	**Reserved for future use**
AMBA_ERROR_RESPONSE_VENDORID	2	rw	**Vendor ID Non-existent Slave Error Response Mapping** Determines the AXI slave response for errors on reads to non-existent Vendor ID register. Note: This register field is sticky. 0_B **_OK_**: OKAY (with FFFF data) 1_B **_ERR_**: SLVERR/DECERR (the AMBA_ERROR_RESPONSE_MAP field determines the PCIe-to-AXI Slave error response mapping)
AMBA_ERROR_RESPONSE_CRS	4:3	rw	**CRS Slave Error Response Mapping** Determines the AXI slave response for CRS completions. Note: This register field is sticky. 00_B **OK_**: OKAY 01_B **OK_CRS_CMPL**: OKAY with all FFFF_FFFF data for all CRS completions 10_B **OK_READ_REQ**: OKAY with FFFF_0001 data for CRS completions to vendor ID read requests, OKAY with FFFF_FFFF data for all other CRS completions 11_B **SLVERR_DECERR**: SLVERR/DECERR (the AMBA_ERROR_RESPONSE_MAP field determines the PCIe-to-AXI Slave error response mapping)
RSVDP_5	9:5	r	**Reserved for future use**

(table continues...)

(continued)

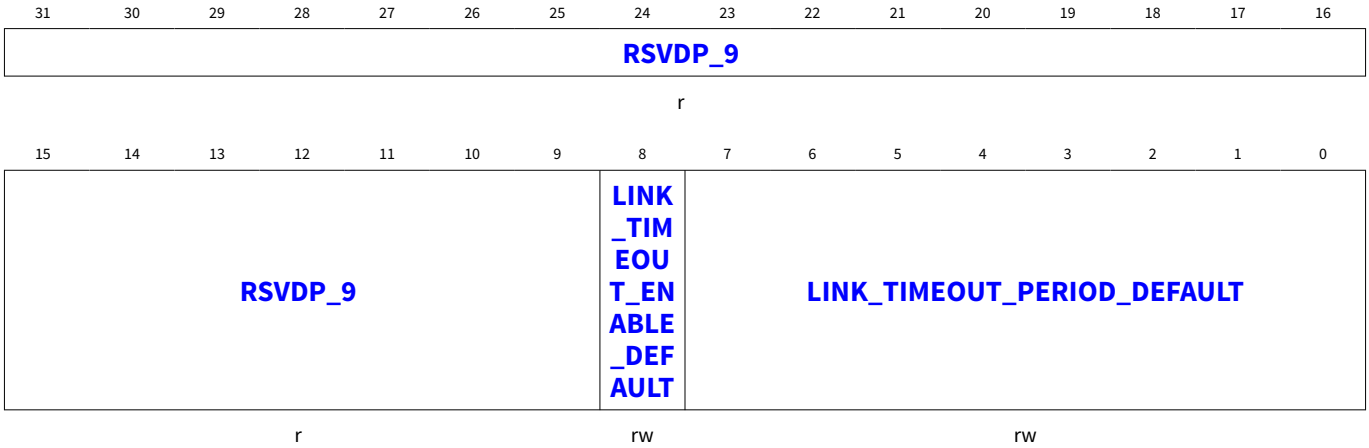
Field	Bits	Type	Description
AMBA_ERROR_RESPONSE_MAP	15:10	rw	**AXI Slave Response Error Map** Allows you to selectively map the errors received from the PCIe completion (for non-posted requests) to the AXI slave responses, slv_rresp or slv_bresp. The recommended setting is SLVERR. CRS is always mapped to OKAY. • [0] -- 0: UR (unsupported request) -> DECERR -- 1: UR (unsupported request) -> SLVERR • [1] -- 0: CRS (configuration retry status) -> DECERR -- 1: CRS (configuration retry status) -> SLVERR • [2] -- 0: CA (completer abort) -> DECERR -- 1: CA (completer abort) -> SLVERR • [3]: RESERVED (0x0) • [4]: RESERVED (0x0) • [5] -- 0: Completion Timeout -> DECERR -- 1: Completion Timeout -> SLVERR. The AXI bridge internally drops (processes internally but not passed to your application) a completion that has been marked by the Rx filter as UC or MLF, and does not pass its status directly down to the slave interface. It waits for a timeout and then signals "Completion Timeout" to the slave interface. The controller sets the AXI slave read databus to 0xFFFF for all error responses. Note: This register field is sticky.
RSVDP_16	31:16	r	**Reserved for future use**

19.4.1.268 Link Down AXI Bridge Slave Timeout Register

If your application issues outbound requests to the AXI bridge slave interface before the PCIe link is operational, the controller starts a "flush" timer. The timeout value of the timer is set by this register. If the timer times out before the PCIe link is operational, the bridge TX request queues are flushed.

AMBA_LINK_TIMEOUT_OFF Offset address: $00048D4_H$

Link Down AXI Bridge Slave Timeout Register Kernel Reset value: $0000\ 0032_H$



Field	Bits	Type	Description
LINK_TIMEOUT_PERIOD_DEFAULT	7:0	rw	**Timeout Value (ms)** The timer will timeout and then flush the bridge TX request queues after this amount of time. The timer counts when there are pending outbound AXI slave interface requests and the PCIe TX link is not transmitting any of these requests. The timer is clocked by core_clk. Note: This register field is sticky.
LINK_TIMEOUT_ENABLE_DEFAULT	8	rw	**Disable Flush** Note: This register field is sticky. 0_B **ENABLE**: Enable 1_B **DISABLE**: You can disable the flush feature by setting this field to '1'
RSVDP_9	31:9	r	**Reserved for future use**

19.4.1.269 AXI Bridge Ordering Control

This register controls the AXI Bridge Ordering.

AMBA_ORDERING_CTRL_OFF Offset address: $00048D8_H$

AXI Bridge Ordering Control Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_8															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVDP_8								AX_MSTR_ZEROLREAD_FW	RSVDP_5		AX_MSTR_ORDR_P_EVENT_SEL		RSVDP_2	AX_SNP_EN	RSVDP_0
r								rw	r		rw		r	rw	r

Field	Bits	Type	Description
RSVDP_0	0	r	**Reserved for future use**
AX_SNP_EN	1	rw	**AXI Serialize Non-Posted Requests Enable** This field enables the AXI Bridge to serialize same ID Non-Posted Read/Write Requests on the wire. Serialization implies one outstanding same ID NP Read or Write on the wire and used to avoid AXI RAR and WAW hazards at the remote link partner. 0_B **DISABLE**: Disable 1_B **ENABLE**: Enable
RSVDP_2	2	r	**Reserved for future use**

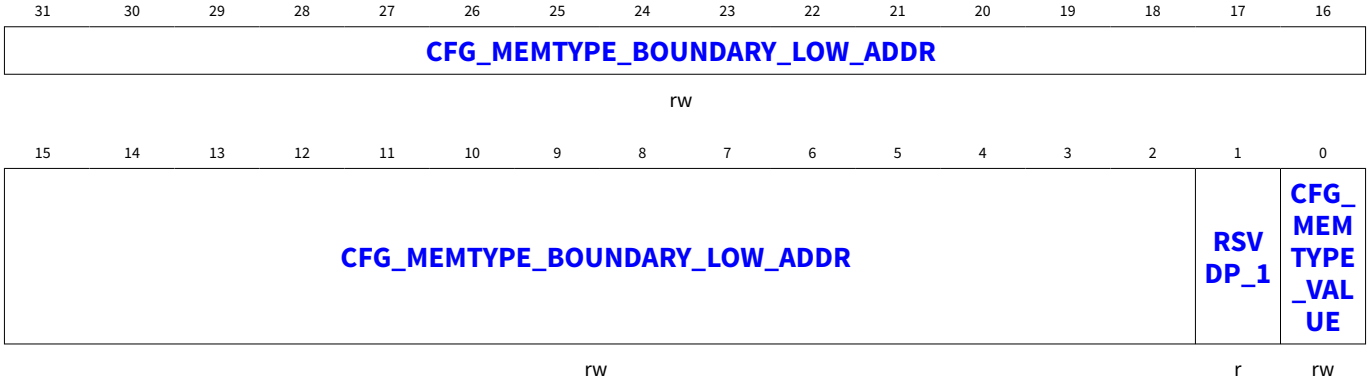
(table continues...)

(continued)

Field	Bits	Type	Description
AX_MSTR_ORDR_P_EVENT_SEL	4:3	rw	**AXI Master Posted Ordering Event Selector** This field selects how the master interface determines when a P write is completed when enforcing the PCIe ordering rule, "NP must not pass P" at the AXI Master Interface. The AXI protocol does not support ordering between channels. Therefore, NP reads can pass P on your AXI bus fabric. This can result in an ordering violation when the read overtakes a P that is going to the same address. Therefore, the bridge master does not issue any NP requests until all outstanding P writes reach their destination. It does this by waiting for the all of the write responses on the B channel. This can affect the performance of the master read channel. For scenarios where the interconnect serializes the AXI master "AW", "W" and "AR" channels,you can increase the performance by reducing the need to wait until the complete Posted transaction has effectively reached the application slave. Note: This setting will not affect: • MSI interrupt catcher and P data ordering. This is always driven by the B'last event • DMA read engine TLP ordering. This is always driven by the B'last event • NP write transactions which are always serialized with P write transactions 00_B **B**: B'last event: wait for the all of the write responses on the B channel thereby ensuring that the complete Posted transaction has effectively reached the application slave (default) 01_B **AW**: AW'last event: wait until the complete Posted transaction has left the AXI address channel at the bridge master 10_B **W**: W'last event: wait until the complete Posted transaction has left the AXI data channel at the bridge master 11_B **RSVD**: Reserved
RSVDP_5	6:5	r	**Reserved for future use**
AX_MSTR_ZEROLREAD_FW	7	rw	**AXI Master Zero Length Read Forward to the application** The AXI bridge is able to terminate in order with the Posted transactions the zero length read, implementing the PCIe express flush semantics of the Posted transactions. 0_B **_0_LN_RD_TERMINATE**: The zero length Read is terminated at the DW PCIe AXI bridge master 1_B **_0_LN_RD_FWD**: The zero length Read is forward to the application
RSVDP_8	31:8	r	**Reserved for future use**

19.4.1.270 ACE Cache Coherency Control Register 1

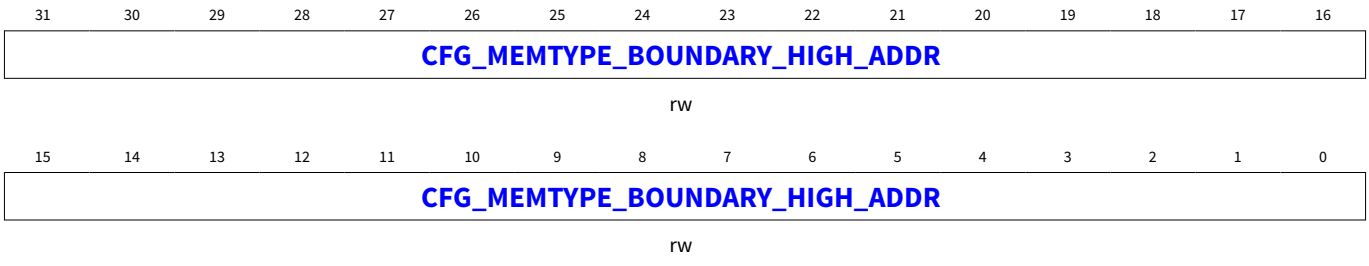
This register controls the ACE cache coherency operation.



Field	Bits	Type	Description
CFG_MEMTYPE_VALUE	0	rw	**Sets the memory type for the lower and upper parts of the address space:** Note: This register field is sticky. 0_B **LOWER_PREPH**: lower = Peripheral; upper = Memory 1_B **LOWER_MEM**: lower = Memory type; upper = Peripheral
RSVDP_1	1	r	**Reserved for future use**
CFG_MEMTYPE_BOUNDARY_LOW_ADDR	31:2	rw	**Boundary Lower Address For Memory Type** Bits [31:0] of dword-aligned address of the boundary for Memory type. The two lower address LSBs are '00'. Addresses up to but not including this value are in the lower address space region; addresses equal or greater than this value are in the upper address space region. Note: This register field is sticky.

19.4.1.271 ACE Cache Coherency Control Register 2

This register controls the ACE cache coherency operation.



Field	Bits	Type	Description
CFG_MEMTYPE_BOUNDARY_HIGH_ADDR	31:0	rw	**Boundary Upper Address For Memory Type** Bits [63:32] of the 64-bit dword-aligned address of the boundary for Memory type. Note: This register field is sticky.

19.4.1.272 ACE Cache Coherency Control Register 3

This register controls the ACE cache coherency operation.

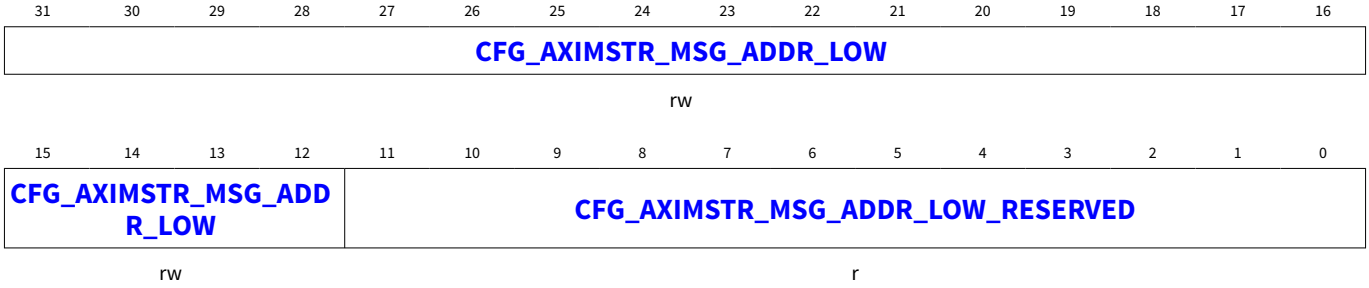
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	CFG_MSTR_AWCACHE_VALUE				0				CFG_MSTR_ARCACHE_VALUE				0		
r	rw				r				rw				r		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	CFG_MSTR_AWCACHE_MODE				0				CFG_MSTR_ARCACHE_MODE				0		
r	rw				r				rw				r		

Field	Bits	Type	Description
CFG_MSTR_ARCACHE_MODE	6:3	rw	**Master Read CACHE Signal Behavior** Defines how the individual bits in mstr_arcache are controlled. Note: This register field is sticky. 0_H **SET_BY_AXI_MASTER**: set automatically by the AXI master 1_H **SET_BY_FIELD**: set by the value of the corresponding bit of the CFG_MSTR_ARCACHE_VALUE field **others,** Reserved
CFG_MSTR_AWCACHE_MODE	14:11	rw	**Master Write CACHE Signal Behavior** Defines how the individual bits in mstr_awcache are controlled. Note: for message requests the value of mstr_awcache is always "0000" regardless of the value of this bit. Note: This register field is sticky. 0_H **SET_BY_AXI_MASTER**: set automatically by the AXI master 1_H **SET_BY_FIELD**: set by the value of the corresponding bit of the CFG_MSTR_AWCACHE_VALUE field **others,** Reserved
CFG_MSTR_ARCACHE_VALUE	22:19	rw	**Master Read CACHE Signal Value** Value of the individual bits in mstr_arcache when CFG_MSTR_ARCACHE_MODE is '1'. Note: This register field is sticky.
CFG_MSTR_AWCACHE_VALUE	30:27	rw	**Master Write CACHE Signal Value** Value of the individual bits in mstr_awcache when CFG_MSTR_AWCACHE_MODE is '1'. Note: Not applicable to message requests; for message requests the value of mstr_awcache is always '0000'. Note: This register field is sticky.
0	2:0, 10:7, 18:15, 26:23, 31	r	**Reserved** Read as all 0's; should be written with all 0's.

19.4.1.273 Lower 32-bits of the Programmable AXI Address

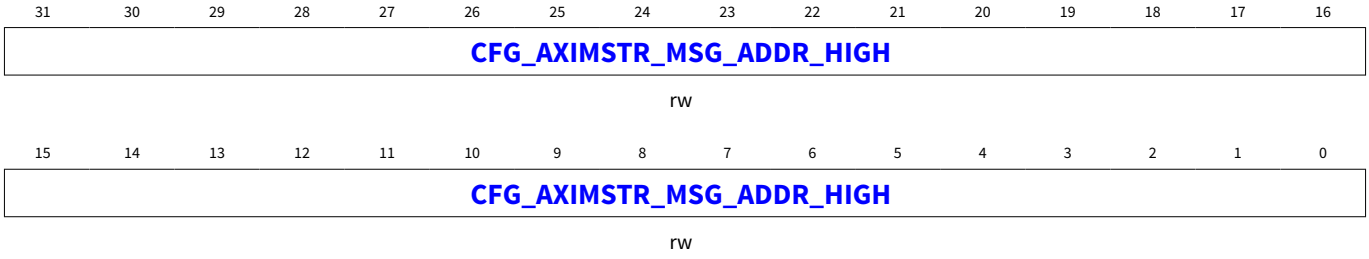
Lower 20 bits of the programmable AXI address to which Messages coming from wire are mapped. Bits [11:0] of the register are tied to zero for the address to be 4k-aligned. In previous releases, the third and fourth DWORDs of a message (Msg/MsgD) TLP header were delivered though the AXI master address bus (mstr_awaddr). These DWORDS are now supplied through the mstr_awmisc_info_hdr_34dw[63:0] output; and the value on mstr_awaddr is driven to the value you have programmed into the AXI_MSTR_MSG_ADDR_LOW_OFF and AXI_MSTR_MSG_ADDR_HIGH_OFF registers.



Field	Bits	Type	Description
CFG_AXIMSTR_MSG_ADDR_LOW_RESERVED	11:0	r	**Reserved for future use** Note: This register field is sticky.
CFG_AXIMSTR_MSG_ADDR_LOW	31:12	rw	**Lower 20-bits of the programmable AXI address for Messages** Note: This register field is sticky.

19.4.1.274 Upper 32-bits of the Programmable AXI Address

Upper 32 bits of the programmable AXI address to which Messages coming from wire are mapped.



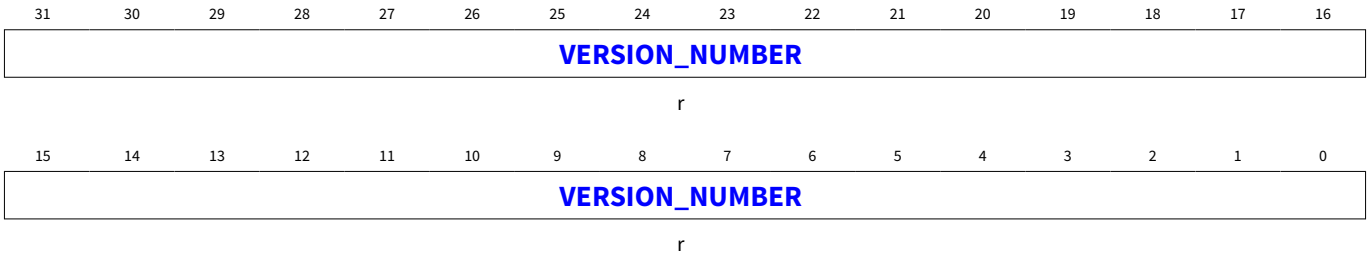
Field	Bits	Type	Description
CFG_AXIMSTR_MSG_ADDR_HIGH	31:0	rw	**Upper 32 bits of the programmable AXI address for Messages** Note: This register field is sticky.

19.4.1.275 PCIe Controller IIP Release Version Number

The version number is given in hex format. You should convert each pair of hex characters to ASCII to interpret. Using 4.70a (GA) as an example: - VERSION_NUMBER = 0x3437302a which translates to 470* - VERSION_TYPE = 0x67612a2a which translates to ga** Using 4.70a-ea01 as an example: - VERSION_NUMBER = 0x3437302a which translates to 470* - VERSION_TYPE = 0x65613031 which translates to ea01 GA is a general release available on www.designware.com EA is an early release available on a per-customer basis.

PCIE_VERSION_NUMBER_OFF Offset address: $00048F8_H$

PCIe Controller IIP Release Version Number Kernel Reset value: $3537\ 312A_H$



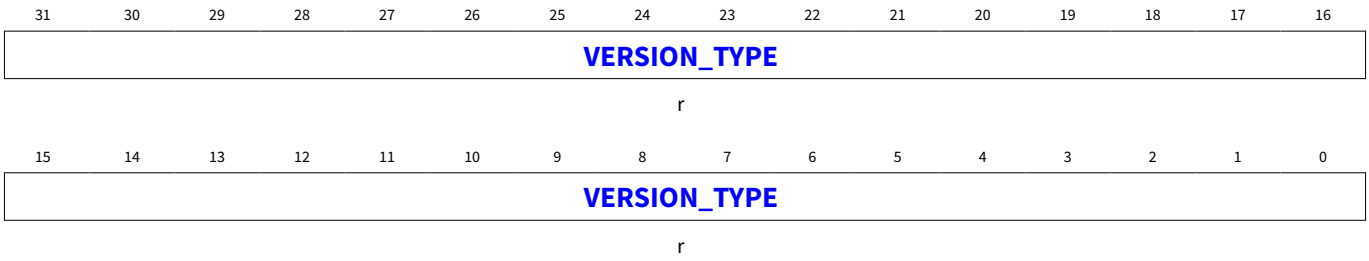
Field	Bits	Type	Description
VERSION_NUMBER	31:0	r	**Version Number** 00000000_H **MIN_VAL**: Zero value $FFFFFFFF_H$ **MAX_VAL**: Max value

19.4.1.276 PCIe Controller IIP Release Version Type

The type is given in hex format. You should convert each pair of hex characters to ASCII to interpret. Using 4.70a (GA) as an example: - VERSION_NUMBER = 0x3437302a which translates to 470* - VERSION_TYPE = 0x67612a2a which translates to ga** Using 4.70a-ea01 as an example: - VERSION_NUMBER = 0x3437302a which translates to 470* - VERSION_TYPE = 0x65613031 which translates to ea01 GA is a general release available on www.designware.com EA is an early release available on a per-customer basis.

PCIE_VERSION_TYPE_OFF Offset address: $00048FC_H$

PCIe Controller IIP Release Version Type Kernel Reset value: $6C70\ 3031_H$



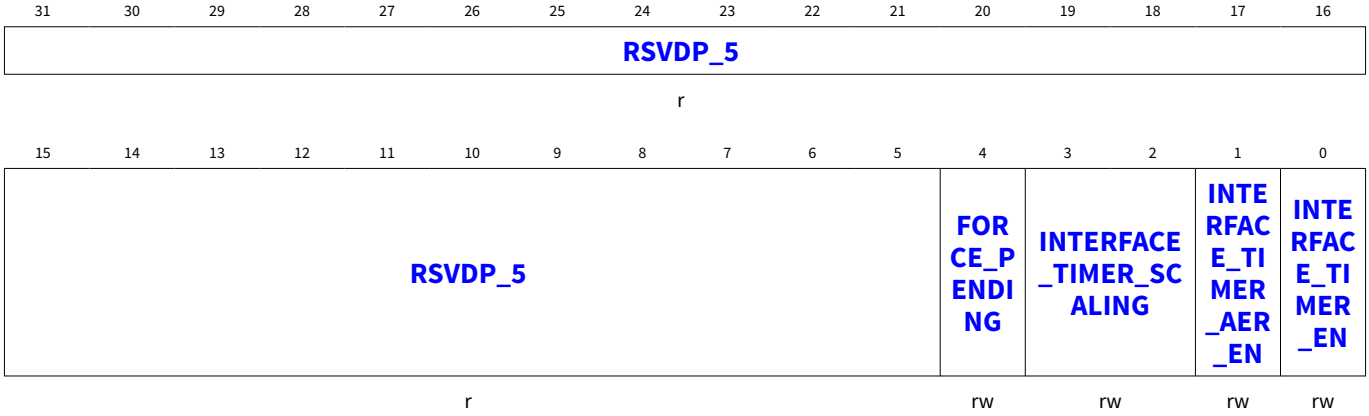
Field	Bits	Type	Description
VERSION_TYPE	31:0	r	**Version Type** 00000000_H **MIN_VAL**: Zero value $FFFFFFFF_H$ **MAX_VAL**: Max value

19.4.1.277 Interface Timer Control Register

This register controls the interface timer.

INTERFACE_TIMER_CONTROL_OFF Offset address: 0004930_H

Interface Timer Control Register Kernel Reset value: $0000\ 0000_H$



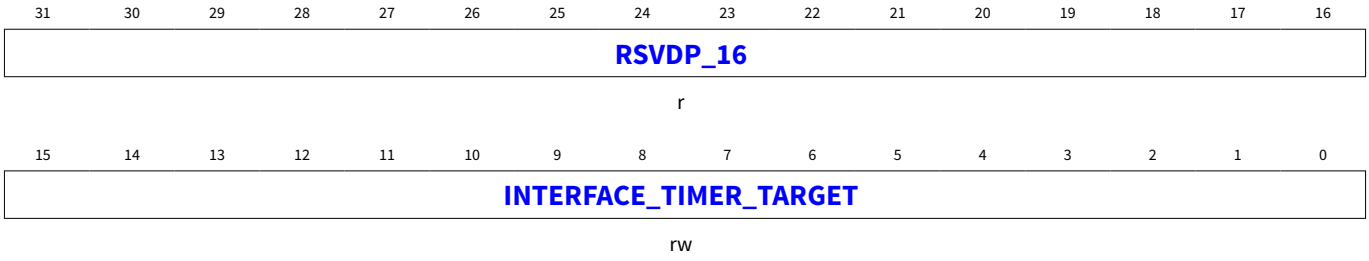
Field	Bits	Type	Description
INTERFACE_TIMER_EN	0	rw	**Interface timer enable** Note: This register field is sticky.
INTERFACE_TIMER_AER_EN	1	rw	**When set to 1 the Interface timer internal uncorrectable error generation is enabled** Note: This register field is sticky. 0_B **CLEAR**: Clear 1_B **SET**: The Interface timer internal uncorrectable error generation is enabled
INTERFACE_TIMER_SCALING	3:2	rw	**Interface timer scaling** This field can be used to reduce the timer duration. This field should only be programmed when the INTERFACE_TIMER_EN bit is set to 1'b0. Note: This register field is sticky.
FORCE_PENDING	4	rw	**Writing to this bit forces the value of the pending flags** Note: This register field is sticky.
RSVDP_5	31:5	r	**Reserved for future use**

19.4.1.278 Interface Timer Target Register

This is the interface timer target register.

INTERFACE_TIMER_TARGET_OFF Offset address: 0004934_H

Interface Timer Target Register Kernel Reset value: $0000\ 0032_H$



Field	Bits	Type	Description
INTERFACE_TIMER_TARGET	15:0	rw	**Interface timer target value** This field should only be programmed when the INTERFACE_TIMER_EN bit is set to 1'b0. Note: This register field is sticky.
RSVDP_16	31:16	r	**Reserved for future use**

19.4.1.279 Interface Timer Status Register

This is the interface timer status register.

INTERFACE_TIMER_STATUS_OFF Offset address: 0004938_H

Interface Timer Status Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				SLAVE_RD_ADD_TIMEOUT	SLAVE_WR_DATA_TIMEOUT	SLAVE_WR_ADD_TIMEOUT	0		MASTER_RD_DATA_TIMEOUT	MASTER_WR_RES_TIMEOUT	CLIENT2_INTERFACE_TIMEOUT	CLIENT1_INTERFACE_TIMEOUT	0	CPL_INTERFACE_TIMEOUT	MESSAGE_INTERFACE_TIMEOUT
r				rw1ch	rw1ch	rw1ch	r		rw1ch	rw1ch	rw1ch	rw1ch	r	rw1ch	rw1ch

Field	Bits	Type	Description
MESSAGE_INTERFACE_TIMEOUT	0	rw1ch	**Message interface timeout** Note: This register field is sticky. Software write with 1 clears this bit-field.
CPL_INTERFACE_TIMEOUT	1	rw1ch	**CPL interface timeout** Note: This register field is sticky. Software write with 1 clears this bit-field.
CLIENT1_INTERFACE_TIMEOUT	3	rw1ch	**Client1 interface (XALI1) timeout** Note: This register field is sticky. Software write with 1 clears this bit-field.
CLIENT2_INTERFACE_TIMEOUT	4	rw1ch	**Client2 interface (XALI2) timeout** Note: This register field is sticky. Software write with 1 clears this bit-field.
MASTER_WR_RES_TIMEOUT	5	rw1ch	**Master write response channel timeout** Note: This register field is sticky. Software write with 1 clears this bit-field.
MASTER_RD_DATA_TIMEOUT	6	rw1ch	**Master read data channel timeout** Note: This register field is sticky. Software write with 1 clears this bit-field.

(table continues...)

(continued)

Field	Bits	Type	Description
SLAVE_WR_ADD_TIMEOUT	9	rw1ch	**Slave write address channel timeout** Note: This register field is sticky. Software write with 1 clears this bit-field.
SLAVE_WR_DATA_TIMEOUT	10	rw1ch	**Slave write data channel timeout** Note: This register field is sticky. Software write with 1 clears this bit-field.
SLAVE_RD_ADD_TIMEOUT	11	rw1ch	**Slave read address channel timeout** Note: This register field is sticky. Software write with 1 clears this bit-field.
0	2, 8:7, 31:12	r	**Reserved** Read as 0; should be written with 0.

19.4.1.280 MSI-X Address Match Low Register

This feature is not supported in this hardware configuration. When your local AXI application writes (MWr) to the address defined in this register (and MSIX_ADDRESS_MATCH_HIGH_OFF), the controller will load the MSIX_DOORBELL_OFF register with the contents of the MWr and subsequently create and send MSI-X TLPs

MSIX_ADDRESS_MATCH_LOW_OFF Offset address: 0004940_H

MSI-X Address Match Low Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MSIX_ADDRESS_MATCH_LOW															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MSIX_ADDRESS_MATCH_LOW														MSIX_ADDRESS_MATCH_RESERVED_1	MSIX_ADDRESS_MATCH_EN
rw														r	rw

Field	Bits	Type	Description
MSIX_ADDRESS_MATCH_EN	0	rw	**MSI-X Match Enable** Enable the MSI-X Address Match feature when the AXI bridge is present. Note: This register field is sticky. 0_B **DISABLE**: Disable 1_B **ENABLE**: Enable
MSIX_ADDRESS_MATCH_RESERVED_1	1	r	**Reserved** Note: This register field is sticky.

(table continues...)

(continued)

Field	Bits	Type	Description
MSIX_ADDRESS_MATCH_LOW	31:2	rw	**MSI-X Address Match Low Address** Note: This register field is sticky.

19.4.1.281 MSIX Address Match High Register

This feature is not supported in this hardware configuration. When your local AXI application writes (MWr) to the address defined in this register (and MSIX_ADDRESS_MATCH_LOW_OFF), the controller will load the MSIX_DOORBELL_OFF register with the contents of the MWr and subsequently create and send MSI-X TLPs

MSIX_ADDRESS_MATCH_HIGH_OFF Offset address: 0004944_H

MSIX Address Match High Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MSIX_ADDRESS_MATCH_HIGH															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MSIX_ADDRESS_MATCH_HIGH															
rw															

Field	Bits	Type	Description
MSIX_ADDRESS_MATCH_HIGH	31:0	rw	**MSI-X Address Match High Address** Note: This register field is sticky. 00000000_H **MIN_VAL**: Zero value $FFFFFFFF_H$ **MAX_VAL**: Max value

19.4.1.282 MSI-X Doorbell Register

When your local application writes to this register, the controller will create and send MSI-X TLPs.

MSIX_DOORBELL_OFF Offset address: 0004948_H

MSI-X Doorbell Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MSIX_DOORBELL_RESERVED_29_31			MSIX_DOORBELL_PF					MSIX_DOORBELL_VF							
w			w					w							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MSIX_DOORBELL_VF_ACTIVE	MSIX_DOORBELL_TC			MSIX_DOORBELL_RESERVED_11	MSIX_DOORBELL_VECTOR										
w	w			w	w										

Field	Bits	Type	Description
MSIX_DOORBELL_VECTOR	10:0	w	**MSI-X Doorbell Vector** This register determines which vector to generate the MSI-X transaction for.
MSIX_DOORBELL_RESERVED_11	11	w	**Reserved**
MSIX_DOORBELL_TC	14:12	w	**MSIX Doorbell Traffic Class** This register determines which traffic class to generate the MSI-X transaction with.
MSIX_DOORBELL_VF_ACTIVE	15	w	**MSIX Doorbell Virtual Function Active** This register determines whether a Virtual Function is used to generate the MSI-X transaction.
MSIX_DOORBELL_VF	23:16	w	**MSIX Doorbell Virtual Function** This register determines the Virtual Function for the MSI-X transaction.
MSIX_DOORBELL_PF	28:24	w	**MSIX Doorbell Physical Function** This register determines the Physical Function for the MSI-X transaction.
MSIX_DOORBELL_RESERVED_29_31	31:29	w	**Reserved**

19.4.1.283 MSI-X RAM power mode and debug control register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MSIX_RAM_CTRL_RESERVED_26_31						MSIX_RAM_CTRL_DBG_PBA	MSIX_RAM_CTRL_DBG_TABLE	MSIX_RAM_CTRL_RESERVED_17_23							MSIX_RAM_CTRL_BYPASS
r						rw	rw	r							rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MSIX_RAM_CTRL_RESERVED_10_15						MSIX_RAM_CTRL_PBA_SD	MSIX_RAM_CTRL_PBA_DS	MSIX_RAM_CTRL_RESERVED_2_7						MSIX_RAM_CTRL_TABLE_SD	MSIX_RAM_CTRL_TABLE_DS
r						rw	rw	r						rw	rw

Field	Bits	Type	Description
MSIX_RAM_CTRL_TABLE_DS	0	rw	**MSIX Table RAM Deep Sleep** Note: This register field is sticky. 0_B **CLEAR**: Clear 1_B **SET**: Set this bit to drive the cfg_msix_table_ds output to signal your external logic to place the MSIX Table RAM in Deep Sleep low-power mode
MSIX_RAM_CTRL_TABLE_SD	1	rw	**MSIX Table RAM Shut Down** Note: This register field is sticky. 0_B **CLEAR**: Clear 1_B **SET**: Set this bit to drive the cfg_msix_table_sd output to signal your external logic to place the MSIX Table RAM in Shut Down low-power mode
MSIX_RAM_CTRL_RESERVED_2_7	7:2	r	**Reserved** Note: This register field is sticky.
MSIX_RAM_CTRL_PBA_DS	8	rw	**MSIX PBA RAM Deep Sleep** Note: This register field is sticky. 0_B **CLEAR**: Clear 1_B **SET**: Set this bit to drive the cfg_msix_pba_ds output to signal your external logic to place the MSIX PBA RAM in Deep Sleep low-power mode
MSIX_RAM_CTRL_PBA_SD	9	rw	**MSIX PBA RAM Shut Down** Note: This register field is sticky. 0_B **CLEAR**: Clear 1_B **SET**: Set this bit to drive the cfg_msix_pba_sd output to signal your external logic to place the MSIX PBA RAM in Shut Down low-power mode
MSIX_RAM_CTRL_RESERVED_10_15	15:10	r	**Reserved** Note: This register field is sticky.
MSIX_RAM_CTRL_BYPASS	16	rw	**MSIX RAM Control Bypass** It is up to the application to ensure the RAMs are in the proper power state before trying to access them. Moreover, the application needs to observe all timing requirements of the RAM low power signals before trying to use the MSIX functionality. Note: This register field is sticky. 0_B **CLEAR**: Clear 1_B **SET**: The bypass field, when set, disables the internal generation of low power signals for both RAMs
MSIX_RAM_CTRL_RESERVED_17_23	23:17	r	**Reserved** Note: This register field is sticky.

(table continues...)

(continued)

Field	Bits	Type	Description
MSIX_RAM_CTRL_DBG_TABLE	24	rw	**MSIX Table RAM Debug Mode** You can also use the dbg_table input to activate debug mode. Debug mode turns off the PF/VF/Offset-based addressing into the RAM and maps the entire table linearly from the base address of the BAR (indicated by the BIR) in function 0. Note: This register field is sticky. 0_B **DISABLE**: Disable 1_B **ENABLE**: Enable Use this bit to activate the debug mode and allow direct read/write access to the Table.
MSIX_RAM_CTRL_DBG_PBA	25	rw	**MSIX PBA RAM Debug Mode** You can also use the dbg_pba input to activate debug mode. Debug mode turns off the PF/VF/Offset-based addressing into the RAM and maps the entire table linearly from the base address of the BAR (indicated by the BIR) in function 0. Note: This register field is sticky. 0_B **DISABLE**: Disable 1_B **ENABLE**: Enable Use this bit to activate the debug mode and allow direct read/write access to the PBA.
MSIX_RAM_CTRL_RESERVED_26_31	31:26	r	**Reserved** Note: This register field is sticky.

19.4.1.284 Functional Safety Interrupt Events Mask Register

This registers holds the masks for functional safety interrupt events.

SAFETY_MASK_OFF Offset address: 0004960_H

Functional Safety Interrupt Events Mask Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_6															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVDP_6										SAFETY_INT_MASK_5	SAFETY_INT_MASK_4	SAFETY_INT_MASK_3	SAFETY_INT_MASK_2	SAFETY_INT_MASK_1	SAFETY_INT_MASK_0
r										rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
SAFETY_INT_MASK_0	0	rw	**Mask for functional safety interrupt event 0 (RASDP)** Note: This register field is sticky.
SAFETY_INT_MASK_1	1	rw	**Mask for functional safety interrupt event 1 (Interface timers)** Note: This register field is sticky.

(table continues...)

(continued)

Field	Bits	Type	Description
SAFETY_INT_MASK_2	2	rw	**Mask for functional safety interrupt event 2 (CDM register checker)** Note: This register field is sticky.
SAFETY_INT_MASK_3	3	rw	**Mask for functional safety interrupt event 3 (PCIe uncorrectable)** Note: This register field is sticky.
SAFETY_INT_MASK_4	4	rw	**Mask for functional safety interrupt event 4 (PCIe correctable)** Note: This register field is sticky.
SAFETY_INT_MASK_5	5	rw	**Mask for functional safety interrupt event 5 (RASDP correctable)** Note: This register field is sticky.
RSVDP_6	31:6	r	**Reserved for future use**

19.4.1.285 Status for functional safety interrupt events

This register provides the functional safety interrupt events status.

SAFETY_STATUS_OFF Offset address: 0004964_H

Status for functional safety interrupt events Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_6															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVDP_6										SAFETY_INT_STATUS_5	SAFETY_INT_STATUS_4	SAFETY_INT_STATUS_3	SAFETY_INT_STATUS_2	SAFETY_INT_STATUS_1	SAFETY_INT_STATUS_0
r										rw1ch	rw1ch	rw1ch	rw1ch	rw1ch	rw1ch

Field	Bits	Type	Description
SAFETY_INT_STATUS_0	0	rw1ch	**Status for functional safety interrupt event 0 (RASDP)** Note: This register field is sticky. Software write with 1 clears this bit-field.
SAFETY_INT_STATUS_1	1	rw1ch	**Status for functional safety interrupt event 1 (Interface timers)** Note: This register field is sticky. Software write with 1 clears this bit-field.
SAFETY_INT_STATUS_2	2	rw1ch	**Status for functional safety interrupt event 2 (CDM register checker)** Note: This register field is sticky. Software write with 1 clears this bit-field.
SAFETY_INT_STATUS_3	3	rw1ch	**Status for functional safety interrupt event 3 (PCIe uncorrectable)** Note: This register field is sticky. Software write with 1 clears this bit-field.

(table continues...)

(continued)

Field	Bits	Type	Description
SAFETY_INT_STATUS_4	4	rw1ch	**Status for functional safety interrupt event 4 (PCIe correctable)** Note: This register field is sticky. Software write with 1 clears this bit-field.
SAFETY_INT_STATUS_5	5	rw1ch	**Status for functional safety interrupt event 5 (RASDP correctable)** Note: This register field is sticky. Software write with 1 clears this bit-field.
RSVDP_6	31:6	r	**Reserved for future use**

19.4.1.286 Application driven bus and device number register

This register reflects the application driven bus and device number.

PL_APP_BUS_DEV_NUM_STATUS_OFF Offset address: $0004B10_H$

Application driven bus and device number register Kernel Reset value: $0000\ 0000_H$

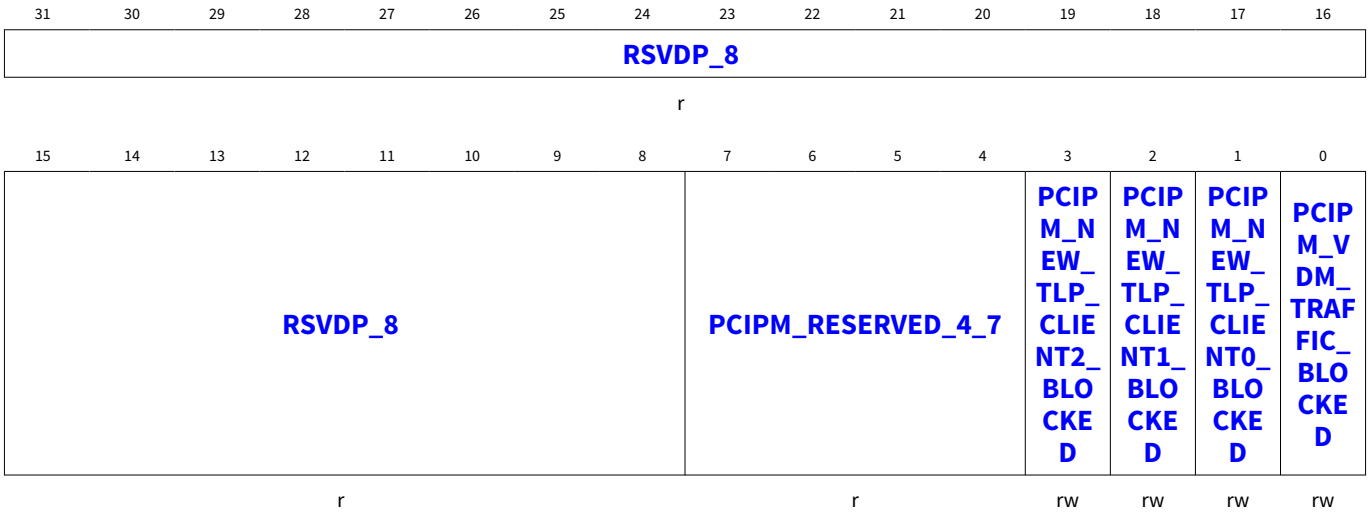
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_16															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RC_DSW_BUS_NUM								RC_DSW_DEV_NUM					RSVDP_0		
r								r					r		

Field	Bits	Type	Description
RSVDP_0	2:0	r	**Reserved for future use**
RC_DSW_DEV_NUM	7:3	r	**This field reflects the value of device number driven on CFGBD.DEVNUM input signal by your application** Note: This register field is sticky.
RC_DSW_BUS_NUM	15:8	r	**This field reflects the value of bus number driven on CFGBD.BUSNUM input signal by your application** Note: This register field is sticky.
RSVDP_16	31:16	r	**Reserved for future use**

19.4.1.287 TLP Traffic during Non-D0 State Control Register

This register provides control over TLP Traffic during Non-D0 States.



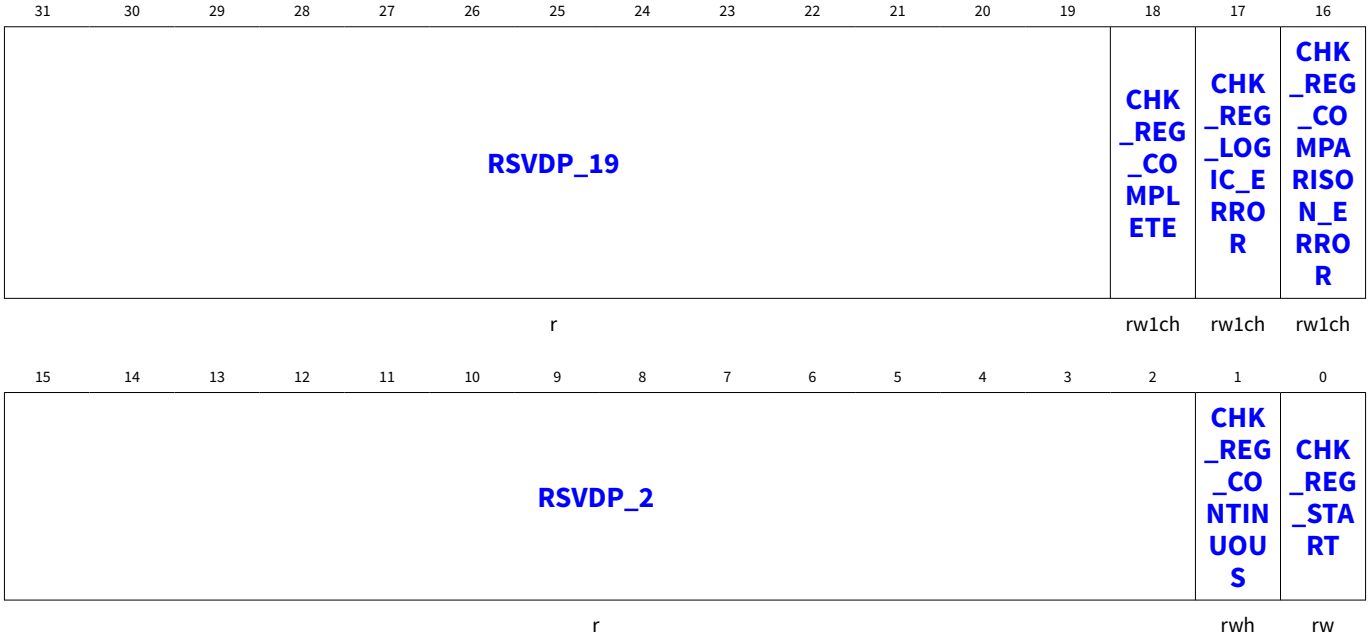
Field	Bits	Type	Description
PCIPM_VDM_TRAFFIC_BLOCKED	0	rw	**This field indicates that VDM Message TLPs are blocked during non-D0 states** Note: This register field is sticky.
PCIPM_NEW_TLP_CLIENT0_BLOCKED	1	rw	**This field indicates that all TLPs transmitted by Client 0 interface are blocked during non-D0 states** Note: This register field is sticky.
PCIPM_NEW_TLP_CLIENT1_BLOCKED	2	rw	**This field indicates that all TLPs transmitted by Client 1 interface are blocked during non-D0 states** Note: This register field is sticky.
PCIPM_NEW_TLP_CLIENT2_BLOCKED	3	rw	**This field indicates that all TLPs transmitted by Client 2 interface are blocked during non-D0 states** Note: This register field is sticky.
PCIPM_RESERVED_4_7	7:4	r	**Reserved** Note: This register field is sticky.
RSVDP_8	31:8	r	**Reserved for future use**

19.4.1.288 CDM Register Checking Control and Status Register

Controls register checking and displays status of register checking.

PL_CHK_REG_CONTROL_STATUS_OFF Offset address: $0004B20_H$

CDM Register Checking Control and Status Register Kernel Reset value: $0000\ 0000_H$



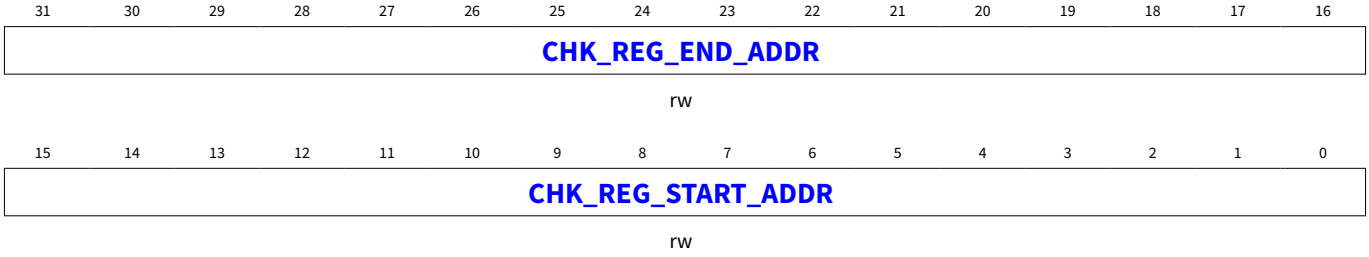
Field	Bits	Type	Description
CHK_REG_START	0	rw	**Begins a checking sequence** Note: This register field is sticky. 0_B **CLEAR**: Clear 1_B **SET**: Set
CHK_REG_CONTINUOUS	1	rwh	**Set Continuous Checking Sequence** Note: This register field is sticky. 0_B **CLEAR**: Clear 1_B **SET**: Set
RSVDP_2	15:2	r	**Reserved for future use**
CHK_REG_COMPARISON_ERROR	16	rw1ch	**The system has detected that there is a bit error in the CDM Register Data** Note: This register field is sticky. Software write with 1 clears this bit-field. 0_B **CLEAR**: Clear 1_B **SET**: Indicates comparison error
CHK_REG_LOGIC_ERROR	17	rw1ch	**The system has detected an error in its own checking logic** Note: This register field is sticky. Software write with 1 clears this bit-field. 0_B **CLEAR**: Clear 1_B **SET**: Indicates an error in comparison logic
CHK_REG_COMPLETE	18	rw1ch	**The system has completed a checking cycle** Note: This register field is sticky. Software write with 1 clears this bit-field. 0_B **CLEAR**: Clear 1_B **SET**: Indicates the system has completed a checking cycle
RSVDP_19	31:19	r	**Reserved for future use**

19.4.1.289 CDM Register Checking First and Last address to check

This register holds the first and last address to check.

PL_CHK_REG_START_END_OFF Offset address: $0004B24_H$

CDM Register Checking First and Last address to check Kernel Reset value: $0BFF\ 0000_H$



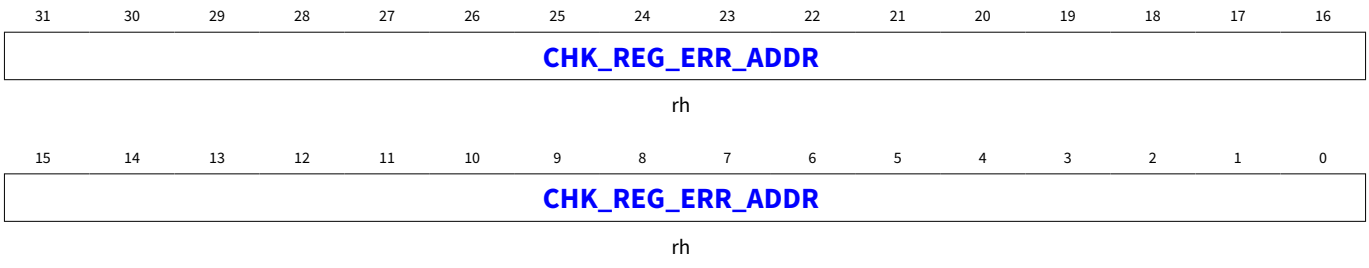
Field	Bits	Type	Description
CHK_REG_START_ADDR	15:0	rw	**The first address that is checked by the system** Note: This register field is sticky. 0000_H **MIN_VAL**: Zero value $FFFF_H$ **MAX_VAL**: Max value
CHK_REG_END_ADDR	31:16	rw	**The last address that is checked by the system** Note: This register field is sticky. 0000_H **MIN_VAL**: Zero value $FFFF_H$ **MAX_VAL**: Max value

19.4.1.290 CDM Register Checking Error Address

This register holds the CDM register checking error address.

PL_CHK_REG_ERR_ADDR_OFF Offset address: $0004B28_H$

CDM Register Checking Error Address Kernel Reset value: $0000\ 0000_H$



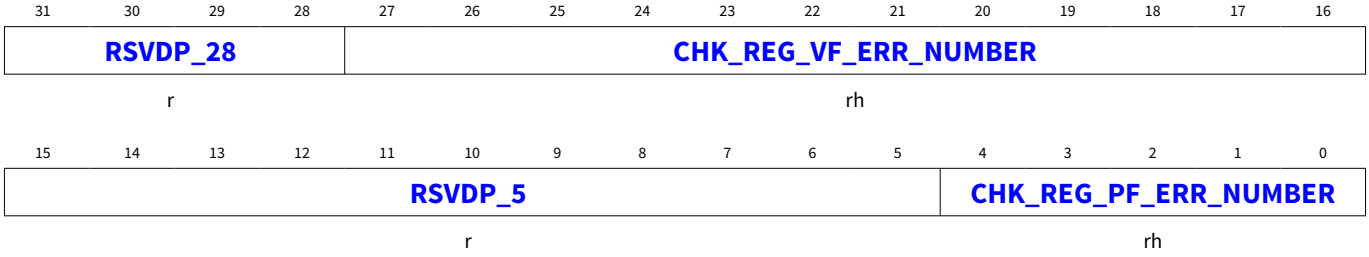
Field	Bits	Type	Description
CHK_REG_ERR_ADDR	31:0	rh	**The address at which an error has been detected** Valid only when the CDM Register Checker Comparison Error bit is set in the status register. Note: This register field is sticky. 00000000_H **MIN_VAL**: Zero value $FFFFFFFF_H$ **MAX_VAL**: Max value

19.4.1.291 CDM Register Checking error PF Numbers

This register holds the CDM Register checking error PF numbers.

PL_CHK_REG_ERR_PF_VF_OFF Offset address: $0004B2C_H$

CDM Register Checking error PF Numbers Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
CHK_REG_PF_ERR_NUMBER	4:0	rh	**The PF number at which the error was detected** Valid only when the CDM Register Checker Comparison Error bit is set in the status register. Note: This register field is sticky. 00_H **MIN_VAL**: Zero value $1F_H$ **MAX_VAL**: Max value
RSVDP_5	15:5	r	**Reserved for future use**
CHK_REG_VF_ERR_NUMBER	27:16	rh	**The VF number at which the error was detected** Valid only when the CDM Register Checker Comparison Error bit is set in the status register. Note: This register field is sticky. 000_H **MIN_VAL**: Zero value $7FF_H$ **MAX_VAL**: Max value **others,** Reserved
RSVDP_28	31:28	r	**Reserved for future use**

19.4.1.292 LTR Latency Register

The function of this register field (and all other fields in this register) differs between an upstream port and a downstream port. For an upstream port, the register fields capture the corresponding fields in the LTR messages that are transmitted by the port. For a downstream port, the register fields capture the corresponding fields in the LTR messages that are received by the port.

PL_LTR_LATENCY_OFF Offset address: $0004B30_H$

LTR Latency Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
NO_SNOOP_LATENCY_REQUIRE	RSVDP_29		NO_SNOOP_LATENCY_SCALE			NO_SNOOP_LATENCY_VALUE									
rw	r		rw			rw									

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SNOOP_LATENCY_REQUIRE	RSVDP_13		SNOOP_LATENCY_SCALE			SNOOP_LATENCY_VALUE									
rw	r		rw			rw									

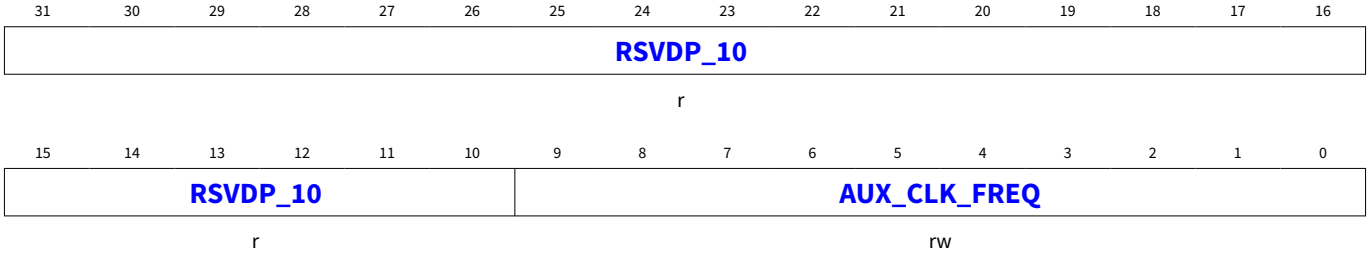
Field	Bits	Type	Description
SNOOP_LATENCY_VALUE	9:0	rw	**Snoop Latency Value** Note: The access attributes of this field are as follows: - Wire: R - Dbi: R/W
SNOOP_LATENCY_SCALE	12:10	rw	**Snoop Latency Scale** Note: The access attributes of this field are as follows: - Wire: R - Dbi: R/W
RSVDP_13	14:13	r	**Reserved for future use**
SNOOP_LATENCY_REQUIRE	15	rw	**Snoop Latency Requirement** Note: The access attributes of this field are as follows: - Wire: R - Dbi: R/W
NO_SNOOP_LATENCY_VALUE	25:16	rw	**No Snoop Latency Value** Note: The access attributes of this field are as follows: - Wire: R - Dbi: R/W
NO_SNOOP_LATENCY_SCALE	28:26	rw	**No Snoop Latency Scale** Note: The access attributes of this field are as follows: - Wire: R - Dbi: R/W
RSVDP_29	30:29	r	**Reserved for future use**
NO_SNOOP_LATENCY_REQUIRE	31	rw	**No Snoop Latency Requirement** Note: The access attributes of this field are as follows: - Wire: R - Dbi: R/W

19.4.1.293 Auxiliary Clock Frequency Control Register

This register controls the auxiliary clock frequency.

AUX_CLK_FREQ_OFF Offset address: $0004B40_H$

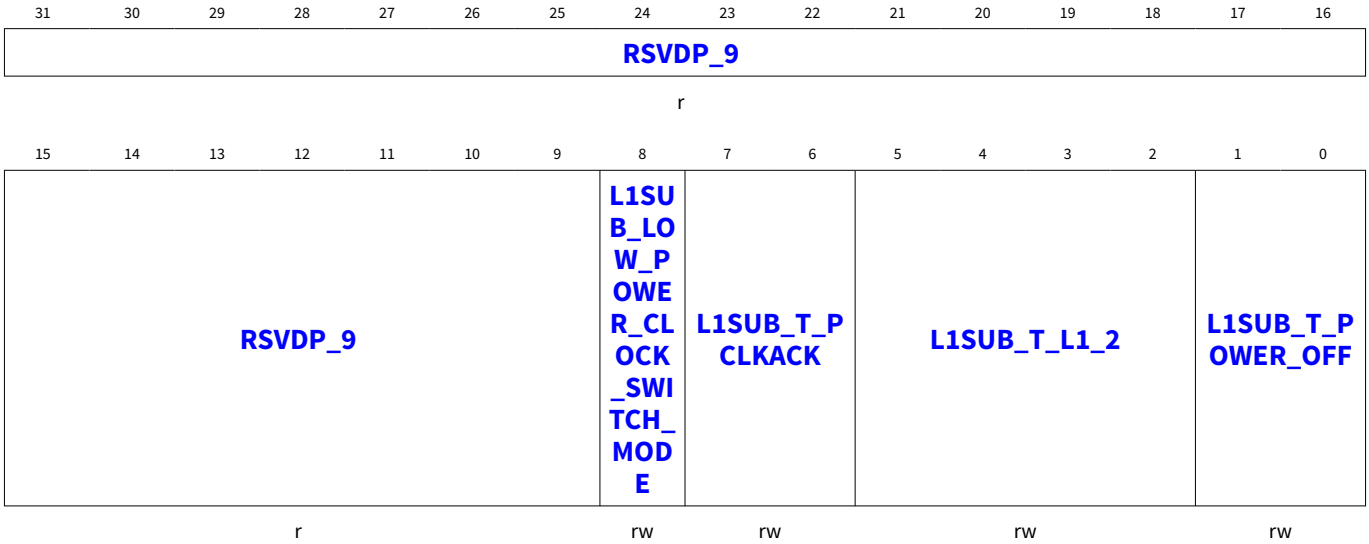
Auxiliary Clock Frequency Control Register Kernel Reset value: $0000\ 000A_H$



Field	Bits	Type	Description
AUX_CLK_FREQ	9:0	rw	**The aux_clk frequency in MHz** This value is used to provide a 1 µs reference for counting time during low-power states with aux_clk when the PHY has removed the pipe_clk. If the actual frequency (f) of aux_clk does not exactly match the programmed frequency (f_prog), then there is an error in the time counted by the controller that can be expressed in percentage as: err% = (f_prog/f-1)*100. For example if f=2.5 MHz and f_prog=3 MHz, then err% =(3/2.5-1)*100 =20%, meaning that the time counted by the controller on aux_clk will be 20% greater than the time in µs programmed in the corresponding time register (for example T_POWER_ON). Note: This register field is sticky.
RSVDP_10	31:10	r	**Reserved for future use**

19.4.1.294 L1 Substates Timing Register

This register holds L1 substates timing information.



Field	Bits	Type	Description
L1SUB_T_POWER_OFF	1:0	rw	**Duration (in 1 µs units) of L1.2.Entry** Range is 1..4. Note: This register field is sticky. 00_B **MIN_VAL**: Min value 11_B **MAX_VAL**: Max value

(table continues...)

(continued)

Field	Bits	Type	Description
L1SUB_T_L1_2	5:2	rw	**Duration (in 1 µs units) of L1.2** Range is 1..16. Note: This register field is sticky. 0_H **MIN_VAL**: Min value F_H **MAX_VAL**: Max value
L1SUB_T_PCLKACK	7:6	rw	**Max delay (in 1 µs units) between a MAC request to remove the clock and a PHY response** If the PHY does not respond within this time the request is aborted. Range is 1..4 Note: This register field is sticky. 00_B **MIN_VAL**: Min value 11_B **MAX_VAL**: Max value
L1SUB_LOW_POWER_CLOCK_SWITCH_MODE	8	rw	**If the bit is set to 1'b1 the controller will delay the switching of aux_clk to the slow platform clock until it detects that the link partner has de-asserted CLKREQ#** Note: This register field is sticky. 0_B **CLEAR**: The reference clock may be gated off when CLKREQ# is de-asserted 1_B **SET**: The reference clock shall be kept running regardless of the CLKREQ# setting
RSVDP_9	31:9	r	**Reserved for future use**

19.4.1.295 Powerdown Control and Status Register

This is the Powerdown Control and Status register.

POWERDOWN_CTRL_STATUS_OFF Offset address: $0004B48_H$

Powerdown Control and Status Register Kernel Reset value: $0000\ 0220_H$

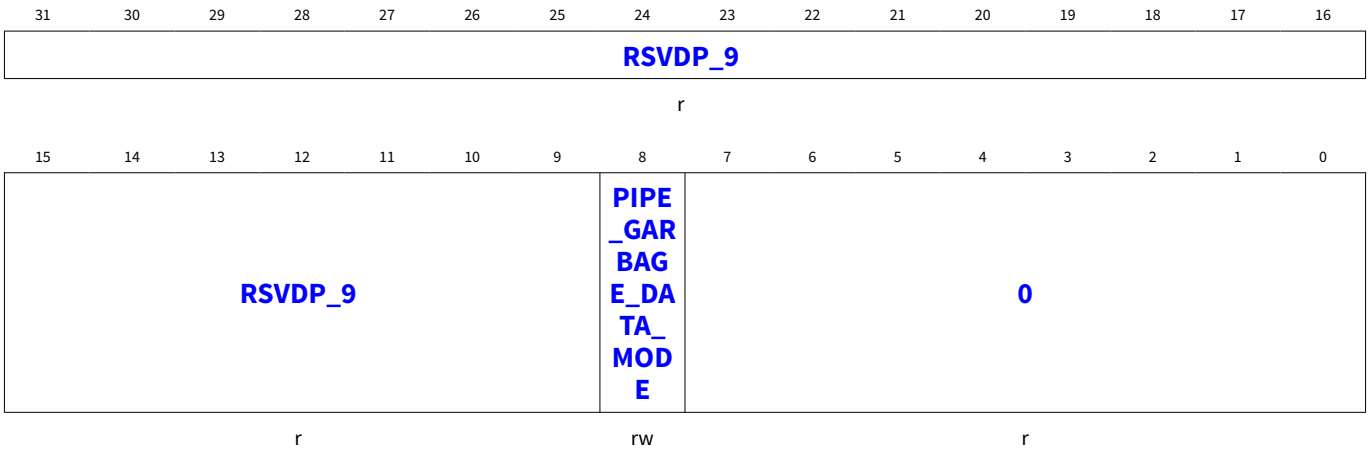
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_12															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVDP_12				POWERDOWN_PHY_POWERDOWN				POWERDOWN_MAC_POWERDOWN				RSVDP_2		POWERDOWN_VMAIN_ACK	POWERDOWN_FORCE
r				rh				rh				r		rw	rwh

Field	Bits	Type	Description
POWERDOWN_FORCE	0	rwh	**This field is a one shot field** This field could be used for debug purposes in event that the P2 Powerdown transition does not complete. It will allow the controller to proceed with the transition to the P1 Powerdown state. This field always reads back as 1'b0. 0_B **CLEAR**: Clear 1_B **SET**: Writing a value of 1 to this field causes the controller to complete the P2 Powerdown handshake regardless of whether the PHY has returned Phystatus
POWERDOWN_VMAIN_ACK	1	rw	**Set this bit to 1 if you do not want to perform the handshake with the power-switch after PERST# assertion** By default the controller will perform the handshake with the power-switch if L1 power gating is enabled Note: This register field is sticky. 0_B **CLEAR**: Clear 1_B **SET**: If you do not want to perform the handshake with the power-switch after PERST# assertion
RSVDP_2	3:2	r	**Reserved for future use**
POWERDOWN_MAC_POWERDOWN	7:4	rh	**This field represents the Powerdown value driven by the controller to the PHY**
POWERDOWN_PHY_POWERDOWN	11:8	rh	**This field represents the Powerdown value that has been acknowledged by the PHY** It is updated with the value of Powerdown driven by the controller, when the PHY has returned the Phystatus acknowledgment for the Powerdown transition.
RSVDP_12	31:12	r	**Reserved for future use**

19.4.1.296 PIPE Related Register

This register controls the PIPE's capabitity, control, and status parameters.



Field	Bits	Type	Description
PIPE_GARBAGE_DATA_MODE	8	rw	**PIPE Garbage Data Mode** • RxValid is deasserted - a valid RxStartBlock is received at 128b/130b encoding • a valid COM symbol is received at 8b/10b encoding Note: This register field is sticky. 0_B **COMPLIANT_MODE**: PIPE Spec compliant mode: The MAC discards any symbols received after the electrical idle ordered-set until RxValid is deasserted 1_B **PHY_SUP_MODE**: Special PHY Support mode: The MAC discards any symbols received after the electrical idle ordered-set until when any of the following three conditions are true:
RSVDP_9	31:9	r	**Reserved for future use**
0	7:0	r	**Reserved** Read as all 0's; should be written with all 0's.

19.4.1.297 MSIX table x DWORD0

MSIXTABLEx_LADDR (x=0-63) Offset address: $0005A00_H+x*10_H$

MSIX table x DWORD0 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LADDR															
rwh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LADDR															
rwh															

Field	Bits	Type	Description
LADDR	31:0	rwh	**Msg lower address**

19.4.1.298 MSIX table x DWORD1

MSIXTABLEx_UADDR (x=0-63) Offset address: $0005A04_H+x*10_H$

MSIX table x DWORD1 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UADDR															
rwh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UADDR															
rwh															

Field	Bits	Type	Description
UADDR	31:0	rwh	**Msg upper address**

19.4.1.299 MSIX table x DWORD2

MSIXTABLEx_DATA (x=0-63) Offset address: $0005A08_H+x*10_H$

MSIX table x DWORD2 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DATA															
rwh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA															
rwh															

Field	Bits	Type	Description
DATA	31:0	rwh	**Msg data**

19.4.1.300 MSIX table x DWORD3

MSIXTABLEx_VECTRL (x=0-63) Offset address: $0005A0C_H+x*10_H$

MSIX table x DWORD3 Kernel Reset value: $0000\ 0001_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
STUP								STLOW							
rwh								rwh							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															MASK
r															rwh

Field	Bits	Type	Description
MASK	0	rwh	**Mask bit** When this bit is Set, the Function is prohibited from sending a message using this MSI-X Table entry. However, any other MSI-X Table entries programmed with the same vector will still be capable of sending an equivalent message unless they are also masked. Default value of this bit is 1b (entry is masked).
STLOW	23:16	rwh	**ST lower** As defined in the PCIe spec.
STUP	31:24	rwh	**ST Upper** As defined in the PCIe spec.

(table continues...)

(continued)

Field	Bits	Type	Description
0	15:1	r	**Reserved** Read as 0; should be written with 0.

19.4.1.301 MSIX xtable DWORD

MSIXPBAx_DWORD (x=0-1) Offset address: $0005E10_H+x*4$

MSIX xtable DWORD Kernel Reset value: $0000\ 0080_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PB31	PB30	PB29	PB28	PB27	PB26	PB25	PB24	PB23	PB22	PB21	PB20	PB19	PB18	PB17	PB16
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PB15	PB14	PB13	PB12	PB11	PB10	PB9	PB8	PB7	PB6	PB5	PB4	PB3	PB2	PB1	PB0
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
PBy (y=0-31)	y	rh	**Pending bit, as defined in the PCIe spec.**

19.4.1.302 BAR i Mask Register

This register is the mask for BAR_REG. Normally, the BAR masks are used for indicating the amount of address space that each BAR requests from host software. The BAR masks determine which bits in each BAR are non-writable by host software, which determines the size of the address space claimed by each BAR. The BAR mask values indicate the range of low-order bits, in each implemented BAR, not to use for address matching. The BAR mask value also indicates the range of low-order bits in the BAR that cannot be written from the host. The application can write to all BAR bits to allow setting of memory, I/O, and other standard BAR options. Your local CPU can change the mask at runtime using the DBI. The mask register is invisible to the PCIe wire but visible to your local CPU through the DBI. The reset value of BARMASK_REG0 is 0x3FFF instead of 0xFFFF. You cannot read the mask register but you can write to it.

BARi_MASK_REG (i=0-5) Offset address: 0006010_H+i*4

BAR i Mask Register Kernel Reset value: $0000\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PCI_TYPE0_BAR0_MASK															
w															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PCI_TYPE0_BAR0_MASK															PCI_TYPE0_BAR0_ENABLED
w															w

Field	Bits	Type	Description
PCI_TYPE0_BAR0_ENABLED	0	w	**BAR0 Mask Enabled** Note: This register field is sticky.
PCI_TYPE0_BAR0_MASK	31:1	w	**BAR0 Mask** Note: The access attributes of this field are as follows: - Wire: No access - Dbi: No access - Dbi2: W (sticky) Note: This register field is sticky.

19.4.1.303 MSI-X Capability ID, Next Pointer, Control Register

This is the shadow register of the MSI-X Capability ID, Next Pointer, and Control Register.

SHADOW_PCI_MSIX_CAP_ID_NEXT_CTRL_REG Offset address: $00060B0_H$

MSI-X Capability ID, Next Pointer, Control Register Kernel Reset value: $0040\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PCI_MSIX_RESERVED1					PCI_MSIX_TABLE_SIZE										
r					rh										

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PCI_MSIX_RESERVED0															
r															

Field	Bits	Type	Description
PCI_MSIX_RESERVED0	15:0	r	**reserved field 0 in the shadow register** Note: This register field is sticky.
PCI_MSIX_TABLE_SIZE	26:16	rh	**MSI-X Table Size in the shadow register** Note: The access attributes of this field are as follows: - Wire: No access - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R Note: This register field is sticky.
PCI_MSIX_RESERVED1	31:27	r	**reserved field 1 in the shadow register** Note: This register field is sticky.

19.4.1.304 MSI-X Table Offset and BIR Register

This register provides Table BIR and MSI-x Table offset select.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PCI_MSIX_TABLE_OFFSET															
rwh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PCI_MSIX_TABLE_OFFSET													PCI_MSIX_BIR		
rwh													rwh		

Field	Bits	Type	Description
PCI_MSIX_BIR	2:0	rwh	**MSI-X Table BAR Indicator Register Field** This field indicates which one of a Function's Base Address Registers , located beginning at 10h in Configuration Space, or entry in the Enhanced Allocation capability with a matching BEI , is used to map the Function's MSI-X Table into Memory Space. All encodings other than the defined encodings are reserved. For a 64-bit Base Address Register , the Table BIR indicates the lower DWORD. For Functions with Type 1 Configuration Space headers, BIR values 2 through 5 are also Reserved. Note: The access attributes of this field are as follows: - Wire: No access - Dbi: if (DBI_RO_WR_EN == 1 && MSIX_CAP_ENABLE=1) then R/W else R Note: This register field is sticky. 000_B **BAR_10**: Base Address Register 10h 001_B **BAR_14**: Base Address Register 14h 010_B **BAR_18**: Base Address Register 18h 011_B **BAR_1C**: Base Address Register 1Ch 100_B **BAR_20**: Base Address Register 20h 101_B **BAR_24**: Base Address Register 24h **others,** Reserved
PCI_MSIX_TABLE_OFFSET	31:3	rwh	**MSI-X Table Offset** Used as an offset from the address contained by one of the Function's Base Address Registers to point to the base of the MSI-X Table. The lower 3 Table BIR bits are masked off (set to zero) by software to form a 32-bit QWORD-aligned offset. Note: The access attributes of this field are as follows: - Wire: No access - Dbi: if (DBI_RO_WR_EN == 1 && MSIX_CAP_ENABLE=1) then R/W else R Note: This register field is sticky.

19.4.1.305 MSI-X PBA Offset and BIR Register

This register provides PBA Offset and PBA BIR value.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PCI_MSIX_PBA_OFFSET															
rwh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PCI_MSIX_PBA_OFFSET													PCI_MSIX_PBA_BIR		
rwh													rwh		

Field	Bits	Type	Description
PCI_MSIX_PBA_BIR	2:0	rwh	**MSI-X PBA BIR** This field indicates which one of a Function's Base Address Registers , located beginning at 10h in Configuration Space, or entry in the Enhanced Allocation capability with a matching BEI, is used to map the Function's MSI-X PBA into Memory Space. The PBA BIR value definitions are identical to those for the Table BIR . Note: The access attributes of this field are as follows: • Wire: No access • Dbi: if (DBI_RO_WR_EN == 1 && MSIX_CAP_ENABLE=1) then R/W else R Note: This register field is sticky.
PCI_MSIX_PBA_OFFSET	31:3	rwh	**MSI-X PBA Offset** Used as an offset from the address contained by one of the Function's Base Address Registers to point to the base of the MSI-X PBA. The lower 3 PBA BIR bits are masked off (set to zero) by software to form a 32-bit QWORD-aligned offset. Note: The access attributes of this field are as follows: • Wire: No access • Dbi: if (DBI_RO_WR_EN == 1 && MSIX_CAP_ENABLE=1) then R/W else R Note: This register field is sticky.

19.4.1.306 DMA Arbitration Scheme for TRGT1 Interface

This register is used to control traffic priorities among various sources that are delivered to your application through TRGT1 where 0x0 represents the highest priority. - Non-DMA Rx Requests - DMA Write Channel MRd Requests (DMA data requests and LL element/descriptor access) - DMA Read Channel MRd Requests (LL element/descriptor access) - DMA Read Channel MWr Requests Concurrent traffic from channels with same priority are sorted according to Round-Robin arbitration rules. The arbitration priority defaults to Non-DMA requests (highest), Write Channel MRd, Read Channel MRd, Read Channel MWr.

DMA_CTRL_DATA_ARB_PRIOR_OFF Offset address: 0007000_H

DMA Arbitration Scheme for TRGT1 Interface Kernel Reset value: $0000\ 0688_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_17															VERSION
r															r

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VERSION				RDBUFF_TRGT_WEIGHT			RD_CTRL_TRGT_WEIGHT			WR_CTRL_TRGT_WEIGHT			RTRGT1_WEIGHT		
r				rw			rw			rw			rw		

Field	Bits	Type	Description
RTRGT1_WEIGHT	2:0	rw	**Non-DMA Rx Requests** Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
WR_CTRL_TRGT_WEIGHT	5:3	rw	**DMA Write Channel MRd Requests (for DMA data requests and LL element/descriptor access)** Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
RD_CTRL_TRGT_WEIGHT	8:6	rw	**DMA Read Channel MRd Requests (for LL element/descriptor access)** Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
RDBUFF_TRGT_WEIGHT	11:9	rw	**DMA Read Channel MWr Requests** Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
VERSION	16:12	r	**Reports the version of Register Map of eDMA**
RSVDP_17	31:17	r	**Reserved for future use**

19.4.1.307 DMA Number of Channels Register

This register provides information regarding the number of configured DMA read and write channels. Note: When this register does not exist, value is fixed to 32'hFFFF_FFFF.

DMA_CTRL_OFF Offset address: 0007008_H

DMA Number of Channels Register Kernel Reset value: $0004\ 0004_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_26						Res		RSVDP_20				NUM_DMA_RD_CHAN			
r						r		r				r			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVDP_4												NUM_DMA_WR_CHAN			
r												r			

Field	Bits	Type	Description
NUM_DMA_WR_CHAN	3:0	r	**Number of Write Channels** You can read this register to determine the number of write channels configured.
RSVDP_4	15:4	r	**Reserved for future use**
NUM_DMA_RD_CHAN	19:16	r	**Number of Read Channels** You can read this register to determine the number of read channels configured.
RSVDP_20	23:20	r	**Reserved for future use**
RSVDP_26	31:26	r	**Reserved for future use**

19.4.1.308 DMA Write Engine Enable Register

This register indicates the status of DMA write engine and the status of DMA write engine handshake feature (per-channel).

DMA_WRITE_ENGINE_EN_OFF Offset address: $000700C_H$

DMA Write Engine Enable Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_24								DMA_WRITE_ENGINE_EN_HSHAKE_CH7	DMA_WRITE_ENGINE_EN_HSHAKE_CH6	DMA_WRITE_ENGINE_EN_HSHAKE_CH5	DMA_WRITE_ENGINE_EN_HSHAKE_CH4	DMA_WRITE_ENGINE_EN_HSHAKE_CH3	DMA_WRITE_ENGINE_EN_HSHAKE_CH2	DMA_WRITE_ENGINE_EN_HSHAKE_CH1	DMA_WRITE_ENGINE_EN_HSHAKE_CH0
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVDP_1															DMA_WRITE_ENGINE
r															rwh

Field	Bits	Type	Description
DMA_WRITE_ENGINE	0	rwh	**DMA Write Engine Enable** For normal operation, you must initially set this bit to '1', before any other software setup action. You do not need to toggle or rewrite to this bit during normal operation. You should set this bit to '0' when you want to "Soft Reset" the DMA controller write logic. There are three possible reasons for resetting the DMA controller write logic • The "Abort Interrupt Status" bit is set in the DMA Write Interrupt Status Register (DMA_WRITE_INT_STATUS_OFF), and any of the bits in the DMA Write Error Status Register (DMA_WRITE_ERR_STATUS_OFF) are set. Resetting the DMA controller write logic re-initializes the control logic, ensuring that the next DMA write transfer is executed successfully. • You have executed the procedure outlined in "Stop Bit", after which, the "Abort Interrupt Status" bit is set and the Channel Status(CS) field of the DMA Write Channel Control 1 Register (DMA_CH_CONTROL1_OFF_WRCH_i) is set to "Stopped". Resetting the DMA controller write logic re-initializes the control logic ensuring that the next DMA write transfer is executed successfully • During software development, when you incorrectly program the DMA write engine To "Soft Reset" the DMA controller write logic, you must: • De-assert the DMA write engine enable bit. • Wait for the DMA to complete any in-progress TLP transfer, by waiting until a read on the DMA write engine enable bit returns a '0'. • Assert the DMA write engine enable bit. This "Soft Reset" does not clear the DMA configuration registers. The DMA write transfer does not start until you write to the DMA Write Doorbell Register (DMA_WRITE_DOORBELL_OFF) Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W 0_B **DISABLE**: Disable (Soft Reset) 1_B **ENABLE**: Enable
RSVDP_1	15:1	r	**Reserved for future use**
DMA_WRITE_ENGINE_EN_HSHAKE_CH0	16	rw	**Enable Handshake for DMA Write Engine Channel 0** Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
DMA_WRITE_ENGINE_EN_HSHAKE_CH1	17	rw	**Enable Handshake for DMA Write Engine Channel 1** Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
DMA_WRITE_ENGINE_EN_HSHAKE_CH2	18	rw	**Enable Handshake for DMA Write Engine Channel 2** Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
DMA_WRITE_ENGINE_EN_HSHAKE_CH3	19	rw	**Enable Handshake for DMA Write Engine Channel 3** Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W

(table continues...)

(continued)

Field	Bits	Type	Description
DMA_WRITE_ENGINE_EN_HSHAKE_CH4	20	rw	**Enable Handshake for DMA Write Engine Channel 4** Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
DMA_WRITE_ENGINE_EN_HSHAKE_CH5	21	rw	**Enable Handshake for DMA Write Engine Channel 5** Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
DMA_WRITE_ENGINE_EN_HSHAKE_CH6	22	rw	**Enable Handshake for DMA Write Engine Channel 6** Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
DMA_WRITE_ENGINE_EN_HSHAKE_CH7	23	rw	**Enable Handshake for DMA Write Engine Channel 7** Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
RSVDP_24	31:24	r	**Reserved for future use**

19.4.1.309 DMA Write Doorbell Register

This register controls the DMA write channel doorbell.

DMA_WRITE_DOORBELL_OFF Offset address: 0007010_H

DMA Write Doorbell Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
WR_STOP	RSVDP_3														
rwh	r														

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVDP_3													WR_DOORBELL_NUM		
r													rwh		

Field	Bits	Type	Description
WR_DOORBELL_NUM	2:0	rwh	**Doorbell Number** You must write the channel number to this register to start the DMA write transfer for that channel. The DMA detects a write to this register field even if the value of this field does not change. You do not need to toggle or write any other value to this register to start a new transfer. The range of this field is 0x0 to 0x7, where 0x0 corresponds to channel 0. Note: A write to this field triggers the controller to exit L1 substates. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
RSVDP_3	30:3	r	**Reserved for future use**

(table continues...)

(continued)

Field	Bits	Type	Description
WR_STOP	31	rwh	**Stop** Set in conjunction with the Doorbell Number field. The DMA write channel stops issuing requests, sets the channel status to "Stopped", and asserts the "Abort" interrupt if it is enabled. Before setting the Stop bit, you must read the channel Status field (CS) of the DMA Write Channel Control 1 Register (DMA_CH_CONTROL1_OFF_WRCH_0) to ensure that the write channel is "Running" (transferring data). For more information, see "Stopping the DMA Transfer (Software Stop)". Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W

19.4.1.310 DMA Write Engine Channel Arbitration Weight Low Register

The 5-bit channel weight (for write channels 0-3) specifies the maximum number of TLP requests that the DMA can issue for that channel before it must return to the arbitration routine. When the channel weight count is reached or DMA channel request transfer size reaches zero, the WWR arbiter selects the next channel to be processed. Your software must initialize this register before ringing the doorbell. Value range is (0-0x1F) corresponding to (1-32) transaction requests.

DMA_WRITE_CHANNEL_ARB_WEIGHT_LOW_OFF Offset address: 0007018_H

DMA Write Engine Channel Arbitration Weight Low Register — Kernel Reset value: $0000\ 8421_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_20												WRITE_CHANNEL3_WEIGHT			
r												rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WRITE_CHANNEL3_WEIGHT	WRITE_CHANNEL2_WEIGHT					WRITE_CHANNEL1_WEIGHT					WRITE_CHANNEL0_WEIGHT				
rw	rw					rw					rw				

Field	Bits	Type	Description
WRITE_CHANNEL0_WEIGHT	4:0	rw	**Channel 0 Weight** The weight is initialized by software before ringing the doorbell. The value is used by the channel weighted round robin arbiter to select the next channel read request. A value of '0' means that one TLP is issued before moving to the next channel. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
WRITE_CHANNEL1_WEIGHT	9:5	rw	**Channel 1 Weight** The weight is initialized by software before ringing the doorbell. The value is used by the channel weighted round robin arbiter to select the next channel read request. A value of '0' means that one TLP is issued before moving to the next channel. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W

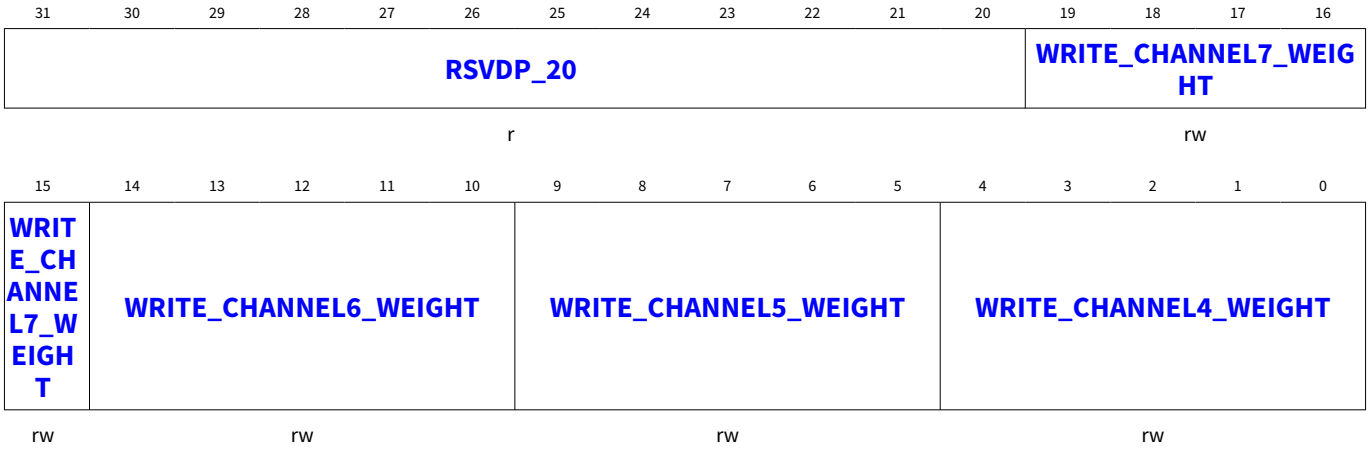
(table continues...)

(continued)

Field	Bits	Type	Description
WRITE_CHANNEL2_WEIGHT	14:10	rw	**Channel 2 Weight** The weight is initialized by software before ringing the doorbell. The value is used by the channel weighted round robin arbiter to select the next channel read request. A value of '0' means that one TLP is issued before moving to the next channel. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
WRITE_CHANNEL3_WEIGHT	19:15	rw	**Channel 3 Weight** The weight is initialized by software before ringing the doorbell. The value is used by the channel weighted round robin arbiter to select the next channel read request. A value of '0' means that one TLP is issued before moving to the next channel. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
RSVDP_20	31:20	r	**Reserved for future use**

19.4.1.311 DMA Write Engine Channel Arbitration Weight High Register

The 5-bit channel weight (for write channels 4-7) specifies the maximum number of TLP requests that the DMA can issue for that channel before it must return to the arbitration routine. When the channel weight count is reached or DMA channel request transfer size reaches zero, the WWR arbiter selects the next channel to be processed. Your software must initialize this register before ringing the doorbell. For more details, see "Multichannel Arbitration". Value range is (0-0x1F) corresponding to (1-32) transaction requests.



Field	Bits	Type	Description
WRITE_CHANNEL4_WEIGHT	4:0	rw	**Channel 4 Weight** The weight is initialized by software before ringing the doorbell. The value is used by the channel weighted round robin arbiter to select the next channel read request. A value of '0' means that one TLP is issued before moving to the next channel. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W

(table continues...)

(continued)

Field	Bits	Type	Description
WRITE_CHANNEL5_WEIGHT	9:5	rw	**Channel 5 Weight** The weight is initialized by software before ringing the doorbell. The value is used by the channel weighted round robin arbiter to select the next channel read request. A value of '0' means that one TLP is issued before moving to the next channel. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
WRITE_CHANNEL6_WEIGHT	14:10	rw	**Channel 6 Weight** The weight is initialized by software before ringing the doorbell. The value is used by the channel weighted round robin arbiter to select the next channel read request. A value of '0' means that one TLP is issued before moving to the next channel. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
WRITE_CHANNEL7_WEIGHT	19:15	rw	**Channel 7 Weight** The weight is initialized by software before ringing the doorbell. The value is used by the channel weighted round robin arbiter to select the next channel read request. A value of '0' means that one TLP is issued before moving to the next channel. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
RSVDP_20	31:20	r	**Reserved for future use**

19.4.1.312 DMA Read Engine Enable Register

This register indicates the status of DMA read engine and the status of DMA read engine handshake feature (per-channel).

DMA_READ_ENGINE_EN_OFF Offset address: $000702C_H$

DMA Read Engine Enable Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_24								Res				DMA_READ_ENGINE_EN_HSHAKE_CH3	DMA_READ_ENGINE_EN_HSHAKE_CH2	DMA_READ_ENGINE_EN_HSHAKE_CH1	DMA_READ_ENGINE_EN_HSHAKE_CH0
r								r				rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVDP_1															DMA_READ_ENGINE
r															rwh

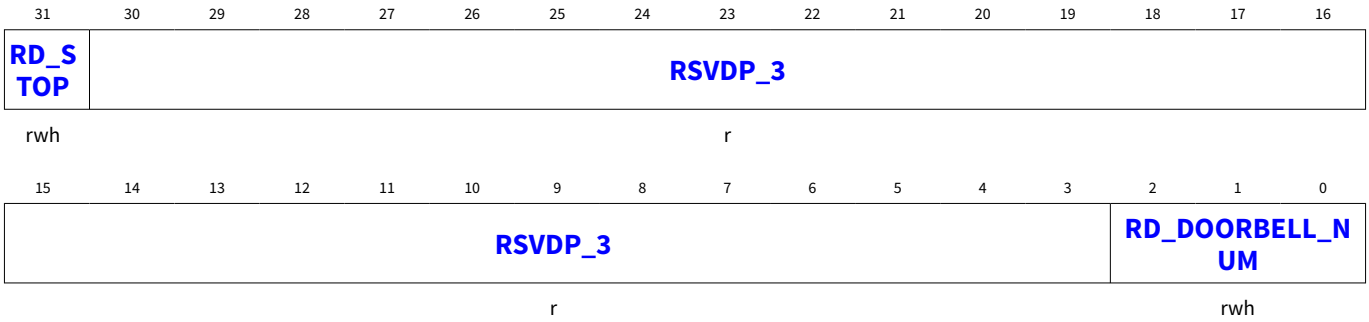
Field	Bits	Type	Description
DMA_READ_ENGINE	0	rwh	**DMA Read Engine Enable** For normal operation, you must initially set this bit to '1', before any other software setup action. You do not need to toggle or rewrite to this bit during normal operation. You should set this field to '0' when you want to "Soft Reset" the DMA controller read logic. There are three possible reasons for resetting the DMA controller read logic: - The "Abort Interrupt Status" bit is set in the DMA Read Interrupt Status Register (DMA_READ_INT_STATUS_OFF), and any of the bits in the DMA Read Error Status Low Register (DMA_READ_ERR_STATUS_LOW_OFF) is set. Resetting the DMA controller read logic re-initializes the control logic, ensuring that the next DMA read transfer is executed successfully. - You have executed the procedure outlined in "Stop Bit", after which, the "Abort Interrupt Status" bit is set and the channel Status field (CS) of the DMA Read Channel Control 1 Register (DMA_CH_CONTROL1_OFF_WRCH_i) is set to "Stopped". Resetting the DMA controller read logic re-initializes the control logic ensuring that the next DMA read transfer is executed successfully. - During software development, when you incorrectly program the DMA read engine. To "Soft Reset" the DMA controller read logic, you must: - De-assert the DMA read engine enable bit. - Wait for the DMA to complete any in-progress TLP transfer, by waiting until a read on the DMA read engine enable bit returns a '0'. - Assert the DMA read engine enable bit. This "Soft Reset" does not clear the DMA configuration registers. The DMA read transfer does not start until you write to the DMA Read Doorbell Register (DMA_READ_DOORBELL_OFF). Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W 0_B **DISABLE**: Disable (Soft Reset) 1_B **ENABLE**: Enable
RSVDP_1	15:1	r	**Reserved for future use**
DMA_READ_ENGINE_EN_HSHAKE_CH0	16	rw	**Enable Handshake for DMA Read Engine Channel 0** Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
DMA_READ_ENGINE_EN_HSHAKE_CH1	17	rw	**Enable Handshake for DMA Read Engine Channel 1** Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
DMA_READ_ENGINE_EN_HSHAKE_CH2	18	rw	**Enable Handshake for DMA Read Engine Channel 2** Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
DMA_READ_ENGINE_EN_HSHAKE_CH3	19	rw	**Enable Handshake for DMA Read Engine Channel 3** Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
RSVDP_24	31:24	r	**Reserved for future use**

19.4.1.313 DMA Read Doorbell Register

This register controls the DMA read channel doorbell.

DMA_READ_DOORBELL_OFF Offset address: 0007030_H

DMA Read Doorbell Register Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
RD_DOORBELL_NUM	2:0	rwh	**Doorbell Number** You must write 0x0 to this register to start the DMA read transfer for that channel. The DMA detects a write to this register field even if the value of this field does not change. The range of this field is 0x0 to 0x7, where 0x0 corresponds to channel 0. Note: A write to this field triggers the controller to exit L1 substates. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
RSVDP_3	30:3	r	**Reserved for future use**
RD_STOP	31	rwh	**Stop** Set in conjunction with the Doorbell Number field. The DMA read channel stops issuing requests, sets the channel status to "Stopped", and asserts the "Abort" interrupt if it is enabled. Before setting the Stop bit, you must read the channel Status field (CS) of the DMA Channel Control 1 Register (DMA_CH_CONTROL1_OFF_RDCH_i) to ensure that the read channel is "Running" (transferring data). For more information, see "Stopping the DMA Transfer (Software Stop)". Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W

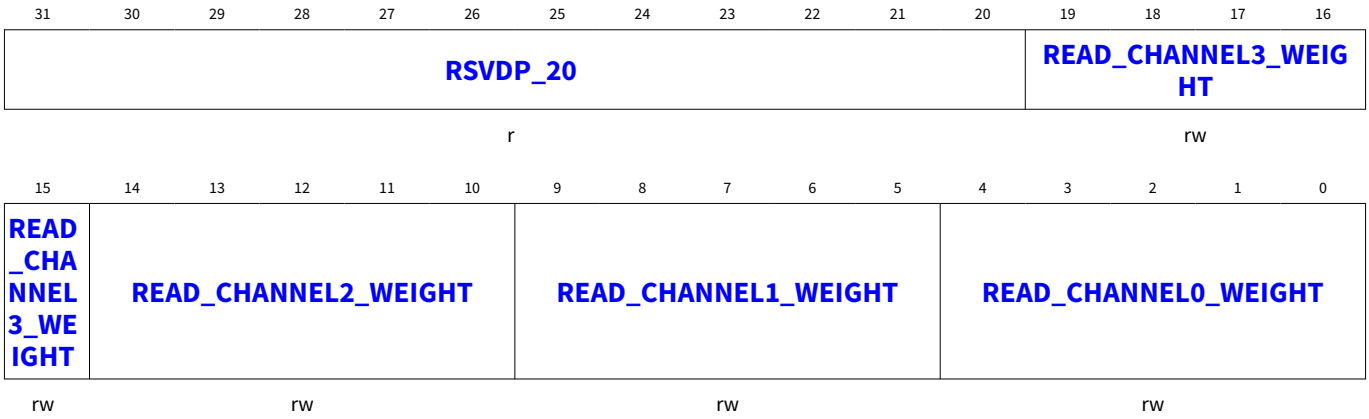
19.4.1.314 DMA Read Engine Channel Arbitration Weight Low Register

The 5-bit channel weight (for read channels 0-3) specifies the maximum number of TLP requests that the DMA can issue for that channel before it must return to the arbitration routine. When the channel weight count is reached or DMA channel request transfer size reaches zero, the WWR arbiter selects the next channel to be processed. Your software must initialize this register before ringing the doorbell. For more details, see "Multichannel Arbitration". Value range is (0-0x1F) corresponding to (1-32) transaction requests.

DMA_READ_CHANNEL_ARB_WEIGHT_LOW_OFF Offset address: 0007038_H

DMA Read Engine Channel Arbitration Weight Low Register

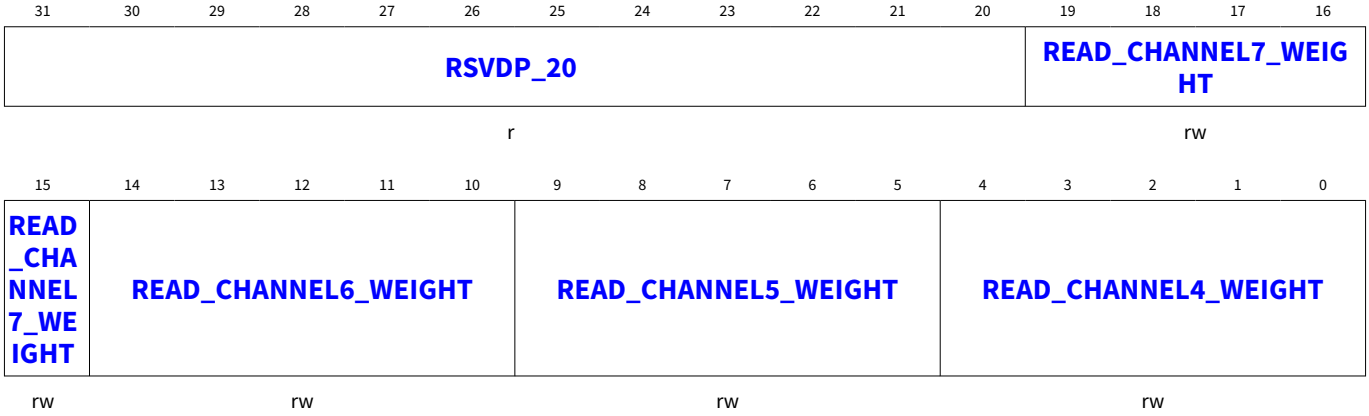
Kernel Reset value: $0000\ 8421_H$



Field	Bits	Type	Description
READ_CHANNEL0_WEIGHT	4:0	rw	**Channel 0 Weight** The weight is initialized by software before ringing the doorbell. The value is used by the channel weighted round robin arbiter to select the next channel read request. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
READ_CHANNEL1_WEIGHT	9:5	rw	**Channel 1 Weight** The weight is initialized by software before ringing the doorbell. The value is used by the channel weighted round robin arbiter to select the next channel read request. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
READ_CHANNEL2_WEIGHT	14:10	rw	**Channel 2 Weight** The weight is initialized by software before ringing the doorbell. The value is used by the channel weighted round robin arbiter to select the next channel read request. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
READ_CHANNEL3_WEIGHT	19:15	rw	**Channel 3 Weight** The weight is initialized by software before ringing the doorbell. The value is used by the channel weighted round robin arbiter to select the next channel read request. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
RSVDP_20	31:20	r	**Reserved for future use**

19.4.1.315 DMA Read Engine Channel Arbitration Weight High Register

The 5-bit channel weight (for read channels 4-7) specifies the maximum number of TLP requests that the DMA can issue for that channel before it must return to the arbitration routine. When the channel weight count is reached or DMA channel request transfer size reaches zero, the WWR arbiter selects the next channel to be processed. Your software must initialize this register before ringing the doorbell. For more details, see "Multichannel Arbitration". Value range is (0-0x1F) corresponding to (1-32) transaction requests.



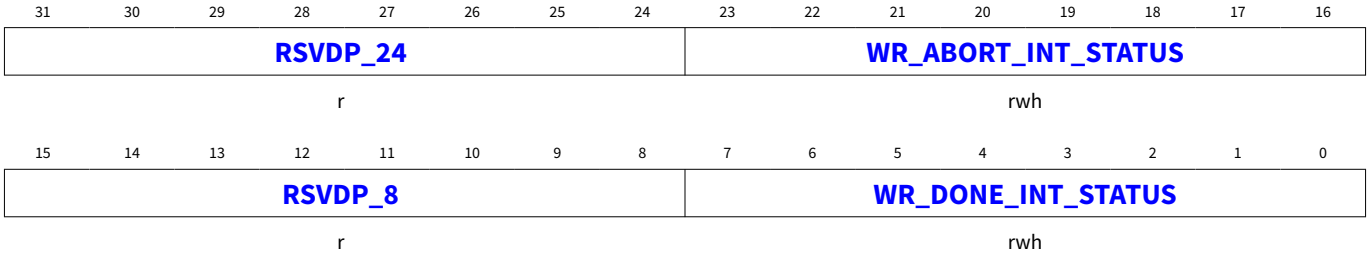
Field	Bits	Type	Description
READ_CHANNEL4_WEIGHT	4:0	rw	**Channel 4 Weight** The weight is initialized by software before ringing the doorbell. The value is used by the channel weighted round robin arbiter to select the next channel read request. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
READ_CHANNEL5_WEIGHT	9:5	rw	**Channel 5 Weight** The weight is initialized by software before ringing the doorbell. The value is used by the channel weighted round robin arbiter to select the next channel read request. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
READ_CHANNEL6_WEIGHT	14:10	rw	**Channel 6 Weight** The weight is initialized by software before ringing the doorbell. The value is used by the channel weighted round robin arbiter to select the next channel read request. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
READ_CHANNEL7_WEIGHT	19:15	rw	**Channel 7 Weight** The weight is initialized by software before ringing the doorbell. The value is used by the channel weighted round robin arbiter to select the next channel read request. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
RSVDP_20	31:20	r	**Reserved for future use**

19.4.1.316 DMA Write Interrupt Status Register

This register indicates the status of Abort and Done interrupts for the DMA write channels.

DMA_WRITE_INT_STATUS_OFF Offset address: $000704C_H$

DMA Write Interrupt Status Register Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
WR_DONE_INT_STATUS	7:0	rwh	**Done Interrupt Status** The DMA write channel has successfully completed the DMA transfer. Each bit corresponds to a DMA channel. Bit [0] corresponds to channel 0. • Enabling: For more details, see "Interrupts and Error Handling" • Masking: The DMA write interrupt Mask register has no effect on this register • Clearing: You must write a 1'b1 to the corresponding channel bit in the DMA write interrupt Clear register to clear this interrupt bit. Note: You can write to this register to emulate interrupt generation, during software or hardware testing. A write to the address triggers an interrupt, but the DMA does not set the Done or Abort bits in this register . Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
RSVDP_8	15:8	r	**Reserved for future use**
WR_ABORT_INT_STATUS	23:16	rwh	**Abort Interrupt Status** The DMA write channel has detected an error, or you manually stopped the transfer as described in "Error Handling Assistance by Remote Software". Each bit corresponds to a DMA channel. Bit [0] corresponds to channel 0. • Enabling: For more details, see "Interrupts and Error Handling" • Masking: The DMA write interrupt Mask register has no effect on this register • Clearing: You must write a 1'b1 to the corresponding channel bit in the DMA write interrupt Clear register to clear this interrupt bit Note: You can write to this register to emulate interrupt generation, during software or hardware testing. A write to the address triggers an interrupt, but the DMA does not set the Done or Abort bits in this register. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
RSVDP_24	31:24	r	**Reserved for future use**

19.4.1.317 DMA Write Interrupt Mask Register

This register holds the Abort and Done DMA write interrupt mask.

DMA_WRITE_INT_MASK_OFF Offset address: 0007054_H

DMA Write Interrupt Mask Register Kernel Reset value: $000F\ 000F_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_24								0				WR_ABORT_INT_MASK			
r								r				rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVDP_8								0				WR_DONE_INT_MASK			
r								r				rw			

Field	Bits	Type	Description
WR_DONE_INT_MASK	3:0	rw	**Done Interrupt Mask** Prevents the Done interrupt status field in the DMA write interrupt status register from asserting the edma_int output. Each bit corresponds to a DMA channel. Bit [0] corresponds to channel 0. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
RSVDP_8	15:8	r	**Reserved for future use**
WR_ABORT_INT_MASK	19:16	rw	**Abort Interrupt Mask** Prevents the Abort interrupt status field in the DMA write interrupt status register from asserting the edma_int output. Each bit corresponds to a DMA channel. Bit [0] corresponds to channel 0. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
RSVDP_24	31:24	r	**Reserved for future use**
0	7:4, 23:20	r	**Reserved** Read as all 0's; should be written with all 0's.

19.4.1.318 DMA Write Interrupt Clear Register

You can write to this register to clear the Abort and Done write interrupts.

DMA_WRITE_INT_CLEAR_OFF Offset address: 0007058_H

DMA Write Interrupt Clear Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_24								0				WR_ABORT_INT_CLEAR			
r								r				rwh			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVDP_8								0				WR_DONE_INT_CLEAR			
r								r				rwh			

Field	Bits	Type	Description
WR_DONE_INT_CLEAR	3:0	rwh	**Done Interrupt Clear** You must write a 1'b1 to clear the corresponding bit in the Done interrupt status field of the DMA write interrupt status register. Each bit corresponds to a DMA channel. Bit [0] corresponds to channel 0. Note: Reading from this self-clearing register field always returns a '0'.
RSVDP_8	15:8	r	**Reserved for future use**
WR_ABORT_INT_CLEAR	19:16	rwh	**Abort Interrupt Clear** You must write a 1'b1 to clear the corresponding bit in the Abort interrupt status field of the DMA write interrupt status register. Each bit corresponds to a DMA channel. Bit [0] corresponds to channel 0. Note: Reading from this self-clearing register field always returns a '0'.
RSVDP_24	31:24	r	**Reserved for future use**
0	7:4, 23:20	r	**Reserved** Read as all 0's; should be written with all 0's.

19.4.1.319 DMA Write Error Status Register

This register provides information pertaining to errors incurred during write channel operation.

DMA_WRITE_ERR_STATUS_OFF Offset address: 000705C$_H$

DMA Write Error Status Register Kernel Reset value: 0000 0000$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_24								LINKLIST_ELEMENT_FETCH_ERR_DETECT							
r								rh							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVDP_8								APP_READ_ERR_DETECT							
r								rh							

Field	Bits	Type	Description
APP_READ_ERR_DETECT	7:0	rh	**Application Read Error Detected** The DMA write channel has received an error response from the AXI bus while reading data from it. Each bit corresponds to a DMA channel. Bit [0] corresponds to channel 0. • Enabling: For more details, see "Interrupts and Error Handling" • Masking: The DMA write interrupt Mask register has no effect on this register • Clearing: You must write a 1'b1 to the corresponding channel bit in the Abort interrupt field of the DMA Write Interrupt Clear Register (DMA_WRITE_INT_CLEAR_OFF) to clear this error bit.
RSVDP_8	15:8	r	**Reserved for future use**
LINKLIST_ELEMENT_FETCH_ERR_DETECT	23:16	rh	**Linked List Element Fetch Error Detected** The DMA write channel has received an error response from the AXI bus while reading a linked list element from local memory. Each bit corresponds to a DMA channel. Bit [0] corresponds to channel 0. • Enabling: For more details, see "Interrupts and Error Handling" • Masking: The DMA write interrupt Mask register has no effect on this register • Clearing: You must write a 1'b1 to the corresponding channel bit in the Abort interrupt field of the DMA Write Interrupt Clear Register (DMA_WRITE_INT_CLEAR_OFF) to clear this error bit.
RSVDP_24	31:24	r	**Reserved for future use**

19.4.1.320 DMA Write Done IMWr Address Low Register

This register holds the lower 32 bits of the Done IMWr TLP address.

DMA_WRITE_DONE_IMWR_LOW_OFF Offset address: 0007060$_H$

DMA Write Done IMWr Address Low Register Kernel Reset value: 0000 0000$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMA_WRITE_DONE_LOW_REG															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMA_WRITE_DONE_LOW_REG															
rw															

Field	Bits	Type	Description
DMA_WRITE_DONE_LOW_REG	31:0	rw	**The DMA uses this field to generate bits [31:0] of the address field for the Done IMWr TLP** Bits [1:0] must be '00' as this address must be dword aligned. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W

19.4.1.321 DMA Write Done IMWr Interrupt Address High Register

This register holds the higher 32 bits of the Done IMWr TLP address.

DMA_WRITE_DONE_IMWR_HIGH_OFF Offset address: 0007064_H

DMA Write Done IMWr Interrupt Address High Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMA_WRITE_DONE_HIGH_REG															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMA_WRITE_DONE_HIGH_REG															
rw															

Field	Bits	Type	Description
DMA_WRITE_DONE_HIGH_REG	31:0	rw	**The DMA uses this field to generate bits [63:32] of the address field for the Done IMWr TLP** Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W

19.4.1.322 DMA Write Abort IMWr Address Low Register

This register holds the lower 32 bits of Abort IMWr TLP.

DMA_WRITE_ABORT_IMWR_LOW_OFF Offset address: 0007068_H

DMA Write Abort IMWr Address Low Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMA_WRITE_ABORT_LOW_REG															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMA_WRITE_ABORT_LOW_REG															
rw															

Field	Bits	Type	Description
DMA_WRITE_ABORT_LOW_REG	31:0	rw	**The DMA uses this field to generate bits [31:0] of the address field for the Abort IMWr TLP it generates** Bits [1:0] must be '00' as this address must be dword aligned. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W

19.4.1.323 DMA Write Abort IMWr Address High Register

This register holds the higher 32 bits of Abort IMWr TLP.

DMA_WRITE_ABORT_IMWR_HIGH_OFF Offset address: $000706C_H$

DMA Write Abort IMWr Address High Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMA_WRITE_ABORT_HIGH_REG															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMA_WRITE_ABORT_HIGH_REG															
rw															

Field	Bits	Type	Description
DMA_WRITE_ABORT_HIGH_REG	31:0	rw	**The DMA uses this field to generate bits [63:32] of the address field for the Abort IMWr TLP** Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W

19.4.1.324 DMA Write Channel 1 and 0 IMWr Data Register

This register holds the Channel 1 and 0 IMWr Done or Abort TLP Data.

DMA_WRITE_CH01_IMWR_DATA_OFF Offset address: 0007070_H

DMA Write Channel 1 and 0 IMWr Data Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
WR_CHANNEL_1_DATA															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WR_CHANNEL_0_DATA															
rw															

Field	Bits	Type	Description
WR_CHANNEL_0_DATA	15:0	rw	**The DMA uses this field to generate the data field for the Done or Abort IMWr TLPs it generates for write channel 0** Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W

(table continues...)

(continued)

Field	Bits	Type	Description
WR_CHANNEL_1_DATA	31:16	rw	**The DMA uses this field to generate the data field for the Done or Abort IMWr TLPs it generates for write channel 1** Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W

19.4.1.325 DMA Write Channel 3 and 2 IMWr Data Register

This register holds the Channel 3 and 2 IMWr Done or Abort TLP Data.

DMA_WRITE_CH23_IMWR_DATA_OFF Offset address: 0007074_H

DMA Write Channel 3 and 2 IMWr Data Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
WR_CHANNEL_3_DATA															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WR_CHANNEL_2_DATA															
rw															

Field	Bits	Type	Description
WR_CHANNEL_2_DATA	15:0	rw	**The DMA uses this field to generate the data field for the Done or Abort IMWr TLPs it generates for write channel 2** Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
WR_CHANNEL_3_DATA	31:16	rw	**The DMA uses this field to generate the data field for the Done or Abort IMWr TLPs it generates for write channel 3** Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W

19.4.1.326 DMA Write Channel 5 and 4 IMWr Data Register

This register holds the Channel 5 and 4 IMWr Done or Abort TLP Data.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
WR_CHANNEL_5_DATA															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WR_CHANNEL_4_DATA															
rw															

Field	Bits	Type	Description
WR_CHANNEL_4_DATA	15:0	rw	**The DMA uses this field to generate the data field for the Done or Abort IMWr TLPs it generates for write channel 4** Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
WR_CHANNEL_5_DATA	31:16	rw	**The DMA uses this field to generate the data field for the Done or Abort IMWr TLPs it generates for write channel 5** Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W

19.4.1.327 DMA Write Channel 7 and 6 IMWr Data Register

This register holds the Channel 7 and 6 IMWr Done or Abort TLP Data.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
WR_CHANNEL_7_DATA															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WR_CHANNEL_6_DATA															
rw															

Field	Bits	Type	Description
WR_CHANNEL_6_DATA	15:0	rw	**The DMA uses this field to generate the data field for the Done or Abort IMWr TLPs it generates for write channel 6** Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
WR_CHANNEL_7_DATA	31:16	rw	**The DMA uses this field to generate the data field for the Done or Abort IMWr TLPs it generates for write channel 7** Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W

19.4.1.328 DMA Write Linked List Error Enable Register

The LIE and RIE bits in the LL element enable the channel "done" interrupts (local and remote). The LLLAIE and LLRAIE bits enable the channel "abort" interrupts (local and remote).

DMA_WRITE_LINKED_LIST_ERR_EN_OFF Offset address: 0007090_H

DMA Write Linked List Error Enable Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_24								0				WR_CHANNEL_LLLAIE			
r								r				rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVDP_8								0				WR_CHANNEL_LLRAIE			
r								r				rw			

Field	Bits	Type	Description
WR_CHANNEL_LLRAIE	3:0	rw	**Write Channel LL Remote Abort Interrupt Enable (LLRAIE)** You enable the write channel remote abort interrupt through this bit. The LIE and RIE bits in the LL element enable the write channel done interrupts. Each bit corresponds to a DMA channel. Bit [0] corresponds to channel 0. Used in linked list mode only. For more details, see "Interrupt Handling". Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
RSVDP_8	15:8	r	**Reserved for future use**
WR_CHANNEL_LLLAIE	19:16	rw	**Write Channel LL Local Abort Interrupt Enable (LLLAIE)** You enable the write channel local abort interrupt through this bit. The LIE and RIE bits in the LL element enable the write channel done interrupts. Each bit corresponds to a DMA channel. Bit [0] corresponds to channel 0. Used in linked list mode only. For more details, see "Interrupt Handling". Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
RSVDP_24	31:24	r	**Reserved for future use**
0	7:4, 23:20	r	**Reserved** Read as all 0's; should be written with all 0's.

19.4.1.329 DMA Read Interrupt Status Register

This register indicates the status of Abort and Done interrupts for the DMA read channels.

DMA_READ_INT_STATUS_OFF Offset address: $00070A0_H$

DMA Read Interrupt Status Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_24								RD_ABORT_INT_STATUS							
r								rwh							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVDP_8								RD_DONE_INT_STATUS							
r								rwh							

Field	Bits	Type	Description
RD_DONE_INT_STATUS	7:0	rwh	**Done Interrupt Status** The DMA read channel has successfully completed the DMA read transfer. Each bit corresponds to a DMA channel. Bit [0] corresponds to channel 0. • Enabling: For more details, see "Interrupts and Error Handling" • Masking: The DMA read interrupt Mask register has no effect on this register • Clearing: You must write a 1'b1 to the corresponding channel bit in the DMA read interrupt Clear register to clear this interrupt bit Note: You can write to this register to emulate interrupt generation, during software or hardware testing. A write to the address triggers an interrupt, but the DMA does not set the Done or Abort bits in this register. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
RSVDP_8	15:8	r	**Reserved for future use**
RD_ABORT_INT_STATUS	23:16	rwh	**Abort Interrupt Status** The DMA read channel has detected an error, or you manually stopped the transfer as described in "Stopping the DMA Transfer (Software Stop)". Each bit corresponds to a DMA channel. Bit [0] corresponds to channel 0. You can read the DMA Read Error Status Low Register (DMA_READ_ERR_STATUS_LOW_OFF) and DMA Read Error Status High Register (DMA_READ_ERR_STATUS_HIGH_OFF) to determine the source of the error. • Enabling: For more details, see "Interrupts and Error Handling" • Masking: The DMA read interrupt Mask register has no effect on this register • Clearing: You must write a 1'b1 to the corresponding channel bit in the DMA read interrupt Clear register to clear this interrupt bit. Note: You can write to this register to emulate interrupt generation, during software or hardware testing. A write to the address triggers an interrupt, but the DMA does not set the Done or Abort bits in this register. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
RSVDP_24	31:24	r	**Reserved for future use**

19.4.1.330 DMA Read Interrupt Mask Register

This register holds the Abort and Done DMA read interrupt mask.

DMA_READ_INT_MASK_OFF Offset address: $00070A8_H$

DMA Read Interrupt Mask Register Kernel Reset value: $000F\ 000F_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_24								0				RD_ABORT_INT_MASK			
r								r				rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVDP_8								0				RD_DONE_INT_MASK			
r								r				rw			

Field	Bits	Type	Description
RD_DONE_INT_MASK	3:0	rw	**Done Interrupt Mask** Prevents the Done interrupt status field in the DMA read interrupt status register from asserting the edma_int output. Each bit corresponds to a DMA channel. Bit [0] corresponds to channel 0. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
RSVDP_8	15:8	r	**Reserved for future use**
RD_ABORT_INT_MASK	19:16	rw	**Abort Interrupt Mask** Prevents the Abort interrupt status field in the DMA read interrupt status register from asserting the edma_int output. Each bit corresponds to a DMA channel. Bit [0] corresponds to channel 0. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
RSVDP_24	31:24	r	**Reserved for future use**
0	7:4, 23:20	r	**Reserved** Read as all 0's; should be written with all 0's.

19.4.1.331 DMA Read Interrupt Clear Register

You can write to this register to clear the Abort and Done read interrupts.

DMA_READ_INT_CLEAR_OFF Offset address: $00070AC_H$

DMA Read Interrupt Clear Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_24								RD_ABORT_INT_CLEAR							
r								rwh							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVDP_8								RD_DONE_INT_CLEAR							
r								rwh							

Field	Bits	Type	Description
RD_DONE_INT_CLEAR	7:0	rwh	**Done Interrupt Clear** You must write a 1'b1 to clear the corresponding bit in the Done interrupt status field of the DMA read interrupt status register. Each bit corresponds to a DMA channel. Bit [0] corresponds to channel 0. Note: Reading from this self-clearing register field always returns a '0'.
RSVDP_8	15:8	r	**Reserved for future use**
RD_ABORT_INT_CLEAR	23:16	rwh	**Abort Interrupt Clear** You must write a 1'b1 to clear the corresponding bit in the Abort interrupt status field of the DMA read interrupt status register. Each bit corresponds to a DMA channel. Bit [0] corresponds to channel 0. Note: Reading from this self-clearing register field always returns a '0'.
RSVDP_24	31:24	r	**Reserved for future use**

19.4.1.332 DMA Read Error Status Low Register

This register provides information pertaining to linked list element fetch error and application write error.

DMA_READ_ERR_STATUS_LOW_OFF Offset address: $00070B4_H$

DMA Read Error Status Low Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_24								LINK_LIST_ELEMENT_FETCH_ERR_DETECT							
r								rh							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVDP_8								APP_WR_ERR_DETECT							
r								rh							

Field	Bits	Type	Description
APP_WR_ERR_DETECT	7:0	rh	**Application Write Error Detected** The DMA read channel has received an error response from the AXI bus while writing data to it. This error is fatal. You must restart the transfer from the beginning, as the channel context is corrupted, and the transfer is not rolled back. For more details, see "Linked List Mode". Each bit corresponds to a DMA channel. Bit [0] corresponds to channel 0. • Enabling: For more details, see "Interrupts and Error Handling" • Masking: The DMA read interrupt Mask register has no effect on this register • Clearing: You must write a 1'b1 to the corresponding channel bit in the Abort interrupt field of the "DMA Read Interrupt Clear Register" (DMA_READ_INT_CLEAR_OFF) to clear this error bit Note, this clears all bits in this register, and also the DMA Read Error Status High register (DMA_READ_ERR_STATUS_HIGH_OFF).
RSVDP_8	15:8	r	**Reserved for future use**
LINK_LIST_ELEMENT_FETCH_ERR_DETECT	23:16	rh	**Linked List Element Fetch Error Detected** - The DMA read channel has received an error response from the AXI bus while reading a linked list element from local memory. Each bit corresponds to a DMA channel. Bit [0] corresponds to channel 0. • Enabling: For details, see "Interrupts and Error Handling" • Masking: The DMA read interrupt Mask register has no effect on this register • Clearing: You must write a 1'b1 to the corresponding channel bit in the Abort interrupt field of the DMA Read Interrupt Clear Register (DMA_READ_INT_CLEAR_OFF) to clear this error bit Note, this clears all bits in this register, and also the DMA Read Error Status High register (DMA_READ_ERR_STATUS_HIGH_OFF).
RSVDP_24	31:24	r	**Reserved for future use**

19.4.1.333 DMA Read Error Status High Register

This register provides information pertaining to completion errors.

DMA_READ_ERR_STATUS_HIGH_OFF Offset address: $00070B8_H$

DMA Read Error Status High Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DATA_POISIONING								CPL_TIMEOUT							
rh								rh							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CPL_ABORT								UNSUPPORTED_REQ							
rh								rh							

Field	Bits	Type	Description
UNSUPPORTED_REQ	7:0	rh	**Unsupported Request** The DMA read channel has received a PCIe unsupported request completion status from the remote device in response to the MRd request. For more details, see "Linked List Mode". Each bit corresponds to a DMA channel. Bit [0] corresponds to channel 0. • Enabling: For more details, see "Interrupts and Error Handling" • Masking: The DMA read interrupt Mask register has no effect on this register • Clearing: You must write a 1'b1 to the corresponding channel bit in the Abort interrupt field of the DMA Read Interrupt Clear Register (DMA_READ_INT_CLEAR_OFF) to clear this error bit. Note, this also clears the other error bits for the same channel in this register and in the DMA Read Error Status Low register.
CPL_ABORT	15:8	rh	**Completer Abort** The DMA read channel has received a PCIe completer abort completion status from the remote device in response to the MRd request. For more details, see "Linked List Mode". Each bit corresponds to a DMA channel. Bit [0] corresponds to channel 0. • Enabling: For more details, see "Interrupts and Error Handling" • Masking: The DMA read interrupt Mask register has no effect on this register • Clearing: You must write a 1'b1 to the corresponding channel bit in the Abort interrupt field of the DMA Read Interrupt Clear Register (DMA_READ_INT_CLEAR_OFF) to clear this error bit. Note, this also clears the other error bits for the same channel in this register and in the DMA Read Error Status Low register.

(table continues...)

(continued)

Field	Bits	Type	Description
CPL_TIMEOUT	23:16	rh	**Completion Time Out** The DMA read channel has timed-out while waiting for the remote device to respond to the MRd request, or a malformed CplD has been received. For more details, see "Linked List Mode". Each bit corresponds to a DMA channel. Bit [0] corresponds to channel 0. • Enabling: For details, see "Interrupts and Error Handling" • Masking: The DMA read interrupt Mask register has no effect on this register • Clearing: You must write a 1'b1 to the corresponding channel bit in the Abort interrupt field of the DMA Read Interrupt Clear Register (DMA_READ_INT_CLEAR_OFF) to clear this error bit Note, this also clears the other error bits for the same channel in this register and in the DMA Read Error Status Low register.
DATA_POISIONING	31:24	rh	**Data Poisoning** The DMA read channel has detected data poisoning in the completion from the remote device (in response to the MRd request). The DMA read channel will drop the completion and then be halted. The CX_FLT_MASK_UR_POIS filter rule does not affect this behavior. Each bit corresponds to a DMA channel. Bit [0] corresponds to channel 0. • Enabling: For more details, see "Interrupts and Error Handling" • Masking: The DMA read interrupt Mask register has no effect on this register • Clearing: You must write a 1'b1 to the corresponding channel bit in the Abort interrupt field of the DMA Read Interrupt Clear Register (DMA_READ_INT_CLEAR_OFF) to clear this error bit Note, this also clears the other error bits for the same channel in this register and in the DMA Read Error Status Low register.

19.4.1.334 DMA Read Linked List Error Enable Register

The LIE and RIE bits in the LL element enable the channel "done" interrupts (local and remote). The LLLAIE and LLRAIE bits enable the channel "abort" interrupts (local and remote).

DMA_READ_LINKED_LIST_ERR_EN_OFF Offset address: $00070C4_H$

DMA Read Linked List Error Enable Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_24								0				RD_CHANNEL_LLLAIE			
r								r				rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVDP_8								0				RD_CHANNEL_LLRAIE			
r								r				rw			

Field	Bits	Type	Description
RD_CHANNEL_LLRAIE	3:0	rw	**Read Channel LL Remote Abort Interrupt Enable (LLRAIE)** You enable the read channel Remote Abort interrupt through this bit. The LIE and RIE bits in the LL element enable the read channel done interrupts. Each bit corresponds to a DMA channel. Bit [0] corresponds to channel 0. Used in linked list mode only. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
RSVDP_8	15:8	r	**Reserved for future use**
RD_CHANNEL_LLLAIE	19:16	rw	**Read Channel LL Local Abort Interrupt Enable (LLLAIE)** You enable the read channel Local Abort interrupt through this bit. The LIE and RIE bits in the LL element enable the read channel done interrupts. Each bit corresponds to a DMA channel. Bit [0] corresponds to channel 0. Used in linked list mode only. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
RSVDP_24	31:24	r	**Reserved for future use**
0	7:4, 23:20	r	**Reserved** Read as all 0's; should be written with all 0's.

19.4.1.335 DMA Read Done IMWr Address Low Register

This register holds the lower 32 bits of the Done IMWr TLP address.

DMA_READ_DONE_IMWR_LOW_OFF Offset address: $00070CC_H$

DMA Read Done IMWr Address Low Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMA_READ_DONE_LOW_REG															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMA_READ_DONE_LOW_REG															
rw															

Field	Bits	Type	Description
DMA_READ_DONE_LOW_REG	31:0	rw	**The DMA uses this field to generate bits [31:0] of the address field for the Done IMWr TLP** Bits [1:0] must be '00' as this address must be dword aligned. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W

19.4.1.336 DMA Read Done IMWr Address High Register

This register holds the higher 32 bits of the Done IMWr TLP address.

DMA_READ_DONE_IMWR_HIGH_OFF Offset address: $00070D0_H$

DMA Read Done IMWr Address High Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMA_READ_DONE_HIGH_REG															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMA_READ_DONE_HIGH_REG															
rw															

Field	Bits	Type	Description
DMA_READ_DONE_HIGH_REG	31:0	rw	**The DMA uses this field to generate bits [63:32] of the address field for the Done IMWr TLP** Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W

19.4.1.337 DMA Read Abort IMWr Address Low Register

This register holds the lower 32 bits of the Abort IMWr TLP address.

DMA_READ_ABORT_IMWR_LOW_OFF Offset address: $00070D4_H$

DMA Read Abort IMWr Address Low Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMA_READ_ABORT_LOW_REG															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMA_READ_ABORT_LOW_REG															
rw															

Field	Bits	Type	Description
DMA_READ_ABORT_LOW_REG	31:0	rw	**The DMA uses this field to generate bits [31:0] of the address field for the Abort IMWr TLP** Bits [1:0] must be '00' as this address must be dword aligned. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W

19.4.1.338 DMA Read Abort IMWr Address High Register

This register holds the higher 32 bits of the Abort IMWr TLP address.

DMA_READ_ABORT_IMWR_HIGH_OFF Offset address: $00070D8_H$

DMA Read Abort IMWr Address High Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMA_READ_ABORT_HIGH_REG															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMA_READ_ABORT_HIGH_REG															
rw															

Field	Bits	Type	Description
DMA_READ_ABORT_HIGH_REG	31:0	rw	**The DMA uses this field to generate bits [63:32] of the address field for the Abort IMWr TLP** Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W

19.4.1.339 DMA Read Channel 1 and 0 IMWr Data Register

This register holds the Channel 1 and 0 IMWr Done or Abort TLP Data.

DMA_READ_CH01_IMWR_DATA_OFF Offset address: $00070DC_H$

DMA Read Channel 1 and 0 IMWr Data Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RD_CHANNEL_1_DATA															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RD_CHANNEL_0_DATA															
rw															

Field	Bits	Type	Description
RD_CHANNEL_0_DATA	15:0	rw	**The DMA uses this field to generate the data field for the Done or Abort IMWr TLPs it generates for read channel 0** Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
RD_CHANNEL_1_DATA	31:16	rw	**The DMA uses this field to generate the data field for the Done or Abort IMWr TLPs it generates for read channel 1** Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W

19.4.1.340 DMA Read Channel 3 and 2 IMWr Data Register

This register holds the Channel 3 and 2 IMWr Done or Abort TLP Data.

DMA_READ_CH23_IMWR_DATA_OFF Offset address: $00070E0_H$

DMA Read Channel 3 and 2 IMWr Data Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RD_CHANNEL_3_DATA															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RD_CHANNEL_2_DATA															
rw															

Field	Bits	Type	Description
RD_CHANNEL_2_DATA	15:0	rw	**The DMA uses this field to generate the data field for the Done or Abort IMWr TLPs it generates for read channel 2** Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
RD_CHANNEL_3_DATA	31:16	rw	**The DMA uses this field to generate the data field for the Done or Abort IMWr TLPs it generates for read channel 3** Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W

19.4.1.341 DMA Read Channel 5 and 4 IMWr Data Register

This register holds the Channel 5 and 4 IMWr Done or Abort TLP Data.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RD_CHANNEL_5_DATA															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RD_CHANNEL_4_DATA															
rw															

Field	Bits	Type	Description
RD_CHANNEL_4_DATA	15:0	rw	**The DMA uses this field to generate the data field for the Done or Abort IMWr TLPs it generates for read channel 4** Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
RD_CHANNEL_5_DATA	31:16	rw	**The DMA uses this field to generate the data field for the Done or Abort IMWr TLPs it generates for read channel 5** Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W

19.4.1.342 DMA Read Channel 7 and 6 IMWr Data Register

This register holds the Channel 7 and 6 IMWr Done or Abort TLP Data.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RD_CHANNEL_7_DATA															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RD_CHANNEL_6_DATA															
rw															

Field	Bits	Type	Description
RD_CHANNEL_6_DATA	15:0	rw	**The DMA uses this field to generate the data field for the Done or Abort IMWr TLPs it generates for read channel 6** Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
RD_CHANNEL_7_DATA	31:16	rw	**The DMA uses this field to generate the data field for the Done or Abort IMWr TLPs it generates for read channel 7** Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W

19.4.1.343 DMA Write Engine Handshake Counter Channel 0/1/2/3 Register

This register holds the handshake counters write channels 0 - 3.

DMA_WRITE_ENGINE_HSHAKE_CNT_LOW_OFF Offset address: 0007108_H

DMA Write Engine Handshake Counter Channel 0/1/2/3 Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_29			DMA_WRITE_ENGINE_HSHAKE_CNT_CH3					RSVDP_21			DMA_WRITE_ENGINE_HSHAKE_CNT_CH2				
r			r					r			r				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVDP_13			DMA_WRITE_ENGINE_HSHAKE_CNT_CH1					RSVDP_5			DMA_WRITE_ENGINE_HSHAKE_CNT_CH0				
r			r					r			r				

Field	Bits	Type	Description
DMA_WRITE_ENGINE_HSHAKE_CNT_CH0	4:0	r	**DMA handshake counter for DMA Write Engine Channel 0** The data transfer in Linked List mode starts only when the counter is non-zero.
RSVDP_5	7:5	r	**Reserved for future use**
DMA_WRITE_ENGINE_HSHAKE_CNT_CH1	12:8	r	**DMA handshake counter for DMA Write Engine Channel 1** The data transfer in Linked List mode starts only when the counter is non-zero.
RSVDP_13	15:13	r	**Reserved for future use**
DMA_WRITE_ENGINE_HSHAKE_CNT_CH2	20:16	r	**DMA handshake counter for DMA Write Engine Channel 2** The data transfer in Linked List mode starts only when the counter is non-zero.
RSVDP_21	23:21	r	**Reserved for future use**
DMA_WRITE_ENGINE_HSHAKE_CNT_CH3	28:24	r	**DMA handshake counter for DMA Write Engine Channel 3** The data transfer in Linked List mode starts only when the counter is non-zero.
RSVDP_29	31:29	r	**Reserved for future use**

19.4.1.344 DMA Write Engine Handshake Counter Channel 4/5/6/7 Register

This register holds the handshake counters write channels 4 - 7.

31 30 29	28 27 26 25 24	23 22 21	20 19 18 17 16
RSVDP_29	DMA_WRITE_ENGINE_HSHAKE_CNT_CH7	RSVDP_21	DMA_WRITE_ENGINE_HSHAKE_CNT_CH6
r	r	r	r

15 14 13	12 11 10 9 8	7 6 5	4 3 2 1 0
RSVDP_13	DMA_WRITE_ENGINE_HSHAKE_CNT_CH5	RSVDP_5	DMA_WRITE_ENGINE_HSHAKE_CNT_CH4
r	r	r	r

Field	Bits	Type	Description
DMA_WRITE_ENGINE_HSHAKE_CNT_CH4	4:0	r	**DMA handshake counter for DMA Write Engine Channel 4** The data transfer in Linked List mode starts only when the counter is non-zero.
RSVDP_5	7:5	r	**Reserved for future use**
DMA_WRITE_ENGINE_HSHAKE_CNT_CH5	12:8	r	**DMA handshake counter for DMA Write Engine Channel 5** The data transfer in Linked List mode starts only when the counter is non-zero.
RSVDP_13	15:13	r	**Reserved for future use**
DMA_WRITE_ENGINE_HSHAKE_CNT_CH6	20:16	r	**DMA handshake counter for DMA Write Engine Channel 6** The data transfer in Linked List mode starts only when the counter is non-zero.
RSVDP_21	23:21	r	**Reserved for future use**
DMA_WRITE_ENGINE_HSHAKE_CNT_CH7	28:24	r	**DMA handshake counter for DMA Write Engine Channel 7** The data transfer in Linked List mode starts only when the counter is non-zero.
RSVDP_29	31:29	r	**Reserved for future use**

19.4.1.345 DMA Read Engine Handshake Counter Channel 0/1/2/3 Register

This register holds the handshake counters read channels 0 - 3.

DMA_READ_ENGINE_HSHAKE_CNT_LOW_OFF Offset address: 0007118_H

DMA Read Engine Handshake Counter Channel 0/1/2/3 Register — Kernel Reset value: $0000\ 0000_H$

31 30 29	28 27 26 25 24	23 22 21	20 19 18 17 16
RSVDP_29	DMA_READ_ENGINE_HSHAKE_CNT_CH3	RSVDP_21	DMA_READ_ENGINE_HSHAKE_CNT_CH2
r	r	r	r

15 14 13	12 11 10 9 8	7 6 5	4 3 2 1 0
RSVDP_13	DMA_READ_ENGINE_HSHAKE_CNT_CH1	RSVDP_5	DMA_READ_ENGINE_HSHAKE_CNT_CH0
r	r	r	r

Field	Bits	Type	Description
DMA_READ_ENGINE_HSHAKE_CNT_CH0	4:0	r	**DMA handshake counter for DMA Read Engine Channel 0** Tthe data transfer in Linked List mode starts only when the counter is non-zero.
RSVDP_5	7:5	r	**Reserved for future use**
DMA_READ_ENGINE_HSHAKE_CNT_CH1	12:8	r	**DMA handshake counter for DMA Read Engine Channel 1** The data transfer in Linked List mode starts only when the counter is non-zero.
RSVDP_13	15:13	r	**Reserved for future use**
DMA_READ_ENGINE_HSHAKE_CNT_CH2	20:16	r	**DMA handshake counter for DMA Read Engine Channel 2** The data transfer in Linked List mode starts only when the counter is non-zero.
RSVDP_21	23:21	r	**Reserved for future use**
DMA_READ_ENGINE_HSHAKE_CNT_CH3	28:24	r	**DMA handshake counter for DMA Read Engine Channel 3** The data transfer in Linked List mode starts only when the counter is non-zero.
RSVDP_29	31:29	r	**Reserved for future use**

19.4.1.346 DMA Read Engine Handshake Counter Channel 4/5/6/7 Register

This register holds the handshake counters read channels 4 - 7.

31 30 29	28 27 26 25 24	23 22 21	20 19 18 17 16
RSVDP_29	DMA_READ_ENGINE_HSHAKE_CNT_CH7	RSVDP_21	DMA_READ_ENGINE_HSHAKE_CNT_CH6
r	r	r	r

15 14 13	12 11 10 9 8	7 6 5	4 3 2 1 0
RSVDP_13	DMA_READ_ENGINE_HSHAKE_CNT_CH5	RSVDP_5	DMA_READ_ENGINE_HSHAKE_CNT_CH4
r	r	r	r

Field	Bits	Type	Description
DMA_READ_ENGINE_HSHAKE_CNT_CH4	4:0	r	**DMA handshake counter for DMA Read Engine Channel 4** The data transfer in Linked List mode starts only when the counter is non-zero.
RSVDP_5	7:5	r	**Reserved for future use**
DMA_READ_ENGINE_HSHAKE_CNT_CH5	12:8	r	**DMA handshake counter for DMA Read Engine Channel 5** The data transfer in Linked List mode starts only when the counter is non-zero.
RSVDP_13	15:13	r	**Reserved for future use**
DMA_READ_ENGINE_HSHAKE_CNT_CH6	20:16	r	**DMA handshake counter for DMA Read Engine Channel 6** The data transfer in Linked List mode starts only when the counter is non-zero.

(table continues...)

(continued)

Field	Bits	Type	Description
RSVDP_21	23:21	r	**Reserved for future use**
DMA_READ_ENGINE_HSHAKE_CNT_CH7	28:24	r	**DMA handshake counter for DMA Read Engine Channel 7** The data transfer in Linked List mode starts only when the counter is non-zero.
RSVDP_29	31:29	r	**Reserved for future use**

19.4.1.347 DMA Write Channel i Power Enable Register

This register controls power enable/disable for write channel 0 .

DMA_WRITE_CHi_PWR_EN_OFF (i=0-3) Offset address: 0007128_H+i*4

DMA Write Channel i Power Enable Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_1															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVDP_1															DMA_WRITE_CH0_PWR_EN
r															rw

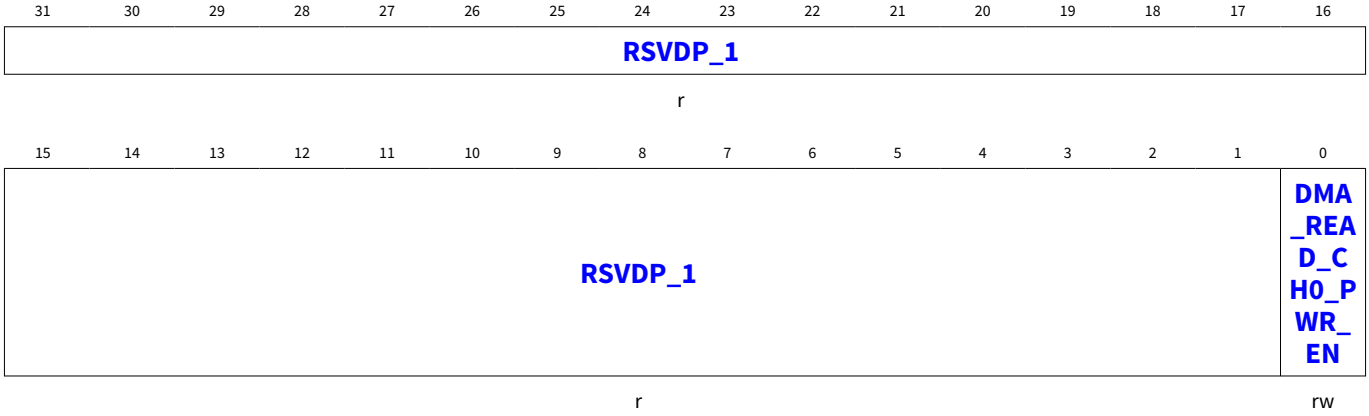
Field	Bits	Type	Description
DMA_WRITE_CH0_PWR_EN	0	rw	**DMA Write Channel 0 Power enable/disable** 0_B **DISABLE**: Disable 1_B **ENABLE**: Enable
RSVDP_1	31:1	r	**Reserved for future use**

19.4.1.348 DMA Read Channel i Power Enable Register

This register controls power enable/disable for read channel 0.

DMA_READ_CHi_PWR_EN_OFF (i=0-3) Offset address: 0007168_H+i*4

DMA Read Channel i Power Enable Register Kernel Reset value: $0000\ 0000_H$



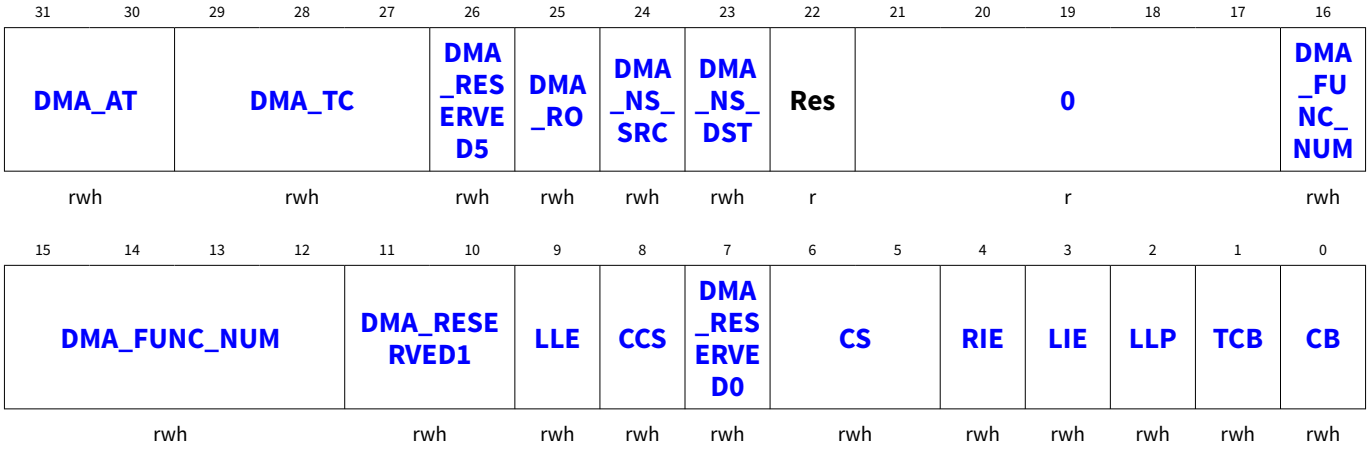
Field	Bits	Type	Description
DMA_READ_CH0_PWR_EN	0	rw	**DMA Read Channel 0 Power enable/disable** 0_B **DISABLE**: Disable 1_B **ENABLE**: Enable
RSVDP_1	31:1	r	**Reserved for future use**

19.4.1.349 DMA Write Channel i Control 1 Register

This register controls the DMA write channel operation.

DMA_CH_CONTROL1_OFF_WRCH_i (i=0-3) Offset address: 0007200_H+i*200_H

DMA Write Channel i Control 1 Register Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
CB	0	rwh	**Cycle Bit (CB)** Used in linked list mode only. It is used to synchronize the producer (software) and the consumer (DMA). For more details, see "PCS-CCS-CB-TCB Producer-Consumer Synchronization". The DMA loads this field with the CB of the linked list element. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W

(table continues...)

(continued)

Field	Bits	Type	Description
TCB	1	rwh	**Toggle Cycle Bit (TCB)** Indicates to the DMA to toggle its interpretation of the CB. Used in linked list mode only. It is used to synchronize the producer (software) and the consumer (DMA). For more details, see "PCS-CCS-CB-TCB Producer-Consumer Synchronization". The DMA loads this field with the TCB of the linked list element. this field is not defined in a data LL element. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
LLP	2	rwh	**Load Link Pointer (LLP)** Used in linked list mode only. Indicates that this linked list element is a link element, and its LL element pointer dwords are pointing to the next (non-contiguous) element. The DMA loads this field with the LLP of the linked list element. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
LIE	3	rwh	**Local Interrupt Enable (LIE)** You must set this bit to enable the generation of the Done or Abort Local interrupts. For more details, see "Interrupts and Error Handling". In LL mode, the DMA overwrites this with the LIE of the LL element. The LIE bit in a LL element only enables the Done interrupt. In non-LL mode, the LIE bit enables the Done and Abort interrupts. This field is not defined in a link LL element. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
RIE	4	rwh	**Remote Interrupt Enable (RIE)** You must set this bit to enable the generation of the Done or Abort Remote interrupts. For more details, see "Interrupts and Error Handling". In LL mode, the DMA overwrites this with the RIE of the LL element. The RIE bit in a LL element only enables the Done interrupt. In non-LL mode, the RIE bit enables the Done and Abort interrupts. This field is not defined in a link LL element. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
CS	6:5	rwh	**Channel Status (CS)** The channel status bits identify the current operational state of the DMA channel. The operation state encoding for each DMA channel is a s follows: 00_B **RESERVED**: Reserved 01_B **RUNNING**: This channel is active and transferring data 10_B **HALTED**: An error condition has been detected, and the DMA has stopped this channel 11_B **STOPPED**: The DMA has transferred all data for this channel, or you have prematurely stopped this channel by writing to the Stop field of the DMA Write Doorbell Register (DMA_WRITE_DOORBELL_OFF) or DMA Read Doorbell Register (DMA_READ_DOORBELL_OFF)
DMA_RESERVED0	7	rwh	**Reserved** Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W

(table continues...)

(continued)

Field	Bits	Type	Description
CCS	8	rwh	**Consumer Cycle State (CCS)** Used in linked list mode only. It is used to synchronize the producer (software) and the consumer (DMA). For more details, see "PCS-CCS-CB-TCB Producer-Consumer Synchronization". You must initialize this bit. The DMA updates this bit during linked list operation. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
LLE	9	rwh	**Linked List Enable (LLE)** Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W 0_B **DISABLE**: Disable linked list operation 1_B **ENABLE**: Enable linked list operation
DMA_RESERVED1	11:10	rwh	**Reserved** Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
DMA_FUNC_NUM	16:12	rwh	**DMA Channel Physical Function Number** This field indicates the physical function associated with the DMA channel. The controller uses this field to form the requester ID of the generated MRd/MWr DMA TLPs. Note: The behavior is undefined if any value outside of the valid range is set for this field Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W 00_H **Min_Val**: Physical Channel Function Number starts at 0 $1F_H$ **Max_Val**: The maximum number of Physical Functions, so the maximum value of this field is 0
DMA_NS_DST	23	rwh	**Destination No Snoop TLP Header Bit** The DMA uses this TLP header field when generating MWr (DAR addressing space) (not IMWr) TLPs. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
DMA_NS_SRC	24	rwh	**Source No Snoop TLP Header Bit** The DMA uses this TLP header field when generating MRd (SAR addressing space) (not IMWr) TLPs. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
DMA_RO	25	rwh	**Relaxed Ordering TLP Header Bit (RO)** DMA uses this TLP header field when generating MRd/MWr (not IMWr) TLPs. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
DMA_RESERVED5	26	rwh	**Reserved** Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
DMA_TC	29:27	rwh	**Traffic Class TLP Header Bit (TC)** The DMA uses this TLP header field when generating MRd/MWr (not IMWr) TLPs. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W

(table continues...)

(continued)

Field	Bits	Type	Description
DMA_AT	31:30	rwh	**Address Translation TLP Header Bit (AT)** The DMA uses this TLP header field when generating MRd/MWr (not IMWr) TLPs. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
0	21:17	r	**Reserved** Read as all 0's; should be written with all 0's.

19.4.1.350 DMA CH i Write Transfer Size Register

This register holds the DMA write transfer size.

DMA_TRANSFER_SIZE_OFF_WRCH_i (i=0-3) Offset address: $0007208_H+i*200_H$

DMA CH i Write Transfer Size Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMA_TRANSFER_SIZE															
rwh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMA_TRANSFER_SIZE															
rwh															

Field	Bits	Type	Description
DMA_TRANSFER_SIZE	31:0	rwh	**DMA Transfer Size** You program this register with the size of the DMA transfer. The maximum DMA transfer size is 4Gbytes. The minimum transfer size is one byte (0x1). This field is automatically decremented by the DMA as the DMA write channel transfer progresses. This field indicates the number bytes remaining to be transferred. When all bytes are successfully transferred the current transfer size is zero. In LL mode, the DMA overwrites this register with the corresponding dword of the LL element. You can read this register to monitor the transfer progress, however in some scenarios this register is updated after a delay. For example, when less than 3 channels are doorbelled, this register is updated only after a descriptor finishes(linked list mode), or the transfer ends (non-linked list mode). Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W

19.4.1.351 DMA CH i Write SAR Low Register

This register holds the lower 32 bits of the Source Address Register (SAR).

DMA_SAR_LOW_OFF_WRCH_i (i=0-3) Offset address: $000720C_H+i*200_H$

DMA CH i Write SAR Low Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SRC_ADDR_REG_LOW															
rwh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SRC_ADDR_REG_LOW															
rwh															

Field	Bits	Type	Description
SRC_ADDR_REG_LOW	31:0	rwh	**Source Address Register (lower 32 bits)** Indicates the next address to be read from. The DMA increments the SAR as the DMA transfer progresses. In LL mode, the DMA overwrites this with the corresponding dword of the LL element. - DMA Read: The SAR is the address of the remote memory. - DMA Write: The SAR is the address of the local memory. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W

19.4.1.352 DMA CH i Write SAR High Register

This register holds the higher 32 bits of the Source Address Register (SAR).

DMA_SAR_HIGH_OFF_WRCH_i (i=0-3) Offset address: $0007210_H+i*200_H$

DMA CH i Write SAR High Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SRC_ADDR_REG_HIGH															
rwh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SRC_ADDR_REG_HIGH															
rwh															

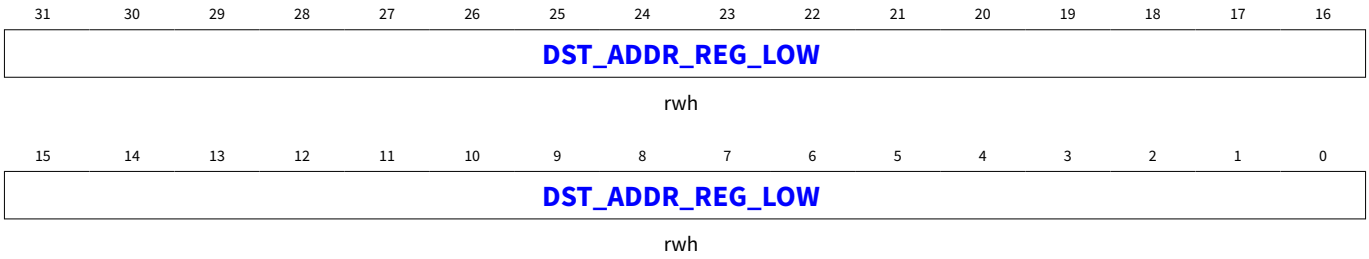
Field	Bits	Type	Description
SRC_ADDR_REG_HIGH	31:0	rwh	**Source Address Register (higher 32 bits)** In LL mode, the DMA overwrites this with the corresponding dword of the LL element. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W

19.4.1.353 DMA CH i Write DAR Low Register

This register holds the lower 32 bits of the Destination Address Register (DAR).

DMA_DAR_LOW_OFF_WRCH_i (i=0-3) Offset address: $0007214_H+i*200_H$

DMA CH i Write DAR Low Register Kernel Reset value: $0000\ 0000_H$



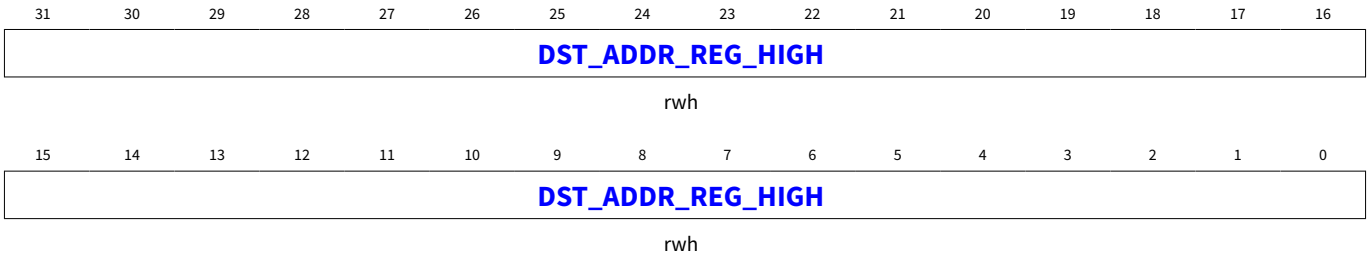
Field	Bits	Type	Description
DST_ADDR_REG_LOW	31:0	rwh	**Destination Address Register (lower 32 bits)** Indicates the next address to be written to. The DMA increments the DAR as the DMA transfer progresses. In LL mode, the DMA overwrites this with the corresponding dword of the LL element. - DMA Read: The DAR is the address of the local memory. - DMA Write: The DAR is the address of the remote memory. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W

19.4.1.354 DMA CH i Write DAR High Register

This register holds the higher 32 bits of the Destination Address Register (DAR).

DMA_DAR_HIGH_OFF_WRCH_i (i=0-3) Offset address: $0007218_H+i*200_H$

DMA CH i Write DAR High Register Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
DST_ADDR_REG_HIGH	31:0	rwh	**Destination Address Register (Higher 32 bits)** In LL mode, the DMA overwrites this with the corresponding dword of the LL element. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W

19.4.1.355 DMA CH i Write Linked List Pointer Low Register

This register holds the lower 32 bits of the DMA write linked list pointer.

DMA_LLP_LOW_OFF_WRCH_i (i=0-3) Offset address: $000721C_H+i*200_H$

DMA CH i Write Linked List Pointer Low Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LLP_LOW															
rwh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LLP_LOW															
rwh															

Field	Bits	Type	Description
LLP_LOW	31:0	rwh	**Lower bits of the address of the linked list transfer list in local memory** Used in linked list mode only. When to point to the next element in the transfer list after the previous element is consumed. • When the current element is a data element; this field is incremented by 6 DWORDs. • When the current element is a link element; this field is overwritten by the LL Element Pointer of the element. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W

19.4.1.356 DMA CH i Write Linked List Pointer High Register

This register holds the higher 32 bits of the DMA write linked list pointer.

DMA_LLP_HIGH_OFF_WRCH_i (i=0-3) Offset address: $0007220_H+i*200_H$

DMA CH i Write Linked List Pointer High Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LLP_HIGH															
rwh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LLP_HIGH															
rwh															

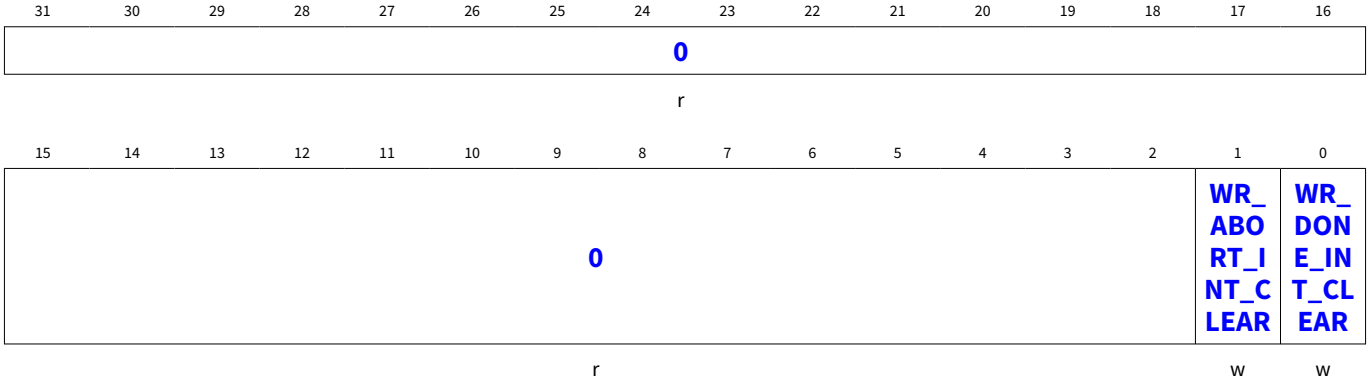
Field	Bits	Type	Description
LLP_HIGH	31:0	rwh	**Higher 32 bits of the address of the linked list transfer list in local memory** Used in linked list mode only. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W

19.4.1.357 Interrupt clear for DMA write channel i

This register has dedicated control over a DMA channel and it has the same functionality as DMA Write Interrupt Clear Register.

DMACAP_CHi_WRCHN_WR_INT_CLEAR (i=0-3) Offset address: $0007224_H+i*200_H$

Interrupt clear for DMA write channel i Kernel Reset value: $0000\ 0000_H$



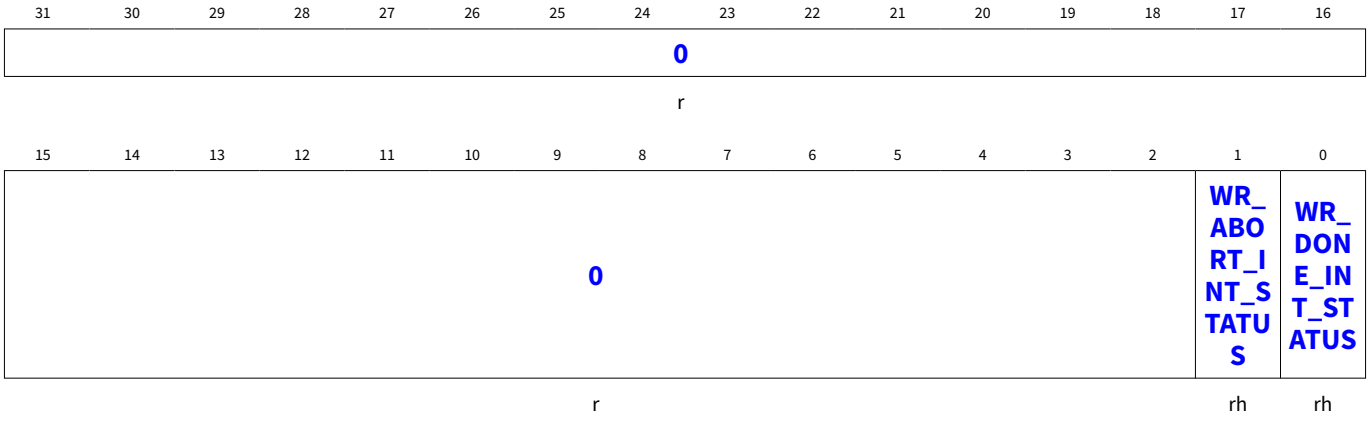
Field	Bits	Type	Description
WR_DONE_INT_CLEAR	0	w	**Done Interrupt Clear for the DMA channel i** You must write a 1'b1 to clear the corresponding bit in the Done interrupt status field of the DMA write interrupt status register. Note: Reading from this self-clearing register field always returns a '0'. 0_B No effect 1_B Clear status
WR_ABORT_INT_CLEAR	1	w	**Abort Interrupt Clear for the DMA channel i** You must write a 1'b1 to clear the corresponding bit in the Abort interrupt status field of the DMA write interrupt status register. Note: Reading from this self-clearing register field always returns a '0'. 0_B No effect 1_B Clear status
0	31:2	r	**Reserved** Read as 0; should be written with 0.

19.4.1.358 Interrupt status for DMA write channel i

This register has dedicated control over a DMA channel and it has the same functionality as DMA Write Interrupt Status Register.

DMACAP_CHi_WRCHN_WR_INT_STATUS (i=0-3) Offset address: $0007228_H+i*200_H$

Interrupt status for DMA write channel i Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
WR_DONE_INT_STATUS	0	rh	**Done Interrupt Status for the DMA channel i** The DMA write channel has successfully completed the DMA transfer.
WR_ABORT_INT_STATUS	1	rh	**Abort Interrupt Status for the DMA channel i** The DMA write channel has detected an error, or you manually stopped the transfer.
0	31:2	r	**Reserved** Read as 0; should be written with 0.

19.4.1.359 Error status for DMA write channel i

This register has dedicated control over a DMA channel and it has the same functionality as DMA Write Error Status Register.

DMACAP_CHi_WRCHN_WR_ERR_STATUS (i=0-3) Offset address: $000722C_H+i*200_H$

Error status for DMA write channel i Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0														LINKLIST_ELEMENT_FETCH_ERR_DETECT	APP_READ_ERR_DETECT
r														rh	rh

Field	Bits	Type	Description
APP_READ_ERR_DETECT	0	rh	**Application Read Error Detected for the DMA channel i** The DMA write channel has received an error response from the AXI bus while reading data from it. . - Clearing: You must write a 1'b1 to the WR_ABORT_INT_CLEAR to clear this error bit.
LINKLIST_ELEMENT_FETCH_ERR_DETECT	1	rh	**Linked List Element Fetch Error Detected for the DMA channel i** The DMA write channel has received an error response from the AXI bus while reading a linked list element from local memory.- Clearing: You must write a 1'b1 to the corresponding channel bit in the WR_ABORT_INT_CLEAR to clear this error bit.
0	31:2	r	**Reserved** Read as 0; should be written with 0.

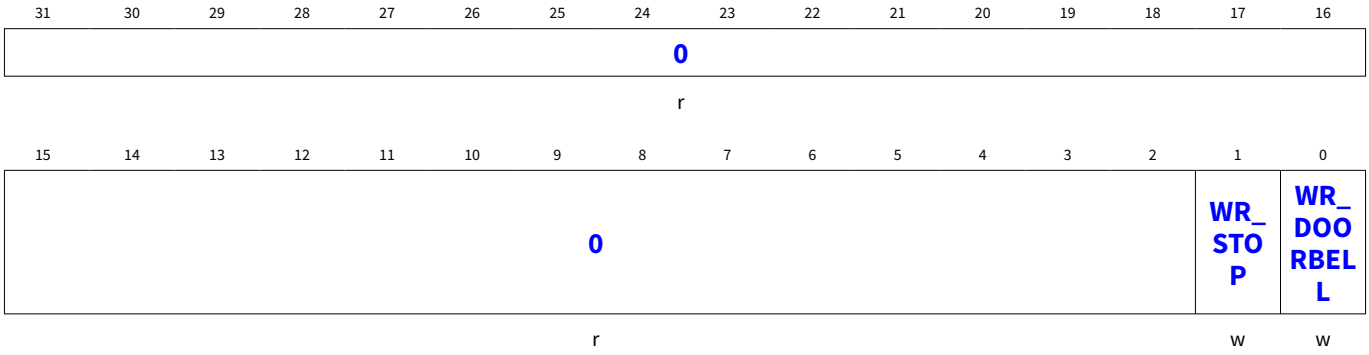
19.4.1.360 Doorbell for DMA write channel i

This register has dedicated control over a DMA channel and it has the same functionality as DMA Write Doorbell Register.

Note: This register is strictly a write only register. Reading this register may result in undefined behavior.

DMACAP_CHi_WRCHN_WR_DOORBELL (i=0-3) Offset address: $0007230_H+i*200_H$

Doorbell for DMA write channel i Kernel Reset value: $0000\ 0000_H$



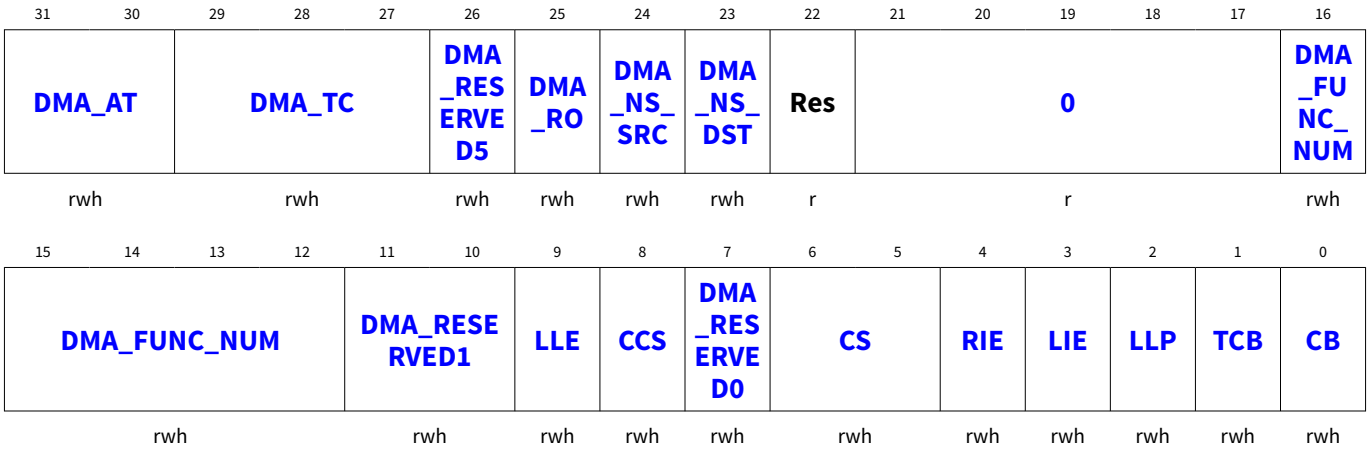
Field	Bits	Type	Description
WR_DOORBELL	0	w	**Doorbell Number for the DMA channel i** You must write the channel number 1'b1 to this register to start the DMA write transfer for that channel. Note: A write to this field triggers the controller to exit L1 substates 0_B No effect 1_B Start DMA transfer
WR_STOP	1	w	**Stop for the DMA channel i** The DMA write channel stops issuing requests, sets the channel status to "Stopped", and asserts the "Abort" interrupt if it is enabled. Before setting the Stop bit, you must read the channel Status field (CS) of the DMA Write Channel Control 1 Register (DMA_CH_CONTROL1_OFF_WRCH_0) to ensure that the write channel is "Running" (transferring data). Note: Use this feature with caution. DMA stop feature is not meant to be used in operational mode. It is only a debug feature and has associated hazards. 0_B No effect 1_B Stop the DMA
0	31:2	r	**Reserved** Read as 0; should be written with 0.

19.4.1.361 DMA Read Channel i Control 1 Register

This register controls the DMA read channel operation.

DMA_CH_CONTROL1_OFF_RDCH_i (i=0-3) Offset address: $0007300_H+i*200_H$

DMA Read Channel i Control 1 Register Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
CB	0	rwh	**Cycle Bit (CB)** Used in linked list mode only. It is used to synchronize the producer (software) and the consumer (DMA). For more details, see "PCS-CCS-CB-TCB Producer-Consumer Synchronization". The DMA loads this field with the CB of the linked list element. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
TCB	1	rwh	**Toggle Cycle Bit (TCB)** Indicates to the DMA to toggle its interpretation of the CB. Used in linked list mode only. It is used to synchronize the producer (software) and the consumer (DMA). For more details, see "PCS-CCS-CB-TCB Producer-Consumer Synchronization". The DMA loads this field with the TCB of the linked list element. This field is not defined in a data LL element. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
LLP	2	rwh	**Load Link Pointer (LLP)** Used in linked list mode only. Indicates that this linked list element is a link element, and its LL element pointer dwords are pointing to the next (non-contiguous) element. The DMA loads this field with the LLP of the linked list element. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
LIE	3	rwh	**Local Interrupt Enable (LIE)** You must set this bit to enable the generation of the Done or Abort Local interrupts. For more details, see "Interrupts and Error Handling". In LL mode, the DMA overwrites this with the LIE of the LL element. The LIE bit in a LL element only enables the Done interrupt. In non-LL mode, the LIE bit enables the Done and Abort interrupts. This field is not defined in a link LL element. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W

(table continues...)

(continued)

Field	Bits	Type	Description
RIE	4	rwh	**Remote Interrupt Enable (RIE)** You must set this bit to enable the generation of the Done or Abort Remote interrupts. For more details, see "Interrupts and Error Handling". In LL mode, the DMA overwrites this with the RIE of the LL element. The RIE bit in a LL element only enables the Done interrupt. In non-LL mode, the RIE bit enables the Done and Abort interrupts. This field is not defined in a link LL element. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
CS	6:5	rwh	**Channel Status (CS)** The channel status bits identify the current operational state of the DMA channel. The operation state encoding for each DMA channel is a s follows: 00_B **RSVD**: Reserved 01_B **RUN**: Running This channel is active and transferring data. 10_B **HALT**: Halted An error condition has been detected, and the DMA has stopped this channel. 11_B **STOP**: Stopped The DMA has transferred all data for this channel, or you have prematurely stopped this channel by writing to the Stop field of the DMA Read Doorbell Register (DMA_WRITE_DOORBELL_OFF) or DMA Read Doorbell Register (DMA_READ_DOORBELL_OFF).
DMA_RESERVED0	7	rwh	**Reserved** Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
CCS	8	rwh	**Consumer Cycle State (CCS)** Used in linked list mode only. It is used to synchronize the producer (software) and the consumer (DMA). For more details, see "PCS-CCS-CB-TCB Producer-Consumer Synchronization". You must initialize this bit. The DMA updates this bit during linked list operation. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
LLE	9	rwh	**Linked List Enable (LLE)** Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W 0_B **DISABLE**: Disable linked list operation 1_B **ENABLE**: Enable linked list operation
DMA_RESERVED1	11:10	rwh	**Reserved** Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W

(table continues...)

(continued)

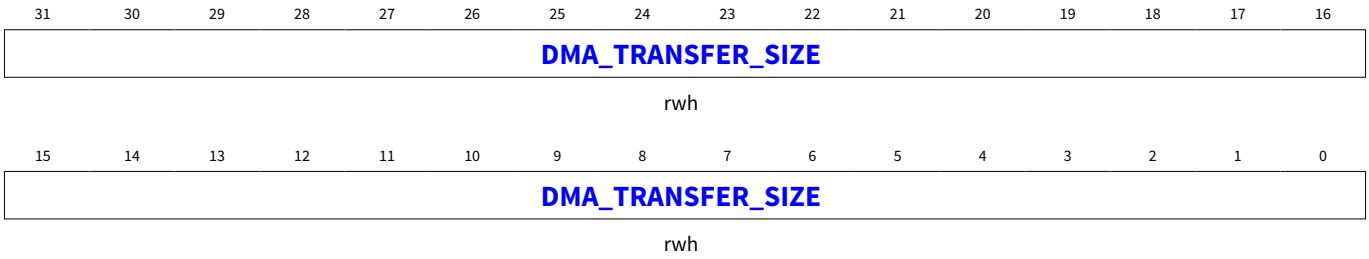
Field	Bits	Type	Description
DMA_FUNC_NUM	16:12	rwh	**DMA Channel Physical Function Number** This field indicates the physical function associated with the DMA channel. The controller uses this field to form the requester ID of the generated MRd/MWr DMA TLPs. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W 00_H **Min_Val**: Physical Channel Function Number starts at 0 $1F_H$ **Max_Val**: The maximum number of Physical Functions, so the maximum value of this field is 0
DMA_NS_DST	23	rwh	**Destination No Snoop TLP Header Bit** The DMA uses this TLP header field when generating MWr (DAR addressing space) (not IMWr) TLPs. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
DMA_NS_SRC	24	rwh	**Source No Snoop TLP Header Bit** The DMA uses this TLP header field when generating MRd (SAR addressing space) (not IMWr) TLPs. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
DMA_RO	25	rwh	**Relaxed Ordering TLP Header Bit (RO) The DMA uses this TLP header field when generating MRd/MWr (not IMWr) TLPs** Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
DMA_RESERVED5	26	rwh	**Reserved** Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
DMA_TC	29:27	rwh	**Traffic Class TLP Header Bit (TC)** The DMA uses this TLP header field when generating MRd/MWr (not IMWr) TLPs. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
DMA_AT	31:30	rwh	**Address Translation TLP Header Bit (AT)** The DMA uses this TLP header field when generating MRd/MWr (not IMWr) TLPs. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
0	21:17	r	**Reserved** Read as all 0's; should be written with all 0's.

19.4.1.362 DMA CH i Read Transfer Size Register

This register holds the DMA read transfer size.

DMA_TRANSFER_SIZE_OFF_RDCH_i (i=0-3) Offset address: $0007308_H+i*200_H$

DMA CH i Read Transfer Size Register Kernel Reset value: $0000\ 0000_H$



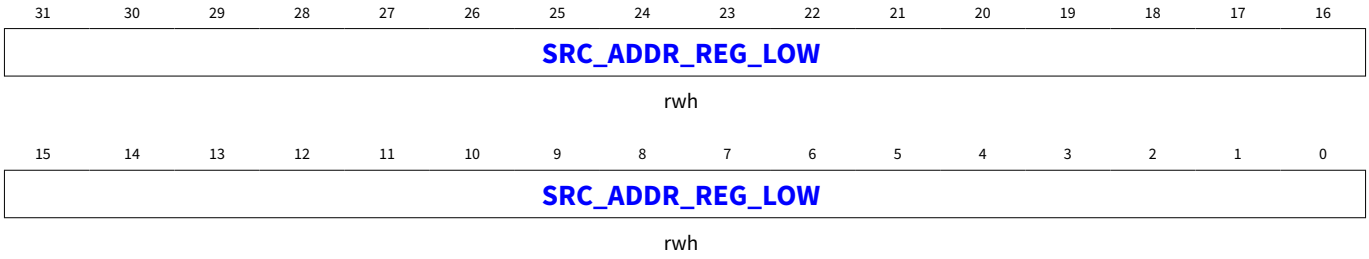
Field	Bits	Type	Description
DMA_TRANSFER_SIZE	31:0	rwh	**DMA Transfer Size** You program this register with the size of the DMA transfer. The maximum DMA transfer size is 4Gbytes. The minimum transfer size is one byte (0x1). This field is automatically decremented by the DMA as the DMA read channel transfer progresses. This field indicates the number bytes remaining to be transferred. When all bytes are successfully transferred the current transfer size is zero. In LL mode, the DMA overwrites this register with the corresponding dword of the LL element. You can read this register to monitor the transfer progress, however in some scenarios this register is updated after a delay. For example, when less than 3 channels are doorbelled, this register is updated only after a descriptor finishes(linked list mode), or the transfer ends (non-linked list mode). Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W

19.4.1.363 DMA CH i Read SAR Low Register

This register holds the lower 32 bits of the Source Address Register (SAR).

DMA_SAR_LOW_OFF_RDCH_i (i=0-3) Offset address: $000730C_H+i*200_H$

DMA CH i Read SAR Low Register Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
SRC_ADDR_REG_LOW	31:0	rwh	**Source Address Register (Lower 32 bits)** Indicates the next address to be read from. The DMA increments the SAR as the DMA transfer progresses. In LL mode, the DMA overwrites this with the corresponding dword of the LL element. • DMA Read: The SAR is the address of the remote memory • DMA Read: The SAR is the address of the local memory Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W

19.4.1.364 DMA CH i Read SAR High Register

This register holds the higher 32 bits of the Source Address Register (SAR).

DMA_SAR_HIGH_OFF_RDCH_i (i=0-3) Offset address: $0007310_H+i*200_H$

DMA CH i Read SAR High Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SRC_ADDR_REG_HIGH															
rwh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SRC_ADDR_REG_HIGH															
rwh															

Field	Bits	Type	Description
SRC_ADDR_REG_HIGH	31:0	rwh	**Source Address Register (higher 32 bits)** In LL mode, the DMA overwrites this with the corresponding dword of the LL element. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W

19.4.1.365 DMA CH i Read DAR Low Register

This register holds the lower 32 bits of the Destination Address Register (DAR).

DMA_DAR_LOW_OFF_RDCH_i (i=0-3) Offset address: $0007314_H+i*200_H$

DMA CH i Read DAR Low Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DST_ADDR_REG_LOW															
rwh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DST_ADDR_REG_LOW															
rwh															

Field	Bits	Type	Description
DST_ADDR_REG_LOW	31:0	rwh	**Destination Address Register (Lower 32 bits)** Indicates the next address to be written to. The DMA increments the DAR as the DMA transfer progresses. In LL mode, the DMA overwrites this with the corresponding dword of the LL element. • DMA Read: The DAR is the address of the local memory • DMA Read: The DAR is the address of the remote memory Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W

19.4.1.366 DMA CH i Read DAR High Register

This register holds the higher 32 bits of the Destination Address Register (DAR).

DMA_DAR_HIGH_OFF_RDCH_i (i=0-3) Offset address: $0007318_H+i*200_H$

DMA CH i Read DAR High Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DST_ADDR_REG_HIGH															
rwh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DST_ADDR_REG_HIGH															
rwh															

Field	Bits	Type	Description
DST_ADDR_REG_HIGH	31:0	rwh	**Destination Address Register (higher 32 bits)** In LL mode, the DMA overwrites this with the corresponding dword of the LL element. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W

19.4.1.367 DMA CH i Read Linked List Pointer Low Register

This register holds the lower 32 bits of the DMA read linked list pointer.

DMA_LLP_LOW_OFF_RDCH_i (i=0-3) Offset address: $000731C_H+i*200_H$

DMA CH i Read Linked List Pointer Low Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LLP_LOW															
rwh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LLP_LOW															
rwh															

Field	Bits	Type	Description
LLP_LOW	31:0	rwh	**Lower bits of the address of the linked list transfer list in local memory** Used in linked list mode only. When using DMA to point to the next element in the transfer list after the previous element is consumed. • When the current element is a data element; this field is incremented by 6 DWORDs • When the current element is a link element; this field is overwritten by the LL Element Pointer of the element. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W

19.4.1.368 DMA CH i Read Linked List Pointer High Register

This register holds the higher 32 bits of the DMA read linked list pointer.

DMA_LLP_HIGH_OFF_RDCH_i (i=0-3) Offset address: $0007320_H+i*200_H$

DMA CH i Read Linked List Pointer High Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LLP_HIGH															
rwh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LLP_HIGH															
rwh															

Field	Bits	Type	Description
LLP_HIGH	31:0	rwh	**Higher 32 bits of the address of the linked list transfer list in local memory** Used in linked list mode only. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W

19.4.1.369 Interrupt clear for DMA read channel i

This register has dedicated control over a DMA channel and it has the same functionality as DMA Read Interrupt Clear Register.

DMACAP_CHi_RDCHN_RD_INT_CLEAR (i=0-3) Offset address: $0007324_H+i*200_H$

Interrupt clear for DMA read channel i Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0														RD_ABORT_INT_CLEAR	RD_DONE_INT_CLEAR
r														w	w

Field	Bits	Type	Description
RD_DONE_INT_CLEAR	0	w	**Done Interrupt Clear for the DMA channel i** You must write a 1'b1 to clear the corresponding bit in the Done interrupt status field of the DMA read interrupt status register. Note: Reading from this self-clearing register field always returns a '0'. 0_B No effect 1_B Clear status

(table continues...)

(continued)

Field	Bits	Type	Description
RD_ABORT_INT_CLEAR	1	w	**Abort Interrupt Clear for the DMA channel i** You must write a 1'b1 to clear the corresponding bit in the Abort interrupt status field of the DMA read interrupt status register.Note: Reading from this self-clearing register field always returns a '0'. 0_B No effect 1_B Clear status
0	31:2	r	**Reserved** Read as 0; should be written with 0.

19.4.1.370 Interrupt status for DMA read channel i

This register has dedicated control over a DMA channel and it has the same functionality as DMA Read Interrupt Status Register.

DMACAP_CHi_RDCHN_RD_INT_STATUS (i=0-3) Offset address: $0007328_H+i*200_H$

Interrupt status for DMA read channel i Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0														RD_ABORT_INT_STATUS	RD_DONE_INT_STATUS
r														rh	rh

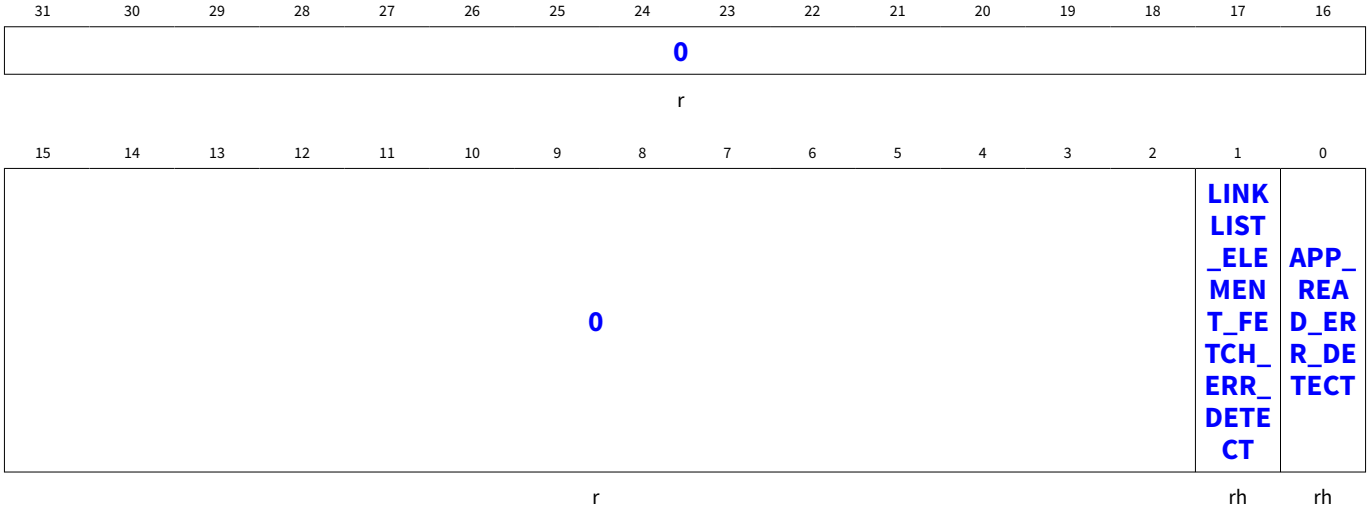
Field	Bits	Type	Description
RD_DONE_INT_STATUS	0	rh	**Done Interrupt Status for the DMA channel i** The DMA read channel has successfully completed the DMA transfer.
RD_ABORT_INT_STATUS	1	rh	**Abort Interrupt Status for the DMA channel i** The DMA read channel has detected an error, or you manually stopped the transfer.
0	31:2	r	**Reserved** Read as 0; should be written with 0.

19.4.1.371 Error status for DMA read channel i

This register has dedicated control over a DMA channel and it has the same functionality as DMA Read Error Status Low Register.

DMACAP_CHi_RDCHN_RD_ERR_STATUS (i=0-3) Offset address: $000732C_H+i*200_H$

Error status for DMA read channel i Kernel Reset value: $0000\ 0000_H$



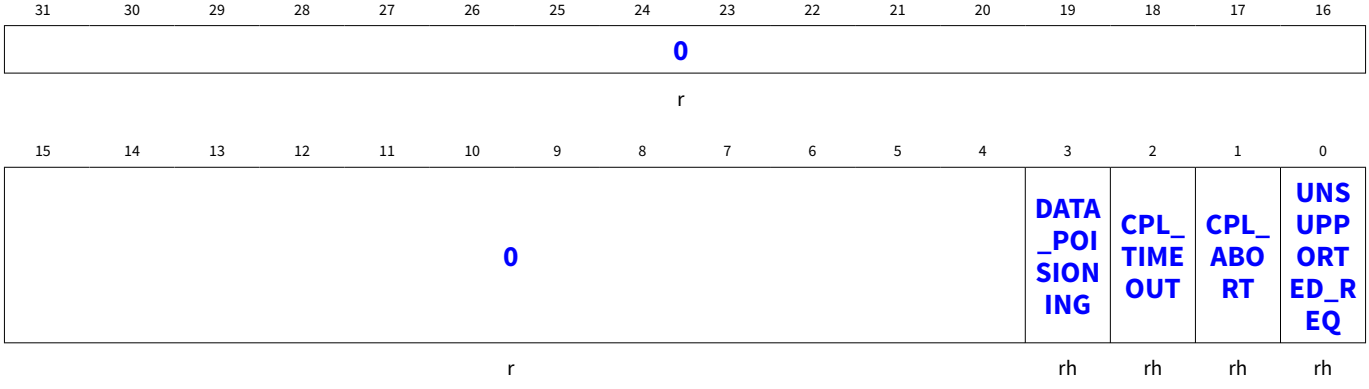
Field	Bits	Type	Description
APP_READ_ERR_DETECT	0	rh	**Application Read Error Detected for the DMA channel i** The DMA read channel has received an error response from the AXI bus while reading data from it. . - Clearing: You must write a 1'b1 to the RD_ABORT_INT_CLEAR to clear this error bit.
LINKLIST_ELEMENT_FETCH_ERR_DETECT	1	rh	**Linked List Element Fetch Error Detected for the DMA channel i** The DMA read channel has received an error response from the AXI bus while reading a linked list element from local memory.- Clearing: You must write a 1'b1 to the corresponding channel bit in the RD_ABORT_INT_CLEAR to clear this error bit.
0	31:2	r	**Reserved** Read as 0; should be written with 0.

19.4.1.372 Higher error status for DMA read channel i

This register has dedicated control over a DMA channel and it has the same functionality as DMA Read Error Status High Register.

DMACAP_CHi_RDCHN_RD_ERR_STATUS_HIGH (i=0-3) Offset address: $0007330_H+i*200_H$

Higher error status for DMA read channel i Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
UNSUPPORTED_REQ	0	rh	**Unsupported Request for the DMA channel i** The DMA read channel has received a PCIe unsupported request completion status from the remote device in response to the MRd request. . - Clearing: You must write a 1'b1 to RD_ABORT_INT_CLEAR to clear this error bit. Note, this also clears the other error bits for the same channel in this register and in the DMA Read Error Status Low register.
CPL_ABORT	1	rh	**Completer Abort for the DMA channel i** The DMA read channel has received a PCIe completer abort completion status from the remote device in response to the MRd request.- Clearing: You must write a 1'b1 to the corresponding channel bit in the RD_ABORT_INT_CLEAR to clear this error bit. Note, this also clears the other error bits for the same channel in this register and in the DMA Read Error Status Low register.
CPL_TIMEOUT	2	rh	**Completion Time Out for the DMA channel i** The DMA read channel has timed-out while waiting for the remote device to respond to the MRd request, or a malformed CplD has been received. - Clearing: You must write a 1'b1 to the RD_ABORT_INT_CLEAR to clear this error bit. Note, this also clears the other error bits for the same channel in this register and in the DMA Read Error Status Low register.
DATA_POISIONING	3	rh	**Data Poisoning for the DMA channel i** The DMA read channel has detected data poisoning in the completion from the remote device (in response to the MRd request). The DMA read channel will drop the completion and then be halted. - Clearing: You must write a 1'b1 to the RD_ABORT_INT_CLEAR to clear this error bit. Note, this also clears the other error bits for the same channel in this register and in the DMA Read Error Status Low register.
0	31:4	r	**Reserved** Read as 0; should be written with 0.

19.4.1.373 Doorbell for DMA read channel i

This register has dedicated control over a DMA channel and it has the same functionality as DMA Read Doorbell Register.

Note: This register is strictly a write only register. Reading this register may result in undefined behavior.

DMACAP_CHi_RDCHN_RD_DOORBELL (i=0-3) Offset address: $0007334_H+i*200_H$

Doorbell for DMA read channel i Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0														RD_STOP	RD_DOORBELL
r														w	w

Field	Bits	Type	Description
RD_DOORBELL	0	w	**Doorbell Number for the DMA channel $ii** You must write the channel number 1'b1 to this register to start the DMA write transfer for that channel. Note: A write to this field triggers the controller to exit L1 substates 0_B No effect 1_B Start DMA transfer
RD_STOP	1	w	**Stop for the DMA channel $ii** The DMA read channel stops issuing requests, sets the channel status to "Stopped", and asserts the "Abort" interrupt if it is enabled. Before setting the Stop bit, you must read the channel Status field (CS) of the DMA Write Channel Control 1 Register (DMA_CH_CONTROL1_OFF_WRCH_0) to ensure that the write channel is "Running" (transferring data). Note: Use this feature with caution. DMA stop feature is not meant to be used in operational mode. It is only a debug feature and has associated hazards. 0_B No effect 1_B Stop the DMA
0	31:2	r	**Reserved** Read as 0; should be written with 0.

19.4.1.374 Write access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

S2A_ACCENy_WRA (y=0-7) Offset address: $3FFFE00_H+y*20_H$

Write access enable register A Application Reset value: $1000\ 0003_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access

19.4.1.375 Write access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

S2A_ACCENy_WRB (y=0-7) Offset address: $3FFFE04_H+y*20_H$

Write access enable register B Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN63	EN62	EN61	EN60	EN59	EN58	EN57	EN56	EN55	EN54	EN53	EN52	EN51	EN50	EN49	EN48
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN47	EN46	EN45	EN44	EN43	EN42	EN41	EN40	EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-63)	q-32	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access

19.4.1.376 Read access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

S2A_ACCENy_RDA (y=0-7) Offset address: $3FFFE08_H+y*20_H$

Read access enable register A Application Reset value: $FFFF\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access

19.4.1.377 Read access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

S2A_ACCENy_RDB (y=0-7) Offset address: $3FFFE0C_H+y*20_H$

Read access enable register B Application Reset value: $FFFF\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN63	EN62	EN61	EN60	EN59	EN58	EN57	EN56	EN55	EN54	EN53	EN52	EN51	EN50	EN49	EN48
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN47	EN46	EN45	EN44	EN43	EN42	EN41	EN40	EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-63)	q-32	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access

19.4.1.378 VM access enable register

This register defines which virtual machine encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each VMx.

S2A_ACCENy_VM (y=0-7) Offset address: $3FFFE10_H+y*20_H$

VM access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for VM ID q** This bit enables read access to the access protected region for transactions with the VM ID q. 0_B Disabled for read access 1_B Enabled for read access

(table continues...)

(continued)

Field	Bits	Type	Description
WRq (q=00-07)	q+16	rw	**Write access enable for VM ID q** This bit enables write access to the access protected region for transactions with the VM ID q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

19.4.1.379 PRS access enable register

This register defines which protection register set encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each PRSx.

S2A_ACCENy_PRS (y=0-7) Offset address: $3FFFE14_H+y*20_H$

PRS access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

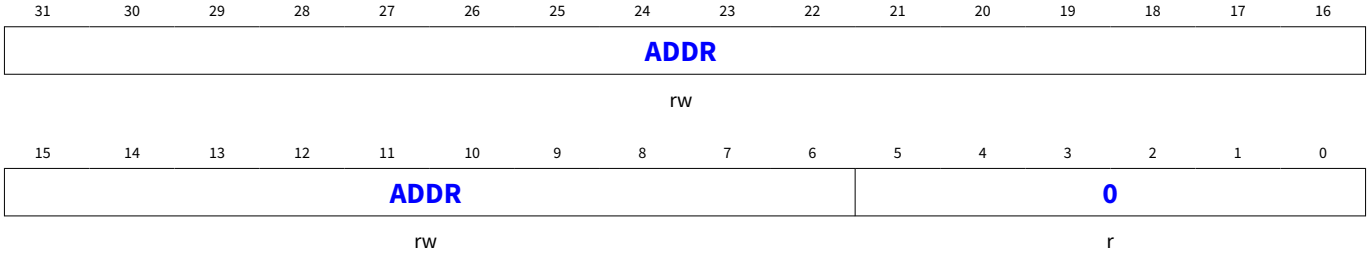
Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for PRS q** This bit enables read access to the access protected region for transactions with the PRS q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for PRS q** This bit enables write access to the access protected region for transactions with the PRS q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

19.4.1.380 Region lower address register

This register sets the lower bound of the access protected region. Addresses greater than or equal to the lower bound are considered part of the access protected region.

S2A_ACCENy_RGNLA (y=0-7) Offset address: $3FFFE18_H+y*20_H$

Region lower address register Application Reset value: $0000\ 0000_H$



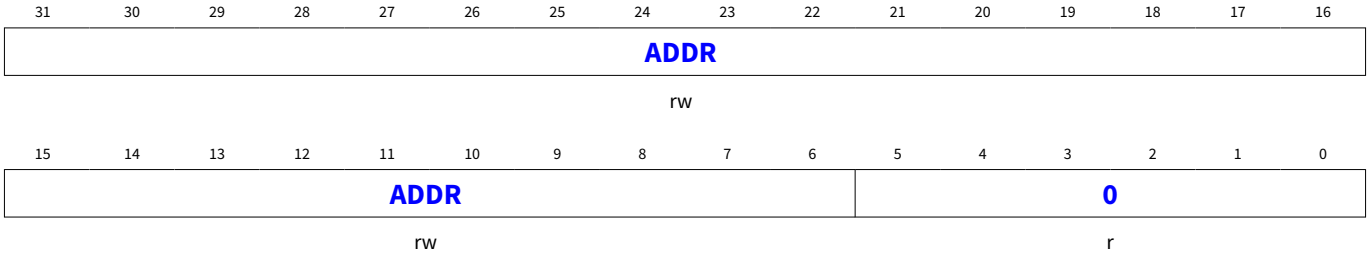
Field	Bits	Type	Description
ADDR	31:6	rw	**Bits 31:6 of the lower bound of the access protected region**
0	5:0	r	**Reserved** Read as 0; should be written with 0.

19.4.1.381 Region upper address register

This register sets the upper bound of the access protected region. Addresses stricly less than the upper bound are considered part of the access protected region.

S2A_ACCENy_RGNUA (y=0-7) Offset address: $3FFFE1C_H+y*20_H$

Region upper address register Application Reset value: $FFFF\ FFC0_H$



Field	Bits	Type	Description
ADDR	31:6	rw	**Bits 31:6 of the upper bound of the access protected region**
0	5:0	r	**Reserved** Read as 0; should be written with 0.

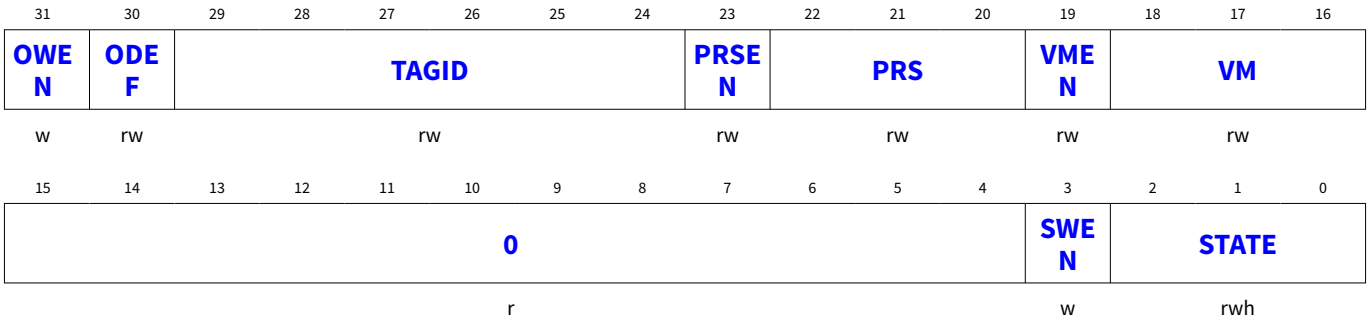
19.4.1.382 PROT Register Safe Endinit

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner and in some PROT states, the secure master to update the PROT state as described in the PROT mechanism.

The PROTSE register controls lock / unlock of the local Safe Endinit (SE) protected control registers.

S2A_PROTSE Offset address: $3FFFF04_H$

PROT Register Safe Endinit Application Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
VM	18:16	rw	**Virtual machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual machine definition enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.
PRSEN	23	rw	**Protection set definition enable for PROT owner** 0_B **Disable**: PRS is not part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG-ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner

(table continues...)

(continued)

Field	Bits	Type	Description
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; should be written with 0.

Table 876 Access types of S2A_PROTSE in order of evaluation

Condition	Type		Description
write 1 to .**SWEN**	rwh	STATE	
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
otherwise	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

19.4.2 Registers of PCIE_DSP

19.4.2.1 Register overview - access mode glossary

Table 877 Register overview - access mode glossary

Keyword	Description
E	Access protection using PROT register PROTE.
SE	Access protection using PROT register PROTSE or S2A_PROTSE.
APU-P	Protection group consisting of registers ACCEN_WRA, ACCEN_WRB, ACCEN_RDA, ACCEN_RDB, ACCEN_VM, ACCEN_PRS.
P	Access protection using *APU-P* registers.
APU-Py (y=0-7)	Protection group consisting of registers S2A_ACCENy_WRA, S2A_ACCENy_WRB, S2A_ACCENy_RDA, S2A_ACCENy_RDB, S2A_ACCENy_VM, S2A_ACCENy_PRS, S2A_ACCENy_RGNLA, S2A_ACCENy_RGNUA.
Py	Access protection using *APU-Py* registers.
APU-P_DMAy (y=0-3)	Protection group consisting of registers DMA_ACCENy_WRA, DMA_ACCENy_WRB, DMA_ACCENy_RDA, DMA_ACCENy_RDB, DMA_ACCENy_VM, DMA_ACCENy_PRS.
P_DMAy	Access protection using *APU-P_DMAy* registers.
BE	Always returns a Bus Error.
U	No access restrictions.
PROT	Access restrictions as defined in the PROT register access rules.

19.4.2.2 Register overview - PCIE_DSP domain FPI (ascending offset address)

Table 878 Register overview - PCIE_DSP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CLC	Clock control register	0000000 H	P	P, SV, E	Application Reset	3218
OCS	OCDS control and status register	0000004 H	P	SV, P	Debug Reset	3219
ID	Module identification register	0000008 H	P	BE	PowerOn Reset	3219
RST_CTRLA	Reset control register A	000000C H	P	P, SV, E	Application Reset	3220
RST_CTRLB	Reset control register B	0000010 H	P	P, SV, E	Application Reset	3220
RST_STAT	Reset status register	0000014 H	P	BE	Application Reset	3221
PROTE	PROT register endinit	0000020 H	U	SV, PROT	Application Reset	3222
PROTSE	PROT register safe endinit	0000024 H	U	SV, PROT	Application Reset	3223
ACCEN_WRA	Write access enable register A	0000040 H	U	SE, SV	Application Reset	3225
ACCEN_WRB	Write access enable register B	0000044 H	U	SE, SV	Application Reset	3225
ACCEN_RDA	Read access enable register A	0000048 H	U	SE, SV	Application Reset	3226
ACCEN_RDB	Read access enable register B	000004C H	U	SE, SV	Application Reset	3226
ACCEN_VM	VM access enable register	0000050 H	U	SE, SV	Application Reset	3227
ACCEN_PRS	PRS access enable register	0000054 H	U	SE, SV	Application Reset	3227
DMA_ACCENy_WRA	CHy Write access enable register A	0000060 H+y*20H	U	SE, SV	Application Reset	3228
DMA_ACCENy_WRB	CHy Write access enable register B	0000064 H+y*20H	U	SE, SV	Application Reset	3229
DMA_ACCENy_RDA	CHy Read access enable register A	0000068 H+y*20H	U	SE, SV	Application Reset	3229
DMA_ACCENy_RDB	CHy Read access enable register B	000006C H+y*20H	U	SE, SV	Application Reset	3230
DMA_ACCENy_VM	CHy VM access enable register	0000070 H+y*20H	U	SE, SV	Application Reset	3230

(table continues...)

Table 878 (continued) Register overview - PCIE_DSP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
DMA_ACCENy_PRS	CHy PRS access enable register	0000074$_H$+y*20$_H$	U	SE, SV	Application Reset	3231
UPCS_OVRD1	UPCS layer override register 1	0000300$_H$	P	P	Kernel Reset	3231
UPCS_OVRD2	UPCS layer override register 2	0000304$_H$	P	P	Kernel Reset	3232
UPCS_OVRD3	UPCS layer override register 3	0000308$_H$	P	P	Kernel Reset	3232
UPCS_OVRD4	UPCS layer override register 4	000030C$_H$	P	P	Kernel Reset	3233
UPCS_OVRD5	UPCS layer override register 5	0000310$_H$	P	P	Kernel Reset	3233
UPCS_OVRD6	UPCS layer override register 6	0000314$_H$	P	P	Kernel Reset	3234
UPCS_OVRD7	UPCS layer override register 7	0000318$_H$	P	P	Kernel Reset	3234
UPCS_OVRD8	UPCS layer override register 8	000031C$_H$	P	P	Kernel Reset	3235
UPCS_OVRD9	UPCS layer override register 9	0000320$_H$	P	P	Kernel Reset	3235
UPCS_OVRD10	UPCS layer override register 10	0000324$_H$	P	P	Kernel Reset	3236
UPCS_OVRD11	UPCS layer override register 11	0000328$_H$	P	P	Kernel Reset	3236
UPCS_OVRD12	UPCS layer override register 12	000032C$_H$	P	P	Kernel Reset	3236
UPCS_OVRD13	UPCS layer override register 13	0000330$_H$	P	P	Kernel Reset	3237
UPCS_OVRD14	UPCS layer override register 14	0000334$_H$	P	P	Kernel Reset	3237
UPCS_OVRD15	UPCS layer override register 15	0000338$_H$	P	P	Kernel Reset	3238
UPCS_OVRD16	UPCS layer override register 16	000033C$_H$	P	P	Kernel Reset	3238
UPCS_OVRD17	UPCS layer override register 17	0000340$_H$	P	P	Kernel Reset	3239
INT_MSICTRL	Request MSI interrupt, EP only.	0000348$_H$	P	P	Kernel Reset	3239
INT_MSISTAT	Pending status of MSI interrupt request, EP only.	000034C$_H$	P	P	Kernel Reset	3240
INT_INTXCTRL	Trigger INTX interrupt, EP only.	0000350$_H$	P	P	Kernel Reset	3240

(table continues...)

Table 878 (continued) Register overview - PCIE_DSP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
INT_TRIGSTAT	Trigger status	0000358H	P	P	Kernel Reset	3241
INT_TRIGCLR	Interrupt clear, EP only.	000035CH	P	P	Kernel Reset	3241
INT_TRIGEN	Interrupt enable EP only.	0000360H	P	P	Kernel Reset	3242
INT_REQTRIG	Receive incoming interrupt request, EP only.	0000364H	P	P	Kernel Reset	3242
CFG_INFO	Configuration information	0000368H	P	P	Kernel Reset	3243
CFG_BUSNUM	Bus/device number	000036CH	P	P	Kernel Reset	3244
LNK_DVCTYP	Select device/port type	0000370H	P	P, E	PowerOn Reset	3245
LNK_STAT	Indicate link status	0000374H	P	P	Kernel Reset	3246
LNK_CORECTRL	Controller control	0000378H	P	P	Kernel Reset	3247
LNK_INTEN	Enable the interrupt requests to SRC_PCIELNK	000037CH	P	P	Kernel Reset	3250
TXRX_RXCTRL	Rx control	0000380H	P	P	Kernel Reset	3250
TXRX_TXCTRL0	Tx control 0	0000384H	P	P	Kernel Reset	3251
PAYLOADz	Message payload z information	000038CH+z*4	P	P	Kernel Reset	3252
MSG_INFO	Message information	0000394H	P	P	Kernel Reset	3253
PM_CTRL	Power management control	0000398H	P	P	Kernel Reset	3254
PM_STAT0	Power management status 0	000039CH	P	P	Kernel Reset	3256
PM_STAT1	Power management status 1	00003A4H	P	P	Kernel Reset	3258
PM_INTCLR	Power management interrupt clear	00003A8H	P	P	Kernel Reset	3258
PM_INTEN	Power management interrupt enable	00003ACH	P	P	Kernel Reset	3259
ERR_STAT	Error status	00003B0H	P	P	Kernel Reset	3260
ERR_INTCLR	Error clear	00003B4H	P	P	Kernel Reset	3260

(table continues...)

Table 878 (continued) Register overview - PCIE_DSP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ERR_INTEN	Error enable	00003B8$_H$	P	P	Kernel Reset	3261
DBG_STAT	Debug status	00003BC$_H$	P	P	Kernel Reset	3262
DBG_DATA	Debug information	00003C0$_H$	P	P	Kernel Reset	3263
LTR_GNRTN	LTR message	00003C4$_H$	P	P	Kernel Reset	3263
LTR_CTRL	LTR control	00003CC$_H$	P	P	Kernel Reset	3263
PTM_CTRL	PTM control	00003D0$_H$	P	P	Kernel Reset	3264
LCLCLKz	Local Clock z value	00003D4$_H$+z*4	P	P	Kernel Reset	3265
CLKCORRz	Clock z correction	00003DC$_H$+z*4	P	P	Kernel Reset	3265
MSKCLKz	PTM z External Master Time	00003E4$_H$+z*4	P	P	Kernel Reset	3266
OBFFMSG	OBFF message control	00003F0$_H$	P	P	Kernel Reset	3267
OBFFWAKE	OBFF WAKE control	00003F4$_H$	P	P	Kernel Reset	3269
CFGBD	PCIe configuration control	00003F8$_H$	P	P	Kernel Reset	3271
DATAx	ELBI x mailbox register	00003FC$_H$+x*4	P	P	Kernel Reset	3271
LLCTRL	DMA link list control register	000041C$_H$	P	P	Kernel Reset	3272
OTSS	PCIe OCDS Trigger Set Select	0000420$_H$	P	SV, P	Debug Reset	3272
ALARM_STAT	Interrupt/alarm status	0000424$_H$	P	P	Kernel Reset	3273
ALARM_INTCLR	Interrupt/alarm clear	0000428$_H$	P	P	Kernel Reset	3275
ALARM_INTEN	Interrupt/alarm enable	000042C$_H$	P	P	Kernel Reset	3276
IATU_REGION_CTRL_1_OFF_OUTBOUND_i	iATU Region i Control 1 Register Outbound	0003000$_H$+i*200$_H$	P	P	Kernel Reset	3277
IATU_REGION_CTRL_2_OFF_OUTBOUND_i	iATU Region i Control 2 Register Outbound	0003004$_H$+i*200$_H$	P	P	Kernel Reset	3278

(table continues...)

Table 878 (continued) Register overview - PCIE_DSP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
IATU_LWR_BASE_ADDR_OFF_OUTBOUND_i	iATU i Lower Base Address Register Outbound	0003008$_H$+i*200$_H$	P	P	Kernel Reset	3280
IATU_UPPER_BASE_ADDR_OFF_OUTBOUND_i	iATU i Upper Base Address Register Outbound	000300C$_H$+i*200$_H$	P	P	Kernel Reset	3281
IATU_LIMIT_ADDR_OFF_OUTBOUND_i	iATU i Limit Address Register Outbound	0003010$_H$+i*200$_H$	P	P	Kernel Reset	3281
IATU_LWR_TARGET_ADDR_OFF_OUTBOUND_i	iATU i Lower Target Address Register Outbound	0003014$_H$+i*200$_H$	P	P	Kernel Reset	3282
IATU_UPPER_TARGET_ADDR_OFF_OUTBOUND_i	iATU i Upper Target Address Register Outbound	0003018$_H$+i*200$_H$	P	P	Kernel Reset	3282
IATU_REGION_CTRL_1_OFF_INBOUND_i	iATU Region i Control 1 Register Inbound	0003100$_H$+i*200$_H$	P	P	Kernel Reset	3283
IATU_REGION_CTRL_2_OFF_INBOUND_i	iATU Region i Control 2 Register Inbound	0003104$_H$+i*200$_H$	P	P	Kernel Reset	3284
IATU_LWR_BASE_ADDR_OFF_INBOUND_i	iATU i Lower Base Address Register Inbound	0003108$_H$+i*200$_H$	P	P	Kernel Reset	3288
IATU_UPPER_BASE_ADDR_OFF_INBOUND_i	iATU i Upper Base Address Register Inbound	000310C$_H$+i*200$_H$	P	P	Kernel Reset	3288
IATU_LIMIT_ADDR_OFF_INBOUND_i	iATU i Limit Address Register Inbound	0003110$_H$+i*200$_H$	P	P	Kernel Reset	3289
IATU_LWR_TARGET_ADDR_OFF_INBOUND_i	iATU i Lower Target Address Register Inbound	0003114$_H$+i*200$_H$	P	P	Kernel Reset	3290
TYPE1_DEV_ID_VEND_ID_REG	Device ID and Vendor ID Register	0004000$_H$	P	P	Kernel Reset	3290
TYPE1_STATUS_COMMAND_REG	Status and Command Register	0004004$_H$	P	P	Kernel Reset	3291
TYPE1_CLASS_CODE_REV_ID_REG	Class Code and Revision ID Register	0004008$_H$	P	P	Kernel Reset	3295
TYPE1_BIST_HDR_TYPE_LAT_CACHE_LINE_SIZE_REG	BIST, Header Type, Latency Timer, and Cache Line Size Register	000400C$_H$	P	P	Kernel Reset	3296

(table continues...)

Table 878 (continued) Register overview - PCIE_DSP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
BARi_REG	BAR0 Register i	0004010$_H$+i*4	P	P	Kernel Reset	3297
SEC_LAT_TIMER_SUB_BUS_SEC_BUS_PRI_BUS_REG	Secondary Latency Timer, Subordinate Bus Number, Secondary Bus Number, and Primary Bus Number Register	0004018$_H$	P	P	Kernel Reset	3299
SEC_STAT_IO_LIMIT_IO_BASE_REG	Secondary Status, and I/O Limit and Base Register	000401C$_H$	P	P	Kernel Reset	3300
MEM_LIMIT_MEM_BASE_REG	Memory Limit and Base Register	0004020$_H$	P	P	Kernel Reset	3303
PREF_MEM_LIMIT_PREF_MEM_BASE_REG	Prefetchable Memory Limit and Base Register	0004024$_H$	P	P	Kernel Reset	3304
PREF_BASE_UPPER_REG	Prefetchable Base Upper 32 Bits Register	0004028$_H$	P	P	Kernel Reset	3306
PREF_LIMIT_UPPER_REG	Prefetchable Limit Upper 32 Bits Register	000402C$_H$	P	P	Kernel Reset	3306
IO_LIMIT_UPPER_IO_BASE_UPPER_REG	I/O Limit and Base Upper 16 Bits Register	0004030$_H$	P	P	Kernel Reset	3307
TYPE1_CAP_PTR_REG	Capabilities Pointer Register	0004034$_H$	P	P	Kernel Reset	3308
BRIDGE_CTRL_INT_PIN_INT_LINE_REG	Bridge Control, Interrupt Pin, and Interrupt Line Register	000403C$_H$	P	P	Kernel Reset	3309
CAP_ID_NXT_PTR_REG	Power Management Capabilities Register	0004040$_H$	P	P	Kernel Reset	3312
CON_STATUS_REG	Power Management Control and Status Register	0004044$_H$	P	P	Kernel Reset	3314
PCI_MSI_CAP_ID_NEXT_CTRL_REG	MSI Capability Header and Message Control Register	0004050$_H$	P	P	Kernel Reset	3317
MSI_CAP_OFF_04H_REG	Message Address Register for MSI (Offset 04h)	0004054$_H$	P	P	Kernel Reset	3319
MSI_CAP_OFF_08H_REG	Message Address Register for MSI (Offset 08h)	0004058$_H$	P	P	Kernel Reset	3320
MSI_CAP_OFF_0CH_REG	Message Address Register for MSI (Offset 0Ch)	000405C$_H$	P	P	Kernel Reset	3321
MSI_CAP_OFF_10H_REG	Message Address Register for MSI (Offset 10h)	0004060$_H$	P	P	Kernel Reset	3322

(table continues...)

Table 878 (continued) Register overview - PCIE_DSP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
MSI_CAP_OFF_14H_REG	Message Address Register for MSI (Offset 14h)	0004064H	P	P	Kernel Reset	3322
PCIE_CAP_ID_PCIE_NEXT_CAP_PTR_PCIE_CAP_REG	PCI Express Capabilities, ID, Next Pointer Register	0004070H	P	P	Kernel Reset	3323
DEVICE_CAPABILITIES_REG	Device Capabilities Register	0004074H	P	P	Kernel Reset	3325
DEVICE_CONTROL_DEVICE_STATUS	Device Control and Device Status Register	0004078H	P	P	Kernel Reset	3327
LINK_CAPABILITIES_REG	Link Capabilities Register	000407CH	P	P	Kernel Reset	3332
LINK_CONTROL_LINK_STATUS_REG	Link Control and Link Status Register	0004080H	P	P	Kernel Reset	3337
SLOT_CAPABILITIES_REG	Slot Capabilities Register	0004084H	P	P	Kernel Reset	3344
SLOT_CONTROL_SLOT_STATUS	Slot Control and Status Register	0004088H	P	P	Kernel Reset	3346
ROOT_CONTROL_ROOT_CAPABILITIES_REG	Root Control and Capabilities Register	000408CH	P	P	Kernel Reset	3351
ROOT_STATUS_REG	Root Status Register	0004090H	P	P	Kernel Reset	3352
DEVICE_CAPABILITIES2_REG	Device Capabilities 2 Register	0004094H	P	P	Kernel Reset	3353
DEVICE_CONTROL2_DEVICE_STATUS2_REG	Device Control 2 and Status 2 Register	0004098H	P	P	Kernel Reset	3356
LINK_CAPABILITIES2_REG	Link Capabilities 2 Register	000409CH	P	P	Kernel Reset	3359
LINK_CONTROL2_LINK_STATUS2_REG	Link Control 2 and Status 2 Register	00040A0H	P	P	Kernel Reset	3361
PCI_MSIX_CAP_ID_NEXT_CTRL_REG	MSI-X Capability ID, Next Pointer, Control Register	00040B0H	P	P	Kernel Reset	3366
MSIX_TABLE_OFFSET_REG	MSI-X Table Offset and BIR Register	00040B4H	P	P	Kernel Reset	3368
MSIX_PBA_OFFSET_REG	MSI-X PBA Offset and BIR Register	00040B8H	P	P	Kernel Reset	3368

(table continues...)

Table 878 (continued) Register overview - PCIE_DSP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
AER_EXT_CAP_HDR_OFF	Advanced Error Reporting Extended Capability Header	0004100H	P	P	Kernel Reset	3369
UNCORR_ERR_STATUS_OFF	Uncorrectable Error Status Register	0004104H	P	P	Kernel Reset	3370
UNCORR_ERR_MASK_OFF	Uncorrectable Error Mask Register	0004108H	P	P	Kernel Reset	3372
UNCORR_ERR_SEV_OFF	Uncorrectable Error Severity Register	000410CH	P	P	Kernel Reset	3374
CORR_ERR_STATUS_OFF	Correctable Error Status Register	0004110H	P	P	Kernel Reset	3376
CORR_ERR_MASK_OFF	Correctable Error Mask Register	0004114H	P	P	Kernel Reset	3377
ADV_ERR_CAP_CTRL_OFF	Advanced Error Capabilities and Control Register	0004118H	P	P	Kernel Reset	3378
HDR_LOG_i_OFF	Header i Log Register 0	000411CH+i*4	P	P	Kernel Reset	3379
AERCAP_ROOT_ROOT_ERR_CMD_OFF	Root Error Command Register	000412CH	P	P	Kernel Reset	3380
AERCAP_ROOT_ROOT_ERR_STATUS_OFF	Root Error Status Register	0004130H	P	P	Kernel Reset	3381
AERCAP_ERR_SRC_ID_OFF	Error Source Identification Register	0004134H	P	P	Kernel Reset	3382
TLP_PREFIX_LOG_i_OFF	TLP Prefix i Log Register 1	0004138H+i*4	P	P	Kernel Reset	3383
SN_BASE	Device Serial Number Extended Capability Header	0004148H	P	P	Kernel Reset	3384
SER_NUM_REG_DW_1	Serial Number 1 Register	000414CH	P	P	Kernel Reset	3385
SER_NUM_REG_DW_2	Serial Number 2 Register	0004150H	P	P	Kernel Reset	3385
SPCIE_CAP_HEADER_REG	SPCIE Capability Header	0004158H	P	P	Kernel Reset	3386
LINK_CONTROL3_REG	Link Control 3 Register	000415CH	P	P	Kernel Reset	3386
LANE_ERR_STATUS_REG	Lane Error Status Register	0004160H	P	P	Kernel Reset	3387
SPCIE_CAP_OFF_0CH_REG	Lane Equalization Control Register for lanes 1 and 0	0004164H	P	P	Kernel Reset	3388
L1SUB_CAP_HEADER_REG	L1 Substates Extended Capability Header	0004170H	P	P	Kernel Reset	3390

(table continues...)

Table 878 (continued) Register overview - PCIE_DSP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
L1SUB_CAPABILITY_REG	L1 Substates Capability Register	0004174H	P	P	Kernel Reset	3391
L1SUB_CONTROL1_REG	L1 Substates Control 1 Register	0004178H	P	P	Kernel Reset	3392
L1SUB_CONTROL2_REG	L1 Substates Control 2 Register	000417CH	P	P	Kernel Reset	3394
RAS_DES_CAP_HEADER_REG	Vendor-Specific Extended Capability Header	0004180H	P	P	Kernel Reset	3395
VENDOR_SPECIFIC_HEADER_REG	Vendor-Specific Header	0004184H	P	P	Kernel Reset	3396
EVENT_COUNTER_CONTROL_REG	Event Counter Control	0004188H	P	P	Kernel Reset	3397
EVENT_COUNTER_DATA_REG	Event Counter Data	000418CH	P	P	Kernel Reset	3399
TIME_BASED_ANALYSIS_CONTROL_REG	Time-based Analysis Control	0004190H	P	P	Kernel Reset	3400
TIME_BASED_ANALYSIS_DATA_REG	Time-based Analysis Data	0004194H	P	P	Kernel Reset	3401
TIME_BASED_ANALYSIS_DATA_63_32_REG	Upper 32 bits of Time-based Analysis Data	0004198H	P	P	Kernel Reset	3402
EINJ_ENABLE_REG	Error Injection Enable	00041B0H	P	P	Kernel Reset	3402
EINJ0_CRC_REG	Error Injection Control 0 (CRC Error)	00041B4H	P	P	Kernel Reset	3404
EINJ1_SEQNUM_REG	Error Injection Control 1 (Sequence Number Error). 	00041B8H	P	P	Kernel Reset	3405
EINJ2_DLLP_REG	Error Injection Control 2 (DLLP Error)	00041BCH	P	P	Kernel Reset	3406
EINJ3_SYMBOL_REG	Error Injection Control 3 (Symbol Error)	00041C0H	P	P	Kernel Reset	3407
EINJ4_FC_REG	Error Injection Control 4 (FC Credit Error)	00041C4H	P	P	Kernel Reset	3408
EINJ5_SP_TLP_REG	Error Injection Control 5 (Specific TLP Error)	00041C8H	P	P	Kernel Reset	3409
EINJ6_COMPARE_POINT_Hi_REG	Error Injection Control 6 (Compare Point Header DWORD #0) i	00041CCH+i*4	P	P	Kernel Reset	3410

(table continues...)

Table 878 (continued) Register overview - PCIE_DSP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
EINJ6_COMPARE_VALUE_Hi_REG	Error Injection Control 6 (Compare Value Header DWORD #0) i	00041DC$_H$+i*4	P	P	Kernel Reset	3411
EINJ6_CHANGE_POINT_Hi_REG	Error Injection Control 6 (Change Point Header DWORD #0) i	00041EC$_H$+i*4	P	P	Kernel Reset	3411
EINJ6_CHANGE_VALUE_Hi_REG	Error Injection Control 6 (Change Value Header DWORD #0) i	00041FC$_H$+i*4	P	P	Kernel Reset	3412
EINJ6_TLP_REG	Error Injection Control 6 (Packet Error)	000420C$_H$	P	P	Kernel Reset	3413
SD_CONTROL1_REG	Silicon Debug Control 1	0004220$_H$	P	P	Kernel Reset	3414
SD_CONTROL2_REG	Silicon Debug Control 2	0004224$_H$	P	P	Kernel Reset	3415
SD_STATUS_L1LANE_REG	Silicon Debug Status(Layer1 Per-lane)	0004230$_H$	P	P	Kernel Reset	3416
SD_STATUS_L1LTSSM_REG	Silicon Debug Status(Layer1 LTSSM)	0004234$_H$	P	P	Kernel Reset	3417
SD_STATUS_PM_REG	Silicon Debug Status(PM)	0004238$_H$	P	P	Kernel Reset	3420
SD_STATUS_L2_REG	Silicon Debug Status(Layer2)	000423C$_H$	P	P	Kernel Reset	3423
SD_STATUS_L3FC_REG	Silicon Debug Status(Layer3 FC)	0004240$_H$	P	P	Kernel Reset	3423
SD_STATUS_L3_REG	Silicon Debug Status(Layer3)	0004244$_H$	P	P	Kernel Reset	3425
SD_EQ_CONTROL1_REG	Silicon Debug EQ Control 1	0004250$_H$	P	P	Kernel Reset	3426
SD_EQ_CONTROL2_REG	Silicon Debug EQ Control 2	0004254$_H$	P	P	Kernel Reset	3428
SD_EQ_CONTROL3_REG	Silicon Debug EQ Control 3	0004258$_H$	P	P	Kernel Reset	3429
SD_EQ_STATUS1_REG	Silicon Debug EQ Status 1	0004260$_H$	P	P	Kernel Reset	3430
SD_EQ_STATUS2_REG	Silicon Debug EQ Status 2	0004264$_H$	P	P	Kernel Reset	3431
SD_EQ_STATUS3_REG	Silicon Debug EQ Status 3	0004268$_H$	P	P	Kernel Reset	3432
RASDP_EXT_CAP_HDR_OFF	PCIe Extended capability ID, Capability version and Next capability offset	0004280$_H$	P	P	Kernel Reset	3433

(table continues...)

Table 878 (continued) Register overview - PCIE_DSP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
RASDP_VENDOR_SPECIFIC_HDR_OFF	Vendor Specific Header	0004284 H	P	P	Kernel Reset	3434
RASDP_ERROR_PROT_CTRL_OFF	ECC Error Correction and Control Register	0004288 H	P	P	Kernel Reset	3435
RASDP_ERROR_MODE_EN_OFF	RASDP error mode enable.	00042A8 H	P	P	Kernel Reset	3447
RASDP_ERROR_MODE_CLEAR_OFF	RASDP Error Mode Clear Register	00042AC H	P	P	Kernel Reset	3448
RASDP_RAM_ADDR_CORR_ERROR_OFF	RAM Address where a corrected error (1-bit ECC) is detected	00042B0 H	P	P	Kernel Reset	3449
RASDP_RAM_ADDR_UNCORR_ERROR_OFF	RAM Address where an uncorrected error (2-bit ECC) is detected	00042B4 H	P	P	Kernel Reset	3449
PTM_EXT_CAP_HDR_OFF	Precision Time Measurement Capability Header	00042B8 H	P	P	Kernel Reset	3450
PTM_CAP_OFF	PTM Capability Register	00042BC H	P	P	Kernel Reset	3451
PTM_CONTROL_OFF	PTM Control Register	00042C0 H	P	P	Kernel Reset	3452
PTM_RES_CAP_HDR_OFF	Precision Time Measurement Responder Capability Header (VSEC)	00042C4 H	P	P	Kernel Reset	3453
PTM_RES_HDR_OFF	Precision Time Measurement Responder Vendor Specific Header	00042C8 H	P	P	Kernel Reset	3454
PTM_RES_CONTROL_OFF	PTM Responder Control Register	00042CC H	P	P	Kernel Reset	3455
PTM_RES_STATUS_OFF	PTM Responder Status Register	00042D0 H	P	P	Kernel Reset	3456
PTM_RES_LOCAL_LSB_OFF	PTM Responder Local Clock LSB	00042D4 H	P	P	Kernel Reset	3457
PTM_RES_LOCAL_MSB_OFF	PTM Responder Local Clock MSB	00042D8 H	P	P	Kernel Reset	3457
PTM_RES_T2_LSB_OFF	PTM Responder T2 Timestamp LSB	00042DC H	P	P	Kernel Reset	3458
PTM_RES_T2_MSB_OFF	PTM Responder T2 Timestamp MSB	00042E0 H	P	P	Kernel Reset	3458
PTM_RES_T2P_LSB_OFF	PTM Responder T2 Previous Timestamp LSB	00042E4 H	P	P	Kernel Reset	3458

(table continues...)

Table 878 (continued) Register overview - PCIE_DSP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PTM_RES_T2P_MSB_OFF	PTM Responder T2 Previous Timestamp MSB	00042E8H	P	P	Kernel Reset	3459
PTM_RES_T3_LSB_OFF	PTM Responder T3 Timestamp LSB	00042ECH	P	P	Kernel Reset	3459
PTM_RES_T3_MSB_OFF	PTM Responder T3 Timestamp MSB	00042F0H	P	P	Kernel Reset	3460
PTM_RES_T3P_LSB_OFF	PTM Responder T3 Previous Timestamp LSB	00042F4H	P	P	Kernel Reset	3460
PTM_RES_T3P_MSB_OFF	PTM Responder T3 Previous Timestamp MSB	00042F8H	P	P	Kernel Reset	3461
PTM_RES_TX_LATENCY_OFF	PTM Responder TX Latency	00042FCH	P	P	Kernel Reset	3461
PTM_RES_RX_LATENCY_OFF	PTM Responder RX Latency	0004300H	P	P	Kernel Reset	3462
PTM_RES_NOM_CLOCK_T_OFF	PTM Responder Nominal Clock Period	0004304H	P	P	Kernel Reset	3463
PTM_RES_SCALED_CLOCK_T_OFF	PTM Responder Scaled Clock Period	0004308H	P	P	Kernel Reset	3463
PTM_RES_LATENCY_REG_SEL_OFF	PTM Responder Latency Viewport Register Select	000430CH	P	P	Kernel Reset	3464
VSECDMA_EXT_CAP_HDR_OFF	PCIe Extended Capability ID, Capability Version, and Next Capability Offset Register	000432CH	P	P	Kernel Reset	3465
VSECDMA_VENDOR_SPECIFIC_HDR_OFF	Vendor Specific Header Register	0004330H	P	P	Kernel Reset	3466
VSECDMA_DEVICE_INFORMATION_OFF	DMA and related AXI Bridge Implementation Information	0004334H	P	P	Kernel Reset	3467
VSECDMA_NUM_CHAN_OFF	Number of Implemented Channels Register	0004338H	P	P	Kernel Reset	3469
VSECDMA_UNROLL_ADDR_OFFSET_LOW_OFF	DMA Register Map Start Address Offset Low Register	000433CH	P	P	Kernel Reset	3469
VSECDMA_UNROLL_ADDR_OFFSET_HIGH_OFF	DMA Register Map Start Address Offset High Register	0004340H	P	P	Kernel Reset	3470
ACK_LATENCY_TIMER_OFF	Ack Latency Timer and Replay Timer Register	0004700H	P	P	Kernel Reset	3470
VENDOR_SPEC_DLLP_OFF	Vendor Specific DLLP Register	0004704H	P	P	Kernel Reset	3471

(table continues...)

Table 878 (continued) Register overview - PCIE_DSP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PORT_FORCE_OFF	Port Force Link Register	0004708H	P	P	Kernel Reset	3471
ACK_F_ASPM_CTRL_OFF	Ack Frequency and L0-L1 ASPM Control Register	000470CH	P	P	Kernel Reset	3473
PORT_LINK_CTRL_OFF	Port Link Control Register	0004710H	P	P	Kernel Reset	3475
LANE_SKEW_OFF	Lane Skew Register	0004714H	P	P	Kernel Reset	3477
TIMER_CTRL_MAX_FUNC_NUM_OFF	Timer Control and Max Function Number Register	0004718H	P	P	Kernel Reset	3479
SYMBOL_TIMER_FILTER_1_OFF	Symbol Timer Register and Filter Mask 1 Register	000471CH	P	P	Kernel Reset	3480
FILTER_MASK_2_OFF	Filter Mask 2 Register	0004720H	P	P	Kernel Reset	3483
PL_DEBUG0_OFF	Debug Register 0	0004728H	P	P	Kernel Reset	3485
PL_DEBUG1_OFF	Debug Register 1	000472CH	P	P	Kernel Reset	3486
TX_P_FC_CREDIT_STATUS_OFF	Transmit Posted FC Credit Status	0004730H	P	P	Kernel Reset	3487
TX_NP_FC_CREDIT_STATUS_OFF	Transmit Non-Posted FC Credit Status	0004734H	P	P	Kernel Reset	3488
TX_CPL_FC_CREDIT_STATUS_OFF	Transmit Completion FC Credit Status	0004738H	P	P	Kernel Reset	3489
QUEUE_STATUS_OFF	Queue Status	000473CH	P	P	Kernel Reset	3490
VC0_P_RX_Q_CTRL_OFF	Segmented-Buffer VC0 Posted Receive Queue Control	0004748H	P	P	Kernel Reset	3492
VC0_NP_RX_Q_CTRL_OFF	Segmented-Buffer VC0 Non-Posted Receive Queue Control	000474CH	P	P	Kernel Reset	3494
VC0_CPL_RX_Q_CTRL_OFF	Segmented-Buffer VC0 Completion Receive Queue Control	0004750H	P	P	Kernel Reset	3495
GEN2_CTRL_OFF	Link Width and Speed Change Control Register	000480CH	P	P	Kernel Reset	3496
MSI_CTRL_ADDR_OFF	Integrated MSI Reception Module (iMRM) Address Register	0004820H	P	P	Kernel Reset	3500

(table continues...)

Table 878 (continued) Register overview - PCIE_DSP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
MSI_CTRL_UPPER_ADDR_OFF	Integrated MSI Reception Module Upper Address Register	0004824$_H$	P	P	Kernel Reset	3501
MSI_CTRL_INT_i_EN_OFF	Integrated MSI i Reception Module Interrupt0 Enable Register	0004828$_H$+i*12	P	P	Kernel Reset	3501
MSI_CTRL_INT_i_MASK_OFF	Integrated MSI i Reception Module Interrupt0 Mask Register	000482C$_H$+i*12	P	P	Kernel Reset	3502
MSI_CTRL_INT_i_STATUS_OFF	Integrated MSI i Reception Module Interrupt0 Status Register	0004830$_H$+i*12	P	P	Kernel Reset	3502
MSI_GPIO_IO_OFF	Integrated MSI Reception Module General Purpose IO Register	0004888$_H$	P	P	Kernel Reset	3503
CLOCK_GATING_CTRL_OFF	Clock Gating Control Register	000488C$_H$	P	P	Kernel Reset	3503
GEN3_RELATED_OFF	Gen3 Control Register	0004890$_H$	P	P	Kernel Reset	3504
GEN3_EQ_CONTROL_OFF	Gen3 EQ Control Register	00048A8$_H$	P	P	Kernel Reset	3507
GEN3_EQ_FB_MODE_DIR_CHANGE_OFF	Gen3 EQ Direction Change Feedback Mode Control Register	00048AC$_H$	P	P	Kernel Reset	3510
ORDER_RULE_CTRL_OFF	Order Rule Control Register	00048B4$_H$	P	P	Kernel Reset	3511
PIPE_LOOPBACK_CONTROL_OFF	PIPE Loopback Control Register	00048B8$_H$	P	P	Kernel Reset	3512
MISC_CONTROL_1_OFF	DBI Read-Only Write Enable Register	00048BC$_H$	P	P	Kernel Reset	3513
LINK_FLUSH_CONTROL_OFF	Link Reset Request Flush Control Register	00048CC$_H$	P	P	Kernel Reset	3519
AMBA_ERROR_RESPONSE_DEFAULT_OFF	AXI Bridge Slave Error Response Register	00048D0$_H$	P	P	Kernel Reset	3520
AMBA_LINK_TIMEOUT_OFF	Link Down AXI Bridge Slave Timeout Register	00048D4$_H$	P	P	Kernel Reset	3521
AMBA_ORDERING_CTRL_OFF	AXI Bridge Ordering Control	00048D8$_H$	P	P	Kernel Reset	3522
PCIE_VERSION_NUMBER_OFF	PCIe Controller IIP Release Version Number	00048F8$_H$	P	P	Kernel Reset	3526
PCIE_VERSION_TYPE_OFF	PCIe Controller IIP Release Version Type	00048FC$_H$	P	P	Kernel Reset	3527

(table continues...)

Table 878 (continued) Register overview - PCIE_DSP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
INTERFACE_TIMER_CONTROL_OFF	Interface Timer Control Register	0004930H	P	P	Kernel Reset	3527
INTERFACE_TIMER_TARGET_OFF	Interface Timer Target Register	0004934H	P	P	Kernel Reset	3528
INTERFACE_TIMER_STATUS_OFF	Interface Timer Status Register	0004938H	P	P	Kernel Reset	3529
MSIX_ADDRESS_MATCH_LOW_OFF	MSI-X Address Match Low Register	0004940H	P	P	Kernel Reset	3530
MSIX_ADDRESS_MATCH_HIGH_OFF	MSIX Address Match High Register	0004944H	P	P	Kernel Reset	3531
MSIX_DOORBELL_OFF	MSI-X Doorbell Register	0004948H	P	P	Kernel Reset	3531
SAFETY_MASK_OFF	Functional Safety Interrupt Events Mask Register	0004960H	P	P	Kernel Reset	3534
SAFETY_STATUS_OFF	Status for functional safety interrupt events	0004964H	P	P	Kernel Reset	3535
PL_APP_BUS_DEV_NUM_STATUS_OFF	Application driven bus and device number register	0004B10H	P	P	Kernel Reset	3536
PL_CHK_REG_CONTROL_STATUS_OFF	CDM Register Checking Control and Status Register	0004B20H	P	P	Kernel Reset	3537
PL_CHK_REG_START_END_OFF	CDM Register Checking First and Last address to check	0004B24H	P	P	Kernel Reset	3539
PL_CHK_REG_ERR_ADDR_OFF	CDM Register Checking Error Address	0004B28H	P	P	Kernel Reset	3539
PL_CHK_REG_ERR_PF_VF_OFF	CDM Register Checking error PF Numbers	0004B2CH	P	P	Kernel Reset	3540
PL_LTR_LATENCY_OFF	LTR Latency Register	0004B30H	P	P	Kernel Reset	3540
AUX_CLK_FREQ_OFF	Auxiliary Clock Frequency Control Register	0004B40H	P	P	Kernel Reset	3541
POWERDOWN_CTRL_STATUS_OFF	Powerdown Control and Status Register	0004B48H	P	P	Kernel Reset	3543
MSIXPBAx_DWORD	MSIX x table DWORD	0005E10H+x*4	P	P	Kernel Reset	3545
SHADOW_PCI_MSIX_CAP_ID_NEXT_CTRL_REG	MSI-X Capability ID, Next Pointer, Control Register	00060B0H	P	P	Kernel Reset	3545

(table continues...)

Table 878 (continued) Register overview - PCIE_DSP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
DMA_CTRL_DATA_ARB_PRIOR_OFF	DMA Arbitration Scheme for TRGT1 Interface	0007000H	P	P	Kernel Reset	3547
DMA_CTRL_OFF	DMA Number of Channels Register	0007008H	P	P	Kernel Reset	3548
DMA_WRITE_ENGINE_EN_OFF	DMA Write Engine Enable Register	000700CH	P	P	Kernel Reset	3549
DMA_WRITE_DOORBELL_OFF	DMA Write Doorbell Register	0007010H	P	P	Kernel Reset	3551
DMA_WRITE_CHANNEL_ARB_WEIGHT_LOW_OFF	DMA Write Engine Channel Arbitration Weight Low Register	0007018H	P	P	Kernel Reset	3552
DMA_READ_ENGINE_EN_OFF	DMA Read Engine Enable Register	000702CH	P	P	Kernel Reset	3554
DMA_READ_DOORBELL_OFF	DMA Read Doorbell Register	0007030H	P	P	Kernel Reset	3555
DMA_READ_CHANNEL_ARB_WEIGHT_LOW_OFF	DMA Read Engine Channel Arbitration Weight Low Register	0007038H	P	P	Kernel Reset	3556
DMA_WRITE_INT_STATUS_OFF	DMA Write Interrupt Status Register	000704CH	P	P	Kernel Reset	3558
DMA_WRITE_INT_MASK_OFF	DMA Write Interrupt Mask Register	0007054H	P	P	Kernel Reset	3559
DMA_WRITE_INT_CLEAR_OFF	DMA Write Interrupt Clear Register	0007058H	P	P	Kernel Reset	3560
DMA_WRITE_ERR_STATUS_OFF	DMA Write Error Status Register	000705CH	P	P	Kernel Reset	3561
DMA_WRITE_DONE_IMWR_LOW_OFF	DMA Write Done IMWr Address Low Register	0007060H	P	P	Kernel Reset	3561
DMA_WRITE_DONE_IMWR_HIGH_OFF	DMA Write Done IMWr Interrupt Address High Register	0007064H	P	P	Kernel Reset	3562
DMA_WRITE_ABORT_IMWR_LOW_OFF	DMA Write Abort IMWr Address Low Register	0007068H	P	P	Kernel Reset	3562
DMA_WRITE_ABORT_IMWR_HIGH_OFF	DMA Write Abort IMWr Address High Register	000706CH	P	P	Kernel Reset	3563
DMA_WRITE_CH01_IMWR_DATA_OFF	DMA Write Channel 1 and 0 IMWr Data Register	0007070H	P	P	Kernel Reset	3563

(table continues...)

Table 878 (continued) Register overview - PCIE_DSP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
DMA_WRITE_CH23_IMWR_DATA_OFF	DMA Write Channel 3 and 2 IMWr Data Register	0007074H	P	P	Kernel Reset	3564
DMA_WRITE_LINKED_LIST_ERR_EN_OFF	DMA Write Linked List Error Enable Register	0007090H	P	P	Kernel Reset	3565
DMA_READ_INT_STATUS_OFF	DMA Read Interrupt Status Register	00070A0H	P	P	Kernel Reset	3566
DMA_READ_INT_MASK_OFF	DMA Read Interrupt Mask Register	00070A8H	P	P	Kernel Reset	3567
DMA_READ_INT_CLEAR_OFF	DMA Read Interrupt Clear Register	00070ACH	P	P	Kernel Reset	3568
DMA_READ_ERR_STATUS_LOW_OFF	DMA Read Error Status Low Register	00070B4H	P	P	Kernel Reset	3569
DMA_READ_ERR_STATUS_HIGH_OFF	DMA Read Error Status High Register	00070B8H	P	P	Kernel Reset	3570
DMA_READ_LINKED_LIST_ERR_EN_OFF	DMA Read Linked List Error Enable Register	00070C4H	P	P	Kernel Reset	3571
DMA_READ_DONE_IMWR_LOW_OFF	DMA Read Done IMWr Address Low Register	00070CCH	P	P	Kernel Reset	3572
DMA_READ_DONE_IMWR_HIGH_OFF	DMA Read Done IMWr Address High Register	00070D0H	P	P	Kernel Reset	3572
DMA_READ_ABORT_IMWR_LOW_OFF	DMA Read Abort IMWr Address Low Register	00070D4H	P	P	Kernel Reset	3573
DMA_READ_ABORT_IMWR_HIGH_OFF	DMA Read Abort IMWr Address High Register	00070D8H	P	P	Kernel Reset	3573
DMA_READ_CH01_IMWR_DATA_OFF	DMA Read Channel 1 and 0 IMWr Data Register	00070DCH	P	P	Kernel Reset	3574
DMA_READ_CH23_IMWR_DATA_OFF	DMA Read Channel 3 and 2 IMWr Data Register	00070E0H	P	P	Kernel Reset	3574
DMA_WRITE_ENGINE_HSHAKE_CNT_LOW_OFF	DMA Write Engine Handshake Counter Channel 0/1/2/3 Register	0007108H	P	P	Kernel Reset	3576

(table continues...)

Table 878 (continued) Register overview - PCIE_DSP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
DMA_READ_ENGINE_HSHAKE_CNT_LOW_OFF	DMA Read Engine Handshake Counter Channel 0/1/2/3 Register	0007118$_H$	P	P	Kernel Reset	3577
DMA_WRITE_CHi_PWR_EN_OFF	DMA Write Channel i Power Enable Register	0007128$_H$+i*4	P	P	Kernel Reset	3579
DMA_READ_CHi_PWR_EN_OFF	DMA Read Channel i Power Enable Register	0007168$_H$+i*4	P	P	Kernel Reset	3579
DMA_CH_CONTROL1_OFF_WRCH_i	DMA Write Channel i Control 1 Register	0007200$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3580
DMA_TRANSFER_SIZE_OFF_WRCH_i	DMA CH i Write Transfer Size Register	0007208$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3583
DMA_SAR_LOW_OFF_WRCH_i	DMA CH i Write SAR Low Register	000720C$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3583
DMA_SAR_HIGH_OFF_WRCH_i	DMA CH i Write SAR High Register	0007210$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3584
DMA_DAR_LOW_OFF_WRCH_i	DMA CH i Write DAR Low Register	0007214$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3584
DMA_DAR_HIGH_OFF_WRCH_i	DMA CH i Write DAR High Register	0007218$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3585
DMA_LLP_LOW_OFF_WRCH_i	DMA CH i Write Linked List Pointer Low Register	000721C$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3585
DMA_LLP_HIGH_OFF_WRCH_i	DMA CH i Write Linked List Pointer High Register	0007220$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3586
DMACAP_CHi_WRCHN_WR_INT_CLEAR	Interrupt clear for DMA write channel i	0007224$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3586
DMACAP_CHi_WRCHN_WR_INT_STATUS	Interrupt status for DMA write channel i	0007228$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3587
DMACAP_CHi_WRCHN_WR_ERR_STATUS	Error status for DMA write channel i	000722C$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3588
DMACAP_CHi_WRCHN_WR_DOORBELL	Doorbell for DMA write channel i	0007230$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3589
DMA_CH_CONTROL1_OFF_RDCH_i	DMA Read Channel i Control 1 Register	0007300$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3589
DMA_TRANSFER_SIZE_OFF_RDCH_i	DMA Read CH i Transfer Size Register	0007308$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3592

(table continues...)

Table 878 (continued) Register overview - PCIE_DSP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
DMA_SAR_LOW_OFF_RDCH_i	DMA Read CH i SAR Low Register	000730C$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3593
DMA_SAR_HIGH_OFF_RDCH_i	DMA Read CH i SAR High Register	0007310$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3594
DMA_DAR_LOW_OFF_RDCH_i	DMA Read CH i DAR Low Register	0007314$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3594
DMA_DAR_HIGH_OFF_RDCH_i	DMA Read CH i DAR High Register	0007318$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3595
DMA_LLP_LOW_OFF_RDCH_i	DMA Read CH i Linked List Pointer Low Register	000731C$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3595
DMA_LLP_HIGH_OFF_RDCH_i	DMA CH i Read Linked List Pointer High Register	0007320$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3596
DMACAP_CHi_RDCHN_RD_INT_CLEAR	Interrupt clear for DMA read channel i	0007324$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3596
DMACAP_CHi_RDCHN_RD_INT_STATUS	Interrupt status for DMA read channel i	0007328$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3597
DMACAP_CHi_RDCHN_RD_ERR_STATUS	Error status for DMA read channel i	000732C$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3597
DMACAP_CHi_RDCHN_RD_ERR_STATUS_HIGH	Higher error status for DMA read channel i	0007330$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3598
DMACAP_CHi_RDCHN_RD_DOORBELL	Doorbell for DMA read channel i	0007334$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3599

19.4.2.3 Register overview - PCIE_DSP_SRI domain SRI (ascending offset address)

Table 879 Register overview - PCIE_DSP_SRI domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
S2A_ACCENy_WRA	Write access enable register A	3FFFE00$_H$+y*20$_H$	U	SE, SV	Application Reset	3600
S2A_ACCENy_WRB	Write access enable register B	3FFFE04$_H$+y*20$_H$	U	SE, SV	Application Reset	3601
S2A_ACCENy_RDA	Read access enable register A	3FFFE08$_H$+y*20$_H$	U	SE, SV	Application Reset	3601
S2A_ACCENy_RDB	Read access enable register B	3FFFE0C$_H$+y*20$_H$	U	SE, SV	Application Reset	3602

(table continues...)

Table 879 (continued) Register overview - PCIE_DSP_SRI domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
S2A_ACCENy_VM	VM access enable register	3FFFE10$_H$+y*20$_H$	U	SE, SV	Application Reset	3602
S2A_ACCENy_PRS	PRS access enable register	3FFFE14$_H$+y*20$_H$	U	SE, SV	Application Reset	3603
S2A_ACCENy_RGNLA	Region lower address register	3FFFE18$_H$+y*20$_H$	U	SE, SV	Application Reset	3603
S2A_ACCENy_RGNUA	Region upper address register	3FFFE1C$_H$+y*20$_H$	U	SE, SV	Application Reset	3604
S2A_PROTSE	PROT Register Safe Endinit	3FFFF04$_H$	U	SV	Application Reset	3604

19.4.2.4 Clock control register

The Clock Control Register CLC allows the programmer to adapt the functionality and power consumption of the module to the requirements of the application.

Register CLC controls the module clock signal and the reactivity to the sleep signal.

CLC Offset address: 0000000$_H$

Clock control register Application Reset value: 0000 0003$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												EDIS	0	DISS	DISR
r												rw	r	rh	rw

Field	Bits	Type	Description
DISR	0	rw	**Module Disable Request Bit** Used for enable/disable control of the module. 0_B On request: enable the module clock 1_B Off request: stop the module clock
DISS	1	rh	**Module Disable Status Bit** When CLC.DISS=1_B, registers CLC, OCS, ID, PROTE, PROTSE, ACCEN_* and DMA_ACCEN_* are still accessible. Accesses to other PCIe registers are not executed and will result in a bus error. 0_B Module clock is enabled 1_B Off: module is not clocked

(table continues...)

(continued)

Field	Bits	Type	Description
EDIS	3	rw	**Reserved** Used to control the module's reaction to sleep mode. Note: this bit field has no functionality implemented for PCIe. 0_B Sleep mode request is enabled and functional 1_B Module disregards the sleep mode control signal
0	2, 31:4	r	**Reserved** Read as 0; should be written with 0.

19.4.2.5 OCDS control and status register

OCS Offset address: 0000004_H

OCDS control and status register Debug Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															
r															

Field	Bits	Type	Description
0	31:0	r	**Reserved** Read as 0; should be written with 0.

19.4.2.6 Module identification register

ID Offset address: 0000008_H

Module identification register PowerOn Reset value: $0106\ C002_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MOD_NUM															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MOD_TYPE								MOD_REV							
r								r							

Field	Bits	Type	Description
MOD_REV	7:0	r	**Module Revision** Indicates the revision number of the implementation.
MOD_TYPE	15:8	r	**Module Type** This internal marker is fixed to $C0_H$.

(table continues...)

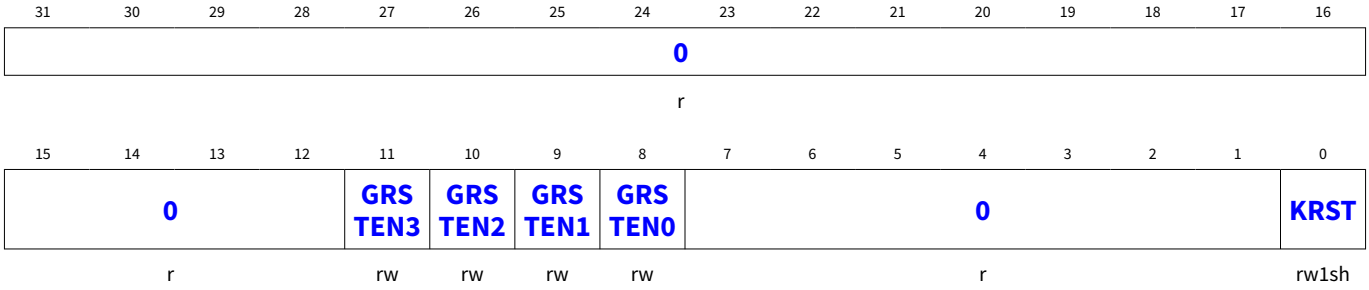
(continued)

Field	Bits	Type	Description
MOD_NUM	31:16	r	**Module Number** Indicates the module identification number. 0x0106 for PCIe.

19.4.2.7 Reset control register A

RST_CTRLA Offset address: $000000C_H$

Reset control register A Application Reset value: $0000\ 0000_H$

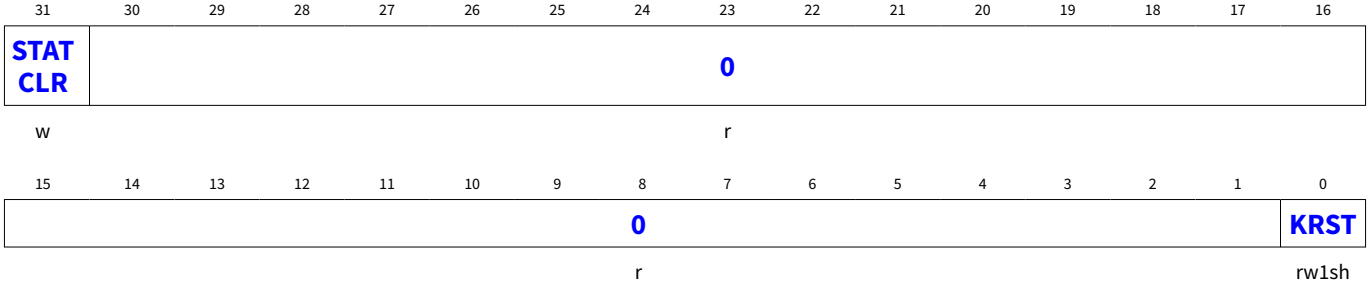


Field	Bits	Type	Description
KRST	0	rw1sh	**Kernel Reset** Request a kernel reset. The requested reset is executed if CTRLA.KRST=1 and CTRLB.KRST=1. KRST is cleared after the kernel reset was executed. 0_B No action 1_B A kernel reset was requested
GRSTENx (x=0-3)	x+8	rw	**Enable for Global Module Reset Group x** 0_B Global module reset group x does not have any effect 1_B Global module reset group x results in a kernel reset
0	7:1, 31:12	r	**Reserved** Read as 0; should be written with 0.

19.4.2.8 Reset control register B

RST_CTRLB Offset address: 0000010_H

Reset control register B Application Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
KRST	0	rw1sh	**Kernel Reset** Request a kernel reset. The requested reset is executed if CTRLA.KRST=1 and CTRLB.KRST=1. KRST is cleared after the kernel reset was executed. 0_B No action 1_B A kernel reset was requested
STATCLR	31	w	**Kernel Reset Status Clear** Clears all status bits in RST_STAT when 1 is written. Read will return 0. 0_B No action 1_B Write with ´1´ clears bits STAT.GRSTx and bit STAT.KRST .
0	30:1	r	**Reserved** Read as 0; should be written with 0.

19.4.2.9 Reset status register

The reset status register contains the status bits for kernel reset (KRST) and the global module reset groups (GRSTx).

RST_STAT Offset address: 0000014_H

Reset status register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				GRST3	GRST2	GRST1	GRST0	0							KRST
r				rh	rh	rh	rh	r							rh

Field	Bits	Type	Description
KRST	0	rh	**Kernel Reset Status** Indicates an executed kernel reset. RST_STAT.KRST is set after the execution of a kernel reset in the same clock cycle in which the reset bits are cleared. Clear KRST by setting bit STATCLR in register RST_CTRLB.
GRSTx (x=0-3)	x+8	rh	**Status for Global Module Reset Group x** 0_B Reset was not triggered by Global Reset Group x 1_B Reset was triggered by Global Reset Group x
0	7:1, 31:12	r	**Reserved** Read as 0.

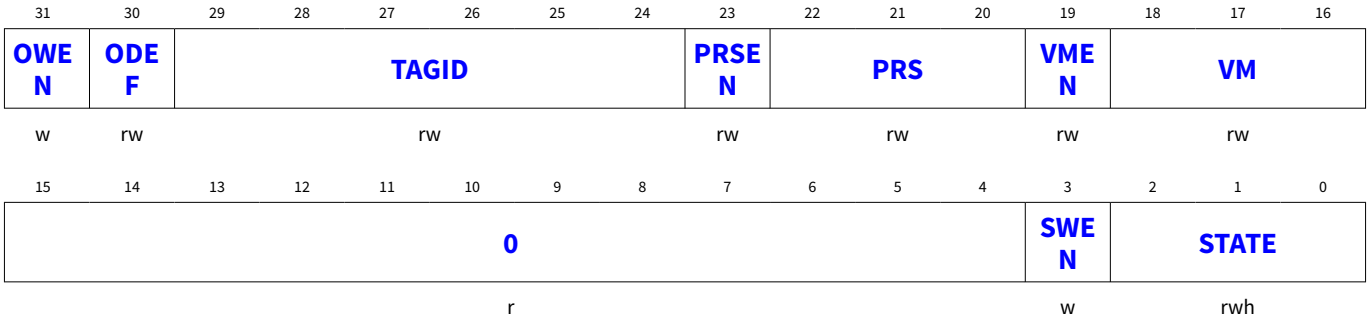
19.4.2.10 PROT register endinit

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner and in some PROT states, the secure master to update the PROT state as described in the PROT mechanism.

The PROTE register controls lock / unlock of the local Endinit (E) protected control registers.

PROTE Offset address: 0000020_H

PROT register endinit Application Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
VM	18:16	rw	**Virtual machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual machine definition enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.

(table continues...)

(continued)

Field	Bits	Type	Description
PRSEN	23	rw	**Protection set definition enable for PROT owner** 0_B **Disable**: PRS is not part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG-ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; should be written with 0.

Table 880 Access types of PROTE in order of evaluation

Condition	Type		Description
write 1 to .**SWEN**	rwh	STATE	
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
otherwise	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

19.4.2.11 PROT register safe endinit

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner and in some PROT states, the secure master to update the PROT state as described in the PROT mechanism.

The PROTSE register controls lock / unlock of the local Safe Endinit (SE) protected control registers.

PROTSE Offset address: 0000024_H

PROT register safe endinit Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OWEN	ODEF	TAGID						PRSEN	PRS			VMEN	VM		
w	rw	rw						rw	rw			rw	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												SWEN	STATE		
r												w	rwh		

Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
VM	18:16	rw	**Virtual machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual machine definition enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.
PRSEN	23	rw	**Protection set definition enable for PROT owner** 0_B **Disable**: PRS is not part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG-ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; should be written with 0.

Table 881 Access types of PROTSE in order of evaluation

Condition	Type		Description
write 1 to .**SWEN**	rwh	STATE	
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
otherwise	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

19.4.2.12 Write access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

ACCEN_WRA Offset address: 0000040_H

Write access enable register A Application Reset value: $1000\ 0003_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	**EN30**	**EN29**	**EN28**	**EN27**	**EN26**	**EN25**	**EN24**	**EN23**	**EN22**	**EN21**	**EN20**	**EN19**	**EN18**	**EN17**	**EN16**
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	**EN14**	**EN13**	**EN12**	**EN11**	**EN10**	**EN09**	**EN08**	**EN07**	**EN06**	**EN05**	**EN04**	**EN03**	**EN02**	**EN01**	**EN00**
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access

19.4.2.13 Write access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

ACCEN_WRB Offset address: 0000044_H

Write access enable register B Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								**EN39**	**EN38**	**EN37**	**EN36**	**EN35**	**EN34**	**EN33**	**EN32**
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access
0	31:8	r	**Reserved** Read as 0; should be written with 0.

19.4.2.14 Read access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

ACCEN_RDA Offset address: 0000048_H

Read access enable register A Application Reset value: $FFFF\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access

19.4.2.15 Read access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

ACCEN_RDB Offset address: $000004C_H$

Read access enable register B Application Reset value: $0000\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access
0	31:8	r	**Reserved** Read as 0; should be written with 0.

19.4.2.16 VM access enable register

This register defines which virtual machine encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each VMx.

ACCEN_VM Offset address: 0000050_H

VM access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for VM ID q** This bit enables read access to the access protected region for transactions with the VM ID q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for VM ID q** This bit enables write access to the access protected region for transactions with the VM ID q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

19.4.2.17 PRS access enable register

This register defines which protection register set encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each PRSx.

ACCEN_PRS Offset address: 0000054_H

PRS access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for PRS q** This bit enables read access to the access protected region for transactions with the PRS q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for PRS q** This bit enables write access to the access protected region for transactions with the PRS q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

19.4.2.18 CHy Write access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

DMA_ACCENy_WRA (y=0-3) Offset address: 0000060_H+y*20_H

CHy Write access enable register A Application Reset value: $1000\ 0003_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access

19.4.2.19 CHy Write access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

DMA_ACCENy_WRB (y=0-3) Offset address: 0000064_H+y*20_H

CHy Write access enable register B Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access
0	31:8	r	**Reserved** Read as 0; should be written with 0.

19.4.2.20 CHy Read access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

DMA_ACCENy_RDA (y=0-3) Offset address: 0000068_H+y*20_H

CHy Read access enable register A Application Reset value: $FFFF\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access

19.4.2.21 CHy Read access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

DMA_ACCENy_RDB (y=0-3) Offset address: $000006C_H+y*20_H$

CHy Read access enable register B Application Reset value: $0000\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access
0	31:8	r	**Reserved** Read as 0; should be written with 0.

19.4.2.22 CHy VM access enable register

This register defines which virtual machine encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each VMx.

DMA_ACCENy_VM (y=0-3) Offset address: 0000070_H+y*20_H

CHy VM access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for VM ID q** This bit enables read access to the access protected region for transactions with the VM ID q. 0_B Disabled for read access 1_B Enabled for read access

(table continues...)

(continued)

Field	Bits	Type	Description
WRq (q=00-07)	q+16	rw	**Write access enable for VM ID q** This bit enables write access to the access protected region for transactions with the VM ID q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

19.4.2.23 CHy PRS access enable register

This register defines which protection register set encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each PRSx.

DMA_ACCENy_PRS (y=0-3) Offset address: 0000074_H+y*20_H

CHy PRS access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for PRS q** This bit enables read access to the access protected region for transactions with the PRS q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for PRS q** This bit enables write access to the access protected region for transactions with the PRS q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

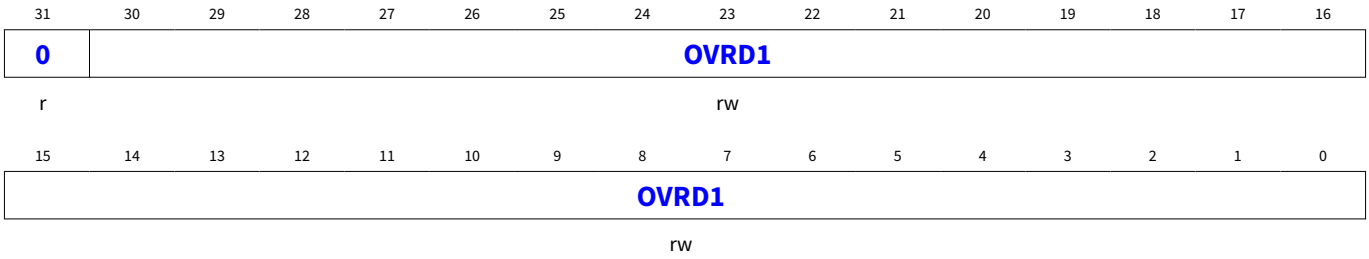
19.4.2.24 UPCS layer override register 1

UPCS Override Settings

Note: the settings can be overwritten from these top-level when OVRD17.CTRLSEL is asserted.

UPCS_OVRD1 Offset address: 0000300_H

UPCS layer override register 1 Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
OVRD1	30:0	rw	**UPCS override 1**
0	31	r	**Reserved** Read as 0; should be written with 0.

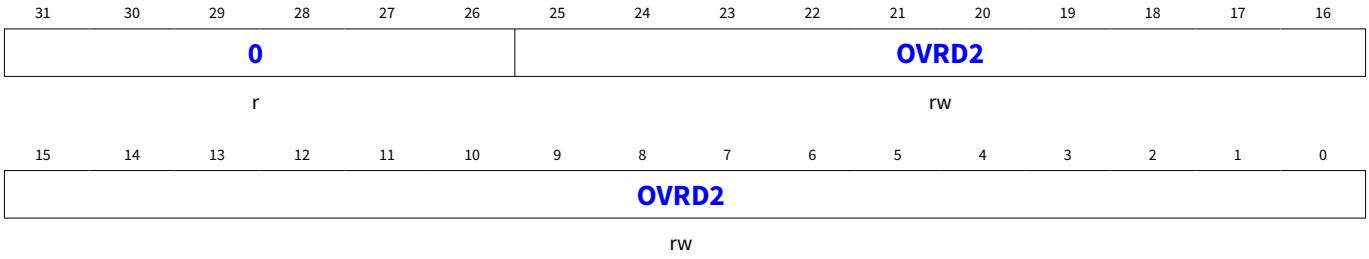
19.4.2.25 UPCS layer override register 2

UPCS Override Settings

Note: the settings can be overwritten from these top-level when OVRD17.CTRLSEL is asserted.

UPCS_OVRD2 Offset address: 0000304_H

UPCS layer override register 2 Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
OVRD2	25:0	rw	**UPCS override**
0	31:26	r	**Reserved** Read as 0; should be written with 0.

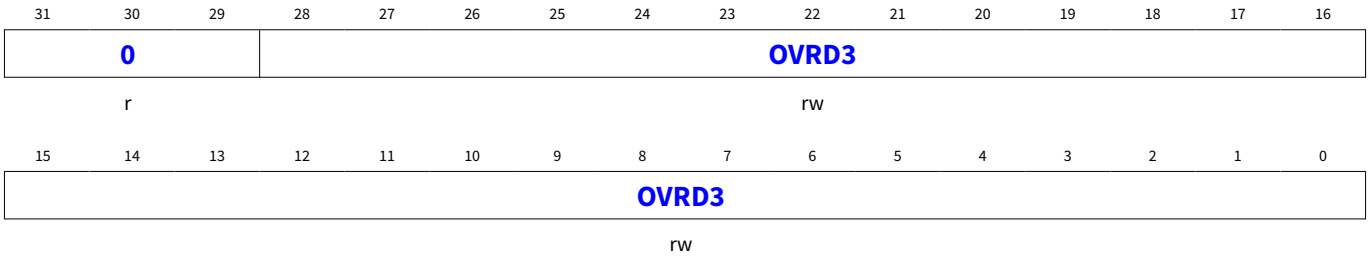
19.4.2.26 UPCS layer override register 3

UPCS Override Settings

Note: the settings can be overwritten from these top-level when OVRD17.CTRLSEL is asserted.

UPCS_OVRD3 Offset address: 0000308_H

UPCS layer override register 3 Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
OVRD3	28:0	rw	**UPCS override 3**
0	31:29	r	**Reserved** Read as 0; should be written with 0.

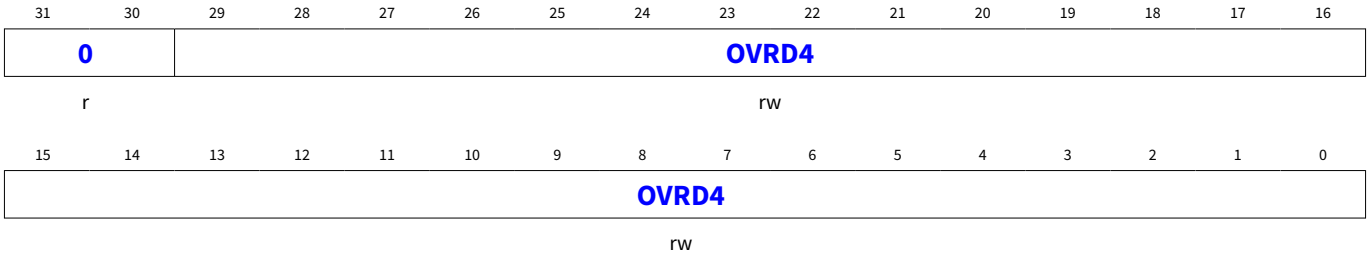
19.4.2.27 UPCS layer override register 4

UPCS Override Settings

Note: the settings can be overwritten from these top-level when OVRD17.CTRLSEL is asserted.

UPCS_OVRD4 Offset address: $000030C_H$

UPCS layer override register 4 Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
OVRD4	29:0	rw	**UPCS override 4**
0	31:30	r	**Reserved** Read as 0; should be written with 0.

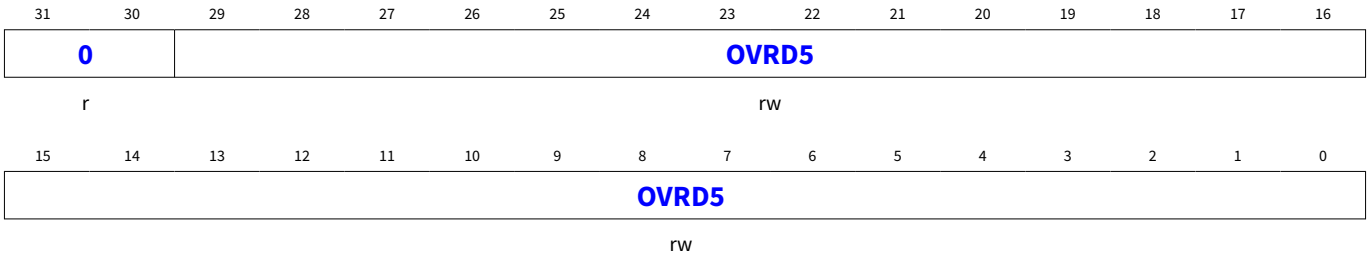
19.4.2.28 UPCS layer override register 5

UPCS Override Settings

Note: the settings can be overwritten from these top-level when OVRD17.CTRLSEL is asserted.

UPCS_OVRD5 Offset address: 0000310_H

UPCS layer override register 5 Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
OVRD5	29:0	rw	**UPCS override 5**
0	31:30	r	**Reserved** Read as 0; should be written with 0.

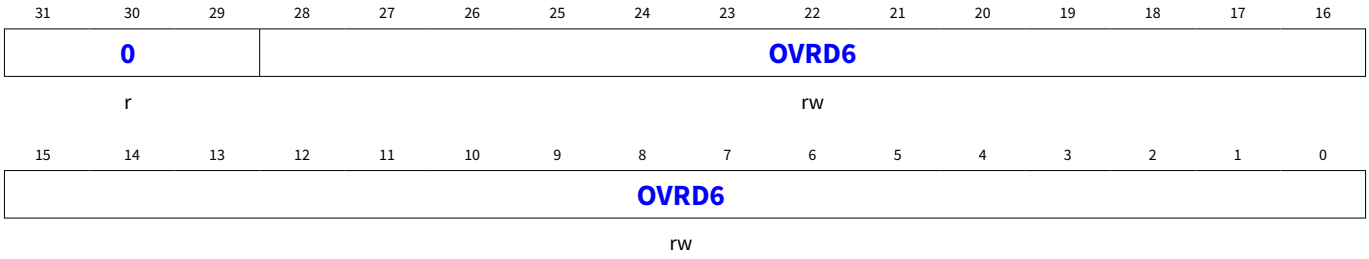
19.4.2.29 UPCS layer override register 6

UPCS Override Settings

Note: the settings can be overwritten from these top-level when OVRD17.CTRLSEL is asserted.

UPCS_OVRD6 Offset address: 0000314_H

UPCS layer override register 6 Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
OVRD6	28:0	rw	**UPCS override 6**
0	31:29	r	**Reserved** Read as 0; should be written with 0.

19.4.2.30 UPCS layer override register 7

UPCS Override Settings

Note: the settings can be overwritten from these top-level when OVRD17.CTRLSEL is asserted.

UPCS_OVRD7 Offset address: 0000318_H

UPCS layer override register 7 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OVRD7															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OVRD7															
rw															

Field	Bits	Type	Description
OVRD7	31:0	rw	**UPCS override 7**

19.4.2.31 UPCS layer override register 8

UPCS Override Settings

Note: the settings can be overwritten from these top-level when OVRD17.CTRLSEL is asserted.

UPCS_OVRD8 Offset address: $000031C_H$

UPCS layer override register 8 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OVRD8															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OVRD8															
rw															

Field	Bits	Type	Description
OVRD8	31:0	rw	**UPCS override 8**

19.4.2.32 UPCS layer override register 9

UPCS Override Settings

Note: the settings can be overwritten from these top-level when OVRD17.CTRLSEL is asserted.

UPCS_OVRD9 Offset address: 0000320_H

UPCS layer override register 9 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OVRD9															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OVRD9															
rw															

Field	Bits	Type	Description
OVRD9	31:0	rw	**UPCS override 9**

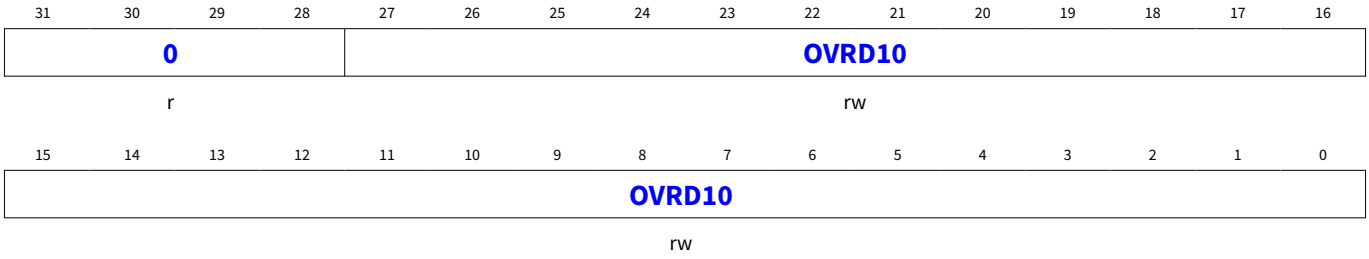
19.4.2.33 UPCS layer override register 10

UPCS Override Settings

Note: the settings can be overwritten from these top-level when OVRD17.CTRLSEL is asserted.

UPCS_OVRD10 Offset address: 0000324_H

UPCS layer override register 10 Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
OVRD10	27:0	rw	**UPCS override 10**
0	31:28	r	**Reserved** Read as 0; should be written with 0.

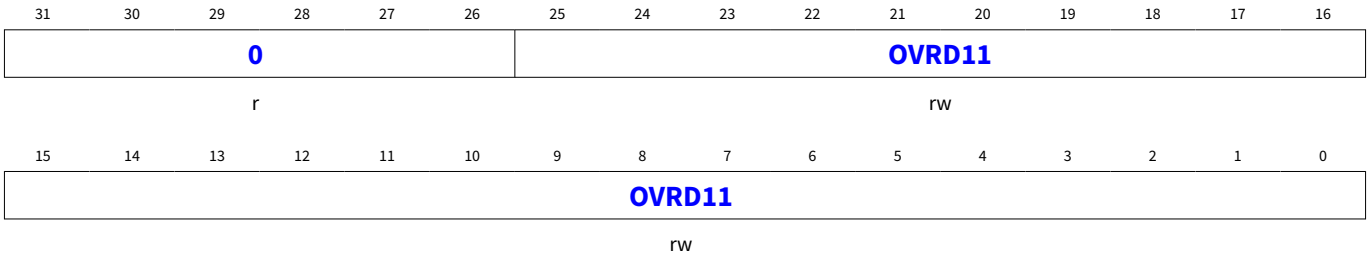
19.4.2.34 UPCS layer override register 11

UPCS Override Settings

Note: the settings can be overwritten from these top-level when OVRD17.CTRLSEL is asserted.

UPCS_OVRD11 Offset address: 0000328_H

UPCS layer override register 11 Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
OVRD11	25:0	rw	**UPCS override 11**
0	31:26	r	**Reserved** Read as 0; should be written with 0.

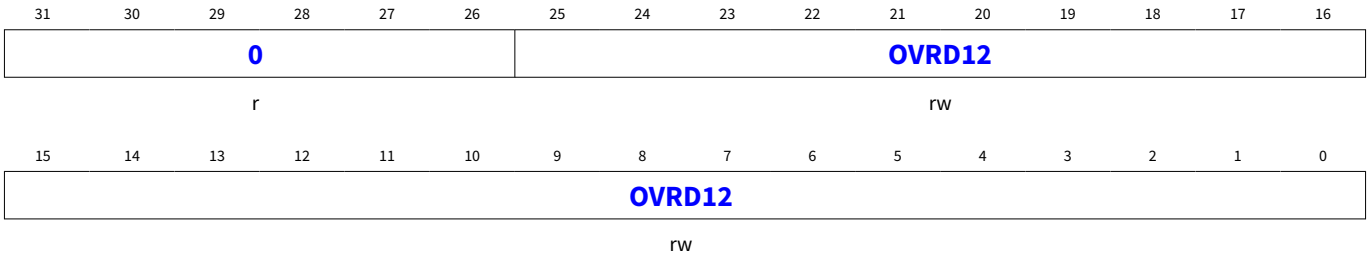
19.4.2.35 UPCS layer override register 12

UPCS Override Settings

Note: the settings can be overwritten from these top-level when OVRD17.CTRLSEL is asserted.

UPCS_OVRD12 Offset address: $000032C_H$

UPCS layer override register 12 Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
OVRD12	25:0	rw	**UPCS override 12**
0	31:26	r	**Reserved** Read as 0; should be written with 0.

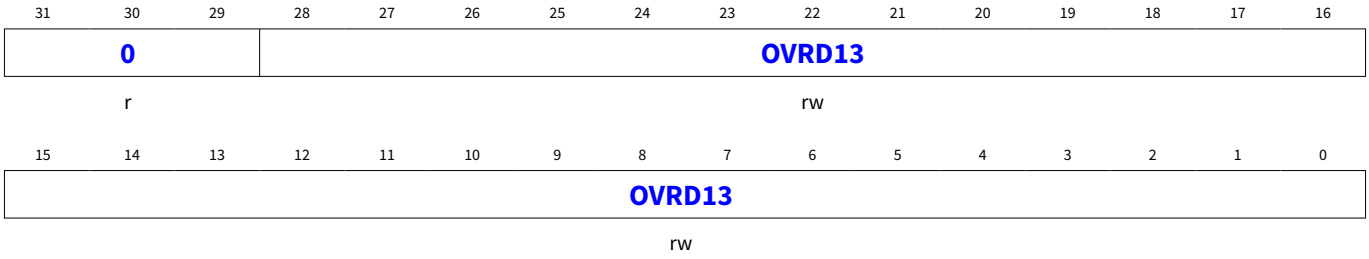
19.4.2.36 UPCS layer override register 13

UPCS Override Settings

Note: the settings can be overwritten from these top-level when OVRD17.CTRLSEL is asserted.

UPCS_OVRD13 Offset address: 0000330_H

UPCS layer override register 13 Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
OVRD13	28:0	rw	**UPCS override 13**
0	31:29	r	**Reserved** Read as 0; should be written with 0.

19.4.2.37 UPCS layer override register 14

UPCS Override Settings

Note: the settings can be overwritten from these top-level when OVRD17.CTRLSEL is asserted.

UPCS_OVRD14 Offset address: 0000334_H

UPCS layer override register 14 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OVRD14															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OVRD14															
rw															

Field	Bits	Type	Description
OVRD14	31:0	rw	**UPCS override 14**

19.4.2.38 UPCS layer override register 15

UPCS Override Settings

Note: the settings can be overwritten from these top-level when OVRD17.CTRLSEL is asserted.

UPCS_OVRD15 Offset address: 0000338_H

UPCS layer override register 15 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0					OVRD15										
r					rw										

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OVRD15															
rw															

Field	Bits	Type	Description
OVRD15	26:0	rw	**UPCS override 15**
0	31:27	r	**Reserved** Read as 0; should be written with 0.

19.4.2.39 UPCS layer override register 16

UPCS Override Settings

Note: the settings can be overwritten from these top-level when OVRD17.CTRLSEL is asserted.

UPCS_OVRD16 Offset address: $000033C_H$

UPCS layer override register 16 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OVRD16															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OVRD16															
rw															

Field	Bits	Type	Description
OVRD16	31:0	rw	**UPCS override 16**

19.4.2.40 UPCS layer override register 17

UPCS Override Settings

Note: the settings can be overwritten from these top-level when OVRD17.CTRLSEL is asserted.

UPCS_OVRD17 Offset address: 0000340_H

UPCS layer override register 17 Kernel Reset value: $0004\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0													OVRD17		
r													rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OVRD17															
rw															

Field	Bits	Type	Description
OVRD17	18:0	rw	**UPCS override 17**
0	31:19	r	**Reserved** Read as 0; should be written with 0.

19.4.2.41 Request MSI interrupt, EP only.

INT_MSICTRL Offset address: 0000348_H

Request MSI interrupt, EP only. Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
INT31	INT30	INT29	INT28	INT27	INT26	INT25	INT24	INT23	INT22	INT21	INT20	INT19	INT18	INT17	INT16
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
INT15	INT14	INT13	INT12	INT11	INT10	INT9	INT8	INT7	INT6	INT5	INT4	INT3	INT2	INT1	INT0
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

Field	Bits	Type	Description
INTz (z=0-31)	z	w	**MSI interrupt request INT+ $z** Write 1 to generate an MSI interrupt $z MWr TLP to RC. This bit is write only. An interrupt request may be lost if the second request comes before the first request is processed by hardware. The status of request processing is indicated by MSISTAT.PD. 0_B Write access will not be executed 1_B Request an interrupt

19.4.2.42 Pending status of MSI interrupt request, EP only.

INT_MSISTAT Offset address: 000034C$_H$

Pending status of MSI interrupt request, EP only. Kernel Reset value: 0000 0000$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PD31	PD30	PD29	PD28	PD27	PD26	PD25	PD24	PD23	PD22	PD21	PD20	PD19	PD18	PD17	PD16
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PD15	PD14	PD13	PD12	PD11	PD10	PD9	PD8	PD7	PD6	PD5	PD4	PD3	PD2	PD1	PD0
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
PDz (z=0-31)	z	rh	**Pending status of interrupt request** Indicates if an MSI interrupt request is pending. 1 means pending and 0 means not pending. 0_B No interrupt request pending 1_B Interrupt pending

19.4.2.43 Trigger INTX interrupt, EP only.

Legay PCI interrupt INTx support, is supposed to be connected to sys_int

INT_INTXCTRL Offset address: 0000350$_H$

Trigger INTX interrupt, EP only. Kernel Reset value: 0000 0000$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0														PB	INTX
r														rh	rw

Field	Bits	Type	Description
INTX	0	rw	**INTX interrupt** When this bit goes from low to high, the controller generates an Assert_INTx Message. When it goes from high to low, the controller generates a Deassert_INTx Message. The Interrupt Pin register for the corresponding function determines which INTx Message the controller generates (INTA, INTB, INTC, or INTD). Legacy and native PCIe devices capable of generating an interrupt must support both Assert_INTx/ Deassert_INTx and MSI or MSI-X.

(table continues...)

(continued)

Field	Bits	Type	Description
PB	1	rh	**Pending status of INTx interrupt request** Indicates if an INTx interrupt request is pending and the message is not yet sent. 1 means pending and 0 means not pending. 0_B No interrupt request pending 1_B Interrupt pending
0	31:2	r	**Reserved** Read as 0; should be written with 0.

19.4.2.44 Trigger status

After the decoding of the 5 bits, 32 interrupt requests are passed to the Bit control block that consist of three different registers namely EN, STAT and CLR. The Bit control block decides which interrupt is masked in the 32, indicates the current interrupt status and clears the interrupts after service. The above three registers are designed for these functionalities respectively. The 32 interrupt requests will then merged into 8 interrupts that will be handed over to IR and software interrupts are then triggered.

INT_TRIGSTAT Offset address: 0000358_H

Trigger status Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RQ31	RQ30	RQ29	RQ28	RQ27	RQ26	RQ25	RQ24	RQ23	RQ22	RQ21	RQ20	RQ19	RQ18	RQ17	RQ16
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RQ15	RQ14	RQ13	RQ12	RQ11	RQ10	RQ9	RQ8	RQ7	RQ6	RQ5	RQ4	RQ3	RQ2	RQ1	RQ0
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
RQy (y=0-31)	y	rh	**MSI request received** Indicate that interrupt request $y is received. This bit is read only. 0_B No interrupt received 1_B Interrupt received

19.4.2.45 Interrupt clear, EP only.

INT_TRIGCLR Offset address: $000035C_H$

Interrupt clear, EP only. Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RQCLR31	RQCLR30	RQCLR29	RQCLR28	RQCLR27	RQCLR26	RQCLR25	RQCLR24	RQCLR23	RQCLR22	RQCLR21	RQCLR20	RQCLR19	RQCLR18	RQCLR17	RQCLR16
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RQCLR15	RQCLR14	RQCLR13	RQCLR12	RQCLR11	RQCLR10	RQCLR9	RQCLR8	RQCLR7	RQCLR6	RQCLR5	RQCLR4	RQCLR3	RQCLR2	RQCLR1	RQCLR0
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

Field	Bits	Type	Description
RQCLRy (y=0-31)	y	w	**Clears the corresponding incoming interrupt request.** Clear the interrupt request status when the interrupt is serviced. This bit is write only. 0_B Keep the status 1_B Clear the interrupt status

19.4.2.46 Interrupt enable EP only.

INT_TRIGEN Offset address: 0000360_H

Interrupt enable EP only. Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RQEN31	RQEN30	RQEN29	RQEN28	RQEN27	RQEN26	RQEN25	RQEN24	RQEN23	RQEN22	RQEN21	RQEN20	RQEN19	RQEN18	RQEN17	RQEN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

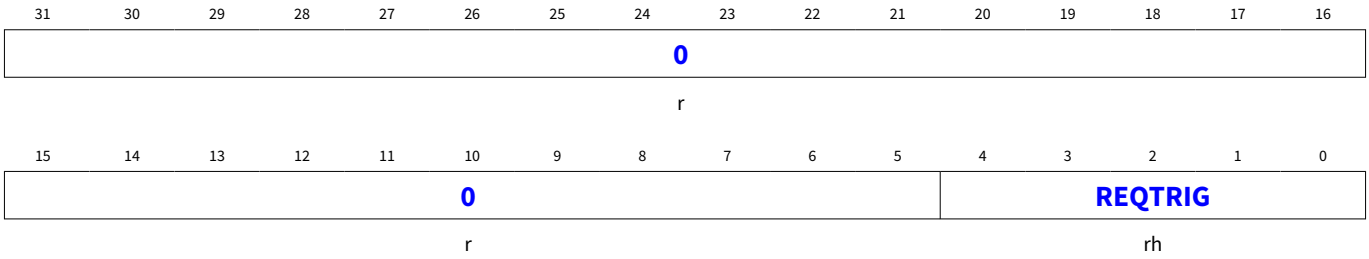
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RQEN15	RQEN14	RQEN13	RQEN12	RQEN11	RQEN10	RQEN9	RQEN8	RQEN7	RQEN6	RQEN5	RQEN4	RQEN3	RQEN2	RQEN1	RQEN0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RQENy (y=0-31)	y	rw	**Enable bit to block the incoming interrupt to the local interrupt router.** 0_B Disable interrupt 1_B Enable interrupt

19.4.2.47 Receive incoming interrupt request, EP only.

INT_REQTRIG Offset address: 0000364_H

Receive incoming interrupt request, EP only. Kernel Reset value: $0000\ 0000_H$

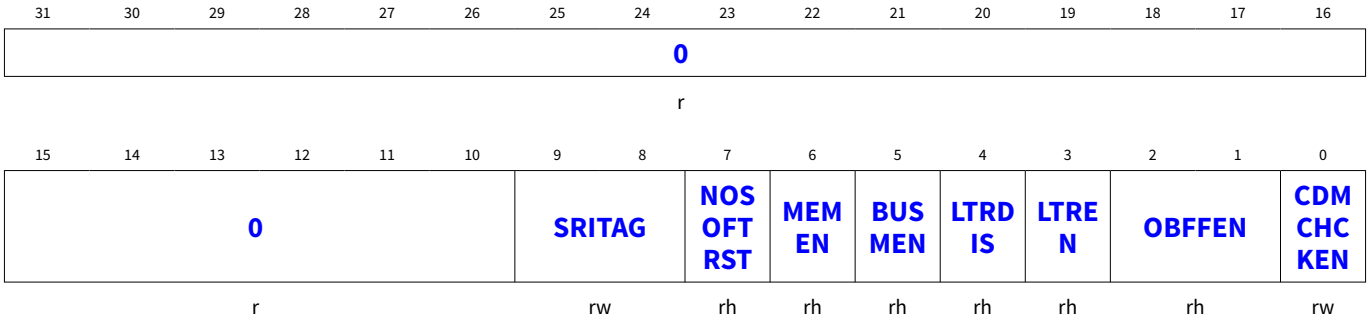


Field	Bits	Type	Description
REQTRIG	4:0	rh	**Receive incoming interrupt request** This field can only be written from wire. The received value is parsed as 32 interrupt requests.
0	31:5	r	**Reserved** Read as 0; should be written with 0.

19.4.2.48 Configuration information

CFG_INFO Offset address: 0000368_H

Configuration information Kernel Reset value: $0000\ 0080_H$



Field	Bits	Type	Description
CDMCHCKEN	0	rw	**Signal to enter test mode in the Register Checking Logic.** Asserted to enter test mode in the Register Checking Logic and disable writes to shadow CDM registers. Allows a value to be written to CDM registers to compare against the previous value in shadow CDM registers. If the value written to CDM is different to the value held in shadow CDM registers, comparison error is generated when the address is checked, indicating the Register Content Check function is working correctly. Note: In normal operation mode this bit should be driven to 0.
OBFFEN	2:1	rh	**The OBFF Enable field of the Device Control 2 register of function 0** Indicates The OBFF Enable field of the Device Control 2 register of function 0.

(table continues...)

(continued)

Field	Bits	Type	Description
LTREN	3	rh	**The LTR Mechanism Enable field of the Device Control 2 register of function 0.** Indicates the LTR Mechanism Enable field of the Device Control 2 register of function 0.
LTRDIS	4	rh	**Disable the autonomous generation of LTR clear message.**
BUSMEN	5	rh	**The state of the bus master enable bit in the PCI-compatible Command register.** Indicates the state of the bus master enable bit in the PCI-compatible Command register.
MEMEN	6	rh	**The state of the Memory Space Enable bit in the PCI-compatible** Indicates the state of the Memory Space Enable bit in the PCI-compatible Command register.
NOSOFTRST	7	rh	**No Soft Reset bit** This is the value of the No Soft Reset bit in the Power Management Control and Status Register. When set, you should not reset any controller registers when transitioning from D3hot to D0.
SRITAG	9:8	rw	**Selection of the VM or the PRS on the SRI** This bit selects which SRI tag (the VM or the PRS) is used to indicate the DMA index number, when an SRI bus transaction is initiated by the DMA. 00_B No VM/PRS tag used 01_B The VM tag is used in the SRI transaction 10_B The PRS tag is used in the SRI transaction 11_B Reserved, no VM/PRS tag used
0	31:10	r	**Reserved** Read as 0; should be written with 0.

19.4.2.49 Bus/device number

CFG_BUSNUM Offset address: $000036C_H$

Bus/device number Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0			SUBBUSNUM								SECBUSNUM				
r			rh								rh				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SECBUSNUM			DEVNUM					PBUSNUM							
rh			rh					rh							

Field	Bits	Type	Description
PBUSNUM	7:0	rh	**The primary bus number assigned to the function.**
DEVNUM	12:8	rh	**The device number assigned to the function.**

(table continues...)

(continued)

Field	Bits	Type	Description
SECBUSNUM	20:13	rh	**Configured Secondary Bus Number.** The secondary bus number assigned to the device. When the host software detects the upstream switch port, it assigns a secondary bus number. This bus number is then used in configuration requests to detect the downstream ports. The switch controller must see this so that it can route configuration requests.
SUBBUSNUM	28:21	rh	**Configured Subordinate Bus Number.** When the switch controller receives a configuration request from the upstream port, it must see the secondary and subordinate bus number fields from the downstream ports to route the request.
0	31:29	r	**Reserved** Read as 0; should be written with 0.

19.4.2.50 Select device/port type

LNK_DVCTYP Offset address: 0000370_H

Select device/port type PowerOn Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0											PWD	TYP			
r											rh	rw			

Field	Bits	Type	Description
TYP	3:0	rw	**Device/port type** Indicates the specific type of this PCI Express function. It is also used to set the 'Device/Port Type' field of the 'PCI Express Capabilities Register'. The controller uses this input to determine the operating mode of the controller at run time. Encodings except for the below ones are not supported. 0_H PCIe endpoint 4_H PCIe root complex
PWD	4	rh	**Power down status of the PCIe controller** Indicate the power-down status of the PCIe controler. The power-down status can be triggered by PERST# reset, SW kernal reset, the hot reset and link reset. The S2A bridge and the SPB slave interface will refuse to accept new bus request when this bit is asserted.
0	31:5	r	**Reserved** Read as 0; should be written with 0.

19.4.2.51 Indicate link status

LNK_STAT Offset address: 0000374_H

Indicate link status Kernel Reset value: $0000\ 0004_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0									PERSTINT	EQINT	BWINT	LINKINT	LINKRQRST	Res	LINKUP
r									rh	rh	rh	rh	rh	r	rh

Field	Bits	Type	Description
LINKUP	0	rh	**PHY Link up/down indicator** 0_B Link is down 1_B Link is up
LINKRQRST	2	rh	**Reset request** Reset request because the link has gone down or the controller received a hot-reset request. A low level indicates that the controller is requesting external logic to reset the controller because the PHY link is down. The link reset request is delayed until all pending bus transfers have completed.
LINKINT	3	rh	**Link Autonomous Bandwidth Interrupt. The interrupt can be triggered when the LNK_INTEN.LINKINTEN is asserted.** indicate that the Link Autonomous Bandwidth Status bit has been set. This bit is asserted when all of the following conditions are true When INTx is used • The INTx assertion disable bit in the Command register is 0, and • The Link Autonomous Bandwidth Interrupt Enable bit in the Link Control register is set to 1, and • The Link Autonomous Bandwidth Interrupt Status bit in the Link Status register is set to 1 Or when MSI is used: • MSI or MSI-X is enabled. • The Link Autonomous Bandwidth Status register (Link Status register bit 15) is updated. • The Link Autonomous Bandwidth Interrupt Enable (Link Control register bit 11) is set.

(table continues...)

(continued)

Field	Bits	Type	Description
BWINT	4	rh	**Link Bandwidth Management Interrupt. The interrupt can be triggered when the LNK_INTEN.BWINTEN is asserted.** Indicate that the Link Bandwidth Management Status bit has been set. This bit is asserted when all of the following conditions are true: When INTx is used • The INTx assertion disable bit in the Command register is 0, and • The Bandwidth Management Interrupt Enable bit in the Link Control register is set to 1, and • The Bandwidth Management Interrupt Status bit in the Link Status register is set to 1. Or when MSI is used: • MSI or MSI-X is enabled. • The Link Bandwidth Management Status register (Link Control Status register bit 14) is updated • The Link Bandwidth Management Interrupt Enable (Link Control register bit 10) is set..
EQINT	5	rh	**Link Equalization Request. The interrupt can be triggered when the LNK_INTEN.EQINTEN is asserted.** Interrupt indicating to your application that the Link Equalization Request bit in the Link Status 2 Register has been set and the Link Equalization Request Interrupt Enable (Link Control 3 Register bit 1) is set.
PERSTINT	6	rh	**Interrupt status of the PERST#. The interrupt can be triggered when the LNK_INTEN.PERSTEN is asserted.** It indicates the PERST# interrupt (detecting PERST# become active 0) arrives.
0	31:7	r	**Reserved** Read as 0; should be written with 0.

19.4.2.52 Controller control

LNK_CORECTRL Offset address: 0000378_H

Controller control Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0														CLKREQEN	WAKEN
r														rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CLKREQ		WAKE		PERST		PERSTCLR	PERSTOVR	PERSTVAL	PERSTSTA	HOTRST	0	PFRQRTYEN	RQRTYEN	LTSSMEN	SRISMD
rw		rw		rw		w	rw	rw	rh	w	r	rw	rw	rw	rw

Field	Bits	Type	Description
SRISMD	0	rw	**SRIS operating mode** 0_B non-SRIS mode 1_B SRIS mode
LTSSMEN	1	rw	**Start link training.** Driven low default after cold, warm or hot reset to hold the LTSSM in the Detect state until your application is ready for the link training to begin. 0_B Hold LTSSM and delay link training 1_B Start link training
RQRTYEN	2	rw	**Defer incoming configuration requests** Provides a capability to defer incoming configuration requests until initialization is complete. When this bit is asserted, the controller completes incoming configuration requests with a configuration request retry status. Other incoming requests complete normally. When the Readiness Notification mechanism is supported, DRS messaging is blocked when RQRTYEN=1. When RQRTYEN=0 and MSG_INFO.DRSRDY=1, the controller autonomously transmits a DRS message when the link transitions from DL_Down to DL_Up.
PFRQRTYEN	3	rw	**Per Physical Function (PF) to defer incoming configuration requests** Provides a per Physical Function (PF) capability to defer incoming configuration requests until initialization is complete. When PFRQRTYEN is asserted for a certain PF, the controller completes incoming configuration requests with a configuration request retry status only for that particular PF; other incoming requests complete normally. When RQRTYEN is also asserted, the controller completes incoming configuration requests with a configuration request retry status for ALL PFs and VFs, regardless of the status of PFRQRTYEN. When the Readiness Notification mechanism is supported, DRS messaging is blocked when PFRQRTYEN =1 for at least one PF. The same applies when PFRQRTYEN =0 for all PFs, but RQRTYEN is asserted. When PFRQRTYEN =0 for a certain PF, and RQRTYEN =0, and MSG_INFO.DRSRDY=1, the controller autonomously transmits a DRS message when the link transitions from DL_Down to DL_Up.
HOTRST	5	w	**Request from your application to send a hot reset to the upstream port, RC only.** A downstream port (DSP) can hot reset an upstream port (USP) by sending two consecutive TS1 ordered sets with the hot reset bit asserted. This field requests from your application to send a hot reset to the upstream port. In an upstream port, you should not use this bit. Note: This bit is not used by the controller to set the SBR field in the BRIDGE_CTRL_INT_PIN_INT_LINE_REG register. During the transition from DL_Active to DL_inactive, assertion of this bit indicates a Surprise Down Error, but setting of SBR field in the BRIDGE_CTRL_INT_PIN_INT_LINE_REG register through FPI does not trigger a Surprise Down Error.
PERSTSTA	6	rh	**PERST# pin status** Indicate the PERST# pin status. 0_B Asserted (active low) 1_B Deasserted (active low)

(table continues...)

(continued)

Field	Bits	Type	Description
PERSTVAL	7	rw	**PERST# override value** This value decides the override value of the PERST# when the PERSTOVR is enabled. 0_B Asserted (active low) 1_B Deasserted (active low)
PERSTOVR	8	rw	**PERST# override enable** When this bit is enabled, the PERSTVAL bit will override the PERST# pin value in the controller.
PERSTCLR	9	w	**Clear the interrupt status of the PERSTINT.** Clear the interrupt status of the PERSTINT. 0_B Keep the status 1_B Clear the interrupt status
PERST	11:10	rw	**Alternate Input Select** Selects the alternate input for PERST#. 00_B Alternate Input A PERSTA_N 01_B Alternate Input B PERSTB_N 10_B Alternate Input A PERSTA_N 11_B Alternate Input A PERSTA_N
WAKE	13:12	rw	**Alternate input and output select** Selects the alternate input and output for WAKE#. 00_B Alternate input and output A WAKEA_IN_N 01_B Alternate input and output B WAKEB_IN_N 10_B Alternate input and output A WAKEA_IN_N 11_B Alternate input and output A WAKEA_IN_N
CLKREQ	15:14	rw	**Alternate input and output select** Selects the alternate input and output for CLKREQ#. 00_B Alternate input and output A CLKREQA_IN_N 01_B Alternate input and output B CLKREQB_IN_N 10_B Alternate input and output A CLKREQA_IN_N 11_B Alternate input and output A CLKREQA_IN_N
WAKEN	16	rw	**WAKE# enable control.** Enable the WAKE# input and output. If this bit is disabled, the alternative selection of the WAKE bit will have no effect 0_B WAKE# disable. None of the alternative pins is driven. 1_B WAKE# enable. One of the alternative pins is driven.
CLKREQEN	17	rw	**CLKREQ# enable control.** Enable the CLKREQ# input and output. If this bit is disabled, the alternative selection of the CLKREQ bit will have no effect 0_B CLKREQ# disable. None of the alternative pins is driven. 1_B CLKREQ# enable. One of the alternative pins is driven.
0	4, 31:18	r	**Reserved** Read as 0; should be written with 0.

19.4.2.53 Enable the interrupt requests to SRC_PCIELNK

LNK_INTEN Offset address: 000037C$_H$

Enable the interrupt requests to SRC_PCIELNK Kernel Reset value: 0000 0000$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												PERSTEN	EQINTEN	BWINTEN	LINKINTEN
r												rw	rw	rw	rw

Field	Bits	Type	Description
LINKINTEN	0	rw	**Enable the LINKINT** 0_B Disable interrupt 1_B Enable interrupt
BWINTEN	1	rw	**Enable the BWINT bit.** 0_B Disable interrupt 1_B Enable interrupt
EQINTEN	2	rw	**Enable the EQINT bit.** 0_B Disable interrupt 1_B Enable interrupt
PERSTEN	3	rw	**Enable the PERSTINT bit.** 0_B Disable interrupt 1_B Enable interrupt
0	31:4	r	**Reserved** Read as 0; should be written with 0.

19.4.2.54 Rx control

TXRX_RXCTRL Offset address: 0000380$_H$

Rx control Kernel Reset value: 0000 0000$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0									CPLLEN						
r									rh						

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CPLLEN					CPLATTR		CPLTC			0					CPLTO
rh					rh		rh			r					rh

Field	Bits	Type	Description
CPLTO	0	rh	**Indicates that the completion TLP for a request has not been received**
CPLTC	8:6	rh	**The Traffic Class of the timed out completion**
CPLATTR	10:9	rh	**The Attributes field of the timed out completion**
CPLLEN	22:11	rh	**Length (in bytes) of the timed out completion.** For a split completion, it indicates the number of bytes remaining to be delivered when the completion timed out.
0	5:1, 31:23	r	**Reserved** Read as 0; should be written with 0.

19.4.2.55 Tx control 0

TXRX_TXCTRL0 Offset address: 0000384_H

Tx control 0 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0							CPLLEN								
r							rh								

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CPLLEN			CPLATTR		CPLTC			CPLFNUM			CPLTO	PRTYERR			BLCKTLP
rh			rh		rh			rh			rh	rh			rh

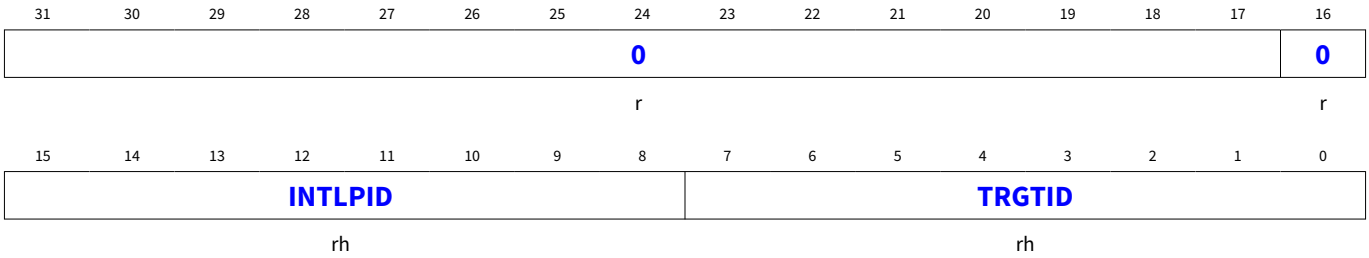
Field	Bits	Type	Description
BLCKTLP	0	rh	**Block TLP generation** Indicates that your application must stop generating new outgoing request TLPs due to the current power management state. Your application can continue to generate completion TLPs. This output is not used in RC mode.
PRTYERR	3:1	rh	**Indicates that the controller detected a datapath parity error** The controller handles the parity error as an uncorrectable error. • PRTYERR[0]: Parity error at front end of the transmit datapath. • PRTYERR[1]: Parity error at back end of the transmit datapath. • PRTYERR[2]: Parity error the receive datapath. They do not indicate which packet contained the parity error.
CPLTO	4	rh	**No completion for an incoming request** Information about the timed out completion is available the fields listed later. When a completion timeout occurs, the controller does not clear the corresponding entry from the completion lookup table.
CPLFNUM	7:5	rh	**The function number of the timed out completion.**
CPLTC	10:8	rh	**The TC of the timed out completion.**
CPLATTR	12:11	rh	**The Attributes value of the timed out completion.**

(table continues...)

(continued)

Field	Bits	Type	Description
CPLLEN	24:13	rh	**The Length of the timed out completion.**
0	31:25	r	**Reserved** Read as 0; should be written with 0.

19.4.2.56 Tx control 1

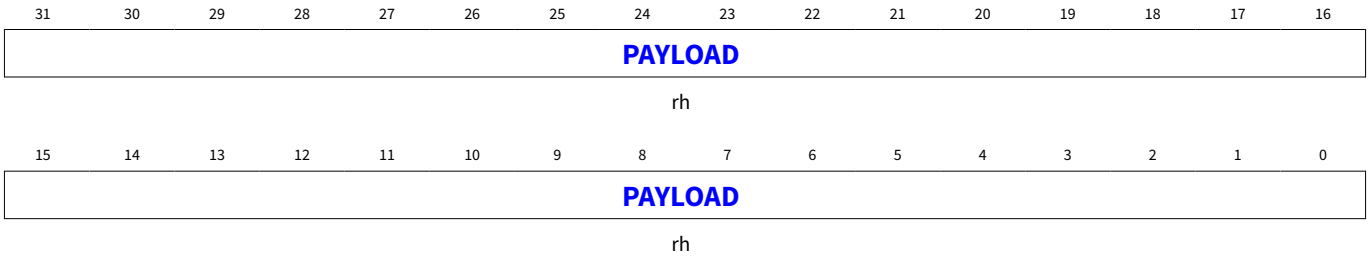


Field	Bits	Type	Description
TRGTID	7:0	rh	**The target completion LUT lookup ID of the timed out completion**
INTLPID	15:8	rh	**The target completion LUT lookup ID for the incoming request TLP.**
0	16, 31:17	r	**Reserved** Read as 0; should be written with 0.

19.4.2.57 Message payload z information

PAYLOADz (z=0-1) Offset address: $000038C_H+z*4$

Message payload z information Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
PAYLOAD	31:0	rh	**Received message header information.** When a vendor-defined or ltr message is received, the controller maps the message payload to the Rx TLP header dwords as follows: • PAYLOAD0 = bytes 12-15 (4th dword), where [7:0] =byte 15 • PAYLOAD1 = bytes 8-11 (3rd dword)

19.4.2.58 Message information

MSG_INFO Offset address: 0000394_H

Message information Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0							DRSEN	DRSCLR	DRSSTAT	DRSRDY	0	SURPDWN	FATERR	NONFAT	CORERR
r							rw	w	rh	rw	r	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
REQID															
rh															

Field	Bits	Type	Description
REQID	15:0	rh	**The requester ID of the received Message.** • [2:0]: Function number • [7:3]: Device number • [15:8]: Bus number
CORERR	16	rh	**Sent Correctable Error, EP only.** Core has sent a message towards the Root Complex indicating that an Rx TLP that contained an error, and that can be corrected, was received by the EndPoint.
NONFAT	17	rh	**Sent Non-Fatal Error, EP only.** Controller has sent a message towards the Root Complex indicating that an Rx TLP that contained an non-fatal error, and that can not be corrected, was received by the EndPoint.
FATERR	18	rh	**Sent Fatal Error, EP only.** Controller has sent a message towards the Root Complex indicating that an Rx TLP that contained a fatal error, and that can not be corrected, was received by the EndPoint
SURPDWN	19	rh	**Indicates that a surprise down event is occurring in the controller.**
DRSRDY	21	rw	**Defers DRS messaging when set to '0', EP only.**
DRSSTAT	22	rh	**DRS Message Received Interrupt, RC only. The interrupt can be triggered when the DRSEN is asserted.** The DSP controller asserts this bit when all of the following are true: • It receives a DRS message • PCIE_CAP_DRS_SIGNALING_CONTROL in LINK_CONTROL_LINK_STATUS_REG is 2'b01 • MSI or MSI-X is enabled
DRSCLR	23	w	**Clear the DRS state, RC only.** 0_B Keep the status 1_B Clear the interrupt status

(table continues...)

(continued)

Field	Bits	Type	Description
DRSEN	24	rw	**Enable the DRS interrupt, RC only.** 0_B Disable interrupt 1_B Enable interrupt
0	20, 31:25	r	**Reserved** Read as 0; should be written with 0.

19.4.2.59 Power management control

PM_CTRL Offset address: 0000398_H

Power management control Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0														TRN	TRN ACK
r														rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PME MSG	TUR NOFF	0				CLK RM	PMEI NT	CLKP MEN	CLK REQ	XFER PND	REQ EXIT L1	RDYE NTR L23	REQ ENT RL1	AUX PWR DET	WKU PCM D
rh	w	r				rw	rh	rw	rh	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
WKUPCMD	0	rw	**Wake Up.** Wake Up. If PME is enabled and PME support is configured for current PMCSR D-state asserting this signal causes the controller to wake from either L1 or L2 state.
AUXPWRDET	1	rw	**Auxiliary Power Detected.** Used to report to the host software that auxiliary power (Vaux) is present.
REQENTRL1	2	rw	**Application request to Enter L1 ASPM state.** This bit is for use by applications that need to control L1 entry instead of using the L1 entry timer as defined in the PCI Express Specification. It is only effective when L1 is enabled. The controller latches this request when in L0 or L0s; to be acted upon later. Note: The controller ignores this input in RC mode
RDYENTRL23	3	rw	**Application Ready to Enter L23.** Indication from your application that it is ready to enter the L23 state. This bit is provided for applications that must control L23 entry (in case certain tasks must be performed before going into L23). The controller delays sending PM_Enter_L23 (in response to PM_Turn_Off) until this signal becomes active. When this signal has been asserted by the application, it must be kept asserted until L2 entry has completed.

(table continues...)

(continued)

Field	Bits	Type	Description
REQEXITL1	4	rw	**Application request to Exit L1.** Request from your application to exit L1. It is only effective when L1 is enabled.
XFERPND	5	rw	**Indicates that your application has transfers pending** If the entry into L1 is already in progress, assertion of app_xfer_pending causes an exit from L1. This is a level signal used to inform the controller about the state of external queues and pipeline stages that contain transactions to be transmitted by the controller. The controller uses this information to determine when to enter/exit L1. When this signal is asserted, it indicates that there are transactions outside the controller that the controller needs to transmit. When de-asserted, it indicates that there are no transactions outside the controller. The controller responds to an assertion on this signal as follows: • Upstream Ports: Prevents generation of requests to enter L1. Triggers exit if already in L1. • Downstream Ports: Triggers exit if already in L1.
CLKREQ	6	rh	**Status of the CLKREQ#** Status of the CLKREQ# bidirectional CMOS board-level signal. This should be connected to the GPIO. Used by the controller to determine when to enter and exit L1 Substates when using the CLKREQ#-based mechanism. For more details, see 'L1 Substates'.
CLKPMEN	7	rw	**Clock PM feature enabled by application.** Used to inhibit the programming of the Clock PM in Link Control Register.
PMEINT	8	rh	**PME interrupt received. The interrupt can be triggered when the PM_INTEN.PMEINTEN is asserted.** The controller asserts this bit when all of the following conditions are true: • The INTx Assertion Disable bit in the Command register is 0. • The PME Interrupt Enable bit in the Root Control register is set to 1. • The PME Status bit in the Root Status register is set to 1. Or • MSI or MSI-X is enabled. • The PME Interrupt Enable bit in the Root Control register is set to 1. • The PME Status bit in the Root Status register is set to 1.
CLKRM	9	rw	**Indicates that the application is ready to have reference clock removed.** The application should set this bit to 1'b1 if it supports reference clock removal. If the application does not want to remove reference clock it should set this bit to 1'b0. This bit should be asserted or de-asserted when L1 CPM or L1 Sub-states are being configured, it should not be changed dynamically during L1.
TURNOFF	14	w	**Request from your application to generate a PM_Turn_Off message.** You must assert this signal for one clock cycle. You must not assert the same bit again, until the previous message has been transmitted.

(table continues...)

(continued)

Field	Bits	Type	Description
PMEMSG	15	rh	**The controller received a PM_PME message.The interrupt can be triggered when the PM_INTEN.PMEMSGEN is asserted.**
TRNACK	16	rh	**The controller received a PME_TO_Ack message. Upstream port: Reserved. The interrupt can be triggered when the PM_INTEN.TOACKEN is asserted.**
TRN	17	rh	**The controller received a PME Turnoff message. The interrupt can be triggered when the PM_INTEN.TOEN is asserted.**
0	13:10, 31:18	r	**Reserved** Read as 0; should be written with 0.

19.4.2.60 Power management status 0

PM_STAT0 Offset address: 000039C$_H$

Power management status 0 Kernel Reset value: 0000 0000$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
L1SUBSTAT			SLVSTAT					MSTRSTAT					LINKSTL2E	LINKSTL2	LINKSTL1
rh			rh					rh					rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	DSTAT			0	WAKE	LTSSMSTAT						CURNTSTAT		
r	r	rh			r	rh	rh						rh		

Field	Bits	Type	Description
CURNTSTAT	2:0	rh	**Indicates the current power state.** 000_B L0 000_B L0s 000_B L1 000_B L2 (Aux Power Present) 000_B L3

(table continues...)

(continued)

Field	Bits	Type	Description
LTSSMSTAT	8:3	rh	**Current state of the LTSSM. Encoding is defined as follows:** 00_H S_DETECT_QUIET 01_H S_DETECT_ACT 02_H S_POLL_ACTIVE 03_H S_POLL_COMPLIANCE 04_H S_POLL_CONFIG 05_H S_PRE_DETECT_QUIET 06_H S_DETECT_WAIT 07_H S_CFG_LINKWD_START 08_H S_CFG_LINKWD_ACEPT 09_H S_CFG_LANENUM_WAI $0A_H$ S_CFG_LANENUM_ACEPT $0B_H$ S_CFG_COMPLETE $0C_H$ S_CFG_IDLE $0D_H$ S_RCVRY_LOCK $0E_H$ S_RCVRY_SPEED $0F_H$ S_RCVRY_RCVRCFG 10_H S_RCVRY_IDLE 11_H S_L0 12_H S_L0S 13_H S_L123_SEND_EIDLE 14_H S_L1_IDLE 15_H S_L2_IDLE 16_H S_L2_WAKE 17_H S_DISABLED_ENTRY 18_H S_DISABLED_IDLE 19_H S_DISABLED $1A_H$ S_LPBK_ENTRY $1B_H$ S_LPBK_ACTIVE $1C_H$ S_LPBK_EXIT $1D_H$ S_LPBK_EXIT_TIMEOUT $1E_H$ S_HOT_RESET_ENTRY $1F_H$ S_HOT_RESET 20_H S_RCVRY_EQ0 21_H S_RCVRY_EQ1 22_H S_RCVRY_EQ2 23_H S_RCVRY_EQ3
WAKE	9	rh	**Wake Up. Driving WAKE# for USP**
DSTAT	13:11	rh	**The current power management D-state of the function:** 000_B D0 001_B D1 010_B D2 011_B D3 100_B Uninitialized 101_B Reserved

(table continues...)

(continued)

Field	Bits	Type	Description
LINKSTL1	16	rh	**Power management is in L1 state.**
LINKSTL2	17	rh	**Power management is in L2 state.**
LINKSTL2E	18	rh	**Power management is exiting L2 state.** Not applicable for downstream port.
MSTRSTAT	23:19	rh	**Power management master FSM state.** OTGB trace
SLVSTAT	28:24	rh	**Power management slave FSM state.** OTGB trace
L1SUBSTAT	31:29	rh	**Power management L1 sub-states FSM state**
0	10, 14, 15	r	**Reserved** Read as 0; should be written with 0.

19.4.2.61 Power management status 1

PM_STAT1 Offset address: 00003A4$_H$

Power management status 1 Kernel Reset value: 0000 0000$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												0	LINKSTL1SUB	0	WAKINT
r												r	rh	r	rh

Field	Bits	Type	Description
WAKINT	0	rh	**For DSP, indicate that the WAKE# is driven from high to low. The interrupt can be triggered when the PM_INTEN.WAKINTEN is asserted.** An interrupt can be triggered by this transition.
LINKSTL1SUB	2	rh	**Power management is in L1 substate.** Indicates when the link has entered L1 substates.
0	1, 3, 31:4	r	**Reserved** Read as 0; should be written with 0.

19.4.2.62 Power management interrupt clear

PM_INTCLR Offset address: 00003A8$_H$

Power management interrupt clear Kernel Reset value: 0000 0000$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0											WAKINTCLR	TOCLR	TOACKCLR	PMEMSGCLR	PMEINTCLR
r											w	w	w	w	w

Field	Bits	Type	Description
PMEINTCLR	0	w	**Clear the PMEINT bit.**
PMEMSGCLR	1	w	**Clear the PMEMSG bit.**
TOACKCLR	2	w	**Clear the TRNACK bit.**
TOCLR	3	w	**Clear the TRN bit.**
WAKINTCLR	4	w	**Clear the WAKINT bit.**
0	31:5	r	**Reserved** Read as 0; should be written with 0.

19.4.2.63 Power management interrupt enable

PM_INTEN Offset address: $00003AC_H$

Power management interrupt enable Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0											WAKINTEN	TOEN	TOACKEN	PMEMSGEN	PMEINTEN
r											rw	rw	rw	rw	rw

Field	Bits	Type	Description
PMEINTEN	0	rw	**Enable PME interrupt** 0_B Disable interrupt 1_B Enable interrupt
PMEMSGEN	1	rw	**Enable the PMEMSG interrupt** 0_B Disable interrupt 1_B Enable interrupt
TOACKEN	2	rw	**Enable the TRNACK interrupt** 0_B Disable interrupt 1_B Enable interrupt

(table continues...)

(continued)

Field	Bits	Type	Description
TOEN	3	rw	**Enable the TRN interrupt** 0_B Disable interrupt 1_B Enable interrupt
WAKINTEN	4	rw	**Enable the WAKINT interrupt** 0_B Disable interrupt 1_B Enable interrupt
0	31:5	r	**Reserved** Read as 0; should be written with 0.

19.4.2.64 Error status

ERR_STAT Offset address: $00003B0_H$

Error status Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

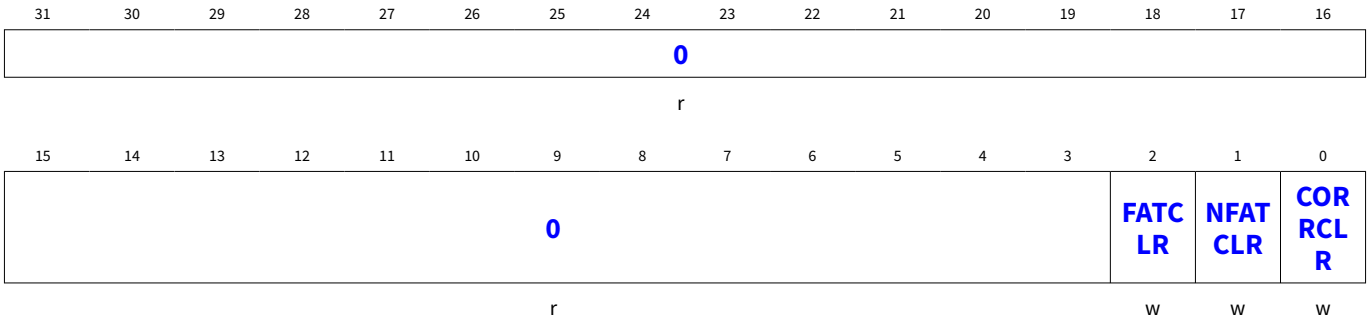
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0													FAT	NFAT	CORR
r													rh	rh	rh

Field	Bits	Type	Description
CORR	0	rh	**Received an ERR_COR message. The interrupt can be triggered when the ERR_INTEN.CORREN is asserted.** Indicates that the controller received an ERR_COR message. The controller makes the message header available in the MSG_PAYLOAD.
NFAT	1	rh	**Indicates that the controller received an ERR_NONFATAL message. The interrupt can be triggered when the ERR_INTEN.NFATEN is asserted.**
FAT	2	rh	**Indicates that the controller received an ERR_FATAL message. The interrupt can be triggered when the ERR_INTEN.FATEN is asserted.**
0	31:3	r	**Reserved** Read as 0; should be written with 0.

19.4.2.65 Error clear

ERR_INTCLR Offset address: $00003B4_H$

Error clear Kernel Reset value: $0000\ 0000_H$

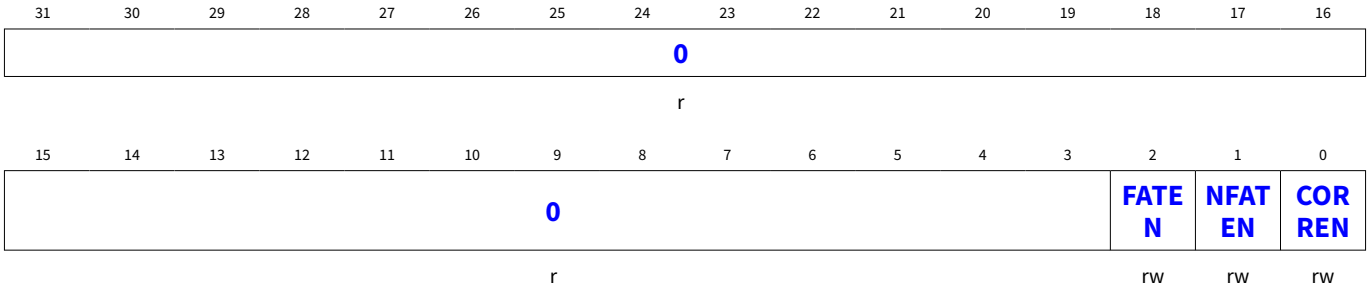


Field	Bits	Type	Description
CORRCLR	0	w	**Clear the CORR bit.** 0_B Keep the status 1_B Clear the interrupt status
NFATCLR	1	w	**Clear the NFAT bit.** 0_B Keep the status 1_B Clear the interrupt status
FATCLR	2	w	**Clear the FAT bit.** 0_B Keep the status 1_B Clear the interrupt status
0	31:3	r	**Reserved** Read as 0; should be written with 0.

19.4.2.66 Error enable

ERR_INTEN Offset address: $00003B8_H$

Error enable Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
CORREN	0	rw	**Enable the CORR bit.** 0_B Disable interrupt 1_B Enable interrupt
NFATEN	1	rw	**Enable the NFAT bit.** 0_B Disable interrupt 1_B Enable interrupt

(table continues...)

(continued)

Field	Bits	Type	Description
FATEN	2	rw	**Enable the FAT bit.** 0_B Disable interrupt 1_B Enable interrupt
0	31:3	r	**Reserved** Read as 0; should be written with 0.

19.4.2.67 Debug status

DBG_STAT Offset address: $00003BC_H$

Debug status Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0									Res	QNEMPTY	0	XFERPD	DMAPD	BRDGPD	LINKUP
r									r	rh	r	rh	rh	rh	rh

Field	Bits	Type	Description
LINKUP	0	rh	**Data link layer up/down indicator:** This status from the flow control initialization state machine indicates that flow control has been initiated and the Data link layer is ready to transmit and receive packets.
BRDGPD	1	rh	**AXI Slave non-DBI transfer pending status.** Indicates AXI Slave Read or Write transfers are pending, that is, AXI Slave transfers are awaiting a response from the controller. For debugging purposes.
DMAPD	2	rh	**eDMA transfer pending status.** Indicates eDMA Write or Read Channel transfers are pending, that is, DMA Write or Read Channels have not finished transferring data. For debugging purposes.
XFERPD	3	rh	**Receive request pending status.** Indicates Receive TLP requests are pending, that is, requests sent to the bus interfaces are awaiting a response from your application. For debugging purposes.
QNEMPTY	5	rh	**Level indicating that the receive queues contain TLP header/data**
0	4, 31:7	r	**Reserved** Read as 0; should be written with 0.

19.4.2.68 Debug information

DBG_DATA Offset address: $00003C0_H$

Debug information Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RFCDAT															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RFCDAT															
rh															

Field	Bits	Type	Description
RFCDAT	31:0	rh	**The data from the previously received flow control update DLLP.** .

19.4.2.69 LTR message

LTR_GNRTN Offset address: $00003C4_H$

LTR message Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LTRLAT															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LTRLAT															
rw															

Field	Bits	Type	Description
LTRLAT	31:0	rw	**LTR message that your application is requesting to send.**

19.4.2.70 LTR control

LTR_CTRL Offset address: $00003CC_H$

LTR control Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0						RLTREN	RLTRCLR	RLTR	0						MSGREQ
r						rw	w	rh	r						rwh

Field	Bits	Type	Description
MSGREQ	0	rwh	**Trigger to send an LTR message.** Once asserted, it must remain asserted until the controller grants the reqeust. It gets cleared when the request is granted. .
RLTR	7	rh	**Received an LTR message. The interrupt can be triggered when the RLTREN is asserted.** The controller makes the message header available the MSG.PAYLOAD. It is also available the PL_LTR_LATENCY_OFF.
RLTRCLR	8	w	**Clear the RLTR bit.** 0_B Keep the status 1_B Clear the interrupt status
RLTREN	9	rw	**Enable the RLTR bit.** 0_B Disable interrupt 1_B Enable interrupt
0	6:1, 31:10	r	**Reserved** Read as 0; should be written with 0.

19.4.2.71 PTM control

PTM_CTRL Offset address: $00003D0_H$

PTM control Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			DUPEN	DUPCLR	DUP	TRNEN	TRNCLR	TRN	UPDATED	UPDATING	TRGG	AUTOUPDT	MSTSTRB	RDYVLDT	CNTXTVLD
r			rw	w	rh	rw	w	rh	rh	rh	rh	rw	w	rh	rh

Field	Bits	Type	Description
CNTXTVLD	0	rh	**Context Valid.**
RDYVLDT	1	rh	**PTM Responder Ready to Validate.**
MSTSTRB	2	w	**PTM External Master Time Strobe.**
AUTOUPDT	3	rw	**Update the Context and Clock automatically** Configure the controller to update the PTM Requester Context and Clock automatically every 10ms.
TRGG	4	rh	**Indicates that a PTM Requester manual update trigger is allowed**
UPDATING	5	rh	**Indicates that a PTM update is in progress.**
UPDATED	6	rh	**Indicates that the controller has updated the Local Clock**

(table continues...)

(continued)

Field	Bits	Type	Description
TRN	7	rh	**PTM Requester Response Timeout status. The interrupt can be triggered when the TRNEN is asserted.** Indicating 100 us timeout occurred while waiting for a PTM Response or PTM ResponseD message.
TRNCLR	8	w	**Clear the TRN bit.** 0_B Keep the status 1_B Clear the interrupt status
TRNEN	9	rw	**Enable the TRN bit.** 0_B Disable interrupt 1_B Enable interrupt
DUP	10	rh	**PTM Requester Duplicate Received. The interrupt can be triggered when the DUPEN is asserted.** Single-cycle pulse indicating PTM Requester received a duplicate TLP while,
DUPCLR	11	w	**Clear the DUP bit.** 0_B Keep the status 1_B Clear the interrupt status
DUPEN	12	rw	**Enable the DUP bit.** 0_B Disable interrupt 1_B Enable interrupt
0	31:13	r	**Reserved** Read as 0; should be written with 0.

19.4.2.72 Local Clock z value

LCLCLKz (z=0-1) Offset address: $00003D4_H+z*4$

Local Clock z value Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LCLCLK															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LCLCLK															
rh															

Field	Bits	Type	Description
LCLCLK	31:0	rh	**Local Clock value.**

19.4.2.73 Clock z correction

CLKCORRz (z=0-1) Offset address: $00003DC_H+z*4$

Clock z correction Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CLKCORR															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CLKCORR															
rh															

Field	Bits	Type	Description
CLKCORR	31:0	rh	**Amount by which Local Clock has been corrected**

19.4.2.74 PTM z External Master Time

MSKCLKz (z=0-1) Offset address: $00003E4_H+z*4$

PTM z External Master Time Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MSKCLK															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MSKCLK															
rw															

Field	Bits	Type	Description
MSKCLK	31:0	rw	**PTM External Master Time.**

19.4.2.75 Hot-plug control

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

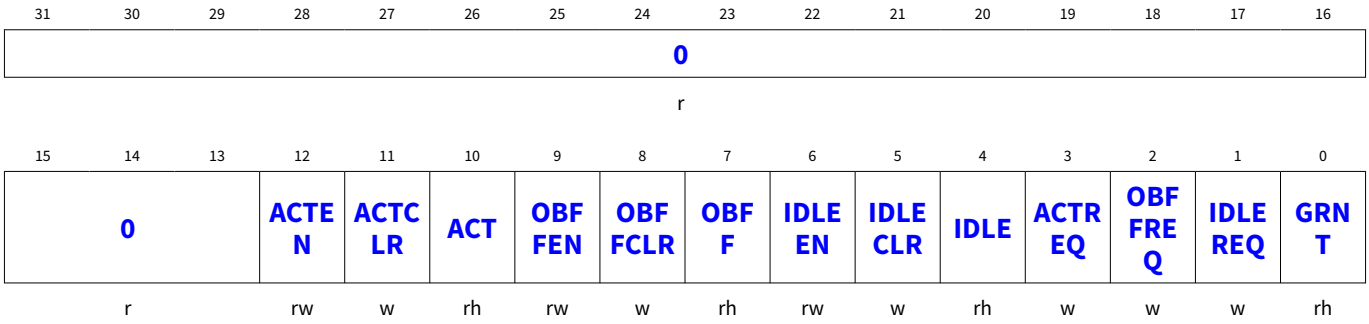
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												HPINTCLR	HPINTEN	HPINT	HPPME
r												w	rw	rh	rh

Field	Bits	Type	Description
HPPME	0	rh	**The controller asserts hp_pme when all of the following conditions are true:** • The PME Enable bit in the Power Management Control and Status register is set to 1. • Any bit in the Slot Status register transitions from 0 to 1 and the associated event notification is enabled in the Slot Control register. The controller does not check if the PM state is D1, D2, or D3hot. It is up to your application to check the value pm_dstate to make sure the device is in D1, D2, or D3hot. There is one bit of HPPME for each configured function. The controller pulses this bit output only when any hot plug status bit changes from 0 to 1 (as is HPINT). hp_int stays asserted as long as the status bit is set. In addition, it asserts HPPME only if PME is enabled, but it does not matter if hot-plug interrupts are enabled.
HPINT	1	rh	**The controller asserts hp_int when all of the following conditions are true:** • The INTx Assertion Disable bit in the Command register is 0. • Hot-Plug interrupts are enabled in the Slot Control register. • Any bit in the Slot Status register is equal to 1, and the associated event notification is enabled in the Slot Control register. Or • MSI or MSI-X is enabled. • Hot-Plug interrupts are enabled in the Slot Control register. • Any bit in the Slot Status register transitions from 0 to 1 and the associated event notification is enabled in the Slot Control register.
HPINTEN	2	rw	**Hot plug enable** 0_B Disable interrupt 1_B Enable interrupt
HPINTCLR	3	w	**Clear the HPINT bit.** 0_B Keep the status 1_B Clear the interrupt status
0	31:4	r	**Reserved** Read as 0; should be written with 0.

19.4.2.76 OBFF message control

OBFFMSG Offset address: $00003F0_H$

OBFF message control Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
GRNT	0	rh	**Request accepted to generate an OBFF message**
IDLEREQ	1	w	**To generate an 'IDLE' OBFF message.** Request from your application to generate an 'IDLE' OBFF message. Only usable in a downstream port.
OBFFREQ	2	w	**Request to generate an 'OBFF' OBFF message.** Request from your application to generate an 'OBFF' OBFF message. Only usable in a downstream port.
ACTREQ	3	w	**Request to generate a 'CPU Active' OBFF message.** Request from your application to generate a 'CPU Active' OBFF message. Only usable in a downstream port.
IDLE	4	rh	**The controller received an 'IDLE' OBFF message. The interrupt can be triggered when the IDLEEN is asserted.** Indicating that the controller received an 'IDLE' OBFF message. Only usable in an upstream port
IDLECLR	5	w	**Clear the IDLE bit** 0_B Keep the status 1_B Clear the interrupt status
IDLEEN	6	rw	**Enable the idle bit.** 0_B Disable interrupt 1_B Enable interrupt
OBFF	7	rh	**The controller received an 'OBFF' OBFF message. The interrupt can be triggered when the OBFFEN is asserted.** Indicating that the controller received an 'OBFF' OBFF message. Only usable in an upstream port.
OBFFCLR	8	w	**Clear the OBFFbit** 0_B Keep the status 1_B Clear the interrupt status
OBFFEN	9	rw	**Enable the OBFFbit.** 0_B Disable interrupt 1_B Enable interrupt
ACT	10	rh	**Received a 'CPU Active' OBFF message. The interrupt can be triggered when the ACTEN is asserted.** Indicating that the controller received a 'CPU Active' OBFF message. Only usable in an upstream port.

(table continues...)

(continued)

Field	Bits	Type	Description
ACTCLR	11	w	**Clear the ACT bit** 0_B Keep the status 1_B Clear the interrupt status
ACTEN	12	rw	**Enable the ACT bit.** 0_B Disable interrupt 1_B Enable interrupt
0	31:13	r	**Reserved** Read as 0; should be written with 0.

19.4.2.77 OBFF WAKE control

OBFFWAKE Offset address: $00003F4_H$

OBFF WAKE control Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0					EFSMSTAT				DFSMSTAT				INITSTAT		DEC RST
r					rh				rh				rw		w

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ERREN	ERRCLR	WAKERR	ACTEN	ACTCLR	ACT	OBFFEN	OBFFCLR	OBFF	IDLEEN	IDLECLR	IDLE	ACTREQ	OBFFREQ	IDLEREQ	GRNT
rw	w	rh	rw	w	rh	rw	w	rh	rw	w	rh	w	w	w	rh

Field	Bits	Type	Description
GRNT	0	rh	**Request received to generate a WAKE# signal.** Indicates that the controller has accepted your request to generate a WAKE# signal. Only usable in a downstream port
IDLEREQ	1	w	**Request to generate an 'IDLE' WAKE# signal** Request from your application to generate an 'IDLE' WAKE# signalOnly usable in a downstream port.
OBFFREQ	2	w	**Request to generate an 'OBFF' WAKE# signal.** Request from your application to generate an 'OBFF' WAKE# signal. Only usable in a downstream port.
ACTREQ	3	w	**Request to generate a 'CPU Active' WAKE# signal.** Request from your application to generate a 'CPU Active' WAKE# signal. Only usable in a downstream port.
IDLE	4	rh	**Received an 'IDLE' OBFF message. The interrupt can be triggered when the IDLEEN is asserted.** Indicating that the controller received an 'IDLE' OBFF message. Only usable in an upstream port

(table continues...)

(continued)

Field	Bits	Type	Description
IDLECLR	5	w	**Clear the IDLE bit** 0_B Keep the status 1_B Clear the interrupt status
IDLEEN	6	rw	**Enable the idle bit.** 0_B Disable interrupt 1_B Enable interrupt
OBFF	7	rh	**Received an 'OBFF' OBFF message. The interrupt can be triggered when the OBFFEN is asserted.** Indicating that the controller received an 'OBFF' OBFF message. Only usable in an upstream port.
OBFFCLR	8	w	**Clear the OBFFbit** 0_B Keep the status 1_B Clear the interrupt status
OBFFEN	9	rw	**Enable the OBFFbit.** 0_B Disable interrupt 1_B Enable interrupt
ACT	10	rh	**Received a 'CPU Active' OBFF message. The interrupt can be triggered when the ACTEN is asserted.** Indicating that the controller received a 'CPU Active' OBFF message. Only usable in an upstream port.
ACTCLR	11	w	**Clear the ACT bit** 0_B Keep the status 1_B Clear the interrupt status
ACTEN	12	rw	**Enable the ACT bit.** 0_B Disable interrupt 1_B Enable interrupt
WAKERR	13	rh	**The WAKE# signal decoder has received an invalid WAKE# pattern. The interrupt can be triggered when the ERREN is asserted.**
ERRCLR	14	w	**Clear the WAKERR bit** 0_B Keep the status 1_B Clear the interrupt status
ERREN	15	rw	**Enable the WAKERR bit.** 0_B Disable interrupt 1_B Enable interrupt
DECRST	16	w	**Reset the OBFF WAKE# signaling decoder.**
INITSTAT	18:17	rw	**The intial state after the OBFF WAKE# signaling decoder is reset by teh DECRST.** 00_B "IDLE" 01_B "OBFF" 10_B "CPU active" 11_B Reserved

(table continues...)

(continued)

Field	Bits	Type	Description
DFSMSTAT	22:19	rh	**Current decoder FSM state for debug. Not equal to OBFF code.**
EFSMSTAT	26:23	rh	**Current encoder FSM state for debug. Not equal to OBFF code.**
0	31:27	r	**Reserved** Read as 0; should be written with 0.

19.4.2.78 PCIe configuration control

app_dev_num_i and app_bus_num_i for ATU MEM to CFG translation

CFGBD Offset address: $00003F8_H$

PCIe configuration control Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			DEVNUM					BUSNUM							
r			rw					rw							

Field	Bits	Type	Description
BUSNUM	7:0	rw	**Bus number** Used to indicate the requester bus number information when a RC sends out CFG TLPs.
DEVNUM	12:8	rw	**Device number** Used to indicate the requester device number information when a RC sends out CFG TLPs.
0	31:13	r	**Reserved** Read as 0; should be written with 0.

19.4.2.79 ELBI x mailbox register

DATAx (x=0-7) Offset address: $00003FC_H+x*4$

ELBI x mailbox register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DATA															
rwh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA															
rwh															

Field	Bits	Type	Description
DATA	31:0	rwh	**ELBI mailbox registers.** These registers can be used as buffers to communicate information for debugging purpose.

19.4.2.80 DMA link list control register

LLCTRL Offset address: 000041C$_H$

DMA link list control register Kernel Reset value: 0000 0000$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

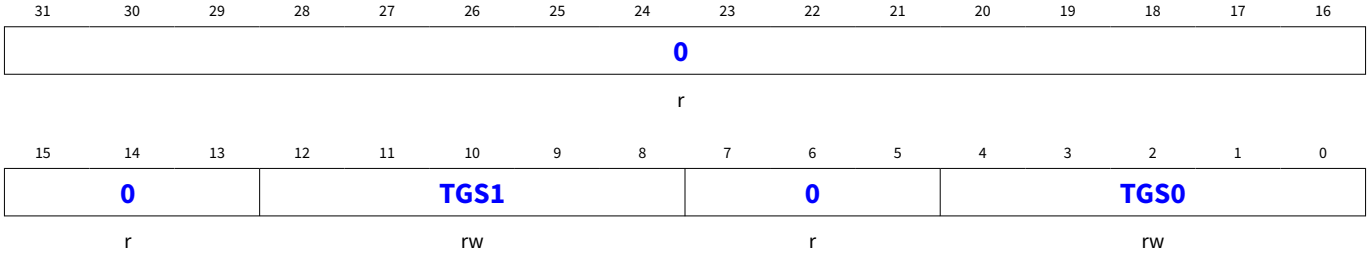
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RDDONE3	RDDONE2	RDDONE1	RDDONE0	WRDONE3	WRDONE2	WRDONE1	WRDONE0	RDTOGG3	RDTOGG2	RDTOGG1	RDTOGG0	WRTOGG3	WRTOGG2	WRTOGG1	WRTOGG0
rh	rh	rh	rh	rh	rh	rh	rh	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
WRTOGGz (z=0-3)	z	rw	**DMA Write engine descriptor transfer go toggle** DMA Write engine descriptor transfer go toggle signal. This signal is ignored whenever a DMA channel operates in non-linked mode.
RDTOGGz (z=0-3)	z+4	rw	**DMA Read engine descriptor transfer go toggle** DMA Read engine descriptor transfer go toggle signal. This signal is ignored whenever a DMA channel operates in non-linked mode.
WRDONEz (z=0-3)	z+8	rh	**DMA write engine descriptor transfer done toggle** DMA write engine descriptor transfer done toggle signal. After a doorbell command, this signal is de-asserted. When a channel is operating in non-linked list mode this signal does not toggles
RDDONEz (z=0-3)	z+12	rh	**DMA write engine descriptor transfer done toggle** DMA read engine descriptor transfer done toggle signal. After a doorbell command, this signal is de-asserted. When a channel is operating in non-linked list mode this signal does not toggles
0	31:16	r	**Reserved** Read as 0; should be written with 0.

19.4.2.81 PCIe OCDS Trigger Set Select

OTSS Offset address: 0000420$_H$

PCIe OCDS Trigger Set Select Debug Reset value: 0000 0000$_H$

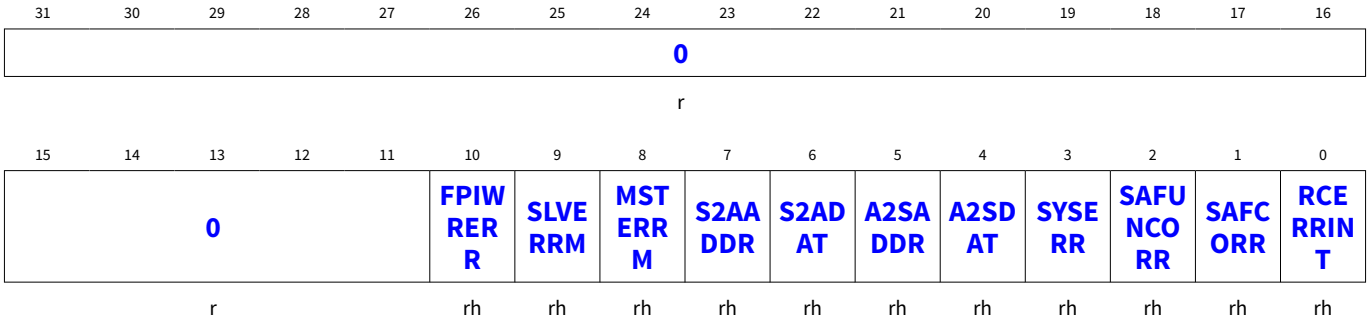


Field	Bits	Type	Description
TGS0	4:0	rw	**Trigger set select 0** 00_H No Trigger Set output 01_H Trigger Set 0 Trigger Set TS16_PCIe0 02_H Trigger Set 1 Trigger Set TS16_PCIe1 03_H Trigger Set 2 Trigger Set TS16_PCIe2 04_H Reserved
TGS1	12:8	rw	**Trigger Set for OTGB1** 00_H No Trigger Set output 01_H Trigger Set 0 Trigger Set TS16_PCIe0 02_H Trigger Set 1 Trigger Set TS16_PCIe1 03_H Trigger Set 2 Trigger Set TS16_PCIe2 04_H Reserved
0	7:5, 31:13	r	**Reserved** Read as 0; should be written with 0.

19.4.2.82 Interrupt/alarm status

ALARM_STAT Offset address: 0000424_H

Interrupt/alarm status Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
RCERRINT	0	rh	**Error reported. The interrupt can be triggered when the ALARM_INTEN.RCERREN is asserted.** Asserted when a reported error condition causes a bit to be set in the Root Error Status register and the associated error message reporting enable bit is set in the Root Error Command register. This bit is set when the RC internally generates an error or when an error message is received by the RC. Because the RC itself generates it, this needs to be propagated up to the system software which would then need to read the error registers to see which error occurred.
SAFCORR	1	rh	**Indicate correctable error. The interrupt can be triggered when the ALARM_INTEN.SAFCORREN is asserted.** The controller asserts this bit when a PCIe protocol correctable error or a Data Path protection correctable error is indicated and the associated mask for that event is not set.
SAFUNCORR	2	rh	**Indicate uncorrectable error. The interrupt can be triggered when the ALARM_INTEN.SAFUNCORR is asserted.** The controller asserts this bit when any of the internal safety mechanisms reports an uncorrectable error and the associated mask for that event is not set.
SYSERR	3	rh	**System error detected. The interrupt can be triggered when the ALARM_INTEN.SYSERREN is asserted.** Assertion of this bit indicates if any device in the hierarchy reports any of the following errors and the associated enable bit is set in the Root Control register: ERR_COR, ERR_FATAL, ERR_NONFATAL. Also asserted when an internal error is detected.
A2SDAT	4	rh	**Error detected during the data phase in the A2S bridge. The interrupt can be triggered when the ALARM_INTEN.A2SDEN is asserted.**
A2SADDR	5	rh	**Error detected during the address phase in the A2S bridge. The interrupt can be triggered when the ALARM_INTEN.A2SAEN is asserted.**
S2ADAT	6	rh	**Error detected during the data phase in the S2A bridge. The interrupt can be triggered when the ALARM_INTEN.S2ADEN is asserted.**
S2AADDR	7	rh	**Error detected during the address phase in the S2A bridge. The interrupt can be triggered when the ALARM_INTEN.S2AAEN is asserted.**
MSTERRM	8	rh	**The controller has entered RASDP error mode at the master interface of AXI.** The controller enters RASDP error mode (if the ERROR_MODE_EN register field =1) upon detection of the first uncorrectable error. During this mode: • Controller sets SLVERRM=1 • Controller sets MSTERRM=1 • Rx TLPs that are forwarded to your application are not guaranteed to be correct; you must discard them.

(table continues...)

(continued)

Field	Bits	Type	Description
SLVERRM	9	rh	**The controller has entered RASDP error mode at the slave interface of AXI.** The controller enters RASDP error mode (if the ERROR_MODE_EN register field =1) upon detection of the first uncorrectable error. During this mode: • Controller sets SLVERRM=1 • Controller sets MSTERRM=1 • Rx TLPs that are forwarded to your application are not guaranteed to be correct; you must discard them.
FPIWRERR	10	rh	**FPI write error status** Status to indicate write returned an error on the FPI interface
0	31:11	r	**Reserved** Read as 0; should be written with 0.

19.4.2.83 Interrupt/alarm clear

ALARM_INTCLR Offset address: 0000428_H

Interrupt/alarm clear Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res								0							
r								r							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0					FPIWRERRCLR	0		S2AACLR	S2ADCLR	A2SACLR	A2SDCLR	SYSERRCLR	SAFUNCORRCLR	SAFTCORRCLR	RCERRCLR
r					w	r		w	w	w	w	w	w	w	w

Field	Bits	Type	Description
RCERRCLR	0	w	**Clear the RCERRINT bit.** 0_B Keep the status 1_B Clear the interrupt status
SAFTCORRCLR	1	w	**Clear the SAFCORR bit.** 0_B Keep the status 1_B Clear the interrupt status
SAFUNCORRCLR	2	w	**Clear the SAFUNCORR bit.** 0_B Keep the status 1_B Clear the interrupt status
SYSERRCLR	3	w	**Clear the SYSERR bit.** 0_B Keep the status 1_B Clear the interrupt status

(table continues...)

(continued)

Field	Bits	Type	Description
A2SDCLR	4	w	**Clear the A2SDAT bit.** 0_B Keep the status 1_B Clear the interrupt status
A2SACLR	5	w	**Clear the A2SADDR bit.** 0_B Keep the status 1_B Clear the interrupt status
S2ADCLR	6	w	**Clear the S2ADAT bit.** 0_B Keep the status 1_B Clear the interrupt status
S2AACLR	7	w	**Clear the S2AADDR bit.** 0_B Keep the status 1_B Clear the interrupt status
FPIWRERRCLR	10	w	**clear FPIWRERR** clear FPIWRERR 0_B Keep the status 1_B Clear the interrupt status
0	9:8, 30:11	r	**Reserved** Read as 0; should be written with 0.

19.4.2.84 Interrupt/alarm enable

ALARM_INTEN Offset address: $000042C_H$

Interrupt/alarm enable Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res								0							
r								r							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0					FPIWRERREN	0		S2AAEN	S2ADEN	A2SAEN	A2SDEN	SYSERREN	SAFUNCORREN	SAFCORREN	RCERREN
r					rw	r		rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RCERREN	0	rw	**Enable the RCERRINT bit.** 0_B Disable interrupt 1_B Enable interrupt
SAFCORREN	1	rw	**Enable the SAFCORR bit.** 0_B Disable interrupt 1_B Enable interrupt

(table continues...)

(continued)

Field	Bits	Type	Description
SAFUNCORREN	2	rw	**Enable the SAFUNCORR bit.** 0_B Disable interrupt 1_B Enable interrupt
SYSERREN	3	rw	**Enable the SYSERR bit.** 0_B Disable interrupt 1_B Enable interrupt
A2SDEN	4	rw	**Enable the A2SDAT bit to generate alarm.** 0_B Disable interrupt 1_B Enable interrupt
A2SAEN	5	rw	**Enable the A2SADDR bit to generate alarm.** 0_B Disable interrupt 1_B Enable interrupt
S2ADEN	6	rw	**Enable the S2ADAT bit to generate alarm.** 0_B Disable interrupt 1_B Enable interrupt
S2AAEN	7	rw	**Enable the S2AADDR bit to generate alarm.** 0_B Disable interrupt 1_B Enable interrupt
FPIWRERREN	10	rw	**Enable FPIWRERR bit to generate alarm.** Enable FPIWRERR bit to generate alarm 0_B Disable 1_B Enable
0	9:8, 30:11	r	**Reserved** Read as 0; should be written with 0.

19.4.2.85 iATU Region i Control 1 Register Outbound

This register controls the iATU outbound region access based on the optional iATU outbound features enabled using iATU Region Control 2 Register.

IATU_REGION_CTRL_1_OFF_OUTBOUND_i (i=0-7) Offset address: $0003000_H+i*200_H$

iATU Region i Control 1 Register Outbound Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0									CTRL_1_FUNC_NUM			0			
r									rw			r			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		Res	0		ATTR		TD	TC			TYPE				
r		r	r		rw		rw	rw			rw				

Field	Bits	Type	Description
TYPE	4:0	rw	**When the address of an outbound TLP is matched to this region, then the TYPE field of the TLP is changed to the value in this register** Note: This register field is sticky.
TC	7:5	rw	**When the address of an outbound TLP is matched to this region, then the TC field of the TLP is changed to the value in this register** Note: This register field is sticky.
TD	8	rw	**This is a reserved field** Do not use. Note: This register field is sticky.
ATTR	10:9	rw	**When the address of an outbound TLP is matched to this region, then the ATTR field of the TLP is changed to the value in this register** Note: This register field is sticky.
CTRL_1_FUNC_NUM	22:20	rw	**Function Number** The value in this register must be 0x0 as only one function is supported by this controller. . When the address of an outbound TLP is matched to this region and the FUNC_BYPASS field in the "iATU Region Control 2 Register" is '0', then the function number used in generating the function part of the requester ID (RID) field of the TLP is taken from this 5-bit register. This field is swapped before AXI decomposition occurs so that the correct "Max_Read_Request_Size" and "Max_Payload_Size" values are used. Note: This register field is sticky
0	12:11, 19:14, 31:23	r	**Reserved** Read as all 0's; should be written with all 0's.

19.4.2.86 iATU Region i Control 2 Register Outbound

Using this register you can enable/disable the outbound iATU optional features.

IATU_REGION_CTRL_2_OFF_OUTBOUND_i (i=0-7) Offset address: $0003004_H+i*200_H$

iATU Region i Control 2 Register Outbound Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
REGION_EN	0	INVERT_MODE	Res	DMA_BYPASS	0			HEADER_SUBSTITUTE_EN	INHIBIT_PAYLOAD	Res		FUNC_BYPASS	0		TAG_SUBSTITUTE_EN
rw	r	rw	r	rw	r			rw	rw	r		rw	r		rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAG								MSG_CODE							
rw								rw							

Field	Bits	Type	Description
MSG_CODE	7:0	rw	**MSG TLPs (Message Code)** When the address of an outbound TLP is matched to this region, and the translated TLP TYPE field is Msg or MsgD; then the message field of the TLP is changed to the value in this register. Note: This register field is sticky.
TAG	15:8	rw	**TAG** The substituted TAG field (byte 6) in the outgoing TLP header when TAG_SUBSTITUTE_EN is set. Note: This register field is sticky.
TAG_SUBSTITUTE_EN	16	rw	**TAG Substitute Enable** When enabled and region address is matched, the iATU substitutes the TAG field of the outbound TLP header with the contents of the TAG field in this register. Note: This register field is sticky. 0_B **DISABLE**: Disable 1_B **ENABLE**: Enable
FUNC_BYPASS	19	rw	**Function Number Translation Bypass** In this mode, the function number of the translated TLP is taken from your application transmit interface and not from the CTRL_1_FUNC_NUM field of the "iATU Region Control 1 Register" Note: This register field is sticky. 0_B **DISABLE**: Disable 1_B **ENABLE**: Enable
INHIBIT_PAYLOAD	22	rw	**Inhibit TLP Payload Data for TLP's in Matched Region; assign iATU region to be TLP without data** When enabled and region address is matched, the iATU marks all TLPs as having no payload data by forcing the TLP header Fmt[1] bit =0b. Encoding are define as above. Note: This register field is sticky. 0_B **SUPPORT_DATA_TLP**: Fmt[1] =0/1 so that TLPs with or without data can be sent 1_B **NO_SUPPORT_FOR_DATA_TLP**: Fmt[1] =0 so that only TLP type without data is sent For example, a Msg instead of MsgD will be sent.
HEADER_SUBSTITUTE_EN	23	rw	**Header Substitute Enable** When enabled and region address is matched, the iATU fully substitutes bytes 8-11 (for 3 DWORD header) or bytes 12-15 (for 4 DWORD header) of the outbound TLP header with the contents of the LWR_TARGET_RW field in IATU_LWR_TARGET_ADDR_OFF_OUTBOUND_i. Encodings are as above. Note: This register field is sticky. 0_B **FORM_ADDR**: LWR_TARGET_RW in the iATU_LWR_TARGET_ADDR_OFF_OUTBOUND_i register forms the new address of the translated region 1_B **FILL_BYTES**: LWR_TARGET_RW in the iATU_LWR_TARGET_ADDR_OFF_OUTBOUND_i register is used to fill bytes 8-to-11 (for 3 DWORD header) or bytes 12-to-15 (for 4 DWORD header) of the translated TLP header

(table continues...)

(continued)

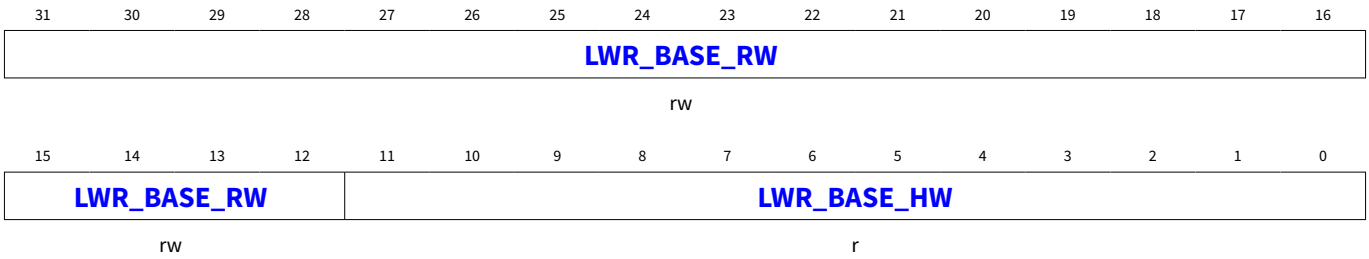
Field	Bits	Type	Description
DMA_BYPASS	27	rw	**DMA Bypass Mode** Allows request TLPs which are initiated by the DMA controller to pass through the iATU untranslated. Note: This register field is sticky. 0_B **DISABLE**: Disable 1_B **ENABLE**: Enable
INVERT_MODE	29	rw	**Invert Mode** When set the address matching region is inverted. Therefore, an address match occurs when the untranslated address is in the region outside the defined range (Base Address to Limit Address). Note: This register field is sticky. 0_B **DISABLE**: Disable 1_B **ENABLE**: Enable
REGION_EN	31	rw	**Region Enable** This bit must be set to '1' for address translation to take place. Note: This register field is sticky.
0	18:17, 26:24, 30	r	**Reserved** Read as all 0's; should be written with all 0's.

19.4.2.87 iATU i Lower Base Address Register Outbound

The CX_ATU_MIN_REGION_SIZE configuration parameter (Value Range: 4 kB, 8 kB, 16 kB, 32 kB, 64 kB defaults to 64 kB) specifies the minimum size of an address translation region. For example, if set to 64 kB; the lower 16 bits of the Base, Limit and Target registers are zero and all address regions are aligned on 64 kB boundaries. More precisely, the lower log2(CX_ATU_MIN_REGION_SIZE) bits are zero.

IATU_LWR_BASE_ADDR_OFF_OUTBOUND_i (i=0-7) Offset address: $0003008_H+i*200_H$

iATU i Lower Base Address Register Outbound Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
LWR_BASE_HW	11:0	r	**Forms bits of the start address of the address region to be translated** The start address must be aligned to a 4 kB boundary, so these bits are always 0. A write to this location is ignored by the PCIe controller.
LWR_BASE_RW	31:12	rw	**Forms bits of the start address of the address region to be translated** Note: This register field is sticky.

19.4.2.88 iATU i Upper Base Address Register Outbound

This register holds the upper 32-bits of the start (and end) address of the address region to be translated.

IATU_UPPER_BASE_ADDR_OFF_OUTBOUND_i (i=0-7) Offset address: $000300C_H+i*200_H$

iATU i Upper Base Address Register Outbound Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UPPER_BASE_RW															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UPPER_BASE_RW															
rw															

Field	Bits	Type	Description
UPPER_BASE_RW	31:0	rw	**Forms bits [63:32] of the start (and end) address of the address region to be translated** In systems with a 32-bit address space, this register is not used and therefore writing to this register has no effect. Note: This register field is sticky.

19.4.2.89 iATU i Limit Address Register Outbound

This register holds the end address of the address region to be translated.

IATU_LIMIT_ADDR_OFF_OUTBOUND_i (i=0-7) Offset address: $0003010_H+i*200_H$

iATU i Limit Address Register Outbound Kernel Reset value: $0000\ 0FFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LIMIT_ADDR_RW															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LIMIT_ADDR_RW				LIMIT_ADDR_HW								CBUF_INCR			
rw				r								r			

Field	Bits	Type	Description
CBUF_INCR	3:0	r	**Circular Buffer** Note: This register field is sticky.
LIMIT_ADDR_HW	11:4	r	**Forms lower bits of the end address of the address region to be translated** The end address must be aligned to a 4 kB boundary, so these bits are always all ones. A write to this location is ignored by the PCIe controller. Note: This register field is sticky.
LIMIT_ADDR_RW	31:12	rw	**Forms upper bits of the end address of the address region to be translated** Note: This register field is sticky.

19.4.2.90 iATU i Lower Target Address Register Outbound

This register holds the Lower Target part of the new address of the translated region, or the outbound TLP header information, depending on the setting of HEADER_SUBSTITUTE_EN field of IATU_REGION_CTRL_2_VIEWPORT_OFF_OUTBOUND_i register.

IATU_LWR_TARGET_ADDR_OFF_OUTBOUND_i (i=0-7) Offset address: $0003014_H+i*200_H$

iATU i Lower Target Address Register Outbound Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LWR_TARGET_RW_OUTBOUND															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LWR_TARGET_RW_OUTBOUND															
rw															

Field	Bits	Type	Description
LWR_TARGET_RW_OUTBOUND	31:0	rw	**Containing the Lower Target part of the new address of the translated region, or the outbound TLP header information** When HEADER_SUBSTITUTE_EN is 0, for lower target address, the start address must be aligned to a 4 kB boundary, so the lower bits of the start address of the new address of the translated region (bits 11:0) are always '0'. When HEADER_SUBSTITUTE_EN in IATU_REGION_CTRL_2_OFF_OUTBOUND_i is '1': - this field forms bytes 8-11 (for 3 dword header) or bytes 12-15 (for 4 dword header) of the outbound TLP header. Note: This register field is sticky.

19.4.2.91 iATU i Upper Target Address Register Outbound

This register holds the upper 32 bits of the start address (Upper Target part) of the new address of the translated region.

IATU_UPPER_TARGET_ADDR_OFF_OUTBOUND_i (i=0-7) Offset address: $0003018_H+i*200_H$

iATU i Upper Target Address Register Outbound Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UPPER_TARGET_RW															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UPPER_TARGET_RW															
rw															

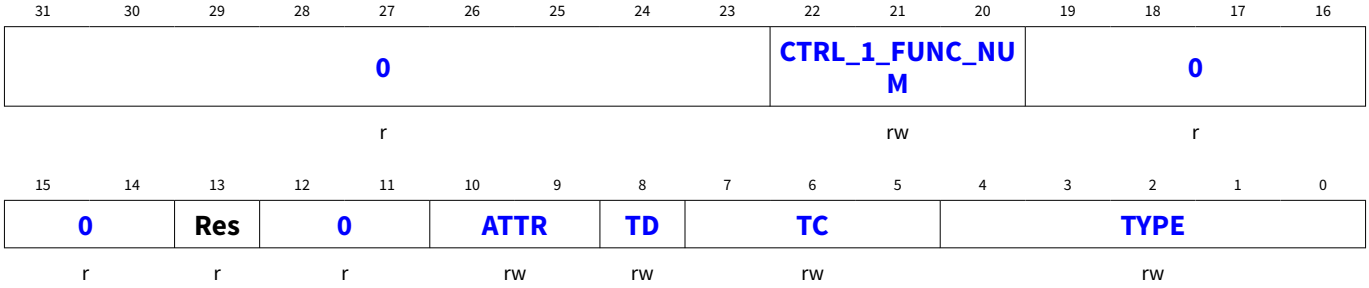
Field	Bits	Type	Description
UPPER_TARGET_RW	31:0	rw	**Forms bits [63:32] of the start address (Upper Target part) of the new address of the translated region** Note: This register field is sticky.

19.4.2.92 iATU Region i Control 1 Register Inbound

This register controls the iATU inbound region access based on the optional iATU inbound features enabled using iATU Region Control 2 Register.

IATU_REGION_CTRL_1_OFF_INBOUND_i (i=0-7) Offset address: $0003100_H+i*200_H$

iATU Region i Control 1 Register Inbound Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
TYPE	4:0	rw	**When the TYPE field of an inbound TLP is matched to this value, then address translation proceeds (when all other enabled field-matches are successful)** Note: This register field is sticky.
TC	7:5	rw	**When the TC field of an inbound TLP is matched to this value, then address translation proceeds (when all other enabled field-matches are successful)** This check is only performed if the "TC Match Enable" bit of the "iATU Region Control 2 Register" is set. Note: This register field is sticky.
TD	8	rw	**When the TD field of an inbound TLP is matched to this value, then address translation proceeds (when all other enabled field-matches are successful)** This check is only performed if the "TD Match Enable" bit of the "iATU Region Control 2 Register" is set. Note: This register field is sticky.
ATTR	10:9	rw	**When the ATTR field of an inbound TLP is matched to this value, then address translation proceeds (when all other enabled field-matches are successful)** This check is only performed if the "ATTR Match Enable" bit of the "iATU Region Control 2 Register" is set. Note: This register field is sticky.
CTRL_1_FUNC_NUM	22:20	rw	**Function Number** - MEM-I/O: When the Address and BAR matching logic in the controller indicate that a MEM-I/O transaction matches a BAR in the function corresponding to this value, then address translation proceeds. This check is only performed if the "Function Number Match Enable" bit of the "iATU Region Control 2 Register" is set. - CFG0/CFG1: When the destination function number as specified in the routing ID of the TLP header matches the function, then address translation proceeds. This check is only performed if the "Function Number Match Enable" bit of the "iATU Region Control 2 Register" is set. Note: This register field is sticky.

(table continues...)

(continued)

Field	Bits	Type	Description
0	12:11, 19:14, 31:23	r	**Reserved** Read as all 0's; should be written with all 0's.

19.4.2.93 iATU Region i Control 2 Register Inbound

Using this register you can enable/disable the inbound iATU optional features.

IATU_REGION_CTRL_2_OFF_INBOUND_i (i=0-7) Offset address: $0003104_H+i*200_H$

iATU Region i Control 2 Register Inbound Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
REGION_EN	MATCH_MODE	INVERT_MODE	Res		0	RESPONSE_CODE		SINGLE_ADDR_LOC_TRANS_EN	0	MSG_CODE_MATCH_EN	0	FUNC_NUM_MATCH_EN	0		ATTR_MATCH_EN
rw	rw	rw	r		r	rw		rw	r	rw	r	rw	r		rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TD_MATCH_EN	TC_MATCH_EN	MSG_TYPE_MATCH_MODE	0		BAR_NUM			MSG_CODE							
rw	rw	rw	r		rw			rw							

Field	Bits	Type	Description
MSG_CODE	7:0	rw	**MSG TLPs: (Message Code)** When the TYPE field of an inbound Msg/MsgD TLP is matched to this value, then address translation proceeds (when all other enabled field-matches are successful). This check is only performed if the "Message Code Match Enable" bit of the "iATU Region Control 2 Register" is set. Note: This register field is sticky.

(table continues...)

(continued)

Field	Bits	Type	Description
BAR_NUM	10:8	rw	**BAR Number** When the BAR number of an inbound MEM or IO TLP " that is matched by the normal internal BAR address matching mechanism " is the same as this field, address translation proceeds (when all other enabled field-matches are successful). This check is only performed if the "Match Mode" bit of the "iATU Region Control 2 Register" is set. IO translation would require either 00100b or 00101b in the inbound TLP TYPE; the BAR Number set in the range 000b - 101b and that BAR configured as an IO BAR. Note: This register field is sticky. 000_B **BAR0**: BAR0 001_B **BAR1**: BAR1 010_B **BAR2**: BAR2 011_B **BAR3**: BAR3 100_B **BAR4**: BAR4 101_B **BAR5**: BAR5 110_B **ROM**: ROM 111_B **RSVD**: reserved
MSG_TYPE_MATCH_MODE	13	rw	**Message Type Match Mode** When enabled, and if single address location translate enable is set, then inbound TLPs of type MSG/MSGd which match the type field of the IATU_REGION_CTRL_1_OFF_INBOUND_i register (TYPE[4:3]=2'b10) will be translated. Message type match mode overrides any value of MATCH_MODE field in this register. Usage scenarios for this are translation of VDM or ATS messages when AXI bridge is configured on client interface. Note: This register field is sticky. 0_B **DISABLE**: Disable 1_B **ENABLE**: Enable
TC_MATCH_EN	14	rw	**TC Match Enable** Ensures that a successful TC TLP field comparison match (see TC field of the "iATU Region Control 1 Register") occurs for address translation to proceed. Note: This register field is sticky. 0_B **DISABLE**: TC Match Disable 1_B **ENABLE**: TC Match Enable
TD_MATCH_EN	15	rw	**TD Match Enable** Ensures that a successful TD TLP field comparison match (see TD field of the "iATU Region Control 1 Register") occurs for address translation to proceed. Note: This register field is sticky. 0_B **DISABLE**: TD Match Disable 1_B **ENABLE**: TD Match Enable
ATTR_MATCH_EN	16	rw	**ATTR Match Enable** Ensures that a successful ATTR TLP field comparison match (see ATTR field of the "iATU Region Control 1 Register") occurs for address translation to proceed. Note: This register field is sticky. 0_B **DISABLE**: ATTR Match Disable 1_B **ENABLE**: ATTR Match Enable

(table continues...)

(continued)

Field	Bits	Type	Description
FUNC_NUM_MATCH_EN	19	rw	**Function Number Match Enable** Ensures that a successful Function Number TLP field comparison match (see Function Number field of the "iATU Region Control 1 Register") occurs (in MEM-I/O and CFG0/CFG1 transactions) for address translation to proceed. Note: This register field is sticky. 0_B **DISABLE**: Function Number Match Disable 1_B **ENABLE**: Function Number Match Enable
MSG_CODE_MATCH_EN	21	rw	**Message Code Match Enable (Msg TLPS)** Ensures that a successful message Code TLP field comparison match (see Message Code field of the "iATU Region Control 2 Register") occurs (in MSG transactions) for address translation to proceed. Note: This register field is sticky. 0_B **DISABLE**: Virtual Function Number Match Disable 1_B **ENABLE**: Message Code Match Enable (for Msg TLPS) or ST Match Enable (for Mem TLPs)
SINGLE_ADDR_LOC_TRANS_EN	23	rw	**Single Address Location Translate Enable** When enabled, Rx TLPs can be translated to a single address location as determined by the target address register of the iATU region. The main usage scenario is translation of Messages to MWr TLPs when the AXI bridge is enabled. Note: This register field is sticky. 0_B **DISABLE**: Disable 1_B **ENABLE**: Enable
RESPONSE_CODE	25:24	rw	**Response Code** Defines the type of response to give for accesses matching this region. This overrides the normal RADM filter response. Note that this feature is not available for any region where Single Address Location Translate is enabled. Note: This register field is sticky. 00_B **NORMAL_RADM**: Normal RADM filter response is used 01_B **UNSUP_REQ**: Unsupported request (UR) 10_B **COMPL_ABORT**: Completer abort (CA) 11_B **NOT_USED**: Not used / undefined / reserved
INVERT_MODE	29	rw	**Invert Mode Enable** When set the address matching region is inverted. Therefore, an address match occurs when the untranslated address is in the region outside the defined range (Base Address to Limit Address). When set all regions of that type must use address match mode. Note: This register field is sticky. 0_B **DISABLE**: Invert Mode Disable 1_B **ENABLE**: Invert Mode Enable

(table continues...)

(continued)

Field	Bits	Type	Description
MATCH_MODE	30	rw	**Match Mode** Determines Inbound matching mode for TLPs. The mode depends on the type of TLP that is received as follows: For MEM-I/O TLPs, this field is interpreted as follows: • 0: Address Match Mode. The iATU operates using addresses as in the outbound direction. The Region Base and Limit Registers must be setup. • 1: BAR Match Mode. BAR matching is used. The "BAR Number" field is relevant. Not used for RC. For CFG0 TLPs, this field is interpreted as follows: • 0: Routing ID Match Mode. The iATU interprets the Routing ID (Bytes 8 to 11 of TLP header) as an address. This corresponds to the upper 16 bits of the address in MEM-I/O transactions. The Routing ID of the TLP must be within the base and limit of the iATU region for matching to proceed • 1: Accept Mode. The iATU accepts all CFG0 transactions as address matches. The routing ID in the CFG0 TLP is ignored. This is useful as all received CFG0 TLPs should be processed regardless of the Bus number For MSG/MSGD TLPs, this field is interpreted as follows: • 0: Address Match Mode. The iATU treats the third dword and fourth dword of the inbound MSG/MSGD TLP as an address and it is matched against the Region Base and Limit Registers • 1: Vendor ID Match Mode. The iATU ignores the Routing ID (Bus, Device, Function) in bits [31:16] of the third dword of the TLP header, but matches against the Vendor ID in bits [15:0] of the third dword of the TLP header. Bits [15:0] of the Region Upper Base register should be programmed with the required Vendor ID. The lower Base and Limit Register should be programmed to translate TLPs based on vendor specific information in the fourth dword of the TLP header If SINGLE_ADDRESS_LOCATION_TRANSLATE_EN = 1 AND MSG_TYPE_MATCH_MODE =1, then Match Mode is ignored. Note: This register field is sticky. 0_B **ZERO**: The interpretation is dependent on TLP type, that is, MEM/IO, CFG0, or MSG/MSGD TLPs 1_B **ONE**: The interpretation is dependent on TLP type, that is, MEM/IO, CFG0, or MSG/MSGD TLPs
REGION_EN	31	rw	**Region Enable** This bit must be set to '1' for address translation to take place. Note: This register field is sticky. 0_B **DISABLE**: Disable 1_B **ENABLE**: Enable

(table continues...)

(continued)

Field	Bits	Type	Description
0	12:11, 18:17, 20, 22, 26	r	**Reserved** Read as all 0's; should be written with all 0's.

19.4.2.94 iATU i Lower Base Address Register Inbound

The CX_ATU_MIN_REGION_SIZE configuration parameter (Value Range: 4 kB, 8 kB, 16 kB, 32 kB, 64 kB defaults to 64 kB) specifies the minimum size of an address translation region. For example, if set to 64 kB; the lower 16 bits of the Base, Limit and Target registers are zero and all address regions are aligned on 64 kB boundaries. More precisely, the lower log2(CX_ATU_MIN_REGION_SIZE) bits are zero.

IATU_LWR_BASE_ADDR_OFF_INBOUND_i (i=0-7) Offset address: $0003108_H+i*200_H$

iATU i Lower Base Address Register Inbound Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LWR_BASE_RW															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LWR_BASE_RW				LWR_BASE_HW											
rw				r											

Field	Bits	Type	Description
LWR_BASE_HW	11:0	r	**Forms bits of the start address of the address region to be translated** The start address must be aligned to a 4 kB boundary, so these bits are always 0. A write to this location is ignored by the PCIe controller.
LWR_BASE_RW	31:12	rw	**Forms bits of the start address of the address region to be translated** Note: This register field is sticky.

19.4.2.95 iATU i Upper Base Address Register Inbound

This register holds the upper 32 bits of the start (and end) address of the address region to be translated.

IATU_UPPER_BASE_ADDR_OFF_INBOUND_i (i=0-7) Offset address: $000310C_H+i*200_H$

iATU i Upper Base Address Register Inbound Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UPPER_BASE_RW															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UPPER_BASE_RW															
rw															

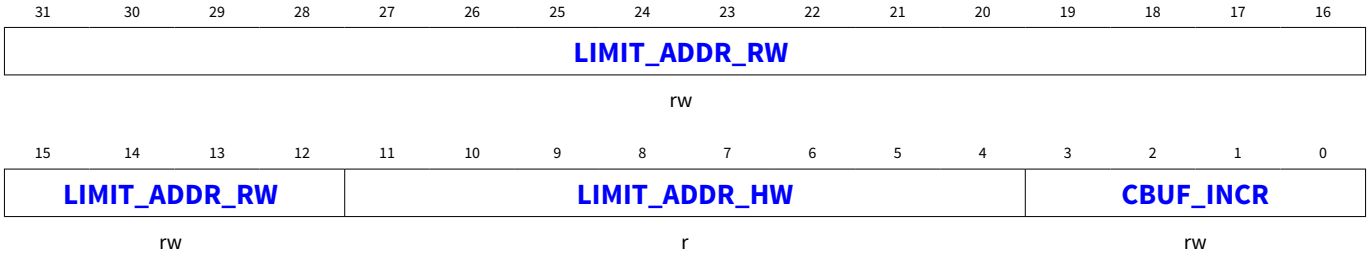
Field	Bits	Type	Description
UPPER_BASE_RW	31:0	rw	**Forms bits [63:32] of the start (and end) address of the address region to be translated** Note: This register field is sticky.

19.4.2.96 iATU i Limit Address Register Inbound

This register holds the end address of the address region to be translated.

IATU_LIMIT_ADDR_OFF_INBOUND_i (i=0-7) Offset address: $0003110_H+i*200_H$

iATU i Limit Address Register Inbound Kernel Reset value: $0000\ 0FF0_H$



Field	Bits	Type	Description
CBUF_INCR	3:0	rw	**Circular Buffer Increment** This field is R/W and forms the upper bits of the Circular Buffer Increment size (CBUF_INCR) field for Single Location Address translation. The increment value (in bytes) is decoded as follows: • 0000b: 0 (Default; legacy Single Address Location mode) • 0001b: 4 • 0010b: 8 • 0011b: 16 • 0100b: 32 • 0101b: 64 • 0110b: 128 • 0111b: 256 • 1000b: 512 • 1001b: 1024 • 1010b: 2048 • 1011b: 4096 • 1100b: 8192 • 1101b: rsvd • 1110b: rsvd • 1111b: rsvd Note: A write to any bit in the CBUF_INCR field resets the circular buffer pointer. This field must be written to AFTER the target and limit registers have been updated. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: R (sticky) Note: This register field is sticky.

(table continues...)

(continued)

Field	Bits	Type	Description
LIMIT_ADDR_HW	11:4	r	**Forms lower bits of the end address of the address region to be translated** The end address must be aligned to a 4 kB boundary, so these bits are always all ones. A write to this location is ignored by the PCIe controller. Note: This register field is sticky.
LIMIT_ADDR_RW	31:12	rw	**Forms upper bits of the end address of the address region to be translated** Note: This register field is sticky.

19.4.2.97 iATU i Lower Target Address Register Inbound

This register holds the Lower Target part of the new address of the translated region.

IATU_LWR_TARGET_ADDR_OFF_INBOUND_i (i=0-7) Offset address: $0003114_H+i*200_H$

iATU i Lower Target Address Register Inbound Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LWR_TARGET_RW															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LWR_TARGET_RW				LWR_TARGET_HW											
rw				r											

Field	Bits	Type	Description
LWR_TARGET_HW	11:0	r	**Forms the LSB's of the Lower Target part of the new address of the translated region** The start address must be aligned to a 4 kB boundary (in address match mode); and to the Bar size boundary (in BAR match mode) so that these bits are always '0'. If the BAR is smaller than the iATU region size, then the iATU target address must align to the iATU region size; otherwise it must align to the BAR size. A write to this location is ignored by the PCIe controller. .
LWR_TARGET_RW	31:12	rw	**Forms MSB's of the Lower Target part of the new address of the translated region** These bits are always '0'. Note: This register field is sticky.

19.4.2.98 Device ID and Vendor ID Register

This register holds the device ID and vendor ID.

TYPE1_DEV_ID_VEND_ID_REG Offset address: 0004000_H

Device ID and Vendor ID Register Kernel Reset value: $ABCD\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DEVICE_ID															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VENDOR_ID															
rw															

Field	Bits	Type	Description
VENDOR_ID	15:0	rw	**Vendor ID** The Vendor ID register identifies the manufacturer of the Function. Valid vendor identifiers are allocated by the PCI-SIG to ensure uniqueness. It is not permitted to keep this register populated with a value of FFFFh, which is an invalid value for Vendor ID. Note: The access attributes of this field are as follows: - Wire: No access. - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky.
DEVICE_ID	31:16	rw	**Device ID** The Device ID register identifies the particular Function. This identifier is allocated by the vendor. Note: The access attributes of this field are as follows: - Wire: No access. - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky.

19.4.2.99 Status and Command Register

This register provides the status and controls the behavior of a function.

TYPE1_STATUS_COMMAND_REG Offset address: 0004004_H

Status and Command Register Kernel Reset value: $0010\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DETECTED_PARITY_ERROR	SIGNALED_SYS_ERROR	RCVD_MASTER_ABORT	RCVD_TARGET_ABORT	SIGNALED_TARGET_ABORT	DEV_SEL_TIMING		MASTER_DPE	FAST_B2B_CAP	RSVDP_22	FAST_66MHZ_CAP	CAP_LIST	INT_STATUS	RSVDP_17		0
rw1ch	rw1ch	rw1ch	rw1ch	rw1ch	r		rw1ch	r	r	r	r	rh	r		r

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERV					INT_EN	RSVDP_9	SERREN	IDSEL	PERREN	VGAPS	MWI_EN	SCO	BME	MSE	IO_EN
r					rw	r	rw	r	rw	r	r	r	rw	rwh	rwh

Field	Bits	Type	Description
IO_EN	0	rwh	**IO Space Enable** This bit controls a Function's response to I/O Space accesses received on its primary side. You cannot write to this register if your configuration has no IO BARs; that is, the internal signal has_io_bar =0. Note: The access attributes of this field are as follows: - Wire: No access. - Dbi: !has_io_bar ? RO : RW 0_B **CLEAR**: All received I/O accesses are caused to be handled as Unsupported Requests 1_B **SET**: The Function is enabled to decode the address and further process I/O Space accesses
MSE	1	rwh	**Memory Space Enable** This bit controls a Function's response to Memory Space accesses received on its primary side. You cannot write to this register if your configuration has no MEM BARs; that is, the internal signal has_mem_bar =0. Note: The access attributes of this field are as follows: - Wire: No access. - Dbi: !has_mem_bar ? RO : RW 0_B **CLEAR**: All received Memory Space accesses are caused to be handled as Unsupported Requests 1_B **SET**: The Function is enabled to decode the address and further process Memory Space accesses
BME	2	rw	**Bus Master Enable** This bit controls forwarding of Memory or I/O requests by a port in the Upstream direction. When this bit is 0b, Memory and I/O Requests received at a Root Port must be handled as Unsupported Requests (UR) For Non-Posted Requests a Completion with UR completion status must be returned. This bit does not affect forwarding of Completions in either the Upstream or Downstream direction. The forwarding of Requests other than Memory or I/O Requests is not controlled by this bit.
SCO	3	r	**Special Cycle Enable** This bit was originally described in the PCI Local Bus Specification. Its functionality does not apply to PCI Express. The controller hardwires this bit to 0b.
MWI_EN	4	r	**Memory Write and Invalidate** This bit was originally described in the PCI Local Bus Specification and the PCI-to-PCI Bridge Architecture Specification. Its functionality does not apply to PCI Express. The controller hardwires this bit to 0b. For PCI Express to PCI/PCI-X Bridges, refer to the PCI Express to PCI/PCI-X Bridge Specification for requirements for this register.
VGAPS	5	r	**VGA Palette Snoop** This bit was originally described in the PCI Local Bus Specification and the PCI-to-PCI Bridge Architecture Specification. Its functionality does not apply to PCI Express. The controller hardwires this bit to 0b.

(table continues...)

(continued)

Field	Bits	Type	Description
PERREN	6	rw	**Parity Error Response** This bit controls the logging of poisoned TLPs in the Master Data Parity Error bit in the Status register. 0_B **DISABLE**: Parity Error Response disable 1_B **ENABLE**: Parity Error Response enable
IDSEL	7	r	**IDSEL Stepping/Wait Cycle Control** This bit was originally described in the PCI Local Bus Specification. Its functionality does not apply to PCI Express. The controller hardwires this bit to 0b.
SERREN	8	rw	**SERR# Enable** Note: - The errors are reported if enabled either through this bit or through the PCI Express specific bits in the Device Control register. - In addition, this bit controls transmission by the primary interface of ERR_NONFATAL and ERR_FATAL error Messages forwarded from the secondary interface. This bit does not affect the transmission of forwarded ERR_COR messages. 1_B **SET**: This bit enables reporting upstream of Non-fatal and Fatal errors detected by the Function
RSVDP_9	9	r	**Reserved for future use**
INT_EN	10	rw	**Interrupt Disable** Controls the ability of a Function to generate INTx emulation interrupts. Note: • Any INTx emulation interrupts already asserted by the Function must be deasserted when this bit is set. INTx interrupts use virtual wires that must, if asserted, be deasserted using the appropriate Deassert_INTx message(s) when this bit is set • Only the INTx virtual wire interrupt(s) associated with the Function(s) for which this bit is set are affected • For Functions that generate INTx interrupts on their own behalf, this bit is required. This bit has no effect on interrupts forwarded from the secondary side. 1_B **SET**: When set, Functions are prevented from asserting INTx interrupts
RESERV	15:11	r	**Reserved** 00_H **RESERVED**: Reserved **others,** Reserved
RSVDP_17	18:17	r	**Reserved for future use**
INT_STATUS	19	rh	**Interrupt Status** INTx emulation interrupts forwarded by Functions from the secondary side are not reflected in this bit. Setting the Interrupt Disable bit has no effect on the state of this bit. For Functions that do not generate INTx interrupts, the controller hardwires this bit to 0b. 1_B **SET**: Indicates that an INTx emulation interrupt is pending internally in the Function

(table continues...)

(continued)

Field	Bits	Type	Description
CAP_LIST	20	r	**Capabilities List** Indicates the presence of an Extended Capability list item. Since all PCI Express device Functions are required to implement the PCI Express Capability structure, the controller hardwires this bit to 1b.
FAST_66MHZ_CAP	21	r	**66 MHz Capable** This bit was originally described in the PCI Local Bus Specification. Its functionality does not apply to PCI Express. The controller hardwires this bit to 0b.
RSVDP_22	22	r	**Reserved for future use**
FAST_B2B_CAP	23	r	**Fast Back-to-Back Transactions Capable** This bit was originally described in the PCI Local Bus Specification. Its functionality does not apply to PCI Express. The controller hardwires this bit to 0b.
MASTER_DPE	24	rw1ch	**Master Data Parity Error** This bit is set by a Function if the Parity Error Response bit in the Command register is 1b and either of the following two conditions occurs: • Port receives a Poisoned Completion going downstream • Port transmits a Poisoned Request upstream If the Parity Error Response bit is 0b, this bit is never set.
DEV_SEL_TIMING	26:25	r	**DEVSEL Timing** This field was originally described in the PCI Local Bus Specification. Its functionality does not apply to PCI Express. The controller hardwires it to 00b.
SIGNALED_TARGET_ABORT	27	rw1ch	**Signaled Target Abort** This bit is set when a Function completes a Posted or Non-Posted Request as a Completer Abort error. This applies to a Function when the Completer Abort was generated by its primary side. Software write with 1 clears this bit-field.
RCVD_TARGET_ABORT	28	rw1ch	**Received Target Abort** This bit is set when a Requester receives a Completion with Completer Abort Completion status. The bit is set when the Completer Abort is received by a Function's primary side. Software write with 1 clears this bit-field.
RCVD_MASTER_ABORT	29	rw1ch	**Received Master Abort** This bit is set when a Requester receives a Completion with Unsupported Request Completion status. The bit is set when the Unsupported Request is received by a Function's primary side. Software write with 1 clears this bit-field.
SIGNALED_SYS_ERROR	30	rw1ch	**Signaled System Error** Software write with 1 clears this bit-field.. 1_B **SET**: This bit is set when a Function sends an ERR_FATAL or ERR_NONFATAL Message, and the SERR# Enable bit in the Command register is 1b

(table continues...)

(continued)

Field	Bits	Type	Description
DETECTED_PARITY_ERROR	31	rw1ch	**Detected Parity Error** The bit is set when the Poisoned TLP is received by a Function's primary side. Software write with 1 clears this bit-field. 1_B **SET**: This bit is set by a Function whenever it receives a Poisoned TLP, regardless of the state the Parity Error Response bit in the Command register
0	16	r	**Reserved** Read as 0; should be written with 0.

19.4.2.100 Class Code and Revision ID Register

This register specifies the class code and revision ID of a function.

TYPE1_CLASS_CODE_REV_ID_REG Offset address: 0004008_H

Class Code and Revision ID Register Kernel Reset value: $0000\ 0001_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BASE_CLASS_CODE								SUBCLASS_CODE							
rw								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PROGRAM_INTERFACE								REVISION_ID							
rw								rw							

Field	Bits	Type	Description
REVISION_ID	7:0	rw	**Revision ID** The value of this field specifies a Function specific revision identifier. The value is chosen by the vendor. Zero is an acceptable value. The Revision ID should be viewed as a vendor defined extension to the Device ID. Note: The access attributes of this field are as follows: - Wire: No access. - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky.
PROGRAM_INTERFACE	15:8	rw	**Programming Interface** This field identifies a specific register level programming interface (if any) so that device independent software can interact with the Function. Encodings for interface are provided in the PCI Code and ID Assignment Specification. All unspecified encodings are reserved. Note: The access attributes of this field are as follows: - Wire: No access. - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky.
SUBCLASS_CODE	23:16	rw	**Sub-Class Code** Specifies a base class sub-class, which identifies more specifically the operation of the Function. Encodings for sub-class are provided in the PCI Code and ID Assignment Specification. All unspecified encodings are reserved. Note: The access attributes of this field are as follows: - Wire: No access. - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky.

(table continues...)

(continued)

Field	Bits	Type	Description
BASE_CLASS_CODE	31:24	rw	**Base Class Code** A code that broadly classifies the type of operation the Function performs. Encodings for base class, are provided in the PCI Code and ID Assignment Specification. All unspecified encodings are reserved. Note: The access attributes of this field are as follows: - Wire: No access. - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky.

19.4.2.101 BIST, Header Type, Latency Timer, and Cache Line Size Register

This register provides the status and controls BIST. It also holds information regarding the header layout, latency timer, and cache line size.

TYPE1_BIST_HDR_TYPE_LAT_CACHE_LINE_SIZE_REG

Offset address: $000400C_H$

BIST, Header Type, Latency Timer, and Cache Line Size Register

Kernel Reset value: $0001\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BIST								MULTI_FUNC	HEADER_TYPE						
r								r	r						

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LATENCY_MASTER_TIMER								CACHE_LINE_SIZE							
r								rw							

Field	Bits	Type	Description
CACHE_LINE_SIZE	7:0	rw	**Cache Line Size** The Cache Line Size register is programmed by the system firmware or the operating system to system cache line size. However, legacy conventional PCI software may not always be able to program this register correctly especially in the case of Hot-Plug devices. This read-write register is implemented for legacy compatibility purposes but has no effect on any PCI Express device behavior.
LATENCY_MASTER_TIMER	15:8	r	**Latency Timer** This register is also referred to as Primary Latency Timer. The Latency Timer was originally described in the PCI Local Bus Specification and the PCI-to-PCI Bridge Architecture Specification. Its functionality does not apply to PCI Express. The controller hardwires this register to 00h.
HEADER_TYPE	22:16	r	**Header Layout** This field identifies the layout of the second part of the predefined header. The controller uses 000 0001b encoding. The encoding 000 0010b is reserved. This encoding was originally described in the PC Card Standard Electrical Specification and is used in previous versions of the programming model. Careful consideration should be given to any attempt to re-purpose it.

(table continues...)

(continued)

Field	Bits	Type	Description
MULTI_FUNC	23	r	**Multi-Function Device** 0_B **CLEAR**: Software must not probe for Functions other than Function 0 1_B **SET**: Indicates that the device may contain multiple Functions, but not necessarily Software is permitted to probe for Functions other than Function 0.
BIST	31:24	r	**BIST** This register is used for control and status of BIST. Functions that do not support BIST must hardwire the register to 00h. A Function whose BIST is invoked must not prevent normal operation of the PCI Express Link. Bit descriptions: • [31]: BIST Capable. When set, this bit indicates that the Function supports BIST. When Clear, the Function does not support BIST • [30]: Start BIST. If BIST Capable is set, set this bit to invoke BIST. The Function resets the bit when BIST is complete. Software is permitted to fail the device if this bit is not Clear (BIST is not complete) 2 seconds after it had been set. Writing this bit to 0b has no effect. The controller hardwires this bit to 0b if BIST Capable is clear • [29:28]: Reserved • [27:24]: Completion Code. This field encodes the status of the most recent test. A value of 0000b means that the Function has passed its test. Non-zero values mean the Function failed. Function-specific failure codes can be encoded in the non-zero values. This field's value is only meaningful when BIST Capable is set and Start BIST is Clear. This field must be hardwired to 0000b if BIST Capable is clear.

19.4.2.102 BAR0 Register i

System software must build a consistent address map before booting the machine to an operating system. This means it has to determine how much memory is in the system, and how much address space the Functions in the system require. After determining this information, system software can map the Functions into reasonable locations and proceed with system boot. In order to do this mapping in a device-independent manner, the base registers for this mapping are placed in the predefined header portion of Configuration Space. It is strongly recommended that power-up firmware/software also support the optional Enhanced Configuration Access Mechanism (ECAM).

BARi_REG (i=0-1) Offset address: 0004010_H+i*4

BAR0 Register i Kernel Reset value: $0000\ 0000_H$

Field	Bits	Type	Description
BAR0_MEM_IO	0	rwh	**BAR0 Memory Space Indicator** This bit is used to determine whether the register maps into Memory or I/O Space. Base Address registers that map to Memory Space must return a 0b. Base Address registers that map to I/O Space must return a 1b. Note: The access attributes of this field are as follows: - Wire: No access. - Dbi: if (BAR_ENABLED == 1) then (if [DBI_RO_WR_EN == 1] then R(Sticky)/W(Sticky) else R(Sticky)) else RO(Sticky) Note: This register field is sticky.
BAR0_TYPE	2:1	rwh	**BAR0 Type** • Memory Space: Base Address registers that map into Memory Space can be 32 bits or 64 bits wide (to support mapping into a 64-bit address space). The encodings defined in Values: apply • IO Space: Bit 1 is reserved and must return 0b on reads. Bits[31:2] are used to map the function into IO space. The encodings defined in Values: do not apply Note: The access attributes of this field are as follows: - Wire: No access. - Dbi: if (BAR_ENABLED == 1) then (if [DBI_RO_WR_EN == 1] then R(Sticky)/W(Sticky) else R(Sticky)) else RO(Sticky) Note: This register field is sticky. 00_B **BAR_32**: Base register is 32 bits wide and can be mapped anywhere in the 32 address bit Memory Space 01_B **RSVD_1**: Reserved 10_B **BAR_64**: Base register is 64 bits wide and can be mapped anywhere in the 64 address bit Memory Space 11_B **RSVD_2**: Reserved
BAR0_PREFETCH	3	rwh	**BAR0 Prefetchable** • Memory Space: Set to one if data is prefetchable. A Function is permitted to mark a range as prefetchable. If there are no side effects on reads, the function returns all bytes on reads regardless of the byte enables, and host bridges can merge processor writes into this range without causing errors. Bit must me clear otherwise • IO Space: Not applicable Note: The access attributes of this field are as follows: - Wire: No access. - Dbi: if (BAR_ENABLED == 1) then (if [DBI_RO_WR_EN == 1] then R(Sticky)/W(Sticky) else R(Sticky)) else RO(Sticky) Note: This register field is sticky.

(table continues...)

(continued)

Field	Bits	Type	Description
BAR0_START	31:4	rwh	**BAR0 Base Address** • Memory Space: Base Address • IO Space: bits[31:2] are used to map the function into IO space/ Base Address. Note: The access attributes of this field are as follows: - Wire: No access. - Dbi: R(Sticky)/W(Sticky) if enabled else R(Sticky) Note: This register field is sticky.

19.4.2.103 Secondary Latency Timer, Subordinate Bus Number, Secondary Bus Number, and Primary Bus Number Register

This register holds information regarding secondary latency timer, subordinate bus number, secondary bus number, and primary bus number.

SEC_LAT_TIMER_SUB_BUS_SEC_BUS_PRI_BUS_REG Offset address: 0004018_H

Secondary Latency Timer, Subordinate Bus Number, Secondary Bus Number, and Primary Bus Number Register

Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SEC_LAT_TIMER								SUB_BUS							
r								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SEC_BUS								PRIM_BUS							
rw								rw							

Field	Bits	Type	Description
PRIM_BUS	7:0	rw	**Primary Bus Number** This register is not used by PCI Express Functions. It is implemented for compatibility with legacy software.
SEC_BUS	15:8	rw	**Secondary Bus Number** The Secondary Bus Number register is used to record the bus number of the PCI bus segment to which the secondary interface of the bridge is connected. Configuration software programs the value in this register. The bridge uses this register to determine when to respond to and convert a Type 1 configuration transaction on the primary interface into a Type 0 transaction on the secondary interface.
SUB_BUS	23:16	rw	**Subordinate Bus Number** The Subordinate Bus Number register is used to record the bus number of the highest numbered PCI bus segment which is behind (or subordinate to) the bridge. Configuration software programs the value in this register. The bridge uses this register in conjunction with the Secondary Bus Number register to determine when to respond to and pass on a Type 1 configuration transaction on the primary interface to the secondary interface.

(table continues...)

(continued)

Field	Bits	Type	Description
SEC_LAT_TIMER	31:24	r	**Secondary Latency Timer** This register does not apply to PCI Express. The controller hardwires it to 00h.

19.4.2.104 Secondary Status, and I/O Limit and Base Register

The I/O Limit and I/O Base registers are optional and define an address range that is used by the bridge to determine when to forward I/O transactions from one interface to the other. If a bridge does not implement an I/O address range, then both the I/O Limit and I/O Base registers must be implemented as read-only registers that return zero when read. If a bridge supports an I/O address range, then these registers must be initialized by configuration software so default states are not specified.

SEC_STAT_IO_LIMIT_IO_BASE_REG Offset address: $000401C_H$

Secondary Status, and I/O Limit and Base Register Reset values see: Table 882

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SEC_STAT_DPE	SEC_STAT_RCVD_SYS_ERR	SEC_STAT_RCVD_MSTR_ABRT	SEC_STAT_RCVD_TRGT_ABRT	SEC_STAT_SIG_TRGT_ABRT	RSVDP_25		SEC_STAT_MDPE	RSVDP_23	SEC_STAT_RESERV						
rw1ch	rw1ch	rw1ch	rw1ch	rw1ch	r		rw1ch	r	r						

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IO_LIMIT				IO_RESERV1			IO_DECODE_BIT8	IO_BASE				IO_RESERV			IO_DECODE
rw				r			rwh	rw				r			rw

Field	Bits	Type	Description
IO_DECODE	0	rw	**I/O Addressing Encode (IO Base Address)** This bit encodes the IO addressing capability of the bridge. Note: • For 16bit I/O address decoding, the bridge must still perform a full 32-bit decode of the I/O address (that is, check that Address[31:16] are 0000h). In this case, the I/O address range supported by the bridge will be to the first 64 KB of I/O Space (0000 0000hto 0000 F FFFh) • For 16bit I/O address decoding, the bridge assumes that the upper 16 address bits, Address[31:16], of the I/O limit address (not implemented in I/O Limit register) are zero • For 32bit I/O address decoding, and the I/O Limit Upper 16 Bits hold the upper 16 bits, corresponding to Address[31:16], of the 32-bit Limit address. In this case, system configuration software is permitted to locate the I/O address range supported by the bridge anywhere in the 4-GB I/O Space • The 4-KB alignment and granularity restrictions still apply when the bridge supports 32-bit I/O addressing Note: The access attributes of this field are as follows: - Wire: No access. - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R 0_B **_16b_SUP_IO_ADDR**: The bridge supports only 16-bit I/O addressing (for ISA compatibility) 1_B **_32b_SUP_IO_ADDR**: The bridge supports 32-bit I/O addressing
IO_RESERV	3:1	r	**Reserved**
IO_BASE	7:4	rw	**I/O Base Address** These bits correspond to the address[15:12] of IO address range. For the purpose of address decoding, the bridge assumes that the lower 12 address bits, address[11:0], of the I/O base address (not implemented in the I/O Base register) are zero.

(table continues...)

(continued)

Field	Bits	Type	Description
IO_DECODE_BIT8	8	rwh	**I/O Addressing Encode (IO Limit Address)** This bit encodes the IO addressing capability of the bridge. Note: • For 16bit I/O address decoding, the bridge must still perform a full 32-bit decode of the I/O address (that is, check that Address[31:16] are 0000h). In this case, the I/O address range supported by the bridge will be to the first 64 KB of I/O Space (0000 0000hto 0000 F FFFh) • For 16bit I/O address decoding, the bridge assumes that the upper 16 address bits, Address[31:16], of the I/O limit address (not implemented in I/O Limit register) are zero • For 32bit I/O address decoding, and the I/O Limit Upper 16 Bits hold the upper 16 bits, corresponding to Address[31:16], of the 32-bit Limit address. In this case, system configuration software is permitted to locate the I/O address range supported by the bridge anywhere in the 4-GB I/O Space • The 4-KB alignment and granularity restrictions still apply when the bridge supports 32-bit I/O addressing Note: The access attributes of this field are as follows: - Wire: No access. - Dbi: R 0_B **_16b_SUP_IO_ADDR**: The bridge supports only 16-bit I/O addressing (for ISA compatibility) 1_B **_32b_SUP_IO_ADDR**: The bridge supports 32-bit I/O addressing
IO_RESERV1	11:9	r	**Reserved**
IO_LIMIT	15:12	rw	**I/O Limit Address** These bits correspond to the address[15:12] of IO address range. For the purpose of address decoding, the bridge assumes that the lower 12 address bits, address[11:0], of the I/O limit address (not implemented in the I/O Limit register) are FFFh. The I/O Limit register can be programmed to a smaller value than the I/O Base register, if there are no I/O addresses on the secondary side of the bridge. In this case, the bridge will not forward any I/O transactions from the primary bus to the secondary and will forward all I/O transactions from the secondary bus to the primary bus.
SEC_STAT_RESERV	22:16	r	**Reserved**
RSVDP_23	23	r	**Reserved for future use**
SEC_STAT_MDPE	24	rw1ch	**Master Data Parity Error** This bit is set by a Function if the Parity Error Response Enable bit in the Bridge Control register is set, and either of the following two conditions occurs: • Port receives a Poisoned Completion coming Upstream • Port transmits a Poisoned Request Downstream If the Parity Error Response Enable bit is clear, this bit is never set
RSVDP_25	26:25	r	**Reserved for future use**

(table continues...)

(continued)

Field	Bits	Type	Description
SEC_STAT_SIG_TRGT_ABRT	27	rw1ch	**Signaled Target Abort** This bit is set when the secondary side of the Function (for Requests completed by the Type 1 header Function itself) completes a Posted or Non-Posted request as a Completer Abort error. Software write with 1 clears this bit-field.
SEC_STAT_RCVD_TRGT_ABRT	28	rw1ch	**Received Target Abort** This bit is set when the secondary side of a Function (for requests initiated by the Type 1 header Function itself) receives a Completion with Completer Abort Completion status. Software write with 1 clears this bit-field.
SEC_STAT_RCVD_MSTR_ABRT	29	rw1ch	**Received Master Abort** This bit is set when the secondary side of a Function (for requests initiated by the Type 1 header Function itself) receives a Completion with Unsupported Request Completion status. Software write with 1 clears this bit-field.
SEC_STAT_RCVD_SYS_ERR	30	rw1ch	**Received System Error** This bit is set when the secondary side of a Function receives an ERR_FATAL or ERR_NONFATAL message. Software write with 1 clears this bit-field.
SEC_STAT_DPE	31	rw1ch	**Detected Parity Error** This bit is set by a Function when a Poisoned TLP is received by its secondary side, regardless of the state the Parity Error Response Enable bit in the Bridge Control register. Software write with 1 clears this bit-field.

Table 882 Reset values of SEC_STAT_IO_LIMIT_IO_BASE_REG

Reset	Reset value	Note
Kernel Reset	0000 0000 0000 0000 0000 000X 0000 0000_B	

19.4.2.105 Memory Limit and Base Register

The Memory Limit and Memory Base registers define a memory mapped address range which is used by the bridge to determine when to forward memory transactions from one interface to the other. If there is no prefetchable memory space, and there is no memory-mapped space on the secondary side of the bridge, then the bridge will not forward any memory transactions from the primary bus to the secondary bus and will forward all memory transactions from the secondary bus to the primary bus.

MEM_LIMIT_MEM_BASE_REG Offset address: 0004020_H

Memory Limit and Base Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MEM_LIMIT												MEM_LIMIT_RESERV			
rw												r			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MEM_BASE												MEM_BASE_RESERV			
rw												r			

Field	Bits	Type	Description
MEM_BASE_RESERV	3:0	r	**Reserved**
MEM_BASE	15:4	rw	**Memory Base Address** These bits correspond to the upper 12 address bits, Address[31:20], of 32-bit addresses. For the purpose of address decoding, the bridge assumes that the lower 20 address bits, Address[19:0], of the memory base address (not implemented in the Memory Base register) are zero.
MEM_LIMIT_RESERV	19:16	r	**Reserved**
MEM_LIMIT	31:20	rw	**Memory Limit Address** These bits correspond to the upper 12 address bits, Address[31:20], of 32-bit addresses. For the purpose of address decoding, the bridge assumes that the lower 20 address bits, Address[19:0], of the memory limit address (not implemented in the Memory Limit register) are F FFFFh. The Memory Limit register must be programmed to a smaller value than the Memory Base register if there is no memory-mapped address space on the secondary side of the bridge.

19.4.2.106 Prefetchable Memory Limit and Base Register

The Prefetchable Memory Limit and Prefetchable Memory Base registers must indicate that 64-bit addresses are supported. The Prefetchable Memory Limit and Prefetchable Memory Base registers are optional. They define a prefetchable memory address range which is used by the bridge to determine when to forward memory transactions from one interface to the other (see the PCI-to-PCI Bridge Architecture Specification for additional details).

PREF_MEM_LIMIT_PREF_MEM_BASE_REG Offset address: 0004024_H

Prefetchable Memory Limit and Base Register Kernel Reset value: $0001\ 0001_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PREF_MEM_LIMIT												PREF_RESERV1			PREF_MEM_LIMIT_DECODE
rwh												r			rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PREF_MEM_BASE												PREF_RESERV			PREF_MEM_DECODE
rwh												r			rw

Field	Bits	Type	Description
PREF_MEM_DECODE	0	rw	**Prefetchable Memory Base Decode** This bit encodes whether or not the bridge supports 64-bit addresses. Note: By default the bit is set to 1'b1 indicating that 64-bit addresses are supported. If the bridge only supports 32-bit prefetchable memory address range, or if there is no prefetchable memory address range, then the configuration parameter MEM_DECODE_64_0 must be changed to 0. Note: The access attributes of this field are as follows: - Wire: No access. - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R Note: This register field is sticky. 0_B **SUP_32b_ADDR**: Indicates that the bridge supports only 32 bit addresses 1_B **SUP_64b_ADDR**: Indicates that the bridge supports 64 bit addresses Prefetchable Base Upper 32 Bits registers holds the rest of the 64-bit prefetchable base address.
PREF_RESERV	3:1	r	**Reserved**
PREF_MEM_BASE	15:4	rwh	**Prefetchable Memory Base Address** • If the Prefetchable Memory Base register indicates support for 32-bit addressing, then the Prefetchable Base Upper 32 Bits register is implemented as a read-only register that returns zero when read • If the Prefetchable Memory Base register indicates support for 64-bit addressing, then the Prefetchable Limit Upper 32 Bits register is implemented as a read/write register which must be initialized by configuration software • If a 64-bit prefetchable memory address range is supported, the Prefetchable Base Upper 32 Bits register specifies the upper 32 bits, corresponding to Address[63:32], of the 64-bit base addresses which specify the prefetchable memory address range
PREF_MEM_LIMIT_DECODE	16	rh	**Prefetchable Memory Limit Decode** This bit encodes whether or not the bridge supports 64-bit addresses. Note: This bit is a copy of the PREF_MEM_DECODE bit and always reflects the current value of that bit. 0_B **SUP_32b_ADDR**: Indicates that the bridge supports only 32 bit addresses 1_B **SUP_64b_ADDR**: Indicates that the bridge supports 64 bit addresses Prefetchable Limit Upper 32 Bits registers holds the rest of the 64-bit prefetchable limit address.
PREF_RESERV1	19:17	r	**Reserved**

(table continues...)

(continued)

Field	Bits	Type	Description
PREF_MEM_LIMIT	31:20	rwh	**Prefetchable Memory Limit Address** • If the Prefetchable Memory Limit register indicates support for 32-bit addressing, then the Prefetchable Limit Upper 32 Bits register is implemented as a read-only register that returns zero when read • If the Prefetchable Memory Limit registers indicate support for 64-bit addressing, then the Prefetchable Limit Upper 32 Bits register is implemented as a read/write register which must be initialized by configuration software • If a 64-bit prefetchable memory address range is supported, the Prefetchable Limit Upper 32 Bits register specifies the upper 32 bits, corresponding to Address[63:32], of the 64-bit limit addresses which specify the prefetchable memory address range

19.4.2.107 Prefetchable Base Upper 32 Bits Register

The Prefetchable Base Upper 32 Bits register is an optional extension to the Prefetchable Memory Base register.

PREF_BASE_UPPER_REG Offset address: 0004028_H

Prefetchable Base Upper 32 Bits Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PREF_MEM_BASE_UPPER															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PREF_MEM_BASE_UPPER															
rw															

Field	Bits	Type	Description
PREF_MEM_BASE_UPPER	31:0	rw	**Prefetchable Base Upper 32 Bit** • If the Prefetchable Memory Base register indicates support for 32-bit addressing, then this register is implemented as read-only register that returns zero when read • If the Prefetchable Memory Base register indicate support for 64-bit addressing, then this register is implemented as read/write register which must be initialized by configuration software • This register specifies the upper 32 bits, corresponding to Address[63:32], of the 64-bit base addresses which specify the prefetchable memory address range Note: The access attributes of this field are as follows: - Wire: No access. - Dbi: PREF_MEM_LIMIT_PREF_MEM_BASE_REG.PREF_MEM_DECODE ? RW : RO

19.4.2.108 Prefetchable Limit Upper 32 Bits Register

The Prefetchable Limit Upper 32 Bits register is an optional extension to the Prefetchable Memory Limit register.

PREF_LIMIT_UPPER_REG Offset address: $000402C_H$

Prefetchable Limit Upper 32 Bits Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PREF_MEM_LIMIT_UPPER															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PREF_MEM_LIMIT_UPPER															
rw															

Field	Bits	Type	Description
PREF_MEM_LIMIT_UPPER	31:0	rw	**Prefetchable Limit Upper 32 Bit** • If the Prefetchable Memory Limit register indicate support for 64-bit addressing, then this register is implemented as read/write register which must be initialized by configuration software • This register specifies the upper 32 bits, corresponding to Address[63:32], of the 64-bit base addresses which specify the prefetchable memory address range Note: The access attributes of this field are as follows: - Wire: No access. - Dbi: PREF_MEM_LIMIT_PREF_MEM_BASE_REG.PREF_MEM_DECODE ? RW : RO

19.4.2.109 I/O Limit and Base Upper 16 Bits Register

The I/O Limit Upper 16 Bits and I/O Base Upper 16 Bits registers are optional extensions to the I/O Limit and I/O Base registers.

IO_LIMIT_UPPER_IO_BASE_UPPER_REG Offset address: 0004030_H

I/O Limit and Base Upper 16 Bits Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
IO_LIMIT_UPPER															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IO_BASE_UPPER															
r															

Field	Bits	Type	Description
IO_BASE_UPPER	15:0	r	**I/O Base Upper 16 Bits** • If the I/O Base register indicates support for 16-bit I/O address decoding, then this register is implemented as a read-only register which return zero when read • If the I/O base register indicates support for 32-bit I/O addressing, then this register must be initialized by configuration software • If 32-bit I/O address decoding is supported, this register specifies the upper 16 bits, corresponding to Address[31:16], of the 32-bit base address, that specify the I/O address range • See the PCI-to-PCI Bridge Architecture Specification for additional details Note: The access attributes of this field are as follows: - Wire: No access. - Dbi: SEC_STAT_IO_LIMIT_IO_BASE_REG.IO_DECODE ? RW : RO
IO_LIMIT_UPPER	31:16	r	**I/O Limit Upper 16 Bits** • If the I/O Limit register indicates support for 16-bit I/O address decoding, then this register is implemented as a read-only register which return zero when read • If the I/O Limit register indicates support for 32-bit I/O addressing, then this register must be initialized by configuration software • If 32-bit I/O address decoding is supported, this register specifies the upper 16 bits, corresponding to Address[31:16], of the 32-bit limit address, that specify the I/O address range • See the PCI-to-PCI Bridge Architecture Specification for additional details) Note: The access attributes of this field are as follows: - Wire: No access. - Dbi: SEC_STAT_IO_LIMIT_IO_BASE_REG.IO_DECODE ? RW : RO

19.4.2.110 Capabilities Pointer Register

This register is used to point to a linked list of capabilities implemented by a Function.

TYPE1_CAP_PTR_REG Offset address: 0004034_H

Capabilities Pointer Register Kernel Reset value: $0000\ 0040_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_8															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVDP_8								CAP_POINTER							
r								rw							

Field	Bits	Type	Description
CAP_POINTER	7:0	rw	**Capabilities Pointer** This register is used to point to a linked list of capabilities implemented by this Function. Since all PCI Express Functions are required to implement the PCI Express Capability structure, this register must point to a valid capability structure and either this structure is the PCI Express Capability structure, or a subsequent list item points to the PCI Express Capability structure. The bottom two bits are Reserved and must be set to 00b. Software must mask these bits off before using this register as a pointer in Configuration Space to the first entry of a linked list of new capabilities. Note: The access attributes of this field are as follows: - Wire: No access. - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky.
RSVDP_8	31:8	r	**Reserved for future use**

19.4.2.111 Bridge Control, Interrupt Pin, and Interrupt Line Register

This register provides many of the same controls for the secondary interface that are provided by the Command Register for the primary interface. This register also identifies the legacy interrupt Message(s) a Function uses and communicates interrupt line routing information.

BRIDGE_CTRL_INT_PIN_INT_LINE_REG Offset address: 000403C$_H$

Bridge Control, Interrupt Pin, and Interrupt Line Register — Kernel Reset value: 0000 01FF$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BRIDGE_CTRL_RESERV									SBR	MSTR_ABORT_MODE	VGA_16B_DEC	VGA_EN	ISA_EN	SERR_EN	PERE
r									rw	rh	r	r	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
INT_PIN								INT_LINE							
rw								rw							

Field	Bits	Type	Description
INT_LINE	7:0	rw	**Interrupt Line** The Interrupt Line register communicates interrupt line routing information. The register must be implemented by any Function that uses an interrupt pin. Values in this register are programmed by system software and are system architecture specific. The Function itself does not use this value; rather the value in this register is used by device drivers and operating systems.

(table continues...)

(continued)

Field	Bits	Type	Description
INT_PIN	15:8	rw	**Interrupt PIN** The Interrupt Pin register register that identifies the legacy interrupt Message(s) the Function uses. All encodings other than the defined encodings are reserved. - PCI Express defines one legacy interrupt Message for a single Function device and up to four legacy interrupt Messages for a multi-Function device. For a single Function device, only INTA may be used. - Any Function on a multi-Function device can use any of the INTx Messages. If a device implements a single legacy interrupt Message, it must be INTA; if it implements two legacy interrupt Messages, they must be INTA and INTB; and so forth. - For a multi-Function device, all Functions may use the same INTx Message or each may have its own (up to a maximum of four Functions) or any combination thereof. - A single Function can never generate an interrupt request on more than one INTx Message. Note: The access attributes of this field are as follows: - Wire: No access. - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R Note: This register field is sticky. 00_H **NO_INT**: Indicates that the Function uses no legacy interrupt Message(s) 01_H **INTA**: Map to legacy interrupt Messages for INTA 02_H **INTB**: Map to legacy interrupt Messages for INTB 03_H **INTC**: Map to legacy interrupt Messages for INTC 04_H **INTD**: Map to legacy interrupt Messages for INTD **others,** Reserved
PERE	16	rw	**Parity Error Response Enable** This bit controls the logging of poisoned TLPs in the Master Data Parity Error bit in the Secondary Status register.
SERR_EN	17	rw	**SERR# Enable** This bit controls forwarding of ERR_COR, ERR_NONFATAL and ERR_FATAL from secondary to primary.
ISA_EN	18	rw	**ISA Enable** Modifies the response by the bridge to ISA I/O addresses. This applies only to I/O addresses that are enabled by the I/O Base and I/O Limit registers and are in the first 64 KB of I/O address space (0000 0000h to 0000 FFFFh). If this bit is set, the bridge will block any forwarding from primary to secondary of I/O transactions addressing the last 768 bytes in each 1-KB block. In the opposite direction (secondary to primary), I/O transactions will be forwarded if they address the last 768 bytes in each 1-KB block. 0_B **FRWD_DSP**: Forward downstream all I/O addresses in the address range defined by the I/O Base and I/O Limit registers 1_B **FRWD_USP**: Forward upstream ISA I/O addresses in the address range defined by the I/O Base and I/O Limit registers that are in the first 64 KB of PCI I/O address space (top 768 bytes of each 1-KB block

(table continues...)

(continued)

Field	Bits	Type	Description
VGA_EN	19	r	**VGA Enable** Modifies the response by the bridge to VGA compatible addresses. If the VGA Enable bit is set, the bridge will positively decode and forward the following accesses on the primary interface to the secondary interface (and, conversely, block the forwarding of these addresses from the secondary to primary interface): • Memory accesses in the range 000A 0000h to 000B FFFFh • I/O addresses in the first 64 KB of the I/O address space (Address[31:16] are 0000h) where Address[9:0] are in the ranges 3B0h to 3BBh and 3C0h to 3DFh (inclusive of ISA address aliases determined by the setting of VGA 16-bit Decode) If the VGA Enable bit is set, forwarding of these accesses is independent of the I/O address range and memory address ranges defined by the I/O Base and Limit registers, the Memory Base and Limit registers, and the Prefetchable Memory Base and Limit registers of the bridge. (Forwarding of these accesses is also independent of the setting of the ISA Enable bit (in the Bridge Control register) when the VGA Enable bit is set. Forwarding of these accesses is qualified by the I/O Space Enable and Memory Space Enable bits in the Command register.) For Functions that do not support VGA, the controller hardwires this bit to 0b. Note: The access attributes of this field are as follows: - Wire: No access. - Dbi: R 0_B **NO_FRWD_VGA**: Do not forward VGA compatible memory and I/O addresses from the primary to the secondary interface (addresses defined above) unless they are enabled for forwarding by the defined I/O and memory address ranges 1_B **FRWD_VGA**: Forward VGA compatible memory and I/O addresses (addresses defined above) from the primary interface to the secondary interface (if the I/O Space Enable and Memory Space Enable bits are set) independent of the I/O and memory address ranges and independent of the ISA Enable bit
VGA_16B_DEC	20	r	**VGA 16 bit decode** This bit only has meaning if VGA Enable bit is set. This bit enables system configuration software to select between 10-bit and 16-bit I/O address decoding for all VGA I/O register accesses that are forwarded from primary to secondary. For Functions that do not support VGA, the controller hardwires this bit to 0b. Note: The access attributes of this field are as follows: - Wire: No access. - Dbi: R 0_B **_10b_ADDR_DECODE**: Execute 10-bit address decodes on VGA I/O accesses 1_B **_16b_ADDR_DECODE**: Execute 16-bit address decodes on VGA I/O accesses
MSTR_ABORT_MODE	21	rh	**Master Abort Mode** This bit was originally described in the PCI-to-PCI Bridge Architecture Specification. Its functionality does not apply to PCI Express. The controller hardwires this bit to 0b. Note: The access attributes of this field are as follows: - Wire: No access. - Dbi: R/W

(table continues...)

(continued)

Field	Bits	Type	Description
SBR	22	rw	**Secondary Bus Reset** Setting this bit triggers a hot reset on the corresponding PCI Express Port. Software must ensure a minimum reset duration (Trst) as defined in the PCI Local Bus Specification. Port configuration registers must not be changed, except as required to update Port status. Software and systems must honor first-access-following-reset timing requirements, unless the Readiness Notifications mechanism is used.
BRIDGE_CTRL_RESERV	31:23	r	**Reserved**

19.4.2.112 Power Management Capabilities Register

This register provides information refarding the Power Management Capabilities.

CAP_ID_NXT_PTR_REG Offset address: 0004040_H

Power Management Capabilities Register Reset values see: Table 883

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PME_SUPPORT					D2_SUPPORT	D1_SUPPORT	AUX_CURR			DSI	0	PME_CLK	PM_SPEC_VER		
rwh					rw	rw	rw			rw	r	r	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PM_NEXT_POINTER								PM_CAP_ID							
rw								r							

Field	Bits	Type	Description
PM_CAP_ID	7:0	r	**Capability ID** This field returns 01h to indicate that this is the PCI Power Management Capability. Each function may have only one item in its capability list with Capability ID set to 01h.
PM_NEXT_POINTER	15:8	rw	**Next Capability Pointer** This field provides an offset into the function's configuration space pointing to the location of next item in the capabilities list. If there are no additional items in the capabilities list, this field is set to 00h. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R Note: This register field is sticky.
PM_SPEC_VER	18:16	rw	**Version** This field provides the Power Management specification version. The controller hardwires this field to 011b for functions compliant to PCI Express Base Specification, Revision 4.0, Version 1.0>. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R Note: This register field is sticky.
PME_CLK	19	r	**PME Clock** Does not apply to PCI Express, the controller hardwires it to 0b. Note: This register field is sticky.

(table continues...)

(continued)

Field	Bits	Type	Description
DSI	21	rw	**Device Specific Initialization** The DSI bit indicates whether special initialization of this function is required. When set, indicates that the function requires a device specific initialization sequence following a transition to the D0uninitialized state. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R Note: This register field is sticky. DSI capability is not supported in this implementation
AUX_CURR	24:22	rw	**Aux_Current** This 3 bit field reports the Vaux auxiliary current requirements for the function. If this function implements the Data Register, the controller hardwires this field to 000b. If PME_Support is 0 xxxxb (PME assertion from D3cold is not supported), the controller hardwires this field to 0000b. For functions where PME_Support is 1 xxxxb (PME assertion from D3cold is supported), and which do not implement the Data field, the encodings defined in Values: apply: Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R Note: This register field is sticky. 000_B **SELF_**: 0 self powered 001_B **_055mA_**: 55mA Vaux Max Current Required 010_B **_100mA_**: 100mA Vaux Max Current Required 011_B **_160mA_**: 160mA Vaux Max Current Required 100_B **_220mA_**: 220mA Vaux Max Current Required 101_B **_270mA_**: 270mA Vaux Max Current Required 110_B **_320mA_**: 320mA Vaux Max Current Required 111_B **_370mA_**: 375mA Vaux Max Current Required
D1_SUPPORT	25	rw	**D1_Support** If this bit is set, this function supports the D1 Power Management state. Functions that do not support D1 must always return a value of 0b for this bit. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R Note: This register field is sticky. D1-support capability is not supported in this implementation
D2_SUPPORT	26	rw	**D2_Support** If this bit is set, this function supports the D2 Power Management state. Functions that do not support D2 must always return a value of 0b for this bit. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R Note: This register field is sticky. D2-support capability is not supported in this implementation

(table continues...)

(continued)

Field	Bits	Type	Description
PME_SUPPORT	31:27	rwh	**PME_Support** This 5-bit field indicates the power states in which the function may generate a PME and/or forward PME messages. A value of 0b for any bit indicates that the function is not capable of asserting PME while in that power state. • bit(27) X XXX1b - PME can be generated from D0 • bit(28) X XX1Xb - PME can be generated from D1 • bit(29) X X1XXb - PME can be generated from D2 • bit(30) X 1XXXb - PME can be generated from D3hot • bit(31) 1 XXXXb - PME can be generated from D3cold Bit 31 (PME can be asserted from D3cold) represents a special case. Functions that set this bit require some sort of auxiliary power source. Implementation specific mechanisms are recommended to validate that the power source is available before setting this bit. Each bit that corresponds to a supported D-state must be set for PCI-PCI Bridge structures representing Ports on Root Complexes/Switches to indicate that the Bridge will forward PME Messages. Bit 31 must only be set if the Port is still able to forward PME Messages when main power is not available. The read value from this field is the write value && (sys_aux_pwr_det, 1'b1, D2_SUPPORT, D1_SUPPORT, 1'b1), where D1_SUPPORT and D2_SUPPORT are fields in this register. The reset value PME_SUPPORT_n && (sys_aux_pwr_det, 1'b1, D2_SUPPORT, D1_SUPPORT, 1'b1), where PME_SUPPORT_n is a configuration parameter. Note: The access attributes of this field are as follows: - Wire: R - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R Note: This register field is sticky.
0	20	r	**Reserved** Read as 0; should be written with 0.

Table 883 Reset values of CAP_ID_NXT_PTR_REG

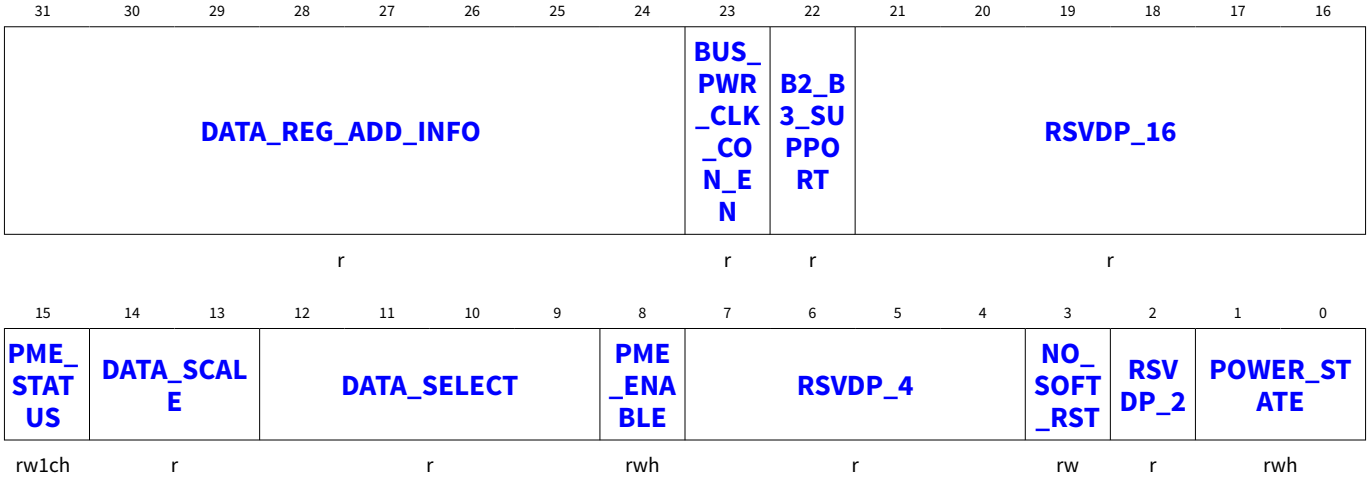
Reset	Reset value	Note
Kernel Reset	XXXX X001 1100 0011 0101 0000 0000 0001_B	

19.4.2.113 Power Management Control and Status Register

This register is used to manage the PCI function's power management state as well as to enable/monitor PMEs.

CON_STATUS_REG Offset address: 0004044_H

Power Management Control and Status Register Reset values see: Table 884



Field	Bits	Type	Description
POWER_STATE	1:0	rwh	**PowerState** This 2-bit field is used both to determine the current power state of a function and to set the function into a new power state. You can write to this register; however, the read-back value is the actual power state, not the write value. If you attempt to write an unsupported, optional state to this field, the write operation completes normally; however, the data is discarded and no state change occurs. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W 00_B **D0**: D0 power state 01_B **D1**: D1 power state 10_B **D2**: D2 power state 11_B **D3hot**: D3hot D3hot power state
RSVDP_2	2	r	**Reserved for future use**
NO_SOFT_RST	3	rw	**No_Soft_Reset** This bit indicates the state of the function after writing the PowerState field to transition the function from D3hot to D0. • When set, this transition preserves internal function state. The function is in D0Active and no additional software intervention is required • When clear, this transition results in undefined internal function state Regardless of this bit, functions that transition from D3hot to D0 by Fundamental Reset will return to D0Uninitialized with only PME context preserved if PME is supported and enabled. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R Note: This register field is sticky.
RSVDP_4	7:4	r	**Reserved for future use**

(table continues...)

(continued)

Field	Bits	Type	Description
PME_ENABLE	8	rwh	**PME_En** • When set, the function is permitted to generate a PME • When clear, the function is not permitted to generate a PME. If PME_Support is 1 xxxxb (PME generation from D3cold) or the function consumes Aux power and Aux power is available this bit is RWS and the bit is not modified by Conventional Reset If PME_Support is 0 xxxxb, this field is not sticky (RW). If PME_Support is 0 0000b, the controller hardwires this bit to 0b. Note: This register field is sticky.
DATA_SELECT	12:9	r	**Data_Select** This 4-bit field is used to select which data is to be reported through the Data and Data_Scale field. If the Data field is not implemented, this field must be hardwired to 0000b.
DATA_SCALE	14:13	r	**Data_Scale** This field indicates the scaling factor to be used when interpreting the value of the Data field. The value and meaning of this field varies depending on which data value has been selected by the Data_Select field. For more details, see 7.5.2.3 section of PCI Express Base Specification.
PME_STATUS	15	rw1ch	**PME_Status** This bit is set when the function normally generates a PME signal. The value of this bit is not affected by the value of the PME_En bit. If PME_Support bit 31 of the Power Management Capabilities register is clear, this bit is permitted to be hardwired to 0b. Functions that consume Aux power must preserve the value of this sticky register when Aux power is available. In such functions, this register value is not modified by Conventional Reset . Note: This register field is sticky. Software write with 1 clears this bit-field.
RSVDP_16	21:16	r	**Reserved for future use**
B2_B3_SUPPORT	22	r	**B2B3 Support for D3hot** If this field is set, B2B3 support for D3hot is available.
BUS_PWR_CLK_CON_EN	23	r	**Bus Power/Clock Control Enable** If this field is set, Bus Power/Clock Control is Enable.
DATA_REG_ADD_INFO	31:24	r	**Data** This field is used to report the state dependent data requested by the Data_Select field. The value of this field is scaled by the value reported by the Data_Scale field.

Table 884 Reset values of CON_STATUS_REG

Reset	Reset value	Note
Kernel Reset	0000 0000 0000 0000 0000 000X 0000 $10XX_B$	

19.4.2.114 MSI Capability Header and Message Control Register

This register holds MSI Capability Header information and controls the MSI behaviour.

PCI_MSI_CAP_ID_NEXT_CTRL_REG Offset address: 0004050_H

MSI Capability Header and Message Control Register Kernel Reset value: $018A\ 7005_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_27					PCI_MSI_EXT_DATA_EN	PCI_MSI_EXT_DATA_CAP	PCI_PVM_SUPPORT	PCI_MSI_64_BIT_ADDR_CAP	PCI_MSI_MULTIPLE_MSG_EN			PCI_MSI_MULTIPLE_MSG_CAP			PCI_MSI_ENABLE
r					rw	rw	r	rw	rw			rw			rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PCI_MSI_CAP_NEXT_OFFSET								PCI_MSI_CAP_ID							
rw								r							

Field	Bits	Type	Description
PCI_MSI_CAP_ID	7:0	r	**Capability ID** Indicates the MSI Capability structure. This field returns a Capability ID of 05h indicating that this is an MSI Capability structure.
PCI_MSI_CAP_NEXT_OFFSET	15:8	rw	**Next Capability Pointer** This field contains the offset to the next PCI Capability structure or 00h if no other items exist in the linked list of Capabilities. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R Note: This register field is sticky.
PCI_MSI_ENABLE	16	rw	**MSI Enable** • If set and the MSI-X Enable bit in the MSI-X Message Control register is clear, the function is permitted to use MSI to request service and is prohibited from using INTx interrupts. System configuration software sets this bit to enable MSI. A device driver is prohibited from writing this bit to mask a function's service request. For more details on control of INTx interrupts, see section 7.5.1.1 of PCI Express Base Specification. • If clear, the function is prohibited from using MSI to request service Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W

(table continues...)

(continued)

Field	Bits	Type	Description
PCI_MSI_MULTIPLE_MSG_CAP	19:17	rw	**Multiple Message Capable** System software reads this field to determine the number of requested vectors. The number of requested vectors must be aligned to a power of two (if a function requires three vectors, it requests four by initializing this field to 010b). All encodings other than the defined encodings are reserved. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R Note: This register field is sticky. 000_B **_1_VECTOR**: 1 vector requested 001_B **_2_VECTOR**: 2 vectors requested 010_B **_4_VECTOR**: 4 vectors requested 011_B **_8_VECTOR**: 8 vectors requested 100_B **_16_VECTOR**: 16 vectors requested 101_B **_32_VECTOR**: 32 vectors requested **others,** Reserved
PCI_MSI_MULTIPLE_MSG_EN	22:20	rw	**Multiple Message Enable** Software writes to this field to indicate the number of allocated vectors (equal to or less than the number of requested vectors). The number of allocated vectors is aligned to a power of two. If a function requests four vectors (indicated by a Multiple Message Capable encoding of 010b), system software can allocate either four, two, or one vector by writing a 010b, 001b, or 000b to this field, respectively. When MSI is enabled, a function will be allocated at least 1 vector. All encodings other than the defined encodings are reserved. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W 000_B **_1_VECTOR**: 1 vector allocated 001_B **_2_VECTOR**: 2 vectors allocated 010_B **_4_VECTOR**: 4 vectors allocated 011_B **_8_VECTOR**: 8 vectors allocated 100_B **_16_VECTOR**: 16 vectors allocated 101_B **_32_VECTOR**: 32 vectors allocated **others,** Reserved
PCI_MSI_64_BIT_ADDR_CAP	23	rw	**64 bit address capable** • If set, the function is capable of sending a 64-bit message address • If clear, the function is not capable of sending a 64-bit message address. This bit must be set if the function is a PCI Express Endpoint Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R Note: This register field is sticky.
PCI_PVM_SUPPORT	24	r	**Per-Vector Masking Capable** • If set, the function supports MSI Per-Vector Masking • If clear, the function does not support MSI Per-Vector Masking

(table continues...)

(continued)

Field	Bits	Type	Description
PCI_MSI_EXT_DATA_CAP	25	rw	**Extended Message Data Capable** • If set, the function is capable of providing Extended Message Data • If clear, the function does not support providing Extended Message Data Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R Note: This register field is sticky. Extended Message Data Capability is not supported in the current implementation
PCI_MSI_EXT_DATA_EN	26	rw	**Extended Message Data Enable** • If set, the function is enabled to provide Extended Message Data • If clear, the function is not enabled to provide Extended Message Data Note: The access attributes of this field are as follows: • Wire: PCI_MSI_CAP_ID_NEXT_CTRL_REG.PCI_MSI_EXT_DATA_CAP ? RW : RO • Dbi: PCI_MSI_CAP_ID_NEXT_CTRL_REG.PCI_MSI_EXT_DATA_CAP ? RW : RO
RSVDP_27	31:27	r	**Reserved for future use**

19.4.2.115 Message Address Register for MSI (Offset 04h)

This register holds the system specified message address for an MSI transaction.

MSI_CAP_OFF_04H_REG Offset address: 0004054_H

Message Address Register for MSI [Offset 04h) Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PCI_MSI_CAP_OFF_04H															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PCI_MSI_CAP_OFF_04H														RSVDP_0	
rw														r	

Field	Bits	Type	Description
RSVDP_0	1:0	r	**Reserved for future use**
PCI_MSI_CAP_OFF_04H	31:2	rw	**Message Address - System-specified message address** If the Message Enable bit (bit 16 of the PCI_MSI_CAP_ID_NEXT_CTRL_REG register) is set, the contents of this field specify the DWORD-aligned address (Address[31:02]) for the MSI transaction. Address[1:0] are set to 00b. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W

19.4.2.116 Message Address Register for MSI (Offset 08h)

This register represents the Message Upper Address Register for MSI (Offset 08h). It specifies the Message Upper Address (System-specified message upper address). This register is required for PCI Express Endpoints and is optional for other function types. If the Message Enable bit (bit 0 of the Message Control register) is set, the contents of this register (if non-zero) specify the upper 32-bits of a 64-bit message address (Address[63:32]). If the contents of this register are zero, the Function uses the 32 bit address specified by the Message Address register.

MSI_CAP_OFF_08H_REG Offset address: 0004058_H

Message Address Register for MSI [Offset 08h) Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PCI_MSI_CAP_OFF_0AH															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PCI_MSI_CAP_OFF_08H															
rw															

Field	Bits	Type	Description
PCI_MSI_CAP_OFF_08H	15:0	rw	**For a function that supports a 32-bit message address, this field contains Message Data (System-specified message data)** If the Message Enable bit (bit 16 of the PCI_MSI_CAP_ID_NEXT_CTRL_REG) is set, the function sends a DWORD Memory Write transaction using Message Data for the lower 16 bits. All 4 Byte Enables are set. The Multiple Message Enable field (bits 22:20 of the PCI_MSI_CAP_ID_NEXT_CTRL_REG) defines the number of low order message data bits the function is permitted to modify to generate its system software allocated vectors. For example, a Multiple Message Enable encoding of 010b indicates the function has been allocated four vectors and is permitted to modify message data bits 1 and 0 (a function modifies the lower message data bits to generate the allocated number of vectors). If the Multiple Message Enable field is 000b, the Function is not permitted to modify the message data. For a function that supports a 64-bit message address, it contains lower 16 bits of the Message Upper Address. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W

(table continues...)

(continued)

Field	Bits	Type	Description
PCI_MSI_CAP_OFF_0AH	31:16	rw	**For a function that supports a 32-bit message address, this field contains Extended Message Data (System-specified message data)** Extended Message Data Capability is not supported in the current implementation For the MSI Capability structures without per-vector masking, it must be implemented if the Extended Message Data Capable bit is set; otherwise, it is outside the MSI Capability structure and undefined. For the MSI Capability structures with Per-vector Masking, it must be implemented if the Extended Message Data Capable bit is set; otherwise, it is RsvdP. If the Extended Message Data Enable bit (bit 26 of the PCI_MSI_CAP_ID_NEXT_CTRL_REG) is set, the DWORD Memory Write transaction uses Extended Message Data for the upper 16 bits; otherwise, it uses 0000h for the upper 16 bits. For a function that supports a 64-bit message address, it contains upper 16 bits of the Message Upper Address. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W

19.4.2.117 Message Address Register for MSI (Offset 0Ch)

This register contains Message Data.

MSI_CAP_OFF_0CH_REG Offset address: $000405C_H$

Message Address Register for MSI [Offset 0Ch) Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PCI_MSI_CAP_OFF_0EH															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PCI_MSI_CAP_OFF_0CH															
rw															

Field	Bits	Type	Description
PCI_MSI_CAP_OFF_0CH	15:0	rw	**For a function that supports a 32-bit message address, this field contains the lower Mask Bits when the Per Vector Masking Capable bit (PCI_MSI_CAP_ID_NEXT_CTRL_REG.PCI_PVM_SUPPORT) is set** This field contains Message Data (System-specified message data). If the Message Enable bit (bit 16 of the PCI_MSI_CAP_ID_NEXT_CTRL_REG) is set, the function sends a DWORD Memory Write transaction using Message Data for the lower 16 bits. All 4 Byte Enables are set. The Multiple Message Enable field (bits 22:20 of the PCI_MSI_CAP_ID_NEXT_CTRL_REG) defines the number of low order message data bits the function is permitted to modify to generate its system software allocated vectors. For example, a Multiple Message Enable encoding of 010b indicates the function has been allocated four vectors and is permitted to modify message data bits 1 and 0 (a function modifies the lower message data bits to generate the allocated number of vectors). If the Multiple Message Enable field is 000b, the Function is not permitted to modify the message data. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W

(table continues...)

(continued)

Field	Bits	Type	Description
PCI_MSI_CAP_OFF_0EH	31:16	rw	**For a function that supports a 32-bit message address, this field contains the upper Mask Bits when the Per Vector Masking Capable bit (PCI_MSI_CAP_ID_NEXT_CTRL_REG.PCI_PVM_SUPPORT) is set** For a function that supports a 64-bit message address, this field contains Message Data (System-specified message data). Note: The access attributes of this field are as follows: - Wire: RW - Dbi: RW

19.4.2.118 Message Address Register for MSI (Offset 10h)

This register contains the Mask Bits.

MSI_CAP_OFF_10H_REG Offset address: 0004060_H

Message Address Register for MSI [Offset 10h) Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PCI_MSI_CAP_OFF_10H															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PCI_MSI_CAP_OFF_10H															
rw															

Field	Bits	Type	Description
PCI_MSI_CAP_OFF_10H	31:0	rw	**Used for MSI when the Per Vector Masking Capable bit (PCI_MSI_CAP_ID_NEXT_CTRL_REG.PCI_PVM_SUPPORT) is set** It contains Mask Bits. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W

19.4.2.119 Message Address Register for MSI (Offset 14h)

Pending Bits Register for MSI.

MSI_CAP_OFF_14H_REG Offset address: 0004064_H

Message Address Register for MSI [Offset 14h) Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PCI_MSI_CAP_OFF_14H															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PCI_MSI_CAP_OFF_14H															
r															

Field	Bits	Type	Description
PCI_MSI_CAP_OFF_14H	31:0	r	**Pending Bits** For each pending bit that is set, the function has a pending associated message.

19.4.2.120 PCI Express Capabilities, ID, Next Pointer Register

This is the PCI Express Capabilities, ID, and Next Pointer Register.

PCIE_CAP_ID_PCIE_NEXT_CAP_PTR_PCIE_CAP_REG Offset address: 0004070_H

PCI Express Capabilities, ID, Next Pointer Register Kernel Reset value: $0042\ B010_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_31	RSVD	PCIE_INT_MSG_NUM					PCIE_SLOT_IMP	PCIE_DEV_PORT_TYPE				PCIE_CAP_REG			
r	r	rw					rw	rh				r			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PCIE_CAP_NEXT_PTR								PCIE_CAP_ID							
rw								r							

Field	Bits	Type	Description
PCIE_CAP_ID	7:0	r	**Capability ID** Indicates the PCI Express Capability structure. This field must return a Capability ID of 10h indicating that this is a PCI Express Capability structure.
PCIE_CAP_NEXT_PTR	15:8	rw	**Next Capability Pointer** This field contains the offset to the next PCI Capability structure or 00h if no other items exist in the linked list of Capabilities. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R Note: This register field is sticky.
PCIE_CAP_REG	19:16	r	**Capability Version** Indicates PCI-SIG defined PCI Express Capability structure version number. A version of the specification that changes the PCI Express Capability structure in a way that is not otherwise identifiable (for example, through a new Capability field) is permitted to increment this field. All such changes to the PCI Express Capability structure must be software-compatible. Software must check for Capability Version numbers that are greater than or equal to the highest number defined when the software is written, as functions reporting any such Capability Version numbers will contain a PCI Express Capability structure that is compatible with that piece of software. The controller hardwires this field to 2h for functions compliant to PCI Express Base Specification, Revision 4.0, Version 1.0. Note: This register field is sticky.

(table continues...)

(continued)

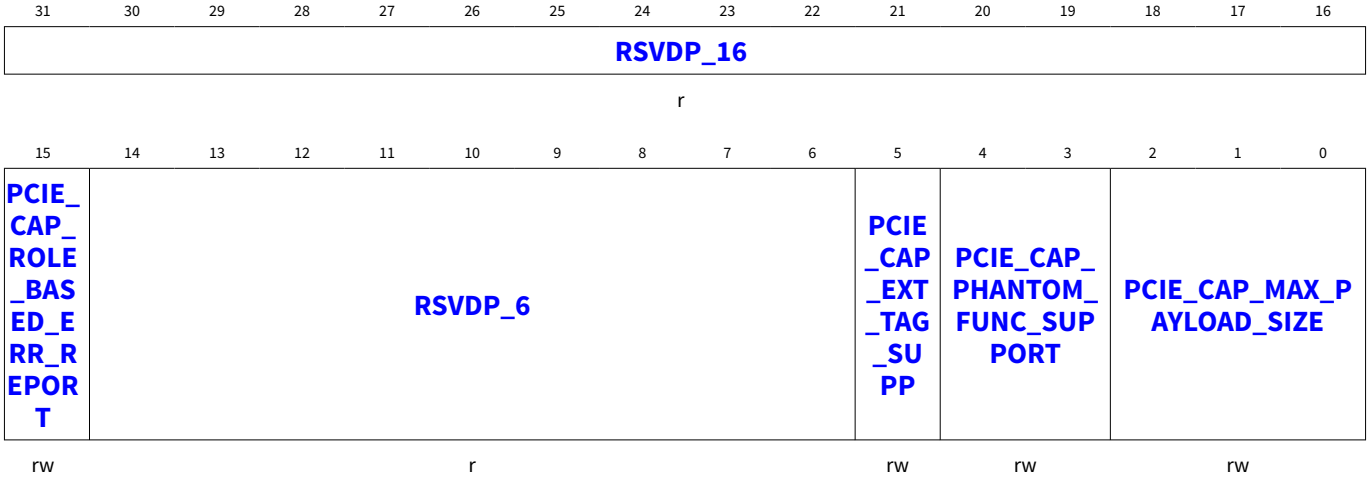
Field	Bits	Type	Description
PCIE_DEV_PORT_TYPE	23:20	rh	**Device/Port Type** Indicates the specific type of this PCI Express function. Note: Different functions in a Multi-Function Device can generally be of different types. Defined encodings for functions that implement a Type 00h PCI Configuration Space header are: Defined encodings for functions that implement a Type 01h PCI Configuration Space header are: All other encodings are Reserved. Note: Different Endpoint types have notably different requirements in Section 1.3.2 of PCI Express Base Specification regarding I/O resources, Extended Configuration Space, and other capabilities. 0_H **PCIE_EP**: PCI Express Endpoint 1_H **PCIE_LEGACY_EP**: Legacy PCI Express Endpoint 4_H **ROOT_PORT_PCIE_RC**: Root Port of PCI Express Root Complex 5_H **USP_PCIE_SWITCH**: Upstream Port of PCI Express Switch 6_H **DSP_PCIE_SWITCH**: Downstream Port of PCI Express Switch **others,** Reserved
PCIE_SLOT_IMP	24	rw	**Slot Implemented** When set, this bit indicates that the Link associated with this Port is connected to a slot (as compared to being connected to a system-integrated device or being disabled). This bit is valid for Downstream Ports. This bit is undefined for Upstream Ports. Note: The access attributes of this field are as follows: - Wire: HWINIT - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R
PCIE_INT_MSG_NUM	29:25	rw	**PCIE Interrupt Message Number** Interrupt Message Number. This field indicates which MSI/MSI-X vector is used for the interrupt message generated in association with any of the status bits of this Capability structure. For MSI, the value in this field indicates the offset between the base Message Data and the interrupt message that is generated. Hardware is required to update this field so that it is correct if the number of MSI Messages assigned to the Function changes when software writes to the Multiple Message Enable field in the MSI Message Control register. For MSI-X, the value in this field indicates which MSI-X Table entry is used to generate the interrupt message. The entry must be one of the first 32 entries even if the Function implements more than 32 entries. For a given MSI-X implementation, the entry must remain constant. If both MSI and MSI-X are implemented, they are permitted to use different vectors, though software is permitted to enable only one mechanism at a time. If MSI-X is enabled, the value in this field must indicate the vector for MSI-X. If MSI is enabled or neither is enabled, the value in this field must indicate the vector for MSI. If software enables both MSI and MSI-X at the same time, the value in this field is undefined. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R Note: This register field is sticky.
RSVD	30	r	**Reserved**
RSVDP_31	31	r	**Reserved for future use**

19.4.2.121 Device Capabilities Register

The Device Capabilities register identifies PCI Express device function specific capabilities.

DEVICE_CAPABILITIES_REG Offset address: 0004074_H

Device Capabilities Register Kernel Reset value: $0000\ 8001_H$



Field	Bits	Type	Description
PCIE_CAP_MAX_PAYLOAD_SIZE	2:0	rw	**Max_Payload_Size Supported** This field indicates the maximum payload size that the function can support for TLPs. All encodings other than the defined encodings are reserved. The functions of a Multi-Function Device are permitted to report different values for this field. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R Note: This register field is sticky. 000_B **MAX_128B_SIZE**: 128 bytes max payload size 001_B **MAX_256B_SIZE**: 256 bytes max payload size 010_B **MAX_512B_SIZE**: 512 bytes max payload size 011_B **MAX_1024B_SIZE**: 1024 bytes max payload size 100_B **MAX_2048B_SIZE**: 2048 bytes max payload size 101_B **MAX_4096B_SIZE**: 4096 bytes max payload size **others,** Reserved

(table continues...)

(continued)

Field	Bits	Type	Description
PCIE_CAP_PHANTOM_FUNC_SUPPORT	4:3	rw	**Phantom Functions Supported** For functions that do not implement this capability (Phantom function capability is not available in this implementation), this bit will be hardwired to 0b. This field indicates the support for use of unclaimed function numbers to extend the number of outstanding transactions allowed by logically combining unclaimed function numbers (called Phantom Functions) with the Tag identifier (see Section 2.2.6.2 of PCI Express Base Specification for a description of Tag Extensions). With every Function in an ARI Device, the Phantom Functions Supported field must be set to 00b. The remainder of this field description applies only to non-ARI Multi-Function Devices. This field indicates the number of most significant bits of the Function Number portion of Requester ID that are logically combined with the Tag identifier. Note: Phantom Function support for the function must be enabled by the Phantom Functions Enable field in the Device Control register before the Function is permitted to use the Function Number field in the Requester ID for Phantom Functions. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R Note: This register field is sticky. 00_B **NO_PHNATOM_FUNC**: No Function Number bits are used for Phantom Functions Multi-Function Devices are permitted to implement up to 8 independent functions. 01_B **MAX_1_PHANTOM_PER_FUNC**: The most significant bit of the Function number in Requester ID is used for Phantom Functions; a Multi-Function Device is permitted to implement Functions 0-3 Functions 0, 1, 2, and 3 are permitted to use Function Numbers 4, 5, 6, and 7 respectively as Phantom Functions. 10_B **MAX_3_PHANTOM_PER_FUNC**: The two most significant bits of Function Number in Requester ID are used for Phantom Functions; a Multi-Function Device is permitted to implement Functions 0-1 Function 0 is permitted to use Function Numbers 2, 4, and 6 for Phantom Functions. Function 1 is permitted to use Function Numbers 3, 5, and 7 as Phantom Functions. 11_B **SINGLE_FUNC_MAX_7_PHANTOM_FUNC**: All 3 bits of Function Number in Requester ID used for Phantom Functions The device must have a single Function 0 that is permitted to use all other Function Numbers as Phantom Functions.

(table continues...)

(continued)

Field	Bits	Type	Description
PCIE_CAP_EXT_TAG_SUPP	5	rw	**Extended Tag Field Supported** This bit, in combination with the 10-Bit Tag Requester Supported bit in the Device Capabilities 2 register, indicates the maximum supported size of the Tag field as a Requester. This bit must be set if the 10-Bit Tag Requester Supported bit is set. Note: 8-bit Tag field generation must be enabled by the Extended Tag Field Enable bit in the Device Control register of the Requester Function before 8-bit Tags can be generated by the Requester. See Section 2.2.6.2 of PCI Express Base Specification for interactions with enabling the use of 10-Bit Tags. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R Note: This register field is sticky. 0_B **_5b_TAG**: 5-bit Tag field supported 1_B **_8b_TAG**: 8-bit Tag field supported
RSVDP_6	14:6	r	**Reserved for future use**
PCIE_CAP_ROLE_BASED_ERR_REPORT	15	rw	**Role-Based Error Reporting** When set, this bit indicates that the function implements the functionality originally defined in the Error Reporting ECN for PCI Express Base Specification, Revision 1.0a, and later incorporated into PCI Express Base Specification, Revision 1.1. This bit must be set by all functions conforming to the ECN, PCI Express Base Specification, Revision 1.1., or subsequent PCI Express Base Specification revisions. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R Note: This register field is sticky.
RSVDP_16	31:16	r	**Reserved for future use**

19.4.2.122 Device Control and Device Status Register

This register controls PCI Express device specific parameters and provides information about PCI Express device (function) specific parameters.

DEVICE_CONTROL_DEVICE_STATUS Offset address: 0004078_H

Device Control and Device Status Register Reset values see: Table 885

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_22										PCIE_CAP_TRANS_PENDING	PCIE_CAP_AUX_POWER_DETECTED	PCIE_CAP_UNSUPORTED_REQ_DETECTED	PCIE_CAP_FATAL_ERR_DETECTED	PCIE_CAP_NON_FATAL_ERR_DETECTED	PCIE_CAP_CORR_ERR_DETECTED
r										r	rh	rw1ch	rw1ch	rw1ch	rw1ch

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	PCIE_CAP_MAX_READ_REQ_SIZE			PCIE_CAP_EN_NO_SNOOP	PCIE_CAP_AUX_POWER_PM_EN	PCIE_CAP_PHANTOM_FUNC_EN	PCIE_CAP_EXT_TAG_EN	PCIE_CAP_MAX_PAYLOAD_SIZE_CS			PCIE_CAP_EN_REL_ORDER	PCIE_CAP_UNSUPORT_REQ_REP_EN	PCIE_CAP_FATAL_ERR_REPORT_EN	PCIE_CAP_NON_FATAL_ERR_REPORT_EN	PCIE_CAP_CORR_ERR_REPORT_EN
r	rw			rw	rwh	rh	rh	rw			rw	rw	rw	rw	rw

Field	Bits	Type	Description
PCIE_CAP_CORR_ERR_REPORT_EN	0	rw	**Correctable Error Reporting Enable** This bit, in conjunction with other bits, controls sending ERR_COR Messages (for more details, see section 6.2.5, section 6.2.6, and section 6.2.10.2 of PCI Express Base Specification). For a Multi-Function device, this bit controls error reporting for each function from point-of-view of the respective function. For a Root Port, the reporting of correctable errors is internal to the root. No external ERR_COR Message is generated.
PCIE_CAP_NON_FATAL_ERR_REPORT_EN	1	rw	**Non-Fatal Error Reporting Enable** This bit, in conjunction with other bits, controls sending ERR_NONFATAL Messages (for more details, see section 6.2.5 and Section 6.2.6 of PCI Express Base Specification). For a Multi-Function Device, this bit controls error reporting for each function from point-of-view of the respective Function. For a Root Port, the reporting of Non-fatal errors is internal to the root. No external ERR_NONFATAL Message is generated.
PCIE_CAP_FATAL_ERR_REPORT_EN	2	rw	**Fatal Error Reporting Enable** This bit, in conjunction with other bits, controls sending ERR_FATAL Messages (for more details, see section 6.2.5 and section 6.2.6 of of PCI Express Base Specification). For a Multi-Function device, this bit controls error reporting for each function from point-of-view of the respective function. For a Root Port, the reporting of Fatal errors is internal to the root. No external ERR_FATAL Message is generated.

(table continues...)

(continued)

Field	Bits	Type	Description
PCIE_CAP_UNSUPPORT_REQ_REP_EN	3	rw	**Unsupported Request Reporting Enable** This bit, in conjunction with other bits, controls the signaling of Unsupported Request Errors by sending error Messages (for more details, see section 6.2.5 and section 6.2.6 of PCI Express Base Specification). For a Multi-Function Device, this bit controls error reporting for each Function from point-of-view of the respective Function.
PCIE_CAP_EN_REL_ORDER	4	rw	**Enable Relaxed Ordering** If this bit is set, the function is permitted to set the Relaxed Ordering bit in the Attributes field of transactions it initiates that do not require strong write ordering (for more details, see section 2.2.6.4 and section 2.4 of PCI Express Base Specification). For a function that never sets the Relaxed Ordering attribute in transactions it initiates as a Requester, the controller hardwires this bit to 0b. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
PCIE_CAP_MAX_PAYLOAD_SIZE_CS	7:5	rw	**Max_Payload_Size** This field sets maximum TLP payload size for the Function. As a Receiver, the Function must handle TLPs as large as the set value. As a Transmitter, the Function must not generate TLPs exceeding the set value. Permissible values that can be programmed are indicated by the Max_Payload_Size Supported field (PCIE_CAP_MAX_PAYLOAD_SIZE) in the Device Capabilities (DEVICE_CAPABILITIES_REG) register (for more details, see section 7.5.3.3 of PCI Express Base Specification). This field sets the maximum Read Request size for the function as a Requester. The function must not generate Read Requests with a size exceeding the set value. For Functions that support only the 128-byte max payload size, the controller hardwires this field to 000b. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W 000_B **MAX_128B_SIZE**: 128 bytes maximum Read Request size 001_B **MAX_256B_SIZE**: 256 bytes maximum Read Request size 010_B **MAX_512B_SIZE**: 512 bytes maximum Read Request size 011_B **MAX_1024B_SIZE**: 1024 bytes maximum Read Request size 100_B **MAX_2048B_SIZE**: 2048 bytes maximum Read Request size 101_B **MAX_4096B_SIZE**: 4096 bytes maximum Read Request size 110_B **RESERVED1**: RESERVED 111_B **RESERVED2**: RESERVED

(table continues...)

(continued)

Field	Bits	Type	Description
PCIE_CAP_EXT_TAG_EN	8	rh	**Extended Tag Field Enable** This bit, in combination with the 10-Bit Tag Requester Enable bit in the Device Control 2 register, determines how many Tag field bits a Requester is permitted to use. When the 10-Bit Tag Requester Enable bit is clear, • If the Extended Tag Field Enable bit is set, the function is permitted to use an 8-bit Tag field as a Requester • If the Extended Tag Field Enable bit is clear, the Function is to a 5-bit Tag field See section 2.2.6.2 of PCI Express Base Specification for required behavior when the 10-Bit Tag Requester Enable bit is set. If software changes the value of the Extended Tag Field Enable bit while the function has outstanding Non-Posted Requests, the result is undefined. For functions that do not implement this capability, the controller hardwires this bit to 0b. Note: The access attributes of this field are as follows: - Wire: DEVICE_CAPABILITIES_REG.PCIE_CAP_EXT_TAG_SUPP ? RW : RO - Dbi: DEVICE_CAPABILITIES_REG.PCIE_CAP_EXT_TAG_SUPP ? RW : RO
PCIE_CAP_PHANTOM_FUNC_EN	9	rh	**Phantom Functions Enable** For functions that do not implement this capability (Phantom function capability is not available in this implementation), this bit will be hardwired to 0b. This bit, in combination with the 10-Bit Tag Requester Enable bit in the Device Control 2 register, determines how many Tag field bits a Requester is permitted to use. When the 10-Bit Tag Requester Enable bit is clear, • If this bit is set, it enables a function to use unclaimed functions as Phantom functions to extend the number of outstanding transaction identifiers • If this bit is clear, the function is not allowed to use Phantom functions For more details, see section 2.2.6.2 of PCI Express Base Specification. Software should not change the value of this bit while the function has outstanding Non-Posted Requests; otherwise, the result is undefined.Note: The access attributes of this field are as follows: • Wire: DEVICE_CAPABILITIES_REG.PCIE_CAP_PHANTOM_FUNC_SUPPORT ? RW : RO • Dbi: DEVICE_CAPABILITIES_REG.PCIE_CAP_PHANTOM_FUNC_SUPPORT ? RW : R

(table continues...)

(continued)

Field	Bits	Type	Description
PCIE_CAP_AUX_POWER_PM_EN	10	rwh	**Aux Power PM Enable** This bit is derived by sampling the sys_aux_pwr_det input. When set this bit, enables a function to draw Aux power independent of PME Aux power. Functions that require Aux power on legacy operating systems should continue to indicate PME Aux power requirements. Aux power is allocated as requested in the Aux_Current field of the Power Management Capabilities register (PMC), independent of the PME_En bit in the Power Management Control/Status register (PMCSR). For Multi-Function devices, a component is allowed to draw Aux power if at least one of the functions has this bit set. Note: Functions that consume Aux power must preserve the value of this sticky register when Aux power is available. In such functions, this bit is not modified by Conventional Reset. For functions that do not implement this capability, the controller hardwires this bit to 0b. Note: This register field is sticky.
PCIE_CAP_EN_NO_SNOOP	11	rw	**Enable No Snoop** If this bit is set, the function is permitted to Set the No Snoop bit in the Requester Attributes of transactions it initiates that do not require hardware enforced cache coherency (see section 2.2.6.5 in PCI Express Base Specification). Note: Setting this bit to 1b should not cause a function to set the No Snoop attribute on all transactions that it initiates. Even when this bit is set, a function is only permitted to set the No Snoop attribute on a transaction when it can guarantee that the address of the transaction is not stored in any cache in the system. The controller hardwires this bit 0b if a function would never set the No Snoop attribute in transactions it initiates. Note: The access attributes of this field are as follows: - Wire: R - Dbi: R
PCIE_CAP_MAX_READ_REQ_SIZE	14:12	rw	**Max_Read_Request_Size** This field sets the maximum Read Request size for the function as a Requester. The function must not generate Read Requests with a size exceeding the set value. For functions that do not generate Read Requests larger than 128 bytes and functions that do not generate Read Requests on their own behalf, the controller implements this field as Read Only (RO) with a value of 000b. 000_B **MAX_128B_SIZE**: 128 bytes maximum Read Request size 001_B **MAX_256B_SIZE**: 256 bytes maximum Read Request size 010_B **MAX_512B_SIZE**: 512 bytes maximum Read Request size 011_B **MAX_1024B_SIZE**: 1024 bytes maximum Read Request size 100_B **MAX_2048B_SIZE**: 2048 bytes maximum Read Request size 101_B **MAX_4096B_SIZE**: 4096 bytes maximum Read Request size 110_B **RESERVED1**: RESERVED 111_B **RESERVED2**: RESERVED
PCIE_CAP_CORR_ERR_DETECTED	16	rw1ch	**Correctable Error Detected** This bit indicates status of correctable errors detected. Errors are logged in this register regardless of whether error reporting is enabled or not in the Device Control register. For functions supporting Advanced Error Handling, errors are logged in this register regardless of the settings of the Correctable Error Mask register. Software write with 1 clears this bit-field.

(table continues...)

(continued)

Field	Bits	Type	Description
PCIE_CAP_NON_FATAL_ERR_DETECTED	17	rw1ch	**Non-Fatal Error Detected** This bit indicates status of Non-fatal errors detected. Errors are logged in this register regardless of whether error reporting is enabled or not in the Device Control register. For functions supporting Advanced Error Handling, errors are logged in this register regardless of the settings of the Uncorrectable Error Mask register. Software write with 1 clears this bit-field.
PCIE_CAP_FATAL_ERR_DETECTED	18	rw1ch	**Fatal Error Detected** This bit indicates status of Fatal errors detected. Errors are logged in this register regardless of whether error reporting is enabled or not in the Device Control register. For Functions supporting Advanced Error Handling, errors are logged in this register regardless of the settings of the Uncorrectable Error Mask register. Software write with 1 clears this bit-field.
PCIE_CAP_UNSUPPORTED_REQ_DETECTED	19	rw1ch	**Unsupported Request Detected** This bit indicates that the function received an Unsupported Request. Errors are logged in this register regardless of whether error reporting is enabled or not in the Device Control register. Software write with 1 clears this bit-field.
PCIE_CAP_AUX_POWER_DETECTED	20	rh	**AUX Power Detected** Functions that require Aux power report this bit as set if Aux power is detected by the function. .
PCIE_CAP_TRANS_PENDING	21	r	**Transactions Pending** Endpoints: When set, this bit indicates that the function has issued Non-Posted Requests that have not been completed. A Function reports this bit cleared only when all outstanding Non-Posted Requests have completed or have been terminated by the Completion Timeout mechanism. Root and Switch Ports: The controller hardwires this bit to 0b.
RSVDP_22	31:22	r	**Reserved for future use**
0	15	r	**Reserved** Read as 0; should be written with 0.

Table 885 Reset values of DEVICE_CONTROL_DEVICE_STATUS

Reset	Reset value	Note
Kernel Reset	0000 0000 000X 0000 0010 1XXX 0001 0000_B	

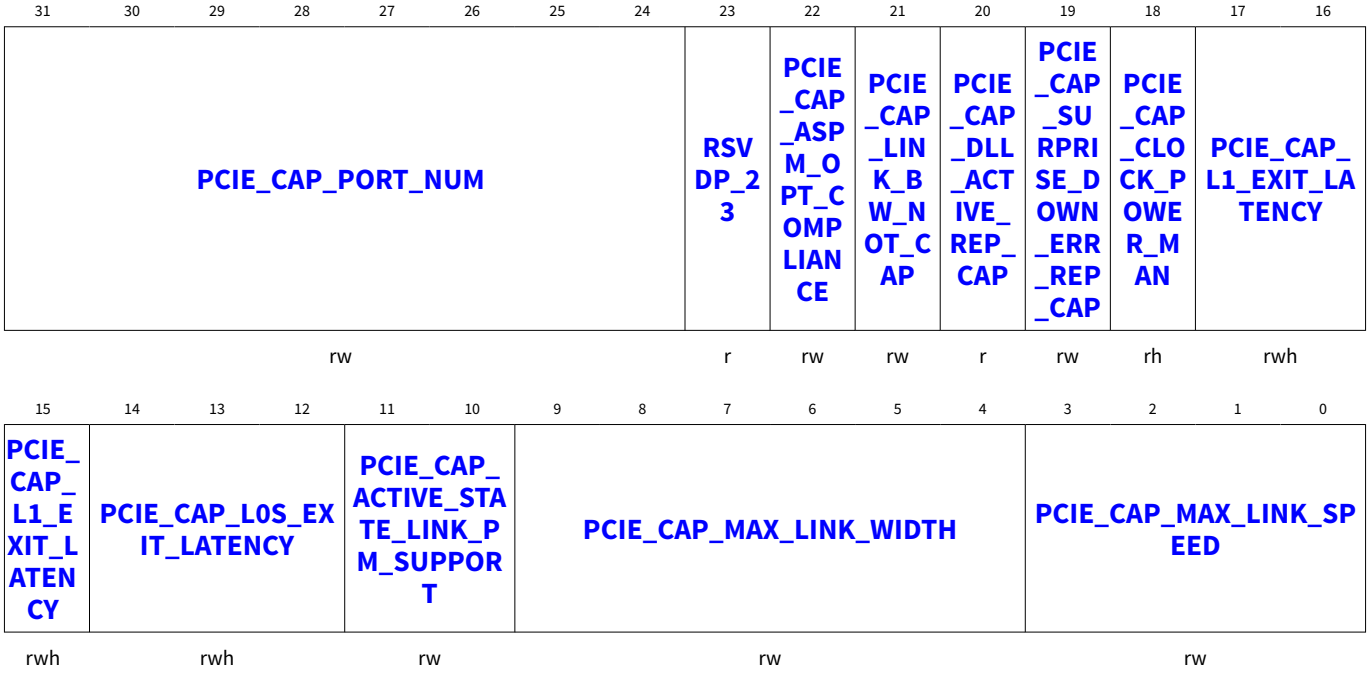
19.4.2.123 Link Capabilities Register

The Link Capabilities register identifies PCI Express Link specific capabilities.

LINK_CAPABILITIES_REG Offset address: $000407C_H$

Link Capabilities Register Reset values see: Table 886

19 Peripheral Component Interconnect Express (PCIe)



Field	Bits	Type	Description
PCIE_CAP_MAX_LINK_SPEED	3:0	rw	**Max Link Speed** This field indicates the maximum Link speed of the associated Port. The encoded value specifies a bit location in the Supported Link Speeds Vector (in the Link Capabilities 2 register) that corresponds to the maximum Link speed. All encodings other than the defined encodings are reserved. Multi-Function Devices associated with an Upstream Port must report the same value in this field for all functions. In M-PCIe mode, the reset and dynamic values of this field are calculated by the controller. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R Note: This register field is sticky. 1_H **SUP_LINK_SPEED_FIELD_BIT_0**: Supported Link Speeds Vector field bit 0 2_H **SUP_LINK_SPEED_FIELD_BIT_1**: Supported Link Speeds Vector field bit 1 3_H **SUP_LINK_SPEED_FIELD_BIT_2**: Supported Link Speeds Vector field bit 2 4_H **SUP_LINK_SPEED_FIELD_BIT_3**: Supported Link Speeds Vector field bit 3 5_H **SUP_LINK_SPEED_FIELD_BIT_4**: Supported Link Speeds Vector field bit 4 6_H **SUP_LINK_SPEED_FIELD_BIT_5**: Supported Link Speeds Vector field bit 5 7_H **SUP_LINK_SPEED_FIELD_BIT_6**: Supported Link Speeds Vector field bit 6 **others,** Reserved

(table continues...)

(continued)

Field	Bits	Type	Description
PCIE_CAP_MAX_LINK_WIDTH	9:4	rw	**Maximum Link Width** This field indicates the maximum Link width (xN - corresponding to N Lanes) implemented by the component. This value is permitted to exceed the number of Lanes routed to the slot (Downstream Port), adapter connector (Upstream Port), or in the case of component-to-component connections, the actual wired connection width. All encodings other than the defined encodings are reserved. Multi-Function devices associated with an Upstream Port must report the same value in this field for all functions. In M-PCIe mode, the reset and dynamic values of this field are calculated by the controller. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R Note: This register field is sticky. 01_H **X1**: x1 02_H **X2**: x2 04_H **X4**: x4 08_H **X8**: x8 $0C_H$ **X12**: x12 10_H **X16**: x16 20_H **X32**: x32 **others,** Reserved
PCIE_CAP_ACTIVE_STATE_LINK_PM_SUPPORT	11:10	rw	**Active State Power Management (ASPM) Support** This field indicates the level of ASPM supported on the given PCI Express Link. For more details on ASPM support requirements, see section 5.4.1 of PCI Express Base Specification. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R Note: This register field is sticky. 00_B **NO_ASPM_SUP**: No ASPM Support 01_B **L0S_SUP**: L0s Supported 10_B **L1_SUP**: L1 Supported 11_B **L0S_L1_SUP**: L0s and L1 Supported

(table continues...)

(continued)

Field	Bits	Type	Description
PCIE_CAP_L0S_EXIT_LATENCY	14:12	rwh	**L0s Exit Latency** This field indicates the L0s exit latency for the given PCI Express Link. The value reported indicates the length of time this Port requires to complete transition from L0s to L0. If L0s is not supported, the value is undefined; however, see the Implementation Note "Potential Issues With Legacy Software When L0s is Not Supported" in section 5.4.1.1 of PCI Express Base Specification for the recommended value. Note: Exit latencies may be influenced by PCI Express reference clock configuration depending upon whether a component uses a common or separate reference clock. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R Note: This register field is sticky. 000_B **LESS_THAN_64NS**: Less than 64 ns 001_B **_64NS_TO_128NS**: 64 ns to less than 128 ns 010_B **_128NS_TO_256NS**: 128 ns to less than 256 ns 011_B **_256NS_TO_512NS**: 256 ns to less than 512 ns 100_B **_512NS_TO_1US**: 512 ns to less than 1 µs 101_B **_1US_TO_2US**: 1 µs to less than 2 µs 110_B **_2US_TO_4US**: 2 µs to 4 µs 111_B **GREATER_THAN_4US**: More than 4 µs
PCIE_CAP_L1_EXIT_LATENCY	17:15	rwh	**L1 Exit Latency** This field indicates the L1 exit latency for the given PCI Express Link. The value reported indicates the length of time this Port requires to complete transition from ASPM L1 to L0. If ASPM L1 is not supported, the value is undefined. Note: Exit latencies may be influenced by PCI Express reference clock configuration depending upon whether a component uses a common or separate reference clock. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R Note: This register field is sticky. 000_B **LESS_THAN_1US**: Less than 1 µs 001_B **_1US_TO_2US**: 1 µs to less than 2 µs 010_B **_2US_TO_4US**: 2 µs to less than 4 µs 011_B **_4US_TO_8US**: 4 µs to less than 8 µs 100_B **_8US_TO_16US**: 8 µs to less than 16 µs 101_B **_16US_TO_32US**: 16 µs to less than 32 µs 110_B **_32US_TO_64US**: 32 µs to 64 µs 111_B **GREATER_THAN_64US**: More than 64 µs

(table continues...)

(continued)

Field	Bits	Type	Description
PCIE_CAP_CLOCK_POWER_MAN	18	rh	**Clock Power Management** For Upstream Ports, a value of 1b in this bit indicates that the component tolerates the removal of any reference clock(s) via the "clock request" (CLKREQ#) mechanism when the Link is in the L1 and L2/L3 Ready Link states. A value of 0b indicates the component does not have this capability and that reference clock(s) must not be removed in these Link states. L1 PM Substates defines other semantics for the CLKREQ# signal, which are managed independently of Clock Power Management. This Capability is applicable only in form factors that support "clock request" (CLKREQ#) capability. The reset value of this bit field is decided by the value of PM_CTRL.CLKPMEN. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky.
PCIE_CAP_SURPRISE_DOWN_ERR_REP_CAP	19	rw	**Surprise Down Error Reporting Capable** For a Downstream Port, this bit must be set if the component supports the optional capability of detecting and reporting a Surprise Down error condition. For Upstream Ports and components that do not support this optional capability, the controller hardwires this bit to 0b. Note: This register field is sticky.
PCIE_CAP_DLL_ACTIVE_REP_CAP	20	r	**Data Link Layer Link Active Reporting Capable** For a Downstream Port, the controller hardwires this bit to 1b if the component supports the optional capability of reporting the DL_Active state of the Data Link Control and Management State Machine. For a hot-plug capable Downstream Port (as indicated by the Hot-Plug Capable bit of the Slot Capabilities register) or a Downstream Port that supports Link speeds greater than 5.0 GT/s, the controller hardwires this bit to 1b. For Upstream Ports and components that do not support this optional capability, the controller hardwires this bit to 0b.
PCIE_CAP_LINK_BW_NOT_CAP	21	rw	**Link Bandwidth Notification Capability** A value of 1b indicates support for the Link Bandwidth Notification status and interrupt mechanisms. This capability is required for all Root Ports and Switch Downstream Ports supporting Links wider than x1 and/or multiple Link speeds. This field is not applicable and is Reserved for Endpoints, PCI Express to PCI/PCI-X bridges, and Upstream Ports of Switches. For functions that do not implement the Link Bandwidth Notification Capability the controller hardwires this bit to 0b. Note: This register field is sticky.
PCIE_CAP_ASPM_OPT_COMPLIANCE	22	rw	**ASPM Optionality Compliance** This field must be set to 1b in all functions. Components implemented against certain earlier versions of this specification will have this bit set to 0b. Software is permitted to use the value of this bit to help determine whether to enable ASPM or whether to run ASPM compliance tests. Note: The access attributes of this field are as follows: - Wire: HWINIT (hardware initialized) - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R
RSVDP_23	23	r	**Reserved for future use**

(table continues...)

(continued)

Field	Bits	Type	Description
PCIE_CAP_PORT_NUM	31:24	rw	**Port Number** This field indicates the PCI Express Port number for the given PCI Express Link. Multi-Function Devices associated with an Upstream Port must report the same value in this field for all functions. Note: The access attributes of this field are as follows: - Wire: HWINIT - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R

Table 886 Reset values of LINK_CAPABILITIES_REG

Reset	Reset value	Note
Kernel Reset	0000 0000 0111 1X11 0110 1000 0001 0011_B	

19.4.2.124 Link Control and Link Status Register

This register controls and provides information about PCI Express Link specific parameters.

LINK_CONTROL_LINK_STATUS_REG Offset address: 0004080_H

Link Control and Link Status Register Reset values see: Table 887

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PCIE_CAP_LINK_AUTO_BW_STATUS	**PCIE_CAP_LINK_BW_MAN_STATUS**	**PCIE_CAP_DLL_ACTIVE**	**PCIE_CAP_SLOT_CLK_CONFIG**	**PCIE_CAP_LINK_TRAINING**	**RSVDP_26**	**PCIE_CAP_NEGO_LINK_WIDTH**						**PCIE_CAP_LINK_SPEED**			
w1c	rw1ch	rh	rw	rh	r	rh						rh			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PCIE_CAP_DRS_SIGNALING_CONTROL		**RSVDP_12**		**PCIE_CAP_LINK_AUTO_BW_INT_EN**	**PCIE_CAP_LINK_BW_MAN_INT_EN**	**PCIE_CAP_HW_AUTO_WIDTH_DISABLE**	**PCIE_CAP_EN_CLK_POWER_MAN**	**PCIE_CAP_EXTENDED_SYNCH**	**PCIE_CAP_COMMON_CLK_CONFIG**	**PCIE_CAP_RETRAIN_LINK**	**PCIE_CAP_LINK_DISABLE**	**PCIE_CAP_RCB**	**RSVDP_2**	**PCIE_CAP_ACTIVE_STATE_LINK_PM_CONTROL**	
rw		r		rwh	rwh	rw	rwh	rwh	rw	rwh	rwh	rw	r	rw	

Field	Bits	Type	Description
PCIE_CAP_ACTIVE_STATE_LINK_PM_CONTROL	1:0	rw	**Active State Power Management (ASPM) Control** This field controls the level of ASPM enabled on the given PCI Express Link. See section 5.4.1.3 of PCI Express Base Specification for requirements on when and how to enable ASPM. Note: "L0s Entry Enabled" enables the Transmitter to enter L0s. If L0s is supported, the Receiver must be capable of entering L0s even when the Transmitter is disabled from entering L0s (00b or 10b). ASPM L1 must be enabled by software in the Upstream component on a Link prior to enabling ASPM L1 in the Downstream component on that Link. When disabling ASPM L1, software must disable ASPM L1 in the Downstream component on a Link prior to disabling ASPM L1 in the Upstream component on that Link. ASPM L1 must only be enabled on the Downstream component if both components on a Link support ASPM L1. Software must not enable L0s in either direction on a given Link unless components on both sides of the Link each support L0s; otherwise, the result is undefined. 00_B **DISABLED**: Disabled 01_B **L0S_ENTRY_EN**: L0s Entry Enabled 10_B **L1_ENTRY_En**: L1 Entry Enabled 11_B **L0S_L1_ENTRY_EN**: L0s and L1 Entry Enabled
RSVDP_2	2	r	**Reserved for future use**
PCIE_CAP_RCB	3	rw	**Read Completion Boundary (RCB)** Root Ports: Indicates the RCB value for the Root Port. Refer to section 2.3.1.1 of PCI Express Base Specification for the definition of the parameter RCB. The controller hardwires this bit for a Root Port and returns its RCB support capabilities. Endpoints and Bridges: Optionally set by configuration software to indicate the RCB value of the Root Port Upstream from the Endpoint or Bridge. Refer to Section 2.3.1.1 of PCI Express Base Specification for the definition of the parameter RCB is same as Root Port. Configuration software must only set this bit if the Root Port Upstream from the Endpoint or Bridge reports an RCB value of 128 bytes (a value of 1b in the Read Completion Boundary bit). For functions that do not implement this feature, the controller hardwires this bit to 0b. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W 0_B **_64B**: 64 byte 1_B **_128B**: 128 byte
PCIE_CAP_LINK_DISABLE	4	rwh	**Link Disable** This bit disables the Link by directing the LTSSM to the Disabled state when set; this bit is Reserved on Endpoints, PCI Express to PCI/PCI-X bridges, and Upstream Ports of Switches. Writes to this bit are immediately reflected in the value read from the bit, regardless of actual Link state. After clearing this bit, software must honor timing requirements defined in Section 6.6.1 with respect to the first Configuration Read following a Conventional Reset. Note: The access attributes of this field are as follows: - Wire: dsp=1 ? RW : RO - Dbi: dsp=1? RW : RO

(table continues...)

(continued)

Field	Bits	Type	Description
PCIE_CAP_RETRAIN_LINK	5	rwh	**Retrain Link** A write of 1b to this bit initiates Link retraining by directing the Physical Layer LTSSM to the Recovery state. If the LTSSM is already in Recovery or Configuration, re-entering Recovery is permitted but not required. Reads of this bit always return 0b. It is permitted to write 1b to this bit while simultaneously writing modified values to other fields in this register. If the LTSSM is not already in Recovery or Configuration, the resulting Link training must use the modified values. If the LTSSM is already in Recovery or Configuration, the modified values are not required to affect the Link training that's already in progress. This bit is not applicable and is Reserved for Endpoints, PCI Express to PCI/PCI-X bridges, and Upstream Ports of Switches. This bit always returns 0b when read.
PCIE_CAP_COMMON_CLK_CONFIG	6	rw	**Common Clock Configuration** When set, this bit indicates that this component and the component at the opposite end of this Link are operating with a distributed common reference clock. A value of 0b indicates that this component and the component at the opposite end of this Link are operating with asynchronous reference clock. Components utilize this common clock configuration information to report the correct L0s and L1 Exit Latencies. After changing the value in this bit in both components on a Link, software must trigger the Link to retrain by writing a 1b to the Retrain Link bit of the Downstream Port.
PCIE_CAP_EXTENDED_SYNCH	7	rwh	**Extended Synch** When set, this bit forces the transmission of additional Ordered Sets when exiting the L0s state (see section 4.2.4.5 of PCI Express Base Specification) and when in the Recovery state (see section 4.2.6.4.1 of PCI Express Base Specification). This mode provides external devices (for example, logic analyzers) monitoring the Link time to achieve bit and Symbol lock before the Link enters the L0 state and resumes communication. For Multi-Function devices if any function has this bit set, then the component must transmit the additional Ordered Sets when exiting L0s or when in Recovery.

(table continues...)

(continued)

Field	Bits	Type	Description
PCIE_CAP_EN_CLK_POWER_MAN	8	rwh	**Enable Clock Power Management** Applicable only for Upstream Ports and with form factors that support a "Clock Request" (CLKREQ#) mechanism, this bit operates as follows: . For Downstream Ports and components that do not support Clock Power Management (as indicated by a 0b value in the Clock Power Management bit of the Link Capabilities register), the controller hardwires this bit to 0b. The write value is gated with the PCIE_CAP_CLOCK_POWER_MAN field in LINK_CAPABILITIES_REG. Note: The access attributes of this field are as follows: • Wire: LINK_CAPABILITIES_REG.PCIE_CAP_CLOCK_POWER_MAN ? RWS : ROS • Dbi: LINK_CAPABILITIES_REG.PCIE_CAP_CLOCK_POWER_MAN ? RWS : ROS Note: This register field is sticky 0_B **PM_DISABLE_LOW_CLKREQ**: Clock power management is disabled and device must hold CLKREQ# signal low 1_B **CLKREQ_IN_USE**: When this bit is set, the device is permitted to use CLKREQ# signal to power manage Link clock according to protocol defined in appropriate form factor specification
PCIE_CAP_HW_AUTO_WIDTH_DISABLE	9	rw	**Hardware Autonomous Width Disable** When set, this bit disables hardware from changing the Link width for reasons other than attempting to correct unreliable Link operation by reducing Link width. For components that do not implement the ability autonomously to change Link width, the ciontroller hardwires this bit to 0b. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
PCIE_CAP_LINK_BW_MAN_INT_EN	10	rwh	**Link Bandwidth Management Interrupt Enable** When set, this bit enables the generation of an interrupt to indicate that the Link Bandwidth Management Status bit has been set. The write value is gated with the PCIE_CAP_LINK_BW_NOT_CAP field in LINK_CAPABILITIES_REG. This bit is not applicable and is Reserved for Endpoints, PCI Express-to-PCI/PCI-X bridges, and Upstream Ports of Switches. For functions that do not implement the Link Bandwidth Notification Capability, the controller hardwires this bit to 0b. Note: The access attributes of this field are as follows: • Wire: LINK_CAPABILITIES_REG.PCIE_CAP_LINK_BW_NOT_CAP ? RW : RO • Dbi: LINK_CAPABILITIES_REG.PCIE_CAP_LINK_BW_NOT_CAP ? RW : RO

(table continues...)

(continued)

Field	Bits	Type	Description
PCIE_CAP_LINK_AUTO_BW_INT_EN	11	rwh	**Link Autonomous Bandwidth Management Interrupt Enable** When set, this bit enables the generation of an interrupt to indicate that the Link Autonomous Bandwidth Status bit has been set. The write value is gated with the PCIE_CAP_LINK_BW_NOT_CAP field in LINK_CAPABILITIES_REG. This bit is not applicable and is Reserved for Endpoints, PCI Express-to-PCI/PCI-X bridges, and Upstream Ports of Switches. For functions that do not implement the Link Bandwidth Notification Capability, the controller hardwires this bit to 0b. Note: The access attributes of this field are as follows: • Wire: LINK_CAPABILITIES_REG.PCIE_CAP_LINK_BW_NOT_CAP ? RW : RO • Dbi: LINK_CAPABILITIES_REG.PCIE_CAP_LINK_BW_NOT_CAP ? RW : RO
RSVDP_12	13:12	r	**Reserved for future use**
PCIE_CAP_DRS_SIGNALING_CONTROL	15:14	rw	**DRS Signaling Control** Indicates the mechanism used to report reception of a DRS message. Must be implemented for Downstream Ports with the DRS Supported bit Set in the Link Capabilities 2 Register. Encodings are: • If DRS Supported is set, receiving a DRS Message will set DRS Message Received in the Link Control 2 and Status 2 Register Register but will otherwise have no effect • If the DRS Message Received bit in the Link Control 2 and Status 2 Register Register transitions from 0 to 1, and either MSI or MSI-X is enabled, an MSI or MSI-X interrupt is generated using the vector in Interrupt Message Number (section 7.5.3.2) Behavior is undefined if this field is set to 10b as the FRS capability is not Supported. Behavior is undefined if this field is set to 11b. For Downstream Ports with the DRS Supported bit clear in the Link Capabilities 2 register, the controller hardwires this field to 00b. This field is Reserved for Upstream Ports. 00_B **DRS_NOT_REPORTED**: DRS not Reported 01_B **DRS_INTRRUPT_EN**: DRS Interrupt Enabled 10_B **FRS_EN**: DRS to FRS Signaling Enabled **others,** Reserved

(table continues...)

(continued)

Field	Bits	Type	Description
PCIE_CAP_LINK_SPEED	19:16	rh	**Current Link Speed** This field indicates the negotiated Link speed of the given PCI Express Link. The encoded value specifies a bit location in the Supported Link Speeds Vector (in the Link Capabilities 2 register) that corresponds to the current Link speed. All encodings other than the defined encodings are reserved. The value in this field is undefined when the Link is not up. 1_H **SUP_LINK_SPEED_FIELD_BIT_0**: Supported Link Speeds Vector field bit 0 2_H **SUP_LINK_SPEED_FIELD_BIT_1**: Supported Link Speeds Vector field bit 1 3_H **SUP_LINK_SPEED_FIELD_BIT_2**: Supported Link Speeds Vector field bit 2 4_H **SUP_LINK_SPEED_FIELD_BIT_3**: Supported Link Speeds Vector field bit 3 5_H **SUP_LINK_SPEED_FIELD_BIT_4**: Supported Link Speeds Vector field bit 4 6_H **SUP_LINK_SPEED_FIELD_BIT_5**: Supported Link Speeds Vector field bit 5 7_H **SUP_LINK_SPEED_FIELD_BIT_6**: Supported Link Speeds Vector field bit 6 **others,** Reserved
PCIE_CAP_NEGO_LINK_WIDTH	25:20	rh	**Negotiated Link Width** This field indicates the negotiated width of the given PCI Express Link. All encodings other than the defined encodings are reserved. The value in this field is undefined when the Link is not up. 01_H **X1**: x1 02_H **X2**: x2 04_H **X4**: x4 08_H **X8**: x8 $0C_H$ **X12**: x12 10_H **X16**: x16 20_H **X32**: x32 **others,** Reserved
RSVDP_26	26	r	**Reserved for future use**
PCIE_CAP_LINK_TRAINING	27	rh	**Link Training** This read-only bit indicates that the Physical Layer LTSSM is in the Configuration or Recovery state, or that 1b was written to the Retrain Link bit but Link training has not yet begun. Hardware clears this bit when the LTSSM exits the Configuration/Recovery state. This bit is not applicable and Reserved for Endpoints, PCI Express to PCI/PCI-X bridges, and Upstream Ports of Switches, and the controller hardwires it to 0b. Note: The access attributes of this field are as follows: - Wire: RSVDP - Dbi: R

(table continues...)

(continued)

Field	Bits	Type	Description
PCIE_CAP_SLOT_CLK_CONFIG	28	rw	**Slot Clock Configuration** This bit indicates that the component uses the same physical reference clock that the platform provides on the connector. If the device uses an independent clock irrespective of the presence of a reference clock on the connector, this bit must be clear. For a Multi-Function Device, each Function must report the same value for this bit. Note: The access attributes of this field are as follows: - Wire: HWINIT - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R
PCIE_CAP_DLL_ACTIVE	29	rh	**Data Link Layer Link Active** This bit indicates the status of the Data Link Control and Management State Machine. It returns a 1b to indicate the DL_Active state, 0b otherwise. This bit must be implemented if the Data Link Layer Link Active Reporting Capable bit is 1b. Otherwise, the controller hardwires it to 0b.
PCIE_CAP_LINK_BW_MAN_STATUS	30	rw1ch	**Link Bandwidth Management Status** This bit is set by hardware to indicate that either of the following has occurred without the Port transitioning through DL_Down status: • A Link retraining has completed following a write of 1b to the Retrain Link bit. Note: This bit is set following any write of 1b to the Retrain Link bit, including when the Link is in the process of retraining for some other reason • Hardware has changed Link speed or width to attempt to correct unreliable Link operation, either through an LTSSM timeout or a higher level process. This bit must be set if the Physical Layer reports a speed or width change was initiated by the Downstream component that was not indicated as an autonomous change. This bit is not applicable and is Reserved for Endpoints, PCI Express-to-PCI/PCI-X bridges, and Upstream Ports of Switches. The default value of this bit is 0b. The write value is gated with the PCIE_CAP_LINK_BW_NOT_CAP field in LINK_CAPABILITIES_REG. Note: The access attributes of this field are as follows: - Wire: RSVDP - Dbi: R Software write with 1 clears this bit-field.
PCIE_CAP_LINK_AUTO_BW_STATUS	31	w1c	**Link Autonomous Bandwidth Status** This bit is set by hardware to indicate that hardware has autonomously changed Link speed or width, without the Port transitioning through DL_Down status, for reasons other than to attempt to correct unreliable Link operation. This bit must be set if the Physical Layer reports a speed or width change was initiated by the Downstream component that was indicated as an autonomous change. The write value is gated with the PCIE_CAP_LINK_BW_NOT_CAP field in LINK_CAPABILITIES_REG. This bit is not applicable and is Reserved for Endpoints, PCI Express-to-PCI/PCI-X bridges, and Upstream Ports of Switches. For functions that do not implement the Link Bandwidth Notification Capability, the controller hardwires this bit to 0b. Note: The access attributes of this field are as follows: - Wire: RSVDP - Dbi: R Note: This register field is sticky. Software write with 1 clears this bit field.

Table 887 Reset values of LINK_CONTROL_LINK_STATUS_REG

Reset	Reset value	Note
Kernel Reset	XXX1 X0XX XXXX XXXX 0000 XX0X 00XX 1000_B	

19.4.2.125 Slot Capabilities Register

The Slot Capabilities register identifies PCI Express slot specific capabilities.

SLOT_CAPABILITIES_REG Offset address: 0004084_H

Slot Capabilities Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PCIE_CAP_PHY_SLOT_NUM													PCIE_CAP_NO_CMD_CPL_SUPPORT	PCIE_CAP_ELECTROMECH_INTERLOCK	PCIE_CAP_SLOT_POWER_LIMIT_SCALE
rw													rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PCIE_CAP_SLOT_POWER_LIMIT_SCALE	PCIE_CAP_SLOT_POWER_LIMIT_VALUE								PCIE_CAP_HOT_PLUG_CAPABLE	PCIE_CAP_HOT_PLUG_SURPRISE	PCIE_CAP_POWER_INDICATOR	PCIE_CAP_ATTENTION_INDICATOR	PCIE_CAP_MRL_SENSOR	PCIE_CAP_POWER_CONTROLLER	PCIE_CAP_ATTENTION_INDICATOR_BUTTON
rw	rw								rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
PCIE_CAP_ATTENTION_INDICATOR_BUTTON	0	rw	**Attention Button Present** Slot attention capability is not present in this implementation. When set, this bit indicates that an Attention Button for this slot is electrically controlled by the chassis. Note: The access attributes of this field are as follows: - Wire: No access. - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R
PCIE_CAP_POWER_CONTROLLER	1	rw	**Power Controller Present** Slot power controller capability is not present in this implementation. When set, this bit indicates that a software programmable Power Controller is implemented for this slot/adapter (depending on form factor). Note: The access attributes of this field are as follows: - Wire: No access. - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R

(table continues...)

(continued)

Field	Bits	Type	Description
PCIE_CAP_MRL_SENSOR	2	rw	**MRL Sensor Present** When set, this bit indicates that an MRL Sensor is implemented on the chassis for this slot. Note: The access attributes of this field are as follows: - Wire: No access. - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R
PCIE_CAP_ATTENTION_INDICATOR	3	rw	**Attention Indicator Present** Power attention indicator capability is not supported in this implementation. When set, this bit indicates that an Attention Indicator is electrically controlled by the chassis. Note: The access attributes of this field are as follows: - Wire: No access. - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R
PCIE_CAP_POWER_INDICATOR	4	rw	**Power Indicator Present** Power Indicator capability is not supported in this implementation. When set, this bit indicates that a Power Indicator is electrically controlled by the chassis for this slot. Note: The access attributes of this field are as follows: - Wire: No access. - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R
PCIE_CAP_HOT_PLUG_SURPRISE	5	rw	**Hot-Plug Surprise** Hot-Plug surprise capability is not supported in this implementation. When set, this bit indicates that an adapter present in this slot might be removed from the system without any prior notification. This is a form factor specific capability. This bit is an indication to the operating system to allow for such removal without impacting continued software operation. Note: The access attributes of this field are as follows: - Wire: No access. - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R
PCIE_CAP_HOT_PLUG_CAPABLE	6	rw	**Hot-Plug Capable** Hot plug capabilities are not supported in this implementation. When set, this bit indicates that this slot is capable of supporting hot-plug operations. Note: The access attributes of this field are as follows: - Wire: No access. - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R
PCIE_CAP_SLOT_POWER_LIMIT_VALUE	14:7	rw	**Slot Power Limit Value** In combination with the Slot Power Limit Scale value, specifies the upper limit on power supplied by the slot (for more detais, see Section 6.9 of PCI Express Base Specification) or by other means to the adapter. Power limit (in Watts) is calculated by multiplying the value in this field by the value in the Slot Power Limit Scale field except when the Slot Power Limit Scale field equals 00b (1.0x) and Slot Power Limit Value exceeds EFh, the alternative encodings defined in Values: are used. Value F3h to FFh are Reserved for Slot Power Limit values above 300 W. This register must be implemented if the Slot Implemented bit is set. Writes to this register also cause the Port to send the Set_Slot_Power_Limit Message. The default value prior to hardware/ firmware initialization is 0000 0000b. Note: The access attributes of this field are as follows: - Wire: No access. - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R $F0_H$ **_250W_LIMIT**: 250 W Slot Power Limit $F1_H$ **_275W_LIMIT**: 275 W Slot Power Limit $F2_H$ **_300W_LIMIT**: 300 W Slot Power Limit **others,** Reserved

(table continues...)

(continued)

Field	Bits	Type	Description
PCIE_CAP_SLOT_POWER_LIMIT_SCALE	16:15	rw	**Slot Power Limit Scale** Specifies the scale used for the Slot Power Limit Value (for more details, see Section 6.9 of PCI Express Base Specification). This register must be implemented if the Slot Implemented bit is set. Writes to this register also cause the Port to send the Set_Slot_Power_Limit Message. The default value prior to hardware/firmware initialization is 00b. Note: The access attributes of this field are as follows: - Wire: No access. - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R 00_B **_1_0X**: 1.0x 01_B **_0_1X**: 0.1x 10_B **_0_01X**: 0.01x 11_B **_0_001X**: 0.001x
PCIE_CAP_ELECTROMECH_INTERLOCK	17	rw	**Electromechanical Interlock Present** Electromechanical Interlock capability is not supported in this implementation. When set, this bit indicates that an Electromechanical Interlock is implemented on the chassis for this slot. Note: The access attributes of this field are as follows: - Wire: No access. - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R
PCIE_CAP_NO_CMD_CPL_SUPPORT	18	rw	**No Command Completed Support** When set, this bit indicates that this slot does not generate software notification when an issued command is completed by the Hot-Plug Controller. This bit is only permitted to be set if the hot-plug capable Port is able to accept writes to all fields of the Slot Control register without delay between successive writes. Note: The access attributes of this field are as follows: - Wire: No access. - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R
PCIE_CAP_PHY_SLOT_NUM	31:19	rw	**Physical Slot Number** This field indicates the physical slot number attached to this Port. This field must be hardware initialized to a value that assigns a slot number that is unique within the chassis, regardless of the form factor associated with the slot. This field must be initialized to zero for Ports connected to devices that are either integrated on the system board or integrated within the same silicon as the Switch device or Root Port. Note: The access attributes of this field are as follows: - Wire: No access. - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R

19.4.2.126 Slot Control and Status Register

This register controls and provides information about PCI Express Slot specific parameters.

SLOT_CONTROL_SLOT_STATUS Offset address: 0004088_H

Slot Control and Status Register Reset values see: Table 888

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_25							PCIE_CAP_DLL_STATE_CHANGED	PCIE_CAP_ELECTROMECH_INTERLOCK_STATUS	PCIE_CAP_PRESENCE_DETECT_STATE	PCIE_CAP_MRL_SENSOR_STATE	PCIE_CAP_CMD_CPLD	PCIE_CAP_PRESENCE_DETECTED_CHANGED	PCIE_CAP_MRL_SENSOR_CHANGED	PCIE_CAP_POWER_FAULT_DETECTED	PCIE_CAP_ATTENTION_BUTTON_PRESSED
r							w1c	r	rh	r	w1c	w1c	w1c	w1c	w1c

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVDP_14		0	PCIE_CAP_DLL_STATE_CHANGED_EN	PCIE_CAP_ELECTROMECH_INTERLOCK_CTRL	PCIE_CAP_POWER_CONTROLLER_CTRL	PCIE_CAP_POWER_INDICATOR_CTRL		PCIE_CAP_ATTENTION_INDICATOR_CTRL		PCIE_CAP_HOT_PLUG_INT_EN	PCIE_CAP_CMD_CPL_INT_EN	PCIE_CAP_PRESENCE_DETECT_CHANGE_EN	PCIE_CAP_MRL_SENSOR_CHANGED_EN	PCIE_CAP_POWER_FAULT_DETECTED_EN	PCIE_CAP_ATTENTION_BUTTON_PRESSED_EN
r		r	rw	rwh	rw	rw		rw		rw	rwh	rw	rw	rw	rw

Field	Bits	Type	Description
PCIE_CAP_ATTENTION_BUTTON_PRESSED_EN	0	rw	**Attention Button Pressed Enable** When set to 1b, this bit enables software notification on an attention button pressed event (for more details, see Section 6.7.3 of PCI Express Base Specification). If the Attention Button Present bit in the Slot Capabilities register is 0b, this bit is permitted to be read-only with a value of 0b.
PCIE_CAP_POWER_FAULT_DETECTED_EN	1	rw	**Power Fault Detected Enable** When set, this bit enables software notification on a power fault event (for more details, see Section 6.7.3 of PCI Express Base Specification). If a Power Controller that supports power fault detection is not implemented, this bit is permitted to be read-only with a value of 0b.
PCIE_CAP_MRL_SENSOR_CHANGED_EN	2	rw	**MRL Sensor Changed Enable** When set, this bit enables software notification on a MRL sensor changed event (for more details, see Section 6.7.3 of PCI Express Base Specification). If the MRL Sensor Present bit in the Slot Capabilities register is Clear, this bit is permitted to be read-only with a value of 0b.
PCIE_CAP_PRESENCE_DETECT_CHANGE_EN	3	rw	**Presence Detect Changed Enable** When set, this bit enables software notification on a presence detect changed event (for more details, see Section 6.7.3 of PCI Express Base Specification). If the Hot-Plug Capable bit in the Slot Capabilities register is 0b, this bit is permitted to be read-only with a value of 0b.

(table continues...)

(continued)

Field	Bits	Type	Description
PCIE_CAP_CMD_CPL_INT_EN	4	rwh	**Command Completed Interrupt Enable** If Command Completed notification is supported (if the No Command Completed Support bit in the Slot Capabilities register is 0b), when set, this bit enables software notification when a hot-plug command is completed by the Hot-Plug Controller. If Command Completed notification is not supported, the controller hardwires this bit must to 0b. Write value is gated with PCIE_CAP_NO_CMD_CPL_SUPPORT field in SLOT_CAPABILITIES_REG. Note: The access attributes of this field are as follows: - Wire: No access. - Dbi: SLOT_CAPABILITIES_REG.PCIE_CAP_NO_CMD_CPL_SUPPORT ? RO : RW
PCIE_CAP_HOT_PLUG_INT_EN	5	rw	**Hot-Plug Interrupt Enable** When set, this bit enables generation of an interrupt on enabled hot-plug events. Since the Hot-Plug capable bit in the Slot Capabilities register is clear, this bit is to be considered as read-only with a value of 0b.
PCIE_CAP_ATTENTION_INDICATOR_CTRL	7:6	rw	**Attention Indicator Control** If an Attention Indicator is implemented, writes to this field set the Attention Indicator to the written state. Reads of this field must reflect the value from the latest write, even if the corresponding hot-plug command is not complete, unless software issues a write without waiting, if required to, for the previous command to complete in which case the read value is undefined. Note: The default value of this field must be one of the non-Reserved values. If the Attention Indicator Present bit in the Slot Capabilities register is 0b, this bit is permitted to be read-only with a value of 00b. 00_B **RESERVED**: Reserved 01_B **ON**: On 10_B **BLINK**: Blink 11_B **OFF**: Off
PCIE_CAP_POWER_INDICATOR_CTRL	9:8	rw	**Power Indicator Control** If a Power Indicator is implemented, writes to this field set the Power Indicator to the written state. Reads of this field must reflect the value from the latest write, even if the corresponding hot-plug command is not complete, unless software issues a write without waiting, if required to, for the previous command to complete in which case the read value is undefined. Note: The default value of this field must be one of the non-Reserved values. If the Power Indicator Present bit in the Slot Capabilities register is 0b, this bit is permitted to be read-only with a value of 00b. 00_B **RESERVED**: Reserved 01_B **ON**: On 10_B **BLINK**: Blink 11_B **OFF**: Off

(table continues...)

(continued)

Field	Bits	Type	Description
PCIE_CAP_POWER_CONTROLLER_CTRL	10	rw	**Power Controller Control** If a Power Controller is implemented, this bit when written sets the power state of the slot per the defined encodings. Reads of this bit must reflect the value from the latest write, even if the corresponding hot-plug command is not complete, unless software issues a write, if required to, without waiting for the previous command to complete in which case the read value is undefined. Note: In some cases the power controller may autonomously remove slot power or not respond to a power-up request based on a detected fault condition, independent of the Power Controller Control setting. If the Power Controller Present bit in the Slot Capabilities register is clear, then writes to this bit have no effect and the read value of this bit is undefined. 0_B **PWR_ON**: Power On 1_B **PWR_OFF**: Power Off
PCIE_CAP_ELECTROMECH_INTERLOCK_CTRL	11	rwh	**Electromechanical Interlock Control** If an Electromechanical Interlock is implemented, a write of 1b to this bit causes the state of the interlock to toggle. A write of 0b to this bit has no effect. A read of this bit always returns a 0b.
PCIE_CAP_DLL_STATE_CHANGED_EN	12	rw	**Data Link Layer State Changed Enable** If the Data Link Layer Link Active Reporting capability is 1b, this bit enables software notification when Data Link Layer Link Active bit is changed. If the Data Link Layer Link Active Reporting Capable bit is 0b, this bit is permitted to be read-only with a value of 0b.
RSVDP_14	15:14	r	**Reserved for future use**
PCIE_CAP_ATTENTION_BUTTON_PRESSED	16	w1c	**Attention Button Pressed** If an Attention Button is implemented, this bit is set when the attention button is pressed. If an Attention Button is not supported, this bit must not be set.
PCIE_CAP_POWER_FAULT_DETECTED	17	w1c	**Power Fault Detected** If a Power Controller that supports power fault detection is implemented, this bit issSet when the Power Controller detects a power fault at this slot. Note: Depending on hardware capability, it is possible that a power fault can be detected at any time, independent of the Power Controller Control setting or the occupancy of the slot. If power fault detection is not supported, this bit must not be set.
PCIE_CAP_MRL_SENSOR_CHANGED	18	w1c	**MRL Sensor Changed** If an MRL sensor is implemented, this bit is set when a MRL Sensor state change is detected. If an MRL sensor is not implemented, this bit must not be set.
PCIE_CAP_PRESENCE_DETECTED_CHANGED	19	w1c	**Presence Detect Changed** This bit is set when the value reported in the Presence Detect State bit is changed.

(table continues...)

(continued)

Field	Bits	Type	Description
PCIE_CAP_CMD_CPLD	20	w1c	**Command Completed** If Command Completed notification is supported (if the No Command Completed Support bit in the Slot Capabilities register is 0b), this bit is set when a hot-plug command has completed and the Hot-Plug Controller is ready to accept a subsequent command. The Command Completed status bit is set as an indication to host software that the Hot-Plug Controller has processed the previous command and is ready to receive the next command; it provides no guarantee that the action corresponding to the command is complete. If Command Completed notification is not supported, the controller hardwires this bit to 0b.
PCIE_CAP_MRL_SENSOR_STATE	21	r	**MRL Sensor State** This bit reports the status of the MRL sensor if implemented. Encodings are define as above. 0_B **MRL_CLOSED**: MRL Closed 1_B **MRL_OPEN**: MRL Open
PCIE_CAP_PRESENCE_DETECT_STATE	22	rh	**Presence Detect State** This bit indicates the presence of an adapter in the slot, reflected by the logical "OR" of the Physical Layer in-band presence detect mechanism and, if present, any out-of-band presence detect mechanism defined for the slot's corresponding form factor. Note that the in-band presence detect mechanism requires that power be applied to an adapter for its presence to be detected. Consequently, form factors that require a power controller for hot-plug must implement a physical pin presence detect mechanism. This bit must be implemented on all Downstream Ports that implement slots. For Downstream Ports not connected to slots (where the Slot Implemented bit of the PCI Express Capabilities register is 0b), the controller hardwires this bit to 1b. Note: The access attributes of this field are as follows: - Wire: No access. - Dbi: R 0_B **EMPTY_SLOT**: Slot Empty 1_B **ADAPTER_PRESENT_IN_SLOT**: Adapter Present in slot
PCIE_CAP_ELECTROMECH_INTERLOCK_STATUS	23	r	**Electromechanical Interlock Status** If an Electromechanical Interlock is implemented, this bit indicates the status of the Electromechanical Interlock. 0_B **DISENGAGED**: Electromechanical Interlock Disengaged 1_B **ENGAGED**: Electromechanical Interlock Engaged
PCIE_CAP_DLL_STATE_CHANGED	24	w1c	**Data Link Layer State Changed** This bit is set when the value reported in the Data Link Layer Link Active bit of the Link Status register is changed. In response to a Data Link Layer State Changed event, software must read the Data Link Layer Link Active bit of the Link Status register to determine if the Link is active before initiating configuration cycles to the hot plugged device.
RSVDP_25	31:25	r	**Reserved for future use**
0	13	r	**Reserved** Read as 0; should be written with 0.

Table 888 Reset values of SLOT_CONTROL_SLOT_STATUS

Reset	Reset value	Note
Kernel Reset	0000 0000 0X00 0000 0000 0011 110X 0000$_B$	

19.4.2.127 Root Control and Capabilities Register

This register controls PCI Express Root Complex specific parameters and identifies PCI Express Root Port specific capabilities.

ROOT_CONTROL_ROOT_CAPABILITIES_REG Offset address: 000408C$_H$

Root Control and Capabilities Register Kernel Reset value: 0001 0000$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_17															PCIE_CAP_CRS_SW_VISIBILITY
r															rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVDP_5											PCIE_CAP_CRS_SW_VISIBILITY_EN	PCIE_CAP_PME_INT_EN	PCIE_CAP_SYS_ERR_ON_FATAL_ERR_EN	PCIE_CAP_SYS_ERR_ON_NON_FATAL_ERR_EN	PCIE_CAP_SYS_ERR_ON_CORR_ERR_EN
r											rw	rw	rw	rw	rw

Field	Bits	Type	Description
PCIE_CAP_SYS_ERR_ON_CORR_ERR_EN	0	rw	**System Error on Correctable Error Enable** If set, this bit indicates that a System Error should be generated if a correctable error (ERR_COR) is reported by any of the devices in the Hierarchy Domain associated with this Root Port, or by the Root Port itself. The mechanism for signaling a System Error to the system is system specific.
PCIE_CAP_SYS_ERR_ON_NON_FATAL_ERR_EN	1	rw	**System Error on Non-Fatal Error Enable** If set, this bit indicates that a System Error should be generated if a Non-fatal error (ERR_NONFATAL) is reported by any of the devices in the Hierarchy Domain associated with this Root Port, or by the Root Port itself. The mechanism for signaling a System Error to the system is system specific.

(table continues...)

(continued)

Field	Bits	Type	Description
PCIE_CAP_SYS_ERR_ON_FATAL_ERR_EN	2	rw	**System Error on Fatal Error Enable** If set, this bit indicates that a System Error should be generated if a Fatal error (ERR_FATAL) is reported by any of the devices in the Hierarchy Domain associated with this Root Port, or by the Root Port itself. The mechanism for signaling a System Error to the system is system specific.
PCIE_CAP_PME_INT_EN	3	rw	**PME Interrupt Enable** When set, this bit enables PME interrupt generation upon receipt of a PME Message as reflected in the PME Status bit (for more details, see Table 7-29 of PCI Express Base Specification). A PME interrupt is also generated if the PME Status bit is set when this bit is changed from clear to set.
PCIE_CAP_CRS_SW_VISIBILITY_EN	4	rw	**CRS Software Visibility Enable** When set, this bit enables the Root Port to return Configuration Request Retry Status (CRS) Completion Status to software (for more details, see section 2.3.1 of PCI Express Base Specification). For Root Ports that do not implement this capability, the controller hardwires this bit to 0b. Note: The access attributes of this field are as follows: - Wire: No access. - Dbi: ROOT_CONTROL_ROOT_CAPABILITIES_REG.PCIE_CAP_CRS_SW_VISIBILITY ? RW : RO
RSVDP_5	15:5	r	**Reserved for future use**
PCIE_CAP_CRS_SW_VISIBILITY	16	rw	**CRS Software Visibility Capable** When set, this bit indicates that the Root Port is capable of returning Configuration Request Retry Status (CRS) Completion Status to software (for more details, see section 2.3.1 of PCI Express Base Specification). Note: The access attributes of this field are as follows: - Wire: No access. - Dbi: if (DBI_RO_WR_EN == 1) then R/W (Sticky) else R (Sticky) Note: This register field is sticky.
RSVDP_17	31:17	r	**Reserved for future use**

19.4.2.128 Root Status Register

The Root Status register provides information about PCI Express device specific parameters.

ROOT_STATUS_REG Offset address: 0004090_H

Root Status Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_18														PCIE_CAP_PME_PENDING	PCIE_CAP_PME_STATUS
r														r	w1c

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PCIE_CAP_PME_REQ_ID															
r															

Field	Bits	Type	Description
PCIE_CAP_PME_REQ_ID	15:0	r	**PME Requester ID** This field indicates the PCI Requester ID of the last PME Requester. This field is only valid when the PME Status bit is set.
PCIE_CAP_PME_STATUS	16	w1c	**PME Status** This bit indicates that PME was asserted by the PME Requester indicated in the PME Requester ID field. Subsequent PMEs are kept pending until the status register is cleared by software by writing a 1b.
PCIE_CAP_PME_PENDING	17	r	**PME Pending** This bit indicates that another PME is pending when the PME Status bit is s et. When the PME Status bit is cleared by software; the PME is delivered by hardware by setting the PME Status bit again and updating the PME Requester ID field appropriately. The PME Pending bit is cleared by hardware if no more PMEs are pending.
RSVDP_18	31:18	r	**Reserved for future use**

19.4.2.129 Device Capabilities 2 Register

This register identifies PCI Express device specific capabilities; in addition to the Device Capabilities Register.

DEVICE_CAPABILITIES2_REG Offset address: 0004094_H

Device Capabilities 2 Register Kernel Reset value: $000C\ 0C1F_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	RSVDP_24							0				PCIE_CAP_OBFF_SUPPORT		PCIE_CAP2_10_BIT_TAG_REQ_SUPPORT	PCIE_CAP2_10_BIT_TAG_COMP_SUPPORT
r	r							r				rw		r	r

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		PCIE_CAP_TPH_CMPLT_SUPPORT_1	PCIE_CAP_TPH_CMPLT_SUPPORT_0	PCIE_CAP_LTR_SUPP	PCIE_CAP_NO_RO_EN_PR2PR_PAR	PCIE_CAP_128_CAS_CPL_SUPP	PCIE_CAP_64_ATOMIC_CPL_SUPP	PCIE_CAP_32_ATOMIC_CPL_SUPP	PCIE_CAP_ATOMIC_ROUTING_SUPP	PCIE_CAP_ARI_FORWARD_SUPPORT	PCIE_CAP_CPL_TIMEOUT_DISABLE_SUPPORT	PCIE_CAP_CPL_TIMEOUT_RANGE			
r		r	r	rw	r	r	r	r	r	r	r	r			

Field	Bits	Type	Description
PCIE_CAP_CPL_TIMEOUT_RANGE	3:0	r	**Completion Timeout Ranges Supported** This field indicates device Function support for the optional Completion Timeout programmability mechanism. This mechanism allows system software to modify the Completion Timeout value. This field is applicable only to Root Ports, Endpoints that issue Requests on their own behalf, and PCI Express to PCI/PCI-X Bridges that take ownership of Requests issued on PCI Express. For all other Functions this field is Reserved and must be hardwired to 0000b. Four time value ranges are defined: • Range A: 50 µs to 10 ms • Range B: 10 ms to 250 ms • Range C: 250 ms to 4 s • Range D: 4 s to 64 s Bits are set according to the list below to show timeout value ranges supported. All encodings other than the defined encodings are reserved. It is strongly recommended that the Completion Timeout mechanism not expire in less than 10 ms. 0_H **NOT_SUP**: Completion Timeout programming not supported, the Function must implement a timeout value in the range 50 µs to 50 ms 1_H **RANGE_A**: Range A 2_H **RANGE_B**: Range B 3_H **RANGE_A_B**: Ranges A and B 6_H **RANGE_B_C**: Ranges B and C 7_H **RANGE_A_B_C**: Ranges A, B, and C E_H **RANGE_B_C_D**: Ranges B, C, and D F_H **RANGE_A_B_C_D**: Ranges A, B, C, and D
PCIE_CAP_CPL_TIMEOUT_DISABLE_SUPPORT	4	r	**Completion Timeout Disable Supported** A value of 1b indicates support for the Completion Timeout Disable mechanism. The Completion Timeout Disable mechanism is required for Endpoints that issue Requests on their own behalf and PCI Express to PCI/PCI-X Bridges that take ownership of Requests issued on PCI Express. This mechanism is optional for Root Ports. For all other Functions this field is Reserved and the controller hardwires this bit to 0b.
PCIE_CAP_ARI_FORWARD_SUPPORT	5	r	**ARI Forwarding Supported** Alternate routing ID capability is not available in this implementation. Applicable only to Switch Downstream Ports and Root Ports; must be 0b for other Function types. This bit must be set to 1b if a Switch Downstream Port or Root Port supports this optional capability. For more details, see section 6.13 of PCI Express Base Specification.
PCIE_CAP_ATOMIC_ROUTING_SUPP	6	r	**AtomicOp Routing Supported** Applicable only to Switch Upstream Ports, Switch Downstream Ports, and Root Ports; must be 0b for other Function types. This bit must be set to 1b if the Port supports this optional capability. For more details, see section 6.15 of PCI Express Base Specification.

(table continues...)

(continued)

Field	Bits	Type	Description
PCIE_CAP_32_ATOMIC_CPL_SUPP	7	r	**32-bit AtomicOp Completer Supported** Applicable to Functions with Memory Space BARs as well as all Root Ports; must be 0b otherwise. Includes FetchAdd, Swap, and CAS AtomicOps. This bit must be set to 1b if the Function supports this optional capability. For more details on additional RC requirements, see section 6.15.3.1 of PCI Express Base Specification.
PCIE_CAP_64_ATOMIC_CPL_SUPP	8	r	**64-bit AtomicOp Completer Supported** Applicable to Functions with Memory Space BARs as well as all Root Ports; must be 0b otherwise. Includes FetchAdd, Swap, and CAS AtomicOps. This bit must be set to 1b if the Function supports this optional capability. For more details on additional RC requirements, see section 6.15.3.1 of PCI Express Base Specification.
PCIE_CAP_128_CAS_CPL_SUPP	9	r	**128-bit CAS Completer Supported** Applicable to Functions with Memory Space BARs as well as all Root Ports; must be 0b otherwise. This bit must be set to 1b if the Function supports this optional capability. For more details, see section 6.15 of PCI Express Base Specification.
PCIE_CAP_NO_RO_EN_PR2PR_PAR	10	r	**No RO-enabled PR-PR Passing** If this bit is set, the routing element never carries out the passing permitted by Table 2-39 of PCI Express Base Specification entry A2b that is associated with the Relaxed Ordering Attribute field being Set. This bit applies only for Switches and RCs that support peer-to-peer traffic between Root Ports. This bit applies only to Posted Requests being forwarded through the Switch or RC and does not apply to traffic originating or terminating within the Switch or RC itself. All Ports on a Switch or RC must report the same value for this bit. For all other functions, this bit must be 0b.
PCIE_CAP_LTR_SUPP	11	rw	**LTR Mechanism Supported** A value of 1b indicates support for the optional Latency Tolerance Reporting (LTR) mechanism. Root Ports, Switches and Endpoints are permitted to implement this capability. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W(Sticky) else R(Sticky) Note: This register field is sticky.
PCIE_CAP_TPH_CMPLT_SUPPORT_0	12	r	**TPH Completer Supported Bit 0** TLP Processing Hints(TPH) is not supported in this implementation. Value of this bit along with TPH Completer Supported Bit 1 indicates Completer support for TPH or Extended TPH. Applicable only to Root Ports and Endpoints. For all other Functions, this field is Reserved. For more details, see section 6.17 of PCI Express Base Specification. 0_B **TPH_EXT_TPH_CMPL_NOT_SUP_OR_RSVD**: TPH and Extended TPH Completer not supported (if TPH Completer Supported Bit is 0) or Reserved ((if TPH Completer Supported Bit is 1) 1_B **TPH_SUP_EXT_TPH_NOT_SUP_OR_BOTH_SUPP**: TPH Completer supported; Extended TPH Completer not supported(if TPH Completer Supported Bit is 0) or Both TPH and Extended TPH Completer supported ((if TPH Completer Supported Bit is 1)
PCIE_CAP_TPH_CMPLT_SUPPORT_1	13	r	**TPH Completer Supported Bit 1** TLP Processing Hints(TPH) is not supported in this implementation

(table continues...)

(continued)

Field	Bits	Type	Description
PCIE_CAP2_10_BIT_TAG_COMP_SUPPORT	16	r	**10-Bit Tag Completer Supported** If this bit is set, the Function supports 10-Bit Tag Completer capability; otherwise, the Function does not. For more details, see section 2.2.6.2. of PCI Express Base Specification.
PCIE_CAP2_10_BIT_TAG_REQ_SUPPORT	17	r	**10-Bit Tag Requester Supported** If this bit is set, the Function supports 10-Bit Tag Requester capability; otherwise, the Function does not. This bit must not be set if the 10-Bit Tag Completer Supported bit is clear. Note: 10-Bit Tag field generation must be enabled by the 10-Bit Tag Requester Enable bit in the Device Control 2 register of the Requester Function before 10-Bit Tags can be generated by the Requester. For more details, see section 2.2.6.2. of PCI Express Base Specification.
PCIE_CAP_OBFF_SUPPORT	19:18	rw	**OBFF Supported** This field indicates if OBFF is supported and, if so, what signaling mechanism is used. The value reported in this field must indicate support for WAKE# signaling only if: • for a Downstream Port, driving the WAKE# signal for OBFF is supported and the connector or component connected Downstream is known to receive that same WAKE# signal • for an Upstream Port, receiving the WAKE# signal for OBFF is supported and, if the component is on an add-in-card, that the component is connected to the WAKE# signal on the connector. Root Ports, Switch Ports, and Endpoints are permitted to implement this capability. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W(Sticky) else R(Sticky) Note: This register field is sticky. 00_B **NOT_SUP**: OBFF Not Supported 01_B **SUP_USING_MSG**: OBFF supported using Message signaling only 10_B **SUP_USING_WAKE**: OBFF supported using WAKE# signaling only 11_B **SUP_USING_MSG_AND_WAKE**: OBFF supported using WAKE# and Message signaling
RSVDP_24	30:24	r	**Reserved for future use**
0	15:14, 23:20, 31	r	**Reserved** Read as all 0's; should be written with all 0's.

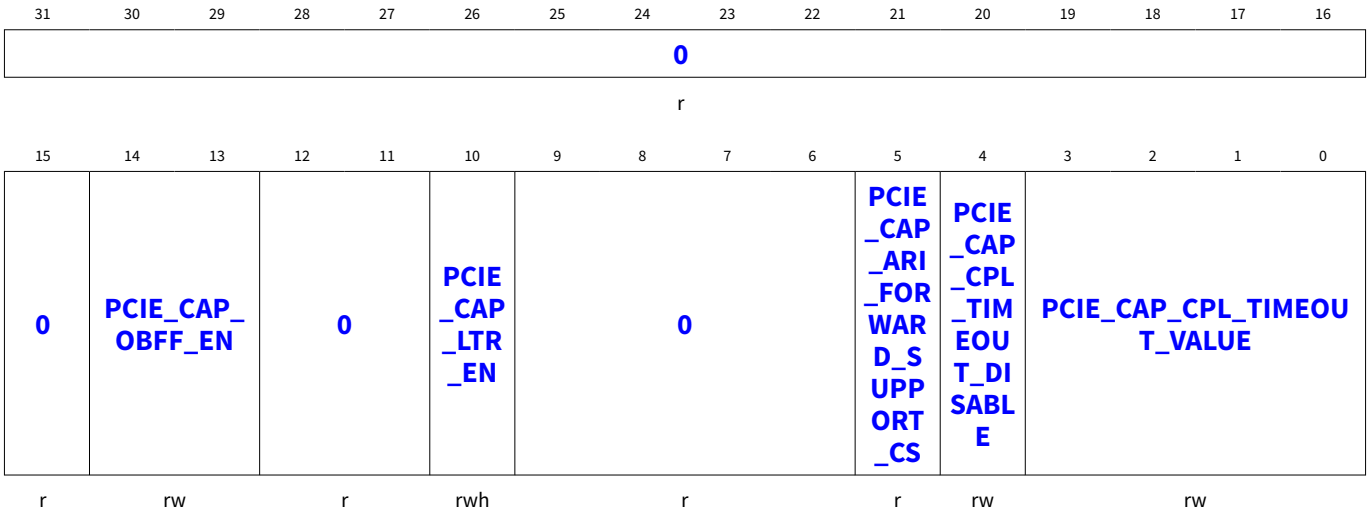
19.4.2.130 Device Control 2 and Status 2 Register

This register controls PCI Express device specific parameters and provides information about PCI Express device (function) specific parameters; in addition to the Device Control and Device Status Register.

DEVICE_CONTROL2_DEVICE_STATUS2_REG Offset address: 0004098_H

Device Control 2 and Status 2 Register Reset values see: Table 889

19 Peripheral Component Interconnect Express (PCIe)



Field	Bits	Type	Description
PCIE_CAP_CPL_TIMEOUT_VALUE	3:0	rw	**Completion Timeout Value** In device Functions that support Completion Timeout programmability, this field allows system software to modify the Completion Timeout value. This field is applicable to Root Ports, Endpoints that issue Requests on their own behalf. Functions that support Completion Timeout programmability must support the values given below corresponding to the programmability ranges indicated in the Completion Timeout Ranges Supported field. All encodings other than the defined encodings are reserved. It is strongly recommended that the Completion Timeout mechanism not expire in less than 10 ms. Software is permitted to change the value in this field at any time. For Requests already pending when the Completion Timeout Value is changed, hardware is permitted to use either the new or the old value for the outstanding Requests, and is permitted to base the start time for each Request either on when this value was changed or on when each request was issued. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W 0_H **DEFAULT**: Default range: 50 µs to 50 ms 1_H **_50_TO_100_US**: 50 µs to 100 µs 2_H **_1_TO_10_MS**: 1 ms to 10 ms 5_H **_16_TO_55_MS**: 16 ms to 55 ms 6_H **_65_TO_210_MS**: 65 ms to 210 ms 9_H **_260_TO_900_MS**: 260 ms to 900 ms A_H **_1_TO_3_5_S**: 1 s to 3.5 s D_H **_4_TO_13_S**: 4 s to 13 s E_H **_17_TO_64_S**: 17 s to 64 s **others,** Reserved

(table continues...)

(continued)

Field	Bits	Type	Description
PCIE_CAP_CPL_TIMEOUT_DISABLE	4	rw	**Completion Timeout Disable** When set, this bit disables the Completion Timeout mechanism. This bit is required for all Functions that support the Completion Timeout Disable Capability. Functions that do not support this optional capability are permitted to hardwire this bit to 0b Software is permitted to set or clear this bit at any time. When set, the Completion Timeout detection mechanism is disabled. If there are outstanding Requests when the bit is cleared, it is permitted but not required for hardware to apply the completion timeout mechanism to the outstanding Requests. If this is done, it is permitted to base the start time for each Request on either the time this bit was cleared or the time each Request was issued.
PCIE_CAP_ARI_FORWARD_SUPPORT_CS	5	r	**ARI Forwarding Enable** Alternate routing ID capability is not available in this implementation. When set, the Downstream Port disables its traditional Device Number field being 0 enforcement when turning a Type 1 Configuration Request into a Type 0 Configuration Request, permitting access to Extended Functions in an ARI Device immediately below the Port. For more details, see Section 6.13 of PCI Express Base Specification.
PCIE_CAP_LTR_EN	10	rwh	**LTR Mechanism Enable** When set to 1b, this bit enables Upstream Ports to send LTR messages and Downstream Ports to process LTR Messages. For Downstream Ports, this bit must be reset to the default value if the Port goes to DL_Down status. The write value is gated with the PCIE_CAP_LTR_SUPP field of DEVICE_CAPABILITIES2_REG. Note: RW for function #0 and RsdvP for all other functions. Note: The access attributes of this field are as follows: - Wire: if (pf=0 && DEVICE_CAPABILITIES2_REG.PCIE_CAP_LTR_SUPP) then R/W else R - Dbi: if (pf=0 && DEVICE_CAPABILITIES2_REG.PCIE_CAP_LTR_SUPP) then R/W else R
PCIE_CAP_OBFF_EN	14:13	rw	**OBFF Enable** This field enables the OBFF mechanism and selects the signaling method. For more details on these encodings, see section 6.19 of PCI Express Base Specification. This field is required for all Ports that support the OBFF Capability. Note: RW for function #0 and RsdvP for all other functions. Note: The access attributes of this field are as follows: • Wire: if (DEVICE_CAPABILITIES2_REG.PCIE_CAP_OBFF_SUPPORT) then R/W else R • Dbi: if (DEVICE_CAPABILITIES2_REG.PCIE_CAP_OBFF_SUPPORT) then R/W else R 00_B **DISABLED**: Disabled 01_B **EN_USING_MSG_SIG_A**: Enabled using Message signaling [Variation A] 10_B **EN_USING_MSG_SIG_B**: Enabled using Message signaling [Variation B] 11_B **EN_USING_WAKE_SIG**: Enabled using WAKE# signaling

(table continues...)

(continued)

Field	Bits	Type	Description
0	9:6, 12:11, 31:15	r	**Reserved** Read as all 0's; should be written with all 0's.

Table 889 Reset values of DEVICE_CONTROL2_DEVICE_STATUS2_REG

Reset	Reset value	Note
Kernel Reset	0000 0000 0000 0000 0000 0X00 0000 0000_B	

19.4.2.131 Link Capabilities 2 Register

This register identifies PCI Express Link specific capabilities; in addition to the Link Capabilities Register.

LINK_CAPABILITIES2_REG Offset address: $000409C_H$

Link Capabilities 2 Register Reset values see: Table 890

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DRS_SUPPORTED	**RSVDP_25**						**0**		**RSVDP_9**						
rw	r						r		r						

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVDP_9							**PCIE_CAP_CROSS_LINK_SUPPORT**	**PCIE_CAP_SUPPORT_LINK_SPEED_VECTOR**							**RSVDP_0**
r							r	rh							r

Field	Bits	Type	Description
RSVDP_0	0	r	**Reserved for future use**

(table continues...)

(continued)

Field	Bits	Type	Description
PCIE_CAP_SUPPORT_LINK_SPEED_VECTOR	7:1	rh	**Supported Link Speeds Vector** This field indicates the supported Link speed(s) of the associated Port. For each bit, a value of 1b indicates that the corresponding Link speed is supported; otherwise, the Link speed is not supported. For more details, see section 8.2.1 of PCI Express Base Specification. Bit definitions within this field are: • Bit 0 2.5 GT/s • Bit 1 5.0 GT/s • Bit 2 8.0 GT/s • Bit 3 16.0 GT/s • Bit 4 32.0 GT/s • Bits 6:5 RsvdP This field has a default of (PCIE_CAP_MAX_LINK_SPEED == 0101) ? 0011111 : (PCIE_CAP_MAX_LINK_SPEED == 0100) ? 0001111 : (PCIE_CAP_MAX_LINK_SPEED == 0011) ? 0000111 : (PCIE_CAP_MAX_LINK_SPEED == 0010) ? 0000011 : 0000001 where PCIE_CAP_MAX_LINK_SPEED is a field in the LINK_CAPABILITIES_REG register.
PCIE_CAP_CROSS_LINK_SUPPORT	8	r	**Crosslink Supported** When set to 1b, this bit indicates that the associated Port supports crosslinks (for more details, see section 4.2.6.3.1 of PCI Express Base Specification). When set to 0b on a Port that supports Link speeds of 8.0 GT/s or higher, this bit indicates that the associated Port does not support crosslinks. When set to 0b on a Port that only supports Link speeds of 2.5 GT/s or 5.0 GT/s, this bit provides no information regarding the Port's level of crosslink support. It is recommended that this bit be Set in any Port that supports crosslinks even though doing so is only required for Ports that also support operating at 8.0 GT/s or higher Link speeds. Note: Software should use this bit when referencing fields whose definition depends on whether or not the Port supports crosslinks (for more details, see section 7.7.3.4 of PCI Express Base Specification). Multi-Function Devices associated with an Upstream Port must report the same value in this field for all Functions.No crosslink capability in this implementation
RSVDP_9	22:9	r	**Reserved for future use**
RSVDP_25	30:25	r	**Reserved for future use**
DRS_SUPPORTED	31	rw	**DRS Supported** When set, indicates support for the optional Device Readiness Status (DRS) capability. Must be Set in Downstream Ports that support DRS. Must be Set in Downstream Ports that support FRS. For Upstream Ports that support DRS, it is strongly recommended that this bit be Set in Function 0. For all other Functions associated with an Upstream Port, this bit must be Clear.127 Must be Clear in Functions that are not associated with a Port. RsvdP in all other Functions. Note: The access attributes of this field are as follows: - Wire: HWINIT - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R

(table continues...)

(continued)

Field	Bits	Type	Description
0	24:23	r	**Reserved** Read as all 0's; should be written with all 0's.

Table 890 Reset values of LINK_CAPABILITIES2_REG

Reset	Reset value	Note
Kernel Reset	1000 0000 0000 0000 0000 0000 XXXX XXX0$_B$	

19.4.2.132 Link Control 2 and Status 2 Register

This register controls and provides information about PCI Express Link specific parameters; in addition to the Link Control and Link Status Register.

LINK_CONTROL2_LINK_STATUS2_REG Offset address: 00040A0$_H$

Link Control 2 and Status 2 Register Kernel Reset value: 0001 0003$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DRS_MESSAGE_RECEIVED	DOWNSTREAM_COMPO_PRESENCE			RSVDP_26		0				PCIE_CAP_LINK_EQ_REQ	PCIE_CAP_EQ_CPL_P3	PCIE_CAP_EQ_CPL_P2	PCIE_CAP_EQ_CPL_P1	PCIE_CAP_EQ_CPL	PCIE_CAP_CURR_DEEMPHASIS
rw1ch	r			r		r				rw1ch	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PCIE_CAP_COMPLIANCE_PRESET				PCIE_CAP_COMPLIANCE_SOS	PCIE_CAP_ENTER_MODIFIED_COMPLIANCE	PCIE_CAP_TX_MARGIN			PCIE_CAP_SEL_DEEMPHASIS	PCIE_CAP_HW_AUTO_SPEED_DISABLE	PCIE_CAP_ENTER_COMPLIANCE	PCIE_CAP_TARGET_LINK_SPEED			
rwh				rwh	rw	rwh			rw	rw	rwh	rw			

Field	Bits	Type	Description
PCIE_CAP_TARGET_LINK_SPEED	3:0	rw	**Target Link Speed** For Downstream Ports, this field sets an upper limit on Link operational speed by restricting the values advertised by the Upstream component in its training sequences. The encoded value specifies a bit location in the Supported Link Speeds Vector (in the Link Capabilities 2 register) that corresponds to the desired target Link speed. All encodings other than the defined encodings are reserved. . If a value is written to this field that does not correspond to a supported speed (as indicated by the Supported Link Speeds Vector), the result is undefined. If either of the Enter Compliance or Enter Modified Compliance bits are implemented, then this field must also be implemented. The default value of this field is the highest Link speed supported by the component (as reported in the Max Link Speed field of the Link Capabilities register) unless the corresponding platform/form factor requires a different default value. For both Upstream and Downstream Ports, this field is used to set the target compliance mode speed when software is using the Enter Compliance bit to force a Link into compliance mode. For Upstream Ports, if the Enter Compliance bit is Clear, this field is permitted to have no effect. Note: The access attributes of this field are as follows: - Wire: R/W (sticky) - Dbi: R/W (sticky) Note: This register field is sticky 1_H **SUP_LINK_SPEED_VECTOR_BIT_0**: Supported Link Speeds Vector field bit 0 2_H **SUP_LINK_SPEED_VECTOR_BIT_1**: Supported Link Speeds Vector field bit 1 3_H **SUP_LINK_SPEED_VECTOR_BIT_2**: Supported Link Speeds Vector field bit 2 4_H **SUP_LINK_SPEED_VECTOR_BIT_3**: Supported Link Speeds Vector field bit 3 5_H **SUP_LINK_SPEED_VECTOR_BIT_4**: Supported Link Speeds Vector field bit 4 6_H **SUP_LINK_SPEED_VECTOR_BIT_5**: Supported Link Speeds Vector field bit 5 7_H **SUP_LINK_SPEED_VECTOR_BIT_6**: Supported Link Speeds Vector field bit 6 **others,** Reserved
PCIE_CAP_ENTER_COMPLIANCE	4	rwh	**Enter Compliance** Software is permitted to force a Link to enter Compliance mode (at the speed indicated in the Target Link Speed field and the de-emphasis/ preset level indicated by the Compliance Preset/De-emphasis field) by setting this bit to 1b in both components on a Link and then initiating a hot reset on the Link. Default value of this bit following Fundamental Reset is 0b. Components that support only the 2.5 GT/s speed are permitted to hardwire this bit to 0b. This bit is intended for debug, compliance testing purposes only. System firmware and software is allowed to modify this bit only during debug or compliance testing. In all other cases, the system must ensure that this bit is set to the default value. Note: The access attributes of this field are as follows: - Wire: R/W (sticky) - Dbi: R/W (sticky) Note: This register field is sticky.

(table continues...)

(continued)

Field	Bits	Type	Description
PCIE_CAP_HW_AUTO_SPEED_DISABLE	5	rw	**Hardware Autonomous Speed Disable** When set, this bit disables hardware from changing the Link speed for device-specific reasons other than attempting to correct unreliable Link operation by reducing Link speed. Initial transition to the highest supported common link speed is not blocked by this bit. Functions that do not implement the associated mechanism are permitted to hardwire this bit to 0b. Note: The access attributes of this field are as follows: - Wire: R/W (sticky) - Dbi: R/W (sticky) Note: This register field is sticky.
PCIE_CAP_SEL_DEEMPHASIS	6	rw	**Selectable De-emphasis** When the Link is operating at 5.0 GT/s speed, this bit is used to control the transmit de-emphasis of the link in specific situations. For more details, see section 4.2.6 of PCI Express Base Specification. When the Link is not operating at 5.0 GT/s speed, the setting of this bit has no effect. Components that support only the 2.5 GT/s speed are permitted to hardwire this bit to 0b. This bit is not applicable and Reserved for Endpoints, PCI Express to PCI/PCI-X bridges, and Upstream Ports of Switches. Note: This register field is sticky. 0_B **_6_DB**: -6 dB 1_B **_3_5_DB**: -3.5 dB
PCIE_CAP_TX_MARGIN	9:7	rwh	**Transmit Margin, This field controls the value of the non-deemphasized voltage level at the Transmitter pins** This field is reset to 000b on entry to the LTSSM Polling.Configuration substate (see Chapter 4 of PCI Express Base Specification for details of how the Transmitter voltage level is determined in various states). - 001b-111b: As defined in Section 8.3.4 not all encodings are required to be implemented. For components that support only the 2.5 GT/s speed, the controller hardwires this bit to 000b. This field is intended for debug, compliance testing purposes only. System firmware and software is allowed to modify this field only during debug or compliance testing. In all other cases, the system must ensure that this field is set to the default value. Note: This register field is sticky. 000_B **NORMAL_RANGE**: Normal operating range **others,** Reserved
PCIE_CAP_ENTER_MODIFIED_COMPLIANCE	10	rw	**Enter Modified Compliance** When this bit is set to 1b, the device transmits Modified Compliance Pattern if the LTSSM enters Polling.Compliance substate. Components that support only the 2.5 GT/s speed are permitted to hardwire this bit to 0b. This bit is intended for debug, compliance testing purposes only. System firmware and software is allowed to modify this bit only during debug or compliance testing. In all other cases, the system must ensure that this bit is set to the default value. Note: The access attributes of this field are as follows: - Wire: R/W (sticky) - Dbi: R/W (sticky) Note: This register field is sticky.

(table continues...)

(continued)

Field	Bits	Type	Description
PCIE_CAP_COMPLIANCE_SOS	11	rwh	**Compliance SOS** When set to 1b, the LTSSM is required to send SKP Ordered Sets between sequences when sending the Compliance Pattern or Modified Compliance Pattern. This bit is applicable when the Link is operating at 2.5 GT/s or 5.0 GT/s data rates only. Note: The access attributes of this field are as follows: - Wire: R/W (sticky) - Dbi: R/W (sticky) Note: This register field is sticky.
PCIE_CAP_COMPLIANCE_PRESET	15:12	rwh	**Compliance Preset/De-emphasis** • For 8.0 GT/s and higher Data Rate: This field sets the Transmitter Preset in Polling.Compliance state if the entry occurred due to the Enter Compliance bit being 1b. The encodings are defined in section 4.2.3.2 of PCI Express Base Specification . Results are undefined if a reserved preset encoding is used when entering Polling.Compliance in this way. • For 5.0 GT/s Data Rate: This field sets the de-emphasis level in Polling.Compliance state if the entry occurred due to the Enter Compliance bit being 1b • When the Link is operating at 2.5 GT/s, the setting of this field has no effect. Components that support only 2.5 GT/s speed are permitted to hardwire this field to 0000b This field is intended for debug and compliance testing purposes. System firmware and software is allowed to modify this field only during debug or compliance testing. In all other cases, the system must ensure that this field is set to the default value. Note: The access attributes of this field are as follows: - Wire: R/W (sticky) - Dbi: R/W (sticky) Note: This register field is sticky. 0_H **_6_DB**: -6 dB (for 5.0 GT/s Data Rate) 1_H **_3_5_DB**: -3.5 dB (for 5.0 GT/s Data Rate) **others,** Reserved
PCIE_CAP_CURR_DEEMPHASIS	16	rh	**Current De-emphasis Level** When the Link is operating at 5.0 GT/s speed, this bit reflects the level of de-emphasis. The value in this bit is undefined when the Link is not operating at 5.0 GT/s speed. 0_B **_6DB**: -6 dB 1_B **_3_5DB**: -3.5 dB
PCIE_CAP_EQ_CPL	17	rh	**Equalization 8.0 GT/s Complete** When set to 1b, this bit indicates that the Transmitter Equalization procedure at the 8.0 GT/s data rate has completed. Details of the Transmitter Equalization process and when this bit needs to be set to 1b is provided in section 4.2.6.4.2 of PCI Express Base Specification. For Multi-Function Upstream Port, this bit must be implemented in Function 0 and RsvdZ in other Functions. For components that only support speeds below 8.0 GT/s, the controller hardwires this bit to 0b. Note: This register field is sticky.

(table continues...)

(continued)

Field	Bits	Type	Description
PCIE_CAP_EQ_CPL_P1	18	rh	**Equalization 8.0 GT/s Phase 1 Successful** When set to 1b, this bit indicates that Phase 1 of the 8.0 GT/s Transmitter Equalization procedure has successfully completed. Details of the Transmitter Equalization process and when this bit needs to be set to 1b is provided in section 4.2.6.4.2 of PCI Express Base Specification. Note: This register field is sticky.
PCIE_CAP_EQ_CPL_P2	19	rh	**Equalization 8.0 GT/s Phase 2 Successful** When set to 1b, this bit indicates that Phase 2 of the 8.0 GT/s Transmitter Equalization procedure has successfully completed. Details of the Transmitter Equalization process and when this bit needs to be set to 1b is provided in section 4.2.6.4.2 of PCI Express Base Specification. Note: This register field is sticky.
PCIE_CAP_EQ_CPL_P3	20	rh	**EEqualization 8.0 GT/s Phase 3 Successful** When set to 1b, this bit indicates that Phase 3 of the 8.0 GT/s Transmitter Equalization procedure has successfully completed. Details of the Transmitter Equalization process and when this bit needs to be set to 1b is provided in section 4.2.6.4.2 of PCI Express Base Specification. Note: This register field is sticky.
PCIE_CAP_LINK_EQ_REQ	21	rw1ch	**Link Equalization Request 8.0 GT/s** This bit is set by hardware to request the 8.0 GT/s Link equalization process to be performed on the Link. For more details, see sections 4.2.3 and 4.2.6.4.2 of PCI Express Base Specification. Note: This register field is sticky. Software write with 1 clears this bit-field.
RSVDP_26	27:26	r	**Reserved for future use**

(table continues...)

(continued)

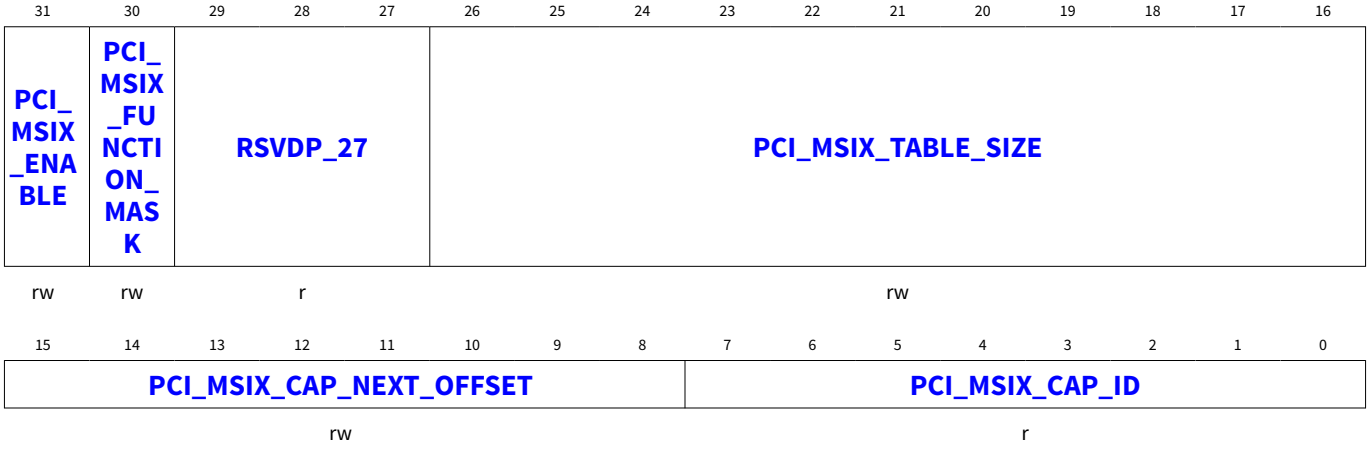
Field	Bits	Type	Description
DOWNSTREAM_COMPO_PRESENCE	30:28	r	**Downstream Component Presence** This field indicates the presence and DRS status for the Downstream Component, if any, connected to the Link. All encodings other than the defined encodings are reserved. Component Presence state must be determined by the logical "OR" of the Physical Layer in-band presence detect mechanism and, if present, any out-of-band presence detect mechanism implemented for the Link. If no out-of-band presence detect mechanism is implemented, then Component Presence state must be determined solely by the Physical Layer in-band presence detect mechanism. This field must be implemented in any Downstream Port where the DRS Supported bit is Set in the Link Capabilities 2 register. This field is RsvdZ for all other Functions. 000_B **LINK_DN_NOT_DETERMINED**: Link Down - Presence Not Determined 001_B **LINK_DN_DSP_NOT_PRESENT**: Link Down - Component Not Present indicates the Downstream Port (DP) has determined that a Downstream Component is not present 010_B **LINK_DN_DLL_NOT_ACTIVE**: Link Down - Component Present indicates the DP has determined that a Downstream Component is present, but the Data Link Layer is not active 100_B **LINK_UP_DRS_NOT_RECEIVED**: Link Up - Component Presentindicates the DP has determined that a Downstream Component is present, but no DRS Message has been received since the Data Link Layer became active 101_B **LINK_UP_DRS_RECEIVED**: Link Up - Component Present and DRS Received indicates the DP has received a DRS Message since the Data Link Layer became active **others,** Reserved
DRS_MESSAGE_RECEIVED	31	rw1ch	**DRS Message Received** This bit must be set whenever the Port receives a DRS Message. This bit must be cleared in DL_Down. This bit must be implemented in any Downstream Port where the DRS Supported bit is Set in the Link Capabilities 2 register. This bit is RsvdZ for all other Functions. Software write with 1 clears this bit-field.
0	25:22	r	**Reserved** Read as all 0's; should be written with all 0's.

19.4.2.133 MSI-X Capability ID, Next Pointer, Control Register

This Register holds MSI-X Capability ID, Next Capability pointer. It also controls the MSI-X behaviour.

PCI_MSIX_CAP_ID_NEXT_CTRL_REG Offset address: $00040B0_H$

MSI-X Capability ID, Next Pointer, Control Register Kernel Reset value: $0040\ 0011_H$



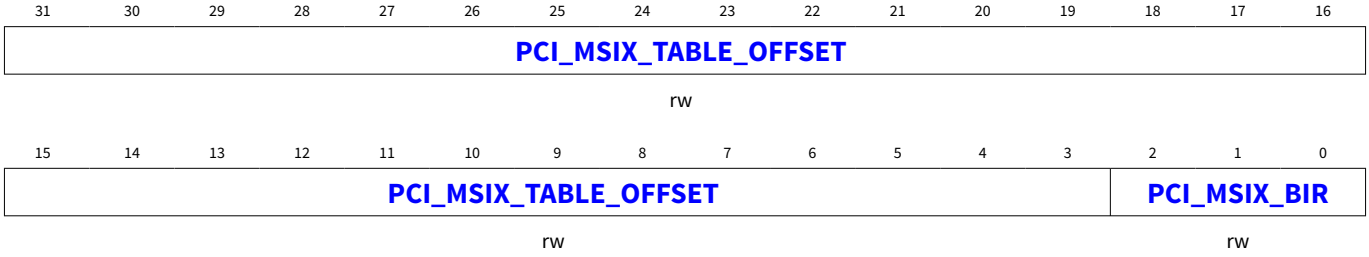
Field	Bits	Type	Description
PCI_MSIX_CAP_ID	7:0	r	**MSI-X Capability ID** This field indicates the MSI-X Capability structure. This field must return a Capability ID of 11h indicating that this is an MSI-X Capability structure.
PCI_MSIX_CAP_NEXT_OFFSET	15:8	rw	**MSI-X Next Capability Pointer** This field contains the offset to the next PCI Capability structure or 00h if no other items exist in the linked list of Capabilities. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R Note: This register field is sticky.
PCI_MSIX_TABLE_SIZE	26:16	rw	**MSI-X Table Size** System software reads this field to determine the MSI-X Table Size N, which is encoded as N-1. For example, a returned value of 000 0000 0011b indicates a table size of 4. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R Note: This register field is sticky.
RSVDP_27	29:27	r	**Reserved for future use**
PCI_MSIX_FUNCTION_MASK	30	rw	**Function Mask** If Set, all of the vectors associated with the Function are masked, regardless of their per-vector Mask bit values. If Clear, each vector's Mask bit determines whether the vector is masked or not. Setting or Clearing the MSI-X Function Mask bit has no effect on the value of the per-vector Mask bits. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
PCI_MSIX_ENABLE	31	rw	**MSI-X Enable** If Set and the MSI Enable bit in the MSI Message Control Register for MSI is Clear, the Function is permitted to use MSI-X to request service and is prohibited from using INTx interrupts (if implemented). System configuration software Sets this bit to enable MSI-X. A device driver is prohibited from writing this bit to mask a Function's service request. If Clear, the Function is prohibited from using MSI-X to request service.

19.4.2.134 MSI-X Table Offset and BIR Register

This register provides Table BIR and MSI-x Table offset select.

MSIX_TABLE_OFFSET_REG Offset address: $00040B4_H$

MSI-X Table Offset and BIR Register Kernel Reset value: $0000\ 0A00_H$



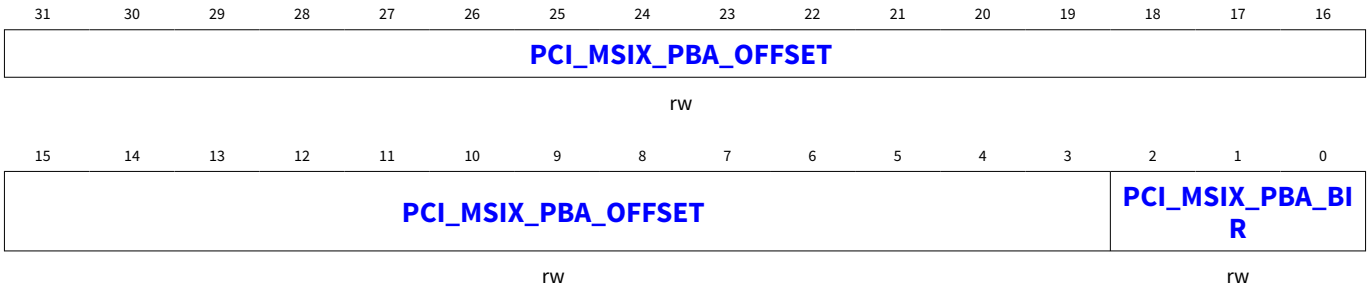
Field	Bits	Type	Description
PCI_MSIX_BIR	2:0	rw	**MSI-X Table BAR Indicator Register Field** This field indicates which one of a Function's Base Address Registers , located beginning at 10h in Configuration Space, or entry in the Enhanced Allocation capability with a matching BEI , is used to map the Function's MSI-X Table into Memory Space. All encodings other than the defined encodings are reserved. For a 64-bit Base Address Register , the Table BIR indicates the lower DWORD. For Functions with Type 1 Configuration Space headers, BIR values 2 through 5 are also Reserved. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R Note: This register field is sticky. 000_B **BAR_10**: Base Address Register 10h 001_B **BAR_14**: Base Address Register 14h 010_B **BAR_18**: Base Address Register 18h 011_B **BAR_1C**: Base Address Register 1Ch 100_B **BAR_20**: Base Address Register 20h 101_B **BAR_24**: Base Address Register 24h **others,** Reserved
PCI_MSIX_TABLE_OFFSET	31:3	rw	**MSI-X Table Offset** Used as an offset from the address contained by one of the Function's Base Address Registers to point to the base of the MSI-X Table. The lower 3 Table BIR bits are masked off (set to zero) by software to form a 32-bit QWORD-aligned offset. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R Note: This register field is sticky.

19.4.2.135 MSI-X PBA Offset and BIR Register

This register provides PBA Offset and PBA BIR value.

MSIX_PBA_OFFSET_REG Offset address: $00040B8_H$

MSI-X PBA Offset and BIR Register Kernel Reset value: $0000\ 0E10_H$



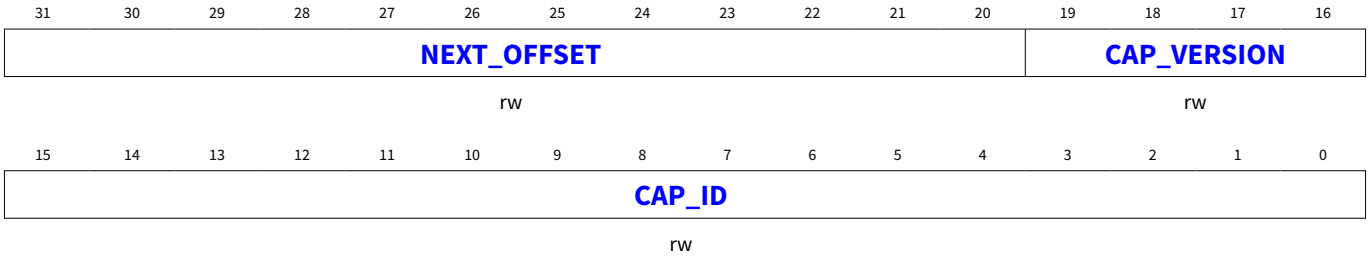
Field	Bits	Type	Description
PCI_MSIX_PBA_BIR	2:0	rw	**MSI-X PBA BIR** This field indicates which one of a Function's Base Address Registers , located beginning at 10h in Configuration Space, or entry in the Enhanced Allocation capability with a matching BEI, is used to map the Function's MSI-X PBA into Memory Space. The PBA BIR value definitions are identical to those for the Table BIR . Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R Note: This register field is sticky.
PCI_MSIX_PBA_OFFSET	31:3	rw	**MSI-X PBA Offset** Used as an offset from the address contained by one of the Function's Base Address Registers to point to the base of the MSI-X PBA. The lower 3 PBA BIR bits are masked off (set to zero) by software to form a 32-bit QWORD-aligned offset. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R Note: This register field is sticky.

19.4.2.136 Advanced Error Reporting Extended Capability Header

Advanced Error Reporting Extended Capability Header provides information about Capability ID, Version, and next offset.

AER_EXT_CAP_HDR_OFF Offset address: 0004100_H

Advanced Error Reporting Extended Capability Header Kernel Reset value: $1482\ 0001_H$



Field	Bits	Type	Description
CAP_ID	15:0	rw	**AER Extended Capability ID** This field is a PCI-SIG defined ID number that indicates the nature and format of the Extended Capability. The Extended Capability ID for the Advanced Error Reporting Capability is 0001h. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky.
CAP_VERSION	19:16	rw	**Capability Version** This field is a PCI-SIG defined version number that indicates the version of the Capability structure present. This field must be 2h if the End-End TLP Prefix Supported bit is set and must be 1h or 2h otherwise. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky.
NEXT_OFFSET	31:20	rw	**Next Capability Offset** This field contains the offset to the next PCI Express Capability structure or 000h if no other items exist in the linked list of Capabilities. For Extended Capabilities implemented in Configuration Space, this offset is relative to the beginning of PCI-compatible Configuration Space and thus must always be either 000h (for terminating list of Capabilities) or greater than 0FFh. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky.

19.4.2.137 Uncorrectable Error Status Register

The Uncorrectable Error Status Register (Offset 04h) indicates error detection status of individual errors on a PCI Express device Function. An individual error status bit that is Set indicates that a particular error was detected; software may clear an error status by writing a 1b to the respective bit. Register bits not implemented by the Function are hardwired to 0b.

UNCORR_ERR_STATUS_OFF Offset address: 0004104_H

Uncorrectable Error Status Register Reset values see: Table 891

19 Peripheral Component Interconnect Express (PCIe)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_27					0			RSVDP_23	INTERNAL_ERR_STATUS	0	UNSUPPORTED_REQ_ERR_STATUS	ECRC_ERR_STATUS	MALF_TLP_ERR_STATUS	RECOVERFLOW_ERR_STATUS	UNEXP_CMPLT_ERR_STATUS
r					r			r	rw1ch	r	rw1ch	rw1ch	rw1ch	rw1ch	rw1ch

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CMPLT_ABORT_ERR_STATUS	CMPLT_TIMEOUT_ERR_STATUS	FC_PROTOCOL_ERR_STATUS	POIS_TLP_ERR_STATUS	RSVDP_6						SURPRISE_DOWN_ERR_STATUS	DL_PROTOCOL_ERR_STATUS	RSVDP_0			
rw1ch	rw1ch	rw1ch	rw1ch	r						rw1ch	rw1ch	r			

Field	Bits	Type	Description
RSVDP_0	3:0	r	**Reserved for future use**
DL_PROTOCOL_ERR_STATUS	4	rw1ch	**Data Link Protocol Error Status** Status bit for Data Link Protocol Error. Note: This register field is sticky. Software write with 1 clears this bit-field.
SURPRISE_DOWN_ERR_STATUS	5	rw1ch	**Surprise Down Error Status (Optional)** Status bit for Surprise Down Error. Note: This register field is sticky. Software write with 1 clears this bit-field.
RSVDP_6	11:6	r	**Reserved for future use**
POIS_TLP_ERR_STATUS	12	rw1ch	**Poisoned TLP Status** Status bit for Poisoned TLP. Note: This register field is sticky. Software write with 1 clears this bit-field.
FC_PROTOCOL_ERR_STATUS	13	rw1ch	**Flow Control Protocol Error Status** Status bit for Flow Control Protocol Error. Note: This register field is sticky. Software write with 1 clears this bit-field.
CMPLT_TIMEOUT_ERR_STATUS	14	rw1ch	**Completion Timeout Status** Status for Completion Timeout. Note: This register field is sticky. Software write with 1 clears this bit-field.
CMPLT_ABORT_ERR_STATUS	15	rw1ch	**Completer Abort Status** Status bit for Completer Abort. Note: This register field is sticky. Software write with 1 clears this bit-field.
UNEXP_CMPLT_ERR_STATUS	16	rw1ch	**Unexpected Completion Status** Status bit for Unexpected Completion. Note: This register field is sticky. Software write with 1 clears this bit-field.

(table continues...)

(continued)

Field	Bits	Type	Description
REC_OVERFLOW_ERR_STATUS	17	rw1ch	**Receiver Overflow Status** Status bit for Receiver Overflow. Note: This register field is sticky. Software write with 1 clears this bit-field.
MALF_TLP_ERR_STATUS	18	rw1ch	**Malformed TLP Status** This field represents status of Malformed TLP. Note: This register field is sticky. Software write with 1 clears this bit-field.
ECRC_ERR_STATUS	19	rw1ch	**ECRC Error Status** This field represents status of ECRC Error. Note: This register field is sticky. Software write with 1 clears this bit-field.
UNSUPPORTED_REQ_ERR_STATUS	20	rw1ch	**Unsupported Request Error Status** This field represents status of Unsupported Request Error. Note: This register field is sticky. Software write with 1 clears this bit-field.
INTERNAL_ERR_STATUS	22	rw1ch	**Uncorrectable Internal Error Status** This field gives status of the Uncorrectable Internal Error.Note: This register field is sticky. Software write with 1 clears this bit-field.
RSVDP_23	23	r	**Reserved for future use**
RSVDP_27	31:27	r	**Reserved for future use**
0	21, 26:24	r	**Reserved** Read as 0; should be written with 0.

Table 891 Reset values of UNCORR_ERR_STATUS_OFF

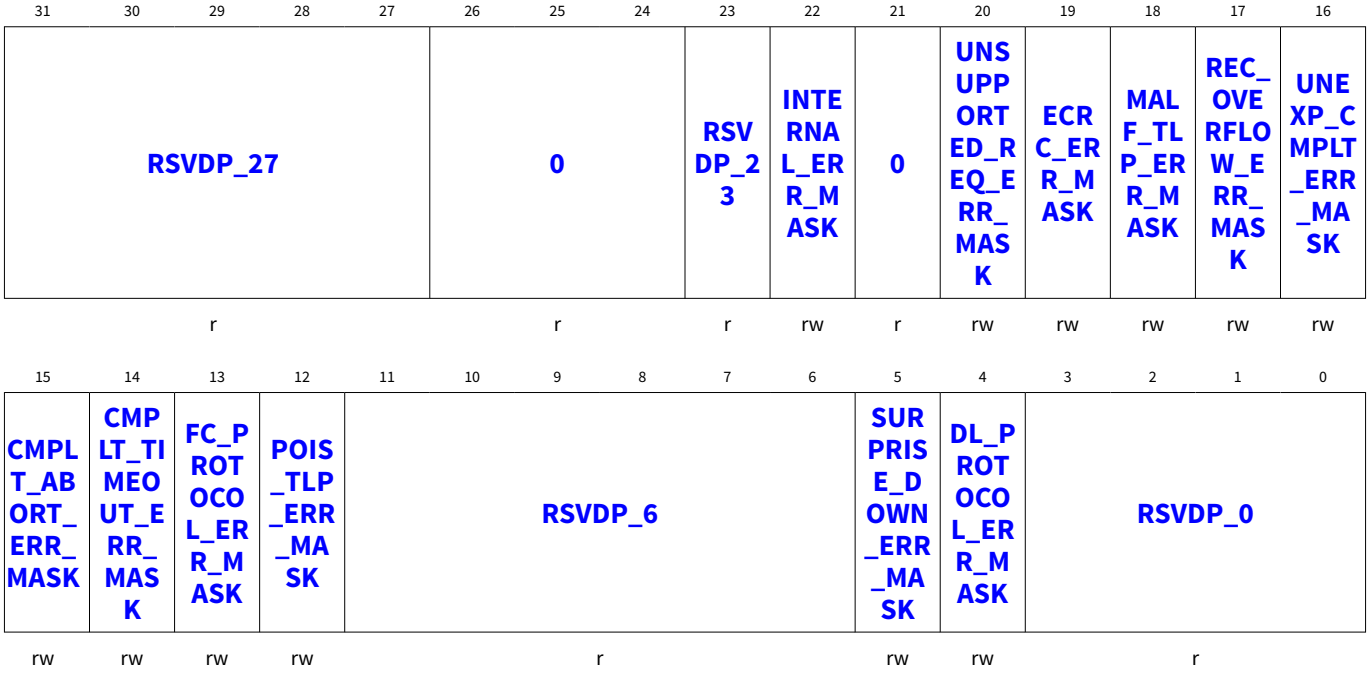
Reset	Reset value	Note
Kernel Reset	0000 0000 0000 0XX0 00X0 0000 00XX 0000_B	

19.4.2.138 Uncorrectable Error Mask Register

The Uncorrectable Error Mask Register controls reporting of individual errors by the device Function to the PCI Express Root Complex through a PCI Express error Message. A masked error (respective bit Set in the mask register) is not recorded or reported in the Header Log, TLP Prefix Log, or First Error Pointer, and is not reported to the PCI Express Root Complex by this Function. There is a mask bit per error bit of the Uncorrectable Error Status register. Register fields for bits not implemented by the Function are hardwired to 0b.

UNCORR_ERR_MASK_OFF Offset address: 0004108_H

Uncorrectable Error Mask Register Kernel Reset value: $0040\ 0000_H$



Field	Bits	Type	Description
RSVDP_0	3:0	r	**Reserved for future use**
DL_PROTOCOL_ERR_MASK	4	rw	**Data Link Protocol Error Mask** This field informs whether Data Link Protocol Error is masked or not. Note: This register field is sticky.
SURPRISE_DOWN_ERR_MASK	5	rw	**Surprise Down Error Mask** Mask bit for Surprise Down Error. Note: This register field is sticky.
RSVDP_6	11:6	r	**Reserved for future use**
POIS_TLP_ERR_MASK	12	rw	**Poisoned TLP Error Mask** Mask bit for Poisoned TLP Error. Note: This register field is sticky.
FC_PROTOCOL_ERR_MASK	13	rw	**Flow Control Protocol Error Mask** Mask bit for Flow Control Protocol Error. Note: This register field is sticky.
CMPLT_TIMEOUT_ERR_MASK	14	rw	**Completion Timeout Error Mask** Mask bit for Completion Timeout Error. Note: This register field is sticky.
CMPLT_ABORT_ERR_MASK	15	rw	**Completer Abort Error Mask (Optional)** Mask bit for Completer Abort Error. Note: This register field is sticky.
UNEXP_CMPLT_ERR_MASK	16	rw	**Unexpected Completion Mask** Mask bit for Unexpected Completion Error. Note: This register field is sticky.
REC_OVERFLOW_ERR_MASK	17	rw	**Receiver Overflow Mask (Optional)** This field represents Receiver Overflow Mask. Note: This register field is sticky.
MALF_TLP_ERR_MASK	18	rw	**Malformed TLP Mask** Mask bit for Malformed TLP. Note: This register field is sticky.

(table continues...)

(continued)

Field	Bits	Type	Description
ECRC_ERR_MASK	19	rw	**ECRC Error Mask (Optional)** Mask bit for ECRC Error. Note: The access attributes of this field are as follows: - Wire: R/W (sticky) - Dbi: R/W (sticky) Note: This register field is sticky.
UNSUPPORTED_REQ_ERR_MASK	20	rw	**Unsupported Request Error Mask** Mask bit for Unsupported Request Error. Note: This register field is sticky.
INTERNAL_ERR_MASK	22	rw	**Uncorrectable Internal Error Mask (Optional)** Mask bit for Uncorrectable Internal Error. Note: This register field is sticky.
RSVDP_23	23	r	**Reserved for future use**
RSVDP_27	31:27	r	**Reserved for future use**
0	21, 26:24	r	**Reserved** Read as 0; should be written with 0.

19.4.2.139 Uncorrectable Error Severity Register

The Uncorrectable Error Severity Register controls whether an individual error is reported as a Non-fatal or Fatal error. An error is reported as fatal when the corresponding error bit in the severity register is Set. If the bit is Clear, the corresponding error is considered non-fatal.

UNCORR_ERR_SEV_OFF Offset address: $000410C_H$

Uncorrectable Error Severity Register Kernel Reset value: $0046\ 2030_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_27					0			RSVDP_23	INTERNAL_ERR_SEVERITY	0	UNSUPPORTED_REQ_ERR_SEVERITY	ECRC_ERR_SEVERITY	MALF_TLP_ERR_SEVERITY	REC_OVERFLOW_ERR_SEVERITY	UNEXP_CMPLT_ERR_SEVERITY
r					r			r	rw	r	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CMPLT_ABORT_ERR_SEVERITY	CMPLT_TIMEOUT_ERR_SEVERITY	FC_PROTOCOL_ERR_SEVERITY	POIS_TLP_ERR_SEVERITY	RSVDP_6						SURPRISE_DOWN_ERR_SVRITY	DL_PROTOCOL_ERR_SEVERITY	RSVDP_0			
rw	rw	rw	rw	r						rw	rw	r			

Field	Bits	Type	Description
RSVDP_0	3:0	r	**Reserved for future use**

(table continues...)

(continued)

Field	Bits	Type	Description
DL_PROTOCOL_ERR_SEVERITY	4	rw	**Data Link Protocol Error Severity** Severity bit for Data Link Protocol Error. Note: This register field is sticky.
SURPRISE_DOWN_ERR_SVRITY	5	rw	**Surprise Down Error Severity (Optional)** Severity bit for Surprise Down Error. Note: This register field is sticky.
RSVDP_6	11:6	r	**Reserved for future use**
POIS_TLP_ERR_SEVERITY	12	rw	**Poisoned TLP Severity** Severity bit for Poisoned TLP. Note: This register field is sticky.
FC_PROTOCOL_ERR_SEVERITY	13	rw	**Flow Control Protocol Error Severity (Optional)** Severity bit for Flow Control Protocol Error. Note: This register field is sticky.
CMPLT_TIMEOUT_ERR_SEVERITY	14	rw	**Completion Timeout Error Severity** Severity bit for Completion Timeout Error. Note: This register field is sticky.
CMPLT_ABORT_ERR_SEVERITY	15	rw	**Completer Abort Error Severity (Optional)** Severity bit for Completer Abort Error. Note: This register field is sticky.
UNEXP_CMPLT_ERR_SEVERITY	16	rw	**Unexpected Completion Error Severity** Severity bit for Unexpected Completion Error. Note: This register field is sticky.
REC_OVERFLOW_ERR_SEVERITY	17	rw	**Receiver Overflow Error Severity (Optional)** Severity bit for Receiver Overflow Error. Note: This register field is sticky.
MALF_TLP_ERR_SEVERITY	18	rw	**Malformed TLP Severity** Severity bit for Malformed TLP. Note: This register field is sticky.
ECRC_ERR_SEVERITY	19	rw	**ECRC Error Severity (Optional)** Severity bit for ECRC Error. Note: The access attributes of this field are as follows: - Wire: R/W (sticky) - Dbi: R/W (sticky) Note: This register field is sticky.
UNSUPPORTED_REQ_ERR_SEVERITY	20	rw	**Unsupported Request Error Severity** Severity bit for Unsupported Request Error. Note: This register field is sticky.
INTERNAL_ERR_SEVERITY	22	rw	**Uncorrectable Internal Error Severity (Optional)** Severity bit for Uncorrectable Internal Error. Note: This register field is sticky.
RSVDP_23	23	r	**Reserved for future use**
RSVDP_27	31:27	r	**Reserved for future use**
0	21, 26:24	r	**Reserved** Read as 0; should be written with 0.

19.4.2.140 Correctable Error Status Register

The Correctable Error Status register reports error status of individual correctable error sources on a PCI Express device Function. When an individual error status bit is Set, it indicates that a particular error occurred; software may clear an error status by writing a 1b to the respective bit. Register bits not implemented by the Function are hardwired to 0b by the controller.

CORR_ERR_STATUS_OFF Offset address: 0004110_H

Correctable Error Status Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_16															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HEADER_LOG_OVERFLOW_STATUS	CORRECTED_INT_ERR_STATUS	ADVISORY_NON_FATAL_ERR_STATUS	RPL_TIMER_TIMEOUT_STATUS	RSVDP_9			REPLAY_NO_ROLEOVER_STATUS	BAD_DLLP_STATUS	BAD_TLP_STATUS	RSVDP_1					RX_ERR_STATUS
rw1ch	rw1ch	rw1ch	rw1ch	r			rw1ch	rw1ch	rw1ch	r					rw1ch

Field	Bits	Type	Description
RX_ERR_STATUS	0	rw1ch	**Receiver Error Status (Optional)** This field provides status of Receiver Error. Note: This register field is sticky. Software write with 1 clears this bit-field.
RSVDP_1	5:1	r	**Reserved for future use**
BAD_TLP_STATUS	6	rw1ch	**Bad TLP Status** Status bit for Bad TLP. Note: This register field is sticky. Software write with 1 clears this bit-field.
BAD_DLLP_STATUS	7	rw1ch	**Bad DLLP Status** Status bit for Bad DLLP. Note: This register field is sticky. Software write with 1 clears this bit-field.
REPLAY_NO_ROLEOVER_STATUS	8	rw1ch	**REPLAY_NUM Rollover Status** Status bit for REPLAY_NUM Rollover. Note: This register field is sticky. Software write with 1 clears this bit-field.
RSVDP_9	11:9	r	**Reserved for future use**
RPL_TIMER_TIMEOUT_STATUS	12	rw1ch	**Replay Timer Timeout Status** Status bit for Replay Timer Timeout. Note: This register field is sticky. Software write with 1 clears this bit-field.
ADVISORY_NON_FATAL_ERR_STATUS	13	rw1ch	**Advisory Non-Fatal Error Status** Status bit for Advisory Non-Fatal Error. Note: This register field is sticky. Software write with 1 clears this bit-field.

(table continues...)

(continued)

Field	Bits	Type	Description
CORRECTED_INT_ERR_STATUS	14	rw1ch	**Corrected Internal Error Status (Optional)** This field provides status of Corrected Internal Error. Note: This register field is sticky. Software write with 1 clears this bit-field.
HEADER_LOG_OVERFLOW_STATUS	15	rw1ch	**Header Log Overflow Error Status (Optional)** This field provides status of Header Log Overflow Error. Note: This register field is sticky. Software write with 1 clears this bit-field.
RSVDP_16	31:16	r	**Reserved for future use**

19.4.2.141 Correctable Error Mask Register

The Correctable Error Mask Register controls reporting of individual correctable errors by this Function to the PCI Express Root Complex through a PCI Express error Message. A masked error (respective bit Set in the mask register) is not reported to the PCI Express Root Complex by this Function. There is a mask bit per error bit in the Correctable Error Status register. Register fields for bits not implemented by the Function are hardwired to 0b by the controller.

CORR_ERR_MASK_OFF Offset address: 0004114_H

Correctable Error Mask Register Kernel Reset value: $0000\ E000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_16															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HEADER_LOG_OVERFLOW_MASK	CORRECTED_INT_ERR_MASK	ADVISORY_NON_FATAL_ERR_MASK	RPL_TIMER_TIMEOUT_MASK	RSVDP_9			REPLAY_NO_ROLEOVER_MASK	BAD_DLLP_MASK	BAD_TLP_MASK	RSVDP_1					RX_ERR_MASK
rw	rw	rw	rw	r			rw	rw	rw	r					rw

Field	Bits	Type	Description
RX_ERR_MASK	0	rw	**Receiver Error Mask (Optional)** Masking bit for Receiver Error. Note: This register field is sticky.
RSVDP_1	5:1	r	**Reserved for future use**
BAD_TLP_MASK	6	rw	**Bad TLP Mask** Masking bit for Bad TLP. Note: This register field is sticky.
BAD_DLLP_MASK	7	rw	**Bad DLLP Mask** Masking bit for Bad DLLP. Note: This register field is sticky.
REPLAY_NO_ROLEOVER_MASK	8	rw	**REPLAY_NUM Rollover Mask** Masking bit for REPLAY_NUM Rollover. Note: This register field is sticky.
RSVDP_9	11:9	r	**Reserved for future use**

(table continues...)

(continued)

Field	Bits	Type	Description
RPL_TIMER_TIMEOUT_MASK	12	rw	**Replay Timer Timeout Mask** Masking bit for Replay Timer Timeout. Note: This register field is sticky.
ADVISORY_NON_FATAL_ERR_MASK	13	rw	**Advisory Non-Fatal Error Mask** Masking bit for Advisory Non-Fatal Error. Note: This register field is sticky.
CORRECTED_INT_ERR_MASK	14	rw	**Corrected Internal Error Mask (Optional)** Masking bit for Corrected Internal Error Mask. Note: This register field is sticky.
HEADER_LOG_OVERFLOW_MASK	15	rw	**Header Log Overflow Error Mask (Optional)** Masking bit for Header Log Overflow Error. Note: This register field is sticky.
RSVDP_16	31:16	r	**Reserved for future use**

19.4.2.142 Advanced Error Capabilities and Control Register

Advanced Error Capabilities and Control Register provides information whether the individual capability is supported or not. If the capability is supported then it is enabled or not.

ADV_ERR_CAP_CTRL_OFF Offset address: 0004118_H

Advanced Error Capabilities and Control Register Kernel Reset value: $0000\ 02A0_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_13															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVDP_13			CTO_PRFX_HDR_LOG_CAP	0	MULTIPLE_HEADER_EN	MULTIPLE_HEADER_CAP	ECRC_CHECK_EN	ECRC_CHECK_CAP	ECRC_GEN_EN	ECRC_GEN_CAP	FIRST_ERR_POINTER				
r			r	r	rw	r	rw	r	rw	r	rh				

Field	Bits	Type	Description
FIRST_ERR_POINTER	4:0	rh	**First Error Pointer** The First Error Pointer is a field that identifies the bit position of the first error reported in the Uncorrectable Error Status register. Note: This register field is sticky.
ECRC_GEN_CAP	5	r	**ECRC Generation Capable** If Set, this bit indicates that the Function is capable of generating ECRC. Note: This register field is sticky.
ECRC_GEN_EN	6	rw	**ECRC Generation Enable** When Set, ECRC generation is enabled. Functions that do not implement the associated mechanism are permitted to hardwire this bit to 0b. Default value of this bit is 0b. Note: This register field is sticky.

(table continues...)

(continued)

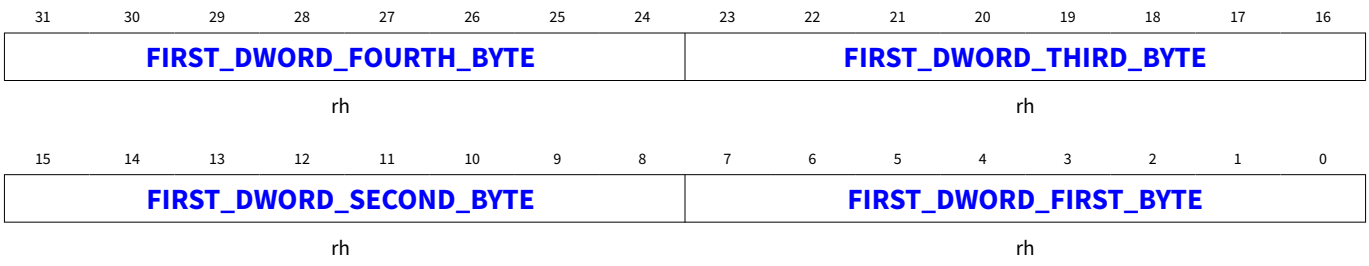
Field	Bits	Type	Description
ECRC_CHECK_CAP	7	r	**ECRC Check Capable** If Set, this bit indicates that the Function is capable of checking ECRC. Note: This register field is sticky.
ECRC_CHECK_EN	8	rw	**ECRC Check Enable** When Set, ECRC checking is enabled. Functions that do not implement the associated mechanism are permitted to hardwire this bit to 0b. Default value of this bit is 0b. Note: This register field is sticky.
MULTIPLE_HEADER_CAP	9	r	**Multiple Header Recording Capable** If Set, this bit indicates that the Function is capable of recording more than one error header. Note: This register field is sticky.
MULTIPLE_HEADER_EN	10	rw	**Multiple Header Recording Enable** When Set, this bit enables the Function to record more than one error header. Functions that do not implement the associated mechanism are permitted to hardwire this bit to 0b. Default value of this bit is 0b. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: R (sticky) Note: This register field is sticky.
CTO_PRFX_HDR_LOG_CAP	12	r	**Completion Timeout Prefix/Header Log Capable** If Set, this bit indicates that the Function records the prefix/header of Request TLPs that experience a Completion Timeout error.
RSVDP_13	31:13	r	**Reserved for future use**
0	11	r	**Reserved** Read as 0; should be written with 0.

19.4.2.143 Header i Log Register 0

The Header Log Register 0 contains the header for the TLP corresponding to a detected error; The header is captured such that, when read using DW accesses, the fields of the header are laid out in the same way the headers are presented in the specification. Therefore, byte 0 of the header is located in byte 3 of the Header Log Register 0, byte 1 of the header is in byte 2 of the Header Log Register 0, and so forth. For 12-byte headers, only bytes 0 through 11 of the Header Log Register are used and values in bytes 12 through 15 are undefined. In certain cases where a Malformed TLP is reported, the Header Log Register may contain TLP Prefix information.

HDR_LOG_i_OFF (i=0-3) Offset address: $000411C_H+i*4$

Header i Log Register 0 Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
FIRST_DWORD_FIRST_BYTE	7:0	rh	**Byte 0 of Header log register of First 32-bit Data Word** This field represents first byte of First DW of Header. Note: This register field is sticky.
FIRST_DWORD_SECOND_BYTE	15:8	rh	**Byte 1 of Header log register of First 32-bit Data Word** This field represents second byte of First DW of Header. Note: This register field is sticky.
FIRST_DWORD_THIRD_BYTE	23:16	rh	**Byte 2 of Header log register of First 32-bit Data Word** This field represents third byte of First DW of Header. Note: This register field is sticky.
FIRST_DWORD_FOURTH_BYTE	31:24	rh	**Byte 3 of Header log register of First 32-bit Data Word** This field represents fourth byte of First DW of Header. Note: This register field is sticky.

19.4.2.144 Root Error Command Register

The Root Error Command Register allows further control of Root Complex response to Correctable, Non-Fatal, and Fatal error Messages than the basic Root Complex capability to generate system errors in response to error Messages.

AERCAP_ROOT_ROOT_ERR_CMD_OFF Offset address: $000412C_H$

Root Error Command Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_3															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVDP_3													FATAL_ERR_REPORTING_EN	NON_FATAL_ERR_REPORTING_EN	CORR_ERR_REPORTING_EN
r													rw	rw	rw

Field	Bits	Type	Description
CORR_ERR_REPORTING_EN	0	rw	**Correctable Error Reporting Enable** When Set, this bit enables the generation of an interrupt when a correctable error is reported by any of the Functions in the Hierarchy Domain associated with this Root Port. Root Complex Event Collectors provide support for the above described functionality for RCiEPs (Root Complex integrated Endpoint).

(table continues...)

(continued)

Field	Bits	Type	Description
NON_FATAL_ERR_REPORTING_EN	1	rw	**Non-Fatal Error Reporting Enable** When Set, this bit enables the generation of an interrupt when a Non-fatal error is reported by any of the Functions in the Hierarchy Domain associated with this Root Port. Root Complex Event Collectors provide support for the above described functionality for RCiEPs.
FATAL_ERR_REPORTING_EN	2	rw	**Fatal Error Reporting Enable** When Set, this bit enables the generation of an interrupt when a Fatal error is reported by any of the Functions in the Hierarchy Domain associated with this Root Port. Root Complex Event Collectors provide support for the above described functionality for RCiEPs.
RSVDP_3	31:3	r	**Reserved for future use**

19.4.2.145 Root Error Status Register

The Root Error Status Register reports status of error Messages (ERR_COR, ERR_NONFATAL, and ERR_FATAL) received by the Root Port, and of errors detected by the Root Port itself(The Root Port had sent an error Message to itself). In order to update this register, error Messages received by the Root Port and/or internally generated error Messages must be enabled for “transmission” by the primary interface of the Root Port.

AERCAP_ROOT_ROOT_ERR_STATUS_OFF Offset address: 0004130_H

Root Error Status Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADV_ERR_INT_MSG_NUM					RSVDP_7										
rw					r										

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVDP_7									FATAL_ERR_MSG_RX	NON_FATAL_ERR_MSG_RX	FIRST_UNCORR_FATAL	MUL_ERR_FATAL_NON_FATAL_RX	ERR_FATAL_NON_FATAL_RX	MUL_ERR_COR_RX	ERR_COR_RX
r									rw1ch	rw1ch	rw1ch	rw1ch	rw1ch	rw1ch	rw1ch

Field	Bits	Type	Description
ERR_COR_RX	0	rw1ch	**Correctable Error Received** Set when a Correctable error Message is received and this bit is not already Set. Note: This register field is sticky. Software write with 1 clears this bit-field.
MUL_ERR_COR_RX	1	rw1ch	**Multiple Correctable Errors Received** Set when a Correctable error Message is received and ERR_COR_RX is already Set. Note: This register field is sticky. Software write with 1 clears this bit-field.

(table continues...)

(continued)

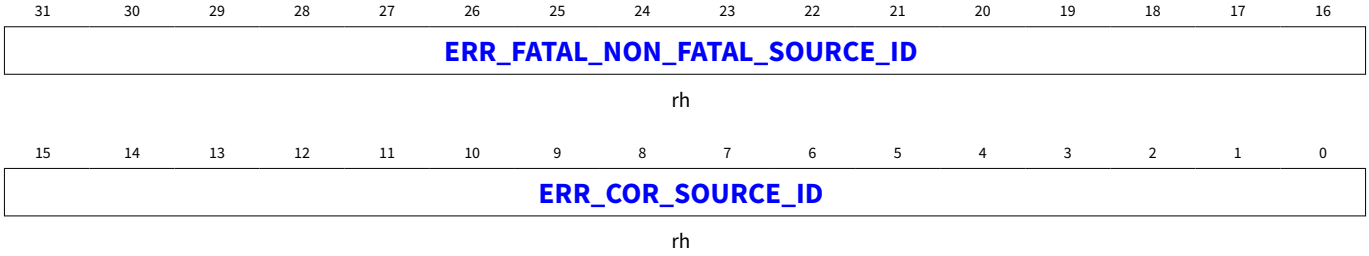
Field	Bits	Type	Description
ERR_FATAL_NON_FATAL_RX	2	rw1ch	**Fatal or Non-Fatal Error Received** Set when either a Fatal or a Non-fatal error Message is received and this bit is not already Set. Note: This register field is sticky. Software write with 1 clears this bit-field.
MUL_ERR_FATAL_NON_FATAL_RX	3	rw1ch	**Multiple Fatal or Non-Fatal Errors Received** Set when either a Fatal or a Non-fatal error is received and ERR_FATAL_NON_FATAL_RX is already Set. Note: This register field is sticky. Software write with 1 clears this bit-field.
FIRST_UNCOR_FATAL	4	rw1ch	**First Uncorrectable Error is Fatal** Set when the first Uncorrectable error Message received is for a Fatal error. Note: This register field is sticky. Software write with 1 clears this bit-field.
NON_FATAL_ERR_MSG_RX	5	rw1ch	**One or more Non-Fatal Error Messages Received** Set when one or more Non-Fatal Uncorrectable error Messages have been received. Note: This register field is sticky. Software write with 1 clears this bit-field.
FATAL_ERR_MSG_RX	6	rw1ch	**One or more Fatal Error Messages Received** Set when one or more Fatal Uncorrectable error Messages have been received. Note: This register field is sticky. Software write with 1 clears this bit-field.
RSVDP_7	26:7	r	**Reserved for future use**
ADV_ERR_INT_MSG_NUM	31:27	rw	**Advanced Error Interrupt Message Number** This register indicates which MSI/MSI-X vector is used for the interrupt message generated in association with any of the status bits of this Capability. For MSI, the value in this register indicates the offset between the base Message Data and the interrupt message that is generated. For MSI-X, the value in this register indicates which MSI-X Table entry is used to generate the interrupt message. The entry must be one of the first 32 entries even if the Function implements more than 32 entries. For a given MSI-X implementation, the entry must remain constant. If MSI-X is enabled, the value in this register must indicate the vector for MSI-X. If MSI is enabled or neither is enabled, the value in this register must indicate the vector for MSI. If software enables both MSI and MSI-X at the same time, the value in this register is undefined. Note: The access attributes of this field are as follows: - Wire: No access. - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky.

19.4.2.146 Error Source Identification Register

The Error Source Identification Register identifies the source (Requester ID) of first correctable and uncorrectable (Non-fatal/Fatal) errors reported in the Root Error Status Register. This register is updated regardless of the settings of the Root Control register and the Root Error Command Register.

AERCAP_ERR_SRC_ID_OFF Offset address: 0004134_H

Error Source Identification Register Kernel Reset value: $0000\ 0000_H$



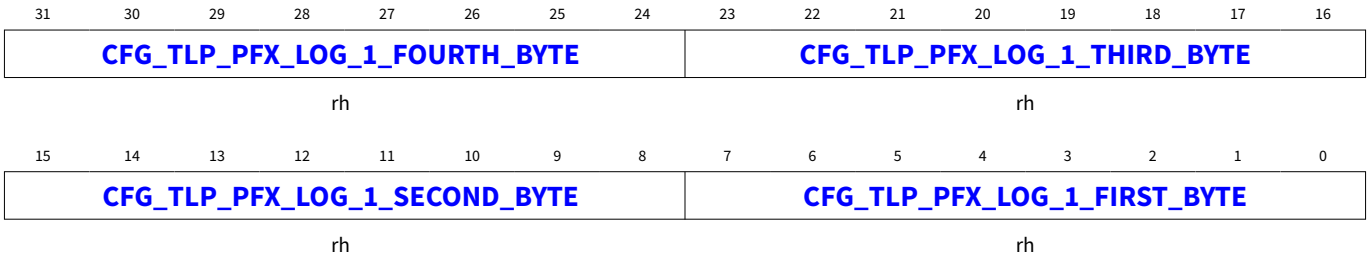
Field	Bits	Type	Description
ERR_COR_SOURCE_ID	15:0	rh	**Source of Correctable Error** Loaded with the Requester ID indicated in the received ERR_COR Message when the ERR_COR Received bit is not already set. Note: This register field is sticky.
ERR_FATAL_NON_FATAL_SOURCE_ID	31:16	rh	**Source of Fatal/Non-Fatal Error** Loaded with the Requester ID indicated in the received ERR_FATAL or ERR_NONFATAL Message when the ERR_FATAL/NONFATAL Received bit is not already set. Note: This register field is sticky.

19.4.2.147 TLP Prefix i Log Register 1

The First TLP Prefix Log Register contains the first End-End TLP Prefix from the TLP corresponding to the detected error. The TLP Prefix Log Register is only meaningful when the TLP Prefix Log Present bit is Set. The TLP Prefixes are captured such that, when read using DW accesses, the fields of the TLP Prefix are laid out in the same way the fields of the TLP Prefix are described. Therefore, byte 0 of a TLP Prefix is located in byte 3 of the associated TLP Prefix Log Register; byte 1 of a TLP Prefix is located in byte 2; and so forth. The TLP Prefix Log Registers beyond the number supported by the Function are hardwired to zero by controller. If the End-End TLP Prefix Supported bit is Clear, the TLP Prefix Log Register is not required to be implemented.

TLP_PREFIX_LOG_i_OFF (i=0-3) Offset address: 0004138_H+i*4

TLP Prefix i Log Register 1 Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
CFG_TLP_PFX_LOG_1_FIRST_BYTE	7:0	rh	**Byte 0 of Error TLP Prefix Log 1** This field contains first byte of First DW of TLP Prefix. Note: This register field is sticky.
CFG_TLP_PFX_LOG_1_SECOND_BYTE	15:8	rh	**Byte 1 of Error TLP Prefix Log 1** This field contains second byte of First DW of TLP Prefix. Note: This register field is sticky.

(table continues...)

(continued)

Field	Bits	Type	Description
CFG_TLP_PFX_LOG_1_THIRD_BYTE	23:16	rh	**Byte 2 of Error TLP Prefix Log 1** This field contains third byte of First DW of TLP Prefix. Note: This register field is sticky.
CFG_TLP_PFX_LOG_1_FOURTH_BYTE	31:24	rh	**Byte 3 of Error TLP Prefix Log 1** This field contains fourth byte of First DW of TLP Prefix. Note: This register field is sticky.

19.4.2.148 Device Serial Number Extended Capability Header

The Device Serial Number Extended Capability is an optional Extended Capability that may be implemented by any PCI Express device Function. This register provides Capability Id, Capability version, and Next offset of Capability structure.

SN_BASE Offset address: 0004148_H

Device Serial Number Extended Capability Header Kernel Reset value: $1581\ 0003_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SN_NEXT_OFFSET												SN_CAP_VERSION			
rw												rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SN_PCIE_EXTENDED_CAP_ID															
rw															

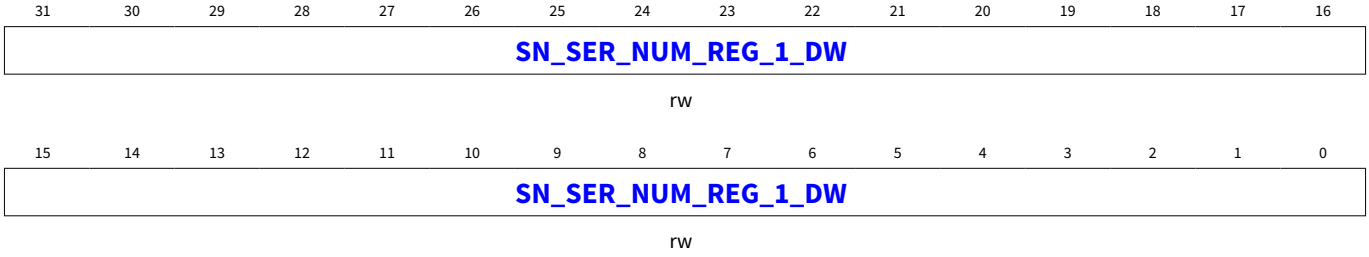
Field	Bits	Type	Description
SN_PCIE_EXTENDED_CAP_ID	15:0	rw	**Serial Number Extended Capability ID** This field is a PCI-SIG defined ID number that indicates the nature and format of the Extended Capability. Extended Capability ID for the Device Serial Number Extended Capability is 0003h. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky.
SN_CAP_VERSION	19:16	rw	**Capability Version** This field is a PCI-SIG defined version number that indicates the version of the Capability structure present. This field is depends on the version of specification. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky.
SN_NEXT_OFFSET	31:20	rw	**Next Capability Offset** This field contains the offset to the next PCI Express Capability structure or 000h if no other items exist in the linked list of Capabilities. For Extended Capabilities implemented in Configuration Space, this offset is relative to the beginning of PCI-compatible Configuration Space and thus must always be either 000h (for terminating list of Capabilities) or greater than 0FFh. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky.

19.4.2.149 Serial Number 1 Register

The Serial Number register is a 64-bit field that contains the IEEE defined 64-bit extended unique identifier.

SER_NUM_REG_DW_1 Offset address: $000414C_H$

Serial Number 1 Register Kernel Reset value: $0000\ 0000_H$



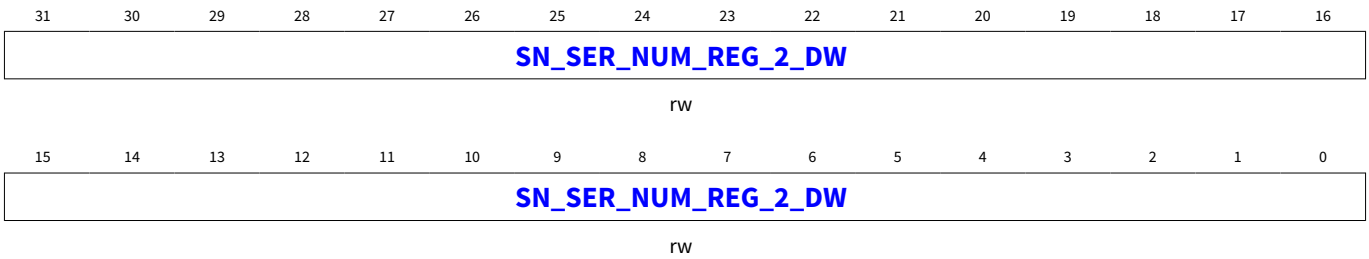
Field	Bits	Type	Description
SN_SER_NUM_REG_1_DW	31:0	rw	**IEEE 64 bit Device Serial Number (DW 1)** This field contains the IEEE defined 64-bit Extended Unique Identifier [EUI-64]. This identifier includes a 24-bit company id value assigned by IEEE registration authority and a 40-bit extension identifier assigned by the manufacturer. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky.

19.4.2.150 Serial Number 2 Register

The Serial Number register is a 64-bit field that contains the IEEE defined 64-bit extended unique identifier.

SER_NUM_REG_DW_2 Offset address: 0004150_H

Serial Number 2 Register Kernel Reset value: $0000\ 0000_H$



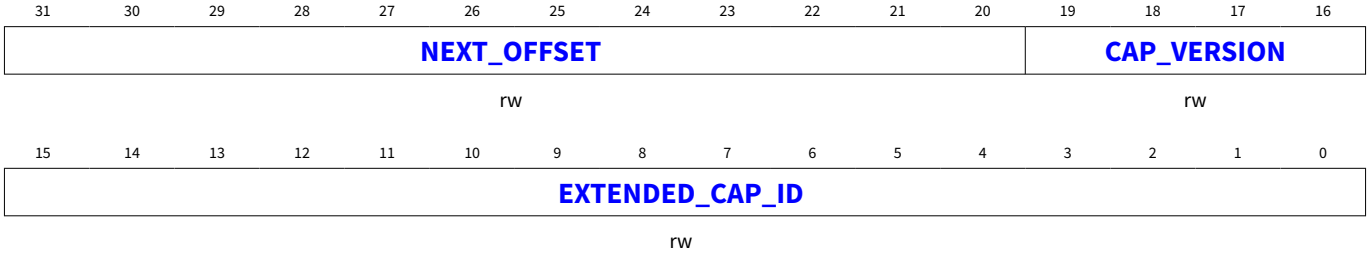
Field	Bits	Type	Description
SN_SER_NUM_REG_2_DW	31:0	rw	**IEEE 64 bit Device Serial Number (DW 2)** This field contains the IEEE defined 64-bit Extended Unique Identifier [EUI-64]. This identifier includes a 24-bit company id value assigned by IEEE registration authority and a 40-bit extension identifier assigned by the manufacturer. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky.

19.4.2.151 SPCIE Capability Header

This Register provides Capability Id, Capability Version, and next Offset of SPCIE Structure.

SPCIE_CAP_HEADER_REG Offset address: 0004158_H

SPCIE Capability Header Kernel Reset value: $1701\ 0019_H$



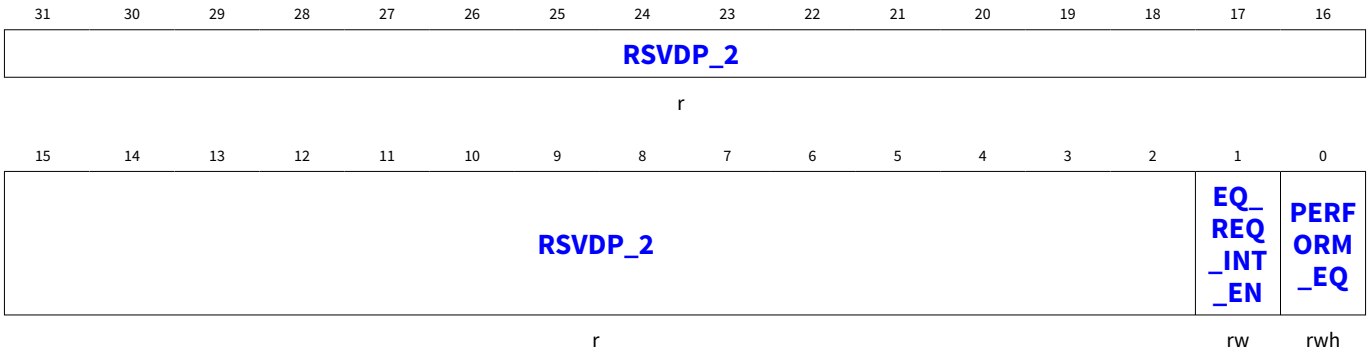
Field	Bits	Type	Description
EXTENDED_CAP_ID	15:0	rw	**Secondary PCI Express Extended Capability ID** This field is a PCI-SIG defined ID number that indicates the nature and format of the Extended Capability. PCI Express Extended Capability ID for the Secondary PCI Express Extended Capability is 0019h. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky. 0000_H **MIN_VAL**: Min value $FFFF_H$ **MAX_VAL**: Max value
CAP_VERSION	19:16	rw	**Capability Version** This field is a PCI-SIG defined version number that indicates the version of the Capability structure present. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky. 0_H **MIN_VAL**: Min value F_H **MAX_VAL**: Max value
NEXT_OFFSET	31:20	rw	**Next Capability Offset** This field contains the offset to the next PCI Express Extended Capability structure or 000h if no other items exist in the linked list of Capabilities. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky. 000_H **MIN_VAL**: Min value FFF_H **MAX_VAL**: Max value

19.4.2.152 Link Control 3 Register

This Register controls equilization and equilization interrupt.

LINK_CONTROL3_REG Offset address: $000415C_H$

Link Control 3 Register Reset values see: Table 892



Field	Bits	Type	Description
PERFORM_EQ	0	rwh	**Perform Equalization** This bit is RW for Downstream Ports and for Upstream Ports when Crosslink Supported is 1b. This bit is not applicable and is RsvdP for Upstream Ports when the Crosslink Supported bit is 0b. Note: The access attributes of this field are as follows: - Wire: RSVDP - Dbi: RSVDP 0_B **CLEAR**: Clear 1_B **SET**: When this bit is 1b and a 1b is written to the Retrain Link bit with the Target Link Speed field set to 8.0 GT/s or higher, the Downstream Port must perform Link Equalization
EQ_REQ_INT_EN	1	rw	**Link Equalization Request Interrupt Enable** This bit is RW for Downstream Ports and for Upstream Ports when Crosslink Supported is 1b. This bit is not applicable and is RsvdP for Upstream Ports when the Crosslink Supported bit is 0b. Note: The access attributes of this field are as follows: - Wire: RSVDP - Dbi: RSVDP 0_B **CLEAR**: Clear 1_B **SET**: When Set, this bit enables the generation of an interrupt to indicate that the Link Equalization Request 8.0 GT/s bit, the Link Equalization Request 16.0 GT/s bit, or the Link Equalization Request 32.0 GT/s bit has been set
RSVDP_2	31:2	r	**Reserved for future use**

Table 892 Reset values of LINK_CONTROL3_REG

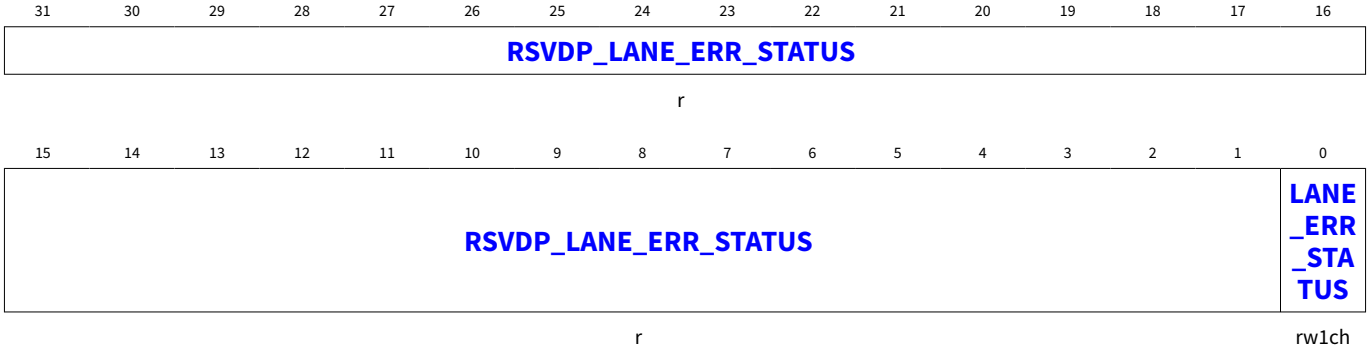
Reset	Reset value	Note
Kernel Reset	0000 0000 0000 0000 0000 0000 0000 000X_B	

19.4.2.153 Lane Error Status Register

This Register contains Lane Error Status Bits per Lane.

LANE_ERR_STATUS_REG Offset address: 0004160_H

Lane Error Status Register Kernel Reset value: $0000\ 0000_H$



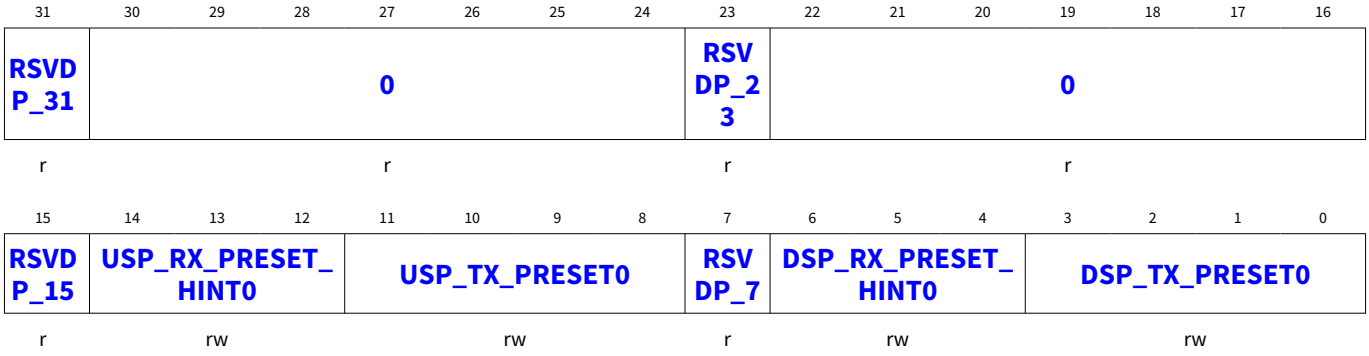
Field	Bits	Type	Description
LANE_ERR_STATUS	0	rw1ch	**Lane Error Status Bits per Lane** Each bit indicates if the corresponding Lane detected a Lane-based error. A value of 1b indicates that a Lane based-error was detected on the corresponding Lane Number. For Ports that are narrower than 32 Lanes, the unused upper bits [31: Maximum Link Width] are RsvdZ. Note: This register field is sticky. Software write with 1 clears this bit-field.
RSVDP_LANE_ERR_STATUS	31:1	r	**Reserved for future use**

19.4.2.154 Lane Equalization Control Register for lanes 1 and 0

This register provides Transmitter Preset and Receiver Preset Hint for Downstream Port and Upstream Port.

SPCIE_CAP_OFF_0CH_REG Offset address: 0004164_H

Lane Equalization Control Register for lanes 1 and 0 Kernel Reset value: $0000\ 7777_H$



Field	Bits	Type	Description
DSP_TX_PRESET0	3:0	rw	**Downstream Port 8.0 GT/s Transmitter Preset 0** Transmitter preset 0 value used for 8.0 GT/s equalization by this Port when the Port is operating as a Downstream Port. This field is ignored when the Port is operating as an Upstream Port. For an Upstream Port if Crosslink Supported is 0b, this field is RsvdP. Otherwise, this field is HwInit. Note: The access attributes of this field are as follows: - Wire: RSVDP - Dbi: RSVDP

(table continues...)

(continued)

Field	Bits	Type	Description
DSP_RX_PRESET_HINT0	6:4	rw	**Downstream Port 8.0 GT/s Receiver Preset Hint 0** Receiver preset hint 0 value that may be used as a suggested setting for 8.0 GT/s receiver equalization by this Port when the Port is operating as a Downstream Port. This field is ignored when the Port is operating as an Upstream Port. For an Upstream Port if Crosslink Supported is 0b, this field is RsvdP. Otherwise, this field is HwInit. Note: The access attributes of this field are as follows: - Wire: RSVDP - Dbi: RSVDP
RSVDP_7	7	r	**Reserved for future use**
USP_TX_PRESET0	11:8	rw	**Upstream Port 8.0 GT/s Transmitter Preset 0** The write value is gated with the PCIE_CAP_CROSS_LINK_SUPPORT field of LINK_CAPABILITIES2_REG. Field contains the Transmitter preset 0 value sent or received during 8.0 GT/s Link Equalization. • Case A: When Operating port is Downstream port and whether the Crosslink is supported or not, this field representss the value sent on Lane 0 during 8.0 GT/s equalization • Case B: When Operating port is Upstream port and Crosslink Supported bit is 0, this field is intended for debug and diagnostics. It contains the value captured from Lane 0 during Link Equalization • Case C: When Operating port is Upstream port and Crosslink Supported bit is 1, Field is not used or affected by the current Link Equalization. Field value will be used if a future crosslink negotiation switches the Operating Port Direction so that case A applies For case A and C, Field is HwInit. For case B, Field is RO. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: R (sticky) Note: This register field is sticky.
USP_RX_PRESET_HINT0	14:12	rw	**Upstream Port 8.0 GT/s Receiver Preset Hint 0** The write value is gated with the PCIE_CAP_CROSS_LINK_SUPPORT field of LINK_CAPABILITIES2_REG. Field contains the Receiver preset hint 0 value sent or received during 8.0 GT/s Link Equalization. • Case A: When Operating port is Downstream port and whether the Crosslink is supported or not, this field representss the value sent on Lane 0 during 8.0 GT/s equalization • Case B: When Operating port is Upstream port and Crosslink Supported bit is 0, this field is intended for debug and diagnostics. It contains the value captured from Lane 0 during Link Equalization • Case C: When Operating port is Upstream port and Crosslink Supported bit is 1, Field is not used or affected by the current Link Equalization. Field value will be used if a future crosslink negotiation switches the Operating Port Direction so that case A applies For case A and C, Field is HwInit. For case B, Field is RO. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: R (sticky) Note: This register field is sticky.
RSVDP_15	15	r	**Reserved for future use**
RSVDP_23	23	r	**Reserved for future use**
RSVDP_31	31	r	**Reserved for future use**

(table continues...)

(continued)

Field	Bits	Type	Description
0	22:16, 30:24	r	**Reserved** Read as all 0's; should be written with all 0's.

19.4.2.155 L1 Substates Extended Capability Header

L1 Substates Extended Capability Header provides capability ID, capability version and next offset value for L1 Substates.

L1SUB_CAP_HEADER_REG Offset address: 0004170_H

L1 Substates Extended Capability Header Kernel Reset value: $1801\ 001E_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
NEXT_OFFSET												CAP_VERSION			
rw												rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EXTENDED_CAP_ID															
rw															

Field	Bits	Type	Description
EXTENDED_CAP_ID	15:0	rw	**L1SUB Extended Capability ID** This field is a PCI-SIG defined ID number that indicates the nature and format of the Extended Capability. Extended Capability ID for L1 PM Substates is 001Eh. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky.
CAP_VERSION	19:16	rw	**Capability Version** This field is a PCI-SIG defined version number that indicates the version of the Capability structure present. This field is depends on version of the specification. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky.
NEXT_OFFSET	31:20	rw	**Next Capability Offset** This field contains the offset to the next PCI Express Capability structure or 000h, if no other items exist in the linked list of Capabilities. For Extended Capabilities implemented in Configuration Space, this offset is relative to the beginning of PCI-compatible Configuration Space and thus must always be either 000h (for terminating list of Capabilities) or greater than 0FFh. The bottom 2 bits of this offset are Reserved and must be implemented as 00b although software must mask them to allow for future uses of these bits. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky.

19.4.2.156 L1 Substates Capability Register

This register provides extended capability of L1 Substates.

L1SUB_CAPABILITY_REG Offset address: 0004174_H

L1 Substates Capability Register Kernel Reset value: $0038\ 0A1F_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_24								PWR_ON_VALUE_SUPPORT					RSVDP_18	PWR_ON_SCALE_SUPPORT	
r								rw					r	rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
COMM_MODE_SUPPORT								RSVDP_5			L1_PMSUB_SUPPORT	L1_1_ASPM_SUPPORT	L1_2_ASPM_SUPPORT	L1_1_PCIPM_SUPPORT	L1_2_PCIPM_SUPPORT
rw								r			rw	rw	rw	rw	rw

Field	Bits	Type	Description
L1_2_PCIPM_SUPPORT	0	rw	**PCI-PM L12 Supported** When Set this field indicates that PCI-PM L1.2 is supported. Note: The access attributes of this field are as follows: - Wire: HWINIT - Dbi: R/W (sticky)
L1_1_PCIPM_SUPPORT	1	rw	**PCI-PM L11 Supported** When Set this field indicates that PCI-PM L1.1 is supported, and must be Set by all Ports implementing L1 PM Substates. Note: The access attributes of this field are as follows: - Wire: HWINIT - Dbi: R/W (sticky)
L1_2_ASPM_SUPPORT	2	rw	**ASPM L12 Supported** When Set this field indicates that ASPM L1.2 is supported. Note: The access attributes of this field are as follows: - Wire: HWINIT - Dbi: R/W (sticky)
L1_1_ASPM_SUPPORT	3	rw	**ASPM L11 Supported** When Set this field indicates that ASPM L1.1 is supported. Note: The access attributes of this field are as follows: - Wire: HWINIT - Dbi: R/W (sticky)
L1_PMSUB_SUPPORT	4	rw	**L1 PM Substates ECN Supported** When Set this field indicates that this Port supports L1 PM Substates. Note: The access attributes of this field are as follows: - Wire: HWINIT - Dbi: R/W (sticky)
RSVDP_5	7:5	r	**Reserved for future use**

(table continues...)

(continued)

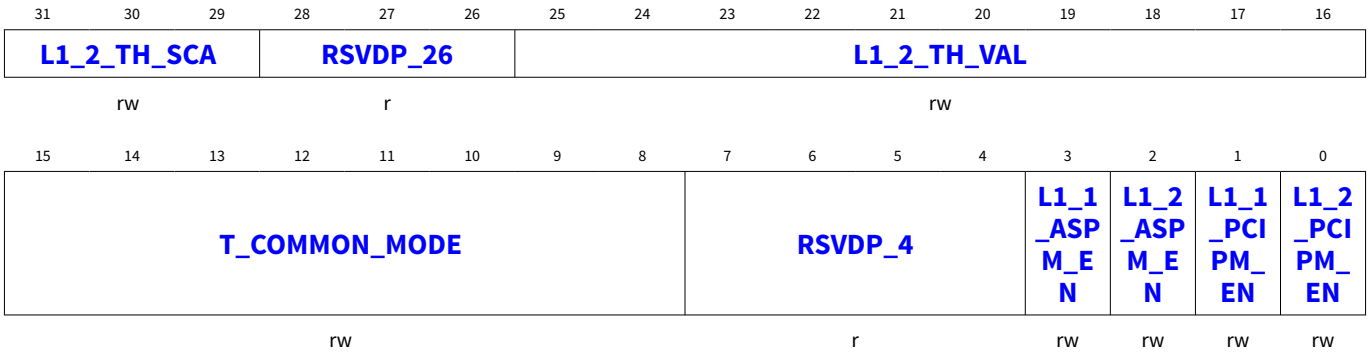
Field	Bits	Type	Description
COMM_MODE_SUPPORT	15:8	rw	**Port Common Mode Restore Time** Time (in µs) required for this Port to re-establish common mode. Required for all Ports for which either the PCI-PM L1.2 Supported bit is Set, ASPM L1.2 Supported bit is Set, or both are Set, otherwise this field is of type RsvdP. Note: The access attributes of this field are as follows: - Wire: HWINIT - Dbi: L1SUB_CAPABILITY_REG.L1_2_ASPM_SUPPORT|| L1SUB_CAPABILITY_REG.L1_2_PCIPM_SUPPORT ? RWS : RSVDP
PWR_ON_SCALE_SUPPORT	17:16	rw	**Port T Power On Scale** Specifies the scale used for the Port T_POWER_ON_VALUE field in the L1 PM Substates Capabilities register. Range of values are given below. Required for all Ports for which either the PCI-PM L1.2 Supported bit is Set, ASPM L1.2 Supported bit is Set, or both are Set, otherwise this field is of type RsvdP. Note: The access attributes of this field are as follows: - Wire: HWINIT - Dbi: L1SUB_CAPABILITY_REG.L1_2_ASPM_SUPPORT|| L1SUB_CAPABILITY_REG.L1_2_PCIPM_SUPPORT ? RWS : RSVDP 00_B **_2us**: Scale is 2 µs 01_B **_10us**: Scale is 10 µs 10_B **_100us**: Scale is 100 µs 11_B **Reserved**: Reserved
RSVDP_18	18	r	**Reserved for future use**
PWR_ON_VALUE_SUPPORT	23:19	rw	**Port T Power On Value** Along with the Port T_POWER_ON_SCALE field in the L1 PM Substates Capabilities register sets the time (in µs) that this Port requires the port on the opposite side of Link to wait in L1.2.Exit after sampling CLKREQ# asserted before actively driving the interface. The value of Port T_POWER_ON is calculated by multiplying the value in this field by the scale value in the Port T_POWER_ON_SCALE field in the L1 PM Substates Capabilities register. Required for all Ports for which either the PCI-PM L1.2 Supported bit is Set, ASPM L1.2 Supported bit is Set, or both are Set, otherwise this field is of type RsvdP. Note: The access attributes of this field are as follows: • Wire: HWINIT • Dbi: L1SUB_CAPABILITY_REG.L1_2_ASPM_SUPPORT|| L1SUB_CAPABILITY_REG.L1_2_PCIPM_SUPPORT ? RWS : RSVDP
RSVDP_24	31:24	r	**Reserved for future use**

19.4.2.157 L1 Substates Control 1 Register

This register Controls that the individual extended capability is enabled ot not.

L1SUB_CONTROL1_REG Offset address: 0004178_H

L1 Substates Control 1 Register Kernel Reset value: $0000\ 0A00_H$



Field	Bits	Type	Description
L1_2_PCIPM_EN	0	rw	**PCI-PM L12 Enable** When Set this field, enables PCI-PM L1.2. For Ports for which the PCI-PM L1.2 Supported bit is clear, this bit is permitted to be hardwired to 0. Required for both Upstream and Downstream Ports. For compatibility with possible future extensions, software must not enable L1 PM Substates unless the L1 PM Substates Supported field in the L1 PM Substates Capabilities Register is Set.
L1_1_PCIPM_EN	1	rw	**PCI-PM L11 Enable** When Set this field, enables PCI-PM L1.1. Required for both Upstream and Downstream Ports. For compatibility with possible future extensions, software must not enable L1 PM Substates unless the L1 PM Substates Supported field in the L1 PM Substates Capabilities Register is Set.
L1_2_ASPM_EN	2	rw	**ASPM L12 Enable** When Set this field, enables ASPM L1.2. For Ports for which the ASPM L1.2 Supported bit is clear, this bit is permitted to be hardwired to 0. Required for both Upstream and Downstream Ports. For compatibility with possible future extensions, software must not enable L1 PM Substates unless the L1 PM Substates Supported field in the L1 PM Substates Capabilities Register is Set.
L1_1_ASPM_EN	3	rw	**ASPM L11 Enable** When Set this field, enables ASPM L1.1. For Ports for which the ASPM L1.1 Supported bit is clear, this bit is permitted to be hardwired to 0. Required for both Upstream and Downstream Ports. For compatibility with possible future extensions, software must not enable L1 PM Substates unless the L1 PM Substates Supported field in the L1 PM Substates Capabilities Register is Set.
RSVDP_4	7:4	r	**Reserved for future use**
T_COMMON_MODE	15:8	rw	**Common Mode Restore Time** Sets value of TCOMMONMODE (in µs), which must be used by the Downstream Ports for timing the re-establishment of common mode. This field is of type RsvdP for Upstream Ports.

(table continues...)

(continued)

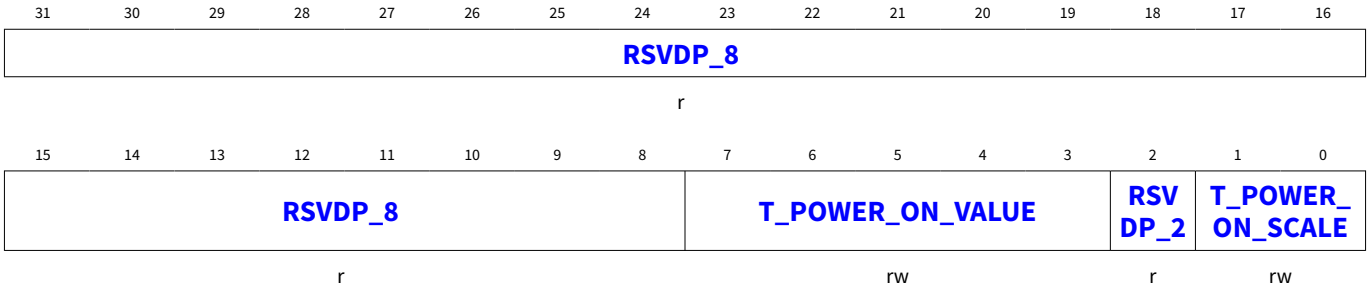
Field	Bits	Type	Description
L1_2_TH_VAL	25:16	rw	**LTR L12 Threshold Value** Along with the LTR_L1.2_THRESHOLD_Scale, this field indicates the LTR threshold used to determine if entry into L1 results in L1.1 (if enabled) or L1.2 (if enabled). Required for all Ports for which the ASPM L12 Supported bit is Set, otherwise this field is of type RsvdP. Note: The access attributes of this field are as follows • Wire: L1SUB_CAPABILITY_REG.L1_2_ASPM_SUPPORT ? RW : RSVDP • Dbi: L1SUB_CAPABILITY_REG.L1_2_ASPM_SUPPORT ? RW : RSVDP
RSVDP_26	28:26	r	**Reserved for future use**
L1_2_TH_SCA	31:29	rw	**LTR L12 Threshold Scale** This field provides a scale for the value contained within the LTR_L1.2_THRESHOLD_Value. Hardware operation is undefined if software writes a Not-Permitted value to this field. Required for all Ports Ports for which the ASPM L12 Supported bit is Set, otherwise this field is of type RsvdP. Note: The access attributes of this field are as follows: • Wire: L1SUB_CAPABILITY_REG.L1_2_ASPM_SUPPORT ? RW : RSVDP • Dbi: L1SUB_CAPABILITY_REG.L1_2_ASPM_SUPPORT ? RW : RSVD

19.4.2.158 L1 Substates Control 2 Register

This register Controls that the individual extended capability is enabled ot not.

L1SUB_CONTROL2_REG Offset address: $000417C_H$

L1 Substates Control 2 Register Kernel Reset value: $0000\ 0028_H$



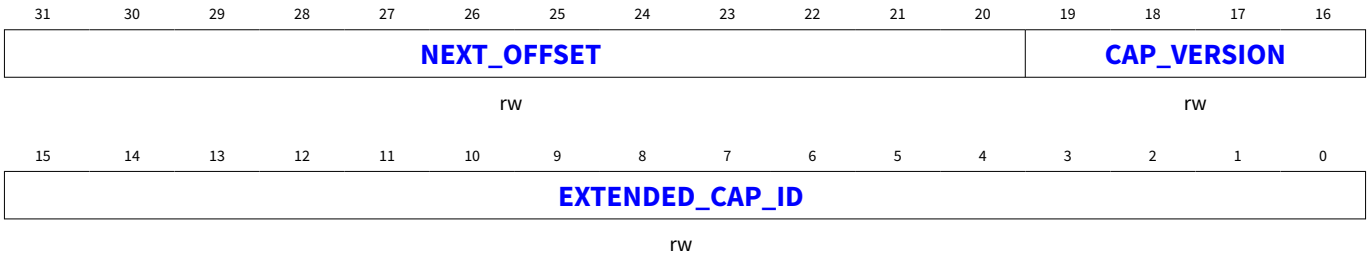
Field	Bits	Type	Description
T_POWER_ON_SCALE	1:0	rw	**T Power On Scale** Specifies the scale used for T_POWER_ON_VALUE. Range of values are given below. Required for all Ports that support L1.2, otherwise this field is of type RsvdP. This field must only be modified when the ASPM L1.2 Enable and PCI-PM L1.2 Enable bits are both Clear. The Port behavior is undefined if this field is modified when either the ASPM L1.2 Enable and/or PCI-PM L1.2 Enable bit(s) are set. Note: The access attributes of this field are as follows: • Wire: L1SUB_CAPABILITY_REG.L1_2_ASPM_SUPPORT|| L1SUB_CAPABILITY_REG.L1_2_PCIPM_SUPPORT ? RWS : RSVDP • Dbi: L1SUB_CAPABILITY_REG.L1_2_ASPM_SUPPORT|| L1SUB_CAPABILITY_REG.L1_2_PCIPM_SUPPORT ? RWS : RSVDP 00_B **_2us**: Scale is 2 µs 01_B **_10us**: Scale is 10 µs 10_B **_100us**: Scale is 100 µs 11_B **Reserved**: Reserved
RSVDP_2	2	r	**Reserved for future use**
T_POWER_ON_VALUE	7:3	rw	**T Power On Value** Along with the T_POWER_ON_SCALE sets the minimum amount of time (in µs) that the Port must wait in L1.2.Exit after sampling CLKREQ# asserted before actively driving the interface. T_POWER_ON is calculated by multiplying the value in this field by the value in the T_POWER_ON_SCALE field. Required for all Ports that support L1.2, otherwise this field is of type RsvdP.This field must only be modified when the ASPM L1.2 Enable and PCI-PM L1.2 Enable bits are both Clear. The Port behavior is undefined if this field is modified when either the ASPM L1.2 Enable and/or PCI-PM L1.2 Enable bit(s) are set. Note: The access attributes of this field are as follows: • Wire: L1SUB_CAPABILITY_REG.L1_2_ASPM_SUPPORT|| L1SUB_CAPABILITY_REG.L1_2_PCIPM_SUPPORT ? RWS : RSVDP • Dbi: L1SUB_CAPABILITY_REG.L1_2_ASPM_SUPPORT|| L1SUB_CAPABILITY_REG.L1_2_PCIPM_SUPPORT ? RWS : RSVDP
RSVDP_8	31:8	r	**Reserved for future use**

19.4.2.159 Vendor-Specific Extended Capability Header

The Vendor-Specific Extended Capability (VSEC Capability) is an optional Extended Capability that is permitted to be implemented by any PCI Express Function or RCRB. This Register contains Capability Id, Capability Version and Next Offset value for Vendor-Specific Extended Capability.

RAS_DES_CAP_HEADER_REG Offset address: 0004180_H

Vendor-Specific Extended Capability Header Kernel Reset value: $2801\ 000B_H$



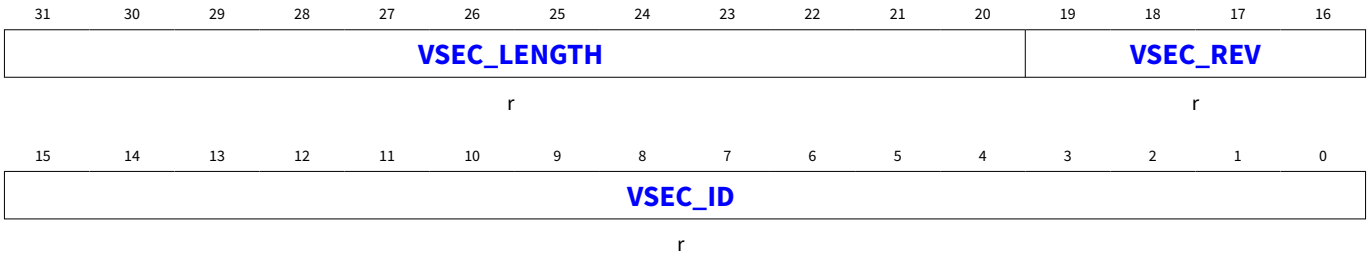
Field	Bits	Type	Description
EXTENDED_CAP_ID	15:0	rw	**PCI Express Extended Capability ID** This field is a PCI-SIG defined ID number that indicates the nature and format of the Extended Capability. Extended Capability ID for the Vendor-Specific Extended Capability is 000Bh. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky. 0000_H **MIN_VAL**: Min value $FFFF_H$ **MAX_VAL**: Max value
CAP_VERSION	19:16	rw	**Capability Version** This field is a PCI-SIG defined version number that indicates the version of the Capability structure present. Value of this field is depends on the version of the specification. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky. 0_H **MIN_VAL**: Min value F_H **MAX_VAL**: Max value
NEXT_OFFSET	31:20	rw	**Next Capability Offset** This field contains the offset to the next PCI Express Capability structure or 000h if no other items exist in the linked list of Capabilities. For Extended Capabilities implemented in Configuration Space, this offset is relative to the beginning of PCI-compatible Configuration Space and thus must always be either 000h (for terminating list of Capabilities) or greater than 0FFh. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky. 000_H **MIN_VAL**: Min value FFF_H **MAX_VAL**: Max value

19.4.2.160 Vendor-Specific Header

This Register field provides VSEC Length, VSEC ID and VSEC Rev(Version Number). Vendor-specific software must qualify the associated Vendor ID of the PCI Express Function or RCRB before attempting to interpret the values in the VSEC ID or VSEC Rev fields.

VENDOR_SPECIFIC_HEADER_REG Offset address: 0004184_H

Vendor-Specific Header Kernel Reset value: $1004\ 0002_H$



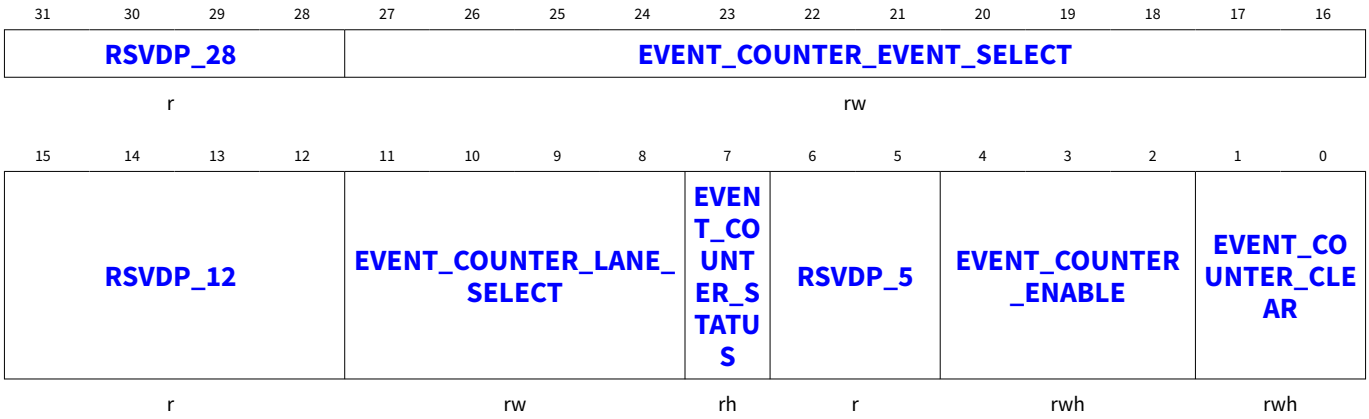
Field	Bits	Type	Description
VSEC_ID	15:0	r	**VSEC ID** This field is a vendor-defined ID number that indicates the nature and format of the VSEC structure. Software must qualify the Vendor ID before interpreting this field. 0000_H **MIN_VAL**: Min value $FFFF_H$ **MAX_VAL**: Max value
VSEC_REV	19:16	r	**VSEC Rev** This field is a vendor-defined version number that indicates the version of the VSEC structure. Software must qualify the Vendor ID and VSEC ID before interpreting this field. 0_H **MIN_VAL**: Min value F_H **MAX_VAL**: Max value
VSEC_LENGTH	31:20	r	**VSEC Length** This field indicates the number of bytes in the entire VSEC structure, including the Vendor-Specific Extended Capability Header, the Vendor-Specific Header, and the vendor-specific registers. 000_H **MIN_VAL**: Min value FFF_H **MAX_VAL**: Max value

19.4.2.161 Event Counter Control

This is a viewport control register. - Setting the EVENT_COUNTER_EVENT_SELECT and EVENT_COUNTER_LANE_SELECT fields in this register determine the Event Counter data returned by the EVENT_COUNTER_DATA_REG viewport register. - Setting the EVENT_COUNTER_ENABLE field in this register enables the Event Counter selected by the EVENT_COUNTER_EVENT_SELECT and EVENT_COUNTER_LANE_SELECT fields in this register. - Setting the EVENT_COUNTER_CLEAR field in this register clears the Event Counter selected by the EVENT_COUNTER_EVENT_SELECT and EVENT_COUNTER_LANE_SELECT fields in this register. - Reading the EVENT_COUNTER_STATUS field in this register returns the Enable status of the Event Counter selected by the EVENT_COUNTER_EVENT_SELECT and EVENT_COUNTER_LANE_SELECT fields in this register.

EVENT_COUNTER_CONTROL_REG Offset address: 0004188_H

Event Counter Control Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
EVENT_COUNTER_CLEAR	1:0	rwh	**Event Counter Clear** Clears the Event Counter selected by the EVENT_COUNTER_EVENT_SELECT and EVENT_COUNTER_LANE_SELECT fields in this register. You can clear the value of a specific Event Counter by writing the 'per clear' code and you can clear all event counters at once by writing the 'all clear' code. The read value is always '0'. Other values are reserved. 00_B **NO_CHANGE**: no change 01_B **PER_CLEAR**: per clear 10_B **NO_CHANGE_2**: no change 11_B **ALL_CLEAR**: all clear
EVENT_COUNTER_ENABLE	4:2	rwh	**Event Counter Enable** Enables/disables the Event Counter selected by the EVENT_COUNTER_EVENT_SELECT and EVENT_COUNTER_LANE_SELECT fields in this register. By default, all event counters are disabled. You can enable/disable a specific Event Counter by writing the 'per event off' or 'per event on' codes. You can enable/disable all event counters by writing the 'all on' or 'all off' codes. The read value is always '0'. For other values no change. 001_B **PER_EVENT_OFF**: per event off 011_B **PER_EVENT_ON**: per event on 101_B **ALL_OFF**: all off 111_B **ALL_ON**: all on
RSVDP_5	6:5	r	**Reserved for future use**
EVENT_COUNTER_STATUS	7	rh	**Event Counter Status** This register returns the current value of the Event Counter selected by the following fields: - EVENT_COUNTER_EVENT_SELECT - EVENT_COUNTER_LANE_SELECT Note: This register field is sticky. 0_B **CLEAR**: Clear 1_B **SET**: Set

(table continues...)

(continued)

Field	Bits	Type	Description
EVENT_COUNTER_LANE_SELECT	11:8	rw	**Event Counter Lane Select** This field in conjunction with EVENT_COUNTER_EVENT_SELECT indexes the Event Counter data returned by the EVENT_COUNTER_DATA_REG register. Note: This register field is sticky. 0_H **MIN_VAL**: Lane0 F_H **MAX_VAL**: Lane15
RSVDP_12	15:12	r	**Reserved for future use**
EVENT_COUNTER_EVENT_SELECT	27:16	rw	**Event Counter Data Select** This field in conjunction with the EVENT_COUNTER_LANE_SELECT field indexes the Event Counter data returned by the EVENT_COUNTER_DATA_REG register • 27-24: Group number(4-bit: 0..0x7) • 23-16: Event number(8-bit: 0..0x13) within the Group Note: This register field is sticky. 000_H **EBUF_OVERFLOW**: Ebuf Overflow 001_H **EBUF_UNDERRUN**: Ebuf Underrun 700_H **TX_MEM_WRITE**: Tx Memory Write 713_H **RX_MSG_TLP**: Rx Message TLP **others,** Reserved
RSVDP_28	31:28	r	**Reserved for future use**

19.4.2.162 Event Counter Data

This viewport register returns the data selected by the following fields: - EVENT_COUNTER_EVENT_SELECT in EVENT_COUNTER_CONTROL_REG - EVENT_COUNTER_LANE_SELECT in EVENT_COUNTER_CONTROL_REG

EVENT_COUNTER_DATA_REG Offset address: $000418C_H$

Event Counter Data Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EVENT_COUNTER_DATA															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EVENT_COUNTER_DATA															
rh															

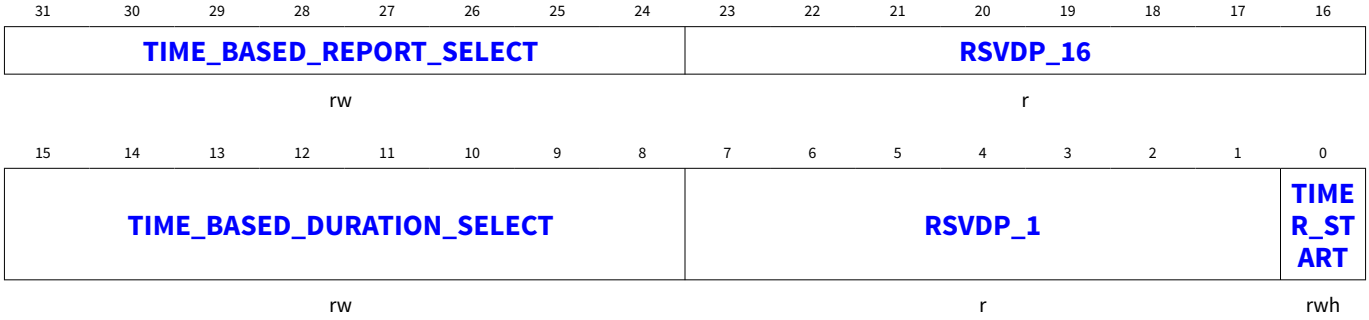
Field	Bits	Type	Description
EVENT_COUNTER_DATA	31:0	rh	**Event Counter Data** This register returns the data selected by the following fields: - EVENT_COUNTER_EVENT_SELECT in EVENT_COUNTER_CONTROL_REG Note: This register field is sticky. 00000000_H **MIN_VAL**: Min value $FFFFFFFF_H$ **MAX_VAL**: Max value

19.4.2.163 Time-based Analysis Control

Used for controlling the measurement of RX/TX data throughput and time spent in each low-power LTSSM state.

TIME_BASED_ANALYSIS_CONTROL_REG Offset address: 0004190_H

Time-based Analysis Control Kernel Reset value: $0000\ 0100_H$



Field	Bits	Type	Description
TIMER_START	0	rwh	**Timer Start** This bit will be cleared automatically when the measurement is finished. Note: This register field is sticky. 0_B **STOP**: Stop 1_B **START_RESTART**: Start/Restart
RSVDP_1	7:1	r	**Reserved for future use**
TIME_BASED_DURATION_SELECT	15:8	rw	**Time-based Duration Select** Selects the duration of time-based analysis. When "manual control" is selected and TIMER_START is set to '1', this analysis never stops until TIMER_STOP is set to '0'. All encodings other than the defined encodings are reserved. Note: This register field is sticky. 00_H **MANUAL**: Manual control 01_H **_1_MS**: 1ms 02_H **_2_MS**: 10ms 03_H **_100_MS**: 100ms 04_H **_1_S**: 1s 05_H **_2_S**: 2s 06_H **_4_S**: 4s FF_H **_4_US**: 4 µs (Debug purpose)
RSVDP_16	23:16	r	**Reserved for future use**

(table continues...)

(continued)

Field	Bits	Type	Description
TIME_BASED_REPORT_SELECT	31:24	rw	**Time-based Report Select** Selects what type of data is measured for the selected duration (TIME_BASED_DURATION_SELECT), and returned in TIME_BASED_ANALYSIS_DATA. Each type of data is measured using one of three types of units: • Core_clk Cycles for 2.5GT/s, 5.0GT/s, 8.0GT/s. Total time in ps is [Value of TIME_BASED_ANALYSIS_DATA returned when TIME_BASED_REPORT_SELECT=0x00] * TIME_BASED_ANALYSIS_DATA. Values 0x0-0x4 and 0x7-0x8 correspond to Core_clk Cycles for 2.5GT/s, 5.0GT/s, 8.0GT/s. • Aux_clk Cycles. Total time in ps is [Period of platform specific clock] * TIME_BASED_ANALYSIS_DATA. Values 0x5, 0x6, and 0x9 correspond to aux_clk Cycles • Data Bytes. Actual amount of bytes is 16 * TIME_BASED_ANALYSIS_DATA. Values 0x20-0x23 correspond to data bytes All encodings other than the defined encodings are reserved. Note: This register field is sticky. 00_H **ONE_CYCLE**: Duration of 1 cycle 01_H **TX_L0S**: TxL0s 02_H **RX_L0S**: RxL0s 03_H **L0**: L0 04_H **L1**: L1 05_H **L1_1**: L1.1 06_H **L1_2**: L1.2 07_H **CFG_RCVRY**: Configuration/Recovery 08_H **TX_RX_L0S**: TxL0s and RxL0s 09_H **L1_AUX**: L1 aux 10_H **_1_CYCLE**: Duration of 1 cycle 11_H **TX_L0S_**: TxL0s 12_H **RX_L0S_**: RxL0s 13_H **L0_**: L0 14_H **L1_**: L1 17_H **CFG_RCVRY_**: Configuration/Recovery 18_H **TX_RX_L0S_**: TxL0s and RxL0s 20_H **TX_PCIE_TLP**: Tx PCIe TLP data payload Bytes 21_H **RX_PCIE_TLP**: Rx PCIe TLP data payload Bytes 22_H **TX_CCIX_TLP**: Tx CCIX TLP data payload Bytes 23_H **RX_CCIX_TLP**: Rx CCIX TLP data payload Bytes **others,** Reserved

19.4.2.164 Time-based Analysis Data

Contains the measurement results of RX/TX data throughput and time spent in each low-power LTSSM state. This viewport register returns the data selected by the TIME_BASED_REPORT_SELECT field in TIME_BASED_ANALYSIS_CONTROL_REG.

TIME_BASED_ANALYSIS_DATA_REG Offset address: 0004194_H

Time-based Analysis Data Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TIME_BASED_ANALYSIS_DATA															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TIME_BASED_ANALYSIS_DATA															
rh															

Field	Bits	Type	Description
TIME_BASED_ANALYSIS_DATA	31:0	rh	**Time Based Analysis Data** This register returns the data selected by the TIME_BASED_REPORT_SELECT field in TIME_BASED_ANALYSIS_CONTROL_REG. The results are cleared when next measurement starts. Note: This register field is sticky. 00000000_H **MIN_VAL**: Min value $FFFFFFFF_H$ **MAX_VAL**: Max value

19.4.2.165 Upper 32 bits of Time-based Analysis Data

This viewport register returns the data selected by the TIME_BASED_REPORT_SELECT field in TIME_BASED_ANALYSIS_CONTROL_REG.

TIME_BASED_ANALYSIS_DATA_63_32_REG Offset address: 0004198_H

Upper 32 bits of Time-based Analysis Data Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TIME_BASED_ANALYSIS_DATA_63_32															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TIME_BASED_ANALYSIS_DATA_63_32															
rh															

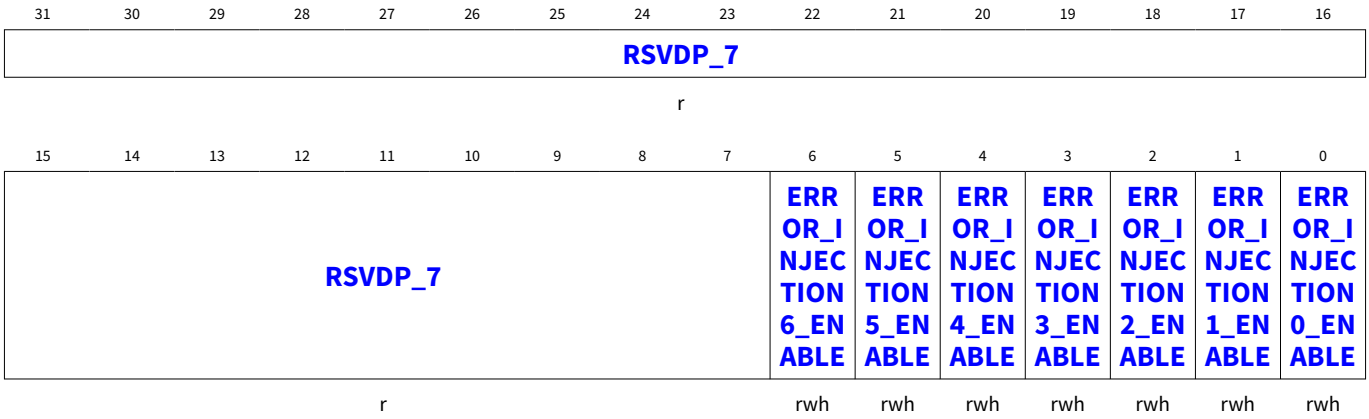
Field	Bits	Type	Description
TIME_BASED_ANALYSIS_DATA_63_32	31:0	rh	**Upper 32 bits of Time Based Analysis Data** Note: This register field is sticky. 00000000_H **MIN_VAL**: Min value $FFFFFFFF_H$ **MAX_VAL**: Max value

19.4.2.166 Error Injection Enable

Each type of error insertion is enabled by the corresponding bit in this register. The specific injection controls for each type of error are defined in the following registers: - 0: CRC Error: EINJ0_CRC_REG - 1: Sequence Number Error: EINJ1_SEQNUM_REG - 2: DLLP Error: EINJ2_DLLP_REG - 3: Symbol DataK Mask Error or Sync Header Error: EINJ3_SYMBOL_REG - 4: FC Credit Update Error: EINJ4_FC_REG - 5: TLP Duplicate/Nullify Error: EINJ5_SP_TLP_REG - 6: Specific TLP Error: EINJ6_COMPARE_*_REG/EINJ6_CHANGE_*_REG/EINJ6_TLP_REG After the errors have been inserted by controller, it will clear each bit here.

EINJ_ENABLE_REG Offset address: $00041B0_H$

Error Injection Enable Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
ERROR_INJECTION0_ENABLE	0	rwh	**Error Injection0 Enable (CRC Error)** Enables insertion of errors into various CRC. For more details, see the EINJ0_CRC_REG register. Note: This register field is sticky. 0_B **DISABLE**: Disable 1_B **ENABLE**: Enable
ERROR_INJECTION1_ENABLE	1	rwh	**Error Injection1 Enable (Sequence Number Error)** Enables insertion of errors into sequence numbers. For more details, see the EINJ1_SEQNUM_REG register. Note: This register field is sticky. 0_B **DISABLE**: Disable 1_B **ENABLE**: Enable
ERROR_INJECTION2_ENABLE	2	rwh	**Error Injection2 Enable (DLLP Error)** Enables insertion of DLLP errors. For more details, see the EINJ2_DLLP_REG register. Note: This register field is sticky. 0_B **DISABLE**: Disable 1_B **ENABLE**: Enable
ERROR_INJECTION3_ENABLE	3	rwh	**Error Injection3 Enable (Symbol DataK Mask Error or Sync Header Error)** Enables DataK masking of special symbols or the breaking of the sync header. For more details, see the EINJ3_SYMBOL_REG register. Note: This register field is sticky. 0_B **DISABLE**: Disable 1_B **ENABLE**: Enable
ERROR_INJECTION4_ENABLE	4	rwh	**Error Injection4 Enable (FC Credit Update Error)** Enables insertion of errors into UpdateFCs. For more details, see the EINJ4_FC_REG register. Note: This register field is sticky. 0_B **DISABLE**: Disable 1_B **ENABLE**: Enable
ERROR_INJECTION5_ENABLE	5	rwh	**Error Injection5 Enable (TLP Duplicate/Nullify Error)** Enables insertion of duplicate/nullified TLPs. For more details, see the EINJ5_SP_TLP_REG register. Note: This register field is sticky. 0_B **DISABLE**: Disable 1_B **ENABLE**: Enable

(table continues...)

(continued)

Field	Bits	Type	Description
ERROR_INJECTION6_ENABLE	6	rwh	**Error Injection6 Enable (Specific TLP Error)** Enables insertion of errors into the packets that you select. For more details, see the EINJ6_COMPARE_*_REG/EINJ6_CHANGE_*_REG/ EINJ6_TLP_REG registers. Note: This register field is sticky. 0_B **DISABLE**: Disable 1_B **ENABLE**: Enable
RSVDP_7	31:7	r	**Reserved for future use**

19.4.2.167 Error Injection Control 0 (CRC Error)

Controls the insertion of errors into the CRC, and parity of ordered sets for the selected type of the packets as follows: - LCRC. Bad TLP will be detected at the receiver side; receiver responds with NAK DLLP; Data Link Retry starts. - 16-bit CRC of ACK/NAK DLLPs. Bad DLLP occurs at the receiver side; Replay NUM Rollover occurs. - 16-bit CRC of UpdateFC DLLPs. Error insertion continues for the specific time; LTSSM transitions to the Recovery state because of the UpdateFC timeout (if the timeout is implemented at the receiver of the UpdateFCs). - ECRC. If ECRC check is enabled, ECRC error is detected at the receiver side. - FCRC. Framing error will be detected, TLP is discarded, and the LTSSM transitions to Recovery state. - Parity of TSOS. Error insertion continues for the specific time; LTSSM Recovery/Configuration timeout will occur. - Parity of SKPOS. Lane error will be detected at the receiver side.

EINJ0_CRC_REG Offset address: $00041B4_H$

Error Injection Control 0 [CRC Error) Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_12															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVDP_12				EINJ0_CRC_TYPE				EINJ0_COUNT							
r				rw				rwh							

Field	Bits	Type	Description
EINJ0_COUNT	7:0	rwh	**Error injection count** Indicates the number of errors. This register is decremented when the errors have been inserted. - If the counter value is 0x01 and error is inserted, ERROR_INJECTION0_ENABLE in EINJ_ENABLE_REG returns 0b. - If the counter value is 0x00 and ERROR_INJECTION0_ENABLE=1, the errors are inserted until ERROR_INJECTION0_ENABLE is set to '0'. Note: This register field is sticky.

(table continues...)

(continued)

Field	Bits	Type	Description
EINJ0_CRC_TYPE	11:8	rw	**Error injection type** Selects the type of CRC error to be inserted. All encodings other than the defined encodings are reserved. Note: This register field is sticky. 0_H **TLP_LCRC_ERR**: (TX Path) New TLP's LCRC error injection 1_H **_16b_CRC_ERR_ACK_NAK_DLLP**: (TX Path) 16bCRC error injection of ACK/NAK DLLP 2_H **_16b_CRC_ERR_UPD_FC**: (TX Path) 16bCRC error injection of Update-FC DLLP 3_H **TLP_ECRC_ERR**: (TX Path) New TLP's ECRC error injection 4_H **FCRC_ERR_TLP**: (TX Path) TLP's FCRC error injection (128b/130b) 5_H **PARITY_TSOS_ERR**: (TX Path) Parity error of TSOS (128b/130b) 6_H **PARITY_SKPOS_ERR**: (TX Path) Parity error of SKPOS (128b/130b) 8_H **LCRC_ERR**: (RX Path) LCRC error injection B_H **ECRC_ERR**: (RX Path) ECRC error injection **others,** Reserved
RSVDP_12	31:12	r	**Reserved for future use**

19.4.2.168 Error Injection Control 1 (Sequence Number Error).

Controls the sequence number of the specific TLPs and ACK/NAK DLLPs. Data Link Protocol Error will be detected at the Rx side of ACK/NAL DLLPs when one of these conditions is true: - ((NEXT_TRANSMIT_SEQ -1) - AckNak_Seq_Num) mod 4096 > 2048 - (AckNak_Seq_Num - ACKD_SEQ) mod 4096 >= 2048 TLP is treated as Duplicate TLP at the Rx side when all these conditions are true: - Sequence Number != NEXT_RCV_SEQ - (NEXT_RCV_SEQ - Sequence Number) mod 4096 <= 2048 TLP is treated as Bad TLP at the Rx side when all these conditions are true: - Sequence Number != NEXT_RCV_SEQ and - (NEXT_RCV_SEQ - Sequence Number) mod 4096 > 2048

EINJ1_SEQNUM_REG Offset address: $00041B8_H$

Error Injection Control 1 [Sequence Number Error).
 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_29			EINJ1_BAD_SEQNUM												
r			rw												

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVDP_9							EINJ1_SEQNUM_TYPE	EINJ1_COUNT							
r							rw	rwh							

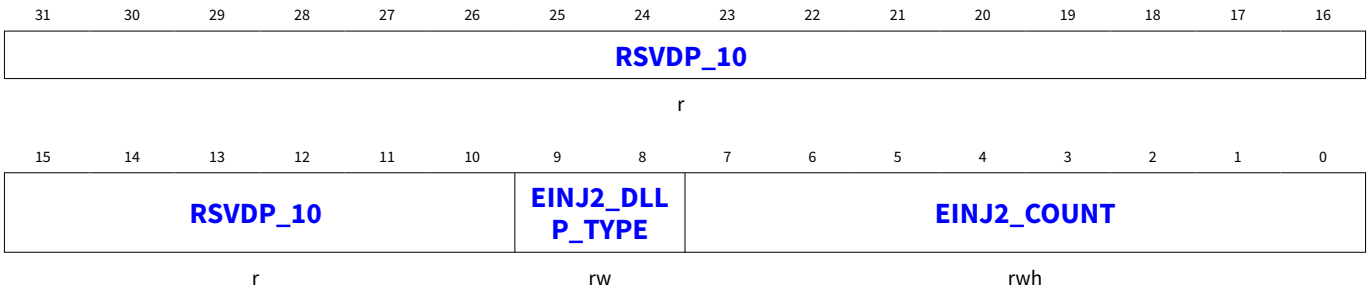
Field	Bits	Type	Description
EINJ1_COUNT	7:0	rwh	**Error injection count** Indicates the number of errors. This register is decremented as the errors are being inserted. - If the counter value is 0x01 and error is inserted, ERROR_INJECTION1_ENABLE in EINJ_ENABLE_REG returns '0'. - If the counter value is 0x00 and ERROR_INJECTION1_ENABLE=1, the errors are inserted until ERROR_INJECTION1_ENABLE is set to '0'. Note: This register field is sticky.
EINJ1_SEQNUM_TYPE	8	rw	**Sequence number type** Selects the type of sequence number. Note: This register field is sticky. 0_B **TLP_ERR**: Insertion of New TLP's SEQ# error 1_B **ACK_NAK_DLLP_ERR**: Insertion of ACK/NAK DLLP's SEQ# Error
RSVDP_9	15:9	r	**Reserved for future use**
EINJ1_BAD_SEQNUM	28:16	rw	**Bad sequence number** Indicates the value to add/subtract from the naturally-assigned sequence numbers. This value is represented by two's complement. For example: - Set Type, SEQ# and Count -- EINJ1_SEQNUM_TYPE =0 (Insert errors into new TLPs) -- EINJ1_BAD_SEQNUM =0x1FFD (represents -3) -- EINJ1_COUNT =1 - Enable Error Injection -- ERROR_INJECTION1_ENABLE =1 - Send a TLP From the Core's Application Interface -- Assume SEQ#5 is given to the TLP. - The SEQ# is Changed to #2 by the Error Injection Function in Layer2. -- 5 + (-3) = 2 - The TLP with SEQ#2 is Transmitted to PCIe Link. Note: This register field is sticky. $0FFF_H$ **MAX_VAL**: +4095 1001_H **MIN_VAL**: -4095 **others,** Reserved
RSVDP_29	31:29	r	**Reserved for future use**

19.4.2.169 Error Injection Control 2 (DLLP Error)

Controls the transmission of DLLPs and inserts the following errors: - If "ACK/NAK DLLP's transmission block" is selected, replay timeout error will occur at the transmitter of the TLPs and then Data Link Retry will occur. - If "Update FC DLLP's transmission block" is selected, LTSSM will transition to the Recovery state because of the UpdateFC timeout (if the timeout is implemented at the receiver of the UpdateFCs). - If "Always Transmission for NAK DLLP" is selected, Data Link Retry will occur at the transmitter of the TLPs. Furthermore, Replay NUM Rollover will occur when the transmitter has been requested four times to send the TLP with the same sequence number.

EINJ2_DLLP_REG Offset address: $00041BC_H$

Error Injection Control 2 [DLLP Error) Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
EINJ2_COUNT	7:0	rwh	**Error injection count** Indicates the number of errors. This register is decremented as the errors are being inserted. - If the counter value is 0x01 and the error is inserted, ERROR_INJECTION2_ENABLE in EINJ_ENABLE_REG returns '0'. - If the counter value is 0x00 and ERROR_INJECTION2_ENABLE =1, the errors are inserted until ERROR_INJECTION2_ENABLE is set to '0'. This register is affected only when EINJ2_DLLP_TYPE =2'10b. Note: This register field is sticky.
EINJ2_DLLP_TYPE	9:8	rw	**DLLP Type** Selects the type of DLLP errors to be inserted. Note: This register field is sticky. 00_B **ACK_NACK_DLLP**: ACK/NAK DLLP's transmission block 01_B **UPD_FC_DLLP**: Update FC DLLP's transmission block 10_B **NAK_DLLP**: Always Transmission for NAK DLLP 11_B **RSVD**: Reserved
RSVDP_10	31:10	r	**Reserved for future use**

19.4.2.170 Error Injection Control 3 (Symbol Error)

When 8b/10b encoding is used, this register controls error insertion into the special (K code) symbols. - If TS1/TS2/FTS/E-Idle/SKP is selected, it affects whole of the ordered set. It might cause timeout of the LTSSM. - If END/EDB/STP/SDP is selected, TLP/DLLP will be discarded at the receiver side. When 128b/130b encoding is used, this register controls error insertion into the sync-header.

EINJ3_SYMBOL_REG Offset address: $00041C0_H$

Error Injection Control 3 [Symbol Error) Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_11															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVDP_11					EINJ3_SYMBOL_TYPE			EINJ3_COUNT							
r					rw			rwh							

Field	Bits	Type	Description
EINJ3_COUNT	7:0	rwh	**Error injection count** Indicates the number of errors. This register is decremented as the errors are being inserted. - If the counter value is 0x01 and error is inserted, ERROR_INJECTION3_ENABLE in EINJ_ENABLE_REG returns '0'. - If the counter value is 0x00 and ERROR_INJECTION3_ENABLE =1, the errors are inserted until ERROR_INJECTION3_ENABLE is set to '0'. Note: This register field is sticky.

(table continues...)

(continued)

Field	Bits	Type	Description
EINJ3_SYMBOL_TYPE	10:8	rw	**Error Type** 8b/10b encoding - Mask K symbol. All encodings other than the defined encodings are reserved. Note: This register field is sticky. 000_B **RSVD_OR_INVRT_SYNC_HDR**: Invert sync header for 128b/130b encoding or this field is reserved for 8b/10b encoding 001_B **COM_PAD_TS1**: COM/PAD(TS1 Order set) 010_B **COM_PAD_TS2**: COM/PAD(TS2 Order set) 011_B **COM_FTS**: COM/FTS(FTS Order set) 100_B **COM_IDL**: COM/IDL(E-Idle Order set) 101_B **END_EDB**: END/EDB Symbol 110_B **STP_SDP**: STP/SDP Symbol 111_B **COM_SKP**: COM/SKP(SKP Order set)
RSVDP_11	31:11	r	**Reserved for future use**

19.4.2.171 Error Injection Control 4 (FC Credit Error)

Controls error insertion into the credit value in the UpdateFCs. It is possible to insert errors for any of the following types: - Posted TLP Header credit - Non-Posted TLP Header credit - Completion TLP Header credit - Posted TLP Data credit - Non-Posted TLP Data credit - Completion TLP Data credit These errors are not correctable while error insertion is enabled. Receiver buffer overflow error might occur.

EINJ4_FC_REG Offset address: $00041C4_H$

Error Injection Control 4 [FC Credit Error) Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_29			EINJ4_BAD_UPDFC_VALUE												
r			rw												

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVDP_15	EINJ4_VC_NUMBER			RSVDP_11	EINJ4_UPDFC_TYPE			EINJ4_COUNT							
r	rw			r	rw			rwh							

Field	Bits	Type	Description
EINJ4_COUNT	7:0	rwh	**Error injection count** Indicates the number of errors. This register is decremented as the errors are being inserted. - If the counter value is 0x01 and error is inserted, ERROR_INJECTION4_ENABLE in EINJ_ENABLE_REG returns '0'. - If the counter value is 0x00 and ERROR_INJECTION4_ENABLE =1, the errors are inserted until ERROR_INJECTION4_ENABLE is set to '0'. Note: This register field is sticky.

(table continues...)

(continued)

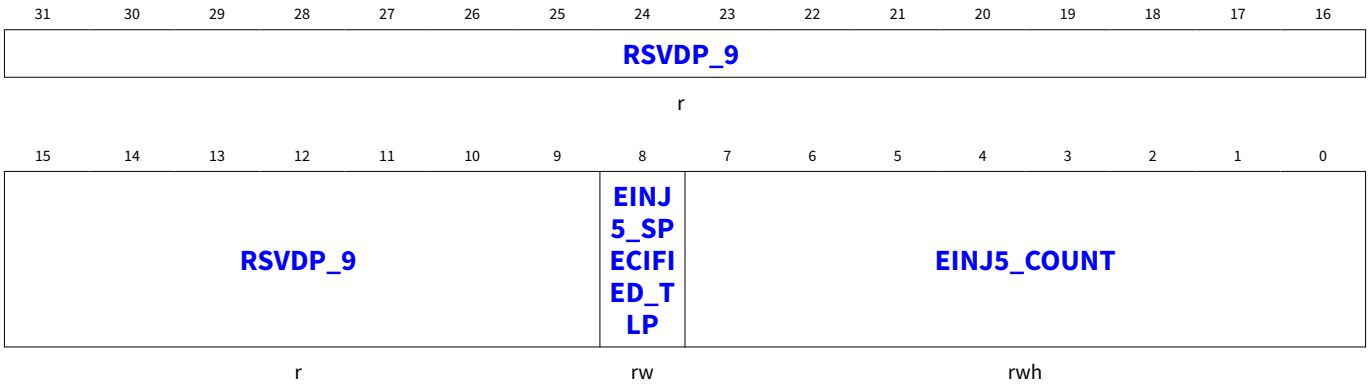
Field	Bits	Type	Description
EINJ4_UPDFC_TYPE	10:8	rw	**Update-FC type** Selects the credit type. Note: This register field is sticky. 000_B **POSTED_TLP_HDR**: Posted TLP Header Credit value control 001_B **NON_POSTED_TLP_HDR**: Non-Posted TLP Header Credit value control 010_B **CMPL_TLP_HDR**: Completion TLP Header Credit value control 011_B **RSERVED**: Reserved 100_B **POSTED_TLP_DATA**: Posted TLP Data Credit value control 101_B **NON_POSTED_TLP_DATA**: Non-Posted TLP Data Credit value control 110_B **CMPL_TLP_DATA**: Completion TLP Data Credit value control 111_B **RSVD**: Reserved
RSVDP_11	11	r	**Reserved for future use**
EINJ4_VC_NUMBER	14:12	rw	**VC Number** Indicates target VC Number. Note: This register field is sticky. 000_B **MIN_VAL**: Min Value 111_B **MAX_VAL**: Min Value
RSVDP_15	15	r	**Reserved for future use**
EINJ4_BAD_UPDFC_VALUE	28:16	rw	**Bad update-FC credit value** Indicates the value to add/subtract from the UpdateFC credit. This value is represented by two's complement. Note: This register field is sticky. $0FFF_H$ **MAX_VAL**: +4095 1001_H **MIN_VAL**: -4095 **others,** Reserved
RSVDP_29	31:29	r	**Reserved for future use**

19.4.2.172 Error Injection Control 5 (Specific TLP Error)

Controls the generation of specified TLPs. Correctable errors will occur which will be fixed by the PCIe protocol. - For Duplicate TLP, the controller initiates Data Link Retry by handling ACK DLLP as NAK DLLP. These TLPs will be duplicate TLPs at the receiver side. - For Nullified TLP, the TLPs that the controller transmits are changed into nullified TLPs and the original TLPs are stored in the retry buffer. The receiver of these TLPs will detect the lack of seq# and send NAK DLLP at the next TLP. Then the original TLPs are sent from retry buffer and the data controls are recovered. For 128 bit controller or more than 128 bit, the controller inserts errors the number of times of EINJ5_COUNT but doesn't ensure that the errors are continuously inserted into TLPs.

EINJ5_SP_TLP_REG Offset address: $00041C8_H$

Error Injection Control 5 [Specific TLP Error) Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
EINJ5_COUNT	7:0	rwh	**Error injection count** Indicates the number of errors. This register is decremented as the errors are being inserted. - If the counter value is 0x01 and error is inserted, ERROR_INJECTION5_ENABLE in EINJ_ENABLE_REG returns '0'. - If the counter value is 0x00 and ERROR_INJECTION5_ENABLE =1, the errors are inserted until ERROR_INJECTION5_ENABLE is set to '0'. Note: This register field is sticky.
EINJ5_SPECIFIED_TLP	8	rw	**Specified TLP** Selects the specified TLP to be inserted. Note: This register field is sticky. 0_B **DUPLICATE_DLLP**: Generates duplicate TLPs by handling ACK DLLP as NAK DLLP 1_B **NULLIFIED_TLP**: Generates Nullified TLP (Original TLP will be stored in retry buffer)
RSVDP_9	31:9	r	**Reserved for future use**

19.4.2.173 Error Injection Control 6 (Compare Point Header DWORD #0) i

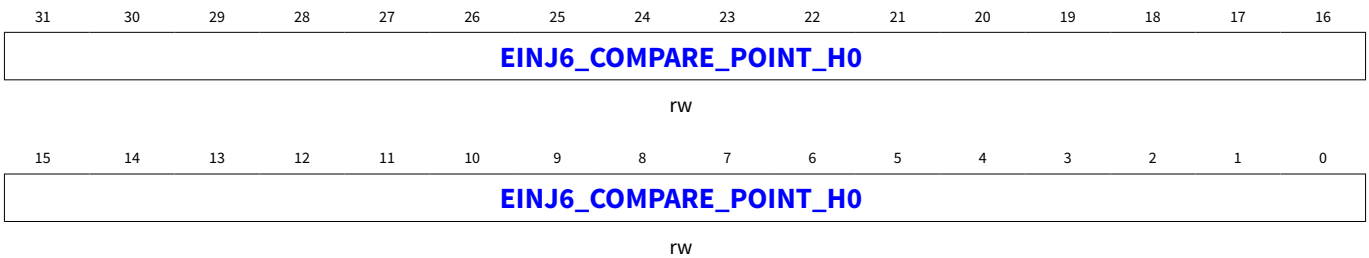
Program this register for the 1st DWORD of TLP header/prefix. It is necessary to carefully consider the endianness when you program this register. Bits [31:0] = TLP_DW0[7:0], TLP_DW0[15:8], TLP_DW0[23:16], TLP_DW0[31:24] The Packet Compare Point registers (EINJ6_COMPARE_POINT*) specify which Tx TLP header bits to compare with the corresponding bits in the Packet Compare Value registers (EINJ6_COMPARE_VALUE*). When all specified bits (in the Tx TLP header and EINJ6_COMPARE_VALUE*) match, the controller inserts errors into the TLP. The type and number of errors are specified by the EINJ6_TLP_REG register.

EINJ6_COMPARE_POINT_Hi_REG (i=0-3)

Error Injection Control 6 [Compare Point Header DWORD #0) i

Offset address: $00041CC_H+i*4$

Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
EINJ6_COMPARE_POINT_H0	31:0	rw	**Packet Compare Point: 1st DWORD** Specifies which Tx TLP header DWORD#0 bits to compare with the corresponding bits in the Packet Compare Value registers (EINJ6_COMPARE_VALUE*). When all specified bits (in the Tx TLP header and EINJ6_COMPARE_VALUE*) match, the controller inserts errors into the TLP. Note: This register field is sticky.

19.4.2.174 Error Injection Control 6 (Compare Value Header DWORD #0) i

Program this register for the 1st DWORD of TLP header/prefix. It is necessary to carefully consider the endianness when you program this register. Bits [31:0] = TLP_DW0[7:0], TLP_DW0[15:8], TLP_DW0[23:16], TLP_DW0[31:24] The Packet Compare Point registers (EINJ6_COMPARE_POINT*) specify which Tx TLP header bits to compare with the corresponding bits in the Packet Compare Value registers (EINJ6_COMPARE_VALUE*). When all specified bits (in the Tx TLP header and EINJ6_COMPARE_VALUE*) match, the controller inserts errors into the TLP. The type and number of errors are specified by the EINJ6_TLP_REG register.

EINJ6_COMPARE_VALUE_Hi_REG (i=0-3) Offset address: $00041DC_H+i*4$

Error Injection Control 6 [Compare Value Header DWORD #0) i — Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EINJ6_COMPARE_VALUE_H0															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EINJ6_COMPARE_VALUE_H0															
rw															

Field	Bits	Type	Description
EINJ6_COMPARE_VALUE_H0	31:0	rw	**Packet Compare Value: 1st DWORD** Specifies the value to compare against Tx the TLP header DWORD#0 bits specified in the Packet Compare Point registers (EINJ6_COMPARE_POINT*). Note: This register field is sticky.

19.4.2.175 Error Injection Control 6 (Change Point Header DWORD #0) i

Program this register for the 1st DWORD of TLP header/prefix. It is necessary to carefully consider the endianness when you program this register. Bits [31:0] = TLP_DW0[7:0], TLP_DW0[15:8], TLP_DW0[23:16], TLP_DW0[31:24] The Packet Change Point registers (EINJ6_CHANGE_POINT*) specify which Tx TLP header bits to replace with the corresponding bits in the Packet Change Value registers (EINJ6_CHANGE_VALUE*). The type and number of errors are specified by the EINJ6_TLP_REG register. Only applies when EINJ6_INVERTED_CONTROL in EINJ6_TLP_REG =0.

EINJ6_CHANGE_POINT_Hi_REG (i=0-3) Offset address: $00041EC_H+i*4$

Error Injection Control 6 [Change Point Header DWORD #0) i — Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EINJ6_CHANGE_POINT_H0															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EINJ6_CHANGE_POINT_H0															
rw															

Field	Bits	Type	Description
EINJ6_CHANGE_POINT_H0	31:0	rw	**Packet Change Point: 1st DWORD** Specifies which Tx TLP header DWORD#0 bits to replace with the corresponding bits in the Packet Change Value registers (EINJ6_CHANGE_VALUE*). Note: This register field is sticky.

19.4.2.176 Error Injection Control 6 (Change Value Header DWORD #0) i

Program this register for the 1st DWORD of TLP header/prefix. It is necessary to carefully consider the endianness when you program this register. Bits [31:0] = TLP_DW0[7:0], TLP_DW0[15:8], TLP_DW0[23:16], TLP_DW0[31:24] The Packet Change Point registers (EINJ6_CHANGE_POINT*) specify which Tx TLP header bits to replace with the corresponding bits in the Packet Change Value registers (EINJ6_CHANGE_VALUE*). The type and number of errors are specified by the EINJ6_TLP_REG register. Only applies when EINJ6_INVERTED_CONTROL in EINJ6_TLP_REG =0.

EINJ6_CHANGE_VALUE_Hi_REG (i=0-3)

Error Injection Control 6 [Change Value Header DWORD #0) i

Offset address: $00041FC_H+i*4$

Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EINJ6_CHANGE_VALUE_H0															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EINJ6_CHANGE_VALUE_H0															
rw															

Field	Bits	Type	Description
EINJ6_CHANGE_VALUE_H0	31:0	rw	**Packet Change Value: 1st DWORD** Specifies replacement values for the Tx TLP header DWORD#0 bits defined in the Packet Change Point registers (EINJ6_CHANGE_POINT*). Only applies when the EINJ6_INVERTED_CONTROL field in EINJ6_TLP_REG is '0'. Note: This register field is sticky.

19.4.2.177 Error Injection Control 6 (Packet Error)

The Packet Compare Point registers (EINJ6_COMPARE_POINT*) specify which Tx TLP header bits to compare with the corresponding bits in the Packet Compare Value registers (EINJ6_COMPARE_VALUE*). When all specified bits (in the Tx TLP header and EINJ6_COMPARE_VALUE*) match, the controller inserts errors into the TLP. The type and number of errors are specified by the this register. The Packet Change Point registers (EINJ6_CHANGE_POINT*) specify which Tx TLP header bits to replace with the corresponding bits in the Packet Change Value registers (EINJ6_CHANGE_VALUE*). The type and number of errors are specified by the this register. Only applies when EINJ6_INVERTED_CONTROL in this register =0. The TLP into that errors are injected will not arrive at the transaction layer of the remote device when all of the following conditions are true. - Using 128b/130b encoding - Injecting errors into TLP Length field / TLP digest bit

EINJ6_TLP_REG Offset address: $000420C_H$

Error Injection Control 6 [Packet Error) Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_12															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVDP_12				EINJ6_PACKET_TYPE			EINJ6_INVERTED_CONTROL	EINJ6_COUNT							
r				rw			rw	rwh							

Field	Bits	Type	Description
EINJ6_COUNT	7:0	rwh	**Error Injection Count** Indicates the number of errors to insert. This counter is decremented while errors are been inserted. • If the counter value is 0x01 and error is inserted, ERROR_INJECTION6_ENABLE in EINJ_ENABLE_REG returns '0' • If the counter value is 0x00 and ERROR_INJECTION6_ENABLE=1, errors are inserted until ERROR_INJECTION6_ENABLE is set to '0' Note: This register field is sticky.
EINJ6_INVERTED_CONTROL	8	rw	**Inverted Error Injection Control** Encodded vlues given as above. Note: This register field is sticky. 0_B **REPLACE**: EINJ6_CHANGE_VALUE_H[0/1/2/3] is used to replace bits specified by EINJ6_CHANGE_POINT_H[0/1/2/3] 1_B **IGNORE**: EINJ6_CHANGE_VALUE_H[0/1/2/3] is ignored and inverts bits specified by EINJ6_CHANGE_POINT_H[0/1/2/3]
EINJ6_PACKET_TYPE	11:9	rw	**Packet type** Selects the TLP packets to inject errors into. All encodings other than the specified encodings are reserved. Note: This register field is sticky. 000_B **HEADER**: TLP Header 001_B **TLP_FIRST_4_DW**: TLP Prefix 1st 4-DWORDs 010_B **TLP_SECOND_DW**: TLP Prefix 2nd -DWORDs **others,** Reserved

(table continues...)

(continued)

Field	Bits	Type	Description
RSVDP_12	31:12	r	**Reserved for future use**

19.4.2.178 Silicon Debug Control 1

SD_CONTROL1_REG Offset address: 0004220_H

Silicon Debug Control 1 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_24								LOW_POWER_INTERVAL		TX_EIOS_NUM		RSVDP_17			FORCE_DETECT_LANE_EN
r								rw		rw		r			rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FORCE_DETECT_LANE															
rw															

Field	Bits	Type	Description
FORCE_DETECT_LANE	15:0	rw	**Force Detect Lane** When the FORCE_DETECT_LANE_EN field is set, the controller ignores receiver detection from PHY during LTSSM Detect state and uses this value instead. Value represents lane number. Note: This register field is sticky. 0000_H **MIN_VAL**: Lane0 $000F_H$ **MAX_VAL**: Lane15 **others,** Reserved
FORCE_DETECT_LANE_EN	16	rw	**Force Detect Lane Enable** Note: This register field is sticky. 0_B **CLEAR**: Clear 1_B **SET**: When this bit is set, the controller ignores receiver detection from PHY during LTSSM Detect state and uses FORCE_DETECT_LANE
RSVDP_17	19:17	r	**Reserved for future use**

(table continues...)

(continued)

Field	Bits	Type	Description
TX_EIOS_NUM	21:20	rw	**Number of Tx EIOS** This register sets the number of transmit EIOS for L0s/L1 entry and Disable/Loopback/Hot-reset exit. The controller selects the greater value between this register and the value defined by the PCI-SIG specification. Note: This register field is sticky. 00_B **G2_2_EIOS_OTHER_SPEED_1_EIOS**: (2.5GT/s, 8.0GT/s or higher) 1 or (5.0GT/s) 2 01_B **G2_8_EIOS_OTHER_SPEED_4_EIOS**: (2.5GT/s, 8.0GT/s or higher) 4 or (5.0GT/s) 8 10_B **G2_16_EIOS_OTHER_SPEED_8_EIOS**: (2.5GT/s, 8.0GT/s or higher) 8 or (5.0GT/s) 16 11_B **G2_32_EIOS_OTHER_SPEED_16_EIOS**: (2.5GT/s, 8.0GT/s or higher) 16 or (5.0GT/s) 32
LOW_POWER_INTERVAL	23:22	rw	**Low Power Entry Interval Time** Interval Time that the controller starts monitoring RXELECIDLE signal after L0s/L1/L2 entry. You should set the value according to the latency from receiving EIOS to, RXELECIDLE assertion at the PHY. Note: This register field is sticky. 00_B **_40NS**: 40ns 01_B **_160NS**: 160ns 10_B **_320NS**: 320ns 11_B **_640NS**: 640ns
RSVDP_24	31:24	r	**Reserved for future use**

19.4.2.179 Silicon Debug Control 2

SD_CONTROL2_REG Offset address: 0004224_H

Silicon Debug Control 2 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_17															FRAMING_ERR_RECOVERY_DISABLE
r															rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVDP_11					DIRECT_LPBKSLV_TO_EXIT	DIRECT_POLCOMP_TO_DETECT	DIRECT_RECIDLE_TO_CONFIG	RSVDP_3					NOACK_FORCE_LINKDOWN	RECOVERY_REQUEST	HOLD_LTSSM
r					rw	rw	rw	r					rw	rwh	rw

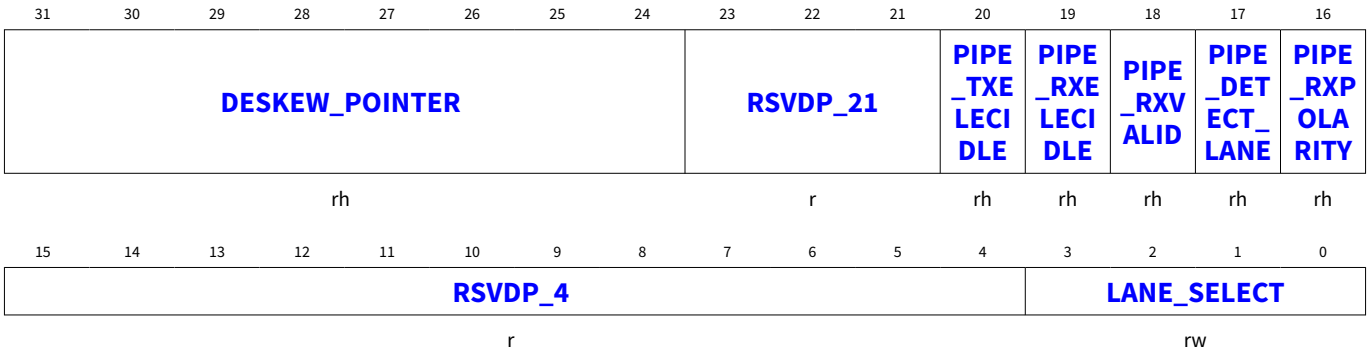
Field	Bits	Type	Description
HOLD_LTSSM	0	rw	**Hold and Release LTSSM** Note: This register field is sticky. 0_B **CLEAR**: Clear 1_B **SET**: As long as this field is '1', the controller stays in the current LTSSM
RECOVERY_REQUEST	1	rwh	**Recovery Request. Reading register field always returns a '0'.** 0_B **CLEAR**: Clear 1_B **SET**: When this bit is set to '1' in L0 or L0s, the LTSSM starts transitioning to Recovery State This request does not cause a speed change or re-equalization.
NOACK_FORCE_LINKDOWN	2	rw	**Force LinkDown** Note: This register field is sticky. 0_B **CLEAR**: Clear 1_B **SET**: When this bit is set and the controller detects REPLY_NUM rolling over 4 times, the LTSSM transitions to Detect State
RSVDP_3	7:3	r	**Reserved for future use**
DIRECT_RECIDLE_TO_CONFIG	8	rw	**Direct Recovery.Idle to Configuration** Note: This register field is sticky. 0_B **CLEAR**: Clear 1_B **SET**: When this bit is set and the LTSSM is in Recovery Idle State, the LTSSM transitions to Configuration state
DIRECT_POLCOMP_TO_DETECT	9	rw	**Direct Polling.Compliance to Detect** Note: This register field is sticky. 0_B **CLEAR**: Clear 1_B **SET**: When this bit is set and the LTSSM is in Polling Compliance State, the LTSSM transitions to Detect state
DIRECT_LPBKSLV_TO_EXIT	10	rw	**Direct Loopback Slave To Exit** Note: This register field is sticky. 0_B **CLEAR**: Clear 1_B **SET**: When this bit is set and the LTSSM is in Loopback Slave Active State, the LTSSM transitions to Loopback Slave Exit state
RSVDP_11	15:11	r	**Reserved for future use**
FRAMING_ERR_RECOVERY_DISABLE	16	rw	**Framing Error Recovery Disable** This bit disables a transition to Recovery state when a Framing Error is occurred. Note: This register field is sticky.
RSVDP_17	31:17	r	**Reserved for future use**

19.4.2.180 Silicon Debug Status(Layer1 Per-lane)

This viewport register returns the data selected by the following field: - LANE_SELECT in SD_STATUS_L1LANE_REG

SD_STATUS_L1LANE_REG Offset address: 0004230_H

Silicon Debug Status[Layer1 Per-lane) Kernel Reset value: $0018\ 0000_H$



Field	Bits	Type	Description
LANE_SELECT	3:0	rw	**Lane Select** Lane Select register for Silicon Debug Status Register of Layer1-PerLane. Note: This register field is sticky. 0_H **MIN_VAL**: Lane0 F_H **MAX_VAL**: Lane15
RSVDP_4	15:4	r	**Reserved for future use**
PIPE_RXPOLARITY	16	rh	**PIPE:RxPolarity** Indicates PIPE RXPOLARITY signal of selected lane number(LANE_SELECT). Note: This register field is sticky.
PIPE_DETECT_LANE	17	rh	**PIPE:Detect Lane** Indicates whether PHY indicates receiver detection or not on selected lane number(LANE_SELECT). Note: This register field is sticky.
PIPE_RXVALID	18	rh	**PIPE:RxValid** Indicates PIPE RXVALID signal of selected lane number(LANE_SELECT). Note: This register field is sticky.
PIPE_RXELECIDLE	19	rh	**PIPE:RxElecIdle** Indicates PIPE RXELECIDLE signal of selected lane number(LANE_SELECT). Note: This register field is sticky.
PIPE_TXELECIDLE	20	rh	**PIPE:TxElecIdle** Indicates PIPE TXELECIDLE signal of selected lane number(LANE_SELECT). Note: This register field is sticky.
RSVDP_21	23:21	r	**Reserved for future use**
DESKEW_POINTER	31:24	rh	**Deskew Pointer** Indicates Deskew pointer of internal Deskew buffer of selected lane number(LANE_SELECT). Note: This register field is sticky.

19.4.2.181 Silicon Debug Status(Layer1 LTSSM)

SD_STATUS_L1LTSSM_REG Offset address: 0004234_H

Silicon Debug Status[Layer1 LTSSM) Kernel Reset value: $0000\ 0200_H$

19 Peripheral Component Interconnect Express (PCIe)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LTSSM_VARIABLE															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LANE_REVERSAL	RSVDP_11				PIPE_POWER_DOWN			FRAMING_ERR	FRAMING_ERR_PTR						
rh	r				rh			rw1ch	rh						

Field	Bits	Type	Description
FRAMING_ERR_PTR	6:0	rh	**First Framing Error Pointer** Identifies the first Framing Error using the following encoding. The field contents are only valid value when FRAMING_ERR =1. • Received Unexpected Framing Token (Values 01h to 06h) - Received Unexpected STP Token (Values 11h to 13h) • Received Unexpected Block (Values 21h to 2Eh) All encodings other than the defined encodings are reserved. Note: This register field is sticky. 01_H **STP_OR_SDP_OR_IDL_RECEIVED_NO_TLP_OR_DLLP**: When non- STP/SDP/IDL Token was received and it was not in TLP/DLLP reception 02_H **CURRENT_NO_VALID_EDB_PREVIOUS_VALID_EDB**: When current token was not a valid EDB token and previous token was an EDB (128/256 bit controller only) 03_H **SDP_RECEIVED_NOT_EXPECTED**: When SDP token was received but not expected (128 bit & (x8 | x16) controller only) 04_H **STP_RECEIVED_NOT_EXPECTED**: When STP token was received but not expected (128 bit & (x8 | x16) controller only) 05_H **EDS_RECEIVED_NOT_EXPECTED**: When EDS token was expected but not received or whenever an EDS token was received but not expected 06_H **FRE_ERR_DESKEW_PKT_IN_PROGRESS**: When a framing error was detected in the deskew block while a packet has been in progress in token_finder 11_H **CRC_STP_NOT_MATCH**: When Framing CRC in STP token did not match 12_H **PARITY_STP_NOT_MATCH**: When Framing Parity in STP token did not match 13_H **TLP_LENGTH_SMALLER_THEN_5DW**: When Framing TLP Length in STP token was smaller than 5 DWORDs 21_H **RECEVING_OS_AFTER_SDS_IN_DATA_STREAM**: When Receiving an OS Block following SDS in Datastream state 22_H **AFTER_DATA_BLK_OS_BLK_NOT_SKP_EI_EIE**: When Data Block followed by OS Block different from SKP, EI, EIE in Datastream state 23_H **UNDEFINE_BLK_TYPE**: When Block with an undefined Block Type in Datastream state 24_H **DATA_STREAM_WITHOUT_3_CYCLE_DATA_STREAM_S**: When Data Stream without data over three cycles in Datastream state 25_H **OS_BLK_DURING_DATA_STREAM**: When OS Block during Data Stream in Datastream state 26_H **RXSTATUS_ERR_DATA_STREAM_STATE**: When RxStatus Error was detected in Datastream state 27_H **NOT_ALL_LANE_START_RECEVING_SKP_OS_SAME_T**: When Not all active lanes receiving SKP OS starting at same cycle time in SKPOS state

(table continues...)

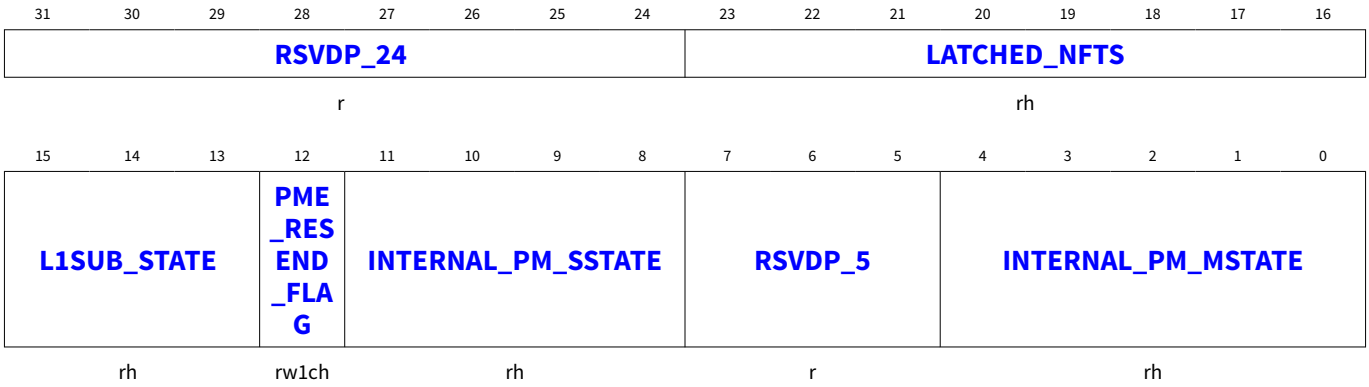
(continued)

Field	Bits	Type	Description
			28_H **_2_BLK_TIMEOUT_SKP_OS_SKPOS_STATE**: When a 2-Block timeout occurs for SKP OS in SKPOS state 29_H **CONSECUTIVE_OS_WITHIN_DATA_IN_SKPOS_STATE**: When Receiving consecutive OS Blocks within a Data Stream in SKPOS state $2A_H$ **PHYSTATUS_ERR_IN_SKPOS_STATE**: When Phy status error was detected in SKPOS state $2B_H$ **NOT_ALL_LANE_START_RECEVING_EIOS_SAME_T**: When Not all active lanes receiving EIOS starting at same cycle time in EIOS state $2C_H$ **AT_LEAST_1_SYM_IS_NOT_EIOS_FROM_4_SYM**: When At least one Symbol from the first 4 Symbols is not EIOS Symbol in EIOS state (CX_NB=2 only) $2D_H$ **NOT_ALL_LANE_START_RECEVING_EIEOS_SAME_T**: When Not all active lanes receiving EIEOS starting at same cycle time in EIEOS state $2E_H$ **NOT_16_EIEOS_SYM_RECEIVED**: When Not full 16 eieos symbols are received in EIEOS state **others,** Reserved
FRAMING_ERR	7	rw1ch	**Framing Error** Indicates Framing Error detection status. Note: This register field is sticky. Software write with 1 clears this bit-field.
PIPE_POWER_DOWN	10:8	rh	**PIPE:PowerDown** Indicates PIPE PowerDown signal. Note: This register field is sticky.
RSVDP_11	14:11	r	**Reserved for future use**
LANE_REVERSAL	15	rh	**Lane Reversal Operation** Receiver detected lane reversal. This field is only valid in the L0 LTSSM state. Note: This register field is sticky.
LTSSM_VARIABLE	31:16	rh	**LTSSM Variable** Indicates internal LTSSM variables defined in the PCI Express Base Specification. . Note: This register field is sticky. 0000_H **DIR_SPEED_CHANGE**: directed_speed_change 0001_H **CHANGED_SPEED_RCVRY**: changed_speed_recovery 0002_H **SUCCESSFUL_SPEED_NEGO**: successful_speed_negotiation 0003_H **UPCFG_CAPABLE**: upconfigure_capable; Set to '1' if both ports advertised the UpConfigure capability in the last Config.Complete 0004_H **SEL_DE_EMPHASIS**: select_deemphasis 0005_H **START_EQ_W_PRESET**: start_equalization_w_preset 0006_H **EQ_DONE_8GT**: equalization_done_8GT_data_rate 0007_H **EQ_DONE_16GT**: equalization_done_16GT_data_rate **others,** Reserved

19.4.2.182 Silicon Debug Status(PM)

SD_STATUS_PM_REG Offset address: 0004238_H

Silicon Debug Status[PM] Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
INTERNAL_PM_MSTATE	4:0	rh	**Internal PM State(Master)** Indicates internal state machine of Power Management Master controller. Note: This register field is sticky. 00_H **IDLE**: IDLE 01_H **L0**: L0 02_H **L0S**: L0S 03_H **ENTER_L0S**: ENTER_L0S 04_H **EXIT_L0S**: L0S_EXIT 05_H **WAIT_PMCSR_CPL_SENT**: WAIT_PMCSR_CPL_SENT 08_H **L1**: L1 09_H **L1_BLOCK_TLP**: L1_BLOCK_TLP $0A_H$ **L1_WAIT_LAST_TLP_ACK**: L1_WAIT_LAST_TLP_ACK $0B_H$ **L1_WAIT_PMDLLP_ACK**: L1_WAIT_PMDLLP_ACK $0C_H$ **L1_LINK_ENTR_L1**: L1_LINK_ENTR_L1 $0D_H$ **L1_EXIT**: L1_EXIT $0F_H$ **PREP_4L1**: PREP_4L1 10_H **L23_BLOCK_TLP**: L23_BLOCK_TLP 11_H **L23_WAIT_LAST_TLP_ACK**: L23_WAIT_LAST_TLP_ACK 12_H **L23_WAIT_PMDLLP_ACK**: L23_WAIT_PMDLLP_ACK 13_H **L23_ENTR_L23**: L23_ENTR_L23 14_H **L23RDY**: L23RDY 15_H **PREP_4L23**: PREP_4L23 16_H **L23RDY_WAIT4ALIVE**: L23RDY_WAIT4ALIVE 17_H **L0S_BLOCK_TLP**: L0S_BLOCK_TLP 18_H **WAIT_LAST_PMDLLP**: WAIT_LAST_PMDLLP 19_H **WAIT_DSTATE_UPDATE**: WAIT_DSTATE_UPDATE **others,** Reserved
RSVDP_5	7:5	r	**Reserved for future use**

(table continues...)

(continued)

Field	Bits	Type	Description
INTERNAL_PM_SSTATE	11:8	rh	**Internal PM State(Slave)** Indicates internal state machine of Power Management Slave controller. Note: This register field is sticky. 0_H **S_IDLE**: S_IDLE 1_H **S_RESPOND_NAK**: S_RESPOND_NAK 2_H **S_BLOCK_TLP**: S_BLOCK_TLP 3_H **S_WAIT_LAST_TLP_ACK**: S_WAIT_LAST_TLP_ACK 4_H **S_WAIT_EIDLE**: S_WAIT_EIDLE 5_H **S_LINK_ENTR_L1**: S_LINK_ENTR_L1 6_H **S_L1**: S_L1 7_H **S_L1_EXIT**: S_L1_EXIT 8_H **S_L23RDY**: S_L23RDY 9_H **S_LINK_ENTR_L23**: S_LINK_ENTR_L23 A_H **S_L23RDY_WAIT4ALIVE**: S_L23RDY_WAIT4ALIVE B_H **S_ACK_WAIT4IDLE**: S_ACK_WAIT4IDLE C_H **S_WAIT_LAST_PMDLLP**: S_WAIT_LAST_PMDLLP **others,** Reserved
PME_RESEND_FLAG	12	rw1ch	**PME Re-send flag** When the DUT sends a PM_PME message TLP, the DUT sets PME_Status bit. If host software does not clear PME_Status bit for 100ms(+50%/-5%), the DUT resends the PM_PME Message. This bit indicates that a PM_PME was resent. Note: This register field is sticky. Software write with 1 clears this bit-field.
L1SUB_STATE	15:13	rh	**L1Sub State** Indicates internal state machine of L1Sub state. Note: This register field is sticky. 000_B **S_L1_U**: S_L1_U : idle state 001_B **S_L1_0**: S_L1_0 : wait for aux_clk_active 010_B **S_L1_0_WAIT4_ACK**: S_L1_0_WAIT4_ACK : wait for pclkack 011_B **S_L1_0_WAIT4_CLKREQ**: S_L1_0_WAIT4_CLKREQ : wait for clkreq 100_B **S_L1_N_ENTRY**: S_L1_N_ENTRY : check clkreq_in_n is de-asserted for t_power_off time (only for L1.2, reduces to one cycle for L1.1) 101_B **S_L1_N**: S_L1_N : L1 substate, turn off txcommonmode circuits (L1.2 only) and rx electrical idle detection circuits 110_B **S_L1_N_EXIT**: S_L1_N_EXIT : locally/remotely initiated exit, assert pclkreq, wait for pclkack 111_B **S_L1_N_ABORT**: S_L1_N_ABORT : wait for pclkack when aborting an attempt to enter L1_N
LATCHED_NFTS	23:16	rh	**Latched N_FTS** Indicates the value of N_FTS in the received TS Ordered Sets from the Link Partner. Note: This register field is sticky.
RSVDP_24	31:24	r	**Reserved for future use**

19.4.2.183 Silicon Debug Status(Layer2)

SD_STATUS_L2_REG Offset address: 000423C$_H$

Silicon Debug Status[Layer2) Kernel Reset value: 00FF F000$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_28				FC_INIT2	FC_INIT1	DLCMSM		RX_ACK_SEQ_NO							
r				rh	rh	rh		rh							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RX_ACK_SEQ_NO				TX_TLP_SEQ_NO											
rh				rh											

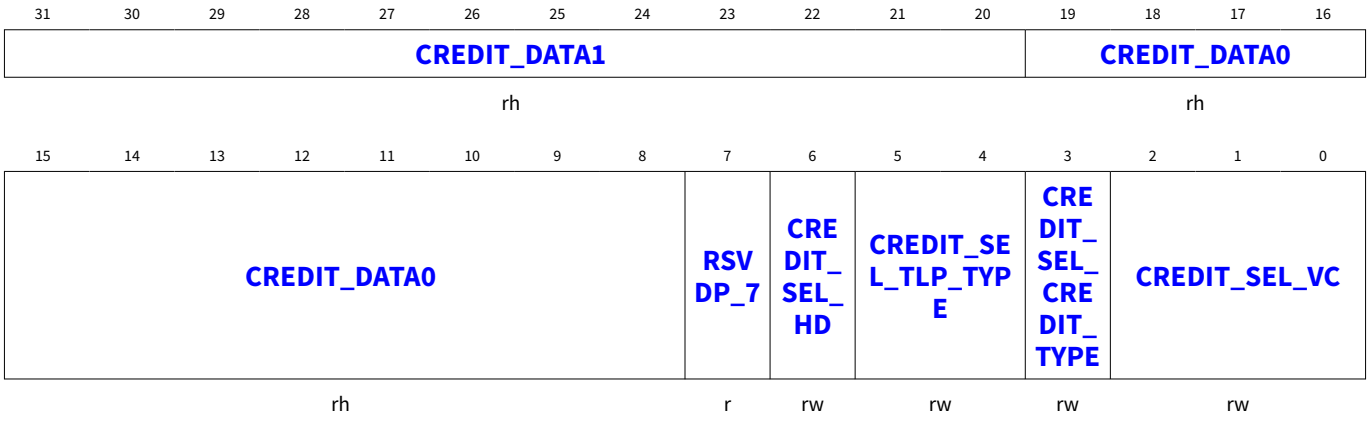
Field	Bits	Type	Description
TX_TLP_SEQ_NO	11:0	rh	**Tx Tlp Sequence Number** Indicates next transmit sequence number for transmit TLP. Note: This register field is sticky.
RX_ACK_SEQ_NO	23:12	rh	**Tx Ack Sequence Number** Indicates ACKD_SEQ which is updated by receiving ACK/NAK DLLP. Note: This register field is sticky.
DLCMSM	25:24	rh	**DLCMSM** Indicates the current DLCMSM. Note: This register field is sticky. 00_B **DL_INACTIVE**: DL_INACTIVE 01_B **DL_FC_INIT**: DL_FC_INIT 11_B **DL_ACTIVE**: DL_ACTIVE
FC_INIT1	26	rh	**FC_INIT1** Indicates the controller is in FC_INIT1(VC0) state. Note: This register field is sticky.
FC_INIT2	27	rh	**FC_INIT2** Indicates the controller is in FC_INIT2(VC0) state. Note: This register field is sticky.
RSVDP_28	31:28	r	**Reserved for future use**

19.4.2.184 Silicon Debug Status(Layer3 FC)

The CREDIT_DATA[0/1] fields in this viewport register return the data for the VC and TLP Type selected by the following fields: - CREDIT_SEL_VC - CREDIT_SEL_CREDIT_TYPE - CREDIT_SEL_TLP_TYPE - CREDIT_SEL_HD

SD_STATUS_L3FC_REG Offset address: 0004240$_H$

Silicon Debug Status[Layer3 FC) Kernel Reset value: 0000 0000$_H$



Field	Bits	Type	Description
CREDIT_SEL_VC	2:0	rw	**Credit Select(VC)** This field in conjunction with the CREDIT_SEL_CREDIT_TYPE, CREDIT_SEL_TLP_TYPE, and CREDIT_SEL_HD viewport-select fields determines that data that is returned by the CREDIT_DATA0 and CREDIT_DATA1 data fields. Note: This register field is sticky. 000_B **MIN_VAL**: VC0 111_B **MAX_VAL**: VC7
CREDIT_SEL_CREDIT_TYPE	3	rw	**Credit Select(Credit Type)** This field in conjunction with the CREDIT_SEL_VC, CREDIT_SEL_TLP_TYPE, and CREDIT_SEL_HD viewport-select fields determines that data that is returned by the CREDIT_DATA0 and CREDIT_DATA1 data fields. Note: This register field is sticky. 0_B **RX**: Rx 1_B **TX**: Tx
CREDIT_SEL_TLP_TYPE	5:4	rw	**Credit Select(TLP Type)** This field in conjunction with the CREDIT_SEL_VC, CREDIT_SEL_CREDIT_TYPE, and CREDIT_SEL_HD viewport-select fields determines that data that is returned by the CREDIT_DATA0 and CREDIT_DATA1 data fields. Note: This register field is sticky. 00_B **POSTED**: Posted 01_B **NON_POSTED**: Non-Posted 10_B **COMPLETION**: Completion **others,** Reserved
CREDIT_SEL_HD	6	rw	**Credit Select(HeaderData)** This field in conjunction with the CREDIT_SEL_VC, CREDIT_SEL_CREDIT_TYPE, and CREDIT_SEL_TLP_TYPE viewport-select fields determines that data that is returned by the CREDIT_DATA0 and CREDIT_DATA1 data fields. Note: This register field is sticky. 0_B **HEADER_CREDIT**: Header Credit 1_B **DATA_CREDIT**: Data Credit
RSVDP_7	7	r	**Reserved for future use**

(table continues...)

(continued)

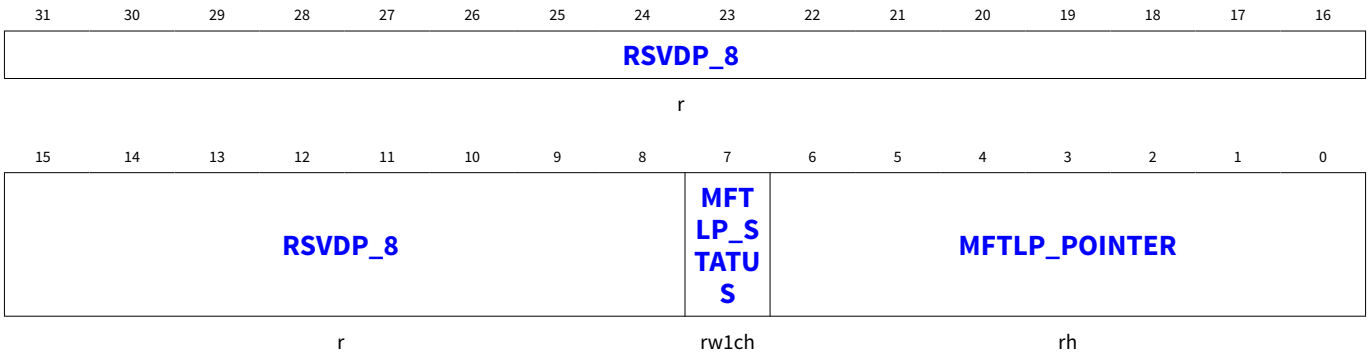
Field	Bits	Type	Description
CREDIT_DATA0	19:8	rh	**Credit Data0** Current FC credit data selected by the CREDIT_SEL_VC, CREDIT_SEL_CREDIT_TYPE, CREDIT_SEL_TLP_TYPE, and CREDIT_SEL_HD viewport-select fields. - Rx: Credit Received Value - Tx: Credit Consumed Value Note: This register field is sticky.
CREDIT_DATA1	31:20	rh	**Credit Data1** Current FC credit data selected by the CREDIT_SEL_VC, CREDIT_SEL_CREDIT_TYPE, CREDIT_SEL_TLP_TYPE, and CREDIT_SEL_HD viewport-select fields. - Rx: Credit Allocated Value - Tx: Credit Limit Value. This value is valid when DLCMSM=0x3(DL_ACTIVE). Note: This register field is sticky.

19.4.2.185 Silicon Debug Status(Layer3)

Silicon Debug Status(Layer3).

SD_STATUS_L3_REG Offset address: 0004244_H

Silicon Debug Status[Layer3] Kernel Reset value: $0000\ 0000_H$



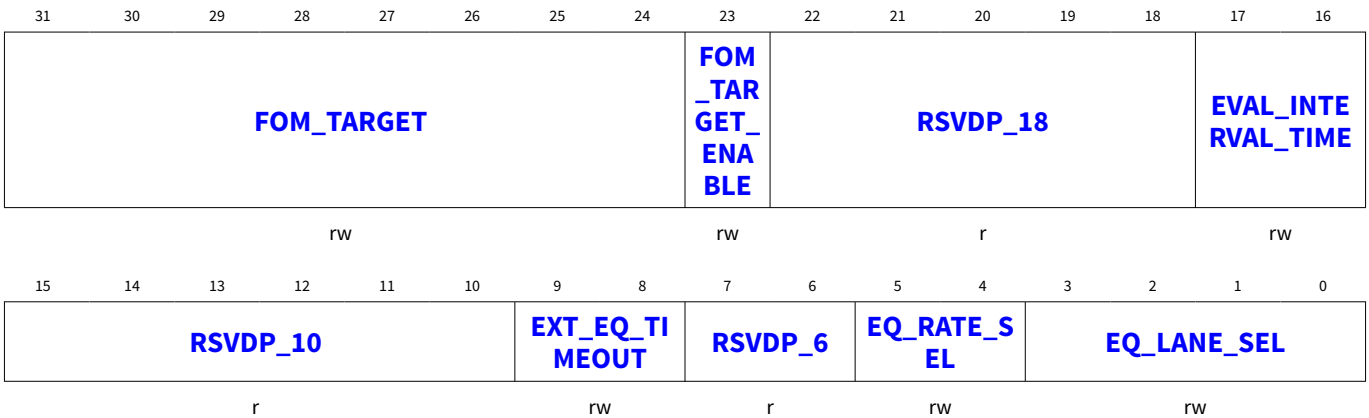
Field	Bits	Type	Description
MFTLP_POINTER	6:0	rh	**First Malformed TLP Error Pointer** Indicates the element of the received first malformed TLP. This pointer is validated by MFTLP_STATUS. All encodings other than the defined encodings are reserved. Note: This register field is sticky. 01_H **ATMC_OP_ALIGN**: AtomicOp address alignment 02_H **ATMC_OP_OPERAND**: AtomicOp operand 03_H **ATMC_OP_BYTE_EN**: AtomicOp byte enable 04_H **TLP_LENGTH_MISMATCH**: TLP length miss match 05_H **MAX_PAYLOAD_SIZE**: Max payload size 06_H **TLP_WITHOUT_TC0**: Message TLP without TC0 07_H **INVALID_TC**: Invalid TC 08_H **UNXPCTD_ROUTE_BIT_MSG_TLP**: Unexpected route bit in Message TLP 09_H **UNXPCTD_CRS_STATUS_CMPL_TLP**: Unexpected CRS status in Completion TLP $0A_H$ **BYTE_ENABLE**: Byte enable $0B_H$ **MEM_ADDR_4KB_BOUNDARY**: Memory Address 4KB boundary $0C_H$ **TLP_PREFIX_RULES**: TLP prefix rules $0D_H$ **TRANSLATION_RULES**: Translation request rules $0E_H$ **INVALID_TLP_TYPE**: Invalid TLP type $0F_H$ **CMPL_RULES**: Completion rules $7F_H$ **APPLICATION**: Application
MFTLP_STATUS	7	rw1ch	**Malformed TLP Status** Indicates malformed TLP has occurred. Note: This register field is sticky. Software write with 1 clears this bit-field.
RSVDP_8	31:8	r	**Reserved for future use**

19.4.2.186 Silicon Debug EQ Control 1

This is a viewport control register. Setting the EQ_RATE_SEL and EQ_LANE_SEL fields in this register determine the per-lane Silicon Debug EQ Status data returned by the SD_EQ_STATUS[1/2/3] viewport registers.

SD_EQ_CONTROL1_REG Offset address: 0004250_H

Silicon Debug EQ Control 1 Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
EQ_LANE_SEL	3:0	rw	**EQ Status Lane Select** Setting this field in conjunction with the EQ_RATE_SEL field determines the per-lane Silicon Debug EQ Status data returned by the SD_EQ_CONTROL[2/3] and SD_EQ_STATUS[1/2/3] viewport registers. Note: This register field is sticky. 0_H **MIN_VAL**: Lane0 F_H **MAX_VAL**: Lane15
EQ_RATE_SEL	5:4	rw	**EQ Status Rate Select** Setting this field in conjunction with the EQ_LANE_SEL field determines the per-lane Silicon Debug EQ Status data returned by the SD_EQ_CONTROL[2/3] and SD_EQ_STATUS[1/2/3] viewport registers. Note: This register field is sticky. 00_B **_8GT**: 8.0GT/s Speed (include ESM data rate) 01_B **_16GT**: 16.0GT/s Speed (include ESM data rate) 10_B **_32GT**: 32.0GT/s Speed **others,** Reserved
RSVDP_6	7:6	r	**Reserved for future use**
EXT_EQ_TIMEOUT	9:8	rw	**Extends EQ Phase2/3 Timeout** This field is used when the LTSSM is in Recovery.EQ2/3. When this field is set, the value of EQ2/3 timeout is extended. Note: This register field is sticky. 00_B **_24MS_OR_32MS**: [EQ Master(DSP in EQ Phase3/USP in EQ Phase2)] 24ms (default) or [EQ Slave(DSP in EQ Phase2/USP in EQ Phase3)] 32ms (default) 01_B **_48MS_OR_56MS**: [EQ Master(DSP in EQ Phase3/USP in EQ Phase2)] 48ms (x2) or [EQ Slave(DSP in EQ Phase2/USP in EQ Phase3)] 56ms (32ms+24ms) 10_B **_240MS_OR_248MS**: [EQ Master(DSP in EQ Phase3/USP in EQ Phase2)] 240ms (x10) or [EQ Slave(DSP in EQ Phase2/USP in EQ Phase3)] 248ms (32ms +9*24ms) 11_B **NO_TIMEOUT**: [EQ Master(DSP in EQ Phase3/USP in EQ Phase2)] No timeout or [EQ Slave(DSP in EQ Phase2/USP in EQ Phase3)] No timeout
RSVDP_10	15:10	r	**Reserved for future use**
EVAL_INTERVAL_TIME	17:16	rw	**Eval Interval Time** Indicates interval time of RxEqEval assertion. This field is used for EQ Master(DSP in EQ Phase3/USP in EQ Phase2). Note: This register field is sticky. 00_B **_500NS**: 500 ns 01_B **_1US**: 1 µs 10_B **_2US**: 2 µs 11_B **_4US**: 4 µs
RSVDP_18	22:18	r	**Reserved for future use**
FOM_TARGET_ENABLE	23	rw	**FOM Target Enable** Enables the FOM_TARGET fields. Note: This register field is sticky.

(table continues...)

(continued)

Field	Bits	Type	Description
FOM_TARGET	31:24	rw	**FOM Target** Indicates figure of merit target criteria value of EQ Master(DSP in EQ Phase3/USP in EQ Phase2). This field is only valid when GEN3_EQ_FB_MODE is 0001b(Figure Of Merit). Note: This register field is sticky.

19.4.2.187 Silicon Debug EQ Control 2

This viewport register returns the value for the rate and lane selected by the EQ_RATE_SEL and EQ_LANE_SEL fields in the SD_EQ_CONTROL1_REG register.

SD_EQ_CONTROL2_REG Offset address: 0004254_H

Silicon Debug EQ Control 2 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_31	FORCE_LOCAL_TX_PRESET_ENABLE	FORCE_LOCAL_RX_HINT_ENABLE	FORCE_LOCAL_TX_COEF_ENABLE	FORCE_LOCAL_TX_PRESET				RSVDP_21			FORCE_LOCAL_RX_HINT			FORCE_LOCAL_TX_POST_CURSOR	
r	rwh	rwh	rwh	rwh				r			rwh			rwh	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FORCE_LOCAL_TX_POST_CURSOR				FORCE_LOCAL_TX_CURSOR						FORCE_LOCAL_TX_PRE_CURSOR					
rwh				rwh						rwh					

Field	Bits	Type	Description
FORCE_LOCAL_TX_PRE_CURSOR	5:0	rwh	**Force Local Transmitter Pre-cursor** Indicates the coefficient value of EQ Slave(DSP in EQ Phase2/USP in EQ Phase3), instead of the value instructed from link partner. Note: This register field is sticky.
FORCE_LOCAL_TX_CURSOR	11:6	rwh	**Force Local Transmitter Cursor** Indicates the coefficient value of EQ Slave(DSP in EQ Phase2/USP in EQ Phase3), instead of the value instructed from link partner. Note: This register field is sticky.
FORCE_LOCAL_TX_POST_CURSOR	17:12	rwh	**Force Local Transmitter Post-Cursor** Indicates the coefficient value of EQ Slave(DSP in EQ Phase2/USP in EQ Phase3), instead of the value instructed from link partner. Note: This register field is sticky.
FORCE_LOCAL_RX_HINT	20:18	rwh	**Force Local Receiver Preset Hint** Indicates the RxPresetHint value of EQ Slave(DSP in EQ Phase2/USP in EQ Phase3), instead of received or set value. If select rate in the EQ_RATE_SEL field is other than 8.0GT/s Speed, this feature is not available. Note: This register field is sticky.

(table continues...)

(continued)

Field	Bits	Type	Description
RSVDP_21	23:21	r	**Reserved for future use**
FORCE_LOCAL_TX_PRESET	27:24	rwh	**Force Local Transmitter Preset** Indicates initial preset value of USP in EQ Slave(EQ Phase2) instead of receiving EQ TS2. If select rate in the EQ_RATE_SEL field is 32.0GT/s Speed, this feature is not available. Note: This register field is sticky.
FORCE_LOCAL_TX_COEF_ENABLE	28	rwh	**Force Local Transmitter Coefficient Enable** Enables the following fields: - FORCE_LOCAL_TX_PRE_CURSOR - FORCE_LOCAL_TX_CURSOR - FORCE_LOCAL_TX_POST_CURSOR Note: This register field is sticky.
FORCE_LOCAL_RX_HINT_ENABLE	29	rwh	**Force Local Receiver Preset Hint Enable** Enables the FORCE_LOCAL_RX_HINT field. If select rate in the EQ_RATE_SEL field is other than 8.0GT/s Speed, this feature is not available. Note: This register field is sticky.
FORCE_LOCAL_TX_PRESET_ENABLE	30	rwh	**Force Local Transmitter Preset Enable** Enables the FORCE_LOCAL_TX_PRESET field. If select rate in the EQ_RATE_SEL field is 32.0GT/s Speed, this feature is not available. Note: This register field is sticky.
RSVDP_31	31	r	**Reserved for future use**

19.4.2.188 Silicon Debug EQ Control 3

This viewport register returns the value for the rate and lane selected by the EQ_RATE_SEL and EQ_LANE_SEL fields in the SD_EQ_CONTROL1_REG register..

SD_EQ_CONTROL3_REG Offset address: 0004258_H

Silicon Debug EQ Control 3 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_29			FORCE_REMOTE_TX_COEF_ENABLE	RSVDP_18										FORCE_REMOTE_TX_POST_CURSOR	
r			rwh	r										rwh	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FORCE_REMOTE_TX_POST_CURSOR				FORCE_REMOTE_TX_CURSOR						FORCE_REMOTE_TX_PRE_CURSOR					
rwh				rwh						rwh					

Field	Bits	Type	Description
FORCE_REMOTE_TX_PRE_CURSOR	5:0	rwh	**Force Remote Transmitter Pre-Cursor** Indicates the coefficient value of EQ Master(DSP in EQ Phase3/USP in EQ Phase2), instead of the value instructed from local phy in dirchange mode. Note: This register field is sticky.
FORCE_REMOTE_TX_CURSOR	11:6	rwh	**Force Remote Transmitter Cursor** Indicates the coefficient value of EQ Master(DSP in EQ Phase3/USP in EQ Phase2), instead of the value instructed from local phy in dirchange mode. Note: This register field is sticky.
FORCE_REMOTE_TX_POST_CURSOR	17:12	rwh	**Force Remote Transmitter Post-Cursor** Indicates the coefficient value of EQ Master(DSP in EQ Phase3/USP in EQ Phase2), instead of the value instructed from local phy in dirchange mode. Note: This register field is sticky.
RSVDP_18	27:18	r	**Reserved for future use**
FORCE_REMOTE_TX_COEF_ENABLE	28	rwh	**Force Remote Transmitter Coefficient Enable** Enables the following fields: - FORCE_REMOTE_TX_PRE_CURSOR - FORCE_REMOTE_TX_CURSOR - FORCE_REMOTE_TX_POST_CURSOR This function can only be used when GEN3_EQ_FB_MODE = 0000b(Direction Change) Note: This register field is sticky.
RSVDP_29	31:29	r	**Reserved for future use**

19.4.2.189 Silicon Debug EQ Status 1

This viewport register returns the first of three words of Silicon Debug EQ Status data for the rate and lane selected by the EQ_RATE_SEL and EQ_LANE_SEL fields in the SD_EQ_CONTROL1_REG register. The following fields are available when Equalization finished unsuccessfully(EQ_CONVERGENCE_INFO=2). - EQ_RULEA_VIOLATION - EQ_RULEB_VIOLATION - EQ_RULEC_VIOLATION - EQ_REJECT_EVENT

SD_EQ_STATUS1_REG Offset address: 0004260_H

Silicon Debug EQ Status 1 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_8															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVDP_8								EQ_REJECT_EVENT	EQ_RULEC_VIOLATION	EQ_RULEB_VIOLATION	EQ_RULEA_VIOLATION	RSVDP_3	EQ_CONVERGENCE_INFO		EQ_SEQUENCE
r								rh	rh	rh	rh	r	rh		rh

Field	Bits	Type	Description
EQ_SEQUENCE	0	rh	**EQ Sequence** Indicates that the controller is starting the equalization sequence. Note: This register field is sticky.

(table continues...)

(continued)

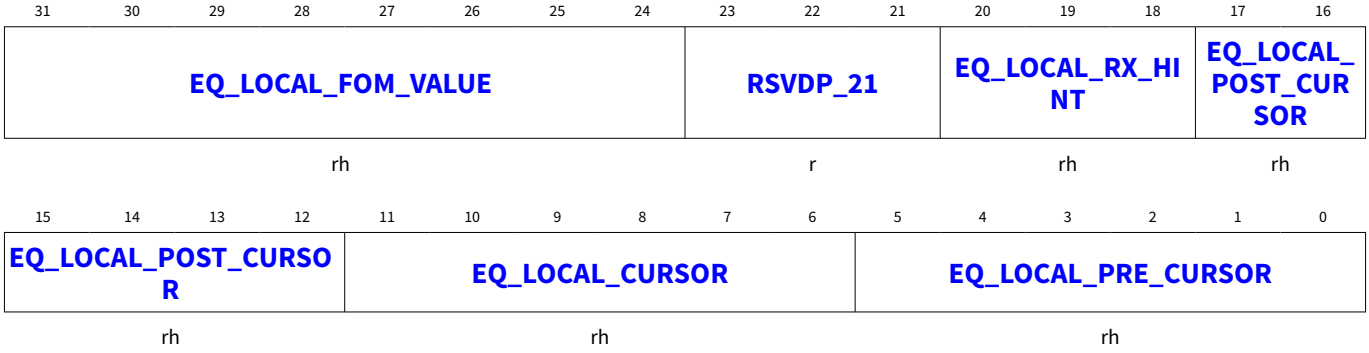
Field	Bits	Type	Description
EQ_CONVERGENCE_INFO	2:1	rh	**EQ Convergence Info** Indicates equalization convergence information. This bit is automatically cleared when the controller starts EQ Master phase again. Note: This register field is sticky. 00_B **EQ_NOT_ATTEMPTED**: Equalization is not attempted 01_B **EQ_SUCCESSFUL**: Equalization finished successfully 10_B **EQ_UNSUCCESSFUL**: Equalization finished unsuccessfully 11_B **RSVD**: Reserved
RSVDP_3	3	r	**Reserved for future use**
EQ_RULEA_VIOLATION	4	rh	**EQ Rule A Violation** Indicates that coefficients rule A violation is detected in the values provided by PHY using direction change method during EQ Master phase(DSP in EQ Phase3/USP in EQ Phase2). The coefficients rules A correspond to the rules a) from section "Rules for Transmitter Coefficents" in the PCI Express Base Specification. This bit is automatically cleared when the controller starts EQ Master phase again. Note: This register field is sticky.
EQ_RULEB_VIOLATION	5	rh	**EQ Rule B Violation** Indicates that coefficients rule B violation is detected in the values provided by PHY using direction change method during EQ Master phase(DSP in EQ Phase3/USP in EQ Phase2). The coefficients rules B correspond to the rules b) from section "Rules for Transmitter Coefficents" in the PCI Express Base Specification. This bit is automatically cleared when the controller starts EQ Master phase again. Note: This register field is sticky.
EQ_RULEC_VIOLATION	6	rh	**EQ Rule C Violation** Indicates that coefficients rule C violation is detected in the values provided by PHY using direction change method during EQ Master phase(DSP in EQ Phase3/USP in EQ Phase2). The coefficients rule C correspond to the rules c) from section "Rules for Transmitter Coefficents" in the PCI Express Base Specification. This bit is automatically cleared when the controller starts EQ Master phase again. Note: This register field is sticky.
EQ_REJECT_EVENT	7	rh	**EQ Reject Event** Indicates that the controller receives two consecutive TS1 OS w/ Reject=1b during EQ Master phase(DSP in EQ Phase3/USP in EQ Phase2). This bit is automatically cleared when the controller starts EQ Master phase again. Note: This register field is sticky.
RSVDP_8	31:8	r	**Reserved for future use**

19.4.2.190 Silicon Debug EQ Status 2

This viewport register returns the second of three words of Silicon Debug EQ Status data for the rate and lane selected by the EQ_RATE_SEL and EQ_LANE_SEL fields in the SD_EQ_CONTROL1_REG register. Each field is available when Equalization finished successfully(EQ_CONVERGENCE_INFO=1).

SD_EQ_STATUS2_REG Offset address: 0004264_H

Silicon Debug EQ Status 2 Kernel Reset value: $0000\ 0000_H$



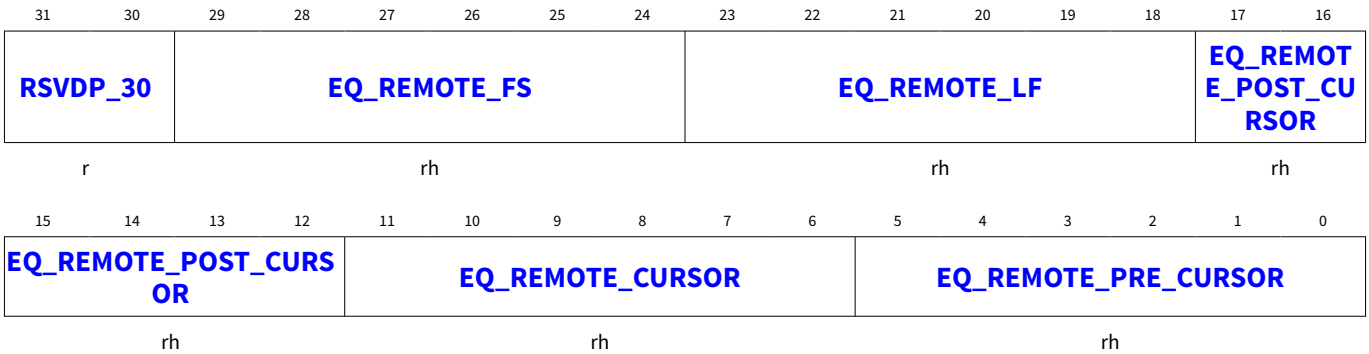
Field	Bits	Type	Description
EQ_LOCAL_PRE_CURSOR	5:0	rh	**EQ Local Pre-Cursor** Indicates Local pre cursor coefficient value. Note: This register field is sticky.
EQ_LOCAL_CURSOR	11:6	rh	**EQ Local Cursor** Indicates Local cursor coefficient value. Note: This register field is sticky.
EQ_LOCAL_POST_CURSOR	17:12	rh	**EQ Local Post-Cursor** Indicates Local post cursor coefficient value. Note: This register field is sticky.
EQ_LOCAL_RX_HINT	20:18	rh	**EQ Local Receiver Preset Hint** Indicates Local Receiver Preset Hint value. If select rate in the EQ_RATE_SEL field is other than 8.0GT/s Speed, this feature is not available. Note: This register field is sticky.
RSVDP_21	23:21	r	**Reserved for future use**
EQ_LOCAL_FOM_VALUE	31:24	rh	**EQ Local Figure of Merit** Indicates Local maximum Figure of Merit value. Note: This register field is sticky.

19.4.2.191 Silicon Debug EQ Status 3

This viewport register returns the third of three words of Silicon Debug EQ Status data for the rate and lane selected by the EQ_RATE_SEL and EQ_LANE_SEL fields in the SD_EQ_CONTROL1_REG register. Each field is available when Equalization finished successfully(EQ_CONVERGENCE_INFO=1).

SD_EQ_STATUS3_REG Offset address: 0004268_H

Silicon Debug EQ Status 3 Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
EQ_REMOTE_PRE_CURSOR	5:0	rh	**EQ Remote Pre-Cursor** Indicates Remote pre cursor coefficient value. Note: This register field is sticky.
EQ_REMOTE_CURSOR	11:6	rh	**EQ Remote Cursor** Indicates Remote cursor coefficient value. Note: This register field is sticky.
EQ_REMOTE_POST_CURSOR	17:12	rh	**EQ Remote Post-Cursor** Indicates Remote post cursor coefficient value. Note: This register field is sticky.
EQ_REMOTE_LF	23:18	rh	**EQ Remote LF** Indicates Remote LF value. Note: This register field is sticky.
EQ_REMOTE_FS	29:24	rh	**EQ Remote FS** Indicates Remote FS value. Note: This register field is sticky.
RSVDP_30	31:30	r	**Reserved for future use**

19.4.2.192 PCIe Extended capability ID, Capability version and Next capability offset

This Register provides capability ID, Capability version, and Next capability offset for PCIe Extended capability stucture.

RASDP_EXT_CAP_HDR_OFF

PCIe Extended capability ID, Capability version and Next capability offset

Offset address: 0004280_H

Kernel Reset value: $2B81\,000B_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
NEXT_OFFSET												CAP			
rw												rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ID															
rw															

Field	Bits	Type	Description
ID	15:0	rw	**PCI Express Extended Capability ID** This field is a PCI-SIG defined ID number that indicates the nature and format of the Extended Capability. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky. 0000_H **MIN_VAL**: Min value $FFFF_H$ **MAX_VAL**: Max value

(table continues...)

(continued)

Field	Bits	Type	Description
CAP	19:16	rw	**Capability Version** This field is a PCI-SIG defined version number that indicates the version of the Capability structure present. A version of the specification that changes the Extended Capability in a way that is not otherwise identifiable (e.g., through a new Capability field) is permitted to increment this field. All such changes to the Capability structure must be software-compatible. Software must check for Capability Version numbers that are greater than or equal to the highest number defined when the software is written, as Functions reporting any such Capability Version numbers will contain a Capability structure that is compatible with that piece of software. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky. 0_H **MIN_VAL**: Min value F_H **MAX_VAL**: Max value
NEXT_OFFSET	31:20	rw	**Next Capability Offset** This field contains the offset to the next PCI Express Capability structure or 000h if no other items exist in the linked list of Capabilities. For Extended Capabilities implemented in Configuration Space, this offset is relative to the beginning of PCI-compatible Configuration Space and thus must always be either 000h (for terminating list of Capabilities) or greater than 0FFh. The bottom 2 bits of this offset are Reserved and must be implemented as 00b although software must mask them to allow for future uses of these bits. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky. 000_H **MIN_VAL**: Min value FFF_H **MAX_VAL**: Max value

19.4.2.193 Vendor Specific Header

This Register provides VSEC Length, VSEC ID and VSEC Rev (Version Number). Vendor-specific software must qualify the associated Vendor ID of the PCI Express Function or RCRB before attempting to interpret the values in the VSEC ID or VSEC Rev fields.

RASDP_VENDOR_SPECIFIC_HDR_OFF Offset address: 0004284_H

Vendor Specific Header Kernel Reset value: $0381\ 0001_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
VSEC_LENGTH												VSEC_REV			
r												r			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VSEC_ID															
r															

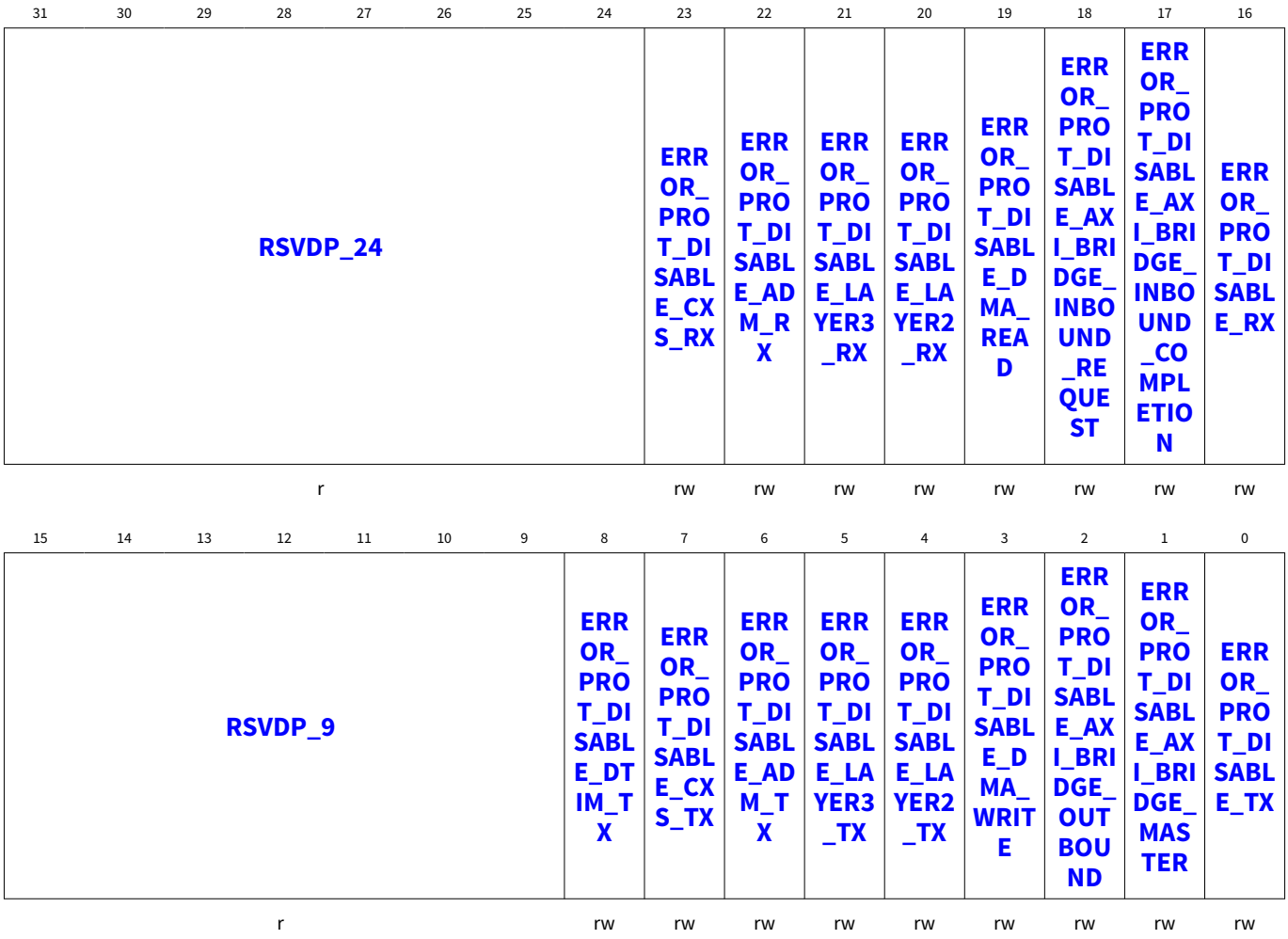
Field	Bits	Type	Description
VSEC_ID	15:0	r	**VSEC ID** This field is a vendor-defined ID number that indicates the nature and format of the VSEC structure. Software must qualify the Vendor ID before interpreting this field. Note: This register field is sticky. 0000_H **MIN_VAL**: Min value $FFFF_H$ **MAX_VAL**: Max value
VSEC_REV	19:16	r	**VSEC Rev** This field is a vendor-defined version number that indicates the version of the VSEC structure. Software must qualify the Vendor ID and VSEC ID before interpreting this field. Note: This register field is sticky. 0_H **MIN_VAL**: Min value F_H **MAX_VAL**: Max value
VSEC_LENGTH	31:20	r	**VSEC Length** This field indicates the number of bytes in the entire VSEC structure, including the Vendor-Specific Extended Capability Header, the Vendor-Specific Header, and the vendor-specific registers. Note: This register field is sticky. 000_H **MIN_VAL**: Min value FFF_H **MAX_VAL**: Max value

19.4.2.194 ECC Error Correction and Control Register

Allows you to disable ECC error correction for RAMs and datapath. When the AXI Bridge Module is implemented and the master / slave clocks are asynchronous to the PCIe native controller clock (core_clk), you must not write this register while operations are in progress in the AXI master / slave interface.

RASDP_ERROR_PROT_CTRL_OFF Offset address: 0004288_H

ECC Error Correction and Control Register Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
ERROR_PROT_DISABLE_TX	0	rw	**Global error correction disable for all Tx layers** Does not disable the error detection reporting for 1-bit and 2-bit ECC errors. Note: This register field is sticky.
ERROR_PROT_DISABLE_AXI_BRIDGE_MASTER	1	rw	**Error correction disable for AXI bridge master completion buffer** Note: This register field is sticky.
ERROR_PROT_DISABLE_AXI_BRIDGE_OUTBOUND	2	rw	**Error correction disable for AXI bridge outbound request path** Note: This register field is sticky.
ERROR_PROT_DISABLE_DMA_WRITE	3	rw	**Error correction disable for DMA write engine** Note: This register field is sticky.
ERROR_PROT_DISABLE_LAYER2_TX	4	rw	**Error correction disable for layer 2 Tx path** Note: This register field is sticky.

(table continues...)

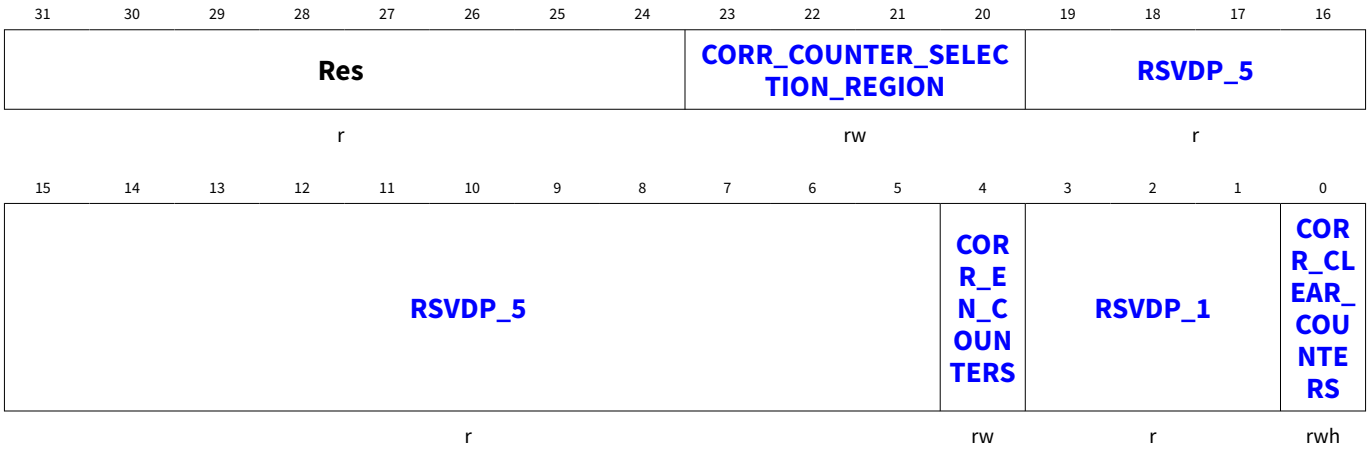
(continued)

Field	Bits	Type	Description
ERROR_PROT_DISABLE_LAYER3_TX	5	rw	**Error correction disable for layer 3 Tx path** Note: This register field is sticky.
ERROR_PROT_DISABLE_ADM_TX	6	rw	**Error correction disable for Adm Tx path** Note: This register field is sticky.
ERROR_PROT_DISABLE_CXS_TX	7	rw	**Error correction disable for CXS Rx path (PCIe Tx path)** Note: This register field is sticky.
ERROR_PROT_DISABLE_DTIM_TX	8	rw	**Error correction disable for DTIM Tx path** Note: This register field is sticky.
RSVDP_9	15:9	r	**Reserved for future use**
ERROR_PROT_DISABLE_RX	16	rw	**Global error correction disable for all Rx layers** Note: This register field is sticky.
ERROR_PROT_DISABLE_AXI_BRIDGE_INBOUND_COMPLETION	17	rw	**Error correction disable for AXI bridge inbound completion composer** Does not disable the error detection reporting for 1-bit and 2-bit ECC errors. Note: This register field is sticky.
ERROR_PROT_DISABLE_AXI_BRIDGE_INBOUND_REQUEST	18	rw	**Error correction disable for AXI bridge inbound request path** Note: This register field is sticky.
ERROR_PROT_DISABLE_DMA_READ	19	rw	**Error correction disable for DMA read engine** Note: This register field is sticky.
ERROR_PROT_DISABLE_LAYER2_RX	20	rw	**Error correction disable for layer 2 Rx path** Note: This register field is sticky.
ERROR_PROT_DISABLE_LAYER3_RX	21	rw	**Error correction disable for layer 3 Rx path** Note: This register field is sticky.
ERROR_PROT_DISABLE_ADM_RX	22	rw	**Error correction disable for ADM Rx path** Note: This register field is sticky.
ERROR_PROT_DISABLE_CXS_RX	23	rw	**Error correction disable for CXS Tx path (PCIe Rx path)** Note: This register field is sticky.
RSVDP_24	31:24	r	**Reserved for future use**

19.4.2.195 Corrected Error (1-bit ECC) Counter Selection and Control Register

This is a viewport control register. Setting the CORR_COUNTER_SELECTION_REGION and CORR_COUNTER_SELECTION fields in this register determine the counter data returned by the RASDP_CORR_COUNT_REPORT_OFF viewport data register.

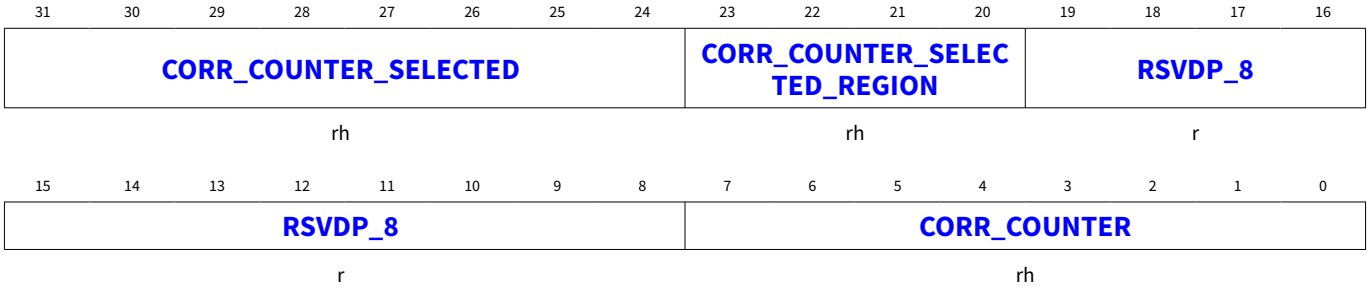
19 Peripheral Component Interconnect Express (PCIe)



Field	Bits	Type	Description
CORR_CLEAR_COUNTERS	0	rwh	**Clear all correctable error counters** Reading register field always returns a '0'.
RSVDP_1	3:1	r	**Reserved for future use**
CORR_EN_COUNTERS	4	rw	**Enable correctable errors counters** The counters are enabled by default. Note: This register field is sticky. 0_B **COUNTER_FROZEN**: Counters are frozen 1_B **COUNTER_EN**: Counters increment when the controller detects a correctable error
RSVDP_5	19:5	r	**Reserved for future use**
CORR_COUNTER_SELECTION_REGION	23:20	rw	**Select correctable counter region** All encodings other than the defined encodings are reserved. Note: This register field is sticky. 0_H **ADM_RX_PATH**: Region select for Adm Rx path 1_H **L3_RX_PATH**: Region select for layer 3 Rx path 2_H **L2_RX_PATH**: Region select for layer 2 Rx path 3_H **DMA_INBOUND_PATH**: Region select for DMA inbound path 4_H **AXI_INBOUND_REQ_PATH**: Region select for AXI bridge inbound request path 5_H **AXI_INBOUND_CMPL_PATH**: Region select for AXI bridge inbound completion composer path 6_H **ADM_TX_PATH**: Region select for Adm Tx path 7_H **L3_TX_PATH**: Region select for layer 3 Tx path 8_H **L2_TX_PATH**: Region select for layer 2 Tx path 9_H **DMA_OUTBOUND_PATH**: Region select for DMA outbound path A_H **AXI_OUTBOUND_REQ_PATH**: Region select for AXI bridge outbound request path B_H **AXI_OUTBOUND_CMPL_PATH**: Region select for AXI bridge outbound master completion buffer path C_H **CXS_TX_RX_PATH**: Region select for CXS Tx/Rx path D_H **DTIM_TX_RX_PATH**: Region select for DTIM Tx/Rx path **others,** Reserved

19.4.2.196 Corrected Error (1-bit ECC) Counter Data Register

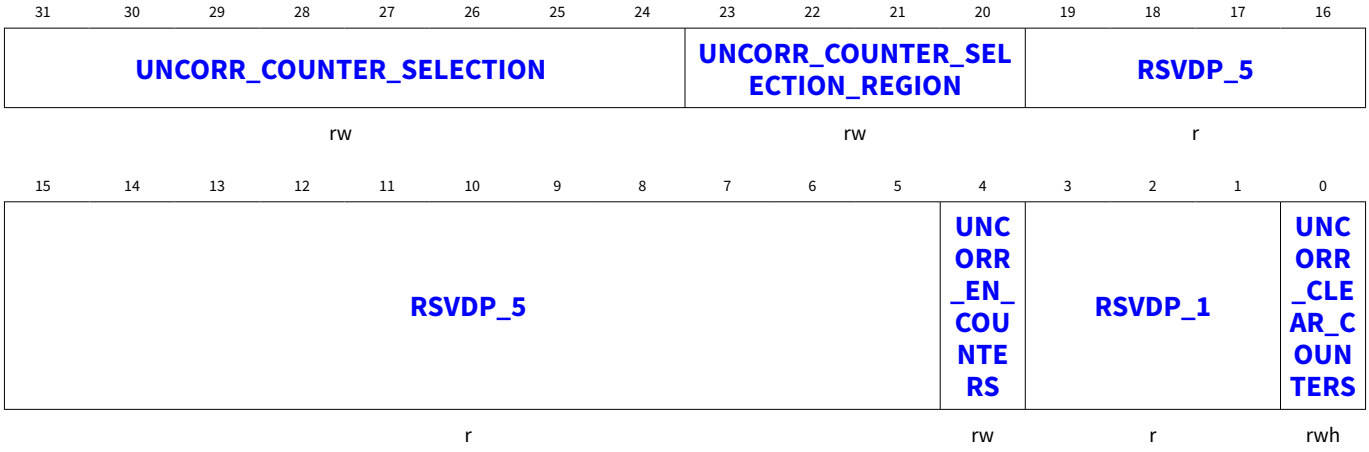
This viewport register returns the counter data selected by the CORR_COUNTER_SELECTION_REGION and CORR_COUNTER_SELECTION fields in the RASDP_CORR_COUNTER_CTRL_OFF register.



Field	Bits	Type	Description
CORR_COUNTER	7:0	rh	**Current corrected error count for the selected counter** Note: This register field is sticky.
RSVDP_8	19:8	r	**Reserved for future use**
CORR_COUNTER_SELECTED_REGION	23:20	rh	**Selected correctable counter region** All encodings other than the defined encodings are reserved. Note: This register field is sticky. 0_H **ADM_RX_PATH**: Region select for Adm Rx path 1_H **L3_RX_PATH**: Region select for layer 3 Rx path 2_H **L2_RX_PATH**: Region select for layer 2 Rx path 3_H **DMA_INBOUND_PATH**: Region select for DMA inbound path 4_H **AXI_INBOUND_REQ_PATH**: Region select for AXI bridge inbound request path 5_H **AXI_INBOUND_CMPL_PATH**: Region select for AXI bridge inbound completion composer path 6_H **ADM_TX_PATH**: Region select for Adm Tx path 7_H **L3_TX_PATH**: Region select for layer 3 Tx path 8_H **L2_TX_PATH**: Region select for layer 2 Tx path 9_H **DMA_OUTBOUND_PATH**: Region select for DMA outbound path A_H **AXI_OUTBOUND_REQ_PATH**: Region select for AXI bridge outbound request path B_H **AXI_OUTBOUND_CMPL_PATH**: Region select for AXI bridge outbound master completion buffer path C_H **CXS_TX_RX_PATH**: Region select for CXS Tx/Rx path D_H **DTIM_TX_RX_PATH**: Region select for DTIM Tx/Rx path **others,** Reserved
CORR_COUNTER_SELECTED	31:24	rh	**Counter selection** Returns the value set in the CORR_COUNTER_SELECTION field of the RASDP_CORR_COUNTER_CTRL_OFF register. Note: This register field is sticky.

19.4.2.197 Uncorrected Error (2-bit ECC and parity) Counter Selection and Control Register

This is a viewport control register. Setting the UNCORR_COUNTER_SELECTION_REGION and UNCORR_COUNTER_SELECTION fields in this register determine the counter data returned by the RASDP_UNCORR_COUNT_REPORT_OFF viewport data register.



Field	Bits	Type	Description
UNCORR_CLEAR_COUNTERS	0	rwh	**Clear uncorrectable errors counters** When asserted causes all counters tracking the uncorrectable errors to be cleared. Reading register field always returns a '0'.
RSVDP_1	3:1	r	**Reserved for future use**
UNCORR_EN_COUNTERS	4	rw	**Enable uncorrectable errors counters** The counters are enabled by default. Note: This register field is sticky. 0_B **COUNTER_FROZEN**: Counters are frozen 1_B **ENABLE_COUNTER**: Enables the counters to increment on detected correctable errors
RSVDP_5	19:5	r	**Reserved for future use**

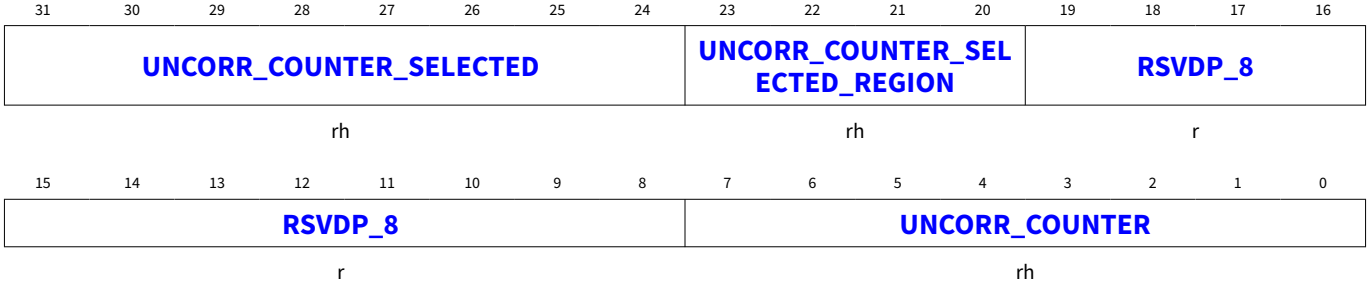
(table continues...)

(continued)

Field	Bits	Type	Description
UNCORR_COUNTER_SELECTION_REGION	23:20	rw	**Select uncorrectable counter region** All encodings other than the defined encodings are reserved. Note: This register field is sticky. 0_H **ADM_RX_PATH**: Region select for Adm Rx path 1_H **L3_RX_PATH**: Region select for layer 3 Rx path 2_H **L2_RX_PATH**: Region select for layer 2 Rx path 3_H **DMA_INBOUND_PATH**: Region select for DMA inbound path 4_H **AXI_INBOUND_REQ_PATH**: Region select for AXI bridge inbound request path 5_H **AXI_INBOUND_CMPL_PATH**: Region select for AXI bridge inbound completion composer path 6_H **ADM_TX_PATH**: Region select for Adm Tx path 7_H **L3_TX_PATH**: Region select for layer 3 Tx path 8_H **L2_TX_PATH**: Region select for layer 2 Tx path 9_H **DMA_OUTBOUND_PATH**: Region select for DMA outbound path A_H **AXI_OUTBOUND_REQ_PATH**: Region select for AXI bridge outbound request path B_H **AXI_OUTBOUND_CMPL_PATH**: Region select for AXI bridge outbound master completion buffer path C_H **CXS_TX_RX_PATH**: Region select for CXS Tx/Rx path D_H **DTIM_TX_RX_PATH**: Region select for DTIM Tx/Rx path **others,** Reserved
UNCORR_COUNTER_SELECTION	31:24	rw	**Counter selection** This field selects the counter ID (within the region defined by UNCORR_COUNTER_SELECTION_REGION) whose contents can be read from the RASDP_UNCORR_COUNT_REPORT_OFF register. You can cycle this field value from 0 to 255 to access all counters according to the detailed report of check points at http://www.synopsys.com/dw/doc.php/iip/DWC_pcie/latest/doc/RASDP_CheckPoints.pdf Note: This register field is sticky.

19.4.2.198 Uncorrected Error (2-bit ECC and parity) Counter Data Register

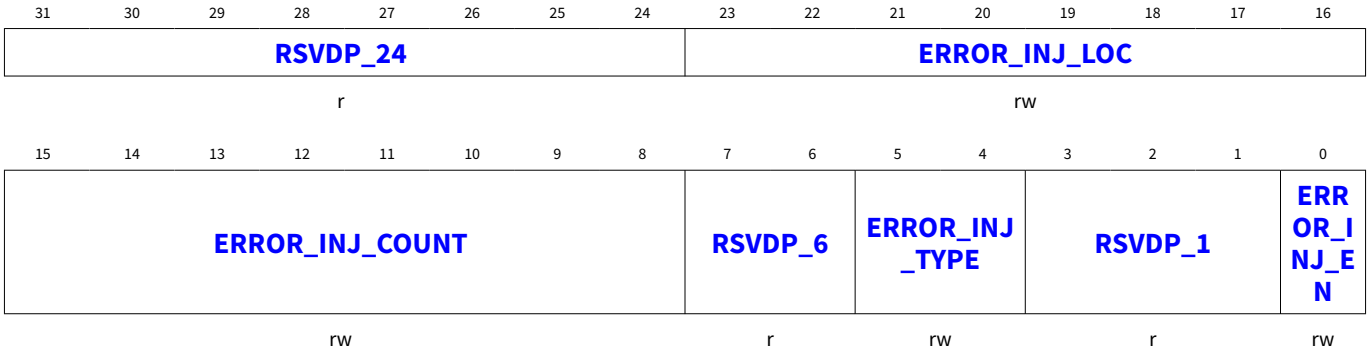
This viewport register returns the counter data selected by the UNCORR_COUNTER_SELECTION_REGION and UNCORR_COUNTER_SELECTION fields in the RASDP_UNCORR_COUNTER_CTRL_OFF register.



Field	Bits	Type	Description
UNCORR_COUNTER	7:0	rh	**Current uncorrected error count for the selected counter** Note: This register field is sticky.
RSVDP_8	19:8	r	**Reserved for future use**
UNCORR_COUNTER_SELECTED_REGION	23:20	rh	**Selected uncorrectable counter region** All encodings other than the defined encodings are reserved. Note: This register field is sticky. 0_H **ADM_RX_PATH**: Region select for Adm Rx path 1_H **L3_RX_PATH**: Region select for layer 3 Rx path 2_H **L2_RX_PATH**: Region select for layer 2 Rx path 3_H **DMA_INBOUND_PATH**: Region select for DMA inbound path 4_H **AXI_INBOUND_REQ_PATH**: Region select for AXI bridge inbound request path 5_H **AXI_INBOUND_CMPL_PATH**: Region select for AXI bridge inbound completion composer path 6_H **ADM_TX_PATH**: Region select for Adm Tx path 7_H **L3_TX_PATH**: Region select for layer 3 Tx path 8_H **L2_TX_PATH**: Region select for layer 2 Tx path 9_H **DMA_OUTBOUND_PATH**: Region select for DMA outbound path A_H **AXI_OUTBOUND_REQ_PATH**: Region select for AXI bridge outbound request path B_H **AXI_OUTBOUND_CMPL_PATH**: Region select for AXI bridge outbound master completion buffer path C_H **CXS_TX_RX_PATH**: Region select for CXS Tx/Rx path D_H **DTIM_TX_RX_PATH**: Region select for DTIM Tx/Rx path **others,** Reserved
UNCORR_COUNTER_SELECTED	31:24	rh	**Counter selection** Returns the value set in the UNCORR_COUNTER_SELECTION field of the RASDP_UNCORR_COUNTER_CTRL_OFF register. Note: This register field is sticky.

19.4.2.199 Error Injection Control Register

Error injection control for the following features: - 1-bit or 2-bit injection - Continuous or fixed-number (n) injection modes - Global enable/disable - Selectable location where injection occurs



Field	Bits	Type	Description
ERROR_INJ_EN	0	rw	**Error injection global enable** When set, enables the error insertion logic.
RSVDP_1	3:1	r	**Reserved for future use**
ERROR_INJ_TYPE	5:4	rw	**Error injection type** Note: This register field is sticky. 00_B **NONE**: none 01_B **_1_BIT**: 1-bit 10_B **_2_BIT**: 2-bit **others,** Reserved
RSVDP_6	7:6	r	**Reserved for future use**
ERROR_INJ_COUNT	15:8	rw	**Error injection count** If value is n, n amount of errors injected. Note: This register field is sticky. 00_H **ERR_INJ_EVERY_TLP**: Errors are inserted in every TLP until you clear ERROR_INJ_EN 01_H **ERR_INJ_1**: One errors injected 02_H **ERR_INJ_2**: Two errors injected **others,** Reserved
ERROR_INJ_LOC	23:16	rw	**Error injection location** Selects where error injection takes place. You can cycle this field value from 0 to 255 to access all locations according to the detailed report of check points at RASDP_CheckPoints.pdf Note: This register field is sticky.
RSVDP_24	31:24	r	**Reserved for future use**

19.4.2.200 Corrected Errors Location Register

Corrected errors locations.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LOC_LAST_CORR_ERROR								REG_LAST_CORR_ERROR				RSVDP_16			
rh								rh				r			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LOC_FIRST_CORR_ERROR								REG_FIRST_CORR_ERROR				RSVDP_0			
rh								rh				r			

Field	Bits	Type	Description
RSVDP_0	3:0	r	**Reserved for future use**

(table continues...)

(continued)

Field	Bits	Type	Description
REG_FIRST_CORR_ERROR	7:4	rh	**Region of the first corrected error** All encodings other than the defined encodings are reserved. Note: This register field is sticky. 0_H **ADM_RX_PATH**: Region select for Adm Rx path 1_H **L3_RX_PATH**: Region select for layer 3 Rx path 2_H **L2_RX_PATH**: Region select for layer 2 Rx path 3_H **DMA_INBOUND_PATH**: Region select for DMA inbound path 4_H **AXI_INBOUND_REQ_PATH**: Region select for AXI bridge inbound request path 5_H **AXI_INBOUND_CMPL_PATH**: Region select for AXI bridge inbound completion composer path 6_H **ADM_TX_PATH**: Region select for Adm Tx path 7_H **L3_TX_PATH**: Region select for layer 3 Tx path 8_H **L2_TX_PATH**: Region select for layer 2 Tx path 9_H **DMA_OUTBOUND_PATH**: Region select for DMA outbound path A_H **AXI_OUTBOUND_REQ_PATH**: Region select for AXI bridge outbound request path B_H **AXI_OUTBOUND_CMPL_PATH**: Region select for AXI bridge outbound master completion buffer path C_H **CXS_TX_RX_PATH**: Region select for CXS Tx/Rx path D_H **DTIM_TX_RX_PATH**: Region select for DTIM Tx/Rx path **others,** Reserved
LOC_FIRST_CORR_ERROR	15:8	rh	**Location/ID of the first corrected error within the region defined by REG_FIRST_CORR_ERROR** You can cycle this field value from 0 to 255 to access all counters according to the detailed report of check points RASDP_CheckPoints.pdf Note: This register field is sticky.
RSVDP_16	19:16	r	**Reserved for future use**

(table continues...)

(continued)

Field	Bits	Type	Description
REG_LAST_CORR_ERROR	23:20	rh	**Region of the last corrected error** All encodings other than the defined encodings are reserved. Note: This register field is sticky. 0_H **ADM_RX_PATH**: Region select for Adm Rx path 1_H **L3_RX_PATH**: Region select for layer 3 Rx path 2_H **L2_RX_PATH**: Region select for layer 2 Rx path 3_H **DMA_INBOUND_PATH**: Region select for DMA inbound path 4_H **AXI_INBOUND_REQ_PATH**: Region select for AXI bridge inbound request path 5_H **AXI_INBOUND_CMPL_PATH**: Region select for AXI bridge inbound completion composer path 6_H **ADM_TX_PATH**: Region select for Adm Tx path 7_H **L3_TX_PATH**: Region select for layer 3 Tx path 8_H **L2_TX_PATH**: Region select for layer 2 Tx path 9_H **DMA_OUTBOUND_PATH**: Region select for DMA outbound path A_H **AXI_OUTBOUND_REQ_PATH**: Region select for AXI bridge outbound request path B_H **AXI_OUTBOUND_CMPL_PATH**: Region select for AXI bridge outbound master completion buffer path C_H **CXS_TX_RX_PATH**: Region select for CXS Tx/Rx path D_H **DTIM_TX_RX_PATH**: Region select for DTIM Tx/Rx path **others,** Reserved
LOC_LAST_CORR_ERROR	31:24	rh	**Location/ID of the last corrected error within the region defined by REG_LAST_CORR_ERROR** You can cycle this field value from 0 to 255 to access all counters according to the detailed report of check points at http://www.synopsys.com/dw/doc.php/iip/DWC_pcie/latest/doc/RASDP_CheckPoints.pdf Note: This register field is sticky.

19.4.2.201 Uncorrected Errors Location Register

Uncorrected errors locations. For more details, see the RAS Data Protection (DP) section in the Core Operations chapter of the user manual.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LOC_LAST_UNCORR_ERROR								REG_LAST_UNCORR_ERROR				RSVDP_16			
rh								rh				r			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LOC_FIRST_UNCORR_ERROR								REG_FIRST_UNCORR_ERROR				RSVDP_0			
rh								rh				r			

Field	Bits	Type	Description
RSVDP_0	3:0	r	**Reserved for future use**

(table continues...)

(continued)

Field	Bits	Type	Description
REG_FIRST_UNCORR_ERROR	7:4	rh	**Region of the first uncorrected error** All encodings other than the defined encodings are reserved. Note: This register field is sticky. 0_H **ADM_RX_PATH**: Region select for Adm Rx path 1_H **L3_RX_PATH**: Region select for layer 3 Rx path 2_H **L2_RX_PATH**: Region select for layer 2 Rx path 3_H **DMA_INBOUND_PATH**: Region select for DMA inbound path 4_H **AXI_INBOUND_REQ_PATH**: Region select for AXI bridge inbound request path 5_H **AXI_INBOUND_CMPL_PATH**: Region select for AXI bridge inbound completion composer path 6_H **ADM_TX_PATH**: Region select for Adm Tx path 7_H **L3_TX_PATH**: Region select for layer 3 Tx path 8_H **L2_TX_PATH**: Region select for layer 2 Tx path 9_H **DMA_OUTBOUND_PATH**: Region select for DMA outbound path A_H **AXI_OUTBOUND_REQ_PATH**: Region select for AXI bridge outbound request path B_H **AXI_OUTBOUND_CMPL_PATH**: Region select for AXI bridge outbound master completion buffer path C_H **CXS_TX_RX_PATH**: Region select for CXS Tx/Rx path D_H **DTIM_TX_RX_PATH**: Region select for DTIM Tx/Rx path **others,** Reserved
LOC_FIRST_UNCORR_ERROR	15:8	rh	**Location/ID of the first uncorrected error within the region defined by REG_FIRST_UNCORR_ERROR** You can cycle this field value from 0 to 255 to access all counters according to the detailed report of check points at http://www.synopsys.com/dw/doc.php/iip/DWC_pcie/latest/doc/RASDP_CheckPoints.pdf Note: This register field is sticky.
RSVDP_16	19:16	r	**Reserved for future use**

(table continues...)

(continued)

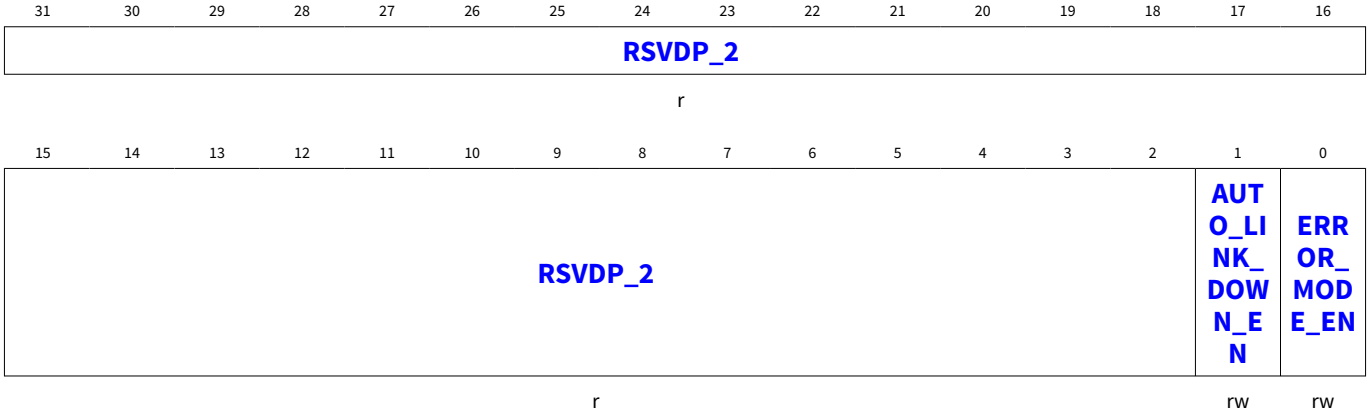
Field	Bits	Type	Description
REG_LAST_UNCORR_ERROR	23:20	rh	**Region of the last uncorrected error** All encodings other than the defined encodings are reserved. Note: This register field is sticky. 0_H **ADM_RX_PATH**: Region select for Adm Rx path 1_H **L3_RX_PATH**: Region select for layer 3 Rx path 2_H **L2_RX_PATH**: Region select for layer 2 Rx path 3_H **DMA_INBOUND_PATH**: Region select for DMA inbound path 4_H **AXI_INBOUND_REQ_PATH**: Region select for AXI bridge inbound request path 5_H **AXI_INBOUND_CMPL_PATH**: Region select for AXI bridge inbound completion composer path 6_H **ADM_TX_PATH**: Region select for Adm Tx path 7_H **L3_TX_PATH**: Region select for layer 3 Tx path 8_H **L2_TX_PATH**: Region select for layer 2 Tx path 9_H **DMA_OUTBOUND_PATH**: Region select for DMA outbound path A_H **AXI_OUTBOUND_REQ_PATH**: Region select for AXI bridge outbound request path B_H **AXI_OUTBOUND_CMPL_PATH**: Region select for AXI bridge outbound master completion buffer path C_H **CXS_TX_RX_PATH**: Region select for CXS Tx/Rx path D_H **DTIM_TX_RX_PATH**: Region select for DTIM Tx/Rx path **others,** Reserved
LOC_LAST_UNCORR_ERROR	31:24	rh	**Location/ID of the last uncorrected error within the region defined by REG_LAST_UNCORR_ERROR** You can cycle this field value from 0 to 255 to access all counters according to the detailed report of check points at http://www.synopsys.com/dw/doc.php/iip/DWC_pcie/latest/doc/RASDP_CheckPoints.pdf Note: This register field is sticky.

19.4.2.202 RASDP error mode enable.

RASDP error mode enable. The controller enters RASDP error mode (if ERROR_MODE_EN =1) upon detection of the first uncorrectable error. During this mode: - Rx TLPs that are forwarded to your application are not guaranteed to be correct; you must discard them. For more details, see the RAS Data Protection (DP) section in the Core Operations chapter of the user manual.

RASDP_ERROR_MODE_EN_OFF Offset address: $00042A8_H$

RASDP error mode enable. Kernel Reset value: $0000\ 0001_H$



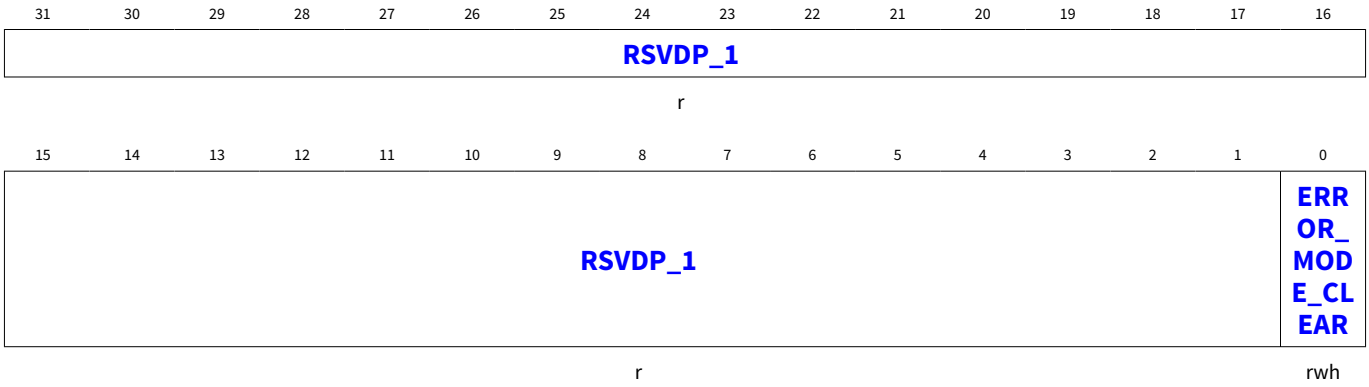
Field	Bits	Type	Description
ERROR_MODE_EN	0	rw	**Write '1' to enable the controller enter RASDP error mode when it detects an uncorrectable error** Note: This register field is sticky.
AUTO_LINK_DOWN_EN	1	rw	**Write '1' to enable the controller to bring the link down when the controller enters RASDP error mode** Note: This register field is sticky.
RSVDP_2	31:2	r	**Reserved for future use**

19.4.2.203 RASDP Error Mode Clear Register

Exit RASDP error mode. For more details, see the RAS Data Protection (DP) section in the Core Operations chapter of the user manual.

RASDP_ERROR_MODE_CLEAR_OFF Offset address: $00042AC_H$

RASDP Error Mode Clear Register Kernel Reset value: $0000\ 0000_H$



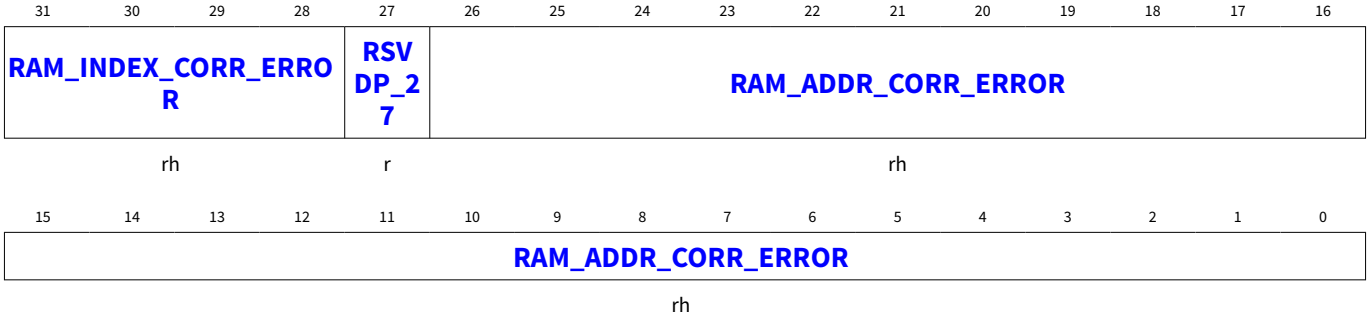
Field	Bits	Type	Description
ERROR_MODE_CLEAR	0	rwh	**Write '1' to take the controller out of RASDP error mode** The controller will then report uncorrectable errors (through AER internal error reporting) and also stop nullifying/discarding TLPs. Reading register field always returns a '0'.
RSVDP_1	31:1	r	**Reserved for future use**

19.4.2.204 RAM Address where a corrected error (1-bit ECC) is detected

For more details, see the RAS Data Protection (DP) section in the Core Operations chapter of the user manual.

RASDP_RAM_ADDR_CORR_ERROR_OFF Offset address: $00042B0_H$

RAM Address where a corrected error [1-bit ECC) is detected Kernel Reset value: $0000\ 0000_H$



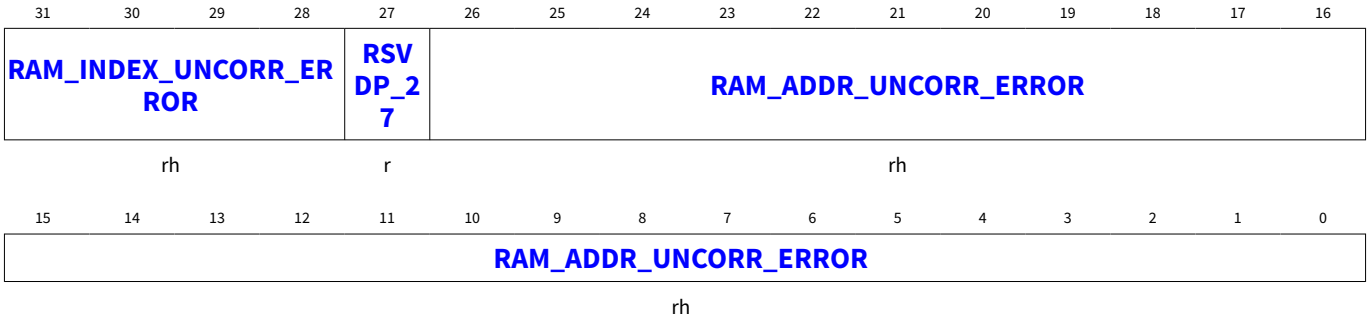
Field	Bits	Type	Description
RAM_ADDR_CORR_ERROR	26:0	rh	**RAM Address where a corrected error (1-bit ECC) is detected** Note: This register field is sticky.
RSVDP_27	27	r	**Reserved for future use**
RAM_INDEX_CORR_ERROR	31:28	rh	**RAM index where a corrected error (1-bit ECC) is detected** Note: This register field is sticky.

19.4.2.205 RAM Address where an uncorrected error (2-bit ECC) is detected

For more details, see the RAS Data Protection (DP) section in the Core Operations chapter of the user manual.

RASDP_RAM_ADDR_UNCORR_ERROR_OFF Offset address: $00042B4_H$

RAM Address where an uncorrected error [2-bit ECC) is detected Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
RAM_ADDR_UNCORR_ERROR	26:0	rh	**RAM Address where an uncorrected error (2-bit ECC) is detected** Note: This register field is sticky.
RSVDP_27	27	r	**Reserved for future use**

(table continues...)

(continued)

Field	Bits	Type	Description
RAM_INDEX_UNCORR_ERROR	31:28	rh	**RAM index where an uncorrected error (2-bit ECC) is detected** Note: This register field is sticky.

19.4.2.206 Precision Time Measurement Capability Header

This Register provides Capability ID, Cabability Version, and Next Offset of Precision Time Measurement Capability structure.

PTM_EXT_CAP_HDR_OFF Offset address: $00042B8_H$

Precision Time Measurement Capability Header Kernel Reset value: $2C41\ 001F_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PTM_NEXT_OFFSET												PTM_CAP_VERSION			
rw												rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PTM_CAP_ID															
rw															

Field	Bits	Type	Description
PTM_CAP_ID	15:0	rw	**Precision Time Measurement PCI Express Extended Capability ID** This field is a PCI-SIG defined ID number that indicates the nature and format of the Extended Capability. PCI Express Extended Capability ID for the Precision Time Measurement Capability is 001Fh. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky. 0000_H **MIN_VAL**: Min value $FFFF_H$ **MAX_VAL**: Max value
PTM_CAP_VERSION	19:16	rw	**Precision Time Measurement PCI Express Extended Capability Version** This field is a PCI-SIG defined version number that indicates the version of the Capability structure present. This bit depends on the version of the specification. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky. 0_H **MIN_VAL**: Min value F_H **MAX_VAL**: Max value
PTM_NEXT_OFFSET	31:20	rw	**Precision Time Measurement PCI Express Extended Capability Next Offset** This field contains the offset to the next PCI Express Extended Capability structure or 000h if no other items exist in the linked list of Capabilities. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky. 000_H **MIN_VAL**: Min value FFF_H **MAX_VAL**: Max value

19.4.2.207 PTM Capability Register

This register describes a Function's support for Precision Time Measurement. Not all fields within this register apply to all Functions capable of implementing PTM.

PTM_CAP_OFF Offset address: $00042BC_H$

PTM Capability Register Kernel Reset value: $0000\ 040F_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_16															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PTM_CLK_GRAN								RSVDP_4				EPTM_CAPABLE	PTM_ROOT_CAPABLE	PTM_RES_CAPABLE	PTM_REQ_CAPABLE
rwh								r				rw	rw	rwh	rw

Field	Bits	Type	Description
PTM_REQ_CAPABLE	0	rw	**PTM Requester Capable** Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky. 0_B **CLEAR**: Clear 1_B **SET**: Endpoints and RCiEPs are permitted to, and Switches supporting PTM must, set this bit to 1b to indicate they implement the PTM Requester role
PTM_RES_CAPABLE	1	rwh	**PTM Responder Capable** Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky. 0_B **CLEAR**: Clear 1_B **SET**: Root Ports and RCRB's are permitted to, and Switches supporting PTM must, set this bit to 1b to indicate they implement the PTM Responder role If PTM Root Capable is Set, this bit must be Set to 1b.
PTM_ROOT_CAPABLE	2	rw	**PTM Root Capable** Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky. 0_B **CLEAR**: Clear 1_B **SET**: Root Ports, RCRBs and Switches are permitted to set this bit to 1b, if they implement a PTM Time Source Role and are capable of serving as the PTM Root

(table continues...)

(continued)

Field	Bits	Type	Description
EPTM_CAPABLE	3	rw	**ePTM Capable** Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky. 0_B **CLEAR**: Clear 1_B **SET**: Indicates Enhanced Precision Time Management (ePTM) is supported Defaults to 0x1.
RSVDP_4	7:4	r	**Reserved for future use**
PTM_CLK_GRAN	15:8	rwh	**PTM Local Clock Granularity** For other than this value (between b00000001-b111111110) indicates the period of this Time Source's local clock in ns. If the PTM Root Select bit is Set, this local clock is used to provide PTM Master Time. Otherwise, the Time Source uses this local clock to locally track PTM Master Time received from further Upstream within a PTM Hierarchy. This field is reserved for Functions that do not implement the PTM Time Source role. Note: The access attributes of this field are as follows: - Wire: RSVDP - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) 00_H **No_local_clock**: Time Source does not implement a local clock It simply propagates timing information obtained from further Upstream in the PTM Hierarchy when responding to PTM Request messages. FF_H **Greater_than_254ns**: Indicates the period of this Time Source's local clock is greater than 254 ns
RSVDP_16	31:16	r	**Reserved for future use**

19.4.2.208 PTM Control Register

This register controls a Function's participation in the Precision Time Measurement mechanism. Not all fields within this register apply to all Functions capable of implementing PTM.

PTM_CONTROL_OFF Offset address: $00042C0_H$

PTM Control Register Kernel Reset value: $0000\ 0000_H$

Field	Bits	Type	Description
PTM_ENABLE	0	rw	**PTM Enable** 0_B **CLEAR**: Clear 1_B **SET**: When Set, this Function is permitted to participate in the PTM mechanism according to its selected role(s)
ROOT_SELECT	1	rw	**PTM Root Select** If the value of the PTM Root Capable bit is 0b, this bit is permitted to be hardwired to 0b by the controller. Within each PTM Hierarchy, it is recommended that system software select only the furthest Upstream Time Source to be the PTM Root. Note: The access attributes of this field are as follows: - Wire: HWINIT - Dbi: HWINIT 0_B **CLEAR**: Clear 1_B **SET**: When Set, if the PTM Enable bit is also Set, this Time Source is the PTM Root
RSVDP_2	7:2	r	**Reserved for future use**
EFF_GRAN	15:8	rh	**PTM Effective Granularity** For Functions implementing the PTM Requester Role, this field provides information relating to the expected accuracy of the PTM clock, but does not otherwise affect the PTM mechanism. For Endpoints, system software must program this field to the value representing the maximum Local Clock Granularity reported by the PTM Root and all intervening PTM Time Sources. For RCiEPs, system software must set this field to the value reported in the Local Clock Granularity field by the associated PTM Time Source. For other than this value (between b00000001-b111111110) indicates the effective PTM granularity in ns. Note: The access attributes of this field are as follows: - Wire: HWINIT - Dbi: HWINIT 00_H **No_local_clock**: Unknown PTM granularity - one or more Switches between this Function and the PTM Root reported a Local Clock Granularity value of 0000 0000b FF_H **Greater_than_254ns**: Indicates the effective PTM granularity is greater than 254 ns
RSVDP_16	31:16	r	**Reserved for future use**

19.4.2.209 Precision Time Measurement Responder Capability Header (VSEC)

For more details, see the PTM section in the Databook.

PTM_RES_CAP_HDR_OFF Offset address: $00042C4_H$

Precision Time Measurement Responder Capability Header [VSEC] Kernel Reset value: $32C1\ 000B_H$

Field	Bits	Type	Description
PTM_RES_EXT_CAP_ID	15:0	rw	**Precision Time Measurement Responder VSEC ID** For more details, see the PTM section in the Databook. Note: The access attributes of this field are as follows: - Wire: No access. - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky. 0000_H **MIN_VAL**: Min value $FFFF_H$ **MAX_VAL**: Max value
PTM_RES_EXT_CAP_VER	19:16	rw	**Precision Time Measurement Responder VSEC Version** For more details, see the PTM section in the Databook. Note: The access attributes of this field are as follows: - Wire: No access. - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky. 0_H **MIN_VAL**: Min value F_H **MAX_VAL**: Max value
PTM_RES_EXT_CAP_NEXT_OFFS	31:20	rw	**Precision Time Measurement Responder VSEC Next Pointer** For more details, see the PTM section in the Databook. Note: The access attributes of this field are as follows: - Wire: No access. - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky. 000_H **MIN_VAL**: Min value FFF_H **MAX_VAL**: Max value

19.4.2.210 Precision Time Measurement Responder Vendor Specific Header

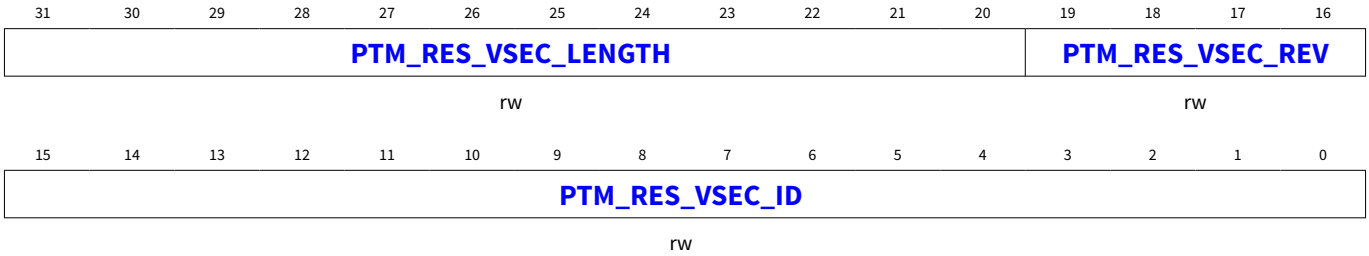
For more details, see the PTM section in the Databook.

PTM_RES_HDR_OFF

Precision Time Measurement Responder Vendor Specific Header

Offset address: $00042C8_H$

Kernel Reset value: $0541\ 0004_H$



Field	Bits	Type	Description
PTM_RES_VSEC_ID	15:0	rw	**PTM Responder VSEC ID** For more details, see the PTM section in the Databook. Note: The access attributes of this field are as follows: - Wire: No access. - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky. 0000_H **MIN_VAL**: Min value $FFFF_H$ **MAX_VAL**: Max value

(table continues...)

(continued)

Field	Bits	Type	Description
PTM_RES_VSEC_REV	19:16	rw	**PTM Responder VSEC Revision** For more details, see the PTM section in the Databook. Note: The access attributes of this field are as follows: - Wire: No access. - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky. 0_H **MIN_VAL**: Min value F_H **MAX_VAL**: Max value
PTM_RES_VSEC_LENGTH	31:20	rw	**PTM Responder VSEC Length** For more details, see the PTM section in the Databook. Note: The access attributes of this field are as follows: - Wire: No access. - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky. 000_H **MIN_VAL**: Min value FFF_H **MAX_VAL**: Max value

19.4.2.211 PTM Responder Control Register

For more details, see the PTM section in the Databook.

PTM_RES_CONTROL_OFF Offset address: $00042CC_H$

PTM Responder Control Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_2															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVDP_2														PTM_RES_PDEL_BY TE_REV	PTM_RES_CCONTEXT_VALID
r														rw	rwh

Field	Bits	Type	Description
PTM_RES_CCONTEXT_VALID	0	rwh	**PTM Responder Control Context Valid** PTM Local Timing is valid. This bit is set over the DBI. A speed change or aux_clk_active will set this bit low. For more details, see the PTM section in the Databook. Note: The access attributes of this field are as follows: - Wire: No access. - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky. 0_B **CLEAR**: Clear 1_B **SET**: PTM Local Timing is valid

(table continues...)

(continued)

Field	Bits	Type	Description
PTM_RES_PDEL_BYTE_REV	1	rw	**PTM Requester Propagation Delay Byte Reversal** For more details, see the PTM section in the Databook. Note: This register field is sticky. 0_B **CLEAR**: Clear 1_B **SET**: When set, reverse the order of bytes in the PTM Propagation Delay data word of the PTM ResponseD Message, for compatibility with previous revisions
RSVDP_2	31:2	r	**Reserved for future use**

19.4.2.212 PTM Responder Status Register

For more details, see the PTM section in the Databook.

PTM_RES_STATUS_OFF Offset address: $00042D0_H$

PTM Responder Status Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_2															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVDP_2														PTM_RES_REQUEST_RECEIVED	PTM_RES_CONTEXT_VALID
r														rh	rh

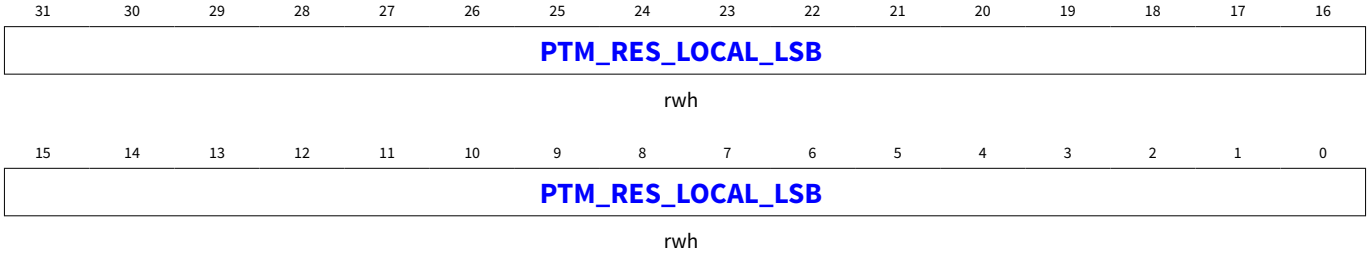
Field	Bits	Type	Description
PTM_RES_CONTEXT_VALID	0	rh	**PTM Responder Status Context Valid** PTM Local Timing Context is Valid. Value set from upstream port Requester in a Switch. Shadows the same in the control register in all other products. For more details, see the PTM section in the Databook. Note: This register field is sticky. 0_B **CLEAR**: Clear 1_B **SET**: Set
PTM_RES_REQUEST_RECEIVED	1	rh	**PTM 1st Request Received** PTM Responder has received the first PTM Request Message. Upon receipt of a second PTM Request Message a ResponseD message with timing information will be sent from the Responder, if the context is valid. If the context is invalid a Response message will be sent instead. For more details, see the PTM section in the Databook. Note: This register field is sticky. 0_B **CLEAR**: Clear 1_B **SET**: Set
RSVDP_2	31:2	r	**Reserved for future use**

19.4.2.213 PTM Responder Local Clock LSB

For more details, see the PTM section in the Databook.

PTM_RES_LOCAL_LSB_OFF Offset address: $00042D4_H$

PTM Responder Local Clock LSB Kernel Reset value: $0000\ 0000_H$



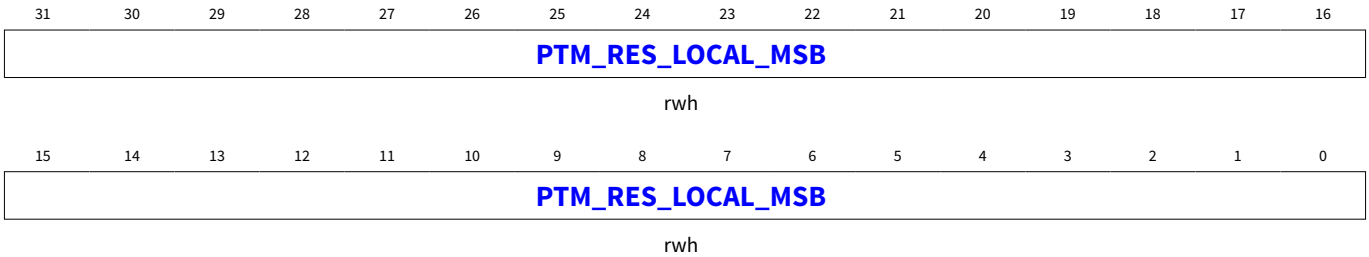
Field	Bits	Type	Description
PTM_RES_LOCAL_LSB	31:0	rwh	**PTM Responder Local Clock LSB** Lower 32 bits of local timer value. For more details, see the PTM section in the Databook. Note: The access attributes of this field are as follows: - Wire: No access. - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky. 00000000_H **MIN_VAL**: Min value $FFFFFFFF_H$ **MAX_VAL**: Max value

19.4.2.214 PTM Responder Local Clock MSB

For more details, see the PTM section in the Databook.

PTM_RES_LOCAL_MSB_OFF Offset address: $00042D8_H$

PTM Responder Local Clock MSB Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
PTM_RES_LOCAL_MSB	31:0	rwh	**PTM Responder Local Clock MSB** Upper 32 bits of local timer value. For more details, see the PTM section in the Databook. Note: The access attributes of this field are as follows: - Wire: No access. - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky. 00000000_H **MIN_VAL**: Min value $FFFFFFFF_H$ **MAX_VAL**: Max value

19.4.2.215 PTM Responder T2 Timestamp LSB

For more details, see the PTM section in the Databook.

PTM_RES_T2_LSB_OFF Offset address: 00042DC$_H$

PTM Responder T2 Timestamp LSB Kernel Reset value: 0000 0000$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PTM_RES_T2_LSB															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PTM_RES_T2_LSB															
rh															

Field	Bits	Type	Description
PTM_RES_T2_LSB	31:0	rh	**PTM Responder T2 Timestamp LSB** Lower 32 bits of the T2 Timestamp value. For more details, see the PTM section in the Databook. Note: This register field is sticky. 00000000$_H$ **MIN_VAL**: Min value FFFFFFFF$_H$ **MAX_VAL**: Max value

19.4.2.216 PTM Responder T2 Timestamp MSB

For more details, see the PTM section in the Databook.

PTM_RES_T2_MSB_OFF Offset address: 00042E0$_H$

PTM Responder T2 Timestamp MSB Kernel Reset value: 0000 0000$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PTM_RES_T2_MSB															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PTM_RES_T2_MSB															
rh															

Field	Bits	Type	Description
PTM_RES_T2_MSB	31:0	rh	**PTM Responder T2 Timestamp MSB** Upper 32 bits of the T2 Timestamp value. For more details, see the PTM section in the Databook. Note: This register field is sticky. 00000000$_H$ **MIN_VAL**: Min value FFFFFFFF$_H$ **MAX_VAL**: Max value

19.4.2.217 PTM Responder T2 Previous Timestamp LSB

For more details, see the PTM section in the Databook.

PTM_RES_T2P_LSB_OFF Offset address: 00042E4$_H$

PTM Responder T2 Previous Timestamp LSB Kernel Reset value: 0000 0000$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PTM_RES_T2P_LSB															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PTM_RES_T2P_LSB															
rh															

Field	Bits	Type	Description
PTM_RES_T2P_LSB	31:0	rh	**PTM Responder T2 Previous Timestamp LSB** Lower 32 bits of the previously stored T2 Timestamp value. For more details, see the PTM section in the Databook. Note: This register field is sticky. 00000000_H **MIN_VAL**: Min value $FFFFFFFF_H$ **MAX_VAL**: Max value

19.4.2.218 PTM Responder T2 Previous Timestamp MSB

For more details, see the PTM section in the Databook.

PTM_RES_T2P_MSB_OFF Offset address: $00042E8_H$

PTM Responder T2 Previous Timestamp MSB Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PTM_RES_T2P_MSB															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PTM_RES_T2P_MSB															
rh															

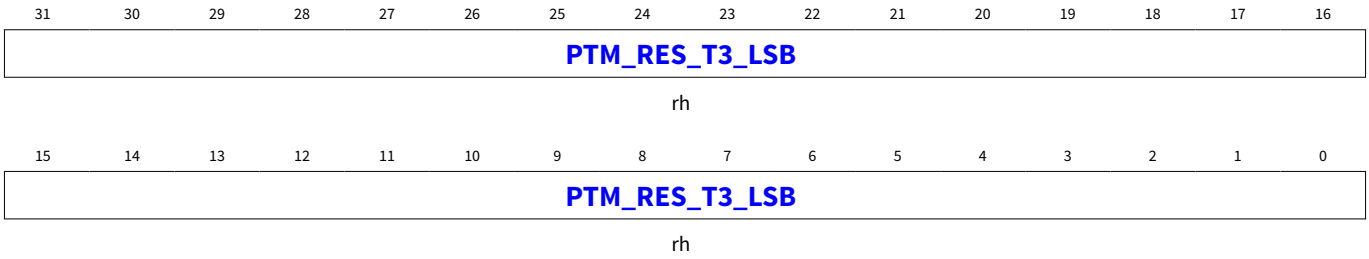
Field	Bits	Type	Description
PTM_RES_T2P_MSB	31:0	rh	**PTM Responder T2 Previous Timestamp MSB** Upper 32 bits of the previously stored T2 Timestamp value. For more details, see the PTM section in the Databook. Note: This register field is sticky. 00000000_H **MIN_VAL**: Min value $FFFFFFFF_H$ **MAX_VAL**: Max value

19.4.2.219 PTM Responder T3 Timestamp LSB

For more details, see the PTM section in the Databook.

PTM_RES_T3_LSB_OFF Offset address: $00042EC_H$

PTM Responder T3 Timestamp LSB Kernel Reset value: $0000\ 0000_H$



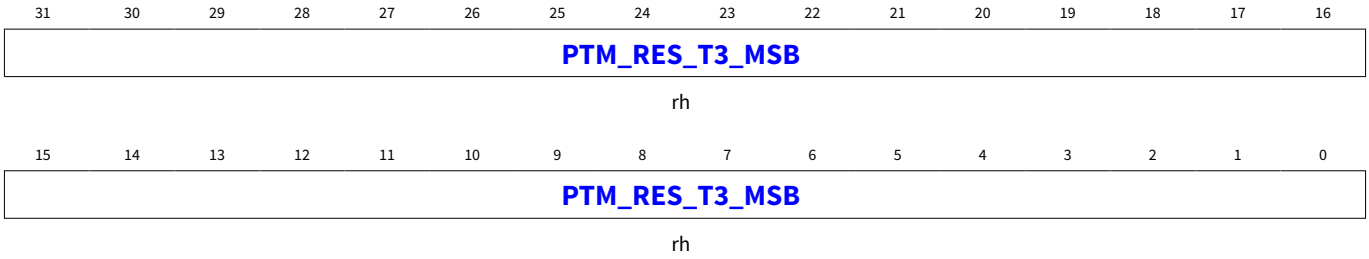
Field	Bits	Type	Description
PTM_RES_T3_LSB	31:0	rh	**PTM Responder T3 Timestamp LSB** Lower 32 bits of the T3 Timestamp value. For more details, see the PTM section in the Databook. Note: This register field is sticky. 00000000_H **MIN_VAL**: Min value $FFFFFFFF_H$ **MAX_VAL**: Max value

19.4.2.220 PTM Responder T3 Timestamp MSB

For more details, see the PTM section in the Databook.

PTM_RES_T3_MSB_OFF Offset address: $00042F0_H$

PTM Responder T3 Timestamp MSB Kernel Reset value: $0000\ 0000_H$



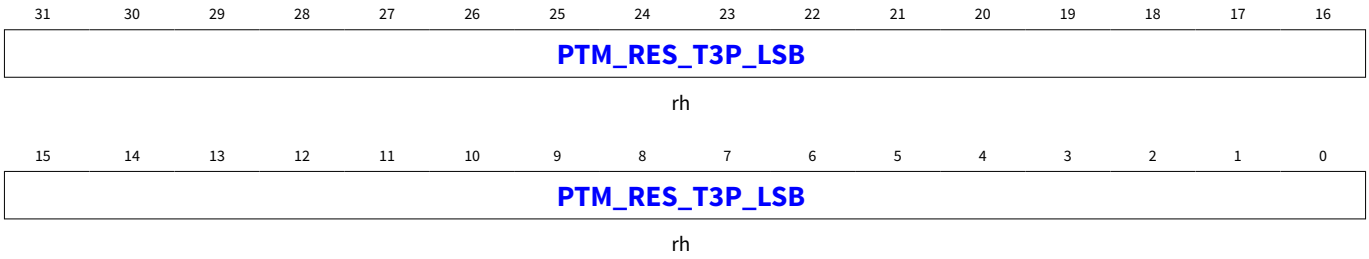
Field	Bits	Type	Description
PTM_RES_T3_MSB	31:0	rh	**PTM Responder T3 Timestamp MSB** Upper 32 bits of the T2 Timestamp value. For more details, see the PTM section in the Databook. Note: This register field is sticky. 00000000_H **MIN_VAL**: Min value $FFFFFFFF_H$ **MAX_VAL**: Max value

19.4.2.221 PTM Responder T3 Previous Timestamp LSB

For more details, see the PTM section in the Databook.

PTM_RES_T3P_LSB_OFF Offset address: $00042F4_H$

PTM Responder T3 Previous Timestamp LSB Kernel Reset value: $0000\ 0000_H$



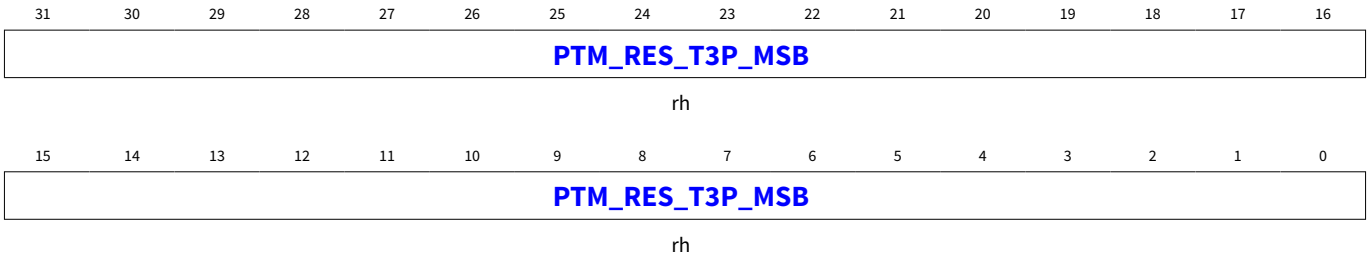
Field	Bits	Type	Description
PTM_RES_T3P_LSB	31:0	rh	**PTM Responder T3 Previous Timestamp LSB** Lower 32 bits of the previously stored T3 Timestamp value. For more details, see the PTM section in the Databook. Note: This register field is sticky. 00000000_H **MIN_VAL**: Min value $FFFFFFFF_H$ **MAX_VAL**: Max value

19.4.2.222 PTM Responder T3 Previous Timestamp MSB

For more details, see the PTM section in the Databook.

PTM_RES_T3P_MSB_OFF Offset address: $00042F8_H$

PTM Responder T3 Previous Timestamp MSB Kernel Reset value: $0000\ 0000_H$



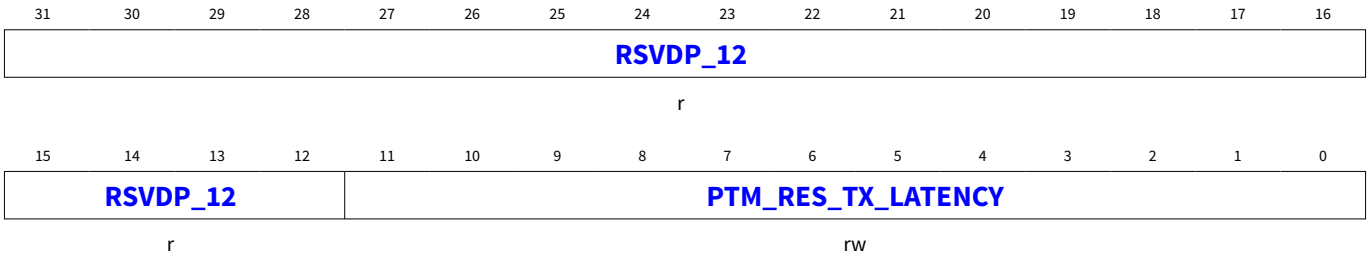
Field	Bits	Type	Description
PTM_RES_T3P_MSB	31:0	rh	**PTM Responder T3 Previous Timestamp MSB** Upper 32 bits of the previously stored T3 Timestamp value. For more details, see the PTM section in the Databook. Note: This register field is sticky. 00000000_H **MIN_VAL**: Min value $FFFFFFFF_H$ **MAX_VAL**: Max value

19.4.2.223 PTM Responder TX Latency

For more details, see the PTM section in the Databook.

PTM_RES_TX_LATENCY_OFF Offset address: $00042FC_H$

PTM Responder TX Latency Kernel Reset value: $0000\ 000E_H$



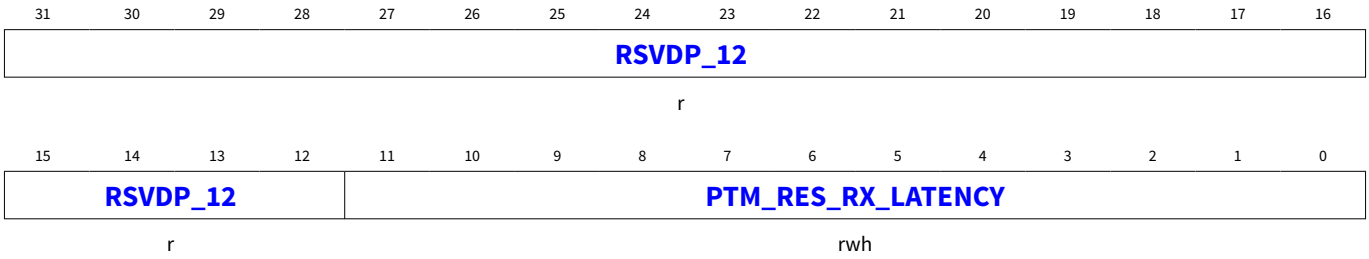
Field	Bits	Type	Description
PTM_RES_TX_LATENCY	11:0	rw	**PTM Responder TX Latency** Responder Transmit path latency viewport register. A register is provided for each supported link speed, and the value used for timestamp adjustment is automatically selected based on the current speed. The PTM Responder Latency Register Select register specifies the register to be written or read. For more details, see the PTM section in the Databook. Note: This register field is sticky. 000_H **MIN_VAL**: Min value FFF_H **MAX_VAL**: Max value
RSVDP_12	31:12	r	**Reserved for future use**

19.4.2.224 PTM Responder RX Latency

For more details, see the PTM section in the Databook.

PTM_RES_RX_LATENCY_OFF Offset address: 0004300_H

PTM Responder RX Latency Kernel Reset value: $0000\ 002A_H$



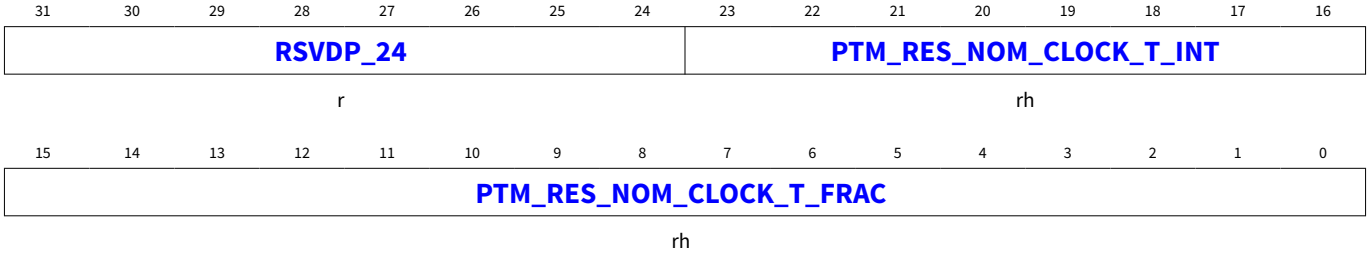
Field	Bits	Type	Description
PTM_RES_RX_LATENCY	11:0	rwh	**PTM Responder RX Latency** Responder Receive path latency viewport register. A register is provided for each supported link speed, and the value used for timestamp adjustment is automatically selected based on the current speed. The PTM Responder Latency Register Select register specifies the register to be written or read. For more details, see the PTM section in the Databook. Note: This register field is sticky. 000_H **MIN_VAL**: Min value FFF_H **MAX_VAL**: Max value
RSVDP_12	31:12	r	**Reserved for future use**

19.4.2.225 PTM Responder Nominal Clock Period

For more details, see the PTM section in the Databook.

PTM_RES_NOM_CLOCK_T_OFF Offset address: 0004304_H

PTM Responder Nominal Clock Period Kernel Reset value: $0000\ 0000_H$



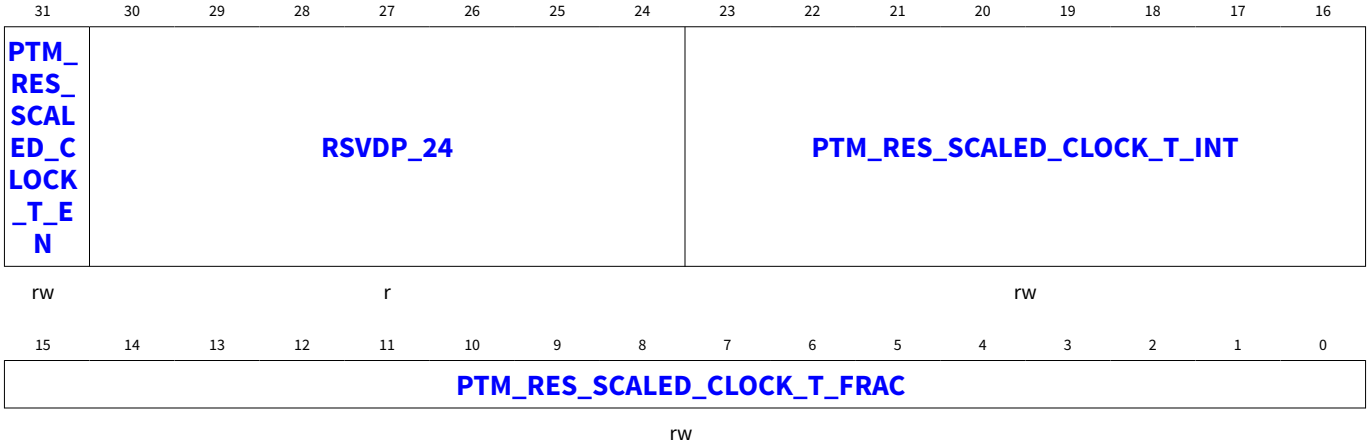
Field	Bits	Type	Description
PTM_RES_NOM_CLOCK_T_FRAC	15:0	rh	**PTM Responder Nominal Clock Period Fractional** Fractional part of the nominal PTM local clock period. LSB is 1/(2^16) nanoseconds. For more details, see the PTM section in the Databook. Note: This register field is sticky. 0000_H **MIN_VAL**: Min value $FFFF_H$ **MAX_VAL**: Max value
PTM_RES_NOM_CLOCK_T_INT	23:16	rh	**PTM Responder Nominal Clock Period Integral – Integral part of the nominal PTM local clock period, in nanoseconds** For more details, see the PTM section in the Databook. Note: This register field is sticky. 00_H **MIN_VAL**: Min value $0F_H$ **MAX_VAL**: Max value **others,** Reserved
RSVDP_24	31:24	r	**Reserved for future use**

19.4.2.226 PTM Responder Scaled Clock Period

For more details, see the PTM section in the Databook.

PTM_RES_SCALED_CLOCK_T_OFF Offset address: 0004308_H

PTM Responder Scaled Clock Period Kernel Reset value: $0000\ 0000_H$



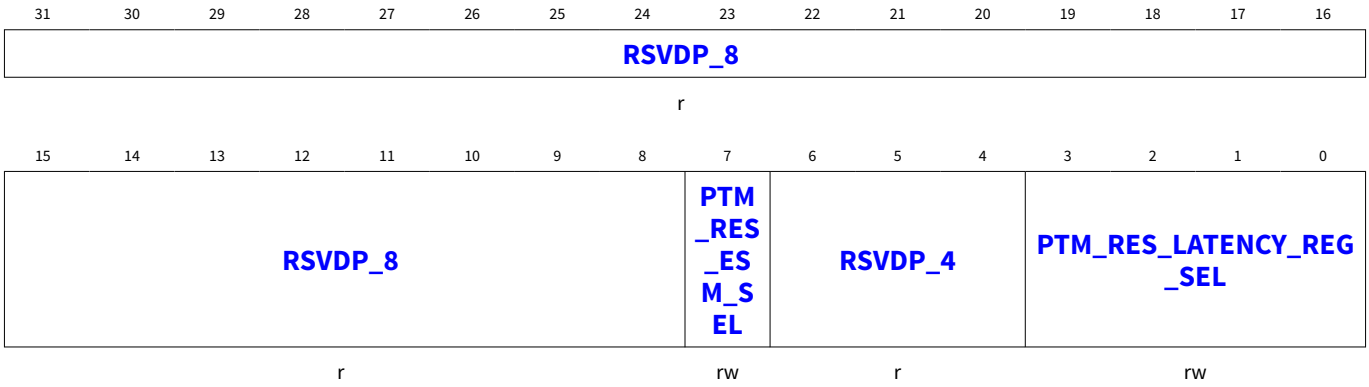
Field	Bits	Type	Description
PTM_RES_SCALED_CLOCK_T_FRAC	15:0	rw	**PTM Responder Scaled Clock Period Fractional** Fractional part of the nominal PTM local clock period. LSB is 1/(2^16) nanoseconds. For more details, see the PTM section in the Databook. Note: This register field is sticky. 0000_H **MIN_VAL**: Min value $FFFF_H$ **MAX_VAL**: Max value
PTM_RES_SCALED_CLOCK_T_INT	23:16	rw	**PTM Responder Scaled Clock Period Integral** Integral part of the nominal PTM local clock period, in nanoseconds. For more details, see the PTM section in the Databook. Note: This register field is sticky. 00_H **MIN_VAL**: Min value $0F_H$ **MAX_VAL**: Max value **others,** Reserved
RSVDP_24	30:24	r	**Reserved for future use**
PTM_RES_SCALED_CLOCK_T_EN	31	rw	**PTM Responder Scaled Clock Period Enable** Use the programmed scaled PTM clock period rather than the nominal values. This bit is cleared when the core_clk rate starts to change and can only be set when the clock period change is complete. For more details, see the PTM section in the Databook. Note: This register field is sticky. 0_B **DISABLE**: Disable 1_B **ENABLE**: Enable

19.4.2.227 PTM Responder Latency Viewport Register Select

For more details, see the PTM section in the Databook.

PTM_RES_LATENCY_REG_SEL_OFF Offset address: $000430C_H$

PTM Responder Latency Viewport Register Select Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
PTM_RES_LATENCY_REG_SEL	3:0	rw	**Selects the PTM Responder Tx and Rx Latency Viewport register to be written or read, together with the PTM_RES_ESM_SEL field** For more details, see the PTM section in the Databook. Note: This register field is sticky.
RSVDP_4	6:4	r	**Reserved for future use**
PTM_RES_ESM_SEL	7	rw	**Selection Mode for PTM Responder Tx and Rx Latency Viewport register to be written or read** For more details, see the PTM section in the Databook. Note: This register field is sticky. 0_B **SEL_PCIE_REGS**: write or read latency registers for PCIe rates 1_B **SEL_CCIX_ESM_REGS**: write or read latency registers for additional ESM rates
RSVDP_8	31:8	r	**Reserved for future use**

19.4.2.228 PCIe Extended Capability ID, Capability Version, and Next Capability Offset Register

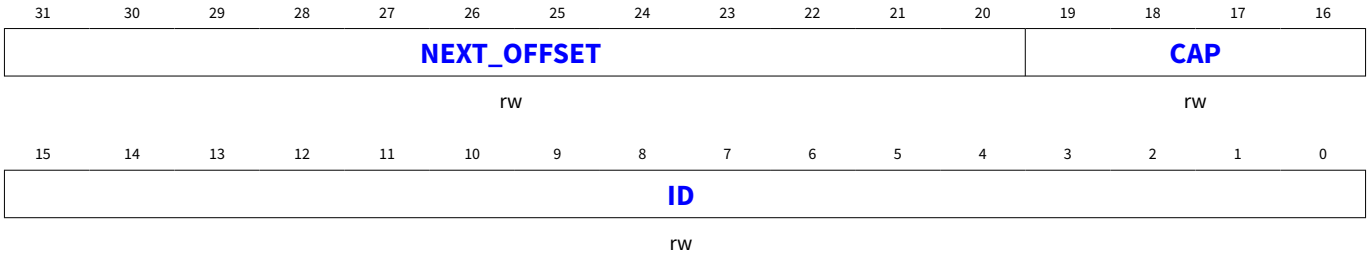
This register provides capability ID, capability version, and next capability offset for PCIe extended capability structure.

VSECDMA_EXT_CAP_HDR_OFF

PCIe Extended Capability ID, Capability Version, and Next Capability Offset Register

Offset address: $000432C_H$

Kernel Reset value: $0001\ 000B_H$



Field	Bits	Type	Description
ID	15:0	rw	**PCI Express Extended Capability ID** This field is a PCI-SIG defined ID number that indicates the nature and format of the Extended Capability. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky.
CAP	19:16	rw	**Capability Version** This field is a PCI-SIG defined version number that indicates the version of the Capability structure present. A version of the specification that changes the Extended Capability in a way that is not otherwise identifiable (that is, through a new Capability field) is permitted to increment this field. All such changes to the Capability structure must be software-compatible. Software must check for Capability Version numbers that are greater than or equal to the highest number defined when the software is written, as Functions reporting any such Capability Version numbers will contain a Capability structure that is compatible with that piece of software. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky.
NEXT_OFFSET	31:20	rw	**Next Capability Offset** This field contains the offset to the next PCI Express Capability structure or 000h if no other items exist in the linked list of Capabilities. For Extended Capabilities implemented in Configuration Space, this offset is relative to the beginning of PCI-compatible Configuration Space and thus must always be either 000h (for terminating list of Capabilities) or greater than 0FFh. The bottom 2 bits of this offset are Reserved and must be implemented as 00b although software must mask them to allow for future uses of these bits. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky.

19.4.2.229 Vendor Specific Header Register

This Register provides VSEC Length, VSEC ID, and VSEC Rev(Version Number). Vendor-specific software must qualify the associated Vendor ID of the PCI Express Function or RCRB before attempting to interpret the values in the VSEC ID or VSEC Rev fields.

VSECDMA_VENDOR_SPECIFIC_HDR_OFF Offset address: 0004330_H

Vendor Specific Header Register Kernel Reset value: $0180\ 0006_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
VSEC_LENGTH												VSEC_REV			
r												r			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VSEC_ID															
r															

Field	Bits	Type	Description
VSEC_ID	15:0	r	**VSEC ID** This field is a vendor-defined ID number that indicates the nature and format of the VSEC structure. Software must qualify the Vendor ID before interpreting this field.
VSEC_REV	19:16	r	**VSEC Rev** This field is a vendor-defined version number that indicates the version of the VSEC structure. Software must qualify the Vendor ID and VSEC ID before interpreting this field.
VSEC_LENGTH	31:20	r	**VSEC Length** This field indicates the number of bytes in the entire VSEC structure, including the Vendor-Specific Extended Capability Header, the Vendor-Specific Header, and the vendor-specific registers.

19.4.2.230 DMA and related AXI Bridge Implementation Information

This register provides DMA and AXI Bridge implementation-specific information: - DMA Architecture: Legacy DMA or Hyper DMA (HDMA). - Register Location: Port-logic or Mapped to a Function and BAR. - Register Map: Legacy DMA or HDMA. - Channel Separation: Address distance between read and write channels. - AXI Bridge: Used or Not Used - AXI Master Bus Specification: AXI Master Bus Width, Burst Length, and Boundary Pointer Width You must use this information along with the IP version registers: - PCIE_VERSION_NUMBER_OFF - PCIE_VERSION_TYPE_OFF

VSECDMA_DEVICE_INFORMATION_OFF Offset address: 0004334_H

DMA and related AXI Bridge Implementation Information Kernel Reset value: $3498\ 0001_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_30		MASTER_PAGE_BOUNDARY_POINTER_WIDTH				MASTER_BURST_LENGTH			MASTER_BUS_WIDTH			AXI	CHANNEL_SEPARATION		
r		r				r			r			r	r		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PFN					BARN			RSVDP_3					MAP_FORMAT		
r					r			r					r		

Field	Bits	Type	Description
MAP_FORMAT	2:0	r	**Defines the register map format and features to be one of the following values: Other values are reserved** 000_B **EDMA_LEGACY_PL**: Legacy DMA register map accessed by the port-logic registers 001_B **EDMA_LEGACY_UNROLL**: Legacy DMA register map, mapped to a PF/BAR 101_B **HDMA_COMPATIBILITY_MODE**: HDMA compatibility mode (CC_LEGACY_DMA_MAP =1) register map, mapped to a PF/BAR 111_B **HDMA_NATIVE**: HDMA native (CC_LEGACY_DMA_MAP =0) register map, mapped to a PF/BAR
RSVDP_3	7:3	r	**Reserved for future use**

(table continues...)

(continued)

Field	Bits	Type	Description
BARN	10:8	r	**Bar Number** This field provides information regarding the DMA register and BAR number mapping.
PFN	15:11	r	**Physical Function Number** This field provides information regarding the DMA register and physical function mapping.
CHANNEL_SEPARATION	18:16	r	**If the MAP_FORMAT is set to HDMA_NATIVE, this field specifies the read write channel address separation** Other values are reserved. 000_B **CHSEP_256**: 256 separated 001_B **CHSEP_512**: 512 separated 010_B **CHSEP_1K**: 1k separated 011_B **CHSEP_2K**: 2k separated 100_B **CHSEP_4K**: 4k separated 101_B **CHSEP_8K**: 8k separated 110_B **CHSEP_16K**: 16k separated 111_B **CHSEP_32K**: 32k separated
AXI	19	r	**This field provides information about AXI interface usage** It reports the value of AXI_POPULATED configuration parameter.
MASTER_BUS_WIDTH	22:20	r	**This field provides information regarding the AXI master data bus width** It reports the value of width of the AXI master databus. 000_B **MSTR_BUSWD_32**: 32 bits 001_B **MSTR_BUSWD_64**: 64 bits 010_B **MSTR_BUSWD_128**: 128 bits 011_B **MSTR_BUSWD_256**: 256 bits 100_B **MSTR_BUSWD_512**: 512 bits **others,** Reserved
MASTER_BURST_LENGTH	25:23	r	**Reports the maximum burst length (in beats) of an AXI transfer at the master interface** 000_B **MSTR_BRSTLEN_8**: 8 bits 001_B **MSTR_BRSTLEN_16**: 16 bits 010_B **MSTR_BRSTLEN_32**: 32 bits 011_B **MSTR_BRSTLEN_64**: 64 bits 100_B **MSTR_BRSTLEN_128**: 128 bits 101_B **MSTR_BRSTLEN_256**: 256 bits **others,** Reserved
MASTER_PAGE_BOUNDARY_POINTER_WIDTH	29:26	r	**This field provides address page boundary information** It reports the value of the page boundary of 128, 256, 512, 1K, 2K or 4K (bytes). .
RSVDP_30	31:30	r	**Reserved for future use**

19.4.2.231 Number of Implemented Channels Register

This register specifies the number of read and write channels implemented.

VSECDMA_NUM_CHAN_OFF Offset address: 0004338_H

Number of Implemented Channels Register Kernel Reset value: $0004\ 0004_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_26						NUM_DMA_RD_CHAN									
r						r									

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVDP_10						NUM_DMA_WR_CHAN									
r						r									

Field	Bits	Type	Description
NUM_DMA_WR_CHAN	9:0	r	**This field provides information regarding the number of implemented write channels** It reports the value of the number of write channels implemented in the DMA logic.
RSVDP_10	15:10	r	**Reserved for future use**
NUM_DMA_RD_CHAN	25:16	r	**This field provides information regarding the number of implemented read channels** It reports the value of the number of read channels implemented in the DMA logic.
RSVDP_26	31:26	r	**Reserved for future use**

19.4.2.232 DMA Register Map Start Address Offset Low Register

This register specifies the lower 32 bits of the offset of the start address of the DMA register map.

VSECDMA_UNROLL_ADDR_OFFSET_LOW_OFF Offset address: $000433C_H$

DMA Register Map Start Address Offset Low Register Kernel Reset value: $0000\ 4000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UNROLL_ADDR_OFFSET_LOW															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UNROLL_ADDR_OFFSET_LOW															
r															

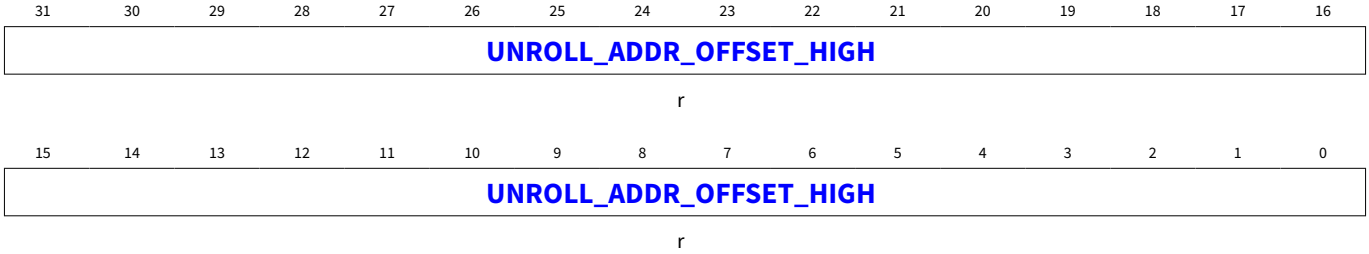
Field	Bits	Type	Description
UNROLL_ADDR_OFFSET_LOW	31:0	r	**BAR address offset, 32-bit LSB**

19.4.2.233 DMA Register Map Start Address Offset High Register

This register specifies the upper 32 bits of the offset of the start address of the DMA register map.

VSECDMA_UNROLL_ADDR_OFFSET_HIGH_OFF Offset address: 0004340_H

DMA Register Map Start Address Offset High Register Kernel Reset value: $0000\ 0000_H$



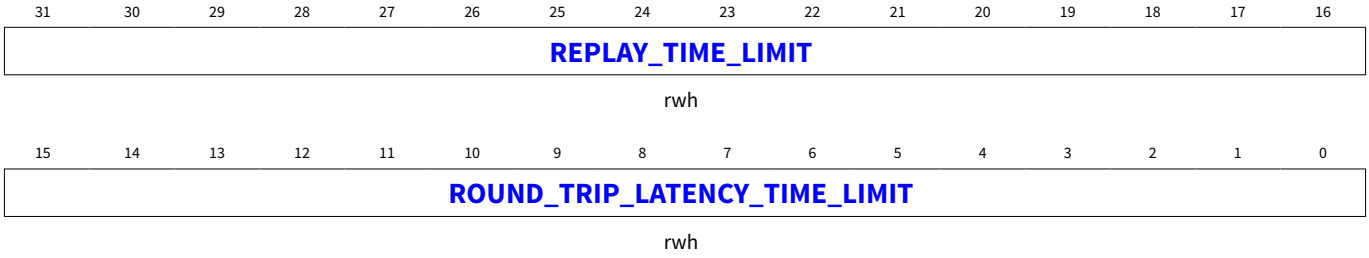
Field	Bits	Type	Description
UNROLL_ADDR_OFFSET_HIGH	31:0	r	**BAR address offset, 32-bit MSB**

19.4.2.234 Ack Latency Timer and Replay Timer Register

This register holds the ack latency timer limit and replay timer limit values.

ACK_LATENCY_TIMER_OFF Offset address: 0004700_H

Ack Latency Timer and Replay Timer Register Kernel Reset value: $0C23\ 040B_H$



Field	Bits	Type	Description
ROUND_TRIP_LATENCY_TIME_LIMIT	15:0	rwh	**Ack Latency Timer Limit** The Ack latency timer expires when it reaches this limit. - You can modify the effective timer limit through the TIMER_MOD_ACK_NAK field of the TIMER_CTRL_MAX_FUNC_NUM_OFF register. - After reset, the controller updates the default according to the Negotiated Link Width, Max_Payload_Size, and speed. The value is determined from Tables 3-7, 3-8, and 3-9 of the PCI Express Base Specification. - The limit must reflect the round trip latency from requester to completer. - If there is a change in the payload size or link width, the controller overrides any value that you have written to this register field, and resets the field back to the specification-defined value. The controller does not change the value in the TIMER_MOD_ACK_NAK field of the TIMER_CTRL_MAX_FUNC_NUM_OFF register.

(table continues...)

(continued)

Field	Bits	Type	Description
REPLAY_TIME_LIMIT	31:16	rwh	**Replay Timer Limit** The replay timer expires when it reaches this limit. The controller initiates a replay upon reception of a NAK or when the replay timer expires. - You can modify the effective timer limit through the TIMER_MOD_REPLAY_TIMER field of the TIMER_CTRL_MAX_FUNC_NUM_OFF register. - After reset, the controller updates the default according to the Negotiated Link Width, Max_Payload_Size, and speed. The value is determined from Tables 3-4, 3-5, and 3-6 of the PCI Express Base Specification. - If there is a change in the payload size or link speed, the controller overrides any value that you have written to this register field, and resets the field back to the specification-defined value. The controller does not change the value in the TIMER_MOD_REPLAY_TIMER field of the TIMER_CTRL_MAX_FUNC_NUM_OFF register.

19.4.2.235 Vendor Specific DLLP Register

This register holds the vendor specific DLLP.

VENDOR_SPEC_DLLP_OFF Offset address: 0004704_H

Vendor Specific DLLP Register Kernel Reset value: $FFFF\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
VENDOR_SPEC_DLLP															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VENDOR_SPEC_DLLP															
rw															

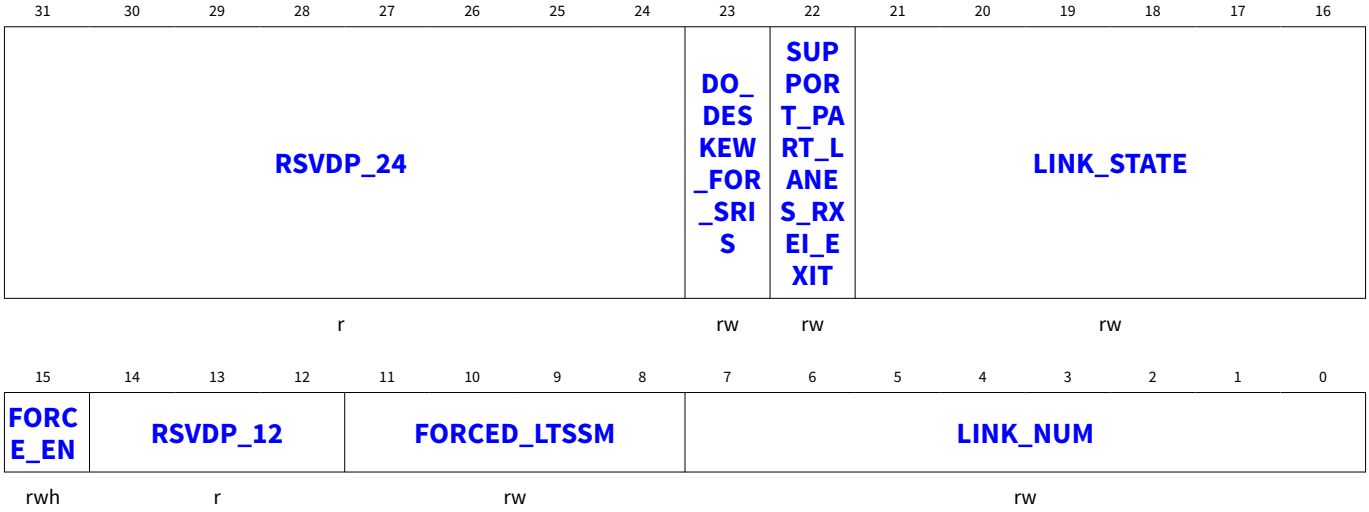
Field	Bits	Type	Description
VENDOR_SPEC_DLLP	31:0	rw	**Vendor Specific DLLP Register** You can use this register to send a specific PCI Express DLLP. Your application can write 8-bit DLLP Type and 24-bit Payload data into this register, and set the VENDOR_SPECIFIC_DLLP_REQ field of the PORT_LINK_CTRL_OFF, to send the DLLP. - Bits[7:0]: DLLP Type - Bits[31:8]: Vendor Defined Payload (24 bits) Note: This register field is sticky. 00000000_H **ZERO**: Zero value $FFFFFFFF_H$ **MAX**: Max value

19.4.2.236 Port Force Link Register

This register can be used for testing and debugging the link.

PORT_FORCE_OFF Offset address: 0004708_H

Port Force Link Register Kernel Reset value: $0080\ 0001_H$



Field	Bits	Type	Description
LINK_NUM	7:0	rw	**Link Number** Not used for endpoint. Note: This register field is sticky.
FORCED_LTSSM	11:8	rw	**Forced Link Command** The link command that the controller is forced to transmit when you set FORCE_EN bit (Force Link). Link command encoding is defined by the ltssm_cmd variable in workspace/src/Layer1/smlh_ltssm.v. Note: This register field is sticky.
RSVDP_12	14:12	r	**Reserved for future use**
FORCE_EN	15	rwh	**Force Link** The controller supports a testing and debug capability to allow your software to force the LTSSM state machine into a specific state, and to force the controller to transmit a specific Link Command. Asserting this bit triggers the following actions: • Forces the LTSSM to the state specified by the Forced LTSSM State field • Forces the controller to transmit the command specified by the Forced Link Command field This is a self-clearing register field. Reading from this register field always returns a '0'. 0_B **DISABLE**: Disable 1_B **ENABLE**: Enable
LINK_STATE	21:16	rw	**Forced LTSSM State** The LTSSM state that the controller is forced to when you set the FORCE_EN bit (Force Link). . Note: This register field is sticky.

(table continues...)

(continued)

Field	Bits	Type	Description
SUPPORT_PART_LANES_RXEI_EXIT	22	rw	**Support LTSSM transition from Polling.Active to Polling.Config based on Rx 8 TSs on any lanes which are Rx EI exit too from base spec after 24ms timeout** This prevents some lanes detected but not Rx EI exit and LTSSM cannot move to Polling.Config. Note: This register field is sticky. 0_B **UNSUPPORT**: Any lanes receives 8 consecutive TS OS and all predetermined lanes have Rx ElecIdle exit, LTSSM moves from Polling.Active to Polling.Config This is legacy mode from Base Spec. Any predetermined lanes are still on Rx ElecIdle, LTSSM moves from Polling.Active to Polling.Compliance. 1_B **SUPPORT**: Any lanes receives 8 consecutive TS OSs, LTSSM moves from Polling.Active to Polling.Config If all lanes do not receive 8 consecutive TS OSs and any predetermined lanes are still on Rx ElecIdle, LTSSM moves from Polling.Active to Polling.Compliance.
DO_DESKEW_FOR_SRIS	23	rw	**Use the transitions from TS2 to Logical Idle Symbol, SKP OS to Logical Idle Symbol, EIEOS to Logical Idle Symbol, and FTS Sequence to SKP OS to do deskew instead of using received SKP OS or TS1 to TS2 transition if DO_DESKEW_FOR_SRIS is set to '1'** Note: This register field is sticky.
RSVDP_24	31:24	r	**Reserved for future use**

19.4.2.237 Ack Frequency and L0-L1 ASPM Control Register

This register is used to control ack frequency and L0-L1 ASPM behaviour.

ACK_F_ASPM_CTRL_OFF Offset address: $000470C_H$

Ack Frequency and L0-L1 ASPM Control Register Kernel Reset value: $09B4\ B400_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_31	ENTER_ASPM	L1_ENTRANCE_LATENCY			L0S_ENTRANCE_LATENCY			COMMON_CLK_N_FTS							
r	rw	rw			rw			r							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ACK_N_FTS								ACK_FREQ							
rw								rw							

Field	Bits	Type	Description
ACK_FREQ	7:0	rw	**Ack Frequency** The controller accumulates the number of pending ACKs specified here (up to 255) before scheduling an ACK DLLP • 0: Indicates that this Ack Frequency Counter feature is turned off. The controller generates a low-priority ACK request for every TLP that it receives. The controller waits until the ACK Latency Timer expires, then converts the current low-priority ACK request to a high-priority ACK request and schedules the DLLP for transmission to the remote link partner • 1-255: Indicates that the controller will schedule a high-priority ACK after receiving this number of TLPs. It might schedule the ACK before receiving this number of TLPs if the ACK Latency Timer expires, but never later For a typical system, you do not have to modify the default setting. For more details, see "ACK/NAK Scheduling" in the user manual. Note: This register field is sticky. 00_H **MIN_VAL**: The value '0' indicates that this Ack Frequency Counter feature is turned off FF_H **MAX_VAL**: Any value between 1 and 255 indicates that the controller will schedule a high-priority ACK after receiving the specified number of TLPs
ACK_N_FTS	15:8	rw	**The number of Fast Training Sequence(N_FTS) ordered sets to be transmitted when transitioning from L0s to L0** Note: This register field is sticky. 00_H **ZERO**: The controller does not support a value of zero; a value of zero can cause the LTSSM to go into the recovery state when exiting from L0s FF_H **MAX_VAL**: The maximum number of FTS ordered-sets that a component can request is 255
COMMON_CLK_N_FTS	23:16	r	**Common Clock N_FTS** This is the N_FTS when common clock is used. The number of Fast Training Sequence ordered sets to be transmitted when transitioning from L0s to L0. Note: The access attributes of this field are as follows: - Wire: R - Dbi: R 00_H **ZERO**: The controller does not support a value of zero; a value of zero can cause the LTSSM to go into the recovery state when exiting from L0s FF_H **MAX_VAL**: The maximum number of FTS ordered-sets that a component can request is 255

(table continues...)

(continued)

Field	Bits	Type	Description
L0S_ENTRANCE_LATENCY	26:24	rw	**L0s Entrance Latency** Note: This register field is sticky. 000_B **_1_US**: 1 µs 001_B **_2_US**: 2 µs 010_B **_3_US**: 3 µs 011_B **_4_US**: 4 µs 100_B **_5_US**: 5 µs 101_B **_6_US**: 6 µs 110_B **_7_US**: 7 µs 111_B **_7US_**: 7 µs
L1_ENTRANCE_LATENCY	29:27	rw	**L1 Entrance Latency** Note: Programming this timer with a value greater that 32 µs has no effect unless extended sync is used, or all of the credits are infinite. Note: This register field is sticky. 000_B **_1_US**: 1 µs 001_B **_2_US**: 2 µs 010_B **_4_US**: 4 µs 011_B **_8_US**: 8 µs 100_B **_16_US**: 16 µs 101_B **_32_US**: 32 µs 110_B **_64_US**: 64 µs 111_B **_64US_**: 64 µs
ENTER_ASPM	30	rw	**ASPM L1 Entry Control** Note: This register field is sticky. 0_B **L0S_TO_L1**: Controller enters ASPM L1 only after idle period during which both receive and transmit are in L0s 1_B **IDLE_TO_L1**: Controller enters ASPM L1 after a period in which it has been idle
RSVDP_31	31	r	**Reserved for future use**

19.4.2.238 Port Link Control Register

Using this register you can control the port link behaviour.

PORT_LINK_CTRL_OFF Offset address: 0004710_H

Port Link Control Register Kernel Reset value: $0001\ 0120_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_28				TRANSMIT_LANE_REVERSALE_ENABLE	EXTENDED_SYNCH	CORRUPT_LCRC_ENABLE	BEACON_ENABLE	0		LINK_CAPABLE					
r				rw	rw	rw	rw	r		rw					

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVDP_12				LINK_RATE				Res	LINK_DISABLE	DLL_LINK_EN	RSVDP_4	RESET_ASSERT	LOOPBACK_ENABLE	SCRAMBLE_DISABLE	VENDOR_SPECIFIC_DLLP_REQ
r				rw				r	rw	rw	r	rw	rw	rw	rw1ch

Field	Bits	Type	Description
VENDOR_SPECIFIC_DLLP_REQ	0	rw1ch	**Vendor Specific DLLP Request** Reading from this self-clearing register field always returns a '0'. Software write with 1 clears this bit-field. 0_B **CLEAR**: This is a self clearing register 1_B **SET**: When software writes a '1' to this bit, the controller transmits the DLLP contained in the VENDOR_SPEC_DLLP field of VENDOR_SPEC_DLLP_OFF
SCRAMBLE_DISABLE	1	rw	**Scramble Disable** Turns off data scrambling. Note: The access attributes of this field are as follows: - Wire: R/W (sticky) - Dbi: R/W (sticky) Note: This register field is sticky. 0_B **DISABLE**: Disable 1_B **ENABLE**: Enable
LOOPBACK_ENABLE	2	rw	**Loopback Enable** Turns on loopback. For more details, see "Loopback" in the user manual. Note: The access attributes of this field are as follows: - Wire: R/W (sticky) - Dbi: R/W (sticky) Note: This register field is sticky. 0_B **DISABLE**: Disable 1_B **ENABLE**: Enable
RESET_ASSERT	3	rw	**Reset Assert** Triggers a recovery and forces the LTSSM to the hot reset state (downstream port only). Note: This register field is sticky. 0_B **CLEAR**: Clear 1_B **SET**: Set
RSVDP_4	4	r	**Reserved for future use**

(table continues...)

(continued)

Field	Bits	Type	Description
DLL_LINK_EN	5	rw	**DLL Link Enable** Note: This register field is sticky. 0_B **DISABLE**: The controller does not transmit InitFC DLLPs and does not establish a link 1_B **ENABLE**: Enables link initialization
LINK_DISABLE	6	rw	**LINK DISABLE is an internally reserved field** Do not use. this register field is sticky
LINK_RATE	11:8	rw	**LINK_RATE is an internally reserved field** Do not use. Note: This register field is sticky.
RSVDP_12	15:12	r	**Reserved for future use**
LINK_CAPABLE	21:16	rw	**Link Mode Enable** Sets the number of lanes in the link that you want to connect to the link partner.Note: The access attributes of this field are as follows: - Wire: R/W (sticky) - Dbi: R/W (sticky) Note: This register field is sticky. 01_H **X1**: x1 03_H **X2**: x2 07_H **X4**: x4 $0F_H$ **X8**: x8 $1F_H$ **X16**: x16 $3F_H$ **X32**: x32 (not supported)
BEACON_ENABLE	24	rw	**BEACON_ENABLE is an internally reserved field** Do not use. Note: This register field is sticky.
CORRUPT_LCRC_ENABLE	25	rw	**CORRUPT_LCRC_ENABLE is an internally reserved field** Do not use. Note: This register field is sticky.
EXTENDED_SYNCH	26	rw	**EXTENDED_SYNCH is an internally reserved field** Do not use. Note: This register field is sticky.
TRANSMIT_LANE_REVERSALE_ENABLE	27	rw	**TRANSMIT_LANE_REVERSALE_ENABLE is an internally reserved field** Do not use. Note: This register field is sticky.
RSVDP_28	31:28	r	**Reserved for future use**
0	23:22	r	**Reserved** Read as all 0's; should be written with all 0's.

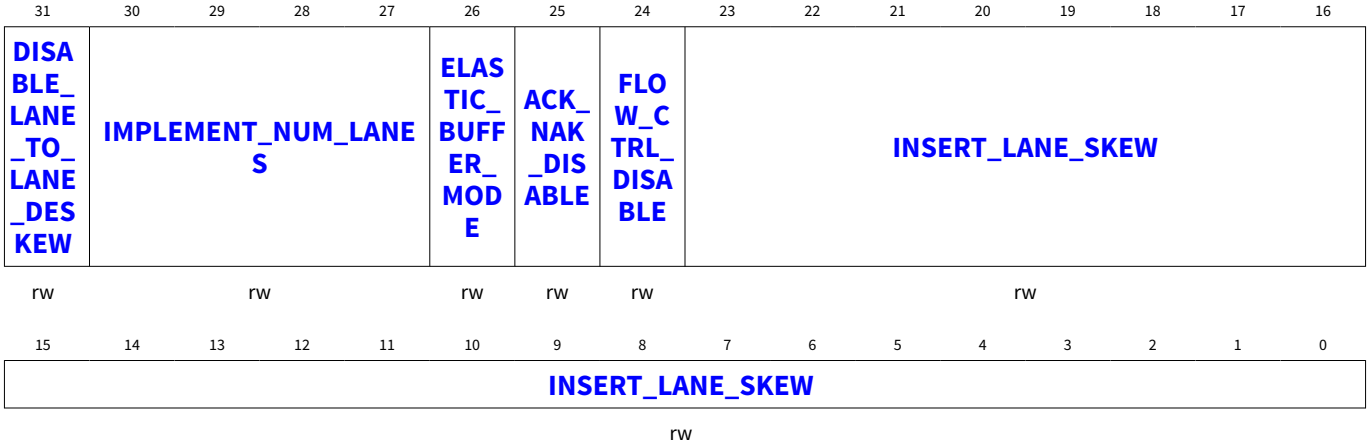
19.4.2.239 Lane Skew Register

This register is used to control the lane skew behaviour.

LANE_SKEW_OFF Offset address: 0004714_H

Lane Skew Register Kernel Reset value: $0400\ 0000_H$

19 Peripheral Component Interconnect Express (PCIe)



Field	Bits	Type	Description
INSERT_LANE_SKEW	23:0	rw	**INSERT_LANE_SKEW is an internally reserved field** Do not use. Note: This register field is sticky. 000000_H **MIN**: Zero value $0000FF_H$ **MAX**: Max value **others,** Reserved
FLOW_CTRL_DISABLE	24	rw	**Flow Control Disable** Prevents the controller from sending FC DLLPs. Note: This register field is sticky. 0_B **DISABLE**: Disable 1_B **ENABLE**: Enable
ACK_NAK_DISABLE	25	rw	**Ack/Nak Disable** Prevents the controller from sending ACK and NAK DLLPs. Note: This register field is sticky. 0_B **DISABLE**: Disable 1_B **ENABLE**: Enable
ELASTIC_BUFFER_MODE	26	rw	**Selects Elasticity Buffer operating mode:** Note: The access attributes of this field are as follows: - Wire: R/W (sticky) - Dbi: R/W (sticky) Note: This register field is sticky. 0_B **HALF_FULL**: Nominal Half Full Buffer mode 1_B **EMPTY**: Nominal Empty Buffer Mode

(table continues...)

(continued)

Field	Bits	Type	Description
IMPLEMENT_NUM_LANES	30:27	rw	**Implementation-specific Number of Lanes** Set the implementation-specific number of lanes. The number of lanes to be used when in Loopback Master. The number of lanes programmed must be equal to or less than the valid number of lanes set in LINK_CAPABLE field. You must configure this field before initiating Loopback by writing in the LOOPBACK_ENABLE field. The controller will transition from Loopback.Entry to Loopback.Active after receiving two consecutive TS1 Ordered Sets with the Loopback bit asserted on the implementation specific number of lanes configured in this field. Note: This register field is sticky. 0_H **_1_LANE**: 1 lane 1_H **_2_LANE**: 2 lanes 3_H **_4_LANE**: 4 lanes 7_H **_8_LANE**: 8 lanes F_H **_16_LANE**: 16 lanes
DISABLE_LANE_TO_LANE_DESKEW	31	rw	**Disable Lane-to-Lane Deskew** Causes the controller to disable the internal Lane-to-Lane deskew logic. Note: The access attributes of this field are as follows: - Wire: R/W (sticky) - Dbi: R/W (sticky) Note: This register field is sticky. 0_B **DISABLE**: Disable 1_B **ENABLE**: Enable

19.4.2.240 Timer Control and Max Function Number Register

This register holds the ack frequency, latency, replay, and max function number values.

TIMER_CTRL_MAX_FUNC_NUM_OFF Offset address: 0004718_H

Timer Control and Max Function Number Register Reset values see: Table 893

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_31	Res		UPDATE_FREQ_TIMER					TIMER_MOD_ACK_NAK					TIMER_MOD_REPLAY_TIMER		
r	r		rw					rw					rwh		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TIMER_MOD_REPLAY_TIMER			RSVDP_8					MAX_FUNC_NUM							
rwh			r					rw							

Field	Bits	Type	Description
MAX_FUNC_NUM	7:0	rw	**Maximum function number that can be used in a request** Configuration requests targeted at function numbers above this value are returned with UR (unsupported request). Note: This register field is sticky. 00_H **MIN**: Zero value FF_H **MAX**: Max value
RSVDP_8	13:8	r	**Reserved for future use**
TIMER_MOD_REPLAY_TIMER	18:14	rwh	**Replay Timer Limit Modifier** Increases the time-out value for the replay timer in increments of 64 clock cycles at Gen1 or Gen2 speed, and in increments of 256 clock cycles at Gen3 speed. A value of '0' represents no modification to the timer limit. For more details, see the REPLAY_TIME_LIMIT field of the ACK_LATENCY_TIMER_OFF register. At Gen3 speed, the controller automatically changes the value of this field to DEFAULT_GEN3_REPLAY_ADJ. Note: This register field is sticky.
TIMER_MOD_ACK_NAK	23:19	rw	**Ack Latency Timer Modifier** Increases the timer value for the Ack latency timer in increments of 64 clock cycles. A value of '0' represents no modification to the timer value. For more details, see the ROUND_TRIP_LATENCY_TIME_LIMIT field of the ACK_LATENCY_TIMER_OFF register. Note: This register field is sticky.
UPDATE_FREQ_TIMER	28:24	rw	**UPDATE_FREQ_TIMER is an internally reserved field** Do not use. Note: This register field is sticky.
RSVDP_31	31	r	**Reserved for future use**

Table 893 Reset values of TIMER_CTRL_MAX_FUNC_NUM_OFF

Reset	Reset value	Note
Kernel Reset	0100 0000 0000 0XXX XX00 0000 0000 0000_B	

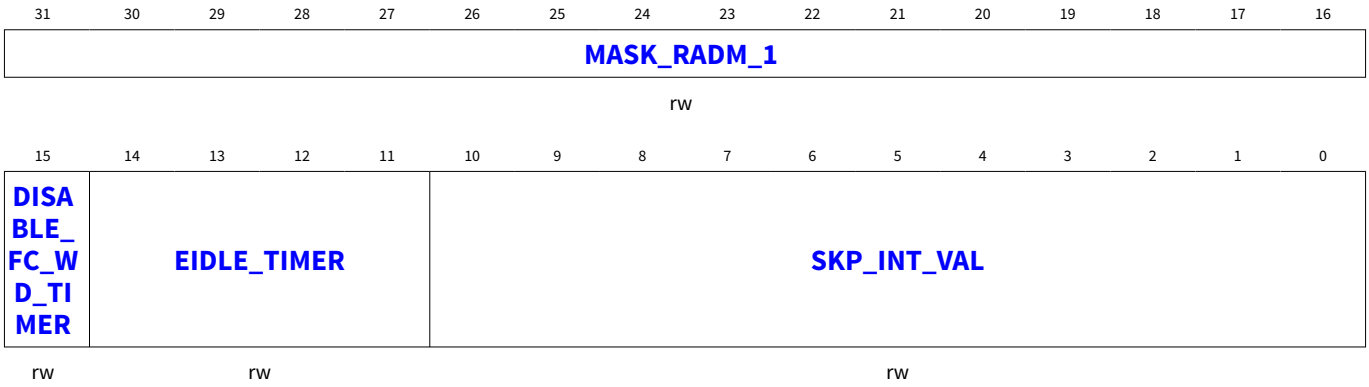
19.4.2.241 Symbol Timer Register and Filter Mask 1 Register

The Filter Mask 1 Register modifies the RADM filtering and error handling rules. For more details, see the "Receive Filtering" section. In each case, '0' applies the associated filtering rule and '1' masks the associated filtering rule.

SYMBOL_TIMER_FILTER_1_OFF Offset address: $000471C_H$

Symbol Timer Register and Filter Mask 1 Register Kernel Reset value: $0000\ 0140_H$

19 Peripheral Component Interconnect Express (PCIe)



Field	Bits	Type	Description
SKP_INT_VAL	10:0	rw	**SKP Interval Value** The number of symbol times to wait between transmitting SKP ordered sets. The controller waits the number of symbol times in this register plus 1, between transmitting SKP ordered sets. Your application must program this register accordingly. For example, if 384 were programmed into this register (62.5 MHz for GEN 1, 4 symbols/cycle), then the controller actually transmits SKP ordered sets once every 1537 (384*4+1) symbol times. The value programmed to this register is actually clock ticks and not symbol times. Note: This value is not used at Gen3 speed; the skip interval is hardcoded to 370 blocks. Note: This register field is sticky.
EIDLE_TIMER	14:11	rw	**EIDLE_TIMER is an internally reserved field** Do not use. Note: This register field is sticky.
DISABLE_FC_WD_TIMER	15	rw	**Disable FC Watchdog Timer** Note: This register field is sticky. 0_B **DISABLE**: Disable 1_B **ENABLE**: Enable

(table continues...)

(continued)

Field	Bits	Type	Description
MASK_RADM_1	31:16	rw	**Filter Mask 1** The Filter Mask 1 Register modifies the RADM filtering and error handling rules. For more details, see the "Receive Filtering" section. In each case, '0' applies the associated filtering rule and '1' masks the associated filtering rule. • [31]: CX_FLT_MASK_RC_CFG_DISCARD - 0: For RADM RC filter to not allow CFG transaction being received - 1: For RADM RC filter to allow CFG transaction being received • [30]: CX_FLT_MASK_RC_IO_DISCARD - 0: For RADM RC filter to not allow IO transaction being received - 1: For RADM RC filter to allow IO transaction being received • [29]: CX_FLT_MASK_MSG_DROP - 0: Drop MSG TLP (except for Vendor MSG). Send decoded message on the SII - 1: Do not Drop MSG (except for Vendor MSG). Send message TLPs to your application on TRGT1 and send decoded message on the SII - The default for this bit is the inverse of FLT_DROP_MSG. That is, if FLT_DROP_MSG =1, then the default of this bit is '0' (drop message TLPs). This bit only controls message TLPs other than Vendor MSGs. Vendor MSGs are controlled by Filter Mask Register 2, bits [1:0]. The controller never passes ATS Invalidate messages to the SII interface regardless of this filter rule setting. The controller passes all ATS Invalidate messages to TRGT1 (or AXI bridge master), as they are too big for the SII • [28]: CX_FLT_MASK_CPL_ECRC_DISCARD - Only used when completion queue is advertised with infinite credits and is in store-and-forward mode. - 0: Discard completions with ECRC errors - 1: Allow completions with ECRC errors to be passed up - Reserved field for SW • [27]: CX_FLT_MASK_ECRC_DISCARD - 0: Discard TLPs with ECRC errors - 1: Allow TLPs with ECRC errors to be passed up • [26]: CX_FLT_MASK_CPL_LEN_MATCH - 0: Enforce length match for completions; a violation results in cpl_abort, and possibly AER of unexp_cpl_err - 1: Mask length match for completions • [25]: CX_FLT_MASK_CPL_ATTR_MATCH - 0: Enforce attribute match for completions; a violation results in a malformed TLP error, and possibly AER of unexp_cpl_err, cpl_rcvd_ur, cpl_rcvd_ca - 1: Mask attribute match for completions • [24]: CX_FLT_MASK_CPL_TC_MATCH - 0: Enforce Traffic Class match for completions; a violation results in a malformed TLP error, and possibly AER of unexp_cpl_err, cpl_rcvd_ur, cpl_rcvd_ca - 1: Mask Traffic Class match for completions • [23]: CX_FLT_MASK_CPL_FUNC_MATCH - 0: Enforce function match for completions; a violation results in cpl_abort, and possibly AER of unexp_cpl_err, cpl_rcvd_ur, cpl_rcvd_ca - 1: Mask function match for completions • [22]: CX_FLT_MASK_CPL_REQID_MATCH - 0: Enforce Req. Id match for completions; a violation result in cpl_abort, and possibly AER of

(table continues...)

(continued)

Field	Bits	Type	Description
			unexp_cpl_err, cpl_rcvd_ur, cpl_rcvd_ca - 1: Mask Req. Id match for completions • [21]: CX_FLT_MASK_CPL_TAGERR_MATCH - 0: Enforce Tag Error Rules for completions; a violation result in cpl_abort, and possibly AER of unexp_cpl_err, cpl_rcvd_ur, cpl_rcvd_ca - 1: Mask Tag Error Rules for completions • [20]: CX_FLT_MASK_LOCKED_RD_AS_UR - 0: Treat locked Read TLPs as UR for EP; Supported for RC - 1: Treat locked Read TLPs as Supported for EP; UR for RC • [19]: CX_FLT_MASK_CFG_TYPE1_REQ_AS_UR - 0: Treat CFG type1 TLPs as UR for EP; Supported for RC - 1: Treat CFG type1 TLPs as Supported for EP; UR for RC • [18]: CX_FLT_MASK_UR_OUTSIDE_BAR - 0: Treat out-of-bar TLPs as UR - 1: Do not treat out-of-bar TLPs as UR • [17]: CX_FLT_MASK_UR_POIS - 0: Treat poisoned request TLPs as UR - 1: Do not treat poisoned request TLPs as UR - The native controller always passes poisoned completions to your application except when you are using the DMA read channel • [16]: CX_FLT_MASK_UR_FUNC_MISMATCH - 0: Treat Function MisMatched TLPs as UR - 1: Do not treat Function MisMatched TLPs as UR. CX_FLT_MASK_UR_FUNC_MISMATCH is not allowed to be set unless CONFIG_LIMIT_REG =0 and TARGET_ABOVE_CONFIG_LIMIT_REG =2 in MISC_CONTROL_1_OFF register or CFG_TLP_BYPASS_EN_REG =1 and TARGET_ABOVE_CONFIG_LIMIT_REG =2 in MISC_CONTROL_1_OFF register Note: This register field is sticky.

19.4.2.242 Filter Mask 2 Register

This register modifies the RADM filtering and error handling rules. For more details, see the "Receive Filtering" section. In each case, '0' applies the associated filtering rule and '1' masks the associated filtering rule.

FILTER_MASK_2_OFF Offset address: 0004720_H

Filter Mask 2 Register Kernel Reset value: $0000\ 0000_H$

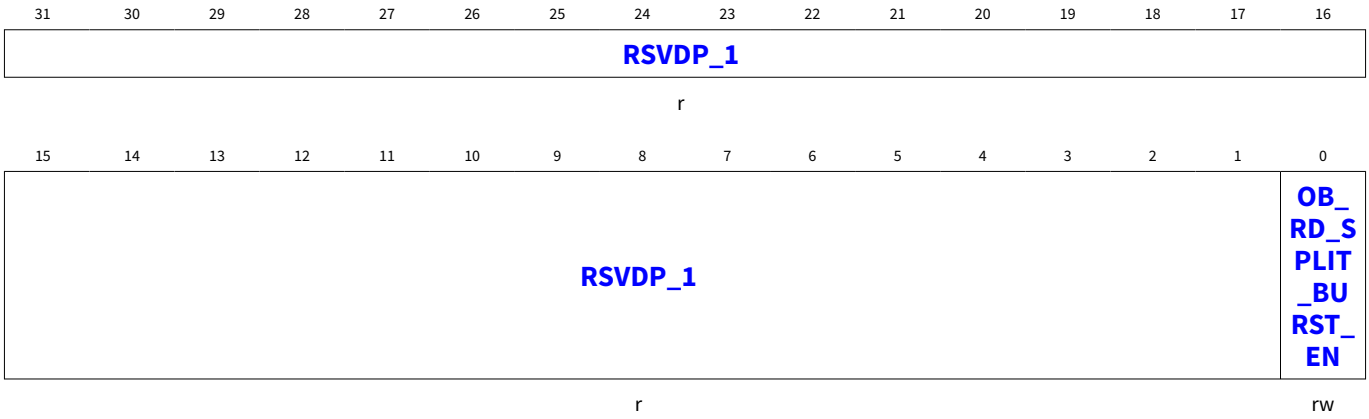
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MASK_RADM_2															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MASK_RADM_2															
rw															

Field	Bits	Type	Description
MASK_RADM_2	31:0	rw	**Filter Mask 2** This field modifies the RADM filtering and error handling rules. For more details, see the "Receive Filtering" in the user manual. In each case, '0' applies the associated filtering rule and '1' masks the associated filtering rule. • [31:10]: Reserved • [9]: CX_FLT_MASK_CPL_IN_LUT_CHECK - 0: Disable masking of checking if the tag of CPL is registered in LUT - 1: Enable masking of checking if the tag of CPL is registered in LUT • [8]: CX_FLT_MASK_POIS_ERROR_REPORTING - 0: Disable masking of error reporting for Poisoned TLPs - 1: Enable masking of error reporting for Poisoned TLPs • [7]: CX_FLT_MASK_PRS_DROP - 0: Allow PRS message to pass through - 1: Drop PRS Messages silently - This bit is ignored when the CX_FLT_MASK_MSG_DROP bit in the MASK_RADM_1 field of the SYMBOL_TIMER_FILTER_1_OFF register is set to '1' • [6]: CX_FLT_UNMASK_TD - 0: Disable unmask TD bit if CX_STRIP_ECRC_ENABLE - 1: Enable unmask TD bit if CX_STRIP_ECRC_ENABLE • [5]: CX_FLT_UNMASK_UR_POIS_TRGT0 - 0: Disable unmask CX_FLT_MASK_UR_POIS with TRGT0 destination - 1: Enable unmask CX_FLT_MASK_UR_POIS with TRGT0 destination • [4]: CX_FLT_MASK_LN_VENMSG1_DROP - 0: Allow LN message to pass through - 1: Drop LN Messages silently • [3]: CX_FLT_MASK_HANDLE_FLUSH - 0: Disable controller Filter to handle flush request - 1: Enable controller Filter to handle flush request • [2]: CX_FLT_MASK_DABORT_4UCPL - 0: Enable DLLP abort for unexpected completion - 1: Do not enable DLLP abort for unexpected completion • [1]: CX_FLT_MASK_VENMSG1_DROP - 0: Vendor MSG Type 1 dropped silently - 1: Vendor MSG Type 1 not dropped • [0]: CX_FLT_MASK_VENMSG0_DROP - 0: Vendor MSG Type 0 dropped with UR error reporting - 1: Vendor MSG Type 0 not dropped Note: This register field is sticky.

19.4.2.243 AMBA Multiple Outbound Decomposed NP SubRequests Control Register

This register controls multiple outbound decomposed NP subRequests operation.



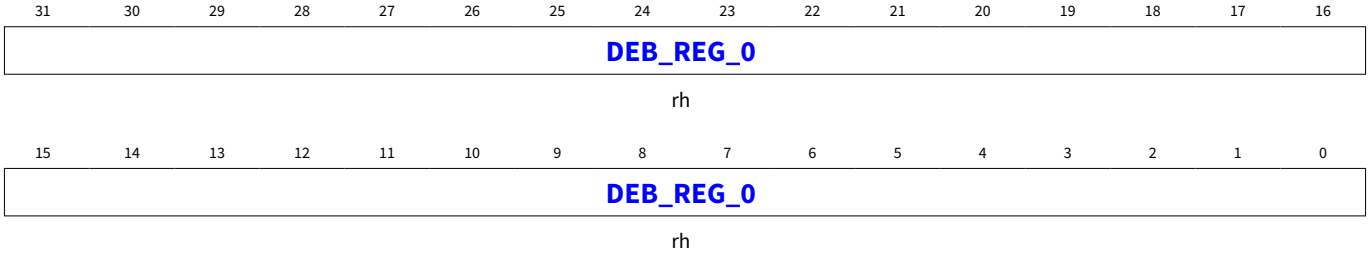
Field	Bits	Type	Description
OB_RD_SPLIT_BURST_EN	0	rw	**Enable AMBA Multiple Outbound Decomposed NP SubRequests** You should not clear this register unless your application master is requesting an amount of read data greater than Max_Read_Request_Size, and the remote device (or switch) is reordering completions that have different tags. Note: The access attributes of this field are as follows: - Wire: R/W (sticky) - Dbi: R/W (sticky) Note: This register field is sticky. 0_B **DISABLE**: This bit when set to '0' disables the possibility of having multiple outstanding non-posted requests that were derived from decomposition of an outbound AMBA request 1_B **ENABLE**: Enable
RSVDP_1	31:1	r	**Reserved for future use**

19.4.2.244 Debug Register 0

This register holds the debug info.

PL_DEBUG0_OFF Offset address: 0004728_H

Debug Register 0 Kernel Reset value: $XXXX\ XXXX_H$



Field	Bits	Type	Description
DEB_REG_0	31:0	rh	**The value on cxpl_debug_info[31:0]** [31:28]: Link control bits advertised by link partner [27]: Currently receiving k237 (PAD) in place of lane number [26]: Currently receiving k237 (PAD) in place of link number [25]: Receiver is receiving logical idle [24]: 2n symbol is also idle (16bit PHY interface only) [23:8]: PIPE(RMMI for M-PCIe) Transmit data [7:6]: PIPE(RMMI for M-PCIe) transmit K indication [5:0]: LTSSM current state. Encoding is same as the dedicated smlh_ltssm_state output. 00000000_H **ZERO**: Zero value $FFFFFFFF_H$ **MAX**: Max value

19.4.2.245 Debug Register 1

This register holds the debug info.

PL_DEBUG1_OFF Offset address: $000472C_H$

Debug Register 1 Kernel Reset value: $XXXX\ XXXX_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DEB_REG_1															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DEB_REG_1															
rh															

Field	Bits	Type	Description
DEB_REG_1	31:0	rh	**The value on cxpl_debug_info[63:32]** [63]: Scrambling disabled for the link [62]: LTSSM in DISABLE state. Link inoperable [61]: LTSSM performing link training [60]: LTSSM is in Polling.Configuration state [59]: LTSSM-negotiated link reset [58:55]: 0000b: Reserved [54]: PIPE receiver detect/loopback request [53]: PIPE transmit electrical idle request [52]: PIPE transmit compliance request [51]: Application request to initiate training reset [50:48]: 000b: Reserved [47:40]: rmlh_ts_link_num: Link number advertised/confirmed by link partner [39:38]: 00b: Reserved [37]: A skip ordered set has been transmitted [36]: LTSSM reports PHY link up or LTSSM is in Loopback. Active for Loopback Master [35]: Receiver reports skip reception [34]: TS1 training sequence received (pulse) [33]: TS2 training sequence received (pulse) [32]: Receiver detected lane reversal 00000000_H **ZERO**: Zero value $FFFFFFFF_H$ **MAX**: Max value

19.4.2.246 Transmit Posted FC Credit Status

This register provides transmit posted FC credit status.

TX_P_FC_CREDIT_STATUS_OFF Offset address: 0004730_H

Transmit Posted FC Credit Status Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_TX_P_FC_CREDIT_STATUS												TX_P_HEADER_FC_CREDIT			
r												rh			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TX_P_HEADER_FC_CREDIT				TX_P_DATA_FC_CREDIT											
rh				rh											

Field	Bits	Type	Description
TX_P_DATA_FC_CREDIT	11:0	rh	**Transmit Posted Data FC Credits** • The posted Data credits advertised by the receiver at the other end of the link, updated with each UpdateFC DLLP • Default value depends on the number of advertised credits for header and data • Scaled Flow Control: when the number of advertised posted credits (both header and data) are infinite • No Scaling: when the number of advertised posted credits (both header and data) are infinite
TX_P_HEADER_FC_CREDIT	19:12	rh	**Transmit Posted Header FC Credits** • The posted Header credits advertised by the receiver at the other end of the link, updated with each UpdateFC DLLP • Default value depends on the number of advertised credits for header and data • Scaled Flow Control: when the number of advertised posted credits (both header and data) are infinite • No Scaling: when the number of advertised posted credits (both header and data) are infinite
RSVDP_TX_P_FC_CREDIT_STATUS	31:20	r	**Reserved for future use**

19.4.2.247 Transmit Non-Posted FC Credit Status

This register provides the transmit Non-Posted FC credit status.

TX_NP_FC_CREDIT_STATUS_OFF Offset address: 0004734_H

Transmit Non-Posted FC Credit Status Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_TX_NP_FC_CREDIT_STATUS												TX_NP_HEADER_FC_CREDIT			
r												rh			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TX_NP_HEADER_FC_CREDIT				TX_NP_DATA_FC_CREDIT											
rh				rh											

Field	Bits	Type	Description
TX_NP_DATA_FC_CREDIT	11:0	rh	**Transmit Non-Posted Data FC Credits** • The non-posted Data credits advertised by the receiver at the other end of the link, updated with each UpdateFC DLLP • Default value depends on the number of advertised credits for header and data • Scaled Flow Control: when the number of advertised non-posted credits (both header and data) are infinite • No Scaling: when the number of advertised non-posted credits (both header and data) are infinite
TX_NP_HEADER_FC_CREDIT	19:12	rh	**Transmit Non-Posted Header FC Credits** • The non-posted Header credits advertised by the receiver at the other end of the link, updated with each UpdateFC DLLP • Default value depends on the number of advertised credits for header and data
RSVDP_TX_NP_FC_CREDIT_STATUS	31:20	r	**Reserved for future use**

19.4.2.248 Transmit Completion FC Credit Status

This register provides transmit completion FC credit status.

TX_CPL_FC_CREDIT_STATUS_OFF Offset address: 0004738_H

Transmit Completion FC Credit Status Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_TX_CPL_FC_CREDIT_STATUS												TX_CPL_HEADER_FC_CREDIT			
r												rh			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TX_CPL_HEADER_FC_CREDIT				TX_CPL_DATA_FC_CREDIT											
rh				rh											

Field	Bits	Type	Description
TX_CPL_DATA_FC_CREDIT	11:0	rh	**Transmit Completion Data FC Credits** • The Completion Data credits advertised by the receiver at the other end of the link, updated with each UpdateFC DLLP • Default value depends on the number of advertised credits for header and data
TX_CPL_HEADER_FC_CREDIT	19:12	rh	**Transmit Completion Header FC Credits** • The Completion Header credits advertised by the receiver at the other end of the link, updated with each UpdateFC DLLP • Default value depends on the number of advertised credits for header and data

(table continues...)

(continued)

Field	Bits	Type	Description
RSVDP_TX_CPL_FC_CREDIT_STATUS	31:20	r	**Reserved for future use**

19.4.2.249 Queue Status

This register provides the queue status.

QUEUE_STATUS_OFF Offset address: 000473C$_H$

Queue Status Kernel Reset value: 0000 0000$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TIMER_MOD_FLOW_CONTROL_EN		RSVDP_29							TIMER_MOD_FLOW_CONTROL						
rw		r							rw						

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		RX_SERIALIZATION_Q_NON_EMPTY	RSVDP_4									RX_QUEUE_OVERFLOW	RX_QUEUE_NON_EMPTY	TX_RETRY_BUFFER_NE	RX_TLP_FC_CREDIT_NON_RETURN
r		rh	r									rw1ch	rh	rh	rh

Field	Bits	Type	Description
RX_TLP_FC_CREDIT_NON_RETURN	0	rh	**Received TLP FC Credits Not Returned** 0_B **CLEAR**: Clear 1_B **SET**: Indicates that the controller has received a TLP but has not yet sent an UpdateFC DLLP indicating that the credits for that TLP have been restored by the receiver at the other end of the link
TX_RETRY_BUFFER_NE	1	rh	**Transmit Retry Buffer Not Empty** 0_B **CLEAR**: Clear 1_B **SET**: Indicates that there is data in the transmit retry buffer
RX_QUEUE_NON_EMPTY	2	rh	**Receive Credit Queue Not Empty** 0_B **CLEAR**: Clear 1_B **SET**: Indicates there is data in one or more of the receive buffers

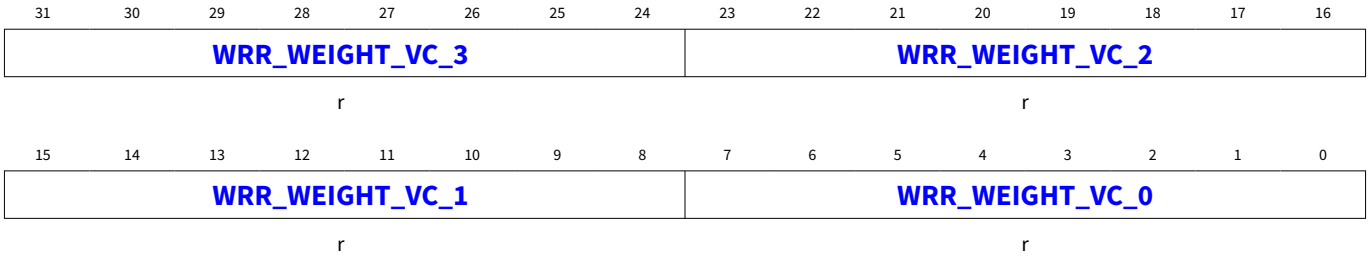
(table continues...)

(continued)

Field	Bits	Type	Description
RX_QUEUE_OVERFLOW	3	rw1ch	**Receive Credit Queue Overflow** Software write with 1 clears this bit-field.. 0_B **CLEAR**: Clear 1_B **SET**: Indicates insufficient buffer space available to write to the P/NP/CPL credit queue
RSVDP_4	12:4	r	**Reserved for future use**
RX_SERIALIZATION_Q_NON_EMPTY	13	rh	**Receive Serialization Queue Not Empty** 0_B **CLEAR**: Clear 1_B **SET**: Indicates there is data in the serialization queue
TIMER_MOD_FLOW_CONTROL	28:16	rw	**FC Latency Timer Override Value** When you set the "FC Latency Timer Override Enable" in this register, the value in this field will override the FC latency timer value that the controller calculates according to the PCIe specification. Note: This register field is sticky.
RSVDP_29	30:29	r	**Reserved for future use**
TIMER_MOD_FLOW_CONTROL_EN	31	rw	**FC Latency Timer Override Enable** Note: This register field is sticky. 0_B **CLEAR**: Clear 1_B **SET**: When this bit is set, the value from the "FC Latency Timer Override Value" field in this register will override the FC latency timer value that the controller calculates according to the PCIe specification
0	15:14	r	**Reserved** Read as all 0's; should be written with all 0's.

19.4.2.250 VC Transmit Arbitration Register 1

This register is used for setting the WRR weights for VC0 - VC3.



Field	Bits	Type	Description
WRR_WEIGHT_VC_0	7:0	r	**WRR Weight for VC0** Note: The access attributes of this field are as follows: - Wire: R - Dbi: R
WRR_WEIGHT_VC_1	15:8	r	**WRR Weight for VC1** Note: The access attributes of this field are as follows: - Wire: R - Dbi: R

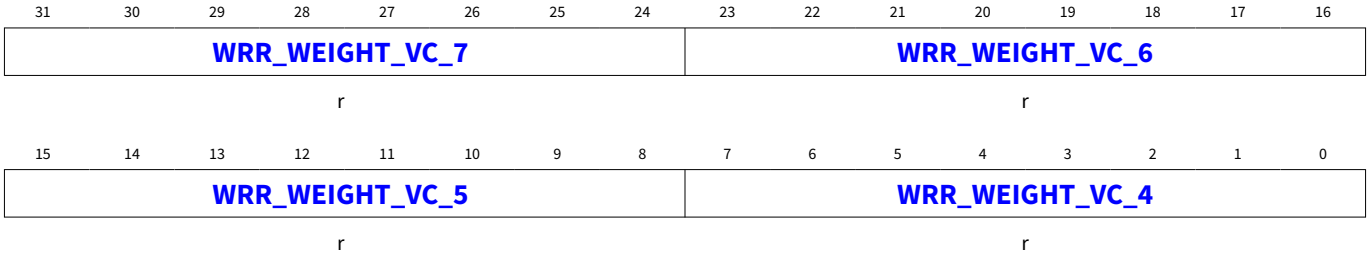
(table continues...)

(continued)

Field	Bits	Type	Description
WRR_WEIGHT_VC_2	23:16	r	**WRR Weight for VC2** Note: The access attributes of this field are as follows: - Wire: R - Dbi: R
WRR_WEIGHT_VC_3	31:24	r	**WRR Weight for VC3** Note: The access attributes of this field are as follows: - Wire: R - Dbi: R

19.4.2.251 VC Transmit Arbitration Register 2

This register is used for setting the WRR weights for VC4 - VC7.



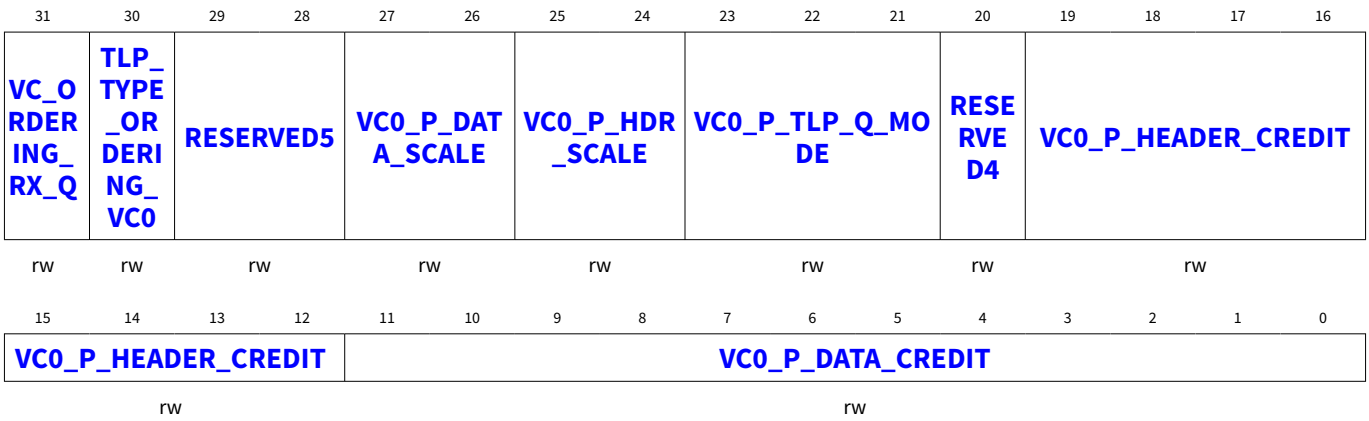
Field	Bits	Type	Description
WRR_WEIGHT_VC_4	7:0	r	**WRR Weight for VC4** Note: The access attributes of this field are as follows: - Wire: R - Dbi: R
WRR_WEIGHT_VC_5	15:8	r	**WRR Weight for VC5** Note: The access attributes of this field are as follows: - Wire: R - Dbi: R
WRR_WEIGHT_VC_6	23:16	r	**WRR Weight for VC6** Note: The access attributes of this field are as follows: - Wire: R - Dbi: R
WRR_WEIGHT_VC_7	31:24	r	**WRR Weight for VC7** Note: The access attributes of this field are as follows: - Wire: R - Dbi: R

19.4.2.252 Segmented-Buffer VC0 Posted Receive Queue Control

This register controls segmented-buffer VC0 posted receive queue operation.

VC0_P_RX_Q_CTRL_OFF Offset address: 0004748_H

Segmented-Buffer VC0 Posted Receive Queue Control Kernel Reset value: $C522\ 2050_H$



Field	Bits	Type	Description
VC0_P_DATA_CREDIT	11:0	rw	**VC0 Posted Data Credits** The number of initial posted data credits for VC0, used only in the segmented-buffer configuration. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: R (sticky) Note: This register field is sticky. 000_H **MIN_VAL**: Zero value FFF_H **MAX_VAL**: Max value
VC0_P_HEADER_CREDIT	19:12	rw	**VC0 Posted Header Credits** The number of initial posted header credits for VC0, used only in the segmented-buffer configuration. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: R (sticky) Note: This register field is sticky. 00_H **MIN_VAL**: Zero value FF_H **MAX_VAL**: Max value
RESERVED4	20	rw	**Reserved** Note: This register field is sticky.
VC0_P_TLP_Q_MODE	23:21	rw	**Reserved** Note: This register field is sticky.
VC0_P_HDR_SCALE	25:24	rw	**VC0 Scale Posted Header Credits** Note: This register field is sticky. 00_B **MIN_VAL**: Zero value 11_B **MAX_VAL**: Max value
VC0_P_DATA_SCALE	27:26	rw	**VC0 Scale Posted Data Credits** Note: This register field is sticky. 00_B **MIN_VAL**: Zero value 11_B **MAX_VAL**: Max value
RESERVED5	29:28	rw	**Reserved** Note: This register field is sticky.
TLP_TYPE_ORDERING_VC0	30	rw	**TLP Type Ordering for VC0** Determines the TLP type ordering rule for VC0 receive queues, used only in the segmented-buffer configuration: Note: This register field is sticky. 0_B **STRICT_ORDERING**: Strict ordering: posted, completion, then non-posted 1_B **PCIE_ORDERING_RULE**: PCIe ordering rules (recommended)
VC_ORDERING_RX_Q	31	rw	**VC Ordering for Receive Queues** Determines the VC ordering rule for the receive queues, used only in the segmented-buffer configuration: Note: This register field is sticky. 0_B **ROUND_ROBIN**: Round robin 1_B **STRICT_ORDERING**: Strict ordering, higher numbered VCs have higher priority

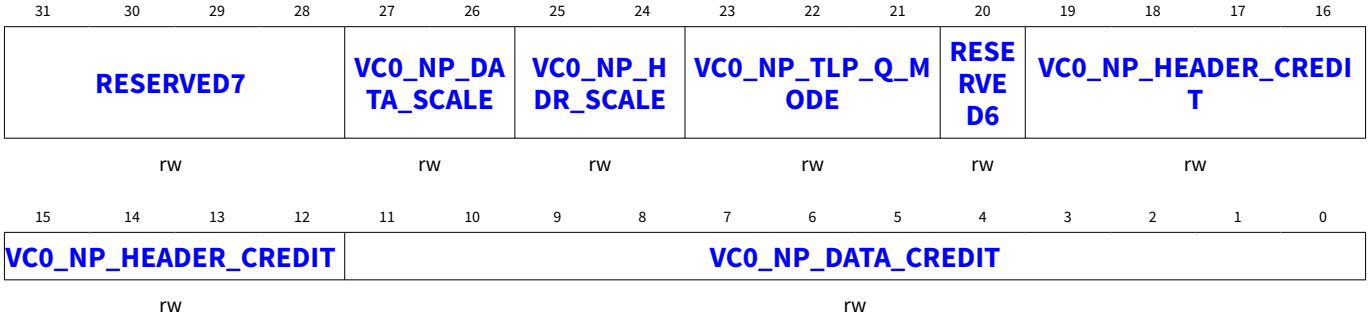
19.4.2.253 Segmented-Buffer VC0 Non-Posted Receive Queue Control

This register controls the segmented-buffer VC0 non-posted receive queue operation.

VC0_NP_RX_Q_CTRL_OFF Offset address: $000474C_H$

Segmented-Buffer VC0 Non-Posted Receive Queue Control

Kernel Reset value: $0522\ 2008_H$



Field	Bits	Type	Description
VC0_NP_DATA_CREDIT	11:0	rw	**VC0 Non-Posted Data Credits** The number of initial non-posted data credits for VC0, used only in the segmented-buffer configuration. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: R (sticky) Note: This register field is sticky. 000_H **MIN_VAL**: Zero value FFF_H **MAX_VAL**: Max value
VC0_NP_HEADER_CREDIT	19:12	rw	**VC0 Non-Posted Header Credits** The number of initial non-posted header credits for VC0, used only in the segmented-buffer configuration. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: R (sticky) Note: This register field is sticky. 00_H **MIN_VAL**: Zero value FF_H **MAX_VAL**: Max value
RESERVED6	20	rw	**Reserved** Note: This register field is sticky.
VC0_NP_TLP_Q_MODE	23:21	rw	**Reserved** Note: This register field is sticky.
VC0_NP_HDR_SCALE	25:24	rw	**VC0 Scale Non-Posted Header Credits** Note: This register field is sticky. 00_B **MIN_VAL**: Zero value 11_B **MAX_VAL**: Max value
VC0_NP_DATA_SCALE	27:26	rw	**VC0 Scale Non-Posted Data Credits** Note: This register field is sticky. 00_B **MIN_VAL**: Zero value 11_B **MAX_VAL**: Max value
RESERVED7	31:28	rw	**Reserved** Note: This register field is sticky.

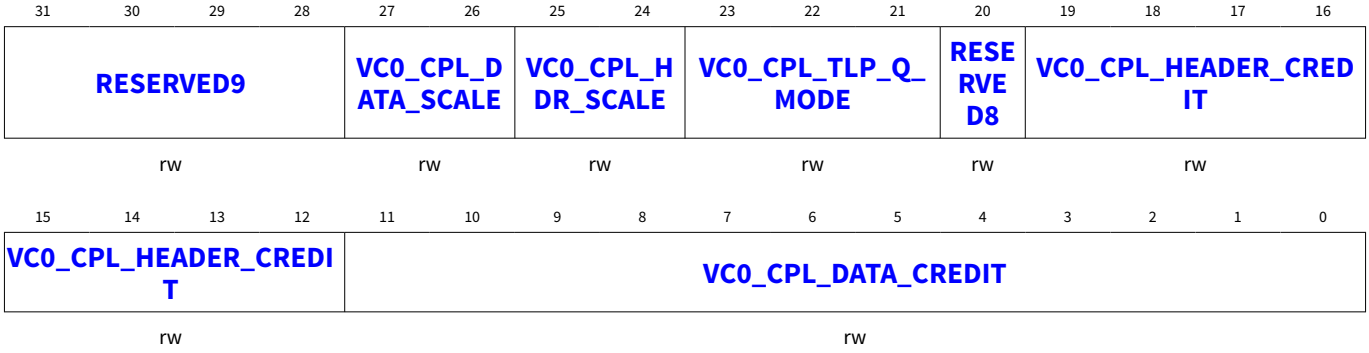
19.4.2.254 Segmented-Buffer VC0 Completion Receive Queue Control

This register controls the segmented-buffer VC0 completion receive queue operation.

VC0_CPL_RX_Q_CTRL_OFF Offset address: 0004750_H

Segmented-Buffer VC0 Completion Receive Queue Control

Kernel Reset value: $0520\ 0000_H$



Field	Bits	Type	Description
VC0_CPL_DATA_CREDIT	11:0	rw	**VC0 Completion Data Credits** The number of initial Completion data credits for VC0, used only in the segmented-buffer configuration. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: R (sticky) Note: This register field is sticky. 000_H **MIN_VAL**: Zero value FFF_H **MAX_VAL**: Max value
VC0_CPL_HEADER_CREDIT	19:12	rw	**VC0 Completion Header Credits** The number of initial Completion header credits for VC0, used only in the segmented-buffer configuration. Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: R (sticky) Note: This register field is sticky. 00_H **MIN_VAL**: Zero value FF_H **MAX_VAL**: Max value
RESERVED8	20	rw	**Reserved** Note: This register field is sticky.
VC0_CPL_TLP_Q_MODE	23:21	rw	**Reserved** Note: This register field is sticky.
VC0_CPL_HDR_SCALE	25:24	rw	**VC0 Scale CPL Header Credits** Note: This register field is sticky. 00_B **MIN_VAL**: Zero value 11_B **MAX_VAL**: Max value
VC0_CPL_DATA_SCALE	27:26	rw	**VC0 Scale CPL Data Credits** Note: This register field is sticky. 00_B **MIN_VAL**: Zero value 11_B **MAX_VAL**: Max value
RESERVED9	31:28	rw	**Reserved** Note: This register field is sticky.

19.4.2.255 Link Width and Speed Change Control Register

This register controls various functions of the controller related to link training, lane reversal, and equalization.

GEN2_CTRL_OFF Offset address: 000480C$_H$

Link Width and Speed Change Control Register Kernel Reset value: 0002 01B4$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								SELECTABLE_DEEMPH_BIT_MUX	SELECT_DEEMPH_VAR_MUX	GEN1_EI_INFERENCE	SEL_DEEMPHASIS	CONFIG_TX_COMP_RX	Res	DIRECT_SPEED_CHANGE	AUTO_LANE_FLIP_CTRL_EN
r								rw	rw	rw	rw	rw	r	rwh	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PRE_DET_LANE			NUM_OF_LANES					FAST_TRAINING_SEQ							
rw			rw					rw							

Field	Bits	Type	Description
FAST_TRAINING_SEQ	7:0	rw	**Sets the Number of Fast Training Sequences (N_FTS) that the controller advertises as its N_FTS during Gen2 or Gen3 link training** This value is used to inform the link partner about the PHY's ability to recover synchronization after a low power state. Do not set N_FTS to zero; doing so can cause the LTSSM to go into the recovery state when exiting from L0s. Note: The access attributes of this field are as follows: - Wire: R/W (sticky) - Dbi: R/W (sticky) Note: This register field is sticky.
NUM_OF_LANES	12:8	rw	**Predetermined Number of Lanes** Defines the number of lanes which are connected and not bad. Used to limit the effective link width to ignore 'broken" or "unused" lanes that detect a receiver. Indicates the number of lanes to check for exit from Electrical Idle in Polling.Active and L2.Idle. It is possible that the LTSSM might detect a receiver on a bad or broken lane during the Detect Substate. However, it is also possible that such a lane might also fail to exit Electrical Idle and therefore prevent a valid link from being configured. This value is referred to as the "Predetermined Number of Lanes" in section 4.2.6.2.1 of the PCI Express Base Specification. Note: The access attributes of this field are as follows: - Wire: R/W (sticky) - Dbi: R/W (sticky) Note: This register field is sticky. 01_H **_1_LANE**: 1 lane 02_H **_2_LANE**: 2 lanes 03_H **_3_LANE**: 3 lanes **others,** Reserved

(table continues...)

(continued)

Field	Bits	Type	Description
PRE_DET_LANE	15:13	rw	**Predetermined Lane for Auto Flip** This field defines which physical lane is connected to logical Lane0 by the flip operation performed in Detect. This field is used to restrict the receiver detect procedure to a particular lane when the default detect and polling procedure performed on all lanes cannot be successful. A notable example of when it is useful to program this field to a value different from the default, is when a lane is asymmetrically broken, that is, it is detected in Detect LTSSM state but it cannot exit Electrical Idle in Polling LTSSM state. Note: The access attributes of this field are as follows: - Wire: R/W (sticky) - Dbi: R/W (sticky) Note: This register field is sticky. 000_B **LANE0**: Connect logical Lane0 to physical lane 0, depending on which lane is detected 001_B **LANE1**: Connect logical Lane0 to physical lane 1 010_B **LANE3**: Connect logical Lane0 to physical lane 3 011_B **LANE7**: connect logical lane0 to physical lane 7 100_B **LANE15**: Connect logical Lane0 to physical lane 15 **others,** Reserved
AUTO_LANE_FLIP_CTRL_EN	16	rw	**Enable Auto flipping of the lanes** This feature is not supported by hardware. Note: The access attributes of this field are as follows: - Wire: R/W (sticky) - Dbi: R/W (sticky) Note: This register field is sticky. 0_B **DISABLE**: Disable 1_B **ENABLE**: Enable
DIRECT_SPEED_CHANGE	17	rwh	**Directed Speed Change** When the speed change occurs, the controller will clear the contents of this field; and a read to this field by your software will return a '0'. To manually initiate the speed change: • Write to LINK_CONTROL2_LINK_STATUS2_REG.PCIE_CAP_TARGET_LINK_SPEED in the local device • Deassert this field • Assert this field If you set the default of this field using the DEFAULT_GEN2_SPEED_CHANGE configuration parameter to '1', then the speed change is initiated automatically after link up, and the controller clears the contents of this field. If you want to prevent this automatic speed change, then write a lower speed value to the Target Link Speed field of the Link Control 2 register (LINK_CONTROL2_LINK_STATUS2_OFF.PCIE_CAP_TARGET_LINK_SPEED) through the DBI before link up. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W 0_B **CLEAR**: Clear 1_B **SET**: Writing '1' to this field instructs the LTSSM to initiate a speed change to Gen2 or Gen3 after the link is initialized at Gen1 speed

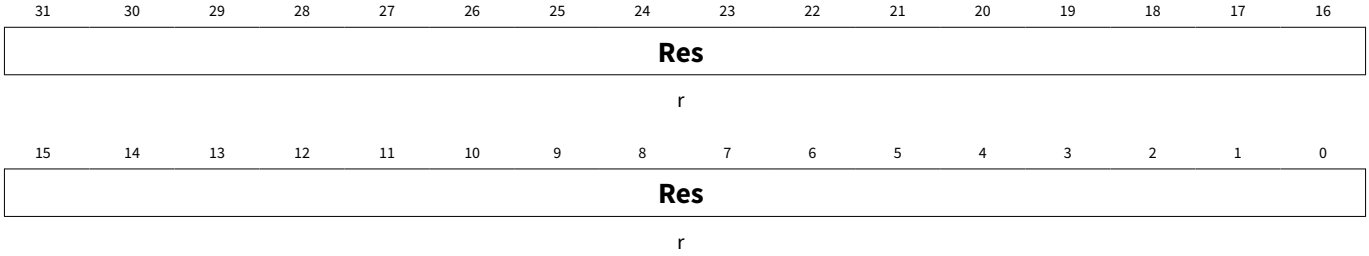
(table continues...)

(continued)

Field	Bits	Type	Description
CONFIG_TX_COMP_RX	19	rw	**Config Tx Compliance Receive Bit** Note: The access attributes of this field are as follows: - Wire: R/W (sticky) - Dbi: R/W (sticky) Note: This register field is sticky. 0_B **CLEAR**: Clear 1_B **SET**: When set to 1, signals LTSSM to transmit TS ordered sets with the compliance receive bit assert (equal to '1')
SEL_DEEMPHASIS	20	rw	**Used to set the de-emphasis level for upstream ports** This bit selects the level of de-emphasis the link operates at. Note: The access attributes of this field are as follows: - Wire: R/W (sticky) - Dbi: R/W (sticky) Note: This register field is sticky. 0_B **N_6DB**: -6 dB 1_B **N_3_5DB**: -3.5 dB
GEN1_EI_INFERENCE	21	rw	**Electrical Idle Inference Mode at Gen1 Rate** Programmable mode to determine inferred electrical idle (EI) in Recovery.Speed or Loopback.Active (as slave) state at Gen1 speed by looking for a '1' value on RxElecIdle instead of looking for a '0' on RxValid. If the PHY fails to deassert the RxValid signal in Recovery.Speed or Loopback.Active (because of corrupted EIOS for example), then EI cannot be inferred successfully in the controller by just detecting the condition RxValid=0. Note: This register field is sticky. 0_B **RXELECIDLE_TO_INFER**: Use RxElecIdle signal to infer Electrical Idle 1_B **RXVALID_TO_INFER**: Use RxValid signal to infer Electrical Idle
SELECT_DEEMPH_VAR_MUX	22	rw	**The select_deemphasis variable for DSP on entry to Recovery.RcvrCfg state is muxed between the Selectable De-emphasis field in the Link Control 2 register and the value requested by the Upstream Port in the eight consecutive TS1 Ordered Sets it received** Note: This register field is sticky. 0_B **REQUESTED_BY_USP**: The value requested by USP in Recovery.RcvrLock state through Tx TS1s from USP 1_B **FROM_LINK_CTRL_2_REG**: The value from the Selectable De-emphasis field in the Link Control 2 register
SELECTABLE_DEEMPH_BIT_MUX	23	rw	**The selectable deemphasis bit (Symbol 4 bit 6) of the transmitted TS2 Ordered Sets for DSP in Recovery.RcvrCfg state is muxed between the Selectable De-emphasis field in the Link Control 2 register and the value requested by USP in Recovery.RcvrLock state through Tx TS1s from USP** Note: This register field is sticky. 0_B **FROM_LINK_CTRL_2_REG**: The value from the Selectable De-emphasis field in the Link Control 2 register 1_B **REQUESTED_BY_USP**: The value requested by USP in Recovery.RcvrLock state through Tx TS1s from USP
0	31:24	r	**Reserved** Read as all 0's; should be written with all 0's.

19.4.2.256 PHY Status Register

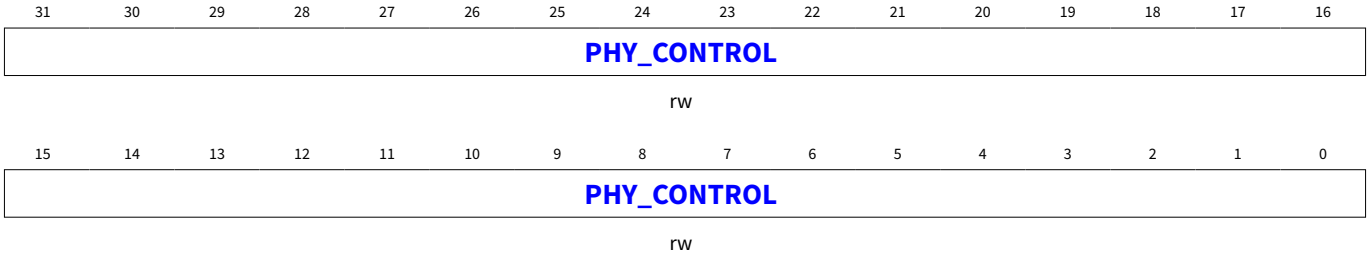
Memory mapped register from phy_cfg_status GPIO input pins.



Field	Bits	Type	Description

19.4.2.257 PHY Control Register

Memory mapped register to cfg_phy_control GPIO output pins.



Field	Bits	Type	Description
PHY_CONTROL	31:0	rw	**PHY Control** Data sent directly to the cfg_phy_control bus. This is a GPIO register driving the values on the static cfg_phy_control output signals, and does not in any way influence controller functionality. It can be used for any static sideband control signaling requirements that you have for your PHY. Usage of this register and the associated GPIO bus is RESERVED when the controller is connected to a Synopsys PHY. Note: This register field is sticky. 00000000_H **MIN_VAL**: Zero value $FFFFFFFF_H$ **MAX_VAL**: Max value

19.4.2.258 Programmable Target Map Control Register

This register controls the target map.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TARGET_MAP_RESERVED_21_31											TARGET_MAP_INDEX				
r											rwh				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TARGET_MAP_RESERVED_13_15			0						TARGET_MAP_ROM	TARGET_MAP_PF					
r			r						rwh	rwh					

Field	Bits	Type	Description
TARGET_MAP_PF	5:0	rwh	**Target Values for each BAR on the PF Function selected by the index number** This register does not respect the Byte Enable setting, any write will affect all register bits.
TARGET_MAP_ROM	6	rwh	**Target Value for the ROM page of the PF Function selected by the index number** This register does not respect the Byte Enable setting, any write will affect all register bits.
TARGET_MAP_RESERVED_13_15	15:13	r	**Reserved** Note: The access attributes of this field are as follows: - Wire: RSVDP - Dbi: R (sticky)
TARGET_MAP_INDEX	20:16	rwh	**The number of the PF Function on which the Target Values are set** This register does not respect the Byte Enable setting, any write will affect all register bits.
TARGET_MAP_RESERVED_21_31	31:21	r	**Reserved** Note: The access attributes of this field are as follows: - Wire: RSVDP - Dbi: R (sticky)
0	12:7	r	**Reserved** Read as all 0's; should be written with all 0's.

19.4.2.259 Integrated MSI Reception Module (iMRM) Address Register

This register holds the integrated MSI reception module address.

MSI_CTRL_ADDR_OFF Offset address: 0004820_H

Integrated MSI Reception Module [iMRM) Address Register

Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MSI_CTRL_ADDR															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MSI_CTRL_ADDR															
rw															

Field	Bits	Type	Description
MSI_CTRL_ADDR	31:0	rw	**Integrated MSI Reception Module Address** System specified address for MSI memory write transaction termination. Within the AXI Bridge, every received Memory Write request is examined to see if it targets the MSI Address that has been specified in this register; and also to see if it satisfies the definition of an MSI interrupt request. When these conditions are satisfied the Memory Write request is marked as an MSI request. Note: This register field is sticky. 00000000_H **MIN_VAL**: Zero value $FFFFFFFF_H$ **MAX_VAL**: Max value

19.4.2.260 Integrated MSI Reception Module Upper Address Register

This register holds the integrated MSI reception module upper address.

MSI_CTRL_UPPER_ADDR_OFF Offset address: 0004824_H

Integrated MSI Reception Module Upper Address Register — Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MSI_CTRL_UPPER_ADDR															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MSI_CTRL_UPPER_ADDR															
rw															

Field	Bits	Type	Description
MSI_CTRL_UPPER_ADDR	31:0	rw	**Integrated MSI Reception Module Upper Address** System specified upper address for MSI memory write transaction termination. Allows functions to support a 64-bit MSI address. Note: This register field is sticky. 00000000_H **MIN_VAL**: Zero value $FFFFFFFF_H$ **MAX_VAL**: Max value

19.4.2.261 Integrated MSI i Reception Module Interrupt0 Enable Register

This register enables integrated MSI reception module interrupt0.

MSI_CTRL_INT_i_EN_OFF (i=0-7) Offset address: 0004828_H+i*12

Integrated MSI i Reception Module Interrupt0 Enable Register — Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MSI_CTRL_INT_0_EN															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MSI_CTRL_INT_0_EN															
rw															

Field	Bits	Type	Description
MSI_CTRL_INT_0_EN	31:0	rw	**MSI Interrupt0 Enable** Specifies which interrupts are enabled. When an MSI is received from a disabled interrupt, no status bit gets set in MSI controller interrupt status register. Each bit corresponds to a single MSI Interrupt Vector. Note: This register field is sticky. 00000000_H **MIN_VAL**: Zero value $FFFFFFFF_H$ **MAX_VAL**: Max value

19.4.2.262 Integrated MSI i Reception Module Interrupt0 Mask Register

This register provides information regarding the MSI Interrupt0 mask.

MSI_CTRL_INT_i_MASK_OFF (i=0-7) Offset address: $000482C_H$+i*12

Integrated MSI i Reception Module Interrupt0 Mask Register — Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MSI_CTRL_INT_0_MASK															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MSI_CTRL_INT_0_MASK															
rw															

Field	Bits	Type	Description
MSI_CTRL_INT_0_MASK	31:0	rw	**MSI Interrupt0 Mask** Allows enabled interrupts to be masked. When an MSI is received for a masked interrupt, the corresponding status bit gets set in the interrupt status register but the msi_ctrl_int output is not set HIGH. Each bit corresponds to a single MSI Interrupt Vector. Note: This register field is sticky.

19.4.2.263 Integrated MSI i Reception Module Interrupt0 Status Register

This register provides the MSI Interrupt0 status.

MSI_CTRL_INT_i_STATUS_OFF (i=0-7) Offset address: 0004830_H+i*12

Integrated MSI i Reception Module Interrupt0 Status Register — Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MSI_CTRL_INT_0_STATUS															
rw1ch															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MSI_CTRL_INT_0_STATUS															
rw1ch															

Field	Bits	Type	Description
MSI_CTRL_INT_0_STATUS	31:0	rw1ch	**MSI Interrupt0 Status** When an MSI is detected for EP#i, one bit in this register is set. The decoding of the data payload of the MSI Memory Write request determines which bit gets set. Each bit corresponds to a single MSI Interrupt Vector. Software write with 1 clears this bit-field. 00000000_H **MIN_VAL**: Zero value $FFFFFFFF_H$ **MAX_VAL**: Max value

19.4.2.264 Integrated MSI Reception Module General Purpose IO Register

The contents of this register drive the top-level GPIO msi_ctrl_io[31:0].

MSI_GPIO_IO_OFF Offset address: 0004888_H

Integrated MSI Reception Module General Purpose IO Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MSI_GPIO_REG															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MSI_GPIO_REG															
rw															

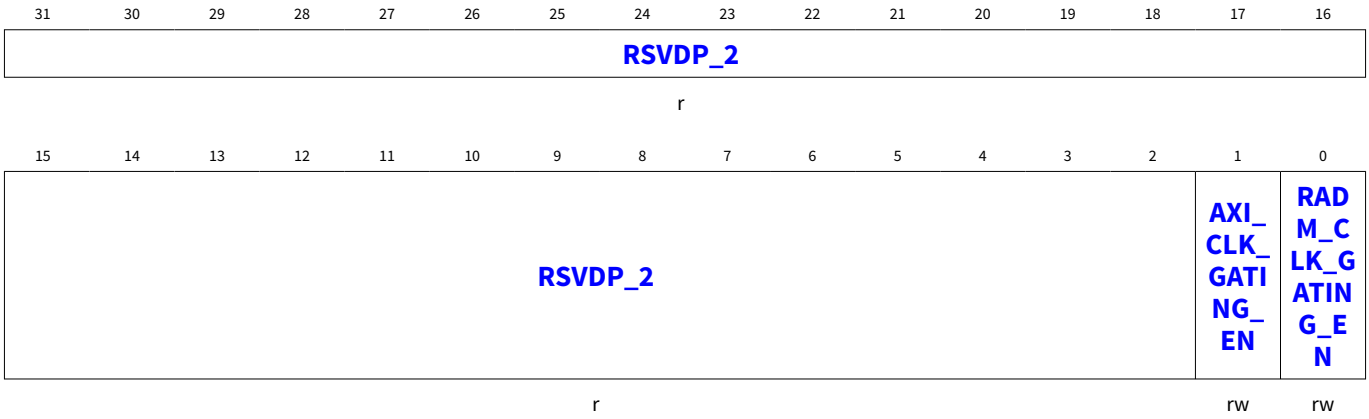
Field	Bits	Type	Description
MSI_GPIO_REG	31:0	rw	**MSI GPIO Register** The contents of this register drive the top-level GPIO msi_ctrl_io[31:0]. Note: This register field is sticky. 00000000_H **MIN_VAL**: Zero value $FFFFFFFF_H$ **MAX_VAL**: Max value

19.4.2.265 Clock Gating Control Register

This register enables you to disable dynamic clock gating. By default dynamic clock gating is on, allowing the controller to autonomously enable and disable its clocks. The clock gating is performed in the clock and reset module, and is initiated by the controllers clock enable signals. The following modules support dynamic clock gating: - AXI Bridge - RADM

CLOCK_GATING_CTRL_OFF Offset address: $000488C_H$

Clock Gating Control Register Kernel Reset value: $0000\ 0003_H$



Field	Bits	Type	Description
RADM_CLK_GATING_EN	0	rw	**RADM Clock Gating Enable** This register, if set, enables the RADM to autonomously enable and disable its clock. Note: This register field is sticky. 0_B **DISABLE**: Disable 1_B **ENABLE**: Enable (default)
AXI_CLK_GATING_EN	1	rw	**AXI Clock Gating Enable** This register enables the AXI Bridge to autonomously enable and disable the AXI Master clock, the AXI Slave clock and the AXI DBI slave clock. Note: This register field is sticky. 0_B **DISABLE**: Disable 1_B **ENABLE**: Enable (default)
RSVDP_2	31:2	r	**Reserved for future use**

19.4.2.266 Gen3 Control Register

There is no Gen3-specific N_FTS field. The N_FTS field in the "Link Width and Speed Change Control Register" is used for both Gen2 and Gen3 speed modes. There is no Gen3-specific "Directed Speed Change" field. The "Directed Speed Change" field in the "Link Width and Speed Change Control Register" is used to change to Gen2 or Gen3 speed. A speed change to Gen3 occurs if (1) the "Directed Speed Change" field is set to '1' and (2) the "Target Link Speed" field in the Link Control 2 Register is set to Gen3. Gen3 support is advertised by both sides of the link during link training.

GEN3_RELATED_OFF Offset address: 0004890_H

Gen3 Control Register Kernel Reset value: $0000\ 2001_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_26						0		GEN3_EQ_INVREQ_EVAL_DIFF_DISABLE	0		RSVDP_19		GEN3_DC_BALANCE_DISABLE	GEN3_DLLP_XMT_DELAY_DISABLE	GEN3_EQUALIZATION_DISABLE
r						r		rw	r		r		rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVDP_14		RXEQ_RGRDLESS_RXTS	RXEQ_PH01_EN	EQ_REDO	EQ_EIEOS_CNT	EQ_PHASE_2_3	DISABLE_SCRAMBLER_GEN_3	RSVDP_2						NO_SEED_VALUE_CHANGE	GEN3_ZRXDC_NONCOMPL
r		rw	rw	rw	rw	rw	rw	r						rw	rw

Field	Bits	Type	Description
GEN3_ZRXDC_NONCOMPL	0	rw	**Gen3 Receiver Impedance ZRX-DC Not Compliant** Receivers that operate at 8.0 GT/s with an impedance other than the range defined by the ZRX-DC parameter for 2.5 GT/s (40-60 Ohms) must meet additional behavior requirements in the following LTSSM states: Polling, Rx_L0s, L1, L2, and Disabled. Note: This register field is sticky. 0_B **RCVR_COMPLIES**: The receiver complies with the ZRX-DC parameter for 2.5 GT/s when operating at 8 GT/s or higher 1_B **RCVR_NOT_COMPLIES**: The receiver does not comply with the ZRX-DC parameter for 2.5 GT/s when operating at 8 GT/s or higher
NO_SEED_VALUE_CHANGE	1	rw	**If this bit is set to 1, the seed value of LFSR for scrambler at Gen3 rate does not change after LinkUp = 1** This feature is not supported. Note: this register is shared for Gen3 and Gen4/Gen5 data rates. Note: This register field is sticky. 1_B **NO_CHANGE**: Not Change
RSVDP_2	7:2	r	**Reserved for future use**
DISABLE_SCRAMBLER_GEN_3	8	rw	**Disable Scrambler for Gen3 and Gen4 Data Rate** The Gen3 and Gen4 scrambler/descrambler within the controller needs to be disabled when the scrambling function is implemented outside of the controller (for example within the PHY). Note: This register field is sticky.
EQ_PHASE_2_3	9	rw	**Equalization Phase 2 and Phase 3 Disable** This applies to downstream ports only. Note: The access attributes of this field are as follows: - Wire: see description - Dbi: see description Note: This register field is sticky. 0_B **NO_TX_EQ_RX_EQ_PH01**: Rx equalization in phase 0/1 1_B **TX_EQ_23_RX_EQ_PH01**: No Rx equalization

(table continues...)

(continued)

Field	Bits	Type	Description
EQ_EIEOS_CNT	10	rw	**Equalization EIEOS Count Reset Disable** Disable requesting reset of EIEOS count during equalization. Note: This register field is sticky. 1_B **DISABLE**: Disable
EQ_REDO	11	rw	**Equalization Redo Disable** Disable autonomous mechanism for requesting to redo the equalization process. The received presets or coefficients mismatch in Recovery.RcvrLock after Recovery EQ phases causes the EQ redo requests. If the EQ redo is infinite or you do not want eq requests and redo, setting this bit to 1 will stop the EQ requests and EQ redo so that the link can go ahead to L0 state for packet trasmissions. Note: This register field is sticky.
RXEQ_PH01_EN	12	rw	**Rx Equalization Phase 0/Phase 1 Hold Enable** When this bit is set the upstream port holds phase 0 (the downstream port holds phase 1) for 10ms. Holding phase 0 or phase 1 can be used to allow sufficient time for Rx Equalization to be performed by the PHY. This bit is used during Virtex-7 Gen3 equalization. The programmable bits [RXEQ_PH01_EN, EQ_PHASE_2_3] can be used to obtain the following variations of the equalization procedure: Note: The access attributes of this field are as follows: - Wire: see description - Dbi: see description Note: This register field is sticky. 0_B **TX_EQ_23**: Tx equalization only in phase 2/3 1_B **NO_TX_EQ_NO_RX_EQ**: No Tx equalization
RXEQ_RGRDLESS_RXTS	13	rw	**When set to '1', the controller as Gen3 EQ master asserts RxEqEval to instruct the PHY to do Rx adaptation and evaluation after a 500ns timeout from a new preset request** . Note: The access attributes of this field are as follows: - Wire: see description - Dbi: see description Note: This register field is sticky. 0_B **ASSERT_1US**: mac_phy_rxeqeval asserts after 1 µs and 2 TS1 received from remote partner 1_B **ASSERT_500NS**: mac_phy_rxeqeval asserts after 500ns regardless of TS's received or not
RSVDP_14	15:14	r	**Reserved for future use**
GEN3_EQUALIZATION_DISABLE	16	rw	**Equalization Disable** Disable equalization feature. This bit cannot be changed once the LTSSM starts link training. Note: This register field is sticky.
GEN3_DLLP_XMT_DELAY_DISABLE	17	rw	**DLLP Transmission Delay Disable** Disable delay transmission of DLLPs before equalization. Note: This register field is sticky.
GEN3_DC_BALANCE_DISABLE	18	rw	**DC Balance Disable** Disable DC Balance feature. Note: This register field is sticky.
RSVDP_19	20:19	r	**Reserved for future use**
GEN3_EQ_INVREQ_EVAL_DIFF_DISABLE	23	rw	**Eq InvalidRequest and RxEqEval Different Time Assertion Disable** Disable the assertion of Eq InvalidRequest and RxEqEval at different time. Note: This register field is sticky.

(table continues...)

(continued)

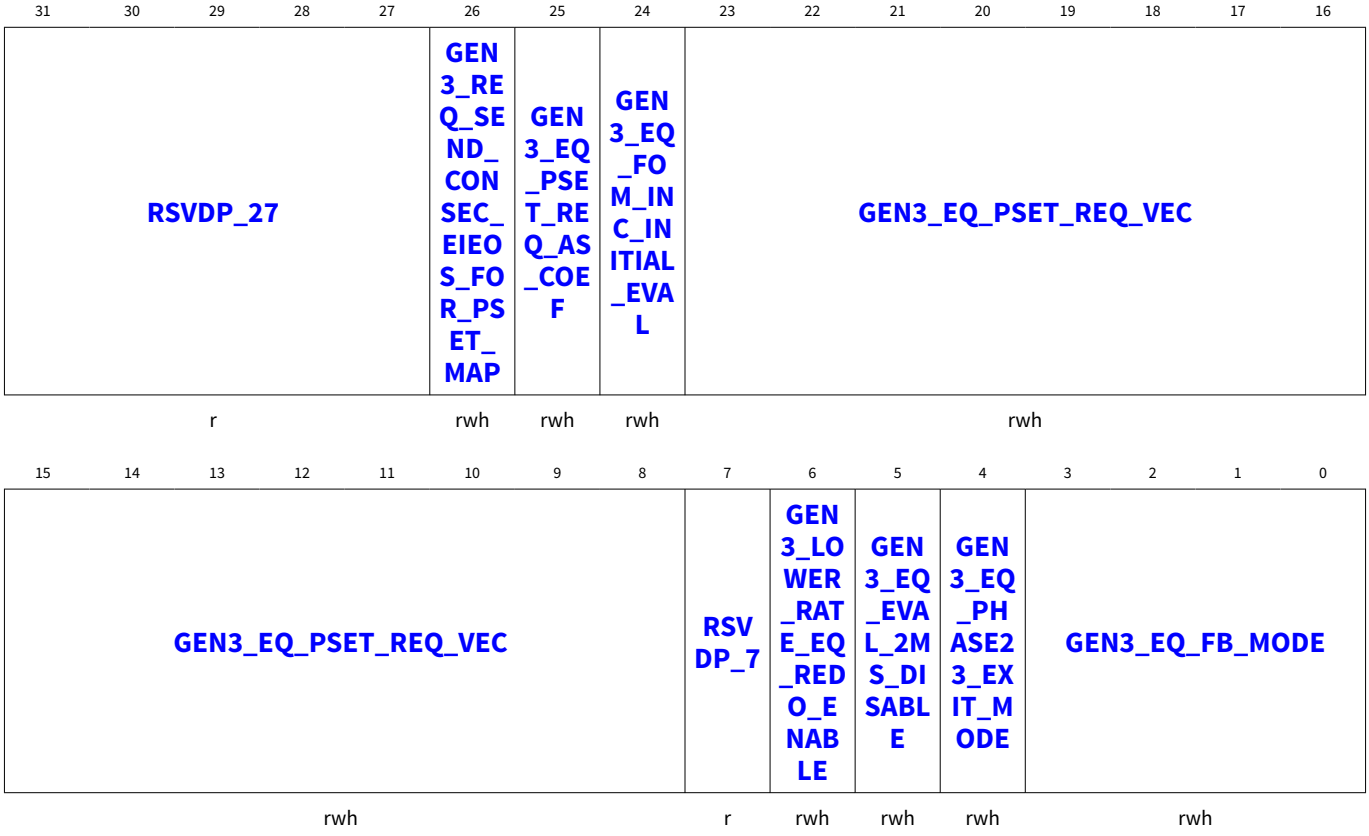
Field	Bits	Type	Description
RSVDP_26	31:26	r	**Reserved for future use**
0	22:21, 25:24	r	**Reserved** Read as all 0's; should be written with all 0's.

19.4.2.267 Gen3 EQ Control Register

This register controls equalization for Phase2 in an upstream port (USP), or Phase3 in a downstream port (DSP).

GEN3_EQ_CONTROL_OFF Offset address: $00048A8_H$

Gen3 EQ Control Register Kernel Reset value: $0501\ 7271_H$



Field	Bits	Type	Description
GEN3_EQ_FB_MODE	3:0	rwh	**Feedback Mode** Other values are reserved. Note: This register field is sticky. 0_H **DIR_CHG**: Direction Change 1_H **FOM**: Figure Of Merit **others,** Reserved

(table continues...)

(continued)

Field	Bits	Type	Description
GEN3_EQ_PHASE23_EXIT_MODE	4	rwh	**Behavior After 24 ms Timeout (when optimal settings are not found)** For a USP: Determine next LTSSM state from Phase2 after 24ms Timeout • 0: Recovery.Speed • 1: Recovery.Equalization.Phase3 When optimal settings are not found then: • Equalization Phase 2 Successful status bit is not set in the "Link Status Register 2" when GEN3_EQ_PHASE23_EXIT_MODE = 0 • Equalization Phase 2 Successful status bit is set in the "Link Status Register 2" when GEN3_EQ_PHASE23_EXIT_MODE = 1 • Equalization Phase 2 Complete status bit is set in the "Link Status Register 2" For a DSP: Determine next LTSSM state from Phase3 after 24ms Timeout • 0: Recovery.Speed • 1: Recovery.Equalization.RcvrLock When optimal settings are not found then: • Equalization Phase 3 Successful status bit is not set in the "Link Status Register 2" when GEN3_EQ_PHASE23_EXIT_MODE = 0 • Equalization Phase 3 Successful status bit is set in the "Link Status Register 2" when GEN3_EQ_PHASE23_EXIT_MODE = 1 • Equalization Phase 3 Complete status bit is set in the "Link Status Register 2" Note: GEN3_EQ_PHASE23_EXIT_MODE = 1 affects Direction Change feed back mode. EQ requests for Figure Of Merit mode complete before 24 ms timeout. Please see GEN3_EQ_PSET_REQ_VEC Register for more. Note: This register field is sticky. 0_B **RCVRY_SPEED**: Recovery.Speed 1_B **RCVRY_EQ**: USP: Recovery.Equalization.Phase3; DSP: Recovery.Equalization.RcvrLock
GEN3_EQ_EVAL_2MS_DISABLE	5	rwh	**Phase2_3 2 ms Timeout Disable** Determine behavior in Phase2 for USP (Phase3 if DSP) when the PHY does not respond within 2ms to the assertion of RxEqEval. Note: This register field is sticky. 0_B **ABORT_CURRENT_EVA**: Abort the current evaluation, stop any attempt to modify the remote transmitter settings, Phase2 is terminated by the 24ms timeout 1_B **IGNORE_2MS_TIMEOUT**: Ignore the 2ms timeout and continue as normal This is used to support PHYs that require more than 2ms to respond to the assertion of RxEqEval.
GEN3_LOWER_RATE_EQ_REDO_ENABLE	6	rwh	**Support EQ redo and lower rate change** To access this field, RATE_SHADOW_SEL should be set to 0. Note: This register field is sticky. 0_B **NO_SUP**: Not supported 1_B **SUP**: Supported
RSVDP_7	7	r	**Reserved for future use**

(table continues...)

(continued)

Field	Bits	Type	Description
GEN3_EQ_PSET_REQ_VEC	23:8	rwh	**Preset Request Vector** Requesting of Presets during the initial part of the EQ Master Phase. Encoding scheme is as follows: Bit [15:0] =0x0: No preset is requested and evaluated in EQ Master Phase. Bit [i] =1: "Preset=i" is requested and evaluated in EQ Master Phase. • 0000000000000000: No preset be requested and evaluated in EQ Master Phase • 000000xxxxxxxxx1: Preset 0 is requested and evaluated in EQ Master Phase • 000000xxxxxxxx1x: Preset 1 is requested and evaluated in EQ Master Phase • 000000xxxxxxx1xx: Preset 2 is requested and evaluated in EQ Master Phase • 000000xxxxxx1xxx: Preset 3 is requested and evaluated in EQ Master Phase • 000000xxxxx1xxxx: Preset 4 is requested and evaluated in EQ Master Phase • 000000xxxx1xxxxx: Preset 5 is requested and evaluated in EQ Master Phase • 000000xxx1xxxxxx: Preset 6 is requested and evaluated in EQ Master Phase • 000000xx1xxxxxxx: Preset 7 is requested and evaluated in EQ Master Phase • 000000x1xxxxxxxx: Preset 8 is requested and evaluated in EQ Master Phase • 00000x1xxxxxxxxx: Preset 9 is requested and evaluated in EQ Master Phase • 000001xxxxxxxxxx: Preset 10 is requested and evaluated in EQ Master Phase • All other encodings: Reserved Note: This register field is sticky.
GEN3_EQ_FOM_INC_INITIAL_EVAL	24	rwh	**Include Initial FOM** Include or not the FOM feedback from the initial preset evaluation performed in the EQ Master, when finding the highest FOM among all preset evaluations. Note: This register field is sticky. 0_B **DO_NOT_INCLUDE**: Do not include 1_B **INCLUDE**: Include
GEN3_EQ_PSET_REQ_AS_COEF	25	rwh	**GEN3_EQ_PSET_REQ_AS_COEF is an internally reserved field** Do not use. Note: This register field is sticky.

(table continues...)

(continued)

Field	Bits	Type	Description
GEN3_REQ_SEND_CONSEC_EIEOS_FOR_PSET_MAP	26	rwh	**Request controller to send back-to-back EIEOS in Recovery.RcvrLock state until presets to coefficients mapping is complete** Note: Gen3 and Gen4 share the same register bit and have the same feature. Note: This register field is sticky. 0_B **DO_NOT_REQ**: Do not request 1_B **REQ**: Request
RSVDP_27	31:27	r	**Reserved for future use**

19.4.2.268 Gen3 EQ Direction Change Feedback Mode Control Register

Equalization controls to be used in Phase2 (USP) or Phase 3 (DSP), when you set the Feedback Mode in "Gen3 EQ Control Register" to "Direction Change." These fields allow control over the initial starting point for the search of optimal coefficient settings, and allow control over the criteria used to determine when the optimal settings have been achieved. The values are applied to all the lanes.

GEN3_EQ_FB_MODE_DIR_CHANGE_OFF — Offset address: $00048AC_H$

Gen3 EQ Direction Change Feedback Mode Control Register — Kernel Reset value: $0000\ 0040_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_18														GEN3_EQ_FMDC_MAX_POST_CUSROR_DELTA	
r														rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GEN3_EQ_FMDC_MAX_POST_CUSROR_DELTA		GEN3_EQ_FMDC_MAX_PRE_CUSROR_DELTA				GEN3_EQ_FMDC_N_EVALS					GEN3_EQ_FMDC_T_MIN_PHASE23				
rw		rw				rw					rw				

Field	Bits	Type	Description
GEN3_EQ_FMDC_T_MIN_PHASE23	4:0	rw	**Minimum Time (in ms) To Remain in EQ Master Phase** The LTSSM stays in EQ Master phase for at least this amount of time, before starting to check for convergence of the coefficients. Allowed values 0,1,...,24. Note: This register field is sticky. 00_H **MIN_VAL**: Min value 18_H **MAX_VAL**: Max value **others,** Reserved

(table continues...)

(continued)

Field	Bits	Type	Description
GEN3_EQ_FMDC_N_EVALS	9:5	rw	**Convergence Window Depth** Number of consecutive evaluations considered in Phase 2/3 when determining if optimal coefficients have been found. Allowed range: 0,1,2,.. up to 8. When set to 0, EQ Master is performed without sending any requests to the remote partner in Phase 2 for USP and Phase 3 for DSP. Therefore, the remote partner will not change its transmitter coefficients and will move to the next state. Note: This register field is sticky. 00_H **MIN_VAL**: Min value $1F_H$ **MAX_VAL**: Maximum of CX_GEN3_EQ_COEFQ_DEPTH
GEN3_EQ_FMDC_MAX_PRE_CUSROR_DELTA	13:10	rw	**Convergence Window Aperture for C-1** Pre-cursor coefficients maximum delta within the convergence window depth. Allowed range: 0,1,2,..15. Note: This register field is sticky. 0_H **MIN_VAL**: Min value F_H **MAX_VAL**: Max value
GEN3_EQ_FMDC_MAX_POST_CUSROR_DELTA	17:14	rw	**Convergence Window Aperture for C+1** Post-cursor coefficients maximum delta within the convergence window depth. Allowed range: 0,1,2,..15. Note: This register field is sticky. 0_H **MIN_VAL**: Min value F_H **MAX_VAL**: Max value
RSVDP_18	31:18	r	**Reserved for future use**

19.4.2.269 Order Rule Control Register

This register controls the order rule.

ORDER_RULE_CTRL_OFF Offset address: $00048B4_H$

Order Rule Control Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_16															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CPL_PASS_P								NP_PASS_P							
rw								rw							

Field	Bits	Type	Description
NP_PASS_P	7:0	rw	**Non-Posted Passing Posted Ordering Rule Control** Determines if NP can pass halted P queue. 00_H **NP_CAN_NOT_PASS**: NP can not pass P (recommended) 01_H **NP_CAN_PASS**: NP can pass P **others,** Reserved

(table continues...)

(continued)

Field	Bits	Type	Description
CPL_PASS_P	15:8	rw	**Completion Passing Posted Ordering Rule Control** Determines if CPL can pass halted P queue. 00_H **CPL_CAN_NOT_PASS**: CPL can not pass P (recommended) 01_H **CPL_CAN_PASS**: CPL can pass P **others,** Reserved
RSVDP_16	31:16	r	**Reserved for future use**

19.4.2.270 PIPE Loopback Control Register

This register controls the PIPE Loopback.

PIPE_LOOPBACK_CONTROL_OFF Offset address: $00048B8_H$

PIPE Loopback Control Register Reset values see: Table 894

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PIPE_LOOPBACK	RSVDP_27				RXSTATUS_VALUE			RSVDP_22		RXSTATUS_LANE					
rw	r				rwh			r		rw					

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LPBK_RXVALID															
rw															

Field	Bits	Type	Description
LPBK_RXVALID	15:0	rw	**LPBK_RXVALID is an internally reserved field** Do not use. Note: This register field is sticky.
RXSTATUS_LANE	21:16	rw	**RXSTATUS_LANE is an internally reserved field** Do not use. Note: This register field is sticky.
RSVDP_22	23:22	r	**Reserved for future use**
RXSTATUS_VALUE	26:24	rwh	**RXSTATUS_VALUE is an internally reserved field** Do not use.
RSVDP_27	30:27	r	**Reserved for future use**
PIPE_LOOPBACK	31	rw	**PIPE Loopback Enable** Indicates RMMI Loopback if M-PCIe. Note: This register field is sticky.

Table 894 Reset values of PIPE_LOOPBACK_CONTROL_OFF

Reset	Reset value	Note
Kernel Reset	0000 0XXX 0000 0000 0000 0000 0000 0001_B	

19.4.2.271 DBI Read-Only Write Enable Register

This is the DBI Read-Only write enable register.

MISC_CONTROL_1_OFF Offset address: $00048BC_H$

DBI Read-Only Write Enable Register Kernel Reset value: $0047\ FF41_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_23									PORT_LOGIC_WR_DISABLE	P2P_ERR_RPT_CTRL	P2P_TRACK_CPL_TO_REG	TARGET_ABOVE_CONFIG_LIMIT_REG		CONFIG_LIMIT_REG	
r									rw	rw	rw	rw		rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CONFIG_LIMIT_REG								CFG_TLP_BYPASS_EN_REG	CPLQ_MNG_EN	ARI_DEVICE_NUMBER	DISABLE_AUTO_LTR_CLR_MSG	SIMPLIFIED_REPLAY_TIMER	UR_CA_MASK_4_TRGT1	DEFAULT_TARGET	DBI_RO_WR_EN
rw								rw	rw	rw	rw	rw	rw	rw	rwh

Field	Bits	Type	Description
DBI_RO_WR_EN	0	rwh	**Write to RO Registers Using DBI** This bit field protects the access to the registers to avoid intended modifications.. Note: This register field is sticky. 0_B **CLEAR**: Clear 1_B **SET**: Your application can write to some RO and HwInit register fields through the DBI when you set this field to '1'
DEFAULT_TARGET	1	rw	**Default target for an IO or MEM request with UR/CA/CRS received** Based on the value of this field the controller either drops or forwards these requests to your application. Note: This register field is sticky. 0_B **DROP_IO_MEM_REQ**: The controller drops all incoming I/O or MEM requests (after corresponding error reporting) A completion with UR status is generated for non-posted requests. 1_B **FWD_IO_MEM_UR_CA_CRS**: The controller forwards all incoming I/O or MEM requests with UR/CA/CRS status to your application
UR_CA_MASK_4_TRGT1	2	rw	**When this field is set to '1', the controller suppresses error logging, error message generation, and CPL generation for non-posted requests TLPs (with UR filtering status) forwarded to your application (that is, when DEFAULT_TARGET =1)** Note: This register field is sticky. 0_B **CLEAR**: Clear 1_B **SET**: Set

(table continues...)

(continued)

Field	Bits	Type	Description
SIMPLIFIED_REPLAY_TIMER	3	rw	**Enables Simplified Replay Timer (Gen4)** Simplified Replay Timer can have the following Values: • A value from 24,000 to 31,000 Symbol Times when Extended Synch is 0b • A value from 80,000 to 100,000 Symbol Times when Extended Synch is 1b The Simplified Replay Timer value must not be changed while the link is in use. Note: This register field is sticky. 0_B **DISABLE**: Disable 1_B **ENABLE**: Enable
DISABLE_AUTO_LTR_CLR_MSG	4	rw	**Disable the autonomous generation of LTR clear message in upstream port** This field can have the following values. Note: This register field is sticky. 0_B **ALLOW**: Allow the autonomous generation of LTR clear message 1_B **DISABLE**: Disable the autonomous generation of LTR clear message
ARI_DEVICE_NUMBER	5	rw	**When ARI is enabled, this field enables use of the device ID** Note: This register field is sticky. ARI capability is not available in this implementation 0_B **DISABLE**: Disable 1_B **ENABLE**: Enable
CPLQ_MNG_EN	6	rw	**This field enables the Completion Queue Management feature** Note: This register field is sticky. 0_B **DISABLE**: Disable 1_B **ENABLE**: Enable
CFG_TLP_BYPASS_EN_REG	7	rw	**Setting of this field defines how to decide the destination of Configuration requests** Note: When LNK_CORECTRL.RQRTYEN is asserted, the setting of this field is ignored. Note: This register field is sticky. 0_B **ZERO**: Configuration TLPs are routed according to the setting of TARGET_ABOVE_CONFIG_LIMIT_REG, depending on the setting of CONFIG_LIMIT_REG Refer to the definition of CONFIG_LIMIT_REG for details. 1_B **ONE**: Configuration TLPs are routed according to the setting of TARGET_ABOVE_CONFIG_LIMIT_REG, regardless the value of CONFIG_LIMIT_REG

(table continues...)

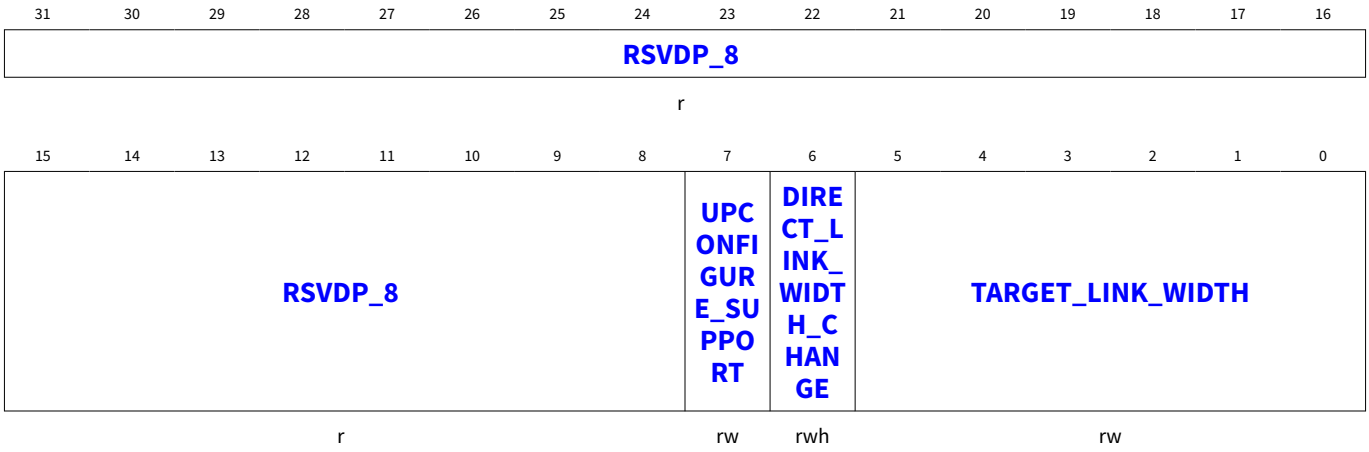
(continued)

Field	Bits	Type	Description
CONFIG_LIMIT_REG	17:8	rw	**Configuration requests are directed either to CDM or ELBI/RTRGT1 based on the value of this field** • Configuration requests with an address less CONFIG_LIMIT_REG are directed to the CDM • Configuration requests with an address greater than CONFIG_LIMIT_REG are directed to either ELBI or TRGT1 interface based on the setting of TARGET_ABOVE_CONFIG_LIMIT_REG field Your application must set a proper value for this field based on your extended configuration registers. Note: This register field is sticky.
TARGET_ABOVE_CONFIG_LIMIT_REG	19:18	rw	**Configuration requests with an address greater than CONFIG_LIMIT_REG are directed to either ELBI or TRGT1 interface based on the setting of this field** This field can have the following values: Note: This register field is sticky. 01_B **ELBI**: ELBI 10_B **TRGT1**: TRGT1 **others,** Reserved
P2P_TRACK_CPL_TO_REG	20	rw	**Determines whether to track completion of transmitted Non-Posted TLPs in P2P mode** Note: This register field is sticky. 0_B **DO_NOT_TRACk**: Do not track completion 1_B **TRACK**: Track completion
P2P_ERR_RPT_CTRL	21	rw	**Determines whether to enable Peer to Peer (P2P) error reporting** Note: This register field is sticky. 0_B **DISABLE**: Disable P2P error reporting 1_B **ENABLE**: Enable P2P error reporting
PORT_LOGIC_WR_DISABLE	22	rw	**Disable port logic register write from wire side** Note: The access attributes of this field are as follows: - Wire: R (sticky) - Dbi: R/W (sticky) Note: This register field is sticky.
RSVDP_23	31:23	r	**Reserved for future use**

19.4.2.272 UpConfigure Multi-lane Control Register

Used when upsizing or downsizing the link width through Configuration state without bringing the link down.

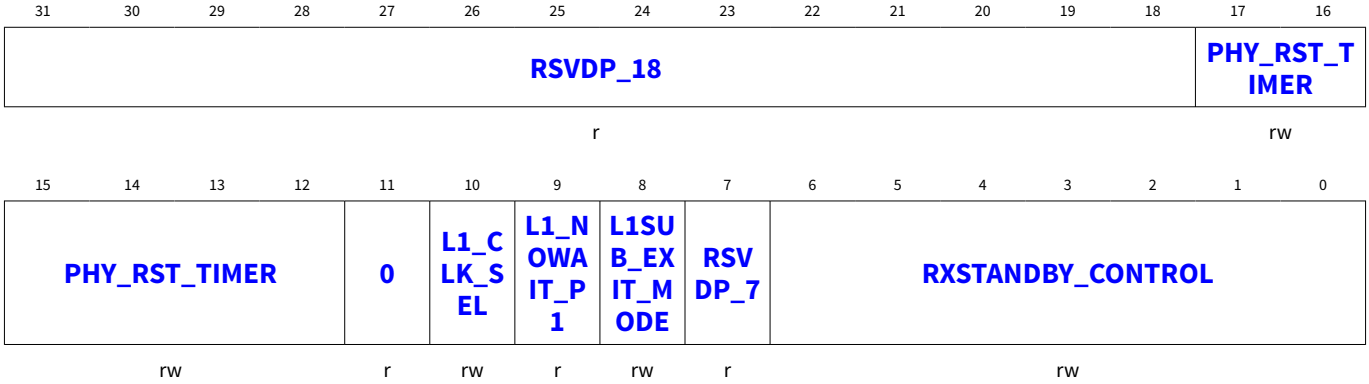
19 Peripheral Component Interconnect Express (PCIe)



Field	Bits	Type	Description
TARGET_LINK_WIDTH	5:0	rw	**Target Link Width** Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W 00_H **NOT_START**: Controller does not start upconfigure or autonomous width downsizing in the Configuration state 01_H **X1**: x1 02_H **X2**: x2 04_H **X4**: x4 08_H **X8**: x8 10_H **X16**: x16 20_H **X32**: x32 **others,** Reserved
DIRECT_LINK_WIDTH_CHANGE	6	rwh	**Directed Link Width Change** • If the upconfigure_capable variable is '1' and the PCIE_CAP_HW_AUTO_WIDTH_DISABLE bit in LINK_CONTROL_LINK_STATUS_REG is '0', the controller starts upconfigure or autonomous width downsizing (to the TARGET_LINK_WIDTH value) in the Configuration state • If TARGET_LINK_WIDTH value is 0x0, the controller does not start upconfigure or autonomous width downsizing in the Configuration state The controller self-clears this field when the controller accepts this request. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W 0_B **CLEAR**: Clear 1_B **SET**: The controller always moves to Configuration state through Recovery state when this bit is set to '1'
UPCONFIGURE_SUPPORT	7	rw	**Upconfigure Support** The controller sends this value as the Link Upconfigure Capability in TS2 Ordered Sets in Configuration.Complete state. Note: The access attributes of this field are as follows: - Wire: R/W (sticky) - Dbi: R/W (sticky) Note: This register field is sticky.
RSVDP_8	31:8	r	**Reserved for future use**

19.4.2.273 PHY Interoperability Control Register

This register controls the PHY interoperability.



Field	Bits	Type	Description
RXSTANDBY_CONTROL	6:0	rw	**Rxstandby Control** Bits 0..5 determine if the controller asserts the RxStandby signal (mac_phy_rxstandby) in the indicated condition. Bit 6 enables the controller to perform the RxStandby/RxStandbyStatus handshake. This field is reserved for internal use. You should not write to this field and change the default unless specifically instructed by Synopsys support. • [0]: Rx EIOS and subsequent T TX-IDLE-MIN • [1]: Rate Change • [2]: Inactive lane for upconfigure/downconfigure • [3]: PowerDown=P1orP2 • [4]: RxL0s.Idle • [5]: EI Infer in L0 • [6]: Execute RxStandby/RxStandbyStatus Handshake Note: This register field is sticky.
RSVDP_7	7	r	**Reserved for future use**
L1SUB_EXIT_MODE	8	rw	**L1 Exit Control Using phy_mac_pclkack_n** This field is reserved for internal use. You should not write to this field and change the default unless specifically instructed by Synopsys support. If PCLK as PHY input is selected, you should not write to this field and change the default value since PCLK as PHY input doesn't support any value other than the default. Note: This register field is sticky. 0_B **WAIT_FOR_PCLKACK**: Controller waits for the PHY to assert phy_mac_pclkack_n before exiting L1 1_B **NO_WAIT_FOR_PCLKACK**: Controller exits L1 without waiting for the PHY to assert phy_mac_pclkack_n

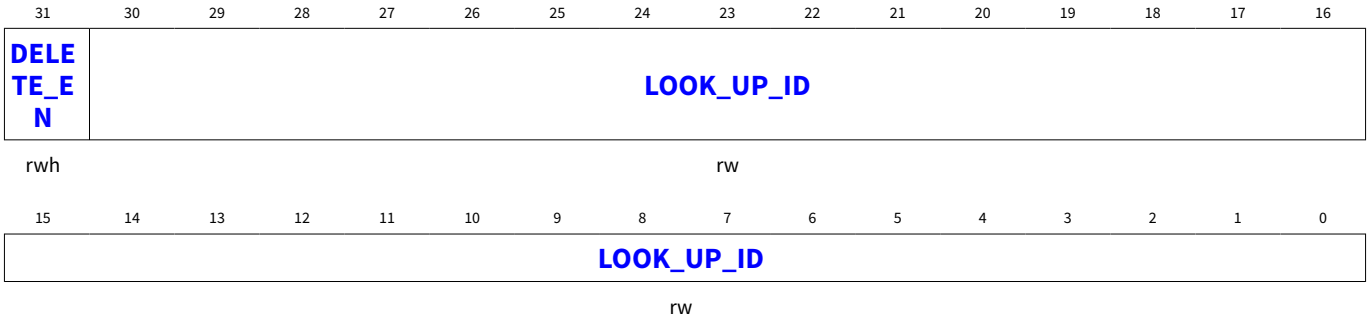
(table continues...)

(continued)

Field	Bits	Type	Description
L1_NOWAIT_P1	9	r	**L1 entry control bit** This field is reserved for internal use. You should not write to this field and change the default unless specifically instructed by Synopsys support. Note: The access attributes of this field are as follows: - Wire: R/W (sticky) - Dbi: R/W (sticky) Note: This register field is sticky. 0_B **WAIT_FOR_ACK**: Controller waits for the PHY to acknowledge transition to P1 before entering L1 1_B **NO_WAIT_FOR_ACK**: Controller does not wait for PHY to acknowledge transition to P1 before entering L1
L1_CLK_SEL	10	rw	**L1 Clock control bit** This field is reserved for internal use. You should not write to this field and change the default unless specifically instructed by Synopsys support. Note: This register field is sticky. 0_B **CAN_REQ**: Controller requests aux_clk switch and core_clk gating in L1 1_B **NO_REQ**: Controller does not request aux_clk switch and core_clk gating in L1
PHY_RST_TIMER	17:12	rw	**Decide how many aux clock cycles the PHY reset lasts (0 to 63 aux clock cycles)** Note: This register field is sticky. 00_H **MIN_VAL**: Zero value $3F_H$ **MAX_VAL**: Max value
RSVDP_18	31:18	r	**Reserved for future use**
0	11	r	**Reserved** Read as 0; should be written with 0.

19.4.2.274 TRGT_CPL_LUT Delete Entry Control register

Using this register you can delete one entry in the target completion LUT. You should only use this register when you know that your application will never send the completion because of an FLR or any other reason. Note:: The target completion LUT (and associated target completion timeout event) is watching for completions (from your application on XALI0/1/2 or AXI master read channel) corresponding to previously received non-posted requests from the PCIe wire.



Field	Bits	Type	Description
LOOK_UP_ID	30:0	rw	**This number selects one entry to delete of the TRGT_CPL_LUT**

(table continues...)

(continued)

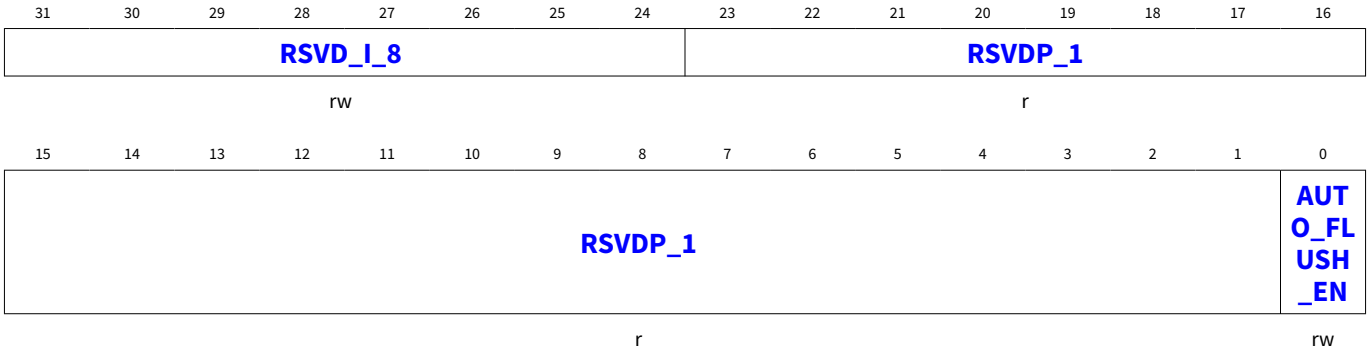
Field	Bits	Type	Description
DELETE_EN	31	rwh	**This is a one-shot bit** This is a self-clearing register field. Reading from this register field always returns a '0'. 0_B **CLEAR**: Clear 1_B **SET**: A '1' write to this bit triggers the deletion of the target completion LUT entry that is specified in the LOOK_UP_ID field

19.4.2.275 Link Reset Request Flush Control Register

This register controls link reset request flush behaviour.

LINK_FLUSH_CONTROL_OFF Offset address: $00048CC_H$

Link Reset Request Flush Control Register Kernel Reset value: $FF00\ 0001_H$



Field	Bits	Type	Description
AUTO_FLUSH_EN	0	rw	**Enables automatic flushing of pending requests before sending the reset request to the application logic to reset the PCIe controller and the AXI Bridge** The flushing process is initiated if any of the following events occur: • Hot reset request. A downstream port (DSP) can "hot reset" an upstream port (USP) by sending two consecutive TS1 ordered sets with the hot reset bit asserted • Warm (Soft) reset request. Generated when exiting from D3 to D0 and CFG_INFO.NOSOFTRST=0 • Link down reset request. A high to low transition on LNK_STAT.LINKRQRST indicates the link has gone down and the controller is requesting a reset If you disable automatic flushing, your application is responsible for resetting the PCIe controller and the AXI Bridge. Note: This register field is sticky. 0_B **DISABLE**: Disable 1_B **ENABLE**: Enable
RSVDP_1	23:1	r	**Reserved for future use**
RSVD_I_8	31:24	rw	**This is an internally reserved field** Do not use. Note: This register field is sticky.

19.4.2.276 AXI Bridge Slave Error Response Register

This register holds the AXI bridge slave error responses.

AMBA_ERROR_RESPONSE_DEFAULT_OFF Offset address: 00048D0$_H$

AXI Bridge Slave Error Response Register Kernel Reset value: 0000 9C00$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_16															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AMBA_ERROR_RESPONSE_MAP						RSVDP_5					AMBA_ERROR_RESPONSE_CRS		AMBA_ERROR_RESPONSE_VENDORID	RSVDP_1	AMBA_ERROR_RESPONSE_GLOBAL
rw						r					rw		rw	r	rw

Field	Bits	Type	Description
AMBA_ERROR_RESPONSE_GLOBAL	0	rw	**Global Slave Error Response Mapping** Determines the AXI slave response for all error scenarios on non-posted requests. Note: This register field is sticky. 0_B **OK_**: OKAY (with FFFF data for non-posted requests) 1_B **ERR_**: SLVERR/DECERR (the AMBA_ERROR_RESPONSE_MAP field determines the PCIe-to-AXI Slave error response mapping)
RSVDP_1	1	r	**Reserved for future use**
AMBA_ERROR_RESPONSE_VENDORID	2	rw	**Vendor ID Non-existent Slave Error Response Mapping** Determines the AXI slave response for errors on reads to non-existent Vendor ID register. Note: This register field is sticky. 0_B **_OK_**: OKAY (with FFFF data) 1_B **_ERR_**: SLVERR/DECERR (the AMBA_ERROR_RESPONSE_MAP field determines the PCIe-to-AXI Slave error response mapping)
AMBA_ERROR_RESPONSE_CRS	4:3	rw	**CRS Slave Error Response Mapping** Determines the AXI slave response for CRS completions. Note: This register field is sticky. 00_B **OK_**: OKAY 01_B **OK_CRS_CMPL**: OKAY with all FFFF_FFFF data for all CRS completions 10_B **OK_READ_REQ**: OKAY with FFFF_0001 data for CRS completions to vendor ID read requests, OKAY with FFFF_FFFF data for all other CRS completions 11_B **SLVERR_DECERR**: SLVERR/DECERR (the AMBA_ERROR_RESPONSE_MAP field determines the PCIe-to-AXI Slave error response mapping)
RSVDP_5	9:5	r	**Reserved for future use**

(table continues...)

(continued)

Field	Bits	Type	Description
AMBA_ERROR_RESPONSE_MAP	15:10	rw	**AXI Slave Response Error Map** Allows you to selectively map the errors received from the PCIe completion (for non-posted requests) to the AXI slave responses, slv_rresp or slv_bresp. The recommended setting is SLVERR. CRS is always mapped to OKAY. • [0] -- 0: UR (unsupported request) -> DECERR -- 1: UR (unsupported request) -> SLVERR • [1] -- 0: CRS (configuration retry status) -> DECERR -- 1: CRS (configuration retry status) -> SLVERR • [2] -- 0: CA (completer abort) -> DECERR -- 1: CA (completer abort) -> SLVERR • [3]: RESERVED (0x0) • [4]: RESERVED (0x0) • [5] -- 0: Completion Timeout -> DECERR -- 1: Completion Timeout -> SLVERR. The AXI bridge internally drops (processes internally but not passed to your application) a completion that has been marked by the Rx filter as UC or MLF, and does not pass its status directly down to the slave interface. It waits for a timeout and then signals "Completion Timeout" to the slave interface. The controller sets the AXI slave read databus to 0xFFFF for all error responses. Note: This register field is sticky.
RSVDP_16	31:16	r	**Reserved for future use**

19.4.2.277 Link Down AXI Bridge Slave Timeout Register

If your application issues outbound requests to the AXI bridge slave interface before the PCIe link is operational, the controller starts a "flush" timer. The timeout value of the timer is set by this register. If the timer times out before the PCIe link is operational, the bridge TX request queues are flushed.

AMBA_LINK_TIMEOUT_OFF Offset address: $00048D4_H$

Link Down AXI Bridge Slave Timeout Register Kernel Reset value: $0000\ 0032_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_9															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVDP_9							LINK_TIMEOUT_ENABLE_DEFAULT	LINK_TIMEOUT_PERIOD_DEFAULT							
r							rw	rw							

Field	Bits	Type	Description
LINK_TIMEOUT_PERIOD_DEFAULT	7:0	rw	**Timeout Value (ms)** The timer will timeout and then flush the bridge TX request queues after this amount of time. The timer counts when there are pending outbound AXI slave interface requests and the PCIe TX link is not transmitting any of these requests. The timer is clocked by core_clk. Note: This register field is sticky.
LINK_TIMEOUT_ENABLE_DEFAULT	8	rw	**Disable Flush** Note: This register field is sticky. 0_B **ENABLE**: Enable 1_B **DISABLE**: You can disable the flush feature by setting this field to '1'
RSVDP_9	31:9	r	**Reserved for future use**

19.4.2.278 AXI Bridge Ordering Control

This register controls the AXI Bridge Ordering.

AMBA_ORDERING_CTRL_OFF Offset address: $00048D8_H$

AXI Bridge Ordering Control Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_8															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVDP_8								AX_MSTR_ZEROLREAD_FW	RSVDP_5		AX_MSTR_ORDR_P_EVENT_SEL		RSVDP_2	AX_SNP_EN	RSVDP_0
r								rw	r		rw		r	rw	r

Field	Bits	Type	Description
RSVDP_0	0	r	**Reserved for future use**
AX_SNP_EN	1	rw	**AXI Serialize Non-Posted Requests Enable** This field enables the AXI Bridge to serialize same ID Non-Posted Read/Write Requests on the wire. Serialization implies one outstanding same ID NP Read or Write on the wire and used to avoid AXI RAR and WAW hazards at the remote link partner. 0_B **DISABLE**: Disable 1_B **ENABLE**: Enable
RSVDP_2	2	r	**Reserved for future use**

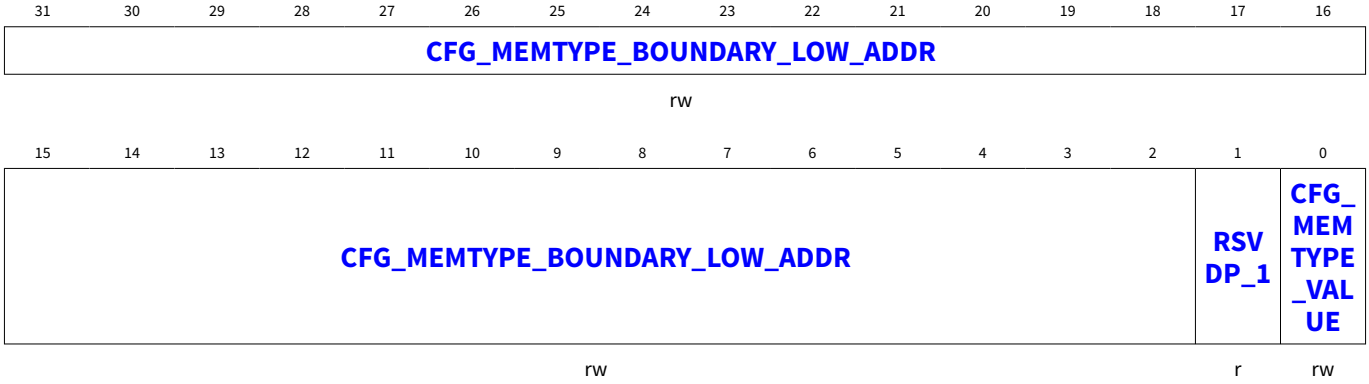
(table continues...)

(continued)

Field	Bits	Type	Description
AX_MSTR_ORDR_P_EVENT_SEL	4:3	rw	**AXI Master Posted Ordering Event Selector** This field selects how the master interface determines when a P write is completed when enforcing the PCIe ordering rule, "NP must not pass P" at the AXI Master Interface. The AXI protocol does not support ordering between channels. Therefore, NP reads can pass P on your AXI bus fabric. This can result in an ordering violation when the read overtakes a P that is going to the same address. Therefore, the bridge master does not issue any NP requests until all outstanding P writes reach their destination. It does this by waiting for the all of the write responses on the B channel. This can affect the performance of the master read channel. For scenarios where the interconnect serializes the AXI master "AW", "W" and "AR" channels,you can increase the performance by reducing the need to wait until the complete Posted transaction has effectively reached the application slave. Note: This setting will not affect: • MSI interrupt catcher and P data ordering. This is always driven by the B'last event • DMA read engine TLP ordering. This is always driven by the B'last event • NP write transactions which are always serialized with P write transactions 00_B **B**: B'last event: wait for the all of the write responses on the B channel thereby ensuring that the complete Posted transaction has effectively reached the application slave (default) 01_B **AW**: AW'last event: wait until the complete Posted transaction has left the AXI address channel at the bridge master 10_B **W**: W'last event: wait until the complete Posted transaction has left the AXI data channel at the bridge master 11_B **RSVD**: Reserved
RSVDP_5	6:5	r	**Reserved for future use**
AX_MSTR_ZEROLREAD_FW	7	rw	**AXI Master Zero Length Read Forward to the application** The AXI bridge is able to terminate in order with the Posted transactions the zero length read, implementing the PCIe express flush semantics of the Posted transactions. 0_B **_0_LN_RD_TERMINATE**: The zero length Read is terminated at the DW PCIe AXI bridge master 1_B **_0_LN_RD_FWD**: The zero length Read is forward to the application
RSVDP_8	31:8	r	**Reserved for future use**

19.4.2.279 ACE Cache Coherency Control Register 1

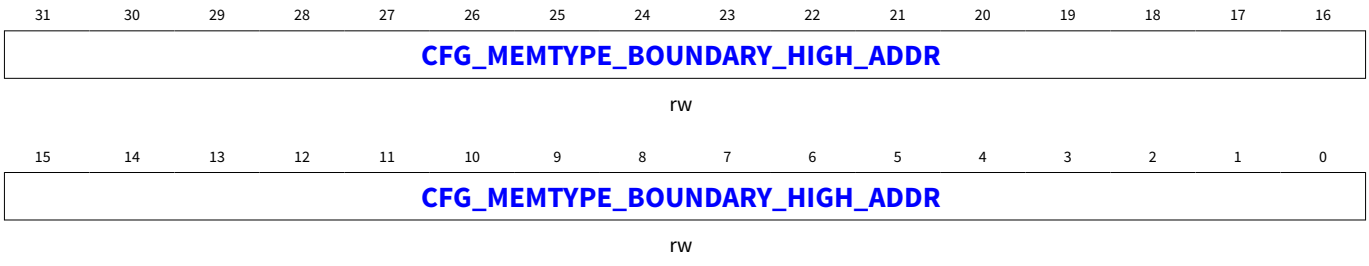
This register controls the ACE cache coherency operation.



Field	Bits	Type	Description
CFG_MEMTYPE_VALUE	0	rw	**Sets the memory type for the lower and upper parts of the address space:** Note: This register field is sticky. 0_B **LOWER_PREPH**: lower = Peripheral; upper = Memory 1_B **LOWER_MEM**: lower = Memory type; upper = Peripheral
RSVDP_1	1	r	**Reserved for future use**
CFG_MEMTYPE_BOUNDARY_LOW_ADDR	31:2	rw	**Boundary Lower Address For Memory Type** Bits [31:0] of dword-aligned address of the boundary for Memory type. The two lower address LSBs are '00'. Addresses up to but not including this value are in the lower address space region; addresses equal or greater than this value are in the upper address space region. Note: This register field is sticky.

19.4.2.280 ACE Cache Coherency Control Register 2

This register controls the ACE cache coherency operation.



Field	Bits	Type	Description
CFG_MEMTYPE_BOUNDARY_HIGH_ADDR	31:0	rw	**Boundary Upper Address For Memory Type** Bits [63:32] of the 64-bit dword-aligned address of the boundary for Memory type. Note: This register field is sticky.

19.4.2.281 ACE Cache Coherency Control Register 3

This register controls the ACE cache coherency operation.

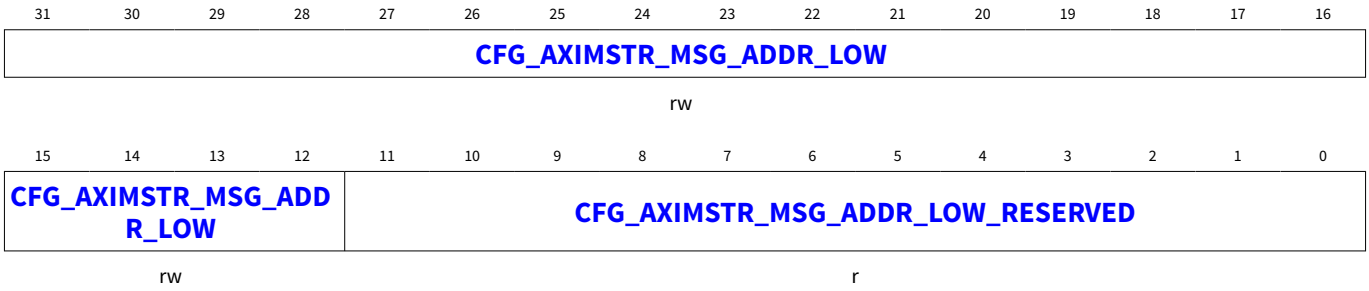
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	CFG_MSTR_AWCACHE_VALUE				0				CFG_MSTR_ARCACHE_VALUE				0		
r	rw				r				rw				r		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	CFG_MSTR_AWCACHE_MODE				0				CFG_MSTR_ARCACHE_MODE				0		
r	rw				r				rw				r		

Field	Bits	Type	Description
CFG_MSTR_ARCACHE_MODE	6:3	rw	**Master Read CACHE Signal Behavior** Defines how the individual bits in mstr_arcache are controlled. Note: This register field is sticky. 0_H **SET_BY_AXI_MASTER**: set automatically by the AXI master 1_H **SET_BY_FIELD**: set by the value of the corresponding bit of the CFG_MSTR_ARCACHE_VALUE field **others,** Reserved
CFG_MSTR_AWCACHE_MODE	14:11	rw	**Master Write CACHE Signal Behavior** Defines how the individual bits in mstr_awcache are controlled. Note: for message requests the value of mstr_awcache is always "0000" regardless of the value of this bit. Note: This register field is sticky. 0_H **SET_BY_AXI_MASTER**: set automatically by the AXI master 1_H **SET_BY_FIELD**: set by the value of the corresponding bit of the CFG_MSTR_AWCACHE_VALUE field **others,** Reserved
CFG_MSTR_ARCACHE_VALUE	22:19	rw	**Master Read CACHE Signal Value** Value of the individual bits in mstr_arcache when CFG_MSTR_ARCACHE_MODE is '1'. Note: This register field is sticky.
CFG_MSTR_AWCACHE_VALUE	30:27	rw	**Master Write CACHE Signal Value** Value of the individual bits in mstr_awcache when CFG_MSTR_AWCACHE_MODE is '1'. Note: Not applicable to message requests; for message requests the value of mstr_awcache is always '0000'. Note: This register field is sticky.
0	2:0, 10:7, 18:15, 26:23, 31	r	**Reserved** Read as all 0's; should be written with all 0's.

19.4.2.282 Lower 32-bits of the Programmable AXI Address

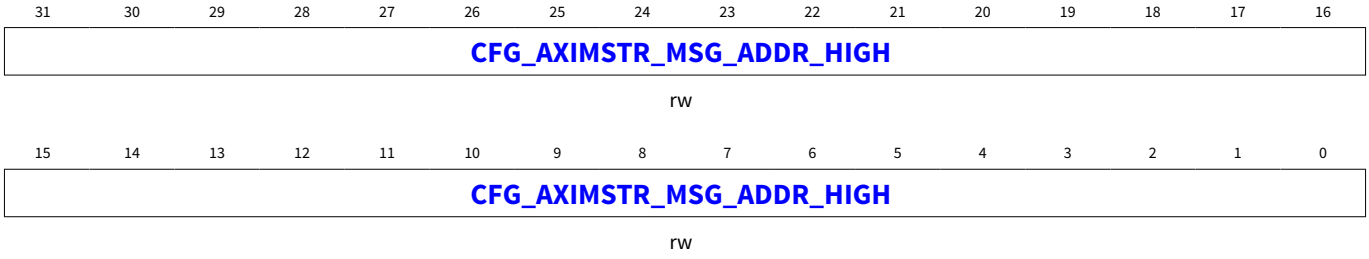
Lower 20 bits of the programmable AXI address to which Messages coming from wire are mapped. Bits [11:0] of the register are tied to zero for the address to be 4k-aligned. In previous releases, the third and fourth DWORDs of a message (Msg/MsgD) TLP header were delivered though the AXI master address bus (mstr_awaddr). These DWORDS are now supplied through the mstr_awmisc_info_hdr_34dw[63:0] output; and the value on mstr_awaddr is driven to the value you have programmed into the AXI_MSTR_MSG_ADDR_LOW_OFF and AXI_MSTR_MSG_ADDR_HIGH_OFF registers.



Field	Bits	Type	Description
CFG_AXIMSTR_MSG_ADDR_LOW_RESERVED	11:0	r	**Reserved for future use** Note: This register field is sticky.
CFG_AXIMSTR_MSG_ADDR_LOW	31:12	rw	**Lower 20-bits of the programmable AXI address for Messages** Note: This register field is sticky.

19.4.2.283 Upper 32-bits of the Programmable AXI Address

Upper 32 bits of the programmable AXI address to which Messages coming from wire are mapped.



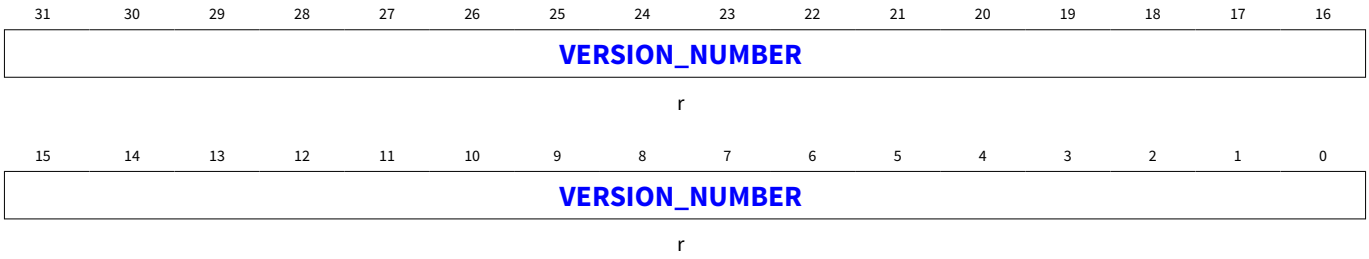
Field	Bits	Type	Description
CFG_AXIMSTR_MSG_ADDR_HIGH	31:0	rw	**Upper 32 bits of the programmable AXI address for Messages** Note: This register field is sticky.

19.4.2.284 PCIe Controller IIP Release Version Number

The version number is given in hex format. You should convert each pair of hex characters to ASCII to interpret. Using 4.70a (GA) as an example: - VERSION_NUMBER = 0x3437302a which translates to 470* - VERSION_TYPE = 0x67612a2a which translates to ga** Using 4.70a-ea01 as an example: - VERSION_NUMBER = 0x3437302a which translates to 470* - VERSION_TYPE = 0x65613031 which translates to ea01 GA is a general release available on www.designware.com EA is an early release available on a per-customer basis.

PCIE_VERSION_NUMBER_OFF Offset address: $00048F8_H$

PCIe Controller IIP Release Version Number Kernel Reset value: $3537\ 312A_H$



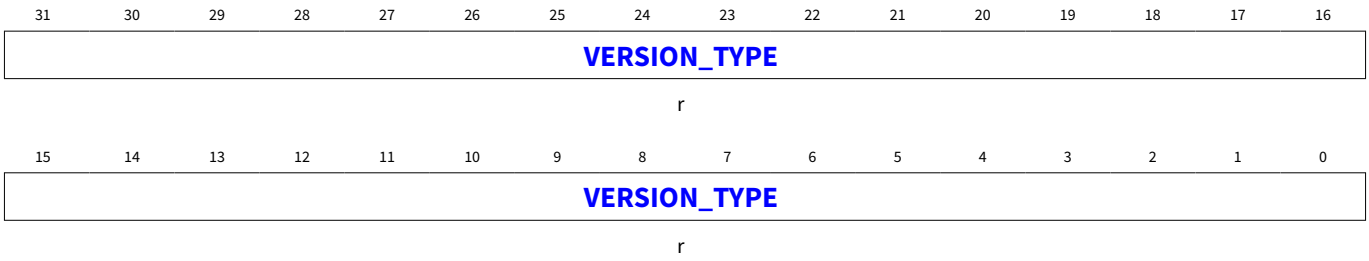
Field	Bits	Type	Description
VERSION_NUMBER	31:0	r	**Version Number** 00000000_H **MIN_VAL**: Zero value $FFFFFFFF_H$ **MAX_VAL**: Max value

19.4.2.285 PCIe Controller IIP Release Version Type

The type is given in hex format. You should convert each pair of hex characters to ASCII to interpret. Using 4.70a (GA) as an example: - VERSION_NUMBER = 0x3437302a which translates to 470* - VERSION_TYPE = 0x67612a2a which translates to ga** Using 4.70a-ea01 as an example: - VERSION_NUMBER = 0x3437302a which translates to 470* - VERSION_TYPE = 0x65613031 which translates to ea01 GA is a general release available on www.designware.com EA is an early release available on a per-customer basis.

PCIE_VERSION_TYPE_OFF Offset address: $00048FC_H$

PCIe Controller IIP Release Version Type Kernel Reset value: $6C70\ 3031_H$



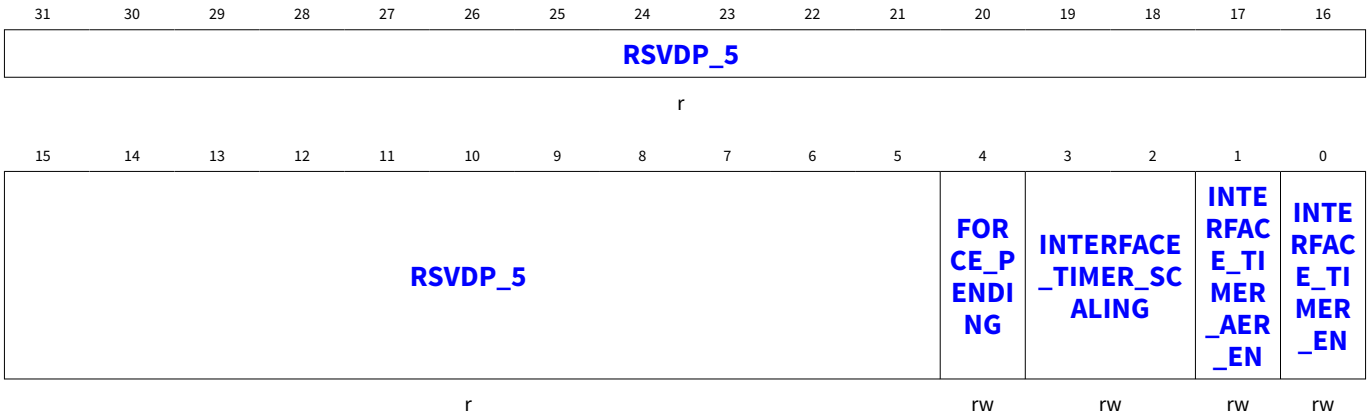
Field	Bits	Type	Description
VERSION_TYPE	31:0	r	**Version Type** 00000000_H **MIN_VAL**: Zero value $FFFFFFFF_H$ **MAX_VAL**: Max value

19.4.2.286 Interface Timer Control Register

This register controls the interface timer.

INTERFACE_TIMER_CONTROL_OFF Offset address: 0004930_H

Interface Timer Control Register Kernel Reset value: $0000\ 0000_H$



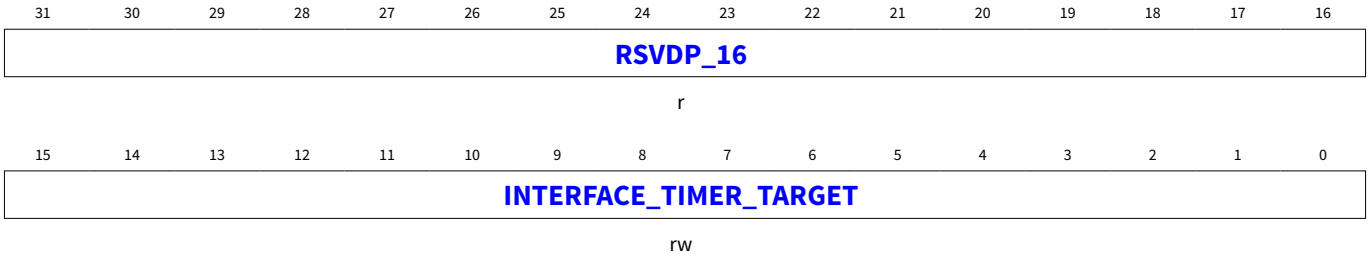
Field	Bits	Type	Description
INTERFACE_TIMER_EN	0	rw	**Interface timer enable** Note: This register field is sticky.
INTERFACE_TIMER_AER_EN	1	rw	**When set to 1 the Interface timer internal uncorrectable error generation is enabled** Note: This register field is sticky. 0_B **CLEAR**: Clear 1_B **SET**: The Interface timer internal uncorrectable error generation is enabled
INTERFACE_TIMER_SCALING	3:2	rw	**Interface timer scaling** This field can be used to reduce the timer duration. This field should only be programmed when the INTERFACE_TIMER_EN bit is set to 1'b0. Note: This register field is sticky.
FORCE_PENDING	4	rw	**Writing to this bit forces the value of the pending flags** Note: This register field is sticky.
RSVDP_5	31:5	r	**Reserved for future use**

19.4.2.287 Interface Timer Target Register

This is the interface timer target register.

INTERFACE_TIMER_TARGET_OFF Offset address: 0004934_H

Interface Timer Target Register Kernel Reset value: $0000\ 0032_H$



Field	Bits	Type	Description
INTERFACE_TIMER_TARGET	15:0	rw	**Interface timer target value** This field should only be programmed when the INTERFACE_TIMER_EN bit is set to 1'b0. Note: This register field is sticky.
RSVDP_16	31:16	r	**Reserved for future use**

19.4.2.288 Interface Timer Status Register

This is the interface timer status register.

INTERFACE_TIMER_STATUS_OFF Offset address: 0004938_H

Interface Timer Status Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				SLAVE_RD_ADD_TIMEOUT	SLAVE_WR_DATA_TIMEOUT	SLAVE_WR_ADD_TIMEOUT	0		MASTER_RD_DATA_TIMEOUT	MASTER_WR_RES_TIMEOUT	CLIENT2_INTERFACE_TIMEOUT	CLIENT1_INTERFACE_TIMEOUT	0	CPL_INTERFACE_TIMEOUT	MESSAGE_INTERFACE_TIMEOUT
r				rw1ch	rw1ch	rw1ch	r		rw1ch	rw1ch	rw1ch	rw1ch	r	rw1ch	rw1ch

Field	Bits	Type	Description
MESSAGE_INTERFACE_TIMEOUT	0	rw1ch	**Message interface timeout** Note: This register field is sticky. Software write with 1 clears this bit-field.
CPL_INTERFACE_TIMEOUT	1	rw1ch	**CPL interface timeout** Note: This register field is sticky. Software write with 1 clears this bit-field.
CLIENT1_INTERFACE_TIMEOUT	3	rw1ch	**Client1 interface (XALI1) timeout** Note: This register field is sticky. Software write with 1 clears this bit-field.
CLIENT2_INTERFACE_TIMEOUT	4	rw1ch	**Client2 interface (XALI2) timeout** Note: This register field is sticky. Software write with 1 clears this bit-field.
MASTER_WR_RES_TIMEOUT	5	rw1ch	**Master write response channel timeout** Note: This register field is sticky. Software write with 1 clears this bit-field.
MASTER_RD_DATA_TIMEOUT	6	rw1ch	**Master read data channel timeout** Note: This register field is sticky. Software write with 1 clears this bit-field.

(table continues...)

(continued)

Field	Bits	Type	Description
SLAVE_WR_ADD_TIMEOUT	9	rw1ch	**Slave write address channel timeout** Note: This register field is sticky. Software write with 1 clears this bit-field.
SLAVE_WR_DATA_TIMEOUT	10	rw1ch	**Slave write data channel timeout** Note: This register field is sticky. Software write with 1 clears this bit-field.
SLAVE_RD_ADD_TIMEOUT	11	rw1ch	**Slave read address channel timeout** Note: This register field is sticky. Software write with 1 clears this bit-field.
0	2, 8:7, 31:12	r	**Reserved** Read as 0; should be written with 0.

19.4.2.289 MSI-X Address Match Low Register

This feature is not supported in this hardware configuration. When your local AXI application writes (MWr) to the address defined in this register (and MSIX_ADDRESS_MATCH_HIGH_OFF), the controller will load the MSIX_DOORBELL_OFF register with the contents of the MWr and subsequently create and send MSI-X TLPs

MSIX_ADDRESS_MATCH_LOW_OFF Offset address: 0004940_H

MSI-X Address Match Low Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MSIX_ADDRESS_MATCH_LOW															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MSIX_ADDRESS_MATCH_LOW														MSIX_ADDRESS_MATCH_RESERVED_1	MSIX_ADDRESS_MATCH_EN
rw														r	rw

Field	Bits	Type	Description
MSIX_ADDRESS_MATCH_EN	0	rw	**MSI-X Match Enable** Enable the MSI-X Address Match feature when the AXI bridge is present. Note: This register field is sticky. 0_B **DISABLE**: Disable 1_B **ENABLE**: Enable
MSIX_ADDRESS_MATCH_RESERVED_1	1	r	**Reserved** Note: This register field is sticky.

(table continues...)

(continued)

Field	Bits	Type	Description
MSIX_ADDRESS_MATCH_LOW	31:2	rw	**MSI-X Address Match Low Address** Note: This register field is sticky.

19.4.2.290 MSIX Address Match High Register

This feature is not supported in this hardware configuration. When your local AXI application writes (MWr) to the address defined in this register (and MSIX_ADDRESS_MATCH_LOW_OFF), the controller will load the MSIX_DOORBELL_OFF register with the contents of the MWr and subsequently create and send MSI-X TLPs

MSIX_ADDRESS_MATCH_HIGH_OFF Offset address: 0004944_H

MSIX Address Match High Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MSIX_ADDRESS_MATCH_HIGH															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MSIX_ADDRESS_MATCH_HIGH															
rw															

Field	Bits	Type	Description
MSIX_ADDRESS_MATCH_HIGH	31:0	rw	**MSI-X Address Match High Address** Note: This register field is sticky. 00000000_H **MIN_VAL**: Zero value $FFFFFFFF_H$ **MAX_VAL**: Max value

19.4.2.291 MSI-X Doorbell Register

When your local application writes to this register, the controller will create and send MSI-X TLPs.

MSIX_DOORBELL_OFF Offset address: 0004948_H

MSI-X Doorbell Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MSIX_DOORBELL_RESERVED_29_31			MSIX_DOORBELL_PF					MSIX_DOORBELL_VF							
w			w					w							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MSIX_DOORBELL_VF_ACTIVE	MSIX_DOORBELL_TC			MSIX_DOORBELL_RESERVED_11	MSIX_DOORBELL_VECTOR										
w	w			w	w										

Field	Bits	Type	Description
MSIX_DOORBELL_VECTOR	10:0	w	**MSI-X Doorbell Vector** This register determines which vector to generate the MSI-X transaction for.
MSIX_DOORBELL_RESERVED_11	11	w	**Reserved**
MSIX_DOORBELL_TC	14:12	w	**MSIX Doorbell Traffic Class** This register determines which traffic class to generate the MSI-X transaction with.
MSIX_DOORBELL_VF_ACTIVE	15	w	**MSIX Doorbell Virtual Function Active** This register determines whether a Virtual Function is used to generate the MSI-X transaction.
MSIX_DOORBELL_VF	23:16	w	**MSIX Doorbell Virtual Function** This register determines the Virtual Function for the MSI-X transaction.
MSIX_DOORBELL_PF	28:24	w	**MSIX Doorbell Physical Function** This register determines the Physical Function for the MSI-X transaction.
MSIX_DOORBELL_RESERVED_29_31	31:29	w	**Reserved**

19.4.2.292 MSI-X RAM power mode and debug control register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MSIX_RAM_CTRL_RESERVED_26_31						MSIX_RAM_CTRL_DBG_PBA	MSIX_RAM_CTRL_DBG_TABLE	MSIX_RAM_CTRL_RESERVED_17_23							MSIX_RAM_CTRL_BYPASS
r						rw	rw	r							rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MSIX_RAM_CTRL_RESERVED_10_15						MSIX_RAM_CTRL_PBA_SD	MSIX_RAM_CTRL_PBA_DS	MSIX_RAM_CTRL_RESERVED_2_7						MSIX_RAM_CTRL_TABLE_SD	MSIX_RAM_CTRL_TABLE_DS
r						rw	rw	r						rw	rw

Field	Bits	Type	Description
MSIX_RAM_CTRL_TABLE_DS	0	rw	**MSIX Table RAM Deep Sleep** Note: This register field is sticky. 0_B **CLEAR**: Clear 1_B **SET**: Set this bit to drive the cfg_msix_table_ds output to signal your external logic to place the MSIX Table RAM in Deep Sleep low-power mode
MSIX_RAM_CTRL_TABLE_SD	1	rw	**MSIX Table RAM Shut Down** Note: This register field is sticky. 0_B **CLEAR**: Clear 1_B **SET**: Set this bit to drive the cfg_msix_table_sd output to signal your external logic to place the MSIX Table RAM in Shut Down low-power mode
MSIX_RAM_CTRL_RESERVED_2_7	7:2	r	**Reserved** Note: This register field is sticky.
MSIX_RAM_CTRL_PBA_DS	8	rw	**MSIX PBA RAM Deep Sleep** Note: This register field is sticky. 0_B **CLEAR**: Clear 1_B **SET**: Set this bit to drive the cfg_msix_pba_ds output to signal your external logic to place the MSIX PBA RAM in Deep Sleep low-power mode
MSIX_RAM_CTRL_PBA_SD	9	rw	**MSIX PBA RAM Shut Down** Note: This register field is sticky. 0_B **CLEAR**: Clear 1_B **SET**: Set this bit to drive the cfg_msix_pba_sd output to signal your external logic to place the MSIX PBA RAM in Shut Down low-power mode
MSIX_RAM_CTRL_RESERVED_10_15	15:10	r	**Reserved** Note: This register field is sticky.
MSIX_RAM_CTRL_BYPASS	16	rw	**MSIX RAM Control Bypass** It is up to the application to ensure the RAMs are in the proper power state before trying to access them. Moreover, the application needs to observe all timing requirements of the RAM low power signals before trying to use the MSIX functionality. Note: This register field is sticky. 0_B **CLEAR**: Clear 1_B **SET**: The bypass field, when set, disables the internal generation of low power signals for both RAMs
MSIX_RAM_CTRL_RESERVED_17_23	23:17	r	**Reserved** Note: This register field is sticky.

(table continues...)

(continued)

Field	Bits	Type	Description
MSIX_RAM_CTRL_DBG_TABLE	24	rw	**MSIX Table RAM Debug Mode** You can also use the dbg_table input to activate debug mode. Debug mode turns off the PF/VF/Offset-based addressing into the RAM and maps the entire table linearly from the base address of the BAR (indicated by the BIR) in function 0. Note: This register field is sticky. 0_B **DISABLE**: Disable 1_B **ENABLE**: Enable Use this bit to activate the debug mode and allow direct read/write access to the Table.
MSIX_RAM_CTRL_DBG_PBA	25	rw	**MSIX PBA RAM Debug Mode** You can also use the dbg_pba input to activate debug mode. Debug mode turns off the PF/VF/Offset-based addressing into the RAM and maps the entire table linearly from the base address of the BAR (indicated by the BIR) in function 0. Note: This register field is sticky. 0_B **DISABLE**: Disable 1_B **ENABLE**: Enable Use this bit to activate the debug mode and allow direct read/write access to the PBA.
MSIX_RAM_CTRL_RESERVED_26_31	31:26	r	**Reserved** Note: This register field is sticky.

19.4.2.293 Functional Safety Interrupt Events Mask Register

This registers holds the masks for functional safety interrupt events.

SAFETY_MASK_OFF Offset address: 0004960_H

Functional Safety Interrupt Events Mask Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_6															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVDP_6										SAFETY_INT_MASK_5	SAFETY_INT_MASK_4	SAFETY_INT_MASK_3	SAFETY_INT_MASK_2	SAFETY_INT_MASK_1	SAFETY_INT_MASK_0
r										rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
SAFETY_INT_MASK_0	0	rw	**Mask for functional safety interrupt event 0 (RASDP)** Note: This register field is sticky.
SAFETY_INT_MASK_1	1	rw	**Mask for functional safety interrupt event 1 (Interface timers)** Note: This register field is sticky.

(table continues...)

(continued)

Field	Bits	Type	Description
SAFETY_INT_MASK_2	2	rw	**Mask for functional safety interrupt event 2 (CDM register checker)** Note: This register field is sticky.
SAFETY_INT_MASK_3	3	rw	**Mask for functional safety interrupt event 3 (PCIe uncorrectable)** Note: This register field is sticky.
SAFETY_INT_MASK_4	4	rw	**Mask for functional safety interrupt event 4 (PCIe correctable)** Note: This register field is sticky.
SAFETY_INT_MASK_5	5	rw	**Mask for functional safety interrupt event 5 (RASDP correctable)** Note: This register field is sticky.
RSVDP_6	31:6	r	**Reserved for future use**

19.4.2.294 Status for functional safety interrupt events

This register provides the functional safety interrupt events status.

SAFETY_STATUS_OFF Offset address: 0004964_H

Status for functional safety interrupt events Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_6															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVDP_6										SAFETY_INT_STATUS_5	SAFETY_INT_STATUS_4	SAFETY_INT_STATUS_3	SAFETY_INT_STATUS_2	SAFETY_INT_STATUS_1	SAFETY_INT_STATUS_0
r										rw1ch	rw1ch	rw1ch	rw1ch	rw1ch	rw1ch

Field	Bits	Type	Description
SAFETY_INT_STATUS_0	0	rw1ch	**Status for functional safety interrupt event 0 (RASDP)** Note: This register field is sticky. Software write with 1 clears this bit-field.
SAFETY_INT_STATUS_1	1	rw1ch	**Status for functional safety interrupt event 1 (Interface timers)** Note: This register field is sticky. Software write with 1 clears this bit-field.
SAFETY_INT_STATUS_2	2	rw1ch	**Status for functional safety interrupt event 2 (CDM register checker)** Note: This register field is sticky. Software write with 1 clears this bit-field.
SAFETY_INT_STATUS_3	3	rw1ch	**Status for functional safety interrupt event 3 (PCIe uncorrectable)** Note: This register field is sticky. Software write with 1 clears this bit-field.

(table continues...)

(continued)

Field	Bits	Type	Description
SAFETY_INT_STATUS_4	4	rw1ch	**Status for functional safety interrupt event 4 (PCIe correctable)** Note: This register field is sticky. Software write with 1 clears this bit-field.
SAFETY_INT_STATUS_5	5	rw1ch	**Status for functional safety interrupt event 5 (RASDP correctable)** Note: This register field is sticky. Software write with 1 clears this bit-field.
RSVDP_6	31:6	r	**Reserved for future use**

19.4.2.295 Application driven bus and device number register

This register reflects the application driven bus and device number.

PL_APP_BUS_DEV_NUM_STATUS_OFF Offset address: $0004B10_H$

Application driven bus and device number register Kernel Reset value: $0000\ 0000_H$

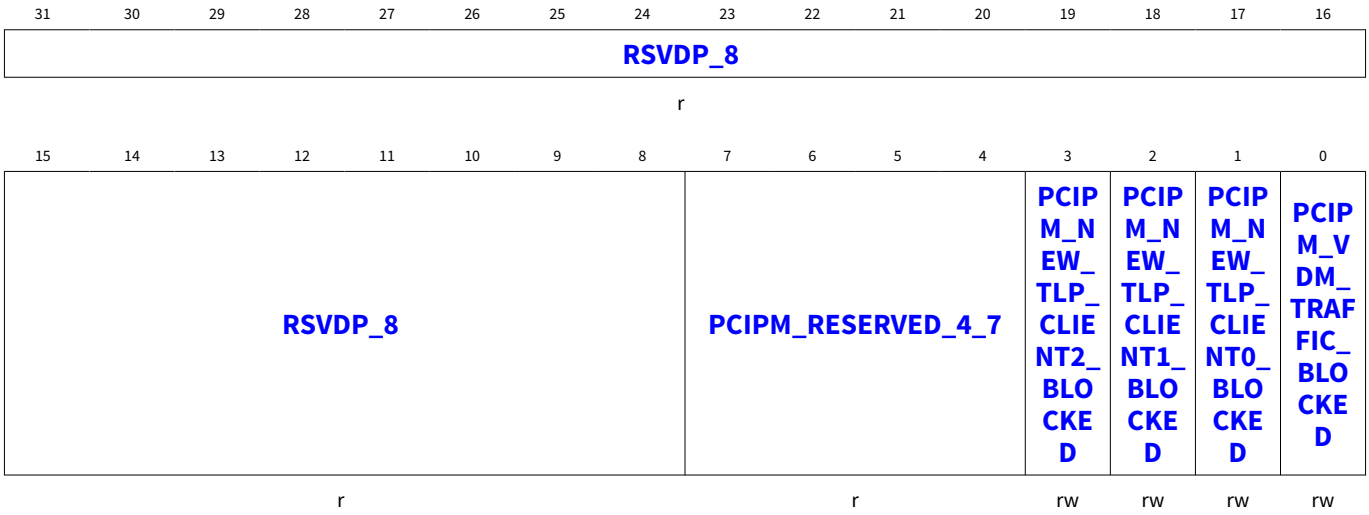
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_16															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RC_DSW_BUS_NUM								RC_DSW_DEV_NUM					RSVDP_0		
rh								rh					r		

Field	Bits	Type	Description
RSVDP_0	2:0	r	**Reserved for future use**
RC_DSW_DEV_NUM	7:3	rh	**This field reflects the value of device number driven on CFGBD.DEVNUM input signal by your application** Note: This register field is sticky.
RC_DSW_BUS_NUM	15:8	rh	**This field reflects the value of bus number driven on CFGBD.BUSNUM input signal by your application** Note: This register field is sticky.
RSVDP_16	31:16	r	**Reserved for future use**

19.4.2.296 TLP Traffic during Non-D0 State Control Register

This register provides control over TLP Traffic during Non-D0 States.



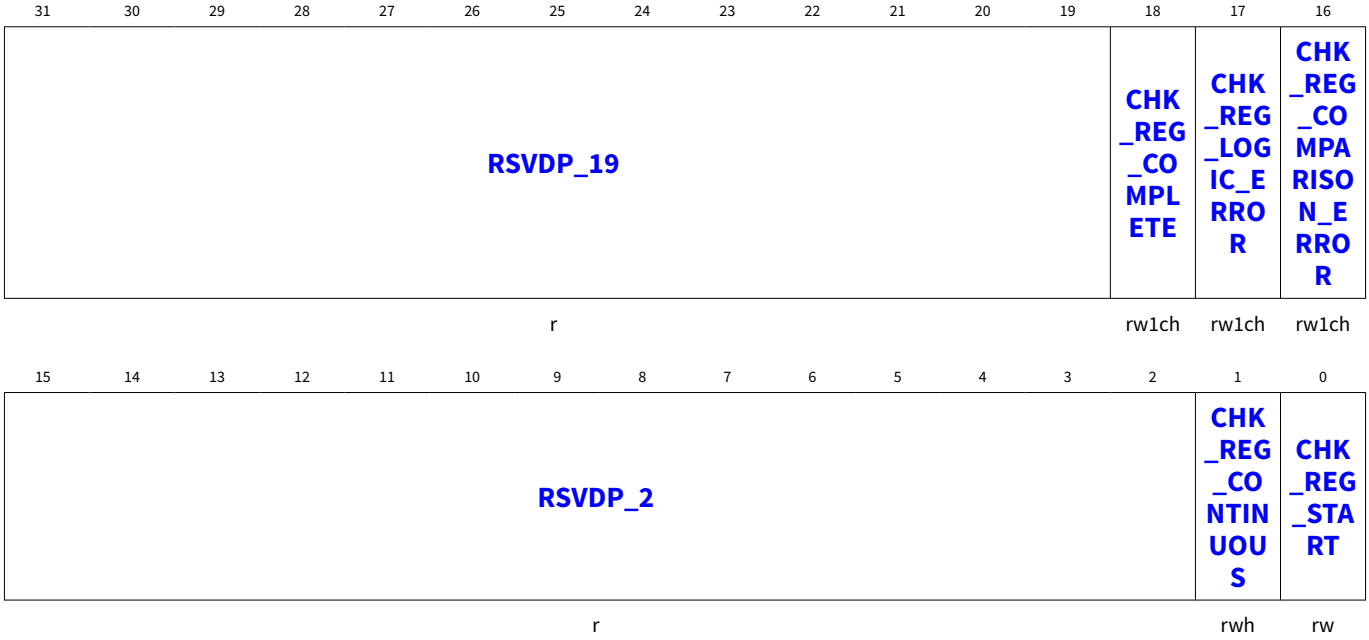
Field	Bits	Type	Description
PCIPM_VDM_TRAFFIC_BLOCKED	0	rw	**This field indicates that VDM Message TLPs are blocked during non-D0 states** Note: This register field is sticky.
PCIPM_NEW_TLP_CLIENT0_BLOCKED	1	rw	**This field indicates that all TLPs transmitted by Client 0 interface are blocked during non-D0 states** Note: This register field is sticky.
PCIPM_NEW_TLP_CLIENT1_BLOCKED	2	rw	**This field indicates that all TLPs transmitted by Client 1 interface are blocked during non-D0 states** Note: This register field is sticky.
PCIPM_NEW_TLP_CLIENT2_BLOCKED	3	rw	**This field indicates that all TLPs transmitted by Client 2 interface are blocked during non-D0 states** Note: This register field is sticky.
PCIPM_RESERVED_4_7	7:4	r	**Reserved** Note: This register field is sticky.
RSVDP_8	31:8	r	**Reserved for future use**

19.4.2.297 CDM Register Checking Control and Status Register

Controls register checking and displays status of register checking.

PL_CHK_REG_CONTROL_STATUS_OFF Offset address: $0004B20_H$

CDM Register Checking Control and Status Register Kernel Reset value: $0000\ 0000_H$



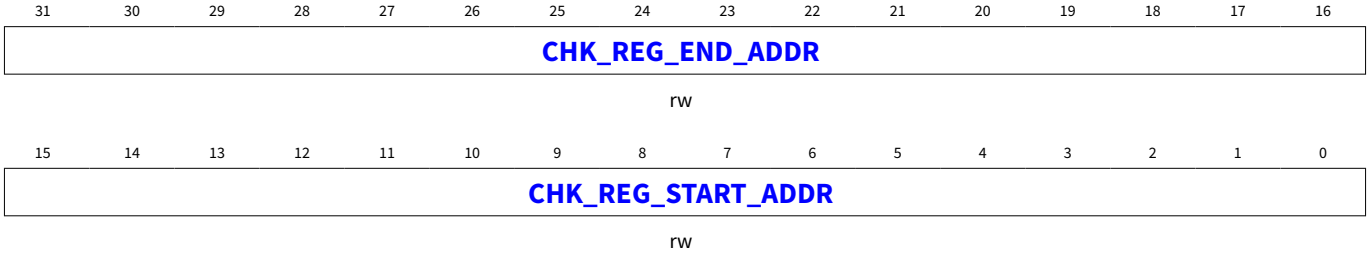
Field	Bits	Type	Description
CHK_REG_START	0	rw	**Begins a checking sequence** Note: This register field is sticky. 0_B **CLEAR**: Clear 1_B **SET**: Set
CHK_REG_CONTINUOUS	1	rwh	**Set Continuous Checking Sequence** Note: This register field is sticky. 0_B **CLEAR**: Clear 1_B **SET**: Set
RSVDP_2	15:2	r	**Reserved for future use**
CHK_REG_COMPARISON_ERROR	16	rw1ch	**The system has detected that there is a bit error in the CDM Register Data** Note: This register field is sticky. Software write with 1 clears this bit-field. 0_B **CLEAR**: Clear 1_B **SET**: Indicates comparison error
CHK_REG_LOGIC_ERROR	17	rw1ch	**The system has detected an error in its own checking logic** Note: This register field is sticky. Software write with 1 clears this bit-field. 0_B **CLEAR**: Clear 1_B **SET**: Indicates an error in comparison logic
CHK_REG_COMPLETE	18	rw1ch	**The system has completed a checking cycle** Note: This register field is sticky. Software write with 1 clears this bit-field. 0_B **CLEAR**: Clear 1_B **SET**: Indicates the system has completed a checking cycle
RSVDP_19	31:19	r	**Reserved for future use**

19.4.2.298 CDM Register Checking First and Last address to check

This register holds the first and last address to check.

PL_CHK_REG_START_END_OFF Offset address: $0004B24_H$

CDM Register Checking First and Last address to check Kernel Reset value: $0BFF\ 0000_H$



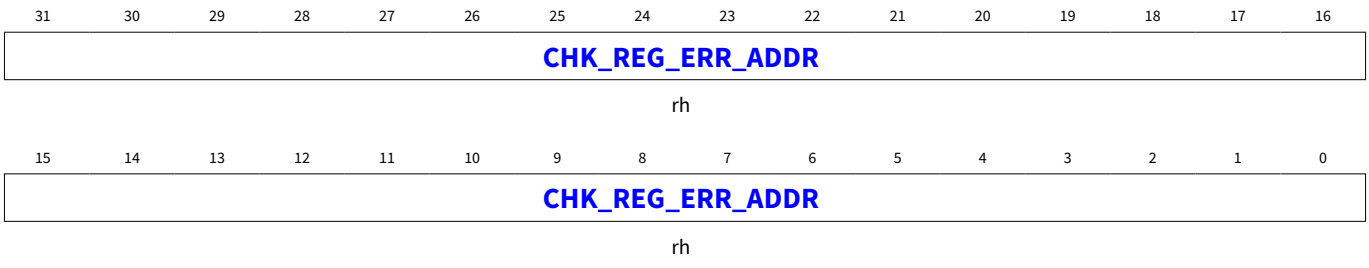
Field	Bits	Type	Description
CHK_REG_START_ADDR	15:0	rw	**The first address that is checked by the system** Note: This register field is sticky. 0000_H **MIN_VAL**: Zero value $FFFF_H$ **MAX_VAL**: Max value
CHK_REG_END_ADDR	31:16	rw	**The last address that is checked by the system** Note: This register field is sticky. 0000_H **MIN_VAL**: Zero value $FFFF_H$ **MAX_VAL**: Max value

19.4.2.299 CDM Register Checking Error Address

This register holds the CDM register checking error address.

PL_CHK_REG_ERR_ADDR_OFF Offset address: $0004B28_H$

CDM Register Checking Error Address Kernel Reset value: $0000\ 0000_H$



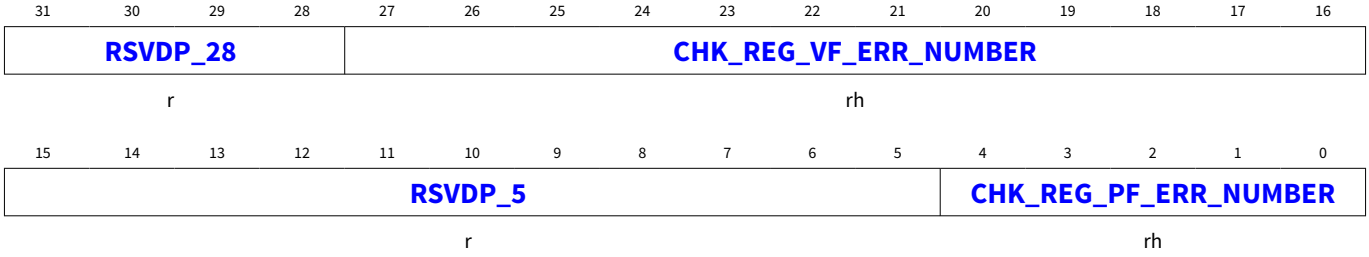
Field	Bits	Type	Description
CHK_REG_ERR_ADDR	31:0	rh	**The address at which an error has been detected** Valid only when the CDM Register Checker Comparison Error bit is set in the status register. Note: This register field is sticky. 00000000_H **MIN_VAL**: Zero value $FFFFFFFF_H$ **MAX_VAL**: Max value

19.4.2.300 CDM Register Checking error PF Numbers

This register holds the CDM Register checking error PF numbers.

PL_CHK_REG_ERR_PF_VF_OFF Offset address: $0004B2C_H$

CDM Register Checking error PF Numbers Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
CHK_REG_PF_ERR_NUMBER	4:0	rh	**The PF number at which the error was detected** Valid only when the CDM Register Checker Comparison Error bit is set in the status register. Note: This register field is sticky. 00_H **MIN_VAL**: Zero value $1F_H$ **MAX_VAL**: Max value
RSVDP_5	15:5	r	**Reserved for future use**
CHK_REG_VF_ERR_NUMBER	27:16	rh	**The VF number at which the error was detected** Valid only when the CDM Register Checker Comparison Error bit is set in the status register. Note: This register field is sticky. 000_H **MIN_VAL**: Zero value $7FF_H$ **MAX_VAL**: Max value **others,** Reserved
RSVDP_28	31:28	r	**Reserved for future use**

19.4.2.301 LTR Latency Register

The function of this register field (and all other fields in this register) differs between an upstream port and a downstream port. For an upstream port, the register fields capture the corresponding fields in the LTR messages that are transmitted by the port. For a downstream port, the register fields capture the corresponding fields in the LTR messages that are received by the port.

PL_LTR_LATENCY_OFF Offset address: $0004B30_H$

LTR Latency Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
NO_SNOOP_LATENCY_REQUIRE	RSVDP_29		NO_SNOOP_LATENCY_SCALE			NO_SNOOP_LATENCY_VALUE									
rw	r		rw			rw									

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SNOOP_LATENCY_REQUIRE	RSVDP_13		SNOOP_LATENCY_SCALE			SNOOP_LATENCY_VALUE									
rw	r		rw			rw									

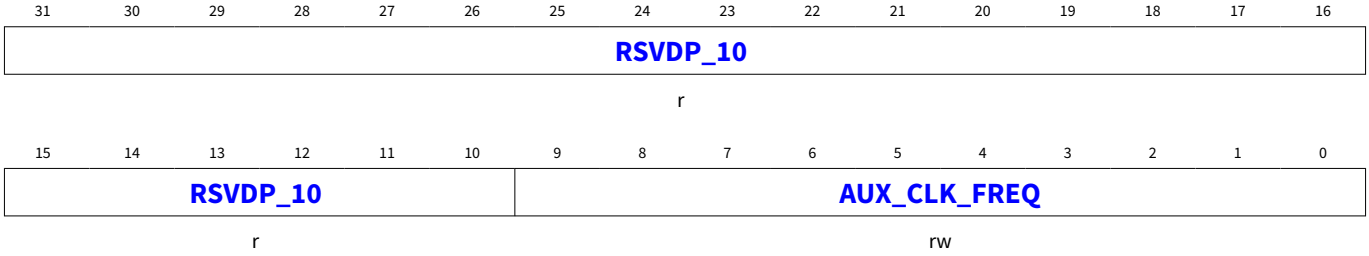
Field	Bits	Type	Description
SNOOP_LATENCY_VALUE	9:0	rw	**Snoop Latency Value** Note: The access attributes of this field are as follows: - Wire: R - Dbi: R/W
SNOOP_LATENCY_SCALE	12:10	rw	**Snoop Latency Scale** Note: The access attributes of this field are as follows: - Wire: R - Dbi: R/W
RSVDP_13	14:13	r	**Reserved for future use**
SNOOP_LATENCY_REQUIRE	15	rw	**Snoop Latency Requirement** Note: The access attributes of this field are as follows: - Wire: R - Dbi: R/W
NO_SNOOP_LATENCY_VALUE	25:16	rw	**No Snoop Latency Value** Note: The access attributes of this field are as follows: - Wire: R - Dbi: R/W
NO_SNOOP_LATENCY_SCALE	28:26	rw	**No Snoop Latency Scale** Note: The access attributes of this field are as follows: - Wire: R - Dbi: R/W
RSVDP_29	30:29	r	**Reserved for future use**
NO_SNOOP_LATENCY_REQUIRE	31	rw	**No Snoop Latency Requirement** Note: The access attributes of this field are as follows: - Wire: R - Dbi: R/W

19.4.2.302 Auxiliary Clock Frequency Control Register

This register controls the auxiliary clock frequency.

AUX_CLK_FREQ_OFF Offset address: $0004B40_H$

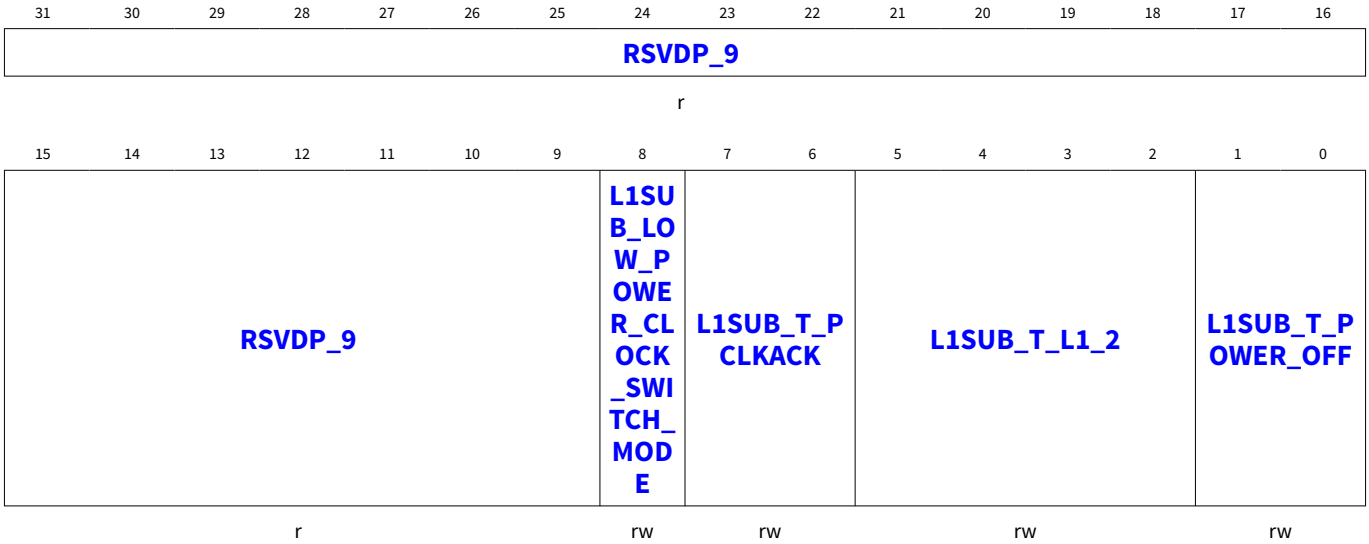
Auxiliary Clock Frequency Control Register Kernel Reset value: $0000\ 000A_H$



Field	Bits	Type	Description
AUX_CLK_FREQ	9:0	rw	**The aux_clk frequency in MHz** This value is used to provide a 1 µs reference for counting time during low-power states with aux_clk when the PHY has removed the pipe_clk. If the actual frequency (f) of aux_clk does not exactly match the programmed frequency (f_prog), then there is an error in the time counted by the controller that can be expressed in percentage as: err% = (f_prog/f-1)*100. For example if f=2.5 MHz and f_prog=3 MHz, then err% =(3/2.5-1)*100 =20%, meaning that the time counted by the controller on aux_clk will be 20% greater than the time in µs programmed in the corresponding time register (for example T_POWER_ON). Note: This register field is sticky.
RSVDP_10	31:10	r	**Reserved for future use**

19.4.2.303 L1 Substates Timing Register

This register holds L1 substates timing information.



Field	Bits	Type	Description
L1SUB_T_POWER_OFF	1:0	rw	**Duration (in 1 µs units) of L1.2.Entry** Range is 1..4. Note: This register field is sticky. 00_B **MIN_VAL**: Min value 11_B **MAX_VAL**: Max value

(table continues...)

(continued)

Field	Bits	Type	Description
L1SUB_T_L1_2	5:2	rw	**Duration (in 1 µs units) of L1.2** Range is 1..16. Note: This register field is sticky. 0_H **MIN_VAL**: Min value F_H **MAX_VAL**: Max value
L1SUB_T_PCLKACK	7:6	rw	**Max delay (in 1 µs units) between a MAC request to remove the clock and a PHY response** If the PHY does not respond within this time the request is aborted. Range is 1..4 Note: This register field is sticky. 00_B **MIN_VAL**: Min value 11_B **MAX_VAL**: Max value
L1SUB_LOW_POWER_CLOCK_SWITCH_MODE	8	rw	**If the bit is set to 1'b1 the controller will delay the switching of aux_clk to the slow platform clock until it detects that the link partner has de-asserted CLKREQ#** Note: This register field is sticky. 0_B **CLEAR**: The reference clock may be gated off when CLKREQ# is de-asserted 1_B **SET**: The reference clock shall be kept running regardless of the CLKREQ# setting
RSVDP_9	31:9	r	**Reserved for future use**

19.4.2.304 Powerdown Control and Status Register

This is the Powerdown Control and Status register.

POWERDOWN_CTRL_STATUS_OFF Offset address: $0004B48_H$

Powerdown Control and Status Register Kernel Reset value: $0000\ 0220_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_12															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVDP_12				POWERDOWN_PHY_POWERDOWN				POWERDOWN_MAC_POWERDOWN				RSVDP_2		POWERDOWN_VMAIN_ACK	POWERDOWN_FORCE
r				rh				rh				r		rw	rwh

Field	Bits	Type	Description
POWERDOWN_FORCE	0	rwh	**This field is a one shot field** This field could be used for debug purposes in event that the P2 Powerdown transition does not complete. It will allow the controller to proceed with the transition to the P1 Powerdown state. This field always reads back as 1'b0. 0_B **CLEAR**: Clear 1_B **SET**: Writing a value of 1 to this field causes the controller to complete the P2 Powerdown handshake regardless of whether the PHY has returned Phystatus
POWERDOWN_VMAIN_ACK	1	rw	**Set this bit to 1 if you do not want to perform the handshake with the power-switch after PERST# assertion** By default the controller will perform the handshake with the power-switch if L1 power gating is enabled Note: This register field is sticky. 0_B **CLEAR**: Clear 1_B **SET**: If you do not want to perform the handshake with the power-switch after PERST# assertion
RSVDP_2	3:2	r	**Reserved for future use**
POWERDOWN_MAC_POWERDOWN	7:4	rh	**This field represents the Powerdown value driven by the controller to the PHY**
POWERDOWN_PHY_POWERDOWN	11:8	rh	**This field represents the Powerdown value that has been acknowledged by the PHY** It is updated with the value of Powerdown driven by the controller, when the PHY has returned the Phystatus acknowledgment for the Powerdown transition.
RSVDP_12	31:12	r	**Reserved for future use**

19.4.2.305 PIPE Related Register

This register controls the PIPE's capabitity, control, and status parameters.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_9															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVDP_9							PIPE_GARBAGE_DATA_MODE	0							
r							rw	r							

Field	Bits	Type	Description
PIPE_GARBAGE_DATA_MODE	8	rw	**PIPE Garbage Data Mode** • RxValid is deasserted - a valid RxStartBlock is received at 128b/130b encoding • a valid COM symbol is received at 8b/10b encoding Note: This register field is sticky. 0_B **COMPLIANT_MODE**: PIPE Spec compliant mode: The MAC discards any symbols received after the electrical idle ordered-set until RxValid is deasserted 1_B **PHY_SUP_MODE**: Special PHY Support mode: The MAC discards any symbols received after the electrical idle ordered-set until when any of the following three conditions are true:
RSVDP_9	31:9	r	**Reserved for future use**
0	7:0	r	**Reserved** Read as all 0's; should be written with all 0's.

19.4.2.306 MSIX x table DWORD

MSIXPBAx_DWORD (x=0-1) Offset address: $0005E10_H$+x*4

MSIX x table DWORD Kernel Reset value: $0000\ 0080_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PB31	PB30	PB29	PB28	PB27	PB26	PB25	PB24	PB23	PB22	PB21	PB20	PB19	PB18	PB17	PB16
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PB15	PB14	PB13	PB12	PB11	PB10	PB9	PB8	PB7	PB6	PB5	PB4	PB3	PB2	PB1	PB0
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
PBy (y=0-31)	y	rh	**Pending bit, as defined in the PCIe spec.**

19.4.2.307 MSI-X Capability ID, Next Pointer, Control Register

This is the shadow register of the MSI-X Capability ID, Next Pointer, and Control Register.

SHADOW_PCI_MSIX_CAP_ID_NEXT_CTRL_REG Offset address: $00060B0_H$

MSI-X Capability ID, Next Pointer, Control Register Kernel Reset value: $0040\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PCI_MSIX_RESERVED1					PCI_MSIX_TABLE_SIZE										
r					rh										

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PCI_MSIX_RESERVED0															
r															

Field	Bits	Type	Description
PCI_MSIX_RESERVED0	15:0	r	**reserved field 0 in the shadow register** Note: This register field is sticky.
PCI_MSIX_TABLE_SIZE	26:16	rh	**MSI-X Table Size in the shadow register** Note: The access attributes of this field are as follows: - Wire: No access - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R Note: This register field is sticky.
PCI_MSIX_RESERVED1	31:27	r	**reserved field 1 in the shadow register** Note: This register field is sticky.

19.4.2.308 MSI-X Table Offset and BIR Register

This register provides Table BIR and MSI-x Table offset select.

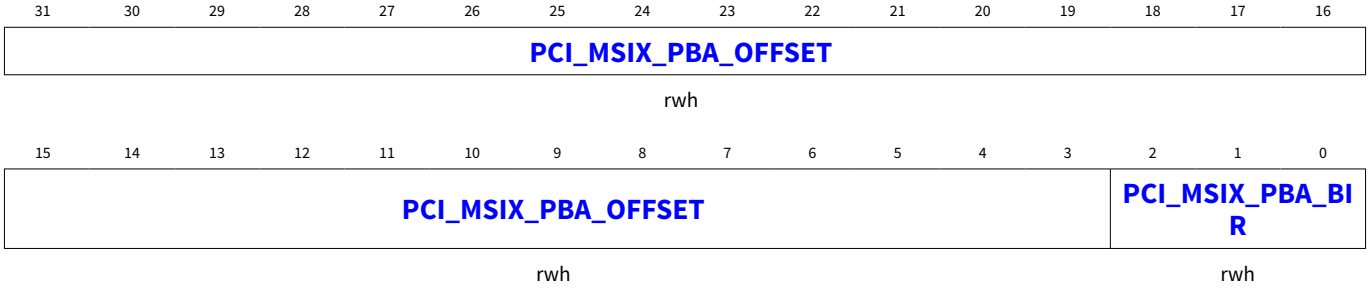
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PCI_MSIX_TABLE_OFFSET															
rwh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PCI_MSIX_TABLE_OFFSET													PCI_MSIX_BIR		
rwh													rwh		

Field	Bits	Type	Description
PCI_MSIX_BIR	2:0	rwh	**MSI-X Table BAR Indicator Register Field** This field indicates which one of a Function's Base Address Registers , located beginning at 10h in Configuration Space, or entry in the Enhanced Allocation capability with a matching BEI , is used to map the Function's MSI-X Table into Memory Space. All encodings other than the defined encodings are reserved. For a 64-bit Base Address Register , the Table BIR indicates the lower DWORD. For Functions with Type 1 Configuration Space headers, BIR values 2 through 5 are also Reserved. Note: The access attributes of this field are as follows: - Wire: No access - Dbi: if (DBI_RO_WR_EN == 1 && MSIX_CAP_ENABLE=1) then R/W else R Note: This register field is sticky. 000_B **BAR_10**: Base Address Register 10h 001_B **BAR_14**: Base Address Register 14h 010_B **BAR_18**: Base Address Register 18h 011_B **BAR_1C**: Base Address Register 1Ch 100_B **BAR_20**: Base Address Register 20h 101_B **BAR_24**: Base Address Register 24h **others,** Reserved
PCI_MSIX_TABLE_OFFSET	31:3	rwh	**MSI-X Table Offset** Used as an offset from the address contained by one of the Function's Base Address Registers to point to the base of the MSI-X Table. The lower 3 Table BIR bits are masked off (set to zero) by software to form a 32-bit QWORD-aligned offset. Note: The access attributes of this field are as follows: - Wire: No access - Dbi: if (DBI_RO_WR_EN == 1 && MSIX_CAP_ENABLE=1) then R/W else R Note: This register field is sticky.

19.4.2.309 MSI-X PBA Offset and BIR Register

This register provides PBA Offset and PBA BIR value.



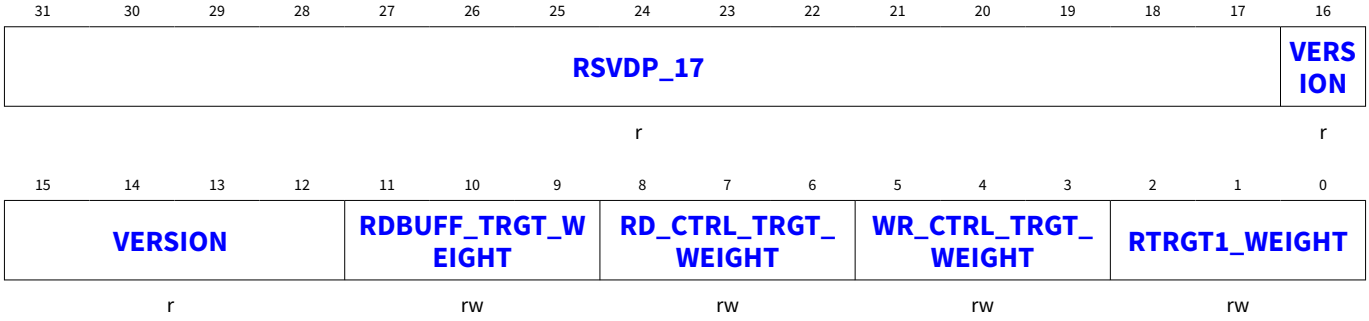
Field	Bits	Type	Description
PCI_MSIX_PBA_BIR	2:0	rwh	**MSI-X PBA BIR** This field indicates which one of a Function's Base Address Registers , located beginning at 10h in Configuration Space, or entry in the Enhanced Allocation capability with a matching BEI, is used to map the Function's MSI-X PBA into Memory Space. The PBA BIR value definitions are identical to those for the Table BIR . Note: The access attributes of this field are as follows: • Wire: No access • Dbi: if (DBI_RO_WR_EN == 1 && MSIX_CAP_ENABLE=1) then R/W else R Note: This register field is sticky.
PCI_MSIX_PBA_OFFSET	31:3	rwh	**MSI-X PBA Offset** Used as an offset from the address contained by one of the Function's Base Address Registers to point to the base of the MSI-X PBA. The lower 3 PBA BIR bits are masked off (set to zero) by software to form a 32-bit QWORD-aligned offset. Note: The access attributes of this field are as follows: • Wire: No access • Dbi: if (DBI_RO_WR_EN == 1 && MSIX_CAP_ENABLE=1) then R/W else R Note: This register field is sticky.

19.4.2.310 DMA Arbitration Scheme for TRGT1 Interface

This register is used to control traffic priorities among various sources that are delivered to your application through TRGT1 where 0x0 represents the highest priority. - Non-DMA Rx Requests - DMA Write Channel MRd Requests (DMA data requests and LL element/descriptor access) - DMA Read Channel MRd Requests (LL element/descriptor access) - DMA Read Channel MWr Requests Concurrent traffic from channels with same priority are sorted according to Round-Robin arbitration rules. The arbitration priority defaults to Non-DMA requests (highest), Write Channel MRd, Read Channel MRd, Read Channel MWr.

DMA_CTRL_DATA_ARB_PRIOR_OFF Offset address: 0007000_H

DMA Arbitration Scheme for TRGT1 Interface Kernel Reset value: $0000\ 0688_H$



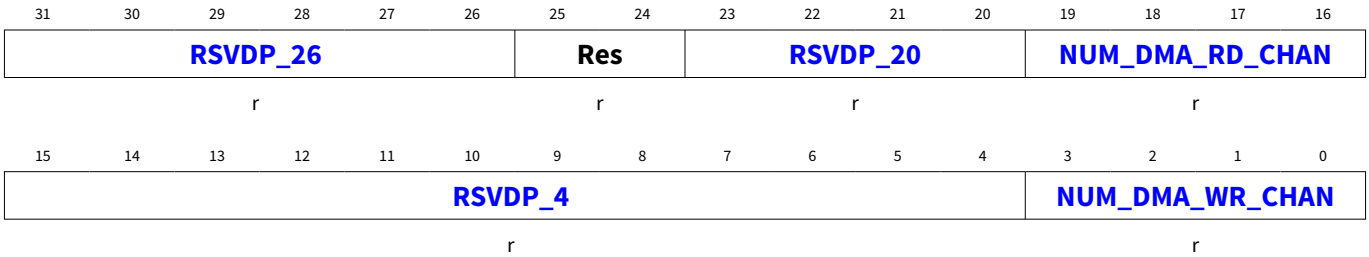
Field	Bits	Type	Description
RTRGT1_WEIGHT	2:0	rw	**Non-DMA Rx Requests** Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
WR_CTRL_TRGT_WEIGHT	5:3	rw	**DMA Write Channel MRd Requests (for DMA data requests and LL element/descriptor access)** Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
RD_CTRL_TRGT_WEIGHT	8:6	rw	**DMA Read Channel MRd Requests (for LL element/descriptor access)** Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
RDBUFF_TRGT_WEIGHT	11:9	rw	**DMA Read Channel MWr Requests** Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
VERSION	16:12	r	**Reports the version of Register Map of eDMA**
RSVDP_17	31:17	r	**Reserved for future use**

19.4.2.311 DMA Number of Channels Register

This register provides information regarding the number of configured DMA read and write channels. Note: When this register does not exist, value is fixed to 32'hFFFF_FFFF.

DMA_CTRL_OFF Offset address: 0007008_H

DMA Number of Channels Register Kernel Reset value: $0004\ 0004_H$



Field	Bits	Type	Description
NUM_DMA_WR_CHAN	3:0	r	**Number of Write Channels** You can read this register to determine the number of write channels configured.
RSVDP_4	15:4	r	**Reserved for future use**
NUM_DMA_RD_CHAN	19:16	r	**Number of Read Channels** You can read this register to determine the number of read channels configured.
RSVDP_20	23:20	r	**Reserved for future use**
RSVDP_26	31:26	r	**Reserved for future use**

19.4.2.312 DMA Write Engine Enable Register

This register indicates the status of DMA write engine and the status of DMA write engine handshake feature (per-channel).

DMA_WRITE_ENGINE_EN_OFF Offset address: 000700C$_H$

DMA Write Engine Enable Register Kernel Reset value: 0000 0000$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_24								DMA_WRITE_ENGINE_EN_HSHAKE_CH7	DMA_WRITE_ENGINE_EN_HSHAKE_CH6	DMA_WRITE_ENGINE_EN_HSHAKE_CH5	DMA_WRITE_ENGINE_EN_HSHAKE_CH4	DMA_WRITE_ENGINE_EN_HSHAKE_CH3	DMA_WRITE_ENGINE_EN_HSHAKE_CH2	DMA_WRITE_ENGINE_EN_HSHAKE_CH1	DMA_WRITE_ENGINE_EN_HSHAKE_CH0
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVDP_1															DMA_WRITE_ENGINE
r															rwh

Field	Bits	Type	Description
DMA_WRITE_ENGINE	0	rwh	**DMA Write Engine Enable** For normal operation, you must initially set this bit to '1', before any other software setup action. You do not need to toggle or rewrite to this bit during normal operation. You should set this bit to '0' when you want to "Soft Reset" the DMA controller write logic. There are three possible reasons for resetting the DMA controller write logic • The "Abort Interrupt Status" bit is set in the DMA Write Interrupt Status Register (DMA_WRITE_INT_STATUS_OFF), and any of the bits in the DMA Write Error Status Register (DMA_WRITE_ERR_STATUS_OFF) are set. Resetting the DMA controller write logic re-initializes the control logic, ensuring that the next DMA write transfer is executed successfully. • You have executed the procedure outlined in "Stop Bit", after which, the "Abort Interrupt Status" bit is set and the Channel Status(CS) field of the DMA Write Channel Control 1 Register (DMA_CH_CONTROL1_OFF_WRCH_i) is set to "Stopped". Resetting the DMA controller write logic re-initializes the control logic ensuring that the next DMA write transfer is executed successfully • During software development, when you incorrectly program the DMA write engine To "Soft Reset" the DMA controller write logic, you must: • De-assert the DMA write engine enable bit. • Wait for the DMA to complete any in-progress TLP transfer, by waiting until a read on the DMA write engine enable bit returns a '0'. • Assert the DMA write engine enable bit. This "Soft Reset" does not clear the DMA configuration registers. The DMA write transfer does not start until you write to the DMA Write Doorbell Register (DMA_WRITE_DOORBELL_OFF) Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W 0_B **DISABLE**: Disable (Soft Reset) 1_B **ENABLE**: Enable
RSVDP_1	15:1	r	**Reserved for future use**
DMA_WRITE_ENGINE_EN_HSHAKE_CH0	16	rw	**Enable Handshake for DMA Write Engine Channel 0** Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
DMA_WRITE_ENGINE_EN_HSHAKE_CH1	17	rw	**Enable Handshake for DMA Write Engine Channel 1** Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
DMA_WRITE_ENGINE_EN_HSHAKE_CH2	18	rw	**Enable Handshake for DMA Write Engine Channel 2** Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
DMA_WRITE_ENGINE_EN_HSHAKE_CH3	19	rw	**Enable Handshake for DMA Write Engine Channel 3** Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W

(table continues...)

(continued)

Field	Bits	Type	Description
DMA_WRITE_ENGINE_EN_HSHAKE_CH4	20	rw	**Enable Handshake for DMA Write Engine Channel 4** Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
DMA_WRITE_ENGINE_EN_HSHAKE_CH5	21	rw	**Enable Handshake for DMA Write Engine Channel 5** Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
DMA_WRITE_ENGINE_EN_HSHAKE_CH6	22	rw	**Enable Handshake for DMA Write Engine Channel 6** Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
DMA_WRITE_ENGINE_EN_HSHAKE_CH7	23	rw	**Enable Handshake for DMA Write Engine Channel 7** Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
RSVDP_24	31:24	r	**Reserved for future use**

19.4.2.313 DMA Write Doorbell Register

This register controls the DMA write channel doorbell.

DMA_WRITE_DOORBELL_OFF Offset address: 0007010_H

DMA Write Doorbell Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
WR_STOP	RSVDP_3														
rwh	r														

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVDP_3													WR_DOORBELL_NUM		
r													rwh		

Field	Bits	Type	Description
WR_DOORBELL_NUM	2:0	rwh	**Doorbell Number** You must write the channel number to this register to start the DMA write transfer for that channel. The DMA detects a write to this register field even if the value of this field does not change. You do not need to toggle or write any other value to this register to start a new transfer. The range of this field is 0x0 to 0x7, where 0x0 corresponds to channel 0. Note: A write to this field triggers the controller to exit L1 substates. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
RSVDP_3	30:3	r	**Reserved for future use**

(table continues...)

(continued)

Field	Bits	Type	Description
WR_STOP	31	rwh	**Stop** Set in conjunction with the Doorbell Number field. The DMA write channel stops issuing requests, sets the channel status to "Stopped", and asserts the "Abort" interrupt if it is enabled. Before setting the Stop bit, you must read the channel Status field (CS) of the DMA Write Channel Control 1 Register (DMA_CH_CONTROL1_OFF_WRCH_0) to ensure that the write channel is "Running" (transferring data). For more information, see "Stopping the DMA Transfer (Software Stop)". Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W

19.4.2.314 DMA Write Engine Channel Arbitration Weight Low Register

The 5-bit channel weight (for write channels 0-3) specifies the maximum number of TLP requests that the DMA can issue for that channel before it must return to the arbitration routine. When the channel weight count is reached or DMA channel request transfer size reaches zero, the WWR arbiter selects the next channel to be processed. Your software must initialize this register before ringing the doorbell. Value range is (0-0x1F) corresponding to (1-32) transaction requests.

DMA_WRITE_CHANNEL_ARB_WEIGHT_LOW_OFF Offset address: 0007018_H

DMA Write Engine Channel Arbitration Weight Low Register — Kernel Reset value: $0000\ 8421_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_20												WRITE_CHANNEL3_WEIGHT			
r												rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WRITE_CHANNEL3_WEIGHT	WRITE_CHANNEL2_WEIGHT					WRITE_CHANNEL1_WEIGHT					WRITE_CHANNEL0_WEIGHT				
rw	rw					rw					rw				

Field	Bits	Type	Description
WRITE_CHANNEL0_WEIGHT	4:0	rw	**Channel 0 Weight** The weight is initialized by software before ringing the doorbell. The value is used by the channel weighted round robin arbiter to select the next channel read request. A value of '0' means that one TLP is issued before moving to the next channel. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
WRITE_CHANNEL1_WEIGHT	9:5	rw	**Channel 1 Weight** The weight is initialized by software before ringing the doorbell. The value is used by the channel weighted round robin arbiter to select the next channel read request. A value of '0' means that one TLP is issued before moving to the next channel. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W

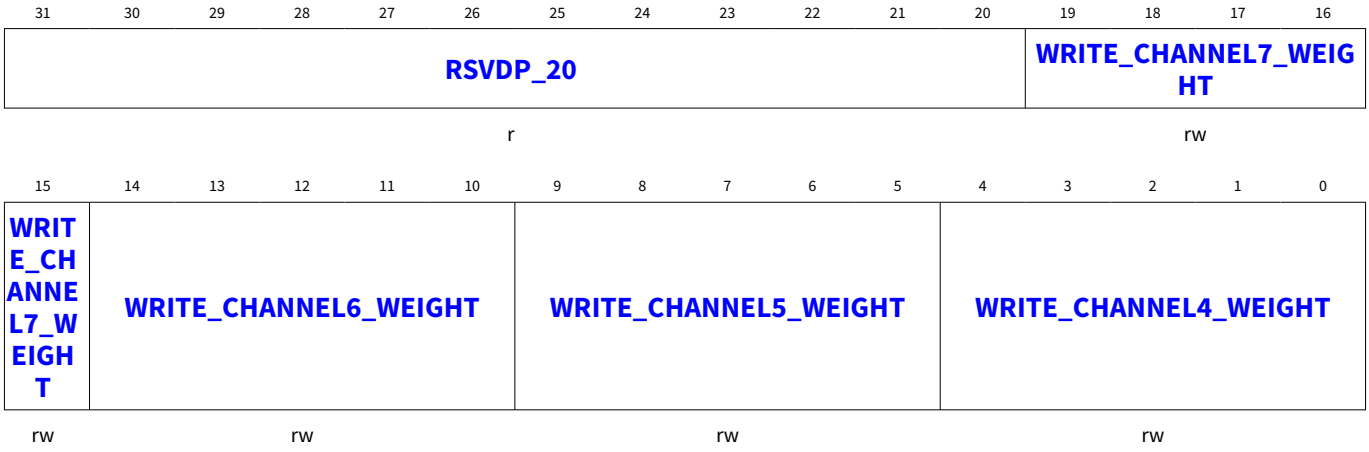
(table continues...)

(continued)

Field	Bits	Type	Description
WRITE_CHANNEL2_WEIGHT	14:10	rw	**Channel 2 Weight** The weight is initialized by software before ringing the doorbell. The value is used by the channel weighted round robin arbiter to select the next channel read request. A value of '0' means that one TLP is issued before moving to the next channel. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
WRITE_CHANNEL3_WEIGHT	19:15	rw	**Channel 3 Weight** The weight is initialized by software before ringing the doorbell. The value is used by the channel weighted round robin arbiter to select the next channel read request. A value of '0' means that one TLP is issued before moving to the next channel. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
RSVDP_20	31:20	r	**Reserved for future use**

19.4.2.315 DMA Write Engine Channel Arbitration Weight High Register

The 5-bit channel weight (for write channels 4-7) specifies the maximum number of TLP requests that the DMA can issue for that channel before it must return to the arbitration routine. When the channel weight count is reached or DMA channel request transfer size reaches zero, the WWR arbiter selects the next channel to be processed. Your software must initialize this register before ringing the doorbell. For more details, see "Multichannel Arbitration". Value range is (0-0x1F) corresponding to (1-32) transaction requests.



Field	Bits	Type	Description
WRITE_CHANNEL4_WEIGHT	4:0	rw	**Channel 4 Weight** The weight is initialized by software before ringing the doorbell. The value is used by the channel weighted round robin arbiter to select the next channel read request. A value of '0' means that one TLP is issued before moving to the next channel. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W

(table continues...)

(continued)

Field	Bits	Type	Description
WRITE_CHANNEL5_WEIGHT	9:5	rw	**Channel 5 Weight** The weight is initialized by software before ringing the doorbell. The value is used by the channel weighted round robin arbiter to select the next channel read request. A value of '0' means that one TLP is issued before moving to the next channel. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
WRITE_CHANNEL6_WEIGHT	14:10	rw	**Channel 6 Weight** The weight is initialized by software before ringing the doorbell. The value is used by the channel weighted round robin arbiter to select the next channel read request. A value of '0' means that one TLP is issued before moving to the next channel. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
WRITE_CHANNEL7_WEIGHT	19:15	rw	**Channel 7 Weight** The weight is initialized by software before ringing the doorbell. The value is used by the channel weighted round robin arbiter to select the next channel read request. A value of '0' means that one TLP is issued before moving to the next channel. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
RSVDP_20	31:20	r	**Reserved for future use**

19.4.2.316 DMA Read Engine Enable Register

This register indicates the status of DMA read engine and the status of DMA read engine handshake feature (per-channel).

DMA_READ_ENGINE_EN_OFF Offset address: $000702C_H$

DMA Read Engine Enable Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_24								Res				DMA_READ_ENGINE_EN_HSHAKE_CH3	DMA_READ_ENGINE_EN_HSHAKE_CH2	DMA_READ_ENGINE_EN_HSHAKE_CH1	DMA_READ_ENGINE_EN_HSHAKE_CH0
r								r				rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVDP_1															DMA_READ_ENGINE
r															rwh

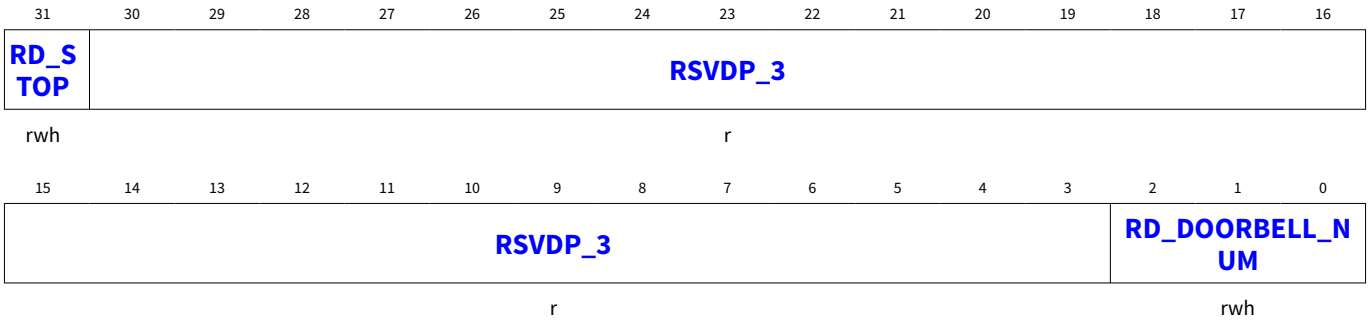
Field	Bits	Type	Description
DMA_READ_ENGINE	0	rwh	**DMA Read Engine Enable** For normal operation, you must initially set this bit to '1', before any other software setup action. You do not need to toggle or rewrite to this bit during normal operation. You should set this field to '0' when you want to "Soft Reset" the DMA controller read logic. There are three possible reasons for resetting the DMA controller read logic: - The "Abort Interrupt Status" bit is set in the DMA Read Interrupt Status Register (DMA_READ_INT_STATUS_OFF), and any of the bits in the DMA Read Error Status Low Register (DMA_READ_ERR_STATUS_LOW_OFF) is set. Resetting the DMA controller read logic re-initializes the control logic, ensuring that the next DMA read transfer is executed successfully. - You have executed the procedure outlined in "Stop Bit", after which, the "Abort Interrupt Status" bit is set and the channel Status field (CS) of the DMA Read Channel Control 1 Register (DMA_CH_CONTROL1_OFF_WRCH_i) is set to "Stopped". Resetting the DMA controller read logic re-initializes the control logic ensuring that the next DMA read transfer is executed successfully. - During software development, when you incorrectly program the DMA read engine. To "Soft Reset" the DMA controller read logic, you must: - De-assert the DMA read engine enable bit. - Wait for the DMA to complete any in-progress TLP transfer, by waiting until a read on the DMA read engine enable bit returns a '0'. - Assert the DMA read engine enable bit. This "Soft Reset" does not clear the DMA configuration registers. The DMA read transfer does not start until you write to the DMA Read Doorbell Register (DMA_READ_DOORBELL_OFF). Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W 0_B **DISABLE**: Disable (Soft Reset) 1_B **ENABLE**: Enable
RSVDP_1	15:1	r	**Reserved for future use**
DMA_READ_ENGINE_EN_HSHAKE_CH0	16	rw	**Enable Handshake for DMA Read Engine Channel 0** Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
DMA_READ_ENGINE_EN_HSHAKE_CH1	17	rw	**Enable Handshake for DMA Read Engine Channel 1** Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
DMA_READ_ENGINE_EN_HSHAKE_CH2	18	rw	**Enable Handshake for DMA Read Engine Channel 2** Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
DMA_READ_ENGINE_EN_HSHAKE_CH3	19	rw	**Enable Handshake for DMA Read Engine Channel 3** Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
RSVDP_24	31:24	r	**Reserved for future use**

19.4.2.317 DMA Read Doorbell Register

This register controls the DMA read channel doorbell.

DMA_READ_DOORBELL_OFF Offset address: 0007030_H

DMA Read Doorbell Register Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
RD_DOORBELL_NUM	2:0	rwh	**Doorbell Number** You must write 0x0 to this register to start the DMA read transfer for that channel. The DMA detects a write to this register field even if the value of this field does not change. The range of this field is 0x0 to 0x7, where 0x0 corresponds to channel 0. Note: A write to this field triggers the controller to exit L1 substates. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
RSVDP_3	30:3	r	**Reserved for future use**
RD_STOP	31	rwh	**Stop** Set in conjunction with the Doorbell Number field. The DMA read channel stops issuing requests, sets the channel status to "Stopped", and asserts the "Abort" interrupt if it is enabled. Before setting the Stop bit, you must read the channel Status field (CS) of the DMA Channel Control 1 Register (DMA_CH_CONTROL1_OFF_RDCH_i) to ensure that the read channel is "Running" (transferring data). For more information, see "Stopping the DMA Transfer (Software Stop)". Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W

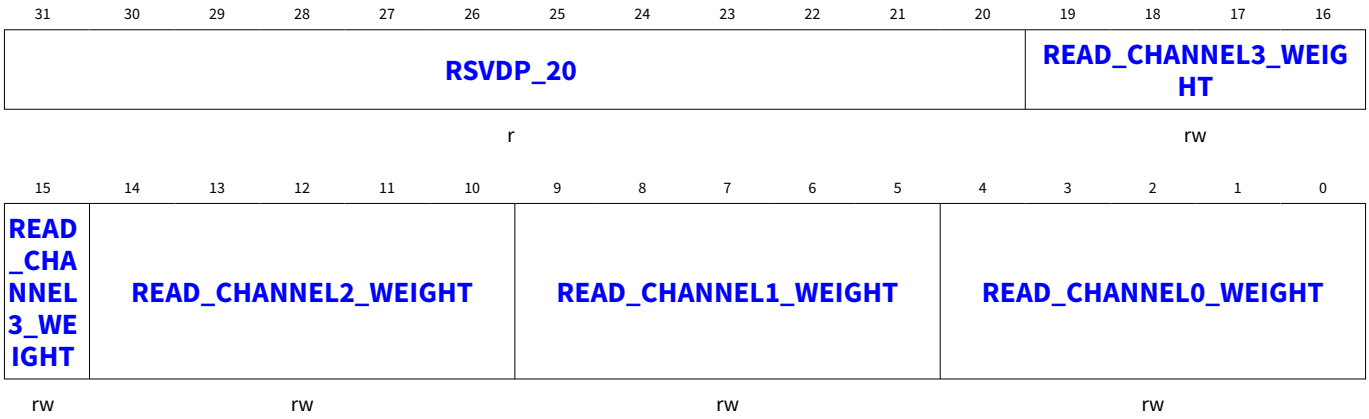
19.4.2.318 DMA Read Engine Channel Arbitration Weight Low Register

The 5-bit channel weight (for read channels 0-3) specifies the maximum number of TLP requests that the DMA can issue for that channel before it must return to the arbitration routine. When the channel weight count is reached or DMA channel request transfer size reaches zero, the WWR arbiter selects the next channel to be processed. Your software must initialize this register before ringing the doorbell. For more details, see "Multichannel Arbitration". Value range is (0-0x1F) corresponding to (1-32) transaction requests.

DMA_READ_CHANNEL_ARB_WEIGHT_LOW_OFF Offset address: 0007038_H

DMA Read Engine Channel Arbitration Weight Low Register

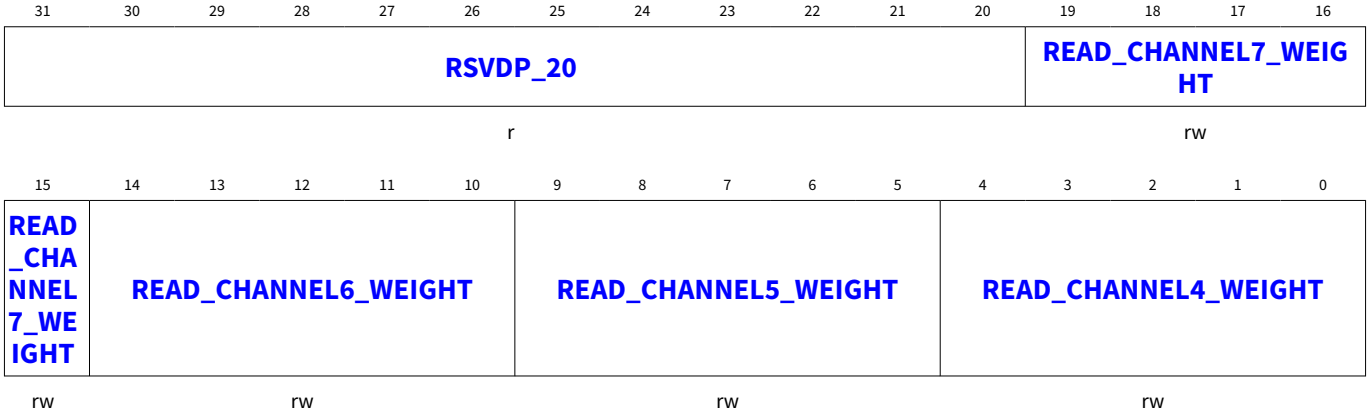
Kernel Reset value: $0000\ 8421_H$



Field	Bits	Type	Description
READ_CHANNEL0_WEIGHT	4:0	rw	**Channel 0 Weight** The weight is initialized by software before ringing the doorbell. The value is used by the channel weighted round robin arbiter to select the next channel read request. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
READ_CHANNEL1_WEIGHT	9:5	rw	**Channel 1 Weight** The weight is initialized by software before ringing the doorbell. The value is used by the channel weighted round robin arbiter to select the next channel read request. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
READ_CHANNEL2_WEIGHT	14:10	rw	**Channel 2 Weight** The weight is initialized by software before ringing the doorbell. The value is used by the channel weighted round robin arbiter to select the next channel read request. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
READ_CHANNEL3_WEIGHT	19:15	rw	**Channel 3 Weight** The weight is initialized by software before ringing the doorbell. The value is used by the channel weighted round robin arbiter to select the next channel read request. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
RSVDP_20	31:20	r	**Reserved for future use**

19.4.2.319 DMA Read Engine Channel Arbitration Weight High Register

The 5-bit channel weight (for read channels 4-7) specifies the maximum number of TLP requests that the DMA can issue for that channel before it must return to the arbitration routine. When the channel weight count is reached or DMA channel request transfer size reaches zero, the WWR arbiter selects the next channel to be processed. Your software must initialize this register before ringing the doorbell. For more details, see "Multichannel Arbitration". Value range is (0-0x1F) corresponding to (1-32) transaction requests.



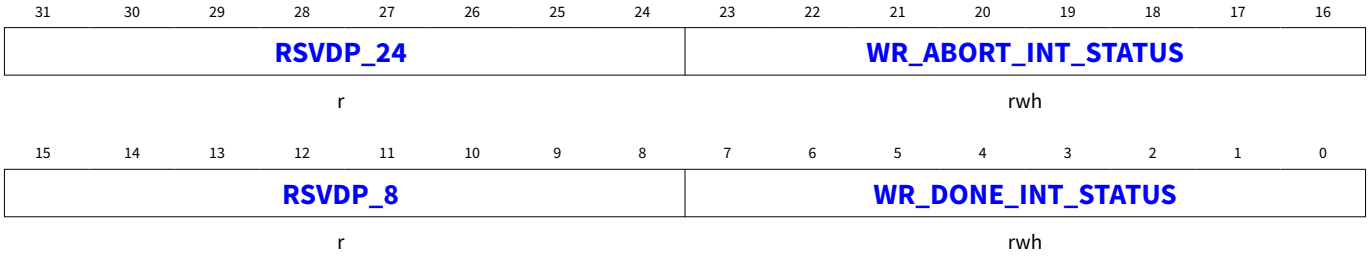
Field	Bits	Type	Description
READ_CHANNEL4_WEIGHT	4:0	rw	**Channel 4 Weight** The weight is initialized by software before ringing the doorbell. The value is used by the channel weighted round robin arbiter to select the next channel read request. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
READ_CHANNEL5_WEIGHT	9:5	rw	**Channel 5 Weight** The weight is initialized by software before ringing the doorbell. The value is used by the channel weighted round robin arbiter to select the next channel read request. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
READ_CHANNEL6_WEIGHT	14:10	rw	**Channel 6 Weight** The weight is initialized by software before ringing the doorbell. The value is used by the channel weighted round robin arbiter to select the next channel read request. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
READ_CHANNEL7_WEIGHT	19:15	rw	**Channel 7 Weight** The weight is initialized by software before ringing the doorbell. The value is used by the channel weighted round robin arbiter to select the next channel read request. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
RSVDP_20	31:20	r	**Reserved for future use**

19.4.2.320 DMA Write Interrupt Status Register

This register indicates the status of Abort and Done interrupts for the DMA write channels.

DMA_WRITE_INT_STATUS_OFF Offset address: $000704C_H$

DMA Write Interrupt Status Register Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
WR_DONE_INT_STATUS	7:0	rwh	**Done Interrupt Status** The DMA write channel has successfully completed the DMA transfer. Each bit corresponds to a DMA channel. Bit [0] corresponds to channel 0. • Enabling: For more details, see "Interrupts and Error Handling" • Masking: The DMA write interrupt Mask register has no effect on this register • Clearing: You must write a 1'b1 to the corresponding channel bit in the DMA write interrupt Clear register to clear this interrupt bit. Note: You can write to this register to emulate interrupt generation, during software or hardware testing. A write to the address triggers an interrupt, but the DMA does not set the Done or Abort bits in this register . Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
RSVDP_8	15:8	r	**Reserved for future use**
WR_ABORT_INT_STATUS	23:16	rwh	**Abort Interrupt Status** The DMA write channel has detected an error, or you manually stopped the transfer as described in "Error Handling Assistance by Remote Software". Each bit corresponds to a DMA channel. Bit [0] corresponds to channel 0. • Enabling: For more details, see "Interrupts and Error Handling" • Masking: The DMA write interrupt Mask register has no effect on this register • Clearing: You must write a 1'b1 to the corresponding channel bit in the DMA write interrupt Clear register to clear this interrupt bit Note: You can write to this register to emulate interrupt generation, during software or hardware testing. A write to the address triggers an interrupt, but the DMA does not set the Done or Abort bits in this register. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
RSVDP_24	31:24	r	**Reserved for future use**

19.4.2.321 DMA Write Interrupt Mask Register

This register holds the Abort and Done DMA write interrupt mask.

DMA_WRITE_INT_MASK_OFF Offset address: 0007054_H

DMA Write Interrupt Mask Register Kernel Reset value: $000F\ 000F_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_24								0				WR_ABORT_INT_MASK			
r								r				rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVDP_8								0				WR_DONE_INT_MASK			
r								r				rw			

Field	Bits	Type	Description
WR_DONE_INT_MASK	3:0	rw	**Done Interrupt Mask** Prevents the Done interrupt status field in the DMA write interrupt status register from asserting the edma_int output. Each bit corresponds to a DMA channel. Bit [0] corresponds to channel 0. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
RSVDP_8	15:8	r	**Reserved for future use**
WR_ABORT_INT_MASK	19:16	rw	**Abort Interrupt Mask** Prevents the Abort interrupt status field in the DMA write interrupt status register from asserting the edma_int output. Each bit corresponds to a DMA channel. Bit [0] corresponds to channel 0. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
RSVDP_24	31:24	r	**Reserved for future use**
0	7:4, 23:20	r	**Reserved** Read as all 0's; should be written with all 0's.

19.4.2.322 DMA Write Interrupt Clear Register

You can write to this register to clear the Abort and Done write interrupts.

DMA_WRITE_INT_CLEAR_OFF Offset address: 0007058_H

DMA Write Interrupt Clear Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_24								0				WR_ABORT_INT_CLEAR			
r								r				rwh			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVDP_8								0				WR_DONE_INT_CLEAR			
r								r				rwh			

Field	Bits	Type	Description
WR_DONE_INT_CLEAR	3:0	rwh	**Done Interrupt Clear** You must write a 1'b1 to clear the corresponding bit in the Done interrupt status field of the DMA write interrupt status register. Each bit corresponds to a DMA channel. Bit [0] corresponds to channel 0. Note: Reading from this self-clearing register field always returns a '0'.
RSVDP_8	15:8	r	**Reserved for future use**
WR_ABORT_INT_CLEAR	19:16	rwh	**Abort Interrupt Clear** You must write a 1'b1 to clear the corresponding bit in the Abort interrupt status field of the DMA write interrupt status register. Each bit corresponds to a DMA channel. Bit [0] corresponds to channel 0. Note: Reading from this self-clearing register field always returns a '0'.
RSVDP_24	31:24	r	**Reserved for future use**
0	7:4, 23:20	r	**Reserved** Read as all 0's; should be written with all 0's.

19.4.2.323 DMA Write Error Status Register

This register provides information pertaining to errors incurred during write channel operation.

DMA_WRITE_ERR_STATUS_OFF Offset address: 000705C$_H$

DMA Write Error Status Register Kernel Reset value: 0000 0000$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_24								LINKLIST_ELEMENT_FETCH_ERR_DETECT							
r								rh							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVDP_8								APP_READ_ERR_DETECT							
r								rh							

Field	Bits	Type	Description
APP_READ_ERR_DETECT	7:0	rh	**Application Read Error Detected** The DMA write channel has received an error response from the AXI bus while reading data from it. Each bit corresponds to a DMA channel. Bit [0] corresponds to channel 0. • Enabling: For more details, see "Interrupts and Error Handling" • Masking: The DMA write interrupt Mask register has no effect on this register • Clearing: You must write a 1'b1 to the corresponding channel bit in the Abort interrupt field of the DMA Write Interrupt Clear Register (DMA_WRITE_INT_CLEAR_OFF) to clear this error bit.
RSVDP_8	15:8	r	**Reserved for future use**
LINKLIST_ELEMENT_FETCH_ERR_DETECT	23:16	rh	**Linked List Element Fetch Error Detected** The DMA write channel has received an error response from the AXI bus while reading a linked list element from local memory. Each bit corresponds to a DMA channel. Bit [0] corresponds to channel 0. • Enabling: For more details, see "Interrupts and Error Handling" • Masking: The DMA write interrupt Mask register has no effect on this register • Clearing: You must write a 1'b1 to the corresponding channel bit in the Abort interrupt field of the DMA Write Interrupt Clear Register (DMA_WRITE_INT_CLEAR_OFF) to clear this error bit.
RSVDP_24	31:24	r	**Reserved for future use**

19.4.2.324 DMA Write Done IMWr Address Low Register

This register holds the lower 32 bits of the Done IMWr TLP address.

DMA_WRITE_DONE_IMWR_LOW_OFF Offset address: 0007060$_H$

DMA Write Done IMWr Address Low Register Kernel Reset value: 0000 0000$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMA_WRITE_DONE_LOW_REG															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMA_WRITE_DONE_LOW_REG															
rw															

Field	Bits	Type	Description
DMA_WRITE_DONE_LOW_REG	31:0	rw	**The DMA uses this field to generate bits [31:0] of the address field for the Done IMWr TLP** Bits [1:0] must be '00' as this address must be dword aligned. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W

19.4.2.325 DMA Write Done IMWr Interrupt Address High Register

This register holds the higher 32 bits of the Done IMWr TLP address.

DMA_WRITE_DONE_IMWR_HIGH_OFF Offset address: 0007064_H

DMA Write Done IMWr Interrupt Address High Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMA_WRITE_DONE_HIGH_REG															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMA_WRITE_DONE_HIGH_REG															
rw															

Field	Bits	Type	Description
DMA_WRITE_DONE_HIGH_REG	31:0	rw	**The DMA uses this field to generate bits [63:32] of the address field for the Done IMWr TLP** Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W

19.4.2.326 DMA Write Abort IMWr Address Low Register

This register holds the lower 32 bits of Abort IMWr TLP.

DMA_WRITE_ABORT_IMWR_LOW_OFF Offset address: 0007068_H

DMA Write Abort IMWr Address Low Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMA_WRITE_ABORT_LOW_REG															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMA_WRITE_ABORT_LOW_REG															
rw															

Field	Bits	Type	Description
DMA_WRITE_ABORT_LOW_REG	31:0	rw	**The DMA uses this field to generate bits [31:0] of the address field for the Abort IMWr TLP it generates** Bits [1:0] must be '00' as this address must be dword aligned. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W

19.4.2.327 DMA Write Abort IMWr Address High Register

This register holds the higher 32 bits of Abort IMWr TLP.

DMA_WRITE_ABORT_IMWR_HIGH_OFF Offset address: $000706C_H$

DMA Write Abort IMWr Address High Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMA_WRITE_ABORT_HIGH_REG															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMA_WRITE_ABORT_HIGH_REG															
rw															

Field	Bits	Type	Description
DMA_WRITE_ABORT_HIGH_REG	31:0	rw	**The DMA uses this field to generate bits [63:32] of the address field for the Abort IMWr TLP** Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W

19.4.2.328 DMA Write Channel 1 and 0 IMWr Data Register

This register holds the Channel 1 and 0 IMWr Done or Abort TLP Data.

DMA_WRITE_CH01_IMWR_DATA_OFF Offset address: 0007070_H

DMA Write Channel 1 and 0 IMWr Data Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
WR_CHANNEL_1_DATA															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WR_CHANNEL_0_DATA															
rw															

Field	Bits	Type	Description
WR_CHANNEL_0_DATA	15:0	rw	**The DMA uses this field to generate the data field for the Done or Abort IMWr TLPs it generates for write channel 0** Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W

(table continues...)

(continued)

Field	Bits	Type	Description
WR_CHANNEL_1_DATA	31:16	rw	**The DMA uses this field to generate the data field for the Done or Abort IMWr TLPs it generates for write channel 1** Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W

19.4.2.329 DMA Write Channel 3 and 2 IMWr Data Register

This register holds the Channel 3 and 2 IMWr Done or Abort TLP Data.

DMA_WRITE_CH23_IMWR_DATA_OFF Offset address: 0007074_H

DMA Write Channel 3 and 2 IMWr Data Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
WR_CHANNEL_3_DATA															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WR_CHANNEL_2_DATA															
rw															

Field	Bits	Type	Description
WR_CHANNEL_2_DATA	15:0	rw	**The DMA uses this field to generate the data field for the Done or Abort IMWr TLPs it generates for write channel 2** Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
WR_CHANNEL_3_DATA	31:16	rw	**The DMA uses this field to generate the data field for the Done or Abort IMWr TLPs it generates for write channel 3** Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W

19.4.2.330 DMA Write Channel 5 and 4 IMWr Data Register

This register holds the Channel 5 and 4 IMWr Done or Abort TLP Data.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
WR_CHANNEL_5_DATA															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WR_CHANNEL_4_DATA															
rw															

Field	Bits	Type	Description
WR_CHANNEL_4_DATA	15:0	rw	**The DMA uses this field to generate the data field for the Done or Abort IMWr TLPs it generates for write channel 4** Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
WR_CHANNEL_5_DATA	31:16	rw	**The DMA uses this field to generate the data field for the Done or Abort IMWr TLPs it generates for write channel 5** Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W

19.4.2.331 DMA Write Channel 7 and 6 IMWr Data Register

This register holds the Channel 7 and 6 IMWr Done or Abort TLP Data.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
WR_CHANNEL_7_DATA															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WR_CHANNEL_6_DATA															
rw															

Field	Bits	Type	Description
WR_CHANNEL_6_DATA	15:0	rw	**The DMA uses this field to generate the data field for the Done or Abort IMWr TLPs it generates for write channel 6** Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
WR_CHANNEL_7_DATA	31:16	rw	**The DMA uses this field to generate the data field for the Done or Abort IMWr TLPs it generates for write channel 7** Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W

19.4.2.332 DMA Write Linked List Error Enable Register

The LIE and RIE bits in the LL element enable the channel "done" interrupts (local and remote). The LLLAIE and LLRAIE bits enable the channel "abort" interrupts (local and remote).

DMA_WRITE_LINKED_LIST_ERR_EN_OFF Offset address: 0007090_H

DMA Write Linked List Error Enable Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_24								0				WR_CHANNEL_LLLAIE			
r								r				rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVDP_8								0				WR_CHANNEL_LLRAIE			
r								r				rw			

Field	Bits	Type	Description
WR_CHANNEL_LLRAIE	3:0	rw	**Write Channel LL Remote Abort Interrupt Enable (LLRAIE)** You enable the write channel remote abort interrupt through this bit. The LIE and RIE bits in the LL element enable the write channel done interrupts. Each bit corresponds to a DMA channel. Bit [0] corresponds to channel 0. Used in linked list mode only. For more details, see "Interrupt Handling". Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
RSVDP_8	15:8	r	**Reserved for future use**
WR_CHANNEL_LLLAIE	19:16	rw	**Write Channel LL Local Abort Interrupt Enable (LLLAIE)** You enable the write channel local abort interrupt through this bit. The LIE and RIE bits in the LL element enable the write channel done interrupts. Each bit corresponds to a DMA channel. Bit [0] corresponds to channel 0. Used in linked list mode only. For more details, see "Interrupt Handling". Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
RSVDP_24	31:24	r	**Reserved for future use**
0	7:4, 23:20	r	**Reserved** Read as all 0's; should be written with all 0's.

19.4.2.333 DMA Read Interrupt Status Register

This register indicates the status of Abort and Done interrupts for the DMA read channels.

DMA_READ_INT_STATUS_OFF Offset address: $00070A0_H$

DMA Read Interrupt Status Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_24								RD_ABORT_INT_STATUS							
r								rwh							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVDP_8								RD_DONE_INT_STATUS							
r								rwh							

Field	Bits	Type	Description
RD_DONE_INT_STATUS	7:0	rwh	**Done Interrupt Status** The DMA read channel has successfully completed the DMA read transfer. Each bit corresponds to a DMA channel. Bit [0] corresponds to channel 0. • Enabling: For more details, see "Interrupts and Error Handling" • Masking: The DMA read interrupt Mask register has no effect on this register • Clearing: You must write a 1'b1 to the corresponding channel bit in the DMA read interrupt Clear register to clear this interrupt bit Note: You can write to this register to emulate interrupt generation, during software or hardware testing. A write to the address triggers an interrupt, but the DMA does not set the Done or Abort bits in this register. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
RSVDP_8	15:8	r	**Reserved for future use**
RD_ABORT_INT_STATUS	23:16	rwh	**Abort Interrupt Status** The DMA read channel has detected an error, or you manually stopped the transfer as described in "Stopping the DMA Transfer (Software Stop)". Each bit corresponds to a DMA channel. Bit [0] corresponds to channel 0. You can read the DMA Read Error Status Low Register (DMA_READ_ERR_STATUS_LOW_OFF) and DMA Read Error Status High Register (DMA_READ_ERR_STATUS_HIGH_OFF) to determine the source of the error. • Enabling: For more details, see "Interrupts and Error Handling" • Masking: The DMA read interrupt Mask register has no effect on this register • Clearing: You must write a 1'b1 to the corresponding channel bit in the DMA read interrupt Clear register to clear this interrupt bit. Note: You can write to this register to emulate interrupt generation, during software or hardware testing. A write to the address triggers an interrupt, but the DMA does not set the Done or Abort bits in this register. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
RSVDP_24	31:24	r	**Reserved for future use**

19.4.2.334 DMA Read Interrupt Mask Register

This register holds the Abort and Done DMA read interrupt mask.

DMA_READ_INT_MASK_OFF Offset address: $00070A8_H$

DMA Read Interrupt Mask Register Kernel Reset value: $000F\ 000F_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_24								0				RD_ABORT_INT_MASK			
r								r				rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVDP_8								0				RD_DONE_INT_MASK			
r								r				rw			

Field	Bits	Type	Description
RD_DONE_INT_MASK	3:0	rw	**Done Interrupt Mask** Prevents the Done interrupt status field in the DMA read interrupt status register from asserting the edma_int output. Each bit corresponds to a DMA channel. Bit [0] corresponds to channel 0. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
RSVDP_8	15:8	r	**Reserved for future use**
RD_ABORT_INT_MASK	19:16	rw	**Abort Interrupt Mask** Prevents the Abort interrupt status field in the DMA read interrupt status register from asserting the edma_int output. Each bit corresponds to a DMA channel. Bit [0] corresponds to channel 0. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
RSVDP_24	31:24	r	**Reserved for future use**
0	7:4, 23:20	r	**Reserved** Read as all 0's; should be written with all 0's.

19.4.2.335 DMA Read Interrupt Clear Register

You can write to this register to clear the Abort and Done read interrupts.

DMA_READ_INT_CLEAR_OFF Offset address: $00070AC_H$

DMA Read Interrupt Clear Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_24								RD_ABORT_INT_CLEAR							
r								rwh							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVDP_8								RD_DONE_INT_CLEAR							
r								rwh							

Field	Bits	Type	Description
RD_DONE_INT_CLEAR	7:0	rwh	**Done Interrupt Clear** You must write a 1'b1 to clear the corresponding bit in the Done interrupt status field of the DMA read interrupt status register. Each bit corresponds to a DMA channel. Bit [0] corresponds to channel 0. Note: Reading from this self-clearing register field always returns a '0'.
RSVDP_8	15:8	r	**Reserved for future use**
RD_ABORT_INT_CLEAR	23:16	rwh	**Abort Interrupt Clear** You must write a 1'b1 to clear the corresponding bit in the Abort interrupt status field of the DMA read interrupt status register. Each bit corresponds to a DMA channel. Bit [0] corresponds to channel 0. Note: Reading from this self-clearing register field always returns a '0'.
RSVDP_24	31:24	r	**Reserved for future use**

19.4.2.336 DMA Read Error Status Low Register

This register provides information pertaining to linked list element fetch error and application write error.

DMA_READ_ERR_STATUS_LOW_OFF Offset address: $00070B4_H$

DMA Read Error Status Low Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_24								LINK_LIST_ELEMENT_FETCH_ERR_DETECT							
r								rh							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVDP_8								APP_WR_ERR_DETECT							
r								rh							

Field	Bits	Type	Description
APP_WR_ERR_DETECT	7:0	rh	**Application Write Error Detected** The DMA read channel has received an error response from the AXI bus while writing data to it. This error is fatal. You must restart the transfer from the beginning, as the channel context is corrupted, and the transfer is not rolled back. For more details, see "Linked List Mode". Each bit corresponds to a DMA channel. Bit [0] corresponds to channel 0. • Enabling: For more details, see "Interrupts and Error Handling" • Masking: The DMA read interrupt Mask register has no effect on this register • Clearing: You must write a 1'b1 to the corresponding channel bit in the Abort interrupt field of the "DMA Read Interrupt Clear Register" (DMA_READ_INT_CLEAR_OFF) to clear this error bit Note, this clears all bits in this register, and also the DMA Read Error Status High register (DMA_READ_ERR_STATUS_HIGH_OFF).
RSVDP_8	15:8	r	**Reserved for future use**
LINK_LIST_ELEMENT_FETCH_ERR_DETECT	23:16	rh	**Linked List Element Fetch Error Detected** - The DMA read channel has received an error response from the AXI bus while reading a linked list element from local memory. Each bit corresponds to a DMA channel. Bit [0] corresponds to channel 0. • Enabling: For details, see "Interrupts and Error Handling" • Masking: The DMA read interrupt Mask register has no effect on this register • Clearing: You must write a 1'b1 to the corresponding channel bit in the Abort interrupt field of the DMA Read Interrupt Clear Register (DMA_READ_INT_CLEAR_OFF) to clear this error bit Note, this clears all bits in this register, and also the DMA Read Error Status High register (DMA_READ_ERR_STATUS_HIGH_OFF).
RSVDP_24	31:24	r	**Reserved for future use**

19.4.2.337 DMA Read Error Status High Register

This register provides information pertaining to completion errors.

DMA_READ_ERR_STATUS_HIGH_OFF Offset address: $00070B8_H$

DMA Read Error Status High Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DATA_POISIONING								CPL_TIMEOUT							
rh								rh							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CPL_ABORT								UNSUPPORTED_REQ							
rh								rh							

Field	Bits	Type	Description
UNSUPPORTED_REQ	7:0	rh	**Unsupported Request** The DMA read channel has received a PCIe unsupported request completion status from the remote device in response to the MRd request. For more details, see "Linked List Mode". Each bit corresponds to a DMA channel. Bit [0] corresponds to channel 0. • Enabling: For more details, see "Interrupts and Error Handling" • Masking: The DMA read interrupt Mask register has no effect on this register • Clearing: You must write a 1'b1 to the corresponding channel bit in the Abort interrupt field of the DMA Read Interrupt Clear Register (DMA_READ_INT_CLEAR_OFF) to clear this error bit. Note, this also clears the other error bits for the same channel in this register and in the DMA Read Error Status Low register.
CPL_ABORT	15:8	rh	**Completer Abort** The DMA read channel has received a PCIe completer abort completion status from the remote device in response to the MRd request. For more details, see "Linked List Mode". Each bit corresponds to a DMA channel. Bit [0] corresponds to channel 0. • Enabling: For more details, see "Interrupts and Error Handling" • Masking: The DMA read interrupt Mask register has no effect on this register • Clearing: You must write a 1'b1 to the corresponding channel bit in the Abort interrupt field of the DMA Read Interrupt Clear Register (DMA_READ_INT_CLEAR_OFF) to clear this error bit. Note, this also clears the other error bits for the same channel in this register and in the DMA Read Error Status Low register.

(table continues...)

(continued)

Field	Bits	Type	Description
CPL_TIMEOUT	23:16	rh	**Completion Time Out** The DMA read channel has timed-out while waiting for the remote device to respond to the MRd request, or a malformed CplD has been received. For more details, see "Linked List Mode". Each bit corresponds to a DMA channel. Bit [0] corresponds to channel 0. • Enabling: For details, see "Interrupts and Error Handling" • Masking: The DMA read interrupt Mask register has no effect on this register • Clearing: You must write a 1'b1 to the corresponding channel bit in the Abort interrupt field of the DMA Read Interrupt Clear Register (DMA_READ_INT_CLEAR_OFF) to clear this error bit Note, this also clears the other error bits for the same channel in this register and in the DMA Read Error Status Low register.
DATA_POISIONING	31:24	rh	**Data Poisoning** The DMA read channel has detected data poisoning in the completion from the remote device (in response to the MRd request). The DMA read channel will drop the completion and then be halted. The CX_FLT_MASK_UR_POIS filter rule does not affect this behavior. Each bit corresponds to a DMA channel. Bit [0] corresponds to channel 0. • Enabling: For more details, see "Interrupts and Error Handling" • Masking: The DMA read interrupt Mask register has no effect on this register • Clearing: You must write a 1'b1 to the corresponding channel bit in the Abort interrupt field of the DMA Read Interrupt Clear Register (DMA_READ_INT_CLEAR_OFF) to clear this error bit Note, this also clears the other error bits for the same channel in this register and in the DMA Read Error Status Low register.

19.4.2.338 DMA Read Linked List Error Enable Register

The LIE and RIE bits in the LL element enable the channel "done" interrupts (local and remote). The LLLAIE and LLRAIE bits enable the channel "abort" interrupts (local and remote).

DMA_READ_LINKED_LIST_ERR_EN_OFF Offset address: $00070C4_H$

DMA Read Linked List Error Enable Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_24								0				RD_CHANNEL_LLLAIE			
r								r				rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVDP_8								0				RD_CHANNEL_LLRAIE			
r								r				rw			

Field	Bits	Type	Description
RD_CHANNEL_LLRAIE	3:0	rw	**Read Channel LL Remote Abort Interrupt Enable (LLRAIE)** You enable the read channel Remote Abort interrupt through this bit. The LIE and RIE bits in the LL element enable the read channel done interrupts. Each bit corresponds to a DMA channel. Bit [0] corresponds to channel 0. Used in linked list mode only. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
RSVDP_8	15:8	r	**Reserved for future use**
RD_CHANNEL_LLLAIE	19:16	rw	**Read Channel LL Local Abort Interrupt Enable (LLLAIE)** You enable the read channel Local Abort interrupt through this bit. The LIE and RIE bits in the LL element enable the read channel done interrupts. Each bit corresponds to a DMA channel. Bit [0] corresponds to channel 0. Used in linked list mode only. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
RSVDP_24	31:24	r	**Reserved for future use**
0	7:4, 23:20	r	**Reserved** Read as all 0's; should be written with all 0's.

19.4.2.339 DMA Read Done IMWr Address Low Register

This register holds the lower 32 bits of the Done IMWr TLP address.

DMA_READ_DONE_IMWR_LOW_OFF Offset address: $00070CC_H$

DMA Read Done IMWr Address Low Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMA_READ_DONE_LOW_REG															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMA_READ_DONE_LOW_REG															
rw															

Field	Bits	Type	Description
DMA_READ_DONE_LOW_REG	31:0	rw	**The DMA uses this field to generate bits [31:0] of the address field for the Done IMWr TLP** Bits [1:0] must be '00' as this address must be dword aligned. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W

19.4.2.340 DMA Read Done IMWr Address High Register

This register holds the higher 32 bits of the Done IMWr TLP address.

DMA_READ_DONE_IMWR_HIGH_OFF Offset address: $00070D0_H$

DMA Read Done IMWr Address High Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMA_READ_DONE_HIGH_REG															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMA_READ_DONE_HIGH_REG															
rw															

Field	Bits	Type	Description
DMA_READ_DONE_HIGH_REG	31:0	rw	**The DMA uses this field to generate bits [63:32] of the address field for the Done IMWr TLP** Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W

19.4.2.341 DMA Read Abort IMWr Address Low Register

This register holds the lower 32 bits of the Abort IMWr TLP address.

DMA_READ_ABORT_IMWR_LOW_OFF Offset address: $00070D4_H$

DMA Read Abort IMWr Address Low Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMA_READ_ABORT_LOW_REG															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMA_READ_ABORT_LOW_REG															
rw															

Field	Bits	Type	Description
DMA_READ_ABORT_LOW_REG	31:0	rw	**The DMA uses this field to generate bits [31:0] of the address field for the Abort IMWr TLP** Bits [1:0] must be '00' as this address must be dword aligned. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W

19.4.2.342 DMA Read Abort IMWr Address High Register

This register holds the higher 32 bits of the Abort IMWr TLP address.

DMA_READ_ABORT_IMWR_HIGH_OFF Offset address: $00070D8_H$

DMA Read Abort IMWr Address High Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMA_READ_ABORT_HIGH_REG															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMA_READ_ABORT_HIGH_REG															
rw															

Field	Bits	Type	Description
DMA_READ_ABORT_HIGH_REG	31:0	rw	**The DMA uses this field to generate bits [63:32] of the address field for the Abort IMWr TLP** Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W

19.4.2.343 DMA Read Channel 1 and 0 IMWr Data Register

This register holds the Channel 1 and 0 IMWr Done or Abort TLP Data.

DMA_READ_CH01_IMWR_DATA_OFF Offset address: $00070DC_H$

DMA Read Channel 1 and 0 IMWr Data Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RD_CHANNEL_1_DATA															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RD_CHANNEL_0_DATA															
rw															

Field	Bits	Type	Description
RD_CHANNEL_0_DATA	15:0	rw	**The DMA uses this field to generate the data field for the Done or Abort IMWr TLPs it generates for read channel 0** Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
RD_CHANNEL_1_DATA	31:16	rw	**The DMA uses this field to generate the data field for the Done or Abort IMWr TLPs it generates for read channel 1** Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W

19.4.2.344 DMA Read Channel 3 and 2 IMWr Data Register

This register holds the Channel 3 and 2 IMWr Done or Abort TLP Data.

DMA_READ_CH23_IMWR_DATA_OFF Offset address: $00070E0_H$

DMA Read Channel 3 and 2 IMWr Data Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RD_CHANNEL_3_DATA															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RD_CHANNEL_2_DATA															
rw															

Field	Bits	Type	Description
RD_CHANNEL_2_DATA	15:0	rw	**The DMA uses this field to generate the data field for the Done or Abort IMWr TLPs it generates for read channel 2** Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
RD_CHANNEL_3_DATA	31:16	rw	**The DMA uses this field to generate the data field for the Done or Abort IMWr TLPs it generates for read channel 3** Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W

19.4.2.345 DMA Read Channel 5 and 4 IMWr Data Register

This register holds the Channel 5 and 4 IMWr Done or Abort TLP Data.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RD_CHANNEL_5_DATA															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RD_CHANNEL_4_DATA															
rw															

Field	Bits	Type	Description
RD_CHANNEL_4_DATA	15:0	rw	**The DMA uses this field to generate the data field for the Done or Abort IMWr TLPs it generates for read channel 4** Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
RD_CHANNEL_5_DATA	31:16	rw	**The DMA uses this field to generate the data field for the Done or Abort IMWr TLPs it generates for read channel 5** Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W

19.4.2.346 DMA Read Channel 7 and 6 IMWr Data Register

This register holds the Channel 7 and 6 IMWr Done or Abort TLP Data.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RD_CHANNEL_7_DATA															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RD_CHANNEL_6_DATA															
rw															

Field	Bits	Type	Description
RD_CHANNEL_6_DATA	15:0	rw	**The DMA uses this field to generate the data field for the Done or Abort IMWr TLPs it generates for read channel 6** Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
RD_CHANNEL_7_DATA	31:16	rw	**The DMA uses this field to generate the data field for the Done or Abort IMWr TLPs it generates for read channel 7** Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W

19.4.2.347 DMA Write Engine Handshake Counter Channel 0/1/2/3 Register

This register holds the handshake counters write channels 0 - 3.

DMA_WRITE_ENGINE_HSHAKE_CNT_LOW_OFF Offset address: 0007108_H

DMA Write Engine Handshake Counter Channel 0/1/2/3 Register — Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_29			DMA_WRITE_ENGINE_HSHAKE_CNT_CH3					RSVDP_21			DMA_WRITE_ENGINE_HSHAKE_CNT_CH2				
r			r					r			r				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVDP_13			DMA_WRITE_ENGINE_HSHAKE_CNT_CH1					RSVDP_5			DMA_WRITE_ENGINE_HSHAKE_CNT_CH0				
r			r					r			r				

Field	Bits	Type	Description
DMA_WRITE_ENGINE_HSHAKE_CNT_CH0	4:0	r	**DMA handshake counter for DMA Write Engine Channel 0** The data transfer in Linked List mode starts only when the counter is non-zero.
RSVDP_5	7:5	r	**Reserved for future use**
DMA_WRITE_ENGINE_HSHAKE_CNT_CH1	12:8	r	**DMA handshake counter for DMA Write Engine Channel 1** The data transfer in Linked List mode starts only when the counter is non-zero.
RSVDP_13	15:13	r	**Reserved for future use**
DMA_WRITE_ENGINE_HSHAKE_CNT_CH2	20:16	r	**DMA handshake counter for DMA Write Engine Channel 2** The data transfer in Linked List mode starts only when the counter is non-zero.
RSVDP_21	23:21	r	**Reserved for future use**
DMA_WRITE_ENGINE_HSHAKE_CNT_CH3	28:24	r	**DMA handshake counter for DMA Write Engine Channel 3** The data transfer in Linked List mode starts only when the counter is non-zero.
RSVDP_29	31:29	r	**Reserved for future use**

19.4.2.348 DMA Write Engine Handshake Counter Channel 4/5/6/7 Register

This register holds the handshake counters write channels 4 - 7.

31 30 29	28 27 26 25 24	23 22 21	20 19 18 17 16
RSVDP_29	DMA_WRITE_ENGINE_HSHAKE_CNT_CH7	RSVDP_21	DMA_WRITE_ENGINE_HSHAKE_CNT_CH6
r	r	r	r

15 14 13	12 11 10 9 8	7 6 5	4 3 2 1 0
RSVDP_13	DMA_WRITE_ENGINE_HSHAKE_CNT_CH5	RSVDP_5	DMA_WRITE_ENGINE_HSHAKE_CNT_CH4
r	r	r	r

Field	Bits	Type	Description
DMA_WRITE_ENGINE_HSHAKE_CNT_CH4	4:0	r	**DMA handshake counter for DMA Write Engine Channel 4** The data transfer in Linked List mode starts only when the counter is non-zero.
RSVDP_5	7:5	r	**Reserved for future use**
DMA_WRITE_ENGINE_HSHAKE_CNT_CH5	12:8	r	**DMA handshake counter for DMA Write Engine Channel 5** The data transfer in Linked List mode starts only when the counter is non-zero.
RSVDP_13	15:13	r	**Reserved for future use**
DMA_WRITE_ENGINE_HSHAKE_CNT_CH6	20:16	r	**DMA handshake counter for DMA Write Engine Channel 6** The data transfer in Linked List mode starts only when the counter is non-zero.
RSVDP_21	23:21	r	**Reserved for future use**
DMA_WRITE_ENGINE_HSHAKE_CNT_CH7	28:24	r	**DMA handshake counter for DMA Write Engine Channel 7** The data transfer in Linked List mode starts only when the counter is non-zero.
RSVDP_29	31:29	r	**Reserved for future use**

19.4.2.349 DMA Read Engine Handshake Counter Channel 0/1/2/3 Register

This register holds the handshake counters read channels 0 - 3.

DMA_READ_ENGINE_HSHAKE_CNT_LOW_OFF Offset address: 0007118_H

DMA Read Engine Handshake Counter Channel 0/1/2/3 Register — Kernel Reset value: $0000\ 0000_H$

31 30 29	28 27 26 25 24	23 22 21	20 19 18 17 16
RSVDP_29	DMA_READ_ENGINE_HSHAKE_CNT_CH3	RSVDP_21	DMA_READ_ENGINE_HSHAKE_CNT_CH2
r	r	r	r

15 14 13	12 11 10 9 8	7 6 5	4 3 2 1 0
RSVDP_13	DMA_READ_ENGINE_HSHAKE_CNT_CH1	RSVDP_5	DMA_READ_ENGINE_HSHAKE_CNT_CH0
r	r	r	r

Field	Bits	Type	Description
DMA_READ_ENGINE_HSHAKE_CNT_CH0	4:0	r	**DMA handshake counter for DMA Read Engine Channel 0** Tthe data transfer in Linked List mode starts only when the counter is non-zero.
RSVDP_5	7:5	r	**Reserved for future use**
DMA_READ_ENGINE_HSHAKE_CNT_CH1	12:8	r	**DMA handshake counter for DMA Read Engine Channel 1** The data transfer in Linked List mode starts only when the counter is non-zero.
RSVDP_13	15:13	r	**Reserved for future use**
DMA_READ_ENGINE_HSHAKE_CNT_CH2	20:16	r	**DMA handshake counter for DMA Read Engine Channel 2** The data transfer in Linked List mode starts only when the counter is non-zero.
RSVDP_21	23:21	r	**Reserved for future use**
DMA_READ_ENGINE_HSHAKE_CNT_CH3	28:24	r	**DMA handshake counter for DMA Read Engine Channel 3** The data transfer in Linked List mode starts only when the counter is non-zero.
RSVDP_29	31:29	r	**Reserved for future use**

19.4.2.350 DMA Read Engine Handshake Counter Channel 4/5/6/7 Register

This register holds the handshake counters read channels 4 - 7.

31 30 29	28 27 26 25 24	23 22 21	20 19 18 17 16
RSVDP_29	DMA_READ_ENGINE_HSHAKE_CNT_CH7	RSVDP_21	DMA_READ_ENGINE_HSHAKE_CNT_CH6
r	r	r	r

15 14 13	12 11 10 9 8	7 6 5	4 3 2 1 0
RSVDP_13	DMA_READ_ENGINE_HSHAKE_CNT_CH5	RSVDP_5	DMA_READ_ENGINE_HSHAKE_CNT_CH4
r	r	r	r

Field	Bits	Type	Description
DMA_READ_ENGINE_HSHAKE_CNT_CH4	4:0	r	**DMA handshake counter for DMA Read Engine Channel 4** The data transfer in Linked List mode starts only when the counter is non-zero.
RSVDP_5	7:5	r	**Reserved for future use**
DMA_READ_ENGINE_HSHAKE_CNT_CH5	12:8	r	**DMA handshake counter for DMA Read Engine Channel 5** The data transfer in Linked List mode starts only when the counter is non-zero.
RSVDP_13	15:13	r	**Reserved for future use**
DMA_READ_ENGINE_HSHAKE_CNT_CH6	20:16	r	**DMA handshake counter for DMA Read Engine Channel 6** The data transfer in Linked List mode starts only when the counter is non-zero.

(table continues...)

(continued)

Field	Bits	Type	Description
RSVDP_21	23:21	r	**Reserved for future use**
DMA_READ_ENGINE_HSHAKE_CNT_CH7	28:24	r	**DMA handshake counter for DMA Read Engine Channel 7** The data transfer in Linked List mode starts only when the counter is non-zero.
RSVDP_29	31:29	r	**Reserved for future use**

19.4.2.351 DMA Write Channel i Power Enable Register

This register controls power enable/disable for write channel 0 .

DMA_WRITE_CHi_PWR_EN_OFF (i=0-3) Offset address: 0007128_H+i*4

DMA Write Channel i Power Enable Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVDP_1															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVDP_1															DMA_WRITE_CH0_PWR_EN
r															rw

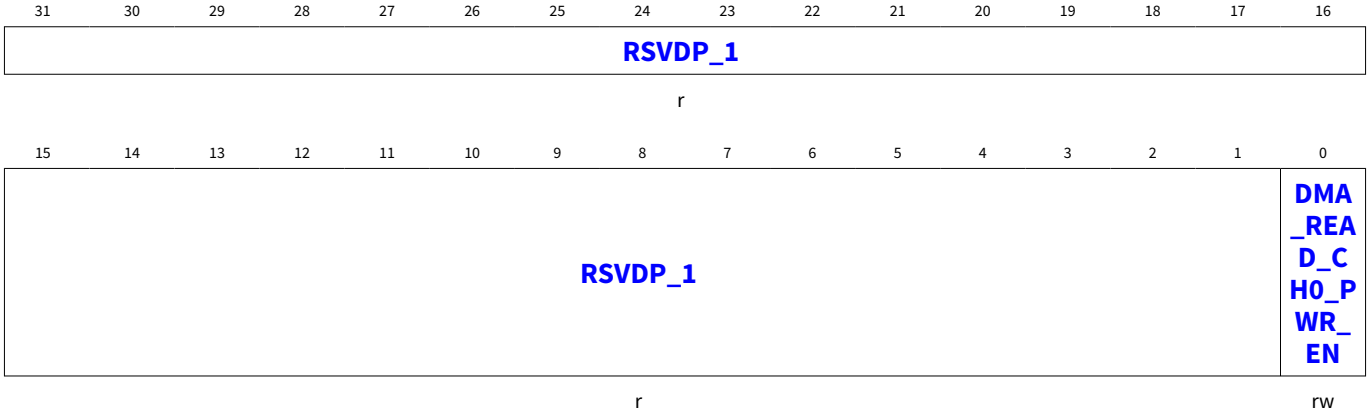
Field	Bits	Type	Description
DMA_WRITE_CH0_PWR_EN	0	rw	**DMA Write Channel 0 Power enable/disable** 0_B **DISABLE**: Disable 1_B **ENABLE**: Enable
RSVDP_1	31:1	r	**Reserved for future use**

19.4.2.352 DMA Read Channel i Power Enable Register

This register controls power enable/disable for read channel 0.

DMA_READ_CHi_PWR_EN_OFF (i=0-3) Offset address: 0007168_H+i*4

DMA Read Channel i Power Enable Register Kernel Reset value: $0000\ 0000_H$



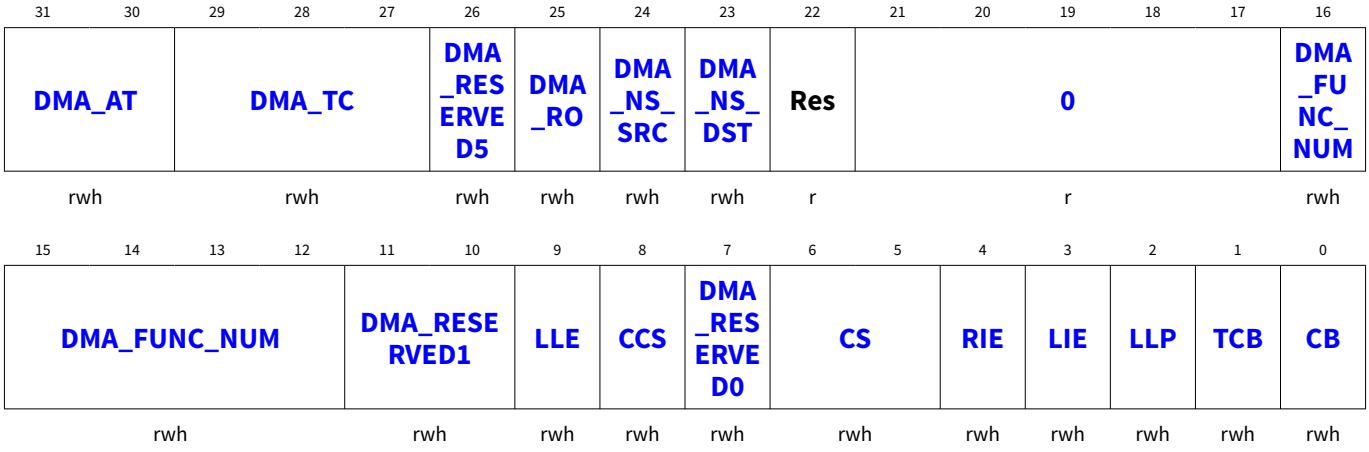
Field	Bits	Type	Description
DMA_READ_CH0_PWR_EN	0	rw	**DMA Read Channel 0 Power enable/disable** 0_B **DISABLE**: Disable 1_B **ENABLE**: Enable
RSVDP_1	31:1	r	**Reserved for future use**

19.4.2.353 DMA Write Channel i Control 1 Register

This register controls the DMA write channel operation.

DMA_CH_CONTROL1_OFF_WRCH_i (i=0-3) Offset address: 0007200_H+i*200_H

DMA Write Channel i Control 1 Register Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
CB	0	rwh	**Cycle Bit (CB)** Used in linked list mode only. It is used to synchronize the producer (software) and the consumer (DMA). For more details, see "PCS-CCS-CB-TCB Producer-Consumer Synchronization". The DMA loads this field with the CB of the linked list element. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W

(table continues...)

(continued)

Field	Bits	Type	Description
TCB	1	rwh	**Toggle Cycle Bit (TCB)** Indicates to the DMA to toggle its interpretation of the CB. Used in linked list mode only. It is used to synchronize the producer (software) and the consumer (DMA). For more details, see "PCS-CCS-CB-TCB Producer-Consumer Synchronization". The DMA loads this field with the TCB of the linked list element. this field is not defined in a data LL element. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
LLP	2	rwh	**Load Link Pointer (LLP)** Used in linked list mode only. Indicates that this linked list element is a link element, and its LL element pointer dwords are pointing to the next (non-contiguous) element. The DMA loads this field with the LLP of the linked list element. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
LIE	3	rwh	**Local Interrupt Enable (LIE)** You must set this bit to enable the generation of the Done or Abort Local interrupts. For more details, see "Interrupts and Error Handling". In LL mode, the DMA overwrites this with the LIE of the LL element. The LIE bit in a LL element only enables the Done interrupt. In non-LL mode, the LIE bit enables the Done and Abort interrupts. This field is not defined in a link LL element. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
RIE	4	rwh	**Remote Interrupt Enable (RIE)** You must set this bit to enable the generation of the Done or Abort Remote interrupts. For more details, see "Interrupts and Error Handling". In LL mode, the DMA overwrites this with the RIE of the LL element. The RIE bit in a LL element only enables the Done interrupt. In non-LL mode, the RIE bit enables the Done and Abort interrupts. This field is not defined in a link LL element. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
CS	6:5	rwh	**Channel Status (CS)** The channel status bits identify the current operational state of the DMA channel. The operation state encoding for each DMA channel is a s follows: 00_B **RESERVED**: Reserved 01_B **RUNNING**: This channel is active and transferring data 10_B **HALTED**: An error condition has been detected, and the DMA has stopped this channel 11_B **STOPPED**: The DMA has transferred all data for this channel, or you have prematurely stopped this channel by writing to the Stop field of the DMA Write Doorbell Register (DMA_WRITE_DOORBELL_OFF) or DMA Read Doorbell Register (DMA_READ_DOORBELL_OFF)
DMA_RESERVED0	7	rwh	**Reserved** Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W

(table continues...)

(continued)

Field	Bits	Type	Description
CCS	8	rwh	**Consumer Cycle State (CCS)** Used in linked list mode only. It is used to synchronize the producer (software) and the consumer (DMA). For more details, see "PCS-CCS-CB-TCB Producer-Consumer Synchronization". You must initialize this bit. The DMA updates this bit during linked list operation. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
LLE	9	rwh	**Linked List Enable (LLE)** Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W 0_B **DISABLE**: Disable linked list operation 1_B **ENABLE**: Enable linked list operation
DMA_RESERVED1	11:10	rwh	**Reserved** Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
DMA_FUNC_NUM	16:12	rwh	**DMA Channel Physical Function Number** This field indicates the physical function associated with the DMA channel. The controller uses this field to form the requester ID of the generated MRd/MWr DMA TLPs. Note: The behavior is undefined if any value outside of the valid range is set for this field Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W 00_H **Min_Val**: Physical Channel Function Number starts at 0 $1F_H$ **Max_Val**: The maximum number of Physical Functions, so the maximum value of this field is 0
DMA_NS_DST	23	rwh	**Destination No Snoop TLP Header Bit** The DMA uses this TLP header field when generating MWr (DAR addressing space) (not IMWr) TLPs. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
DMA_NS_SRC	24	rwh	**Source No Snoop TLP Header Bit** The DMA uses this TLP header field when generating MRd (SAR addressing space) (not IMWr) TLPs. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
DMA_RO	25	rwh	**Relaxed Ordering TLP Header Bit (RO)** DMA uses this TLP header field when generating MRd/MWr (not IMWr) TLPs. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
DMA_RESERVED5	26	rwh	**Reserved** Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
DMA_TC	29:27	rwh	**Traffic Class TLP Header Bit (TC)** The DMA uses this TLP header field when generating MRd/MWr (not IMWr) TLPs. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W

(table continues...)

(continued)

Field	Bits	Type	Description
DMA_AT	31:30	rwh	**Address Translation TLP Header Bit (AT)** The DMA uses this TLP header field when generating MRd/MWr (not IMWr) TLPs. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
0	21:17	r	**Reserved** Read as all 0's; should be written with all 0's.

19.4.2.354 DMA CH i Write Transfer Size Register

This register holds the DMA write transfer size.

DMA_TRANSFER_SIZE_OFF_WRCH_i (i=0-3) Offset address: $0007208_H+i*200_H$

DMA CH i Write Transfer Size Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMA_TRANSFER_SIZE															
rwh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMA_TRANSFER_SIZE															
rwh															

Field	Bits	Type	Description
DMA_TRANSFER_SIZE	31:0	rwh	**DMA Transfer Size** You program this register with the size of the DMA transfer. The maximum DMA transfer size is 4Gbytes. The minimum transfer size is one byte (0x1). This field is automatically decremented by the DMA as the DMA write channel transfer progresses. This field indicates the number bytes remaining to be transferred. When all bytes are successfully transferred the current transfer size is zero. In LL mode, the DMA overwrites this register with the corresponding dword of the LL element. You can read this register to monitor the transfer progress, however in some scenarios this register is updated after a delay. For example, when less than 3 channels are doorbelled, this register is updated only after a descriptor finishes(linked list mode), or the transfer ends (non-linked list mode). Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W

19.4.2.355 DMA CH i Write SAR Low Register

This register holds the lower 32 bits of the Source Address Register (SAR).

DMA_SAR_LOW_OFF_WRCH_i (i=0-3) Offset address: $000720C_H+i*200_H$

DMA CH i Write SAR Low Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SRC_ADDR_REG_LOW															
rwh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SRC_ADDR_REG_LOW															
rwh															

Field	Bits	Type	Description
SRC_ADDR_REG_LOW	31:0	rwh	**Source Address Register (lower 32 bits)** Indicates the next address to be read from. The DMA increments the SAR as the DMA transfer progresses. In LL mode, the DMA overwrites this with the corresponding dword of the LL element. - DMA Read: The SAR is the address of the remote memory. - DMA Write: The SAR is the address of the local memory. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W

19.4.2.356 DMA CH i Write SAR High Register

This register holds the higher 32 bits of the Source Address Register (SAR).

DMA_SAR_HIGH_OFF_WRCH_i (i=0-3) Offset address: $0007210_H+i*200_H$

DMA CH i Write SAR High Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SRC_ADDR_REG_HIGH															
rwh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SRC_ADDR_REG_HIGH															
rwh															

Field	Bits	Type	Description
SRC_ADDR_REG_HIGH	31:0	rwh	**Source Address Register (higher 32 bits)** In LL mode, the DMA overwrites this with the corresponding dword of the LL element. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W

19.4.2.357 DMA CH i Write DAR Low Register

This register holds the lower 32 bits of the Destination Address Register (DAR).

DMA_DAR_LOW_OFF_WRCH_i (i=0-3) Offset address: $0007214_H+i*200_H$

DMA CH i Write DAR Low Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DST_ADDR_REG_LOW															
rwh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DST_ADDR_REG_LOW															
rwh															

Field	Bits	Type	Description
DST_ADDR_REG_LOW	31:0	rwh	**Destination Address Register (lower 32 bits)** Indicates the next address to be written to. The DMA increments the DAR as the DMA transfer progresses. In LL mode, the DMA overwrites this with the corresponding dword of the LL element. - DMA Read: The DAR is the address of the local memory. - DMA Write: The DAR is the address of the remote memory. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W

19.4.2.358 DMA CH i Write DAR High Register

This register holds the higher 32 bits of the Destination Address Register (DAR).

DMA_DAR_HIGH_OFF_WRCH_i (i=0-3) Offset address: $0007218_H+i*200_H$

DMA CH i Write DAR High Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DST_ADDR_REG_HIGH															
rwh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DST_ADDR_REG_HIGH															
rwh															

Field	Bits	Type	Description
DST_ADDR_REG_HIGH	31:0	rwh	**Destination Address Register (Higher 32 bits)** In LL mode, the DMA overwrites this with the corresponding dword of the LL element. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W

19.4.2.359 DMA CH i Write Linked List Pointer Low Register

This register holds the lower 32 bits of the DMA write linked list pointer.

DMA_LLP_LOW_OFF_WRCH_i (i=0-3) Offset address: $000721C_H+i*200_H$

DMA CH i Write Linked List Pointer Low Register Kernel Reset value: $0000\ 0000_H$

Field	Bits	Type	Description
LLP_LOW	31:0	rwh	**Lower bits of the address of the linked list transfer list in local memory** Used in linked list mode only. When to point to the next element in the transfer list after the previous element is consumed. • When the current element is a data element; this field is incremented by 6 DWORDs. • When the current element is a link element; this field is overwritten by the LL Element Pointer of the element. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W

19.4.2.360 DMA CH i Write Linked List Pointer High Register

This register holds the higher 32 bits of the DMA write linked list pointer.

DMA_LLP_HIGH_OFF_WRCH_i (i=0-3) Offset address: $0007220_H+i*200_H$

DMA CH i Write Linked List Pointer High Register Kernel Reset value: $0000\ 0000_H$

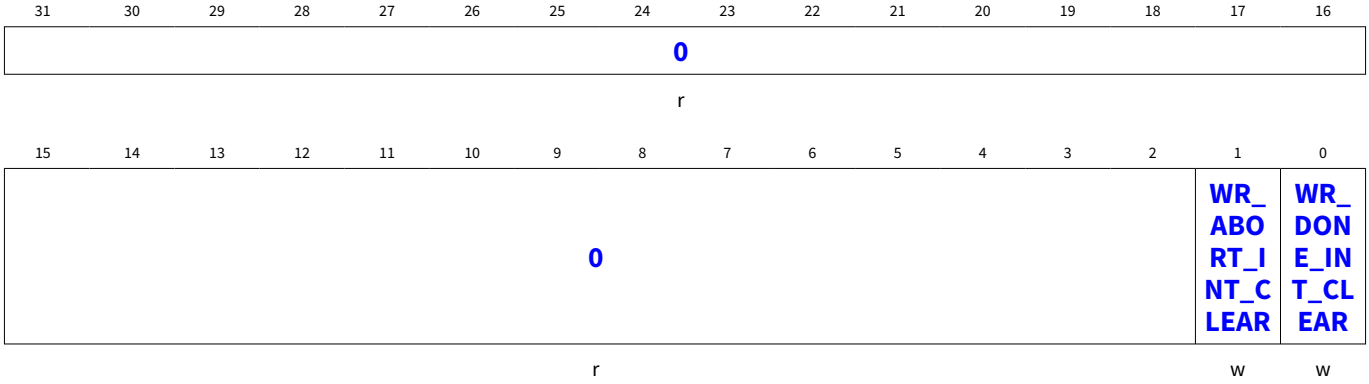
Field	Bits	Type	Description
LLP_HIGH	31:0	rwh	**Higher 32 bits of the address of the linked list transfer list in local memory** Used in linked list mode only. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W

19.4.2.361 Interrupt clear for DMA write channel i

This register has dedicated control over a DMA channel and it has the same functionality as DMA Write Interrupt Clear Register.

DMACAP_CHi_WRCHN_WR_INT_CLEAR (i=0-3) Offset address: $0007224_H+i*200_H$

Interrupt clear for DMA write channel i Kernel Reset value: $0000\ 0000_H$



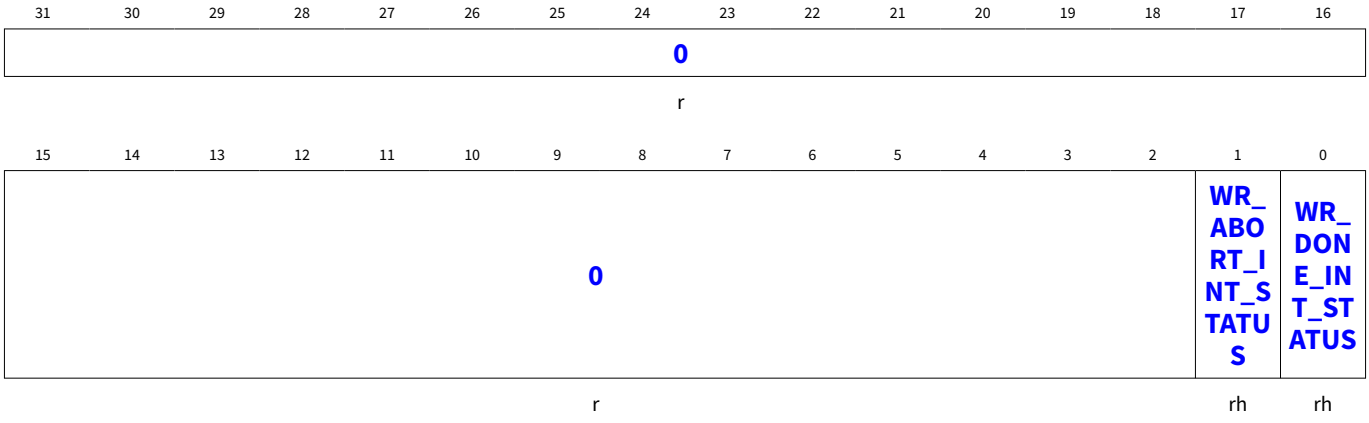
Field	Bits	Type	Description
WR_DONE_INT_CLEAR	0	w	**Done Interrupt Clear for the DMA channel i** You must write a 1'b1 to clear the corresponding bit in the Done interrupt status field of the DMA write interrupt status register. Note: Reading from this self-clearing register field always returns a '0'. 0_B No effect 1_B Clear status
WR_ABORT_INT_CLEAR	1	w	**Abort Interrupt Clear for the DMA channel i** You must write a 1'b1 to clear the corresponding bit in the Abort interrupt status field of the DMA write interrupt status register. Note: Reading from this self-clearing register field always returns a '0'. 0_B No effect 1_B Clear status
0	31:2	r	**Reserved** Read as 0; should be written with 0.

19.4.2.362 Interrupt status for DMA write channel i

This register has dedicated control over a DMA channel and it has the same functionality as DMA Write Interrupt Status Register.

DMACAP_CHi_WRCHN_WR_INT_STATUS (i=0-3) Offset address: $0007228_H+i*200_H$

Interrupt status for DMA write channel i Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
WR_DONE_INT_STATUS	0	rh	**Done Interrupt Status for the DMA channel i** The DMA write channel has successfully completed the DMA transfer.
WR_ABORT_INT_STATUS	1	rh	**Abort Interrupt Status for the DMA channel i** The DMA write channel has detected an error, or you manually stopped the transfer.
0	31:2	r	**Reserved** Read as 0; should be written with 0.

19.4.2.363 Error status for DMA write channel i

This register has dedicated control over a DMA channel and it has the same functionality as DMA Write Error Status Register.

DMACAP_CHi_WRCHN_WR_ERR_STATUS (i=0-3) Offset address: $000722C_H+i*200_H$

Error status for DMA write channel i Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0														LINKLIST_ELEMENT_FETCH_ERR_DETECT	APP_READ_ERR_DETECT
r														rh	rh

Field	Bits	Type	Description
APP_READ_ERR_DETECT	0	rh	**Application Read Error Detected for the DMA channel i** The DMA write channel has received an error response from the AXI bus while reading data from it. . - Clearing: You must write a 1'b1 to the WR_ABORT_INT_CLEAR to clear this error bit.
LINKLIST_ELEMENT_FETCH_ERR_DETECT	1	rh	**Linked List Element Fetch Error Detected for the DMA channel i** The DMA write channel has received an error response from the AXI bus while reading a linked list element from local memory.- Clearing: You must write a 1'b1 to the corresponding channel bit in the WR_ABORT_INT_CLEAR to clear this error bit.
0	31:2	r	**Reserved** Read as 0; should be written with 0.

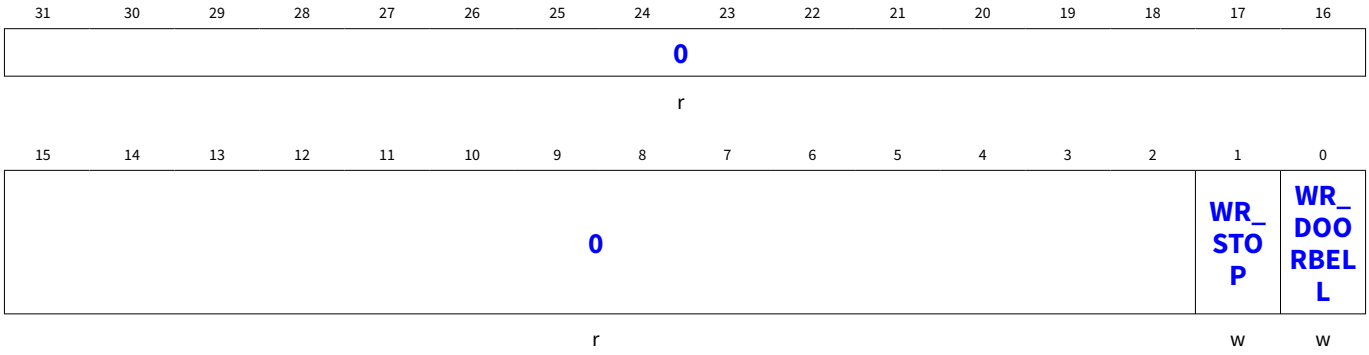
19.4.2.364 Doorbell for DMA write channel i

This register has dedicated control over a DMA channel and it has the same functionality as DMA Write Doorbell Register.

Note: This register is strictly a write only register. Reading this register may result in undefined behavior.

DMACAP_CHi_WRCHN_WR_DOORBELL (i=0-3) Offset address: $0007230_H+i*200_H$

Doorbell for DMA write channel i Kernel Reset value: $0000\ 0000_H$



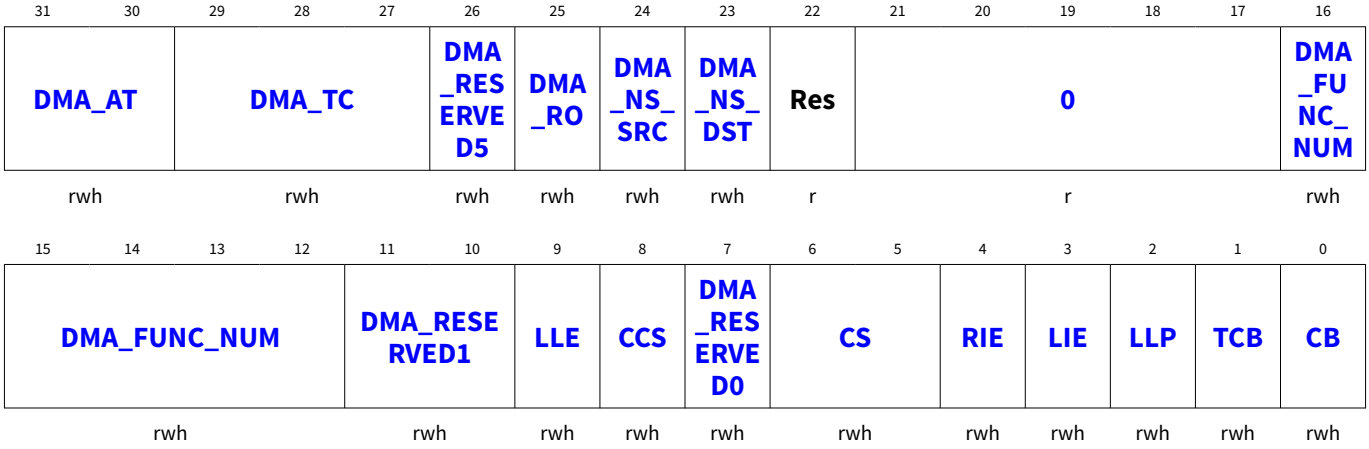
Field	Bits	Type	Description
WR_DOORBELL	0	w	**Doorbell Number for the DMA channel i** You must write the channel number 1'b1 to this register to start the DMA write transfer for that channel. Note: A write to this field triggers the controller to exit L1 substates 0_B No effect 1_B Start DMA transfer
WR_STOP	1	w	**Stop for the DMA channel i** The DMA write channel stops issuing requests, sets the channel status to "Stopped", and asserts the "Abort" interrupt if it is enabled. Before setting the Stop bit, you must read the channel Status field (CS) of the DMA Write Channel Control 1 Register (DMA_CH_CONTROL1_OFF_WRCH_0) to ensure that the write channel is "Running" (transferring data). Note: Use this feature with caution. DMA stop feature is not meant to be used in operational mode. It is only a debug feature and has associated hazards. 0_B No effect 1_B Stop the DMA
0	31:2	r	**Reserved** Read as 0; should be written with 0.

19.4.2.365 DMA Read Channel i Control 1 Register

This register controls the DMA read channel operation.

DMA_CH_CONTROL1_OFF_RDCH_i (i=0-3) Offset address: $0007300_H+i*200_H$

DMA Read Channel i Control 1 Register Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
CB	0	rwh	**Cycle Bit (CB)** Used in linked list mode only. It is used to synchronize the producer (software) and the consumer (DMA). For more details, see "PCS-CCS-CB-TCB Producer-Consumer Synchronization". The DMA loads this field with the CB of the linked list element. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
TCB	1	rwh	**Toggle Cycle Bit (TCB)** Indicates to the DMA to toggle its interpretation of the CB. Used in linked list mode only. It is used to synchronize the producer (software) and the consumer (DMA). For more details, see "PCS-CCS-CB-TCB Producer-Consumer Synchronization". The DMA loads this field with the TCB of the linked list element. This field is not defined in a data LL element. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
LLP	2	rwh	**Load Link Pointer (LLP)** Used in linked list mode only. Indicates that this linked list element is a link element, and its LL element pointer dwords are pointing to the next (non-contiguous) element. The DMA loads this field with the LLP of the linked list element. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
LIE	3	rwh	**Local Interrupt Enable (LIE)** You must set this bit to enable the generation of the Done or Abort Local interrupts. For more details, see "Interrupts and Error Handling". In LL mode, the DMA overwrites this with the LIE of the LL element. The LIE bit in a LL element only enables the Done interrupt. In non-LL mode, the LIE bit enables the Done and Abort interrupts. This field is not defined in a link LL element. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W

(table continues...)

(continued)

Field	Bits	Type	Description
RIE	4	rwh	**Remote Interrupt Enable (RIE)** You must set this bit to enable the generation of the Done or Abort Remote interrupts. For more details, see "Interrupts and Error Handling". In LL mode, the DMA overwrites this with the RIE of the LL element. The RIE bit in a LL element only enables the Done interrupt. In non-LL mode, the RIE bit enables the Done and Abort interrupts. This field is not defined in a link LL element. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
CS	6:5	rwh	**Channel Status (CS)** The channel status bits identify the current operational state of the DMA channel. The operation state encoding for each DMA channel is a s follows: 00_B **RSVD**: Reserved 01_B **RUN**: Running This channel is active and transferring data. 10_B **HALT**: Halted An error condition has been detected, and the DMA has stopped this channel. 11_B **STOP**: Stopped The DMA has transferred all data for this channel, or you have prematurely stopped this channel by writing to the Stop field of the DMA Read Doorbell Register (DMA_WRITE_DOORBELL_OFF) or DMA Read Doorbell Register (DMA_READ_DOORBELL_OFF).
DMA_RESERVED0	7	rwh	**Reserved** Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
CCS	8	rwh	**Consumer Cycle State (CCS)** Used in linked list mode only. It is used to synchronize the producer (software) and the consumer (DMA). For more details, see "PCS-CCS-CB-TCB Producer-Consumer Synchronization". You must initialize this bit. The DMA updates this bit during linked list operation. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
LLE	9	rwh	**Linked List Enable (LLE)** Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W 0_B **DISABLE**: Disable linked list operation 1_B **ENABLE**: Enable linked list operation
DMA_RESERVED1	11:10	rwh	**Reserved** Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W

(table continues...)

(continued)

Field	Bits	Type	Description
DMA_FUNC_NUM	16:12	rwh	**DMA Channel Physical Function Number** This field indicates the physical function associated with the DMA channel. The controller uses this field to form the requester ID of the generated MRd/MWr DMA TLPs. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W 00_H **Min_Val**: Physical Channel Function Number starts at 0 $1F_H$ **Max_Val**: The maximum number of Physical Functions, so the maximum value of this field is 0
DMA_NS_DST	23	rwh	**Destination No Snoop TLP Header Bit** The DMA uses this TLP header field when generating MWr (DAR addressing space) (not IMWr) TLPs. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
DMA_NS_SRC	24	rwh	**Source No Snoop TLP Header Bit** The DMA uses this TLP header field when generating MRd (SAR addressing space) (not IMWr) TLPs. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
DMA_RO	25	rwh	**Relaxed Ordering TLP Header Bit (RO) The DMA uses this TLP header field when generating MRd/MWr (not IMWr) TLPs** Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
DMA_RESERVED5	26	rwh	**Reserved** Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
DMA_TC	29:27	rwh	**Traffic Class TLP Header Bit (TC)** The DMA uses this TLP header field when generating MRd/MWr (not IMWr) TLPs. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
DMA_AT	31:30	rwh	**Address Translation TLP Header Bit (AT)** The DMA uses this TLP header field when generating MRd/MWr (not IMWr) TLPs. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W
0	21:17	r	**Reserved** Read as all 0's; should be written with all 0's.

19.4.2.366 DMA Read CH i Transfer Size Register

This register holds the DMA read transfer size.

DMA_TRANSFER_SIZE_OFF_RDCH_i (i=0-3) Offset address: $0007308_H+i*200_H$

DMA Read CH i Transfer Size Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMA_TRANSFER_SIZE															
rwh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMA_TRANSFER_SIZE															
rwh															

Field	Bits	Type	Description
DMA_TRANSFER_SIZE	31:0	rwh	**DMA Transfer Size** You program this register with the size of the DMA transfer. The maximum DMA transfer size is 4Gbytes. The minimum transfer size is one byte (0x1). This field is automatically decremented by the DMA as the DMA read channel transfer progresses. This field indicates the number bytes remaining to be transferred. When all bytes are successfully transferred the current transfer size is zero. In LL mode, the DMA overwrites this register with the corresponding dword of the LL element. You can read this register to monitor the transfer progress, however in some scenarios this register is updated after a delay. For example, when less than 3 channels are doorbelled, this register is updated only after a descriptor finishes(linked list mode), or the transfer ends (non-linked list mode). Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W

19.4.2.367 DMA Read CH i SAR Low Register

This register holds the lower 32 bits of the Source Address Register (SAR).

DMA_SAR_LOW_OFF_RDCH_i (i=0-3) Offset address: $000730C_H+i*200_H$

DMA Read CH i SAR Low Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SRC_ADDR_REG_LOW															
rwh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SRC_ADDR_REG_LOW															
rwh															

Field	Bits	Type	Description
SRC_ADDR_REG_LOW	31:0	rwh	**Source Address Register (Lower 32 bits)** Indicates the next address to be read from. The DMA increments the SAR as the DMA transfer progresses. In LL mode, the DMA overwrites this with the corresponding dword of the LL element. • DMA Read: The SAR is the address of the remote memory • DMA Read: The SAR is the address of the local memory Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W

19.4.2.368 DMA Read CH i SAR High Register

This register holds the higher 32 bits of the Source Address Register (SAR).

DMA_SAR_HIGH_OFF_RDCH_i (i=0-3) Offset address: $0007310_H+i*200_H$

DMA Read CH i SAR High Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SRC_ADDR_REG_HIGH															
rwh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SRC_ADDR_REG_HIGH															
rwh															

Field	Bits	Type	Description
SRC_ADDR_REG_HIGH	31:0	rwh	**Source Address Register (higher 32 bits)** In LL mode, the DMA overwrites this with the corresponding dword of the LL element. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W

19.4.2.369 DMA Read CH i DAR Low Register

This register holds the lower 32 bits of the Destination Address Register (DAR).

DMA_DAR_LOW_OFF_RDCH_i (i=0-3) Offset address: $0007314_H+i*200_H$

DMA Read CH i DAR Low Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DST_ADDR_REG_LOW															
rwh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DST_ADDR_REG_LOW															
rwh															

Field	Bits	Type	Description
DST_ADDR_REG_LOW	31:0	rwh	**Destination Address Register (Lower 32 bits)** Indicates the next address to be written to. The DMA increments the DAR as the DMA transfer progresses. In LL mode, the DMA overwrites this with the corresponding dword of the LL element. • DMA Read: The DAR is the address of the local memory • DMA Read: The DAR is the address of the remote memory Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W

19.4.2.370 DMA Read CH i DAR High Register

This register holds the higher 32 bits of the Destination Address Register (DAR).

DMA_DAR_HIGH_OFF_RDCH_i (i=0-3) Offset address: $0007318_H+i*200_H$

DMA Read CH i DAR High Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DST_ADDR_REG_HIGH															
rwh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DST_ADDR_REG_HIGH															
rwh															

Field	Bits	Type	Description
DST_ADDR_REG_HIGH	31:0	rwh	**Destination Address Register (higher 32 bits)** In LL mode, the DMA overwrites this with the corresponding dword of the LL element. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W

19.4.2.371 DMA Read CH i Linked List Pointer Low Register

This register holds the lower 32 bits of the DMA read linked list pointer.

DMA_LLP_LOW_OFF_RDCH_i (i=0-3) Offset address: $000731C_H+i*200_H$

DMA Read CH i Linked List Pointer Low Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LLP_LOW															
rwh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LLP_LOW															
rwh															

Field	Bits	Type	Description
LLP_LOW	31:0	rwh	**Lower bits of the address of the linked list transfer list in local memory** Used in linked list mode only. When using DMA to point to the next element in the transfer list after the previous element is consumed. • When the current element is a data element; this field is incremented by 6 DWORDs • When the current element is a link element; this field is overwritten by the LL Element Pointer of the element. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W

19.4.2.372 DMA CH i Read Linked List Pointer High Register

This register holds the higher 32 bits of the DMA read linked list pointer.

DMA_LLP_HIGH_OFF_RDCH_i (i=0-3) Offset address: $0007320_H+i*200_H$

DMA CH i Read Linked List Pointer High Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LLP_HIGH															
rwh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LLP_HIGH															
rwh															

Field	Bits	Type	Description
LLP_HIGH	31:0	rwh	**Higher 32 bits of the address of the linked list transfer list in local memory** Used in linked list mode only. Note: The access attributes of this field are as follows: - Wire: R/W - Dbi: R/W

19.4.2.373 Interrupt clear for DMA read channel i

This register has dedicated control over a DMA channel and it has the same functionality as DMA Read Interrupt Clear Register.

DMACAP_CHi_RDCHN_RD_INT_CLEAR (i=0-3) Offset address: $0007324_H+i*200_H$

Interrupt clear for DMA read channel i Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0														RD_ABORT_INT_CLEAR	RD_DONE_INT_CLEAR
r														w	w

Field	Bits	Type	Description
RD_DONE_INT_CLEAR	0	w	**Done Interrupt Clear for the DMA channel i** You must write a 1'b1 to clear the corresponding bit in the Done interrupt status field of the DMA read interrupt status register. Note: Reading from this self-clearing register field always returns a '0'. 0_B No effect 1_B Clear status

(table continues...)

(continued)

Field	Bits	Type	Description
RD_ABORT_INT_CLEAR	1	w	**Abort Interrupt Clear for the DMA channel i** You must write a 1'b1 to clear the corresponding bit in the Abort interrupt status field of the DMA read interrupt status register.Note: Reading from this self-clearing register field always returns a '0'. 0_B No effect 1_B Clear status
0	31:2	r	**Reserved** Read as 0; should be written with 0.

19.4.2.374 Interrupt status for DMA read channel i

This register has dedicated control over a DMA channel and it has the same functionality as DMA Read Interrupt Status Register.

DMACAP_CHi_RDCHN_RD_INT_STATUS (i=0-3) Offset address: $0007328_H+i*200_H$

Interrupt status for DMA read channel i Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0														RD_ABORT_INT_STATUS	RD_DONE_INT_STATUS
r														rh	rh

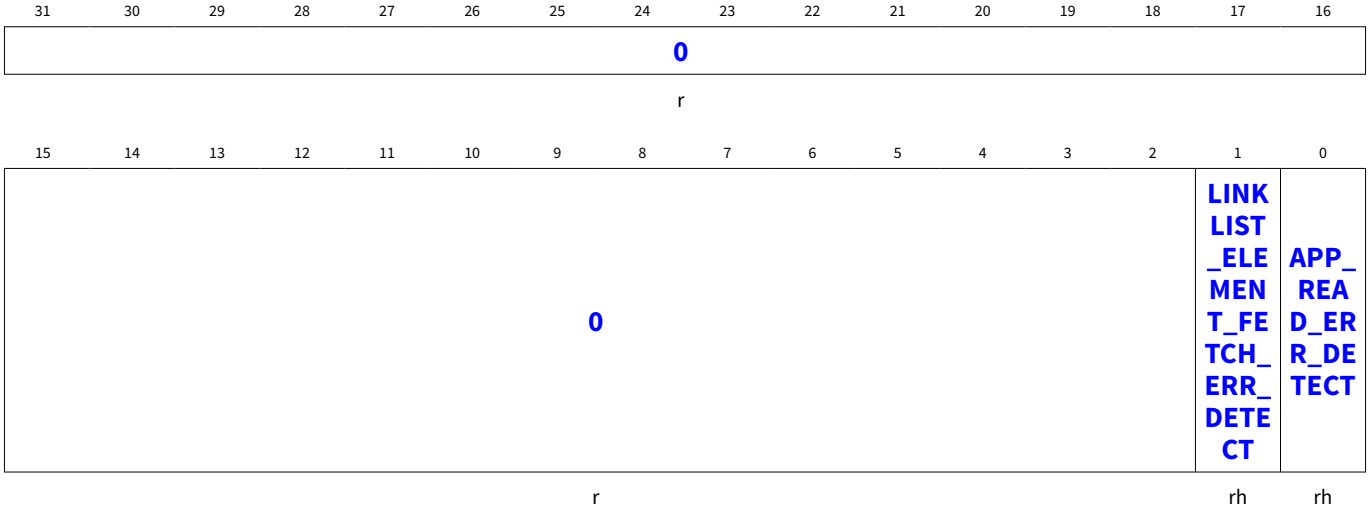
Field	Bits	Type	Description
RD_DONE_INT_STATUS	0	rh	**Done Interrupt Status for the DMA channel i** The DMA read channel has successfully completed the DMA transfer.
RD_ABORT_INT_STATUS	1	rh	**Abort Interrupt Status for the DMA channel i** The DMA read channel has detected an error, or you manually stopped the transfer.
0	31:2	r	**Reserved** Read as 0; should be written with 0.

19.4.2.375 Error status for DMA read channel i

This register has dedicated control over a DMA channel and it has the same functionality as DMA Read Error Status Low Register.

DMACAP_CHi_RDCHN_RD_ERR_STATUS (i=0-3) Offset address: $000732C_H+i*200_H$

Error status for DMA read channel i Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
APP_READ_ERR_DETECT	0	rh	**Application Read Error Detected for the DMA channel i** The DMA read channel has received an error response from the AXI bus while reading data from it. . - Clearing: You must write a 1'b1 to the RD_ABORT_INT_CLEAR to clear this error bit.
LINKLIST_ELEMENT_FETCH_ERR_DETECT	1	rh	**Linked List Element Fetch Error Detected for the DMA channel i** The DMA read channel has received an error response from the AXI bus while reading a linked list element from local memory.- Clearing: You must write a 1'b1 to the corresponding channel bit in the RD_ABORT_INT_CLEAR to clear this error bit.
0	31:2	r	**Reserved** Read as 0; should be written with 0.

19.4.2.376 Higher error status for DMA read channel i

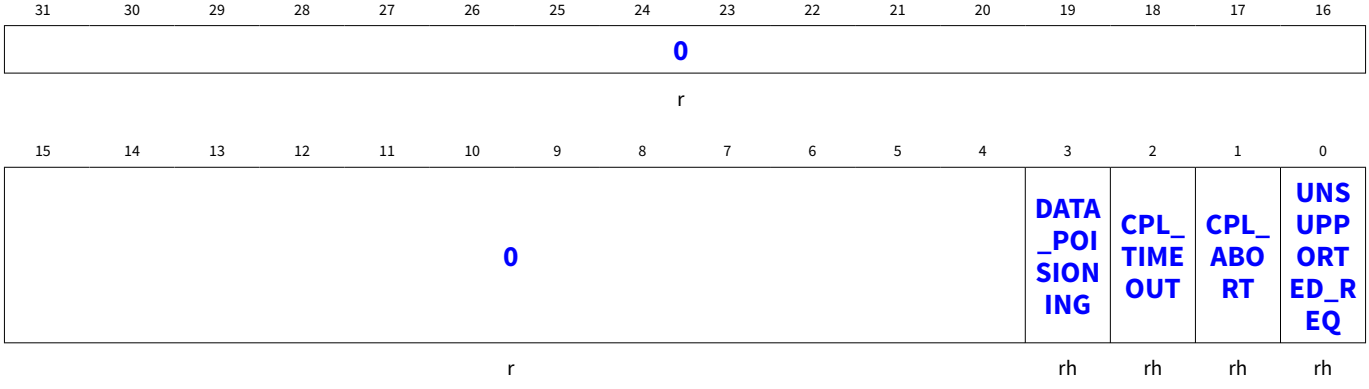
This register has dedicated control over a DMA channel and it has the same functionality as DMA Read Error Status High Register.

DMACAP_CHi_RDCHN_RD_ERR_STATUS_HIGH (i=0-3)

Offset address: $0007330_H+i*200_H$

Higher error status for DMA read channel i

Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
UNSUPPORTED_REQ	0	rh	**Unsupported Request for the DMA channel i** The DMA read channel has received a PCIe unsupported request completion status from the remote device in response to the MRd request. . - Clearing: You must write a 1'b1 to RD_ABORT_INT_CLEAR to clear this error bit. Note, this also clears the other error bits for the same channel in this register and in the DMA Read Error Status Low register.
CPL_ABORT	1	rh	**Completer Abort for the DMA channel i** The DMA read channel has received a PCIe completer abort completion status from the remote device in response to the MRd request.- Clearing: You must write a 1'b1 to the corresponding channel bit in the RD_ABORT_INT_CLEAR to clear this error bit. Note, this also clears the other error bits for the same channel in this register and in the DMA Read Error Status Low register.
CPL_TIMEOUT	2	rh	**Completion Time Out for the DMA channel i** The DMA read channel has timed-out while waiting for the remote device to respond to the MRd request, or a malformed CplD has been received. - Clearing: You must write a 1'b1 to the RD_ABORT_INT_CLEAR to clear this error bit. Note, this also clears the other error bits for the same channel in this register and in the DMA Read Error Status Low register.
DATA_POISIONING	3	rh	**Data Poisoning for the DMA channel i** The DMA read channel has detected data poisoning in the completion from the remote device (in response to the MRd request). The DMA read channel will drop the completion and then be halted. - Clearing: You must write a 1'b1 to the RD_ABORT_INT_CLEAR to clear this error bit. Note, this also clears the other error bits for the same channel in this register and in the DMA Read Error Status Low register.
0	31:4	r	**Reserved** Read as 0; should be written with 0.

19.4.2.377 Doorbell for DMA read channel i

This register has dedicated control over a DMA channel and it has the same functionality as DMA Read Doorbell Register.

Note: This register is strictly a write only register. Reading this register may result in undefined behavior.

DMACAP_CHi_RDCHN_RD_DOORBELL (i=0-3) Offset address: $0007334_H+i*200_H$

Doorbell for DMA read channel i Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0														RD_STOP	RD_DOORBELL
r														w	w

Field	Bits	Type	Description
RD_DOORBELL	0	w	**Doorbell Number for the DMA channel i** You must write the channel number 1'b1 to this register to start the DMA write transfer for that channel. Note: A write to this field triggers the controller to exit L1 substates 0_B No effect 1_B Start DMA transfer
RD_STOP	1	w	**Stop for the DMA channel i** The DMA read channel stops issuing requests, sets the channel status to "Stopped", and asserts the "Abort" interrupt if it is enabled. Before setting the Stop bit, you must read the channel Status field (CS) of the DMA Write Channel Control 1 Register (DMA_CH_CONTROL1_OFF_WRCH_0) to ensure that the write channel is "Running" (transferring data). Note: Use this feature with caution. DMA stop feature is not meant to be used in operational mode. It is only a debug feature and has associated hazards. 0_B No effect 1_B Stop the DMA
0	31:2	r	**Reserved** Read as 0; should be written with 0.

19.4.2.378 Write access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

S2A_ACCENy_WRA (y=0-7) Offset address: $3FFFE00_H+y*20_H$

Write access enable register A Application Reset value: $1000\ 0003_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access

19.4.2.379 Write access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

S2A_ACCENy_WRB (y=0-7) Offset address: $3FFFE04_H+y*20_H$

Write access enable register B Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN63	EN62	EN61	EN60	EN59	EN58	EN57	EN56	EN55	EN54	EN53	EN52	EN51	EN50	EN49	EN48
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN47	EN46	EN45	EN44	EN43	EN42	EN41	EN40	EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-63)	q-32	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access

19.4.2.380 Read access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

S2A_ACCENy_RDA (y=0-7) Offset address: $3FFFE08_H+y*20_H$

Read access enable register A Application Reset value: $FFFF\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access

19.4.2.381 Read access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

S2A_ACCENy_RDB (y=0-7) Offset address: $3FFFE0C_H+y*20_H$

Read access enable register B Application Reset value: $FFFF\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN63	EN62	EN61	EN60	EN59	EN58	EN57	EN56	EN55	EN54	EN53	EN52	EN51	EN50	EN49	EN48
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN47	EN46	EN45	EN44	EN43	EN42	EN41	EN40	EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-63)	q-32	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access

19.4.2.382 VM access enable register

This register defines which virtual machine encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each VMx.

S2A_ACCENy_VM (y=0-7) Offset address: $3FFFE10_H+y*20_H$

VM access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for VM ID q** This bit enables read access to the access protected region for transactions with the VM ID q. 0_B Disabled for read access 1_B Enabled for read access

(table continues...)

(continued)

Field	Bits	Type	Description
WRq (q=00-07)	q+16	rw	**Write access enable for VM ID q** This bit enables write access to the access protected region for transactions with the VM ID q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

19.4.2.383 PRS access enable register

This register defines which protection register set encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each PRSx.

S2A_ACCENy_PRS (y=0-7) Offset address: $3FFFE14_H+y*20_H$

PRS access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

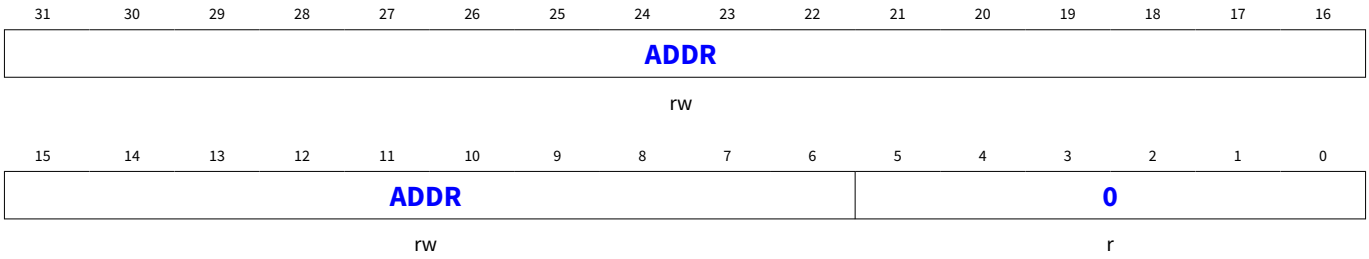
Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for PRS q** This bit enables read access to the access protected region for transactions with the PRS q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for PRS q** This bit enables write access to the access protected region for transactions with the PRS q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

19.4.2.384 Region lower address register

This register sets the lower bound of the access protected region. Addresses greater than or equal to the lower bound are considered part of the access protected region.

S2A_ACCENy_RGNLA (y=0-7) Offset address: $3FFFE18_H+y*20_H$

Region lower address register Application Reset value: $0000\ 0000_H$



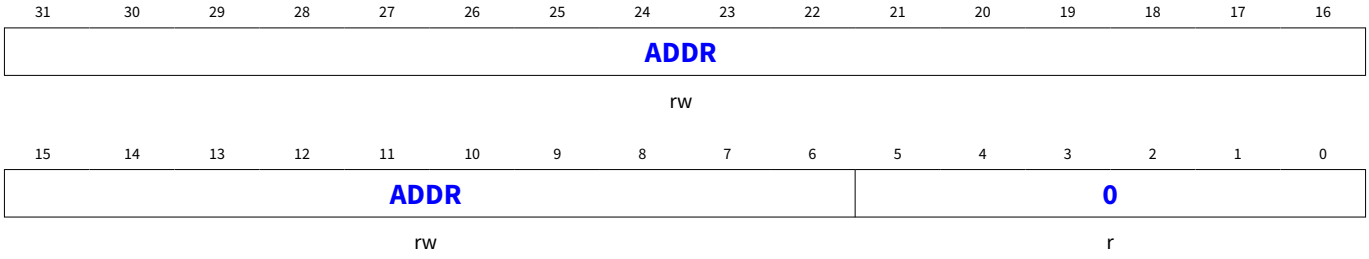
Field	Bits	Type	Description
ADDR	31:6	rw	**Bits 31:6 of the lower bound of the access protected region**
0	5:0	r	**Reserved** Read as 0; should be written with 0.

19.4.2.385 Region upper address register

This register sets the upper bound of the access protected region. Addresses stricly less than the upper bound are considered part of the access protected region.

S2A_ACCENy_RGNUA (y=0-7) Offset address: $3FFFE1C_H+y*20_H$

Region upper address register Application Reset value: $FFFF\ FFC0_H$



Field	Bits	Type	Description
ADDR	31:6	rw	**Bits 31:6 of the upper bound of the access protected region**
0	5:0	r	**Reserved** Read as 0; should be written with 0.

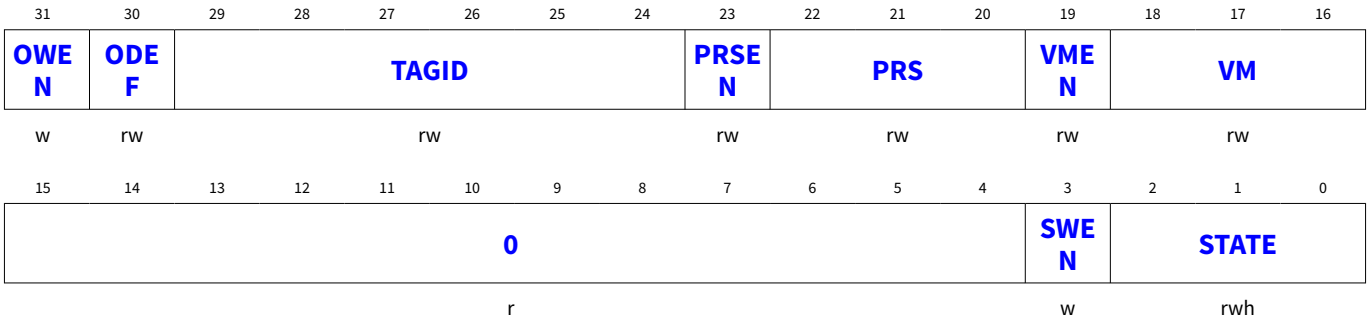
19.4.2.386 PROT Register Safe Endinit

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner and in some PROT states, the secure master to update the PROT state as described in the PROT mechanism.

The PROTSE register controls lock / unlock of the local Safe Endinit (SE) protected control registers.

S2A_PROTSE Offset address: $3FFFF04_H$

PROT Register Safe Endinit Application Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
VM	18:16	rw	**Virtual machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual machine definition enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.
PRSEN	23	rw	**Protection set definition enable for PROT owner** 0_B **Disable**: PRS is not part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG-ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner

(table continues...)

(continued)

Field	Bits	Type	Description
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; should be written with 0.

Table 895 Access types of S2A_PROTSE in order of evaluation

Condition	Type		Description
write 1 to .**SWEN**	rwh	STATE	
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
otherwise	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

19.5 Debug information

OTGB0[15:0] and OTGB1[15:0] can be used for tracing a lot of debug information for PCIe. Trigger sets are created to define which signals are traced. Different colors stand for different trigger sets. For the current plan, PCIe has three trigger sets. The third trigger set TS3 is tied to 0. Two trigger sets can be traced to OTGB0 and OTGB1 in parallel, as shown below .

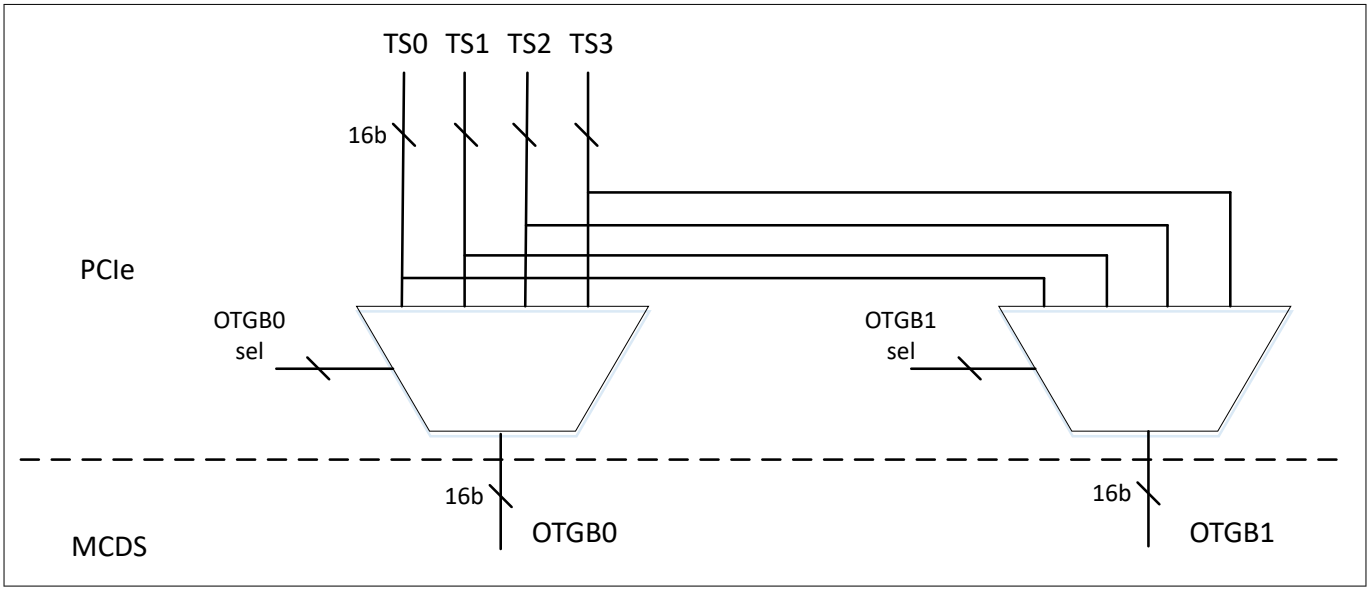


Figure 324 OTGB trigger sets selection logic

Table 896 TS16_PCIe0: Link information

Bits	Description
[0:5]	smlh_ltssm_state: LTSSM current state. Encoding is same as the dedicated smlh_ltssm_state
[6]	rmlh_rcvd_idle[0]: Receiver is receiving logical idle
[7]	smlh_ts2_rcvd: TS2 training sequence received (pulse)
[8]	smlh_ts1_rcvd: TS1 training sequence received (pulse)
[9]	smlh_inskip_rcv: Receiver reports skip reception
[10]	smlh_link_up: LTSSM reports PHY link up or LTSSM is in Loopback.Active for Loopback Master
[11]	xmtbyte_skip_sent: A skip ordered set has been transmitted
[12]	app_init_rst: Application request to initiate training reset
[13]	smlh_ltssm_in_pollconfig: LTSSM is in Polling.Configuration state
[14]	smlh_link_in_training: LTSSM performing link training
[15]	smlh_link_disable: TSSM in DISABLE state. Link inoperable

Table 897 TS16_PCIe1: LTSSM stats

Bits of cxpl_debug_info_ei	Description
[0]	EIOS detected
[1]	L1
[2]	L2

(table continues...)

Table 897 (continued) TS16_PCIe1: LTSSM stats

Bits of cxpl_debug_info_ei	Description
[3]	RxL0s
[4]	Disabled
[5]	Detect.Quiet
[6]	Polling.Active
[7]	Polling.Compliance
[8]	LTSSM is in a transitory state prior to L1 or L2
[9]	LTSSM is in a transitory state prior to Disabled
[10]	LTSSM is in Loopback.Active as a Slave at Gen1
[11]	LTSSM is in Polling.Active
[12]	LTSSM enters Recovery from L0 with EI inferred
[13]	LTSSM enters Recovery.Speed from Recovery.RcvrCfg with EI inferred
[14]	EI inferred while LTSSM in Recovery.Speed
[15]	EI inferred while LTSSM in Loopback.Active as a slave

Table 898 TS16_PCIe2: Power management and controller status

Bits	Description
[0:2]	pm_l1sub_state: PM L1 sub states
[3:5]	pm_curnt_state: PM current link states
[6:8]	pm_dstate: PM current device states
[9]	radm_q_not_empty: level indicating that the receive queues containTLP header and data.
[10]	rtlh_rfc_upd: indicates that the controller received a flowcontrol update DLLP.
[11]	radm_xfer_pending: receive request pending status.
[12]	edma_xfer_pending: DMA transfer pending status.
[13]	brdg_slv_xfer_pending: AXI Slave non-DBI transfer pending status.
[14]	rdlh_link_up: data link layer up or down indicator.
[15]	Reserved

19.5.1 Error injection

The controller provides support for your application to inject the following types of errors:

- CRC Error
- Sequence Number Error
- DLLP Error
- Symbol DataK Mask Error or Sync Header Error
- FC Credit Update Error
- TLP Duplicate/Nullify Error
- Specific TLP Error

Your application can use the registers to set up error injection.

Table 899 Error injection vendor specific extended capability (VSEC) registers

Register	Description
RAS_DES_CAP_HEADER_REG	Vendor-Specific Extended Capability Header.
VENDOR_SPECIFIC_HEADER_REG	Vendor-Specific Header.
.....other registers (for example, Event Counter, Time-based Analysis, Error Injection)	
EINJ_ENABLE_REG	Error Injection Enable.
EINJ0_CRC_REG	Error Injection Control 0 (CRC Error)
EINJ1_SEQNUM_REG	Error Injection Control 1 (SeqNum Error)
EINJ2_DLLP_REG	Error Injection Control 2 (DLLP Error)
EINJ3_SYMBOL_REG	Error Injection Control 3 (Symbol Error)
EINJ4_FC_REG	Error Injection Control 4 (FC Credit Error)
EINJ5_SP_TLP_REG	Error Injection Control 5 (Specific TLP Error)

19.5.2 Statistics

To facilitate collection of statistics the controller provides the following features:

- Time Based Analysis (RX/TX data throughput and time spent in each low-power LTSSM state)
- Event counters (Error and Non-Error)

Time based analysis

Using this feature you can obtain information regarding RX/TX data throughput and time spent in each low-power LTSSM state by the controller. Your application can use the outputs in Table 900 and registers in Table 902.

Table 900 Time-based analysis signals

Signal name	
app_ras_des_tba_ctrl	This bus controls the start/end of time based analysis
cdm_ras_des_tba_info_common	This is common event signal bus that provides RAS D.E.S. time based analysis information.

Table 901 Time-based analysis RAM signals

Signal name	
cdm_ras_des_tb_ram_*	This signal bus provides RAS D.E.S. time based analysis information for the RAMs.

Table 902 Time-based analysis vendor specific extended capability (VSEC) registers

Register	Description
RAS_DES_CAP_HEADER_REG	Vendor-Specific Extended Capability Header
VENDOR_SPECIFIC_HEADER_REG	Vendor-Specific Header.
.....other registers (for example, Event Counter).....	
TIME_BASED_ANALYSIS_CONTROL_REG	Time-based Analysis Control.
TIME_BASED_ANALYSIS_DATA_REG	Time-based Analysis Data.

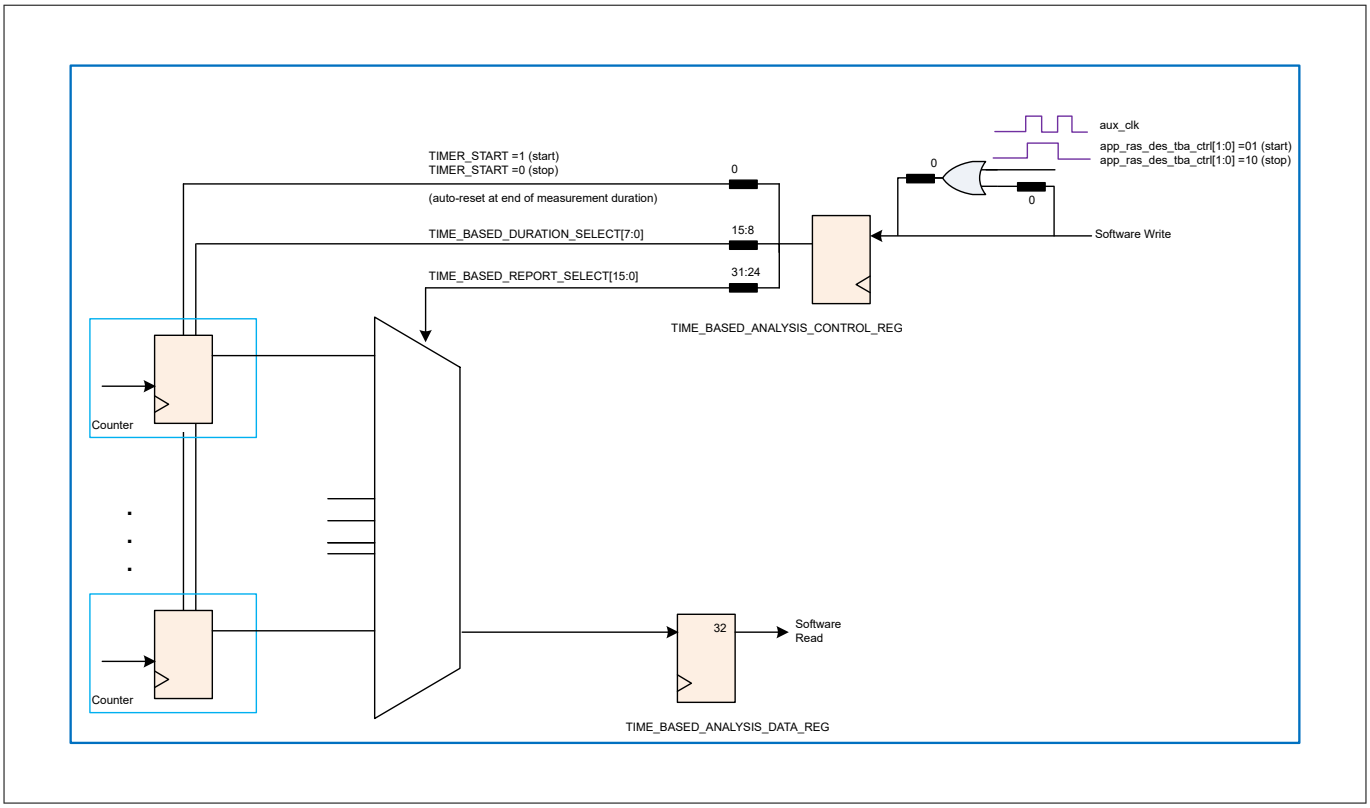


Figure 325 Time-based analysis register access

The counters measure, for example, what percentage of time does the controller stay in L0 in a one second window (configurable through TIME_BASED_DURATION_SELECT field). The measurement range of each Event in Group#0 is shown in Figure 326.

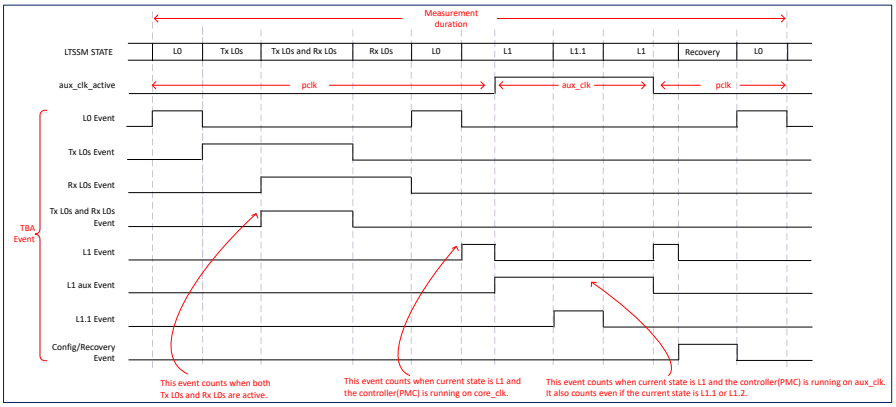


Figure 326 Measurement ranges of time-based analysis events in group #0

Table 903 Time-based analysis counter group #0 (32-bit low-power cycle counter (RAM)

Event #	% of measurement window spent in LTSSM state	Note
0x01	Tx L0s	-
0x02	Rx L0s	-
0x03	L0	-

(table continues...)

Table 903 (continued) Time-based analysis counter group #0 (32-bit low-power cycle counter (RAM)

Event #	% of measurement window spent in LTSSM state	Note
0x04	L1	The controller (PMC) is running on core_clk
0x05	L1.1	-
0x06	L1.2	-
0x07	Configuration/Recovery	-
0x08	L1 aux	The controller (PMC) is running on aux_clk
0x09	Tx L0s and Rx L0s	-

Table 904 Time-based analysis counter group #1 (32-bit throughput 4-DWORD counter)

Event #	Amount of data processed (units of 16 Bytes)	Note
0x00	Tx PCIe TLP Data Payload	-
0x01	Rx PCIe TLP Data Payload	-

L2 Exit: All RASDES analysis results are cleared when the controller exits from L2.

Event counters

Your application can use the outputs in Table 905 and registers in Table 907.

Table 905 Event counter signals

Output	Description
cdm_ras_des_ec_info_common	This is a common signal bus that provides RAS D.E.S. event counter information.
cdm_ras_des_ec_info_i(for i = 0; i <CX_NL))	This is Lane*i* event signal bus that provides RAS D.E.S. event counter information.

Table 906 Event counter RAM signals

Signal names	Description
cdm_ras_des_ec_ram_*	This signal bus provides RAS D.E.S. Event Counter information for the RAMs.

Table 907 Event counter VSEC registers

Register	Description
RAS_DES_CAP_HEADER_REG	Vendor-Specific Extended Capability Header.
VENDOR_SPECIFIC_HEADER_REG	Vendor-Specific Header.
EVENT_COUNTER_CONTROL_REG	Event Counter Control.
EVENT_COUNTER_DATA_REG	Event Counter Data.

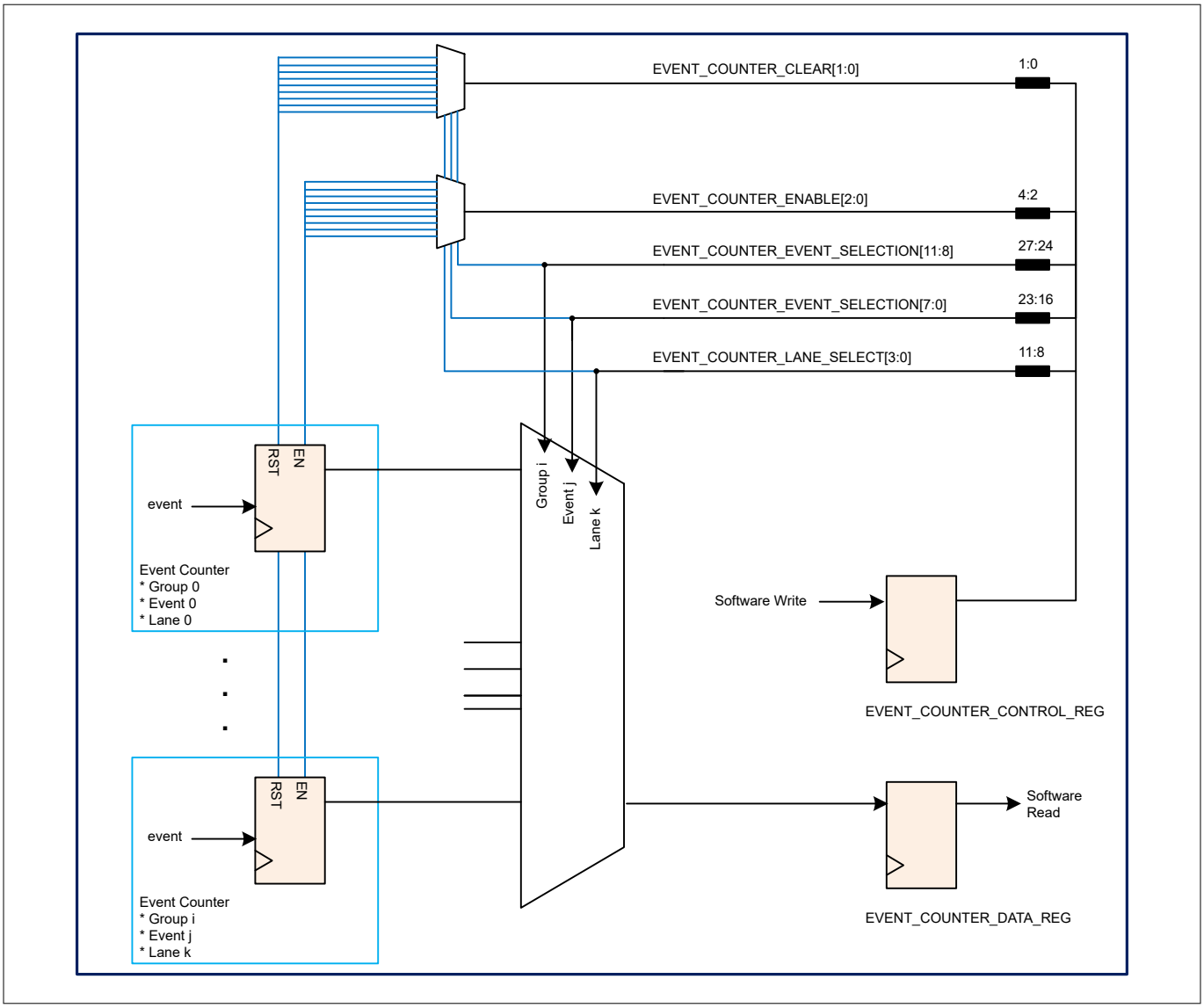


Figure 327 Event counter register access

Event counter data details

The following tables provide the Group/Event numbers which are required when using the EVENT_COUNTER_CONTROL_REG viewport control register to read each of the event counters. Some of the event counters only exist for specific configurations. Groups 5-7 are implemented in RAM as indicated. The group # is set by EVENT_COUNTER_EVENT_SELECTION[11:8]and the event # is set by EVENT_COUNTER_EVENT_SELECTION[7:0]

Table 908 Event counter group #0 (4-bit layer1 error counter per-lane)

Event #	Description	Note
0x00	EBUF Overflow	-
0x01	EBUF Under-run	-
0x02	Decode Error	-
0x03	Running Disparity Error	-
0x04	SKP OS Parity Error	Gen3 Only
0x05	SYNC Header Error	Gen3 or greater

(table continues...)

Table 908 (continued) Event counter group #0 (4-bit layer1 error counter per-lane)

Event #	Description	Note
0x06	Rx Valid de-assertion	RxValid de-assertion without EIOSs
0x07	CTL SKP OS Parity Error	Gen4
0x08	1st Retimer Parity Error	Gen4
0x09	2nd Retimer Parity Error	Gen4
0x0A	Margin CRC and Parity Error	Gen4

Table 909 Event counter group #1 (8-bit layer1 error counter common-lane)

Event #	Description	Note
0x00	Reserved	-
0x01	Reserved	-
0x02	Reserved	-
0x03	Reserved	-
0x04	Reserved	-
0x05	Detect EI Infer	-
0x06	Receiver Error	-
0x07	Rx Recovery Request	When the controller receives TS1 OS in L0s state.
0x08	N_FTS Timeout	Timeout when the controller's Rx condition is in Rx_L0s.FTS.
0x09	Framing Error	Gen3
0x0A	Deskew Error	-

Table 910 Event counter group #2 (8-bit layer2 error counter common-lane)

Event #	Description	Note
0x00	BAD TLP	-
0x01	LCRC Error	-
0x02	BAD DLLP	-
0x03	Replay Number Rollover	-
0x04	Replay Timeout	-
0x05	Rx Nak DLLP	-
0x06	Tx Nak DLLP	-
0x07	Retry TLP	-

Table 911 Event counter group #3 (8-bit layer3 error counter common-lane)

Event[1)] #	Description	Note
0x00	FC Timeout	-
0x01	Poisoned TLP	Any function
0x02	ECRC Error	Any function
0x03	Unsupported Request	Any function

(table continues...)

Table 911 (continued) Event counter group #3 (8-bit layer3 error counter common-lane)

Event[1] #	Description	Note
0x04	Completer Abort	Any function
0x05	Completion Timeout	Any function

1) Malformed TLP, FC & DL Protocol errors are excluded because of low frequency.

Table 912 Event counter group #4 (4-bit layer1 non-error counter per-lane)

Event #	Description	Note
0x00	EBUF SKP Add	-
0x01	EBUF SKP Del	-

Table 913 Event counter group #5 (32-bit layer1 non-error counter [RAM] common lane)

Event #	Description	Note
0x00	L0 to Recovery Entry	-
0x01	L1 to Recovery Entry	-
0x02	Tx L0s Entry	-
0x03	Rx L0s Entry	-
0x04	ASPM L1 reject	RC: Send PM_Active_State_Nak EP: Received PM_Active_State_Nak
0x05	L1 Entry	-
0x06	L1 CPM	CX_L1_SUBSTATES_ENABLE
0x07	L1.1 Entry	CX_L1_SUBSTATES_ENABLE
0x08	L1.2 Entry	CX_L1_SUBSTATES_ENABLE
0x09	L1 short duration	(CX_L1_SUBSTATES_ENABLE) No. of times link entered L1.0 state but exited to L0 without entering L1.1 or L1.2 states
0x0A	L1.2 abort	(CX_L1_SUBSTATES_ENABLE) No. of times link entered L1.2_Entry state but is forced to exit to L1.0 then L0 because it saw clkreq# asserted while in this state
0x0B	L2 Entry	-
0x0C	Speed Change	Gen2
0x0D	Link width Change	x2
0x0E	Reserved	-

Table 914 Event counter group #6 (32-bit layer2 non-error counter [RAM])

Event #	Description	Note
0x00	Tx Ack DLLP	-
0x01	Tx Update FC DLLP	-
0x02	Rx Ack DLLP	-

(table continues...)

Table 914 (continued) Event counter group #6 (32-bit layer2 non-error counter [RAM])

Event #	Description	Note
0x03	Rx Update FC DLLP	-
0x04	Rx Nullified TLP	-
0x05	Tx Nullified TLP	-
0x06	Rx Duplicate TLP	-

Table 915 Event counter group #7 (32-bit layer3 non-error counter [RAM])

Event #	Description	Note
0x00	Tx Memory Write	-
0x01	Tx Memory Read	-
0x02	Tx Configuration Write	-
0x03	Tx Configuration Read	-
0x04	Tx IO Write	-
0x05	Tx IO Read	-
0x06	Tx Completion without data	-
0x07	Tx Completion w data	-
0x08	Tx Message TLP	PCIe VC only
0x09	Tx Atomic	-
0x0A	Tx TLP with Prefix	-
0x0B	Rx Memory Write	-
0x0C	Rx Memory Read	-
0x0D	Rx Configuration Write	-
0x0E	Rx Configuration Read	-
0x0F	Rx IO Write	-
0x10	Rx IO Read	-
0x11	Rx Completion without data	-
0x12	Rx Completion w data	-
0x13	Rx Message TLP	PCIe VC only
0x14	Rx Atomic	-
0x15	Rx TLP with Prefix	-

Limitations

If the controller (PMC) is running on `aux_clk` during the measurement:

- The measurement timers (`TIME_BASED_DURATION_SELECT`: 1 ms, 10 ms, …, 2 s, 4 s) have measurement error less than 4 µs
- The measurement timers and event timers of all the events are stopped temporarily when `aux_clk_active` or `pm_sel_aux_clk` are controlled. Figure 328 describes the timing details when `aux_clk_active` or `pm_sel_aux_clk` are controlled

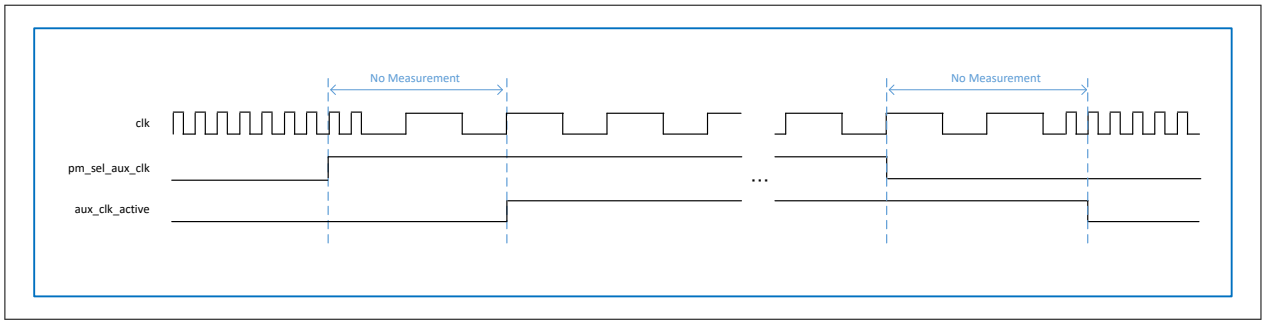


Figure 328 No measurement period

19.6 References

This section is not applicable for the module being described.

19.7 PCIe revision history

Reference	Description of change(s)
Date range: 2024-08-17 to 2025-01-31	
Minimum working example of RC Minimum working example of EP	Common reference clock PHY configuration added to minimum initialization for RC and EP
Register section	Added UPCS registers to control PHY configuration Unsupported Expansion ROM BAR description removed References to AXI_ERROR_RESP corrected to AMBA_ERROR_RESP PERST logic states clarified
Local interrupts	Improving local and global interrupt descriptions

19.8 TC4Dx PCIe information

19.8.1 TC4Dx PCIe configuration

There are no deviations from the generic specification.

Table 916 feature support in TC4D

Parameter	Number of instances	Comment
PCIe in SoC	2	

19.8.2 TC4Dx PCIe features

There are no deviations from the generic specification.

19.8.3 TC4Dx PCIe functional description

In TC4Dx device, the PCIE module is connected to ComPB bus interface.

Local Bus accesses

***Note**: Be aware that the upper region addresses of the SRI APU instances 0-7 defined inside ACCEN_RGNUA is evaluated as "less than or equal to" instead of "strictly less than". Therefore, the upper addresses of ATU and APU cannot be exactly aligned.*

19.8.3.1 TC4Dx specific UPCS override settings for common refclk PHY settings

There is one select field, UPCS_OVRD17, bit 11, to override all of the PHY settings. PHY overrides should be set according to the table below.

Table 917 UPCS common refclk override settings

Register	Value
UPCS_OVRD1	$2804011D_H$
UPCS_OVRD2	$000115C5_H$
UPCS_OVRD3	00000000_H
UPCS_OVRD4	01341914_H
UPCS_OVRD5	$029330C1_H$
UPCS_OVRD6	$0D6C8C08_H$
UPCS_OVRD7	$0019020D_H$
UPCS_OVRD8	$000030C6_H$
UPCS_OVRD9	$35430A00_H$
UPCS_OVRD10	$0A2A1654_H$
UPCS_OVRD11	$019A2842_H$
UPCS_OVRD12	$007D03E8_H$

(table continues...)

Table 917 (continued) UPCS common refclk override settings

Register	Value
UPCS_OVRD13	$006223E8_H$
UPCS_OVRD14	00000303_H
UPCS_OVRD15	00000000_H
UPCS_OVRD16	$0000F000_H$
UPCS_OVRD17	$00040C00_H$

***Note**: To apply UPCS overrides, bit 11 in register OVRD17 must be set to 1_B*

Related information

UPCS on page 2802

19.8.4 TC4Dx PCIe registers

There are no deviations from the generic specification.

19.8.4.1 Register address space - PCIE_USP domain FPI

Table 918 Registers address space - PCIE_USP domain FPI

Module	Domain	Base address	End address	Note
PCIE0_USP	FPI	$F46A8000_H$	$F46AFFFF_H$	FPI Interface
PCIE1_USP	FPI	$F46A0000_H$	$F46A7FFF_H$	FPI Interface

19.8.4.2 Register address space - PCIE_USP domain SRI

Table 919 Registers address space - PCIE_USP domain SRI

Module	Domain	Base address	End address	Note
PCIE0_USP_SRI	SRI	$FC000000_H$	$FFFFFFFF_H$	SRI interface
PCIE1_USP_SRI	SRI	$EC000000_H$	$EFFFFFFF_H$	SRI interface

19.8.4.3 Register overview - access mode glossary

Table 920 Register overview - access mode glossary

Keyword	Description
E	Access protection using PROT register PCIE0_USP_PROTE or PCIE1_USP_PROTE .
SE	Access protection using PROT register PCIE0_USP_PROTSE or PCIE1_USP_PROTSE or PCIE0_USP_S2A_PROTSE or PCIE1_USP_S2A_PROTSE .
APU-P	Protection group consisting of registers PCIE0_USP_ACCEN_WRA , PCIE0_USP_ACCEN_WRB , PCIE0_USP_ACCEN_RDA , PCIE0_USP_ACCEN_RDB , PCIE0_USP_ACCEN_VM , PCIE0_USP_ACCEN_PRS or PCIE1_USP_ACCEN_WRA , PCIE1_USP_ACCEN_WRB , PCIE1_USP_ACCEN_RDA , PCIE1_USP_ACCEN_RDB , PCIE1_USP_ACCEN_VM , PCIE1_USP_ACCEN_PRS .
P	Access protection using *APU-P* registers.
APU-Py (y=0-7)	Protection group consisting of registers PCIE0_USP_S2A_ACCENy_WRA , PCIE0_USP_S2A_ACCENy_WRB , PCIE0_USP_S2A_ACCENy_RDA , PCIE0_USP_S2A_ACCENy_RDB , PCIE0_USP_S2A_ACCENy_VM , PCIE0_USP_S2A_ACCENy_PRS , PCIE0_USP_S2A_ACCENy_RGNLA , PCIE0_USP_S2A_ACCENy_RGNUA or PCIE1_USP_S2A_ACCENy_WRA , PCIE1_USP_S2A_ACCENy_WRB , PCIE1_USP_S2A_ACCENy_RDA , PCIE1_USP_S2A_ACCENy_RDB , PCIE1_USP_S2A_ACCENy_VM , PCIE1_USP_S2A_ACCENy_PRS , PCIE1_USP_S2A_ACCENy_RGNLA , PCIE1_USP_S2A_ACCENy_RGNUA .
Py	Access protection using *APU-Py* registers.

(table continues...)

Table 920 (continued) Register overview - access mode glossary

Keyword	Description
APU-P_DMAy (y=0-3)	Protection group consisting of registers PCIE0_USP_DMA_ACCENy_WRA , PCIE0_USP_DMA_ACCENy_WRB , PCIE0_USP_DMA_ACCENy_RDA , PCIE0_USP_DMA_ACCENy_RDB , PCIE0_USP_DMA_ACCENy_VM , PCIE0_USP_DMA_ACCENy_PRS or PCIE1_USP_DMA_ACCENy_WRA , PCIE1_USP_DMA_ACCENy_WRB , PCIE1_USP_DMA_ACCENy_RDA , PCIE1_USP_DMA_ACCENy_RDB , PCIE1_USP_DMA_ACCENy_VM , PCIE1_USP_DMA_ACCENy_PRS .
P_DMAy	Access protection using *APU-P_DMAy* registers.
BE	Always returns a Bus Error.
U	No access restrictions.
PROT	Access restrictions as defined in the PROT register access rules.

19.8.4.4 Register overview - PCIE0_USP domain FPI (ascending offset address)

Table 921 Register overview - PCIE0_USP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE0_USP_CLC	Clock control register	0000000H	P	P, SV, E	Application Reset	2822
PCIE0_USP_OCS	OCDS control and status register	0000004H	P	SV, P	Debug Reset	2823
PCIE0_USP_ID	Module identification register	0000008H	P	BE	PowerOn Reset	2823
PCIE0_USP_RST_CTRLA	Reset control register A	000000CH	P	P, SV, E	Application Reset	2824
PCIE0_USP_RST_CTRLB	Reset control register B	0000010H	P	P, SV, E	Application Reset	2825
PCIE0_USP_RST_STAT	Reset status register	0000014H	P	BE	Application Reset	2825
PCIE0_USP_PROTE	PROT register endinit	0000020H	U	SV, PROT	Application Reset	2826
PCIE0_USP_PROTSE	PROT register safe endinit	0000024H	U	SV, PROT	Application Reset	2828
PCIE0_USP_ACCEN_WRA	Write access enable register A	0000040H	U	SE, SV	Application Reset	2829
PCIE0_USP_ACCEN_WRB	Write access enable register B	0000044H	U	SE, SV	Application Reset	2830
PCIE0_USP_ACCEN_RDA	Read access enable register A	0000048H	U	SE, SV	Application Reset	2830
PCIE0_USP_ACCEN_RDB	Read access enable register B	000004CH	U	SE, SV	Application Reset	2831

(table continues...)

Table 921 (continued) Register overview - PCIE0_USP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE0_USP_ACCEN_VM	VM access enable register	0000050H	U	SE, SV	Application Reset	2831
PCIE0_USP_ACCEN_PRS	PRS access enable register	0000054H	U	SE, SV	Application Reset	2832
PCIE0_USP_DMA_ACCENy_WRA (y=0-3)	CHy Write access enable register A	0000060H+y*20H	U	SE, SV	Application Reset	2833
PCIE0_USP_DMA_ACCENy_WRB (y=0-3)	CHy Write access enable register B	0000064H+y*20H	U	SE, SV	Application Reset	2833
PCIE0_USP_DMA_ACCENy_RDA (y=0-3)	CHy Read access enable register A	0000068H+y*20H	U	SE, SV	Application Reset	2834
PCIE0_USP_DMA_ACCENy_RDB (y=0-3)	CHy Read access enable register B	000006CH+y*20H	U	SE, SV	Application Reset	2834
PCIE0_USP_DMA_ACCENy_VM (y=0-3)	CHy VM access enable register	0000070H+y*20H	U	SE, SV	Application Reset	2835
PCIE0_USP_DMA_ACCENy_PRS (y=0-3)	CHy PRS access enable register	0000074H+y*20H	U	SE, SV	Application Reset	2835
PCIE0_USP_UPCS_OVRD1	UPCS layer override register 1	0000300H	P	P	Kernel Reset	2836
PCIE0_USP_UPCS_OVRD2	UPCS layer override register 2	0000304H	P	P	Kernel Reset	2836
PCIE0_USP_UPCS_OVRD3	UPCS layer override register 3	0000308H	P	P	Kernel Reset	2837
PCIE0_USP_UPCS_OVRD4	UPCS layer override register 4	000030CH	P	P	Kernel Reset	2837
PCIE0_USP_UPCS_OVRD5	UPCS layer override register 5	0000310H	P	P	Kernel Reset	2838
PCIE0_USP_UPCS_OVRD6	UPCS layer override register 6	0000314H	P	P	Kernel Reset	2838
PCIE0_USP_UPCS_OVRD7	UPCS layer override register 7	0000318H	P	P	Kernel Reset	2839
PCIE0_USP_UPCS_OVRD8	UPCS layer override register 8	000031CH	P	P	Kernel Reset	2839
PCIE0_USP_UPCS_OVRD9	UPCS layer override register 9	0000320H	P	P	Kernel Reset	2840
PCIE0_USP_UPCS_OVRD10	UPCS layer override register 10	0000324H	P	P	Kernel Reset	2840

(table continues...)

Table 921 (continued) Register overview - PCIE0_USP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE0_USP_UPCS_OVRD11	UPCS layer override register 11	0000328H	P	P	Kernel Reset	2841
PCIE0_USP_UPCS_OVRD12	UPCS layer override register 12	000032CH	P	P	Kernel Reset	2841
PCIE0_USP_UPCS_OVRD13	UPCS layer override register 13	0000330H	P	P	Kernel Reset	2841
PCIE0_USP_UPCS_OVRD14	UPCS layer override register 14	0000334H	P	P	Kernel Reset	2842
PCIE0_USP_UPCS_OVRD15	UPCS layer override register 15	0000338H	P	P	Kernel Reset	2842
PCIE0_USP_UPCS_OVRD16	UPCS layer override register 16	000033CH	P	P	Kernel Reset	2843
PCIE0_USP_UPCS_OVRD17	UPCS layer override register 17	0000340H	P	P	Kernel Reset	2843
PCIE0_USP_INT_MSICTRL	Request MSI interrupt, EP only.	0000348H	P	P	Kernel Reset	2844
PCIE0_USP_INT_MSISTAT	Pending status of MSI interrupt request, EP only.	000034CH	P	P	Kernel Reset	2844
PCIE0_USP_INT_INTXCTRL	Trigger INTX interrupt, EP only.	0000350H	P	P	Kernel Reset	2845
PCIE0_USP_INT_TRIGSTAT	Trigger status	0000358H	P	P	Kernel Reset	2845
PCIE0_USP_INT_TRIGCLR	Interrupt clear, EP only.	000035CH	P	P	Kernel Reset	2846
PCIE0_USP_INT_TRIGEN	Interrupt enable EP only.	0000360H	P	P	Kernel Reset	2846
PCIE0_USP_INT_REQTRIG	Receive incoming interrupt request, EP only.	0000364H	P	P	Kernel Reset	2847
PCIE0_USP_CFG_INFO	Configuration information	0000368H	P	P	Kernel Reset	2847
PCIE0_USP_CFG_BUSNUM	Bus/device number	000036CH	P	P	Kernel Reset	2849
PCIE0_USP_LNK_DVCTYP	Select device/port type	0000370H	P	P, E	PowerOn Reset	2849
PCIE0_USP_LNK_STAT	Indicate link status	0000374H	P	P	Kernel Reset	2850
PCIE0_USP_LNK_CORECTRL	Controller control	0000378H	P	P	Kernel Reset	2852
PCIE0_USP_LNK_INTEN	Enable the interrupt requests to SRC_PCIELNK	000037CH	P	P	Kernel Reset	2854
PCIE0_USP_TXRX_RXCTRL	Rx control	0000380H	P	P	Kernel Reset	2855

(table continues...)

Table 921 (continued) Register overview - PCIE0_USP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE0_USP_TXRX_TXCTRL0	Tx control 0	0000384H	P	P	Kernel Reset	2856
PCIE0_USP_PAYLOADz (z=0-1)	Message payload z information	000038CH+z*4	P	P	Kernel Reset	2857
PCIE0_USP_MSG_INFO	Message information	0000394H	P	P	Kernel Reset	2857
PCIE0_USP_PM_CTRL	Power management control	0000398H	P	P	Kernel Reset	2859
PCIE0_USP_PM_STAT0	Power management status 0	000039CH	P	P	Kernel Reset	2861
PCIE0_USP_PM_STAT1	Power management status 1	00003A4H	P	P	Kernel Reset	2863
PCIE0_USP_PM_INTCLR	Power management interrupt clear	00003A8H	P	P	Kernel Reset	2863
PCIE0_USP_PM_INTEN	Power management interrupt enable	00003ACH	P	P	Kernel Reset	2864
PCIE0_USP_ERR_STAT	Error status	00003B0H	P	P	Kernel Reset	2865
PCIE0_USP_ERR_INTCLR	Error clear	00003B4H	P	P	Kernel Reset	2865
PCIE0_USP_ERR_INTEN	Error enable	00003B8H	P	P	Kernel Reset	2866
PCIE0_USP_DBG_STAT	Debug status	00003BCH	P	P	Kernel Reset	2867
PCIE0_USP_DBG_DATA	Debug information	00003C0H	P	P	Kernel Reset	2868
PCIE0_USP_LTR_GNRTN	LTR message	00003C4H	P	P	Kernel Reset	2868
PCIE0_USP_LTR_CTRL	LTR control	00003CCH	P	P	Kernel Reset	2868
PCIE0_USP_PTM_CTRL	PTM control	00003D0H	P	P	Kernel Reset	2869
PCIE0_USP_LCLCLKz (z=0-1)	Local Clock z value	00003D4H+z*4	P	P	Kernel Reset	2870
PCIE0_USP_CLKCORRz (z=0-1)	Clock z correction	00003DCH+z*4	P	P	Kernel Reset	2870
PCIE0_USP_MSKCLKz (z=0-1)	PTM z External Master Time	00003E4H+z*4	P	P	Kernel Reset	2871

(table continues...)

Table 921 (continued) Register overview - PCIE0_USP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE0_USP_OBFFMSG	OBFF message control	00003F0$_H$	P	P	Kernel Reset	2872
PCIE0_USP_OBFFWAKE	OBFF WAKE control	00003F4$_H$	P	P	Kernel Reset	2874
PCIE0_USP_CFGBD	PCIe configuration control	00003F8$_H$	P	P	Kernel Reset	2876
PCIE0_USP_DATAx (x=0-7)	ELBI x mailbox register	00003FC$_H$+x*4	P	P	Kernel Reset	2876
PCIE0_USP_LLCTRL	DMA link list control register	000041C$_H$	P	P	Kernel Reset	2877
PCIE0_USP_OTSS	PCIe OCDS Trigger Set Select	0000420$_H$	P	SV, P	Debug Reset	2877
PCIE0_USP_ALARM_STAT	Interrupt/alarm status	0000424$_H$	P	P	Kernel Reset	2878
PCIE0_USP_ALARM_INTCLR	Interrupt/alarm clear	0000428$_H$	P	P	Kernel Reset	2880
PCIE0_USP_ALARM_INTEN	Interrupt/alarm enable	000042C$_H$	P	P	Kernel Reset	2881
PCIE0_USP_IATU_REGION_CTRL_1_OFF_OUTBOUND_i (i=0-7)	iATU Region i Control 1 Register Outbound	0003000$_H$+i*200$_H$	P	P	Kernel Reset	2882
PCIE0_USP_IATU_REGION_CTRL_2_OFF_OUTBOUND_i (i=0-7)	iATU Region i Control 2 Register Outbound	0003004$_H$+i*200$_H$	P	P	Kernel Reset	2883
PCIE0_USP_IATU_LWR_BASE_ADDR_OFF_OUTBOUND_i (i=0-7)	iATU i Lower Base Address Register Outbound	0003008$_H$+i*200$_H$	P	P	Kernel Reset	2885
PCIE0_USP_IATU_UPPER_BASE_ADDR_OFF_OUTBOUND_i (i=0-7)	iATU i Upper Base Address Register Outbound	000300C$_H$+i*200$_H$	P	P	Kernel Reset	2886
PCIE0_USP_IATU_LIMIT_ADDR_OFF_OUTBOUND_i (i=0-7)	iATU i Limit Address Register Outbound	0003010$_H$+i*200$_H$	P	P	Kernel Reset	2886

(table continues...)

Table 921 (continued) Register overview - PCIE0_USP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE0_USP_IATU_LWR_TARGET_ADDR_OFF_OUTBOUND_i (i=0-7)	iATU i Lower Target Address Register Outbound	0003014$_H$+i*200$_H$	P	P	Kernel Reset	2887
PCIE0_USP_IATU_UPPER_TARGET_ADDR_OFF_OUTBOUND_i (i=0-7)	iATU i Upper Target Address Register Outbound	0003018$_H$+i*200$_H$	P	P	Kernel Reset	2887
PCIE0_USP_IATU_REGION_CTRL_1_OFF_INBOUND_i (i=0-7)	iATU Region i Control 1 Register Inbound	0003100$_H$+i*200$_H$	P	P	Kernel Reset	2888
PCIE0_USP_IATU_REGION_CTRL_2_OFF_INBOUND_i (i=0-7)	iATU Region i Control 2 Register Inbound	0003104$_H$+i*200$_H$	P	P	Kernel Reset	2889
PCIE0_USP_IATU_LWR_BASE_ADDR_OFF_INBOUND_i (i=0-7)	iATU i Lower Base Address Register Inbound	0003108$_H$+i*200$_H$	P	P	Kernel Reset	2893
PCIE0_USP_IATU_UPPER_BASE_ADDR_OFF_INBOUND_i (i=0-7)	iATU i Upper Base Address Register Inbound	000310C$_H$+i*200$_H$	P	P	Kernel Reset	2893
PCIE0_USP_IATU_LIMIT_ADDR_OFF_INBOUND_i (i=0-7)	iATU i Limit Address Register Inbound	0003110$_H$+i*200$_H$	P	P	Kernel Reset	2894
PCIE0_USP_IATU_LWR_TARGET_ADDR_OFF_INBOUND_i (i=0-7)	iATU i Lower Target Address Register Inbound	0003114$_H$+i*200$_H$	P	P	Kernel Reset	2895
PCIE0_USP_DEVICE_ID_VENDOR_ID_REG	Device ID and Vendor ID Register	0004000$_H$	P	P	Kernel Reset	2895
PCIE0_USP_STATUS_COMMAND_REG	Status and Command Register	0004004$_H$	P	P	Kernel Reset	2896

(table continues...)

Table 921 (continued) Register overview - PCIE0_USP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE0_USP_CLASS_CODE_REVISION_ID	Class Code and Revision ID Register	0004008H	P	P	Kernel Reset	2900
PCIE0_USP_BIST_HEADER_TYPE_LATENCY_CACHE_LINE_SIZE_REG	BIST, Header Type, Latency Timer, and Cache Line Size Register	000400CH	P	P	Kernel Reset	2901
PCIE0_USP_BARi_REG (i=0-5)	BAR0 Register i	0004010H+i*4	P	P	Kernel Reset	2902
PCIE0_USP_CARDBUS_CIS_PTR_REG	CardBus CIS Pointer Register	0004028H	P	P	Kernel Reset	2903
PCIE0_USP_SUBSYSTEM_ID_SUBSYSTEM_VENDOR_ID_REG	Subsystem ID and Subsystem Vendor ID Register	000402CH	P	P	Kernel Reset	2904
PCIE0_USP_PCI_CAP_PTR_REG	Capabilities Pointer Register	0004034H	P	P	Kernel Reset	2905
PCIE0_USP_MAX_LATENCY_MIN_GRANT_INTERRUPT_PIN_INTERRUPT_LINE_REG	Max_Lat, Min_Gnt, Interrupt Pin, and Interrupt Line Register	000403CH	P	P	Kernel Reset	2905
PCIE0_USP_CAP_ID_NXT_PTR_REG	Power Management Capabilities Register	0004040H	P	P	Kernel Reset	2906
PCIE0_USP_CON_STATUS_REG	Power Management Control and Status Register	0004044H	P	P	Kernel Reset	2909
PCIE0_USP_PCI_MSI_CAP_ID_NEXT_CTRL_REG	MSI Capability Header and Message Control Register	0004050H	P	P	Kernel Reset	2912
PCIE0_USP_MSI_CAP_OFF_04H_REG	Message Address Register for MSI (Offset 04h)	0004054H	P	P	Kernel Reset	2914
PCIE0_USP_MSI_CAP_OFF_08H_REG	Message Address Register for MSI (Offset 08h)	0004058H	P	P	Kernel Reset	2915
PCIE0_USP_MSI_CAP_OFF_0CH_REG	Message Address Register for MSI (Offset 0Ch)	000405CH	P	P	Kernel Reset	2916
PCIE0_USP_MSI_CAP_OFF_10H_REG	Message Address Register for MSI (Offset 10h)	0004060H	P	P	Kernel Reset	2917

(table continues...)

Table 921 (continued) Register overview - PCIE0_USP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE0_USP_MSI_CAP_OFF_14H_REG	Message Address Register for MSI (Offset 14h)	0004064H	P	P	Kernel Reset	2917
PCIE0_USP_PCIE_CAP_ID_PCIE_NEXT_CAP_PTR_PCIE_CAP_REG	PCI Express Capabilities, ID, Next Pointer Register	0004070H	P	P	Kernel Reset	2918
PCIE0_USP_DEVICE_CAPABILITIES_REG	Device Capabilities Register	0004074H	P	P	Kernel Reset	2920
PCIE0_USP_DEVICE_CONTROL_DEVICE_STATUS	Device Control and Device Status Register	0004078H	P	P	Kernel Reset	2924
PCIE0_USP_LINK_CAPABILITIES_REG	Link Capabilities Register	000407CH	P	P	Kernel Reset	2929
PCIE0_USP_LINK_CONTROL_LINK_STATUS_REG	Link Control and Link Status Register	0004080H	P	P	Kernel Reset	2934
PCIE0_USP_DEVICE_CAPABILITIES2_REG	Device Capabilities 2 Register	0004094H	P	P	Kernel Reset	2941
PCIE0_USP_DEVICE_CONTROL2_DEVICE_STATUS2_REG	Device Control 2 and Status 2 Register	0004098H	P	P	Kernel Reset	2945
PCIE0_USP_LINK_CAPABILITIES2_REG	Link Capabilities 2 Register	000409CH	P	P	Kernel Reset	2947
PCIE0_USP_LINK_CONTROL2_LINK_STATUS2_REG	Link Control 2 and Status 2 Register	00040A0H	P	P	Kernel Reset	2949
PCIE0_USP_PCI_MSIX_CAP_ID_NEXT_CTRL_REG	MSI-X Capability ID, Next Pointer, Control Register	00040B0H	P	P	Kernel Reset	2955
PCIE0_USP_MSIX_TABLE_OFFSET_REG	MSI-X Table Offset and BIR Register	00040B4H	P	P	Kernel Reset	2957
PCIE0_USP_MSIX_PBA_OFFSET_REG	MSI-X PBA Offset and BIR Register	00040B8H	P	P	Kernel Reset	2957
PCIE0_USP_AER_EXT_CAP_HDR_OFF	Advanced Error Reporting Extended Capability Header	0004100H	P	P	Kernel Reset	2958

(table continues...)

Table 921 (continued) Register overview - PCIE0_USP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE0_USP_UNCORR_ERR_STATUS_OFF	Uncorrectable Error Status Register	0004104H	P	P	Kernel Reset	2959
PCIE0_USP_UNCORR_ERR_MASK_OFF	Uncorrectable Error Mask Register	0004108H	P	P	Kernel Reset	2961
PCIE0_USP_UNCORR_ERR_SEV_OFF	Uncorrectable Error Severity Register	000410CH	P	P	Kernel Reset	2963
PCIE0_USP_CORR_ERR_STATUS_OFF	Correctable Error Status Register	0004110H	P	P	Kernel Reset	2965
PCIE0_USP_CORR_ERR_MASK_OFF	Correctable Error Mask Register	0004114H	P	P	Kernel Reset	2966
PCIE0_USP_ADV_ERR_CAP_CTRL_OFF	Advanced Error Capabilities and Control Register	0004118H	P	P	Kernel Reset	2967
PCIE0_USP_HDR_LOG_i_OFF (i=0-3)	Header i Log Register 0	000411CH+i*4	P	P	Kernel Reset	2969
PCIE0_USP_TLP_PREFIX_LOG_i_OFF (i=0-3)	TLP Prefix i Log Register 1	0004138H+i*4	P	P	Kernel Reset	2969
PCIE0_USP_SN_BASE	Device Serial Number Extended Capability Header	0004148H	P	P	Kernel Reset	2970
PCIE0_USP_SER_NUM_REG_DW_1	Serial Number 1 Register	000414CH	P	P	Kernel Reset	2971
PCIE0_USP_SER_NUM_REG_DW_2	Serial Number 2 Register	0004150H	P	P	Kernel Reset	2971
PCIE0_USP_SPCIE_CAP_HEADER_REG	SPCIE Capability Header	0004158H	P	P	Kernel Reset	2972
PCIE0_USP_LINK_CONTROL3_REG	Link Control 3 Register	000415CH	P	P	Kernel Reset	2973
PCIE0_USP_LANE_ERR_STATUS_REG	Lane Error Status Register	0004160H	P	P	Kernel Reset	2974

(table continues...)

Table 921 (continued) Register overview - PCIE0_USP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE0_USP_SPCIE_CAP_OFF_0CH_REG	Lane Equalization Control Register for lanes 1 and 0	0004164H	P	P	Kernel Reset	2975
PCIE0_USP_LTR_CAP_HDR_REG	LTR Extended Capability Header	0004168H	P	P	Kernel Reset	2976
PCIE0_USP_LTR_LATENCY_REG	LTR Max Snoop and No-Snoop Latency Register	000416CH	P	P	Kernel Reset	2977
PCIE0_USP_L1SUB_CAP_HEADER_REG	L1 Substates Extended Capability Header	0004170H	P	P	Kernel Reset	2978
PCIE0_USP_L1SUB_CAPABILITY_REG	L1 Substates Capability Register	0004174H	P	P	Kernel Reset	2979
PCIE0_USP_L1SUB_CONTROL1_REG	L1 Substates Control 1 Register	0004178H	P	P	Kernel Reset	2981
PCIE0_USP_L1SUB_CONTROL2_REG	L1 Substates Control 2 Register	000417CH	P	P	Kernel Reset	2982
PCIE0_USP_RAS_DES_CAP_HEADER_REG	Vendor-Specific Extended Capability Header	0004180H	P	P	Kernel Reset	2984
PCIE0_USP_VENDOR_SPECIFIC_HEADER_REG	Vendor-Specific Header	0004184H	P	P	Kernel Reset	2985
PCIE0_USP_EVENT_COUNTER_CONTROL_REG	Event Counter Control	0004188H	P	P	Kernel Reset	2985
PCIE0_USP_EVENT_COUNTER_DATA_REG	Event Counter Data	000418CH	P	P	Kernel Reset	2987
PCIE0_USP_TIME_BASED_ANALYSIS_CONTROL_REG	Time-based Analysis Control	0004190H	P	P	Kernel Reset	2988
PCIE0_USP_TIME_BASED_ANALYSIS_DATA_REG	Time-based Analysis Data	0004194H	P	P	Kernel Reset	2989
PCIE0_USP_TIME_BASED_ANALYSIS_DATA_63_32_REG	Upper 32 bits of Time-based Analysis Data	0004198H	P	P	Kernel Reset	2990
PCIE0_USP_EINJ_ENABLE_REG	Error Injection Enable	00041B0H	P	P	Kernel Reset	2990

(table continues...)

Table 921 (continued) Register overview - PCIE0_USP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE0_USP_EINJ0_CRC_REG	Error Injection Control 0 (CRC Error)	00041B4$_H$	P	P	Kernel Reset	2992
PCIE0_USP_EINJ1_SEQNUM_REG	Error Injection Control 1 (Sequence Number Error). 	00041B8$_H$	P	P	Kernel Reset	2993
PCIE0_USP_EINJ2_DLLP_REG	Error Injection Control 2 (DLLP Error)	00041BC$_H$	P	P	Kernel Reset	2994
PCIE0_USP_EINJ3_SYMBOL_REG	Error Injection Control 3 (Symbol Error)	00041C0$_H$	P	P	Kernel Reset	2995
PCIE0_USP_EINJ4_FC_REG	Error Injection Control 4 (FC Credit Error)	00041C4$_H$	P	P	Kernel Reset	2996
PCIE0_USP_EINJ5_SP_TLP_REG	Error Injection Control 5 (Specific TLP Error)	00041C8$_H$	P	P	Kernel Reset	2997
PCIE0_USP_EINJ6_COMPARE_POINT_Hi_REG (i=0-3)	Error Injection Control 6 (Compare Point Header DWORD #0) i	00041CC$_H$+i*4	P	P	Kernel Reset	2998
PCIE0_USP_EINJ6_COMPARE_VALUE_Hi_REG (i=0-3)	Error Injection Control 6 (Compare Value Header DWORD #0) i	00041DC$_H$+i*4	P	P	Kernel Reset	2999
PCIE0_USP_EINJ6_CHANGE_POINT_Hi_REG (i=0-3)	Error Injection Control 6 (Change Point Header DWORD #0) i	00041EC$_H$+i*4	P	P	Kernel Reset	2999
PCIE0_USP_EINJ6_CHANGE_VALUE_Hi_REG (i=0-3)	Error Injection Control 6 (Change Value Header DWORD #0) i	00041FC$_H$+i*4	P	P	Kernel Reset	3000
PCIE0_USP_EINJ6_TLP_REG	Error Injection Control 6 (Packet Error)	000420C$_H$	P	P	Kernel Reset	3001
PCIE0_USP_SD_CONTROL1_REG	Silicon Debug Control 1	0004220$_H$	P	P	Kernel Reset	3002
PCIE0_USP_SD_CONTROL2_REG	Silicon Debug Control 2	0004224$_H$	P	P	Kernel Reset	3003
PCIE0_USP_SD_STATUS_L1LANE_REG	Silicon Debug Status(Layer1 Per-lane)	0004230$_H$	P	P	Kernel Reset	3004
PCIE0_USP_SD_STATUS_L1LTSSM_REG	Silicon Debug Status(Layer1 LTSSM)	0004234$_H$	P	P	Kernel Reset	3005
PCIE0_USP_SD_STATUS_PM_REG	Silicon Debug Status(PM)	0004238$_H$	P	P	Kernel Reset	3008

(table continues...)

Table 921 (continued) Register overview - PCIE0_USP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE0_USP_SD_STATUS_L2_REG	Silicon Debug Status(Layer2)	000423C H	P	P	Kernel Reset	3011
PCIE0_USP_SD_STATUS_L3FC_REG	Silicon Debug Status(Layer3 FC)	0004240 H	P	P	Kernel Reset	3011
PCIE0_USP_SD_STATUS_L3_REG	Silicon Debug Status(Layer3)	0004244 H	P	P	Kernel Reset	3013
PCIE0_USP_SD_EQ_CONTROL1_REG	Silicon Debug EQ Control 1	0004250 H	P	P	Kernel Reset	3014
PCIE0_USP_SD_EQ_CONTROL2_REG	Silicon Debug EQ Control 2	0004254 H	P	P	Kernel Reset	3016
PCIE0_USP_SD_EQ_CONTROL3_REG	Silicon Debug EQ Control 3	0004258 H	P	P	Kernel Reset	3017
PCIE0_USP_SD_EQ_STATUS1_REG	Silicon Debug EQ Status 1	0004260 H	P	P	Kernel Reset	3018
PCIE0_USP_SD_EQ_STATUS2_REG	Silicon Debug EQ Status 2	0004264 H	P	P	Kernel Reset	3019
PCIE0_USP_SD_EQ_STATUS3_REG	Silicon Debug EQ Status 3	0004268 H	P	P	Kernel Reset	3020
PCIE0_USP_RASDP_EXT_CAP_HDR_OFF	PCIe Extended capability ID, Capability version and Next capability offset	0004280 H	P	P	Kernel Reset	3021
PCIE0_USP_RASDP_VENDOR_SPECIFIC_HDR_OFF	Vendor Specific Header	0004284 H	P	P	Kernel Reset	3022
PCIE0_USP_RASDP_ERROR_PROT_CTRL_OFF	ECC Error Correction and Control Register	0004288 H	P	P	Kernel Reset	3023
PCIE0_USP_RASDP_ERROR_MODE_EN_OFF	RASDP error mode enable.	00042A8 H	P	P	Kernel Reset	3035
PCIE0_USP_RASDP_ERROR_MODE_CLEAR_OFF	RASDP Error Mode Clear Register	00042AC H	P	P	Kernel Reset	3036
PCIE0_USP_RASDP_RAM_ADDR_CORR_ERROR_OFF	RAM Address where a corrected error (1-bit ECC) is detected	00042B0 H	P	P	Kernel Reset	3037

(table continues...)

Table 921 (continued) Register overview - PCIE0_USP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE0_USP_RASDP_RAM_ADDR_UNCORR_ERROR_OFF	RAM Address where an uncorrected error (2-bit ECC) is detected	00042B4H	P	P	Kernel Reset	3037
PCIE0_USP_PTM_EXT_CAP_HDR_OFF	Precision Time Measurement Capability Header	00042B8H	P	P	Kernel Reset	3038
PCIE0_USP_PTM_CAP_OFF	PTM Capability Register	00042BCH	P	P	Kernel Reset	3039
PCIE0_USP_PTM_CONTROL_OFF	PTM Control Register	00042C0H	P	P	Kernel Reset	3040
PCIE0_USP_PTM_REQ_CAP_HDR_OFF	Precision Time Measurement Requester Capability Header (VSEC)	00042C4H	P	P	Kernel Reset	3041
PCIE0_USP_PTM_REQ_HDR_OFF	Precision Time Measurement Requester Vendor Specific Header	00042C8H	P	P	Kernel Reset	3042
PCIE0_USP_PTM_REQ_CONTROL_OFF	PTM Requester Vendor Specific Control Register	00042CCH	P	P	Kernel Reset	3043
PCIE0_USP_PTM_REQ_STATUS_OFF	PTM Requester Vendor Specific Status Register	00042D0H	P	P	Kernel Reset	3044
PCIE0_USP_PTM_REQ_LOCAL_LSB_OFF	PTM Requester Local Clock LSB	00042D4H	P	P	Kernel Reset	3045
PCIE0_USP_PTM_REQ_LOCAL_MSB_OFF	PTM Requester Local Clock MSB	00042D8H	P	P	Kernel Reset	3046
PCIE0_USP_PTM_REQ_T1_LSB_OFF	PTM Requester T1 Timestamp LSB	00042DCH	P	P	Kernel Reset	3046
PCIE0_USP_PTM_REQ_T1_MSB_OFF	PTM Requester T1 Timestamp MSB	00042E0H	P	P	Kernel Reset	3047
PCIE0_USP_PTM_REQ_T1P_LSB_OFF	PTM Requester T1 Previous Timestamp LSB	00042E4H	P	P	Kernel Reset	3047
PCIE0_USP_PTM_REQ_T1P_MSB_OFF	PTM Requester T1 Previous Timestamp MSB	00042E8H	P	P	Kernel Reset	3047
PCIE0_USP_PTM_REQ_T4_LSB_OFF	PTM Requester T4 Timestamp LSB	00042ECH	P	P	Kernel Reset	3048

(table continues...)

Table 921 (continued) Register overview - PCIE0_USP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE0_USP_PTM_REQ_T4_MSB_OFF	PTM Requester T4 Timestamp MSB	00042F0H	P	P	Kernel Reset	3048
PCIE0_USP_PTM_REQ_T4P_LSB_OFF	PTM Requester T4 Previous Timestamp LSB	00042F4H	P	P	Kernel Reset	3049
PCIE0_USP_PTM_REQ_T4P_MSB_OFF	PTM Requester T4 Previous Timestamp MSB	00042F8H	P	P	Kernel Reset	3049
PCIE0_USP_PTM_REQ_MASTER_LSB_OFF	PTM Requester Master Time LSB	00042FCH	P	P	Kernel Reset	3050
PCIE0_USP_PTM_REQ_MASTER_MSB_OFF	PTM Requester Master Time MSB	0004300H	P	P	Kernel Reset	3050
PCIE0_USP_PTM_REQ_PROP_DELAY_OFF	PTM Requester Propagation Delay	0004304H	P	P	Kernel Reset	3051
PCIE0_USP_PTM_REQ_MASTERT1_LSB_OFF	PTM Requester Master Time at T1 LSB	0004308H	P	P	Kernel Reset	3051
PCIE0_USP_PTM_REQ_MASTERT1_MSB_OFF	PTM Requester Master Time at T1 MSB	000430CH	P	P	Kernel Reset	3052
PCIE0_USP_PTM_REQ_TX_LATENCY_OFF	PTM Requester TX Latency	0004310H	P	P	Kernel Reset	3052
PCIE0_USP_PTM_REQ_RX_LATENCY_OFF	PTM Requester RX Latency	0004314H	P	P	Kernel Reset	3053
PCIE0_USP_PTM_REQ_CLOCK_CORR_LSB_OFF	PTM Requester Clock Correction LSB	0004318H	P	P	Kernel Reset	3054
PCIE0_USP_PTM_REQ_CLOCK_CORR_MSB_OFF	PTM Requester Clock Correction MSB	000431CH	P	P	Kernel Reset	3054
PCIE0_USP_PTM_REQ_NOM_CLOCK_T_OFF	PTM Requester Nominal Clock Period	0004320H	P	P	Kernel Reset	3055
PCIE0_USP_PTM_REQ_SCALED_CLOCK_T_OFF	PTM Requester Scaled Clock Period	0004324H	P	P	Kernel Reset	3055
PCIE0_USP_PTM_REQ_LATENCY_REG_SEL_OFF	PTM Requester Latency Viewport Register Select	0004328H	P	P	Kernel Reset	3056

(table continues...)

Table 921 (continued) Register overview - PCIE0_USP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE0_USP_VSECDMA_EXT_CAP_HDR_OFF	PCIe Extended Capability ID, Capability Version, and Next Capability Offset Register	000432C H	P	P	Kernel Reset	3057
PCIE0_USP_VSECDMA_VENDOR_SPECIFIC_HDR_OFF	Vendor Specific Header Register	0004330 H	P	P	Kernel Reset	3058
PCIE0_USP_VSECDMA_DEVICE_INFORMATION_OFF	DMA and related AXI Bridge Implementation Information	0004334 H	P	P	Kernel Reset	3059
PCIE0_USP_VSECDMA_NUM_CHAN_OFF	Number of Implemented Channels Register	0004338 H	P	P	Kernel Reset	3060
PCIE0_USP_VSECDMA_UNROLL_ADDR_OFFSET_LOW_OFF	DMA Register Map Start Address Offset Low Register	000433C H	P	P	Kernel Reset	3061
PCIE0_USP_VSECDMA_UNROLL_ADDR_OFFSET_HIGH_OFF	DMA Register Map Start Address Offset High Register	0004340 H	P	P	Kernel Reset	3061
PCIE0_USP_ACK_LATENCY_TIMER_OFF	Ack Latency Timer and Replay Timer Register	0004700 H	P	P	Kernel Reset	3062
PCIE0_USP_VENDOR_SPEC_DLLP_OFF	Vendor Specific DLLP Register	0004704 H	P	P	Kernel Reset	3063
PCIE0_USP_PORT_FORCE_OFF	Port Force Link Register	0004708 H	P	P	Kernel Reset	3063
PCIE0_USP_ACK_F_ASPM_CTRL_OFF	Ack Frequency and L0-L1 ASPM Control Register	000470C H	P	P	Kernel Reset	3065
PCIE0_USP_PORT_LINK_CTRL_OFF	Port Link Control Register	0004710 H	P	P	Kernel Reset	3067
PCIE0_USP_LANE_SKEW_OFF	Lane Skew Register	0004714 H	P	P	Kernel Reset	3069
PCIE0_USP_TIMER_CTRL_MAX_FUNC_NUM_OFF	Timer Control and Max Function Number Register	0004718 H	P	P	Kernel Reset	3071
PCIE0_USP_SYMBOL_TIMER_FILTER_1_OFF	Symbol Timer Register and Filter Mask 1 Register	000471C H	P	P	Kernel Reset	3072

(table continues...)

Table 921 (continued) Register overview - PCIE0_USP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE0_USP_FILTER_MASK_2_OFF	Filter Mask 2 Register	0004720H	P	P	Kernel Reset	3075
PCIE0_USP_PL_DEBUG0_OFF	Debug Register 0	0004728H	P	P	Kernel Reset	3077
PCIE0_USP_PL_DEBUG1_OFF	Debug Register 1	000472CH	P	P	Kernel Reset	3078
PCIE0_USP_TX_P_FC_CREDIT_STATUS_OFF	Transmit Posted FC Credit Status	0004730H	P	P	Kernel Reset	3079
PCIE0_USP_TX_NP_FC_CREDIT_STATUS_OFF	Transmit Non-Posted FC Credit Status	0004734H	P	P	Kernel Reset	3080
PCIE0_USP_TX_CPL_FC_CREDIT_STATUS_OFF	Transmit Completion FC Credit Status	0004738H	P	P	Kernel Reset	3081
PCIE0_USP_QUEUE_STATUS_OFF	Queue Status	000473CH	P	P	Kernel Reset	3082
PCIE0_USP_VC0_P_RX_Q_CTRL_OFF	Segmented-Buffer VC0 Posted Receive Queue Control	0004748H	P	P	Kernel Reset	3084
PCIE0_USP_VC0_NP_RX_Q_CTRL_OFF	Segmented-Buffer VC0 Non-Posted Receive Queue Control	000474CH	P	P	Kernel Reset	3086
PCIE0_USP_VC0_CPL_RX_Q_CTRL_OFF	Segmented-Buffer VC0 Completion Receive Queue Control	0004750H	P	P	Kernel Reset	3087
PCIE0_USP_GEN2_CTRL_OFF	Link Width and Speed Change Control Register	000480CH	P	P	Kernel Reset	3088
PCIE0_USP_CLOCK_GATING_CTRL_OFF	Clock Gating Control Register	000488CH	P	P	Kernel Reset	3092
PCIE0_USP_GEN3_RELATED_OFF	Gen3 Control Register	0004890H	P	P	Kernel Reset	3093
PCIE0_USP_GEN3_EQ_CONTROL_OFF	Gen3 EQ Control Register	00048A8H	P	P	Kernel Reset	3096
PCIE0_USP_GEN3_EQ_FB_MODE_DIR_CHANGE_OFF	Gen3 EQ Direction Change Feedback Mode Control Register	00048ACH	P	P	Kernel Reset	3099
PCIE0_USP_ORDER_RULE_CTRL_OFF	Order Rule Control Register	00048B4H	P	P	Kernel Reset	3100

(table continues...)

Table 921 (continued) Register overview - PCIE0_USP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE0_USP_PIPE_LOOPBACK_CONTROL_OFF	PIPE Loopback Control Register	00048B8H	P	P	Kernel Reset	3101
PCIE0_USP_MISC_CONTROL_1_OFF	DBI Read-Only Write Enable Register	00048BCH	P	P	Kernel Reset	3102
PCIE0_USP_LINK_FLUSH_CONTROL_OFF	Link Reset Request Flush Control Register	00048CCH	P	P	Kernel Reset	3108
PCIE0_USP_AMBA_ERROR_RESPONSE_DEFAULT_OFF	AXI Bridge Slave Error Response Register	00048D0H	P	P	Kernel Reset	3109
PCIE0_USP_AMBA_LINK_TIMEOUT_OFF	Link Down AXI Bridge Slave Timeout Register	00048D4H	P	P	Kernel Reset	3110
PCIE0_USP_AMBA_ORDERING_CTRL_OFF	AXI Bridge Ordering Control	00048D8H	P	P	Kernel Reset	3111
PCIE0_USP_PCIE_VERSION_NUMBER_OFF	PCIe Controller IIP Release Version Number	00048F8H	P	P	Kernel Reset	3115
PCIE0_USP_PCIE_VERSION_TYPE_OFF	PCIe Controller IIP Release Version Type	00048FCH	P	P	Kernel Reset	3116
PCIE0_USP_INTERFACE_TIMER_CONTROL_OFF	Interface Timer Control Register	0004930H	P	P	Kernel Reset	3116
PCIE0_USP_INTERFACE_TIMER_TARGET_OFF	Interface Timer Target Register	0004934H	P	P	Kernel Reset	3117
PCIE0_USP_INTERFACE_TIMER_STATUS_OFF	Interface Timer Status Register	0004938H	P	P	Kernel Reset	3118
PCIE0_USP_MSIX_ADDRESS_MATCH_LOW_OFF	MSI-X Address Match Low Register	0004940H	P	P	Kernel Reset	3119
PCIE0_USP_MSIX_ADDRESS_MATCH_HIGH_OFF	MSIX Address Match High Register	0004944H	P	P	Kernel Reset	3120
PCIE0_USP_MSIX_DOORBELL_OFF	MSI-X Doorbell Register	0004948H	P	P	Kernel Reset	3120
PCIE0_USP_SAFETY_MASK_OFF	Functional Safety Interrupt Events Mask Register	0004960H	P	P	Kernel Reset	3123

(table continues...)

Table 921 (continued) Register overview - PCIE0_USP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE0_USP_SAFETY_STATUS_OFF	Status for functional safety interrupt events	0004964$_H$	P	P	Kernel Reset	3124
PCIE0_USP_PL_APP_BUS_DEV_NUM_STATUS_OFF	Application driven bus and device number register	0004B10$_H$	P	P	Kernel Reset	3125
PCIE0_USP_PL_CHK_REG_CONTROL_STATUS_OFF	CDM Register Checking Control and Status Register	0004B20$_H$	P	P	Kernel Reset	3126
PCIE0_USP_PL_CHK_REG_START_END_OFF	CDM Register Checking First and Last address to check	0004B24$_H$	P	P	Kernel Reset	3128
PCIE0_USP_PL_CHK_REG_ERR_ADDR_OFF	CDM Register Checking Error Address	0004B28$_H$	P	P	Kernel Reset	3128
PCIE0_USP_PL_CHK_REG_ERR_PF_VF_OFF	CDM Register Checking error PF Numbers	0004B2C$_H$	P	P	Kernel Reset	3129
PCIE0_USP_PL_LTR_LATENCY_OFF	LTR Latency Register	0004B30$_H$	P	P	Kernel Reset	3129
PCIE0_USP_AUX_CLK_FREQ_OFF	Auxiliary Clock Frequency Control Register	0004B40$_H$	P	P	Kernel Reset	3130
PCIE0_USP_POWERDOWN_CTRL_STATUS_OFF	Powerdown Control and Status Register	0004B48$_H$	P	P	Kernel Reset	3132
PCIE0_USP_MSIXTABLEx_LADDR (x=0-63)	MSIX table x DWORD0	0005A00$_H$+x*10$_H$	P	P	Kernel Reset	3134
PCIE0_USP_MSIXTABLEx_UADDR (x=0-63)	MSIX table x DWORD1	0005A04$_H$+x*10$_H$	P	P	Kernel Reset	3134
PCIE0_USP_MSIXTABLEx_DATA (x=0-63)	MSIX table x DWORD2	0005A08$_H$+x*10$_H$	P	P	Kernel Reset	3135
PCIE0_USP_MSIXTABLEx_VECTRL (x=0-63)	MSIX table x DWORD3	0005A0C$_H$+x*10$_H$	P	P	Kernel Reset	3135
PCIE0_USP_MSIXPBAx_DWORD (x=0-1)	MSIX xtable DWORD	0005E10$_H$+x*4	P	P	Kernel Reset	3136

(table continues...)

Table 921 (continued) Register overview - PCIE0_USP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE0_USP_BARi_MASK_REG (i=0-5)	BAR i Mask Register	0006010$_H$+i*4	P	P	Kernel Reset	3136
PCIE0_USP_SHADOW_PCI_MSIX_CAP_ID_NEXT_CTRL_REG	MSI-X Capability ID, Next Pointer, Control Register	00060B0$_H$	P	P	Kernel Reset	3137
PCIE0_USP_DMA_CTRL_DATA_ARB_PRIOR_OFF	DMA Arbitration Scheme for TRGT1 Interface	0007000$_H$	P	P	Kernel Reset	3139
PCIE0_USP_DMA_CTRL_OFF	DMA Number of Channels Register	0007008$_H$	P	P	Kernel Reset	3140
PCIE0_USP_DMA_WRITE_ENGINE_EN_OFF	DMA Write Engine Enable Register	000700C$_H$	P	P	Kernel Reset	3141
PCIE0_USP_DMA_WRITE_DOORBELL_OFF	DMA Write Doorbell Register	0007010$_H$	P	P	Kernel Reset	3143
PCIE0_USP_DMA_WRITE_CHANNEL_ARB_WEIGHT_LOW_OFF	DMA Write Engine Channel Arbitration Weight Low Register	0007018$_H$	P	P	Kernel Reset	3144
PCIE0_USP_DMA_READ_ENGINE_EN_OFF	DMA Read Engine Enable Register	000702C$_H$	P	P	Kernel Reset	3146
PCIE0_USP_DMA_READ_DOORBELL_OFF	DMA Read Doorbell Register	0007030$_H$	P	P	Kernel Reset	3147
PCIE0_USP_DMA_READ_CHANNEL_ARB_WEIGHT_LOW_OFF	DMA Read Engine Channel Arbitration Weight Low Register	0007038$_H$	P	P	Kernel Reset	3148
PCIE0_USP_DMA_WRITE_INT_STATUS_OFF	DMA Write Interrupt Status Register	000704C$_H$	P	P	Kernel Reset	3150
PCIE0_USP_DMA_WRITE_INT_MASK_OFF	DMA Write Interrupt Mask Register	0007054$_H$	P	P	Kernel Reset	3151
PCIE0_USP_DMA_WRITE_INT_CLEAR_OFF	DMA Write Interrupt Clear Register	0007058$_H$	P	P	Kernel Reset	3152
PCIE0_USP_DMA_WRITE_ERR_STATUS_OFF	DMA Write Error Status Register	000705C$_H$	P	P	Kernel Reset	3153

(table continues...)

Table 921 (continued) Register overview - PCIE0_USP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE0_USP_DMA_WRITE_DONE_IMWR_LOW_OFF	DMA Write Done IMWr Address Low Register	0007060H	P	P	Kernel Reset	3153
PCIE0_USP_DMA_WRITE_DONE_IMWR_HIGH_OFF	DMA Write Done IMWr Interrupt Address High Register	0007064H	P	P	Kernel Reset	3154
PCIE0_USP_DMA_WRITE_ABORT_IMWR_LOW_OFF	DMA Write Abort IMWr Address Low Register	0007068H	P	P	Kernel Reset	3154
PCIE0_USP_DMA_WRITE_ABORT_IMWR_HIGH_OFF	DMA Write Abort IMWr Address High Register	000706CH	P	P	Kernel Reset	3155
PCIE0_USP_DMA_WRITE_CH01_IMWR_DATA_OFF	DMA Write Channel 1 and 0 IMWr Data Register	0007070H	P	P	Kernel Reset	3155
PCIE0_USP_DMA_WRITE_CH23_IMWR_DATA_OFF	DMA Write Channel 3 and 2 IMWr Data Register	0007074H	P	P	Kernel Reset	3156
PCIE0_USP_DMA_WRITE_LINKED_LIST_ERR_EN_OFF	DMA Write Linked List Error Enable Register	0007090H	P	P	Kernel Reset	3157
PCIE0_USP_DMA_READ_INT_STATUS_OFF	DMA Read Interrupt Status Register	00070A0H	P	P	Kernel Reset	3158
PCIE0_USP_DMA_READ_INT_MASK_OFF	DMA Read Interrupt Mask Register	00070A8H	P	P	Kernel Reset	3159
PCIE0_USP_DMA_READ_INT_CLEAR_OFF	DMA Read Interrupt Clear Register	00070ACH	P	P	Kernel Reset	3160
PCIE0_USP_DMA_READ_ERR_STATUS_LOW_OFF	DMA Read Error Status Low Register	00070B4H	P	P	Kernel Reset	3161
PCIE0_USP_DMA_READ_ERR_STATUS_HIGH_OFF	DMA Read Error Status High Register	00070B8H	P	P	Kernel Reset	3162
PCIE0_USP_DMA_READ_LINKED_LIST_ERR_EN_OFF	DMA Read Linked List Error Enable Register	00070C4H	P	P	Kernel Reset	3163
PCIE0_USP_DMA_READ_DONE_IMWR_LOW_OFF	DMA Read Done IMWr Address Low Register	00070CCH	P	P	Kernel Reset	3164

(table continues...)

Table 921 (continued) Register overview - PCIE0_USP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE0_USP_DMA_READ_DONE_IMWR_HIGH_OFF	DMA Read Done IMWr Address High Register	00070D0H	P	P	Kernel Reset	3164
PCIE0_USP_DMA_READ_ABORT_IMWR_LOW_OFF	DMA Read Abort IMWr Address Low Register	00070D4H	P	P	Kernel Reset	3165
PCIE0_USP_DMA_READ_ABORT_IMWR_HIGH_OFF	DMA Read Abort IMWr Address High Register	00070D8H	P	P	Kernel Reset	3165
PCIE0_USP_DMA_READ_CH01_IMWR_DATA_OFF	DMA Read Channel 1 and 0 IMWr Data Register	00070DCH	P	P	Kernel Reset	3166
PCIE0_USP_DMA_READ_CH23_IMWR_DATA_OFF	DMA Read Channel 3 and 2 IMWr Data Register	00070E0H	P	P	Kernel Reset	3166
PCIE0_USP_DMA_WRITE_ENGINE_HSHAKE_CNT_LOW_OFF	DMA Write Engine Handshake Counter Channel 0/1/2/3 Register	0007108H	P	P	Kernel Reset	3168
PCIE0_USP_DMA_READ_ENGINE_HSHAKE_CNT_LOW_OFF	DMA Read Engine Handshake Counter Channel 0/1/2/3 Register	0007118H	P	P	Kernel Reset	3169
PCIE0_USP_DMA_WRITE_CHi_PWR_EN_OFF (i=0-3)	DMA Write Channel i Power Enable Register	0007128H+i*4	P	P	Kernel Reset	3171
PCIE0_USP_DMA_READ_CHi_PWR_EN_OFF (i=0-3)	DMA Read Channel i Power Enable Register	0007168H+i*4	P	P	Kernel Reset	3171
PCIE0_USP_DMA_CH_CONTROL1_OFF_WRCH_i (i=0-3)	DMA Write Channel i Control 1 Register	0007200H+i*200H	P_DMAi	P_DMAi	Kernel Reset	3172
PCIE0_USP_DMA_TRANSFER_SIZE_OFF_WRCH_i (i=0-3)	DMA CH i Write Transfer Size Register	0007208H+i*200H	P_DMAi	P_DMAi	Kernel Reset	3175
PCIE0_USP_DMA_SAR_LOW_OFF_WRCH_i (i=0-3)	DMA CH i Write SAR Low Register	000720CH+i*200H	P_DMAi	P_DMAi	Kernel Reset	3175

(table continues...)

Table 921 (continued) Register overview - PCIE0_USP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE0_USP_DMA_SAR_HIGH_OFF_WRCH_i (i=0-3)	DMA CH i Write SAR High Register	0007210$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3176
PCIE0_USP_DMA_DAR_LOW_OFF_WRCH_i (i=0-3)	DMA CH i Write DAR Low Register	0007214$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3176
PCIE0_USP_DMA_DAR_HIGH_OFF_WRCH_i (i=0-3)	DMA CH i Write DAR High Register	0007218$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3177
PCIE0_USP_DMA_LLP_LOW_OFF_WRCH_i (i=0-3)	DMA CH i Write Linked List Pointer Low Register	000721C$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3177
PCIE0_USP_DMA_LLP_HIGH_OFF_WRCH_i (i=0-3)	DMA CH i Write Linked List Pointer High Register	0007220$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3178
PCIE0_USP_DMA CAP_CHi_WRCH N_WR_INT_CLEA R (i=0-3)	Interrupt clear for DMA write channel i	0007224$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3178
PCIE0_USP_DMA CAP_CHi_WRCH N_WR_INT_STAT US (i=0-3)	Interrupt status for DMA write channel i	0007228$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3179
PCIE0_USP_DMA CAP_CHi_WRCH N_WR_ERR_STAT US (i=0-3)	Error status for DMA write channel i	000722C$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3180
PCIE0_USP_DMA CAP_CHi_WRCH N_WR_DOORBEL L (i=0-3)	Doorbell for DMA write channel i	0007230$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3181
PCIE0_USP_DMA_CH_CONTROL1_OFF_RDCH_i (i=0-3)	DMA Read Channel i Control 1 Register	0007300$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3181

(table continues...)

Table 921 (continued) Register overview - PCIE0_USP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE0_USP_DMA_TRANSFER_SIZE_OFF_RDCH_i (i=0-3)	DMA CH i Read Transfer Size Register	0007308$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3184
PCIE0_USP_DMA_SAR_LOW_OFF_RDCH_i (i=0-3)	DMA CH i Read SAR Low Register	000730C$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3185
PCIE0_USP_DMA_SAR_HIGH_OFF_RDCH_i (i=0-3)	DMA CH i Read SAR High Register	0007310$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3186
PCIE0_USP_DMA_DAR_LOW_OFF_RDCH_i (i=0-3)	DMA CH i Read DAR Low Register	0007314$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3186
PCIE0_USP_DMA_DAR_HIGH_OFF_RDCH_i (i=0-3)	DMA CH i Read DAR High Register	0007318$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3187
PCIE0_USP_DMA_LLP_LOW_OFF_RDCH_i (i=0-3)	DMA CH i Read Linked List Pointer Low Register	000731C$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3187
PCIE0_USP_DMA_LLP_HIGH_OFF_RDCH_i (i=0-3)	DMA CH i Read Linked List Pointer High Register	0007320$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3188
PCIE0_USP_DMA CAP_CHi_RDCHN_RD_INT_CLEAR (i=0-3)	Interrupt clear for DMA read channel i	0007324$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3188
PCIE0_USP_DMA CAP_CHi_RDCHN_RD_INT_STATUS (i=0-3)	Interrupt status for DMA read channel i	0007328$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3189
PCIE0_USP_DMA CAP_CHi_RDCHN_RD_ERR_STATUS (i=0-3)	Error status for DMA read channel i	000732C$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3189

(table continues...)

Table 921 (continued) Register overview - PCIE0_USP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE0_USP_DMA CAP_CHi_RDCHN _RD_ERR_STATU S_HIGH (i=0-3)	Higher error status for DMA read channel i	0007330 $_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3190
PCIE0_USP_DMA CAP_CHi_RDCHN _RD_DOORBELL (i=0-3)	Doorbell for DMA read channel i	0007334 $_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3191

19.8.4.5 Register overview - PCIE1_USP domain FPI (ascending offset address)

Table 922 Register overview - PCIE1_USP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE1_USP_CLC	Clock control register	0000000 $_H$	P	P, SV, E	Application Reset	2822
PCIE1_USP_OCS	OCDS control and status register	0000004 $_H$	P	SV, P	Debug Reset	2823
PCIE1_USP_ID	Module identification register	0000008 $_H$	P	BE	PowerOn Reset	2823
PCIE1_USP_RST _CTRLA	Reset control register A	000000C $_H$	P	P, SV, E	Application Reset	2824
PCIE1_USP_RST _CTRLB	Reset control register B	0000010 $_H$	P	P, SV, E	Application Reset	2825
PCIE1_USP_RST _STAT	Reset status register	0000014 $_H$	P	BE	Application Reset	2825
PCIE1_USP_PRO TE	PROT register endinit	0000020 $_H$	U	SV, PROT	Application Reset	2826
PCIE1_USP_PRO TSE	PROT register safe endinit	0000024 $_H$	U	SV, PROT	Application Reset	2828
PCIE1_USP_ACC EN_WRA	Write access enable register A	0000040 $_H$	U	SE, SV	Application Reset	2829
PCIE1_USP_ACC EN_WRB	Write access enable register B	0000044 $_H$	U	SE, SV	Application Reset	2830
PCIE1_USP_ACC EN_RDA	Read access enable register A	0000048 $_H$	U	SE, SV	Application Reset	2830
PCIE1_USP_ACC EN_RDB	Read access enable register B	000004C $_H$	U	SE, SV	Application Reset	2831
PCIE1_USP_ACC EN_VM	VM access enable register	0000050 $_H$	U	SE, SV	Application Reset	2831

(table continues...)

Table 922 (continued) Register overview - PCIE1_USP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE1_USP_ACCEN_PRS	PRS access enable register	0000054$_H$	U	SE, SV	Application Reset	2832
PCIE1_USP_DMA_ACCENy_WRA (y=0-3)	CHy Write access enable register A	0000060$_H$+y*20$_H$	U	SE, SV	Application Reset	2833
PCIE1_USP_DMA_ACCENy_WRB (y=0-3)	CHy Write access enable register B	0000064$_H$+y*20$_H$	U	SE, SV	Application Reset	2833
PCIE1_USP_DMA_ACCENy_RDA (y=0-3)	CHy Read access enable register A	0000068$_H$+y*20$_H$	U	SE, SV	Application Reset	2834
PCIE1_USP_DMA_ACCENy_RDB (y=0-3)	CHy Read access enable register B	000006C$_H$+y*20$_H$	U	SE, SV	Application Reset	2834
PCIE1_USP_DMA_ACCENy_VM (y=0-3)	CHy VM access enable register	0000070$_H$+y*20$_H$	U	SE, SV	Application Reset	2835
PCIE1_USP_DMA_ACCENy_PRS (y=0-3)	CHy PRS access enable register	0000074$_H$+y*20$_H$	U	SE, SV	Application Reset	2835
PCIE1_USP_UPCS_OVRD1	UPCS layer override register 1	0000300$_H$	P	P	Kernel Reset	2836
PCIE1_USP_UPCS_OVRD2	UPCS layer override register 2	0000304$_H$	P	P	Kernel Reset	2836
PCIE1_USP_UPCS_OVRD3	UPCS layer override register 3	0000308$_H$	P	P	Kernel Reset	2837
PCIE1_USP_UPCS_OVRD4	UPCS layer override register 4	000030C$_H$	P	P	Kernel Reset	2837
PCIE1_USP_UPCS_OVRD5	UPCS layer override register 5	0000310$_H$	P	P	Kernel Reset	2838
PCIE1_USP_UPCS_OVRD6	UPCS layer override register 6	0000314$_H$	P	P	Kernel Reset	2838
PCIE1_USP_UPCS_OVRD7	UPCS layer override register 7	0000318$_H$	P	P	Kernel Reset	2839
PCIE1_USP_UPCS_OVRD8	UPCS layer override register 8	000031C$_H$	P	P	Kernel Reset	2839
PCIE1_USP_UPCS_OVRD9	UPCS layer override register 9	0000320$_H$	P	P	Kernel Reset	2840
PCIE1_USP_UPCS_OVRD10	UPCS layer override register 10	0000324$_H$	P	P	Kernel Reset	2840
PCIE1_USP_UPCS_OVRD11	UPCS layer override register 11	0000328$_H$	P	P	Kernel Reset	2841

(table continues...)

Table 922 (continued) Register overview - PCIE1_USP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE1_USP_UPCS_OVRD12	UPCS layer override register 12	000032C H	P	P	Kernel Reset	2841
PCIE1_USP_UPCS_OVRD13	UPCS layer override register 13	0000330 H	P	P	Kernel Reset	2841
PCIE1_USP_UPCS_OVRD14	UPCS layer override register 14	0000334 H	P	P	Kernel Reset	2842
PCIE1_USP_UPCS_OVRD15	UPCS layer override register 15	0000338 H	P	P	Kernel Reset	2842
PCIE1_USP_UPCS_OVRD16	UPCS layer override register 16	000033C H	P	P	Kernel Reset	2843
PCIE1_USP_UPCS_OVRD17	UPCS layer override register 17	0000340 H	P	P	Kernel Reset	2843
PCIE1_USP_INT_MSICTRL	Request MSI interrupt, EP only.	0000348 H	P	P	Kernel Reset	2844
PCIE1_USP_INT_MSISTAT	Pending status of MSI interrupt request, EP only.	000034C H	P	P	Kernel Reset	2844
PCIE1_USP_INT_INTXCTRL	Trigger INTX interrupt, EP only.	0000350 H	P	P	Kernel Reset	2845
PCIE1_USP_INT_TRIGSTAT	Trigger status	0000358 H	P	P	Kernel Reset	2845
PCIE1_USP_INT_TRIGCLR	Interrupt clear, EP only.	000035C H	P	P	Kernel Reset	2846
PCIE1_USP_INT_TRIGEN	Interrupt enable EP only.	0000360 H	P	P	Kernel Reset	2846
PCIE1_USP_INT_REQTRIG	Receive incoming interrupt request, EP only.	0000364 H	P	P	Kernel Reset	2847
PCIE1_USP_CFG_INFO	Configuration information	0000368 H	P	P	Kernel Reset	2847
PCIE1_USP_CFG_BUSNUM	Bus/device number	000036C H	P	P	Kernel Reset	2849
PCIE1_USP_LNK_DVCTYP	Select device/port type	0000370 H	P	P, E	PowerOn Reset	2849
PCIE1_USP_LNK_STAT	Indicate link status	0000374 H	P	P	Kernel Reset	2850
PCIE1_USP_LNK_CORECTRL	Controller control	0000378 H	P	P	Kernel Reset	2852
PCIE1_USP_LNK_INTEN	Enable the interrupt requests to SRC_PCIELNK	000037C H	P	P	Kernel Reset	2854
PCIE1_USP_TXRX_RXCTRL	Rx control	0000380 H	P	P	Kernel Reset	2855
PCIE1_USP_TXRX_TXCTRL0	Tx control 0	0000384 H	P	P	Kernel Reset	2856

(table continues...)

Table 922 (continued) Register overview - PCIE1_USP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE1_USP_PAYLOADz (z=0-1)	Message payload z information	000038C$_H$+z*4	P	P	Kernel Reset	2857
PCIE1_USP_MSG_INFO	Message information	0000394$_H$	P	P	Kernel Reset	2857
PCIE1_USP_PM_CTRL	Power management control	0000398$_H$	P	P	Kernel Reset	2859
PCIE1_USP_PM_STAT0	Power management status 0	000039C$_H$	P	P	Kernel Reset	2861
PCIE1_USP_PM_STAT1	Power management status 1	00003A4$_H$	P	P	Kernel Reset	2863
PCIE1_USP_PM_INTCLR	Power management interrupt clear	00003A8$_H$	P	P	Kernel Reset	2863
PCIE1_USP_PM_INTEN	Power management interrupt enable	00003AC$_H$	P	P	Kernel Reset	2864
PCIE1_USP_ERR_STAT	Error status	00003B0$_H$	P	P	Kernel Reset	2865
PCIE1_USP_ERR_INTCLR	Error clear	00003B4$_H$	P	P	Kernel Reset	2865
PCIE1_USP_ERR_INTEN	Error enable	00003B8$_H$	P	P	Kernel Reset	2866
PCIE1_USP_DBG_STAT	Debug status	00003BC$_H$	P	P	Kernel Reset	2867
PCIE1_USP_DBG_DATA	Debug information	00003C0$_H$	P	P	Kernel Reset	2868
PCIE1_USP_LTR_GNRTN	LTR message	00003C4$_H$	P	P	Kernel Reset	2868
PCIE1_USP_LTR_CTRL	LTR control	00003CC$_H$	P	P	Kernel Reset	2868
PCIE1_USP_PTM_CTRL	PTM control	00003D0$_H$	P	P	Kernel Reset	2869
PCIE1_USP_LCLCLKz (z=0-1)	Local Clock z value	00003D4$_H$+z*4	P	P	Kernel Reset	2870
PCIE1_USP_CLKCORRz (z=0-1)	Clock z correction	00003DC$_H$+z*4	P	P	Kernel Reset	2870
PCIE1_USP_MSKCLKz (z=0-1)	PTM z External Master Time	00003E4$_H$+z*4	P	P	Kernel Reset	2871
PCIE1_USP_OBFFMSG	OBFF message control	00003F0$_H$	P	P	Kernel Reset	2872

(table continues...)

Table 922 (continued) Register overview - PCIE1_USP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE1_USP_OBFFWAKE	OBFF WAKE control	00003F4$_H$	P	P	Kernel Reset	2874
PCIE1_USP_CFGBD	PCIe configuration control	00003F8$_H$	P	P	Kernel Reset	2876
PCIE1_USP_DATAx (x=0-7)	ELBI x mailbox register	00003FC$_H$+x*4	P	P	Kernel Reset	2876
PCIE1_USP_LLCTRL	DMA link list control register	000041C$_H$	P	P	Kernel Reset	2877
PCIE1_USP_OTSS	PCIe OCDS Trigger Set Select	0000420$_H$	P	SV, P	Debug Reset	2877
PCIE1_USP_ALARM_STAT	Interrupt/alarm status	0000424$_H$	P	P	Kernel Reset	2878
PCIE1_USP_ALARM_INTCLR	Interrupt/alarm clear	0000428$_H$	P	P	Kernel Reset	2880
PCIE1_USP_ALARM_INTEN	Interrupt/alarm enable	000042C$_H$	P	P	Kernel Reset	2881
PCIE1_USP_IATU_REGION_CTRL_1_OFF_OUTBOUND_i (i=0-7)	iATU Region i Control 1 Register Outbound	0003000$_H$+i*200$_H$	P	P	Kernel Reset	2882
PCIE1_USP_IATU_REGION_CTRL_2_OFF_OUTBOUND_i (i=0-7)	iATU Region i Control 2 Register Outbound	0003004$_H$+i*200$_H$	P	P	Kernel Reset	2883
PCIE1_USP_IATU_LWR_BASE_ADDR_OFF_OUTBOUND_i (i=0-7)	iATU i Lower Base Address Register Outbound	0003008$_H$+i*200$_H$	P	P	Kernel Reset	2885
PCIE1_USP_IATU_UPPER_BASE_ADDR_OFF_OUTBOUND_i (i=0-7)	iATU i Upper Base Address Register Outbound	000300C$_H$+i*200$_H$	P	P	Kernel Reset	2886
PCIE1_USP_IATU_LIMIT_ADDR_OFF_OUTBOUND_i (i=0-7)	iATU i Limit Address Register Outbound	0003010$_H$+i*200$_H$	P	P	Kernel Reset	2886

(table continues...)

Table 922 (continued) Register overview - PCIE1_USP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE1_USP_IATU_LWR_TARGET_ADDR_OFF_OUTBOUND_i (i=0-7)	iATU i Lower Target Address Register Outbound	0003014$_H$+i*200$_H$	P	P	Kernel Reset	2887
PCIE1_USP_IATU_UPPER_TARGET_ADDR_OFF_OUTBOUND_i (i=0-7)	iATU i Upper Target Address Register Outbound	0003018$_H$+i*200$_H$	P	P	Kernel Reset	2887
PCIE1_USP_IATU_REGION_CTRL_1_OFF_INBOUND_i (i=0-7)	iATU Region i Control 1 Register Inbound	0003100$_H$+i*200$_H$	P	P	Kernel Reset	2888
PCIE1_USP_IATU_REGION_CTRL_2_OFF_INBOUND_i (i=0-7)	iATU Region i Control 2 Register Inbound	0003104$_H$+i*200$_H$	P	P	Kernel Reset	2889
PCIE1_USP_IATU_LWR_BASE_ADDR_OFF_INBOUND_i (i=0-7)	iATU i Lower Base Address Register Inbound	0003108$_H$+i*200$_H$	P	P	Kernel Reset	2893
PCIE1_USP_IATU_UPPER_BASE_ADDR_OFF_INBOUND_i (i=0-7)	iATU i Upper Base Address Register Inbound	000310C$_H$+i*200$_H$	P	P	Kernel Reset	2893
PCIE1_USP_IATU_LIMIT_ADDR_OFF_INBOUND_i (i=0-7)	iATU i Limit Address Register Inbound	0003110$_H$+i*200$_H$	P	P	Kernel Reset	2894
PCIE1_USP_IATU_LWR_TARGET_ADDR_OFF_INBOUND_i (i=0-7)	iATU i Lower Target Address Register Inbound	0003114$_H$+i*200$_H$	P	P	Kernel Reset	2895
PCIE1_USP_DEVICE_ID_VENDOR_ID_REG	Device ID and Vendor ID Register	0004000$_H$	P	P	Kernel Reset	2895
PCIE1_USP_STATUS_COMMAND_REG	Status and Command Register	0004004$_H$	P	P	Kernel Reset	2896

(table continues...)

Table 922 (continued) Register overview - PCIE1_USP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE1_USP_CLASS_CODE_REVISION_ID	Class Code and Revision ID Register	0004008H	P	P	Kernel Reset	2900
PCIE1_USP_BIST_HEADER_TYPE_LATENCY_CACHE_LINE_SIZE_REG	BIST, Header Type, Latency Timer, and Cache Line Size Register	000400CH	P	P	Kernel Reset	2901
PCIE1_USP_BARi_REG (i=0-5)	BAR0 Register i	0004010H+i*4	P	P	Kernel Reset	2902
PCIE1_USP_CARDBUS_CIS_PTR_REG	CardBus CIS Pointer Register	0004028H	P	P	Kernel Reset	2903
PCIE1_USP_SUBSYSTEM_ID_SUBSYSTEM_VENDOR_ID_REG	Subsystem ID and Subsystem Vendor ID Register	000402CH	P	P	Kernel Reset	2904
PCIE1_USP_PCI_CAP_PTR_REG	Capabilities Pointer Register	0004034H	P	P	Kernel Reset	2905
PCIE1_USP_MAX_LATENCY_MIN_GRANT_INTERRUPT_PIN_INTERRUPT_LINE_REG	Max_Lat, Min_Gnt, Interrupt Pin, and Interrupt Line Register	000403CH	P	P	Kernel Reset	2905
PCIE1_USP_CAP_ID_NXT_PTR_REG	Power Management Capabilities Register	0004040H	P	P	Kernel Reset	2906
PCIE1_USP_CON_STATUS_REG	Power Management Control and Status Register	0004044H	P	P	Kernel Reset	2909
PCIE1_USP_PCI_MSI_CAP_ID_NEXT_CTRL_REG	MSI Capability Header and Message Control Register	0004050H	P	P	Kernel Reset	2912
PCIE1_USP_MSI_CAP_OFF_04H_REG	Message Address Register for MSI (Offset 04h)	0004054H	P	P	Kernel Reset	2914
PCIE1_USP_MSI_CAP_OFF_08H_REG	Message Address Register for MSI (Offset 08h)	0004058H	P	P	Kernel Reset	2915
PCIE1_USP_MSI_CAP_OFF_0CH_REG	Message Address Register for MSI (Offset 0Ch)	000405CH	P	P	Kernel Reset	2916
PCIE1_USP_MSI_CAP_OFF_10H_REG	Message Address Register for MSI (Offset 10h)	0004060H	P	P	Kernel Reset	2917

(table continues...)

Table 922 (continued) Register overview - PCIE1_USP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE1_USP_MSI_CAP_OFF_14H_REG	Message Address Register for MSI (Offset 14h)	0004064H	P	P	Kernel Reset	2917
PCIE1_USP_PCIE_CAP_ID_PCIE_NEXT_CAP_PTR_PCIE_CAP_REG	PCI Express Capabilities, ID, Next Pointer Register	0004070H	P	P	Kernel Reset	2918
PCIE1_USP_DEVICE_CAPABILITIES_REG	Device Capabilities Register	0004074H	P	P	Kernel Reset	2920
PCIE1_USP_DEVICE_CONTROL_DEVICE_STATUS	Device Control and Device Status Register	0004078H	P	P	Kernel Reset	2924
PCIE1_USP_LINK_CAPABILITIES_REG	Link Capabilities Register	000407CH	P	P	Kernel Reset	2929
PCIE1_USP_LINK_CONTROL_LINK_STATUS_REG	Link Control and Link Status Register	0004080H	P	P	Kernel Reset	2934
PCIE1_USP_DEVICE_CAPABILITIES2_REG	Device Capabilities 2 Register	0004094H	P	P	Kernel Reset	2941
PCIE1_USP_DEVICE_CONTROL2_DEVICE_STATUS2_REG	Device Control 2 and Status 2 Register	0004098H	P	P	Kernel Reset	2945
PCIE1_USP_LINK_CAPABILITIES2_REG	Link Capabilities 2 Register	000409CH	P	P	Kernel Reset	2947
PCIE1_USP_LINK_CONTROL2_LINK_STATUS2_REG	Link Control 2 and Status 2 Register	00040A0H	P	P	Kernel Reset	2949
PCIE1_USP_PCI_MSIX_CAP_ID_NEXT_CTRL_REG	MSI-X Capability ID, Next Pointer, Control Register	00040B0H	P	P	Kernel Reset	2955
PCIE1_USP_MSIX_TABLE_OFFSET_REG	MSI-X Table Offset and BIR Register	00040B4H	P	P	Kernel Reset	2957
PCIE1_USP_MSIX_PBA_OFFSET_REG	MSI-X PBA Offset and BIR Register	00040B8H	P	P	Kernel Reset	2957
PCIE1_USP_AER_EXT_CAP_HDR_OFF	Advanced Error Reporting Extended Capability Header	0004100H	P	P	Kernel Reset	2958

(table continues...)

Table 922 (continued) Register overview - PCIE1_USP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE1_USP_UNCORR_ERR_STATUS_OFF	Uncorrectable Error Status Register	0004104H	P	P	Kernel Reset	2959
PCIE1_USP_UNCORR_ERR_MASK_OFF	Uncorrectable Error Mask Register	0004108H	P	P	Kernel Reset	2961
PCIE1_USP_UNCORR_ERR_SEV_OFF	Uncorrectable Error Severity Register	000410CH	P	P	Kernel Reset	2963
PCIE1_USP_CORR_ERR_STATUS_OFF	Correctable Error Status Register	0004110H	P	P	Kernel Reset	2965
PCIE1_USP_CORR_ERR_MASK_OFF	Correctable Error Mask Register	0004114H	P	P	Kernel Reset	2966
PCIE1_USP_ADV_ERR_CAP_CTRL_OFF	Advanced Error Capabilities and Control Register	0004118H	P	P	Kernel Reset	2967
PCIE1_USP_HDR_LOG_i_OFF (i=0-3)	Header i Log Register 0	000411CH+i*4	P	P	Kernel Reset	2969
PCIE1_USP_TLP_PREFIX_LOG_i_OFF (i=0-3)	TLP Prefix i Log Register 1	0004138H+i*4	P	P	Kernel Reset	2969
PCIE1_USP_SN_BASE	Device Serial Number Extended Capability Header	0004148H	P	P	Kernel Reset	2970
PCIE1_USP_SER_NUM_REG_DW_1	Serial Number 1 Register	000414CH	P	P	Kernel Reset	2971
PCIE1_USP_SER_NUM_REG_DW_2	Serial Number 2 Register	0004150H	P	P	Kernel Reset	2971
PCIE1_USP_SPCIE_CAP_HEADER_REG	SPCIE Capability Header	0004158H	P	P	Kernel Reset	2972
PCIE1_USP_LINK_CONTROL3_REG	Link Control 3 Register	000415CH	P	P	Kernel Reset	2973
PCIE1_USP_LANE_ERR_STATUS_REG	Lane Error Status Register	0004160H	P	P	Kernel Reset	2974

(table continues...)

Table 922 (continued) Register overview - PCIE1_USP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE1_USP_SPCIE_CAP_OFF_0CH_REG	Lane Equalization Control Register for lanes 1 and 0	0004164H	P	P	Kernel Reset	2975
PCIE1_USP_LTR_CAP_HDR_REG	LTR Extended Capability Header	0004168H	P	P	Kernel Reset	2976
PCIE1_USP_LTR_LATENCY_REG	LTR Max Snoop and No-Snoop Latency Register	000416CH	P	P	Kernel Reset	2977
PCIE1_USP_L1SUB_CAP_HEADER_REG	L1 Substates Extended Capability Header	0004170H	P	P	Kernel Reset	2978
PCIE1_USP_L1SUB_CAPABILITY_REG	L1 Substates Capability Register	0004174H	P	P	Kernel Reset	2979
PCIE1_USP_L1SUB_CONTROL1_REG	L1 Substates Control 1 Register	0004178H	P	P	Kernel Reset	2981
PCIE1_USP_L1SUB_CONTROL2_REG	L1 Substates Control 2 Register	000417CH	P	P	Kernel Reset	2982
PCIE1_USP_RAS_DES_CAP_HEADER_REG	Vendor-Specific Extended Capability Header	0004180H	P	P	Kernel Reset	2984
PCIE1_USP_VENDOR_SPECIFIC_HEADER_REG	Vendor-Specific Header	0004184H	P	P	Kernel Reset	2985
PCIE1_USP_EVENT_COUNTER_CONTROL_REG	Event Counter Control	0004188H	P	P	Kernel Reset	2985
PCIE1_USP_EVENT_COUNTER_DATA_REG	Event Counter Data	000418CH	P	P	Kernel Reset	2987
PCIE1_USP_TIME_BASED_ANALYSIS_CONTROL_REG	Time-based Analysis Control	0004190H	P	P	Kernel Reset	2988
PCIE1_USP_TIME_BASED_ANALYSIS_DATA_REG	Time-based Analysis Data	0004194H	P	P	Kernel Reset	2989
PCIE1_USP_TIME_BASED_ANALYSIS_DATA_63_32_REG	Upper 32 bits of Time-based Analysis Data	0004198H	P	P	Kernel Reset	2990
PCIE1_USP_EINJ_ENABLE_REG	Error Injection Enable	00041B0H	P	P	Kernel Reset	2990

(table continues...)

Table 922 (continued) Register overview - PCIE1_USP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE1_USP_EINJ0_CRC_REG	Error Injection Control 0 (CRC Error)	00041B4$_H$	P	P	Kernel Reset	2992
PCIE1_USP_EINJ1_SEQNUM_REG	Error Injection Control 1 (Sequence Number Error). 	00041B8$_H$	P	P	Kernel Reset	2993
PCIE1_USP_EINJ2_DLLP_REG	Error Injection Control 2 (DLLP Error)	00041BC$_H$	P	P	Kernel Reset	2994
PCIE1_USP_EINJ3_SYMBOL_REG	Error Injection Control 3 (Symbol Error)	00041C0$_H$	P	P	Kernel Reset	2995
PCIE1_USP_EINJ4_FC_REG	Error Injection Control 4 (FC Credit Error)	00041C4$_H$	P	P	Kernel Reset	2996
PCIE1_USP_EINJ5_SP_TLP_REG	Error Injection Control 5 (Specific TLP Error)	00041C8$_H$	P	P	Kernel Reset	2997
PCIE1_USP_EINJ6_COMPARE_POINT_Hi_REG (i=0-3)	Error Injection Control 6 (Compare Point Header DWORD #0) i	00041CC$_H$+i*4	P	P	Kernel Reset	2998
PCIE1_USP_EINJ6_COMPARE_VALUE_Hi_REG (i=0-3)	Error Injection Control 6 (Compare Value Header DWORD #0) i	00041DC$_H$+i*4	P	P	Kernel Reset	2999
PCIE1_USP_EINJ6_CHANGE_POINT_Hi_REG (i=0-3)	Error Injection Control 6 (Change Point Header DWORD #0) i	00041EC$_H$+i*4	P	P	Kernel Reset	2999
PCIE1_USP_EINJ6_CHANGE_VALUE_Hi_REG (i=0-3)	Error Injection Control 6 (Change Value Header DWORD #0) i	00041FC$_H$+i*4	P	P	Kernel Reset	3000
PCIE1_USP_EINJ6_TLP_REG	Error Injection Control 6 (Packet Error)	000420C$_H$	P	P	Kernel Reset	3001
PCIE1_USP_SD_CONTROL1_REG	Silicon Debug Control 1	0004220$_H$	P	P	Kernel Reset	3002
PCIE1_USP_SD_CONTROL2_REG	Silicon Debug Control 2	0004224$_H$	P	P	Kernel Reset	3003
PCIE1_USP_SD_STATUS_L1LANE_REG	Silicon Debug Status(Layer1 Per-lane)	0004230$_H$	P	P	Kernel Reset	3004
PCIE1_USP_SD_STATUS_L1LTSSM_REG	Silicon Debug Status(Layer1 LTSSM)	0004234$_H$	P	P	Kernel Reset	3005
PCIE1_USP_SD_STATUS_PM_REG	Silicon Debug Status(PM)	0004238$_H$	P	P	Kernel Reset	3008

(table continues...)

Table 922 (continued) Register overview - PCIE1_USP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE1_USP_SD_STATUS_L2_REG	Silicon Debug Status(Layer2)	000423C H	P	P	Kernel Reset	3011
PCIE1_USP_SD_STATUS_L3FC_REG	Silicon Debug Status(Layer3 FC)	0004240 H	P	P	Kernel Reset	3011
PCIE1_USP_SD_STATUS_L3_REG	Silicon Debug Status(Layer3)	0004244 H	P	P	Kernel Reset	3013
PCIE1_USP_SD_EQ_CONTROL1_REG	Silicon Debug EQ Control 1	0004250 H	P	P	Kernel Reset	3014
PCIE1_USP_SD_EQ_CONTROL2_REG	Silicon Debug EQ Control 2	0004254 H	P	P	Kernel Reset	3016
PCIE1_USP_SD_EQ_CONTROL3_REG	Silicon Debug EQ Control 3	0004258 H	P	P	Kernel Reset	3017
PCIE1_USP_SD_EQ_STATUS1_REG	Silicon Debug EQ Status 1	0004260 H	P	P	Kernel Reset	3018
PCIE1_USP_SD_EQ_STATUS2_REG	Silicon Debug EQ Status 2	0004264 H	P	P	Kernel Reset	3019
PCIE1_USP_SD_EQ_STATUS3_REG	Silicon Debug EQ Status 3	0004268 H	P	P	Kernel Reset	3020
PCIE1_USP_RASDP_EXT_CAP_HDR_OFF	PCIe Extended capability ID, Capability version and Next capability offset	0004280 H	P	P	Kernel Reset	3021
PCIE1_USP_RASDP_VENDOR_SPECIFIC_HDR_OFF	Vendor Specific Header	0004284 H	P	P	Kernel Reset	3022
PCIE1_USP_RASDP_ERROR_PROT_CTRL_OFF	ECC Error Correction and Control Register	0004288 H	P	P	Kernel Reset	3023
PCIE1_USP_RASDP_ERROR_MODE_EN_OFF	RASDP error mode enable.	00042A8 H	P	P	Kernel Reset	3035
PCIE1_USP_RASDP_ERROR_MODE_CLEAR_OFF	RASDP Error Mode Clear Register	00042AC H	P	P	Kernel Reset	3036
PCIE1_USP_RASDP_RAM_ADDR_CORR_ERROR_OFF	RAM Address where a corrected error (1-bit ECC) is detected	00042B0 H	P	P	Kernel Reset	3037

(table continues...)

Table 922 (continued) Register overview - PCIE1_USP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE1_USP_RASDP_RAM_ADDR_UNCORR_ERROR_OFF	RAM Address where an uncorrected error (2-bit ECC) is detected	00042B4H	P	P	Kernel Reset	3037
PCIE1_USP_PTM_EXT_CAP_HDR_OFF	Precision Time Measurement Capability Header	00042B8H	P	P	Kernel Reset	3038
PCIE1_USP_PTM_CAP_OFF	PTM Capability Register	00042BCH	P	P	Kernel Reset	3039
PCIE1_USP_PTM_CONTROL_OFF	PTM Control Register	00042C0H	P	P	Kernel Reset	3040
PCIE1_USP_PTM_REQ_CAP_HDR_OFF	Precision Time Measurement Requester Capability Header (VSEC)	00042C4H	P	P	Kernel Reset	3041
PCIE1_USP_PTM_REQ_HDR_OFF	Precision Time Measurement Requester Vendor Specific Header	00042C8H	P	P	Kernel Reset	3042
PCIE1_USP_PTM_REQ_CONTROL_OFF	PTM Requester Vendor Specific Control Register	00042CCH	P	P	Kernel Reset	3043
PCIE1_USP_PTM_REQ_STATUS_OFF	PTM Requester Vendor Specific Status Register	00042D0H	P	P	Kernel Reset	3044
PCIE1_USP_PTM_REQ_LOCAL_LSB_OFF	PTM Requester Local Clock LSB	00042D4H	P	P	Kernel Reset	3045
PCIE1_USP_PTM_REQ_LOCAL_MSB_OFF	PTM Requester Local Clock MSB	00042D8H	P	P	Kernel Reset	3046
PCIE1_USP_PTM_REQ_T1_LSB_OFF	PTM Requester T1 Timestamp LSB	00042DCH	P	P	Kernel Reset	3046
PCIE1_USP_PTM_REQ_T1_MSB_OFF	PTM Requester T1 Timestamp MSB	00042E0H	P	P	Kernel Reset	3047
PCIE1_USP_PTM_REQ_T1P_LSB_OFF	PTM Requester T1 Previous Timestamp LSB	00042E4H	P	P	Kernel Reset	3047
PCIE1_USP_PTM_REQ_T1P_MSB_OFF	PTM Requester T1 Previous Timestamp MSB	00042E8H	P	P	Kernel Reset	3047
PCIE1_USP_PTM_REQ_T4_LSB_OFF	PTM Requester T4 Timestamp LSB	00042ECH	P	P	Kernel Reset	3048

(table continues...)

Table 922 (continued) Register overview - PCIE1_USP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE1_USP_PTM_REQ_T4_MSB_OFF	PTM Requester T4 Timestamp MSB	00042F0H	P	P	Kernel Reset	3048
PCIE1_USP_PTM_REQ_T4P_LSB_OFF	PTM Requester T4 Previous Timestamp LSB	00042F4H	P	P	Kernel Reset	3049
PCIE1_USP_PTM_REQ_T4P_MSB_OFF	PTM Requester T4 Previous Timestamp MSB	00042F8H	P	P	Kernel Reset	3049
PCIE1_USP_PTM_REQ_MASTER_LSB_OFF	PTM Requester Master Time LSB	00042FCH	P	P	Kernel Reset	3050
PCIE1_USP_PTM_REQ_MASTER_MSB_OFF	PTM Requester Master Time MSB	0004300H	P	P	Kernel Reset	3050
PCIE1_USP_PTM_REQ_PROP_DELAY_OFF	PTM Requester Propagation Delay	0004304H	P	P	Kernel Reset	3051
PCIE1_USP_PTM_REQ_MASTERT1_LSB_OFF	PTM Requester Master Time at T1 LSB	0004308H	P	P	Kernel Reset	3051
PCIE1_USP_PTM_REQ_MASTERT1_MSB_OFF	PTM Requester Master Time at T1 MSB	000430CH	P	P	Kernel Reset	3052
PCIE1_USP_PTM_REQ_TX_LATENCY_OFF	PTM Requester TX Latency	0004310H	P	P	Kernel Reset	3052
PCIE1_USP_PTM_REQ_RX_LATENCY_OFF	PTM Requester RX Latency	0004314H	P	P	Kernel Reset	3053
PCIE1_USP_PTM_REQ_CLOCK_CORR_LSB_OFF	PTM Requester Clock Correction LSB	0004318H	P	P	Kernel Reset	3054
PCIE1_USP_PTM_REQ_CLOCK_CORR_MSB_OFF	PTM Requester Clock Correction MSB	000431CH	P	P	Kernel Reset	3054
PCIE1_USP_PTM_REQ_NOM_CLOCK_T_OFF	PTM Requester Nominal Clock Period	0004320H	P	P	Kernel Reset	3055
PCIE1_USP_PTM_REQ_SCALED_CLOCK_T_OFF	PTM Requester Scaled Clock Period	0004324H	P	P	Kernel Reset	3055
PCIE1_USP_PTM_REQ_LATENCY_REG_SEL_OFF	PTM Requester Latency Viewport Register Select	0004328H	P	P	Kernel Reset	3056

(table continues...)

Table 922 (continued) Register overview - PCIE1_USP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE1_USP_VSECDMA_EXT_CAP_HDR_OFF	PCIe Extended Capability ID, Capability Version, and Next Capability Offset Register	000432C H	P	P	Kernel Reset	3057
PCIE1_USP_VSECDMA_VENDOR_SPECIFIC_HDR_OFF	Vendor Specific Header Register	0004330 H	P	P	Kernel Reset	3058
PCIE1_USP_VSECDMA_DEVICE_INFORMATION_OFF	DMA and related AXI Bridge Implementation Information	0004334 H	P	P	Kernel Reset	3059
PCIE1_USP_VSECDMA_NUM_CHAN_OFF	Number of Implemented Channels Register	0004338 H	P	P	Kernel Reset	3060
PCIE1_USP_VSECDMA_UNROLL_ADDR_OFFSET_LOW_OFF	DMA Register Map Start Address Offset Low Register	000433C H	P	P	Kernel Reset	3061
PCIE1_USP_VSECDMA_UNROLL_ADDR_OFFSET_HIGH_OFF	DMA Register Map Start Address Offset High Register	0004340 H	P	P	Kernel Reset	3061
PCIE1_USP_ACK_LATENCY_TIMER_OFF	Ack Latency Timer and Replay Timer Register	0004700 H	P	P	Kernel Reset	3062
PCIE1_USP_VENDOR_SPEC_DLLP_OFF	Vendor Specific DLLP Register	0004704 H	P	P	Kernel Reset	3063
PCIE1_USP_PORT_FORCE_OFF	Port Force Link Register	0004708 H	P	P	Kernel Reset	3063
PCIE1_USP_ACK_F_ASPM_CTRL_OFF	Ack Frequency and L0-L1 ASPM Control Register	000470C H	P	P	Kernel Reset	3065
PCIE1_USP_PORT_LINK_CTRL_OFF	Port Link Control Register	0004710 H	P	P	Kernel Reset	3067
PCIE1_USP_LANE_SKEW_OFF	Lane Skew Register	0004714 H	P	P	Kernel Reset	3069
PCIE1_USP_TIMER_CTRL_MAX_FUNC_NUM_OFF	Timer Control and Max Function Number Register	0004718 H	P	P	Kernel Reset	3071
PCIE1_USP_SYMBOL_TIMER_FILTER_1_OFF	Symbol Timer Register and Filter Mask 1 Register	000471C H	P	P	Kernel Reset	3072

(table continues...)

Table 922 (continued) Register overview - PCIE1_USP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE1_USP_FILTER_MASK_2_OFF	Filter Mask 2 Register	0004720H	P	P	Kernel Reset	3075
PCIE1_USP_PL_DEBUG0_OFF	Debug Register 0	0004728H	P	P	Kernel Reset	3077
PCIE1_USP_PL_DEBUG1_OFF	Debug Register 1	000472CH	P	P	Kernel Reset	3078
PCIE1_USP_TX_P_FC_CREDIT_STATUS_OFF	Transmit Posted FC Credit Status	0004730H	P	P	Kernel Reset	3079
PCIE1_USP_TX_NP_FC_CREDIT_STATUS_OFF	Transmit Non-Posted FC Credit Status	0004734H	P	P	Kernel Reset	3080
PCIE1_USP_TX_CPL_FC_CREDIT_STATUS_OFF	Transmit Completion FC Credit Status	0004738H	P	P	Kernel Reset	3081
PCIE1_USP_QUEUE_STATUS_OFF	Queue Status	000473CH	P	P	Kernel Reset	3082
PCIE1_USP_VC0_P_RX_Q_CTRL_OFF	Segmented-Buffer VC0 Posted Receive Queue Control	0004748H	P	P	Kernel Reset	3084
PCIE1_USP_VC0_NP_RX_Q_CTRL_OFF	Segmented-Buffer VC0 Non-Posted Receive Queue Control	000474CH	P	P	Kernel Reset	3086
PCIE1_USP_VC0_CPL_RX_Q_CTRL_OFF	Segmented-Buffer VC0 Completion Receive Queue Control	0004750H	P	P	Kernel Reset	3087
PCIE1_USP_GEN2_CTRL_OFF	Link Width and Speed Change Control Register	000480CH	P	P	Kernel Reset	3088
PCIE1_USP_CLOCK_GATING_CTRL_OFF	Clock Gating Control Register	000488CH	P	P	Kernel Reset	3092
PCIE1_USP_GEN3_RELATED_OFF	Gen3 Control Register	0004890H	P	P	Kernel Reset	3093
PCIE1_USP_GEN3_EQ_CONTROL_OFF	Gen3 EQ Control Register	00048A8H	P	P	Kernel Reset	3096
PCIE1_USP_GEN3_EQ_FB_MODE_DIR_CHANGE_OFF	Gen3 EQ Direction Change Feedback Mode Control Register	00048ACH	P	P	Kernel Reset	3099
PCIE1_USP_ORDER_RULE_CTRL_OFF	Order Rule Control Register	00048B4H	P	P	Kernel Reset	3100

(table continues...)

Table 922 (continued) Register overview - PCIE1_USP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE1_USP_PIPE_LOOPBACK_CONTROL_OFF	PIPE Loopback Control Register	00048B8H	P	P	Kernel Reset	3101
PCIE1_USP_MISC_CONTROL_1_OFF	DBI Read-Only Write Enable Register	00048BCH	P	P	Kernel Reset	3102
PCIE1_USP_LINK_FLUSH_CONTROL_OFF	Link Reset Request Flush Control Register	00048CCH	P	P	Kernel Reset	3108
PCIE1_USP_AMBA_ERROR_RESPONSE_DEFAULT_OFF	AXI Bridge Slave Error Response Register	00048D0H	P	P	Kernel Reset	3109
PCIE1_USP_AMBA_LINK_TIMEOUT_OFF	Link Down AXI Bridge Slave Timeout Register	00048D4H	P	P	Kernel Reset	3110
PCIE1_USP_AMBA_ORDERING_CTRL_OFF	AXI Bridge Ordering Control	00048D8H	P	P	Kernel Reset	3111
PCIE1_USP_PCIE_VERSION_NUMBER_OFF	PCIe Controller IIP Release Version Number	00048F8H	P	P	Kernel Reset	3115
PCIE1_USP_PCIE_VERSION_TYPE_OFF	PCIe Controller IIP Release Version Type	00048FCH	P	P	Kernel Reset	3116
PCIE1_USP_INTERFACE_TIMER_CONTROL_OFF	Interface Timer Control Register	0004930H	P	P	Kernel Reset	3116
PCIE1_USP_INTERFACE_TIMER_TARGET_OFF	Interface Timer Target Register	0004934H	P	P	Kernel Reset	3117
PCIE1_USP_INTERFACE_TIMER_STATUS_OFF	Interface Timer Status Register	0004938H	P	P	Kernel Reset	3118
PCIE1_USP_MSIX_ADDRESS_MATCH_LOW_OFF	MSI-X Address Match Low Register	0004940H	P	P	Kernel Reset	3119
PCIE1_USP_MSIX_ADDRESS_MATCH_HIGH_OFF	MSIX Address Match High Register	0004944H	P	P	Kernel Reset	3120
PCIE1_USP_MSIX_DOORBELL_OFF	MSI-X Doorbell Register	0004948H	P	P	Kernel Reset	3120
PCIE1_USP_SAFETY_MASK_OFF	Functional Safety Interrupt Events Mask Register	0004960H	P	P	Kernel Reset	3123

(table continues...)

Table 922 (continued) Register overview - PCIE1_USP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE1_USP_SAFETY_STATUS_OFF	Status for functional safety interrupt events	0004964$_H$	P	P	Kernel Reset	3124
PCIE1_USP_PL_APP_BUS_DEV_NUM_STATUS_OFF	Application driven bus and device number register	0004B10$_H$	P	P	Kernel Reset	3125
PCIE1_USP_PL_CHK_REG_CONTROL_STATUS_OFF	CDM Register Checking Control and Status Register	0004B20$_H$	P	P	Kernel Reset	3126
PCIE1_USP_PL_CHK_REG_START_END_OFF	CDM Register Checking First and Last address to check	0004B24$_H$	P	P	Kernel Reset	3128
PCIE1_USP_PL_CHK_REG_ERR_ADDR_OFF	CDM Register Checking Error Address	0004B28$_H$	P	P	Kernel Reset	3128
PCIE1_USP_PL_CHK_REG_ERR_PF_VF_OFF	CDM Register Checking error PF Numbers	0004B2C$_H$	P	P	Kernel Reset	3129
PCIE1_USP_PL_LTR_LATENCY_OFF	LTR Latency Register	0004B30$_H$	P	P	Kernel Reset	3129
PCIE1_USP_AUX_CLK_FREQ_OFF	Auxiliary Clock Frequency Control Register	0004B40$_H$	P	P	Kernel Reset	3130
PCIE1_USP_POWERDOWN_CTRL_STATUS_OFF	Powerdown Control and Status Register	0004B48$_H$	P	P	Kernel Reset	3132
PCIE1_USP_MSIXTABLEx_LADDR (x=0-63)	MSIX table x DWORD0	0005A00$_H$+x*10$_H$	P	P	Kernel Reset	3134
PCIE1_USP_MSIXTABLEx_UADDR (x=0-63)	MSIX table x DWORD1	0005A04$_H$+x*10$_H$	P	P	Kernel Reset	3134
PCIE1_USP_MSIXTABLEx_DATA (x=0-63)	MSIX table x DWORD2	0005A08$_H$+x*10$_H$	P	P	Kernel Reset	3135
PCIE1_USP_MSIXTABLEx_VECTRL (x=0-63)	MSIX table x DWORD3	0005A0C$_H$+x*10$_H$	P	P	Kernel Reset	3135
PCIE1_USP_MSIXPBAx_DWORD (x=0-1)	MSIX xtable DWORD	0005E10$_H$+x*4	P	P	Kernel Reset	3136

(table continues...)

Table 922 (continued) Register overview - PCIE1_USP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE1_USP_BARi_MASK_REG (i=0-5)	BAR i Mask Register	0006010$_H$+i*4	P	P	Kernel Reset	3136
PCIE1_USP_SHADOW_PCI_MSIX_CAP_ID_NEXT_CTRL_REG	MSI-X Capability ID, Next Pointer, Control Register	00060B0$_H$	P	P	Kernel Reset	3137
PCIE1_USP_DMA_CTRL_DATA_ARB_PRIOR_OFF	DMA Arbitration Scheme for TRGT1 Interface	0007000$_H$	P	P	Kernel Reset	3139
PCIE1_USP_DMA_CTRL_OFF	DMA Number of Channels Register	0007008$_H$	P	P	Kernel Reset	3140
PCIE1_USP_DMA_WRITE_ENGINE_EN_OFF	DMA Write Engine Enable Register	000700C$_H$	P	P	Kernel Reset	3141
PCIE1_USP_DMA_WRITE_DOORBELL_OFF	DMA Write Doorbell Register	0007010$_H$	P	P	Kernel Reset	3143
PCIE1_USP_DMA_WRITE_CHANNEL_ARB_WEIGHT_LOW_OFF	DMA Write Engine Channel Arbitration Weight Low Register	0007018$_H$	P	P	Kernel Reset	3144
PCIE1_USP_DMA_READ_ENGINE_EN_OFF	DMA Read Engine Enable Register	000702C$_H$	P	P	Kernel Reset	3146
PCIE1_USP_DMA_READ_DOORBELL_OFF	DMA Read Doorbell Register	0007030$_H$	P	P	Kernel Reset	3147
PCIE1_USP_DMA_READ_CHANNEL_ARB_WEIGHT_LOW_OFF	DMA Read Engine Channel Arbitration Weight Low Register	0007038$_H$	P	P	Kernel Reset	3148
PCIE1_USP_DMA_WRITE_INT_STATUS_OFF	DMA Write Interrupt Status Register	000704C$_H$	P	P	Kernel Reset	3150
PCIE1_USP_DMA_WRITE_INT_MASK_OFF	DMA Write Interrupt Mask Register	0007054$_H$	P	P	Kernel Reset	3151
PCIE1_USP_DMA_WRITE_INT_CLEAR_OFF	DMA Write Interrupt Clear Register	0007058$_H$	P	P	Kernel Reset	3152
PCIE1_USP_DMA_WRITE_ERR_STATUS_OFF	DMA Write Error Status Register	000705C$_H$	P	P	Kernel Reset	3153

(table continues...)

Table 922 (continued) Register overview - PCIE1_USP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE1_USP_DMA_WRITE_DONE_IMWR_LOW_OFF	DMA Write Done IMWr Address Low Register	0007060H	P	P	Kernel Reset	3153
PCIE1_USP_DMA_WRITE_DONE_IMWR_HIGH_OFF	DMA Write Done IMWr Interrupt Address High Register	0007064H	P	P	Kernel Reset	3154
PCIE1_USP_DMA_WRITE_ABORT_IMWR_LOW_OFF	DMA Write Abort IMWr Address Low Register	0007068H	P	P	Kernel Reset	3154
PCIE1_USP_DMA_WRITE_ABORT_IMWR_HIGH_OFF	DMA Write Abort IMWr Address High Register	000706CH	P	P	Kernel Reset	3155
PCIE1_USP_DMA_WRITE_CH01_IMWR_DATA_OFF	DMA Write Channel 1 and 0 IMWr Data Register	0007070H	P	P	Kernel Reset	3155
PCIE1_USP_DMA_WRITE_CH23_IMWR_DATA_OFF	DMA Write Channel 3 and 2 IMWr Data Register	0007074H	P	P	Kernel Reset	3156
PCIE1_USP_DMA_WRITE_LINKED_LIST_ERR_EN_OFF	DMA Write Linked List Error Enable Register	0007090H	P	P	Kernel Reset	3157
PCIE1_USP_DMA_READ_INT_STATUS_OFF	DMA Read Interrupt Status Register	00070A0H	P	P	Kernel Reset	3158
PCIE1_USP_DMA_READ_INT_MASK_OFF	DMA Read Interrupt Mask Register	00070A8H	P	P	Kernel Reset	3159
PCIE1_USP_DMA_READ_INT_CLEAR_OFF	DMA Read Interrupt Clear Register	00070ACH	P	P	Kernel Reset	3160
PCIE1_USP_DMA_READ_ERR_STATUS_LOW_OFF	DMA Read Error Status Low Register	00070B4H	P	P	Kernel Reset	3161
PCIE1_USP_DMA_READ_ERR_STATUS_HIGH_OFF	DMA Read Error Status High Register	00070B8H	P	P	Kernel Reset	3162
PCIE1_USP_DMA_READ_LINKED_LIST_ERR_EN_OFF	DMA Read Linked List Error Enable Register	00070C4H	P	P	Kernel Reset	3163
PCIE1_USP_DMA_READ_DONE_IMWR_LOW_OFF	DMA Read Done IMWr Address Low Register	00070CCH	P	P	Kernel Reset	3164

(table continues...)

Table 922 (continued) Register overview - PCIE1_USP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE1_USP_DMA_READ_DONE_IMWR_HIGH_OFF	DMA Read Done IMWr Address High Register	00070D0H	P	P	Kernel Reset	3164
PCIE1_USP_DMA_READ_ABORT_IMWR_LOW_OFF	DMA Read Abort IMWr Address Low Register	00070D4H	P	P	Kernel Reset	3165
PCIE1_USP_DMA_READ_ABORT_IMWR_HIGH_OFF	DMA Read Abort IMWr Address High Register	00070D8H	P	P	Kernel Reset	3165
PCIE1_USP_DMA_READ_CH01_IMWR_DATA_OFF	DMA Read Channel 1 and 0 IMWr Data Register	00070DCH	P	P	Kernel Reset	3166
PCIE1_USP_DMA_READ_CH23_IMWR_DATA_OFF	DMA Read Channel 3 and 2 IMWr Data Register	00070E0H	P	P	Kernel Reset	3166
PCIE1_USP_DMA_WRITE_ENGINE_HSHAKE_CNT_LOW_OFF	DMA Write Engine Handshake Counter Channel 0/1/2/3 Register	0007108H	P	P	Kernel Reset	3168
PCIE1_USP_DMA_READ_ENGINE_HSHAKE_CNT_LOW_OFF	DMA Read Engine Handshake Counter Channel 0/1/2/3 Register	0007118H	P	P	Kernel Reset	3169
PCIE1_USP_DMA_WRITE_CHi_PWR_EN_OFF (i=0-3)	DMA Write Channel i Power Enable Register	0007128H+i*4	P	P	Kernel Reset	3171
PCIE1_USP_DMA_READ_CHi_PWR_EN_OFF (i=0-3)	DMA Read Channel i Power Enable Register	0007168H+i*4	P	P	Kernel Reset	3171
PCIE1_USP_DMA_CH_CONTROL1_OFF_WRCH_i (i=0-3)	DMA Write Channel i Control 1 Register	0007200H+i*200H	P_DMAi	P_DMAi	Kernel Reset	3172
PCIE1_USP_DMA_TRANSFER_SIZE_OFF_WRCH_i (i=0-3)	DMA CH i Write Transfer Size Register	0007208H+i*200H	P_DMAi	P_DMAi	Kernel Reset	3175
PCIE1_USP_DMA_SAR_LOW_OFF_WRCH_i (i=0-3)	DMA CH i Write SAR Low Register	000720CH+i*200H	P_DMAi	P_DMAi	Kernel Reset	3175

(table continues...)

Table 922 (continued) Register overview - PCIE1_USP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE1_USP_DMA_SAR_HIGH_OFF_WRCH_i (i=0-3)	DMA CH i Write SAR High Register	0007210$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3176
PCIE1_USP_DMA_DAR_LOW_OFF_WRCH_i (i=0-3)	DMA CH i Write DAR Low Register	0007214$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3176
PCIE1_USP_DMA_DAR_HIGH_OFF_WRCH_i (i=0-3)	DMA CH i Write DAR High Register	0007218$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3177
PCIE1_USP_DMA_LLP_LOW_OFF_WRCH_i (i=0-3)	DMA CH i Write Linked List Pointer Low Register	000721C$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3177
PCIE1_USP_DMA_LLP_HIGH_OFF_WRCH_i (i=0-3)	DMA CH i Write Linked List Pointer High Register	0007220$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3178
PCIE1_USP_DMACAP_CHi_WRCHN_WR_INT_CLEAR (i=0-3)	Interrupt clear for DMA write channel i	0007224$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3178
PCIE1_USP_DMACAP_CHi_WRCHN_WR_INT_STATUS (i=0-3)	Interrupt status for DMA write channel i	0007228$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3179
PCIE1_USP_DMACAP_CHi_WRCHN_WR_ERR_STATUS (i=0-3)	Error status for DMA write channel i	000722C$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3180
PCIE1_USP_DMACAP_CHi_WRCHN_WR_DOORBELL (i=0-3)	Doorbell for DMA write channel i	0007230$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3181
PCIE1_USP_DMA_CH_CONTROL1_OFF_RDCH_i (i=0-3)	DMA Read Channel i Control 1 Register	0007300$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3181

(table continues...)

Table 922 (continued) Register overview - PCIE1_USP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE1_USP_DMA_TRANSFER_SIZE_OFF_RDCH_i (i=0-3)	DMA CH i Read Transfer Size Register	0007308$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3184
PCIE1_USP_DMA_SAR_LOW_OFF_RDCH_i (i=0-3)	DMA CH i Read SAR Low Register	000730C$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3185
PCIE1_USP_DMA_SAR_HIGH_OFF_RDCH_i (i=0-3)	DMA CH i Read SAR High Register	0007310$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3186
PCIE1_USP_DMA_DAR_LOW_OFF_RDCH_i (i=0-3)	DMA CH i Read DAR Low Register	0007314$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3186
PCIE1_USP_DMA_DAR_HIGH_OFF_RDCH_i (i=0-3)	DMA CH i Read DAR High Register	0007318$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3187
PCIE1_USP_DMA_LLP_LOW_OFF_RDCH_i (i=0-3)	DMA CH i Read Linked List Pointer Low Register	000731C$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3187
PCIE1_USP_DMA_LLP_HIGH_OFF_RDCH_i (i=0-3)	DMA CH i Read Linked List Pointer High Register	0007320$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3188
PCIE1_USP_DMA CAP_CHi_RDCHN_RD_INT_CLEAR (i=0-3)	Interrupt clear for DMA read channel i	0007324$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3188
PCIE1_USP_DMA CAP_CHi_RDCHN_RD_INT_STATUS (i=0-3)	Interrupt status for DMA read channel i	0007328$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3189
PCIE1_USP_DMA CAP_CHi_RDCHN_RD_ERR_STATUS (i=0-3)	Error status for DMA read channel i	000732C$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3189

(table continues...)

Table 922 (continued) Register overview - PCIE1_USP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE1_USP_DMACAP_CHi_RDCHN_RD_ERR_STATUS_HIGH (i=0-3)	Higher error status for DMA read channel i	0007330$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3190
PCIE1_USP_DMACAP_CHi_RDCHN_RD_DOORBELL (i=0-3)	Doorbell for DMA read channel i	0007334$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3191

19.8.4.6 Register overview - PCIE0_USP_SRI domain SRI (ascending offset address)

Table 923 Register overview - PCIE0_USP_SRI domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE0_USP_S2A_ACCENy_WRA (y=0-7)	Write access enable register A y	3FFFE00$_H$+y*20$_H$	U	SE, SV	Application Reset	3192
PCIE0_USP_S2A_ACCENy_WRB (y=0-7)	Write access enable register B y	3FFFE04$_H$+y*20$_H$	U	SE, SV	Application Reset	3193
PCIE0_USP_S2A_ACCENy_RDA (y=0-7)	Read access enable register A y	3FFFE08$_H$+y*20$_H$	U	SE, SV	Application Reset	3193
PCIE0_USP_S2A_ACCENy_RDB (y=0-7)	Read access enable register B y	3FFFE0C$_H$+y*20$_H$	U	SE, SV	Application Reset	3194
PCIE0_USP_S2A_ACCENy_VM (y=0-7)	VM access enable register y	3FFFE10$_H$+y*20$_H$	U	SE, SV	Application Reset	3194
PCIE0_USP_S2A_ACCENy_PRS (y=0-7)	PRS access enable register y	3FFFE14$_H$+y*20$_H$	U	SE, SV	Application Reset	3195
PCIE0_USP_S2A_ACCENy_RGNLA (y=0-7)	Region lower address register y	3FFFE18$_H$+y*20$_H$	U	SE, SV	Application Reset	3195
PCIE0_USP_S2A_ACCENy_RGNUA (y=0-7)	Region upper address register y	3FFFE1C$_H$+y*20$_H$	U	SE, SV	Application Reset	3196
PCIE0_USP_S2A_PROTSE	PROT Register Safe Endinit	3FFFF04$_H$	U	SV	Application Reset	3196

19.8.4.7 Register overview - PCIE1_USP_SRI domain SRI (ascending offset address)

Table 924 Register overview - PCIE1_USP_SRI domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE1_USP_S2A_ACCENy_WRA (y=0-7)	Write access enable register A y	3FFFE00$_H$+y*20$_H$	U	SE, SV	Application Reset	3192
PCIE1_USP_S2A_ACCENy_WRB (y=0-7)	Write access enable register B y	3FFFE04$_H$+y*20$_H$	U	SE, SV	Application Reset	3193
PCIE1_USP_S2A_ACCENy_RDA (y=0-7)	Read access enable register A y	3FFFE08$_H$+y*20$_H$	U	SE, SV	Application Reset	3193
PCIE1_USP_S2A_ACCENy_RDB (y=0-7)	Read access enable register B y	3FFFE0C$_H$+y*20$_H$	U	SE, SV	Application Reset	3194
PCIE1_USP_S2A_ACCENy_VM (y=0-7)	VM access enable register y	3FFFE10$_H$+y*20$_H$	U	SE, SV	Application Reset	3194
PCIE1_USP_S2A_ACCENy_PRS (y=0-7)	PRS access enable register y	3FFFE14$_H$+y*20$_H$	U	SE, SV	Application Reset	3195
PCIE1_USP_S2A_ACCENy_RGNLA (y=0-7)	Region lower address register y	3FFFE18$_H$+y*20$_H$	U	SE, SV	Application Reset	3195
PCIE1_USP_S2A_ACCENy_RGNUA (y=0-7)	Region upper address register y	3FFFE1C$_H$+y*20$_H$	U	SE, SV	Application Reset	3196
PCIE1_USP_S2A_PROTSE	PROT Register Safe Endinit	3FFFF04$_H$	U	SV	Application Reset	3196

19.8.4.8 Device specific registers

There are no device specific register changes.

19.8.4.9 Register address space - PCIE_DSP domain FPI

Table 925 Registers address space - PCIE_DSP domain FPI

Module	Domain	Base address	End address	Note
PCIE0_DSP	FPI	$F46A8000_H$	$F46AFFFF_H$	FPI Interface
PCIE1_DSP	FPI	$F46A0000_H$	$F46A7FFF_H$	FPI Interface

19.8.4.10 Register address space - PCIE_DSP domain SRI

Table 926 Registers address space - PCIE_DSP domain SRI

Module	Domain	Base address	End address	Note
PCIE0_DSP_SRI	SRI	$FC000000_H$	$FFFFFFFF_H$	SRI interface
PCIE1_DSP_SRI	SRI	$EC000000_H$	$EFFFFFFF_H$	SRI interface

19.8.4.11 Register overview - access mode glossary

Table 927 Register overview - access mode glossary

Keyword	Description
E	Access protection using PROT register PCIE0_DSP_PROTE or PCIE1_DSP_PROTE .
SE	Access protection using PROT register PCIE0_DSP_PROTSE or PCIE1_DSP_PROTSE or PCIE0_DSP_S2A_PROTSE or PCIE1_DSP_S2A_PROTSE .
APU-P	Protection group consisting of registers PCIE0_DSP_ACCEN_WRA , PCIE0_DSP_ACCEN_WRB , PCIE0_DSP_ACCEN_RDA , PCIE0_DSP_ACCEN_RDB , PCIE0_DSP_ACCEN_VM , PCIE0_DSP_ACCEN_PRS or PCIE1_DSP_ACCEN_WRA , PCIE1_DSP_ACCEN_WRB , PCIE1_DSP_ACCEN_RDA , PCIE1_DSP_ACCEN_RDB , PCIE1_DSP_ACCEN_VM , PCIE1_DSP_ACCEN_PRS .
P	Access protection using *APU-P* registers.
APU-Py (y=0-7)	Protection group consisting of registers PCIE0_DSP_S2A_ACCENy_WRA , PCIE0_DSP_S2A_ACCENy_WRB , PCIE0_DSP_S2A_ACCENy_RDA , PCIE0_DSP_S2A_ACCENy_RDB , PCIE0_DSP_S2A_ACCENy_VM , PCIE0_DSP_S2A_ACCENy_PRS , PCIE0_DSP_S2A_ACCENy_RGNLA , PCIE0_DSP_S2A_ACCENy_RGNUA or PCIE1_DSP_S2A_ACCENy_WRA , PCIE1_DSP_S2A_ACCENy_WRB , PCIE1_DSP_S2A_ACCENy_RDA , PCIE1_DSP_S2A_ACCENy_RDB , PCIE1_DSP_S2A_ACCENy_VM , PCIE1_DSP_S2A_ACCENy_PRS , PCIE1_DSP_S2A_ACCENy_RGNLA , PCIE1_DSP_S2A_ACCENy_RGNUA .
Py	Access protection using *APU-Py* registers.

(table continues...)

Table 927 (continued) Register overview - access mode glossary

Keyword	Description
APU-P_DMAy (y=0-3)	Protection group consisting of registers PCIE0_DSP_DMA_ACCENy_WRA , PCIE0_DSP_DMA_ACCENy_WRB , PCIE0_DSP_DMA_ACCENy_RDA , PCIE0_DSP_DMA_ACCENy_RDB , PCIE0_DSP_DMA_ACCENy_VM , PCIE0_DSP_DMA_ACCENy_PRS or PCIE1_DSP_DMA_ACCENy_WRA , PCIE1_DSP_DMA_ACCENy_WRB , PCIE1_DSP_DMA_ACCENy_RDA , PCIE1_DSP_DMA_ACCENy_RDB , PCIE1_DSP_DMA_ACCENy_VM , PCIE1_DSP_DMA_ACCENy_PRS .
P_DMAy	Access protection using *APU-P_DMAy* registers.
BE	Always returns a Bus Error.
U	No access restrictions.
PROT	Access restrictions as defined in the PROT register access rules.

19.8.4.12 Register overview - PCIE0_DSP domain FPI (ascending offset address)

Table 928 Register overview - PCIE0_DSP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE0_DSP_CLC	Clock control register	0000000H	P	P, SV, E	Application Reset	3218
PCIE0_DSP_OCS	OCDS control and status register	0000004H	P	SV, P	Debug Reset	3219
PCIE0_DSP_ID	Module identification register	0000008H	P	BE	PowerOn Reset	3219
PCIE0_DSP_RST_CTRLA	Reset control register A	000000CH	P	P, SV, E	Application Reset	3220
PCIE0_DSP_RST_CTRLB	Reset control register B	0000010H	P	P, SV, E	Application Reset	3220
PCIE0_DSP_RST_STAT	Reset status register	0000014H	P	BE	Application Reset	3221
PCIE0_DSP_PROTE	PROT register endinit	0000020H	U	SV, PROT	Application Reset	3222
PCIE0_DSP_PROTSE	PROT register safe endinit	0000024H	U	SV, PROT	Application Reset	3223
PCIE0_DSP_ACCEN_WRA	Write access enable register A	0000040H	U	SE, SV	Application Reset	3225
PCIE0_DSP_ACCEN_WRB	Write access enable register B	0000044H	U	SE, SV	Application Reset	3225
PCIE0_DSP_ACCEN_RDA	Read access enable register A	0000048H	U	SE, SV	Application Reset	3226
PCIE0_DSP_ACCEN_RDB	Read access enable register B	000004CH	U	SE, SV	Application Reset	3226

(table continues...)

Table 928 (continued) Register overview - PCIE0_DSP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE0_DSP_ACCEN_VM	VM access enable register	0000050$_H$	U	SE, SV	Application Reset	3227
PCIE0_DSP_ACCEN_PRS	PRS access enable register	0000054$_H$	U	SE, SV	Application Reset	3227
PCIE0_DSP_DMA_ACCENy_WRA (y=0-3)	CHy Write access enable register A	0000060$_H$+y*20$_H$	U	SE, SV	Application Reset	3228
PCIE0_DSP_DMA_ACCENy_WRB (y=0-3)	CHy Write access enable register B	0000064$_H$+y*20$_H$	U	SE, SV	Application Reset	3229
PCIE0_DSP_DMA_ACCENy_RDA (y=0-3)	CHy Read access enable register A	0000068$_H$+y*20$_H$	U	SE, SV	Application Reset	3229
PCIE0_DSP_DMA_ACCENy_RDB (y=0-3)	CHy Read access enable register B	000006C$_H$+y*20$_H$	U	SE, SV	Application Reset	3230
PCIE0_DSP_DMA_ACCENy_VM (y=0-3)	CHy VM access enable register	0000070$_H$+y*20$_H$	U	SE, SV	Application Reset	3230
PCIE0_DSP_DMA_ACCENy_PRS (y=0-3)	CHy PRS access enable register	0000074$_H$+y*20$_H$	U	SE, SV	Application Reset	3231
PCIE0_DSP_UPCS_OVRD1	UPCS layer override register 1	0000300$_H$	P	P	Kernel Reset	3231
PCIE0_DSP_UPCS_OVRD2	UPCS layer override register 2	0000304$_H$	P	P	Kernel Reset	3232
PCIE0_DSP_UPCS_OVRD3	UPCS layer override register 3	0000308$_H$	P	P	Kernel Reset	3232
PCIE0_DSP_UPCS_OVRD4	UPCS layer override register 4	000030C$_H$	P	P	Kernel Reset	3233
PCIE0_DSP_UPCS_OVRD5	UPCS layer override register 5	0000310$_H$	P	P	Kernel Reset	3233
PCIE0_DSP_UPCS_OVRD6	UPCS layer override register 6	0000314$_H$	P	P	Kernel Reset	3234
PCIE0_DSP_UPCS_OVRD7	UPCS layer override register 7	0000318$_H$	P	P	Kernel Reset	3234
PCIE0_DSP_UPCS_OVRD8	UPCS layer override register 8	000031C$_H$	P	P	Kernel Reset	3235
PCIE0_DSP_UPCS_OVRD9	UPCS layer override register 9	0000320$_H$	P	P	Kernel Reset	3235
PCIE0_DSP_UPCS_OVRD10	UPCS layer override register 10	0000324$_H$	P	P	Kernel Reset	3236

(table continues...)

Table 928 (continued) Register overview - PCIE0_DSP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE0_DSP_UPCS_OVRD11	UPCS layer override register 11	0000328H	P	P	Kernel Reset	3236
PCIE0_DSP_UPCS_OVRD12	UPCS layer override register 12	000032CH	P	P	Kernel Reset	3236
PCIE0_DSP_UPCS_OVRD13	UPCS layer override register 13	0000330H	P	P	Kernel Reset	3237
PCIE0_DSP_UPCS_OVRD14	UPCS layer override register 14	0000334H	P	P	Kernel Reset	3237
PCIE0_DSP_UPCS_OVRD15	UPCS layer override register 15	0000338H	P	P	Kernel Reset	3238
PCIE0_DSP_UPCS_OVRD16	UPCS layer override register 16	000033CH	P	P	Kernel Reset	3238
PCIE0_DSP_UPCS_OVRD17	UPCS layer override register 17	0000340H	P	P	Kernel Reset	3239
PCIE0_DSP_INT_MSICTRL	Request MSI interrupt, EP only.	0000348H	P	P	Kernel Reset	3239
PCIE0_DSP_INT_MSISTAT	Pending status of MSI interrupt request, EP only.	000034CH	P	P	Kernel Reset	3240
PCIE0_DSP_INT_INTXCTRL	Trigger INTX interrupt, EP only.	0000350H	P	P	Kernel Reset	3240
PCIE0_DSP_INT_TRIGSTAT	Trigger status	0000358H	P	P	Kernel Reset	3241
PCIE0_DSP_INT_TRIGCLR	Interrupt clear, EP only.	000035CH	P	P	Kernel Reset	3241
PCIE0_DSP_INT_TRIGEN	Interrupt enable EP only.	0000360H	P	P	Kernel Reset	3242
PCIE0_DSP_INT_REQTRIG	Receive incoming interrupt request, EP only.	0000364H	P	P	Kernel Reset	3242
PCIE0_DSP_CFG_INFO	Configuration information	0000368H	P	P	Kernel Reset	3243
PCIE0_DSP_CFG_BUSNUM	Bus/device number	000036CH	P	P	Kernel Reset	3244
PCIE0_DSP_LNK_DVCTYP	Select device/port type	0000370H	P	P, E	PowerOn Reset	3245
PCIE0_DSP_LNK_STAT	Indicate link status	0000374H	P	P	Kernel Reset	3246
PCIE0_DSP_LNK_CORECTRL	Controller control	0000378H	P	P	Kernel Reset	3247
PCIE0_DSP_LNK_INTEN	Enable the interrupt requests to SRC_PCIELNK	000037CH	P	P	Kernel Reset	3250
PCIE0_DSP_TXRX_RXCTRL	Rx control	0000380H	P	P	Kernel Reset	3250

(table continues...)

Table 928 (continued) Register overview - PCIE0_DSP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE0_DSP_TXRX_TXCTRL0	Tx control 0	0000384H	P	P	Kernel Reset	3251
PCIE0_DSP_PAYLOADz (z=0-1)	Message payload z information	000038CH+z*4	P	P	Kernel Reset	3252
PCIE0_DSP_MSG_INFO	Message information	0000394H	P	P	Kernel Reset	3253
PCIE0_DSP_PM_CTRL	Power management control	0000398H	P	P	Kernel Reset	3254
PCIE0_DSP_PM_STAT0	Power management status 0	000039CH	P	P	Kernel Reset	3256
PCIE0_DSP_PM_STAT1	Power management status 1	00003A4H	P	P	Kernel Reset	3258
PCIE0_DSP_PM_INTCLR	Power management interrupt clear	00003A8H	P	P	Kernel Reset	3258
PCIE0_DSP_PM_INTEN	Power management interrupt enable	00003ACH	P	P	Kernel Reset	3259
PCIE0_DSP_ERR_STAT	Error status	00003B0H	P	P	Kernel Reset	3260
PCIE0_DSP_ERR_INTCLR	Error clear	00003B4H	P	P	Kernel Reset	3260
PCIE0_DSP_ERR_INTEN	Error enable	00003B8H	P	P	Kernel Reset	3261
PCIE0_DSP_DBG_STAT	Debug status	00003BCH	P	P	Kernel Reset	3262
PCIE0_DSP_DBG_DATA	Debug information	00003C0H	P	P	Kernel Reset	3263
PCIE0_DSP_LTR_GNRTN	LTR message	00003C4H	P	P	Kernel Reset	3263
PCIE0_DSP_LTR_CTRL	LTR control	00003CCH	P	P	Kernel Reset	3263
PCIE0_DSP_PTM_CTRL	PTM control	00003D0H	P	P	Kernel Reset	3264
PCIE0_DSP_LCLCLKz (z=0-1)	Local Clock z value	00003D4H+z*4	P	P	Kernel Reset	3265
PCIE0_DSP_CLKCORRz (z=0-1)	Clock z correction	00003DCH+z*4	P	P	Kernel Reset	3265
PCIE0_DSP_MSKCLKz (z=0-1)	PTM z External Master Time	00003E4H+z*4	P	P	Kernel Reset	3266

(table continues...)

Table 928 (continued) Register overview - PCIE0_DSP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE0_DSP_OBFFMSG	OBFF message control	00003F0$_H$	P	P	Kernel Reset	3267
PCIE0_DSP_OBFFWAKE	OBFF WAKE control	00003F4$_H$	P	P	Kernel Reset	3269
PCIE0_DSP_CFGBD	PCIe configuration control	00003F8$_H$	P	P	Kernel Reset	3271
PCIE0_DSP_DATAx (x=0-7)	ELBI x mailbox register	00003FC$_H$+x*4	P	P	Kernel Reset	3271
PCIE0_DSP_LLCTRL	DMA link list control register	000041C$_H$	P	P	Kernel Reset	3272
PCIE0_DSP_OTSS	PCIe OCDS Trigger Set Select	0000420$_H$	P	SV, P	Debug Reset	3272
PCIE0_DSP_ALARM_STAT	Interrupt/alarm status	0000424$_H$	P	P	Kernel Reset	3273
PCIE0_DSP_ALARM_INTCLR	Interrupt/alarm clear	0000428$_H$	P	P	Kernel Reset	3275
PCIE0_DSP_ALARM_INTEN	Interrupt/alarm enable	000042C$_H$	P	P	Kernel Reset	3276
PCIE0_DSP_IATU_REGION_CTRL_1_OFF_OUTBOUND_i (i=0-7)	iATU Region i Control 1 Register Outbound	0003000$_H$+i*200$_H$	P	P	Kernel Reset	3277
PCIE0_DSP_IATU_REGION_CTRL_2_OFF_OUTBOUND_i (i=0-7)	iATU Region i Control 2 Register Outbound	0003004$_H$+i*200$_H$	P	P	Kernel Reset	3278
PCIE0_DSP_IATU_LWR_BASE_ADDR_OFF_OUTBOUND_i (i=0-7)	iATU i Lower Base Address Register Outbound	0003008$_H$+i*200$_H$	P	P	Kernel Reset	3280
PCIE0_DSP_IATU_UPPER_BASE_ADDR_OFF_OUTBOUND_i (i=0-7)	iATU i Upper Base Address Register Outbound	000300C$_H$+i*200$_H$	P	P	Kernel Reset	3281
PCIE0_DSP_IATU_LIMIT_ADDR_OFF_OUTBOUND_i (i=0-7)	iATU i Limit Address Register Outbound	0003010$_H$+i*200$_H$	P	P	Kernel Reset	3281

(table continues...)

Table 928 (continued) Register overview - PCIE0_DSP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE0_DSP_IATU_LWR_TARGET_ADDR_OFF_OUTBOUND_i (i=0-7)	iATU i Lower Target Address Register Outbound	0003014$_H$+i*200$_H$	P	P	Kernel Reset	3282
PCIE0_DSP_IATU_UPPER_TARGET_ADDR_OFF_OUTBOUND_i (i=0-7)	iATU i Upper Target Address Register Outbound	0003018$_H$+i*200$_H$	P	P	Kernel Reset	3282
PCIE0_DSP_IATU_REGION_CTRL_1_OFF_INBOUND_i (i=0-7)	iATU Region i Control 1 Register Inbound	0003100$_H$+i*200$_H$	P	P	Kernel Reset	3283
PCIE0_DSP_IATU_REGION_CTRL_2_OFF_INBOUND_i (i=0-7)	iATU Region i Control 2 Register Inbound	0003104$_H$+i*200$_H$	P	P	Kernel Reset	3284
PCIE0_DSP_IATU_LWR_BASE_ADDR_OFF_INBOUND_i (i=0-7)	iATU i Lower Base Address Register Inbound	0003108$_H$+i*200$_H$	P	P	Kernel Reset	3288
PCIE0_DSP_IATU_UPPER_BASE_ADDR_OFF_INBOUND_i (i=0-7)	iATU i Upper Base Address Register Inbound	000310C$_H$+i*200$_H$	P	P	Kernel Reset	3288
PCIE0_DSP_IATU_LIMIT_ADDR_OFF_INBOUND_i (i=0-7)	iATU i Limit Address Register Inbound	0003110$_H$+i*200$_H$	P	P	Kernel Reset	3289
PCIE0_DSP_IATU_LWR_TARGET_ADDR_OFF_INBOUND_i (i=0-7)	iATU i Lower Target Address Register Inbound	0003114$_H$+i*200$_H$	P	P	Kernel Reset	3290
PCIE0_DSP_TYPE1_DEV_ID_VEND_ID_REG	Device ID and Vendor ID Register	0004000$_H$	P	P	Kernel Reset	3290
PCIE0_DSP_TYPE1_STATUS_COMMAND_REG	Status and Command Register	0004004$_H$	P	P	Kernel Reset	3291

(table continues...)

Table 928 (continued) Register overview - PCIE0_DSP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE0_DSP_TYPE1_CLASS_CODE_REV_ID_REG	Class Code and Revision ID Register	0004008$_H$	P	P	Kernel Reset	3295
PCIE0_DSP_TYPE1_BIST_HDR_TYPE_LAT_CACHE_LINE_SIZE_REG	BIST, Header Type, Latency Timer, and Cache Line Size Register	000400C$_H$	P	P	Kernel Reset	3296
PCIE0_DSP_BARi_REG (i=0-1)	BAR0 Register i	0004010$_H$+i*4	P	P	Kernel Reset	3297
PCIE0_DSP_SEC_LAT_TIMER_SUB_BUS_SEC_BUS_PRI_BUS_REG	Secondary Latency Timer, Subordinate Bus Number, Secondary Bus Number, and Primary Bus Number Register	0004018$_H$	P	P	Kernel Reset	3299
PCIE0_DSP_SEC_STAT_IO_LIMIT_IO_BASE_REG	Secondary Status, and I/O Limit and Base Register	000401C$_H$	P	P	Kernel Reset	3300
PCIE0_DSP_MEM_LIMIT_MEM_BASE_REG	Memory Limit and Base Register	0004020$_H$	P	P	Kernel Reset	3303
PCIE0_DSP_PREF_MEM_LIMIT_PREF_MEM_BASE_REG	Prefetchable Memory Limit and Base Register	0004024$_H$	P	P	Kernel Reset	3304
PCIE0_DSP_PREF_BASE_UPPER_REG	Prefetchable Base Upper 32 Bits Register	0004028$_H$	P	P	Kernel Reset	3306
PCIE0_DSP_PREF_LIMIT_UPPER_REG	Prefetchable Limit Upper 32 Bits Register	000402C$_H$	P	P	Kernel Reset	3306
PCIE0_DSP_IO_LIMIT_UPPER_IO_BASE_UPPER_REG	I/O Limit and Base Upper 16 Bits Register	0004030$_H$	P	P	Kernel Reset	3307
PCIE0_DSP_TYPE1_CAP_PTR_REG	Capabilities Pointer Register	0004034$_H$	P	P	Kernel Reset	3308
PCIE0_DSP_TYPE1_EXP_ROM_BASE_REG	Expansion ROM BAR Register	0004038$_H$	P	P	Kernel Reset	
PCIE0_DSP_BRIDGE_CTRL_INT_PIN_INT_LINE_REG	Bridge Control, Interrupt Pin, and Interrupt Line Register	000403C$_H$	P	P	Kernel Reset	3309

(table continues...)

Table 928 (continued) Register overview - PCIE0_DSP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE0_DSP_CAP_ID_NXT_PTR_REG	Power Management Capabilities Register	0004040H	P	P	Kernel Reset	3312
PCIE0_DSP_CON_STATUS_REG	Power Management Control and Status Register	0004044H	P	P	Kernel Reset	3314
PCIE0_DSP_PCI_MSI_CAP_ID_NEXT_CTRL_REG	MSI Capability Header and Message Control Register	0004050H	P	P	Kernel Reset	3317
PCIE0_DSP_MSI_CAP_OFF_04H_REG	Message Address Register for MSI (Offset 04h)	0004054H	P	P	Kernel Reset	3319
PCIE0_DSP_MSI_CAP_OFF_08H_REG	Message Address Register for MSI (Offset 08h)	0004058H	P	P	Kernel Reset	3320
PCIE0_DSP_MSI_CAP_OFF_0CH_REG	Message Address Register for MSI (Offset 0Ch)	000405CH	P	P	Kernel Reset	3321
PCIE0_DSP_MSI_CAP_OFF_10H_REG	Message Address Register for MSI (Offset 10h)	0004060H	P	P	Kernel Reset	3322
PCIE0_DSP_MSI_CAP_OFF_14H_REG	Message Address Register for MSI (Offset 14h)	0004064H	P	P	Kernel Reset	3322
PCIE0_DSP_PCIE_CAP_ID_PCIE_NEXT_CAP_PTR_PCIE_CAP_REG	PCI Express Capabilities, ID, Next Pointer Register	0004070H	P	P	Kernel Reset	3323
PCIE0_DSP_DEVICE_CAPABILITIES_REG	Device Capabilities Register	0004074H	P	P	Kernel Reset	3325
PCIE0_DSP_DEVICE_CONTROL_DEVICE_STATUS	Device Control and Device Status Register	0004078H	P	P	Kernel Reset	3327
PCIE0_DSP_LINK_CAPABILITIES_REG	Link Capabilities Register	000407CH	P	P	Kernel Reset	3332
PCIE0_DSP_LINK_CONTROL_LINK_STATUS_REG	Link Control and Link Status Register	0004080H	P	P	Kernel Reset	3337
PCIE0_DSP_SLOT_CAPABILITIES_REG	Slot Capabilities Register	0004084H	P	P	Kernel Reset	3344
PCIE0_DSP_SLOT_CONTROL_SLOT_STATUS	Slot Control and Status Register	0004088H	P	P	Kernel Reset	3346

(table continues...)

Table 928 (continued) Register overview - PCIE0_DSP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE0_DSP_ROOT_CONTROL_ROOT_CAPABILITIES_REG	Root Control and Capabilities Register	000408C H	P	P	Kernel Reset	3351
PCIE0_DSP_ROOT_STATUS_REG	Root Status Register	0004090 H	P	P	Kernel Reset	3352
PCIE0_DSP_DEVICE_CAPABILITIES2_REG	Device Capabilities 2 Register	0004094 H	P	P	Kernel Reset	3353
PCIE0_DSP_DEVICE_CONTROL2_DEVICE_STATUS2_REG	Device Control 2 and Status 2 Register	0004098 H	P	P	Kernel Reset	3356
PCIE0_DSP_LINK_CAPABILITIES2_REG	Link Capabilities 2 Register	000409C H	P	P	Kernel Reset	3359
PCIE0_DSP_LINK_CONTROL2_LINK_STATUS2_REG	Link Control 2 and Status 2 Register	00040A0 H	P	P	Kernel Reset	3361
PCIE0_DSP_PCI_MSIX_CAP_ID_NEXT_CTRL_REG	MSI-X Capability ID, Next Pointer, Control Register	00040B0 H	P	P	Kernel Reset	3366
PCIE0_DSP_MSIX_TABLE_OFFSET_REG	MSI-X Table Offset and BIR Register	00040B4 H	P	P	Kernel Reset	3368
PCIE0_DSP_MSIX_PBA_OFFSET_REG	MSI-X PBA Offset and BIR Register	00040B8 H	P	P	Kernel Reset	3368
PCIE0_DSP_AER_EXT_CAP_HDR_OFF	Advanced Error Reporting Extended Capability Header	0004100 H	P	P	Kernel Reset	3369
PCIE0_DSP_UNCORR_ERR_STATUS_OFF	Uncorrectable Error Status Register	0004104 H	P	P	Kernel Reset	3370
PCIE0_DSP_UNCORR_ERR_MASK_OFF	Uncorrectable Error Mask Register	0004108 H	P	P	Kernel Reset	3372
PCIE0_DSP_UNCORR_ERR_SEV_OFF	Uncorrectable Error Severity Register	000410C H	P	P	Kernel Reset	3374
PCIE0_DSP_CORR_ERR_STATUS_OFF	Correctable Error Status Register	0004110 H	P	P	Kernel Reset	3376

(table continues...)

Table 928 (continued) Register overview - PCIE0_DSP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE0_DSP_CORR_ERR_MASK_OFF	Correctable Error Mask Register	0004114H	P	P	Kernel Reset	3377
PCIE0_DSP_ADV_ERR_CAP_CTRL_OFF	Advanced Error Capabilities and Control Register	0004118H	P	P	Kernel Reset	3378
PCIE0_DSP_HDR_LOG_i_OFF (i=0-3)	Header i Log Register 0	000411CH+i*4	P	P	Kernel Reset	3379
PCIE0_DSP_AERCAP_ROOT_ROOT_ERR_CMD_OFF	Root Error Command Register	000412CH	P	P	Kernel Reset	3380
PCIE0_DSP_AERCAP_ROOT_ROOT_ERR_STATUS_OFF	Root Error Status Register	0004130H	P	P	Kernel Reset	3381
PCIE0_DSP_AERCAP_ERR_SRC_ID_OFF	Error Source Identification Register	0004134H	P	P	Kernel Reset	3382
PCIE0_DSP_TLP_PREFIX_LOG_i_OFF (i=0-3)	TLP Prefix i Log Register 1	0004138H+i*4	P	P	Kernel Reset	3383
PCIE0_DSP_SN_BASE	Device Serial Number Extended Capability Header	0004148H	P	P	Kernel Reset	3384
PCIE0_DSP_SER_NUM_REG_DW_1	Serial Number 1 Register	000414CH	P	P	Kernel Reset	3385
PCIE0_DSP_SER_NUM_REG_DW_2	Serial Number 2 Register	0004150H	P	P	Kernel Reset	3385
PCIE0_DSP_SPCIE_CAP_HEADER_REG	SPCIE Capability Header	0004158H	P	P	Kernel Reset	3386
PCIE0_DSP_LINK_CONTROL3_REG	Link Control 3 Register	000415CH	P	P	Kernel Reset	3386
PCIE0_DSP_LANE_ERR_STATUS_REG	Lane Error Status Register	0004160H	P	P	Kernel Reset	3387
PCIE0_DSP_SPCIE_CAP_OFF_0CH_REG	Lane Equalization Control Register for lanes 1 and 0	0004164H	P	P	Kernel Reset	3388

(table continues...)

Table 928 (continued) Register overview - PCIE0_DSP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE0_DSP_L1SUB_CAP_HEADER_REG	L1 Substates Extended Capability Header	0004170H	P	P	Kernel Reset	3390
PCIE0_DSP_L1SUB_CAPABILITY_REG	L1 Substates Capability Register	0004174H	P	P	Kernel Reset	3391
PCIE0_DSP_L1SUB_CONTROL1_REG	L1 Substates Control 1 Register	0004178H	P	P	Kernel Reset	3392
PCIE0_DSP_L1SUB_CONTROL2_REG	L1 Substates Control 2 Register	000417CH	P	P	Kernel Reset	3394
PCIE0_DSP_RAS_DES_CAP_HEADER_REG	Vendor-Specific Extended Capability Header	0004180H	P	P	Kernel Reset	3395
PCIE0_DSP_VENDOR_SPECIFIC_HEADER_REG	Vendor-Specific Header	0004184H	P	P	Kernel Reset	3396
PCIE0_DSP_EVENT_COUNTER_CONTROL_REG	Event Counter Control	0004188H	P	P	Kernel Reset	3397
PCIE0_DSP_EVENT_COUNTER_DATA_REG	Event Counter Data	000418CH	P	P	Kernel Reset	3399
PCIE0_DSP_TIME_BASED_ANALYSIS_CONTROL_REG	Time-based Analysis Control	0004190H	P	P	Kernel Reset	3400
PCIE0_DSP_TIME_BASED_ANALYSIS_DATA_REG	Time-based Analysis Data	0004194H	P	P	Kernel Reset	3401
PCIE0_DSP_TIME_BASED_ANALYSIS_DATA_63_32_REG	Upper 32 bits of Time-based Analysis Data	0004198H	P	P	Kernel Reset	3402
PCIE0_DSP_EINJ_ENABLE_REG	Error Injection Enable	00041B0H	P	P	Kernel Reset	3402
PCIE0_DSP_EINJ0_CRC_REG	Error Injection Control 0 (CRC Error)	00041B4H	P	P	Kernel Reset	3404
PCIE0_DSP_EINJ1_SEQNUM_REG	Error Injection Control 1 (Sequence Number Error). 	00041B8H	P	P	Kernel Reset	3405
PCIE0_DSP_EINJ2_DLLP_REG	Error Injection Control 2 (DLLP Error)	00041BCH	P	P	Kernel Reset	3406

(table continues...)

Table 928 (continued) Register overview - PCIE0_DSP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE0_DSP_EINJ3_SYMBOL_REG	Error Injection Control 3 (Symbol Error)	00041C0$_H$	P	P	Kernel Reset	3407
PCIE0_DSP_EINJ4_FC_REG	Error Injection Control 4 (FC Credit Error)	00041C4$_H$	P	P	Kernel Reset	3408
PCIE0_DSP_EINJ5_SP_TLP_REG	Error Injection Control 5 (Specific TLP Error)	00041C8$_H$	P	P	Kernel Reset	3409
PCIE0_DSP_EINJ6_COMPARE_POINT_Hi_REG (i=0-3)	Error Injection Control 6 i (Compare Point Header DWORD #0)	00041CC$_H$+i*4	P	P	Kernel Reset	3410
PCIE0_DSP_EINJ6_COMPARE_VALUE_Hi_REG (i=0-3)	Error Injection Control 6 i (Compare Value Header DWORD #0)	00041DC$_H$+i*4	P	P	Kernel Reset	3411
PCIE0_DSP_EINJ6_CHANGE_POINT_Hi_REG (i=0-3)	Error Injection Control 6 i (Change Point Header DWORD #0)	00041EC$_H$+i*4	P	P	Kernel Reset	3411
PCIE0_DSP_EINJ6_CHANGE_VALUE_Hi_REG (i=0-3)	Error Injection Control 6 i (Change Value Header DWORD #0)	00041FC$_H$+i*4	P	P	Kernel Reset	3412
PCIE0_DSP_EINJ6_TLP_REG	Error Injection Control 6 (Packet Error)	000420C$_H$	P	P	Kernel Reset	3413
PCIE0_DSP_SD_CONTROL1_REG	Silicon Debug Control 1	0004220$_H$	P	P	Kernel Reset	3414
PCIE0_DSP_SD_CONTROL2_REG	Silicon Debug Control 2	0004224$_H$	P	P	Kernel Reset	3415
PCIE0_DSP_SD_STATUS_L1LANE_REG	Silicon Debug Status(Layer1 Per-lane)	0004230$_H$	P	P	Kernel Reset	3416
PCIE0_DSP_SD_STATUS_L1LTSSM_REG	Silicon Debug Status(Layer1 LTSSM)	0004234$_H$	P	P	Kernel Reset	3417
PCIE0_DSP_SD_STATUS_PM_REG	Silicon Debug Status(PM)	0004238$_H$	P	P	Kernel Reset	3420
PCIE0_DSP_SD_STATUS_L2_REG	Silicon Debug Status(Layer2)	000423C$_H$	P	P	Kernel Reset	3423
PCIE0_DSP_SD_STATUS_L3FC_REG	Silicon Debug Status(Layer3 FC)	0004240$_H$	P	P	Kernel Reset	3423
PCIE0_DSP_SD_STATUS_L3_REG	Silicon Debug Status(Layer3)	0004244$_H$	P	P	Kernel Reset	3425

(table continues...)

Table 928 (continued) Register overview - PCIE0_DSP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE0_DSP_SD_EQ_CONTROL1_REG	Silicon Debug EQ Control 1	0004250H	P	P	Kernel Reset	3426
PCIE0_DSP_SD_EQ_CONTROL2_REG	Silicon Debug EQ Control 2	0004254H	P	P	Kernel Reset	3428
PCIE0_DSP_SD_EQ_CONTROL3_REG	Silicon Debug EQ Control 3	0004258H	P	P	Kernel Reset	3429
PCIE0_DSP_SD_EQ_STATUS1_REG	Silicon Debug EQ Status 1	0004260H	P	P	Kernel Reset	3430
PCIE0_DSP_SD_EQ_STATUS2_REG	Silicon Debug EQ Status 2	0004264H	P	P	Kernel Reset	3431
PCIE0_DSP_SD_EQ_STATUS3_REG	Silicon Debug EQ Status 3	0004268H	P	P	Kernel Reset	3432
PCIE0_DSP_RASDP_EXT_CAP_HDR_OFF	PCIe Extended capability ID, Capability version and Next capability offset	0004280H	P	P	Kernel Reset	3433
PCIE0_DSP_RASDP_VENDOR_SPECIFIC_HDR_OFF	Vendor Specific Header	0004284H	P	P	Kernel Reset	3434
PCIE0_DSP_RASDP_ERROR_PROT_CTRL_OFF	ECC Error Correction and Control Register	0004288H	P	P	Kernel Reset	3435
PCIE0_DSP_RASDP_ERROR_MODE_EN_OFF	RASDP error mode enable.	00042A8H	P	P	Kernel Reset	3447
PCIE0_DSP_RASDP_ERROR_MODE_CLEAR_OFF	RASDP Error Mode Clear Register	00042ACH	P	P	Kernel Reset	3448
PCIE0_DSP_RASDP_RAM_ADDR_CORR_ERROR_OFF	RAM Address where a corrected error (1-bit ECC) is detected	00042B0H	P	P	Kernel Reset	3449
PCIE0_DSP_RASDP_RAM_ADDR_UNCORR_ERROR_OFF	RAM Address where an uncorrected error (2-bit ECC) is detected	00042B4H	P	P	Kernel Reset	3449
PCIE0_DSP_PTM_EXT_CAP_HDR_OFF	Precision Time Measurement Capability Header	00042B8H	P	P	Kernel Reset	3450

(table continues...)

Table 928 (continued) Register overview - PCIE0_DSP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE0_DSP_PTM_CAP_OFF	PTM Capability Register	00042BC H	P	P	Kernel Reset	3451
PCIE0_DSP_PTM_CONTROL_OFF	PTM Control Register	00042C0 H	P	P	Kernel Reset	3452
PCIE0_DSP_PTM_RES_CAP_HDR_OFF	Precision Time Measurement Responder Capability Header (VSEC)	00042C4 H	P	P	Kernel Reset	3453
PCIE0_DSP_PTM_RES_HDR_OFF	Precision Time Measurement Responder Vendor Specific Header	00042C8 H	P	P	Kernel Reset	3454
PCIE0_DSP_PTM_RES_CONTROL_OFF	PTM Responder Control Register	00042CC H	P	P	Kernel Reset	3455
PCIE0_DSP_PTM_RES_STATUS_OFF	PTM Responder Status Register	00042D0 H	P	P	Kernel Reset	3456
PCIE0_DSP_PTM_RES_LOCAL_LSB_OFF	PTM Responder Local Clock LSB	00042D4 H	P	P	Kernel Reset	3457
PCIE0_DSP_PTM_RES_LOCAL_MSB_OFF	PTM Responder Local Clock MSB	00042D8 H	P	P	Kernel Reset	3457
PCIE0_DSP_PTM_RES_T2_LSB_OFF	PTM Responder T2 Timestamp LSB	00042DC H	P	P	Kernel Reset	3458
PCIE0_DSP_PTM_RES_T2_MSB_OFF	PTM Responder T2 Timestamp MSB	00042E0 H	P	P	Kernel Reset	3458
PCIE0_DSP_PTM_RES_T2P_LSB_OFF	PTM Responder T2 Previous Timestamp LSB	00042E4 H	P	P	Kernel Reset	3458
PCIE0_DSP_PTM_RES_T2P_MSB_OFF	PTM Responder T2 Previous Timestamp MSB	00042E8 H	P	P	Kernel Reset	3459
PCIE0_DSP_PTM_RES_T3_LSB_OFF	PTM Responder T3 Timestamp LSB	00042EC H	P	P	Kernel Reset	3459
PCIE0_DSP_PTM_RES_T3_MSB_OFF	PTM Responder T3 Timestamp MSB	00042F0 H	P	P	Kernel Reset	3460
PCIE0_DSP_PTM_RES_T3P_LSB_OFF	PTM Responder T3 Previous Timestamp LSB	00042F4 H	P	P	Kernel Reset	3460

(table continues...)

Table 928 (continued) Register overview - PCIE0_DSP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE0_DSP_PTM_RES_T3P_MSB_OFF	PTM Responder T3 Previous Timestamp MSB	00042F8H	P	P	Kernel Reset	3461
PCIE0_DSP_PTM_RES_TX_LATENCY_OFF	PTM Responder TX Latency	00042FCH	P	P	Kernel Reset	3461
PCIE0_DSP_PTM_RES_RX_LATENCY_OFF	PTM Responder RX Latency	0004300H	P	P	Kernel Reset	3462
PCIE0_DSP_PTM_RES_NOM_CLOCK_T_OFF	PTM Responder Nominal Clock Period	0004304H	P	P	Kernel Reset	3463
PCIE0_DSP_PTM_RES_SCALED_CLOCK_T_OFF	PTM Responder Scaled Clock Period	0004308H	P	P	Kernel Reset	3463
PCIE0_DSP_PTM_RES_LATENCY_REG_SEL_OFF	PTM Responder Latency Viewport Register Select	000430CH	P	P	Kernel Reset	3464
PCIE0_DSP_VSECDMA_EXT_CAP_HDR_OFF	PCIe Extended Capability ID, Capability Version, and Next Capability Offset Register	000432CH	P	P	Kernel Reset	3465
PCIE0_DSP_VSECDMA_VENDOR_SPECIFIC_HDR_OFF	Vendor Specific Header Register	0004330H	P	P	Kernel Reset	3466
PCIE0_DSP_VSECDMA_DEVICE_INFORMATION_OFF	DMA and related AXI Bridge Implementation Information	0004334H	P	P	Kernel Reset	3467
PCIE0_DSP_VSECDMA_NUM_CHAN_OFF	Number of Implemented Channels Register	0004338H	P	P	Kernel Reset	3469
PCIE0_DSP_VSECDMA_UNROLL_ADDR_OFFSET_LOW_OFF	DMA Register Map Start Address Offset Low Register	000433CH	P	P	Kernel Reset	3469
PCIE0_DSP_VSECDMA_UNROLL_ADDR_OFFSET_HIGH_OFF	DMA Register Map Start Address Offset High Register	0004340H	P	P	Kernel Reset	3470
PCIE0_DSP_ACK_LATENCY_TIMER_OFF	Ack Latency Timer and Replay Timer Register	0004700H	P	P	Kernel Reset	3470

(table continues...)

Table 928 (continued) Register overview - PCIE0_DSP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE0_DSP_VENDOR_SPEC_DLLP_OFF	Vendor Specific DLLP Register	0004704H	P	P	Kernel Reset	3471
PCIE0_DSP_PORT_FORCE_OFF	Port Force Link Register	0004708H	P	P	Kernel Reset	3471
PCIE0_DSP_ACK_F_ASPM_CTRL_OFF	Ack Frequency and L0-L1 ASPM Control Register	000470CH	P	P	Kernel Reset	3473
PCIE0_DSP_PORT_LINK_CTRL_OFF	Port Link Control Register	0004710H	P	P	Kernel Reset	3475
PCIE0_DSP_LANE_SKEW_OFF	Lane Skew Register	0004714H	P	P	Kernel Reset	3477
PCIE0_DSP_TIMER_CTRL_MAX_FUNC_NUM_OFF	Timer Control and Max Function Number Register	0004718H	P	P	Kernel Reset	3479
PCIE0_DSP_SYMBOL_TIMER_FILTER_1_OFF	Symbol Timer Register and Filter Mask 1 Register	000471CH	P	P	Kernel Reset	3480
PCIE0_DSP_FILTER_MASK_2_OFF	Filter Mask 2 Register	0004720H	P	P	Kernel Reset	3483
PCIE0_DSP_PL_DEBUG0_OFF	Debug Register 0	0004728H	P	P	Kernel Reset	3485
PCIE0_DSP_PL_DEBUG1_OFF	Debug Register 1	000472CH	P	P	Kernel Reset	3486
PCIE0_DSP_TX_P_FC_CREDIT_STATUS_OFF	Transmit Posted FC Credit Status	0004730H	P	P	Kernel Reset	3487
PCIE0_DSP_TX_NP_FC_CREDIT_STATUS_OFF	Transmit Non-Posted FC Credit Status	0004734H	P	P	Kernel Reset	3488
PCIE0_DSP_TX_CPL_FC_CREDIT_STATUS_OFF	Transmit Completion FC Credit Status	0004738H	P	P	Kernel Reset	3489
PCIE0_DSP_QUEUE_STATUS_OFF	Queue Status	000473CH	P	P	Kernel Reset	3490
PCIE0_DSP_VC0_P_RX_Q_CTRL_OFF	Segmented-Buffer VC0 Posted Receive Queue Control	0004748H	P	P	Kernel Reset	3492
PCIE0_DSP_VC0_NP_RX_Q_CTRL_OFF	Segmented-Buffer VC0 Non-Posted Receive Queue Control	000474CH	P	P	Kernel Reset	3494

(table continues...)

Table 928 (continued) Register overview - PCIE0_DSP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE0_DSP_VC0_CPL_RX_Q_CTRL_OFF	Segmented-Buffer VC0 Completion Receive Queue Control	0004750H	P	P	Kernel Reset	3495
PCIE0_DSP_GEN2_CTRL_OFF	Link Width and Speed Change Control Register	000480CH	P	P	Kernel Reset	3496
PCIE0_DSP_MSI_CTRL_ADDR_OFF	Integrated MSI Reception Module (iMRM) Address Register	0004820H	P	P	Kernel Reset	3500
PCIE0_DSP_MSI_CTRL_UPPER_ADDR_OFF	Integrated MSI Reception Module Upper Address Register	0004824H	P	P	Kernel Reset	3501
PCIE0_DSP_MSI_CTRL_INT_i_EN_OFF (i=0-7)	Integrated MSI i Reception Module Interrupt0 Enable Register	0004828H+i*12	P	P	Kernel Reset	3501
PCIE0_DSP_MSI_CTRL_INT_i_MASK_OFF (i=0-7)	Integrated MSI i Reception Module Interrupt0 Mask Register	000482CH+i*12	P	P	Kernel Reset	3502
PCIE0_DSP_MSI_CTRL_INT_i_STATUS_OFF (i=0-7)	Integrated MSI i Reception Module Interrupt0 Status Register	0004830H+i*12	P	P	Kernel Reset	3502
PCIE0_DSP_MSI_GPIO_IO_OFF	Integrated MSI Reception Module General Purpose IO Register	0004888H	P	P	Kernel Reset	3503
PCIE0_DSP_CLOCK_GATING_CTRL_OFF	Clock Gating Control Register	000488CH	P	P	Kernel Reset	3503
PCIE0_DSP_GEN3_RELATED_OFF	Gen3 Control Register	0004890H	P	P	Kernel Reset	3504
PCIE0_DSP_GEN3_EQ_CONTROL_OFF	Gen3 EQ Control Register	00048A8H	P	P	Kernel Reset	3507
PCIE0_DSP_GEN3_EQ_FB_MODE_DIR_CHANGE_OFF	Gen3 EQ Direction Change Feedback Mode Control Register	00048ACH	P	P	Kernel Reset	3510
PCIE0_DSP_ORDER_RULE_CTRL_OFF	Order Rule Control Register	00048B4H	P	P	Kernel Reset	3511
PCIE0_DSP_PIPE_LOOPBACK_CONTROL_OFF	PIPE Loopback Control Register	00048B8H	P	P	Kernel Reset	3512

(table continues...)

Table 928 (continued) Register overview - PCIE0_DSP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE0_DSP_MISC_CONTROL_1_OFF	DBI Read-Only Write Enable Register	00048BC H	P	P	Kernel Reset	3513
PCIE0_DSP_LINK_FLUSH_CONTROL_OFF	Link Reset Request Flush Control Register	00048CC H	P	P	Kernel Reset	3519
PCIE0_DSP_AMBA_ERROR_RESPONSE_DEFAULT_OFF	AXI Bridge Slave Error Response Register	00048D0 H	P	P	Kernel Reset	3520
PCIE0_DSP_AMBA_LINK_TIMEOUT_OFF	Link Down AXI Bridge Slave Timeout Register	00048D4 H	P	P	Kernel Reset	3521
PCIE0_DSP_AMBA_ORDERING_CTRL_OFF	AXI Bridge Ordering Control	00048D8 H	P	P	Kernel Reset	3522
PCIE0_DSP_PCIE_VERSION_NUMBER_OFF	PCIe Controller IIP Release Version Number	00048F8 H	P	P	Kernel Reset	3526
PCIE0_DSP_PCIE_VERSION_TYPE_OFF	PCIe Controller IIP Release Version Type	00048FC H	P	P	Kernel Reset	3527
PCIE0_DSP_INTERFACE_TIMER_CONTROL_OFF	Interface Timer Control Register	0004930 H	P	P	Kernel Reset	3527
PCIE0_DSP_INTERFACE_TIMER_TARGET_OFF	Interface Timer Target Register	0004934 H	P	P	Kernel Reset	3528
PCIE0_DSP_INTERFACE_TIMER_STATUS_OFF	Interface Timer Status Register	0004938 H	P	P	Kernel Reset	3529
PCIE0_DSP_MSIX_ADDRESS_MATCH_LOW_OFF	MSI-X Address Match Low Register	0004940 H	P	P	Kernel Reset	3530
PCIE0_DSP_MSIX_ADDRESS_MATCH_HIGH_OFF	MSIX Address Match High Register	0004944 H	P	P	Kernel Reset	3531
PCIE0_DSP_MSIX_DOORBELL_OFF	MSI-X Doorbell Register	0004948 H	P	P	Kernel Reset	3531
PCIE0_DSP_SAFETY_MASK_OFF	Functional Safety Interrupt Events Mask Register	0004960 H	P	P	Kernel Reset	3534
PCIE0_DSP_SAFETY_STATUS_OFF	Status for functional safety interrupt events	0004964 H	P	P	Kernel Reset	3535

(table continues...)

Table 928 (continued) Register overview - PCIE0_DSP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE0_DSP_PL_APP_BUS_DEV_NUM_STATUS_OFF	Application driven bus and device number register	0004B10H	P	P	Kernel Reset	3536
PCIE0_DSP_PL_CHK_REG_CONTROL_STATUS_OFF	CDM Register Checking Control and Status Register	0004B20H	P	P	Kernel Reset	3537
PCIE0_DSP_PL_CHK_REG_START_END_OFF	CDM Register Checking First and Last address to check	0004B24H	P	P	Kernel Reset	3539
PCIE0_DSP_PL_CHK_REG_ERR_ADDR_OFF	CDM Register Checking Error Address	0004B28H	P	P	Kernel Reset	3539
PCIE0_DSP_PL_CHK_REG_ERR_PF_VF_OFF	CDM Register Checking error PF Numbers	0004B2CH	P	P	Kernel Reset	3540
PCIE0_DSP_PL_LTR_LATENCY_OFF	LTR Latency Register	0004B30H	P	P	Kernel Reset	3540
PCIE0_DSP_AUX_CLK_FREQ_OFF	Auxiliary Clock Frequency Control Register	0004B40H	P	P	Kernel Reset	3541
PCIE0_DSP_POWERDOWN_CTRL_STATUS_OFF	Powerdown Control and Status Register	0004B48H	P	P	Kernel Reset	3543
PCIE0_DSP_MSIXPBAx_DWORD (x=0-1)	MSIX x table DWORD	0005E10H+x*4	P	P	Kernel Reset	3545
PCIE0_DSP_SHADOW_PCI_MSIX_CAP_ID_NEXT_CTRL_REG	MSI-X Capability ID, Next Pointer, Control Register	00060B0H	P	P	Kernel Reset	3545
PCIE0_DSP_DMA_CTRL_DATA_ARB_PRIOR_OFF	DMA Arbitration Scheme for TRGT1 Interface	0007000H	P	P	Kernel Reset	3547
PCIE0_DSP_DMA_CTRL_OFF	DMA Number of Channels Register	0007008H	P	P	Kernel Reset	3548
PCIE0_DSP_DMA_WRITE_ENGINE_EN_OFF	DMA Write Engine Enable Register	000700CH	P	P	Kernel Reset	3549
PCIE0_DSP_DMA_WRITE_DOORBELL_OFF	DMA Write Doorbell Register	0007010H	P	P	Kernel Reset	3551

(table continues...)

Table 928 (continued) Register overview - PCIE0_DSP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE0_DSP_DMA_WRITE_CHANNEL_ARB_WEIGHT_LOW_OFF	DMA Write Engine Channel Arbitration Weight Low Register	0007018H	P	P	Kernel Reset	3552
PCIE0_DSP_DMA_READ_ENGINE_EN_OFF	DMA Read Engine Enable Register	000702CH	P	P	Kernel Reset	3554
PCIE0_DSP_DMA_READ_DOORBELL_OFF	DMA Read Doorbell Register	0007030H	P	P	Kernel Reset	3555
PCIE0_DSP_DMA_READ_CHANNEL_ARB_WEIGHT_LOW_OFF	DMA Read Engine Channel Arbitration Weight Low Register	0007038H	P	P	Kernel Reset	3556
PCIE0_DSP_DMA_WRITE_INT_STATUS_OFF	DMA Write Interrupt Status Register	000704CH	P	P	Kernel Reset	3558
PCIE0_DSP_DMA_WRITE_INT_MASK_OFF	DMA Write Interrupt Mask Register	0007054H	P	P	Kernel Reset	3559
PCIE0_DSP_DMA_WRITE_INT_CLEAR_OFF	DMA Write Interrupt Clear Register	0007058H	P	P	Kernel Reset	3560
PCIE0_DSP_DMA_WRITE_ERR_STATUS_OFF	DMA Write Error Status Register	000705CH	P	P	Kernel Reset	3561
PCIE0_DSP_DMA_WRITE_DONE_IMWR_LOW_OFF	DMA Write Done IMWr Address Low Register	0007060H	P	P	Kernel Reset	3561
PCIE0_DSP_DMA_WRITE_DONE_IMWR_HIGH_OFF	DMA Write Done IMWr Interrupt Address High Register	0007064H	P	P	Kernel Reset	3562
PCIE0_DSP_DMA_WRITE_ABORT_IMWR_LOW_OFF	DMA Write Abort IMWr Address Low Register	0007068H	P	P	Kernel Reset	3562
PCIE0_DSP_DMA_WRITE_ABORT_IMWR_HIGH_OFF	DMA Write Abort IMWr Address High Register	000706CH	P	P	Kernel Reset	3563
PCIE0_DSP_DMA_WRITE_CH01_IMWR_DATA_OFF	DMA Write Channel 1 and 0 IMWr Data Register	0007070H	P	P	Kernel Reset	3563
PCIE0_DSP_DMA_WRITE_CH23_IMWR_DATA_OFF	DMA Write Channel 3 and 2 IMWr Data Register	0007074H	P	P	Kernel Reset	3564

(table continues...)

Table 928 (continued) Register overview - PCIE0_DSP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE0_DSP_DMA_WRITE_LINKED_LIST_ERR_EN_OFF	DMA Write Linked List Error Enable Register	0007090H	P	P	Kernel Reset	3565
PCIE0_DSP_DMA_READ_INT_STATUS_OFF	DMA Read Interrupt Status Register	00070A0H	P	P	Kernel Reset	3566
PCIE0_DSP_DMA_READ_INT_MASK_OFF	DMA Read Interrupt Mask Register	00070A8H	P	P	Kernel Reset	3567
PCIE0_DSP_DMA_READ_INT_CLEAR_OFF	DMA Read Interrupt Clear Register	00070ACH	P	P	Kernel Reset	3568
PCIE0_DSP_DMA_READ_ERR_STATUS_LOW_OFF	DMA Read Error Status Low Register	00070B4H	P	P	Kernel Reset	3569
PCIE0_DSP_DMA_READ_ERR_STATUS_HIGH_OFF	DMA Read Error Status High Register	00070B8H	P	P	Kernel Reset	3570
PCIE0_DSP_DMA_READ_LINKED_LIST_ERR_EN_OFF	DMA Read Linked List Error Enable Register	00070C4H	P	P	Kernel Reset	3571
PCIE0_DSP_DMA_READ_DONE_IMWR_LOW_OFF	DMA Read Done IMWr Address Low Register	00070CCH	P	P	Kernel Reset	3572
PCIE0_DSP_DMA_READ_DONE_IMWR_HIGH_OFF	DMA Read Done IMWr Address High Register	00070D0H	P	P	Kernel Reset	3572
PCIE0_DSP_DMA_READ_ABORT_IMWR_LOW_OFF	DMA Read Abort IMWr Address Low Register	00070D4H	P	P	Kernel Reset	3573
PCIE0_DSP_DMA_READ_ABORT_IMWR_HIGH_OFF	DMA Read Abort IMWr Address High Register	00070D8H	P	P	Kernel Reset	3573
PCIE0_DSP_DMA_READ_CH01_IMWR_DATA_OFF	DMA Read Channel 1 and 0 IMWr Data Register	00070DCH	P	P	Kernel Reset	3574
PCIE0_DSP_DMA_READ_CH23_IMWR_DATA_OFF	DMA Read Channel 3 and 2 IMWr Data Register	00070E0H	P	P	Kernel Reset	3574
PCIE0_DSP_DMA_WRITE_ENGINE_HSHAKE_CNT_LOW_OFF	DMA Write Engine Handshake Counter Channel 0/1/2/3 Register	0007108H	P	P	Kernel Reset	3576

(table continues...)

Table 928 (continued) Register overview - PCIE0_DSP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE0_DSP_DMA_READ_ENGINE_HSHAKE_CNT_LOW_OFF	DMA Read Engine Handshake Counter Channel 0/1/2/3 Register	0007118$_H$	P	P	Kernel Reset	3577
PCIE0_DSP_DMA_WRITE_CHi_PWR_EN_OFF (i=0-3)	DMA Write Channel i Power Enable Register	0007128$_H$+i*4	P	P	Kernel Reset	3579
PCIE0_DSP_DMA_READ_CHi_PWR_EN_OFF (i=0-3)	DMA Read Channel i Power Enable Register	0007168$_H$+i*4	P	P	Kernel Reset	3579
PCIE0_DSP_DMA_CH_CONTROL1_OFF_WRCH_i (i=0-3)	DMA Write Channel i Control 1 Register	0007200$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3580
PCIE0_DSP_DMA_TRANSFER_SIZE_OFF_WRCH_i (i=0-3)	DMA CH i Write Transfer Size Register	0007208$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3583
PCIE0_DSP_DMA_SAR_LOW_OFF_WRCH_i (i=0-3)	DMA CH i Write SAR Low Register	000720C$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3583
PCIE0_DSP_DMA_SAR_HIGH_OFF_WRCH_i (i=0-3)	DMA CH i Write SAR High Register	0007210$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3584
PCIE0_DSP_DMA_DAR_LOW_OFF_WRCH_i (i=0-3)	DMA CH i Write DAR Low Register	0007214$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3584
PCIE0_DSP_DMA_DAR_HIGH_OFF_WRCH_i (i=0-3)	DMA CH i Write DAR High Register	0007218$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3585
PCIE0_DSP_DMA_LLP_LOW_OFF_WRCH_i (i=0-3)	DMA CH i Write Linked List Pointer Low Register	000721C$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3585
PCIE0_DSP_DMA_LLP_HIGH_OFF_WRCH_i (i=0-3)	DMA CH i Write Linked List Pointer High Register	0007220$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3586

(table continues...)

Table 928 (continued) Register overview - PCIE0_DSP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE0_DSP_DMA CAP_CHi_WRCH N_WR_INT_CLEA R (i=0-3)	Interrupt clear for DMA write channel i	0007224$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3586
PCIE0_DSP_DMA CAP_CHi_WRCH N_WR_INT_STAT US (i=0-3)	Interrupt status for DMA write channel i	0007228$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3587
PCIE0_DSP_DMA CAP_CHi_WRCH N_WR_ERR_STAT US (i=0-3)	Error status for DMA write channel i	000722C$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3588
PCIE0_DSP_DMA CAP_CHi_WRCH N_WR_DOORBEL L (i=0-3)	Doorbell for DMA write channel i	0007230$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3589
PCIE0_DSP_DMA _CH_CONTROL1 _OFF_RDCH_i (i=0-3)	DMA Read Channel i Control 1 Register	0007300$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3589
PCIE0_DSP_DMA _TRANSFER_SIZ E_OFF_RDCH_i (i=0-3)	DMA Read CH i Transfer Size Register	0007308$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3592
PCIE0_DSP_DMA _SAR_LOW_OFF_ RDCH_i (i=0-3)	DMA Read CH i SAR Low Register	000730C$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3593
PCIE0_DSP_DMA _SAR_HIGH_OFF _RDCH_i (i=0-3)	DMA Read CH i SAR High Register	0007310$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3594
PCIE0_DSP_DMA _DAR_LOW_OFF_ RDCH_i (i=0-3)	DMA Read CH i DAR Low Register	0007314$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3594
PCIE0_DSP_DMA _DAR_HIGH_OFF _RDCH_i (i=0-3)	DMA Read CH i DAR High Register	0007318$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3595

(table continues...)

Table 928 (continued) Register overview - PCIE0_DSP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE0_DSP_DMA_LLP_LOW_OFF_RDCH_i (i=0-3)	DMA Read CH i Linked List Pointer Low Register	000731C$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3595
PCIE0_DSP_DMA_LLP_HIGH_OFF_RDCH_i (i=0-3)	DMA CH i Read Linked List Pointer High Register	0007320$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3596
PCIE0_DSP_DMACAP_CHi_RDCHN_RD_INT_CLEAR (i=0-3)	Interrupt clear for DMA read channel i	0007324$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3596
PCIE0_DSP_DMACAP_CHi_RDCHN_RD_INT_STATUS (i=0-3)	Interrupt status for DMA read channel i	0007328$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3597
PCIE0_DSP_DMACAP_CHi_RDCHN_RD_ERR_STATUS (i=0-3)	Error status for DMA read channel i	000732C$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3597
PCIE0_DSP_DMACAP_CHi_RDCHN_RD_ERR_STATUS_HIGH (i=0-3)	Higher error status for DMA read channel i	0007330$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3598
PCIE0_DSP_DMACAP_CHi_RDCHN_RD_DOORBELL (i=0-3)	Doorbell for DMA read channel i	0007334$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3599

19.8.4.13 Register overview - PCIE1_DSP domain FPI (ascending offset address)

Table 929 Register overview - PCIE1_DSP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE1_DSP_CLC	Clock control register	0000000$_H$	P	P, SV, E	Application Reset	3218
PCIE1_DSP_OCS	OCDS control and status register	0000004$_H$	P	SV, P	Debug Reset	3219

(table continues...)

Table 929 (continued) Register overview - PCIE1_DSP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE1_DSP_ID	Module identification register	0000008$_H$	P	BE	PowerOn Reset	3219
PCIE1_DSP_RST_CTRLA	Reset control register A	000000C$_H$	P	P, SV, E	Application Reset	3220
PCIE1_DSP_RST_CTRLB	Reset control register B	0000010$_H$	P	P, SV, E	Application Reset	3220
PCIE1_DSP_RST_STAT	Reset status register	0000014$_H$	P	BE	Application Reset	3221
PCIE1_DSP_PROTE	PROT register endinit	0000020$_H$	U	SV, PROT	Application Reset	3222
PCIE1_DSP_PROTSE	PROT register safe endinit	0000024$_H$	U	SV, PROT	Application Reset	3223
PCIE1_DSP_ACCEN_WRA	Write access enable register A	0000040$_H$	U	SE, SV	Application Reset	3225
PCIE1_DSP_ACCEN_WRB	Write access enable register B	0000044$_H$	U	SE, SV	Application Reset	3225
PCIE1_DSP_ACCEN_RDA	Read access enable register A	0000048$_H$	U	SE, SV	Application Reset	3226
PCIE1_DSP_ACCEN_RDB	Read access enable register B	000004C$_H$	U	SE, SV	Application Reset	3226
PCIE1_DSP_ACCEN_VM	VM access enable register	0000050$_H$	U	SE, SV	Application Reset	3227
PCIE1_DSP_ACCEN_PRS	PRS access enable register	0000054$_H$	U	SE, SV	Application Reset	3227
PCIE1_DSP_DMA_ACCENy_WRA (y=0-3)	CHy Write access enable register A	0000060$_H$+y*20$_H$	U	SE, SV	Application Reset	3228
PCIE1_DSP_DMA_ACCENy_WRB (y=0-3)	CHy Write access enable register B	0000064$_H$+y*20$_H$	U	SE, SV	Application Reset	3229
PCIE1_DSP_DMA_ACCENy_RDA (y=0-3)	CHy Read access enable register A	0000068$_H$+y*20$_H$	U	SE, SV	Application Reset	3229
PCIE1_DSP_DMA_ACCENy_RDB (y=0-3)	CHy Read access enable register B	000006C$_H$+y*20$_H$	U	SE, SV	Application Reset	3230
PCIE1_DSP_DMA_ACCENy_VM (y=0-3)	CHy VM access enable register	0000070$_H$+y*20$_H$	U	SE, SV	Application Reset	3230
PCIE1_DSP_DMA_ACCENy_PRS (y=0-3)	CHy PRS access enable register	0000074$_H$+y*20$_H$	U	SE, SV	Application Reset	3231

(table continues...)

Table 929 (continued) Register overview - PCIE1_DSP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE1_DSP_UPCS_OVRD1	UPCS layer override register 1	0000300H	P	P	Kernel Reset	3231
PCIE1_DSP_UPCS_OVRD2	UPCS layer override register 2	0000304H	P	P	Kernel Reset	3232
PCIE1_DSP_UPCS_OVRD3	UPCS layer override register 3	0000308H	P	P	Kernel Reset	3232
PCIE1_DSP_UPCS_OVRD4	UPCS layer override register 4	000030CH	P	P	Kernel Reset	3233
PCIE1_DSP_UPCS_OVRD5	UPCS layer override register 5	0000310H	P	P	Kernel Reset	3233
PCIE1_DSP_UPCS_OVRD6	UPCS layer override register 6	0000314H	P	P	Kernel Reset	3234
PCIE1_DSP_UPCS_OVRD7	UPCS layer override register 7	0000318H	P	P	Kernel Reset	3234
PCIE1_DSP_UPCS_OVRD8	UPCS layer override register 8	000031CH	P	P	Kernel Reset	3235
PCIE1_DSP_UPCS_OVRD9	UPCS layer override register 9	0000320H	P	P	Kernel Reset	3235
PCIE1_DSP_UPCS_OVRD10	UPCS layer override register 10	0000324H	P	P	Kernel Reset	3236
PCIE1_DSP_UPCS_OVRD11	UPCS layer override register 11	0000328H	P	P	Kernel Reset	3236
PCIE1_DSP_UPCS_OVRD12	UPCS layer override register 12	000032CH	P	P	Kernel Reset	3236
PCIE1_DSP_UPCS_OVRD13	UPCS layer override register 13	0000330H	P	P	Kernel Reset	3237
PCIE1_DSP_UPCS_OVRD14	UPCS layer override register 14	0000334H	P	P	Kernel Reset	3237
PCIE1_DSP_UPCS_OVRD15	UPCS layer override register 15	0000338H	P	P	Kernel Reset	3238
PCIE1_DSP_UPCS_OVRD16	UPCS layer override register 16	000033CH	P	P	Kernel Reset	3238
PCIE1_DSP_UPCS_OVRD17	UPCS layer override register 17	0000340H	P	P	Kernel Reset	3239
PCIE1_DSP_INT_MSICTRL	Request MSI interrupt, EP only.	0000348H	P	P	Kernel Reset	3239
PCIE1_DSP_INT_MSISTAT	Pending status of MSI interrupt request, EP only.	000034CH	P	P	Kernel Reset	3240
PCIE1_DSP_INT_INTXCTRL	Trigger INTX interrupt, EP only.	0000350H	P	P	Kernel Reset	3240
PCIE1_DSP_INT_TRIGSTAT	Trigger status	0000358H	P	P	Kernel Reset	3241

(table continues...)

Table 929 (continued) Register overview - PCIE1_DSP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE1_DSP_INT_TRIGCLR	Interrupt clear, EP only.	000035C$_H$	P	P	Kernel Reset	3241
PCIE1_DSP_INT_TRIGEN	Interrupt enable EP only.	0000360$_H$	P	P	Kernel Reset	3242
PCIE1_DSP_INT_REQTRIG	Receive incoming interrupt request, EP only.	0000364$_H$	P	P	Kernel Reset	3242
PCIE1_DSP_CFG_INFO	Configuration information	0000368$_H$	P	P	Kernel Reset	3243
PCIE1_DSP_CFG_BUSNUM	Bus/device number	000036C$_H$	P	P	Kernel Reset	3244
PCIE1_DSP_LNK_DVCTYP	Select device/port type	0000370$_H$	P	P, E	PowerOn Reset	3245
PCIE1_DSP_LNK_STAT	Indicate link status	0000374$_H$	P	P	Kernel Reset	3246
PCIE1_DSP_LNK_CORECTRL	Controller control	0000378$_H$	P	P	Kernel Reset	3247
PCIE1_DSP_LNK_INTEN	Enable the interrupt requests to SRC_PCIELNK	000037C$_H$	P	P	Kernel Reset	3250
PCIE1_DSP_TXRX_RXCTRL	Rx control	0000380$_H$	P	P	Kernel Reset	3250
PCIE1_DSP_TXRX_TXCTRL0	Tx control 0	0000384$_H$	P	P	Kernel Reset	3251
PCIE1_DSP_PAYLOADz (z=0-1)	Message payload z information	000038C$_H$+z*4	P	P	Kernel Reset	3252
PCIE1_DSP_MSG_INFO	Message information	0000394$_H$	P	P	Kernel Reset	3253
PCIE1_DSP_PM_CTRL	Power management control	0000398$_H$	P	P	Kernel Reset	3254
PCIE1_DSP_PM_STAT0	Power management status 0	000039C$_H$	P	P	Kernel Reset	3256
PCIE1_DSP_PM_STAT1	Power management status 1	00003A4$_H$	P	P	Kernel Reset	3258
PCIE1_DSP_PM_INTCLR	Power management interrupt clear	00003A8$_H$	P	P	Kernel Reset	3258
PCIE1_DSP_PM_INTEN	Power management interrupt enable	00003AC$_H$	P	P	Kernel Reset	3259
PCIE1_DSP_ERR_STAT	Error status	00003B0$_H$	P	P	Kernel Reset	3260
PCIE1_DSP_ERR_INTCLR	Error clear	00003B4$_H$	P	P	Kernel Reset	3260

(table continues...)

Table 929 (continued) Register overview - PCIE1_DSP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE1_DSP_ERR_INTEN	Error enable	00003B8$_H$	P	P	Kernel Reset	3261
PCIE1_DSP_DBG_STAT	Debug status	00003BC$_H$	P	P	Kernel Reset	3262
PCIE1_DSP_DBG_DATA	Debug information	00003C0$_H$	P	P	Kernel Reset	3263
PCIE1_DSP_LTR_GNRTN	LTR message	00003C4$_H$	P	P	Kernel Reset	3263
PCIE1_DSP_LTR_CTRL	LTR control	00003CC$_H$	P	P	Kernel Reset	3263
PCIE1_DSP_PTM_CTRL	PTM control	00003D0$_H$	P	P	Kernel Reset	3264
PCIE1_DSP_LCLCLKz (z=0-1)	Local Clock z value	00003D4$_H$+z*4	P	P	Kernel Reset	3265
PCIE1_DSP_CLKCORRz (z=0-1)	Clock z correction	00003DC$_H$+z*4	P	P	Kernel Reset	3265
PCIE1_DSP_MSKCLKz (z=0-1)	PTM z External Master Time	00003E4$_H$+z*4	P	P	Kernel Reset	3266
PCIE1_DSP_OBFFMSG	OBFF message control	00003F0$_H$	P	P	Kernel Reset	3267
PCIE1_DSP_OBFFWAKE	OBFF WAKE control	00003F4$_H$	P	P	Kernel Reset	3269
PCIE1_DSP_CFGBD	PCIe configuration control	00003F8$_H$	P	P	Kernel Reset	3271
PCIE1_DSP_DATAx (x=0-7)	ELBI x mailbox register	00003FC$_H$+x*4	P	P	Kernel Reset	3271
PCIE1_DSP_LLCTRL	DMA link list control register	000041C$_H$	P	P	Kernel Reset	3272
PCIE1_DSP_OTSS	PCIe OCDS Trigger Set Select	0000420$_H$	P	SV, P	Debug Reset	3272
PCIE1_DSP_ALARM_STAT	Interrupt/alarm status	0000424$_H$	P	P	Kernel Reset	3273
PCIE1_DSP_ALARM_INTCLR	Interrupt/alarm clear	0000428$_H$	P	P	Kernel Reset	3275
PCIE1_DSP_ALARM_INTEN	Interrupt/alarm enable	000042C$_H$	P	P	Kernel Reset	3276

(table continues...)

Table 929 (continued) Register overview - PCIE1_DSP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE1_DSP_IATU_REGION_CTRL_1_OFF_OUTBOUND_i (i=0-7)	iATU Region i Control 1 Register Outbound	0003000$_H$+i*200$_H$	P	P	Kernel Reset	3277
PCIE1_DSP_IATU_REGION_CTRL_2_OFF_OUTBOUND_i (i=0-7)	iATU Region i Control 2 Register Outbound	0003004$_H$+i*200$_H$	P	P	Kernel Reset	3278
PCIE1_DSP_IATU_LWR_BASE_ADDR_OFF_OUTBOUND_i (i=0-7)	iATU i Lower Base Address Register Outbound	0003008$_H$+i*200$_H$	P	P	Kernel Reset	3280
PCIE1_DSP_IATU_UPPER_BASE_ADDR_OFF_OUTBOUND_i (i=0-7)	iATU i Upper Base Address Register Outbound	000300C$_H$+i*200$_H$	P	P	Kernel Reset	3281
PCIE1_DSP_IATU_LIMIT_ADDR_OFF_OUTBOUND_i (i=0-7)	iATU i Limit Address Register Outbound	0003010$_H$+i*200$_H$	P	P	Kernel Reset	3281
PCIE1_DSP_IATU_LWR_TARGET_ADDR_OFF_OUTBOUND_i (i=0-7)	iATU i Lower Target Address Register Outbound	0003014$_H$+i*200$_H$	P	P	Kernel Reset	3282
PCIE1_DSP_IATU_UPPER_TARGET_ADDR_OFF_OUTBOUND_i (i=0-7)	iATU i Upper Target Address Register Outbound	0003018$_H$+i*200$_H$	P	P	Kernel Reset	3282
PCIE1_DSP_IATU_REGION_CTRL_1_OFF_INBOUND_i (i=0-7)	iATU Region i Control 1 Register Inbound	0003100$_H$+i*200$_H$	P	P	Kernel Reset	3283
PCIE1_DSP_IATU_REGION_CTRL_2_OFF_INBOUND_i (i=0-7)	iATU Region i Control 2 Register Inbound	0003104$_H$+i*200$_H$	P	P	Kernel Reset	3284

(table continues...)

Table 929 (continued) Register overview - PCIE1_DSP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE1_DSP_IATU_LWR_BASE_ADDR_OFF_INBOUND_i (i=0-7)	iATU i Lower Base Address Register Inbound	0003108$_H$+i*200$_H$	P	P	Kernel Reset	3288
PCIE1_DSP_IATU_UPPER_BASE_ADDR_OFF_INBOUND_i (i=0-7)	iATU i Upper Base Address Register Inbound	000310C$_H$+i*200$_H$	P	P	Kernel Reset	3288
PCIE1_DSP_IATU_LIMIT_ADDR_OFF_INBOUND_i (i=0-7)	iATU i Limit Address Register Inbound	0003110$_H$+i*200$_H$	P	P	Kernel Reset	3289
PCIE1_DSP_IATU_LWR_TARGET_ADDR_OFF_INBOUND_i (i=0-7)	iATU i Lower Target Address Register Inbound	0003114$_H$+i*200$_H$	P	P	Kernel Reset	3290
PCIE1_DSP_TYPE1_DEV_ID_VEND_ID_REG	Device ID and Vendor ID Register	0004000$_H$	P	P	Kernel Reset	3290
PCIE1_DSP_TYPE1_STATUS_COMMAND_REG	Status and Command Register	0004004$_H$	P	P	Kernel Reset	3291
PCIE1_DSP_TYPE1_CLASS_CODE_REV_ID_REG	Class Code and Revision ID Register	0004008$_H$	P	P	Kernel Reset	3295
PCIE1_DSP_TYPE1_BIST_HDR_TYPE_LAT_CACHE_LINE_SIZE_REG	BIST, Header Type, Latency Timer, and Cache Line Size Register	000400C$_H$	P	P	Kernel Reset	3296
PCIE1_DSP_BARi_REG (i=0-1)	BAR0 Register i	0004010$_H$+i*4	P	P	Kernel Reset	3297
PCIE1_DSP_SEC_LAT_TIMER_SUB_BUS_SEC_BUS_PRI_BUS_REG	Secondary Latency Timer, Subordinate Bus Number, Secondary Bus Number, and Primary Bus Number Register	0004018$_H$	P	P	Kernel Reset	3299
PCIE1_DSP_SEC_STAT_IO_LIMIT_IO_BASE_REG	Secondary Status, and I/O Limit and Base Register	000401C$_H$	P	P	Kernel Reset	3300

(table continues...)

Table 929 (continued) Register overview - PCIE1_DSP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE1_DSP_MEM_LIMIT_MEM_BASE_REG	Memory Limit and Base Register	0004020H	P	P	Kernel Reset	3303
PCIE1_DSP_PREF_MEM_LIMIT_PREF_MEM_BASE_REG	Prefetchable Memory Limit and Base Register	0004024H	P	P	Kernel Reset	3304
PCIE1_DSP_PREF_BASE_UPPER_REG	Prefetchable Base Upper 32 Bits Register	0004028H	P	P	Kernel Reset	3306
PCIE1_DSP_PREF_LIMIT_UPPER_REG	Prefetchable Limit Upper 32 Bits Register	000402CH	P	P	Kernel Reset	3306
PCIE1_DSP_IO_LIMIT_UPPER_IO_BASE_UPPER_REG	I/O Limit and Base Upper 16 Bits Register	0004030H	P	P	Kernel Reset	3307
PCIE1_DSP_TYPE1_CAP_PTR_REG	Capabilities Pointer Register	0004034H	P	P	Kernel Reset	3308
PCIE1_DSP_TYPE1_EXP_ROM_BASE_REG	Expansion ROM BAR Register	0004038H	P	P	Kernel Reset	
PCIE1_DSP_BRIDGE_CTRL_INT_PIN_INT_LINE_REG	Bridge Control, Interrupt Pin, and Interrupt Line Register	000403CH	P	P	Kernel Reset	3309
PCIE1_DSP_CAP_ID_NXT_PTR_REG	Power Management Capabilities Register	0004040H	P	P	Kernel Reset	3312
PCIE1_DSP_CON_STATUS_REG	Power Management Control and Status Register	0004044H	P	P	Kernel Reset	3314
PCIE1_DSP_PCI_MSI_CAP_ID_NEXT_CTRL_REG	MSI Capability Header and Message Control Register	0004050H	P	P	Kernel Reset	3317
PCIE1_DSP_MSI_CAP_OFF_04H_REG	Message Address Register for MSI (Offset 04h)	0004054H	P	P	Kernel Reset	3319
PCIE1_DSP_MSI_CAP_OFF_08H_REG	Message Address Register for MSI (Offset 08h)	0004058H	P	P	Kernel Reset	3320
PCIE1_DSP_MSI_CAP_OFF_0CH_REG	Message Address Register for MSI (Offset 0Ch)	000405CH	P	P	Kernel Reset	3321
PCIE1_DSP_MSI_CAP_OFF_10H_REG	Message Address Register for MSI (Offset 10h)	0004060H	P	P	Kernel Reset	3322

(table continues...)

Table 929 (continued) Register overview - PCIE1_DSP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE1_DSP_MSI_CAP_OFF_14H_REG	Message Address Register for MSI (Offset 14h)	0004064H	P	P	Kernel Reset	3322
PCIE1_DSP_PCIE_CAP_ID_PCIE_NEXT_CAP_PTR_PCIE_CAP_REG	PCI Express Capabilities, ID, Next Pointer Register	0004070H	P	P	Kernel Reset	3323
PCIE1_DSP_DEVICE_CAPABILITIES_REG	Device Capabilities Register	0004074H	P	P	Kernel Reset	3325
PCIE1_DSP_DEVICE_CONTROL_DEVICE_STATUS	Device Control and Device Status Register	0004078H	P	P	Kernel Reset	3327
PCIE1_DSP_LINK_CAPABILITIES_REG	Link Capabilities Register	000407CH	P	P	Kernel Reset	3332
PCIE1_DSP_LINK_CONTROL_LINK_STATUS_REG	Link Control and Link Status Register	0004080H	P	P	Kernel Reset	3337
PCIE1_DSP_SLOT_CAPABILITIES_REG	Slot Capabilities Register	0004084H	P	P	Kernel Reset	3344
PCIE1_DSP_SLOT_CONTROL_SLOT_STATUS	Slot Control and Status Register	0004088H	P	P	Kernel Reset	3346
PCIE1_DSP_ROOT_CONTROL_ROOT_CAPABILITIES_REG	Root Control and Capabilities Register	000408CH	P	P	Kernel Reset	3351
PCIE1_DSP_ROOT_STATUS_REG	Root Status Register	0004090H	P	P	Kernel Reset	3352
PCIE1_DSP_DEVICE_CAPABILITIES2_REG	Device Capabilities 2 Register	0004094H	P	P	Kernel Reset	3353
PCIE1_DSP_DEVICE_CONTROL2_DEVICE_STATUS2_REG	Device Control 2 and Status 2 Register	0004098H	P	P	Kernel Reset	3356
PCIE1_DSP_LINK_CAPABILITIES2_REG	Link Capabilities 2 Register	000409CH	P	P	Kernel Reset	3359
PCIE1_DSP_LINK_CONTROL2_LINK_STATUS2_REG	Link Control 2 and Status 2 Register	00040A0H	P	P	Kernel Reset	3361

(table continues...)

Table 929 (continued) Register overview - PCIE1_DSP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE1_DSP_PCI_MSIX_CAP_ID_NEXT_CTRL_REG	MSI-X Capability ID, Next Pointer, Control Register	00040B0H	P	P	Kernel Reset	3366
PCIE1_DSP_MSIX_TABLE_OFFSET_REG	MSI-X Table Offset and BIR Register	00040B4H	P	P	Kernel Reset	3368
PCIE1_DSP_MSIX_PBA_OFFSET_REG	MSI-X PBA Offset and BIR Register	00040B8H	P	P	Kernel Reset	3368
PCIE1_DSP_AER_EXT_CAP_HDR_OFF	Advanced Error Reporting Extended Capability Header	0004100H	P	P	Kernel Reset	3369
PCIE1_DSP_UNCORR_ERR_STATUS_OFF	Uncorrectable Error Status Register	0004104H	P	P	Kernel Reset	3370
PCIE1_DSP_UNCORR_ERR_MASK_OFF	Uncorrectable Error Mask Register	0004108H	P	P	Kernel Reset	3372
PCIE1_DSP_UNCORR_ERR_SEV_OFF	Uncorrectable Error Severity Register	000410CH	P	P	Kernel Reset	3374
PCIE1_DSP_CORR_ERR_STATUS_OFF	Correctable Error Status Register	0004110H	P	P	Kernel Reset	3376
PCIE1_DSP_CORR_ERR_MASK_OFF	Correctable Error Mask Register	0004114H	P	P	Kernel Reset	3377
PCIE1_DSP_ADV_ERR_CAP_CTRL_OFF	Advanced Error Capabilities and Control Register	0004118H	P	P	Kernel Reset	3378
PCIE1_DSP_HDR_LOG_i_OFF (i=0-3)	Header i Log Register 0	000411CH+i*4	P	P	Kernel Reset	3379
PCIE1_DSP_AERCAP_ROOT_ROOT_ERR_CMD_OFF	Root Error Command Register	000412CH	P	P	Kernel Reset	3380
PCIE1_DSP_AERCAP_ROOT_ROOT_ERR_STATUS_OFF	Root Error Status Register	0004130H	P	P	Kernel Reset	3381
PCIE1_DSP_AERCAP_ERR_SRC_ID_OFF	Error Source Identification Register	0004134H	P	P	Kernel Reset	3382

(table continues...)

Table 929 (continued) Register overview - PCIE1_DSP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE1_DSP_TLP_PREFIX_LOG_i_OFF (i=0-3)	TLP Prefix i Log Register 1	0004138$_H$+i*4	P	P	Kernel Reset	3383
PCIE1_DSP_SN_BASE	Device Serial Number Extended Capability Header	0004148$_H$	P	P	Kernel Reset	3384
PCIE1_DSP_SER_NUM_REG_DW_1	Serial Number 1 Register	000414C$_H$	P	P	Kernel Reset	3385
PCIE1_DSP_SER_NUM_REG_DW_2	Serial Number 2 Register	0004150$_H$	P	P	Kernel Reset	3385
PCIE1_DSP_SPCIE_CAP_HEADER_REG	SPCIE Capability Header	0004158$_H$	P	P	Kernel Reset	3386
PCIE1_DSP_LINK_CONTROL3_REG	Link Control 3 Register	000415C$_H$	P	P	Kernel Reset	3386
PCIE1_DSP_LANE_ERR_STATUS_REG	Lane Error Status Register	0004160$_H$	P	P	Kernel Reset	3387
PCIE1_DSP_SPCIE_CAP_OFF_0CH_REG	Lane Equalization Control Register for lanes 1 and 0	0004164$_H$	P	P	Kernel Reset	3388
PCIE1_DSP_L1SUB_CAP_HEADER_REG	L1 Substates Extended Capability Header	0004170$_H$	P	P	Kernel Reset	3390
PCIE1_DSP_L1SUB_CAPABILITY_REG	L1 Substates Capability Register	0004174$_H$	P	P	Kernel Reset	3391
PCIE1_DSP_L1SUB_CONTROL1_REG	L1 Substates Control 1 Register	0004178$_H$	P	P	Kernel Reset	3392
PCIE1_DSP_L1SUB_CONTROL2_REG	L1 Substates Control 2 Register	000417C$_H$	P	P	Kernel Reset	3394
PCIE1_DSP_RAS_DES_CAP_HEADER_REG	Vendor-Specific Extended Capability Header	0004180$_H$	P	P	Kernel Reset	3395
PCIE1_DSP_VENDOR_SPECIFIC_HEADER_REG	Vendor-Specific Header	0004184$_H$	P	P	Kernel Reset	3396

(table continues...)

Table 929 (continued) Register overview - PCIE1_DSP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE1_DSP_EVENT_COUNTER_CONTROL_REG	Event Counter Control	0004188H	P	P	Kernel Reset	3397
PCIE1_DSP_EVENT_COUNTER_DATA_REG	Event Counter Data	000418CH	P	P	Kernel Reset	3399
PCIE1_DSP_TIME_BASED_ANALYSIS_CONTROL_REG	Time-based Analysis Control	0004190H	P	P	Kernel Reset	3400
PCIE1_DSP_TIME_BASED_ANALYSIS_DATA_REG	Time-based Analysis Data	0004194H	P	P	Kernel Reset	3401
PCIE1_DSP_TIME_BASED_ANALYSIS_DATA_63_32_REG	Upper 32 bits of Time-based Analysis Data	0004198H	P	P	Kernel Reset	3402
PCIE1_DSP_EINJ_ENABLE_REG	Error Injection Enable	00041B0H	P	P	Kernel Reset	3402
PCIE1_DSP_EINJ0_CRC_REG	Error Injection Control 0 (CRC Error)	00041B4H	P	P	Kernel Reset	3404
PCIE1_DSP_EINJ1_SEQNUM_REG	Error Injection Control 1 (Sequence Number Error). 	00041B8H	P	P	Kernel Reset	3405
PCIE1_DSP_EINJ2_DLLP_REG	Error Injection Control 2 (DLLP Error)	00041BCH	P	P	Kernel Reset	3406
PCIE1_DSP_EINJ3_SYMBOL_REG	Error Injection Control 3 (Symbol Error)	00041C0H	P	P	Kernel Reset	3407
PCIE1_DSP_EINJ4_FC_REG	Error Injection Control 4 (FC Credit Error)	00041C4H	P	P	Kernel Reset	3408
PCIE1_DSP_EINJ5_SP_TLP_REG	Error Injection Control 5 (Specific TLP Error)	00041C8H	P	P	Kernel Reset	3409
PCIE1_DSP_EINJ6_COMPARE_POINT_Hi_REG (i=0-3)	Error Injection Control 6 i (Compare Point Header DWORD #0)	00041CCH+i*4	P	P	Kernel Reset	3410
PCIE1_DSP_EINJ6_COMPARE_VALUE_Hi_REG (i=0-3)	Error Injection Control 6 i (Compare Value Header DWORD #0)	00041DCH+i*4	P	P	Kernel Reset	3411
PCIE1_DSP_EINJ6_CHANGE_POINT_Hi_REG (i=0-3)	Error Injection Control 6 i (Change Point Header DWORD #0)	00041ECH+i*4	P	P	Kernel Reset	3411

(table continues...)

Table 929 (continued) Register overview - PCIE1_DSP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE1_DSP_EINJ6_CHANGE_VALUE_Hi_REG (i=0-3)	Error Injection Control 6 i (Change Value Header DWORD #0)	00041FC$_H$+i*4	P	P	Kernel Reset	3412
PCIE1_DSP_EINJ6_TLP_REG	Error Injection Control 6 (Packet Error)	000420C$_H$	P	P	Kernel Reset	3413
PCIE1_DSP_SD_CONTROL1_REG	Silicon Debug Control 1	0004220$_H$	P	P	Kernel Reset	3414
PCIE1_DSP_SD_CONTROL2_REG	Silicon Debug Control 2	0004224$_H$	P	P	Kernel Reset	3415
PCIE1_DSP_SD_STATUS_L1LANE_REG	Silicon Debug Status(Layer1 Per-lane)	0004230$_H$	P	P	Kernel Reset	3416
PCIE1_DSP_SD_STATUS_L1LTSSM_REG	Silicon Debug Status(Layer1 LTSSM)	0004234$_H$	P	P	Kernel Reset	3417
PCIE1_DSP_SD_STATUS_PM_REG	Silicon Debug Status(PM)	0004238$_H$	P	P	Kernel Reset	3420
PCIE1_DSP_SD_STATUS_L2_REG	Silicon Debug Status(Layer2)	000423C$_H$	P	P	Kernel Reset	3423
PCIE1_DSP_SD_STATUS_L3FC_REG	Silicon Debug Status(Layer3 FC)	0004240$_H$	P	P	Kernel Reset	3423
PCIE1_DSP_SD_STATUS_L3_REG	Silicon Debug Status(Layer3)	0004244$_H$	P	P	Kernel Reset	3425
PCIE1_DSP_SD_EQ_CONTROL1_REG	Silicon Debug EQ Control 1	0004250$_H$	P	P	Kernel Reset	3426
PCIE1_DSP_SD_EQ_CONTROL2_REG	Silicon Debug EQ Control 2	0004254$_H$	P	P	Kernel Reset	3428
PCIE1_DSP_SD_EQ_CONTROL3_REG	Silicon Debug EQ Control 3	0004258$_H$	P	P	Kernel Reset	3429
PCIE1_DSP_SD_EQ_STATUS1_REG	Silicon Debug EQ Status 1	0004260$_H$	P	P	Kernel Reset	3430
PCIE1_DSP_SD_EQ_STATUS2_REG	Silicon Debug EQ Status 2	0004264$_H$	P	P	Kernel Reset	3431
PCIE1_DSP_SD_EQ_STATUS3_REG	Silicon Debug EQ Status 3	0004268$_H$	P	P	Kernel Reset	3432

(table continues...)

Table 929 (continued) Register overview - PCIE1_DSP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE1_DSP_RASDP_EXT_CAP_HDR_OFF	PCIe Extended capability ID, Capability version and Next capability offset	0004280H	P	P	Kernel Reset	3433
PCIE1_DSP_RASDP_VENDOR_SPECIFIC_HDR_OFF	Vendor Specific Header	0004284H	P	P	Kernel Reset	3434
PCIE1_DSP_RASDP_ERROR_PROT_CTRL_OFF	ECC Error Correction and Control Register	0004288H	P	P	Kernel Reset	3435
PCIE1_DSP_RASDP_ERROR_MODE_EN_OFF	RASDP error mode enable.	00042A8H	P	P	Kernel Reset	3447
PCIE1_DSP_RASDP_ERROR_MODE_CLEAR_OFF	RASDP Error Mode Clear Register	00042ACH	P	P	Kernel Reset	3448
PCIE1_DSP_RASDP_RAM_ADDR_CORR_ERROR_OFF	RAM Address where a corrected error (1-bit ECC) is detected	00042B0H	P	P	Kernel Reset	3449
PCIE1_DSP_RASDP_RAM_ADDR_UNCORR_ERROR_OFF	RAM Address where an uncorrected error (2-bit ECC) is detected	00042B4H	P	P	Kernel Reset	3449
PCIE1_DSP_PTM_EXT_CAP_HDR_OFF	Precision Time Measurement Capability Header	00042B8H	P	P	Kernel Reset	3450
PCIE1_DSP_PTM_CAP_OFF	PTM Capability Register	00042BCH	P	P	Kernel Reset	3451
PCIE1_DSP_PTM_CONTROL_OFF	PTM Control Register	00042C0H	P	P	Kernel Reset	3452
PCIE1_DSP_PTM_RES_CAP_HDR_OFF	Precision Time Measurement Responder Capability Header (VSEC)	00042C4H	P	P	Kernel Reset	3453
PCIE1_DSP_PTM_RES_HDR_OFF	Precision Time Measurement Responder Vendor Specific Header	00042C8H	P	P	Kernel Reset	3454
PCIE1_DSP_PTM_RES_CONTROL_OFF	PTM Responder Control Register	00042CCH	P	P	Kernel Reset	3455
PCIE1_DSP_PTM_RES_STATUS_OFF	PTM Responder Status Register	00042D0H	P	P	Kernel Reset	3456
PCIE1_DSP_PTM_RES_LOCAL_LSB_OFF	PTM Responder Local Clock LSB	00042D4H	P	P	Kernel Reset	3457

(table continues...)

Table 929 (continued) Register overview - PCIE1_DSP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE1_DSP_PTM_RES_LOCAL_MSB_OFF	PTM Responder Local Clock MSB	00042D8H	P	P	Kernel Reset	3457
PCIE1_DSP_PTM_RES_T2_LSB_OFF	PTM Responder T2 Timestamp LSB	00042DCH	P	P	Kernel Reset	3458
PCIE1_DSP_PTM_RES_T2_MSB_OFF	PTM Responder T2 Timestamp MSB	00042E0H	P	P	Kernel Reset	3458
PCIE1_DSP_PTM_RES_T2P_LSB_OFF	PTM Responder T2 Previous Timestamp LSB	00042E4H	P	P	Kernel Reset	3458
PCIE1_DSP_PTM_RES_T2P_MSB_OFF	PTM Responder T2 Previous Timestamp MSB	00042E8H	P	P	Kernel Reset	3459
PCIE1_DSP_PTM_RES_T3_LSB_OFF	PTM Responder T3 Timestamp LSB	00042ECH	P	P	Kernel Reset	3459
PCIE1_DSP_PTM_RES_T3_MSB_OFF	PTM Responder T3 Timestamp MSB	00042F0H	P	P	Kernel Reset	3460
PCIE1_DSP_PTM_RES_T3P_LSB_OFF	PTM Responder T3 Previous Timestamp LSB	00042F4H	P	P	Kernel Reset	3460
PCIE1_DSP_PTM_RES_T3P_MSB_OFF	PTM Responder T3 Previous Timestamp MSB	00042F8H	P	P	Kernel Reset	3461
PCIE1_DSP_PTM_RES_TX_LATENCY_OFF	PTM Responder TX Latency	00042FCH	P	P	Kernel Reset	3461
PCIE1_DSP_PTM_RES_RX_LATENCY_OFF	PTM Responder RX Latency	0004300H	P	P	Kernel Reset	3462
PCIE1_DSP_PTM_RES_NOM_CLOCK_T_OFF	PTM Responder Nominal Clock Period	0004304H	P	P	Kernel Reset	3463
PCIE1_DSP_PTM_RES_SCALED_CLOCK_T_OFF	PTM Responder Scaled Clock Period	0004308H	P	P	Kernel Reset	3463
PCIE1_DSP_PTM_RES_LATENCY_REG_SEL_OFF	PTM Responder Latency Viewport Register Select	000430CH	P	P	Kernel Reset	3464

(table continues...)

Table 929 (continued) Register overview - PCIE1_DSP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE1_DSP_VSECDMA_EXT_CAP_HDR_OFF	PCIe Extended Capability ID, Capability Version, and Next Capability Offset Register	000432C H	P	P	Kernel Reset	3465
PCIE1_DSP_VSECDMA_VENDOR_SPECIFIC_HDR_OFF	Vendor Specific Header Register	0004330 H	P	P	Kernel Reset	3466
PCIE1_DSP_VSECDMA_DEVICE_INFORMATION_OFF	DMA and related AXI Bridge Implementation Information	0004334 H	P	P	Kernel Reset	3467
PCIE1_DSP_VSECDMA_NUM_CHAN_OFF	Number of Implemented Channels Register	0004338 H	P	P	Kernel Reset	3469
PCIE1_DSP_VSECDMA_UNROLL_ADDR_OFFSET_LOW_OFF	DMA Register Map Start Address Offset Low Register	000433C H	P	P	Kernel Reset	3469
PCIE1_DSP_VSECDMA_UNROLL_ADDR_OFFSET_HIGH_OFF	DMA Register Map Start Address Offset High Register	0004340 H	P	P	Kernel Reset	3470
PCIE1_DSP_ACK_LATENCY_TIMER_OFF	Ack Latency Timer and Replay Timer Register	0004700 H	P	P	Kernel Reset	3470
PCIE1_DSP_VENDOR_SPEC_DLLP_OFF	Vendor Specific DLLP Register	0004704 H	P	P	Kernel Reset	3471
PCIE1_DSP_PORT_FORCE_OFF	Port Force Link Register	0004708 H	P	P	Kernel Reset	3471
PCIE1_DSP_ACK_F_ASPM_CTRL_OFF	Ack Frequency and L0-L1 ASPM Control Register	000470C H	P	P	Kernel Reset	3473
PCIE1_DSP_PORT_LINK_CTRL_OFF	Port Link Control Register	0004710 H	P	P	Kernel Reset	3475
PCIE1_DSP_LANE_SKEW_OFF	Lane Skew Register	0004714 H	P	P	Kernel Reset	3477
PCIE1_DSP_TIMER_CTRL_MAX_FUNC_NUM_OFF	Timer Control and Max Function Number Register	0004718 H	P	P	Kernel Reset	3479
PCIE1_DSP_SYMBOL_TIMER_FILTER_1_OFF	Symbol Timer Register and Filter Mask 1 Register	000471C H	P	P	Kernel Reset	3480

(table continues...)

Table 929 (continued) Register overview - PCIE1_DSP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE1_DSP_FILTER_MASK_2_OFF	Filter Mask 2 Register	0004720H	P	P	Kernel Reset	3483
PCIE1_DSP_PL_DEBUG0_OFF	Debug Register 0	0004728H	P	P	Kernel Reset	3485
PCIE1_DSP_PL_DEBUG1_OFF	Debug Register 1	000472CH	P	P	Kernel Reset	3486
PCIE1_DSP_TX_P_FC_CREDIT_STATUS_OFF	Transmit Posted FC Credit Status	0004730H	P	P	Kernel Reset	3487
PCIE1_DSP_TX_NP_FC_CREDIT_STATUS_OFF	Transmit Non-Posted FC Credit Status	0004734H	P	P	Kernel Reset	3488
PCIE1_DSP_TX_CPL_FC_CREDIT_STATUS_OFF	Transmit Completion FC Credit Status	0004738H	P	P	Kernel Reset	3489
PCIE1_DSP_QUEUE_STATUS_OFF	Queue Status	000473CH	P	P	Kernel Reset	3490
PCIE1_DSP_VC0_P_RX_Q_CTRL_OFF	Segmented-Buffer VC0 Posted Receive Queue Control	0004748H	P	P	Kernel Reset	3492
PCIE1_DSP_VC0_NP_RX_Q_CTRL_OFF	Segmented-Buffer VC0 Non-Posted Receive Queue Control	000474CH	P	P	Kernel Reset	3494
PCIE1_DSP_VC0_CPL_RX_Q_CTRL_OFF	Segmented-Buffer VC0 Completion Receive Queue Control	0004750H	P	P	Kernel Reset	3495
PCIE1_DSP_GEN2_CTRL_OFF	Link Width and Speed Change Control Register	000480CH	P	P	Kernel Reset	3496
PCIE1_DSP_MSI_CTRL_ADDR_OFF	Integrated MSI Reception Module (iMRM) Address Register	0004820H	P	P	Kernel Reset	3500
PCIE1_DSP_MSI_CTRL_UPPER_ADDR_OFF	Integrated MSI Reception Module Upper Address Register	0004824H	P	P	Kernel Reset	3501
PCIE1_DSP_MSI_CTRL_INT_i_EN_OFF (i=0-7)	Integrated MSI i Reception Module Interrupt0 Enable Register	0004828H+i*12	P	P	Kernel Reset	3501
PCIE1_DSP_MSI_CTRL_INT_i_MASK_OFF (i=0-7)	Integrated MSI i Reception Module Interrupt0 Mask Register	000482CH+i*12	P	P	Kernel Reset	3502

(table continues...)

Table 929 (continued) Register overview - PCIE1_DSP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE1_DSP_MSI_CTRL_INT_i_STATUS_OFF (i=0-7)	Integrated MSI i Reception Module Interrupt0 Status Register	0004830$_H$+i*12	P	P	Kernel Reset	3502
PCIE1_DSP_MSI_GPIO_IO_OFF	Integrated MSI Reception Module General Purpose IO Register	0004888$_H$	P	P	Kernel Reset	3503
PCIE1_DSP_CLOCK_GATING_CTRL_OFF	Clock Gating Control Register	000488C$_H$	P	P	Kernel Reset	3503
PCIE1_DSP_GEN3_RELATED_OFF	Gen3 Control Register	0004890$_H$	P	P	Kernel Reset	3504
PCIE1_DSP_GEN3_EQ_CONTROL_OFF	Gen3 EQ Control Register	00048A8$_H$	P	P	Kernel Reset	3507
PCIE1_DSP_GEN3_EQ_FB_MODE_DIR_CHANGE_OFF	Gen3 EQ Direction Change Feedback Mode Control Register	00048AC$_H$	P	P	Kernel Reset	3510
PCIE1_DSP_ORDER_RULE_CTRL_OFF	Order Rule Control Register	00048B4$_H$	P	P	Kernel Reset	3511
PCIE1_DSP_PIPE_LOOPBACK_CONTROL_OFF	PIPE Loopback Control Register	00048B8$_H$	P	P	Kernel Reset	3512
PCIE1_DSP_MISC_CONTROL_1_OFF	DBI Read-Only Write Enable Register	00048BC$_H$	P	P	Kernel Reset	3513
PCIE1_DSP_LINK_FLUSH_CONTROL_OFF	Link Reset Request Flush Control Register	00048CC$_H$	P	P	Kernel Reset	3519
PCIE1_DSP_AMBA_ERROR_RESPONSE_DEFAULT_OFF	AXI Bridge Slave Error Response Register	00048D0$_H$	P	P	Kernel Reset	3520
PCIE1_DSP_AMBA_LINK_TIMEOUT_OFF	Link Down AXI Bridge Slave Timeout Register	00048D4$_H$	P	P	Kernel Reset	3521
PCIE1_DSP_AMBA_ORDERING_CTRL_OFF	AXI Bridge Ordering Control	00048D8$_H$	P	P	Kernel Reset	3522
PCIE1_DSP_PCIE_VERSION_NUMBER_OFF	PCIe Controller IIP Release Version Number	00048F8$_H$	P	P	Kernel Reset	3526

(table continues...)

Table 929 (continued) Register overview - PCIE1_DSP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE1_DSP_PCIE_VERSION_TYPE_OFF	PCIe Controller IIP Release Version Type	00048FC H	P	P	Kernel Reset	3527
PCIE1_DSP_INTERFACE_TIMER_CONTROL_OFF	Interface Timer Control Register	0004930 H	P	P	Kernel Reset	3527
PCIE1_DSP_INTERFACE_TIMER_TARGET_OFF	Interface Timer Target Register	0004934 H	P	P	Kernel Reset	3528
PCIE1_DSP_INTERFACE_TIMER_STATUS_OFF	Interface Timer Status Register	0004938 H	P	P	Kernel Reset	3529
PCIE1_DSP_MSIX_ADDRESS_MATCH_LOW_OFF	MSI-X Address Match Low Register	0004940 H	P	P	Kernel Reset	3530
PCIE1_DSP_MSIX_ADDRESS_MATCH_HIGH_OFF	MSIX Address Match High Register	0004944 H	P	P	Kernel Reset	3531
PCIE1_DSP_MSIX_DOORBELL_OFF	MSI-X Doorbell Register	0004948 H	P	P	Kernel Reset	3531
PCIE1_DSP_SAFETY_MASK_OFF	Functional Safety Interrupt Events Mask Register	0004960 H	P	P	Kernel Reset	3534
PCIE1_DSP_SAFETY_STATUS_OFF	Status for functional safety interrupt events	0004964 H	P	P	Kernel Reset	3535
PCIE1_DSP_PL_APP_BUS_DEV_NUM_STATUS_OFF	Application driven bus and device number register	0004B10 H	P	P	Kernel Reset	3536
PCIE1_DSP_PL_CHK_REG_CONTROL_STATUS_OFF	CDM Register Checking Control and Status Register	0004B20 H	P	P	Kernel Reset	3537
PCIE1_DSP_PL_CHK_REG_START_END_OFF	CDM Register Checking First and Last address to check	0004B24 H	P	P	Kernel Reset	3539
PCIE1_DSP_PL_CHK_REG_ERR_ADDR_OFF	CDM Register Checking Error Address	0004B28 H	P	P	Kernel Reset	3539
PCIE1_DSP_PL_CHK_REG_ERR_PF_VF_OFF	CDM Register Checking error PF Numbers	0004B2C H	P	P	Kernel Reset	3540
PCIE1_DSP_PL_LTR_LATENCY_OFF	LTR Latency Register	0004B30 H	P	P	Kernel Reset	3540

(table continues...)

Table 929 (continued) Register overview - PCIE1_DSP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE1_DSP_AUX_CLK_FREQ_OFF	Auxiliary Clock Frequency Control Register	0004B40$_H$	P	P	Kernel Reset	3541
PCIE1_DSP_POWERDOWN_CTRL_STATUS_OFF	Powerdown Control and Status Register	0004B48$_H$	P	P	Kernel Reset	3543
PCIE1_DSP_MSIXPBAx_DWORD (x=0-1)	MSIX x table DWORD	0005E10$_H$+x*4	P	P	Kernel Reset	3545
PCIE1_DSP_SHADOW_PCI_MSIX_CAP_ID_NEXT_CTRL_REG	MSI-X Capability ID, Next Pointer, Control Register	00060B0$_H$	P	P	Kernel Reset	3545
PCIE1_DSP_DMA_CTRL_DATA_ARB_PRIOR_OFF	DMA Arbitration Scheme for TRGT1 Interface	0007000$_H$	P	P	Kernel Reset	3547
PCIE1_DSP_DMA_CTRL_OFF	DMA Number of Channels Register	0007008$_H$	P	P	Kernel Reset	3548
PCIE1_DSP_DMA_WRITE_ENGINE_EN_OFF	DMA Write Engine Enable Register	000700C$_H$	P	P	Kernel Reset	3549
PCIE1_DSP_DMA_WRITE_DOORBELL_OFF	DMA Write Doorbell Register	0007010$_H$	P	P	Kernel Reset	3551
PCIE1_DSP_DMA_WRITE_CHANNEL_ARB_WEIGHT_LOW_OFF	DMA Write Engine Channel Arbitration Weight Low Register	0007018$_H$	P	P	Kernel Reset	3552
PCIE1_DSP_DMA_READ_ENGINE_EN_OFF	DMA Read Engine Enable Register	000702C$_H$	P	P	Kernel Reset	3554
PCIE1_DSP_DMA_READ_DOORBELL_OFF	DMA Read Doorbell Register	0007030$_H$	P	P	Kernel Reset	3555
PCIE1_DSP_DMA_READ_CHANNEL_ARB_WEIGHT_LOW_OFF	DMA Read Engine Channel Arbitration Weight Low Register	0007038$_H$	P	P	Kernel Reset	3556
PCIE1_DSP_DMA_WRITE_INT_STATUS_OFF	DMA Write Interrupt Status Register	000704C$_H$	P	P	Kernel Reset	3558
PCIE1_DSP_DMA_WRITE_INT_MASK_OFF	DMA Write Interrupt Mask Register	0007054$_H$	P	P	Kernel Reset	3559

(table continues...)

Table 929 (continued) Register overview - PCIE1_DSP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE1_DSP_DMA_WRITE_INT_CLEAR_OFF	DMA Write Interrupt Clear Register	0007058H	P	P	Kernel Reset	3560
PCIE1_DSP_DMA_WRITE_ERR_STATUS_OFF	DMA Write Error Status Register	000705CH	P	P	Kernel Reset	3561
PCIE1_DSP_DMA_WRITE_DONE_IMWR_LOW_OFF	DMA Write Done IMWr Address Low Register	0007060H	P	P	Kernel Reset	3561
PCIE1_DSP_DMA_WRITE_DONE_IMWR_HIGH_OFF	DMA Write Done IMWr Interrupt Address High Register	0007064H	P	P	Kernel Reset	3562
PCIE1_DSP_DMA_WRITE_ABORT_IMWR_LOW_OFF	DMA Write Abort IMWr Address Low Register	0007068H	P	P	Kernel Reset	3562
PCIE1_DSP_DMA_WRITE_ABORT_IMWR_HIGH_OFF	DMA Write Abort IMWr Address High Register	000706CH	P	P	Kernel Reset	3563
PCIE1_DSP_DMA_WRITE_CH01_IMWR_DATA_OFF	DMA Write Channel 1 and 0 IMWr Data Register	0007070H	P	P	Kernel Reset	3563
PCIE1_DSP_DMA_WRITE_CH23_IMWR_DATA_OFF	DMA Write Channel 3 and 2 IMWr Data Register	0007074H	P	P	Kernel Reset	3564
PCIE1_DSP_DMA_WRITE_LINKED_LIST_ERR_EN_OFF	DMA Write Linked List Error Enable Register	0007090H	P	P	Kernel Reset	3565
PCIE1_DSP_DMA_READ_INT_STATUS_OFF	DMA Read Interrupt Status Register	00070A0H	P	P	Kernel Reset	3566
PCIE1_DSP_DMA_READ_INT_MASK_OFF	DMA Read Interrupt Mask Register	00070A8H	P	P	Kernel Reset	3567
PCIE1_DSP_DMA_READ_INT_CLEAR_OFF	DMA Read Interrupt Clear Register	00070ACH	P	P	Kernel Reset	3568
PCIE1_DSP_DMA_READ_ERR_STATUS_LOW_OFF	DMA Read Error Status Low Register	00070B4H	P	P	Kernel Reset	3569
PCIE1_DSP_DMA_READ_ERR_STATUS_HIGH_OFF	DMA Read Error Status High Register	00070B8H	P	P	Kernel Reset	3570

(table continues...)

Table 929 (continued) Register overview - PCIE1_DSP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE1_DSP_DMA_READ_LINKED_LIST_ERR_EN_OFF	DMA Read Linked List Error Enable Register	00070C4$_H$	P	P	Kernel Reset	3571
PCIE1_DSP_DMA_READ_DONE_IMWR_LOW_OFF	DMA Read Done IMWr Address Low Register	00070CC$_H$	P	P	Kernel Reset	3572
PCIE1_DSP_DMA_READ_DONE_IMWR_HIGH_OFF	DMA Read Done IMWr Address High Register	00070D0$_H$	P	P	Kernel Reset	3572
PCIE1_DSP_DMA_READ_ABORT_IMWR_LOW_OFF	DMA Read Abort IMWr Address Low Register	00070D4$_H$	P	P	Kernel Reset	3573
PCIE1_DSP_DMA_READ_ABORT_IMWR_HIGH_OFF	DMA Read Abort IMWr Address High Register	00070D8$_H$	P	P	Kernel Reset	3573
PCIE1_DSP_DMA_READ_CH01_IMWR_DATA_OFF	DMA Read Channel 1 and 0 IMWr Data Register	00070DC$_H$	P	P	Kernel Reset	3574
PCIE1_DSP_DMA_READ_CH23_IMWR_DATA_OFF	DMA Read Channel 3 and 2 IMWr Data Register	00070E0$_H$	P	P	Kernel Reset	3574
PCIE1_DSP_DMA_WRITE_ENGINE_HSHAKE_CNT_LOW_OFF	DMA Write Engine Handshake Counter Channel 0/1/2/3 Register	0007108$_H$	P	P	Kernel Reset	3576
PCIE1_DSP_DMA_READ_ENGINE_HSHAKE_CNT_LOW_OFF	DMA Read Engine Handshake Counter Channel 0/1/2/3 Register	0007118$_H$	P	P	Kernel Reset	3577
PCIE1_DSP_DMA_WRITE_CHi_PWR_EN_OFF (i=0-3)	DMA Write Channel i Power Enable Register	0007128$_H$+i*4	P	P	Kernel Reset	3579
PCIE1_DSP_DMA_READ_CHi_PWR_EN_OFF (i=0-3)	DMA Read Channel i Power Enable Register	0007168$_H$+i*4	P	P	Kernel Reset	3579
PCIE1_DSP_DMA_CH_CONTROL1_OFF_WRCH_i (i=0-3)	DMA Write Channel i Control 1 Register	0007200$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3580

(table continues...)

Table 929 (continued) Register overview - PCIE1_DSP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE1_DSP_DMA_TRANSFER_SIZE_OFF_WRCH_i (i=0-3)	DMA CH i Write Transfer Size Register	0007208$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3583
PCIE1_DSP_DMA_SAR_LOW_OFF_WRCH_i (i=0-3)	DMA CH i Write SAR Low Register	000720C$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3583
PCIE1_DSP_DMA_SAR_HIGH_OFF_WRCH_i (i=0-3)	DMA CH i Write SAR High Register	0007210$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3584
PCIE1_DSP_DMA_DAR_LOW_OFF_WRCH_i (i=0-3)	DMA CH i Write DAR Low Register	0007214$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3584
PCIE1_DSP_DMA_DAR_HIGH_OFF_WRCH_i (i=0-3)	DMA CH i Write DAR High Register	0007218$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3585
PCIE1_DSP_DMA_LLP_LOW_OFF_WRCH_i (i=0-3)	DMA CH i Write Linked List Pointer Low Register	000721C$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3585
PCIE1_DSP_DMA_LLP_HIGH_OFF_WRCH_i (i=0-3)	DMA CH i Write Linked List Pointer High Register	0007220$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3586
PCIE1_DSP_DMACAP_CHi_WRCHN_WR_INT_CLEAR (i=0-3)	Interrupt clear for DMA write channel i	0007224$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3586
PCIE1_DSP_DMACAP_CHi_WRCHN_WR_INT_STATUS (i=0-3)	Interrupt status for DMA write channel i	0007228$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3587
PCIE1_DSP_DMACAP_CHi_WRCHN_WR_ERR_STATUS (i=0-3)	Error status for DMA write channel i	000722C$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3588

(table continues...)

Table 929 (continued) Register overview - PCIE1_DSP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE1_DSP_DMA_CAP_CHi_WRCHN_WR_DOORBELL (i=0-3)	Doorbell for DMA write channel i	0007230$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3589
PCIE1_DSP_DMA_CH_CONTROL1_OFF_RDCH_i (i=0-3)	DMA Read Channel i Control 1 Register	0007300$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3589
PCIE1_DSP_DMA_TRANSFER_SIZE_OFF_RDCH_i (i=0-3)	DMA Read CH i Transfer Size Register	0007308$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3592
PCIE1_DSP_DMA_SAR_LOW_OFF_RDCH_i (i=0-3)	DMA Read CH i SAR Low Register	000730C$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3593
PCIE1_DSP_DMA_SAR_HIGH_OFF_RDCH_i (i=0-3)	DMA Read CH i SAR High Register	0007310$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3594
PCIE1_DSP_DMA_DAR_LOW_OFF_RDCH_i (i=0-3)	DMA Read CH i DAR Low Register	0007314$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3594
PCIE1_DSP_DMA_DAR_HIGH_OFF_RDCH_i (i=0-3)	DMA Read CH i DAR High Register	0007318$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3595
PCIE1_DSP_DMA_LLP_LOW_OFF_RDCH_i (i=0-3)	DMA Read CH i Linked List Pointer Low Register	000731C$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3595
PCIE1_DSP_DMA_LLP_HIGH_OFF_RDCH_i (i=0-3)	DMA CH i Read Linked List Pointer High Register	0007320$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3596
PCIE1_DSP_DMA_CAP_CHi_RDCHN_RD_INT_CLEAR (i=0-3)	Interrupt clear for DMA read channel i	0007324$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3596

(table continues...)

Table 929 (continued) Register overview - PCIE1_DSP domain FPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE1_DSP_DMACAP_CHi_RDCHN_RD_INT_STATUS (i=0-3)	Interrupt status for DMA read channel i	0007328$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3597
PCIE1_DSP_DMACAP_CHi_RDCHN_RD_ERR_STATUS (i=0-3)	Error status for DMA read channel i	000732C$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3597
PCIE1_DSP_DMACAP_CHi_RDCHN_RD_ERR_STATUS_HIGH (i=0-3)	Higher error status for DMA read channel i	0007330$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3598
PCIE1_DSP_DMACAP_CHi_RDCHN_RD_DOORBELL (i=0-3)	Doorbell for DMA read channel i	0007334$_H$+i*200$_H$	P_DMAi	P_DMAi	Kernel Reset	3599

19.8.4.14 Register overview - PCIE0_DSP_SRI domain SRI (ascending offset address)

Table 930 Register overview - PCIE0_DSP_SRI domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE0_DSP_S2A_ACCENy_WRA (y=0-7)	Write access enable register A y	3FFFE00$_H$+y*20$_H$	U	SE, SV	Application Reset	3600
PCIE0_DSP_S2A_ACCENy_WRB (y=0-7)	Write access enable register B y	3FFFE04$_H$+y*20$_H$	U	SE, SV	Application Reset	3601
PCIE0_DSP_S2A_ACCENy_RDA (y=0-7)	Read access enable register A y	3FFFE08$_H$+y*20$_H$	U	SE, SV	Application Reset	3601
PCIE0_DSP_S2A_ACCENy_RDB (y=0-7)	Read access enable register B y	3FFFE0C$_H$+y*20$_H$	U	SE, SV	Application Reset	3602
PCIE0_DSP_S2A_ACCENy_VM (y=0-7)	VM access enable register y	3FFFE10$_H$+y*20$_H$	U	SE, SV	Application Reset	3602

(table continues...)

Table 930 (continued) Register overview - PCIE0_DSP_SRI domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE0_DSP_S2A_ACCENy_PRS (y=0-7)	PRS access enable register y	3FFFE14$_H$+y*20$_H$	U	SE, SV	Application Reset	3603
PCIE0_DSP_S2A_ACCENy_RGNLA (y=0-7)	Region lower address register y	3FFFE18$_H$+y*20$_H$	U	SE, SV	Application Reset	3603
PCIE0_DSP_S2A_ACCENy_RGNUA (y=0-7)	Region upper address register y	3FFFE1C$_H$+y*20$_H$	U	SE, SV	Application Reset	3604
PCIE0_DSP_S2A_PROTSE	PROT Register Safe Endinit	3FFFF04$_H$	U	SV	Application Reset	3604

19.8.4.15 Register overview - PCIE1_DSP_SRI domain SRI (ascending offset address)

Table 931 Register overview - PCIE1_DSP_SRI domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE1_DSP_S2A_ACCENy_WRA (y=0-7)	Write access enable register A y	3FFFE00$_H$+y*20$_H$	U	SE, SV	Application Reset	3600
PCIE1_DSP_S2A_ACCENy_WRB (y=0-7)	Write access enable register B y	3FFFE04$_H$+y*20$_H$	U	SE, SV	Application Reset	3601
PCIE1_DSP_S2A_ACCENy_RDA (y=0-7)	Read access enable register A y	3FFFE08$_H$+y*20$_H$	U	SE, SV	Application Reset	3601
PCIE1_DSP_S2A_ACCENy_RDB (y=0-7)	Read access enable register B y	3FFFE0C$_H$+y*20$_H$	U	SE, SV	Application Reset	3602
PCIE1_DSP_S2A_ACCENy_VM (y=0-7)	VM access enable register y	3FFFE10$_H$+y*20$_H$	U	SE, SV	Application Reset	3602
PCIE1_DSP_S2A_ACCENy_PRS (y=0-7)	PRS access enable register y	3FFFE14$_H$+y*20$_H$	U	SE, SV	Application Reset	3603
PCIE1_DSP_S2A_ACCENy_RGNLA (y=0-7)	Region lower address register y	3FFFE18$_H$+y*20$_H$	U	SE, SV	Application Reset	3603
PCIE1_DSP_S2A_ACCENy_RGNUA (y=0-7)	Region upper address register y	3FFFE1C$_H$+y*20$_H$	U	SE, SV	Application Reset	3604

(table continues...)

Table 931 (continued) Register overview - PCIE1_DSP_SRI domain SRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PCIE1_DSP_S2A_PROTSE	PROT Register Safe Endinit	3FFFF04H	U	SV	Application Reset	3604

19.8.4.16 Device specific registers

There are no device specific register changes.

19.8.5 TC4Dx PCIE connectivity

The table below lists interfaces to and from this functional block to other blocks in the device.

Table 932 List of PCIE interface signals

Interface signals	I/O	Description
FPI_PCIE_SIF	In	SPB interface
PCIE_SRI_MIF	Out	SRI master interface
SRI_PCIE_SIF	In	SRI slave interface
PCIE_IR_SR[21:0]	Out	PCIE service requests
PORTS_PCIE_PERST#	In	The PCIE PERST# sideband pin
PORTS_PCIE_WAKE#	In	The PCIE WAKE# sideband pin
PCIE_PORTS_WAKE#	Out	The PCIE WAKE# sideband pin
PCIE_PORTS_CLKREQ#	Out	The PCIE CLKREQ# sideband pin
PORTS_PCIE_CLKREQ#	In	The PCIE CLKREQ# sideband pin
HSPHY_PCIE_RawPCS	In	The interface between the PCIE.UPCS and the HSPHY
PCIE_HSPHY_RawPCS	Out	The interface between the PCIE.UPCS and the HSPHY
HSPHY_PCIE_fPHY	In	The interface between the PCIE.UPCS and the HSPHY
PCIE_TRACE_OTGB[0:1]	Out	Debug related signals
CLOCK_PCIE_fSPB	In	SPB clock input
CLOCK_PCIE_fSRI	In	SRI clock input
CLOCK_PCIE_fOSC	In	PCIE auxiliary clock, xtal clock
PCIE_SMU_ALARMS	Out	PCIE alarms

19.8.6 TC4Dx PCIe revision history

Reference	Description of change(s)
Date range: 2024-08-17 to 2025-01-31	
TC4Dx specific UPCS override settings for common refclk PHY settings	UPCS override information required for common reference clock PHY configuration

20 Data Routing Engine (DRE)

The DRE is a hardware accelerator used to route Controller Area Network (CAN) frames and Ethernet frames.

The DRE routes CAN frames between Ethernet interfaces. Additionally, it can assist in routing CAN frames from one CAN interface to another CAN interface or to a user-configured memory location, and in forwarding Ethernet frames between Ethernet interfaces.

The DRE is aligned to the AVTP Control frame format defined by IEEE 1722-2016.

In this chapter the following notations from IEEE 1722-2016 are used:

- AVTP frame refers to an Ethernet frame containing Audio Video Transport Protocol
- ACF (AVTP Control Format) frame refers to an AVTP frame containing control frames
- ACF_CAN_BRIEF message refers to CAN frames contained within an ACF frame

  The following chapters describe the core elements of the DRE, their role in routing/forwarding and the configuration possibilities

20.1 Feature list

- Routes Controller Area Network (CAN) frames to or from Ethernet interfaces using IEEE 1722 ACF frames
  - Supports different Ethernet transmit trigger modes
  - User-configured Ethernet MAC Header (Layer 2) for ACF transmit frames
  - Automatic Ethernet DMA transmit and receive descriptor handling
  - Configurable AVTP stream-ID
  - CAN-ID based CAN interface destination search
  - Filtering of CAN frames tunneled over Ethernet
  - User-configured CAN Routing Table to identify the destination CAN interface for a CAN frame tunneled over Ethernet
- Assists CAN Routing Engine (CRE) to perform inter-MCMCAN transfer of CAN frames from a CAN interface to another CAN interface
- Assists CRE to perform routing of received CAN frame(s) or CAN I-PDU(s) from a CAN interface to a configurable internal memory address location
  - Automatic memory address increment and wrap around
  - User-configured watermark interrupt generation for CPU notification
- Supports 1:1 uni-cast routing and 1:4 multi-cast Ethernet to CAN routing
- Provides non-starving arbitration between routing transfer requests
- Supports 1:1 uni-cast or 1:6 multi-cast forwarding of Ethernet frames between interfaces
- Provides non-starving arbitration between routing and forwarding requests
- Provides error monitoring mechanisms for accelerated detection of faults

20.2 Functional overview

The DRE block diagram is shown in the following figure:

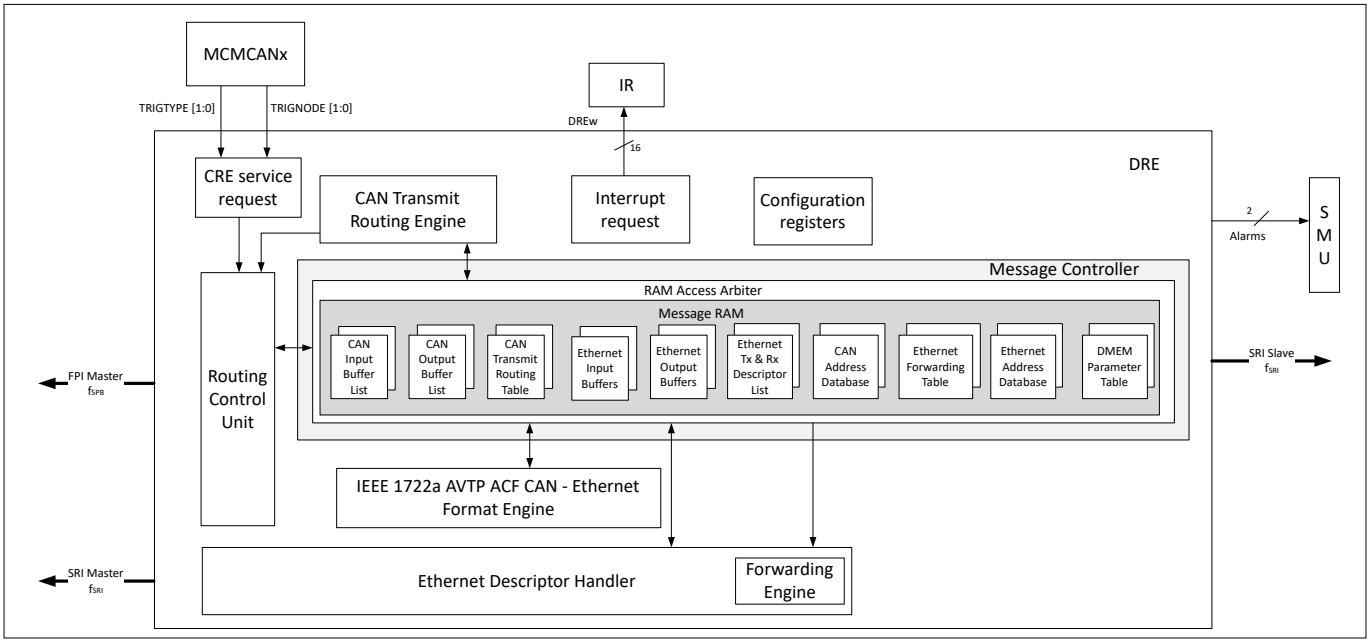


Figure 329 DRE block diagram

For the alarm description please refer to the alarm mapping tables in the SMU functional block and the Safety Manual.

- The DRE contains a central message storage RAM which buffers the CAN frames and the Ethernet frames
- The Routing Control Unit assists in the transfer of received CAN frames from the MCMCAN module to the internal message RAM. The Routing Control Unit also transfers the CAN frame to the identified CAN interface for transmission
- The ACF CAN - Ethernet Format Engine performs the translation of CAN frames to ACF frame format and vice versa
- The CAN Transmit Routing Engine, along with a user-configured Routing Table, decides the destination CAN interface from which a CAN frame has to be transmitted. The Routing Control Unit also transfers the CAN frame to the identified CAN interface for transmission
- The Ethernet Descriptor Handler maintains the Ethernet Transmit (Tx) and Receive (Rx) descriptors which will then be used by Ethernet DMA or the software to transmit or receive Ethernet frames
- The Forwarding Engine, along with a user-configured Forwarding Table, decides the destination Ethernet interface(s) to which the Ethernet frame has to be forwarded

20.3 Functional description

The following sections describe in detail the functions of the DRE.

20.4 Message controller

The Message Controller manages the CAN and Ethernet frame buffers in the Message RAM and arbitrates the RAM accesses.

The image in the chapter Functional overview below shows the Message Controller relative to the DRE Message RAM.

The Message RAM stores control information and CAN and Ethernet data. The SRI Slave accesses are always given the highest priority followed by the MCMCAN module accesses. All the RAM accesses by other control logic

of the DRE are arbitrated with equal priority. Read and write accesses to the Message RAM are of up to 64-bit data width.

20.4.1 Feature list

- Message RAM is of size 32 Kbytes (4 kwords x 64 bits) for buffering frame data and control information as follows:
  - CAN frames received from CAN interface - CAN Input Buffer List to buffer maximum of 20 CAN frames (as there are a total of 20 CAN interfaces)
  - CAN Address Database for address and control configuration of CAN Receive Host Buffers and CAN Transmit Host Buffers
  - Ethernet Address Database for address configuration of LETH and GETH DMA channels
  - Ethernet Input Buffers having 192 double words each can be used to store Ethernet frames received from Ethernet interfaces
  - CAN frames to be transmitted to CAN interface in CAN Output Buffer List; stores a maximum of 64 CAN frames
  - Ethernet Output Buffers having 192 double words each can be used to store Ethernet frames received from Ethernet interfaces
  - CAN Transmit Routing Tables (1 to 4); each routing table can accommodate a maximum of 128 routing rules
  - Ethernet Transmit (Tx) and Receive (Rx) Descriptor List
  - Ethernet Forwarding Table with a maximum of 128 forwarding rules
  - DMEM Parameter Table that consists of configuration of 28 memory destinations
- Management of message buffers and routing control information
- Allows direct access of Message RAM for application software through SRI Slave interface for monitoring
- Arbitration of RAM accesses

20.4.2 Functional overview

This chapter describes the contents of the DRE Message RAM.

Message RAM

The contents of the Message RAM and its interfaces are shown in the figure below.

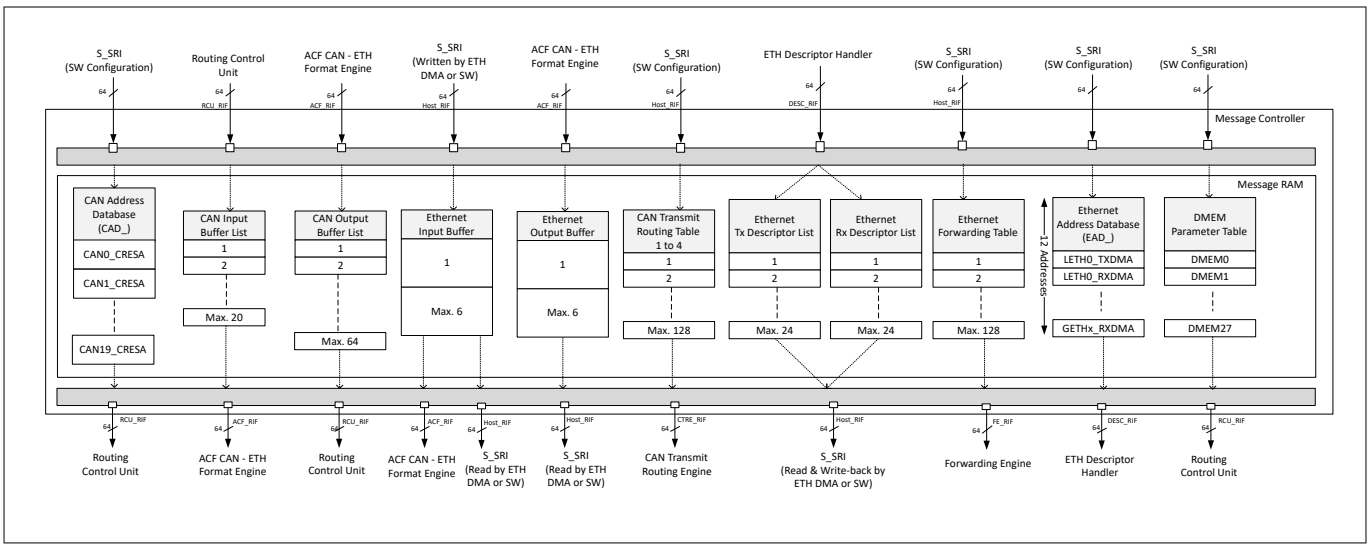


Figure 330 Message RAM block diagram

20 Data Routing Engine (DRE)

The Message RAM contains the following buffers and control information:

- CAN Address Database (CAD)
- CAN Input Buffer List (CIBL)
- CAN Output Buffer List (COBL)
- Ethernet Input Buffers (EIBUF)
- Ethernet Output Buffers (EOBUF)
- CAN Transmit Routing Tables (1 to 4)
- Ethernet Descriptor Lists (Tx and Rx)
- Ethernet Forwarding Table
- Ethernet Address Database (EAD)
- DMEM Parameter Table

20.4.3 Functional description

The following section describes the configuration and contents of DRE Message RAM.

Fixed setup of Message RAM layout

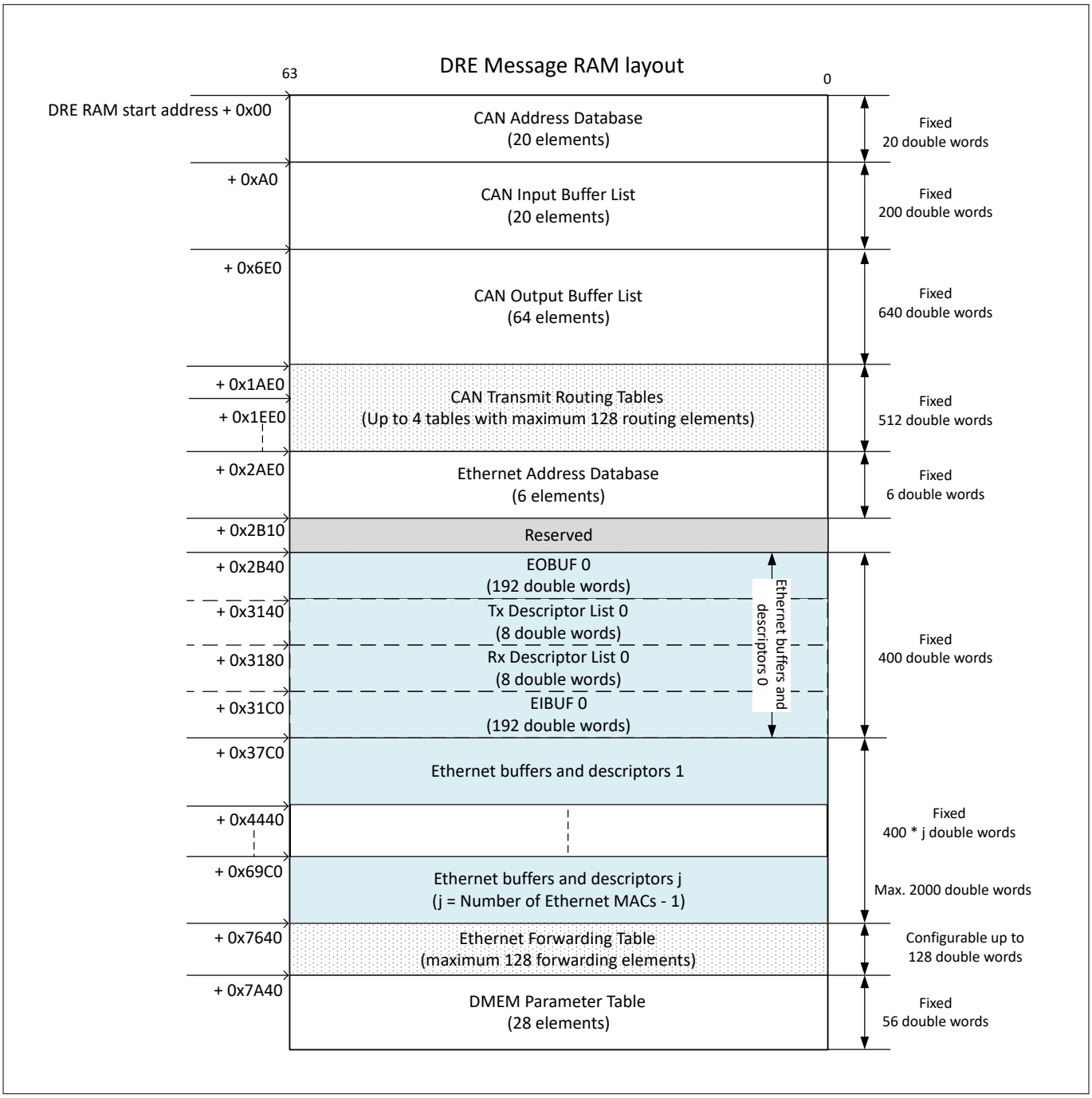


Figure 331 Setup of Message RAM layout

***Note**: The Message RAM is not accessible during Kernel reset*

20.4.3.1 CAN address database

The CAN Address Database (CAD) contains the control and address configuration of the CRE Receive host buffers and the Transmit Host Buffers.

The CRE of the CAN peripherals (refer to CAN chapter for detailed information) contains host buffers for Receive and Transmit CAN frames. The CAD needs to be configured by the user in order for the DRE to perform routing operations to or from CAN peripherals. The database is stored in the Message RAM of the DRE and always starts at the offset 0_H. The structure of the database is given in the figure below.

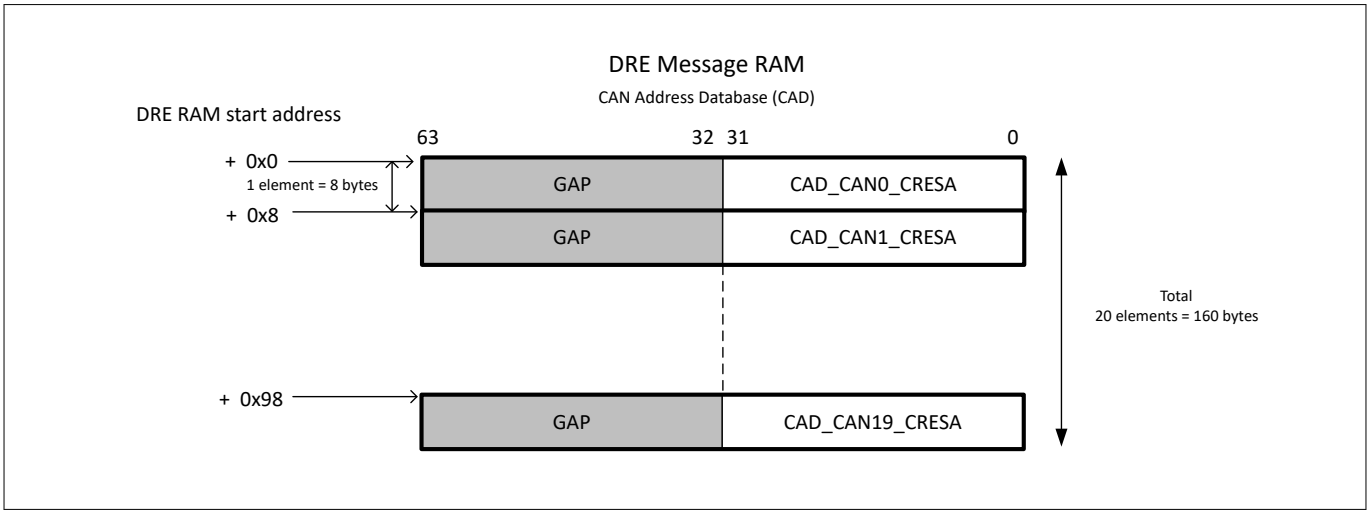


Figure 332 CAN address database RAM structure

Each CAD element contains the following configurations for the corresponding CAN peripheral:

- The absolute start address of the CRE address space (CAD_CANi_CRESA.ADR)

Related information

CAN input buffer list on page 3726
CAN output buffer list on page 3728
Routing control unit on page 3760

20.4.3.2 CAN input buffer list

The CAN Input Buffer List stores the CAN frames received by CAN interfaces which are to be transmitted through ACF Ethernet frames. The Routing Control Unit transfers the received CAN frames with destination $\mathrm{ID} = 18_H$ to $\mathrm{ID} = 1\mathrm{D}_H$ from MCMCAN to the CAN Input Buffer List.

The start address of the CAN Input Buffer List is DRE RAM start address + 0xA0. There are a total of 20 CAN input buffers. Each buffer stores 16 bytes of header data and 64 CAN data bytes. The structure of the buffers is shown in the figure below.

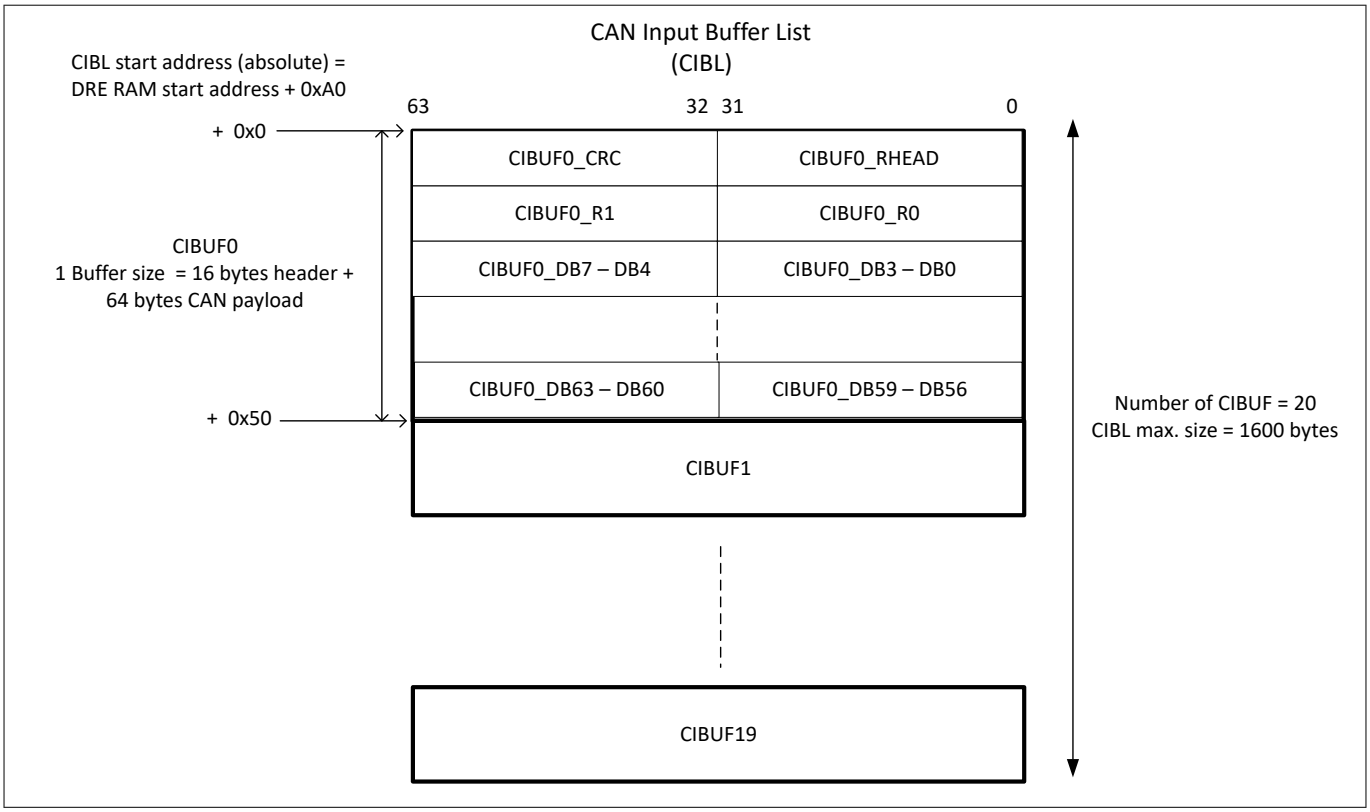


Figure 333 RAM structure of CAN input buffer list

The buffers are organized in the form of a circular list. The list contains a Put Index (CIBL_STATUS.PIDX).

The Put Index is controlled by the Message Controller. A new CAN frame is stored in the buffer indexed by the Put Index. The Put Index is incremented to the next free buffer after storing the CAN frame. The Put Index wraps around to the next free buffer starting from index 0 after it has reached the maximum number of assigned buffers in the list. The buffer pending request (BPR) in CIBL_BPR is set by hardware to indicate that the CAN buffer contains a frame yet to be processed by the ACF CAN-Ethernet Format Engine.

The pending requests are scanned by the ACF CAN-Ethernet Format Engine to identify the CAN frame to be processed. After processing the identified CAN frame, the pending request is cleared by the ACF CAN-Ethernet Format Engine.

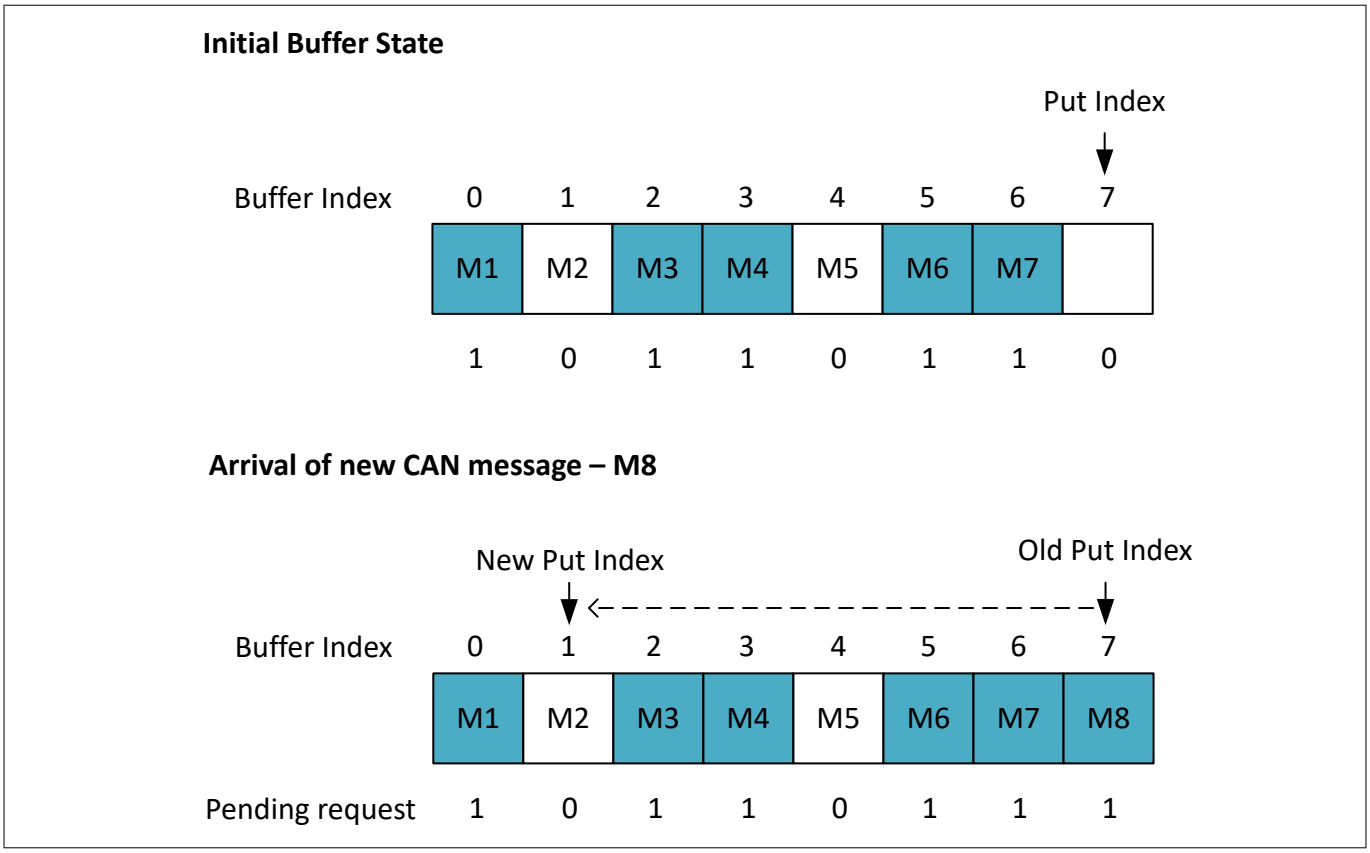


Figure 334 Example of CAN input buffer list handling with 8 CAN input buffers

In the example figure:

- The Put Index is currently at buffer 7
- When a new CAN frame (M8) is fetched by the data Move Engine, it stores it in buffer 7 as indicated by the Put Index
- The Put Index is then circularly incremented to the next free buffer location, which is buffer index 1

The Buffer Full flag (CIBL_STATUS.BF) is set by the Message Controller when the EOBUF is not available to store the incoming CAN frame and an interrupt is triggered. It is cleared by hardware after the frame is fetched from the EOBUF.

The Buffer Empty flag (CIBL_STATUS.BE) is set by the Message Controller when no pending requests are set. It is cleared by hardware when at least 1 buffer has a pending request set.

Related information

CAN-to-Ethernet ACF assembler on page 3746
Routing control unit on page 3760
CAN address database on page 3726
Ethernet output buffer on page 3730

20.4.3.3 CAN output buffer list

The CAN Output Buffer List stores the CAN frames which are to be transmitted through CAN interfaces.

After the CAN Transmit Routing Engine decides the routing destination of a CAN frame to be transmitted, the ACF CAN-Ethernet Format Engine writes the CAN frame into the CAN Output Buffer List. The Routing Control Unit transfers the CAN frame from the output buffers to their corresponding destinations.

The start address of the CAN Output Buffer List is DRE RAM start address + 0x6E0. There are a total of 64 CAN output buffers. Each buffer stores 16 bytes of header data and 64 CAN data bytes. The structure of the buffers is shown in figure below.

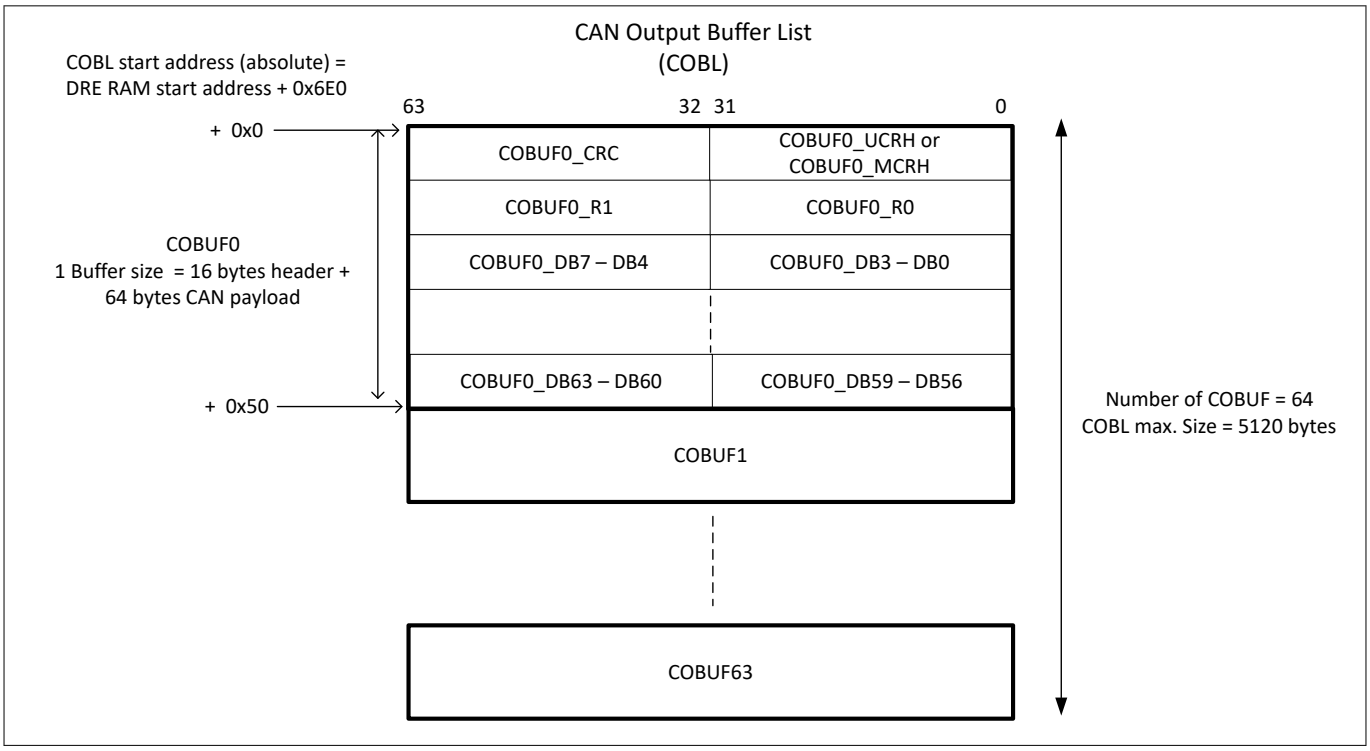


Figure 335 RAM structure of CAN output buffer list

The CAN output buffers are organized in the form of a circular list. The list contains a Put Index (COBL_STATUS.PIDX).

The Put Index is controlled by the Message Controller. A new transmit CAN frame is stored in the buffer indexed by the Put Index. The Put Index is incremented to the next free buffer after storing the CAN frame. The Put Index wraps around to the next free buffer starting from index 0 after the configured maximum number of buffers.

A pending request (COBL_BPR0.PRj (j=0-31) and COBL_BPR1.PRj (j=32-63)) is set by the ACF CAN-Ethernet Format Engine to indicate that the CAN buffer contains a frame yet to be transferred to its corresponding destination. The pending requests are scanned by the Routing Control Unit to identify the CAN frames to be transferred. After transfer of the fetched CAN frame to its corresponding destination, the pending request is cleared by the Routing Control Unit.

The Buffer Full flag (COBL_STATUS.BF) is set by the Message Controller when there are no free buffers to store the incoming CAN frame and an interrupt is triggered. It is cleared by hardware when there are free buffers available. The Buffer Empty flag (COBL_STATUS.BE) is set by the Message Controller when no pending requests are set. It is cleared by hardware when at least one buffer has a pending request set.

Related information

ACF_CAN_BRIEF message format decoder on page 3754
Routing control unit on page 3760
Ethernet ACF-to-CAN disassembler on page 3753
CAN address database on page 3726

20.4.3.4 CAN transmit routing tables (1 to 4)

The CAN Transmit Routing Tables store the routing elements which identify the CAN interface to which a CAN frame is to be transmitted. The routing tables are used by the CAN Transmit Routing Engine to identify the transmit CAN interface.

Up to 4 routing tables with a maximum of 128 routing elements per routing table can be configured. The start address of the first routing table is DRE RAM start address + 0x1AE0. The start address of the consecutive routing tables are at fixed offsets of 0x400 from the previous table. The number of routing elements in each routing table is configured using RTi_CONFIG.NRULES.

Each routing element contains:

1. CAN ID filter (RT_REj_CIDFC (j = 0-127)) and
2. Routing rule with CAN destinations (RT_REj_UCR or MCR (j = 0-127))

The CAN ID filter conditions are used by the CAN Transmit Routing Engine to identify the routing rule corresponding to a CAN frame based on the configured CAN ID filters.

The routing rule configuration is used by the CAN Transmit Routing Engine to identify the destination CAN interface.

Related information

CAN acceptance filters on page 3759

20.4.3.5 Ethernet output buffer

The Ethernet Output Buffer stores the transmit ACF frames containing one or more ACF_CAN_BRIEF messages. There are six EOBUFs available but usability depends on the number of Ethernet interfaces available in the device.

The start address of Ethernet Output Buffer j (j = 0 - 5) is configured at DRE RAM start address + 0x2B40 + j*0xC80. The buffers have the following three sections:

1. IEEE 802.3 MAC header
2. NTSCF header
3. ACF payload

The structure of the Ethernet Output Buffer is shown in the figure below.

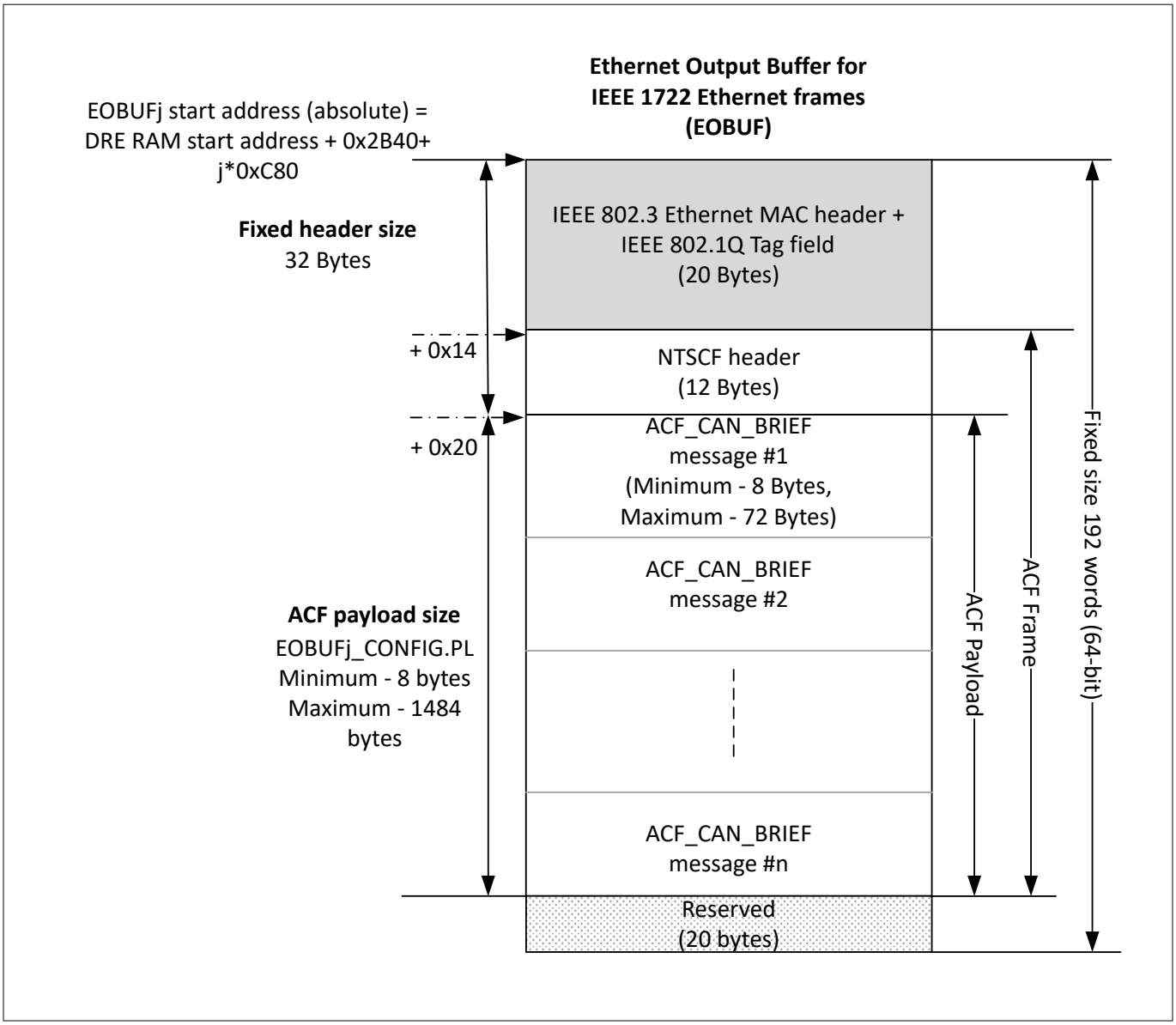


Figure 336 Ethernet output buffer structure

IEEE 802.3 MAC header

This section contains the user-configured Ethernet MAC header with optional IEEE 802.1Q tag field. The header always starts at the offset of 2_H from the start address of the buffer. The MAC header is configured in EOBUFj_MAC_H0, EOBUFj_MAC_H1, EOBUFj_MAC_H2, EOBUFj_MAC_H3 and EOBUFj_MAC_H4. The header is assembled by the ACF CAN-Ethernet Format Engine when an Ethernet frame transmit trigger condition occurs. The structure of the IEEE 802.3 MAC header is shown in the figure below.

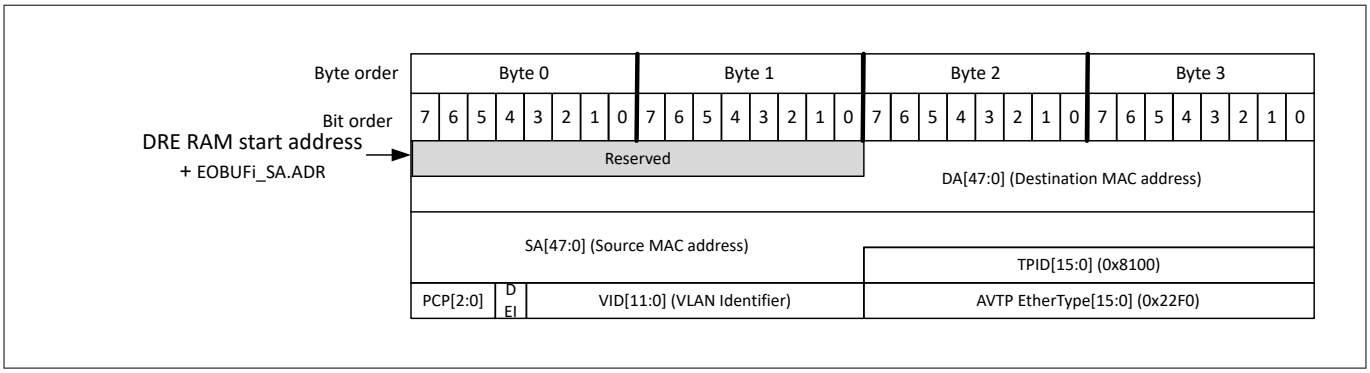


Figure 337 IEEE 802.3 MAC header

NTSCF header

This section contains the NTSCF header of the transmit Ethernet frame. The header always starts at the offset of 14_H from the start address of the buffer. The header information is shown in EOBUFj_NTSCF_H0. The header is assembled by the ACF CAN-Ethernet Format Engine when an Ethernet frame transmit trigger condition occurs. The structure of the NTSCF header is shown in the figure below.

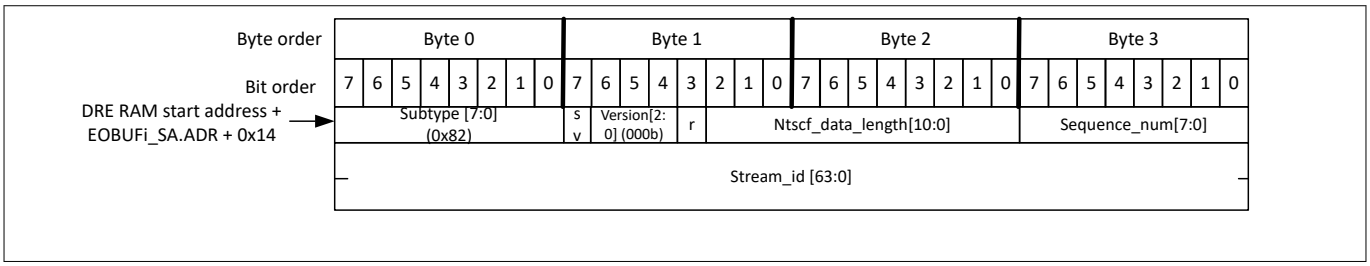


Figure 338 NTSCF header

ACF payload

This section contains one or more ACF_CAN_BRIEF messages. The ACF payload section always starts at the offset of 20_H from the start address of the buffer. The length of the ACF payload section is configured by the user at EOBUFj_CONFIG.PL. The structure of the ACF_CAN_BRIEF message is shown in the figure below.

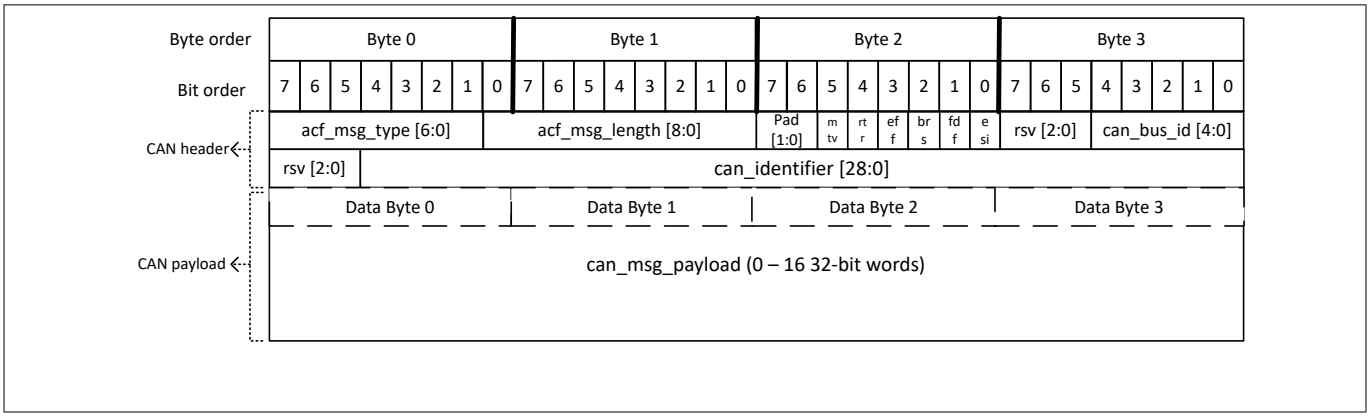


Figure 339 ACF_CAN_BRIEF message

Related information

References on page 3907
CAN-to-Ethernet ACF assembler on page 3746
Tx descriptor handler on page 3781
CAN input buffer list on page 3726

20.4.3.6 Ethernet input buffer

The Ethernet Input Buffer stores the received Ethernet frames from the Ethernet module. The ACF CAN-Ethernet Format Engine disassembles CAN frames from the ACF frames. Alternatively, the Ethernet Descriptor Handler forwards the Ethernet frame if the destination is an Ethernet interface.

The Ethernet Input Buffer stores the Ethernet frames received from Ethernet interfaces which are to be transmitted to CAN interfaces or forwarded to a different Ethernet interface. There are six EIBUFs available but usability depends on the number of Ethernet interfaces available in the device. The start address of Ethernet Input Buffer j (j = 0 - 5) is configured at DRE RAM start address + 0x31C0 + j*0xC80 . The size of the buffer is 192 double words which include the complete Ethernet frame (including MAC header, NTSCF header and ACF payload) plus some additional padding bytes to achieve 64-byte alignment. The state machine gets the size of the frame from the Rx descriptor in the case of Ethernet to Ethernet forwarding or the NTSCF header in the case of Ethernet to CAN routing. The structure of the header and the payload is the same as that of the Ethernet Output Buffer.

Related information

Ethernet ACF-to-CAN disassembler on page 3753
Rx descriptor handler on page 3784
Interrupt grouping on page 3789
Tx descriptor handler on page 3781

20.4.3.7 Ethernet descriptor lists

The Ethernet descriptor lists consist of a set of Rx or Tx descriptors to receive and transmit Ethernet frames.

The Ethernet descriptors are updated by the Ethernet Descriptor Handler (Read format) and the Ethernet DMA channels (Write-back format). The ETH Descriptor Handler consists of 6 Transmit Descriptor Lists (0-5) and 6 Receive Descriptor Lists (0-5) but usability depends on the number of Ethernet interfaces available in the device. There is one Tx list and one Rx list per Tx/Rx DMA channel respectively. The Message RAM consists of 4 Rx descriptors and 4 Tx descriptors per Ethernet interface. Each descriptor list contains 4 descriptors, and each descriptor is made up of 4 32-bit words.

The descriptors are used to exchange packet data and packet control or status information between the DRE and Ethernet DMA channel.

The start address of Ethernet Tx descriptor list j (j = 0 - 5) is configured at DRE RAM start address + 0x3140 + j*0xC80.

The start address of Ethernet Rx descriptor list j (j = 0 - 5) is configured at DRE RAM start address + 0x3180 + j*0xC80.

Related information

Tx descriptor handler on page 3781
Rx descriptor handler on page 3784
Forwarding table on page 3739

20.4.3.7.1 Descriptor structure configuration

This section describes the configuration of descriptors for use in the DRE.

In the ring structure, each descriptor is made of 4 32-bit words. The descriptor structure is configured by the user by configuring the following registers:

- The Rx and Tx descriptor lists can be assigned to an Ethernet interface by setting RETHDLi_CTRL.EIF and RETHDLi_CTRL.EIFID. For example, if EIBUF0 is used for receiving Ethernet frames from GETH0 Rx DMA channel 1:
  - RETHDL0_CTRL.EIF = 0
  - RETHDL0_CTRL.EIFID = 0
  - RETHDL0_CTRL.DMACH = 1

  Note: *Bit-fields RETHDLi_CTRL.DMACH and TETHDLi_CTRL.DMACH should be the same. Irrespective of the value configured in this bit-field, the DRE receives/transmits frames only from/to the Ethernet DMA channel whose Tail Pointer address is configured in the Ethernet Address Database (EAD) at the corresponding index RETHDLi_CTRL.EIF and RETHDLi_CTRL.EIFID. Bit-field RETHDLi_CTRL.DMACH is used in generating the FWDID while forwarding and bit-field TETHDLi_CTRL.DMACH is only for status information*

- RETHDLi_CTRL.TRIG and TETHDLi_CTRL.TRIG set the type of trigger from the Descriptor Handler. When a TRIG is set, Ethernet frames are sent to and received from software by triggering an interrupt. In this case, the descriptors are not prepared by the DRE

The user must pre-configure the descriptor structure of the DRE within the Ethernet DMA channel by updating the registers with the start addresses of the Tx and Rx descriptor list within the Message RAM:

- For GETH, DMA_CHj_TxDesc_List_LAddress and DMA_CHj_RxDesc_List_LAddress
- For LETH, DMA_CHy_TxDesc_List_Address and DMA_CHy_RxDesc_List_Address

The user must also update the registers with the total number Tx and Rx descriptors per respective DMA channel:

- For GETH, DMA_CHj_Tx_Control2.TDRL and DMA_CHj_Rx_Control2.RDRL
- For LETH, DMA_CHy_TxDesc_Ring_Length.TDRL and DMA_CHy_Rx_Control2.RDRL

Related information

Ethernet address database (EAD) on page 3740
Tx descriptor handler on page 3781
Rx descriptor handler on page 3784
Descriptor error handling on page 3789

20.4.3.7.2 Tx descriptors

This chapter describes the use of Tx descriptors in the DRE.

Whenever a new Ethernet frame is to be transmitted to a Tx DMA channel of an Ethernet interface, transmit descriptors (TDESC_RD0-3) are prepared by the DRE in Read format in the Message RAM. The Tx DMA reads the descriptor prepared by the DRE and then overwrites it with the Write-back format (TDESC_WR0-3) after reading the frame from the Ethernet Output Buffer. The following figures give an overview of the Read and Write-back formats. A detailed description can be found in the chapter Registers of this manual.

Read format

The following image shows an example of the structure of a Tx descriptor. Each Read format descriptor contains 4 TDESCi_RD descriptor words (i=0-3). Each Tx descriptor list in Read format contains 4 descriptors, giving 16 words for each list in total.

Words marked reserved contain no valid input.

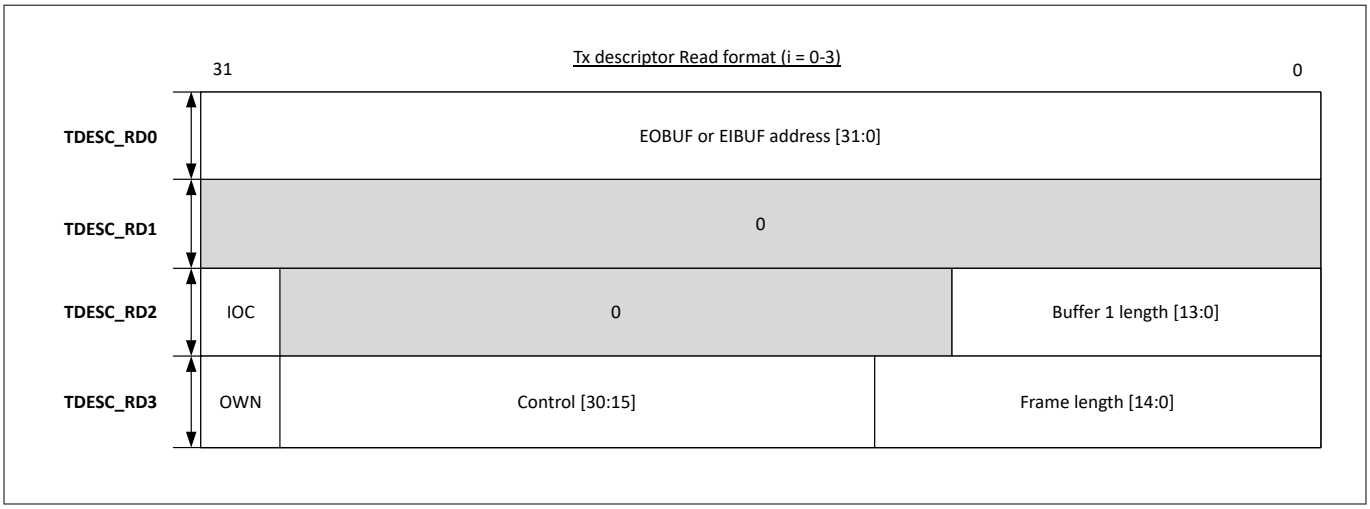


Figure 340 Tx descriptor - read format

The 4 descriptor words prepared by the DRE are: TDESCi_RD0, TDESCi_RD1, TDESCi_RD2 and TDESCi_RD3

Write-back format

The following image shows an example of the structure of a Tx descriptor. Each Write-back format descriptor contains 4 TDESCi_WR descriptor words (i=0-3). Each Tx descriptor list in Write-back format contains 4 descriptors, giving 16 words for each list in total.

Words marked reserved contain no valid input.

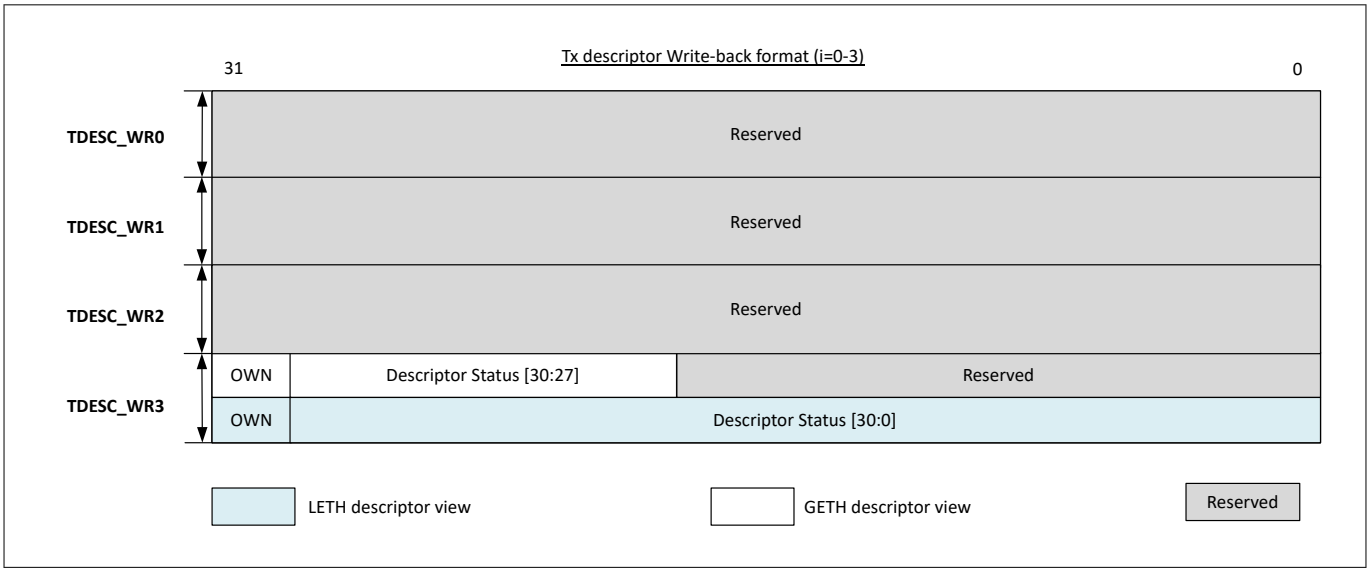


Figure 341 Tx descriptor - write-back format

- The GETH Tx DMA channel prepares the 4 descriptor words: TDESCi_WR0, TDESCi_WR1, TDESCi_WR2 and TDESCi_WR3G
- The LETH Tx DMA channel prepares the 4 descriptor words: TDESCi_WR0L, TDESCi_WR1L, TDESCi_WR2 and TDESCi_WR3L

Related information

Tx descriptor handler on page 3781

20.4.3.7.3 Rx descriptors

This chapter describes the use of Rx descriptors in the DRE.

Whenever a free EIBUF is available to receive an Ethernet frame from an Ethernet interface, receive descriptors (RDESC_RD0-3) are prepared by the DRE in Read format in the Message RAM. The Rx DMA reads the descriptor and then overwrites it with the Write-back format (RDESC_RD0-3) after writing the frame to the Ethernet Input buffer. The following figures give an overview of the Read and Write-back formats. A detailed description can be found in the chapter Registers in this manual.

Read format

The following image shows an example of the structure of an Rx descriptor. Each Read format descriptor contains 4 TDESCi_RD descriptor words (i=0-3). Each Rx descriptor list in Read format contains 4 descriptors, giving 16 words for each list in total.

Words marked reserved contain no valid input.

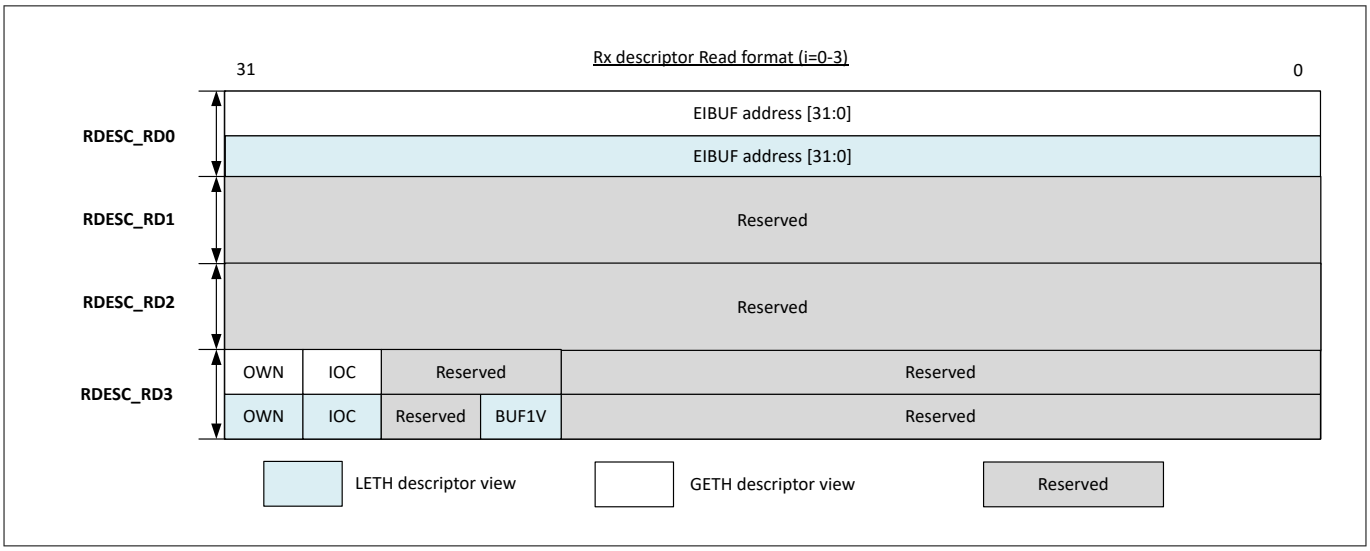


Figure 342 Rx descriptor - read format

The 4 descriptor words prepared by the DRE are: RDESCi_RD0, RDESCi_RD1, RDESCi_RD2, and RDESCi_RD3G for GETH or RDESCi_RD3L for LETH.

Write-back format

The following image shows an example of the structure of an Rx descriptor. Each Write-back format descriptor contains 4 TDESCi_WR descriptor words (i=0-3). Each Rx descriptor list in Write-back format contains 4 descriptors, giving 16 words for each list in total.

Words marked reserved contain no valid input.

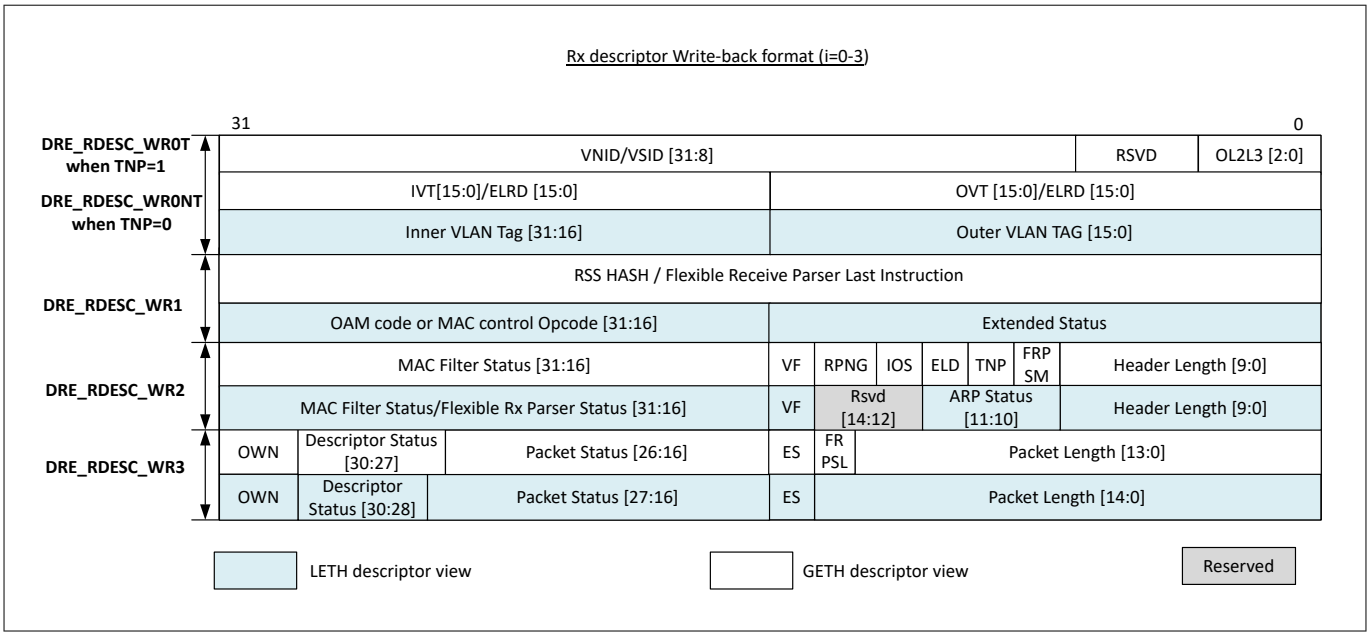


Figure 343 Rx descriptor - write-back format

The image above shows the differences for tunneled and non-tunneled frames for RDESCi_WR0. When TNP=1, RDESCi_WROT applies to both GETH and LETH interfaces. When TNP=0, RDESCi_WR0NT applies for LETH and GETH as shown by the colors in the image.

- The GETH Rx DMA channel prepares the 4 descriptor words: RDESCi_WR0T (for tunneled frames) or RDESCi_WR0NT (for non-tunneled frames), RDESCi_WR1G, RDESCi_WR2G and RDESCi_WR3G
- The LETH Rx DMA channel prepares the 4 descriptor words: RDESCi_WR0NT, RDESCi_WR1L, RDESCi_WR2L and RDESCi_WR3L

Related information

Rx descriptor handler on page 3784
Descriptor structure configuration on page 3733
Descriptor error handling on page 3789

20.4.3.8 Message RAM access protection

The Ethernet section is organized such that each EOBUF, it's corresponding EIBUF and the corresponding Tx and RX descriptor lists are located together in a consecutive RAM address range, as shown in the diagram below. Each address range has it's own Access Protection Unit (APU) associated to it, so each address range can be individually protected. Each APU corresponds to a Ethernet interface (SRI master) as shown in the image below. The start address of the APU is the same as the EOBUF start address. The owner of the APU is determined based on which master requires write access to the EIBUF and the descriptors (RETHDLi_CTRL.EIF, EIFID). While the owner of the APU must have read and write access, the other masters must have only read access.

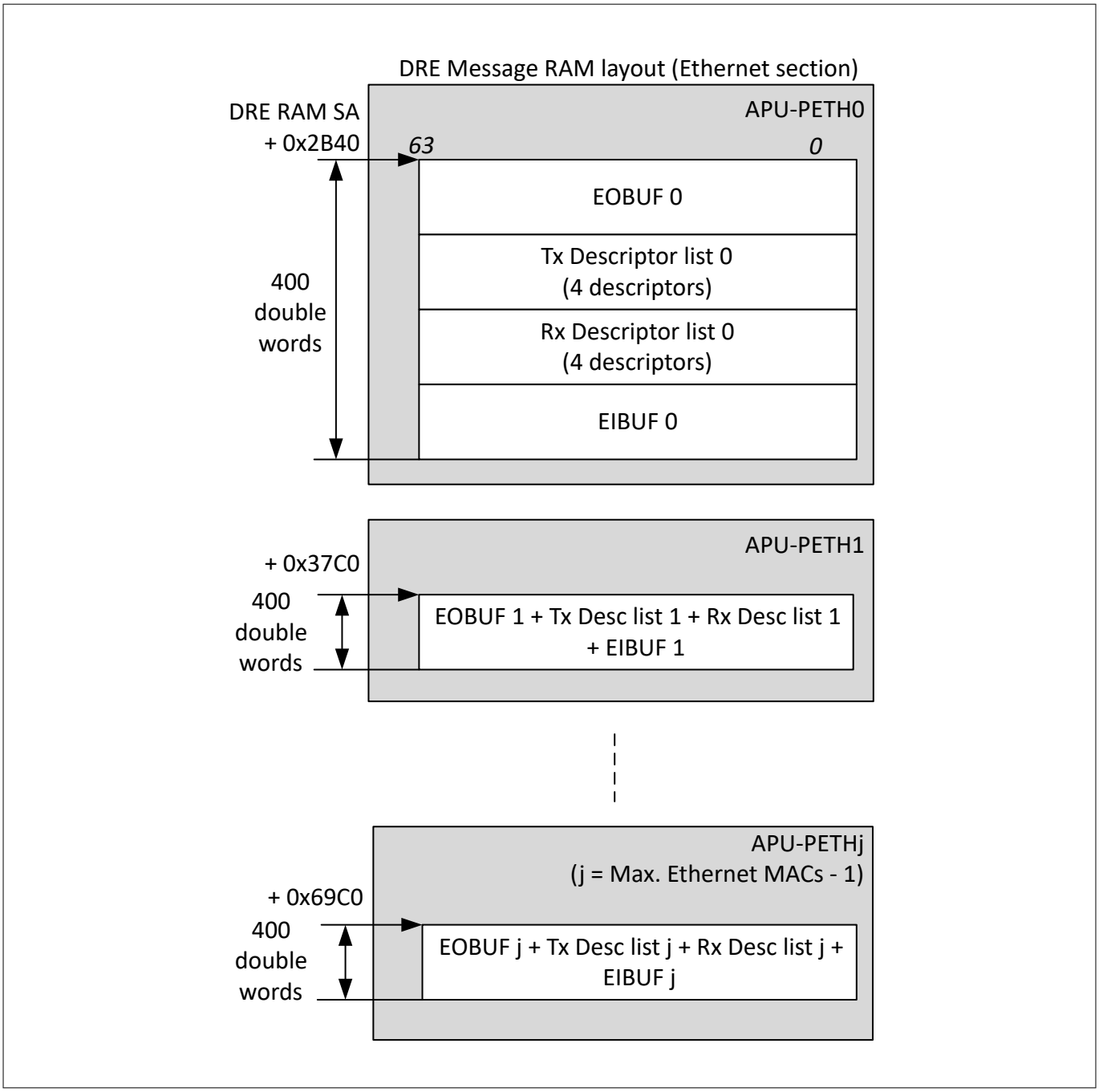


Figure 344 DRE message RAM layout for Ethernet

Example access matrices for GETH and LETH are given in Table 1 and Table 2 below. The APUs can be configured using the APU-PETHj (ETHj_ACCEN) registers (refer to the Registers chapters for details).

Table 933 Example access matrix (APU owned by GETH0)

Read access	Write access
• GETH0 • LETH interfaces	GETH0

Table 934 Example access matrix (APU owned by LETH0)

Read access	Write access
• LETH0 • GETH interfaces	LETH0

20.4.3.9 Forwarding table

The forwarding table is used by the Forwarding Engine to identify the transmit Ethernet interface.

The forwarding table stores the forwarding rules which identify the Ethernet interface to which an Ethernet frame is to be forwarded.

Up to a maximum of 128 forwarding elements can be configured. The start address of the forwarding table is DRE RAM start address + 0x7640. The number of forwarding elements can be configured through FTCFG.NRULES as shown in the figure below.

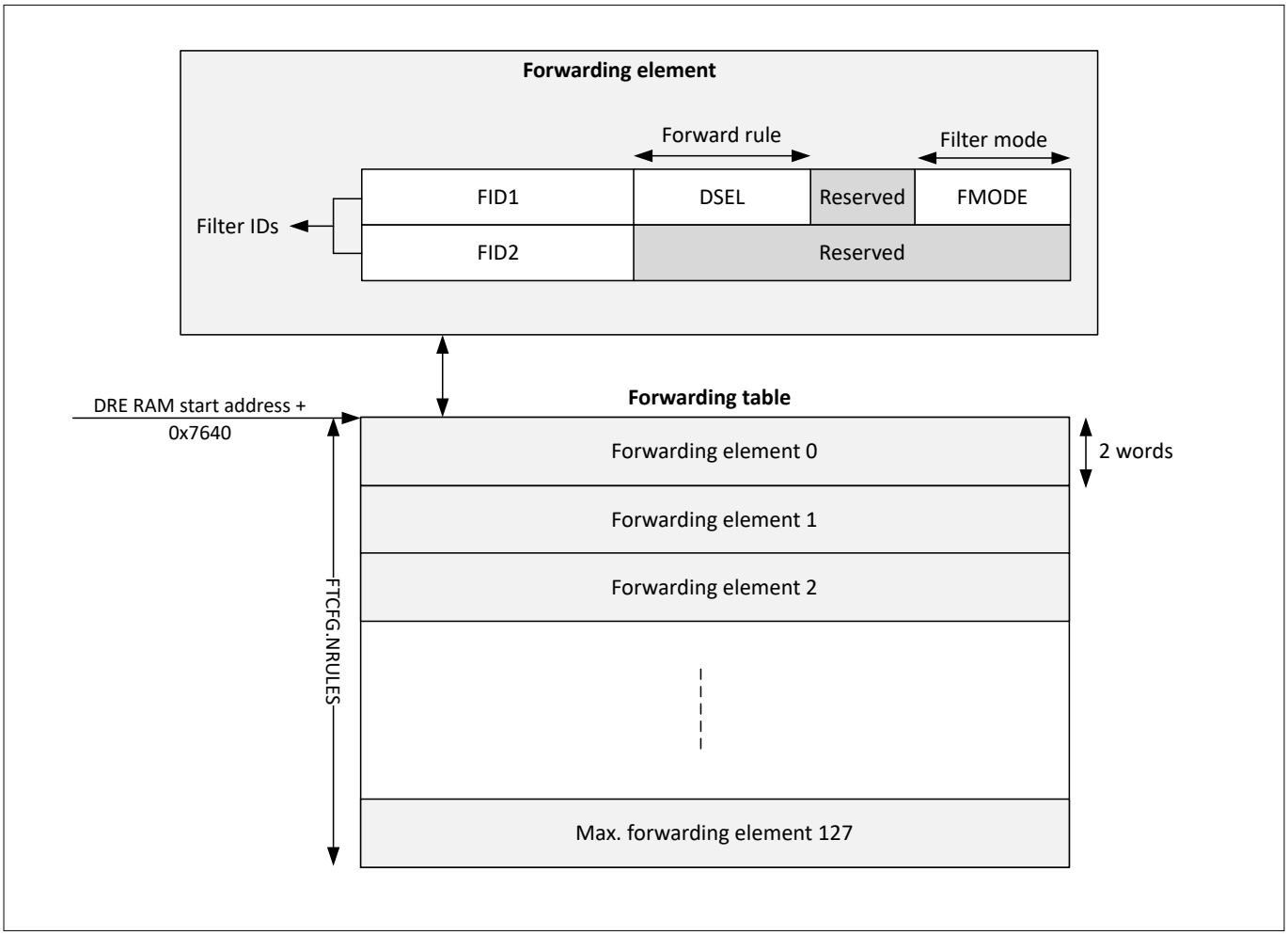


Figure 345 Forwarding table

Each forwarding element contains:

1. Uni-cast or multi-cast Forward ID filters Filter ID1 in bit-field FT_FEj_FRULE.FID1 and Filter ID2 in bit-field FT_FEj_FID2.FID2
2. Forwarding rule FT_FEj_FRULE.DSEL. The forwarding rule is bit-encoded. Single bit is set in the case of uni-cast and multiple bits are set in the case of multi-cast forwarding
3. Filter mode FT_FEj_FRULE.FMODE

The FWD ID filter conditions are used by the Forwarding Engine to identify the forwarding rule corresponding to an Ethernet frame based on the configured FWD ID based filters.

The forwarding rule configuration is used by the Forwarding Engine to identify the destination Ethernet interface of the corresponding Ethernet frame.

Related information

Ethernet descriptor lists on page 3733

20.4.3.10 Ethernet address database (EAD)

The Ethernet Descriptor Handler receives Ethernet frames from or transmits them to Ethernet DMA channels.

The Ethernet Address Database (EAD) needs to be configured by the user in order for the DRE to perform routing and forwarding operations to or from Ethernet DMA channels. The database is stored in the Message RAM of the DRE and always starts at address DRE RAM start address + 0x2AE0. The EAD is indexed based on RETHDLi_CTRL.EIF and EIFID configured. The Ethernet address database stores the addresses of the following Rx and Tx Tail Pointer registers of each Ethernet DMA channel, as shown in the figure below:

- DMA_CHy_RxDesc_Tail_Pointer (y=0-7) and DMA_CHy_TxDesc_Tail_Pointer (y=0-7) for LETH
- DMA_CHj_RxDesc_Tail_LPointer (j=0-7) and DMA_CHj_TxDesc_Tail_LPointer (j=0-7) for GETH

***Note**: One Tx and one Rx DMA channel per Ethernet MAC is configured to receive and/or transmit Ethernet frames from/to the DRE*

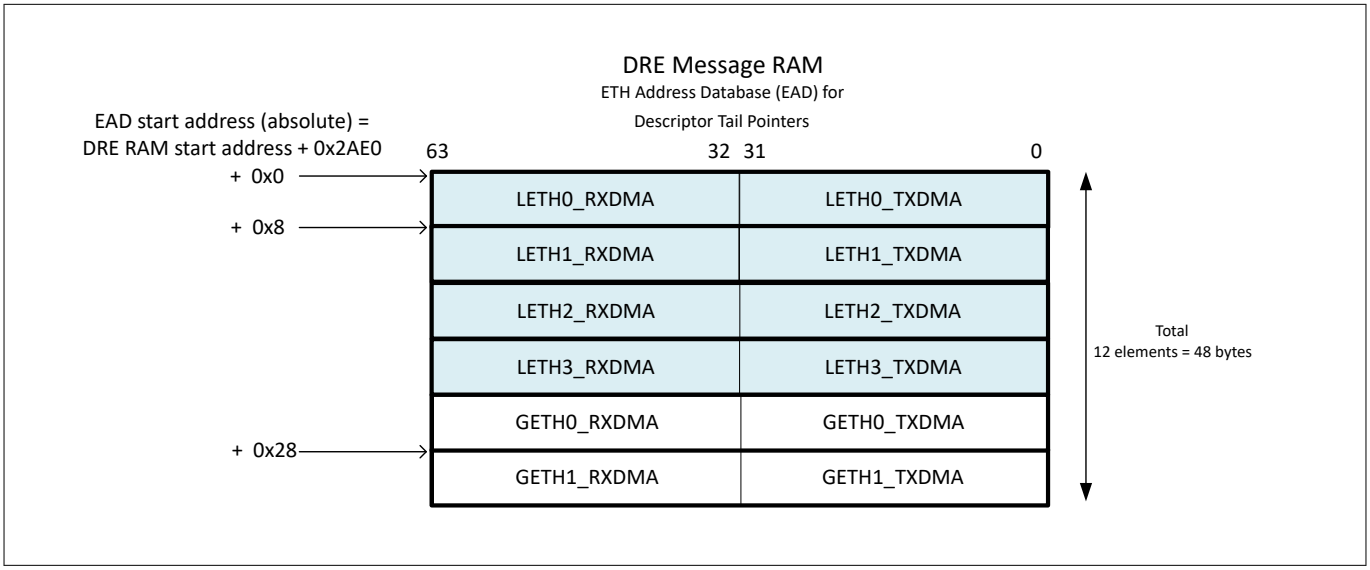


Figure 346 Ethernet address database structure

The Ethernet Descriptor Handler reads the Ethernet DMA Status register DMA_CH_Status during error or polls it to determine the status of the DMA. The bits highlighted in the figure below are checked by the Descriptor Handler to determine the status.

1. GETH DMA_CHj_Status (j=0-7) register address = DMA_CHj_RxDesc_Tail_LPointer (j=0-7) + EADCFG.GOV
2. LETH DMA_CHy_Status (y=0-7) register address = DMA_CHy_RxDesc_Tail_Pointer (y=0-7) + EADCFG.LOV

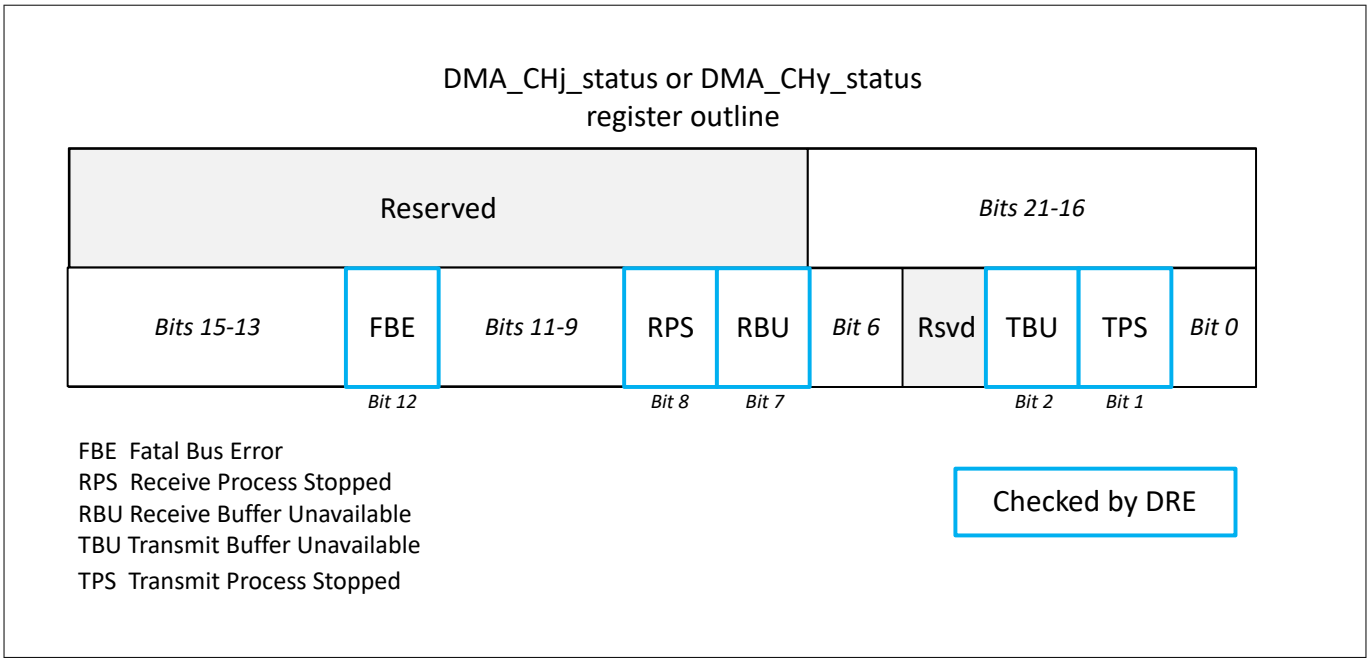


Figure 347 DMA status register outline

Related information

Descriptor structure configuration on page 3733
Descriptor error handling on page 3789
Tx descriptor handler on page 3781
Rx descriptor handler on page 3784

20.4.3.11 DMEM parameter table

The DMEM parameter table contains the control and address configuration of the destination memories used for CAN to system memory routing.

The start address of the DMEM parameter table is DRE RAM start address + 0x7A40. The DMEM parameter table consists of 28 elements. Each element corresponds to an individual destination memory (DMEMi) configuration. This way 28 independent destination memories can be configured. The DMEM parameter table layout is shown in the following figure.

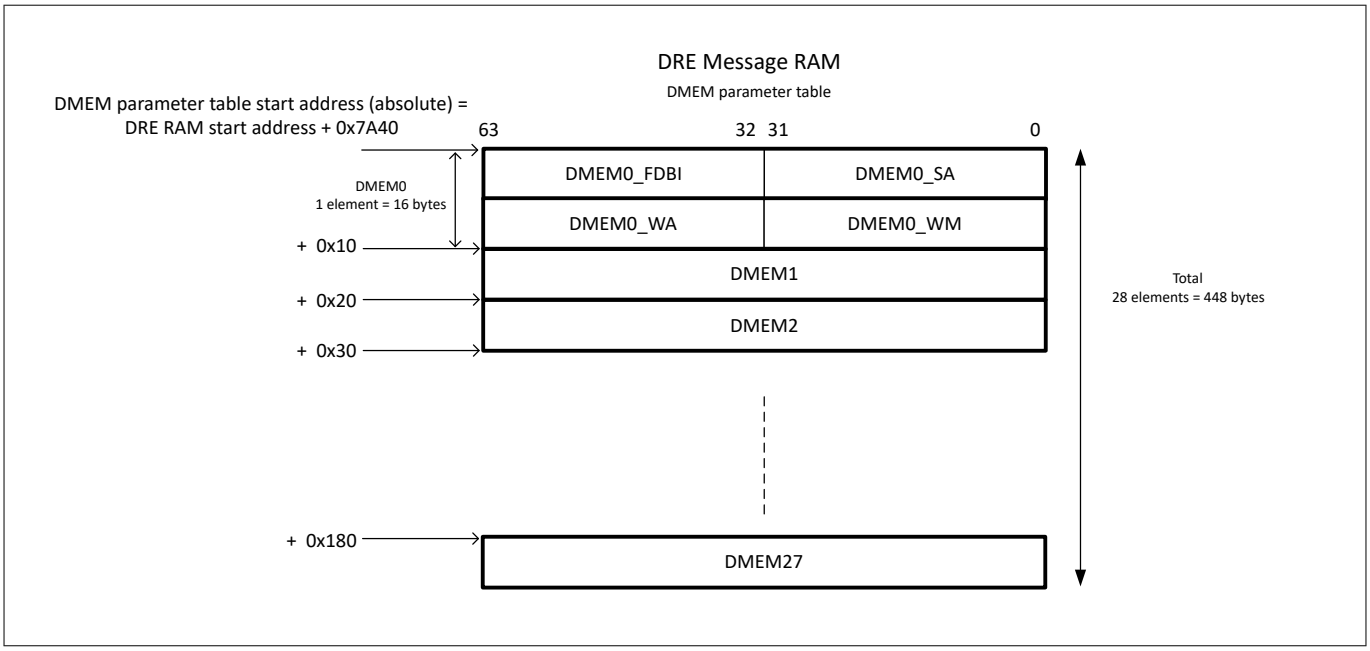


Figure 348 DMEM parameter table layout

The DMEM table is configured by the user with the following parameters:

1. DMEMi_SA defines the start address of DMEMi (i= 0 to 27)
2. DMEMi_WM defines the watermark level of DMEMi (i= 0 to 27). The watermark level is configured as number of CAN messages or number of words depending on the trigger mode.
3. DMEMi_WA defines the wraparound level of DMEMi (i= 0 to 27). The wraparound level is configured as number of CAN messages or number of words depending on the trigger mode.

The DMEM table also consists of DMEMi_FDBI, which is updated by the DRE after each CAN message is transferred to the destination memory.

Related information

System memory as destination (ID = 20 to 3B) on page 3768

20.5 ACF CAN-Ethernet format engine

This section describes the CAN-to-ACF and ACF-to-CAN frame formatting.

20.5.1 Feature list

- Concurrent assembly and disassembly of CAN (FD) frames into and from ACF frames according to IEEE 1722-2016
  - Supports Non-Time-Synchronous Control Format header (NTSCF)
  - Supports Abbreviated CAN/CAN FD message type (ACF_CAN_BRIEF)
- Support of different ACF frame transmission modes
  - Frame Count transmit mode
  - Buffer Fill Level transmit mode
  - Time-Triggered transmit mode
  - Software-Triggered transmit mode
- User-configurable stream ID for the ACF transmit frames
- Optional user configuration for Ethernet 802.3 MAC header and 802.1Q tag field corresponding to ACF frames contained in Layer 2 Ethernet frame

- Automatic stream ID-based sequence number generation and validation for ACF transmit and receive frames respectively
- Flexible identification of AVTP EtherType field start offset address in received ACF frames
- Stream ID-based acceptance filters for received ACF frames
- Support of the following error detections
  - Invalid NTSCF sub-type received
  - Invalid CAN/CAN FD message type received
  - ACF received frame length error

20.5.2 Functional overview

The block diagram of the ACF CAN-to-Ethernet Format Engine is shown in the following figure. It contains four blocks:

- Transmit Trigger Unit
- CAN-to-Ethernet ACF Assembler
- Ethernet ACF-to-CAN Disassembler
- Stream ID Filter

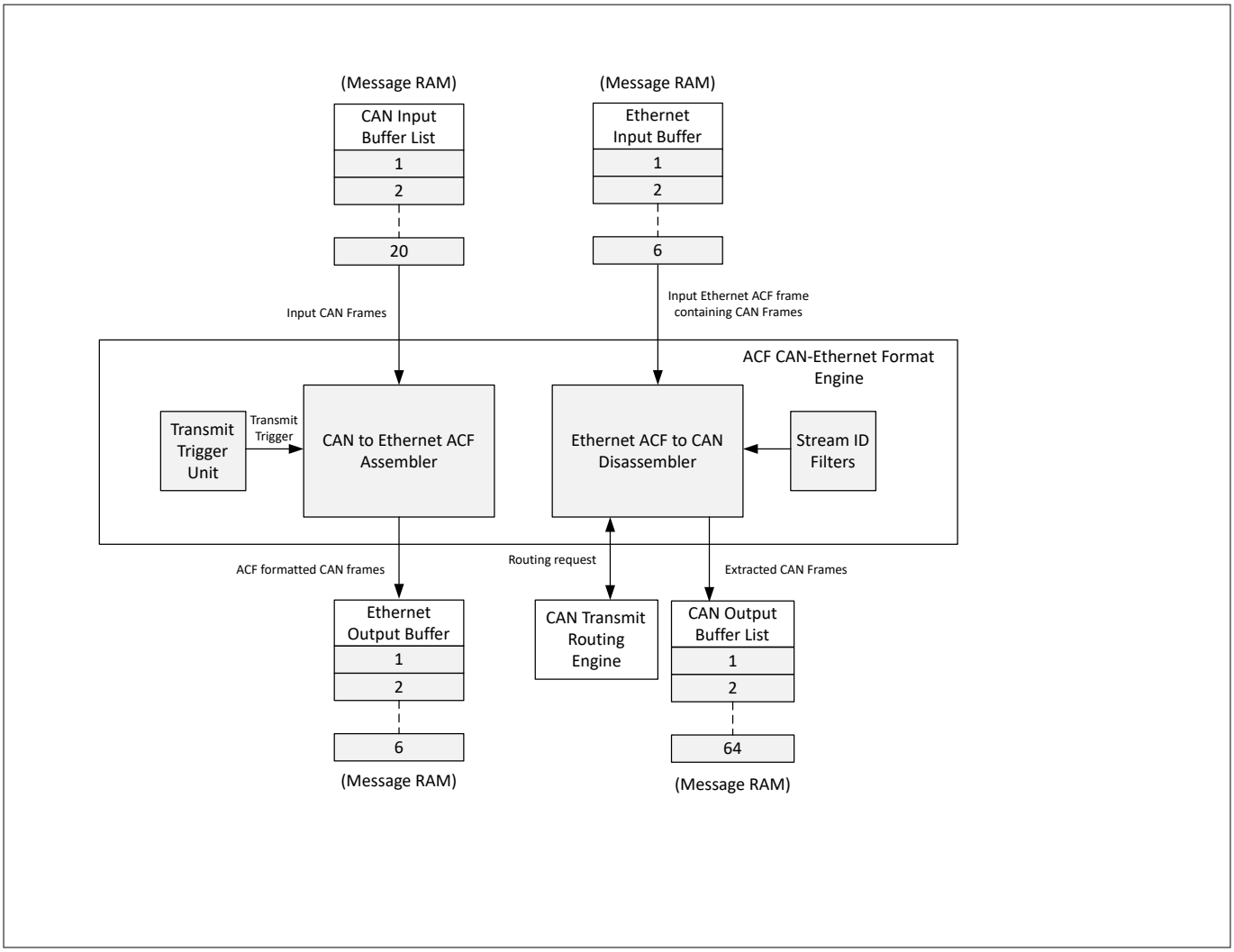


Figure 349 IEEE 1722a ACF format engine block diagram

Transmit Trigger Unit

The Transmit Trigger Unit controls the completion criteria of assembling the CAN-over-Ethernet frame which is performed by the CAN-to-Ethernet ACF Assembler. Based on the configuration of the Transmit Trigger Unit, a transmit trigger is sent to the CAN-to-Ethernet ACF Assembler, which marks that the corresponding Ethernet Output Buffer is ready for transmission to an Ethernet interface. The Transmit Trigger Unit supports 4 modes of operation:

- Frame Count transmit mode
- Buffer Fill Level transmit mode
- Time-Triggered transmit mode
- Software-Triggered transmit mode

CAN-to-Ethernet ACF Assembler

The CAN-to-Ethernet ACF Assembler performs the formatting of the CAN frame to Abbreviated CAN/CAN FD message type (ACF_CAN_BRIEF) format as specified by the IEEE 1722-2016 standard. The CAN frames are fetched from the CAN Input Buffer List and the formatted CAN frames are written to the Ethernet Output Buffer corresponding to the destination ID contained in the routing header of the input CAN frame. In addition to the ACF payload, it also prepares the NTSCF header.

Ethernet ACF-to-CAN Disassembler

The Ethernet ACF-to-CAN Disassembler performs the extraction of Abbreviated CAN/CAN FD messages (ACF_CAN_BRIEF) from an ACF frame in the Ethernet Input Buffer and formats it to the CAN frame transmit format. The extracted and formatted transmit CAN frames are handed to the CAN Transmit Routing Engine to determine the destination CAN interface.

Stream ID Filter

The Stream ID Filters contain the configuration for accepted Stream IDs of the input ACF frames. They also contain routing table indexes for each Stream ID filter. The routing table index is used by the CAN Transmit Routing Engine to select a corresponding CAN Transmit Routing Table which is compared to extracted CAN frames to identify the destination CAN interface.

20.5.3 Functional description

In this section, the packing and un-packing of CAN frames to or from ACF frames are described in detail.

20.5.3.1 Transmit trigger unit

The Transmit Trigger Unit enables the user to configure the transmit condition of the ACF frame. The following transmit modes are supported and can be configured with EOBUFj_CONFIG.TTM:

- Frame Count
- Buffer Fill Level
- Time-Triggered
- Software-Triggered

20.5.3.1.1 Frame count transmit mode

The transmission of the ACF frame is triggered when the number of ACF_CAN_BRIEF messages in an Ethernet Output Buffer has reached the user-configured frame count value in the EOBUFj_TTC.TFL bit-field.

The Frame Count transmit mode is enabled when EOBUFj_CONFIG.TTM = 01_B. The current number of CAN frames packed in an Ethernet Output Buffer is seen at EOBUFj_STATUS.ACFL. When it reaches the EOBUFj_TTC.TFL of an Ethernet Output Buffer, the EOBUFj_STATUS.TXRDY flag is set.

20.5.3.1.2 Buffer fill level transmit mode

The transmission of the ACF frame is triggered when the ACF payload size is greater than or equal to the configured Buffer Fill level in EOBUFj_TTC.BUFT.

The Buffer Fill Level transmit mode is enabled when EOBUFj_CONFIG.TTM = 10_B. The current ACF payload size of an Ethernet Output Buffer is shown in EOBUFj_NTSCF_H0.NTSCFDL. When the ACF payload size is greater than or equal to the configured Buffer Fill level (EOBUFj_TTC.BUFT), the Transmit Ready flag (EOBUFj_STATUS.TXRDY) is set.

20.5.3.1.3 Time-triggered transmit mode

The transmission of the ACF frame is triggered after a configured time interval.

The Time-Triggered transmit mode is enabled when EOBUFj_CONFIG.TTM = 11_B. The clock source of the timer is fSRI. The pre-scaler value for the timer is configured in EOBUFj_TTC.TP. Writing a non-zero value to the Timer Reload Value (EOBUFj_TTS.TRV) starts the timer. At the start of the timer, the TRV value is loaded to the Current Timer Value (EOBUFj_TTC.CTV) and it is decremented at the frequency of the clock source fSRI (pre-scaled by EOBUFj_TTC.TP). When EOBUFj_TTC.CTV reaches 00_H, the Transmit Ready flag (EOBUFj_STATUS.TXRDY) is set. Upon a trigger condition, when the corresponding Ethernet Output Buffer is empty, the Transmit Ready flag is not set and the Transmit Trigger Lost flag (EOBUFj_STATUS.TTL) is set.

20.5.3.1.4 Software-triggered transmit mode

The transmission of the ACF Ethernet frame is triggered from the application software.

When EOBUFj_CONFIG.TTM = 0_B, the application software can trigger the transmission of the ACF frame by setting EOBUFj_STATUS.TXRDY. When the TXRDY flag is set and the corresponding EOBUF is empty, the Transmit Trigger Lost flag (EOBUFj_STATUS.TTL) is set.

20.5.3.2 CAN-to-Ethernet ACF assembler

The CAN-to-Ethernet ACF Assembler:

1. Formats the CAN frames in CIBUF from CAN Input Buffer format to ACF_CAN_BRIEF format
2. Writes the ACF_CAN_BRIEF messages to the Ethernet Output Buffer
3. Writes the Ethernet header and NTSCF header to the Ethernet Output Buffer

The state machine of the CAN-to-Ethernet ACF Assembler is shown in the figure below.

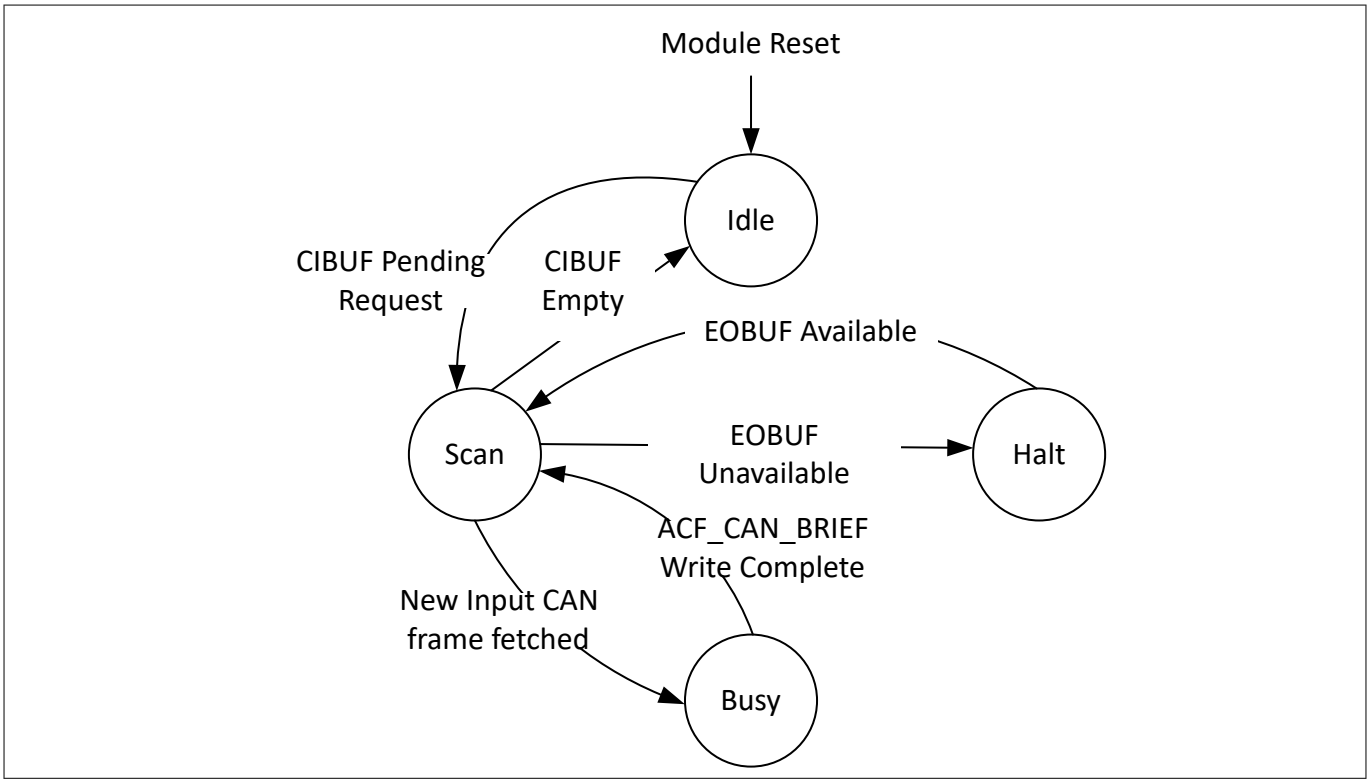


Figure 350 CAN-to-Ethernet state machine diagram

The transition conditions are described in the following table.

Table 935 CAN-to-Ethernet state transition condition

Current state	Next state	Transition condition
Any	Idle	Module Reset triggered
Idle	Scan	Pending Request of CAN Input Buffer
Scan	Idle	CAN Input Buffer List is empty
Scan	Busy	New CAN frame from CAN Input Buffer is ready to be processed
Busy	Scan	Assembling CAN frame in Ethernet Output Buffer is completed
Scan	Halt	No relevant Ethernet Output Buffer is available
Halt	Scan	An Ethernet Output Buffer becomes available

Related information

CAN input buffer list on page 3726
Ethernet output buffer on page 3730

20.5.3.2.1 Fetch input CAN frame

The Pending Request bit (CIBL_BPR) is used to identify the CAN frames to be processed from the CAN Input Buffer List. Upon a new CAN Input Buffer pending request, the state machine transits from Idle to Scan state. In this state, the pending requests of the CAN Input Buffers are scanned from CAN Input Buffer Index 0 up to Index 19. A CAN frame from a CAN Input Buffer is processed when the following conditions are satisfied:

1. The CAN Input Buffer has a pending request set, and
2. All destination Ethernet Output Buffer(s) are enabled and there is a free Ethernet Output Buffer (EOBUFj_STATUS.TXRDY = 0 and EOBUFj_STATUS.TXREQ = 0)

The destination Ethernet Output Buffer(s) is determined by the Routing header of the input CAN frame (CIBUFj_RHEAD). When an Ethernet destination(s) is enabled in CIBUFj_RHEAD, the corresponding Ethernet Destination IDs are compared with configured matching Destination IDs of the Ethernet Output Buffers 0 to 5 (EOBUFj_CONFIG.DID). The EOBUF with the matching Destination ID is used as the destination Ethernet Output Buffer. When no match is found, the Invalid Routing Destination Error flag (ME_ERR.IRDE) is set.

The CAN frame satisfying the above-mentioned conditions is then formatted with the ACF_CAN_BRIEF frame structure. The CAN Input Buffer Index which matches the conditions is reflected in the CIBL_STATUS.CBI bit-field and the state is set to Busy. In case the above conditions are not met by a CAN Input Buffer and the corresponding CAN frame, the next CAN Input Buffer is scanned, until the configured maximum number of buffers is reached. A transition from Scan to Idle state is performed when the Buffer Empty flag (CIBL_STATUS.BE) is set. A transition from Scan to Halt state is performed when no CAN frame with pending request has destination Ethernet Output Buffer(s) free.

***Note**: The order of the ACF_CAN_BRIEF messages assembled in the EOBUF is not maintained as per the order of incoming CAN requests*

20.5.3.2.2 ACF_CAN_BRIEF message format encoder

ACF_CAN_BRIEF message format encoder

The input CAN frame is converted to the ACF_CAN_BRIEF format as defined by IEEE 1722-2016. The format of the ACF_CAN_BRIEF message type is shown in figure below.

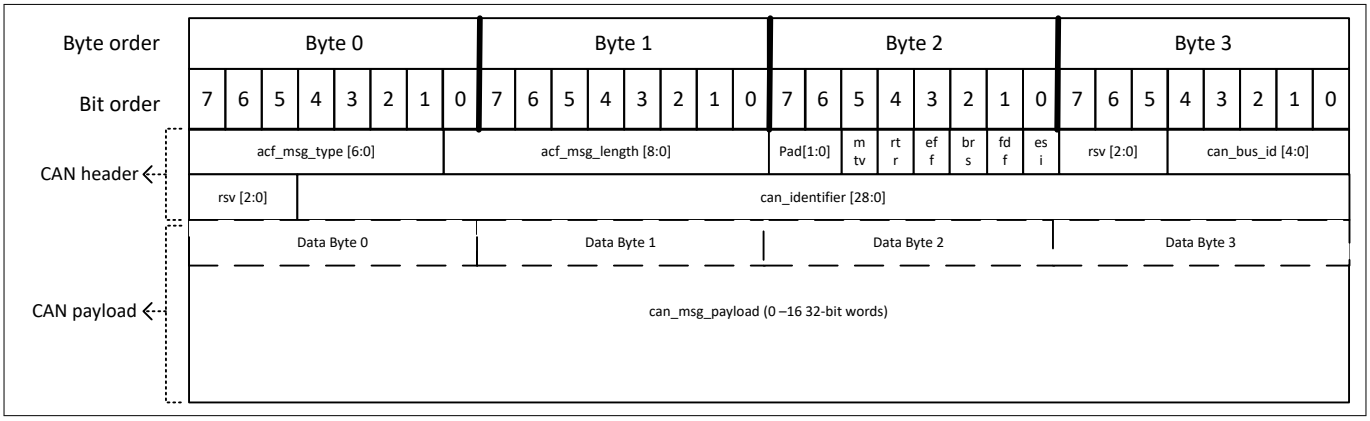


Figure 351 ACF_CAN_BRIEF frame format

The ACF_CAN_BRIEF message has the following fields:

- acf_msg_type: 7 bits
- acf_msg_length: 9 bits
- pad (padding length): 2 bits
- mtv (message_timestamp valid): 1 bit
- rtr (remote transmission request): 1 bit

- eff (extended frame format): 1 bit
- brs (bit rate switch): 1 bit
- fdf (CAN Flexible Data-rate Format): 1 bit
- esi (error state indicator): 1 bit
- rsv (reserved): 3 bits
- can_bus_id: 5 bits
- can_identifier: 29 bits
- rsv: 3 bits
- can_msg_payload: 0 to 16 (32-bit) words

acf_msg_type field

The acf_msg_type field describes the type of message contained in the ACF message payload. The ACF_CAN_BRIEF frames have a fixed acf_msg_type value of 0x02.

acf_msg_length field

The acf_ msg_length field indicates the number of words contained in the ACF_CAN_BRIEF frame. It includes the number of words in the CAN header and in the CAN payload. The length of the input CAN frame payload is indicated in the DLC field of the CAN Input Buffer. The acf_msg_length field always represents the number of 32-bit words in the frame. The minimum length is always 2, that is, when a CAN message contains no payload, then the acf_msg_length contains a value of 2 representing only the words occupied by the CAN header. When a CAN frame has a payload length which is not an integral multiple of 32-bit words, then the remaining bytes of the can_msg_payload are padded with zero (0) to reach an integral multiple of 32-bit words. This is described in detail below in the pad field description.

pad (padding length) field

The pad field indicates the length of padding at the end of the CAN message payload in bytes. Padding is added to the payload of the input CAN frame so that the can_msg_payload ends on a word boundary. A value of zero (0) is used in case there is no padding required for the CAN message payload. The table below shows the encoding of the acf_msg_length and pad fields values for corresponding input CAN message length (DLC) values.

Table 936 CAN message payload length transformation

Input CAN message data length code (DLC)	Input CAN message data length (in bytes)	can_message_payload length (in words)	Pad value (in bytes)	acf_msg_length value (in words)
0_H	0_D	0_D	0_D	2_D
1_H	1_D	1_D	3_D	3_D
2_H	2_D	1_D	2_D	3_D
3_H	3_D	1_D	1_D	3_D
4_H	4_D	1_D	0_D	3_D
5_H	5_D	2_D	3_D	4_D
6_H	6_D	2_D	2_D	4_D
7_H	7_D	2_D	1_D	4_D
8_H	8_D	2_D	0_D	4_D
9_H	12_D	3_D	0_D	5_D
A_H	16_D	4_D	0_D	6_D

(table continues...)

Table 936 (continued) CAN message payload length transformation

Input CAN message data length code (DLC)	Input CAN message data length (in bytes)	can_message_payload length (in words)	Pad value (in bytes)	acf_msg_length value (in words)
B_H	20_D	5_D	0_D	7_D
C_H	24_D	6_D	0_D	8_D
D_H	32_D	8_D	0_D	10_D
E_H	48_D	12_D	0_D	14_D
F_H	64_D	16_D	0_D	18_D

mtv (message_timestamp_valid) field

The mtv field is always set to zero (0) for the ACF_CAN_BRIEF message format.

rtr (remote_transmission_request) field

The rtr field contains the CAN message Remote Transmission Request bit. The value of this bit-field corresponds to CIBUFj_R0.RTR bit-field of the CAN Input Buffer. In case of Remote frames, there is no payload data copied and the frame ends at R1.

eff (extended_frame_format) field

The eff field contains the CAN message Extended Frame Format bit. The value of this bit-field corresponds to the CIBUFj_R0.XTD bit-field of the CAN Input Buffer.

Input CAN frames with standard 11-bit CAN identifier have their eff bit set to zero (0).

Input CAN frames with extended 29-bit CAN identifier have their eff bit set to one (1).

brs (bit_rate_switch) field

The brs field contains the CAN message Bit Rate Switch bit. The value of this bit-field corresponds to the CIBUFj_R1.BRS bit-field of the CAN Input Buffer.

fdf (CAN Flexible Data-rate [FD] Format) field

The fdf field contains the CAN message Flexible Data-rate Format bit. The value of this bit-field corresponds to the CIBUFj_R1.FDF bit-field of the CAN Input Buffer.

If the fdf bit is set to zero (0), then it is a classical CAN message and the valid lengths for data in the can_msg_payload field are 0 through 8 bytes.

If the fdf bit is set to one (1), then it is a CAN FD message and the valid lengths for data in the can_msg_payload field are 0 through 8, 12, 16, 20, 24, 32, 48 and 64 bytes.

esi (error_state_indicator) field

The esi field contains the CAN message Error State Indicator bit. The value of this bit-field corresponds to the CIBUFj_R0.ESI bit-field of the CAN Input Buffer.

can_bus_id field

The can_bus_id field contains an identifier for the CAN bus on which this frame is received. The value of this bit-field corresponds to the bits CIBUFj_RHEAD.SCBID[4:0] of the CAN Input Buffer.

can_identifier field

The can_identifier field contains the CAN frame identifier. The CAN frame identifier is either 11 or 29 bits in length. The length of the can_identifier is indicated by the eff field. 11-bit CAN message identifiers CIBUFj_R0.ID[28:18] are stored in byte 2, bits 2 through 0 and byte 3, bits 7 through 0. The remaining bits of the field are set to zero (0). 29-bit CAN frame identifiers CIBUFj_R0.ID[28:0] occupy the entire 29-bit-field length of the can_identifier field. The mapping of the CAN ID from the CAN Input Buffer to the ACF can_identifier field is shown in the figure below. Valid CAN ID bits are marked as 'v' in the figure.

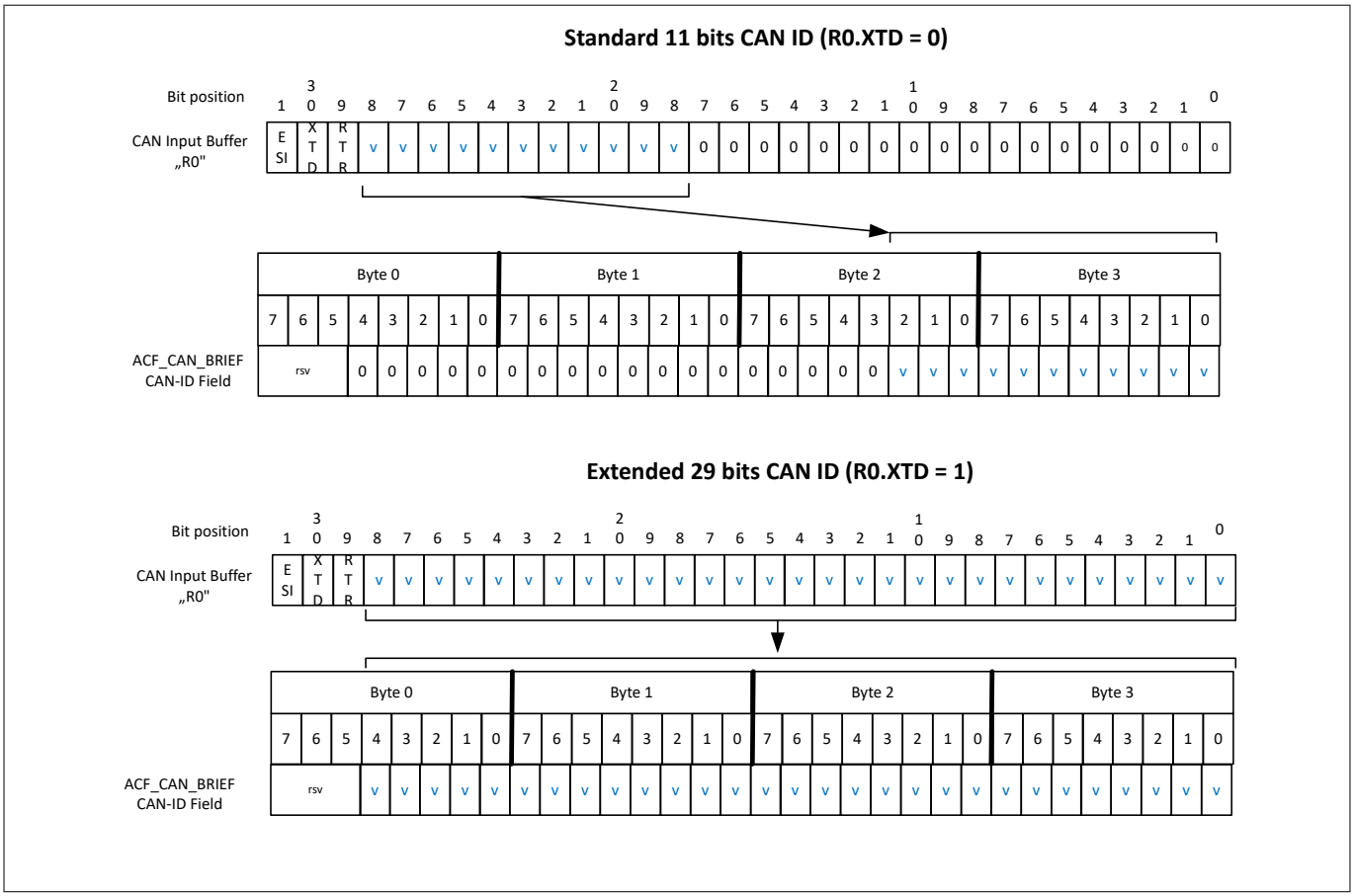


Figure 352 CAN ID mapping

can_msg_payload field

The can_msg_payload field contains the CAN frame payload data. The size of this field can be from 0 to 16 words, depending on the DLC (Data Length Code) of the input CAN frame. Only the data bytes of length corresponding to CIBUFj_R1.DLC from the CAN Input Buffer are written to the can_msg_payload field. The overall size of the can_msg_payload field is always an integral multiple of words (one word is of 4-bytes). If the data length is not word aligned, padding is always added at the end of the CAN frame payload to satisfy the length condition as shown in the previous table.

20.5.3.2.3 Assembling the CAN frame

This section describes the steps performed by the CAN-to-Ethernet ACF Assembler to assemble CAN frames.

Identification of the destination Ethernet Output Buffer

The destination of the input CAN frame is identified through the Destination ID in the Routing header of the CAN frame (CIBUFj_RHEAD.ETHDID). Valid Destination ID values for Ethernet are from 18_H until $1D_H$. The Ethernet Output Buffer corresponding to the Destination ID values are configured in EOBUFj_CONFIG.DID. The Ethernet Output Buffer that matches the Destination ID of the input CAN frame is chosen to store the corresponding ACF_CAN_BRIEF message. In case the Ethernet Output buffer configurations contain the same Destination ID values, Ethernet Output Buffer 0 is given the highest priority followed by the other buffers, provided that the Buffer 0 is free to accept a new CAN frame.

Write operation to the ACF payload section

The input CAN frames which are encoded in the ACF_CAN_BRIEF message format are written to the ACF Payload section of the Ethernet Output Buffer. The ACF Payload section starts at an offset address of 0x20 from the Ethernet Output Buffer start address. The offset address at which the current ACF_CAN_BRIEF message is to

be written is indicated in EOBUFj_STATUS.ACF_CAN_ADDR. The ACF_CAN_BRIEF message is written to the indicated payload address. Then following occurs after completion of the write operation:

1. The EOBUFj_STATUS.ACF_CAN_ADDR is incremented by $\text{can_msg_payload} \times 4$ (multiple of 4 is done to convert the size from 32-bit words to bytes)
2. The ntscf_data_length (EOBUFj_NTSCF_H0.NTSCFDL) of the NTSCF header is incremented by $\text{can_msg_payload} \times 4$

The Assembler verifies the 16-bit CRC (CIBUFj_CRC) which is calculated by the CRE over R0, R1 (except ESI, ANMF, RXTS, FIDX and BRS which are considered as zero), safety critical CAN payload and the DID (which is padded with zeros to make it 32-bits) which is part of the input CAN frame in the CIBUF before it writes the final block of data into the EOBUF. The CRC is calculated over 16 bits of data at a time starting with the MSB. It uses the CCITT CRC16 polynomial: 0x1021 $x^{16} + x^{12} + x^5 + 1$. If there is a mismatch in the CRC, the CRC Error flag (CIBL_STATUS.CRCE) is set and an error interrupt is triggered but the CAN frame is still processed by the DRE.

***Note**: In case of CAN to CAN routing, the DRE does not calculate the CRC. It only copies the CRC from the source CAN node to the destination CAN node.*

Acknowledgement of the CAN Input Buffer

After finishing the write operation of the completed ACF_CAN_BRIEF message to the Ethernet Output Buffer, the pending request of the corresponding CAN Input Buffer is cleared by the hardware. EOBUFj_STATUS.ACFL is incremented after the ACF_CAN_BRIEF message is assembled inside the EOBUF. The CAN-to-Ethernet ACF Assembler then proceeds to fetch the next available input CAN frame.

The DRE monitors for delayed processing of the CAN frame in CIBUF by the engine. The watchdog timer counter starts incrementing when the Timeout Enable (CWDCFG.EN) is set. The CAN watchdog timer triggers periodic events E1, E2,…,En after a user-configured timeout prescaler CWDCFG.CTO. If the start condition is true and the end condition is not true within two events Ei and Ei+1, the CAN watchdog Timeout Error flag CIBL_STATUS.WDTE is set and the error interrupt INT_8 is triggered if CWDCFG.WTOE is enabled. The buffer with the Timeout error is indicated by the CITO register. Whenever there is a timeout, the corresponding CIBUF is taken out of the arbitration. The software shall clear the Timeout status within the CITO register and also clear the pending request if needed after reading the frame. In case the same CIBUF is to be reused (included again in the next arbitration), the software shall clear only the Timeout status.

The Timeout period between the events Ei is calculated as follows:

- Default Timeout period: $T_{def} = \frac{16}{f_{SRI}}$
- User-configured Timeout period: $T_{out} = T_{def} * (CWDCFG.CTO + 1)$

The table below shows the conditions which trigger a Timeout interrupt.

Table 937 Timeout interrupt trigger

Timeout check	Timeout interrupt trigger
End condition is true within a single event Ei	No Timeout error. No Timeout interrupt triggered
End condition is true within two consecutive events Ei and Ei+1	If the presence of start condition is detected at Ei, then the end condition is expected to happen before Ei+1. If not, the Timeout interrupt is triggered
End condition is true after two consecutive events Ei and Ei+1	Timeout interrupt is triggered

The table below shows the start and end conditions for the CIBUF.

Table 938 CIBUF : Start and end condition list

Start condition	End condition
Pending request is set for the CAN frame in an individual buffer CIBUF. The CIBL_BPR.PRj indicates the pending requests	Pending request is cleared for the CIBUF

Checking the Ethernet Output Buffer full condition

The maximum size of the ACF Payload section is configured at EOBUFj_CONFIG.PL. Before storing a new ACF_CAN_BRIEF frame into an Ethernet Output Buffer, if the EOBUFj_NTSCF_H0.NTSCFDL will exceed the configured EOBUF payload size (EOBUFj_CONFIG.PL), the Buffer Full flag (EOBUFj_STATUS.BF) is set by the hardware and an interrupt is triggered. There is no transmit ready from the DRE in case of Buffer Full scenario. The software shall use the Buffer Full interrupt to set the TXRDY.

The hardware then checks for the next available Ethernet Output Buffer with a matching Destination ID. If available, the ACF_CAN_BRIEF message is stored in that buffer, otherwise the ACF_CAN_BRIEF message processing is aborted and the pending request flag (CIBL_BPR.PRj) of the corresponding CAN Input Buffer is not cleared.

Write operation to the Ethernet header section

The Transmit Trigger Unit controls the Transmit Ready (EOBUFj_STATUS.TXRDY) condition of each Ethernet Output Buffer. The CAN-to-Ethernet ACF Assembler continues to stack ACF_CAN_BRIEF messages to the Ethernet Output Buffer until EOBUFj_STATUS.TXRDY is set by the Transmit Trigger Unit. When EOBUFj_STATUS.TXRDY is set, the CAN-to-Ethernet ACF Assembler prepares the Ethernet header section of the Ethernet Output Buffer as follows:

1. When EOBUFj_CONFIG.HE is set, the contents of the Ethernet header (EOBUFj_MAC_H0, EOBUFj_MAC_H1, EOBUFj_MAC_H2, EOBUFj_MAC_H3 and EOBUFj_MAC_H4) are written to the header section of the Ethernet Output Buffer, starting at offset address of 0_H from the EOBUF start address
2. The NTSCF header (EOBUFj_NTSCF_H0, EOBUFj_NTSCF_STREAM0_ID and EOBUFj_NTSCF_STREAM1_ID) is written to the NTSCF header section of the Ethernet Output Buffer, starting at an offset address of 14_H from the EOBUF start address
3. The Sequence Number (EOBUFj_NTSCF_H0.SN) is incremented by 1. Upon overflow, the Sequence Number is wrapped around to 0
4. The Ethernet Transmit Request bit for the corresponding EOBUF is set in EREQ.TXi_REQ
5. The assembled Ethernet frame is handed over to the Ethernet Descriptor Handler which triggers an interrupt to the Interrupt Router module or prepares the Tx descriptors depending on the TETHDLi_CTRL.TRIG

20.5.3.3 Ethernet ACF-to-CAN disassembler

This section describes the extraction of CAN frames from the input ACF frame.

Related information

Ethernet input buffer on page 3733

20.5.3.3.1 Ethernet input frame

The received Ethernet frames are stored in the Ethernet Input Buffer by the ETH DMA channel or the software. A new received Ethernet frame in the Ethernet Input Buffer is indicated by the pending request set by the ETH Descriptor Handler. The Ethernet Input Buffer with a pending request set is processed by the ACF-to-CAN Disassembler. The Buffer 0 has the highest priority while the Buffer 5 has the lowest priority.

Validation of the Ethernet input frame

The Ethernet input frame is checked for a valid subtype and stream_id in the NTSCF header.

Header type validation

The subtype field starts at offset 0 of the NTSCF header field and is 8 bits long. The value of the subtype field should be 82_H, which corresponds to the Non-Time-Synchronous Control format.

In case of a mismatch to the subtype field, it is checked if the Ethernet frame needs to be forwarded to an Ethernet destination by performing the Forward ID filtering within the forwarding table (refer to the Forwarding Engine chapter). When there is no match, the NTSCF Header Format Error flag is set. It is considered an Ethernet Frame error (EIBUFi_STATUS.FE=1), as the Ethernet input frame cannot be processed further. In this case, the FE flag is set and the pending request is cleared by the hardware.

Stream ID validation

The stream_id field starts at offset 04_H from the start of the NTSCF header and is 64 bits long. The stream_id value is checked against the Stream-ID filter configuration for acceptance of the Ethernet frame and to determine the CAN Transmit Routing Table to be used for the routing operation of the CAN frames contained within the Ethernet input ACF frame. The CAN Transmit Routing Table is identified by the SIDFi_FC.RTI configuration.

The Stream ID filters support up to 8 filter configurations. The filter mode can be configured as Range filter or Mask filter using SIDFi_FC.MODE. In Range filter mode, the Stream ID of the Ethernet input frame must fall within the values between Filter 1 (SIDFi_FIL1_L, SIDFi_FIL1_H) and Filter 2 (SIDFi_FIL2_L, SIDFi_FIL2_H) in order to pass. In Mask filter mode, the Stream ID of the Ethernet input frame is checked for a match with the Filter 1 configuration and the Filter 2 value is used as mask for the compare operation. The filtering operation is terminated with the first matching filter element. When no configured filter is matched, the Ethernet input frame is checked if it needs to be forwarded to an Ethernet destination by performing the Forward ID filtering within the Forwarding Table (refer to the Forwarding Engine chapter). When there is still no match, the Invalid Destination ID flag (EIBUFi_STATUS.IDID) is set and an interrupt INT_10 is triggered. The software fetches the Ethernet frame from the buffer and clears the pending request.

***Note**: In case RETHDLi_CTRL.TRIG is set to 1, the Forward ID filtering is not performed when the Stream ID filtering returns no match. The Invalid Destination ID flag (EIBUFi_STATUS.IDID) is set and an interrupt INT_10 is triggered.*

Related information

Forwarding engine on page 3787

20.5.3.3.2 ACF_CAN_BRIEF message format decoder

The ACF_CAN_BRIEF messages contained in the Ethernet input frame ACF payload are extracted and formatted to the CAN frame format of the CAN Output Buffer. In addition, the routing destination of each extracted CAN frame is requested from the CAN Transmit Routing Engine.

Fetch ACF_CAN_BRIEF

ACF_CAN_BRIEF messages start at offset C_H from the NTSCF start address. Initially, the offset address of the ACF_CAN_BRIEF message to be fetched is indicated in EIBUFi_STATUS.ACF_CAN_ADR. After the fetched ACF_CAN_BRIEF message is processed, the offset address of the next ACF_CAN_BRIEF message is determined by the acf_msg_length value of the last fetched message.

$$\text{New ACF_CAN_ADDR} = \text{ACF_CAN_ADDR} + \text{acf_msg_length} << 2_H$$

For each fetched ACF_CAN_BRIEF message, the length of the ACF payload is incremented as follows:

$$\text{ACF_Payload_Length} = \text{ACF_Payload_Length} + \text{acf_msg_length} << 2_H$$

ACF_CAN_BRIEF messages are fetched from the Ethernet Input Buffer until the calculated ACF_Payload_Length is equal to the Ntscf_data_length value of the NTSCF header. If it is greater, then the fetched ACF_CAN_BRIEF message is discarded, the AVTP Length Mismatch Error flag (EIBUFi_STATUS.LME) is set, the Ethernet input frame is discarded and the EIBUF pending request is cleared.

Validation of ACF_CAN_BRIEF

Each fetched ACF_CAN_BRIEF message is validated for Frame Format and Remote Frame errors before it is decoded.

Frame format error

The acf_msg_type value of the ACF_CAN_BRIEF message shall be 02_H. In case of a mismatch, the CAN Frame Format Error flag is triggered, the corresponding Ethernet input frame is dropped and the Pending Request flag is reset.

Remote frame error

When Reject Remote Frame is configured as 1_B and the RTR field of the ACF_CAN_BRIEF frame is 1_B and FDF is set to 0_B, the EIBUFi_STATUS.RFE flag is set and the next ACF_CAN_BRIEF message is fetched (if available). When Reject Remote Frame is configured as 0_B , the frame is copied without any data bytes.

As there are no remote frames in case of CAN FD format, the DRE_COBUFj_R0.RTR bit is ignored if the FDF bit is set to 1_B.

Routing request

The CAN Identifier field of the validated ACF_CAN_BRIEF message and the Routing Table Index is used to request the CAN Transmit Routing Engine in order to identify the destination of the CAN frames.

A new routing request is performed when RREQ_CONFIG.REQ bit is cleared. The CAN Identifier and eff values of the ACF_CAN_BRIEF frame are written to RREQ_CID. The SIDFi_FC.RTI of the matched Stream ID filter is written to the RREQ_CONFIG.RTI. A routing request is initiated by setting the RREQ_CONFIG.REQ bit by hardware.

The CAN Transmit Routing Engine, after identifying a matching routing rule for the requested CAN frame, writes the corresponding routing rule in the Routing header (UCRH or MCRH SFR). The index of the Routing Element used by the CAN Transmit Routing Engine is written to RS.RE. The CAN Transmit Routing Engine acknowledges back the completion of the routing request by clearing the RREQ_CONFIG.REQ bit.

Frame decoder

The ACF_CAN_BRIEF frame is decoded and written to the CAN Output Buffer.

The ACF_CAN_BRIEF message is formatted and written to the available CAN Output Buffer indicated by the Put Index of CAN Output Buffer List.

The ACF_CAN_BRIEF message is formatted as follows:

- **Routing header**

  The Routing header as given by the CAN Transmit Routing Engine is copied to COBUFj_UCRH, COBUFj_MCRH
- **CAN identifier**

  The value of the CAN_ID field of the ACF_CAN_BRIEF message is copied to COBUFj_R0.ID
- **Remote transmit request**

  The value of the RTR field of the ACF_CAN_BRIEF message is copied to COBUFj_R0.RTR
- **Extended CAN identifier**

  The value of the EFF field of the ACF_CAN_BRIEF message is copied to COBUFj_R0.XTD
- **Error state indicator**

  The value of the EDI field of the ACF_CAN_BRIEF message is copied to COBUFj_R0.ESI

- **Data length code**

  The acf_msg_length and pad bit values are translated to a CAN Data Length Code as specified in the ACF_CAN_BRIEF message format encoder section. The translated Data Length Code is written to COBUFj_R1.DLC. In case of an invalid Data Length Code, the CAN Frame Length Error is set and the CAN frame is dropped
- **Bit-rate switch**

  The value of the BRS field of the ACF_CAN_BRIEF message is copied to COBUFj_R1.BRS
- **Frame format**

  The value of the FDF field of the ACF_CAN BRIEF message is copied to COBUFj_R1.FDF
- **CAN payload data**

  The can_msg_payload data bytes as determined by the calculated DLC of the ACF_CAN_BRIEF message are copied to the COBUFj_DB fields. The padded bytes are not written to the CAN Output Buffer

Related information

ACF_CAN_BRIEF message format encoder on page 3748

CAN frame completion

The ACF_CAN_BRIEF message decoder also computes a 16-bit CRC (COBUFj_CRC) over the R0, R1 (except ESI, ANMF, RXTS, FIDX and BRS which are considered as zero), safety critical CAN payload and the DID (which is padded with zeros to make it 32-bits) of the CAN frame stored in the CAN Output Buffer. It uses the CCITT CRC16 polynomial: 0x1021 $x^{16} + x^{12} + x^5 + 1$. The CRC is calculated over 16 bits of data at a time starting with the MSB.

After successful completion of the write operation of the CRC and the CAN frame to the CAN Output Buffer, the corresponding Transmit Request bit is set by hardware. The next ACF_CAN_BRIEF message is fetched and decoded.

The DRE monitors for delayed processing of the CAN frame in COBUF by the Routing Control Unit. The watchdog timer counter starts incrementing when the Timeout Enable (CWDCFG.EN) is set. The CAN watchdog timer triggers periodic events E1, E2,…, En after a user-configured timeout prescaler CWDCFG.CTO. If the start condition is true and the end condition is not true within two events E_i and E_{i+1}, the CAN watchdog Timeout Error flag COBL_STATUS.WDTE is set and error interrupt INT_9 is triggered if CWDCFG.WTOE is enabled. The buffer with the Timeout error is indicated by the COTO0 and COTO1 registers. Whenever there is a timeout, the corresponding COBUF is taken out of the arbitration. The software shall clear the Timeout status within the COTO0 and COTO1 registers and also clear the pending request if needed after reading the frame. In case the same COBUF is to be reused (included again in the next arbitration), the software shall clear only the Timeout status.

The Timeout period between the events E_i is calculated as follows:

- Default Timeout period: $T_{def} = \frac{16}{f_{SRI}}$
- User-configured Timeout period: $T_{out} = T_{def} * (CWDCFG.CTO + 1)$

The table below shows the conditions which trigger a Timeout interrupt.

Table 939 Timeout interrupt trigger

Timeout check	Timeout interrupt trigger
End condition is true within a single event E_i	No Timeout error. No Timeout interrupt triggered

(table continues...)

Table 939 (continued) Timeout interrupt trigger

Timeout check	Timeout interrupt trigger
End condition is true within two consecutive events E_i and E_{i+1}	If the presence of start condition is detected at Ei, then the end condition is expected to happen before Ei+1. If not, the Timeout interrupt is triggered
End condition is true after two consecutive events E_i and E_{i+1}	Timeout interrupt is triggered

The table below shows the start and end conditions for the COBUF.

Table 940 COBUF : Start and end condition list

Start condition	End condition
Pending request is set for the CAN frame in an individual buffer COBUF. The COBL_BPR0 and COBL_BPR1 indicate the pending requests	Pending request is cleared for the COBUF

Related information

Fetch ACF_CAN_BRIEF on page 3754

20.5.3.3.3 Completion of the Ethernet frame

All CAN frames contained within the input ACF frame are considered to be processed when the calculated ACF payload length matches with the NTSCF_data_length value of the NTSCF header. Upon successful completion, the pending request of the Ethernet Input Buffer (EIBUFi_STATUS.BPR) is cleared by hardware.

20.6 CAN transmit routing engine and routing control unit

The CAN Transmit Routing Engine and the Routing Control Unit are responsible for the routing of CAN frames to CAN peripherals.

20.6.1 Feature list

- Up to 4 user-configurable CAN Transmit Routing Tables
- Up to 128 user-configurable routing rules per CAN Transmit Routing Table
- The following routing modes are supported:
  - Uni-cast
  - Multi-cast (up to 4 configurable destination CAN interfaces)
- Up to 128 user-configurable CAN ID-based acceptance or rejection filters per CAN Transmit Routing Table
- Processing of multiple routing requests:
  - Requests from source CAN nodes
  - Requests from the CAN Output Buffer List
- Non-starving arbitration of routing requests
- CAN routing transfer to the following destinations:
  - CAN nodes
  - ACF Ethernet frames
  - System memory
- FPI and SRI transactions by the Move Engine

20.6.2 Functional overview

The figure below shows the block diagram of the CAN Transmit Routing Engine and the Routing Control Unit.

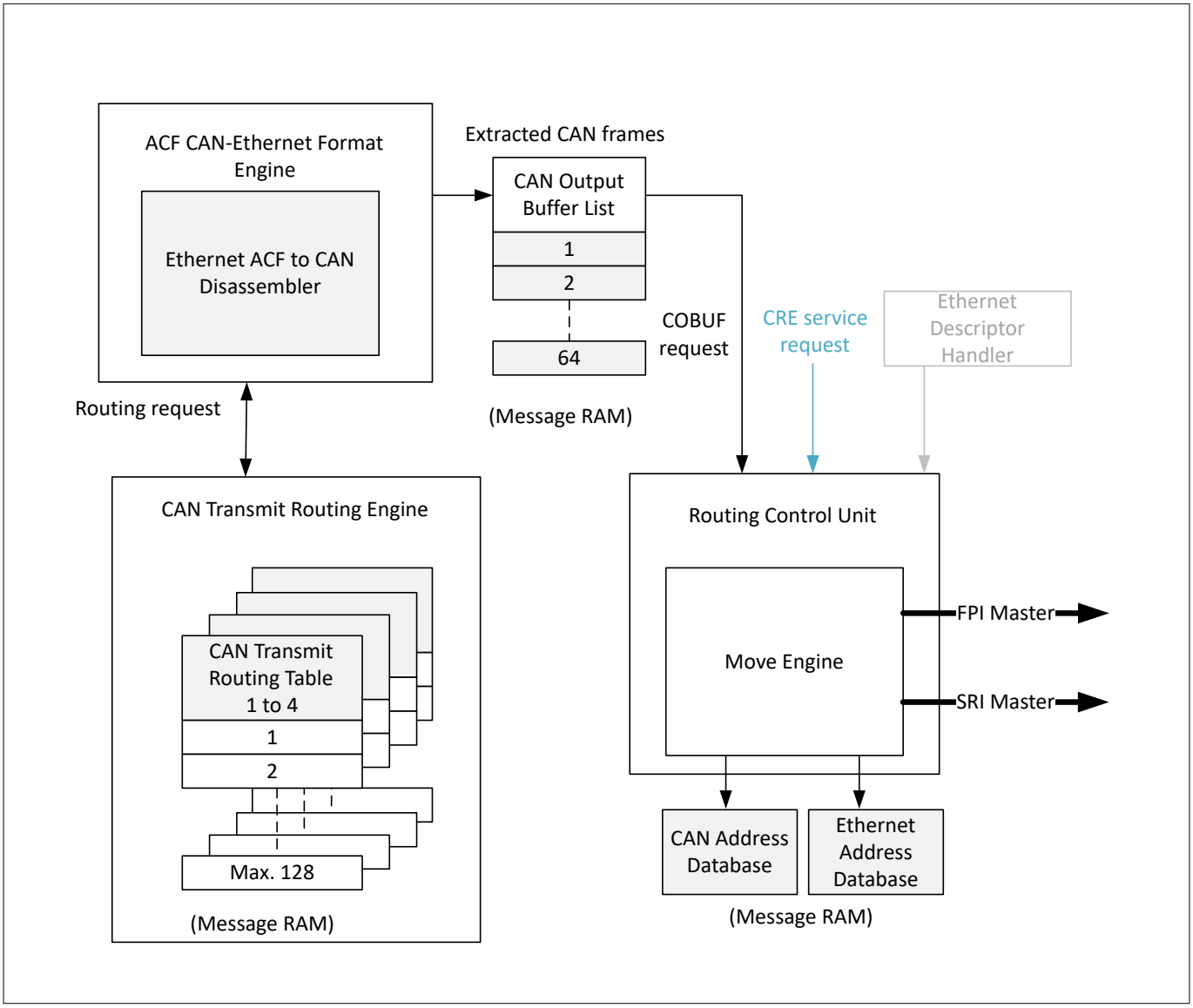


Figure 353 CAN Transmit Routing Engine and Routing Control Unit block diagram

The CAN Transmit Routing Engine is responsible for identifying the transmit destination of a CAN frame in the CAN Output Buffer List (Ethernet to CAN routing use case), whereas the Routing Control Unit assists, together with the Move Engine, in the transfer of CAN frames and Ethernet frames.

20.6.3 Functional description

In this section, the routing of transmit CAN frames and the functions of the CAN Transmit Routing Engine and the Routing Control Unit are described in detail.

20.6.3.1 CAN transmit routing engine

The CAN Transmit Routing Engine is responsible for identifying the transmit destination CAN interface of a CAN frame.

20.6.3.1.1 Initiating routing operation

The routing function of the CAN Transmit Routing Engine is initiated when a CAN Routing Request is set. When the CAN Routing Request is set, the CAN Transmit Routing Engine uses the following registers to identify the destination of the CAN frame:

- RREQ_CONFIG.RTI to identify the CAN Transmit Routing Table (1 to 4)
- RREQ_CID.ID and XTD for acceptance filter scanning

20.6.3.1.2 CAN acceptance filters

Each routing element in a CAN Transmit Routing Table contains the following:

- CAN ID filter (RT_REj_CIDFC (j = 0-127))
- Routing rule (RT_REj_UCR or RT_REj_MCR (j = 0-127))

The CAN ID filter contains the acceptance condition for a given CAN ID. The CAN ID (RREQ_CID.ID) and the Extended Identifier (RREQ_CID.XTD) of the CAN Routing Request are scanned through the CAN ID filters of each routing element. When a match occurs, the corresponding routing rule is used to identify the destination of the requested CAN frame. Disabled routing rules are not used for the filter operation. The scanning operation is performed starting from routing element 0 and ends with the routing element with the first matching CAN ID filter.

The CAN ID filter supports 3 modes of operation (RT_REj_CIDFC.MODE):

- Classical ID filter (MODE = 00_B):
  - RT_REj_CIDFC.CANID1 is used as the CAN ID filter and RT_REj_CIDFC.CANID2 is used as the mask. A mask value with bit positions having "0b" causes the corresponding bits of CAN ID1 to be ignored for matching with the requested CAN ID (RREQ_CID.ID). A mask value of all 1 causes a successful match only if the requested CAN ID is exactly same as the RT_REj_CIDFC.CANID1 value. Similarly, a mask value of all 0 causes any CAN ID value to match
- Dual ID filter (MODE = 01_B):
  - RT_REj_CIDFC.CANID1 or RT_REj_CIDFC.CANID2 is used as the CAN ID filter. The requested CAN ID must exactly match to either RT_REj_CIDFC.CANID1 or RT_REj_CIDFC.CANID2. No masking bits are used
- Range ID filter (MODE = 10_B):
  - The range of IDs between RT_REj_CIDFC.CANID1 and RT_REj_CIDFC.CANID2, inclusive, are used as CAN ID filters. The requested CAN ID must be between RT_REj_CIDFC.CANID1 (inclusive) and RT_REj_CIDFC.CANID2 (inclusive) for a match

On successful identification of the routing element with a matching CAN ID filter, the corresponding routing rule (RT_REj_UCR or RT_REj_MCR) is written to the routing header (UCRH or MCRH) of the CAN frame and the Routing Request flag (RREQ_CONFIG.REQ) is reset to 0. The corresponding routing element index is written to the RS.RE bit-field.

Related information

CAN transmit routing tables (1 to 4) on page 3730

20.6.3.1.3 Error scenarios

Invalid routing table error

When the CAN Transmit Routing Table indexed by the routing request (RREQ_CONFIG.REQ) is disabled or reserved, then the Invalid Routing Table flag (RS.IRT) is set.

Non-matched filter element error

When no matching CAN ID filter is found in the indexed CAN Transmit Routing Table, the Non-Matching Filter Element Error flag (RS.NMFE) is set.

20.6.3.2 Routing control unit

This section describes the handing of the CAN routing operation.

Related information

CAN address database on page 3726

20.6.3.2.1 CAN trigger request

The MCMCAN module generates Receive and Transmit Host Buffer triggers for the DRE.

The trigger request from a CAN node consists of the following signals:

- **TRIGTYPE(0:1):** Indicates the host buffer (Receive Host Buffer 0/1 or Transmit Host Buffer 0) related to the trigger request
- **TRIGNODE(0:1):** Indicates the CAN node within the MCMCAN module related to the trigger request

The table below shows the encoding of the TRIGTYPE trigger.

Table 941 TRIGTYPE description

TRIGTYPE(1)	TRIGTYPE(0)	Description
0_B	0_B	No trigger or idle
0_B	1_B	Transmit Host Buffer 0 free
1_B	0_B	New Receive Host Buffer 0
1_B	1_B	New Receive Host Buffer 1

Although the MCMCAN module provides two Transmit Host Buffers, only the Transmit Host Buffer 0 can be configured to trigger the DRE. CAN Receive Host Buffer 0 or 1 requests from the MCMCAN module indicates to the DRE that a new Receive CAN frame from the corresponding CAN node is pending to be fetched. The DRE in this case sets a pending Receive Request flag for the corresponding CAN node (CANRXR0, CANRXR1). A CAN Transmit Host Buffer 0 request from MCMCAN module indicates to the DRE that the corresponding CAN node has a free Transmit Host Buffer. The DRE in this case sets a pending Transmit Request flag for the corresponding CAN node (CANTXR).

The table below shows the encoding of the TRIGNODE trigger.

Table 942 TRIGNODE description

TRIGNODE(1)	TRIGNODE(0)	Description
0_B	0_B	CANx_N0
0_B	1_B	CANx_N1
1_B	0_B	CANx_N2
1_B	1_B	CANx_N3

20.6.3.2.2 Routing requests

The Routing Control Unit receives routing requests from CAN nodes (through CAN trigger request inputs) and from CAN Output Buffers.

Routing requests from CAN nodes

CAN Receive Host Buffers pending requests (CANRXR0, CANRXR1) are processed by the Routing Control Unit in a round-robin (non-starving) priority scheme, starting with the MCMCAN0 Node 0 request until the maximum number of available CAN node requests is reached. Within a CAN node request, the Receive Host Buffer 0 (CANRXR0.Ci_RH0R, CANRXR1.Ci_RH0R) gets a higher priority than the Receive Host Buffer 1 (CANRXR0.Ci_RH1R, CANRXR1.Ci_RH1R).

CAN node address decoding

The CAN Address Database in the DRE RAM always starts at the DRE RAM base address. The CAN Address Database is configured by the user with the CRE start address (CAD_CANi_CRESA.ADR). An overview of the Receive and Transmit Host Buffers structure and address decoding is given in the figure below. Refer to the CAN chapter for detailed description.

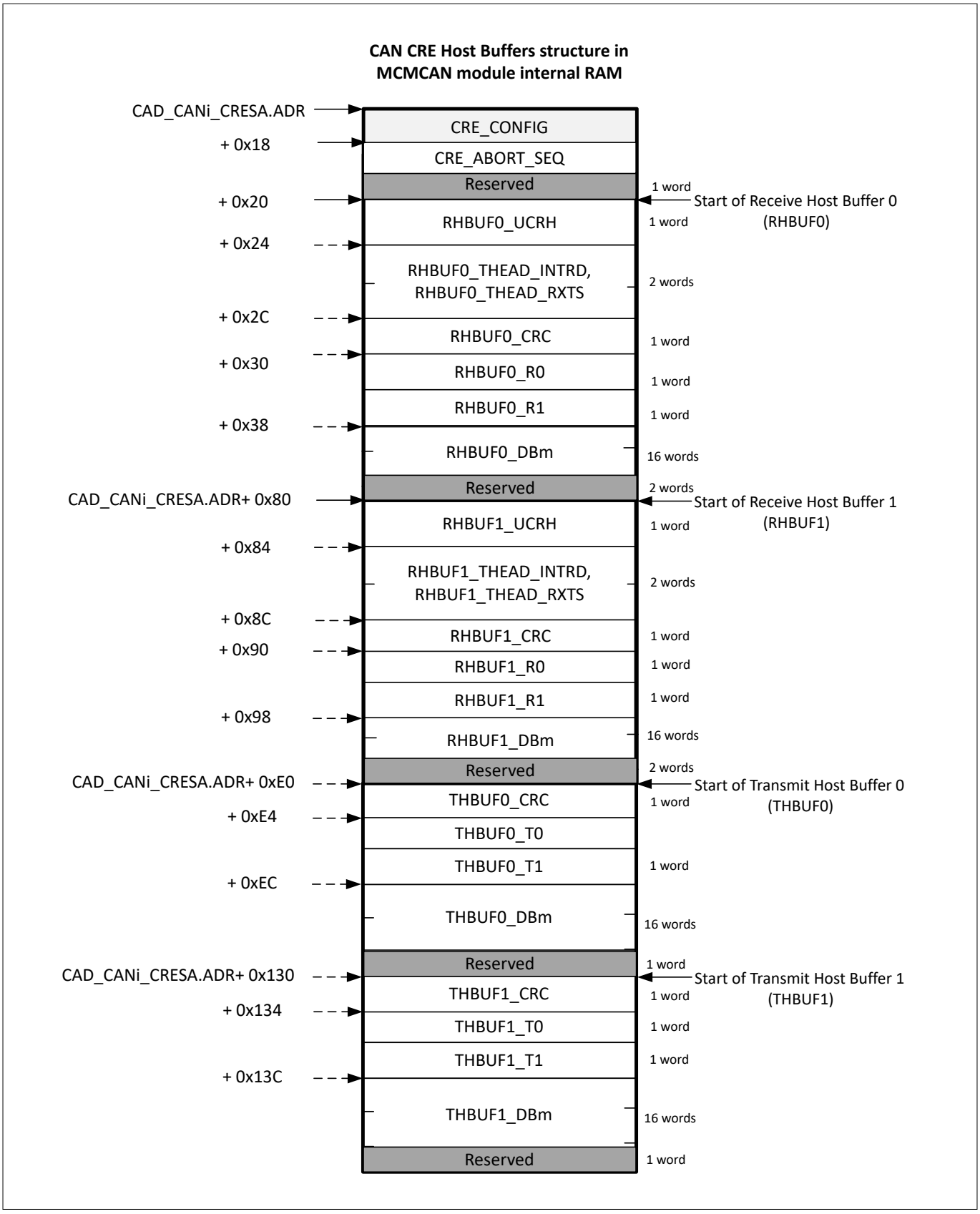


Figure 354 CAN CRE host buffer structure

When a new request from a CAN node is received, the DRE Move Engine transfers the CAN frame from the respective CAN Receive Host Buffer to the corresponding destination(s) as indicated in the routing header (RHBUFk_UCRH).

Requests from RHBUF0

The DRE performs the following after receiving a request from RHBUF0 of a CAN node CANi:

- The Routing header is read from the user-configured CAD_CANi_CRESA.ADR + 20_H address
- The Timing header (2 words) is read from the CAD_CANi_CRESA.ADR +24_H address
- The CRC header (1 word) is read from the CAD_CANi_CRESA.ADR +$2C_H$ address
- The CAN header or PDU header (2 words) RHBUF0_R0 and R1 are read from the CAD_CANi_CRESA.ADR +30_H address
- The CAN data bytes RHBUF0_DBm are read from the CAD_CANi_CRESA.ADR +38_H address. The number of data bytes to be read is defined by the Data Length Code (DLC) of the CAN frame (RHBUF0_R1.DLC)

Requests from RHBUF1

The DRE performs the following after receiving a request from RHBUF1 of CAN Node CANi:

- The Routing header is read from the user-configured CAD_CANi_CRESA.ADR +80_H address
- The Timing header (2 words) is read from the CAD_CANi_CRESA.ADR +84_H address
- The CRC header (1 word) is read from the CAD_CANi_CRESA.ADR +$8C_H$ address
- The CAN header or PDU header (2 words) RHBUF1_R0 and R1 are read from the CAD_CANi_CRESA.ADR +90_H address
- The CAN data bytes RHBUF1_DBm are read from the CAD_CANi_CRESA.ADR +98_H address. The number of data bytes to be read is defined by the Data Length Code (DLC) of the CAN frame (RHBUF1_R1.DLC)

The fetched CAN frame is routed to the assigned destination(s) as described in the chapter Routing transfer. After a successful routing transfer, the CAN Receive Host Buffer pending requests (CANRXR0.Ci_RH0R, Ci_RH1R, CANRXR1.Ci_RH0R, Ci_RH1R) are cleared by hardware.

Related information

Routing transfer on page 3765

Routing requests from CAN output buffers

CAN Output Buffers store CAN frames to be transmitted to CAN nodes. Pending CAN frames in the CAN Output Buffer are indicated by COBL_BPR0 or COBL_BPR1. Among pending CAN frames, the highest priority CAN frame which has all its destinations available is selected for routing transfer. The priority of the CAN frame is defined by the corresponding CAN ID. CAN frames with the lowest CAN ID value have the highest priority as defined by ISO 11898-1. When multiple buffers have CAN frames of equal priority with the same CAN ID, the CAN frame in the buffer with the lowest index is selected for routing transfer.

The Routing Control Unit determines the availability of a destination according to the following:

- A CAN node as a destination is said to be available when the corresponding Transmit Host Buffer request bit (CANTXR.Ci_THR) is set. The DRE always writes only into the Transmit Host Buffer 0 (CAD_CANi_CRESA.ADR +$E0_H$)

  ***Note**:* *In the MCMCAN module, the CRE Tx Host Buffer 0 has to be configured to trigger the DRE by setting Ni_CRE_HBUF_TX0_CONFIG.INTEN to 0_B*

- An Ethernet frame as a destination is said to be available when the CAN Input Buffer List has free buffers (CIBL_STATUS.BF = 0)
- A memory destination is said to be available when enabled in DMEMi_CONFIG.EN = 1

After successful routing transfer, the pending request of the CAN Output Buffer (COBL_BPR0.PRj, COBL_BPR1.PRj) is cleared by hardware.

Related information

CAN output buffer pending request 0 on page 3816
CAN output buffer list on page 3728

Priority between requests from CAN nodes and CAN output buffers

When CAN Receive Host Buffers and CAN Output Buffers have pending requests at the same time, the routing requests are serviced alternately between them, giving them equal priority. The following figure shows an example with 4 CAN nodes (C0 to C3) with 2 pending receive requests and 4 CAN Output Buffers with 3 pending requests.

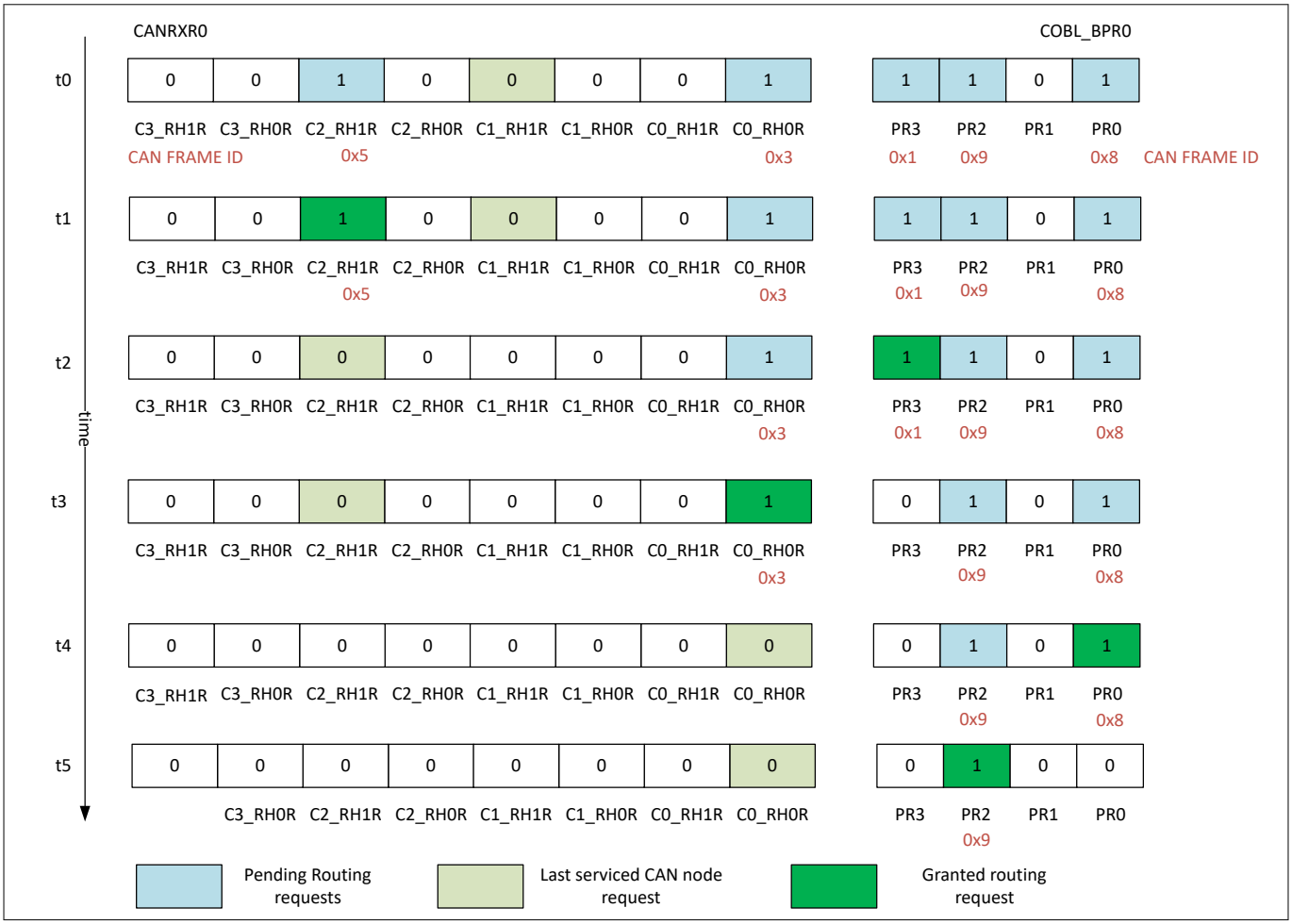


Figure 355 An example of priority handling between CAN receive requests and CAN output buffer requests

In the previous figure, it can be assumed that before time 't0', the pending request PR1 in the CAN Output Buffer List is serviced, then at 't1', the CAN Receive Host Buffer pending requests are granted an opportunity to perform routing transfer. Since the last serviced CANRXR0 is C1_RH1R, the next pending request in the round-robin scan is C2_RH1R. Therefore, the C2_RH1R routing request is serviced at 't1'. After successful routing operation, the CAN Output Buffer List pending requests are handled. At time 't2', the PR3 request is granted since the corresponding CAN frame has the highest priority CAN ID (ID=0x1) among the pending requests in the CAN Output Buffer List (PR0 & PR2). Similarly, at 't3' and 't4' CAN receive requests and CAN Output Buffer

pending requests are serviced alternately. At 't5', since there are no more pending CAN receive requests, the CAN Output Buffer List pending request PR2 is chosen again for routing transfer.

In case of any error (such as bus error, IRDE or DBOE), where the ongoing transaction has to be canceled by the DRE, the DRE writes into the CRE_ABORT_SEQ at the CAD_CANi_CRESA.ADR + 18_H address. The abort sequence is triggered by the DRE only after it starts reading the CAN payload and before the last read or write operation is launched.

Related information

Routing requests from CAN nodes on page 3761

Routing requests from CAN output buffers on page 3763

20.6.3.2.3 Routing transfer

The routing transfer function moves CAN frames from a source to their corresponding destination(s).

CAN routing header

The destination of a CAN frame is indicated by the routing header of the corresponding CAN frame. When a routing transfer request for a pending CAN frame is received, the routing header is read by the Routing Control Unit to identify the destination of the CAN frame.

The following CAN routing header types are supported:

- Uni-cast routing: In the case of CAN to CAN routing, CAN to Ethernet routing and vice versa
- Multi-cast routing: In the case of Ethernet to CAN routing (by the DRE). The multi-cast routing of CAN frames is performed as a sequence of uni-cast routing operations by the CRE
- PDU routing: In the case of CAN to memory routing

Uni-cast routing

In uni-cast routing, the CAN frame is routed to only a single destination. It is indicated in the routing header with MODE = 00_B.

Multi-cast routing

In multi-cast routing, the CAN frame is routed to up to 4 destinations. It is indicated in the routing header with MODE = 01_B.

Multi-cast routing is supported by the DRE for Ethernet to CAN routing. Multi-cast routing for CAN frames is handled by the CRE.

The multi-cast rule must follow the below configuration rules:

- Unused DIDs are set to zero
- At least DID0 and DID1 should have valid destination IDs (1_H to 14_H)
  - Invalid configuration: DID0 or/and DID1 are set to zero
- All valid DIDs are different. If the DIDs are the same, the CAN frame is routed only once
- No gaps are allowed between valid DIDs. For example:
  - Valid configuration: DID0, DID1 and DID2 are valid
  - Invalid configuration: DID0, DID1 and DID3 are valid. DID2 is set to zero

The routing transfer is performed in order from destination 1 to destination 4. The CAN frame is routed only when all relevant destinations are available. Destinations with DID=0 are ignored. The IRDE bit-field is set when there are no valid DIDs for a CAN frame and the routing operation is aborted. The DRE writes into the

CRE_ABORT_SEQ at the CAD_CANi_CRESA.ADR + 18_H address. The abort sequence is triggered by the DRE only after it starts reading the CAN frame and before the last read or write operation is launched. The DRE cancels the ongoing read sequence by writing 1_B to CRE_ABORT_SEQ.CRHBUF0 or CRHBUF1 if it has already started reading the CAN frame (R0, R1, DBm) after the error has occurred. The DRE cancels the ongoing write sequence to Tx Host Buffer 0 by writing 1_B to CRE_ABORT_SEQ.CTHBUF0 if it has already started writing the CAN frame (T0, T1, DBm) after the error has occurred.

In multi-cast routing, the Ethernet ACF-to-CAN Disassembler computes a 16-bit CRC over the R0, R1 (except ESI, ANMF, RXTS, FIDX and BRS which are considered as zero) and the safety critical CAN payload (without DID). This intermediate CRC is stored in the COBUF. While writing the frame from the COBUF to the Tx Host Buffer 0, at the moment of writing the CRC, the CRC is re-computed by the DRE including the DID0. The same is done for all 4 DIDs (DID0 to DID3) before the corresponding write operations.

PDU routing

In PDU routing, the CAN I-PDU is routed to a single destination. It is indicated in the uni-cast routing header with MODE = 10_B. The CAN I-PDU is routed to the system memory location indicated by the DID bit in the routing header.

Routing destinations

This section describes the possible routing destinations for a CAN frame.

A CAN frame can have the following routing destinations:

- CAN node
- ACF Ethernet frame
- User-configured memory location

Each destination is allocated with a 6-bit unique ID. The table below shows the IDs relevant for each destination type and the respective routing transfer destinations.

Table 943 Routing destinations ID

Destination	ID (6 bits)	Transfer destination
Destination disabled	0_H	-
MCMCAN0_Ni (i=0:3)	1_H to 4_H	CAN Transmit Host Buffer 0 (MCMCAN0_Ni_THBUF0)
MCMCAN1_Ni (i=0:3)	5_H to 8_H	CAN Transmit Host Buffer 0 (MCMCAN1_Ni_THBUF0)
MCMCAN2_Ni (i=0:3)	9_H to C_H	CAN Transmit Host Buffer 0 (MCMCAN2_Ni_THBUF0)
MCMCAN3_Ni (i=0:3)	D_H to 10_H	CAN Transmit Host Buffer 0 (MCMCAN3_Ni_THBUF0)
MCMCAN4_Ni (i=0:3)	11_H to 14_H	CAN Transmit Host Buffer 0 (MCMCAN4_Ni_THBUF0)
ACF Ethernet frames	18_H to $1D_H$	DRE CAN Input Buffer List
System memory locations (1 to 28)	20_H to $3B_H$	User-configured by DMEM (0-27)
Reserved for future extension	Others	-

When the DID in the routing header from the CRE does not belong to any valid destination, the IRDE bit is set by the DRE and an interrupt is triggered.

CAN node as destination (ID = 1_H to 14_H)

When the destination of the CAN frame is a CAN node (Destination ID = 1_H to 14_H), then the CAN frame is transferred to the CAN Transmit Host Buffer 0 of the destination CAN node that is configured for the DRE.

The following requests can have a CAN node as destination:

- Requests from a source CAN node
- Requests from the CAN Output Buffer List

Requests from a source CAN node

The following occurs if the source request is from a CAN node (CANRXR0, CANRXR1):

1. The pending request of the Transmit Host Buffer 0 (CANTXR) of the destination CAN node is checked
2. If a pending request for the destination CAN node is set (CANTXR), the CAN frame starting from RHBUFk_R0 until the data bytes defined by DLC is read from the source and written to the destination CAN frame Transmit Host Buffer 0 (the length of data bytes to be written is defined by DLC). The source CAN node request (CANRXR0, CANRXR1) is cleared when the DRE starts reading the RHBUFk_R0
3. After a successful transfer to the destination CAN Transmit Host Buffer 0, the destination Transmit Host Buffer 0 request (CANTXR.Ci_THR) is cleared
4. If the destination CAN node has its Transmit Host Buffer 0 pending request cleared, and if a CAN Output Buffer is free (COBL_STATUS.BF = 0), then the CAN frame along with the routing header is transferred to the CAN Output Buffer and the corresponding COBL_BPR1 and COBL_BPR0 pending request is set. In this case, the source request (CANRXR0, CANRXR1) is cleared after the CAN frame is written successfully to the CAN Output Buffer. Otherwise, the source CAN frame is aborted from the routing transfer, without clearing the source request
5. During a routing transfer, if the length of the source CAN frame is bigger than the destination buffer (Transmit Host Buffer 0 size or CAN Output Buffer size), the source CAN frame is dropped with the Destination Buffer Overflow Error flag set (ME_ERR.DBOE). In this case, the corresponding source request is cleared

Requests from the CAN output buffer list

The following occurs if the source request is from the CAN Output Buffer List (COBL_BPR1 and COBL_BPR0):

1. The pending request of the Transmit Host Buffer 0 (CANTXR) of the destination CAN node is checked
2. If a pending request for a destination CAN node is set (CANTXR), the CAN frame starting from COBUFj_R0 until the data bytes defined by COBUFj_R1.DLC are read from the source and written to the destination CAN node Transmit Host Buffer 0
3. After a successful transfer to the destination CAN Transmit Host Buffer 0, the source request (CAN Output Buffer request) and the destination Transmit Host Buffer 0 request (CANTXR.Ci_THR) are cleared
4. If the destination CAN node has the Transmit Host Buffer 0 pending request cleared, the source CAN frame is aborted from routing transfer without clearing the source request

ACF Ethernet frame as destination (ID = 18_H to $1D_H$)

When the destination of the CAN frame is an ACF Ethernet frame (Destination ID = 18_H to $1D_H$), then the CAN frame is transferred to an available CAN Input Buffer indexed by CIBL_STATUS.PIDX. The CAN frame starting from the RHBUFk_CRC, RHBUFk_R0 until the data bytes defined by RHBUFk_R1.DLC is read from the source and written to the CAN Input Buffer. After a successful transfer, the source request (CANRXR0 and CANRXR1) is cleared and the pending request in the corresponding CAN Input Buffer (CIBL_BPR) is set.

If the CAN Input Buffer List is full (CIBL_STATUS.BF=1), then the source CAN frame is aborted from routing transfer without clearing the source request.

System memory as destination (ID = 20_H to $3B_H$)

When the destination of the CAN message is a system memory location (Destination ID = 20_H to $3B_H$) and the corresponding destination memory (DMEMi) is enabled (DMEMi_CONFIG.EN1=1_B), then the CAN message is transferred to the system memory location with the starting address (DMEMi_SA.ADR) configured by the user in the DMEM parameter table. The index i of DMEMi is derived from the destination ID (i = destination ID - 20_H).

The user should configure the following levels for each DMEM in the DMEM parameter table:

- Watermark (DMEMi_WM) if required
- Wraparound (DMEMi_WA)

The user should also configure the CAN message type being received by a DMEM (DMEMi_CONFIG.CTYP). The following CAN message types are supported:

- CAN frames (DMEMi_CONFIG.CTYP = 0_B)
- CAN I-PDUs (DMEMi_CONFIG.CTYP = 1_B)

Related information

DMEM parameter table on page 3741

CAN message buffer layout

The user can create a circular list of CAN Message Buffers to store CAN messages in the DMEM. The following figure shows the CAN Message Buffer layout for CAN frames (DMEMi_CONFIG.CTYP = 0) within the DMEMi.

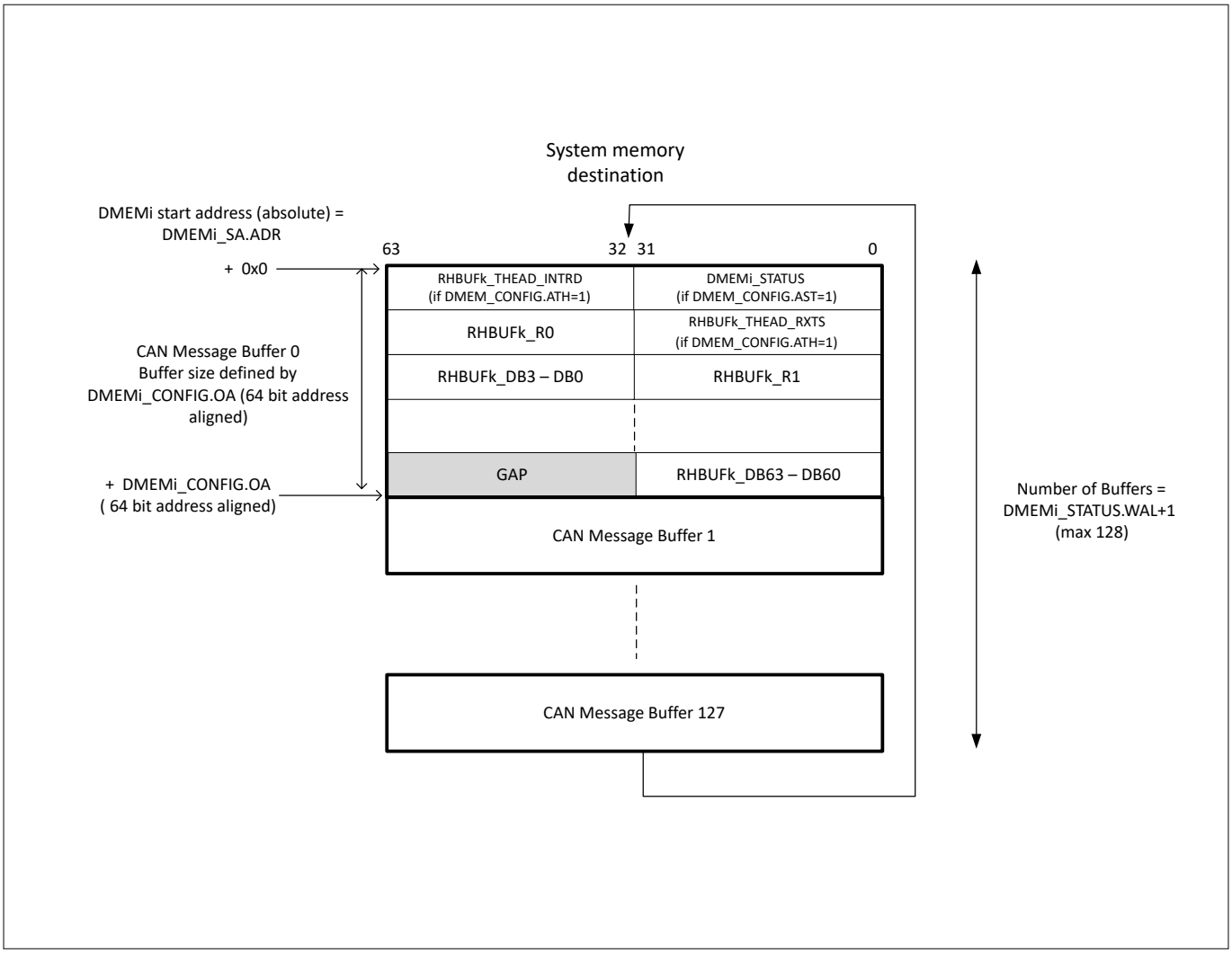


Figure 356 CAN Message Buffer layout for CAN frames

The CAN Message Buffer layout must have the following characteristics:

- The size of the CAN Message Buffers is defined by DMEMi_CONFIG.OA
- The total number of CAN Message Buffers in the circular list is defined by the configured Wraparound level (DMEMi_WA.WAL+1)
- The CAN Message Buffer Index at which the next CAN message is to be stored is shown in DMEMi_STATUS.BC
- The start address of each CAN Message Buffer can be identified as DMEMi_SA.ADR + (DMEMi_STATUS.BC * DMEMi_CONFIG.OA)

***Note**: The Watermark level (DMEMi_WM) must always be smaller than the Wraparound level.*

CAN message buffer layout mode

The user can configure DMEMi_MODE.BUF to set the layout mode of the CAN Message Buffer. The following layout modes are supported:

- Single (DMEMi_MODE.BUF = 0_B)
- Continuous (DMEMi_MODE.BUF = 1_B)

These layout modes apply for DMEMs configured for both CAN message types, CAN frames (DMEMi_CONFIG.CTYP = 0_B) and CAN I-PDUs (DMEMi_CONFIG.CTYP = 1_B).

Single mode

In Single mode (SINGLE, DMEMi_MODE.BUF = 0_B), the CAN Message Buffer is split into the following two buffers:

- Frame Data Buffer 1 (based on the Watermark level DMEMi_WM)
- Frame Data Buffer 2 (based on the Wraparound level DMEMi_WA)

Both Frame Data Buffers always start at the 64-bit aligned address. The DRE triggers a Watermark and a Wraparound interrupt when the set DMEMi_WM and DMEMi_WA levels are reached.

Single mode supports the following trigger modes:

- Index-based trigger (INDEX, DMEMi_MODE.TRIG = 0_B): This mode is valid only for CAN I-PDUs. For CAN frames this bit is ignored and the Count-based trigger mode is used. In the Index-based trigger mode, the Watermark and Wraparound interrupts are triggered based on the DMEMi_FDBI value. The DMEMi_WM and DMEMi_WA levels are configured as words in this case
- Count-based trigger (COUNT, DMEMi_MODE.TRIG = 1_B): In this mode, the Watermark and Wraparound interrupts are triggered based on the DMEMi_STATUS.BC value. The DMEMi_WM and DMEMi_WA levels are configured as number of CAN frames or CAN I-PDUs. The size of Frame Data Buffer 1 and Frame Data Buffer 2 are allocated based on DMEMi_WM, DMEMi_WA and DMEMi_CONFIG.OA

Continuous mode

In Continuous mode (CONT, DMEMi_MODE.BUF = 1_B), the CAN Message Buffer is considered as a whole buffer and the CAN messages are assembled continuously until the Wraparound level (DMEMi_WA) is reached.

Continuous mode supports the following trigger modes:

- Index-based trigger (INDEX, DMEMi_MODE.TRIG = 0_B): This mode is valid only for CAN I-PDUs. For CAN frames this bit is ignored and the Count-based trigger mode is used. In the Index-based trigger mode, the Wraparound interrupt is triggered based on the DMEMi_FDBI value. The DMEMi_WA level is configured as words in this case
- Count-based trigger (COUNT, DMEMi_MODE.TRIG = 1_B): In this mode, the Wraparound interrupt is triggered based on the DMEMi_STATUS.BC value. The DMEMi_WA level is configured as number of CAN frames or CAN I-PDUs. The size of Frame Data Buffer 1 is allocated based on DMEMi_WA and DMEMi_CONFIG.OA

CAN message buffer packing

DMEMi_MODE.TYP enables the user to configure the CAN Message Buffer packing type. The following packing types can be configured:

- Multiplexing disabled (MUXDIS, DMEMi_MODE.TYP = 0_B): The CAN messages (CAN frames or CAN I-PDUs) are packed in the same way as shown in the figure CAN Message Buffer layout for CAN frames (see above). The size of the CAN Message Buffer is defined by DMEMi_CONFIG.OA. In this case, there will be padding bits between the CAN messages as each CAN message starts at a 32-bit aligned address and no FDBI is present to indicate the exact length
- Multiplexing enabled (MUXEN, DMEMi_MODE.TYP = 1_B): This mode is valid only for CAN I-PDUs. The CAN I-PDUs are packed without any padding bits in between them. This mode can be used when the CAN I-PDUs are to be multiplexed into a frame, for example, an Ethernet frame. The size of the assembled frame is defined by DMEMi_FDBI which is incremented by the size of the CAN I-PDU after each CAN I-PDU is transferred to the DMEM

Related information

CAN message buffer layout on page 3768

Destination memory transfer mode configurations

The following table shows a summary with the possible applicable DMEMi transfer mode configurations (DMEMi_MODE) depending on the CAN message type being received by the DMEMi (DMEMi_CONFIG.CTYP).

Table 944 Applicable DMEMi transfer mode configurations

DMEMi_MODE			DMEMi_CONFIG.CTYP		Description
TYP	TRIG	BUF	CAN frame[1)]	CAN I-PDU[1)]	
MUXDIS	COUNT	CONT	✔	✔	• Space between CAN frames/CAN I-PDUs depending on the DLC • Interrupt trigger based on the number of CAN frames/CAN I-PDUs • CAN frames/CAN I-PDUs are assembled continuously with no DMEMi division
MUXDIS	COUNT	SINGLE	✔	✔	• Space between CAN frames/CAN I-PDUs depending on the DLC • Interrupt trigger based on the number of CAN frames/CAN I-PDUs • DMEMi divided into FDB1 and FDB2. CAN frames/CAN I-PDUs are assembled first in FDB1 followed by FDB2
MUXDIS	INDEX	CONT	✔	✔	• Space between CAN frames/CAN I-PDUs depending on the DLC • Interrupt trigger based on the index (FDBI) • CAN frames/CAN I-PDUs are assembled continuously with no DMEMi division
MUXDIS	INDEX	SINGLE	✔	✔	• Space between CAN frames/CAN I-PDUs depending on the DLC • Interrupt trigger based on the index (FDBI) • DMEMi divided into FDB1 and FDB2. CAN frames/CAN I-PDUs are assembled first in FDB1 followed by FDB2
MUXEN	COUNT	CONT	✘	✔	• No space between CAN I-PDUs • Interrupt trigger based on the number of CAN I-PDUs • CAN I-PDUs are assembled continuously with no DMEMi division
MUXEN	COUNT	SINGLE	✘	✔	• No space between CAN I-PDUs • Interrupt trigger based on the number of CAN I-PDUs • DMEMi divided into FDB1 and FDB2. CAN I-PDUs are assembled first in FDB1 followed by FDB2

(table continues...)

Table 944 (continued) Applicable DMEMi transfer mode configurations

DMEMi_MODE			DMEMi_CONFIG.CTYP		Description
TYP	TRIG	BUF	CAN frame[1)]	CAN I-PDU[1)]	
MUXEN	INDEX	CONT	✘	✔	• No padding bytes between CAN I-PDUs • Interrupt trigger based on the index (FDBI) • CAN I-PDUs are assembled continuously with no DMEMi division
MUXEN	INDEX	SINGLE	✘	✔	• No space between CAN I-PDUs • The interrupt trigger is based on the index (FDBI) • DMEMi divided into FDB1 and FDB2. CAN I-PDUs are assembled first in FDB1 followed by FDB2

1) ✔ = applicable, ✘ = not applicable

CAN frames to system memory

For every CAN frame routing where DMEMi_CONFIG.CTYP is set to 0_B (CAN frame) by the user, the following tasks are performed:

1. If DMEMi_CONFIG.ATH = 1, the CAN frame Timing headers (RHBUFk_THEAD_INTRD and RHBUFk_THEAD_RXTS) are copied as the first two words of the current CAN Message Buffer
2. If both DMEMi_CONFIG.AST=1 and DMEMi_CONFIG.ATH = 1 are set, DMEMi_STATUS and the Timing headers are copied
3. The CAN frame is then transferred to destination memory starting from RHBUFk_R0 until the end of the CAN data bytes defined by RHBUFk_R1.DLC and aligned to 32-bit words
4. The following occurs after a successful transfer:
   a. If DMEMi_WM.WML = DMEMi_STATUS.BC (except when WML=0), the Watermark flag is set (DMEMi_STATUS.WMF = 1) and an interrupt is triggered
   b. If DMEMi_WA = DMEMi_STATUS.BC, the Wraparound flag is set (DMEMi_STATUS.WAF = 1) and an interrupt is triggered. After the wraparound event, the next CAN frame is stored again at the start address DMEMi_SA.ADR, making the destination a circular list
   c. The buffer counter (DMEMi_STATUS.BC) is increased by 1. When a Wraparound event occurs, the buffer counter is reset to 0. The BC value is still zero within the DMEMi after the first message is transferred. The register DMEMi_STATUS indicates the correct count value
   d. The message counter (DMEMi_STATUS.MC) is increased by 1. After overflow of the counter, the counter is reset to 0
5. DMEMi_STATUS.SID is updated with the source ID of the CAN frame and the New Message bit (DMEMi_STATUS.NM) is set to 1
6. If only DMEMi_CONFIG.AST = 1 is set, DMEMi_STATUS is copied as the first word of the current virtual CAN buffer after the frame is copied

Consider the following when transferring CAN frames to the system memory:

- The message counter (DMEMi_STATUS.MC) resets to 0 either after a counter overflow or after a software write with a non-zero value
- If the CAN frame (including Status and Timing headers, if enabled) does not fit the destination buffer size defined by DMEMi_CONFIG.OA, the Destination Buffer Overflow Error flag (ME_ERR.DBOE) is set and the source CAN request is cleared. The ongoing transaction has to be canceled by the DRE by writing into the CRE_ABORT_SEQ at the CAD_CANi_CRESA.ADR + 18_H address.

- If a CAN I-PDU is received instead of a CAN frame where DMEMi_CONFIG.CTYP is set to 0_B, the CAN I-PDU is discarded, the Invalid Routing Destination Error flag (ME_ERR.IRDE) is set and the source CAN request is cleared. The ongoing transaction has to be canceled by the DRE by writing into the CRE_ABORT_SEQ at the CAD_CANi_CRESA.ADR + 18_H address. The abort sequence is triggered by the DRE only after it starts reading the CAN frame and before the last read or write operation is launched. The DRE cancels the ongoing read sequence by writing 1_B to CRE_ABORT_SEQ.CRHBUF0 or CRHBUF1 if it has already started reading the CAN frame (R0, R1, DBm) after the error has occurred.
- The DRE locks the buffer by setting DMEMi_CONFIG.EN1 or EN2 to 0_B after a Wraparound or Watermark event. It is up to the software to unlock the buffer again by setting the DMEMi_CONFIG.EN1 or EN2 to 1_B

CAN I-PDUs to system memory with multiplexing enabled

If DMEMi_MODE.TYP is set to 1_B (MUXEN), the CAN I-PDUs in the data buffers are assembled with no padding between them. The SRI transactions are broken down based on the payload length which is part of the PDU header in order to avoid padding, as shown in the alignment of CAN message 1 - 3 in the figure below.

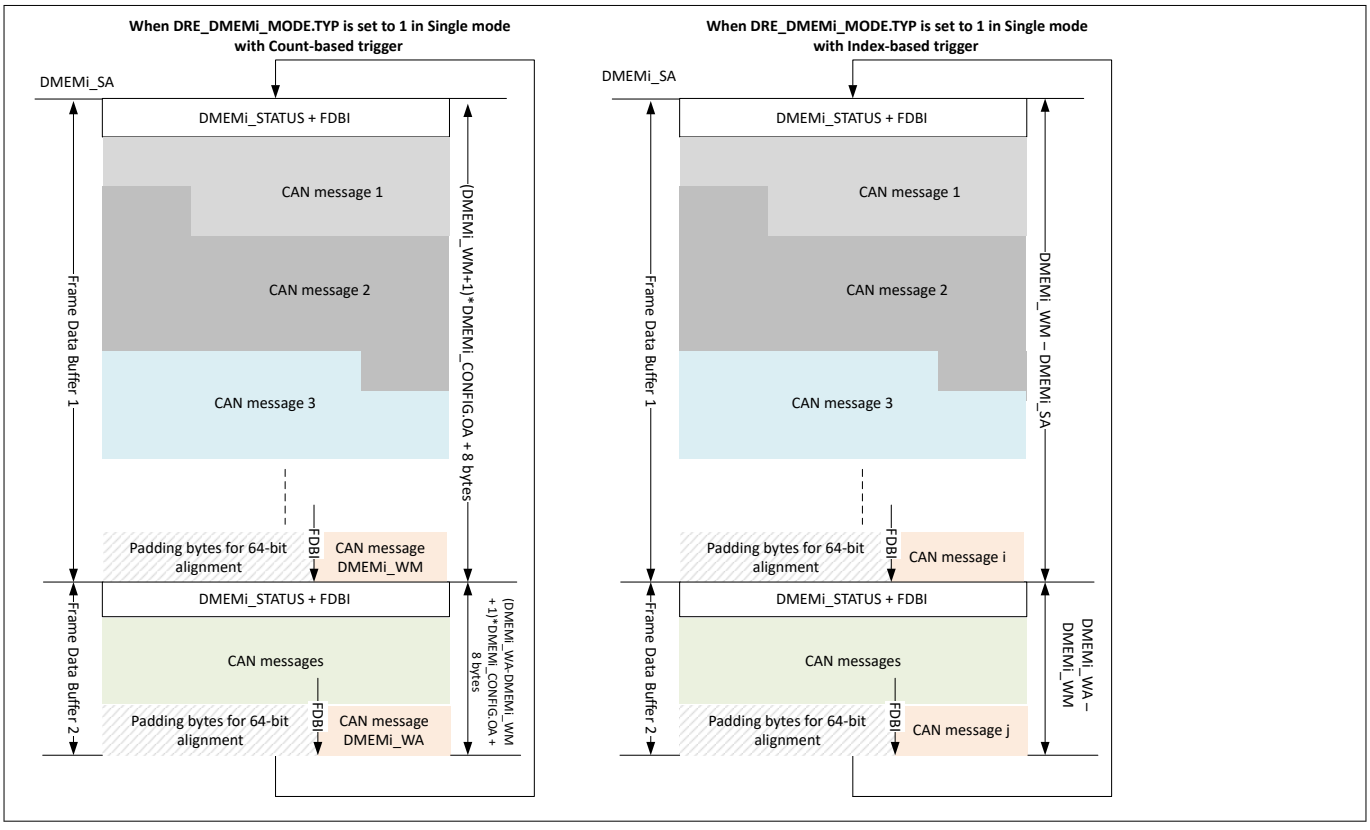


Figure 357 Trigger modes comparison

The DMEMi_FDBI denotes where the next CAN I-PDU is to be stored in the Frame Data Buffer.

When the first CAN I-PDU is received, it is copied at the index DMEMi_SA as long as the WML is large enough to accommodate a whole CAN I-PDU including the status and header information. The FDBI is then updated as follows:

$$FDBI(bytes) = 4\,bytes\,DMEMi_STATUS\,(optional) + 4\,bytes\,DMEMi_FDBI + 8\,bytes\,THEAD\,(optional) + Short\ \ or\ \ Long\,PDU\ \ header\,(4\,or\,8\,bytes) + Data\ \ (bytes)$$

The second CAN I-PDU is then copied starting from the index FDBI as shown in the figure below.

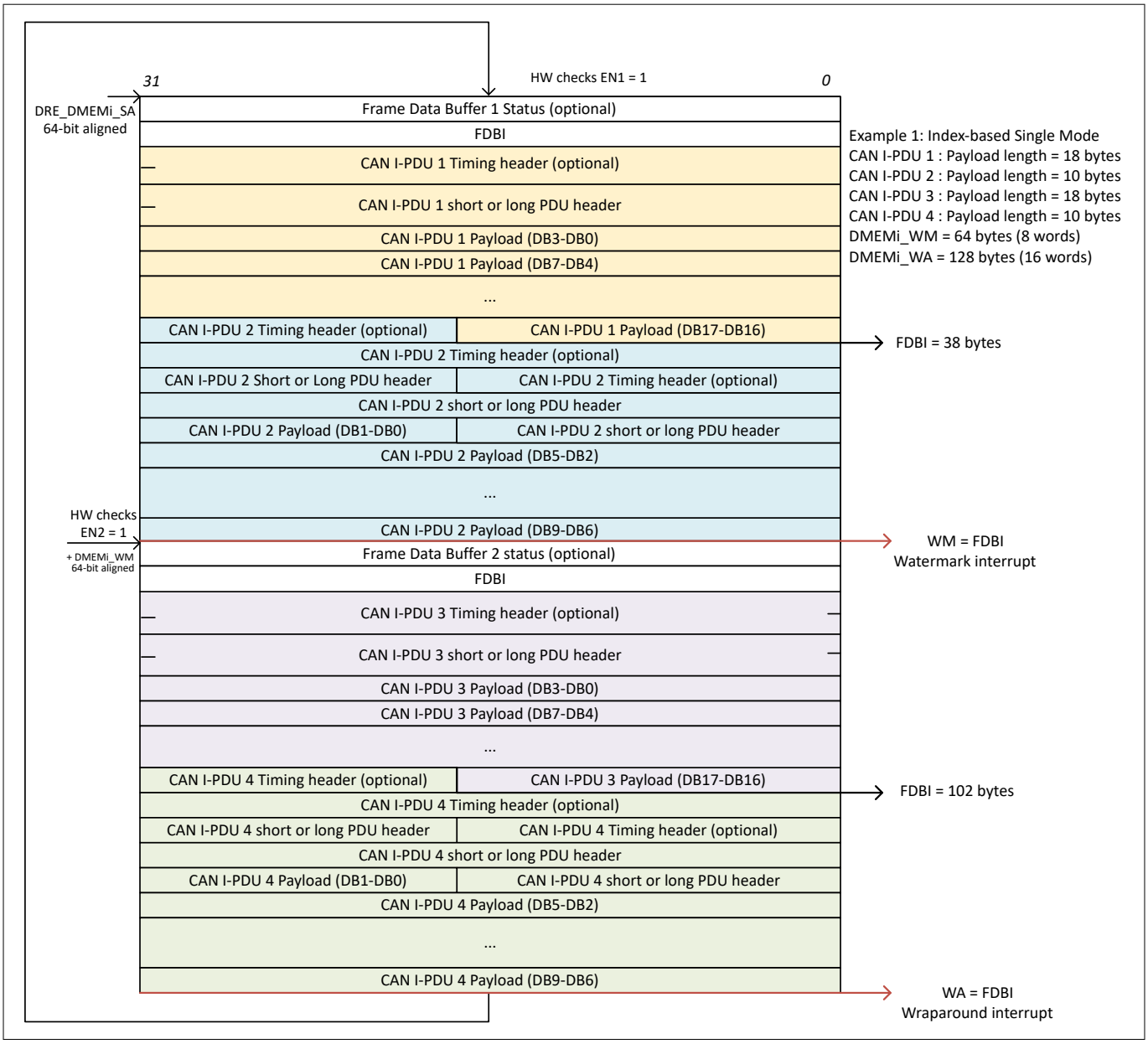


Figure 358 Example 1: Index-based trigger in Single mode

The following occurs if the new CAN I-PDU (including Timing header, if enabled) does not fit to the destination buffer size or if the buffer becomes unavailable:

- If DMEMi_MODE.FOM is set to 0_B, the CAN I-PDU is stored in the next available Frame Data Buffer 1, or Frame Data Buffer 2 when the corresponding EN bit is set to 1_B. When the EN bit is set to 0_B, this means that the Frame Data Buffer is disabled or that the software has not read the data yet. The CAN I-PDU is discarded, the Destination Buffer Overflow Error flag (ME_ERR.DBOE) is set and the source CAN request is cleared
- If DMEMi_MODE.FOM is set to 1_B, the CAN I-PDU is discarded, the Destination Buffer Overflow Error flag (ME_ERR.DBOE) is set and the source CAN request is cleared

  The ongoing transaction has to be canceled by the DRE in the case of error by writing into CRE_ABORT_SEQ at the CAD_CANi_CRESA.ADR + 18_H address. The abort sequence is triggered by the DRE only after it starts reading the CAN frame and before the last read or write operation is launched. The DRE cancels the ongoing read sequence by writing 1_B to CRE_ABORT_SEQ.CRHBUF0 or CRHBUF1 if it has already started reading the CAN frame (R0, R1, DBm) after the error has occurred

For each CAN request, the DRE reads the Routing header RHBUFk_UCRH from the Rx Host Buffer to determine the destination DMEMi and performs the following tasks:

1. Checks if the bit DMEMi_CONFIG.EN1 is 1_B
2. If DMEMi_CONFIG.ATH = 1 is set, then the Timing header (THEAD of two 32-bit words in size) of the CAN I-PDU is copied after DMEMi_FDBI
3. The CAN I-PDU is transferred to the destination memory starting from the PDU header (always two 32-bit words in size) until end of CAN data bytes defined by DLC in the PDU header
4. DMEMi_FDBI is updated with the index where the next CAN I-PDU is to be stored. DMEMi_STATUS.BC is incremented by 1. The BC value is still zero within the DMEMi after the first message is transferred. The register DMEMi_STATUS indicates the correct count value
5. The next CAN I-PDU is fetched. The following occurs depending on the configured layout mode:
   a. **Single mode**: Steps 4 to 6 are repeated until the Watermark level is reached. The DRE proceeds with Step 9 for the Frame Data Buffer 1. The DRE repeats Steps 1 to 6 (if DMEMi_CONFIG.EN2 is 1_B) until the Wraparound level is reached for the Frame Data Buffer 2. See the previous figure Example 1: Index-based trigger in Single mode
   b. **Continuous mode**: Steps 5 to 7 are repeated until the Wraparound level is reached, as shown in the figure below:

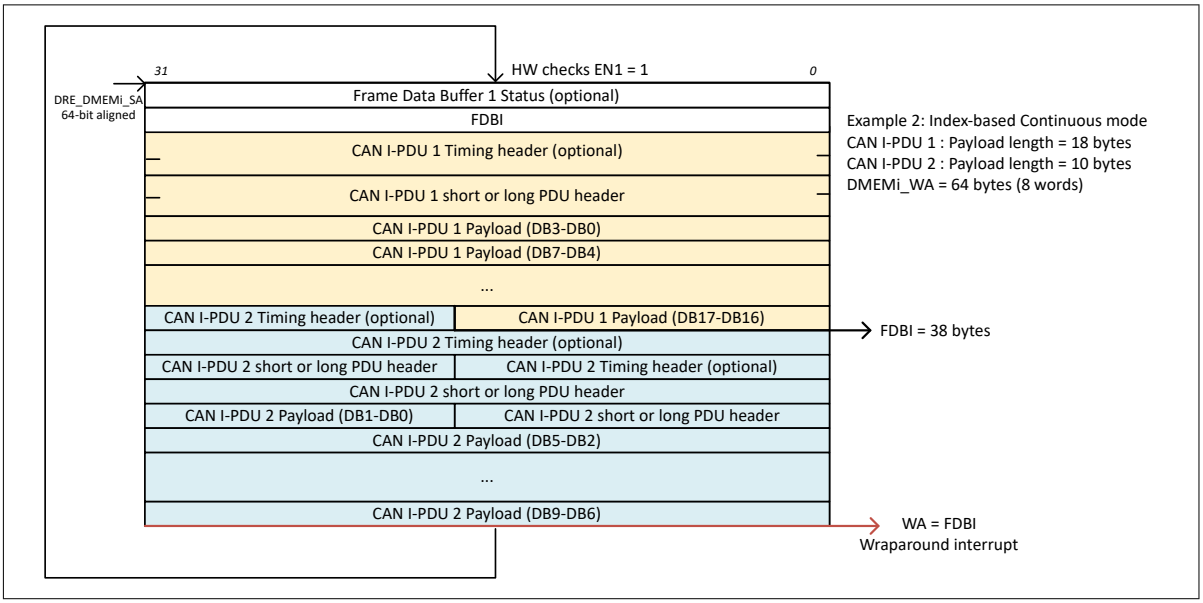


Figure 359 Example 2: Index-based trigger in Continuous mode

6. The DRE then locks the buffer by setting DMEMi_CONFIG.EN1 or EN2 to 0_B. DMEMi_STATUS.WMF or DMEMi_STATUS.WAF is set and interrupt is triggered. The message counter (DMEMi_STATUS .MC) is increased by 1 to indicate a message transfer to the DMEM. Upon overflow of the counter, the counter is reset to 0
7. If DMEMi_CONFIG.AST = 1, then DMEMi_STATUS is copied as the first word of the destination memory just before the Watermark or Wraparound interrupt is triggered
8. DMEMi_FDBI is copied as the 2nd word (or first word) just before the Watermark or Wraparound interrupt is triggered
9. The software initiates the frame processing in the DMEMi until the configured Watermark or Wraparound level is reached. FDBI indicates the exact length of the frame in the buffer. It is up to the software to unlock the buffer again by setting the DMEMi_CONFIG.EN1 or EN2 to 1_B

***Note**: The start addresses of Frame Data Buffer 1 and Frame Data Buffer 2 are always 64-bit aligned. If the configured DMEMi_WM is not 64-bit aligned, there are padding bits added to the end of Frame Data Buffer 1. Even in Count-based trigger mode, when FDBI is not 64-bit aligned at the watermark, there are padding bits added to the end of Frame Data Buffer 1. Nevertheless, the exact length of the frame is indicated by the FDBI*

Handling buffer overflow and unavailability for CAN I-PDUs

In Continuous mode (DMEMi_MODE.BUF = 1_B), the CAN I-PDU is discarded when DMEMi_CONFIG.EN1 is not set, the CAN request is cleared and the DBOE interrupt is triggered. In Single mode (DMEMi_MODE.BUF = 0_B), the DRE handles CAN I-PDUs depending on the buffer availability DMEMi_CONFIG.EN1 and DMEMi_CONFIG.EN2, and the set buffer overflow mode (DMEMi_MODE).

The following table shows how the DRE handles CAN requests in Single mode when the EN bits are not set or when a buffer overflow scenario occurs where the size of the incoming CAN I-PDU exceeds the configured Watermark (DMEMi_WM) or Wraparound (DMEMi_WA) level.

Table 945 Buffer overflow or buffer unavailability (Single mode)

Scenario	Condition	EN1	EN2	FOM	Result
Buffer available	PDU for Frame Data Buffer 1	1	-	-	Copy PDU to Frame Data Buffer 1
	PDU for Frame Data Buffer 2	-	1	-	Copy PDU to Frame Data Buffer 2
Buffer unavailable	PDU for Frame Data Buffer 1 or 2	0	0	1	PDU is discarded, CAN request cleared and DBOE interrupt triggered
	PDU for Frame Data Buffer 1	0 (buffer full)	1	0	Copy PDU to Frame Data Buffer 2
	PDU for Frame Data Buffer 2	1	0 (buffer full)	0	Copy PDU to Frame Data Buffer 1
Buffer overflow	Size of PDU for Frame Data Buffer 1 exceeds DMEMi_WM	1	1	0	Copy PDU to Frame Data Buffer 2 and trigger Watermark interrupt
		1	0	0	PDU is discarded, CAN request cleared and DBOE interrupt triggered
		1	-	1	
	Size of PDU for Frame Data Buffer 2 exceeds DMEMi_WA	1	1	0	Copy PDU to Frame Data Buffer 1 and trigger Wraparound interrupt
		0	1	0	PDU is discarded, CAN request cleared and DBOE interrupt triggered
		-	1	1	

***Note**: The DMEMi_STATUS that is copied to the DMEMi (if DMEMi_CONFIG.AST = 1) is only updated in the buffer that receives a message. In case the FOM bit is set in order to copy the message to the spare Frame data buffer, the DMEMi_STATUS is not updated for the buffer which is full but is updated for the Frame data buffer to which the CAN message is copied to. In case of buffer overflow, the software shall read the error information from the DMEMi_STATUS register and not from the DMEMi*

Summary of allowed scenarios during CAN message transfer to system memory

Scenarios that lead to a message being stored in the configured DMEM. The following table shows a summary of allowed scenarios for CAN frame or CAN I-PDU transfer to the system memory.

Table 946 Summary of allowed CAN node to memory cases

Configured mode in DMEMi_MODE (TYP_TRIG_BUF)[1]	Operating mode	Condition[2]	EN 1	EN 2	Result
-	-	CTYP matches	-	-	Message stored, WMF and WAF flags set accordingly, FDBI updated, EN1 cleared if WML reached
MUXEN_COUNT_SINGLE	MUXEN_COUNT_SINGLE	Message destined for FDB1	1	x	Message stored, WMF and WAF flags set accordingly, FDBI updated, EN1 cleared if WML reached
		Message destined for FDB2	x	1	Same handling as FDB1
MUXEN_COUNT_CONT MUXEN_COUNT_SINGLE with WM>=WA	MUXEN_COUNT_CONT	Message destined for DMEM	1	x	Message stored, WMF and WAF flags set accordingly, FDBI updated, EN1 cleared if WAL reached[3]
MUXEN_INDEX_SINGLE	MUXEN_INDEX_SINGLE	Message destined for FDB1	1	x	Message accepted in FDB1, WMF flag set if WML reached, FDBI set to end of messaged(+1) delivered in FDB1, EN1 cleared if at WML
		Message destined for FDB2	x	1	Same handling as FDB1
MUXEN_INDEX_CONT MUXEN_INDEX_SINGLE with WM>=WA	MUXEN_INDEX_CONT	Message destined for DMEM	1	x	Message stored and flags set accordingly, FDBI updated
MUXDIS_*	MUXDIS_COUNT_CONT	Message destined for DMEM	1	x	Message stored[4] , WMF and WAF flags set accordingly, EN1 cleared if wrap around reached, EN2 untouched

1) Configuration should be static
2) The requirement is that the WML/WAL set to at least one maximum frame size plus some bytes
3) WMF always set according to the programmed levels, independent of the mode, but no WM interrupt triggered
4) General rule is that a message coming to a buffer whose EN is 0 is dropped, BO is triggered, and status is unchanged

***Note**: Other scenarios not included in this table are illegal or cause the message to be dropped and trigger the corresponding interrupt or error.*

20.6.3.2.4 Move engine

The Move Engine performs the FPI and SRI transactions as required by the routing transfer operation. The Move Engine also has direct read and write access to the DRE internal RAM. The read/write operation is directed to the corresponding interface based on the address map.

The current status of the routing transfer operation and the bus access is shown in ME_STATE SFR. The read and write address of a ME transaction is shown in ME_SRCA and ME_DESTA. In case of an FPI or SRI transaction error, the corresponding error flags are set in ME_ERR. After an error, the first error status is shown in ME_ERR.FESID, FEDID, ME_FESRCA and ME_FEDESTA. In case of an error, the current routing transfer is aborted and the Move Engine continues to perform the next routing transfer request from the Routing Control Unit. The DRE writes into the CRE_ABORT_SEQ at the CAD_CANi_CRESA.ADR + 18_H address to indicate the CRE about the abort of the routing transfer.

Resource partition configuration

CAN nodes and system memory destinations can be assigned to one of the Resource Partitions (RP) using the registers CANi_RP, DMEMi_RP and DMAi_RP. These registers point to the RP configuration MODEr that defines the DRE Move Engine access condition to the on-chip buses when performing a routing transfer. The Move Engine inherits the RP configuration of CAN nodes during read/write access to CAN nodes. Similarly, it inherits the RP configuration of system memory destinations, during write access to the same.

While accessing the Ethernet DMA channels, the DRE inherits the RP configuration pointed to by the DMAi_RP register. DMA0_RP and DMA1_RP indicates the index of RP configuration of GETH0 and GETH1 respectively. DMA2_RP to DMA5_RP indicates the index of RP configuration of LETH0 to LETH3 respectively.

The RP has a master tag identifier driven onto the on-chip bus during the Move Engine access to the corresponding resource. The master tag identifier for a RP is derived from the DRE base master tag ID and corresponding MODEr.TAG_OFF bit-field.

Each RP may further be assigned to a virtual machine number and protection set (PRS). The virtual machine number and the PRS programmed in the RP MODEr SFR will be driven on the bus during read or write access to the corresponding resource.

MODEr.MODE determines if the DRE performs the corresponding bus accesses in supervisor mode or user mode.

20.7 Ethernet descriptor handler

The ETH Descriptor Handler prepares the normal Transmit and Receive ETH descriptors within the Message RAM to exchange ETH frames and descriptors that contain control or status information between the DRE and ETH MAC.

20.7.1 Feature list

- Offloads CPU by preparing the Rx and Tx descriptors
- A maximum of 24 Rx descriptors can be configured (4 per Ethernet Rx DMA channel) to receive an Ethernet frame in the EIBUF. One Rx descriptor is used to receive one Ethernet frame in EIBUF
- A total of 24 Tx descriptors can be configured (4 per Ethernet Tx DMA channel) and used to transmit or forward an Ethernet frame:
  - Transmit an IEEE 1722 ACF Ethernet frame from EOBUF or
  - Forward an Ethernet frame received in EIBUF
- Consists of a Forwarding Engine to forward Ethernet frames between Ethernet interfaces

20.7.2 Functional overview

The block diagram of the Ethernet Descriptor Handler is shown in the following figure. It contains three blocks:

- Tx Descriptor Handler
- Rx Descriptor Handler
- Forwarding Engine

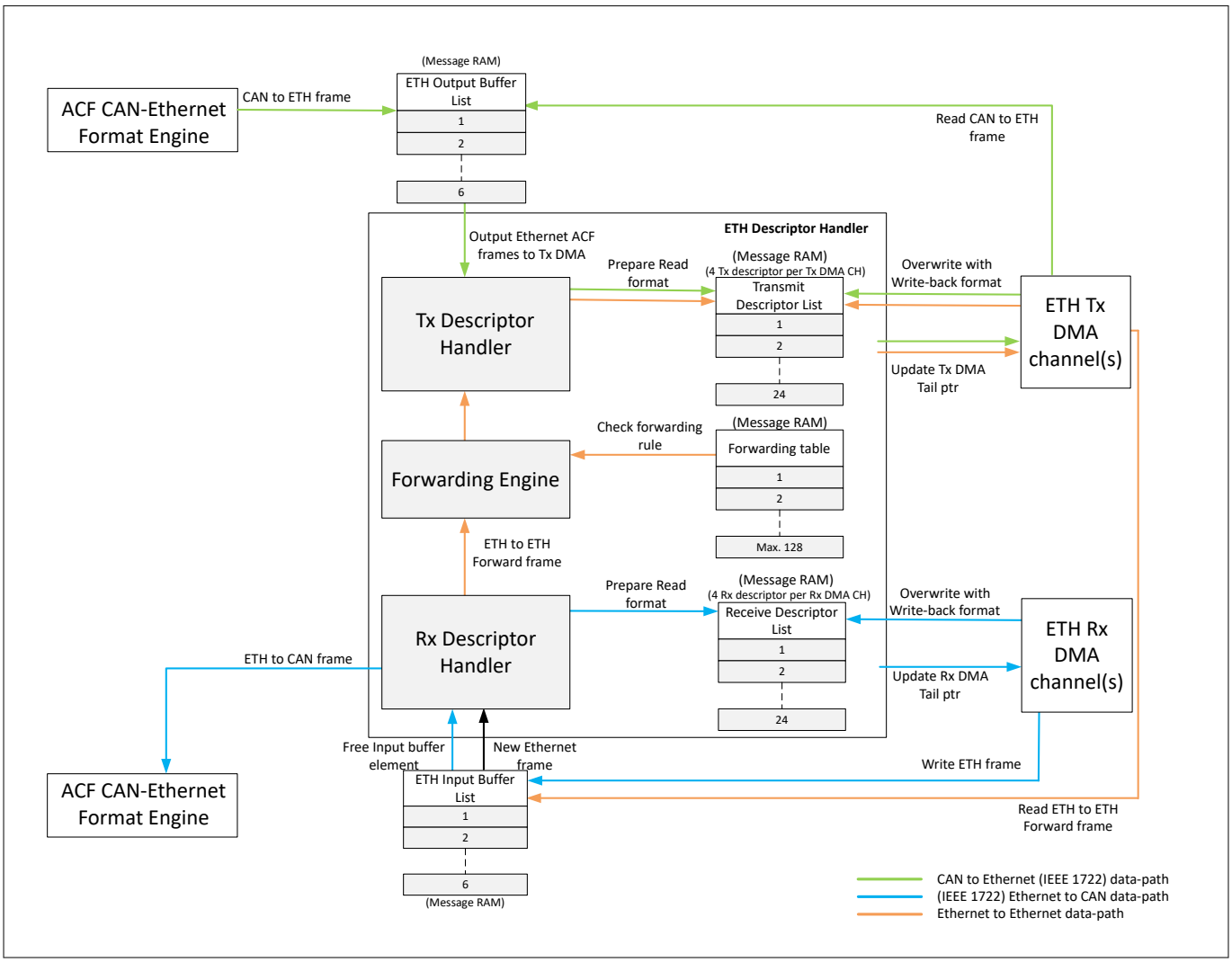


Figure 360 Ethernet descriptor handler block diagram

Tx Descriptor Handler

The Tx Descriptor Handler prepares the Tx descriptor(s) in Read format within the Message RAM for each Ethernet frame that is to be routed or forwarded to an Ethernet interface through an Ethernet Tx DMA channel.

- CAN to Ethernet frame: IEEE 1722 ACF Ethernet frame in EOBUF
- Ethernet to Ethernet frame: Ethernet forward frame in EIBUF

It also updates the DMA Tail Pointer registers of LETH (DMA_CHy_TxDesc_Tail_Pointer (y=0-7)) and GETH (DMA_CHj_TxDesc_Tail_LPointer (j=0-7)) Tx DMA channels. For the LETH Tx address register, see RAM LETH Tx DMA channel address. For the GETH Tx address register, see RAM GETH Tx DMA channel address . The DMA channel overwrites the descriptor with the Write-back format after reading the Ethernet frame from the buffer. The Tx Descriptor Handler also checks the status within the Write-back format of the Tx descriptor and triggers an interrupt in the event of any errors.

Rx Descriptor Handler

The Rx Descriptor Handler prepares the Rx descriptor(s) in Read format within the Message RAM whenever there is a free EIBUF element to receive an Ethernet frame from an Ethernet Rx DMA channel.

It also updates the DMA Tail Pointer registers of the LETH (DMA_CHy_RxDesc_Tail_Pointer (y=0-7)) and GETH (DMA_CHj_RxDesc_Tail_LPointer (j=0-7)) Rx DMA channels. For the LETH Rx address register, see RAM LETH Rx DMA channel address. For the GETH Rx address register, see RAM GETH Rx DMA channel address. The DMA channel overwrites the descriptor with the Write-back format after writing the Ethernet frame to the buffer. The Rx Descriptor Handler also checks the status within the Write-back format of the Rx descriptor and triggers an interrupt in the event of any errors.

The Rx Descriptor Handler also separates the Ethernet to CAN traffic and the Ethernet to Ethernet forward traffic.

Forwarding Engine

The Forwarding Engine generates the FTCFG.FID from the Write-back format of the Rx descriptor. It performs the Forward ID filtering to determine the forwarding destination of the Ethernet frame.

20.7.3 Functional description

In this section, the transmit, receive and forward functions of the Ethernet frames are described in detail.

20.7.3.1 Tx descriptor handler

The Tx Descriptor Handler transmits ACF Ethernet frames from the EOBUF and also forwards an Ethernet frame from the EIBUF to an Ethernet Tx DMA channel.

Whenever there is an Ethernet frame to be transmitted or forwarded, the corresponding EREQ.TXi_REQ or EREQ.FWDi_REQ request bit is set.

When TETHDLi_CTRL.TRIG is set to 1, it sets EOBUFj_STATUS.TXREQ and triggers an Ethernet output frame transmit request interrupt to the IR indicating that the software can read the Ethernet frame from the buffer. When TETHDLi_CTRL.TRIG is set to 0, the following happens:

1. The Tx Descriptor Handler prepares the 4 words of all Tx descriptors in Read format:
   a. For the ACF Ethernet frame in the EOBUF
   b. For the Ethernet frame in the EIBUF in the case of Ethernet frame forwarding
2. The Tx Descriptor Handler sets the following bits:
   a. TDESCi_RD3.FD bit is set to indicate to the DMA that this is the first Tx descriptor of the Ethernet frame
   b. TDESCi_RD3.LD bit is set to indicate to the DMA that this is the last Tx descriptor of the Ethernet frame

   The Tx Descriptor Handler prepares the remaining unused Tx descriptors as well with buffer pointers to EOBUF. If a forwarding request arrives, then the Tx descriptors are used for the frame in EIBUF by changing the buffer pointers. The OWN bit is not set for these descriptors as unused Tx descriptors are still owned by the DRE.
3. After preparing the Tx descriptor, the Tx Descriptor Handler sets the TDESCi_RD3.OWN bit in the descriptor, indicating that the DMA channel TETHDLi_CTRL.DMACH owns the descriptor. The OWN bit of the second Tx descriptor is set for the second frame, the OWN bit of the third descriptor is set for the third frame, and so on. After the fourth frame, the first descriptor is used again
4. The Tx Descriptor Handler then reads the address of the DMA Tail Pointer from the Ethernet address database and updates the Tail Pointer DMA_CHj_TxDesc_Tail_LPointer or DMA_CHy_TxDesc_Tail_Pointer of the Ethernet DMA channel TETHDLi_CTRL.DMACH, indicating the total number of Tx descriptors. The Tx Descriptor Tail LPointer register has the 32-bit end address of the Transmit descriptor list. This is the Transmit Poll Demand from the DRE.
   a. $DMA_CHj_TxDesc_Tail_LPointer = Start\ address\ of\ TETHDLi + 10_H$ for GETH if the first descriptor has OWN bit set
   b. $DMA_CHy_TxDesc_Tail_Pointer = Start\ address\ of\ TETHDLi + 10_H$ for LETH if the first descriptor has OWN bit set

   The Tail Pointer is updated until the descriptor which has the OWN bit set
5. While in Run state, the Ethernet DMA selects the next Tx DMA channel from which the packets requiring transmission should be processed. The Ethernet DMA channel fetches each Tx descriptor (DRE_TDESC) and performs the following actions:
   a. Reads the buffer address and payload length from the Tx descriptor and fetches the ACF Ethernet frame from the EOBUF. In case of forwarding it fetches the Ethernet frame from the EIBUF
   b. Clears OWN bit and overwrites the Tx descriptors with the Write-back format. The TDESCi_WR3G.LD bit or TDESCi_WR3L.LD must be set by DMA once the last descriptor is processed instructing the Tx Descriptor Handler to release the DRE_TDESC to be used by another frame

c. It then fetches the next descriptor and if the OWN bit is not set, it goes into Suspend state

d. The DMA comes out of Suspend state and goes into Run state with the next Transmit Poll Demand from the DRE

***Note**: The DRE issues only one Transmit poll demand at a time. Any new request for the same Tx DMA channel would be considered only after the write-back format is written by the DMA channel for the previous request*

6. If there is no error in the Write-back format, the Tx Descriptor Handler clears the following bits, indicating buffer free:

- EREQ.TXi_REQ, after the entire frame is read from the EOBUF
- EREQ.FWDi_REQ, after the entire frame is read from the EIBUF

7. When there is a new frame to be transmitted or forwarded, the Tx Descriptor Handler then issues another Transmit Poll Demand after setting the OWN bit of the next Tx descriptor pointed to by the TETHDLi_CTRL.PTR. The PTR is always pointing to the next descriptor following the last word write-back format write operation from the Tx DMA channel. This Transmit Poll Demand brings the DMA out of Suspend state. If the DMA does not read the descriptor pointed to by TETHDLi_CTRL.PTR, error flag EOBUFj_ERROR.TDESE is set and Transmit descriptor error interrupt INT_14 is triggered

8. After the successful routing and forwarding of each Ethernet frame, EDLSTAT.TXCNT is incremented

9. The transmit of an Ethernet frame is complete upon the detection of the write of the last word of the descriptor Write-back format TDESCi_WR3 (RAM TDESC word 3 Write-back format for GETH or RAM TDESC word 3 Write-back format for LETH). The Tx DMA updates the Write-back descriptor error status TDESCi_WR3G.DERR in most error scenarios. In the case of LETH, TDESCi_WR3L.DE and ES bits indicate error. But when an AXI bus error occurs (refer GETH or LETH chapters) while writing the Write-back format, the transaction is aborted immediately by the Tx DMA channel and the last word TDESCi_WR3 might not be updated. When Ethernet Transmit Timeout Monitoring is enabled and the last word is not written before the timer expires, a timeout interrupt is triggered. When Ethernet Transmit Timeout Monitoring is disabled, the Tx Descriptor Handler polls the GETH DMA_CHj_Status (j=0-7) and LETH DMA_CHy_Status (y=0-7) registers every 200 clock cycles in order to monitor the status of the DMA channel. The polling starts at the first word read of the descriptor Read format and is done in a round-robin fashion for each DMA channel. If the DMA detects one of the following conditions, the transmission from that channel is suspended or stopped and the Tx DMA proceeds to go into Suspend state or Stop state.

Tip: As Suspend state is considered an abnormal event by Ethernet, it triggers AIS interrupt. It is recommended that this interrupt is disabled when the DMA channel is configured to route or forward frames using DRE

Bit TBU (bit 2) of the Status register of the corresponding DMA channel is set in the event that transmission is suspended:

a. The descriptor is flagged as owned by the DRE (OWN bit is cleared): In this case the DRE Tx Descriptor Handler gives the ownership to the DMA by setting the OWN bit and then issues a Poll Demand command by writing to the Tail Pointer register

b. The descriptor Tail Pointer is equal to the current descriptor pointer in Ring Descriptor list mode: In this case the DRE Tx Descriptor Handler modifies the Tail Pointer such that the following condition is true:

Current Descriptor Pointer < Descriptor Tail Pointer

c. An error condition occurs because of transmit underflow: The packet transmission is aborted and TDESCi_WR3.DERR (in the case of GETH) or TDESCi_WR3.DE (in the case of LETH) is set accordingly by the DMA channel. DRE Tx Descriptor Handler triggers a Transmit descriptor error interrupt INT_14

d. In the event of a bus error, the DMA proceeds to Stop state, bit FBE is set and the bit-field TPS of the Status register is set accordingly. In this case, the packet transmission is aborted, TDESE is set and Transmit descriptor error interrupt INT_14 is triggered

e. The software can reset the TETHDLi_CTRL.PTR if required by setting the TETHDLi_CTRL.STOP bit. The software must wait for the DRE to set the TETHDLi_CTRL.STOPACK before resetting the PTR. The software must then clear the STOP bit. Clearing of this STOP bit will trigger DRE to clear STOPACK and set OWN bit of the TDESCi pointed to by the new updated TETHDLi_CTRL.PTR and trigger a Transmit Poll Demand to the Tx DMA channel

***Note**: The software can ensure that the Ethernet DMA current pointer in this case is also pointing to the same descriptor by resetting the Ethernet DMA current pointer or by restarting the DMA*

Requests arbitration

All the requests to forward or transmit Ethernet frames are processed in a round-robin fashion. A requests arbitration example is shown in the image below. The priority is as follows:

1. Write-back format processing
2. If the previous processed request was TXi_REQ, then FWDi_REQ is processed
3. If the previous processed request was FWDi_REQ, then TXi_REQ is processed

   This ensures a non-starving arbitration of TXi_REQ and FWDi_REQ

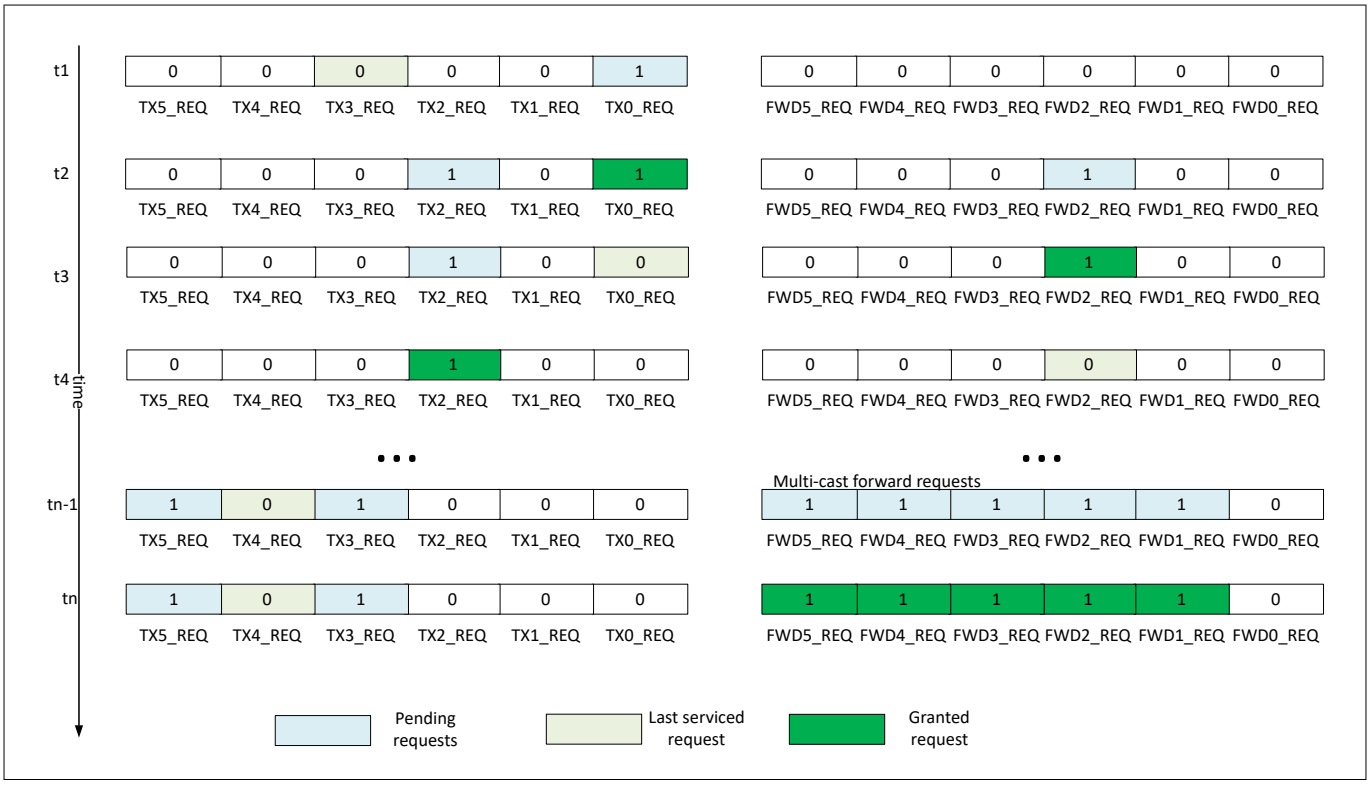


Figure 361 Requests arbitration example

Ethernet transmit timeout monitoring

The DRE monitors for delayed Ethernet frame availability to transmit within the EOBUF and also monitors for delayed read of Ethernet frames from EOBUF by the Ethernet DMA channel. The Ethernet watchdog counter starts incrementing when the Ethernet watchdog timeout is enabled in EWDCFG.EN. The watchdog timer triggers periodic events E1, E2, …, En after a user-configured Ethernet timeout period EWDCFG.ETO. The Ethernet timeout period is configured by the user based on the use-case. If the start condition is true and the

end condition is not true within two events E_i and E_{i+1}, the Ethernet watchdog timeout error flag EOBUFj_ERROR.WDTE is set and error interrupt INT_14 is triggered if EWDCFG.WTOE is enabled. The software can read the frame in the event of such a timeout and must clear the error flag. Once the error flag is cleared, the EOBUF is used for the next frame.

The timeout period between the events E_i is calculated as follows:

- Default timeout period: $T_{def} = \frac{16}{f_{SRI}}$
- User-configured timeout period: $T_{out} = T_{def} * (EWDCFG.ETO + 1)$

Table 947 Timeout interrupt trigger

Timeout check	Timeout interrupt trigger
End condition is true within a single event E_i	No Timeout error. No timeout interrupt triggered
End condition is true within two consecutive events E_i and E_{i+1}	If the presence of start condition is detected at Ei, then the end condition is expected to happen before Ei+1. If not, timeout interrupt is triggered
End condition is true after two consecutive events E_i and E_{i+1}	Timeout interrupt is triggered

Table 948 EOBUF : Start and end condition list

Start condition	End condition
EOBUF full (ready for transmit). EREQ.TXi_REQ is set to indicate this	EOBUF empty (read complete by Ethernet DMA channel or software). EREQ.TXi_REQ is cleared
EOBUF not empty (partially full)	EOBUF full (ready for transmit). EREQ.TXi_REQ is set

Related information

Ethernet output buffer on page 3730
Ethernet input buffer on page 3733
Ethernet descriptor lists on page 3733
Descriptor structure configuration on page 3733
Descriptor error handling on page 3789
Tx descriptors on page 3734
Forwarding engine on page 3787
Ethernet address database (EAD) on page 3740

20.7.3.2 Rx descriptor handler

The Rx Descriptor Handler receives an Ethernet frame in the EIBUF from an ETH Rx DMA channel.

Whenever there is a free EIBUF available to receive an ETH frame, the EIBUFi_STATUS.BPR is 0 indicating buffer empty.

When RETHDLi_CTRL.TRIG is set to 1, the Descriptor Handler sets EIBUFi_STATUS.RXREQ and triggers an Ethernet Frame Receive Request Interrupt INT_11 to the IR indicating that the software can write an Ethernet frame to the free buffer. The software writes the Ethernet frame to the EIBUF and clears the RXREQ. The handler sets the Buffer Pending Request (BPR) bit after the Ethernet frame is received in the EIBUF. When RETHDLi_CTRL.TRIG is set to 0, the following actions are performed:

1. The Rx Descriptor Handler prepares the 4 words of all Rx descriptors in Read format for the ETH frame to be stored in EIBUF but only sets the OWN bit of the first Rx descriptor. The OWN bit of the second Rx descriptor is set for the second frame, the OWN bit of the third descriptor is set for the third frame, and

so on. After the fourth frame, the first descriptor is used again. RDESC_RD0.BUF1AP points to the EIBUF start address

2. After preparing the Rx descriptors, the Rx Descriptor Handler sets the RDESCi_RD3.OWN bit (GETH) or RDESCi_RD3.OWN bit (LETH) in the first descriptor, indicating that the DMA channel RETHDLi_CTRL.DMACH owns the descriptor
3. The Rx Descriptor Handler reads the address of the DMA Tail Pointer from the Ethernet address database and updates the Tail Pointer DMA_CHj_RxDesc_Tail_LPointer (GETH) or DMA_CHy_RxDesc_Tail_Pointer (LETH) of the Ethernet DMA channel RETHDLi_CTRL.DMACH indicating the total number of Rx descriptors. The Rx Descriptor Tail LPointer register has the 32-bit end address of the Receive descriptor list. This is the Receive Poll Demand from the DRE:
   a. $DMA_CHj_RxDesc_Tail_LPointer = Start\ address\ of RETHDLi + 10_H$ for GETH if the first descriptor has OWN bit set
   b. $DMA_CHy_RxDesc_Tail_Pointer = Start\ address\ of RETHDLi + 10_H$ for LETH if the first descriptor has OWN bit set

   The Tail Pointer is updated up to the descriptor which has the OWN bit set
4. While in Run state, the Ethernet DMA channel fetches each Rx descriptor (DRE_RDESC) and performs the following actions:
   a. Reads the buffer address from the Rx descriptor and writes the Ethernet frame to this address in the EIBUF
   b. Clears OWN bit and overwrites the Rx descriptor with the Write-back format which includes the packet length RDESCi_WR3G.PL (GETH) or RDESCi_WR3L.PL (LETH). The RDESCi_WR3G.LD bit (GETH) or RDESCi_WR3L.LD (LETH) must be set by the DMA once the last descriptor is processed, indicating to the DRE to release the DRE_RDESCi to be used by another frame. If the LD bit is not set, the error flag EIBUFi_ERROR.RDESE is set and error interrupt INT_10 is triggered
   c. The Ethernet DMA starts to process the next Rx descriptor but goes into Suspend state as the second descriptor is still owned by the DRE
   d. The DMA comes out of Suspend state and goes into Run state with the next Receive Poll Demand from the DRE
5. If there is no error in the Write-back format, after the last word write to the Write-back format Rx descriptor DRE_RDESCi_WR3 (RDESCi_WR3G for GETH or RDESCi_WR3L for LETH), the OWN bit is checked if cleared and BPR bit is set to 1 by the Rx Descriptor Handler indicating buffer full
6. The DRE then begins processing the new frame in EIBUF. After the frame in EIBUF is successfully processed and once the buffer is empty, the Rx Descriptor Handler sets the OWN bit of the next Rx descriptor pointed to by the RETHDLi_CTRL.PTR and issues a Receive Poll Demand . The PTR is always pointing to the next descriptor following the last word Write-back format write operation from the Rx DMA channel. This Receive Poll Demand brings the DMA out of Suspend state to start processing the next descriptor. If the DMA does not read the descriptor pointed to by RETHDLi_CTRL.PTR, error flag EIBUFi_ERROR.RDESE is set and Receive descriptor error interrupt INT_10 is triggered
7. After successful forwarding of each Ethernet frame, EDLSTAT.RXCNT is incremented
8. The reception of an Ethernet frame is complete upon the detection of the write of the last word of the descriptor Write-back format RDESCi_WR3 (RAM RDESC word 3 Write-back format GETH or RAM RDESC word 3 Write-back format LETH). The Rx DMA updates the Write-back descriptor error summary RDESCi_WR3G.ES bit or RDESCi_WR3L.ES bit in most error scenarios. But when an AXI bus error (refer LETH or GETH chapters) occurs while writing the Write-back format, the transaction is aborted immediately by the Rx DMA channel and the last word RDESCi_WR3 might not be updated. When Ethernet Receive Timeout Monitoring is enabled and the last word is not written before the timer expires, a timeout interrupt will be triggered. When Ethernet Receive Timeout Monitoring is disabled, the Rx Descriptor Handler polls the GETH DMA_CHj_Status (j=0-7) and LETH DMA_CHy_Status (y=0-7) registers every 200 clock cycles in order to monitor the status of the DMA channel. The polling starts at the first word read of descriptor Read format and is done in a round-robin fashion for each DMA channel.

If the DMA detects one of the following conditions, the transmission from that channel is suspended or stopped and the Rx DMA proceeds to go into Suspend state or Stop state

a. The Ethernet DMA enters Suspend state when there are no free descriptors (based on Rx current descriptor and descriptor Tail Pointer register values). Bit RBU (bit 7) of the Status register of the corresponding DMA channel is set in the event that transmission is suspended. It exits the suspend state on a Receive Poll Demand from the DRE after it advances the Tail Pointer of the channel.

Tip: As Suspend state is considered an abnormal event by Ethernet, it triggers AIS interrupt. It is recommended that this interrupt is disabled when the DMA channel is configured to route or forward frames using DRE

b. If there is an error indication by RDESCi_WR3G.ES or RDESCi_WR3L.ES bit, the Rx Descriptor Handler stores the frame in the EIBUF, sets the error flag EIBUFi_ERROR.RDESE and triggers a Receive descriptor error interrupt INT_10

c. In the event of a bus error, the DMA proceeds to Stop state, bit FBE is set and the bit-field RPS of the Status register is set accordingly. In this case, the transaction is aborted, RDESE is set and Receive descriptor error interrupt INT_10 is triggered

d. The software can reset the RETHDLi_CTRL.PTR if required by setting the RETHDLi_CTRL.STOP bit. The software must wait for the DRE to set the RETHDLi_CTRL.STOPACK before resetting the PTR. The software must then clear the STOP bit. Clearing of this STOP bit will trigger DRE to clear STOPACK and set OWN bit of the RDESCi pointed to by the new updated RETHDLi_CTRL.PTR and trigger a Receive Poll Demand to the Rx DMA channel

***Note**: The software can ensure that the Ethernet DMA current pointer in this case is also pointing to the same descriptor by resetting the Ethernet DMA current pointer or by restarting the DMA*

Ethernet receive timeout monitoring

The DRE monitors for a delayed write of an Ethernet frame by the Ethernet DMA channel or software to the EIBUF and also monitors for delayed processing of the Ethernet frame by the ACF Format Engine or by the Tx Descriptor Handler in the case of Ethernet forwarding. The Ethernet watchdog counter starts incrementing when the Ethernet watchdog timeout is enabled in EWDCFG.EN is set. The watchdog timer triggers periodic events E1, E2, …, En after a user-configured Ethernet timeout period EWDCFG.ETO. The Ethernet timeout period is configured by the user based on the use-case. If the start condition is true and the end condition is not true within two events E_i and E_{i+1}, the Ethernet watchdog timeout error flag EIBUFi_ERROR.WDTE is set and error interrupt INT_10 is triggered if EWDCFG.WTOE is enabled. The software can read the frame in the event of such a timeout and must clear the error flag. Once the error flag is cleared, the EIBUF is used for the next frame.

The timeout period between the events E_i is calculated as follows:

- Default timeout period: $T_{def} = \frac{16}{f_{SRI}}$
- User-configured timeout period: $T_{out} = T_{def} * (EWDCFG.ETO + 1)$

Table 949 Timeout interrupt trigger

Timeout check	Timeout interrupt trigger
End condition is true within a single event E_i	No timeout error. No timeout interrupt triggered
End condition is true within two consecutive events E_i and E_{i+1}	If the presence of start condition is detected at Ei, then the end condition is expected to happen before Ei+1. If not, timeout interrupt is triggered

(table continues...)

Table 949 (continued) Timeout interrupt trigger

Timeout check	Timeout interrupt trigger
End condition is true after two consecutive events E_i and E_{i+1}	Timeout interrupt is triggered

Table 950 EIBUF : Start and end condition list

Start condition	End condition
EIBUF empty (EIBUFi_STATUS.BPR is 0)	EIBUF full (EIBUFi_STATUS.BPR is 1). In the event of an error scenario such as overflow, the EIBUFi_STATUS bit BPR, Buffer Pending Request, is not set by the DRE, which triggers a timeout. In this case, two error interrupts (Buffer full and timeout) are triggered. If the end condition and the timeout occur at the same time, INT_10 is triggered even if the Ethernet frame is processed successfully
EIBUF full	EIBUF empty (processed by the Ethernet to CAN Disassembler or by the Tx Descriptor Handler in the case of forwarding)

Related information

Ethernet input buffer on page 3733
Ethernet descriptor lists on page 3733
Descriptor structure configuration on page 3733
Descriptor error handling on page 3789
Rx descriptors on page 3736
Forwarding engine on page 3787
Ethernet address database (EAD) on page 3740

20.7.3.3 Forwarding engine

The Forwarding Engine assists in forwarding Ethernet frames between Ethernet interfaces.

The Rx Descriptor Handler decides the path of the Ethernet frame in the EIBUF. The Rx Descriptor Handler can give control of the Ethernet frame to the ACF CAN-Ethernet Format Engine (to the Ethernet ACF to CAN Disassembler) in the case of ETH to CAN routing or to the Forwarding Engine in the case of Ethernet forwarding. It gives control to the ACF CAN-Ethernet Format Engine under the following scenarios:

1. In the case of an IEEE 1722 Ethernet frame containing AV tagged data,
   a. For GETH, RDESCi_WR2G bit AVTDP or AVTCP is set

      ***Note**: The DRE does not support split header feature. The DRE identifies the frame to be AV tagged when AVTDP or AVTCP is set irrespective of L34T value. If the frame is not AV tagged but AVTDP or AVTCP is still set, the subtype check or failed StreamID filtering described below would give control to the Forwarding Engine*

   b. For LETH, RDESCi_WR1L bits 2:0 (Payload Type) are 110_B or 111_B
2. If the frame is identified to be an IEEE 1722 Ethernet frame (refer to the section Validation of the Ethernet input frame):
   a. Subtype = 0x82
   b. StreamID filtering passed

In all other scenarios, the Rx Descriptor Handler gives control of the Ethernet frame to the Forwarding Engine.

The Forwarding Engine assists in forwarding Ethernet frames between Ethernet interfaces. The Ethernet MAC consists of a built-in Ethernet packet filtering which consists of L2, L3, L4 filters and a Flexible Receive Parser (FRP). Acceptance and rejection of packets is based on the configuration of these filters:

- For LETH, the L3L4FM (3 bits) is not included in filtering
- For GETH, the L3L4FM (3 bits) is not included in filtering when both L34M and L4M are zero

The DRE_RDESCi Write-back format consists of the packet filtering and FRP status information. This is used by the Forwarding Engine to create a FTCFG.FID as shown in the following figure.

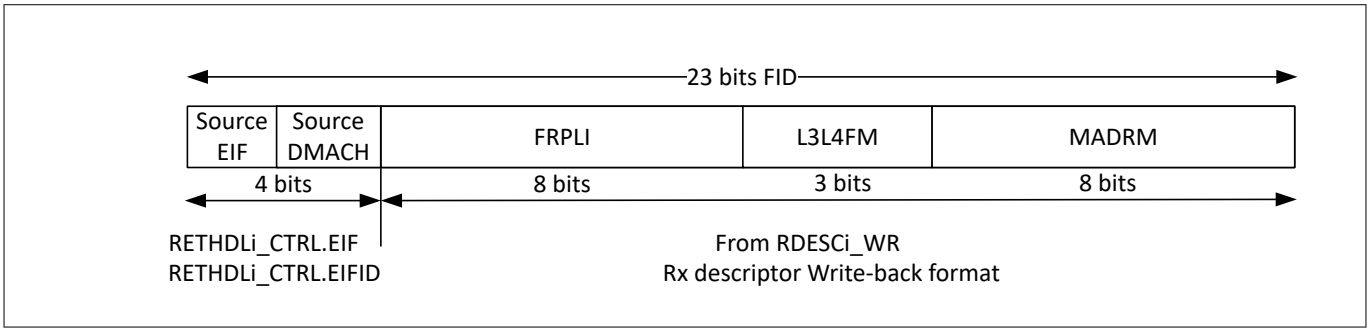


Figure 362 Forwarding ID

The Forwarding Engine determines the destination of the Ethernet frame using this FID within the Forwarding table. The Forwarding table consists of a set of user-configurable forward filter IDs FT_FEj_FRULE.FID1, FT_FEj_FID2.FID2 and a bit-encoded forward rule FT_FEj_FRULE.DSEL consisting of 6 bits. The forward rule is used to select the Tx descriptor list of the corresponding destination DMA channel. The Forwarding Engine performs the following operations:

1. **Forward ID filtering:** Each filter element is executed until the first matching element is found. Forward ID filtering stops at the first matching element. The remaining filter elements are not evaluated for this frame. The following types of filtering mode can be configured by setting FT_FEj_FRULE.FMODE accordingly:
   a. Classic ID filter: FID1 used as the filter ID and FID2 used as mask
   b. Dual ID filter: FID1 or FID2 used as filter ID and FTCFG.FID should be an exact match
   c. Range ID filter: Range of IDs between FID1 and FID2 are used as filter IDs. The FTCFG.FID should be between FID1 and FID2
2. **Frame forwarding:** When there is a matching filter element, the Forwarding Engine sets the EREQ.FWDi_REQ of the corresponding Tx descriptor list FT_FEj_FRULE.DSEL. It gives control of the Ethernet frame in the EIBUF to the Tx Descriptor Handler. The Tx Ethernet Descriptor Handler then prepares the Tx descriptors for the frame in EIBUF depending on the configured forward rule FT_FEj_FRULE.DSEL

Related information

Tx descriptor handler on page 3781
Rx descriptor handler on page 3784
Ethernet input buffer on page 3733
Ethernet address database (EAD) on page 3740
Forwarding table on page 3739
Validation of the Ethernet input frame on page 3754

20.7.3.4 Descriptor error handling

The Ethernet DMA channel signals an error through the status information within the Write-back format of the descriptors.

During any SRI bus error while sending the Transmit or Receive Poll demand, the EOBUFj_ERROR.TDESE or EIBUFi_ERROR.RDESE is set for transmit and receive respectively. ME_ERR.SRIBE is set and FEDID and FESID bits are set accordingly.

Error in Tx descriptors

- In the case of GETH, error in Tx descriptors is indicated by bit TDESCi_WR3G.DERR
- In the case of LETH, error in Tx descriptors is indicated by bit TDESCi_WR3L.DE and bit TDESCi_WR3L.ES indicates the type of error

In the event of a Tx descriptor error, the Tx Descriptor Handler retains the Tx descriptors and the frame in EOBUF or EIBUF. It sets the error flag EOBUFj_ERROR.TDESE and EIBUFi_STATUS.FE (in the case of FWDi_REQ from EIBUF) and triggers an interrupt. The software then fixes the root cause of the error, it fetches the Ethernet frame from the buffer, clears the request TXi_REQ or FWDi_REQ and then clears the TDESE or RDESE.

Meanwhile, the Tx Descriptor Handler processes the remaining requests in round-robin fashion, as described in the chapter Tx Descriptor Handler. During any Tx descriptor error caused by a FWDi_REQ, the TXi_REQ is blocked until the TDESE belonging to the EOBUFi is cleared by the software as the Tx descriptors are mapped to the same Tx DMA channel.

Error in Rx descriptors

- In the case of GETH, error in Rx descriptors or in the received packet is indicated by bit RDESCi_WR3G.ES
- In the case of LETH, error in Rx descriptors or in the received packet is indicated by RDESCi_WR3L.ES

In the event of an Rx descriptor error, the Rx Descriptor Handler retains the Rx descriptor Write-back format and the frame in the EIBUF. It sets the error flag EIBUFi_ERROR.RDESE, EIBUFi_STATUS.FE and triggers an interrupt. The software then fetches the Ethernet frame from the buffer and clears the error flag. Once the error flag is cleared, the EIBUF is used for the next frame.

Meanwhile, the Rx Descriptor Handler processes the next free EIBUF as described in the chapter Rx Descriptor Handler.

Related information

Tx descriptor handler on page 3781

Rx descriptor handler on page 3784

20.7.3.5 Interrupt grouping

DRE generates status and error events on 16 interrupt lines (INT_0 until INT_15) to the Interrupt Router.

The corresponding interrupt lines are enabled by configuring the IE register. A trigger event on an interrupt line is shown in the INTSIG register even if the interrupt is disabled. INT_i (i= 0 to 15) output will be a pulse when a trigger event for interrupt i happens even if INTSIG[i] is already set at that point in time. The 16 interrupt lines are mapped to the following status and error events:

Table 951 Interrupt grouping table

Interrupt line	Interrupt type	Status and error events mapped
INT_0 to INT_7	Watermark and Wraparound Interrupts	Groups watermark and wraparound events of destination memory routing operations. DMEMi_STATUS.WAF and WMF status events are mapped to one of INT[0:7] as selected by DMEMi_CONFIG.INP
INT_8	CAN Input Buffer List Interrupts	Groups the following CIBL error interrupts: • CIBL_STATUS.BF • CIBL_STATUS.CRCE • CIBL_STATUS.WDTE
INT_9	CAN Output Buffer List Interrupts	Groups the following COBL error interrupts: • COBL_STATUS.BF • COBL_STATUS.WDTE
INT_10	EIBUF Frame Error Interrupts	Groups the following EIBUF error interrupts: • EIBUFi_STATUS.FE • EIBUFi_STATUS.IFT • EIBUFi_STATUS.LME • EIBUFi_STATUS.RFE • EIBUFi_STATUS.IDID • EIBUFi_STATUS.CFE • EIBUFi_ERROR.WDTE • EIBUFi_ERROR.RDESE • EIBUFi_ERROR.BF
INT_11	Ethernet Frame Receive Request Interrupt	EIBUFi_STATUS.RXREQ
INT_12	Routing Table Error Interrupts	Groups RS.IRT and NMFE
INT_13	ME Routing Transaction Lost Interrupts	Groups the following ME error interrupts: • ME_ERR.DBOE • ME_ERR.SRIBE • ME_ERR.SPBBE • ME_ERR.IRDE

(table continues...)

Table 951 (continued) Interrupt grouping table

Interrupt line	Interrupt type	Status and error events mapped
INT_14	EOBUF Frame Error Interrupts	Groups the following EOBUF error interrupts: • EOBUFj_STATUS.BF • EOBUFj_ERROR.WDTE • EOBUFj_ERROR.TDESE
INT_15	Ethernet Frame Transmit Request Interrupt	EOBUFj_STATUS.TXREQ

When multiple events are mapped to the same interrupt lines, the corresponding interrupt line is triggered when at least one of the relevant status or error events has occurred (logical OR operation on status or error event).

Related information

Ethernet input buffer on page 3733

20.8 Registers

20.8.1 Register overview - access mode glossary

Table 952 Register overview - access mode glossary

Keyword	Description
E	Access protection using PROT register PROTE.
SE	Access protection using PROT register PROTSE.
APU-PETHj (j=0-5)	Protection group consisting of registers ETHj_ACCEN_WRA, ETHj_ACCEN_WRB, ETHj_ACCEN_RDA, ETHj_ACCEN_RDB, ETHj_ACCEN_VM, ETHj_ACCEN_PRS.
PETHj	Access protection using *APU-PETHj* registers.
APU-PG	Protection group consisting of registers ACCEN_WRA, ACCEN_WRB, ACCEN_RDA, ACCEN_RDB, ACCEN_VM, ACCEN_PRS.
PG	Access protection using *APU-PG* registers.
32	Access only when using 32-bit width.
SV	Access only when supervisor mode is active on the interconnect.
BE	Always returns a Bus Error.
U	No access restrictions.
PROT	Access restrictions as defined in the PROT register access rules.
P	Description can be found in global access mode definition.
nBE	Indicates that no Bus Error is generated when accessing this address range, even though it is either an access to an undefined address or the access does not follow the given rules.

20.8.2 Register overview - DRE (ascending offset address)

Table 953 Register overview - DRE (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CLC	Clock Control Register	00000_H	PG, 32	PG, SV, E, 32	Application Reset	3798
OCS	OCDS Control and Status Register	00004_H	PG, 32	SV, PG, 32	Debug Reset	3799
ID	Module Identification Register	00008_H	PG, 32	BE	PowerOn Reset	3800
RST_CTRLA	Reset Control Register A	$0000C_H$	PG, 32	PG, SV, E, 32	Application Reset	3801
RST_CTRLB	Reset Control Register B	00010_H	PG, 32	PG, SV, E, 32	Application Reset	3801
RST_STAT	Reset Status Register	00014_H	PG, 32	BE	Application Reset	3802
PROTE	PROT Register Endinit	00018_H	U	SV, PROT	Application Reset	3803
PROTSE	PROT Register Safe Endinit	$0001C_H$	U	SV, PROT	Application Reset	3804
ACCEN_WRA	Write access enable register A	00020_H	32	SE, SV, 32	Application Reset	3806
ACCEN_WRB	Write access enable register B	00024_H	32	SE, SV, 32	Application Reset	3807
ACCEN_RDA	Read access enable register A	00028_H	32	SE, SV, 32	Application Reset	3807
ACCEN_RDB	Read access enable register B	$0002C_H$	32	SE, SV, 32	Application Reset	3808
ACCEN_VM	VM access enable register	00030_H	32	SE, SV, 32	Application Reset	3808
ACCEN_PRS	PRS access enable register	00034_H	32	SE, SV, 32	Application Reset	3809
ETHj_ACCEN_WRA	Write access enable register A j	00040_H+j *20_H	32	SE, SV, 32	Application Reset	3810
ETHj_ACCEN_WRB	Write access enable register B j	00044_H+j *20_H	32	SE, SV, 32	Application Reset	3810
ETHj_ACCEN_RDA	Read access enable register A j	00048_H+j *20_H	32	SE, SV, 32	Application Reset	3811
ETHj_ACCEN_RDB	Read access enable register B j	$0004C_H$+ j*20_H	32	SE, SV, 32	Application Reset	3811
ETHj_ACCEN_VM	VM access enable register j	00050_H+j *20_H	32	SE, SV, 32	Application Reset	3812
ETHj_ACCEN_PRS	PRS access enable register j	00054_H+j *20_H	32	SE, SV, 32	Application Reset	3812

(table continues...)

Table 953 (continued) Register overview - DRE (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
MODEr	RP r mode register	01040_H+r*4	PG, 32	E, SV, PG, 32	Application Reset	3813
CANi_RP	CAN i resource partition	01060_H+i*4	PG, 32	E, SV, PG, 32	Kernel Reset	3814
CIBL_BPR	CAN input buffer pending request	$010B8_H$	PG, 32	PG, 32	Kernel Reset	3814
CIBL_STATUS	CAN input buffer list status	$010BC_H$	PG, 32	PG, 32	Kernel Reset	3815
COBL_BPR0	CAN output buffer pending request 0	$010C8_H$	PG, 32	PG, 32	Kernel Reset	3816
COBL_BPR1	CAN output buffer pending request 1	$010CC_H$	PG, 32	PG, 32	Kernel Reset	3817
COBL_STATUS	CAN output buffer list status	$010D0_H$	PG, 32	PG, 32	Kernel Reset	3818
EIBUFi_CONFIG	Ethernet input buffer i configuration	$010D8_H$+i*14_H	PG, 32	E, SV, PG, 32	Kernel Reset	3818
EIBUFi_ERROR	Ethernet input buffer i error	$010E0_H$+i*14_H	PG, 32	PG, 32	Kernel Reset	3819
EIBUFi_STATUS	Ethernet input buffer i status	$010E4_H$+i*14_H	PG, 32	PG, 32	Kernel Reset	3821
EOBUFj_CONFIG	Ethernet output buffer j configuration	01150_H+j*38_H	PG, 32	E, SV, PG, 32	Kernel Reset	3823
EOBUFj_MAC_H0	Ethernet output buffer j MAC header 0	01154_H+j*38_H	PG, 32	PG, 32	Kernel Reset	3825
EOBUFj_MAC_H1	Ethernet output buffer j MAC header 1	01158_H+j*38_H	PG, 32	PG, 32	Kernel Reset	3826
EOBUFj_MAC_H2	Ethernet output buffer j MAC header 2	$0115C_H$+j*38_H	PG, 32	PG, 32	Kernel Reset	3826
EOBUFj_MAC_H3	Ethernet output buffer j MAC header 3	01160_H+j*38_H	PG, 32	PG, 32	Kernel Reset	3827
EOBUFj_MAC_H4	Ethernet output buffer j MAC header 4	01164_H+j*38_H	PG, 32	PG, 32	Kernel Reset	3827
EOBUFj_NTSCF_H0	Ethernet output buffer j NTSCF header	01168_H+j*38_H	PG, 32	PG, 32	Kernel Reset	3828
EOBUFj_NTSCF_STREAM0_ID	Ethernet output buffer j Stream ID configuration 0	$0116C_H$+j*38_H	PG, 32	PG, 32	Kernel Reset	3829
EOBUFj_NTSCF_STREAM1_ID	Ethernet output buffer j Stream ID configuration 1	01170_H+j*38_H	PG, 32	PG, 32	Kernel Reset	3830
EOBUFj_STATUS	Ethernet output buffer j status	01174_H+j*38_H	PG, 32	PG, 32	Kernel Reset	3830
EOBUFj_TTC	Ethernet output buffer j Transmit trigger configuration	01178_H+j*38_H	PG, 32	PG, 32	Kernel Reset	3832

(table continues...)

Table 953 (continued) Register overview - DRE (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
EOBUFj_TTS	Ethernet output buffer j Timer threshold and status	$0117C_H$+ j*38_H	PG, 32	PG, 32	Kernel Reset	3832
EOBUFj_ERROR	Ethernet output buffer j error	01180_H+j *38_H	PG, 32	PG, 32	Kernel Reset	3833
SIDFi_FC	Stream ID filter i configuration	$0129C_H$+ i*14_H	PG, 32	PG, 32	Kernel Reset	3834
SIDFi_FIL1_L	Stream ID filter i configuration Stream ID filter 1 lower	$012A0_H$+ i*14_H	PG, 32	PG, 32	Kernel Reset	3835
SIDFi_FIL1_H	Stream ID filter i configuration Stream ID filter 1 higher	$012A4_H$+ i*14_H	PG, 32	PG, 32	Kernel Reset	3835
SIDFi_FIL2_L	Stream ID filter i configuration Stream ID filter 2 lower	$012A8_H$+ i*14_H	PG, 32	PG, 32	Kernel Reset	3836
SIDFi_FIL2_H	Stream ID filter i configuration Stream ID filter 2 higher	$012AC_H$+ i*14_H	PG, 32	PG, 32	Kernel Reset	3836
RTi_CONFIG	CAN transmit routing table i configuration	01340_H+i *8	PG, 32	E, SV, PG, 32	Kernel Reset	3837
RREQ_CONFIG	Routing request configuration	$0135C_H$	PG, 32	BE	Kernel Reset	3837
RREQ_CID	CAN ID request	01360_H	PG, 32	BE	Kernel Reset	3838
UCRH	Uni-cast routing header	01364_H	PG, 32	BE	Kernel Reset	3838
MCRH	Multi-cast routing header	01364_H	PG, 32	BE	Kernel Reset	3839
RS	Routing status	01368_H	PG, 32	PG, 32	Kernel Reset	3840
CANRXR0	CAN receive request 0	$0136C_H$	PG, 32	BE	Kernel Reset	3840
CANRXR1	CAN receive request 1	01370_H	PG, 32	BE	Kernel Reset	3841
CANTXR	CAN transmit buffer available request	01374_H	PG, 32	BE	Kernel Reset	3842
DMEMi_CONFIG	Destination memory i configuration	$0137C_H$+ i*20_H	PG, 32	E, SV, PG, 32	Kernel Reset	3842
DMEMi_MODE	Destination memory i transfer mode configuration	01380_H+i *20_H	PG, 32	E, SV, PG, 32	Kernel Reset	3844
DMEMi_STATUS	Destination memory i status	01390_H+i *20_H	PG, 32	PG, 32	Kernel Reset	3845
DMEMi_RP	Destination memory i resource partition	01394_H+i *20_H	PG, 32	E, SV, PG, 32	Kernel Reset	3846
ME_SRCA	Move engine source address	$016F8_H$	PG, 32	BE	Kernel Reset	3847

(table continues...)

Table 953 (continued) Register overview - DRE (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ME_DESTA	Move engine destination address	$016FC_H$	PG, 32	BE	Kernel Reset	3847
ME_STATE	Move engine state	01700_H	PG, 32	BE	Kernel Reset	3847
ME_FESRCA	Move engine first error source address	01704_H	PG, 32	BE	Kernel Reset	3848
ME_FEDESTA	Move engine first error destination address	01708_H	PG, 32	BE	Kernel Reset	3849
ME_ERR	Move engine error register	$0170C_H$	PG, 32	PG, 32	Kernel Reset	3849
INTSIG	Interrupt signal	01710_H	PG, 32	BE	Kernel Reset	3851
IE	Interrupt line enable	01714_H	PG, 32	PG, 32	Kernel Reset	3853
RETHDLi_CTRL	Rx Ethernet descriptor list i configuration and control	$0171C_H$+ i*8	PG, 32	E, SV, PG, 32	Kernel Reset	3854
TETHDLi_CTRL	Tx Ethernet descriptor list i configuration and control	$0174C_H$+ i*10_H	PG, 32	E, SV, PG, 32	Kernel Reset	3856
EDLSTAT	Ethernet descriptor list status	$017D8_H$	PG, 32	PG, 32	Kernel Reset	3858
EREQ	Ethernet requests summary	$017DC_H$	PG, 32	PG, 32	Kernel Reset	3859
FTCFG	Forwarding table configuration	$017E4_H$	PG, 32	E, SV, PG, 32	Kernel Reset	3860
CWDCFG	DRE CAN watchdog configuration	$017EC_H$	PG, 32	E, SV, PG, 32	Kernel Reset	3861
EWDCFG	DRE Ethernet watchdog configuration	$017F4_H$	PG, 32	E, SV, PG, 32	Kernel Reset	3862
EADCFG	Ethernet address database configuration	$017F8_H$	PG, 32	E, SV, PG, 32	Kernel Reset	3862
DMAi_RP	DMA i resource partition	$017FC_H$+ i*4	PG, 32	E, SV, PG, 32	Kernel Reset	3863
CITO	CAN Input buffer timeout status	01818_H	PG, 32	PG, 32	Kernel Reset	3863
COTO0	CAN Output buffer timeout status 0	$0181C_H$	PG, 32	PG, 32	Kernel Reset	3864
COTO1	CAN Output buffer timeout status 1	01820_H	PG, 32	PG, 32	Kernel Reset	3864

20.8.3 Register overview - DRE CAN Address Database RAM interface (ascending offset address)

Table 954 Register overview - DRE CAN Address Database RAM interface (ascending offset address)

Short name	Long name	Offset address	See
CAD_CANi_CRESA	RAM CAN address database CRE start address	00000_H+i*8	3865

20.8.4 Register overview - DRE CAN Input Buffer List RAM interface (ascending offset address)

Table 955 Register overview - DRE CAN Input Buffer List RAM interface (ascending offset address)

Short name	Long name	Offset address	See
CIBUFj_RHEAD	RAM CIBUF routing header	$000A0_H$+j*50_H	3865
CIBUFj_CRC	RAM CIBUF CRC computed by CRE	$000A4_H$+j*50_H	3866
CIBUFj_R0	RAM CIBUF register 0	$000A8_H$+j*50_H	3866
CIBUFj_R1	RAM CIBUF register 1	$000AC_H$+j*50_H	3867
CIBUFj_DBm	RAM CIBUF data byte m	$000B0_H$+j*50_H +m	3868

20.8.5 Register overview - DRE CAN Output Buffer List RAM interface (ascending offset address)

Table 956 Register overview - DRE CAN Output Buffer List RAM interface (ascending offset address)

Short name	Long name	Offset address	See
COBUFj_UCRH	RAM uni-cast routing header	$006E0_H$+j*50_H	3868
COBUFj_MCRH	RAM multi-cast routing header	$006E0_H$+j*50_H	3869
COBUFj_CRC	RAM COBUF CRC computed by DRE	$006E4_H$+j*50_H	3870
COBUFj_R0	RAM COBUF register 0	$006E8_H$+j*50_H	3871
COBUFj_R1	RAM COBUF register 1	$006EC_H$+j*50_H	3872
COBUFj_DB	RAM COBUF data byte m	$006F0_H$+j*50_H +m	3873

20.8.6 Register overview - DRE Ethernet descriptors RAM interface (ascending offset address)

Table 957 Register overview - DRE Ethernet descriptors RAM interface (ascending offset address)

Short name	Long name	Offset address	See
TDESCi_RD0	RAM TDESC word 0 read format	03140_H+i*10_H	3877
TDESCi_WR0	RAM TDESC word 0 Write-back format	03140_H+i*10_H	3878
TDESCi_RD1	RAM TDESC word 1 read format	03144_H+i*10_H	3878
TDESCi_WR1	RAM TDESC word 1 Write-back format	03144_H+i*10_H	3879
TDESCi_RD2	RAM TDESC word 2 read format	03148_H+i*10_H	3879
TDESCi_WR2	RAM TDESC word 2 Write-back format	03148_H+i*10_H	3880
TDESCi_RD3	RAM TDESC word 3 read format	$0314C_H$+i*10_H	3880
TDESCi_WR3G	RAM TDESC word 3 Write-back format for GETH	$0314C_H$+i*10_H	3882
TDESCi_WR3L	RAM TDESC word 3 Write-back format for LETH	$0314C_H$+i*10_H	3883

(table continues...)

Table 957 (continued) Register overview - DRE Ethernet descriptors RAM interface (ascending offset address)

Short name	Long name	Offset address	See
RDESCi_RD0	RAM RDESC word 0 read format	03180_H+i*10_H	3886
RDESCi_WR0NT	RAM RDESC word 0 Write-back format Non Tunneled frames (also LETH)	03180_H+i*10_H	3886
RDESCi_WR0T	RAM RDESC word 0 Write-back format Tunneled frames	03180_H+i*10_H	3887
RDESCi_RD1	RAM RDESC word 1 read format	03184_H+i*10_H	3887
RDESCi_WR1G	RAM RDESC word 1 Write-back format GETH	03184_H+i*10_H	3888
RDESCi_WR1L	RAM RDESC word 1 Write-back format LETH	03184_H+i*10_H	3888
RDESCi_RD2	RAM RDESC word 2 read format	03188_H+i*10_H	3891
RDESCi_WR2G	RAM RDESC word 2 Write-back format GETH	03188_H+i*10_H	3891
RDESCi_WR2L	RAM RDESC word 2 Write-back format LETH	03188_H+i*10_H	3894
RDESCi_RD3G	RAM RDESC word 3 read format GETH	$0318C_H$+i*10_H	3896
RDESCi_RD3L	RAM RDESC word 3 read format LETH	$0318C_H$+i*10_H	3897
RDESCi_WR3G	RAM RDESC word 3 Write-back format GETH	$0318C_H$+i*10_H	3898
RDESCi_WR3L	RAM RDESC word 3 Write-back format LETH	$0318C_H$+i*10_H	3900

20.8.7 Register overview - DRE DMEM parameter table RAM interface (ascending offset address)

Table 958 Register overview - DRE DMEM parameter table RAM interface (ascending offset address)

Short name	Long name	Offset address	See
DMEMi_SA	RAM Destination memory start address	$07A40_H$+i*10_H	3904
DMEMi_FDBI	RAM Destination memory frame data buffer index	$07A44_H$+i*10_H	3905
DMEMi_WM	RAM Destination memory watermark level	$07A48_H$+i*10_H	3905
DMEMi_WA	RAM Destination memory wraparound level	$07A4C_H$+i*10_H	3906

20.8.8 Register overview - DRE Ethernet Address Database RAM interface (ascending offset address)

Table 959 Register overview - DRE Ethernet Address Database RAM interface (ascending offset address)

Short name	Long name	Offset address	See
EAD_LETHj_TXDMA	RAM LETH Tx DMA channel address	$02AE0_H$+j*8	3876
EAD_LETHj_RXDMA	RAM LETH Rx DMA channel address	$02AE4_H$+j*8	3876
EAD_GETHj_TXDMA	RAM GETH Tx DMA channel address	$02B00_H$+j*8	3876
EAD_GETHj_RXDMA	RAM GETH Rx DMA channel address	$02B04_H$+j*8	3877

20.8.9 Register overview - DRE Forwarding table RAM interface (ascending offset address)

Table 960 Register overview - DRE Forwarding table RAM interface (ascending offset address)

Short name	Long name	Offset address	See
FT_FEj_FRULE	RAM Forwarding rule and FID1	07640_H+j*8	3902
FT_FEj_FID2	RAM Forward filter ID2	07644_H+j*8	3904

20.8.10 Register overview - DRE CAN Transmit Routing Table RAM interface (ascending offset address)

Table 961 Register overview - DRE CAN Transmit Routing Table RAM interface (ascending offset address)

Short name	Long name	Offset address	See
RT_REj_CIDFC	RAM routing table CAN ID filter configuration	$01AE0_H$+j*8	3873
RT_REj_UCR	RAM routing table uni-cast routing	$01AE4_H$+j*8	3874
RT_REj_MCR	RAM routing table multi-cast routing	$01AE4_H$+j*8	3875

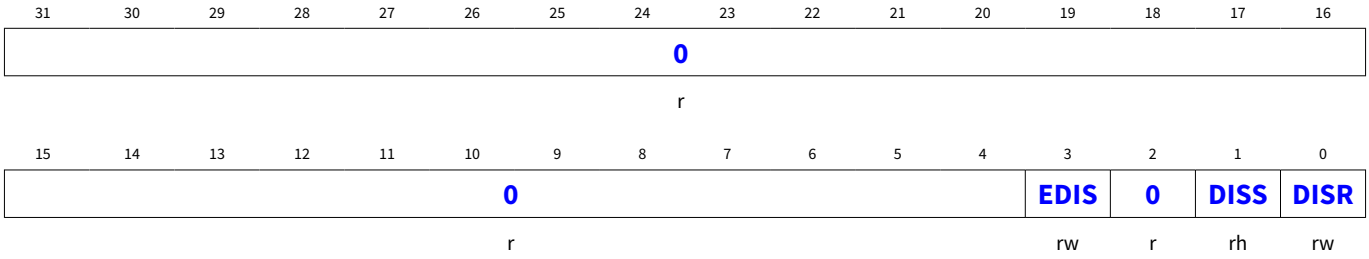
20.8.11 Clock Control Register

The Clock Control Register CLC allows the programmer to adapt the functionality and power consumption of the module to the requirements of the application.

Register CLC controls the module clock signal and the reactivity to the sleep signal.

CLC Offset address: 00000_H

Clock Control Register Application Reset value: $0000\ 0003_H$



Field	Bits	Type	Description
DISR	0	rw	**Module Disable Request Bit** Used for enable/disable control of the module. 0_B On request: enable the module clock 1_B Off request: stop the module clock

(table continues...)

(continued)

Field	Bits	Type	Description
DISS	1	rh	**Module Disable Status Bit** When CLC.DISS=1B, registers CLC, OCS, ID, RST_CTRLA, RST_CTRLB, PROTE, PROTSE, ACCEN* and ETH_ACCEN* are still accessible. Accesses to other DRE registers are not executed and will result in a bus error. 0_B Module clock is enabled 1_B Off: module is not clocked
EDIS	3	rw	**Sleep Mode Enable Control** Used to control the module's reaction to sleep mode. Note: Sleep mode should not be enabled when there are ongoing or pending transactions between the DRE and Ethernet 0_B Sleep mode request is enabled and functional 1_B Module disregards the sleep mode control signal
0	2, 31:4	r	**Reserved** Read as 0; should be written with 0.

20.8.12 OCDS Control and Status Register

The OCDS Control and Status register OCS controls the debug and trace behavior by selecting suspend modes and OTGB trigger sets. When OCDS is disabled the suspend control is ineffective.

OCS Offset address: 00004_H

OCDS Control and Status Register Debug Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		SUSSTA	SUS_P	SUS				0							
r		rh	w	rw				r							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															
r															

Field	Bits	Type	Description
SUS	27:24	rw	**OCDS Suspend Control** Controls the sensitivity to the suspend signal coming from the OCDS Trigger Switch (OTGS) Not listed combinations are reserved. 0_H Will not suspend 1_H Reserved, Do not use 2_H Soft suspend mode Clocks will be disabled after kernel acknowledge. The DRE waits for the onging transactions on the SRI and FPI Master interfaces to finish. SRI access remains operational. Read accesses are allowed on the SRI slave interface for the entire SFR space and the RAM memory. Write access are allowed on the SRI slave interface for the RAM memory and the registers on the application domain (common registers). Write access to Kernel domain registers will have no effect. The DRE resumes operation from the point where it was stopped upon a suspend request. Note: Suspend mode should not be enabled when there are ongoing or pending transactions. Rapid sequence of suspend requests should be avoided in order to avoid false acknowledge. **others,** Reserved, do not use
SUS_P	28	w	**SUS Write Protection** SUS is only written when SUS_P is 1, otherwise unchanged. Read as 0.
SUSSTA	29	rh	**Suspend State** 0_B Module is not (yet) suspended 1_B Module is suspended
0	23:0, 31:30	r	**Reserved** Read as 0; should be written with 0.

Table 962 Access mode restrictions of OCS sorted by descending priority

Mode name	Access mode		Description
write 1 to .**SUS_P**	rw	SUS	Set SUS_P during write access
(default)	r	SUS	

20.8.13 Module Identification Register

ID Offset address: 00008_H

Module Identification Register PowerOn Reset value: $0102\ C002_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MOD_NUM															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MOD_TYPE								MOD_REV							
r								r							

Field	Bits	Type	Description
MOD_REV	7:0	r	**Module Revision** Indicates the revision number of the implementation.
MOD_TYPE	15:8	r	**Module Type** The bit-field is set to $C0_H$ which defines the module as a 32-bit module.
MOD_NUM	31:16	r	**Module Number Value** This bit-field defines a module identification number.

20.8.14 Reset Control Register A

RST_CTRLA Offset address: $0000C_H$

Reset Control Register A Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

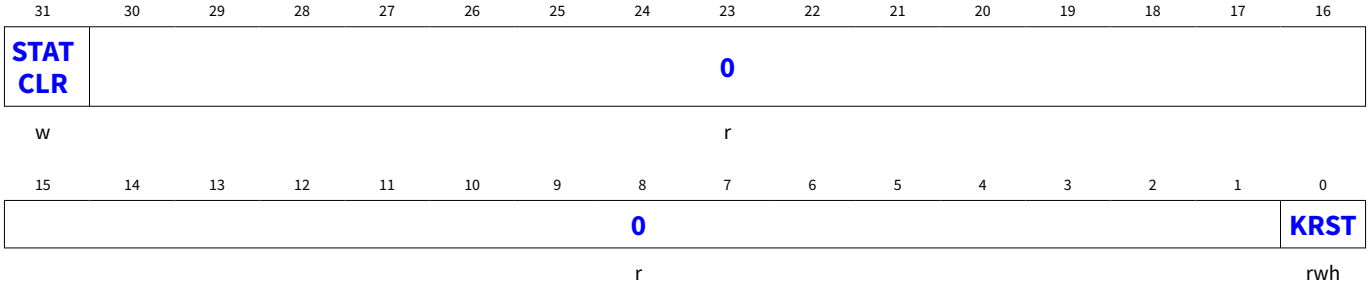
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				GRSTEN3	GRSTEN2	GRSTEN1	GRSTEN0	0							KRST
r				rw	rw	rw	rw	r							rwh

Field	Bits	Type	Description
KRST	0	rwh	**Kernel Reset** Request a kernel reset. The reset is executed if the reset bits of both kernel reset registers are set. KRST is cleared after the kernel reset was executed. 0_B No action 1_B A kernel reset was requested
GRSTENx (x=0-3)	x+8	rw	**Enable for Global Module Reset Group x** 0_B Global module reset group x does not have any effect 1_B Global module reset group x results in a kernel reset
0	7:1, 31:12	r	**Reserved** Read as 0; should be written with 0.

20.8.15 Reset Control Register B

RST_CTRLB Offset address: 00010_H

Reset Control Register B Application Reset value: $0000\ 0000_H$



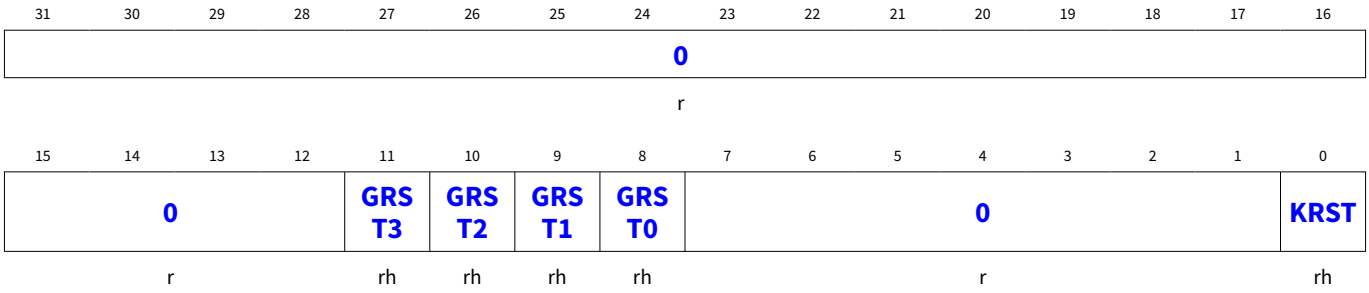
Field	Bits	Type	Description
KRST	0	rwh	**Kernel Reset** Request a kernel reset. The reset is executed if the reset bits of both kernel reset registers are set. KRST is cleared after the kernel reset was executed. 0_B No action 1_B A kernel reset was requested
STATCLR	31	w	**Kernel Reset Status Clear** Clears all status bits in RST_STAT when 1 is written. Read will return 0. 0_B No action 1_B Write with ´1´ clears bits STAT.GRSTx and bit STAT.KRST .
0	30:1	r	**Reserved** Read as 0; should be written with 0.

20.8.16 Reset Status Register

The reset status register contains the status bits for kernel reset (KRST) and the global module reset groups (GRSTx).

RST_STAT Offset address: 00014_H

Reset Status Register Application Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
KRST	0	rh	**Kernel Reset Status** Indicates an executed kernel reset. RST_STAT.KRST is set after the execution of a kernel reset in the same clock cycle in which the reset bits are cleared. Clear KRST by setting bit STATCLR in register RST_CTRLB.

(table continues...)

(continued)

Field	Bits	Type	Description
GRSTx (x=0-3)	x+8	rh	**Status for Global Module Reset Group x** 0_B Reset was not triggered by global reset group x 1_B Reset was triggered by global reset group x
0	7:1, 31:12	r	**Reserved** Read as 0; should be written with 0.

20.8.17 PROT Register Endinit

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner and in some PROT states, the secure master to update the PROT state as described in the PROT mechanism.

The PROTE register controls lock / unlock of the local Endinit (E) protected control registers.

PROTE Offset address: 00018_H

PROT Register Endinit Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OWEN	ODEF	TAGID						PRSEN	PRS			VMEN	VM		
w	rw	rw						rw	rw			rw	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												SWEN	STATE		
r												w	rwh		

Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
VM	18:16	rw	**Virtual machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.

(table continues...)

(continued)

Field	Bits	Type	Description
VMEN	19	rw	**Virtual machine definition enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.
PRSEN	23	rw	**Protection set definition enable for PROT owner** 0_B **Disable**: PRS is not part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG-ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; should be written with 0.

Table 963 Access mode restrictions of PROTE sorted by descending priority

Mode name	Access mode		Description
write 1 to .**SWEN**	rwh	STATE	
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

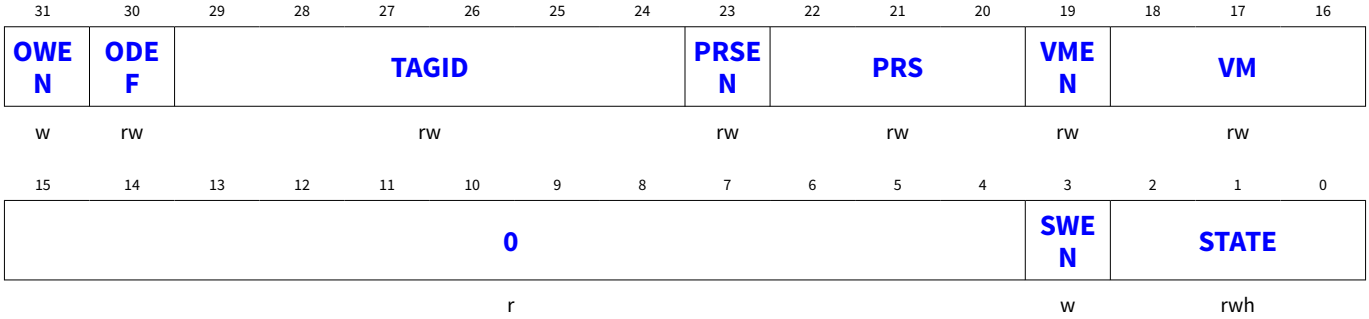
20.8.18 PROT Register Safe Endinit

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner and in some PROT states, the secure master to update the PROT state as described in the PROT mechanism.

The PROTSE register controls lock / unlock of the local Safe Endinit (SE) protected control registers.

PROTSE Offset address: $0001C_H$

PROT Register Safe Endinit Application Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
VM	18:16	rw	**Virtual machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual machine definition enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.
PRSEN	23	rw	**Protection set definition enable for PROT owner** 0_B **Disable**: PRS is not part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG-ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner

(table continues...)

(continued)

Field	Bits	Type	Description
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; should be written with 0.

Table 964 Access mode restrictions of PROTSE sorted by descending priority

Mode name	Access mode		Description
write 1 to .**SWEN**	rwh	STATE	
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

20.8.19 Write access enable register A

ACCEN register set

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

ACCEN_WRA Offset address: 00020_H

Write access enable register A Application Reset value: $1000\ 0003_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	**EN30**	**EN29**	**EN28**	**EN27**	**EN26**	**EN25**	**EN24**	**EN23**	**EN22**	**EN21**	**EN20**	**EN19**	**EN18**	**EN17**	**EN16**
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	**EN14**	**EN13**	**EN12**	**EN11**	**EN10**	**EN09**	**EN08**	**EN07**	**EN06**	**EN05**	**EN04**	**EN03**	**EN02**	**EN01**	**EN00**
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access

20.8.20 Write access enable register B

ACCEN register set

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

ACCEN_WRB Offset address: 00024_H

Write access enable register B Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN63	EN62	EN61	EN60	EN59	EN58	EN57	EN56	EN55	EN54	EN53	EN52	EN51	EN50	EN49	EN48
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN47	EN46	EN45	EN44	EN43	EN42	EN41	EN40	EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-63)	q-32	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access

20.8.21 Read access enable register A

ACCEN register set

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

ACCEN_RDA Offset address: 00028_H

Read access enable register A Application Reset value: $FFFF\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access

20.8.22 Read access enable register B

ACCEN register set

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

ACCEN_RDB Offset address: $0002C_H$

Read access enable register B Application Reset value: $FFFF\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN63	EN62	EN61	EN60	EN59	EN58	EN57	EN56	EN55	EN54	EN53	EN52	EN51	EN50	EN49	EN48
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN47	EN46	EN45	EN44	EN43	EN42	EN41	EN40	EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-63)	q-32	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access

20.8.23 VM access enable register

ACCEN register set

This register defines which virtual machine encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each VMx.

ACCEN_VM Offset address: 00030_H

VM access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for VM ID q** This bit enables read access to the access protected region for transactions with the VM ID q. 0_B Disabled for read access 1_B Enabled for read access

(table continues...)

(continued)

Field	Bits	Type	Description
WRq (q=00-07)	q+16	rw	**Write access enable for VM ID q** This bit enables write access to the access protected region for transactions with the VM ID q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

20.8.24 PRS access enable register

ACCEN register set

This register defines which protection register set encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each PRSx.

ACCEN_PRS Offset address: 00034_H

PRS access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for PRS q** This bit enables read access to the access protected region for transactions with the PRS q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for PRS q** This bit enables write access to the access protected region for transactions with the PRS q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

20.8.25 Write access enable register A j

ACCEN register set

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

ETHj_ACCEN_WRA (j=0-5) Offset address: 00040_H+j*20_H

Write access enable register A j Application Reset value: $1000\ 0003_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access

20.8.26 Write access enable register B j

ACCEN register set

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

ETHj_ACCEN_WRB (j=0-5) Offset address: 00044_H+j*20_H

Write access enable register B j Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN63	EN62	EN61	EN60	EN59	EN58	EN57	EN56	EN55	EN54	EN53	EN52	EN51	EN50	EN49	EN48
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN47	EN46	EN45	EN44	EN43	EN42	EN41	EN40	EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-63)	q-32	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access

20.8.27 Read access enable register A j

ACCEN register set

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

ETHj_ACCEN_RDA (j=0-5) Offset address: 00048_H+j*20_H

Read access enable register A j Application Reset value: FFFF $FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access

20.8.28 Read access enable register B j

ACCEN register set

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

ETHj_ACCEN_RDB (j=0-5) Offset address: $0004C_H+j*20_H$

Read access enable register B j Application Reset value: FFFF $FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN63	EN62	EN61	EN60	EN59	EN58	EN57	EN56	EN55	EN54	EN53	EN52	EN51	EN50	EN49	EN48
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN47	EN46	EN45	EN44	EN43	EN42	EN41	EN40	EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-63)	q-32	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access

20.8.29 VM access enable register j

ACCEN register set

This register defines which virtual machine encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each VMx.

ETHj_ACCEN_VM (j=0-5) Offset address: 00050_H+j*20_H

VM access enable register j Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for VM ID q** This bit enables read access to the access protected region for transactions with the VM ID q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for VM ID q** This bit enables write access to the access protected region for transactions with the VM ID q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

20.8.30 PRS access enable register j

ACCEN register set

This register defines which protection register set encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each PRSx.

ETHj_ACCEN_PRS (j=0-5) Offset address: 00054_H+j*20_H

PRS access enable register j Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for PRS q** This bit enables read access to the access protected region for transactions with the PRS q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for PRS q** This bit enables write access to the access protected region for transactions with the PRS q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

20.8.31 RP r mode register

The Mode Register defines the attributes used by the DRE bus master interfaces when a routing operation related to a Resource Partition (RP) accesses the on chip buses.

MODEr (r=0-7) Offset address: 01040_H+r*4

RP r mode register Application Reset value: $0000\ 0001_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0							TAGOFF	PRSEN	PRS			VMEN	VM		
r							rw	rw	rw			rw	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															MODE
r															rw

Field	Bits	Type	Description
MODE	0	rw	**Resource Partition Supervisor Mode** 0_B Bus master interface accesses on chip bus in user mode. 1_B Bus master interface accesses on chip bus in supervisor mode.
VM	18:16	rw	**Virtual Machine** Bus master interface accesses fabric with extended VM tag ID configured in this bit-field
VMEN	19	rw	**Virtual Machine Enable** 0_B **Disable**: VM disabled 1_B **Enable**: VM enabled
PRS	22:20	rw	**Protection Set** Bus master interface accesses fabric with extended PRS tag ID configured in this bit-field

(table continues...)

(continued)

Field	Bits	Type	Description
PRSEN	23	rw	**Protection Set Enable** 0_B **Disable**: PRS disabled 1_B **Enable**: PRS enabled
TAGOFF	24	rw	**Tag Offset** Bus master interface accesses fabric with extended tag ID. TAG# = DRE_BASE_TAG (011110b) + TAG_OFF
0	15:1, 31:25	r	**Reserved** Read as 0; should be written with 0.

20.8.32 CAN i resource partition

This SFR configures the resource allocation for access made to the CAN Interfaces

CANi_RP (i=0-19) Offset address: 01060_H+i*4

CAN i resource partition Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0													RPI		
r													rw		

Field	Bits	Type	Description
RPI	2:0	rw	**CAN Resource Partition Index** This bit-field indicates the resource partition allocated to CAN interfaces.
0	31:3	r	**Reserved** Read as 0; should be written with 0.

20.8.33 CAN input buffer pending request

The pending requests for CAN input buffers are set by hardware in this SFR.

CIBL_BPR Offset address: $010B8_H$

CAN input buffer pending request Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0												PR19	PR18	PR17	PR16
r												rwh	rwh	rwh	rwh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PR15	PR14	PR13	PR12	PR11	PR10	PR9	PR8	PR7	PR6	PR5	PR4	PR3	PR2	PR1	PR0
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
PRj (j=0-19)	j	rwh	**Buffer Pending Request** Indicates a pending CAN frame to be proceessed in corresponding buffer. A pending request is set by hardware when a new CAN frame is stored in a buffer. It is cleared by hardware when the CAN frame is processed by ACF CAN-Ethernet Format Engine. Whenever there is a timeout, the corresponding CIBUF is taken out of the arbitration. The software shall clear the timeout status within the CITO register and also clear this bit if needed after reading the frame. In case the same CIBUF is to be reused (included again in the next arbitration), the software shall clear only the timeout status. Software write with 1b also clears the pending request. 0_B No pending Request No request for the corresponding buffer 1_B New pending request A new pending request for the corresponding buffer
0	31:20	r	**Reserved** Read as 0; should be written with 0

20.8.34 CAN input buffer list status

CIBL_STATUS Offset address: $010BC_H$

CAN input buffer list status Kernel Reset value: $0002\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0												WDTE	CRCE	BE	BF
r												rwh	rwh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		CBI						0		PIDX					
r		rh						r		rh					

Field	Bits	Type	Description
PIDX	5:0	rh	**Put Index** This bit-field shows the index corresponding to the buffer at which a new CAN frame will be stored. When Buffer Full flag (BF) is set, value of this bit-field is ignored.
CBI	13:8	rh	**Current CAN Input Buffer Index** This bit-field indicates that CAN input buffer index currently processed by the ACF CAN-Ethernet Format Engine. Valid values are 1 to 20.

(table continues...)

(continued)

Field	Bits	Type	Description
BF	16	rh	**Buffer Full** This bit-field shows the buffer full status. This flag is set by hardware when all the buffers have pending requests set. The flag is cleared by hardware when there is a free buffer available Note: When all the EOBUFs corresponding to the DIDs of the CAN frames in the CIBL are disabled, it leads to a buffer full condition. The software shall enable the EOBUF based on the buffer full interrupt 0_B Free buffers available 1_B CAN Input Buffer List is full
BE	17	rh	**Buffer Empty** This flag shows the buffer empty condition. The flag is set by hardware when there are no buffers with a pending request. The flag is cleared by hardware when atleast one buffer has a pending request set. 0_B Buffer not empty Buffers have pending requests set. 1_B Buffer empty Buffers have no pending requests set
CRCE	18	rwh	**CRC error flag** This bit is set by the hardware to indicate an CRC mismatch while verifying the 16-bit CRC part of the CAN frame in the CIBUF before writing the final block of data into the EOBUF. The software write with 1 clears this bit.
WDTE	19	rwh	**Watchdog timeout error flag** This bit is set by the hardware to indicate an CAN Watchdog timeout error in case of delayed processing of the CAN frame in CIBUF by the ACF CAN-ETH assembler. Software write with 1 clears this bit
0	7:6, 15:14, 31:20	r	**Reserved** Read as 0; should be written with 0.

20.8.35 CAN output buffer pending request 0

The pending requests for CAN output buffers from index 0 to 31 are set by hardware in this register.

COBL_BPR0 — Offset address: $010C8_H$

CAN output buffer pending request 0 — Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PR31	PR30	PR29	PR28	PR27	PR26	PR25	PR24	PR23	PR22	PR21	PR20	PR19	PR18	PR17	PR16
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PR15	PR14	PR13	PR12	PR11	PR10	PR9	PR8	PR7	PR6	PR5	PR4	PR3	PR2	PR1	PR0
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
PRj (j=0-31)	j	rwh	**Buffer Pending Request** Indicates a pending CAN frame to be proceessed in corresponding buffer. A pending request is set by CAN Transmit Routing Engine when a new CAN frame is stored in a buffer. It is cleared by Routing Control Unit when the CAN frame is transferred to corresponding destination(s). Whenever there is a timeout, the corresponding COBUF is taken out of the arbitration. The software shall clear the timeout status within the COTO0 and COTO1 registers and also clear this bit if needed after reading the frame. In case the same COBUF is to be reused (included again in the next arbitration), the software shall clear only the timeout status. Software write with 1 clears this bit 0_B No pending Request No request for the corresponding buffer 1_B New pending request A new pending request for the corresponding buffer

20.8.36 CAN output buffer pending request 1

The pending requests for CAN output buffers from index 32 to 63 are set by hardware in this register.

COBL_BPR1 Offset address: $010CC_H$

CAN output buffer pending request 1 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PR63	PR62	PR61	PR60	PR59	PR58	PR57	PR56	PR55	PR54	PR53	PR52	PR51	PR50	PR49	PR48
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PR47	PR46	PR45	PR44	PR43	PR42	PR41	PR40	PR39	PR38	PR37	PR36	PR35	PR34	PR33	PR32
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
PRj (j=32-63)	j-32	rwh	**Buffer Pending Request** Indicates a pending CAN frame to be proceessed in corresponding buffer. A pending request is set by CAN Transmit Routing Engine when a new CAN frame is stored in a buffer. It is cleared by Routing Control Unit when the CAN frame is transferred to corresponding destination(s). Whenever there is a timeout, the corresponding COBUF is taken out of the arbitration. The software shall clear the timeout status within the COTO0 and COTO1 registers and also clear this bit if needed after reading the frame. In case the same COBUF is to be reused (included again in the next arbitration), the software shall clear only the timeout status. Software write with 1 clears this bit 0_B No pending Request No request for the corresponding buffer 1_B New pending request A new pending request for the corresponding buffer

20.8.37 CAN output buffer list status

COBL_STATUS Offset address: $010D0_H$

CAN output buffer list status Kernel Reset value: $0002\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0												WDTE	0	BE	BF
r												rwh	r	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0									PIDX						
r									rh						

Field	Bits	Type	Description
PIDX	6:0	rh	**Put Index** This bit-field shows the index corresponding to the buffer at which a new CAN frame is stored. When Buffer Full (BF) flag is set, value of this bit-field is ignored.
BF	16	rh	**Buffer Full** This bit-field shows the buffer full status. This flag is set by hardware when all the buffers have pending requests set. The flag is cleared by hardware when there is a free buffer available. 0_B Free buffers available 1_B CAN Output Buffer List is full
BE	17	rh	**Buffer Empty** This flag shows the buffer empty condition. The flag is set by hardware when there are no buffers with a pending request. The flag is cleared by hardware when atleast one buffer has a pending request set. 0_B Buffers have pending requests set 1_B Buffers have no pending requests set
WDTE	19	rwh	**Watchdog timeout error flag** This bit is set by the hardware to indicate an CAN Watchdog timeout error in case of delayed processing of the CAN frame in COBUF by the Routing control unit. Software write with 1 clears this bit
0	15:7, 18, 31:20	r	**Reserved** Read as 0; should be written with 0.

20.8.38 Ethernet input buffer i configuration

EIBUFi_CONFIG (i=0-5) Offset address: $010D8_H+i*14_H$

Ethernet input buffer i configuration Kernel Reset value: $0000\ 0000_H$

Field	Bits	Type	Description
NTSCF_SA	7:2	rw	**Start Address of the NTSCF Header** This bit-field defines the bits [7:2] of the offset address from EIBUF start address which the NTSCF header starts. Note: The start address offset is always 32 bit aligned.
RRF	28	rw	**Reject Remote Frame** When this bit-field is set, ACF_CAN_BRIEF messages with RTR = 1 are not processed. When this bit is not set, there is no CAN payload data copied by the DRE. The frame is copied until R1
0	1:0, 26:8, 27, 31:29	r	**Reserved**

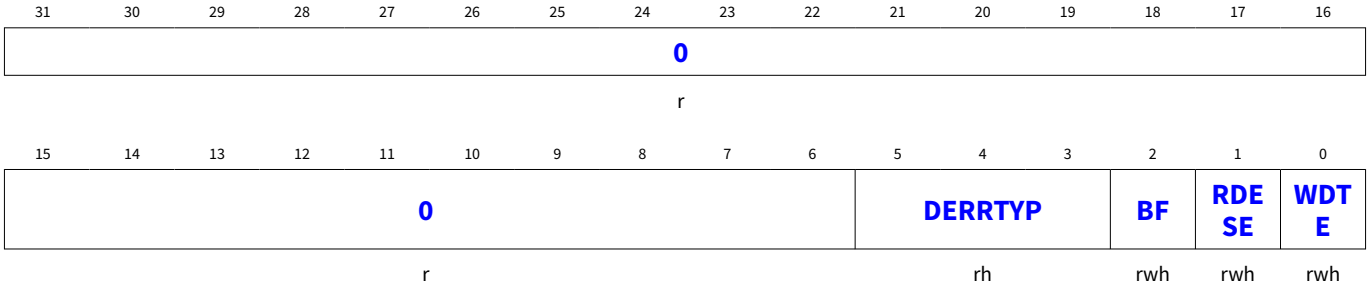
20.8.39 Ethernet input buffer i error

EIBUFi_ERROR (i=0-5)

Ethernet input buffer i error

Offset address: $010E0_H+i*14_H$

Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
WDTE	0	rwh	**Ethernet watchdog timeout error** This bit is set by the hardware to indicate an Ethernet Watchdog timeout error in case of delayed processing of the Ethernet frame in EIBUF by the ACF engine or by the Tx Descriptor Handler. The software shall read the frame directly from the EIBUF and clear this flag. Software write with 1 clears this bit. In case a WDTE event occurs during a forwarding request, the software identifies the EIBUF by checking the EREQ.CBIi and EREQ.FWDi_REQ bits. Clearing the EREQ.FWDi_REQ would automatically clear the EIBUF_STATUS.BPR bit. EIBUF_STATUS.BPR bit could already be cleared by DRE in some cases even after the WDTE error occurs as the forwarding request could be already processed within the time the WDTE error is triggered When there is a watchdog timeout error, the software has to restart the Tx Descriptor Handler by setting the TETHDLi_CTRL.STOP request in order to avoid the Tx Descriptor Handler waiting indefinitely for the Write-back format
RDESE	1	rwh	**Rx descriptor error** This bit is also set by hardware under following scenarios: Error in Rx descriptor is indicated by bits RDESCi_WR3L.ES, RDESCi_WR3G.ES, and RDESCi_WR3G.ETLT. The software shall read the frame directly from the EIBUF and clear this flag. Software write with 1 clears this bit. This bit shall be cleared by the software only after clearing the corresponding TXi_REQ or FWDi_REQ
BF	2	rwh	**Buffer full** This bit is set by the hardware to indicate that the EIBUF is full. It is set by the DRE when the calculated size of the Ethernet frame from the NTSCF header or the packet length from the receive descriptor Write-back format RDESCi_WR3.PL exceeds the maxumum size of the EIBUF. The software shall read the frame directly from the EIBUF and clear this flag. Software write with 1 clears the bit. 0_B Free buffer 1_B EIBUF full

(table continues...)

(continued)

Field	Bits	Type	Description
DERRTYP	5:3	rh	**Descriptor error type** This bit-field indicates the reason for RDESE error. This is set by the hardware when RDESE error occurs and cleared by hardware when the software clears RDESE 000_B No error 001_B Rx Poll Demand error Bus error while Rx Tail Pointer register update 010_B Rx DMA state error The Rx DMA channel is in STOP state 011_B Incorrect OWN bit In case the OWN bit is not set while the Rx DMA channel is reading the Rx descriptor read format 100_B PTR mismatch read Mismatch between the Read format Rx descriptor pointed to by RETHDLi_CTRL.PTR and the descriptor being read by Rx DMA channel 101_B PTR mismatch Write-back Mismatch between the Rx descriptor pointed to by RETHDLi_CTRL.PTR and the descriptor being written back by Rx DMA channel 110_B Write-back error Write-back error in GETH Rx descriptor is indicated by bits RDESCi_WR3G.ES and LETH Rx descriptor is indicated by RDESCi_WR3L.ES. Write-back error also occurs when the Write-back format has the OWN bit set and LD bit not set 111_B Reserved
0	31:6	r	**Reserved** Read as 0; should be written with 0.

20.8.40 Ethernet input buffer i status

EIBUFi_STATUS (i=0-5) Offset address: $010E4_H+i*14_H$

Ethernet input buffer i status Kernel Reset value: $2003\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BPR	BPRC	RXREQ	0	MSFE			ACF_CAN_ADDR								
rh	rwh	rwh	r	rh			rh								

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FC	RFE	CFE	LME	IDID	IFT	FE	LE_ACF_CAN_ADDR								
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rh								

Field	Bits	Type	Description
LE_ACF_CAN_ADDR	8:0	rh	**Last Error ACF CAN Address Pointer** This bit-field indicates the latest ACF_CAN_ADDR (bits 10 to 2 of the 32-bit aligned address) of the ACF_CAN_BRIEF message which had either a Remote Frame Error (RFE) or a CAN Format Error (CFE). The value of this bit-field is valid only when either RFE or CFE error status bits are set. This bit-field is reset by the hardware once the BPR is set again for the next request
FE	9	rwh	**Input Ethernet Frame Error** This flag is set in case of Ethernet frame error and the frame cannot be processed by the ACF CAN-Ethernet Format Engine or the Forwarding engine. This bit-field is set when any of the IFT, IDID, CFE,LME, WDTE, RDESE or BF flags are set. Hardware sets the flag and software write with 1 clears the flag.
IFT	10	rwh	**Invalid Frame Type** This flag is set when the "Subtype" field of NTSCF header has a value other than 0x82. Hardware sets the flag and a software write with 1b clears the flag. In case the descriptor mode is used, this bit is set only when IDID is set to indicate the software the type of frame that caused the error
IDID	11	rwh	**Invalid destination ID** This bit-field is set when there is no matching Stream ID Filter element corresponding to the "Stream ID" of the input frame and there is also no matching FWDID Filter element. Hardware sets the flag and a software write with 1b clears the flag. Once set, the IDID status bit is sticky until SW clears it. The next incoming forwarding frame shall be forwarded irrespective of this bit being set unless there is again an IDID error for this frame.
LME	12	rwh	**AVTP Length Mismatch Error** This bit-field indicates if the calculated ACF_PAYLOAD_LENGTH is greater than the "Ntscf_data_length" value of the NTSCF header of input Ethernet frame. Hardware sets the flag and a software write with 1b clears the flag.
CFE	13	rwh	**CAN Format Error** This bit-field is set when the "ACF_MSG_TYPE" of the CAN frame contained in the input Ethernet frame has a value other than 0x2. Hardware sets the flag and a software write with 1b clears the flag.
RFE	14	rwh	**Remote Frame Error** This bit-field is set when EIBUF_CONFIG.RRF = 1 and a CAN frame has RTR = 1. Hardware sets the flag and a software write with 1b clears the flag.
FC	15	rwh	**Ethernet Frame Complete** This flag is set when the input Ethernet frame is processed by ACF CAN-Ethernet Format Engine that is when EIBUFi_STATUS.BPR bit-field is cleared by hardware. Software write with 1 clears the bit-field. Note: The SW should clear this bit when the RETHDLi_CTRL.TRIG mode is changed from interrupt to Rx descriptor mode

(table continues...)

(continued)

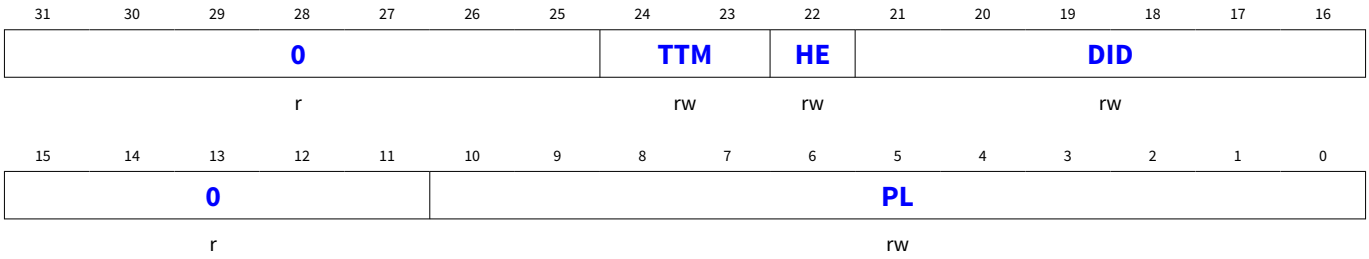
Field	Bits	Type	Description
ACF_CAN_ADDR	24:16	rh	**ACF CAN Address Pointer** This bit-field indicates the bits 10 to 2 of offset address from the NTSCF header start address of the Ethernet Input Buffer from which a new ACF_CAN_BRIEF message is to be read from. The bits 1 and 0 of the physical address to be considered as 0, as the address is always 32-bit aligned. The first ACF_CAN_BRIEF message is always read from offset of 0xC. The address is modified to the start of next ACF_CAN_BRIEF message after processing the fetch ACF_CAN_BRIEF message by the ACF CAN-Ethernet Format Engine. The bit-field is reset to 0x3 (32-bit aligned address representation of the offset address 0xC) when EIBUFi_STATUS.BPR is set again for the next request. BPR is cleared by hardware (when frame processing is completed) or by software setting BPRC = 1 (software abort).
MSFE	27:25	rh	**Matched Stream ID Filter Element** This bit-field indicates the index of the matched Stream-ID filter element. Valid match indicated only when IDID = 0. In case there is no Stream-ID match found, this bit-field is set to 7 and IDID is set to 1 if there is no forwarding filter match
RXREQ	29	rwh	**Ethernet receive request** This bit-field indicates a free Ethernet Input Buffer available to receive a Ethernet frame. 1_B - The bit-field is set by hardware and cleared by software with write of 1B after a frame is written to the EIBUF. The software to clear the bit after reset or after writing the Ethernet frame into the EIBUF. 0_B - A software write of 0B has no effect
BPRC	30	rwh	**Buffer Pending Request Clear** This bit-field allows software to clear the BPR of EIBUF. Software write with 1b clears the BPR bit-field of EIBUF. This bit-field is cleared at once the BPR bit-field is cleared by hardware. A software write with 0b has no effect.
BPR	31	rh	**Buffer Pending Request** This bit-field indicates a pending request for the EIBUF. The Ethernet Descriptor Handler writes 1b to indicate a new input Ethernet frame in EIBUF. The ACF CAN-Ethernet Format Engine clears the bit-field by hardware after completing extraction all CAN Frames from the input Ethernet frame. Software can also clear the flag (example: Software abort) by setting the BPRC bit-field to 1b.
0	28	r	**Reserved** Read as 0; Written as 0

20.8.41 Ethernet output buffer j configuration

EOBUFj_CONFIG (j=0-5) Offset address: 01150_H+j*38_H

Ethernet output buffer j configuration Kernel Reset value: $0032\ 0000_H$

20 Data Routing Engine (DRE)



Field	Bits	Type	Description
PL	10:0	rw	**ACF Payload Length** This bit-field configures the size in bytes of the ACF payload of Ethernet Output Buffer. Only the value of bits PL[10:2] is considered and PL[1:0] is always considered by hardware as 00b, so that the payload size is always an integral multiple of 32-bits. The minimum value is 8 bytes and maximum value can be 1484 bytes. 0 - Ethernet Output Buffer is disabled 4 - Considered as 8 bytes 8 to 1484 - Specifies the configured number of bytes (always a multiple of 4) >1484 - Considered as 1484 Note: When PL is not configured according to the Transmit trigger mode TTM and configuration TTC, the Buffer full flag BF is set when the EOBUF cannot accomodate the incoming CAN frame. The PL shall be configured such that there is at least a single ACF_CAN_BRIEF message within the EOBUF. There is no TXREQ triggered if the EOBUF is empty
DID	21:16	rw	**Destination ID of the Buffer** This bit-field assigns a routing destination ID for the Ethernet Output Buffer. The input CAN frames with a matching destination ID with this bit-field value are assembled to the Ethernet Output Buffer. Valid values are from 0x18 till 0x1D. Other values are reserved. Only consecutive EOBUFs can be configured with similar DIDs. For e.g.: EOBUF0, EOBUF1 can have DID=18H. But if the configurations of DIDs is EOBUF0=18H, EOBUF1=19H and EOBUF2=18H, then it is incorrect.
HE	22	rw	**Header Enable** This bit-field enables the Ethernet header content of the buffer. 0_B Ethernet header insertion is disabled 1_B Ethernet header insertion is enabled

(table continues...)

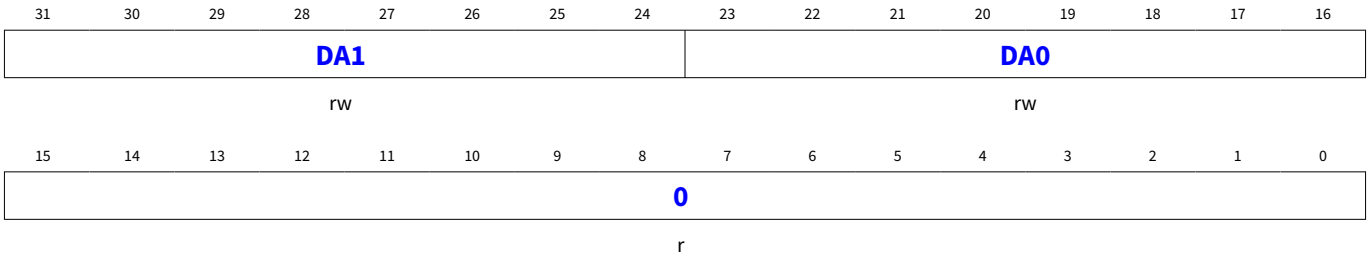
(continued)

Field	Bits	Type	Description
TTM	24:23	rw	**Transmit Trigger Mode** This bit-field configures the AVTP frame transmit trigger modes. Note : When Buffer full flag BF is set, the DRE forces a SW transmit trigger mode for that EOBUF irrespective of the trigger mode configured. The software shall set the TXRDY flag based on the Buffer full interrupt. When Buffer full flag BF is set in descriptor mode, the Tx descriptors are prepared and there is no transmit trigger generated In case of SW trigger mode, the TXRDY bit should not be set when the TXREQ bit or TXi_REQ bit is already set 00_B **SWTM**: Software Triggered Transmit Mode Transmit trigger is initiated when software sets EOBUFj_STATUS.TXRDY. 01_B **FCTM**: Frame Count Transmit Mode The transmit trigger is based on configured number of CAN frames (EOBUFj_TTC.TFL) in a transmit Ethernet frame. 10_B **BFTM**: Buffer Fill Level Transmit Mode The transmit trigger condition is based on the fill size of the Ethernet Output Buffer as configured in EOBUFj_TTC.BUFT. 11_B **TTTM**: Time Triggered Transmit Mode The transmit trigger condition is based on a timer as configured in EOBUFj_TTC and EOBUFj_TTS.
0	15:11, 31:25	r	**Reserved** Read as 0; should be written with 0.

20.8.42 Ethernet output buffer j MAC header 0

EOBUFj_MAC_H0 (j=0-5) Offset address: 01154_H+j*38_H

Ethernet output buffer j MAC header 0 Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
DA0	23:16	rw	**MAC Destination Address 0** This bit-field corresponds to bits 47 to 40 of the MAC destination address of the transmit Ethernet frame.
DA1	31:24	rw	**MAC Destination Address 1** This bit-field corresponds to bits 39 to 32 of the MAC destination address of the transmit Ethernet frame.
0	15:0	r	**Reserved** Read as 0; should be written with 0.

20.8.43 Ethernet output buffer j MAC header 1

EOBUFj_MAC_H1 (j=0-5) Offset address: 01158_H+j*38_H

Ethernet output buffer j MAC header 1 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DA5								DA4							
rw								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DA3								DA2							
rw								rw							

Field	Bits	Type	Description
DA2	7:0	rw	**MAC Destination Address 2** This bit-field corresponds to bits 31 to 24 of the MAC destination address of the transmit Ethernet frame.
DA3	15:8	rw	**MAC Destination Address 3** This bit-field corresponds to bits 23 to 16 of the MAC destination address of the transmit Ethernet frame.
DA4	23:16	rw	**MAC Destination Address 4** This bit-field corresponds to bits 15 to 8 of the MAC destination address of the transmit Ethernet frame.
DA5	31:24	rw	**MAC Destination Address 5** This bit-field corresponds to bits 7 to 0 of the MAC destination address of the transmit Ethernet frame.

20.8.44 Ethernet output buffer j MAC header 2

EOBUFj_MAC_H2 (j=0-5) Offset address: $0115C_H+j*38_H$

Ethernet output buffer j MAC header 2 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SA3								SA2							
rw								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SA1								SA0							
rw								rw							

Field	Bits	Type	Description
SA0	7:0	rw	**MAC Source Address** This bit-field corresponds to bits 47 to 40 of the MAC source address of the transmit Ethernet frame.

(table continues...)

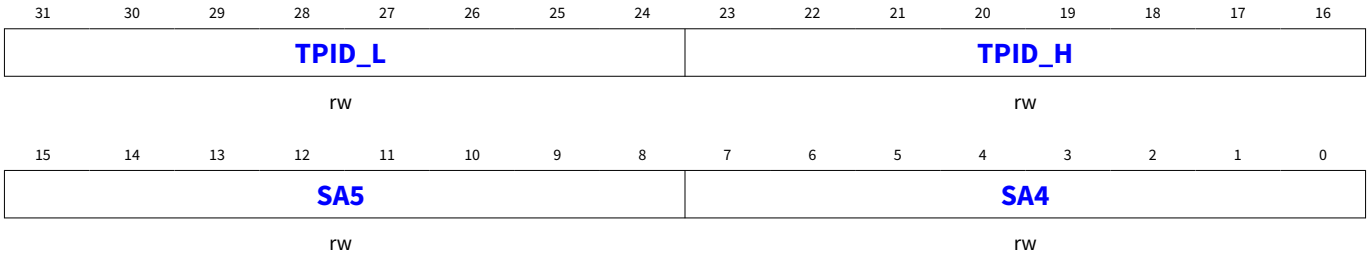
(continued)

Field	Bits	Type	Description
SA1	15:8	rw	**MAC Source Address** This bit-field corresponds to bits 39 to 32 of the MAC source address of the transmit Ethernet frame.
SA2	23:16	rw	**MAC Source Address** This bit-field corresponds to bits 31 to 24 of the MAC source address of the transmit Ethernet frame.
SA3	31:24	rw	**MAC Source Address** This bit-field corresponds to bits 23 to 16 of the MAC source address of the transmit Ethernet frame.

20.8.45 Ethernet output buffer j MAC header 3

EOBUFj_MAC_H3 (j=0-5) Offset address: 01160_H+j*38_H

Ethernet output buffer j MAC header 3 Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
SA4	7:0	rw	**MAC Source Address** This bit-field corresponds to bits 15 to 8 of the MAC source address of the transmit Ethernet frame.
SA5	15:8	rw	**MAC Source Address** This bit-field corresponds to bits 7 to 0 of the MAC source address of the transmit Ethernet frame.
TPID_H	23:16	rw	**Tagged Protocol Identifier High** This bit-field represents the MSB of TPID field. The TPID field is configured to a value of 8100h for tagged Ethernet frames, as required by IEEE Std. 802.1Q
TPID_L	31:24	rw	**Tagged Protocol Identifier Low** This bit-field represents the LSB of TPID field. The TPID field is configured to a value of 8100h for tagged Ethernet frames, as required by IEEE Std. 802.1Q

20.8.46 Ethernet output buffer j MAC header 4

EOBUFj_MAC_H4 (j=0-5) Offset address: 01164_H+j*38_H

Ethernet output buffer j MAC header 4 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AVTPET_L								AVTPET_H							
rw								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VTAG_L								VTAG_H							
rw								rw							

Field	Bits	Type	Description
VTAG_H	7:0	rw	**VLAN Tag High** This bit-field is configured by user with MSB of VLAN tag for the transmit Ethernet frame as specified by IEEE Std 802.1Q. The VLAN tag contains PCP (3 bits), DEI (1 bit) & VID (12 bits).
VTAG_L	15:8	rw	**VLAN Tag Low** This bit-field is configured by user with LSB of VLAN tag for the transmit Ethernet frame as specified by IEEE Std 802.1Q. The VLAN tag contains PCP (3 bits), DEI (1 bit) & VID (12 bits).
AVTPET_H	23:16	rw	**AVTP Ethertype High** This bit-field contains the MSB of AVTPET field. This AVTPET field is used to identify the transmit Ethernet frame as AVTP frames. The corresponding Ethertype for AVTP frame is 0x22F0
AVTPET_L	31:24	rw	**AVTP Ethertype Low** This bit-field contains the LSB of AVTPET field. This AVTPET field is used to identify the transmit Ethernet frame as AVTP frames. The corresponding Ethertype for AVTP frame is 0x22F0

20.8.47 Ethernet output buffer j NTSCF header

EOBUFj_NTSCF_H0 (j=0-5) Offset address: 01168_H+j*38_H

Ethernet output buffer j NTSCF header Kernel Reset value: $0000\ 0082_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SN								NTSCFDL_L							
rwh								rh							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SV	0				NTSCFDL_H			SUBTYP							
rw	r				rh			r							

Field	Bits	Type	Description
SUBTYP	7:0	r	**Subtype of AVTP Frame** This bit-field is used to identify the format being carried by AVTP frame. Non-Time Synchronous Control frames (NTSCF) has subtype value of 0x82.

(table continues...)

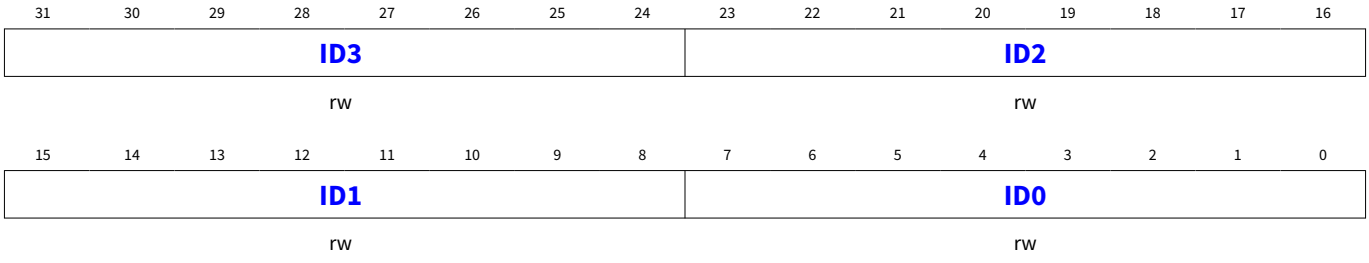
(continued)

Field	Bits	Type	Description
NTSCFDL_H	10:8	rh	**NTSCF Data Length High** This bit-field contains bits 10 to 8 of NTSCF data length field of the ACF payload data in bytes. The ACF CAN-Ethernet Format Engine updates the length on every ACF_CAN_BRIEF message written to the ACF payload section of Ethernet Output Buffer. It is reset to 0 when EOBUFj_STATUS.TXREQ is cleared to 0.
SV	15	rw	**Stream ID Valid** This bit-field indicates whether the STREAM_ID field contains an ID for a stream with a valid stream reservation.
NTSCFDL_L	23:16	rh	**NTSCF Data Length Low** This bit-field contains bits 7 to 0 of NTSCF data length field of the ACF payload data in bytes. The ACF CAN-Ethernet Format Engine updates the length on every ACF_CAN_BRIEF message written to the ACF payload section of Ethernet Output Buffer.
SN	31:24	rwh	**Sequence Number** This bit-field indicates the sequence of AVTP frames in a stream. Software can configure an initial value for a stream and the sequence number if incremented by the ACF CAN-Ethernet Format engine for every new transmit AVTP frame. Upon overflow, the value is wrapped around to 0x00 by hardware.
0	14:11	r	**Reserved** Read as 0; should be written with 0.

20.8.48 Ethernet output buffer j Stream ID configuration 0

EOBUFj_NTSCF_STREAM0_ID (j=0-5) Offset address: $0116C_H+j*38_H$

Ethernet output buffer j Stream ID configuration 0 Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
ID0	7:0	rw	**Stream ID** This bit-field defines bits 63 to 56 of the identifier for stream identification. User must configure a valid Stream ID as defined in subclause 35.2.2.8.2 of IEEE Std 802.1Q - 2014.
ID1	15:8	rw	**Stream ID** This bit-field defines bits 55 to 48 of the identifier for stream identification. User must configure a valid Stream ID as defined in subclause 35.2.2.8.2 of IEEE Std 802.1Q - 2014.

(table continues...)

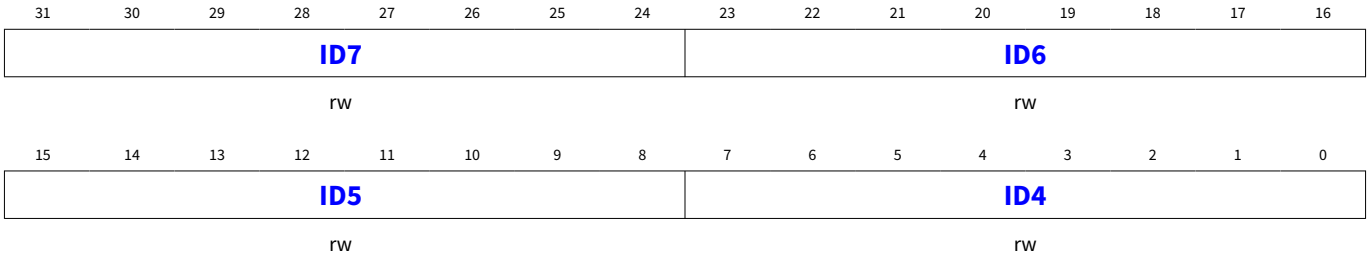
(continued)

Field	Bits	Type	Description
ID2	23:16	rw	**Stream ID** This bit-field defines bits 47 to 40 of the identifier for stream identification. User must configure a valid Stream ID as defined in subclause 35.2.2.8.2 of IEEE Std 802.1Q - 2014.
ID3	31:24	rw	**Stream ID** This bit-field defines bits 39 to 32 of the identifier for stream identification. User must configure a valid Stream ID as defined in subclause 35.2.2.8.2 of IEEE Std 802.1Q - 2014.

20.8.49 Ethernet output buffer j Stream ID configuration 1

EOBUFj_NTSCF_STREAM1_ID (j=0-5) Offset address: 01170_H+j*38_H

Ethernet output buffer j Stream ID configuration 1 Kernel Reset value: $0000\ 0000_H$

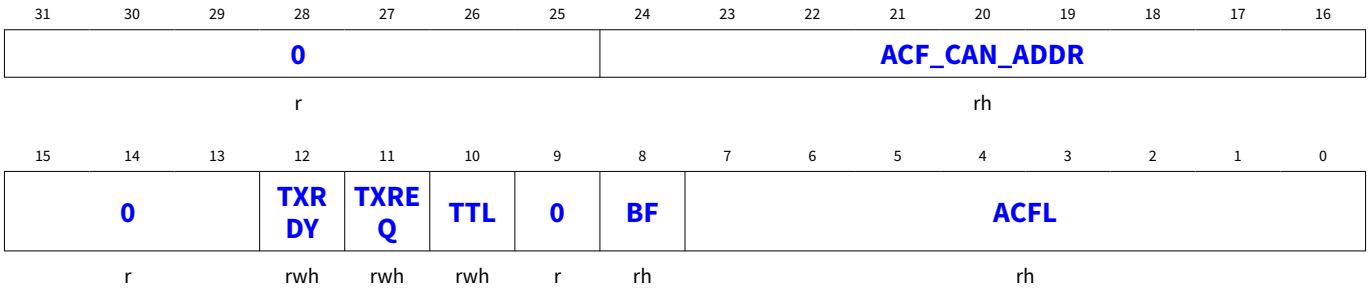


Field	Bits	Type	Description
ID4	7:0	rw	**Stream ID** This bit-field defines bits 31 to 24 of the identifier for stream identification. User must configure a valid Stream ID as defined in subclause 35.2.2.8.2 of IEEE Std 802.1Q - 2014.
ID5	15:8	rw	**Stream ID** This bit-field defines bits 23 to 16 of the identifier for stream identification. User must configure a valid Stream ID as defined in subclause 35.2.2.8.2 of IEEE Std 802.1Q - 2014.
ID6	23:16	rw	**Stream ID** This bit-field defines bits 15 to 8 of the identifier for stream identification. User must configure a valid Stream ID as defined in subclause 35.2.2.8.2 of IEEE Std 802.1Q - 2014.
ID7	31:24	rw	**Stream ID** This bit-field defines bits 7 to 0 of the identifier for stream identification. User must configure a valid Stream ID as defined in subclause 35.2.2.8.2 of IEEE Std 802.1Q - 2014.

20.8.50 Ethernet output buffer j status

EOBUFj_STATUS (j=0-5) Offset address: 01174_H+j*38_H

Ethernet output buffer j status Kernel Reset value: $0008\ 0000_H$



Field	Bits	Type	Description
ACFL	7:0	rh	**ACF CAN Fill Level** This bit-field indicates the number of CAN frames assembled in the Ethernet Output Buffer. The field is reset to 0x0 when EOBUFj_STATUS.TXREQ is cleared to 0.
BF	8	rh	**Buffer Full** This flag is set the EOBUF cannot accommodate the currently processed CAN frame. This bit shall only be cleared by hardware after TXREQ (in case of interrupt trigger) is cleared by software. In case the descriptor mode is used, this bit shall be cleared by hardware simultaneously while clearing EREQ.TXi_REQ. Note : The EOBUFj_CONFIG.PL needs to be configured according the Transmit trigger mode TTM and configuration TTC 0_B Free buffer 1_B EOBUF full
TTL	10	rwh	**Transmit Trigger Lost** This status bit is set by hardware when a transmit trigger condition is skipped as the corresponding Ethernet Output Buffer is empty (ACFL = 0). A software write with 1b clears the bit-field. A write with 0b has no effect.
TXREQ	11	rwh	**Ethernet Output Buffer Transmit Request** This bit-field indicates a valid transmit Ethernet frame in the Ethernet Output Buffer. The bit-field is set by hardware and cleared by software with write of 1b. A software write of 0b has no effect.
TXRDY	12	rwh	**Ethernet Output Buffer Transmit Ready** This bit-field is set by hardware when an Ethernet Output Buffer has a transmit trigger as configured in EOBUFj_CONFIG.TTM. When TTM = 0, the software write of 1b to this bit-field marks the transmit trgger condition. The flag is cleared by hardware, when the transmit Ethernet frame is prepared. A software write of 0b has no effect.
ACF_CAN_ADDR	24:16	rh	**ACF CAN Address Offset** This bit-field indicates the bits 10 to 2 of offset address from the start address of the Ethernet Output Buffer at which a new ACF_CAN_BRIEF message is to be stored. The bits 1 and 0 of the physical address to be considered as 0, as the address is always 32-bit aligned. The first ACF_CAN_BRIEF message is always stored at the offset address of 0x20. This field is reset to 0x8 (32-bit aligned representation of the offset address 0x20) when EOBUFj_STATUS.TXREQ is cleared to 0.

(table continues...)

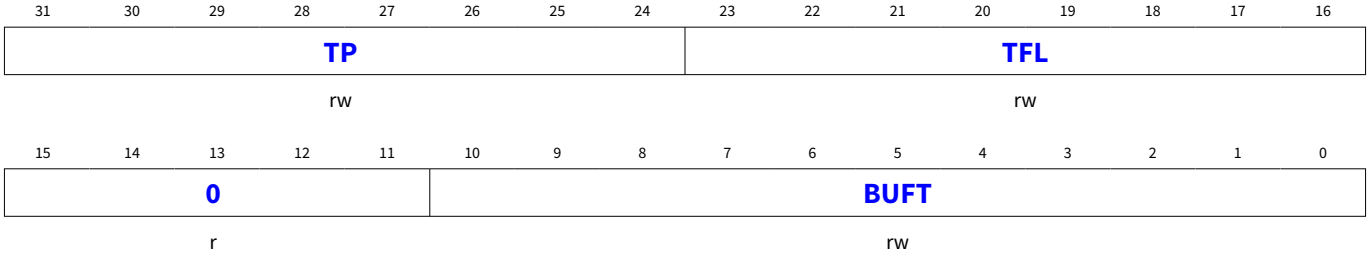
(continued)

Field	Bits	Type	Description
0	9, 15:13, 31:25	r	**Reserved** Read as 0. should be written with 0.

20.8.51 Ethernet output buffer j Transmit trigger configuration

EOBUFj_TTC (j=0-5) Offset address: 01178_H+j*38_H

Ethernet output buffer j Transmit trigger configuration Kernel Reset value: $0000\ 0000_H$

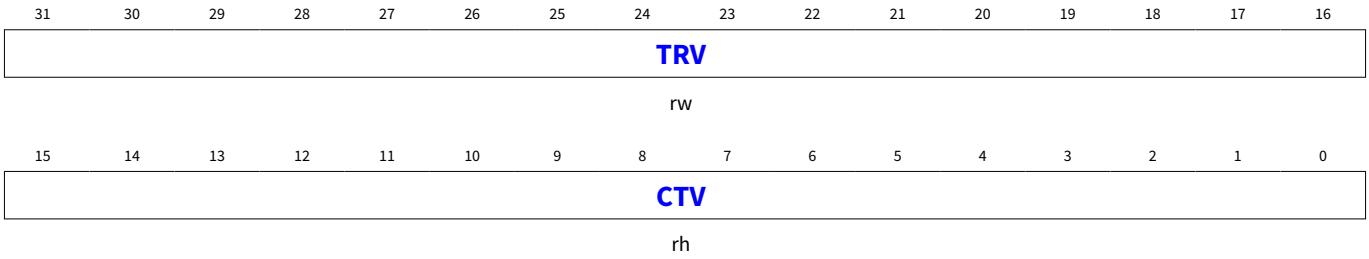


Field	Bits	Type	Description
BUFT	10:0	rw	**Buffer Threshold** When EOBUFj_CONFIG.TTM = 0x2, this bit-field configures the threshold size of the Ethernet Output Buffer ACF payload for transmit trigger. A valid value of this bit-field is between 0x1 till the maximum size of the Ethernet Output Buffer ACF payload as configured in EOBUFj_CONFIG.PL
TFL	23:16	rw	**Trigger Fill Level** When EOBUFj_CONFIG.TTM = 0x1, this bit-field defines the threshold for number of CAN frames in an Ethernet Output Buffer for transmit trigger condition. This bit-field cannot be zero. Note: This value has to be set based on the size of the incoming CAN frames and the size of the EOBUF. If this value is set too high, there might not be a transmit trigger
TP	31:24	rw	**Pre-scale for Timer** This bit-field defines the divider factor of the clock frequency used for transmit trigger. The prescaler used is (TP+1).
0	15:11	r	**Reserved** Read as 0; should be written with 0.

20.8.52 Ethernet output buffer j Timer threshold and status

EOBUFj_TTS (j=0-5) Offset address: $0117C_H+j*38_H$

Ethernet output buffer j Timer threshold and status Kernel Reset value: $0000\ 0000_H$

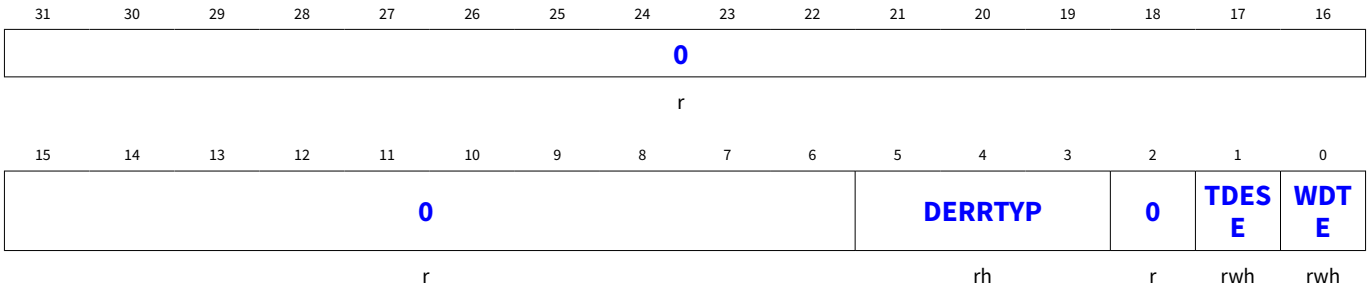


Field	Bits	Type	Description
CTV	15:0	rh	**Current Timer Value** This bit-field shows the current timer value. It is reloaded with the value of TRV upon a non-zero write to TRV bit-field and is decremented at the timer frequency. Transmit trigger condition occurs when it reaches 0. The timer is halted when TRV is written with 0.
TRV	31:16	rw	**Timer Reload Value** This bit-field defines the reload value of the transmit timers. It is configured by the software. A non-zero TRV value indicates that the timer is running. When TRV transitions from zero to non-zero value, the timer is started. The first transmit trigger is generated immediately. All subsequent transmit triggers would be at the intervals of the value configured. When the timer is already running and the SW changes the TRV to a different non-zero value, the timer automatically adjusts the time period without generating any surplus transmit trigger

20.8.53 Ethernet output buffer j error

EOBUFj_ERROR (j=0-5) Offset address: 01180_H+j*38_H

Ethernet output buffer j error Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
WDTE	0	rwh	**Ethernet watchdog timeout error** This bit is set by the hardware to indicate an Ethernet Watchdog timeout error in case of delayed availability of Ethernet frame to transmit within the EOBUF and delayed read of the Ethernet frame from EOBUF. Software write with 1 clears this bit

(table continues...)

(continued)

Field	Bits	Type	Description
TDESE	1	rwh	**Tx descriptor error** This bit is also set by hardware under following scenarios: Error in Tx descriptor is indicated by bit TDESCi_WR3G.DERR or TDESCi_WR3L.DE. Software write with 1 clears this bit. This bit shall be cleared by software only after clearing the corresponding TXi_REQ or FWDi_REQ
DERRTYP	5:3	rh	**Descriptor error type** This bit-field indicates the reason for TDESE error. This is set by the hardware when TDESE error occurs and cleared by hardware when the software clears TDESE 000_B No error 001_B Tx Poll Demand error Bus error during the Tx Tail Pointer register update 010_B Tx DMA state error The Tx DMA channel is in STOP state 011_B Incorrect OWN bit In case the OWN bit is not set while the Tx DMA channel is reading the Tx descriptor read format 100_B PTR mismatch read Mismatch between the Read format Tx descriptor pointed to by TETHDLi_CTRL.PTR and the descriptor being read by Tx DMA channel 101_B PTR mismatch Write-back Mismatch between the Tx descriptor pointed to by TETHDLi_CTRL.PTR and the descriptor being written back by Tx DMA channel 110_B Write-back error Write-back error in GETH Tx descriptor is indicated by bits TDESCi_WR3G.DERR and LETH Tx descriptor is indicated by TDESCi_WR3L.DE. Write-back error also occurs when the Write-back format has the OWN bit set and LD bit not set 111_B Reserved
0	2, 31:6	r	**Reserved** Read as 0; should be written with 0.

20.8.54 Stream ID filter i configuration

SIDFi_FC (i=0-7) Offset address: $0129C_H+i*14_H$

Stream ID filter i configuration Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0									RTI			0	MODE		FE
r									rw			r	rw		rw

Field	Bits	Type	Description
FE	0	rw	**Filter Enable** This bit-field indicates whether the filter element is enabled (FE=1) or disabled (FE=0).
MODE	2:1	rw	**Filter Mode** This bit-field defines the mode of Stream ID filters 00_B Classical Filter Stream Filter 1 is used as Filter element and Stream Filter 2 is used as mask. 01_B Range Filter The range of values between Stream Filter 1 and Stream Filter 2 are used for filtering. 10_B Reserved Reserved; not to be used 11_B Reserved Reserved; not to be used
RTI	6:4	rw	**Routing Table Index** This bit filed indicates the Routing Table index which will be used by the CAN Transmit Routing Engine for the CAN frames within the Ethernet frame. Valid values are 0 to 3.
0	3, 31:7	r	**Reserved** Read as 0; should be written with 0.

20.8.55 Stream ID filter i configuration Stream ID filter 1 lower

SIDFi_FIL1_L (i=0-7) Offset address: $012A0_H+i*14_H$

Stream ID filter i configuration Stream ID filter 1 lower Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
F1L															
rw															

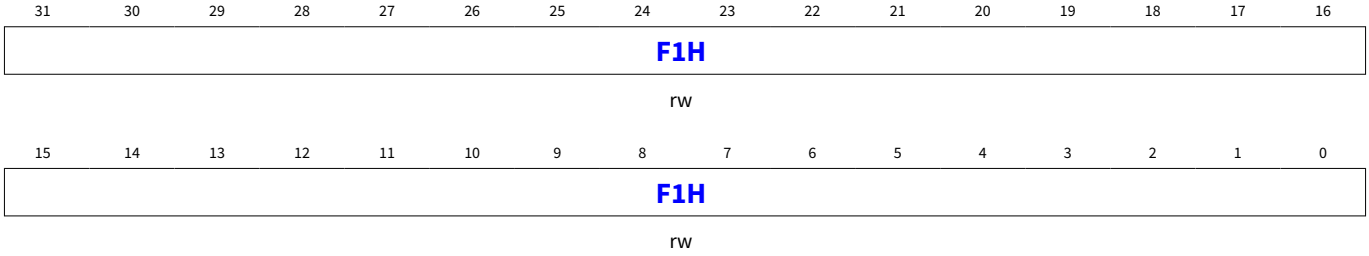
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
F1L															
rw															

Field	Bits	Type	Description
F1L	31:0	rw	**Lower Filter 1 Stream ID** This bit-field defines the Filter 1 for Stream IDs

20.8.56 Stream ID filter i configuration Stream ID filter 1 higher

SIDFi_FIL1_H (i=0-7) Offset address: $012A4_H+i*14_H$

Stream ID filter i configuration Stream ID filter 1 higher Kernel Reset value: $0000\ 0000_H$

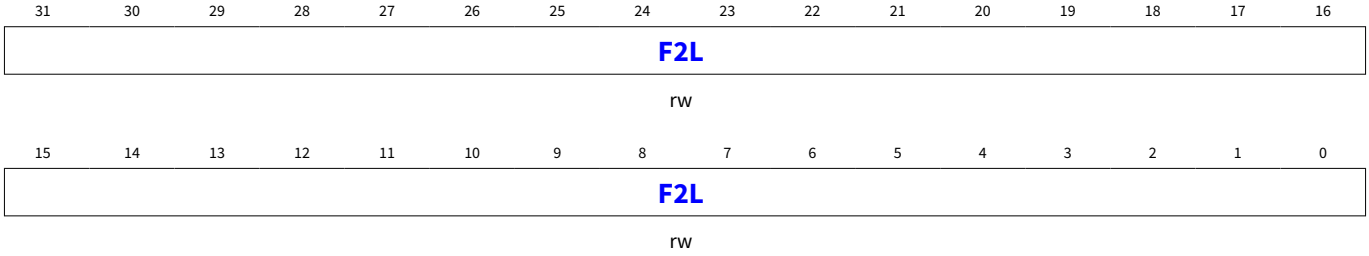


Field	Bits	Type	Description
F1H	31:0	rw	**Higher Filter 1 Stream ID** This bit-field defines the Filter 1 for Stream IDs

20.8.57 Stream ID filter i configuration Stream ID filter 2 lower

SIDFi_FIL2_L (i=0-7) Offset address: $012A8_H+i*14_H$

Stream ID filter i configuration Stream ID filter 2 lower Kernel Reset value: $0000\ 0000_H$

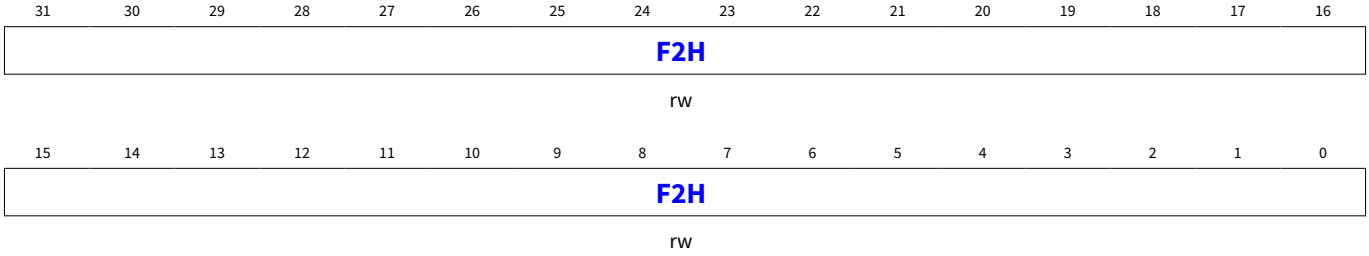


Field	Bits	Type	Description
F2L	31:0	rw	**Lower Filter 2 Stream ID** This bit-field defines the Filter 2 for Stream IDs

20.8.58 Stream ID filter i configuration Stream ID filter 2 higher

SIDFi_FIL2_H (i=0-7) Offset address: $012AC_H+i*14_H$

Stream ID filter i configuration Stream ID filter 2 higher Kernel Reset value: $0000\ 0000_H$

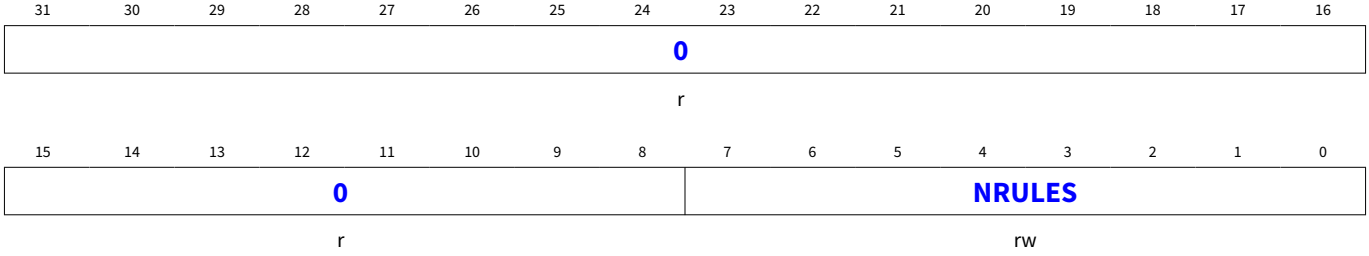


Field	Bits	Type	Description
F2H	31:0	rw	**Higher Filter 2 Stream ID** This bit-field defines the Filter 2 for Stream IDs

20.8.59 CAN transmit routing table i configuration

RTi_CONFIG (i=0-3) Offset address: 01340_H+i*8

CAN transmit routing table i configuration Kernel Reset value: $0000\ 0000_H$

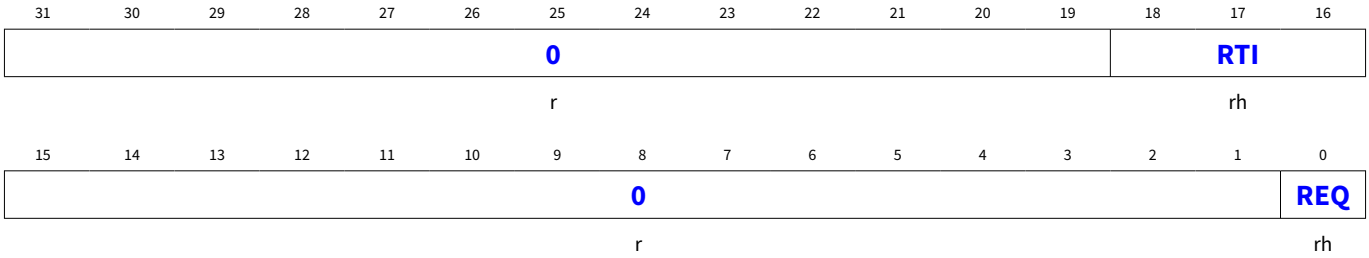


Field	Bits	Type	Description
NRULES	7:0	rw	**Number of routing rules** This bit-field configures the number of routing rules contained in the CAN Transmit Routing Table (RTi). Valid values are from 0 to 128. 0 - CAN Transmit Routing Table is disabled 1 to 128 - Number of routing rules in the CAN Transmit Routing Table >128 - Invalid configuration
0	31:8	r	**Reserved** Read as 0; should be written with 0.

20.8.60 Routing request configuration

RREQ_CONFIG Offset address: $0135C_H$

Routing request configuration Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
REQ	0	rh	**New Routing Request** This bit-field indicates a new routing request for CAN Transmit Routing Engine.
RTI	18:16	rh	**Routing Table Index** This bit-field indicates the CAN Transmit Routing Table to be used for the routing search done by the CAN Transmit Routing Engine. Valid values are 0 to 3.
0	15:1, 31:19	r	**Reserved** Read as 0; should be written with 0.

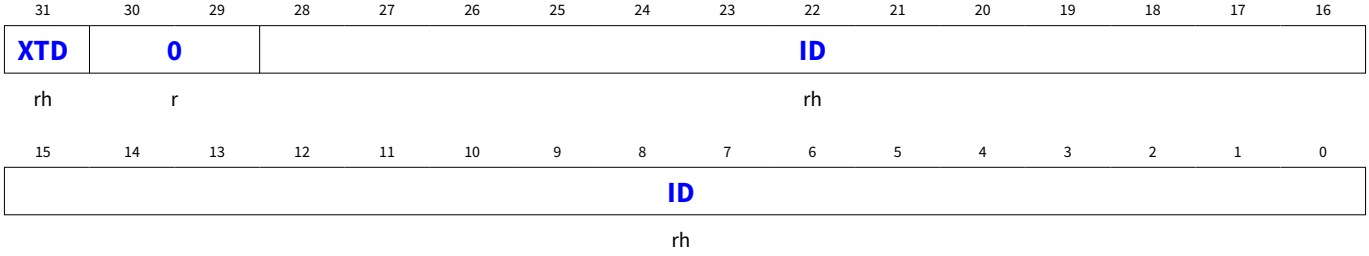
20.8.61 CAN ID request

RREQ_CID

CAN ID request

Offset address: 01360_H

Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
ID	28:0	rh	**CAN Identifier** This bit-field indicates the CAN ID for which the routing request is initiated.
XTD	31	rh	**Extended Identifier** This bit-field indicates the ID format of the CAN ID. 0_B Standard 11-bit ID 1_B Extended 29-bit CAN ID
0	30:29	r	**Reserved** Read as 0; should be written with 0.

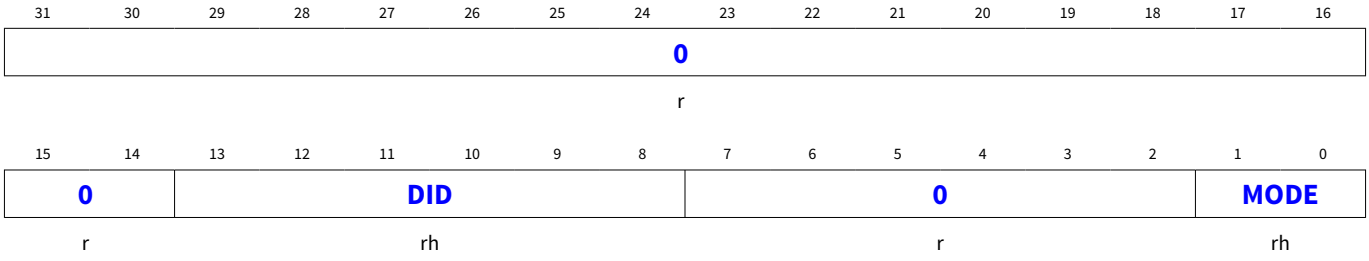
20.8.62 Uni-cast routing header

UCRH

Uni-cast routing header

Offset address: 01364_H

Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
MODE	1:0	rh	**Routing Rule Mode** It defines the type of the routing rule 0 - Uni-cast rule 1 - Multi-cast rule other - Reserved, considered as Uni-cast rule

(table continues...)

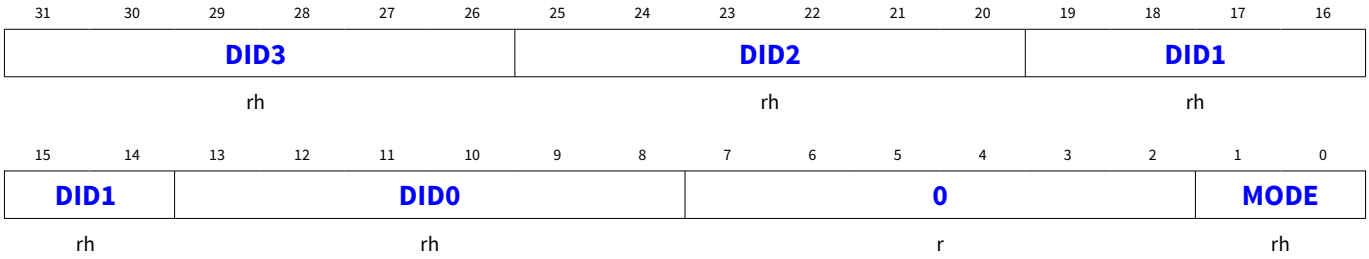
(continued)

Field	Bits	Type	Description
DID	13:8	rh	**Destination ID** This bit-field indicates the destination CAN node to which the received CAN frame has to be transferred. The destination CAN node is referenced using a unique ID as given in the Routing destination ID table.
0	7:2, 31:14	r	**Reserved** Read as 0; should be written with 0.

20.8.63 Multi-cast routing header

MCRH Offset address: 01364_H

Multi-cast routing header Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
MODE	1:0	rh	**Routing Rule Mode** It defines the type of the routing rule 0 - Uni-cast rule 1 - Multi-cast rule other - Reserved, considered as Uni-cast rule
DID0	13:8	rh	**Destination ID 1** This bit-field indicates the 1st destination CAN node to which the received CAN frame has to be transferred. The destination CAN node is referenced using a unique ID as given in the Routing destination ID table.
DID1	19:14	rh	**Destination ID 2** This bit-field indicates the 2nd destination CAN node to which the received CAN frame has to be transferred. The destination CAN node is referenced using a unique ID as given in the Routing destination ID table.
DID2	25:20	rh	**Destination ID 3** This bit-field indicates the 3rd destination CAN node to which the received CAN frame has to be transferred. The destination CAN node is referenced using a unique ID as given in the Routing destination ID table.

(table continues...)

(continued)

Field	Bits	Type	Description
DID3	31:26	rh	**Destination ID 4** This bit-field indicates the 4th destination CAN node to which the received CAN frame has to be transferred. The destination CAN node is referenced using a unique ID as given in the Routing destination ID table.
0	7:2	r	**Reserved** Read as 0; should be written with 0.

20.8.64 Routing status

RS Offset address: 01368_H

Routing status Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0														NMFE	IRT
r														rwh	rwh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0							RE								
r							rh								

Field	Bits	Type	Description
RE	8:0	rh	**Routing Element** This bit-field indicates Routing Element index which matches the CAN ID of the routing request. This bit-field is reset by the hardware once the BPR bit is cleared. This bit-field is not valid in case of IRT or NMFE error
IRT	16	rwh	**Invalid Routing Table** This bit-field is set when the CAN Transmit Routing Table indexed in the routing request is either disabled or invalid. Hardware sets the flag and a software write with 1b clears the flag.
NMFE	17	rwh	**Non-Matching Filter Element Error** This bit-field is set when there are no matching filter elements found for the corresponding routing request. Hardware sets the flag and a software write with 1b clears the flag.
0	15:9, 31:18	r	**Reserved** Read as 0; should be written with 0.

20.8.65 CAN receive request 0

This register indicates Receive request from CAN interfaces

CANRXR0 Offset address: $0136C_H$

CAN receive request 0 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
C15_RH1R	C15_RH0R	C14_RH1R	C14_RH0R	C13_RH1R	C13_RH0R	C12_RH1R	C12_RH0R	C11_RH1R	C11_RH0R	C10_RH1R	C10_RH0R	C9_RH1R	C9_RH0R	C8_RH1R	C8_RH0R
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
C7_RH1R	C7_RH0R	C6_RH1R	C6_RH0R	C5_RH1R	C5_RH0R	C4_RH1R	C4_RH0R	C3_RH1R	C3_RH0R	C2_RH1R	C2_RH0R	C1_RH1R	C1_RH0R	C0_RH1R	C0_RH0R
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
Ci_RH0R (i=0-15)	2*i	rh	**CANi Receive Host Buffer 0 Pending Request** This bit-field indicates a pending receive request from CANi interface Receive Host Buffer 0
Ci_RH1R (i=0-15)	2*i+1	rh	**CANi Receive Host Buffer 1 Pending Request** This bit-field indicates a pending receive request from CANi interface Receive Host Buffer 1

20.8.66 CAN receive request 1

This register indicates Receive request from CAN interfaces

CANRXR1 Offset address: 01370_H

CAN receive request 1 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								C19_RH1R	C19_RH0R	C18_RH1R	C18_RH0R	C17_RH1R	C17_RH0R	C16_RH1R	C16_RH0R
r								rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
Ci_RH0R (i=16-19)	2*i-32	rh	**CANi Receive Host Buffer 0 Pending Request** This bit-field indicates a pending receive request from CANi interface Receive Host Buffer 0
Ci_RH1R (i=16-19)	2*i-31	rh	**CANi Receive Host Buffer 1 Pending Request** This bit-field indicates a pending receive request from CANi interface Receive Host Buffer 1
0	31:8	r	**Reserved** Read as 0; should be written with 0.

20.8.67 CAN transmit buffer available request

This register indicates Transmit Host Buffer 0 available request from CAN interfaces

CANTXR Offset address: 01374_H

CAN transmit buffer available request Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0												C19_THR	C18_THR	C17_THR	C16_THR
r												rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
C15_THR	C14_THR	C13_THR	C12_THR	C11_THR	C10_THR	C9_THR	C8_THR	C7_THR	C6_THR	C5_THR	C4_THR	C3_THR	C2_THR	C1_THR	C0_THR
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
Ci_THR (i=0-19)	i	rh	**CANi Transmit Host Buffer 0 Request** This bit-field indicates a free transmit buffer available in CANi interface
0	31:20	r	**Reserved** Read as 0; should be written with 0.

20.8.68 Destination memory i configuration

DMEMi_CONFIG (i=0-27) Offset address: $0137C_H+i*20_H$

Destination memory i configuration Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															CTYP
r															rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OA								INP				ATH	AST	EN2	EN1
rw								rw				rw	rw	rwh	rwh

Field	Bits	Type	Description
EN1	0	rwh	**Enable Memory Destination 1** This bit-field is used to enable or disable the corresponding memory destination by the software. The software sets this bit by writing 1 to indicate that the buffer is ready for the CAN frame or CAN I-PDU to be stored. Software write with 0 has no effect. The hardware clears this bit after the CAN frames or CAN I-PDUs are assembled till the waternark level. 0_B DMEM not available 1_B DMEM available

(table continues...)

(continued)

Field	Bits	Type	Description
EN2	1	rwh	**Enable Memory Destination 2** This bit-field is used to enable or disable the corresponding memory destination by the software. The software sets this bit by writing 1 to indicate that the buffer is ready for the CAN frame or CAN I-PDU to be stored. Software write with 0 has no effect. The hardware clears this bit after the CAN frames or CAN I-PDUs are assembled till the wraparound level. 0_B DMEM not available 1_B DMEM available
AST	2	rw	**Append Status Information** When set to 1, the DMEMi_STATUS register content is appended along with the CAN message in the memory destination 0_B **DIS**: DMEMi_STATUS not appended 1_B **EN**: DMEMi_STATUS appended
ATH	3	rw	**Append Timing Header** When set to 1, the CAN timing header is appended to the CAN message at the memory destination 0_B **DIS**: THEAD not appended 1_B **EN**: THEAD appended
INP	7:4	rw	**Interrupt Node Pointer** This bit-field configures the interrupt line to be triggered in case of a watermark or address wraparound event. Valid values are 0 till 7. Other values are reserved and will be considered as 0.
OA	15:8	rw	**Offset Address** The offset address by which the destination address is incremented after every routing operation to the destination memory. It defines the size of each buffer at the destination memory. OA[0:2] is considered always as 000b so that the calculated absolute destination address is always 64-bit aligned. In case of CAN I-PDU routing, the OA is considered for the count based trigger mode to calculate the size of the frame data buffers 1 and 2. The OA should be configured by the user based upon the number of I-PDUs (DMEMi_WM and DMEMi_WA levels) and the size of the I-PDUs that are to be assembled within the DMEMi. The OA value should also include the Timing header (optional) and I-PDU header. It is illegal to configure the OA smaller than the combined sizes of the incoming CAN messages Note: Configuring the OA for Classical CAN I-PDU (Timing header (optional) + PDU header + Payload), but incoming CAN FD I-PDUs into the DMEMi would lead to buffer overflow as the size would be too small Note: Avoid setting OA as 0x20 for CAN frames with larger payload of more than 8 bytes. Such frames are rejected and an error is triggered but the status flags WAF, WMF and BC are updated even though the frame is rejected

(table continues...)

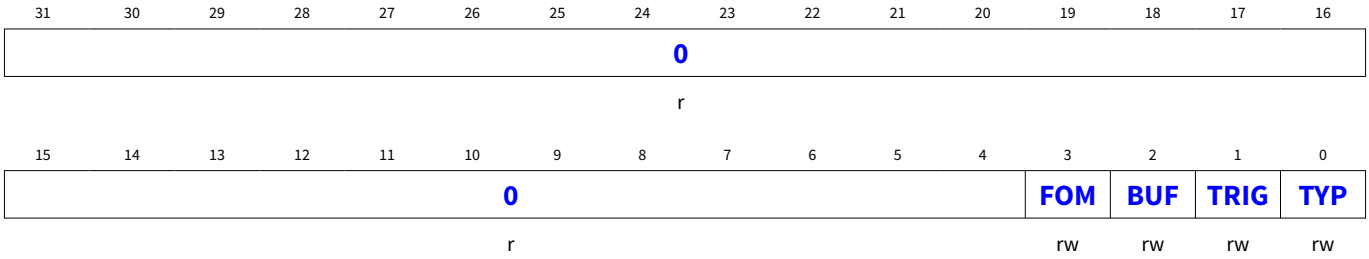
(continued)

Field	Bits	Type	Description
CTYP	16	rw	**Type of CAN message** This bit indicates the type of CAN message that is stored in the DMEM. When the incoming CAN frame has the DID which is configured to accept CAN I-PDU or vice-versa, the CAN message is discarded and Invalid destination ID error is triggered 0_B CAN Frame 1_B CAN I-PDU
0	31:17	r	**Reserved**

20.8.69 Destination memory i transfer mode configuration

DMEMi_MODE (i=0-27) Offset address: 01380_H+i*20_H

Destination memory i transfer mode configuration Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
TYP	0	rw	**PDU Mode enabled** The bit-field configures the destination memory in the case of a CAN frame or CAN I-PDU 0_B **MUXDIS**: Padding Padding bits are present between CAN messages 1_B **MUXEN**: No padding No padding bits present between CAN messages
TRIG	1	rw	**Trigger Mode** This bit-field selects between two interrupt trigger modes 0_B **INDEX**: Index-based interrupt trigger This index based trigger is valid only for CAN I-PDUs. The watermark and wraparound interrupts are triggered based on the DMEMi_FDBI 1_B **COUNT**: Count-based interrupt trigger The watermark and wraparound interrupts are triggered based on the buffer count DMEMi_STATUS.BC

(table continues...)

(continued)

Field	Bits	Type	Description
BUF	2	rw	**Buffer Mode** This bit configures the buffer split of the DMEM 0_B **SINGLE**: Single buffer mode The DMEM is divided into two parts Frame data buffer 1 and Frame data buffer 2 based on the Watermark level and Wraparound levels respectively. Note: Both the Frame data buffers always starts at a 64-bit aligned address. 1_B **CONT**: Continuous buffer mode The DMEM is treated as a continuous buffer. Only wraparound level is considered
FOM	3	rw	**FDBI overflow mode** This bit controls the data path of the new CAN frame or CAN I-PDU in case of no space in the Frame data buffer 0_B Continue If frame or I-PDU is received that does not fit in the current part of the DMEM Frame data buffer, it is placed into the other buffer when the EN bit is set. Otherwise there is a buffer overflow error triggered. 1_B Overflow The frame or I-PDU is discarded and an overflow error is triggered
0	31:4	r	**Reserved** Read as 0; should be written as 0

20.8.70 Destination memory i status

DMEMi_STATUS (i=0-27) Offset address: 01390_H+i*20_H

Destination memory i status Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MC												0	BO	WAF	WMF
rwh												r	rwh	rwh	rwh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BC								0	SID						NM
rh								r	rh						rwh

Field	Bits	Type	Description
NM	0	rwh	**New Message** This bit indicates a new message is transferred to destination memory. Hardware sets this bit and software write with 1 clears the bit. Software write with 0 has no effect.
SID	6:1	rh	**Source ID** This bit-field indicates the source from which the message is fetched.

(table continues...)

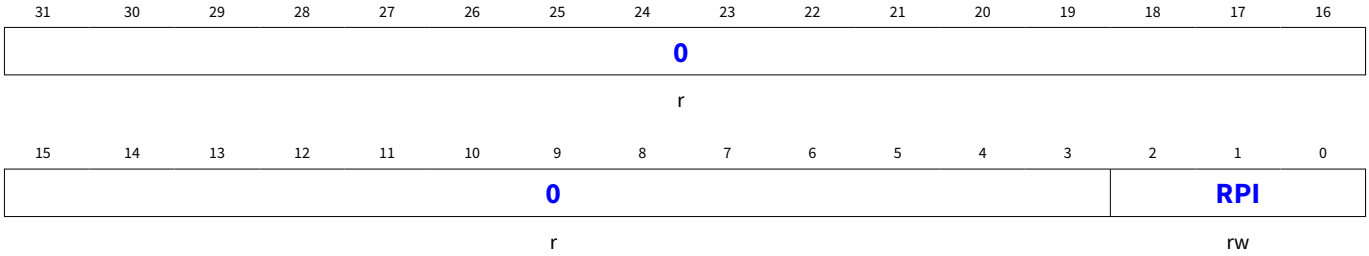
(continued)

Field	Bits	Type	Description
BC	15:8	rh	**Buffer Counter** This bit-field is incremented on every transfer of message to the destination memory. The counter is reset to 0 when a wraparound event has occurred. Note: In index mode, if CAN messages with zero payload or less payload are received, the count could exceed 255 depending on the configuration. In this case, the counter wraps around to 0
WMF	16	rwh	**Watermark Flag** This bit-field indicates watermark event. Hardware sets the flag and a software write with 1b clears the flag.
WAF	17	rwh	**Wraparound Flag** This bit-field is set when address wraparound has occurred. Hardware sets the flag and a software write with 1b clears the flag.
BO	18	rwh	**Buffer Overflow** This bit indicates a buffer overflow. This bit is set by hardware in case of an buffer overflow error (DBOE) and cleared by software by writing 1. Writing 0 has no effect.
MC	31:20	rwh	**Message Counter** This bit-field is incremented on every message transfer to the destination. A software write with a non zero value resets the counter value to 0. The counter wraps around to 0 upon overflow.
0	7, 19	r	**Reserved** Read as 0; should be written as 0

20.8.71 Destination memory i resource partition

DMEMi_RP (i=0-27) Offset address: 01394_H+i*20_H

Destination memory i resource partition Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
RPI	2:0	rw	**Resource Partition Index** This bit-field indicates the Resource Partition allocated to the destination memory i.
0	31:3	r	**Reserved** Read as 0; should be written with 0.

20.8.72 Move engine source address

This SFR indicates the first address the Move Engine is reading from.

ME_SRCA Offset address: $016F8_H$

Move engine source address Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADR															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADR															
rh															

Field	Bits	Type	Description
ADR	31:0	rh	**Source Address** This bit-field indicates the current address the Move Engine is reading from.

20.8.73 Move engine destination address

This SFR indicates the first address which the Move Engine is writing to.

ME_DESTA Offset address: $016FC_H$

Move engine destination address Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADR															
rh															

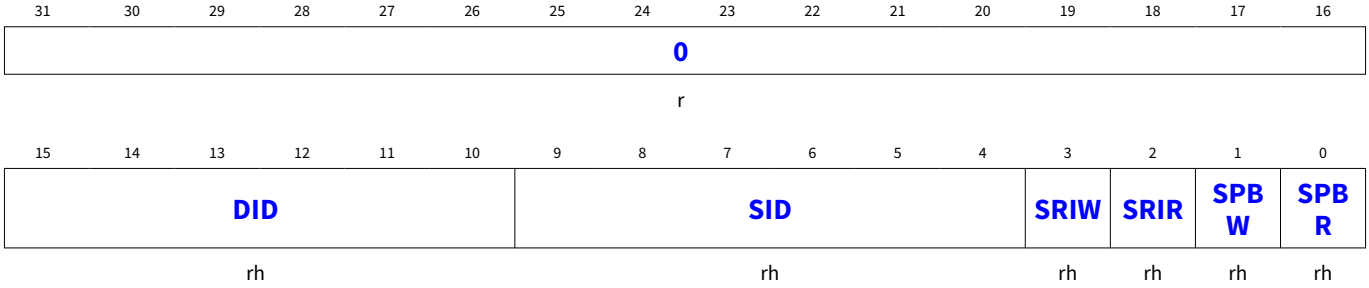
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADR															
rh															

Field	Bits	Type	Description
ADR	31:0	rh	**Destination Address** This bit-field indicates the current address which the Move Engine is writing to.

20.8.74 Move engine state

ME_STATE Offset address: 01700_H

Move engine state Kernel Reset value: $0000\ 0000_H$



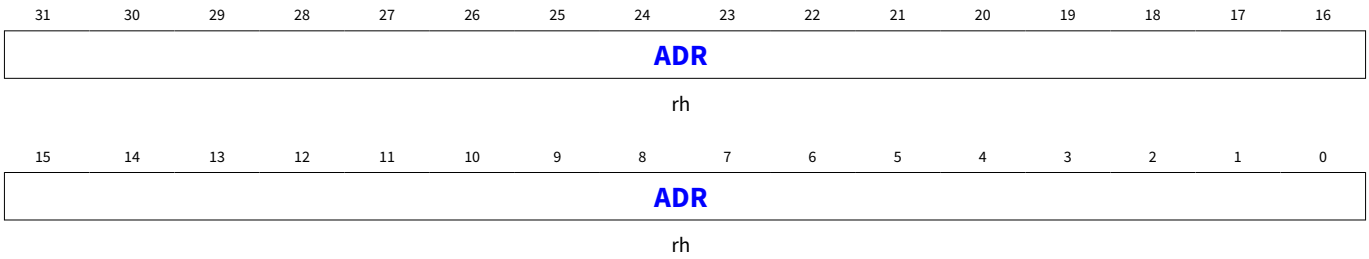
Field	Bits	Type	Description
SPBR	0	rh	**SPB Read** This bit-field is set when an FPI/SPB read operation is currently performed by the Move Engine
SPBW	1	rh	**SPB Write** This bit-field is set when an FPI/SPB write operation is performed by the Move Engine
SRIR	2	rh	**SRI Read** This bit-field is set when a SRI read operation is currently performed by the Move Engine
SRIW	3	rh	**SRI Write** This bit-field is set when a SRI write operation is performed by the Move Engine
SID	9:4	rh	**Source ID** This bit-field indicates the source ID of the current routing operation
DID	15:10	rh	**Destination ID** This bit-field indicates the destination ID of the current routing operation.
0	31:16	r	**Reserved** Read as 0; Write with 0

20.8.75 Move engine first error source address

This SFR indicates the first read address at which the Move Engine has an error.

ME_FESRCA Offset address: 01704_H

Move engine first error source address Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
ADR	31:0	rh	**First Error Source Address** This bit-field indicates the source address (that is start address of CAN Receive Host Buffer or CAN Output Buffer) for the routing request which has at least one of the IRDE, DBOE, SPBBE, SRIBE errors. The value of this bit-field is frozen once any of the aforementioned error flags are set. The value of this bit-field is ignored when there is no error (all Move Engine error status flags IRDE, DBOE, SPBBE, SRIBE are 0).

20.8.76 Move engine first error destination address

This SFR indicates the first write address at which the Move Engine has an error.

ME_FEDESTA Offset address: 01708_H

Move engine first error destination address Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADR															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADR															
rh															

Field	Bits	Type	Description
ADR	31:0	rh	**First Error Destination Address** This bit-field indicates the destination address (that is start address of virtual CAN buffer at memory destination or CAN Input Buffer or CAN Output Buffer) for the routing request which has at least one of the DBOE, SPBBE, SRIBE errors. The value of this bit-field is frozen once any of the aforementioned error flags are set. The value of this bit-field is ignored when there is no error (Move Engine error status flags DBOE, SPBBE, SRIBE are 0) or when FEDID has reserved destination ID values.

20.8.77 Move engine error register

ME_ERR Offset address: $0170C_H$

Move engine error register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FEDID						FESID						FEDIR	FEC		
rh						rh						rh	rh		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0											IRDE	DBOE	SRIBE	SPBBE	0
r											rwh	rwh	rwh	rwh	r

Field	Bits	Type	Description
SPBBE	1	rwh	**SPB Bus Transaction error** This bit-field indicates an error occurred during an FPI/SPB transaction performed by Move Engine. Hardware sets the flag and a software write with 1b clears the flag.
SRIBE	2	rwh	**SRI Transaction Error** This bit-field indicates an error occurred during an SRI transaction performed by the Move Engine. Hardware sets the flag and a software write with 1b clears the flag. Note: The hardware sets this bit and triggers interrupt INT_13 also in the event of read data error detection and correction (EDC) error and read data transaction ID error on SRI master interface (MIF)
DBOE	3	rwh	**Destination Buffer Overflow Error** This bit-field indicates the destination buffer overflow condition that is, the CAN frame size is larger than the buffer size of the corresponding destination (shown in FEDID). Hardware sets the flag and a software write with 1b clears the flag.
IRDE	4	rwh	**Invalid Routing Destination Error** This bit-field indicates in case of routing requests with a destination ID value corresponding to a reserved value or when the corresponding destination is disabled. Hardware sets the flag and a software write with 1b clears the flag.
FEC	18:16	rh	**First error code** This bit-field is bit encoded to indicate which error first occurred. This denotes that the FEDIR, FESID and FEDID corresponds to the error indicated by this bit-field 000_B No error 001_B IRDE 010_B DBOE 011_B SRIBE 100_B SPBBE **others,** Reserved
FEDIR	19	rh	**First Error Direction** This bit indicates whether access error was for read or write access 0_B Read 1_B Write

(table continues...)

(continued)

Field	Bits	Type	Description
FESID	25:20	rh	**First Error Source ID** This bit-field indicates the source ID of the first routing operation which has at least one of IRDE, DBOE, SPBBE or SRIBE errors. The value of this bit-field is ignored when there is no error (all Move Engine error status flags IRDE, DBOE, SPBBE or SRIBE are 0). The value of this bit-field is frozen once any of the aforementioned error flags are set. When the source ID of the routing or forwarding request is known for the corresponding transaction, this bit-field is updated with the SID. Otherwise in the case of the following scenarios where the source ID is unknown, this bit-field is set to: b111100 when a bus transaction error occurs during the write operation of the Rx descriptor Tail Pointer. RDESE is set and interrupt INT_10 triggered b111101 when a bus transaction error occurs during the write operation of the Tx descriptor Tail Pointer. TDESE is set and interrupt INT_14 triggered b111110 when a bus transaction error occurs during the read operation of the DMA status register.
FEDID	31:26	rh	**First Error Destination ID** This bit-field indicates the destination ID of the first routing operation which has at least one of IRDE, DBOE, SPBBE or SRIBE errors. The value of this bit-field is ignored when there is no error (all Move Engine error status flags IRDE, DBOE, SPBBE or SRIBE are 0). The value of this bit-field is frozen once any of the aforementioned error flags are set. When the destination ID of the routing or forwarding request is known for the corresponding transaction, this bit-field is updated with the DID. Otherwise in the case of the following scenarios when the destination ID is unknown, this bit-field is set to: b111111 when a bus transaction error occurs during the read operation of the routing header (RHEAD). b111100 when a bus transaction error occurs during the write operation of the Rx descriptor Tail Pointer. RDESE is set and interrupt INT_10 triggered b111101 when a bus transaction error occurs during the write operation of the Tx descriptor Tail Pointer. TDESE is set and interrupt INT_14 triggered b111110 when a bus transaction error occurs during the read operation of the DMA status register.
0	0, 15:5	r	**Reserved** Read as 0; should be written as 0

20.8.78 Interrupt signal

DRE signals events to the Interrupt Router module through 16 interrupt lines INT[15:0]. This register shows the status of interrupt signalled for the events grouped to an interrupt line. It means at least one status event has occurred within an interrupt line. The bit-fields value is purely OR-ing of the status bits corresponding to the interrupt line.

INTSIG Offset address: 01710_H

Interrupt signal Kernel Reset value: $0000\ 0800_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
INT15	INT14	INT13	INT12	INT11	INT10	INT9	INT8	INT7	INT6	INT5	INT4	INT3	INT2	INT1	INT0
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
INTi (i=0-7)	i	rh	**Interrupt Line i** The interrupt line INT0 to INT7 corresponds to DMEMi_STATUS.WAF, WMF events. Each destination memory can assign its events to one of the INT0 till INT7 lines by configuring DMEMi_CONFIG.INP. 0_B **DIS**: No WM or WA event 1_B **EN**: WM or WA event
INT8	8	rh	**CAN input buffer status interrupt** This interrupt line corresponds to • CAN Input Buffer List full event (CIBL_STATUS.BF). • CAN CRC error (CIBL_STATUS.CRCE) • CAN WDT error (CIBL_STATUS.WDTE) 0_B **OK**: CIBUF status OK 1_B **NOK**: CIBUF status Not OK
INT9	9	rh	**CAN output buffer status interrupt** This interrupt line corresponds to • CAN Output Buffer List full event (COBL_STATUS.BF). • CAN WDT error (COBL_STATUS.WDTE) 0_B **OK**: COBUF status OK 1_B **NOK**: COBUF status not OK
INT10	10	rh	**Ethernet input buffer frame error and status interrupts** This interrupt line corresponds to • Input Ethernet frame error events (EIBUFi_STATUS.FE) • Invalid frame type (EIBUFi_STATUS.IFT) • AVTP Length Mismatch Error (EIBUFi_STATUS.LME) • Remote frame error (EIBUFi_STATUS.RFE) • Invalid destination ID (EIBUFi_STATUS.IDID) • CAN Format Error (EIBUFi_STATUS.CFE) • Ethernet watchdog timeout error (EIBUFi_ERROR.WDTE) • Rx descriptor error (EIBUFi_ERROR.RDESE) • Ethernet Input buffer full (EIBUFi_ERROR.BF) 0_B **OK**: EIBUF status OK 1_B **NOK**: EIBUF status not OK

(table continues...)

(continued)

Field	Bits	Type	Description
INT11	11	rh	**Ethernet Frame Receive Request** This interrupt line corresponds to input EIBUFi_STATUS.RXREQ 0_B **DIS**: Request disabled 1_B **EN**: Request enabled
INT12	12	rh	**Routing Table Error Interrupt** This interrupt line corresponds to CAN Transmit Routing Table error events (RS.IRT, NMFE). 0_B **OK**: No error 1_B **NOK**: Routing table error
INT13	13	rh	**ME Routing Transaction Lost** This interrupt line corresponds to Move Engine error events (ME_ERR.DBOE, IRDE, SRIBE, SPBBE). 0_B **OK**: No error 1_B **NOK**: Move engine error
INT14	14	rh	**Ethernet output buffer error and status interrupts** This interrupt line corresponds to • Output Ethernet frame WDT error (EOBUFj_ERROR.WDTE) • Output Ethernet frame Tx descriptor error (EOBUFj_ERROR.TDESE) • Ethernet output buffer full (EOBUFi_STATUS.BF) 0_B **OK**: EOBUF status OK 1_B **NOK**: EOBUF status not OK
INT15	15	rh	**Ethernet Frame Transmit Request** This interrupt line corresponds to • Ethernet output frame transmit request events (EOBUFj_STATUS.TXREQ) 0_B **DIS**: Request disabled 1_B **EN**: Request enabled
0	31:16	r	**Reserved** Read as 0; should be written with 0.

20.8.79 Interrupt line enable

The interrupt lines INT[15:0] are enabled through this SFR.

IE Offset address: 01714_H

Interrupt line enable Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IE15	IE14	IE13	IE12	IE11	IE10	IE9	IE8	IE7	IE6	IE5	IE4	IE3	IE2	IE1	IE0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
IEi (i=0-15)	i	rw	**Interrupt Line i Enabled** This bit-field enables the corresponding interrupt. Upon relevant status events, the corresponding interrupt lines are triggered only when they are enabled by setting this bit-field to 1. 0_B Interrupt line is disabled 1_B Interrupt line is enabled
0	31:16	r	**Reserved** Read as 0; should be written with 0.

20.8.80 Rx Ethernet descriptor list i configuration and control

RETHDLi_CTRL (i=0-5) Offset address: $0171C_H$+i*8

Rx Ethernet descriptor list i configuration and control Kernel Reset value: $0000\ 0180_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PTR			STOP	STOPACK	IOC	0									
rwh			rw	rh	rw	r									

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0							FCS	TRIG	0	EIFID		DMACH			EIF
r							rw	rw	r	rw		rw			rw

Field	Bits	Type	Description
EIF	0	rw	**Ethernet Interface** This bit-field selects between the LETH and GETH interface to which the Rx and Tx descriptor list is mapped. One descriptor list consists of 4 descriptors 0_B GETH 1_B LETH
DMACH	3:1	rw	**DMA channel number** This bit-field configures the Ethernet DMA CH to which the Rx descriptor list belongs. Two Rx descriptor lists cannot be mapped to the same Rx DMA channel. This bit-field is used by the hardware only to generate the FWD ID and not in identifying the DMA channel mapped. Note: Irrespective of the value configured in this bit-field, the DRE only receives/transmits frames from/to the Ethernet DMA channel whose Tail Pointer address is configured in the Ethernet Address Database (EAD) at the corresponding index RETHDLi_CTRL.EIF and EIFID

(table continues...)

(continued)

Field	Bits	Type	Description
EIFID	5:4	rw	**Ethernet interface ID** This bit-field indicates the source or destination Ethernet Interface ID. x is determined by bit EIF Note: When EIFID > 1, EIF will be LETH by default 00_B **xETH0**: xETH0 interface 01_B **xETH1**: xETH1 interface 10_B **xETH2**: xETH2 interface EIF shall be LETH 11_B **xETH3**: xETH3 interface EIF shall be LETH
TRIG	7	rw	**Trigger type** This bit-field configures the type of trigger from the Ethernet Descriptor Handler. This bit-field is cleared by software only at the end, after the configuration of the Rx DMA channel, in order for the DRE to start preparing the Rx descriptors 0_B **DMA**: Rx from Ethernet DMA This is the descriptor mode where Rx descriptors are prepared by the DRE. Ethernet frame is received from Ethernet DMA. The Ethernet Descriptor Handler gives a Receive Poll Demand by updating the Ethernet Rx DMA channel Tail Pointer. 1_B **IR**: Trigger interrupt An interrupt is triggerd to the IR. The SW shall write an Ethernet frame into the EIBUF based on the interrupt. The software shall clear the TRIG bit only at the end, after the configuration of the Rx descriptor list, in order to trigger the DRE to start preparing the Rx descriptors.
FCS	8	rw	**Receive Ethernet packet has FCS field** This bit-field indicates if the Ethernet frame received by the corresponding Rx descriptor has FCS field included or not. This bit-field must be configured by SW during the initialization phase of DRE and shall not be changed during run-time. This bit-field value is used by DRE only for Ethernet-to-Ethernet forwarding use-case, to configure the TDESCi_RD3.CPC bit-field of the corresponding Transmit descriptor. If FCS = 0, then the corresponding TDESCi_RD3.CPC bit-field is set to b00 (Enable CRC and PAD insertion) or else TDESCi_RD3.CPC bit-field is set to b10 (Disable CRC insertion).
IOC	26	rw	**Interrupt on completion** This bit is the static configuration for Rx descriptors RDESCi_RD3G.IOC for GETH and RDESCi_RD3L.IOC for LETH

(table continues...)

(continued)

Field	Bits	Type	Description
STOPACK	27	rh	**STOP acknowledge** This bit is set by the hardware in case the STOP bit is set by the software. The Rx Descriptor Handler sets this bit after it finishes the ongoing processes and goes into IDLE state. The software must then update the RETHDLi_CTRL.PTR after STOPACK is set. The RETHDLi_CTRL.STOP bit must be cleared by software after updating the RETHDLi_CTRL.PTR. Clearing of STOP bit will trigger DRE to clear STOPACK and set OWN bit of the RDESCi pointed to by the new updated RETHDLi_CTRL.PTR and trigger a Receive Poll Demand to the Rx DMA channel. Note: All writes to RETHDLi_CTRL.PTR bit while the STOPACK bit is 0 are ignored and have no effect
STOP	28	rw	**STOP bit** This bit is set by the software in the event that the descriptor pointer RETHDLi_CTRL.PTR needs to be reset. The Rx Descriptor Handler sets the RETHDLi_CTRL.STOPACK after it finishes the ongoing processes and goes into IDLE state. The software must then update the RETHDLi_CTRL.PTR after STOPACK is set. The STOP bit must be cleared by software after updating the RETHDLi_CTRL.PTR. Clearing of STOP bit will trigger DRE to clear STOPACK and set OWN bit of the RDESCi pointed to by the newly updated RETHDLi_CTRL.PTR and trigger a Receive Poll Demand to the Rx DMA channel.
PTR	31:29	rwh	**Descriptor pointer** This bit-field points to the index of the next Rx descriptor to be prepared and whose OWN bit should be set by the Ethernet Descriptor Handler. Valid values are 0 to 3. This bit-field wraps around after 3 Note: For descriptors owned by the DRE, the OWN bit of RDESCi_WR3G for GETH or RDESCi_WR3L for LETH is reset to 0. For descriptors owned by the DMA, the OWN bit is set to 1 All writes to this bit-field are ignored when RETHDLi_CTRL.STOPACK is not set to 1
0	6, 25:9	r	**Reserved** Read as 0; should be written with 0.

20.8.81 Tx Ethernet descriptor list i configuration and control

TETHDLi_CTRL (i=0-5) Offset address: $0174C_H+i*10_H$

Tx Ethernet descriptor list i configuration and control Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PTR			STOP	STOPACK	IOC	SAIC			SLOTNUM				0		
rwh			rw	rh	rw	rw			rw				r		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0									TRIG	0		DMACH			0
r									rw	r		rw			r

Field	Bits	Type	Description
DMACH	3:1	rw	**DMA channel number** This bit-field is redundant to RETHDLi_CTRL.DMACH and indicates the Ethernet DMA CH to which the Tx descriptor list belongs. Two Tx descriptor lists cannot be mapped to the same Tx DMA channel. This bit-field acts as status information only. The hardware considers that Rx and Tx DMA channels have the same ID. For example, LETH0 has Rx DMA channel 0 and Tx DMA channel 0 mapped to DRE. Note: Irrespective of the value configured in this bit-field, the DRE only receives/transmits frames from/to the Ethernet DMA channel whose Tail Pointer address is configured in the Ethernet Address Database (EAD) at the corresponding index RETHDLi_CTRL.EIF and EIFID
TRIG	6	rw	**Trigger type** This bit-field configures the type of trigger from the Ethernet Descriptor Handler 0_B **DMA**: Tx to Ethernet DMA This is the descriptor mode where Tx descriptors are prepared by DRE. Ethernet frame is transmitted to Ethernet DMA. The Ethernet Descriptor Handler gives a Transmit Poll Demand by updating the Ethernet Tx DMA channel Tail Pointer. 1_B **IR**: Trigger interrupt An interrupt is triggerd to the IR. The SW shall read the Ethernet frame from the EOBUF/EIBUF based on the interrupt
SLOTNUM	22:19	rw	**Slot Number configuration** This bit is the static configuration for all Tx descriptors TDESCi_RD3.SLOTNUM.
SAIC	25:23	rw	**SA insertion control configuration** This bit is the static configuration for all Tx descriptors TDESCi_RD3.SAIC. • 2'b00: Do not include the source address • 2'b01: Reserved • 2'b10: Replace the source address. For reliable transmission, the application must provide frames with source addresses • 2'b11: Reserved
IOC	26	rw	**Interrupt on completion** This bit is the static configuration for all Tx descriptors TDESCi_RD2.IOC
STOPACK	27	rh	**STOP acknowledge** This bit is set by the hardware in case the STOP bit is set by the software. The Tx Descriptor Handler sets this bit after it finishes the ongoing processes and goes into IDLE state. The software must then update the TETHDLi_CTRL.PTR after STOPACK is set. The TETHDLi_CTRL.STOP bit must be cleared by software after updating the TETHDLi_CTRL.PTR. Clearing of STOP bit will trigger DRE to clear STOPACK and set OWN bit of the TDESCi pointed to by the new updated TETHDLi_CTRL.PTR and trigger a Transmit Poll Demand to the Tx DMA channel. Note: All writes to TETHDLi_CTRL.PTR bit while the STOPACK bit is 0 are ignored and have no effect

(table continues...)

(continued)

Field	Bits	Type	Description
STOP	28	rw	**STOP bit** This bit is set by the software in case the descriotor pointer TETHDLi_CTRL.PTR needs to be reset. The Tx Descriptor Handler sets the TETHDLi_CTRL.STOPACK after it finishes the ongoing processes and goes into IDLE state. The software must then update the TETHDLi_CTRL.PTR after STOPACK is set. The STOP bit must be cleared by software after updating the TETHDLi_CTRL.PTR. Clearing of STOP bit will trigger DRE to clear STOPACK and set OWN bit of the TDESCi pointed to by the new updated TETHDLi_CTRL.PTR and trigger a Transmit Poll Demand to the Tx DMA channel.
PTR	31:29	rwh	**Descriptor pointer** This bit-field points to the index of index of the next Tx descriptor to be prepared and whose OWN bit should be set by the Ethernet Descriptor Handler Valid values are 0 to 3. This bit-field wraps around after 3 Note : For descriptors owned by the DRE, the OWN bit of TDES3 is reset to 0. For descriptors owned by the DMA, the OWN bit is set to 1 All writes to this bit-field is ignored when TETHDLi_CTRL.STOPACK is not set to 1
0	0, 5:4, 18:7	r	**Reserved** Read as 0; should be written with 0

20.8.82 Ethernet descriptor list status

EDLSTAT Offset address: $017D8_H$

Ethernet descriptor list status Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		TXCNT								RXCNT					
r		rwh								rwh					

Field	Bits	Type	Description
RXCNT	5:0	rwh	**Receive count** This bit-field is incremented by 1 after every succesful receive of Ethernet frame. The counter wraps around to 0 by hardware upon overflow. A write operation of any non zero value resets the counter value to 0.

(table continues...)

(continued)

Field	Bits	Type	Description
TXCNT	13:6	rwh	**Transmit and forward count** This bit-field is incremented by 1 after every succesful trasmit or forward of Ethernet frame. The counter wraps around to 0 by hardware upon overflow. SW write has no effect
0	31:14	r	**Reserved** Read as 0; Written as 0

20.8.83 Ethernet requests summary

EREQ

Ethernet requests summary

Offset address: 017DC$_H$

Kernel Reset value: 0000 0000$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CBI5			CBI4			CBI3			CBI2			CBI1			0
rh			rh			rh			rh			rh			r

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CBI0			0	FWD5_REQ	FWD4_REQ	FWD3_REQ	FWD2_REQ	FWD1_REQ	FWD0_REQ	TX5_REQ	TX4_REQ	TX3_REQ	TX2_REQ	TX1_REQ	TX0_REQ
rh			r	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
TXi_REQ (i=0-5)	i	rwh	**Tx request** This bit-field is set by hardware whenever there is an ACF Ethernet frame to be transmitted in the EOBUF. It is cleared by hardware after the Ethernet frame is sucessfully transmitted and there is no error Write-back format of the descriptor. In the case of an interrupt (INT_15), this bit is cleared by the hardware after the frame is read from EOBUF by the software. In the case of error, this bit is cleared by software after reading the frame from the buffer. If there is a buffer full condition, this would be cleared by hardware together with the EOBUFj_STATUS.BF flag. Software write with 1 clears this flag. Software write with 0 has no effect
FWDi_REQ (i=0-5)	i+6	rwh	**Forward request** This bit-field is set by the Forwarding engine whenever there is an Ethernet frame to be forwarded in the EIBUF. When there is a matching filter element, the Forwarding engine sets the EREQ.FWDi_REQ of the corresponding Tx descriptor list FT_FEj_FRULE.DSEL. It is cleared by the Descriptor Handler after the Ethernet frame is sucessfully forwarded and there is no error Write-back format of the descriptor. In case of error, this bit is cleared by software after reading the frame from the buffer. Software write with 1 clears this flag. Software write with 0 has no effect

(table continues...)

(continued)

Field	Bits	Type	Description
CBI0	15:13	rh	**Current buffer index 0** Indicates the buffer index of EIBUF for which the Tx descriptors of Tx descriptor list 0 are being prepared in case of forwarding
CBI1	19:17	rh	**Current buffer index 1** Indicates the buffer index of EIBUF for which the Tx descriptors of Tx descriptor list 1 are being prepared in case of forwarding
CBI2	22:20	rh	**Current buffer index 2** Indicates the buffer index of EIBUF for which the Tx descriptors of Tx descriptor list 2 are being prepared in case of forwarding
CBI3	25:23	rh	**Current buffer index 3** Indicates the buffer index of EIBUF for which the Tx descriptors of Tx descriptor list 3 are being prepared in case of forwarding
CBI4	28:26	rh	**Current buffer index 4** Indicates the buffer index of EIBUF for which the Tx descriptors of Tx descriptor list 4 are being prepared in case of forwarding
CBI5	31:29	rh	**Current buffer index 5** Indicates the buffer index of EIBUF for which the Tx descriptors of Tx descriptor list 5 are being prepared in case of forwarding
0	12, 16	r	**Reserved** Read as 0; Written as 0

20.8.84 Forwarding table configuration

FTCFG Offset address: $017E4_H$

Forwarding table configuration Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								FID							
r								rh							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FID								NRULES							
rh								rw							

Field	Bits	Type	Description
NRULES	7:0	rw	**Number of forwarding rules** This bit-field configures the number of forwarding rules contained in the Forwarding Table. Valid values are from 0 to 128 0 - Forwarding Table is disabled 1 to 128 - Number of forwarding rules in the Forwarding Table >128 - Invalid configuration

(table continues...)

(continued)

Field	Bits	Type	Description
FID	30:8	rh	**Forwarding ID** Forwarding ID generated by the forwarding engine **1.** GETH FID : EIF | DMACH | FRPLI | L3L4FM | MADRM • FRPLI : Bits [7:0] of RDESCi_WR1G[31:0] • L3L4FM : RDESCi_WR2G[31:29] • MADRM : RDESCi_WR2G[26:19] **2.** LETH FID : EIF | DMACH | FRPLI | xxxx| MADRM • FRPLI : Bits [31:24] of RDESCi_WR2L[31:16] • MADRM : RDESCi_WR2L[26:19]
0	31	r	**Reserved** Read as 0; Write with 0

20.8.85 DRE CAN watchdog configuration

The DRE Watchdog timer register to monitor CAN timeout.

CWDCFG Offset address: $017EC_H$

DRE CAN watchdog configuration Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CTO															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0														WTOE	EN
r														rw	rw

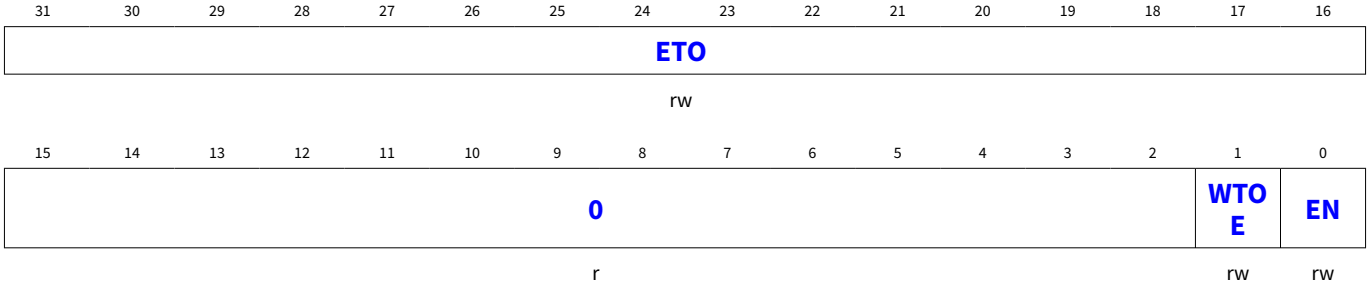
Field	Bits	Type	Description
EN	0	rw	**Enable CAN watchdog** This bit-field enables the CAN watchdog timer 0_B **DIS**: CAN watchdog disabled 1_B **EN**: CAN watchdog enabled
WTOE	1	rw	**CAN watchdog timeout error** This bit field enables / disables the generation of interrupt when there is a timeout. 0_B **DIS**: Timeout interrupt disabled 1_B **EN**: Timeout interrupt enabled
CTO	31:16	rw	**CAN timeout value** This bit-field configures the timeout prescaler of the fSRI clock. This bit-field can only be written when the watchdog timer is disabled
0	15:2	r	**Reserved** Read as 0; should be written as 0

20.8.86 DRE Ethernet watchdog configuration

The DRE Watchdog timer register to monitor Ethernet timeout.

EWDCFG Offset address: $017F4_H$

DRE Ethernet watchdog configuration Kernel Reset value: $0000\ 0000_H$

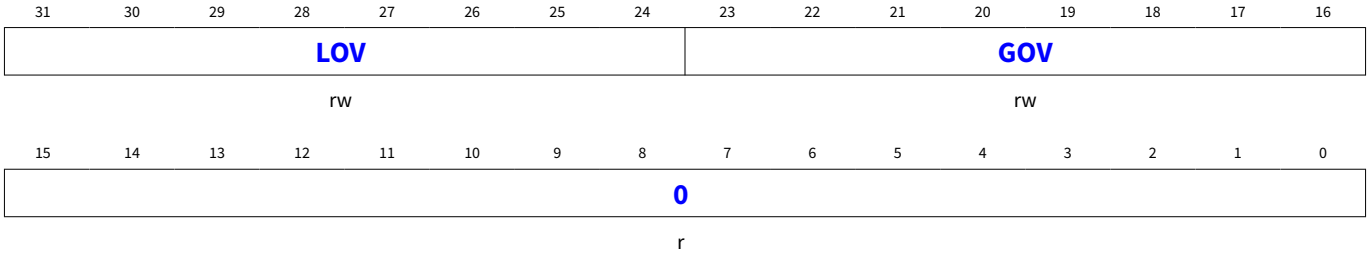


Field	Bits	Type	Description
EN	0	rw	**Enable Ethernet watchdog** This bit-field enables the Ethernet watchdog timer 0_B **DIS**: Ethernet watchdog disabled 1_B **EN**: Ethernet watchdog enabled
WTOE	1	rw	**Ethernet watchdog timeout error** This bit field enables / disables the generation of interrupt when there is a timeout. 0_B **DIS**: Timeout interrupt disabled 1_B **EN**: Timeout interrupt enabled
ETO	31:16	rw	**Ethernet timeout value** This bit-field configures the timeout prescaler of the fSRI clock. This bit-field can only be written when the watchdog timer is disabled.
0	15:2	r	**Reserved** Read as 0; should be written as 0

20.8.87 Ethernet address database configuration

EADCFG Offset address: $017F8_H$

Ethernet address database configuration Kernel Reset value: $3834\ 0000_H$



Field	Bits	Type	Description
GOV	23:16	rw	**GETH offset value** This bit-field configures the offset of the GETH DMA_CHj_Status (j=0-7) from the GETH DMA_CHj_RxDesc_Tail_LPointer (i=0-7) register
LOV	31:24	rw	**LETH offset value** This bit-field configures the offset of the LETH DMA_CHy_Status (y=0-7) from the LETH DMA_CHy_RxDesc_Tail_Pointer (y=0-7) register
0	15:0	r	**Reserved** Read as 0; Written as 0

20.8.88 DMA i resource partition

DMAi_RP (i=0-5) Offset address: 017FC$_H$+i*4

DMA i resource partition Kernel Reset value: 0000 0000$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0													RPI		
r													rw		

Field	Bits	Type	Description
RPI	2:0	rw	**Resource partition index** This bit-field indicates the Resource Partition allocated to the Rx and Tx DMA of Ethernet MAC.
0	31:3	r	**Reserved** Read as 0; should be written as 0

20.8.89 CAN Input buffer timeout status

CITO Offset address: 01818$_H$

CAN Input buffer timeout status Kernel Reset value: 0000 0000$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0												TOUT19	TOUT18	TOUT17	TOUT16
r												rwh	rwh	rwh	rwh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TOUT15	TOUT14	TOUT13	TOUT12	TOUT11	TOUT10	TOUT9	TOUT8	TOUT7	TOUT6	TOUT5	TOUT4	TOUT3	TOUT2	TOUT1	TOUT0
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
TOUTj (j=0-19)	j	rwh	**Timeout error** When set, the corresponding CIBUF has a timeout error. The software shall clear this bit by writing 1. Writing zero has no effect. The software can also clear the BPR if required. But if the software does not clear the BPR bit, the CAN frame in the CIBUF is processed again by the ACF engine after this timeout bit is cleared by software
0	31:20	r	**Reserved** Read 0; write 0

20.8.90 CAN Output buffer timeout status 0

COTO0 Offset address: $0181C_H$

CAN Output buffer timeout status 0 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TOUT31	TOUT30	TOUT29	TOUT28	TOUT27	TOUT26	TOUT25	TOUT24	TOUT23	TOUT22	TOUT21	TOUT20	TOUT19	TOUT18	TOUT17	TOUT16
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TOUT15	TOUT14	TOUT13	TOUT12	TOUT11	TOUT10	TOUT9	TOUT8	TOUT7	TOUT6	TOUT5	TOUT4	TOUT3	TOUT2	TOUT1	TOUT0
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
TOUTj (j=0-31)	j	rwh	**Timeout error** When set, the corresponding COBUF has a timeout error. The software shall clear this bit by writing 1. Writing zero has no effect. The software can also clear the BPR if required. But if the software does not clear the BPR bit, the CAN frame in the CIBUF is processed again by the ACF engine after this timeout bit is cleared by software

20.8.91 CAN Output buffer timeout status 1

COTO1 Offset address: 01820_H

CAN Output buffer timeout status 1 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TOUT63	TOUT62	TOUT61	TOUT60	TOUT59	TOUT58	TOUT57	TOUT56	TOUT55	TOUT54	TOUT53	TOUT52	TOUT51	TOUT50	TOUT49	TOUT48
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TOUT47	TOUT46	TOUT45	TOUT44	TOUT43	TOUT42	TOUT41	TOUT40	TOUT39	TOUT38	TOUT37	TOUT36	TOUT35	TOUT34	TOUT33	TOUT32
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
TOUTj (j=32-63)	j-32	rwh	**Timeout error** When set, the corresponding COBUF has a timeout error. The software shall clear this bit by writing 1. Writing zero has no effect. The software can also clear the BPR if required. But if the software does not clear the BPR bit, the CAN frame in the CIBUF is processed again by the ACF engine after this timeout bit is cleared by software

20.8.92 CAN Address Database RAM interface

20.8.92.1 RAM CAN address database CRE start address

The CAN address database elements are stored in DRE RAM starting from DRE RAM base address.

The CAN Address Database is stored in DRE RAM starting from DRE RAM base address. This RAM location configured by user with start address of the CAN CRE RAM allocation.

CAD_CANi_CRESA (i=0-19) Offset address: $00000_H + i*8$

RAM CAN address database CRE start address RAMInit value: $XXXX\ XXXX_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADR															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADR															
rw															

Field	Bits	Type	Description
ADR	31:0	rw	**Address** This bit-field is configured by user with start address of the CANi interface CRE RAM allocation.

20.8.93 CAN Input Buffer List RAM interface

20.8.93.1 RAM CIBUF routing header

The CAN Input Buffer List is stored in DRE RAM starting from DRE RAM base address + 0xA0. The number of buffers in the list is 20.

CIBUFj_RHEAD (j=0-19) Offset address: $000A0_H+j*50_H$

RAM CIBUF routing header RAMInit value: $XXXX\ XXXX_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		ETHDID						SCBID						0	
r		rh						rh						r	

Field	Bits	Type	Description
SCBID	7:2	rh	**Source CAN Bus ID** This bit-field indicates the source CAN interface ID from which the CAN frame is received. Refer to Routing destination ID table
ETHDID	13:8	rh	**Ethernet Destination ID** This bit-field indicates the destination ID of Ethernet frame. Valid IDs are from 0x18 till 0x1D. The destination is disabled when ID value is 0x0. Refer to the Routing destination ID table
0	1:0, 31:14	r	**Reserved** Read as 0; should be written with 0

20.8.93.2 RAM CIBUF CRC computed by CRE

The CAN Input Buffer List is stored in DRE RAM starting from DRE RAM base address + 0xA0. The number of buffers in the list is 20.

CIBUFj_CRC (j=0-19) Offset address: $000A4_H+j*50_H$

RAM CIBUF CRC computed by CRE RAMInit value: $XXXX\ XXXX_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CRC															
rwh															

Field	Bits	Type	Description
CRC	15:0	rwh	**CRC** CRC computed by CRE
0	31:16	r	**Reserved**

20.8.93.3 RAM CIBUF register 0

The CAN Input Buffer List is stored in DRE RAM starting from DRE RAM base address + 0xA0. The number of buffers in the list is 20.

CIBUFj_R0 (j=0-19) Offset address: $000A8_H+j*50_H$

RAM CIBUF register 0 RAMInit value: $XXXX\ XXXX_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ESI	XTD	RTR	ID												
rh	rh	rh	rh												

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ID															
rh															

Field	Bits	Type	Description
ID	28:0	rh	**CAN Identifier** Standard or extended identifier depending on bit XTD. A standard identifier is stored into ID[28:18].
RTR	29	rh	**Remote Transmission Request** This bit-field indicates that the CAN frame is a remote frame. There are no remote frames in CAN FD format. In case of CAN FD frame (FDF = 1), this bit-field is ignored. 0_B Received frame is a data frame 1_B Received frame is a remote frame
XTD	30	rh	**Extended Identifier** Indicates whether the CAN frame has a standard or extended identifier. 0_B 11-bit standard identifier 1_B 29-bit extended identifier
ESI	31	rh	**Error State Indicator** 0_B Transmitting node is error active 1_B Transmitting node is error passive

20.8.93.4 RAM CIBUF register 1

The CAN Input Buffer List is stored in DRE RAM starting from DRE RAM base address + 0xA0. The number of buffers in the list is 20.

CIBUFj_R1 (j=0-19) Offset address: $000AC_H+j*50_H$

RAM CIBUF register 1 RAMInit value: $XXXX\ XXXX_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0										FDF	BRS	DLC			
r										rwh	rwh	rwh			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															
r															

Field	Bits	Type	Description
DLC	19:16	rwh	**Data Length Code** 0_H CAN + CAN FD: received frame has 0 data bytes … 8_H CAN + CAN FD: received frame has 8 data bytes 9_H CAN FD: received frame has 12 (9*4-24) data bytes CAN: received frame has 8 data bytes … C_H CAN FD: received frame has 24 (12*4-24) data bytes CAN: received frame has 8 data bytes D_H CAN FD: received frame has 32 (13*16-176) data bytes CAN: received frame has 8 data bytes … F_H CAN FD: received frame has 64 (15*16-176) data bytes CAN: received frame has 8 data bytes
BRS	20	rwh	**Bit Rate Switch** 0_B Frame received without bit rate switching 1_B Frame received with bit rate switching
FDF	21	rwh	**Frame Data Format** 0_B Standard frame format 1_B CAN FD frame format (new DLC-coding and CRC)
0	15:0, 31:22	r	**Reserved** Read as 0; should be written with 0.

20.8.93.5 RAM CIBUF data byte m

This register indicates value of CAN payload data byte m.

CIBUFj_DBm (j=0-19;m=0-63) Offset address: $000B0_H+j*50_H+m$

RAM CIBUF data byte m RAMInit value: XX_H

7	6	5	4	3	2	1	0
DB							
rwh							

Field	Bits	Type	Description
DB	7:0	rwh	**Data Byte m**

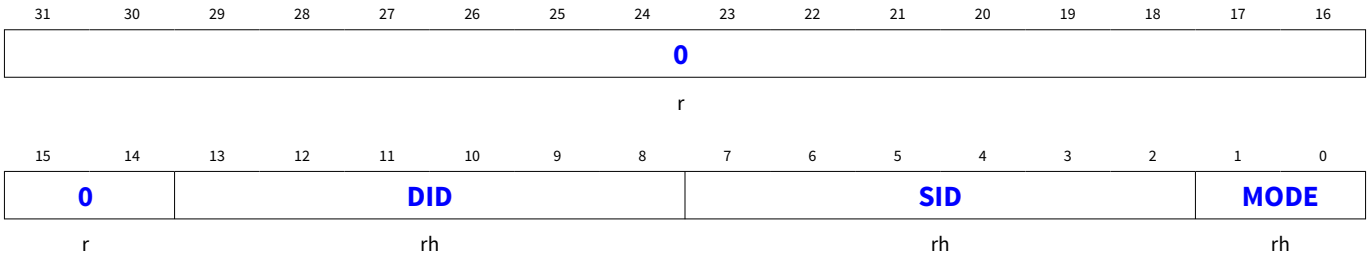
20.8.94 CAN Output Buffer List RAM interface

20.8.94.1 RAM uni-cast routing header

The CAN Output Buffer List is stored in DRE RAM starting from DRE RAM base address + 0x6E0. The number of buffers in the list is 64.

COBUFj_UCRH (j=0-63) Offset address: $006E0_H+j*50_H$

RAM uni-cast routing header RAMInit value: $XXXX\ XXXX_H$



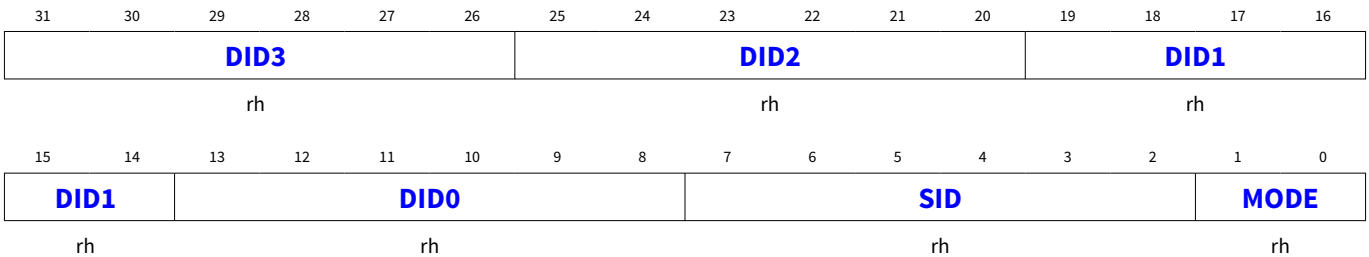
Field	Bits	Type	Description
MODE	1:0	rh	**Routing Rule Mode** It defines the type of the routing rule 0 - Uni-cast rule 1 - Multi-cast rule other - Reserved, considered as Uni-cast rule
SID	7:2	rh	**Source ID** This bit-field indicates the source from which the CAN frame is received. The source module are referenced using a unique ID as given in the Routing destination ID table. • EIBUF0 : 0x18 • EIBUF1 : 0x19 • EIBUF2 : 0x1A • EIBUF3 : 0x1B • EIBUF4 : 0x1C • EIBUF5 : 0x1D
DID	13:8	rh	**Destination ID** This bit-field indicates the destination CAN node to which the received CAN frame has to be transferred. The destination CAN node are referenced using a unique ID as given in the Routing destination ID table.
0	31:14	r	**Reserved** Read as 0; should be written with 0.

20.8.94.2 RAM multi-cast routing header

The CAN Output Buffer List is stored in DRE RAM starting from DRE RAM base address + 0x6E0. The number of buffers in the list is 64.

COBUFj_MCRH (j=0-63) Offset address: $006E0_H+j*50_H$

RAM multi-cast routing header RAMInit value: $XXXX\ XXXX_H$



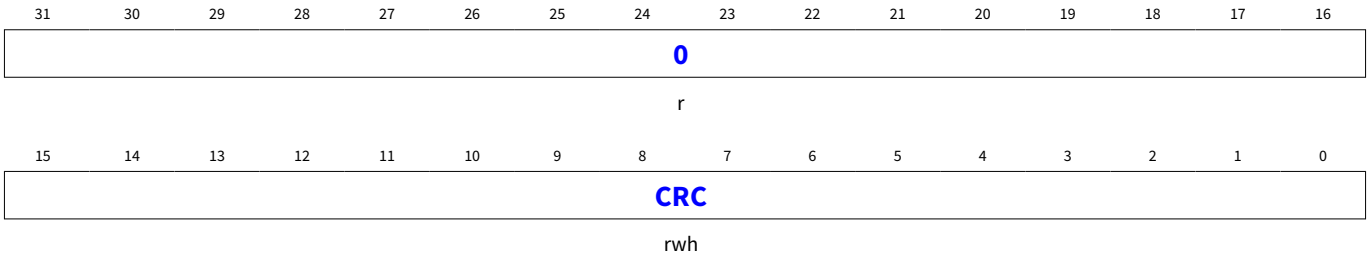
Field	Bits	Type	Description
MODE	1:0	rh	**Routing Rule Mode** It defines the type of the routing rule 0 - Uni-cast rule 1 - Multi-cast rule other - Reserved, considered as Uni-cast rule
SID	7:2	rh	**Source ID** This bit-field indicates the source from which the CAN frame is received. The source module are referenced using a unique ID as given in the Routing destination ID table. • EIBUF0 : 0x18 • EIBUF1 : 0x19 • EIBUF2 : 0x1A • EIBUF3 : 0x1B • EIBUF4 : 0x1C • EIBUF5 : 0x1D
DID0	13:8	rh	**Destination ID 1** This bit-field indicates the 1st destination CAN node to which the received CAN frame has to be transferred. The destination CAN node is referenced using a unique ID as given in the Routing destination ID table.
DID1	19:14	rh	**Destination ID 2** This bit-field indicates the 2nd destination CAN node to which the received CAN frame has to be transferred. The destination CAN node is referenced using a unique ID as given in the Routing destination ID table.
DID2	25:20	rh	**Destination ID 3** This bit-field indicates the 3rd destination CAN node to which the received CAN frame has to be transferred. The destination CAN node is referenced using a unique ID as given in the Routing destination ID table.
DID3	31:26	rh	**Destination ID 4** This bit-field indicates the 4th destination CAN node to which the received CAN frame has to be transferred. The destination CAN node is referenced using a unique ID as given in the Routing destination ID table.

20.8.94.3 RAM COBUF CRC computed by DRE

The CAN Output Buffer List is stored in DRE RAM starting from DRE RAM base address + 0x6E0. The number of buffers in the list is 64.

COBUFj_CRC (j=0-63) Offset address: $006E4_H+j*50_H$

RAM COBUF CRC computed by DRE RAMInit value: $XXXX\ XXXX_H$



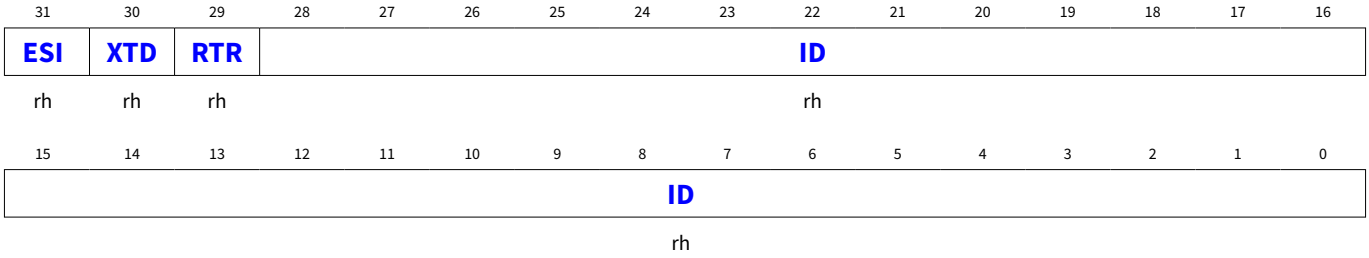
Field	Bits	Type	Description
CRC	15:0	rwh	**CRC** CRC computed by DRE Note: In case of Multi-cast, an intermediate CRC calculated over R0, R1 (except ANMF, RXTS, FIDX and BRS which are considered as zero), safety critical CAN payload (without DID) is stored
0	31:16	r	**Reserved**

20.8.94.4 RAM COBUF register 0

The CAN Output Buffer List is stored in DRE RAM starting from DRE RAM base address + 0x6E0. The number of buffers in the list is 64.

COBUFj_R0 (j=0-63) Offset address: $006E8_H+j*50_H$

RAM COBUF register 0 RAMInit value: $XXXX\ XXXX_H$



Field	Bits	Type	Description
ID	28:0	rh	**CAN Identifier** Standard or extended identifier depending on bit XTD. A standard identifier is stored into ID[28:18].
RTR	29	rh	**Remote Transmission Request** This bit-field indicates that the CAN frame is a remote frame. There are no remote frames in CAN FD format. In case of CAN FD frame (FDF = 1), this bit-field is ignored. 0_B Received frame is a data frame 1_B Received frame is a remote frame
XTD	30	rh	**Extended Identifier** Indicates whether the CAN frame has a standard or extended identifier. 0_B 11-bit standard identifier 1_B 29-bit extended identifier

(table continues...)

(continued)

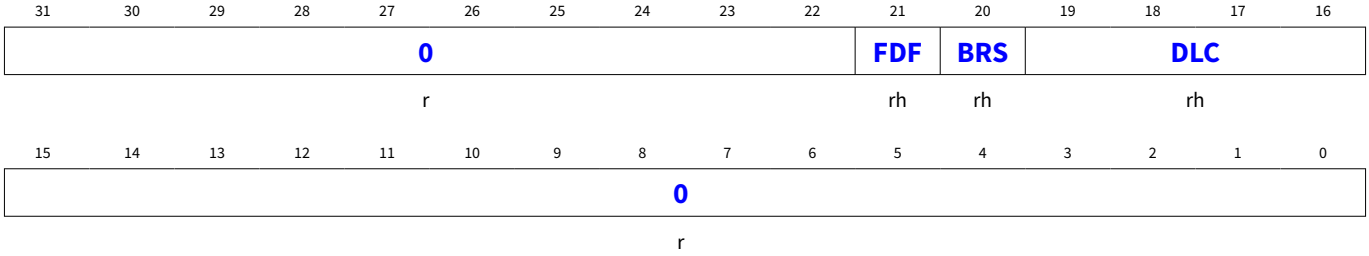
Field	Bits	Type	Description
ESI	31	rh	**Error State Indicator** 0_B Transmitting node is error active 1_B Transmitting node is error passive

20.8.94.5 RAM COBUF register 1

The CAN Output Buffer List is stored in DRE RAM starting from DRE RAM base address + 0x6E0. The number of buffers in the list is 64.

COBUFj_R1 (j=0-63) Offset address: $006EC_H+j*50_H$

RAM COBUF register 1 RAMInit value: $XXXX\ XXXX_H$



Field	Bits	Type	Description
DLC	19:16	rh	**Data Length Code** 0_H CAN + CAN FD: received frame has 0 data bytes … 8_H CAN + CAN FD: received frame has 8 data bytes 9_H CAN FD: received frame has 12 (9*4-24) data bytes CAN: received frame has 8 data bytes … C_H CAN FD: received frame has 24 (12*4-24) data bytes CAN: received frame has 8 data bytes D_H CAN FD: received frame has 32 (13*16-176) data bytes CAN: received frame has 8 data bytes … F_H CAN FD: received frame has 64 (15*16-176) data bytes CAN: received frame has 8 data bytes
BRS	20	rh	**Bit Rate Switch** 0_B Frame received without bit rate switching 1_B Frame received with bit rate switching
FDF	21	rh	**Frame Data Format** 0_B Standard frame format 1_B CAN FD frame format (new DLC-coding and CRC)
0	15:0, 31:22	r	**Reserved** Read as 0; should be written with 0.

20.8.94.6 RAM COBUF data byte m

This register indicates value of CAN payload data byte m.

COBUFj_DB (j=0-63;m=0-63) Offset address: $006F0_H+j*50_H+m$

RAM COBUF data byte m RAMInit value: XX_H

7	6	5	4	3	2	1	0
DB							
rh							

Field	Bits	Type	Description
DB	7:0	rh	**Data Byte m**

20.8.95 CAN Transmit Routing Table RAM interface

20.8.95.1 RAM routing table CAN ID filter configuration

The CAN Transmit Routing Table i elements are stored in DRE RAM starting from DRE RAM base address + 0x1AE0 + i*RTi_CONFIG.NRULES. The number of routing elements within each routing table is configured in RTi_CONFIG.NRULES.

RT_REj_CIDFC (j=0-127) Offset address: $01AE0_H+j*8$

RAM routing table CAN ID filter configuration RAMInit value: $XXXX\ XXXX_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	MODE		CANID2												
r	rw		rw												

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IDS			CANID1												
rw			rw												

Field	Bits	Type	Description
CANID1	12:0	rw	**CAN ID 1** The first CAN identifier used for filter operation. The function of CAN ID 1 depends on the configured "MODE" bit-field. For standard CAN IDs (RREQ_CID.XTD = 0b) only bits 10 to 0 are considered. For extended CAN IDs (RREQ_CID.XTD = 1b) all 13 bits are considered.
IDS	15:13	rw	**ID Shift** This bit-field is used only when RREQ_CID.XTD = 1b. The extended ID is shifted left by this bit-field value and the most significant 13 bits of the shifted CAN ID are used for filtering operation.
CANID2	28:16	rw	**CAN ID 2** The second CAN identifier used for filter operation. The function of CAN ID 2 depends on the configured "MODE" bit-field. For standard CAN IDs (RREQ_CID.XTD = 0b) only bits 10 to 0 are considered. For extended CAN IDs (RREQ_CID.XTD = 1b) all 13 bits are considered.

(table continues...)

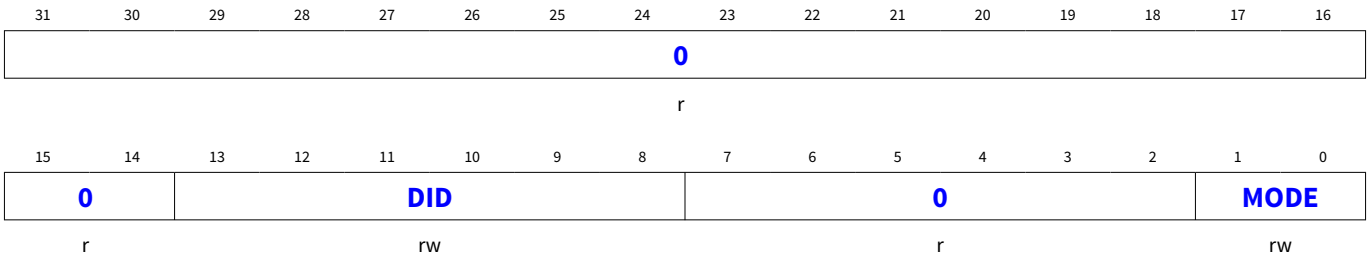
(continued)

Field	Bits	Type	Description
MODE	30:29	rw	**Filter Mode** This bit-field configures the filter mode. 00_B Classical ID Filter "CAN ID 1" is used as the filter ID and "CAN ID 2" is used as mask 01_B Dual ID Filter "CANID1" or "CANID2" is used as the filter ID. No masking is done. 10_B Range ID Filter The range of IDs from "CANID1" till "CANID2" are used as filter IDs. It has to be ensured by the user that "CANID1" < "CANID2" 11_B Reserved Reserved; not to be used
0	31	r	**Reserved** Read as 0; should be written with 0.

20.8.95.2 RAM routing table uni-cast routing

RT_REj_UCR (j=0-127) Offset address: $01AE4_H+j*8$

RAM routing table uni-cast routing RAMInit value: $XXXX\ XXXX_H$



Field	Bits	Type	Description
MODE	1:0	rw	**Routing Rule Mode** It defines the type of the routing rule 0 - Uni-cast rule 1 - Multi-cast rule other - Reserved, considered as Uni-cast rule
DID	13:8	rw	**Destination ID** This bit-field indicates the destination CAN node to which the received CAN frame has to be transferred. The destination CAN node is referenced using a unique ID as given in the Routing destination ID table.
0	7:2, 31:14	r	**Reserved** Read as 0; should be written with 0.

20.8.95.3 RAM routing table multi-cast routing

RT_REj_MCR (j=0-127) Offset address: $01AE4_H+j*8$

RAM routing table multi-cast routing RAMInit value: $XXXX\ XXXX_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DID3						DID2						DID1			
rw						rw						rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DID1		DID0						0						MODE	
rw		rw						r						rw	

Field	Bits	Type	Description
MODE	1:0	rw	**Routing Rule Mode** It defines the type of the routing rule 0 - Uni-cast rule 1 - Multi-cast rule other - Reserved, considered as Uni-cast rule
DID0	13:8	rw	**Destination ID 1** This bit-field indicates the 1st destination CAN node to which the received CAN frame has to be transferred. The destination CAN node is referenced using a unique ID as given in the Routing destination ID table.
DID1	19:14	rw	**Destination ID 2** This bit-field indicates the 2nd destination CAN node to which the received CAN frame has to be transferred. The destination CAN node is referenced using a unique ID as given in the Routing destination ID table.
DID2	25:20	rw	**Destination ID 3** This bit-field indicates the 3rd destination CAN node to which the received CAN frame has to be transferred. The destination CAN node is referenced using a unique ID as given in the Routing destination ID table.
DID3	31:26	rw	**Destination ID 4** This bit-field indicates the 4th destination CAN node to which the received CAN frame has to be transferred. The destination CAN node is referenced using a unique ID as given in the Routing destination ID table.
0	7:2	r	**Reserved** Read as 0; should be written with 0.

20.8.96 Ethernet Address Database RAM interface

20.8.96.1 RAM LETH Tx DMA channel address

The Ethernet address database elements are stored in DRE RAM starting from DRE RAM base address + 0x2AE0.

EAD_LETHj_TXDMA (j=0-3) Offset address: $02AE0_H+j*8$

RAM LETH Tx DMA channel address RAMInit value: $XXXX\ XXXX_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADR															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADR															
rw															

Field	Bits	Type	Description
ADR	31:0	rw	**Address** This bit-field is configured by user with address of the LETH DMA channel Tail Pointer.

20.8.96.2 RAM LETH Rx DMA channel address

The Ethernet address database elements are stored in DRE RAM starting from DRE RAM base address + 0x2AE0.

EAD_LETHj_RXDMA (j=0-3) Offset address: $02AE4_H+j*8$

RAM LETH Rx DMA channel address RAMInit value: $XXXX\ XXXX_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADR															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADR															
rw															

Field	Bits	Type	Description
ADR	31:0	rw	**Address** This bit-field is configured by user with address of the LETH DMA channel Tail Pointer.

20.8.96.3 RAM GETH Tx DMA channel address

The Ethernet address database elements are stored in DRE RAM starting from DRE RAM base address + 0x2AE0

EAD_GETHj_TXDMA (j=0-1) Offset address: $02B00_H+j*8$

RAM GETH Tx DMA channel address RAMInit value: $XXXX\ XXXX_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADR															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADR															
rw															

Field	Bits	Type	Description
ADR	31:0	rw	**Address** This bit-field is configured by user with address of the GETH DMA channel Tail Pointer.

20.8.96.4 RAM GETH Rx DMA channel address

The Ethernet address database elements are stored in DRE RAM starting from DRE RAM base address + 0x2AE0

EAD_GETHj_RXDMA (j=0-1) Offset address: $02B04_H$+j*8

RAM GETH Rx DMA channel address RAMInit value: $XXXX\ XXXX_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADR															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADR															
rw															

Field	Bits	Type	Description
ADR	31:0	rw	**Address** This bit-field is configured by user with address of the GETH DMA channel Tail Pointer.

20.8.97 Ethernet descriptors RAM interface

20.8.97.1 RAM TDESC word 0 read format

The Ethernet Transmit descriptor list j is stored in DRE RAM at DRE RAM base address + 0x3140 + j*0xC80
There are 4 descriptors (each of which is made up of four 32-bit words) within each descriptor list.

TDESCi_RD0 (i=0-3) Offset address: 03140_H+i*10_H

RAM TDESC word 0 read format RAMInit value: $XXXX\ XXXX_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BUF1AP															
rwh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BUF1AP															
rwh															

Field	Bits	Type	Description
BUF1AP	31:0	rwh	**Ethernet Buffer Address Pointer 1** These bits indicate the physical address of the frame in EOBUF or EIBUF. Bit FD shall be set.

20.8.97.2 RAM TDESC word 0 Write-back format

The Ethernet Transmit descriptor list j is stored in DRE RAM at DRE RAM base address + 0x3140 + j*0xC80
There are 4 descriptors (each of which is made up of four 32-bit words) within each descriptor list.

TDESCi_WR0 (i=0-3) Offset address: 03140_H+i*10_H

RAM TDESC word 0 Write-back format RAMInit value: $XXXX\ XXXX_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															
r															

Field	Bits	Type	Description
0	31:0	r	**Reserved** Read as 0; should be written as 0

20.8.97.3 RAM TDESC word 1 read format

The Ethernet Transmit descriptor list j is stored in DRE RAM at DRE RAM base address + 0x3140 + j*0xC80
There are 4 descriptors (each of which is made up of four 32-bit words) within each descriptor list.

TDESCi_RD1 (i=0-3) Offset address: 03144_H+i*10_H

RAM TDESC word 1 read format RAMInit value: $XXXX\ XXXX_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BUF2AP															
rwh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BUF2AP															
rwh															

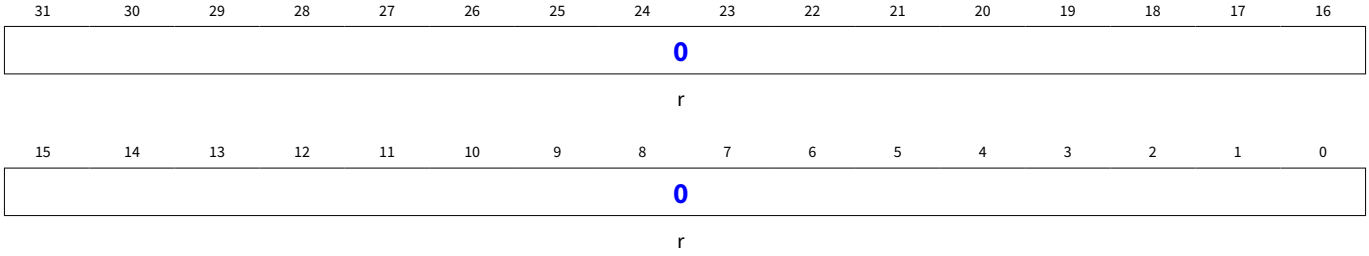
Field	Bits	Type	Description
BUF2AP	31:0	rwh	**Ethernet Buffer Address Pointer 2** Shall be 0 as there is no segmentation of frame

20.8.97.4 RAM TDESC word 1 Write-back format

The Ethernet Transmit descriptor list j is stored in DRE RAM at DRE RAM base address + 0x3140 + j*0xC80
There are 4 descriptors (each of which is made up of four 32-bit words) within each descriptor list.

TDESCi_WR1 (i=0-3) Offset address: 03144_H+i*10_H

RAM TDESC word 1 Write-back format RAMInit value: $XXXX\ XXXX_H$



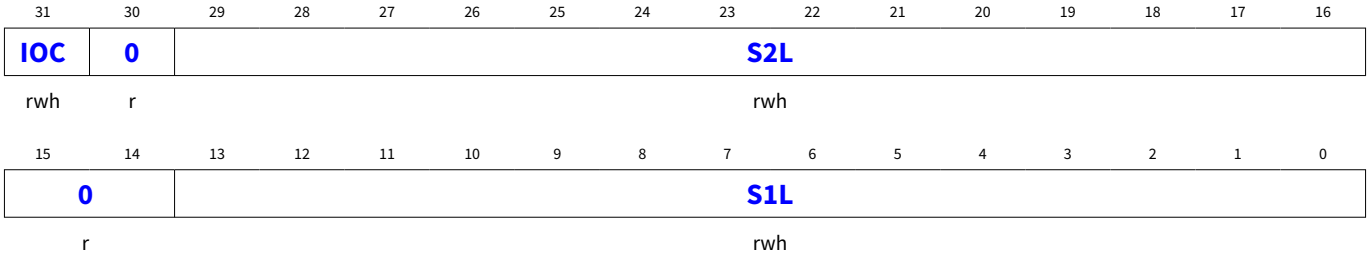
Field	Bits	Type	Description
0	31:0	r	**Reserved** Read as 0; should be written as 0

20.8.97.5 RAM TDESC word 2 read format

The Ethernet Transmit descriptor list j is stored in DRE RAM at DRE RAM base address + 0x3140 + j*0xC80
There are 4 descriptors (each of which is made up of four 32-bit words) within each descriptor list.

TDESCi_RD2 (i=0-3) Offset address: 03148_H+i*10_H

RAM TDESC word 2 read format RAMInit value: $XXXX\ XXXX_H$



Field	Bits	Type	Description
S1L	13:0	rwh	**Size of buffer 1** Indicates the size of the frame EOBUF/EIBUF (in bytes)
S2L	29:16	rwh	**Size of buffer 2** Shall be 0 as there is no second buffer in DRE

(table continues...)

(continued)

Field	Bits	Type	Description
IOC	31	rwh	**Interrupt on Completion** This bit sets the TI bit in the DMA_CH(#i)_Status register after the present packet has been transmitted.
0	15:14, 30	r	**Reserved** Read as 0; should be written as 0

20.8.97.6 RAM TDESC word 2 Write-back format

The Ethernet Transmit descriptor list j is stored in DRE RAM at DRE RAM base address + 0x3140 + j*0xC80
There are 4 descriptors (each of which is made up of four 32-bit words) within each descriptor list.

TDESCi_WR2 (i=0-3) Offset address: 03148_H+i*10_H

RAM TDESC word 2 Write-back format RAMInit value: $XXXX\ XXXX_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															
r															

Field	Bits	Type	Description
0	31:0	r	**Reserved** Read as 0; should be written as 0

20.8.97.7 RAM TDESC word 3 read format

The Ethernet Transmit descriptor list j is stored in DRE RAM at DRE RAM base address + 0x3140 + j*0xC80
There are 4 descriptors (each of which is made up of four 32-bit words) within each descriptor list.

TDESCi_RD3 (i=0-3) Offset address: $0314C_H+i*10_H$

RAM TDESC word 3 read format RAMInit value: $XXXX\ XXXX_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OWN	CTXT	FD	LD	CPC		SAIC			SLOTNUM				0	CIC	
rwh	rwh	rwh	rwh	rwh		rwh			rwh				r	rwh	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	FL														
r	rwh														

Field	Bits	Type	Description
FL	14:0	rwh	**Packet Length** Length of the packet to be transmitted in bytes. This equals the total length of the packet to be transmitted: Header length + Payload length

(table continues...)

(continued)

Field	Bits	Type	Description
CIC	17:16	rwh	**Checksum Insertion control** Checksum insertion is disabled. Shall be set to 0
SLOTNUM	22:19	rwh	**Slot Number** Holds the static value configured in TETHDLi_CTRL.SLOTNUM. These bits indicate the slot interval in which the data should be fetched from the corresponding buffers addressed by D0 or D1. When the Transmit descriptor is fetched, the DMA compares the slot number value in this field with the slot interval maintained in the RSN field DMA_CH(#i)_Slot_Function_Control_Status. It fetches the data from the buffers only if a value matches. These bits are valid only for the AV channels.
SAIC	25:23	rwh	**SA insertion control** Holds the static value configured in TETHDLi_CTRL.SAIC These bits request the MAC to add or replace the Source Address field in the Ethernet packet with the value given in the MAC Address 0 register. If the Source Address field is modified in a packet, the MAC automatically recalculates and replaces the CRC bytes. Bit 25 specifies the MAC Address Register (1 or 0) value that is used for Source Address insertion or replacement. The following list describes the values of bits [24:23]: • 2'b00: Do not include the source address • 2'b01: Reserved • 2'b10: Replace the source address. For reliable transmission, the application must provide frames with source addresses • 2'b11: Reserved
CPC	27:26	rwh	**CRC Pad Control** This field controls the CRC and Pad Insertion for TX packet. This field is valid only when the first descriptor bit (TDESCi_RD3[29]) is set • 2'b10 Disable CRC Insertion: The MAC does not append the CRC at the end of the transmitted packet. The application should ensure that the padding and CRC bytes are present in the packet being transferred from the Transmit Buffer • 2'b00 CRC and Pad Insertion: The MAC appends the cyclic redundancy check (CRC) at the end of the transmitted packet of length greater than or equal to 60 bytes. The MAC automatically adds padding and CRC to a frame shorter than 60 bytes In case of Ethernet frames to be forwarded from EIBUF, this bit is configured based on the RETHDLi_CTRL.FCS bit. If FCS = 0, then the corresponding TDESCi_RD3.CPC bit-field is set to b00 (Enable CRC and PAD insertion) or else TDESCi_RD3.CPC bit-field is set to b10 (Disable CRC insertion) This field shall be set to 0 in case the Ethernet frame is in EOBUF
LD	28	rwh	**Last Descriptor** When this bit is set, it indicates that the buffer contains last segment of the packet

(table continues...)

(continued)

Field	Bits	Type	Description
FD	29	rwh	**First Descriptor** When this bit is set, it indicates that the buffer contains first segment of the packet.
CTXT	30	rwh	**Context Type** Shall be set to 0 for Normal descriptor
OWN	31	rwh	**OWN bit** When this bit is set, it indicates that the DMA owns the descriptor. When this bit is reset, it indicates that the DRE owns the descriptor. The DMA clears this bit after it completes the transfer of data given in the associated buffers.
0	15, 18	r	**Reserved** Read as 0; should be written as 0

20.8.97.8 RAM TDESC word 3 Write-back format for GETH

The Ethernet Transmit descriptor list j is stored in DRE RAM at DRE RAM base address + 0x3140 + j*0xC80
There are 4 descriptors (each of which is made up of four 32-bit words) within each descriptor list.

TDESCi_WR3G (i=0-3) Offset address: $0314C_H+i*10_H$

RAM TDESC word 3 Write-back format for GETH RAMInit value: $XXXX\ XXXX_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OWN	CTXT	FD	LD	DERR	0										
rwh	rwh	rwh	rwh	rwh	r										

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															
r															

Field	Bits	Type	Description
DERR	27	rwh	**Descriptor Error** When this bit is set, it indicates that the descriptor content is incorrect.
LD	28	rwh	**Last Descriptor** When this bit is set, it indicates that the buffer contains last segment of the packet
FD	29	rwh	**First Descriptor** When this bit is set, it indicates that the buffer contains first segment of the packet.
CTXT	30	rwh	**Context Type** Shall be set to 0 for Normal descriptor

(table continues...)

(continued)

Field	Bits	Type	Description
OWN	31	rwh	**OWN bit** When this bit is set, it indicates that the DMA owns the descriptor. When this bit is reset, it indicates that the DRE owns the descriptor. The DMA clears this bit after it completes the transfer of data given in the associated buffers.
0	26:0	r	**Reserved** Read as 0; should be written as 0

20.8.97.9 RAM TDESC word 3 Write-back format for LETH

The Ethernet Transmit descriptor list j is stored in DRE RAM at DRE RAM base address + 0x3140 + j*0xC80
There are 4 descriptors (each of which is made up of four 32-bit words) within each descriptor list.

TDESCi_WR3L (i=0-3) Offset address: $0314C_H+i*10_H$

RAM TDESC word 3 Write-back format for LETH RAMInit value: $XXXX\ XXXX_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OWN	CTXT	FD	LD	0				DE	0					TTSS	EUE
rwh	rwh	rwh	rwh	r				rwh	r					rwh	rwh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ES	JT	FF	PCE	LOC	NC	LC	EC	CC				ED	UF	DB	IHE
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh				rwh	rwh	rwh	rwh

Field	Bits	Type	Description
IHE	0	rwh	**IP Header Error** When IP Header Error is set, this bit indicates that the Checksum Offload engine detected an IP header error. This bit is valid only when Tx Checksum Offload is enabled. Otherwise, it is reserved. If COE detects an IP header error, it still inserts an IPv4 header checksum if the Ethernet Type field indicates an IPv4 payload. In full duplex mode, when EST/Qbv is enabled and this bit is set, it indicates the frame drop status due to Frame Size error or Schedule Error.
DB	1	rwh	**Deferred Bit** This bit indicates that the MAC deferred before transmitting because of presence of carrier. This bit is valid only in the half-duplex mode.
UF	2	rwh	**Underflow Error** This bit indicates that the MAC aborted the packet because the data arrived late from the system memory. The underflow error can occur because of either of the following conditions: • The DMA encountered an empty Transmit Buffer while transmitting the packet • The application filled the MTL Tx FIFO slower than the MAC transmit rate The transmission process enters the suspended state and sets the underflow bit corresponding to a queue in the MTL_Interrupt_Status register.

(table continues...)

(continued)

Field	Bits	Type	Description
ED	3	rwh	**Excessive Deferral** This bit indicates that the transmission ended because of excessive deferral of over 24,288 bit times (155,680 bits times in 1000 Mbps mode or Jumbo Packet enabled mode) if DC bit is set in the MAC_Configuration register. When TBS is enabled in full duplex mode and this bit is set, it indicates that the frame has been dropped after the expiry time has reached.
CC	7:4	rwh	**Collision Count** This 4-bit counter value indicates the number of collisions occurred before the packet was transmitted. The count is not valid when the EC bit is set.
EC	8	rwh	**Excessive Collision** This bit indicates that the transmission was aborted after 16 successive collisions while attempting to transmit the current packet. If the DR bit is set in the MAC_Configuration register, this bit is set after first collision and the transmission of the packet is aborted.
LC	9	rwh	**Late Collision** This bit indicates that packet transmission was aborted because a collision occurred after the collision window (64 byte times including Preamble in MII mode and 512 byte times including Preamble and Carrier Extension in GMII mode). This bit is not valid if Underflow Error is set.
NC	10	rwh	**No Carrier** This bit indicates that the carrier sense signal form the PHY was not asserted during transmission.
LOC	11	rwh	**Loss of Carrier** This bit indicates that Loss of Carrier occurred during packet transmission (that is, the gmii_crs_i signal was inactive for one or more transmit clock periods during packet transmission). This is valid only for the packets transmitted without collision and when the MAC operates in the half-duplex mode.
PCE	12	rwh	**Payload Checksum Error** This bit indicates that the Checksum Offload engine had a failure and did not insert any checksum into the encapsulated TCP, UDP, or ICMP payload. This failure can be either because of insufficient bytes, as indicated by the Payload Length field of the IP Header or the MTL starting to forward the packet to the MAC transmitter in Store-and-Forward mode without the checksum having been calculated yet. This second error condition only occurs when the Transmit FIFO depth is less than the length of the Ethernet packet being transmitted to avoid deadlock, the MTL starts forwarding the packet when the FIFO is full, even in the store-and-forward mode. This error can also occur when Bus Error is detected during packet transfer. When the Full Checksum Offload engine is not enabled, this bit is reserved.
FF	13	rwh	**Packet Flushed** This bit indicates that the DMA or MTL flushed the packet because of a software flush command given by the CPU.

(table continues...)

(continued)

Field	Bits	Type	Description
JT	14	rwh	**Jabber Timeout** This bit indicates that the MAC transmitter has experienced a jabber time-out. This bit is set only when the JD bit of the MAC_Configuration register is not set.
ES	15	rwh	**Error Summary** This bit indicates the logical OR of the following bits: • TDES3[0]: IP Header Error • TDES3[14]: Jabber Timeout • TDES3[13]: Packet Flush • TDES3[12]: Payload Checksum Error • TDES3[11]: Loss of Carrier • TDES3[10]: No Carrier • TDES3[9]: Late Collision • TDES3[8]: Excessive Collision • TDES3[3]: Excessive Deferral • TDES3[2]: Underflow Error This bit is also set when EUE (bit 16) is set.
EUE	16	rwh	**ECC Uncorrectable Error Status** Indicates the ECC uncorrectable error in the TSO memory. Uncorrectable error in Transmit FIFO memory is reported with (Bit 13) FF = 1. This is because, all such packets are flushed by LETH.
TTSS	17	rwh	**Tx Timestamp Status** This status bit indicates that a timestamp has been captured for the corresponding transmit packet. When this bit is set, TDES0 and TDES1 have timestamp values that were captured for the Transmit packet. This field is valid only when the Last Segment control bit (TDES3 [28]) in a descriptor is set. This bit is valid only when IEEE 1588 timestamping feature is enabled; otherwise, it is reserved.
DE	23	rwh	**Descriptor Error** When this bit is set, it indicates that the descriptor content is incorrect. The DMA sets this bit during Write-back while closing the descriptor.
LD	28	rwh	**Last Descriptor** When this bit is set, it indicates that the buffer contains last segment of the packet
FD	29	rwh	**First Descriptor** When this bit is set, it indicates that the buffer contains first segment of the packet.
CTXT	30	rwh	**Context Type** Shall be set to 0 for Normal descriptor
OWN	31	rwh	**OWN bit** When this bit is set, it indicates that the DMA owns the descriptor. When this bit is reset, it indicates that the DRE owns the descriptor. The DMA clears this bit after it completes the transfer of data given in the associated buffers.

(table continues...)

(continued)

Field	Bits	Type	Description
0	22:18, 27:24	r	**Reserved** Read as 0; should be written as 0

20.8.97.10 RAM RDESC word 0 read format

The Ethernet Receive descriptor list j is stored in DRE RAM at DRE RAM base address + 0x3180 + j*0xC80
There are 4 descriptors (each of which is made up of four 32-bit words) within each descriptor list.

RDESCi_RD0 (i=0-3) Offset address: 03180_H+i*10_H

RAM RDESC word 0 read format RAMInit value: $XXXX\ XXXX_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BUF1AP															
rwh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BUF1AP															
rwh															

Field	Bits	Type	Description
BUF1AP	31:0	rwh	**Ethernet input buffer address pointer** These bits indicate the physical address EIBUF to which the input Ethernet frame is stored.

20.8.97.11 RAM RDESC word 0 Write-back format Non Tunneled frames (also LETH)

The Ethernet Receive descriptor list j is stored in DRE RAM at DRE RAM base address + 0x3180 + j*0xC80
There are 4 descriptors (each of which is made up of four 32-bit words) within each descriptor list.

RDESCi_WR0NT (i=0-3) Offset address: 03180_H+i*10_H

RAM RDESC word 0 Write-back format Non Tunneled frames [also LETH) RAMInit value: $XXXX\ XXXX_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
IVT															
rwh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OVT															
rwh															

Field	Bits	Type	Description
OVT	15:0	rwh	**Outer VLAN Tag or External Lookup Result Data** This bit-field is not used by the DRE

(table continues...)

(continued)

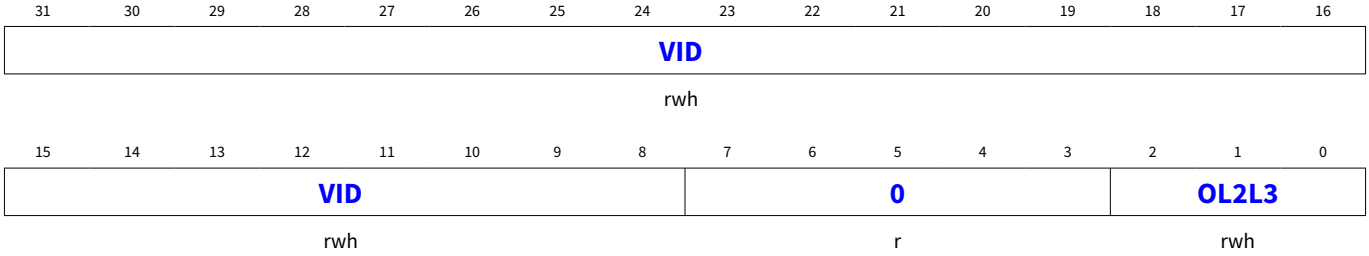
Field	Bits	Type	Description
IVT	31:16	rwh	**Inner VLAN Tag or External Lookup Result Data** This bit-field is not used by the DRE

20.8.97.12 RAM RDESC word 0 Write-back format Tunneled frames

The Ethernet Receive descriptor list j is stored in DRE RAM at DRE RAM base address + 0x3180 + j*0xC80
There are 4 descriptors (each of which is made up of four 32-bit words) within each descriptor list.

RDESCi_WR0T (i=0-3) Offset address: 03180_H+i*10_H

RAM RDESC word 0 Write-back format Tunneled frames RAMInit value: $XXXX\ XXXX_H$



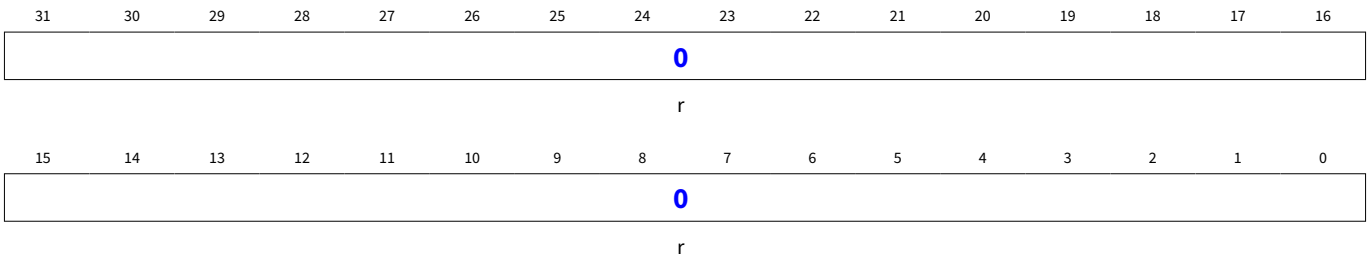
Field	Bits	Type	Description
OL2L3	2:0	rwh	**Outer L2 L3 Type** This bit-field is not used by the DRE
VID	31:8	rwh	**VNID or VSID** This bit-field is not used by the DRE
0	7:3	r	**Reserved** Read as 0; should be written as 0

20.8.97.13 RAM RDESC word 1 read format

The Ethernet Receive descriptor list j is stored in DRE RAM at DRE RAM base address + 0x3180 + j*0xC80
There are 4 descriptors (each of which is made up of four 32-bit words) within each descriptor list.

RDESCi_RD1 (i=0-3) Offset address: 03184_H+i*10_H

RAM RDESC word 1 read format RAMInit value: $XXXX\ XXXX_H$



Field	Bits	Type	Description
0	31:0	r	**Reserved** Read as 0; should be written as 0

20.8.97.14 RAM RDESC word 1 Write-back format GETH

The Ethernet Receive descriptor list j is stored in DRE RAM at DRE RAM base address + 0x3180 + j*0xC80
There are 4 descriptors (each of which is made up of four 32-bit words) within each descriptor list.

RDESCi_WR1G (i=0-3) Offset address: 03184_H+i*10_H

RAM RDESC word 1 Write-back format GETH RAMInit value: $XXXX\ XXXX_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FRPLI															
rwh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FRPLI															
rwh															

Field	Bits	Type	Description
FRPLI	31:0	rwh	**RSS HASH/Flexible Recieve Parser Last Instruction** This field contains the FRP Last Instruction number where FRP result is generated. This bit-field is used by the Forwarding engine to generate the GETH Forward ID

20.8.97.15 RAM RDESC word 1 Write-back format LETH

The Ethernet Receive descriptor list j is stored in DRE RAM at DRE RAM base address + 0x3180 + j*0xC80
There are 4 descriptors (each of which is made up of four 32-bit words) within each descriptor list.

RDESCi_WR1L (i=0-3) Offset address: 03184_H+i*10_H

RAM RDESC word 1 Write-back format LETH RAMInit value: $XXXX\ XXXX_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OPC															
rwh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TD	TSA	PV	PFT	PMT				IPCE	IPCB	IPV6	IPV4	IPHE	PT		
rwh	rwh	rwh	rwh	rwh				rwh	rwh	rwh	rwh	rwh	rwh		

Field	Bits	Type	Description
PT	2:0	rwh	**Payload type** These bits indicate the type of payload encapsulated in the IP datagram processed by the Receive Checksum Offload Engine (COE): • 3'b000: Unknown type or IP/AV payload not processed • 3'b001: UDP • 3'b010: TCP • 3'b011: ICMP • 3'b110: AV Tagged Data Packet • 3'b111: AV Tagged Control Packet • 3'b101: AV Untagged Control Packet • 3'b100: IGMP if IPV4 Header Present bit is set else DCB (LLDP) Control Packet If the COE does not process the payload of an IP datagram because there is an IP header error or fragmented IP, it sets these bits to 3'b000.
IPHE	3	rwh	**IP Header Error** When this bit is set, it indicates either of the following: • The 16-bit IPv4 header checksum calculated by the MAC does not match the received checksum bytes. • The IP datagram version is not consistent with the Ethernet Type value. • Ethernet packet does not have the expected number of IP header bytes. This bit is valid when either bit 5 or bit 4 is set. This bit is available when you select the Enable Receive TCP/IP Checksum Check feature.
IPV4	4	rwh	**IPV4 Header Present** This bit indicates that an IPV4 header is detected. When the SPH bit of RDESCi_WR3 is set, the IPV4 header is available in the header buffer area to which RDESCi_WR0 is pointing.
IPV6	5	rwh	**IPV6 Header Present** This bit indicates that an IPV6 header is detected. When the Enable Split Header Feature option is selected and the SPH bit of Control Register of a channel is set, the IPV6 header is available in the header buffer area to which RDES0 is pointing.
IPCB	6	rwh	**IP Checksum Bypassed** This bit indicates that the checksum offload engine is bypassed. This bit is available when you select the Enable Receive TCP/IP Checksum Check feature.

(table continues...)

(continued)

Field	Bits	Type	Description
IPCE	7	rwh	**IP Payload Error** When this bit is set, it indicates either of the following: • The 16-bit IP payload checksum (that is, the TCP, UDP, or ICMP checksum) calculated by the MAC does not match the corresponding checksum field in the received segment. • The TCP, UDP, or ICMP segment length does not match the payload length value in the IP Header field. • The TCP, UDP, or ICMP segment length is less than minimum allowed segment length for TCP, UDP, or ICMP. Bit 15 (ES) of RDESCi_WR3 is not set when this bit is set.
PMT	11:8	rwh	**PTP Message Type** These bits are encoded to give the type of the message received: • 0000: No PTP message received • 0001: SYNC (all clock types) • 0010: Follow_Up (all clock types) • 0011: Delay_Req (all clock types) • 0100: Delay_Resp (all clock types) • 0101: Pdelay_Req (in peer-to-peer transparent clock) • 0110: Pdelay_Resp (in peer-to-peer transparent clock) • 0111: Pdelay_Resp_Follow_Up (in peer-to-peer transparent clock) • 1000: Announce • 1001: Management • 1010: Signaling • 1011 to 1110: Reserved • 1111: PTP packet with Reserved message type These bits are available only when you select the Timestamp feature.
PFT	12	rwh	**PTP Packet Type** This bit indicates that the PTP message is sent directly over Ethernet. This bit is available only when you select the Timestamp feature. Otherwise, this bit is reserved.
PV	13	rwh	**PTP Version** This bit indicates that the received PTP message has the IEEE 1588 version 2 format. When this bit is reset, it indicates the IEEE 1588 version 1 format. This bit is available only when you select the Timestamp feature. Otherwise, this bit is reserved.
TSA	14	rwh	**Timestamp Available** When Timestamp is present, this bit indicates that the timestamp value is available in a context descriptor word 2 (RDESCi_WR2) and word 1 (RDESCi_WR1). This is valid only when the Last Descriptor bit (RDESCi_WR3 [28]) is set. The context descriptor is written in the next descriptor just after the last normal descriptor for a packet.

(table continues...)

(continued)

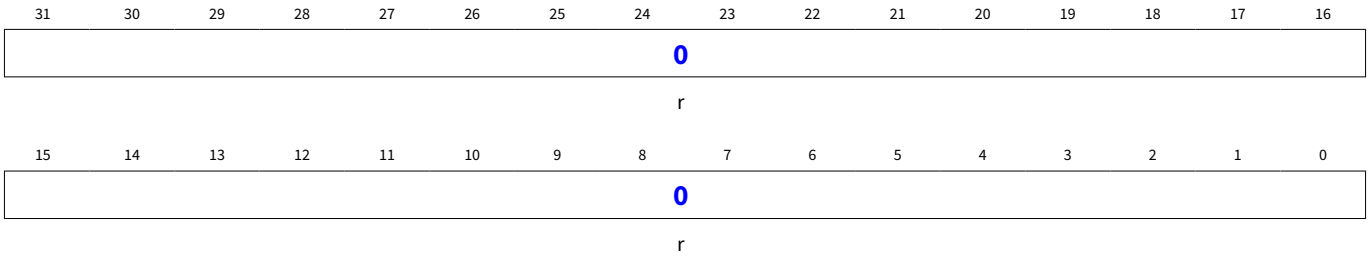
Field	Bits	Type	Description
TD	15	rwh	**Timestamp Dropped** This bit indicates that the timestamp was captured for this packet but it got dropped in the MTL Rx FIFO because of overflow. This bit is available only when you select the Timestamp feature. Otherwise, this bit is reserved.
OPC	31:16	rwh	**OAM Sub-Type Code, or MAC Control Packet opcode** • OAM Sub-Type Code If bits[18:16] of RDESCi_WR3 are set to 3'b111, this field contains the OAM sub-type and code fields. • MAC Control Packet opcode If bits[18:16] of RDESCi_WR3 are set to 3'b110, this field contains the MAC Control packet opcode field.

20.8.97.16 RAM RDESC word 2 read format

The Ethernet Receive descriptor list j is stored in DRE RAM at DRE RAM base address + 0x3180 + j*0xC80
There are 4 descriptors (each of which is made up of four 32-bit words) within each descriptor list.

RDESCi_RD2 (i=0-3) Offset address: 03188_H+i*10_H

RAM RDESC word 2 read format RAMInit value: $XXXX\ XXXX_H$



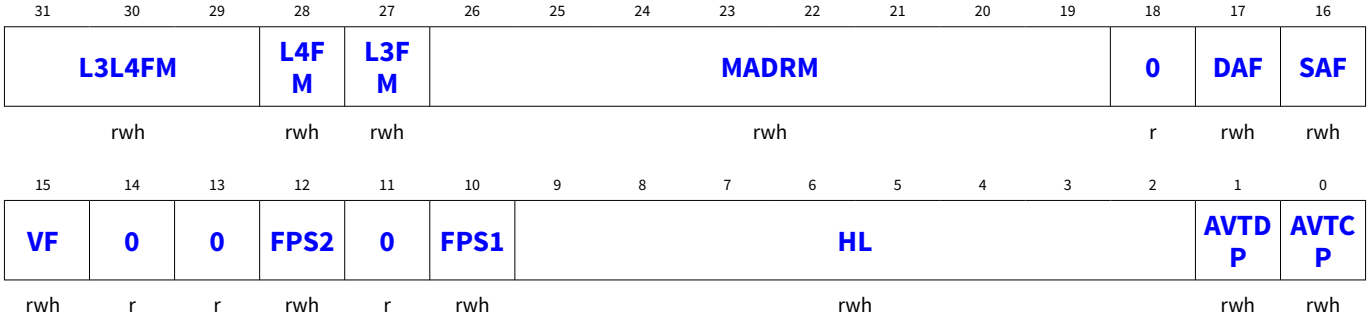
Field	Bits	Type	Description
0	31:0	r	**Reserved** Read as 0; should be written as 0

20.8.97.17 RAM RDESC word 2 Write-back format GETH

The Ethernet Receive descriptor list j is stored in DRE RAM at DRE RAM base address + 0x3180 + j*0xC80
There are 4 descriptors (each of which is made up of four 32-bit words) within each descriptor list.

RDESCi_WR2G (i=0-3) Offset address: 03188_H+i*10_H

RAM RDESC word 2 Write-back format GETH RAMInit value: $XXXX\ XXXX_H$



Field	Bits	Type	Description
AVTCP	0	rwh	**AV Tagged Control Packet** When L34T=0b0000, this bit is set to 1 when AV Tagged Control packet is received. When L34T is non-zero, this bit is HL[0] for packet split by the MAC at L3 or L4 header. Note: The DRE does not support split header feature. The DRE identifies the frame to be AV tagged when AVTDP or AVTCP is set irrespective of L34T value.
AVTDP	1	rwh	**AV Tagged Data Packet** When L34T=0b0000, this bit is set to 1 when AV Tagged Data packet is received. When L34T is non-zero, this bit is HL[1] for packet split by the MAC at L3 or L4 header Note: The DRE does not support split header feature. The DRE identifies the frame to be AV tagged when AVTDP or AVTCP is set irrespective of L34T value.
HL	9:2	rwh	**L3/L4 Header Length** This field contains the length of the header of the packet split by the MAC at L3 or L4 header boundary as identified by the MAC receiver. This field is valid only when the first descriptor bit is set (FD = 1). The header data is written to the Buffer 1 address of corresponding descriptor. If header length is zero, it implies that the MAC did not identify and split the header and both Buffer 1 and Buffer 2 are used for storing the packet.
FPS1	10	rwh	**Flexible Receive Parser bit 1**
FPS2	12	rwh	**Flexible Receive Parser bit 2**
VF	15	rwh	**VLAN Filter Status** When this bit is set, it indicates that the VLAN Tag of received packet passed the VLAN filter. For a tunneled packet, only the outer Ethernet header fields are used for filtering
SAF	16	rwh	**SA Address Filter Fail** When this bit is set, it indicates that the packet failed the SA Filter in the MAC. For a tunneled packet, only the outer Ethernet header fields are used for filtering
DAF	17	rwh	**Destination Address Filter Fail** When this bit is set, it indicates that the packet failed the DA Filter in the MAC. For a tunneled packet, only the outer Ethernet header fields are used for filtering.

(table continues...)

(continued)

Field	Bits	Type	Description
MADRM	26:19	rwh	**MAC Address Match or Hash Value** This field contains the MAC address register number that matched the Destination address of the received packet. This field is valid only if the DAF bit is reset. This bit-field is used by the DRE to generate the forward ID.
L3FM	27	rwh	**Layer 3 Filter Match** When this bit is set, it indicates that the received packet matches one of the enabled Layer 3 IP Address fields. This status is given only when one of the following conditions is true: • All enabled Layer 3 fields match and all enabled Layer 4 fields are bypassed • All enabled filter fields match When more than one filter matches, this bit gives the layer 3 filter status of filter indicated by bits[31:29]. For a tunneled packet, only the outer L3-L4 header fields are used for filter comparison
L4FM	28	rwh	**Layer 4 Filter Match** When this bit is set, it indicates that the received packet matches one of the enabled Layer 4 Port Number fields. This status is given only when one of the following conditions is true: • Layer 3 fields are not enabled and all enabled Layer 4 fields match • All enabled Layer 3 and Layer 4 filter fields match When more than one filter matches, this bit gives the layer 4 filter status of filter indicated by bits[31:29] For a tunneled packet, only the outer L3-L4 header fields are used for filter comparison.
L3L4FM	31:29	rwh	**Layer 3 and Layer 4 Filter Number Matched** This bit-field is used by the DRE to generate the forward ID. These bits indicate the number of the Layer 3 and Layer 4 Filter that matched the received packet: • 000: Filter 0 • 001: Filter 1 • 010: Filter 2 • 011: Filter 3 • 100: Filter 4 • 101: Filter 5 • 110: Filter 6 • 111: Filter 7 This field is valid only when bit 28 or bit 27 is set high. When more than one filter matches, these bits give the number of lowest filter. For a tunneled packet, only the outer L3-L4 header fields are used for comparison

(table continues...)

(continued)

Field	Bits	Type	Description
0	11, 13, 14, 18	r	**Reserved**

20.8.97.18 RAM RDESC word 2 Write-back format LETH

The Ethernet Receive descriptor list j is stored in DRE RAM at DRE RAM base address + 0x3180 + j*0xC80
There are 4 descriptors (each of which is made up of four 32-bit words) within each descriptor list.
The LSB 8 bits of this RAM word also holds the FRPLI used to generate Forward ID

RDESCi_WR2L (i=0-3) Offset address: 03188_H+i*10_H

RAM RDESC word 2 Write-back format LETH RAMInit value: $XXXX\ XXXX_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
L3L4FM			L4FM	L3FM	MADRM								HF	DAF	SAF
rwh			rwh	rwh	rwh								rwh	rwh	rwh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OTS	ITS	0			ARPNR	HL									
rwh	rwh	r			rwh	rwh									

Field	Bits	Type	Description
HL	9:0	rwh	**L3/L4 Header Length** This field contains the length of the header of the packet split by the MAC at L3 or L4 header boundary as identified by the MAC receiver. This field is valid only when the first descriptor bit is set (FD = 1). The header data is written to the Buffer 1 address of corresponding descriptor. If header length is zero, this field is not valid. It implies that the MAC did not identify and split the header. This field is valid when the Enable Split Header Feature option is selected.
ARPNR	10	rwh	**ARP Reply Not Generated** When this bit is set, it indicates that the MAC did not generate the ARP Reply for received ARP Request packet. This bit is set when the MAC is busy transmitting ARP reply to earlier ARP request (only one ARP request is processed at a time). This bit is reserved when the Enable IPv4 ARP Offload option is not selected.
ITS	14	rwh	**Inner VLAN Tag Filter Status** This bit is valid only when DWC_EQOS_ERVFE is enabled. This bit is valid only for Double VLAN Tagged frames, when Double VLAN Processing is enabled. For more information, see the Filter Status topic of LETH.

(table continues...)

(continued)

Field	Bits	Type	Description
OTS	15	rwh	**VLAN Filter Status** When set, this bit indicates that the VLAN Tag of the received packed passed the VLAN filter. This bit is valid only when DWC_EQOS_ERVFE is not enabled. If DWC_EQOS_ERVFE is enabled, the bit is redefined as Outer VLAN Tag Filter Status (OTS). This bit is valid for both Single and Double VLAN Tagged frames
SAF	16	rwh	**SA Filter Fail or Packet dropped** SA Address Filter Fail When Flexible RX Parser is disabled, and this bit is set, it indicates that the packet failed the SA Filter in the MAC. When Flexible RX Parser is enabled, this bit is set to indicate that the packet is dropped (RXPD) by the parser. Note: When this bit is set, ES bit of RDESCi_WR3 is also set.
DAF	17	rwh	**DA Filter Fail or Packet parsing incomplete** Destination Address Filter Fail When Flexible RX Parser is disabled, and this bit is set, it indicates that the packet failed the DA Filter in the MAC. When Flexible RX Parser is enabled, this bit is set to indicate that the packet parsing is incomplete (RXPI) due to ECC error. Note: When this bit is set, ES bit of RDESCi_WR3 is also set.
HF	18	rwh	**Hash Filter Status** When this bit is set, it indicates that the packet passed the MAC address hash filter. Bits[26:19] indicate the hash value. Note: This status is not available when Flexible RX Parser is enabled.
MADRM	26:19	rwh	**MAC Address Match or Hash Value** When the HF bit is reset, this field contains the MAC address register number that matched the Destination address of the received packet. This field is valid only if the DAF bit is reset. When the HF bit is set, this field contains the hash value computed by the MAC. A packet passes the hash filter when the bit corresponding to the hash value is set in the hash filter register. Note: This status is not available when Flexible RX Parser is enabled.
L3FM	27	rwh	**Layer 3 Filter Match** When this bit is set, it indicates that the received packet matches one of the enabled Layer 3 IP Address fields. This status is given only when one of the following conditions is true: • All enabled Layer 3 fields match and all enabled Layer 4 fields are bypassed • All enabled filter fields match When more than one filter matches, this bit gives the layer 3 filter status of filter indicated by bits[31:29]. Note: This status is not available when Flexible RX Parser is enabled.

(table continues...)

(continued)

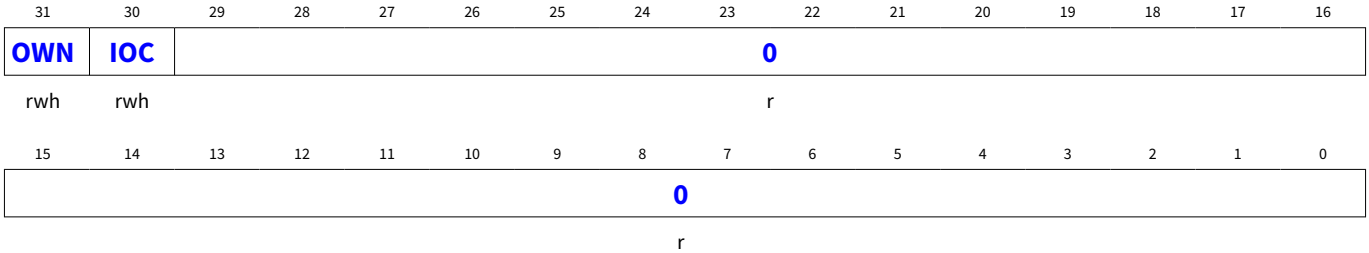
Field	Bits	Type	Description
L4FM	28	rwh	**Layer 4 Filter Match** When this bit is set, it indicates that the received packet matches one of the enabled Layer 4 Port Number fields. This status is given only when one of the following conditions is true: • Layer 3 fields are not enabled and all enabled Layer 4 fields match • All enabled Layer 3 and Layer 4 filter fields match When more than one filter matches, this bit gives the layer 4 filter status of filter indicated by bits[31:29]. Note: This status is not available when Flexible RX Parser is enabled.
L3L4FM	31:29	rwh	**Layer 3 and Layer 4 Filter Number Matched** These bits indicate the number of the Layer 3 and Layer 4 Filter that matched the received packet: • 000: Filter 0 • 001: Filter 1 • 010: Filter 2 • 011: Filter 3 • 100: Filter 4 • 101: Filter 5 • 110: Filter 6 • 111: Filter 7 This field is valid only when bit 28 or bit 27 is set high. When more than one filter matches, these bits give the number of lowest filter. Note: This status is not available when Flexible RX Parser is enabled.
0	13:11	r	**Reserved** Read as 0; should be written as 0

20.8.97.19 RAM RDESC word 3 read format GETH

The Ethernet Receive descriptor list j is stored in DRE RAM at DRE RAM base address + 0x3180 + j*0xC80
There are 4 descriptors (each of which is made up of four 32-bit words) within each descriptor list.

RDESCi_RD3G (i=0-3) Offset address: $0318C_H+i*10_H$

RAM RDESC word 3 read format GETH RAMInit value: $XXXX\ XXXX_H$



Field	Bits	Type	Description
IOC	30	rwh	**Interrupt on Completion** This bit sets the TI bit in the DMA_CH(#i)_Status register after the present packet has been transmitted.
OWN	31	rwh	**OWN bit** When this bit is set, it indicates that the DMA owns the descriptor. When this bit is reset, it indicates that the DRE owns the descriptor. The DMA clears this bit after it completes the transfer of data given in the associated buffers.
0	29:0	r	**Reserved** Read as 0; should be written as 0

20.8.97.20 RAM RDESC word 3 read format LETH

The Ethernet Receive descriptor list j is stored in DRE RAM at DRE RAM base address + 0x3180 + j*0xC80
There are 4 descriptors (each of which is made up of four 32-bit words) within each descriptor list.

RDESCi_RD3L (i=0-3) Offset address: $0318C_H+i*10_H$

RAM RDESC word 3 read format LETH RAMInit value: $XXXX\ XXXX_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OWN	IOC	0					BUF1V	0							
rwh	rwh	r					rwh	r							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															
r															

Field	Bits	Type	Description
BUF1V	24	rwh	**Buffer 1 address valid** When this bit is set, it indicates the DMA that the EIBUF address specified in the RDESCi_RD0.BUF1AP is valid.
IOC	30	rwh	**Interrupt on Completion** This bit sets the TI bit in the DMA_CH(#i)_Status register after the present packet has been transmitted.
OWN	31	rwh	**OWN bit** When this bit is set, it indicates that the DMA owns the descriptor. When this bit is reset, it indicates that the DRE owns the descriptor. The DMA clears this bit after it completes the transfer of data given in the associated buffers.
0	23:0, 29:25	r	**Reserved** Read as 0; should be written as 0

20.8.97.21 RAM RDESC word 3 Write-back format GETH

The Ethernet Receive descriptor list j is stored in DRE RAM at DRE RAM base address + 0x3180 + j*0xC80
There are 4 descriptors (each of which is made up of four 32-bit words) within each descriptor list.

RDESCi_WR3G (i=0-3) Offset address: $0318C_H+i*10_H$

RAM RDESC word 3 Write-back format GETH RAMInit value: $XXXX\ XXXX_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OWN	CTXT	FD	LD	CDA	0		ETM NCP	L34T				ETLT			
rwh	rwh	rwh	rwh	rwh	r		rwh	rwh				rwh			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ES	FRPSL	PL													
rwh	rwh	rwh													

Field	Bits	Type	Description
PL	13:0	rwh	**Packet length** These bits indicate the byte length of the received packet that was transferred to system memory. This does not include the VLAN/Pad/CRC bytes if VLAN/Pad/CRC stripping is enabled. This field is valid when the LD (bit[28]) is set and ES (bit[15]) is reset. Note: Only this bit-field is checked by the DRE for the packet length. So this value should include the complete length including the headers
FRPSL	14	rwh	**Flexible Receive Parser Status LSB bit** Indicates the status of the Flexible receive parser. For details see, GETH/LETH specification
ES	15	rwh	**Error Summary** This bit indicates that the received packet has an error. The type of error is indicated by bits [19:16]. This field is valid only when the LD bit of RDESCi_WR3 is set. When this bit is set, INT_10 is triggered • Bit[0]: IP Header Error • Bit[14]: Jabber Timeout • Bit[13]: Packet Flush • Bit[12]: Payload Checksum Error • Bit[11]: Remote Fault • Bit[10]: Local Fault • Bit[2]: Underflow Error
ETLT	19:16	rwh	**Error Type or L2 Type** When the 15th bit of this descriptor (ES) is set to 1, this field indicates the Error Type. When the 15th bit of this descriptor (ES) is set to 0, this field indicates the L2 Packet Type. Refer GETH/LETH UM for the encoding

(table continues...)

(continued)

Field	Bits	Type	Description
L34T	23:20	rwh	**Layer3/ Layer4 packet Type** These bits indicate the type of payload encapsulated in the IP datagram and processed by the Receive Checksum Offload Engine
ETMNCP	24	rwh	**Ether Type Match or No Coagulation Packet** When L34T=0b0000, this bit is set to 1 (ETM Status) when the Type field of the received packet matches the value programmed in the MAC_Rx_Eth_Type_Match register. When L34T has any value other than zero and this bit is set to 1 (NCP Status), it indicates that the DMA has received a packet that must not be kept holding for TCP packet coagulation. It must be forwarded to the TCP/IP stack immediately even though it may be in sequence with respect to the previous packet. NCP Status is valid only when the OoS function is enabled in the MAC_RSS_Control register. For a tunneled packet, ETM refers to the Type Match of the Inner Ethernet Header. NCP is not supported for a tunneled packet.
CDA	27	rwh	**Context Descriptor Available** The DRE does not support context descriptors
LD	28	rwh	**Last Descriptor** When this bit is set, it indicates that the buffers to which this descriptor is pointing are the last buffers of the packet.
FD	29	rwh	**First Descriptor** When this bit is set, it indicates that this descriptor contains the first buffer of the packet. If the size of the first buffer is 0, the second buffer contains the beginning of the packet. If the size of the second buffer is also 0, the next descriptor contains the beginning of the packet. CTXT, FD, and LD bits together indicate Descriptor Definition Error. During Write-back, all three bits are set to 1 {CTXT, FD, LD} = 3'b111 to indicate Descriptor Definition Error in the Rx Descriptor. Descriptor Definition Error for Rx Descriptor is indicated when both Buffer-1 and Buffer-2 are all 1s.
CTXT	30	rwh	**Receive Context Descriptor** When this bit is set, it indicates that the current descriptor is a context type descriptor. The DMA writes 1'b0 to this bit for normal receive descriptor.
OWN	31	rwh	**OWN bit** When this bit is set, it indicates that the DMA owns the descriptor. When this bit is reset, it indicates that the DRE owns the descriptor. The DMA clears this bit after it completes the transfer of data given in the associated buffers.
0	26:25	r	**Reserved** Read as 0; should be written as 0

20.8.97.22 RAM RDESC word 3 Write-back format LETH

The Ethernet Receive descriptor list j is stored in DRE RAM at DRE RAM base address + 0x3180 + j*0xC80
There are 4 descriptors (each of which is made up of four 32-bit words) within each descriptor list.

RDESCi_WR3L (i=0-3) Offset address: $0318C_H+i*10_H$

RAM RDESC word 3 Write-back format LETH RAMInit value: $XXXX\ XXXX_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OWN	CTXT	FD	LD	RS2V	RS1V	RS0V	CE	GP	RWT	OE	RE	DE	LT		
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ES	PL														
rwh	rwh														

Field	Bits	Type	Description
PL	14:0	rwh	**Packet length** These bits indicate the byte length of the received packet that was transferred to system memory (including CRC). This field is valid when the LD bit of RDESCi_WR3 is set and ES bits are reset. The packet length also includes the two bytes appended to the Ethernet packet when IP checksum calculation is enabled and the received packet is not a MAC control packet. This field is valid when the LD bit of RDESCi_WR3 is set. When the Last Descriptor and Error Summary bits are not set, this field indicates the accumulated number of bytes that have been transferred for the current packet. Note: Only this bit-field is checked by the DRE for the packet length. So this value should include the complete length including the headers
ES	15	rwh	**Error Summary** When this bit is set, it indicates the logical OR of the following bits: • RDESCi_WR3[24]: CRC Error • RDESCi_WR3[19]: Dribble Error • RDESCi_WR3[20]: Receive Error • RDESCi_WR3[22]: Watchdog Timeout • RDESCi_WR3[21]: Overflow Error • RDESCi_WR3[23]: Giant Packet • RDESCi_WR2[17]: Destination Address Filter Fail, when Flexible RX Parser is enabled • RDESCi_WR2[16]: SA Address Filter Fail, when Flexible RX Parser is enabled This field is valid only when the LD bit of RDESCi_WR3 is set.

(table continues...)

(continued)

Field	Bits	Type	Description
LT	18:16	rwh	**Length/Type Field** This field indicates if the packet received is a length packet or a type packet. The encoding of the 3 bits is as follows: • 3'b000: The packet is a length packet • 3'b001: The packet is a type packet. • 3'b011: The packet is a ARP Request packet type • 3'b100: The packet is a type packet with VLAN Tag • 3'b101: The packet is a type packet with Double VLAN Tag • 3'b110: The packet is a MAC Control packet type • 3'b111: The packet is a OAM packet type • 3'b010: Reserved
DE	19	rwh	**Dribble Bit Error** When this bit is set, it indicates that the received packet has a non-integer multiple of bytes (odd nibbles). This bit is valid only in the MII Mode.
RE	20	rwh	**Receive Error** When this bit is set, it indicates that the gmii_rxer_i signal is asserted while the gmii_rxdv_i signal is asserted during packet reception. This error also includes carrier extension error in the GMII and half-duplex mode. Error can be of less or no extension, or error (rxd!= 0f) during extension.
OE	21	rwh	**Overflow Error** When this bit is set, it indicates that the received packet is damaged because of buffer overflow in Rx FIFO. Note: This bit is set only when the DMA transfers a partial packet to the application. This happens only when the Rx FIFO is operating in the threshold mode. In the store-and-forward mode, all partial packets are dropped completely in Rx FIFO. NCP Status is valid only when the OoS function is enabled in the MAC_RSS_Control register. For a tunneled packet, ETM refers to the Type Match of the Inner Ethernet Header. NCP is not supported for a tunneled packet.
RWT	22	rwh	**Receive Watchdog Timeout** When this bit is set, it indicates that the Receive Watchdog Timer has expired while receiving the current packet. The current packet is truncated after watchdog timeout.
GP	23	rwh	**Giant Packet** When this bit is set, it indicates that the packet length exceeds the specified maximum Ethernet size of 1518, 1522, or 2000 bytes (9018 or 9022 bytes if jumbo packet enable is set). Note: Giant packet indicates only the packet length. It does not cause any packet truncation.
CE	24	rwh	**CRC Error** When this bit is set, it indicates that a Cyclic Redundancy Check (CRC) Error occurred on the received packet. This field is valid only when the LD bit of RDESCi_WR3 is set.

(table continues...)

(continued)

Field	Bits	Type	Description
RS0V	25	rwh	**Receive Status RDESCi_WR0 Valid** When this bit is set, it indicates that the status in RDESCi_WR0 is valid and it is written by the DMA. This bit is valid only when the LD bit of RDESCi_WR3 is set.
RS1V	26	rwh	**Receive Status RDESCi_WR1 Valid** When this bit is set, it indicates that the status in RDESCi_WR1 is valid and it is written by the DMA. This bit is valid only when the LD bit of RDESCi_WR3 is set.
RS2V	27	rwh	**Receive Status RDESCi_WR2 Valid** When this bit is set, it indicates that the status in RDESCi_WR2 is valid and it is written by the DMA. This bit is valid only when the LD bit of RDESCi_WR3 is set.
LD	28	rwh	**Last Descriptor** When this bit is set, it indicates that the buffers to which this descriptor is pointing are the last buffers of the packet.
FD	29	rwh	**First Descriptor** When this bit is set, it indicates that this descriptor contains the first buffer of the packet. If the size of the first buffer is 0, the second buffer contains the beginning of the packet. If the size of the second buffer is also 0, the next descriptor contains the beginning of the packet. See the CTXT bit description for details of using the CTXT bit and FD bit together.
CTXT	30	rwh	**Receive Context Descriptor** When this bit is set, it indicates that the current descriptor is a context type descriptor. The DMA writes 1'b0 to this bit for normal receive descriptor. When CTXT and FD bits are used together, {CTXT, FD} • 00: Intermediate Descriptor • 01: First Descriptor • 10: Reserved • 11: Descriptor Error (due to all 1s)
OWN	31	rwh	**OWN bit** When this bit is set, it indicates that the DMA owns the descriptor. When this bit is reset, it indicates that the DRE owns the descriptor. The DMA clears this bit after it completes the transfer of data given in the associated buffers.

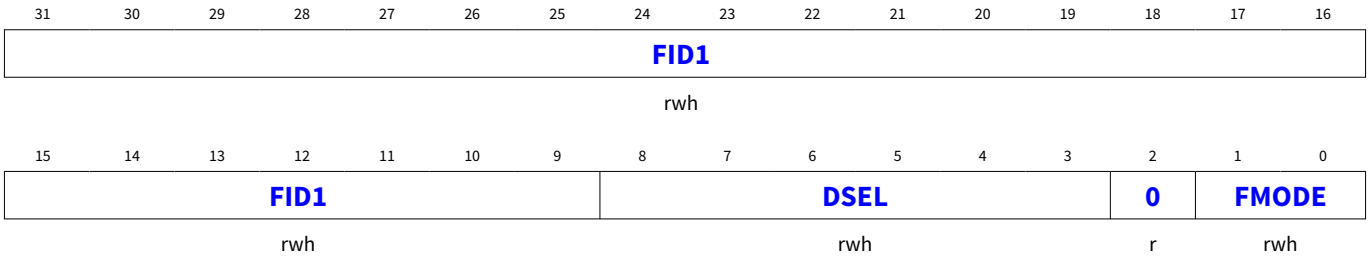
20.8.98 Forwarding table RAM interface

20.8.98.1 RAM Forwarding rule and FID1

Consists of the forwarding rule and Filter ID.

FT_FEj_FRULE (j=0-127) Offset address: 07640_H+j*8

RAM Forwarding rule and FID1 RAMInit value: $XXXX\ XXXX_H$



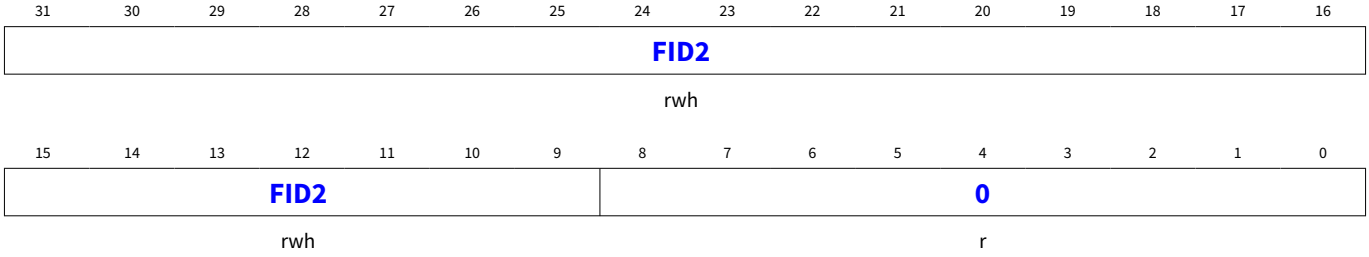
Field	Bits	Type	Description
FMODE	1:0	rwh	**Filter mode** Forwarding filter modes 00_B **CLASSIC**: Classic filter Filter ID 1 used as the filter and ID 2 used as Mask 01_B **DUAL**: Dual ID filter Filter ID1 or Filter ID2 used as filter and FWD ID should be exact match with one of the Filter ID 10_B **RANGE**: Range ID filter Range of IDs between Filter ID1 and Filter ID2 are used as Filter IDs. The FWD ID must match the values between Filter ID1 and Filter ID2. 11_B **OTHER**: Other Treated as Classsic Filter
DSEL	8:3	rwh	**Destination select** Bit encoded destination Tx descriptor list select. The Ethernet desriptor handler prepares the corresponding Tx descriptors for the frame in EIBUF • Bit[0] : Tx desc list 0 • Bit[1] : Tx desc list 1 • Bit[2] : Tx desc list 2 • Bit[3] : Tx desc list 3 • Bit[4] : Tx desc list 4 • Bit[5] : Tx desc list 5 In case of uni-cast forward only one bit is set. In case of multi-cast forward, multiple bits can be set. If the Tx descriptor list selected has TETHDLi_CTRL.TRIG bit set, the frame is not forwarded to that particular destination and this particular DSEL bit is treated as cleared When no bit is set, the frame is not forwarded and a IDID bit is set and INT_10 is triggered
FID1	31:9	rwh	**Forward filter ID1** Forward filter ID 1
0	2	r	**Reserved** Read as 0; should be written as 0

20.8.98.2 RAM Forward filter ID2

Forward filter ID2

FT_FEj_FID2 (j=0-127) Offset address: 07644_H+j*8

RAM Forward filter ID2 RAMInit value: $XXXX\ XXXX_H$



Field	Bits	Type	Description
FID2	31:9	rwh	**Forward filter ID2** Forward filter ID 2
0	8:0	r	**Reserved** Read as 0; should be written as 0

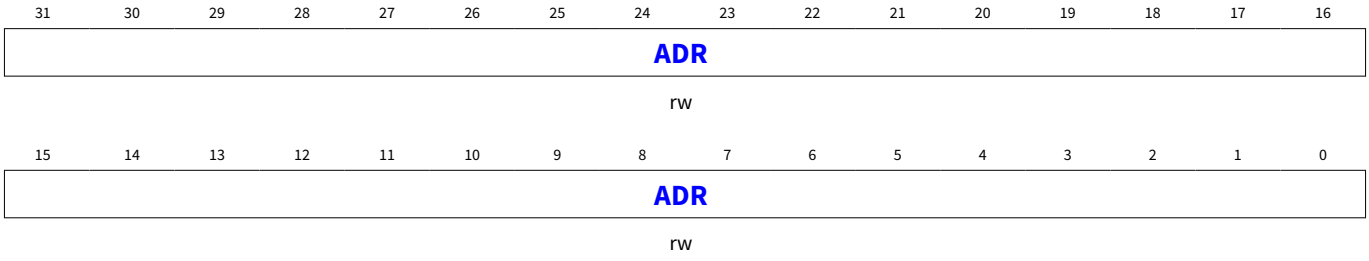
20.8.99 DMEM parameter table RAM interface

20.8.99.1 RAM Destination memory start address

The DMEM parameter table elements are stored in DRE RAM starting from DRE RAM base address + 0x7A40. This register configures the start address of the destination memory region

DMEMi_SA (i=0-27) Offset address: $07A40_H+i*10_H$

RAM Destination memory start address RAMInit value: $XXXX\ XXXX_H$



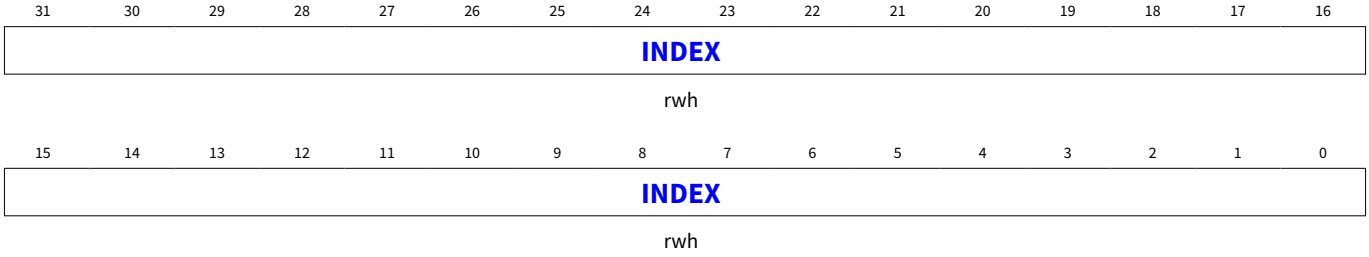
Field	Bits	Type	Description
ADR	31:0	rw	**Absolute Start Address** The start address of the destination memory. Only ADR[31:3] are considered for address decoding, ADR[2:0] are by default 000b. Note: The start address offset is always 64 bit aligned.

20.8.99.2 RAM Destination memory frame data buffer index

The DMEM parameter table elements are stored in DRE RAM starting from DRE RAM base address + 0x7A40.

DMEMi_FDBI (i=0-27) Offset address: $07A44_H+i*10_H$

RAM Destination memory frame data buffer index RAMInit value: $XXXX\ XXXX_H$



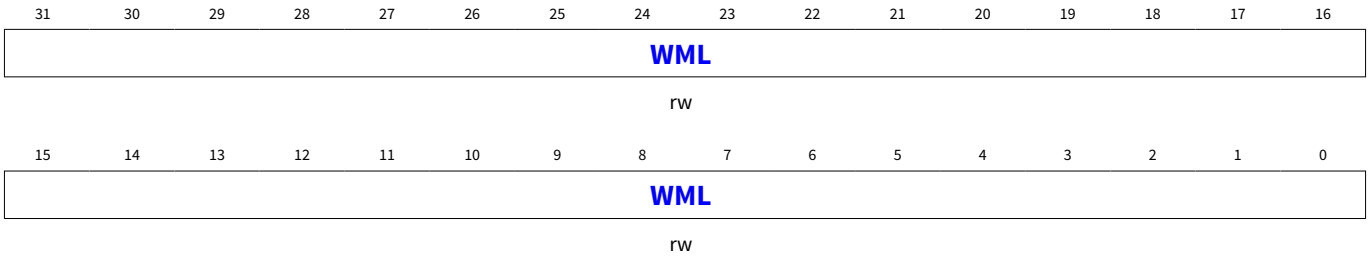
Field	Bits	Type	Description
INDEX	31:0	rwh	**Data buffer index** This bit-field is calculated by the hardware on every transfer of message to the destination memory. It denotes where the next CAN message is to be stored in the Frame data buffer. The counter is reset to 0 when a wraparound event has occurred.

20.8.99.3 RAM Destination memory watermark level

The DMEM parameter table elements are stored in DRE RAM starting from DRE RAM base address + 0x7A40.
This bit-field configures the Watermark level in case of Single Mode
When WML= WAL, the behaviour is as observed in Continuos mode.

DMEMi_WM (i=0-27) Offset address: $07A48_H+i*10_H$

RAM Destination memory watermark level RAMInit value: $XXXX\ XXXX_H$



Field	Bits	Type	Description
WML	31:0	rw	**Watermark level** • Index-based trigger mode : This is only valid for CAN I-PDU. The watermark level is configured as absolute address from the DMEMi_SA. The watermark flag DMEMi_STATUS.WMF is set (after successful transfer of CAN I-PDU to destination memory) when this address value is reached. This bit-field has to be configured to a value higher than the size of a single CAN message (plus headers) • Count-based trigger mode : The watermark level is configured in terms of number of frames or I-PDUs. The watermark flag DMEMi_STATUS.WMF is set (after successful transfer of CAN frame or CAN I-PDU to destination memory) when the DMEMi_STATUS.BC value of current CAN frame equals to the configured WML bit-field value. When WML is configured 0, the Watermark interrupt would be triggered after 1 CAN message is copied to DMEMi. When WML>=WAL, the behaviour is as observed in Continuos mode. In order to disable watermark, the user shall set WML>=WAL

20.8.99.4 RAM Destination memory wraparound level

The DMEM parameter table elements are stored in DRE RAM starting from DRE RAM base address + 0x7A40.
Wraparound level for both Single and Continuos mode

DMEMi_WA (i=0-27) Offset address: $07A4C_H+i*10_H$

RAM Destination memory wraparound level RAMInit value: $XXXX\ XXXX_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
WAL															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WAL															
rw															

Field	Bits	Type	Description
WAL	31:0	rw	**Wraparound level** • Index-based trigger mode : This is only valid for CAN I-PDU. The wraparound level is configured as absolute address from the DMEMi_WM. The wraparound flag DMEMi_STATUS.WAF is set (after successful transfer of CAN I-PDU to destination memory) when this address value is reached. This bit-field has to be configured to a value higher than the size of a single CAN message (plus headers) • Count-based trigger mode : The wraparound level is configured in terms of number of frames or I-PDUs. The watermark flag DMEMi_STATUS.WAF is set (after successful transfer of CAN frame or CAN I-PDU to destination memory) when the DMEMi_STATUS.BC value of current CAN frame equals to the configure WAL bit-field value.

20.9 Debug information

This section is not applicable for the module being described.

20.10 References

1. IEEE 1722-2016, IEEE Standard for a Transport Protocol for Time-Sensitive Applications in Bridged Local Area Networks
2. IEEE 802.1Q-2018 - IEEE Standard for Local and Metropolitan Area Networks—Bridges and Bridged Networks
3. IEEE 802.3-2018 - IEEE Standard for Ethernet
4. ISO 11898-1:2015 - Road vehicles — Controller area network (CAN) — Part 1: Data link layer and physical signalling

20.11 DRE revision history

Reference	Description of change(s)
Date range: 2024-08-17 to 2024-10-25	
Registers	• Changed representation of register base addresses and offsets (no change of physical register addresses)

20.12 TC4Dx DRE information

20.12.1 TC4Dx DRE configuration

There is no device specific DRE configuration

20.12.2 TC4Dx DRE features

Forwarding between Ethernet interfaces include:

1. GETH to LETH forwarding
2. LETH to GETH forwarding

20.12.3 TC4Dx DRE functional description

1. DRE acts as a Master on the ComPB bus and uses the system clock frequency f_{SPB}. In TC4Dx, the ComPB also uses this clock

20.12.4 TC4Dx DRE registers

There are no deviations from the generic specification.

20.12.4.1 Memory overview tables of DRE domain SFR

Table 965 Memory overview - DRE domain SFR (ascending address)

Short name	Long name	Address
DRE_RAM	Embedded SRAM for DRE (08000_H Byte)	$F9038000_H$

20.12.4.2 Register address space - DRE

Table 966 Registers address space - DRE

Module	Base address	End address	Note
DRE	$F9030000_H$	$F903FFFF_H$	SRI Slave Interface

20.12.4.3 Register address space - DRE domain Floating

Table 967 Registers address space - DRE domain Floating

Module	Domain	Base address	End address	Note
DRE	Floating	$F9038000_H$	$F903809F_H$	CAN Address Database RAM interface
	Floating	$F90380A0_H$	$F90386DF_H$	CAN Input Buffer List RAM interface
	Floating	$F90386E0_H$	$F9039ADF_H$	CAN Output Buffer List RAM interface
	Floating	$F903AAE0_H$	$F903AEDF_H$	CAN Transmit Routing Table RAM interface
	Floating	$F903AAE0_H$	$F903AB0F_H$	Ethernet Address Database RAM interface
	Floating	$F903AB40_H$	$F903B7BF_H$	Ethernet descriptors RAM interface
	Floating	$F903F640_H$	$F903FA3F_H$	Forwarding table RAM interface
	Floating	$F903FA40_H$	$F903FBFF_H$	DMEM parameter table RAM interface

20.12.4.4 Register overview - access mode glossary

Table 968 Register overview - access mode glossary

Keyword	Description
E	Access protection using PROT register DRE_PROTE .
SE	Access protection using PROT register DRE_PROTSE .
APU-PETHj (j=0-5)	Protection group consisting of registers DRE_ETHj_ACCEN_WRA , DRE_ETHj_ACCEN_WRB , DRE_ETHj_ACCEN_RDA , DRE_ETHj_ACCEN_RDB , DRE_ETHj_ACCEN_VM , DRE_ETHj_ACCEN_PRS .
PETHj	Access protection using *APU-PETHj* registers.

(table continues...)

Table 968 (continued) Register overview - access mode glossary

Keyword	Description
APU-PG	Protection group consisting of registers DRE_ACCEN_WRA , DRE_ACCEN_WRB , DRE_ACCEN_RDA , DRE_ACCEN_RDB , DRE_ACCEN_VM , DRE_ACCEN_PRS .
PG	Access protection using *APU-PG* registers.
32	Access only when using 32-bit width.
SV	Access only when supervisor mode is active on the interconnect.
BE	Always returns a Bus Error.
U	No access restrictions.
PROT	Access restrictions as defined in the PROT register access rules.
P	Description can be found in global access mode definition.
nBE	Indicates that no Bus Error is generated when accessing this address range, even though it is either an access to an undefined address or the access does not follow the given rules.

20.12.4.5 Register overview - DRE (ascending offset address)

Table 969 Register overview - DRE (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
DRE_CLC	Clock Control Register	00000_H	PG, 32	PG, SV, E, 32	Application Reset	3798
DRE_OCS	OCDS Control and Status Register	00004_H	PG, 32	SV, PG, 32	Debug Reset	3799
DRE_ID	Module Identification Register	00008_H	PG, 32	BE	PowerOn Reset	3800
DRE_RST_CTRLA	Reset Control Register A	$0000C_H$	PG, 32	PG, SV, E, 32	Application Reset	3801
DRE_RST_CTRLB	Reset Control Register B	00010_H	PG, 32	PG, SV, E, 32	Application Reset	3801
DRE_RST_STAT	Reset Status Register	00014_H	PG, 32	BE	Application Reset	3802
DRE_PROTE	PROT Register Endinit	00018_H	U	SV, PROT	Application Reset	3803
DRE_PROTSE	PROT Register Safe Endinit	$0001C_H$	U	SV, PROT	Application Reset	3804
DRE_ACCEN_WRA	Write access enable register A	00020_H	32	SE, SV, 32	Application Reset	3806
DRE_ACCEN_WRB	Write access enable register B	00024_H	32	SE, SV, 32	Application Reset	3807
DRE_ACCEN_RDA	Read access enable register A	00028_H	32	SE, SV, 32	Application Reset	3807
DRE_ACCEN_RDB	Read access enable register B	$0002C_H$	32	SE, SV, 32	Application Reset	3808

(table continues...)

Table 969 (continued) Register overview - DRE (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
DRE_ACCEN_VM	VM access enable register	00030_H	32	SE, SV, 32	Application Reset	3808
DRE_ACCEN_PRS	PRS access enable register	00034_H	32	SE, SV, 32	Application Reset	3809
DRE_ETHj_ACCEN_WRA (j=0-5)	Write access enable register A j	00040_H+j *20_H	32	SE, SV, 32	Application Reset	3810
DRE_ETHj_ACCEN_WRB (j=0-5)	Write access enable register B j	00044_H+j *20_H	32	SE, SV, 32	Application Reset	3810
DRE_ETHj_ACCEN_RDA (j=0-5)	Read access enable register A j	00048_H+j *20_H	32	SE, SV, 32	Application Reset	3811
DRE_ETHj_ACCEN_RDB (j=0-5)	Read access enable register B j	$0004C_H$+ j*20_H	32	SE, SV, 32	Application Reset	3811
DRE_ETHj_ACCEN_VM (j=0-5)	VM access enable register j	00050_H+j *20_H	32	SE, SV, 32	Application Reset	3812
DRE_ETHj_ACCEN_PRS (j=0-5)	PRS access enable register j	00054_H+j *20_H	32	SE, SV, 32	Application Reset	3812
DRE_MODEr (r=0-7)	RP r mode register	01040_H+ r*4	PG, 32	E, SV, PG, 32	Application Reset	3813
DRE_CANi_RP (i=0-19)	CAN i resource partition	01060_H+i *4	PG, 32	E, SV, PG, 32	Kernel Reset	3814
DRE_CIBL_BPR	CAN input buffer pending request	$010B8_H$	PG, 32	PG, 32	Kernel Reset	3814
DRE_CIBL_STATUS	CAN input buffer list status	$010BC_H$	PG, 32	PG, 32	Kernel Reset	3815
DRE_COBL_BPR0	CAN output buffer pending request 0	$010C8_H$	PG, 32	PG, 32	Kernel Reset	3816
DRE_COBL_BPR1	CAN output buffer pending request 1	$010CC_H$	PG, 32	PG, 32	Kernel Reset	3817
DRE_COBL_STATUS	CAN output buffer list status	$010D0_H$	PG, 32	PG, 32	Kernel Reset	3818
DRE_EIBUFi_CONFIG (i=0-5)	Ethernet input buffer i configuration	$010D8_H$+ i*14_H	PG, 32	E, SV, PG, 32	Kernel Reset	3818

(table continues...)

Table 969 (continued) Register overview - DRE (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
DRE_EIBUFi_ERROR (i=0-5)	Ethernet input buffer i error	$010E0_H$+i*14_H	PG, 32	PG, 32	Kernel Reset	3819
DRE_EIBUFi_STATUS (i=0-5)	Ethernet input buffer i status	$010E4_H$+i*14_H	PG, 32	PG, 32	Kernel Reset	3821
DRE_EOBUFj_CONFIG (j=0-5)	Ethernet output buffer j configuration	01150_H+j*38_H	PG, 32	E, SV, PG, 32	Kernel Reset	3823
DRE_EOBUFj_MAC_H0 (j=0-5)	Ethernet output buffer j MAC header 0	01154_H+j*38_H	PG, 32	PG, 32	Kernel Reset	3825
DRE_EOBUFj_MAC_H1 (j=0-5)	Ethernet output buffer j MAC header 1	01158_H+j*38_H	PG, 32	PG, 32	Kernel Reset	3826
DRE_EOBUFj_MAC_H2 (j=0-5)	Ethernet output buffer j MAC header 2	$0115C_H$+j*38_H	PG, 32	PG, 32	Kernel Reset	3826
DRE_EOBUFj_MAC_H3 (j=0-5)	Ethernet output buffer j MAC header 3	01160_H+j*38_H	PG, 32	PG, 32	Kernel Reset	3827
DRE_EOBUFj_MAC_H4 (j=0-5)	Ethernet output buffer j MAC header 4	01164_H+j*38_H	PG, 32	PG, 32	Kernel Reset	3827
DRE_EOBUFj_NTSCF_H0 (j=0-5)	Ethernet output buffer j NTSCF header	01168_H+j*38_H	PG, 32	PG, 32	Kernel Reset	3828
DRE_EOBUFj_NTSCF_STREAM0_ID (j=0-5)	Ethernet output buffer j Stream ID configuration 0	$0116C_H$+j*38_H	PG, 32	PG, 32	Kernel Reset	3829
DRE_EOBUFj_NTSCF_STREAM1_ID (j=0-5)	Ethernet output buffer j Stream ID configuration 1	01170_H+j*38_H	PG, 32	PG, 32	Kernel Reset	3830
DRE_EOBUFj_STATUS (j=0-5)	Ethernet output buffer j status	01174_H+j*38_H	PG, 32	PG, 32	Kernel Reset	3830
DRE_EOBUFj_TTC (j=0-5)	Ethernet output buffer j Transmit trigger configuration	01178_H+j*38_H	PG, 32	PG, 32	Kernel Reset	3832

(table continues...)

Table 969 (continued) Register overview - DRE (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
DRE_EOBUFj_TTS (j=0-5)	Ethernet output buffer j Timer threshold and status	$0117C_H$+j*38_H	PG, 32	PG, 32	Kernel Reset	3832
DRE_EOBUFj_ERROR (j=0-5)	Ethernet output buffer j error	01180_H+j*38_H	PG, 32	PG, 32	Kernel Reset	3833
DRE_SIDFi_FC (i=0-7)	Stream ID filter i configuration	$0129C_H$+i*14_H	PG, 32	PG, 32	Kernel Reset	3834
DRE_SIDFi_FIL1_L (i=0-7)	Stream ID filter i configuration Stream ID filter 1 lower	$012A0_H$+i*14_H	PG, 32	PG, 32	Kernel Reset	3835
DRE_SIDFi_FIL1_H (i=0-7)	Stream ID filter i configuration Stream ID filter 1 higher	$012A4_H$+i*14_H	PG, 32	PG, 32	Kernel Reset	3835
DRE_SIDFi_FIL2_L (i=0-7)	Stream ID filter i configuration Stream ID filter 2 lower	$012A8_H$+i*14_H	PG, 32	PG, 32	Kernel Reset	3836
DRE_SIDFi_FIL2_H (i=0-7)	Stream ID filter i configuration Stream ID filter 2 higher	$012AC_H$+i*14_H	PG, 32	PG, 32	Kernel Reset	3836
DRE_RTi_CONFIG (i=0-3)	CAN transmit routing table i configuration	01340_H+i*8	PG, 32	E, SV, PG, 32	Kernel Reset	3837
DRE_RREQ_CONFIG	Routing request configuration	$0135C_H$	PG, 32	BE	Kernel Reset	3837
DRE_RREQ_CID	CAN ID request	01360_H	PG, 32	BE	Kernel Reset	3838
DRE_UCRH	Uni-cast routing header	01364_H	PG, 32	BE	Kernel Reset	3838
DRE_MCRH	Multi-cast routing header	01364_H	PG, 32	BE	Kernel Reset	3839
DRE_RS	Routing status	01368_H	PG, 32	PG, 32	Kernel Reset	3840
DRE_CANRXR0	CAN receive request 0	$0136C_H$	PG, 32	BE	Kernel Reset	3840
DRE_CANRXR1	CAN receive request 1	01370_H	PG, 32	BE	Kernel Reset	3841
DRE_CANTXR	CAN transmit buffer available request	01374_H	PG, 32	BE	Kernel Reset	3842
DRE_DMEMi_CONFIG (i=0-27)	Destination memory i configuration	$0137C_H$+i*20_H	PG, 32	E, SV, PG, 32	Kernel Reset	3842
DRE_DMEMi_MODE (i=0-27)	Destination memory i transfer mode configuration	01380_H+i*20_H	PG, 32	E, SV, PG, 32	Kernel Reset	3844

(table continues...)

Table 969 (continued) Register overview - DRE (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
DRE_DMEMi_STATUS (i=0-27)	Destination memory i status	01390_H+i*20_H	PG, 32	PG, 32	Kernel Reset	3845
DRE_DMEMi_RP (i=0-27)	Destination memory i resource partition	01394_H+i*20_H	PG, 32	E, SV, PG, 32	Kernel Reset	3846
DRE_ME_SRCA	Move engine source address	$016F8_H$	PG, 32	BE	Kernel Reset	3847
DRE_ME_DESTA	Move engine destination address	$016FC_H$	PG, 32	BE	Kernel Reset	3847
DRE_ME_STATE	Move engine state	01700_H	PG, 32	BE	Kernel Reset	3847
DRE_ME_FESRCA	Move engine first error source address	01704_H	PG, 32	BE	Kernel Reset	3848
DRE_ME_FEDESTA	Move engine first error destination address	01708_H	PG, 32	BE	Kernel Reset	3849
DRE_ME_ERR	Move engine error register	$0170C_H$	PG, 32	PG, 32	Kernel Reset	3849
DRE_INTSIG	Interrupt signal	01710_H	PG, 32	BE	Kernel Reset	3851
DRE_IE	Interrupt line enable	01714_H	PG, 32	PG, 32	Kernel Reset	3853
DRE_RETHDLi_CTRL (i=0-5)	Rx Ethernet descriptor list i configuration and control	$0171C_H$+i*8	PG, 32	E, SV, PG, 32	Kernel Reset	3854
DRE_TETHDLi_CTRL (i=0-5)	Tx Ethernet descriptor list i configuration and control	$0174C_H$+i*10_H	PG, 32	E, SV, PG, 32	Kernel Reset	3856
DRE_EDLSTAT	Ethernet descriptor list status	$017D8_H$	PG, 32	PG, 32	Kernel Reset	3858
DRE_EREQ	Ethernet requests summary	$017DC_H$	PG, 32	PG, 32	Kernel Reset	3859
DRE_FTCFG	Forwarding table configuration	$017E4_H$	PG, 32	E, SV, PG, 32	Kernel Reset	3860
DRE_CWDCFG	DRE CAN watchdog configuration	$017EC_H$	PG, 32	E, SV, PG, 32	Kernel Reset	3861
DRE_EWDCFG	DRE Ethernet watchdog configuration	$017F4_H$	PG, 32	E, SV, PG, 32	Kernel Reset	3862
DRE_EADCFG	Ethernet address database configuration	$017F8_H$	PG, 32	E, SV, PG, 32	Kernel Reset	3862
DRE_DMAi_RP (i=0-5)	DMA i resource partition	$017FC_H$+i*4	PG, 32	E, SV, PG, 32	Kernel Reset	3863
DRE_CITO	CAN Input buffer timeout status	01818_H	PG, 32	PG, 32	Kernel Reset	3863
DRE_COTO0	CAN Output buffer timeout status 0	$0181C_H$	PG, 32	PG, 32	Kernel Reset	3864

(table continues...)

Table 969 (continued) Register overview - DRE (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
DRE_COTO1	CAN Output buffer timeout status 1	01820_H	PG, 32	PG, 32	Kernel Reset	3864

20.12.4.6 Device specific registers

There are no device specific register changes.

20.12.5 TC4Dx DRE connectivity

The table below lists interfaces to and from this functional block to other blocks in the device.

Table 970 List of DRE interface signals

Interface signals	I/O	Description
CLOCK_DRE_fSPB	In	SPB clock input
CLOCK_DRE_fSRI	In	SRI clock input
CAN0_DRE_TRIGTYPE[1:0]	In	Trigger input per MCMCAN
CAN1_DRE_TRIGTYPE[1:0]	In	Trigger input per MCMCAN
CAN2_DRE_TRIGTYPE[1:0]	In	Trigger input per MCMCAN
CAN3_DRE_TRIGTYPE[1:0]	In	Trigger input per MCMCAN
CAN4_DRE_TRIGTYPE[1:0]	In	Trigger input per MCMCAN
CAN0_DRE_TRIGNODE[1:0]	In	Trigger input per MCMCAN
CAN1_DRE_TRIGNODE[1:0]	In	Trigger input per MCMCAN
CAN2_DRE_TRIGNODE[1:0]	In	Trigger input per MCMCAN
CAN3_DRE_TRIGNODE[1:0]	In	Trigger input per MCMCAN
CAN4_DRE_TRIGNODE[1:0]	In	Trigger input per MCMCAN
DRE_IR_DREw[15:0]	Out	Interrupt request

20.12.6 TC4Dx DRE revision history

Initial release of the chapter.

21 Controller Area Network interface (CAN)

The CAN module provides a communication interface according to the ISO 11898-1 standard, supporting both classical CAN and CAN FD communication. It provides the CAN Routing Engine as an extension which is used to route data within different CAN nodes based on the CAN-ID, and it also provides a virtual host interface to the RxFIFOs and TxQueues/TxFIFO of CAN nodes. The CAN module in this document is referred as MCMCAN. The CAN nodes in this document is referred as M_CAN.

21.1 Feature list

- Compatibility to ISO 11898-1
- CAN Error Logging
- AUTOSAR optimized
- SAE J1939 optimized
- Direct Message RAM access for Host CPU
- Multiple M_CANs share the same Message RAM
- Maskable module interrupts
- Power-down support
- Debug on CAN support
- CAN Routing Engine (CRE)
  - Abstraction of RxFIFO, TxFIFO/TxQueue operation with virtual buffers
  - Hardware accelerated CAN to CAN routing within the same MCMCAN module
  - Assists Data Routing Engine (DRE) to perform hardware accelerated
    - CAN to CAN routing between different MCMCAN Modules
    - CAN to Ethernet (in IEEE 1722 ACF frame) routing
    - CAN to a user-configurable System RAM location transfer
    - CAN PDU to a user configurable system RAM location transfer
  - Consists of a measurement unit for Intrusion Detection (IDMU)

21.2 Functional overview

This figure gives an overview of the MCMCAN block.

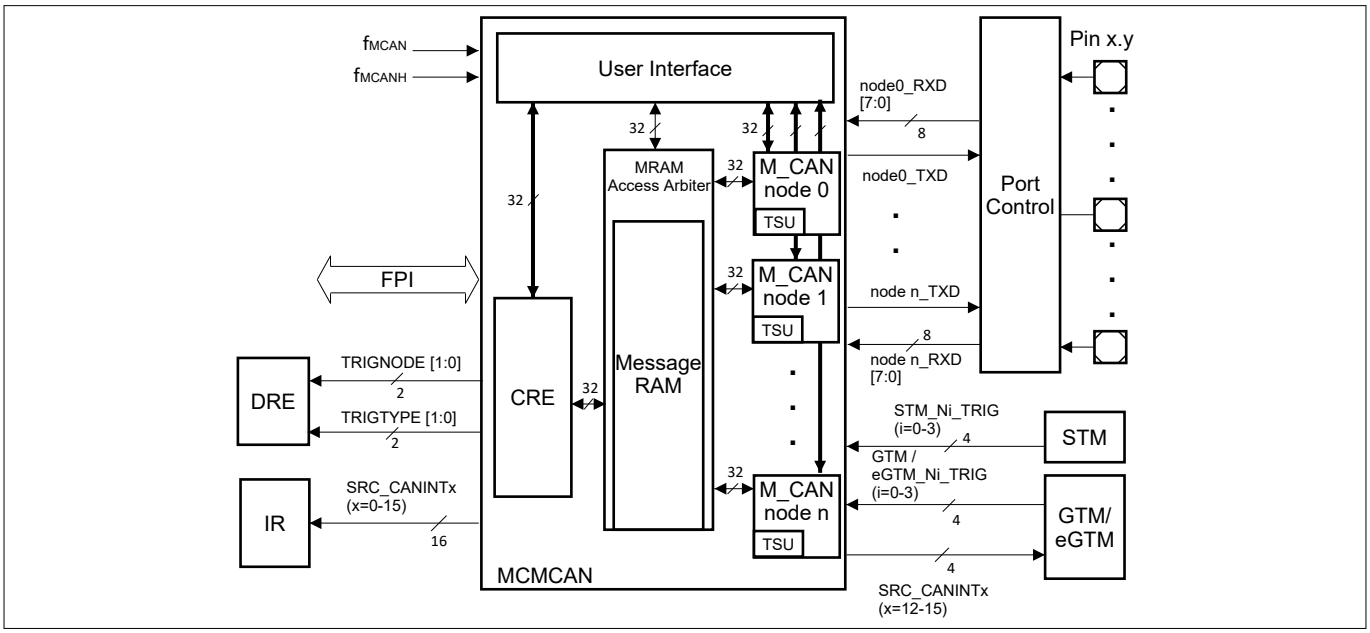


Figure 363 MCMCAN block diagram

The MCMCAN contains of the following sub blocks: User Interface, M_CAN nodes, the CAN routing engine and the Message RAM.

Related information

TC4Dx SMU alarm mapping tables on page 7233

21.3 Functional description

Please refer to each of the CAN functional sub-blocks for the functional description.

21.4 User interface (UI)

The user interface is the wrapper which provides the interface for control of the M_CAN, CRE and MRAM to the user and to the CPU. It contains configuration for clocks, trigger signal, interrupts, timers and the modules connection to the outside world.

21.4.1 Feature list

- Clock selection and generation through Clock Control Block
  - Up to 160 MHz input clock for CAN baud rate generation
- Flexible interrupt structure generation available
  - Interrupt groups
- Timer based transmission of CAN frame and timeouts for reception of CAN frames
- Read and write access protection for Message RAM

21.4.2 Functional overview

The overview of the User Interface is shown in the following figure. The clock generation is handled in Clock Control Block, where the required frequencies are derived from the f_{MCAN} and f_{MCANH} inputs.

The Timing functions module is used for calculating the delays and generating triggers when automatic transmission of messages is needed.

The Interrupt Generation module consists of the interrupt grouping logic for interrupt triggers and the node specific transmit trigger.

For test purposes, the user can connect the nodes to an CAN internal bus, this mode is also referred as loop back mode. Additionally, the user can select the node in the loop back mode out, where a connection between the internal CAN bus and an external CAN bus is possible. This is provided from the Test functions module.

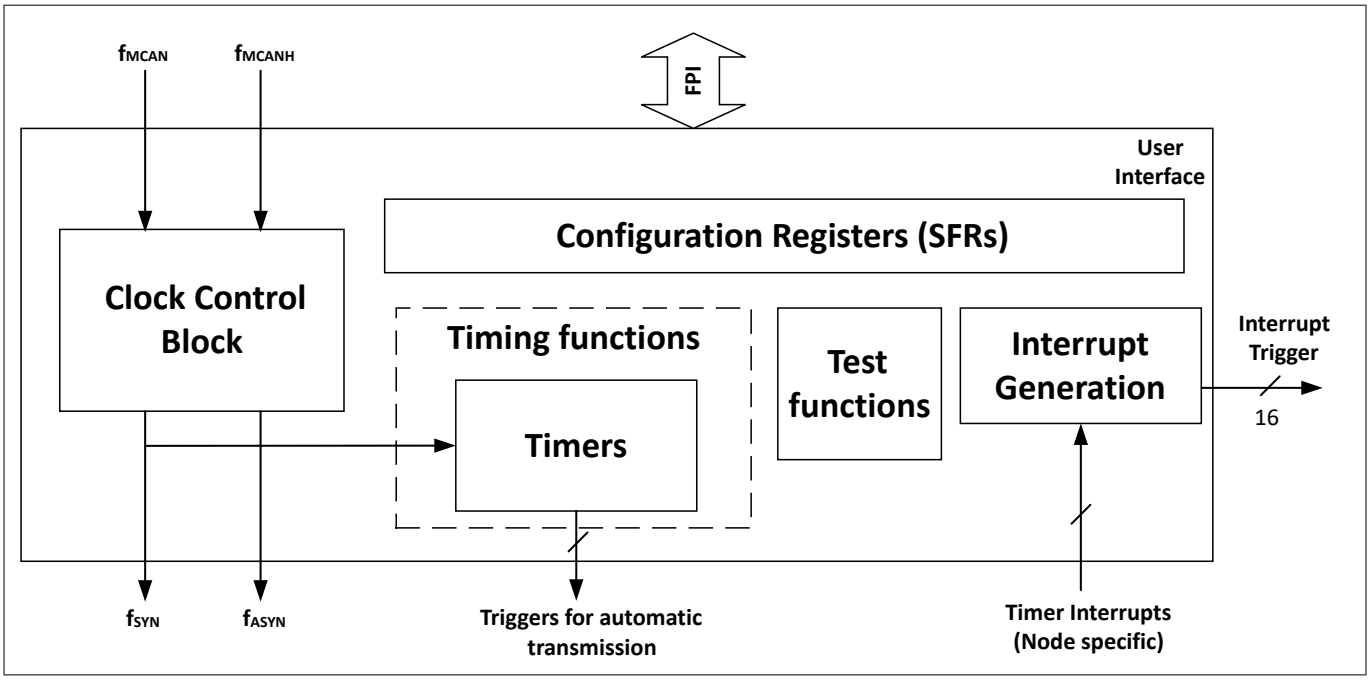


Figure 364 UI block diagram

21.4.3 Functional description

The MCMCAN User Interface describes about the clock generation, grouping of interrupts, external connections to the module and IO configurations.

21.4.3.1 MCMCAN clockpaths

A general overview of clocks within the MCMCAN module.

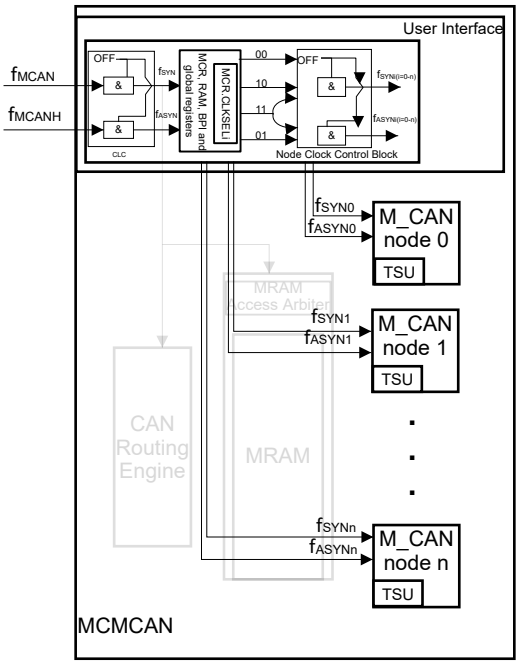


Figure 365 MCMCAN clock interconnects and clock control unit

The MCMCAN module clock inputs are connected to the clock control unit (CCU). The CLC setting supplies the global module registers with its clocks. To supply the M_CAN nodes with the corresponding clocks, the MCR.CLKSELi registers have to be set.

In the figure above the clock switch through CLC and clock control for single nodes is shown. The asynchronous clock (f_{ASYN}) as well as the synchronous clock (f_{SYN})of each single M_CAN node can be switched on and off through MCR.CLKSELi register bit-fields.

The f_{SYN} is supplied from f_{MCANH} and f_{ASYN} is supplied from f_{MCAN} from CCU. The f_{SYN} is used as the clock source for registers and RAM interface, f_{ASYN} is used to generate the nominal and fast CAN FD baudrates. It is recommended to use f_{ASYN} as 20 MHz, 40 MHz, 80 MHz, or 160 MHz from the Peripheral clock or also from f_{OSC}, in order to achieve commonly used nominal and fast CAN FD baudrates. The condition that $f_{SYN} >= f_{ASYN}$ is essential for proper functioning of MCMCAN.

Table 971 MCMCAN clock interconnects

CAN clock inputs	Connected to	Description
f_{MCAN}	CCU	f_{MCAN} of the MCMCAN module is one of the clock inputs of the Clock Control Block, providing the MCMCAN with the clock for the asynchronous clock path f_{ASYN}
f_{MCANH}	CCU	f_{MCANH} of the MCMCAN module is the clock input of the Clock Control Block, providing the main kernel clock f_{SYN}
f_{SYN}	CLC	f_{MCANH} becomes f_{SYN} within the module
f_{ASYN}	CLC	f_{MCAN} becomes f_{ASYN} within the module
f_{SYNi}	MCR.CLKSELi	f_{SYN} becomes f_{SYNi} clocking the synchronous part of M_CAN node i (Nodes can be switched on or off individually)
f_{ASYNi}	MCR.CLKSELi	f_{ASYN} becomes f_{ASYNi} clocking the asynchronous part of M_CAN node i (Nodes can be switched on or off individually)

Clock selection

The asynchronous clock part of the M_CAN and the rest of the MCMCAN module are separate frequency domains and can be driven by separate independent frequencies. The clocks for the module are chosen within the clock control unit. The asynchronous clock can be chosen in the CCU among the Peripheral clock f_{PLL} or with direct drive from the oscillator f_{OSC}.

The purpose of supplying the asynchronous clock part with a direct oscillator clock is to avoid the clock jitter added by the PLL, necessary when the chip is driven by a low cost ceramic resonator instead of by a high precision quartz crystal.

As shown in the figure above the clock signals for the MCMCAN module are generated and controlled by a clock control unit. This clock control unit is responsible for the enable or disable control, the clock frequency adjustment.

Clock control register

The Module clock is enabled using the CLC register. The module control clock f_{SYN} is used inside the MCMCAN module for control purposes such as clocking of control logic and register operations. The frequency of f_{SYN} is sourced by f_{MCANH} from CCU module. This clock is independent to f_{SPB} and allows M_CAN to continue operation when f_{SPB} is reduced in frequency, therefore enabling pretended networking. The clock control register CLC makes it possible to enable or disable f_{SYN} and f_{ASYN} under certain conditions.

21.4.3.2 Interrupt groups

Interrupt grouping is fixed, as shown in the following figures. For the complete module, 16 interrupt nodes are existing. The interrupt groups can be freely assigned to the interrupt line by using Ni_G0INTR (i=0-3), Ni_G1INTR (i=0-3) and Ni_G2INTR (i=0-3) .

21.4.3.2.1 Mapping of interrupts

The mapping of interrupt from the interrupt registers into groups, out of which only enabled interrupt sources are forwarded, is as following:

- REINT - Receive Interrupt
- RxF0F - Rx FIFO 0 Full
- RxF1F - Rx FIFO 1 Full
- RxF0N - Rx FIFO 0 New Message
- RxF1N - Rx FIFO 1 New Message
- RXTI - Receive Timeout
- TRAQ - Transmission Queue
- TRACO - Transmission Control
- TEFIFO - Tx Event FIFO incident
- HPE - High Priority Event
- WATI - Watermark Interrupt
- ALRT - Alert
- MOER - Module Error
- SAFE - Safety
- BOFF - Bus Off
- LOI - Last Error Interrupt

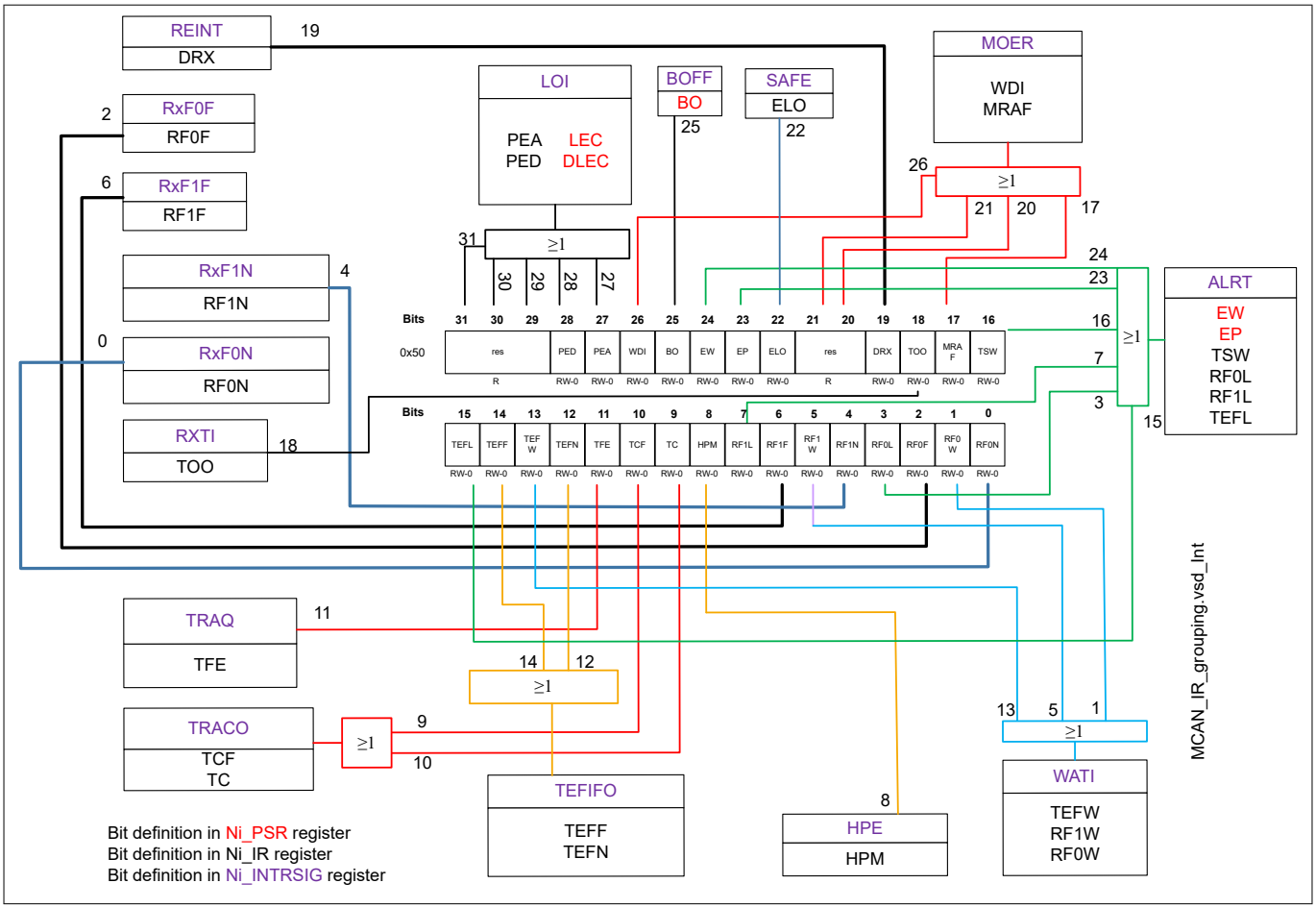


Figure 366 Mapping of interrupts into groups

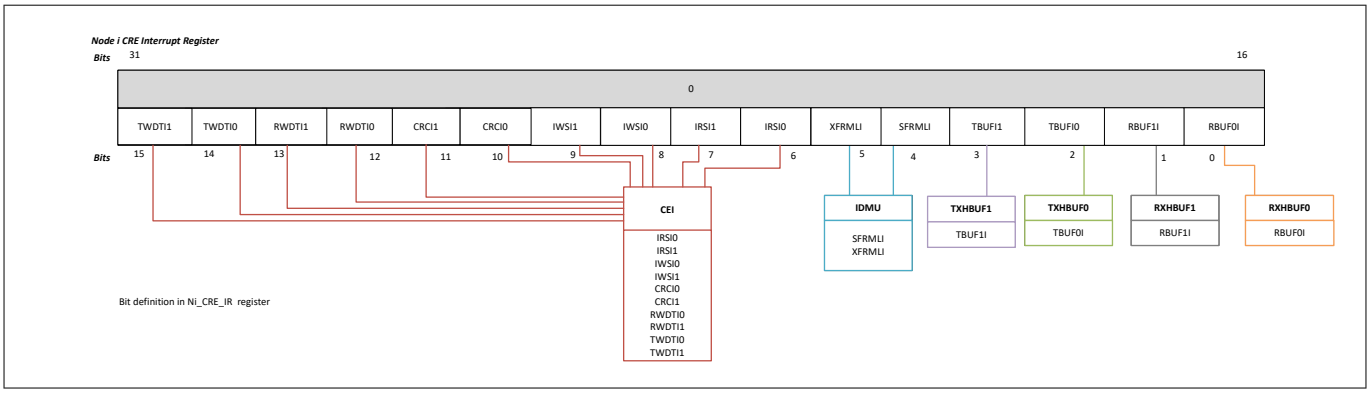


Figure 367 Mapping of CRE interrupts into groups

21.4.3.2.2 Signalling interrupts of groups

The groups defined in the previous paragraph are also shown in Ni_INTRSIG (i=0-3). For each group, 0 in the corresponding bit-field means that no interrupt is pending and 1 means pending interrupt.

21.4.3.2.3 Interrupt control

The general interrupt structure is shown in the following figure:

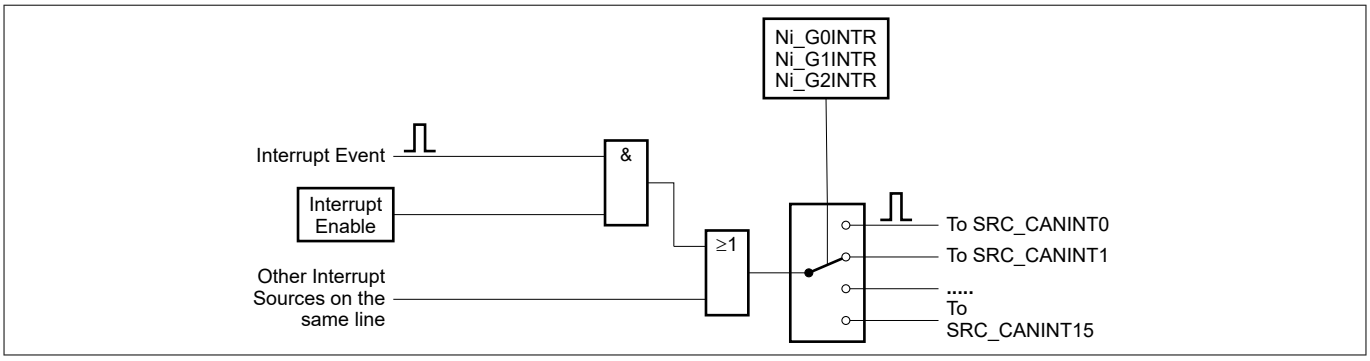


Figure 368 General interrupt structure

The interrupt event can trigger the interrupt generation. The interrupt pulse is generated based on the interrupt flag and the interrupt enable bit. The interrupt flag registers are: Ni_IR (i=0-3) and Ni_CRE_IR (i=0-3). The interrupts are enabled at Ni_IE (i=0-3) and CRE configuration registers. The pulse generation is purely based on AND logic between the interrupt flags and interrupt enable bit-fields. The interrupt flag can be reset by software by writing a '1' to the Ni_IR and Ni_CRE_IR respective bit-field.

If enabled by the related interrupt enable bit in the corresponding interrupt enable register (Ni_IE (i=0-3), Ni_TIMER_RXTOUT (i=0-3).TEIE and CRE configuration registers), an interrupt pulse can be generated at one of the 16 interrupt output lines SRC_CANINTn of the module using Ni_G0INTR (i=0-3), Ni_G1INTR (i=0-3) and Ni_G2INTR (i=0-3). If more than one interrupt source is connected to the same interrupt line the requests are combined to one common line.

The interrupt groups are only OR - ing the interrupts of the corresponding CAN nodes. The interrupt request has to be reset in the node.

***Note**: Enabling an interrupt in Interrupt Enable register when the corresponding flag is already set in Interrupt Register will also generate an interrupt pulse. Hence it is recommended to clear the interrupt flags before enabling the corresponding interrupts.*

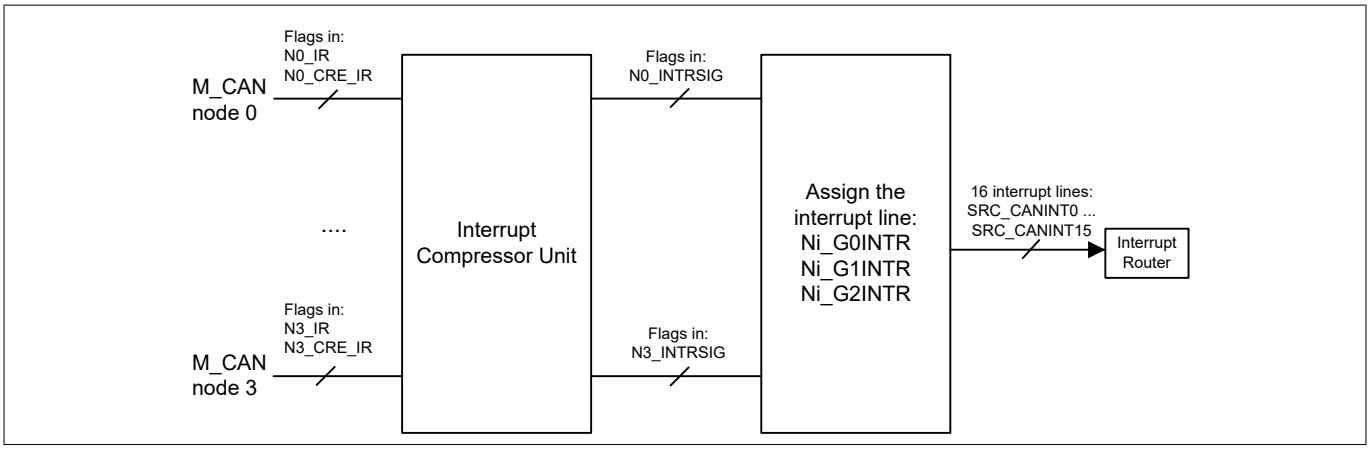


Figure 369 Interrupt compression Unit

21.4.3.3 Connecting the module to the outside world

Each node in the MCMCAN module can be connected to an internal or external CAN bus via configuring the Ni_PORTCTRL (i=0-3).LBM.

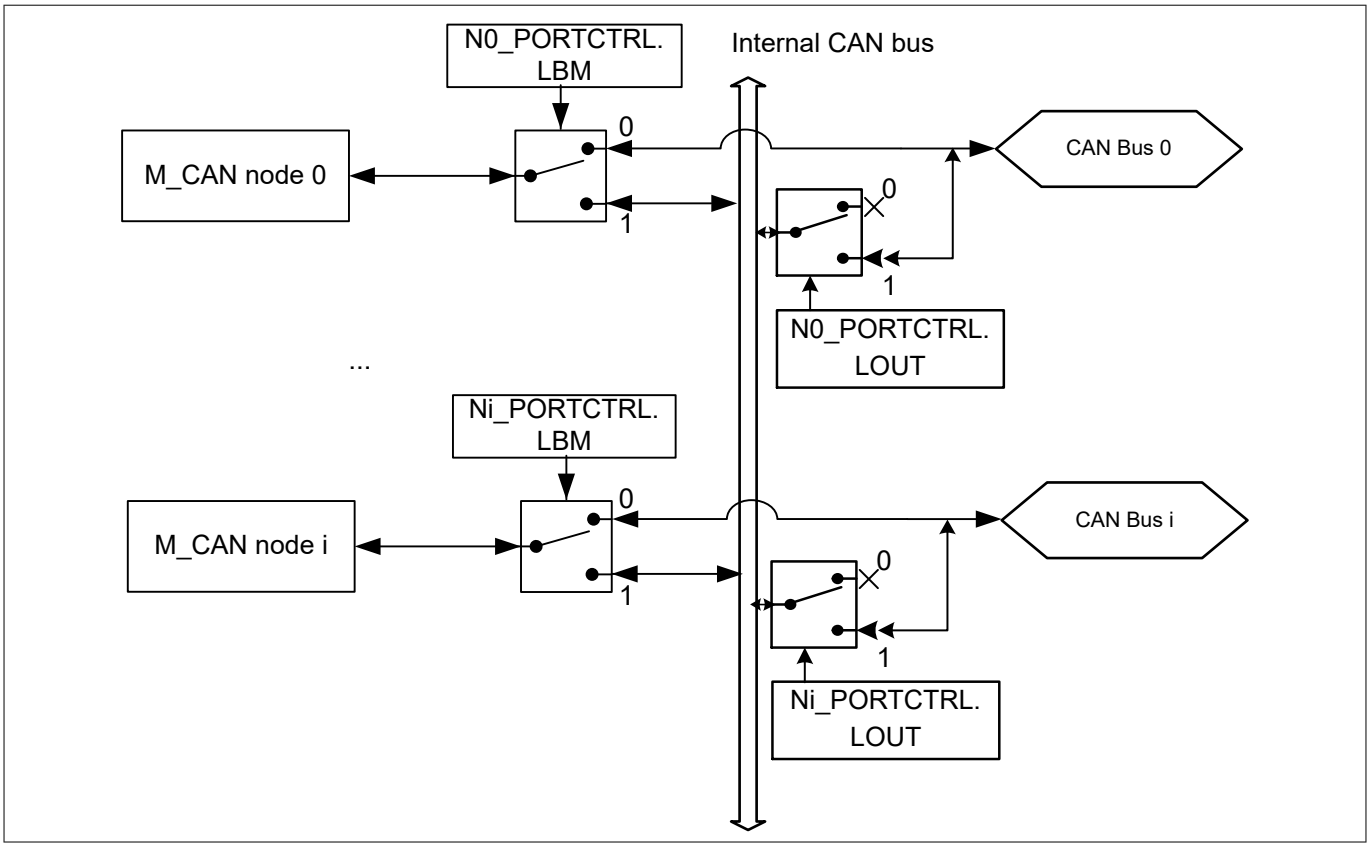


Figure 370 Module internal loop-back mode and loop-back mode out

21.4.3.3.1 Module internal loop-back mode

The MCMCAN module provides a module internal loop-back mode to enable an in-system test of the MCMCAN module as well as the development of CAN driver software without access to an external CAN bus.

The loop-back feature consists of an internal CAN bus (inside the MCMCAN module) and a bus select switch for each CAN node. With the switch, each CAN node can be connected either to the internal CAN bus (internal loop-back mode activated) or the external CAN bus, respectively to transmit and receive pins (normal operation). The CAN bus that is not currently selected is driven recessive; this means the transmit pin is held at 1, and the receive pin is ignored by the CAN nodes that are in Loop-Back Mode.

The internal loop-back mode is selected for CAN node x by setting the Node x Port Control Register bit Ni_PORTCTRL (i=0-3).LBM. All CAN nodes that are in loop-back mode may communicate together through the internal CAN bus without affecting the normal operation of the other CAN nodes that are not in Loop-Back Mode.

21.4.3.3.2 Module loop back mode out

Setting Ni_PORTCTRL (i=0-3).LOUT bit will send the signals from the internal loop back bus to the corresponding pins of the CAN node and vice versa. Therefore a receive and transmit between internal loop back bus and a bus system outside is possible. When Ni_PORTCTRL (i=0-3).LOUT is set to one, the internal loop back bus and the external bus will be communicating with each other. The table below explains the behavior of the CAN module for different combinations Ni_PORTCTRL (i=0-3).LOUT and Ni_PORTCTRL (i=0-3).LBM.

Table 972 Combinations of internal loop back mode and loop back mode out

Ni_PORTCTRL.LOUT	Ni_PORTCTRL.LBM	Description
0	0	Normal CAN operation. Internal and loop back out mode deactivated
0	1	Internal Loop Back mode
1	0	Loop Back Mode Out
1	1	Reserved

21.4.3.4 Setting the address protection for the CAN nodes

Inside the Register User Interface, the address protection is defined. Here the nodes become protectable, as well as the control area. More information regarding the protection scheme is found in the PROT chapter.

MCMCAN has 5 APU set of registers to protect read/write access condition of separate parts of the module. APU with ID 0-3, shown in Register overview chapter as PNi(i=0-3) controls CAN node i(i=0-3) read/write access protection. APU with ID 4 is used for other registers read/write protection (shared by all CAN nodes). Therefore each node has isolated read and write access protection configuration for its configuration registers. Individual masters could be configured for each and every CAN node.

The Node Start Address (Ni_STARTADR (i=0-3)) and Node End Address (Ni_ENDADR (i=0-3)) registers, give the possibility to define a range within CAN and CRE RAM, belonging to a node, which can only be written and read by masters configured in CAN node APU register. The specified RAM area is write and read protected. An overlap of the protected areas, is not found by the hardware.

21.4.3.5 Node timing functions

A CAN node offers the following timing functions:

- A receive timeout mode that can detect the reception within message buffers. The timeout expires, when no message has been received
- Without CPU involvement, a selectable message buffer can be transmitted periodically, triggered by a timer, the System Timer (STM), the General Timer Module (GTM) or the Enhanced General Timer Module (eGTM)

The clocking options for the node timer is controlled by Node Timer Clock Control Register, Ni_TIMER_CCR (i=0-3), the node timing functions for Receive Timeout Mode is controlled by Ni_TIMER_RXTOUT (i=0-3) and for Transmit Trigger Mode is controlled by Node i Timer 0/1/2 Transmit Trigger Registers (Ni_TIMER_TXTRIG0 (i=0-3), Ni_TIMER_TXTRIG1 (i=0-3), Ni_TIMER_TXTRIG2 (i=0-3)). The timers will start running after setting the STRT bit and setting the RELOAD value, to a value different than 0.

Modes with timer usage

A CAN node timer is driven by the synchronous clock or by the corresponding clock sources, divided by a prescaler selected through bit-field Ni_TIMER_CCR (i=0-3).TPSC in the corresponding timer control register. The timers are enabled by writing to the RELOAD bits in the relevant node timer registers. Then it decrements from its initial value. The further behavior depends on the selected timer mode:

- Receive Timeout Mode: The receive timeout function is valid for the receive buffers. A receive time-out check is enabled, which may be a received frame or remote data frame. If any of the message buffers receives before the timer is 0, the timer will be reloaded. When the timer reaches 0, it will stop. Bit TE in Ni_TIMER_RXTOUT (i=0-3) register will be set. With bit Ni_TIMER_RXTOUT (i=0-3).TEIE = 1, an interrupt will be generated.

***Note**: To avoid unintended receive timeout interrupts, the corresponding receive interrupts shall be enabled via bits Ni_IE.RF0NE, Ni_IE.RF1NE or Ni_IE.DRXE depending on the usage of RxFIFO0/1 or dedicated Rx buffers for proper function of the receive timeout interrupt*

- Transmit Trigger Mode: When the timer reaches 0, the TXRQ of the corresponding message buffer is selected

Transmit trigger by system timer or general timer module

For Node i Timer the trigger for transmission of a message can also be set by a System Timer (STM) trigger event, the General Timer Module (GTM), or the Enhanced General Timer Module (eGTM) trigger event, see the following figure. The bit-field Ni_TIMER_CCR (i=0-3).TRIGSRC in the timer clock control register enables this feature and the timer is started once values are written to the RELOAD bits of the relevant Ni_TIMER_TXTRIG0 (i=0-3), Ni_TIMER_TXTRIG1 (i=0-3) or Ni_TIMER_TXTRIG2 (i=0-3) registers. In transmit trigger mode, when a trigger event occurs (STM, GTM or eGTM), the node timer will be decremented per trigger event timing prescaled by (TPSC+1). The transmit request is set for the corresponding transmit message buffer one trigger event after the RELOAD value reaches zero.

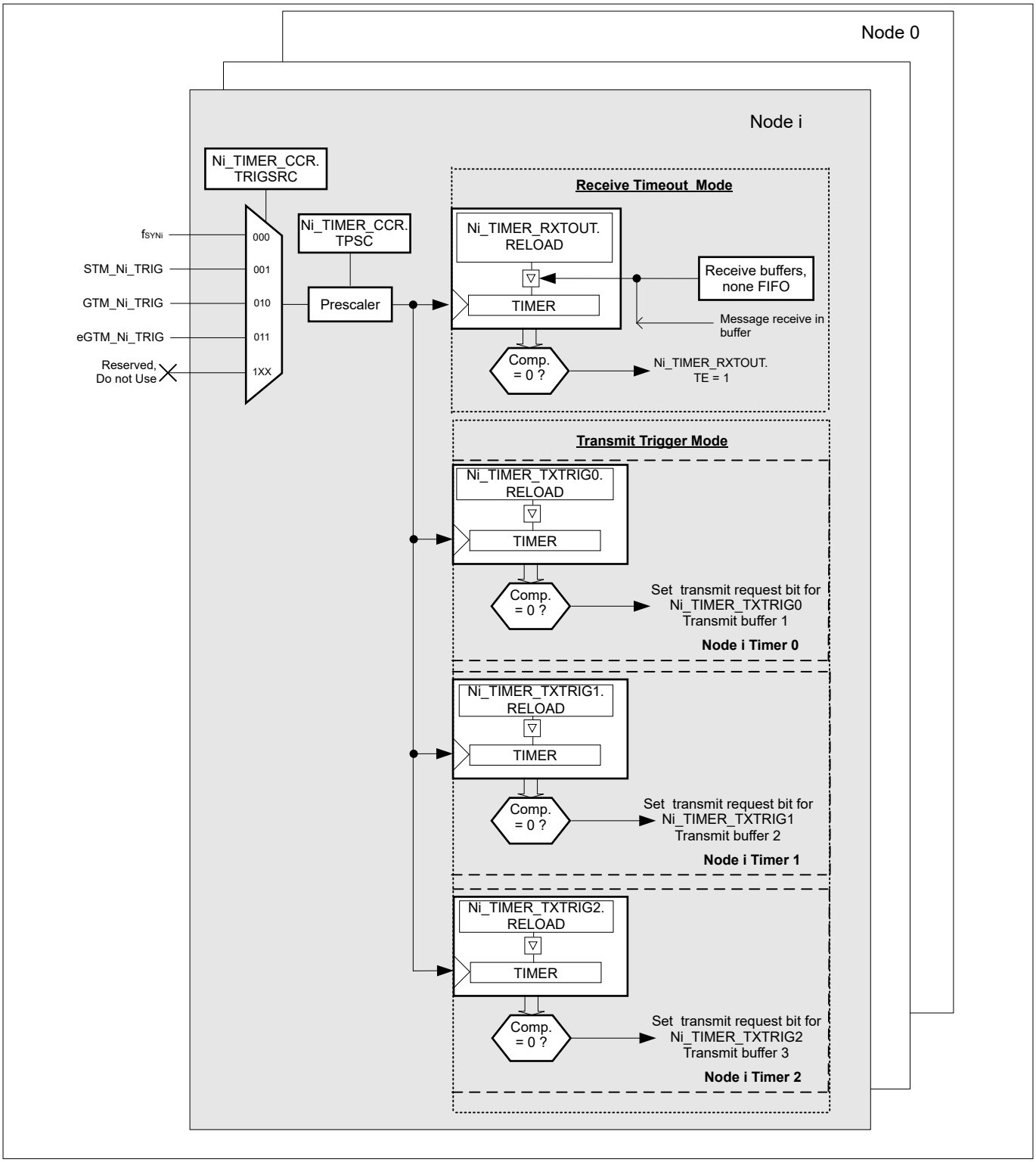


Figure 371 CAN node timing modes

21.4.3.5.1 Automatic transferring of messages

Ni_TIMER_TXTRIG0 (i=0-3), Ni_TIMER_TXTRIG1 (i=0-3), Ni_TIMER_TXTRIG2 (i=0-3) registers enable the M_CAN to trigger messages on timer events. The timer per module have one clock source. The counting events can take place on an STM interrupt, a GTM, an eGTM interrupt or by f_{SYNi}. These timers can be used to enable Pretended Networking, or to have hardware support for a gateway functionality. This functionality needs a receive

interrupt or a watermark interrupt within a FIFO, triggering a DMA transfer. The transmit objects will be automatically triggered to transfer the message buffer, through timer underflow. As the transmit triggers are fixed to message RAM buffers 1-3, this feature only works if these buffers are available.

21.4.3.6 Message RAM

The Message RAM is shared by all the CAN nodes within the module. The elements of the message RAM are configured per CAN node. When operated in CAN FD mode the required Message RAM size strongly depends on the element size configured for Rx FIFO0, Rx FIFO1, Rx Buffers, and Tx Buffers through Ni_RXESC.F0DS, Ni_RXESC.F1DS, Ni_RXESC.RBDS, and Ni_TXESC.TBDS.

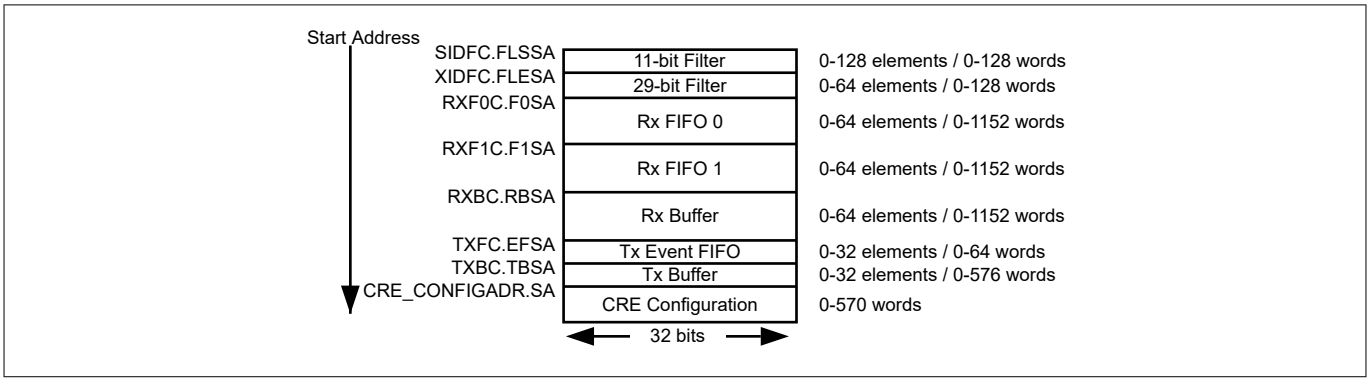


Figure 372 Message RAM configuration

When the M_CAN addresses the Message RAM it addresses 32-bit words, not single bytes. The configurable start addresses are 32-bit word addresses this means only bits 15 to 2 are evaluated, the two least significant bits are ignored.

The CRE part of RAM structure is shown in the figure above.

***Note**: The M_CAN does not check for erroneous configuration of the Message RAM. Especially the configuration of the start addresses of the different sections and the number of elements of each section has to be done carefully to avoid falsification or loss of data.*

21.4.3.6.1 Rx buffer and FIFO element

Up to 64 Rx buffers and two Rx FIFOs can be configured in the Message RAM. Each Rx FIFO section can be configured to store up to 64 received messages. The structure of a Rx buffer/FIFO element is shown in the table below. The element size can be configured for storage of CAN FD messages with up to 64 bytes data field through register Ni_RXESC (i=0-3).

RXMSGk_R1A(k=0-63): When no TSU is used (Ni_CCCR (i=0-3).UTSU = '0'), R1A.RXTS[15:0] holds the 16-bit timestamp generated by the M_CAN's internal timestamp logic.

RXMSGk_R1B (k=0-63): When a TSU is used (Ni_CCCR (i=0-3).UTSU = '1') and when bit SSYNC/ESYNC of the matching filter element is set, R1B .TSC = '1' and R1B .RXTSP [3:0] holds the number of the TSU's Timestamp registers which holds the 32-bit timestamp captured by the TSU. Else RXMSGk_R1B(k=0-63).TSC= '0' and RXMSGk_R1B (k=0-63).RXTSP[3:0] is not valid.

Table 973 Message layout - Rx buffer and FIFO element

	31							24	23						16	15							8	7						0
R0	ESI	XTD	RTR	ID[28:0]																										
R1A	ANMF	FIDX[6:0]							0	FDF	BRS	DLC[3:0]				RXTS[15:0]														
R1B	ANMF	FIDX[6:0]							0	FDF	BRS	DLC[3:0]				0												TSC	RXTSP[3:0]		
DBm(0-3)	DB3[7:0]								DB2[7:0]							DB1[7:0]								DB0[7:0]						
DBm(4-7)	DB7[7:0]								DB6[7:0]							DB5[7:0]								DB4[7:0]						
…	…								…							…								…						
DBm(m-3-m)	DBm[7:0]								DBm-1[7:0]							DBm-2[7:0]								DBm-3[7:0]						

***Note**: Depending on the configuration of the element size (RXESC), between two and sixteen 32-bit words are used for storage of a CAN message's data field.*

21.4.3.6.2 Tx buffer element

The Tx buffers section can be configured to hold dedicated Tx buffers as well as a Tx FIFO/Tx Queue. If the Tx buffers section is shared by dedicated Tx buffers and a Tx FIFO/Tx Queue, the dedicated Tx buffers start at the beginning of the Tx buffers section followed by the buffers assigned to the Tx FIFO or Tx Queue. The Tx Handler distinguishes between dedicated Tx buffers and Tx FIFO/Tx Queue by evaluating the Tx buffer configuration Ni_TXBC.TFQS and Ni_TXBC.NDTB. The element size can be configured for storage of CAN FD messages with up to 64 bytes data field through register Ni_TXESC.

Table 974 Message layout - Tx buffer element

	31							24	23						16	15						8	7						0
T0	ESI	XTD	RTR	ID[28:0]																										

(table continues...)

Table 974 (continued) Message layout - Tx buffer element

T1	MM0[7:0]	EFC	TSCE	FDF	BRS	DLC[3:0]	MM1[15:8]	0
DBm(0-3)	DB3[7:0]	DB2[7:0]					DB1[7:0]	DB0[7:0]
DBm(4-7)	DB7[7:0]	DB6[7:0]					DB5[7:0]	DB4[7:0]
…	…	…					…	…
DBm(m-3-m)	DBm[7:0]	DBm-1[7:0]					DBm-2[7:0]	DBm-3[7:0]

***Note**: Depending on the configuration of the element size (TXESC), between two and sixteen 32-bit words are used for storage of a CAN message's data field.*

21.4.3.6.3 Tx event FIFO element

Each element stores information about transmitted messages. By reading the Tx event FIFO the Host CPU gets this information in the order the messages were transmitted. Status information about the Tx event FIFO can be obtained from register Ni_TXEFS.

E1A: When Ni_CCCR.WMM ='0' and no TSU is used (Ni_CCCR.UTSU = '0'). E1A.TXTS[15:0] holds the 16-bit timestamp generated by the M_CAN's internal timestamping logic.

E1B: When 16-bit Message Marker are enabled (Ni_CCCR.WMM = '1') or when Ni_CCCR.UTSU = '1', E1B.MM1[15:8] holds the upper 8 bit of the Wide Message Marker. When a TSU is used (Ni_CCCR.UTSU = '1') and when bit TSCE of the related Tx_Buffer element is set, E1B.TSC = '1' and E1B .TXTSP[3:0] holds the number of the TSU's Timestamp register which holds the 32-bit timestamp captured by the TSU. Else E1B.TSC = '0' and E1B.TXTSP[3:0] is not valid.

Table 975 Message layout - Tx event FIFO element

	31							24	23						16	15						8	7						0
E0	ESI	XTD	RTR	ID[28:0]																									

(table continues…)

Table 975 (continued) Message layout - Tx event FIFO element

E1A	MM[7:0]	ET[1:0]	FDF	BRS	DLC[3:0]	TXTS[15:0]			
E1B	MM0[7:0]	ET[1:0]	FDF	BRS	DLC[3:0]	MM1[15:8]	0	TSC	TXTSP[3:0]

21.4.3.6.4 Standard message ID filter element

Up to 128 filter elements can be configured for 11-bit standard IDs. When accessing a standard message ID Filter element, its address is the filter list standard start address SIDFCi.FLSSA plus the index of the filter element (0…127).

Table 976 Message layout - standard message ID Filter element

	31							24	23						16	15						8	7							0
S0	SFT[1:0]		SFEC[2:0]			SFID1[10:0]										SSYNC	0					SFID2[10:0]								

21.4.3.6.5 Extended message ID filter element

Up to 64 filter elements can be configured for 29-bit extended IDs. When accessing an extended message ID filter element, its address is the filter list extended start address XIDFCi.FLESA plus two times the index of the filter element (0…63).

Table 977 Message layout - extended message ID filter element

	31							24	23						16	15						8	7							0
F0	EFEC[2:0]			EFID1[28:0]																										
F1	EFT[1:0]		ESYNC	EFID2[28:0]																										

21.4.3.7 Debug over CAN (DXCM feature)

The MCMCAN controller supports debugging using standard CAN tool access in parallel to regular CAN bus traffic. This is achieved by transmitting DAP telegrams and replies as regular CAN messages (DXCM DAP over CAN Messages). DXCM uses the lowest message buffers and it is strongly recommended to use also the same CAN pins as for DXCPL (DAP over CAN Physical Layer). DXCM is enabled with the MCR.DXCM bit. Please refer to the Debug and Trace chapter for more information about DAP, DXCM and DXCPL.

Debug over CAN shall be only available on CAN0. TX Buffer 0 will be the sending transmit object. For receive at least one message buffer has to be configured. Meaning that RX Buffer 0 will be used for receiving DAP telegrams. The starting address of the message buffer and the receiving address of the message buffer have to be configured within BUFADR register.

21.5 Modular CAN (M_CAN)

This section describes the Modular CAN nodes.

21.5.1 Feature list

- Transmit and receive CAN frames with compatibility to ISO 11898-1
  - CAN FD with up to 64 data bytes supported
- Improved acceptance filtering
- Separate event signaling on reception of high priority messages
- Flexible functional configuration for software
  - Configurable transmit FIFO
  - Configurable transmit queue
  - Configurable transmit event FIFO
  - Programmable loopback test mode
  - Two configurable receive FIFOs
- Data storage:
  - Up to 64 dedicated receive buffers
  - Up to 32 dedicated transmit buffers
  - Up to sixteen 32-bit timestamps supported
- Timestamping:
  - Hardware timestamping according to CiA 603
  - AUTOSAR synchronization method supported

21.5.2 Functional overview

This part gives a overview of the M_CAN module.

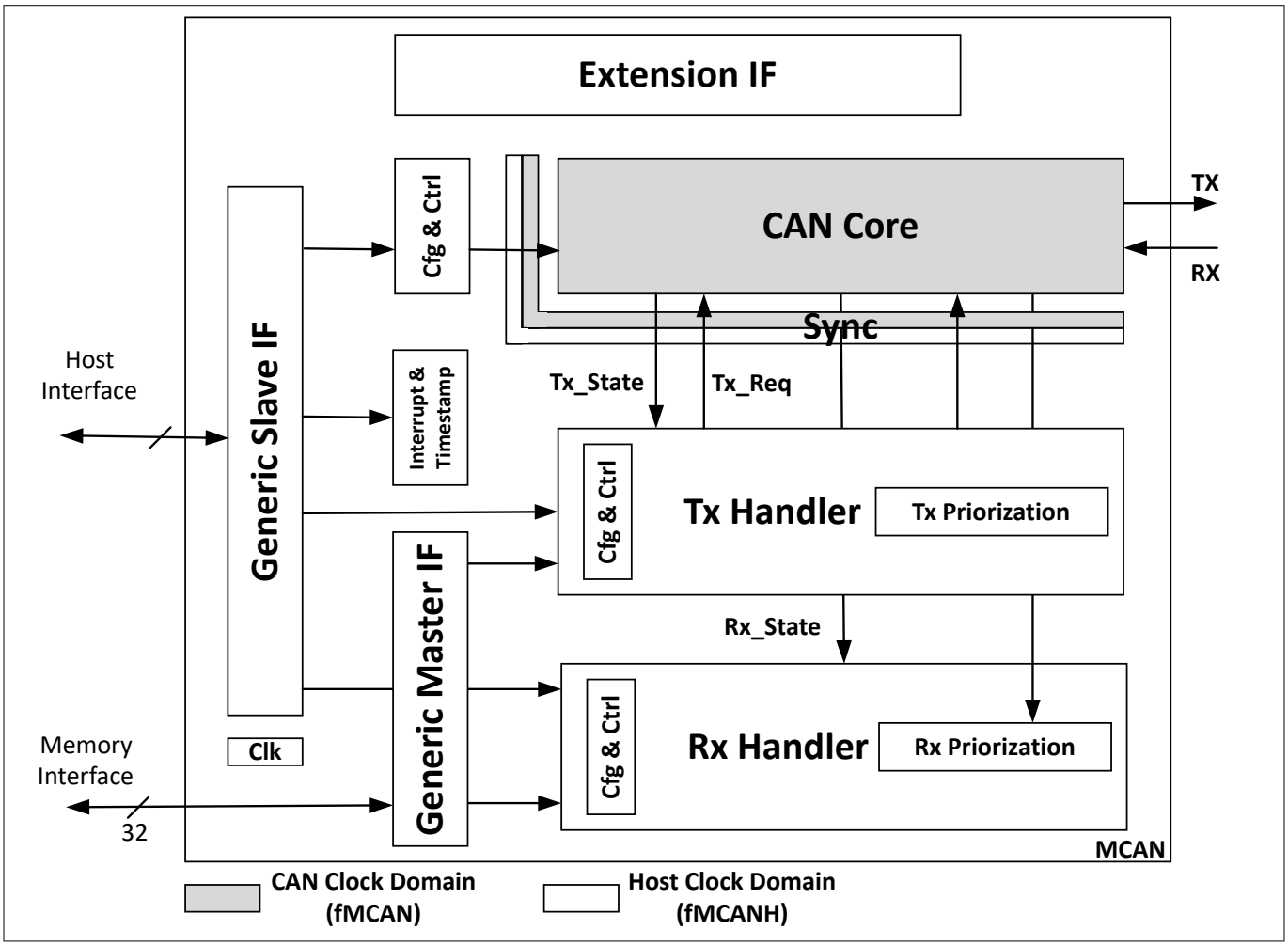


Figure 373 M_CAN block diagram

CAN core

CAN Protocol Controller and Rx or Tx Shift register. Handles all ISO 11898-1 protocol functions. Supports 11-bit and 29-bit identifiers.

Sync

Synchronizes signals from the Host clock domain to the CAN clock domain and vice versa.

Clk

Synchronizes reset signal to the Host clock domain and to the CAN clock domain.

Cfg and Ctrl

CAN core related configuration and control bits.

Interrupt and timestamp

Interrupt control and 16-bit CAN bit time counter for receive and transmit timestamp generation. An externally generated 16-bit vector may substitute the integrated 16-bit CAN bit time counter for receive and transmit timestamp generation.

Tx handler

Controls the message transfer from the external message RAM to the CAN core. A maximum of 32 Tx buffers can be configured for transmission. Tx buffers can be used as dedicated Tx buffers, as Tx FIFO, part of a Tx queue, or as a combination of them. A Tx event FIFO stores Tx timestamps together with the corresponding message ID. Transmit cancellation is also supported.

The Tx Handler also implements the Frame Synchronization Entity (FSE) which controls time-triggered communication according to ISO11898-4. It synchronizes itself to the reference messages on the CAN bus, controls cycle time and global time, and handles transmissions according to the predefined message schedule, the system matrix. It also handles the time marks of the system matrix that are linked to the messages in the Message RAM. Stop Watch Trigger, Event Trigger, and Time Mark Interrupt are synchronization interfaces.

Rx handler

Controls the transfer of received messages from the CAN core to the external message RAM. The Rx Handler supports two receive FIFOs, each of configurable size, and up to 64 dedicated Rx buffers for storage of all messages that have passed acceptance filtering. A dedicated Rx buffer, in contrast to a receive FIFO, is used to store only messages with a specific identifier. An Rx timestamp is stored together with each message. Up to 128 filters can be defined for 11-bit IDs and up to 64 filters for 29-bit IDs.

Generic slave interface

Connects the M_CAN to a customer specific Host CPU.

Generic master interface

Connects the M_CAN access to an external 32-bit message RAM.

Extension interface

All flags from the interrupt register (Ni_IR SFR) and selected internal status and control signals are routed to this interface.

21.5.3 Functional description

The CAN module consists of four M_CAN nodes. Each M_CAN node represents a CAN channel interface, which is configured individually. The functional description of a M_CAN node is given in this chapter.

21.5.3.1 Operating modes

This chapter provides the details of configuration options for a M_CAN operation, which includes: software initialization, CAN and CAN FD operation, sleep mode and test modes.

21.5.3.1.1 Software initialization

Software initialization is started by setting bit Ni_CCCR (i=0-3).INIT, either by software or by a hardware reset, or by going Bus_Off. While Ni_CCCR (i=0-3).INIT is set, message transfer from and to the CAN bus is stopped, the status of the CAN bus output TXD is recessive (HIGH). The counters of the Error Management Logic EML are unchanged. Setting Ni_CCCR (i=0-3).INIT does not change any configuration register. Resetting Ni_CCCR (i=0-3).INIT finishes the software initialization. Afterwards the Bit Stream Processor BSP synchronizes itself to the data transfer on the CAN bus by waiting for the occurrence of a sequence of 11 consecutive recessive bits (Bus_Idle) before it can take part in bus activities and start the message transfer.

Access to the M_CAN configuration registers is only enabled when both bits Ni_CCCR (i=0-3).INIT and Ni_CCCR (i=0-3).CCE are set (protected write).

Ni_CCCR (i=0-3).CCE can only be set/reset while Ni_CCCR (i=0-3).INIT = 1. Ni_CCCR (i=0-3).CCE is automatically reset, when Ni_CCCR (i=0-3).INIT is cleared.

The following registers are reset when Ni_CCCR (i=0-3).CCE is set

- Ni_HPMS (i=0-3) - High Priority Message Status
- Ni_RXF0S (i=0-3) - Rx FIFO 0 Status
- Ni_RXF1S (i=0-3) - Rx FIFO 1 Status
- Ni_TXFQS (i=0-3) - Tx FIFO/Queue Status
- Ni_TXBRP (i=0-3) - Tx Buffer Request Pending
- Ni_TXBTO (i=0-3) - Tx Buffer Transmission Occurred
- Ni_TXBCF (i=0-3) - Tx Buffer Cancellation Finished
- Ni_TXEFS (i=0-3) - Tx Event FIFO Status

The Timeout Counter value Ni_TOCV (i=0-3).TOC is preset to the value configured by Ni_TOCC (i=0-3).TOP when Ni_CCCR (i=0-3).CCE is set.

In addition the state machines of the Tx Handler and Rx Handler are held in idle state while Ni_CCCR (i=0-3).CCE = 1.

The following registers are only writeable while Ni_CCCR (i=0-3).CCE = ‘0’

- Ni_TXBAR (i=0-3) - Tx Buffer Add Request
- Ni_TXBCR (i=0-3) - Tx Buffer Cancellation Request

Ni_CCCR (i=0-3).TEST and Ni_CCCR (i=0-3).MON can only be set by the Host while Ni_CCCR (i=0-3).INIT = 1 and Ni_CCCR (i=0-3).CCE = 1. Both bits may be reset at any time. Ni_CCCR (i=0-3).DAR can only be set/reset while Ni_CCCR (i=0-3).INIT = 1 and Ni_CCCR (i=0-3).CCE = 1.

21.5.3.1.2 Normal operation

The M_CAN's default operating mode after hardware reset is event-driven CAN communication.

Once the M_CAN is initialized and Ni_CCCR (i=0-3).INIT is reset to zero, the M_CAN synchronizes itself to the CAN bus and is ready for communication.

After passing the acceptance filtering, received messages including Message ID and DLC are stored into a dedicated Rx Buffer or into Rx FIFO 0 or Rx FIFO 1.

For messages to be transmitted dedicated Tx Buffers and/or a Tx FIFO or a Tx Queue can be initialized or updated. Automated transmission on reception of remote frames is not implemented.

21.5.3.1.3 CAN FD operation

There are two variants in the CAN FD frame transmission:

- The CAN FD frame without bit-rate switching
- The CAN FD frame where control field, data field, and CRC field are transmitted with a higher bit rate than the beginning and the end of the frame

The previously reserved bit in CAN frames with 11-bit identifiers and the first previously reserved bit in CAN frames with 29-bit identifiers will now be decoded as FDF bit. If FDF is equal to:

- Recessive: It signifies a CAN FD frame
- Dominant It signifies a Classical CAN frame

In a CAN FD frame, the two bits following FDF, res and BRS, decide whether the bit rate inside of this CAN FD frame is switched. A CAN FD bit rate switch is signified by res = dominant and BRS = recessive. The coding of res = recessive is reserved for future expansion of the protocol.

In case the M_CAN receives a frame with FDF = recessive and res = recessive, it will signal a Protocol Exception Event by setting bit Ni_PSR (i=0-3).PXE. When Protocol Exception Handling is:

- Enabled (Ni_CCCR (i=0-3).PXHD = 0): This causes the operation state to change from Receiver (Ni_PSR (i=0-3).ACT = "10") to Synchronizing (Ni_PSR (i=0-3).ACT = "00") at the next sample point
- Disabled (Ni_CCCR (i=0-3).PXHD = 1): The M_CAN will treat a recessive res bit as an form error and will respond with an error frame

CAN FD operation is enabled by programming Ni_CCCR (i=0-3).FDOE. In case Ni_CCCR (i=0-3).FDOE = 1, transmission and reception of CAN FD frames is enabled. Transmission and reception of classical CAN frames is always possible. Whether a CAN FD frame or a classical CAN frame is transmitted can be configured through bit FDF in the respective Tx Buffer element. With Ni_CCCR (i=0-3).FDOE = 0, received frames are interpreted as classical CAN frames, which leads to the transmission of an error frame when receiving a CAN FD frame. When CAN FD operation is disabled, no CAN FD frames are transmitted even if bit FDF of a Tx Buffer element is set. Ni_CCCR (i=0-3).FDOE and Ni_CCCR (i=0-3).BRSE can only be changed while Ni_CCCR (i=0-3).INIT and Ni_CCCR (i=0-3).CCE are both set. With Ni_CCCR (i=0-3).FDOE = 0, the setting of bits FDF and BRS is ignored and frames are transmitted in Classical CAN format. With Ni_CCCR (i=0-3).FDOE = 1 and Ni_CCCR (i=0-3).BRSE = 0, only bit FDF of a Tx Buffer element is evaluated. With Ni_CCCR (i=0-3).FDOE = 1 and Ni_CCCR (i=0-3).BRSE = 1, transmission of CAN FD frames with bit rate switching is enabled. All Tx Buffer elements with bits FDF and BRS set are transmitted in CAN FD format with bit rate switching.

A mode change during CAN operation is only recommended under the following conditions:

- The failure rate in the CAN FD data phase is significantly higher than in the CAN FD arbitration phase. In this case, disable the CAN FD bit rate switching option for transmissions
- During system start-up all nodes are transmitting Classical CAN messages until it is verified that they are able to communicate in CAN FD format. If this is true, all nodes switch to CAN FD operation
- Wake-up messages in CAN partial networking have to be transmitted in classical CAN format
- End-of-line programming in case not all nodes are CAN FD capable. Non-CAN FD nodes are held in silent mode until programming has completed. Then all nodes switch back to classical CAN communication

In the CAN FD format, the coding of the DLC differs from the standard CAN format. The DLC codes 0 to 8 have the same coding as in standard CAN. The DLC codes 9 to 15, which in standard CAN all code a data field of 8 bytes, are coded according to the table below.

Table 978 Coding of DLC in CAN FD

DLC	9	10	11	12	13	14	15
Number of data bytes	12	16	20	24	32	48	64

In CAN FD frames, the bit timing will be switched inside the frame, after the BRS (Bit Rate Switch) bit, if this bit is recessive. Before the BRS bit, in the CAN FD arbitration phase, the standard CAN bit timing is used as defined by the Nominal Bit Timing and Prescaler Register Ni_DBTP (i=0-3). In the following CAN FD data phase, the data phase bit timing is used as defined by the Data Bit Timing and Prescaler Register Ni_DBTP (i=0-3). The bit timing is switched back from the fast timing at the CRC delimiter or when an error is detected, whichever occurs first.

The maximum configurable bit rate in the CAN FD data phase depends on the CAN clock frequency (asynchronous clock). Example: with a CAN clock frequency of 20 MHz and the shortest configurable bit time of 4 tq, the bit rate in the data phase is 5 Mbit/s.

***Note**: MCMCAN supports up to 5 Mbit/s considering the physical medium CAN-FD timing requirements from ISO 11898-2:2016. At fM_CAN = 80 MHz, CAN-FD data phase bit rate can be extended to 8 Mbit/s, while the bit asymmetry effect from CAN-FD transceiver, physical layer network topology and any other dependencies, has to be considered by the user.*

In both data frame formats, CAN FD long and CAN FD fast, the value of the bit ESI (Error Status Indicator) is determined by the transmitter's error state at the start of the transmission. If the transmitter is error passive, ESI is transmitted recessive, else it is transmitted dominant.

21.5.3.1.4 Bus off recovery

The M_CAN enters Bus off state according to CAN protocol conditions specified in ISO 11898-1. The Bus off state is reported by setting Ni_PSR.BO. Additionally, the M_CAN sets Ni_CCCR.INIT to stop all CAN operation.

To restart CAN operation, the application software needs to clear Ni_CCCR.INIT. After Ni_CCCR.INIT is cleared, the M_CAN's CAN state machine waits for the completion of the Bus off recovery Sequence according to CAN protocol (at least 128 occurrences of Bus Idle Condition, which is the detection of 11 consecutive recessive bits).

The M_CAN uses its Receive Error Counter to count the occurrences of the Bus Idle Condition. If need be, that can be monitored at Ni_ECR.REC. Additionally, each occurrence of the Bus Idle condition is flagged by Ni_PSR.LEC = 5 = Bit0Error, which triggers an interrupt (Ni_IR.PEA) when Ni_IE.PEAE is enabled.

While the Bus off recovery proceeds, the CAN activity is reported as "Synchronizing", Ni_PSR.ACT = 0 and Ni_PSR.BO remains set. The time from resetting Ni_CCCR.INIT to the clearing of Ni_PSR.BO will be (in the absence of dominant bits on the CAN bus) 1420 (11 * 129 + 1) CAN bit times plus synchronization delay between clock domains.

During Bus off recovery, the M_CAN does not receive or start transmission of messages. When a transmission is requested while the Bus off recovery proceeds, it will be started after the recovery has completed and CAN activity entered Idle state, Ni_PSR.ACT = 1.

When the Bus off recovery has completed, Ni_PSR.BO, Ni_ECR.TEC, and Ni_ECR.REC are cleared, and one CAN bit time later Ni_PSR.ACT is set to Idle.

After Ni_PSR.ACT reaches Idle, it will remain in Idle for at least one CAN bit time. The M_CAN's CAN state machine will become receiver (Ni_PSR.ACT = 2) when it samples a dominant bit during Idle state or it will become transmitter (Ni_PSR.ACT = 3) when it detects a pending transmission request during Idle state.

21.5.3.1.5 Transmitter delay compensation

During the data phase of a CAN FD transmission only one node is transmitting, all others are receivers. The length of the bus line has no impact. When transmitting through pin TXD the protocol controller receives the transmitted data from its local CAN transceiver through pin RXD. The received data is delayed by the CAN transmitter delay. In case this delay is greater than TSEG1 (time segment before sample point), a bit error is detected. In order to enable a data phase bit time that is even shorter than the transmitter delay, the delay compensation is introduced. Without transmitter delay compensation, the bit rate in the data phase of a CAN FD frame is limited by the transmitter delay.

Configuration

The CAN FD protocol unit has implemented a delay compensation mechanism to compensate the CAN transceiver's loop delay, the transmitter delay, thereby enabling transmission with higher bit rates during the CAN FD data phase independent of the delay of a specific CAN transceiver and port delay.

For the transmitter delay compensation the following boundary conditions have to be considered:

To check for bit errors during the data phase of transmitting nodes, the delayed transmit data is compared against the received data at the Secondary Sample Point (SSP). If a bit error is detected, the transmitter will react on this bit error at the next following regular sample point. During arbitration phase the delay compensation is always disabled.

The transmitter delay compensation enables configurations where the data bit time is shorter than the transmitter delay, it is described in detail in the ISO11898-1. It is enabled by setting bit Ni_DBTP (i=0-3).TDC. The received bit is compared against the transmitted bit at the Secondary Sample Point. The Secondary Sample Point position is defined as the sum of the measured delay from the M_CAN's transmit output TX through the transceiver to the receive input RX plus the transmitter delay compensation offset as configured by Ni_TDCR (i=0-3).TDCO. The transmitter delay compensation offset is used to adjust the position of the SSP inside the received bit (for example half of the bit time in the data phase). The position of the secondary sample point is rounded down to the next integer number of minimum time quanta "mtq". Ni_PSR.TDCV shows the actual transmitter delay compensation value. Ni_PSR (i=0-3).TDCV is cleared when Ni_CCCR.INIT is set and is updated at each transmission of an FD frame while Ni_DBTP (i=0-3).TDC is set.

The following boundary conditions have to be considered for the transmitter delay compensation implemented in the M_CAN:

- The sum of the measured delay from TX to RX and the configured transmitter delay compensation offset Ni_TDCR (i=0-3).TDCO has to be less than 6-bit times in the data phase. The sum of the measured delay from RX to RX and the configured transmitter delay compensation offset Ni_TDCR (i=0-3).TDCO has to be less than or equal to 127 mtq. In case this sum exceeds 127 mtq, the maximum value of 127 mtq is used for transmitter delay compensation. The data phase ends at the sample point of the CRC delimiter, that stops checking of receive bits at the secondary sample points

Measurement

If transmitter delay compensation is enabled by programming Ni_DBTP (i=0-3).TDC = '1', the measurement is started within each transmitted CAN FD frame at the falling edge of bit FDF to bit res. The measurement is stopped when this edge is seen at the receive input RX of the transmitter. The resolution of this measurement is one mtq.

The figure below describes how the transmitter loop delay is measured.

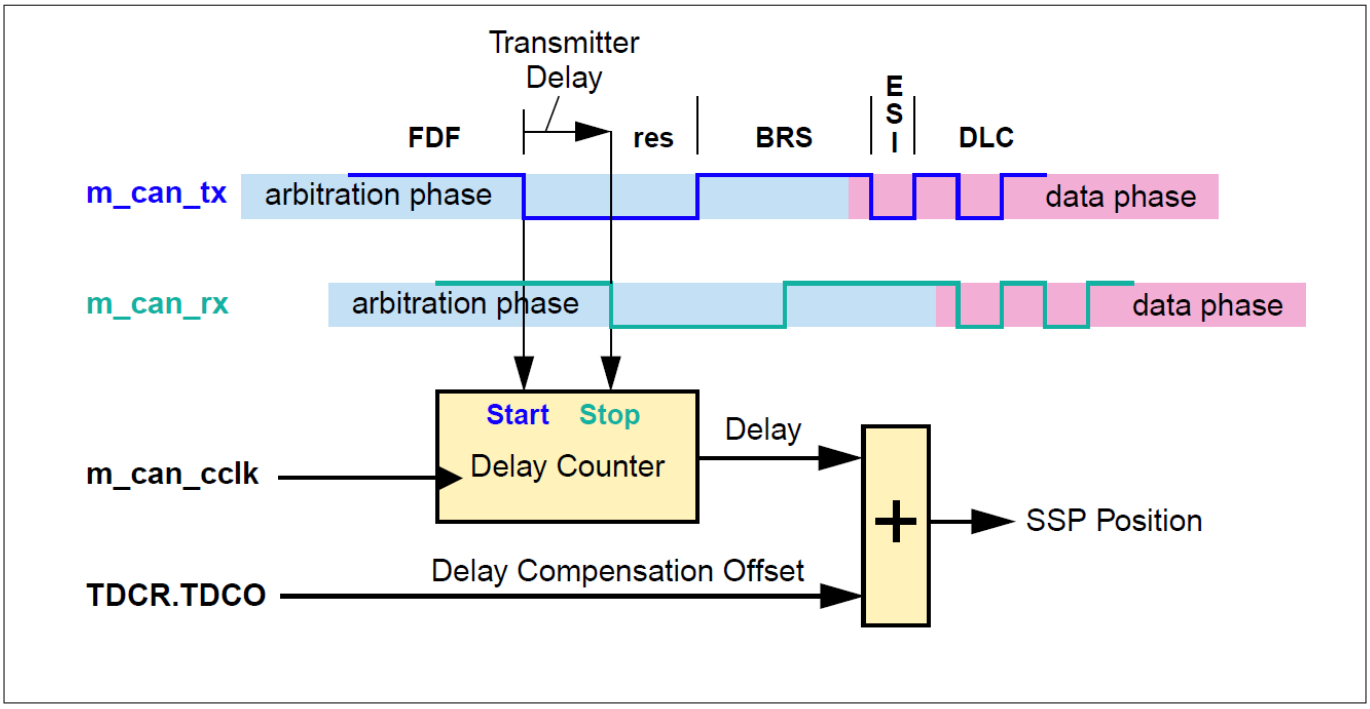


Figure 374 Transmitter delay measurement

To avoid that a dominant glitch inside the received FDF bit ends the delay compensation measurement before the falling edge of the received “res” bit, resulting in a to early Secondary Sample Point position, the use of a transmitter delay compensation filter window can be enabled by programming Ni_TDCR (i=0-3).TDCF. This defines a minimum value for the Secondary Sample Point position. Dominant edges on RX, that would result in an earlier Secondary Sample Point position are ignored for transmitter delay measurement. The measurement is stopped when the Secondary Sample Point position is at least Ni_TDCR (i=0-3).TDCF AND TX is low.

21.5.3.1.6 Restricted operation mode

In operation mode the node is able to:• Rece

ive data

- Receive remote frames, and
- Give acknowledge to valid frames

The node does not send:

- Data frames
- Remote frames
- Active error frames, or
- Overload frames

In case of an error condition or overload condition, it does not send dominant bits, instead it waits for the occurrence of bus idle to resynchronize itself to the CAN communication. The error counters (Ni_ECR (i=0-3).REC and Ni_ECR (i=0-3).TEC) are frozen while Error Logging (Ni_ECR (i=0-3).CEL) is active. The Host can set the M_CAN into Restricted Operation mode by setting bit Ni_CCCR (i=0-3).ASM. The bit can only be set by the Host when both Ni_CCCR (i=0-3).CCE and Ni_CCCR (i=0-3).INIT are set to ‘1’. The bit can be reset by the Host at any time.

Restricted Operation Mode is automatically entered when the Tx Handler was not able to read data from the Message RAM in time. To leave Restricted Operation Mode, the Host CPU has to reset Ni_CCCR (i=0-3).ASM.

The Restricted Operation Mode can be used in applications that adapt themselves to different CAN bit rates. In this case the application tests different bit rates and leaves the Restricted Operation Mode after it has received a valid frame.

21.5.3.1.7 Bus monitoring mode

The M_CAN is set in bus monitoring mode by programming Ni_CCCR (i=0-3).MON to one. In bus monitoring mode (see ISO11898-1, 10.12 Bus monitoring), the M_CAN is able to receive valid data frames and valid remote frames, but cannot start a transmission. In this mode, it sends only recessive bits on the CAN bus, if the M_CAN is required to send a dominant bit (ACK bit, overload flag, active error flag), the bit is rerouted internally so that the M_CAN monitors this dominant bit, although the CAN bus may remain in recessive state. In bus monitoring mode register TXBRP is held in reset state.

The bus monitoring mode can be used to analyze the traffic on a CAN bus without affecting it by the transmission of dominant bits. The following figure shows the connection of signals TXD and RXD to the M_CAN in bus monitoring mode.

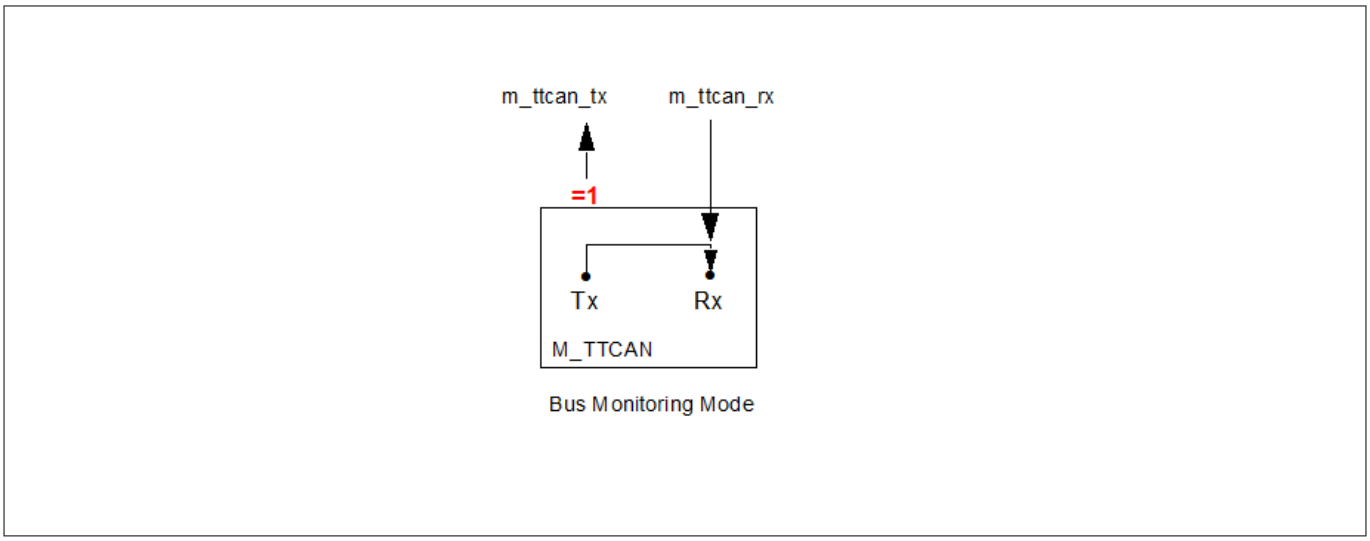


Figure 375 Pin control in bus monitoring mode

21.5.3.1.8 Disabled automatic retransmission

According to the CAN Specification (see ISO11898-1, 6.3.3 Recovery Management), the M_CAN provides means for automatic retransmission of frames that have lost arbitration or that have been disturbed by errors during transmission. By default automatic retransmission is enabled. To support time-triggered communication as described in ISO 11898-1 in Chapter 9.2, the automatic retransmission may be disabled through Ni_CCCR (i=0-3).DAR.

Frame transmission in DAR mode

In DAR mode all transmissions are automatically canceled after they started on the CAN bus. A Tx Buffer's Tx Request Pending bit Ni_TXBRP (i=0-3).TRPx is reset after successful transmission, when a transmission has not yet been started at the point of cancellation, has been aborted due to lost arbitration, or when an error occurred during frame transmission.

- Successful transmission:
  - Corresponding Tx Buffer Transmission Occurred bit Ni_TXBTO (i=0-3).TOx set
  - Corresponding Tx Buffer Cancellation Finished bit Ni_TXBCF (i=0-3).CFx not set

- Successful transmission in spite of cancellation:
  - Corresponding Tx Buffer Transmission Occurred bit Ni_TXBTO (i=0-3.TOx set
  - Corresponding Tx Buffer Cancellation Finished bit Ni_TXBCF (i=0-3).CFx set
- Arbitration lost or frame transmission disturbed:
  - Corresponding Tx Buffer Transmission Occurred bit Ni_TXBTO (i=0-3.TOx not set
  - Corresponding Tx Buffer Cancellation Finished bit Ni_TXBCF (i=0-3).CFx set

In case of a successful frame transmission, and if storage of Tx events is enabled, a Tx Event FIFO element is written with Event Type ET = “10” (transmission in spite of cancellation).

21.5.3.1.9 Power down (sleep mode)

The M_CAN can be set into power down mode controlled by input signal clock stop request or by CC Control Register Ni_CCCR (i=0-3).CSR. As long as the clock stop request signal is active, bit Ni_CCCR (i=0-3).CSR is read as one.

When all pending transmission requests have completed, the M_CAN waits until bus idle state is detected. Then the M_CAN sets then Ni_CCCR (i=0-3).INIT to one to prevent any further CAN transfers. Now the M_CAN acknowledges that it is ready for power down by setting Ni_CCCR (i=0-3).CSA to one. In this state, before the clocks are switched off, further register accesses can be made. A write access to Ni_CCCR (i=0-3).INIT will have no effect. Now the module clocks may be switched off.

To leave power down mode, the application has to turn on the module clocks before resetting CC Control Register flag Ni_CCCR (i=0-3).CSR. The M_CAN will acknowledge this by resetting Ni_CCCR (i=0-3).CSA. Afterwards, the application can restart CAN communication by resetting bit Ni_CCCR (i=0-3).INIT.

21.5.3.1.10 Test modes

To enable write access to register TEST (see Ni_TEST (i=0-3)), bit Ni_CCCR (i=0-3).TEST has to be set to one. This allows the configuration of the test modes and test functions.

Four output functions are available for the CAN transmit pin TXD by programming Ni_TEST (i=0-3).TX. Additionally to its default function – the serial data output – it can drive the CAN Sample Point signal to monitor the M_CAN’s bit timing and it can drive constant dominant or recessive values. The actual value at pin RXD can be read from Ni_TEST (i=0-3).RX. Both functions can be used to check the CAN bus’ physical layer.

Due to the synchronization mechanism between CAN clock and Host clock domain, there may be a delay of several Host clock periods between writing to Ni_TEST (i=0-3).TX until the new configuration is visible at output pin TXD. This applies also when reading input pin RXD through Ni_TEST (i=0-3).RX.

***Note**: Test modes should be used for production tests or self test only. The software control for pin TXD interferes with all CAN protocol functions. It is not recommended to use test modes for application.*

Node external loop back mode

The M_CAN can be set in external loop back Mode by programming Ni_TEST (i=0-3).LBCK to one. In loop back mode, the M_CAN treats its own transmitted messages as received messages and stores them (if they pass acceptance filtering) into an Rx Buffer or an Rx FIFO0/1. The following figure shows the connection of TXD and RXD to the M_CAN in external loop back mode.

This mode is provided for hardware self-test. To be independent from external stimulation, the M_CAN ignores acknowledge errors (recessive bit sampled in the acknowledge slot of a data or remote frame) in loop back mode. In this mode the M_CAN performs an internal feedback from its Tx output to its Rx input. The actual value of the RXD input pin is disregarded by the M_CAN. The transmitted messages can be monitored at the TXD pin.

Node internal loop back mode

Internal loop back mode is entered by programming bits Ni_TEST (i=0-3).LBCK and Ni_CCCR (i=0-3).MON to one. This mode can be used for a “Hot Selftest”, meaning the M_CAN can be tested without affecting a running CAN system connected to the pins TXD and RXD. In this mode pin RXD is disconnected from the M_CAN and pin TXD is held recessive. The following figure shows the connection of TXD and RXD to the M_CAN in case of internal loop back mode.

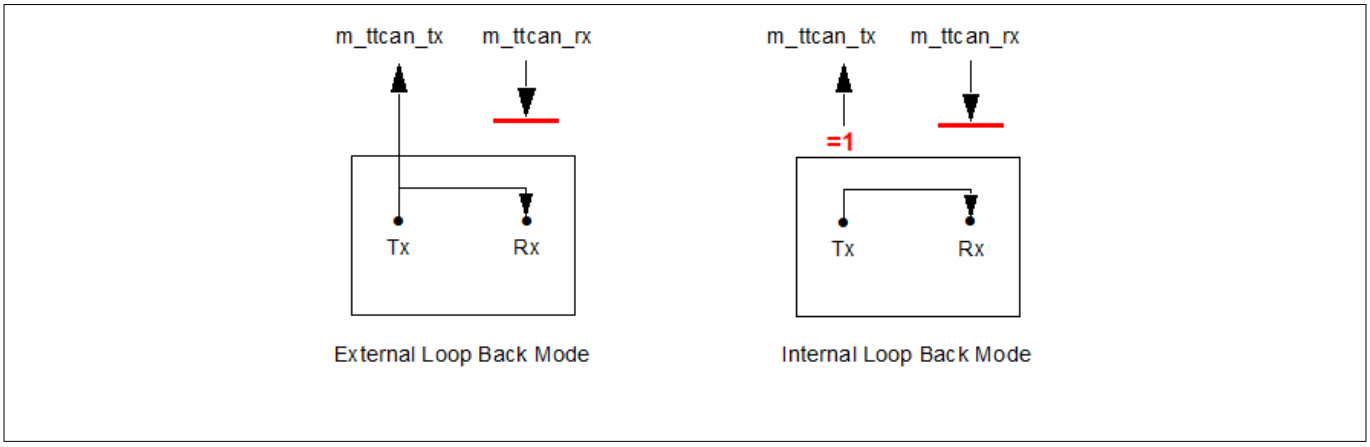


Figure 376 Pin control in loop back modes

21.5.3.2 Timestamp generation

M_CAN supports two methods for frame timestamping:

1. Internal M_CAN timestamping feature 16-bit wide timestamps, using a selectable clock source, compatible with previous versions of MCMCAN module
2. Hardware timestamping 32-bit wide timestamps using TSU module, according to CiA 603

21.5.3.2.1 Internal timestamp generation

For timestamp generation the M_CAN supplies a 16-bit wrap-around counter. A prescaler Ni_TSCC (i=0-3).TCP can be configured to clock the counter in multiples of CAN bit times (1…16). The counter is readable through Ni_TSCV (i=0-3).TSC. A write access to register TSCV resets the counter to zero. When the timestamp counter wraps around interrupt flag Ni_IR (i=0-3).TSW is set.

On start of frame reception/transmission the counter value is captured and stored into the timestamp section of an Rx Buffer/Rx FIFO (RXTS[15:0]) or Tx Event FIFO (TXTS[15:0]) element.

By programming bit Ni_TSCC (i=0-3).TSS an external 16-bit timestamp can be used. The external timer can be controlled using the Ni_TIMER_CCR (i=0-3) register. The external source for the timer is selected by programming the Ni_TIMER_CCR (i=0-3).TRGIGSRC. The external timer can be started by setting Ni_TIMER_CCR (i=0-3).STSTART. It can be reset to zero by setting Ni_TIMER_CCR (i=0-3).STRESET. A write of ‘1b’ to Ni_TIMER_CCR (i=0-3).STSTART is effective only when Ni_TIMER_CCR (i=0-3).STRESET is ‘0b’ that is, the external timer cannot be started when Ni_TIMER_CCR (i=0-3).STRESET is ‘1b’.

21.5.3.2.2 Timestamping unit (TSU)

This section gives a description of configuring and using the timestamping unit for hardware captured timestamps of received or transmitted frames.

To enable hardware timestamping together with the TSU, M_CAN’s configuration bit Ni_CCCR (i=0-3).UTSU has to be configured to 1. In case Ni_CCCR (i=0-3).UTSU = ‘0’ (default value), the M_CAN’s internal 16-bit timestamp counter is used, if enabled.

TSU Configuration

The TSU is configured by Ni_TSU_TSCFG. This register is only writable while Ni_CCCR (i=0-3).CCE = ‘1’. With Ni_TSU_TSCFG.TSUE the TSU is enabled. Both Ni_CCCR (i=0-3).UTSU and Ni_TSU_TSCFG.TSUE shall be set to '1', for proper functioning of timestamping using TSU. By default the TSU is disabled. The Timebase Prescaler Ni_TSU_TSCFG.TBPRE[7:0] is used to adapt the count rate of the TSU’s internal counter, if enabled, to the requirements of the application.

The TSU provides the user the possibility of using an internal timebase or an external timebase input for timestamping, selected through Ni_TSU_TSCFG (i=1-3).TBCS. When this bit is set to ‘0’, the internal free running timebase counter is used, and if this bit is set to ‘1’, the timestamp value is taken from the external timebase input. CAN node 0 has no external timebase input. CAN node 1-3 external timebase input is the output of the CAN node 0 internal timebase. This gives the possibility of synchronized timestamping accross the nodes. The following figure shows the connection and its configuration.

Ni_TSU_TSCFG.SCP configures the point in time when a timestamp is captured for Sync messages (0: at EOF, 1: at SOF of the received or transmitted Sync message).

When Ni_TSU_TSCFG.SCP = '0', the timestamp is captured directly from the Actual Timebase Ni_TSU_ATB (i=0-3) to the timestamp register Ni_TSU_TSm (i=0-3;m=0-15) when the message gets valid at EOF. This configuration is intended for AUTOSAR conformant applications.

When Ni_TSU_TSCFG.SCP = '1', the Actual Timebase Ni_TSU_ATB (i=0-3) is captured into a temporary buffer at each SOF. When the message gets valid at EOF the timestamp is copied from the temporary buffer to the timestamp register Ni_TSU_TSm (i=0-3;m=0-15).

The Timestamp Pointer is incremented after capturing the timestamp.

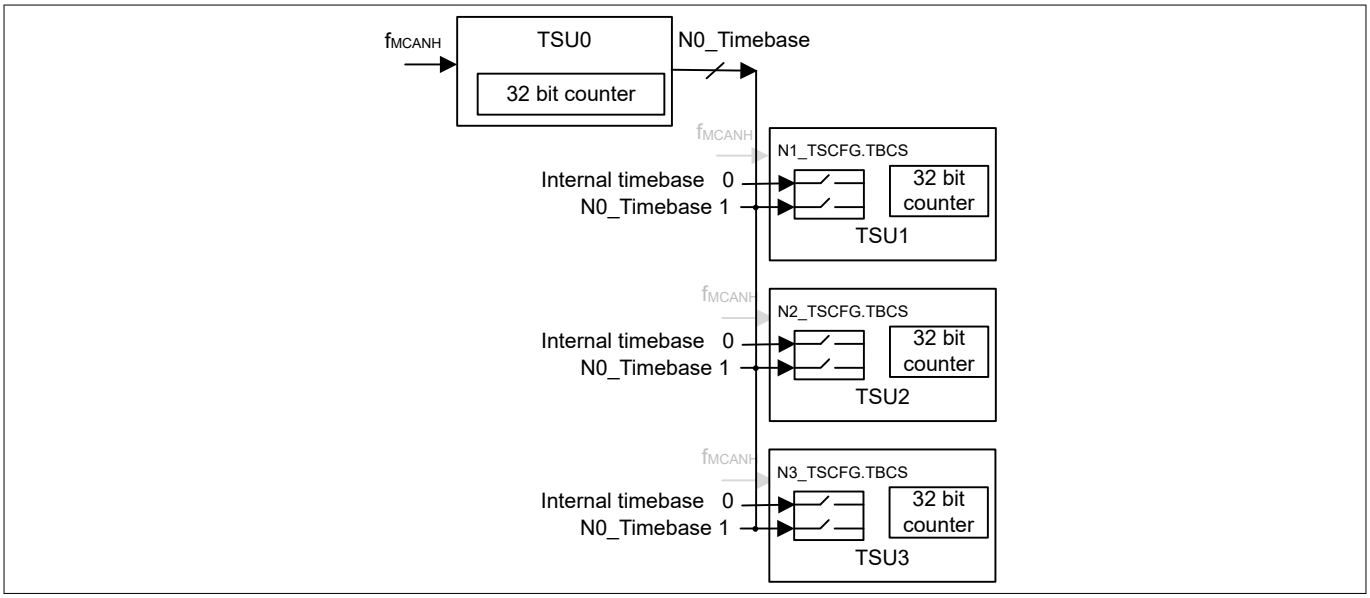


Figure 377 TSU connectivity across the nodes

Reception of timestamped messages

To configure the M_CAN for reception of timestamped messages, a Standard/Extended Message ID Filter Element has to be set up by configuration of STDMSGk_S0.SSYNC = ‘1’ respectively EXTMSGk_F1.ESYNC = ‘1’. In case the filter element matches the received frame and the received frame is valid, M_CAN output at the EOF triggers the storage of the value of the TSU’s Actual Timebase ATB to the TSU's timestamp register selected by the Timestamp Pointer Ni_TSU_TSS2 (i=0-3).TSP[3:0]. At the same pulse, the M_CAN captures the Timestamp Pointer value and this value is written to the M_CAN’s RXMSGk_R1B.RXTSP[3:0] of the related Rx Buffer or Rx FIFO element. R1B.TSC is set to ‘1’ when a timestamp has been captured by the TSU and RXMSGk_R1B.RXTSP[3:0] holds a valid timestamp pointer.

Transmission of timestamped messages

In case bits TXMSGk_T1.TSCE and TXMSGk_T1.EFC of a Tx Buffer element are set, a successful transmission of that Tx Buffer generates a pulse that triggers the storage of the value of the TSU's Actual Timebase ATB to the TSU's timestamp register selected by the Timestamp Pointer Ni_TSU_TSS2 (i=0-3).TSP[3:0]. At the same pulse, the M_CAN captures the Timestamp Pointer value and this value is written to the M_CAN's TXEVENTk_E1B.TXTSP[3:0] of the related Tx Event FIFO. TXEVENTk_E1B.TSC is set to '1' when a timestamp has been captured by the TSU and TXEVENTk_E1B.TXTSP[3:0] holds a valid timestamp pointer element.

Handling of timestamps

The TSU stores sixteen 32-bit timestamps. They are written to TS0 to TS15 in a cyclic manner.

Each timestamp register has assigned two status bits:

- Ni_TSU_TSS1 (i=0-3).TSNn is set whenever a timestamp was stored to the related timestamp register Ni_TSU_TSm (i=0-3;m=0-15)
- Ni_TSU_TSS1 (i=0-3).TSLn is set when a new timestamp was stored to the related timestamp register TSn before the previously store timestamp has been read out

Reading a Timestamp register resets the related TSS1 bits, if Ni_PORTCTRL(i=0-3).DELE is set.

21.5.3.3 Timeout counter

To signal timeout conditions for Rx FIFO 0, Rx FIFO 1, and the Tx Event FIFO the M_CAN supplies a 16-bit Timeout Counter. It operates as down-counter and uses the same prescaler controlled by Ni_TSCC (i=0-3).TCP as the Timestamp Counter. The Timeout Counter is configured by register Ni_TOCC (i=0-3). The actual counter value can be read from Ni_TOCV (i=0-3).TOC.

The Timeout Counter can only be started while Ni_CCCR (i=0-3).INIT = '0'. It is stopped when Ni_CCCR (i=0-3).INIT = '1', for example when the M_CAN enters Bus_Off state.

The operation mode is selected by Ni_TOCC (i=0-3).TOS. When operating in Continuous Mode, the counter starts when Ni_CCCR.INIT is reset. A write to Ni_TOCV (i=0-3) presets the counter to the value configured by Ni_TOCC (i=0-3).TOP and continues down-counting.

When the Timeout Counter is controlled by one of the FIFOs, an empty FIFO presets the counter to the value configured by Ni_TOCC (i=0-3).TOP. Down-counting is started when the first FIFO element is stored. Writing to Ni_TOCV (i=0-3) has no effect.

When the counter reaches zero, interrupt flag Ni_IR (i=0-3).TOO is set. In Continuous Mode, the counter is immediately restarted at Ni_TOCC (i=0-3).TOP.

***Note**: The clock signal for the Timeout Counter is derived from the CAN Core's sample point signal. Therefore the point in time where the Timeout Counter is decremented may vary due to the synchronization/ re-synchronization mechanism of the CAN Core. If the baud rate switch feature in CAN FD is used, the timeout counter is clocked differently in arbitration and data field.*

21.5.3.4 Rx handling

The Rx Handler controls the acceptance filtering, the transfer of received messages to the Rx Buffers or to one of the two Rx FIFOs, as well as the Rx FIFO's Put and Get Indices.

21.5.3.4.1 Acceptance filtering

The M_CAN offers the possibility to configure two sets of acceptance filters, one for standard identifiers and one for extended identifiers. These filters can be assigned to an Rx Buffer or to Rx FIFO 0,1. For acceptance filtering each list of filters is executed from element #0 until the first matching element. Acceptance filtering stops at the first matching element. The following filter elements are not evaluated for this message.

The main features are:

- Each filter element can be configured as:
  - Range filter (from - to)
  - Filter for one or two dedicated IDs, or
  - Classic bit mask filter
- Each filter element is configurable for acceptance or rejection filtering
- Each filter element can be enabled and disabled individually
- Filters are checked sequentially, execution stops with the first matching filter element

Related configuration registers are:

- Global Filter Configuration Ni_GFC (i=0-3)
- Standard ID Filter Configuration Ni_SIDFC (i=0-3)
- Extended ID Filter Configuration Ni_XIDFC (i=0-3)
- Extended ID AND Mask Ni_XIDAM (i=0-3)

Depending on the configuration of the filter element (STDMSGk_S0.SFEC or EXTMSGk_F0.EFEC) a match triggers one of the following actions:

- Store received frame in FIFO 0 or FIFO 1
- Store received frame in Rx Buffer
- Store received frame in Rx Buffer
- Reject received frame
- Set High Priority Message interrupt flag Ni_IR (i=0-3).HPM
- Set High Priority Message interrupt flag Ni_IR (i=0-3).HPM and store received frame in FIFO 0 or FIFO 1

Acceptance filtering is started after the complete identifier has been received. After acceptance filtering has completed, and if a matching Rx Buffer or Rx FIFO has been found, the Message Handler starts writing the received message data in portions of 32 bits to the matching Rx Buffer or Rx FIFO. If the CAN protocol controller has detected an error condition (for example CRC error), this message is discarded with the following impact on the affected Rx Buffer or Rx FIFO:

Rx Buffer New Data flag of matching Rx Buffer is not set, but Rx Buffer (partly) overwritten with received data.

Rx FIFO Put index of matching Rx FIFO is not updated, but related Rx FIFO element (partly) overwritten with received data. In case the matching Rx FIFO is operated in overwrite mode, the boundary conditions described in Rx FIFO Overwrite Mode (see related information) have to be considered.

For error type see Ni_PSR (i=0-3).LEC respectively Ni_PSR (i=0-3).FLEC.

***Note**: When an accepted message is written to one of the two Rx FIFOs, or into an Rx Buffer, the unmodified received identifier is stored independent of the filter(s) used. The result of the acceptance filter process is strongly depending on the sequence of configured filter elements.*

Related information

Rx FIFO overwrite mode on page 3952

Range filter

The filter matches for all received frames with Message IDs in the range defined by STDMSGk_S0.SFID1/SFID2 respectively EXTMSGk_F0.EFID1/EFID2.

There are two possibilities when range filtering is used together with extended frames:

- EXTMSGk_F1.EFT = 00: The Message ID of received frames is ANDed with the Extended ID AND Mask (Ni_XIDAM) before the range filter is applied
- EXTMSGk_F1.EFT = 11: The Extended ID AND Mask (Ni_XIDAM) is not used for range filtering

Filter for specific IDs

A filter element can be configured to filter for one or two specific Message IDs. To filter for one specific Message ID, the filter element has to be configured with STDMSGk_S0.SFID1 = STDMSGk_S0.SFID2 respectively EXTMSGk_F0.EFID1 = EXTMSGk_F1.EFID2.

Classic bit mask filter

Classic bit mask filtering is intended to filter groups of Message IDs by masking single bits of a received Message ID. With classic bit mask filtering STDMSGk_S0.SFID1/EXTMSGk_F0.EFID1 is used as Message ID filter, while STDMSGk_S0.SFID2/EXTMSGk_F1.EFID2 is used as filter mask.

A zero bit at the filter mask will mask out the corresponding bit position of the configured ID filter, for example the value of the received Message ID at that bit position is not relevant for acceptance filtering. Only those bits of the received Message ID where the corresponding mask bits are one are relevant for acceptance filtering.

In case all mask bits are one, a match occurs only when the received Message ID and the Message ID filter are identical. If all mask bits are zero, all Message IDs match.

Standard message ID filtering

The following figure shows the flow for standard message ID (11-bit identifier) filtering. The standard message ID Filter element is described in related information.

Controlled by the global filter configuration Ni_GFC (i=0-3) and the standard ID filter configuration Ni_SIDFC message ID, remote transmission request bit (RTR), and the identifier extension bit (IDE) of received frames are compared against the list of configured filter elements.

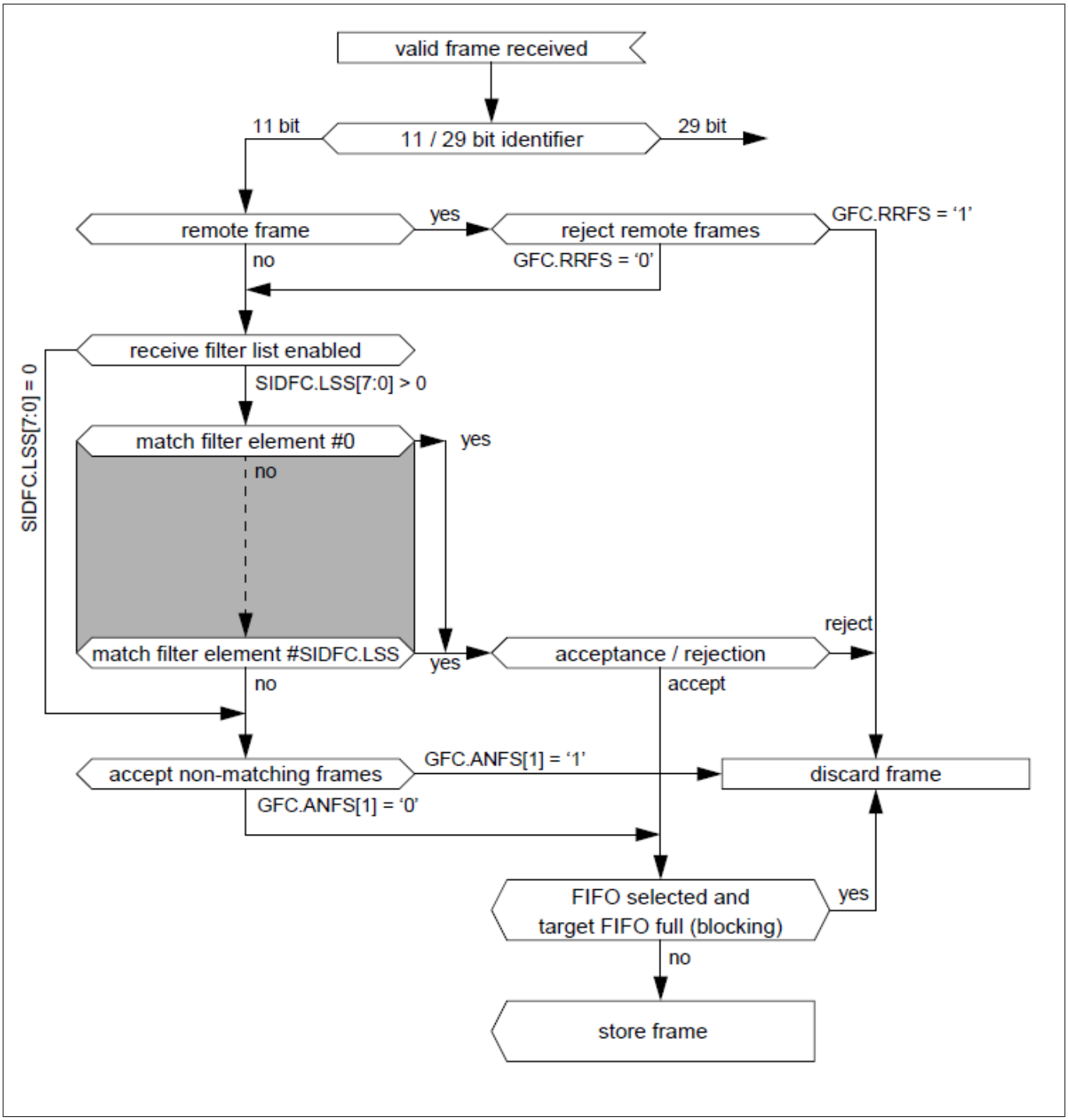


Figure 378 Standard message ID filter path

Related information

Standard message ID filter element on page 3932

Extended message ID filtering

The following figure shows the flow for extended message ID (29-bit identifier) filtering. The extended message ID filter element is described in related information.

Controlled by the Global Filter Configuration Ni_GFC and the Extended ID Filter Configuration Ni_XIDFC Message ID, Remote Transmission Request bit (RTR), and the Identifier Extension bit (IDE) of received frames are compared against the list of configured filter elements.

The Extended ID AND Mask Ni_XIDAM is ANDed with the received identifier before the filter list is executed.

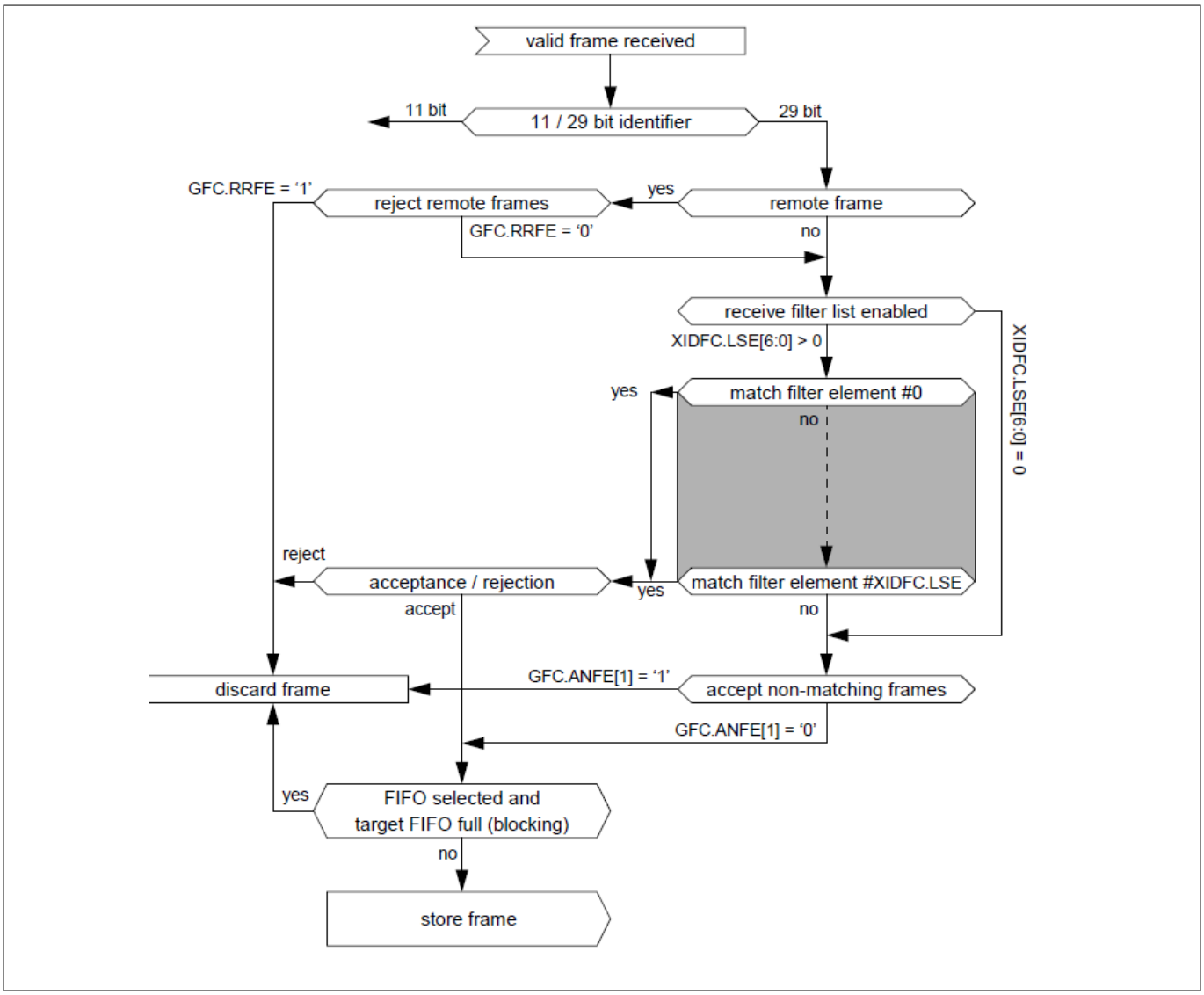


Figure 379 Extended message ID filter path

Related information

Standard message ID filter element on page 3932

21.5.3.4.2 Rx FIFOs

Rx FIFO 0 and Rx FIFO 1 can be configured to hold up to 64 elements each. Configuration of the two Rx FIFOs is done by registers Ni_RXF0C (i=0-3) and Ni_RXF1C (i=0-3).

Received messages that passed acceptance filtering are transferred to the Rx FIFO as configured by the matching filter element. For a description of the filter mechanisms available for Rx FIFO 0 and Rx FIFO 1 see Acceptance Filtering in related information. The Rx FIFO element is described in Rx Buffer and FIFO Element in related information.

To avoid an Rx FIFO overflow, the Rx FIFO watermark can be used. When the Rx FIFO fill level reaches the Rx FIFO watermark configured by RXFnC.FnWM, interrupt flag Ni_IR.RFnW is set. When the Rx FIFO Put Index

reaches the Rx FIFO Get Index an Rx FIFO Full condition is signalled by Ni_RXFnS.FnF. In addition interrupt flag Ni_IR (i=0-3).RFnF is set.

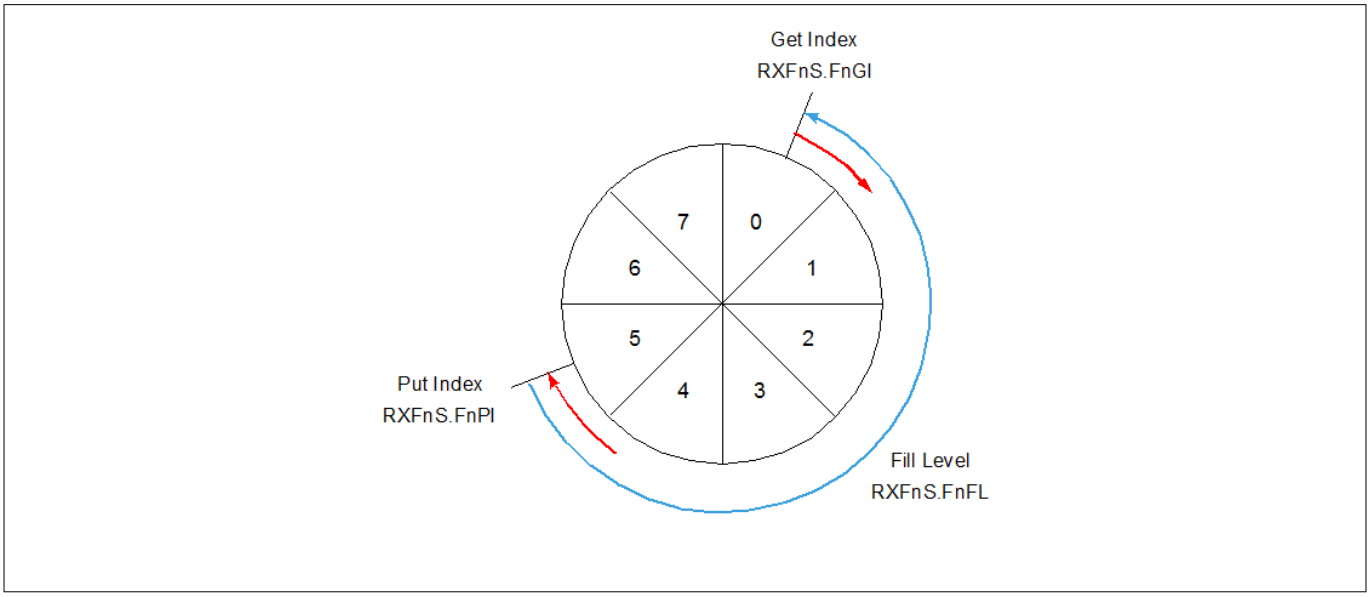


Figure 380 Rx FIFO status

When reading from an Rx FIFO, Rx FIFO Get Index Ni_RXFnS.FnGI • FIFO Element Size has to be added to the corresponding Rx FIFO start address Ni_RXFnC.FnSA.

Table 979 Rx buffer/FIFO element size

Ni_RXESC.RBDS[2:0] Ni_RXESC.FnDS[2:0]	Data field [bytes]	FIFO element size [RAM words]
000	8	4
001	12	5
010	16	6
011	20	7
100	24	8
101	32	10
110	48	14
111	64	18

Related information

Acceptance filtering on page 3946
Rx buffer and FIFO element on page 3929

Rx FIFO blocking mode

The Rx FIFO blocking mode is configured by Ni_RXFnC.FnOM = '0'. This is the default operation mode for the Rx FIFOs.

When an Rx FIFO full condition is reached (Ni_RXFnS.FnPI = Ni_RXFnS.FnGI), no further messages are written to the corresponding Rx FIFO until at least one message has been read out and the Rx FIFO Get Index has been incremented. An Rx FIFO full condition is signalled by Ni_RXFnS.FnF = '1'. In addition interrupt flag Ni_IR (i=0-3).RFnF is set.

In case a message is received while the corresponding Rx FIFO is full, this message is discarded and the message lost condition is signalled by Ni_RXFnS.RFnL = ‘1’. In addition interrupt flag Ni_IR (i=0-3).RFnL is set.

Rx FIFO overwrite mode

The Rx FIFO overwrite mode is configured by Ni_RXFnC.FnOM = ‘1’.

When an Rx FIFO full condition (Ni_RXFnS.FnPI = RXFnS.FnGI) is signalled by Ni_RXFnS.FnF = ‘1’, the next message accepted for the FIFO will overwrite the oldest FIFO message. Put and get index are both incremented by one.

When an Rx FIFO is operated in overwrite mode and an Rx FIFO full condition is signalled, reading of the Rx FIFO elements should start at least at get index + 1. The reason for that is, that it might happen, that a received message is written to the Message RAM (put index) while the CPU is reading from the Message RAM (get index). In this case inconsistent data may be read from the respective Rx FIFO element. Adding an offset to the get index when reading from the Rx FIFO avoids this problem. The offset depends on how fast the CPU accesses the Rx FIFO. The following figure shows an offset of two with respect to the get index when reading the Rx FIFO. In this case the two messages stored in element 1 and 2 are lost.

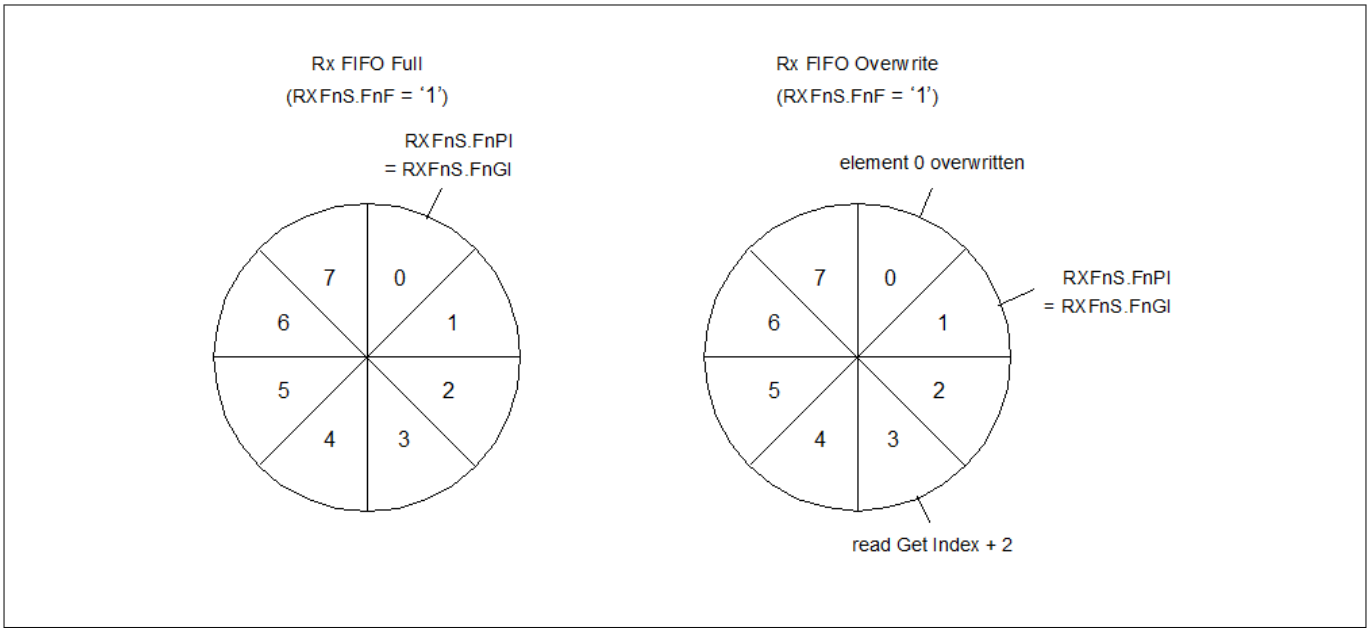


Figure 381 Rx FIFO overflow handling

After reading from the Rx FIFO, the number of the last element read has to be written to the Rx FIFO Acknowledge Index Ni_RXFnA.FnA. This increments the get index to that element number. In case the put index has not been incremented to this Rx FIFO element, the Rx FIFO full condition is reset (Ni_RXFnS.FnF = ‘0’).

21.5.3.4.3 Dedicated Rx buffers

The M_CAN supports up to 64 dedicated Rx Buffers. The start address of the dedicated Rx Buffer section is configured by Ni_RXBC (i=0-3).RBSA.

For each Rx Buffer a Standard or Extended Message ID Filter Element with STDMSGk_S0.SFEC/ EXTMSGk_F0.EFEC = “111” and STDMSGk_S0.SFID2/EXTMSGk_F1.EFID2[10:9] = “00” has to be configured (see related information).

After a received message has been accepted by a filter element, the message is stored into the Rx Buffer in the Message RAM referenced by the filter element. The format is the same as for an Rx FIFO element. In addition the flag Ni_IR.DRX (Message stored in dedicated Rx Buffer) in the interrupt register is set.

Table 980 Example filter configuration for Rx buffers

Filter element	SFID1[10:0] EFID1[28:0]	SFID2[10:9] EFID2[10:9]	SFID2[5:0] EFID2[5:0]
0	ID message 1	00	00 0000
1	ID message 2	00	00 0001
2	ID message 3	00	00 0010

After the last word of a matching received message has been written to the Message RAM, the respective New Data flag in register Ni_NDAT1 (i=0-3),Ni_NDAT2 (i=0-3) is set. As long as the New Data flag is set, the respective Rx Buffer is locked against updates from received matching frames. The New Data flags have to be reset by the Host by writing a '1' to the respective bit position.

While an Rx Buffer's New Data flag is set, a Message ID Filter Element referencing this specific Rx Buffer will not match, causing the acceptance filtering to continue. Following Message ID Filter Elements may cause the received message to be stored into another Rx Buffer, or into an Rx FIFO, or the message may be rejected, depending on filter configuration.

Related information

Standard message ID filter element on page 3932
Extended message ID filter element on page 3932

Rx buffer handling

- Reset interrupt flag Ni_IR (i=0-3).DRX
- Read new data registers
- Read messages from Message RAM
- Reset new data flags of processed messages

21.5.3.5 Tx handling

The Tx Handler handles transmission requests for the dedicated Tx Buffers, the Tx FIFO, and the Tx Queue. It controls the transfer of transmit messages to the CAN Core, the Put and Get Indices, and the Tx Event FIFO. Up to 32 Tx Buffers can be set up for message transmission. The CAN mode for transmission (Classic CAN or CAN FD) can be configured separately for each Tx Buffer element. The Tx Buffer element is described in related information. The following table describes the possible configurations for frame transmission.

Table 981 Possible configuration for frame transmission

CCCR		Tx buffer element		Frame transmission
BRSE	FDOE	FDF	BRS	
Ignored	0	Ignored	Ignored	Classical CAN
0	1	0	Ignored	Classical CAN
0	1	1	Ignored	FD without bit rate switching
1	1	0	Ignored	Classical CAN
1	1	1	0	FD without bit rate switching
1	1	1	1	FD with bit rate switching

***Note**: AUTOSAR requires at least three Tx Queue Buffers and support of transmit cancellation*

The Tx Handler starts a Tx scan to check for the highest priority pending Tx request (Tx Buffer with lowest Message ID) when the Tx Buffer Request Pending register Ni_TXBRP (i=0-3) is updated, or when a transmission has been started.

Related information

Tx buffer element on page 3930

21.5.3.5.1 Transmit pause

The transmit pause feature is intended for use in CAN systems where the CAN message identifiers are (permanently) specified to specific values and cannot easily be changed. These message identifiers may have a higher CAN arbitration priority than other defined messages, while in a specific application their relative arbitration priority should be inverse. This may lead to a case where one ECU sends a burst of CAN messages that cause another ECU's CAN messages to be delayed because that other messages have a lower CAN arbitration priority.

If for example CAN ECU-1 has the transmit pause feature enabled and is requested by its application software to transmit four messages, it will, after the first successful message transmission, wait for two CAN bit times of bus idle before it is allowed to start the next requested message. If there are other ECUs with pending messages, those messages are started in the idle time, they would not need to arbitrate with the next message of ECU-1. After having received a message, ECU-1 is allowed to start its next transmission as soon as the received message releases the CAN bus.

The transmit pause feature is controlled by bit Ni_CCCR (i=0-3).TXP. If the bit is set, the M_CAN will, each time it has successfully transmitted a message, pause for two CAN bit times before starting the next transmission. This enables other CAN nodes in the network to transmit messages even if their messages have lower prior identifiers. Default is transmit pause disabled (Ni_CCCR (i=0-3).TXP = '0').

This feature loosens up burst transmissions coming from a single node and it protects against "babbling idiot" scenarios where the application program erroneously requests too many transmissions.

21.5.3.5.2 Dedicated Tx buffers

Dedicated Tx Buffers are intended for message transmission under complete control of the Host CPU. Each dedicated Tx Buffer is configured with a specific Message ID. In case that multiple dedicated Tx Buffers are configured with the same Message ID, the Tx Buffer with the lowest buffer number is transmitted first. These Tx buffers shall be requested in ascending order with lowest buffer number first. Alternatively all Tx buffers configured with the same Message ID can be requested simultaneously by a single write access to Ni_TXBAR (i=0-3).

If the data section has been updated, a transmission is requested by an "Add Request" through Ni_TXBAR (i=0-3).ARn. The requested messages arbitrate internally with messages from an optional Tx FIFO or Tx Queue and externally with messages on the CAN bus, and are sent out according to their Message ID.

A dedicated Tx Buffer allocates Element Size 32-bit words in the Message RAM (see related information). Therefore the start address of a dedicated Tx Buffer in the Message RAM is calculated by adding transmit buffer index (0…31) • Element Size to the Tx Buffer Start Address Ni_TXBC (i=0-3).TBSA.

Table 982 Tx buffer/FIFO/queue element size

Ni_TXESC.TBDS[2:0]	Data field [bytes]	Element size [RAM words]
000	8	4
001	12	5

(table continues...)

Table 982 (continued) Tx buffer/FIFO/queue element size

Ni_TXESC.TBDS[2:0]	Data field [bytes]	Element size [RAM words]
010	16	6
011	20	7
100	24	8
101	32	10
110	48	14
111	64	18

Related information

Tx FIFO on page 3956

21.5.3.5.3 Tx FIFO

Tx FIFO operation is configured by programming Ni_TXBC (i=0-3).TFQM to '0'. Messages stored in the Tx FIFO are transmitted starting with the message referenced by the Get Index Ni_TXFQS (i=0-3).TFGI. After each transmission the Get Index is incremented cyclically until the Tx FIFO is empty. The Tx FIFO enables transmission of messages with the same Message ID from different Tx Buffers in the order these messages have been written to the Tx FIFO. The M_CAN calculates the Tx FIFO Free Level Ni_TXFQS (i=0-3).TFFL as difference between Get and Put Index. It indicates the number of available (free) Tx FIFO elements.

New transmit messages have to be written to the Tx FIFO starting with the Tx Buffer referenced by the Put Index Ni_TXFQS (i=0-3).TFQPI. An "Add Request" increments the Put Index to the next free Tx FIFO element. When the Put Index reaches the Get Index, Tx FIFO Full (Ni_TXFQS (i=0-3).TFQF = '1') is signalled. In this case no further messages should be written to the Tx FIFO until the next message has been transmitted and the Get Index has been incremented.

When a single message is added to the Tx FIFO, the transmission is requested by writing a '1' to the TXBAR bit related to the Tx Buffer referenced by the Tx FIFO's Put Index.

When multiple (n) messages are added to the Tx FIFO, they are written to n consecutive Tx Buffers starting with the Put Index. The transmissions are then requested by Ni_TXBAR (i=0-3). The Put Index is then cyclically incremented by n. The number of requested Tx buffers should not exceed the number of free Tx Buffers as indicated by the Tx FIFO Free Level.

When a transmission request for the Tx Buffer referenced by the Get Index is cancelled, the Get Index is incremented to the next Tx Buffer with pending transmission request and the Tx FIFO Free Level is recalculated. When transmission cancellation is applied to any other Tx Buffer, the Get Index and the FIFO Free Level remain unchanged.

A Tx FIFO element allocates Element Size 32-bit words in the Message RAM (see table in related information). Therefore the start address of the next available (free) Tx FIFO Buffer is calculated by adding Tx FIFO/Queue Put Index Ni_TXFQS (i=0-3).TFQPI (0…31) • Element Size to the Tx Buffer Start Address Ni_TXBC (i=0-3).TBSA.

Related information

Dedicated Tx buffers on page 3954

21.5.3.5.4 Tx queue

Tx Queue operation is configured by programming Ni_TXBC (i=0-3).TFQM to '1'. Messages stored in the Tx Queue are transmitted starting with the message with the lowest Message ID (highest priority). In case that multiple Tx Queue buffers are configured with the same Message ID, the transmission order depends on numbers of the buffers where the messages were stored for transmission. As these buffer numbers depend on the then current states of the Put Index, a prediction of the transmission order is not possible.

New messages have to be written to the Tx Buffer referenced by the Put Index Ni_TXFQS (i=0-3).TFQPI. The Put Index always points to that free buffer of the Tx Queue with the lowest buffer number. In case that the Tx Queue is full (Ni_TXFQS (i=0-3).TFQF = "1"), the Put Index is not valid and no further message should be written to the Tx Queue until at least one of the requested messages has been sent out or a pending transmission request has been cancelled.

The application may use register TXBRP instead of the Put Index and may place messages to any Tx Buffer without pending transmission request.

A Tx Queue Buffer allocates Element Size 32-bit words in the Message RAM (see table in related information). Therefore the start address of the next available (free) Tx Queue Buffer is calculated by adding Tx FIFO/Queue Put Index Ni_TXFQS (i=0-3).TFQPI (0…31) • Element Size to the Tx Buffer Start Address Ni_TXBC (i=0-3).TBSA.

Related information

Dedicated Tx buffers on page 3954

21.5.3.5.5 Mixed dedicated Tx buffers/Tx FIFO

In this case the Tx Buffers section in the Message RAM is subdivided into a set of dedicated Tx Buffers and a Tx FIFO. The number of dedicated Tx Buffers is configured by Ni_TXBC (i=0-3).NDTB. The number of Tx Buffers assigned to the Tx FIFO is configured by Ni_TXBC (i=0-3).TFQSi. In case Ni_TXBC (i=0-3).TFQS is programmed to 0x0, only dedicated Tx Buffers are used.

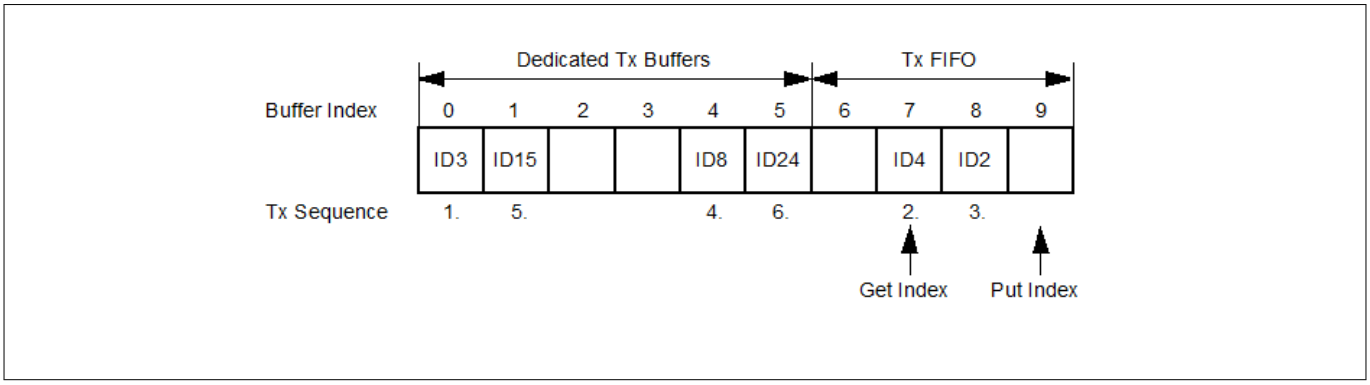


Figure 382 Example of mixed configuration dedicated Tx buffers/Tx FIFO

Tx prioritization:

- Scan dedicated Tx Buffers and oldest pending Tx FIFO Buffer (referenced by Ni_TXFQS (i=0-3).TFGI)
- Buffer with lowest Message ID gets highest priority and is transmitted next

21.5.3.5.6 Mixed dedicated Tx buffers/Tx queue

In this case the Tx Buffers section in the Message RAM is subdivided into a set of dedicated Tx Buffers and a Tx Queue. The number of dedicated Tx Buffers is configured by Ni_TXBC (i=0-3).NDTB. The number of Tx Queue Buffers is configured by Ni_TXBC (i=0-3).TFQSi. In case Ni_TXBC (i=0-3).TFQS is programmed to zero, only dedicated Tx Buffers are used.

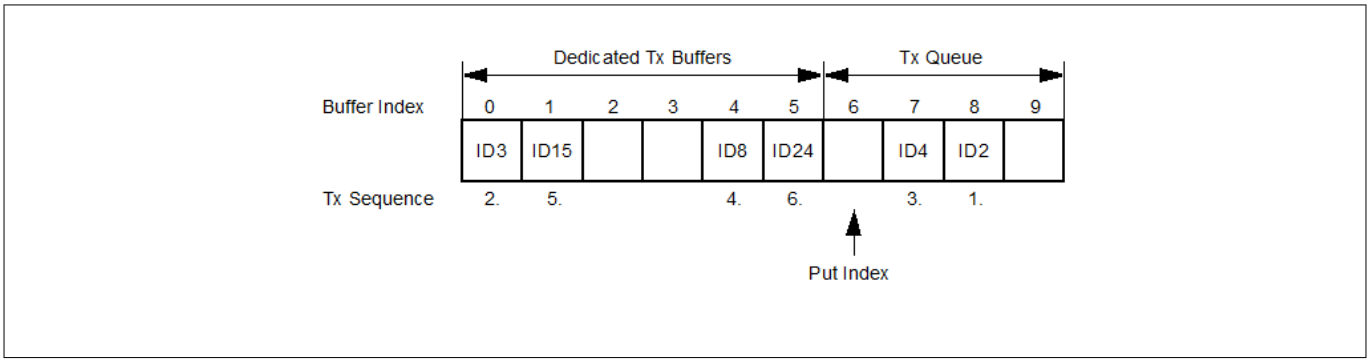


Figure 383 Example of mixed configuration dedicated Tx buffers/Tx queue

Tx prioritization:

- Scan all Tx Buffers with activated transmission request
- Tx Buffer with lowest Message ID gets highest priority and is transmitted next

21.5.3.5.7 Transmit cancellation

This feature is especially intended for gateway applications and AUTOSAR based applications. To cancel a requested transmission from a dedicated Tx Buffer or a Tx Queue buffer the Host has to write a '1' to the corresponding bit position (=number of Tx Buffer) of register Ni_TXBCR (i=0-3). Transmit cancellation is not intended for Tx FIFO operation.

Successful cancellation is signalled by setting the corresponding bit of register TXBCF to '1'.

In case a transmit cancellation is requested while a transmission from a Tx Buffer is already ongoing, the corresponding TXBRP bit remains set as long as the transmission is in progress. If the transmission was successful, the corresponding TXBTO and TXBCF bits are set. If the transmission was not successful, it is not repeated and only the corresponding TXBCF bit is set.

***Note**: In case a pending transmission is cancelled immediately before this transmission could have been started, there follows a short time window where no transmission is started even if another message is also pending in this node. This may enable another node to transmit a message which may have a lower priority than the second message in this node.*

21.5.3.5.8 Tx event handling

To support Tx event handling the M_CAN has implemented a Tx Event FIFO. After the M_CAN has transmitted a message on the CAN bus, Message ID and timestamp are stored in a Tx Event FIFO element. To link a Tx event to a Tx Event FIFO element, the Message Marker from the transmitted Tx Buffer is copied into the Tx Event FIFO element.

The Tx Event FIFO can be configured to a maximum of 32 elements. The Tx Event FIFO element is described in related information.

The purpose of the Tx Event FIFO is to decouple handling transmit status information from transmit message handling that is a Tx Buffer holds only the message to be transmitted, while the transmit status is stored separately in the Tx Event FIFO. This has the advantage, especially when operating a dynamically managed transmit queue, that a Tx Buffer can be used for a new message immediately after successful transmission. There is no need to save transmit status information from a Tx Buffer before overwriting that Tx Buffer.

When a Tx Event FIFO full condition is signalled by Ni_IR (i=0-3).TEFF, no further elements are written to the Tx Event FIFO until at least one element has been read out and the Tx Event FIFO Get Index has been incremented. In case a Tx event occurs while the Tx Event FIFO is full, this event is discarded and interrupt flag Ni_IR (i=0-3).TEFL is set.

To avoid a Tx Event FIFO overflow, the Tx Event FIFO watermark can be used. When the Tx Event FIFO fill level reaches the Tx Event FIFO watermark configured by Ni_TXEFC (i=0-3).EFWM, interrupt flag Ni_IR.TEFW is set.

When reading from the Tx Event FIFO, two times the Tx Event FIFO Get Index Ni_TXEFS (i=0-3).EFGI has to be added to the Tx Event FIFO start address Ni_TXEFC (i=0-3).EFSA.

Related information

Tx event FIFO element on page 3931

21.5.3.6 FIFO acknowledge handling

The Get Indices of Rx FIFO 0, Rx FIFO 1, and the Tx Event FIFO are controlled by writing to the corresponding FIFO Acknowledge Index (see Ni_RXF0A, Ni_RXF1A and Ni_TXEFA). Writing to the FIFO Acknowledge Index will set the FIFO Get Index to the FIFO Acknowledge Index plus one and thereby updates the FIFO Fill Level. There are two use cases:

When only a single element has been read from the FIFO (the one being pointed to by the Get Index), this Get Index value is written to the FIFO Acknowledge Index.

When a sequence of elements has been read from the FIFO, it is sufficient to write the FIFO Acknowledge Index only once at the end of that read sequence (value: Index of the last element read), to update the FIFO's Get Index.

Due to the fact that the CPU has free access to the M_CAN's Message RAM, special care has to be taken when reading FIFO elements in an arbitrary order (Get Index not considered). This might be useful when reading a High Priority Message from one of the two Rx FIFOs. In this case the FIFO's Acknowledge Index should not be written because this would set the Get Index to a wrong position and also alters the FIFO's Fill Level. In this case some of the older FIFO elements would be lost.

***Note**: The application has to ensure that a valid value is written to the FIFO Acknowledge Index. The M_CAN does not check for erroneous values.*

21.6 CAN Routing Engine (CRE)

The CAN Routing Engine (CRE) is an extension of MCMCAN which is used to route CAN frames by determining the destination(s) of a received CAN frame based on the received CAN-ID.

The destination of the CAN frame can be another CAN interface or Ethernet interface or a System RAM location. The CRE standalone routes the CAN frames with destination within the MCMCAN it belongs to. For CAN frames with destination outside the MCMCAN, CRE writes the frame to respective Host Buffer and triggers the Data Routing Engine (DRE) or an interrupt to the Interrupt Router based on the configuration. The CRE provides a virtual host interface to the RxFIFOs and TxQueues/TxFIFO of M_CAN, for ease of reading the received CAN frame and writing the transmit CAN frame without any additional processing by host to perform address calculation/acknowledge for CAN RxFIFO and TxQueue/TxFIFO. The CRE also consists of a Intrusion Detection Measurement Unit (IDMU) which provides parameters that can be used by intrusion detection software.

The CRE is enabled by setting Ni_CRE_CONFIG.EN to 1. When CRE is enabled, the virtual host interfaces RX and TX Host Buffers are also enabled. In this case, the received frames are read from the receive Host Buffer RHBUF by the FPI master instead of the RxFIFO. Frames are written to the Transmit Host Buffer THBUF by the FPI master instead of TxQueue/TxFIFO. The CRE must be enabled before enabling routing or IDMU.

***Note**: When CRE is disabled, both routing and IDMU are disabled. The host buffers are disabled and the frames are written to or read from TxQueue/TxFIFO and RxFIFO.*

The routing operation is enabled by setting both Ni_CRE_CONFIG.EN and Ni_CRE_CONFIG.REN to 1. This CRE should be enabled both at the source and the destination nodes for a successful routing operation.

The IDMU is enabled by setting both Ni_CRE_CONFIG.EN and Ni_CRE_CONFIG.IDMUEN to 1.

21.6.1 Feature list

The following are the features of CRE:

- Offloads CPU by hardware accelerated routing of received CAN frames to a user configured destination(s)
- Hardware accelerated CAN to CAN routing within the same MCMCAN module
- Assists Data Routing Engine (DRE) to perform hardware accelerated
  - CAN to CAN routing between different MCMCAN module
  - CAN to Ethernet (in IEEE 1722 ACF Frame) routing
  - CAN frame to a user configurable system RAM location transfer
  - CAN PDU to a user configurable system RAM location transfer
- Supports DRE in multi-cast routing, that is routing of a received CAN frame to a maximum of up to four destinations
- Provides a configurable routing table to define the rules of the routing operation

- Abstracts the M_CAN RxFIFOs and TxFIFO/TxQueues operation to the host interface by providing virtual buffers
  - Enables system DMA, CPU software or DRE to write transmitted CAN frames and to read received CAN frames with ease
- Consists of a measurement unit for Intrusion Detection (IDMU)

21.6.2 Functional overview

The CAN Routing Engine is shared by all four CAN interfaces in a MCMCAN module as shown in the figure below.

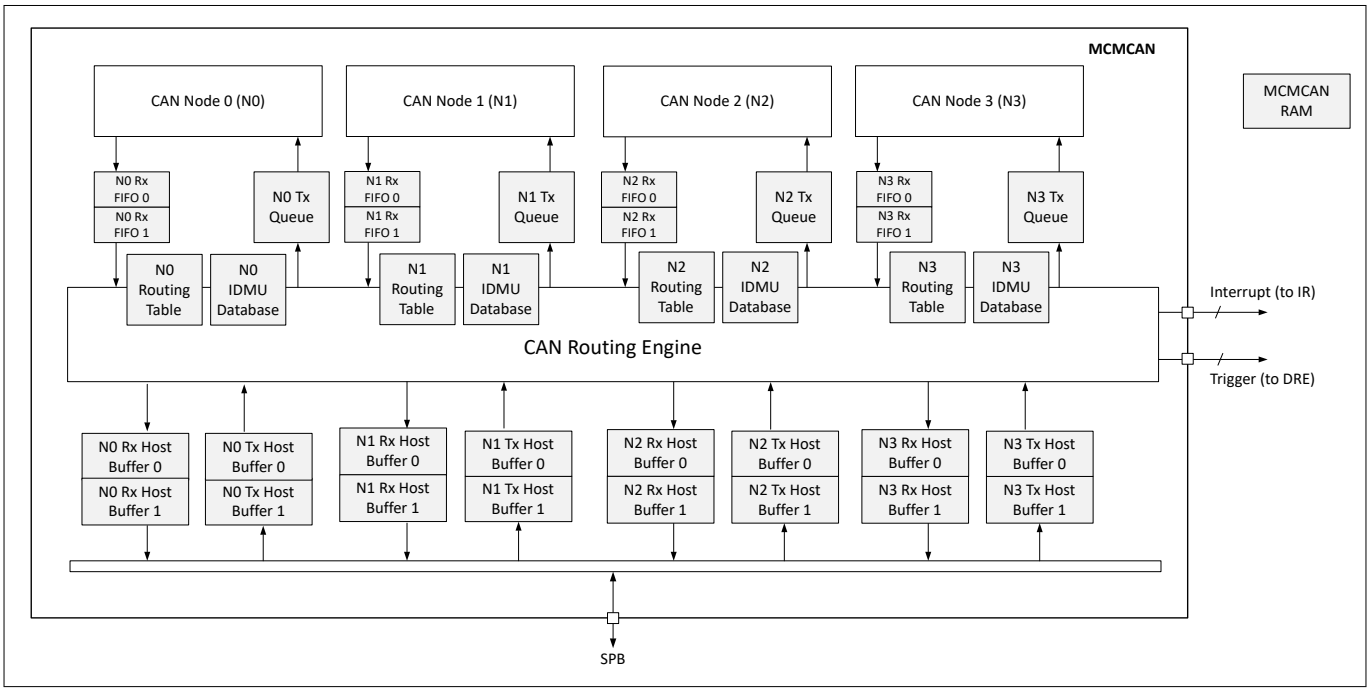


Figure 384 CRE interfaces to CAN nodes

Routing table

The Routing Table contains the rules which decide the routing path of the received CAN frame. It also has other configuration parameters related to its destination. The CAN Routing Engine uses the Routing Table to perform all the routing operations.

IDMU

The Intrusion Detection Measurement Unit (IDMU) contains a Timestamp database and a Frame Rate Measure table, which provide measures that can be used for intrusion detection. The CAN Routing Engine uses the Timestamp database to calculate the Inter Arrival Measure (IAM) of frames. The Frame Rate Measure table consists of counters that increment with reception of each frame.

Receive host buffer

The Receive Host Buffer 0/1 contains the pending received CAN frame from RxFIFO0/RxFIFO1. The CRE identifies the routing destination of the newly received CAN frame from RxFIFO0 or RxFIFO1. If the Routing Destination is not one of the CAN Nodes belonging to the CRE (within the same MCMCAN the CRE belongs to), then CRE writes the CAN frame to the respective Receive Host Buffer. If the destination of the CAN frame is within the same MCMCAN the CRE belongs to, the frame is not written to respective Receive Host Buffer. The CRE transmit interface writes the CAN frame directly to the destination CAN node TxQueue or TxFIFO.

Event Notification: Each new frame on the Rx Host Buffer triggers either an interrupt to IR or a trigger to DRE, as configured by the user.

Transmit host buffer

The Tx Host Buffer0/1 contains the CAN frame to be transmitted. Upon detecting a valid CAN frame, the CRE copies the frame to TxQueue or to TxFIFO and initiates transmission by setting Transmit request (Ni_TXBAR).

Event Notification: When a free element in TxQueue/TxFIFO is available, triggers either an interrupt to IR or a trigger to DRE, indicating the possibility to initiate transmission of a new CAN frame through the Tx Host Buffer.

21.6.3 Functional description

The block diagram of CRE is shown in Figure below

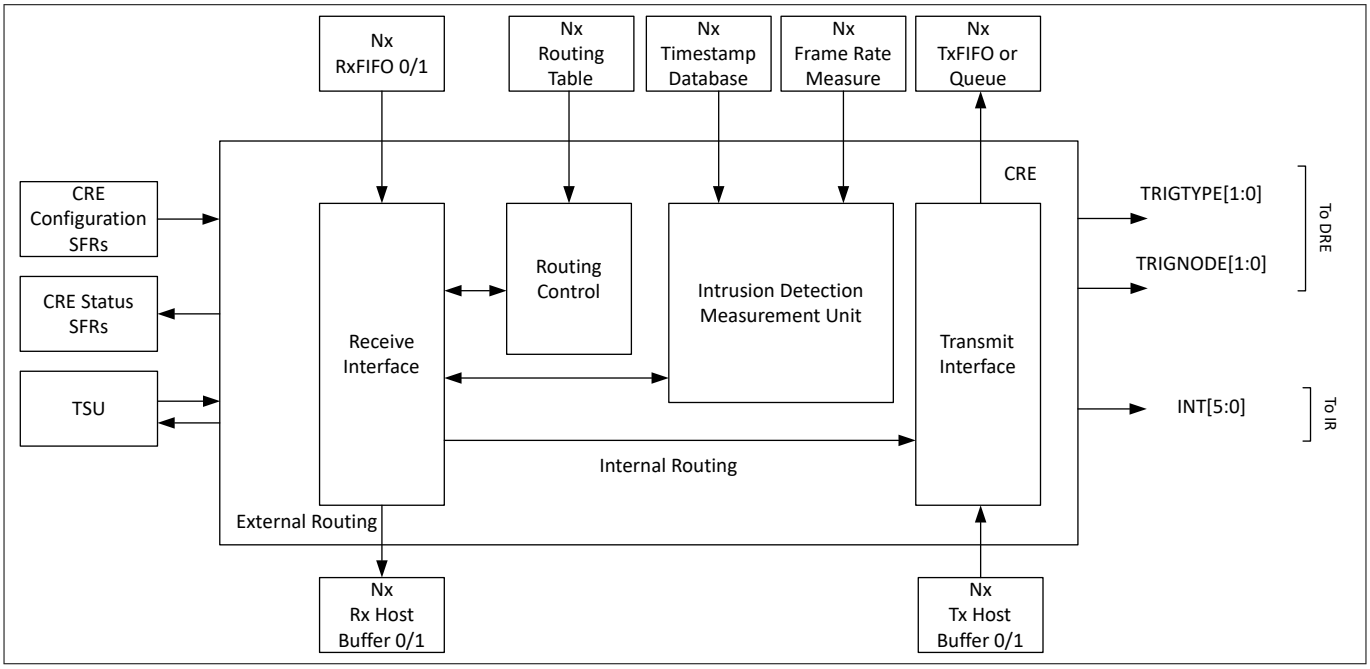


Figure 385 CRE block diagram

The CRE consists of four major blocks:

- Receive Interface
- Routing Control
- Intrusion Detection Measurement Unit (IDMU)
- Transmit Interface

In addition to it, the CRE Host Buffers, Routing Tables, IDMU Tables and databases are stored in the MCMCAN RAM. Total RAM size of 36KBytes or 20KBytes (Refer to Device specific CAN configuration chapter for the total RAM size for the CAN module) is shared between CRE and CAN. Based on the use case, user shall do the RAM configuration for CRE and CAN. For example, 4KB for CRE and 32 KB of RAM can be configured for MCMCAN0. Ni_CRE_CONFIGADR.SA defines the start address of the CRE configurations in the RAM. The start addresses and size of the tables are part of the CRE tables specific configuration parameters as shown in the following figure.

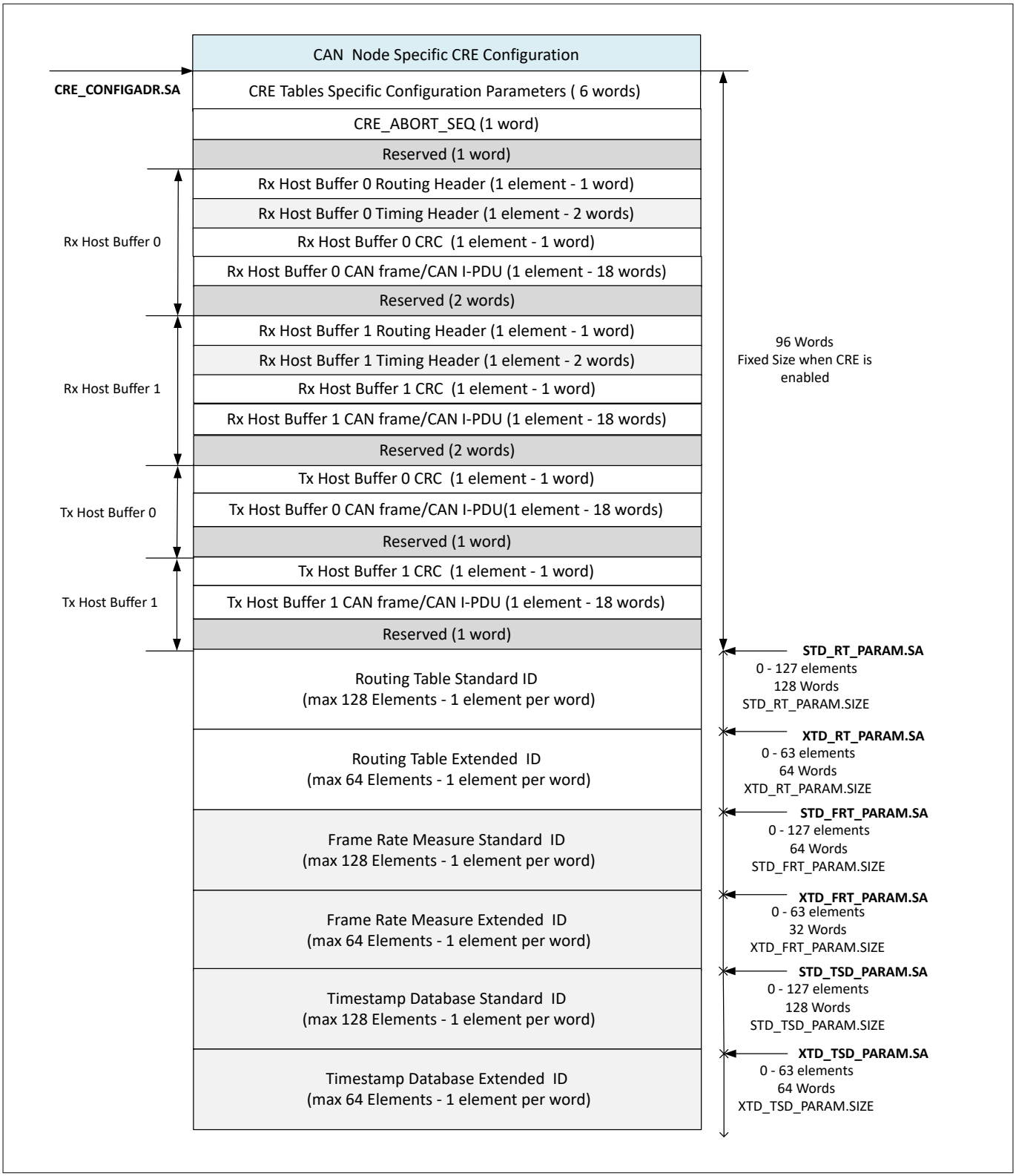


Figure 386 Message RAM layout for CRE

***Note**: Offset addresses for the CRE Standard Routing Table, CRE Extended Routing Table, CRE Standard Frame Rate Table, CRE Extended Frame Rate Table, CRE Standard Timestamp Database and CRE Extended Timestamp Database are relative to the configured start address by the user in the appropriate CRE configuration table. These configured start addresses are absolute in the CAN message RAM and not relative to the start of the CRE RAM section.*

In following sections, detailed functionality of CRE with respect to transmit and receive CAN frames are described.

21.6.3.1 Receive interface

The Receive Interface abstracts the following operations relating to RxFIFO for the user:

- Address calculation to identify the pending RxFIFO0/1 element
- Acknowledge the RxFIFO 0/1 after the read operation
- Fetching the external 32-bit timestamp from the TSU
- Fetching the internally generated 16-bit timestamp from RXMSGk_R1A (k=0-63).RXTS[15:0]

When Ni_CRE_CONFIG (i=0-3).EN = 1, a master of FPI can always read a pending received CAN frame only from the Rx Host Buffer 0/1 instead of accessing from the RxFIFO0/1. The CRE assists the read operation by providing a interrupt/trigger for each newly available CAN frame stored in Rx Host Buffer 0/1. The Receive Interface interacts with the Routing Control to identify the destination of a corresponding received CAN frame. It fetches the receive timestamp from the Timestamping Unit (TSU) (external timestamp) or from RXMSGk_R1A (k=0-63).RXTS[15:0] (internal timestamp). It interacts with the IDMU to calculate the intrusion detection measures. The Receive Interface also updates the timing header with the received timestamp.

Identify a new received frame from RxFIFO

The Receive Interface monitors the RxFIFO fill level Ni_RXF0S (i=0-3).F0FL (or) Ni_RXF1S (i=0-3).F1FL. When the F0FL or F1FL is not 0, then CRE reads the newly received CAN frame as indicated by RxFIFO Get index Ni_RXF0S.F0GI or Ni_RXF1S.F1GI. The corresponding Get index of the RxFIFO is also indicated in Ni_CRE_HBUF_RXz_STAT (i=0-3;z=0-1).INDEX.

Fetching the receive timestamp

Every frame consists of a 16-bit internal (Internal Timestamp Generation, see related information) or 32-bit external timestamp (Timestamping Unit (TSU), see related information). Receive Interface first checks if the TSU is enabled (CCCR.UTSU = 1 and TSUE = 1) and then if RXMSGk_R1B (k=0-63).TSC is set. R1B.TSC is set when a timestamp has been captured by the TSU and RXMSGk_R1B (k=0-63).RXTSP holds a valid timestamp pointer. If R1B.TSC is set and the TSU is enabled, the Receive Interface fetches the 32-bit timestamp captured by the TSU using the RXMSGk_R1B (k=0-63).RXTSP which holds the pointer of the TSU's timestamp register.

The Timing Header (THEAD) is composed of two words (32-bit each) as shown in the figure above. RHBUFk_THEAD_RXTS (k=0-1) holds the timestamp and RHBUFk_THEAD_INTRD (k=0-1) consists of status flags (TSCLEN, TSC and TSL) associated with the THEAD_RXTS. The Receive Interface updates the THEAD with the timestamp received in the following manner:

- Case 1: When a 16-bit internal timestamp is used, Timing Header (THEAD) RHBUFk_THEAD_RXTS (k=0-1) is updated with RXMSGk_R1A (k=0-63).RXTS[15:0]. TSCLEN and TSC are set to 0 by the hardware
- Case 2: When a 32-bit external timestamp is used, Timing Header (THEAD) RHBUFk_THEAD_RXTS (k=0-1) is updated with the timestamp from TSU pointed to by RXMSGk_R1B (k=0-63).RXTSP[3:0]. TSCLEN and TSC are set to 1 by the hardware. The TSC bit is set to 1 by the hardware to indicate that the 32-bit timestamp RHBUFk_THEAD_RXTS (k=0-1) is valid. The timestamp is only valid when bits Ni_CCCR.UTSU, TSUE, SSYNC/ESYNC, RXMSGk_R1B.TSC and Ni_TSU_TSS1.TSN are set. TSL bit is set in the THEAD when there is a loss of timestamp in the TSU (Ni_TSU_TSS1.TSLn = 1)

***Note**: The Intrusion Detection Measurement Unit (IDMU) (see related information) updates the THEAD further with the intrusion detection measure IAM and IAMSTAT*

Relevance for routing operation

When the routing operation is enabled for a CAN node (Ni_CRE_CONFIG (i=0-3).REN), then the Receive Interface decides if the received CAN frame is relevant for routing operation or not, based on the Acceptance Filter from which the CAN frame has been received. REN should be enabled at the source CAN node. Enabling REN makes the CAN node a routing source. The frame is routed to the destination DID irrespective of the REN bit at the destination node.

The CRE Receive Interface checks the Acceptance Filter ID (RXMSGk_R1A (k=0-63).FIDX) and Accept Non-Matching Frame (RXMSGk_R1A (k=0-63).ANMF) in the Rx FIFO element of the received CAN frame.

- Case 1: When R1.ANMF = 1, then the Rx frame is not relevant for routing
- Case 2: When R1.ANMF = 0, and R1.FIDX < CRE_STD_RT_PARAM.SIZE (when R0.XTD = 0) or CRE_XTD_RT_PARAM.SIZE (when R0.XTD = 1), the Rx frame is relevant for routing. Otherwise, the Rx frame is not relevant for routing

In case routing is disabled (REN is 0), all the frames are considered as routing non-relevant. REN bit is checked only at the source CAN node of the Routing operation.

Case 1: Routing non-relevant Rx CAN frame

The Received CAN frame is processed according to Table 984 without performing a Routing operation. Ni_CRE_HBUF_RXz_STAT (i=0-3;z=0-1).VRH is set to 0, to indicate the corresponding CAN frame stored in Rx Host Buffer does not have a Routing Header (Routing non-relevant).

Case 2: Routing relevant Rx CAN frame

The CRE Receive Interface passes the CAN frame over to the Routing Control. Routing Control consists of Routing Tables for both standard ID and extended ID CAN frames. The Routing Tables should be configured by the user with routing rules (standard ID: SRTk_UCR or SRTk_MCR and extended ID: XRTk_UCR or XRTk_MCR) that give the destination ID (DID) of CAN frames. The routing destination ID (DID) map is shown in the following table. After Routing Control has identified the destination of the CAN frame, the Receive Interface performs an internal CAN routing or external CAN routing depending on the DID.

1. Internal routing of CAN frames: If the DID belongs to the IDs of the CAN node belonging to the same MCMCAN as that of the CRE, the CRE Receive Interface provides the CAN frame along to the Transmit Interface. For example, if the Ni_CRE_CONFIG (i=0-3).ID = 1_H and corresponding routing rule configured in the Routing Table SRTk_UCR (k=0-127).DID = 2_H, the frame is routed internally within MCMCAN0 from CAN node with ID 1_H to CAN node with ID 2_H

   ***Note**: No Routing header is generated and CAN frame is not written to the respective RHBUF (Rx Host Buffer) in case of internal routing.*

2. External routing of CAN frames: If the DID does not belong to the IDs of the CAN node belonging to the same MCMCAN as that of the CRE (Ni_CRE_CONFIG (i=0-3).ID), then CRE Receive Interface generates a Routing Header (RHBUFk_UCRH (k=0-1) or RHBUFk_MCRH (k=0-1)), writes the Routing Header and the modified CAN frame to the corresponding Rx Host Buffer. Table 985 explains the processing of external routing relevant frames. VRH is set to 1, to indicate the corresponding CAN frame stored in Rx Host Buffer has a valid Routing Header (Routing relevant). When the DID does not belong to any valid destination, the VRH is not set. The format of the Routing Header is described in RHBUFk_UCRH (k=0-1) or RHBUFk_MCRH (k=0-1)

The CRE Receive Interface also computes an 16-bit CRC over the CAN headers R0 and R1 (except ESI, ANMF, RXTS and FIDX which are considered zero for CRC calculation), the safety critical CAN payload and the DID of CAN frame stored in the Rx Host Buffer. The CRC is calculated over 16-bit data at a time starting with the MSB. The 32-bit RAM word is divided into two 16-bit words and the highest 16 bits (31:16) is taken first for the CRC calculation followed by the lower 16 bits (15:0). The CRC calculation and verification is always enabled when

Ni_CRE_HBUF_RXz_CONFIG (i=0-3;z=0-1).TRIGEN is enabled. The CRC calculation and verification can be enabled or disabled using Ni_CRE_HBUF_RXz_CONFIG (i=0-3;z=0-1).CRCEN. In case of DIDs belonging to system memory locations (20_H till $3B_H$), the CRC calculation is not valid and it is ignored by the DRE. The CRC should be read exactly once as the first element from the Rx Host Buffer by the DRE or by the application. The Receive Interface stores the computed CRC along with the CAN frame in the Rx Host Buffer. The CRE uses CCITT CRC16 polynomial: 0x1021 $x^{16} + x^{12} + x^5 + 1$.

Table 983 Routing destination IDs

Destination	ID (6 bits)	Transfer destination
Destination disabled	0_H	-
MCMCAN0_Ni (i=0:3)	1_H till 4_H	CAN0 Transmit Host Buffer 0 (MCMCAN0_Ni_THBUF0)
MCMCAN1_Ni (i=0:3)	5_H till 8_H	CAN1 Transmit Host Buffer 0 (MCMCAN1_Ni_THBUF0)
MCMCAN2_Ni (i=0:3)	9_H till C_H	CAN2 Transmit Host Buffer 0 (MCMCAN2_Ni_THBUF0)
MCMCAN3_Ni (i=0:3)	D_H till 10_H	CAN3 Transmit Host Buffer 0 (MCMCAN3_Ni_THBUF0)
MCMCAN4_Ni (i=0:3)	11_H till 14_H	CAN4 Transmit Host Buffer 0 (MCMCAN4_Ni_THBUF0)
ACF Ethernet frames	18_H till $1D_H$	DRE CAN Input Buffer List
System memory locations (1 to 28)	20_H till $3B_H$	User configured by DMEMi (i=0:27)
Reserved for future extension	Others	-

In case of a routing non-relevant frames as shown in Case 1 above or for CAN frames with invalid destination ID, for example, in case of PDU routing with DID outside the range of System memory locations 20_H till $3B_H$, the behavior would be as follows:

Table 984 Behavior in case of routing non-relevant frames and invalid DID

Ni_CRE_HBUF_RXz_CONFIG.TRIGEN	Ni_CRE_HBUF_RXz_CONFIG.INTEN	Behavior
0	0	Invalid configuration. The frame just remains in the Receive Host Buffer.
1	0	The frame is discarded from the routing operation and there is no trigger to the DRE
0	1	The frame is written to the Receive Host Buffer and interrupt is triggered
1	1	The frame is written to the Receive Host Buffer and interrupt is triggered. There is no trigger to the DRE

Table 985 Behavior in case of external routing relevant frames

Ni_CRE_HBUF_RXz_CONFIG.TRIGEN	Ni_CRE_HBUF_RXz_CONFIG.INTEN	Behavior
0	0	Invalid configuration. The frame just remains in the Rx Host Buffer.
1	0	The frame is written to Rx Host Buffer and there is a trigger to the DRE
0	1	The frame is written to the Rx Host Buffer and interrupt is triggered
1	1	The frame is written to Rx Host Buffer and there is a trigger to the DRE. There is an interrupt trigger as well. The application needs to ensure it does not read any message content that is covered by the sequence checker in order not to disrupt the reading of message by the DRE

Event generation

After the Receive Interface writes the complete frame to the Receive Host Buffer (RHBUF), it generates the new event by triggering either the DRE or an interrupt to the IR.

- Trigger to DRE is done through TRIGTYPE[1:0] and TRIGNODE[1:0] interfaces when Ni_CRE_HBUF_RXz_CONFIG (i=0-3;z=0-1).TRIGEN is set
- Interrupt to IR Ni_CRE_IR (i=0-3).RBUFzI is triggered when Ni_CRE_HBUF_RXz_CONFIG (i=0-3;z=0-1).INTEN is set

After the trigger, the Rx Host Buffer New Event (Ni_CRE_HBUF_RXz_STAT (i=0-3;z=0-1).RHREQ) flag is set to 1 by hardware indicating a frame is ready to be read from the RHBUF. The software can also trigger the DRE by setting Ni_CRE_HBUF_RXz_STAT (i=0-3;z=0-1).SWTRIG to 1 after writing a frame into the RHBUF. The hardware clears the SWTRIG flag after trigger to DRE.

Multi-cast routing

In case of 1 up to 4 multi-cast routing, the Receive Interface hands over the frame to the Transmit interface in case of an internal routing. In case of destinations outside the same MCMCAN, the Receive Interface writes the complete frame to the respective Receive Host Buffer and triggers the DRE. The multi-cast routing with up to 4 DIDs outside the same MCMCAN is broken down into 4 individual uni-cast routing by the CRE. In this case, the CRE overwrites the Routing header with a new DID and triggers the DRE. The CRE clears the Receive Host Buffer (RHBUF) contents only after the DRE forwards the frames to the corresponding DID specified in the Routing Headers. The multi-cast routing rule SRTk_MCR or XRTk_MCR should follow the below configuration rules:

- At least DID0 and DID1 should have valid destination IDs (1_H till 14_H or 20_H till $3B_H$)
- All valid DIDs are different
- Unused DIDs are set to zero
- No gaps allowed between valid DIDs. For example,
  - Valid configuration: DID0, DID1 and DID2 are valid
  - Invalid configuration: DID0, DID1 and DID3 are valid. DID2 is set to zero

PDU routing

PDU routing is handled together by CRE and DRE. In case of PDU routing, the CRE Receive Interface:

- Converts the CAN frame into an CAN I-PDU (Interaction layer Protocol Data Unit) by generating a PDU Header along with the Routing and Timing Headers. The PDU Header is of two 32-bit RAM words and consists of the PDU ID and Payload length.

  ***Note**:* *PDU Header has the same size of 2 32-bit RAM words as that of R0 and R1 of the CAN frame so that data location remains the same*

- Writes the CAN I-PDU to the Rx Host Buffer
- Triggers the DRE

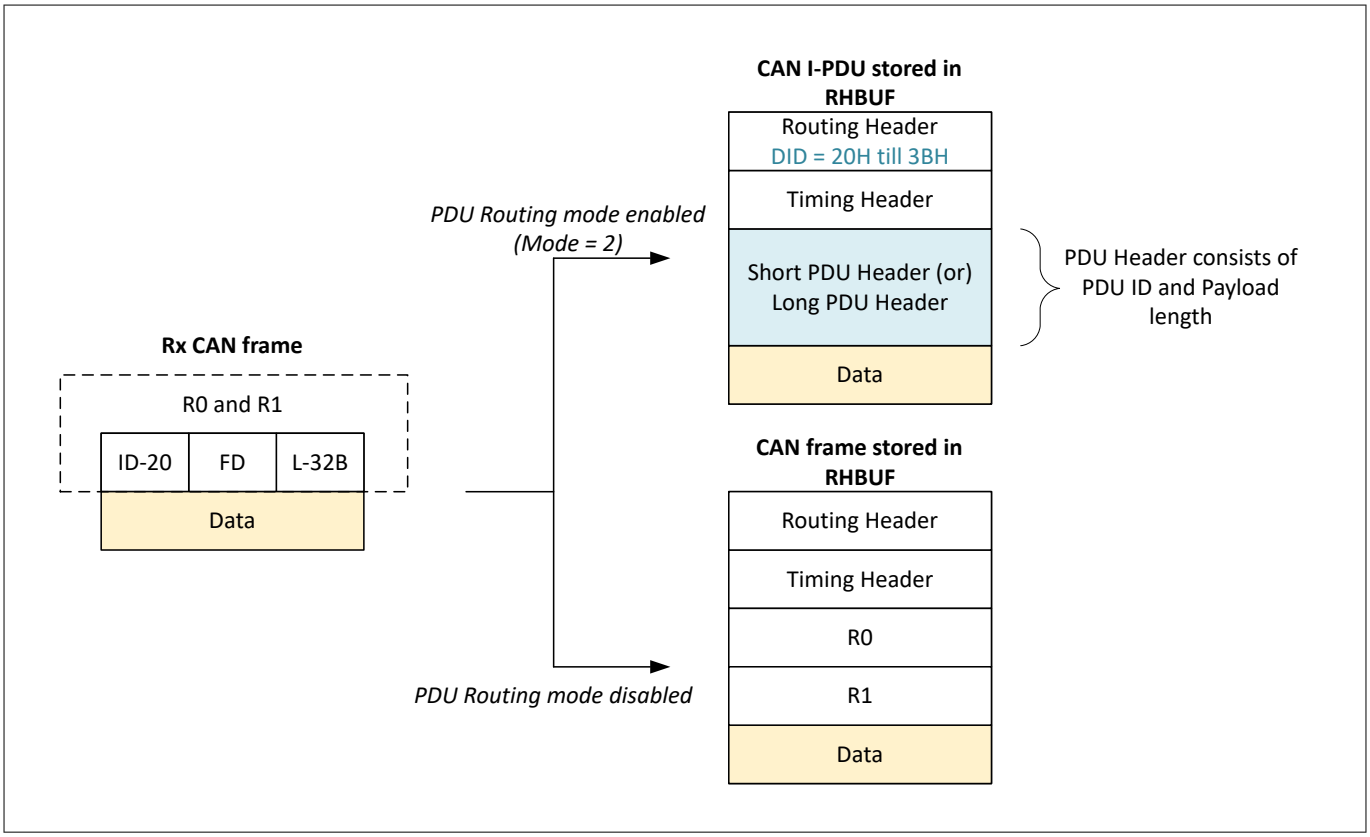


Figure 387 CAN frame to CAN I-PDU conversion

Read completion event

The Receive Host Buffer interrupt or trigger can be used by a master of FPI to read the CAN frame (R0, R1 and data) from the Receive Host Buffer. The Receive Interface also monitors the address of the Last Word of data to be read from FPI interface. The 'Last Word' can be configured either as Dynamic Data Length (based on DLC) or Fixed Data Length through Ni_CRE_HBUF_RXz_CONFIG (i=0-3;z=0-1).LEN. The DRE always reads the Rx Host Buffer in Dynamic Data Length mode. Fixed Data Length mode is only for software. One data word (DW) corresponds to 4 bytes.

For example, If the CAN Message has DLC = 4, and the Rx Host Buffer LEN = 2, then:

- - Dynamic Data Length
    - The last read operation is DW0 (RHBUFk_RHBUF_DBm (k=0-1;m=0-63))
  - Fixed Data Length
    - The last read operation is DW1 (RHBUFk_RHBUF_DBm (k=0-1;m=0-63))

***Note**:* *When the RTR bit is set, the frame ends at R1 because it has no data. Remote frames configured for PDU routing are rejected.*

On the 'Last Word' read completion event (the entire CAN frame is read from the Rx Host Buffer), the Receive Interface performs the following actions:

1. The Rx Host Buffer New Event (Ni_CRE_HBUF_RXz_STAT (i=0-3;z=0-1).RHREQ) flag is cleared by hardware
2. Acknowledge the corresponding RxFIFO by updating the Ni_RXF0A or Ni_RXF1A.FA with the index of the FIFO element currently processed
3. On each successful read completion of a frame from the Receive Host Buffer, the Receive Host Buffer counter (Ni_CRE_HBUF_RXz_STAT (i=0-3;z=0-1).COUNT) is incremented by 1 (resets to 0 on overflow)
4. The Receive Interface starts processing the next received CAN frame

   In case the DRE Move Engine has an error, it cancels the ongoing read sequence by writing 1 to CRE_ABORT_SEQ(i=0-3).CRHBUF0 or CRHBUF1 as long as the last read has not been started. A read sequence error is triggered in this case. Ni_CRE_IR(i=0-3).IRSI0 or IRSI1 error flag (depending on Rx Host Buffer) is set and interrupt is triggered when Ni_ERRCTRL.IRSIE is enabled. There is no re-trigger to the DRE. The SW clears the error flags and could read the frame from the respective Rx Host Buffer, or set the Ni_CRE_HBUF_RXz_STAT.SWTRIG in order to re-trigger the DRE, or write '1' into Ni_CRE_HBUF_RXz_STAT.RHREQ to free Rx Host Buffer for new frame and discard existing one from Rx Host Buffer.

Error monitoring

1. **Sequence check:** The CRE Receive Interface monitors the incorrect read access to the Rx Host buffer by an FPI master by checking the sequence in which the message is being read from the Rx Host Buffer (R0, R1 and DBm). The Routing header, Timing header and CRC are not part of the sequence check. The sequence checking starts once the FPI master starts reading the first word of the CAN frame or CAN I-PDU from the Rx Host buffer.

   If the message is read in an incorrect sequence, the Ni_CRE_IR(i=0-3).IRSI0 or IRSI1 (depending on Rx Host Buffer) error flag is set and interrupt is triggered when Ni_ERRCTRL.IRSIE is enabled. DRE is not re-triggered in this scenario. The SW clears the error flags and could read the frame from the respective Rx Host Buffer, or set the Ni_CRE_HBUF_RXz_STAT.SWTRIG in order to re-trigger the DRE, or write '1' into Ni_CRE_HBUF_RXz_STAT.RHREQ to free Rx Host Buffer for new frame and discard existing one from Rx Host Buffer.
2. **Timeout monitoring:**

   The CRE Receive Interface also monitors for a delayed read operation by master of FPI. The watchdog timer counter starts incrementing when timeout enable WDT.EN is set. The watchdog timer triggers periodic events E1, E2, …, En after a user configured timeout prescaler WDT.FWDP and SWDP. The FWDP pre-scaler is configured by user for monitoring fast events as shown in Table below. The SWDP prescaler is configured by user for monitoring slow events as shown in Table below. If FWDP or SWDP are set to zero, then a default value of 16 is used for the corresponding prescaler. The fast timeout prescaler and slow timeout prescaler is configured by the user based on the usage of both the RHBUF and THBUF. If the presence of start condition is detected at Ei, then the end condition is expected to happen before Ei+1. If not, the corresponding watchdog timeout error flag Ni_CRE_IR(i=0-3).RWDTI0 or RWDTI1 is set and interrupt is triggered whenNi_ERRCTRL(i=0-3).RWD0IE or RWD1IE is enabled. DRE is not re-triggered in this scenario. The SW clears the error flags and could read the frame from the respective Rx Host Buffer, or set the Ni_CRE_HBUF_RXz_STAT.SWTRIG in order to re-trigger the DRE, or write '1' into Ni_CRE_HBUF_RXz_STAT.RHREQ to free Rx Host Buffer for new frame and discard existing one from Rx Host Buffer.

   The watchdog counter is running at the host clock frequency f_{MCANH} . The timeout period between the events Ei is calculated as follows:

   - Default timeout period : $T_{def} = \frac{16}{f_{MCANH}}$

- Fast timeout period : $T_{fast} = T_{def} * (FWDP + 1)$
- Slow timeout period : $T_{slow} = T_{def} * (SWDP + 1)$

Table 986 Timeout interrupt trigger

Timeout check	Timeout interrupt trigger
End condition is true within a single event Ei	No timeout error. No timeout interrupt triggered
End condition is true within two consecutive events Ei	If the presence of start condition is detected at Ei, then the end condition is expected to happen before Ei+1. If not, timeout interrupt is triggered
End condition is true after two consecutive events Ei	Timeout interrupt is triggered

Table 987 RHBUF : Start and end condition list

Start condition (SC)	End condition (EC)	Direction	Event type	Enable
RxFIFO not empty	RxFIFO processing done	Source	Fast event	When Ni_ERRCTRL.WDG1 is enabled
RHBUF empty	RHBUF full	Source	Slow event	When Ni_ERRCTRL.WDG2 is enabled
RHBUF full	RHBUF empty	Source	Slow event	When Ni_ERRCTRL.WDG3 is enabled

***Note**: Timeout monitoring does not apply to routing non-relevant frames*

Related information

Timestamping unit (TSU) on page 3944
Internal timestamp generation on page 3944
Intrusion detection measurement unit (IDMU) on page 3978
Transmit Interface on page 3973

21.6.3.2 Routing control

The Routing Control along with the Routing Table decides the destination location of the received CAN frame and formats the CAN frame as defined by the Routing rule.

Routing table

The Routing Table contains a number of user configured Routing rules. The Routing rules define the destination of the CAN frame. The destination can be another CAN interface or Ethernet interface or a system memory location. It also provides the possibility to change the CAN Frame Format Classical/CAN FD Frame, Data Length and ID of the CAN frame.

Each CAN node can have two Routing Tables

- Standard ID Routing Table - Containing the routing rules for Rx CAN frames with standard 11-bit CAN ID
- Extended ID Routing Table - Containing the routing rules for Rx CAN frames with extended 29-bit CAN ID

The start address of the Routing Table is configured in CRE_STD_RT_PARAM.SA and number of Routing rules for standard CAN IDs is configured using CRE_STD_RT_PARAM.SIZE. Similarly for extended CAN IDs at CRE_XTD_RT_PARAM.SA and CRE_XTD_RT_PARAM.SIZE.

Each Routing rule, supports three modes of routing:

- Uni-cast Routing rule (Mode = 0)
  - Only one configurable destination
  - CAN frame modification is possible
- Multi-cast Routing rule (Mode = 1)
  - Up to 4 configurable destinations
  - CAN frame modification not possible
- PDU Routing rule (Mode = 2)
  - Only one configurable destination
  - PDU header generation and automatic PDU ID creation is based on the PDU header type configuration
  - Optional software configurable "Metadata"

The format of the standard ID Uni-cast and Multi-cast Routing rules are given in SRT_UCR and SRT_MCR respectively.

The format of the extended ID Uni-cast and Multi-cast Routing rules are given in XRT_UCR and XRT_MCR respectively.

CAN frame modifiers

When Uni-cast Routing mode is used (Mode = 0), then the CRE can modify the following properties of the received CAN frame based on the following configuration parameters of the Uni-cast Routing rule.

1. FDF (CAN FD Format)
   a. If SRT_UCR.FDFM = 0, the FDF bit is ignored and the CAN frame format is not changed. The frame format is kept same as that of the received CAN frame
   b. If SRT_UCR.FDFM = 1, the CAN frame format is changed to either Classical CAN frame (FDF = 0) or CAN FD frame format (FDF = 1)

   ***Note**: When a CAN-FD frame of more than 8 bytes is configured to be changed to classic CAN format, only the first 8 bytes will be transmitted and remaining data is truncated*

2. DLC (Data Length Code)
   a. If SRT_UCR or XRT_UCR.DLC = 0, the Destination CAN frame length is same as the source CAN frame
   b. If SRT_UCR or XRT_UCR.DLC != 0, the Destination CAN frame length is as defined by ISO 11898-1. The DLC configured in the routing rule must be smaller than the actual frame DLC. When DLC configured in the Routing rule is larger than the frame DLC, the additional bytes will contain undefined values in the Rx Host Buffer

3. IDXOR (CAN ID Modifier)
   a. For Standard ID, the Destination CAN Frame ID = SRT_UCR.IDXOR (XORed) Rx_CANID
   b. For Extended ID, the Destination CAN frame ID = [XRT_UCR.IDXOR << XRT_UCR.IDSHIFT] (XORed) Rx_CANID

***Note**: Both the source and destination nodes in the system should be aware of the frame modifications so that any potential truncation or modification of the safety critical payload can be avoided or properly handled by the system integrator.*

Routing table index

The indexing of the Routing Table is done based upon the receive acceptance filter ID. The Acceptance Filter ID is stored along with the CAN frame in the Receive Buffer, RXMSGk_R1A or RXMSGk_R1B.FIDX. The Rx CAN frames which are accepted by Acceptance Filter ID0 has the corresponding Routing rule stored at address (in words)

Offset ’0’ from the start of the Routing Table, similarly Acceptance Filter ID1 corresponds to address (in words) Offset '1' of Routing Table and so on.

Desired Routing Index:

Standard ID Routing rule Address = CRE_STD_RT_PARAM.SA + $(R1.\mathrm{FIDX} \times 4)$

Extended ID Routing rule Address = CRE_XTD_RT_PARAM.SA + $(R1.\mathrm{FIDX} \times 4)$

Hence it is essential to consider during the Acceptance Filter (AF) configuration, that AF configuration corresponding to the CAN Frames which are to be routed should start with the first AF element 0. A total of up to 128 standard ID Routing rules and 64 extended ID Routing rules can be configured.

PDU routing rule

The PDU routing mode is enabled, when SRTk_PR.Mode is configured as 0x2. The user shall configure the Routing rule "DID" to one of the System memory locations (20_H till $3B_H$). The Routing rule also consists of the "PDU header type", which is used to configure the PDU header contents. Short PDU Header is of 1 RAM word and Long PDU Header is of 2 RAM words.

- Type 1: 11-bit Standard CAN ID with no Metadata
- Type 2: 11-bit Standard CAN ID with Metadata
- Type 3: 11-bit Standard CAN ID with optional Metadata
- Type 4: 29-bit Extended CAN ID with no Metadata
- Type 5: 29-bit Extended CAN ID with no Metadata and no Source ID

***Note**:* *Type 5: If the CAN ID is unique per CAN node, the system can identify the CAN node without the Source ID. Last 3 bits are reserved*

Type 4: Only first 27 bits of the 29-bit ID are used

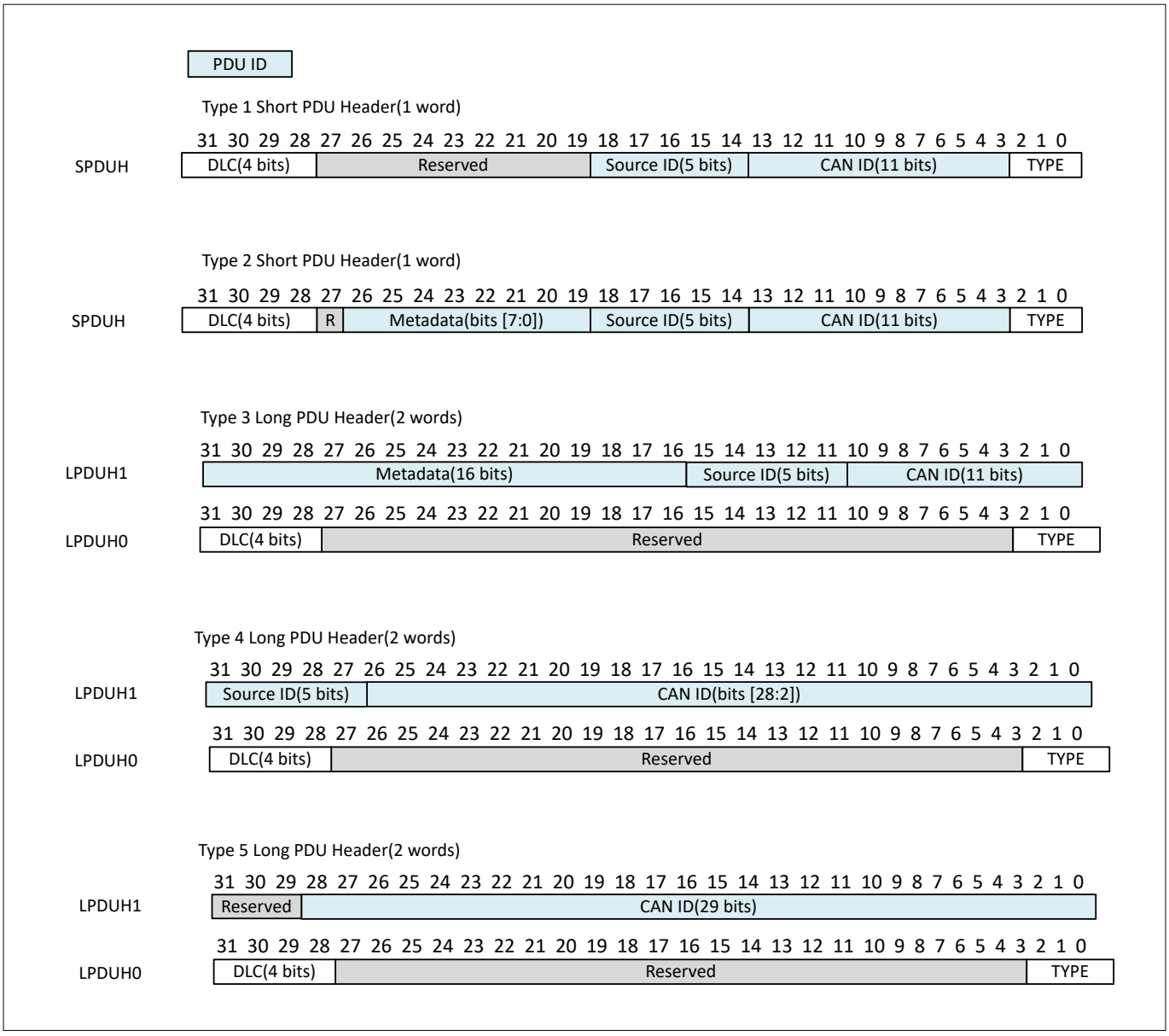


Figure 388 PDU Header types

21.6.3.3 Transmit Interface

The Transmit Interface abstracts the following operation related to TxFIFO or TxQueue for the user

- Address calculation to identify the available TxFIFO (or Tx Queue) element
- Setting a new transmit request for the corresponding TxFIFO (or Tx Queue) element

The Transmit Interface can get two types of transmit requests

- External transmit requests - a master of FPI initiates transmission of a new CAN frame.

  A master of FPI can write a CAN frame to be transmitted to the CAN Tx Host Buffer (THBUF) instead of the TxFIFO(Queue). The CRE assists the writing by providing a interrupt/trigger for each free TxFIFO(Queue) element
- Internal transmit requests - the CRE Receive Interface initiates transmission of a CAN frame due to internal CAN routing operation.

Transmit Interface can also get a new CAN frame from the CRE Receive Interface, if the destination of the CAN frame is configured by the Routing Table as another CAN Node belonging to the same MCMCAN module. The Transmit Interface identifies the destination CAN Node, with the help of DID in Routing Header

External transmit request

The CRE Transmit Interface first identifies a free element in the TxFIFO or Tx Queue based on the Put Index.

Identify free Tx buffer element in TxFIFO or Tx Queue

If the TxFIFO/TxQueue is not full, that is, Ni_TXFQS (i=0-3).TFQF = 0, then Tx FIFO Put Index Ni_TXFQS (i=0-3).TFQPI, points to the next Free Transmit Buffer in the Tx FIFO/TxQueue. The Ni_CRE_HBUF_TXz_STAT (i=0-3;z=0-1).INDEX points to the identified Put Index of TxFIFO/Tx Queue. Upon identifying a free transmit buffer element,

- Checks for a pending frame in the Tx Host Buffer (Ni_CRE_HBUF_TXz_STAT (i=0-3;z=0-1).THREQ is set to 1_B)
- In case of a pending frame in the Tx Host Buffer, the Transmit Interface
  - Copies the CAN frame from the Tx Host Buffer to the identified Tx Buffer element in TxFIFO or Tx Queue
  - Set transmit request (Ni_TXBAR (i=0-3).AR) of the corresponding Tx buffer element
  - On each successful write completion of a new transmit frame to the TxFIFO or Tx Queue, the Ni_CRE_HBUF_TXz_STAT (i=0-3;z=0-1).COUNT is incremented by 1_B(resets to 0 upon overflow)
  - Sets Ni_CRE_HBUF_TXz_STAT (i=0-3;z=0-1).THREQ to 0_B. Then it starts identifying the next available Tx Buffer element in the TxFIFO or TxQueue
- In case a Tx Host Buffer is empty, the Transmit Interface triggers either
  - Trigger to DRE: TRIGTYPE[1:0], TRIGNODE[1:0] or
  - Interrupt to IR: Ni_CRE_IR.TBUF0I or TBUF1I

  correspondingly based on the configuration of Ni_CRE_HBUF_TXz_CONFIG (i=0-3;z=0-1).INTEN.

***Note**: Only Tx Host Buffer 0 shall be configured to trigger to DRE.*

At the destination CAN node, Ni_CRE_CONFIG (i=0-3).DEN is checked by the CRE before the CAN frame is written to the TxFIFO.

Table 988 Routing behavior based on DEN configuration

Use-case	Ni_CRE_CONFIG.EN	Ni_CRE_CONFIG.DEN	Behavior
Internal	0	0	No routing enabled. Frame is discarded
Internal	0	1	Frame is routed to the destination TxFIFO
Internal	1	0	Frame is discarded
Internal	1	1	Frame is routed to the destination TxFIFO
External	0	0	No routing enabled. No triggers
External	0	1	No routing enabled. No triggers
External	1	0	Frame remains in the Tx Host buffer. No re-trigger is performed.
External	1	1	Frame is routed to the destination TxFIFO

***Note**: DEN=0 is used to isolate a CAN node in case of any bus error where the CAN frame is not expected to be written to the TxFIFO by the CRE*

The CRE should be enabled and the Host Buffers need to be configured after the DRE is enabled. A free Tx Host Buffer is indicated by Ni_CRE_HBUF_TXz_STAT (i=0-3;z=0-1).THREQ = 0_B. The trigger or the interrupt is used by master of FPI to write a new Transmit CAN frame to the Tx Host Buffer.

Tx Host Buffer 1 shall be configured to trigger interrupt to IR. Based on the interrupt, Tx Host Buffer 1 can be dedicated for System DMA. One System DMA channel is configured per CAN node. The interrupt Ni_CRE_IR(i=0-3).TBUF0/1I indicates the System DMA that the THBUF is ready to receive a new CAN frame. The software then writes the CAN frame to the system memory in dedicated buffers per CAN node. The System DMA reads each CAN frame from the system memory location and writes the CAN frame to the corresponding THBUF. The number of CAN frames transmitted is indicated in the Ni_CRE_HBUF_TXz_STAT (i=0-3;z=0-1).COUNT bit-field.

The Transmit Interface identifies the start of Transmit frame (T0, T1 and DBm) write by a FPI master by monitoring the write to the First Word of the Tx Host Buffer. The format of the Transmit Host Buffer is given in THBUFk_T0, THBUFk_T1 and THBUFk_DBm (m=0-63).

Write completion event

The Transmit Interface monitors the address of the "Last Word" of the Tx Host Buffer to be written by master of FPI interface. The "Last Word" can be configured either as Dynamic Data Length (based on DLC) or Fixed Data Length (Ni_CRE_HBUF_TXz_CONFIG.LEN). The configuration of last word can be done by programming Ni_CRE_HBUF_TXz_CONFIG (i=0-3;z=0-1).LWM. The DRE always writes into the Tx Host Buffer 0 in Dynamic Data Length mode. Fixed Data Length mode is only for the software. One data word (DW) corresponds to 4 bytes.

For example: If the Tx CAN frame has DLC = 4 and the Tx Host Buffer LEN = 2, then

- Dynamic Data Length: The last write operation is "DW0" (THBUFk_DBm (m=0-63))
- Fixed Data Length (8 bytes): The last write operation is "DW1" (THBUFk_DBm (m=0-63))

Upon the "the Last Word" write completion event (the entire CAN frame is written to the Tx Host Buffer), the Transmit Interface performs the following actions

1. Sets Ni_CRE_HBUF_TXz_STAT (i=0-3;z=0-1).THREQ to 1_B indicating Tx Host Buffer full
2. Copies the CAN frame from the Tx Host Buffer to the identified Tx Buffer element in TxFIFO or Tx Queue
3. Set transmit request (Ni_TXBAR (i=0-3).AR) of the corresponding Tx buffer element
4. On each successful write completion of a new transmit frame to the TxFIFO or Tx Queue, the Ni_CRE_HBUF_TXz_STAT (i=0-3;z=0-1).COUNT is incremented by 1_B(resets to 0 upon overflow)
5. The Transmit Interface sets Ni_CRE_HBUF_TXz_STAT (i=0-3;z=0-1).THREQ to 0_B. Then it starts identifying the next available Tx Buffer element in the TxFIFO or TxQueue

   In case the DRE Move Engine has an error, it cancels the ongoing write sequence by writing 1 to CRE_ABORT_SEQ(i=0-3).CTHBUF as long as the last write has not been launched. A write sequence error is launched in this case. Ni_CRE_IR.IWSI0 error flag is set and interrupt is triggered when Ni_ERRCTRL.IWSIE is enabled. There is no re-trigger to the DRE. The SW clears the error flags and could read the frame from the respective Tx Host Buffer, or set the Ni_CRE_HBUF_TXz_STAT.SWTRIG in order to re-trigger the DRE, or write '1' into Ni_CRE_HBUF_TXz_STAT.THREQ to free Tx Host Buffer for new frame and discard existing one from Tx Host Buffer.

Internal transmit request

The Transmit Interface gets the CAN frame along with a Routing Header from the Receive Interface in case of internal MCMCAN routing. The Transmit Interface identifies the destination CAN Node from the DID in the Routing Header. The Transmit Interface identifies a free Tx FIFO/Queue Element from the corresponding CAN node. It writes the CAN frame to the identified Tx FIFO/Queue element and sets the corresponding Transmit Request (Ni_TXBAR.AR). The Transmit Interface then acknowledges back the Receive Interface to indicate that the internal routing is performed. After the acknowledge, the Receive Interface performs the following operation

1. Acknowledge the corresponding RxFIFO by updating the Ni_RXF0A or Ni_RXF1A.FA with the index of the FIFO element currently processed
2. Clear RxFIFO New Message Flag Ni_IR (i=0-3).RF0N or RF1N

The Transmit Interface after setting the Transmit Request (Ni_TXBAR.AR), it starts identifying the next available Tx Buffer element in the TxFIFO or TxQueue.

Error monitoring

1. **Sequence check:** The CRE Transmit Interface monitors the incorrect write access to the Tx Host Buffer by an FPI master by checking the sequence in which the message is being written to the Tx Host Buffer(T0, T1 and DBm). The sequence checking starts once the FPI master starts writing the first word of the CAN frame to the Tx Host buffer.

   If the message is later written in an incorrect sequence, the Ni_CRE_IR.IWSI0 or IWSI1 (depending on the Tx Host Buffer) error flag is set and interrupt is triggered when Ni_ERRCTRL.IWSIE is enabled. There is no re-trigger of the DRE. The SW clears the error flags and could read the frame from the respective Tx Host Buffer, or set the Ni_CRE_HBUF_TXz_STAT.SWTRIG in order to re-trigger the DRE, or write '1' into Ni_CRE_HBUF_TXz_STAT.THREQ to free Tx Host Buffer for new frame and discard existing one from Tx Host Buffer.
2. **CRC verification:** The DRE or application generates an 16-bit CRC over the CAN headers R0 and R1 (except ESI, ANMF, RXTS and FIDX), the safety critical CAN payload and the DID using CCITT CRC16 polynomial: 0x1021 $x^{16} + x^{12} + x^5 + 1$. The CRC is calculated over 16 bits of data at a time starting with the MSB. The DRE or the application should write the CRC along with the CAN frame into the Tx Host Buffer. After the CAN frame is written to the Tx Host Buffer, CRE Transmit Interface verifies the CRC and the DID before writing the CAN frame to the Tx FIFO/Tx Queue. If there is a mismatch between the calculated CRC and the received CRC, the Ni_CRE_IR.CRCI0 or CRCI1 (depending on the Tx Host Buffer) interrupt is triggered when the CRC error interrupt is enabled in Ni_ERRCTRL.CRCIE. The processing of the frame is based on the Ni_CRE_HBUF_TXz_CONFIG.CRCG. In case the CRCG is set to 0, the CAN frame is processed further and written to the TxFIFO. In case the CRCG is set to 1, the CAN frame is discarded in the Host buffer. There is no re-trigger of the DRE. The SW clears the error flags and could read the frame from the respective Tx Host Buffer, or set the Ni_CRE_HBUF_TXz_STAT.SWTRIG in order to re-trigger the DRE. The user must ensure that the size of the TxFIFO or TxQueue is large enough to store the data from the Tx Host Buffer in order to avoid unwanted CRC error.
3. **Timeout monitoring:** The CRE Transmit Interface monitors for a delayed write operation by master of FPI. The watchdog timer counter starts incrementing when timeout enable WDT.EN is set. The watchdog timer triggers periodic events E1, E2, …, En after a user configured timeout prescaler WDT.FWDP or SWDP. The FWDP pre-scaler is configured by user for monitoring fast events as shown in Table below. The SWDP pre-scaler is configured by user for monitoring slow events as shown in Table below. If FWDP or SWDP are set to zero, then a default value of 16 is used for the corresponding prescaler. The fast timeout prescaler and slow timeout prescaler is configured by the user based on the usage of both the RHBUF and THBUF. If the presence of start condition is detected at Ei, then the end condition is expected to happen before Ei+1. If not, the corresponding watchdog timeout error flag Ni_CRE_IR.TWDTI0 or TWDT1 is set and interrupt triggered when Ni_ERRCTRL.TWD0IE or TWD1IE is enabled. There is no re-trigger of the DRE. The SW clears the error flags and could read the frame from the respective Tx Host Buffer, or set the Ni_CRE_HBUF_TXz_STAT.SWTRIG in order to re-trigger the DRE, or write '1' into Ni_CRE_HBUF_TXz_STAT.THREQ to free Tx Host Buffer for new frame and discard existing one from Tx Host Buffer.

The watchdog counter is running at the host clock frequency f_{MCANH} . The timeout period between the events Ei is calculated as follows:

- Default timeout period : $T_{def} = \frac{16}{f_{MCANH}}$
- Fast timeout period : $T_{fast} = T_{def} * (FWDP + 1)$
- Slow timeout period : $T_{slow} = T_{def} * (SWDP + 1)$

Table 989 Timeout interrupt trigger

Timeout check	Timeout interrupt trigger
End condition is true within a single event Ei	No timeout error. No timeout interrupt triggered
End condition is true within two consecutive events Ei	If the presence of start condition is detected at Ei, then the end condition is expected to happen before Ei+1. If not, timeout interrupt is triggered
End condition is true after two consecutive events Ei	Timeout interrupt is triggered

Table 990 THBUF : Start and end condition list

Start condition	End condition	Direction	Event type	Enable
THBUF empty	THBUF full	Destination	Slow event	When Ni_ERRCTRL.WDG2 is enabled
THBUF full	THBUF empty	Destination	Fast event	When Ni_ERRCTRL.WDG1 is enabled

21.6.3.4 Trigger function

The CRE can generate triggers to DRE for Rx Host Buffers and Tx Host Buffers. The trigger events are encoded as given in the following tables.

The signal TRIGTYPE[1:0] indicates a new trigger event corresponding to a Host Buffer. The signal TRIGNODE[1:0] indicates the CAN Node which the Host Buffer belongs to. Initially, the TRIGTYPE[1:0] and TRIGNODE[1:0] are 00b. The CRE, upon detecting the trigger event of a Host Buffer, sets the TRIGTYPE[1:0] and TRIGNODE[1:0] to corresponding level.

Table 991 TRIGTYPE description

TRIGTYPE(1)	TRIGTYPE(0)	Description
0_B	0_B	No Trigger or Idle
0_B	1_B	Transmit Host Buffer 0 free
1_B	0_B	New Receive Host Buffer 0
1_B	1_B	New Receive Host Buffer 1

Table 992 TRIGNODE description

TRIGNODE(1)	TRIGNODE(0)	Description
0_B	0_B	CANx_N0
0_B	1_B	CANx_N1
1_B	0_B	CANx_N2
1_B	1_B	CANx_N3

21.6.3.5 Intrusion detection measurement unit (IDMU)

The IDMU assists with intrusion detection by measuring the following timing parameters and updating the Timing Header (THEAD) for each received frame that has the routing destination outside the MCMCAN.

- Inter Arrival Measure (RHBUFk_THEAD_INTRD (k = 0-1).IAM)
- Frame Rate Measure (SFRk_FR (k = 0-63) or XFRk_FR (k = 0-31))
- Rx Throughput Measure (Ni_IDMU_RXTPCFG (i = 0-3).TP)

The IDMU consists of a Timestamp Database and Frame Rate Measure Table as shown in the following figure. All the IDMU Tables are part of the CRE RAM as shown in table CRE RAM Structure, see related information. The Timestamp Database stores the previous received timestamp (Reference Timestamp) of CAN frame(s) corresponding to a Acceptance Filter ID. The Frame Rate Measure Table consists of frame counters for each Acceptance Filter ID, that increment with reception of each frame accepted by the filter.

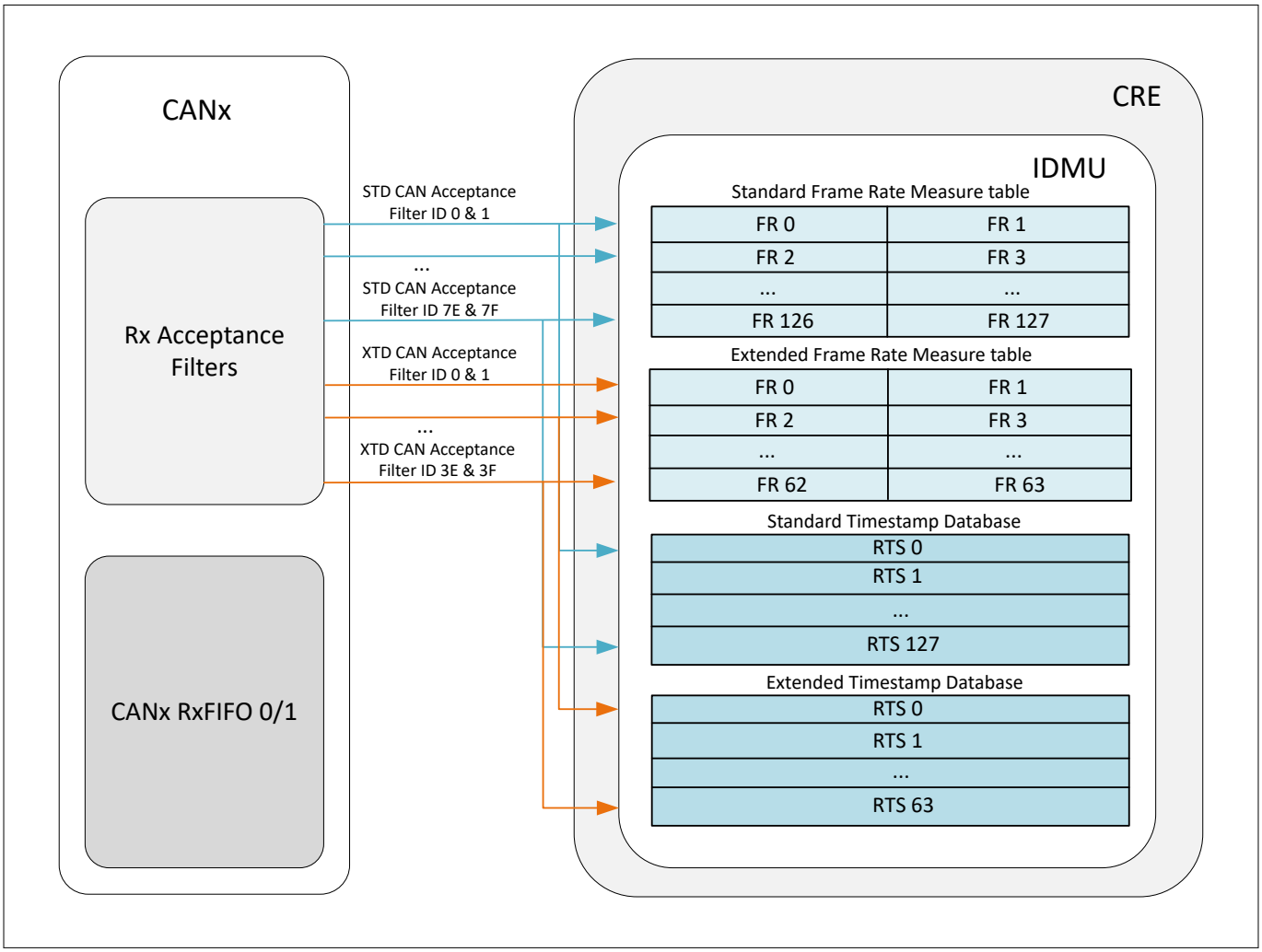


Figure 389 IDMU

Inter arrival measure

When there is valid timestamp available, the IAM is calculated for each frame by calculating the difference between the current received timestamp RHBUFk_THEAD_RXTS (k=0-1) and the Reference Timestamp STSDk_RTS (k=0-127) or XTSDk_RTS (k=0-63) which is fetched from the Reference Timestamp Database for the corresponding Acceptance Filter ID. The Reference Timestamp Database is updated with the last timestamp to compute the IAM of the next frame. Similar to the routing table, the indexing of the Reference Timestamp Database and Frame Rate Measure Table are done based upon the receive Acceptance Filter ID as shown in the figure above.

The IAM of each frame is part of the Timing Header THEAD.

For Standard ID CAN frame(s) :

$$RHBUF\text{k_THEAD_INTRD.IAM} = RHBUF\text{k_THEAD_RXTS.RXTS} - \text{STSDk_RTS.TS}$$

For Extended ID CAN frame(s) :

$$RHBUF\text{k_THEAD_INTRD.IAM} = RHBUF\text{k_ THEAD_RXTS.RXTS} - \text{XTSDk_RTS.TS}$$

When there is an overflow of the timebase counter and the current timestamp is smaller than the Reference Timestamp,

$$RHBUF\text{k_THEAD_INTRD.IAM} = (\text{MaxValue Of Timestamp} - \text{STSDk_RTS.TS}) + (RHBUF\text{k_THEAD_RXTS.RXTS} + 1)$$

The value of "MaxValueOfTimestamp" is computed based on the maximum value of TSCLEN.

The calculated IAM is written to the Timing Header RHBUFk_THEAD_INTRD (k=0-1).IAM and RHBUFk_THEAD_INTRD (k=0-1).IAMSTAT is set to 1.

If there was a loss of timestamp the RHBUFk_THEAD_INTRD (k=0-1).TSL is set. But the IAM is calculated with the latest available valid timestamp.

The Receive Interface fetches only valid timestamps. In case of no valid timestamp available or in case of the first frame received which will not have a Reference Timestamp to calculate the IAM, there is no IAM calculated and RHBUFk_THEAD_INTRD (k=0-1).IAMSTAT = 0.

Frame rate measure

The Frame Rate Measures SFRk_FR (k=0-63) are counters associated with each Acceptance Filter ID, which increment with reception of each frame. The start address of the Frame Rate Measure table and the size can be configured in CRE_STD_FRT_PARAM. Similarly for extended CAN IDs at CRE_XTD_FRT_PARAM.

Each entry of the Frame Rate Measure Table consists of two Frame Rate Measure counters FR1 and FR2, corresponding to two Acceptance Filter IDs (FID). The indexing of the Frame Rate Measures are based on the LSB of the FID. Each Frame Rate Measure counter is of 16 bits. The Frame Rate Measure counters are controlled by Ni_IDMU_FRTCONFIG (i=0-3).STDLOCK or XTDLOCK. Reading of the frame rate is recommended to be done in a timer-based interrupt service routine which sets the STDLOCK or XTDLOCK is '1' correspondingly . The Frame Rate Measures freeze whenever the STDLOCK/XTDLOCK is set to 1. The Frame Rate Measure counters and the STDLOCK/XTDLOCK bit must be reset to 0 by software after reading. The Frame Rate Measure counters start incrementing again with the reception of the next frame.

If frames are received while the STDLOCK/XTDLOCK bit is set, there is a loss of Frame Rate Measure count and an interrupt SFRMLI or XFRMLI is triggered when Ni_IDMU_FRTCONFIG.INTEN0 or Ni_IDMU_FRTCONFIG.INTEN1 is enabled.

Rx throughput measure

The Rx Throughput Measure Ni_IDMU_RXTPCFG (i=0-3).TP is a counter per CAN node. The Receive Interface identifies newly received CAN frames based on the RxFIFO fill level Ni_RXF0S (i=0-3).F0FL (or) Ni_RXF1S (i=0-3).F1FL. The Rx Throughput Measure counter TP increments whenever there is a new message. Reading of the Rx Throughput Measure counter is recommended to be done in a timer-based interrupt service routine. TP can be reset to 0 by software after reading.

Related information

Functional description on page 3962

21.7 Registers

21.7.1 Register overview - access mode glossary

Table 993 Register overview - access mode glossary

Keyword	Description
E	Access protection using PROT register PROTE.
SE	Access protection using PROT register PROTSE.
APU-PNi (i=0-3)	Protection group consisting of registers Ni_ACCEN_WRA, Ni_ACCEN_WRB, Ni_ACCEN_RDA, Ni_ACCEN_RDB, Ni_ACCEN_VM, Ni_ACCEN_PRS.
PNi	Access protection using *APU-PNi* registers.
APU-P4	Protection group consisting of registers ACCEN_WRA, ACCEN_WRB, ACCEN_RDA, ACCEN_RDB, ACCEN_VM, ACCEN_PRS.
P4	Access protection using *APU-P4* registers.
BE	Always returns a Bus Error.
U	No access restrictions.
PROT	Access restrictions as defined in the PROT register access rules.
nBE	Indicates that no Bus Error is generated when accessing this address range, even though it is either an access to an undefined address or the access does not follow the given rules.
M	Indicates a module specific access condition. Refer to the register description for details of the specific access condition.

21.7.2 Register overview - CAN domain SFR (ascending offset address)

Table 994 Register overview - CAN domain SFR (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CLC	Clock Control Register	00000_H	P4	P4, SV, E	See 3989	3989
OCS	OCDS Control and Status Register	00004_H	P4	SV, P4	Debug Reset	3990
ID	Module Identification Register	00008_H	P4	BE	PowerOn Reset	3991
RST_CTRLA	Reset Control Register A	$0000C_H$	P4	P4, SV, E	Application Reset	3991
RST_CTRLB	Reset Control Register B	00010_H	P4	P4, SV, E	Application Reset	3992
RST_STAT	Reset Status Register	00014_H	P4	BE	Application Reset	3993
PROTE	PROT Register Endinit	00018_H	U	SV, PROT	Application Reset	3993
PROTSE	PROT Register Safe Endinit	$0001C_H$	U	SV, PROT	Application Reset	3995

(table continues...)

Table 994 (continued) Register overview - CAN domain SFR (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ACCEN_WRA	Write access enable register A	00030_H	U	SE, SV	Application Reset	3997
ACCEN_WRB	Write access enable register B	00034_H	U	SE, SV	Application Reset	3997
ACCEN_RDA	Read access enable register A	00038_H	U	SE, SV	Application Reset	3998
ACCEN_RDB	Read access enable register B	$0003C_H$	U	SE, SV	Application Reset	3998
ACCEN_VM	VM access enable register	00040_H	U	SE, SV	Application Reset	3999
ACCEN_PRS	PRS access enable register	00044_H	U	SE, SV	Application Reset	3999
MCR	Module Control Register	00070_H	P4	P4	See 4001	4001
BUFADR	Buffer receive address and transmit address	00074_H	P4	P4	Kernel Reset	4003
MECR	Measure Control Register	00080_H	P4	P4	Kernel Reset	4004
MESTAT	Measure Status Register	00084_H	P4	P4	Kernel Reset	4005
WDT	CRE Watchdog timer register	00088_H	P4	SV, E, P4	Kernel Reset	4006
Ni_ACCEN_WRA	Node i Write access enable register A	00100_H+i *400_H	U	SE, SV	Application Reset	4007
Ni_ACCEN_WRB	Node i Write access enable register B	00104_H+i *400_H	U	SE, SV	Application Reset	4008
Ni_ACCEN_RDA	Node i Read access enable register A	00108_H+i *400_H	U	SE, SV	Application Reset	4008
Ni_ACCEN_RDB	Node i Read access enable register B	$0010C_H$+ i*400_H	U	SE, SV	Application Reset	4009
Ni_ACCEN_VM	Node i VM access enable register	00110_H+i *400_H	U	SE, SV	Application Reset	4009
Ni_ACCEN_PRS	Node i PRS access enable register	00114_H+i *400_H	U	SE, SV	Application Reset	4010
Ni_STARTADR	Node i Start Address	00120_H+i *400_H	PNi	SV, E, PNi	Application Reset	4010
Ni_ENDADR	Node i End Address	00124_H+i *400_H	PNi	SV, E, PNi	Application Reset	4011
Ni_INTRSIG	Node i Interrupt Signalling Register	00128_H+i *400_H	PNi	nBE	Kernel Reset	4011
Ni_G0INTR	Node i Interrupt routing for Group 0	$0012C_H$+ i*400_H	PNi	PNi	Kernel Reset	4013
Ni_G1INTR	Node i Interrupt routing for Group 1	00130_H+i *400_H	PNi	PNi	Kernel Reset	4014

(table continues...)

Table 994 (continued) Register overview - CAN domain SFR (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
Ni_G2INTR	Node i Interrupt routing for Group 2	00134_H+i *400_H	PNi	PNi	Kernel Reset	4015
Ni_TIMER_CCR	Node i Timer Clock Control Register	00138_H+i *400_H	PNi	PNi	Kernel Reset	4015
Ni_TIMER_TXTRIG0	Node i Timer Transmit Trigger 0 Register	$0013C_H$+ i*400_H	PNi	PNi	Kernel Reset	4016
Ni_TIMER_TXTRIG1	Node i Timer Transmit Trigger 1 Register	00140_H+i *400_H	PNi	PNi	Kernel Reset	4017
Ni_TIMER_TXTRIG2	Node i Timer Transmit Trigger 2 Register	00144_H+i *400_H	PNi	PNi	Kernel Reset	4018
Ni_TIMER_RXTOUT	Node i Timer Receive Timeout Register	00148_H+i *400_H	PNi	PNi	Kernel Reset	4018
Ni_PORTCTRL	Node i Port Control Register	$0014C_H$+ i*400_H	PNi	PNi	Kernel Reset	4019
Ni_CRE_CONFIG	Node i CRE Configuration Register	00150_H+i *400_H	PNi	SV, E, PNi	Kernel Reset	4020
Ni_CRE_CONFIGADR	Node i CRE Configuration Start Address	00154_H+i *400_H	PNi	SV, E, PNi	Kernel Reset	4022
Ni_CRE_HBUF_RXz_CONFIG	Node i Receive Host Buffer z Configuration	00158_H+i *400_H+z*8	PNi	PNi	Kernel Reset	4022
Ni_CRE_HBUF_RXz_STAT	Node i Receive Host Buffer z Status	$0015C_H$+ i*400_H+z*8	PNi	PNi	Kernel Reset	4024
Ni_CRE_HBUF_TXz_CONFIG	Node i Transmit Host Buffer z Configuration	00168_H+i *400_H+z*8	PNi	PNi	Kernel Reset	4025
Ni_CRE_HBUF_TXz_STAT	Node i Transmit Host Buffer z Status	$0016C_H$+ i*400_H+z*8	PNi	PNi	Kernel Reset	4026
Ni_CRE_IR	Node i CRE Interrupt Register	$0018C_H$+ i*400_H	PNi	PNi	Kernel Reset	4028
Ni_IDMU_FRTCONFIG	Node i Frame Rate Measure Table Configuration	00190_H+i *400_H	PNi	SV, E, PNi	Kernel Reset	4029
Ni_IDMU_RXTPCFG	Node i Rx Throughput Measure configuration	00194_H+i *400_H	PNi	SV, E, PNi	Kernel Reset	4030
Ni_ERRCTRL	Node i CRE Error control register	00198_H+i *400_H	P4	SV, E, P4	Kernel Reset	4031
Ni_CREL	Node i Core Release Register	00200_H+i *400_H	PNi	nBE	Kernel Reset	4032
Ni_ENDN	Node i Endian Register	00204_H+i *400_H	PNi	nBE	Kernel Reset	4033

(table continues...)

Table 994 (continued) Register overview - CAN domain SFR (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
Ni_DBTP	Node i Data Bit Timing & Prescaler Register	$0020C_H$+ i*400_H	PNi	PNi	See 4034	4034
Ni_TEST	Node i Test Register	00210_H+i *400_H	PNi	PNi	See 4035	4035
Ni_RWD	Node i RAM Watchdog	00214_H+i *400_H	PNi	PNi	Kernel Reset	4036
Ni_CCCR	Node i CC Control Register	00218_H+i *400_H	PNi	PNi	See 4037	4037
Ni_NBTP	Node i Nominal Bit Timing & Prescaler Register	$0021C_H$+ i*400_H	PNi	PNi	See 4040	4040
Ni_TSCC	Node i Timestamp Counter Configuration	00220_H+i *400_H	PNi	PNi	Kernel Reset	4041
Ni_TSCV	Node i Timestamp Counter Value	00224_H+i *400_H	PNi	PNi	Kernel Reset	4042
Ni_TOCC	Node i Timeout Counter Configuration	00228_H+i *400_H	PNi	PNi	Kernel Reset	4043
Ni_TOCV	Node i Timeout Counter Value	$0022C_H$+ i*400_H	PNi	PNi	Kernel Reset	4044
Ni_ECR	Node i Error Counter Register	00240_H+i *400_H	PNi	nBE	Kernel Reset	4045
Ni_PSR	Node i Protocol Status Register	00244_H+i *400_H	PNi	nBE	Kernel Reset	4045
Ni_TDCR	Node i Transmitter Delay Compensation Register	00248_H+i *400_H	PNi	PNi	Kernel Reset	4049
Ni_IR	Node i Interrupt Register	00250_H+i *400_H	PNi	PNi	Kernel Reset	4050
Ni_IE	Node i Interrupt Enable	00254_H+i *400_H	PNi	PNi	Kernel Reset	4053
Ni_GFC	Node i Global Filter Configuration	00280_H+i *400_H	PNi	PNi	Kernel Reset	4056
Ni_SIDFC	Node i Standard ID Filter Configuration	00284_H+i *400_H	PNi	PNi	See 4057	4057
Ni_XIDFC	Node i Extended ID Filter Configuration	00288_H+i *400_H	PNi	PNi	Kernel Reset	4058
Ni_XIDAM	Node i Extended ID AND Mask	00290_H+i *400_H	PNi	PNi	Kernel Reset	4058
Ni_HPMS	Node i High Priority Message Status	00294_H+i *400_H	PNi	nBE	Kernel Reset	4059
Ni_NDAT1	Node i New Data 1	00298_H+i *400_H	PNi	PNi	Kernel Reset	4060
Ni_NDAT2	Node i New Data 2	$0029C_H$+ i*400_H	PNi	PNi	Kernel Reset	4060

(table continues...)

Table 994 (continued) Register overview - CAN domain SFR (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
Ni_RXF0C	Node i Rx FIFO 0 Configuration	$002A0_H$+ i*400_H	PNi	PNi	Kernel Reset	4061
Ni_RXF0S	Node i Rx FIFO 0 Status	$002A4_H$+ i*400_H	PNi	nBE	Kernel Reset	4062
Ni_RXF0A	Node i Rx FIFO 0 Acknowledge	$002A8_H$+ i*400_H	PNi	PNi	Kernel Reset	4063
Ni_RXBC	Node i Rx Buffer Configuration	$002AC_H$+ i*400_H	PNi	PNi	See 4063	4063
Ni_RXF1C	Node i Rx FIFO 1 Configuration	$002B0_H$+ i*400_H	PNi	PNi	Kernel Reset	4064
Ni_RXF1S	Node i Rx FIFO 1 Status	$002B4_H$+ i*400_H	PNi	nBE	Kernel Reset	4065
Ni_RXF1A	Node i Rx FIFO 1 Acknowledge	$002B8_H$+ i*400_H	PNi	PNi	Kernel Reset	4066
Ni_RXESC	Node i Rx Buffer/FIFO Element Size Configuration	$002BC_H$+ i*400_H	PNi	PNi	Kernel Reset	4067
Ni_TXBC	Node i Tx Buffer Configuration	$002C0_H$+ i*400_H	PNi	PNi	See 4068	4068
Ni_TXFQS	Node i Tx FIFO/Queue Status	$002C4_H$+ i*400_H	PNi	nBE	Kernel Reset	4069
Ni_TXESC	Node i Tx Buffer Element Size Configuration	$002C8_H$+ i*400_H	PNi	PNi	See 4070	4070
Ni_TXBRP	Node i Tx Buffer Request Pending	$002CC_H$+ i*400_H	PNi	PNi	Kernel Reset	4072
Ni_TXBAR	Node i Tx Buffer Add Request	$002D0_H$+ i*400_H	PNi	PNi	Kernel Reset	4073
Ni_TXBCR	Node i Tx Buffer Cancellation Request	$002D4_H$+ i*400_H	PNi	PNi	Kernel Reset	4073
Ni_TXBTO	Node i Tx Buffer Transmission Occurred	$002D8_H$+ i*400_H	PNi	nBE	Kernel Reset	4074
Ni_TXBCF	Node i Tx Buffer Cancellation Finished	$002DC_H$+ i*400_H	PNi	nBE	Kernel Reset	4074
Ni_TXBTIE	Node i Tx Buffer Transmission Interrupt Enable	$002E0_H$+i *400_H	PNi	PNi	Kernel Reset	4075
Ni_TXBCIE	Node i Tx Buffer Cancellation Finished Interrupt Enable	$002E4_H$+i *400_H	PNi	PNi	Kernel Reset	4075
Ni_TXEFC	Node i Tx Event FIFO Configuration	$002F0_H$+i *400_H	PNi	PNi	Kernel Reset	4076
Ni_TXEFS	Node i Tx Event FIFO Status	$002F4_H$+i *400_H	PNi	nBE	Kernel Reset	4077

(table continues...)

Table 994 (continued) Register overview - CAN domain SFR (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
Ni_TXEFA	Node i Tx Event FIFO Acknowledge	$002F8_H$+i*400_H	PNi	PNi	Kernel Reset	4077
Ni_TSU_CREL	Node i TSU Core Release Register	00360_H+i*400_H	PNi	nBE	Kernel Reset	4078
Ni_TSU_TSCFG	Node i Timestamp Configuration	00364_H+i*400_H	PNi	PNi	Kernel Reset	4078
Ni_TSU_TSS1	Node i Timestamp Status 1	00368_H+i*400_H	PNi	nBE	Kernel Reset	4080
Ni_TSU_TSS2	Node i Timestamp Status 2	$0036C_H$+i*400_H	PNi	nBE	Kernel Reset	4081
Ni_TSU_TSm	Node i Timestamp m	00370_H+i*400_H+m*4	PNi	nBE	Kernel Reset	4082
Ni_TSU_ATB	Node i Actual Timebase	$003B0_H$+i*400_H	PNi	U	Kernel Reset	4082

21.7.3 Register overview - CAN CRE Receive Host Buffers (ascending offset address)

Table 995 Register overview - CAN CRE Receive Host Buffers (ascending offset address)

Short name	Long name	Offset address	See
RHBUFk_UCRH	Uni-cast Routing Header k	00020_H+k*60_H	4101
RHBUFk_MCRH	Multi-cast Routing Header k	00020_H+k*60_H	4102
RHBUFk_THEAD_INTRD	Timing Header k Intrusion Detection Information	00024_H+k*60_H	4103
RHBUFk_THEAD_RXTS	Timing Header k Rx Timestamp	00028_H+k*60_H	4104
RHBUFk_CRC	CRE computed CRC	$0002C_H$+k*60_H	4104
RHBUFk_R0	RHBUF k Register 0	00030_H+k*60_H	4105
RHBUFk_LPDUH1	Long PDU header 1	00030_H+k*60_H	4105
RHBUFk_R1	RHBUF k Register 1	00034_H+k*60_H	4106
RHBUFk_LPDUH0	Long PDU header 0	00034_H+k*60_H	4107
RHBUFk_SPDUH	Short PDU header	00034_H+k*60_H	4108
RHBUFk_RHBUF_DBm	RHBUF k Data Byte m	00038_H+k*60_H+m	4109

21.7.4 Register overview - CAN CRE Standard Frame Rate Table (ascending offset address)

Table 996 Register overview - CAN CRE Standard Frame Rate Table (ascending offset address)

Short name	Long name	Offset address	See
SFRk_FR	Standard ID Frame Rate Measure	00468_H+k*4	4119

21.7.5 Register overview - CAN CRE Standard Routing Table (ascending offset address)

Table 997 Register overview - CAN CRE Standard Routing Table (ascending offset address)

Short name	Long name	Offset address	See
SRTk_UCR	Standard routing table Uni-cast Rule	00168_H+k*4	4113
SRTk_MCR	Standard routing table Multi-cast Rule	00168_H+k*4	4114
SRTk_PR	Standard routing table PDU Routing Rule	00168_H+k*4	4115

21.7.6 Register overview - CAN CRE Standard Timestamp Database (ascending offset address)

Table 998 Register overview - CAN CRE Standard Timestamp Database (ascending offset address)

Short name	Long name	Offset address	See
STSDk_RTS	Standard ID Reference Timestamp	$005E8_H$+k*4	4120

21.7.7 Register overview - CAN CRE configuration table (ascending offset address)

Table 999 Register overview - CAN CRE configuration table (ascending offset address)

Short name	Long name	Offset address	See
CRE_STD_RT_PARAM	STD ID Routing table parameters	00000_H	4097
CRE_XTD_RT_PARAM	XTD ID Routing table parameters	00004_H	4098
CRE_STD_FRT_PARAM	STD ID Frame rate measure table parameters	00008_H	4098
CRE_XTD_FRT_PARAM	XTD ID Frame rate measure table parameters	$0000C_H$	4099
CRE_STD_TSD_PARAM	STD ID Timestamp database parameters	00010_H	4099
CRE_XTD_TSD_PARAM	XTD ID Timestamp database parameters	00014_H	4100
CRE_ABORT_SEQ	CRE abort sequence register	00018_H	4100

21.7.8 Register overview - CAN CRE Transmit Host Buffers (ascending offset address)

Table 1000 Register overview - CAN CRE Transmit Host Buffers (ascending offset address)

Short name	Long name	Offset address	See
THBUFk_CRC	Tx Host Buffer CRC	$000E0_H$+k*50_H	4109
THBUFk_T0	Transmit Host Buffer Word 0	$000E4_H$+k*50_H	4110
THBUFk_T1	Transmit Host Buffer Word 1	$000E8_H$+k*50_H	4111
THBUFk_DBm	Transmit Host Buffer Data Byte m	$000EC_H$+k*50_H +m	4113

21.7.9 Register overview - CAN CRE Extended Frame Rate Table (ascending offset address)

Table 1001 Register overview - CAN CRE Extended Frame Rate Table (ascending offset address)

Short name	Long name	Offset address	See
XFRk_FR	Extended ID Frame Rate Measure	00568_H+k*4	4120

21.7.10 Register overview - CAN CRE Extended Routing Table (ascending offset address)

Table 1002 Register overview - CAN CRE Extended Routing Table (ascending offset address)

Short name	Long name	Offset address	See
XRTk_UCR	Extended routing table Uni-cast Rule	00368_H+k*4	4116
XRTk_MCR	Extended routing table Multi-cast Rule	00368_H+k*4	4117
XRTk_PR	Extended ID routing table PDU Routing Rule	00368_H+k*4	4118

21.7.11 Register overview - CAN CRE Extended Timestamp Database (ascending offset address)

Table 1003 Register overview - CAN CRE Extended Timestamp Database (ascending offset address)

Short name	Long name	Offset address	See
XTSDk_RTS	Extended ID Reference Timestamp	$007E8_H$+k*4	4121

21.7.12 Register overview - CAN Rx Buffer and FIFO Element (ascending offset address)

Table 1004 Register overview - CAN Rx Buffer and FIFO Element (ascending offset address)

Short name	Long name	Offset address	See
RXMSGk_R0	Register 0	00000_H+k*48_H	4087
RXMSGk_R1A	Register 1 A	00004_H+k*48_H	4088
RXMSGk_R1B	Register 1 B	00004_H+k*48_H	4089
RXMSGk_DBm	Data Byte m	00008_H+k*48_H +m	4090

21.7.13 Register overview - CAN Standard Message ID Filter Element (ascending offset address)

Table 1005 Register overview - CAN Standard Message ID Filter Element (ascending offset address)

Short name	Long name	Offset address	See
STDMSGk_S0	Standard Message 0	00000_H+k*4	4082

21.7.14 Register overview - CAN Tx Buffer Element (ascending offset address)

Table 1006 Register overview - CAN Tx Buffer Element (ascending offset address)

Short name	Long name	Offset address	See
TXMSGk_T0	Transmit Buffer 0	00000_H+k*48_H	4094
TXMSGk_T1	Transmit Buffer 1	00004_H+k*48_H	4095
TXMSGk_DBm	Data Byte m	00008_H+k*48_H +m	4097

21.7.15 Register overview - CAN Tx Event FIFO Element (ascending offset address)

Table 1007 Register overview - CAN Tx Event FIFO Element (ascending offset address)

Short name	Long name	Offset address	See
TXEVENTk_E0	Event 0	00000_H+k*8	4091
TXEVENTk_E1A	Event 1A	00004_H+k*8	4091
TXEVENTk_E1B	Event 1B	00004_H+k*8	4093

21.7.16 Register overview - CAN Extended Message ID Filter Element (ascending offset address)

Table 1008 Register overview - CAN Extended Message ID Filter Element (ascending offset address)

Short name	Long name	Offset address	See
EXTMSGk_F0	Filter Element 0	00000_H+k*8	4084
EXTMSGk_F1	Filter Element 1	00004_H+k*8	4085

21.7.17 Clock Control Register

The Clock Control Register CLC allows the programmer to adapt the functionality and power consumption of the module to the requirements of the application.

Register CLC controls the module clock signal and the reactivity to the sleep signal.

CLC Offset address: 00000_H

Clock Control Register Reset values see: Table 1009

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												EDIS	0	DISS	DISR
r												rw	r	rh	rw

Field	Bits	Type	Description
DISR	0	rw	**Module Disable Request Bit** Used for enable/disable control of the module. The synchronous and asynchronous clock is switched on/off. Note that no register access is possible to any register while module is disabled. A disable request is granted, if the M_CAN clock is disabled, or all M_CAN nodes acknowledge the disable request. 0_B On request: enable the module clock 1_B Off request: stop the module clock
DISS	1	rh	**Module Disable Status Bit** 0_B Module clock is enabled 1_B Off: module is not clocked
EDIS	3	rw	**Sleep Mode Enable Control** Used to control the module's reaction to sleep mode. 0_B Sleep mode request is enabled and functional 1_B Module disregards the sleep mode control signal
0	2, 31:4	r	**Reserved** Read as 0; should be written with 0.

Table 1009 Reset values of CLC

Reset type	Reset value	Note
Application Reset	0000 0003$_H$	
After Boot-FW Value	0000 0000$_H$	After CAN BSL execution

21.7.18 OCDS Control and Status Register

The OCDS Control and Status register OCS controls the debug and trace behavior by selecting suspend modes and OTGB trigger sets. When OCDS is disabled the suspend control is ineffective.

OCS Offset address: 00004$_H$

OCDS Control and Status Register Debug Reset value: 0000 0000$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		SUSSTA	SUS_P	SUS				0							
r		rh	w	rw				r							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												TG_P	TGB	TGS	
r												w	rw	rw	

Field	Bits	Type	Description
TGS	1:0	rw	**Trigger Set for OTGB0/1** 00_B No Trigger Set output 01_B TS16_CAN 10_B Reserved 11_B Reserved
TGB	2	rw	**OTGB0/1 Bus Select** 0_B Trigger Set is output on OTGB0 1_B Trigger Set is output on OTGB1
TG_P	3	w	**TGS, TGB Write Protection** TGS and TGB are only written when TG_P is 1, otherwise unchanged. Read as 0.
SUS	27:24	rw	**OCDS Suspend Control** Controls the sensitivity to the suspend signal coming from the OCDS Trigger Switch (OTGS) Not listed combinations are reserved. 0_H Will not suspend 1_H Hard suspend. Clock is off immediately. Do not use this mode in normal CAN applications, this mode is meant for debugging the peripheral IP. 2_H Soft suspend mode Soft suspend of CAN nodes. **others,** Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SUS_P	28	w	**SUS Write Protection** SUS is only written when SUS_P is 1, otherwise unchanged. Read as 0.
SUSSTA	29	rh	**Suspend State** 0_B Module is not (yet) suspended 1_B Module is suspended
0	23:4, 31:30	r	**Reserved** Read as 0; should be written with 0.

Table 1010 Access mode restrictions of OCS sorted by descending priority

Mode name	Access mode		Description
write 1 to .**TG_P**	rw	TGB, TGS	Set TG_P during write access
write 1 to .**SUS_P**	rw	SUS	Set SUS_P during write access
(default)	r	SUS, TGB, TGS	

21.7.19 Module Identification Register

ID Offset address: 00008_H

Module Identification Register PowerOn Reset value: $00B8\ C004_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MOD_NUM															
r															

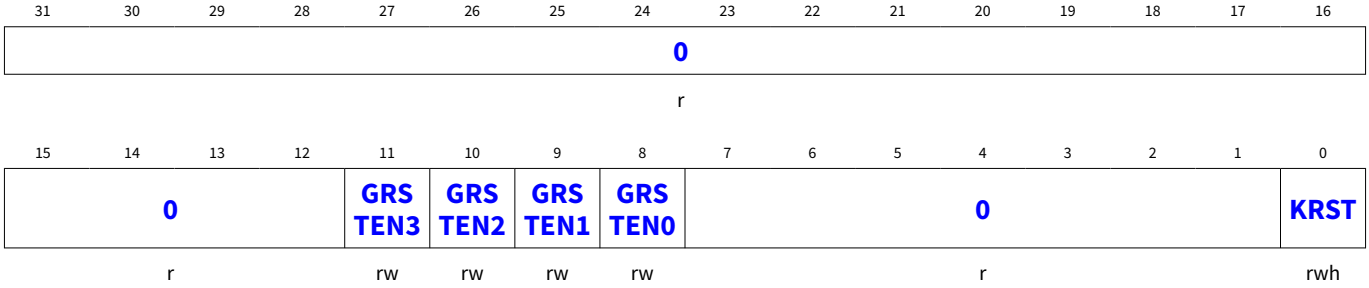
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MOD_TYPE								MOD_REV							
r								r							

Field	Bits	Type	Description
MOD_REV	7:0	r	**Module Revision** Indicates the revision number of the implementation.
MOD_TYPE	15:8	r	**Module Type** This internal marker is fixed to C0.
MOD_NUM	31:16	r	**Module Number** Indicates the module identification number (00B8 = CAN).

21.7.20 Reset Control Register A

RST_CTRLA Offset address: $0000C_H$

Reset Control Register A Application Reset value: $0000\ 0000_H$

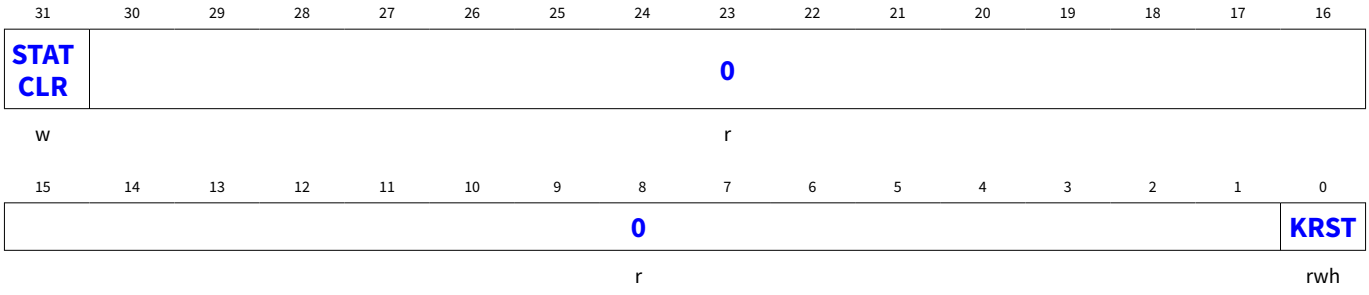


Field	Bits	Type	Description
KRST	0	rwh	**Kernel Reset** Request a kernel reset. The reset is executed if the reset bits of both kernel reset registers are set. KRST is cleared after the kernel reset was executed. 0_B No action 1_B A kernel reset was requested
GRSTENx (x=0-3)	x+8	rw	**Enable for Global Module Reset Group x** 0_B global module reset group x does not have any effect 1_B global module reset group x results in a kernel reset
0	7:1, 31:12	r	**Reserved** Read as 0; should be written with 0.

21.7.21 Reset Control Register B

RST_CTRLB Offset address: 00010_H

Reset Control Register B Application Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
KRST	0	rwh	**Kernel Reset** Request a kernel reset. The reset is executed if the reset bits of both kernel reset registers are set. KRST is cleared after the kernel reset was executed. 0_B No action 1_B A kernel reset was requested

(table continues...)

(continued)

Field	Bits	Type	Description
STATCLR	31	w	**Kernel Reset Status Clear** Clears all status bits in RST_STAT when 1 is written. Read will return 0. 0_B No action 1_B Write with ´1´ clears bits STAT.GRSTx and bit STAT.KRST .
0	30:1	r	**Reserved** Read as 0; should be written with 0.

21.7.22 Reset Status Register

The reset status register contains the status bits for kernel reset (KRST) and the global module reset groups (GRSTx).

RST_STAT Offset address: 00014_H

Reset Status Register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				GRST3	GRST2	GRST1	GRST0	0							KRST
r				rh	rh	rh	rh	r							rh

Field	Bits	Type	Description
KRST	0	rh	**Kernel Reset Status** Indicates an executed kernel reset. RST_STAT.KRST is set after the execution of a kernel reset in the same clock cycle in which the reset bits are cleared. Clear KRST by setting bit STATCLR in register RST_CTRLB.
GRSTx (x=0-3)	x+8	rh	**Status for Global Module Reset Group x** 0_B Reset was not triggered by global reset group x 1_B Reset was triggered by global reset group x
0	7:1, 31:12	r	**Reserved** Read as 0; should be written with 0.

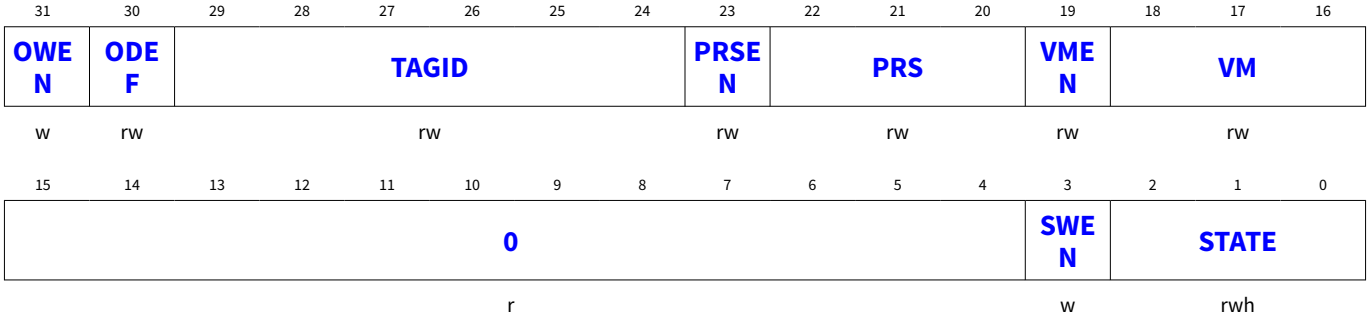
21.7.23 PROT Register Endinit

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner and in some PROT states, the secure master to update the PROT state as described in the PROT mechanism.

The PROTE register controls lock / unlock of the local Endinit (E) protected control registers.

PROTE Offset address: 00018_H

PROT Register Endinit Application Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
VM	18:16	rw	**Virtual machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual machine definition enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.
PRSEN	23	rw	**Protection set definition enable for PROT owner** 0_B **Disable**: PRS is not part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG-ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner

(table continues...)

(continued)

Field	Bits	Type	Description
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; should be written with 0.

Table 1011 Access mode restrictions of PROTE sorted by descending priority

Mode name	Access mode		Description
write 1 to .**SWEN**	rwh	STATE	
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

21.7.24 PROT Register Safe Endinit

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner and in some PROT states, the secure master to update the PROT state as described in the PROT mechanism.

The PROTSE register controls lock / unlock of the local Safe Endinit (SE) protected control registers.

PROTSE Offset address: $0001C_H$

PROT Register Safe Endinit Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OWEN	ODEF	TAGID						PRSEN	PRS			VMEN	VM		
w	rw	rw						rw	rw			rw	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												SWEN	STATE		
r												w	rwh		

Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
VM	18:16	rw	**Virtual machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual machine definition enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.
PRSEN	23	rw	**Protection set definition enable for PROT owner** 0_B **Disable**: PRS is not part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG-ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; should be written with 0.

Table 1012 Access mode restrictions of PROTSE sorted by descending priority

Mode name	Access mode		Description
write 1 to .**SWEN**	rwh	STATE	
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

21.7.25 Write access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

ACCEN_WRA Offset address: 00030_H

Write access enable register A Application Reset value: $1000\ 0003_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	**EN30**	**EN29**	**EN28**	**EN27**	**EN26**	**EN25**	**EN24**	**EN23**	**EN22**	**EN21**	**EN20**	**EN19**	**EN18**	**EN17**	**EN16**
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	**EN14**	**EN13**	**EN12**	**EN11**	**EN10**	**EN09**	**EN08**	**EN07**	**EN06**	**EN05**	**EN04**	**EN03**	**EN02**	**EN01**	**EN00**
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access

21.7.26 Write access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

ACCEN_WRB Offset address: 00034_H

Write access enable register B Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								**EN39**	**EN38**	**EN37**	**EN36**	**EN35**	**EN34**	**EN33**	**EN32**
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access
0	31:8	r	**Reserved** Read as 0; should be written with 0.

21.7.27 Read access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

ACCEN_RDA Offset address: 00038_H

Read access enable register A Application Reset value: $FFFF\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access

21.7.28 Read access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

ACCEN_RDB Offset address: $0003C_H$

Read access enable register B Application Reset value: $0000\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access
0	31:8	r	**Reserved** Read as 0; should be written with 0.

21.7.29 VM access enable register

This register defines which virtual machine encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each VMx.

ACCEN_VM Offset address: 00040_H

VM access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for VM ID q** This bit enables read access to the access protected region for transactions with the VM ID q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for VM ID q** This bit enables write access to the access protected region for transactions with the VM ID q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

21.7.30 PRS access enable register

This register defines which protection register set encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each PRSx.

ACCEN_PRS Offset address: 00044_H

PRS access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for PRS q** This bit enables read access to the access protected region for transactions with the PRS q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for PRS q** This bit enables write access to the access protected region for transactions with the PRS q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

21.7.31 Module Control Register

The Module Control Register MCR contains basic settings that determine the operation of the MCMCAN module.

The write access to the lowest byte of the MCR register becomes only valid, if and only if, MCR.CCCE and MCR.CI are already set during write access. To switch the clocks on or off, the bits of MCR.CCCE and MCR.CI have to be reset afterwards. Before this sequence hasn't taken place, no write access to the corresponding nodes, can be done.

***Note**: If the baud rate logic is supplied from an unstable clock source, or no clock at all, the CAN functionality is not guaranteed.*

To be able to change the clock settings the following programming sequence needs to be met:
uwTemp = CANn_MCR.U;
uwTemp |= (0xC0000000 | CLKSELx);
CANn_MCR.U = uwTemp;
uwTemp &= ~0xC0000000;
CANn_MCR.U = uwTemp;
The clock settings for CAN nodes becomes active.
To be able to start the RAM initialization, the following programming sequence need to be met:
CANn_MCR |= 0xC0000000;
Wait until CANn_MCR.RBUSY is 0b
Set CANn_MCR.RINIT to 0b
Set CANn_MCR.RINIT to 1b
Dummy read CANn_MCR
Wait until CANn_MCR.RBUSY is 0b
Set CANn_MCR.RINIT to 0b
CANn_MCR &= ~0xC0000000;
RAM initialization is finished

MCR Offset address: 00070_H

Module Control Register Reset values see: Table 1013

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CCCE	CI	RINIT	RBUSY	DXCM	NODE			0							
rw	rw	rw	rh	rw	rw			r							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								CLKSEL3		CLKSEL2		CLKSEL1		CLKSEL0	
r								rw		rw		rw		rw	

Field	Bits	Type	Description
CLKSEL0	1:0	rw	**Clock Select 0** This bit-field is MCR.CI and MCR.CCCE protected. 00_B No clock supplied 01_B The asynchronous clock source is switched on 10_B The synchronous clock source is switched on 11_B Both clock sources are switched on

(table continues...)

(continued)

Field	Bits	Type	Description
CLKSEL1	3:2	rw	**Clock Select 1** This bit-field is MCR.CI and MCR.CCCE protected. 00_B No clock supplied 01_B The asynchronous clock source is switched on 10_B The synchronous clock source is switched on 11_B Both clock sources are switched on
CLKSEL2	5:4	rw	**Clock Select 2** This bit-field is MCR.CI and MCR.CCCE protected. 00_B No clock supplied 01_B The asynchronous clock source is switched on 10_B The synchronous clock source is switched on 11_B Both clock sources are switched on
CLKSEL3	7:6	rw	**Clock Select 3** This bit-field is MCR.CI and MCR.CCCE protected. 00_B No clock supplied 01_B The asynchronous clock source is switched on 10_B The synchronous clock source is switched on 11_B Both clock sources are switched on
NODE	26:24	rw	**Node** This bit-field determines the M_CAN node i which is used for debug over M_CAN. This bit-field only exists on CAN module 0. 000_B M_CAN0 … 011_B M_CAN3
DXCM	27	rw	**Debug Over CAN Messages Enable** This bit enables the debug over serial connections between DAP and CAN module 0. If enabled the lowest receive/transmit message buffer is reserved for debugger communication. DXCM is described in detail in the OCDS chapter. This bit only exists on CAN module 0. 0_B DXCM disabled 1_B DXCM enabled
RBUSY	28	rh	**RAM BUSY** This bit shows that the RAM Initialization is running. This bit is set back to 0b by hardware when the RAM intialization is completed.
RINIT	29	rw	**RAM Init** This bit is MCR.CI and MCR.CCCE protected. This bit starts the initialization of the RAM block to all 0x0. The RAM initialization is started only when this bit is changed from 0b to 1b and also RBUSY is 0b.

(table continues...)

(continued)

Field	Bits	Type	Description
CI	30	rw	**Change Init** Needs to be set to enable and disable clocks. 0_B Change Init disabled 1_B Change Init enabled (takes effect with CCCE:=1)
CCCE	31	rw	**Clock and RAM Change Enable** Needs to be set to enable and disable the clocks. 0_B Clock and RAM Change disabled 1_B Clock and RAM Change enabled (takes effect with CI:=1)
0	23:8	r	**Reserved** Shall read 0; shall be written with 0.

Table 1013 Reset values of MCR

Reset type	Reset value	Note
Kernel Reset	$0000\ 0000_H$	
After Boot-FW Value	$0000\ 000C_H$	After CAN BSL execution

21.7.32 Buffer receive address and transmit address

This register is used only when Debug over CAN Messages feature is used. This register assigns the start address to all features needing the message buffers inside the corresponding M_CAN, which are for receive and transmit

BUFADR Offset address: 00074_H

Buffer receive address and transmit address Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		RXBUF													
r		rw													

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		TXBUF													
r		rw													

Field	Bits	Type	Description
TXBUF	13:0	rw	**Transmit Buffer start address** This is the start address of the first dedicated transmit buffer.
RXBUF	29:16	rw	**Receive Buffer start address** This is the start address of the first dedicated receive buffer.
0	15:14, 31:30	r	**Reserved** Read as 0; should be written with 0.

21.7.33 Measure Control Register

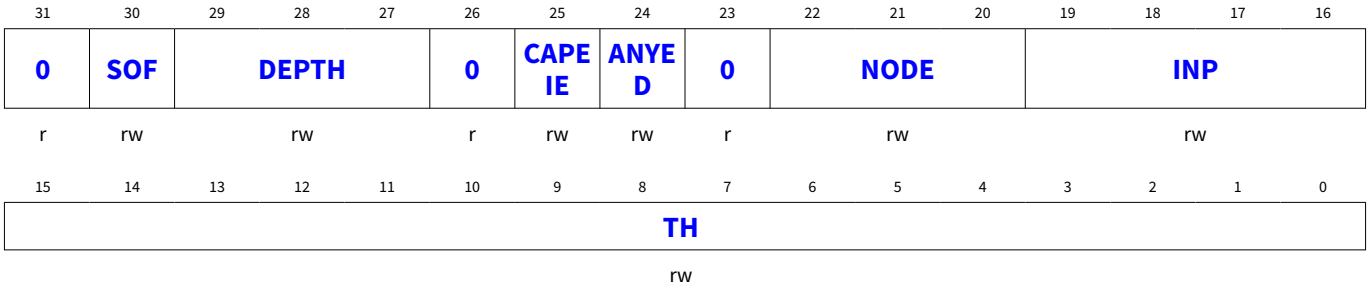
The Measure Control Register MECR controls the CAN edge timing measurement function for calibration purposes. This feature only exists on CAN module 0.

Oscillator calibration

This register supports the oscillator calibration on which the decision is taken to increase or decrease the frequency.

MECR Offset address: 00080_H

Measure Control Register Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
TH	15:0	rw	**Threshold** This bit-field contains the threshold value for the measurement timer. If TH = 0000, the timer is stopped and the capture function is disabled.
INP	19:16	rw	**Interrupt Node Pointer** INP selects the interrupt output line SRC_CANINTx (x = 0-15) for a capture event interrupt. 0_H Interrupt output line SRC_CANINT0 is selected … F_H Interrupt output line SRC_CANINT15 is selected
NODE	22:20	rw	**Node** This bit-field determines the CAN node i whose input line RXDCANi is used for start and capture of the measurement timer. 000_B Node 0 … 011_B Node 3
ANYED	24	rw	**Any Edge** This bit enables capture on any edge of CAN input line specified by NODE. 0_B Capture on falling (dominant) edge only 1_B Capture on rising (recessive) or falling (dominant) edge
CAPEIE	25	rw	**Capture Event Interrupt Enable** This bit enables the capture event interrupt. Bit-field INP selects the interrupt output line which becomes activated at this type of interrupt. 0_B Capture event interrupt is disabled 1_B Capture event interrupt is enabled

(table continues...)

(continued)

Field	Bits	Type	Description
DEPTH	29:27	rw	**Digital Glitch Filter Depth** DEPTH determines the number of input samples clocked with f_{SYNi} that are taken into account for the calculation of the floating average. The higher DEPTH is chosen to be, the longer the glitches that are suppressed and the longer the delay of the input signal introduced by this filter. 000_B off, default 001_B Filter depth of 8 cycles 010_B Filter depth of 16 cycles 011_B Filter depth of 32 cycles 100_B Filter depth of 64 cycles 101_B Filter depth of 128 cycles 110_B Filter depth of 255 cycles 111_B not allowed, reserved
SOF	30	rw	**Start Of Frame** This bit selects falling edge or any edge as measurement for start of frame detection. 0_B Measurement starts with any falling edge 1_B Measurement starts with falling Start of Frame edge. i.e any falling edge that occurs while the CAN node is in idle state
0	23, 26, 31	r	**Reserved** Read as 0; should be written with 0.

21.7.34 Measure Status Register

The Measure Status Register MESTAT contains the status information of the CAN edge timing measurement. This feature only exists on CAN module 0.

MESTAT Offset address: 00084_H

Measure Status Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0														CAPE	CAPRED
r														rwh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CAPT															
rh															

Field	Bits	Type	Description
CAPT	15:0	rh	**Captured Timer** This bit-field contains the captured measurement timer content. The timer itself is cleared and started by the first falling (dominant) edge of a CAN frame on the input line of the CAN node specified by MECR.NODE. The timer is incremented by the module control clock f_{SYNi} and will be stopped when $FFFF_H$ is reached. If MECR.TH = 0000, the timer is always stopped. A capture will take place if all the following conditions are met: **1.** MECR.TH > 0000 **2.** Timer is cleared and started by new frame **3.** Timer reaches MECR.TH **4.** This node is not sending and first edge (as specified by MECR.ANYED) after 3. occurs on input line Capture will be repeated for the following CAN frames until MECR.TH is cleared.
CAPRED	16	rh	**Captured Rising Edge** This bit indicates the type of edge that caused the last capture event. 0_B Capture occurred on falling (dominant) edge 1_B Capture occurred on rising (recessive) edge
CAPE	17	rwh	**Capture Event** This flag is set on a capture event. It must be reset by software. An interrupt request is generated if MECR.CAPEIE = 1. If CAPE=1, then no further measurement results are posted to MESTAT.CAPT and MESTAT.CAPRED. CAPE bit has to be cleared to re-enable update of MESTAT.CAPT and MESTAT.CAPRED. 0_B No capture event has occurred since last flag reset 1_B Capture event has occurred since last flag reset
0	31:18	r	**Reserved** Read as 0; should be written with 0.

21.7.35 CRE Watchdog timer register

The CRE Watchdog timer register to monitor timeout.

WDT Offset address: 00088_H

CRE Watchdog timer register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SWDP															
rwh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FWDP															EN
rw															rw

Field	Bits	Type	Description
EN	0	rw	**Enable watchdog** This bit enables/disables the watchdog timer
FWDP	15:1	rw	**Fast Watchdog prescaler** This bit-field is a configurable pre-scalar. This pre-scaler is configured by user for monitoring fast events that are part of WDG1 and WDG2. By default the pre-scaler is fixed to 16 when this bit-field is not set **1.** • Start condition : RxFIFO not empty • End condition : RxFIFO processing done **2.** • Start condition : THBUF full • End condition : THBUF empty
SWDP	31:16	rwh	**Slow Watchdog prescaler** This bit-field is a configurable pre-scalar. This pre-scaler is configured by user for monitoring slow events that are part of WDG2 and WDG3. By default the pre-scaler is fixed to 16 when this bit-field is not set **1.** • Start condition : THBUF empty • End condition : THBUF full **2.** • Start condition : RHBUF full • End condition : RHBUF empty **3.** • Start condition : RHBUF empty • End condition : RHBUF full

21.7.36 Node i Write access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

Ni_ACCEN_WRA (i=0-3) Offset address: $00100_H+i*400_H$

Node i Write access enable register A Application Reset value: $1000\ 0003_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access

21.7.37 Node i Write access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

Ni_ACCEN_WRB (i=0-3) Offset address: $00104_H+i*400_H$

Node i Write access enable register B Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access
0	31:8	r	**Reserved** Read as 0; should be written with 0.

21.7.38 Node i Read access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

Ni_ACCEN_RDA (i=0-3) Offset address: $00108_H+i*400_H$

Node i Read access enable register A Application Reset value: $FFFF\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access

21.7.39 Node i Read access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

Ni_ACCEN_RDB (i=0-3) Offset address: $0010C_H+i*400_H$

Node i Read access enable register B Application Reset value: $0000\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access
0	31:8	r	**Reserved** Read as 0; should be written with 0.

21.7.40 Node i VM access enable register

This register defines which virtual machine encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each VMx.

Ni_ACCEN_VM (i=0-3) Offset address: $00110_H+i*400_H$

Node i VM access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for VM ID q** This bit enables read access to the access protected region for transactions with the VM ID q. 0_B Disabled for read access 1_B Enabled for read access

(table continues...)

(continued)

Field	Bits	Type	Description
WRq (q=00-07)	q+16	rw	**Write access enable for VM ID q** This bit enables write access to the access protected region for transactions with the VM ID q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

21.7.41 Node i PRS access enable register

This register defines which protection register set encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each PRSx.

Ni_ACCEN_PRS (i=0-3) Offset address: $00114_H+i*400_H$

Node i PRS access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

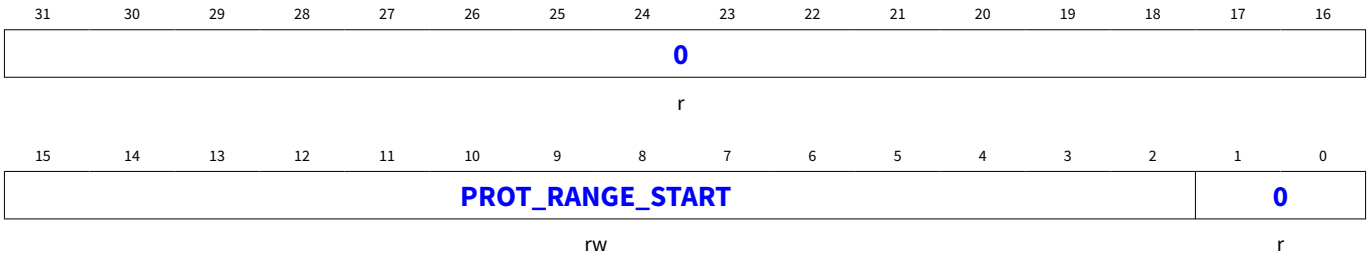
Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for PRS q** This bit enables read access to the access protected region for transactions with the PRS q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for PRS q** This bit enables write access to the access protected region for transactions with the PRS q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

21.7.42 Node i Start Address

In case the RAM shall not be protected, the STARTADR has to be higher than the corresponding ENDADR of the node.

Ni_STARTADR (i=0-3) Offset address: $00120_H+i*400_H$

Node i Start Address Application Reset value: $0000\ 0000_H$

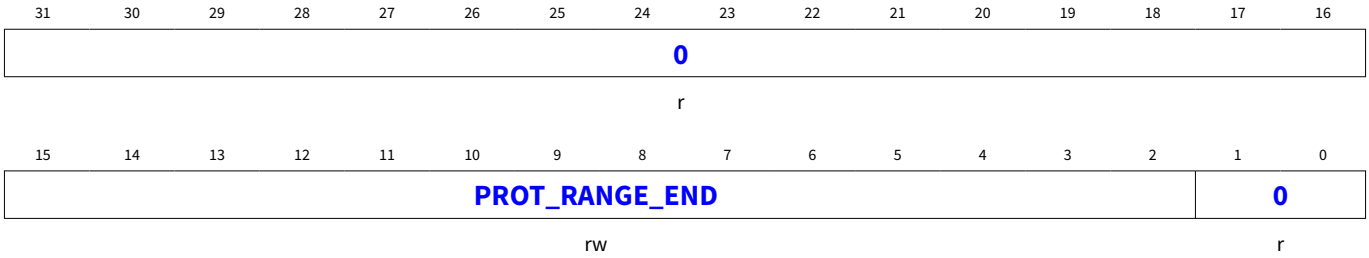


Field	Bits	Type	Description
PROT_RANGE_START	15:2	rw	**Message RAM start - START** The address within the RAM area of the MCMCAN, of node i, where the message RAM to be protected starts
0	1:0, 31:16	r	**Reserved** Read as 0; should be written with 0.

21.7.43 Node i End Address

Ni_ENDADR (i=0-3) Offset address: $00124_H+i*400_H$

Node i End Address Application Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
PROT_RANGE_END	15:2	rw	**Message RAM end - END** The address within the RAM area of the MCMCAN, of node i, where the message RAM to be protected ends
0	1:0, 31:16	r	**Reserved** Read as 0; should be written with 0.

21.7.44 Node i Interrupt Signalling Register

The groups by the Ni_G0INTR, Ni_G1INTR and G2INTR registers are also shown inside the INTRSIG (interrupt signalling register) register. Inside the interrupt signalling register a 1 means, that one of the corresponding bits inside the interrupt (status) register of the corresponding M_CAN node, at least one group member is showing an interrupt. Ni_INTRSIG is purely OR-ing the interrupt status bits of the group to enable SW to have proper handling of the bits. Writing to Ni_INTRSIG has no effect.

If Ni_INTRSIG is written, this shall have no effect on the interrupt status inside the M_CAN nodes. The bits have to be reset inside the corresponding M_CAN nodes, see register Ni_IR.

Ni_INTRSIG (i=0-3) Offset address: $00128_H+i*400_H$

Node i Interrupt Signalling Register Kernel Reset value: $0000\ 0000_H$

21 Controller Area Network interface (CAN)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0										CEI	IDMU	TXHBUF1	TXHBUF0	RXHBUF1	RXHBUF0
r										rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LOI	BOFF	SAFE	MOER	ALRT	WATI	HPE	TEFIFO	TRACO	TRAQ	RETI	RxF0N	RxF1N	RxF0F	RxF1F	REINT
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
REINT	0	rh	**A message stored in a receive buffer interrupt**
RxF1F	1	rh	**Receive FIFO1 is full interrupt**
RxF0F	2	rh	**Receive FIFO0 is full interrupt**
RxF1N	3	rh	**Receive FIFO1 got a new message interrupt**
RxF0N	4	rh	**Receive FIFO0 got a new message interrupt**
RETI	5	rh	**A receive timeout event interrupt**
TRAQ	6	rh	**A transmission queue event interrupt**
TRACO	7	rh	**A transmission control event interrupt**
TEFIFO	8	rh	**A Transmit Event FIFO Incident interrupt**
HPE	9	rh	**A high priority event interrupt**
WATI	10	rh	**A watermark interrupt has been reached**
ALRT	11	rh	**An alert interrupt**
MOER	12	rh	**Module error interrupt**
SAFE	13	rh	**The safety counter interrupt ELO**
BOFF	14	rh	**Bus Off Interrupt**
LOI	15	rh	**Last Error Interrupt**
RXHBUF0	16	rh	**CRE Rx Host Buffer 0 interrupt**
RXHBUF1	17	rh	**CRE Rx Host Buffer 1 interrupt**
TXHBUF0	18	rh	**CRE Tx Host Buffer 0 interrupt**
TXHBUF1	19	rh	**CRE Tx Host Buffer 1 interrupt**
IDMU	20	rh	**IDMU interrupts** Ni_CRE_IR.SFRMLI and XFRMLI events are mapped here.
CEI	21	rh	**CRE error interrupts** Ni_CRE_IR.IRSI, IWSI, CRCI, RWDTI and TWDTI events are mapped here
0	31:22	r	**Reserved** Read as 0; should be written with 0.

21.7.45 Node i Interrupt routing for Group 0

Ni_G0INTR is the first of three grouping registers. In this register, the interrupt line within the module is fixed. Please be reminded, that the interrupt sources need to be enabled to be mapped. The total module has 16 interrupts and the interrupt node can be chosen within Ni_G0INTR, Ni_G1INTR and Ni_G2INTR.

Meaning:

0000_B Interrupt output line SRC_CANINT0 is selected.

0001_B Interrupt output line SRC_CANINT1 is selected.

…B…

1110_B Interrupt output line SRC_CANINT14 is selected.

1111_B Interrupt output line SRC_CANINT15 is selected.

Note: MCMCAN has 22 interrupt groups and 16 interrupt lines, and the user can decide how the interrupt lines are shared within the groups.

Ni_G0INTR (i=0-3) Offset address: $0012C_H+i*400_H$

Node i Interrupt routing for Group 0 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LOI				BOFF				SAFE				MOER			
rw				rw				rw				rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ALRT				WATI				HPE				TEFIFO			
rw				rw				rw				rw			

Field	Bits	Type	Description
TEFIFO	3:0	rw	**Transmit Event FIFO Incidents** are mapped here. Ni_IR.TEFF (Transmit Event FIFO Full) and Ni_IR.TEFN (Transmit Event FIFO New Entry)
HPE	7:4	rw	**High Priority Events** are mapped here, giving Ni_IR.HPM an interrupt level
WATI	11:8	rw	**Watermark interrupts** are mapped here: Ni_IR.TEFW (Transmit FIFO warning interrupt reached), Ni_IR.RF1W (Receive FIFO 1 warning interrupt reached). Ni_IR.RF0W (Receive FIFO 0 warning interrupt reached)
ALRT	15:12	rw	**ALERTS** All kind of alerts are mapped here. Ni_IR.EW (warning status), Ni_IR.EP (error passive), Ni_IR.TSW (timestamp wrap around), Ni_IR.TEFL (Transmit Event FIFO Element Lost), Ni_IR.RF0L (Receive FIFO 0 Message Lost), Ni_IR.RF1L (Receive FIFO 1 Message Lost).
MOER	19:16	rw	**Module errors** Ni_IR.WDI (watchdog interrupt) and Ni_IR.MRAF (message RAM access failure) are mapped here.
SAFE	23:20	rw	**Safety counter overflow** The interrupt node for Ni_IR.ELO showing a safety counter overflow

(table continues...)

(continued)

Field	Bits	Type	Description
BOFF	27:24	rw	**Bus Off has been reached** Mapped to Ni_IR.BO flag indication the change in Bus_Off status. To get out of bus off, the Ni_CCCR.INIT bit has to be reset.
LOI	31:28	rw	**Last Error Interrupts** The interrupt sources Ni_IR.PED (Protocol Error in Data Phase) and Ni_IR.PEA (Protocol Error in Arbitration Phase) are signalled here.

21.7.46 Node i Interrupt routing for Group 1

Ni_ G1INTR has the same functionality as Ni_ G0INTR, but for other interrupt sources. The interrupt sources need to be enabled to be mapped.

Ni_G1INTR (i=0-3) Offset address: $00130_H+i*400_H$

Node i Interrupt routing for Group 1 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TRACO				TRAQ				RETI				RxF0N			
rw				rw				rw				rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RxF1N				RxF0F				RxF1F				REINT			
rw				rw				rw				rw			

Field	Bits	Type	Description
REINT	3:0	rw	**Message stored in dedicated receive buffer interrupt (Ni_IR.DRX)** is assigned to interrupt node.
RxF1F	7:4	rw	**IR.RF1F** Receive FIFO1 full interrupt assigned to an interrupt node
RxF0F	11:8	rw	**IR.RF0F** Receive FIFO0 full interrupt assigned to an interrupt node
RxF1N	15:12	rw	**IR.RF1N** Receive FIFO1 new message assigned to an interrupt node
RxF0N	19:16	rw	**IR.RF0N** Receive FIFO0 new message assigned to an interrupt node
RETI	23:20	rw	**Receive Timeouts** can be assigned here. Ni_IR.TOO (time-out event) and TE (Timer Event)
TRAQ	27:24	rw	**Transmission Queue Events** can be assigned here. Ni_IR.TFE Transmission FIFO Empty
TRACO	31:28	rw	**Interrupts of the transmission control** can be assigned here. Ni_IR.TCF (Transmission Cancellation Finished) and Ni_IR.TF (Transmission Completed).

21.7.47 Node i Interrupt routing for Group 2

Ni_ G2INTR has the same functionality as Ni_ G0INTR, but for other interrupt sources. The interrupt sources need to be enabled to be mapped.

Ni_G2INTR (i=0-3) Offset address: $00134_H+i*400_H$

Node i Interrupt routing for Group 2 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								CEI				IDMU			
r								rw				rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXHBUF1				TXHBUF0				RXHBUF1				RXHBUF0			
rw				rw				rw				rw			

Field	Bits	Type	Description
RXHBUF0	3:0	rw	**CRE Rx host buffer 0 interrupt** are mapped here: Ni_CRE_IR.RBUF0I (Receive Host Buffer 0 interrupt)
RXHBUF1	7:4	rw	**CRE Rx host buffer 1 interrupt** are mapped here: Ni_CRE_IR.RBUF0I (Receive Host Buffer 1 interrupt)
TXHBUF0	11:8	rw	**CRE Tx host buffer interrupt** are mapped here: Ni_CRE_IR.TBUFI 0 (Transmit Host Buffer 0 interrupt)
TXHBUF1	15:12	rw	**CRE Tx host buffer interrupt** are mapped here: Ni_CRE_IR.TBUFI 1 (Transmit Host Buffer 1 interrupt)
IDMU	19:16	rw	**IDMU relevant interrupts** are mapped here: Ni_CRE_IR.SFRMLI, Ni_CRE_IR.XFRMLI (Frame rate measure lost interrupts)
CEI	23:20	rw	**CRE error interrupts** are mapped here: Ni_CRE_IR.IRSI, Ni_CRE_IR.IWSI, Ni_CRE_IR.CRCI, Ni_CRE_IR.RWDTI, Ni_CRE_IR.TWDTI (Error interrupts)
0	31:24	r	**Reserved** Read as 0; should be written with 0.

21.7.48 Node i Timer Clock Control Register

The Node i Timer Clock Control Register Ni_TIMER_CCR controls the functions of the node timer.

Ni_TIMER_CCR (i=0-3) Offset address: $00138_H+i*400_H$

Node i Timer Clock Control Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0											TRIGSRC			0	
r											rw			r	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STSTART	STRESET	0		TPSC				0							
rw	rw	r		rw				r							

Field	Bits	Type	Description
TPSC	11:8	rw	**Timer Prescaler** The duration of one timer clock is given by Ni_TIMER_CCR.TRIGSRC frequency divided by (TPSC + 1).
STRESET	14	rw	**Stamping Reset** This bit gives the possibility to reset the time stamp for CAN FD messages.
STSTART	15	rw	**Stamping Start** This bit starts the external timer used for CAN FD messages. The source and the prescaler are identical to the Ni_TIMER_CCR(i=0-3).
TRIGSRC	20:18	rw	**Trigger Source** This bit selects the trigger source for the different modes in the node timer. 000_B Node i Timer is decremented per f_{SYNi} prescaled by (TPSC + 1) timing to 0. 001_B System Timer (STM) trigger event enabled Node i Timer is decremented per STM trigger event prescaled by (TPSC + 1). 010_B General Timer (GTM) trigger event enabled Node i Timer is decremented per GTM trigger event prescaled by (TPSC + 1). 011_B Enhanced General Timer (eGTM) trigger event enabled Node i Timer is decremented per eGTM trigger event prescaled by (TPSC + 1). **others,** Reserved, do not use
0	7:0, 13:12, 17:16, 31:21	r	**Reserved** Read as 0; should be written with 0.

21.7.49 Node i Timer Transmit Trigger 0 Register

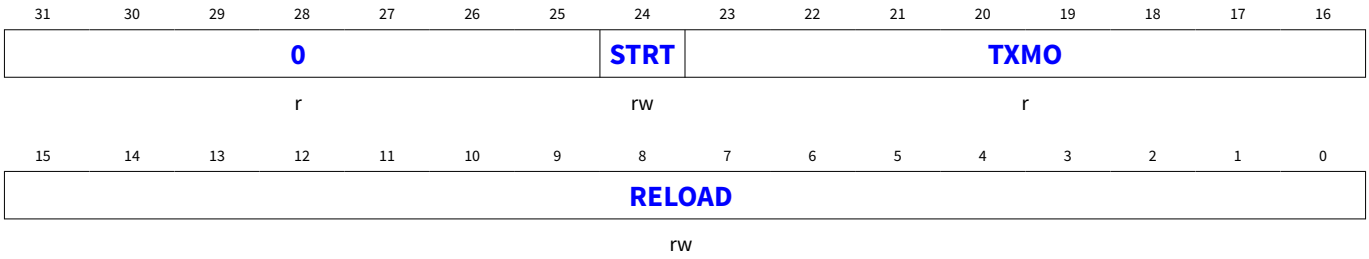
The Node i Timer Transmit Trigger 0 Register Ni_TIMER_TXTRIG0 controls the node timing functions for Transmit Trigger Mode.

Pretended Networking

The timer registers are intended to support Pretended Networking. As an application example, the SPB bus is clocked at 40MHz. The asynchronous module part is clocked either with 40MHz as well or even more power saving with direct drive from the oscillator. The cores are in idle mode. Messages can be received and a receive interrupt can be generated. It is possible to trigger messages with or without changing the content by the timers provided. For example the network management message and two related messages can be triggered without any CPU interaction. As mostly the operating system is still running, messages can be changed without any problem during the time, where the operating system is active.

Ni_TIMER_TXTRIG0 (i=0-3) Offset address: $0013C_H+i*400_H$

Node i Timer Transmit Trigger 0 Register Kernel Reset value: $0001\ 0000_H$



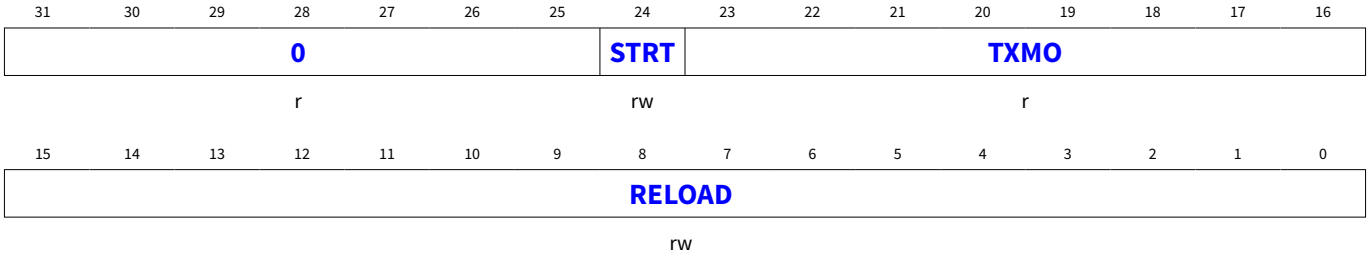
Field	Bits	Type	Description
RELOAD	15:0	rw	**Reload Value** This bit-field contains the reload value for the timer. The timer will restart when RELOAD is written.
TXMO	23:16	r	**Transmit Message Object** This transmit trigger is fixed to transmit buffer 1
STRT	24	rw	**Timer Start** This bit-field controls the operation of the timer. 0_B Timer is stopped. 1_B Timer is started.
0	31:25	r	**Reserved** Read as 0; should be written with 0.

21.7.50 Node i Timer Transmit Trigger 1 Register

The Node i Timer Transmit Trigger 1 Register Ni_TIMER_TXTRIG1 controls the node timing functions for Transmit Trigger Mode.

Ni_TIMER_TXTRIG1 (i=0-3) Offset address: $00140_H+i*400_H$

Node i Timer Transmit Trigger 1 Register Kernel Reset value: $0002\ 0000_H$



Field	Bits	Type	Description
RELOAD	15:0	rw	**Reload Value** This bit-field contains the reload value for the timer. The timer will restart when RELOAD is written.
TXMO	23:16	r	**Transmit Message Object** This transmit object is fixed to transmit buffer 2

(table continues...)

(continued)

Field	Bits	Type	Description
STRT	24	rw	**Timer Start** This bit-field controls the operation of the timer. 0_B Timer is stopped. 1_B Timer is started.
0	31:25	r	**Reserved** Read as 0; should be written with 0.

21.7.51 Node i Timer Transmit Trigger 2 Register

The Node i Timer Transmit Trigger 2 Register Ni_TIMER_TXTRIG2 controls the node timing functions for Transmit Trigger Mode.

Ni_TIMER_TXTRIG2 (i=0-3) Offset address: $00144_H+i*400_H$

Node i Timer Transmit Trigger 2 Register Kernel Reset value: $0003\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0							STRT	TXMO							
r							rw	r							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RELOAD															
rw															

Field	Bits	Type	Description
RELOAD	15:0	rw	**Reload Value** This bit-field contains the reload value for the timer. The timer will restart when RELOAD is written.
TXMO	23:16	r	**Transmit Message Object** This transmit trigger is fixed to transmit buffer 3
STRT	24	rw	**Timer Start** This bit-field controls the operation of the timer. 0_B Timer is stopped. 1_B Timer is started.
0	31:25	r	**Reserved** Read as 0; should be written with 0.

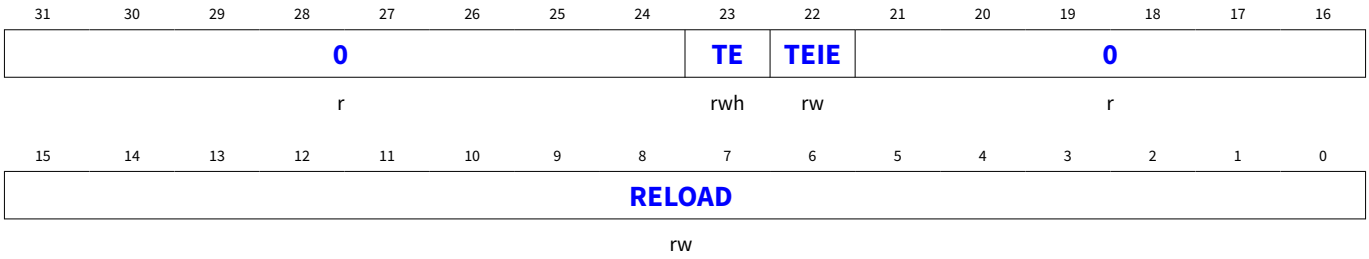
21.7.52 Node i Timer Receive Timeout Register

The Node i Timer Receive Timeout Register Ni_TIMER_RXTOUT controls the node timing functions for Receive Timeout Mode. This feature is independent of Classical CAN and CAN FD.

This mode exists, to have for example network management supervision.

Ni_TIMER_RXTOUT (i=0-3) Offset address: $00148_H+i*400_H$

Node i Timer Receive Timeout Register Kernel Reset value: $0000\ 0000_H$



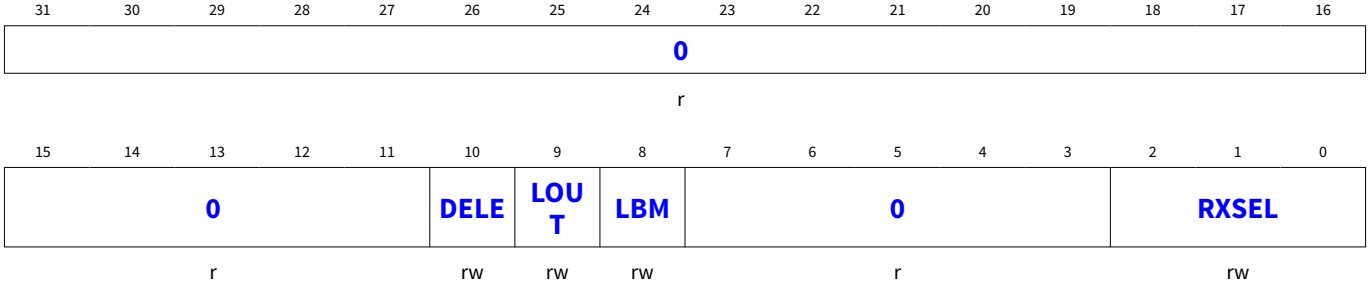
Field	Bits	Type	Description
RELOAD	15:0	rw	**Reload Value** This bit-field contains the reload value for the timer. The timer will start when RELOAD ≠ 0 is written. After half the time of the RELOAD value, the interrupt flags of the receive buffers will be cleared automatically, to ensure, that no message receive will be missed.
TEIE	22	rw	**Timer Event Interrupt Enable** This bit enables the node timer event interrupt of CAN node i. Bit-field Ni_G1INTR.RETI selects the interrupt output line which becomes activated at this type of interrupt. 0_B Timer event interrupt is disabled 1_B Timer event interrupt is enabled
TE	23	rwh	**Timer Event** This flag is set on a node timer transition from 1 to 0 in Receive Timeout Mode. This bit must be reset (i.e Write to '0') by software, writing a '1' has no effect. An interrupt request is generated if TEIE = 1. 0_B No timer event has occurred since last flag reset 1_B Timer event has occurred since last flag reset
0	21:16, 31:24	r	**Reserved** Read as 0; should be written with 0.

21.7.53 Node i Port Control Register

The Node Port Control Register Ni_PORTCTRL configures the CAN bus transmit/receive ports.

Ni_PORTCTRL (i=0-3) Offset address: $0014C_H+i*400_H$

Node i Port Control Register Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
RXSEL	2:0	rw	**Receive Select** RXSEL selects one out of 8 possible receive inputs. The CAN receive signal is performed by the selected input.
LBM	8	rw	**Loop-Back Mode** 0_B Loop-Back Mode is disabled. 1_B Loop-Back Mode is enabled. This node is connected to an internal (virtual) loop-back CAN bus. All CAN nodes which are in Loop-Back Mode are connected to this virtual CAN bus so that they can communicate with each other internally. The external transmit line is forced recessive in Loop-Back Mode.
LOUT	9	rw	**Loop Back Mode Out** The loop back bus is switched to the external CAN bus of the node.
DELE	10	rw	**Enable destructive read on Ni_ECR.CEL** If this bit is set, the destructive read on Ni_ECR.CEL and on the PSR register takes place. Meaning, that with read access on Ni_ECR, the CEL is reset. The same is true for the Ni_PSR register, for the bits PXE, RFDF, RBRS, RESI, LEC and DLEC. When this bitfield is set, reading the timestamp value from Ni_TSU_TSn register resets the related Ni_TSU_TSS1 bits. After the destructive read it is advised to reset the bit again.
0	7:3, 31:11	r	**Reserved** Read as 0; should be written with 0.

21.7.54 Node i CRE Configuration Register

The Node i CRE Configuration Register Ni_CRE_CONFIG configures the Routing Engine functionality.

Ni_CRE_CONFIG (i=0) Offset address: $00150_H+i*400_H$

Node i CRE Configuration Register Kernel Reset value: $0000\ 0100_H$

Ni_CRE_CONFIG (i=1) Offset address: $00150_H+i*400_H$

Node i CRE Configuration Register Kernel Reset value: $0000\ 0200_H$

Ni_CRE_CONFIG (i=2) Offset address: $00150_H+i*400_H$

Node i CRE Configuration Register Kernel Reset value: $0000\ 0300_H$

Ni_CRE_CONFIG (i=3) Offset address: $00150_H+i*400_H$

Node i CRE Configuration Register Kernel Reset value: $0000\ 0400_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		ID						0				DEN	IDM UEN	REN	EN
r		r						r				rw	rw	rw	rw

Field	Bits	Type	Description
EN	0	rw	**Enable** Note: When the EN is set, the respective host buffers are enabled. The RxFIFO0/1 and Tx FIFO/Queues shall not be used in this case. The frames are read/written from/to CRE host buffers instead. 0_B **DIS**: CRE function is disabled 1_B **EN**: CRE function is enabled. When CRE is enabled, then RxFIFO0/1 and TxFIFO/Queue host interface handling is also enabled when corresponding FIFO/Queue are enabled by CAN node
REN	1	rw	**Routing Enable** This bit indicates that the Routing is enabled for the Source node. This bit is checked only at the Source node. This bit-field can be written only when Ni_CRE_CONFIG.EN = 1. Note: This bit should be enabled at the source node for correct routing 0_B **DIS**: Routing Disable The Routing function of CRE is disabled. Note: If Ni_CRE_CFG.EN is set, the corresponding host buffers are still enabled. 1_B **EN**: Routing Enable The Routing function of CRE is enabled for the Source CAN node
IDMUEN	2	rw	**IDMU Enable** This bit-field can be written only when Ni_CRE_CFG.EN = 1 0_B **DIS**: IDMU Disable The Intrusion detection measurement unit (IDMU) of CRE is disabled 1_B **EN**: IDMU Enable The Intrusion detection measurement unit (IDMU) of CRE is enabled
DEN	3	rw	**Destination Enable** This bit is checked only at the Destination node. This bit controls the CRE write of the CAN frame to TxFIFO Note: In case of external routing, the CAN frame remains in the Tx Host Buffer and there will be no re-trigger to the DRE if DEN is set to 0 after there has already been a trigger to the DRE. In case of internal routing, the CAN frame would be discarded if DEN is set to 0 0_B **DIS**: Destination Disable The CAN frame is discarded 1_B **EN**: Destination Enable The CRE will write the CAN frame to the TxFIFO
ID	13:8	r	**Unique ID** A Unique ID of the CAN Node i 1_H till 4_H - MCMCAN 0 CAN Node 0-Node 3 respectively 5_H till 8_H - MCMCAN 1 CAN Node 0-Node 3 respectively 9_H till C_H - MCMCAN 2 CAN Node 0-Node 3 respectively D_H till 10_H - MCMCAN 3 CAN Node 0-Node 3 respectively 11_H till 14_H - MCMCAN 4 CAN Node 0-Node 3 respectively

(table continues...)

(continued)

Field	Bits	Type	Description
0	7:4, 31:14	r	**Reserved** Read as 0; shall be written with 0.

21.7.55 Node i CRE Configuration Start Address

This register defines the CRE RAM start address of CRE Configurations

Ni_CRE_CONFIGADR (i=0-3) Offset address: $00154_H+i*400_H$

Node i CRE Configuration Start Address Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SA														0	
rw														r	

Field	Bits	Type	Description
SA	15:2	rw	**Start Address** The start address from which the CRE Configuration section starts (32 bit word address and 16-byte aligned). The Routing Headers and Host buffers are allocated from this address.
0	1:0, 31:16	r	**Reserved** Read as 0; should be written with 0.

21.7.56 Node i Receive Host Buffer z Configuration

This register configures the Host Buffer for RxFIFO

Ni_CRE_HBUF_RXz_CONFIG (i=0-3;z=0-1) Offset address: $00158_H+i*400_H+z*8$

Node i Receive Host Buffer z Configuration Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0					CRCEN	INTEN	TRIGEN	0	LEN						LRM
r					rw	rw	rw	r	rw						rw

Field	Bits	Type	Description
LRM	0	rw	**Last Read Mode** This bit-field defines the last word of the CAN payload data expected to be read by the host from Rx Host Buffer 0_B Dynamic Data Length: The last word to be read by the host is defined by the DLC of the received CAN frame 1_B Fixed Data Length The last word to be read by the host is defined by LEN bit-field of this register
LEN	6:1	rw	**Fixed Data Length** This bit-field contains the length of CAN Data field in words (4 bytes) which is stored in the Receive Host Buffer. It defines the last word to be read by the host when LRM=1. This bit-field is ignored when LRM=0. 0 - No payload data 1 - Data length of 4 bytes or 1 word 2 - Data length of 8 bytes or 2 words . . 16 - Data length of 64 bytes or 16 words others - Incorrect (shall not be more than 16)
TRIGEN	8	rw	**Enable Trigger** This bit-field enables / disables the generation of Trigger signal when a new CAN frame is stored in Receive Host Buffer. 0_B Disable The Receive Host Buffer z trigger signal is disabled 1_B Enabled The Receive Host Buffer z trigger signal is enabled
INTEN	9	rw	**Interrupt Enable** This bit-field enables / disables the generation of interrupt when a new CAN frame is stored in Receive Host Buffer. 0_B Disable The Receive Host Buffer z interrupt is disabled 1_B Enabled The Receive Host Buffer z interrupt is enabled
CRCEN	10	rw	**CRC calculation enable** This bit controls inclusion of CRC in the sequence check 0_B CRC not included 1_B CRC included
0	7, 31:11	r	**Reserved** Read as 0; should be written with 0.

21.7.57 Node i Receive Host Buffer z Status

This register provides the status information of the Receive Host buffer.

Ni_CRE_HBUF_RXz_STAT (i=0-3;z=0-1) Offset address: $0015C_H+i*400_H+z*8$

Node i Receive Host Buffer z Status Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								COUNT							
r								rwh							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WEG		INDEX						0		SWTRIG	VTH	RHREQ	VRH	0	
rwh		rh						r		rwh	rh	rwh	rh	r	

Field	Bits	Type	Description
VRH	2	rh	**Valid Routing Header** This bit-field indicates if there is a valid routing header stored in Routing Header for the current CAN frame. 0 - No routing header (routing non-relevant) 1 - Valid routing header
RHREQ	3	rwh	**New Receive Host Buffer z Transfer Request** This bit-field indicates a pending transfer request of Receive Host buffer. It is set to 1 by hardware, when there is a new CAN frame stored in the Receive Host Buffer. After successful read operation of the host from the RX Host Buffer, this bit is reset to 0 by hardware. SW write to 1 also clears the bit. A SW write by 0 has no effect. A transition of 0 to 1 triggers RHBUF_INT when enabled in Ni_CRE_HBUF_RXz_CONFIG.INTEN=1.
VTH	4	rh	**Valid Timing Header** This bit-field indicates if there is a valid timing header stored in Timing Header for the current CAN frame. 0 - No Timing header (Intrusion parameters not available) 1 - Valid Timing header
SWTRIG	5	rwh	**Software DRE Tigger** SW trigger to the DRE. SW writes 1 to request trigger to DRE. HW clears it after trigger to DRE. SW write with 0 has no effect
INDEX	13:8	rh	**RxFIFOz Index** This bit-field indicates the Get Index of RxFIFOz, from which the current CAN frame is fetched form.

(table continues...)

(continued)

Field	Bits	Type	Description
WEG	15:14	rwh	**Watchdog Event Group** Set by hardware when there is a watchdog timeout error. This bit-field indicates which event group has watchdog timeout error. Write operation of any non-zero value resets the value to 0. Writing 0 does not change the value of the bit-field 00_B No Error 01_B WDG1 Error 10_B WDG2 Error 11_B WDG3 Error
COUNT	23:16	rwh	**Counter** The count field increments by 1 for every successful read of a CAN frame by host from the Receive Host Buffer. The Counter wraps around to 0 by hardware upon overflow. A write operation of any non zero value resets the counters value to 0. Writing of 0 does not change the value
0	1:0, 7:6, 31:24	r	**Reserved** Reserved. Read as 0; should be written with 0

21.7.58 Node i Transmit Host Buffer z Configuration

This register configures the Host Buffer for TxFIFO / TxQueue

Ni_CRE_HBUF_TXz_CONFIG (i=0-3;z=0-1) Offset address: $00168_H+i*400_H+z*8$

Node i Transmit Host Buffer z Configuration Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				CRCG	CRCEN	INTEN	0		LEN						LWM
r				rw	rw	rw	r		rw						rw

Field	Bits	Type	Description
LWM	0	rw	**Last Write Mode** This bit-field defines the last word of the CAN payload data expected to be written by the host to Tx Host Buffer 0_B Dynamic Data Length: The last word to be read by the host is defined by the DLC of the transmit CAN frame 1_B Fixed Data Length The last word to be written by the host is defined by Ni_CRE_HBUF_TXz_CONFIG.LEN. Ni_CRE_HBUF_TXz_CONFIG.LEN shall be configured greater than 0

(table continues...)

(continued)

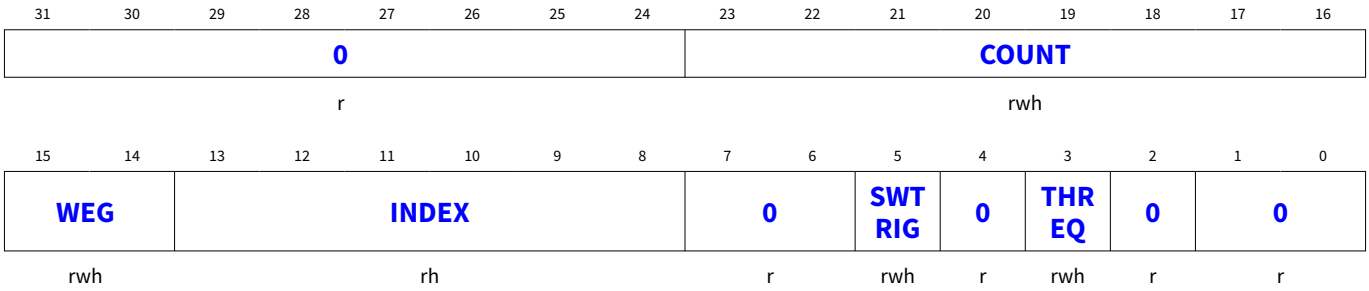
Field	Bits	Type	Description
LEN	6:1	rw	**Fixed Data Length** This bit-field contains the length of CAN Data field in words (4 bytes) which is stored in the Transmit Host Buffer. It defines the last word to be written by the host when LWM=1. This bit-field is ignored when LWM=0. 0 - No payload data 1 - Data length of 4 bytes or 1 word 2 - Data length of 8 bytes or 2 words . . 16 - Data length of 64 bytes or 16 words others - Incorrect (shall not be more than 16)
INTEN	9	rw	**Interrupt or Trigger Enable** This bit-field enables the generation of trigger signal to DRE/interrupt to IR when the Transmit Host Buffer is available to transmit a new CAN frame. Only Tx Host Buffer 0 shall be used to trigger DRE. In case the Tx Host Buffer 1 INTEN is set to 0, there is no trigger to DRE nor an interrupt Note: The interrupt is triggered only when the Tx Host Buffer is emptied after being filled. 0_B TRIG Enabled The Transmit Host Buffer trigger signal is enabled 1_B Interrupt Enabled The Transmit Host Buffer interrupt is enabled
CRCEN	10	rw	**CRC calculation enable** This bit controls enable and disable of CRC calculation when INTEN is 1 CRC calculation is always enabled when INTEN is 0 0_B CRC calculation disabled 1_B CRC calculation enabled
CRCG	11	rw	**CRC gate** This bit controls whether the CAN frame with CRC error is processed further or discarded by the CRE 0_B CAN frame is not discarded 1_B CAN frame discarded
0	8:7, 31:12	r	**Reserved** Read as 0; should be written with 0.

21.7.59 Node i Transmit Host Buffer z Status

This register provides the status information of the Transmit Host buffer.

Ni_CRE_HBUF_TXz_STAT (i=0-3;z=0-1) Offset address: $0016C_H+i*400_H+z*8$

Node i Transmit Host Buffer z Status Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
THREQ	3	rwh	**Free Transmit Host Buffer z** This bit-field indicates a pending transfer request of Transmit Host Buffer. It is set to 0 by hardware, when the Transmit Host Buffer is free to accept a new transmit CAN frame. Upon "last word" write completion event to the Tx Host Buffer the HW sets this bit to 1 indicating buffer full. SW write to 1 also clears the bit. A SW write by 0 has no effect. An interrupt TXHBUF_INT is triggered when enabled in Ni_CRE_HBUF_TXz_CONFIG.INTEN=1 to indicate free buffer element..
SWTRIG	5	rwh	**Software DRE Tigger** SW trigger to the DRE. SW writes 1 to request trigger to DRE. HW clears it after trigger to DRE. SW write with 0 has no effect. Note: Write to this bit is valid only for Tx Host buffer 0. SW write has no effect for Tx Host buffer 1
INDEX	13:8	rh	**TxFIFO/Queue Index** This bit-field indicates the Put Index of TxFIFO / Queue, to which the current CAN frame will be written to.
WEG	15:14	rwh	**Watchdog Event Group** Set by hardware when there is a watchdog timeout error. This bit-field indicates which event group has watchdog timeout error. Write operation of any non-zero value resets the value to 0. Writing 0 does not change the value of the bit-field 00_B No Error 01_B WDG1 Error 10_B WDG2 Error 11_B WDG3 Error
COUNT	23:16	rwh	**Counter** The count field increments by 1 for every successful write of a CAN frame by host to the Transmit Host Buffer. The Counter wraps around to 0 by hardware upon overflow. A write operation of any non zero value resets the counters value to 0. Writing of 0 does not change the value
0	1:0, 2, 4, 7:6, 31:24	r	**Reserved** Reserved. Read as 0; should be written with 0

21.7.60 Node i CRE Interrupt Register

The flags are set when one of the listed conditions is detected (edge-sensitive). The flags remain set until the Host clears them. A flag is cleared by writing a 1 to the corresponding bit position. Writing a 0 has no effect.

Ni_CRE_IR (i=0-3) Offset address: $0018C_H+i*400_H$

Node i CRE Interrupt Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TWDTI1	TWDTI0	RWDTI1	RWDTI0	CRCI1	CRCI0	IWSI1	IWSI0	IRSI1	IRSI0	XFRMLI	SFRMLI	TBUF1I	TBUF0I	RBUF1I	RBUF0I
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
RBUF0I	0	rwh	**Node i Rx Host Buffer 0 interrupt flag** This flag is set after the Receive Interface writes the complete frame to the Receive Host Buffer 0. The Receive Host Buffer interrupt or trigger can be used by a master of FPI to read the CAN frame from the Receive Host Buffer 0.
RBUF1I	1	rwh	**Node i Rx Host Buffer 1 interrupt flag** This flag is set after the Receive Interface writes the complete frame to the Receive Host Buffer 1. The Receive Host Buffer interrupt or trigger can be used by a master of FPI to read the CAN frame from the Receive Host Buffer 1.
TBUF0I	2	rwh	**Node i Tx Host Buffer 0 interrupt flag** This flag is set after the Transmit Interface identifies a free Transmit Host Buffer 0 element. The Transmit Host Buffer interrupt or trigger can be used by a master of FPI to write the CAN frame to the Transmit Host Buffer 0.
TBUF1I	3	rwh	**Node i Tx Host Buffer 1 interrupt flag** This flag is set after the Transmit Interface identifies a free Transmit Host Buffer 1 element. The Transmit Host Buffer interrupt or trigger can be used by a master of FPI to write the CAN frame to the Transmit Host Buffer 1.
SFRMLI	4	rwh	**Node i STD ID Frame measure lost interrupt flag** This flag is set when there is loss of standard ID frame measure count while the STDLOCK is set by the user in the Ni_IDMU_FRTCONFIG SFR
XFRMLI	5	rwh	**Node i XTD ID Frame measure lost interrupt flag** This flag is set when there is loss of extended ID frame measure count while the XTDLOCK is set by the user in the Ni_IDMU_FRTCONFIG SFR
IRSI0	6	rwh	**Node i Incorrect read sequence error interrupt 0** This flag is set when there is an incorrect read sequence error interrupt triggered for Rx Host Buffer 0

(table continues...)

(continued)

Field	Bits	Type	Description
IRSI1	7	rwh	**Node i Incorrect read sequence error interrupt 1** This flag is set when there is an incorrect read sequence error interrupt triggered for Rx Host Buffer 1
IWSI0	8	rwh	**Node i Incorrect write sequence error interrupt 0** This flag is set when an incorrect write sequence error interrupt is triggered for Tx Host Buffer 0
IWSI1	9	rwh	**Node i Incorrect write sequence error interrupt 1** This flag is set when an incorrect write sequence error interrupt is triggered for Tx Host Buffer 1
CRCI0	10	rwh	**Node i CRC error interrupt 0** This flag is set when CRC error interrupt is triggered for Tx Host buffer 0
CRCI1	11	rwh	**Node i CRC error interrupt 1** This flag is set when CRC error interrupt is triggered for Tx Host buffer 1
RWDTI0	12	rwh	**Node i RHBUF0 watchdog timeout error interrupt 0** This flag is set when RHBUF0 watchdog timeout error interrupt is triggered
RWDTI1	13	rwh	**Node i RHBUF1 watchdog timeout error interrupt 1** This flag is set when RHBUF1 watchdog timeout error interrupt is triggered
TWDTI0	14	rwh	**Node i THBUF0 watchdog timeout error interrupt 0** This flag is set when THBUF0 watchdog timeout error interrupt is triggered
TWDTI1	15	rwh	**Node i THBUF1 watchdog timeout error interrupt 1** This flag is set when THBUF1 watchdog timeout error interrupt is triggered
0	31:16	r	**Reserved** Read as 0; should be written with 0.

21.7.61 Node i Frame Rate Measure Table Configuration

Configuration registers for IDMU

This register controls the Frame Rate Measure Tables

Ni_IDMU_FRTCONFIG (i=0-3) Offset address: $00190_H+i*400_H$

Node i Frame Rate Measure Table Configuration Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0					INTEN1	0	XTDLOCK	0					INTEN0	0	STDLOCK
r					rw	r	rwh	r					rw	r	rwh

Field	Bits	Type	Description
STDLOCK	0	rwh	**Lock STD ID Frame Rate Measures** Lock bit to freeze the Standard ID Frame Rate Measures. This bit is set by the user (eg. in a timer-based interrupt service routine) to read the STD ID frame rate measure counters
INTEN0	2	rw	**STD ID Frame rate measure Interrupt Enable** This bit-field enables / disables the generation of interrupt when a frame is received while the Frame Rate Measure counters (FRMs) are frozen i.e when the STDLOCK is set and there is a loss of frame count. 0_B Disable The STD ID Frame rate measure interrupt is disabled 1_B Enabled The STD ID Frame rate measure interrupt is enabled
XTDLOCK	8	rwh	**Lock XTD ID Frame Rate Measures** Lock bit to freeze the Extended ID Frame Rate Measures. This bit is set by the user (eg. in a timer-based interrupt service routine) to read the XTD ID frame rate measure counters
INTEN1	10	rw	**XTD ID Frame rate measure Interrupt Enable** This bit-field enables / disables the generation of interrupt when a frame is received while the Frame Rate Measure counters (FRMs) are frozen i.e when the XTDLOCK is set and there is a loss of frame count. 0_B Disable The XTD ID Frame rate measure interrupt is disabled 1_B Enabled The XTD ID Frame rate measure interrupt is enabled
0	1, 7:3, 9, 31:11	r	**Reserved** Read as 0; should be written with 0

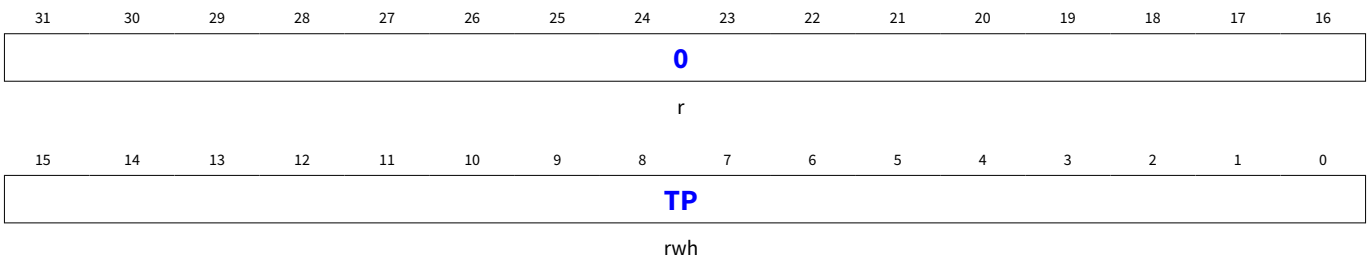
21.7.62 Node i Rx Throughput Measure configuration

Configuration registers for IDMU

This register provides the Rx Throughput Measured over a set of frames of the MCAN and its corresponding control bits

Ni_IDMU_RXTPCFG (i=0-3) Offset address: $00194_H+i*400_H$

Node i Rx Throughput Measure configuration Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
TP	15:0	rwh	**Rx Throughput Measure** Rx Throughput Measure is relevant for Intrusion detection. It indicates the total number of frames received by the MCMCAN.
0	31:16	r	**Reserved** Read as 0; should be written with 0.

21.7.63 Node i CRE Error control register

The CRE error control register is used to enable particular monitoring group of start and end conditions and control the interrupt enable.

Ni_ERRCTRL (i=0-3) Offset address: $00198_H+i*400_H$

Node i CRE Error control register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0						TWD1IE	TWD0IE	IWSIE	IRSIE	CRCIE	RWD1IE	RWD0IE	WDG3	WDG2	WDG1
r						rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
WDG1	0	rwh	**Enable bit for watchdog timeout monitoring event group 1** When this bit is set, the following start and end conditions are monitored. These events are compelety under the control of CRE **1.** • Start condition : RxFIFO not empty • End condition : RxFIFO processing done **2.** • Start condition : THBUF full • End condition : THBUF empty
WDG2	1	rwh	**Enable bit for watchdog timeout monitoring event group 2** When this bit is set, the following start and end conditions are monitored. These events are dependent on the System, DRE and message frequency **1.** • Start condition : RHBUF empty • End condition : RHBUF full **2.** • Start condition : THBUF empty • End condition : THBUF full
WDG3	2	rwh	**Enable bit for watchdog timeout monitoring event group 3** When this bit is set, the following start and end conditions are monitored. This is dependent on DRE or System **1.** • Start condition : RHBUF full • End condition : RHBUF empty

(table continues...)

(continued)

Field	Bits	Type	Description
RWD0IE	3	rwh	**Receive watchdog timeout 0 interrupt enable** 0_B **DIS**: Disable RHBUF watchdog timeout error interrupt 1_B **EN**: Enable RHBUF watchdog timeout error interrupt
RWD1IE	4	rwh	**Receive watchdog timeout 1 interrupt enable** 0_B **DIS**: Disable RHBUF watchdog timeout error interrupt 1_B **EN**: Enable RHBUF watchdog timeout error interrupt
CRCIE	5	rwh	**CRC error interrupt enable** 0_B **DIS**: Disable CRC error interrupt 1_B **EN**: Enable CRC error interrupt
IRSIE	6	rwh	**Incorrect read sequence error interrupt enable** 0_B **DIS**: Disable invalid read sequence error interrupt 1_B **EN**: Enable invalid read sequence error interrupt
IWSIE	7	rwh	**Incorrect write sequence error interrupt enable** 0_B **DIS**: Disable invalid write sequence error interrupt 1_B **EN**: Enable invalid write sequence error interrupt
TWD0IE	8	rwh	**Transmit watchdog timeout 0 interrupt enable** 0_B **DIS**: Disable THBUF watchdog timeout error interrupt 1_B **EN**: Enable THBUF watchdog timeout error interrupt
TWD1IE	9	rwh	**Transmit watchdog timeout 1 interrupt enable** 0_B **DIS**: Disable THBUF watchdog timeout error interrupt 1_B **EN**: Enable THBUF watchdog timeout error interrupt
0	31:10	r	**Reserved**

21.7.64 Node i Core Release Register

Ni_CREL (i=0-3) Offset address: $00200_H+i*400_H$

Node i Core Release Register Kernel Reset value: $3308\ 1114_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
REL				STEP				SUBSTEP				YEAR			
r				r				r				r			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MON								DAY							
r								r							

Field	Bits	Type	Description
DAY	7:0	r	**Time Stamp Day**
MON	15:8	r	**Time Stamp Month**
YEAR	19:16	r	**Time Stamp Year**

(table continues...)

(continued)

Field	Bits	Type	Description
SUBSTEP	23:20	r	**Sub-step of Core Release** One digit, BCD-coded.
STEP	27:24	r	**Step of Core Release** One digit, BCD-coded.
REL	31:28	r	**Core Release** One digit, BCD-coded.

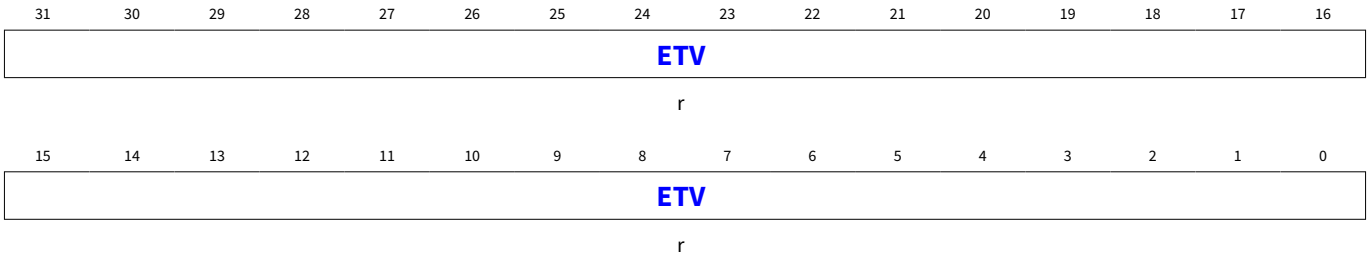
Table 1014 Example for Coding of Revisions

Release	Step	SubStep	Year	Month	Day	Name
0	1	0	0	03	10	Revision 0.1.0, Date 2010/03/10

21.7.65 Node i Endian Register

Ni_ENDN (i=0-3) Offset address: $00204_H+i*400_H$

Node i Endian Register Kernel Reset value: $8765\ 4321_H$



Field	Bits	Type	Description
ETV	31:0	r	**Endianness Test Value** The endianness test value is 0x87654321.

21.7.66 Node i Data Bit Timing & Prescaler Register

This register is only writable if bits Ni_CCCR.CCE and Ni_CCCR.INIT are set. The CAN bit time may be programmed in the range of 4 to 49 time quanta. The CAN time quantum may be programmed in the range of 1 to 32 clock cycles. t_q = (DBRP + 1) clock cycles.

DTSEG1 is the sum of Prop_Seg and Phase_Seg1. DTSEG2 is Phase_Seg2.

Therefore the length of the bit time is (programmed values) [DTSEG1 + DTSEG2 + 3] t_q or (functional values) [Sync_Seg + Prop_Seg + Phase_Seg1 + Phase_Seg2] t_q.

The Information Processing Time (IPT) is zero, meaning the data for the next bit is available at the first clock edge after the sample point.

***Note**: With a CAN clock of 8 MHz, the reset value of 0x00000A33 configures the M_CAN for a fast bit rate of 500 kbit/s.*

The bit rate configured for the CAN FD data phase via DBTP must be higher or equal to the bit rate configured for the arbitration phase via NBTP.

Ni_DBTP (i=0-3) Offset address: $0020C_H+i*400_H$

Node i Data Bit Timing & Prescaler Register Reset values see: Table 1015

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								TDC	0		DBRP				
r								rw	r		rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			DTSEG1					DTSEG2				DSJW			
r			rw					rw				rw			

Field	Bits	Type	Description
DSJW	3:0	rw	**Data (Re) Synchronization Jump Width** This bit-field is CCE and INIT protected. Writes will only have effect, if both bits are set. Valid values are 0 to 15. The actual interpretation by the hardware of this value is such that one more than the value programmed here is used.
DTSEG2	7:4	rw	**Data time segment after sample point** This bit-field is CCE and INIT protected. Writes will only have effect, if both bits are set. Valid values are 0 to 15. The actual interpretation by the hardware of this value is such that one more than the programmed value is used.
DTSEG1	12:8	rw	**Data time segment before sample point** This bit-field is CCE and INIT protected. Writes will only have effect, if both bits are set. Valid values are 0 to 31. The actual interpretation by the hardware of this value is such that one more than the programmed value is used.

(table continues...)

(continued)

Field	Bits	Type	Description
DBRP	20:16	rw	**Data Baud Rate Prescaler** This bit-field is CCE and INIT protected. Writes will only have effect, if both bits are set. The value by which the oscillator frequency is divided for generating the bit time quanta. The bit time is built up from a multiple of this quanta. Valid values for the Baud Rate Prescaler are 0 to 31. The actual interpretation by the hardware of this value is such that one more than the value programmed here is used.
TDC	23	rw	**Transmitter Delay Compensation** This bit-field is CCE and INIT protected. Writes will only have effect, if both bits are set. 0_B Transmitter Delay Compensation disabled 1_B Transmitter Delay Compensation enabled
0	15:13, 22:21, 31:24	r	**Reserved** Read as 0; should be written with 0.

Table 1015 Reset values of Ni_DBTP (i=0-3)

Reset type	Reset value	Note
Kernel Reset	0000 $0A33_H$	
After Boot-FW Value	0000 0000 –XX– ---- XXX– ---- ---- ----$_B$	After CAN BSL execution (applicable only for CAN0 and Node 1)

21.7.67 Node i Test Register

Write access to the Test Register has to be enabled by setting bit Ni_CCCR.TEST to ‘1’. All Test Register functions are set to their reset values when bit Ni_CCCR.TEST is reset.

Loop Back Mode and software control of transmit pin are hardware test modes. Programming of Ni_TEST.TX ≠ “00” may disturb the message transfer on the CAN bus.

Ni_TEST (i=0-3) Offset address: $00210_H+i*400_H$

Node i Test Register Reset values see: Table 1016

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RX	TX		LBCK	0	0	0	0
r								rh	rw		rw	r	r	r	r

Field	Bits	Type	Description
LBCK	4	rw	**Loop Back Mode** This bit-field is CCE and INIT protected. Writes will only have effect, if both bits are set. This is the external loop back mode, visible on the outside. 0_B Reset value, Loop Back Mode is disabled 1_B Loop Back Mode is enabled
TX	6:5	rw	**Control of Transmit Pin** This bit-field is CCE and INIT protected. Writes will only have effect, if both bits are set. 00_B Reset value, TX pin controlled by the CAN Core, updated at the end of the CAN bit time 01_B Sample Point can be monitored at the TX pin 10_B Dominant (‘0’) level at TX pin. 11_B Recessive (‘1’) at TX pin.
RX	7	rh	**Receive Pin** Monitors the actual value of RX pin. 0_B The CAN bus is dominant (RXD = ‘0’) 1_B The CAN bus is recessive (RXD = ‘1’)
0	0, 1, 2, 3, 31:8	r	**Reserved** Read as 0; should be written with 0.

Table 1016 Reset values of Ni_TEST (i=0-3)

Reset type	Reset value	Note
Kernel Reset	$0000\ 0000_H$	Value of RX depends on RX signal value.

21.7.68 Node i RAM Watchdog

The RAM Watchdog monitors the READY output of the Message RAM. A Message RAM access via the M_CAN’s Generic Master Interface starts the Message RAM Watchdog Counter with the value configured by Ni_RWD.WDC. The counter is reloaded with RWD.WDC when the Message RAM signals successful completion. In case there is no response from the Message RAM until the counter has counted down to zero, the counter stops and interrupt flag Ni_IR.WDI is set. The RAM Watchdog Counter is clocked by the Host clock.

Ni_RWD (i=0-3) Offset address: $00214_H+i*400_H$

Node i RAM Watchdog Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WDV								WDC							
rh								rw							

Field	Bits	Type	Description
WDC	7:0	rw	**Watchdog Configuration** This bit-field is CCE and INIT protected. Writes will only have effect, if both bits are set. Start value of the Message RAM Watchdog Counter. With the reset value of “00” the counter is disabled. 00_H Watchdog disabled **others,** Start value of the Message RAM Watchdog Counter
WDV	15:8	rh	**Watchdog Value** Actual Message RAM Watchdog Counter Value.
0	31:16	r	**Reserved** Read as 0; should be written with 0.

21.7.69 Node i CC Control Register

The Ni_CCCR register enables and disables CAN bus participation and basic protocol functions. Due to synchronization mechanisms between the clock domains, after a write operation to Ni_CCCR, the register shall be read back, until the set values are written to the register. Please keep in mind, that the register also includes hardware influenced bits.

***Note**: After enabling the CAN clocks in MCR register, the application software has to wait for 10 host clock cycles before accessing the kernel registers.*

***Note**: LDMST or SWAPMSK.W instructions should be used only with bit mask enabled for rwh bits in this register.*

Ni_CCCR (i=0-3) Offset address: $00218_H+i*400_H$

Node i CC Control Register Reset values see: Table 1017

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NISO	TXP	EFBI	PXHD	WMM	UTSU	BRSE	FDOE	TEST	DAR	MON	CSR	CSA	ASM	CCE	INIT
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rh	rwh	rw	rwh

Field	Bits	Type	Description
INIT	0	rwh	**Initialization** ***Note**: Due to the synchronization mechanism between the two clock domains, there may be a delay until the value written to INIT can be read back. Therefore the programmer has to assure that the previous value written to INIT has been accepted by reading INIT before setting INIT to a new value.* 0_B Normal Operation 1_B Initialization is started
CCE	1	rw	**Configuration Change Enable** 0_B The CPU has no write access to the protected configuration registers 1_B The CPU has write access to the protected configuration registers (while CCCR.INIT = '1')
ASM	2	rwh	**Restricted Operation Mode** Bit ASM can only be set by the Host when both CCE and INIT are set to '1'. In can also be set by the M_CAN. The bit can be reset by the Host at any time. For a description of the Restricted Operation Mode see paragraph Restricted Operation Mode. 0_B Normal CAN operation 1_B Restricted Operation Mode active
CSA	3	rh	**Clock Stop Acknowledge** 0_B No clock stop acknowledged 1_B M_CAN may be set in power down by stopping the synchronous and the asynchronous clock source
CSR	4	rw	**Clock Stop Request** 0_B No clock stop is requested 1_B Clock stop requested. When clock stop is requested, first INIT and then CSA will be set after all pending transfer requests have been completed and the CAN bus reached idle.
MON	5	rw	**Bus Monitoring Mode** Bit MON can only be set by the Host when both CCE and INIT are set to '1'. The bit can be reset by the Host at any time. 0_B Bus Monitoring Mode is disabled 1_B Bus Monitoring Mode is enabled
DAR	6	rw	**Disable Automatic Retransmission** This bit-field is CCE and INIT protected. Writes will only have effect, if both bits are set. 0_B Automatic retransmission of messages not transmitted successfully enabled 1_B Automatic retransmission disabled

(table continues...)

(continued)

Field	Bits	Type	Description
TEST	7	rw	**Test Mode Enable** The TEST register can only be set, if CCE, INIT and TEST are set. Writes to test will only have effect, if all three bits are set. 0_B Normal operation, register TEST holds reset values 1_B Test Mode, write access to register TEST enabled
FDOE	8	rw	**FD Operation Enable** This bit-field is CCE and INIT protected. Writes will only have effect, if both bits are set. 0_B CAN FD frame format disabled. 1_B CAN FD frame format enabled.
BRSE	9	rw	**Bit Rate Switch Enable** This bit-field is CCE and INIT protected. Writes will only have effect, if both bits are set. 0_B Bit rate switching for transmission disabled. 1_B Bit rate switching for transmission enabled.
UTSU	10	rw	**Use Timestamping register** When UTSU is set, 16-bit Wide Message Markers are also enabled regardless of the value of WMM. 0_B Internal time stamping 1_B External time stamping by TSU
WMM	11	rw	**Wide Message Marker** Enables the use of 16-bit Wide Message Markers. When 16-bit Wide Message Markers are used (WMM = '1'), 16-bit internal timestamping is disabled for the Tx Event FIFO. 0_B 8-bit Message Marker used 1_B 16-bit Message Marker used, replacing 16-bit timestamps in Tx Event FIFO
PXHD	12	rw	**Protocol Exception Handling Disable** This bit-field is CCE and INIT protected. Writes will only have effect, if both bits are set. 0_B Protocol exception handling enabled. 1_B Protocol exception handling disabled. (When protocol exception handling is disabled, the M_CAN will transmit an error frame when it detects a protocol exception condition.
EFBI	13	rw	**Edge Filtering during Bus Integration** This bit-field is CCE and INIT protected. Writes will only have effect, if both bits are set. 0_B Edge filter disabled 1_B Two consecutive dominant tq required to detect an edge for hard synchronization.

(table continues...)

(continued)

Field	Bits	Type	Description
TXP	14	rw	**Transmit Pause** This bit-field is CCE and INIT protected. Writes will only have effect, if both bits are set. If this bit is set, the M_CAN pauses for two CAN bit times before starting the next transmission after itself has successfully transmitted a frame. 0_B Transmit pause disabled 1_B Transmit pause enabled
NISO	15	rw	**Non ISO Operation** If this bit is set, the M_CAN uses the CAN FD frame format as specified by the Bosch CAN FD Specification V1.0. 0_B CAN FD frame format according to ISO11898-1 1_B CAN FD frame format according to Bosch CAN FD Specification V1.0
0	31:16	r	**Reserved** Read as 0; should be written with 0.

Table 1017 Reset values of Ni_CCCR (i=0-3)

Reset type	Reset value	Note
Kernel Reset	$0000\ 0001_H$	
After Boot-FW Value	$0000\ 0000\ 0000\ 0000\ 0000\ 00--\ 0000\ 0000_B$	After CAN BSL execution (applicable only for CAN0 and Node 1)

21.7.70 Node i Nominal Bit Timing & Prescaler Register

This register is only writable if bits Ni_CCCR.CCE and Ni_CCCR.INIT are set.

The CAN bit time may be programmed in the range of 4 to 385 time quanta. The CAN time quantum may be programmed in the range of 1 to 512 clock periods. t_q = (NBRP + 1) clock periods.

NTSEG1 is the sum of Prop_Seg and Phase_Seg1. NTSEG2 is Phase_Seg2.

Therefore the length of the bit time is (programmed values) [NTSEG1 + NTSEG2 + 3] t_q or (functional values) [Sync_Seg + Prop_Seg + Phase_Seg1 + Phase_Seg2] t_q.

The Information Processing Time (IPT) is zero, meaning the data for the next bit is available at the first clock edge after the sample point.

***Note**: With a CAN clock of 8 MHz, the reset value of 0600_H0A03 configures the M_CAN for a bit rate of 500 kbit/s.*

Ni_NBTP (i=0-3) Offset address: $0021C_H+i*400_H$

Node i Nominal Bit Timing & Prescaler Register Reset values see: Table 1018

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
NSJW							NBRP								
rw							rw								

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NTSEG1								0	NTSEG2						
rw								r	rw						

Field	Bits	Type	Description
NTSEG2	6:0	rw	**Nominal Time segment after sample point** This bit-field is CCE and INIT protected. Writes will only have effect, if both bits are set. Valid values are 1 to 127. The actual interpretation by the hardware of this value is such that one more than the programmed value is used.
NTSEG1	15:8	rw	**Nominal Time segment before sample point** This bit-field is CCE and INIT protected. Writes will only have effect, if both bits are set. Valid values are 1 to 255. The actual interpretation by the hardware of this value is such that one more than the programmed value is used.
NBRP	24:16	rw	**Baud Rate Prescaler** This bit-field is CCE and INIT protected. Writes will only have effect, if both bits are set. The value by which the oscillator frequency is divided for generating the bit time quanta. The bit time is built up from a multiple of this quanta. Valid values for the Baud Rate Prescaler are 0 to 511. The actual interpretation by the hardware of this value is such that one more than the value programmed here is used.
NSJW	31:25	rw	**(Re) Synchronization Jump Width** This bit-field is CCE and INIT protected. Writes will only have effect, if both bits are set. Valid values are 0 to 127. The actual interpretation by the hardware of this value is such that one more than the value programmed here is used.
0	7	r	**Reserved** Read as 0; should be written with 0.

Table 1018 Reset values of Ni_NBTP (i=0-3)

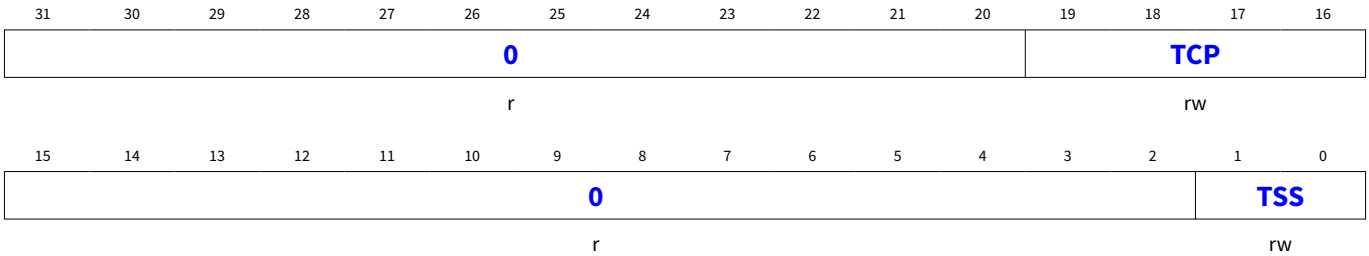
Reset type	Reset value	Note
Kernel Reset	0600 $0A03_H$	
After Boot-FW Value	---- ---- ---- ---- ---- ---- X--- ----$_B$	After CAN BSL execution (applicable only for CAN0 Node 1)

21.7.71 Node i Timestamp Counter Configuration

For a description of the Timestamp Counter see chapter Timestamp Generation

Ni_TSCC (i=0-3) Offset address: $00220_H+i*400_H$

Node i Timestamp Counter Configuration Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
TSS	1:0	rw	**Time segment before sample point** This bit-field is CCE and INIT protected. Writes will only have effect, if both bits are set. 01_B Timestamp counter value incremented according to TCP 10_B External timestamp counter value used, timer to be started in Ni_TIMER_CCR, the clock source as well as the chosen prescaler has to be configured before using this feature. **others,** Timestamp counter value always 0x0000
TCP	19:16	rw	**Timestamp Counter Prescaler** This bit-field is CCE and INIT protected. Writes will only have effect, if both bits are set. Configures the timestamp and timeout counters time unit in multiples of CAN bit times [1…16]. The actual interpretation by the hardware of this value is such that one more than the value programmed here is used. ***Note**: With CAN FD an external counter is required for timestamp generation (TSS = "10")*
0	15:2, 31:20	r	**Reserved** Read as 0; should be written with 0.

21.7.72 Node i Timestamp Counter Value

Ni_TSCV (i=0-3) Offset address: $00224_H+i*400_H$

Node i Timestamp Counter Value Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TSC															
rwh															

Field	Bits	Type	Description
TSC	15:0	rwh	**Timestamp Counter** The internal/external Timestamp Counter value is captured on start of frame (both Rx and Tx). When Ni_TSCC.TSS = “01”, the Timestamp Counter is incremented in multiples of CAN bit times [1…16] depending on the configuration of Ni_TSCC.TCP. A wrap around sets interrupt flag Ni_IR.TSW. Write access resets the counter to zero. When Ni_TSCC.TSS = “10”, TSC reflects the external Timestamp Counter value. A write access has no impact. ***Note**: A “wrap around” is a change of the Timestamp Counter value from non-zero to zero not caused by write access to Ni_TSCV.*
0	31:16	r	**Reserved** Read as 0; should be written with 0.

21.7.73 Node i Timeout Counter Configuration

The Timeout Counter register is used to configure signaling timeout conditions for Rx FIFO 0, Rx FIFO 1, and the Tx Event FIFO.

Ni_TOCC (i=0-3) Offset address: $00228_H+i*400_H$

Node i Timeout Counter Configuration Kernel Reset value: $FFFF\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TOP															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0													TOS		ETOC
r													rw		rw

Field	Bits	Type	Description
ETOC	0	rw	**Enable Timeout Counter** This bit-field is CCE and INIT protected. Writes will only have effect, if both bits are set. ***Note**: For use of timeout function with CAN FD see chapter Timeout Counter.* 0_B Timeout Counter disabled 1_B Timeout Counter enabled

(table continues...)

(continued)

Field	Bits	Type	Description
TOS	2:1	rw	**Timeout Select** This bit-field is CCE and INIT protected. Writes will only have effect, if both bits are set. When operating in Continuous mode, a write to Ni_TOCV presets the counter to the value configured by TOCC.TOP and continues down-counting. When the Timeout Counter is controlled by one of the FIFOs, an empty FIFO presets the counter to the value configured by Ni_TOCC.TOP. Down-counting is started when the first FIFO element is stored. 00_B Continuous operation 01_B Timeout controlled by Tx Event FIFO 10_B Timeout controlled by Rx FIFO 0 11_B Timeout controlled by Rx FIFO 1
TOP	31:16	rw	**Timeout Period** This bit-field is CCE and INIT protected. Writes will only have effect, if both bits are set. Start value of the Timeout Counter (down-counter). Configures the Timeout Period.
0	15:3	r	**Reserved** Read as 0; should be written with 0.

21.7.74 Node i Timeout Counter Value

Ni_TOCV (i=0-3) Offset address: $0022C_H+i*400_H$

Node i Timeout Counter Value Kernel Reset value: $0000\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

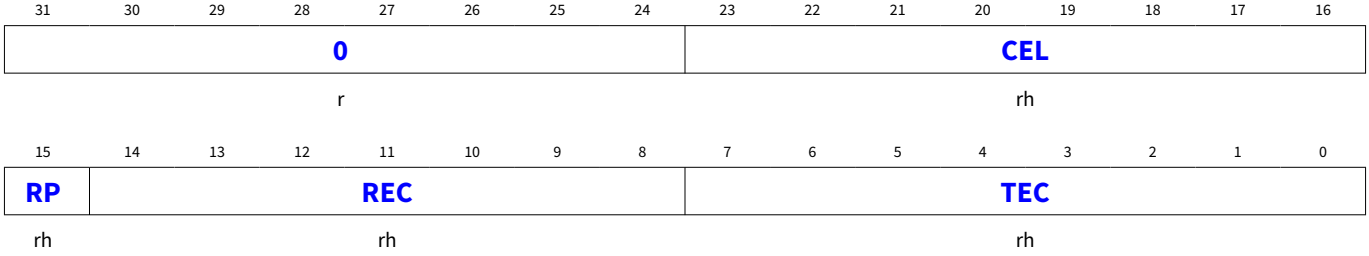
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TOC															
rwh															

Field	Bits	Type	Description
TOC	15:0	rwh	**Timeout Counter** The Timeout Counter is decremented in multiples of CAN bit times [1…16] depending on the configuration of Ni_TSCC.TCP. When decremented to zero, interrupt flag Ni_IR.TOO is set and the Timeout Counter is stopped. Start and reset/restart conditions are configured via Ni_TOCC.TOS. Any write access will lead to clearing of the counter.
0	31:16	r	**Reserved** Read as 0; should be written with 0.

21.7.75 Node i Error Counter Register

Ni_ECR (i=0-3) Offset address: $00240_H+i*400_H$

Node i Error Counter Register Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
TEC	7:0	rh	**Transmit Error Counter** Actual state of the Transmit Error Counter, values between 0 and 255 ***Note**: When CCCR.ASM is set, the CAN protocol controller does not increment TEC and REC when a CAN protocol error is detected, but CEL is still incremented.*
REC	14:8	rh	**Receive Error Counter** Actual state of the Receive Error Counter, values between 0 and 127 ***Note**: When CCCR.ASM is set, the CAN protocol controller does not increment TEC and REC when a CAN protocol error is detected, but CEL is still incremented.*
RP	15	rh	**Receive Error Passive** 0_B The Receive Error Counter is below the error passive level of 128 1_B The Receive Error Counter has reached the error passive level of 128
CEL	23:16	rh	**CAN Error Logging** The counter is incremented each time when a CAN protocol error causes the Transmit Error Counter or the Receive Error Counter to be incremented. It is reset by read access to CEL. The counter stops at 0xFF; the next increment of TEC or REC sets interrupt flag Ni_IR.ELO. The counter is reset on read, if the bit Ni_PORTCTRL.DELE is set for the node.
0	31:24	r	**Reserved** Read as 0; should be written with 0.

21.7.76 Node i Protocol Status Register

Ni_PSR (i=0-3) Offset address: $00244_H+i*400_H$

Node i Protocol Status Register Kernel Reset value: $0000\ 0707_H$

21 Controller Area Network interface (CAN)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0									TDCV						
r									rh						

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	PXE	RFDF	RBRS	RESI	DLEC			BO	EW	EP	ACT		LEC		
r	rh	rh	rh	rh	rh			rh	rh	rh	rh		rh		

Field	Bits	Type	Description
LEC	2:0	rh	**Last Error Code** The LEC indicates the type of the last error to occur on the CAN bus. This field will be cleared to '0' when a message has been transferred (reception or transmission) without error. This bit-field is set to 0x7 on read, if Ni_PORTCTRL.DELE is set. ***Note**: The Bus_Off recovery sequence (see ISO11898-1) cannot be shortened by setting or resetting Ni_CCCR.INIT. If the device goes Bus_Off, it will set Ni_CCCR.INIT of its own accord, stopping all bus activities. Once Ni_CCCR.INIT has been cleared by the CPU, the device will then wait for 129 occurrences of Bus Idle (129 * 11 consecutive recessive bits) before resuming normal operation. At the end of the Bus_Off recovery sequence, the Error Management Counters will be reset. During the waiting time after the resetting of Ni_CCCR.INIT, each time a sequence of 11 recessive bits has been monitored, a Bit0 Error code is written to Ni_PSR.LEC, enabling the CPU to readily check up whether the CAN bus is stuck at dominant or continuously disturbed and to monitor the Bus_Off recovery sequence. Ni_ECR.REC is used to count these sequences.* 000_B No Error: No error occurred since LEC has been reset by successful reception or transmission. 001_B Stuff Error: More than 5 equal bits in a sequence have occurred in a part of a received message where this is not allowed. 010_B Form Error: A fixed format part of a received frame has the wrong format. 011_B Ack Error: The message transmitted by the M_CAN was not acknowledged by another node. 100_B Bit1 Error: During the transmission of a message (with the exception of the arbitration field), the device wanted to send a recessive level (bit of logical value '1'), but the monitored bus value was dominant. 101_B Bit0 Error: During the transmission of a message (or acknowledge bit, or active error flag, or overload flag), the device wanted to send a dominant level (data or identifier bit logical value '0'), but the monitored bus value was recessive. During Bus_Off recovery this status is set each time a sequence of 11 recessive bits has been monitored. This enables the CPU to monitor the proceeding of the Bus_Off recovery sequence (indicating the bus is not stuck at dominant or continuously disturbed). 110_B CRC Error: The CRC check sum of a received message was incorrect. The CRC of an incoming message does not match with the CRC calculated from the received data. 111_B No Change: Any read access to the Protocol Status Register re-initializes the LEC to '7'. When the LEC shows the value '7', no CAN bus event was detected since the last CPU read access to the Protocol Status Register.

(table continues...)

(continued)

Field	Bits	Type	Description
ACT	4:3	rh	**Activity** Monitors the module's CAN communication state. ***Note:*** *ACT is set to "00" by a Protocol Exception Event.* 00_B Synchronizing - node is synchronizing on CAN communication 01_B Idle - node is neither receiver nor transmitter 10_B Receiver - node is operating as receiver 11_B Transmitter - node is operating as transmitter
EP	5	rh	**Error Passive** 0_B The M_CAN is in the Error_Active state. It normally takes part in bus communication and sends an active error flag when an error has been detected 1_B The M_CAN is in the Error_Passive state
EW	6	rh	**Warning Status** 0_B Both error counters are below the Error_Warning limit of 96 1_B At least one of error counter has reached the Error_Warning limit of 96
BO	7	rh	**Bus_Off Status** 0_B The M_CAN is not in Bus_Off[1)] 1_B The M_CAN is in Bus_Off state
DLEC	10:8	rh	**Data Phase Last Error Code** Type of last error that occurred in the data phase of a CAN FD format frame with its BRS flag set. Coding is the same as for LEC. This field will be cleared to zero when a CAN FD format frame with its BRS flag set has been transferred (reception or transmission) without error. This bit-field is set to 0x7 on read, if Ni_PORTCTRL.DELE is set. ***Note:*** *When a frame in CAN FD format has reached the data phase with BRS flag set, the next CAN event (error or valid frame) will be shown in DLEC instead of LEC. An error in a fixed stuff bit of a CAN FD CRC sequence will be shown as a Form Error, not Stuff Error.*
RESI	11	rh	**ESI flag of last received CAN FD Message** This bit is set together with REDF, independent of acceptance filtering. This bit is reset after read access, if Ni_PORTCTRL.DELE is set. 0_B Last received CAN FD message did not have its ESI flag set 1_B Last received CAN FD message had its ESI flag set
RBRS	12	rh	**BRS flag of last received CAN FD Message** This bit is set together with REDF, independent of acceptance filtering. This bit is reset after read access, if Ni_PORTCTRL.DELE is set. 0_B Last received CAN FD message did not have its BRS flag set 1_B Last received CAN FD message had its BRS flag set

(table continues...)

(continued)

Field	Bits	Type	Description
RFDF	13	rh	**Received a CAN FD Message** This bit is set independent of acceptance filtering. This bit is reset after read access, if Ni_PORTCTRL.DELE is set. 0_B Since this bit was reset by the CPU, no CAN FD message has been received 1_B Message in CAN FD format with FDF flag set, has been received
PXE	14	rh	**Protocol Exception Event** This bit is reset after read access, if Ni_PORTCTRL.DELE is set. 0_B No protocol exception event occurred since last read access 1_B Protocol exception event occurred
TDCV	22:16	rh	**Transmitter Delay Compensation Value** Position of the secondary sample point, defined by the sum of the measured delay from TX to RX and Ni_TDCR.TDCO. The SSP position is, in the data phase, the number of mtq between the start of the transmitted bit and the secondary sample point. Valid values are 0 to 127 mtq.
0	15, 31:23	r	**Reserved** Read as 0; should be written with 0.

1) The Bus_Off recovery sequence (see ISO11898-1) cannot be shortened by setting or resetting CCCR.INIT. If the device goes Bus_Off, it will set CCCR.INIT of its own accord, stopping all bus activities. Once CCCR.INIT has been cleared by the CPU, the device will then wait for 129 occurrences of Bus Idle (129 * 11 consecutive recessive bits) before resuming normal operation. At the end of the Bus_Off recovery sequence, the Error Management Counters will be reset. During the waiting time after the resetting of CCCR.INIT, each time a sequence of 11 recessive bits has been monitored, a Bit0Error code is written to PSR.LEC, enabling the CPU to readily check up whether the CAN bus is stuck at dominant or continuously disturbed and to monitor the Bus_Off recovery sequence. ECR.REC is used to count these sequences.

21.7.77 Node i Transmitter Delay Compensation Register

Ni_TDCR (i=0-3) Offset address: $00248_H+i*400_H$

Node i Transmitter Delay Compensation Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	TDCO							0	TDCF						
r	rw							r	rw						

Field	Bits	Type	Description
TDCF	6:0	rw	**Transmitter Delay Compensation Filter Window Length** This bit-field is CCE and INIT protected. Writes will only have effect, if both bits are set. Defines the minimum value for the Secondary Sample Point position, dominant edges on RX that would result in an earlier Secondary Sample Point position are ignored for transmitter delay measurement. This feature is enabled when TDCF is configured to a value greater than TDCO. Valid values are from 0 to 127 mtq.
TDCO	14:8	rw	**Transmitter Delay Compensation Offset** This bit-field is CCE and INIT protected. Writes will only have effect, if both bits are set. Offset value defining the distance between the measured delay from TX to RX and the secondary sample point. Valid values are 0 to 127 mtq. The duration of one mtq is equal to the fASYNi clock period.
0	7, 31:15	r	**Reserved** Read as 0; should be written with 0.

21.7.78 Node i Interrupt Register

The flags are set when one of the listed conditions is detected (edge-sensitive). The flags remain set until the Host clears them. A flag is cleared by writing a “1” to the corresponding bit position. Writing a “0” has no effect. The configuration of Ni_IE controls whether an interrupt is generated.

***Note**: LDMST or SWAPMSK.W instructions should be used only with bit mask enabled for all rwh bits in this register.*

Ni_IR (i=0-3) Offset address: $00250_H+i*400_H$

Node i Interrupt Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		0	PED	PEA	WDI	BO	EW	EP	ELO	0	0	DRX	TOO	MRAF	TSW
r		r	rwh	rwh	rwh	rwh	rwh	rwh	rwh	r	r	rwh	rwh	rwh	rwh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TEFL	TEFF	TEFW	TEFN	TFE	TCF	TC	HPM	RF1L	RF1F	RF1W	RF1N	RF0L	RF0F	RF0W	RF0N
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
RF0N	0	rwh	**Rx FIFO 0 New Message** 0_B No new message written to Rx FIFO 0 1_B New message written to Rx FIFO 0
RF0W	1	rwh	**Rx FIFO 0 Watermark Reached** 0_B Rx FIFO 0 fill level below watermark 1_B Rx FIFO 0 fill level reached watermark

(table continues...)

(continued)

Field	Bits	Type	Description
RF0F	2	rwh	**Rx FIFO 0 Full** 0_B Rx FIFO 0 not full 1_B Rx FIFO 0 full
RF0L	3	rwh	**Rx FIFO 0 Message Lost** 0_B No Rx FIFO 0 message lost 1_B Rx FIFO 0 message lost, also set after write attempt to Rx FIFO 0 of size zero
RF1N	4	rwh	**Rx FIFO 1 New Message** 0_B No new message written to Rx FIFO 1 1_B New message written to Rx FIFO 1
RF1W	5	rwh	**Rx FIFO 1 Watermark Reached** 0_B Rx FIFO 1 fill level below watermark 1_B Rx FIFO 1 fill level reached watermark
RF1F	6	rwh	**Rx FIFO 1 Full** 0_B Rx FIFO 1 not full 1_B Rx FIFO 1 full
RF1L	7	rwh	**Rx FIFO 1 Message Lost** 0_B No Rx FIFO 1 message lost 1_B Rx FIFO 1 message lost, also set after write attempt to Rx FIFO 1 of size zero
HPM	8	rwh	**High Priority Message** 0_B No high priority message received 1_B High priority message received
TC	9	rwh	**Transmission Completed** 0_B No transmission completed 1_B Transmission completed
TCF	10	rwh	**Transmission Cancellation Finished** 0_B No transmission cancellation finished 1_B Transmission cancellation finished
TFE	11	rwh	**Tx FIFO Empty** 0_B Tx FIFO non-empty 1_B Tx FIFO empty
TEFN	12	rwh	**Tx Event FIFO New Entry** 0_B Tx Event FIFO unchanged 1_B Tx Handler wrote Tx Event FIFO element
TEFW	13	rwh	**Tx Event FIFO Watermark Reached** 0_B Tx Event FIFO fill level below watermark 1_B Tx Event FIFO fill level reached watermark

(table continues...)

(continued)

Field	Bits	Type	Description
TEFF	14	rwh	**Tx Event FIFO Full** 0_B Tx Event FIFO not full 1_B Tx Event FIFO full
TEFL	15	rwh	**Tx Event FIFO Element Lost** 0_B No Tx Event FIFO element lost 1_B Tx Event FIFO element lost, also set after write attempt to Tx Event FIFO of size zero
TSW	16	rwh	**Timestamp Wraparound** 0_B No timestamp counter wrap-around 1_B Timestamp counter wrapped around
MRAF	17	rwh	**Message RAM Access Failure** The flag is set, when the Rx Handler • has not completed acceptance filtering or storage of an accepted message until the arbitration field of the following message has been received. In this case acceptance filtering or message storage is aborted and the Rx Handler starts processing of the following message. • was not able to write a message to the Message RAM. In this case message storage is aborted. In both cases the FIFO put index is not updated resp. the New Data flag for a dedicated Rx Buffer is not set, a partly stored message is overwritten when the next message is stored to this location. The flag is also set when the Tx Handler was not able to read a message from the Message RAM in time. In this case message transmission is aborted. In case of a Tx Handler access failure the M_CAN is switched into Restricted Operation Mode. To leave Restricted Operation Mode, the Host CPU has to reset Ni_CCCR.ASM. 0_B No Message RAM access failure occurred 1_B Message RAM access failure occurred
TOO	18	rwh	**Timeout Occurred** 0_B No timeout 1_B Timeout reached
DRX	19	rwh	**Message stored to Dedicated Rx Buffer** The flag is set whenever a received message has been stored into a dedicated Rx Buffer. 0_B No Rx Buffer updated 1_B At least one received message stored into an Rx Buffer
ELO	22	rwh	**Error Logging Overflow** 0_B CAN Error Logging Counter did not overflow 1_B Overflow of CAN Error Logging Counter occurred
EP	23	rwh	**Error Passive** 0_B Error_Passive status unchanged 1_B Error_Passive status changed

(table continues...)

(continued)

Field	Bits	Type	Description
EW	24	rwh	**Warning Status** 0_B Error_Warning status unchanged 1_B Error_Warning status changed
BO	25	rwh	**Bus_Off Status** 0_B Bus_Off status unchanged 1_B Bus_Off status changed
WDI	26	rwh	**Watchdog Interrupt** 0_B No Message RAM Watchdog event occurred 1_B Message RAM Watchdog event due to missing READY
PEA	27	rwh	**Protocol Error in Arbitration Phase** (Nominal Bit Time is used) 0_B No protocol error in arbitration phase 1_B Protocol error in arbitration phase detected (Ni_PSR.LEC ≠ 0,7)
PED	28	rwh	**Protocol Error in Data Phase** (Data Bit Time is used) 0_B No protocol error in data phase detected 1_B Protocol error in data phase detected (Ni_PSR.DLEC ≠ 0,7)
0	20, 21, 29, 31:30	r	**Reserved** Read as 0; shall be written with 0.

21.7.79 Node i Interrupt Enable

The settings in the Interrupt Enable register determine which status changes in the Interrupt Register will be signalled on an interrupt line.

Ni_IE (i=0-3) Offset address: $00254_H+i*400_H$

Node i Interrupt Enable Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		0	PEDE	PEAE	WDIE	BOE	EWE	EPE	ELOE	0	0	DRXE	TOOE	MRAFE	TSWE
r		r	rw	rw	rw	rw	rw	rw	rw	r	r	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TEFLE	TEFFE	TEFWE	TEFNE	TFEE	TCFE	TCE	HPME	RF1LE	RF1FE	RF1WE	RF1NE	RF0LE	RF0FE	RF0WE	RF0NE
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RF0NE	0	rw	**Rx FIFO 0 New Message Interrupt Enable** 0_B Interrupt disabled 1_B Interrupt enabled
RF0WE	1	rw	**Rx FIFO 0 Watermark Reached Interrupt Enable** 0_B Interrupt disabled 1_B Interrupt enabled
RF0FE	2	rw	**Rx FIFO 0 Full Interrupt Enable** 0_B Interrupt disabled 1_B Interrupt enabled
RF0LE	3	rw	**Rx FIFO 0 Message Lost Interrupt Enable** 0_B Interrupt disabled 1_B Interrupt enabled
RF1NE	4	rw	**Rx FIFO 1 New Message Interrupt Enable** 0_B Interrupt disabled 1_B Interrupt enabled
RF1WE	5	rw	**Rx FIFO 1 Watermark Reached Interrupt Enable** 0_B Interrupt disabled 1_B Interrupt enabled
RF1FE	6	rw	**Rx FIFO 1 Full Interrupt Enable** 0_B Interrupt disabled 1_B Interrupt enabled
RF1LE	7	rw	**Rx FIFO 1 Message Lost Interrupt Enable** 0_B Interrupt disabled 1_B Interrupt enabled
HPME	8	rw	**High Priority Message Interrupt Enable** 0_B Interrupt disabled 1_B Interrupt enabled
TCE	9	rw	**Transmission Completed Interrupt Enable** 0_B Interrupt disabled 1_B Interrupt enabled
TCFE	10	rw	**Transmission Cancellation Finished Interrupt Enable** 0_B Interrupt disabled 1_B Interrupt enabled
TFEE	11	rw	**Tx FIFO Empty Interrupt Enable** 0_B Interrupt disabled 1_B Interrupt enabled
TEFNE	12	rw	**Tx Event FIFO New Entry Interrupt Enable** 0_B Interrupt disabled 1_B Interrupt enabled

(table continues...)

(continued)

Field	Bits	Type	Description
TEFWE	13	rw	**Tx Event FIFO Watermark Reached Interrupt Enable** 0_B Interrupt disabled 1_B Interrupt enabled
TEFFE	14	rw	**Tx Event FIFO Full Interrupt Enable** 0_B Interrupt disabled 1_B Interrupt enabled
TEFLE	15	rw	**Tx Event FIFO Element Lost Interrupt Enable** 0_B Interrupt disabled 1_B Interrupt enabled
TSWE	16	rw	**Timestamp Wraparound Interrupt Enable** 0_B Interrupt disabled 1_B Interrupt enabled
MRAFE	17	rw	**Message RAM Access Failure Interrupt Enable** 0_B Interrupt disabled 1_B Interrupt enabled
TOOE	18	rw	**Timeout Occurred Interrupt Enable** 0_B Interrupt disabled 1_B Interrupt enabled
DRXE	19	rw	**Message stored to Dedicated Rx Buffer Interrupt Enable** 0_B Interrupt disabled 1_B Interrupt enabled
ELOE	22	rw	**Error Logging Overflow Interrupt Enable** 0_B Interrupt disabled 1_B Interrupt enabled
EPE	23	rw	**Error Passive Interrupt Enable** 0_B Interrupt disabled 1_B Interrupt enabled
EWE	24	rw	**Warning Status Interrupt Enable** 0_B Interrupt disabled 1_B Interrupt enabled
BOE	25	rw	**Bus_Off Status Interrupt Enable** 0_B Interrupt disabled 1_B Interrupt enabled
WDIE	26	rw	**Watchdog Interrupt Enable** 0_B Interrupt disabled 1_B Interrupt enabled
PEAE	27	rw	**Protocol Error in Arbitration Phase Enable** 0_B Interrupt disabled 1_B Interrupt enabled

(table continues...)

(continued)

Field	Bits	Type	Description
PEDE	28	rw	**Protocol Error in Data Phase Enable** 0_B Interrupt disabled 1_B Interrupt enabled
0	20, 21, 29, 31:30	r	**Reserved** Read as 0; shall be written with 0.

21.7.80 Node i Global Filter Configuration

Global settings for Message ID filtering. The Global Filter Configuration controls the filter path for standard and extended messages as described in Acceptance Filtering Chapter.

Ni_GFC (i=0-3) Offset address: $00280_H+i*400_H$

Node i Global Filter Configuration Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0										ANFS		ANFE		RRFS	RRFE
r										rw		rw		rw	rw

Field	Bits	Type	Description
RRFE	0	rw	**Reject Remote Frames Extended** This bit-field is CCE and INIT protected. Writes will only have effect, if both bits are set. 0_B Filter remote frames with 29-bit extended IDs 1_B Reject all remote frames with 29-bit extended IDs
RRFS	1	rw	**Reject Remote Frames Standard** This bit-field is CCE and INIT protected. Writes will only have effect, if both bits are set. 0_B Filter remote frames with 11-bit standard IDs 1_B Reject all remote frames with 11-bit standard IDs
ANFE	3:2	rw	**Accept Non-matching Frames Extended** This bit-field is CCE and INIT protected. Writes will only have effect, if both bits are set. Defines how received messages with 29-bit IDs that do not match any element of the filter list are treated. 00_B Accept in Rx FIFO 0 01_B Accept in Rx FIFO 1 10_B Reject 11_B Reject

(table continues...)

(continued)

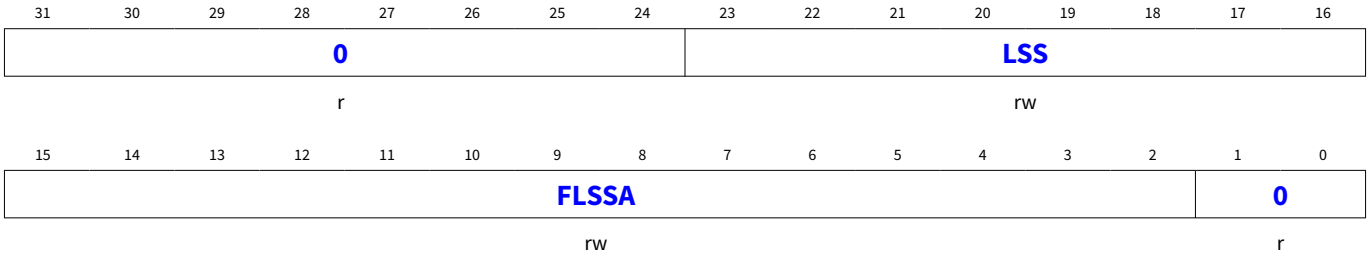
Field	Bits	Type	Description
ANFS	5:4	rw	**Accept Non-matching Frames Standard** This bit-field is CCE and INIT protected. Writes will only have effect, if both bits are set. Defines how received messages with 11-bit IDs that do not match any element of the filter list are treated. 00_B Accept in Rx FIFO 0 01_B Accept in Rx FIFO 1 10_B Reject 11_B Reject
0	31:6	r	**Reserved** Read as 0; should be written with 0.

21.7.81 Node i Standard ID Filter Configuration

Settings for 11-bit standard Message ID filtering. The Standard ID Filter Configuration controls the filter path for standard messages.

Ni_SIDFC (i=0-3) Offset address: $00284_H+i*400_H$

Node i Standard ID Filter Configuration Reset values see: Table 1019



Field	Bits	Type	Description
FLSSA	15:2	rw	**Filter List Standard Start Address** This bit-field is CCE and INIT protected. Writes will only have effect, if both bits are set. Start address of standard Message ID filter list (32-bit word address).
LSS	23:16	rw	**List Size Standard** This bit-field is CCE and INIT protected. Writes will only have effect, if both bits are set. 00_H No standard Message ID filter 01_H 1 Message ID filter elements … 80_H 128 Message ID filter elements **others,** 128 Message ID filter elements
0	1:0, 31:24	r	**Reserved** Read as 0; should be written with 0.

Table 1019 Reset values of Ni_SIDFC (i=0-3)

Reset type	Reset value	Note
Kernel Reset	0000 0000$_H$	
After Boot-FW Value	0001 0000$_H$	After CAN BSL execution (applicable only for CAN0 and Node 1)

21.7.82 Node i Extended ID Filter Configuration

Settings for 29-bit extended Message ID filtering. The Extended ID Filter Configuration controls the filter path for standard messages as described in Acceptance Filtering Chapter.

Ni_XIDFC (i=0-3) Offset address: 00288$_H$+i*400$_H$

Node i Extended ID Filter Configuration Kernel Reset value: 0000 0000$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0									LSE						
r									rw						

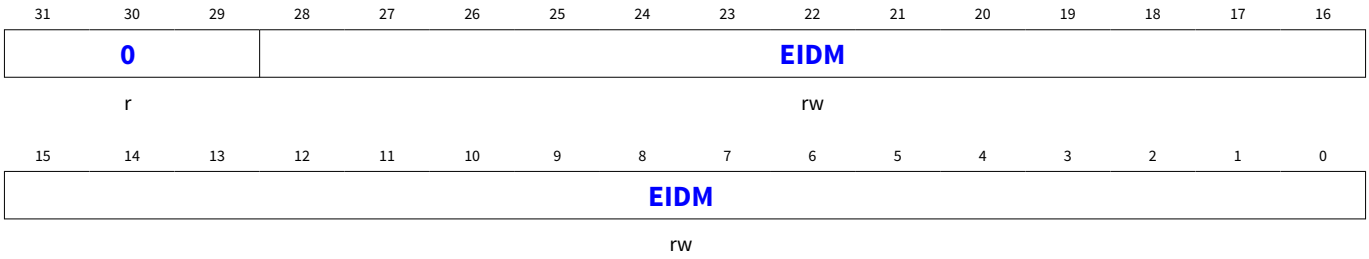
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FLESA														0	
rw														r	

Field	Bits	Type	Description
FLESA	15:2	rw	**Filter List Extended Start Address** This bit-field is CCE and INIT protected. Writes will only have effect, if both bits are set. Start address of extended Message ID filter list (32-bit word addess).
LSE	22:16	rw	**List Size Extended** This bit-field is CCE and INIT protected. Writes will only have effect, if both bits are set. 00$_H$ No standard Message ID filter 01$_H$ 1 extended Message ID filter element … 40$_H$ 64 extended Message ID filter element **others,** 64 extended Message ID filter elements
0	1:0, 31:23	r	**Reserved** Read as 0; should be written with 0.

21.7.83 Node i Extended ID AND Mask

Ni_XIDAM (i=0-3) Offset address: 00290$_H$+i*400$_H$

Node i Extended ID AND Mask Kernel Reset value: 1FFF FFFF$_H$



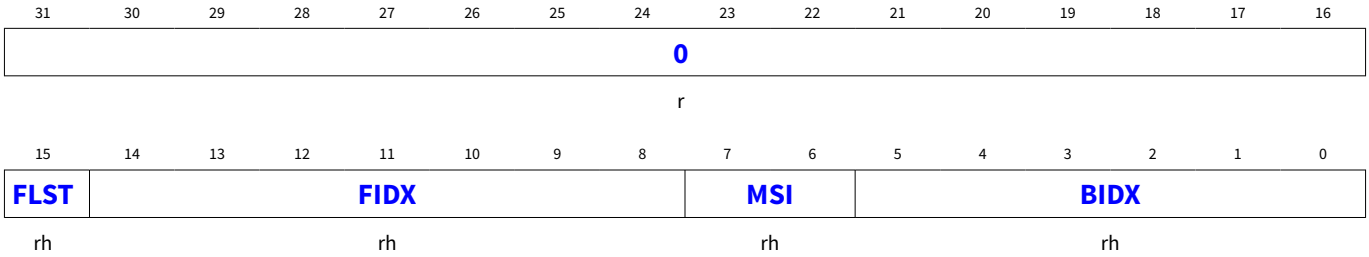
Field	Bits	Type	Description
EIDM	28:0	rw	**Extended ID Mask** This bit-field is CCE and INIT protected. Writes will only have effect, if both bits are set. For acceptance filtering of extended frames the Extended ID AND Mask is ANDed with the Message ID of a received frame. Intended for masking of 29-bit IDs in SAE J1939. With the reset value of all bits set to one the mask is not active.
0	31:29	r	**Reserved** Read as 0; should be written with 0.

21.7.84 Node i High Priority Message Status

This register is updated every time a Message ID filter element configured to generate a priority event matches. This can be used to monitor the status of incoming high priority messages and to enable fast access to these messages.

Ni_HPMS (i=0-3) Offset address: $00294_H+i*400_H$

Node i High Priority Message Status Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
BIDX	5:0	rh	**Buffer Index** Index of Rx FIFO element to which the message was stored. Only valid when MSI[1] = '1'.
MSI	7:6	rh	**Message Storage Indicator** 00_B No FIFO selected 01_B FIFO message lost 10_B Message stored in FIFO 0 11_B Message stored in FIFO 1

(table continues...)

(continued)

Field	Bits	Type	Description
FIDX	14:8	rh	**Filter Index** Index of matching filter element. Range is 0 to Ni_SIDFC.LSS - 1 resp. Ni_XIDFC.LSE - 1.
FLST	15	rh	**Filter List** Indicates the filter list of the matching filter element. 0_B Standard Filter List 1_B Extended Filter List
0	31:16	r	**Reserved** Read as 0; should be written with 0.

21.7.85 Node i New Data 1

The register holds the New Data flags of Rx Buffers 0 to 31.

***Note**: LDMST or SWAPMSK.W instructions should be used only with bit mask enabled for all rwh bits in this register.*

Ni_NDAT1 (i=0-3) Offset address: $00298_H+i*400_H$

Node i New Data 1 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ND31	ND30	ND29	ND28	ND27	ND26	ND25	ND24	ND23	ND22	ND21	ND20	ND19	ND18	ND17	ND16
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ND15	ND14	ND13	ND12	ND11	ND10	ND9	ND8	ND7	ND6	ND5	ND4	ND3	ND2	ND1	ND0
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
NDy (y=0-31)	y	rwh	**New Data in Rx Buffer y - ND** The flag is set when the respective Rx Buffer has been updated from a received frame. The flags remain set until the Host clears them. A flag is cleared by writing a “1” to the corresponding bit position. Writing a “0” has no effect. 0_B Rx Buffer not updated 1_B Rx Buffer updated from new message

21.7.86 Node i New Data 2

The register holds the New Data flags of Rx Buffers 32 to 63.

***Note**: LDMST or SWAPMSK.W instructions should be used only with bit mask enabled for all rwh bits in this register.*

Ni_NDAT2 (i=0-3) Offset address: $0029C_H+i*400_H$

Node i New Data 2 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ND63	ND62	ND61	ND60	ND59	ND58	ND57	ND56	ND55	ND54	ND53	ND52	ND51	ND50	ND49	ND48
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ND47	ND46	ND45	ND44	ND43	ND42	ND41	ND40	ND39	ND38	ND37	ND36	ND35	ND34	ND33	ND32
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
NDy (y=32-63)	y-32	rwh	**New Data in Rx Buffer y - ND** The flag is set when the respective Rx Buffer has been updated from a received frame. The flags remain set until the Host clears them. A flag is cleared by writing a “1” to the corresponding bit position. Writing a “0” has no effect. 0_B Rx Buffer not updated 1_B Rx Buffer updated from new message

21.7.87 Node i Rx FIFO 0 Configuration

Ni_RXF0C (i=0-3) Offset address: $002A0_H+i*400_H$

Node i Rx FIFO 0 Configuration Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
F0OM	F0WM							0	F0S						
rw	rw							r	rw						

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
F0SA														0	
rw														r	

Field	Bits	Type	Description
F0SA	15:2	rw	**Rx FIFO 0 Start Address** This bit-field is CCE and INIT protected. Writes will only have effect, if both bits are set. Start address of Rx FIFO 0 in Message RAM (32-bit word address, see Message RAM Configuration Chapter).
F0S	22:16	rw	**Rx FIFO 0 Size** This bit-field is CCE and INIT protected. Writes will only have effect, if both bits are set. 00_H No Rx FIFO 0 01_H 1 Rx FIFO 0 elements … 40_H 64 Rx FIFO 0 elements **others,** 64 Rx FIFO 0 elements

(table continues...)

(continued)

Field	Bits	Type	Description
F0WM	30:24	rw	**Rx FIFO 0 Watermark** This bit-field is CCE and INIT protected. Writes will only have effect, if both bits are set. 01_H Level 1 for Rx FIFO 0 watermark interrupt (IR.RF0W) … 40_H Level 64 for Rx FIFO 0 watermark interrupt (IR.RF0W) **others,** Watermark interrupt disabled
F0OM	31	rw	**FIFO 0 Operation Mode** This bit-field is CCE and INIT protected. Writes will only have effect, if both bits are set. FIFO 0 can be operated in blocking or in overwrite mode (see Rx FIFOs Chapter). 0_B FIFO 0 blocking mode 1_B FIFO 0 overwrite mode
0	1:0, 23	r	**Reserved** Read as 0; should be written with 0.

21.7.88 Node i Rx FIFO 0 Status

Ni_RXF0S (i=0-3) Offset address: $002A4_H+i*400_H$

Node i Rx FIFO 0 Status Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0						RF0L	F0F	0		F0PI					
r						rh	rh	r		rh					

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		F0GI						0		F0FL					
r		rh						r		rh					

Field	Bits	Type	Description
F0FL	6:0	rh	**Rx FIFO 0 Fill Level** Number of elements stored in Rx FIFO 0, range 0 to 64.
F0GI	13:8	rh	**Rx FIFO 0 Get Index** Rx FIFO 0 read index pointer, range 0 to 63.
F0PI	21:16	rh	**Rx FIFO 0 Put Index** Rx FIFO 0 write index pointer, range 0 to 63.
F0F	24	rh	**Rx FIFO 0 Full** 0_B Rx FIFO 0 not full 1_B Rx FIFO 0 full

(table continues...)

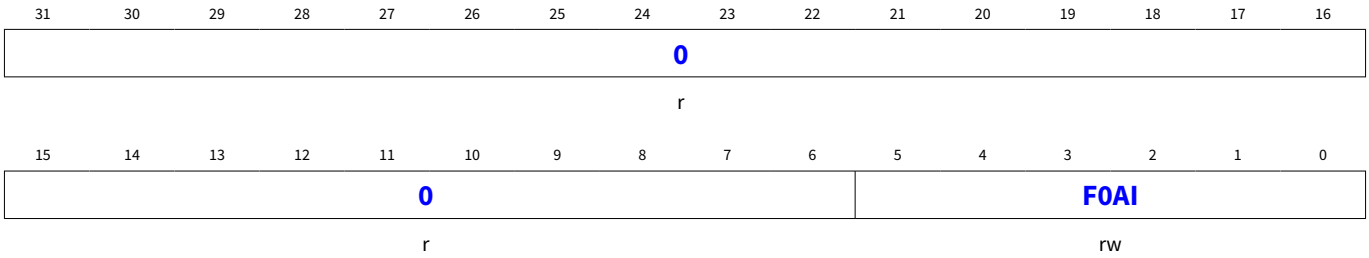
(continued)

Field	Bits	Type	Description
RF0L	25	rh	**Rx FIFO 0 Message Lost** This bit is a copy of interrupt flag Ni_IR.RF0L. When Ni_IR.RF0L is reset, this bit is also reset. ***Note:*** *Overwriting the oldest message when Ni_RXF0C.F0OM = '1' will not set this flag.* 0_B No Rx FIFO 0 message lost 1_B Rx FIFO 0 message lost, also set after write attempt to Rx FIFO 0 of size zero
0	7, 15:14, 23:22, 31:26	r	**Reserved** Read as 0; should be written with 0.

21.7.89 Node i Rx FIFO 0 Acknowledge

Ni_RXF0A (i=0-3) Offset address: $002A8_H+i*400_H$

Node i Rx FIFO 0 Acknowledge Kernel Reset value: $0000\ 0000_H$

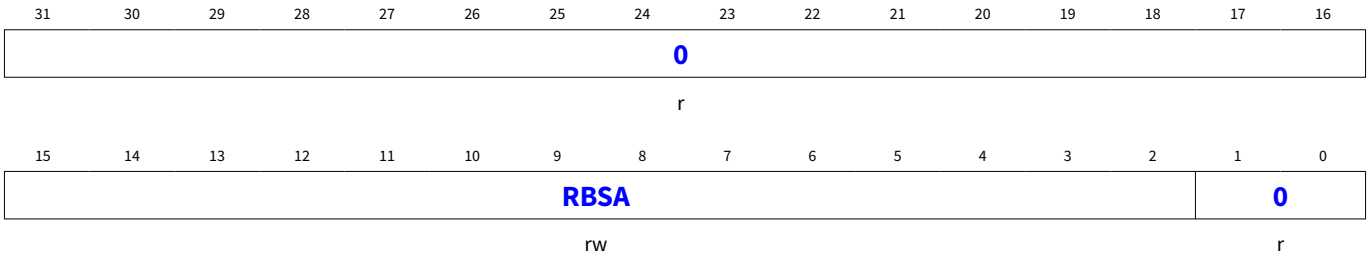


Field	Bits	Type	Description
F0AI	5:0	rw	**Rx FIFO 0 Acknowledge Index** After the Host has read a message or a sequence of messages from Rx FIFO 0 it has to write the buffer index of the last element read from Rx FIFO 0 to F0AI. This will set the Rx FIFO 0 Get Index Ni_RXF0S.F0GI to F0AI + 1 and update the FIFO 0 Fill Level Ni_RXF0S.F0FL.
0	31:6	r	**Reserved** Read as 0; should be written with 0.

21.7.90 Node i Rx Buffer Configuration

Ni_RXBC (i=0-3) Offset address: $002AC_H+i*400_H$

Node i Rx Buffer Configuration Reset values see: Table 1020



Field	Bits	Type	Description
RBSA	15:2	rw	**Rx Buffer Start Address** This bit-field is CCE and INIT protected. Writes will only have effect, if both bits are set. Configures the start address of the Rx Buffers section in the Message RAM (32-bit word address).
0	1:0, 31:16	r	**Reserved** Read as 0; should be written with 0.

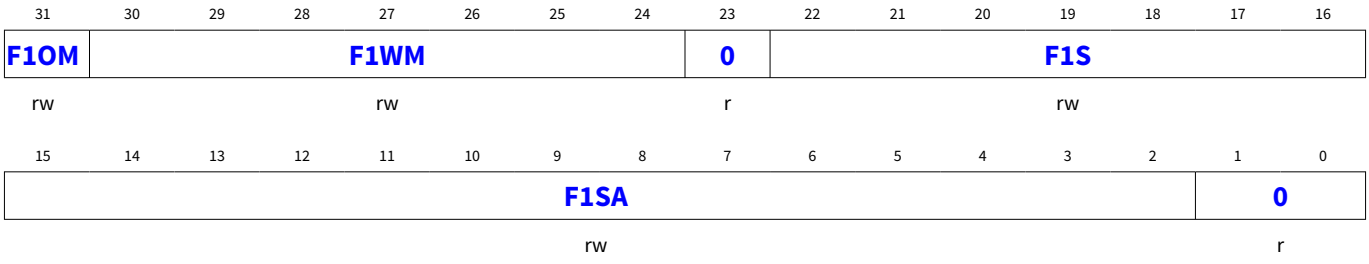
Table 1020 Reset values of Ni_RXBC (i=0-3)

Reset type	Reset value	Note
Kernel Reset	0000 0000$_H$	
After Boot-FW Value	0000 0008$_H$	After CAN BSL execution (applicable only for CAN0 and Node 1)

21.7.91 Node i Rx FIFO 1 Configuration

Ni_RXF1C (i=0-3) Offset address: 002B0$_H$+i*400$_H$

Node i Rx FIFO 1 Configuration Kernel Reset value: 0000 0000$_H$



Field	Bits	Type	Description
F1SA	15:2	rw	**Rx FIFO 1 Start Address** This bit-field is CCE and INIT protected. Writes will only have effect, if both bits are set. Start address of Rx FIFO 1 in Message RAM (32-bit word address).

(table continues...)

(continued)

Field	Bits	Type	Description
F1S	22:16	rw	**Rx FIFO 1 Size** This bit-field is CCE and INIT protected. Writes will only have effect, if both bits are set. 00_H No Rx FIFO 1 01_H 1 Rx FIFO 1 elements … 40_H 64 Rx FIFO 1 elements **others,** 64 Rx FIFO 1 elements
F1WM	30:24	rw	**Rx FIFO 1 Watermark** This bit-field is CCE and INIT protected. Writes will only have effect, if both bits are set. 01_H Level 1 for Rx FIFO 1 watermark interrupt (IR.RF1W) … 40_H Level 64 for Rx FIFO 1 watermark interrupt (IR.RF1W) **others,** Watermark interrupt disabled
F1OM	31	rw	**FIFO 1 Operation Mode** This bit-field is CCE and INIT protected. Writes will only have effect, if both bits are set. FIFO 1 can be operated in blocking or in overwrite mode. 0_B FIFO 1 blocking mode 1_B FIFO 1 overwrite mode
0	1:0, 23	r	**Reserved** Read as 0; should be written with 0.

21.7.92 Node i Rx FIFO 1 Status

Ni_RXF1S (i=0-3) Offset address: $002B4_H+i*400_H$

Node i Rx FIFO 1 Status Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		0				RF1L	F1F	0		F1PI					
r		r				rh	rh	r		rh					

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		F1GI						0		F1FL					
r		rh						r		rh					

Field	Bits	Type	Description
F1FL	6:0	rh	**Rx FIFO 1 Fill Level** Number of elements stored in Rx FIFO 1, range 0 to 64.
F1GI	13:8	rh	**Rx FIFO 1 Get Index** Rx FIFO 1 read index pointer, range 0 to 63.

(table continues...)

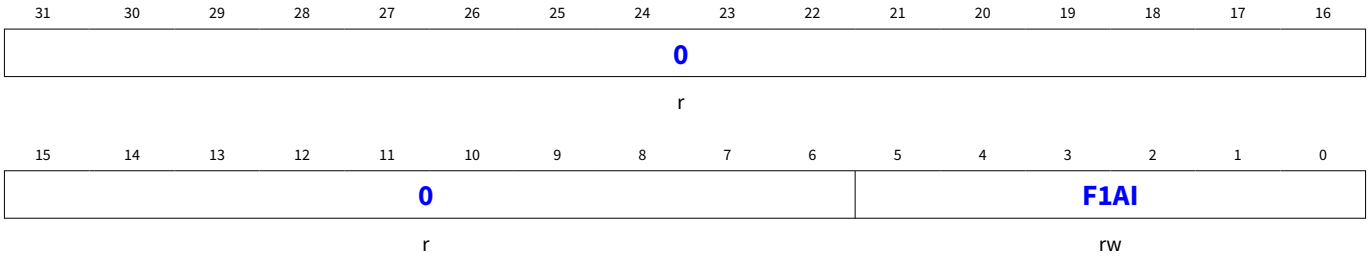
(continued)

Field	Bits	Type	Description
F1PI	21:16	rh	**Rx FIFO 1 Put Index** Rx FIFO 1 write index pointer, range 0 to 63.
F1F	24	rh	**Rx FIFO 1 Full** 0_B Rx FIFO 1 not full 1_B Rx FIFO 1 full
RF1L	25	rh	**Rx FIFO 1 Message Lost** This bit is a copy of interrupt flag Ni_IR.RF1L. When Ni_IR.RF1L is reset, this bit is also reset. ***Note**: Overwriting the oldest message when Ni_RXF1C.F1OM = '1' will not set this flag.* 0_B No Rx FIFO 1 message lost 1_B Rx FIFO 1 message lost, also set after write attempt to Rx FIFO 1 of size zero
0	7, 15:14, 23:22, 29:26, 31:30	r	**Reserved** Read as 0; should be written with 0.

21.7.93 Node i Rx FIFO 1 Acknowledge

Ni_RXF1A (i=0-3) Offset address: $002B8_H+i*400_H$

Node i Rx FIFO 1 Acknowledge Kernel Reset value: $0000\ 0000_H$



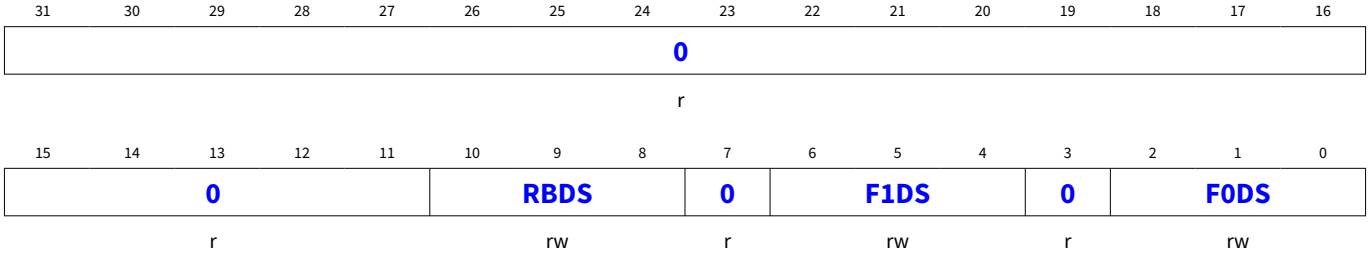
Field	Bits	Type	Description
F1AI	5:0	rw	**Rx FIFO 1 Acknowledge Index** After the Host has read a message or a sequence of messages from Rx FIFO 1 it has to write the buffer index of the last element read from Rx FIFO 1 to F1AI. This will set the Rx FIFO 1 Get Index Ni_RXF1S.F1GI to F1AI + 1 and update the FIFO 1 Fill Level Ni_RXF1S.F1FL
0	31:6	r	**Reserved** Read as 0; should be written with 0.

21.7.94 Node i Rx Buffer/FIFO Element Size Configuration

Configures the number of data bytes belonging to an Rx Buffer / Rx FIFO element. Data field sizes > 8 bytes are intended for CAN FD operation only.

Ni_RXESC (i=0-3) Offset address: $002BC_H+i*400_H$

Node i Rx Buffer/FIFO Element Size Configuration Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
F0DS	2:0	rw	**Rx FIFO 0 Data Field Size** This bit-field is CCE and INIT protected. Writes will only have effect, if both bits are set. ***Note**: In case the data field size of an accepted CAN frame exceeds the data field size configured for the matching Rx Buffer or Rx FIFO, only the number of bytes as configured by this bit-field are stored to the Rx Buffer resp. Rx FIFO element. The rest of the frame's data field is ignored.* 000_B 8-byte data field 001_B 12-byte data field 010_B 16-byte data field 011_B 20-byte data field 100_B 24-byte data field 101_B 32-byte data field 110_B 48-byte data field 111_B 64-byte data field
F1DS	6:4	rw	**Rx FIFO 1 Data Field Size** This bit-field is CCE and INIT protected. Writes will only have effect, if both bits are set. 000_B 8-byte data field 001_B 12-byte data field 010_B 16-byte data field 011_B 20-byte data field 100_B 24-byte data field 101_B 32-byte data field 110_B 48-byte data field 111_B 64-byte data field

(table continues...)

(continued)

Field	Bits	Type	Description
RBDS	10:8	rw	**Rx Buffer Data Field Size** This bit-field is CCE and INIT protected. Writes will only have effect, if both bits are set. 000_B 8-byte data field 001_B 12-byte data field 010_B 16-byte data field 011_B 20-byte data field 100_B 24-byte data field 101_B 32-byte data field 110_B 48-byte data field 111_B 64-byte data field
0	3, 7, 31:11	r	**Reserved** Read as 0; should be written with 0.

21.7.95 Node i Tx Buffer Configuration

Ni_TXBC (i=0-3) Offset address: $002C0_H+i*400_H$

Node i Tx Buffer Configuration Reset values see: Table 1021

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	TFQM	TFQS						0		NDTB					
r	rw	rw						r		rw					

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TBSA														0	
rw														r	

Field	Bits	Type	Description
TBSA	15:2	rw	**Tx Buffers Start Address** This bit-field is CCE and INIT protected. Writes will only have effect, if both bits are set. Start address of Tx Buffers section in Message RAM (32-bit word address).

(table continues...)

(continued)

Field	Bits	Type	Description
NDTB	21:16	rw	**Number of Dedicated Transmit Buffers** This bit-field is CCE and INIT protected. Writes will only have effect, if both bits are set. ***Note:*** *Be aware that the sum of TFQS and NDTB may be not greater than 32. There is no check for erroneous configurations. The Tx Buffers section in the Message RAM starts with the dedicated Tx Buffers.* 00_H No Dedicated Tx Buffers 01_H 1 Dedicated Tx Buffers … 20_H 32 Dedicated Tx Buffers **others,** 32 Dedicated Tx Buffers
TFQS	29:24	rw	**Transmit FIFO/Queue Size** This bit-field is CCE and INIT protected. Writes will only have effect, if both bits are set. 00_H No Tx FIFO/Queue 01_H 1 Tx Buffers used for Tx FIFO/Queue … 20_H 32 Tx Buffers used for Tx FIFO/Queue **others,** 32 Tx Buffers used for Tx FIFO/Queue
TFQM	30	rw	**Tx FIFO/Queue Mode** This bit-field is CCE and INIT protected. Writes will only have effect, if both bits are set. 0_B Tx FIFO operation 1_B Tx Queue operation
0	1:0, 23:22, 31	r	**Reserved** Read as 0; should be written with 0.

Table 1021 Reset values of Ni_TXBC (i=0-3)

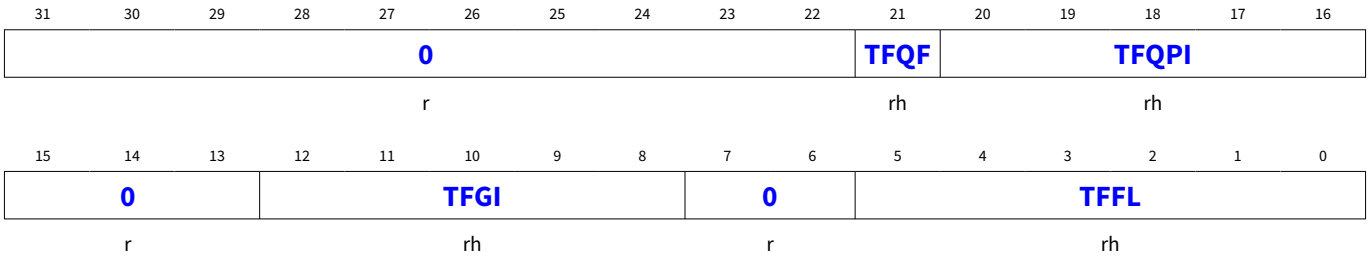
Reset type	Reset value	Note
Kernel Reset	$0000\ 0000_H$	
After Boot-FW Value	$0000\ 1208_H$	After CAN BSL execution (applicable only for CAN0 and Node 1)

21.7.96 Node i Tx FIFO/Queue Status

The Tx FIFO/Queue status is related to the pending Tx requests listed in register Ni_TXBRP. Therefore the effect of Add/Cancellation requests may be delayed due to a running Tx scan (Ni_TXBRP not yet updated).

Ni_TXFQS (i=0-3) Offset address: $002C4_H+i*400_H$

Node i Tx FIFO/Queue Status Kernel Reset value: $0000\ 0000_H$



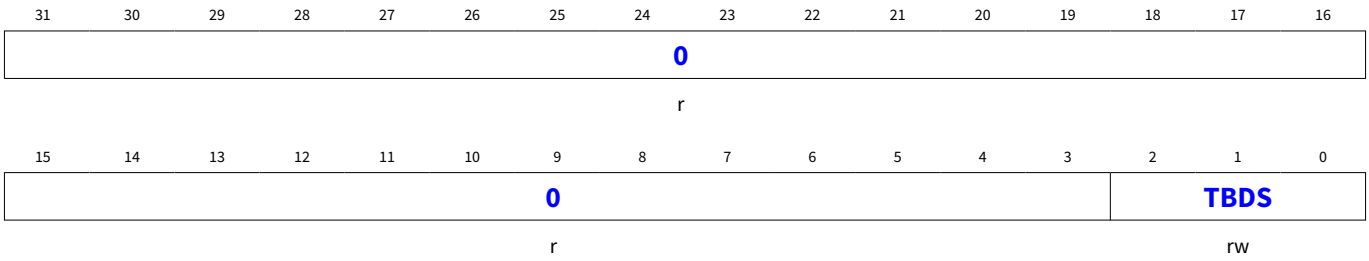
Field	Bits	Type	Description
TFFL	5:0	rh	**Tx FIFO Free Level** Number of consecutive free Tx FIFO elements starting from TFGI, range 0 to 32. Read as zero when Tx Queue operation is configured (Ni_TXBC.TFQM = '1') ***Note**: In case of mixed configurations where dedicated Tx Buffers are combined with a Tx FIFO or a Tx Queue, the Put and Get Indices indicate the number of the Tx Buffer starting with the first dedicated Tx Buffers. Example: For a configuration of 12 dedicated Tx Buffers and a Tx FIFO of 20 Buffers a Put Index of 15 points to the fourth buffer of the Tx FIFO.*
TFGI	12:8	rh	**Tx FIFO Get Index** Tx FIFO read index pointer, range 0 to 31. Read as zero when Tx Queue operation is configured (Ni_TXBC.TFQM = '1').
TFQPI	20:16	rh	**Tx FIFO/Queue Put Index** Tx FIFO/Queue write index pointer, range 0 to 31.
TFQF	21	rh	**Tx FIFO/Queue Full** 0_B Tx FIFO/Queue not full 1_B Tx FIFO/Queue full
0	7:6, 15:13, 31:22	r	**Reserved** Read as 0; should be written with 0.

21.7.97 Node i Tx Buffer Element Size Configuration

Configures the number of data bytes belonging to a Tx Buffer element. Data field sizes > 8 bytes are intended for CAN FD operation only.

Ni_TXESC (i=0-3) Offset address: $002C8_H+i*400_H$

Node i Tx Buffer Element Size Configuration Reset values see: Table 1022



Field	Bits	Type	Description
TBDS	2:0	rw	**Tx Buffer Data Field Size** This bit-field is CCE and INIT protected. Writes will only have effect, if both bits are set. ***Note**: In case the data length code DLC of a Tx Buffer element is configured to a value higher than the Tx Buffer data field size Ni_TXESC.TBDS, the bytes not defined by the Tx Buffer are transmitted as "0xCC" (padding bytes).* 000_B 8-byte data field 001_B 12-byte data field 010_B 16-byte data field 011_B 20-byte data field 100_B 24-byte data field 101_B 32-byte data field 110_B 48-byte data field 111_B 64-byte data field
0	31:3	r	**Reserved** Read as 0; should be written with 0.

Table 1022 Reset values of Ni_TXESC (i=0-3)

Reset type	Reset value	Note
Kernel Reset	$0000\ 0000_H$	
After Boot-FW Value	$0000\ 0007_H$	After CAN BSL execution (applicable only for CAN0 and Node 1)

21.7.98 Node i Tx Buffer Request Pending

Each Tx Buffer has its own Transmission Request Pending bit. The bits are set via register Ni_TXBAR. The bits are reset after a requested transmission has completed or has been cancelled via register Ni_TXBCR.

Ni_TXBRP bits are set only for those Tx Buffers configured via Ni_TXBC. After a TXBRP bit has been set, a Tx scan (Tx Handling) is started to check for the pending Tx request with the highest priority (Tx Buffer with lowest Message ID).

A cancellation request resets the corresponding transmission request pending bit of register Ni_TXBRP. In case a transmission has already been started when a cancellation is requested, this is done at the end of the transmission, regardless whether the transmission was successful or not. The cancellation request bits are reset directly after the corresponding TXBRP bit has been reset.

After a cancellation has been requested, a finished cancellation is signalled via Ni_TXBCF

- after successful transmission together with the corresponding Ni_TXBTO bit
- when the transmission has not yet been started at the point of cancellation
- when the transmission has been aborted due to lost arbitration
- when an error occurred during frame transmission

In "Disable Automatic Retransmission" mode all transmissions are automatically cancelled if they are not successful. The corresponding Ni_TXBCF bit is set for all unsuccessful transmissions.

***Note**: TXBRP bits which are set while a Tx scan is in progress are not considered during this particular Tx scan. In case a cancellation is requested for such a Tx Buffer, this "Add Request" is cancelled immediately, the corresponding Ni_TXBRP bit is reset.*

Ni_TXBRP (i=0-3) Offset address: $002CC_H+i*400_H$

Node i Tx Buffer Request Pending Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TRP31	TRP30	TRP29	TRP28	TRP27	TRP26	TRP25	TRP24	TRP23	TRP22	TRP21	TRP20	TRP19	TRP18	TRP17	TRP16
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TRP15	TRP14	TRP13	TRP12	TRP11	TRP10	TRP9	TRP8	TRP7	TRP6	TRP5	TRP4	TRP3	TRP2	TRP1	TRP0
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
TRPz (z=0-31)	z	rh	**Transmission Request Pending Tx Buffer z - TRP** 0_B No transmission request pending 1_B Transmission request pending

21.7.99 Node i Tx Buffer Add Request

Each Tx Buffer has its own "Add Request" bit. Writing a ‘1’ will set the corresponding "Add Request" bit; writing a ‘0’ has no impact. This enables the Host to set transmission requests for multiple Tx Buffers with one write to Ni_TXBAR. Ni_TXBAR bits are set only for those Tx Buffers configured via Ni_TXBC. When no Tx scan is running, the bits are reset immediately, else the bits remain set until the Tx scan process has completed.

***Note**: If an add request is applied for a Tx Buffer with pending transmission request (corresponding Ni_TXBRP bit already set), this add request is ignored. LDMST or SWAPMSK.W instructions should be used only with bit mask enabled for all rwh bits in this register.*

Ni_TXBAR (i=0-3) Offset address: $002D0_H+i*400_H$

Node i Tx Buffer Add Request Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AR31	AR30	AR29	AR28	AR27	AR26	AR25	AR24	AR23	AR22	AR21	AR20	AR19	AR18	AR17	AR16
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AR15	AR14	AR13	AR12	AR11	AR10	AR9	AR8	AR7	AR6	AR5	AR4	AR3	AR2	AR1	AR0
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
ARz (z=0-31)	z	rwh	**Add Request Tx Buffer z - AR** 0_B No transmission request added 1_B Transmission requested added

21.7.100 Node i Tx Buffer Cancellation Request

Each Tx Buffer has its own Cancellation Request bit. Writing a ‘1’ will set the corresponding Cancellation Request bit; writing a ‘0’ has no impact. This enables the Host to set cancellation requests for multiple Tx Buffers with one write to Ni_TXBCR. Ni_TXBCR bits are set only for those Tx Buffers configured via Ni_TXBC. The bits remain set until the corresponding bit of Ni_TXBRP is reset.

***Note**: LDMST or SWAPMSK.W instructions should be used only with bit mask enabled for all rwh bits in this register.*

Ni_TXBCR (i=0-3) Offset address: $002D4_H+i*400_H$

Node i Tx Buffer Cancellation Request Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CR31	CR30	CR29	CR28	CR27	CR26	CR25	CR24	CR23	CR22	CR21	CR20	CR19	CR18	CR17	CR16
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CR15	CR14	CR13	CR12	CR11	CR10	CR9	CR8	CR7	CR6	CR5	CR4	CR3	CR2	CR1	CR0
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
CRz (z=0-31)	z	rwh	**Cancellation Request Tx Buffer z - CR** 0_B No cancellation pending 1_B Cancellation pending

21.7.101 Node i Tx Buffer Transmission Occurred

Each Tx Buffer has its own Transmission Occurred bit. The bits are set when the corresponding Ni_TXBRP bit is cleared after a successful transmission. The bits are reset when a new transmission is requested by writing a '1' to the corresponding bit of register Ni_TXBAR.

Ni_TXBTO (i=0-3) Offset address: $002D8_H+i*400_H$

Node i Tx Buffer Transmission Occurred Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TO31	TO30	TO29	TO28	TO27	TO26	TO25	TO24	TO23	TO22	TO21	TO20	TO19	TO18	TO17	TO16
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TO15	TO14	TO13	TO12	TO11	TO10	TO9	TO8	TO7	TO6	TO5	TO4	TO3	TO2	TO1	TO0
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
TOz (z=0-31)	z	rh	**Transmission Occurred Tx Buffer z - TO** 0_B No transmission occurred 1_B Transmission occurred

21.7.102 Node i Tx Buffer Cancellation Finished

Each Tx Buffer has its own Cancellation Finished bit. The bits are set when the corresponding Ni_TXBRP bit is cleared after a cancellation was requested via Ni_TXBCR. In case the corresponding Ni_TXBRP bit was not set at the point of cancellation, CF is set immediately. The bits are reset when a new transmission is requested by writing a '1' to the corresponding bit of register Ni_TXBAR.

Ni_TXBCF (i=0-3) Offset address: $002DC_H+i*400_H$

Node i Tx Buffer Cancellation Finished Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CF31	CF30	CF29	CF28	CF27	CF26	CF25	CF24	CF23	CF22	CF21	CF20	CF19	CF18	CF17	CF16
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CF15	CF14	CF13	CF12	CF11	CF10	CF9	CF8	CF7	CF6	CF5	CF4	CF3	CF2	CF1	CF0
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
CFz (z=0-31)	z	rh	**Cancellation Finished Tx Buffer z - CF** 0_B No transmit buffer cancellation 1_B Transmit buffer cancellation finished

21.7.103 Node i Tx Buffer Transmission Interrupt Enable

Each Tx Buffer has its own Transmission Interrupt Enable bit.

Ni_TXBTIE (i=0-3) Offset address: $002E0_H+i*400_H$

Node i Tx Buffer Transmission Interrupt Enable Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TIE31	TIE30	TIE29	TIE28	TIE27	TIE26	TIE25	TIE24	TIE23	TIE22	TIE21	TIE20	TIE19	TIE18	TIE17	TIE16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TIE15	TIE14	TIE13	TIE12	TIE11	TIE10	TIE9	TIE8	TIE7	TIE6	TIE5	TIE4	TIE3	TIE2	TIE1	TIE0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
TIEz (z=0-31)	z	rw	**Transmission Interrupt Enable Tx Buffer z - TIE** 0_B Transmission interrupt disabled 1_B Transmission interrupt enable

21.7.104 Node i Tx Buffer Cancellation Finished Interrupt Enable

Each Tx Buffer has its own Cancellation Finished Interrupt Enable bit.

Ni_TXBCIE (i=0-3) Offset address: $002E4_H+i*400_H$

Node i Tx Buffer Cancellation Finished Interrupt Enable Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CFIE31	CFIE30	CFIE29	CFIE28	CFIE27	CFIE26	CFIE25	CFIE24	CFIE23	CFIE22	CFIE21	CFIE20	CFIE19	CFIE18	CFIE17	CFIE16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CFIE15	CFIE14	CFIE13	CFIE12	CFIE11	CFIE10	CFIE9	CFIE8	CFIE7	CFIE6	CFIE5	CFIE4	CFIE3	CFIE2	CFIE1	CFIE0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
CFIEz (z=0-31)	z	rw	**Cancellation Finished Interrupt Enable Tx Buffer z - CFIE** 0_B Cancellation finished interrupt disabled 1_B Cancellation finished interrupt enabled

21.7.105 Node i Tx Event FIFO Configuration

Ni_TXEFC (i=0-3) Offset address: $002F0_H+i*400_H$

Node i Tx Event FIFO Configuration Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		EFWM						0		EFS					
r		rw						r		rw					

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EFSA														0	
rw														r	

Field	Bits	Type	Description
EFSA	15:2	rw	**Event FIFO Start Address** This bit-field is CCE and INIT protected. Writes will only have effect, if both bits are set. Start address of Tx Event FIFO in Message RAM (32-bit word address).
EFS	21:16	rw	**Event FIFO Size** This bit-field is CCE and INIT protected. Writes will only have effect, if both bits are set. The Tx Event FIFO elements are indexed from 0 to EFS - 1 00_H Tx Event FIFO disabled 01_H 1 Tx Event FIFO elements … 20_H 32 Tx Event FIFO elements **others,** 32 Tx Event FIFO elements
EFWM	29:24	rw	**Event FIFO Watermark** This bit-field is CCE and INIT protected. Writes will only have effect, if both bits are set. 01_H Level 1 for Tx Event FIFO watermark interrupt (Ni_IR.TEFW) … 20_H Level 32 for Tx Event FIFO watermark interrupt (Ni_IR.TEFW) **others,** Watermark interrupt disabled
0	1:0, 23:22, 31:30	r	**Reserved** Read as 0; should be written with 0.

21.7.106 Node i Tx Event FIFO Status

Ni_TXEFS (i=0-3) Offset address: $002F4_H+i*400_H$

Node i Tx Event FIFO Status Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0						TEFL	EFF	0			EFPI				
r						rh	rh	r			rh				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			EFGI					0		EFFL					
r			rh					r		rh					

Field	Bits	Type	Description
EFFL	5:0	rh	**Event FIFO Fill Level** Number of elements stored in Tx Event FIFO, range 0 to 32.
EFGI	12:8	rh	**Event FIFO Get Index** Tx Event FIFO read index pointer, range 0 to 31.
EFPI	20:16	rh	**Event FIFO Put Index** Tx Event FIFO write index pointer, range 0 to 31.
EFF	24	rh	**Event FIFO Full** 0_B Tx Event FIFO not full 1_B Tx Event FIFO full
TEFL	25	rh	**Tx Event FIFO Element Lost** This bit is a copy of interrupt flag Ni_IR.TEFL. When Ni_IR.TEFL is reset, this bit is also reset. 0_B No Tx Event FIFO element lost 1_B Tx Event FIFO element lost, also set after write attempt to Tx Event FIFO of size zero.
0	7:6, 15:13, 23:21, 31:26	r	**Reserved** Read as 0; should be written with 0.

21.7.107 Node i Tx Event FIFO Acknowledge

Ni_TXEFA (i=0-3) Offset address: $002F8_H+i*400_H$

Node i Tx Event FIFO Acknowledge Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

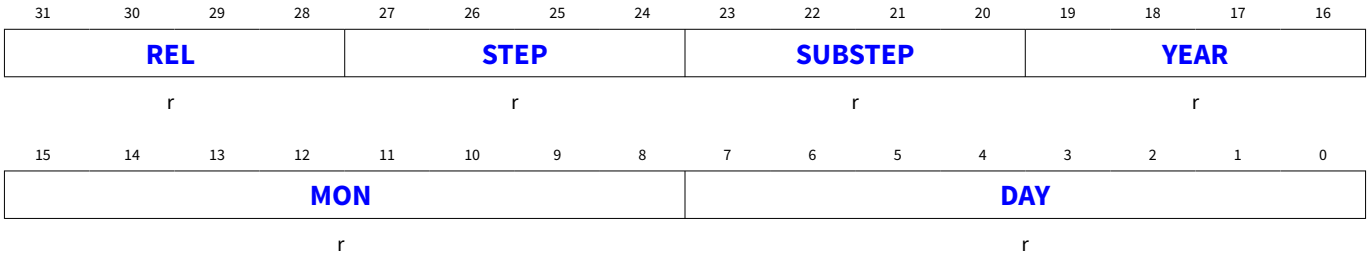
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0											EFAI				
r											rw				

Field	Bits	Type	Description
EFAI	4:0	rw	**Event FIFO Acknowledge Index** After the Host has read an element or a sequence of elements from the Tx Event FIFO it has to write the index of the last element read from Tx Event FIFO to EFAI. This will set the Tx Event FIFO Get Index Ni_TXEFS.EFGI to EFAI + 1 and update the FIFO 0 Fill Level Ni_TXEFS.EFFL.
0	31:5	r	**Reserved** Read as 0; should be written with 0.

21.7.108 Node i TSU Core Release Register

Ni_TSU_CREL (i=0-3) Offset address: $00360_H+i*400_H$

Node i TSU Core Release Register Kernel Reset value: $1008\ 1114_H$



Field	Bits	Type	Description
DAY	7:0	r	**Time Stamp Day**
MON	15:8	r	**Time Stamp Month**
YEAR	19:16	r	**Time Stamp Year**
SUBSTEP	23:20	r	**Sub-step of Core Release** One digit, BCD-coded.
STEP	27:24	r	**Step of Core Release** One digit, BCD-coded.
REL	31:28	r	**Core Release** One digit, BCD-coded.

Table 1023 Example for Coding of Revisions

Release	Step	SubStep	Year	Month	Day	Name
0	1	0	2010	03	10	Revision 0.1.0, Date 2010/03/10

21.7.109 Node i Timestamp Configuration (i=0)

Ni_TSU_TSCFG (i=0) Offset address: $00364_H+i*400_H$

Node i Timestamp Configuration Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TBPRE								0					SCP	0	TSUE
rw								r					rw	r	rw

Field	Bits	Type	Description
TSUE	0	rw	**Timestamp Unit Enable** Timestamp Unit Enable 0_B TSU disabled 1_B TSU enabled
SCP	2	rw	**Select Capturing Position** Capture timestamp at EOF or SOF 0_B Capture Timestamp at EOF 1_B Capture Timestamp at SOF
TBPRE	15:8	rw	**Timebase Prescaler** Timebase Prescaler 0x00 to 0xFF The value by which the oscillator frequency is divided for generating the timebase counter clock. Valid values for the Timebase Prescaler are 0 to 255. The actual interpretation by hardware of this value is such that one more than the value programmed is used. Affects only TSU internal timebase.
0	1, 7:3, 31:16	r	**Reserved** Read as 0; shall be written with 0.

21.7.110 Node i Timestamp Configuration (i=1-3)

Ni_TSU_TSCFG (i=1-3) Offset address: $00364_H+i*400_H$

Node i Timestamp Configuration Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TBPRE								0					SCP	TBCS	TSUE
rw								r					rw	rw	rw

Field	Bits	Type	Description
TSUE	0	rw	**Timestamp Unit Enable** Timestamp Unit Enable 0_B TSU disabled 1_B TSU enabled
TBCS	1	rw	**Timebase Counter Select** Timebase Counter Select 0_B Timestamp value captured from internal timebase counter, Ni_TSU_ATB.TB[31:0] is the internal timebase counter 1_B Timestamp value captured from external input, Ni_TSU_ATB.TB[31:0] is TSU0's output signal.Note: This only applicable for Ni_TSU_TSCFG with Ni: 1-3,while N0 does not have an external input
SCP	2	rw	**Select Capturing Position** Capture timestamp at EOF or SOF 0_B Capture Timestamp at EOF 1_B Capture Timestamp at SOF
TBPRE	15:8	rw	**Timebase Prescaler** Timebase Prescaler 0x00 to 0xFF The value by which the oscillator frequency is divided for generating the timebase counter clock. Valid values for the Timebase Prescaler are 0 to 255. The actual interpretation by hardware of this value is such that one more than the value programmed is used. Affects only TSU internal timebase.
0	7:3, 31:16	r	**Reserved** Read as 0; should be written with 0.

21.7.111 Node i Timestamp Status 1

Ni_TSU_TSS1 (i=0-3) Offset address: $00368_H+i*400_H$

Node i Timestamp Status 1 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TSL15	TSL14	TSL13	TSL12	TSL11	TSL10	TSL9	TSL8	TSL7	TSL6	TSL5	TSL4	TSL3	TSL2	TSL1	TSL0
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TSN15	TSN14	TSN13	TSN12	TSN11	TSN10	TSN9	TSN8	TSN7	TSN6	TSN5	TSN4	TSN3	TSN2	TSN1	TSN0
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
TSNz (z=0-15)	z	rh	**Timestamp New z - TSN** Timestamp New Each Timestamp register (TS0-TS15) is assigned one bit. The bits are set when a timestamp was stored in the related Timestamp register. Reading a Timestamp register resetes the related bit, if Ni_PORTCTRL.DELE is set.
TSLz (z=0-15)	z+16	rh	**Timestamp Lost z - TSL** Timestamp Lost Each Timestamp register (TS0-TS15) is assigned one bit. The bits are set when a timestamp was overwritten before it was read. Reading a Timestamp register resetes the related bit, if Ni_PORTCTRL.DELE is set.

21.7.112 Node i Timestamp Status 2

Ni_TSU_TSS2 (i=0-3) Offset address: $0036C_H+i*400_H$

Node i Timestamp Status 2 Kernel Reset value: $0000\ E000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ITBG		NTSG		0								TSP			
r		r		r								rh			

Field	Bits	Type	Description
TSP	3:0	rh	**Timestamp Counter TSP** Timestamp Pointer The Timestamp Pointer is incremented by one each time a timestamp is captured. From its maximum value (3, 7, or 15 depending on configuration) it is incremented to 0.
NTSG	13:12	r	**Number of Timestamps Generic NTSG** Number of Timestamps Read as 0x10, meaning is 16 timestamp registers
ITBG	15:14	r	**Internal Timebase and SOF select Generic ITBG** Constant value of generic parameter 00_B no SOF option, no internal timebase 01_B no SOF option, internal timebase (default) 10_B SOF option, no internal timebase 11_B SOF option, internal timebase
0	11:4, 31:16	r	**Reserved** Read as 0; should be written with 0.

21.7.113 Node i Timestamp m

Ni_TSU_TSm (i=0-3;m=0-15) Offset address: $00370_H+i*400_H+m*4$

Node i Timestamp m Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TS															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TS															
rh															

Field	Bits	Type	Description
TS	31:0	rh	**Reference Timestamp generated by TSU**

21.7.114 Node i Actual Timebase

Actual Timebase

Ni_TSU_ATB (i=0-3) Offset address: $003B0_H+i*400_H$

Node i Actual Timebase Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TB															
rwh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TB															
rwh															

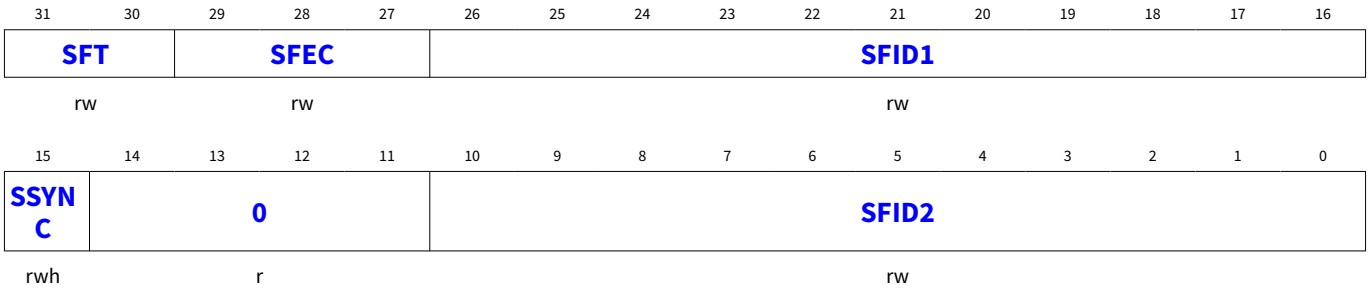
Field	Bits	Type	Description
TB	31:0	rwh	**Timebase for timestamp generation** Timebase for timestamp generation.

21.7.115 Standard Message ID Filter Element

21.7.115.1 Standard Message 0

STDMSGk_S0 (k=0-127) Offset address: $00000_H + k*4$

Standard Message 0 RAMInit value: $XXXX\ XXXX_H$



Field	Bits	Type	Description
SFID2	10:0	rw	**Standard Filter ID 2** This bit-field has a different meaning depending on the configuration of SFEC: 1) SFEC = “001”…”110” Second ID of standard ID filter element 2) SFEC = “111” Filter for Rx Buffers or for debug messages **SFID2[5:0]** defines the offset to the Rx Buffer Start Address Ni_RXBC.RBSA for storage of a matching message. **SFID2[8:6]** is used to control the filter event pins. A one at the respective bit position enables generation of a pulse at the related filter event pin with the duration of one host clock period in case the filter matches. **SFID2[10:9]** decides whether the received message is stored into an Rx Buffer or treated as message A, B, or C of the debug message sequence. 000_H Store message into an Rx Buffer … $1FF_H$ Store message into an Rx Buffer 200_H Debug Message A … $3FF_H$ Debug Message A 400_H Debug Message B … $5FF_H$ Debug Message B 600_H Debug Message C … $7FF_H$ Debug Message C
SSYNC	15	rwh	**Standard Sync Message** Only evaluated when Ni_CCCR.UTSU = '1'. 0_B Timestamping for the matching Sync message disabled. 1_B Timestamping for the matching Sync message enabled.
SFID1	26:16	rw	**Standard Filter ID 1** First ID of standard ID filter element. When filtering for Rx Buffers or for debug messages this field defines the ID of a standard message to be stored. The received identifiers must match exactly, no masking mechanism is used.

(table continues...)

(continued)

Field	Bits	Type	Description
SFEC	29:27	rw	**Standard Filter Element Configuration** All enabled filter elements are used for acceptance filtering of standard frames. Acceptance filtering stops at the first matching enabled filter element or when the end of the filter list is reached. If SFEC = “100”, “101” or “110”, a match sets interrupt flag Ni_IR.HPM and, if enabled, an interrupt is generated. In this case register Ni_HPMS is updated with the status of the priority match. 000_B Disable filter element 001_B Store in Rx FIFO 0 if filter matches 010_B Store in Rx FIFO 1 if filter matches 011_B Reject ID if filter matches 100_B Set priority if filter matches 101_B Set priority and store in FIFO 0 if filter matches 110_B Set priority and store in FIFO 1 if filter matches 111_B Store into Rx Buffer or as debug message, configuration of SFT[1:0] ignored
SFT	31:30	rw	**Standard Filter Type** ***Note**: With SFT = “11” the filter element is disabled and the acceptance filtering continues (same behavior as with SFEC = “000”)* 00_B Range filter from SF1ID to SF2ID (SF2ID ≥ SF1ID) 01_B Dual ID filter for SF1ID or SF2ID 10_B Classic filter: SF1ID = filter, SF2ID = mask 11_B Filter element disabled
0	14:11	r	**Reserved** Read as 0; should be written with 0.

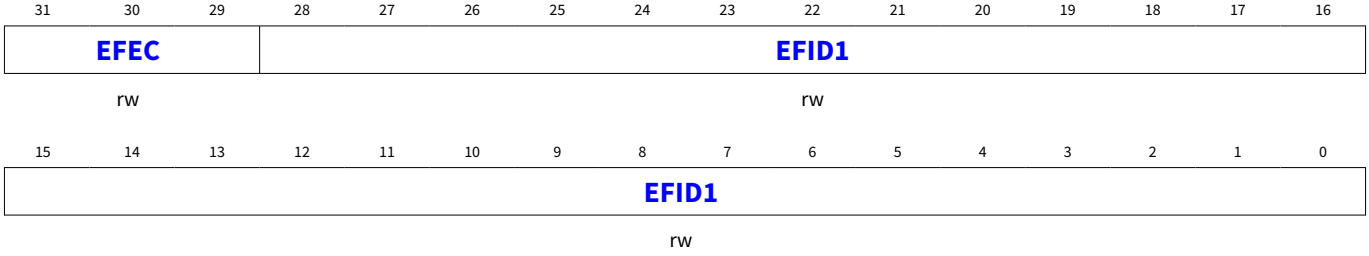
21.7.116 Extended Message ID Filter Element

21.7.116.1 Filter Element 0

See Message layout.

EXTMSGk_F0 (k=0-63) Offset address: 00000_H + k*8

Filter Element 0 RAMInit value: XXXX $XXXX_H$



Field	Bits	Type	Description
EFID1	28:0	rw	**Extended Filter ID 1** First ID of extended ID filter element. When filtering for Rx Buffers or for debug messages this field defines the ID of an extended message to be stored. The received identifiers must match exactly, only XIDAMi masking mechanism is used.
EFEC	31:29	rw	**Extended Filter Element Configuration** All enabled filter elements are used for acceptance filtering of extended frames. Acceptance filtering stops at the first matching enabled filter element or when the end of the filter list is reached. If EFEC = “100”, “101” or “110”, a match sets interrupt flag Ni_IR.HPM and, if enabled, an interrupt is generated. In this case register Ni_HPMS is updated with the status of the priority match. 000_B Disable filter element 001_B Store in Rx FIFO 0 if filter matches 010_B Store in Rx FIFO 1 if filter matches 011_B Reject ID if filter matches 100_B Set priority if filter matches 101_B Set priority and store in FIFO 0 if filter matches 110_B Set priority and store in FIFO 1 if filter matches 111_B Store into Rx Buffer or as debug message, configuration of EFT[1:0] ignored

Related information

Message RAM on page 3929

21.7.116.2 Filter Element 1

See Message layout.

EXTMSGk_F1 (k=0-63) Offset address: 00004_H+k*8

Filter Element 1 RAMInit value: $XXXX\ XXXX_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EFT		ESYNC	EFID2												
rw		rwh	rw												

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EFID2															
rw															

Field	Bits	Type	Description
EFID2	28:0	rw	**Extended Filter ID 2** This bit-field has a different meaning depending on the configuration of EFEC: 1) EFEC = “001”…”110” Second ID of extended ID filter element 2) EFEC = “111” Filter for Rx Buffers or for debug messages **EFID2[5:0]** defines the offset to the Rx Buffer Start Address Ni_RXBC.RBSA for storage of a matching message. **EFID2[8:6]** is used to control the filter event. A one at the respective bit position enables generation of a pulse at the related filter event pin with the duration of one host clock period in case the filter matches. **EFID2[10:9]** decides whether the received message is stored into an Rx Buffer or treated as message A, B, or C of the debug message sequence. 00000000_H Store message into an Rx Buffer … $000001FF_H$ Store message into an Rx Buffer 00000200_H Debug Message A … $000003FF_H$ Debug Message A 00000400_H Debug Message B … $000005FF_H$ Debug Message B 00000600_H Debug Message C … $000007FF_H$ Debug Message C
ESYNC	29	rwh	**Extended Sync Message** Only evaluated when Ni_CCCR.UTSU = '1'. 0_B Timestamping for the matching Sync message disabled. 1_B Timestamping for the matching Sync message enabled.
EFT	31:30	rw	**Extended Filter Type** 00_B Range filter from EF1ID to EF2ID (EF2ID ≥ EF1ID) 01_B Dual ID filter for EF1ID or EF2ID 10_B Classic filter: EF1ID = filter, EF2ID = mask 11_B Range filter from EF1ID to EF2ID (EF2ID ≥ EF1ID), XIDAM mask not applied

Related information

Message RAM on page 3929

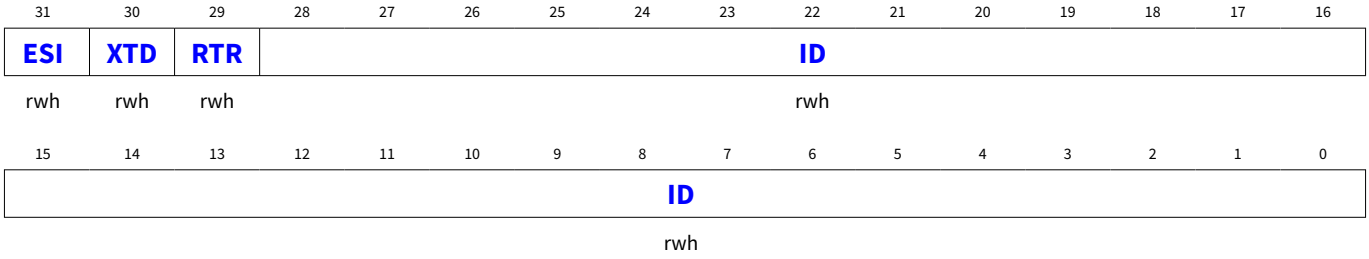
21.7.117 Rx Buffer and FIFO Element

21.7.117.1 Register 0

See Message layout.

RXMSGk_R0 (k=0-63) Offset address: $00000_H + k*48_H$

Register 0 RAMInit value: $XXXX\ XXXX_H$



Field	Bits	Type	Description
ID	28:0	rwh	**Identifier** Standard or extended identifier depending on bit XTD. A standard identifier is stored into ID[28:18].
RTR	29	rwh	**Remote Transmission Request** Signals to the Host whether the received frame is a data frame or a remote frame. ***Note**: There are no remote frames in CAN FD format. In case a CAN FD frame was received (FDF = 1'), bit RTR reflects the state of the reserved bit r1.* 0_B Received frame is a data frame 1_B Received frame is a remote frame
XTD	30	rwh	**Extended Identifier** Signals to the Host whether the received frame has a standard or extended identifier. 0_B 11-bit standard identifier 1_B 29-bit extended identifier
ESI	31	rwh	**Error State Indicator** 0_B Transmitting node is error active 1_B Transmitting node is error passive

Related information

Message RAM on page 3929

21.7.117.2 Register 1 A

See Message layout.

When no TSU is used (Ni_CCCR.UTSU = 0), RXMSGk_R1A.RXTS[15:0] holds the 16-bit timestamp generated by the M_CAN's internal timestamping logic

RXMSGk_R1A (k=0-63) Offset address: 00004_H+k*48_H

Register 1 A RAMInit value: $XXXX\ XXXX_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ANMF	FIDX							0		FDF	BRS	DLC			
rwh	rwh							r		rwh	rwh	rwh			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXTS															
rwh															

Field	Bits	Type	Description
RXTS	15:0	rwh	**Rx Timestamp** Timestamp Counter value captured on start of frame reception. Resolution depending on configuration of the Timestamp Counter Prescaler Ni_TSCC.TCP.
DLC	19:16	rwh	**Data Length Code** 0_H CAN + CAN FD: received frame has 0 data bytes … 8_H CAN + CAN FD: received frame has 8 data bytes 9_H CAN FD: received frame has 12 (9*4-24) data bytes CAN: received frame has 8 data bytes … C_H CAN FD: received frame has 24 (12*4-24) data bytes CAN: received frame has 8 data bytes D_H CAN FD: received frame has 32 (13*16-176) data bytes CAN: received frame has 8 data bytes … F_H CAN FD: received frame has 64 (15*16-176) data bytes CAN: received frame has 8 data bytes
BRS	20	rwh	**Bit Rate Switch** 0_B Frame received without bit rate switching 1_B Frame received with bit rate switching
FDF	21	rwh	**Frame Data Format** 0_B Standard frame format 1_B CAN FD frame format (new DLC-coding and CRC)

(table continues...)

(continued)

Field	Bits	Type	Description
FIDX	30:24	rwh	**Filter Index** 00_H Index of matching Rx acceptance filter element (invalid if ANMF = '1'). Range is 0 to SIDFC.LSS - 1 resp. XIDFC.LSE - 1. … $7F_H$ Index of matching Rx acceptance filter element (invalid if ANMF = '1'). Range is 0 to SIDFC.LSS - 1 resp. XIDFC.LSE - 1.
ANMF	31	rwh	**Accepted Non-matching Frame** Acceptance of non-matching frames may be enabled via GFC.ANFS and GFC.ANFE. 0_B Received frame matching filter index FIDX 1_B Received frame did not match any Rx filter element
0	23:22	r	**Reserved** Read as 0; should be written with 0.

Related information

Message RAM on page 3929

21.7.117.3 Register 1 B

See Message layout.

When a TSU is used (CCCR.UTSU = 1) and when bit SSYNC/ESYNC of the matching filter element is set, R1B.TSC = 1 and R1B.RXTSP[3:0] holds the number of the TSU's Timestamp register which holds the 32-bit timestamp captured by the TSU. Else R1B.TSC = 0 and R1B.RXTSP[3:0] is not valid

RXMSGk_R1B (k=0-63) Offset address: 00004_H+k*48_H

Register 1 B RAMInit value: $XXXX\ XXXX_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ANMF	FIDX							0		FDF	BRS	DLC			
rwh	rwh							r		rwh	rwh	rwh			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0											TSC	RXTSP			
r											rwh	rwh			

Field	Bits	Type	Description
RXTSP	3:0	rwh	**Rx Timestamp Pointer** Number of TSU Time Stamp register (TS0..15) where the related timestamp is stored.
TSC	4	rwh	**Timestamp Captured** 0_B No timestamp captured 1_B Timestamp captured and stored in TSU Timestamp register referenced by R1B.RXTSP

(table continues...)

(continued)

Field	Bits	Type	Description
DLC	19:16	rwh	**Data Length Code** 0_H CAN + CAN FD: received frame has 0 data bytes … 8_H CAN + CAN FD: received frame has 8 data bytes 9_H CAN FD: received frame has 12 (9*4-24) data bytes CAN: received frame has 8 data bytes … C_H CAN FD: received frame has 24 (12*4-24) data bytes CAN: received frame has 8 data bytes D_H CAN FD: received frame has 32 (13*16-176) data bytes CAN: received frame has 8 data bytes … F_H CAN FD: received frame has 64 (15*16-176) data bytes CAN: received frame has 8 data bytes
BRS	20	rwh	**Bit Rate Switch** 0_B Frame received without bit rate switching 1_B Frame received with bit rate switching
FDF	21	rwh	**Frame Data Format** 0_B Standard frame format 1_B CAN FD frame format (new DLC-coding and CRC)
FIDX	30:24	rwh	**Filter Index** 00_H Index of matching Rx acceptance filter element (invalid if ANMF = '1'). Range is 0 to Ni_SIDFC.LSS - 1 resp. Ni_XIDFC.LSE - 1. … $7F_H$ Index of matching Rx acceptance filter element (invalid if ANMF = '1'). Range is 0 to Ni_SIDFC.LSS - 1 resp. Ni_XIDFC.LSE - 1.
ANMF	31	rwh	**Accepted Non-matching Frame** Acceptance of non-matching frames may be enabled via Ni_GFC.ANFS and Ni_GFC.ANFE. 0_B Received frame matching filter index FIDX 1_B Received frame did not match any Rx filter element
0	15:5, 23:22	r	**Reserved** Read as 0; should be written with 0.

Related information

Message RAM on page 3929

21.7.117.4 Data Byte m

RXMSGk_DBm (k=0-63;m=0-63) Offset address: 00008_H+k*48_H+m

Data Byte m RAMInit value: XX_H

7	6	5	4	3	2	1	0
DB							
rwh							

Field	Bits	Type	Description
DB	7:0	rwh	**Data Byte m**

21.7.118 Tx Event FIFO Element

21.7.118.1 Event 0

See Message layout.

TXEVENTk_E0 (k=0-31) Offset address: 00000_H + k*8

Event 0 RAMInit value: XXXX $XXXX_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ESI	XTD	RTR	ID												
rwh	rwh	rwh	rwh												

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ID															
rwh															

Field	Bits	Type	Description
ID	28:0	rwh	**Identifier** Standard or extended identifier depending on bit XTD. A standard identifier is stored into ID[28:18].
RTR	29	rwh	**Remote Transmission Request** 0_B Data frame transmitted 1_B Remote frame transmitted
XTD	30	rwh	**Extended Identifier** 0_B 11-bit standard identifier 1_B 29-bit extended identifier
ESI	31	rwh	**Error State Indicator** 0_B Transmitting node is error active 1_B Transmitting node is error passive

Related information

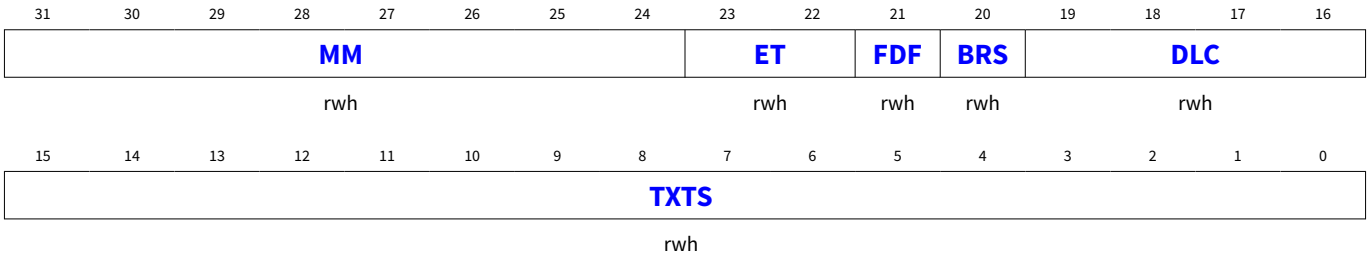
Message RAM on page 3929

21.7.118.2 Event 1A

See Message layout.

TXEVENTk_E1A (k=0-31) Offset address: 00004_H+k*8

Event 1A RAMInit value: XXXX $XXXX_H$



Field	Bits	Type	Description
TXTS	15:0	rwh	**Tx Timestamp** Timestamp Counter value captured on start of frame transmission. Resolution depending on configuration of the Timestamp Counter Prescaler Ni_TSCC.TCP.
DLC	19:16	rwh	**Data Length Code** 0_H CAN + CAN FD: received frame has 0 data bytes … 8_H CAN + CAN FD: received frame has 8 data bytes 9_H CAN FD: received frame has 12 (9*4-24) data bytes CAN: received frame has 8 data bytes … C_H CAN FD: received frame has 24 (12*4-24) data bytes CAN: received frame has 8 data bytes D_H CAN FD: received frame has 32 (13*16-176) data bytes CAN: received frame has 8 data bytes … F_H CAN FD: received frame has 64 (15*16-176) data bytes CAN: received frame has 8 data bytes
BRS	20	rwh	**Bit Rate Switch** 0_B Frame transmitted without bit rate switching 1_B Frame transmitted with bit rate switching
FDF	21	rwh	**FD Format** 0_B Standard frame format 1_B CAN FD frame format (new DLC-coding and CRC)
ET	23:22	rwh	**Event Type** 00_B Reserved 01_B Tx event 10_B Transmission in spite of cancellation (always set for transmissions in DAR mode) 11_B Reserved
MM	31:24	rwh	**Message Marker** Copied from Tx Buffer into Tx Event FIFO element for identification of Tx message status.

Related information

Message RAM on page 3929

21.7.118.3 Event 1B

See Message layout.

TXEVENTk_E1B (k=0-31) Offset address: 00004_H+k*8

Event 1B RAMInit value: $XXXX\ XXXX_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MM0								ET		FDF	BRS	DLC			
rwh								rwh		rwh	rwh	rwh			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MM1								0			TSC	TXTSP			
rw								r			rwh	rwh			

Field	Bits	Type	Description
TXTSP	3:0	rwh	**Tx Timestamp Pointer** Number of TSU Time Stamp register (TS0..15) where the related timestamp is stored.
TSC	4	rwh	**Timestamp Captured** 0_B 0 No timestamp captured 1_B 1Timestamp captured and stored in TSU Timestamp register referenced by TXTSP
MM1	15:8	rw	**Message Marker** High byte of Wide Message Marker, written by CPU during Tx Buffer configuration. Copied into Tx Event FIFO element for identification of Tx message status.
DLC	19:16	rwh	**Data Length Code** 0_H CAN + CAN FD: received frame has 0 data bytes … 8_H CAN + CAN FD: received frame has 8 data bytes 9_H CAN FD: received frame has 12 (9*4-24) data bytes CAN: received frame has 8 data bytes … C_H CAN FD: received frame has 24 (12*4-24) data bytes CAN: received frame has 8 data bytes D_H CAN FD: received frame has 32 (13*16-176) data bytes CAN: received frame has 8 data bytes … F_H CAN FD: received frame has 64 (15*16-176) data bytes CAN: received frame has 8 data bytes
BRS	20	rwh	**Bit Rate Switch** 0_B Frame transmitted without bit rate switching 1_B Frame transmitted with bit rate switching
FDF	21	rwh	**FD Format** 0_B Standard frame format 1_B CAN FD frame format (new DLC-coding and CRC)

(table continues...)

(continued)

Field	Bits	Type	Description
ET	23:22	rwh	**Event Type** 00_B Reserved 01_B Tx event 10_B Transmission in spite of cancellation (always set for transmissions in DAR mode) 11_B Reserved
MM0	31:24	rwh	**Message Marker** Copied from Tx Buffer into Tx Event FIFO element for identification of Tx message status.
0	7:5	r	**Reserved** Read as 0; should be written with 0.

Related information

Message RAM on page 3929

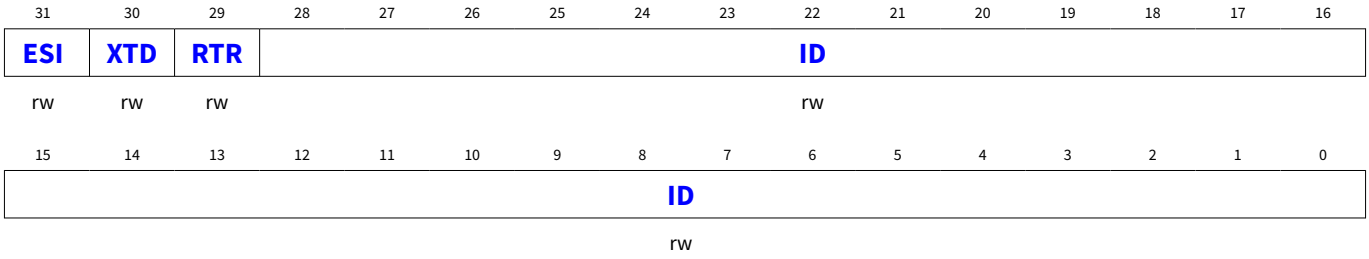
21.7.119 Tx Buffer Element

21.7.119.1 Transmit Buffer 0

See Message layout.

TXMSGk_T0 (k=0-31) Offset address: $00000_H + k*48_H$

Transmit Buffer 0 RAMInit value: $XXXX\ XXXX_H$



Field	Bits	Type	Description
ID	28:0	rw	**Identifier** Standard or extended identifier depending on bit XTD. A standard identifier has to be written to ID[28:18].
RTR	29	rw	**Remote Transmission Request** ***Note**: When RTR = 1, the M_CAN transmits a remote frame according to ISO11898-1, even if Ni_CCCR.FDOE enables the transmission in CAN FD format.* 0_B Transmit data frame 1_B Transmit remote frame

(table continues...)

(continued)

Field	Bits	Type	Description
XTD	30	rw	**Extended Identifier** 0_B 11-bit standard identifier 1_B 29-bit extended identifier
ESI	31	rw	**Error State Indicator** ***Note**:* *The ESI bit of the transmit buffer is OR'ed with the error passive flag to decide the value of the ESI bit in the transmitted FD frame. As required by the CAN FD protocol specification, an error active node may optionally transmit the ESI bit recessive, but an error passive node will always transmit the ESI bit recessive.* 0_B ESI bit in CAN FD format depends only on error passive flag 1_B ESI bit in CAN FD format transmitted recessive

Related information

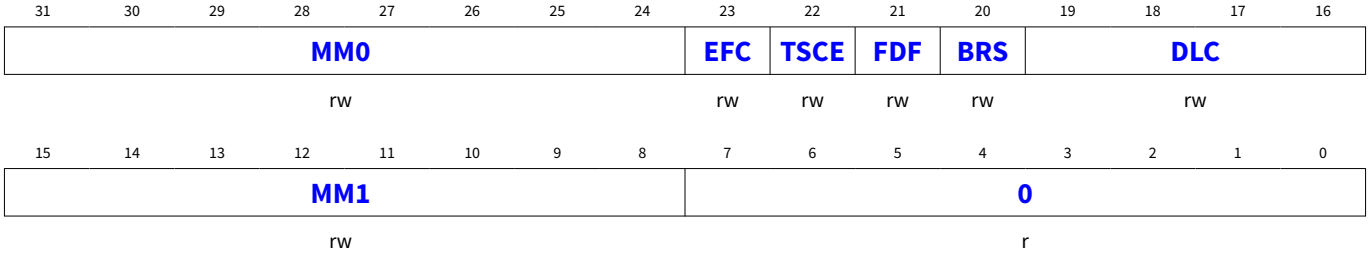
Message RAM on page 3929

21.7.119.2 Transmit Buffer 1

See Message layout.

TXMSGk_T1 (k=0-31) Offset address: 00004_H+k*48_H

Transmit Buffer 1 RAMInit value: $XXXX\ XXXX_H$



Field	Bits	Type	Description
MM1	15:8	rw	**Message Marker** High byte of Wide Message Marker, written by CPU during Tx Buffer configuration. Copied into Tx Event FIFO element for identification of Tx message status. Available only when Ni_CCCR.WMM = '1' or when Ni_CCCR.UTSU = '1'.

(table continues...)

(continued)

Field	Bits	Type	Description
DLC	19:16	rw	**Data Length Code** 0_H CAN + CAN FD: received frame has 0 data bytes … 8_H CAN + CAN FD: received frame has 8 data bytes 9_H CAN FD: received frame has 12 (9*4-24) data bytes CAN: received frame has 8 data bytes … C_H CAN FD: received frame has 24 (12*4-24) data bytes CAN: received frame has 8 data bytes D_H CAN FD: received frame has 32 (13*16-176) data bytes CAN: received frame has 8 data bytes … F_H CAN FD: received frame has 64 (15*16-176) data bytes CAN: received frame has 8 data bytes
BRS	20	rw	**Bit Rate Switching** ***Note**:* *Bits ESI, FDF, and BRS are only evaluated when CAN FD operation is enabled Ni_CCCR.FDOE = 1'. Bit BRS is only evaluated when in addition Ni_CCCR.BRSE = '1'.* 0_B CAN FD frames transmitted without bit rate switching 1_B CAN FD frames transmitted with bit rate switching
FDF	21	rw	**FD Format** 0_B Frame transmitted in Classical CAN format 1_B Frame transmitted in CAN FD format
TSCE	22	rw	**Time Stamp Captuer Enable for TSU** Only available when Ni_CCCR.UTSU = '1'. When this bit is set and the message is transmitted, the transmission of a Sync message is signaled to the Timestamping Unit (TSU) connected to the M_CAN which captures the timestamp. 0_B Time Stamp Capture disabled 1_B Time Stamp Capture enabled
EFC	23	rw	**Event FIFO Control** 0_B Do not store Tx events 1_B Store Tx events
MM0	31:24	rw	**Message Marker** Written by CPU during Tx Buffer configuration. Copied into Tx Event FIFO element for identification of Tx message status.
0	7:0	r	**Reserved** Read as 0; should be written with 0.

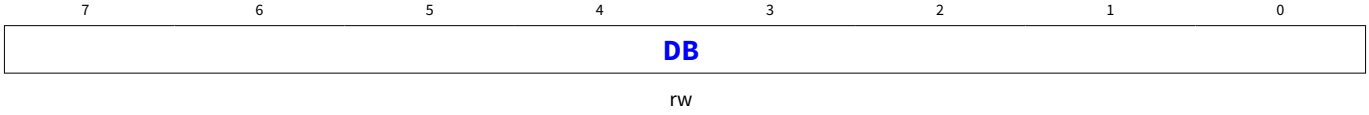
Related information

Message RAM on page 3929

21.7.119.3 Data Byte m

TXMSGk_DBm (k=0-31;m=0-63) Offset address: 00008_H+k*48_H+m

Data Byte m RAMInit value: XX_H



Field	Bits	Type	Description
DB	7:0	rw	**Data Byte m**

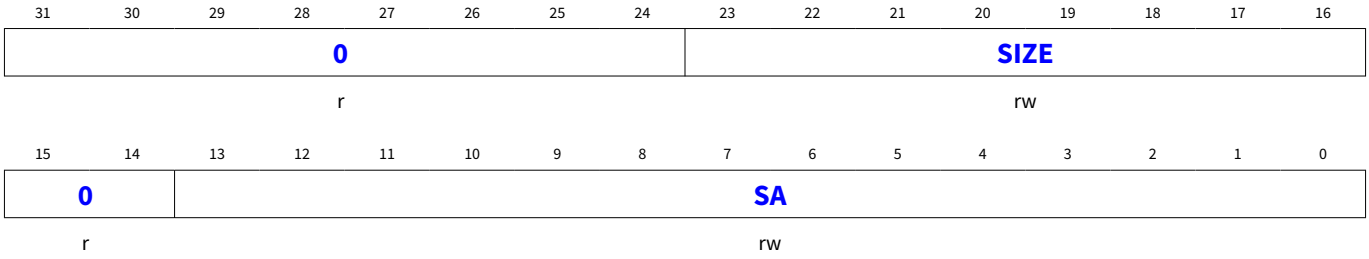
21.7.120 CRE configuration table

21.7.120.1 STD ID Routing table parameters

This register defines the Start Address and size of the Standard ID Routing Table
SA defines the start address of the Standard ID Routing Table (32 bit Word Address).
SIZE defines the number of Routing Rules contained in the Standard ID Routing Table.
0 - The Routing Table is disabled
1 till 128 - Number of Routing Rules
others - Reserved
Note: Ensure the SA does not overlap with the Host Buffer contents. Refer CRE RAM structure

CRE_STD_RT_PARAM Offset address: 00000_H

STD ID Routing table parameters RAMInit value: $XXXX\ XXXX_H$



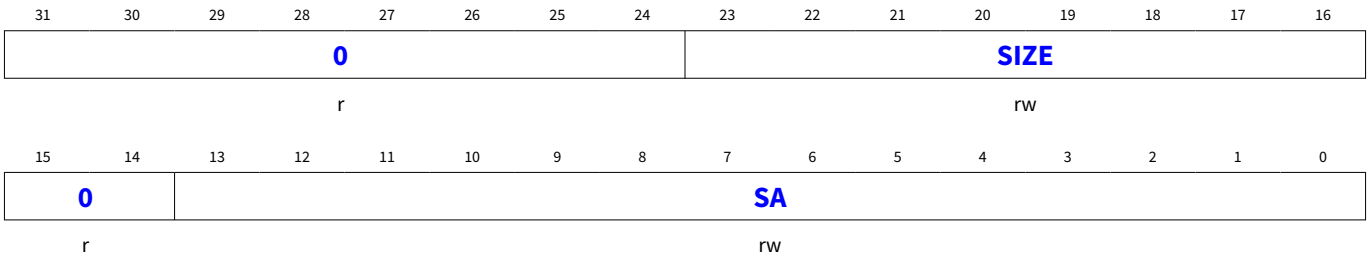
Field	Bits	Type	Description
SA	13:0	rw	**Start address of the table**
SIZE	23:16	rw	**Size of the table**
0	15:14, 31:24	r	**Reserved** Read as 0; should be written with 0.

21.7.120.2 XTD ID Routing table parameters

This register defines the Start Address and size of the Extended ID Routing Table
SA defines the start address of the Extended ID Routing Table (32 bit Word Address).
SIZE defines the number of Routing Rules contained in the Extended ID Routing Table.
0 - The Routing Table is disabled
1 till 64 - Number of Routing Rules
others - Reserved
Note: Ensure the SA does not overlap with the Host Buffer contents. Refer CRE RAM structure

CRE_XTD_RT_PARAM Offset address: 00004_H

XTD ID Routing table parameters RAMInit value: $XXXX\ XXXX_H$



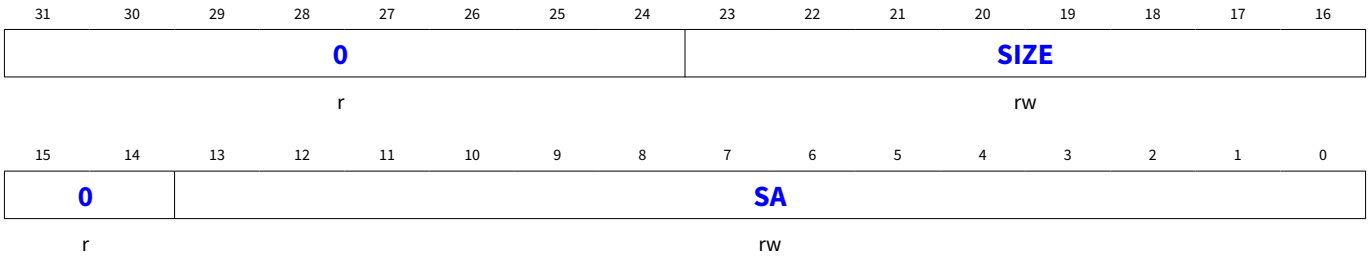
Field	Bits	Type	Description
SA	13:0	rw	**Start address of the table**
SIZE	23:16	rw	**Size of the table**
0	15:14, 31:24	r	**Reserved** Read as 0; should be written with 0.

21.7.120.3 STD ID Frame rate measure table parameters

This register defines the Start Address and size of the Standard ID Frame rate measure Table
SA defines the start address of the Standard ID Frame rate measure Table (32 bit Word Address).
SIZE defines the number of Frame rate measures contained in the Standard ID Frame rate measure Table.
0 - The Frame rate measure Table is disabled
1 till 128 - Number of Frame rate measures
others - Reserved

CRE_STD_FRT_PARAM Offset address: 00008_H

STD ID Frame rate measure table parameters RAMInit value: $XXXX\ XXXX_H$



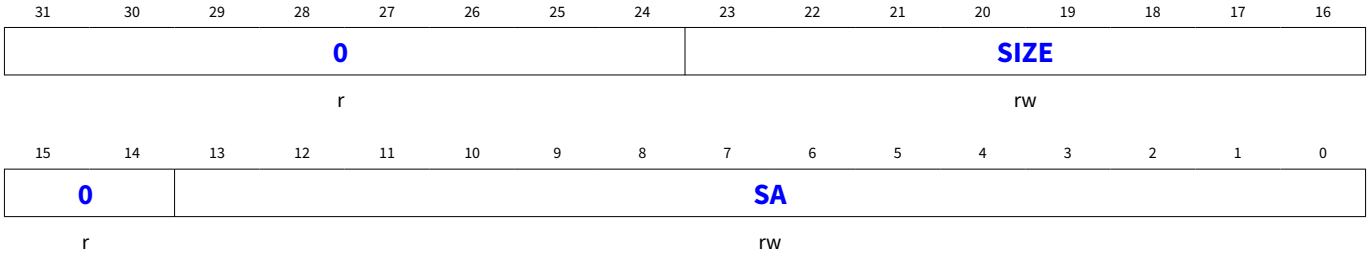
Field	Bits	Type	Description
SA	13:0	rw	**Start address of the table**
SIZE	23:16	rw	**Size of the table**
0	15:14, 31:24	r	**Reserved** Read as 0; should be written with 0.

21.7.120.4 XTD ID Frame rate measure table parameters

This register defines the Start Address and size of the Extended ID Frame rate measure Table
SA defines the start address of the Extended ID Frame rate measure Table (32 bit Word Address).
SIZE defines the number of Frame rate measures contained in the Extended ID Frame rate measure Table.
0 - The Frame rate measure Table is disabled
1 till 64 - Number of Frame rate measures
others - Reserved

CRE_XTD_FRT_PARAM Offset address: $0000C_H$

XTD ID Frame rate measure table parameters RAMInit value: $XXXX\ XXXX_H$



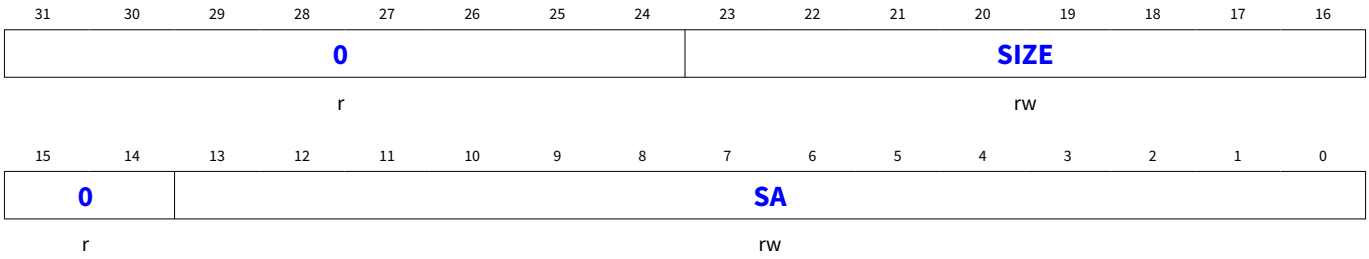
Field	Bits	Type	Description
SA	13:0	rw	**Start address of the table**
SIZE	23:16	rw	**Size of the table**
0	15:14, 31:24	r	**Reserved** Read as 0; should be written with 0.

21.7.120.5 STD ID Timestamp database parameters

This register defines the Start Address and size of the Standard ID Timestamp database
SA defines the start address of the Standard ID Timestamp database (32 bit Word Address).
SIZE defines the number of timestamps contained in the Standard ID Timestamp database.
0 - The Timestamp database is disabled
1 till 128 - Number of timestamps
others - Reserved

CRE_STD_TSD_PARAM Offset address: 00010_H

STD ID Timestamp database parameters RAMInit value: $XXXX\ XXXX_H$



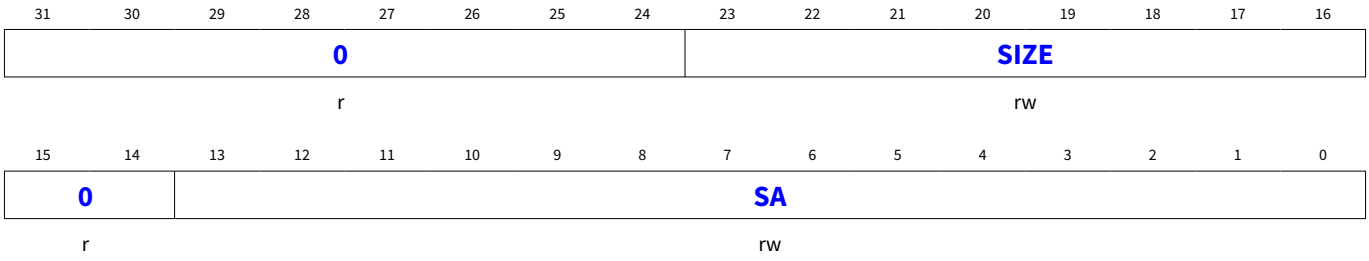
Field	Bits	Type	Description
SA	13:0	rw	**Start address of the table**
SIZE	23:16	rw	**Size of the table**
0	15:14, 31:24	r	**Reserved** Read as 0; should be written with 0.

21.7.120.6 XTD ID Timestamp database parameters

This register defines the Start Address and size of the Extended ID Timestamp database
SA defines the start address of the Extended ID Timestamp database (32 bit Word Address).
SIZE defines the number of timestamps contained in the Extended ID Timestamp database.
0 - The Timestamp database is disabled
1 till 64 - Number of timestamps
others - Reserved

CRE_XTD_TSD_PARAM Offset address: 00014_H

XTD ID Timestamp database parameters RAMInit value: $XXXX\ XXXX_H$

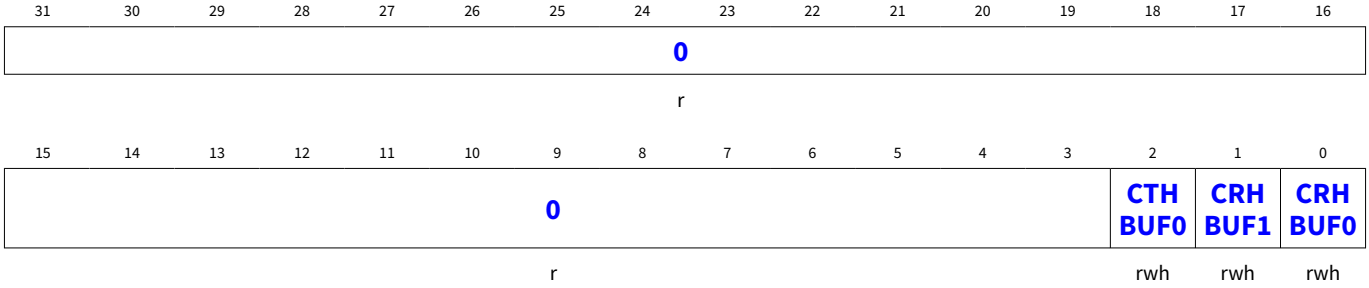


Field	Bits	Type	Description
SA	13:0	rw	**Start address of the table**
SIZE	23:16	rw	**Size of the table**
0	15:14, 31:24	r	**Reserved** Read as 0; should be written with 0.

21.7.120.7 CRE abort sequence register

CRE_ABORT_SEQ Offset address: 00018_H

CRE abort sequence register RAMInit value: $XXXX\ XXXX_H$



Field	Bits	Type	Description
CRHBUF0	0	rwh	**Cancel Rx Host Buffer 0 sequence** Writing 1 to this bit cancels the ongoing Rx Host Buffer 0 sequence. Writing 0 has no effect. The software shall not write to this register
CRHBUF1	1	rwh	**Cancel Rx Host Buffer 1 sequence** Writing 1 to this bit cancels the ongoing Rx Host Buffer 1 sequence. Writing 0 has no effect
CTHBUF0	2	rwh	**Cancel Tx Host Buffer 0 sequence** Writing 1 to this bit cancels the ongoing Tx Host Buffer 0 sequence. Writing 0 has no effect
0	31:3	r	**Reserved**

21.7.121 CRE Receive Host Buffers

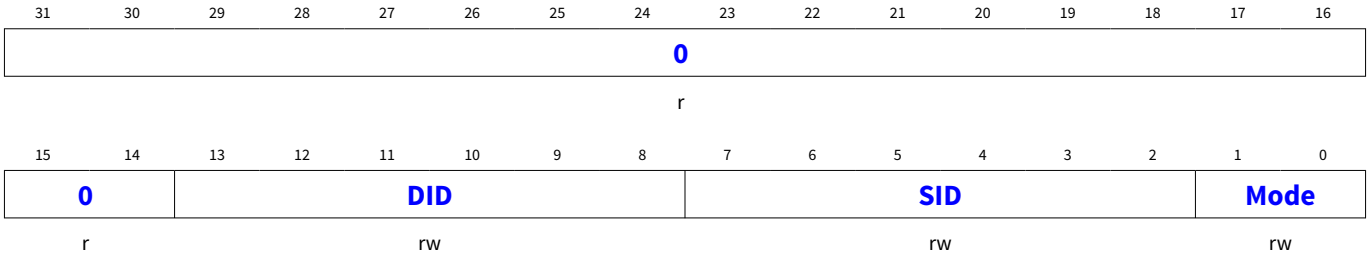
21.7.121.1 Uni-cast Routing Header k

RHBUFk_UCRH (k=0-1)

Offset address: 00020_H+k*60_H

Uni-cast Routing Header k

RAMInit value: $XXXX\ XXXX_H$



Field	Bits	Type	Description
Mode	1:0	rw	**Defines type of routing** It defines the type of the routing rule other - Reserved, considered as Uni-cast Rule 00_B Uni-cast Rule 01_B Multi-cast Rule 10_B PDU Routing Rule 11_B Reserved Considered as uni-cast Rule

(table continues...)

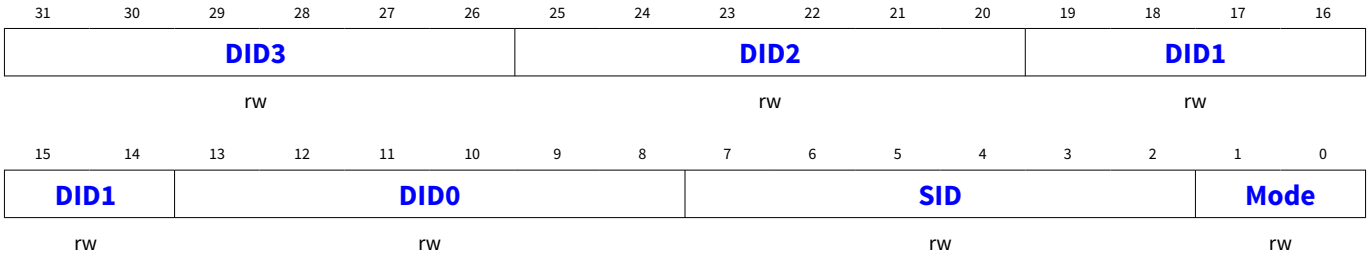
(continued)

Field	Bits	Type	Description
SID	7:2	rw	**ID of Source node receiving CAN Frame** This bit-field indicates the Source node at which the CAN frame is received. The Source nodes are referenced using a unique ID as given in the Routing ID overview table.
DID	13:8	rw	**Destination of Received CAN Frame** This bit-field indicates the Destination to which the received CAN frame has to be transferred to. The Destinations are referenced using a unique ID as given in the Routing ID overview table.
0	31:14	r	**Reserved** Read as 0; should be written with 0

21.7.121.2 Multi-cast Routing Header k

RHBUFk_MCRH (k=0-1) Offset address: 00020_H+k*60_H

Multi-cast Routing Header k RAMInit value: $XXXX\ XXXX_H$



Field	Bits	Type	Description
Mode	1:0	rw	**Defines type of routing** It defines the type of the routing rule 0 - Uni-cast Rule 1 - Multi-cast Rule other - Reserved, considered as Uni-cast Rule
SID	7:2	rw	**ID of Source node receiving CAN Frame** This bit-field indicates the Source node at which the CAN frame is received. The Source node are referenced using a unique ID as given in the Routing ID overview table.
DID0	13:8	rw	**1st Destination of Recieved CAN Frame** This bit-field indicates the 1st Destination to which the received CAN frame has to be transferred to. The Destinations are referenced using a unique ID as given in the Routing ID overview table.
DID1	19:14	rw	**2nd Destination of Recieved CAN Frame** This bit-field indicates the 2nd Destination to which the received CAN frame has to be transferred to. The Destinations are referenced using a unique ID as given in the Routing ID overview table.

(table continues...)

(continued)

Field	Bits	Type	Description
DID2	25:20	rw	**3rd Destination of Recieved CAN Frame** This bit-field indicates the 3rd Destination to which the received CAN frame has to be transferred to. The Destinations are referenced using a unique ID as given in the Routing ID overview table.
DID3	31:26	rw	**4th Destination of Recieved CAN Frame** This bit-field indicates the 4th Destination to which the received CAN frame has to be transferred to. The Destinations are referenced using a unique ID as given in the Routing ID overview table.

21.7.121.3 Timing Header k Intrusion Detection Information

RHBUFk_THEAD_INTRD (k=0-1) Offset address: 00024_H+k*60_H

Timing Header k Intrusion Detection Information RAMInit value: $XXXX\ XXXX_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TSL	IAMSTAT	TSCLEN	TSC	IAM											
rwh	rwh	rwh	rwh	rwh											

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IAM															
rwh															

Field	Bits	Type	Description
IAM	27:0	rwh	**Inter Arrival Measure** Inter arrival mesaure is the difference between the latest timestamp RXTS of the received frame and the previous timestamp obtained from the timestamp database STSDk_RTS.TS or XTSDk_RTS.TS. The IAM is valid only when it is non-zero. IAM is zero in case of unavailabilty of reference timestamp RTS
TSC	28	rwh	**Timestamp Captured** External timestamp from TSU. 0 : No timestamp captured or timestamp invalid 1: Valid timestamp captured and stored in RHBUFk_THEAD_RXTS
TSCLEN	29	rwh	**Timestamp Captured Length** Length of the timestamp stored in RHBUFk_THEAD_RXTS 0 : 16-bit timestamp 1 : 32-bit timestamp (from TSU)
IAMSTAT	30	rwh	**Inter Arrival Measure Status** Availablity of Inter Arrival Measure for Intrusion detection 0 : No reference timestamp (RTS) avaialble to calculate IAM (in case of first frame received, overrun, no entry in the RTS DB). 1 : RTS available to calculate IAM

(table continues...)

(continued)

Field	Bits	Type	Description
TSL	31	rwh	**Timestamp lost** Timestamp register in the TSU was overwritten before it was read. The IAM is calculated using the latest timestamp fetched. 0 : No timestamp lost 1: Timestamp lost

21.7.121.4 Timing Header k Rx Timestamp

16-bit Internal or 32-bit External Timestamp.

RHBUFk_THEAD_RXTS (k=0-1) Offset address: 00028_H+k*60_H

Timing Header k Rx Timestamp RAMInit value: $XXXX\ XXXX_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RXTS															
rwh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXTS															
rwh															

Field	Bits	Type	Description
RXTS	31:0	rwh	**External Timestamp** 16-bit internal or 32 bit external timestamp. Ni_INTRD.TSCLEN specifies the length of the timestamp

21.7.121.5 CRE computed CRC

CRC computed by the CRE over the safety critical CAN payload of the CAN frame stored in the RHBUF. The CRC is computed by the CRE over the CAN frame (R0, R1 and DBm) and the DID of the frame

RHBUFk_CRC (k=0-1) Offset address: $0002C_H+k*60_H$

CRE computed CRC RAMInit value: $XXXX\ XXXX_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CRC															
rwh															

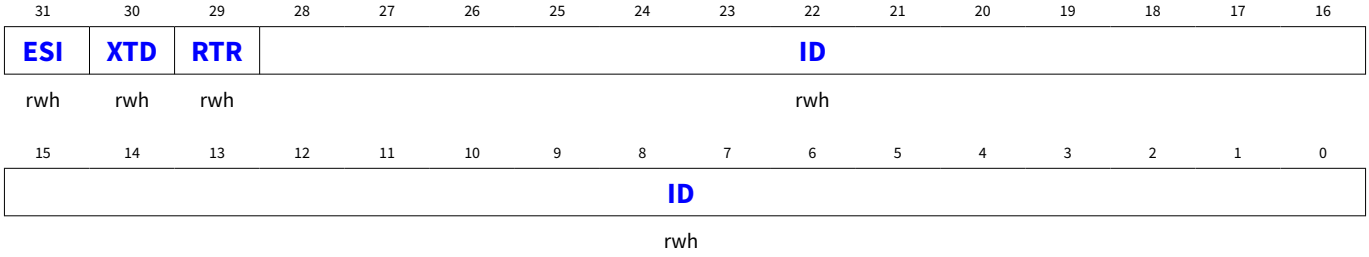
Field	Bits	Type	Description
CRC	15:0	rwh	**CRC** CRC computed by CRE
0	31:16	r	**Reserved**

21.7.121.6 RHBUF k Register 0

See Message layout.

RHBUFk_R0 (k=0-1) Offset address: 00030_H+k*60_H

RHBUF k Register 0 RAMInit value: $XXXX\ XXXX_H$



Field	Bits	Type	Description
ID	28:0	rwh	**Identifier** Standard or extended identifier depending on bit XTD. A standard identifier is stored into ID[28:18].
RTR	29	rwh	**Remote Transmission Request** Signals to the Host whether the received frame is a data frame or a remote frame. ***Note**: There are no remote frames in CAN FD format. In case a CAN FD frame was received (FDF = 1'), bit RTR reflects the state of the reserved bit r1.* 0_B Received frame is a data frame 1_B Received frame is a remote frame
XTD	30	rwh	**Extended Identifier** Signals to the Host whether the received frame has a standard or extended identifier. 0_B 11-bit standard identifier 1_B 29-bit extended identifier
ESI	31	rwh	**Error State Indicator** 0_B Transmitting node is error active 1_B Transmitting node is error passive

Related information

Functional description on page 3962

21.7.121.7 Long PDU header 1

RHBUFk_LPDUH1 (k=0-1) Offset address: 00030_H+k*60_H

Long PDU header 1 RAMInit value: $XXXX\ XXXX_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ID															
rwh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ID															
rwh															

Field	Bits	Type	Description
ID	31:0	rwh	**PDU ID** • Type 3 PDU ID = Source ID + CAN ID (11 bits) + Metadata (16 bits) • Type 4 PDU ID = Source ID + CAN ID (27 bits) • Type 5 PDU ID = CAN ID (29 bits)

21.7.121.8 RHBUF k Register 1

See Message layout.

RHBUFk_R1 (k=0-1) Offset address: 00034_H+k*60_H

RHBUF k Register 1 RAMInit value: $XXXX\ XXXX_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ANMF	FIDX							0		FDF	BRS	DLC			
rwh	rwh							r		rwh	rwh	rwh			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXTS															
rwh															

Field	Bits	Type	Description
RXTS	15:0	rwh	**Rx Timestamp** Timestamp Counter value captured on start of frame reception. Resolution depends on configuration of the Timestamp Counter Prescaler Ni_TSCC.TCP.

(table continues...)

(continued)

Field	Bits	Type	Description
DLC	19:16	rwh	**Data Length Code** 0_H CAN + CAN FD: received frame has 0 data bytes … 8_H CAN + CAN FD: received frame has 8 data bytes 9_H CAN FD: received frame has 12 (9*4-24) data bytes CAN: received frame has 8 data bytes … C_H CAN FD: received frame has 24 (12*4-24) data bytes CAN: received frame has 8 data bytes D_H CAN FD: received frame has 32 (13*16-176) data bytes CAN: received frame has 8 data bytes … F_H CAN FD: received frame has 64 (15*16-176) data bytes CAN: received frame has 8 data bytes
BRS	20	rwh	**Bit Rate Switch** 0_B Frame received without bit rate switching 1_B Frame received with bit rate switching
FDF	21	rwh	**Frame Data Format** 0_B Standard frame format 1_B CAN FD frame format (new DLC-coding and CRC)
FIDX	30:24	rwh	**Filter Index** 00_H Index of matching Rx acceptance filter element (invalid if ANMF = '1'). Range is 0 to Ni_SIDFC.LSS - 1 resp. Ni_XIDFC.LSE - 1. … $7F_H$ Index of matching Rx acceptance filter element (invalid if ANMF = '1'). Range is 0 to Ni_SIDFC.LSS - 1 resp. Ni_XIDFC.LSE - 1.
ANMF	31	rwh	**Accepted Non-matching Frame** Acceptance of non-matching frames may be enabled via Ni_GFC.ANFS and Ni_GFC.ANFE. 0_B Received frame matching filter index FIDX 1_B Received frame did not match any Rx filter element
0	23:22	r	**Reserved** Read as 0; should be written with 0.

Related information

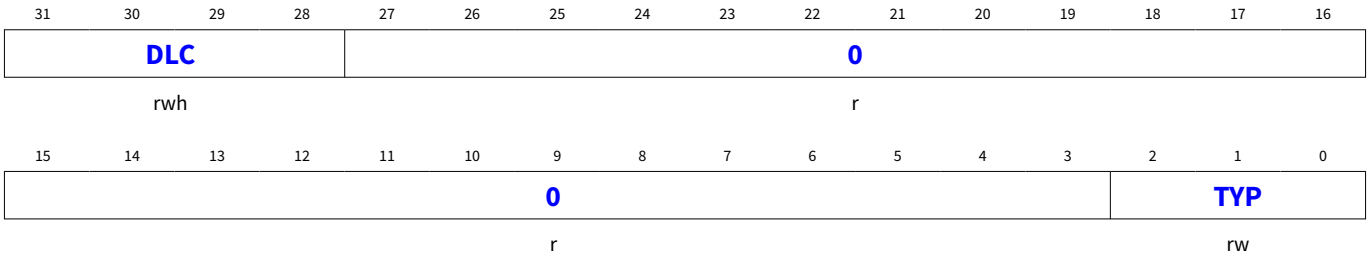
Functional description on page 3962

21.7.121.9 Long PDU header 0

Type 3, Type 4 and Type 5 long PDU header PDU ID

RHBUFk_LPDUH0 (k=0-1) Offset address: 00034_H+k*60_H

Long PDU header 0 RAMInit value: $XXXX\ XXXX_H$



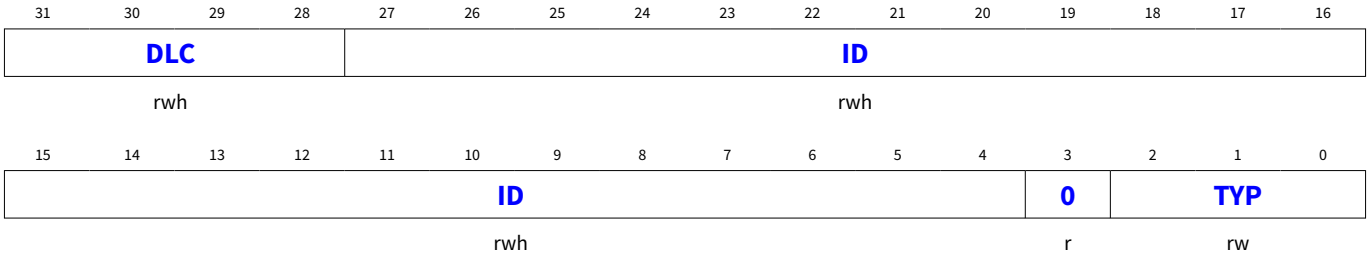
Field	Bits	Type	Description
TYP	2:0	rw	**PDU Header type** Indicates the Long PDU header type
DLC	31:28	rwh	**Data Length Code** 0_H CAN + CAN FD: received frame has 0 data bytes … 8_H CAN + CAN FD: received frame has 8 data bytes 9_H CAN FD: received frame has 12 (9*4-24) data bytes CAN: received frame has 8 data bytes … C_H CAN FD: received frame has 24 (12*4-24) data bytes CAN: received frame has 8 data bytes D_H CAN FD: received frame has 32 (13*16-176) data bytes CAN: received frame has 8 data bytes … F_H CAN FD: received frame has 64 (15*16-176) data bytes CAN: received frame has 8 data bytes
0	27:3	r	**Reserved**

21.7.121.10 Short PDU header

Type 1 and Type 2 short PDU header. All PDU headers consist of a unique PDU ID and the DLC

RHBUFk_SPDUH (k=0-1) Offset address: 00034_H+k*60_H

Short PDU header RAMInit value: XXXX $XXXX_H$



Field	Bits	Type	Description
TYP	2:0	rw	**PDU Header type** Indicates the Short PDU header type

(table continues...)

(continued)

Field	Bits	Type	Description
ID	27:4	rwh	**PDU ID** • Type 1 PDU ID = Source ID + CAN ID (11 bits) • Type 2 PDU ID = Source ID + CAN ID (11 bits) + Metadata (8 bits)
DLC	31:28	rwh	**Data Length Code** 0_H CAN + CAN FD: received frame has 0 data bytes … 8_H CAN + CAN FD: received frame has 8 data bytes 9_H CAN FD: received frame has 12 (9*4-24) data bytes CAN: received frame has 8 data bytes … C_H CAN FD: received frame has 24 (12*4-24) data bytes CAN: received frame has 8 data bytes D_H CAN FD: received frame has 32 (13*16-176) data bytes CAN: received frame has 8 data bytes … F_H CAN FD: received frame has 64 (15*16-176) data bytes CAN: received frame has 8 data bytes
0	3	r	**Reserved**

21.7.121.11 RHBUF k Data Byte m

RHBUFk_RHBUF_DBm (k=0-1;m=0-63) Offset address: 00038_H+k*60_H+m

RHBUF k Data Byte m RAMInit value: XX_H

7	6	5	4	3	2	1	0
DB							
rwh							

Field	Bits	Type	Description
DB	7:0	rwh	**Data Byte m**

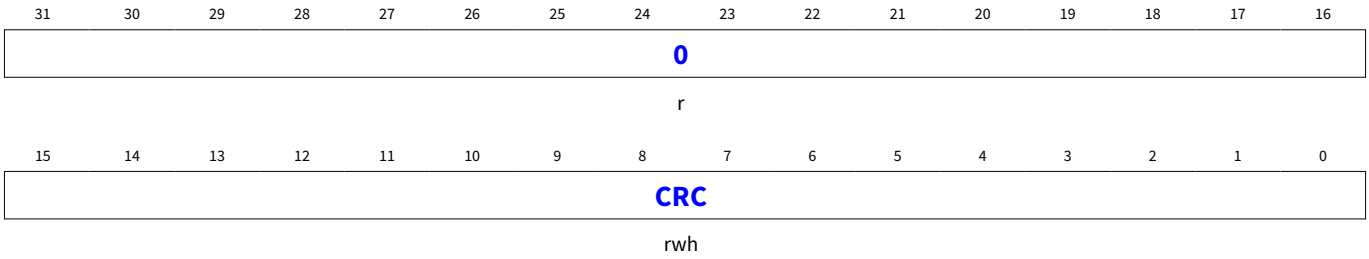
21.7.122 CRE Transmit Host Buffers

21.7.122.1 Tx Host Buffer CRC

CRC computed over the safety critical CAN payload of the CAN frame stored in the THBUF. This CRC is verified by the CRE before writing the CAN frame to the Tx FIFO/Queue

THBUFk_CRC (k=0-1) Offset address: $000E0_H+k*50_H$

Tx Host Buffer CRC RAMInit value: $XXXX\ XXXX_H$



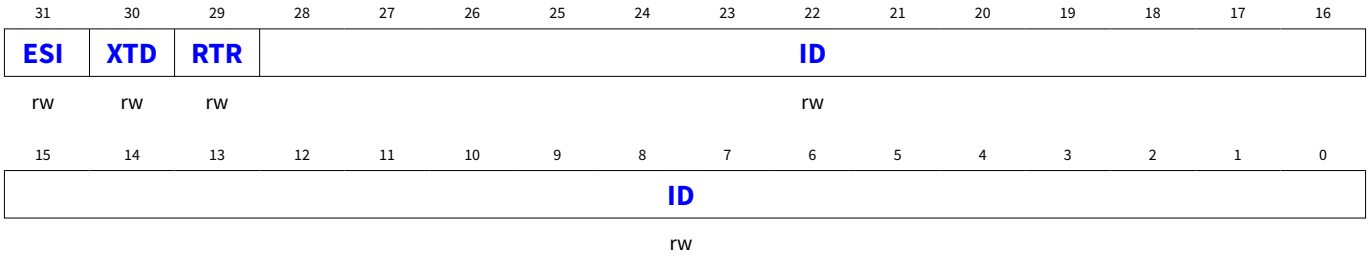
Field	Bits	Type	Description
CRC	15:0	rwh	**CRC** CRC to be verified by CRE
0	31:16	r	**Reserved**

21.7.122.2 Transmit Host Buffer Word 0

See Message layout.

THBUFk_T0 (k=0-1) Offset address: $000E4_H+k*50_H$

Transmit Host Buffer Word 0 RAMInit value: $XXXX\ XXXX_H$



Field	Bits	Type	Description
ID	28:0	rw	**Identifier** Standard or extended identifier depending on bit XTD. A standard identifier has to be written to ID[28:18].
RTR	29	rw	**Remote Transmission Request** ***Note**: When RTR = 1, the M_CAN transmits a remote frame according to ISO11898-1, even if Ni_CCCR.FDOE enables the transmission in CAN FD format.* 0_B Transmit data frame 1_B Transmit remote frame
XTD	30	rw	**Extended Identifier** 0_B 11-bit standard identifier 1_B 29-bit extended identifier

(table continues...)

(continued)

Field	Bits	Type	Description
ESI	31	rw	**Error State Indicator** ***Note**: The ESI bit of the transmit buffer is OR'ed with the error passive flag to decide the value of the ESI bit in the transmitted FD frame. As required by the CAN FD protocol specification, an error active node may optionally transmit the ESI bit recessive, but an error passive node will always transmit the ESI bit recessive.* 0_B ESI bit in CAN FD format depends only on error passive flag 1_B ESI bit in CAN FD format transmitted recessive

Related information

Functional description on page 3962

21.7.122.3 Transmit Host Buffer Word 1

See Message layout.

THBUFk_T1 (k=0-1) Offset address: $000E8_H+k*50_H$

Transmit Host Buffer Word 1 RAMInit value: $XXXX\ XXXX_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MM								EFC	TSCE	FDF	BRS	DLC			
rw								rw	rw	rw	rw	rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MM1								0							
rw								r							

Field	Bits	Type	Description
MM1	15:8	rw	**Message Marker** High byte of Wide Message Marker, written by CPU during Tx Buffer configuration. Copied into Tx Event FIFO element for identification of Tx message status. Available only when Ni_CCCR.WMM = '1' or when Ni_CCCR.UTSU = '1'.

(table continues...)

(continued)

Field	Bits	Type	Description
DLC	19:16	rw	**Data Length Code** 0_H CAN + CAN FD: received frame has 0 data bytes … 8_H CAN + CAN FD: received frame has 8 data bytes 9_H CAN FD: received frame has 12 (9*4-24) data bytes CAN: received frame has 8 data bytes … C_H CAN FD: received frame has 24 (12*4-24) data bytes CAN: received frame has 8 data bytes D_H CAN FD: received frame has 32 (13*16-176) data bytes CAN: received frame has 8 data bytes … F_H CAN FD: received frame has 64 (15*16-176) data bytes CAN: received frame has 8 data bytes
BRS	20	rw	**Bit Rate Switching** ***Note**:* *Bits ESI, FDF, and BRS are only evaluated when CAN FD operation is enabled Ni_CCCR.FDOE = 1'. Bit BRS is only evaluated when in addition Ni_CCCR.BRSE = '1'.* 0_B CAN FD frames transmitted without bit rate switching 1_B CAN FD frames transmitted with bit rate switching
FDF	21	rw	**FD Format** 0_B Frame transmitted in Classical CAN format 1_B Frame transmitted in CAN FD format
TSCE	22	rw	**Timestamp Capture Enable for TSU** Only available when Ni_CCCR.UTSU = '1'. When this bit is set and the message is transmitted, the transmission of a Sync message is signaled to the Timestamping Unit (TSU) connected to the M_CAN which captures the timestamp. 0= Timestamp Capture disabled 1= Timestamp Capture enabled
EFC	23	rw	**Event FIFO Control** 0_B Do not store Tx events 1_B Store Tx events
MM	31:24	rw	**Message Marker** Written by CPU during Tx Buffer configuration. Copied into Tx Event FIFO element for identification of Tx message status.
0	7:0	r	**Reserved** Read as 0; should be written with 0.

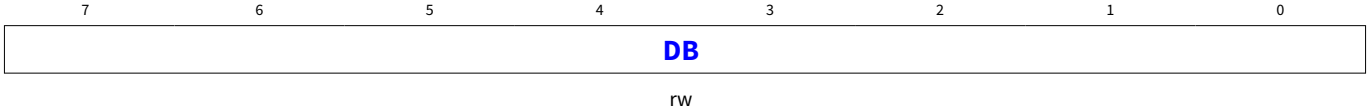
Related information

Functional description on page 3962

21.7.122.4 Transmit Host Buffer Data Byte m

THBUFk_DBm (k=0-1;m=0-63) Offset address: $000EC_H+k*50_H+m$

Transmit Host Buffer Data Byte m RAMInit value: XX_H



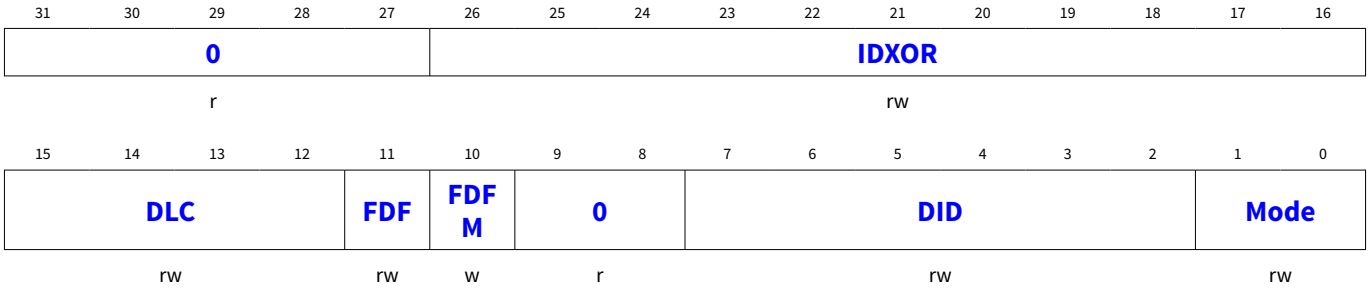
Field	Bits	Type	Description
DB	7:0	rw	**Data Byte m**

21.7.123 CRE Standard Routing Table

21.7.123.1 Standard routing table Uni-cast Rule

SRTk_UCR (k=0-127) Offset address: 00168_H+k*4

Standard routing table Uni-cast Rule RAMInit value: $XXXX\ XXXX_H$



Field	Bits	Type	Description
Mode	1:0	rw	**Defines the type of the routing** It defines the type of the routing rule 0 - Uni-cast Rule 1 - Multi-cast Rule 2 - PDU Routing Rule other - Reserved, considered as Uni-cast Rule
DID	7:2	rw	**Destination of Received CAN Frame** This bit-field indicates the Destination to which the received CAN frame has to be transferred to. The Destinations are referenced using a unique ID as given in the Routing ID overview table.
FDFM	10	w	**CAN Frame Format Modifier enable bit** CAN Frame Format Modifier enable bit 0 - the FDF bit is ignored and the received CAN frame format is not changed 1 - the received CAN frame format is changed either to Classical CAN frame (FDF=0) or to CAN FD frame format (FDF=1)

(table continues...)

(continued)

Field	Bits	Type	Description
FDF	11	rw	**Indicates if Received Frames is Classical or CAN-FD Frame** This bit-field indicates the Frame Format to which the received CAN frame has to be modified to. It is valid only when FDFM = 1. 0 - Classical CAN frame format 1 - CAN FD frame format
DLC	15:12	rw	**Indicates modification of Data Length Code of received frame** Indicates the modification to the Data Length Code of the received CAN frame before transferring the frame to its corresponding destination. DLC = 0, the Destination CAN frame length is same as the received CAN frame DLC != 0, Destination CAN frame length as defined by ISO 11898-1
IDXOR	26:16	rw	**CAN Frame ID modifier** Modification to the ID of the received CAN frame. The Destination CAN Frame ID = IDXOR (Xored) RX_CANID
0	9:8, 31:27	r	**Reserved** Read as 0; should be written with 0

21.7.123.2 Standard routing table Multi-cast Rule

SRTk_MCR (k=0-127) Offset address: 00168_H+k*4

Standard routing table Multi-cast Rule RAMInit value: $XXXX\ XXXX_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DID3						0		DID2						0	
rw						r		rw						r	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DID1						0		DID0						Mode	
rw						r		rw						rw	

Field	Bits	Type	Description
Mode	1:0	rw	**Defines the type of the routing** It defines the type of the routing rule 0 - Uni-cast Rule 1 - Multi-cast Rule 2 - PDU Routing Rule other - Reserved, considered as Uni-cast Rule
DID0	7:2	rw	**1st Destination of Received CAN Frame** This bit-field indicates the 1st Destination to which the received CAN frame has to be transferred to. The Destinations are referenced using a unique ID as given in the Routing ID overview table.

(table continues...)

(continued)

Field	Bits	Type	Description
DID1	15:10	rw	**2nd Destination of Received CAN Frame** This bit-field indicates the 2nd Destination to which the received CAN frame has to be transferred to. The Destinations are referenced using a unique ID as given in the Routing ID overview table.
DID2	23:18	rw	**3rd Destination of Recieved CAN Frame** This bit-field indicates the 3rd Destination to which the received CAN frame has to be transferred to. The Destinations are referenced using a unique ID as given in the Routing ID overview table.
DID3	31:26	rw	**4th Destination of Recieved CAN Frame** This bit-field indicates the 4th Destination to which the received CAN frame has to be transferred to. The Destinations are referenced using a unique ID as given in the Routing ID overview table.
0	9:8, 17:16, 25:24	r	**Reserved** Read as 0; should be written with 0

21.7.123.3 Standard routing table PDU Routing Rule

SRTk_PR (k=0-127) Offset address: 00168_H+k*4

Standard routing table PDU Routing Rule RAMInit value: XXXX $XXXX_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MD															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TYP			0					DID						Mode	
rw			r					rw						rw	

Field	Bits	Type	Description
Mode	1:0	rw	**Defines the type of the routing** It defines the type of the routing rule 0 - Uni-cast Rule 1 - Multi-cast Rule 2 - PDU Routing Rule other - Reserved, considered as Uni-cast Rule
DID	7:2	rw	**Destination ID of system memory** This bit-field indicates the system memory destination to which the received CAN frame has to be transferred to. The Destinations are referenced using a unique ID as given in the Routing ID overview table.

(table continues...)

(continued)

Field	Bits	Type	Description
TYP	15:13	rw	**PDU Header type** Selects the PDU header type. All other values are by default considered as Type 1 001_B **TYPE1**: 11- bit Standard CAN ID with no Metadata 010_B **TYPE2**: 11- bit Standard CAN ID with Metadata 011_B **TYPE3**: 11- bit Standard CAN ID with optional Metadata
MD	31:16	rw	**Metadata** SW configured Metadata used in the PDU ID generation
0	12:8	r	**Reserved** Read as 0; should be written with 0

21.7.124 CRE Extended Routing Table

21.7.124.1 Extended routing table Uni-cast Rule

XRTk_UCR (k=0-63) Offset address: 00368_H+k*4

Extended routing table Uni-cast Rule RAMInit value: XXXX $XXXX_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
IDSHIFT			0		IDXOR										
rw			r		rw										

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DLC				FDF	FDF M	0		DID						Mode	
rw				rw	w	r		rw						rw	

Field	Bits	Type	Description
Mode	1:0	rw	**Defines the type of routing table** It defines the type of the routing rule 0 - Uni-cast Rule 1 - Multi-cast Rule 2 - PDU Routing Rule other - Reserved, considered as Uni-cast Rule
DID	7:2	rw	**Destination of Recieved CAN Frame** This bit-field indicates the Destination to which the received CAN frame has to be transferred to. The Destinations are referenced using a unique ID as given in the Routing ID overview table.

(table continues...)

(continued)

Field	Bits	Type	Description
FDFM	10	w	**CAN Frame Format Modifier enable bit** CAN Frame Format Modifier enable bit 0 - the FDF bit is ignored and the received CAN frame format is not changed 1 - the received CAN frame format is changed either to Classical CAN frame (FDF=0) or to CAN FD frame format (FDF=1)
FDF	11	rw	**Indicates if recieved Frame is Classical or FD Format** This bit-field indicates the Frame Format to which the received CAN frame has to be modified to. It is valid only when FDFM = 1. 0 - Classical CAN frame format 1 - CAN FD frame format
DLC	15:12	rw	**Data Length Code** Indicates the modification to the Data Length Code of the received CAN frame before transferring the frame to its corresponding destination. DLC = 0, the Destination CAN frame length is same as the received CAN frame DLC != 0, Destination CAN frame length as defined by ISO 11898-1
IDXOR	26:16	rw	**CAN Frame ID modifier** Modification to the ID of the Received CAN frame. The Destination CAN Frame ID = [IDXOR<<IDSHIFT] (Xored) RX_CANID
IDSHIFT	31:29	rw	**CAN Frame ID Modifier** Modification to the ID of the Received CAN frame. The Destination CAN Frame ID = [IDXOR<<IDSHIFT] (Xored) RX_CANID
0	9:8, 28:27	r	**Reserved** Read as 0; should be written with 0

21.7.124.2 Extended routing table Multi-cast Rule

XRTk_MCR (k=0-63) Offset address: 00368_H+k*4

Extended routing table Multi-cast Rule RAMInit value: $XXXX\ XXXX_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DID3						0		DID2						0	
rw						r		rw						r	

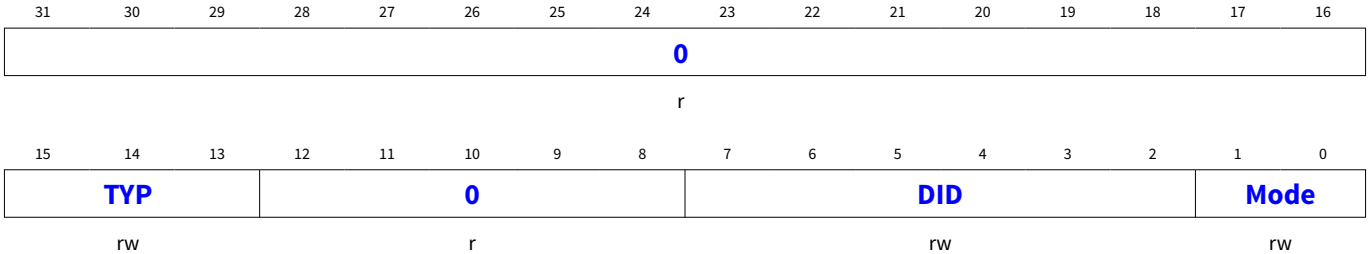
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DID1						0		DID0						Mode	
rw						r		rw						rw	

Field	Bits	Type	Description
Mode	1:0	rw	**Definition of type of routing table** It defines the type of the routing rule 0 - Uni-cast Rule 1 - Multi-cast Rule 2 - PDU Routing Rule other - Reserved, considered as Uni-cast Rule
DID0	7:2	rw	**1st Destination of Received CAN Frame** This bit-field indicates the 1st Destination to which the received CAN frame has to be transferred to. The Destinations are referenced using a unique ID as given in the Routing ID overview table.
DID1	15:10	rw	**2nd Destination of Received CAN Frame** This bit-field indicates the 2nd Destination to which the received CAN frame has to be transferred to. The Destinations are referenced using a unique ID as given in the Routing ID overview table.
DID2	23:18	rw	**3rd Destination of Received CAN Frame** This bit-field indicates the 3rd Destination to which the received CAN frame has to be transferred to. The Destinations are referenced using a unique ID as given in the Routing ID overview table.
DID3	31:26	rw	**4th Destination of Received CAN Frame** This bit-field indicates the 4th Destination to which the received CAN frame has to be transferred to. The Destinations are referenced using a unique ID as given in the Routing ID overview table.
0	9:8, 17:16, 25:24	r	**Reserved** Read as 0; should be written with 0

21.7.124.3 Extended ID routing table PDU Routing Rule

XRTk_PR (k=0-63) Offset address: 00368_H+k*4

Extended ID routing table PDU Routing Rule RAMInit value: $XXXX\ XXXX_H$



Field	Bits	Type	Description
Mode	1:0	rw	**Defines the type of the routing** It defines the type of the routing rule 0 - Uni-cast Rule 1 - Multi-cast Rule 2 - PDU Routing Rule other - Reserved, considered as Uni-cast Rule
DID	7:2	rw	**Destination ID of system memory** This bit-field indicates the system memory destination to which the received CAN frame has to be transferred to. The Destinations are referenced using a unique ID as given in the Routing ID overview table.
TYP	15:13	rw	**PDU Header type** Selects the PDU header type. All other values are considered by default as Type 4 100_B **TYPE4**: 29- bit Extended CAN ID with no Metadata 101_B **TYPE5**: 29- bit Extended CAN ID with no Metadata and no Source ID
0	12:8, 31:16	r	**Reserved** Read as 0; should be written with 0

21.7.125 CRE Standard Frame Rate Table

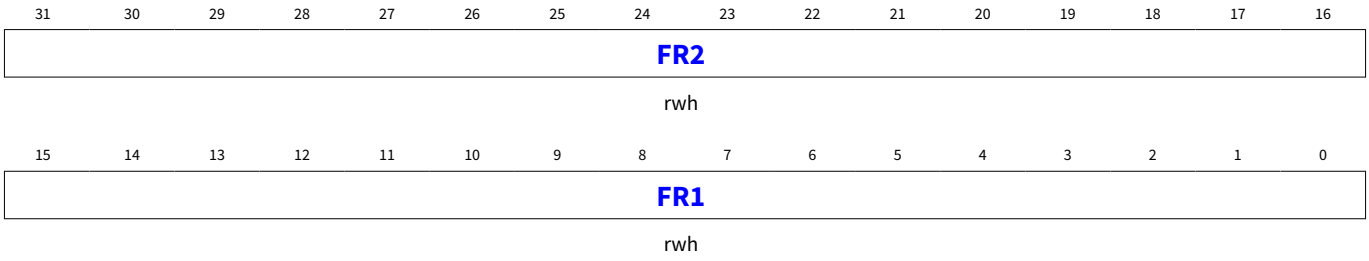
21.7.125.1 Standard ID Frame Rate Measure

Standard CAN ID Frame rate measure table

Consists of two Standard ID Frame rate measures

SFRk_FR (k=0-63) Offset address: 00468_H+k*4

Standard ID Frame Rate Measure RAMinit value: XXXX $XXXX_H$



Field	Bits	Type	Description
FR1	15:0	rwh	**Frame Rate Measure 1** Frame Rate Measure is relevant for Intrusion detection. It indicates the number of frames received
FR2	31:16	rwh	**Frame Rate Measure 2** Frame Rate Measure is relevant for Intrusion detection. It indicates the number of frames received

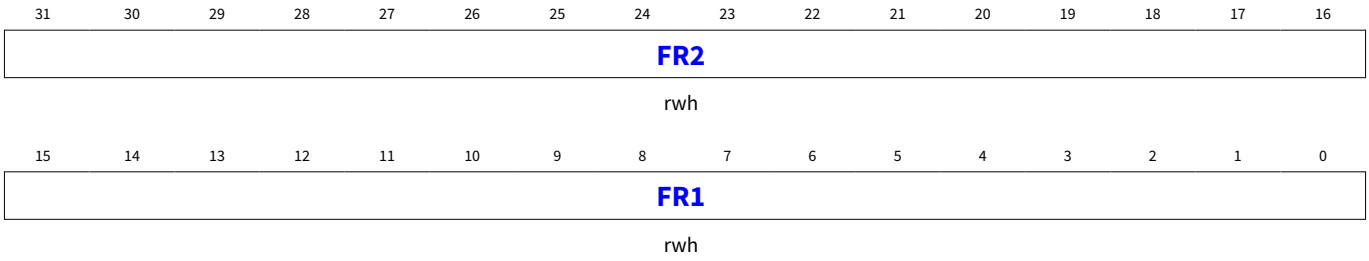
21.7.126 CRE Extended Frame Rate Table

21.7.126.1 Extended ID Frame Rate Measure

Extended CAN ID Frame Rate Mesaure table
Consists of two Extended ID Frame rate measures

XFRk_FR (k=0-31) Offset address: 00568_H+k*4

Extended ID Frame Rate Measure RAMInit value: XXXX $XXXX_H$



Field	Bits	Type	Description
FR1	15:0	rwh	**Frame Rate Measure 1** Frame Rate Measure is relevant for Intrusion detection. It indicates the number of frames received
FR2	31:16	rwh	**Frame Rate Measure 2** Frame Rate Measure is relevant for Intrusion detection. It indicates the number of frames received

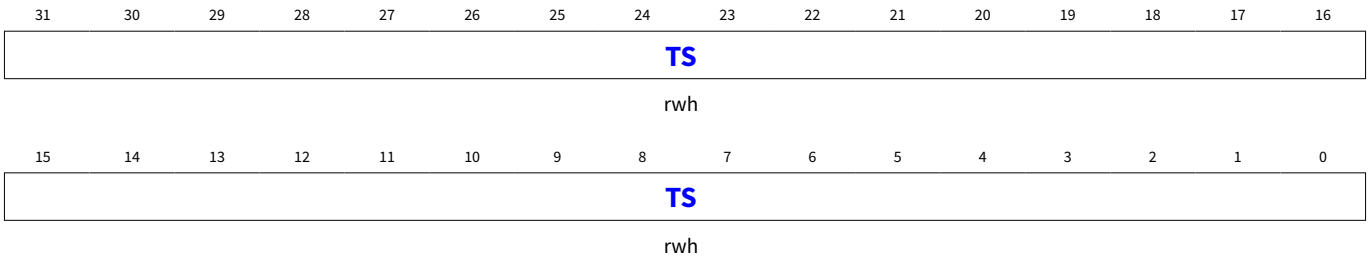
21.7.127 CRE Standard Timestamp Database

21.7.127.1 Standard ID Reference Timestamp

Standard ID Reference Timestamp which is used to calculate the RHBUFk_THEAD_INTRD. IAM
Note: The database needs to be initialized to 0

STSDk_RTS (k=0-127) Offset address: $005E8_H$+k*4

Standard ID Reference Timestamp RAMInit value: XXXX $XXXX_H$



Field	Bits	Type	Description
TS	31:0	rwh	**Reference Timestamp** This Reference Timestamp is the previous received Timestamp used for calculating the Inter Arrival Measure.

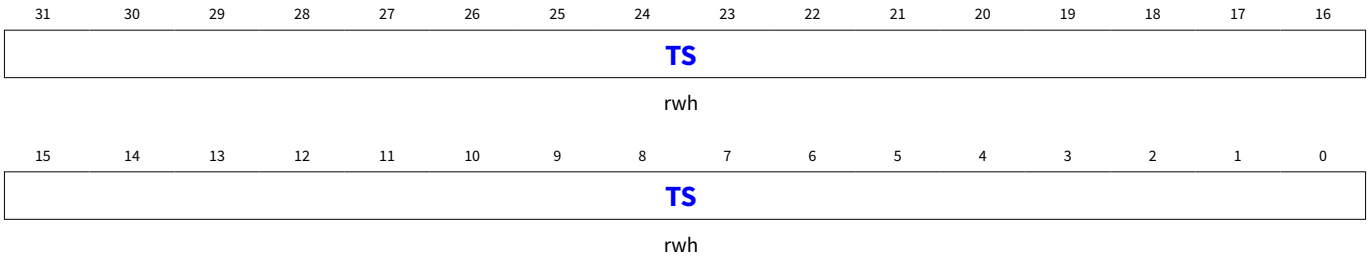
21.7.128 CRE Extended Timestamp Database

21.7.128.1 Extended ID Reference Timestamp

Extended ID Reference Timestamp which is used to calculate the RHBUFk_THEAD_INTRD. IAM
Note: The database needs to be initialized to 0

XTSDk_RTS (k=0-63) Offset address: $007E8_H+k*4$

Extended ID Reference Timestamp RAMInit value: $XXXX\ XXXX_H$



Field	Bits	Type	Description
TS	31:0	rwh	**Reference Timestamp** This Reference Timestamp is the previous received Timestamp used for calculating the Inter Arrival Measure.

21.8 Debug information

OCDS suspend behavior support

- Hard suspend
  - The clock is switched off to the MCMCAN module, and any ongoing transmission or reception event will be suspended immediately
- Soft suspend
  - The clock is switched off to the MCMCAN module after completion of the ongoing transmission and reception events. The Ni_CCCR.INIT bit will be set during the soft suspend. So, when leaving the soft suspend, the debugger has to ensure that the Ni_CCCR.INIT bit is cleared in order to continue normal CAN operation

Destructive debug

On destructive debug entry ie., when GP_CSRM_DBCFG.DESTDBG = 11b and GP_CSRM_DBGMODE.EDDM = 11b, the CAN transmit data output (TXD) is tied to “Recessive state” (logic ‘1b’). This blocks any further CAN transmission when the device has entered the Destructive Debug state.

OCDS trigger bus interface

The CAN trigger set is shown in table. Its output is on OTBG0 or OTGB1 controlled by the OCS register.

Table 1024 TS16_CAN trigger set

Values	Name	Description
i	AF	Acceptance filtering done for Node i
i + 4	MR	Message successfully received on Node i
i + 8	FDR	Fast Data Phase reception on Node i
i + 12	FDT	Fast Data Phase transmission on Node i

21.9 References

1. ISO 11898-1, Road vehicles — Controller area network (CAN) — Part 1: Data link layer and physical signalling, 2015
2. Society of Automotive Engineer, J1939 Recommended Practice for a Serial Control and Communications Vehicle Network
3. CAN in Automation, CiA 603: CAN Frame time-stamping – Requirements for network time management, version 1.0.0, 2017

21.10 CAN revision history

Reference	Description of change(s)
Date range: 2024-08-17 to 2024-11-27	
Node i Test Register, Node i Protocol Status Register and Node i CC Control Register	• Correcting wrong access condition type of bit-fields
Module Control Register, Node i Timer Transmit Trigger 0 Register and Measure Control Register	• Moving misplaced paragraphs from MCR register description to the appropriate register descriptions

21.11 TC4Dx CAN information

21.11.1 TC4Dx CAN configuration

Table 1025 Device specific parameters

Parameters	CAN0	CAN1	CAN2	CAN3	CAN4
Number of CAN nodes	4	4	4	4	4
RAM size (in Kbytes)	36	20	20	20	20
Debug over CAN support	Yes (only in Node 0)	No	No	No	No

21.11.2 TC4Dx CAN features

There are no deviations from the generic specification.

21.11.3 TC4Dx CAN functional description

- In TC4Dx device, the CAN modules are connected to ComPB bus interface. The CAN chapter refers to it as FPI bus
- In TC4Dx, the reset domain for the Node Start Address (Ni_STARTADR (i=0-3)) and Node End Address (Ni_ENDADR (i=0-3)) registers is Kernel Reset and the reset value remains the same as specified in the family specification 4010

21.11.4 TC4Dx CAN registers

21.11.4.1 Memory overview tables of CAN domain SFR

Table 1026 Memory overview - CAN domain SFR (ascending address)

Short name	Long name	Address
CAN0_RAM	Embedded SRAM for messages (09000_H Byte)	$F4700000_H$
CAN1_RAM	Embedded SRAM for messages (05000_H Byte)	$F4720000_H$
CAN2_RAM	Embedded SRAM for messages (05000_H Byte)	$F4740000_H$
CAN3_RAM	Embedded SRAM for messages (05000_H Byte)	$F4760000_H$
CAN4_RAM	Embedded SRAM for messages (05000_H Byte)	$F4780000_H$

21.11.4.2 Register address space - CAN

Table 1027 Registers address space - CAN

Key: The module name in brackets () indicates a memory section name. Without brackets the reference is to a functional block.

Module	Base address	End address	Note
CAN0	$F4710000_H$	$F471FFFF_H$	Slave interface for SFR registers
CAN1	$F4730000_H$	$F473FFFF_H$	Slave interface for SFR registers
CAN2	$F4750000_H$	$F475FFFF_H$	Slave interface for SFR registers
CAN3	$F4770000_H$	$F477FFFF_H$	Slave interface for SFR registers
CAN4	$F4790000_H$	$F479FFFF_H$	Slave interface for SFR registers
(CAN0_RAM)	$F4700000_H$	$F4708FFF_H$	Slave interface for Message RAM
(CAN1_RAM)	$F4720000_H$	$F4724FFF_H$	Slave interface for Message RAM
(CAN2_RAM)	$F4740000_H$	$F4744FFF_H$	Slave interface for Message RAM
(CAN3_RAM)	$F4760000_H$	$F4764FFF_H$	Slave interface for Message RAM
(CAN4_RAM)	$F4780000_H$	$F4784FFF_H$	Slave interface for Message RAM

21.11.4.3 Register overview - access mode glossary

Table 1028 Register overview - access mode glossary

Keyword	Description
E	Access protection using PROT register CAN0_PROTE or CAN1_PROTE or CAN2_PROTE or CAN3_PROTE or CAN4_PROTE .
SE	Access protection using PROT register CAN0_PROTSE or CAN1_PROTSE or CAN2_PROTSE or CAN3_PROTSE or CAN4_PROTSE .
APU-PNi (i=0-3)	Protection group consisting of registers CAN0_Ni_ACCEN_WRA , CAN0_Ni_ACCEN_WRB , CAN0_Ni_ACCEN_RDA , CAN0_Ni_ACCEN_RDB , CAN0_Ni_ACCEN_VM , CAN0_Ni_ACCEN_PRS or CAN1_Ni_ACCEN_WRA , CAN1_Ni_ACCEN_WRB , CAN1_Ni_ACCEN_RDA , CAN1_Ni_ACCEN_RDB , CAN1_Ni_ACCEN_VM , CAN1_Ni_ACCEN_PRS or CAN2_Ni_ACCEN_WRA , CAN2_Ni_ACCEN_WRB , CAN2_Ni_ACCEN_RDA , CAN2_Ni_ACCEN_RDB , CAN2_Ni_ACCEN_VM , CAN2_Ni_ACCEN_PRS or CAN3_Ni_ACCEN_WRA , CAN3_Ni_ACCEN_WRB , CAN3_Ni_ACCEN_RDA , CAN3_Ni_ACCEN_RDB , CAN3_Ni_ACCEN_VM , CAN3_Ni_ACCEN_PRS or CAN4_Ni_ACCEN_WRA , CAN4_Ni_ACCEN_WRB , CAN4_Ni_ACCEN_RDA , CAN4_Ni_ACCEN_RDB , CAN4_Ni_ACCEN_VM , CAN4_Ni_ACCEN_PRS .
PNi	Access protection using *APU-PNi* registers.

(table continues...)

Table 1028 (continued) Register overview - access mode glossary

Keyword	Description
APU-P4	Protection group consisting of registers CAN0_ACCEN_WRA , CAN0_ACCEN_WRB , CAN0_ACCEN_RDA , CAN0_ACCEN_RDB , CAN0_ACCEN_VM , CAN0_ACCEN_PRS or CAN1_ACCEN_WRA , CAN1_ACCEN_WRB , CAN1_ACCEN_RDA , CAN1_ACCEN_RDB , CAN1_ACCEN_VM , CAN1_ACCEN_PRS or CAN2_ACCEN_WRA , CAN2_ACCEN_WRB , CAN2_ACCEN_RDA , CAN2_ACCEN_RDB , CAN2_ACCEN_VM , CAN2_ACCEN_PRS or CAN3_ACCEN_WRA , CAN3_ACCEN_WRB , CAN3_ACCEN_RDA , CAN3_ACCEN_RDB , CAN3_ACCEN_VM , CAN3_ACCEN_PRS or CAN4_ACCEN_WRA , CAN4_ACCEN_WRB , CAN4_ACCEN_RDA , CAN4_ACCEN_RDB , CAN4_ACCEN_VM , CAN4_ACCEN_PRS .
P4	Access protection using *APU-P4* registers.
BE	Always returns a Bus Error.
U	No access restrictions.
PROT	Access restrictions as defined in the PROT register access rules.
nBE	Indicates that no Bus Error is generated when accessing this address range, even though it is either an access to an undefined address or the access does not follow the given rules.
M	Indicates a module specific access condition. Refer to the register description for details of the specific access condition.

21.11.4.4 Register overview - CAN0 domain SFR (ascending offset address)

Table 1029 Register overview - CAN0 domain SFR (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CAN0_CLC	Clock Control Register	00000_H	P4	P4, SV, E	3989	3989
CAN0_OCS	OCDS Control and Status Register	00004_H	P4	SV, P4	Debug Reset	3990
CAN0_ID	Module Identification Register	00008_H	P4	BE	PowerOn Reset	3991
CAN0_RST_CTRLA	Reset Control Register A	$0000C_H$	P4	P4, SV, E	Application Reset	3991
CAN0_RST_CTRLB	Reset Control Register B	00010_H	P4	P4, SV, E	Application Reset	3992
CAN0_RST_STAT	Reset Status Register	00014_H	P4	BE	Application Reset	3993
CAN0_PROTE	PROT Register Endinit	00018_H	U	SV, PROT	Application Reset	3993
CAN0_PROTSE	PROT Register Safe Endinit	$0001C_H$	U	SV, PROT	Application Reset	3995
CAN0_ACCEN_WRA	Write access enable register A	00030_H	U	SE, SV	Application Reset	3997

(table continues...)

Table 1029 (continued) Register overview - CAN0 domain SFR (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CAN0_ACCEN_WRB	Write access enable register B	00034_H	U	SE, SV	Application Reset	3997
CAN0_ACCEN_RDA	Read access enable register A	00038_H	U	SE, SV	Application Reset	3998
CAN0_ACCEN_RDB	Read access enable register B	$0003C_H$	U	SE, SV	Application Reset	3998
CAN0_ACCEN_VM	VM access enable register	00040_H	U	SE, SV	Application Reset	3999
CAN0_ACCEN_PRS	PRS access enable register	00044_H	U	SE, SV	Application Reset	3999
CAN0_MCR	Module Control Register	00070_H	P4	P4	4001	4001
CAN0_BUFADR	Buffer receive address and transmit address	00074_H	P4	P4	Kernel Reset	4003
CAN0_MECR	Measure Control Register	00080_H	P4	P4	Kernel Reset	4004
CAN0_MESTAT	Measure Status Register	00084_H	P4	P4	Kernel Reset	4005
CAN0_WDT	CRE Watchdog timer register	00088_H	P4	SV, E, P4	Kernel Reset	4006
CAN0_Ni_ACCEN_WRA (i=0-3)	Node i Write access enable register A	00100_H+i *400_H	U	SE, SV	Application Reset	4007
CAN0_Ni_ACCEN_WRB (i=0-3)	Node i Write access enable register B	00104_H+i *400_H	U	SE, SV	Application Reset	4008
CAN0_Ni_ACCEN_RDA (i=0-3)	Node i Read access enable register A	00108_H+i *400_H	U	SE, SV	Application Reset	4008
CAN0_Ni_ACCEN_RDB (i=0-3)	Node i Read access enable register B	$0010C_H$+ i*400_H	U	SE, SV	Application Reset	4009
CAN0_Ni_ACCEN_VM (i=0-3)	Node i VM access enable register	00110_H+i *400_H	U	SE, SV	Application Reset	4009
CAN0_Ni_ACCEN_PRS (i=0-3)	Node i PRS access enable register	00114_H+i *400_H	U	SE, SV	Application Reset	4010
CAN0_Ni_STARTADR (i=0-3)	Node i Start Address	00120_H+i *400_H	PNi	SV, E, PNi	Application Reset	4010
CAN0_Ni_ENDADR (i=0-3)	Node i End Address	00124_H+i *400_H	PNi	SV, E, PNi	Application Reset	4011

(table continues...)

Table 1029 (continued) Register overview - CAN0 domain SFR (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CAN0_Ni_INTRSIG (i=0-3)	Node i Interrupt Signalling Register	00128_H+i*400_H	PNi	nBE	Kernel Reset	4011
CAN0_Ni_G0INTR (i=0-3)	Node i Interrupt routing for Group 0	$0012C_H$+i*400_H	PNi	PNi	Kernel Reset	4013
CAN0_Ni_G1INTR (i=0-3)	Node i Interrupt routing for Group 1	00130_H+i*400_H	PNi	PNi	Kernel Reset	4014
CAN0_Ni_G2INTR (i=0-3)	Node i Interrupt routing for Group 2	00134_H+i*400_H	PNi	PNi	Kernel Reset	4015
CAN0_Ni_TIMER_CCR (i=0-3)	Node i Timer Clock Control Register	00138_H+i*400_H	PNi	PNi	Kernel Reset	4015
CAN0_Ni_TIMER_TXTRIG0 (i=0-3)	Node i Timer Transmit Trigger 0 Register	$0013C_H$+i*400_H	PNi	PNi	Kernel Reset	4016
CAN0_Ni_TIMER_TXTRIG1 (i=0-3)	Node i Timer Transmit Trigger 1 Register	00140_H+i*400_H	PNi	PNi	Kernel Reset	4017
CAN0_Ni_TIMER_TXTRIG2 (i=0-3)	Node i Timer Transmit Trigger 2 Register	00144_H+i*400_H	PNi	PNi	Kernel Reset	4018
CAN0_Ni_TIMER_RXTOUT (i=0-3)	Node i Timer Receive Timeout Register	00148_H+i*400_H	PNi	PNi	Kernel Reset	4018
CAN0_Ni_PORTCTRL (i=0-3)	Node i Port Control Register	$0014C_H$+i*400_H	PNi	PNi	Kernel Reset	4019
CAN0_Ni_CRE_CONFIG (i=0-3)	Node i CRE Configuration Register	00150_H+i*400_H	PNi	SV, E, PNi	Kernel Reset	4020
CAN0_Ni_CRE_CONFIGADR (i=0-3)	Node i CRE Configuration Start Address	00154_H+i*400_H	PNi	SV, E, PNi	Kernel Reset	4022
CAN0_Ni_CRE_HBUF_RXz_CONFIG (i=0-3;z=0-1)	Node i Receive Host Buffer z Configuration	00158_H+i*400_H+z*8	PNi	PNi	Kernel Reset	4022

(table continues...)

Table 1029 (continued) Register overview - CAN0 domain SFR (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CAN0_Ni_CRE_HBUF_RXz_STAT (i=0-3;z=0-1)	Node i Receive Host Buffer z Status	$0015C_H$+i*400_H+z*8	PNi	PNi	Kernel Reset	4024
CAN0_Ni_CRE_HBUF_TXz_CONFIG (i=0-3;z=0-1)	Node i Transmit Host Buffer z Configuration	00168_H+i*400_H+z*8	PNi	PNi	Kernel Reset	4025
CAN0_Ni_CRE_HBUF_TXz_STAT (i=0-3;z=0-1)	Node i Transmit Host Buffer z Status	$0016C_H$+i*400_H+z*8	PNi	PNi	Kernel Reset	4026
CAN0_Ni_CRE_IR (i=0-3)	Node i CRE Interrupt Register	$0018C_H$+i*400_H	PNi	PNi	Kernel Reset	4028
CAN0_Ni_IDMU_FRTCONFIG (i=0-3)	Node i Frame Rate Measure Table Configuration	00190_H+i*400_H	PNi	SV, E, PNi	Kernel Reset	4029
CAN0_Ni_IDMU_RXTPCFG (i=0-3)	Node i Rx Throughput Measure configuration	00194_H+i*400_H	PNi	SV, E, PNi	Kernel Reset	4030
CAN0_Ni_ERRCTRL (i=0-3)	Node i CRE Error control register	00198_H+i*400_H	P4	SV, E, P4	Kernel Reset	4031
CAN0_Ni_CREL (i=0-3)	Node i Core Release Register	00200_H+i*400_H	PNi	nBE	Kernel Reset	4032
CAN0_Ni_ENDN (i=0-3)	Node i Endian Register	00204_H+i*400_H	PNi	nBE	Kernel Reset	4033
CAN0_Ni_DBTP (i=0-3)	Node i Data Bit Timing & Prescaler Register	$0020C_H$+i*400_H	PNi	PNi	4034	4034
CAN0_Ni_TEST (i=0-3)	Node i Test Register	00210_H+i*400_H	PNi	PNi	4035	4035
CAN0_Ni_RWD (i=0-3)	Node i RAM Watchdog	00214_H+i*400_H	PNi	PNi	Kernel Reset	4036
CAN0_Ni_CCCR (i=0-3)	Node i CC Control Register	00218_H+i*400_H	PNi	PNi	4037	4037
CAN0_Ni_NBTP (i=0-3)	Node i Nominal Bit Timing & Prescaler Register	$0021C_H$+i*400_H	PNi	PNi	4040	4040
CAN0_Ni_TSCC (i=0-3)	Node i Timestamp Counter Configuration	00220_H+i*400_H	PNi	PNi	Kernel Reset	4041
CAN0_Ni_TSCV (i=0-3)	Node i Timestamp Counter Value	00224_H+i*400_H	PNi	PNi	Kernel Reset	4042

(table continues...)

Table 1029 (continued) Register overview - CAN0 domain SFR (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CAN0_Ni_TOCC (i=0-3)	Node i Timeout Counter Configuration	00228_H+i *400_H	PNi	PNi	Kernel Reset	4043
CAN0_Ni_TOCV (i=0-3)	Node i Timeout Counter Value	$0022C_H$+ i*400_H	PNi	PNi	Kernel Reset	4044
CAN0_Ni_ECR (i=0-3)	Node i Error Counter Register	00240_H+i *400_H	PNi	nBE	Kernel Reset	4045
CAN0_Ni_PSR (i=0-3)	Node i Protocol Status Register	00244_H+i *400_H	PNi	nBE	Kernel Reset	4045
CAN0_Ni_TDCR (i=0-3)	Node i Transmitter Delay Compensation Register	00248_H+i *400_H	PNi	PNi	Kernel Reset	4049
CAN0_Ni_IR (i=0-3)	Node i Interrupt Register	00250_H+i *400_H	PNi	PNi	Kernel Reset	4050
CAN0_Ni_IE (i=0-3)	Node i Interrupt Enable	00254_H+i *400_H	PNi	PNi	Kernel Reset	4053
CAN0_Ni_GFC (i=0-3)	Node i Global Filter Configuration	00280_H+i *400_H	PNi	PNi	Kernel Reset	4056
CAN0_Ni_SIDFC (i=0-3)	Node i Standard ID Filter Configuration	00284_H+i *400_H	PNi	PNi	4057	4057
CAN0_Ni_XIDFC (i=0-3)	Node i Extended ID Filter Configuration	00288_H+i *400_H	PNi	PNi	Kernel Reset	4058
CAN0_Ni_XIDAM (i=0-3)	Node i Extended ID AND Mask	00290_H+i *400_H	PNi	PNi	Kernel Reset	4058
CAN0_Ni_HPMS (i=0-3)	Node i High Priority Message Status	00294_H+i *400_H	PNi	nBE	Kernel Reset	4059
CAN0_Ni_NDAT1 (i=0-3)	Node i New Data 1	00298_H+i *400_H	PNi	PNi	Kernel Reset	4060
CAN0_Ni_NDAT2 (i=0-3)	Node i New Data 2	$0029C_H$+ i*400_H	PNi	PNi	Kernel Reset	4060
CAN0_Ni_RXF0C (i=0-3)	Node i Rx FIFO 0 Configuration	$002A0_H$+ i*400_H	PNi	PNi	Kernel Reset	4061
CAN0_Ni_RXF0S (i=0-3)	Node i Rx FIFO 0 Status	$002A4_H$+ i*400_H	PNi	nBE	Kernel Reset	4062
CAN0_Ni_RXF0A (i=0-3)	Node i Rx FIFO 0 Acknowledge	$002A8_H$+ i*400_H	PNi	PNi	Kernel Reset	4063
CAN0_Ni_RXBC (i=0-3)	Node i Rx Buffer Configuration	$002AC_H$+ i*400_H	PNi	PNi	4063	4063
CAN0_Ni_RXF1C (i=0-3)	Node i Rx FIFO 1 Configuration	$002B0_H$+ i*400_H	PNi	PNi	Kernel Reset	4064

(table continues...)

Table 1029 (continued) Register overview - CAN0 domain SFR (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CAN0_Ni_RXF1S (i=0-3)	Node i Rx FIFO 1 Status	$002B4_H$+ i*400_H	PNi	nBE	Kernel Reset	4065
CAN0_Ni_RXF1A (i=0-3)	Node i Rx FIFO 1 Acknowledge	$002B8_H$+ i*400_H	PNi	PNi	Kernel Reset	4066
CAN0_Ni_RXESC (i=0-3)	Node i Rx Buffer/FIFO Element Size Configuration	$002BC_H$+ i*400_H	PNi	PNi	Kernel Reset	4067
CAN0_Ni_TXBC (i=0-3)	Node i Tx Buffer Configuration	$002C0_H$+ i*400_H	PNi	PNi	4068	4068
CAN0_Ni_TXFQS (i=0-3)	Node i Tx FIFO/Queue Status	$002C4_H$+ i*400_H	PNi	nBE	Kernel Reset	4069
CAN0_Ni_TXESC (i=0-3)	Node i Tx Buffer Element Size Configuration	$002C8_H$+ i*400_H	PNi	PNi	4070	4070
CAN0_Ni_TXBRP (i=0-3)	Node i Tx Buffer Request Pending	$002CC_H$+ i*400_H	PNi	PNi	Kernel Reset	4072
CAN0_Ni_TXBAR (i=0-3)	Node i Tx Buffer Add Request	$002D0_H$+ i*400_H	PNi	PNi	Kernel Reset	4073
CAN0_Ni_TXBCR (i=0-3)	Node i Tx Buffer Cancellation Request	$002D4_H$+ i*400_H	PNi	PNi	Kernel Reset	4073
CAN0_Ni_TXBTO (i=0-3)	Node i Tx Buffer Transmission Occurred	$002D8_H$+ i*400_H	PNi	nBE	Kernel Reset	4074
CAN0_Ni_TXBCF (i=0-3)	Node i Tx Buffer Cancellation Finished	$002DC_H$+ i*400_H	PNi	nBE	Kernel Reset	4074
CAN0_Ni_TXBTIE (i=0-3)	Node i Tx Buffer Transmission Interrupt Enable	$002E0_H$+i *400_H	PNi	PNi	Kernel Reset	4075
CAN0_Ni_TXBCIE (i=0-3)	Node i Tx Buffer Cancellation Finished Interrupt Enable	$002E4_H$+i *400_H	PNi	PNi	Kernel Reset	4075
CAN0_Ni_TXEFC (i=0-3)	Node i Tx Event FIFO Configuration	$002F0_H$+i *400_H	PNi	PNi	Kernel Reset	4076
CAN0_Ni_TXEFS (i=0-3)	Node i Tx Event FIFO Status	$002F4_H$+i *400_H	PNi	nBE	Kernel Reset	4077
CAN0_Ni_TXEFA (i=0-3)	Node i Tx Event FIFO Acknowledge	$002F8_H$+i *400_H	PNi	PNi	Kernel Reset	4077
CAN0_Ni_TSU_CREL (i=0-3)	Node i TSU Core Release Register	00360_H+i *400_H	PNi	nBE	Kernel Reset	4078

(table continues...)

Table 1029 (continued) Register overview - CAN0 domain SFR (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CAN0_Ni_TSU_TSCFG (i=0) (i=1-3)	Node i Timestamp Configuration	00364_H+i*400_H	PNi	PNi	Kernel Reset	4078
CAN0_Ni_TSU_TSS1 (i=0-3)	Node i Timestamp Status 1	00368_H+i*400_H	PNi	nBE	Kernel Reset	4080
CAN0_Ni_TSU_TSS2 (i=0-3)	Node i Timestamp Status 2	$0036C_H$+i*400_H	PNi	nBE	Kernel Reset	4081
CAN0_Ni_TSU_TSm (i=0-3;m=0-15)	Node i Timestamp m	00370_H+i*400_H+m*4	PNi	nBE	Kernel Reset	4082
CAN0_Ni_TSU_ATB (i=0-3)	Node i Actual Timebase	$003B0_H$+i*400_H	PNi	U	Kernel Reset	4082

21.11.4.5 Register overview - CAN1 domain SFR (ascending offset address)

Table 1030 Register overview - CAN1 domain SFR (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CAN1_CLC	Clock Control Register	00000_H	P4	P4, SV, E	Application Reset	4154
CAN1_OCS	OCDS Control and Status Register	00004_H	P4	SV, P4	Debug Reset	3990
CAN1_ID	Module Identification Register	00008_H	P4	BE	PowerOn Reset	3991
CAN1_RST_CTRLA	Reset Control Register A	$0000C_H$	P4	P4, SV, E	Application Reset	3991
CAN1_RST_CTRLB	Reset Control Register B	00010_H	P4	P4, SV, E	Application Reset	3992
CAN1_RST_STAT	Reset Status Register	00014_H	P4	BE	Application Reset	3993
CAN1_PROTE	PROT Register Endinit	00018_H	U	SV, PROT	Application Reset	3993
CAN1_PROTSE	PROT Register Safe Endinit	$0001C_H$	U	SV, PROT	Application Reset	3995
CAN1_ACCEN_WRA	Write access enable register A	00030_H	U	SE, SV	Application Reset	3997
CAN1_ACCEN_WRB	Write access enable register B	00034_H	U	SE, SV	Application Reset	3997

(table continues...)

Table 1030 (continued) Register overview - CAN1 domain SFR (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CAN1_ACCEN_RDA	Read access enable register A	00038_H	U	SE, SV	Application Reset	3998
CAN1_ACCEN_RDB	Read access enable register B	$0003C_H$	U	SE, SV	Application Reset	3998
CAN1_ACCEN_VM	VM access enable register	00040_H	U	SE, SV	Application Reset	3999
CAN1_ACCEN_PRS	PRS access enable register	00044_H	U	SE, SV	Application Reset	3999
CAN1_MCR	Module Control Register	00070_H	P4	P4	Kernel Reset	4156
CAN1_WDT	CRE Watchdog timer register	00088_H	P4	SV, E, P4	Kernel Reset	4006
CAN1_Ni_ACCEN_WRA (i=0-3)	Node i Write access enable register A	00100_H+i *400_H	U	SE, SV	Application Reset	4007
CAN1_Ni_ACCEN_WRB (i=0-3)	Node i Write access enable register B	00104_H+i *400_H	U	SE, SV	Application Reset	4008
CAN1_Ni_ACCEN_RDA (i=0-3)	Node i Read access enable register A	00108_H+i *400_H	U	SE, SV	Application Reset	4008
CAN1_Ni_ACCEN_RDB (i=0-3)	Node i Read access enable register B	$0010C_H$+ i*400_H	U	SE, SV	Application Reset	4009
CAN1_Ni_ACCEN_VM (i=0-3)	Node i VM access enable register	00110_H+i *400_H	U	SE, SV	Application Reset	4009
CAN1_Ni_ACCEN_PRS (i=0-3)	Node i PRS access enable register	00114_H+i *400_H	U	SE, SV	Application Reset	4010
CAN1_Ni_STARTADR (i=0-3)	Node i Start Address	00120_H+i *400_H	PNi	SV, E, PNi	Application Reset	4010
CAN1_Ni_ENDADR (i=0-3)	Node i End Address	00124_H+i *400_H	PNi	SV, E, PNi	Application Reset	4011
CAN1_Ni_INTRSIG (i=0-3)	Node i Interrupt Signalling Register	00128_H+i *400_H	PNi	nBE	Kernel Reset	4011
CAN1_Ni_G0INTR (i=0-3)	Node i Interrupt routing for Group 0	$0012C_H$+ i*400_H	PNi	PNi	Kernel Reset	4013

(table continues...)

Table 1030 (continued) Register overview - CAN1 domain SFR (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CAN1_Ni_G1INTR (i=0-3)	Node i Interrupt routing for Group 1	00130_H+i*400_H	PNi	PNi	Kernel Reset	4014
CAN1_Ni_G2INTR (i=0-3)	Node i Interrupt routing for Group 2	00134_H+i*400_H	PNi	PNi	Kernel Reset	4015
CAN1_Ni_TIMER_CCR (i=0-3)	Node i Timer Clock Control Register	00138_H+i*400_H	PNi	PNi	Kernel Reset	4015
CAN1_Ni_TIMER_TXTRIG0 (i=0-3)	Node i Timer Transmit Trigger 0 Register	$0013C_H$+i*400_H	PNi	PNi	Kernel Reset	4016
CAN1_Ni_TIMER_TXTRIG1 (i=0-3)	Node i Timer Transmit Trigger 1 Register	00140_H+i*400_H	PNi	PNi	Kernel Reset	4017
CAN1_Ni_TIMER_TXTRIG2 (i=0-3)	Node i Timer Transmit Trigger 2 Register	00144_H+i*400_H	PNi	PNi	Kernel Reset	4018
CAN1_Ni_TIMER_RXTOUT (i=0-3)	Node i Timer Receive Timeout Register	00148_H+i*400_H	PNi	PNi	Kernel Reset	4018
CAN1_Ni_PORTCTRL (i=0-3)	Node i Port Control Register	$0014C_H$+i*400_H	PNi	PNi	Kernel Reset	4019
CAN1_Ni_CRE_CONFIG (i=0-3)	Node i CRE Configuration Register	00150_H+i*400_H	PNi	SV, E, PNi	Kernel Reset	4158
CAN1_Ni_CRE_CONFIGADR (i=0-3)	Node i CRE Configuration Start Address	00154_H+i*400_H	PNi	SV, E, PNi	Kernel Reset	4022
CAN1_Ni_CRE_HBUF_RXz_CONFIG (i=0-3;z=0-1)	Node i Receive Host Buffer z Configuration	00158_H+i*400_H+z*8	PNi	PNi	Kernel Reset	4022
CAN1_Ni_CRE_HBUF_RXz_STAT (i=0-3;z=0-1)	Node i Receive Host Buffer z Status	$0015C_H$+i*400_H+z*8	PNi	PNi	Kernel Reset	4024
CAN1_Ni_CRE_HBUF_TXz_CONFIG (i=0-3;z=0-1)	Node i Transmit Host Buffer z Configuration	00168_H+i*400_H+z*8	PNi	PNi	Kernel Reset	4025

(table continues...)

Table 1030 (continued) Register overview - CAN1 domain SFR (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CAN1_Ni_CRE_HBUF_TXz_STAT (i=0-3;z=0-1)	Node i Transmit Host Buffer z Status	$0016C_H$+i*400_H+z*8	PNi	PNi	Kernel Reset	4026
CAN1_Ni_CRE_IR (i=0-3)	Node i CRE Interrupt Register	$0018C_H$+i*400_H	PNi	PNi	Kernel Reset	4028
CAN1_Ni_IDMU_FRTCONFIG (i=0-3)	Node i Frame Rate Measure Table Configuration	00190_H+i*400_H	PNi	SV, E, PNi	Kernel Reset	4029
CAN1_Ni_IDMU_RXTPCFG (i=0-3)	Node i Rx Throughput Measure configuration	00194_H+i*400_H	PNi	SV, E, PNi	Kernel Reset	4030
CAN1_Ni_ERRCTRL (i=0-3)	Node i CRE Error control register	00198_H+i*400_H	P4	SV, E, P4	Kernel Reset	4031
CAN1_Ni_CREL (i=0-3)	Node i Core Release Register	00200_H+i*400_H	PNi	nBE	Kernel Reset	4032
CAN1_Ni_ENDN (i=0-3)	Node i Endian Register	00204_H+i*400_H	PNi	nBE	Kernel Reset	4033
CAN1_Ni_DBTP (i=0-3)	Node i Data Bit Timing & Prescaler Register	$0020C_H$+i*400_H	PNi	PNi	4034	4034
CAN1_Ni_TEST (i=0-3)	Node i Test Register	00210_H+i*400_H	PNi	PNi	4035	4035
CAN1_Ni_RWD (i=0-3)	Node i RAM Watchdog	00214_H+i*400_H	PNi	PNi	Kernel Reset	4036
CAN1_Ni_CCCR (i=0-3)	Node i CC Control Register	00218_H+i*400_H	PNi	PNi	4037	4037
CAN1_Ni_NBTP (i=0-3)	Node i Nominal Bit Timing & Prescaler Register	$0021C_H$+i*400_H	PNi	PNi	4040	4040
CAN1_Ni_TSCC (i=0-3)	Node i Timestamp Counter Configuration	00220_H+i*400_H	PNi	PNi	Kernel Reset	4041
CAN1_Ni_TSCV (i=0-3)	Node i Timestamp Counter Value	00224_H+i*400_H	PNi	PNi	Kernel Reset	4042
CAN1_Ni_TOCC (i=0-3)	Node i Timeout Counter Configuration	00228_H+i*400_H	PNi	PNi	Kernel Reset	4043
CAN1_Ni_TOCV (i=0-3)	Node i Timeout Counter Value	$0022C_H$+i*400_H	PNi	PNi	Kernel Reset	4044
CAN1_Ni_ECR (i=0-3)	Node i Error Counter Register	00240_H+i*400_H	PNi	nBE	Kernel Reset	4045

(table continues...)

Table 1030 (continued) Register overview - CAN1 domain SFR (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CAN1_Ni_PSR (i=0-3)	Node i Protocol Status Register	00244_H+i *400_H	PNi	nBE	Kernel Reset	4045
CAN1_Ni_TDCR (i=0-3)	Node i Transmitter Delay Compensation Register	00248_H+i *400_H	PNi	PNi	Kernel Reset	4049
CAN1_Ni_IR (i=0-3)	Node i Interrupt Register	00250_H+i *400_H	PNi	PNi	Kernel Reset	4050
CAN1_Ni_IE (i=0-3)	Node i Interrupt Enable	00254_H+i *400_H	PNi	PNi	Kernel Reset	4053
CAN1_Ni_GFC (i=0-3)	Node i Global Filter Configuration	00280_H+i *400_H	PNi	PNi	Kernel Reset	4056
CAN1_Ni_SIDFC (i=0-3)	Node i Standard ID Filter Configuration	00284_H+i *400_H	PNi	PNi	4057	4057
CAN1_Ni_XIDFC (i=0-3)	Node i Extended ID Filter Configuration	00288_H+i *400_H	PNi	PNi	Kernel Reset	4058
CAN1_Ni_XIDAM (i=0-3)	Node i Extended ID AND Mask	00290_H+i *400_H	PNi	PNi	Kernel Reset	4058
CAN1_Ni_HPMS (i=0-3)	Node i High Priority Message Status	00294_H+i *400_H	PNi	nBE	Kernel Reset	4059
CAN1_Ni_NDAT1 (i=0-3)	Node i New Data 1	00298_H+i *400_H	PNi	PNi	Kernel Reset	4060
CAN1_Ni_NDAT2 (i=0-3)	Node i New Data 2	$0029C_H$+ i*400_H	PNi	PNi	Kernel Reset	4060
CAN1_Ni_RXF0C (i=0-3)	Node i Rx FIFO 0 Configuration	$002A0_H$+ i*400_H	PNi	PNi	Kernel Reset	4061
CAN1_Ni_RXF0S (i=0-3)	Node i Rx FIFO 0 Status	$002A4_H$+ i*400_H	PNi	nBE	Kernel Reset	4062
CAN1_Ni_RXF0A (i=0-3)	Node i Rx FIFO 0 Acknowledge	$002A8_H$+ i*400_H	PNi	PNi	Kernel Reset	4063
CAN1_Ni_RXBC (i=0-3)	Node i Rx Buffer Configuration	$002AC_H$+ i*400_H	PNi	PNi	4063	4063
CAN1_Ni_RXF1C (i=0-3)	Node i Rx FIFO 1 Configuration	$002B0_H$+ i*400_H	PNi	PNi	Kernel Reset	4064
CAN1_Ni_RXF1S (i=0-3)	Node i Rx FIFO 1 Status	$002B4_H$+ i*400_H	PNi	nBE	Kernel Reset	4065
CAN1_Ni_RXF1A (i=0-3)	Node i Rx FIFO 1 Acknowledge	$002B8_H$+ i*400_H	PNi	PNi	Kernel Reset	4066
CAN1_Ni_RXESC (i=0-3)	Node i Rx Buffer/FIFO Element Size Configuration	$002BC_H$+ i*400_H	PNi	PNi	Kernel Reset	4067

(table continues...)

Table 1030 (continued) Register overview - CAN1 domain SFR (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CAN1_Ni_TXBC (i=0-3)	Node i Tx Buffer Configuration	$002C0_H$+ i*400_H	PNi	PNi	4068	4068
CAN1_Ni_TXFQS (i=0-3)	Node i Tx FIFO/Queue Status	$002C4_H$+ i*400_H	PNi	nBE	Kernel Reset	4069
CAN1_Ni_TXESC (i=0-3)	Node i Tx Buffer Element Size Configuration	$002C8_H$+ i*400_H	PNi	PNi	4070	4070
CAN1_Ni_TXBRP (i=0-3)	Node i Tx Buffer Request Pending	$002CC_H$+ i*400_H	PNi	PNi	Kernel Reset	4072
CAN1_Ni_TXBAR (i=0-3)	Node i Tx Buffer Add Request	$002D0_H$+ i*400_H	PNi	PNi	Kernel Reset	4073
CAN1_Ni_TXBCR (i=0-3)	Node i Tx Buffer Cancellation Request	$002D4_H$+ i*400_H	PNi	PNi	Kernel Reset	4073
CAN1_Ni_TXBTO (i=0-3)	Node i Tx Buffer Transmission Occurred	$002D8_H$+ i*400_H	PNi	nBE	Kernel Reset	4074
CAN1_Ni_TXBCF (i=0-3)	Node i Tx Buffer Cancellation Finished	$002DC_H$+ i*400_H	PNi	nBE	Kernel Reset	4074
CAN1_Ni_TXBTIE (i=0-3)	Node i Tx Buffer Transmission Interrupt Enable	$002E0_H$+i *400_H	PNi	PNi	Kernel Reset	4075
CAN1_Ni_TXBCIE (i=0-3)	Node i Tx Buffer Cancellation Finished Interrupt Enable	$002E4_H$+i *400_H	PNi	PNi	Kernel Reset	4075
CAN1_Ni_TXEFC (i=0-3)	Node i Tx Event FIFO Configuration	$002F0_H$+i *400_H	PNi	PNi	Kernel Reset	4076
CAN1_Ni_TXEFS (i=0-3)	Node i Tx Event FIFO Status	$002F4_H$+i *400_H	PNi	nBE	Kernel Reset	4077
CAN1_Ni_TXEFA (i=0-3)	Node i Tx Event FIFO Acknowledge	$002F8_H$+i *400_H	PNi	PNi	Kernel Reset	4077
CAN1_Ni_TSU_CREL (i=0-3)	Node i TSU Core Release Register	00360_H+i *400_H	PNi	nBE	Kernel Reset	4078
CAN1_Ni_TSU_TSCFG (i=0) (i=1-3)	Node i Timestamp Configuration	00364_H+i *400_H	PNi	PNi	Kernel Reset	4078
CAN1_Ni_TSU_TSS1 (i=0-3)	Node i Timestamp Status 1	00368_H+i *400_H	PNi	nBE	Kernel Reset	4080

(table continues...)

Table 1030 (continued) Register overview - CAN1 domain SFR (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CAN1_Ni_TSU_TSS2 (i=0-3)	Node i Timestamp Status 2	$0036C_H$+i*400_H	PNi	nBE	Kernel Reset	4081
CAN1_Ni_TSU_TSm (i=0-3;m=0-15)	Node i Timestamp m	00370_H+i*400_H+m*4	PNi	nBE	Kernel Reset	4082
CAN1_Ni_TSU_ATB (i=0-3)	Node i Actual Timebase	$003B0_H$+i*400_H	PNi	U	Kernel Reset	4082

21.11.4.6 Register overview - CAN2 domain SFR (ascending offset address)

Table 1031 Register overview - CAN2 domain SFR (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CAN2_CLC	Clock Control Register	00000_H	P4	P4, SV, E	Application Reset	4154
CAN2_OCS	OCDS Control and Status Register	00004_H	P4	SV, P4	Debug Reset	3990
CAN2_ID	Module Identification Register	00008_H	P4	BE	PowerOn Reset	3991
CAN2_RST_CTRLA	Reset Control Register A	$0000C_H$	P4	P4, SV, E	Application Reset	3991
CAN2_RST_CTRLB	Reset Control Register B	00010_H	P4	P4, SV, E	Application Reset	3992
CAN2_RST_STAT	Reset Status Register	00014_H	P4	BE	Application Reset	3993
CAN2_PROTE	PROT Register Endinit	00018_H	U	SV, PROT	Application Reset	3993
CAN2_PROTSE	PROT Register Safe Endinit	$0001C_H$	U	SV, PROT	Application Reset	3995
CAN2_ACCEN_WRA	Write access enable register A	00030_H	U	SE, SV	Application Reset	3997
CAN2_ACCEN_WRB	Write access enable register B	00034_H	U	SE, SV	Application Reset	3997
CAN2_ACCEN_RDA	Read access enable register A	00038_H	U	SE, SV	Application Reset	3998
CAN2_ACCEN_RDB	Read access enable register B	$0003C_H$	U	SE, SV	Application Reset	3998
CAN2_ACCEN_VM	VM access enable register	00040_H	U	SE, SV	Application Reset	3999

(table continues...)

Table 1031 (continued) Register overview - CAN2 domain SFR (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CAN2_ACCEN_PRS	PRS access enable register	00044_H	U	SE, SV	Application Reset	3999
CAN2_MCR	Module Control Register	00070_H	P4	P4	Kernel Reset	4156
CAN2_WDT	CRE Watchdog timer register	00088_H	P4	SV, E, P4	Kernel Reset	4006
CAN2_Ni_ACCEN_WRA (i=0-3)	Node i Write access enable register A	00100_H+i *400_H	U	SE, SV	Application Reset	4007
CAN2_Ni_ACCEN_WRB (i=0-3)	Node i Write access enable register B	00104_H+i *400_H	U	SE, SV	Application Reset	4008
CAN2_Ni_ACCEN_RDA (i=0-3)	Node i Read access enable register A	00108_H+i *400_H	U	SE, SV	Application Reset	4008
CAN2_Ni_ACCEN_RDB (i=0-3)	Node i Read access enable register B	$0010C_H$+ i*400_H	U	SE, SV	Application Reset	4009
CAN2_Ni_ACCEN_VM (i=0-3)	Node i VM access enable register	00110_H+i *400_H	U	SE, SV	Application Reset	4009
CAN2_Ni_ACCEN_PRS (i=0-3)	Node i PRS access enable register	00114_H+i *400_H	U	SE, SV	Application Reset	4010
CAN2_Ni_STARTADR (i=0-3)	Node i Start Address	00120_H+i *400_H	PNi	SV, E, PNi	Application Reset	4010
CAN2_Ni_ENDADR (i=0-3)	Node i End Address	00124_H+i *400_H	PNi	SV, E, PNi	Application Reset	4011
CAN2_Ni_INTRSIG (i=0-3)	Node i Interrupt Signalling Register	00128_H+i *400_H	PNi	nBE	Kernel Reset	4011
CAN2_Ni_G0INTR (i=0-3)	Node i Interrupt routing for Group 0	$0012C_H$+ i*400_H	PNi	PNi	Kernel Reset	4013
CAN2_Ni_G1INTR (i=0-3)	Node i Interrupt routing for Group 1	00130_H+i *400_H	PNi	PNi	Kernel Reset	4014
CAN2_Ni_G2INTR (i=0-3)	Node i Interrupt routing for Group 2	00134_H+i *400_H	PNi	PNi	Kernel Reset	4015

(table continues...)

Table 1031 (continued) Register overview - CAN2 domain SFR (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CAN2_Ni_TIMER_CCR (i=0-3)	Node i Timer Clock Control Register	00138_H+i*400_H	PNi	PNi	Kernel Reset	4015
CAN2_Ni_TIMER_TXTRIG0 (i=0-3)	Node i Timer Transmit Trigger 0 Register	$0013C_H$+i*400_H	PNi	PNi	Kernel Reset	4016
CAN2_Ni_TIMER_TXTRIG1 (i=0-3)	Node i Timer Transmit Trigger 1 Register	00140_H+i*400_H	PNi	PNi	Kernel Reset	4017
CAN2_Ni_TIMER_TXTRIG2 (i=0-3)	Node i Timer Transmit Trigger 2 Register	00144_H+i*400_H	PNi	PNi	Kernel Reset	4018
CAN2_Ni_TIMER_RXTOUT (i=0-3)	Node i Timer Receive Timeout Register	00148_H+i*400_H	PNi	PNi	Kernel Reset	4018
CAN2_Ni_PORTCTRL (i=0-3)	Node i Port Control Register	$0014C_H$+i*400_H	PNi	PNi	Kernel Reset	4019
CAN2_Ni_CRE_CONFIG (i=0-3)	Node i CRE Configuration Register	00150_H+i*400_H	PNi	SV, E, PNi	Kernel Reset	4158
CAN2_Ni_CRE_CONFIGADR (i=0-3)	Node i CRE Configuration Start Address	00154_H+i*400_H	PNi	SV, E, PNi	Kernel Reset	4022
CAN2_Ni_CRE_HBUF_RXz_CONFIG (i=0-3;z=0-1)	Node i Receive Host Buffer z Configuration	00158_H+i*400_H+z*8	PNi	PNi	Kernel Reset	4022
CAN2_Ni_CRE_HBUF_RXz_STAT (i=0-3;z=0-1)	Node i Receive Host Buffer z Status	$0015C_H$+i*400_H+z*8	PNi	PNi	Kernel Reset	4024
CAN2_Ni_CRE_HBUF_TXz_CONFIG (i=0-3;z=0-1)	Node i Transmit Host Buffer z Configuration	00168_H+i*400_H+z*8	PNi	PNi	Kernel Reset	4025
CAN2_Ni_CRE_HBUF_TXz_STAT (i=0-3;z=0-1)	Node i Transmit Host Buffer z Status	$0016C_H$+i*400_H+z*8	PNi	PNi	Kernel Reset	4026
CAN2_Ni_CRE_IR (i=0-3)	Node i CRE Interrupt Register	$0018C_H$+i*400_H	PNi	PNi	Kernel Reset	4028

(table continues...)

Table 1031 (continued) Register overview - CAN2 domain SFR (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CAN2_Ni_IDMU_FRTCONFIG (i=0-3)	Node i Frame Rate Measure Table Configuration	00190_H+i*400_H	PNi	SV, E, PNi	Kernel Reset	4029
CAN2_Ni_IDMU_RXTPCFG (i=0-3)	Node i Rx Throughput Measure configuration	00194_H+i*400_H	PNi	SV, E, PNi	Kernel Reset	4030
CAN2_Ni_ERRCTRL (i=0-3)	Node i CRE Error control register	00198_H+i*400_H	P4	SV, E, P4	Kernel Reset	4031
CAN2_Ni_CREL (i=0-3)	Node i Core Release Register	00200_H+i*400_H	PNi	nBE	Kernel Reset	4032
CAN2_Ni_ENDN (i=0-3)	Node i Endian Register	00204_H+i*400_H	PNi	nBE	Kernel Reset	4033
CAN2_Ni_DBTP (i=0-3)	Node i Data Bit Timing & Prescaler Register	$0020C_H$+i*400_H	PNi	PNi	4034	4034
CAN2_Ni_TEST (i=0-3)	Node i Test Register	00210_H+i*400_H	PNi	PNi	4035	4035
CAN2_Ni_RWD (i=0-3)	Node i RAM Watchdog	00214_H+i*400_H	PNi	PNi	Kernel Reset	4036
CAN2_Ni_CCCR (i=0-3)	Node i CC Control Register	00218_H+i*400_H	PNi	PNi	4037	4037
CAN2_Ni_NBTP (i=0-3)	Node i Nominal Bit Timing & Prescaler Register	$0021C_H$+i*400_H	PNi	PNi	4040	4040
CAN2_Ni_TSCC (i=0-3)	Node i Timestamp Counter Configuration	00220_H+i*400_H	PNi	PNi	Kernel Reset	4041
CAN2_Ni_TSCV (i=0-3)	Node i Timestamp Counter Value	00224_H+i*400_H	PNi	PNi	Kernel Reset	4042
CAN2_Ni_TOCC (i=0-3)	Node i Timeout Counter Configuration	00228_H+i*400_H	PNi	PNi	Kernel Reset	4043
CAN2_Ni_TOCV (i=0-3)	Node i Timeout Counter Value	$0022C_H$+i*400_H	PNi	PNi	Kernel Reset	4044
CAN2_Ni_ECR (i=0-3)	Node i Error Counter Register	00240_H+i*400_H	PNi	nBE	Kernel Reset	4045
CAN2_Ni_PSR (i=0-3)	Node i Protocol Status Register	00244_H+i*400_H	PNi	nBE	Kernel Reset	4045
CAN2_Ni_TDCR (i=0-3)	Node i Transmitter Delay Compensation Register	00248_H+i*400_H	PNi	PNi	Kernel Reset	4049
CAN2_Ni_IR (i=0-3)	Node i Interrupt Register	00250_H+i*400_H	PNi	PNi	Kernel Reset	4050

(table continues...)

Table 1031 (continued) Register overview - CAN2 domain SFR (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CAN2_Ni_IE (i=0-3)	Node i Interrupt Enable	00254_H+i *400_H	PNi	PNi	Kernel Reset	4053
CAN2_Ni_GFC (i=0-3)	Node i Global Filter Configuration	00280_H+i *400_H	PNi	PNi	Kernel Reset	4056
CAN2_Ni_SIDFC (i=0-3)	Node i Standard ID Filter Configuration	00284_H+i *400_H	PNi	PNi	4057	4057
CAN2_Ni_XIDFC (i=0-3)	Node i Extended ID Filter Configuration	00288_H+i *400_H	PNi	PNi	Kernel Reset	4058
CAN2_Ni_XIDAM (i=0-3)	Node i Extended ID AND Mask	00290_H+i *400_H	PNi	PNi	Kernel Reset	4058
CAN2_Ni_HPMS (i=0-3)	Node i High Priority Message Status	00294_H+i *400_H	PNi	nBE	Kernel Reset	4059
CAN2_Ni_NDAT1 (i=0-3)	Node i New Data 1	00298_H+i *400_H	PNi	PNi	Kernel Reset	4060
CAN2_Ni_NDAT2 (i=0-3)	Node i New Data 2	$0029C_H$+ i*400_H	PNi	PNi	Kernel Reset	4060
CAN2_Ni_RXF0C (i=0-3)	Node i Rx FIFO 0 Configuration	$002A0_H$+ i*400_H	PNi	PNi	Kernel Reset	4061
CAN2_Ni_RXF0S (i=0-3)	Node i Rx FIFO 0 Status	$002A4_H$+ i*400_H	PNi	nBE	Kernel Reset	4062
CAN2_Ni_RXF0A (i=0-3)	Node i Rx FIFO 0 Acknowledge	$002A8_H$+ i*400_H	PNi	PNi	Kernel Reset	4063
CAN2_Ni_RXBC (i=0-3)	Node i Rx Buffer Configuration	$002AC_H$+ i*400_H	PNi	PNi	4063	4063
CAN2_Ni_RXF1C (i=0-3)	Node i Rx FIFO 1 Configuration	$002B0_H$+ i*400_H	PNi	PNi	Kernel Reset	4064
CAN2_Ni_RXF1S (i=0-3)	Node i Rx FIFO 1 Status	$002B4_H$+ i*400_H	PNi	nBE	Kernel Reset	4065
CAN2_Ni_RXF1A (i=0-3)	Node i Rx FIFO 1 Acknowledge	$002B8_H$+ i*400_H	PNi	PNi	Kernel Reset	4066
CAN2_Ni_RXESC (i=0-3)	Node i Rx Buffer/FIFO Element Size Configuration	$002BC_H$+ i*400_H	PNi	PNi	Kernel Reset	4067
CAN2_Ni_TXBC (i=0-3)	Node i Tx Buffer Configuration	$002C0_H$+ i*400_H	PNi	PNi	4068	4068
CAN2_Ni_TXFQS (i=0-3)	Node i Tx FIFO/Queue Status	$002C4_H$+ i*400_H	PNi	nBE	Kernel Reset	4069
CAN2_Ni_TXESC (i=0-3)	Node i Tx Buffer Element Size Configuration	$002C8_H$+ i*400_H	PNi	PNi	4070	4070

(table continues...)

Table 1031 (continued) Register overview - CAN2 domain SFR (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CAN2_Ni_TXBRP (i=0-3)	Node i Tx Buffer Request Pending	$002CC_H$+ i*400_H	PNi	PNi	Kernel Reset	4072
CAN2_Ni_TXBAR (i=0-3)	Node i Tx Buffer Add Request	$002D0_H$+ i*400_H	PNi	PNi	Kernel Reset	4073
CAN2_Ni_TXBCR (i=0-3)	Node i Tx Buffer Cancellation Request	$002D4_H$+ i*400_H	PNi	PNi	Kernel Reset	4073
CAN2_Ni_TXBTO (i=0-3)	Node i Tx Buffer Transmission Occurred	$002D8_H$+ i*400_H	PNi	nBE	Kernel Reset	4074
CAN2_Ni_TXBCF (i=0-3)	Node i Tx Buffer Cancellation Finished	$002DC_H$+ i*400_H	PNi	nBE	Kernel Reset	4074
CAN2_Ni_TXBTIE (i=0-3)	Node i Tx Buffer Transmission Interrupt Enable	$002E0_H$+i *400_H	PNi	PNi	Kernel Reset	4075
CAN2_Ni_TXBCIE (i=0-3)	Node i Tx Buffer Cancellation Finished Interrupt Enable	$002E4_H$+i *400_H	PNi	PNi	Kernel Reset	4075
CAN2_Ni_TXEFC (i=0-3)	Node i Tx Event FIFO Configuration	$002F0_H$+i *400_H	PNi	PNi	Kernel Reset	4076
CAN2_Ni_TXEFS (i=0-3)	Node i Tx Event FIFO Status	$002F4_H$+i *400_H	PNi	nBE	Kernel Reset	4077
CAN2_Ni_TXEFA (i=0-3)	Node i Tx Event FIFO Acknowledge	$002F8_H$+i *400_H	PNi	PNi	Kernel Reset	4077
CAN2_Ni_TSU_CREL (i=0-3)	Node i TSU Core Release Register	00360_H+i *400_H	PNi	nBE	Kernel Reset	4078
CAN2_Ni_TSU_TSCFG (i=0) (i=1-3)	Node i Timestamp Configuration	00364_H+i *400_H	PNi	PNi	Kernel Reset	4078
CAN2_Ni_TSU_TSS1 (i=0-3)	Node i Timestamp Status 1	00368_H+i *400_H	PNi	nBE	Kernel Reset	4080
CAN2_Ni_TSU_TSS2 (i=0-3)	Node i Timestamp Status 2	$0036C_H$+ i*400_H	PNi	nBE	Kernel Reset	4081
CAN2_Ni_TSU_TSm (i=0-3;m=0-15)	Node i Timestamp m	00370_H+i *400_H+m *4	PNi	nBE	Kernel Reset	4082
CAN2_Ni_TSU_ATB (i=0-3)	Node i Actual Timebase	$003B0_H$+ i*400_H	PNi	U	Kernel Reset	4082

21.11.4.7 Register overview - CAN3 domain SFR (ascending offset address)

Table 1032 Register overview - CAN3 domain SFR (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CAN3_CLC	Clock Control Register	00000_H	P4	P4, SV, E	Application Reset	4154
CAN3_OCS	OCDS Control and Status Register	00004_H	P4	SV, P4	Debug Reset	3990
CAN3_ID	Module Identification Register	00008_H	P4	BE	PowerOn Reset	3991
CAN3_RST_CTRLA	Reset Control Register A	$0000C_H$	P4	P4, SV, E	Application Reset	3991
CAN3_RST_CTRLB	Reset Control Register B	00010_H	P4	P4, SV, E	Application Reset	3992
CAN3_RST_STAT	Reset Status Register	00014_H	P4	BE	Application Reset	3993
CAN3_PROTE	PROT Register Endinit	00018_H	U	SV, PROT	Application Reset	3993
CAN3_PROTSE	PROT Register Safe Endinit	$0001C_H$	U	SV, PROT	Application Reset	3995
CAN3_ACCEN_WRA	Write access enable register A	00030_H	U	SE, SV	Application Reset	3997
CAN3_ACCEN_WRB	Write access enable register B	00034_H	U	SE, SV	Application Reset	3997
CAN3_ACCEN_RDA	Read access enable register A	00038_H	U	SE, SV	Application Reset	3998
CAN3_ACCEN_RDB	Read access enable register B	$0003C_H$	U	SE, SV	Application Reset	3998
CAN3_ACCEN_VM	VM access enable register	00040_H	U	SE, SV	Application Reset	3999
CAN3_ACCEN_PRS	PRS access enable register	00044_H	U	SE, SV	Application Reset	3999
CAN3_MCR	Module Control Register	00070_H	P4	P4	Kernel Reset	4156
CAN3_WDT	CRE Watchdog timer register	00088_H	P4	SV, E, P4	Kernel Reset	4006
CAN3_Ni_ACCEN_WRA (i=0-3)	Node i Write access enable register A	00100_H+i *400_H	U	SE, SV	Application Reset	4007
CAN3_Ni_ACCEN_WRB (i=0-3)	Node i Write access enable register B	00104_H+i *400_H	U	SE, SV	Application Reset	4008
CAN3_Ni_ACCEN_RDA (i=0-3)	Node i Read access enable register A	00108_H+i *400_H	U	SE, SV	Application Reset	4008

(table continues...)

Table 1032 (continued) Register overview - CAN3 domain SFR (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CAN3_Ni_ACCEN_RDB (i=0-3)	Node i Read access enable register B	$0010C_H$+i*400_H	U	SE, SV	Application Reset	4009
CAN3_Ni_ACCEN_VM (i=0-3)	Node i VM access enable register	00110_H+i*400_H	U	SE, SV	Application Reset	4009
CAN3_Ni_ACCEN_PRS (i=0-3)	Node i PRS access enable register	00114_H+i*400_H	U	SE, SV	Application Reset	4010
CAN3_Ni_STARTADR (i=0-3)	Node i Start Address	00120_H+i*400_H	PNi	SV, E, PNi	Application Reset	4010
CAN3_Ni_ENDADR (i=0-3)	Node i End Address	00124_H+i*400_H	PNi	SV, E, PNi	Application Reset	4011
CAN3_Ni_INTRSIG (i=0-3)	Node i Interrupt Signalling Register	00128_H+i*400_H	PNi	nBE	Kernel Reset	4011
CAN3_Ni_G0INTR (i=0-3)	Node i Interrupt routing for Group 0	$0012C_H$+i*400_H	PNi	PNi	Kernel Reset	4013
CAN3_Ni_G1INTR (i=0-3)	Node i Interrupt routing for Group 1	00130_H+i*400_H	PNi	PNi	Kernel Reset	4014
CAN3_Ni_G2INTR (i=0-3)	Node i Interrupt routing for Group 2	00134_H+i*400_H	PNi	PNi	Kernel Reset	4015
CAN3_Ni_TIMER_CCR (i=0-3)	Node i Timer Clock Control Register	00138_H+i*400_H	PNi	PNi	Kernel Reset	4015
CAN3_Ni_TIMER_TXTRIG0 (i=0-3)	Node i Timer Transmit Trigger 0 Register	$0013C_H$+i*400_H	PNi	PNi	Kernel Reset	4016
CAN3_Ni_TIMER_TXTRIG1 (i=0-3)	Node i Timer Transmit Trigger 1 Register	00140_H+i*400_H	PNi	PNi	Kernel Reset	4017
CAN3_Ni_TIMER_TXTRIG2 (i=0-3)	Node i Timer Transmit Trigger 2 Register	00144_H+i*400_H	PNi	PNi	Kernel Reset	4018
CAN3_Ni_TIMER_RXTOUT (i=0-3)	Node i Timer Receive Timeout Register	00148_H+i*400_H	PNi	PNi	Kernel Reset	4018

(table continues...)

Table 1032 (continued) Register overview - CAN3 domain SFR (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CAN3_Ni_PORTCTRL (i=0-3)	Node i Port Control Register	$0014C_H$+i*400_H	PNi	PNi	Kernel Reset	4019
CAN3_Ni_CRE_CONFIG (i=0-3)	Node i CRE Configuration Register	00150_H+i*400_H	PNi	SV, E, PNi	Kernel Reset	4158
CAN3_Ni_CRE_CONFIGADR (i=0-3)	Node i CRE Configuration Start Address	00154_H+i*400_H	PNi	SV, E, PNi	Kernel Reset	4022
CAN3_Ni_CRE_HBUF_RXz_CONFIG (i=0-3;z=0-1)	Node i Receive Host Buffer z Configuration	00158_H+i*400_H+z*8	PNi	PNi	Kernel Reset	4022
CAN3_Ni_CRE_HBUF_RXz_STAT (i=0-3;z=0-1)	Node i Receive Host Buffer z Status	$0015C_H$+i*400_H+z*8	PNi	PNi	Kernel Reset	4024
CAN3_Ni_CRE_HBUF_TXz_CONFIG (i=0-3;z=0-1)	Node i Transmit Host Buffer z Configuration	00168_H+i*400_H+z*8	PNi	PNi	Kernel Reset	4025
CAN3_Ni_CRE_HBUF_TXz_STAT (i=0-3;z=0-1)	Node i Transmit Host Buffer z Status	$0016C_H$+i*400_H+z*8	PNi	PNi	Kernel Reset	4026
CAN3_Ni_CRE_IR (i=0-3)	Node i CRE Interrupt Register	$0018C_H$+i*400_H	PNi	PNi	Kernel Reset	4028
CAN3_Ni_IDMU_FRTCONFIG (i=0-3)	Node i Frame Rate Measure Table Configuration	00190_H+i*400_H	PNi	SV, E, PNi	Kernel Reset	4029
CAN3_Ni_IDMU_RXTPCFG (i=0-3)	Node i Rx Throughput Measure configuration	00194_H+i*400_H	PNi	SV, E, PNi	Kernel Reset	4030
CAN3_Ni_ERRCTRL (i=0-3)	Node i CRE Error control register	00198_H+i*400_H	P4	SV, E, P4	Kernel Reset	4031
CAN3_Ni_CREL (i=0-3)	Node i Core Release Register	00200_H+i*400_H	PNi	nBE	Kernel Reset	4032
CAN3_Ni_ENDN (i=0-3)	Node i Endian Register	00204_H+i*400_H	PNi	nBE	Kernel Reset	4033
CAN3_Ni_DBTP (i=0-3)	Node i Data Bit Timing & Prescaler Register	$0020C_H$+i*400_H	PNi	PNi	4034	4034

(table continues...)

Table 1032 (continued) Register overview - CAN3 domain SFR (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CAN3_Ni_TEST (i=0-3)	Node i Test Register	00210_H+i *400_H	PNi	PNi	4035	4035
CAN3_Ni_RWD (i=0-3)	Node i RAM Watchdog	00214_H+i *400_H	PNi	PNi	Kernel Reset	4036
CAN3_Ni_CCCR (i=0-3)	Node i CC Control Register	00218_H+i *400_H	PNi	PNi	4037	4037
CAN3_Ni_NBTP (i=0-3)	Node i Nominal Bit Timing & Prescaler Register	$0021C_H$+ i*400_H	PNi	PNi	4040	4040
CAN3_Ni_TSCC (i=0-3)	Node i Timestamp Counter Configuration	00220_H+i *400_H	PNi	PNi	Kernel Reset	4041
CAN3_Ni_TSCV (i=0-3)	Node i Timestamp Counter Value	00224_H+i *400_H	PNi	PNi	Kernel Reset	4042
CAN3_Ni_TOCC (i=0-3)	Node i Timeout Counter Configuration	00228_H+i *400_H	PNi	PNi	Kernel Reset	4043
CAN3_Ni_TOCV (i=0-3)	Node i Timeout Counter Value	$0022C_H$+ i*400_H	PNi	PNi	Kernel Reset	4044
CAN3_Ni_ECR (i=0-3)	Node i Error Counter Register	00240_H+i *400_H	PNi	nBE	Kernel Reset	4045
CAN3_Ni_PSR (i=0-3)	Node i Protocol Status Register	00244_H+i *400_H	PNi	nBE	Kernel Reset	4045
CAN3_Ni_TDCR (i=0-3)	Node i Transmitter Delay Compensation Register	00248_H+i *400_H	PNi	PNi	Kernel Reset	4049
CAN3_Ni_IR (i=0-3)	Node i Interrupt Register	00250_H+i *400_H	PNi	PNi	Kernel Reset	4050
CAN3_Ni_IE (i=0-3)	Node i Interrupt Enable	00254_H+i *400_H	PNi	PNi	Kernel Reset	4053
CAN3_Ni_GFC (i=0-3)	Node i Global Filter Configuration	00280_H+i *400_H	PNi	PNi	Kernel Reset	4056
CAN3_Ni_SIDFC (i=0-3)	Node i Standard ID Filter Configuration	00284_H+i *400_H	PNi	PNi	4057	4057
CAN3_Ni_XIDFC (i=0-3)	Node i Extended ID Filter Configuration	00288_H+i *400_H	PNi	PNi	Kernel Reset	4058
CAN3_Ni_XIDAM (i=0-3)	Node i Extended ID AND Mask	00290_H+i *400_H	PNi	PNi	Kernel Reset	4058
CAN3_Ni_HPMS (i=0-3)	Node i High Priority Message Status	00294_H+i *400_H	PNi	nBE	Kernel Reset	4059
CAN3_Ni_NDAT1 (i=0-3)	Node i New Data 1	00298_H+i *400_H	PNi	PNi	Kernel Reset	4060

(table continues...)

Table 1032 (continued) Register overview - CAN3 domain SFR (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CAN3_Ni_NDAT2 (i=0-3)	Node i New Data 2	$0029C_H$+ i*400_H	PNi	PNi	Kernel Reset	4060
CAN3_Ni_RXF0C (i=0-3)	Node i Rx FIFO 0 Configuration	$002A0_H$+ i*400_H	PNi	PNi	Kernel Reset	4061
CAN3_Ni_RXF0S (i=0-3)	Node i Rx FIFO 0 Status	$002A4_H$+ i*400_H	PNi	nBE	Kernel Reset	4062
CAN3_Ni_RXF0A (i=0-3)	Node i Rx FIFO 0 Acknowledge	$002A8_H$+ i*400_H	PNi	PNi	Kernel Reset	4063
CAN3_Ni_RXBC (i=0-3)	Node i Rx Buffer Configuration	$002AC_H$+ i*400_H	PNi	PNi	4063	4063
CAN3_Ni_RXF1C (i=0-3)	Node i Rx FIFO 1 Configuration	$002B0_H$+ i*400_H	PNi	PNi	Kernel Reset	4064
CAN3_Ni_RXF1S (i=0-3)	Node i Rx FIFO 1 Status	$002B4_H$+ i*400_H	PNi	nBE	Kernel Reset	4065
CAN3_Ni_RXF1A (i=0-3)	Node i Rx FIFO 1 Acknowledge	$002B8_H$+ i*400_H	PNi	PNi	Kernel Reset	4066
CAN3_Ni_RXESC (i=0-3)	Node i Rx Buffer/FIFO Element Size Configuration	$002BC_H$+ i*400_H	PNi	PNi	Kernel Reset	4067
CAN3_Ni_TXBC (i=0-3)	Node i Tx Buffer Configuration	$002C0_H$+ i*400_H	PNi	PNi	4068	4068
CAN3_Ni_TXFQS (i=0-3)	Node i Tx FIFO/Queue Status	$002C4_H$+ i*400_H	PNi	nBE	Kernel Reset	4069
CAN3_Ni_TXESC (i=0-3)	Node i Tx Buffer Element Size Configuration	$002C8_H$+ i*400_H	PNi	PNi	4070	4070
CAN3_Ni_TXBRP (i=0-3)	Node i Tx Buffer Request Pending	$002CC_H$+ i*400_H	PNi	PNi	Kernel Reset	4072
CAN3_Ni_TXBAR (i=0-3)	Node i Tx Buffer Add Request	$002D0_H$+ i*400_H	PNi	PNi	Kernel Reset	4073
CAN3_Ni_TXBCR (i=0-3)	Node i Tx Buffer Cancellation Request	$002D4_H$+ i*400_H	PNi	PNi	Kernel Reset	4073
CAN3_Ni_TXBTO (i=0-3)	Node i Tx Buffer Transmission Occurred	$002D8_H$+ i*400_H	PNi	nBE	Kernel Reset	4074
CAN3_Ni_TXBCF (i=0-3)	Node i Tx Buffer Cancellation Finished	$002DC_H$+ i*400_H	PNi	nBE	Kernel Reset	4074
CAN3_Ni_TXBTIE (i=0-3)	Node i Tx Buffer Transmission Interrupt Enable	$002E0_H$+i *400_H	PNi	PNi	Kernel Reset	4075

(table continues...)

Table 1032 (continued) Register overview - CAN3 domain SFR (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CAN3_Ni_TXBCIE (i=0-3)	Node i Tx Buffer Cancellation Finished Interrupt Enable	$002E4_H$+i *400_H	PNi	PNi	Kernel Reset	4075
CAN3_Ni_TXEFC (i=0-3)	Node i Tx Event FIFO Configuration	$002F0_H$+i *400_H	PNi	PNi	Kernel Reset	4076
CAN3_Ni_TXEFS (i=0-3)	Node i Tx Event FIFO Status	$002F4_H$+i *400_H	PNi	nBE	Kernel Reset	4077
CAN3_Ni_TXEFA (i=0-3)	Node i Tx Event FIFO Acknowledge	$002F8_H$+i *400_H	PNi	PNi	Kernel Reset	4077
CAN3_Ni_TSU_CREL (i=0-3)	Node i TSU Core Release Register	00360_H+i *400_H	PNi	nBE	Kernel Reset	4078
CAN3_Ni_TSU_TSCFG (i=0) (i=1-3)	Node i Timestamp Configuration	00364_H+i *400_H	PNi	PNi	Kernel Reset	4078
CAN3_Ni_TSU_TSS1 (i=0-3)	Node i Timestamp Status 1	00368_H+i *400_H	PNi	nBE	Kernel Reset	4080
CAN3_Ni_TSU_TSS2 (i=0-3)	Node i Timestamp Status 2	$0036C_H$+ i*400_H	PNi	nBE	Kernel Reset	4081
CAN3_Ni_TSU_TSm (i=0-3;m=0-15)	Node i Timestamp m	00370_H+i *400_H+m *4	PNi	nBE	Kernel Reset	4082
CAN3_Ni_TSU_ATB (i=0-3)	Node i Actual Timebase	$003B0_H$+ i*400_H	PNi	U	Kernel Reset	4082

21.11.4.8 Register overview - CAN4 domain SFR (ascending offset address)

Table 1033 Register overview - CAN4 domain SFR (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CAN4_CLC	Clock Control Register	00000_H	P4	P4, SV, E	Application Reset	4154
CAN4_OCS	OCDS Control and Status Register	00004_H	P4	SV, P4	Debug Reset	3990
CAN4_ID	Module Identification Register	00008_H	P4	BE	PowerOn Reset	3991

(table continues...)

Table 1033 (continued) Register overview - CAN4 domain SFR (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CAN4_RST_CTRLA	Reset Control Register A	$0000C_H$	P4	P4, SV, E	Application Reset	3991
CAN4_RST_CTRLB	Reset Control Register B	00010_H	P4	P4, SV, E	Application Reset	3992
CAN4_RST_STAT	Reset Status Register	00014_H	P4	BE	Application Reset	3993
CAN4_PROTE	PROT Register Endinit	00018_H	U	SV, PROT	Application Reset	3993
CAN4_PROTSE	PROT Register Safe Endinit	$0001C_H$	U	SV, PROT	Application Reset	3995
CAN4_ACCEN_WRA	Write access enable register A	00030_H	U	SE, SV	Application Reset	3997
CAN4_ACCEN_WRB	Write access enable register B	00034_H	U	SE, SV	Application Reset	3997
CAN4_ACCEN_RDA	Read access enable register A	00038_H	U	SE, SV	Application Reset	3998
CAN4_ACCEN_RDB	Read access enable register B	$0003C_H$	U	SE, SV	Application Reset	3998
CAN4_ACCEN_VM	VM access enable register	00040_H	U	SE, SV	Application Reset	3999
CAN4_ACCEN_PRS	PRS access enable register	00044_H	U	SE, SV	Application Reset	3999
CAN4_MCR	Module Control Register	00070_H	P4	P4	Kernel Reset	4156
CAN4_WDT	CRE Watchdog timer register	00088_H	P4	SV, E, P4	Kernel Reset	4006
CAN4_Ni_ACCEN_WRA (i=0-3)	Node i Write access enable register A	00100_H+i *400_H	U	SE, SV	Application Reset	4007
CAN4_Ni_ACCEN_WRB (i=0-3)	Node i Write access enable register B	00104_H+i *400_H	U	SE, SV	Application Reset	4008
CAN4_Ni_ACCEN_RDA (i=0-3)	Node i Read access enable register A	00108_H+i *400_H	U	SE, SV	Application Reset	4008
CAN4_Ni_ACCEN_RDB (i=0-3)	Node i Read access enable register B	$0010C_H$+ i*400_H	U	SE, SV	Application Reset	4009
CAN4_Ni_ACCEN_VM (i=0-3)	Node i VM access enable register	00110_H+i *400_H	U	SE, SV	Application Reset	4009

(table continues...)

Table 1033 (continued) Register overview - CAN4 domain SFR (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CAN4_Ni_ACCEN_PRS (i=0-3)	Node i PRS access enable register	00114_H+i*400_H	U	SE, SV	Application Reset	4010
CAN4_Ni_STARTADR (i=0-3)	Node i Start Address	00120_H+i*400_H	PNi	SV, E, PNi	Application Reset	4010
CAN4_Ni_ENDADR (i=0-3)	Node i End Address	00124_H+i*400_H	PNi	SV, E, PNi	Application Reset	4011
CAN4_Ni_INTRSIG (i=0-3)	Node i Interrupt Signalling Register	00128_H+i*400_H	PNi	nBE	Kernel Reset	4011
CAN4_Ni_G0INTR (i=0-3)	Node i Interrupt routing for Group 0	$0012C_H$+i*400_H	PNi	PNi	Kernel Reset	4013
CAN4_Ni_G1INTR (i=0-3)	Node i Interrupt routing for Group 1	00130_H+i*400_H	PNi	PNi	Kernel Reset	4014
CAN4_Ni_G2INTR (i=0-3)	Node i Interrupt routing for Group 2	00134_H+i*400_H	PNi	PNi	Kernel Reset	4015
CAN4_Ni_TIMER_CCR (i=0-3)	Node i Timer Clock Control Register	00138_H+i*400_H	PNi	PNi	Kernel Reset	4015
CAN4_Ni_TIMER_TXTRIG0 (i=0-3)	Node i Timer Transmit Trigger 0 Register	$0013C_H$+i*400_H	PNi	PNi	Kernel Reset	4016
CAN4_Ni_TIMER_TXTRIG1 (i=0-3)	Node i Timer Transmit Trigger 1 Register	00140_H+i*400_H	PNi	PNi	Kernel Reset	4017
CAN4_Ni_TIMER_TXTRIG2 (i=0-3)	Node i Timer Transmit Trigger 2 Register	00144_H+i*400_H	PNi	PNi	Kernel Reset	4018
CAN4_Ni_TIMER_RXTOUT (i=0-3)	Node i Timer Receive Timeout Register	00148_H+i*400_H	PNi	PNi	Kernel Reset	4018
CAN4_Ni_PORTCTRL (i=0-3)	Node i Port Control Register	$0014C_H$+i*400_H	PNi	PNi	Kernel Reset	4019
CAN4_Ni_CRE_CONFIG (i=0-3)	Node i CRE Configuration Register	00150_H+i*400_H	PNi	SV, E, PNi	Kernel Reset	4158

(table continues...)

Table 1033 (continued) Register overview - CAN4 domain SFR (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CAN4_Ni_CRE_CONFIGADR (i=0-3)	Node i CRE Configuration Start Address	00154_H+i*400_H	PNi	SV, E, PNi	Kernel Reset	4022
CAN4_Ni_CRE_HBUF_RXz_CONFIG (i=0-3;z=0-1)	Node i Receive Host Buffer z Configuration	00158_H+i*400_H+z*8	PNi	PNi	Kernel Reset	4022
CAN4_Ni_CRE_HBUF_RXz_STAT (i=0-3;z=0-1)	Node i Receive Host Buffer z Status	$0015C_H$+i*400_H+z*8	PNi	PNi	Kernel Reset	4024
CAN4_Ni_CRE_HBUF_TXz_CONFIG (i=0-3;z=0-1)	Node i Transmit Host Buffer z Configuration	00168_H+i*400_H+z*8	PNi	PNi	Kernel Reset	4025
CAN4_Ni_CRE_HBUF_TXz_STAT (i=0-3;z=0-1)	Node i Transmit Host Buffer z Status	$0016C_H$+i*400_H+z*8	PNi	PNi	Kernel Reset	4026
CAN4_Ni_CRE_IR (i=0-3)	Node i CRE Interrupt Register	$0018C_H$+i*400_H	PNi	PNi	Kernel Reset	4028
CAN4_Ni_IDMU_FRTCONFIG (i=0-3)	Node i Frame Rate Measure Table Configuration	00190_H+i*400_H	PNi	SV, E, PNi	Kernel Reset	4029
CAN4_Ni_IDMU_RXTPCFG (i=0-3)	Node i Rx Throughput Measure configuration	00194_H+i*400_H	PNi	SV, E, PNi	Kernel Reset	4030
CAN4_Ni_ERRCTRL (i=0-3)	Node i CRE Error control register	00198_H+i*400_H	P4	SV, E, P4	Kernel Reset	4031
CAN4_Ni_CREL (i=0-3)	Node i Core Release Register	00200_H+i*400_H	PNi	nBE	Kernel Reset	4032
CAN4_Ni_ENDN (i=0-3)	Node i Endian Register	00204_H+i*400_H	PNi	nBE	Kernel Reset	4033
CAN4_Ni_DBTP (i=0-3)	Node i Data Bit Timing & Prescaler Register	$0020C_H$+i*400_H	PNi	PNi	4034	4034
CAN4_Ni_TEST (i=0-3)	Node i Test Register	00210_H+i*400_H	PNi	PNi	4035	4035
CAN4_Ni_RWD (i=0-3)	Node i RAM Watchdog	00214_H+i*400_H	PNi	PNi	Kernel Reset	4036
CAN4_Ni_CCCR (i=0-3)	Node i CC Control Register	00218_H+i*400_H	PNi	PNi	4037	4037

(table continues...)

Table 1033 (continued) Register overview - CAN4 domain SFR (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CAN4_Ni_NBTP (i=0-3)	Node i Nominal Bit Timing & Prescaler Register	$0021C_H$+ i*400_H	PNi	PNi	4040	4040
CAN4_Ni_TSCC (i=0-3)	Node i Timestamp Counter Configuration	00220_H+i *400_H	PNi	PNi	Kernel Reset	4041
CAN4_Ni_TSCV (i=0-3)	Node i Timestamp Counter Value	00224_H+i *400_H	PNi	PNi	Kernel Reset	4042
CAN4_Ni_TOCC (i=0-3)	Node i Timeout Counter Configuration	00228_H+i *400_H	PNi	PNi	Kernel Reset	4043
CAN4_Ni_TOCV (i=0-3)	Node i Timeout Counter Value	$0022C_H$+ i*400_H	PNi	PNi	Kernel Reset	4044
CAN4_Ni_ECR (i=0-3)	Node i Error Counter Register	00240_H+i *400_H	PNi	nBE	Kernel Reset	4045
CAN4_Ni_PSR (i=0-3)	Node i Protocol Status Register	00244_H+i *400_H	PNi	nBE	Kernel Reset	4045
CAN4_Ni_TDCR (i=0-3)	Node i Transmitter Delay Compensation Register	00248_H+i *400_H	PNi	PNi	Kernel Reset	4049
CAN4_Ni_IR (i=0-3)	Node i Interrupt Register	00250_H+i *400_H	PNi	PNi	Kernel Reset	4050
CAN4_Ni_IE (i=0-3)	Node i Interrupt Enable	00254_H+i *400_H	PNi	PNi	Kernel Reset	4053
CAN4_Ni_GFC (i=0-3)	Node i Global Filter Configuration	00280_H+i *400_H	PNi	PNi	Kernel Reset	4056
CAN4_Ni_SIDFC (i=0-3)	Node i Standard ID Filter Configuration	00284_H+i *400_H	PNi	PNi	4057	4057
CAN4_Ni_XIDFC (i=0-3)	Node i Extended ID Filter Configuration	00288_H+i *400_H	PNi	PNi	Kernel Reset	4058
CAN4_Ni_XIDAM (i=0-3)	Node i Extended ID AND Mask	00290_H+i *400_H	PNi	PNi	Kernel Reset	4058
CAN4_Ni_HPMS (i=0-3)	Node i High Priority Message Status	00294_H+i *400_H	PNi	nBE	Kernel Reset	4059
CAN4_Ni_NDAT1 (i=0-3)	Node i New Data 1	00298_H+i *400_H	PNi	PNi	Kernel Reset	4060
CAN4_Ni_NDAT2 (i=0-3)	Node i New Data 2	$0029C_H$+ i*400_H	PNi	PNi	Kernel Reset	4060
CAN4_Ni_RXF0C (i=0-3)	Node i Rx FIFO 0 Configuration	$002A0_H$+ i*400_H	PNi	PNi	Kernel Reset	4061
CAN4_Ni_RXF0S (i=0-3)	Node i Rx FIFO 0 Status	$002A4_H$+ i*400_H	PNi	nBE	Kernel Reset	4062

(table continues...)

Table 1033 (continued) Register overview - CAN4 domain SFR (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CAN4_Ni_RXF0A (i=0-3)	Node i Rx FIFO 0 Acknowledge	$002A8_H$+ i*400_H	PNi	PNi	Kernel Reset	4063
CAN4_Ni_RXBC (i=0-3)	Node i Rx Buffer Configuration	$002AC_H$+ i*400_H	PNi	PNi	4063	4063
CAN4_Ni_RXF1C (i=0-3)	Node i Rx FIFO 1 Configuration	$002B0_H$+ i*400_H	PNi	PNi	Kernel Reset	4064
CAN4_Ni_RXF1S (i=0-3)	Node i Rx FIFO 1 Status	$002B4_H$+ i*400_H	PNi	nBE	Kernel Reset	4065
CAN4_Ni_RXF1A (i=0-3)	Node i Rx FIFO 1 Acknowledge	$002B8_H$+ i*400_H	PNi	PNi	Kernel Reset	4066
CAN4_Ni_RXESC (i=0-3)	Node i Rx Buffer/FIFO Element Size Configuration	$002BC_H$+ i*400_H	PNi	PNi	Kernel Reset	4067
CAN4_Ni_TXBC (i=0-3)	Node i Tx Buffer Configuration	$002C0_H$+ i*400_H	PNi	PNi	4068	4068
CAN4_Ni_TXFQS (i=0-3)	Node i Tx FIFO/Queue Status	$002C4_H$+ i*400_H	PNi	nBE	Kernel Reset	4069
CAN4_Ni_TXESC (i=0-3)	Node i Tx Buffer Element Size Configuration	$002C8_H$+ i*400_H	PNi	PNi	4070	4070
CAN4_Ni_TXBRP (i=0-3)	Node i Tx Buffer Request Pending	$002CC_H$+ i*400_H	PNi	PNi	Kernel Reset	4072
CAN4_Ni_TXBAR (i=0-3)	Node i Tx Buffer Add Request	$002D0_H$+ i*400_H	PNi	PNi	Kernel Reset	4073
CAN4_Ni_TXBCR (i=0-3)	Node i Tx Buffer Cancellation Request	$002D4_H$+ i*400_H	PNi	PNi	Kernel Reset	4073
CAN4_Ni_TXBTO (i=0-3)	Node i Tx Buffer Transmission Occurred	$002D8_H$+ i*400_H	PNi	nBE	Kernel Reset	4074
CAN4_Ni_TXBCF (i=0-3)	Node i Tx Buffer Cancellation Finished	$002DC_H$+ i*400_H	PNi	nBE	Kernel Reset	4074
CAN4_Ni_TXBTIE (i=0-3)	Node i Tx Buffer Transmission Interrupt Enable	$002E0_H$+i *400_H	PNi	PNi	Kernel Reset	4075
CAN4_Ni_TXBCIE (i=0-3)	Node i Tx Buffer Cancellation Finished Interrupt Enable	$002E4_H$+i *400_H	PNi	PNi	Kernel Reset	4075
CAN4_Ni_TXEFC (i=0-3)	Node i Tx Event FIFO Configuration	$002F0_H$+i *400_H	PNi	PNi	Kernel Reset	4076
CAN4_Ni_TXEFS (i=0-3)	Node i Tx Event FIFO Status	$002F4_H$+i *400_H	PNi	nBE	Kernel Reset	4077

(table continues...)

Table 1033 (continued) Register overview - CAN4 domain SFR (ascending offset address)

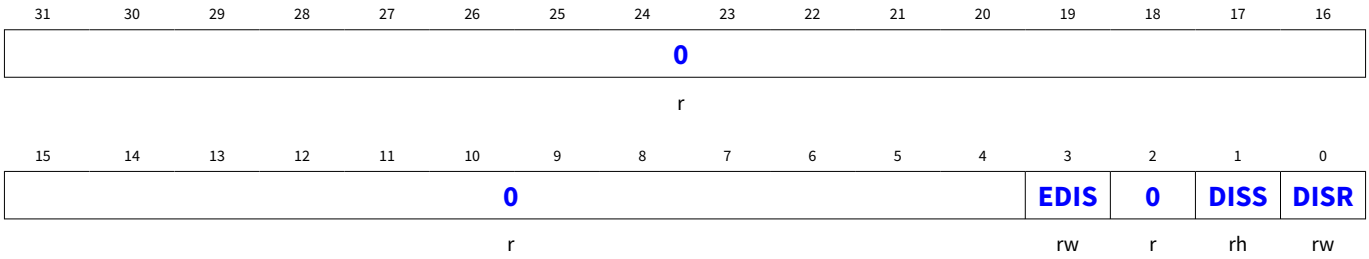
Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CAN4_Ni_TXEFA (i=0-3)	Node i Tx Event FIFO Acknowledge	$002F8_H$+i*400_H	PNi	PNi	Kernel Reset	4077
CAN4_Ni_TSU_CREL (i=0-3)	Node i TSU Core Release Register	00360_H+i*400_H	PNi	nBE	Kernel Reset	4078
CAN4_Ni_TSU_TSCFG (i=0) (i=1-3)	Node i Timestamp Configuration	00364_H+i*400_H	PNi	PNi	Kernel Reset	4078
CAN4_Ni_TSU_TSS1 (i=0-3)	Node i Timestamp Status 1	00368_H+i*400_H	PNi	nBE	Kernel Reset	4080
CAN4_Ni_TSU_TSS2 (i=0-3)	Node i Timestamp Status 2	$0036C_H$+i*400_H	PNi	nBE	Kernel Reset	4081
CAN4_Ni_TSU_TSm (i=0-3;m=0-15)	Node i Timestamp m	00370_H+i*400_H+m*4	PNi	nBE	Kernel Reset	4082
CAN4_Ni_TSU_ATB (i=0-3)	Node i Actual Timebase	$003B0_H$+i*400_H	PNi	U	Kernel Reset	4082

21.11.4.9 Clock Control Register (CANn=1-4)

The Clock Control Register CLC allows the programmer to adapt the functionality and power consumption of the module to the requirements of the application.

Register CLC controls the module clock signal and the reactivity to the sleep signal.

CAN1_CLC	Offset address:	00000_H
Clock Control Register	Application Reset value:	0000 0003_H
CAN2_CLC	Offset address:	00000_H
Clock Control Register	Application Reset value:	0000 0003_H
CAN3_CLC	Offset address:	00000_H
Clock Control Register	Application Reset value:	0000 0003_H
CAN4_CLC	Offset address:	00000_H
Clock Control Register	Application Reset value:	0000 0003_H



Field	Bits	Type	Description
DISR	0	rw	**Module Disable Request Bit** Used for enable/disable control of the module. The synchronous and asynchronous clock is switched on/off. Note that no register access is possible to any register while module is disabled. A disable request is granted, if the M_CAN clock is disabled, or all M_CAN nodes acknowledge the disable request. 0_B On request: enable the module clock 1_B Off request: stop the module clock
DISS	1	rh	**Module Disable Status Bit** 0_B Module clock is enabled 1_B Off: module is not clocked
EDIS	3	rw	**Sleep Mode Enable Control** Used to control the module's reaction to sleep mode. 0_B Sleep mode request is enabled and functional 1_B Module disregards the sleep mode control signal
0	2, 31:4	r	**Reserved** Read as 0; should be written with 0.

21.11.4.10 Module Control Register (CANn=1-4)

The Module Control Register MCR contains basic settings that determine the operation of the MCMCAN module.

The write access to the lowest byte of the MCR register becomes only valid, if and only if, MCR.CCCE and MCR.CI are already set during write access. To switch the clocks on or off, the bits of MCR.CCCE and MCR.CI have to be reset afterwards. Before this sequence hasn't taken place, no write access to the corresponding nodes, can be done.

***Note**: If the baud rate logic is supplied from an unstable clock source, or no clock at all, the CAN functionality is not guaranteed.*

To be able to change the clock settings the following programming sequence needs to be met:
uwTemp = CANn_MCR.U;
uwTemp |= (0xC0000000 | CLKSELx);
CANn_MCR.U = uwTemp;
uwTemp &= ~0xC0000000;
CANn_MCR.U = uwTemp;
The clock settings for CAN nodes becomes active.
To be able to start the RAM initialization, the following programming sequence need to be met:
CANn_MCR |= 0xC0000000;
Wait until CANn_MCR.RBUSY is 0b
Set CANn_MCR.RINIT to 0b
Set CANn_MCR.RINIT to 1b
Dummy read CANn_MCR
Wait until CANn_MCR.RBUSY is 0b
Set CANn_MCR.RINIT to 0b
CANn_MCR &= ~0xC0000000;
RAM initialization is finished

CAN1_MCR	Offset address:	00070_H
Module Control Register	Kernel Reset value:	$0000\ 0000_H$
CAN2_MCR	Offset address:	00070_H
Module Control Register	Kernel Reset value:	$0000\ 0000_H$
CAN3_MCR	Offset address:	00070_H
Module Control Register	Kernel Reset value:	$0000\ 0000_H$
CAN4_MCR	Offset address:	00070_H
Module Control Register	Kernel Reset value:	$0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CCCE	**CI**	**RINIT**	**RBUSY**	**0**				**0**							
rw	rw	rw	rh	r				r							

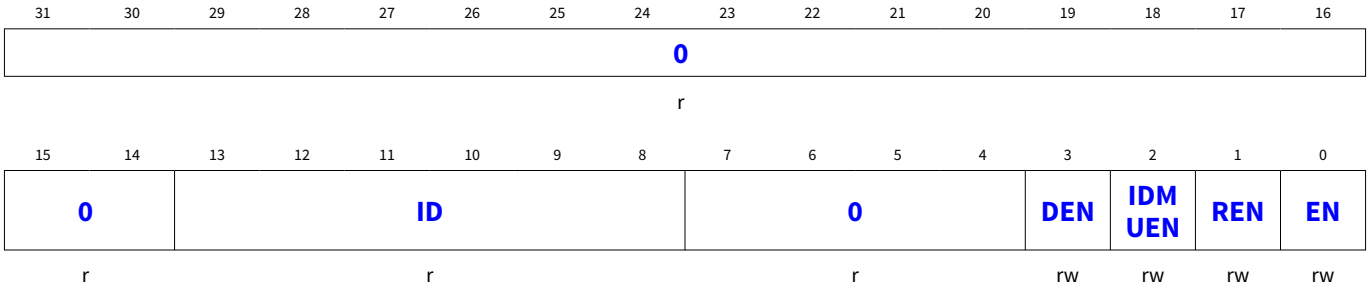
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								**CLKSEL3**		**CLKSEL2**		**CLKSEL1**		**CLKSEL0**	
r								rw		rw		rw		rw	

Field	Bits	Type	Description
CLKSEL0	1:0	rw	**Clock Select 0** This bit-field is MCR.CI and MCR.CCCE protected. 00_B No clock supplied 01_B The asynchronous clock source is switched on 10_B The synchronous clock source is switched on 11_B Both clock sources are switched on
CLKSEL1	3:2	rw	**Clock Select 1** This bit-field is MCR.CI and MCR.CCCE protected. 00_B No clock supplied 01_B The asynchronous clock source is switched on 10_B The synchronous clock source is switched on 11_B Both clock sources are switched on
CLKSEL2	5:4	rw	**Clock Select 2** This bit-field is MCR.CI and MCR.CCCE protected. 00_B No clock supplied 01_B The asynchronous clock source is switched on 10_B The synchronous clock source is switched on 11_B Both clock sources are switched on
CLKSEL3	7:6	rw	**Clock Select 3** This bit-field is MCR.CI and MCR.CCCE protected. 00_B No clock supplied 01_B The asynchronous clock source is switched on 10_B The synchronous clock source is switched on 11_B Both clock sources are switched on
RBUSY	28	rh	**RAM BUSY** This bit shows that the RAM Initialization is running. This bit is set back to 0b by hardware when the RAM intialization is completed.
RINIT	29	rw	**RAM Init** This bit is MCR.CI and MCR.CCCE protected. This bit starts the initialization of the RAM block to all 0x0. The RAM initialization is started only when this bit is changed from 0b to 1b and also RBUSY is 0b.
CI	30	rw	**Change Init** Needs to be set to enable and disable clocks. 0_B Change Init disabled 1_B Change Init enabled (takes effect with CCCE:=1)
CCCE	31	rw	**Clock and RAM Change Enable** Needs to be set to enable and disable the clocks. 0_B Clock and RAM Change disabled 1_B Clock and RAM Change enabled (takes effect with CI:=1)
0	23:8, 27:24	r	**Reserved** Shall read 0; shall be written with 0.

21.11.4.11 Node i CRE Configuration Register (CANn=1-4;i=0) (CANn=1-4;i=1) (CANn=1-2;i=2)

The Node i CRE Configuration Register Ni_CRE_CONFIG configures the Routing Engine functionality.

CAN1_Ni_CRE_CONFIG (i=0)	Offset address:	00150_H+i*400_H
Node i CRE Configuration Register	Kernel Reset value:	0000 0500_H
CAN2_Ni_CRE_CONFIG (i=0)	Offset address:	00150_H+i*400_H
Node i CRE Configuration Register	Kernel Reset value:	0000 0900_H
CAN3_Ni_CRE_CONFIG (i=0)	Offset address:	00150_H+i*400_H
Node i CRE Configuration Register	Kernel Reset value:	0000 $0D00_H$
CAN4_Ni_CRE_CONFIG (i=0)	Offset address:	00150_H+i*400_H
Node i CRE Configuration Register	Kernel Reset value:	0000 1100_H
CAN1_Ni_CRE_CONFIG (i=1)	Offset address:	00150_H+i*400_H
Node i CRE Configuration Register	Kernel Reset value:	0000 0600_H
CAN2_Ni_CRE_CONFIG (i=1)	Offset address:	00150_H+i*400_H
Node i CRE Configuration Register	Kernel Reset value:	0000 $0A00_H$
CAN3_Ni_CRE_CONFIG (i=1)	Offset address:	00150_H+i*400_H
Node i CRE Configuration Register	Kernel Reset value:	0000 $0E00_H$
CAN4_Ni_CRE_CONFIG (i=1)	Offset address:	00150_H+i*400_H
Node i CRE Configuration Register	Kernel Reset value:	0000 1200_H
CAN1_Ni_CRE_CONFIG (i=2)	Offset address:	00150_H+i*400_H
Node i CRE Configuration Register	Kernel Reset value:	0000 0700_H
CAN2_Ni_CRE_CONFIG (i=2)	Offset address:	00150_H+i*400_H
Node i CRE Configuration Register	Kernel Reset value:	0000 $0B00_H$



Field	Bits	Type	Description
EN	0	rw	**Enable** Note: When the EN is set, the respective host buffers are enabled. The RxFIFO0/1 and Tx FIFO/Queues shall not be used in this case. The frames are read/written from/to CRE host buffers instead. 0_B **DIS**: CRE function is disabled 1_B **EN**: CRE function is enabled. When CRE is enabled, then RxFIFO0/1 and TxFIFO/Queue host interface handling is also enabled when corresponding FIFO/Queue are enabled by CAN node

(table continues...)

(continued)

Field	Bits	Type	Description
REN	1	rw	**Routing Enable** This bit indicates that the Routing is enabled for the Source node. This bit is checked only at the Source node. This bit-field can be written only when Ni_CRE_CONFIG.EN = 1. Note: This bit should be enabled at the source node for correct routing 0_B **DIS**: Routing Disable The Routing function of CRE is disabled. Note: If Ni_CRE_CFG.EN is set, the corresponding host buffers are still enabled. 1_B **EN**: Routing Enable The Routing function of CRE is enabled for the Source CAN node
IDMUEN	2	rw	**IDMU Enable** This bit-field can be written only when Ni_CRE_CFG.EN = 1 0_B **DIS**: IDMU Disable The Intrusion detection measurement unit (IDMU) of CRE is disabled 1_B **EN**: IDMU Enable The Intrusion detection measurement unit (IDMU) of CRE is enabled
DEN	3	rw	**Destination Enable** This bit is checked only at the Destination node. This bit controls the CRE write of the CAN frame to TxFIFO Note: In case of external routing, the CAN frame remains in the Tx Host Buffer and there will be no re-trigger to the DRE if DEN is set to 0 after there has already been a trigger to the DRE. In case of internal routing, the CAN frame would be discarded if DEN is set to 0 0_B **DIS**: Destination Disable The CAN frame is discarded 1_B **EN**: Destination Enable The CRE will write the CAN frame to the TxFIFO
ID	13:8	r	**Unique ID** A Unique ID of the CAN Node i 1_H till 4_H - MCMCAN 0 CAN Node 0-Node 3 respectively 5_H till 8_H - MCMCAN 1 CAN Node 0-Node 3 respectively 9_H till C_H - MCMCAN 2 CAN Node 0-Node 3 respectively D_H till 10_H - MCMCAN 3 CAN Node 0-Node 3 respectively 11_H till 14_H - MCMCAN 4 CAN Node 0-Node 3 respectively
0	7:4, 31:14	r	**Reserved** Read as 0; shall be written with 0.

21.11.4.12 Node i CRE Configuration Register (CANn=1-4;i=3) (CANn=3-4;i=2)

CAN3_Ni_CRE_CONFIG (i=2)	Offset address:	$00150_H+i*400_H$
Node i CRE Configuration Register	Kernel Reset value:	$0000\ 0F00_H$
CAN4_Ni_CRE_CONFIG (i=2)	Offset address:	$00150_H+i*400_H$
Node i CRE Configuration Register	Kernel Reset value:	$0000\ 1300_H$
CAN1_Ni_CRE_CONFIG (i=3)	Offset address:	$00150_H+i*400_H$
Node i CRE Configuration Register	Kernel Reset value:	$0000\ 0800_H$
CAN2_Ni_CRE_CONFIG (i=3)	Offset address:	$00150_H+i*400_H$
Node i CRE Configuration Register	Kernel Reset value:	$0000\ 0C00_H$
CAN3_Ni_CRE_CONFIG (i=3)	Offset address:	$00150_H+i*400_H$
Node i CRE Configuration Register	Kernel Reset value:	$0000\ 1000_H$
CAN4_Ni_CRE_CONFIG (i=3)	Offset address:	$00150_H+i*400_H$
Node i CRE Configuration Register	Kernel Reset value:	$0000\ 1400_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		ID						0				DEN	IDM UEN	REN	EN
r		r						r				rw	rw	rw	rw

Field	Bits	Type	Description
EN	0	rw	**Enable** Note: When the EN is set, the respective host buffers are enabled. The RxFIFO0/1 and Tx FIFO/Queues shall not be used in this case. The frames are read/written from/to CRE host buffers instead. 0_B **DIS**: CRE function is disabled 1_B **EN**: CRE function is enabled. When CRE is enabled, then RxFIFO0/1 and TxFIFO/Queue host interface handling is also enabled when corresponding FIFO/Queue are enabled by CAN node
REN	1	rw	**Routing Enable** This bit indicates that the Routing is enabled for the Source node. This bit is checked only at the Source node. This bit-field can be written only when Ni_CRE_CONFIG.EN = 1. Note: This bit should be enabled at the source node for correct routing 0_B **DIS**: Routing Disable The Routing function of CRE is disabled. Note: If Ni_CRE_CFG.EN is set, the corresponding host buffers are still enabled. 1_B **EN**: Routing Enable The Routing function of CRE is enabled for the Source CAN node

(table continues...)

(continued)

Field	Bits	Type	Description
IDMUEN	2	rw	**IDMU Enable** This bit-field can be written only when Ni_CRE_CFG.EN = 1 0_B **DIS**: IDMU Disable The Intrusion detection measurement unit (IDMU) of CRE is disabled 1_B **EN**: IDMU Enable The Intrusion detection measurement unit (IDMU) of CRE is enabled
DEN	3	rw	**Destination Enable** This bit is checked only at the Destination node. This bit controls the CRE write of the CAN frame to TxFIFO Note: In case of external routing, the CAN frame remains in the Tx Host Buffer and there will be no re-trigger to the DRE if DEN is set to 0 after there has already been a trigger to the DRE. In case of internal routing, the CAN frame would be discarded if DEN is set to 0 0_B **DIS**: Destination Disable The CAN frame is discarded 1_B **EN**: Destination Enable The CRE will write the CAN frame to the TxFIFO
ID	13:8	r	**Unique ID** A Unique ID of the CAN Node i 1_H till 4_H - MCMCAN 0 CAN Node 0-Node 3 respectively 5_H till 8_H - MCMCAN 1 CAN Node 0-Node 3 respectively 9_H till C_H - MCMCAN 2 CAN Node 0-Node 3 respectively D_H till 10_H - MCMCAN 3 CAN Node 0-Node 3 respectively 11_H till 14_H - MCMCAN 4 CAN Node 0-Node 3 respectively
0	7:4, 31:14	r	**Reserved** Read as 0; shall be written with 0.

21.11.5 TC4Dx CAN connectivity

The table below lists interfaces to and from this functional block to other blocks in the device.

Table 1034 List of CAN interface signals

Interface signals	I/O	Description
CLOCK_CAN_fSPB	In	SPB clock input
CLOCK_CAN_fMCAN	In	MCAN asynchronous clock input
CLOCK_CAN_fMCANH	In	MCANH synchronous clock input
FPI_CAN_SIF	In	Slave interface
CPU0_CAN0_Nx_TRIG	In	STM Service Request 0 transmit trigger input for node x
CPU0_CAN1_Nx_TRIG	In	STM Service Request 0 transmit trigger input for node x
CPU0_CAN2_Nx_TRIG	In	STM Service Request 0 transmit trigger input for node x

(table continues...)

Table 1034 (continued) List of CAN interface signals

Interface signals	I/O	Description
CPU0_CAN3_Nx_TRIG	In	STM Service Request 0 transmit trigger input for node x
CPU0_CAN4_Nx_TRIG	In	STM Service Request 0 transmit trigger input for node x
EGTM_CAN0_Nx_TRIG[3:0]	In	eGTM transmit trigger input for node x (x=0-3)
EGTM_CAN1_Nx_TRIG[3:0]	In	eGTM transmit trigger input for node x (x=0-3)
EGTM_CAN2_Nx_TRIG[3:0]	In	eGTM transmit trigger input for node x (x=0-3)
EGTM_CAN3_Nx_TRIG[3:0]	In	eGTM transmit trigger input for node x (x=0-3)
EGTM_CAN4_Nx_TRIG[3:0]	In	eGTM transmit trigger input for node x (x=0-3)
PORTS_CAN0_node[3:0]_RXD[7:0]	In	CAN0 node x receive inputs
PORTS_CAN1_node[3:0]_RXD[7:0]	In	CAN1 node x receive inputs
PORTS_CAN2_node[3:0]_RXD[7:0]	In	CAN2 node x receive inputs
PORTS_CAN3_node[3:0]_RXD[7:0]	In	CAN3 node x receive inputs
PORTS_CAN4_node[3:0]_RXD[7:0]	In	CAN4 node x receive inputs
CAN0_PORTS_node[3:0]_TXD	Out	CAN0 node x transmit outputs
CAN1_PORTS_node[3:0]_TXD	Out	CAN1 node x transmit outputs
CAN2_PORTS_node[3:0]_TXD	Out	CAN2 node x transmit outputs
CAN3_PORTS_node[3:0]_TXD	Out	CAN3 node x transmit outputs
CAN4_PORTS_node[3:0]_TXD	Out	CAN4 node x transmit outputs
CAN0_IR_SRC_CANINT[15:0]	Out	CAN0 Service Request
CAN1_IR_SRC_CANINT[15:0]	Out	CAN1 Service Request
CAN2_IR_SRC_CANINT[15:0]	Out	CAN2 Service Request
CAN3_IR_SRC_CANINT[15:0]	Out	CAN3 Service Request
CAN4_IR_SRC_CANINT[15:0]	Out	CAN4 Service Request
CAN0_EGTM_SRC_CANINT[15:12]	Out	CAN0 Service Request to eGTM
CAN1_EGTM_SRC_CANINT[15:12]	Out	CAN1 Service Request to eGTM
CAN2_EGTM_SRC_CANINT[15:12]	Out	CAN2 Service Request to eGTM
CAN3_EGTM_SRC_CANINT[15:12]	Out	CAN3 Service Request to eGTM
CAN4_EGTM_SRC_CANINT[15:12]	Out	CAN4 Service Request to eGTM
CAN0_DRE_TRIGTYPE[1:0]	Out	Trigger event corresponding to a host buffer
CAN1_DRE_TRIGTYPE[1:0]	Out	Trigger event corresponding to a host buffer
CAN2_DRE_TRIGTYPE[1:0]	Out	Trigger event corresponding to a host buffer
CAN3_DRE_TRIGTYPE[1:0]	Out	Trigger event corresponding to a host buffer
CAN4_DRE_TRIGTYPE[1:0]	Out	Trigger event corresponding to a host buffer
CAN0_DRE_TRIGNODE[1:0]	Out	CAN0 node to which host buffer belongs

(table continues...)

Table 1034 (continued) List of CAN interface signals

Interface signals	I/O	Description
CAN1_DRE_TRIGNODE[1:0]	Out	CAN1 node to which host buffer belongs
CAN2_DRE_TRIGNODE[1:0]	Out	CAN2 node to which host buffer belongs
CAN3_DRE_TRIGNODE[1:0]	Out	CAN3 node to which host buffer belongs
CAN4_DRE_TRIGNODE[1:0]	Out	CAN4 node to which host buffer belongs

21.11.6 TC4Dx CAN revision history

Reference	Description of change(s)
Date range: 2024-08-17 to 2024-11-27	
Module Control Register (CANn=1-4)	• Removing misplaced paragraphs from MCR register description

22 Controller Area Network XL interface (CANXL)

The CANXL module provides a communication interface according to the ISO 11898-1:2024 standard, supporting Classical CAN, CAN FD and CAN XL communication. It supports CAN XL data baudrates of up to 20 Mbits/s.

22.1 Feature list

- Classical CAN, CAN FD and CAN XL communication according to ISO 11898-1:2024
- CAN XL payload size of up to 2048 data bytes
- CAN XL communication baud rate up to 20 Mbits/s
- Hardware-based 64-bit time stamping
- Integrated DMA for message transfers without CPU load
- 1 transmit priority queue with maximum of 32 slots for ID-based priority transmission of CAN frames
- Maximum of 8 TX FIFO queues
- Maximum of 8 RX FIFO queues
- TX message filtering with up to 16 filter elements
- RX message filtering with up to 255 filter elements (with 32-bit word compare)

22.2 Functional overview

The figure below shows the block diagram overview of the CANXL module.

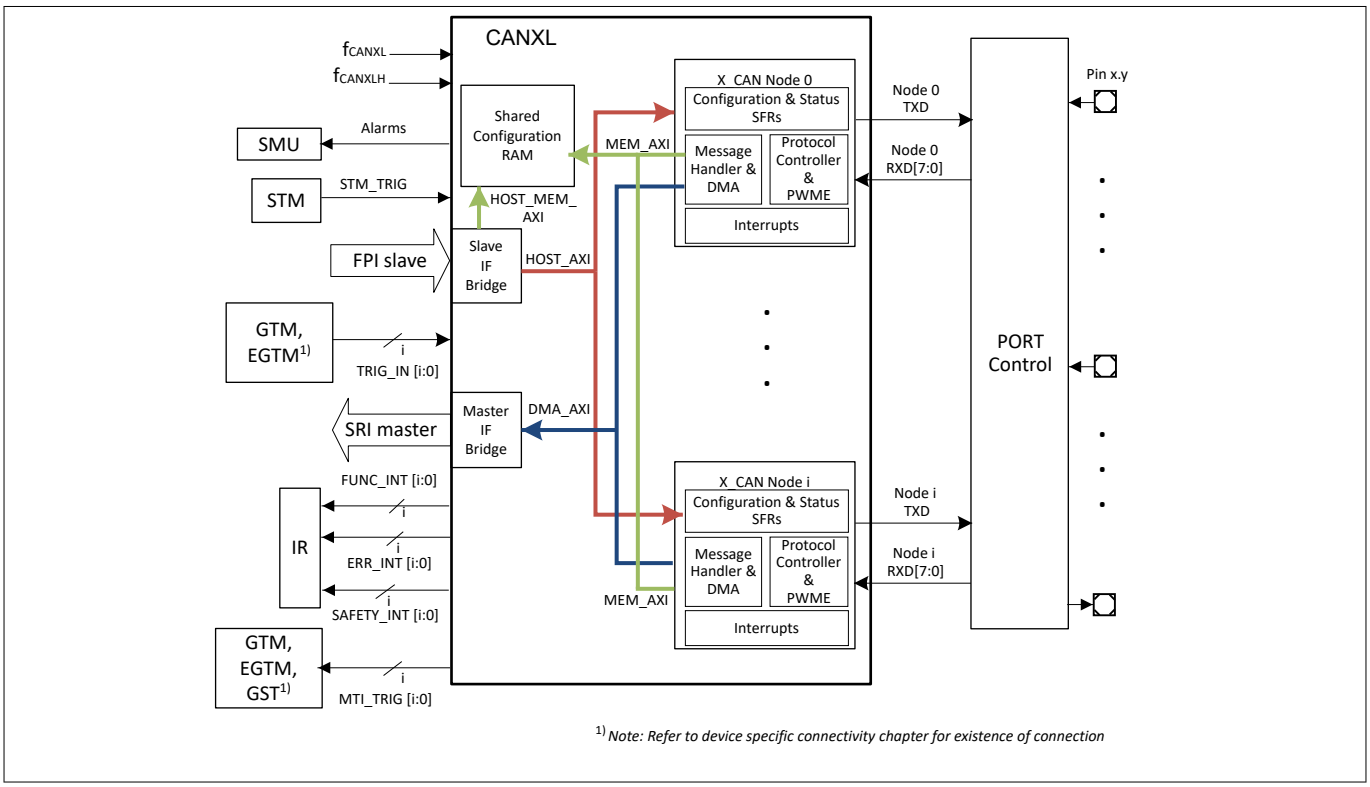


Figure 390 CANXL block diagram

For the alarm description please refer to the alarm mapping tables in the SMU functional block and the Safety Manual.

The Node I are instances of X_CAN IP from Bosch. The Slave Interface Bridge contains the following functions

- Clock control
- Reset functions

- Access protection and
- Slave bus protocol conversion bridges for accessing SFRs and local memory RAM through the 32-bit FPI interface

The Master Interface Bridge contains the following function

- VM and PRS configuration
- Master bus protocol conversion bridges for data and control transfer through 64-bit SRI interface

Related information

TC4Dx SMU alarm mapping tables on page 7233

22.3 Functional description

CANXL Glossary

The following table introduces some acronyms related to the CANXL functional block that are used in the rest of the chapter.

Table 1035 CANXL acronyms

Acronym	Description
CAN	Generic term for any controller area network protocol variant.
CCAN	Classical CAN protocol as per ISO 11898-1:2024.
CAN FD	CAN with flexible datarate protocol as per ISO 11898-1:2024.
CAN XL	CAN XL protocol as per ISO 11898-1:2024.
X_CAN	Bosch IP core that supports CCAN/CAN FD/CAN XL protocols and message handling on a single interface node.
CANXL	Functional block encapsulating multiple X_CAN.
MH	Message Handler module inside X_CAN responsible for DMA of CAN message data and message descriptor data to and from system memory.
PRT	Protocol Controller module inside X_CAN responsible for CAN protocol operation and interface to MH module.
L_MEM	Local Memory. It is also referred in the chapter as "Shared Configuration RAM"
S_MEM	System Memory. It refers to System RAM.
XCAND_MH_DESC	DESCRIPTOR MESSAGE HANDLER
XCAND_MH_DMA	DMA MESSAGE HANDLER

Clock control

The CANXL module has one global clock control special function register (SFR) and each X_CAN node has an independent clock control SFR. The clock control SFR (MODULE_CLC) controls the clock enable or disable of the entire CANXL module. The CLKEN SFR controls the clock enable or disable only for the corresponding X_CAN nodes. The node i clock control is a subset of module clock control, that is switching off the clocks using module clock control switches off all nodes clock domains as well.

***Note**: The clocks must be switched off by hardware only after the respective X_CAN node's Protocol Controller is stopped.*

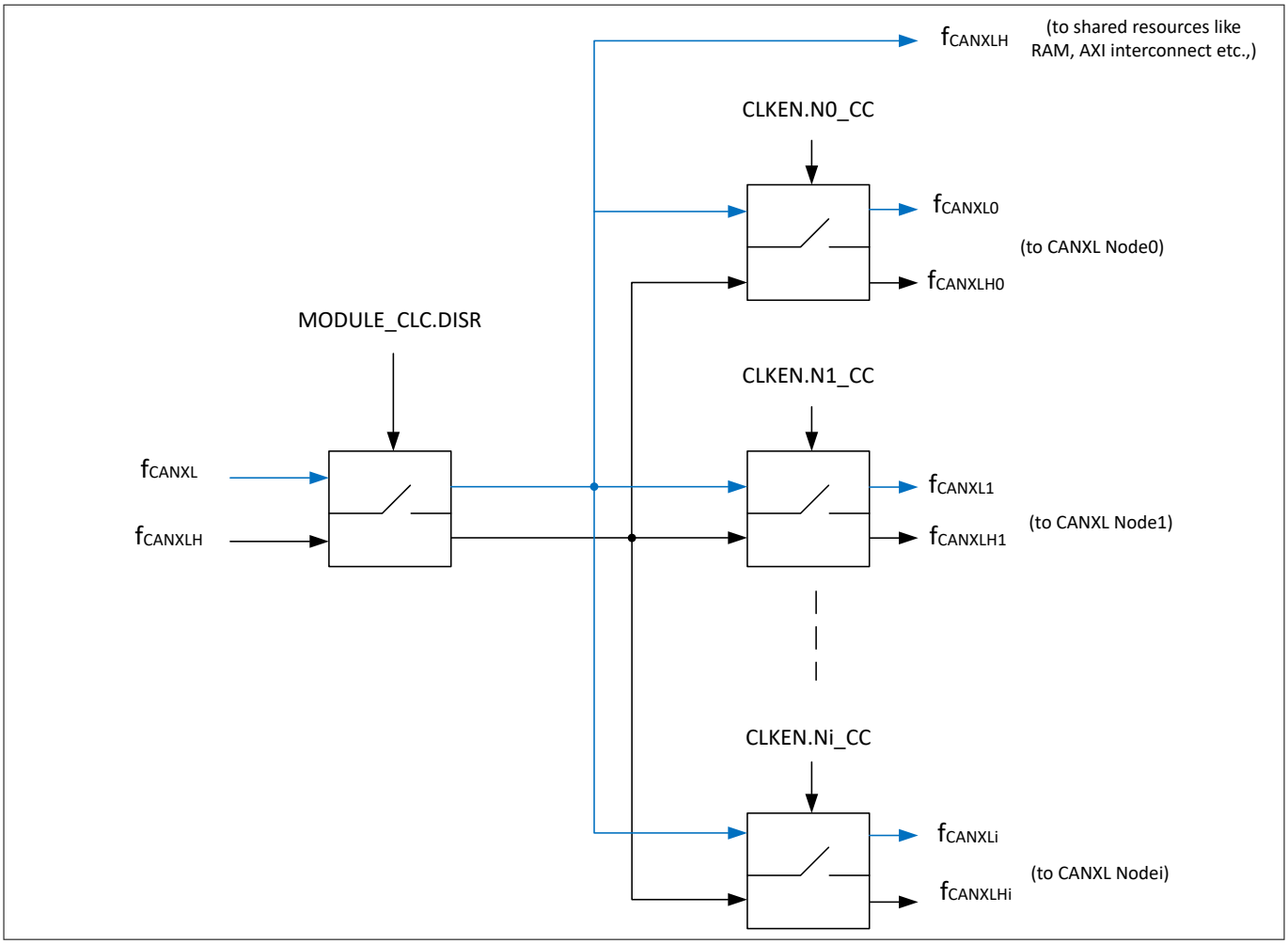


Figure 391 CANXL clock control

Reset control

The CANXL module has one global Reset Control Register set (MODULE_RST_CTRLA and MODULE_RST_CTRLB) and each X_CAN node has an independent Reset Control Register set (NODEi_RST_CTRLA and NODEi_RST_CTRLB). The module reset control SFRs controls the reset of entire CANXL module. The node i reset control SFRs control only for the corresponding X_CAN nodes reset. The node i reset control is a subset of module reset control, that is triggering reset using module reset control resets all nodes also.

***Note**: The kernel reset must be initiated by software only after the corresponding X_CAN node's Message Handler and Protocol Controller are stopped. In case of individual node i kernel reset, it is up to the software to ensure that X_CAN node i's Message Handler and Protocol Controller are stopped, before requesting the reset via node i KRST.*

Access protection

The CANXL module contains global access protection (MODULE_ACCEN SFRs) for access protection configuration to all SFRs which are shared by the X_CAN nodes. Each X_CAN node specific SFRs access protection is configured through the corresponding node specific access protection registers (NODEi_ACCENNODE). The access protection for node specific configuration RAM regions are configured through NODEi_ACCENNODE_RGNLA and NODEi_ACCENNODE_RGNUA registers. Refer to Register Overview table for access protection allocation to SFRs.

Master VM and PRS allocation

All X_CAN nodes within the module share a single SRI master interface and has a single Master TAG-ID assigned. In order to distinguish the SRI transactions initiated by each X_CAN node, NODEi_VMPRSCONFIG SFRs are used. Each X_CAN node DMA transactions can be assigned to a virtual machine number (VM) and protection set (PRS). The VM number and the PRS programmed in the NODEi_VMPRSCONFIG register will be driven on the SRI interconnect during read or write accesses to the corresponding resource (memory), if they are enabled respectively through configuration.

Shared Local Memory (L_MEM)

The shared L_MEM is a 32-bit RAM with allocation of 4kB per X_CAN node. The L_MEM is accessed both by software through HOST_MEM interface and MEM_AXI interface. Arbitration is done in round-robin basis when parallel access is requested through HOST_MEM and by hardware through MEM_AXI. The L_MEM stores the following data for each X_CAN node

- RX filter elements. The base address of the RX filter elements are configured through Ni_RX_FILTER_MEM_ADD register. The amount of RAM space allocated for the RX filter elements is strictly dependent on Ni_RX_FILTER_CTRL.NB_FE bit-field configuration and individual RX filter element configuration.
- Tx FIFO Queue Header and Next descriptors. The base address for TX FIFO Queue descriptors are configured through Ni_TX_DESC_MEM_ADD.FQ_BASE_ADDR bit-field. The amount of RAM space allocated for the Tx FIFO Queue descriptors is strictly dependent on number of active TX FIFO Queues.
- Tx Priority Queue Header descriptors. The base address for TX Priority Queue descriptors are configured through Ni_TX_DESC_MEM_ADD.PQ_BASE_ADDR bit-field. The amount of RAM space allocated for the Tx Priority Queue descriptors is strictly dependent on number of active TX Priority Queue elements.

Interrupt Control

The CANXL module reports status and error events through interrupts to the Interrupt Router module. Each X_CAN node generates following three interrupts

- SRC_CANXLiFUNC interrupt. Each X_CAN node's functional events as described in Ni_FUNC_RAW register can trigger this interrupt, if the corresponding events are enabled through Ni_FUNC_ENA register. The corresponding event flags can be cleared by software through Ni_FUNC_CLR register.
- SRC_CANXLiERR interrupt. Each X_CAN node's error events as described in Ni_ERR_RAW register can trigger this interrupt, if the corresponding events are enabled through Ni_ERR_ENA register. The corresponding event flags can be cleared by software through Ni_ERR_CLR register.
- SRC_CANXLiSAFETY interrupt. Each X_CAN node's error events as described in Ni_SAFETY_RAW register can trigger this interrupt, if the corresponding events are enabled through Ni_SAFETY_ENA register. The corresponding event flags can be cleared by software through Ni_SAFETY_CLR register.

Message transfer trigger to GTM, eGTM and GST

CANXL module provides MTI_TRIG[N_NODES-1:0] trigger output from each CAN XL node to GTM, eGTM and GST. Refer to device specific connectivity chapter for the availability of the trigger to GTM, eGTM and GST.

Each CAN XL node's message transfer relevant events as described in NODEi_MTI_RAW register can trigger this interrupt, if the corresponding events are enabled through NODEi_MTI_ENA register. The corresponding event flags can be cleared by software through NODEi_MTI_CLR register.

Timebase generation for hardware time stamping

Each X_CAN node can capture a 64-bit timestamp for every receive and transmit frames. The source for the timebase can be selected by configuring Ni_TS_CLOCK_CTL.SRC_SEL. One of the four following sources can be selected

1. fCANXLH clock
2. STM trigger
3. GTM trigger and
4. eGTM trigger

The selected source can be be prescaled by a 4-bit prescaler (Ni_TS_CLOCK_CTL.PRESCALER). The prescaled source is used as basis of tick to increment the 64-bit counter which is represented in Ni_TS_CNT_HI and Ni_TS_CNT_LO SFRs. The 64-bit counter is enabled through Ni_TS_CTL.ENABLED. The counter is cleared when Ni_TS_CMD.TS_CLEAR bit-field is written with 1_B.

The X_CAN node instances i with i>0 have an additional feature to synchronizes their time bases with Node 0 64-bit timebase. This can be enabled with Ni_TS_CLOCK_CTL.SYNC_SEL = 1_B. In this scenario, the synchronized nodes timebase is controlled by N0 timebase. The figure below shows the timebase generation and control for each instance of X_CAN node.

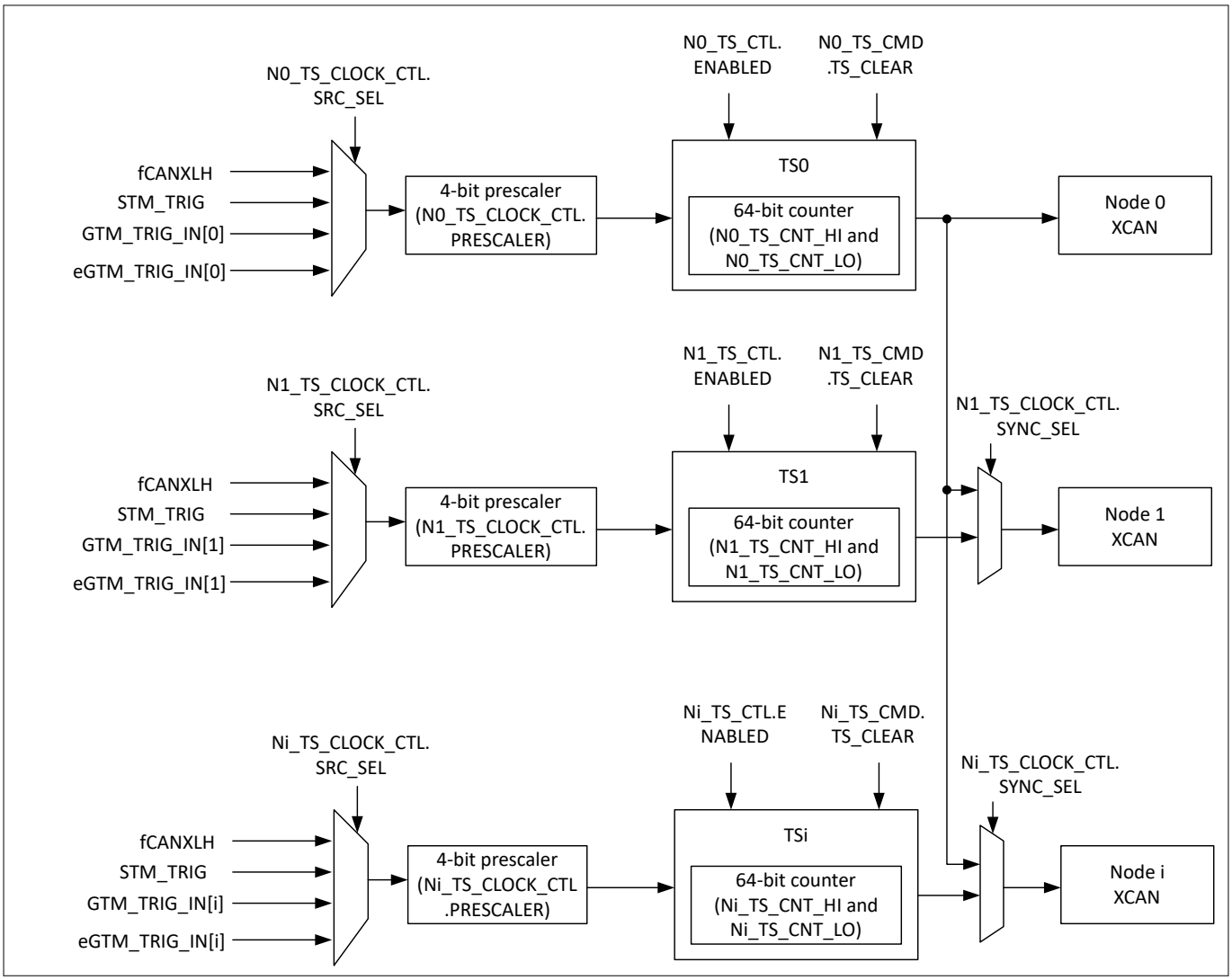


Figure 392 Timebase generation overview

***Note**: Check availability of signals (specially GTM triggers) in connectivity chapter*

Receive Port Selection

Each X_CAN node can select 1 out of 8 possible RX port interfaces. The RX port can be selected using NODEi_PORTCTRL.RXSEL.

22.4 X_CAN

The X_CAN is the new CAN Communication Controller IP supporting CAN XL protocol. The X_CAN performs communication according to ISO 11898-1:2024.

22.4.1 Feature list

- Conform with ISO 11898-1:2024
- Classical CAN with up to 8 data bytes
- CAN FD with up to 64 data bytes
- CAN XL with up to 2048 data bytes
- CAN Error Logging
- Fault Injection Module
- Programmable loop-back test mode
- 1 Priority Queue, up to 32 slots, priority based on the arbitration field of the CAN frame
- 8 TX FIFO queues
- 8 RX FIFO queues
- TX message filtering with up to 16 filter elements
- RX message filtering with up to 255 filter elements, while each can compare one 32-bit word (the actual usable number of filter elements depends on CAN clock frequency, CAN bit rate, and Local Memory performance)
- Maskable module interrupts with three categories: Functional, Functional Error and Safety
- Three clock domains (HOST, CAN, TIMEBASE clock domains)

22.4.2 Functional overview

The X_CAN is the new CAN Communication Controller IP from Bosch supporting CAN XL protocol. The X_CAN performs communication according to ISO 11898-1:2024. The following block diagram shows the internal structure of the X_CAN IP.

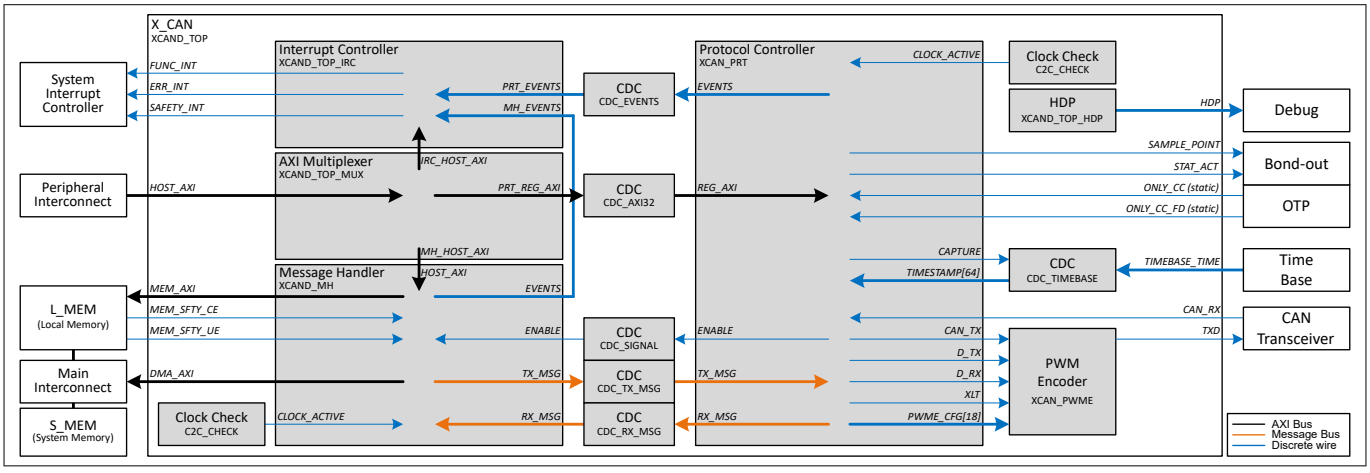


Figure 393 X_CAN block diagram

22.4.3 Functional description

The top level of the X_CAN IP embeds all digital blocks required for communication on one CAN bus.

To start up the X_CAN IP, the Message Handler and the Protocol Controller have to be configured beforehand. The Message Handler must be started first (writing a '1' to the Ni_MH_CTRL.START bit) and afterwards, the Protocol Controller has to be started by writing a ‘1’ to the Ni_CTRL.STRT bit.

22.4.3.1 AXI multiplexer

The X_CAN embeds three register banks, containing configuration, control, status and event information. They are located in the modules Message handler, Protocol controller and Interrupt controller and are accessible through the peripheral interconnect using the HOST_AXI interface and IP internal AXI multiplexer.

- When an access is performed to a non-mapped register in the address range, an access error response is provided
- When a read access to write-only registers or a write access to read-only registers is performed, an access error response is provided

22.4.3.2 Message handler

All functions concerning the storage and scheduling of CAN messages are implemented in the Message handler (MH). The TX path supports the storage of CAN messages in 8 TX FIFO Queues and one TX Priority Queue. The RX path provides 8 RX FIFO Queues.

FIFO data is physically stored in System Memory (S_MEM) and managed by descriptors. TX and RX Filters provide methods to accept or deny CAN Messages and (for RX only) to determine the target RX FIFO for data storage.

The MH will be configured and controlled by HOST CPU through HOST_AXI interface. CAN messages and descriptors are transported between System Memory and local memory autonomously by an internal DMA, which is connected to DMA_AXI. For fast access, the MH needs a Local Memory (L_MEM) which is connected via MEM_AXI interface.

22.4.3.3 Protocol controller

The Protocol Controller (PRT) performs CAN communication as specified in ISO 11898-1:2024. The bitrate can be configured to values up to 20 MBit/s at a clock speed of 160 MHz, depending on the semiconductor technology used. For the connection to the physical layer, additional transceiver hardware is required.

The PRT does not provide internal buffering of frames, so that data has to be transferred by IP internal Message Busses in 32-bit slices in real-time while (de)-serializing them on the CAN Bus. Thus, single data transfers at the internal Message Busses are closely time-synchronised to the schedule at the CAN bus.

22.4.3.4 PWME

The module PWME implements the PWM encoding as specified in ISO 11898-1:2024. When transceiver mode switching is enabled, the PWME encodes the CAN_TX input signal during a CAN XL frame's data phase and during ADH CAN XL bit-field, to generate the PWM encoded output signal TXD.

22.4.3.5 Hardware Debug Port

The X_CAN provides a 16-bit Hardware Debug Port (HDP) to enable debugging using the internal signals of the X_CAN.

The internal signals are organized in pre-defined sets which are selected by Ni_HDP.HDP_SEL. The following tables describe the signal sets.

Table 1036 Hardware Debug Port signal sets

HDP [15:0]	HDP_SEL = 0 (MH debug port)	HDP_SEL = 1 (PRT interface signals)
15	MH_HDP[15]	TX_DU
14	MH_HDP[14]	RX_DO
13	MH_HDP[13]	BUS_OFF

(table continues...)

Table 1036 (continued) Hardware Debug Port signal sets

HDP [15:0]	HDP_SEL = 0 (MH debug port)	HDP_SEL = 1 (PRT interface signals)
12	MH_HDP[12]	E_PASSIVE
11	MH_HDP[11]	E_ACTIVE
10	MH_HDP[10]	BUS_ERR
9	MH_HDP[9]	TX_EVT
8	MH_HDP[8]	RX_EVT
7	MH_HDP[7]	STAT_ACT[1]
6	MH_HDP[6]	STAT_ACT[0]
5	MH_HDP[5]	XLT
4	MH_HDP[4]	D_RX
3	MH_HDP[3]	D_TX
2	MH_HDP[2]	SAMPLE_POINT
1	MH_HDP[1]	CAN_TX
0	MH_HDP[0]	CAN_CLK

22.4.3.6 Interrupt controller

The X_CAN IP is equipped with a central interrupt controller (IRC). It captures all events of the MH and PRT and can be configured for each event individually to interrupt the HOST CPU.

22.5 Message handler (MH)

The Message Handler (MH) is located between the main interconnect and the Protocol Controller (PRT). It is used in transmit direction, to read TX CAN message data from System memory (S_MEM) and to send them to the PRT and in receive direction, to provides S_MEM with the RX CAN message data from PRT.

22.5.1 Feature list

- Functional and Error interrupts
- Safety interrupts
- Safety measures built-in:
  - Data path parity protection
  - Parity protection on address pointers
  - Linked list descriptor protected by CRC
  - Register bank protected by CRC
  - Interface timeout protection (PRT and AXI master interfaces)
- TX message priority based on ID and IDE and SRR and RTR
- Up to 8 TX FIFO queues can be defined
- Up to 8 RX FIFO queues can be defined
- 1 Priority Queue with a programmable number of slot, limited to 32
- TX message filtering with up to 16 filter definition
- RX message filtering with up to 255 filter definition
- Classic CAN and CAN FD supported
- CAN XL supported

22.5.2 Functional overview

The Message Handler (MH) is located between the main interconnect and the Protocol Controller (PRT). It is designed to read TX CAN message data from System memory (S_MEM) and to send them to the PRT.

In the other direction, it provides S_MEM with the RX CAN message data, when received by the PRT. A status feedback is given to the software for every CAN RX and TX messages directly in the S_MEM, avoiding register accesses.

All functions concerning the storage and scheduling of CAN messages are implemented in the Message Handler (MH). The TX path supports the storage of CAN messages in 8 TX FIFO queues and one TX priority queue. The RX path provides 8 RX FIFO queues. FIFO data is physically stored in system memory and managed by descriptors. TX and RX filters provide methods to accept or deny CAN messages and (for RX only) to determine the target RX FIFO for data storage.

The MH will be configured and controlled by software through HOST_AXI interface. CAN messages and descriptors are transported between system memory and local memory (L_MEM) autonomously by an internal DMA, which is connected to DMA_AXI interface. The MH needs a local memory which is connected through MEM_AXI interface.

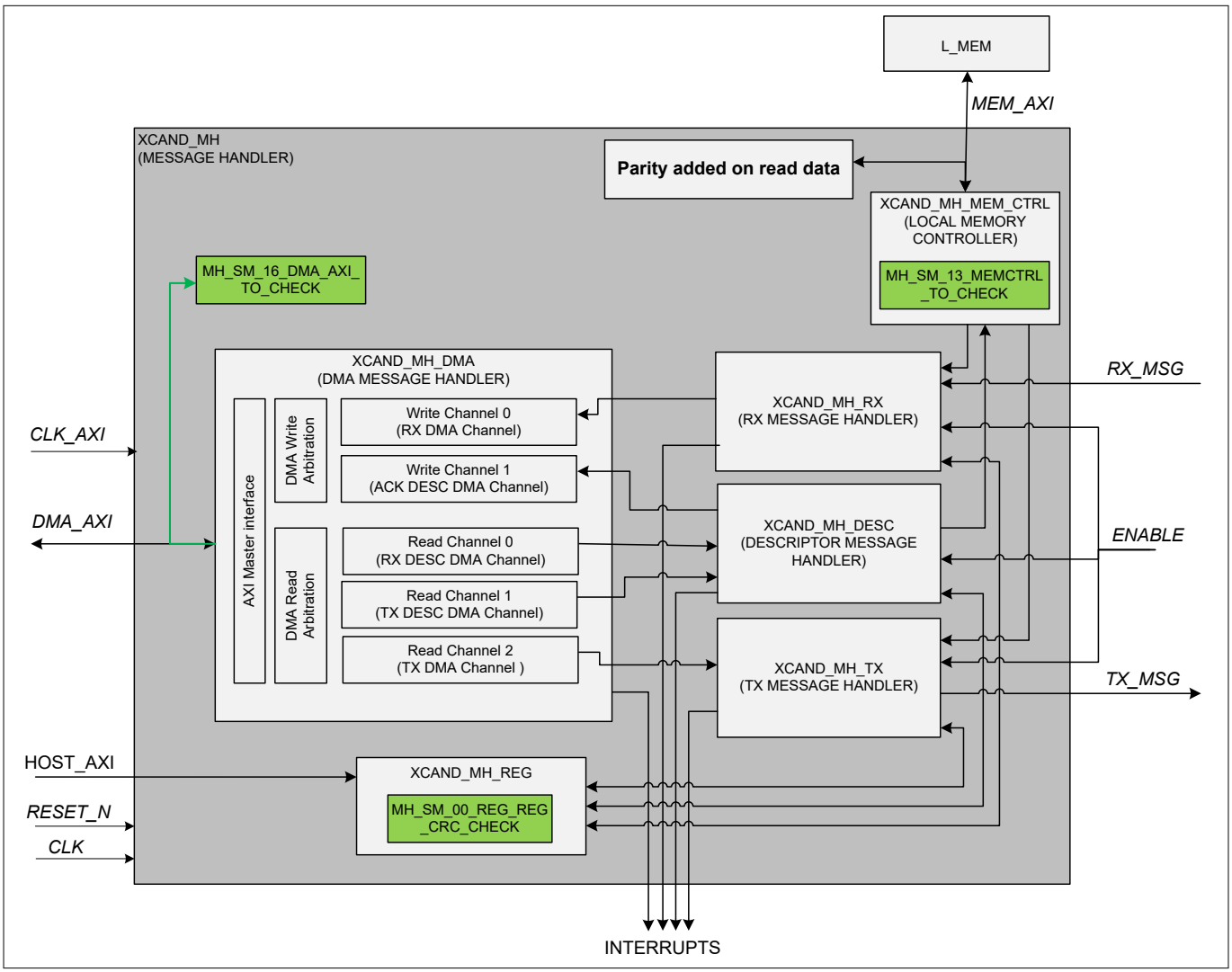


Figure 394 MH block diagram

22.5.3 Functional description

The MH can manage concurrently up to 8 TX FIFO Queues, up to 8 RX FIFO Queues and up to 32 slots defined in a TX Priority Queue. The message handler uses the principle of linked list to define RX and TX FIFO queues, as well as the TX priority queue.

The TX message is made of TX descriptors (see TX descriptor chapter for more information) which define the TX message header information and the address of its payload. The payload buffer can be defined in any location in memory.

An RX message is written to memory based on the information defined in RX descriptors (see RX descriptor chapter for more details) and/or configuration registers. All RX message data can be defined in any location in memory.

The RX FIFO queue can support classic CAN, CAN FD and CAN XL frame format.

The TX FIFO queues and TX priority queue slots can support classic CAN, CAN FD and CAN XL frame format.

The TX MESSAGE HANDLER is managing the TX messages while the RX MESSAGE HANDLER is taking care of RX messages. Both share the Descriptor Message Handler to get their TX and RX descriptors respectively. This module also updates the status at the TX/RX FIFO Queues and TX Priority Queue when a transfer is completed. A dedicated sub-module in the Descriptor Message Handler is assigned to the TX path and one for the RX path, that can run concurrently.

The selection of the highest priority TX message and the RX message filtering is done locally using the L_MEM. Therefore, the highest priority message to be sent is defined in a shorter time. Regarding the RX filtering, the RX filter elements are fetched from the L_MEM to reduce the processing time to accept or reject an RX message before a new one comes in.

The MH can drive only one protocol controller using the TX_MSG and RX_MSG interfaces.

Related information

TX descriptor on page 4187

RX descriptor on page 4194

22.5.3.1 TX MESSAGE HANDLER

The TX MESSAGE HANDLER is in charge of TX FIFO queues and TX priority queue management. Therefore, the TX message handler is requesting the TX descriptor whenever required, arbitrates the TX descriptors according to their IDs and selects the high priority TX message to be sent to the PRT.

Once a TX descriptor is selected and the PRT is winning the arbitration on CAN bus, it fetches the payload data assigned to that descriptor from the system memory.

The internal arbitration on TX descriptors is called TX-SCAN in order to avoid a conflict with the arbitration done on the CAN bus.

A TX filter is put in place to ensure only the relevant TX messages will be sent through the CAN bus.

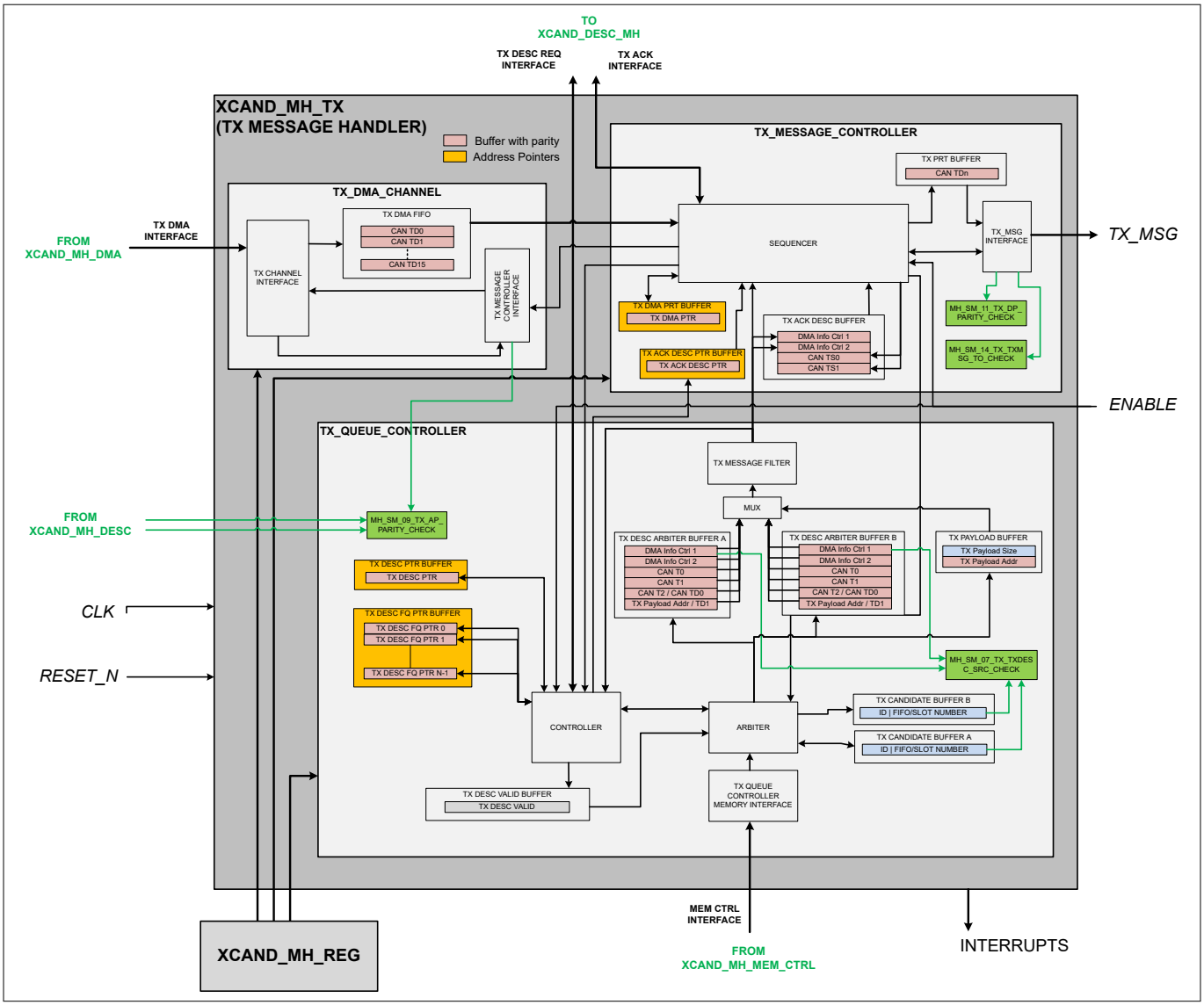


Figure 395 TX MESSAGE HANDLER block diagram

22.5.3.1.1 Block description

Several blocks are used to manage the TX message, TX FIFO Queues and the TX Priority Queue.

TX DMA channel interface

This block is interfacing the DMA MESSAGE HANDLER to send read commands to the system memory. It will also hold the payload data in a local TX DMA FIFO before sending the data to the TX MESSAGE CONTROLLER when available and complete.

The size of the payload data can be different from time to time, as low as 8 bytes for the Classic CAN up to 2048 bytes for the CAN XL.

Every TX descriptor defines the size of the data transfer to be executed. So, only one DMA transfer request is performed per TX descriptor. Every information related to the data transfer is set by the TX MESSAGE CONTROLLER. The TX DMA FIFO size is set to two maximum burst length to allow continuous execution of the data transfer. As soon as the TX DMA FIFO is having enough space to load a new burst the DMA MESSAGE HANDLER will initiate a new fetch from the system memory. Only one data transfer can be defined at a time. As a matter of fact, when the first defined data transfer is finished, meaning the data to be read are inside the TX

DMA FIFO, a second data transfer can be started. The remaining data from the first transfer are used to wait for the second DMA transfer data to be stored in the TX DMA FIFO.

Only the address pointer ADDR_PTR[31:0] to fetch the payload data and the SIZE[10:0] of the transfer are required. This fields can be found on the related TX descriptor.

The other data transfer parameter are static and defined using control registers.

The TX DMA CHANNEL can accept only one data transfer definition at a time in the TX DMA PTR BUFFER, so one data transfer can be performed at a time.

TX message controller

This block is in charge of sequencing the TX message data to the TX_MSG interface. Two sources of data are used to build the TX message. The first data comes from the TX descriptor, which contains the header and the first payload words of the CAN frame.

This TX descriptor comes from the TX QUEUE CONTROLLER and is provided by the ARBITER.

Once the TX descriptor is executed, the address pointer defined in the descriptor is used to fetch further payload data from the S_MEM using the TX DMA CHANNEL.

The TX MESSAGE CONTROLLER is in charge of managing new TX descriptor when several descriptors are used for one TX message. Any new TX message to be sent is provided only by the ARBITER, in all scenarios.

As all TX messages are managed by the TX MESSAGE CONTROLLER, once a message is sent (or not sent) successfully to the PRT, an acknowledge descriptor is provided to the DESC MESSAGE HANDLER to be written back to the source descriptor. If some issues are detected, the current message is canceled and all the traffic from the system memory is aborted. Once done, a new TX message can be considered and should be already provided by the ARBITER.

The PRT signalizes via ENABLE signal whether it requires message handling or not. When this signal goes low, the MH must stop its current activities. This means the TX FIFO Queues and TX Priority Queue are put on hold as well as all the relevant traffic from and to the S_MEM must be aborted.

TX queue controller

This block manages the TX FIFO Queues and the TX Priority Queue as well as the TX-SCAN. As soon as a TX FIFO Queue is started, and/or a TX Priority Queue slot is valid, the TX QUEUE CONTROLLER is fetching the appropriate TX descriptor from the system memory.

Those descriptors are stored in the L_MEM for further processing. The TX descriptors (only part of it) are fetched from the L_MEM and analyzed to find out the TX message having the highest priority.

The one selected is stored locally for the TX MESSAGE CONTROLLER to be read. This block is also computing the address to read the next TX descriptor for every TX FIFO Queues running once used and to manage the active slot from the TX Priority Queue for new one being declared.

All the relevant information to the TX MESSAGE CONTROLLER is provided by this block.

The TX filter uses configuration registers to select between TX messages to be sent to the CAN bus and TX messages to be discarded.

When the TX-SCAN (selection of the TX message with the highest priority) is done, the selected TX descriptor is read from the L_MEM. To ensure that it is the one already selected, some TX descriptor bit fields are checked against the expected value stored locally by the TX-Scan. In case one of the bit fields, listed below, does not match, a TX_DESC_REQ_ERR signal is triggered to the system and Ni_SFTY_INT_STS.TX_DESC_REQ_ERR is set to 1:

- The IN (instance number)
- The FQN (TX FIFO queue number) if PQ = 0
- The PQSN (TX Priority Queue slot number) if PQ = 1

- The PQ (Priority Queue flag)
- The Priority Value assigned to the TX message

22.5.3.2 RX MESSAGE HANDLER

The RX MESSAGE HANDLER is in charge of the RX FIFO queues. Every RX FIFO queue uses a linked list of RX descriptors to identify the exact location in S_MEM to store the message.

The RX MESSAGE HANDLER requests the RX descriptor as required, when for example the RX filter result of an accepted incoming RX Message becomes available.

The RX filter identifies any incoming RX messages and determines whether it has to be rejected (not stored) or accepted (stored into one of the RX FIFO queues, defined by the RX filter).

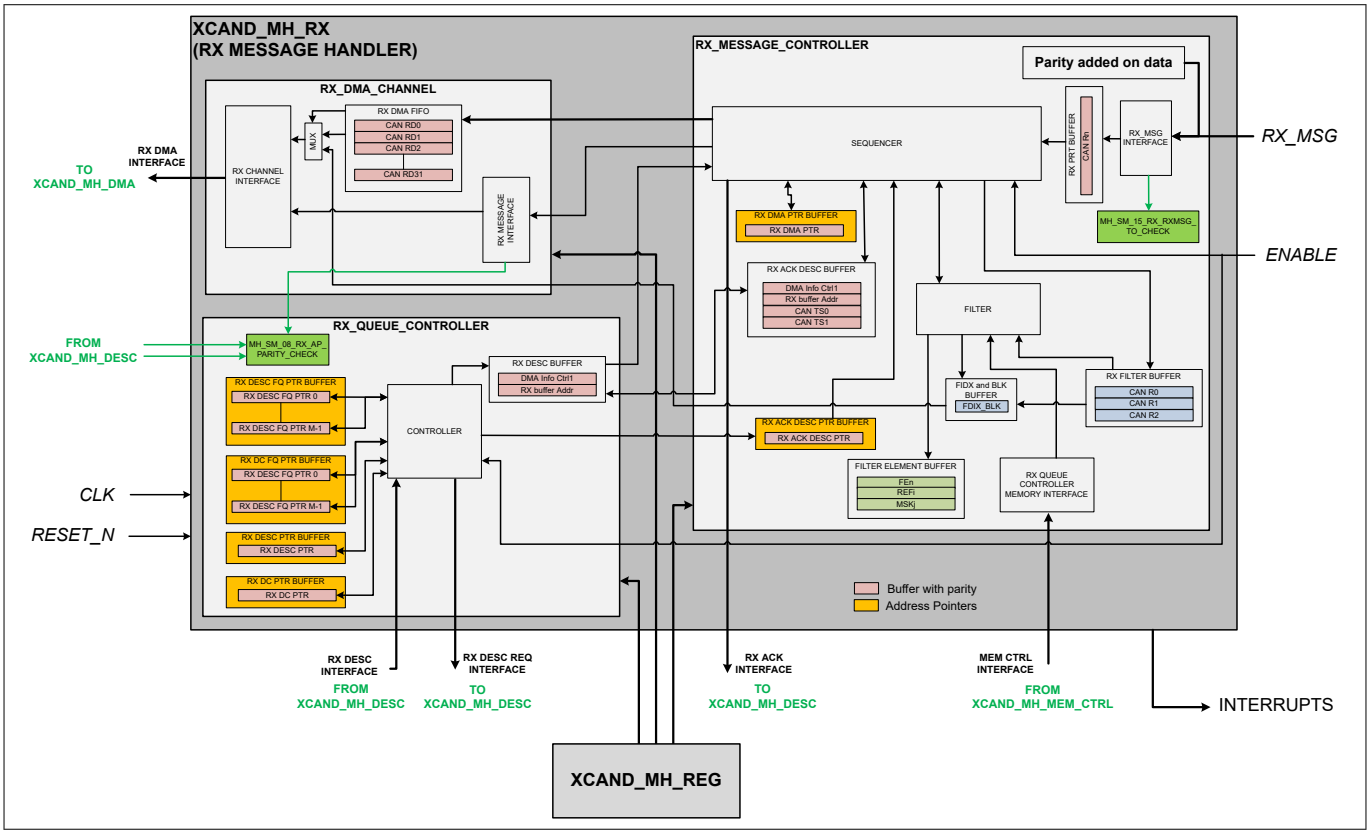


Figure 396 RX MESSAGE HANDLER block diagram

22.5.3.2.1 Block description

Several blocks are used to manage the RX FIFO Queues.

RX DMA channel

This block interfaces the DMA MESSAGE HANDLER to send write commands to the S_MEM. It will also buffer the RX message data in a local RX DMA FIFO before sending the data to the S_MEM.

The size of the payload data for a CAN frame can be the size of 8 bytes for Classical CAN and up to 2048 bytes for CAN XL.

Every RX descriptor of the same RX FIFO Queue has a fixed buffer size to hold data. The size of the overall transfers is stored locally to identify how many descriptors are required for the RX message and what the size of each DMA data transfer is.

As a matter of fact, when the RX message exceeds the maximum buffer pointed by the current descriptor, one or several DMA data transfers are executed. In other words, there are as many DMA data transfer as RX descriptors per RX message.

Only the address pointer ADDR_PTR[31:0] and the SIZE[10:0] (size of the transfer) are required to fetch the payload data. This fields can be found on the RX descriptor. The other data transfer parameters are static and are defined in control registers.

The RX DMA CHANNEL can accept only one data transfer definition in the RX DMA PTR BUFFER, so one data transfer can be performed at a time.

RX message controller

This block is in charge of sequencing the RX message data from the RX_MSG interface to the RX_DMA_CHANNEL and to filter the incoming messages.

The RX message is managed by the RX MESSAGE CONTROLLER. Once a message is received successfully, an acknowledge descriptor is provided to the DESCRIPTOR MESSAGE HANDLER to be written back to the first descriptor of the RX message. This first descriptor is used along the process of receiving a message and is the only one which is acknowledged and holds the header data.

If an error was detected, the current message will be canceled and the storage to the S_MEM will be aborted. Once done, a new RX message can be processed and the RX descriptors of the previously aborted message are reused.

To avoid duplication of buffers, the data from the PRT is stored directly into the RX DMA FIFO without waiting for the result of the filter. Once the result of the filter is known (RX message accepted or not accepted), the CAN data being received is stored in the S_MEM or discarded.

The PRT signals through ENABLE signal whether it is active and requires message handling or not. When this signal is going low, the MH stops current activities. This means that the RX FIFO queues are put on hold as well as the traffic from and to the S_MEM will be aborted.

Related information

RX filter on page 4243

RX queue controller

This block manages the RX FIFO queues and keeps track of the write pointers to use for each of them. As soon as an RX FIFO queue is started, the RX queue controller is allowed to request descriptors from the descriptor message handler.

The descriptor to be used is stored into the local RX desc buffer and is the result of a request to the descriptor message handler when the RX FIFO queue is identified by the FILTER. This block also computes the address to read the next RX descriptor for every RX FIFO queues running, once used. All the relevant information to write data to the S_MEM or to generate an interrupt when receiving a message is provided to the RX message controller.

In case that several descriptors are required for one message, the RX queue controller can request the next descriptor from the descriptor message handler as soon as RX message controller has taken over the current descriptor.

22.5.3.3 Descriptor message handler

This block is in charge of getting, from the S_MEM, the RX descriptors used by the RX MESSAGE HANDLER. For the TX path, it is also fetching all the descriptors for the TX path which will be used later on by the TX message handler.

On top of providing the appropriate descriptor to those sub-modules, as soon as an RX or TX message is received or sent, it will provide the acknowledge data relative to that message to the header descriptor.

This sub-module is only managing the RX or TX descriptors fetched and acknowledged on request from the two control sub-modules; TX MESSAGE HANDLER and RX MESSAGE HANDLER.

As the RX and TX and acknowledge paths are fully concurrent, it is up to the DMA_CONTROLLER managing traffic from or to the S_MEM to decide which request to serve first.

As the CAN bus is unidirectional there should be a low collision rate on the AXI bus interface of the same channel.

The parallel processing of TX/RX descriptors will decouple functions between the two paths. Such approach relaxes the constraints on those two concurrent data flows, considering use cases where both are active at the same time. Furthermore, while receiving a CAN Frame, TX descriptors can be fetched from the S_MEM on request or while executing RX FIFOs. This approach lowers the complexity of use case management.

Regarding the acknowledge of descriptors, the same strategy is used, the acknowledge path does not interfere with the RX and TX data path.

Any configuration register is defined into the main register bank of the message handler.

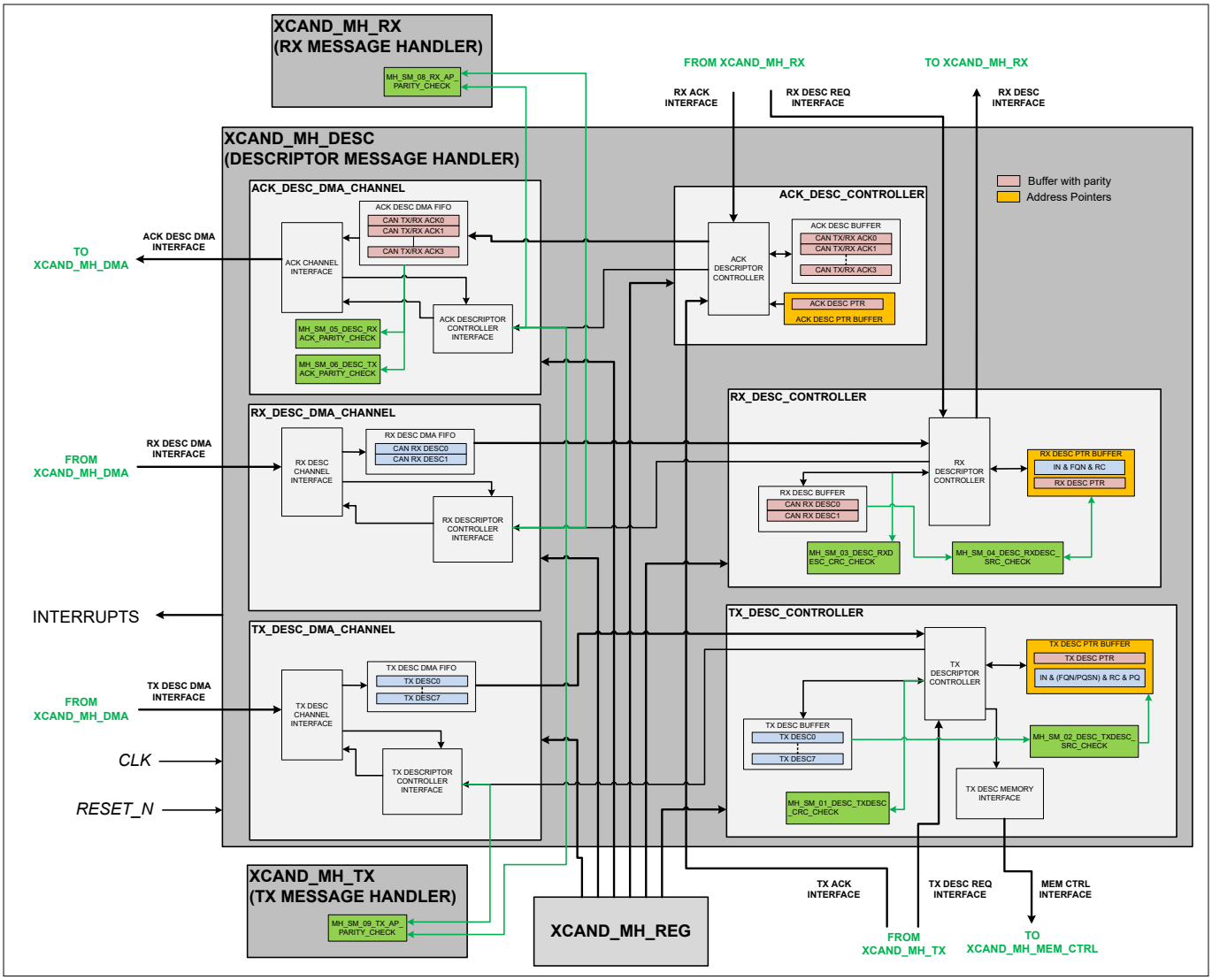


Figure 397 Descriptor message handler block diagram

22.5.3.3.1 Block description

TX_DESC_CONTROLLER

This block stores read descriptor requests from the TX MESSAGE HANDLER and sends them to the TX_DESC_DMA_CHANNEL. It is able to accept up to two requests when there is a need to pre-fetch TX descriptors.

In order to provide the request to the TX_DESC_DMA_CHANNEL, the block sends only the address of the TX descriptor ADDR_PTR[31:0] (the size of the TX descriptor is always identical). Other control signals manage the handshaking. On top of those information, an abort signal is provided to stop the current data transfer on the DMA channel, when requested by the TX MESSAGE HANDLER.

Once a TX descriptor is provided by the TX_DESC_DMA_CHANNEL, several checks are performed to ensure the correctness of the descriptor and its validity. These checks are (in the order):

1. The VALID bit in the TX descriptor is checked to ensure descriptor is valid. The following check is performed only if the VALID bit equals 1
2. A CRC check is done on the TX descriptor. In case a CRC error is detected, a CRC error is triggered to the system using the TX_DESC_REQ_ERR signal and Ni_SFTY_INT_STS.TX_DESC_CRC_ERR flag is set. The following check is performed only if there is no CRC issue
3. The instance number IN[2:0], either the TX FIFO Queue number FQN[3:0] or the TX FIFO Queue slot number PQSN[4:0], the rolling counter RC[4:0] bit fields of the TX descriptor and the Priority Queue bit PQ are checked against the expected values from the request (see TX descriptor definition chapter for more detail on those bit fields). In case that one of the bit fields does not match, a TX_DESC_REQ_ERR signal is triggered to the system and flag Ni_SFTY_INT_STS.TX_DESC_REQ_ERR set.

Whatever the result of the checks done on the TX descriptor, it is always written to the L_MEM. Doing so, the wrong TX descriptor can be read from the L_MEM, if required for debug purpose.

To store the TX descriptor, a write access is performed to the L_MEM through the memory controller interface. As the size of the TX descriptor to write does not change, the number of words to be written is identical for all descriptors. As soon as a TX descriptor is checked and no issue is identified, it is written to the L_MEM and a notification is sent to the TX MESSAGE HANDLER.

The TX descriptor from the S_MEM is stored locally for filtering. Once stored and accepted, it is written to the L_MEM. In case a Header Descriptor is rejected, the TX_FILTER_IRQ interrupt is triggered to the system. The TX MESSAGE CONTROLLER is notified that the requested TX descriptor is rejected and will not be provided. Refer to the TX Filter chapter for a detailed description.

Related information

TX descriptor on page 4187

TX_DESC_DMA_CHANNEL

This block is interfacing the DMA_CONTROLLER to send read commands to the S_MEM. It will also hold the TX descriptors in a local DMA FIFO before sending the data to the TX_DESC_CONTROLLER when available and complete. As the TX descriptor has a fixed size (8 words of 32 bits), the data transfer to be executed by the DMA channel will always be the same. Only the address pointer ADDR_PTR[31:0] to fetch the TX descriptor is required. The other data transfer parameter are static and defined using control registers. As the transmit FIFO can accept only one TX descriptor, only one data transfer can be performed at a time. There is no check performed by this block as everything is done by the TX_DESC_CONTROLLER holding the read request definition. For more details on the DMA_CONTROLLER interface see the relevant chapter.

Related information

DMA message handler on page 4182

RX_DESC_CONTROLLER

This block is in charge of storing read descriptor requests from the RX MESSAGE HANDLER and to send them to the RX_DESC_DMA_CHANNEL. It is possible to accept up to two requests when there is a need to pre-fetch RX descriptors for large payload data defined in RX messages.

In order to provide the request to the RX_DESC_DMA_CHANNEL the block is sending only the address of the RX descriptor ADDR_PTR[31:0] and a PRIORITY signal, other control signals are managing the handshaking. On top of those information an abort signal is provided to stop the current data transfer on the DMA channel when requested by the RX MESSAGE HANDLER.

Once an RX descriptor is providing by the RX_DESC_DMA_CHANNEL several checks are performed to ensure the correctness of the descriptor. These checks are:

1. The VALID bit in the RX descriptor is checked to ensure descriptor is valid. The following check is performed only if the VALID bit equals 1
2. A CRC check is done on the RX descriptor. When the CRC is valid, the RX descriptor is sent to the RX MESSAGE HANDLER, otherwise a CRC error is triggered to the system using the RX_DESC_REQ_ERR signal and flag Ni_SFTY_INT_STS.RX_DESC_REQ_ERR set. The following check is performed only if there is no CRC issue
3. The instance number IN[2:0], the RX FIFO Queue number FQN[3:0] and the rolling counter RC[4:0] bit fields of the RX descriptor are checked against the expected value mentioned in the request (see RX descriptor definition chapter for more detail on those bit fields). In case one of the bit field does not match, an RX_DESC_REQ_ERR signal is triggered to the system and flag Ni_SFTY_INT_STS.RX_DESC_REQ_ERR set.

Related information

RX descriptor on page 4194

RX_DESC_DMA_CHANNEL

This block interfaces the DMA_CONTROLLER to send read commands to the S_MEM. It will also hold the RX descriptors in a local DMA FIFO before sending the data to the RX_DESC_CONTROLLER when available and complete. As the RX descriptors have the same size (2 words of 32 bits), the data transfer to be executed by the DMA channel will always be the same. Only the address pointer ADDR_PTR[31:0], to fetch the RX descriptor, is required. The other data transfer parameters are static and defined using control registers. As the RX FIFO can accept only one RX descriptor, only one data transfer can be performed at a time. There is no check performed by this block as everything is done by the RX_DESC_CONTROLLER holding the read request definition. For more details on the DMA_CONTROLLER interface, see the relevant chapter.

Related information

DMA message handler on page 4182

ACK_DESC_CONTROLLER

This block manages the RX MESSAGE HANDLER and TX MESSAGE HANDLER request when an RX or TX descriptor being executed needs to be acknowledged. As soon as the RX MESSAGE HANDLER has completed its execution using one RX descriptor, the relevant information (transfer status and errors mainly) of that transfer must be sent back to the first descriptor. To do so, the RX MESSAGE HANDLER and TX MESSAGE HANDLER will send a request to the ACK_DESC_CONTROLLER to write acknowledge data into the respective Header Descriptor.

The ACK_DESC_CONTROLLER can only accept data when the ACK DESC DMA FIFO has enough data to store it. As long as this DMA FIFO cannot accept the data, it will hold any request from either RX MESSAGE HANDLER and/or TX MESSAGE HANDLER. Acknowledge data are build and stored in the RX MESSAGE HANDLER and TX MESSAGE HANDLER. This way, any updates along the reception or transmission of a TX message will

automatically be done locally on the sub-module. As the CAN bus is unidirectional, there should be no conflict regarding RX and TX descriptors being acknowledged at the same time. The only exception is when PRT is set in loopback mode. As soon as the ACK DESC DMA FIFO in ACK_DESC_DMA_CHANNEL provides the right FIFO level to receive one burst of data, the ACK_DESC_CONTROLLER will write those data and will push the address pointer of those data. Despite that RX and TX acknowledge requests may not occur at the same time, the higher priority is always given to the RX path. The ACK_DESC_CONTROLLER will always start writing the acknowledge data (always $4 \times 32\ bit$) to the DMA FIFO in ACK_DESC_DMA_CHANNEL whatever the request source is, either TX MESSAGE HANDLER or RX MESSAGE HANDLER. At last it will write the address pointer of that descriptor triggering at the same time a new DMA data transfer. The option, to provide a priority signal to define the urgency of the writing, exists.

ACK_DESC_DMA_CHANNEL

This block is interfacing the DMA_CONTROLLER to send write commands to the S_MEM. It also holds bursts to be sent over the interconnect into a local DMA FIFO before asking the DMA_CONTROLLER to send it to the S_MEM.

As the acknowledge data for RX and TX descriptors is having a fix size (4 words of 32 bits), the data transfer to be executed by the DMA channel will always be the same. Only the address pointer to write the burst is required and a PRIORITY signal to manage the urgency of the request. The other data transfer parameter are static and defined using control registers. As the transmit DMA FIFO can accept only one burst, one transfer can be performed at a time. For more details on the DMA_CONTROLLER interface see the relevant chapter.

22.5.3.4 DMA message handler

The DMA_CONTROLLER reads and writes bursts of data from and to the S_MEM through its AXI4 master interface DMA_AXI (compliant to AMBA 4 ARM™ protocol). The DMA_CONTROLLER manages request commands from the sub-module that is in charge of sending or receiving TX or RX messages, as well as fetching RX or TX descriptors.

It is in charge of providing data to the sub-module responsible to send CAN frames (TX MESSAGE HANDLER) as well as to the sub-module managing the RX and TX descriptors (DESC MESSAGE HANDLER). It is managing all data from a received CAN frames (RX MESSAGE HANDLER) as well as to write back information into RX/TX descriptors when required (DESC MESSAGE HANDLER).

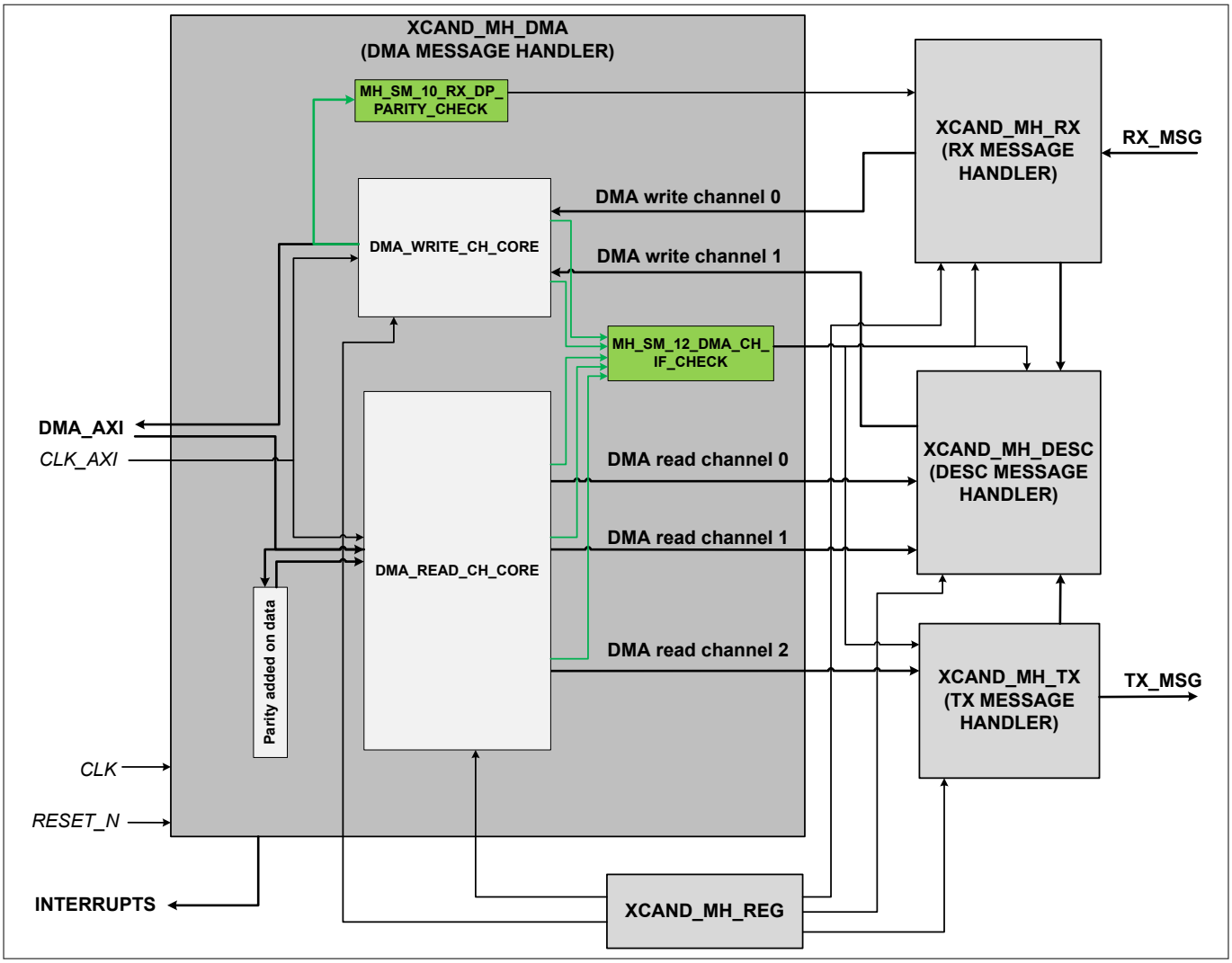


Figure 398 DMA message handler block diagram

22.5.3.4.1 Block description

The DMA build in the DMA MESSAGE HANDLER block has a static configuration, once the software has written the registers. They must not be changed except if all the DMA channels are stopped.

An arbitration process will take place to define which request command is to be served first.

As several concurrent read and write accesses can be foreseen, refer to Ni_AXI_PARAMS.AR_MAX_PEND[1:0] and Ni_AXI_PARAMS.AR_MAX_PEND[1:0] bit field registers.

To maximize the AXI throughput, whatever the number of data transfer to be done, the DMA_CONTROLLER ensures the usage of the maximum burst length whenever possible. To do so, the DMA_CONTROLLER is always trying to generate a burst length for the first transfer to get an aligned address burst size for the next data to be transferred (maximize the usage of maximum burst size for transfers).

The RESP_ERR[1:0] interrupts are used to trigger the system for any bus error, when reading or writing the S_MEM and L_MEM. Some status flags provide the interrupt source, see the Ni_SFTY_INT_STS register.

Before starting any transfer a DMA read/write channel must be enabled. The Ni_TX_FQ_CTRL2.ENABLE[n] and Ni_RX_FQ_CTRL2.ENABLE[n] and Ni_TX_PQ_CTRL2.ENABLE[n] bit-field registers are used to identify when the DMA channels are required. If none of those enable bit are set to 1, no data transfer can occur.

The DMA is intended to:

- Write all the received RX CAN frame data coming from the PRT to the system memory at a defined address location (specified into RX descriptors). This traffic does concern only the RX MESSAGE HANDLER
- Write the acknowledge of TX CAN frame message back to the relative TX descriptor. This traffic is owned by the DESC MESSAGE HANDLER
- Write the acknowledge of RX CAN frame message back to the appropriate RX descriptor. This traffic is owned by the DESC MESSAGE HANDLER
- Read TX descriptors where the TX CAN frame header is defined with some other relevant information like the address pointer of the payload data. This traffic is owned by the DESC MESSAGE HANDLER
- Read RX descriptors according to the RX CAN frame messages being filtered to identify which location to write the received data. This traffic is owned by the DESC MESSAGE HANDLER
- Read TX message payload data from the system memory when the corresponding message header is winning the CAN bus arbitration. This traffic does concern only the TX MESSAGE HANDLER

DMA_WRITE_CH_CORE

This block is in charge of the following:

- Writing data to the system memory and to have those transfers compliant to the AXI4 AMBA protocol
- Providing the appropriate write burst length for a maximum system bus efficiency according to the number of data to be sent
- Reading the relevant number of data from a defined DMA write channel through the read FIFO interface
- Arbitrating among the different DMA write command of those channels
- Stopping any AXI data transfer any time without locking the AXI write system bus interface

The DMA_WRITE_CH_CORE is storing and sending all the write commands to the system memory. As soon as a write command is granted, the required data is fetch from the read FIFO interface of the corresponding channel and written to the AXI write system bus interface.

A classic read FIFO interface is provided at the block interface to avoid embedded data FIFOs. This kind of implementation does allow to scale the data FIFO assigned to any DMA write channel without having to modify the DMA_CONTROLLER. Only the level of the FIFO to be read must be provide to ensure a proper handshaking. The read FIFO interface is defined as, a 32-bit data bus with a read enable and a FIFO level to ensure enough data are present into the FIFO to perform a new burst.

Once a command is received from a DMA write channel, the arbitration process is taking care of the right command to execute.

Any write command selected by the ARBITER must only be issue by a sub-module if all the relevant data of the burst are present into the local FIFO of the sub-module.

As long as the DMA FIFO level is not empty, AXI write commands will be issued according to the write outstanding value set in the Ni_AXI_PARAMS.AW_MAX_PEND[1:0] bit register.

It is not allowed to insert wait state in between data read from the FIFO interface.

DMA_READ_CH_CORE

This block is in charge of:

- Reading data from the system memory and to have those transfers compliant to the AXI4 AMBA protocol
- Providing the appropriate read burst length for a maximum system bus efficiency according to the number of data to be fetched
- Writing the relevant number of data to a defined DMA read channel through the write FIFO interface
- Arbitrating among the different DMA read command of those channels
- Stopping any AXI data transfer any time without locking the AXI read system bus interface

The DMA_READ_CH_CORE is storing and sending all the read commands to the system memory. As soon as a read command is granted, the required data is fetched from the AXI write system bus interface and written to the read FIFO interface of the corresponding channel.

A classic write FIFO interface is provided at the block interface to avoid embedded data FIFOs. This kind of implementation does allow to scale the data FIFO assigned to any DMA read channel without having to modify the DMA_CONTROLLER. Only the level of the FIFO to be written must be provide to ensure a proper handshaking. The write FIFO interface is defined as a 32-bit data bus with a write enable and a FIFO level to ensure enough place are present into the FIFO to receive a new burst.

A read command from the DMA read channel would need to define all the relevant information to describe the read data transfer to be executed, see the common list of signal defined previously.

Once a command is received from a DMA read channel, the arbitration process is taking care of the right command to be executed.

As long as the DMA FIFO level is not full, AXI read commands will be issued according to the read outstanding value set in the Ni_AXI_PARAMS.AR_MAX_PEND[1:0] bit register.

It is not allowed to insert wait state in between data written to the FIFO interface.

22.5.3.4.2 Data transfer mode

Several data transfer type can be defined:

No Transfer: When the register Ni_AXI_PARAMS.AW_MAX_PEND[1:0] is set to 0, no AXI write transfer is executed. Doing so, there is the option to have the MH fully active and running without the need of an external memory to receive RX messages. Acknowledges will not be written, so this mode is considered for debug purpose only. When the Ni_AXI_PARAMS.AR_MAX_PEND[1:0] is set to 0, no read access is performed and without TX/RX descriptor read, the MH will be waiting forever.

The Ni_AXI_PARAMS.AW_MAX_PEND[1:0] and Ni_AXI_PARAMS.AR_MAX_PEND[1:0] can set the maximum number of read/write outstanding commands on the DMA_AXI interface.

22.5.3.4.3 Data transfer description

Address bus

The DMA is able to address up to 4GB memory space (DMA_AXI_AWADDR[31:0] and DMA_AXI_ARADDR[31:0]).

Burst size

The maximum number of bytes to transfer in each data transfer is fixed and set to 4. Any read or write transfer is always using 32 bits.

When considering TX CAN frame for instance, the payload data being define as bytes must be 4 byte aligned when read from the system memory.

For the RX CAN frame, if data to be written to system memory is not properly aligned some padding needs to be added to complete a 4-byte word. The padding bytes are set to 00_H.

As a consequence, the write strobe signals are not managed by the DMA_CONTROLLER as all 4 bytes are always written.

Burst length

The DMA_CONTROLLER for the AXI read and write transfers supports INCR burst lengths from 1 to 8, considering an AXI 32 bits data bus width. The DMA_AXI_AWLEN[3:0] and DMA_AXI_ARLEN[3:0] are sized to support a

maximum burst length of 16 despite only 8 is possible. To be fully compliant with the AXI4 AMBA protocol [3] the DMA_AXI_AWLEN[7:4] and DMA_AXI_ARLEN[7:4] are considered as 0000_B.

The DMA_CONTROLLER will always try to align its burst address to make full benefit of the maximum allowed burst length. The address burst value must be always 32 byte aligned to ensure the maximum burst length ($8 \times 32\ bit$). Whatever the data transfer mode, the DMA engine will reduce (if needed) the size of the first burst to align the address to the maximum burst length. Depending on the amount of data to be transferred the last burst can be shorter.

It is important to optimize the access to the system memory, especially if a low number of data transfer is performed. As an example, if a data transfer of $12 \times 32\ bit$ needs to be executed and the start address is 32 byte aligned, it will result in two burst $8 \times 32\ bit$ and $4 \times 32\ bit$. In case the start address is not aligned and we are in the worst scenario it could lead to 3 bursts $3 \times 32\ bit$ and $8 \times 32\ bit$ and $1 \times 32\ bit$ or $2 \times 32\ bit$ and $8 \times 32\ bit$ and $2 \times 32\ bit$ or $1 \times 32\ bit$ and $8 \times 32\ bit$ and $3 \times 32\ bit$.

In case high latency is expected, it is essential to limit the number of burst and make sure whenever possible to align the start address to the maximum burst size.

The DMA_CONTROLLER provides variable burst length of data according to the sub-module command requests.

The burst lengths from/to sub-modules connected to the DMA_CONTROLLER are defined based on the data type of information to be used.

Here below are the expected burst length from/to the sub-modules:

- TX MESSAGE HANDLER: This sub-module does read the TX payload data from the system memory through the DMA read channel 2. The maximum burst length is limited to $8 \times 32\ bit$. There is no write access from this sub-module
- RX MESSAGE HANDLER: This sub-module writes the RX CAN frame data to the system memory through the DMA write channel 1. The maximum burst length is limited to $8 \times 32\ bit$. There is no read access from this sub-module
- DESCRIPTOR MESSAGE HANDLER: This sub-module performs a fixed burst read of $8 \times 32\ bit$ to read TX descriptors from the system memory through the DMA read channel 2. A fixed burst length of $2 \times 32\ bit$ is used instead to read RX descriptors through the DMA read channel 0. To acknowledge any transfer from and to the CAN bus, a fixed burst length of $4 \times 32\ bit$ is performed to either the RX descriptor for received frame or to the TX descriptor for TX message through the DMA write channel 1

Outstanding transactions

In order to support read and write outstanding commands and to limit the FIFO size, the maximum burst length is limited to $8 \times 32\ bit$. The maximum number of outstanding command/transactions expected at the DMA_AXI interface is programmable, see Ni_AXI_PARAMS.AR_MAX_PEND[1:0] and Ni_AXI_PARAMS.AR_MAX_PEND[1:0] bit field register. Up to 3 read and write outstanding transactions can be specified. Even if set to the maximum value, the maximum number of outstanding transactions performed by the MH will depend on many parameters like the system latency, the CAN Bus bit rate, the MH and PRT clock ratio.

Burst type

The only burst type supported is the burst incrementing INCR. Check [3] for more information

The WRAP/FIXED burst type is not supported.

Memory attributes

The memory attributes for the read or write accesses to memory are Normal, Non-modifiable (Non-cacheable in AXI3) and Non-Bufferable. No read-allocate nor Write-allocate are expected on this interface and would be set to 0_B.

As a reminder, Non-bufferable means:

- The write response must be obtained from the final destination
- Read data must be obtained from the destination
- Transactions are Non-Modifiable
- Read and write transactions from the same ID to addresses that overlap must remain ordered

Non-modifiable means:

- A Non-modifiable transaction must not be split into multiple transactions or merged with other transactions
- In a Non-modifiable transaction, the parameters AxADDR, AxSIZE, AxLEN, AxBURST and AxPROT must not be changed

Transaction ID

The DMA_CONTROLLER is building the ID of every burst access based on the number of channels defined. It provides a way to track on the system bus which DMA channel is doing the access at any time.

For the AXI read interface, the DMA_AXI_ARID[1:0] defines the channel number as follow:

00_B => RX descriptor fetch from S_MEM

01_B => TX descriptor fetch from S_MEM

10_B => TX data payload read from S_MEM

For the AXI write interface, the DMA_AXI_AWID[0] defines the channel number as follow:

0_B => TX/RX descriptor acknowledge to S_MEM

1_B => RX message data write (payload and header) to S_MEM

22.5.3.5 TX descriptor

TX descriptors are used for the TX FIFO Queues and the TX Priority Queue. They can be fetched with one AXI burst, as the overall size is only $8 \times 32\ bit$.

Many bit fields are common but some are different between TX FIFO Queue and TX Priority Queue. Details are provided in the following table.

Further information is provided by the chapter TX Message Header Definition.

Table 1037 TX descriptor description

Element number	Bit-field	Name	Managed by	Description/Constraints
0	[31]	VALID	SW/MH	Valid: The SW must set this bit to 1 to define a TX descriptor is valid for the MH. When the descriptor has been fully used, the MH will clear this bit when writing the acknowledge data information back to this descriptor. This update occurs only when the HD bit is set to 1. In case the descriptor is fetched when this bit is set to 0, an interrupt TX_FQ_IRQ is triggered to the system for the TX FIFO queue n having this descriptor
	[30]	HD	SW only	Must be set to 1

(table continues...)

Table 1037 (continued) TX descriptor description

Element number	Bit-field	Name	Managed by	Description/Constraints
	[29]	WRAP	SW only	Wrap: When set to 1 the next message descriptor is the one declared at the initial start address of the TX FIFO Queue (First Descriptor). This bit provides a way to the SW to keep the next TX message continuous in a memory buffer if less space is available at the end of a data container
	[28]	NEXT	SW only	Must be set to 0
	[27]	IRQ	SW only	Interrupt: when set to 1 an interrupt is triggered to the system when the descriptor execution is complete, meaning when the TX message has been sent to the CAN bus
	[26]	PQ	SW only	TX Priority Queue: when set to 1, the TX descriptor belongs to the TX Priority Queue TX FIFO Queue: must be set to 0
	[25]	END	SW only	For the TX FIFO Queue: when set to 1 the TX FIFO Queue defined its ending, it means, it is set as inactive. Once done, the TX FIFO Queue can be reprogrammed and started
				For the TX Priority Queue: must be set to 0
	[24:16]	CRC[8:0]	SW only	CRC: this CRC is computed by the SW for the current TX descriptor. It must consider all elements assuming this bit-field as set to 0. Any CRC error is triggering an interrupt to the system. The CRC is not evaluated if the Ni_MH_SFTY_CTRL.TX_DESC_CRC_EN bit is set to 0.
	[15:12]	FQN[3:0]	SW only	TX FIFO Queue: define the TX FIFO Queue number allocated to this TX descriptor. Despite being set to 4 bits, only the FQN[2:0] bit range is used
		PQSN[4:1]	SW only	TX Priority Queue: define the TX FIFO Queue slot number allocate to this descriptor
	[11]	Reserved	SW only	TX FIFO Queue: must be set to 0
		PQSN[0]	SW only	TX Priority Queue: Define the TX FIFO Queue slot number allocated to this descriptor
	[10:9]	Not used	SW only	Must be set to 0

(table continues...)

Table 1037 (continued) TX descriptor description

Element number	Bit-field	Name	Managed by	Description/Constraints
	[8:4]	RC[4:0]	SW only	Rolling Counter: use to track the order of TX descriptor fetched when a TX FIFO Queue or a TX Priority Queue slot is running. TX FIFO Queue: When a TX FIFO Queue is started for the first time, its first TX descriptor must have the RC[4:0] set to 00000_B. This value must be incremented for every new TX descriptor up to 11111_B and then back to 00000_B even in case of wrap. TX Priority Queue: This bit field must be set to 00000_B as the initial value for all TX Header descriptor defined in the different slots.
	[3:0]	STS[3:0]	MH only	Status: gives the status of the TX message transmitted. The MH writes back only the Header Descriptor (HD bit set to 1) for status report. The SW must always set it to 0. 0000_B: none 0001_B: message sent successfully 0010_B: message not sent after a number of trials 0011_B: message skipped due to HFI 0100_B: message rejected by TX filter 0101_B to 1110_B: reserved 1111_B: message acknowledge data with parity error
1	[31:27]	Not Used	SW only	Must be set to 0
	[26]	PLSRC	SW only	Payload Source: This bit provides to the MH the information about the need to fetch payload data in the data container when executing only a TX Header Descriptor. When set to 1: the TX descriptor is attached to a data container which would need to be accessed and the bit field SIZE[9:0] defines the number of TX data to send for this descriptor. For CAN XL, as no payload data can be defined in TX descriptor, this bit is always set to 1 for CAN XL. For CAN FD, this bit is set to 1 when the payload data is greater than 4 bytes. When set to 0: the payload data defined in the data container are not required. Therefore, the TX descriptor includes all data payload. For the Classical CAN, all payload data are always included, this bit must always be set to 0. In case of CAN FD it would be set to 0 only when the payload data is less or equal to 4 bytes. Nevertheless, the bit field SIZE[9:0] still defines the number of payload data to send per TX descriptor

(table continues...)

Table 1037 (continued) TX descriptor description

Element number	Bit-field	Name	Managed by	Description/Constraints
	[25:16]	SIZE[9:0]	SW only	Define the buffer size in words (32-bit) for the given TX descriptor to transmit to the PRT. As an example, a payload from 1 to 4 bytes requires SIZE to be set to 1. As only 32-bit read accesses are performed the buffer size containing the payload must be 32 bit aligned. When set to 0, there is no payload data attached to the TX descriptor (only valid for Classical CAN/CAN FD without payload or a Classical CAN remote frame) For CAN XL no data is defined in TX descriptor. The MH replies only on the address pointer defined in element 7 to fetch payload data from S_MEM. For CAN FD: SIZE > 1: The copy of the first data payload (aligned on 32 bits) is required in element 6. The address pointer in element 7 is used to fetch the payload data from S_MEM. SIZE = 1: The copy of the first data payload is required in element 6. The address pointer in element 7 is not used. Nevertheless, it is required to have it set to the address of the payload data in S_MEM SIZE = 0: Element 7 and 6 are not used
	[15:13]	IN[2:0]	SW only	Instance Number: define the X_CAN instance number using that descriptor. This bit field is relevant if several X_CAN are running concurrently. It provides a way to detect descriptor fetch issue between instances. The value defined must be equal to the one defined in the Ni_MH_CFG.INST_NUM bit field register.
	[12]	Not Used	SW only	Must be set to 0
	[11:2]	TDO[9:0]	SW only	For the TX Priority Queue: This bit field must be set to 0.
		NHDO[9:0]	SW only	For the TX FIFO Queue: This bit field must be set to 1.
	[2:0]	Not used	SW only	Must be set to 0
2	[31:0]	TS0[31:0]	MH only	Timestamp 0: LSB of the 64-bit timestamp of the successfully sent TX message (only valid when HD bit is set to 1)
3	[31:0]	TS1[31:0]	MH only	Timestamp 1: MSB of the 64-bit timestamp of the successfully sent TX message (only valid when HD bit is set to 1)
4	[31:0]	T0[31:0]	SW only	Define the TX message header information, see TX message header definition chapter

(table continues...)

Table 1037 (continued) TX descriptor description

Element number	Bit-field	Name	Managed by	Description/Constraints
5	[31:0]	T1[31:0]	SW only	Define the TX message header information, see TX message header definition chapter
6	[31:0]	TD0[31:0]	SW only	Classic CAN and CAN FD: define the first payload of the TX message
		T2[31:0]	SW only	CAN XL: Defined the TX message header information, see TX message header definition chapter
7	[31:0]	TD1[31:0]	SW only	Classical CAN with payload greater equal to 4 byte: define the last payload data of the TX message for the Classical CAN (in case payload data is greater than 4 bytes).
		TX_AP[31:0]	SW only	CAN XL and CAN FD with payload greater than 4 bytes: Address pointer to fetch the TX message payload data. This bit-field is mandatory for the CAN XL as no payload data can be defined into the descriptor. As the Classic CAN has only 8 bytes payload, the whole message can be defined using only one TX descriptor, see TD0 and TD1. As the address pointer must be 32 bit aligned the two LSB will not be considered and so must be set to 0 all time. In case the TX_AP is not used it must be set to 0.

Table 1038 TX Descriptor Element managed by SW

	SW to write information to MH	SW to read information from MH
Element number	**Header descriptor**	**Header descriptor**
0	Mandatory	Mandatory
1	Mandatory	Mandatory
2	NA	Mandatory
3	NA	Mandatory
4	Mandatory	NA
5	Mandatory	NA
6	Mandatory	NA
7	Optional	NA

Table 1039 TX Descriptor Element managed by MH

	MH to write information to SW	MH to read information from SW
Element number	**Header descriptor**	**Header descriptor**
0	Mandatory	Mandatory
1	Mandatory	Mandatory

(table continues...)

Table 1039 (continued) TX Descriptor Element managed by MH

	MH to write information to SW	MH to read information from SW
Element number	**Header descriptor**	**Header descriptor**
2	Mandatory	NA
3	Mandatory	NA
4	NA	Mandatory
5	NA	Mandatory
6	NA	Mandatory
7	NA	Optional

22.5.3.5.1 TX descriptor CRC computation

A dedicated CRC is computed for every TX descriptor. When a CRC error is detected the DESC_ERR interrupt signal is triggered. This way the data transfer setting and description up to the DMA engine are fully protected.

The CRC covers all the relevant data, meaning the 247-bit data in the TX descriptor considering the CRC bit-field in the descriptor as equal to 9'b0. The CRC is part of the Element Number 0.

The CRC ($CRC-9_167$) is computed assuming the following elements in sequence:

Element Number 0[31:25] & 9'b0 & Element Number 0[15:0].

Element Number 1[31:0]

Element Number 2[31:0] set to 32’b0

Element Number 3[31:0] set to 32’b0

Element Number 4[31:0]

Element Number 5[31:0]

Element Number 6[31:0]

Element Number 7[31:0]

***Note**: The Koopman representation of the polynomial CRC-9_167is used to protect TX descriptors: $CRC-9_167=\left(x^9+x^7+x^6+x^3+x^2+x+1\right)$ (CRC polynomial in implicit "+1" hex format, meaning the trailing "+1" is omitted from the polynomial number)*

Using the Ni_MH_SFTY_CTRL.TX_DESC_CRC_EN bit register, the SW can decide to disable this check for all the TX descriptors fetched from S_MEM or L_MEM.

22.5.3.5.2 TX descriptors errors

When a TX descriptor error is detected, the relevant information are logged in the Ni_DESC_ERR_INFO1 register. Furthermore, the source address of the faulty TX descriptor is logged in the Ni_DESC_ERR_INFO0 register. This would help the SW to identify potential root causes when such error occurs. The Ni_DESC_ERR_INFO1.RX_TX bit is set to 0 when a TX descriptor gets an error.

22.5.3.6 TX message header definition

The TX descriptor contains the TX message header. The header data structure depends on the CAN Frame Format (Classical CAN, CAN FD, CAN XL) to be used for this message on the CAN Bus. It can be controlled by the

header bits T0.FDF, T0.XLF and T0.XTD. The following tables describe the three data structures used for the headers.

Table 1040 Classical CAN TX header definition

Tn	Bits	Name	Description/Constraints
T0	[31]	FDF	FD Format
	[30]	XLF	XL Format
	[29]	XTD	Extended Identifier
	[28:18]	BaseID [28:18]	Base ID
	[17:0]	ExtID [17:0]	Extended ID
T1	[31]	Reserved	Not Applicable
	[30]	FIR	Fault Injection Request
	[29:27]	Reserved	Not Applicable
	[26]	RTR	Remote Transmission Request
	[25:20]	Reserved	Not Applicable
	[19:16]	DLC[3:0]	Data Length Code
	[15:0]	Reserved	Not Applicable

***Note**: Classical CAN frames (CBDF, CEDF, CBRF, CERF) require T0.FDF = T0.XLF = 0. The header consist of T0 and T1.*

Table 1041 CAN FD TX header definition

Tn	Bits	Name	Description/Constraints
T0	[31]	FDF	FD Format
	[30]	XLF	XL Format
	[29]	XTD	Extended Identifier
	[28:18]	BaseID [28:18]	Base ID
	[17:0]	ExtID [17:0]	Extended ID
T1	[31]	Reserved	Not Applicable
	[30]	FIR	Fault Injection Request
	[29:27]	Reserved	Not Applicable
	[26]	Must be set to 0	Not Applicable
	[25]	BRS	Bit Rate Switch
	[24:21]	Reserved	Not Applicable
	[20]	ESI	Error State Indicator
	[19:16]	DLC[3:0]	Data Length Code
	[15:0]	Reserved	Not Applicable

***Note**: CAN FD frames (FBDF, FEDF) require T0.FDF = 1 and T0.XLF = 0. The header consist of T0 and T1.*

Table 1042 CAN XL TX header definition

Tn	Bits	Name	Description/Constraints
T0	[31]	FDF	FD Format
	[30]	XLF	XL Format
	[29]	XTD	Extended Identifier
	[28:18]	Priority ID [28:18]	Priority Identifier
	[17]	RRS	Remote Request Substitution
	[16]	SEC	Simple Extended Content
	[15:8]	VCID[7:0]	Virtual CAN Network ID
	[7:0]	SDT[7:0]	SDU Type
T1	[31]	Reserved	Not Applicable
	[30]	FIR	Fault Injection Request
	[29:27]	Reserved	Not Applicable
	[26:16]	DLC-XL [10:0]	Data Length Code with CAN XL encoding
	[15:0]	Reserved	Not Applicable
T2	[31:0]	AF[31:0]	Acceptance Field

***Note**: CAN XL frames (XLFF) require T0.FDF = T0.XLF = 1 and T0.XTD = 0. The header consist of T0, T1 and T2.*

22.5.3.7 RX descriptor

The RX descriptor definition for the RX FIFO is defined in table below. Only 4x32 bit are required to define an RX descriptor. Hence the overall RX descriptor can be fetched with one burst. Some bit field elements are defined in a separate table for the sake of simplicity.

Table 1043 RX descriptor description

Element number	Bit-field	Name	Managed by	Description/Constraints
0	[31]	VALID	SW/MH	Valid: The SW must set this bit to 0 to define an RX descriptor is pointing to a valid data container. As soon as the RX descriptor is executed the MH will set this bit to 1 to indicate to the SW valid data written to the S_MEM. In case the RX descriptor is fetched with this bit set to 1 and interrupt RX_FQ_IRQ is triggered to the system for the RX FIFO Queue having this non valid descriptor. The SW must clear this bit only when all the RX message data attached have been read

(table continues...)

Table 1043 (continued) RX descriptor description

Element number	Bit-field	Name	Managed by	Description/Constraints
	[30]	HD	MH only	Message header: when set to 1 the RX descriptor is defined as containing the header of the RX message. Any other RX descriptor, if several descriptors are used for the same RX message, will contain only payload data. In Continuous Mode HD is always set to 1 as only one RX descriptor is used per RX message
	[29]	reserved	SW only	Must be set to 0
	[28]	NEXT	MH only	Next: Set to 1 by the MH to indicate in the RX Header descriptor that more than one descriptor is used for the RX message. This information is only mentioned in the Header Descriptor, the RX Trailing Descriptors are not modified. This allows the SW to acknowledge only the RX Header Descriptor for any RX messages. In Continuous Mode NEXT is always set to 0 as only one RX descriptor is used per RX message
	[27]	IRQ	SW only	Interrupt: when set to 1, an interrupt is triggered to the system when the descriptor execution is complete and a correctly received RX message was written to it. This interrupt can provide point to the SW a synchronization point to monitor the RX FIFO Queue execution
	[26:25]	Not Used	SW only	Must be set to 0
	[24:16]	CRC[8:0]	SW only	CRC: this CRC is computed by the SW for the current RX descriptor. It must consider all elements assuming this bit-field as set to 0. Any CRC error is triggering an interrupt to the system. The CRC is not evaluated if the Ni_MH_STS.RX_DESC_CRC_EN bit is set to 0
	[15:12]	FQN[3:0]	SW only	RX FIFO Queue number: define the RX FIFO Queue number allocated to this RX descriptor
	[11:9]	IN[2:0]	SW only	Instance Number: define the XCAN instance number using that descriptor. This bit-field is relevant if several XCAN are running concurrently. It provides a way to detect descriptor fetch issue between instances. The value defined must be equal to the one defined in the Ni_MH_CFG.INST_NUM bit-field register.

(table continues...)

Table 1043 (continued) RX descriptor description

Element number	Bit-field	Name	Managed by	Description/Constraints
	[8:4]	RC[4:0]	SW only	Rolling Counter: use to track the order of RX descriptor fetched when an RX FIFO Queue is running. When an RX FIFO Queue is started for the first time, its first RX descriptor must have the RC[4:0] set to 5'b00000. This value must be incremented for every new RX descriptor up to 5'b11111 and then back to 5'b00000. Even if a wrap occurs at to the end of the RX FIFO Queue, there must be continuous. Therefore, the First RX descriptor may have a RC[4:0] different from 5'b00000.To have always RC[4:0] = 5'b00000 for the First RX descriptor, the RX FIFO Queue size must be a multiple of 32 RX descriptor size
	[3:0]	STS[3:0]	MH only	Status: gives the status of the RX message received. This bit-field is written back by the MH when the descriptor has been completed. This bit-field must be set to 0 by SW. 0000_B: none 0001_B: message received successfully 0010_B: message received but not filtered 0011_B to 1110_B: reserved 1111_B: message acknowledged with parity error
1	[31:0]	RX_AP	SW/MH	Normal Mode: the SW defines the address of the RX data container to write RX data Continuous Mode: The SW must set this bit field to 0 as default value. The MH writes this field with the address pointer to find the RX message attached to the RX descriptor. Only the RX Header Descriptor is having this bit field updated, with the RX message address in the data container. This address must be 32 bit aligned, the two LSB bits are assumed to be always 0
2	[31:0]	TS0[31:0]	MH only	Timestamp 0: LSB of the 64-bit timestamp of the successfully received RX message (only valid when HD bit is set to 1)
3	[31:0]	TS1[31:0]	MH only	Timestamp 1: MSB of the 64-bit timestamp of the successfully received RX message (only valid when HD bit is set to 1)

Here is the list of required elements for the various RX descriptor definition to be managed by the SW or the MH:

Table 1044 Element managed by SW

	SW to write information to MH	SW to read information from MH	
Element number	**RX Descriptor**	**Header descriptor**	**Trailing descriptor**
0	Mandatory	Mandatory	Mandatory in Normal mode
1	Mandatory in Normal mode NA in Continuous mode (must be set to 0)	Mandatory	Mandatory in Normal mode
2	NA (must be set to 0)	Mandatory	NA (must be equal to 0)
3	NA (must be set to 0)	Mandatory	NA (must be equal to 0)

Table 1045 Element managed by MH

	MH to write information to SW		MH to read information from SW
Element number	**Header descriptor**	**Trailing descriptor**	**RX descriptor**
0	Mandatory	Not updated	Mandatory
1	Not updated in Normal mode Mandatory in Continuous mode	Not updated	Mandatory in Normal mode NA in Continuous mode
2	Mandatory	Not updated	NA
3	Mandatory	Not updated	NA

When the Element´s content is mentioned as NA, the assumed default value must be 0.

22.5.3.7.1 CRC computation

A dedicated CRC is computed for every RX descriptor. When a CRC issue is detected, the DESC_ERR interrupt signal is triggered (see safety measures section). This way the data transfer setting and description up to the DMA engine are fully protected.

The CRC covers all the relevant data, meaning the 55 bits of data in the RX descriptor considering the CRC bit field in the descriptor as equal to 9'b0. The CRC is part of the Element Number 0.

The CRC (CRC-9_167) is computed assuming the following elements in sequence:

Element Number 0[31:25] & 9'b0 & Element Number 0[15:0]

Element Number 1[31:0]

***Note**: The Koopman representation of the polynomial CRC-9_167is used to protect RX descriptors: CRC-9_167 = ($x^9+x^7+x^6+x^3+x^2+x+1$) (CRC polynomial in implicit "+1" hex format, meaning the trailing "+1" is omitted from the polynomial number)*

Using the Ni_MH_SFTY_CTRL.RX_DESC_CRC_EN bit register, the SW can decide to disable this check for all the RX descriptors fetched from S_MEM or L_MEM.

22.5.3.7.2 RX Descriptor Errors

When an RX descriptor error is detected, the relevant information are logged in the Ni_DESC_ERR_INFO1 register. Furthermore, the source address of the faulty RX descriptor is logged in the Ni_DESC_ERR_INFO0 register. This would help the SW to identify potential root causes when such error occurs. The Ni_DESC_ERR_INFO1.RX_TX bit register is set to 1 when an RX descriptor gets an error.

22.5.3.8 RX message header definition

Messages received from the CAN Bus are stored in the S_MEM, each consisting of a header followed by the payload. The header data structure depends on the CAN Frame Format (Classical CAN, CAN FD, CAN XL) used for this message on the CAN Bus. It can be identified by the header bits FDF and XLF. The following tables describe the three data structures used for the headers, consisting of the words R0, R1 and R2.

Table 1046 Classical CAN RX header definition

Rn	Bits	Name	Source	Description/Constraints
R0	[31]	FDF	CAN	FD Format
	[30]	XLF	CAN	XL Format
	[29]	XTD	CAN	Extended Identifier
	[28:18]	BaseID [28:18]	CAN	Base ID
	[17:0]	ExtID [17:0]	CAN	Extended ID
R1	[31:27]	Reserved	Not applicable	Not Applicable
	[26]	RTR	CAN	Remote Transmission Request
	[25:20]	Reserved	Not applicable	Not Applicable
	[19:16]	DLC[3:0]	CAN	Data Length Code
	[15:11]	Reserved	Not Applicable	Not Applicable
	[10]	FAB	MH	Filter Aborted: when set to 1, the RX filtering process was ending before completing with no match
	[9]	BLK	MH	Black List: When set to 1, the RX message filtered belongs to a black listed
	[8]	FM	MH	Filter Match: When set to 1 one of the filter element (defined by FIDX[7:0]) has detected a match
	[7:0]	FIDX[7:0]	MH	Filter index: provide the information of the filter index which has been triggered
R2	[31:0]	Reserved	Not Applicable	Not Applicable

***Note**: Classical CAN frames (CBDF, CEDF, CBRF, CERF) can be identified by R0.FDF = R0.XLF = 0.*

Table 1047 CAN FD RX header definition

Rn	Bits	Name	Source	Description/Constraints
R0	[31]	FDF	CAN	FD Format
	[30]	XLF	CAN	XL Format
	[29]	XTD	CAN	Extended Identifier
	[28:18]	BaseID [28:18]	CAN	Base ID
	[17:0]	ExtID [17:0]	CAN	Extended ID
R1	[31:26]	Reserved	Not applicable	Not Applicable
	[25]	BRS	CAN	Bit Rate Switch
	[24:21]	Reserved	Not applicable	Not Applicable
	[20]	ESI	CAN	Error State Indicator
	[19:16]	DLC[3:0]	CAN	Data Length Code
	[15:11]	Reserved	Not Applicable	Not Applicable
	[10]	FAB	MH	Filter Aborted: when set to 1, the RX filtering process was ending before completing with no match
	[9]	BLK	MH	Black List: When set to 1, the RX message filtered belongs to a black listed
	[8]	FM	MH	Filter Match: When set to 1 one of the filter element (defined by FIDX[7:0]) has detected a match
	[7:0]	FIDX[7:0]	MH	Filter index: provide the information of the filter index which has been triggered
R2	[31:0]	Reserved	Not Applicable	Not Applicable

***Note**: CAN FD frames (FBDF, FEDF) can be identified by R0.FDF = 1 and R0.XLF = 0.*

Table 1048 CAN XL RX header definition

Rn	Bits	Name	Source	Description/Constraints
R0	[31]	FDF	CAN	FD Format
	[30]	XLF	CAN	XL Format
	[29]	Reserved	Not Applicable	Not Applicable
	[28:18]	Priority ID[28:18]	CAN	Priority Identifier

(table continues...)

Table 1048 (continued) CAN XL RX header definition

Rn	Bits	Name	Source	Description/Constraints
	[17]	RRS	CAN	Remote Request Substitution
	[16]	SEC	CAN	Simple Extended Content
	[15:8]	VCID[7:0]	CAN	Virtual CAN Network ID
	[7:0]	SDT[7:0]	CAN	SDU Type
R1	[31:27]	Reserved	Not applicable	Not Applicable
	[26:16]	DLC-XL[10:0]	CAN	Data Length Code with CAN XL encoding
	[15:11]	Reserved	Not Applicable	Not Applicable
	[10]	FAB	MH	Filter Aborted: when set to 1, the RX filtering process was ending before completing with no match
	[9]	BLK	MH	Black List: When set to 1, the RX message filtered belongs to a black listed
	[8]	FM	MH	Filter Match: When set to 1 one of the filter element (defined by FIDX[7:0]) has detected a match
	[7:0]	FIDX[7:0]	MH	Filter index: provide the information of the filter index which has been triggered
R2	[31:0]	AF[31:0]	MH	Acceptance Field

***Note**: CAN XL frames (XLFF) could be identified by R0.FDF = R0.XLF = 1.*

22.5.3.9 TX message

For a better understanding while reading this chapter, read the TX descriptor chapter first.

A TX message is defined using one TX descriptor and a TX data container where the payload data buffer is defined.

The Header Descriptor (or the only one, in case of one descriptor per message) holds the header data information and for some CAN protocols, the data payload of the message. Such descriptor also provides some additional information to the MH: the interrupt to be triggered, where to write acknowledge data, where to fetch TX message data, etc.

A TX data container is a general term to name the memory space allocated by the SW. This data container is used to hold the payload data buffer. In most of the cases, this TX data container would be identical to the data buffer size to transmit, avoiding the loss of memory space.

A specific TX descriptor is used for the TX FIFO Queue and for the TX Priority Queue due to the structure of the two different implementations.

In order to optimize the fetch of the TX descriptor as well as data payload, a maximum burst length of 8x32 bit is used.

The buffer size which can be defined in a TX descriptor can go up to 2048 byte. This way, a TX message can be defined using a single TX descriptor and one data buffer.

As the maximum efficiency is reached when using the maximum burst length, it is highly recommended to define a data buffer size aligned on the maximum burst length.

If the TX payload data is not a multiple of the burst length, the remaining data in the data container won't be read. Nevertheless, the embedded DMA_CONTROLLER will use the maximum burst length to read the payload whenever possible and will adapt the latest burst length to complete its transfer. Only the relevant data are read from S_MEM when smaller than the maximum burst length.

The address pointer used to fetch the payload data is always 32 bit, despite that payload data is byte aligned.

Every TX descriptor holding the header of the TX message, once a TX message is transmitted, is acknowledged for status, error reporting and timestamping.

Here below are the different types of messages according to CAN protocols.

Related information

TX descriptor on page 4187

22.5.3.9.1 Single TX descriptor usage

A TX message can be defined using one single TX descriptor. This kind of choice requires to have the complete payload data defined in one data container in the S_MEM. In case of Classical CAN, the complete Classical CAN message is embedded in the TX descriptor. This means no payload buffer is required for Classical CAN messages. The NEXT bit in TX descriptor must be set to 0. In case of a TX Priority Queue, the TX descriptor TDO bit field must be set to 0. For the TX FIFO Queues, the NHDO is set to a value equal to 1 in order to define the next TX header descriptor.

For the TX Priority Queue and TX FIFO Queue the same description below applies.

Classical CAN with up to 8 bytes payload

As the Classical CAN payload data is only 8 bytes, it can be defined completely in the TX descriptor (see TD0 and TD1 in chapter TX descriptor). There is no need to define an address pointer to a payload buffer in that case. Despite a data container is mentioned, it is not used. This is to align with the other description in the next sections.

This approach provides a single and simple way to send any Classical CAN TX message in a straight forward manner. Using the T0, T1, TD0 and TD1 in the TX descriptor, the overall Classical CAN message can be defined, refer to the TX descriptor chapter for more details.

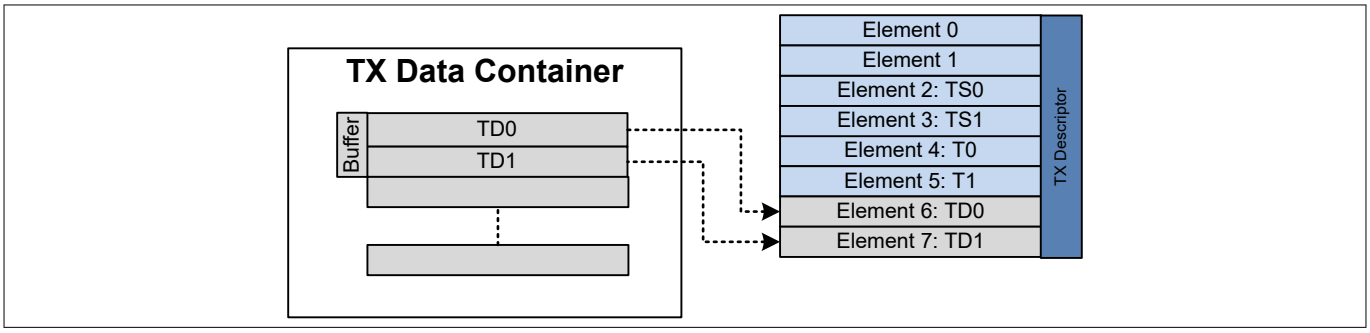


Figure 399 Classical CAN TX message with 8 bytes payload (single descriptor)

CAN FD

As the CAN FD protocol can provide up to 64 bytes, it is mandatory to define an address pointer to read the payload data from the S_MEM when the size is greater than 4 bytes. The first payload data defined in the payload data buffer also needs to be defined in the TX descriptor. For high latency system, the time to fetch the payload data, once the arbitration process is complete, can lead to a potential under-run. To solve this issue, TD0 is declared in the TX descriptor. By the time TD0 is sent through the CAN bus, the payload data will be read from the S_MEM. This approach avoids pre-fetching the payload data before having the arbitration result and to

throw away the complete burst when arbitration is not successful. The TD0 from the first read burst access will then be skipped.

The address pointer points to the buffer holding the overall payload data as depicted in figure below. In case only 4-byte payload data is required, there would be no need to define the address pointer (must be set to 0). For payload data above 4 bytes an address pointer is required. The minimum data container size is either 32 bytes (data payload lower or equal to 32 bytes) or 64 bytes (data payload greater than 32 bytes).

The size of the buffer to be fetched is always 32 bit aligned. When the data payload is lower than a multiple of 32 bits, padding is expected and will be discarded by the PRT.

Using the T0, T1, TD0 and the TX_AP fields, the overall CAN FD TX message can be defined, refer to the TX descriptor chapter for more details

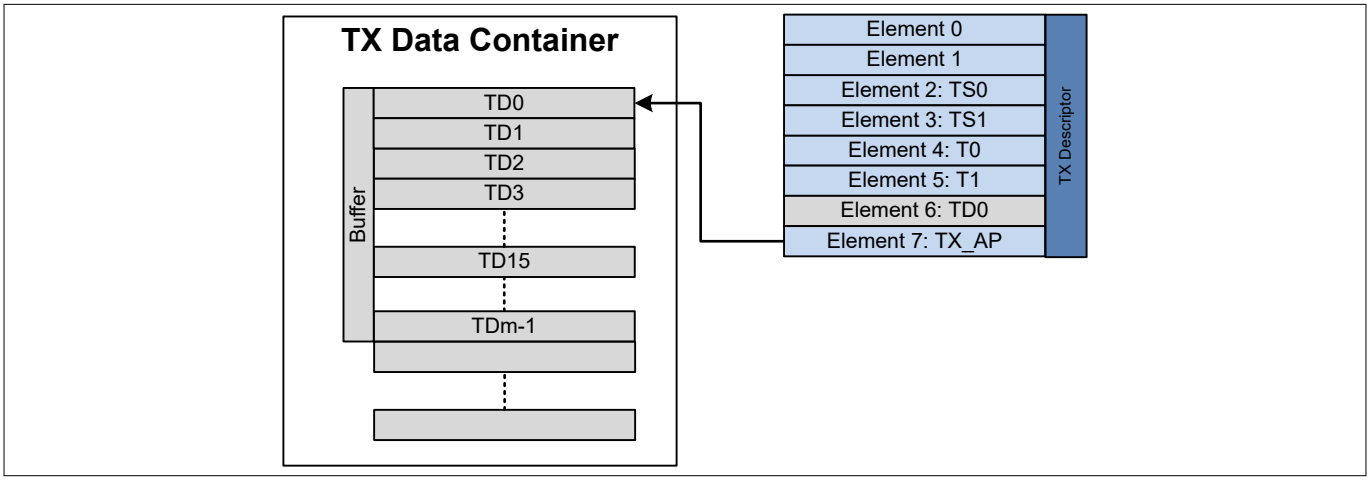


Figure 400 CAN FD TX message with more than 4 byte payload (single descriptor)

CAN XL

As the CAN XL header information requires 3 words of 32 bits, there is no payload data defined in the TX descriptor. T2 is required only when the arbitration on the CAN bus is successful, giving time for the MH to read the payload data from the S_MEM and to avoid the need of prefetching data.

Using the T0, T1, T2 and the TX_AP fields, the overall CAN XL TX message can be defined, refer to the TX descriptor chapter for more details.

The size of the buffer to be fetched is always 32 bit aligned. When the data payload is lower than a multiple of 32 bits, padding is expected and will be discarded by the PRT.

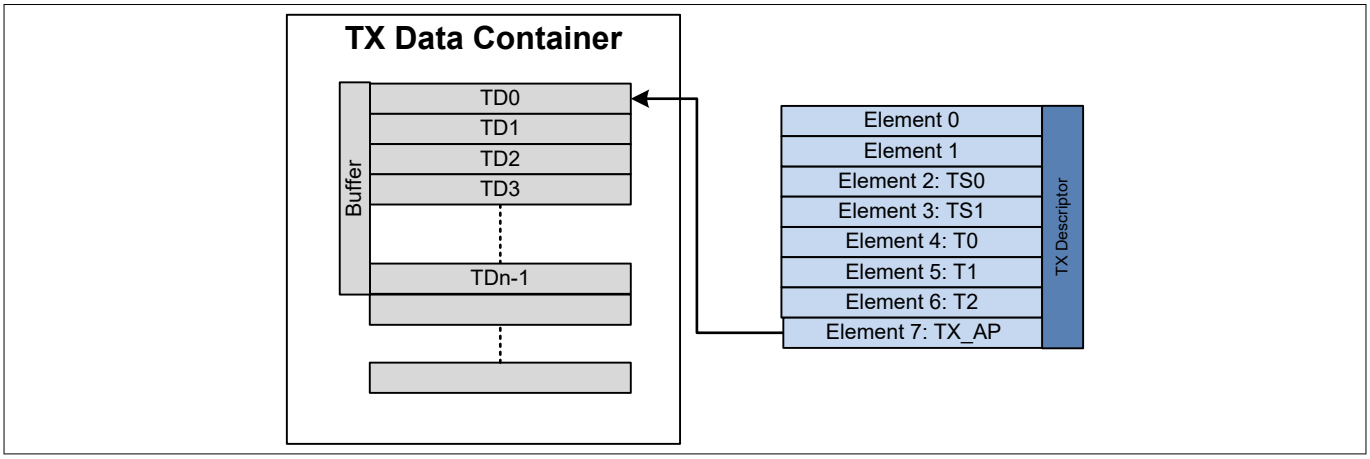


Figure 401 CAN XL TX message (single descriptor)

Related information

TX descriptor on page 4187

22.5.3.10 RX message in normal mode

In order to receive RX messages, an RX descriptor is required to define how the MH must behave and where to write the RX data in Normal mode.

Those RX descriptors are attached to RX FIFO Queues which are selected according to the RX filtering rules. It means, RX descriptors are concatenated and read in sequence.

An RX data container is a general term to name the memory space allocated by the SW. This data container is used to hold the RX message data. In most of the cases, this RX data container would not be fully filled with data, maximum data payload being different for CAN protocols.

Every RX descriptor is assigned to a data container to write incoming data to the S_MEM. The RX data container size is a multiple of the maximum burst length supported, 8x32 bit with a maximum of 4064 bytes (127*32 bytes) and a minimum of 32 bytes. This granularity does provide some flexibility to address several RX message size with only one data container. As defined previously, if an RX data container is smaller than an RX message, several RX descriptors will be assigned to that message.

Compared to the TX message, the header and the payload of the RX message are written together to the S_MEM. This approach gives the flexibility to pass address pointers of the overall message to the application and to avoid copies.

If the payload data does not cover a multiple of the burst length, some data won't be written to the data buffer in the container. The embedded DMA_CONTROLLER will use the maximum burst length whenever possible to write header and payload and will adapt the latest burst length to complete its transfer.

The address pointer used to write the RX message is always 32 bit aligned despite payload data is byte aligned.

The size of the data container defined into the RX descriptor is fixed for a given RX FIFO Queue and for all the RX descriptors of that queue. The smaller the size of data buffer the less RX descriptors a message would require.

As the data container is defined anywhere into the S_MEM, the SW can decide to allocate all the data containers into a continuous way in the S_MEM to ensure, the RX message is not split over several location. It will ease the reading of RX messages and simplify the management of data buffers, see RX FIFO Queue chapter for more details.

The NEXT bit defined into the RX Header Descriptor provides the information to the SW that one or several RX descriptors are used. On top of it, the RX Header Descriptor of an RX message will have the HD bit set to 1 to indicate that the data container got the header of the message.

Only the RX Header Descriptor holding the header data is acknowledged when an RX message is received. This way, despite receiving the timestamp at the end of the data received, it will be written with the header and status reporting.

Here below are the different types of RX messages according to the CAN protocol and some different structures when using one or several RX descriptors.

Related information

RX FIFO queue in normal mode on page 4217

22.5.3.10.1 Single RX descriptor

With such structure the size of the data container defined by the RX descriptor must be large enough to hold the maximum payload size of the expected RX message to receive.

Classical CAN

As depicted in the following figure, the Classical CAN header and payload data can be directly written into a 32 bytes data container ($N = 1$). If such data buffer size is defined then several RX descriptors would be required to support CAN FD (3 RX descriptors) or CAN XL (65 RX descriptors) frame size. It is important to note that according to the RX message type received, the whole message may be written into a bigger data container as every RX FIFO Queue is defining its own data container size.

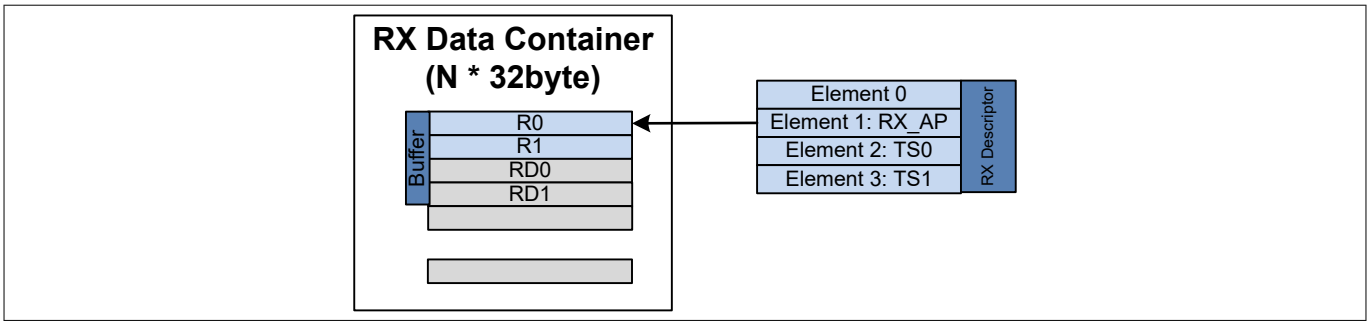


Figure 402 Classical CAN RX message (single descriptor)

CAN FD

Compared to Classical CAN, a larger data buffer is required to hold up to 64 bytes of payload data and the header message data. In this case a data container of 96 bytes ($N = 3$) is allocated to support CAN FD frame format. There will be no issue regarding Classical CAN message as it would fit entirely into the same data container. Doing so, the CAN XL message can be supported but would require up to 22 RX descriptors.

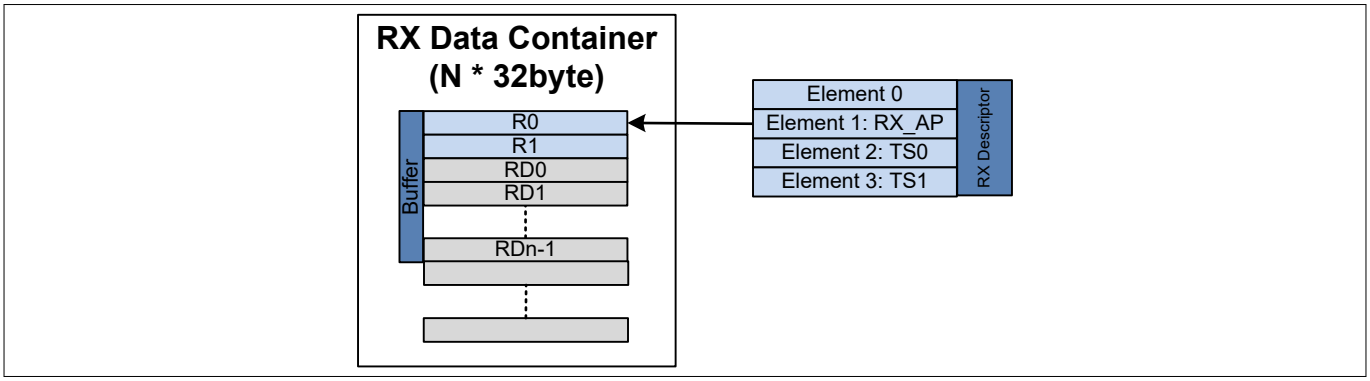


Figure 403 CAN FD RX message (single descriptor)

CAN XL

To ensure only one RX descriptor is pointing to one RX message a data container size of more than 2048 bytes is required ($N = 65$). With this setting, all the different CAN protocols are covered with a single data container per RX descriptor. However, quite some memory space is lost in the data container (when configure to support CAN XL payload size) when receiving Classical CAN or CAN FD messages. To solve this issue, multiple RX descriptors can be used, see next chapter.

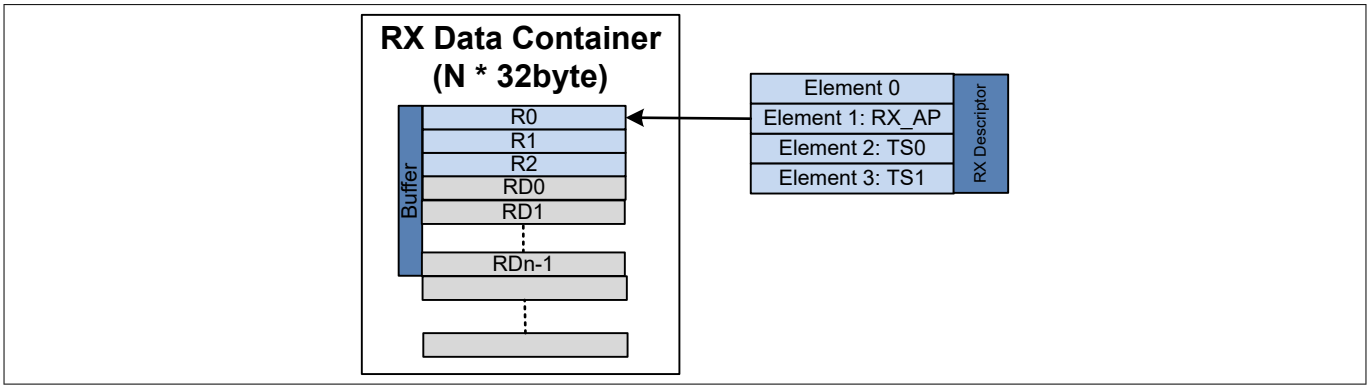


Figure 404 CAN XL RX message (single descriptor)

22.5.3.10.2 Multiple RX descriptors

To optimize the memory usage, regardless of the payload size of the RX message received, several RX descriptors can be assigned to one RX message. Doing so, the RX message is written in several data containers. When one is full, the RX message data is going to the next one. As depicted in the figure below, for a given size of data container (constant per RX FIFO Queue), the RX message can be written anywhere into the S_MEM. The MH takes care of filling the right data container with the RX message data whenever required. As a fixed memory allocation is defined per RX descriptor, the RX message data may be spread over several data containers and RX descriptors (depends on RX message payload data).

The figure below shows three RX descriptors and their assigned data container to hold the entire RX message. If a data container has a size of 96 bytes (N=3) and a CAN XL message payload of 270 bytes is received, then the RX message is depicted in figure below. Although the CAN XL message, in this example, is split over several RX descriptors, this configuration allows to support Classical CAN and CAN FD with only one RX descriptor.

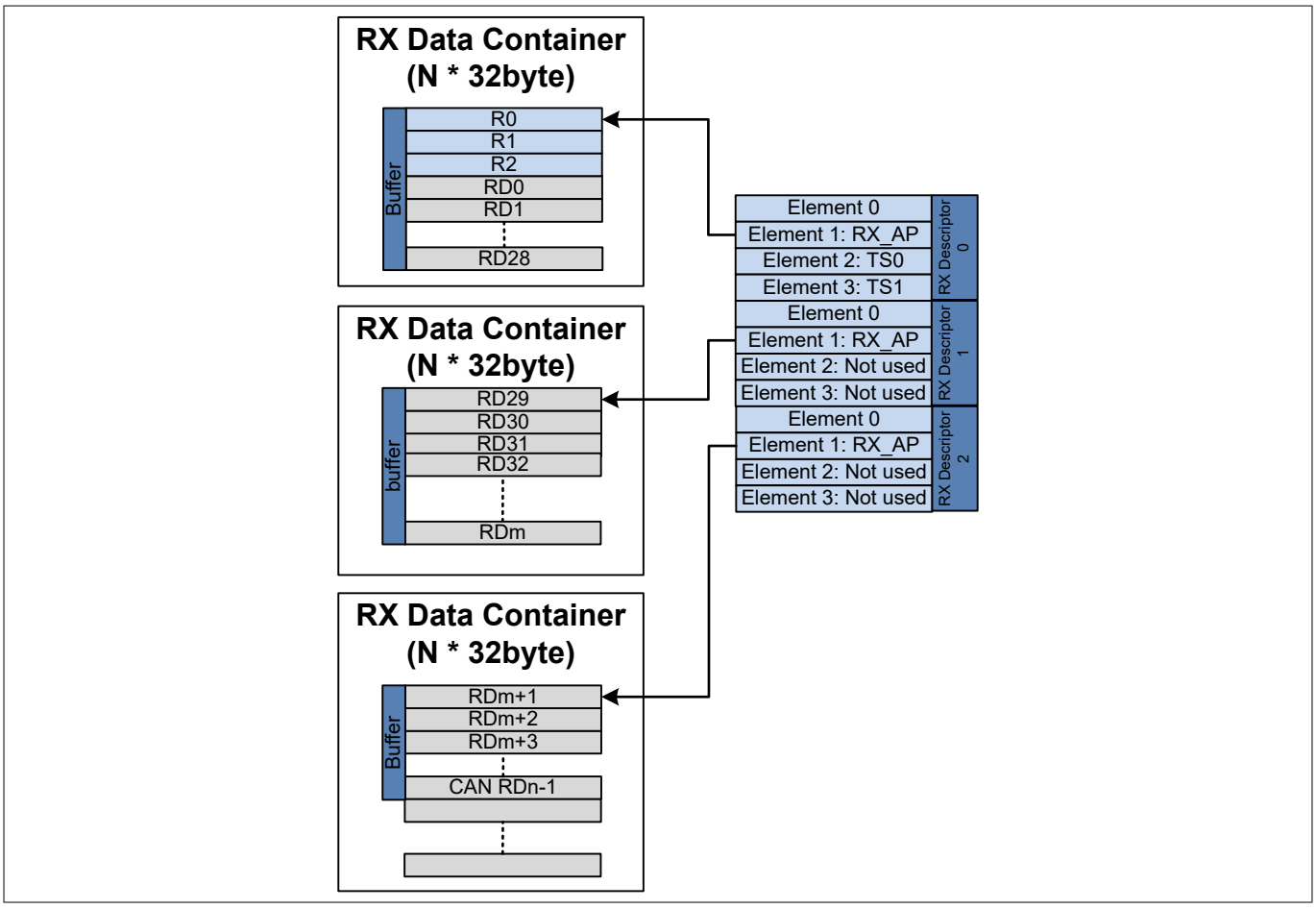


Figure 405 RX message (multiple descriptors)

22.5.3.11 RX message in continuous mode

For a better understanding of this section please read the RX Message in Normal mode chapter.

In the Continuous mode, the RX messages, instead of being split over several data container (see RX Message in Normal mode chapter), are merged in the same big data container one after the other. As depicted below only a single data container is defined per RX FIFO Queue and one RX descriptor is used per RX message.

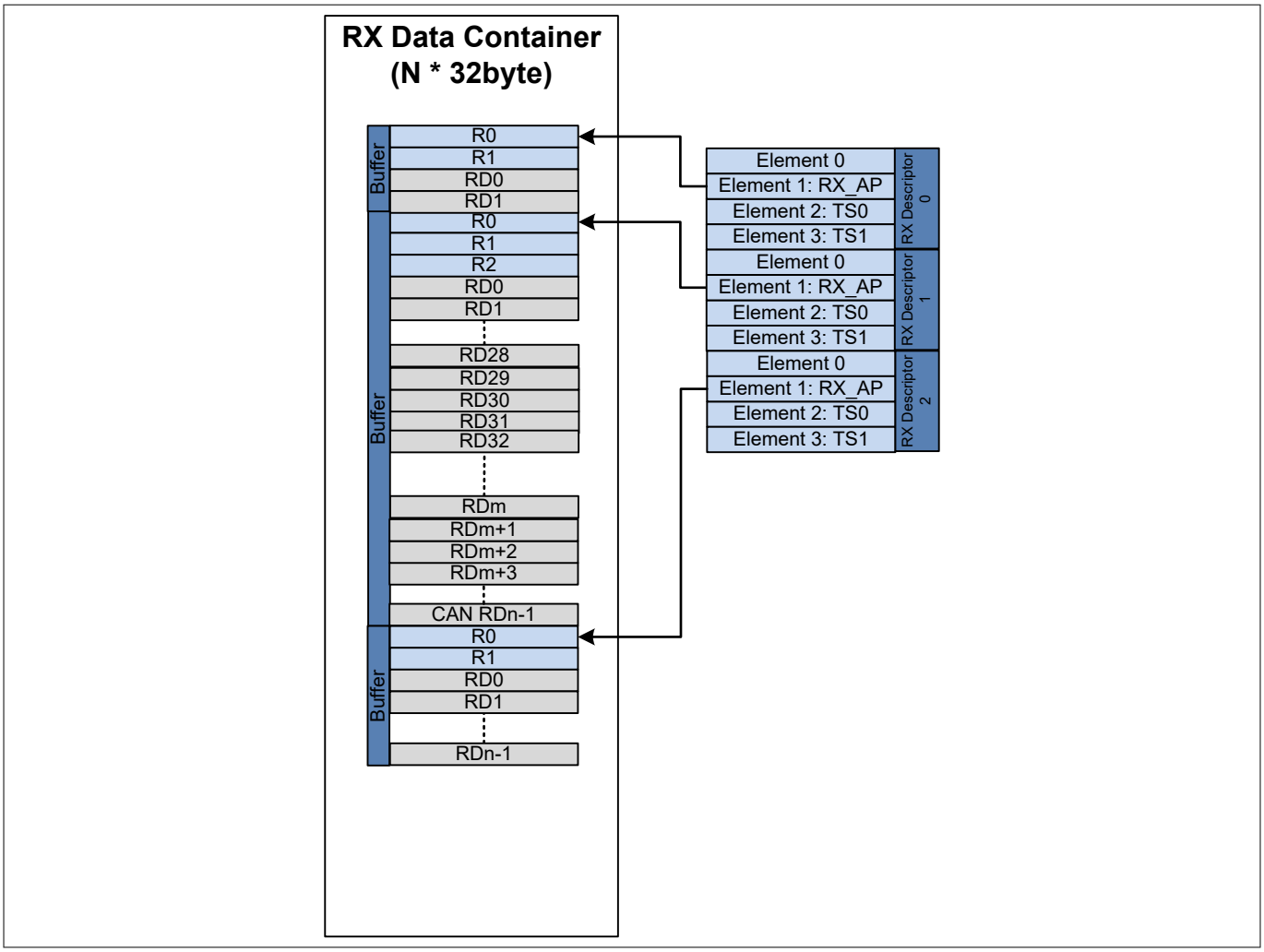


Figure 406 RX message (continuous mode)

The Continuous mode makes use of the already defined RX descriptor list to support the SW management of RX messages, see RX message Normal mode for more details.

It is important to note that the Continuous mode applies to all RX FIFO Queues when set. There is no option to make it available only to some queues.

The RX descriptors are attached to a defined RX FIFO Queue. The RX FIFO Queue, to write the RX message, is defined according to the RX filtering rules, see RX Filter chapter for more details. Once the RX FIFO Queue is identified, the latest RX descriptor (meaning the current one) is fetched from S_MEM. As the RX descriptors in a given FIFO Queue are concatenated, they will be read in sequence up to the end of the RX message.

Every RX descriptor is assigned to only one RX message in this large data container. This data container size is a multiple of the maximum burst lengths supported, 8x32 bit with a maximum of 131040 bytes (4095*32 byte) and a minimum of 32 bytes. Every RX FIFO Queue has its own data container (defined by a start address and a size).

The header and the payload of the RX message are written one after the other to the S_MEM. This approach gives the advantage to have the complete RX message data available in one place in the S_MEM. A copy of the RX message in the S_MEM can then be easily defined by a start address and a size.

Whenever it is possible, the embedded DMA_CONTROLLER uses the maximum burst length to write header and payload data and will adapt the latest burst length to complete its transfer.

The address pointer, used to write the RX message, is always 32 bit aligned despite payload data is byte aligned.

The data container is defined anywhere in the S_MEM. Being defined as a 32 bit address pointer, it can be defined in a 4 Gbyte memory area.

The NEXT bit defined in the RX Descriptor will never be set, as only one RX Descriptor is used per RX message. The RX Descriptor pointing to the RX message has the HD bit set to 1 as it is a Header Descriptor. No Trailing Descriptors are used for such mode, only one RX descriptor is required and it is a Header Descriptor.

Only the RX Header Descriptor holding the header data is acknowledged when an RX message is received, considering the Normal mode. The same applies for the Continuous mode, see RX message in Normal mode chapter for more details.

Compared to the Normal mode, there is no tradeoff to consider regarding the different CAN protocol payload data size. As the RX messages are written in a row, no loss of memory is expected in the data container assigned to an RX FIFO Queue.

This mode will also ensure that RX data are always linearly and continuously written in the S_MEM. At the beginning of the reception of an RX message header, a check is performed to ensure the RX data to be received can fit entirely in the data container. As the RX message cannot be written at the bottom and at the top of the data container, the MH will go to the start address of the data container before writing the first data. This would provide to the SW an easy way to perform memory copy, as one start address and a size can define the overall RX message.

22.5.3.12 Descriptor acknowledgement

For the TX and RX paths, the MH is providing data information back to the RX and TX Header Descriptor.

To do so, some place holders are defined in the RX and TX descriptors for the MH to write RX and TX message status, timestamping and error reporting.

As the CAN bus is not a full duplex interface, there should not be any collision on the acknowledge of RX and TX descriptors, with the exception of the PRT when in loopback mode. In such mode, all TX messages transmitted by the MH are send back by the PRT to the MH, refer to the PRT chapter for detailed description of the loopback.

The process of acknowledgment is completely separated from the reception or transmission of a CAN frame, a dedicated DMA channel is reserved for such purpose.

22.5.3.12.1 RX descriptor

For the RX path, one or several RX descriptors can be used to hold the complete RX message. Once an RX message is received successfully, an acknowledgement is written back to the Header Descriptor when the message is completed. If several descriptors are used per message (Trailing Descriptors), they are not changed by the MH. If the data container assigned to the RX descriptor is sized in such a way that any RX message can fit in entirely, then every RX descriptor (in fact Header descriptors in that case) will be acknowledged.

If several RX descriptors are used to store the RX message and an issue occurs while processing the message, all RX descriptors already used are then released for the next RX message.

Here below is the list of bit fields used by the MH to provide the acknowledgement information to the RX Header descriptor, see RX Descriptor chapter for details:

- VALID: The MH expects this bit to be set to 0 by the SW to ensure the data container is ready to be written again. This bit is written by the MH to 1, when an RX message is received successfully and the data are available in S_MEM. It is true only for the RX descriptor holding the RX message header data
- TS0[31:0] and TS1[31:0]: The MH writes the 64-bit timestamp (TS0 and TS1) in the RX descriptor when the RX message data is received successfully. Only the RX descriptor holding the data will have this bit-field updated
- NEXT: As soon as more than one RX descriptor is required for an RX message, the MH is setting this bit to 1 to indicate to the SW, more descriptors are used for this RX message. Only the RX Header descriptor will have this bit-field updated. The trailing descriptors are not updated. When in Continuous mode this bit will always be 0

- HD: In case several RX descriptors are used to define an RX message, the MH is setting this bit to 1 to identify which descriptor has the message header embedded into its data buffer
- STS[3:0]: This bit-field gets updated by the MH for any usage of RX descriptor. It provides information on the status of the RX message and on any related issue while RX FIFO Queues are running

22.5.3.12.2 TX descriptor

For the TX path, once a TX message is processed, the TX Header Descriptor is written back with the relevant information. The list of conditions to trigger an acknowledge is defined below:

- Message sent successfully
- Message rejected by the TX filter (see TX Filter chapter)
- Message discarded after several retransmission
- Message rejected by the PRT (see HFI codeword in PRT chapter)

Here below are the list of bit-field used by the MH to provide the acknowledgment information to the TX descriptor, see TX Descriptor chapter for details:

- VALID: The MH expects this bit to be set to 1 by SW to ensure the data buffer is ready to be sent, only the TX descriptors having this bit set to 1 are accepted and executed. This rule applies for every TX descriptor with or without the header data (when TX message is split over several descriptors). When the last data defined by this TX descriptor has been sent over the CAN bus, it will be set back to 0 by the MH only to the TX descriptor holding the header data
- TS0[31:0] and TS1[31:0]: When the TX message data is sent successfully a 64-bit timestamp is written back into to the TX descriptor holding the header data
- STS[3:0]: This bit-field gets updated by the MH for any usage of TX descriptor. It provides information on the status of the TX message and on any related issue while TX FIFO Queues are running

22.5.3.13 TX FIFO queue

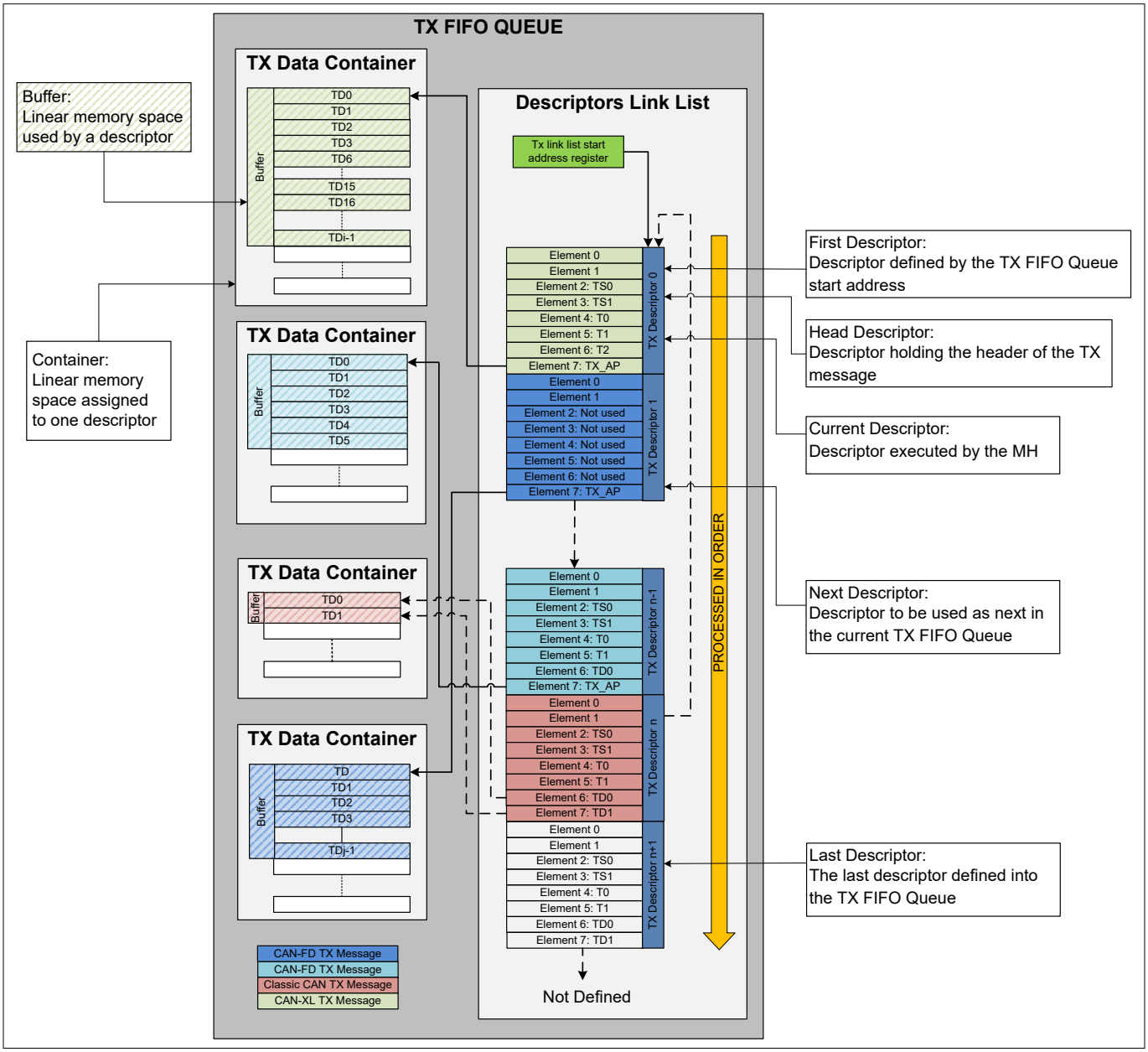


Figure 407 TX FIFO queue description

Up to 8 TX FIFO Queues can be defined and managed by the MH.

When the SW wants to configure N TX FIFO Queues, only the queue number from 0 to N-1 can be used.

A TX FIFO Queue is a list of TX messages to be sent in order to the PRT.

Each one being fully independent from the others, the SW can declare and add new messages to any of the FIFO Queue without stopping the execution of the others or the current one. In this sense, the TX FIFO Queues can be enabled or disabled individually. An abort mechanism is provided to stop and flush each TX FIFO Queue individually.

Prior to launch any TX FIFO Queue, the MH must be started (Ni_MH_CTRL.START written to 1 will drive the Ni_MH_STS.BUSY bit status to 1). To start the TX FIFO Queue n, write 1 to the Ni_TX_FQ_CTRL0.START[n]. Before launching a TX FIFO Queue n, it must be enabled by setting the Ni_TX_FQ_CTRL2.ENABLE[n] bit to 1. Once enabled and started, there is no way to disable it while it is running without a defined procedure. Instead, the abort bit Ni_TX_FQ_CTRL1.ABORT[n] provides a way to stop a TX FIFO Queue n running and to ensure a safe

stop and flush of ongoing data. For more detail on starting and stopping TX FIFO Queues, refer to the Application Information chapter.

To ensure no dead lock can occur at start, the ENABLE signal from the PRT must be set to high to allow any TX FIFO Queue to start. This signal status can be monitored in the Ni_MH_STS.ENABLE bit field.

There is also nothing preventing the SW to declare and run a TX FIFO Queue with a defined list of TX messages, assuming an interrupt at the end of the TX FIFO Queue execution.

However, the TX FIFO Queue can be used as a circular buffer when the Last Descriptor defines a wrap to the First Descriptor (WRAP bit set to 1 in TX descriptor). Doing so, the SW can add new messages in an endless manner over time.

The mechanism, used to manage TX FIFO Queues, is based on the concept of linked list. Any TX FIFO Queue is defined using a linked list of TX descriptors and data buffers to read TX message payload from the S_MEM.

A linked list is made of descriptors, where a descriptor is defined by several data elements of the same size, the element is a 32-bit word. Each element provides some information or would define some actions to perform. A descriptor is built by the SW but will be read and executed by the MH.

Every TX descriptor is of the same size, pointing to a data buffer and also to the next descriptor, as depicted in the figure above. The TX descriptors are continuous in memory (to ease and simplify implementation). Therefore, it is not required to declare or use a bit field to mention the position of the next descriptor as it is implicit.

The linked list is started by fetching the First Descriptor in the list, once it is fully read, it is executed and the data buffer assigned to it, is read. Other actions can be defined into the element data like triggering an interrupt or setting a flag. The linked list is processed one descriptor at a time, once a descriptor is completed, the next one is fetched into the list and the process repeats itself. The process keeps going up to the Last Descriptor of the link list and from this point in, may end or may wrap to the first descriptor in a circular buffer mode.

Every TX FIFO Queue defines its own order of TX messages to be send to the CAN bus, but as several queues are running concurrently, an arbitration process is performed between queues to select the highest priority message. Every TX message is filtered to ensure only the required ones can be sent. The SW builds those queues with messages and the MH takes care of sending them whenever appropriate.

For the TX FIFO Queues, data buffers hold the payload data of the TX message while the descriptor defines header information. In some cases, the first payload data may also be part of the Head Descriptor.

To give a status report and some information like timestamping, the MH is also able to write back some elements in the TX descriptors. Not all of them are written back but only the one having the Header Descriptor data are updated.

It is possible to wrap at the top of the TX FIFO Queue any time but with the following constraints:

- The WRAP bit must be set in the Header Descriptor to identify where is the next TX message

The descriptors are mainly defined on SRAM as they drive the actions to be taken. Nothing prevents the SW to declare and use the E_MEM instead but it can slow down the execution and may create real time issues. However, the data buffers can be either in E_MEM, which is usually the case, or in SRAM. As a matter of fact, if the next descriptor cannot be fetched before the relevant data are fully read or written, the link list execution speed would depend on the data access time. To solve this issue, read prefetchings are performed to hide the system latency whenever possible.

TX Data Containers can be defined at any location in S_MEM. But for performance reason and to optimize the burst access, it is highly recommended to have the TX buffer 32 byte aligned. Those containers are considered as memory space that is allocated by the Memory Management Unit to store buffer. Once a message is sent, the container can be de-allocated, so the memory space is released for further usage.

As soon as the TX FIFO Queue is started, the MH will fetch the First Descriptor and store it to L_MEM. When the TX descriptor is available in L_MEM, it will be part of the arbitration process. As long as the TX message defined by this TX descriptor is not sent to the CAN bus, it will remain for all the arbitration runs. When it is sent successfully, the next TX descriptor of that TX FIFO Queue is fetched automatically.

The MH will proceed with all TX FIFO Queues in the same way. As the TX message to be sent is based on its priority, the TX FIFO Queues will run at a different rate up to the point that all TX messages are sent successfully.

If the Last Descriptor of a TX FIFO Queue sets the END bit, the MH will end the FIFO execution as soon as the TX message defined is transmitted successfully on the CAN bus and the TX descriptor acknowledge is written to the S_MEM.

Up to 1023 TX descriptors can be defined for a TX FIFO Queue. When the maximum number of TX descriptor defined for a TX FIFO Queue is reached, the MH wraps automatically to its initial start address to fetch the next TX descriptor. Despite this default behavior, it is still possible at any time for the SW to mention a wrap using the WRAP bit in the Header Descriptor.

When the END bit is not set for the last TX descriptor, the TX FIFO Queue is considered as endless and any new TX descriptor can be appended to the already defined last descriptor. To allow such way of working, the last descriptor must always be not valid (VALID bit set to 0). This is very important as the detection of the non-valid TX descriptor triggers an interrupt to the system to declare that the TX FIFO Queue is stopped. It would then be up to the SW to append a new TX descriptor and restart the TX FIFO Queue.

If for some reasons a TX FIFO Queue has an error, it is still possible to abort the execution of that TX FIFO Queue. When such action is performed, the TX FIFO Queue will be considered as active as long as the current data transfer assigned to a TX descriptor is not finished. This means, the TX descriptor is not considered for the arbitration anymore, so no more fetches are done and the TX FIFO Queue is set inactive.

Any safety issue related to a TX descriptor executed by a TX FIFO Queue will stop it right away. The TX FIFO Queue is declared as no more valid and is stopped. To identify such issue, some interrupts are triggered to the system, TX_CRC_ERR and TX_SFTY_STS. Despite that the faulty TX FIFO queue is stopped, the others will keep going.

If a message has reached maximum number of retransmission or has declared an invalid header format, the message is skipped and the next one in the TX FIFO Queue is considered instead. The error mentioning such skip is written back to the report status bit field in the TX Header Descriptor.

In a context where a TX descriptor provides the definition of one TX message, the next TX message is the next TX descriptor, an offset of 1 (1x32byte) is required.

A TX FIFO Queue is controlled and monitored using several registers and bit registers:

- The Ni_TX_FQ_START_ADD{n} (where n is 0-7) register to define the start address of the TX FIFO Queue n
- The Ni_TX_FQ_CTRL0.START[n] (where n is 0-7) register to launch the TX FIFO Queue n
- The Ni_TX_FQ_SIZE{n} (where n is 0-7) register to define the maximum number of TX descriptor for the TX FIFO Queue n before looping back to the initial start address
- The Ni_TX_FQ_ADD_PT{n} (where n is 0-7) register to monitor the current address pointer of the TX FIFO Queue n
- The Ni_TX_DESC_ADD_PT register to monitor the current address pointer
- The Ni_TX_FQ_CTRL1.ABORT[n] (where n is 0-7) bit register to abort the execution of the TX FIFO Queue n
- The Ni_TX_FQ_CTRL2.ENABLE[n] (where n is 0-7) bit register to enable the TX FIFO Queue n prior to use it
- The Ni_TX_FQ_INT_STS.SENT[n] and Ni_TX_FQ_INT_STS.UNVALID[n] (where n is 0-7) bit registers to identify respectively, a message is transmitted and an invalid TX descriptor is detected
- The Ni_TX_FQ_STS0.BUSY[n] and Ni_TX_FQ_STS.STOP[n] (where n is 0-7}) bit registers to know respectively, the status of the TX FIFO Queue n, busy (TX FIFO Queue is active) and stopped or running
- The Ni_TX_FQ_STS1.ERROR[n] and Ni_TX_FQ_STS.UNVALID[n] (where n is 0-7) bit registers to identify the root cause of the TX FIFO Queue being stopped, an error is detected or an invalid TX descriptor is detected

A TX FIFO Queue is being controlled for any issue using common bit registers when receiving interrupts:

- The Ni_SFTY_INT_STS.TX_DESC_CRC_ERR and Ni_SFTY_INT_STS.TX_DESC_REQ_ERR bit registers to identify respectively, any CRC issue on TX descriptor running in the TX FIFO Queue n and non-expected TX descriptor

- The Ni_ERR_INT_STS.DP_TX_ACK_DO_ERR bit register to identify overflow on TX ACK data path for the TX FIFO Queues
- The Ni_ERR_INT_STS.DP_TX_SEQ_ERR bit register to identify if an issue occurs on the TX_MSG interface

22.5.3.13.1 Basic mode

The SW defines one TX descriptor per TX message payload data. Thus, a TX message would be:

- One TX descriptor to provide the complete header information
- One TX data container to hold the complete TX message payload data (only required for CAN FD and CAN XL when payload data is over 8 bytes)

Data containers holding the TX payload buffer can be declared anywhere in the S_MEM despite being attached to only one TX descriptor, as depicted in the following figure.

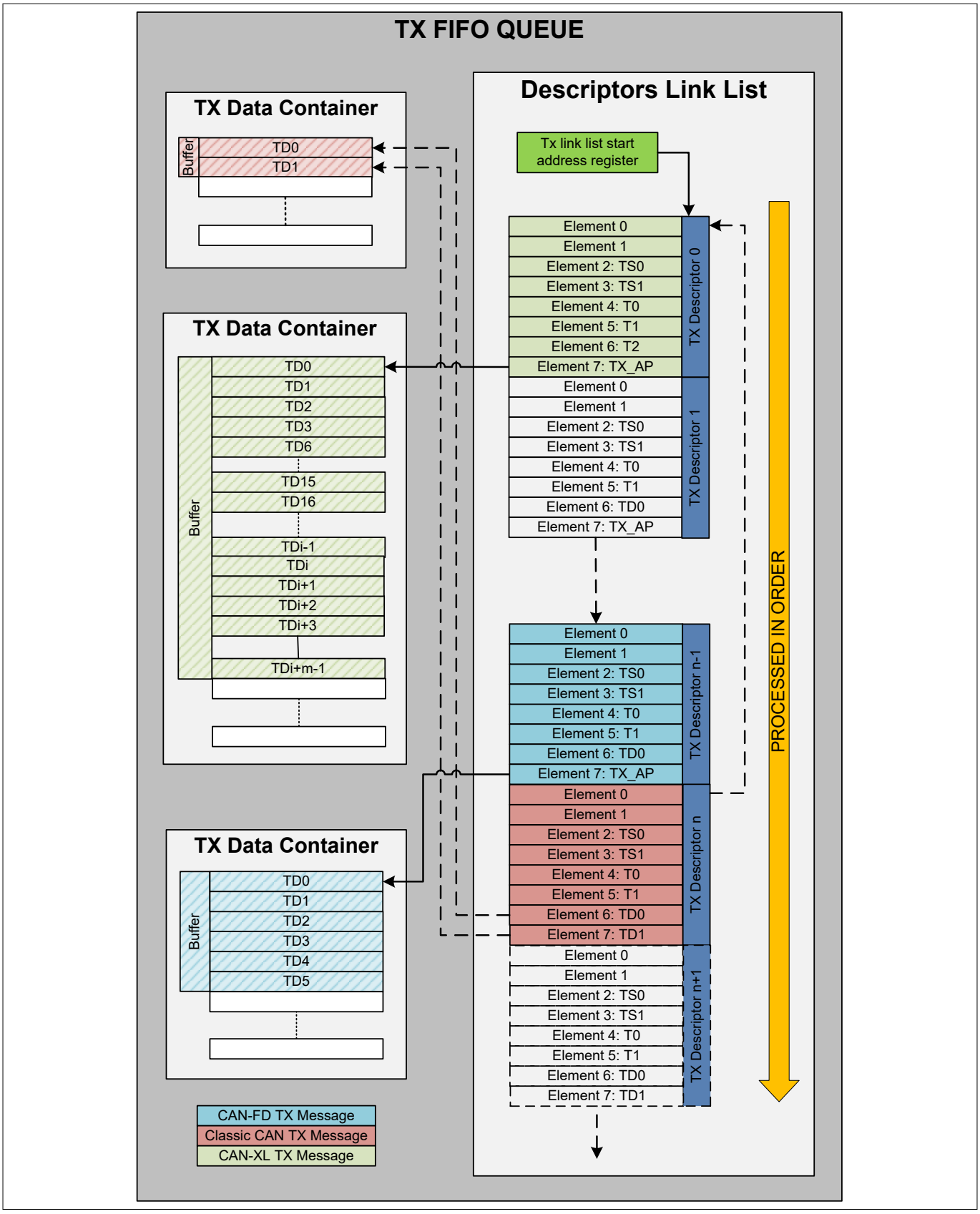


Figure 408 TX FIFO Queue (Basic Mode)

This approach does provide less constraints on system as only one TX descriptor needs to be fetched per TX message. It would be much more efficient in term of performances and memory allocation regarding linked list descriptors.

The only constraint for such configuration would be the memory space allocated to the payload data. As the CAN XL can support up to 2048 bytes, the size of the data container to hold the complete payload could be quite large.

22.5.3.14 TX priority queue

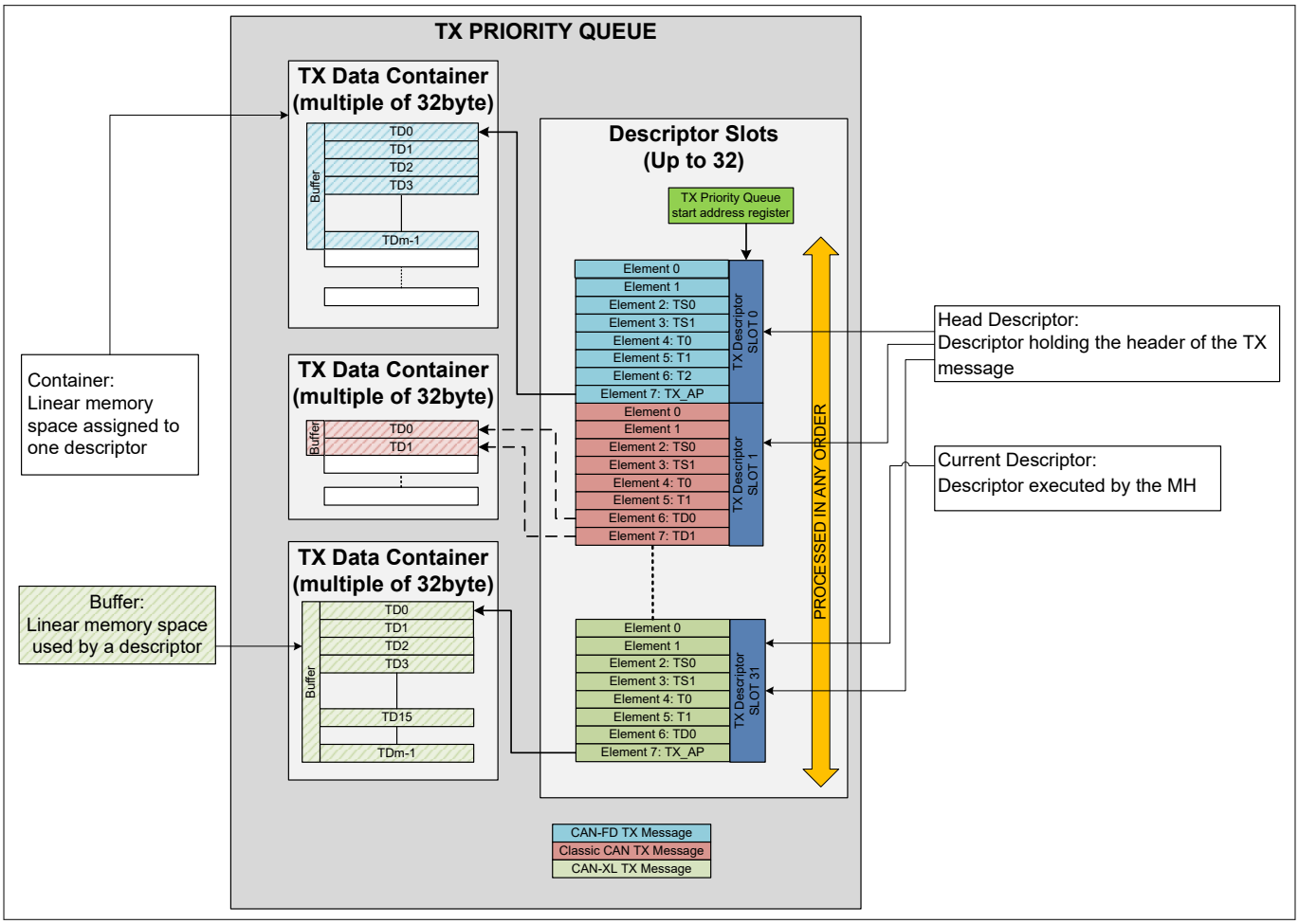


Figure 409 TX priority queue description

This kind of queue does not behave as the TX FIFO Queue but the way messages are defined and how the MH is reading the descriptor are identical.

A TX Priority Queue can be configured with a maximum of 32 slots.

When the SW wants to configure N TX Priority Queue slots, only the slot number from 0 to N-1 can be used.

Every slot is assigned one TX message from a SW point of view. Every slot can be enabled/disabled individually leaving the option to define any number of active slot or none in the SW. Compared to the TX FIFO Queue, there is no order of execution. Any message defined in the TX Priority queue can be selected and executed in any order, only the highest priority message is selected first. Those messages are evaluated against the one currently in use in all TX FIFO Queues.

The same principle is used to define a TX message, meaning some TX descriptors and TX data buffers to define a message. Like the TX FIFO Queues, data buffers hold the payload data of the TX message while descriptor defines header information. In some cases, the first payload data may also be part of the descriptor.

To give a status report and some information like timestamping, the MH writes back some bit field in the TX Header Descriptor. This way the SW is able to track back TX messages sent and identify those with issues.

The TX Priority Queue is using the same data path as the one defined and implemented for the TX FIFO Queue.

The main difference between the two queues is:

- The Priority Queue is managed in any order. As soon as a slot over the 32 is available, a new message can be defined
- The SW needs to trigger the MH to consider a new message in a slot
- The SW needs to read status register to identify which message has been sent

Prior to launch any TX Priority Queue slot, the MH must be started (Ni_MH_CTRL.START written to 1 will drive the Ni_MH_STS.BUSY bit status to 1). To start the TX Priority Queue slot n, write 1 to the Ni_TX_PQ_CTRL0.START[n]. Before launching a TX Priority Queue slot n, it must be enabled by setting the Ni_TX_PQ_CTRL2.ENABLE[n] bit to 1. Once enabled and started, there is no way to disable it while running without a defined procedure. Instead, the abort bit Ni_TX_PQ_CTRL1.ABORT[n] provides a way to stop a TX Priority Queue slot n running and to ensure a safe stop and flush of ongoing data. For more detail on starting and stopping TX Priority Queue slots, refer to the Application Information chapter.

To ensure that no dead lock can occur at start, the ENABLE signal from the PRT must be high to allow any TX Priority Queue to start. This signal status can be monitored in the Ni_MH_STS.ENABLE bit field.

As soon as one or several slots of the TX Priority Queue are started, the MH will fetch the relevant TX descriptors defined at those locations and stores them in L_MEM. When the TX descriptors are available in the L_MEM, they will be part of the arbitration process. As long as the TX messages defined by those TX descriptors are not sent to the CAN bus, they will remain for all the arbitration runs. It is important to note the TX FIFO Queue's messages are also part of this arbitration process.

When the TX message is sent successfully, the TX Priority Queue slot is released and set inactive. A TX_PQ_IRQ interrupt can be triggered to the SW when the last data of the TX message is sent. The other option would be to poll the corresponding status bit register, to identify when the transfer has completed. This last approach requires much more CPU time compare to the interrupt one.

As the TX message to select is based on an arbitration process, the TX Priority Queue execution will run at a different rate compared to the TX FIFO Queues. If TX messages are defined into the TX Priority and have highest priority they will go before TX FIFO Queues. The SW can add new messages at any time when a slot is available.

If for some reason a TX Priority Queue slot n needs to be stopped, it is still possible to abort the execution of that slot. When such action is performed, the TX Priority Queue slot n will be considered as no more active. If the TX message assigned to this slot is already in progress to the CAN bus or has been selected as the next message to be sent, it will not be canceled. By using a register status, it is possible to identify if the slot aborted has been done before or after the sending of the TX message.

Any safety issue related to a TX descriptor executed by a TX Priority Queue slot is declared as no more valid. This means, it will not be part of the arbitration process with the other slots but will not prevent the others to be executed. To identify such issue a TX_DESC_CRC_ERR is sent to the system. Despite this TX Priority Queue slot is stopped, the others will keep going with their own TX descriptors.

If some message does not go through for the two following reasons, maximum number of restart reached or invalid header format, the message is discarded. The error status being detected is written back to the TX descriptor holding the header data.

There is a way to keep track of the TX descriptors used for a given TX FIFO Queue, refer to the Trace and Debug chapter.

A TX Priority Queue is controlled and monitored using several registers and bit registers:

- The Ni_TX_PQ_START_ADD register to define the start address of the TX Priority Queue
- The Ni_TX_PQ_CTRL0.START[n] (where n is 0-31) bit register to launch the TX Priority Queue slot n
- The Ni_TX_PQ_CTRL1.ABORT[n] (where n is 0-31) bit register to abort the execution of the TX Priority Queue slot n

- The Ni_TX_PQ_CTRL2.ENABLE[n] (where n is 0-31) bit register to enable the TX Priority Queue slot n prior to use it
- The Ni_TX_DESC_ADD_PT register to monitor the current address pointer
- The Ni_TX_PQ_STS0.BUSY[n] (where n is 0-31) bit register to know the status of the TX Priority Queue slot n, either busy (TX Priority Queue Slot is having a TX message to send) or not busy (Slot is no more active for reasons like message sent, safety issue, ...)
- The Ni_TX_PQ_STS1.SENT[n] (where n is 0-31) bit register to know the status of the TX message assigned to the TX Priority Queue slot n, either sent (TX message assigned to slot n is sent) or not sent (could be for different reasons like safety issue, max retransmission counter reached, ...)
- The Ni_TX_PQ_INT_STS0.SENT[n]/ Ni_TX_PQ_INT_STS1.SENT[n] and Ni_TX_PQ_INT_STS0.UNVALID[n]/ Ni_TX_PQ_INT_STS1.UNVALID[n] (where n is 0-31) bit register to identify respectively, a message is transmitted or an invalid TX descriptor is detected

A TX Priority Queue is being controlled for any issue using common bit registers:

- The Ni_SFTY_INT_STS.TX_DESC_CRC_ERR and Ni_SFTY_INT_STS.TX_DESC_REQ_ERR bit registers to identify respectively, any CRC issue on TX descriptor running in the TX FIFO Queue n and non-expected TX descriptor
- The Ni_ERR_INT_STS.DP_TX_ACK_DO_ERR bit register to identify overflow on TX ACK data path for the TX FIFO Queues
- The Ni_ERR_INT_STS.DP_TX_SEQ_ERR bit register to identify if an issue occurs on the TX_MSG interface

22.5.3.15 RX FIFO queue in normal mode

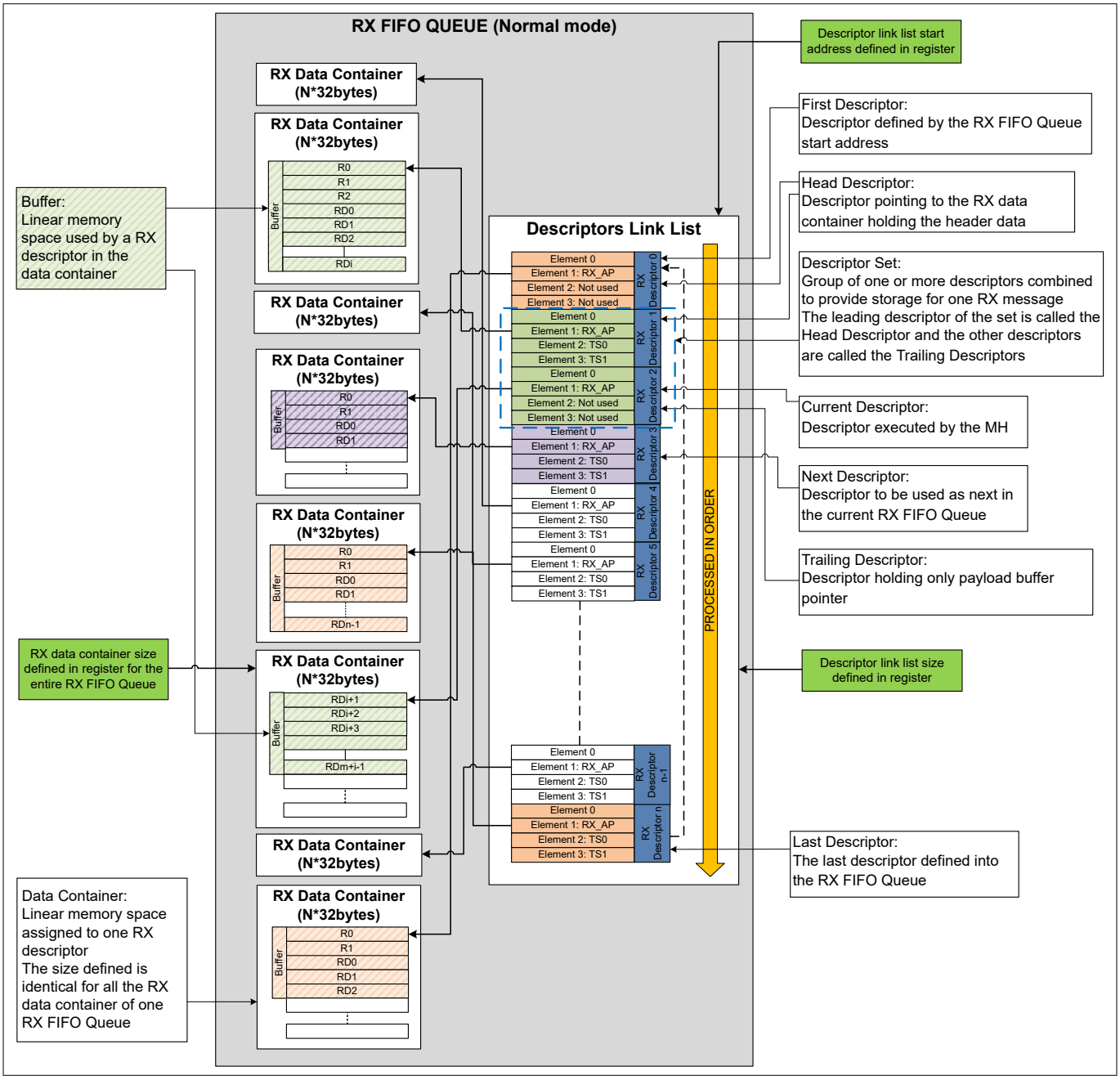


Figure 410 RX FIFO queue description

Up to 8 RX FIFO Queue can be defined and managed by the MH. An RX FIFO Queue is a list of RX descriptors pointing to RX data container to store the RX messages received by the PRT.

The RX filtering rules, programmed by the SW, define if a message is rejected or accepted and in case it is which RX FIFO Queue is receiving the message. If a message is rejected it will not appear in any of the FIFOs. Each one being fully independent from the others, the MH appends new RX message as they arrive on the CAN Bus.

The mechanism to manage RX FIFO Queues is based on the concept of link list. Any RX FIFO Queue is defined using a link list of RX descriptors and RX data container. Those containers are used to write the RX message data to the S_MEM and have fix size over the entire RX FIFO Queue. A different size could be defined per RX FIFO Queue but must always be a multiple of 32 byte.

The size of the data container is programmable to store small or large RX message payload data, if required. Up to Ni_RX_FQ_SIZE{n}.DC_SIZE[6:0] * 32byte data container size can be defined per RX descriptor in an RX FIFO Queue n. As the size is programmable per RX FIFO Queue, it is then possible to limit the memory footprint according to the expected message to be received.

A link list is made of descriptors, where a descriptor is defined by several data elements of the same size, the element is 32 bit word. Each element provides some information or would define some actions to perform. A descriptor is built by the SW but will be read and executed by the MH. Every descriptor is of the same size, pointing to a data container and also to the next descriptor. The link between descriptor is just a fix offset, as they are continuous in memory (to ease and simplify implementation). Therefore, it is not required to declare or use a bit field to mention the position of the next descriptor as it is implicit (dashed lines). As data containers have a fix size and the RX message received may change in size, several descriptors could be required.

As messages are received in a continuous way, the RX FIFO Queue are used in a circular buffer mode. This means when the Last Descriptor is reach the MH will consider the First Descriptor as the next descriptor. The Last Descriptor is defined by the size of the RX FIFO Queue and the start address of the RX FIFO Queue.

An RX filter in MH is used to accept or reject RX messages. If a message is accepted, it is then sent to a defined RX FIFO Queue. The RX filter builds over time those queues with messages based on the filtering result. It is up to the SW to read them in time.

The RX filter observes all the incoming RX messages to identify the right RX FIFO Queue. Once defined, the first RX descriptor attached to the selected RX FIFO Queue is fetched and used to write the incoming data to the S_MEM. As soon as the incoming RX data is increasing above the limit of the data buffer pointed by the current RX descriptor a new one is fetched to keep going. This process repeats up to last RX data received.

The MH will proceed with all the RX FIFO Queues the same way. As the RX FIFO Queue selected depends on the RX filtering result, the RX FIFO Queues will be fill up at a different rate.

Prior to launching any RX FIFO Queue, the MH must be started (Ni_MH_CTRL.START written to 1 will drive the Ni_MH_STS.BUSY bit status to 1). To start the RX FIFO Queue n, write 1 to the Ni_RX_FQ_CTRL0.START[n]. Before launching an RX FIFO Queue n, it must be enabled by setting the Ni_RX_FQ_CTRL2.ENABLE[n] bit to 1. Once enabled and started, there is no way to disable it while running without a defined procedure. Instead, the abort bit Ni_RX_FQ_CTRL1.ABORT[n] provides a way to stop an RX FIFO Queue n running and to ensure a safe stop and flush of on going data. For more detail on starting and stopping RX FIFO Queues, refer to the Application Information chapter. It is essential to configure and start the relevant RX FIFO Queues before starting the PRT. When the MH is not started and so no RX FIFO Queues are started, the MH will not accept any RX data, leading to a PRT data overflow.

Each RX FIFO Queue can be managed individually, SW can decide to enable or disable any queue according to the way RX messages must be managed. Once the RX filter is defined and the PRT is receiving messages any change on the RX FIFO Queue setting is not possible. There is still a mechanism to abort an flush an RX FIFO Queue while others are running.

Once a link list is started and an RX message needs to be written inside, the first descriptor in the list is read. It is executed and the data buffer assigned to it, written. Other actions could be defined into the element data like triggering an interrupt or setting flags. The link list is processed one descriptor at a time, if more RX descriptor is required for a given message, the next one into the list is fetched (or could have been fetched earlier) and the process repeats itself. The process keeps going up to the last descriptor of the link list, a wrap will occur automatically at this time.

If the size of the container is small, the RX message with few payload data may fit in but larger one would require several descriptors and containers. This approach optimizes the memory usage as the number of containers used is very close to the effective size of the RX message received. But on the other way, such strategy requires more descriptors and Data Container.

If we consider the other way round, large data container avoids splitting data buffers and limits the number of descriptors. The main disadvantage would be the usage of more memory per descriptor.

This is up to the SW to find the best tradeoff according to the CAN protocol and to the application required. There is also the option to size differently the data container for every RX FIFO Queue, leaving some flexibility of optimization.

Before receiving any RX message the RX FIFO Queues must be started. In case some messages are received and the RX FIFO Queue to write data is not active the RX message is rejected and an RX_ABORT_IRQ interrupt is triggered to the system.

To give a status report and some information like timestamping, the MH is also able to write back some elements in the RX or TX descriptors. Not all of them are written back but only the one having the header data defined.

The same remark regarding TX descriptors and TX data buffer location into memory applies for the RX descriptors and data buffers.

The SW must always ensure that some RX descriptors in the RX FIFO Queue are always valid (VALID bit set to 0). In case an RX descriptor is not valid the RX FIFO Queue n is stopped and an interrupt RX_FQ_IRQ is sent to the system. If the system provides a valid RX descriptor and restarts the RX FIFO Queue n in time the RX message could may be written into memory, otherwise the message is rejected and the interrupt RX_FQ_IRQ is triggered to the system.

Up to 1023 RX descriptors can be defined for an RX FIFO Queue. The size of the RX FIFO Queue is defined such a way when the Last Descriptor is reached, the MH wraps automatically to its initial start address to get the First Descriptor.

If for some reasons an RX FIFO Queue has an error, it is still possible to abort the execution of that FIFO Queue. When such action is performed, the RX FIFO Queue will be considered as active as long as the current data transfer assigned to an RX descriptor is not finished. This means no more fetches are done and the RX FIFO Queue is set inactive.

Any issue related to an RX descriptor executed by an RX FIFO Queue will stop it right away. To identify such issue some interrupts are triggered to the system, RX_DESC_CRC_ERR or RX_DESC_REQ_ERR. Despite this RX FIFO Queue is stopped, the others will keep going through their own RX descriptors.

An RX FIFO Queue is controlled and monitored using several registers and bit registers:

- The Ni_RX_FQ_START_ADD{n} (where n is 0-7}) bit register to define the start address of the RX FIFO Queue n
- The Ni_RX_FQ_CTRL0.START[n] (n € {0, 1, 2, …, 7}) bit register to launch the RX FIFO Queue n
- The Ni_RX_FQ_CTRL1.ABORT[n] (where n is 0-7}) bit register to abort the execution of the RX FIFO Queue n
- The Ni_RX_FQ_CTRL2.ENABLE[n] (where n is 0-7}) bit register to enable the RX FIFO Queue n prior to use it
- The Ni_RX_FQ_SIZE{n}.MAX_DESC and Ni_RX_FQ_SIZE{n}.DC_SIZE (where n is 0-7}) bit register to define respectively, the maximum number of RX descriptor before looping back to the initial start address and the Data Container size for the RX FIFO Queue n
- The Ni_RX_FQ_ADD_PT{n} (where n is 0-7}) register to monitor the current address pointer of the RX FIFO Queue n
- The Ni_RX_FQ_STS0.BUSY[n] and Ni_RX_FQ_STS0.STOP[n] (where n is 0-7}) bit registers to know respectively, the status of the RX FIFO Queue n, busy (RX FIFO Queue is active) and stopped or running or not started
- The Ni_RX_FQ_STS1.ERROR[n] and Ni_RX_FQ_STS1.UNVALID[n] (where n is 0-7}) bit registers to identify the root cause of the RX FIFO Queue being stopped, an error is detected or a RX descriptor is invalid
- The Ni_RX_FQ_INT_STS.RECEIVED[n] and Ni_RX_FQ_INT_STS.UNVALID[n] (where n is 0-7}) bit registers to identify respectively, a message is received and an invalid RX descriptor is detected

An RX FIFO Queue is being controlled for any issue using common bit registers:

- The Ni_SFTY_INT_STS.RX_DESC_CRC_ERR and Ni_SFTY_INT_STS.RX_DESC_REQ_ERR bit registers to identify respectively, any CRC issue on RX descriptor running in the RX FIFO Queue n and non-expected RX descriptor

- The Ni_ERR_INT_STS.DP_RX_ACK_DO_ERR bit register to identify overflow on RX ACK data path for the TX FIFO Queues
- The Ni_ERR_INT_STS.DP_RX_FIFO_DO_ERR bit register to identify overflow on RX DMA FIFO for the RX FIFO Queues
- The Ni_ERR_INT_STS.DP_RX_SEQ_ERR bit register to identify if an issue occurs on the RX_MSG interface

22.5.3.15.1 Fragmented data container

The RX Data Container can be defined into any location and so an RX message is split across several area in the S_MEM. With such approach, an address pointer is given to the application for any RX message data, no copy is performed. A new Data Container is then allocated to replace the one being sent to the application. It is important to note that in case of RX message received into several Data Container, several address pointers will need to be provided. It is assumed that Data Containers that belongs to the same message can only be released once all RX buffer data have been read.

When the MH has executed the Last Descriptor (the descriptor defined at the latest position in the Descriptor Link List), it wraps automatically to the First descriptor.

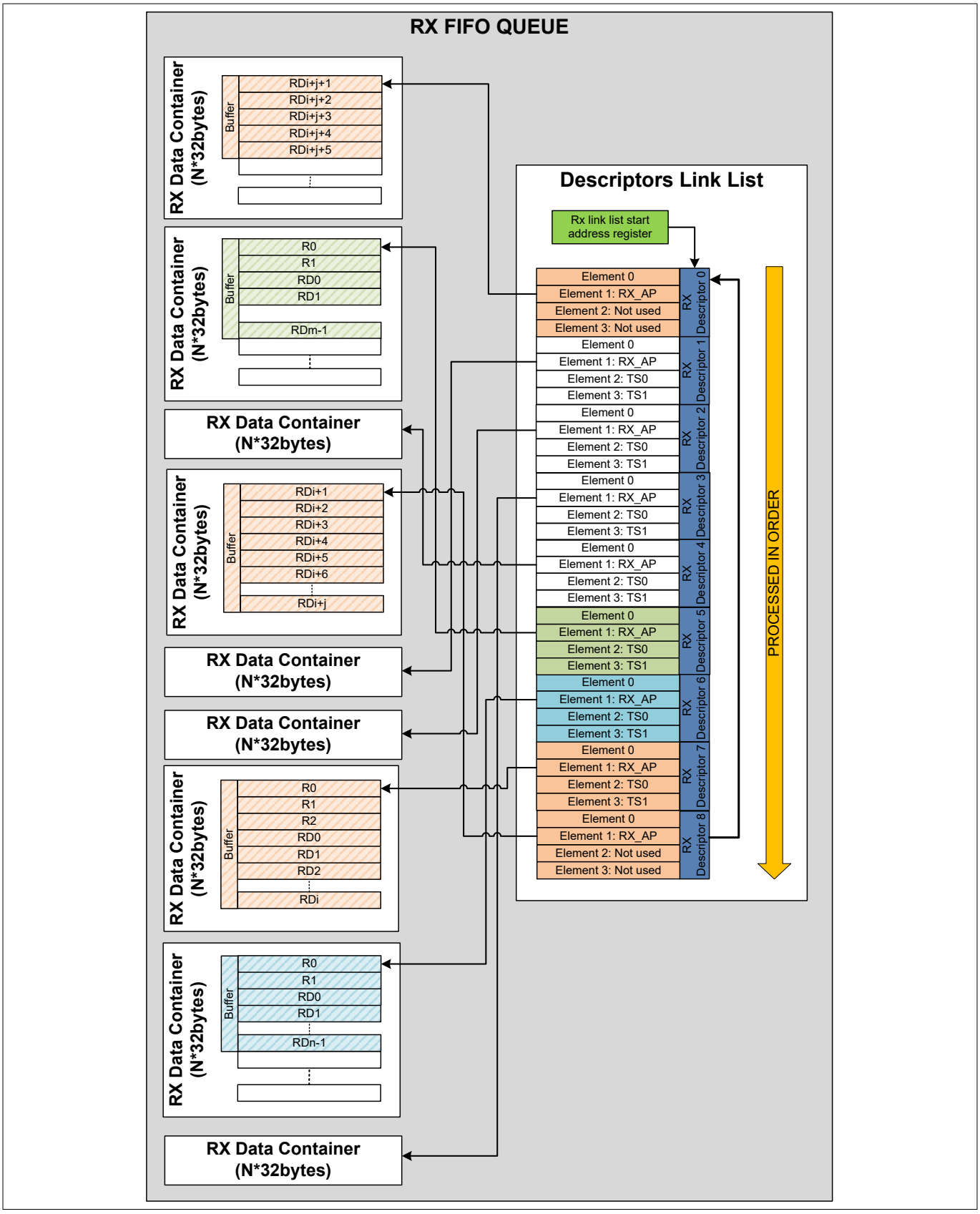


Figure 411 RX FIFO Queue in Normal mode (Fragmented data containers)

In some cases the RX message can then be split across RX descriptor being at the top and at the bottom of an RX FIFO Queue. This mode does make use of all the RX descriptors defined in a given RX FIFO Queue. It may

happen that SW would prefer to rely on linear RX message, having in mind a linear organization of data containers in S_MEM, see Continuous data container chapter.

22.5.3.15.2 Continuous data container

As the RX data containers of the same message are spread to different location in L_MEM, it will not be easy for the SW to read the entire message. To get around this issue, the SW can decide to declare the RX Data Container in a linear memory area and to have them continuous to each other as depicted below.

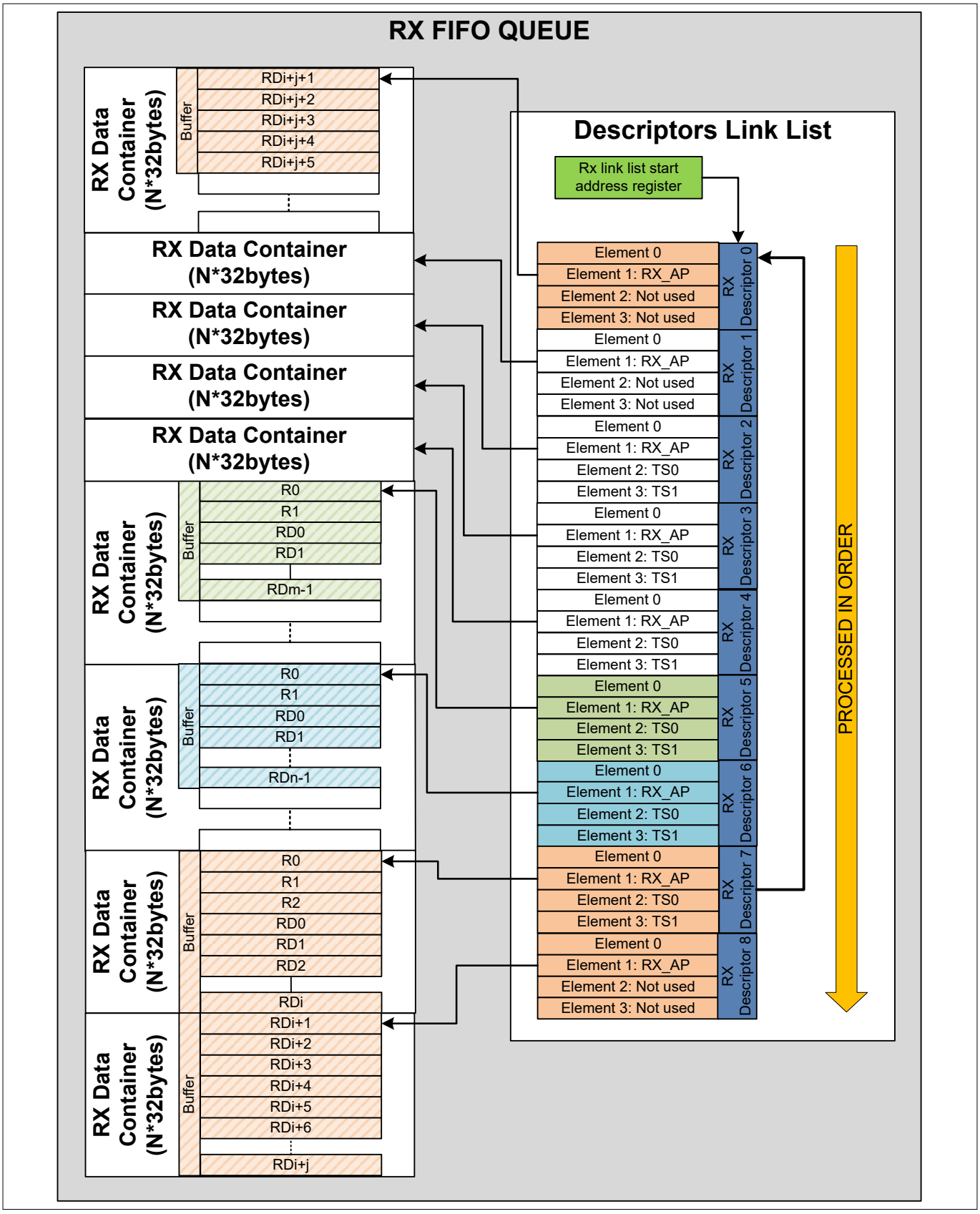


Figure 412 RX FIFO Queue in Normal Mode (Continuous Data Container)

This way of managing the RX message will have the main advantage to provide an RX message written in a linear memory area, despite being split in several data containers.

The only exception would be when the Last Descriptor is used and the message size exceed the Data Container. The MH wraps and uses the First Descriptor to keep going with the current RX message data. In this particular scenario the RX message data is split over the top and the bottom of the data container and the same applies for the link list holding the RX descriptor. This is normal behavior and it must not be an issue for the SW to read the RX message following the RX descriptor list from bottom to top.

This configuration provides a pseudo linearity for the RX messages in S_MEM, excepted at the borders. Doing so, the SW would need to perform a copy of the RX message data to free the memory area for the new incoming messages. Such configuration does not require any update on address pointer in the RX descriptors. Only the VALID bit needs to be written by the SW to acknowledge the reading of the RX message and the update of the read address pointer register.

22.5.3.16 RX FIFO queue in continuous mode

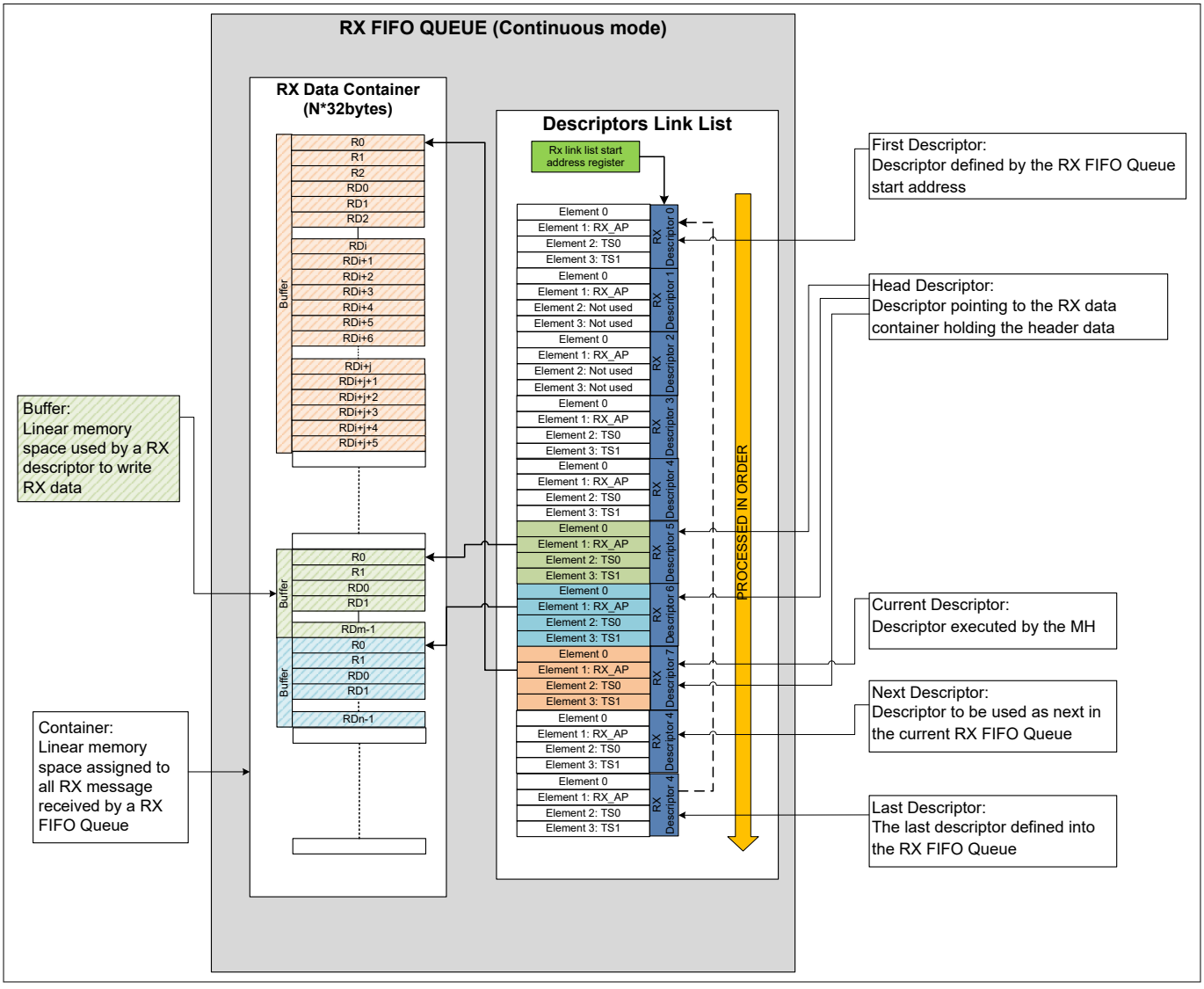


Figure 413 RX FIFO queue continuous mode

The same principle as defined in the RX FIFO Queue in Normal mode applies for the RX descriptors in this Continuous mode. The way they are used, managed and defined remain, see RX FIFO Queue in Normal mode chapter.

The main difference is coming from the structure of the RX message data being stored in the S_MEM. An RX FIFO Queue when in Continuous mode is a list of RX descriptors pointing to a large and single data container to store all the RX messages received by the PRT.

The RX filtering rules, programmed by the SW, define if a message is rejected or accepted, and in case it is which RX FIFO Queue is receiving the message. If a message is rejected it will not appear in any of the FIFOs. Each one being fully independent from the others, the MH appends to RX FIFO Queues new RX message as they arrive on the CAN Bus. The RX filter builds over time those queues with messages based on the filtering result. It is up to the SW to read them in time.

The mechanism to manage RX FIFO Queues is based on the concept of link list. Any RX FIFO Queue when in Continuous mode is defined using a link list of RX descriptors and a large data container.

As messages are received in a continuous way, the RX FIFO Queue are used in a circular buffer mode. This means when the Last Descriptor is reach the MH will consider the First Descriptor as the next descriptor. The Last Descriptor is defined by the size of the RX FIFO Queue and the start address of the RX FIFO Queue.

The RX filter observes all the incoming RX messages to identify the right RX FIFO Queue. Once defined, the current RX descriptor attached to the selected RX FIFO Queue is fetched and used to define the new incoming RX message data.

The MH will proceed with all the RX FIFO Queues the same way. As the RX FIFO Queue selected depends on the RX filtering result, the RX FIFO Queues will be fill up at a different rate.

Every RX FIFO Queue can be managed individually, SW can decide to enable or disable any queue according to the way RX messages must be managed. Once the RX filter is defined and the PRT is receiving messages any change on the RX FIFO Queue setting is not possible. There is still a mechanism to abort and flush an RX FIFO Queue while others are running.

Once an RX FIFO Queue is started and an RX message needs to be written in its corresponding data container, the First descriptor in the descriptor list is read. It is executed and the initial start address of the data container assigned to it. The link list is processed one descriptor at a time every time a new RX message is received by the same RX FIFO Queue. The process keeps going up to the Last descriptor of the link list before making a wrap. Other actions could be defined in the RX descriptor like triggering an interrupt or setting flags.

The MH writes messages as they arrive, to avoid overwritting. The SW needs to write the address value of the current message being read to a MH register. Therefore, the MH can compute the exact memory left to be used by the new RX message.

The size of the data container is programmable to store many CAN XL frames if required. Up to Ni_RX_FQ_SIZE{n}.DC_SIZE[11:0] * 32byte data container size can be defined for a data container assigned to an RX FIFO Queue. As the size is programmable per RX FIFO Queue, it is then possible to limit the memory footprint according to the expected message to be received.

Before receiving any RX message the RX FIFO Queues must be started. In case some messages are received and the RX FIFO Queue to write data is not active the RX message is rejected and an RX_ABORT_IRQ interrupt is triggered to the system.

To give a status report and some information like timestamping, the MH is also able to write back some elements in the RX or TX descriptors. Not all of them are written back but only the one having the header data defined.

The same remark regarding TX descriptors and TX data buffer location into memory applies for the RX descriptors and data buffers.

The SW must always ensure that some RX descriptors in the RX FIFO Queue are always valid (VALID bit set to 0). In case an RX descriptor is not valid the RX FIFO Queue n is stopped and an interrupt RX_FQ_IRQ is sent to the system. If the system provides a valid RX descriptor and restarts the RX FIFO Queue n in time the RX message could may be written into memory, otherwise the message is rejected and the interrupt RX_FQ_IRQ is triggered to the system.

In the Continuous mode, one RX descriptor is assigned to one message, thus, once the SW has read the message, the VALID bit can be set to 0 right away. The SW is in charge, once the message is read, to set the read

address pointer in the MH register. The MH uses this information to estimate if the incoming message data can be written safely in the data container.

Up to 1023 RX descriptors can be defined for an RX FIFO Queue. The size of the RX FIFO Queue is defined such that when the Last Descriptor is reached, the MH wraps to its initial start address to fetch the First Descriptor. As the RX FIFO Queue size for the RX descriptors is defined at first time and cannot be changed once the RX FIFO Queue is started, the MH wraps automatically to keep going.

If, for some reasons, an RX FIFO Queue has an error, it is still possible to abort the execution of that FIFO Queue. When such action is performed, the RX FIFO Queue will be considered as active, as long as the current data transfers, assigned to an RX descriptor, are not finished. This means, there is no pending transaction attached to the RX FIFO Queue, this includes the RX descriptor acknowledge when an RX message is received. The RX_FQ_IRQ interrupt is triggered to the system, if enable, once the last RX message is received by the aborted RX FIFO Queue n.

Any issue related to an RX descriptor that is executed by an RX FIFO Queue will stop it right away. To identify such issue, some interrupts are triggered to the system, RX_DESC_CRC_ERR or RX_DESC_REQ_ERR. Despite of having this RX FIFO Queue stopped, the other ones will keep going through their own RX descriptors.

An RX FIFO Queue is controlled and monitored using several registers and bit registers:

- See the all registers already defined for the Normal mode
- The Ni_RX_FQ_DC_START_ADDn (where n is 0-7) register to be written by the software to indicate to the MH the read address pointer in the data container for the RX FIFO Queue n
- The Ni_RX_FQ_STS2.DC_FULL[n] (where n is 0-7) bit register to identify the root cause of the RX FIFO Queue being stopped, there is no space left on the system memory to write new RX data. This issue may occur only if the MH is set to Continuous Mode
- In case of Continuous mode, the Ni_RX_FQ_RD_ADD_PTn (n € {0, 1, ..., 7}) register must be initialized to {Ni_RX_FQ_RD_ADD_PTn.VAL | 0b11} otherwise left to its default value

An RX FIFO Queue is being controlled for any issue using common bit registers:

- See the all registers already defined for the Normal mode

22.5.3.17 TX FIFO queue data flow

The SW defines the TX descriptors for every TX FIFO Queues to be used and declare the TX data buffers assigned to those TX descriptors.

As soon as the TX FIFO Queues are started the TX MH will process and fetch all the relevant TX descriptors and will store them into the L_MEM for arbitration.

Only the TX message having the highest priority ID is sent first. Those messages will compete against the one defined into the TX Priority Queue. Only the TX descriptor holding the header data is written back with status information of the data transfer and timestamp. As soon as one TX descriptor is used from a TX FIFO Queue it will be replaced by the next one of that queue. The TX MESSAGE HANDLER is managing on its own the request for a new descriptor whenever required.

The following data flow describes how the TX FIFO Queues are running in parallel. Here are the different steps when a TX message is selected and/or used:

1. After having sent a TX message from the TX FIFO queue n, the TX MESSAGE HANDLER is sending a request to the DESCRIPTOR MESSAGE HANDLER for the next TX descriptor from that queue
2. The relevant TX descriptor of the TX FIFO Queue is fetched by the DESCRIPTOR MESSAGE HANDLER and written to the L_MEM
3. As soon as the new TX descriptor is completely written into the L_MEM an arbitration run is performed. This arbitration will identify which TX descriptor is having the highest priority, looping through the current TX descriptor for every TX FIFO Queues and through all the slot of the TX Priority Queue declared as active. Once the two first candidates position (Priority Queue slot number or TX FIFO queue number) are defined, they are loaded in the TX MESSAGE HANDLER

4. The TX MESSAGE HANDLER is then trying to upload the TX message with the highest priority locally. If a TX message is in progress, the TX MESSAGE HANDLER will wait for the end of the current transmission to read from the L_MEM the complete TX descriptor. If nothing is preventing the upload of the next descriptor, it will be done right away. As soon as the TX descriptor is locally stored the first TX message data are sent to the PRT. The TX MESSAGE HANDLER will wait for the PRT to know if it has won the arbitration process. As long as no new TX descriptor is changing the arbitration result, the selected TX descriptor remains in the TX MESSAGE HANDLER for further arbitration trials. As soon as the TX message is winning the arbitration, all the data contained in the TX descriptor is sent to the PRT
5. The payload data assigned to the TX descriptor is fetched from the S_MEM.
6. If the TX message is sent successfully, an acknowledge request is sent to the DESCRIPTOR MESSAGE HANDLER with the status and information of the transfer. The DESCRIPTOR MESSAGE HANDLER writes back the acknowledge of that descriptor in the S_MEM. When the DESCRIPTOR MESSAGE HANDLER has finished writing the TX descriptor an interrupt TX_FQ_IRQ for the TX FIFO Queue n may be triggered to the system

As all TX FIFO Queue are processed the same way, the data flow of only one TX FIFO Queue is depicted in the following figure with the reference number for each step.

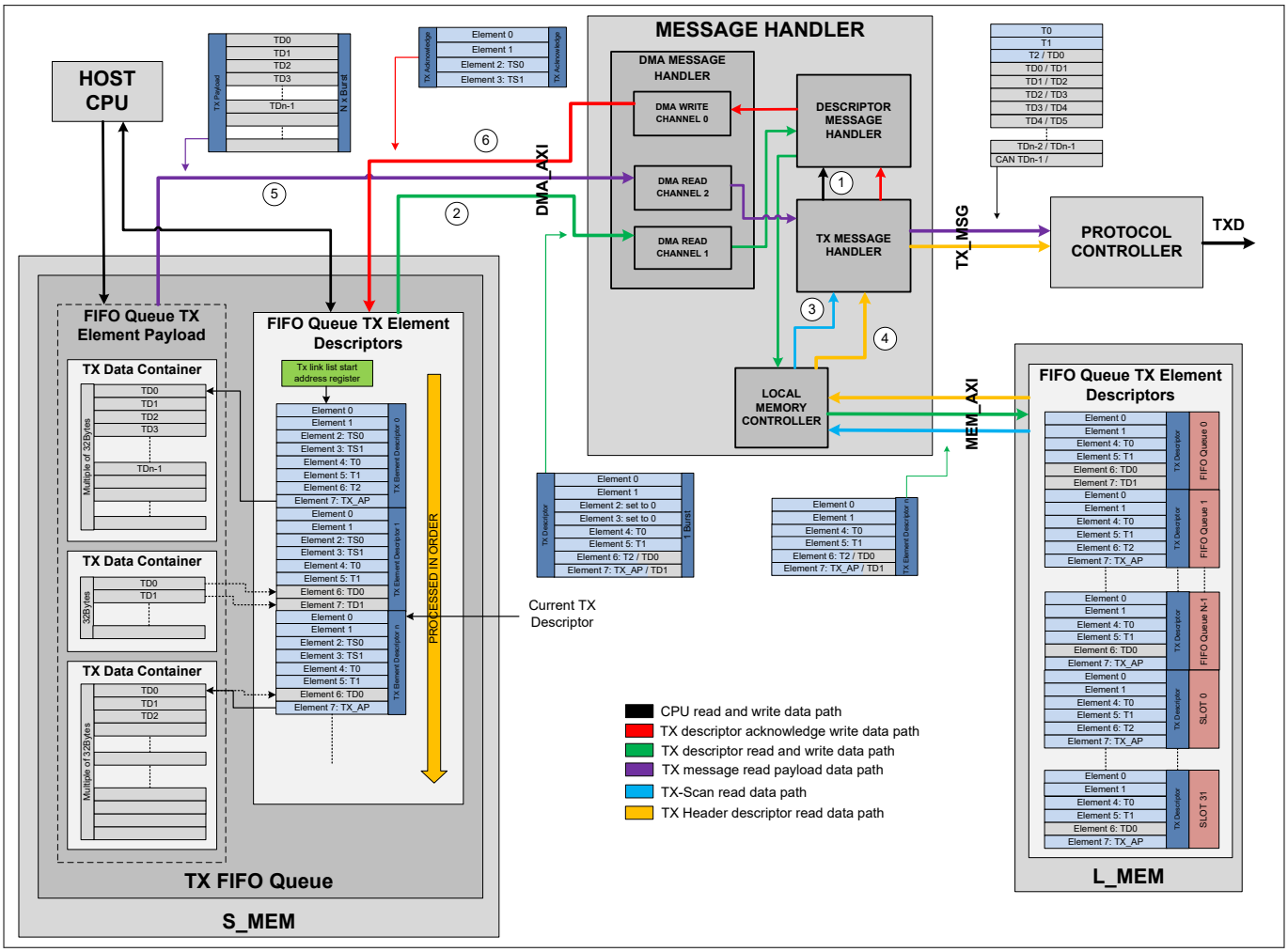


Figure 414 TX FIFO queue data flow

22.5.3.18 TX priority queue data flow

The SW defines the TX descriptors to be sent into the TX Priority Queue slot to be used and declare the TX data buffers assigned to those TX descriptors. Once done the SW triggers the TX MESSAGE HANDLER to have those

messages sent as soon as possible. Those messages will compete against the one defined into the TX FIFO Queues. Only the ID is relevant for the selection of TX messages.

As soon as the TX Priority Queue slots are started the TX MESSAGE HANDLER will process and fetch all the relevant TX descriptors and will store them into the L_MEM for arbitration.

When a TX message is sent, an acknowledge (status information and timestamp) is written back to the TX descriptor holding the header data. As soon as a TX descriptor slot is used from a TX Priority Queue slot, it will be considered as inactive and won't be considered afterwards.

There is a way to keep track of the TX descriptors used for the TX Priority Queue, refer to the Trace and Debug chapter.

The following data flow is relevant for all TX Priority Queue slots.

Here below are the different steps when a TX message is selected and or used:

1. To trigger the TX message defined into the TX Priority Queue, the SW must write the start bit of the corresponding slot. There is nothing preventing the SW to declared several TX message at the same time and to launch them at once. The TX MESSAGE HANDLER is sending requests to the DESCRIPTOR MESSAGE HANDLER for the TX descriptors to be fetched. If several TX descriptors need to be uploaded at once, they would be fetched in the order of their slot number, starting with 0
2. The relevant TX descriptors of the TX FIFO Queue is fetched by the DESCRIPTOR MESSAGE HANDLER and written to the L_MEM
3. As soon as the new TX descriptor is completely written into the L_MEM an arbitration run is performed and only the TX descriptor uploaded for the slot will be considered. This arbitration will identify which TX descriptor is having the highest priority, looping through the current TX descriptor for every TX FIFO Queue and through all slots of the TX Priority Queue declared as active. This selection is performed doing a single read on all defined TX descriptor in the L_MEM. Once the two first candidates are identified, either the TX Priority Queue Slot number and/or the TX FIFO queue number, they are stored locally in the TX MESSAGE HANDLER
4. The TX MESSAGE HANDLER is then trying to upload the TX descriptor with the highest priority locally. If a TX message is in progress, the TX MESSAGE HANDLER will wait for the end of the current transmission to read from the L_MEM the complete TX descriptor. If nothing is preventing the upload of the next descriptor, it will be done right away. As soon as the TX descriptor is stored locally the first TX message data are sent to the PRT. The TX MESSAGE HANDLER will wait for the PRT to know if it has won the arbitration process. As long as no new TX descriptor is changing the arbitration result, the selected TX descriptor remains into the TX MESSAGE HANDLER for further arbitration trials. As soon as the TX message is winning the arbitration, all the data contained into the TX descriptor is sent to the PRT
5. The payload data assigned to the TX descriptor is fetched from the S_MEM.
6. If the TX message is sent successfully, an acknowledge request is sent to the DESCRIPTOR MESSAGE HANDLER with the status and information of the transfer. The DESCRIPTOR MESSAGE HANDLER writes back the acknowledge of that descriptor in the S_MEM. When the DESCRIPTOR MESSAGE HANDLER has finished writing the TX descriptor an interrupt TX_PQ_IRQ for any of the TX Priority Queue slot may be triggered to the system. Once the acknowledge is written, the slot is considered as invalid and will not be used for the next arbitration run, up to the time the SW set it back to active

As all the TX Priority Queue slots are processed the same way, the data flow of one slot is depicted in the following figure with the reference number for each step.

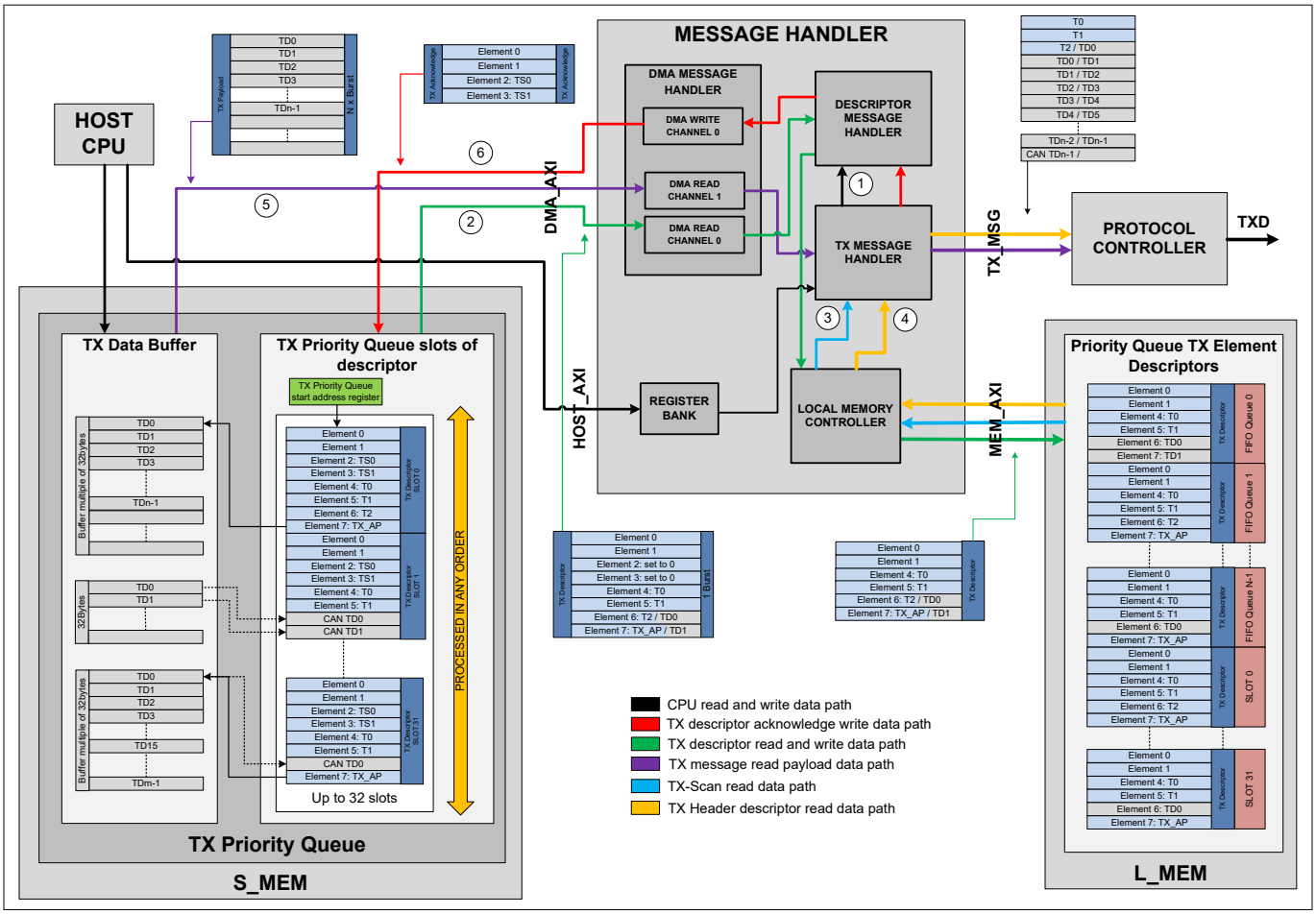


Figure 415 TX Priority queue data flow

22.5.3.19 RX FIFO queue data flow in Normal Mode

The SW needs to prepare into the S_MEM the RX filter elements. Once done the SW writes those elements to the L_MEM. The SW cannot access this memory directly in write mode through the HOST bus interface. As a consequence the L_MEM should provide a way to protect the memory space allocated to the RX filtering elements from being read by any other masters.

The SW defines the RX descriptors for every RX FIFO Queue to be used and allocated the RX data buffers assigned to those RX descriptors. As soon as the RX FIFO Queues are started any RX messages will be filtered, meaning rejected or accepted, and stored into the S_MEM when required.

Here below are the different steps when receiving an RX message:

1. As soon as the RX message data R0, R1 and R2 are received the RX filtering is started. All the incoming data are stored locally into the RX MESSAGE HANDLER waiting for the result of RX filtering. The RX MESSAGE HANDLER identifies RX FIFO Queue to be used
2. The RX MESSAGE HANDLER sends an RX descriptor request to the DESCRIPTOR MESSAGE HANDLER
3. The relevant RX descriptor of the queue identified and fetched by the DESCRIPTOR MESSAGE HANDLER is given to the RX MESSAGE HANDLER
4. The RX MESSAGE HANDLER is using the address pointer of the RX descriptor to write the message data to the S_MEM as soon as a complete burst is available. As long as the data buffer can accept message data, the process of writing can continue. In case the last data can be written into the data buffer pointed by the current RX descriptor, go to 6 otherwise the next RX descriptor of the same queue is requested to the DESCRIPTOR MESSAGE HANDLER

5. When the current RX descriptor is about to be complete, the new RX descriptor should be available for the next DMA data transfer and go to 3
6. The RX MESSAGE HANDLER gets the status of the last part of the RX message and the information of the latest data transfers. Those data are sent to the DESCRIPTOR MESSAGE HANDLER to be written back as acknowledge to RX descriptor into the S_MEM holding the header. The timestamp and report status of the RX message is written at the same time. When the DESCRIPTOR MESSAGE HANDLER has finished writing the RX descriptor an interrupt RX_FQ_IRQ for the RX FIFO Queue n may be triggered to the system

As all the RX FIFO Queues are processed the same way, the data flow of only one RX FIFO Queue is depicted in the following figure with the reference number for each step.

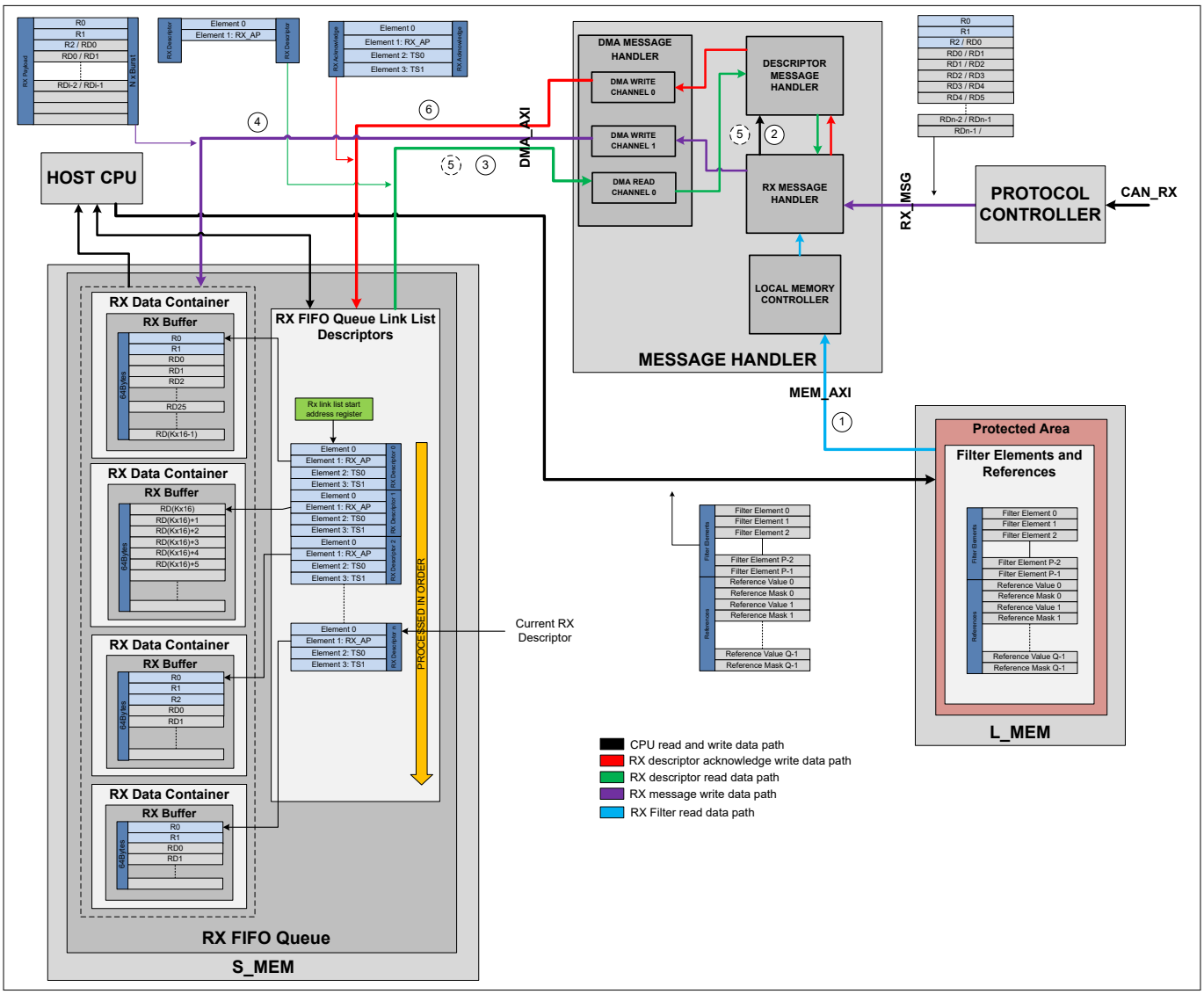


Figure 416 RX FIFO queue data flow in Normal Mode

22.5.3.19.1 RX FIFO Queue data flow in Continuous Mode

The SW needs to prepare the RX filter elements required to accept or reject RX messages. Once done, the SW writes those elements to the L_MEM. The SW cannot access this memory directly in write mode through the HOST bus interface. As a consequence the L_MEM must provide a way to protect the memory space allocated to the RX filtering elements from being read by any other masters.

The SW defines the RX descriptors for every RX FIFO Queue to be used and allocates the single data container for each of them.

As soon as the RX FIFO Queues are started, any RX messages will be filtered, meaning rejected or accepted, and stored into the S_MEM when required.

Here below are the different steps when receiving an RX message:

1. As soon as the RX message data R0, R1 and R2 are received, the RX filtering is started. All the incoming data are stored locally in the RX MESSAGE HANDLER waiting for the result of RX filtering. The RX MESSAGE HANDLER identifies RX FIFO Queue to be used
2. The RX MESSAGE HANDLER sends an RX descriptor request to the DESCRIPTOR MESSAGE HANDLER
3. The relevant RX descriptor of the queue identified and fetched by the DESCRIPTOR MESSAGE HANDLER is given to the RX MESSAGE HANDLER
4. The RX MESSAGE HANDLER holds the RX descriptor of that RX FIFO queue for futher purpose. In case the current RX message cannot fit in the remaining space of the data container, the message is automatically written at the top (if possible). The message data are written to the S_MEM starting after the last RX message stored in the data container. As soon as a complete burst is available, it is written and this process continues up to the last RX message data.
5. The RX MESSAGE HANDLER gets the status of the last part of the RX message and the information of the latest data transfers. Those data are sent to the DESCRIPTOR MESSAGE HANDLER to be written back as an acknowledge to the RX descriptor fetched earlier from the S_MEM. The timestamp, the address of the RX message inside the data container and a report status of the RX message are written at the same time. When the DESCRIPTOR MESSAGE HANDLER has finished writing the RX descriptor, an interrupt RX_FQ_IRQ for the RX FIFO Queue n may be triggered to the system.

As all the RX FIFO Queues are processed the same way, the data flow of only one RX FIFO Queue is depicted in the figure below with the reference number for each step.

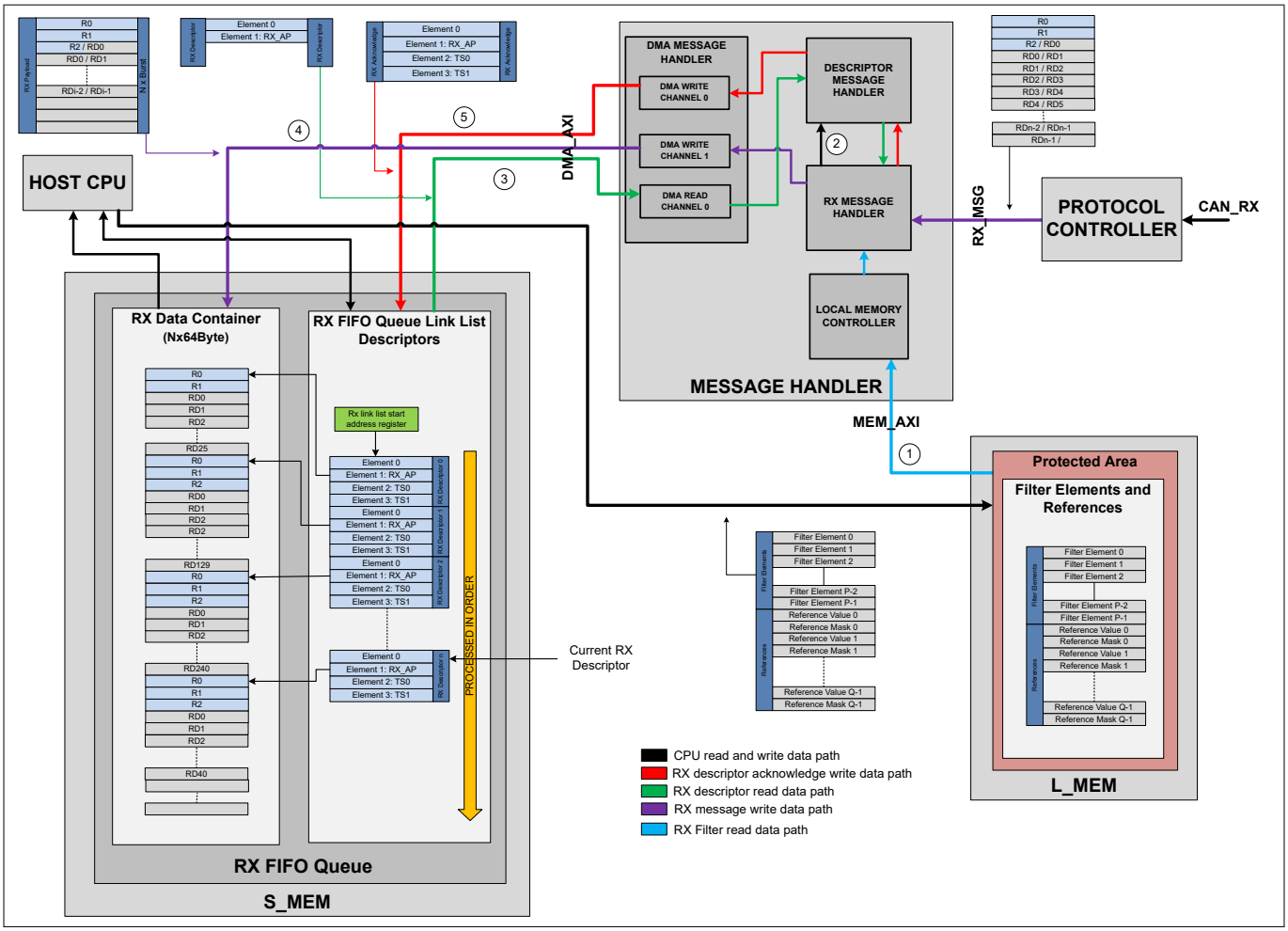


Figure 417 RX FIFO Queue data flow in Continuous Mode

22.5.3.20 TX-SCAN

To avoid any misunderstanding when talking about the selection of the next TX message to be sent to the PRT, the term TX-SCAN is used to define this process.

To arbitrate TX FIFO queues and cope with high latency in S_MEM, the TX Header Descriptor of every active TX FIFO Queues are stored into the L_MEM. The same applies for the TX Priority Queue slots when they are declared as active. It means up to 8 TX FIFO Queues Header Descriptor can be declared in L_MEM and up to 32 for the TX Priority Queue. Doing so, it becomes much easier to parse all the active TX Header Descriptors locally to identify which TX message has the highest priority. The TX-SCAN process would be very fast and the expected TX message order at CAN bus as close as possible to the one expected by the SW.

The TX-SCAN uses the list of TX descriptors available in L_MEM. When a new TX descriptor is added, a flag is set to indicate the availability of a new potential candidate. As long as the TX descriptor is not executed or discarded, it will remain as a valid candidate, see Ni_TX_FQ_DESC_VALID and Ni_TX_PQ_DESC_VALID registers.

Events to trigger a TX-SCAN run are:

- A new TX message written in a TX Priority Queue slot
- A TX message sent successfully
- A TX message discarded after N retransmissions
- A SW abort of a TX Priority Queue slot
- A SW abort of a TX FIFO Queue

- A TX message rejected by the TX Filter
- A TX message starting to be sent (in case of TX FIFO Queue, triggers the fetch of the next descriptor)

Any new TX message appended to a TX FIFO Queue does not trig a new TX-SCAN. The MH is only processing TX descriptor one after the other in every TX FIFO Queue, it does not know if a new message is added to one of them.

A re-transmission counter defines the number of re-transmissions allowed to the same TX message when this one is unsuccessful. For every trial of the same TX message, the re-transmission counter is incremented and compared to a maximum value defined in the Ni_MH_CFG.MAX_RETRANS[2:0]. If the counter exceeds the limit, the current TX message will no longer be considered and is skipped, the next TX message is taken instead. The re-transmission counter is set back to 0 when a new TX message is selected. There is the option to define an unlimited number of trials for TX messages.

The maximum number of re-transmission is defined by the register Ni_MH_CFG.MAX_RETRANS[2:0] and covers the maximum value defined in ISO 11898-1:2024.

Several options are defined:

- 0: No re-transmission
- 1 to 6: 1 to 6 re-transmission
- 7: Unlimited re-transmission (default value)

Here below is the definition of the 32bit priority value when considering Classical CAN, CAN FD and CAN XL. The fields XLF, FDF, XTD, RTR, SRR and ID (defined in CAN protocol [1]) are used to determine the priority value of a given TX message. The priority value is computed for every TX message and then compared with each other to identify the highest priority message (the lowest value gives the highest priority message to transmit).

Only the T0 of the TX Header Descriptor is used for the selection of the highest priority message. As the relevant bits are defined in T0 element, only a single read access from the L_MEM is required. In Classical frame format, a data frame and a remote frame with the same identifier have the same priority in the TX-Scan.

Table 1049 Definition of priority value

CAN Protocol	Protocol Selection			Priority Value						
	XLF	FDF	XTD	31 down to 19	20	19	18	17	16 downto 1	0
Classic CAN	0	0	0	T0[28:18] (Base ID[10:0])	0 (RTR)	0 (XTD)	0 (FDF)	0	16'b0	0
Classic CAN (extended ID)	0	0	1	T0[28:18] (Base ID[10:0])	1 (SRR)	1 (XTD)	T0[17:0] (Identifier Extension[17:0])			0 (RTR)
CAN FD	0	1	0	T0[28:18] (Base ID[10:0])	0 (RRS)	0 (XTD)	1 (FDF)	0	16'b0	0
CAN FD (extended ID)	0	1	1	T0[28:18] (Base ID[10:0])	1 (SRR)	1 (XTD)	T0[17:0] (Identifier Extension[17:0])			0 (RSS)

(table continues...)

Table 1049 (continued) Definition of priority value

CAN Protocol	Protocol Selection			Priority Value						
	XLF	FDF	XTD	31 down to 19	20	19	18	17	16 downto 1	0
CAN XL	1	X	X	T0[28:18] (Priority ID[10:0])	T0[17] (RRS)	0 (XTD)	1 (FDF)	1 (XLF)	16'b0	0

The selection of the TX message is done by looking at the queues in the following order, TX Priority Queue slots from 0 to 31, then the TX FIFO Queues are scanned from 0 to 7. The process of TX message selection will keep the two highest priority messages over the full scan.

When two or more TX messages have the same priority value, the first one will always be kept as the one to be sent first.

***Note**: At initial time, when several TX FIFO Queues are started at the same time, the first TX messages may not be in the right order. Due to the scanning order (TX FIFO Queue slots 0 to 31 and then TX FIFO Queue 0 to 7) and if the system memory latency is high, by the time the last TX descriptor is uploaded to the L_MEM, some TX messages may have been already scanned for the highest priority and sent to the PRT. This is normal behavior and will last only for the first TX messages.*

As soon as a new Header Descriptor is available in the L_MEM, it will be arbitrated automatically if the TX-Scan process is not running. In case that a new TX Header Descriptor is stored in the L_MEM while the TX-Scan is running, the TX-Scan goes up to the end and will restarted to take this new descriptor into account.

Before starting the TX-Scan, the list of all potential candidates (valid) on the L_MEM is stored locally. This process will ensure a proper definition of the best candidates after a complete scan at the time it is done.

The duration of the TX-Scan mainly depends on the access time to the L_MEM and the number of TX FIFO Queues and TX Priority Queue Slots. Here is the list of parameters that will drive the overall time:

- The number of TX FIFO Queues being active at the same time
- The number of TX Priority Queue slots active at the same time
- The L_MEM read latency to fetch one single word

The processing time for one TX -Scan run can be defined as:

$$TX\ Scan\ processing\ time\left(\mu s\right) = Lr * \left(Nbfq + Nbpqs\right) * \frac{1}{CLK(Mhz)}$$ where Nbfq = Number of TX FIFO Queues active, Nbpqs = Number of TX Priority Queue slots active and Lr = read latency from L_MEM defined in number of CLK clock cycles.

In any cases, when a new TX message is scheduled for transmission and it has the highest priority, the maximum delay to have this message selected by the TX-SCAN depends on, the maximum number of TX FIFO Queue and TX priority Queue running concurrently at that time. Considering a maximum of 8 TX FIFO Queues and 32 TX Priority Queue slots running at the same time this leads to (considering the previous formula):

$$Maximum TX\ SCAN\ duration\ \left(\mu s\right) = Lr * \left(\frac{1}{CLK(MHz)}\right) * 40$$

To ensure the continuity of a TX message, it is important to note that regarding TX FIFO Queues, the current and the next TX Header Descriptor for a given FIFO are loaded in the L_MEM. This assumption is valid only if the two TX descriptors are valid. Thus, if several TX messages in the same FIFO have the highest priority over the others, they will be sent back to back. For the TX priority Queue, things are different as one TX message is stored per slot. Only the TX message defined as active in a slot is taken into account at any time.

Two internal buffers are used to hold the TX descriptors in order to send TX messages in a row. One is holding the current TX descriptor to be sent right away to the PRT while the other stores the TX descriptor for the next message. It is important to note that the TX descriptors selected are the result of one TX-Scan. If any new events like, a message sent or a new message to be sent occurs, the two candidates may not be right ones. In this case the TX descriptors already buffered may need to be changed by some new ones. The change is performed step by step, to always have the highest priority message of one TX-Scan run available in one of the two local buffers. The previous TX descriptor with the highest priority is kept in one of the two local buffer while the new highest one is replacing the other. This procedure is repeated, if required, to change the second highest priority message.

The event of any new TX descriptor being loaded and available in the L_MEM triggers a TX-SCAN run. The TX descriptors describing TX messages can only be taken into account by the TX-SCAN if they are available in L_MEM, see Ni_TX_FQ_DESC_VALID and Ni_TX_PQ_DESC_VALID registers.

To prepare the next TX descriptor and to react properly according to some result of the data being sent, there will be several actions to perform:

- The TX-SCAN is computing the two next potential candidates, without considering the TX descriptor set as current in one of the two local buffers (the one with the highest priority). As soon as there are identified, the information related to the source of the two next highest priority messages is stored locally
- The first candidate is compared to the one already in local buffer and having the lowest priority. If the first candidate computed is already in one of the two local buffers, nothing needs to be done. If this is not the case, it is uploaded to provide the next highest priority TX message and will replace the one having the lowest priority in the local buffers. This is mandatory to ensure there is always a valid TX descriptor with high priority to provide to PRT, at any time. This is valid, even if the highest priority TX descriptor in local buffer may, at this time, not be the one with the highest priority. As soon as the first candidate is loaded in the local buffer, it may become the current one if it is having the highest priority or the next one otherwise. It may happen that while loading the first candidate, the current one is used as the next TX message. Nothing can prevent such scenario and either the one with the highest priority is sent first or at the second place
- The second candidate is compared with the one previously defined as current. If the second candidate computed is already in one of the two local buffers, nothing needs to be done. If this is not the case, it is uploaded to provide the second highest TX message. In this particular case the second candidate overwrites the other local buffer

Here below is the flow chart of the TX-SCAN process.

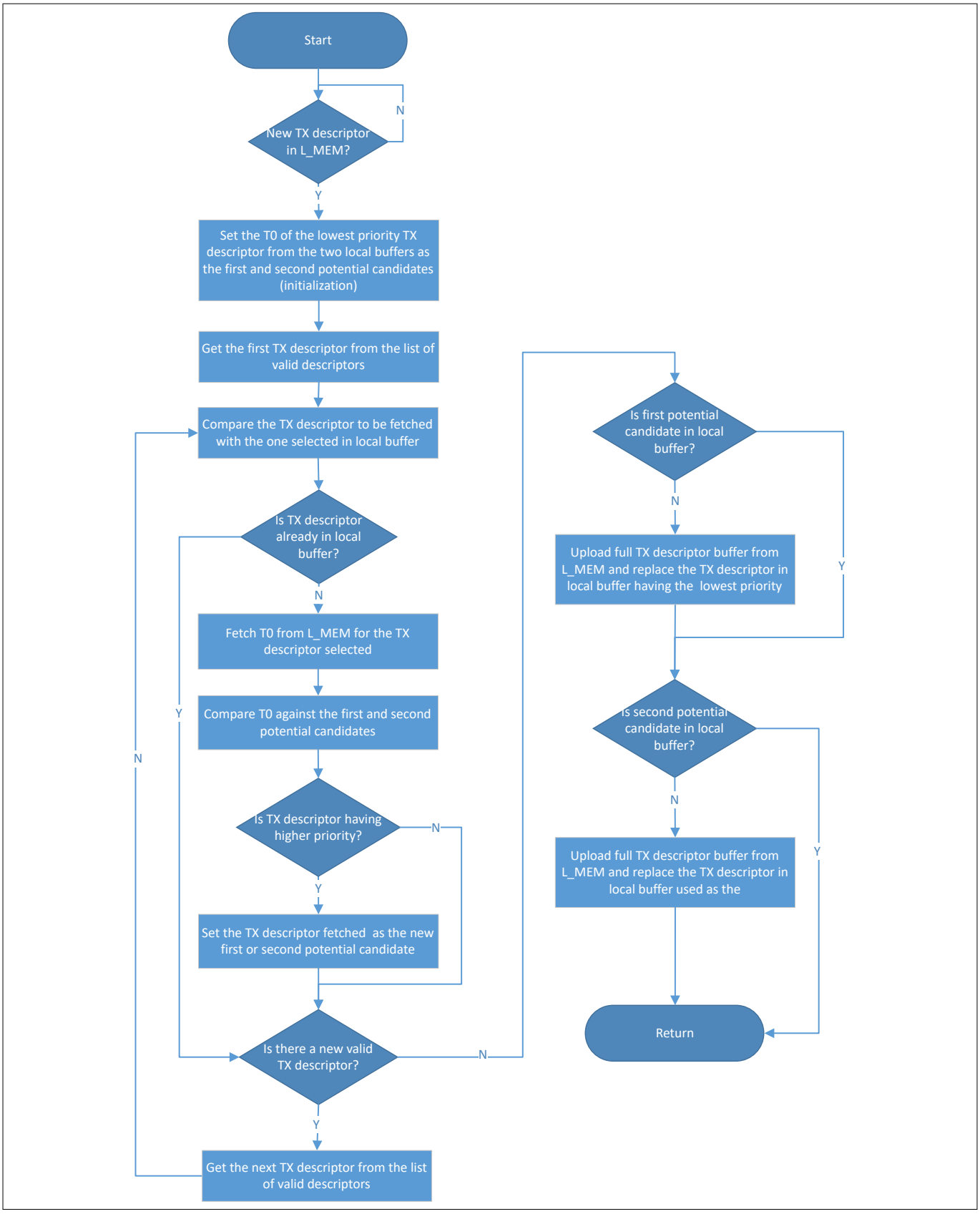


Figure 418 Flow chart of the TX-SCAN process

According to the status of the message sent, there will be two different actions:

- The TX message is sent successfully: the next candidate is considered as the current message. If no new TX descriptor is available in L_MEM, the next candidate to be fetched is already known, otherwise a new TX-SCAN run is launched
- The TX message is not sent successfully: The first next candidate is compare with the current one being not successful. If the current candidate is having a higher priority it will be considered as the one to select otherwise the other candidate is used instead. On top of it, the re-transmission counter defined in the Ni_MH_CFG register will limit the number of possible trials for the same message. If the counter exceeds the limit, the current candidate will no longer be considered, even it has the highest priority. The TX message is skipped and the next candidate is taken instead. If the counter does not reach the maximum value defined and a new message is taken instead, the counter is reset to 0

Some TX-SCAN scenarios are described here below with the following assumptions:

- Three TX FIFO Queues and 3 TX Priority Queue slots are defined
- The TX messages are sent without any pause (no RX message received)
- For the sake of simplicity, only the ID in T0 is used as the main criteria to select the TX message to be sent
- A TX message is defined per TX descriptor (only Header Descriptor are defined)

The first scenario describes the TX-SCAN based only on TX FIFO Queues running and considering every message as sent successfully.

Descriptor number used by TX FIFO QUEUE	TX FIFO QUEUES		
	0	1	2
	ID	ID	ID
-	-	-	-
N	-	18	-
N+1	-	5	-
N+2	-	40	-
N+3	-	30	10
N+4	1	1	38
N+5	0	110	80
N+6	4	24	20
N+7	7	6	6
N+8	11	4	8
N+9	20	29	15
N+10	100	50	39
-	-	-	-

Figure 419 Scenario 1: TX message requests pending in TX FIFO Queues

ID: TX Header Descriptor ID (one TX message per RX descriptor)

TX-SCAN run	TX Header Descriptor in L_MEM									CAN BUS		TX-SCAN results	
	TX FIFO QUEUES						TX PRIORITY Queue slots						
	0		1		2		0	1	2				
	CID	NID	CID	NID	CID	NID	CID	CID	CID	IDIP	MSG Result	NIDMNS	NIDMS
0	-	1	18	-	10	-	-	-	-	-	ok	-	1
1	1	0	18		10		-	-	-	1	ok	1	0
2	0	4	18		10		-	-	-	0	ok	0	4
3	4	7	18		10		-	-	-	4	ok	4	7
4	7	11	18		10		-	-	-	7	ok	7	10
5	11		18		10	38	-	-	-	10	ok	10	11
6	11	20	18		38		-	-	-	11	ok	11	18
7	20		18	5	38		-	-	-	18	ok	18	5
8	20		5	40	38		-	-	-	5	ok	5	20
9	20	100	40		38		-	-	-	20	ok	20	38
10	100		40		38	80	-	-	-	38	ok	38	40
11	100		40	30	80		-	-	-	40	ok	40	30
12	100		30	1	80					30	ok	30	1
13	-	-	-	-	-	-	-	-	-	-		-	-

Figure 420 Scenario 1: TX-SCAN process

CID: Current TX Header Descriptor ID to consider for TX-SCAN

NID: Next TX Header Descriptor ID to consider for TX-SCAN

NIDMS: Next TX Header Descriptor ID if Message Successful

NIDMNS: Next TX Header Descriptor ID if Message Not Successful

IDIP: TX Header Descriptor ID In Progress

The second scenario describes the TX-SCAN based on TX FIFO Queues and TX Priority Queue slots running and considering every message as sent successfully.

Descriptor number used by TX FIFO QUEUE	TX FIFO QUEUES		
	0	1	2
	ID	ID	ID
-	-	-	-
N	-	18	-
N+1	-	5	-
N+2	-	40	-
N+3	-	30	10
N+4	1	1	38
N+5	0	110	80
N+6	4	24	20
N+7	7	6	6
N+8	11	4	8
N+9	20	29	15
N+10	100	50	39
-	-	-	-

Figure 421 Scenario 2: TX message requests pending in TX FIFO Queues and TX priority queue slots

ID: TX Header Descriptor ID (one TX message per RX descriptor)

TX-SCAN run	TX Header Descriptor in L_MEM									CAN BUS		TX-SCAN results	
	TX FIFO QUEUE						PRIORITY QUEUE SLOTS						
	0		1		2		0	1	2				
	CID	NID	CID	NID	CID	NID	CID	CID	CID	IDIP	MSG Result	NIDMNS	NIDMS
0	-	1	18	-	10	-	-	-	-	-	ok	-	1
1	1	0	18		10		-	-	-	1	ok	1	0
2	0	4	18		10		-	-	-	0	ok	0	4
3	4	7	18		10		-	1	-	4	ok	1	1
4	7		18		10		-	1	-	1	ok	1	7
5	7	11	18		10		-	-	12	7	ok	7	10
6	11		18		10	38	5	-	12	10	ok	5	5
7	11		18		38		5	-	12	5	ok	5	11
8	11	20	18		38		-	-	12	11	ok	11	12
9	20		18		38		-	-	12	12	ok	12	18
10	20		18	5	38		-	-	-	18	ok	18	5
11	20		5	40	38		-	-	-	5	ok	5	20
12	20	100	40		38					20	ok	20	38
13	-	-	-	-	-	-	-	-	-	-	-	-	-

Figure 422 Scenario 2: TX-SCAN process

CID: Current TX Header Descriptor ID to consider for TX-SCAN

NID: Next TX Header Descriptor ID to consider for TX-SCAN

NIDMS: Next TX Header Descriptor ID if Message Successful

NIDMNS: Next TX Header Descriptor ID if Message Not Successful

IDIP: TX Header Descriptor ID In Progress The third scenario describes the TX-SCAN based on TX FIFO Queues and TX Priority Queue slots running and considering successful and not successful messages with re-transmission counter set to 1.

Descriptor number used by TX FIFO QUEUE	TX FIFO QUEUES		
	0	1	2
	ID	ID	ID
-	-	-	-
N	-	18	-
N+1	-	5	-
N+2	-	40	-
N+3	-	30	10
N+4	1	1	38
N+5	0	110	80
N+6	4	24	20
N+7	7	6	6
N+8	11	4	8
N+9	20	29	15
N+10	100	50	39
-	-	-	-

Figure 423 Scenario 3: TX message requests pending in TX FIFO Queues and TX priority queue slots with retransmission

ID: TX Header Descriptor ID (one TX message per RX descriptor)

TX-SCAN run	TX Header Descriptor in L_MEM									CAN BUS		TX-SCAN results	
	TX FIFO QUEUE						PRIORITY QUEUE SLOTS						
	0		1		2		0	1	2				
	CID	NID	CID	NID	CID	NID	CID	CID	CID	IDIP	MSG Result	NIDMNS	NIDMS
0	-	1	18	-	10	-	-	-	-	-	ok	-	1
1	1	0	18		10		-	-	-	1	ok	1	0
2	0	4	18		10		-	-	-	0	ok	0	4
3	4	7	18		10		-	1	-	4	nok	1	1
4	4	7						1	-	1	ok	1	4
5	4	7	18		10		-	-	-	4	nok	4	7
6	4	7	18		10		-	-	12	4	ok	7	7
7	7	11	18		10		5	-	12	7	ok	5	5
8	11		18		10		5	-	12	5	ok	5	10
9	11		18		10	38	-	-	12	10	ok	10	11
10	11	20	18		38		-	-	12	11	nok	11	12
11	11	20	18		38		-	-	12	11	nok	12	12
12	20		18		38		-	-	12	12	ok	12	18
13	-	-	-	-	-	-	-	-	-	-	-	-	-

Figure 424 Scenario 3: TX_SCAN process

CID: Current TX Header Descriptor ID to consider for TX-SCAN

NID: Next TX Header Descriptor ID to consider for TX-SCAN

NIDMS: Next TX Header Descriptor ID if Message Successful

NIDMNS: Next TX Header Descriptor ID if Message Not Successful

IDIP: TX Header Descriptor ID In Progress

Some debug registers are used to monitor the activity of the TX-Scan:

- The Ni_TX_SCAN_FC register provides the 2 best candidates selected from the previous TX-Scan run as well as the 2 best candidates for the current run
- The Ni_TX_SCAN_BC register gives all the relevant information (The TX FIFO Queue number and TX descriptor offset in that Queue or the TX Priority Queue slot number) regarding the two best candidates uploaded in the local buffer and ready to be sent to the PRT
- The Ni_TX_FQ_DESC_VALID register identifies which TX descriptor is valid, uploaded in the L_MEM and belonging to the list of potential candidate for the TX-Scan. The information displayed in that register covers for a given TX FIFO Queue, the current and the next TX descriptors of a queue that may be loaded in the L_MEM and valid
- The Ni_TX_PQ_DESC_VALID register provides the information of the slots of the TX Priority Queue loaded in the L_MEM and valid (ready for the TX-Scan)

22.5.3.21 TX filter

To ensure only declared TX messages can go through, the MH provides to the SW a way to define TX acceptance filters. Only the TX messages being filtered are considered for the arbitration process. There is the option to enable or disable this TX filtering process (see Ni_TX_FILTER_CTRL0.EN bit register) and so to leave all TX messages to go through or not. Several TX filter elements are defined and processed to determine if the TX message is accepted or rejected. The Ni_TX_FILTER_CTRL0 control register defines how the TX filter element are used, either standalone or combined.

A TX filter element uses reference values to compare with the TX message header data. The selection of the bit field to do the comparison can be configured for every TX filter element. Up to 16 TX filter elements can be defined and are applied to every TX messages when fetched from the L_MEM. There is no way to define those filters only for some specific queues. They apply to all TX messages whatever the TX FIFO Queues and TX Priority Queue slots.

When a TX filter error occurs, the faulty TX message is acknowledged with the status report "message rejected by TX filter". Thus, the SW is able to identify which one has been rejected, while scanning the TX descriptors from the TX FIFO Queues or TX Priority Queue slots. In order to determine the one being faulty and to avoid waiting for the TX descriptor, the Ni_TX_FILTER_ERR_INFO register provides the relevant information. The Ni_TX_FILTER_ERR_INFO.FQ when set to 1 defines a faulty TX message from a TX FIFO Queue otherwise from the TX Priority Queue. The FIFO Queue number is defined with the Ni_TX_FILTER_ERR_INFO.FQNS_PQS[3:0] bit field and the slot number with the Ni_TX_FILTER_ERR_INFO.FQNS_PQS[4:0] bit field.

22.5.3.21.1 Global configuration

To protect the setting of those TX filter elements, registers assigned to the configuration are protected, they can only be accessed while in privileged mode. Only the required application can modify the TX filter setting.

A global TX filter configuration register can be used to define, if the TX messages are accepted or rejected on match using the Ni_TX_FILTER_CTRL0.MODE bit register.

To notify the system that a TX message is rejected, a TX_FILTER_IRQ interrupt is generated. It is possible to enable and disable the TX filter interrupt using the Ni_TX_FILTER_CTRL0.IRQ_EN bit register. On top of it, the faulty TX descriptor is acknowledged immediately with the status rejected by TX filter.

The TX filter elements can be enabled or disabled independently from each other thanks to the Ni_TX_FILTER_CTRL1.VALID[15:0] bit registers.

In order to define the type of data to be compared with, either the VCID or SDT, the Ni_TX_FILTER_CTRL1.FIELD[15:0] is used. This register bit field is relevant for the CAN XL protocol only.

The definition of those TX filter elements is done through the setting of registers. Compared to the RX filter, the TX filter does not require to have access to the L_MEM, settings are done only in registers, see Ni_TX_FILTER_CTRL0, Ni_TX_FILTER_CTRL1, Ni_TX_FILTER_REFVAL{n} (n € {0, 1, 2, 3}). It is assumed that the TX filter elements once defined are statics and won’t change over time while the MH is running.

Refer to the TX filter registers for a more detailed description of the TX filter elements.

As the MH can support several CAN protocols, different options are possible on the TX filter, see the next sections for more details.

22.5.3.21.2 Classical CAN

All Classical CAN TX messages are either accepted or rejected, see Ni_TX_FILTER_CTRL0.CC_CAN bit register. There is no other option for such Classic CAN protocol. The TX filter elements are only used for the CAN XL protocol.

22.5.3.21.3 CAN FD

All CAN FD messages are either accepted or rejected, see Ni_TX_FILTER_CTRL0.CAN_FD bit register. There is no other option for such CAN FD protocol. The TX filter elements are only used for the CAN XL protocol.

22.5.3.21.4 CAN XL

Several options are possible to define the TX filter elements. Two global modes are defined for the overall TX filter elements, either Allow or Reject on match, see Ni_TX_FILTER_CTRL0.MODE bit-field register. When the Mode is configured to "Allow" (White List Approach), set by default, a frame is only transmitted, if there is a match on one of the TX filter elements. When the Mode is configured to "Reject" (Black List Approach) a frame is only transmitted, if there is no match at all.

Every TX filter elements is provided with a reference value to be compared with and which bit-field in the message header to be used, either VCID or SDT.

There is three different options on how to define a TX filter elements with the previous definition:

- Option 1:

  In normal mode, the reference value defined in Ni_TX_FILTER_REFVAL{n}.REF_VAL3, Ni_TX_FILTER_REFVAL{n}.REF_VAL2, Ni_TX_FILTER_REFVAL{n}.REF_VAL1 and Ni_TX_FILTER_REFVAL{n}.REF_VAL0 (n € {0, 1, 2, 3}) registers, is compared with either SDT or VCID. If e.g. Ni_TX_FILTER_REFVAL{n}.REF_VAL0 is defined to be compared to VCID and Ni_TX_FILTER_REFVAL{n}.REF_VAL1 to be compared to SDT than a CAN message will get a match if one of the two values matches.

  Any of the 16 TX filter elements can be used to compare the VCID or the SDT value, refer to the control bit register Ni_TX_FILTER_CTRL1.FIELD[n] (n € {0, 1, 2, 3, …, 15}) (when the bit n is set to 1, the SDT bit field is selected for the TX filter element n otherwise VCID).

  The Tx filter n is defined as valid or not valid using the Ni_TX_FILTER_CTRL1.VALID[n] (n € {0, 1, 2, 3, …, 15}) bit register. If the TX filter 1 is not considered, just set the Ni_TX_FILTER_CTRL1.VALID[1] to 0. An example of the option 1 is given in the table below.

  For such configuration, the Ni_TX_FILTER_CTRL0.MASK[n] (n € {0, 1, 2, …, 7}) bit register must be set to 0 for the given TX filter element pair (n and n+1 assuming n is even)

  To allow single compare value, meaning one reference value for one match, the Ni_TX_FILTER_CTRL0.COMB[n] must set to 0 for TX filter element n and n+1 (n being even). In this mode, there will be always TX filter element n and n+1 available.
- Option 2:

  Based on the normal mode and to increase the possible filtering options, two TX filter elements can be combined to allow VCID and SDT to be checked as only one filter. However, both values must be identical. As only a pair of TX filter elements reference values can be combined, Ni_TX_FILTER_REFVAL{n}.REF_VAL0 and Ni_TX_FILTER_REFVAL{n}.REF_VAL1 or Ni_TX_FILTER_REFVAL{n}.REF_VAL2 and Ni_TX_FILTER_REFVAL{n}.REF_VAL3 (n € {0, 1, 2, 3}) can be used.

  The selection of the bit field value to be compared with is defined by the Ni_TX_FILTER_CTRL1.FIELD[n] (n € {0, 1, 2, 3, …, 15}) bit register. The setting of this register is identical to the option 1.

  Only two adjacent TX filter elements can be configured as combined, using the Ni_TX_FILTER_CTRL0.COMB[n] (n € {0, 1, 2, …, 7}) bit register. When set to 1, TX filter n and n+1 (n being even) are combined. A an example, for the REF_VAL0/REF_VAL1 in the Ni_TX_FILTER_REFVAL0 register, the Ni_TX_FILTER_CTRL0.COMB[0] bit must be set to 1.

  For such configuration, the Ni_TX_FILTER_CTRL0.MASK[n] (n € {0, 1, 2, …, 7}) bit register must be set to 0 for the given TX filter element pair (n and n+1 assuming n is even).

  The TX filter element n and n+1 which are combined (n being even) need to be set as valid (set to 1) using the Ni_TX_FILTER_CTRL1.VALID[n] and Ni_TX_FILTER_CTRL1.VALID[n+1] (n € {0, 1, 2, 3, …, 15}) bit register. This means that combined TX filter elements require two bits to be set in the Ni_TX_FILTER_CTRL1 register.
- Option 3: In order to compare a range of values, a reference value and a mask are required. To provide such option, two TX filter elements TX filter n and n+ 1 (n is even) can be paired in a way that one is the value to be compared with while the other is the mask. As only a pair of TX filter elements reference values can be combined, Ni_TX_FILTER_REFVAL{n}.REF_VAL0 and Ni_TX_FILTER_REFVAL{n}.REF_VAL1 or Ni_TX_FILTER_REFVAL{n}.REF_VAL2 and Ni_TX_FILTER_REFVAL{n}.REF_VAL3 (n € {0, 1, 2, 3}) can be used.

  In order to set one of the two reference value as a mask, the appropriate bit in the Ni_TX_FILTER_CTRL0.MASK[n] (n € {0, 1, 2, …, 7}) bit register must be set to 1. As an eaxmple, the Ni_TX_FILTER_CTRL0.MASK[0] set to 1 is referring to the pair Ni_TX_FILTER_REFVAL0.REF_VAL0 and Ni_TX_FILTER_REFVAL0.REF_VAL1. In such configuration, the second reference value is the mask, hence the REF_VAL1 when considering the REF_VAL0/REF_VAL1 pair.

  The bit field to be compared with is defined by the Ni_TX_FILTER_CTRL0.FIELD bit field register. For the above example the Ni_TX_FILTER_CTRL0.FIELD[0] must be set either to 1 for SDT or 0 for VCID. When Ni_TX_FILTER_CTRL0.MASK[0]=1 then Ni_TX_FILTER_CTRL0.FIELD[1] is ignored by the MH, REF_VAL1 is interpreted as a mask only.

Only two adjacent TX filter elements can be configured as combined, using the Ni_TX_FILTER_CTRL0.COMB[n] (n € {0, 1, 2, …, 7}) bit register. When set to 1, TX filter n and n+1 (n being even) are combined. A an example, for the Ni_TX_FILTER_REFVAL0.REF_VAL0 and Ni_TX_FILTER_REFVAL0.REF_VAL1 bit field, the Ni_TX_FILTER_CTRL0.COMB[0] bit must be set to 1.

It is essential to enable this pair of TX filter element by setting the appropriate bits in the Ni_TX_FILTER_CTRL1 register. As an example the Ni_TX_FILTER_CTRL1.VALID[1:0] set to 2'b11 will enable the Ni_TX_FILTER_REFVAL0.REF_VAL0 (used as reference value) and Ni_TX_FILTER_REFVAL0.REF_VAL1 (used as a mask).

The following example shows 4 reference values defined in the Ni_TX_FILTER_REFVAL0 register, others behave the same.

Option 1 (single): A TX filter element uses only one reference value and one bit field (VCID or SDT)

Option 2 (Combined with matches): REF_VAL0 and REF_VAL1 are combined to provide a TX filter element that is able to compare VCID and SDT in the same filter. The same holds for REF_VAL2 and REF_VAL3.

Option 3 (Combined with mask and value): same as Option 2 with the difference that, REF_VAL0 is still the reference value to compare with (either VCID or SDT) but the REF_VAL1 is the mask to apply.

Table 1050 TX Filter Element options

Reference value	Option 1 (single)	Option 2 (Combined with matches)	Option 3 (Combined with mask and value)
REF_VAL0	REF_VAL0 = (VCID or SDT)	REF_VAL0 = VCID or SDT AND REF_VAL1 = VCID or SDT	REF_VAL0 (value) = REF_VAL1 (mask) AND (VCID or SDT)
REF_VAL1	REF_VAL1 = (VCID or SDT)		
REF_VAL2	REF_VAL2 = (VCID or SDT)	REF_VAL2 = VCID or SDT AND REF_VAL3 = VCID or SDT	REF_VAL2 (value)= REF_VAL3 (mask) AND (VCID or SDT)
REF_VAL3	REF_VAL3 = (VCID or SDT)		

22.5.3.22 RX filter

The RX filtering provides a way to reject or accept RX messages to the SW as well as to write those messages to a defined RX FIFO Queue.

Up to 255 RX filters can be defined. The RX filter is defined using an RX filter element (word of 32bit) associated with up to 2 pairs of reference(32bit)/mask(32bit) values. Those RX filter elements are continuous in the L_MEM and will be parsed one after the other. Regarding the reference/mask pairs they are defined after the RX filter element list in the L_MEM, up to 256 pairs can be declared.

To be flexible, an RX filter is made of up to 2 comparisons, each one using a reference value (32bit) and mask value (32bit) to apply on one of the incoming header message data word (R0, R1 or R3) from the PRT. A reference/mask pair can apply to any of the RX filter elements.

It is a SW task to define and write the RX filter elements and reference/mask pairs in the L_MEM. There is no direct access to it through the MH. The SW would need to access the L_MEM directly to program the relevant RX filter elements and reference/mask pairs.

The process of RX filtering is started as soon as the first 32bit word from the PRT is received, meaning R0. If the RX filter is fetching a filter element which requires R1, the process is on hold waiting for the 32bit word to be available. The same applies with R2 if only R0 and R1 are available.

The minimum time dedicated to the RX filtering is defined by the reception of an RX message when it is sent back to back. The timing window in this case is defined by the reception of two first 32bit word from the PRT (R0/R1) for the current RX message to the next two 32bit words (R0/R1) of the next message.

22.5.3.22.1 Global configuration

The register Ni_RX_FILTER_CTRL is used to set the general configuration for all RX Filters, a write access in Privileged mode is required. Once the MH is started (Ni_MH_CTRL.START = 1), the register is write protected. Here below are the list RX filter configuration setting:

- The number of RX filter elements is defined in the Ni_RX_FILTER_CTRL.NB_FE[7:0] bit field register
- Non-matching CAN frames can be accepted or rejected by configuring the Ni_RX_FILTER_CTRL.ANMF bit register to 1. If non-matching frames are accepted, they are stored in the default RX FIFO Queue defined by the Ni_RX_FILTER_CTRL.ANMF_FQ[2:0] bit field register
- If the RX filtering is not done in time, the data stored in the RX DMA FIFO may lead to an overflow. There is the option to allow the reception of such frames to the default RX FIFO Queue (defined by the Ni_RX_FILTER_CTRL.ANMF_FQ[2:0] bit field register). This feature is only enabled when setting the Ni_RX_FILTER_CTRL.ANFF bit field register to 1. The Ni_RX_FILTER_CTRL.THRESHOLD[4:0] defines the level in the RX DMA FIFO to reach before sending the non-filtered frames to the default RX FIFO Queue
- The default RX FIFO Queue number, to write RX message data when non-matching frames OR non-filtered frames are accepted, is defined in the Ni_RX_FILTER_CTRL.ANMF_FQ[2:0] bit field register. This default RX FIFO Queue value is considered if either the Ni_RX_FILTER_CTRL.ANMF bit or the Ni_RX_FILTER_CTRL.ANFF bit is set to 1. It is essential to enable and start this default RX FIFO Queue prior starting the PRT

22.5.3.22.2 Reference and Mask pair

Two comparisons in the RX Filter Element can be defined. Each comparison require a reference value (REFm) and a mask (MSKm) (m € {0, 1, 2, …, 255}). The reference and mask value are 32bit raw data. The 32bit reference value is compared with the first (R0), the second (R1) or the third (R3) word (32bit) of the RX message header (coming from the PRT) after applying the 32bit mask value.

The reference value and the mask are defined as a pair of two consecutive words of 32bit in L_MEM, starting with the REFm. Up to 256 pairs can be defined for the overall RX filter elements. All the pairs of reference value and mask are appended after the RX filter elements section in L_MEM, see MH Software Interface chapter. Any of the 256 reference/mask pairs can be selected for a given RX filter element.

The RX filter element will use an index to identify the position of the pairs to be used. The first pair is having the index 0, the second the index 1 and so on. The RX Filter Element is then referring to this index in the bit field CREFI0 and CREFI1 (see RX Filter Element Definition chapter), to identify the right pair to use.

22.5.3.22.3 RX Filter Element definition

For every filter element it is possible to:

- Define which RX FIFO Queue to use if RX message is accepted or rejected on match
- Generate an interrupt when a filter matches, triggering the signal RX_FILTER_IRQ to the system
- Define if the RX message expected is being defined into the black list (BLK bit-field)
- Define up to 2 comparisons with the option to:
  - Select the word index in the header message to look at (limited to R0, R1 and R2), see WI0 or WI1. It is important to note that R2 is a header data for the CAN XL while a payload data in case of Classic CAN and CAN FD. In case a remote frame is detected, any filter looking at R2 will be discarded
  - Define the reference value and mask index to perform the comparison, see CREFI0 or CREFI1 bit-field
  - Reject or accept on match, see AR0 and AR1
  - Perform two comparison for a match. When WI0 and WI1 are both different from 0, both comparisons are performed to know if there is a match. In such configuration, the AR0 and AR1 must be identical. In case they are not, only the comparison 0 is performed to check for a match

RX filter element is described in the following table:

Table 1051 RX filter element

Filter Element	Section	Bit-field	Name	Description
FEn	Control	FEn[31:28]	FIFO[3:0]	RX FIFO Queue number to store the receive CAN data
		FEn[27]	IRQ	Interrupt: When set to 1 an interrupt is triggered to the system when a match is detected
		FEn[26]	BLK	BlackList: When set to 1 the BLK bit defined in the RX message header is set to 1
		FEn[25:24]	Reserved	
	Comparison 1 (only considered with comparison 0)	FEn[23]	Reserved	
		FEn[22]	AR1	See AR0 bit-field description. Must always be equal to AR0.
		FEn[21:20]	WI1[1:0]	See WI0 bit-field description
		FEn[19:12]	CREFI1[7:0]	See CREFI0 bit-field description
	Comparison 0	FEn[11]	Reserved	
		FEn[10]	AR0	Accept or Reject on match: when set to 1 the RX message is rejected on match otherwise accepted on match

(table continues...)

Table 1051 (continued) RX filter element

Filter Element	Section	Bit-field	Name	Description
		FEn[9:8]	WI0[1:0]	Word Index: provide the index of the RX message header word to compare, 1 for R0, 2 for R1 and 3 for R2. R2 will not be considered as a valid index for remote frame or Classical CAN frame with no payload data or CAN FD frame without payload, the comparison is then canceled. When set to 0, no comparison is performed
		FEn[7:0]	CREFI0[7:0]	Comparison Reference Index: provide the index of the reference pair (value and mask) to be used for comparison

The RX filter element can be used in two different modes:

- Comparison 0 only ($WI0 > 0$ and $WI0 = 0$)
- Comparison 0 and comparison 1 ($WI0 > 0$ and $WI1 > 0$)

It is essential to understand that if the Comparison 1 is defined, this RX Filter element will not be considered if the Comparison 0 is not defined. In such case, the filter element is skipped.

Each time an RX message is received, the RX filtering is triggered and start the following sequence:

- Fetch the first filter element from L_MEM. The start address of this first filter element is defined in Ni_RX_FILTER_MEM_ADD.BASE_ADDR[15:0] register
- If the conditions listed below are met, the RX filter element will be discarded and the next filter element be fetched (if there is one available). In all other cases, go to next step:
  - $WI0 = 0$
  - $WI0 = 3$ and the frame is either a Classic CAN/CAN FD without payload or a Classic CAN remote frame
  - $WI1 = 3$ and the frame is either a Classic CAN/CAN FD without payload or a Classic CAN remote frame
  - $WI0 > 3$ and $WI1 > 3$ and AR0/AR1 not equal
- For the comparison 0, using the index CREFI0, the two words for the reference value and mask are fetched from L_MEM
- The comparison 0 is done between the word defined by the index WI0 and the reference value/mask fetched earlier. According to the WI1 bit several actions are taken:
  - WI1 set to 0 (Comparison 0 only): if there is a match, the RX filter will look at the AR0 bit to identify what to do with the RX message. It would then be accepted or rejected and the RX filter stops. If there

is no match, the RX filter keeps going with the next filter element, if there is one available, otherwise stops

- WI1 > 0 (Comparison 0 and Comparison 1 expected): if there is a match on the comparison 0, the RX filter will wait for the result of the comparison 1 to decide what to do with the RX message. If there is no match on comparison 0, the RX filter keeps going with the next filter element, if there is one available, otherwise stops

- The comparison 1 is done between the word defined by the index WI1 and the reference value/mask fetched earlier. If there is a match, the RX filter will look at the AR0 bit to identify what to do with the RX message. It would then be accepted or rejected and the RX filter stops. If there is no match, the RX filter keeps going with the next filter element, if there is one available, otherwise stops

22.5.3.22.4 RX Header Descriptor updates

When an RX message is accepted, the index of the RX filter element which has accepted the message on match is logged inside the RX Header Descriptor. In case the RX message is rejected, no information is provided to the SW.

The filtering process writes filtering information to three bit-fields of the header data (see RX Message header definition chapter):

- The BLK bit in the header data of the RX message can be set by an RX filter element to indicate to the SW a non expected messages. When the SW is parsing the RX message Header Descriptor it can easily identify them once available in S_MEM
- The FIDX[7:0] bit-field is used to provide the information of the filter element index which has been triggered for that message
- The FAB bit field when set to 1, indicates to the SW that there was not enough time to complete the RX filtering process. This bit will be set when such issue occurs and only if the Ni_RX_FILTER_CTRL.ANFF bit register is set to 1 and Ni_RX_FILTER_CTRL.THRESHOLD[4:0] bit field greater than 0 (threshold mechanism active)
- The FM bit-field when set to 1, notifies the SW that there was a match on the RX filtering. When the FAB and FM bit-fields are set to 0, the RX filtering process ending with no match

22.5.3.22.5 MH behavior according to RX Filter setting

The RX filter elements are sequentially read from the L_MEM. This process continues up to the point, where the RX filter result is known and the message can either be accepted or rejected.

In case the RX filter selects an RX FIFO Queue that is not enabled, the incoming frame is considered as a non-matching frame and is discarded, the RX_ABORT_IRQ is set to the system.

In a normal situation, when the RX filter result arrives in time, the RX message data are sent to the appropriate RX FIFO Queue or are rejected.

A default RX FIFO Queue can be defined for some configuration when Ni_RX_FILTER_CTRL.ANMF_FQ[2:0] is defined and Ni_RX_FILTER_CTRL.ANMF set to 1.

A threshold can be defined on the RX DMA FIFO to manage not filtered frames. The Ni_RX_FILTER_CTRL.ANFF must be set to 1 to activate the function and the Ni_RX_FILTER_CTRL.THRESHOLD[4:0] bit field defining the threshold value must be greater than 0. Once activated, those frames are sent to a default RX FIFO Queue, see Ni_RX_FILTER_CTRL.ANMF_FQ[2:0] bit field register.

Here below is the summary of the RX filter behavior when considering Ni_RX_FILTER_CTRL.NB_FE[7:0], Ni_RX_FILTER_CTRL.ANMF and Ni_RX_FILTER_CTRL.ANFF bit fields:

Table 1052 RX Filter behavior

ANMF	NBFE[7:0]	ANFF	RX Filter status
0	0	X	All RX frames are rejected
1	0	X	All RX messages are accepted and are going to the default RX FIFO Queue defined by Ni_RX_FILTER_CTRL.ANMF_FQ[2:0]
0	>0	0	Frames with match are going to RX FIFO Queues Frames with no match are rejected No threshold monitoring is performed during RX filtering
		1	Frames with match are going to RX FIFO Queues Frames with no match are rejected Frames reaching the RX DMA FIFO level set in the Ni_RX_FILTER_CTRL.THRESHOLD[4:0] register and not filtered, are going to the default RX FIFO Queue defined in Ni_RX_FILTER_CTRL.ANMF_FQ[2:0] register
1	>0	0	Frames with match are going to RX FIFO Queues Frames with no match are going to the default RX FIFO Queue defined by Ni_RX_FILTER_CTRL.ANMF_FQ[2:0] No threshold monitoring is performed during RX filtering

(table continues...)

Table 1052 (continued) RX Filter behavior

ANMF	NBFE[7:0]	ANFF	RX Filter status
		1	Frames with match are going to RX FIFO Queues Frames with no match are going to the default RX FIFO Queue defined by Ni_RX_FILTER_CTRL.ANMF_FQ[2:0] Frames reaching the RX DMA FIFO level set in the Ni_RX_FILTER_CTRL.THRESHOLD[4:0] register and not filtered, are going to the default RX FIFO Queue defined in Ni_RX_FILTER_CTRL.ANMF_FQ[2:0] register

The MH will manage the RX message differently if a default RX FIFO Queue is defined or not with or without a threshold defined. The configurations being able to use the two last options are described below:

Scenario 1:

Threshold mechanism is not active (Ni_RX_FILTER_CTRL.ANFF bit set to 0 and Ni_RX_FILTER_CTRL.NBFE[7:0] > 0).

Two scenarios can occur:

- The RX filtering result is not known before receiving the first word of the next RX message and the amount of the CAN frame data is lower than the RX DMA FIFO. The RX message is discarded with the RX_ABORT_IRQ interrupt and the RX_FILTER_ERR is also triggered to the system.
- In case the amount of data received, while waiting the RX filtering result, does exceed the maximum RX DMA FIFO size, the RX message is discarded with the RX_ABORT_IRQ interrupt. The DP_DO_ERR interrupt is triggered to notify an RX DMA FIFO overflow.

Scenario 2: Threshold mechanism is active (Ni_RX_FILTER_CTRL.ANNF bit set to 1 and Ni_RX_FILTER_CTRL.NBFE[7:0] > 0 and Ni_RX_FILTER_CTRL.THRESHOLD[4:0] value is greater than 0). This backup solution to avoid losing the RX message is possible due to the monitoring of the RX DMA FIFO level and a threshold configured in the Ni_RX_FILTER_CTRL.THRESHOLD[4:0] bit field register (see next section for more details).

Two scenarios can occur:

- The RX filtering result takes a very long time and the frame size is large. Once the threshold is reached, the RX descriptor from the default RX FIFO Queue is fetched from the S_MEM. Then, the RX buffer address pointer defined in the RX descriptor is used to write the first burst of data to the default RX FIFO Queue.
- The RX filtering result takes a very long time and the frame size does not reach the threshold value. As long as no new message is received, the RX filtering keeps going and a result could be expected in time. In case a new RX message is received, the current RX message is discarded with the RX_ABORT_IRQ interrupt. The RX_FILTER_ERR is also triggered to the system to identified a potential issue with the RX filter.

Here below is a table to summarize the different scenarios:

Table 1053 RX Filter scenarios

Frame length	Filter result when threshold is reached	Filter result when next RX message arrives	Action of MH	Comments
Short	Not possible	Available	Store frame in RX FIFO Queue defined by RX filter result	Normal behavior
Short	Not possible	Not available	Discard frame	The overall processing time is too long due to a high number of filters and/or a low core clock frequency,
Long	Available and ANFF = X	Not considered	Store frame in RX FIFO Queue defined by RX filter result	Normal behavior
Long	Not available and ANFF = 0	Not considered	Keeps waiting for the RX filter result	Within this case, the message is written in time to the right RX FIFO Queue or discarded due to data overflow on the RX DMA FIFO
Long	Not available and ANFF = 1	Not considered	Store frame in default RX FIFO Queue	Threshold must be set such a way there is enough time to fetch RX descriptor and to write burst data to S_MEM,

22.5.3.22.6 Threshold computation

The MH uses the Ni_RX_FILTER_CTRL.THRESHOLD[4:0] only when the Ni_RX_FILTER_CTRL.ANFF bit is set to 1 and the threshold value is greater than 0. The threshold value to be configured by the user depends on the S_MEM latency, the CAN protocols supported and the CAN XL data bit rate. The RX DMA FIFO has a size of 32 words (128 byte).

Case 1: Only Classical CAN and/or CAN FD messages are received: The RX DMA FIFO is capable of storing a complete Classical CAN or CAN FD message. The feature Ni_RX_FILTER_CTRL.ANFF should not be used.

Note: When Ni_RX_FILTER_CTRL.ANFF=1 and Ni_RX_FILTER_CTRL.THRESHOLD[4:0] is set to 19 or larger, then the threshold will never be reached. This implicitly disables the threshold, because it will be never be reached by a Classical CAN or CAN FD message.

Case 2: Also CAN XL messages are received: CAN XL messages can be much longer than the RX DMA FIFO size. When the fill level of the RX DMA FIFO reaches the threshold, the MH will fetch the RX descriptor from the default RX FIFO Queue and will write the first burst of data stored in the RX DMA FIFO. For the case, that no RX filter result is available at the point in time the threshold is reached, this mechanism prevents a data overflow that would occur on the RX DMA FIFO and allows the reception of the message. The threshold divides the time budget provided by the RX DMA FIFO into two parts: (1) time to do the RX Filtering, (2) Time to fetch the RX Descriptor from S_MEM and to write the first burst of data to S_MEM. The user should configure the threshold as large as possible, to give the RX Filtering enough time, but as low as necessary to be able to receive the message, in case RX Filtering is not finished yet.

22.5.3.22.7 RX Filter processing time

The number of accesses to evaluate an RX filter element is defined as follow:

Classical CAN and CAN FD:

One single word access (definition of the filter element) and 2 words for the mask and value to compare with

CAN XL:

same as Classical CAN / CAN FD or one single word and 2 reads of 2 words for the 2 mask and value to compare with

The RX filter element access time depends on the read latency Lr (defined in number of CLK clock cycles):

Classical CAN and CAN FD:

$$RXFilter\ \ element\ \ processing\ time\left(\mu s\right)\ \ =\ \ \left(\left(Lr+2\right)+\left(Lr+1+2\right)\right)*\left(\frac{1}{CLK(MHz)}\right)$$

CAN XL:

$$RXFilter\ \ element\ \ processing\ time\left(\mu s\right)\ \ =\ \ \left(\left(Lr+2\right)+\left(Lr+1+2\right)\right)*\left(\frac{1}{CLK(MHz)}\right)\ with\ one\ comparison$$

$$RXFilter\ \ element\ \ processing\ time\left(\mu s\right)\ \ =\ \ \left(\left(Lr+2\right)+2*\left(Lr+1+2\right)\right)*\left(\frac{1}{CLK(MHz)}\right)\ with$$

two $comparisons$

The overall RX filter time is computed as follow:

$$RX\ Filter\ processing\ time\left(\mu s\right)\ \ =\ \ \left(Nbfe1c*\left(\left(Lr+2\right)+\left(Lr+1+2\right)\right)+Nbfe2c*\left(\left(Lr+2\right)+2*\left(Lr+1+2\right)\right)\right)*\left(\frac{1}{CLK(MHz)}\right)$$

where Nbfe1c = Number of filter element with 1 comparison, Nbfe2c = Number of filter element with 2 comparisons and the read latency Lr (defined in number of CLK clock cycles).

22.5.3.23 Local Memory Controller

The XCAND_MH_MEM_CTRL block is in charge of reading and writing the L_MEM. The Local Memory Controller is managing all requests and data transfer for the different blocks running concurrently:

- The writes of TX descriptors from the XCAND_DESC block
- The read of RX filter elements from the XCAND_RX_PATH
- The read of TX descriptor from XCAND_TX_PATH when a message has to be sent
- The read of TX descriptor from XCAND_TX_PATH for TX SCAN (selection of the highest priority TX message)

22.5.3.23.1 Local Memory side band signals

It is assumes that the L_MEM provides safety measure to protect data. The safety protection implemented in the L_MEM could either report error when reading a data (Single Error Detection) or be able to correct it in some cases (Single Error Correction and Double Error Detection for example). To address those two options, two input side band signals denominated MEM_SFTY_CE and MEM_SFTY_UE, are provided with the MEM_AXI interface.

Here below are the expected behavior of those signals and the expected response on the MEM_AXI interface:

- Error Detection Only: When a corrupted data is read from the L_MEM, a pulse of one CLK clock cycle must be generated on the MEM_SFTY_UE input signal. The MEM_AXI interface must report a SLVERR response on the MEM_AXI bus. The MEM_SFTY_UE input signal can be fully asynchronous to the MEM_AXI interface.
- Error Detection and Correction: When a corrupted data is read from the L_MEM but is corrected, a pulse of one CLK clock cycle must be generated on the MEM_SFTY_CE input signal. The MEM_AXI interface must report an OKAY response on the MEM_AXI bus. The MEM_SFTY_UE input signal can be fully asynchronous to the MEM_AXI interface.

22.5.3.23.2 Address bus

The XCAND_MH_MEM_CTRL block is able to address up to 64 Kbytes memory space (MEM_AXI_AWADDR[15:0] and MEM_AXI_ARADDR[15:0]).

The address burst value is always 32 bit aligned.

22.5.3.23.3 Burst size

The maximum number of bytes to transfer in each data transfer is fixed and set to 4. Any read or write transfer is always using 32 bit.

As a consequence, the write strobe signals are not managed by the XCAND_MH_MEM_CTRL as all 4 bytes are always written.

22.5.3.23.4 Burst length

The Local Memory Controller for the AXI read and write transfers supports INCR burst length 1, 2 and 8 considering an AXI 32 bit data bus width.

The burst lengths from/to the L_MEM are defined based on the data type of information to be used. Here below are the expected burst length from/to the different sub-blocks:

- XCAND_MH_DESC: This sub-block writes the TX descriptor allocated to TX FIFO Queues and TX Priority Queue slots. The burst length is fixed and set to $8 \times 32\ bit$. There is no read access from this sub-module
- XCAND_MH_RX: This sub-block reads the RX filter elements and reference/mask values to perform the RX message filtering. The burst length is set to $1 \times 32\ bit$ for the RX filter element and $2 \times 32 bit$ for the reference/mask value. There is no write access from this sub module
- XCAND_MH_TX: This sub-block reads two type of information, the TX descriptor to be sent as the next candidate to the TX_MSG interface and part of the TX descriptors assigned to TX FIFO Queues and TX Priority Queue slots. A fixed burst length of $1 \times 32\ bit$ is used for the TX SCAN and $8 \times 32\ bit$ is for the TX descriptor

22.5.3.23.5 Outstanding

As the L_MEM can be shared across several X_CAN instances and many accesses are required for RX filtering and TX SCAN, 2 outstanding read transactions can supported. As only a few writes are required from the MH point of view, only 1 outstanding write transaction is supported.

22.5.3.23.6 Burst type

The only burst type supported is the burst incrementing INCR.

The WRAP/FIXED burst type is not supported.

22.5.3.23.7 Multi-region

The Local Memory Controller does not support multiple region interfaces.

22.5.3.23.8 Memory attributes

The memory attributes for the read or write accesses to the L_MEM are Normal, Non-modifiable (Non-cacheable in AXI3) and Non-bufferable. No read-allocate nor No Write-allocate are expected on this interface and would be set to 1'b0.

As a reminder, Non-bufferable means (See [3] for mode details):

- The write response must be obtained from the destination
- Read data must be obtained from the destination
- Transactions are Non-modifiable
- Read and write transactions from the same ID to addresses that overlap must remain ordered

As a reminder, Non-modifiable means:

- A Non-modifiable transaction must not be split into multiple transactions or merged with other transactions
- In a Non-modifiable transaction, the parameters AxADDR, AxSIZE, AxLEN, AxBURST and AxPROT must not be changed

22.5.3.23.9 Access permissions

It is considered that any access is always defined as a Data, Secure and the operating mode is Unprivileged, see [3] for more details. Those setting cannot be changed by SW.

Therefore, any access from the MH which needs to be non-secure, must be managed with an external and dedicated logic attached to the MEM_AXI interface.

As an example, the RX filter elements and reference/mask can be stored in a secure area in the L_MEM, as a consequence the MEM_AXI_ARPROT[1] and MEM_AXI_AWPROT[1] are set to 0_B. Doing so, the MH is able to read secure and non-secure data in the L_MEM, with the assumption that non-secure area is always accessible by a secure access. This means MEM_AXI_A(W/R)PROT[2:0] is set to 000_B.

22.5.3.23.10 Transaction ID

The L_MEM Controller builds the ID of every burst access based on the source of request. It provides a way to track which sub-block is doing the access at any time on the system bus.

For the AXI read interface, the MEM_AXI_ARID[1:0] defines the channel number as follow:

00_B => XCAND_MH_TX reads TX descriptor from L_MEM

2’b01 => XCAND_MH_TX read part of TX descriptor from L_MEM for TX SCAN

10_B => XCAND_MH_RX reads RX filter elements and reference/mask values from L_MEM

11_B => Reserved

For the AXI write interface, the MEM_AXI_AWID[0] defines the channel number as follow:

0_B => XCAND_MH_DESC writes TX descriptor to L_MEM

1_B => Reserved

22.5.3.24 Trace and debug

22.5.3.24.1 Interrupts

For integration verification it is possible to trig functional and safety interrupts by SW. Here is the procedure:

1. Unlock the Ni_DEBUG_TEST_CTRL register, see section Lock Mechanism Protection in Register Protection chapter
2. Write the Ni_DEBUG_TEST_CTRL.TEST_IRQ_EN bit to 1 in privileged mode
3. Once the access to the Ni_INT_TEST0 and Ni_INT_TEST1 registers are allowed (always accessible once opened), write 1 to the relevant bit to set the appropriate interrupt line.

Re-lock the access to the Ni_INT_TEST0 and Ni_INT_TEST1 registers. Step 1) and 2) needs to be done with Ni_DEBUG_TEST_CTRL.TEST_IRQ_EN bit set to 0 instead.

22.5.3.24.2 Hardware debug port

The 16bit HDP bus provides some visibility to internal signals to debug the MH. By default, there is no activity on the HDP bus.

To enable the toggling of the HW signal on the HDP bus, set the Ni_DEBUG_TEST_CTRL.HDP_EN bit to 1.

1) Unlock the Ni_DEBUG_TEST_CTRL register, see section Lock Mechanism Protection in Register Protection chapter

2) Write the Ni_DEBUG_TEST_CTRL.HDP_EN bit to 1 and the selected set to be monitored on the HDP using the Ni_DEBUG_TEST_CTRL.HDP_SEL[2:0]. This access must be done in privileged mode.

To disable the Hardware Debug Port monitoring, do the previous set with Ni_DEBUG_TEST_CTRL.HDP_EN bit to 0.

Up to 8 sets can be defined using the Ni_DEBUG_TEST_CTRL.HDP_SEL[2:0] bit field. When the value is set to n the set n is selected on the HDP bus.

INTERRUPTS:

The interrupt line assigned to the RX or TX FIFO Queue can be monitored individually. Therefore, it is possible to track the activity of the FIFO Queues while they are running. To allow the visibility of all MH interrupts, on the same HDP set, the TX FIFO Queues interrupt lines are gathered to only one single HW internal signal called TX_FQ_IRQ_ORED (the interrupts are 'ored'). The same is done on the RX FIFO Queues and so the interrupts are 'ored' to provide the HW internal signal RX_FQ_IRQ_ORED.

***Note**: There are two possible sources to trig an interrupt (valid for TX_FQ_IRQ[7:0], RX_FQ_IRQ[7:0] and TX_PQ_IRQ interrupt lines): one is related to functional and the other one is from the Ni_INT_TEST0 and Ni_INT_TEST1 registers (for integration test only). Only the functional interrupt source is displayed on the HDP set. Therefore, when an interrupt is triggered, by a write access to either Ni_INT_TEST0 or Ni_INT_TEST1 register, it will not be visible on the HDP. Nevertheless, the interrupt line is properly set at the MH interface.*

INTERFACES:

To ensure the traceability of the traffic going from and to the MH, the following interfaces can be monitored through one of the HDP sets:

- DMA_AXI interface (control signals) used to manage RX/TX descriptors and RX/TX message data
- MEM_AXI interface (control signals) used to manage TX descriptors for TX-Scan and RX filtering
- TX_MSG interface (control signals) used to transmit TX message from MH to PRT
- RX_MSG interface (control signals) used to receive RX message from PRT to MH

The table below shows the description of sets available on the MH HDP bus.

Table 1054 Signals available on the HDP bus

MH_HDP[15:0]	Set 0 (interrupts)	Set 1 (RX and TX path)	Set 2 (TX Scan)	Set 3 (MH_PRT Interface)	Set 4 (Write AXI DMA interface)	Set 5 (Read AXI DMA interface)	Set 6 (Write AXI MEM interface)	Set 7 (Read AXI MEM interface)
15	TX_FQ_IRQ[7]	CLK	FH_OFFSET[9]	CLK	CLK	CLK	CLK	CLK
14	TX_FQ_IRQ[6]	TX_FQ_IRQ_ORED	FH_OFFSET[8]	TX_MSG_WVALID	DMA_AXI_BID[0]	DMA_AXI_RID[1]	MEM_AXI_BID[0]	MEM_AXI_RID[1]
13	TX_FQ_IRQ[5]	RX_FQ_IRQ_ORED	FH_OFFSET[7]	TX_MSG_WUSER[1]	DMA_AXI_BVALID	DMA_AXI_RID[0]	MEM_AXI_BVALID	MEM_AXI_RID[0]
12	TX_FQ_IRQ[4]	TX_PQ_IRQ	FH_OFFSET[6]	TX_MSG_WUSER[0]	DMA_AXI_BREADY	NA	MEM_AXI_BREADY	NA
11	TX_FQ_IRQ[3]	RX_FILTER_ERR	FH_OFFSET[5]	TX_MSG_WREADY	DMA_AXI_BRESP[1]	DMA_AXI_RRESP[1]	MEM_AXI_BRESP[1]	MEM_AXI_RRESP[1]
10	TX_FQ_IRQ[2]	RX_FILTER_IRQ	FH_OFFSET[4]	TX_MSG_BVALID	DMA_AXI_BRESP[0]	DMA_AXI_RRESP[0]	MEM_AXI_BRESP[0]	MEM_AXI_RRESP[0]
9	TX_FQ_IRQ[1]	TX_FILTER_IRQ	FH_OFFSET[3]	TX_MSG_BUSER_STATUS[2]	DMA_AXI_WREADY	DMA_AXI_RREADY	MEM_AXI_WREADY	MEM_AXI_RREADY
8	TX_FQ_IRQ[0]	STAT_IRQ	FH_OFFSET[2]	TX_MSG_BUSER_STATUS[1]	DMA_AXI_WVALID	DMA_AXI_RVALID	MEM_AXI_WVALID	MEM_AXI_RVALID
7	RX_FQ_IRQ[7]	TX_ABORT_IRQ	FH_OFFSET[1]	TX_MSG_BUSER_STATUS[0]	DMA_AXI_WLAST	DMA_AXI_RLAST	MEM_AXI_WLAST	MEM_AXI_RLAST
6	RX_FQ_IRQ[6]	RX_ABORT_IRQ	FH_OFFSET[0]	TX_MSG_BREADY	NA	DMA_AXI_ARID[1]	NA	MEM_AXI_ARID[1]
5	RX_FQ_IRQ[5]	DP_SEQ_ERR	FH_FQN_PQN[4]	RX_MSG_WVALID	DMA_AXI_AWID[0]	DMA_AXI_ARID[0]	MEM_AXI_AWID[0]	MEM_AXI_ARID[0]
4	RX_FQ_IRQ[4]	DP_DO_ER R	FH_FQN_PQN[3]	RX_MSG_WUSER[2]	DMA_AXI_AWVALID	DMA_AXI_ARVALID	MEM_AXI_AWVALID	MEM_AXI_ARVALID
3	RX_FQ_IRQ[3]	STOP_IRQ	FH_FQN_PQN[2]	RX_MSG_WUSER[1]	DMA_AXI_AWREADY	DMA_AXI_ARREADY	MEM_AXI_AWREADY	MEM_AXI_ARREADY
2	RX_FQ_IRQ[2]	RESP_ER R[1]	FH_FQN_PQN[1]	RX_MSG_WUSER[0]	DMA_AXI_AWLEN[2]	DMA_AXI_ARLEN[2]	MEM_AXI_AWLEN[2]	MEM_AXI_ARLEN[2]
1	RX_FQ_IRQ[1]	RESP_ER R[0]	FH_FQN_PQN[0]	RX_MSG_WREADY	DMA_AXI_AWLEN[1]	DMA_AXI_ARLEN[1]	MEM_AXI_AWLEN[1]	MEM_AXI_ARLEN[1]
0	RX_FQ_IRQ[0]	ENABLE	FH_PQ	ENABLE	DMA_AXI_AWLEN[0]	DMA_AXI_ARLEN[0]	MEM_AXI_AWLEN[0]	MEM_AXI_ARLEN[0]

22.5.3.24.3 TX-scan

In order to keep track of the TX-Scan process, some registers provide the relevant information to observe the selection of the next TX message, meaning, which TX FIFO Queue number and which message within this FIFO Queue is selected, or which TX Priority Queue slot number. The Ni_TX_SCAN_FC and Ni_TX_SCAN_BC registers are monitoring the TX-Scan activity, see Software Interface chapter for more details. The duration of a CAN frame is large enough to make it possible, to read those registers in time and get some valuable information.

The Ni_TX_SCAN_FC register provides the information of the source of the four first candidates selected by the TX-Scan., meaning the TX FIFO Queue number or the TX Priority Queue slot number. The source of the two first candidates are defined by:

- The Ni_TX_SCAN_FC.FQ_PQ{n} (n € {0, 1}) bit register: when set to 0, the first or second candidate is a TX FIFO Queue. In fact, Ni_TX_SCAN_FC.FQ_PQ0 = Ni_TX_SCAN_BC.FH_PQ and Ni_TX_SCAN_FC.FQ_PQ1 = Ni_TX_SCAN_BC.SH_PQ, see Ni_TX_SCAN_BC register description below
- The Ni_TX_SCAN_FC.FQN_PQSN{n} (n € {0, 1}) bit register: define either a TX FIFO Queue number or a TX Priority Queue slot number according to the Ni_TX_SCAN_FC.FQ_PQ{n} (n € {0, 1}) bit register. In fact, Ni_TX_SCAN_FC.FQN_PQSN0 = Ni_TX_SCAN_BC.FH_FQN_PQSN and Ni_TX_SCAN_FC.FQN_PQSN1 = Ni_TX_SCAN_BC.SH_FQN_PQSN, see Ni_TX_SCAN_BC register description below
- The two sources of the last two candidates are monitoring the selection of a TX-Scan. It is essential to note that the value in those registers is not stable, compare to the source of the two first candidate, and will change during a TX-Scan run. When the Ni_TX_SCAN_FC.FQ_PQ{n} (n € {0, 1, …, 3}) is set to 0, the candidate n is a TX FIFO Queue, looking at the Ni_TX_SCAN_FC.FQN_PQSN{n} (n € {0, 1, …, 3} bit field, provides the number. If the Ni_TX_SCAN_FC.FQ_PQ{n} (n € {0, 1, …, 3}) is set to 1, the candidate n is a TX Priority Queue and the slot number is defined by the Ni_TX_SCAN_FC.FQN_PQSN{n} (n € {0, 1, …, 3} bit field.

The value of the Ni_TX_SCAN_FC register is updated when a new TX Scan result is available, see TX-SCAN chapter.

The Ni_TX_SCAN_BC register gives the full reference of the first and second highest priority messages, defined and uploaded at the end of a TX-Scan run (see Buffer A and B in TX MESSAGE HANDLER chapter). The first highest candidate is the one selected and sent to the CAN bus. The second highest priority candidate is the TX message to be sent, once the transmission of the first highest candidate is completed. The register values provide full visibility of the two message candidates stored locally in Buffer A and B, see TX MESSAGE HANDLER chapter for more details. As such, those registers are stable over time and do change only after at the end of a TX-Scan.

The first best candidate is defined by:

- The Ni_TX_SCAN_BC.FH_PQ bit register: when set to 0, the candidate is a TX FIFO Queue
- The Ni_TX_SCAN_BC.FH_FQN_PQSN bit register: define either a TX FIFO Queue number or a TX Priority Queue slot number according to the Ni_TX_SCAN_BC.FH_PQ bit register
- The Ni_TX_SCAN_BC.FH_OFFSET bit register: define the offset (in 32byte) of the TX descriptor selected, starting from the initial start address of the TX FIFO Queue (defined in the Ni_TX_SCAN_BC.FH_FQN_PQS bit register) with TX descriptor address = TX FIFO Queue start address + offset * 32byte. When the candidate is a TX Priority Queue slot, the Ni_TX_SCAN_BC.FH_OFFSET register has no meaning and is set to 0.

The second best candidate is defined by:

- The Ni_TX_SCAN_BC.SH_PQ bit register: when set to 0, the candidate is a TX FIFO Queue
- The Ni_TX_SCAN_BC.SH_FQN_PQSN bit register: define either the TX FIFO Queue number or the TX Priority Queue slot number according to the Ni_TX_SCAN_BC.FH_PQ bit register
- The Ni_TX_SCAN_BC.SH_OFFSET bit register: define the offset (in 32byte) of the TX descriptor selected, starting from the initial start address of the TX FIFO Queue (defined in the Ni_TX_SCAN_BC.SH_FQN_PQSN bit register) with TX descriptor address = TX FIFO Queue start address + offset * 32byte. It is important to note that, when the TX FIFO queue selected for the first best candidate is identical to the one for the second, the offset would be also identical. In such scenario, the second best candidate is always the next

TX descriptor of that TX FIFO Queue. When the candidate is a TX Priority Queue slot, the Ni_TX_SCAN_BC.SH_OFFSET register has no meaning and is set to 0

22.5.3.24.4 TX Descriptor tracking in a TX FIFO Queue

The current and next TX descriptors for a given TX FIFO Queue n are stored in the L_MEM and can be identified in the Ni_TX_FQ_DESC_VALID.DESC_NC_VALID[n] and Ni_TX_FQ_DESC_VALID.DESC_CN_VALID[n] bit registers.

Here are the status for those bit register when progressing with the TX FIFO Queue n:

Initial start:

1. The current TX descriptor (first one in case this is an initial start) fetched from S_MEM and written to L_MEM is leading to Ni_TX_FQ_DESC_VALID.DESC_CN_VALID[n] = 1 and Ni_TX_FQ_DESC_VALID.DESC_NC_VALID[n] = 0.
2. If the current TX descriptor is about to be sent go to 3), otherwise stay in 2) and no updates are done on bit registers.
3. The next descriptor is fetched from S_MEM and written in L_MEM.
   a. If the next TX descriptor is not valid then the TX FIFO Queue n is put on hold. The Ni_TX_FQ_DESC_VALID.DESC_CN_VALID[n] set to 1, goes to 0 once the TX message is sent (Ni_TX_FQ_DESC_VALID.DESC_NC_VALID[n] =0).
   b. If the descriptor is valid, Ni_TX_FQ_DESC_VALID.DESC_CN_VALID[n] = 1 and Ni_TX_FQ_DESC_VALID.DESC_NC_VALID[n] =1, go to 4)
4. When the current TX message is fully sent, Ni_TX_FQ_DESC_VALID.DESC_CN_VALID[n] = 0 and Ni_TX_FQ_DESC_VALID.DESC_NC_VALID[n] =1.
5. If the current TX descriptor is about to be sent go to 6), otherwise stay in 5) and no updates are done on bit registers.
6. The next descriptor is fetched from S_MEM and written in L_MEM.
   a. If the next TX descriptor is not valid then the TX FIFO Queue n is put on hold. The Ni_TX_FQ_DESC_VALID.DESC_NC_VALID[n] set to 1, goes to 0 once the TX message is sent (Ni_TX_FQ_DESC_VALID.DESC_CN_VALID[n] =0).
   b. If the descriptor is valid, Ni_TX_FQ_DESC_VALID.DESC_NC_VALID[n] = 1 and Ni_TX_FQ_DESC_VALID.DESC_CN_VALID[n] =1, go to 7)
7. When the current TX message is fully sent, Ni_TX_FQ_DESC_VALID.DESC_NC_VALID[n] = 0 and Ni_TX_FQ_DESC_VALID.DESC_CN_VALID[n] =1, go to 2)

22.5.3.24.5 TX Descriptor tracking in TX Priority Queue

As soon as a TX Priority Queue slot n is started, the corresponding TX descriptor is fetched from the S_MEM and written to the L_MEM. When the TX descriptor assigned to the slot n is fully written in the L_MEM, the Ni_TX_PQ_DESC_VALID.DESC_VALID[n] is set to 1.

If the TX descriptor fetched is not valid or has any safety issue, the Ni_TX_PQ_DESC_VALID.DESC_VALID[n] is not set. In case the TX message of the slot n is discarded, the Ni_TX_PQ_DESC_VALID.DESC_VALID[n] is set back to 0.

22.5.3.25 RX and TX Statistics

22.5.3.25.1 RX statistic counters

Two 12bit counters are provided to keep track of how many messages have been received successfully/ unsuccessfully, see Ni_RX_STATISTICS register. When a counter has reached the maximum value, it will wrap to zero with the next increment. The counters can be cleared (set to 0) by writing 0 to the dedicated register bit

field. To identify when counters are wrapping, the STATS_IRQ interrupt line is triggered to the system. To identify the counter which has wrapped, a read to the Ni_STATS_INT_STS register is required. Writing a 1 to the corresponding bit will clear the bit.

Here are the list of root cause to increment Ni_RX_STATISTICS.SUCC[11:0] counter:

- When an RX message is stored in S_MEM and its RX Header descriptor is acknowledged

Here are the list of root cause to increment Ni_RX_STATISTICS.UNSUCC[11:0] counter.

Safety or Errors:

- When an RX data parity error is detected
- When an RX address pointer parity error is detected
- When an RX descriptor error (request, CRC or invalid) is detected and used for the current RX message

Functional:

- When an ABORT code word is received from the PRT
- When a DO code word is received from the PRT
- When the RX message cannot be written to the RX FIFO Queue (queue not enable and/or started)
- When the RX filtering has not finished in time before a new one
- When a data overflow occurs on the RX DMA FIFO
- When a new RX message is received while one is already in progress (filtered and RX descriptor already fetched from system memory)
- When the PRT ENABLE signal is going from High to Low when receiving an RX frame

22.5.3.25.2 TX statistic counters

Two 12bit counters are provided to keep track of how many messages have been transmitted successfully/ unsuccessfully, see Ni_TX_STATISTICS register. When a counter has reached the maximum value, it will wrap to zero with the next increment. The counters can be cleared (set to 0) by writing 0 to the dedicated register bit field. To identify when counters are wrapping, the STATS_IRQ interrupt line is triggered to the system. To identify the counter which has wrapped, a read to the Ni_STATS_INT_STS register is required. Writing a 1 to the corresponding bit will clear the bit.

Here are the list of root cause to increment Ni_TX_STATISTICS.SUCC[11:0] counter:

- When a TX message is fully sent to the PRT, and its TX Header descriptor is acknowledged

Here are the list of root cause to increment Ni_TX_STATISTICS.UNSUCC[11:0] counter.

Safety or Errors:

- Not applicable

Functional:

- When a HFI code word is received from the PRT
- When the maximum number of transmission allowed for a given TX message is reached
- When the TX filtering has rejected a TX message

22.5.3.26 Register access

The MH registers are accessible in read/write mode through its AXI4-Lite slave interface HOST_AXI (compliant to AMBA 4 ARM Ltd protocol, see [3]).

Any access to registers, either read or write, must use a 32bit aligned address, otherwise a SLVERR is provided as a response.

When an access is performed to a non-mapped register in the address range, a SLVERR is provided as a response.

When a read access to write-only registers or a write access to read-only registers is performed, a SLVERR is provided as a response.

When an access is performed to a write-only Privileged register in the address range, a SLVERR is provided as a response.

The phrase ‘SLVERR is provided as a response’ means that the HOST_AXI responds with RRESP = ‘SLVERR’ respective BRESP = ‘SLVERR’.

The error is only reported on the AXI4-Lite protocol, no interrupt is triggered for such issue. The register interface does not provide any access to the L_MEM. It would in charge of the integrator to provide a direct access to the L_MEM to write the relevant data for the MH.

22.5.3.27 Register protection

22.5.3.27.1 Lock mechanism protection

To secure the access to some of the critical registers, an unlock key sequence is required prior to any write-modified access. This procedure must be done before any write to the locked registers. As soon as the write is completed, the register is automatically set back to lock mode.

When an access is performed to a locked register, a SLVERR is provided as a response. The error is only reported on the AXI4-Lite protocol, no interrupt is triggered for such issue.

Two locks are provided for two different purposes:

- A lock is protecting the register in charge of stopping RX and TX FIFO Queues as well as TX Priority Queue slots
- A lock is protecting the MH to be set in debug mode

Functional lock

This sequence is based on three step as defined below:

1. Write 0x1234 to the Ni_LOCK.ULK[15:0] bit-field register
2. Write 0x4321 to the Ni_LOCK.ULK[15:0] bit-field register
3. Write to the unlocked register the expected value

Once the write is performed to the register, it will be automatically back to lock.

The following list of registers are using this protection:

- Ni_TX_FQ_CTRL1
- Ni_TX_PQ_CTRL1
- Ni_RX_FQ_CTRL1

Test mode lock

An unlock key sequence is required to access in write mode the registers assigned to debug and test purpose. This procedure must be done before any write to the locked registers.

This sequence is based on three step as defined below:

1. Write 0x6789 to the Ni_LOCK.TMK[15:0] bit-field register
2. Write 0x9876 to the Ni_LOCK.TMK[15:0] bit-field register
3. Write to the unlocked register the expected value

Once the write is performed to the register, it will be automatically back to a lock state. The only register using this specific key sequence is the Ni_DEBUG_TEST_CTRL register as it does control the debug mode.

22.5.3.27.2 Conditional access protection

Some registers can be written if a bit defined in another register allows the access. As long as this conditional bit is having the right value, any single or consecutive writes can be performed. Configuration registers are protected by such mechanism to avoid any changes while the logic is running.

When an access is performed to a write protected register, a SLVERR is provided as a response. The error is only reported on the AXI4-Lite protocol, no interrupt is triggered for such issue.

The registers with conditional accesses are defined in the following table.

Table 1055 Conditional access register list

Register Name	Condition to write access	Description/Constraints
Ni_MH_CFG	Ni_MH_STS.BUSY = 0	The register can be written if the MH is not running
Ni_MH_SFTY_CFG		
Ni_MH_SFTY_CTRL		
Ni_RX_FILTER_MEM_ADD		
Ni_TX_DESC_MEM_ADD		
Ni_AXI_ADD_EXT		
Ni_AXI_PARAMS		
Ni_TX_FILTER_CTRL0		
Ni_TX_FILTER_CTRL1		
Ni_TX_FILTER_REFVAL0		
Ni_TX_FILTER_REFVAL1		
Ni_TX_FILTER_REFVAL2		
Ni_TX_FILTER_REFVAL3		
Ni_RX_FILTER_CTRL		
Ni_TX_FQ_START_ADD{n}	Ni_TX_FQ_STS0.BUSY[n] = 0	The register can be written if the TX FIFO Queue n is not running (where n is 0-7)
Ni_TX_FQ_SIZE0{n}		
Ni_TX_PQ_START_ADD	Ni_TX_PQ_STS0 = 00000000_H	The register can be written if no TX Priority Queue slots are running
Ni_RX_FQ_START_ADD{n}	Ni_RX_FQ_STS0.BUSY[n] = 00_H	The register can be written if the RX FIFO Queue n is not running (where n is 0-7)
Ni_RX_FQ_SIZE{n}		
Ni_RX_FQ_DC_START_ADD{n}		
Ni_INT_TEST0	Ni_DEBUG_TEST_CTRL.TEST_IRQ_EN = 1	The interrupt lines can be trigger by SW if the interrupt test mode is enable
Ni_INT_TEST1		

22.5.3.27.3 Register CRC computation

To protect the MH configuration, control and configuration registers are protected using CRC. A reference CRC, computed by the SW, is set to a register, and compare with an internal CRC value computed by the MH. It is important to note that some of the registers won't be accessible once the MH is started, Ni_MH_STS.BUSY set to 1, refer to Conditional Access Protection and Lock Mechanism Protection sections for more details.

Once the overall MH setting is done and only when there are no more changes on the control and configuration register, do the following:

- The SW must provide the expected CRC for list of protected registers. The CRC reference value must be computed, only for the registers defined as CRC protected, starting from the lowest address offset. The order of the register to be considered for the CRC computation is defined below (all register values will be checked). The CRC is computed using the 32bit value of the register defined in the list
- Once the 32bit CRC value is computed by SW, it must be written to the Ni_CRC_REG register. The write access to this register does not trig a CRC check
- To check the CRC for the registers, set the Ni_CRC_CTRL.START bit to 1. The MH goes through the list of CRC protected registers and compute the global CRC. After a few cycles, the CRC reference value in the Ni_CRC_REG register is compared with the one already computed. If a CRC error is detected, the REG_CRC_ERR interrupt signal is triggered. As the check is only done and control by SW, there is no enable defined

SW can start a CRC check at regular time interval by setting the Ni_CRC_CTRL.START bit to 1. It is recommended to perform a register CRC check, for any new configuration, to ensure a proper setting before starting the MH.

Here below is the list of registers protected by CRC, in the order they need to be considered (refer to sections in Register Protection chapter for register accessibility):

- Ni_VERSION
- Ni_MH_CFG
- Ni_MH_SFTY_CFG
- Ni_MH_SFTY_CTRL
- Ni_RX_FILTER_MEM_ADD
- Ni_TX_DESC_MEM_ADD
- Ni_AXI_ADD_EXT
- Ni_AXI_PARAMS
- LOOP n from 0 to 7
  - Ni_TX_FQ_START_ADD{n}
  - Ni_TX_FQ_SIZE{n}
- END LOOP
- Ni_TX_PQ_START_ADD
- LOOP n from 0 to 7
  - Ni_RX_FQ_START_ADD{n}
  - Ni_RX_FQ_SIZE{n}
  - Ni_RX_FQ_DC_START_ADD{n}
- END LOOP
- Ni_TX_FILTER_CTRL0 (Privileged)
- Ni_TX_FILTER_CTRL1 (Privileged)
- LOOP n from 0 to 3
  - Ni_TX_FILTER_REFVAL{n} (Privileged)
- END LOOP
- Ni_RX_FILTER_CTRL (Privileged)
- Ni_DEBUG_TEST_CTRL (Privileged)
- Ni_INT_TEST0
- Ni_INT_TEST1

Here below is the normal polynomial representation and implementation of the CRC-32 used to protect the registers:

$$CRC - 32 = \left(x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^{8} + x^{7} + x^{5} + x^{4} + x^{2} + x + 1\right)$$

```
Here below is the pseudo code to compute the CRC for the MH register bank:
The reg_table[] is the array of 32bit registers defined previously (in the order they are
listed):
static logic[31:0] rem32 = 32'hffffffff;
static logic[31:0] rem32_old = 32'hffffffff;
static logic[31:0] poly = 32'h4c11db7;
static logic[31:0] crc32;
// initialize CRC shift register
// This algorithm is indirect
rem32 = 32'hffffffff;
foreach (reg_table[i]) begin
for (int j = 31; j >= 0; j--) begin
// to decide whether reduction with polynomial will be required based on MSB before shift
rem32_old = rem32;
// shift out MSB of CRC
rem32 = rem32 << 1;
rem32[0] = reg_table[i].get()[j];
// perform reduction if required
if (rem32_old[31]) rem32 = rem32 ^ poly;
end
end
// processing 32 0s more
repeat(32) begin
// to decide whether reduction with polynomial will be required based on MSB before shift
rem32_old = rem32;
// shift out MSB of CRC
rem32 = rem32 << 1;
rem32[0] = 0;
// perform reduction if required
if (rem32_old[31]) rem32 = rem32 ^ poly;
end
crc32 = rem32;
```

22.5.3.28 Error and exception handling

Here is the list of potential issues the MH may have to handle and how it should react:

Table 1056 Potential issues of the MH

Error	Source	Interrupt	MH behavior
	MH		

(table continues...)

Table 1056 (continued) Potential issues of the MH

Error	Source	Interrupt	MH behavior
RX acknowledge path overflow	Acknowledge data not sent in time before new one needs to be stored	DP_DO_ERR	The current RX message is discarded and an RX_ABORT_IRQ is triggered to the system. The interrupt DP_DO_ERR is triggered to the system and the Ni_ERR_INT_STS.DP_RX_ACK_DO_ERR bit status register is set to 1. The MH finishes its current transactions and stops with Ni_MH_CTRL.BUSY = 0. The SW needs to restart it and the MH will keep going with its current task.
TX acknowledge path overflow	Acknowledge data not sent in time before the new one being stored	DP_DO_ERR	As soon as an acknowledge data locally stored and ready to be send cannot be done (due to some overflow) no new messages will be sent to the PRT. The DP_DO_ERR interrupt is triggered to the system and the Ni_ERR_INT_STS.DP_TX_ACK_DO_ERR bit status register is set to 1. The MH finishes its current transaction and stops with Ni_MH_CTRL.BUSY = 0. The SW needs to restart it and the MH will keep going with its current task.
RX DMA FIFO overflow	The FIFO overflow on the RX path	DP_DO_ERR	The current RX message is discarded and an RX_ABORT_IRQ is sent to the system. The already RX descriptors used are allocated for the next message. No status is sent back to the Header Descriptor. The MH keeps receiving RX message despite this temporary issue. The Ni_ERR_INT_STS.DP_RX_FIFO_DO_ERR bit status register is set to 1.

(table continues...)

Table 1056 (continued) Potential issues of the MH

Error	Source	Interrupt	MH behavior
RX DMA FIFO above threshold while RX filtering in progress	The RX filter has not completed in time to avoid a potential overflow	NONE	The current RX message is sent to the default RX FIFO Queue as backup solution if enable, see the Ni_RX_FILTER_CTRL register. The threshold is defined to provide enough time for the MH to write the message in S_MEM. The RX Header descriptor of that RX message will have its status report bit field set to “message received but not filtered”. The MH keeps receiving RX message.
RX descriptor CRC error when fetched from S_MEM	A CRC error is detected on RX descriptor	DESC_ERR	As the RX descriptor has a CRC error, the related RX FIFO Queue is stopped and the interrupt DESC_ERR is triggered to the system, see Ni_RX_FQ_STS1, Ni_RX_FQ_STS0 registers. The Ni_SFTY_INT_STS.RX_DESC_CRC_ERR bit status register is set to 1. Other RX FIFO Queues would still be running.
Wrong RX descriptor fetched from S_MEM	The expected descriptor is not the one coming back from S_MEM. Several issue on the address could lead to such result	DESC_ERR	As the RX descriptor is not the one expected, the related RX FIFO Queue is stopped and the interrupt DESC_ERR is triggered to the system, see Ni_RX_FQ_STS1, Ni_RX_FQ_STS0 registers. The Ni_SFTY_INT_STS.RX_DESC_REQ_ERR bit status register is set to 1. Other RX FIFO Queues would still be running.
TX descriptor CRC error when fetched from S_MEM	A CRC error is detected on TX descriptor when fetched from S_MEM	DESC_ERR	As the TX descriptor has a CRC error, the related TX FIFO Queue is stopped and the interrupt DESC_ERR is triggered to the system. The Ni_SFTY_INT_STS.TX_DESC_CRC_ERR bit status register is set to 1. Other TX FIFO Queues would still be running. TX priority Queue are managed differently. If an issue occurs on the TX descriptor the slot will have its busy and sent flags set to 0, see Ni_TX_PQ_STS0 and Ni_TX_PQ_STS1 registers.

(table continues...)

Table 1056 (continued) Potential issues of the MH

Error	Source	Interrupt	MH behavior
Wrong TX descriptor fetched from S_MEM	The expected descriptor is not the one coming back from S_MEM. Several issue on the address could lead to such result	DESC_ERR	As the TX descriptor is not the one expected, the related TX FIFO Queue is stopped and the interrupt DESC_ERR is triggered to the system. The Ni_SFTY_INT_STS.TX_DESC_REQ_ERR bit status register is set to 1. Other TX FIFO Queues would still be running. TX priority Queue are managed differently. If an issue occurs on the TX descriptor the slot will have its busy and sent flags set to 0, see Ni_TX_PQ_STS0 and Ni_TX_PQ_STS1 registers.
Wrong TX descriptor fetched from L_MEM	The expected descriptor is not the one coming back from L_MEM. Several issue on the address could lead to such result	DESC_ERR	The TX descriptor selected to be the next message candidate is corrupted. Either the related TX FIFO Queue is stopped (see Ni_TX_FQ_STS0 and Ni_SFTY_INT_STS registers) or the TX Priority Queue slot is set disable (busy flag set to 0), see Ni_TX_PQ_STS0 and Ni_SFTY_INT_STS registers). The interrupt DESC_ERR is triggered to the system. The Ni_SFTY_INT_STS.TX_DESC_REQ_ERR bit status register is set to 1. Other TX FIFO Queues would still be running as well as TX Priority Queue slots.
Parity error detected on TX address pointers	One of the address pointers managing the TX FIFO Queues or the TX Priority Queue is corrupted	AP_PARITY_ERR	As the source of parity issue, could lead to wrong memory accesses, the MH stops. The MH finishes all pending data transfers and then stops with Ni_MH_STS.BUSY = 0. The SW is notified through the AP_PARITY_ERR interrupt and the Ni_SFTY_INT_STS.AP_TX_PARITY_ERR status bit register is set to 1.

(table continues...)

Table 1056 (continued) Potential issues of the MH

Error	Source	Interrupt	MH behavior
Parity error detected on RX address pointers	One of the address pointers managing the RX FIFO Queues is corrupted	AP_PARITY_ERR	As the source of parity issue, could lead to wrong memory accesses, the MH stops. The MH finishes all pending data transfers and then stops with Ni_MH_STS.BUSY = 0. The SW is notified through the AP_PARITY_ERR interrupt and the Ni_SFTY_INT_STS.AP_RX_PARITY_ERR status bit register is set to 1.
Register CRC error	One of the configuration register protected by CRC is corrupted	REG_CRC_ERR	There is no way to define which register is corrupted and to evaluate which part of the logic would be impacted. The MH is stopped. When receiving an RX message, the current message is discarded and all RX FIFO Queues are stopped. When transmitting a message, it is aborted. All TX FIFO Queues are stopped and all TX Priority Queue slot are disabled. The interrupt REG_CRC_ERR is sent to system
TX data path sequence error	Any error sequence detected on the TX_MSG interface	DP_SEQ_ERR	If any code word reported by the PRT does not match the expected sequence the PRT and MH are no more synchronized. The MH finishes all pending data transfers and then stops with Ni_MH_STS.BUSY = 0. the DP_SEQ_ERR interrupt is triggered with the Ni_ERR_INT_STS.DP_TX_SEQ_ERR bit status register set to 1.
RX data path sequence error	Any error sequence detected on the RX_MSG interface	DP_SEQ_ERR	If any code word reported by the PRT does not match the expected sequence the PRT and MH are no more synchronized. The MH finishes all pending data transfers and then stops with Ni_MH_STS.BUSY = 0. the DP_SEQ_ERR interrupt is triggered with the Ni_ERR_INT_STS.DP_RX_SEQ_ERR bit status register set to 1.

(table continues...)

Table 1056 (continued) Potential issues of the MH

Error	Source	Interrupt	MH behavior
RX frame reception in progress when receiving a new RX message	Due to a high number of filter elements and a high latency on the memory, the RX filtering process may cover almost the shortest CAN frame, leading to an overlap on the current and new RX messages	RX_ABORT_IRQ	As the current RX message has not complete prior receiving the next frame, the new frame is discarded to provide the remaining time to complete the process on the current one. Therefore, any new RX message is aborted with an RX_ABORT_IRQ interrupt.
RX filter not done in time before a new RX frame	The RX filter does not complete in time its process to identify the RX FIFO Queue	RX_FILTER_ERR	When the RX filter is taking too much time and a new RX message is coming, the RX_FILTER_ERR interrupt is triggered. The new RX message is discarded, see Ni_RX_FILTER_CTRL register. The MH keeps running on its current frame. Such interrupt is a good indicator for SW to identify large RX filtering time on some frames.
RX FIFO Queue not enable for reception	The RX FIFO Queue selected to receive the RX message is not running, either not set or wrongly set	RX_ABORT_IRQ	The selected RX FIFO Queue defined after the RX filtering process is disable. The MH discards the RX message with the RX_ABORT_IRQ interrupt. Every RX message going to this disabled RX FIFO Queue will trigger this interrupt. The SW must ensure RX FIFO Queues are enable at first time.
	The RX Filter cannot send message to the RX FIFO Queue as it is disable	RX_FILTER_ERR	In case the RX Filter identifies an RX FIFO Queue to receive an RX message but this queue is disable, the RX_FILTER_ERR interrupt is triggered and the message is rejected
TX message rejected by TX filter	The Header Descriptor is filtered to ensure only well-defined TX message can go through	TX_FILTER_IRQ	When a TX message is rejected it will be skipped by the MH. When the Header descriptor is in a TX FIFO Queue, the next message in the FIFO is used instead. An acknowledge is sent to the TX descriptor with the status rejected. Regarding TX Priority Queue, the corresponding slot is disabled. The MH keeps running all other TX FIFO Queues or slots defined as valid

(table continues...)

Table 1056 (continued) Potential issues of the MH

Error	Source	Interrupt	MH behavior
DMA channel interface mixed up	Wrong data sent or received to DMA channel interface detected by the DMA	DMA_CH_ERR	As such issue would mean data are mixed up, there is no way to recover. The MH finishes all pending data transfers and then stops with Ni_MH_STS.BUSY = 0. The system is notified through the DMA_CH_ERR interrupt line. There is no status flag assigned to such interrupt as the DMA channel being faulty cannot be identified
Parity error on RX message data	A bit flip is detected on data from the RX_MSG to the AXI system bus interface	DP_PARITY_ERR	If such issue occurs while receiving data, the RX message would be discarded. No acknowledge data is sent. An interrupt DP_PARITY_ERR is triggered. The Ni_SFTY_INT_STS.DP_RX_PARITY_ERR bit status register is set to 1. As the RX message would be aborted, the RX_ABORT_IRQ interrupt would also be set. The MH keeps going with new messages.
Parity error on TX message data	A bit flip is detected on data from the AXI system bus interface to the TX_MSG	DP_PARITY_ERR	If such issue occurs while transmitting data, the TX message would be aborted. An interrupt DP_PARITY_ERR would be triggered. The Ni_SFTY_INT_STS.DP_TX_PARITY_ERR bit status register is set to 1. The TX_ABORT_IRQ interrupt is set.
Parity error on RX message acknowledge data	A bit flip is detected on data from the RX_MSG to the AXI system bus interface	DP_PARITY_ERR	If such issue occurs on the acknowledge data, the RX message would be discarded. No acknowledge data is sent. An interrupt DP_PARITY_ERR is triggered. The Ni_SFTY_INT_STS.ACK_RX_PARITY_ERR bit status register is set to 1. The MH keeps going with new messages.

(table continues...)

Table 1056 (continued) Potential issues of the MH

Error	Source	Interrupt	MH behavior
Parity error on TX message acknowledge data	A bit flip is detected on payload data from the AXI system bus interface to the TX_MSG	DP_PARITY_ERR	If such issue occurs while acknowledging the TX message. An interrupt DP_PARITY_ERR is triggered. The Ni_SFTY_INT_STS.ACK_TX_PARITY_ERR bit status register is set to 1. The SW can identify such issue reading the report status of that TX descriptor.
RX message received while MH not started	Ni_MH_CTRL.START bit wrongly set to 1	NONE	The MH does not accept RX message data from the PRT. As the PRT cannot sent data to the MH, a data overflow on the PRT will occur leading to an interrupt.
PRT			
RX data path overflow	DO code word received from PRT. . Several issues could lead to such issue for example, peak latency preventing write accesses in time or RX path being stopped and so on.	NA	The RX message is discarded. The already used RX descriptor are reused for the next message. Then, no status is sent back to the Header Descriptor in S_MEM.
RX message on CAN bus not successful	ABORT code word received from PRT. Invalid CAN message detected on CAN bus	NA	This is normal behavior. The RX message is discarded. The already used RX descriptors are allocated for the next message. No acknowledge data is sent back to the S_MEM

(table continues...)

Table 1056 (continued) Potential issues of the MH

Error	Source	Interrupt	MH behavior
TX data path underrun	DU code word received from PRT. TX message data not provided in time	NA	The current TX message, selected and started on the MH side, is aborted but the PRT keeps going with its current frame and will generated a wrong CRC to invalidate the frame at the receiver side. All data transfers from S_MEM is aborted. The issue may be the result of a peak latency. The TX message is still valid and will be part of the next TXScan. The MH can restart to transmit the same message according to the restart counter setting. The PRT is triggered an interrupt to the system when such code word is transmitted to the MH. It is essential to understand that the MH will still be active and fully functional. There is no message loss when such issue occurs
TX message on CAN bus not successful	RESTART code word received from PRT	NA	The current TX message, selected and started on the MH side, is aborted. All data transfers from S_MEM is aborted. The current TX message is still valid and will be part of the next TX-Scan. The MH can restart to transmit the same message according to the restart counter setting or use another one with highest priority.
TX message header invalid	HFI code word received from PRT	NA	The current TX message is discarded, and a data acknowledge is sent back to the Header Descriptor in S_MEM. The report status of the TX descriptor is updated with the issue. If the TX message was in a TX FIFO Queue, the MH keeps running and skips this TX message to fetch the next one. In case of a TX Priority Queue slot, the slot is set as done but not sent (see report status in TX descriptor)

(table continues...)

Table 1056 (continued) Potential issues of the MH

Error	Source	Interrupt	MH behavior
Unexpected Start Of Sequence (USOS) at the TX_MSG interface	When PRT detects USOS, it stops CAN protocol operation and sets ENABLE=0	STOP_IRQ	In case such code word is received, the MH or the PRT are no more synchronized. The MH finishes its current transfers and stops. The STOP_IRQ interrupt is set to notify MH is no more active. In such scenario the only action would be to reset the MH and PRT to recover
PRT entered CAN protocol's Bus-Off state	PRT stops CAN protocol operation and sets ENABLE=0	STOP_IRQ	MH finishes all pending data transfers and then stops (put on hold). All FSM in the MH go to idle. The STOP_IRQ interrupt is set to notify MH is no more active. A write to the Ni_MH_CTRL.START bit register allows the SW to restart everything at the point it was stopped, if required.
PRT stopped by SW	PRT stops CAN protocol operation and sets ENABLE=0	STOP_IRQ	
PRT TX_MSG interface not responding	PRT is having a deadlock and cannot answer to MH request or receive data	DP_TO_ERR	When the timeout assigned to the TX_MSG interface fires, the MH finishes all pending data transfers and then stops with Ni_MH_STS.BUSY = 0. The DP_TO_ERR interrupt is triggered to the system.with the Ni_ERR_INT_STS.DP_TX_TO_ERR bit status set to 1
PRT RX_MSG interface not responding	PRT is having a deadlock and cannot send data to MH	DP_TO_ERR	When the timeout assigned to the RX_MSG interface fires, the MH finishes all pending data transfers and then stops with Ni_MH_STS.BUSY = 0. The DP_TO_ERR interrupt is triggered to the system with the Ni_ERR_INT_STS.DP_RX_TO_ERR bit status set to 1
LOCAL MEMORY (L_MEM)			

(table continues...)

Table 1056 (continued) Potential issues of the MH

Error	Source	Interrupt	MH behavior
Local memory safety error while reading RX filter element. Corrupted data has been corrected	The L_MEM is providing a safety error on MEM_SFTY_CE input signal while reading a data	MEM_SFTY_ERR	As the corrupted data word is corrected, the RX filtering can be done on the current RX message. The MH keeps running and will be able to receive new messages. The interrupt MEM_SFTY_ERR is triggered to the system with the Ni_SFTY_INT_STS.MEM_SFTY_CE bit status register set to 1. It is essential for such issue that there is no error response on the memory interface while reading the corrected data.
Local memory safety error while reading TX descriptor. Corrupted data has been corrected	The L_MEM is providing a safety error on MEM_SFTY_CE input signal while reading a data	MEM_SFTY_ERR	As the TX descriptor selected to be the next message candidate is corrupted but corrected, the related TX FIFO Queue or the TX Priority Queue slot will run as normal. When the TX-Scan is reading a corrected TX descriptor, the all process will complete. The interrupt MEM_SFTY_ERR is triggered to the system with the Ni_SFTY_INT_STS.MEM_SFTY_CE bit status register set to 1. It is essential for such issue that there is no error response on the memory interface while reading the corrected data.
Local memory safety error while reading RX filter element. Corrupted data is not corrected	The L_MEM is providing a safety error on MEM_SFTY_UE input signal while reading a data with SLVERR response	MEM_SFTY_ERR	As no more filtering can be done on the current RX message, it is discarded. As it is not possible to keep going with a corrupted RX filter element, the MH stops The interrupt MEM_SFTY_ERR is triggered to the system with the Ni_SFTY_INT_STS.MEM_SFTY_UE bit status register set to 1. The MH finishes all pending data transfers and then stops with Ni_MH_STS.BUSY = 0. It is essential for such issue, to have the memory interface reporting a SLVERR when reading the corrupted data.

(table continues...)

Table 1056 (continued) Potential issues of the MH

Error	Source	Interrupt	MH behavior
Local memory safety error while reading TX descriptor. Corrupted data is not corrected	The L_MEM is providing a safety error on MEM_SFTY_UE input signal while reading a data with a SLVERR response	MEM_SFTY_ERR	As it is not possible to keep going with a corrupted TX descriptor, the MH stops The interrupt MEM_SFTY_ERR is triggered to the system if such issue occurs with the Ni_SFTY_INT_STS.MEM_SFTY_UE bit status register set to 1. The MH finishes all pending data transfers and then stops with Ni_MH_STS.BUSY = 0. It is essential for such issue, to have the memory interface reporting a SLVERR when reading the corrupted data.
Error response received on local memory write access. A safety issue is not considered here	The L_MEM is providing a DECERR/SLVERR error response on BRESP[1:0] for a write access	RESP_ERR[0]	The TX descriptor written to the L_MEM is not valid. In such cases as the L_MEM cannot be trusted anymore, the MH stops. The MH finishes all pending data transfers and then stops with the Ni_MH_STS.BUSY = 0. The RESP_ERR[0] interrupt is triggered to the system. To identify the issue the BRESP[1:0] and the ID of the transaction are logged in the Ni_AXI_ERR_INFO.MEM_ID[1:0] and Ni_AXI_ERR_INFO.MEM_RESP[1:0] bit status register.
Error response received on local memory read access. A safety issue is not considered here	The L_MEM is providing a DECERR/SLVERR error response on RRESP[1:0] for a read access	RESP_ERR[1]	The TX descriptor or RX Filter element read from the L_MEM is not valid. In such cases as the L_MEM cannot be trusted anymore, the MH stops. The MH finishes all pending data transfers and stops with the Ni_MH_STS.BUSY = 0. The RESP_ERR[0] interrupt is triggered to the system. To identify the issue the RRESP[1:0] and the ID of the transaction are logged in the Ni_AXI_ERR_INFO.MEM_ID[1:0] and Ni_AXI_ERR_INFO.MEM_RESP[1:0] bit status register.

(table continues...)

Table 1056 (continued) Potential issues of the MH

Error	Source	Interrupt	MH behavior
A read from the L_MEM cannot complete	The MH does not complete a read within a defined time frame	MEM_TO_ERR	When the timeout assigned to the L_MEM AXI read channel fires, the MH finishes all pending data transfers and then stops with Ni_MH_STS.BUSY = 0. The MEM_TO_ERR interrupt is triggered to the system and the Ni_SFTY_INT_STS.MEM_AXI_RD_TO_ERR bit status is set to 1
A write to the L_MEM cannot complete	The MH does not complete a write within a defined time frame	MEM_TO_ERR	When the timeout assigned to the L_MEM AXI write channel fires, the MH finishes all pending data transfers and then stops with Ni_MH_STS.BUSY = 0. The MEM_TO_ERR interrupt is triggered to the system and the Ni_SFTY_INT_STS.MEM_AXI_WR_TO_ERR bit status is set to 1
System			
Address decoding error on DMA write channels	Error response from AXI system bus interface, DECERR received on BRESP[1:0] for write access	RESP_ERR[0]	When the error is detected on the RX message data or acknowledge data being written, the interrupt RESP_ERR[0] interrupt is sent to the system. As the S_MEM is not reliable, the MH stops. The MH finishes all pending data transfers and stops with the Ni_MH_STS.BUSY = 0. To identify the issue the BRESP[1:0] and the ID of the transaction are logged in the Ni_AXI_ERR_INFO.DMA_ID[1:0] and Ni_AXI_ERR_INFO.DMA_RESP[1:0] bit status register.
Address decoding error on DMA read channels	Error response from AXI system bus interface, DECERR received on RRESP[1:] for read access	RESP_ERR[1]	When the error is detected on the TX message data, RX or TX descriptors, the interrupt RESP_ERR[1] is sent to the system. As the S_MEM is not reliable, the MH stops. The MH finishes all pending data transfers and stops with the Ni_MH_STS.BUSY = 0. To identify the issue the RRESP[1:0] and the ID of the transaction are logged in the Ni_AXI_ERR_INFO.DMA_ID[1:0] and Ni_AXI_ERR_INFO.DMA_RESP[1:0] bit status register.

(table continues...)

Table 1056 (continued) Potential issues of the MH

Error	Source	Interrupt	MH behavior
System memory CRC error or Access to wrong slave on DMA write channel	Error response from AXI system bus interface, SLVERR received on BRESP[1:0] for write access	RESP_ERR[0]	There is no way to identify the exact error source, either a CRC error or a wrong slave access. See “Address decoding error on DMA write channels” description in this table
System memory CRC error or Access to wrong slave on DMA read channel	Error response from AXI system bus interface, SLVERR received on RRESP[1:0] for read access	RESP_ERR[1]	There is no way to identify the exact source, either a CRC error or a wrong slave access. See “Address decoding error on DMA read channels” description in this table
A read from the S_MEM cannot complete	The MH does not complete a read within a defined time frame	DMA_TO_ERR	When the timeout assigned to the S_MEM AXI read channel fires, the MH finishes all pending data transfers and then stops with Ni_MH_STS.BUSY = 0. The DMA_TO_ERR interrupt is triggered to the system with the Ni_SFTY_INT_STS.DMA_AXI_RD_TO_ERR bit status set to 1
A write to the S_MEM cannot complete	The MH does not complete a write within a defined time frame	DMA_TO_ERR	When the timeout assigned to the S_MEM AXI write channel fires, the MH finishes all pending data transfers and then stops with Ni_MH_STS.BUSY = 0. The DMA_TO_ERR interrupt is triggered to the system with the Ni_SFTY_INT_STS.DMA_AXI_WR_TO_ERR bit status set to 1

22.5.3.29 Interrupts

Table 1057 Interrupts

Interrupt	Description
TX_FQ_IRQ[7:0]	When considering TX FIFO Queues there is a mean, thanks to the IRQ bit-field in TX descriptor, to trigger an interrupt to the system when a TX message is sent successfully or skipped. This interrupt can only be declared in a TX Header Descriptor (HD=1). When a TX descriptor has HD=0 and IRQ=1, no interrupt is generated A dedicated interrupt signal TX_FQ_IRQ[N] is provided per TX FIFO queue n (0 <= n <= 7). When a TX Header Descriptor is mentioning an interrupt (IRQ bit set to 1) and the message is successfully sent or skipped, the DESC_MESSAGE_HANDLER identifies the TX FIFO queue source number of that descriptor and triggers the relative line of the interrupt bus signal. When one or several TX descriptors are used for one TX message, the SW needs to define the interrupt (IRQ bit-field set to 1) only to the Header Descriptor. The interrupt will be effective only when the acknowledge data of that descriptor is fully written in S_MEM. It is then possible to define for a TX FIFO Queue n, having a fix number of messages to be sent, the interrupt TX_FQ_IRQ[N] only to the last Header Descriptor. Doing so, this approach will limit the number of interrupts to the system. The main purpose of the TX FIFO Queue is to append on the fly new messages. A race condition may occur between the SW and the Message Handler regarding the definition of valid TX message in that queue. In case a TX FIFO Queue n does not provide a valid TX descriptor, the MH notifies the SW with the TX_FQ_IRQ[n] that the TX FIFO Queue n is on hold, despite being active. The Ni_TX_FQ_INT_STS register provides the relevant information to detect the root cause. As a summary two different source of events can trig those interrupts: • This interrupt is triggered when the IRQ bit field in TX Header Descriptor is set to 1 and the TX message is sent successfully. The Ni_TX_FQ_INT_STS.SENT[n] bit register is set to 1 for the TX FIFO Queue n and the bit field STS[3:0] in the TX descriptor is set to 0’b0001 • This interrupt trigs when the TX message is skipped. The Ni_TX_FQ_INT_STS.SENT[n] bit register is set to 1 for the TX FIFO Queue n and the bit field STS[3:0] in the TX descriptor is set to 0’b0010 or 0’b0011 • The TX FIFO Queue n execution is stopped due to the fetch of an invalid TX descriptor in this queue (no more TX message defined, and no END bit set to 1 for the last TX message). The Ni_TX_FQ_INT_STS.UNVALID[n] bit register is set to 1 for the TX FIFO Queue n

(table continues...)

Table 1057 (continued) Interrupts

Interrupt	Description
RX_FQ_IRQ[7:0]	When considering RX FIFO Queues there is the option, thanks to the IRQ bit field in RX descriptor, to trigger an interrupt to the system when an RX message is received successfully. The interrupt bus signal RX_FQ_IRQ[n] provides an interrupt line for the RX FIFO queue n (0 <= n <=7). When the DESC_MESSAGE_HANDLER fetches an RX descriptor for a given RX message and identifies an IRQ bit set to 1 in one of them, it stores this information. Once the RX message is received successfully and a IRQ bit set has been detected in one RX descriptor, an interrupt is triggered. This interrupt is triggered only when the acknowledge data (written in the Header Descriptor) is fully written in the S_MEM. As a summary there are two options to define this interrupt bit in RX descriptors: • In case the SW requires an interrupt per RX message, the IRQ bit in all RX descriptors must be set to 1. This setting is valid for Normal and Continuous mode with the same effect. • The SW can set the IRQ bit in a regular interval along an RX FIFO Queue, avoiding interrupts at every RX message. Only the RX message covering the RX descriptor having this IRQ bit set will trigger an interrupt. In Continuous mode, it is then possible to set an interrupt every two, three or N messages. In Normal mode, the interrupt could be defined every two, three or N RX descriptors According to the RX message size, several RX descriptors will be used and so could trig the interrupt. It is important to note that RX messages are received with various bit rate, thus the interrupt time interval will not be identical. A race condition may occur between the SW and the Message Handler regarding the definition of valid RX descriptor in a queue. In case an RX FIFO Queue n does not provide a valid RX descriptor in time, the interrupt notifies the SW with the RX_FQ_IRQ[n] interrupt that the RX FIFO Queue n is on hold despite being active. The Ni_RX_FQ_INT_STS register provides the related information to identify the root cause. As a summary two different source of events can trig those interrupts: • This interrupt is triggered when the IRQ bit field in an RX Descriptor is set to 1 and the RX message is received successfully. The Ni_RX_FQ_INT_STS.RECEIVED[n] bit register is set to 1 for the RX FIFO Queue n and the bit field STS[3:0] in the RX descriptor is set to 0’b0001 • The RX FIFO Queue n execution is stopped due to the fetch of an invalid RX descriptor in this queue. The Ni_RX_FQ_INT_STS.UNVALID[n] bit register is set to 1 for the TX FIFO Queue n

(table continues...)

Table 1057 (continued) Interrupts

Interrupt	Description
TX_PQ_IRQ	A single TX_PQ_IRQ interrupt is assigned to all TX Priority Queue slots. Any TX Priority Queue slot can trigger an interrupt (IRQ = 1) when the relative TX message is successfully sent or skipped. Any TX Header Descriptor having the IRQ bit set and used by the TX Priority Queue trigs this interrupt line. As for the TX FIFO Queue, when a TX message is defined in a TX Priority Queue slot, all the TX descriptors used to define this message must be valid. When the message is sent, the slot is set to inactive and nothing else can occur. Whereas the TX FIFO Queue, which is processing up to the point a TX descriptor is invalid, the TX Priority Queue slot must not fetch any invalid descriptor. To protect the execution of TX message and to have a common TX Queue management, the TX priority Queue can also report invalid descriptor. The SW need to look at the interrupt status register Ni_TX_PQ_INT_STS0 and Ni_TX_PQ_INT_STS1 to identify which slot has generated the interrupt and for which reason. As a summary three different source of events can trig this interrupt: • This interrupt is triggered when the IRQ bit field in TX Header Descriptor is set to 1 and the TX message is sent successfully. The Ni_TX_PQ_INT_STS0.SENT[n] bit register is set to 1 for the TX Priority Queue slot n and the bit field STS[3:0] in the TX descriptor is set to 0’b0001 • This interrupt is triggered when the TX message is skipped. The Ni_TX_PQ_INT_STS0.SENT[n] bit register is set to 1 for the TX Priority Queue slot n and the bit field STS[3:0] in the TX descriptor is set to 0’b0010 or 0’b0011 • The TX Priority Queue slot n execution is stopped due to the fetch of an invalid TX descriptor in this queue (TX descriptor is not valid). The Ni_TX_PQ_INT_STS1.UNVALID[n] bit register is set to 1 for the TX Priority Queue slot n
STATS_IRQ	Four Statistic counters are used to monitor successful and unsuccessful RX and TX messages. As soon as one of those counters overflows the STATS_IRQ is triggers to the system, refer to the RX and TX Statistics chapter for more details. When looking at the Ni_STATS_INT_STS register, the SW can identify which counter has reached its maximum value: • When the number of unsuccessful RX message received has reached the maximum counter value, the Ni_STATS_INT_STS.RX_UNSUCC is set to 1 • When the number of successful RX message received has reached the maximum counter value, the Ni_STATS_INT_STS.RX_SUCC is set to 1 • When the number of unsuccessful TX message received has reached the maximum counter value, the Ni_STATS_INT_STS.TX_UNSUCC is set to 1 • When the number of successful TX message received has reached the maximum counter value, the Ni_STATS_INT_STS.TX_SUCC is set to 1

(table continues...)

Table 1057 (continued) Interrupts

Interrupt	Description
STOP_IRQ	When the PRT is stop (ENABLE signal goes from high to low), the MH finishes its current tasks. It puts all active RX/TX FIFO Queues on hold and discard all active TX Priority Queue slots. Once done, the MH notifies such state by triggering the STOP_IRQ interrupt. The interrupt STOP_IRQ is raised under the following conditions: • Ni_TX_FQ_STS0 = 0x0000 and Ni_RX_FQ_STS0 = 0x0000 and Ni_TX_PQ_STS0 = 0x00000000 • Ni_TX_FQ_STS0 = 0xXYXY and Ni_RX_FQ_STS0 = 0xWVWV and Ni_TX_PQ_STS0 = 0x00000000, where XY defined the active and on hold TX FIFO Queues and WV the active and on hold RX FIFO Queues
RX_FILTER_IRQ	In order to track RX filtering results, an interrupt can be defined when a match is detected on any defined RX filter element. The RX_FILTER_IRQ can only be triggered if the IRQ bit in the RX filter element is set to 1 AND there is a match. When a match is detected, the FM bit (set in RX message header) is set to 1 and the filter element index is defined in the FIDX[7:0] bit field (set in the RX message header). ***Note**: The BLK bit field in the RX Filter element is a side band information and is not considered for the interrupt generation*
TX_FILTER_IRQ	The interrupt is triggered when the TX filter is enabled, and a TX message is rejected. Despite being rejected, the TX descriptor used to define the TX message is acknowledged. To identify the TX descriptor allocated to the TX message rejected, the STS[3:0] bit field in the TX descriptor is set to 0'b0100. The Ni_TX_FILTER_ERR_INFO register provides the relevant information to identify which TX FIFO Queue or TX Priority Queue slot is impacted.
TX_ABORT_IRQ	This interrupt line is only triggered when the MH needs to abort a TX message being sent to the PRT. This interrupt does not have any status flags, as it will always be linked to functional or safety errors. Thus, another interrupt will provide the require information related to the issue. Several source of events can lead to this interrupt: • TX address pointer parity error (refer to AP_PARITY_ERR interrupt) • Timeout on S_MEM, L_MEM or PRT interface (refer to MEM_TO_ERR, DMA_TO_ERR or DP_TO_ERR interrupt) • DMA channel routing error (refer to DMA_CH_ERR interrupt) • A TX_MSG sequence error (refer to DP_SEQ_ERR interrupt) • A DMA AXI or MEM AXI error response (refer to RESP_ERR interrupt) • An uncorrectable error detected on the L_MEM (refer to MEM_SFTY_ERR interrupt) • A TX data parity error (refer to DP_PARITY_ERR interrupt) Aborting a TX FIFO Queue or a TX Priority Queue slot does not set this interrupt as no TX message abort is expected to occur (the MH will complete the current TX message before aborting the TX FIFO Queue).

(table continues...)

Table 1057 (continued) Interrupts

Interrupt	Description
RX_ABORT_IRQ	This interrupt line is triggered when the MH needs to abort an RX message is received from PRT. This interrupt does not have any status flags, as it will always be linked to functional or safety errors. Thus, another interrupt will provide the require information related to the issue. As a summary two different source of events can trig this interrupt: • An RX message is about to be sent to a disabled RX FIFO Queue. • An RX message is in progress and the MH receives a new RX message at the same time. If the RX filter has already completed or if the threshold in the RX DMA FIFO is reached, the new one is aborted to leave more time for the current one to complete. • RX address pointer parity error (refer to AP_PARITY_ERR interrupt) • Timeout on S_MEM, L_MEM or PRT interface (refer to MEM_TO_ERR, DMA_TO_ERR or DP_TO_ERR interrupt) • DMA channel routing error (refer to DMA_CH_ERR interrupt) • An RX_MSG sequence error (refer to DP_SEQ_ERR interrupt) • A DMA AXI or MEM AXI error response (refer to RESP_ERR interrupt) • An uncorrectable error detected on the L_MEM (refer to MEM_SFTY_ERR interrupt) • An RX data parity error (refer to DP_PARITY_ERR interrupt) • An RX descriptor error (refer to DESC_ERR interrupt) • An overflow on RX DMA FIFO or on the RX descriptor acknowledge path (refer to the DP_DO_ERR interrupt) Aborting an RX FIFO Queue will never set this interrupt, as the MH will complete its current reception before this action.
RX_FILTER_ERR	This interrupt line is triggered when the RX filter has not finished in time, to define the RX FIFO Queue number, before the reception of a new RX message. It provides information to the SW about large RX filtering time. Refer to the RX Filter chapter for detailed description. There is no status flag related to this interrupt, as the second source of event, defined below, is a programming issue and should never occur. Two different sources of events can trig this interrupt: • RX filtering not finished before a new RX frame • RX FIFO Queue to receive RX frame not running

(table continues...)

Table 1057 (continued) Interrupts

Interrupt	Description
MEM_SFTY_ERR	Safety error detected at the L_MEM interface. In fact, this interrupt is triggered when either the MEM_SFTY_CE or MEM_SFTY_UE input signal is active. To identify the root cause of such interrupt, refer to the Ni_SFTY_INT_STS register Two different sources of events can trigger this interrupt: • The MEM_SFTY_UE input signal, when set, to indicate an uncorrectable error from the L_MEM when reading (this signal must be generated by the L_MEM memory controller). The Ni_SFTY_INT_STS. MEM_SFTY_UE bit register is set to 1 in this case • The MEM_SFTY_CE input signal, when set, to indicate a correctable error from the L_MEM when reading (this signal must be generated by the L_MEM memory controller). The Ni_SFTY_INT_STS. MEM_SFTY_CE bit register is set to 1 in this case
REG_CRC_ERR	CRC error detected on the register bank. This interrupt is triggered after a few cycles if the CRC written in the Ni_CRC_REG.VAL[31:0], prior writing 1 to the Ni_CRC_CTRL.START bit, is not matching the one computed in hardware. Such interrupt event does not trig any actions in the MH. Therefore, it is a SW task to do the appropriate actions to stop the MH.
DESC_ERR	CRC error detected on RX/TX descriptor or unexpected RX/TX descriptor received. Status flags allow SW to identify the root cause of such interrupt, see Ni_SFTY_INT_STS register. Several source issues could lead to this interrupt: • When the Ni_SFTY_INT_STS.RX_DESC_CRC_ERR is set to 1, an RX descriptor is received and is having a CRC error • When the Ni_SFTY_INT_STS.RX_DESC_REQ_ERR is set to 1, an RX descriptor is received and is not compliant to the one requested (wrong RX FIFO Queue, wrong instance number, wrong position in the queue, …) • When the Ni_SFTY_INT_STS.TX_DESC_CRC_ERR is set to 1, a TX descriptor is received and is having a CRC error • When the Ni_SFTY_INT_STS.TX_DESC_REQ_ERR is set to 1, a TX descriptor is received and is not compliant to the one requested (wrong TX FIFO Queue, wrong instance number, wrong position in the queue, wrong TX Priority Queue slot…) The Ni_DESC_ERR_INFO0 and Ni_DESC_ERR_INFO1 registers provide a detailed description of the faulty RX/TX descriptor. Only the first RX/TX descriptor error will lead to an update of those registers, in case several ones occur. To capture the next descriptor error information, the SW must clear the interrupt source.

(table continues...)

Table 1057 (continued) Interrupts

Interrupt	Description
AP_PARITY_ERR	Address pointers used to manage TX FIFO Queues, RX FIFO Queues and TX Priority Queue are protected using parity bit (1bit per byte). Any issue detected trigs this interrupt. The parity bits are checked only when the address pointer is used for S_MEM accesses. Status flags allow SW to identify the root cause, see Ni_SFTY_INT_STS register. Several source issues could lead to this interrupt: • When the Ni_SFTY_INT_STS.AP_RX_PARITY_ERR is set to 1, an address pointer used to manage the RX path is having a parity error • When the Ni_SFTY_INT_STS.AP_TX_PARITY_ERR is set to 1, an address pointer used to manage the TX path is having a parity error
DP_PARITY_ERR	Parity error detected on RX message data received from PRT to AXI system bus or TX payload data transmitted from AXI system bus to PRT. Any issue detected trigs this interrupt. Status flags allow SW to identify the root cause, see Ni_SFTY_INT_STS register. Several source issues could lead to this interrupt: • When the Ni_SFTY_INT_STS.DP_RX_PARITY_ERR is set to 1, an RX message data is having a parity error • When the Ni_SFTY_INT_STS.DP_TX_PARITY_ERR is set to 1, a TX message data is having a parity error
DP_SEQ_ERR	The RX_MSG or TX_MSG interface used to synchronize the MH and PRT data exchange is not functional. A wrong PRT or MH behavior could lead to this issue. A problem on the logic managing the clock domain crossing on RX_MSG or TX_MSG interface may be one of the source issues. Status flags are available to identify the faulty interface, see Ni_ERR_INT_STS register. Several source issues could lead to this interrupt: • When the Ni_ERR_INT_STS.DP_RX_SEQ_ERR is set to 1, an issue is detected on the RX_MSG interface • When the Ni_ERR_INT_STS.DP_TX_SEQ_ERR is set to 1, an issue is detected on the TX_MSG interface

(table continues...)

Table 1057 (continued) Interrupts

Interrupt	Description
DP_DO_ERR	An overflow is detected on the RX data path or while acknowledging an RX/TX descriptor. Some status flags are provided to identify the interrupt source, see Ni_ERR_INT_STS register Several source issue could trigger this interrupt: • When the Ni_ERR_INT_STS.DP_RX_FIFO_DO_ERR is set to 1, an RX DMA FIFO overflow is detected. Several reasons could explain such issue: a very high system latency (over the expected limit considered for the MH), a system memory no more accessible and a wrong MH behavior. • When the Ni_ERR_INT_STS.DP_RX_ACK_DO_ERR is set to 1, an ACK DMA FIFO overflow is detected. Such issue occurs when the acknowledgment of an RX descriptor is not possible due to some pending ones. A system memory not accessible or a wrong MH behavior (DMA_CONTROLLER not functional, deadlock on RX/TX acknowledge path) could explain such issue. • When the Ni_ERR_INT_STS.DP_TX_ACK_DO_ERR is set to 1, an ACK DMA FIFO overflow is detected. Such issue occurs when the acknowledgment of a TX descriptor is not possible due to some pending ones. A system memory not accessible or a wrong MH behavior (DMA_CONTROLLER not functional, deadlock on RX/TX acknowledge path) could explain such issue
DP_TO_ERR	When the PRT is not responding after a certain amount of time, either on RX or on TX path, the DP_TO_ERR interrupt is triggered. The counter on RX_MSG or TX_MSG interface starts with the Start Of Frame and stop when receiving the timestamp. The timeout value is programmable by SW. Some status flags provide the interrupt source, see Ni_SFTY_INT_STS register. Several source issue could trigger this interrupt: • When the Ni_SFTY_INT_STS.DP_PRT_RX_TO_ERR is set to 1, the timeout value defined on the RX_MSG interface is over. The PRT or MH may be locked, preventing data reception • When the Ni_SFTY_INT_STS.DP_PRT_TX_TO_ERR is set to 1, the timeout value defined on the TX_MSG interface is over. The PRT or MH may be locked, preventing data transmission
DMA_TO_ERR	When the S_MEM is not responding after a defined time interval, the DMA_TO_ERR is triggered. The timeout value is programmable by SW. Some status flags provide the interrupt source, see Ni_SFTY_INT_STS register. Several source issue could trigger this interrupt: • When the Ni_SFTY_INT_STS.DMA_AXI_RD_TO_ERR is set to 1, the timeout value defined on the DMA AXI read channel interface is over. A system memory no more accessible or a DMA_CONTROLLER in deadlock could explain such issue. • When the Ni_SFTY_INT_STS.DMA_AXI_WR_TO_ERR is set to 1, the timeout value defined on the DMA AXI write channel interface is over. A system memory no more accessible or a DMA_CONTROLLER in deadlock could explain such issue.

(table continues...)

Table 1057 (continued) Interrupts

Interrupt	Description
MEM_TO_ERR	When the L_MEM is not responding after a defined time interval the MEM_TO_ERR is triggered. The timeout value is programmable by SW. Some status flags are provided to identify the interrupt source, see Ni_SFTY_INT_STS register. Several source issue could trigger this interrupt: • When the Ni_SFTY_INT_STS.MEM_AXI_RD_TO_ERR is set to 1, the timeout value defined on the MEM AXI read channel interface is over. A local memory no more accessible or a memory controller in deadlock could explain such issue. • When the Ni_SFTY_INT_STS.MEM_AXI_WR_TO_ERR is set to 1, the timeout value defined on the MEM AXI write channel interface is over. A local memory no more accessible or a memory controller in deadlock could explain such issue.
DMA_CH_ERR	Data received or sent are not routed to or from the right DMA channels. Such issue will lead to data corruption and wrong MH behavior. There are no status flags to identify the source channel being faulty.
RESP_ERR[1:0]	Any error response from the DMA AXI and MEM AXI interfaces can lead to a RESP_ERR[1:0] interrupts. Some status flags provide the interrupt source, see Ni_SFTY_INT_STS register. Several source issue could trigger those interrupts: • When the RESP_ERR[0] interrupt is set, a write access error is detected on either the DMA_AXI or MEM_AXI write channel. • When the RESP_ERR[1] interrupt is set, a read access error is detected on either the DMA_AXI or MEM_AXI read channel. The Ni_AXI_ERR_INFO register provides a detailed description of the faulty AXI interface, refer to the Ni_AXI_ERR_INFO.MEM_RESP[1:0] or Ni_AXI_ERR_INFO.DMA_RESP[1:0] bit field to determine which one (must be different from 0'b00). The traffic getting the error response is defined when looking at the Ni_AXI_ERR_INFO.MEM_ID[1:0] (if Ni_AXI_ERR_INFO.MEM_RESP[1:0] is different from 0'b00) or Ni_AXI_ERR_INFO.DMA_ID[1:0] bit field (if Ni_AXI_ERR_INFO.DMA_RESP[1:0] is different from 0'b00). In case several response errors occur on the same interface, only the AXI ID of the last one is captured.

22.5.3.30 Local memory map

In order to perform the RX filtering and the TX-SCAN, the MH requires a local memory. This local memory called L_MEM is addressable through the MEM_AXI interface. The MEM_AXI interface is able to address up to 64 KBytes with a 32 bit data bus width.

The L_MEM is storing all the RX filter elements, Header Descriptor for TX FIFO and Priority Queues, as well as a single Trailing Descriptor when considering TX message defined with several descriptors.

22.5.3.30.1 TX descriptors

The TX FIFO Queue descriptors are organized into the L_MEM starting at the base address defined in Ni_TX_DESC_MEM_ADD.fq_base_add[15:0]. Up to 8 memory location, of size $8 \times 32\ bit$, is required to hold the TX Header Descriptor of every TX FIFO Queues.

Every TX FIFO Queue, when active, has its current and next descriptor defined into the L_MEM for the TX-SCAN process. This means, for a given TX FIFO Queue, memory space must be double. The current and the next TX Header Descriptor are used for the TX-SCAN.

The TX Priority Queue descriptors are organized into the L_MEM starting at the base address defined in Ni_TX_DESC_MEM_ADD.PQ_BASE_ADDR[15:0]. Up to 32 memory location, of size $8 \times 32\ bit$, is required to hold the TX Header Descriptors of every slot. As there is only one TX message per slot, there is no need to allocate more space.

When a TX message is split over several descriptors, a temporary memory location is used to hold the Trailing Descriptor to retrieve payload data (see Temporary Descriptor below).

The TX descriptor element are organized in 32 bit word and so any offset would be a multiple of 8. Here below is the memory organization of the TX descriptors considering N TX FIFO Queues and M TX Priority Queue Slots:

Table 1058 Memory organization of the TX descriptors

Memory base address	Offset	Name	Bit field	Description
FQ_BASE_ADD[15:0]	0x0+0x40*n $0x0 + 0x40 \times n$	TX FIFO Queue n (current/next TX Header Descriptor) (0 <= n < N)	Element 0	TX Header Descriptor, see TX descriptor, TX Message and TX FIFO Queue chapters
	0x4+0x40*n $0x4 + 40 \times n$		Element 1	
	$0x8 + 0x40 \times n$		Element 2: TS0	
	$0xC + 0x40 \times n$		Element 3: TS1	
	$0x10 + 0x40 \times n$		Element 4: T0	
	$0x14 + 0x40 \times n$		Element 5: T1	
	$0x18 + 0x40 \times n$		Element 6: T2/TD0	
	$0x1C + 0x40 \times n$		Element 7: TX_AP/TD1	
	$0x20 + 0x40 \times n$	TX FIFO Queue n (next/current TX Header Descriptor) (0<= n < N)	Element 0	TX Header Descriptor, see TX descriptor, TX Message and TX FIFO Queue chapters
	$0x24 + 0x40 \times n$		Element 1	
	$0x28 + 0x40 \times n$		Element 2: TS0	
	$0x2C + 0x40 \times n$		Element 3: TS1	
	$0x30 + 0x40 \times n$		Element 4: T0	
	$0x34 + 0x40 \times n$		Element 5: T1	
	$0x38 + 0x40 \times n$		Element 6: T2/TD0	
	$0x3C + 0x40 \times n$		Element 7: TX_AP/TD1	
	$0x0 + 0x40 \times N$	Temporary TX Trailing Descriptor	Element 0	TX Trailing Descriptor, see TX descriptor, TX Message and TX FIFO Queue chapters
	$0x4 + 0x40 \times N$		Element 1	
	$0x8 + 0x40 \times N$		Element 2: TS0	
	$0xC + 0x40 \times N$		Element 3: TS1	
	$0x10 + 0x40 \times N$		Element 4: T0	

(table continues...)

Table 1058 (continued) Memory organization of the TX descriptors

Memory base address	Offset	Name	Bit field	Description
	$0x14 + 0x40 \times N$		Element 5: T1	
	$0x18 + 0x40 \times N$		Element 6: T2/TD0	
	$0x1C + 0x40 \times N$		Element 7: TX_AP/TD1	
PQ_BASE_ADD[15:0]	$0x0 + 0x20 \times m$	TX Priority Queue slot n (0<= m < M)	Element 0	TX Header Descriptor, see TX descriptor, TX Message and TX Priority Queue chapters
	$0x4 + 0x20 \times m$		Element 1	
	$0x8 + 0x20 \times m$		Element 2: TS0	
	$0xC + 0x20 \times m$		Element 3: TS1	
	$0x10 + 0x20 \times m$		Element 4: T0	
	$0x14 + 0x20 \times m$		Element 5: T1	
	$0x18 + 0x20 \times m$		Element 6: T2/TD0	
	$0x1C + 0x20 \times m$		Element 7: TX_AP/TD1	

As the L_MEM can be shared across several MH, the SW is having some flexibility to allocate TX FIFO/Priority Queue descriptors anywhere and also according to the usage of the application. As an example, if only 4 TX FIFO Queues is required with a TX Priority Queue with 16 slots the expected memory size would be half compare to the maximum configuration possible. It is obvious that this kind of configuration would assume TX FIFO Queues are continuous, meaning 0, 1, 2 and 3 AND TX Priority Queue slots 0, 1, ... and 15.

One must be careful if more TX FIFO Queue and TX Priority Queue slots are required as more memory space would then need to be allocated. As a matter of fact, if the SW is able to enable any TX FIFO Queue and/or TX Priority Queue slots, the worst configuration would be a memory space configured with 8 TX FIFO Queues and 32 TX Priority Queues.

22.5.3.30.2 RX filter elements

The filter elements to be parsed are stored into the local memory on 32 bit word. The global setting of the RX Filter is defined by the Ni_RX_FILTER_CTRL register and will apply to all filter elements. Up to n filter elements can be defined (where n is from 0 to 255) with up to m reference/mask pair (where m is from 0 to 255). The number of elements is defined in the Ni_RX_FILTER_CTRL.NB_FE[7:0] bit-field register.

Table 1059

Memory Base Address	Offset ($0 \leq n \leq 254$) ($0 < n < 255$)	Name ($0 \leq n \leq 254$) ($0 < n < 255$)	Description
RX_FILTER_ADD[15:0]	$0x0$	FE0	Define the RX filter element 0
	$0x4 \times n$	FEn	Define the RX filter element $n - 1$

(table continues...)

Table 1059 (continued)

Memory Base Address	Offset ($0 \leq n \leq 254$) ($0 < n < 255$)	Name ($0 \leq n \leq 254$) ($0 < n < 255$)	Description
RX_FILTER_ADD[15:0] $+0x4 \times n$	$0x0$	REF0	RX Filter Reference value 0
	0x4 $0x4$	MSK0	RX Filter Reference mask 0
	$0x0 + 0x4 \times m$	REFm	RX Filter Reference value m
	$0x4 + 0x4 \times m$	MSKm	RX Filter Reference mask m

As the local memory can be shared across several Message Handler, the SW is having some flexibility to allocate RX filter elements and references anywhere and also according to the usage of the application. As a memory space of 64 Kbytes is addressable, the start address of those elements is defined in the Ni_RX_FILTER_MEM_ADD.BASE_ADDR[15:0] bit-field register.

22.5.3.31 Application information

General information related to performances, starting and stopping RX/TX FIFO Queues as well as TX Priority Queue.

22.5.3.31.1 Queue status flags

The TX FIFO Queue status is defined according to the bit status in the Ni_TX_FQ_STS0 register, as shown in the table below:

Table 1060 TX FIFO Queue status

Ni_TX_FQ_STS0.BUSY[n]	Ni_TX_FQ_STS0.STOP[n]	Status for TX FIFO QUEUE n (n = 0 to 31)
0	0	Inactive: The TX FIFO Queue can be programmed and started if enabled
0	1	na
1	0	Active and running: The TX FIFO Queue is enabled and has been started. TX messages are sent whenever possible to the PRT
1	1	Active and on hold: When considering no functional or safety errors, this status is reached when an invalid TX descriptor is fetched from S_MEM.

The TX Priority Queue slot n status is defined according to the bit status in the Ni_TX_PQ_STS0 register, as shown in the table below

Table 1061 TX Priority Queue status

Ni_TX_PQ_STS0.BUSY[n]	Status for TX PRIORITY QUEUE slot n (n=0 to 31)
0	Inactive: The TX Priority Queue can be programmed and started if enabled
1	Active and running: The TX Priority Queue is enabled and has been started. TX message in the slot n can be transmitted whenever possible

Compared to the TX FIFO Queues, there is no STOP bits. Any errors related to a TX Priority Queue slot execution sets the slot as inactive.

Table 1062 RX FIFO Queue status

Ni_RX_FQ_STS0.BUSY[n]	Ni_RX_FQ_STS0.STOP[n]	Status for RX FIFO QUEUE n (n = 0 to 31)
0	0	Inactive: The RX FIFO Queue can be programmed and started if enabled
0	1	na
1	0	Active and running: The RX FIFO Queue is enabled and has been started. RX messages can be received from the PRT
1	1	Active and on hold: When considering no functional or safety errors, this status is reached when an invalid RX descriptor is fetched from S_MEM.

22.5.3.31.2 Cluster

The same L_MEM can be shared accross several Message Handler but several points need to be highlighted. A tradeoff needs to be found to ensure every MH will get enough time to complete their RX filter process as well as their TX-Scan for a given L_MEM bandwidth.

- The worst scenario on RX path is defined when all MH in a cluster are receiving an RX message at the same time. Therefore, it is essential to ensure the available bandwidth on the L_MEM is able to support the RX filter process from all concurrent MH. Several measures can be taken to lower the bandwidth for a given value: limit the number of RX filter elements and the number of comparison (1 or 2) per filter element.
- The worst scenario on TX path is defined by all TX FIFO queues active for every MH as well as new TX messages being added to all TX Priority Queue slots. As one message is added or sent at a time for every MH, the impact of the TX-Scan may be limited but may play an important role by generating more arbitration occurences
- The read latency to access the L_MEM is a common factor for all MH and should be as low as possible. This access time is driven the overall performances when in cluster mode

22.5.3.31.3 Performances

Several processing time have a direct impact on the overall MH performances, see sections below.

Core clock frequency

The minimum MH core clock frequency is driven by several parameters:

- The maximum number of RX filter elements to support
- The Classical CAN, CAN FD and CAN XL bit rates (Arbitration and Data Phase)
- The L_MEM read latency
- The maximum number of TX FIFO Queues

To estimate the minimum core clock frequency to set, please refer to excel file [6]. One must keep in mind that the computated value is a minimum. Other clock frequency constraints may require an higher clock speed on the MH.

TX-Scan

The TX-Scan does compute the highest priority message over the TX FIFO queues and the TX Priority Queue slots. The processing time is mainly link to the number of TX FIFO Queues active at the same time as well as the number of TX Priority Queue slots being set active. The higher the number of slots and TX FIFO queues active, the higher the bandwitdh from the L_MEM. The sooner the result is known the better the expected transmission order is. For more detail on TX-Scan refer to the TX-Scan chapter.

RX Filter

As the RX filter elements are defined and read from the L_MEM, any RX message received will generate many accesses. The number of RX filter elements and the number of comparisons per element drive the bandwidth from the L_MEM and so the processing time. The higher the number of filter elements the higher it takes to define if an RX message is accepted or rejected. Despite some measures are in place to avoid to discard the current RX message, the SW would need to sort the non dispatched messages later on. For more detail on RX Filter refer to the RX Filter chapter.

The process of filtering is started as soon as the first RX message header data is received. When an RX filter element expects an RX data word that is not already stored, the process stops and waits for the RX data word. As the RX filter element are fetched linearly from the L_MEM, it is required to have them organized in a specific way to optimize the filtering time. This means, the RX filter elements, having to be compared to the first header data word, must be stored at the beginning, the ones using the second header data word at the last position. The Classical CAN with a low bit rate does provide more margin to complete the RX filtering in time. The critical path is defined when receiving CAN FD frame with no payload data.

As a general rule, it is recommended to define RX filter elements in this order:

1. CAN FD, assuming that only one comparison with the first message header word is required
2. CAN XL, assuming either one or two comparisons could be defined
3. Classical CAN , assuming that only one comparison with the first message header word is required

Such RX filter elements organization will optimize the overall processing time.

RX or TX Descriptors memory organization

RX/TX descriptors are fetched from the S_MEM. The DMA_CONTROLLER is reading and writing data to the S_MEM using burst length of various sizes. As soon as the address to read or write data is aligned on burst length of 8, all the following burst transfer are using maximum burst length of 8. If the address to fetch the RX/TX descriptor does cross a burst of 8 boundary, two read accesses are required.

TX path

As the TX header descriptor is store locally in the MH there is no constraint regarding the access time from the S_MEM. Nevertheless, several recommendation will help to increase access performances and to limit power consumption:

- Align TX FIFO Queue start address on maximum burst length (8 word of 32 bit)
- Align TX Priority Queue start address on maximum burst length (8 word of 32 bit)
- Define TX FIFO and or Priority Queues (link list of TX descriptors) in SRAM to leave more time for payload data fetching. This is best pratice to declare TX descriptor in SRAM whenever possible

RX path

RX descriptors fetches are driving the RX messages write to S_MEM. On top of it, if the RX filtering is taking too much time, an RX DMA FIFO overflow may occur. Several recommendation will help to increase access performances and to limit power consumption:

- Align RX FIFO Queue start address on maximum burst length (8 word of 32 bit)
- Define RX FIFO Queues (link list of RX descriptors) in SRAM to leave more time for RX filtering. This is best practice to declare RX descriptor in SRAM whenever possible

Data payload buffer memory organization

Any accesses done from or to the S_MEM by the DMA will be fully optimized as the address is aligned on the maximum burst length (8x32bit).

TX path

- Align data container start address on maximum burst length (8 word of 32 bit)
- Use data container size multiple of maximum burst length (8 word of 32 bit)

RX path

- Align data container start address on maximum burst length (8 word of 32 bit), whatever the mode (Normal or Continuous)
- Use data container size multiple of maximum burst length (8 word of 32 bit)

High system memory latency

If the latency time to get the first payload data burst is greater than the computed value, an underrun will occur when starting to transmit a TX message to the PRT. As many DMA requests may occur to the system bus at the same time, some critical scenarios could lead to delay the fetch of the first payload data, providing underrun. If one of the TX descriptor DMA requests is preempting the access to the first payload data for the current TX message, the delay would be larger than the one expected. As an example, starting all TX FIFO Queues and TX Priority Queue slots at the same time may increase the probability to have an underrun.

Very high system latency may lead to underrun due to the high constraints on burst accesses. The data underrun is a warning and won't affect the MH behavior and the order of the TX messages. No TX message with underrun is dropped and it will still be considered in the next TX-Scan run.

Nevertheless, here below are a list of recommendations to avoid and limit issues in a system with high latency:

TX path

Every TX descriptor and its payload data are fetched from the S_MEM. The payload data is only read from the S_MEM when the TX message (defined by its TX descriptor) started to be transmitted on the CAN bus (meaning the message has won the CAN bus arbitration). Thus, in case of very high latency system, a data underrun may occur on the PRT. As a matter of fact, the critical path is defined by the first bunch of payload data to be fetched. Several actions can be done to cope with high system latency:

- Align data container start address on maximum burst length (8 word of 32bit)

- Align TX FIFO Queue start address on maximum burst length (8 word of 32bit)
- Align TX Priority Queue start address on maximum burst length (8 word of 32bit)
- Use data container size multiple of maximum burst length (8 word of 32bit)
- The usage of write outstanding transaction may not provide a significant improvement as the write accesses are somehow sequential. Nevertheless, it is recommended to set it to the maximum value, see Ni_AXI_PARAMS.AW_MAX_PEND[1:0] bit register
- Make use of read outstanding transaction (this is mandatory to avoid cases where DMA channel are competing against each other to read the S_MEM). The maximum value is recommended, see Ni_AXI_PARAMS.AR_MAX_PEND[1:0] bit register
- Define TX FIFO/Priority Queues (linked list of TX descriptors) in SRAM to leave more time for payload data fetching. This is best practice to declare descriptor in SRAM whenever possible

RX path

Every RX descriptor is fetched from the S_MEM. As for the TX path, the critical path is defined by the first RX descriptor to be fetched, once the RX FIFO Queue number is defined by the RX filter. As soon as the RX FIFO Queue is known, there is still some time required to read the corresponding RX descriptor and to write the data payload to the S_MEM. To avoid any RX DMA FIFO overflow and to limit the constraints at system level, the faster the RX descriptor is read from the S_MEM the faster the payload data can be written to the S_MEM. Nevertheless, several actions can be done to cope with high system latency:

- Align data container start address on maximum burst length (8 word of 32bit)
- Align RX FIFO Queue start address on maximum burst length (8 word of 32bit)
- Use data container size multiple of maximum burst length (8 word of 32bit)
- The usage of write outstanding transaction may not provide a significant improvement as the write accesses are somehow sequential and the RX DMA FIFO sized to support high latency. Nevertheless, it is recommended to set it to the maximum value, see Ni_AXI_PARAMS.AW_MAX_PEND[1:0] bit register
- Make use of read outstanding transaction (this is mandatory to avoid cases where DMA channel are competing against each other to read the S_MEM). The maximum value is recommended, see Ni_AXI_PARAMS.AR_MAX_PEND[1:0] bit register
- Define RX FIFO Queues (linked list of RX descriptors) in SRAM to shorter the reaction time when receiving an RX message. This is best practice to declare descriptor in SRAM whenever possible

22.5.3.32 PRT and ENABLE signal

The PRT signalizes via ENABLE whether it is active and requires message handling or not. It means a message can be received or transmitted only if the ENABLE signal is set high by the PRT.

As soon as this ENABLE signal goes low, the MH must stop its activity and goes in idle state. The MH is stopped, therefore the active TX FIFO Queue n and RX FIFO Queue m are put on hold, it means: Ni_TX_FQ_STS0.BUSY[n] = 1, Ni_RX_FQ_STS0.BUSY[m] = 1, Ni_TX_FQ_STS0.STOP[n] = 1 and Ni_RX_FQ_STS0.STOP[m] = 1. Any active TX Priority Queue slot k is discarded, Ni_TX_PQ_STS0.BUSY[k] = 0. Any RX message received or TX message transmitted at that time is discarded. Therefore, it is up to the SW to decide what to do next:

- Keep going: As soon as the ENABLE signal is back to high, the SW would need to write the Ni_MH_CTRL.START bit to 1 to set the MH in active mode. Then, the already started TX FIFO Queues would need to be started again, using the Ni_TX_FQ_CTRL0.START[7:0] bit field register, to continue their execution from where they were. Once the Ni_RX_FQ_CTRL0.START[7:0] bit field register would be set to re-started the RX FIFO Queues, they would be back to where they were before being stopped. Within such option, all the registers and internal buffers will remain the same, only start bits are used to continue MH execution at the point of time it was stopped
- Reinitialize MH partly or completely: The SW can decide to cancel the execution of an RX/TX FIFO Queue or TX Priority Queue slot. To do so, use the Ni_TX_FQ_CTRL1.ABORT[7:0], Ni_RX_FQ_CTRL1.ABORT[7:0] or Ni_TX_PQ_CTRL1.ABORT[31:0] bit field register. Only active RX/TX FIFO Queues and TX Priority Queue slots

can be aborted (see Ni_TX_FQ_STS0.BUSY[7:0], Ni_RX_FQ_STS0.BUSY[7:0] and Ni_TX_PQ_STS0.BUSY[31:0] bit field registers). Once the selected RX/TX FIFO Queue and TX Priority Queue slot are aborted, the relevant ones can be started to keep going from where they were. Nothing is preventing the SW to abort all active RX/TX FIFO Queues and TX Priority Queue slots to reinitialize completely the MH.

22.6 Protocol controller (PRT)

This document describes the PRT, the CAN XL protocol controller, and its interfaces with the host controller, a message handler frontend, and the CAN transceiver.

22.6.1 Feature list

- Classical CAN, CAN FD and CAN XL as specified in ISO 11898-1:2024
- Classical CAN bit rate up to 1 Mbps
- Arbitration phase bit rate up to 1 Mbps for CAN FD and for CAN XL
- CAN FD data phase bit rate up to 8 Mbps at a clock speed of 80 MHz or 160 MHz
- CAN XL data phase bit rate up to 20 Mbps at a clock speed of 160 MHz
- Dedicated timebase interfaces

22.6.2 Functional overview

The PRT is a CAN XL Protocol Controller that can be integrated into different CAN modules. The PRT performs CAN communication as specified in ISO 11898-1:2024. The bit-rate can be configured to values up to 20 MBit/s at a clock speed of 160 MHz, depending on the used semiconductor technology. For the connection to the physical layer, additional transceiver hardware is required, which is connected through GPIO ports or may be integrated into the CAN module (see chapter “Transceiver interface”).

The PRT does not provide internal buffering of frames, so that data has to be transferred by IP internal Message Busses in 32-bit slices in real-time while (de)-serialization on the CAN Bus. Thus single data transfers at the internal Message Busses are closely time-synchronized to the schedule at the CAN bus.

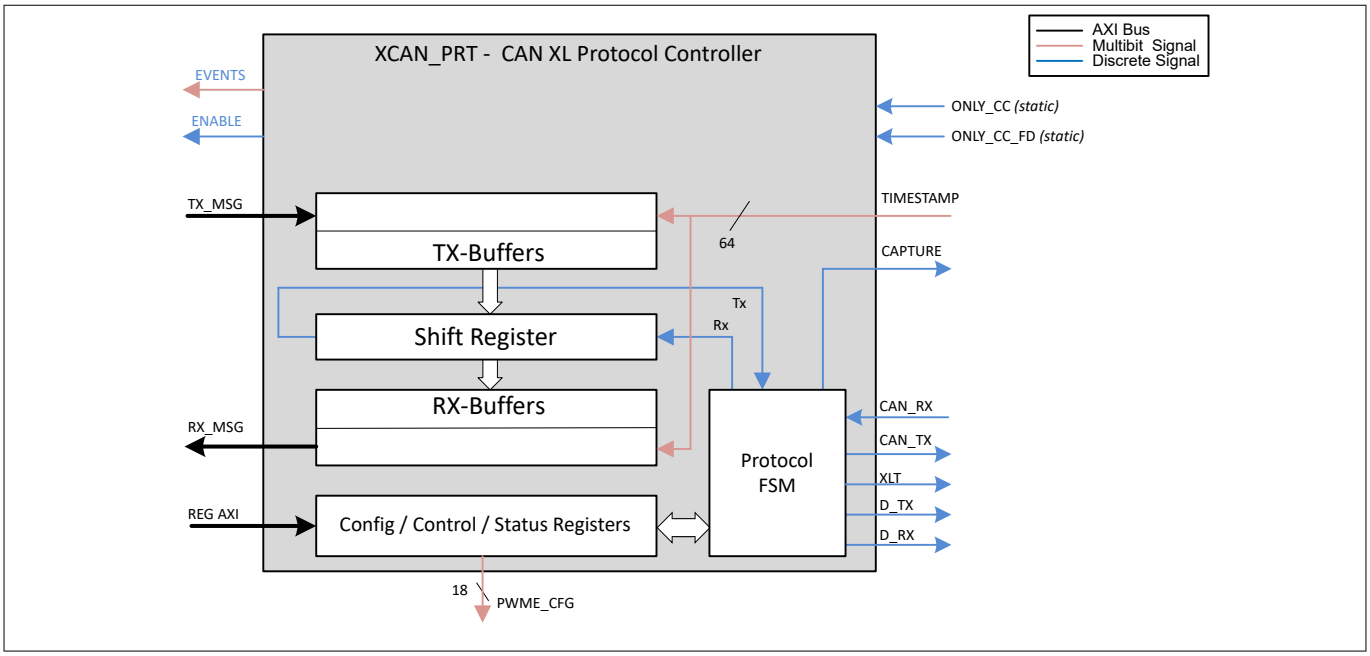


Figure 425 PRT block diagram

The figure above shows principle of the PRT’s functions. When the PRT and the message handler operate in different clock domains, they are connected through CDC modules. The time base is captured inside a CDC module, triggered by the PRT’s output signal CAPTURE.

The PRT consists of the CAN Protocol FSM, an Rx/Tx shift register, a set of interface registers for configuration, control, and status information as well as interfaces to the message handler for received messages and for messages to be transmitted. The PRT is not designed to store complete messages, there is only transient caching for two memory words for each direction during reception or transmission. A separate message handler is needed for the storage of whole messages as well as for functions like acceptance filtering, sorting of received messages into specific message buffers, and ordering the sequence of messages that are requested for transmission. Messages are streamed between message handler and PRT as sequences or 32-bit data words.

The host accesses the PRT's registers through REG_AXI, an AMBA AXI4-Lite interface, for configuration, control, and status information.

22.6.3 Functional description

The PRT does provide several interrupt outputs that signal, with a high-pulse of one CLK period length, the occurrence of specific internal events. For test purposes, the Ni_TEST register has a generate interrupt pulse function GIP so that in hardware test mode HWT an interrupt output pulse can also be triggered by writing a 0b1 to the corresponding Ni_TEST register bit.

Table 1063 Interrupts

Interrupt	TEST bit	Activated when
BUS_OFF	27	Entering Bus_Off state
BUS_ON	26	Starting CAN communication, after starting or end of Bus_Off
E_PASSIVE	25	Switching from error-active to error-passive
E_ACTIVE	24	Switching from error-passive to error-active
BUS_ERR	23	Error detected on CAN bus or protocol exception event detected
RX_EVT	22	Received a valid message
TX_EVT	21	Successfully transmitted a message
IFF_RQ	20	MH requests a message with invalid frame format in header
RX_DO	19	Data overflow condition in RX_MSG sequence detected
TX_DU	18	Data underrun condition in TX_MSG sequence detected
USOS	17	Unexpected start of sequence during TX_MSG sequence detected
ABORTED	16	TX_MSG sequence stopped by TX_MSG_WUSER code ABORT

The PRT outputs internal status information, optionally to be connected to a hardware debug port

- SAMPLE_POINT: This is the CAN sample point
- STAT_ACT: This is the actual 2-bit-value of register Ni_STAT.ACT (see register description)

22.6.3.1 Software reset

The software reset is triggered by writing 0b1 to Ni_CTRL.SRES when the CAN protocol operation is stopped. This does not require an unlocking sequence. The software reset must not be executed when Ni_CTRL.SRES is written while the CAN protocol operation is started. The software reset resets all state machines of the PRT (excluding the error-counters and the error-states) and clears the following readable registers: Ni_STAT.TDCV, Ni_STAT.FIMA, Ni_FIMC.FIP, Ni_TEST.HWT, Ni_TEST.TXC, Ni_TEST.LBCK and all flags of Ni_EVNT. The configuration registers are not changed by a software reset.

22.6.3.2 Operating mode

The operating mode is defined using the Ni_MODE register and only when the CAN communication is stopped, otherwise the register is read only. The register Ni_MODE defines separate operating mode options.

The four configuration bits FDOE, XLOE, EFDI, and XLTR, and are interrelated according the table below.

Table 1064 Frame formats

FDOE	XLOE	XLTR	EFDI	Description
0	0	X	0	When FDOE is not set, the PRT shall be to the Classical CAN frame format. In this case, when PXHD is set, the PRT shall accept both recessive and dominant bits as received reserved bits. When PXHD is not set, the PRT shall treat a recessive received first reserved bit as a Protocol Exception condition and shall enter the Protocol Exception State as defined in [1] and it shall set the flag Ni_EVNT.PXE. FDOE shall be static at 0b0 while the input signal ONLY_CC is 0b1
0	1	X	X	Invalid configuration
0	X	X	1	Invalid configuration
X	0	X	1	Invalid configuration
1	0	X	0	Operating in Classical CAN and CAN FD frame format. When FDOE is set, the PRT shall be able to transmit and to receive Classical CAN frames and CAN FD frames as defined in [1]. When XLOE is not set, the PRT shall not be able to transmit or to receive CAN XL frames as defined in [1]. When FDOE is set but not XLOE, PXHD defines the PRT's reaction on a recessive reserved bit following the recessive FDF bit in a CAN FD frame. In this case, when PXHD is set, the PRT shall treat this condition as a Form Error. When PXHD is not set, the PRT shall treat this condition as a Protocol Exception condition and shall enter the Protocol Exception State as defined in [1] and it shall set the flag Ni_EVNT.PXE. XLOE shall be static at 0b0 while the input signals ONLY_CC_FD or ONLY_CC are at 0b1
1	1	0	0	Operating in all frame formats, without XL transceiver. When FDOE and XLOE are both set and >EFDI is not set, the PRT shall be able to transmit and to receive Classical CAN frames, CAN FD frames and CAN XL frames as defined in [1]. When both FDOE and XLOE are set, PXHD defines the PRT's reaction on a recessive reserved bit following the recessive XLF bit in a CAN XL frame. In this case, when PXHD is set, the PRT shall treat this condition as a Form Error. When PXHD is not set, the PRT shall treat this condition as a Protocol Exception condition and shall enter the Protocol Exception State as defined in [1] and it shall set the flag Ni_EVNT.PXE. When EFDI is not set, the PRT shall send error flags as defined in [1]
1	1	0	1	Operating in XL frame format only, without XL transceiver, error frames are disabled for all communication. It shall be an invalid configuration to set EFDI without setting both FDOE and XLOE. When XLTR is not set, the PRT shall not control the operating mode of the transceiver
1	1	1	0	Invalid configuration

(table continues...)

Table 1064 (continued) Frame formats

FDOE	XLOE	XLTR	EFDI	Description
1	1	1	1	Operating in XL frame format only, enabling XL transceiver, error frames are disabled for all communication. When XLTR is set together with FDOE and XLOE, the PRT shall control the PWME to the transceiver in order to switch the operating mode of the transceiver at the beginning and at the end of the CAN XL data phase, as defined in [1]. When XLTR is set together with FDOE and XLOE, the PRT shall control the PWME to the transceiver in order to switch the operating mode of the transceiver at the beginning and at the end of the CAN XL data phase, as defined in [1]. When EFDI is set together with FDOE and XLOE, the PRT shall not send error flags and it shall not change its transmit error counter or its receive error counter. When an error condition occurs, the PRT shall, instead of sending an error flag, enter Protocol Exception State, like in Restricted Mode, as defined in [1]

22.6.3.3 Starting and stopping the module

The PRT is started by writing 0b1 to Ni_CTRL.STRT. This does not require an unlocking sequence. After the start command, the PRT waits for the occurrence of a sequence of 11 consecutive recessive bits (the idle condition of ISO 11898-1:1995) to finish its integration into the CAN communication on the CAN bus line. When the PRT has detected the idle condition, it switches into idle state. When it sees a recessive bit on entering idle state, the PRT is able to start a pending transmission in the following bit. When the PRT sees a dominant bit on entering idle state, it immediately becomes receiver of that frame.

There are two options to stop the CAN protocol operation under software control, one that waits for the completion of an ongoing message transfer and one that stops the operation immediately.

The two options use different variants of the Ni_CTRL.STOP command. In the first variant, only the STOP bit is written to 0b1. In the second variant, both the Ni_CTRL.STOP bit and the Ni_CTRL.IMMD bit are written to 0b1 at the same time. Both variants require the application of the unlock key sequence, followed by a write access to the Ni_CTRL register. The three consecutive write accesses may not be interrupted by other accesses through *REG_AXI*.

The first variant of the STOP command is asserted this way:

1. Write 0x1234 to Ni_LOCK.ULK 0x40
2. Write 0x4321 to Ni_LOCK.ULK 0x40
3. Write 0b1 to Ni_CTRL.STOP 0x44

The PRT's reaction to the first variant of the STOP command depends on its current activity (Ni_STAT.ACT). When the current activity of this node is Idle, it switches its activity to its inactive state immediately and stop all CAN operation. When the current activity is either Receiver or Transmitter, it continues that activity and it sets the status flag Ni_STAT.STP to show that it is waiting for end of the actual message after a Ni_CTRL.STOP command. If no TX_MSG sequence is already started, the PRT clears *TX_MSG_WREADY* and keep it cleared until all CAN operation is stopped. As soon as the current reception or transmission is finished (either successfully or in failure) the PRT reports the result of that transfer to the MH, clears the status flag Ni_STAT.STP, clears *ENABLE*, switches its activity to its inactive state, and stops all CAN operation. The PRT does not start another reception or transmission until it is started again.

The second variant of the STOP command is asserted this way:

1. Write 0x1234 to Ni_LOCK.ULK 0x40
2. Write 0x4321 to Ni_LOCK.ULK 0x44
3. Write 0b1 to Ni_CTRL.IMMD and 0b1 to Ni_CTRL.STOP 0x44
4. Write 0b1 to Ni_CTRL.SRES 0x44

The PRT's reaction to the second variant of the STOP command is to switch its activity to its inactive state immediately, to stop all CAN operation, and to set its *CAN_TX* output to 0b1 and to clear *ENABLE*. When the current activity was Transmitter, the PRT aborts that transmission. When the current activity was Receiver, it abort that reception. Both interfaces with the MH are reset, outstanding transactions are discontinued. If this happens while an RX_MSG sequence was ongoing, this sequence is discontinued with the ABORT code. The PRT does not start another reception or transmission until it is started again.

The CAN protocol operation stops automatically on the following conditions:

- When the node enters the CAN protocol's Bus-Off state
- Unexpected start of sequence during TX_MSG sequence detected

***Note**: Detecting an Unexpected Start of Sequence (USOS) indicates a mis-synchronization between PRT and MH that requires a restart.*

When the CAN protocol operation is stopped, it is started by writing 0b1 to Ni_CTRL.STRT. This does not require an unlocking sequence.

When the CAN protocol operation was stopped because the PRT entered the CAN Bus_Off state, the start command (writing 0b1 to Ni_CTRL.STRT) causes the PRT to perform the CAN Bus_Off Recovery Sequence before it is again able to participate in CAN communication.

The CAN Bus_Off Recovery Sequence (see ISO 11898-1:2024) cannot be shortened by starting or stopping the PRT. If the PRT goes Bus_Off, it will set Ni_STAT.BO bit and it will, of its own accord, stop all bus activities. Once the PRT has been started again, the PRT clears Ni_STAT.TEC, Ni_STAT.RP, Ni_STAT.REC, and Ni_STAT.EP but it will keep Ni_STAT.BO.

The PRT will then wait for 129 occurrences of Bus Idle (129 * 11 consecutive recessive bits) before resuming normal operation. The PRT uses the Receive Error Counter (Ni_STAT.REC) to count the occurrences of Bus Idle. Additionally, each time a sequence of 11 recessive bits has been monitored, a Bit0 Error (Ni_EVNT.B0E) is reported, enabling the host to readily check up whether the CAN bus is stuck at dominant or continuously disturbed and to monitor the progress of the Bus_Off recovery sequence. When the last-but-one sequence of 11 recessive bits has been monitored, Ni_STAT.RP, and Ni_STAT.EP are set and Ni_STAT.REC is at 0x7F. When the last sequence of 11 recessive bits has been monitored, the end of the Bus_Off recovery sequence is reached and Ni_STAT.TEC, Ni_STAT.RP, Ni_STAT.REC, Ni_STAT.EP and Ni_STAT.BO will all be reset. The PRT switches into idle state.

22.6.3.4 Reaction on exceptions at the TX_MSG and RX_MSG interfaces

22.6.3.4.1 MH requests a message with invalid frame format in header

This is detected when the requested transmit frame format is disabled in the configuration register (see Ni_MODE.FDOE, Ni_MODE.EFDI, or Ni_MODE.XLOE) or there is an internal contradiction in the header content of that frame. On the detection of this condition, the PRT ends the TX_MSG sequence with the response code HFI, generate a pulse on the IFF_RQ interrupt output and it does not transmit that message.

22.6.3.4.2 MH intentionally aborts TX_MSG sequence

If the ABORT command is given with the second transaction of the TX_MSG sequence, the PRT does not start the transmission. If the ABORT command is given after the second transaction of the TX_MSG sequence, the PRT sets an internal flag that causes the FCRC bits of that transmission to be transmitted inverted. This internal flag is cleared at the end of the transmission. In both cases, the PRT generates a pulse on the ABORTED interrupt output and the ongoing TX_MSG sequence is finished.

22.6.3.4.3 Data underrun condition in TX_MSG sequence detected

This is detected when the ongoing transmission on the CAN bus needed another TX_MSG data word but that word was not provided in time. On the detection of this condition, the PRT continues the transmission (to avoid disturbing the message schedule on the CAN bus), but the PRT generates a pulse on the TX_DU interrupt output and the PRT sets an internal flag that causes the FCRC bits of that transmission to be transmitted inverted (to avoid the acceptance of a message containing invalid data). This internal flag is cleared at the end of the transmission. In case of a Data Underrun, the ongoing TX_MSG sequence finishes with the code DU when the MH transfers the missing data word. After the end of the transmission, TX_MSG_WREADY is asserted again to accept the following TX_MSG Sequence (starting again with W0).

22.6.3.4.4 Unexpected start of sequence detected

This is detected when the PRT receives a TX_MSG transaction marked as start of sequence before a previously started TX_MSG sequence has ended. This indicates that MH and PRT operate out-of-phase. On the detection of this condition, the PRT generates a pulse on the USOS interrupt output and the PRT stops CAN protocol operation. Afterward, the PRT needs to be restarted under software control by writing 0b1 to Ni_CTRL.STRT.

22.6.3.4.5 Data overflow condition in RX_MSG sequence detected

This is detected when, during an ongoing reception, the MH has not acknowledged an RX_MSG data word in time. On the detection of this condition, the PRT generates a pulse on the RX_DO interrupt output and the PRT ends such an RX_MSG sequence through a subsequent transfer with code DO. This transfer with code DO must be acknowledged by the MH before the PRT can start a new RX_MSG sequence.

22.6.3.5 Controlling the module's clock input

The PRT has two clock inputs, CLK and CLK_AXI. CLK is the clock input of the PRT excluding its REG_AXI interface, while CLK_AXI is the clock input of the PRT's REG_AXI module. Both clocks are synchronous to each other, driven from the same source. The difference between the two clocks is that CLK_AXI must be always active (to keep the REG_AXI interface operational), but CLK may be switched off (gated) while the PRT is stopped, for example when no CAN communication is needed.

- The recommended clock frequency for CAN XL operation is 160 MHz
- The recommended clock frequency for CAN FD operation only is 80 MHz

The function of the PRT does not depend on a particular duty-cycle of the clock, that means it reacts only on rising clock edges. The duration of the clock high pulse may vary between 10% and 90% of the clock period during operation.

The PRT's input signal CLOCK_ACTIVE shows whether the clock input CLK of the PRT is active. The actual value of the CLOCK_ACTIVE input signal is always readable from the status bit Ni_STAT.CLKA, even when CLK is not active. The signal CLOCK_ACTIVE does not control the PRT's function, it provides only status information, coming from a clock multiplexer outside of the PRT.

While CLK is not active, the PRT has no function and cannot be started. CLK must be reactivated before the PRT needs to be started again.

22.6.3.6 Transceiver interface

The CAN bus is usually implemented as a twisted-pair bus line, its bus wires called CAN_H and CAN_L. An analog CAN transceiver device is connected to the CAN bus, interfacing between the bidirectional CAN bus wires and the CAN protocol controller's unidirectional, digital serial input and output signals. The future CAN XL transceivers will need to switch between two operating modes during the transmission of a CAN XL message. The switching control is coded into signals between protocol controller and transceiver. The coding is

implemented inside a separate, dedicated PWME module that is placed between protocol controller and transceiver, see block diagram.

The transceiver's RxD output is connected to the PRT's CAN_RX input. This asynchronous input signal is synchronized to the CAN clock by routing it through two synchronizer-FFs. This delay of two clock cycles is part of the input delay for the calculation of the propagation segment length of the CAN bit time. The CAN bit time configuration is only functional if the following conditions are fulfilled:

- $((NTSEG1 + 1) \times (BRP \times CLK_period))$ is larger than the transmitter loop delay
- $((DTSEG1 + 1) \times (BRP \times CLK_period))$ and $((XTSEG1 + 1) \times (BRP \times CLK_period))$ are larger than the transmitter loop delay or transmitter delay compensation is enabled

The connection from the PRT to the transceiver is routed through the PWME module (Pulse Width Modulation Encoder, specified in [1]). In CAN XL communication using a transceiver with switchable operating modes, the PWME controls the operating mode of the transceiver, to switch it into the CAN XL data phase modes for transmissions as well as receptions and back. The PWME_CFG configuration data consists of Ni_PCFG.PWMO, Ni_PCFG.PWML, and Ni_PCFG.PWMS, it is an 18-bit vector concatenating the three 6-bit vectors from Ni_PCFG.PWMO[5] downto Ni_PCFG.PWMS[0]. The appropriate switching times are signaled by the PRT via its outputs XLT, D_TX and D_RX (see [1]).

The PWME function is controlled by the following outputs of the PRT: PWME_CFG, XLT, D_TX, D_RX, and CAN_TX.

22.6.3.7 Hardware timestamping

22.6.3.7.1 Timestamping function

Timestamps are captured for each transmitted or received message, captured at either the sample point of the start of frame bit of the message or at the sample point of the bit when the message becomes valid at the end of the frame. The capture position is defined by the configuration bit Ni_MODE.SFS.

22.6.3.7.2 Time stamping offset

Due to internal propagation delays for the timestamping in the Protocol Controller (1 CAN clock cycle) and the clock domain crossing for the time capturing (n clock cycles, depending on the clock frequency ratio between the CAN and the TIMBASE domain) a certain offset has to be considered to derive the correct time.

These are the formulas for the offset:

TS_offset_max = 2 CAN_CLK period + 3 TIMEBASE_CLK period

TS_offset_min = 2 CAN_CLK period + 2 TIMEBASE_CLK period

This is the formula for the corrected timestamp:

Corrected Timestamp = Timestamp in TX/RX descriptor - TS_offset

22.6.3.8 Trace and debug

The hardware test mode functions are disabled by the software reset of the PRT. The status flag Ni_TEST.HWT shows whether the hardware test mode functions are enabled.

When the hardware test mode functions are disabled, all bits of Ni_TEST are cleared with the exception of RXD.

Enabling the hardware test mode functions (see chapter "Hardware Test Functions") requires the application of the test mode key sequence. The test mode key sequence consists of three consecutive write accesses, not interrupted by other accesses through REG_AXI:

1. Write 0x6789 to Ni_LOCK.TMK 0x40

2. Write 0x9876 to Ni_LOCK.TMK 0x40
3. Write 0b1to Ni_CTRL.TEST 0x44

The hardware test mode functions enable the host to directly control the values driven at the transceiver interface pins, to read the actual transceiver RxD output, and to transmit messages in a loop-back mode where all transmitted messages are also reported through the RX_MSG interface as received messages.

The CAN_RX input (output signal of the transceiver) is always readable at RXD.

The CAN_TX output control TXC offers four options:

- 0b00: Normal function of CAN_TX
- 0b01: Normal function of CAN_TX, CAN_RX is ignored (for message loop-back mode)
- 0b10: CAN_TX output set to 0b0 and XLT output set to 0
- 0b11: CAN_TX output set to 0b1 and XLT output set to 0

When LBCK is set, the PRT operates in the message loop-back mode. In message loop-back mode, the PRT reports transmit messages (requested through the TX_MSG interface) through RX_MSG as received messages. The transmit messages are encoded and decoded bitwise inside the PRT, but a transmitted message is treated as successfully transmitted even if it does not get ACK. When the host sets LBCK to 0b1, it also sets TXC to either 0b01 or to 0b11 to control whether messages transmitted in the message loop-back mode are visible at the transceiver pins. In the message loop-back mode with TXC set to a value > 0b00, the actual CAN_RX input (output signal of the transceiver) is ignored by the PRT.

When the host sets TXC = 0b11 in the message loop-back mode, the PRT keeps the CAN_TX output at 0b1 and loops back its internal serial output signal to its internal serial input signal.

When the host sets TXC = 0b01 in the message loop-back mode, the PRT drives the frame bits at its CAN_TX output. In this case, the PRT loops back its internal serial output signal to its internal serial input signal, so it is not able to perform an arbitration or to react on bit errors on the CAN bus.

If TXC is set as zero b00, the loop-back test may be disturbed by errors on the CAN_RX input.

If TXC is set as 0b01, the loop-back test operates independently from the CAN_RX input, the loop-back transmission can be monitored at the CAN_TX output.

If TXC is set as 0b11, the loop-back test operates independently from the CAN_RX input, the LoopBack transmision cannot be monitored at the Tx-pin. This is intended for a self-test in the field, not disturbing the CAN bus.

22.7 Pulse width modulation encoder (PWME)

This document describes the PWME, the Pulse Width Modulation Encoder, and its interfaces with the CAN XL protocol controller and the CAN transceiver.

22.7.1 Feature list

- PWM encoding as specified in [1]

22.7.2 Functional overview

The following figure shows the PWME and its interfaces with the CAN XL protocol controller and the CAN transceiver.

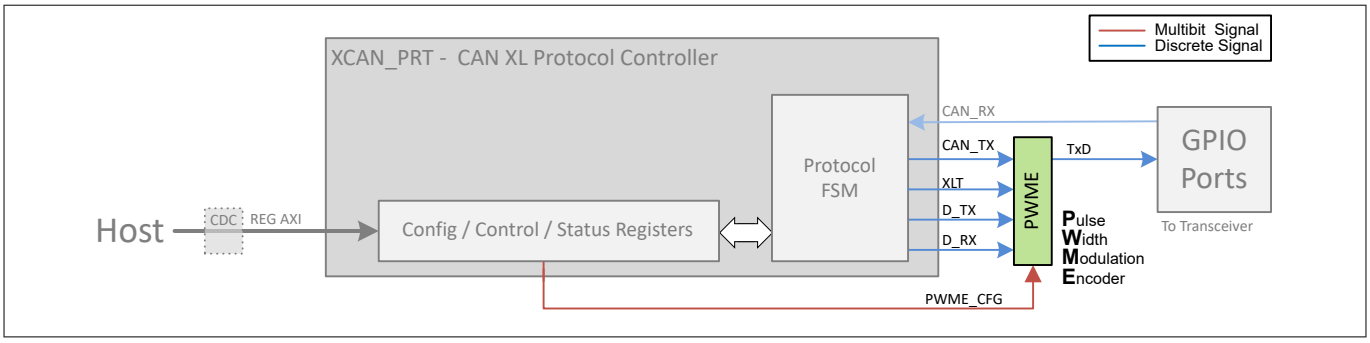


Figure 426 PWME overview

22.7.3 Functional description

PWME implements the PWM encoding as specified in [1]. When transceiver mode switching is enabled, the PWME encodes the CAN_TX input signal during a CAN XL frame's data phase and during ADH bit, to generate the PWM encoded output signal TXD.

The output of the PWM logic is registered by the Flip Flop TXD_PWME. An active reset sets this Flip Flop TXD_PWME to one. All Flip Flops in the PWME change their value with the rising edge of CLK.

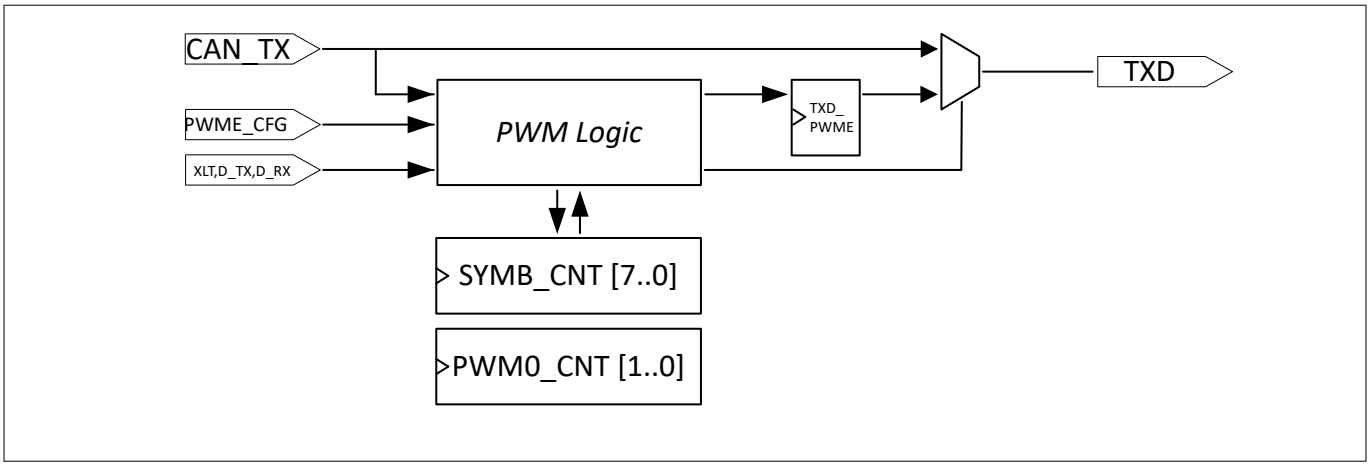


Figure 427 PWME block diagram

22.7.3.1 Transparent mode

While XLT is passive or both D_RX and D_TX are passive, the PWME interface behaves transparent between CAN_TX input and TXD output.

This mode is independent of reset.

22.7.3.2 PWM encoded mode

While XLT is active, the TXD output is PWM encoded for a transmitting node while D_TX is active and for the receiving node while D_RX active. The PWM encoded TXD output has one CLK cycle delay (internal processing delay) relative to the bit boundaries on CAN_TX input.

22.7.3.2.1 Transmitting node

When the PWME detects an edge from passive to active on D_TX and if XLT is active, then the PWME drives a LOW level on TXD for one PWM offset time.

With expiration of the PWM offset time the TXD output drives 2 consecutive PWM_0 symbols. From there onwards all following PWM symbols follow the CAN_TX input.

When D_TX is passive or XLT is passive the PWME switches back to transparent behavior regardless of the actual PWM phase.

22.7.3.2.2 Receiving node

When the PWME detects an edge from passive to active on D_RX and if XLT is active, then the PWME drives consecutive PWM_1 symbols without a leading offset time.

When D_RX is passive or XLT is passive the PWME switches back to transparent behavior regardless of the actual PWM phase.

22.7.3.3 Application information

Not applicable.

22.7.3.3.1 PWME configuration

The PWME_CFG contains the parameters needed for the PWM encoding (as defined in [1]) in the PWME module. The PWME_CFG signal may not change its value while the PRT is started, i.e. while CAN frames can be received and transmitted.

Valid values for the PWM phase Short PWMS are 00_H to $3F_H$. The actual interpretation of this value is that the PWM short phase length is (PWMS + 1) clock cycles long.

Valid values for the PWM phase Long PWML are 00_H to $3F_H$. The actual interpretation of this value is that the PWM long phase length is (PWML + 1) clock cycles long.

The PWM symbol length is the sum of PWM short phase length and PWM long phase length (PWMS + PWML + 2) clock cycles.

Valid values for the PWM Offset PWMO are 00_H to $3F_H$. PWMO shall always be smaller than the PWM symbol length (PWMO < PWMS + PWML + 2).

22.8 Interrupt Controller (IRC)

22.8.1 Feature list

- Interrupt generation for Functional events
- Interrupt generation for Error event
- Interrupt generation for Safety critical events
- Masking of individual events

22.8.2 Functional overview

The X_CAN IP is equipped with a central interrupt controller (IRC). It captures all events of the MH and PRT and can be configured for each event individually to interrupt the HOST CPU.

The events are organized in two categories, i.e. Functional Events and Error Events. Functional Events can trigger the IRC output FUNC_INT. Error Events can trigger the IRC outputs ERR_INT and SAFETY_INT.

22.8.3 Functional description

The following table lists the categories and shows the related IRC registers together with the dedicated IRC outputs:

Table 1065 Interrupt categories

Category	IRC Registers	IRC output
Functional Event	Ni_FUNC_RAW, Ni_FUNC_CLR, Ni_FUNC_ENA	FUNC_INT
Error Event	Ni_ERR_RAW, Ni_ERR_CLR, Ni_ERR_ENA	ERR_INT
	Ni_SAFETY_RAW, Ni_SAFETY_CLR, Ni_SAFETY_ENA	SAFETY_INT

For each category, three registers are implemented: xxx_RAW, xxx_CLR, and xxx_ENA. To capture the events, the xxx_RAW registers are used. These registers provide information about the occurrence of events inside the MH and the PRT. A flag is set when the related event occurred, independent of xxx_ENA. The flags remain set until the HOST CPU clears them by writing a 1 to the corresponding bit position at register xxx_CLR.

The xxx_ENA registers control on bit level, whether a certain bit in the xxx_RAW register can activate the interrupt line xxx_INT. The interrupt line xxx_INT gets active high, when at least one RAW/ENA pair is 1, e.g. ERR_INT gets active high, when Ni_ERR_RAW.MH_RX_FILTER_ERR = Ni_ERR_ENA.MH_RX_FILTER_ERR = 1.

22.9 Registers

22.9.1 Register overview - access mode glossary

Table 1066 Register overview - access mode glossary

Keyword	Description
E	Access protection using PROT register PROTE.
SE	Access protection using PROT register PROTSE.
APU-PM	Protection group consisting of registers MODULE_ACCEN_WRA, MODULE_ACCEN_WRB, MODULE_ACCEN_RDA, MODULE_ACCEN_RDB, MODULE_ACCEN_VM, MODULE_ACCEN_PRS.
PM	Access protection using *APU-PM* registers.
APU-PNi (i=0-3)	Protection group consisting of registers NODEi_ACCENNODE_WRA, NODEi_ACCENNODE_WRB, NODEi_ACCENNODE_RDA, NODEi_ACCENNODE_RDB, NODEi_ACCENNODE_VM, NODEi_ACCENNODE_PRS.
PNi	Access protection using *APU-PNi* registers.
U	No access restrictions.
BE	Always returns a Bus Error.
PROT	Access restrictions as defined in the PROT register access rules.
32	Access only when using 32-bit width.

22.9.2 Registers overview - CANXL (ascending offset address)

Table 1067 Registers overview - CANXL (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ID	Module Identification Register	00000_H	U	BE	PowerOn Reset	4310
OCS	OCDS Control and Status Register	00004_H	PM	SV, PM	Debug Reset	4310
MODULE_CLC	Module Clock Control Register	00008_H	PM	PM, SV, E	Application Reset	4312
MODULE_RST_CTRLA	Module reset control register A	$0000C_H$	PM	PM, SV, E	Application Reset	4312
MODULE_RST_CTRLB	Module reset control register B	00010_H	PM	PM, SV, E	Application Reset	4313
MODULE_RST_STAT	Module reset status register	00014_H	PM	BE	Application Reset	4314
MODULE_ACCEN_WRA	Module Write access enable register A	00018_H	U	SE, SV	Application Reset	4314
MODULE_ACCEN_WRB	Module Write access enable register B	$0001C_H$	U	SE, SV	Application Reset	4315
MODULE_ACCEN_RDA	Module Read access enable register A	00020_H	U	SE, SV	Application Reset	4315
MODULE_ACCEN_RDB	Module Read access enable register B	00024_H	U	SE, SV	Application Reset	4316
MODULE_ACCEN_VM	Module VM access enable register	00028_H	U	SE, SV	Application Reset	4316
MODULE_ACCEN_PRS	Module PRS access enable register	$0002C_H$	U	SE, SV	Application Reset	4317
PROTE	PROT Register Endinit	00038_H	U	SV, PROT	Application Reset	4318
PROTSE	PROT Register Safe Endinit	$0003C_H$	U	SV, PROT	Application Reset	4319
CLKEN	Clock enable	00040_H	clk, PM	clk, PM, SV, E	Application Reset	4321
NODEi_RST_CTRLA	Node i reset control register A	00100_H+i*100_H	PNi	PNi, SV, E	Application Reset	4322
NODEi_RST_CTRLB	Node i reset control register B	00104_H+i*100_H	PNi	PNi, SV, E	Application Reset	4322
NODEi_RST_STAT	Node i reset status register	00108_H+i*100_H	PNi	BE	Application Reset	4323
NODEi_ACCENNODE_WRA	Node i write access enable register A	$0010C_H$+i*100_H	U	SE, SV	Application Reset	4324
NODEi_ACCENNODE_WRB	Node i write access enable register B	00110_H+i*100_H	U	SE, SV	Application Reset	4324

(table continues...)

Table 1067 (continued) Registers overview - CANXL (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
NODEi_ACCENNODE_RDA	Node i read access enable register A	00114_H+i*100_H	U	SE, SV	Application Reset	4325
NODEi_ACCENNODE_RDB	Node i read access enable register B	00118_H+i*100_H	U	SE, SV	Application Reset	4326
NODEi_ACCENNODE_VM	Node i VM access enable register	$0011C_H$+i*100_H	U	SE, SV	Application Reset	4326
NODEi_ACCENNODE_PRS	Node i PRS access enable register	00120_H+i*100_H	U	SE, SV	Application Reset	4327
NODEi_ACCENNODE_RGNLA	Node i region lower address register	00124_H+i*100_H	U	SE, SV	Application Reset	4328
NODEi_ACCENNODE_RGNUA	Node i region upper address register	00128_H+i*100_H	U	SE, SV	Application Reset	4328
NODEi_VMPRSCONFIG	Node i VM and PRS configuration Register	$0012C_H$+i*100_H	U	SE, SV	Application Reset	4328
NODEi_PORTCTRL	Node i Port Control Register	00130_H+i*100_H	PNi	PNi	Kernel Reset	4329
NODEi_MTI_RAW	Node i message transfer interrupt event register	00134_H+i*100_H	PNi	BE	Kernel Reset	4330
NODEi_MTI_CLR	Node i message transfer interrupt clear register	00138_H+i*100_H	PNi	PNi	Kernel Reset	4330
NODEi_MTI_ENA	Node i message transfer interrupt enable register	$0013C_H$+i*100_H	PNi	PNi	Kernel Reset	4331
DEBUG_CTL	Debug control register	10004_H	PM	PM	Kernel Reset	4332
Ni_TS_CTL	Node i timestamp control	20010_H+i*2000_H	PNi	PNi	Kernel Reset	4332
Ni_TS_CLOCK_CTL	Node i timestamp clock control	20014_H+i*2000_H	PNi	PNi	Kernel Reset	4333
Ni_TS_CMD	Node i timestamp command	20020_H+i*2000_H	PNi	PNi	Kernel Reset	4334
Ni_TS_CNT_LO	Node i timestamp counter LSBs	20030_H+i*2000_H	PNi	BE	Kernel Reset	4334
Ni_TS_CNT_HI	Node i timestamp counter MSBs	20034_H+i*2000_H	PNi	BE	Kernel Reset	4335
Ni_VERSION	Node i release identification register	21000_H+i*2000_H	PNi, 32	BE	Kernel Reset	4335
Ni_MH_CTRL	Node i message handler control register	21004_H+i*2000_H	PNi, 32	PNi, 32	Kernel Reset	4336
Ni_MH_CFG	Node i message handler configuration register	21008_H+i*2000_H	PNi, 32	PNi, 32	Kernel Reset	4337
Ni_MH_STS	Node i message handler status register	$2100C_H$+i*2000_H	PNi, 32	BE	Kernel Reset	4338

(table continues...)

Table 1067 (continued) Registers overview - CANXL (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
Ni_MH_SFTY_CFG	Node i message handler safety configuration register	21010_H+i *2000_H	PNi, 32	PNi, 32	Kernel Reset	4339
Ni_MH_SFTY_CTRL	Node i message handler safety control register	21014_H+i *2000_H	PNi, 32	PNi, 32	Kernel Reset	4340
Ni_RX_FILTER_MEM_ADD	Node i RX filter base address register	21018_H+i *2000_H	PNi, 32	PNi, 32	Kernel Reset	4341
Ni_TX_DESC_MEM_ADD	Node i TX descriptor base address register	$201C_H$+ i*2000_H	PNi, 32	PNi, 32	Kernel Reset	4342
Ni_AXI_ADD_EXT	Node i AXI address extension register	21020_H+i *2000_H	PNi, 32	PNi, 32	Kernel Reset	4342
Ni_AXI_PARAMS	Node i AXI parameter register	21024_H+i *2000_H	PNi, 32	PNi, 32	Kernel Reset	4343
Ni_MH_LOCK	Node i message handler lock register	21028_H+i *2000_H	PNi, 32	PNi, 32	Kernel Reset	4343
Ni_TX_DESC_ADD_PT	Node i TX descriptor current address pointer register	21100_H+i *2000_H	PNi, 32	BE	Kernel Reset	4344
Ni_TX_STATISTICS	Node i TX message counter register	21104_H+i *2000_H	PNi, 32	PNi, 32	Kernel Reset	4344
Ni_TX_FQ_STS0	Node i TX FIFO queue status register	21108_H+i *2000_H	PNi, 32	BE	Kernel Reset	4345
Ni_TX_FQ_STS1	Node i TX FIFO queue status register	$210C_H$+ i*2000_H	PNi, 32	BE	Kernel Reset	4346
Ni_TX_FQ_CTRL0	Node i TX FIFO queue control register 0	21110_H+i *2000_H	PNi, 32	PNi, 32	Kernel Reset	4346
Ni_TX_FQ_CTRL1	Node i TX FIFO queue control register 1	21114_H+i *2000_H	PNi, 32	PNi, 32	Kernel Reset	4347
Ni_TX_FQ_CTRL2	Node i TX FIFO queue control register 2	21118_H+i *2000_H	PNi, 32	PNi, 32	Kernel Reset	4348
Ni_TX_FQ_ADD_PTr	Node i TX FIFO queue r current address pointer register	21120_H+i *2000_H+r *10_H	PNi, 32	BE	Kernel Reset	4348
Ni_TX_FQ_START_ADDr	Node i TX FIFO queue r start address register	21124_H+i *2000_H+r *10_H	PNi, 32	PNi, 32	Kernel Reset	4349
Ni_TX_FQ_SIZEr	Node i TX FIFO queue r size register	21128_H+i *2000_H+r *10_H	PNi, 32	PNi, 32	Kernel Reset	4349
Ni_TX_PQ_STS0	Node i TX priority queue status register	21300_H+i *2000_H	PNi, 32	BE	Kernel Reset	4350
Ni_TX_PQ_STS1	Node i TX priority queue status register	21304_H+i *2000_H	PNi, 32	BE	Kernel Reset	4350

(table continues...)

Table 1067 (continued) Registers overview - CANXL (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
Ni_TX_PQ_CTRL0	Node i TX priority queue control register 0	$2130C_H$+i*2000_H	PNi, 32	PNi, 32	Kernel Reset	4351
Ni_TX_PQ_CTRL1	Node i TX priority queue control register 1	21310_H+i*2000_H	PNi, 32	PNi, 32	Kernel Reset	4351
Ni_TX_PQ_CTRL2	Node i TX priority queue control register 2	21314_H+i*2000_H	PNi, 32	PNi, 32	Kernel Reset	4352
Ni_TX_PQ_START_ADD	Node i TX priority queue start address	21318_H+i*2000_H	PNi, 32	PNi, 32	Kernel Reset	4352
Ni_RX_DESC_ADD_PT	Node i RX descriptor current address pointer	21400_H+i*2000_H	PNi, 32	BE	Kernel Reset	4353
Ni_RX_STATISTICS	Node i RX message counter register	21404_H+i*2000_H	PNi, 32	PNi, 32	Kernel Reset	4353
Ni_RX_FQ_STS0	Node i RX FIFO queue status register 0	21408_H+i*2000_H	PNi, 32	BE	Kernel Reset	4354
Ni_RX_FQ_STS1	Node i RX FIFO queue status register 1	$2140C_H$+i*2000_H	PNi, 32	BE	Kernel Reset	4355
Ni_RX_FQ_STS2	Node i RX FIFO queue status register 2	21410_H+i*2000_H	PNi, 32	BE	Kernel Reset	4355
Ni_RX_FQ_CTRL0	Node i RX FIFO queue control register 0	21414_H+i*2000_H	PNi, 32	PNi, 32	Kernel Reset	4356
Ni_RX_FQ_CTRL1	Node i RX FIFO queue control register 1	21418_H+i*2000_H	PNi, 32	PNi, 32	Kernel Reset	4356
Ni_RX_FQ_CTRL2	Node i RX FIFO queue control register 2	$2141C_H$+i*2000_H	PNi, 32	PNi, 32	Kernel Reset	4357
Ni_RX_FQ_ADD_PTs	Node i RX FIFO queue s current address pointer	21420_H+i*2000_H+s*18_H	PNi, 32	BE	Kernel Reset	4357
Ni_RX_FQ_START_ADDs	Node i RX FIFO queue s link list start address	21424_H+i*2000_H+s*18_H	PNi, 32	PNi, 32	Kernel Reset	4358
Ni_RX_FQ_SIZEs	Node i RX FIFO queue s link list and data container size	21428_H+i*2000_H+s*18_H	PNi, 32	PNi, 32	Kernel Reset	4358
Ni_RX_FQ_DC_START_ADDs	Node i RX FIFO queue s data container start address	$2142C_H$+i*2000_H+s*18_H	PNi, 32	PNi, 32	Kernel Reset	4359
Ni_RX_FQ_RD_ADD_PTs	Node i RX FIFO queue s read address pointer	21430_H+i*2000_H+s*18_H	PNi, 32	PNi, 32	Kernel Reset	4360
Ni_TX_FILTER_CTRL0	Node i TX filter control register 0	21600_H+i*2000_H	PNi, 32	SV, PNi, 32	Kernel Reset	4360
Ni_TX_FILTER_CTRL1	Node i TX filter control register 1	21604_H+i*2000_H	PNi, 32	SV, PNi, 32	Kernel Reset	4361

(table continues...)

Table 1067 (continued) Registers overview - CANXL (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
Ni_TX_FILTER_REFVALt	Node i TX filter reference value register t	21608_H+i *2000_H+t *4	PNi, 32	SV, PNi, 32	Kernel Reset	4362
Ni_RX_FILTER_CTRL	Node i RX filter control register	21680_H+i *2000_H	PNi, 32	SV, PNi, 32	Kernel Reset	4363
Ni_TX_FQ_INT_STS	Node i TX FIFO queue interrupt status register	21700_H+i *2000_H	PNi, 32	PNi, 32	Kernel Reset	4364
Ni_RX_FQ_INT_STS	Node i RX FIFO queue interrupt status register	21704_H+i *2000_H	PNi, 32	PNi, 32	Kernel Reset	4365
Ni_TX_PQ_INT_STS0	Node i TX priority queue interrupt status register 0	21708_H+i *2000_H	PNi, 32	PNi, 32	Kernel Reset	4365
Ni_TX_PQ_INT_STS1	Node i TX priority queue interrupt status register 1	$2170C_H$+ i*2000_H	PNi, 32	PNi, 32	Kernel Reset	4366
Ni_STATS_INT_STS	Node i statistics interrupt status register	21710_H+i *2000_H	PNi, 32	PNi, 32	Kernel Reset	4366
Ni_ERR_INT_STS	Node i error interrupt status register	21714_H+i *2000_H	PNi, 32	PNi, 32	Kernel Reset	4367
Ni_SFTY_INT_STS	Node i safety interrupt status register	21718_H+i *2000_H	PNi, 32	PNi, 32	Kernel Reset	4368
Ni_AXI_ERR_INFO	Node i DMA error information	$2171C_H$+ i*2000_H	PNi, 32	BE	Kernel Reset	4369
Ni_DESC_ERR_INFO0	Node i descriptor error information 0	21720_H+i *2000_H	PNi, 32	BE	Kernel Reset	4370
Ni_DESC_ERR_INFO1	Node i descriptor error information 1	21724_H+i *2000_H	PNi, 32	BE	Kernel Reset	4371
Ni_TX_FILTER_ERR_INFO	Node i TX filter error information	21728_H+i *2000_H	PNi, 32	BE	Kernel Reset	4371
Ni_DEBUG_TEST_CTRL	Node i debug control register	21800_H+i *2000_H	PNi, 32	SV, PNi, 32	Kernel Reset	4372
Ni_INT_TEST0	Node i interrupt test register 0	21804_H+i *2000_H	PNi, 32	PNi, 32	Kernel Reset	4373
Ni_INT_TEST1	Node i interrupt test register 1	21808_H+i *2000_H	PNi, 32	PNi, 32	Kernel Reset	4373
Ni_TX_SCAN_FC	Node i TX-SCAN first candidates register	21810_H+i *2000_H	PNi, 32	BE	Kernel Reset	4376
Ni_TX_SCAN_BC	Node i TX-SCAN best candidates register	21814_H+i *2000_H	PNi, 32	BE	Kernel Reset	4377
Ni_TX_FQ_DESC_VALID	Node i valid TX FIFO queue descriptors in local memory	21818_H+i *2000_H	PNi, 32	BE	Kernel Reset	4378
Ni_TX_PQ_DESC_VALID	Node i valid TX priority queue descriptors in local memory	$2181C_H$+ i*2000_H	PNi, 32	BE	Kernel Reset	4379

(table continues...)

Table 1067 (continued) Registers overview - CANXL (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
Ni_CRC_CTRL	Node i CRC control register	21880_H+i *2000_H	PNi, 32	PNi, 32	Kernel Reset	4379
Ni_CRC_REG	Node i CRC register	21884_H+i *2000_H	PNi, 32	PNi, 32	Kernel Reset	4380
Ni_ENDN	Node i endianness test register	21900_H+i *2000_H	PNi, 32	BE	Kernel Reset	4380
Ni_PREL	Node i PRT release identification register	21904_H+i *2000_H	PNi, 32	BE	Kernel Reset	4380
Ni_STAT	Node i PRT status register	21908_H+i *2000_H	PNi, 32	BE	Kernel Reset	4381
Ni_EVNT	Node i event status flags register	21920_H+i *2000_H	PNi, 32	PNi, 32	Kernel Reset	4383
Ni_LOCK	Node i unlock sequence register	21940_H+i *2000_H	PNi, 32	PNi, 32	Kernel Reset	4384
Ni_CTRL	Node i control register	21944_H+i *2000_H	PNi, 32	PNi, 32	Kernel Reset	4385
Ni_FIMC	Node i fault injection module control register	21948_H+i *2000_H	PNi, 32	SV, PNi, 32	Kernel Reset	4386
Ni_TEST	Node i hardware test functions register	$2194C_H$+ i*2000_H	PNi, 32	SV, PNi, 32	Kernel Reset	4386
Ni_MODE	Node i operating mode register	21960_H+i *2000_H	PNi, 32	PNi, 32	Kernel Reset	4388
Ni_NBTP	Node i arbitration phase nominal bit timing register	21964_H+i *2000_H	PNi, 32	PNi, 32	Kernel Reset	4389
Ni_DBTP	Node i CAN FD data phase bit timing register	21968_H+i *2000_H	PNi, 32	PNi, 32	Kernel Reset	4390
Ni_XBTP	Node i CAN XL data phase bit timing register	$2196C_H$+ i*2000_H	PNi, 32	PNi, 32	Kernel Reset	4391
Ni_PCFG	Node i PWME configuration register	21970_H+i *2000_H	PNi, 32	PNi, 32	Kernel Reset	4392
Ni_FUNC_RAW	Node i functional raw event status register	$21A00_H$+ i*2000_H	PNi, 32	BE	Kernel Reset	4393
Ni_ERR_RAW	Node i error raw event status register	$21A04_H$+ i*2000_H	PNi, 32	BE	Kernel Reset	4395
Ni_SAFETY_RAW	Node i safety raw event status register	$21A08_H$+ i*2000_H	PNi, 32	BE	Kernel Reset	4397
Ni_FUNC_CLR	Node i functional raw event clear register	$21A10_H$+ i*2000_H	PNi, 32	PNi, 32	Kernel Reset	4400
Ni_ERR_CLR	Node i error raw event clear register	$21A14_H$+ i*2000_H	PNi, 32	PNi, 32	Kernel Reset	4402
Ni_SAFETY_CLR	Node i safety raw event clear register	$21A18_H$+ i*2000_H	PNi, 32	PNi, 32	Kernel Reset	4403

(table continues...)

Table 1067 (continued) Registers overview - CANXL (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
Ni_FUNC_ENA	Node i functional raw event enable register	$21A20_H$+ i*2000_H	PNi, 32	PNi, 32	Kernel Reset	4405
Ni_ERR_ENA	Node i error raw event enable register	$21A24_H$+ i*2000_H	PNi, 32	PNi, 32	Kernel Reset	4407
Ni_SAFETY_ENA	Node i safety raw event enable register	$21A28_H$+ i*2000_H	PNi, 32	PNi, 32	Kernel Reset	4409
Ni_CAPTURING_MODE	Node i IRC configuration register	$21A30_H$+ i*2000_H	PNi, 32	BE	Kernel Reset	4411
Ni_HDP	Node i hardware debug port control register	$21A40_H$+ i*2000_H	PNi, 32	PNi, 32	Kernel Reset	4411

22.9.3 Module Identification Register

ID Offset address: 00000_H

Module Identification Register PowerOn Reset value: $00B6\ C001_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MOD_NUM															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MOD_TYPE								MOD_REV							
r								r							

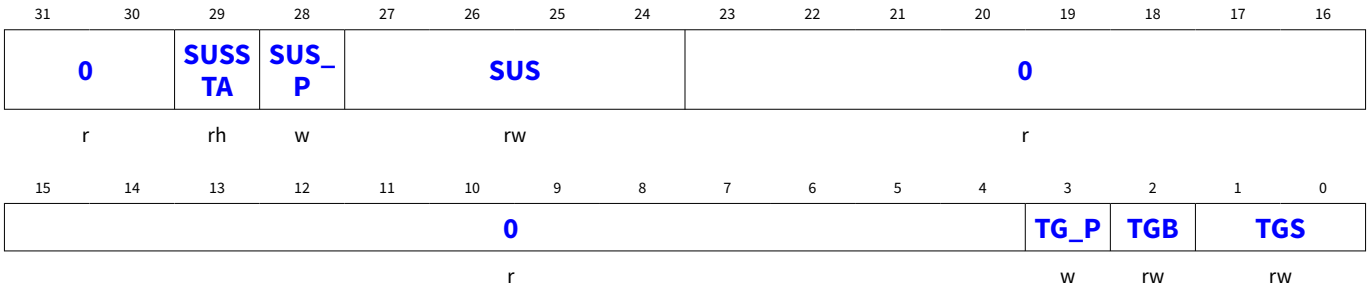
Field	Bits	Type	Description
MOD_REV	7:0	r	**Module Revision** Indicates the revision number of the implementation.
MOD_TYPE	15:8	r	**Module Type** This internal marker is fixed to $C0_H$.
MOD_NUM	31:16	r	**Module Number** Indicates the module identification number.

22.9.4 OCDS Control and Status Register

The OCDS Control and Status register OCS controls the debug and trace behavior by selecting suspend modes and OTGB trigger sets. When OCDS is disabled the suspend control is ineffective.

OCS Offset address: 00004_H

OCDS Control and Status Register Debug Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
TGS	1:0	rw	**Trigger Set for OTGB0/1** 00_B No Trigger Set output 01_B CANXL_HDP16 The HW signals defined by Ni_DEBUG_TEST_CTRL.HDP_SEL[1:0] are selected as OTGB triggers. 10_B Reserved 11_B Reserved
TGB	2	rw	**OTGB0/1 Bus Select** 0_B Trigger Set is output on OTGB0 1_B Trigger Set is output on OTGB1
TG_P	3	w	**TGS, TGB Write Protection** TGS and TGB are only written when TG_P is 1, otherwise unchanged. Read as 0.
SUS	27:24	rw	**OCDS Suspend Control** Controls the sensitivity to the suspend signal coming from the OCDS Trigger Switch (OTGS) Not listed combinations are reserved. 0_H Will not suspend 1_H Hard suspend. Clock is off immediately. Do not use this mode in normal CAN or CAN XL applications, this mode is meant for debugging the peripheral IP. 2_H Soft suspend mode Soft suspend of CAN-XL nodes. The clock is switched off after completion of the ongoing transmission and reception events. **others,** Reserved, do not use
SUS_P	28	w	**SUS Write Protection** SUS is only written when SUS_P is 1, otherwise unchanged. Read as 0.
SUSSTA	29	rh	**Suspend State** 0_B Module is not (yet) suspended 1_B Module is suspended
0	23:4, 31:30	r	**Reserved**

Table 1068 Access mode restrictions of OCS sorted by descending priority

Mode name	Access mode		Description
write 1 to .**TG_P**	rw	TGB, TGS	Set TG_P during write access
write 1 to .**SUS_P**	rw	SUS	Set SUS_P during write access
otherwise	r	SUS, TGB, TGS	

22.9.5 Module Clock Control Register

The Clock Control Register CLC allows the programmer to adapt the functionality and power consumption of the module to the requirements of the application.

Register CLC controls the module clock signal and the reactivity to the sleep signal.

MODULE_CLC Offset address: 00008_H

Module Clock Control Register Application Reset value: $0000\ 0003_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

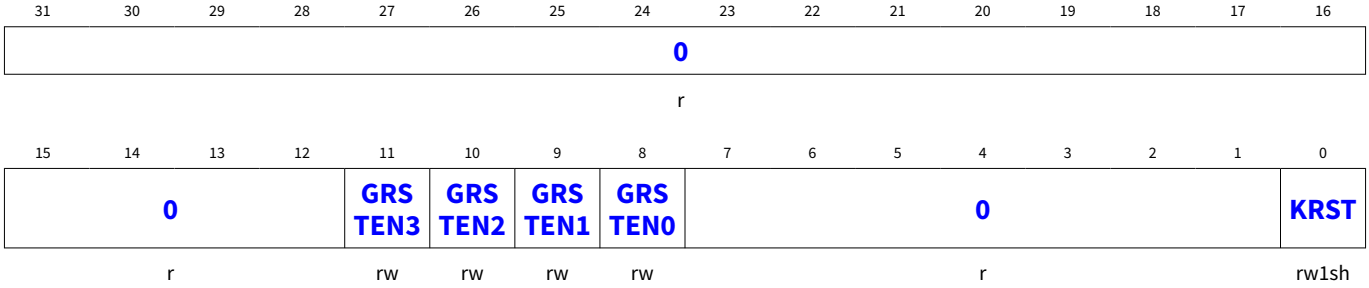
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												EDIS	0	DISS	DISR
r												rw	r	rh	rw

Field	Bits	Type	Description
DISR	0	rw	**Module Disable Request Bit** Used for enable/disable control of the module. This also includes the Slave Interface Bridge and Master Interface Bridge 0_B On request: enable the module clock 1_B Off request: stop the module clock
DISS	1	rh	**Module Disable Status Bit** 0_B Module clock is enabled 1_B Off: module is not clocked
EDIS	3	rw	**Sleep mode enable control** Used to control the module's reaction to sleep mode. 0_B Sleep mode request is enabled and functional 1_B Module disregards the sleep mode control signal
0	2, 31:4	r	**Reserved** Read as 0; should be written with 0.

22.9.6 Module reset control register A

MODULE_RST_CTRLA Offset address: $0000C_H$

Module reset control register A Application Reset value: $0000\ 0000_H$

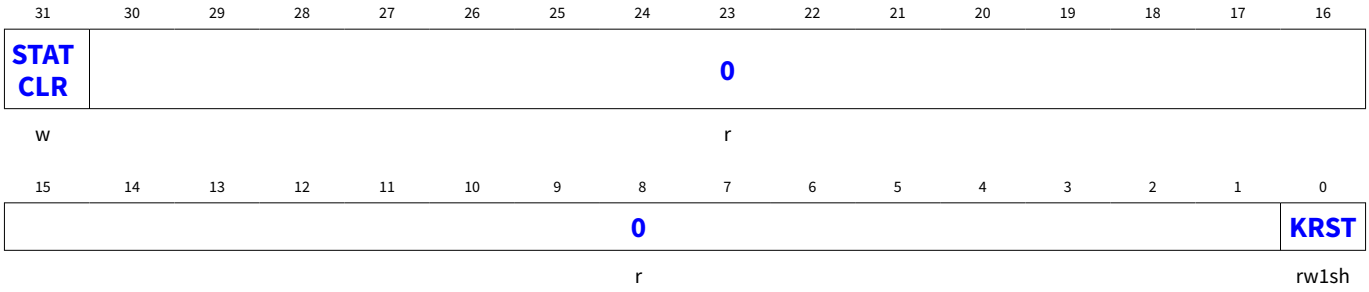


Field	Bits	Type	Description
KRST	0	rw1sh	**Kernel reset** Request a kernel reset. The requested reset is executed if MODULE_RST_CTRLA.KRST=1 and MODULE_RST_CTRLB.KRST=1. KRST is cleared after the kernel reset was executed. 0_B No action 1_B A kernel reset was requested
GRSTENx (x=0-3)	x+8	rw	**Enable for global module reset group x** 0_B Global module reset group x does not have any effect 1_B Global module reset group x results in a kernel reset
0	7:1, 31:12	r	**Reserved** Read as 0; should be written with 0.

22.9.7 Module reset control register B

MODULE_RST_CTRLB Offset address: 00010_H

Module reset control register B Application Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
KRST	0	rw1sh	**Kernel reset** Request a kernel reset. The requested reset is executed if MODULE_RST_CTRLA.KRST=1 and MODULE_RST_CTRLB.KRST=1. KRST is cleared after the kernel reset was executed. 0_B No action 1_B A kernel reset was requested

(table continues...)

(continued)

Field	Bits	Type	Description
STATCLR	31	w	**Kernel reset status clear** Clears all status bits in MODULE_RST_STAT when 1 is written. Read will return 0. 0_B No action 1_B Write with ´1´ clears bits MODULE_RST_STAT.GRSTx and bit MODULE_RST_STAT.KRST .
0	30:1	r	**Reserved** Read as 0; should be written with 0.

22.9.8 Module reset status register

The reset status register contains the status bits for kernel reset (KRST) and the global module reset groups (GRSTx).

MODULE_RST_STAT Offset address: 00014_H

Module reset status register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				GRST3	GRST2	GRST1	GRST0	0							KRST
r				rh	rh	rh	rh	r							rh

Field	Bits	Type	Description
KRST	0	rh	**Kernel reset status** Indicates an executed kernel reset. MODULE_RST_STAT.KRST is set after the execution of a kernel reset in the same clock cycle in which the reset bits are cleared. Clear KRST by setting bit STATCLR in register MODULE_RST_CTRLB.
GRSTx (x=0-3)	x+8	rh	**Status for global module reset group x** 0_B Reset was not triggered by global reset group x 1_B Reset was triggered by global reset group x
0	7:1, 31:12	r	**Reserved** Read as 0.

22.9.9 Module Write access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

MODULE_ACCEN_WRA Offset address: 00018_H

Module Write access enable register A Application Reset value: $1000\ 0003_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access

22.9.10 Module Write access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

MODULE_ACCEN_WRB Offset address: $0001C_H$

Module Write access enable register B Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access
0	31:8	r	**Reserved** Read as 0; should be written with 0.

22.9.11 Module Read access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

MODULE_ACCEN_RDA Offset address: 00020_H

Module Read access enable register A Application Reset value: $FFFF\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access

22.9.12 Module Read access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

MODULE_ACCEN_RDB Offset address: 00024_H

Module Read access enable register B Application Reset value: $0000\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access
0	31:8	r	**Reserved** Read as 0; should be written with 0.

22.9.13 Module VM access enable register

This register defines which virtual machine encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each VMx.

MODULE_ACCEN_VM Offset address: 00028_H

Module VM access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for VM ID q** This bit enables read access to the access protected region for transactions with the VM ID q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for VM ID q** This bit enables write access to the access protected region for transactions with the VM ID q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

22.9.14 Module PRS access enable register

This register defines which protection register set encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each PRSx.

MODULE_ACCEN_PRS Offset address: $0002C_H$

Module PRS access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for PRS q** This bit enables read access to the access protected region for transactions with the PRS q. 0_B Disabled for read access 1_B Enabled for read access

(table continues...)

(continued)

Field	Bits	Type	Description
WRq (q=00-07)	q+16	rw	**Write access enable for PRS q** This bit enables write access to the access protected region for transactions with the PRS q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

22.9.15 PROT Register Endinit

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner and in some PROT states, the secure master to update the PROT state as described in the PROT mechanism.

The PROTE register controls lock / unlock of the local Endinit (E) protected control registers.

PROTE Offset address: 00038_H

PROT Register Endinit Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OWEN	ODEF	TAGID						PRSEN	PRS			VMEN	VM		
w	rw	rw						rw	rw			rw	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												SWEN	STATE		
r												w	rwh		

Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write

(table continues...)

(continued)

Field	Bits	Type	Description
VM	18:16	rw	**Virtual machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual machine definition enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.
PRSEN	23	rw	**Protection set definition enable for PROT owner** 0_B **Disable**: PRS is not part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG-ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; should be written with 0.

Table 1069 Access mode restrictions of PROTE sorted by descending priority

Mode name	Access mode		Description
write 1 to .**SWEN**	rwh	STATE	
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
otherwise	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

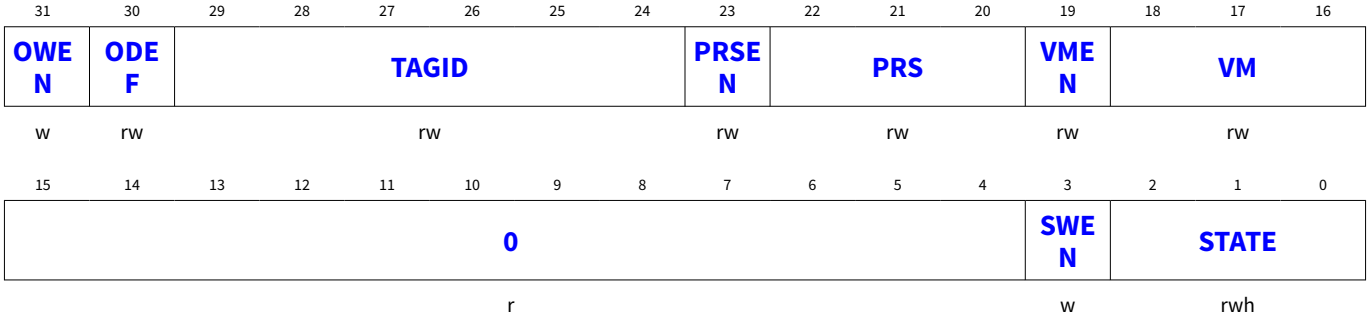
22.9.16 PROT Register Safe Endinit

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner and in some PROT states, the secure master to update the PROT state as described in the PROT mechanism.

The PROTSE register controls lock / unlock of the local Safe Endinit (SE) protected control registers.

PROTSE Offset address: $0003C_H$

PROT Register Safe Endinit Application Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
VM	18:16	rw	**Virtual machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual machine definition enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.
PRSEN	23	rw	**Protection set definition enable for PROT owner** 0_B **Disable**: PRS is not part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG-ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner

(table continues...)

(continued)

Field	Bits	Type	Description
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; should be written with 0.

Table 1070 Access mode restrictions of PROTSE sorted by descending priority

Mode name	Access mode		Description
write 1 to .**SWEN**	rwh	STATE	
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
otherwise	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

22.9.17 Clock enable

This register controls enable or disable of individual CAN XL node clocks.

CLKEN Offset address: 00040_H

Clock enable Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0												N3_CCS	N2_CCS	N1_CCS	N0_CCS
r												rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												N3_CC	N2_CC	N1_CC	N0_CC
r												rw	rw	rw	rw

Field	Bits	Type	Description
Ni_CC (i=0-3)	i	rw	**Node i clock control** This bit-field is used to enable or disable fCANXL and fCANXLH clock of CAN XL Node i. 0_B Disable Node i clocks 1_B Enable Node i clocks

(table continues...)

(continued)

Field	Bits	Type	Description
Ni_CCS (i=0-3)	i+16	rh	**Node i clock control status** This bit-field shows enable or disable status of CAN XL Node i fCANXL and fCANXLH clock. 0_B Node i clocks are disabled 1_B Node i clocks are enabled
0	15:4, 31:20	r	**Reserved, write 0, read as 0**

22.9.18 Node i reset control register A

NODEi_RST_CTRLA (i=0-3) Offset address: $00100_H+i*100_H$

Node i reset control register A Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

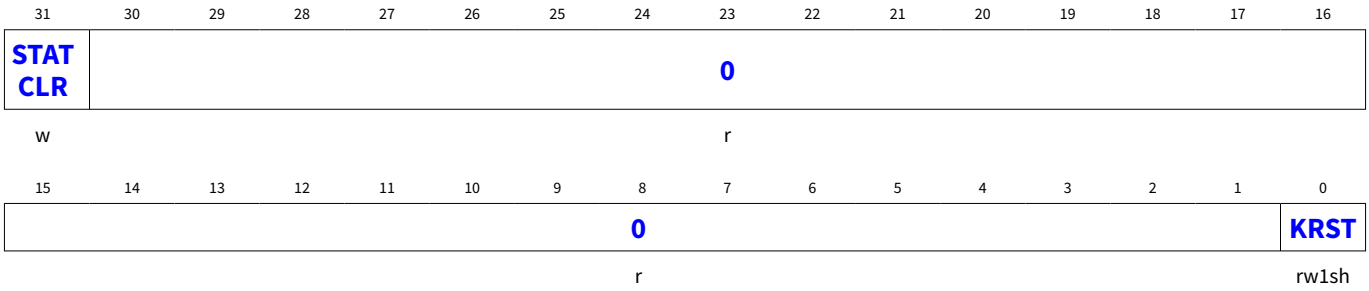
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				GRSTEN3	GRSTEN2	GRSTEN1	GRSTEN0	0							KRST
r				rw	rw	rw	rw	r							rw1sh

Field	Bits	Type	Description
KRST	0	rw1sh	**Kernel reset** Request a kernel reset. The requested reset is executed if NODEi_RST_CTRLA.KRST=1 and NODEi_RST_CTRLB.KRST=1. KRST is cleared after the kernel reset was executed. 0_B No action 1_B A kernel reset was requested
GRSTENx (x=0-3)	x+8	rw	**Enable for Node reset group x** Node group reset is not supported by CANXL. Software must set this bit-field as 0. 0_B Node reset group x does not have any effect 1_B Node reset group x results in a kernel reset - Reserved
0	7:1, 31:12	r	**Reserved** Read as 0; should be written with 0.

22.9.19 Node i reset control register B

NODEi_RST_CTRLB (i=0-3) Offset address: $00104_H+i*100_H$

Node i reset control register B Application Reset value: $0000\ 0000_H$



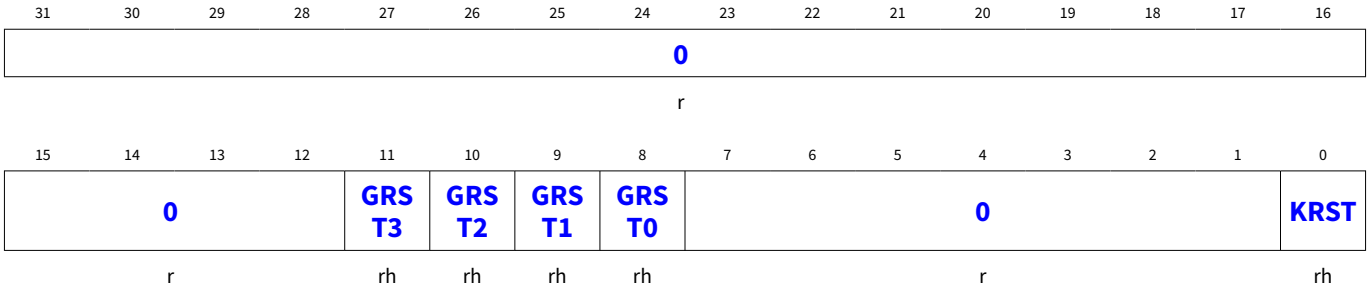
Field	Bits	Type	Description
KRST	0	rw1sh	**Kernel reset** Request a kernel reset. The requested reset is executed if NODEi_RST_CTRLA.KRST=1 and NODEi_RST_CTRLB.KRST=1. KRST is cleared after the kernel reset was executed. 0_B No action 1_B A kernel reset was requested
STATCLR	31	w	**Kernel reset status clear** Clears all status bits in NODEi_RST_STAT when 1 is written. Read will return 0. 0_B No action 1_B Write with ´1´ clears bits NODEi_RST_STAT.GRSTx and bit NODEi_RST_STAT.KRST .
0	30:1	r	**Reserved** Read as 0; should be written with 0.

22.9.20 Node i reset status register

The reset status register contains the status bits for kernel reset (KRST) and the global module reset groups (GRSTx).

NODEi_RST_STAT (i=0-3) Offset address: $00108_H+i*100_H$

Node i reset status register Application Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
KRST	0	rh	**Kernel reset status** Indicates an executed kernel reset. NODEi_RST_STAT.KRST is set after the execution of a kernel reset in the same clock cycle in which the reset bits are cleared. Clear KRST by setting bit STATCLR in register NODEi_RST_CTRLB.
GRSTx (x=0-3)	x+8	rh	**Status for global module reset group x** 0_B Reset was not triggered by global reset group x 1_B Reset was triggered by global reset group x
0	7:1, 31:12	r	**Reserved** Read as 0.

22.9.21 Node i write access enable register A

Per CAN XL node SFR protection.

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

NODEi_ACCENNODE_WRA (i=0-3) Offset address: $0010C_H+i*100_H$

Node i write access enable register A Application Reset value: $1000\ 0003_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access

22.9.22 Node i write access enable register B

Per CAN XL node SFR protection.

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

NODEi_ACCENNODE_WRB (i=0-3) Offset address: $00110_H+i*100_H$

Node i write access enable register B Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access
0	31:8	r	**Reserved** Read as 0; should be written with 0.

22.9.23 Node i read access enable register A

Per CAN XL node SFR protection.

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

NODEi_ACCENNODE_RDA (i=0-3) Offset address: $00114_H+i*100_H$

Node i read access enable register A Application Reset value: FFFF $FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access

22.9.24 Node i read access enable register B

Per CAN XL node SFR protection.

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

NODEi_ACCENNODE_RDB (i=0-3) Offset address: $00118_H+i*100_H$

Node i read access enable register B Application Reset value: $0000\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access
0	31:8	r	**Reserved** Read as 0; should be written with 0.

22.9.25 Node i VM access enable register

Per CAN XL node SFR protection.

This register defines which virtual machine encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each VMx.

NODEi_ACCENNODE_VM (i=0-3) Offset address: $0011C_H+i*100_H$

Node i VM access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for VM ID q** This bit enables read access to the access protected region for transactions with the VM ID q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for VM ID q** This bit enables write access to the access protected region for transactions with the VM ID q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

22.9.26 Node i PRS access enable register

Per CAN XL node SFR protection.

This register defines which protection register set encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each PRSx.

NODEi_ACCENNODE_PRS (i=0-3) Offset address: $00120_H+i*100_H$

Node i PRS access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for PRS q** This bit enables read access to the access protected region for transactions with the PRS q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for PRS q** This bit enables write access to the access protected region for transactions with the PRS q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

22.9.27 Node i region lower address register

Per CAN XL node SFR protection.

This register sets the lower bound of the access protected region. Addresses greater than or equal to the lower bound are considered part of the access protected region.

NODEi_ACCENNODE_RGNLA (i=0-3) Offset address: $00124_H+i*100_H$

Node i region lower address register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADDR															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR										0					
rw										r					

Field	Bits	Type	Description
ADDR	31:6	rw	**Bits 31:6 of the lower bound of the access protected region**
0	5:0	r	**Reserved** Read as 0; should be written with 0.

22.9.28 Node i region upper address register

Per CAN XL node SFR protection.

This register sets the upper bound of the access protected region. Addresses stricly less than the upper bound are considered part of the access protected region.

NODEi_ACCENNODE_RGNUA (i=0-3) Offset address: $00128_H+i*100_H$

Node i region upper address register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADDR															
rw															

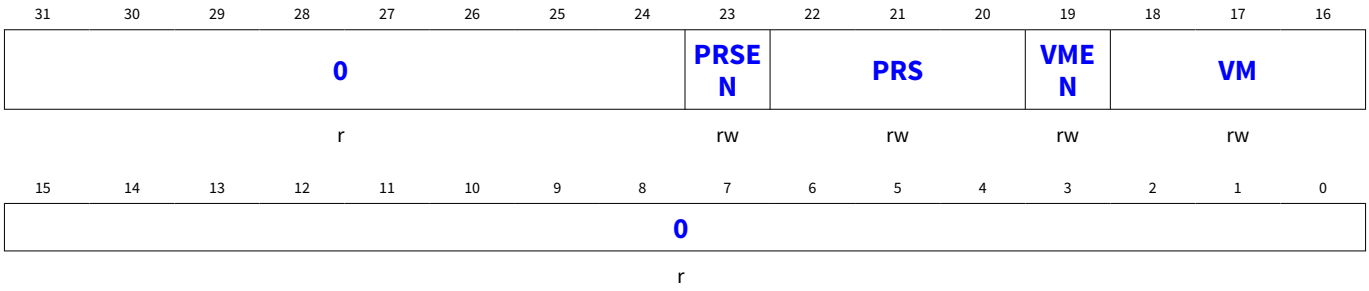
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR										0					
rw										r					

Field	Bits	Type	Description
ADDR	31:6	rw	**Bits 31:6 of the upper bound of the access protected region**
0	5:0	r	**Reserved** Read as 0; should be written with 0.

22.9.29 Node i VM and PRS configuration Register

NODEi_VMPRSCONFIG (i=0-3) Offset address: $0012C_H+i*100_H$

Node i VM and PRS configuration Register Application Reset value: $0000\ 0000_H$



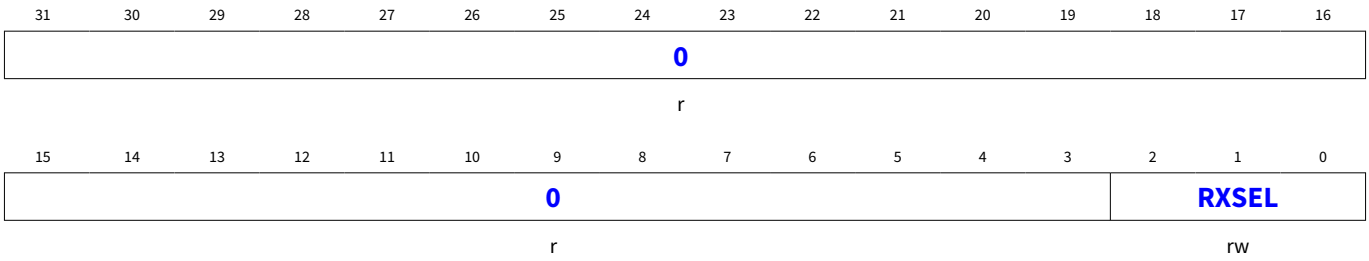
Field	Bits	Type	Description
VM	18:16	rw	**Virtual Machine** CAN-XL Node i's master interface accesses has extended VM tag ID configured in this bit-field
VMEN	19	rw	**Virtual Machine Enable** 0_B **Disable**: VM disabled 1_B **Enable**: VM enabled
PRS	22:20	rw	**Protection Set** CAN XL node i's master interface accesses with extended PRS tag ID configured in this bit-field
PRSEN	23	rw	**Protection Set Enable** 0_B **Disable**: PRS disabled 1_B **Enable**: PRS enabled
0	15:0, 31:24	r	**Reserved** Read as 0; should be written with 0.

22.9.30 Node i Port Control Register

The Node Port Control Register NODEi_PORTCTRL configures the CAN XL bus transmit/receive ports.

NODEi_PORTCTRL (i=0-3) Offset address: $00130_H+i*100_H$

Node i Port Control Register Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
RXSEL	2:0	rw	**Receive Select** RXSEL selects one out of 8 possible receive inputs. The CAN XL receive signal is performed by the selected input. (see the device related chapter for RXSEL)

(table continues...)

(continued)

Field	Bits	Type	Description
0	31:3	r	**Reserved** Read as 0; should be written with 0.

22.9.31 Node i message transfer interrupt event register

A flag is set when the related event is detected, independent of NODEi_MTI_ENA. The flags remain set until the Host CPU clears them by writing a 1 to the corresponding bit position at register NODEi_MTI_CLR.

NODEi_MTI_RAW (i=0-3) Offset address: $00134_H+i*100_H$

Node i message transfer interrupt event register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												TS_CAPTURE	MH_RX_FILTER_IRQ	PRT_RX_EVT	PRT_TX_EVT
r												rh	rh	rh	rh

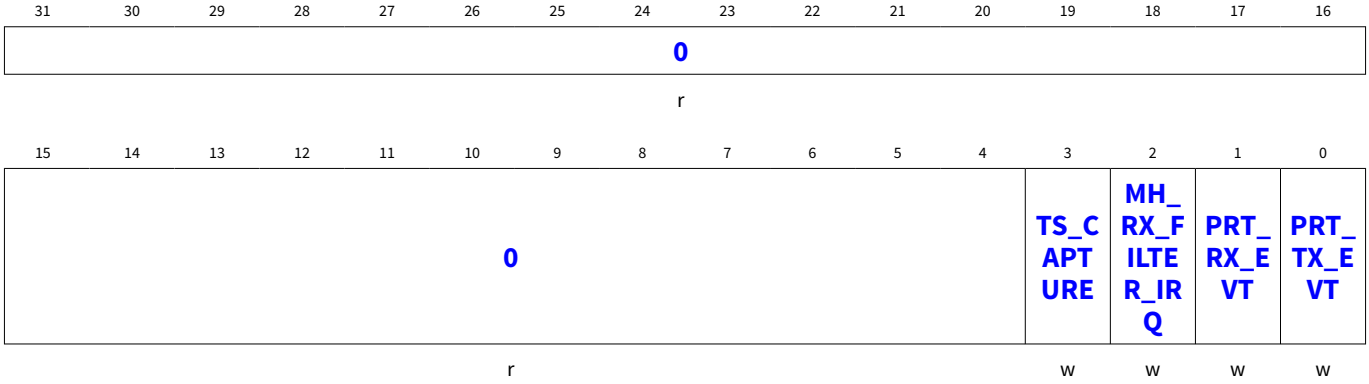
Field	Bits	Type	Description
PRT_TX_EVT	0	rh	**PRT transmitted a valid CAN message**
PRT_RX_EVT	1	rh	**PRT received a valid CAN message**
MH_RX_FILTER_IRQ	2	rh	**Receive filter event** In order to track RX filtering results an interrupt can be triggered when a match is detected on any defined RX filter element
TS_CAPTURE	3	rh	**Timestamp capture event**
0	31:4	r	**Reserved** Read as 0; should be written with 0.

22.9.32 Node i message transfer interrupt clear register

Writing a '1' to a certain bit position clears the correponding bit of register NODEi_MTI_RAW

NODEi_MTI_CLR (i=0-3) Offset address: $00138_H+i*100_H$

Node i message transfer interrupt clear register Kernel Reset value: $0000\ 0000_H$



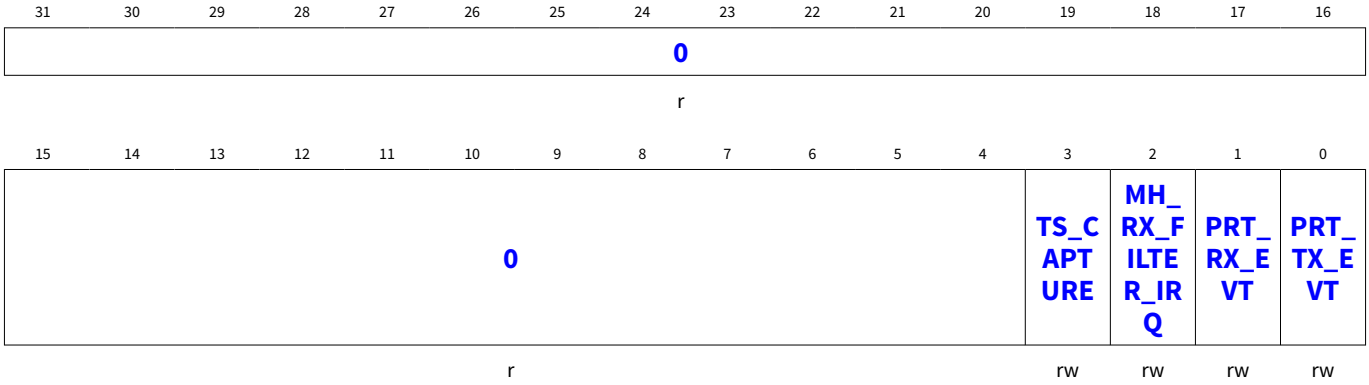
Field	Bits	Type	Description
PRT_TX_EVT	0	w	**Clear PRT transmit event** Clears NODEi_MTI_RAW.PRT_TX_EVT upon writing this bit-field with 1
PRT_RX_EVT	1	w	**Clear PRT receive event** Clears NODEi_MTI_RAW.PRT_RX_EVT upon writing this bit-field with 1
MH_RX_FILTER_IRQ	2	w	**Clear Receive filter event** Clears NODEi_MTI_RAW.MH_RX_FILTER_IRQ upon writing this bit-field with 1
TS_CAPTURE	3	w	**Timestamp capture event** Clears NODEi_MTI_RAW.TS_CAPTURE upon writing this bit-field with 1
0	31:4	r	**Reserved** Read as 0; should be written with 0.

22.9.33 Node i message transfer interrupt enable register

Writing a '1' to a certain bit position enables the correponding bit of register NODEi_MTI_RAW

NODEi_MTI_ENA (i=0-3) Offset address: $0013C_H+i*100_H$

Node i message transfer interrupt enable register Kernel Reset value: $0000\ 0000_H$

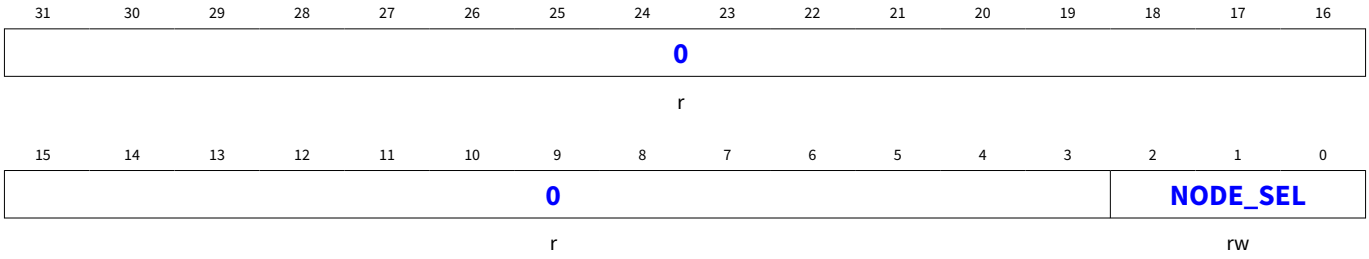


Field	Bits	Type	Description
PRT_TX_EVT	0	rw	**Enable PRT transmit event** Writing this bit-field with 1 enables MTI interrupt trigger on occurrence of corresponding event in MTI SFR. Write with 0 disables the MTI interrupt for the corresponding event.
PRT_RX_EVT	1	rw	**Enable PRT receive event** Writing this bit-field with 1 enables MTI interrupt trigger on occurrence of corresponding event in MTI SFR. Write with 0 disables the MTI interrupt for the corresponding event.
MH_RX_FILTER_IRQ	2	rw	**Enable receive filter event** Writing this bit-field with 1 enables MTI interrupt trigger on occurrence of corresponding event in MTI SFR. Write with 0 disables the MTI interrupt for the corresponding event.
TS_CAPTURE	3	rw	**Enable timestamp capture event** Writing this bit-field with 1 enables MTI interrupt trigger on occurrence of corresponding event in MTI SFR. Write with 0 disables the MTI interrupt for the corresponding event.
0	31:4	r	**Reserved** Read as 0; should be written with 0.

22.9.34 Debug control register

DEBUG_CTL — Offset address: 10004_H

Debug control register — Kernel Reset value: $0000\ 0000_H$



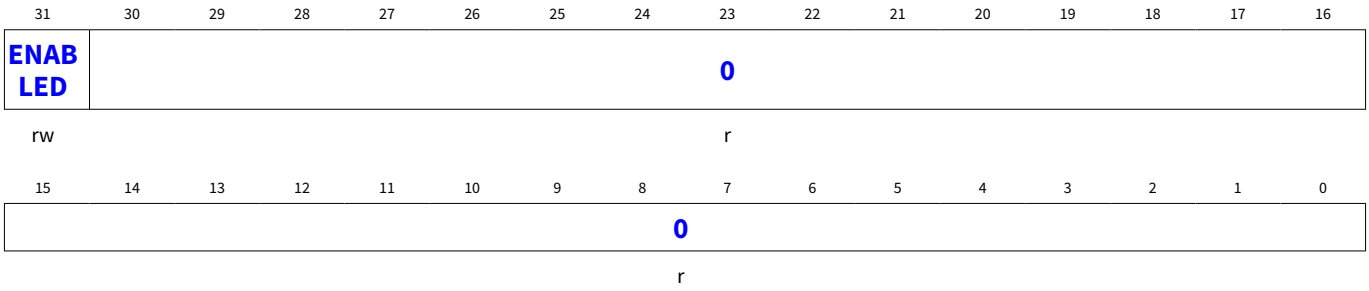
Field	Bits	Type	Description
NODE_SEL	2:0	rw	**Node select** Selection for which X_CAN debug ports are connected to the single top level ports at canxl_base for HDP[15:0], SAMPLE_POINT and STAT_ACT[1:0].
0	31:3	r	**Reserved** Read as all 0's; should be written with all 0's.

22.9.35 Node i timestamp control

This register is used to enable timstamp counters.

Ni_TS_CTL (i=0-3) — Offset address: $20010_H+i*2000_H$

Node i timestamp control — Kernel Reset value: $0000\ 0000_H$



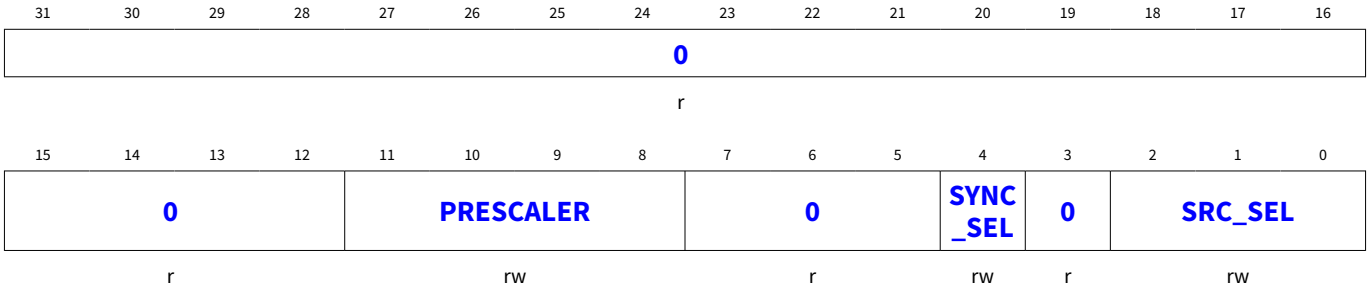
Field	Bits	Type	Description
ENABLED	31	rw	**Timestamp control** When ENABLED is 0 for instance i, TS_CNT* do not increment. Setting the bit to 1 enables incrementing. If disabling a timestamp counter, any X_CAN instance using this counter should be idle and not actively capturing from that timer.
0	30:0	r	**Reserved** Read as all 0's; should be written with all 0's.

22.9.36 Node i timestamp clock control

This register should be adjusted by software when Ni_TS_CTL register ENABLED bit = 0.

Ni_TS_CLOCK_CTL (i=0-3) Offset address: $20014_H+i*2000_H$

Node i timestamp clock control Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
SRC_SEL	2:0	rw	**Clock source select for clk_timebase generation.** 000_B **HOST**: Host clock (fCANXLH) 001_B **EXT_0**: External 0 (STM Trigger) 010_B **EXT_1**: External 1 (GTM Trigger) 011_B **EXT_2**: External 2 (eGTM Trigger) **others,** External 3 (Reserved)
SYNC_SEL	4	rw	**Selection of external or local timebase** 0_B **INTERNAL**: Select internal timestamp counter and clock 1_B **EXTERNAL**: Select external timestamp counter and clock

(table continues...)

(continued)

Field	Bits	Type	Description
PRESCALER	11:8	rw	**Local pre-scaler for clk_timebase generation.** 0_H **DIV_BY_0**: Divide option 0 … F_H **DIV_BY_15**: Divide option 15
0	3, 7:5, 31:12	r	**Reserved** Read as 0; should be written with 0.

22.9.37 Node i timestamp command

Ni_TS_CMD (i=0-3) Offset address: $20020_H+i*2000_H$

Node i timestamp command Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															TS_CLEAR
r															w1c

Field	Bits	Type	Description
TS_CLEAR	0	w1c	**Timestamp counter clear request (resets to 0).** Should be used only if Ni_TS_CTL register ENABLED bit is 0.
0	31:1	r	**Reserved** Read as all 0's; should be written with all 0's.

22.9.38 Node i timestamp counter LSBs

Ni_TS_CNT_LO (i=0-3) Offset address: $20030_H+i*2000_H$

Node i timestamp counter LSBs Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
VALUE															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VALUE															
rh															

Field	Bits	Type	Description
VALUE	31:0	rh	**Bits [31:0] of 64 bit timestamp counter** Note that in the case of TS_PRESENT = 1: If SYNC_SEL = INTERNAL, Ni_TS_CNT_LO is the clk_host synchronised version of the node local timestamp counter (bits [31:0]). If SYNC_SEL = EXTERNAL, Ni_TS_CNT_LO is the clk_host synchronised version of the ext_ts_in external reference timestamp counter (bits [31:0]).

22.9.39 Node i timestamp counter MSBs

Ni_TS_CNT_HI (i=0-3) Offset address: $20034_H+i*2000_H$

Node i timestamp counter MSBs Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
VALUE															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VALUE															
rh															

Field	Bits	Type	Description
VALUE	31:0	rh	**Bits [63:32] of timestamp counter.** Note that in the case of TS_PRESENT = 1: If SYNC_SEL = INTERNAL, Ni_TS_CNT_HI is the clk_host synchronised version of the node local timestamp counter (bits [63:32]). If SYNC_SEL = EXTERNAL, Ni_TS_CNT_HI is the clk_host synchronised version of the ext_ts_in external reference timestamp counter (bits [63:32]).

22.9.40 Node i release identification register

This register is protected by a register bank CRC defined in Ni_CRC_REG register.

Ni_VERSION (i=0-3) Offset address: $21000_H+i*2000_H$

Node i release identification register Kernel Reset value: $0560\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
REL				STEP				SUBSTEP				YEAR			
r				r				r				r			

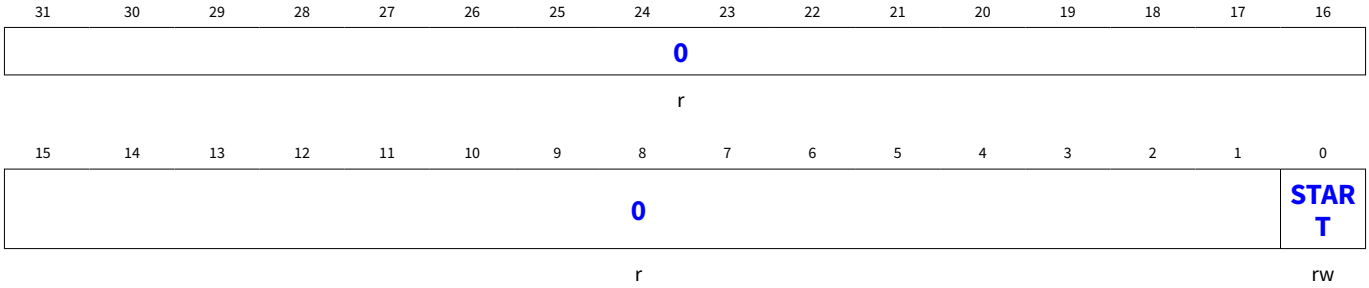
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MON								DAY							
r								r							

Field	Bits	Type	Description
DAY	7:0	r	**DAY** Define the day of the release using a binary coded decimal representation (1 being the first day of the month and so forth). This reset value is defined by the generic parameter DESIGN_TIME_STAMP_G[7:0]. If the generic parameter DESIGN_TIME_STAMP_G is not set, the default value is the one defined here
MON	15:8	r	**MON** Define the month of the release using a binary coded decimal representation (1 being January and so forth). This reset value is defined by the generic parameter DESIGN_TIME_STAMP_G[15:8]. If the generic parameter DESIGN_TIME_STAMP_G is not set, the default value is the one defined here
YEAR	19:16	r	**YEAR** Define the year of the release using a binary coded decimal representation (0 being 2020 and so forth…). This reset value is defined by the generic parameter DESIGN_TIME_STAMP_G[19:16]. If the generic parameter DESIGN_TIME_STAMP_G is not set, the default value is the one defined here
SUBSTEP	23:20	r	**SUBSTEP** Sub-Step value according to Step.
STEP	27:24	r	**STEP** Step value according to Release.
REL	31:28	r	**REL** Release value, used to identify the main release of the X_CAN.

22.9.41 Node i message handler control register

Ni_MH_CTRL (i=0-3) Offset address: $21004_H+i*2000_H$

Node i message handler control register Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
START	0	rw	**START** Before starting any RX/TX FIFO Queues or TX FIFO Queue slots the MH must write 1 to this bit prior launching the PRT. At initial start, as long as the PRT is not started, this bit can be set back to 0. When set to 1, the global configuration registers are write-protected. As soon as the PRT is started, this bit cannot be set to 0. This bit can only be set to back to 0 if Ni_MH_STS.ENABLE = 0 and Ni_MH_STS.BUSY =0. For more details on starting/stopping or restarting the MH, refer to the Programming Guidelines chapter.
0	31:1	r	**Reserved** Read as all 0's; should be written with all 0's.

22.9.42 Node i message handler configuration register

This register is only accessible in write mode if the MH is not started, see Ni_MH_CTRL.START = 0. This register is protected by a register bank CRC defined in Ni_CRC_REG register.

Ni_MH_CFG (i=0-3) Offset address: $21008_H+i*2000_H$

Node i message handler configuration register Kernel Reset value: $0000\ 0700_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0													INST_NUM		
r													rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0					MAX_RETRANS			0							RX_CONT_DC
r					rw			r							rw

Field	Bits	Type	Description
RX_CONT_DC	0	rw	**RX CONT DC** When set to 1, the Continuous mode is active. This mode provides the option to have a linear and continuous memory organization of the RX message data. Only one RX descriptor is used by RX message data and one single data container is required. This bit field register is only accessible in write mode if the MH is not started, see Ni_MH_CTRL.START = 0.
MAX_RETRANS	10:8	rw	**MAX RETRANS** Maximum number of TX message re-transmissions. Different configurations are possible: 0 -> no re-transmission; 1 to 6 -> 1 to 6 re-transmissions; 7-> unlimited re-transmissions; This bit field register is only accessible in write mode if the MH is not started, see Ni_MH_CTRL.START = 0.

(table continues...)

(continued)

Field	Bits	Type	Description
INST_NUM	18:16	rw	**INST NUM** In case that a cluster of X_CAN is defined, this bit field is used as a unique identifier per instance. This identifier is used by the MH to determine if the TX/RX descriptors are fetched by the right instance, see RX/TX description. This bit field register is only accessible in write mode if the MH is not started, see Ni_MH_CTRL.START = 0.
0	7:1, 15:11, 31:19	r	**Reserved** Read as all 0's; should be written with all 0's.

22.9.43 Node i message handler status register

Ni_MH_STS (i=0-3) Offset address: $2100C_H+i*2000_H$

Node i message handler status register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0							CLOCK_ACTIVE	0			ENABLE	0			BUSY
r							rh	r			rh	r			rh

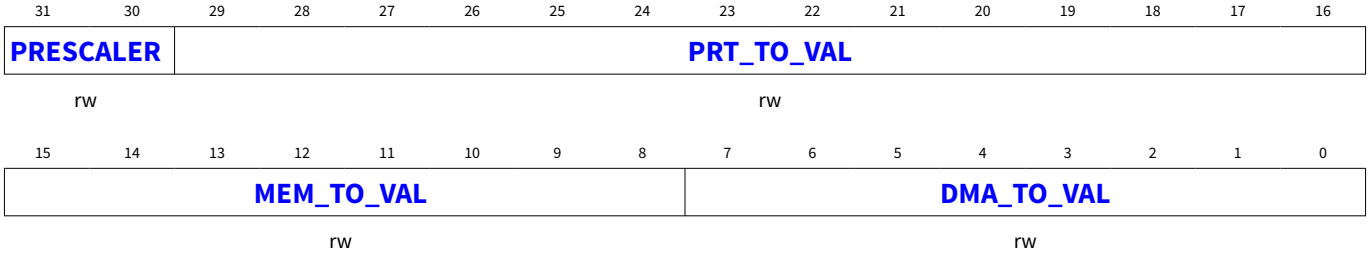
Field	Bits	Type	Description
BUSY	0	rh	**BUSY** This bit is the general busy flag, it is an ORED(RX/TX FIFO Queues and TX Priority Queue slots busy flags).
ENABLE	4	rh	**ENABLE** Value of the ENABLE signal driven by the PRT. The PRT signalizes via ENABLE whether it is active (ENABLE = 1) and requires message handling or not (ENABLE = 0).
CLOCK_ACTIVE	8	rh	**CLOCK ACTIVE** Status of MH core clock: 0 = clock off, 1 = clock on.
0	3:1, 7:5, 31:9	r	**Reserved** Read as all 0's; should be written with all 0's.

22.9.44 Node i message handler safety configuration register

This register is only accessible in write mode if the MH is not started, see Ni_MH_CTRL.START = 0. This register is protected by a register bank CRC defined in Ni_CRC_REG register.

Ni_MH_SFTY_CFG (i=0-3) Offset address: $21010_H+i*2000_H$

Node i message handler safety configuration register Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
DMA_TO_VAL	7:0	rw	**DMA TO VAL** This value is used by the watchdog timer for the DMA_AXI interface and defines the maximum number of timer ticks until a read or write access has to be completed. This value must be configured according to the maximum system latency, expected on the DMA_AXI interface. If this value is set to 0 and Ni_MH_SFTY_CTRL.DMA_TO_EN = 1 then the DMA_TO_ERR interrupt is triggered right away when accessing the S_MEM. This bit field register is only accessible in write mode if the MH is not started, see Ni_MH_CTRL.START = 0.
MEM_TO_VAL	15:8	rw	**MEM TO VAL** This value is used by the watchdog timer for the MEM_AXI interface and defines the maximum number of timer ticks until a read or write access has to be completed. This value must be configured to the expected maximum latency on the MEM_AXI interface. If this value is set to 0 and Ni_MH_SFTY_CTRL.MEM_TO_EN = 1 then the MEM_TO_ERR interrupt is triggered right away when accessing the L_MEM. This bit field register is only accessible in write mode if the MH is not started, see Ni_MH_CTRL.START = 0.
PRT_TO_VAL	29:16	rw	**PRT TO VAL** This value is used by the watchdog timers for the internal RX_MSG and TX_MSG interfaces. It defines the maximum number of timer ticks until a message has to be transferred from PRT to MH respective MH to PRT. The value must be configured according to the CAN frame which requires the longest time to be transported on the CAN bus. If this value is set to 0 and Ni_MH_SFTY_CTRL.PRT_TO_EN = 1 then the DP_TO_ERR interrupt is triggered right away at the beginning of an RX message or when starting a TX message. This bit field register is only accessible in write mode if the MH is not started, see Ni_MH_CTRL.START = 0.
PRESCALER	31:30	rw	**PRESCALER** Prescaler used to generate the timer ticks for the watchdogs. This bit field register is only accessible in write mode if the MH is not started, see Ni_MH_CTRL.START = 0. According to the value a different clock ratio can be selected: 0: clk divided by 32 1: clk divided by 64 2: clk divided by 128 3: clk divided by 512

22.9.45 Node i message handler safety control register

This register is only accessible in write mode if the MH is not started, see Ni_MH_CTRL.START = 0. This register is protected by a register bank CRC defined in Ni_CRC_REG register.

Ni_MH_SFTY_CTRL (i=0-3) Offset address: $21014_H+i*2000_H$

Node i message handler safety control register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0					PRT_TO_EN	MEM_TO_EN	DMA_TO_EN	DMA_CH_CHK_EN	RX_AP_PARITY_EN	TX_AP_PARITY_EN	TX_DP_PARITY_EN	RX_DP_PARITY_EN	MEM_PROT_EN	RX_DESC_CRC_EN	TX_DESC_CRC_EN
r					rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
TX_DESC_CRC_EN	0	rw	**TX DESC CRC EN** When set to 1, the CRC check for the TX descriptors is enabled. This bit field register is only accessible in write mode if the MH is not started, see Ni_MH_CTRL.START = 0.
RX_DESC_CRC_EN	1	rw	**RX DESC CRC EN** When set to 1, the CRC check for the RX descriptors is enabled. This bit field register is only accessible in write mode if the MH is not started, see Ni_MH_CTRL.START = 0.
MEM_PROT_EN	2	rw	**MEM PROT EN** When set to 1, the sfty_err signal from the local memory interface is checked. This bit field register is only accessible in write mode if the MH is not started, see Ni_MH_CTRL.START = 0.
RX_DP_PARITY_EN	3	rw	**RX DP PARITY EN** When set to 1, the data path parity check performed on the RX path is enabled. This bit field register is only accessible in write mode if the MH is not started, see Ni_MH_CTRL.START = 0.
TX_DP_PARITY_EN	4	rw	**TX DP PARITY EN** When set to 1, the data path parity check performed on the TX path is enabled. This bit field register is only accessible in write mode if the MH is not started, see Ni_MH_CTRL.START = 0.
TX_AP_PARITY_EN	5	rw	**TX AP PARITY EN** When set to 1, the address pointer parity check on the TX path is enabled. This bit field register is only accessible in write mode if the MH is not started, see Ni_MH_CTRL.START = 0.
RX_AP_PARITY_EN	6	rw	**RX AP PARITY EN** When set to 1, the address pointer parity check on the RX path is enabled. This bit field register is only accessible in write mode if the MH is not started, see Ni_MH_CTRL.START = 0.

(table continues...)

(continued)

Field	Bits	Type	Description
DMA_CH_CHK_EN	7	rw	**DMA CH CHK EN** When set to 1, the read/write DMA channels routing is checked. This bit field register is only accessible in write mode if the MH is not started, see Ni_MH_CTRL.START = 0.
DMA_TO_EN	8	rw	**DMA TO EN** When set to 1, the watchdog for the DMA_AXI interface is enabled, otherwise disabled. This bit field register is only accessible in write mode if the MH is not started, see Ni_MH_CTRL.START = 0.
MEM_TO_EN	9	rw	**MEM TO EN** When set to 1, the watchdog for the MEM_AXI interface is enabled, otherwise disabled. This bit field register is only accessible in write mode if the MH is not started, see Ni_MH_CTRL.START = 0.
PRT_TO_EN	10	rw	**PRT TO EN** When set to 1, the watchdogs for the internal RX_MSG and TX_MSG interfaces are enabled, otherwise disabled. This bit field register is only accessible in write mode if the MH is not started, see Ni_MH_CTRL.START = 0.
0	31:11	r	**Reserved** Read as all 0's; should be written with all 0's.

22.9.46 Node i RX filter base address register

This register is only accessible in write mode if the MH is not started, see Ni_MH_CTRL.START = 0. This register is protected by a register bank CRC defined in Ni_CRC_REG register.

Ni_RX_FILTER_MEM_ADD (i=0-3) Offset address: $21018_H+i*2000_H$

Node i RX filter base address register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BASE_ADDR															
rw															

Field	Bits	Type	Description
BASE_ADDR	15:0	rw	**BASE ADDR** Define the base address where the RX filter elements are defined in L_MEM (up to 64Kbytes can be addressed). The BASE_ADDR[1:0] bits are always assumed to be 0b00 whatever the value written. This address value must always be word aligned (32bit). This bit field register is only accessible in write mode if the MH is not started, see Ni_MH_CTRL.START = 0.
0	31:16	r	**Reserved** Read as all 0's; should be written with all 0's.

22.9.47 Node i TX descriptor base address register

This register is only accessible in write mode if the MH is not started, see Ni_MH_CTRL.START = 0. This register is protected by a register bank CRC defined in Ni_CRC_REG register.

Ni_TX_DESC_MEM_ADD (i=0-3) Offset address: $2101C_H+i*2000_H$

Node i TX descriptor base address register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PQ_BASE_ADDR															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FQ_BASE_ADDR															
rw															

Field	Bits	Type	Description
FQ_BASE_ADDR	15:0	rw	**FQ BASE ADDR** Define the base address where the TX FIFO Queue descriptors are stored in L_MEM (up to 64Kbytes can be addressed). The FQ_BASE_ADDR[1:0] bits are always assumed to be 0b00 whatever the value written. This address value must always be word aligned (32bit). This bit field register is only accessible in write mode if the MH is not started, see Ni_MH_CTRL.START = 0.
PQ_BASE_ADDR	31:16	rw	**PQ BASE ADDR** Define the base address where the TX Priority Queue descriptors are stored in L_MEM (up to 64Kbytes can be addressed). The PQ_BASE_ADDR[1:0] bits are always assumed to be 0b00 whatever the value written. This address value must always be word aligned (32bit). This bit field register is only accessible in write mode if the MH is not started, see Ni_MH_CTRL.START = 0.

22.9.48 Node i AXI address extension register

This register is only accessible in write mode if the MH is not started, see Ni_MH_CTRL.START = 0. This register is protected by a register bank CRC defined in Ni_CRC_REG register.

Ni_AXI_ADD_EXT (i=0-3) Offset address: $21020_H+i*2000_H$

Node i AXI address extension register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
VAL															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VAL															
rw															

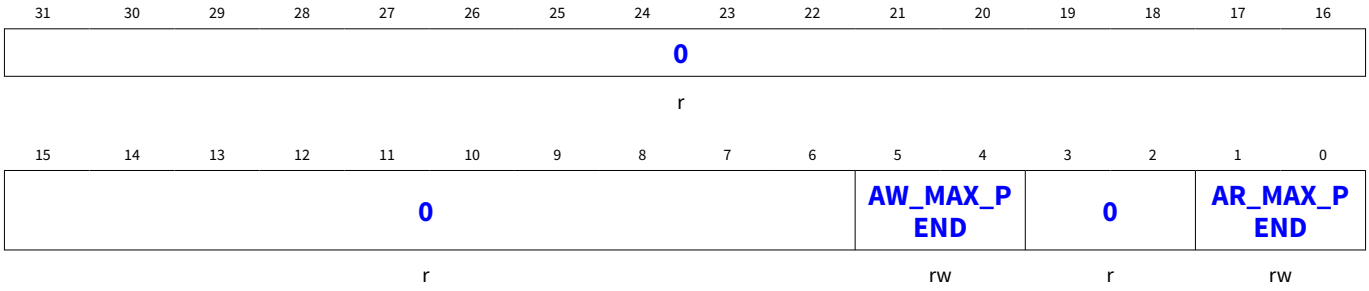
Field	Bits	Type	Description
VAL	31:0	rw	**VAL** Define the MSB of the read/write AXI address bus used on the DMA_AXI interface. If not required, leave the default value and do not connect the upper part of the DMA_AXI read/write address bus. This bit field register is only accessible in write mode if the MH is not started, see Ni_MH_CTRL.START = 0.

22.9.49 Node i AXI parameter register

This register is only accessible in write mode if the MH is not started, see Ni_MH_CTRL.START = 0. This register is protected by a register bank CRC defined in Ni_CRC_REG register.

Ni_AXI_PARAMS (i=0-3) Offset address: $21024_H+i*2000_H$

Node i AXI parameter register Kernel Reset value: $0000\ 0000_H$

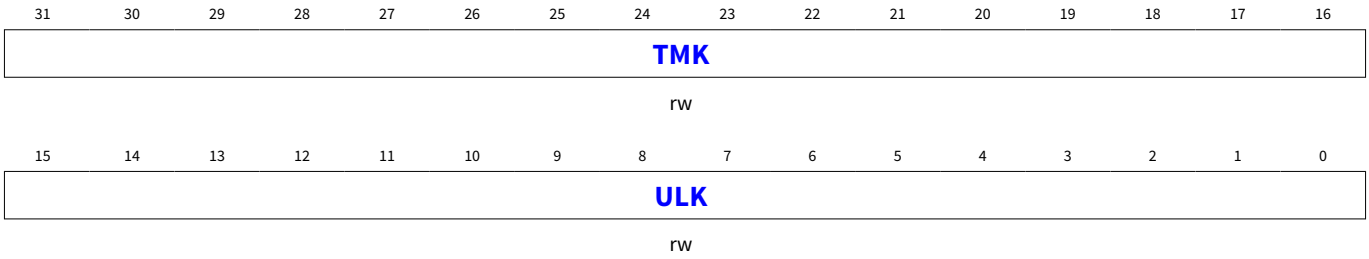


Field	Bits	Type	Description
AR_MAX_PEND	1:0	rw	**AR MAX PEND** AR_MAX_PEND[1:0] defines the maximum read pending transactions on DMA_AXI interface: 0 -> no read transfer; 1 -> 1 outstanding read transaction; 2 -> 2 outstanding read transactions, 3 -> 3 outstanding read transactions. This bit field register is only accessible in write mode if the MH is not started, see Ni_MH_CTRL.START = 0.
AW_MAX_PEND	5:4	rw	**AW MAX PEND** AW_MAX_PEND[1:0] defines the maximum write pending transactions on DMA_AXI interface: 0 -> no write transfer; 1 -> 1 outstanding write transaction allowed; 2 -> 2 outstanding write transactions, 3 -> 3 outstanding write transactions. This bit field register is only accessible in write mode if the MH is not started, see Ni_MH_CTRL.START = 0.
0	3:2, 31:6	r	**Reserved** Read as all 0's; should be written with all 0's.

22.9.50 Node i message handler lock register

Ni_MH_LOCK (i=0-3) Offset address: $21028_H+i*2000_H$

Node i message handler lock register Kernel Reset value: $0000\ 0000_H$

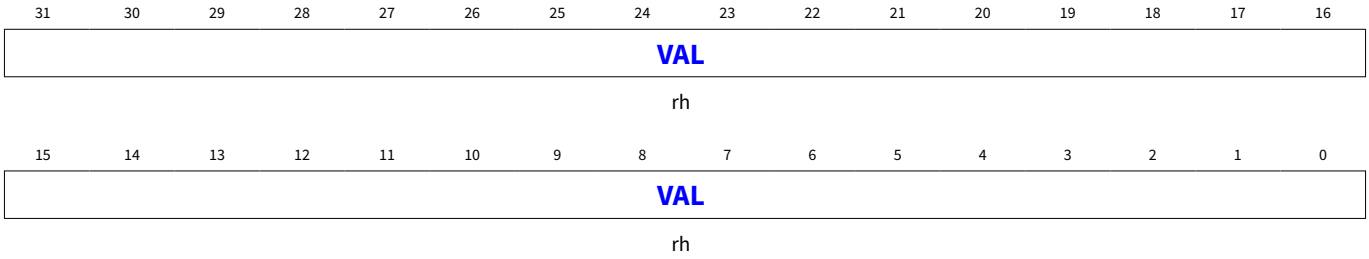


Field	Bits	Type	Description
ULK	15:0	rw	**ULK** Unlock key register. Two consecutive writes to this bit field, starting with 0x1234 and 0x04321, must be done before writing to a register being locked.
TMK	31:16	rw	**TMK** Test mode key register. Two consecutive writes to this bit field, starting with 0x6789 and 0x9876, must be done before writing to the Ni_DEBUG_TEST_CTRL register.

22.9.51 Node i TX descriptor current address pointer register

Ni_TX_DESC_ADD_PT (i=0-3) Offset address: $21100_H+i*2000_H$

Node i TX descriptor current address pointer register Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
VAL	31:0	rh	**VAL** Address used to fetch a TX descriptor for the TX FIFO Queues or TX Priority Queue slots. It could be for several reasons: a new message needs to be fetched from a TX FIFO Queue or a new message is defined in a TX Priority Queue slot. This address value is always word aligned (32bit).

22.9.52 Node i TX message counter register

Ni_TX_STATISTICS (i=0-3) Offset address: $21104_H+i*2000_H$

Node i TX message counter register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0				UNSUCC											
r				rwh											

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				SUCC											
r				rwh											

Field	Bits	Type	Description
SUCC	11:0	rwh	**SUCC** Counter incremented with every successful transmission of a CAN message to the CAN bus. The counter wraps automatically to 0 and can be cleared when writing 0 to the bit field. A STATS_IRQ interrupt is generated when the counter wraps.
UNSUCC	27:16	rwh	**UNSUCC** Counter incremented with every unsuccessful transmission of a CAN message to the CAN bus. The counter wraps automatically to 0 and can be cleared when writing 0 to the bit field. A STATS_IRQ interrupt is generated when the counter wraps.
0	15:12, 31:28	r	**Reserved** Read as all 0's; should be written with all 0's.

22.9.53 Node i TX FIFO queue status register (i=0-3)

Ni_TX_FQ_STS0 (i=0-3) Offset address: $21108_H+i*2000_H$

Node i TX FIFO queue status register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								STOP							
r								rh							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								BUSY							
r								rh							

Field	Bits	Type	Description
BUSY	7:0	rh	**BUSY** When BUSY[n] = 1 the TX FIFO Queue n is active, this means the FIFO Queue is started and running (TX message defined in the TX FIFO Queue n can be processed). When the BUSY[n] = 0, the TX FIFO Queue n is stopped and would require a write to the Ni_TX_FQ_CTRL0.START[n] to make it active again. A TX FIFO Queue can go inactive if the END bit in the last TX descriptor of a TX message is set. In this case the, the BUSY[n] = 0 can occur only if the TX header descriptor of this last message has been acknowledged for the TX FIFO Queue n. When the TX FIFO Queue n is aborted, the BUSY[n] flag is set to 0 only when no acknowledge is pending.

(table continues...)

(continued)

Field	Bits	Type	Description
STOP	23:16	rh	**STOP** When STOP[n] = 1 the TX FIFO Queue n is on hold, this means the FIFO Queue is started and running but waits for the SW to keep going. The STOP[n] can be set only if the BUSY[n] = 1. Several root causes may lead to this state: an error is detected, or a TX descriptor is not valid. To identify the potential issues, refer to the Ni_TX_FQ_STS1 register. In order to keep going with the TX FIFO Queue n, a write to the Ni_TX_FQ_CTRL0.START[n] is required. When BUSY[n] = 0, this bit is automatically set to 0
0	15:8, 31:24	r	**Reserved** Read as all 0's; should be written with all 0's.

22.9.54 Node i TX FIFO queue status register (i=0-3)

Ni_TX_FQ_STS1 (i=0-3) Offset address: $2110C_H+i*2000_H$

Node i TX FIFO queue status register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								ERROR							
r								rh							

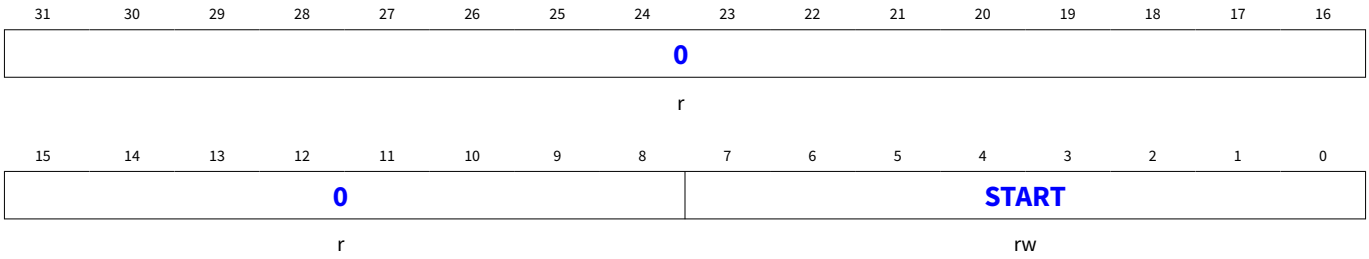
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								UNVALID							
r								rh							

Field	Bits	Type	Description
UNVALID	7:0	rh	**UNVALID** When UNVALID[n] = 1 the TX FIFO Queue n is on hold due to an TX descriptor with VALID=0 was loaded.
ERROR	23:16	rh	**ERROR** When ERROR[n] = 1 the TX FIFO Queue n is on hold due to an inconsistent TX descriptor was loaded, see chapter Descriptor Protection.
0	15:8, 31:24	r	**Reserved** Read as all 0's; should be written with all 0's.

22.9.55 Node i TX FIFO queue control register 0

Ni_TX_FQ_CTRL0 (i=0-3) Offset address: $21110_H+i*2000_H$

Node i TX FIFO queue control register 0 Kernel Reset value: $0000\ 0000_H$



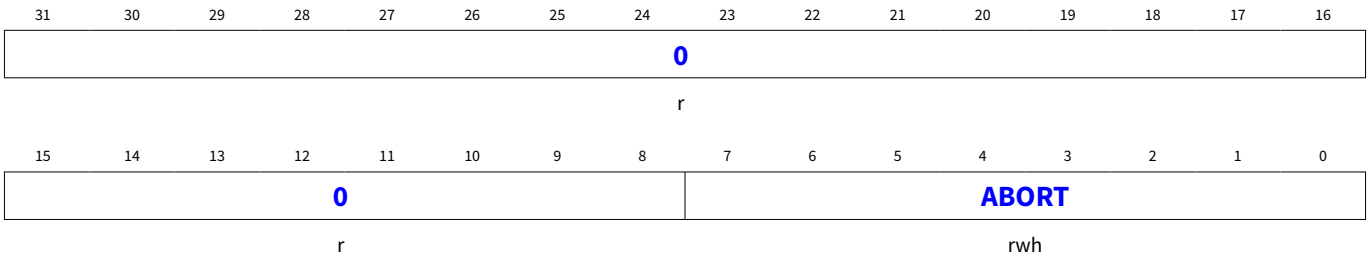
Field	Bits	Type	Description
START	7:0	rw	**START** When writing a 1 to the START[n], the TX FIFO Queue n is started. This bit is autocleared. Once started, the Ni_TX_FQ_STS0.BUSY[n] is set to 1. The MH must be started prior to any TX FIFO Queue start (Ni_MH_STS.BUSY set to 1). A TX FIFO Queue n can only be started if Ni_TX_FQ_CTRL2.ENABLE[n] is set to 1 and in order to avoid a dead lock situation with the PRT, the ENABLE signal from the PRT is high.
0	31:8	r	**Reserved** Read as all 0's; should be written with all 0's.

22.9.56 Node i TX FIFO queue control register 1

This register is only accessible in write mode if the unlock key sequence has been performed prior to write.

Ni_TX_FQ_CTRL1 (i=0-3) Offset address: $21114_H+i*2000_H$

Node i TX FIFO queue control register 1 Kernel Reset value: $0000\ 0000_H$

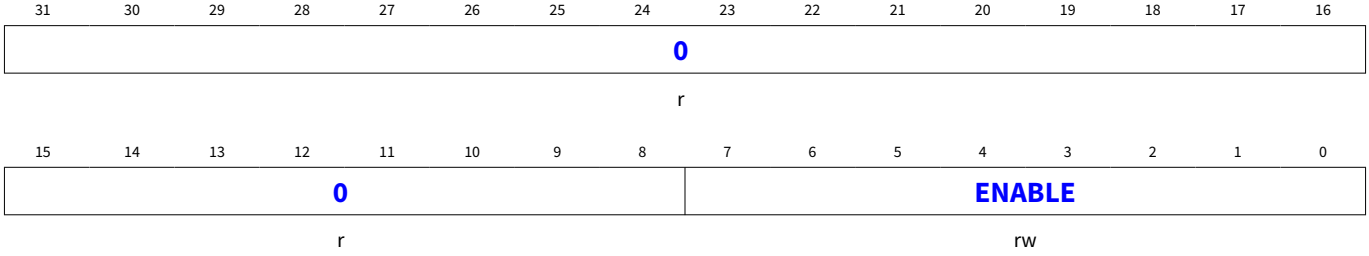


Field	Bits	Type	Description
ABORT	7:0	rwh	**ABORT** When ABORT[n] is set to 1, the TX FIFO Queue n is aborted. Once set to 1, the MH will abort all pending transaction related to the TX FIFO Queue n whenever required. This bit must be set back to 0 only when the TX FIFO Queue n is inactive, Ni_TX_FQ_STS0.BUSY[n] = 0. This bit field register is only accessible in write mode if the unlock key sequence has been performed prior to write.
0	31:8	r	**Reserved** Read as all 0's; should be written with all 0's.

22.9.57 Node i TX FIFO queue control register 2

Ni_TX_FQ_CTRL2 (i=0-3) Offset address: $21118_H+i*2000_H$

Node i TX FIFO queue control register 2 Kernel Reset value: $0000\ 0000_H$

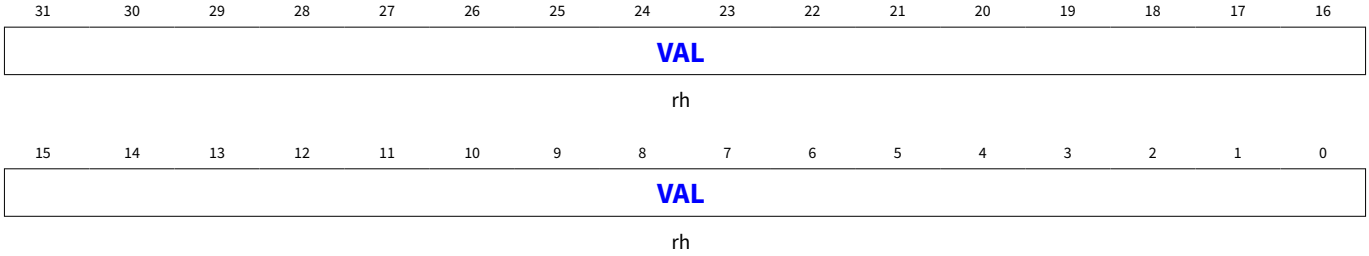


Field	Bits	Type	Description
ENABLE	7:0	rw	**ENABLE** When ENABLE[n] is set to 1, the TX FIFO Queue n is enabled. A TX FIFO Queue cannot be started if it is not enabled. Aborting a not started TX FIFO Queue has no effect.
0	31:8	r	**Reserved** Read as all 0's; should be written with all 0's.

22.9.58 Node i TX FIFO queue r current address pointer register

Ni_TX_FQ_ADD_PTr (i=0-3;r=0-7) Offset address: $21120_H+i*2000_H+r*10_H$

Node i TX FIFO queue r current address pointer register Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
VAL	31:0	rh	**VAL** Provide the header descriptor address of the TX message being in used by the arbiter for the TX FIFO Queue. To follow TX descriptors over time while running TX FIFO Queues, refer to the Ni_TX_DESC_ADD_PT register. This address value is always word aligned (32bit).

22.9.59 Node i TX FIFO queue r start address register

This register is only accessible in write mode if the TX FIFO Queue r is not busy, see BUSY flag in Ni_TX_FQ_STS0 register. This register is protected by a register bank CRC defined in Ni_CRC_REG register.

Ni_TX_FQ_START_ADDr (i=0-3;r=0-7) Offset address: $21124_H+i*2000_H+r*10_H$

Node i TX FIFO queue r start address register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
VAL															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VAL															
rw															

Field	Bits	Type	Description
VAL	31:0	rw	**VAL** Define the start address of the TX FIFO Queue link list descriptor in the system memory. The VAL[1:0] bits are always assumed to be 0b00 whatever the value written. This address value must always be word aligned (32bit). This bit field register is only accessible in write mode if the TX FIFO Queue 0 is not busy, see BUSY flag in Ni_TX_FQ_STS0 register.

22.9.60 Node i TX FIFO queue r size register

This register is only accessible in write mode if the TX FIFO Queue r is not busy, see BUSY flag in Ni_TX_FQ_STS0 register. This register is protected by a register bank CRC defined in Ni_CRC_REG register.

Ni_TX_FQ_SIZEr (i=0-3;r=0-7) Offset address: $21128_H+i*2000_H+r*10_H$

Node i TX FIFO queue r size register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0						MAX_DESC									
r						rw									

Field	Bits	Type	Description
MAX_DESC	9:0	rw	**MAX DESC** Define the maximum number of TX descriptors in the TX FIFO Queue link list descriptors. It is important to note that MAX_DESC = 0 does not prevent the TX FIFO Queue to be enabled and started. An active and running TX FIFO Queue with MAX_DESC = 0 is not allowed and will result in a DESC_ERR interrupt if no TX descriptor is defined. The memory size to allocate is MAX_DESC * 32bytes for MAX_DESC > = 1. This bit field register is only accessible in write mode if the TX FIFO Queue 0 is not busy, see BUSY flag in Ni_TX_FQ_STS0 register.
0	31:10	r	**Reserved** Read as all 0's; should be written with all 0's.

22.9.61 Node i TX priority queue status register (i=0-3)

Ni_TX_PQ_STS0 (i=0-3) Offset address: $21300_H+i*2000_H$

Node i TX priority queue status register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BUSY															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BUSY															
rh															

Field	Bits	Type	Description
BUSY	31:0	rh	**BUSY** When BUSY[n] = 1, the TX Priority Queue slot n is busy, which means that the TX descriptor in the slot n is being loaded in L_MEM and considered by the TX-Scan. As long as this bit remains high, the message attached to the slot n has not been sent yet. The BUSY[n] = 0 can occur only if the TX header descriptor of the slot n has been acknowledged.

22.9.62 Node i TX priority queue status register (i=0-3)

Ni_TX_PQ_STS1 (i=0-3) Offset address: $21304_H+i*2000_H$

Node i TX priority queue status register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SENT															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SENT															
rh															

Field	Bits	Type	Description
SENT	31:0	rh	**SENT** When SENT[n] = 1 the TX message assigned to the TX Priority Queue slot n has been transmitted and the TX descriptor attached to the slot n is acknowledged. This bit will be cleared once a new start on this slot will occur.

22.9.63 Node i TX priority queue control register 0

Ni_TX_PQ_CTRL0 (i=0-3) Offset address: $2130C_H+i*2000_H$

Node i TX priority queue control register 0 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
START															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
START															
rw															

Field	Bits	Type	Description
START	31:0	rw	**START** When writing a 1 to the START[n], the TX Priority Queue slot n is started and running. This bit is autocleared and once started, the Ni_TX_PQ_STS0.BUSY[n] is set to 1. The MH must be started prior to any TX Priority Queue slot start (Ni_MH_STS.BUSY set to 1). A TX Priority Queue slot n can only be started if Ni_TX_PQ_CTRL2.ENABLE[n] is set to 1 and in order to avoid a dead lock situation with the PRT, the ENABLE signal from the PRT is high.

22.9.64 Node i TX priority queue control register 1

This register is only accessible in write mode if the unlock key sequence has been performed prior to write.

Ni_TX_PQ_CTRL1 (i=0-3) Offset address: $21310_H+i*2000_H$

Node i TX priority queue control register 1 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ABORT															
rwh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ABORT															
rwh															

Field	Bits	Type	Description
ABORT	31:0	rwh	**ABORT** When ABORT[n] is set to 1, the TX Priority Queue slot n is aborted. This bit must be set back to 0 only when the TX Priority Queue slot n is inactive, Ni_TX_PQ_STS0.BUSY[n] = 0. A TX message attached to a slot can only be aborted if it is not stored in the two internal buffers holding the two best candidates for the next TX message. Despite a TX message is aborted, it may have been sent, check the Ni_TX_PQ_STS1.SENT[n] bit register for the slot n. This bit field register is only accessible in write mode if the unlock key sequence has been performed prior to write.

22.9.65 Node i TX priority queue control register 2

Ni_TX_PQ_CTRL2 (i=0-3) Offset address: $21314_H+i*2000_H$

Node i TX priority queue control register 2 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ENABLE															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ENABLE															
rw															

Field	Bits	Type	Description
ENABLE	31:0	rw	**ENABLE** When ENABLE[n] is set to 1, the slot n in the TX Priority Queue is enabled. A TX Priority Queue slot cannot be started if not enabled. Aborting a not started slot n has no effect

22.9.66 Node i TX priority queue start address

This register is only accessible in write mode if the TX Priority Queue is not busy, see BUSY flag in TX_PQ_STS register. It means TX_PQ_STS register is equal to 0x0. This register is protected by a register bank CRC defined in Ni_CRC_REG register.

Ni_TX_PQ_START_ADD (i=0-3) Offset address: $21318_H+i*2000_H$

Node i TX priority queue start address Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
VAL															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VAL															
rw															

Field	Bits	Type	Description
VAL	31:0	rw	**VAL** Define the start address of the TX Priority Queue in the system memory. All TX header descriptors in the TX Priority Queue are continuously defined from this start address. The VAL[1:0] bits are always assumed to be 0b00 whatever the value written. This address value must always be word aligned (32bit). This bit field register is only accessible in write mode if the TX Priority Queue is not busy, see BUSY flag in TX_PQ_STS register

22.9.67 Node i RX descriptor current address pointer

Ni_RX_DESC_ADD_PT (i=0-3) Offset address: $21400_H+i*2000_H$

Node i RX descriptor current address pointer Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
VAL															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VAL															
rh															

Field	Bits	Type	Description
VAL	31:0	rh	**VAL** Provide the address used to fetch the current RX descriptor. This address value is always word aligned (32bit).

22.9.68 Node i RX message counter register

Ni_RX_STATISTICS (i=0-3) Offset address: $21404_H+i*2000_H$

Node i RX message counter register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0				UNSUCC											
r				rwh											

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				SUCC											
r				rwh											

Field	Bits	Type	Description
SUCC	11:0	rwh	**SUCC** Counter incremented with every successful reception of a CAN message from the CAN bus. The counter wraps automatically to 0 and can be cleared when writing 0x00 to the bit field. An interrupt is generated when the counter wraps.
UNSUCC	27:16	rwh	**UNSUCC** Counter incremented with every unsuccessful reception of a CAN message from the CAN bus. The counter wraps automatically to 0 and can be cleared when writing 0x00 to the bit field. An interrupt is generated when the counter wraps.
0	15:12, 31:28	r	**Reserved** Read as all 0's; should be written with all 0's.

22.9.69 Node i RX FIFO queue status register 0

Ni_RX_FQ_STS0 (i=0-3) Offset address: $21408_H+i*2000_H$

Node i RX FIFO queue status register 0 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								STOP							
r								rh							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								BUSY							
r								rh							

Field	Bits	Type	Description
BUSY	7:0	rh	**BUSY** When BUSY[n] = 1 the RX FIFO Queue n is busy, this means the FIFO Queue is started and running (RX message to be written to the RX FIFO Queue can be processed). When the BUSY[n] = 0, the RX FIFO Queue n is stopped and would require a write to the Ni_RX_FQ_CTRL0.START[n] to make it active again. When the RX FIFO Queue n is aborted, the BUSY[n] flag is set to 0 only when no acknowledge is pending.
STOP	23:16	rh	**STOP** When STOP[n] = 1 the RX FIFO Queue n is on hold, it means started but waiting for the SW to react. The STOP[n] can be set only if the BUSY[n] = 1. Several root causes may lead to the RX FIFO Queue n to stop: an error is detected, or an RX descriptor is not valid, or the FIFO is full. To identify the potential issues, refer to the Ni_RX_FQ_STS1 and Ni_RX_FQ_STS2 registers. In order to keep going with the RX FIFO Queue n, a write to the Ni_RX_FQ_CTRL0.START[n] is required. When BUSY[n] = 0, this bit is automatically set to 0
0	15:8, 31:24	r	**Reserved** Read as all 0's; should be written with all 0's.

22.9.70 Node i RX FIFO queue status register 1

Ni_RX_FQ_STS1 (i=0-3) Offset address: $2140C_H+i*2000_H$

Node i RX FIFO queue status register 1 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								ERROR							
r								rh							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								UNVALID							
r								rh							

Field	Bits	Type	Description
UNVALID	7:0	rh	**UNVALID** When UNVALID[n] = 1 the RX FIFO Queue n is on hold due to an RX descriptor detected with VALID=0.
ERROR	23:16	rh	**ERROR** When ERROR[n] = 1 the RX FIFO Queue n is on hold due to an inconsistent RX descriptor being loaded, see chapter Descriptor Protection.
0	15:8, 31:24	r	**Reserved** Read as all 0's; should be written with all 0's.

22.9.71 Node i RX FIFO queue status register 2

Ni_RX_FQ_STS2 (i=0-3) Offset address: $21410_H+i*2000_H$

Node i RX FIFO queue status register 2 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

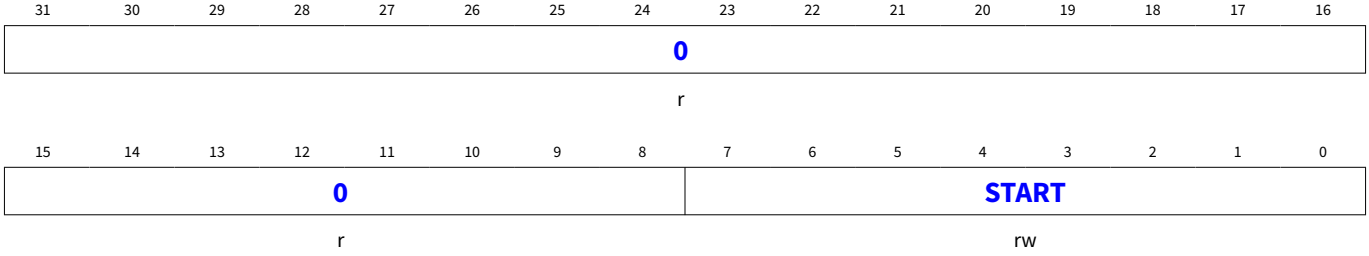
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								DC_FULL							
r								rh							

Field	Bits	Type	Description
DC_FULL	7:0	rh	**DC FULL** When DC_FULL[n] = 1 the RX FIFO Queue n is stopped due to the RX FIFO Queue n being full. This register is relevant only for the Continuous Mode as in Normal mode, there is no need to provide such information to the MH
0	31:8	r	**Reserved** Read as all 0's; should be written with all 0's.

22.9.72 Node i RX FIFO queue control register 0

Ni_RX_FQ_CTRL0 (i=0-3) Offset address: $21414_H+i*2000_H$

Node i RX FIFO queue control register 0 Kernel Reset value: $0000\ 0000_H$



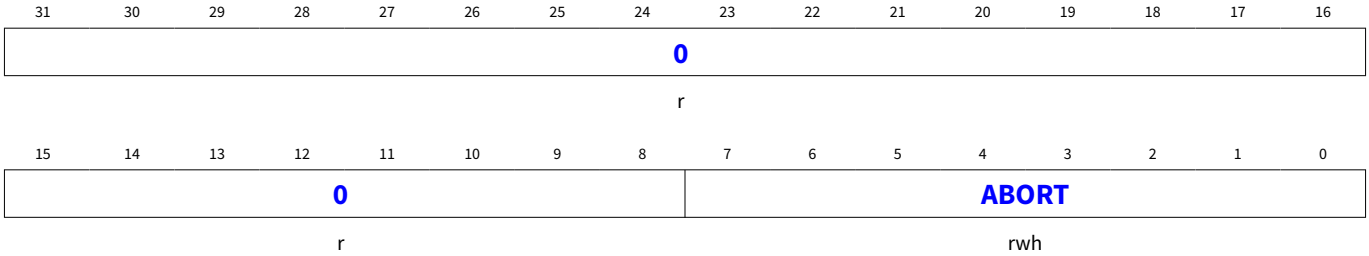
Field	Bits	Type	Description
START	7:0	rw	**START** When writing a 1 to the START[n], the RX FIFO Queue n is started. This bit is autocleared and once started, the Ni_RX_FQ_STS0.BUSY[n] is set to 1. The MH must be started prior to any RX FIFO Queue start (Ni_MH_STS.BUSY set to 1). An RX FIFO Queue n can only be started if Ni_RX_FQ_CTRL2.ENABLE[n] is set to 1.
0	31:8	r	**Reserved** Read as all 0's; should be written with all 0's.

22.9.73 Node i RX FIFO queue control register 1

This register is only accessible in write mode if the unlock key sequence has been performed prior to write.

Ni_RX_FQ_CTRL1 (i=0-3) Offset address: $21418_H+i*2000_H$

Node i RX FIFO queue control register 1 Kernel Reset value: $0000\ 0000_H$

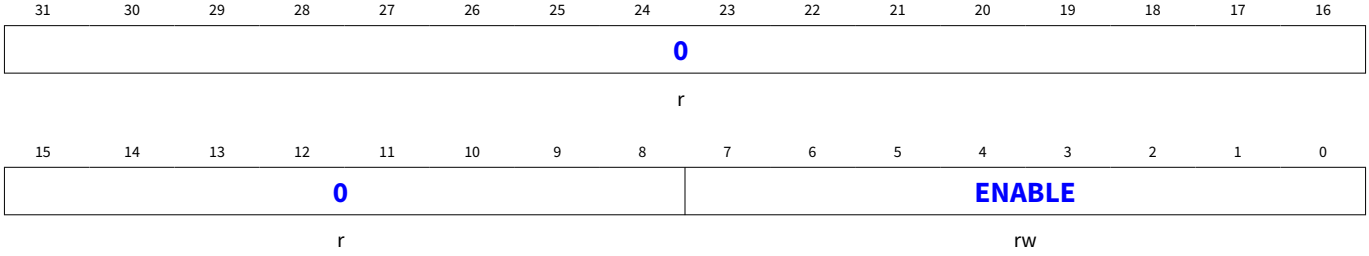


Field	Bits	Type	Description
ABORT	7:0	rwh	**ABORT** When ABORT[n] is set to 1, the RX FIFO Queue n is aborted. Once set to 1, the MH will abort all pending transactions related to the RX FIFO Queue n whenever required. The abort can be effective only if the RX FIFO Queue n is enabled. This bit must be set back to 0 only when the RX FIFO Queue n is inactive, Ni_RX_FQ_STS0.BUSY[n] = 0. This bit field register is only accessible in write mode if the unlock key sequence has been performed prior to write.
0	31:8	r	**Reserved** Read as all 0's; should be written with all 0's.

22.9.74 Node i RX FIFO queue control register 2

Ni_RX_FQ_CTRL2 (i=0-3) Offset address: $2141C_H+i*2000_H$

Node i RX FIFO queue control register 2 Kernel Reset value: $0000\ 0000_H$

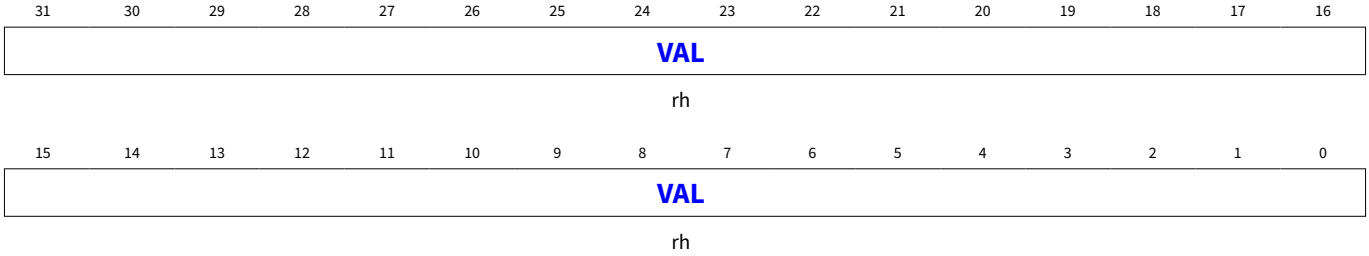


Field	Bits	Type	Description
ENABLE	7:0	rw	**ENABLE** When ENABLE[n] is set to 1, the RX FIFO Queue n is enabled. The RX FIFO Queue n cannot be started if not enabled. The abort of an RX FIFO Queue n not started would have no effect.
0	31:8	r	**Reserved** Read as all 0's; should be written with all 0's.

22.9.75 Node i RX FIFO queue s current address pointer

Ni_RX_FQ_ADD_PTs (i=0-3;s=0-7) Offset address: $21420_H+i*2000_H+s*18_H$

Node i RX FIFO queue s current address pointer Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
VAL	31:0	rh	**VAL** Provide the current RX Header Descriptor address pointer for the RX FIFO Queue 0 in the system memory. To follow RX descriptor over time, refer to the Ni_RX_DESC_ADD_PT register. This address value is always word aligned (32bit).

22.9.76 Node i RX FIFO queue s link list start address

This register is only accessible in write mode if the RX FIFO Queue is not busy, see BUSY flag in Ni_RX_FQ_STS0 register. This register is protected by a register bank CRC defined in Ni_CRC_REG register.

Ni_RX_FQ_START_ADDs (i=0-3;s=0-7) Offset address: $21424_H+i*2000_H+s*18_H$

Node i RX FIFO queue s link list start address Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
VAL															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VAL															
rw															

Field	Bits	Type	Description
VAL	31:0	rw	**VAL** Define the start address of the RX FIFO Queue link list descriptor in system memory. The VAL[1:0] bits are always assumed to be 0b00 whatever the value written. This address value must always be word aligned (32bit). This register is only accessible in write mode if the RX FIFO Queue 0 is not busy, see BUSY flag in Ni_RX_FQ_STS0 register

22.9.77 Node i RX FIFO queue s link list and data container size

This register is only accessible in write mode if the RX FIFO Queue is not busy, see BUSY flag in Ni_RX_FQ_STS0 register. This register is protected by a register bank CRC defined in Ni_CRC_REG register.

Ni_RX_FQ_SIZEs (i=0-3;s=0-7) Offset address: $21428_H+i*2000_H+s*18_H$

Node i RX FIFO queue s link list and data container size Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0				DC_SIZE											
r				rw											

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0						MAX_DESC									
r						rw									

Field	Bits	Type	Description
MAX_DESC	9:0	rw	**MAX DESC** Define the maximum number of descriptors in the RX FIFO Queue link list. It is important to note that MAX_DESC = 0 does not prevent the RX FIFO Queue to be enabled and started. An active and running RX FIFO Queue with MAX_DESC = 0 is not allowed and will result in a DESC_ERR interrupt if no RX descriptor is defined. The size to be allocated to the link list must be equal to MAX_DESC * 16bytes for MAX_DESC >= 1. This register is only accessible in write mode if the RX FIFO Queue 0 is not busy, see BUSY flag in Ni_RX_FQ_STS0 register
DC_SIZE	27:16	rw	**DC SIZE** In Normal mode only the DC_SIZE[6:0] is used to define the maximum size of an RX data container for the RX FIFO Queue. The data container size is DC_SIZE[6:0] * 32bytes and one is attached to every RX descriptor. In continuous mode, it defines the size of the single data container used to write all RX messages. The overall data container size is DC_SIZE[11:0] * 32bytes for MAX_DESC > = 1. When set to 0, the RX FIFO Queue can be enabled but not started. This register is only accessible in write mode if the RX FIFO Queue 0 is not busy, see BUSY flag in Ni_RX_FQ_STS0 register
0	15:10, 31:28	r	**Reserved** Read as all 0's; should be written with all 0's.

22.9.78 Node i RX FIFO queue s data container start address

This register is accessible in write mode if the RX FIFO Queue is not busy, see BUSY flag in Ni_RX_FQ_STS0 register. This register is protected by a register bank CRC defined in Ni_CRC_REG register. This register is used only in Continuous Mode

Ni_RX_FQ_DC_START_ADDs (i=0-3;s=0-7) Offset address: $2142C_H+i*2000_H+s*18_H$

Node i RX FIFO queue s data container start address Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
VAL															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VAL															
rw															

Field	Bits	Type	Description
VAL	31:0	rw	**VAL** Define the Data Container Start Address in system memory. This bit field is relevant only when the MH is configured in Continuous Mode. The VAL[1:0] bits are always assumed to be 0b00 whatever the value written. This address value must always be word aligned (32bit). This register is only accessible in write mode if the RX FIFO Queue 0 is not busy, see BUSY flag in Ni_RX_FQ_STS0 register

22.9.79 Node i RX FIFO queue s read address pointer

This register is used only in Continuous Mode.

Ni_RX_FQ_RD_ADD_PTs (i=0-3;s=0-7) Offset address: $21430_H+i*2000_H+s*18_H$

Node i RX FIFO queue s read address pointer Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
VAL															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VAL															
rw															

Field	Bits	Type	Description
VAL	31:0	rw	**VAL** The SW uses this register to indicate the Data Read Address of the RX message being read to the MH. This address must point to the last word of the RX message considered in the data container. This bit field is relevant only when the MH is configured in Continuous mode. The MH uses this information to ensure that enough memory space is available to write the next message. For an initial start, it is mandatory to set VAL[1:0] to 0b11, to avoid Ni_RX_FQ_RD_ADD_PT0 register to be equal to the Ni_RX_FQ_START_ADDR0 registers. Excepted for the initial value, the address value must always be word aligned (32bit), VAL[1:0] must be set to 0b00.

22.9.80 Node i TX filter control register 0

This register is only accessible in write mode if the MH is not busy, see BUSY flag in Ni_MH_STS register. The register is accessible in write access in privileged mode only. This register is protected by a register bank CRC defined in Ni_CRC_REG register.

Ni_TX_FILTER_CTRL0 (i=0-3) Offset address: $21600_H+i*2000_H$

Node i TX filter control register 0 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0											IRQ_EN	EN	CC_CAN	CAN_FD	MODE
r											rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MASK								COMB							
rw								rw							

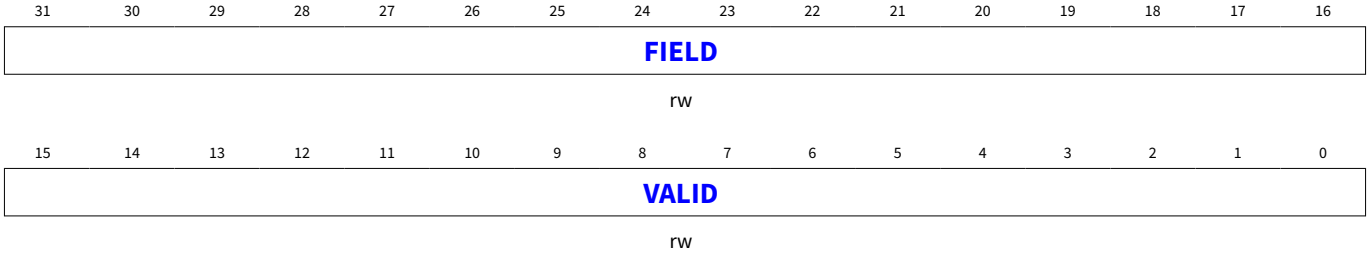
Field	Bits	Type	Description
COMB	7:0	rw	**COMB** When COMB[n] =1 the comparison attached to the reference values (REF_VAL0 and REF_VAL1) or (REF_VAL2 and REF_VAL3) are required to accept a TX message. This bit field register is only accessible in write mode if the MH is not busy, see BUSY flag in Ni_MH_STS register
MASK	15:8	rw	**MASK** When MASK[n] =1 the reference values REF_VAL0/1 or REF_VAL2/3 are combined to define a value (REF_VAL0 or REF_VAL2) and a mask (REF_VAL1 or REF_VAL3). Otherwise, the comparison uses the REF_VAL0/1/2/3 bit field as reference value only. This bit field register is only accessible in write mode if the MH is not busy, see BUSY flag in Ni_MH_STS register
MODE	16	rw	**MODE** When set to 1 accept on match, otherwise reject on match. This bit field register is only accessible in write mode if the MH is not busy, see BUSY flag in Ni_MH_STS register
CAN_FD	17	rw	**CAN FD** When set to 1 reject CAN-FD messages, otherwise accept them. This bit field register is only accessible in write mode if the MH is not busy, see BUSY flag in Ni_MH_STS register
CC_CAN	18	rw	**CC CAN** When set to 1 reject Classic CAN messages, otherwise accept them. This bit field register is only accessible in write mode if the MH is not busy, see BUSY flag in Ni_MH_STS register
EN	19	rw	**EN** When set to 1, enable the TX filter for all TX message to be sent. This bit field register is only accessible in write mode if the MH is not busy, see BUSY flag in Ni_MH_STS register
IRQ_EN	20	rw	**IRQ EN** When set to 1, enable the interrupt tx_filter_irq to be triggered. The interrupt is triggered when a message is rejected. This bit field register is only accessible in write mode if the MH is not busy, see BUSY flag in Ni_MH_STS register
0	31:21	r	**Reserved** Read as all 0's; should be written with all 0's.

22.9.81 Node i TX filter control register 1

This register is only accessible in write mode if the MH is not busy, see BUSY flag in Ni_MH_STS register. The register is accessible in write access in privileged mode only. This register is protected by a register bank CRC defined in Ni_CRC_REG register.

Ni_TX_FILTER_CTRL1 (i=0-3) Offset address: $21604_H+i*2000_H$

Node i TX filter control register 1 Kernel Reset value: $0000\ 0000_H$



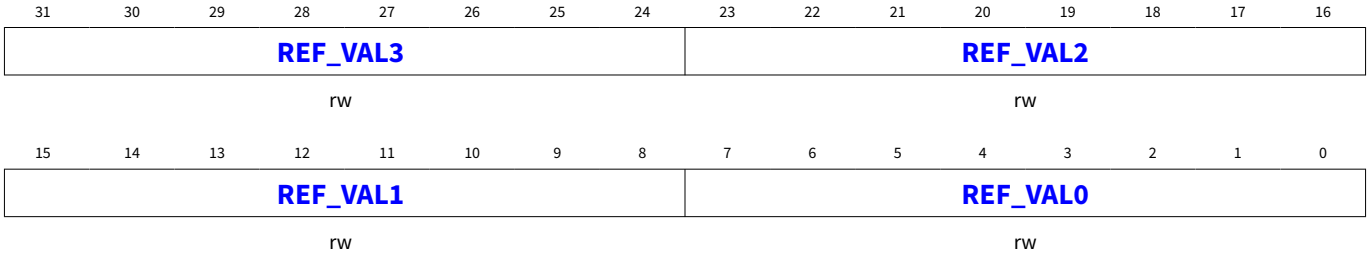
Field	Bits	Type	Description
VALID	15:0	rw	**VALID** When VALID[n] = 1 the reference value defined for the TX filter n is valid. This bit field register is only accessible in write mode if the MH is not busy, see BUSY flag in Ni_MH_STS register. The valid reference value used is defined as follow: VALID[n] is assigned to Ni_TX_FILTER_REFVAL0.REF_VAL{n} (n € {0, 1, 2, 3}) VALID[n+4] is assigned to Ni_TX_FILTER_REFVAL1.REF_VAL{n} (n € {0, 1, 2, 3}) VALID[n+8] is assigned to Ni_TX_FILTER_REFVAL2.REF_VAL{n} (n € {0, 1, 2, 3}) VALID[n+12] is assigned to Ni_TX_FILTER_REFVAL3.REF_VAL{n} (n € {0, 1, 2, 3})
FIELD	31:16	rw	**FIELD** When FIELD[n] = 1 the TX filter element n is considering SDT, otherwise VCID, to compare with the reference value defined in Ni_TX_FILTER_REFVAL0/1/2/3. This bit field register is only accessible in write mode if the MH is not busy, see BUSY flag in Ni_MH_STS register. The reference value to be set is defined as follow: FIELD[n] is assigned to Ni_TX_FILTER_REFVAL0.REF_VAL{n} (n € {0, 1, 2, 3}) FIELD[n+4] is assigned to Ni_TX_FILTER_REFVAL1.REF_VAL{n} (n € {0, 1, 2, 3}) FIELD[n+8] is assigned to Ni_TX_FILTER_REFVAL2.REF_VAL{n} (n € {0, 1, 2, 3}) FIELD[n+12] is assigned to Ni_TX_FILTER_REFVAL3.REF_VAL{n} (n € {0, 1, 2, 3})

22.9.82 Node i TX filter reference value register t

This register is only accessible in write mode if the MH is not busy, see BUSY flag in Ni_MH_STS register. The register is accessible in write access in privileged mode only. This register is protected by a register bank CRC defined in Ni_CRC_REG register.

Ni_TX_FILTER_REFVALt (i=0-3;t=0-3) Offset address: $21608_H+i*2000_H+t*4$

Node i TX filter reference value register t Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
REF_VAL0	7:0	rw	**REF VAL0** Define the reference value 0. This bit field register is only accessible in write mode if the MH is not busy, see BUSY flag in Ni_MH_STS register
REF_VAL1	15:8	rw	**REF VAL1** Define the reference value 1. This bit field register is only accessible in write mode if the MH is not busy, see BUSY flag in Ni_MH_STS register
REF_VAL2	23:16	rw	**REF VAL2** Define the reference value 2. This bit field register is only accessible in write mode if the MH is not busy, see BUSY flag in Ni_MH_STS register
REF_VAL3	31:24	rw	**REF VAL3** Define the reference value 3. This bit field register is only accessible in write mode if the MH is not busy, see BUSY flag in Ni_MH_STS register

22.9.83 Node i RX filter control register

This register is only accessible in write mode if the MH is not busy, see BUSY flag in Ni_MH_STS register. The register is accessible in write access in privileged mode only. This register is protected by a register bank CRC defined in Ni_CRC_REG register.

Ni_RX_FILTER_CTRL (i=0-3) Offset address: $21680_H+i*2000_H$

Node i RX filter control register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0										ANFF	ANMF	0	ANMF_FQ		
r										rw	rw	r	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			THRESHOLD					NB_FE							
r			rw					rw							

Field	Bits	Type	Description
NB_FE	7:0	rw	**NB FE** Define the number of RX filter elements that are defined in the local memory. When set to 0, all RX messages are accepted and stored in the RX FIFO Queue number defined by ANMF_FQ[3:0]. This bit field register is only accessible in write mode if the MH is not busy, see BUSY flag in Ni_MH_STS register

(table continues...)

(continued)

Field	Bits	Type	Description
THRESHOLD	12:8	rw	**THRESHOLD** THRESHOLD defines the latest point in time to wait for the result of the RX filtering process., Once this limit is reached, the MH starts fetching an RX descriptor from S_MEM. THRESHOLD value is only used when greater than 0 and ANFF bit set to 1. See chapter "RX Filter" for an explanation how to configure THRESHOLD. When the RX filtering is not providing the result before the threshold of the RX DMA FIFO is reached, the RX message is sent to the default RX FIFO Queue mentioned in the ANMF_FQ[2:0] (only enabled when ANFF set to 1). When the level is over the threshold and the RX filtering result is already known, no action is taken. Threshold is given in number of word of 32bit. This bit field register is only accessible in write mode if the MH is not busy, see BUSY flag in Ni_MH_STS register
ANMF_FQ	18:16	rw	**ANMF FQ** Define the default RX FIFO Queue number (from 0 to 7) when non matching frames are accepted (ANMF = 1) AND/OR when the threshold mechanism is active (ANFF = 1). This bit field register is only accessible in write mode if the MH is not busy, see BUSY flag in Ni_MH_STS register
ANMF	20	rw	**ANMF** When set to 1, non matching frames are accepted, otherwise rejected. It is mandatory to have the default RX FIFO Queue defined in the ANMF_FQ[2:0] bit field, enabled and started (see Ni_RX_FQ_CTRL2 and Ni_RX_FQ_CTRL0 registers). This bit field register is only accessible in write mode if the MH is not busy, see BUSY flag in Ni_MH_STS register
ANFF	21	rw	**ANFF** When set to 1, frames not filtered in time and over the DMA RX FIFO level defined in THRESHOLD[4:0], are routed to the default RX FIFO Queue (defined by the ANMF_FQ[2:0] bit field). It is mandatory to have the default RX FIFO Queue defined in the ANMF_FQ[2:0] bit field, enabled and started (see Ni_RX_FQ_CTRL2 and Ni_RX_FQ_CTRL0 registers). This bit field register is only accessible in write mode if the MH is not busy, see BUSY flag in Ni_MH_STS register
0	15:13, 19, 31:22	r	**Reserved** Read as all 0's; should be written with all 0's.

22.9.84 Node i TX FIFO queue interrupt status register

Ni_TX_FQ_INT_STS (i=0-3) Offset address: $21700_H+i*2000_H$

Node i TX FIFO queue interrupt status register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								UNVALID							
r								rw1ch							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								SENT							
r								rw1ch							

Field	Bits	Type	Description
SENT	7:0	rw1ch	**SENT** When SENT[n] = 1, a TX message was sent from the TX FIFO Queue n and writing a 1 clears the bit.
UNVALID	23:16	rw1ch	**UNVALID** When TX FIFO Queue n loads a TX descriptor with VALID = 0, the bit UNVALID[n] will be set. Writing 1 to UNVALID[n] clears the bit.
0	15:8, 31:24	r	**Reserved** Read as all 0's; should be written with all 0's.

22.9.85 Node i RX FIFO queue interrupt status register

Ni_RX_FQ_INT_STS (i=0-3) Offset address: $21704_H+i*2000_H$

Node i RX FIFO queue interrupt status register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								UNVALID							
r								rwh							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RECEIVED							
r								rw1ch							

Field	Bits	Type	Description
RECEIVED	7:0	rw1ch	**RECEIVED** When RECEIVED[n] = 1, an RX message was received in the RX FIFO Queue n, writing a 1 clears the bit.
UNVALID	23:16	rwh	**UNVALID** When RX FIFO Queue n loads an RX descriptor with VALID=0, the bit UNVALID[n] will be set. Writing 1 to UNVALID[n] clears the bit.
0	15:8, 31:24	r	**Reserved** Read as all 0's; should be written with all 0's.

22.9.86 Node i TX priority queue interrupt status register 0

Ni_TX_PQ_INT_STS0 (i=0-3) Offset address: $21708_H+i*2000_H$

Node i TX priority queue interrupt status register 0 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SENT															
rw1ch															

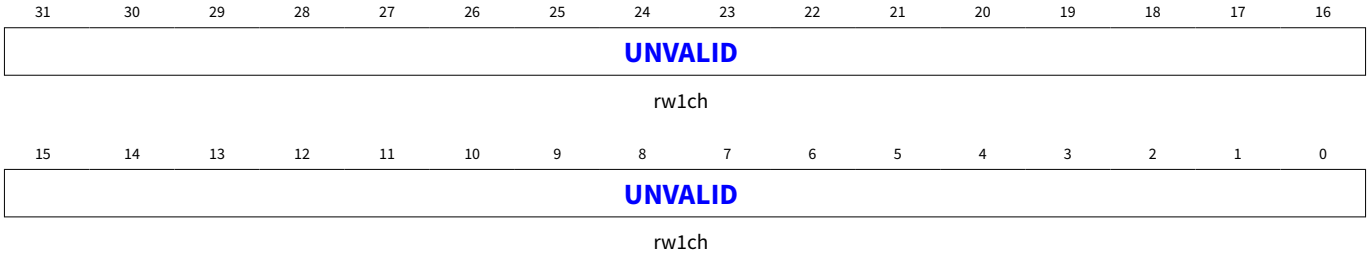
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SENT															
rw1ch															

Field	Bits	Type	Description
SENT	31:0	rw1ch	**SENT** When SENT[n] = 1 TX message was sent from the TX Priority Queue slot n, writing a 1 clears the bit.

22.9.87 Node i TX priority queue interrupt status register 1

Ni_TX_PQ_INT_STS1 (i=0-3) Offset address: $2170C_H+i*2000_H$

Node i TX priority queue interrupt status register Kernel Reset value: $0000\ 0000_H$

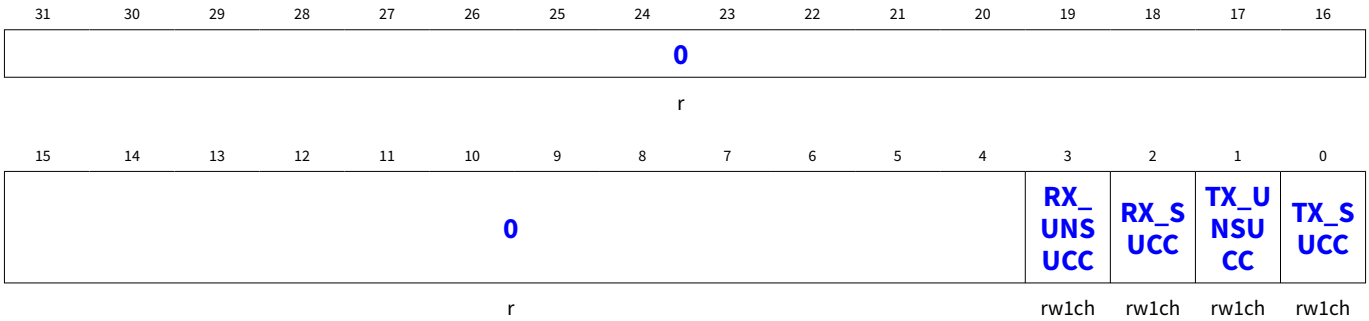


Field	Bits	Type	Description
UNVALID	31:0	rw1ch	**UNVALID** When UNVALID[n] = 1, an invalid RX descriptor is detected while running the TX Priority Queue slot n. Writing a 1 clears the bit. When set to 1, the TX Priority Queue slot n is on hold, waiting for the SW to react. As the TX message is fully defined in system memory before starting the relevant slot, there should not be any invalid TX descriptor interrupts

22.9.88 Node i statistics interrupt status register

Ni_STATS_INT_STS (i=0-3) Offset address: $21710_H+i*2000_H$

Node i statistics interrupt status register Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
TX_SUCC	0	rw1ch	**TX SUCC** Counter of TX message transmitted successfully has wrapped, writing a 1 clears the bit.

(table continues...)

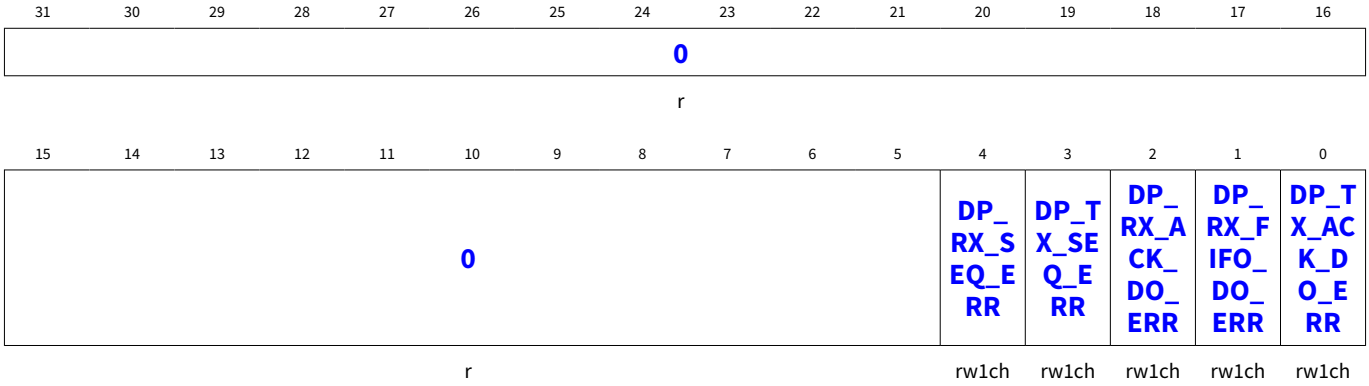
(continued)

Field	Bits	Type	Description
TX_UNSUCC	1	rw1ch	**TX UNSUCC** Counter of TX message transmitted unsuccessfully has wrapped, writing a 1 clears the bit.
RX_SUCC	2	rw1ch	**RX SUCC** Counter of RX message received successfully has wrapped, writing a 1 clears the bit.
RX_UNSUCC	3	rw1ch	**RX UNSUCC** Counter of RX message received unsuccessfully has wrapped, writing a 1 clears the bit.
0	31:4	r	**Reserved** Read as all 0's; should be written with all 0's.

22.9.89 Node i error interrupt status register

Ni_ERR_INT_STS (i=0-3) Offset address: $21714_H+i*2000_H$

Node i error interrupt status register Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
DP_TX_ACK_DO_ERR	0	rw1ch	**DP TX ACK DO ERR** When set to 1, a TX acknowledge data overflow is detected, writing a 1 clears the bit.
DP_RX_FIFO_DO_ERR	1	rw1ch	**DP RX FIFO DO ERR** When set to 1, an RX DMA FIFO overflow issue is detected, writing a 1 clears the bit.
DP_RX_ACK_DO_ERR	2	rw1ch	**DP RX ACK DO ERR** When set to 1, an RX acknowledge data overflow is detected, writing a 1 clears the bit.
DP_TX_SEQ_ERR	3	rw1ch	**DP TX SEQ ERR** When set to 1, a TX sequence issue is detected, writing a 1 clears the bit.
DP_RX_SEQ_ERR	4	rw1ch	**DP RX SEQ ERR** When set to 1, an RX sequence issue is detected, writing a 1 clears the bit.

(table continues...)

(continued)

Field	Bits	Type	Description
0	31:5	r	**Reserved** Read as all 0's; should be written with all 0's.

22.9.90 Node i safety interrupt status register

Ni_SFTY_INT_STS (i=0-3) Offset address: $21718_H+i*2000_H$

Node i safety interrupt status register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0														ACK_RX_PARITY_ERR	ACK_TX_PARITY_ERR
r														rw1ch	rw1ch

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MEM_SFTY_CE	MEM_SFTY_UE	RX_DESC_CRC_ERR	RX_DESC_REQ_ERR	TX_DESC_CRC_ERR	TX_DESC_REQ_ERR	AP_RX_PARITY_ERR	AP_TX_PARITY_ERR	DP_RX_PARITY_ERR	DP_TX_PARITY_ERR	MEM_AXI_RD_TO_ERR	MEM_AXI_WR_TO_ERR	DP_PRT_RX_TO_ERR	DP_PRT_TX_TO_ERR	DMA_AXI_RD_TO_ERR	DMA_AXI_WR_TO_ERR
rw1ch	rw1ch	rw1ch	rw1ch	rw1ch	rw1ch	rw1ch	rw1ch	rw1ch	rw1ch	rw1ch	rw1ch	rw1ch	rw1ch	rw1ch	rw1ch

Field	Bits	Type	Description
DMA_AXI_WR_TO_ERR	0	rw1ch	**DMA AXI WR TO ERR** When set to 1, an AXI write access timeout issue is detected on DMA interface, writing a 1 clears the bit.
DMA_AXI_RD_TO_ERR	1	rw1ch	**DMA AXI RD TO ERR** When set to 1, an AXI read access timeout issue is detected on DMA interface, writing a 1 clears the bit.
DP_PRT_TX_TO_ERR	2	rw1ch	**DP PRT TX TO ERR** When set to 1, a TX_MSG timeout issue is detected, writing a 1 clears the bit.
DP_PRT_RX_TO_ERR	3	rw1ch	**DP PRT RX TO ERR** When set to 1, an RX_MSG timeout issue is detected, writing a 1 clears the bit.
MEM_AXI_WR_TO_ERR	4	rw1ch	**MEM AXI WR TO ERR** When set to 1, an AXI write access timeout issue is detected on local memory interface, writing a 1 clears the bit.
MEM_AXI_RD_TO_ERR	5	rw1ch	**MEM AXI RD TO ERR** When set to 1, an AXI read access timeout issue is detected on local memory interface, writing a 1 clears the bit.

(table continues...)

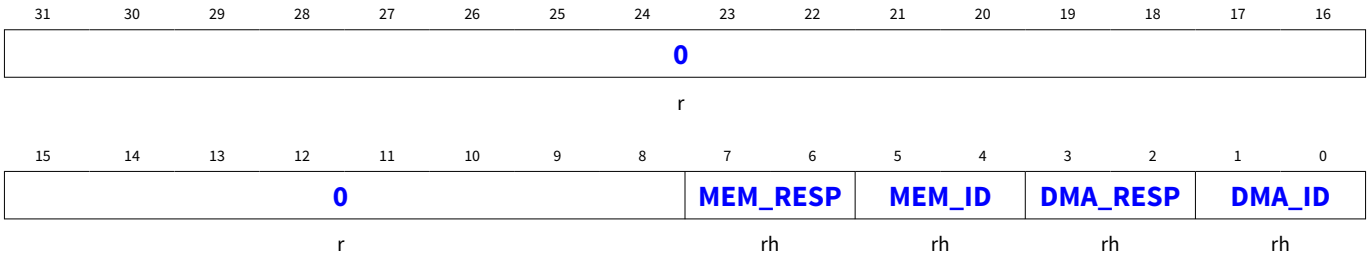
(continued)

Field	Bits	Type	Description
DP_TX_PARITY_ERR	6	rw1ch	**DP TX PARITY ERR** When set to 1, a TX data parity error is detected on datapath, writing a 1 clears the bit.
DP_RX_PARITY_ERR	7	rw1ch	**DP RX PARITY ERR** When set to 1, an RX data parity error is detected on datapath, writing a 1 clears the bit.
AP_TX_PARITY_ERR	8	rw1ch	**AP TX PARITY ERR** When set to 1, a TX address pointer parity issue is detected, writing a 1 clears the bit.
AP_RX_PARITY_ERR	9	rw1ch	**AP RX PARITY ERR** When set to 1, an RX address pointer parity issue is detected, writing a 1 clears the bit.
TX_DESC_REQ_ERR	10	rw1ch	**TX DESC REQ ERR** When set to 1, a TX descriptor fetched does not match the one expected, writing a 1 clears the bit.
TX_DESC_CRC_ERR	11	rw1ch	**TX DESC CRC ERR** When set to 1, a TX descriptor has a wrong CRC, writing a 1 clears the bit.
RX_DESC_REQ_ERR	12	rw1ch	**RX DESC REQ ERR** When set to 1, an RX descriptor fetched does not match the one expected, writing a 1 clears the bit.
RX_DESC_CRC_ERR	13	rw1ch	**RX DESC CRC ERR** When set to 1, an RX descriptor has a wrong CRC, writing a 1 clears the bit.
MEM_SFTY_UE	14	rw1ch	**MEM SFTY UE** When set to 1, an uncorrectable error is detected on the local memory interface, writing a 1 clears the bit.
MEM_SFTY_CE	15	rw1ch	**MEM SFTY CE** When set to 1, a correctable error is detected on the local memory interface, writing a 1 clears the bit.
ACK_TX_PARITY_ERR	16	rw1ch	**ACK TX PARITY ERR** When set to 1, an acknowledge data parity issue is detected on the TX path, writing a 1 clears the bit.
ACK_RX_PARITY_ERR	17	rw1ch	**ACK RX PARITY ERR** When set to 1, an acknowledge data parity issue is detected on the RX path, writing a 1 clears the bit.
0	31:18	r	**Reserved** Read as all 0's; should be written with all 0's.

22.9.91 Node i DMA error information

Ni_AXI_ERR_INFO (i=0-3) Offset address: $2171C_H+i*2000_H$

Node i DMA error information Kernel Reset value: $0000\ 0000_H$



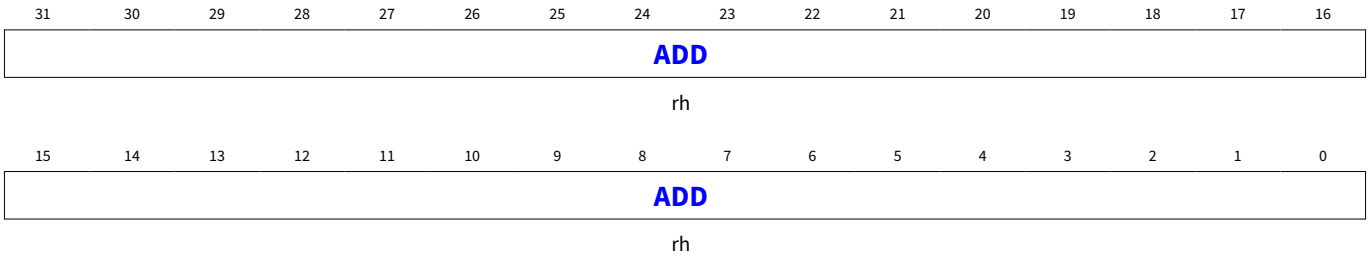
Field	Bits	Type	Description
DMA_ID	1:0	rh	**DMA ID** On DMA_AXI interface. Define the AXI ID used when a write or read error response is detected. According to the value, the DMA channel can be identified and so it is possible to define what's the effect of such issue.
DMA_RESP	3:2	rh	**DMA RESP** On DMA_AXI interface. When set to 0b10, the AXI response is SLVERR. When set to 0b11, the response is DECERR. By default, set to 0b00 (OKAY)
MEM_ID	5:4	rh	**MEM ID** On MEM_AXI interface. Define the AXI ID used when a write or read error response is detected. According to the value, the DMA channel can be identified and so it is possible to define what's the effect of such issue.
MEM_RESP	7:6	rh	**MEM RESP** On MEM_AXI interface. When set to 0b10, the AXI response is SLVERR. When set to 0b11, the response is DECERR. By default, set to 0b00 (OKAY)
0	31:8	r	**Reserved** Read as all 0's; should be written with all 0's.

22.9.92 Node i descriptor error information 0

If the Ni_DESC_ERR_INFO0.ADD[31:16] = 0 and Ni_DESC_ERR_INFO1.CRC[8:0], Ni_DESC_ERR_INFO1.RX_TX and Ni_DESC_ERR_INFO1.RC[4:0] are all equal to 0, the faulty descriptor is a TX descriptor fetched from L_MEM.

Ni_DESC_ERR_INFO0 (i=0-3) Offset address: $21720_H+i*2000_H$

Node i descriptor error information 0 Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
ADD	31:0	rh	**ADD** Descriptor address being used when the error is detected.

22.9.93 Node i descriptor error information 1

When the Ni_DESC_ERR_INFO1.CRC[8:0], Ni_DESC_ERR_INFO1.RX_TX and Ni_DESC_ERR_INFO1.RC[4:0] are all equal to 0, the faulty descriptor is a TX descriptor fetched from L_MEM only if the Ni_DESC_ERR_INFO0.ADD[31:16] = 0.

Ni_DESC_ERR_INFO1 (i=0-3) Offset address: $21724_H+i*2000_H$

Node i descriptor error information 1 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0							CRC								
r							rh								

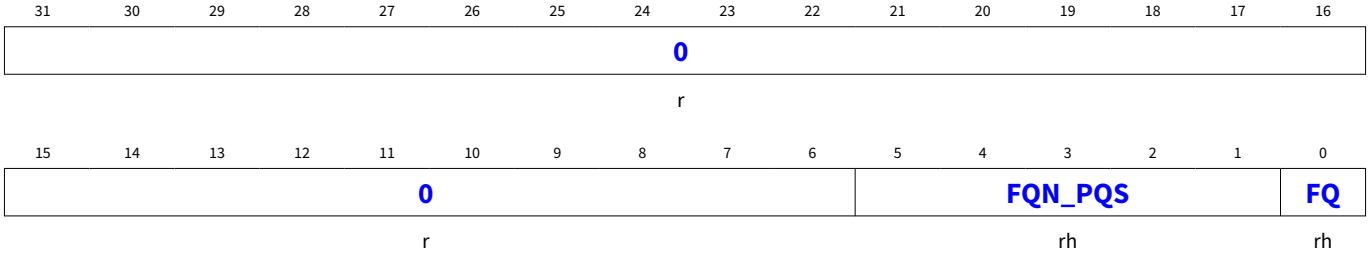
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RX_TX	0	RC					PQ	IN			FQN_PQSN				
rh	r	rh					rh	rh			rh				

Field	Bits	Type	Description
FQN_PQSN	4:0	rh	**FQN PQSN** Provide the information regarding the RX/TX FIFO Queue number or the TX Priority Queue slot having an issue.
IN	7:5	rh	**IN** Provide the instance number defined in RX or TX descriptor logged in.
PQ	8	rh	**PQ** Identify which TX queue is impacted, either the TX Priority Queue (PQ set to 1) or the TX FIFO Queues.
RC	13:9	rh	**RC** Provide the information regarding the Rolling Counter defined in RX or TX descriptor impacted.
RX_TX	15	rh	**RX TX** RX descriptor has an issue (RX_TX set to 1), otherwise the same for a TX descriptor.
CRC	24:16	rh	**CRC** CRC value defined in the RX or TX descriptor logged in.
0	14, 31:25	r	**Reserved** Read as 0; should be written with 0.

22.9.94 Node i TX filter error information

Ni_TX_FILTER_ERR_INFO (i=0-3) Offset address: $21728_H+i*2000_H$

Node i TX filter error information Kernel Reset value: $0000\ 0000_H$



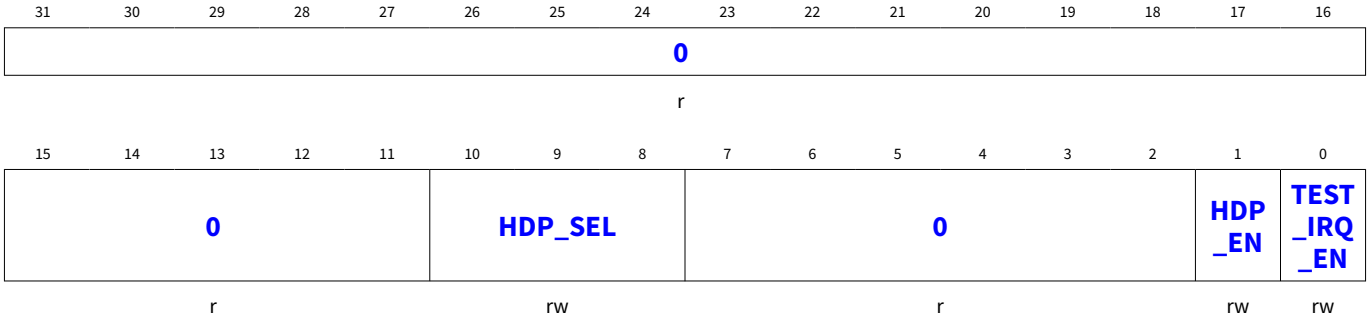
Field	Bits	Type	Description
FQ	0	rh	**FQ** When set to 1, one of the TX FIFO Queues has triggered the TX_FILTER_ERR interrupt, otherwise it is a TX Priority Queue slot.
FQN_PQS	5:1	rh	**FQN PQS** Provide the information of the TX FIFO Queue number or TX Priority Queue slot number which has triggered the TX_FILTER_ERR interrupt.
0	31:6	r	**Reserved** Read as all 0's; should be written with all 0's.

22.9.95 Node i debug control register

This register is only accessible in write mode if the Test Mode Key sequence has been performed prior to write. The register is accessible in write access in privileged mode only. This register is protected by a register bank CRC defined in Ni_CRC_REG register.

Ni_DEBUG_TEST_CTRL (i=0-3) Offset address: $21800_H+i*2000_H$

Node i debug control register Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
TEST_IRQ_EN	0	rw	**TEST IRQ EN** When writing 1, enable the control of the interrupt lines using the Ni_INT_TEST0 and Ni_INT_TEST1 registers. This bit field register is only accessible in write mode if the Test Mode Key sequence has been performed prior to write.
HDP_EN	1	rw	**HDP EN** When writing 1, enable the hardware debug port to monitor MH internal signals. This bit field register is only accessible in write mode if the Test Mode Key sequence has been performed prior to write.

(table continues...)

(continued)

Field	Bits	Type	Description
HDP_SEL	10:8	rw	**HDP SEL** Define the set of signals to be monitored on the HDP[15:0] bus signal interface. This bit field register is only accessible in write mode if the Test Mode Key sequence has been performed prior to write.
0	7:2, 31:11	r	**Reserved** Read as all 0's; should be written with all 0's.

22.9.96 Node i interrupt test register 0

This register is only accessible in write mode if the TEST_IRQ_EN bit in Ni_DEBUG_TEST_CTRL is set.

Ni_INT_TEST0 (i=0-3) Offset address: $21804_H+i*2000_H$

Node i interrupt test register 0 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								RX_FQ_IRQ							
r								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								TX_FQ_IRQ							
r								rw							

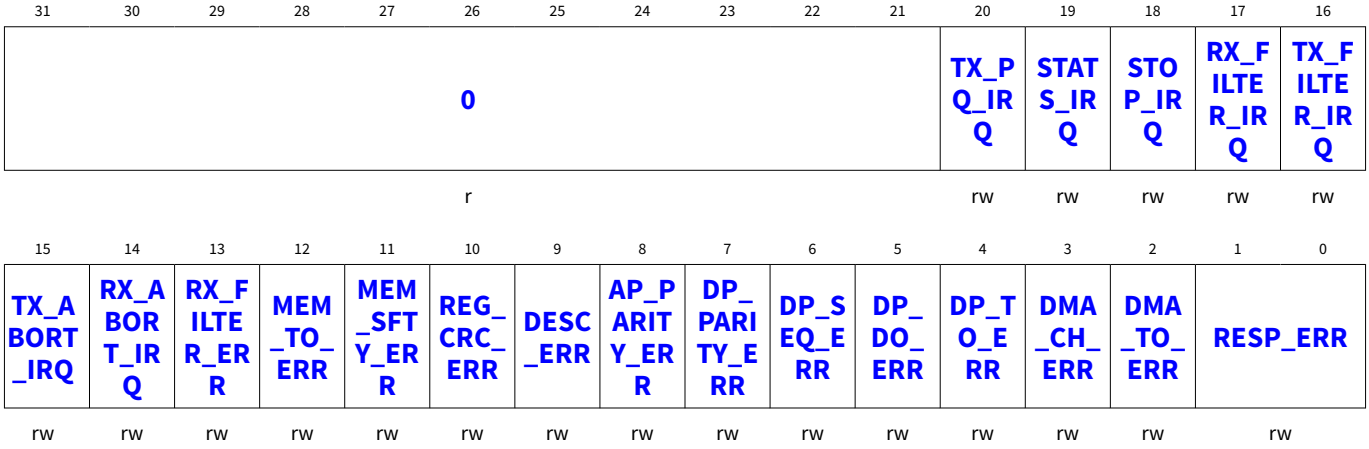
Field	Bits	Type	Description
TX_FQ_IRQ	7:0	rw	**TX FQ IRQ** When writing 1 to TX_FQ_IRQ[n], triggers the interrupt line tx_fq_irq[n], those bits are auto-cleared. This bit field register is only accessible in write mode if the TEST_IRQ_EN bit in Ni_DEBUG_TEST_CTRL is set
RX_FQ_IRQ	23:16	rw	**RX FQ IRQ** When writing 1 to RX_FQ_IRQ[n], triggers the interrupt line rx_fq_irq[n], those bits are auto-cleared. This bit field register is only accessible in write mode if the TEST_IRQ_EN bit in Ni_DEBUG_TEST_CTRL is set
0	15:8, 31:24	r	**Reserved** Read as all 0's; should be written with all 0's.

22.9.97 Node i interrupt test register 1

This register is only accessible in write mode if the TEST_IRQ_EN bit in Ni_DEBUG_TEST_CTRL is set.

Ni_INT_TEST1 (i=0-3) Offset address: $21808_H+i*2000_H$

Node i interrupt test register 1 Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
RESP_ERR	1:0	rw	**RESP ERR** Writing a 1 to the bit field triggers the related interrupt line, this bit is auto-cleared. This bit field register is only accessible in write mode if the TEST_IRQ_EN bit in Ni_DEBUG_TEST_CTRL is set
DMA_TO_ERR	2	rw	**DMA TO ERR** Writing a 1 to the bit field triggers the related interrupt line, this bit is auto-cleared. This bit field register is only accessible in write mode if the TEST_IRQ_EN bit in Ni_DEBUG_TEST_CTRL is set
DMA_CH_ERR	3	rw	**DMA CH ERR** Writing a 1 to the bit field triggers the related interrupt line, this bit is auto-cleared. This bit field register is only accessible in write mode if the TEST_IRQ_EN bit in Ni_DEBUG_TEST_CTRL is set
DP_TO_ERR	4	rw	**DP TO ERR** Writing a 1 to the bit field triggers the related interrupt line, this bit is auto-cleared. This bit field register is only accessible in write mode if the TEST_IRQ_EN bit in Ni_DEBUG_TEST_CTRL is set
DP_DO_ERR	5	rw	**DP DO ERR** Writing a 1 to the bit field triggers the related interrupt line, this bit is auto-cleared. This bit field register is only accessible in write mode if the TEST_IRQ_EN bit in Ni_DEBUG_TEST_CTRL is set
DP_SEQ_ERR	6	rw	**DP SEQ ERR** Writing a 1 to the bit field triggers the related interrupt line, this bit is auto-cleared. This bit field register is only accessible in write mode if the TEST_IRQ_EN bit in Ni_DEBUG_TEST_CTRL is set
DP_PARITY_ERR	7	rw	**DP PARITY ERR** Writing a 1 to the bit field triggers the related interrupt line, this bit is auto-cleared. This bit field register is only accessible in write mode if the TEST_IRQ_EN bit in Ni_DEBUG_TEST_CTRL is set
AP_PARITY_ERR	8	rw	**AP PARITY ERR** Writing a 1 to the bit field triggers the related interrupt line, this bit is auto-cleared. This bit field register is only accessible in write mode if the TEST_IRQ_EN bit in Ni_DEBUG_TEST_CTRL is set

(table continues...)

(continued)

Field	Bits	Type	Description
DESC_ERR	9	rw	**DESC ERR** Writing a 1 to the bit field triggers the related interrupt line, this bit is auto-cleared. This bit field register is only accessible in write mode if the TEST_IRQ_EN bit in Ni_DEBUG_TEST_CTRL is set
REG_CRC_ERR	10	rw	**REG CRC ERR** Writing a 1 to the bit field triggers the related interrupt line, this bit is auto-cleared. This bit field register is only accessible in write mode if the TEST_IRQ_EN bit in Ni_DEBUG_TEST_CTRL is set
MEM_SFTY_ERR	11	rw	**MEM SFTY ERR** Writing a 1 to the bit field triggers the related interrupt line, this bit is auto-cleared. This bit field register is only accessible in write mode if the TEST_IRQ_EN bit in Ni_DEBUG_TEST_CTRL is set
MEM_TO_ERR	12	rw	**MEM TO ERR** Writing a 1 to the bit field triggers the related interrupt line, this bit is auto-cleared. This bit field register is only accessible in write mode if the TEST_IRQ_EN bit in Ni_DEBUG_TEST_CTRL is set
RX_FILTER_ERR	13	rw	**RX FILTER ERR** Writing a 1 to the bit field triggers the related interrupt line, this bit is auto-cleared. This bit field register is only accessible in write mode if the TEST_IRQ_EN bit in Ni_DEBUG_TEST_CTRL is set
RX_ABORT_IRQ	14	rw	**RX ABORT IRQ** Writing a 1 to the bit field triggers the related interrupt line, this bit is auto-cleared. This bit field register is only accessible in write mode if the TEST_IRQ_EN bit in Ni_DEBUG_TEST_CTRL is set
TX_ABORT_IRQ	15	rw	**TX ABORT IRQ** Writing a 1 to the bit field triggers the related interrupt line, this bit is auto-cleared. This bit field register is only accessible in write mode if the TEST_IRQ_EN bit in Ni_DEBUG_TEST_CTRL is set
TX_FILTER_IRQ	16	rw	**TX FILTER IRQ** Writing a 1 to the bit field triggers the related interrupt line, this bit is auto-cleared. This bit field register is only accessible in write mode if the TEST_IRQ_EN bit in Ni_DEBUG_TEST_CTRL is set
RX_FILTER_IRQ	17	rw	**RX FILTER IRQ** Writing a 1 to the bit field triggers the related interrupt line, this bit is auto-cleared. This bit field register is only accessible in write mode if the TEST_IRQ_EN bit in Ni_DEBUG_TEST_CTRL is set
STOP_IRQ	18	rw	**STOP IRQ** Writing a 1 to the bit field triggers the related interrupt line, this bit is auto-cleared. This bit field register is only accessible in write mode if the TEST_IRQ_EN bit in Ni_DEBUG_TEST_CTRL is set
STATS_IRQ	19	rw	**STATS IRQ** Writing a 1 to the bit field triggers the related interrupt line, this bit is auto-cleared. This bit field register is only accessible in write mode if the TEST_IRQ_EN bit in Ni_DEBUG_TEST_CTRL is set

(table continues...)

(continued)

Field	Bits	Type	Description
TX_PQ_IRQ	20	rw	**TX PQ IRQ** Writing a 1 to the bit field triggers the related interrupt line, this bit is auto-cleared. This bit field register is only accessible in write mode if the TEST_IRQ_EN bit in Ni_DEBUG_TEST_CTRL is set
0	31:21	r	**Reserved** Read as all 0's; should be written with all 0's.

22.9.98 Node i TX-SCAN first candidates register

This register gives the 4 best candidates evaluated by the TX-Scan. This register gives the first and second highest priority TX descriptor after a TX-Scan. It provides also the third and fourth candidates during a TX-Scan, considering the first and second candidates as already defined by a previous TX-Scan run.

Ni_TX_SCAN_FC (i=0-3) Offset address: $21810_H+i*2000_H$

Node i TX-SCAN first candidates register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		FQN_PQSN3					FQ_PQ3	0		FQN_PQSN2					FQ_PQ2
r		rh					rh	r		rh					rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		FQN_PQSN1					FQ_PQ1	0		FQN_PQSN0					FQ_PQ0
r		rh					rh	r		rh					rh

Field	Bits	Type	Description
FQ_PQ0	0	rh	**FQ PQ0** The first candidate evaluated by TX-Scan is either a TX Priority Queue (when set to 1) or a TX FIFO Queue (when set to 0). This bit field is identical to Ni_TX_SCAN_BC.FH_PQ bit register
FQN_PQSN0	5:1	rh	**FQN PQSN0** The first candidate is coming from either the TX FIFO Queue number N (defined by FQN in TX descriptor) or the TX Priority Queue Slot number M (defined by the PQSN in TX descriptor). The meaning of this bit field depends on the PQ0. This bit field is identical to Ni_TX_SCAN_BC.FH_FQN_PQSN bit register
FQ_PQ1	8	rh	**FQ PQ1** The second candidate evaluated by TX-Scan is either a TX Priority Queue (when set to 1) or a TX FIFO Queue (when set to 0). This bit field is identical to Ni_TX_SCAN_BC.SH_PQ bit register
FQN_PQSN1	13:9	rh	**FQN PQSN1** The second candidate is coming from either the TX FIFO Queue number N (defined by FQN in TX descriptor) or the TX Priority Queue Slot number M (defined by the PQSN in TX descriptor). The meaning of this bit field depends on the PQ0. This bit field is identical to the Ni_TX_SCAN_BC.SH_FQN_PQSN bit register

(table continues...)

(continued)

Field	Bits	Type	Description
FQ_PQ2	16	rh	**FQ PQ2** The third candidate evaluated by TX-Scan is either a TX Priority Queue (when set to 1) or a TX FIFO Queue (when set to 0).
FQN_PQSN2	21:17	rh	**FQN PQSN2** The third candidate is coming from either the TX FIFO Queue number N (defined by FQN in TX descriptor) or the TX Priority Queue Slot number M (defined by the PQSN in TX descriptor). The meaning of this bit field depends on the PQ2.
FQ_PQ3	24	rh	**FQ PQ3** The fourth candidate evaluated by TX-Scan is either a TX Priority Queue (when set to 1) or a TX FIFO Queue (when set to 0).
FQN_PQSN3	29:25	rh	**FQN PQSN3** The fourth candidate is coming from either the TX FIFO Queue number N (defined by FQN in TX descriptor) or the TX Priority Queue Slot number M (defined by the PQSN in TX descriptor). The meaning of this bit field depends on the PQ3.
0	7:6, 15:14, 23:22, 31:30	r	**Reserved** Read as all 0's; should be written with all 0's.

22.9.99 Node i TX-SCAN best candidates register

This register gives the first and second highest priority TX descriptor after a TX-Scan.

Ni_TX_SCAN_BC (i=0-3) Offset address: $21814_H+i*2000_H$

Node i TX-SCAN best candidates register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SH_OFFSET										SH_FQN_PQSN					SH_PQ
rh										rh					rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FH_OFFSET										FH_FQN_PQSN					FH_PQ
rh										rh					rh

Field	Bits	Type	Description
FH_PQ	0	rh	**FH PQ** First highest priority candidate evaluated by TX-Scan. It is either a TX Priority Queue (when set to 1) or a TX FIFO Queue (when set to 0).

(table continues...)

(continued)

Field	Bits	Type	Description
FH_FQN_PQSN	5:1	rh	**FH FQN PQSN** First highest priority candidate coming from either the TX FIFO Queue number N (defined by FQN in TX descriptor) or the TX Priority Queue Slot number M (defined by the PQSN in TX descriptor). The meaning of this bit field depends on the FH_PQ.
FH_OFFSET	15:6	rh	**FH OFFSET** First highest priority candidate offset in multiple of 32bytes (TX descriptor size). This register is relevant only for the TX FIFO Queue. It provides the index of the TX descriptor in the TX FIFO Queue which is in use on the CAN bus. When FH_PQ = 1 it is set to 0.
SH_PQ	16	rh	**SH PQ** Second highest priority candidate evaluated by TX-Scan. It is either a TX Priority Queue (when set to 1) or a TX FIFO Queue (when set to 0).
SH_FQN_PQSN	21:17	rh	**SH FQN PQSN** Second highest priority candidate coming from either the TX FIFO Queue number N (defined by FQN in TX descriptor) or the TX Priority Queue Slot number M (defined by the PQSN in TX descriptor). The meaning of this bit field depends on the SH_PQ.
SH_OFFSET	31:22	rh	**SH OFFSET** Second highest priority candidate offset in multiple of 32bytes (TX descriptor size). This register is relevant only for the TX FIFO Queue. It provides the index of the TX descriptor in the TX FIFO Queue which is about to be sent on the CAN bus. When SH_PQ = 1 it is set to 0.

22.9.100 Node i valid TX FIFO queue descriptors in local memory

Ni_TX_FQ_DESC_VALID (i=0-3) Offset address: $21818_H+i*2000_H$

Node i valid TX FIFO queue descriptors in local memory Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								DESC_NC_VALID							
r								rh							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								DESC_CN_VALID							
r								rh							

Field	Bits	Type	Description
DESC_CN_VALID	7:0	rh	**DESC CN VALID** When DESC_CN_VALID[n] = 1, the current/next TX descriptor for the TX FIFO Queue n is available in L_MEM.
DESC_NC_VALID	23:16	rh	**DESC NC VALID** When DESC_NC_VALID[n] = 1, the next/current TX descriptor for the TX FIFO Queue is available in L_MEM.

(table continues...)

(continued)

Field	Bits	Type	Description
0	15:8, 31:24	r	**Reserved** Read as all 0's; should be written with all 0's.

22.9.101 Node i valid TX priority queue descriptors in local memory

Ni_TX_PQ_DESC_VALID (i=0-3)

Offset address: $2181C_H+i*2000_H$

Node i valid TX priority queue descriptors in local memory

Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DESC_VALID															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DESC_VALID															
rh															

Field	Bits	Type	Description
DESC_VALID	31:0	rh	**DESC VALID** When DESC_VALID[n] = 1, the TX descriptor assigned to the slot n in local memory is valid.

22.9.102 Node i CRC control register

Ni_CRC_CTRL (i=0-3)

Offset address: $21880_H+i*2000_H$

Node i CRC control register

Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															START
r															w

Field	Bits	Type	Description
START	0	w	**START** Writing a 1 to this bit triggers the HW CRC check of registers. This action can be done any time for a sanity check
0	31:1	r	**Reserved** Read as all 0's; should be written with all 0's.

22.9.103 Node i CRC register

Ni_CRC_REG (i=0-3) Offset address: $21884_H+i*2000_H$

Node i CRC register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
VAL															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VAL															
rw															

Field	Bits	Type	Description
VAL	31:0	rw	**VAL** CRC value of all the registers protected by CRC. Once done, a write to the START bit in the Ni_CRC_CTRL register must be done

22.9.104 Node i endianness test register

Ni_ENDN (i=0-3) Offset address: $21900_H+i*2000_H$

Node i endianness test register Kernel Reset value: $8765\ 4321_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ETV															
r															

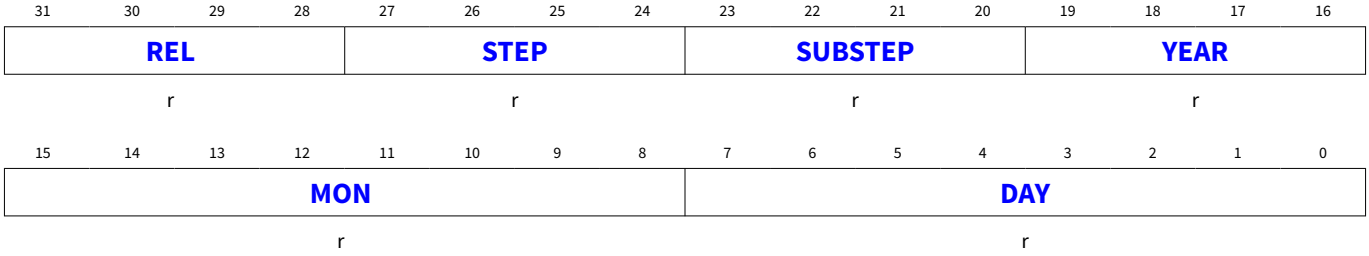
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ETV															
r															

Field	Bits	Type	Description
ETV	31:0	r	**ETV** The purpose of this register is to identify the beginning of the PRT address map in a memory dump and to check the proper endianness data byte mapping when the data word is routed through different busses.

22.9.105 Node i PRT release identification register

Ni_PREL (i=0-3) Offset address: $21904_H+i*2000_H$

Node i PRT release identification register Kernel Reset value: $0540\ 0000_H$

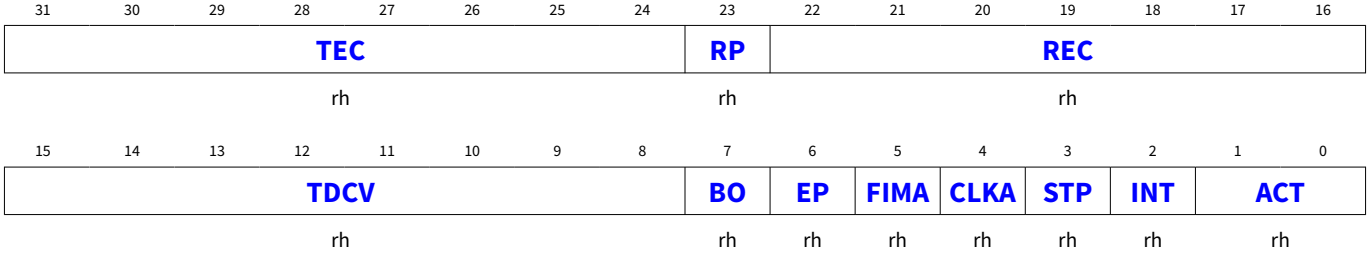


Field	Bits	Type	Description
DAY	7:0	r	**DAY** Define the day of the release using a binary coded decimal representation (1 being the first day of the month and so forth). This reset value is defined by the generic parameter DESIGN_TIME_STAMP_G[7:0]. If the generic parameter DESIGN_TIME_STAMP_G is not set, the default value is the one defined here
MON	15:8	r	**MON** Define the month of the release using a binary coded decimal representation (1 being January and so forth). This reset value is defined by the generic parameter DESIGN_TIME_STAMP_G[15:8]. If the generic parameter DESIGN_TIME_STAMP_G is not set, the default value is the one defined here
YEAR	19:16	r	**YEAR** Define the year of the release using a binary coded decimal representation (0 being 2020 and so forth…). This reset value is defined by the generic parameter DESIGN_TIME_STAMP_G[19:16]. If the generic parameter DESIGN_TIME_STAMP_G is not set, the default value is the one defined here
SUBSTEP	23:20	r	**SUBSTEP** Sub-Step value according to Step.
STEP	27:24	r	**STEP** Step value according to Release.
REL	31:28	r	**REL** Release value, used to identify the main release of the XCAN_PRT.

22.9.106 Node i PRT status register

Ni_STAT (i=0-3) Offset address: $21908_H+i*2000_H$

Node i PRT status register Kernel Reset value: $0000\ 0010_H$



Field	Bits	Type	Description
ACT	1:0	rh	**ACT** The current activity of this node: 0b00: inactive state 0b01: Idle 0b10: Receiver 0b11: Transmitter When the CAN protocol operation is stopped, ACT changes to 0b00 and INT changes to 0. When the CAN protocol operation is started, INT is set to 1, but ACT remains at 0b00 until the CAN protocol’s bus idle detection condition is met, then it changes to 0b01 and INT changes to 0. When the CAN protocol operation is started while BO is set, the PRT remains in integrating state (INT=1 and ACT=0b00) until the Bus-Off recovery sequence is finished, then it changes to 0b01 and INT changes to 0. When PRT detects a protocol exception event (see [1], chapter 10.9.5), ACT changes to 0b00 and INT changes to 1 until the CAN protocol’s bus idle detection condition is met, then ACT changes to 0b01 and INT changes to 0. ACT changes from 0b01 to 0b10 when the PRT has received a Start-of-Frame from the CAN bus. ACT changes from 0b01 to 0b11 when the PRT has sent a Start-of-Frame to the CAN bus. ACT changes from 0b11 to 0b10 when the PRT loses arbitration during a transmission. ACT changes from 0b10 to 0b01 or from 0b11 to 0b01when the PRT detects the second bit of intermission (see [1], chapter 10.4.6.2) to be recessive.
INT	2	rh	**INT** This node is integrating into CAN bus traffic.
STP	3	rh	**STP** Waiting for end of actual message after STOP command, see Starting and Stopping The Module chapter.
CLKA	4	rh	**CLKA** The actual value of the CLOCK_ACTIVE input signal, see Starting and Stopping the Module chapter. As the clock must be active when a reset is performed, the default value should be 1.
FIMA	5	rh	**FIMA** Fault Injection Module Activated, see Safety Measures chapter.
EP	6	rh	**EP** This node is in Error-Passive state. When both error counters drop below 127, or when the Bus-Off recovery sequence is finished, the EP bit is cleared.
BO	7	rh	**BO** This node is in Bus-Off state. This flag is set on an error condition that would have caused an increment of the Transmit Error Counter to a value beyond its 8 bit range. When the PRT enters Bus-Off state, BO is set to 1 and CAN protocol operation is stopped. When the Bus-Off recovery sequence is finished, BO is cleared.
TDCV	15:8	rh	**TDCV** Transmitter Delay Compensation’s delay value. A software reset clears the TDV bit field to 0x00. This register shows the sum of the measured delay plus the configured offset, giving the position of the secondary sample point. It is updated for each frame transmission that includes a data phase.

(table continues...)

(continued)

Field	Bits	Type	Description
REC	22:16	rh	**REC** The CAN protocol’s Receive Error Counter. A software reset does not change the value in this register. When the Bus-Off recovery sequence is finished, the error counter REC is cleared. The REC is a 7-bit-counter, together with the Error-Passive flag EP. When the increment REC+1 or REC+8 would result in a value > 127 (carry-flag), the REC is kept unchanged, but EP is set. When EP is set but REC is below 127 and further errors are detected with an REC+1 condition, the REC will be incremented until it reaches 127. At the reception of a valid message, the REC-1 decrements the actual value of the REC by one AND clears the Error-Passive flag EP.
RP	23	rh	**RP** The Passive flag of the CAN protocol’s Receive Error Counter. This flag is set on an error condition that would have caused an increment of the Receive Error Counter to a value beyond its 7 bit range.
TEC	31:24	rh	**TEC** The CAN protocol’s Transmit Error Counter. A software reset does not change the value in this register. When the Bus-Off recovery sequence is finished, the error counter TEC is cleared. When the increment TEC+8 would result in a value > 255 (carry-flag), the TEC is kept unchanged, but BO is set. The Transmit Error Counter is decremented by one each time a CAN message has been successfully transmitted, but it is not decremented below the value 0.

22.9.107 Node i event status flags register

The Ni_EVNT Register contains event status flags. The flags are set by the PRT when specific events occur. A software reset clears all flags. A host write access to this register, writing a 1 to a specific flag, clears that flag. When a host write access occurs concurrently with a set condition for a flag, the flag is set.

Ni_EVNT (i=0-3) Offset address: $21920_H+i*2000_H$

Node i event status flags register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		ABO	IFR	USO	DU	PXE	TXF	RXF	DO	STE	FRE	AKE	B1E	B0E	CRE
r		rw1ch	rw1ch	rw1ch	rw1ch	rw1ch	rw1ch	rw1ch	rw1ch	rw1ch	rw1ch	rw1ch	rw1ch	rw1ch	rw1ch

Field	Bits	Type	Description
CRE	0	rw1ch	**CRE** CRC Error.

(table continues...)

(continued)

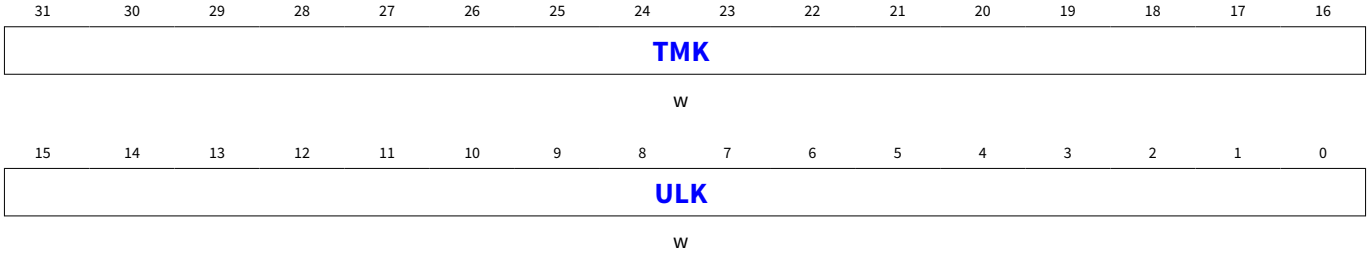
Field	Bits	Type	Description
B0E	1	rw1ch	**B0E** Bit0 Error: The PRT wanted to send a dominant bit (logical value 0), but the monitored CAN bus value was recessive. During Bus-Off recovery, B0E is also set each time a sequence of 11 recessive bits has been monitored, enabling the CPU to readily check whether the CAN bus is stuck at dominant or continuously disturbed, and to monitor the proceeding of the Bus-Off recovery sequence.
B1E	2	rw1ch	**B1E** Bit1Error: During the transmission of a message (with the exception of the arbitration field), the PRT wanted to send a recessive bit (logical value 1), but the monitored CAN bus value was dominant.
AKE	3	rw1ch	**AKE** Acknowledge Error.
FRE	4	rw1ch	**FRE** Form Error or the condition of CAN error counting rule f).
STE	5	rw1ch	**STE** Stuff Error.
DO	6	rw1ch	**DO** Overflow condition in RX_MSG sequence detected.
RXF	7	rw1ch	**RXF** Frame received.
TXF	8	rw1ch	**TXF** Frame transmitted.
PXE	9	rw1ch	**PXE** Protocol Exception Event occurred.
DU	10	rw1ch	**DU** Underrun condition in TX_MSG sequence detected.
USO	11	rw1ch	**USO** Unexpected Start of Sequence during TX_MSG sequence detected.
IFR	12	rw1ch	**IFR** Invalid Frame Format requested in TX_MSG.
ABO	13	rw1ch	**ABO** TX_MSG sequence stopped by TX_MSG_WUSER code ABORT.
0	31:14	r	**Reserved** Read as all 0's; should be written with all 0's.

22.9.108 Node i unlock sequence register

Writing a sequence of specific data words enables the activation of control commands in the registers Ni_CTRL and Ni_FIMC.

Ni_LOCK (i=0-3) Offset address: $21940_H+i*2000_H$

Node i unlock sequence register Kernel Reset value: $0000\ 0000_H$



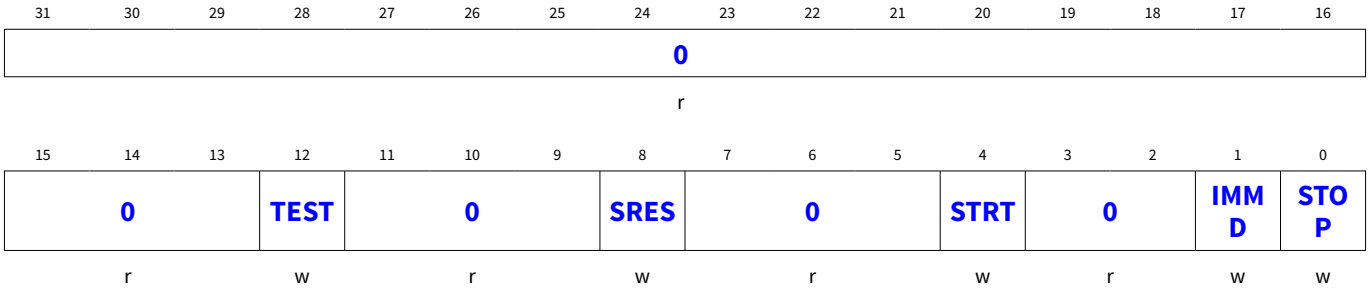
Field	Bits	Type	Description
ULK	15:0	w	**ULK** Unlock Key.
TMK	31:16	w	**TMK** Test Mode Key.

22.9.109 Node i control register

Writing to this register controls the CAN protocol operation. Reading this register gives the value 0x00000000. When writing to this register, only one of the four bits TEST, SRES , STRT, or STOP may be written to 1, otherwise the write access takes no effect. The bit IMMD may be written to 1 together with the bit STOP, but not together with one of the other bits.

Ni_CTRL (i=0-3) Offset address: $21944_H+i*2000_H$

Node i control register Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
STOP	0	w	**STOP** Stop CAN protocol operation. The Unlock Key must be used prior to write to this bit field. When not set together with bit IMMD the PRT waits for an ongoing CAN message to finish before stopping CAN protocol operation.
IMMD	1	w	**IMMD** Stop CAN protocol operation immediately. The Unlock Key must be used prior to write to this bit . This bit is only effective when being set together with the bit STOP.
STRT	4	w	**STRT** Start CAN protocol operation.

(table continues...)

(continued)

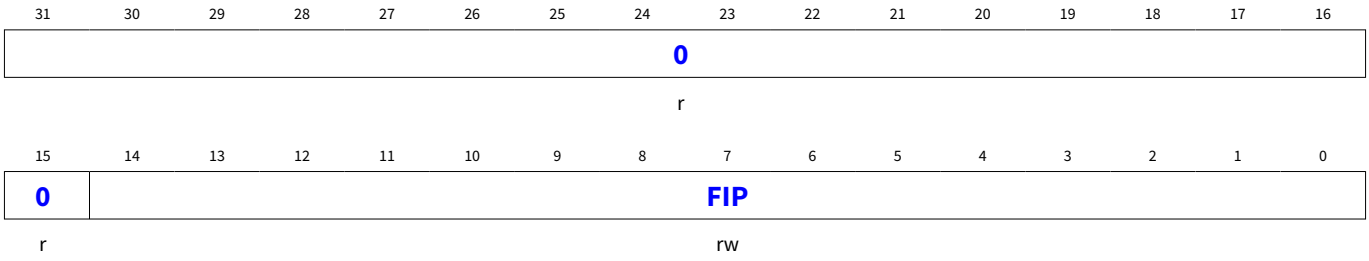
Field	Bits	Type	Description
SRES	8	w	**SRES** Software Reset. When the CAN protocol operation is stopped, the software reset of all state machines of the PRT (excluding the error-counters and the error-states) is triggered by writing 1 to Ni_CTRL.SRES. No unlocking sequence is required. A software reset will not be executed while the CAN protocol operation is started.
TEST	12	w	**TEST** Enable Test Mode. The Test Mode Key must be used prior to write to this bit field.
0	3:2, 7:5, 11:9, 31:13	r	**Reserved** Read as all 0's; should be written with all 0's.

22.9.110 Node i fault injection module control register

Writing the fault injection position number requires the application of the test mode key sequence before writing to Ni_FIMC. This register must be accessed in privileged mode when supported.

Ni_FIMC (i=0-3) Offset address: $21948_H+i*2000_H$

Node i fault injection module control register Kernel Reset value: $0000\ 0000_H$



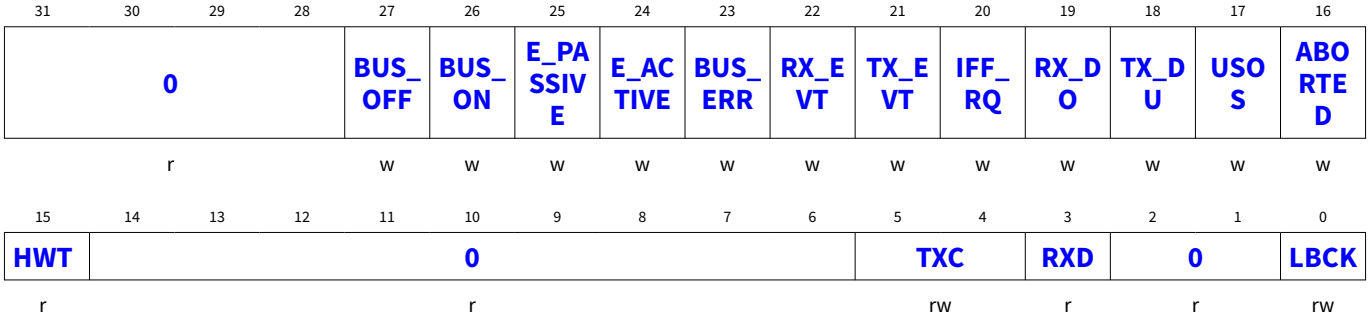
Field	Bits	Type	Description
FIP	14:0	rw	**FIP** Fault Injection Position. Writing to Ni_FIMC while Ni_MODE.FIME is set activates the Fault Injection Module FIM (see Safety Measures chapter). While the FIM is activated, the value of Ni_FIMC.FIP is protected from further write accesses until the FIM is de-activated again.
0	31:15	r	**Reserved** Read as all 0's; should be written with all 0's.

22.9.111 Node i hardware test functions register

This register is writable after the hardware test mode functions are enabled by writing the test mode key sequence to Ni_LOCK and Ni_CTRL registers. While the hardware test mode functions are not enabled, this register is read-only. This register must be accessed in privileged mode when supported. The hardware test mode functions are disabled and cleared by the software reset of the PRT.

Ni_TEST (i=0-3) Offset address: $2194C_H+i*2000_H$

Node i hardware test functions register Kernel Reset value: $0000\ 0008_H$



Field	Bits	Type	Description
LBCK	0	rw	**LBCK** Enable the Message Loop-Back mode, see chapter Trace and Debug.
RXD	3	r	**RXD** Bit value seen at CAN_RX. The CAN_RX input (output signal of the transceiver) is always readable through this bit.
TXC	5:4	rw	**TXC** Control the bit value driven at CAN_TX 0b00: Normal function of CAN TX 0b01: Normal function of CAN TX. CAN RX is ignored (for message look back mode) 0b10: CAN TX output set to 0 0b11: CAN TX output set to 1
HWT	15	r	**HWT** This status flag HWT shows whether the hardware test mode functions are enabled, set to 1 means enable.
ABORTED	16	w	**ABORTED** Writing a 1 to the bit field triggers the related interrupt line, this bit is auto-cleared.
USOS	17	w	**USOS** Writing a 1 to the bit field triggers the related interrupt line, this bit is auto-cleared.
TX_DU	18	w	**TX DU** Writing a 1 to the bit field triggers the related interrupt line, this bit is auto-cleared.
RX_DO	19	w	**RX DO** Writing a 1 to the bit field triggers the related interrupt line, this bit is auto-cleared.
IFF_RQ	20	w	**IFF RQ** Writing a 1 to the bit field triggers the related interrupt line, this bit is auto-cleared.
TX_EVT	21	w	**TX EVT** Writing a 1 to the bit field triggers the related interrupt line, this bit is auto-cleared.
RX_EVT	22	w	**RX EVT** Writing a 1 to the bit field triggers the related interrupt line, this bit is auto-cleared.

(table continues...)

(continued)

Field	Bits	Type	Description
BUS_ERR	23	w	**BUS ERR** Writing a 1 to the bit field triggers the related interrupt line, this bit is auto-cleared.
E_ACTIVE	24	w	**E ACTIVE** Writing a 1 to the bit field triggers the related interrupt line, this bit is auto-cleared.
E_PASSIVE	25	w	**E PASSIVE** Writing a 1 to the bit field triggers the related interrupt line, this bit is auto-cleared.
BUS_ON	26	w	**BUS ON** Writing a 1 to the bit field triggers the related interrupt line, this bit is auto-cleared.
BUS_OFF	27	w	**BUS OFF** Writing a 1 to the bit field triggers the related interrupt line, this bit is auto-cleared.
0	2:1, 14:6, 31:28	r	**Reserved** Read as all 0's; should be written with all 0's.

22.9.112 Node i operating mode register

Configuration register that is writable while the CAN communication is stopped and that is read-only after the CAN communication is started. This register defines separate operating mode options. The four configuration bits FDOE, XLOE, EFDI, and XLTR, are interrelated according to table Frame Formats defined in Operating Mode chapter.

Ni_MODE (i=0-3) Offset address: $21960_H+i*2000_H$

Node i operating mode register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				FIME	EFDI	XLTR	SFS	RSTR	MON	TXP	EFBI	PXHD	TDCE	XLOE	FDOE
r				rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rwh	rwh

Field	Bits	Type	Description
FDOE	0	rwh	**FDOE** FD Frame Format enabled. When set to 1, node is FD enabled according to ISO11898-1:2024. When set to 0, node is FD tolerant according to ISO11898-1:2024 (only Classical CAN frames used). This bit cannot be set to 1 when the static input ONLY_CC is set.

(table continues...)

(continued)

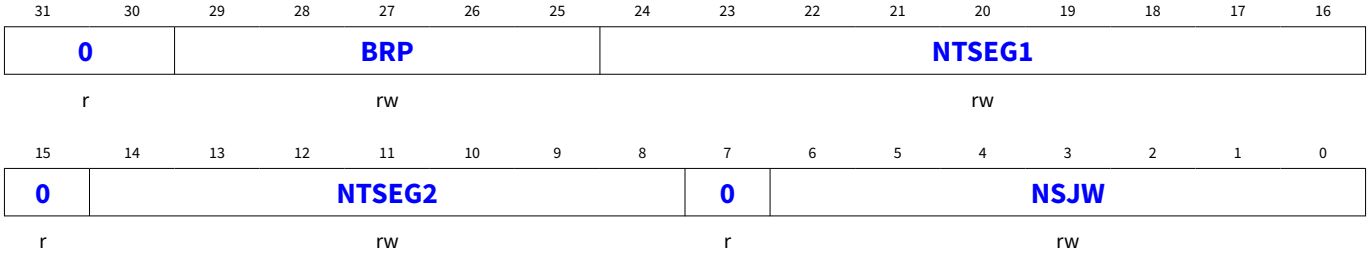
Field	Bits	Type	Description
XLOE	1	rwh	**XLOE** XL Frame Format enabled. When set to 0, node behaves according to ISO11898-1:2024, no arbitration during FDF bit. When set to 1, node behaves according to ISO11898-1:2024, arbitration during FDF bit and XLF bit. This bit cannot be set to 1 when one of the static inputs ONLY_CC or ONLY_CC_FD is set. Setting XLOE without setting FDOE is an invalid configuration.
TDCE	2	rw	**TDCE** Transmitter Delay Compensation Enabled as defined in [1].
PXHD	3	rw	**PXHD** Protocol Exception Handling Disabled.
EFBI	4	rw	**EFBI** Edge Filtering during Bus Integration. If this bit is set, the PRT requires two consecutive dominant to to detect an edge causing the reset of the bit counter for the detection of the idle condition.
TXP	5	rw	**TXP** Transmit Pause. If this bit is set, the PRT pauses for two CAN bit times before starting the next transmission after itself has successfully transmitted a frame
MON	6	rw	**MON** Monitoring Mode Enabled as defined in [1].
RSTR	7	rw	**RSTR** Restricted Mode Enabled as defined in [1].
SFS	8	rw	**SFS** Time stamp position: Start of Frame Stamping 1: Timestamps captured at the start of a frame 0: Timestamps captured at the end of a frame.
XLTR	9	rw	**XLTR** XL Transceiver Connected.
EFDI	10	rw	**EFDI** Error Flag Disable, 1 means Error Signalling is disabled as defined in [1] and the error counters REC and TEC are not incremented. When this bit is set, only CAN XL frames are transmitted and received dominant FDF or XLF bits are treated as form errors.
FIME	11	rw	**FIME** Fault Injection Module Enable, see Safety Measures chapter.
0	31:12	r	**Reserved** Read as all 0's; should be written with all 0's.

22.9.113 Node i arbitration phase nominal bit timing register

Configuration register that is writable while the CAN communication is stopped and that is read-only after the CAN communication is started. This register defines the Nominal Bit Timing as defined in [1].

Ni_NBTP (i=0-3) Offset address: $21964_H+i*2000_H$

Node i arbitration phase nominal bit timing register Kernel Reset value: $0000\ 0000_H$



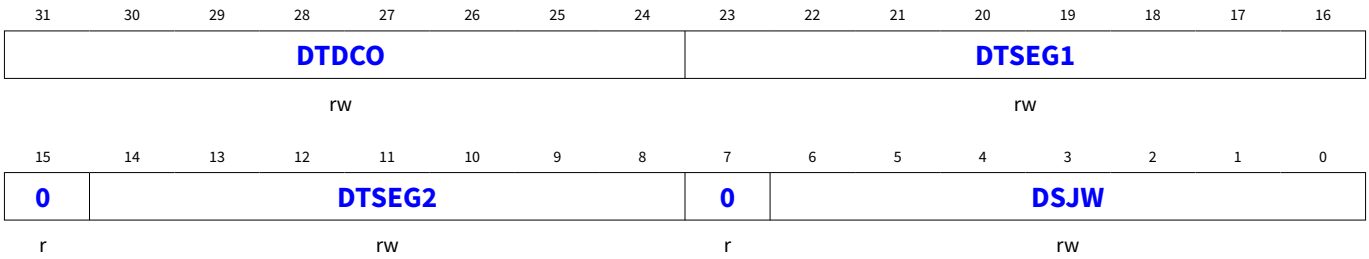
Field	Bits	Type	Description
NSJW	6:0	rw	**NSJW** Nominal SJW. Valid values for the Nominal Synchronization Jump Width NSJW are 0x00-0x7F. The actual interpretation of this value is that the Nominal Synchronization Jump Width is (NSJW + 1) TQ long.
NTSEG2	14:8	rw	**NTSEG2** Nominal Phase_Seg2. Valid values for NTSEG2 are 0x01-0x7F. This value defines the length of Phase_Seg2(N). The actual interpretation of this value is that the phase buffer segment 2 is (NTSEG2 + 1) TQ long.
NTSEG1	24:16	rw	**NTSEG1** Nominal Prop_Seg and Phase_Seg1. Valid values for NTSEG1 are 0x01-0x1FF. This value defines the sum of Prop_Seg(N) and Phase_Seg1(N). The actual interpretation of this value is that these segments together are (NTSEG1 + 1) TQ long.
BRP	29:25	rw	**BRP** Bit Rate Prescaler. Valid values for the Bit Rate Prescaler BRP are 0x00-0x1F. This value defines the length of the Time Quantum TQ for all three bit time configurations. The actual interpretation of this value is that the TQ is (BRP + 1) CLK periods long
0	7, 15, 31:30	r	**Reserved** Read as 0; should be written with 0.

22.9.114 Node i CAN FD data phase bit timing register

Configuration register that is writable while the CAN communication is stopped and that is read-only after the CAN communication is started. This register defines the FD Data Phase Bit Timing as defined in [1].

Ni_DBTP (i=0-3) Offset address: $21968_H+i*2000_H$

Node i CAN FD data phase bit timing register Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
DSJW	6:0	rw	**DSJW** FD data phase SJW. Valid values for the FD data phase Synchronization Jump Width DSJW are 0x00-0x7F. The actual interpretation of this value is that the FD data phase Synchronization Jump Width is (DSJW + 1) TQ long.
DTSEG2	14:8	rw	**DTSEG2** FD data phase Phase_Seg2. Valid values for DTSEG2 are 0x01-0x7F. This value defines the length of Phase_Seg2(D). The actual interpretation of this value is that the phase buffer segment 2 is (DTSEG2 + 1) TQ long.
DTSEG1	23:16	rw	**DTSEG1** FD data phase Prop_Seg and Phase_Seg1. Valid values for DTSEG1 are 0x00-0xFF. This value defines the sum of Prop_Seg(D) and Phase_Seg1(D). The actual interpretation of this value is that these segments together are (DTSEG1 + 1) TQ long
DTDCO	31:24	rw	**DTDCO** Transmitter Delay Compensation Offset for FD frames. Valid values for the FD Transmitter Delay Compensation Offset DTDCO is 0x00-0xFF. This configuration defines the distance between the measured delay from CAN_TX to CAN_RX and the secondary sample point SSP, measured in CLK periods. This value is used when transmitting a CAN FD frame
0	7, 15	r	**Reserved** Read as 0; should be written with 0.

22.9.115 Node i CAN XL data phase bit timing register

Configuration register that is writable while the CAN communication is stopped and is read-only after the CAN communication is started. This register defines the XL Data Phase Bit Timing as defined in [1].

Ni_XBTP (i=0-3) Offset address: $2196C_H+i*2000_H$

Node i CAN XL data phase bit timing register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
XTDCO								XTSEG1							
rw								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	XTSEG2							0	XSJW						
r	rw							r	rw						

Field	Bits	Type	Description
XSJW	6:0	rw	**XSJW** XL data phase SJW. Valid values for the XL data phase Synchronization Jump Width XSJW are 0x00-0x7F. The actual interpretation of this value is that the XL data phase Synchronization Jump Width is (XSJW + 1) TQ long

(table continues...)

(continued)

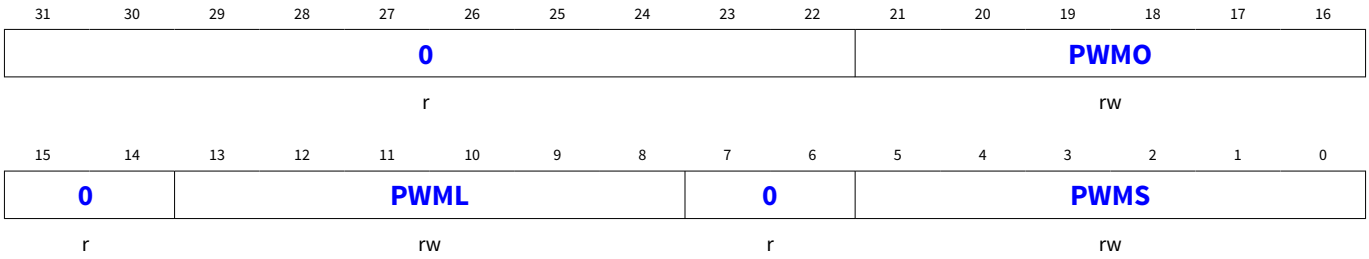
Field	Bits	Type	Description
XTSEG2	14:8	rw	**XTSEG2** XL data phase Phase_Seg2. Valid values for XTSEG2 are 0x01-0x7F. This value defines the length of Phase_Seg2(X). The actual interpretation of this value is that the phase buffer segment 2 is (XTSEG2 + 1) TQ long
XTSEG1	23:16	rw	**XTSEG1** XL data phase Prop_Seg and Phase_Seg1. Valid values for XTSEG1 are 0x00-0xFF. This value defines the sum of Prop_Seg(X) and Phase_Seg1(X). The actual interpretation of this value is that these segments together are (XTSEG1 + 1) TQ long
XTDCO	31:24	rw	**XTDCO** Transmitter Delay Compensation Offset for XL frames. Valid values for the XL Transmitter Delay Compensation Offset XTDCO is 0x00-0xFF. This configuration defines the distance between the measured delay from CAN_TX to CAN_RX and the secondary sample point SSP, measured in CLK periods. This value is used when transmitting a CAN XL frame.
0	7, 15	r	**Reserved** Read as 0; should be written with 0.

22.9.116 Node i PWME configuration register

Configuration register that is writable while the CAN communication is stopped and is read-only after the CAN communication is started. This register defines the parameters needed for the PWM coding (as described in [1]) in the PWME module for CAN XL transceivers with switchable operating modes

Ni_PCFG (i=0-3) Offset address: $21970_H+i*2000_H$

Node i PWME configuration register Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
PWMS	5:0	rw	**PWMS** PWM phase Short.
PWML	13:8	rw	**PWML** PWM phase Long.
PWMO	21:16	rw	**PWMO** PWM Offset.
0	7:6, 15:14, 31:22	r	**Reserved** Read as all 0's; should be written with all 0's.

22.9.117 Node i functional raw event status register

This register provides information about the occurrence of functional relevant events inside the MH and the PRT. A flag is set when the related event is detected, independent of Ni_FUNC_ENA. The flags remain set until the Host CPU clears them by writing a 1 to the corresponding bit position at register FUNC_CLR.

Ni_FUNC_RAW (i=0-3) Offset address: $21A00_H+i*2000_H$

Node i functional raw event status register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0				PRT_RX_EVT	PRT_TX_EVT	PRT_BUS_ON	PRT_E_ACTIVE	0	MH_STATS_IRQ	MH_RX_ABORT_IRQ	MH_TX_ABORT_IRQ	MH_TX_FILTER_IRQ	MH_RX_FILTER_IRQ	MH_STOP_IRQ	MH_TX_PQ_IRQ
r				rh	rh	rh	rh	r	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MH_RX_FQ7_IRQ	MH_RX_FQ6_IRQ	MH_RX_FQ5_IRQ	MH_RX_FQ4_IRQ	MH_RX_FQ3_IRQ	MH_RX_FQ2_IRQ	MH_RX_FQ1_IRQ	MH_RX_FQ0_IRQ	MH_TX_FQ7_IRQ	MH_TX_FQ6_IRQ	MH_TX_FQ5_IRQ	MH_TX_FQ4_IRQ	MH_TX_FQ3_IRQ	MH_TX_FQ2_IRQ	MH_TX_FQ1_IRQ	MH_TX_FQ0_IRQ
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
MH_TX_FQ0_IRQ	0	rh	**MH TX FQ0 IRQ** MH interrupt of the TX FIFO Queue 0. This interrupt is triggered when an invalid TX descriptor is fetched from this TX FIFO Queue, a TX message from that FIFO Queue is sent (if set in TX descriptor), or a TX message of that TX FIFO Queue is skipped, see description of TX_FQ_IRQ[7:0] in MH section.
MH_TX_FQ1_IRQ	1	rh	**MH TX FQ1 IRQ** MH interrupt of the TX FIFO Queue 1. Refer to the description of the MH_TX_FQ0_IRQ
MH_TX_FQ2_IRQ	2	rh	**MH TX FQ2 IRQ** MH interrupt of the TX FIFO Queue 2. Refer to the description of the MH_TX_FQ0_IRQ
MH_TX_FQ3_IRQ	3	rh	**MH TX FQ3 IRQ** MH interrupt of the TX FIFO Queue 3. Refer to the description of the MH_TX_FQ0_IRQ
MH_TX_FQ4_IRQ	4	rh	**MH TX FQ4 IRQ** MH interrupt of the TX FIFO Queue 4. Refer to the description of the MH_TX_FQ0_IRQ
MH_TX_FQ5_IRQ	5	rh	**MH TX FQ5 IRQ** MH interrupt of the TX FIFO Queue 5. Refer to the description of the MH_TX_FQ0_IRQ
MH_TX_FQ6_IRQ	6	rh	**MH TX FQ6 IRQ** MH interrupt of the TX FIFO Queue 6. Refer to the description of the MH_TX_FQ0_IRQ

(table continues...)

(continued)

Field	Bits	Type	Description
MH_TX_FQ7_IRQ	7	rh	**MH TX FQ7 IRQ** MH interrupt of the TX FIFO Queue 7. Refer to the description of the MH_TX_FQ0_IRQ
MH_RX_FQ0_IRQ	8	rh	**MH RX FQ0 IRQ** MH interrupt of the RX FIFO Queue 0. This interrupt is triggered when an invalid RX descriptor is fetched from this RX FIFO Queue, or an RX message is received (if set in RX descriptor) in this RX FIFO Queue, see description of RX_FQ_IRQ[7:0] in MH section.
MH_RX_FQ1_IRQ	9	rh	**MH RX FQ1 IRQ** MH interrupt of the RX FIFO Queue 1. Refer to the description of the MH_RX_FQ0_IRQ
MH_RX_FQ2_IRQ	10	rh	**MH RX FQ2 IRQ** MH interrupt of the RX FIFO Queue 2. Refer to the description of the MH_RX_FQ0_IRQ
MH_RX_FQ3_IRQ	11	rh	**MH RX FQ3 IRQ** MH interrupt of the RX FIFO Queue 3. Refer to the description of the MH_RX_FQ0_IRQ
MH_RX_FQ4_IRQ	12	rh	**MH RX FQ4 IRQ** MH interrupt of the RX FIFO Queue 4. Refer to the description of the MH_RX_FQ0_IRQ
MH_RX_FQ5_IRQ	13	rh	**MH RX FQ5 IRQ** MH interrupt of the RX FIFO Queue 5. Refer to the description of the MH_RX_FQ0_IRQ
MH_RX_FQ6_IRQ	14	rh	**MH RX FQ6 IRQ** MH interrupt of the RX FIFO Queue 6. Refer to the description of the MH_RX_FQ0_IRQ
MH_RX_FQ7_IRQ	15	rh	**MH RX FQ7 IRQ** MH interrupt of the RX FIFO Queue 7. Refer to the description of the MH_RX_FQ0_IRQ
MH_TX_PQ_IRQ	16	rh	**MH TX PQ IRQ** Interrupt of TX Priority Queue. Any TX message sent from the TX Priority Queue can be configured to trigger this interrupt. The SW would then need to look at the MH register Ni_TX_PQ_INT_STS to identify which slot has generated the interrupt and for which reason.
MH_STOP_IRQ	17	rh	**MH STOP IRQ** The interrupt is triggered when the PRT is stopped. The MH finishes its task and switches to idle mode.
MH_RX_FILTER_IRQ	18	rh	**MH RX FILTER IRQ** In order to track RX filtering results, an interrupt can be triggered when the comparison between an RX message header and a defined filter is successful.
MH_TX_FILTER_IRQ	19	rh	**MH TX FILTER IRQ** The interrupt is triggered when the TX filter is enabled, and a TX message is rejected.

(table continues...)

(continued)

Field	Bits	Type	Description
MH_TX_ABORT_IRQ	20	rh	**MH TX ABORT IRQ** This interrupt line is triggered when the MH needs to abort a TX message being sent to the PRT.
MH_RX_ABORT_IRQ	21	rh	**MH RX ABORT IRQ** This interrupt line is triggered when the MH needs to abort an RX message being received from PRT.
MH_STATS_IRQ	22	rh	**MH STATS IRQ** One of the RX/TX counters have reached the threshold.
PRT_E_ACTIVE	24	rh	**PRT E ACTIVE** PRT switched from Error-Passive to Error-Active state.
PRT_BUS_ON	25	rh	**PRT BUS ON** PRT started CAN communication, after start or end of BusOff.
PRT_TX_EVT	26	rh	**PRT TX EVT** PRT transmitted a valid CAN message.
PRT_RX_EVT	27	rh	**PRT RX EVT** PRT received a valid CAN message.
0	23, 31:28	r	**Reserved** Read as 0; should be written with 0.

22.9.118 Node i error raw event status register

This register provides information about the occurrence of functional error relevant events inside the MH and the PRT. A flag is set when the related event is detected, independent of Ni_ERR_ENA. The flags remain set until the Host CPU clears them by writing a 1 to the corresponding bit position at register ERR_CLR.

Ni_ERR_RAW (i=0-3) Offset address: $21A04_H+i*2000_H$

Node i error raw event status register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0			TOP_MUX_TO_ERR	0				PRT_BUS_OFF	PRT_E_PASSIVE	PRT_BUS_ERR	PRT_IFF_RQ	PRT_RX_DO	PRT_TX_DU	PRT_USOS	PRT_ABORTED
r			rh	r				rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		MH_MEM_TO_ERR	MH_WR_RESP_ERR	MH_RD_RESP_ERR	MH_DMA_CH_ERR	MH_DMA_TO_ERR	MH_DP_TO_ERR	MH_DP_DO_ERR	MH_DP_SEQ_ERR	MH_DP_PARITY_ERR	MH_AP_PARITY_ERR	MH_DESC_ERR	MH_REG_CRC_ERR	MH_MEM_SFTY_ERR	MH_RX_FILTER_ERR
r		rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
MH_RX_FILTER_ERR	0	rh	**MH RX FILTER ERR** MH RX filtering has not finished in time, i.e. current RX filtering has not been completed before next incoming RX message requires RX filtering.
MH_MEM_SFTY_ERR	1	rh	**MH MEM SFTY ERR** MH detected error in L_MEM. This interrupt is triggered when either the MEM_SFTY_CE or MEM_SFTY_UE input signal is active. The Message Handler provides the information, which signal was active, see flags MH:SFTY_INT_STS.MEM_SFTY_CE and MH:SFTY_INT_STS.MEM_SFTY_UE.
MH_REG_CRC_ERR	2	rh	**MH REG CRC ERR** MH detected CRC error at the register bank. See also description of REG_CRC_ERR in MH section.
MH_DESC_ERR	3	rh	**MH DESC ERR** CRC error detected on RX/TX descriptor or RX/TX descriptor not expected detected. A status flag can define if it is on TX or RX path, see Ni_SFTY_INT_STS register.
MH_AP_PARITY_ERR	4	rh	**MH AP PARITY ERR** MH detected parity error at address pointers, used to manage the MH Queues (RX/TX FIFO Queues and TX Priority Queues). See also description of AP_PARITY_ERR in MH section.
MH_DP_PARITY_ERR	5	rh	**MH DP PARITY ERR** MH detected parity error at RX message data (received from PRT and written to AXI system bus) respective parity error detected at TX message data (read from AXI system bus and transferred to PRT). Associated information provided by MH register Ni_ERR_INT_STS, e.g. if RX message or TX message was affected.
MH_DP_SEQ_ERR	6	rh	**MH DP SEQ ERR** MH detected an incorrect sequence at RX_MSG respective TX_MSG interfaces located between MH and PRT. Associated information provided by MH register Ni_ERR_INT_STS, e.g. if RX or TX interface was affected.
MH_DP_DO_ERR	7	rh	**MH DP DO ERR** MH detected a data overflow at RX buffer, see description of DP_DO_ERR in MH section.
MH_DP_TO_ERR	8	rh	**MH DP TO ERR** MH detected timeout at TX_MSG interface located between MH and PRT, see description of DP_TO_ERR in MH section.
MH_DMA_TO_ERR	9	rh	**MH DMA TO ERR** MH detected timeout at DMA_AXI interface, see description of DMA_TO_ERR in MH section.
MH_DMA_CH_ERR	10	rh	**MH DMA CH ERR** MH detected routing error, i.e. data received or sent are not properly routed to or from DMA channel interfaces, see description of DMA_CH_ERR in MH section.

(table continues...)

(continued)

Field	Bits	Type	Description
MH_RD_RESP_ERR	11	rh	**MH RD RESP ERR** MH detected a bus error caused by a read access to S_MEM respective L_MEM, see description of RESP_ERR in MH section.
MH_WR_RESP_ERR	12	rh	**MH WR RESP ERR** MH detected a bus error caused by a write access to S_MEM respective L_MEM, see description of RESP_ERR in MH section.
MH_MEM_TO_ERR	13	rh	**MH MEM TO ERR** MH detected timeout at local memory interface MEM_AXI, see description of MEM_TO_ERR in MH section.
PRT_ABORTED	16	rh	**PRT ABORTED** PRT detected stop of TX_MSG sequence by TX_MSG_WUSER code ABORT.
PRT_USOS	17	rh	**PRT USOS** PRT detected unexpected Start of Sequence during TX_MSG sequence.
PRT_TX_DU	18	rh	**PRT TX DU** PRT detected underrun condition at TX_MSG sequence.
PRT_RX_DO	19	rh	**PRT RX DO** PRT detected overflow condition at RX_MSG sequence.
PRT_IFF_RQ	20	rh	**PRT IFF RQ** PRT detected invalid Frame Format at TX_MSG.
PRT_BUS_ERR	21	rh	**PRT BUS ERR** PRT detected error on the CAN Bus.
PRT_E_PASSIVE	22	rh	**PRT E PASSIVE** PRT switched from Error-Active to Error-Passive state.
PRT_BUS_OFF	23	rh	**PRT BUS OFF** PRT entered Bus_Off state.
TOP_MUX_TO_ERR	28	rh	**TOP MUX TO ERR** Timeout at top-level multiplexer for HOST_AXI detected.
0	15:14, 27:24, 31:29	r	**Reserved** Read as all 0's; should be written with all 0's.

22.9.119 Node i safety raw event status register

This register provides information about the occurrence of safety relevant events inside the MH and the PRT. A flag is set when the related event is detected, independent of Ni_SAFETY_ENA. The flags remain set until the Host CPU clears them by writing a 1 to the corresponding bit position at register Ni_SAFETY_CLR.

Ni_SAFETY_RAW (i=0-3) Offset address: $21A08_H+i*2000_H$

Node i safety raw event status register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0			TOP_MUX_TO_ERR	0				PRT_BUS_OFF	PRT_E_PASSIVE	PRT_BUS_ERR	PRT_IFF_RQ	PRT_RX_DO	PRT_TX_DU	PRT_USOS	PRT_ABORTED
r			rh	r				rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		MH_MEM_TO_ERR	MH_WR_RESP_ERR	MH_RD_RESP_ERR	MH_DMA_CH_ERR	MH_DMA_TO_ERR	MH_DP_TO_ERR	MH_DP_DO_ERR	MH_DP_SEQ_ERR	MH_DP_PARITY_ERR	MH_AP_PARITY_ERR	MH_DESC_ERR	MH_REG_CRC_ERR	MH_MEM_SFTY_ERR	MH_RX_FILTER_ERR
r		rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
MH_RX_FILTER_ERR	0	rh	**MH RX FILTER ERR** MH RX filtering has not finished in time, i.e. current RX filtering has not been completed before next incoming RX message requires RX filtering.
MH_MEM_SFTY_ERR	1	rh	**MH MEM SFTY ERR** MH detected error in L_MEM. This interrupt is triggered when either the MEM_SFTY_CE or MEM_SFTY_UE input signal is active. The Message Handler provides the information, which signal was active, see flags MH:SFTY_INT_STS.MEM_SFTY_CE and MH:SFTY_INT_STS.MEM_SFTY_UE.
MH_REG_CRC_ERR	2	rh	**MH REG CRC ERR** MH detected CRC error at the register bank. See also description of REG_CRC_ERR in MH section.
MH_DESC_ERR	3	rh	**MH DESC ERR** CRC error detected on RX/TX descriptor or RX/TX descriptor not expected detected. A status flag can define if it is on TX or RX path, see Ni_SFTY_INT_STS register.
MH_AP_PARITY_ERR	4	rh	**MH AP PARITY ERR** MH detected parity error at address pointers, used to manage the MH Queues (RX/TX FIFO Queues and TX Priority Queues). See also description of AP_PARITY_ERR in MH section.
MH_DP_PARITY_ERR	5	rh	**MH DP PARITY ERR** MH detected parity error at RX message data (received from PRT and written to AXI system bus) respective parity error detected at TX message data (read from AXI system bus and transferred to PRT). Associated information provided by MH register Ni_ERR_INT_STS, e.g. if RX message or TX message was affected.
MH_DP_SEQ_ERR	6	rh	**MH DP SEQ ERR** MH detected an incorrect sequence at RX_MSG respective TX_MSG interfaces located between MH and PRT. Associated information provided by MH register Ni_ERR_INT_STS, e.g. if RX or TX interface was affected.

(table continues...)

(continued)

Field	Bits	Type	Description
MH_DP_DO_ERR	7	rh	**MH DP DO ERR** MH detected a data overflow at RX buffer, see description of DP_DO_ERR in MH section.
MH_DP_TO_ERR	8	rh	**MH DP TO ERR** MH detected timeout at TX_MSG interface located between MH and PRT, see description of DP_TO_ERR in MH section.
MH_DMA_TO_ERR	9	rh	**MH DMA TO ERR** MH detected timeout at DMA_AXI interface, see description of DMA_TO_ERR in MH section.
MH_DMA_CH_ERR	10	rh	**MH DMA CH ERR** MH detected routing error, i.e. data received or sent are not properly routed to or from DMA channel interfaces, see description of DMA_CH_ERR in MH section.
MH_RD_RESP_ERR	11	rh	**MH RD RESP ERR** MH detected a bus error caused by a read access to S_MEM respective L_MEM, see description of RESP_ERR in MH section.
MH_WR_RESP_ERR	12	rh	**MH WR RESP ERR** MH detected a bus error caused by a write access to S_MEM respective L_MEM, see description of RESP_ERR in MH section.
MH_MEM_TO_ERR	13	rh	**MH MEM TO ERR** MH detected timeout at local memory interface MEM_AXI, see description of MEM_TO_ERR in MH section.
PRT_ABORTED	16	rh	**PRT ABORTED** PRT detected stop of TX_MSG sequence by TX_MSG_WUSER code ABORT.
PRT_USOS	17	rh	**PRT USOS** PRT detected unexpected Start of Sequence during TX_MSG sequence.
PRT_TX_DU	18	rh	**PRT TX DU** PRT detected underrun condition at TX_MSG sequence.
PRT_RX_DO	19	rh	**PRT RX DO** PRT detected overflow condition at RX_MSG sequence.
PRT_IFF_RQ	20	rh	**PRT IFF RQ** PRT detected invalid Frame Format at TX_MSG.
PRT_BUS_ERR	21	rh	**PRT BUS ERR** PRT detected error on the CAN Bus.
PRT_E_PASSIVE	22	rh	**PRT E PASSIVE** PRT switched from Error-Active to Error-Passive state.
PRT_BUS_OFF	23	rh	**PRT BUS OFF** PRT entered Bus_Off state.
TOP_MUX_TO_ERR	28	rh	**TOP MUX TO ERR** Timeout at top-level multiplexer for HOST_AXI detected.

(table continues...)

(continued)

Field	Bits	Type	Description
0	15:14, 27:24, 31:29	r	**Reserved** Read as all 0's; should be written with all 0's.

22.9.120 Node i functional raw event clear register

Writing a 1 to a certain bit position clears the corresponding bit of register Ni_FUNC_RAW. Writing a ’0’ has no effect.

Ni_FUNC_CLR (i=0-3) Offset address: $21A10_H+i*2000_H$

Node i functional raw event clear register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0				PRT_RX_EVT	PRT_TX_EVT	PRT_BUS_ON	PRT_E_ACTIVE	0	MH_STATS_IRQ	MH_RX_ABORT_IRQ	MH_TX_ABORT_IRQ	MH_TX_FILTER_IRQ	MH_RX_FILTER_IRQ	MH_STOP_IRQ	MH_TX_PQ_IRQ
r				w	w	w	w	r	w	w	w	w	w	w	w

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MH_RX_FQ7_IRQ	MH_RX_FQ6_IRQ	MH_RX_FQ5_IRQ	MH_RX_FQ4_IRQ	MH_RX_FQ3_IRQ	MH_RX_FQ2_IRQ	MH_RX_FQ1_IRQ	MH_RX_FQ0_IRQ	MH_TX_FQ7_IRQ	MH_TX_FQ6_IRQ	MH_TX_FQ5_IRQ	MH_TX_FQ4_IRQ	MH_TX_FQ3_IRQ	MH_TX_FQ2_IRQ	MH_TX_FQ1_IRQ	MH_TX_FQ0_IRQ
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

Field	Bits	Type	Description
MH_TX_FQ0_IRQ	0	w	**MH TX FQ0 IRQ** Clear bit of Ni_FUNC_RAW.MH_TX_FQ0_IRQ by writing 1.
MH_TX_FQ1_IRQ	1	w	**MH TX FQ1 IRQ** Clear bit of Ni_FUNC_RAW.MH_TX_FQ1_IRQ by writing 1.
MH_TX_FQ2_IRQ	2	w	**MH TX FQ2 IRQ** Clear bit of Ni_FUNC_RAW.MH_TX_FQ2_IRQ by writing 1.
MH_TX_FQ3_IRQ	3	w	**MH TX FQ3 IRQ** Clear bit of Ni_FUNC_RAW.MH_TX_FQ3_IRQ by writing 1.
MH_TX_FQ4_IRQ	4	w	**MH TX FQ4 IRQ** Clear bit of Ni_FUNC_RAW.MH_TX_FQ4_IRQ by writing 1.
MH_TX_FQ5_IRQ	5	w	**MH TX FQ5 IRQ** Clear bit of Ni_FUNC_RAW.MH_TX_FQ5_IRQ by writing 1.
MH_TX_FQ6_IRQ	6	w	**MH TX FQ6 IRQ** Clear bit of Ni_FUNC_RAW.MH_TX_FQ6_IRQ by writing 1.
MH_TX_FQ7_IRQ	7	w	**MH TX FQ7 IRQ** Clear bit of Ni_FUNC_RAW.MH_TX_FQ7_IRQ by writing 1.

(table continues...)

(continued)

Field	Bits	Type	Description
MH_RX_FQ0_IRQ	8	w	**MH RX FQ0 IRQ** Clear bit of Ni_FUNC_RAW.MH_RX_FQ0_IRQ by writing 1.
MH_RX_FQ1_IRQ	9	w	**MH RX FQ1 IRQ** Clear bit of Ni_FUNC_RAW.MH_RX_FQ1_IRQ by writing 1.
MH_RX_FQ2_IRQ	10	w	**MH RX FQ2 IRQ** Clear bit of Ni_FUNC_RAW.MH_RX_FQ2_IRQ by writing 1.
MH_RX_FQ3_IRQ	11	w	**MH RX FQ3 IRQ** Clear bit of Ni_FUNC_RAW.MH_RX_FQ3_IRQ by writing 1.
MH_RX_FQ4_IRQ	12	w	**MH RX FQ4 IRQ** Clear bit of Ni_FUNC_RAW.MH_RX_FQ4_IRQ by writing 1.
MH_RX_FQ5_IRQ	13	w	**MH RX FQ5 IRQ** Clear bit of Ni_FUNC_RAW.MH_RX_FQ5_IRQ by writing 1.
MH_RX_FQ6_IRQ	14	w	**MH RX FQ6 IRQ** Clear bit of Ni_FUNC_RAW.MH_RX_FQ6_IRQ by writing 1.
MH_RX_FQ7_IRQ	15	w	**MH RX FQ7 IRQ** Clear bit of Ni_FUNC_RAW.MH_RX_FQ7_IRQ by writing 1.
MH_TX_PQ_IRQ	16	w	**MH TX PQ IRQ** Clear bit Ni_FUNC_RAW.MH_TX_PQ_IRQ by writing 1.
MH_STOP_IRQ	17	w	**MH STOP IRQ** Clear bit Ni_FUNC_RAW.MH_STOP_IRQ by writing 1.
MH_RX_FILTER_IRQ	18	w	**MH RX FILTER IRQ** Clear bit Ni_FUNC_RAW.MH_RX_FILTER_IRQ by writing 1.
MH_TX_FILTER_IRQ	19	w	**MH TX FILTER IRQ** Clear bit Ni_FUNC_RAW.MH_TX_FILTER_IRQ by writing 1.
MH_TX_ABORT_IRQ	20	w	**MH TX ABORT IRQ** Clear bit Ni_FUNC_RAW.MH_TX_ABORT_IRQ by writing 1.
MH_RX_ABORT_IRQ	21	w	**MH RX ABORT IRQ** Clear bit Ni_FUNC_RAW.MH_RX_ABORT_IRQ by writing 1.
MH_STATS_IRQ	22	w	**MH STATS IRQ** Clear bit Ni_FUNC_RAW.MH_STATS_IRQ by writing 1.
PRT_E_ACTIVE	24	w	**PRT E ACTIVE** Clear bit Ni_FUNC_RAW.PRT_E_ACTIVE by writing 1.
PRT_BUS_ON	25	w	**PRT BUS ON** Clear bit Ni_FUNC_RAW.PRT_BUS_ON by writing 1.
PRT_TX_EVT	26	w	**PRT TX EVT** Clear bit Ni_FUNC_RAW.PRT_TX_EVT by writing 1.
PRT_RX_EVT	27	w	**PRT RX EVT** Clear bit Ni_FUNC_RAW.PRT_RX_EVT by writing 1.
0	23, 31:28	r	**Reserved** Read as 0; should be written with 0.

22.9.121 Node i error raw event clear register

Writing a 1 to a certain bit position clears the corresponding bit of register Ni_ERR_RAW. Writing a ’0’ has no effect.

Ni_ERR_CLR (i=0-3) Offset address: $21A14_H+i*2000_H$

Node i error raw event clear register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0			TOP_MUX_TO_ERR	0				PRT_BUS_OFF	PRT_E_PASSIVE	PRT_BUS_ERR	PRT_IFF_RQ	PRT_RX_DO	PRT_TX_DU	PRT_USOS	PRT_ABORTED
r			w	r				w	w	w	w	w	w	w	w

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		MH_MEM_TO_ERR	MH_WR_RESP_ERR	MH_RD_RESP_ERR	MH_DMA_CH_ERR	MH_DMA_TO_ERR	MH_DP_TO_ERR	MH_DP_DO_ERR	MH_DP_SEQ_ERR	MH_DP_PARITY_ERR	MH_AP_PARITY_ERR	MH_DESC_ERR	MH_REG_CRC_ERR	MH_MEM_SFTY_ERR	MH_RX_FILTER_ERR
r		w	w	w	w	w	w	w	w	w	w	w	w	w	w

Field	Bits	Type	Description
MH_RX_FILTER_ERR	0	w	**MH RX FILTER ERR** Clear bit Ni_ERR_RAW.MH_RX_FILTER_ERR by writing 1.
MH_MEM_SFTY_ERR	1	w	**MH MEM SFTY ERR** Clear bit Ni_ERR_RAW.MH_MEM_SFTY_ERR by writing 1.
MH_REG_CRC_ERR	2	w	**MH REG CRC ERR** Clear bit Ni_ERR_RAW.MH_REG_CRC_ERR by writing 1.
MH_DESC_ERR	3	w	**MH DESC ERR** Clear bit Ni_ERR_RAW.MH_DESC_ERR by writing 1.
MH_AP_PARITY_ERR	4	w	**MH AP PARITY ERR** Clear bit Ni_ERR_RAW.MH_AP_PARITY_ERR by writing 1.
MH_DP_PARITY_ERR	5	w	**MH DP PARITY ERR** Clear bit Ni_ERR_RAW.MH_DP_PARITY_ERR by writing 1.
MH_DP_SEQ_ERR	6	w	**MH DP SEQ ERR** Clear bit Ni_ERR_RAW.MH_DP_SEQ_ERR by writing 1.
MH_DP_DO_ERR	7	w	**MH DP DO ERR** Clear bit Ni_ERR_RAW.MH_DP_DO_ERR by writing 1.
MH_DP_TO_ERR	8	w	**MH DP TO ERR** Clear bit Ni_ERR_RAW.MH_DP_TO_ERR by writing 1.
MH_DMA_TO_ERR	9	w	**MH DMA TO ERR** Clear bit Ni_ERR_RAW.MH_DMA_TO_ERR by writing 1.
MH_DMA_CH_ERR	10	w	**MH DMA CH ERR** Clear bit Ni_ERR_RAW.MH_DMA_CH_ERR by writing 1.

(table continues...)

(continued)

Field	Bits	Type	Description
MH_RD_RESP_ERR	11	w	**MH RD RESP ERR** Clear bit Ni_ERR_RAW.MH_RD_RESP_ERR by writing 1.
MH_WR_RESP_ERR	12	w	**MH WR RESP ERR** Clear bit Ni_ERR_RAW.MH_WR_RESP_ERR by writing 1.
MH_MEM_TO_ERR	13	w	**MH MEM TO ERR** Clear bit Ni_ERR_RAW.MH_MEM_TO_ERR by writing 1.
PRT_ABORTED	16	w	**PRT ABORTED** Clear bit Ni_ERR_RAW.PRT_ABORTED by writing 1.
PRT_USOS	17	w	**PRT USOS** Clear bit Ni_ERR_RAW.PRT_USOS by writing 1.
PRT_TX_DU	18	w	**PRT TX DU** Clear bit Ni_ERR_RAW.PRT_TX_DU by writing 1.
PRT_RX_DO	19	w	**PRT RX DO** Clear bit Ni_ERR_RAW.PRT_RX_DO by writing 1.
PRT_IFF_RQ	20	w	**PRT IFF RQ** Clear bit Ni_ERR_RAW.PRT_IFF_RQ by writing 1.
PRT_BUS_ERR	21	w	**PRT BUS ERR** Clear bit Ni_ERR_RAW.PRT_BUS_ERR by writing 1.
PRT_E_PASSIVE	22	w	**PRT E PASSIVE** Clear bit Ni_ERR_RAW.PRT_E_PASSIVE by writing 1.
PRT_BUS_OFF	23	w	**PRT BUS OFF** Clear bit Ni_ERR_RAW.PRT_BUS_OFF by writing 1.
TOP_MUX_TO_ERR	28	w	**TOP MUX TO ERR** Clear bit Ni_ERR_RAW.TOP_MUX_TO_ERR by writing 1.
0	15:14, 27:24, 31:29	r	**Reserved** Read as all 0's; should be written with all 0's.

22.9.122 Node i safety raw event clear register

Writing a 1 to a certain bit position clears the corresponding bit of register Ni_SAFETY_RAW. Writing a ’0’ has no effect.

Ni_SAFETY_CLR (i=0-3) Offset address: $21A18_H+i*2000_H$

Node i safety raw event clear register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0			TOP_MUX_TO_ERR	0				PRT_BUS_OFF	PRT_E_PASSIVE	PRT_BUS_ERR	PRT_IFF_RQ	PRT_RX_DO	PRT_TX_DU	PRT_USOS	PRT_ABORTED
r			w	r				w	w	w	w	w	w	w	w

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		MH_MEM_TO_ERR	MH_WR_RESP_ERR	MH_RD_RESP_ERR	MH_DMA_CH_ERR	MH_DMA_TO_ERR	MH_DP_TO_ERR	MH_DP_DO_ERR	MH_DP_SEQ_ERR	MH_DP_PARITY_ERR	MH_AP_PARITY_ERR	MH_DESC_ERR	MH_REG_CRC_ERR	MH_MEM_SFTY_ERR	MH_RX_FILTER_ERR
r		w	w	w	w	w	w	w	w	w	w	w	w	w	w

Field	Bits	Type	Description
MH_RX_FILTER_ERR	0	w	**MH RX FILTER ERR** Clear bit Ni_SAFETY_RAW.MH_RX_FILTER_ERR by writing 1.
MH_MEM_SFTY_ERR	1	w	**MH MEM SFTY ERR** Clear bit Ni_SAFETY_RAW.MH_MEM_SFTY_ERR by writing 1.
MH_REG_CRC_ERR	2	w	**MH REG CRC ERR** Clear bit Ni_SAFETY_RAW.MH_REG_CRC_ERR by writing 1.
MH_DESC_ERR	3	w	**MH DESC ERR** Clear bit Ni_SAFETY_RAW.MH_DESC_ERR by writing 1.
MH_AP_PARITY_ERR	4	w	**MH AP PARITY ERR** Clear bit Ni_SAFETY_RAW.MH_AP_PARITY_ERR by writing 1.
MH_DP_PARITY_ERR	5	w	**MH DP PARITY ERR** Clear bit Ni_SAFETY_RAW.MH_DP_PARITY_ERR by writing 1.
MH_DP_SEQ_ERR	6	w	**MH DP SEQ ERR** Clear bit Ni_SAFETY_RAW.MH_DP_SEQ_ERR by writing 1.
MH_DP_DO_ERR	7	w	**MH DP DO ERR** Clear bit Ni_SAFETY_RAW.MH_DP_DO_ERR by writing 1.
MH_DP_TO_ERR	8	w	**MH DP TO ERR** Clear bit Ni_SAFETY_RAW.MH_DP_TO_ERR by writing 1.
MH_DMA_TO_ERR	9	w	**MH DMA TO ERR** Clear bit Ni_SAFETY_RAW.MH_DMA_TO_ERR by writing 1.
MH_DMA_CH_ERR	10	w	**MH DMA CH ERR** Clear bit Ni_SAFETY_RAW.MH_DMA_CH_ERR by writing 1.
MH_RD_RESP_ERR	11	w	**MH RD RESP ERR** Clear bit Ni_SAFETY_RAW.MH_RD_RESP_ERR by writing 1.
MH_WR_RESP_ERR	12	w	**MH WR RESP ERR** Clear bit Ni_SAFETY_RAW.MH_WR_RESP_ERR by writing 1.
MH_MEM_TO_ERR	13	w	**MH MEM TO ERR** Clear bit Ni_SAFETY_RAW.MH_MEM_TO_ERR by writing 1.

(table continues...)

(continued)

Field	Bits	Type	Description
PRT_ABORTED	16	w	**PRT ABORTED** Clear bit Ni_SAFETY_RAW.PRT_ABORTED by writing 1.
PRT_USOS	17	w	**PRT USOS** Clear bit Ni_SAFETY_RAW.PRT_USOS by writing 1.
PRT_TX_DU	18	w	**PRT TX DU** Clear bit Ni_SAFETY_RAW.PRT_TX_DU by writing 1.
PRT_RX_DO	19	w	**PRT RX DO** Clear bit Ni_SAFETY_RAW.PRT_RX_DO by writing 1.
PRT_IFF_RQ	20	w	**PRT IFF RQ** Clear bit Ni_SAFETY_RAW.PRT_IFF_RQ by writing 1.
PRT_BUS_ERR	21	w	**PRT BUS ERR** Clear bit Ni_SAFETY_RAW.PRT_BUS_ERR by writing 1.
PRT_E_PASSIVE	22	w	**PRT E PASSIVE** Clear bit Ni_SAFETY_RAW.PRT_E_PASSIVE by writing 1.
PRT_BUS_OFF	23	w	**PRT BUS OFF** Clear bit Ni_SAFETY_RAW.PRT_BUS_OFF by writing 1.
TOP_MUX_TO_ERR	28	w	**TOP MUX TO ERR** Clear bit Ni_SAFETY_RAW.TOP_MUX_TO_ERR by writing 1.
0	15:14, 27:24, 31:29	r	**Reserved** Read as all 0's; should be written with all 0's.

22.9.123 Node i functional raw event enable register

Any bit in the Ni_FUNC_ENA register enables the corresponding bit in the Ni_FUNC_RAW to trigger the interrupt line FUNC_INT. The interrupt line gets active high, when at least one RAW/ENA pair is 1, e.g. Ni_FUNC_RAW.MH_TX_FQ_IRQ = Ni_FUNC_ENA.MH_TX_FQ_IRQ = 1

Ni_FUNC_ENA (i=0-3) Offset address: $21A20_H+i*2000_H$

Node i functional raw event enable register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0				PRT_RX_EVT	PRT_TX_EVT	PRT_BUS_ON	PRT_E_ACTIVE	0	MH_STATS_IRQ	MH_RX_ABORT_IRQ	MH_TX_ABORT_IRQ	MH_TX_FILTER_IRQ	MH_RX_FILTER_IRQ	MH_STOP_IRQ	MH_TX_PQ_IRQ
r				rw	rw	rw	rw	r	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MH_RX_FQ7_IRQ	MH_RX_FQ6_IRQ	MH_RX_FQ5_IRQ	MH_RX_FQ4_IRQ	MH_RX_FQ3_IRQ	MH_RX_FQ2_IRQ	MH_RX_FQ1_IRQ	MH_RX_FQ0_IRQ	MH_TX_FQ7_IRQ	MH_TX_FQ6_IRQ	MH_TX_FQ5_IRQ	MH_TX_FQ4_IRQ	MH_TX_FQ3_IRQ	MH_TX_FQ2_IRQ	MH_TX_FQ1_IRQ	MH_TX_FQ0_IRQ
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
MH_TX_FQ0_IRQ	0	rw	**MH TX FQ0 IRQ** Enable Ni_FUNC_RAW.MH_TX_FQ0_IRQ to activate FUNC_INT.
MH_TX_FQ1_IRQ	1	rw	**MH TX FQ1 IRQ** Enable Ni_FUNC_RAW.MH_TX_FQ1_IRQ to activate FUNC_INT.
MH_TX_FQ2_IRQ	2	rw	**MH TX FQ2 IRQ** Enable Ni_FUNC_RAW.MH_TX_FQ2_IRQ to activate FUNC_INT.
MH_TX_FQ3_IRQ	3	rw	**MH TX FQ3 IRQ** Enable Ni_FUNC_RAW.MH_TX_FQ3_IRQ to activate FUNC_INT.
MH_TX_FQ4_IRQ	4	rw	**MH TX FQ4 IRQ** Enable Ni_FUNC_RAW.MH_TX_FQ4_IRQ to activate FUNC_INT.
MH_TX_FQ5_IRQ	5	rw	**MH TX FQ5 IRQ** Enable Ni_FUNC_RAW.MH_TX_FQ5_IRQ to activate FUNC_INT.
MH_TX_FQ6_IRQ	6	rw	**MH TX FQ6 IRQ** Enable Ni_FUNC_RAW.MH_TX_FQ6_IRQ to activate FUNC_INT.
MH_TX_FQ7_IRQ	7	rw	**MH TX FQ7 IRQ** Enable Ni_FUNC_RAW.MH_TX_FQ7_IRQ to activate FUNC_INT.
MH_RX_FQ0_IRQ	8	rw	**MH RX FQ0 IRQ** Enable Ni_FUNC_RAW.MH_RX_FQ0_IRQ to activate FUNC_INT.
MH_RX_FQ1_IRQ	9	rw	**MH RX FQ1 IRQ** Enable Ni_FUNC_RAW.MH_RX_FQ1_IRQ to activate FUNC_INT.
MH_RX_FQ2_IRQ	10	rw	**MH RX FQ2 IRQ** Enable Ni_FUNC_RAW.MH_RX_FQ2_IRQ to activate FUNC_INT.
MH_RX_FQ3_IRQ	11	rw	**MH RX FQ3 IRQ** Enable Ni_FUNC_RAW.MH_RX_FQ3_IRQ to activate FUNC_INT.
MH_RX_FQ4_IRQ	12	rw	**MH RX FQ4 IRQ** Enable Ni_FUNC_RAW.MH_RX_FQ4_IRQ to activate FUNC_INT.
MH_RX_FQ5_IRQ	13	rw	**MH RX FQ5 IRQ** Enable Ni_FUNC_RAW.MH_RX_FQ5_IRQ to activate FUNC_INT.
MH_RX_FQ6_IRQ	14	rw	**MH RX FQ6 IRQ** Enable Ni_FUNC_RAW.MH_RX_FQ6_IRQ to activate FUNC_INT.
MH_RX_FQ7_IRQ	15	rw	**MH RX FQ7 IRQ** Enable Ni_FUNC_RAW.MH_RX_FQ7_IRQ to activate FUNC_INT.
MH_TX_PQ_IRQ	16	rw	**MH TX PQ IRQ** Enable Ni_FUNC_RAW.MH_TX_PQ_IRQ to activate FUNC_INT.
MH_STOP_IRQ	17	rw	**MH STOP IRQ** Enable Ni_FUNC_RAW.MH_STOP_IRQ to activate FUNC_INT.
MH_RX_FILTER_IRQ	18	rw	**MH RX FILTER IRQ** Enable Ni_FUNC_RAW.MH_RX_FILTER_IRQ to activate FUNC_INT.

(table continues...)

(continued)

Field	Bits	Type	Description
MH_TX_FILTER_IRQ	19	rw	**MH TX FILTER IRQ** Enable Ni_FUNC_RAW.MH_TX_FILTER_IRQ to activate FUNC_INT.
MH_TX_ABORT_IRQ	20	rw	**MH TX ABORT IRQ** Enable Ni_FUNC_RAW.MH_TX_ABORT_IRQ to activate FUNC_INT.
MH_RX_ABORT_IRQ	21	rw	**MH RX ABORT IRQ** Enable Ni_FUNC_RAW.MH_RX_ABORT_IRQ to activate FUNC_INT.
MH_STATS_IRQ	22	rw	**MH STATS IRQ** Enable Ni_FUNC_RAW.MH_STATS_IRQ to activate FUNC_INT.
PRT_E_ACTIVE	24	rw	**PRT E ACTIVE** Enable Ni_FUNC_RAW.PRT_E_ACTIVE to activate FUNC_INT.
PRT_BUS_ON	25	rw	**PRT BUS ON** Enable Ni_FUNC_RAW.PRT_BUS_ON to activate FUNC_INT.
PRT_TX_EVT	26	rw	**PRT TX EVT** Enable Ni_FUNC_RAW.PRT_TX_EVT to activate FUNC_INT.
PRT_RX_EVT	27	rw	**PRT RX EVT** Enable Ni_FUNC_RAW.PRT_RX_EVT to activate FUNC_INT.
0	23, 31:28	r	**Reserved** Read as 0; should be written with 0.

22.9.124 Node i error raw event enable register

Any bit in the Ni_ERR_ENA register enables the corresponding bit in the Ni_ERR_RAW to trigger the interrupt line ERR_INT. The interrupt line gets active high, when at least one RAW/ENA pair is 1, e.g. Ni_ERR_RAW.MH_TX_FQ_IRQ = Ni_ERR_ENA.MH_TX_FQ_IRQ = 1

Ni_ERR_ENA (i=0-3) Offset address: $21A24_H+i*2000_H$

Node i error raw event enable register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0			TOP_MUX_TO_ERR	0				PRT_BUS_OFF	PRT_E_PASSIVE	PRT_BUS_ERR	PRT_IFF_RQ	PRT_RX_DO	PRT_TX_DU	PRT_USOS	PRT_ABORTED
r			rw	r				rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		MH_MEM_TO_ERR	MH_WR_RESP_ERR	MH_RD_RESP_ERR	MH_DMA_CH_ERR	MH_DMA_TO_ERR	MH_DP_TO_ERR	MH_DP_DO_ERR	MH_DP_SEQ_ERR	MH_DP_PARITY_ERR	MH_AP_PARITY_ERR	MH_DESC_ERR	MH_REG_CRC_ERR	MH_MEM_SFTY_ERR	MH_RX_FILTER_ERR
r		rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
MH_RX_FILTER_ERR	0	rw	**MH RX FILTER ERR** Enable Ni_ERR_RAW.MH_RX_FILTER_ERR to activate ERR_INT.
MH_MEM_SFTY_ERR	1	rw	**MH MEM SFTY ERR** Enable Ni_ERR_RAW.MH_MEM_SFTY_ERR to activate ERR_INT.
MH_REG_CRC_ERR	2	rw	**MH REG CRC ERR** Enable Ni_ERR_RAW.MH_REG_CRC_ERR to activate ERR_INT.
MH_DESC_ERR	3	rw	**MH DESC ERR** Enable Ni_ERR_RAW.MH_DESC_ERR to activate ERR_INT.
MH_AP_PARITY_ERR	4	rw	**MH AP PARITY ERR** Enable Ni_ERR_RAW.MH_AP_PARITY_ERR to activate ERR_INT.
MH_DP_PARITY_ERR	5	rw	**MH DP PARITY ERR** Enable Ni_ERR_RAW.MH_DP_PARITY_ERR to activate ERR_INT.
MH_DP_SEQ_ERR	6	rw	**MH DP SEQ ERR** Enable Ni_ERR_RAW.MH_DP_SEQ_ERR to activate ERR_INT.
MH_DP_DO_ERR	7	rw	**MH DP DO ERR** Enable Ni_ERR_RAW.MH_DP_DO_ERR to activate ERR_INT.
MH_DP_TO_ERR	8	rw	**MH DP TO ERR** Enable Ni_ERR_RAW.MH_DP_TO_ERR to activate ERR_INT.
MH_DMA_TO_ERR	9	rw	**MH DMA TO ERR** Enable Ni_ERR_RAW.MH_DMA_TO_ERR to activate ERR_INT.
MH_DMA_CH_ERR	10	rw	**MH DMA CH ERR** Enable Ni_ERR_RAW.MH_DMA_CH_ERR to activate ERR_INT.
MH_RD_RESP_ERR	11	rw	**MH RD RESP ERR** Enable Ni_ERR_RAW.MH_RD_RESP_ERR to activate ERR_INT.
MH_WR_RESP_ERR	12	rw	**MH WR RESP ERR** Enable Ni_ERR_RAW.MH_WR_RESP_ERR to activate ERR_INT.
MH_MEM_TO_ERR	13	rw	**MH MEM TO ERR** Enable Ni_ERR_RAW.MH_MEM_TO_ERR to activate ERR_INT.
PRT_ABORTED	16	rw	**PRT ABORTED** Enable Ni_ERR_RAW.PRT_ABORTED to activate ERR_INT.
PRT_USOS	17	rw	**PRT USOS** Enable Ni_ERR_RAW.PRT_USOS to activate ERR_INT.
PRT_TX_DU	18	rw	**PRT TX DU** Enable Ni_ERR_RAW.PRT_TX_DU to activate ERR_INT.
PRT_RX_DO	19	rw	**PRT RX DO** Enable Ni_ERR_RAW.PRT_RX_DO to activate ERR_INT.
PRT_IFF_RQ	20	rw	**PRT IFF RQ** Enable Ni_ERR_RAW.PRT_IFF_RQ to activate ERR_INT.

(table continues...)

(continued)

Field	Bits	Type	Description
PRT_BUS_ERR	21	rw	**PRT BUS ERR** Enable Ni_ERR_RAW.PRT_BUS_ERR to activate ERR_INT.
PRT_E_PASSIVE	22	rw	**PRT E PASSIVE** Enable Ni_ERR_RAW.PRT_E_PASSIVE to activate ERR_INT.
PRT_BUS_OFF	23	rw	**PRT BUS OFF** Enable Ni_ERR_RAW.PRT_BUS_OFF to activate ERR_INT.
TOP_MUX_TO_ERR	28	rw	**TOP MUX TO ERR** Enable Ni_ERR_RAW.TOP_MUX_TO_ERR to activate ERR_INT.
0	15:14, 27:24, 31:29	r	**Reserved** Read as all 0's; should be written with all 0's.

22.9.125 Node i safety raw event enable register

Any bit in the Ni_SAFETY_ENA register enables the corresponding bit in the Ni_SAFETY_RAW to trigger the interrupt line SAFETY_INT. The interrupt line gets active high, when at least one RAW/ENA pair is 1, e.g. Ni_SAFETY_RAW.MH_TX_FQ_IRQ = Ni_SAFETY_ENA.MH_TX_FQ_IRQ = 1

Ni_SAFETY_ENA (i=0-3) Offset address: $21A28_H+i*2000_H$

Node i safety raw event enable register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0			TOP_MUX_TO_ERR	0				PRT_BUS_OFF	PRT_E_PASSIVE	PRT_BUS_ERR	PRT_IFF_RQ	PRT_RX_DO	PRT_TX_DU	PRT_USOS	PRT_ABORTED
r			rw	r				rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		MH_MEM_TO_ERR	MH_WR_RESP_ERR	MH_RD_RESP_ERR	MH_DMA_CH_ERR	MH_DMA_TO_ERR	MH_DP_TO_ERR	MH_DP_DO_ERR	MH_DP_SEQ_ERR	MH_DP_PARITY_ERR	MH_AP_PARITY_ERR	MH_DESC_ERR	MH_REG_CRC_ERR	MH_MEM_SFTY_ERR	MH_RX_FILTER_ERR
r		rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
MH_RX_FILTER_ERR	0	rw	**MH RX FILTER ERR** Enable Ni_SAFETY_RAW.MH_RX_FILTER_ERR to activate SAFETY_INT.
MH_MEM_SFTY_ERR	1	rw	**MH MEM SFTY ERR** Enable Ni_SAFETY_RAW.MH_MEM_SFTY_ERR to activate SAFETY_INT.
MH_REG_CRC_ERR	2	rw	**MH REG CRC ERR** Enable Ni_SAFETY_RAW.MH_REG_CRC_ERR to activate SAFETY_INT.

(table continues...)

(continued)

Field	Bits	Type	Description
MH_DESC_ERR	3	rw	**MH DESC ERR** Enable Ni_SAFETY_RAW.MH_DESC_ERR to activate SAFETY_INT.
MH_AP_PARITY_ERR	4	rw	**MH AP PARITY ERR** Enable Ni_SAFETY_RAW.MH_AP_PARITY_ERR to activate SAFETY_INT.
MH_DP_PARITY_ERR	5	rw	**MH DP PARITY ERR** Enable Ni_SAFETY_RAW.MH_DP_PARITY_ERR to activate SAFETY_INT.
MH_DP_SEQ_ERR	6	rw	**MH DP SEQ ERR** Enable Ni_SAFETY_RAW.MH_DP_SEQ_ERR to activate SAFETY_INT.
MH_DP_DO_ERR	7	rw	**MH DP DO ERR** Enable Ni_SAFETY_RAW.MH_DP_DO_ERR to activate SAFETY_INT.
MH_DP_TO_ERR	8	rw	**MH DP TO ERR** Enable Ni_SAFETY_RAW.MH_DP_TO_ERR to activate SAFETY_INT.
MH_DMA_TO_ERR	9	rw	**MH DMA TO ERR** Enable Ni_SAFETY_RAW.MH_DMA_TO_ERR to activate SAFETY_INT.
MH_DMA_CH_ERR	10	rw	**MH DMA CH ERR** Enable Ni_SAFETY_RAW.MH_DMA_CH_ERR to activate SAFETY_INT.
MH_RD_RESP_ERR	11	rw	**MH RD RESP ERR** Enable Ni_SAFETY_RAW.MH_RD_RESP_ERR to activate SAFETY_INT.
MH_WR_RESP_ERR	12	rw	**MH WR RESP ERR** Enable Ni_SAFETY_RAW.MH_WR_RESP_ERR to activate SAFETY_INT.
MH_MEM_TO_ERR	13	rw	**MH MEM TO ERR** Enable Ni_SAFETY_RAW.MH_MEM_TO_ERR to activate SAFETY_INT.
PRT_ABORTED	16	rw	**PRT ABORTED** Enable Ni_SAFETY_RAW.PRT_ABORTED to activate SAFETY_INT.
PRT_USOS	17	rw	**PRT USOS** Enable Ni_SAFETY_RAW.PRT_USOS to activate SAFETY_INT.
PRT_TX_DU	18	rw	**PRT TX DU** Enable Ni_SAFETY_RAW.PRT_TX_DU to activate SAFETY_INT.
PRT_RX_DO	19	rw	**PRT RX DO** Enable Ni_SAFETY_RAW.PRT_RX_DO to activate SAFETY_INT.
PRT_IFF_RQ	20	rw	**PRT IFF RQ** Enable Ni_SAFETY_RAW.PRT_IFF_RQ to activate SAFETY_INT.
PRT_BUS_ERR	21	rw	**PRT BUS ERR** Enable Ni_SAFETY_RAW.PRT_BUS_ERR to activate SAFETY_INT.
PRT_E_PASSIVE	22	rw	**PRT E PASSIVE** Enable Ni_SAFETY_RAW.PRT_E_PASSIVE to activate SAFETY_INT.
PRT_BUS_OFF	23	rw	**PRT BUS OFF** Enable Ni_SAFETY_RAW.PRT_BUS_OFF to activate SAFETY_INT.

(table continues...)

(continued)

Field	Bits	Type	Description
TOP_MUX_TO_ERR	28	rw	**TOP MUX TO ERR** Enable Ni_SAFETY_RAW.TOP_MUX_TO_ERR to activate SAFETY_INT.
0	15:14, 27:24, 31:29	r	**Reserved** Read as all 0's; should be written with all 0's.

22.9.126 Node i IRC configuration register

This register shows the hardware configuration of the IRC concerning the capturing mode of the event inputs. The IP internal events signals coming from the MH and the PRT require an 'edge sensitive' capturing. That is why the value of this register is 0x7 and cannot be changed.

Ni_CAPTURING_MODE (i=0-3) Offset address: $21A30_H+i*2000_H$

Node i IRC configuration register Kernel Reset value: $0000\ 0007_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

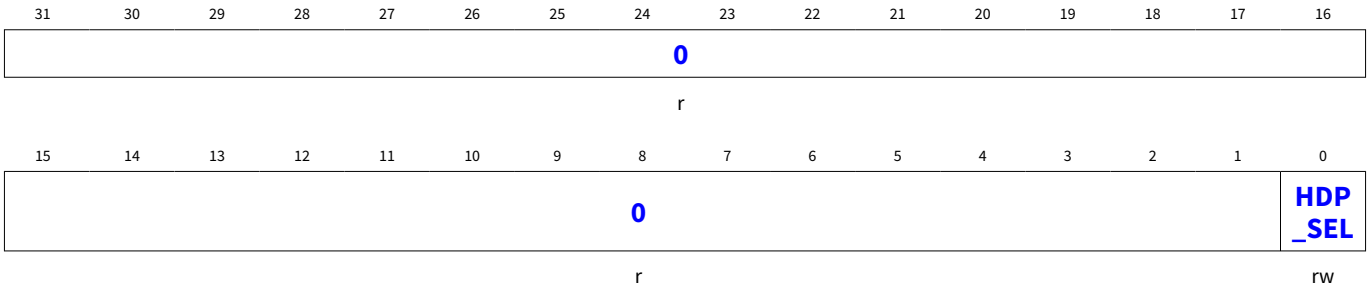
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0													SAFETY	ERR	FUNC
r													r	r	r

Field	Bits	Type	Description
FUNC	0	r	**FUNC** Capturing mode of Ni_FUNC_RAW register. 0 = Level sensitive 1 = Edge sensitive
ERR	1	r	**ERR** Capturing mode of ERR RAW register. 0 = Level sensitive 1 = Edge sensitive
SAFETY	2	r	**SAFETY** Capturing mode of SAFETY RAW register. 0 = Level sensitive 1 = Edge sensitive
0	31:3	r	**Reserved** Read as all 0's; should be written with all 0's.

22.9.127 Node i hardware debug port control register

Ni_HDP (i=0-3) Offset address: $21A40_H+i*2000_H$

Node i hardware debug port control register Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
HDP_SEL	0	rw	**HDP SEL** Select the driver of the Hardware Debug Port. See also chapter HDP. 0 = Message Handler 1 = Protocol Controller
0	31:1	r	**Reserved** Read as all 0's; should be written with all 0's.

22.10 Debug information

Hard suspend

Hard suspend is enabled when OCS.SUS is set to 1_B. Upon OCDS suspend trigger, both fCANXL and fCANXLH clocks are switched off immediately. The status of the suspend state can be monitored through OCS.SUSTA status bit-field. It is not recommended to use this mode in normal CAN or CAN XL applications, as they are meant only for debugging of the module.

Soft suspend

Soft suspend in enabled when OCS.SUS is set to 2_H. User must ensure that the X_CAN node's protocol controller and message handler are stopped, before entering the soft suspend mode. Upon OCDS suspend trigger, both fCANXL and fCANXLH clocks are switched off. The status of the suspend state can be monitored through OCS.SUSTA status bit-field.

OTGB trigger sets

Each X_CAN proved a set of 16 signals which can be traced using OTGB0/1 bus. X_CAN's HDP is selectable through Ni_HDP.HDP_SEL bit-field. The X_CAN node which needs to be traced is selectable through DEBUG_CTL.NODE_SEL bit-field. The OTGB bus 0 or 1 is selected through OCS.TGB bit-field.

***Note**: Limitation: "CLK" signals which are part of HDP[15] cannot be traced by user.*

22.11 References

1. ISO 11898-1:2024 CAN Data link layer and physical coding sublayer
2. AXI4 ARM IHI 0022E (ID022613)

22.12 CANXL revision history

Reference	Description of change(s)
Date range: 2024-08-17 to 2024-11-18	
RX FIFO queue in continuous mode	• Adding the configuration of register Ni_RX_FQ_RD_ADD_PTn to the programming instructions for continuous mode
Controller Area Network XL interface (CANXL), Feature list, Functional description, PWME, References, Feature list, X_CAN, Feature list, Functional overview, Protocol controller, Functional overview, Starting and stopping the module, Node i operating mode register, Transceiver interface , TX-SCAN	• Replacing references to CiA 610-1 and ISO 11898-1:2015 by ISO 11898-1:2024
Controller Area Network XL interface (CANXL), Feature list, Protocol controller, Feature list and Functional overview	• Updating the maximum supported CAN XL baudrate from 15 MBit/s to 20 MBit/s

22.13 TC4Dx CAN-XL information

22.13.1 TC4Dx CANXL configuration

Table 1071 TC4Dx CANXL configuration

Parameter	CANXL0
Number of CANXL nodes (N_NODES)	4
RAM size (in Kbytes)	16

22.13.2 TC4Dx CANXL features

- Supports 4 CAN XL nodes
- 16 Kbytes of shared configuration RAM

22.13.3 TC4Dx CANXL functional description

In TC4Dx device, the CANXL modules are connected to ComPB bus interface. The CANXL chapter refers to it as FPI bus.

The following restriction applies for CANXL implementation in TC4Dx

- Software must ensure that X_CAN node's Message Handler and Protocol Controller are stopped, before Kernel reset request via KRST or via module group reset trigger (when enabled through MODULE_RST_CTRLA.GRSTEN bit-field).

22.13.4 TC4Dx CANXL registers

22.13.4.1 Memory overview tables of CANXL

Table 1072 Memory overview - CANXL (ascending address)

Short name	Long name	Address
CANXL0_RAM	Embedded SRAM for CAN XL nodes configuration (04000_H Byte)	$F47C0000_H$

22.13.4.2 Register address space - CANXL

Table 1073 Registers address space - CANXL

Key: The module name in brackets () indicates a memory section name. Without brackets the reference is to a functional block.

Module	Base address	End address	Note
CANXL0	$F47D0000_H$	$F4FFFFFF_H$	FPI slave interface for SFR registers
(CANXL0_RAM)	$F47C0000_H$	$F47C3FFF_H$	FPI slave interface for L_MEM

22.13.4.3 Register overview - access mode glossary

Table 1074 Register overview - access mode glossary

Keyword	Description
E	Access protection using PROT register CANXL0_PROTE .
SE	Access protection using PROT register CANXL0_PROTSE .
APU-PM	Protection group consisting of registers CANXL0_MODULE_ACCEN_WRA , CANXL0_MODULE_ACCEN_WRB , CANXL0_MODULE_ACCEN_RDA , CANXL0_MODULE_ACCEN_RDB , CANXL0_MODULE_ACCEN_VM , CANXL0_MODULE_ACCEN_PRS .
PM	Access protection using *APU-PM* registers.
APU-PNi (i=0-3)	Protection group consisting of registers CANXL0_NODEi_ACCENNODE_WRA , CANXL0_NODEi_ACCENNODE_WRB , CANXL0_NODEi_ACCENNODE_RDA , CANXL0_NODEi_ACCENNODE_RDB , CANXL0_NODEi_ACCENNODE_VM , CANXL0_NODEi_ACCENNODE_PRS .
PNi	Access protection using *APU-PNi* registers.
U	No access restrictions.
BE	Always returns a Bus Error.
PROT	Access restrictions as defined in the PROT register access rules.
32	Access only when using 32-bit width.

22.13.4.4 Registers overview - CANXL0 (ascending offset address)

Table 1075 Registers overview - CANXL0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CANXL0_ID	Module Identification Register	00000_H	U	BE	PowerOn Reset	4310
CANXL0_OCS	OCDS Control and Status Register	00004_H	PM	SV, PM	Debug Reset	4310
CANXL0_MODULE_CLC	Module Clock Control Register	00008_H	PM	PM, SV, E	Application Reset	4312
CANXL0_MODULE_RST_CTRLA	Module reset control register A	$0000C_H$	PM	PM, SV, E	Application Reset	4312
CANXL0_MODULE_RST_CTRLB	Module reset control register B	00010_H	PM	PM, SV, E	Application Reset	4313
CANXL0_MODULE_RST_STAT	Module reset status register	00014_H	PM	BE	Application Reset	4314
CANXL0_MODULE_ACCEN_WRA	Module Write access enable register A	00018_H	U	SE, SV	Application Reset	4314
CANXL0_MODULE_ACCEN_WRB	Module Write access enable register B	$0001C_H$	U	SE, SV	Application Reset	4315
CANXL0_MODULE_ACCEN_RDA	Module Read access enable register A	00020_H	U	SE, SV	Application Reset	4315
CANXL0_MODULE_ACCEN_RDB	Module Read access enable register B	00024_H	U	SE, SV	Application Reset	4316
CANXL0_MODULE_ACCEN_VM	Module VM access enable register	00028_H	U	SE, SV	Application Reset	4316
CANXL0_MODULE_ACCEN_PRS	Module PRS access enable register	$0002C_H$	U	SE, SV	Application Reset	4317
CANXL0_PROTE	PROT Register Endinit	00038_H	U	SV, PROT	Application Reset	4318
CANXL0_PROTSE	PROT Register Safe Endinit	$0003C_H$	U	SV, PROT	Application Reset	4319
CANXL0_CLKEN	Clock enable	00040_H	clk, PM	clk, PM, SV, E	Application Reset	4321
CANXL0_NODEi_RST_CTRLA (i=0-3)	Node i reset control register A	00100_H+i *100_H	PNi	PNi, SV, E	Application Reset	4322
CANXL0_NODEi_RST_CTRLB (i=0-3)	Node i reset control register B	00104_H+i *100_H	PNi	PNi, SV, E	Application Reset	4322
CANXL0_NODEi_RST_STAT (i=0-3)	Node i reset status register	00108_H+i *100_H	PNi	BE	Application Reset	4323

(table continues...)

Table 1075 (continued) Registers overview - CANXL0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CANXL0_NODEi_ACCENNODE_WRA (i=0-3)	Node i write access enable register A	$0010C_H$+i*100_H	U	SE, SV	Application Reset	4324
CANXL0_NODEi_ACCENNODE_WRB (i=0-3)	Node i write access enable register B	00110_H+i*100_H	U	SE, SV	Application Reset	4324
CANXL0_NODEi_ACCENNODE_RDA (i=0-3)	Node i read access enable register A	00114_H+i*100_H	U	SE, SV	Application Reset	4325
CANXL0_NODEi_ACCENNODE_RDB (i=0-3)	Node i read access enable register B	00118_H+i*100_H	U	SE, SV	Application Reset	4326
CANXL0_NODEi_ACCENNODE_VM (i=0-3)	Node i VM access enable register	$0011C_H$+i*100_H	U	SE, SV	Application Reset	4326
CANXL0_NODEi_ACCENNODE_PRS (i=0-3)	Node i PRS access enable register	00120_H+i*100_H	U	SE, SV	Application Reset	4327
CANXL0_NODEi_ACCENNODE_RGNLA (i=0-3)	Node i region lower address register	00124_H+i*100_H	U	SE, SV	Application Reset	4328
CANXL0_NODEi_ACCENNODE_RGNUA (i=0-3)	Node i region upper address register	00128_H+i*100_H	U	SE, SV	Application Reset	4328
CANXL0_NODEi_VMPRSCONFIG (i=0-3)	Node i VM and PRS configuration Register	$0012C_H$+i*100_H	U	SE, SV	Application Reset	4328
CANXL0_NODEi_PORTCTRL (i=0-3)	Node i Port Control Register	00130_H+i*100_H	PNi	PNi	Kernel Reset	4329
CANXL0_NODEi_MTI_RAW (i=0-3)	Node i message transfer interrupt event register	00134_H+i*100_H	PNi	BE	Kernel Reset	4330
CANXL0_NODEi_MTI_CLR (i=0-3)	Node i message transfer interrupt clear register	00138_H+i*100_H	PNi	PNi	Kernel Reset	4330

(table continues...)

Table 1075 (continued) Registers overview - CANXL0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CANXL0_NODEi_MTI_ENA (i=0-3)	Node i message transfer interrupt enable register	$0013C_H$+i*100_H	PNi	PNi	Kernel Reset	4331
CANXL0_DEBUG_CTL	Debug control register	10004_H	PM	PM	Kernel Reset	4332
CANXL0_Ni_TS_CTL (i=0-3)	Node i timestamp control	20010_H+i*2000_H	PNi	PNi	Kernel Reset	4332
CANXL0_Ni_TS_CLOCK_CTL (i=0-3)	Node i timestamp clock control	20014_H+i*2000_H	PNi	PNi	Kernel Reset	4333
CANXL0_Ni_TS_CMD (i=0-3)	Node i timestamp command	20020_H+i*2000_H	PNi	PNi	Kernel Reset	4334
CANXL0_Ni_TS_CNT_LO (i=0-3)	Node i timestamp counter LSBs	20030_H+i*2000_H	PNi	BE	Kernel Reset	4334
CANXL0_Ni_TS_CNT_HI (i=0-3)	Node i timestamp counter MSBs	20034_H+i*2000_H	PNi	BE	Kernel Reset	4335
CANXL0_Ni_VERSION (i=0-3)	Node i release identification register	21000_H+i*2000_H	PNi, 32	BE	Kernel Reset	4335
CANXL0_Ni_MH_CTRL (i=0-3)	Node i message handler control register	21004_H+i*2000_H	PNi, 32	PNi, 32	Kernel Reset	4336
CANXL0_Ni_MH_CFG (i=0-3)	Node i message handler configuration register	21008_H+i*2000_H	PNi, 32	PNi, 32	Kernel Reset	4337
CANXL0_Ni_MH_STS (i=0-3)	Node i message handler status register	$2100C_H$+i*2000_H	PNi, 32	BE	Kernel Reset	4338
CANXL0_Ni_MH_SFTY_CFG (i=0-3)	Node i message handler safety configuration register	21010_H+i*2000_H	PNi, 32	PNi, 32	Kernel Reset	4339
CANXL0_Ni_MH_SFTY_CTRL (i=0-3)	Node i message handler safety control register	21014_H+i*2000_H	PNi, 32	PNi, 32	Kernel Reset	4340
CANXL0_Ni_RX_FILTER_MEM_ADD (i=0-3)	Node i RX filter base address register	21018_H+i*2000_H	PNi, 32	PNi, 32	Kernel Reset	4341

(table continues...)

Table 1075 (continued) Registers overview - CANXL0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CANXL0_Ni_TX_DESC_MEM_ADD (i=0-3)	Node i TX descriptor base address register	$2101C_H$+i*2000_H	PNi, 32	PNi, 32	Kernel Reset	4342
CANXL0_Ni_AXI_ADD_EXT (i=0-3)	Node i AXI address extension register	21020_H+i*2000_H	PNi, 32	PNi, 32	Kernel Reset	4342
CANXL0_Ni_AXI_PARAMS (i=0-3)	Node i AXI parameter register	21024_H+i*2000_H	PNi, 32	PNi, 32	Kernel Reset	4343
CANXL0_Ni_MH_LOCK (i=0-3)	Node i message handler lock register	21028_H+i*2000_H	PNi, 32	PNi, 32	Kernel Reset	4343
CANXL0_Ni_TX_DESC_ADD_PT (i=0-3)	Node i TX descriptor current address pointer register	21100_H+i*2000_H	PNi, 32	BE	Kernel Reset	4344
CANXL0_Ni_TX_STATISTICS (i=0-3)	Node i TX message counter register	21104_H+i*2000_H	PNi, 32	PNi, 32	Kernel Reset	4344
CANXL0_Ni_TX_FQ_STS0 (i=0-3)	Node i TX FIFO queue status register	21108_H+i*2000_H	PNi, 32	BE	Kernel Reset	4345
CANXL0_Ni_TX_FQ_STS1 (i=0-3)	Node i TX FIFO queue status register	$2110C_H$+i*2000_H	PNi, 32	BE	Kernel Reset	4346
CANXL0_Ni_TX_FQ_CTRL0 (i=0-3)	Node i TX FIFO queue control register 0	21110_H+i*2000_H	PNi, 32	PNi, 32	Kernel Reset	4346
CANXL0_Ni_TX_FQ_CTRL1 (i=0-3)	Node i TX FIFO queue control register 1	21114_H+i*2000_H	PNi, 32	PNi, 32	Kernel Reset	4347
CANXL0_Ni_TX_FQ_CTRL2 (i=0-3)	Node i TX FIFO queue control register 2	21118_H+i*2000_H	PNi, 32	PNi, 32	Kernel Reset	4348
CANXL0_Ni_TX_FQ_ADD_PTr (i=0-3;r=0-7)	Node i TX FIFO queue r current address pointer register	21120_H+i*2000_H+r*10_H	PNi, 32	BE	Kernel Reset	4348
CANXL0_Ni_TX_FQ_START_ADDr (i=0-3;r=0-7)	Node i TX FIFO queue r start address register	21124_H+i*2000_H+r*10_H	PNi, 32	PNi, 32	Kernel Reset	4349
CANXL0_Ni_TX_FQ_SIZEr (i=0-3;r=0-7)	Node i TX FIFO queue r size register	21128_H+i*2000_H+r*10_H	PNi, 32	PNi, 32	Kernel Reset	4349

(table continues...)

Table 1075 (continued) Registers overview - CANXL0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CANXL0_Ni_TX_PQ_STS0 (i=0-3)	Node i TX priority queue status register	21300_H+i*2000_H	PNi, 32	BE	Kernel Reset	4350
CANXL0_Ni_TX_PQ_STS1 (i=0-3)	Node i TX priority queue status register	21304_H+i*2000_H	PNi, 32	BE	Kernel Reset	4350
CANXL0_Ni_TX_PQ_CTRL0 (i=0-3)	Node i TX priority queue control register 0	$2130C_H$+i*2000_H	PNi, 32	PNi, 32	Kernel Reset	4351
CANXL0_Ni_TX_PQ_CTRL1 (i=0-3)	Node i TX priority queue control register 1	21310_H+i*2000_H	PNi, 32	PNi, 32	Kernel Reset	4351
CANXL0_Ni_TX_PQ_CTRL2 (i=0-3)	Node i TX priority queue control register 2	21314_H+i*2000_H	PNi, 32	PNi, 32	Kernel Reset	4352
CANXL0_Ni_TX_PQ_START_ADD (i=0-3)	Node i TX priority queue start address	21318_H+i*2000_H	PNi, 32	PNi, 32	Kernel Reset	4352
CANXL0_Ni_RX_DESC_ADD_PT (i=0-3)	Node i RX descriptor current address pointer	21400_H+i*2000_H	PNi, 32	BE	Kernel Reset	4353
CANXL0_Ni_RX_STATISTICS (i=0-3)	Node i RX message counter register	21404_H+i*2000_H	PNi, 32	PNi, 32	Kernel Reset	4353
CANXL0_Ni_RX_FQ_STS0 (i=0-3)	Node i RX FIFO queue status register 0	21408_H+i*2000_H	PNi, 32	BE	Kernel Reset	4354
CANXL0_Ni_RX_FQ_STS1 (i=0-3)	Node i RX FIFO queue status register 1	$2140C_H$+i*2000_H	PNi, 32	BE	Kernel Reset	4355
CANXL0_Ni_RX_FQ_STS2 (i=0-3)	Node i RX FIFO queue status register 2	21410_H+i*2000_H	PNi, 32	BE	Kernel Reset	4355
CANXL0_Ni_RX_FQ_CTRL0 (i=0-3)	Node i RX FIFO queue control register 0	21414_H+i*2000_H	PNi, 32	PNi, 32	Kernel Reset	4356
CANXL0_Ni_RX_FQ_CTRL1 (i=0-3)	Node i RX FIFO queue control register 1	21418_H+i*2000_H	PNi, 32	PNi, 32	Kernel Reset	4356
CANXL0_Ni_RX_FQ_CTRL2 (i=0-3)	Node i RX FIFO queue control register 2	$2141C_H$+i*2000_H	PNi, 32	PNi, 32	Kernel Reset	4357

(table continues...)

Table 1075 (continued) Registers overview - CANXL0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CANXL0_Ni_RX_FQ_ADD_PTs (i=0-3;s=0-7)	Node i RX FIFO queue s current address pointer	21420_H+i*2000_H+s*18_H	PNi, 32	BE	Kernel Reset	4357
CANXL0_Ni_RX_FQ_START_ADDs (i=0-3;s=0-7)	Node i RX FIFO queue s link list start address	21424_H+i*2000_H+s*18_H	PNi, 32	PNi, 32	Kernel Reset	4358
CANXL0_Ni_RX_FQ_SIZEs (i=0-3;s=0-7)	Node i RX FIFO queue s link list and data container size	21428_H+i*2000_H+s*18_H	PNi, 32	PNi, 32	Kernel Reset	4358
CANXL0_Ni_RX_FQ_DC_START_ADDs (i=0-3;s=0-7)	Node i RX FIFO queue s data container start address	$2142C_H$+i*2000_H+s*18_H	PNi, 32	PNi, 32	Kernel Reset	4359
CANXL0_Ni_RX_FQ_RD_ADD_PTs (i=0-3;s=0-7)	Node i RX FIFO queue s read address pointer	21430_H+i*2000_H+s*18_H	PNi, 32	PNi, 32	Kernel Reset	4360
CANXL0_Ni_TX_FILTER_CTRL0 (i=0-3)	Node i TX filter control register 0	21600_H+i*2000_H	PNi, 32	SV, PNi, 32	Kernel Reset	4360
CANXL0_Ni_TX_FILTER_CTRL1 (i=0-3)	Node i TX filter control register 1	21604_H+i*2000_H	PNi, 32	SV, PNi, 32	Kernel Reset	4361
CANXL0_Ni_TX_FILTER_REFVALt (i=0-3;t=0-3)	Node i TX filter reference value register t	21608_H+i*2000_H+t*4	PNi, 32	SV, PNi, 32	Kernel Reset	4362
CANXL0_Ni_RX_FILTER_CTRL (i=0-3)	Node i RX filter control register	21680_H+i*2000_H	PNi, 32	SV, PNi, 32	Kernel Reset	4363
CANXL0_Ni_TX_FQ_INT_STS (i=0-3)	Node i TX FIFO queue interrupt status register	21700_H+i*2000_H	PNi, 32	PNi, 32	Kernel Reset	4364
CANXL0_Ni_RX_FQ_INT_STS (i=0-3)	Node i RX FIFO queue interrupt status register	21704_H+i*2000_H	PNi, 32	PNi, 32	Kernel Reset	4365
CANXL0_Ni_TX_PQ_INT_STS0 (i=0-3)	Node i TX priority queue interrupt status register 0	21708_H+i*2000_H	PNi, 32	PNi, 32	Kernel Reset	4365
CANXL0_Ni_TX_PQ_INT_STS1 (i=0-3)	Node i TX priority queue interrupt status register 1	$2170C_H$+i*2000_H	PNi, 32	PNi, 32	Kernel Reset	4366

(table continues...)

Table 1075 (continued) Registers overview - CANXL0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CANXL0_Ni_STATS_INT_STS (i=0-3)	Node i statistics interrupt status register	21710_H+i*2000_H	PNi, 32	PNi, 32	Kernel Reset	4366
CANXL0_Ni_ERR_INT_STS (i=0-3)	Node i error interrupt status register	21714_H+i*2000_H	PNi, 32	PNi, 32	Kernel Reset	4367
CANXL0_Ni_SFTY_INT_STS (i=0-3)	Node i safety interrupt status register	21718_H+i*2000_H	PNi, 32	PNi, 32	Kernel Reset	4368
CANXL0_Ni_AXI_ERR_INFO (i=0-3)	Node i DMA error information	$2171C_H$+i*2000_H	PNi, 32	BE	Kernel Reset	4369
CANXL0_Ni_DESC_ERR_INFO0 (i=0-3)	Node i descriptor error information 0	21720_H+i*2000_H	PNi, 32	BE	Kernel Reset	4370
CANXL0_Ni_DESC_ERR_INFO1 (i=0-3)	Node i descriptor error information 1	21724_H+i*2000_H	PNi, 32	BE	Kernel Reset	4371
CANXL0_Ni_TX_FILTER_ERR_INFO (i=0-3)	Node i TX filter error information	21728_H+i*2000_H	PNi, 32	BE	Kernel Reset	4371
CANXL0_Ni_DEBUG_TEST_CTRL (i=0-3)	Node i debug control register	21800_H+i*2000_H	PNi, 32	SV, PNi, 32	Kernel Reset	4372
CANXL0_Ni_INT_TEST0 (i=0-3)	Node i interrupt test register 0	21804_H+i*2000_H	PNi, 32	PNi, 32	Kernel Reset	4373
CANXL0_Ni_INT_TEST1 (i=0-3)	Node i interrupt test register 1	21808_H+i*2000_H	PNi, 32	PNi, 32	Kernel Reset	4373
CANXL0_Ni_TX_SCAN_FC (i=0-3)	Node i TX-SCAN first candidates register	21810_H+i*2000_H	PNi, 32	BE	Kernel Reset	4376
CANXL0_Ni_TX_SCAN_BC (i=0-3)	Node i TX-SCAN best candidates register	21814_H+i*2000_H	PNi, 32	BE	Kernel Reset	4377
CANXL0_Ni_TX_FQ_DESC_VALID (i=0-3)	Node i valid TX FIFO queue descriptors in local memory	21818_H+i*2000_H	PNi, 32	BE	Kernel Reset	4378
CANXL0_Ni_TX_PQ_DESC_VALID (i=0-3)	Node i valid TX priority queue descriptors in local memory	$2181C_H$+i*2000_H	PNi, 32	BE	Kernel Reset	4379

(table continues...)

Table 1075 (continued) Registers overview - CANXL0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CANXL0_Ni_CRC_CTRL (i=0-3)	Node i CRC control register	21880_H+i*2000_H	PNi, 32	PNi, 32	Kernel Reset	4379
CANXL0_Ni_CRC_REG (i=0-3)	Node i CRC register	21884_H+i*2000_H	PNi, 32	PNi, 32	Kernel Reset	4380
CANXL0_Ni_ENDN (i=0-3)	Node i endianness test register	21900_H+i*2000_H	PNi, 32	BE	Kernel Reset	4380
CANXL0_Ni_PREL (i=0-3)	Node i PRT release identification register	21904_H+i*2000_H	PNi, 32	BE	Kernel Reset	4380
CANXL0_Ni_STAT (i=0-3)	Node i PRT status register	21908_H+i*2000_H	PNi, 32	BE	Kernel Reset	4381
CANXL0_Ni_EVNT (i=0-3)	Node i event status flags register	21920_H+i*2000_H	PNi, 32	PNi, 32	Kernel Reset	4383
CANXL0_Ni_LOCK (i=0-3)	Node i unlock sequence register	21940_H+i*2000_H	PNi, 32	PNi, 32	Kernel Reset	4384
CANXL0_Ni_CTRL (i=0-3)	Node i control register	21944_H+i*2000_H	PNi, 32	PNi, 32	Kernel Reset	4385
CANXL0_Ni_FIMC (i=0-3)	Node i fault injection module control register	21948_H+i*2000_H	PNi, 32	SV, PNi, 32	Kernel Reset	4386
CANXL0_Ni_TEST (i=0-3)	Node i hardware test functions register	$2194C_H$+i*2000_H	PNi, 32	SV, PNi, 32	Kernel Reset	4386
CANXL0_Ni_MODE (i=0-3)	Node i operating mode register	21960_H+i*2000_H	PNi, 32	PNi, 32	Kernel Reset	4388
CANXL0_Ni_NBTP (i=0-3)	Node i arbitration phase nominal bit timing register	21964_H+i*2000_H	PNi, 32	PNi, 32	Kernel Reset	4389
CANXL0_Ni_DBTP (i=0-3)	Node i CAN FD data phase bit timing register	21968_H+i*2000_H	PNi, 32	PNi, 32	Kernel Reset	4390
CANXL0_Ni_XBTP (i=0-3)	Node i CAN XL data phase bit timing register	$2196C_H$+i*2000_H	PNi, 32	PNi, 32	Kernel Reset	4391

(table continues...)

Table 1075 (continued) Registers overview - CANXL0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CANXL0_Ni_PCFG (i=0-3)	Node i PWME configuration register	21970_H+i*2000_H	PNi, 32	PNi, 32	Kernel Reset	4392
CANXL0_Ni_FUNC_RAW (i=0-3)	Node i functional raw event status register	$21A00_H$+i*2000_H	PNi, 32	BE	Kernel Reset	4393
CANXL0_Ni_ERR_RAW (i=0-3)	Node i error raw event status register	$21A04_H$+i*2000_H	PNi, 32	BE	Kernel Reset	4395
CANXL0_Ni_SAFETY_RAW (i=0-3)	Node i safety raw event status register	$21A08_H$+i*2000_H	PNi, 32	BE	Kernel Reset	4397
CANXL0_Ni_FUNC_CLR (i=0-3)	Node i functional raw event clear register	$21A10_H$+i*2000_H	PNi, 32	PNi, 32	Kernel Reset	4400
CANXL0_Ni_ERR_CLR (i=0-3)	Node i error raw event clear register	$21A14_H$+i*2000_H	PNi, 32	PNi, 32	Kernel Reset	4402
CANXL0_Ni_SAFETY_CLR (i=0-3)	Node i safety raw event clear register	$21A18_H$+i*2000_H	PNi, 32	PNi, 32	Kernel Reset	4403
CANXL0_Ni_FUNC_ENA (i=0-3)	Node i functional raw event enable register	$21A20_H$+i*2000_H	PNi, 32	PNi, 32	Kernel Reset	4405
CANXL0_Ni_ERR_ENA (i=0-3)	Node i error raw event enable register	$21A24_H$+i*2000_H	PNi, 32	PNi, 32	Kernel Reset	4407
CANXL0_Ni_SAFETY_ENA (i=0-3)	Node i safety raw event enable register	$21A28_H$+i*2000_H	PNi, 32	PNi, 32	Kernel Reset	4409
CANXL0_Ni_CAPTURING_MODE (i=0-3)	Node i IRC configuration register	$21A30_H$+i*2000_H	PNi, 32	BE	Kernel Reset	4411
CANXL0_Ni_HDP (i=0-3)	Node i hardware debug port control register	$21A40_H$+i*2000_H	PNi, 32	PNi, 32	Kernel Reset	4411

22.13.4.5 Device specific registers

There are no device specific register changes.

22.13.5 TC4Dx CANXL connectivity

The table below lists interfaces to and from this functional block to other blocks in the device.

Table 1076 List of CANXL interface signals

Interface signals	I/O	Description
CLOCK_CANXL_fCANXL	In	CANXL asynchronous clock input
CLOCK_CANXL_fCANXLH	In	CANXL synchronous clock input
CPU0_CANXL_STM_TRIG	In	CPU0 VM1 STM Service Request 0 trigger input
EGTM_CANXL_TRIG_IN[3:0]	In	eGTM trigger input
PORTS_CANXL_node[3:0]_RXD[7:0]	In	CANXL node i receive inputs
CANXL_PORTS_node[3:0]_TXD	Out	CANXL node i transmit output
CANXL_IR_FUNC_INT[3:0]	Out	CANXL Functional events Service Request
CANXL_IR_ERR_INT[3:0]	Out	CANXL Error events Service Request
CANXL_IR_SAFETY_INT[3:0]	Out	CANXL Safety events Service Request
CANXL_EGTM_MTI_TRIG[3:0]	Out	CANXL Miscellaneous events Service Request to eGTM

22.13.6 TC4Dx CANXL revision history

23 FlexRay™ controller (ERAY)

The ERAY IP-module performs communication according to the FlexRay™ protocol specification v2.1, developed for automotive applications. With maximum specified clock the bit rate can be programmed to values up to 10 Mbit/s. Additional bus driver (BD) hardware is required for connection to the physical layer.

23.1 Feature list

- Conformance with the FlexRay™ protocol specification version 2.1
- Flexible functional configuration and monitoring for software
  - Configuration of message buffers with different payload lengths possible
  - Up to 128 message buffers
  - Each message buffer can be configured as receive buffer, as transmit buffer, or as part of the receive FIFO (first in first out)
  - Flag indicating which input buffer is currently accessible by the host
- Transmit frames according to the protocol
  - Data rates of up to 10 Mbit/s on each channel
  - Filtering for slot counter, cycle counter, and channel
- Receive frames according to the protocol
  - Data rates of up to 10 Mbit/s on each channel
  - Filtering for slot counter, cycle counter, and channel
- Data storage of 8 Kbyte Message RAM available for message buffers and receive FIFO (first in first out)
  - Storage of for example 128 message buffers with a maximum 48 byte data field or up to 30 message buffers with 254 byte data sections
  - One configurable receive FIFO
  - Host access to message buffers through input and output buffer
    - Input buffer holds message to be transferred to the Message RAM
    - Output buffer holds message read from the message RAM
  - Four input buffers for building up transmission frames in parallel
  - Automatically delayed read access to Input Buffer Command Request (IBCR) register if a data transfer from the input shadow buffer to the Message RAM (initiated by a previous write access to the IBCR) is ongoing
  - Automatically delayed read access to Output Buffer Command Request OBCR register if a data transfer from the message RAM to the output shadow buffer (initiated by a previous write access to the OBCR) is ongoing
- Flexible interrupt structure generation available
  - Maskable module service requests
  - Service request lines
- Time management functions
  - Synchronization to the cluster according to the protocol
  - External clock output
- Network Management support

23.2 Functional overview

The figure shows an overview of the ERAY interface.

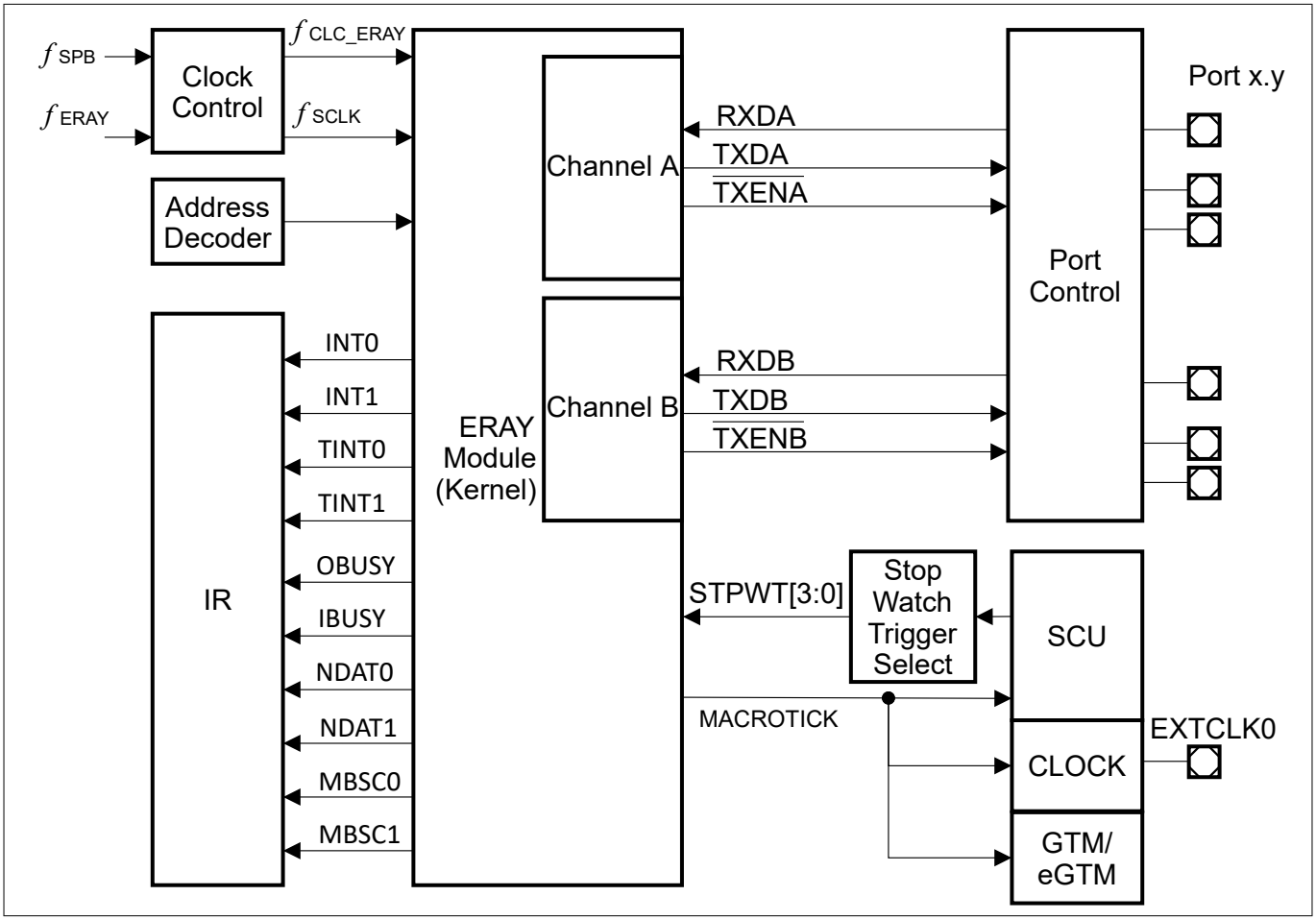


Figure 428 ERAY block diagram

The ERAY module communicates with the external world through three I/O lines on each channel. The RXDAx and RXDBx lines are the receive data input signals, the TXDA and TXDB lines are the transmit output signals, and $\overline{\text{TXENA}}$ and $\overline{\text{TXENB}}$ are the transmit enable signals.

Clock control, address decoding, and service request control are managed outside the ERAY module kernel.

The ERAY is built up of the following main sub-modules:

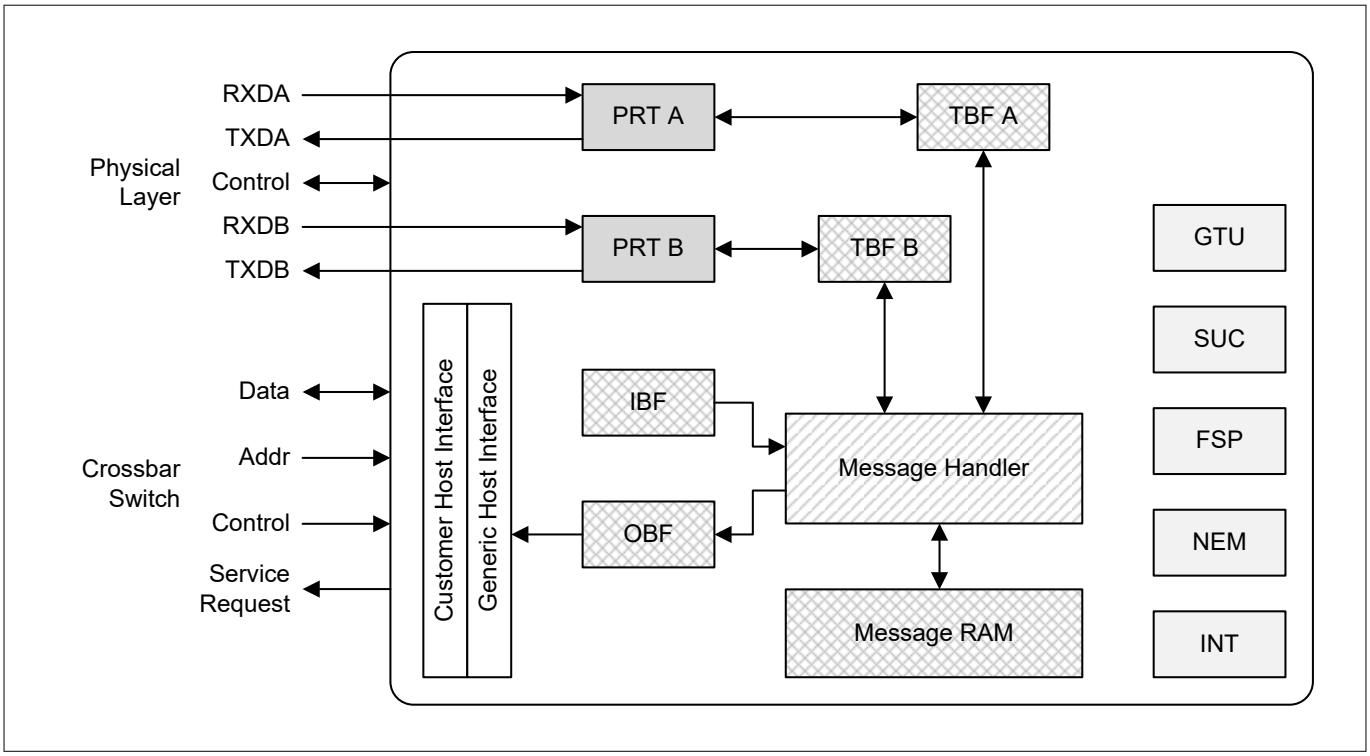


Figure 429 Main sub-modules of ERAY

Customer host interface (CIF)

Connects the FPI bus to the ERAY module through the generic host interface.

Generic host interface (GIF)

The ERAY module is provided with an 8/16/32-bit generic host interface prepared for the connection to a wide range of customer-specific Hosts. Configuration registers, status registers, and service request registers are attached to the respective blocks and can be accessed through the generic host interface.

Input buffer (IBF)

For write access to the message buffers configured in the message RAM, the host can write the header and data section for a specific message buffer to the input buffer. The message handler then transfers the data from the input buffer to the selected message buffer in the message RAM.

There are 4 IBFs. This enables to fill the buffer partly and at the end request transfer into message RAM. Two bits allow to switch between the two banks of IBF and one status bit signals the IBF currently active for host writes.

Output buffer (OBF)

For read access to a Message Buffer configured in the message RAM the message handler transfers the selected message buffer to the output buffer. After the transfer has completed, the host can read the header and data section of the transferred message buffer from the output buffer.

Message handler (MHD)

The ERAY Message Handler controls acceptance filtering, maintaining the transmission schedule, providing message status information and data transfers between the following components:

- Input/output buffer and message RAM
- Transient buffer RAMs of the two FlexRay™ protocol controllers and message RAM

Message RAM (MRAM)

The message RAM consists of a single-ported RAM that stores up to 128 FlexRay™ message buffers together with the related configuration data (header and data partition). For communication on a FlexRay™ network, individual Message Buffers with up to 254 data byte are configurable.

Transient buffer RAM (TBF 1/2)

Stores the data section of two complete messages.

FlexRay™ channel protocol controller (PRT A/B)

The FlexRay™ channel protocol controllers consist of shift register and FlexRay™ protocol FSM. They are connected to the transient buffer RAMs for intermediate message storage and to the physical layer through bus driver (BD).

They perform the following functionality:

- Control and check of bit timing
- Reception and transmission of FlexRay™ frames and symbols
- Check of header CRC
- Generation/check of frame CRC
- Interfacing to bus driver

The FlexRay™ channel protocol controllers have interfaces to:

- Physical layer (bus driver)
- Transient buffer RAM
- Message handler
- Global time unit
- System universal control
- Frame and symbol processing
- Network management
- Service request control

Global time unit (GTU)

The Global time unit performs the following functions:

- Generation of Microtick
- Generation of Macrotick
- Fault tolerant clock synchronization by Fault-Tolerant Midpoint (FTM) algorithm
  - Rate correction
  - Offset correction
- Cycle counter
- Timing control of static segment
- Timing control of dynamic segment (mini-slotting)
- Support of external clock correction

System universal control (SUC)

The System universal control controls the following functions:

- Configuration
- Wakeup
- Startup
- Normal operation

- Passive operation
- Monitor mode

Frame and symbol processing (FSP)

The frame and symbol processing controls the following functions:
- Checks the correct timing of frames and symbols
- Tests the syntactical and semantical correctness of received Frames
- Sets the slot status flags

Network management (NEM)

Handles of the network management vector

Service request control (INT)

The service request controller performs the following functions:
- Provides error and status service request flags
- Enables and disables service request sources
- Assignment of service request sources to one of the two module service request lines
- Enables and disables module service request lines
- Manages the two service request timers
- Stop watch time capturing

23.3 Functional description

This chapter describes the ERAY implementation together with the related FlexRay™ protocol features. More information about the FlexRay™ protocol itself can be found in the FlexRay™ protocol specification v2.1.

Communication on FlexRay™ networks is based on Frames and symbols. The wakeup symbol (WUS) and the collision avoidance symbol (CAS) are transmitted outside the communication cycle to setup the time schedule. Frames and media access test symbols (MTS) are transmitted inside the communication cycle.

23.3.1 Definitions

FlexRay™ Frame: Header Segment + Payload Segment

Message Buffer: Header Section + Data Section

Message RAM: Header Partition + Data Partition

Data Frame: FlexRay™ Frame that is not a NULL Frame

23.3.2 Communication cycle

A communication cycle in FlexRay™ consists of the following elements:

- Static Segment
- Dynamic Segment
- Symbol Window
- Network Idle Time (NIT)

Static segment, dynamic segment, and symbol window form the Network Communication Time (NCT). For each communication channel the slot counter starts at 1 and counts up until the end of the dynamic segment is reached. Both channels share the same arbitration grid which means that they use the same synchronized Macrotick.

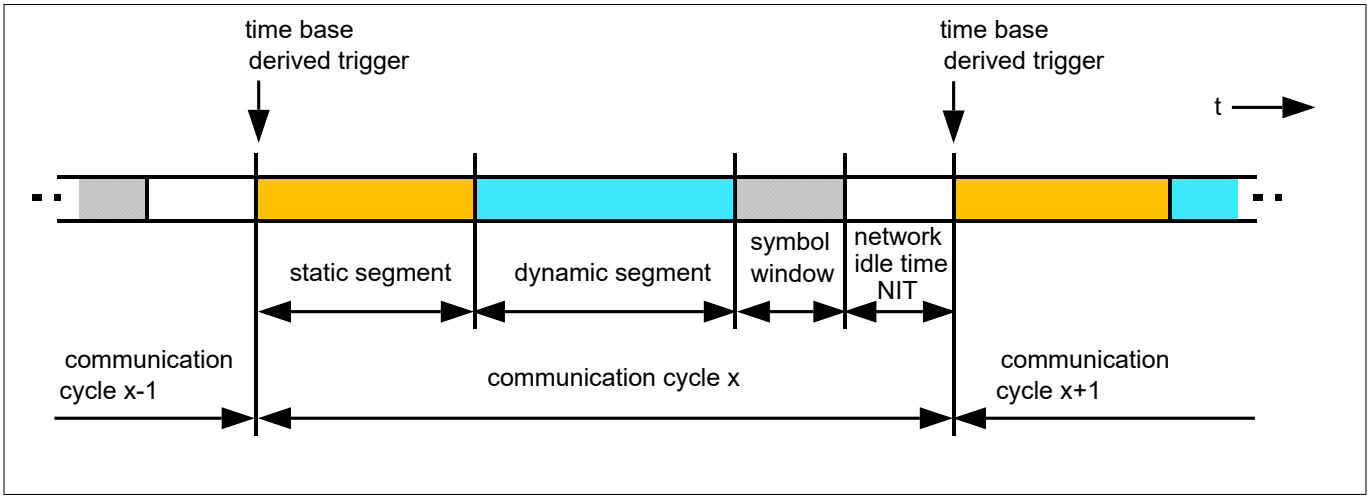


Figure 430 Structure of communication cycle

23.3.2.1 Static segment

The static segment is characterized by the following features:

- Time slots of fixed length (optionally protected by bus guardian)
- Start of Frame transmission at action point of the respective static slot
- Payload length same for all frames on both channel

Parameters: Number of Static Slots GTUC07.NSS, Static Slot Length GTUC07.SSL, Static Frame Data Length MHDC.SFDL, Action Point Offset GTUC09.APO.

23.3.2.2 Dynamic segment

The dynamic segment is characterized by the following features:

- All controllers have bus access (no bus guardian protection possible)
- Variable payload length and duration of slots, different for both channels
- Start of transmission at minislot action point

Parameters: Number of Minislots GTUC08.NMS, Minislot Length GTUC08.MSL Minislot Action Point Offset GTUC09.MAPO, Start of Latest Transmit (last minislot) MHDC.SLT.

23.3.2.3 Symbol window

During the symbol window only one media access test symbol (MTS) may be transmitted per channel. MTS symbols are send in “NORMAL_ACTIVE” state to test the bus guardian.

The symbol window is characterized by the following features:

- Send single symbol
- Transmission of the MTS symbol starts at the symbol windows action point

Parameters: Symbol Window Action Point Offset GTUC09.APO (same as for static slots), Network Idle Time Start GTUC04.NIT.

23.3.2.4 Network idle time (NIT)

During network idle time the communication controller has to perform the following tasks:

- Calculate clock correction terms (offset and rate)
- Distribute offset correction over multiple macroticks
- Perform cluster cycle related tasks

Parameters: Network Idle Time Start GTUC04.NIT, Offset Correction Start GTUC04.OCS.

23.3.2.5 Configuration of communication cycle elements

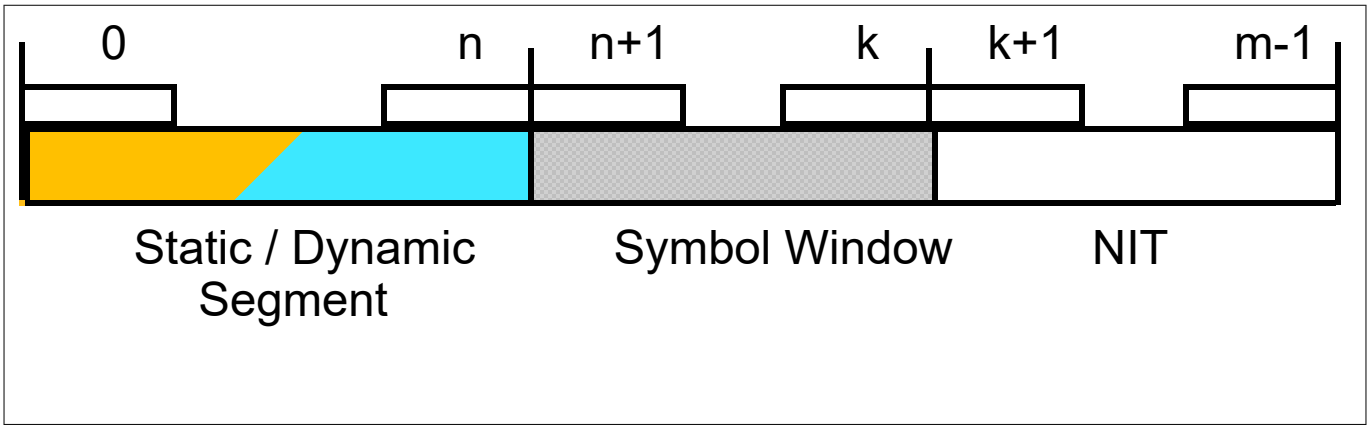


Figure 431 Configuration of network idle time (NIT) start and offset correction start

The number of Macroticks per cycle (gMacroPerCycle) is assumed to be m. It is configured by programming $\mathrm{GTUC02.MPC} = m$.

The static and dynamic segment starts with Macrotick 0 and ends with Macrotick n:

n = static segment length + dynamic segment offset + dynamic segment length − 1 Macrotick, where:

- Static segment length = gNumberOfStaticSlots × gdStaticSlot; the static segment length is configured by GTUC07.NSS and GTUC07.SSL
- Dynamic segment length = gNumberOfMinislots × gdMinislot; the dynamic segment length is configured by GTUC08.NMS and GTUC08.MSL
- The dynamic segment offset is:
  - If gdActionPointOffset ≤ gdMinislotActionPointOffset, dynamic segment offset = 0 Macrotick
  - Else if gdActionPointOffset > gdMinislotActionPointOffset,
    dynamic segment offset = gdActionPointOffset - gdMinislotActionPointOffset

The network idle time (NIT) starts with Macrotick $k + 1$ and ends with the last Macrotick of cycle $m - 1$. It is configured by setting GTUC04.NIT = k.

For ERAY the offset correction start is required to be: GTUC04.OCS ≥ GTU04.NIT + 1 = $k + 1$.

The length of symbol window results from the number of Macroticks between the end of the static and dynamic segment and the beginning of the NIT. It is calculated by: $k - n$.

23.3.3 Communication modes

The FlexRay™ protocol specification v2.1 defines the time-triggered distributed (TT-D) mode.

Time-triggered distributed (TT-D)

In TT-D mode the following configurations are possible:

- Pure static: minimum 2 static slots + symbol window (optional)
- Mixed static/dynamic: minimum 2 static slots + dynamic segment + symbol window (optional)

A minimum of two coldstart nodes need to be configured for distributed time-triggered operation. Two fault-free coldstart nodes are necessary for the cluster startup. Each Startup Frame must be a SYNC Frame, therefore all coldstart nodes are sync nodes.

23.3.4 Clock synchronization

In TT-D mode a distributed clock synchronization is used. Each node individually synchronizes itself to the cluster by observing the timing of received SYNC frames from other nodes.

23.3.4.1 Local time

Internally, nodes time their behavior with Microtick resolution. Microticks are time units derived from the oscillator clock (module clock f_{SCLK}) tick of the specific node. Therefore Microticks are controller-specific units. They may have different duration in different controllers. The precision of a node's local time difference measurements is a Microtick.

Node specific:

- Oscillator clock → prescaler → Microtick
- Microtick = basic unit of time measurement in a Communication Controller, clock correction is done in units of Microticks
- Cycle counter + Macrotick counter = nodes local view of the global time

23.3.4.2 Global time

Activities in a FlexRay™ node, including communication, are based on the concept of a global time, even though each individual node maintains its own view of it. It is the clock synchronization mechanism that differentiates

the FlexRay™ cluster from other node collections with independent clock mechanisms. The global time is a vector of two values; the cycle (cycle counter) and the cycle time (Macrotick counter).

Cluster specific:

- Macrotick = basic unit of time measurement in a FlexRay™ network, a Macrotick consists of an integer number of Microticks
- Cycle length = duration of a communication cycle in units of Macroticks

23.3.4.3 Synchronization process

Clock synchronization is performed by means of SYNC Frames. Only preconfigured nodes (sync nodes) are allowed to send SYNC Frames. In a two-channel cluster a sync node has to send its SYNC Frame on both channels.

For synchronization in FlexRay™ the following constraints have to be considered:

- Maximum one SYNC Frame per node in one communication cycle
- Maximum 15 SYNC Frames per cluster in one communication cycle
- Every node has to use all available SYNC Frames for clock synchronization
- Minimum of two sync nodes required for clock synchronization and start-up

For clock synchronization the time difference between expected and observed arrival time of SYNC Frames received during the static segment, valid on both channels (two-channel cluster), is measured. The calculation of correction terms is done during network idle time (NIT) (offset: every cycle, rate: odd cycle) by using a FTA / FTM algorithm. For details see FlexRay™ protocol specification v2.1, chapter 8.

Offset (phase) correction

- Only deviation values measured and stored in the current cycle used
- For a two channel node the smaller value will be taken
- Calculation during network idle time (NIT) of every communication cycle, value may be negative
- Offset correction value calculated in even cycles used for error checking only
- Checked against limit values (violation: NORMAL_ACTIVE → NORMAL_PASSIVE → HALT)
- Correction value is an integer number of Microticks
- Correction done in odd numbered cycles, distributed over the Macroticks beginning at offset correction start-up to cycle end (end of network idle time (NIT)) to shift nodes next start of cycle (Macroticks lengthened/shortened)

Rate (frequency) correction

- Pairs of deviation values measured and stored in even/odd cycle pair used
- For a two channel node the average of the differences from the two channels is used
- Calculated during network idle time (NIT) of odd numbered cycles, value may be negative
- Cluster drift damping is performed using global damping value
- Checked against limit values
- Correction value is a signed integer number of Microticks
- Distributed over Macroticks comprising the next even/odd cycle pair (Macroticks lengthened/shortened)

SYNC frame transmission

SYNC Frame transmission is only possible from buffer 0 and 1. Message Buffer 1 may be used for SYNC Frame transmission in case that SYNC Frames should have different payloads on the two channels. In this case bit MRC.SPLM has to be programmed to 1.

Message Buffers used for SYNC Frame transmission have to be configured with the key slot ID and can be (re)configured in “DEFAULT_CONFIG” or “CONFIG” state only. For nodes transmitting SYNC Frames SUCC1.TXSY must be set to 1.

23.3.4.4 External clock synchronization

During normal operation, independent clusters can drift significantly. If synchronous operation across independent clusters is desired, external synchronization is necessary; even though the nodes within each cluster are synchronized. This can be accomplished with synchronous application of host-deduced rate and offset correction terms to the clusters.

- External offset/rate correction value is a signed integer
- External offset/rate correction value is added to calculated offset/rate correction value
- Aggregated offset/rate correction term (external + internal) is not checked against configured limits

23.3.5 Error handling

The implemented error handling concept is intended to ensure that in case of a lower layer protocol error in a single node communication between non-affected nodes can be maintained. In some cases, higher layer program command activity is required for the Communication Controller to resume normal operation. A change of the error handling state will set bit EIR.PEMC in the Error Service Request Register and may trigger an service request to the Host if enabled. The actual error mode is signaled by CCEV.ERRM in the Communication Controller Error Vector register.

Table 1077 Error modes of the Protocol Operation Control (POC) (degradation model)

Error mode	Activity
ACTIVE	Full operation, State: “NORMAL_ACTIVE” The Communication Controller is fully synchronized and supports the cluster wide clock synchronization. The host is informed of any error condition(s) or status change by interrupt (if enabled) or by reading the error and status interrupt flags from registers EIR and SIR
PASSIVE	Reduced operation, State: “NORMAL_PASSIVE”, Communication Controller self rescue allowed The Communication Controller stops transmitting Frames and symbols, but received Frames are still processed. Clock synchronization mechanisms are continued based on received Frames. No active contribution to the cluster wide clock synchronization. The host is informed of any error condition(s) or status change by interrupt (if enabled) or by reading the error and status interrupt flags from registers EIR and SIR
COMM_HALT	Operation halted, State: “HALT”, Communication Controller self rescue not allowed The Communication Controller stops Frame and symbol processing, clock synchronization processing, and the Macrotick generation. The host has still access to error and status information by reading the error and status interrupt flags from registers EIR and SIR. The bus drivers are disabled

23.3.5.1 Clock correction failed counter

When the clock correction failed counter reaches the maximum “without clock correction passive” limit defined by SUCC3.WCP, the POC transits from “NORMAL_ACTIVE” to “NORMAL_PASSIVE” state. When it reaches the “maximum without clock correction fatal” limit defined by SUCC3.WCF, it transits from “NORMAL_ACTIVE” or “NORMAL_PASSIVE” to the “HALT” state. Both limits are defined in the SUC configuration register 3.

The clock correction failed counter CCEV.CCFC allows the Host to monitor the duration of the inability of a node to compute clock correction terms after the Communication Controller passed protocol startup phase. It will be incremented by one at the end of any odd numbered communication cycle where either the Missing Offset Correction signal SFS.MOCS or the Missing Rate Correction signal SFS.MRCS flag is set. The two flags are located in the SYNC Frame Status register, while the clock correction failed counter is located in the Communication Controller Error Vector register.

The clock correction failed counter is reset to zero at the end of an odd communication cycle if neither the Missing Offset Correction signal SFS.MOCS nor the Missing Rate Correction signal SFS.MRCS flag is set.

The clock correction failed counter stops incrementing when the “maximum without clock correction fatal” value SUCC3.WCF as defined in the SUC Configuration Register 3 is reached (that is, incrementing the counter at its maximum value will not cause it to “wraparound” back to zero). The clock correction failed counter is initialized to zero when the Communication Controller enters “READY” state or when “NORMAL_ACTIVE” state is entered.

23.3.5.2 Passive to active counter

The passive to active counter controls the transition of the POC from “NORMAL_PASSIVE” to “NORMAL_ACTIVE” state. SUCC1.PTA in the SUC configuration register 1 defines the number of consecutive even/odd cycle pairs that must have valid clock correction terms before the Communication Controller is allowed to transit from “NORMAL_PASSIVE” to “NORMAL_ACTIVE” state. If SUCC1.PTA is reset to zero, the Communication Controller is not allowed to transit from “NORMAL_PASSIVE” to “NORMAL_ACTIVE” state.

23.3.5.3 HALT command

In case the Host wants to stop FlexRay™ communication of the local node it can bring the Communication Controller into “HALT” state by asserting the HALT command. This can be done by writing SUCC1.CMD = 0110_B in the SUC Configuration Register 1. When called in “NORMAL_ACTIVE” or “NORMAL_PASSIVE” state the POC transits to “HALT” state at the end of the current cycle. When called in any other state SUCC1.CMD will be reset to 0000_B = “COMMAND_NOT_ACCEPTED” and bit EIR.CNA in the Error Service Request Register is set to 1. If enabled, an service request to the Host is generated.

23.3.5.4 FREEZE command

In case the Host detects a severe error condition it can bring the Communication Controller into “HALT” state by asserting the FREEZE command. This can be done by writing SUCC1.CMD = 0111_B in the SUC Configuration Register 1. The FREEZE command triggers the entry of the “HALT” state immediately regardless of the actual POC state.

The POC state from which the transition to “HALT” state took place can be read from CCSV.PSL.

23.3.6 Communication controller states

This chapter introduces the states of the communication controller.

23.3.6.1 Communication controller state diagram

State transitions are controlled by externals the application reset or RXDA/B, by the POC state machine, and by the CHI Command Vector SUCC1.CMD located in the SUC configuration register 1.

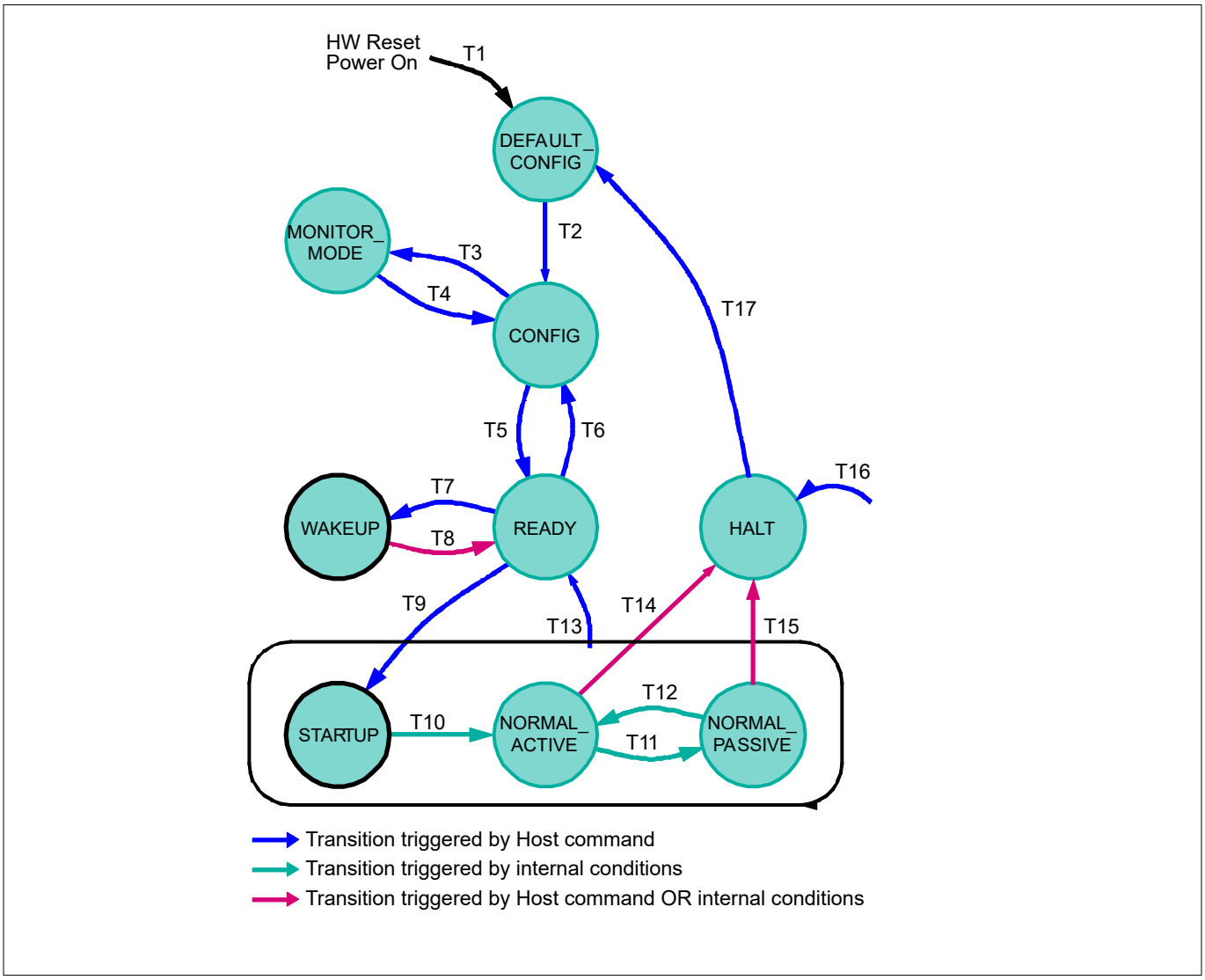


Figure 432 Overall state diagram of ERAY communication controller

The communication controller exits from all states to “HALT” state after application of the FREEZE command (SUCC1.CMD = 0111_B).

Table 1078 State transitions of ERAY overall state machine

T#	Condition	From	To
1	Application reset	Hardware reset	DEFAULT_CONFIG
2	Command CONFIG, SUCC1.CMD = 0001_B	DEFAULT_CONFIG	CONFIG

(table continues...)

Table 1078 (continued) State transitions of ERAY overall state machine

T#	Condition	From	To
3	Unlock sequence followed by command MONITOR_MODE, SUCC1.CMD = 1011_B	CONFIG	MONITOR_MODE
4	Command CONFIG, SUCC1.CMD = 0001_B	MONITOR_MODE	CONFIG
5	Unlock sequence followed by command READY, SUCC1.CMD = 0010_B	CONFIG	READY
6	Command CONFIG, SUCC1.CMD = 0001_B	READY	CONFIG
7	Command WAKEUP, SUCC1.CMD = 0011_B	READY	WAKEUP
8	Complete, non-aborted transmission of wake-up pattern OR received WUP OR received Frame Header OR command READY, SUCC1.CMD = 0010_B	WAKEUP	READY
9	Command RUN SUCC1.CMD = 0100_B	READY	STARTUP
10	Successful start-up	STARTUP	NORMAL_ACTIVE
11	Clock Correction Failed counter reached Maximum Without Clock Correction Passive limit configured by WCP in SUC Configuration Register 3	NORMAL_ACTIVE	NORMAL_PASSIVE
12	Number of valid correction terms reached the Passive to Active limit configured by PTA in SUC Configuration Register 1	NORMAL_PASSIVE	NORMAL_ACTIVE
13	Command READY, SUCC1.CMD = 0010_B	STARTUP, NORMAL_ACTIVE, NORMAL_PASSIVE	READY
14	Clock Correction Failed counter reached Maximum Without Clock Correction Fatal limit configured by WCF in SUC Configuration Register 3 AND bit HCSE in the SUC Configuration Register 1 set to 1 OR command HALT, SUCC1.CMD = 0110_B	NORMAL_ACTIVE	HALT
15	Clock Correction Failed counter reached Maximum Without Clock Correction Fatal limit configured by WCF in SUC Configuration Register 3 AND bit HCSE in the SUC Configuration Register 1 set to 1 OR command HALT, SUCC1.CMD = 0110_B	NORMAL_PASSIVE	HALT
16	Command FREEZE, SUCC1.CMD = 0111_B	All States	HALT
17	Command CONFIG, SUCC1.CMD = 0001_B	HALT	DEFAULT_CONFIG

23.3.6.2 CONFIG state

In “CONFIG” state, the communication controller is stopped. All configuration registers are accessible and the pins to the physical layer are in their inactive state. This state is used to initialize the communication controller configuration.

The communication controller enters this state:

- When exiting from DEFAULT_CONFIG state
- When exiting from MONITOR_MODE or READY state

When the state has been entered through HALT and DEFAULT_CONFIG state, the Host can analyze status information and configuration. Before leaving CONFIG state the Host has to assure that the configuration is fault-free.

To leave “CONFIG” state, the Host has to perform the unlock sequence as described on LCK register section. Directly after unlocking the CONFIG state the Host has to write SUCC1.CMD in the SUC Configuration Register 1 to enter the next state.

Internal counters and the Communication Controller status flags are reset when the Communication Controller leaves CONFIG.

***Note**: The Message Buffer Status Registers (MHDS, TXRQ1 to TXRQ4, NDAT1 to NDAT4, MBSC1 to MBSC4) and status data stored in the Message RAM and are not affected by the transition of the POC from CONFIG to READY state.*

When the Communication Controller is in “CONFIG” state it is also possible to bring the Communication Controller into a power saving mode by halting the module clocks (f_{SCLK}, f_{CLC_ERAY}). To do this the Host has to assure that all Message RAM transfers have finished before turning off the clocks.

DEFAULT_CONFIG state

In DEFAULT_CONFIG state, the communication controller is stopped. All configuration registers are accessible and the pins to the physical layer are in their inactive state.

The communication controller enters this state:

- When leaving application reset
- When exiting from HALT state

To leave DEFAULT_CONFIG state the Host has to write SUCC1.CMD = 0001_B in the SUC Configuration Register 1. The Communication Controller transits to CONFIG state.

23.3.6.3 MONITOR_MODE

After unlocking “CONFIG” state and writing SUCC1.CMD = 1011_B the Communication Controller enters “MONITOR_MODE”. In this mode the Communication Controller is able to receive FlexRay™ Frames and to detect wakeup pattern. The temporal integrity of received Frames is not checked, and therefore cycle counter filtering is not supported. It is not possible to distinguish between static and dynamic frames, because limited functions in “MONITOR_MODE” (FRF.RSS will be ignored, filtering not functional). This mode can be used for debugging purposes in case for example that startup of a FlexRay™ network fails. After writing SUCC1.CMD = 0001_B the Communication Controller transits back to “CONFIG” state.

In “MONITOR_MODE” the “pick first valid” mechanism is disabled. This means that a receive Message Buffer may only be configured to receive on one channel. Received Frames are stored into Message Buffers according to Frame ID and receive channel. NULL Frames are handled like Data Frames. After Frame reception only status bits MBS.VFRA, MBS.MLST, MBS.RCIS, MBS.SFIS, MBS.SYNS, MBS.NFIS, MBS.PPIS, MBS.RESS have valid value.

In “MONITOR_MODE” the Communication Controller is not able to distinguish between CAS and MTS symbols. In case one of these symbols is received on one or both of the two channels, the flags SIR.MTSA respectively SIR.MTSB are set. SIR.CAS has no function in “MONITOR_MODE”.

23.3.6.4 READY state

After unlocking “CONFIG” state and writing SUCC1.CMD = 0010_B the Communication Controller enters “READY” state. From this state the Communication Controller can transit to “WAKEUP” state and perform a cluster wakeup or to “STARTUP” state to perform a coldstart or to integrate into a running communication.

The Communication Controller enters this state:

- When exiting from CONFIG, WAKEUP, STARTUP, NORMAL_ACTIVE, or NORMAL_PASSIVE state by writing SUCC1.CMD = 0010_B (READY command)

The Communication Controller exits from this state:

- To CONFIG state by writing SUCC1.CMD = 0001_B (CONFIG command)
- To WAKEUP state by writing SUCC1.CMD = 0011_B (WAKEUP command)
- To STARTUP state by writing SUCC1.CMD = 0100_B (RUN command)

Internal counters and the Communication Controller status flags are reset when the Communication Controller enters “STARTUP” state.

***Note**: Status bits MHDS, registers TXRQ1 to TXRQ4, and status data stored in the Message RAM are not affected by the transition of the POC from “READY” to “STARTUP” state.*

23.3.6.5 WAKEUP state

The description below is intended to help configuring wakeup for the ERAY module. A detailed description of the wakeup procedure together with the respective Specification and Description Language (SDL) diagrams can be found in the FlexRay™protocol specification v2.1 v2.1, section 7.1.

The Communication Controller enters this state:

- When exiting from READY state by writing SUCC1.CMD = 0011_B (WAKEUP command)

The Communication Controller exits from this state to READY state:

- After complete non-aborted transmission of wakeup pattern
- After WUP reception
- After detecting a WUP collision
- After reception of a Frame Header
- By writing SUCC1.CMD = 0010_B (READY command)

The cluster wakeup must precede the communication startup in order to ensure that all mechanisms defined for the startup work properly. The minimum requirement for a cluster wakeup is that all bus drivers are supplied with power. A bus driver has the ability to wake up the other components of its node when it receives a wakeup pattern on its channel. At least one node in the cluster needs an external wakeup source.

The Host completely controls the wakeup procedure. It is informed about the state of the cluster by the bus driver and the Communication Controller and configures bus guardian (if available) and Communication Controller to perform the cluster wakeup. The Communication Controller provides to the Host the ability to transmit a special wakeup pattern on each of its available channels separately. The Communication Controller needs to recognize the wakeup pattern only during “WAKEUP” state.

Wakeup may be performed on only one channel at a time. The Host has to configure the wakeup channel while the Communication Controller is in “CONFIG” state by writing bit SUCC1.WUCS in the SUC Configuration register 1. The Communication Controller ensures that ongoing communication on this channel is not disturbed. The Communication Controller cannot guarantee that all nodes connected to the configured channel awake upon the transmission of the wakeup pattern, since these nodes cannot give feedback until the startup phase. The wakeup procedure enables single-channel devices in a two-channel system to trigger the wakeup, by only transmitting the wakeup pattern on the single channel to which they are connected. Any

coldstart node that deems a system startup necessary will then wake the remaining channel before initiating communication startup.

The wakeup procedure tolerates any number of nodes simultaneously trying to wakeup a single channel and resolves this situation such that only one node transmits the pattern. Additionally, the wakeup pattern is collision resilient, so even in the presence of a fault causing two nodes to simultaneously transmit a wakeup pattern, the resulting collided signal can still wake the other nodes.

After wakeup the Communication Controller returns to “READY” state and signals the change of the wakeup status to the Host by setting bit SIR.WST in the Status Service Request register. The wakeup status vector can be read from the Communication Controller Status Vector register CCSV.WSV. If a valid wakeup pattern was received, also either flag SIR.WUPA or flag SIR.WUPB in the Status Service Request Register is set.

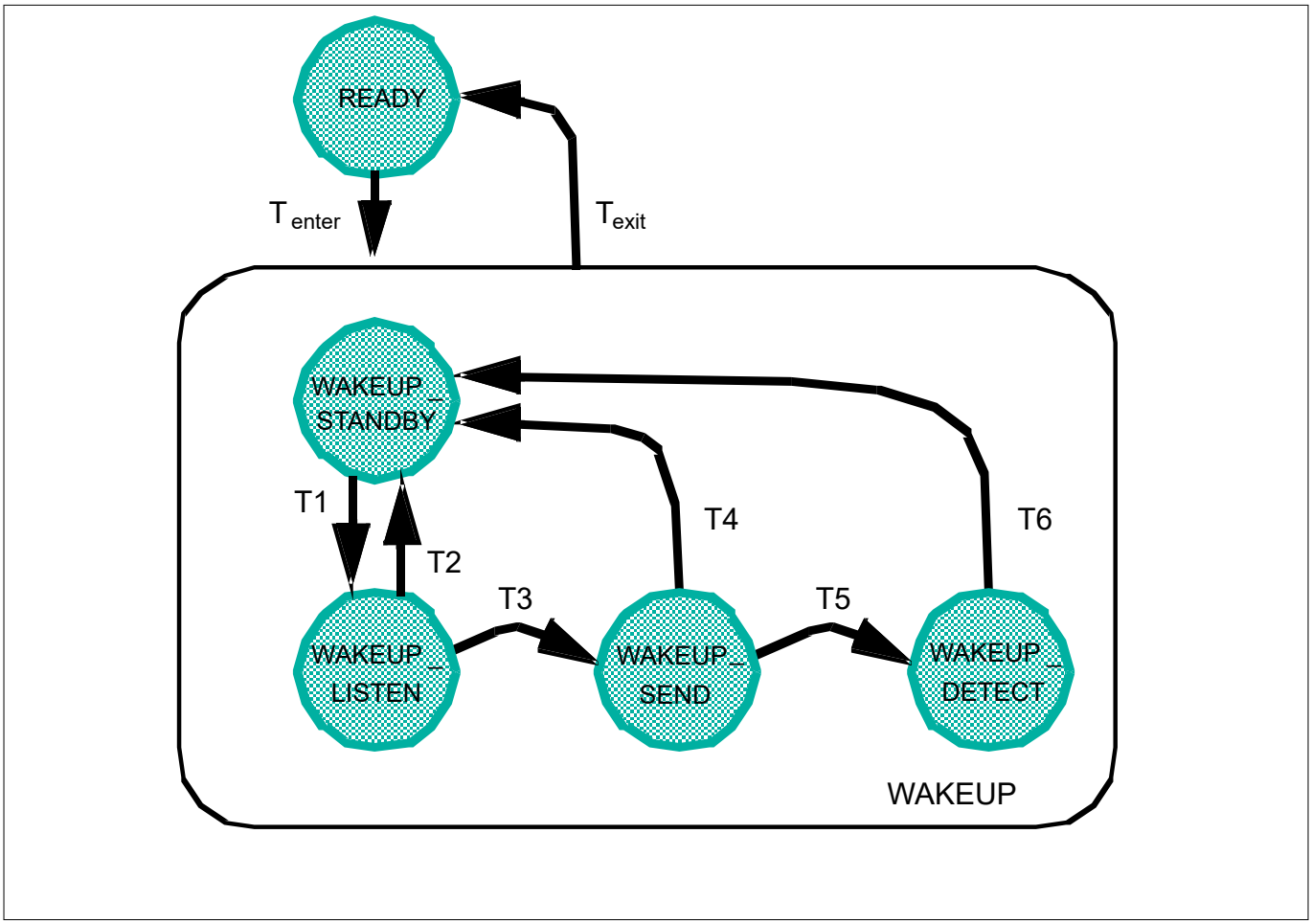


Figure 433 Structure of POC state WAKEUP

Table 1079 State transitions WAKEUP

T#	Condition	From	To
enter	Host commands change to “WAKEUP” state by writing SUCC1.CMD = 0011_B (WAKEUP command)	READY	WAKEUP
1	CHI command WAKEUP triggers wakeup FSM to transit to “WAKEUP_LISTEN” state	WAKEUP_ STANDBY	WAKEUP_LISTEN
2	Received WUP on wakeup channel selected by flag SUCC1.WUCS in the SUC Configuration Register 1 OR Frame Header on either available channel	WAKEUP_ LISTEN	WAKEUP_STAND BY

(table continues...)

Table 1079 (continued) State transitions WAKEUP

T#	Condition	From	To
3	Timer event	WAKEUP_ LISTEN	WAKEUP_SEND
4	Complete, non-aborted transmission of wakeup pattern	WAKEUP_ SEND	WAKEUP_STAND BY
5	Collision detected	WAKEUP_ SEND	WAKEUP_DETEC T
6	Wakeup timer expired OR WUP detected on wakeup channel selected by flag SUCC1.WUCS in the SUC Configuration register 1 OR Frame Header received on either available channel	WAKEUP_ DETECT	WAKEUP_STAND BY
exit	Wakeup completed (after T2 or T4 or T6) OR Host commands change to “READY” state by writing SUCC1.CMD = 0010_B (READY command). This command also resets the wakeup FSM to “WAKEUP_STANDBY” state	WAKEUP	READY

The “WAKEUP_LISTEN” state is controlled by the wakeup timer and the wakeup noise timer. The two timers are controlled by the parameters listen time-out SUCC2.LT and listen time-out noise SUCC2.LTN. Both values can be configured in the SUC Configuration register 2. Listen time-out enables a fast cluster wakeup in case of a noise free environment, while listen time-out noise enables wakeup under more difficult conditions regarding noise interference.

In “WAKEUP_SEND” state the Communication Controller transmits the wakeup pattern on the configured channel and checks for collisions. After return from wakeup the Host has to bring the Communication Controller into “STARTUP” state by CHI command RUN.

In “WAKEUP_DETECT” state the Communication Controller attempts to identify the reason for the wakeup collision detected in “WAKEUP_SEND” state. The monitoring is bounded by the expiration of listen time-out as configured by SUCC2.LT in the SUC Configuration Register 2. Either the detection of a wakeup pattern indicating a wakeup attempt by another node or the reception of a Frame Header indication existing communication, causes the direct transition to “READY” state. Otherwise “WAKEUP_DETECT” is left after expiration of listen time-out; in this case the reason for wakeup collision is unknown.

The Host has to be aware of possible failures of the wakeup and act accordingly. It is advisable to delay any potential startup attempt of the node having instigated the wakeup by the minimal time it takes another coldstart node to become awake and to be configured.

The FlexRay™ Protocol Specification v2.1 recommends that two different Communication Controllers shall awake the two channels.

Host activities

The host must coordinate the wakeup of the two channels and must decide whether or not to wake a specific channel. The sending of the wakeup pattern is initiated by the Host and generated by the Communication Controller. The wakeup pattern is detected by the remote BDs and signalled to their local Hosts.

Wakeup procedure controlled by Host (single-channel wakeup):

- Configure the Communication Controller in CONFIG state
  - Select wakeup channel by programming bit SUCC1.WUCS
- Check local BDs whether a WUP was received
- Activate BD of selected wakeup channel

- Command Communication Controller to start wakeup on the configured channel by writing SUCC1.CMD = 0011_B
  - Communication Controller enters WAKEUP
  - Communication Controller returns to READY state and signals status of wakeup attempt to Host
- Wait predefined time to allow the other nodes to wakeup and configure themselves
- Coldstart node: wait for WUP on the other channel
  - In a dual channel cluster wait for WUP on the other channel
  - Reset coldstart inhibit flag CCSV.CSI by writing SUCC1.CMD = 1001_B (ALLOW_COLDSTART command))
- Command Communication Controller to enter startup by writing SUCC1.CMD = 0100_B (RUN command)

Wakeup procedure triggered by BD:

- Wakeup recognized by BD
- BD triggers power-up of Host (if required)
- BD signals wakeup event to Host
- Host configures its local Communication Controller
- If necessary, Host commands wakeup of second channel and waits predefined time to allow the other nodes to wakeup and configure themselves
- Host commands Communication Controller to enter STARTUP state by writing SUCC1.CMD = 0100_B (RUN command)

Wakeup pattern (WUP)

The wakeup pattern is composed of at least two wakeup symbols (WUS). Wakeup symbol and wakeup pattern are configured by the PRT configuration registers PRTC1 and PRTC2.

- Single channel wakeup, wakeup symbol may not be sent on both channels at the same time
- Wakeup symbol collision resilient for up to two sending nodes (two over-lapping wakeup symbols still recognizable)
- Wakeup symbol must be configured identical in all nodes of a cluster
- Wakeup symbol transmit low phase time configured by PRTC2.TXL
- Wakeup symbol idle time used to listen for activity on the bus, configured by PRTC2.TXI
- A wakeup pattern composed of at least two Tx wakeup symbols needed for wakeup
- Number of repetitions configurable by PRTC1.RWP (2 to 63 repetitions)
- Wakeup symbol receive window length configured by PRTC1.RXW
- Wakeup symbol receive low phase time configured by PRTC2.RXL
- Wakeup symbol receive idle time configured by PRTC2.RXI

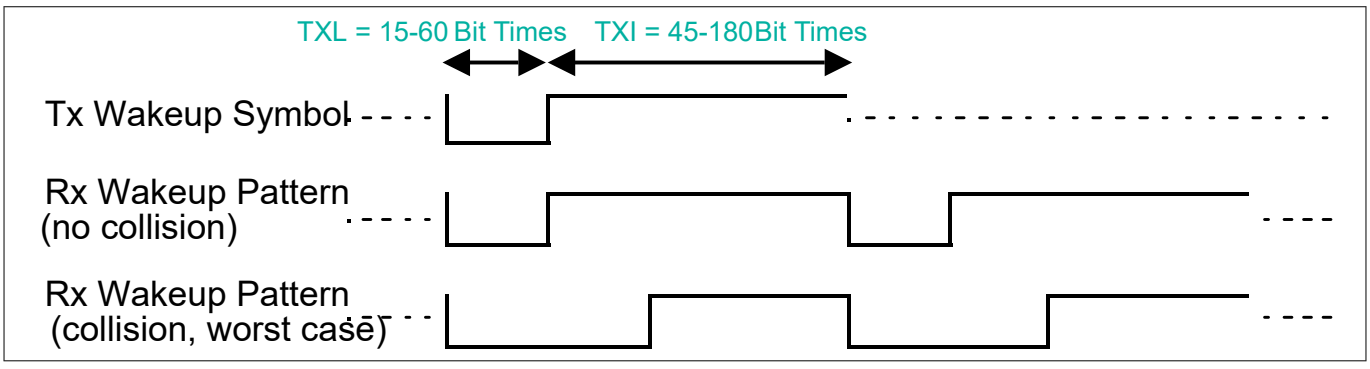


Figure 434 Timing of wakeup pattern

23.3.6.6 STARTUP state

The description here is intended to help configuring startup for the ERAY module. A detailed description of the startup procedure together with the respective SDL diagrams can be found in the FlexRay™ protocol specification v2.1, section 7.2.

Any node entering “STARTUP” state that has coldstart capability should ensure that both channels attached have been woken before initiating coldstart.

It cannot be assumed that all nodes and stars need the same amount of time to become completely awake and to be configured. Since at least two nodes are necessary to start up the cluster communication, it is advisable to delay any potential startup attempt of the node having instigated the wakeup by the minimal amount of time it takes another coldstart node to become awake, to be configured and to enter startup. It may require several hundred milliseconds (depending on the hardware used) before all nodes and stars are completely awakened and configured.

Startup is performed on all channels synchronously. During startup, a node only transmits startup Frames.

A fault-tolerant, distributed startup strategy is specified for initial synchronization of all nodes. In general, a node may enter “NORMAL_ACTIVE” state through:

- Coldstart path initiating the schedule synchronization (leading coldstart node)
- Coldstart path joining other coldstart nodes (following coldstart node)
- Integration path integrating into an existing communication schedule (all other nodes)

See also figure below.

A coldstart attempt begins with the transmission of a collision avoidance symbol (CAS). Only a coldstart node that had transmitted the CAS transmits Frames in the first four cycles after the CAS, it is then joined firstly by the other coldstart nodes and afterwards by all other nodes.

A coldstart node has the Transmit Startup Frame in Key Slot bit SUCC1.TXST and Transmit SYNC Frame in Key Slot bit SUCC1.TXSY set to 1. The Message Buffer 0 holds the key slot ID which defines the slot number where the Startup Frame is send. In the Frame Header of the Startup Frame the Startup Frame indicator bit is set.

In clusters consisting of three or more nodes, at least three nodes shall be configured to be coldstart nodes. In clusters consisting of two nodes, both nodes must be coldstart nodes. At least two fault-free coldstart nodes are necessary for the cluster to startup.

Each Startup Frame must also be a SYNC Frame; therefore each coldstart node will also be a sync node. The number of coldstart attempts is configured by SUCC1.CSA in the SUC Configuration register 1.

A non-coldstart node requires at least two startup Frames from distinct nodes for integration. It may start integration before the coldstart nodes have finished their startup. It will not finish its startup until at least two coldstart nodes have finished their startup.

Both non-coldstart nodes and coldstart nodes start passive integration through the integration path as soon as they receive SYNC Frames from which to derive the schedule information. During integration the node has to adapt its own clock to the global clock (rate and offset) and has to make its cycle time consistent with the global schedule observable at the network. Afterwards, these settings are checked for consistency with all available network nodes. The node can only leave the integration phase and actively participate in communication when these checks are passed.

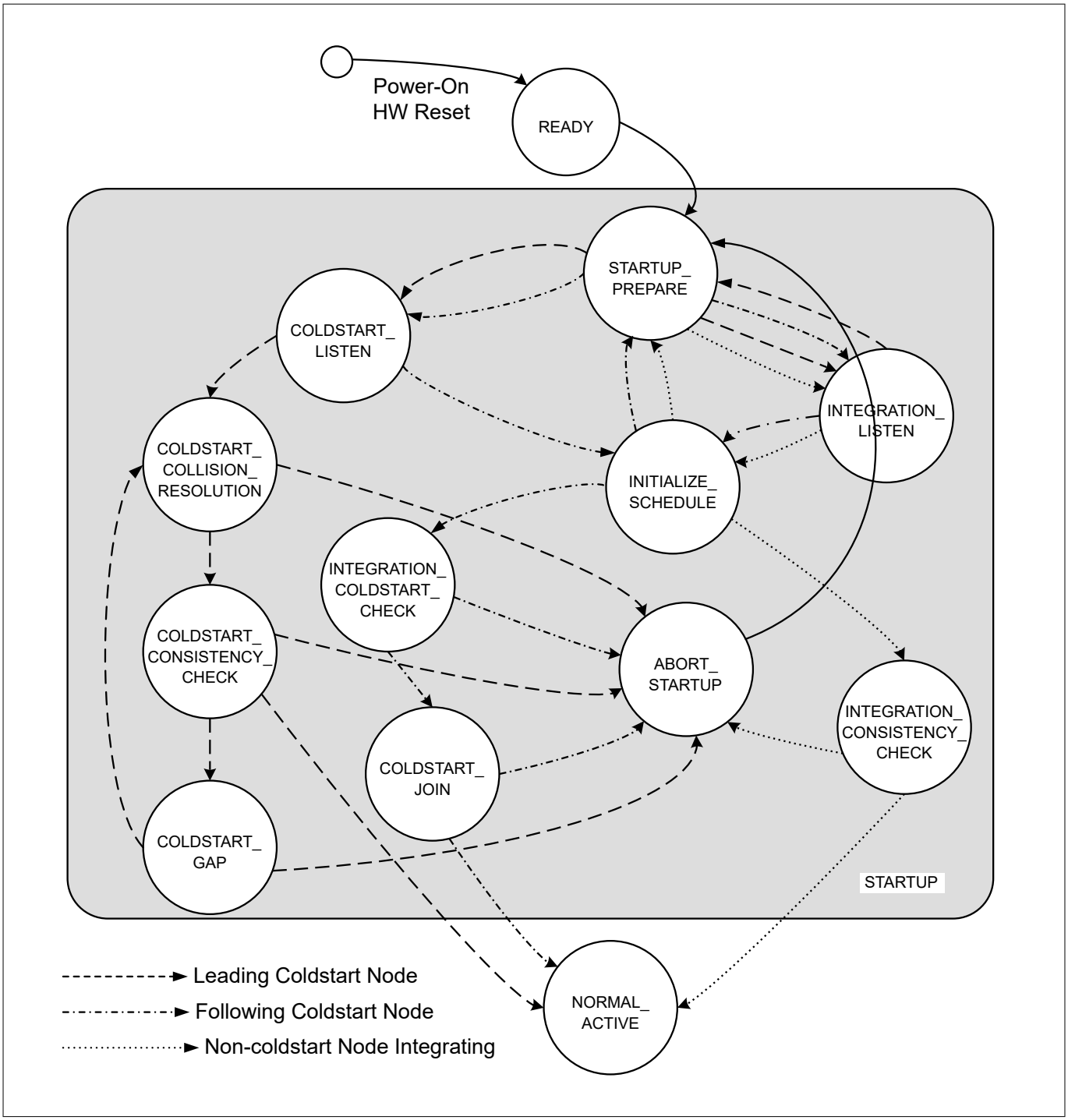


Figure 435 State diagram time-triggered startup

Coldstart inhibit mode

In coldstart inhibit mode the node is prevented from initializing the TDMA communication schedule. If bit CCSV.CSI in the Communication Controller Status Vector register is set, the node is not allowed to initialize the cluster communication, which means that entering the coldstart path is prohibited. The node is allowed to integrate to a running cluster or to transmit startup Frames after another coldstart node started the initialization of the cluster communication.

The coldstart inhibit bit CCSV.CSI is set whenever the POC enters “READY” state. The bit has to be cleared under control of the Host by CHI command ALLOW_COLDSTART (SUCC1.CMD = 1001_B)

23.3.6.6.1 Path of leading coldstart node (initiating coldstart)

When a coldstart node enters “COLDSTART_LISTEN”, it listens to its attached channels.

If no communication is detected, the node enters the “COLDSTART_COLLISION_ RESOLUTION” state and commences a coldstart attempt. The initial transmission of a CAS symbol is succeeded by the first regular cycle. This cycle has the number zero.

From cycle zero on, the node transmits its startup Frame. Since each coldstart node is allowed to perform a coldstart attempt, it may occur that several nodes simultaneously transmit the CAS symbol and enter the coldstart path. This situation is resolved during the first four cycles after CAS transmission.

As soon as a node that initiates a coldstart attempt receives a CAS symbol or a Frame Header during these four cycles, it re-enters the “COLDSTART_LISTEN” state. Thereby, only one node remains in this path. In cycle four, other coldstart nodes begin to transmit their startup Frames.

After four cycles in “COLDSTART_COLLISION_RESOLUTION” state, the node that initiated the coldstart enters the “COLDSTART_CONSISTENCY_CHECK” state. It collects all startup Frames from cycle four and five and performs the clock correction. If the clock correction does not deliver any errors and it has received at least one valid Startup Frame pair, the node leaves “COLDSTART_CONSISTENCY_CHECK” and enters “NORMAL_ACTIVE” state.

The number of coldstart attempts that a node is allowed to perform is configured by SUCC1.CSA in the SUC Configuration Register 1. The number of remaining coldstarts attempts CCSV.RCA can be read from Communication Controller Status Vector register. The number of remaining attempts is reduced by one for each attempted coldstart. A node may enter the “COLDSTART_LISTEN” state only if this value is larger than one and it may enter the “COLDSTART_COLLISION_RESOLUTION” state only if this value is larger than zero. If the number of coldstart attempts is one, coldstart is inhibited but integration is still possible.

Path of following coldstart node (responding to leading coldstart node)

When a coldstart node enters the “COLDSTART_LISTEN” state, it tries to receive a valid pair of startup Frames to derive its schedule and clock correction from the leading coldstart node.

As soon as a valid Startup Frame has been received the “INITIALIZE_SCHEDULE” state is entered. If the clock synchronization can successfully receive a matching second valid Startup Frame and can derive a schedule from this startup Frames, the “INTEGRATION_COLDSTART_CHECK” state is entered.

In “INTEGRATION_COLDSTART_CHECK” state it is assured that the clock correction can be performed correctly and that the coldstart node from which this node has initialized its schedule is still available. The node collects all SYNC Frames and performs clock correction in the following double-cycle. If clock correction does not signal any errors and if the node continues to receive sufficient Frames from the same node it has integrated on, the “COLDSTART_JOIN” state is entered.

In “COLDSTART_JOIN” state integrating coldstart nodes begin to transmit their own startup Frames. Thereby the node that initiated the coldstart and the nodes joining it can check if their schedules agree to each other. If for the following three cycles the clock correction does not signal errors and at least one other coldstart node is visible, the node leaves “COLDSTART_JOIN” state and enters “NORMAL_ACTIVE” state. Thereby it leaves “STARTUP” at least one cycle after the node that initiated the coldstart.

Path of non-coldstart node

When a non-coldstart node enters the “INTEGRATION_LISTEN” state, it listens to its attached channels and tries to receive FlexRay™ Frames.

As soon as a valid Startup Frame has been received, the “INITIALIZE_SCHEDULE” state is entered. If the clock synchronization can successfully receive a matching second valid Startup Frame and derive a schedule from this, the “INTEGRATION_CONSISTENCY_ CHECK” state is entered.

In “INTEGRATION_CONSISTENCY_CHECK” state it is verified that the clock correction can be performed correctly and that enough coldstart nodes (at least 2) send startup Frames that agree to the nodes own schedule. Clock correction is activated, and if any errors are signalled, the integration attempt is aborted.

During the first even cycle in this state, either two valid startup Frames or the Startup Frame of the node that this node has integrated on must be received; otherwise the node aborts the integration attempt.

During the first double-cycle in this state, either two valid Startup Frame pairs or the Startup Frame pair of the node that this node has integrated on must be received; otherwise the node aborts the integration attempt.

If after the first double-cycle less than two valid startup Frames are received within an even cycle, or less than two valid Startup Frame pairs are received within a double-cycle, the startup attempt is aborted.

Nodes in this state need to see two valid Startup Frame pairs for two consecutive double-cycles each to be allowed to leave “STARTUP” and enter “NORMAL_ACTIVE”. Consequently, they leave startup at least one double-cycle after the node that initiated the coldstart and only at the end of a cycle with an odd cycle number.

23.3.6.7 NORMAL_ACTIVE state

As soon as the node that transmitted the first CAS symbol (resolving the potential access conflict and entering STARTUP through coldstart path) and one additional node have entered the NORMAL_ACTIVE state, the startup phase for the cluster has finished. In the NORMAL_ACTIVE state, all configured messages are scheduled for transmission. This includes all Data Frames as well as the SYNC Frames. Rate and offset measurement is started in all even cycles (even/odd cycle pairs required).

In “NORMAL_ACTIVE” state the Communication Controller supports regular communication functions

- The Communication Controller performs transmissions and reception on the FlexRay™ bus as configured
- Clock synchronization is running
- The Host interface is operational

The Communication Controller exits from that state to

- HALT state by writing SUCC1.CMD = 0110_B (HALT command, at the end of the current cycle)
- HALT state by writing SUCC1.CMD = 0111_B (FREEZE command, immediately)
- HALT state due to change of the error state from ACTIVE to COMM_HALT
- NORMAL_PASSIVE state due to change of the error state from ACTIVE to PASSIVE
- READY state by writing SUCC1.CMD = 0010_B (READY command)

23.3.6.8 NORMAL_PASSIVE state

NORMAL_PASSIVE state is entered from NORMAL_ACTIVE state when the error state changes from ACTIVE (green) to PASSIVE (yellow).

In NORMAL_PASSIVE state, the node is able to receive all Frames (node is fully synchronized and performs clock synchronization). In comparison to the NORMAL_ACTIVE state the node does not actively participate in communication, that is neither symbols nor Frames are transmitted.

In NORMAL_PASSIVE state:

- The Communication Controller performs reception on the FlexRay™ bus
- The Communication Controller does not transmit any Frames or symbols on the FlexRay™ bus
- Clock synchronization is running
- The Host interface is operational

The Communication Controller exits from this state to

- HALT state by writing SUCC1.CMD = 0110_B (HALT command, at the end of the current cycle)
- HALT state by writing SUCC1.CMD = 0111_B (FREEZE command, immediately)
- HALT state due to change of the error state from PASSIVE to COMM_HALT
- NORMAL_ACTIVE state due to change of the error state from PASSIVE to ACTIVE. The transition takes place when CCEV.PTAC from the Communication Controller Error Vector register equals SUCC1.PTA - 1
- READY state by writing SUCC1.CMD = 0010_B (READY command)

23.3.6.9 HALT state

In this state all communication (reception and transmission) is stopped.

The Communication Controller enters this state:

- By writing SUCC1.CMD = 0110_B (HALT command) while the Communication Controller is in NORMAL_ACTIVE or NORMAL_PASSIVE state
- By writing SUCC1.CMD = 0111_B (FREEZE command) from all states
- When exiting from NORMAL_ACTIVE state because the clock correction failed counter reached the “maximum without clock correction fatal” limit
- When exiting from NORMAL_PASSIVE state because the clock correction failed counter reached the “maximum without clock correction fatal” limit

The Communication Controller exits from this state to DEFAULT_CONFIG state:

- By writing SUCC1.CMD = 0001_B (CONFIG command)

When the Communication Controller enters HALT state, all configuration and status data is maintained for analyzing purposes.

When the Host writes SUCC1.CMD = 0110_B (HALT command) in the SUC Configuration Register 1, the Communication Controller sets bit CCSV.HRQ in the Communication Controller Status Vector register and enters “HALT” state after the current communication cycle has finished.

When the Host writes SUCC1.CMD = 0111_B (FREEZE command) in the SUC Configuration Register 1, the Communication Controller enters “HALT” state immediately and sets the CCSV.FSI bit in the Communication Controller Status Vector register.

The POC state from which the transition to “HALT” state took place can be read from CCSV.PSL.

23.3.6.10 Startup time-outs

The Communication Controller supplies two different Microtick timers supporting two time-out values, startup time-out and startup noise time-out. The two timers are reset when the Communication Controller enters the COLDSTART_LISTEN state. The expiration of either of these timers causes the node to leave the initial sensing phase (COLDSTART_LISTEN state) with the intention of starting up communication.

***Note**: The startup and startup noise timers are identical with the wakeup and wakeup noise timers and use the same configuration values SUCC2.LT and SUCC2.LTN from the SUC Configuration Register 2.*

Startup time-out

The startup time-out limits the listen time used by a node to determine if there is already communication between other nodes or at least one coldstart node actively requesting the integration of others.

The startup timer is configured by programming SUCC2.LT (pdListenTimeout) in the SUC Configuration Register 2.

The startup timer is restarted upon:

- Entering the COLDSTART_LISTEN state
- Both channels reaching idle state while in COLDSTART_LISTEN state

The startup timer is stopped:

- If communication channel activity is detected on one of the configured channels while the node is in the COLDSTART_LISTEN state
- When the COLDSTART_LISTEN state is left

Once the startup time-out expires, neither an overflow nor a cyclic restart of the timer is performed. The timer status is kept for further processing by the startup state machine.

Startup noise time-out

At the same time the startup timer is started for the first time (transition from STARTUP_PREPARE state to COLDSTART_LISTEN state), the startup noise timer is started. This additional time-out is used to improve reliability of the startup procedure in the presence of noise.

The startup noise timer is configured by programming SUCC2.LTN ($\mathrm{gListenNoise} - 1$) in the SUC Configuration Register 2.

The startup noise time-out is: $\mathrm{pdListenTimeout} \times \mathrm{gListenNoise} = \mathrm{SUCC2.LT} \times (\mathrm{SUCC2.LTN} + 1)$

The startup noise timer is restarted upon:

- Entering the COLDSTART_LISTEN state
- Reception of correctly decoded Headers or CAS symbols while the node is in COLDSTART_LISTEN state

The startup noise timer is stopped when the COLDSTART_LISTEN state is left.

Once the startup noise time-out expires, neither an overflow nor a cyclic restart of the timer is performed. The status is kept for further processing by the startup state machine. Since the startup noise timer won't be restarted when random channel activity is sensed, this time-out defines the fall-back solution that guarantees that a node will try to start up the communication cluster even in the presence of noise.

23.3.7 Network management

The accrued Network Management (NM) vector is located in the Network Management Register 1 to Network Management Register 3 ($\mathrm{NMVx}\ (x = 1 - 3)$). The Communication Controller performs a logical OR operation over all Network Management (NM) vectors out of all received valid Network Management (NM) Frames with the Payload Preamble Indicator (PPI) bit set. Only a static Frame may be configured to hold Network Management (NM) information. The Communication Controller updates the Network Management (NM) vector at the end of each cycle.

The length of the Network Management (NM) vector can be configured from 0 to 12 byte by NML in the NEM Configuration Register. The Network Management (NM) vector length must be configured identically in all nodes of a cluster.

To configure a transmit buffer to send FlexRay™ Frames with the PPI bit set, the PPIT bit in the Header Section of the respective transmit buffer has to be set through WRHS1.PPIT. In addition the Host has to write the Network Management (NM) information to the Data Section of the respective transmit buffer.

The evaluation of the Network Management (NM) vector has to be done by the application running on the Host.

***Note**: In case a Message Buffer is configured for transmission or reception of Network Management Frames, the payload length configured in Header 2 of that Message Buffer should be equal or greater than the length of the NM Vector configured by NEMC.NML. When the Communication Controller transits to "HALT" state, the cycle count is not incremented and therefore the NM Vector is not updated. In this case NMV1 to NMV3 holds the value from the cycle before.*

23.3.8 Filtering and masking

Filtering is done by checking specific fields in a received Frame against the corresponding configuration constants of the valid Message Buffers and the actual slot and cycle counter values (acceptance filtering), or by comparing the configuration constants of the valid Message Buffers against the actual slot and cycle counter values (transmit filtering). A Message Buffer is only updated / transmitted if the required matches occur.

Filtering is done on the following fields:

- Channel ID
- Frame ID
- Cycle Counter

The following filter combinations for acceptance/transmit filtering are allowed:

- Frame ID + Channel ID
- Frame ID + Channel ID + Cycle Counter

In order to store a received message in a Message Buffer all configured filters must match.

***Note**: For the FIFO the acceptance filter is configured by the FIFO Rejection Filter and the FIFO Rejection Filter Mask.*

A message will be transmitted in the time slot corresponding to the configured Frame ID on the configured channel(s). If cycle counter filtering is enabled the configured cycle filter value must also match.

23.3.8.1 Frame ID filtering

Every transmit and receive buffer contains a Frame ID stored in the Header Section. This Frame ID is used differently for receive and transmit buffers.

Receive buffers

A received message is stored in the first receive buffer where the received Frame ID matches the configured Frame ID, provided channel ID and cycle counter criteria are also met.

Transmit buffers

For transmit buffers the configured Frame ID is used to determine the appropriate slot for message transmission. The Frame will be transmitted in the time slot corresponding to the configured Frame ID, provided channel ID and cycle counter criteria are also met.

23.3.8.2 Channel ID filtering

There is a 2-bit channel filtering field (CHA, CHB) located in the Header Section of each Message Buffer in the Message RAM. It serves as a filter for receive buffers, and as a control field for transmit buffers (see Table below).

Table 1080 Channel filtering configuration

CHA	CHB	Transmit buffer transmit frame	Receive buffer store valid receive frame
1	1	on both channels (static segment only)	received on channel A or B (store first semantically valid Frame, static segment only)
1	0	on channel A	received on channel A
0	1	on channel B	received on channel B
0	0	no transmission	ignore Frame

***Note**: If a Message Buffer is configured for the dynamic segment and both bits of the channel filtering field are set to 1, no Frames are transmitted respectively received Frames are ignored (same function as CHA = CHB = 0).*

Receive buffers

Valid received Frames are stored if they are received on the channels specified in the channel filtering field. Only in static segment a receive buffer may be setup for reception on both channels (CHA and CHB set). Other filtering criteria must also be met.

If a valid Header Segment was stored, the respective MBC flag in the Message Buffer Status Changed register is set. If a valid Payload Segment was stored, the respective $\mathrm{NDn}\ (n = 0 - 31)$ to $\mathrm{NDn}\ (n = 96 - 127)$ flag in the New Data NDAT1 to NDAT4 register is set. In both cases, if bit RDHS1.MBI in the Header Section of the respective

Message Buffer is set, the RXI flag in the Status Service Request Register is set to 1. If enabled, an service request is generated.

Transmit buffers

The content of the buffer is transmitted only on the channels specified in the channel filtering field when the Frame ID filtering and cycle counter filtering criteria are also met. Only in static segment a transmit buffer may be setup for transmission on both channels (CHA and CHB set). After transmission has completed, and if bit WRHS1.MBI in the Header Section of the respective Message Buffer is set, the TXI flag in the Status Service Request Register is set to 1. If enabled, an service request is generated.

23.3.8.3 Cycle counter filtering

Cycle counter filtering is based on the notion of a cycle set. For filtering purposes, a match is detected if any one of the elements of the cycle set is matched. The cycle set is defined by the cycle code field in the Header Section of each Message Buffer.

If Message Buffer 0 is configured to hold the startup / SYNC Frame or the single slot Frame by bits SUCC1.TXST, SUCC1TXSY and SUCC1.TSM in the SUC Configuration Register 1, cycle counter filtering for Message Buffer 0 should be disabled.

***Note**: Sharing of a static time slot through cycle counter filtering between different nodes of a FlexRay™ network is not allowed.*

The set of cycle numbers belonging to a cycle set is determined as described in Table below.

Table 1081 Definition of cycle set

Cycle code	Matching cycle counter values		
$000000x_B$	All cycles		
$000001c_B$	Every second cycle	at (cycle count)mod2	= c
$00001cc_B$	Every fourth cycle	at (cycle count)mod4	= cc
$0001ccc_B$	Every eighth cycle	at (cycle count)mod8	= ccc
$001cccc_B$	Every sixteenth cycle	at (cycle count)mod16	= cccc
$01ccccc_B$	Every thirty-second cycle	at (cycle count)mod32	= ccccc
$1cccccc_B$	Every sixty-fourth cycle	at (cycle count)mod64	= cccccc

Table below gives some examples for valid cycle sets to be used for cycle counter filtering:

Table 1082 Examples for valid cycle sets

Cycle code	Matching cycle counter values
0000011_B	1-3-5-7- ….-63 ↵
0000100_B	0-4-8-12- ….-60 ↵
0001110_B	6-14-22-30- ….-62 ↵
0011000_B	8-24-40-56 ↵
0100011_B	3-35 ↵
1001001_B	9 ↵

Receive buffers

The received message is stored only if the received cycle counter matches an element of the receive buffer's cycle set. Channel ID and Frame ID criteria must also be met.

Transmit buffers

The content of the buffer is transmitted on the configured channels when an element of the cycle set matches the current cycle counter value and the Frame ID matches the slot counter value.

23.3.8.4 FIFO filtering

For FIFO filtering there is one rejection filter and one rejection filter mask available. The FIFO rejection filter consists of 20 bits for Channel (2 bits), Frame ID (11 bits), and Cycle Code (7 bits). Rejection filter and rejection filter mask can be configured in "DEFAULT_CONFIG" or "CONFIG" state only. The filter configuration in the Header Sections of Message Buffers belonging to the FIFO is ignored.

A valid received Frame is stored in the FIFO if channel ID, Frame ID, and cycle counter are not rejected by the configured rejection filter and rejection filter mask, and if there is no matching dedicated receive buffer.

23.3.9 Transmit process

The transmit process is described in the following sections.

23.3.9.1 Static segment

For the static segment, if there are several messages pending for transmission, the message with the Frame ID corresponding to the next sending slot is selected for transmission.

The Data Section of transmit buffers assigned to the static segment can be updated until the end of the preceding time slot. This means that a transfer from the Input Buffer has to be started by writing to the Input Buffer Command Request register latest at this time.

23.3.9.2 Dynamic segment

In the dynamic segment, if several messages are pending, the message with the highest priority (lowest Frame ID) is selected next. Only Frame ID's which are higher than the largest static Frame ID are allowed for the dynamic segment.

In the dynamic segment different slot counter sequences are possible (concurrent sending of different Frame ID's on both channels). Therefore pending messages are selected according to their Frame ID and their channel configuration bit.

The Data Section of transmit buffers assigned to the dynamic segment can be updated until the end of the preceding slot. This means that a transfer from the Input Buffer has to be started by writing to the Input Buffer Command Request register latest at this time.

The start of latest transmit configured by MHDC.SLT in the MHD Configuration Register defines the maximum minislot value allowed before inhibiting new Frame transmission in the dynamic segment of the current cycle.

23.3.9.3 Transmit buffers

A portion of the ERAY Message Buffers can be configured as transmit buffers by programming bit CFG in the Header Section of the respective Message Buffer to 1. This can be done through the Write Header Section 1 register.

There exist the following possibilities to assign a transmit buffer to the Communication Controller channels:

- Static segment
  - Channel A or channel B, channel A and channel B
- Dynamic segment
  - Channel A or channel B

Message Buffer 0 is dedicated to hold the startup Frame, the SYNC Frame, or the designated single slot Frame as configured by TXST, TXSY, and TSM in the SUC Configuration Register 1. In this case it can be reconfigured in “DEFAULT_CONFIG” or “CONFIG” state only. This ensures that any node transmits at most one startup / SYNC Frame per communication cycle. Transmission of startup / SYNC Frames from other Message Buffers is not possible.

All other Message Buffers configured for transmission in static or dynamic segment are reconfigurable during run-time. Due to the organization of the Data Partition in the Message RAM (reference by data pointer), re-configuration of the configured payload length and the data pointer in the Header Section of a Message Buffer may lead to erroneous configurations. If a Message Buffer is reconfigured during run-time, it may happen that this Message Buffer is not send out in the respective communication cycle.

The Communication Controller does not have the capability to calculate the Header CRC. The Host is supposed to provide the Header CRCs for all transmit buffers. If Network Management is required, the Host has to set the PPIT bit in the Header Section of the respective Message Buffer to 1 and write the Network Management information to the Data Section of the Message Buffer.

The payload length field configures the data payload length in 2-byte words. If the configured payload length of a static transmit buffer is shorter than the payload length configured for the static segment by SFDL in the Message Handler Configuration Register 1, the Communication Controller generates padding byte to ensure that Frames have proper physical length. The padding pattern is logical zero.

Each transmit buffer provides a transmission mode flag TXM that allows the Host to configure the transmission mode for the transmit buffer in the static segment. If this bit is set, the transmitter operates in the single-shot mode. If this bit is cleared, the transmitter operates in the continuous mode. In dynamic segment the transmitter always works in single-shot mode.

If a Message Buffer is configured in the continuous mode, the Communication Controller does not reset the transmission request flag TXR after successful transmission. In this case a Frame is sent out each time the Frame ID and cycle counter filter match. The TXR flag can be reset by the Host by writing the respective Message Buffer number to the Input Buffer Command Request register while bit STXRH in the Input Buffer Command Mask register is reset to 0.

If two or more transmit buffers are configured with the same Frame ID and cycle counter filter value, the transmit buffer with the lowest Message Buffer number will be transmitted in the respective slot.

23.3.9.4 Frame transmission

To prepare a transmit buffer for transmission the following steps are required:

- Configure the Message Buffer as transmit buffer by writing bit CFG = 1 in the Write Header Section 1 register
- Write transmit message (Header and Data Section) to the Input Buffer
- To transfer a transmit message from Input Buffer to the Message RAM, proceed as described in the “Data Transfer from Input Buffer to Message RAM” topic
- If configured in the Input Buffer Command Mask register, the Transmission Request flag for the respective Message Buffer will be set as soon as the transfer has completed and the Message Buffer is ready for transmission
- Check whether the Message Buffer has been transmitted by checking the TXR bits (TXR = 0) in the Transmission Request 1, 2, 3, 4 registers (single-shot mode only)

In single-shot mode the Communication Controller resets the TXR flag after transmission has been completed. Now the Host may update the transmit buffer with the next message. The Communication Controller does not

transmit the message before the Host has indicated that the update is completed by setting the Transmission Request flag TXR again. The Host can check the actual state of the TXR flags of all Message Buffers by reading the Transmission Request registers. After successful transmission, if bit WRHS1.MBI in the Header Section of the respective Message Buffer is set, the transmit service request flag in the Status Service Request Register is set (TXI = 1). If enabled, an service request is generated.

Related information

Data transfer from input buffer to message RAM on page 4460

23.3.9.5 NULL frame transmission

If in static segment the Host does not set the transmission request flag before transmit time, and if there is no other transmit buffer with matching filter criteria (matching Frame ID and cycle counter filter), the Communication Controller transmits a NULL Frame with the NULL Frame indication bit reset to 0 and the payload data reset to zero.

In the following cases the Communication Controller transmits a NULL Frame with the NULL Frame indication bit reset to 0, and the rest of the Frame Header and the Frame length unchanged (payload data is reset to zero):

- All transmit buffers configured for the slot have cycle counter filters that do not match the current cycle
- There are matching Frame ID's and cycle counter filters, but none of these transmit buffers has the transmission request flag TXR set

NULL Frames are not transmitted in the dynamic segment.

23.3.10 Receive process

The receive process is described in the following sections.

23.3.10.1 Frame reception

To prepare or change a Message Buffer for reception the following steps are required:

- Configure the Message Buffer as receive buffer by writing bit CFG = 0 in the Write Header Section 1 register
- Configure the receive buffer by writing the configuration data (Header Section) to the Input Buffer
- Transfer the configuration from Input Buffer to the Message RAM by writing the number of the target Message Buffer to the Input Buffer Command Request register

Once these steps are performed, the Message Buffer functions as an active receive buffer and participates in the internal acceptance filtering process, which takes place every time the Communication Controller receives a message. The first matching receive buffer is updated from the received message. If the Message Buffer holds an unprocessed Data Section (ND = 1), it is overwritten with the new message and the MLST bit in the respective Message Buffer Status register is set.

If the payload length of a received Frame PLC is longer than the value programmed by PLC in the Header Section of the respective Message Buffer, the data field stored in the Message Buffer is truncated to that length.

If no Frame, a NULL Frame, or a corrupted Frame is received in a slot, the Data Section of the Message Buffer configured for this slot is not updated. In this case only the flags in the Message Buffer Status register are updated to signal the cause of the problem. In addition the respective MBC flag in the Message Buffer Status Changed 1, 2, 3, 4 registers is set.

When the Data Section of a receive buffer has been updated from a received Frame, the respective New Data NDn $\mathrm{NDn}\,(n = 0 - 31)$ to $\mathrm{NDn}\,(n = 96 - 127)$ flag in the New Data NDAT1 to NDAT4 registers is set. When the Message Handler has updated the Message Buffer status, the respective MBC flag in the Message Buffer Status Changed 1, 2, 3, 4 registers is set. If bit RDHS1.MBI in the Header Section of the respective Message Buffer is set, the receive service request flag in the Status Service Request Register is set (RXI = 1). If enabled, an service request is generated.

To read a receive buffer from the Message RAM through the Output Buffer proceed as described on the “Data Transfer from Message RAM to Output Buffer” topic. The ND and MBC flags are automatically cleared by the Message Handler when the received message has been transferred to the Output Buffer.

Related information

Data transfer from message RAM to output buffer on page 4461

23.3.10.2 NULL frame reception

The Payload Segment of a received NULL Frame is not copied into the matching receive buffer. If a NULL Frame has been received, the Header Section of the matching Message Buffer is updated from the received NULL Frame. The NULL Frame indication bit in the Header Section 3 of the respective Message Buffer is reset (NFI = 0) and the respective MBC flag in the Message Buffer Status Changed 1, 2, 3, 4 registers is set.

In case that bit ND and / or MBC were already set before this event because the Host did not read the last received message, bit MLST in the Message Buffer Status register of the respective Message Buffer is also set.

23.3.11 FIFO function

A group of the Message Buffers can be configured as a cyclic First-In-First-Out (FIFO). The group of Message Buffers belonging to the FIFO is contiguous in the register map starting with the Message Buffer referenced by First Buffer of FIFO MRC.FFB bit field and ending with the Message Buffer referenced by Last Configured Buffer MRC.LCB in the Message RAM Configuration register. Up to 128 Message Buffers can be assigned to the FIFO.

23.3.11.1 Description

Every valid incoming message not matching with any dedicated receive buffer but passing the programmable FIFO filter is stored into the FIFO. In this case Frame ID, payload length, receive cycle count, and the status bits of the addressed FIFO Message Buffer are overwritten with Frame ID, payload length, receive cycle count, and the status from the received message and can be read by the Host for message identification. Bit RFNE in the Status Service Request Register shows that the FIFO is not empty, bit RFCL in the Status Service Request Register is set when the receive FIFO fill level FSR.RFFL is equal or greater than the critical level as configured by FSR.CL, bit RFO in the Error Service Request Register shows that a FIFO overrun has been detected. If enabled, service requests are generated.

There are two index registers associated with the FIFO. The PUT Index Register (PIDX) is an index to the next available location in the FIFO. When a new message has been received it is written into the Message Buffer addressed by the PIDX register. The PIDX register is then incremented and addresses the next available Message Buffer. If the PIDX register is incremented past the highest numbered Message Buffer of the FIFO, the PIDX register is loaded with the number of the first (lowest numbered) Message Buffer in the FIFO chain. The GET Index Register (GIDX) is used to address the next Message Buffer of the FIFO to be read. The GIDX register is incremented after transfer of the contents of a Message Buffer belonging to the FIFO to the Output Buffer. The PUT Index Register and the GET Index Register are not accessible by the Host.

The FIFO is completely filled when the PUT index (PIDX) reaches the value of the GET index (GIDX). When the next message is written to the FIFO before the oldest message has been read, both PUT index and GET index are incremented and the new message overwrites the oldest message in the FIFO. This will set FIFO overrun flag RFO in the Error Service Request Register.

A FIFO non empty status is detected when the PUT index (PIDX) differs from the GET index (GIDX). In this case flag RFNE is set. This indicates that there is at least one received message in the FIFO. The FIFO empty, FIFO not empty and the FIFO overrun states for a three Message Buffer FIFO are explained in the figure below.

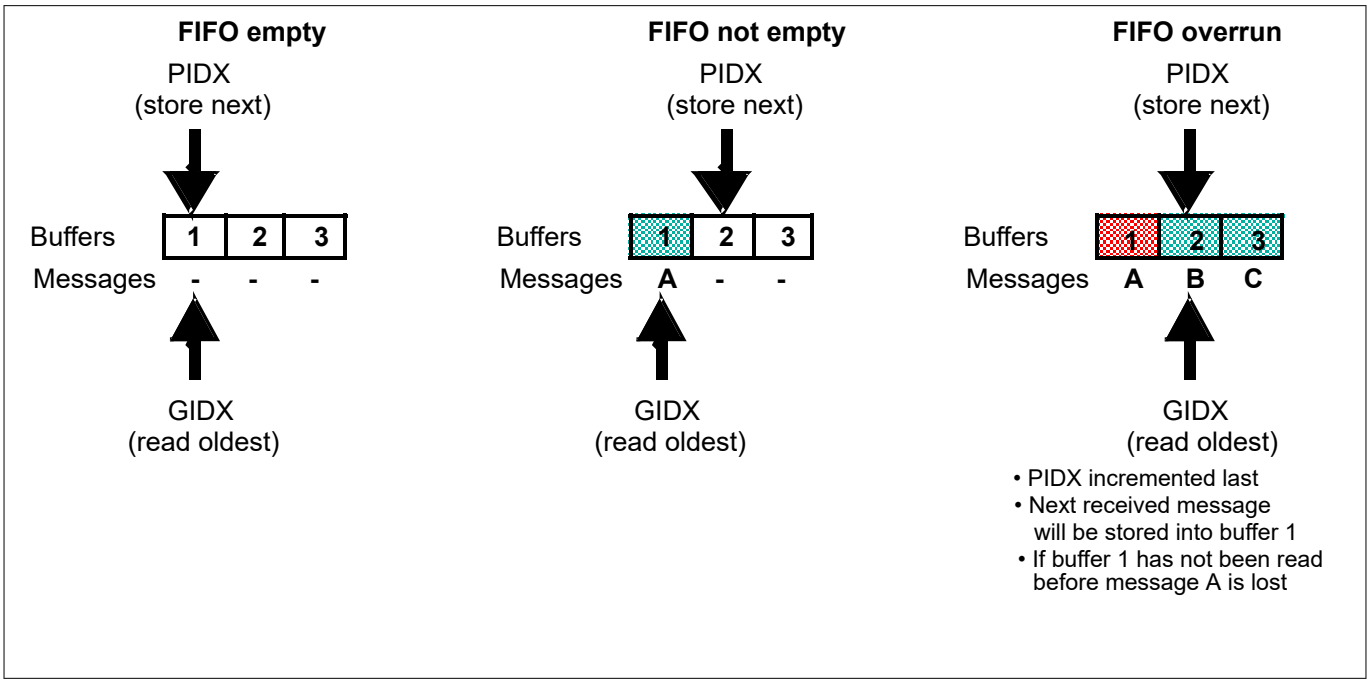


Figure 436 FIFO status: empty, not empty, overrun

There is a programmable FIFO rejection filter for the FIFO. The FIFO Rejection Filter register (FRF) defines a filter pattern for messages to be rejected. The FIFO rejection filter consists of channel filter, Frame ID filter, and cycle counter filter. If bit RSS is set to 1 (default), all messages received in the static segment are rejected by the FIFO. If bit RNF is set to 1 (default), received NULL Frames are not stored in the FIFO.

The FIFO Rejection Filter Mask register (FRFM) specifies which bits of the Frame ID filter in the FIFO Rejection Filter register are marked “do not care” for rejection filtering.

23.3.11.2 Configuration of the FIFO

For all Message Buffers belonging to the FIFO the data pointer to the first 32-bit word of the Data Section of the respective Message Buffer in the Message RAM has to be configured through the Write Header Section 3 register. All information required for acceptance filtering is taken from the FIFO rejection filter and the FIFO rejection filter mask and needs not be configured in the Header Sections of the Message Buffers belonging to the FIFO.

When programming the data pointers for the Message Buffers belonging to the FIFO, the payload length of all Message Buffers should be programmed to the same value.

***Note**: It is recommended to program the MBI bits of the Message Buffers belonging to the FIFO to 0 through WRHS1.MBI to avoid generation of RX interrupts. If the payload length of a received Frame is longer than the value programmed by WRHS2.PLC in the Header Section of the respective Message Buffer, the data field stored in a Message Buffer of the FIFO is truncated to that length.*

23.3.11.3 Access to the FIFO

To read from the FIFO the Host has to trigger a transfer from the Message RAM to the Output Buffer by writing the number of the first Message Buffer of the FIFO (referenced by FFB) to the Output Buffer Command Request register. The Message Handler then transfers the Message Buffer addressed by the GET Index Register (GIDX) to the Output Buffer. After this transfer the GET Index Register (GIDX) is incremented.

23.3.12 Message handling

The Message Handler controls data transfers between the Input or Output Buffer and the Message RAM and between the Message RAM and the two Transient Buffer RAMs. All accesses to the internal RAM’s are 32 bit accesses.

Access to the Message Buffers stored in the Message RAM is done under control of the Message Handler state machine. This avoids conflicts between accesses of the two protocol controllers and the Host to the Message RAM.

Frame IDs of Message Buffers assigned to the static segment have to be in the range from 1 to NSS as configured in the GTU Configuration Register 7. Frame IDs of Message Buffers assigned to the dynamic segment have to be in the range from $\mathrm{NSS} + 1$ to 2047 .

Received messages with no matching dedicated receive buffer (static or dynamic segment) are stored in the receive FIFO (if configured) if they pass the FIFO rejection filter.

23.3.12.1 Host access to message RAM

The message transfer between Input Buffer and Message RAM as well as between Message RAM and Output Buffer is triggered by the Host by writing the number of the target/source Message Buffer to be accessed to the Input or Output Buffer Command Request register.

The Input/Output Buffer Command Mask registers can be used to write/read Header and Data Section of the selected Message Buffer separately. If bit STXRS in the Input Buffer Command Mask register is set (STXRS = 1), the transmission request flag TXR of the selected Message Buffer is automatically set after the Message Buffer has been updated.

If bit STXRS in the Input Buffer Command Mask register is reset (STXRS = 0), the transmission request flag TXR of the selected Message Buffer is reset. This can be used to stop transmission from Message Buffers operated in continuous mode.

Input Buffer (IBF) and the Output Buffer (OBF) are build up as a double buffer structure. One half of this double buffer structure is accessible by the Host (IBF Host/OBF Host), while the other half (IBF Shadow/OBF Shadow) is accessed by the Message Handler for data transfers between IBF/OBF and Message RAM.

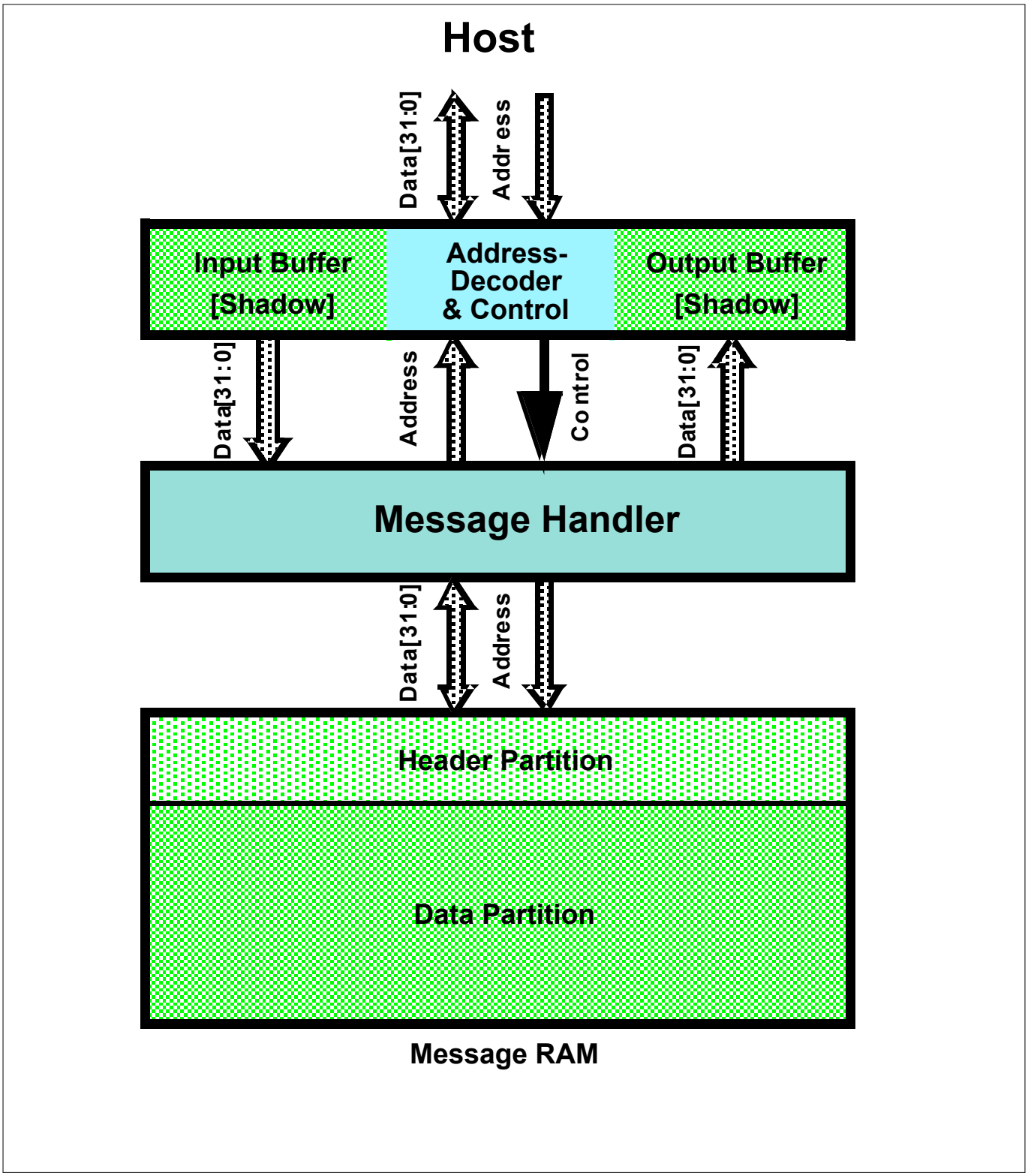


Figure 437 Host access to message RAM

23.3.12.1.1 Data transfer from input buffer to message RAM

To configure or update a Message Buffer in the Message RAM, the Host has to write the data to WRDSn $(n = 1 - 64)$ and the Header to WRHS1, WRHS2, WRHS3. Two sets of WRDSn $(n = 1 - 64)$ are available in parallel and selected by CUST1.IBF1PAG and CUST1.IBF2PAG. CUST1.IBFS shows which Input Buffer is currently used as Input Shadow Buffer and which as Input Host Buffer. WRHS1, WRHS2 and WRHS3 do only exist once. The specific action is selected by configuring the Input Buffer Command Mask IBCM.

When the Host writes the number of the target Message Buffer in the Message RAM to IBRH in the Input Buffer Command Request register IBCR, IBF Host and IBF Shadow are swapped (see figure below).

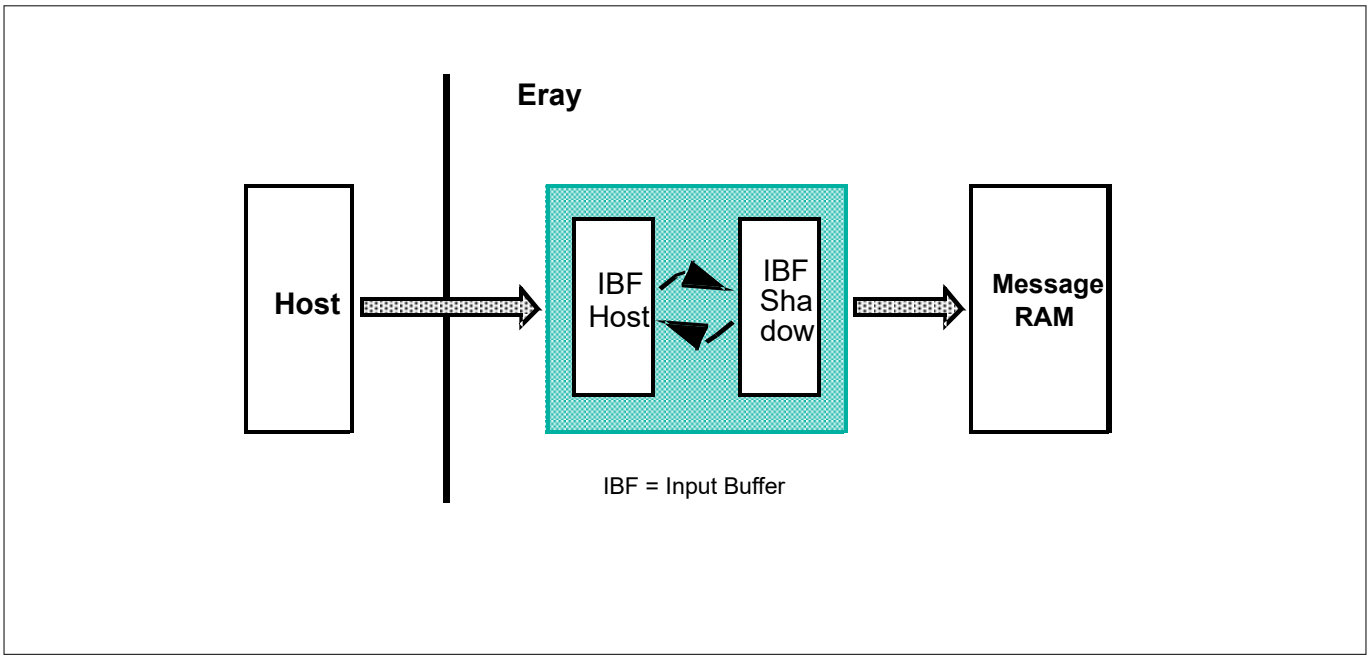


Figure 438 Swapping of IBCM and IBCR Bit

In addition the bits in the Input Buffer Command Mask and Input Buffer Command Request registers are also swapped to keep them attached to the respective IBF section (see Figure below).

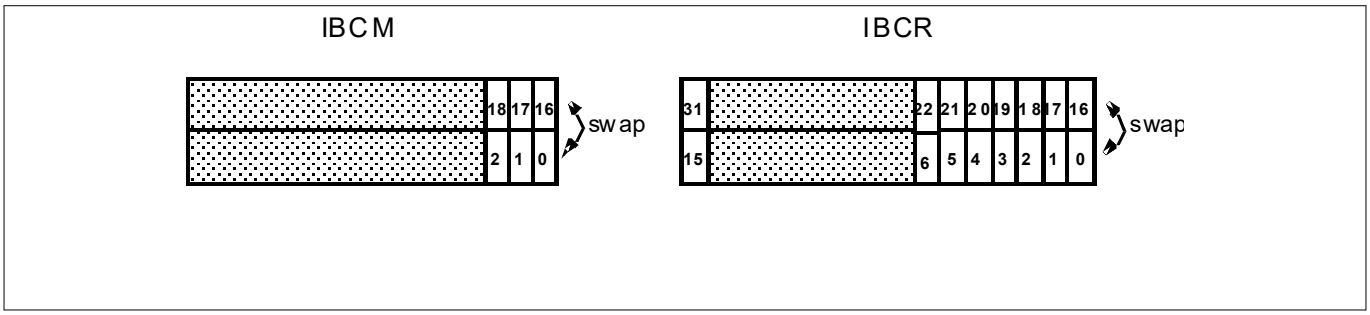


Figure 439 Swapping of IBCM and IBCR Bit

With this write operation the IBSYS bit in the Input Buffer Command Request register is set to 1. The Message Handler then starts to transfer the contents of IBF Shadow to the Message Buffer in the Message RAM selected by IBRS.

While the Message Handler transfers the data from IBF Shadow to the target Message Buffer in the Message RAM, the Host may write the next message to IBF Host. After the transfer between IBF Shadow and the Message RAM has completed, the IBSYS bit is set back to 0 and the next transfer to the Message RAM may be started by the Host by writing the respective target Message Buffer number to IBRH in the Input Buffer Command Request register.

If a write access to IBRH occurs while IBSYS is 1, IBSYH is set to 1. After completion of the ongoing data transfer from IBF Shadow to the Message RAM, IBF Host and IBF Shadow are swapped, IBSYH is reset to 0, IBSYS remains

set to 1, and the next transfer to the Message RAM is started. In addition the Message Buffer numbers stored under IBRH and IBRS and the Command Mask flags are also swapped.

Table 1083 Assignment of input buffer command mask bit

Pos.	Access	Bit	Function
18	rh	STXRS	Set transmission request shadow
17	rh	LDSS	Load data section shadow
16	rh	LHSS	Load header section shadow
2	rwh	STXRH	Set transmission request host
1	rwh	LDSH	Load data section host
0	rwh	LHSH	Load header section host

Table 1084 Assignment of input buffer command request bit

Pos.	Access	Bit	Function
31	rh	IBSYS	IBF Busy Shadow, signals ongoing transfer from IBF Shadow to Message RAM
22…16	rh	IBRS	IBF Request Shadow, number of Message Buffer currently or last updated
15	rh	IBSYH	IBF Busy Host, transfer request pending for Message Buffer referenced by IBRH
6-0	rwh	IBRH	IBF Request Host, number of Message Buffer to be updated next

23.3.12.1.2 Data transfer from message RAM to output buffer

To read a Message Buffer from the Message RAM, the Host has to write to Command Request register OBCR to trigger the data transfer as configured in Output Buffer Command Mask OBCM register. After the transfer has completed, the Host can read the transferred data from $\mathrm{RDDSn}\ (n = 1 - 64)$, RDHS1, RDHS2, RDHS3 and MBS.

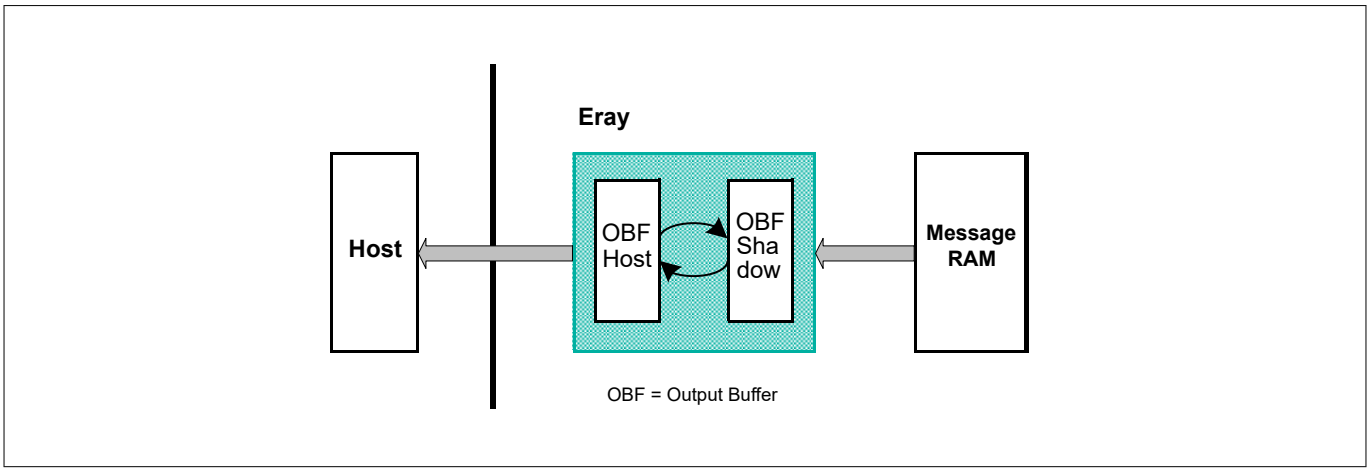


Figure 440 Double buffer structure output buffer

OBF Host and OBF Shadow as well as bits OBCM.RHSS, OBCM.RDSS, OBCM.RHSH, OBCM.RDSH and bits OBCR.OBRS, OBCR.OBRH are swapped under control of bits OBCR.VIEW and OBCR.REQ.

Writing bit OBCR.REQ to 1 copies bits OBCM.RHSS, OBCM.RDSS and bits OBCR.OBRS to an internal storage (see figure below).

After setting OBCR.REQ to 1, OBCR.OBSYS is set to 1, and the transfer of the Message Buffer selected by OBCR.OBRS from the Message RAM to OBF Shadow is started. After the transfer between the Message RAM and OBF Shadow has completed, the OBCR.OBSYS bit is set back to 0. Bits OBCR.REQ and OBCR.VIEW can only be set to 1 while OBCR.OBSYS is 0.

OBF Host and OBF Shadow are swapped by setting bit OBCR.VIEW to 1 while bit OBCR.OBSYS is 0.

In addition bits OBCR.OBRH and bits OBCM.RHSH, OBCM.RDSH are swapped with the registers internal storage thereby assuring that the Message Buffer number stored in OBCR.OBRH and the mask configuration stored in OBCM.RHSH, OBCM.RDSH matches the transferred data stored in OBF Host.

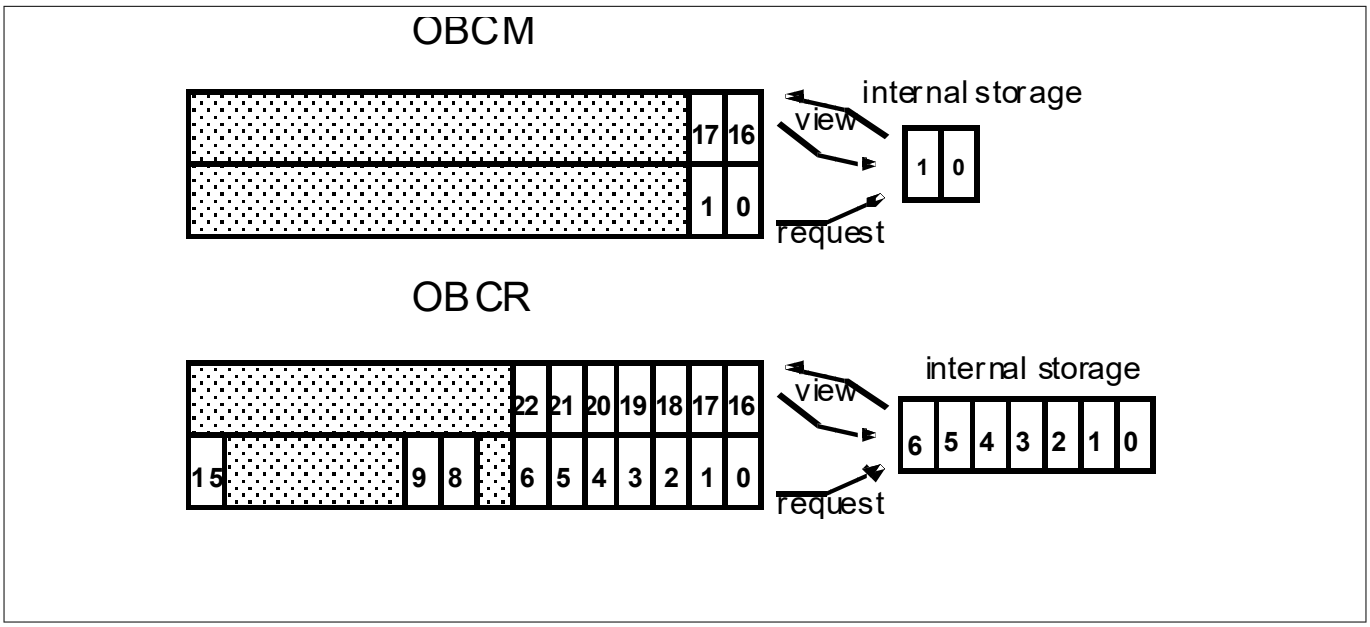


Figure 441 Swapping of OBCM and OBCR Bit

Now the Host can read the transferred Message Buffer from OBF Host while the Message Handler may transfer the next message from the Message RAM to OBF Shadow.

Table 1085 Assignment of output buffer command mask bit

Pos.	Access	Bit	Function
17	rh	RDSH	Data section available for host access
16	rh	RHSH	Header section available for host access
1	rwh	RDSS	Read data section shadow
0	rwh	RHSS	Read header section shadow

Table 1086 Assignment of output buffer command request bit

Pos.	Access	Bit	Function
22…16	rh	OBRH	OBF Request Host number of Message Buffer available for Host access
15	rh	OBSYS	OBF Busy Shadow signals ongoing transfer from Message RAM to OBF Shadow
9	rw	REQ	Request Transfer from Message RAM to OBF Shadow
8	rw	VIEW	View OBF Shadow, swap OBF Shadow, and OBF Host
6…0	rw	OBRS	OBF Request Shadow number of Message Buffer for next request

23.3.12.1.3 Automatic delay write access to OBCR and IBCR

Write and read accesses to the Output Buffer Control Register (OBCR) can be automatically stalled due to a ongoing transfer from the Message Buffer to the Output Buffer. Also write and read accesses to the Input Buffer Control Register (IBCR) may be automatically delayed due to a ongoing transfer from the Input Buffer to the Message Buffer.

This delay scheme can be controlled (enabled or disabled) by CUST1.IEN and CUST1.OEN. The maximum time to stall a write or read access is determined by a single time-out counter preluded with the 32-bit value specified in the bit field CUST3.TO_RV If the time-out counter counts down to zero before the transfer to/from the Message Buffer is completed, the access (read or write) will be canceled and a service request will be generated. A canceled read access provides a 0 value. A canceled write access does not modify any bits in the OBCR or IBCR. In addition the bit CUST1.INT0 of the service request status register will be set and must be reset by the host to disable the service request line.

The read and write access to the Output Buffer Control Register (OBCR) may be configured without automatic delay by clearing CUST1.OEN. Setting OBCR.REQ and immediately afterwards reading or writing OBCR, for example to set OBCR.VIEW will lead to a canceled read or write operation, for example OBCR.VIEW remains cleared, and an error is signaled by a set EIR.IOBA. Besides canceling the erroneous read or write operation, and setting the error bit, no further state change happens. So full operation is granted. OBCR remains read and write inaccessible until the transfer of data from the Message Buffer to the Output Buffer ($\mathrm{MBF} \Rightarrow \mathrm{OBF}$) is completed. During this time span all read and write accesses to the Output Buffer Control Register (OBCR) are canceled. The transfer is completed when OBCR.OBSYS is cleared. Additionally signal SIR.TOBC may be used, for example for service request triggering, DMA triggering, or driving a pin, to communicate the access status.

The read and write access to the Output Buffer Control Register (OBCR) may be configured to be automatic delayed by setting CUST1.OEN and configuring CUST3.TO_RV to the maximum stall time acceptable to the system. If setting OBCR.REQ and immediately afterwards reading or writing to OBCR, for example to set the OBCR.VIEW bit, this read or write will be stalled until either the maximum delay time elapsed (in this case the read or write operation is canceled after the stall time, for example OBCR.VIEW remains cleared, and an error is signaled by setting EIR.IOBA or the read or write completes normally, for example set OBCR.VIEW after the transfer of data from the Message Buffer to the Output Buffer ($\mathrm{MBF} \Rightarrow \mathrm{OBF}$) is finalized. During this time the bus is locked and no further access to the Eray module is possible due to the ongoing stalled read or write operation. Because no access is possible to the Eray module, read or write stall may only be detected through the signal SIR.TOBC or due to other not processed read or write accesses to the Eray module.

The read and write access to the Input Buffer Control Register (IBCR) may also be configured without automatic delay by clearing CUST1.IEN By writing to IBCR.IBRH the Input Buffers are swapped (shadow IBF changes to host IBF and host IBF to shadow IBF), the content of the shadow IBF is copied into the MBF ($\mathrm{IBF} \Rightarrow \mathrm{MBF}$), and IBCR.IBSYS is set. Writing to IBCR.IBRH a second time while IBCR.IBSYS remained set (previously initiated copy process IBF MBF ongoing) will correctly update IBCR.IBRH and set IBCR.IBSYH. This will set the signal IBUSY. A third access, read or write, to IBCR while IBCR.IBSYH remains set will cancel this third access and an error is signaled by setting EIR.IIBA. Besides canceling this last access to IBCR and setting the error bit, no further state change happens. So full operation is granted. IBCR remains read and write inaccessible until the transfer of data from the Input Shadow Buffer to the Message Buffer ($\mathrm{IBF} \Rightarrow \mathrm{MBF}$) is completed and once more the Input Buffers are swapped (shadow IBF changes to host IBF and host IBF to shadow IBF). During this time span all read and write accesses to the Input Buffer Control Register (IBCR) are canceled. The transfer is completed when IBCR.IBSYH is cleared. Additionally signal SIR.TIBC may be used, for example for service request triggering, DMA triggering, or driving a pin, to communicate the access status.

The read and write access to the Input Buffer Control Register (IBCR) may be configured for being automatically delayed by setting CUST1.IEN and configuring CUST3.TO_RV to the maximum stall time acceptable to the system. By writing to IBCR.IBRH the Input Buffers are swapped (shadow IBF changes to host IBF and host IBF to shadow IBF), the content of the shadow IBF copied into the MBF, and IBCR.IBSYS is set. Writing to IBCR.IBRH a second time while IBCR.IBSYS remains set (previously initiated copy process ongoing) will correctly update IBCR.IBRH and set IBCR.IBSYH. This will set the signal IBUSY. A third access to IBCR while IBCR.IBSYH remains set will stall this read or write until either the maximum delay time elapsed (in this case the read or write

operation is cancelled after the stall time and an error is signalled by setting EIR.IOBA or the read or write completes normally, after the transfer of data from the Input Shadow Buffer to the Message Buffer (IBF ⇒ MBF) is finalized and once more the Input Buffers are swapped (shadow IBF changes to host IBF and host IBF to shadow IBF). During this time the bus is locked and no further access to Eray module is possible due to the ongoing stalled read or write operation. Because no access is possible to the Eray module, read or write stall may only be detected through the signal SIR.TIBC or due to other not processed read or write accesses to the Eray module.

So setting CUST3.TO_RV = FFFFFFFFH, CUST1.IEN = 1, and CUST1.OEN = 1 will always grant a consistent data access of the host to the Output and Input Buffers without the need of reading and taking into account the status of OBCR.OBSYS or IBCR.IBSYH. But this simplified access may cause system latencies and system performance loss.

23.3.12.2 Data Transfers between IBF/OBF and message RAM

This document uses the following terms and abbreviations:

Table 1087 Terms and abbreviations

Term	Meaning
MHD	Message Handler
IBF	Input Buffer 1 or 2 RAM
OBF	Output Buffer 1 or 2 RAM
MBF	Message Buffer RAM
TBF	Transient Buffer RAM Channel A (TBF1) or Channel B (TBF2)
IBF ⇒ MBF	Transfer from IBF to MBF
MBF ⇒ OBF	Transfer from MBF to OBF
MBF ⇒ TBF	Transfer from MBF to TBF
TBF ⇒ MBF	Transfer from TBF to MBF
SS	Slot Status
SS ⇒ MBF	Transfer SS to MBF

Message Handler functionality

The MHD controls the access to the MBF. It manages data-transfer between MBF and IBF, OBF, TBF1, TBF2. The data-path are shown in Figure below.

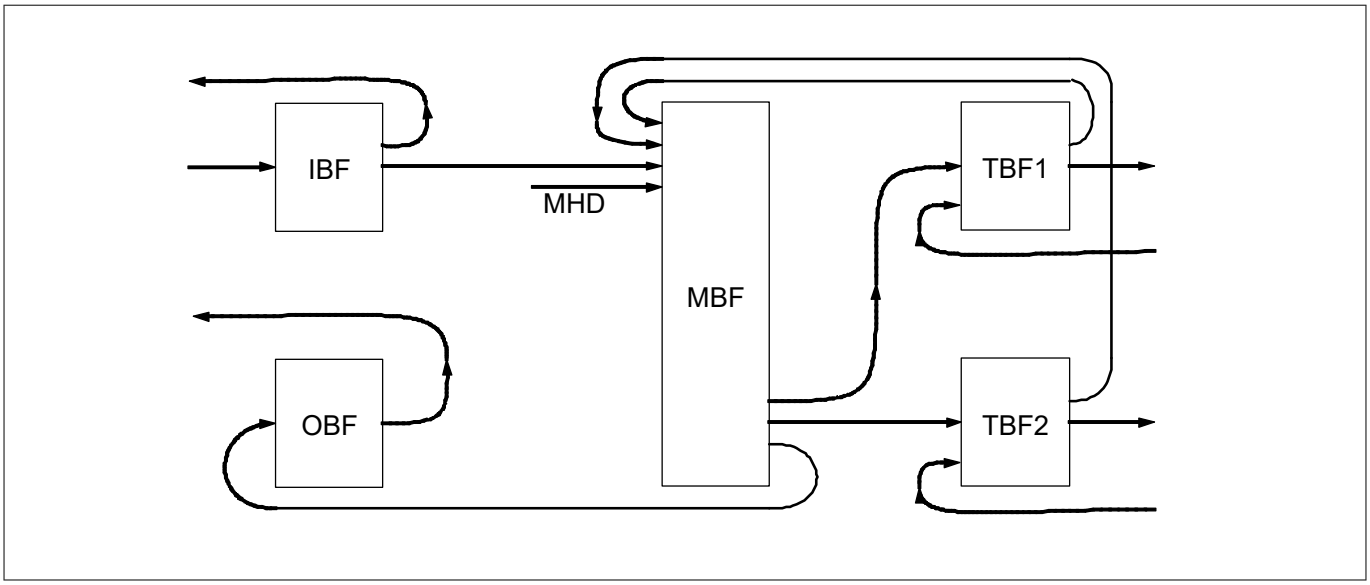


Figure 442 Interconnection of RAMs

Furthermore a search-algorithm allows to find the next valid message object in the MBF for transmission or reception.

Each transfer consists of a setup-time, four time steps to transfer the Header-section and a payload-length-dependent number of time steps to transfer the data-section. The internal data-busses have a width of 32 bits. Thereby it is possible to transfer two 2-byte words in one time step. If the payload consists of an odd number of 2-byte words, the last time step of the data-section contains only 16 bit of valid data. If the Payload-Length (PL) is for example 7, the data-section consists of 4 time steps.

The maximum length for the data-section is 64 time steps, the minimum length is zero time steps.

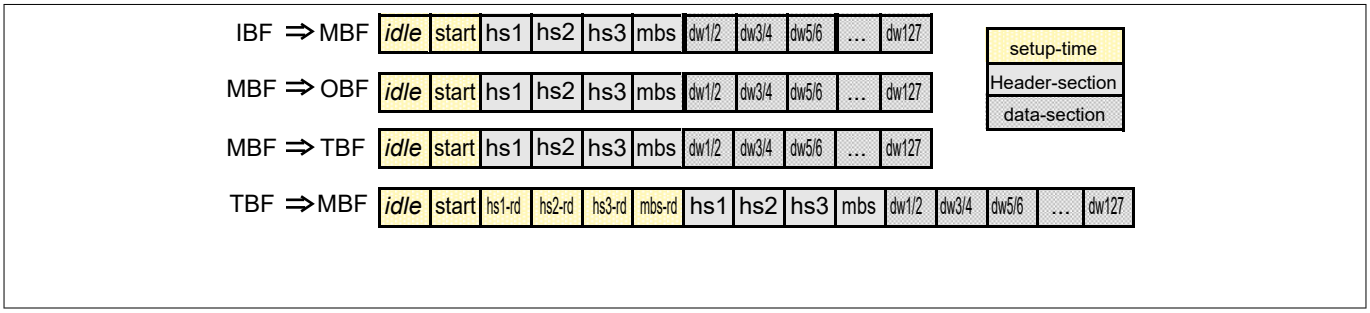


Figure 443 Different possible buffer transfers

The update of the Slot-Status consists of a setup-time and one time-step to write the new Slot-Status.

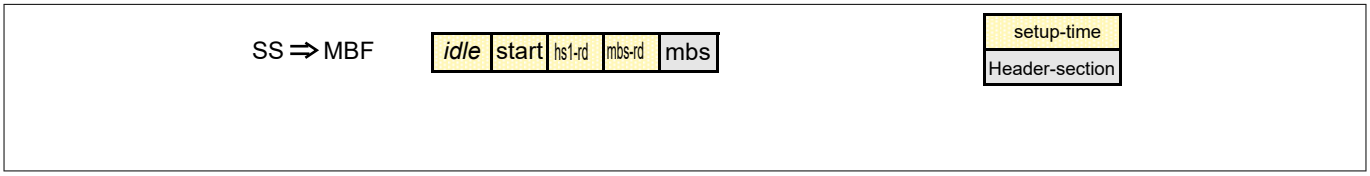


Figure 444 Update of slot status

The length of a time step depends on the number of concurrent tasks.

The following concurrent tasks are executed under control of the Message Handler:

- Data transfer between IBF or OBF and MBF
- Data transfer between TBF1 and MBF, search next TX/RX Message Buffer CHA
- Data transfer between TBF2 and MBF, search next TX/RX Message Buffer CHB

Thereby the time step length can vary between one and three f_{CLC_ERAY} periods.

Under certain conditions it is possible that a transfer is stopped or interrupted for a number of time steps until it is continued.

When a IBF ⇒ MBF is started short after a TBF ⇒ MBF or SS ⇒ MBF the transfer from IBF has to wait until the setup-time of the internal transfer has finished (see Figure below)

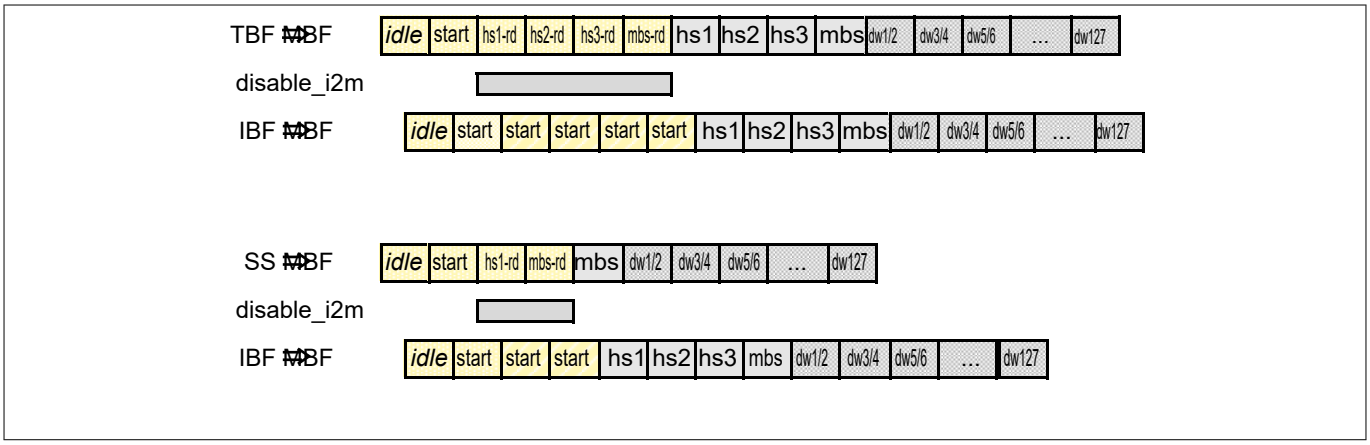


Figure 445 Delay start of IBF ⇒ MBF

The internal signal “disable_i2m” is always active when the TBF ⇒ MBF is in state “hs1-rd”, “hs2-rd”, “hs3-rd” or “mbs-rd” and when the SS ⇒ MBF is in state “hs1-rd” or “mbs-rd”.

The IBF ⇒ MBF is hold in state “start” until the internal signal “disable_i2m” gets inactive.

These additional time-steps are independent of any address-counter-values. This means, the IBF ⇒ MBF has to wait even if it writes to another buffer than the internal transfer.

Multiple requests of transfers between IBF/OBF and message RAM

The time required to transfer the contents of a Message Buffer between IBF / OBF and Message RAM depends on the number of 4-byte words to be transferred, the number of concurrent tasks to be managed by the Message Handler, and in special cases the type and address range of the internal transfer. The number of 4-byte words varies from 4 (Header Section only) to 68 (Header + maximum Data Section) plus a short setup time to start the first transfer, while the number of concurrent task varies from one to three. The 4 Header words have to be included in calculation even if only the Data Section is requested for transfer.

The following concurrent tasks are executed under control of the Message Handler:

- Data transfer between IBF or OBF and MBF
- Data transfer between TBF1 and MBF, search next TX/RX Message Buffer CHA
- Data transfer between TBF2 and MBF, search next TX/RX Message Buffer CHB

Transfers between IBF and MBF respectively MBF and OBF can only be handled one after another. In case that for example a IBF ⇒ MBF has been started shortly before a MBF ⇒ OBF is requested, the MBF ⇒ OBF has to wait until the IBF ⇒ MBF has completed.

In case that for example a second IBF ⇒ MBF is requested, a MBF ⇒ OBF is requested and a IBF ⇒ MBF is ongoing, the MBF ⇒ OBF has to wait until the first IBF ⇒ MBF has completed. The second IBF ⇒ MBF has to wait until the MBF ⇒ OBF has completed (see figure below) independent whether MBF ⇒ OBF or second IBF ⇒ MBF is requested first.

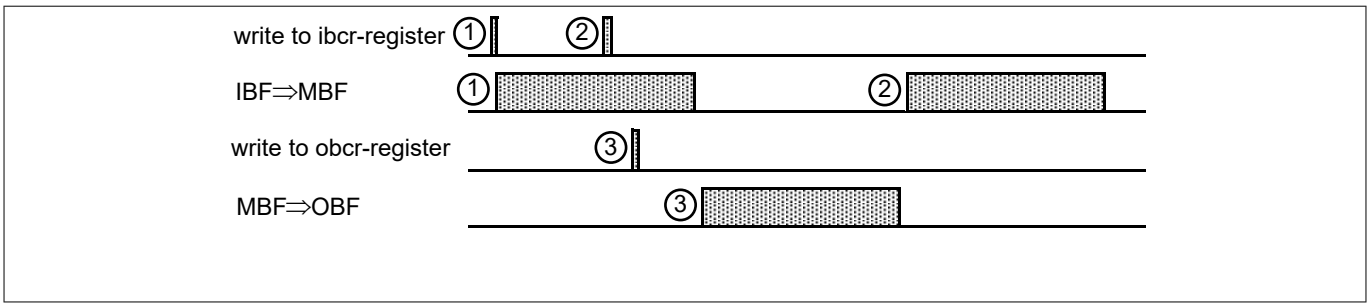


Figure 446 Multiple IBF/OBF request

Worst case for single request

When a message with a large payload length is received the TBF ⇒ MBF is started at the begin of the next slot ($n + 1$). If the next slot is a dynamic slot without transmission/reception (minislot), it may happen that the TBF ⇒ MBF has not finished until begin of the next but one slot ($n + 2$). In this case the TBF ⇒ MBF will be service requested (break) to start a transmission in the next but one slot (MBF ⇒ TBF) and/or to update the slot status (SS ⇒ MBF) for the RX-buffer corresponding with next slot ($n + 1$). After this interruption the TBF ⇒ MBF is continued.

For the transfers IBF ⇒ MBF / MBF ⇒ OBF , TBF ⇒ MBF and MBF ⇒ TBF separate address-counter are implemented, see figure below.

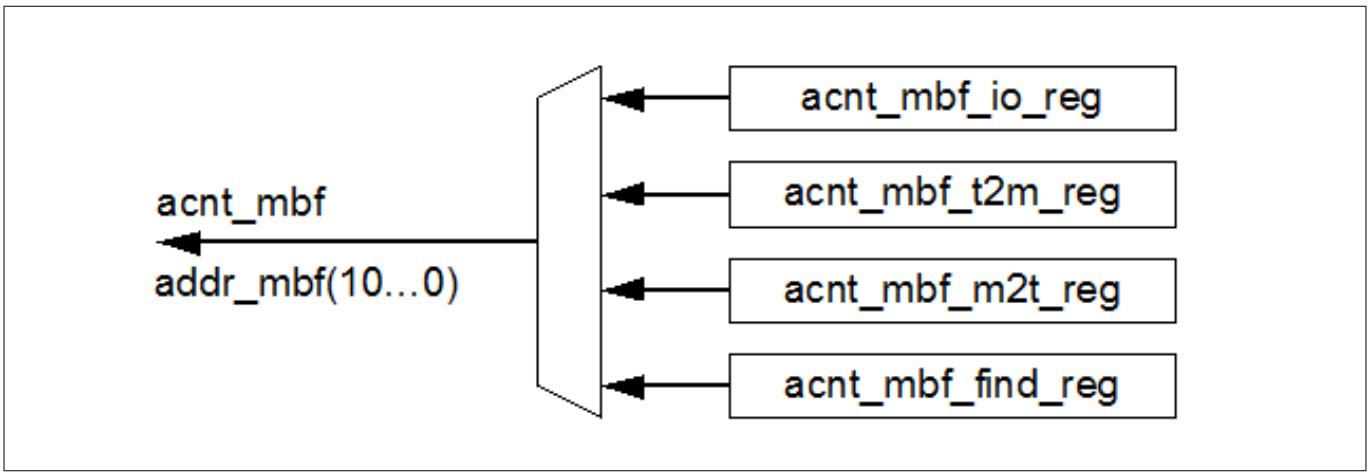


Figure 447 Address counter scheme of message RAM (simplified)

If the address-counter for IBF ⇒ MBF / MBF ⇒ OBF (acnt_mbf_io_reg) reaches the address of the interrupted TBF ⇒ MBF (acnt_mbf_t2m_reg) the IBF ⇒ MBF / MBF ⇒ OBF has to wait until the TBF ⇒ MBF is continued, see figure below.

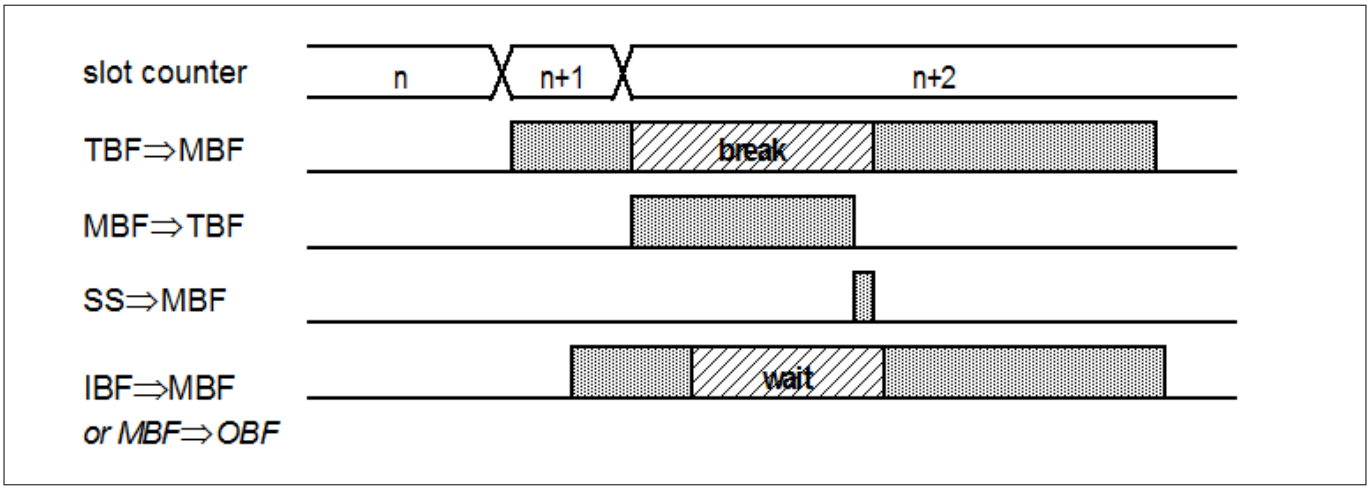


Figure 448 Interruption of TBF to MBF

The relative time is measured in f_{CLC_ERAY} cycles. Absolute time depends on the actual f_{CLC_ERAY} cycle period.

$$\text{tbf_to_mbf_breaktime}_{max} = (\text{setup time} + \text{mbf_to_tbf time}_{max}) + (\text{setup time} + \text{ss_to_mbf})$$

$$\text{cycles}_{req} = \text{number of concurrent tasks} \times \left((\text{setup time} + (\text{number of 4_byte words})_{req}\right) + \text{tbf_to_mbf_break time})$$

$$\text{setup time} = 2 f_{CLC_ERAY} \text{ cycles}$$

Worst case for one transfer

Either IBF ⇒ MBF or MFB ⇒ OBF.

Maximum break time: $\text{tbf_to_mbf_break time}_{max} = (2 + 68) + (4 + 1) = 75$

Maximum number of f_{CLC_ERAY} cycles: $\text{cycles}_{req} = 3 \times (6 + 68 + 75) = 435$

Worst case for multiple transfers

If a second IBF ⇒ MBF and a MBF ⇒ OBF is requested directly after the first IBF ⇒ MBF has started following worst case timing could appear:

$$\text{cycles}_{trans} = (\text{remaining cycles of transfer running}) + (\text{cycles of second request transfer}) + (\text{cycles of third request transfer})$$

$$\text{cycles}_{trans} = \text{cycles}_{rem} + \text{cycles}_{req_2} + \text{cycles}_{req_3}$$

$$\text{cycles}_{trans} = 447 + 435 + 447 = 1329$$

Maximum number of f_{CLC_ERAY} cycles: 1329

23.3.12.3 Minimum f_{CLC_ERAY}

To calculate the minimum f_{CLC_ERAY} the worst case scenario has to be considered.

The worst case scenario depends on the following parameters:

- Maximum payload length
- Minimum minislot length
- Number of configured Message Buffers (excluding FIFO)
- Used channels (single/dual channel)

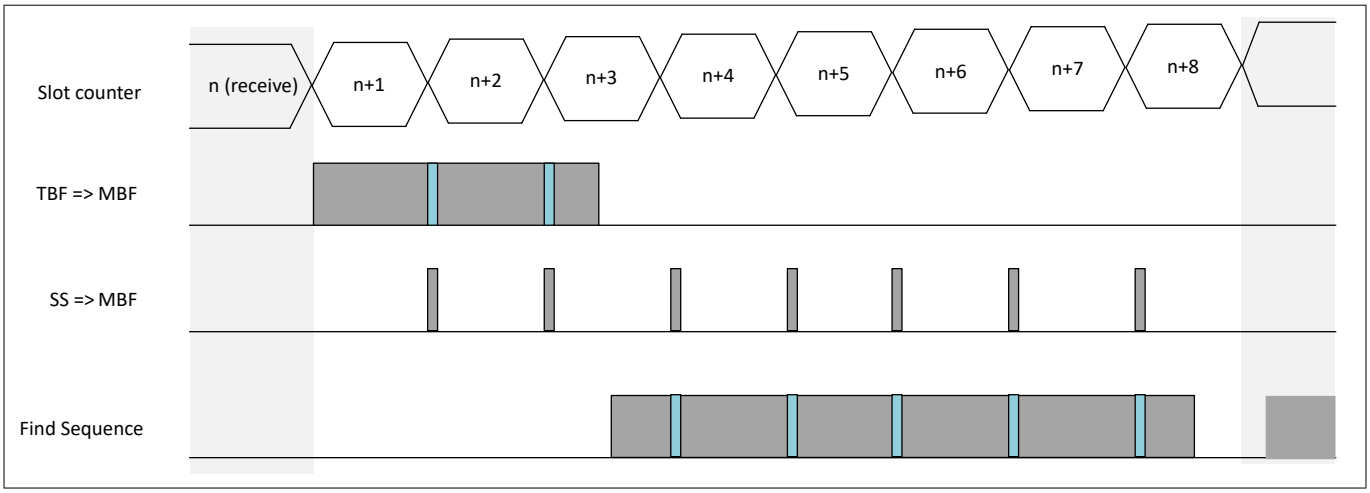


Figure 449 Worst-case scenario

Worst-case scenario:

- Reception of message with a maximum payload length in slot n (n is 7, 15, 23, 31, 39, …)
- Slot $n + 1$ to $n + 7$ are empty dynamic slots (minislot) and configured as receive buffer
- The find-sequence (usually started in slot 8, 16, 24, 32, 40, …) has to scan the maximum number of configured buffers
- The number of concurrent tasks has its maximum value of three

The find-sequence is executed each 8 Slots (slot 8, 16, 24, 32, 40, …). It has to be finished until the next find-sequence is requested.

The length of a TBF ⇒ MBF varies from 4 (Header Section only) to 68 (Header + maximum Data Section) time step plus a setup time of 6 time steps.

$$f_{\mathrm{CLC_ERAY}}\,\mathrm{cycles}_{t2m} = (\text{number of concurrent tasks}) \times (\text{setup time}_{t2m} + (\text{number of 4_byte words})_{t2m})$$

A SS ⇒ MBF has a fixed length of 1 time steps plus a setup time of 4 time steps.

$$f_{\mathrm{CLC_ERAY}}\,\mathrm{cycles}_{\mathrm{ss2}m} = (\text{number ofconcurrent tasks}) \times 5$$

The find sequence has a maximum length of 128 (maximum number of buffers) time steps plus a setup time of 2 time steps.

$$f_{\mathrm{CLC_ERAY}}\,\mathrm{cycles}_{\mathrm{find}} = (\text{number ofconcurrent tasks}) \times (\text{setup time}_{\mathrm{find}} + (\text{number of configured buffers}))$$

A minislot has a length of 2 to 63 Macrotick (gdMinislot). The minimum nominal Macrotick period (cdMinMTNom) is 1 µs. A sequence of 8 minislots has a length of

$$\mathrm{time}_{\mathrm{8minislots}} = 8 \times \mathrm{gdMinislot} \times \mathrm{cdMinMTNom}$$

The maximum period $T_{\mathrm{CLC_ERAY}} = \dfrac{1}{f_{\mathrm{CLC_ERAY}}}$ can be calculated as followed:

$$\mathrm{time}_{\mathrm{8minislots}} \geq (f_{\mathrm{CLC_ERAY}}\,\text{period in µs}) \times (f_{\mathrm{CLC_ERAY}}\,\mathrm{cycles}_{t2m}) + 7 \times (f_{\mathrm{CLC_ERAY}}\,\mathrm{cycles}_{\mathrm{ss2}m}) + (f_{\mathrm{CLC_ERAY}}\,\mathrm{cycles}_{\mathrm{find}}))$$

$$\mathrm{cycletrans} = (\text{remaining cycle of transfer running}) + (\text{cycles of requested transfer})$$

$$\mathrm{minimum} f_{\mathrm{CLC_ERAY}}\,\mathrm{cycles}_{t2m} = 3 \times (6 + 68) = 222$$

$$\mathrm{minimum}\ \mathrm{time}_{\mathrm{8minislots}} = 8 \times 2 \times 1\mu\mathrm{s} = 16\mu\mathrm{s}$$

$$\mathrm{minimum} f_{\mathrm{CLC_ERAY}}\,\mathrm{cycles}_{t2m} = 3 \times (6 + 68) = 222$$

$$\mathrm{minimum} f_{\mathrm{CLC_ERAY}}\,\mathrm{cycles}_{\mathrm{ss2}m} = 3 \times 5 = 15$$

$$\mathrm{minimum} f_{\mathrm{CLC_ERAY}}\,\mathrm{cycles}_{\mathrm{find}} = 3 \times (2 + 128) = 390$$

$$f_{CLC_ERAY}\ \text{period in ms} \leq \frac{16\mu s}{222 + 7 + 15 + 390} \approx 22.315\text{ns}$$

The minimum f_{CLC_ERAY} frequency for this worst case scenario is 44.8125 MHz.

A too low f_{CLC_ERAY} frequency can cause a malfunction of the ERAY.

The ERAY can detect several malfunctions and reports this by setting the corresponding flag in the Message Handler Constraints Flags (MHDF) register.

Minimum f_{CLC_ERAY} for various maximum payload length

Table below summarizes the minimum required f_{CLC_ERAY} frequency for various maximum payload length assuming:

- A minimum minislot length of 2 µs
- A maximum of 128 configured Message Buffers
- Dual channels in use

Table 1088 Minimum f_{CLC_ERAY} for different maximum payload length

Maximum payload length of 32 bit words	4	8	16	32	64
minimum f_{CLC_ERAY}	32.82 MHz	33.57 MHz	35.07 MHz	38.07 MHz	44.1 MHz

Minimum f_{CLC_ERAY} for various minimum minislot length

Table below summarizes the minimum required f_{CLC_ERAY} frequency for various minimum minislot length assuming:

- A maximum payload length of 254 bytes/64 four-byte-words
- A maximum 128 configured Message Buffers
- Dual channels in use

Table 1089 Minimum f_{CLC_ERAY} for different minimum minislot length

gdMinislot at dMinMTNom = 1 µs	2 µs	3 µs	4 µs	7 µs	8 µs
minimum f_{CLC_ERAY}	44.82 MHz	29.88 MHz	22.412 MHz	12.8 MHz	9.96 MHz

Minimum f_{CLC_ERAY} for various amount of configured message buffers

Table below summarizes the minimum required f_{CLC_ERAY} frequency for various amount of configured Message Buffers assuming:

- A maximum payload length of 254 bytes/64 four-byte-words
- A minimum minislot length of 2 µs
- Dual channels in use

Table 1090 Minimum f_{CLC_ERAY} for different amount of configured message buffers

Configured maximum amount of message buffers	128	64	32
minimum f_{CLC_ERAY}	44.82 MHz	32.82 MHz	26.82 MHz

Minimum f_{CLC_ERAY} for a typical configuration

The minimum required f_{CLC_ERAY} frequency for various assuming the following typical ERAY configuration:

- A maximum payload length of 32 bytes/8 four-byte-words

- A minimum minislot length of 7 µs
- A maximum 128 configured Message Buffers
- Dual channels in use

The minimum f_{CLC_ERAY} frequency for this typical example would be 10 MHz.

23.3.12.4 FlexRay™ protocol controller access to message RAM

The two Transient Buffer RAMs (TBF 1, TBF 2) are used to buffer the data for transfer between the two FlexRay™ Protocol Controllers and the Message RAM.

Each Transient Buffer RAM is build up as a double buffer, able to store two complete FlexRay™ messages. There is always one buffer assigned to the corresponding Protocol Controller while the other one is accessible by the Message Handler.

If for example the Message Handler writes the next message to be send to Transient Buffer Tx, the FlexRay™ Channel Protocol Controller can access Transient Buffer Rx to store the message it is actually receiving. During transmission of the message stored in Transient Buffer Tx, the Message Handler transfers the last received message stored in Transient Buffer Rx to the Message RAM (if it passes acceptance filtering) and updates the respective Message Buffer.

Data transfers between the Transient Buffer RAMs and the shift registers of the FlexRay™ Channel Protocol Controllers are done in words of 32 bit. This enables the use of a 32 bit shift register independent of the length of the FlexRay™ messages.

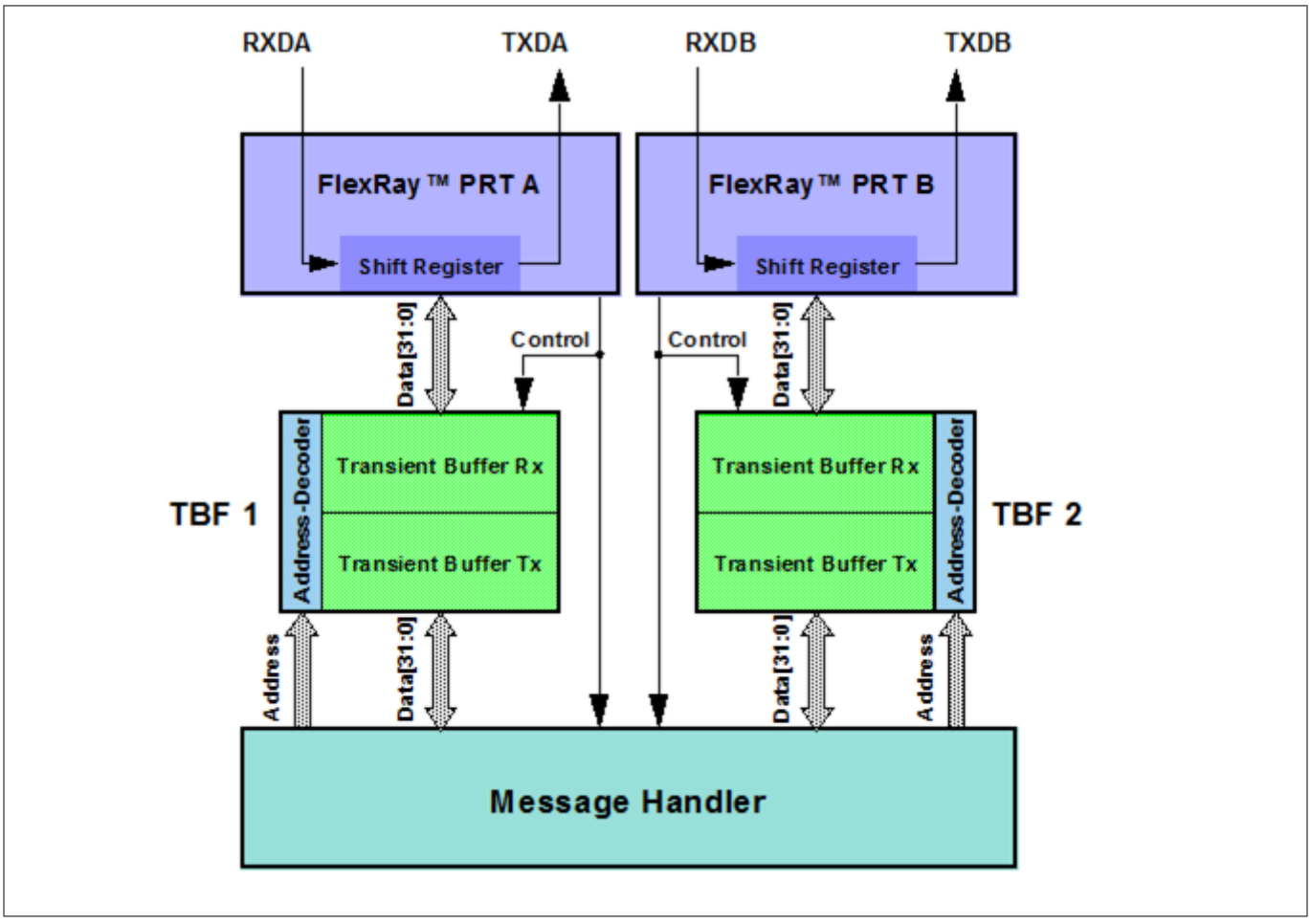


Figure 450 Access to transient buffer RAMs

23.3.13 Message RAM

To avoid conflicts between Host access to the Message RAM and FlexRay™ message reception or transmission, the Host cannot directly access the Message Buffers in the Message RAM. These accesses are handled through the Input and Output Buffers. The Message RAM is able to store up to 128 Message Buffers depending on the configured payload length.

The Message RAM is organized 2048 x 32-bit words. To achieve the required flexibility with respect to different numbers of data byte per FlexRay™ Frame (0 to 254), the Message RAM has a structure as shown in the following figure.

The Data Partition is allowed to start at Message RAM word number: $(\mathrm{MRC.LCB} + 1) \times 4$

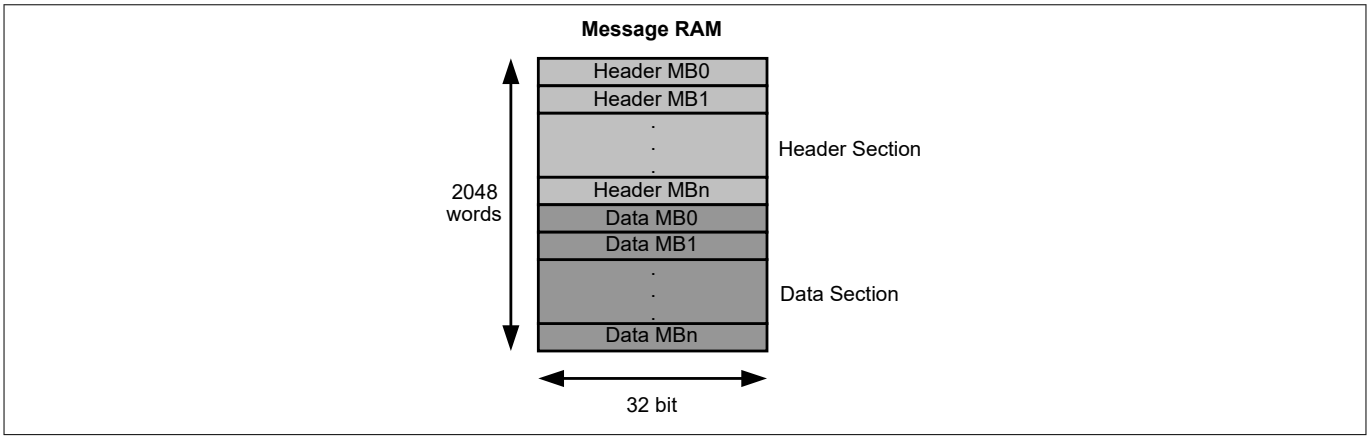


Figure 451 Structure of message RAM

Header partition

Stores Header Segments of FlexRay™ Frames:

- Supports a maximum of 128 Message Buffers
- Each Message Buffer has a Header of four 32 bit words
- Header 3 (Word 2) of each Message Buffer holds the 11 bit pointer to the respective Data Section in the Data Partition

Data partition

Flexible storage of Data Sections with different length. Some maximum values are:

- 30 Message Buffers with 254 byte Data Section each, or
- 56 Message Buffers with 128 byte Data Section each, or
- 128 Message Buffers with 48 byte Data Section each

Restriction: Header Partition + Data Partition may not occupy more than 2048 32-bit words.

23.3.13.1 Header partition

The Header of each Message Buffer occupies four 32-bit words in the Header Partition of the Message RAM. The Header of Message Buffer 0 starts with the first word in the Message RAM.

For transmit buffers the Header CRC has to be calculated by the Host.

Payload Length Received PLR, Receive Cycle Count RCC, Received on Channel Indication RCI, Startup Frame Indication bit SFI, Sync bit SYN, NULL Frame Indication bit NFI, Payload Preamble Indication bit PPI, and Reserved bit RES are only updated from received valid Frames (including valid NULL Frames).

Header word 4 of each configured Message Buffer holds the respective Message Buffer Status MBS information.

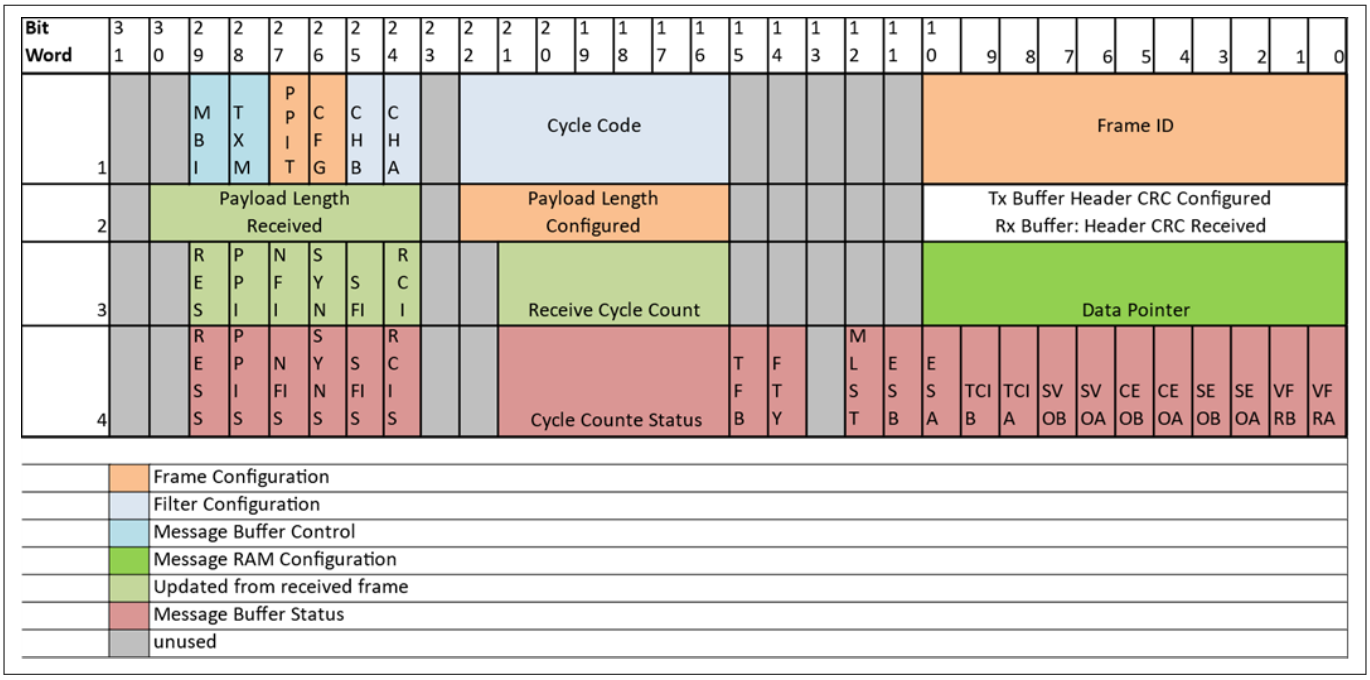


Figure 452 Header section of a message buffer in the message RAM

Header 1 (Word 0)

Write access through WRHS1, read access through RDHS1:

- Frame ID: Slot counter filtering configuration
- Cycle Code: Cycle counter filtering configuration
- CHA, CHB: Channel filtering configuration
- CFG: Message Buffer configuration: receive/transmit
- PPIT: Payload Preamble Indicator Transmit
- TXM: Transmit mode configuration: single-shot/continuous
- MBI: Message Buffer receive/transmit service request enable

Header 2 (Word 1)

Write access through WRHS2, read access through RDHS2:

- Header CRC
  - Transmit Buffer: Configured by the Host (calculated from Frame Header Segment)
  - Receive Buffer: Updated from received Frame

- Payload Length Configured
  - Length of Data Section (2-byte words) as configured by the Host
- Payload Length Received
  - Length of Payload Segment (2-byte words) stored from received Frame

Header 3 (Word 2)

Write access through WRHS3, read access through RDHS3:
- Data Pointer
  - Pointer to the beginning of the corresponding Data Section in the Data Partition

Read access through RDHS3, valid for receive buffers only, updated from received Frames:
- Receive Cycle Count: Cycle count from received Frame
- RCI: Received on Channel Indicator
- SFI: Start-up Frame Indicator
- SYN: SYNC Frame Indicator
- NFI: NULL Frame Indicator
- PPI: Payload Preamble Indicator
- RES: Reserved bit

Message buffer status MBS (Word 3)

Read access through MBS, updated by the Communication Controller at the end of the configured slot.
- VFRA: Valid Frame Received on channel A
- VFRB: Valid Frame Received on channel B
- SEOA: Syntax Error Observed on channel A
- SEOB: Syntax Error Observed on channel B
- CEOA: Content Error Observed on channel A
- CEOB: Content Error Observed on channel B
- SVOA: Slot boundary Violation Observed on channel A
- SVOB: Slot boundary Violation Observed on channel B
- TCIA: Transmission Conflict Indication channel A
- TCIB: Transmission Conflict Indication channel B
- ESA: Empty Slot Channel A
- ESB: Empty Slot Channel B
- MLST: Message LoST
- FTA: Frame Transmitted on Channel A
- FTA: Frame Transmitted on Channel B
- Cycle Count Status: Actual cycle count when status was updated
- RCIS: Received on CHannel Indicator Status
- SFIS: Startup Frame Indicator Status
- SYNS: SYNC Frame Indicator Status
- NFIS: NULL Frame Indicator Status
- PPIS: Payload Preamble Indicator Status
- RESS: Reserved Bit Status

23.3.13.2 Data partition

The Data Partition of the Message RAM stores the Data Sections of the Message Buffers configured for reception / transmission as defined in the Header Partition. The number of data bytes for each Message Buffer

can vary from 0 to 254. To optimize the data transfer between the shift registers of the two FlexRay™ Protocol Controllers and the Message RAM as well as between the Host interface and the Message RAM, the physical width of the Message RAM is set to 4 bytes.

The Data Partition starts after the last word of the Header Partition. When configuring the Message Buffers in the Message RAM the programmer has to assure that the data pointers point to addresses within the Data Partition.

The table below shows an example how the Data Sections of the configured Message Buffers can be stored in the Data Partition of the Message RAM.

The beginning and the end of a Message Buffer's Data Section is determined by the data pointer and the payload length configured in the Message Buffer's Header Section, respectively. This enables a flexible usage of the available RAM space for storage of Message Buffers with different data length.

If the size of the Data Section is an odd number of 2-byte words, the remaining 16 bits in the last 32-bit word are unused.

Table 1091 Example for structure of the data section in the message RAM

Bit Word	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
…	unused								unused								unused								unused							
…	unused								unused								unused								unused							
…	MBn Data3								MBn Data2								MBn Data1								MBn Data0							
…	…								…								…								…							
…	…								…								…								…							
…	MBn Data(m)								MBn Data(m-1)								MBn Data(m-2)								MBn Data(m-3)							
…	…								…								…								…							
…	…								…								…								…							
…	…								…								…								…							
…	MB1 Data3								MB1 Data2								MB1 Data1								MB1 Data0							
…	…								…								…								…							
…	MB1 Data(k)								MB1 Data(k-1)								MB1 Data(k-2)								MB1 Data(k-3)							
2046	MB0 Data3								MB0 Data2								MB0 Data2								MB0 Data0							
2047	unused								unused								MB0 Data5								MB0 Data4							

23.3.13.3 ECC check

There is an ECC checking mechanism implemented to assure the integrity of the data stored in the seven RAM blocks of the module. The RAM blocks have an ECC generator or checker attached as shown in the following figure:

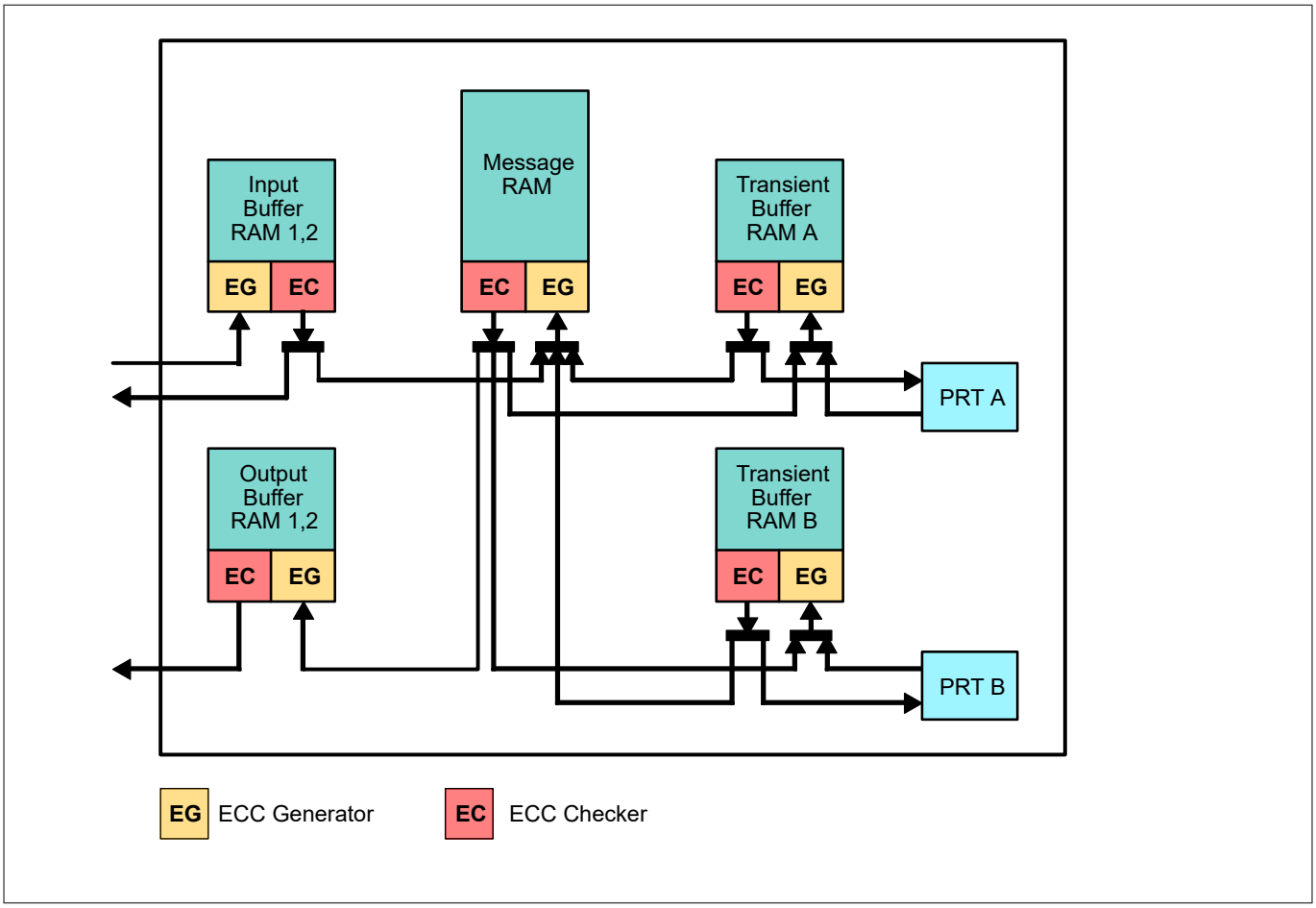


Figure 453 ECC generation and check

When data is written to a RAM block, the local ECC generator generates the ECC data. The ECC data is stored together with the respective data word. The ECC data is checked each time a data word is read from any of the RAM blocks.

If an ECC error is detected, the respective error flag is set. The ECC error flags MHDS.EIBF, MHDS.EOBF, MHDS.EMR, MHDS.ETBF1, MHDS.ETBF2, and the faulty Message Buffer indicators MHDS.FMBD, MHDS.MFMB, MHDS.FMB are located in the Message Handler Status register. These error flags control the error interrupt flag EIR.EERR.

When an ECC error has been detected the following actions will be performed:

In all cases

- The respective ECC error flag in the Message Handler Status MHDS register is set
- The ECC error flag EIR.EERR in the Error Service Request register is set, and if enabled, a module service request to the Host will be generated

Additionally in specific cases

1. ECC error in data transfer from Input Buffer RAM 1, 2 ⇒ Message RAM (Transfer of Header and Data Section)

a. MHDS.EIBF bit is set

b. MHDS.FMBD bit is set to indicate that MHDS.FMB has been updated

c. MHDS.FMB indicates the number of the faulty Message Buffer

d. Transmit buffer: Transmission request for the respective Message Buffer is not set

2. ECC error in data transfer from Input Buffer RAM 1, 2 ⇒ Message RAM (Transfer of Data Section only)
   a. MHDS.EMR bit is set
   b. MHDS.FMBD bit is set to indicate that MHDS.FMB points to a faulty Message Buffer
   c. MHDS.FMB indicates the number of the faulty Message Buffer
   d. The Data Section of the respective Message Buffer is not updated
   e. Transmit buffer: Transmission request for the respective Message Buffer is not set
3. ECC error during host reading Input Buffer RAM
   a. MHDS.EIBF bit is set
4. ECC error during scan of Header Sections in Message RAM
   a. MHDS.EMR bit is set
   b. MHDS.FMBD bit is set to indicate that MHDS.FMB points to a faulty Message Buffer
   c. MHDS.FMB indicates the number of the faulty Message Buffer
   d. Ignore Message Buffer (Message Buffer is skipped)
5. ECC error during data transfer from Message RAM ⇒ Transient Buffer RAM A, B
   a. MHDS.EMR bit is set
   b. MHDS.FMBD bit is set to indicate that MHDS.FMB points to a faulty Message Buffer
   c. MHDS.FMB indicates the number of the faulty Message Buffer
   d. Frame not transmitted, Frames already in transmission are invalidated by setting the Frame CRC to zero
6. ECC error during data transfer from Transient Buffer RAM A, B ⇒ Protocol Controller 1, 2
   a. MHDS.ETBF1, MHDS.ETBF2 bit is set
7. ECC error in data transfer from Transient Buffer RAM A, B ⇒ Message RAM (ECC error when reading Header Section of respective Message Buffer from Message RAM)
   a. MHDS.EMR bit is set
   b. MHDS.FMBD bit is set to indicate that MHDS.FMB points to a faulty Message Buffer
   c. MHDS.FMB indicates the number of the faulty Message Buffer
   d. The Data Section of the respective Message Buffer is not updated
8. ECC error in data transfer from Transient Buffer RAM A, B ⇒ Message RAM (ECC error when reading Transient Buffer RAM A, B)
   a. MHDS.ETBF1, MHDS.ETBF2 bit is set
   b. MHDS.FMBD bit is set to indicate that MHDS.FMB points to a faulty Message Buffer
   c. MHDS.FMB indicates the number of the faulty Message Buffer
9. ECC error during data transfer from Message RAM ⇒ Output Buffer RAM
   a. MHDS.EMR bit is set
   b. MHDS.FMBD bit is set to indicate that MHDS.FMB points to a faulty Message Buffer
   c. MHDS.FMB indicates the number of the faulty Message Buffer
10. ECC error during Host reading Output Buffer RAM
    a. MHDS.EOBF bit is set
11. ECC error during data read of Transient Buffer RAM A, B

If an ECC error occurs while the Message Handler reads a Frame with Network Management information (PPI = 1) from the Transient Buffer RAM A, B the corresponding Network Management vector registers NMV1 to NMV3 are not updated from that Frame.

23.3.13.4 Assignment of message buffers

The assignment of the Message Buffers is done according to the scheme shown in the table below. The number N of available Message Buffers depends on the payload length of the configured Message Buffers. The maximum number of Message Buffers is 128. The maximum payload length supported is 254 byte.

The Message Buffers are separated into three consecutive groups:

- Static Buffers: Transmit or Receive Buffers assigned to static segment

- Static and Dynamic Buffers: Transmit or Receive Buffers assigned to static or dynamic segment
- FIFO- Receive FIFO

Table 1092 Assignment of message buffers

Message Buffer 0	⇓ Static Buffers	
Message Buffer 1		
…		
	⇓ Static + Dynamic Buffers	⇐ FDB
	⇓ FIFO	⇐ FFB
Message Buffer N-1		
Message Buffer N		⇐ LCB

The Message Buffer separation configuration can be changed only in “DEFAULT_CONFIG” or “CONFIG” state by programming the Message RAM Configuration register (MRC).

The first group starts with Message Buffer 0 and consists of static Message Buffers only. Message Buffer 0 is dedicated to hold the startup and SYNC Frame or the single slot Frame, if node transmit one, as configured by SUCC1.TXST, SUCC1.TXSY and SUCC1.TSM in the SUC Configuration Register 1 (SUCC1). Message Buffer 1 may also be used for SYNC Frame transmission in case the SYNC Frames or single-slot Frames should have different payloads on the two channels. In this case bit MRC.SPLM has to be programmed to 1 and Message Buffers 0 and 1 have to be configured with the key slot ID and can be (re)configured in “DEFAULT_CONFIG” or “CONFIG” state only.

The second group consists of Message Buffers assigned to the static or to the dynamic segment. Message Buffers belonging to this group may be reconfigured during run-time from dynamic to static or vice-versa depending on the state of MRC.SEC.

The Message Buffers belonging to the third group are concatenated to a single receive FIFO.

23.3.14 Host handling of errors

An ECC error caused by transient bit flips can be fixed by:

23.3.14.1 Self-healing

ECC errors located in

- Input Buffer RAM 1, 2
- Output Buffer RAM 1, 2
- Data Section of Message RAM
- Transient Buffer RAM A
- Transient Buffer RAM B

are overwritten with the next write access to the disturbed bit(s) caused by Host access or by FlexRay™ communication.

23.3.14.2 CLEAR_RAMS command

After Power-On Reset, the ERAY RAMs hold arbitrary values which causes ECC errors (MHDS) when a read operation is performed in a ERAY RAM location. For this reason the ERAY RAMs should be initialized always after a Power-On reset. When called in DEFAULT_CONFIG or CONFIG state POC command CLEAR_RAMS initializes all module-internal RAMs to zero. In order to perform a safe initialization of RAM blocks, the following programming sequence is suggested:

1. Enable the clock in the CLC register
2. Read the CLC register back to ensure that the clocks are enabled
3. Set the PROT state register to the required one (for example Run state)
4. Enable the test mode. Take care of the unlock sequence. Refer to description of LCK.TMK and TEST1.WRTEN
5. Check if CCSV.POCS is either 0x0 (DEFAULT_CONFIG) or 0xF (CONFIG). If not in any of these states, perform the according command to get to CONFIG state
6. Check if SUCC1.PBSY is equal to 0x0. If 0x1 wait until 0x0
7. Set SUCC1.CMD to 0xC meaning that the CLEAR_RAMS command is entered
8. Read SUCC1.CMD. If 0x0 the command has not been accepted. Repeat up from step 6. Otherwise continue
9. Wait until the status flag MHDS.CRAM is not cleared (or) wait for 1024 module cycles
10. Enable RAM Test mode: TEST1.TMC = 01b. This mode enables access of all RAM blocks in ERAY module to host
11. CUST1.IBF1PAG = 1b
12. CUST1.IBF2PAG = 1b
13. Repeat steps 6 to 9 and then proceed to step 14
14. CUST1.IBF1PAG = 0b
15. CUST1.IBF2PAG = 0b
16. Read at least one address in all the ERAY RAM blocks
17. Switch off Test Mode: TEST1.TMC = 0b and TEST1.WRTEN = 0b
18. Clear ECC error flags in MHDS and EIR registers

23.3.14.3 Temporary unlocking of header section

An ECC error in the header section of a locked message buffer can be fixed by a transfer from the Input Buffer to the locked buffer Header Section. For this transfer, the write-access to the IBCR (specifying the message buffer number) must be immediately preceded by the unlock sequence normally used to leave CONFIG state (see “Lock Register”). For that single transfer the respective message buffer header is unlocked, regardless whether it belongs to the FIFO or whether its locking is controlled by MRC.SEC[1:0], and will be updated with new data.

23.3.15 Module service request

In general, service requests provide a close link to the protocol timing as they are triggered almost immediately when an error or status change is detected by the controller, a Frame is received or transmitted, a configured timer service request is activated, or a stop watch event occurred. This enables the Host to react very quickly on specific error conditions, status changes, or timer events. On the other hand too many service requests can cause the Host to miss deadlines required for the application. Therefore the Communication Controller supports disable and enable controls for each individual service request source separately.

An service request may be triggered when:

- An error was detected
- A status flag is set
- A timer reaches a per-configured value

- A message transfer from Input Buffer to Message RAM or from Message RAM to Output Buffer has completed
- A stop watch event occurred

Tracking status and generating service requests when a status change or an error occurs are two independent tasks. Regardless of whether an service request is enabled or not, the corresponding status is tracked and indicated by the Communication Controller. The Host has access to the actual status and error information by reading the Error Service Request Register EIR and the Status Service Request SIR Register.

Table 1093 Module service request flags and service request line enable

Register	Bit	Function
SIR	WST	Wakeup Status
	CAS	Collision Avoidance Symbol
	CYCS	Cycle Start Service Request
	TXI	Transmit Service Request
	RXI	Receive Service Request
	RFNE	Receive FIFO not Empty
	RFCL	Receive FIFO Critical Level
	NMVC	Network Management Vector Changed
	TI0	Timer Service Request 0
	TI1	Timer Service Request 1
	TIBC	Transfer Input Buffer Completed
	TOBC	Transfer Output Buffer Completed
	SWE	Stop Watch Event
	SUCS	Startup Completed Successfully
	MBSI	Message Buffer Status Interrupt
	SDS	Start of Dynamic Segment
	WUPA	Wakeup Pattern Channel A
	MTSA	MTS Received on Channel A
	WUPB	Wakeup Pattern Channel B
	MTSB	MTS Received on Channel B
ILE	EINT0	Enable Service Request Line 0
	EINT1	Enable Service Request Line 1
EIR	PEMC	Protocol Error Mode Changed
	CNA	Command Not Valid
	SFBM	SYNC Frames Below Minimum
	SFO	SYNC Frame Overflow
	CCF	Clock Correction Failure
	CCL	CHI Command Locked

(table continues...)

Table 1093 (continued) Module service request flags and service request line enable

Register	Bit	Function
	EERR	ECC Error
	RFO	Receive FIFO Overrun
	EFA	Empty FIFO Access
	IIBA	Illegal Input Buffer Access
	IOBA	Illegal Output Buffer Access
	MHF	Message Handler Constraints Flag
	EDA	Error Detected on Channel A
	LTVA	Latest Transmit Violation Channel A
	TABA	Transmission Across Boundary Channel A
	EDB	Error Detected on Channel B
	LTVB	Latest Transmit Violation Channel B
	TABB	Transmission Across Boundary Channel B

The interrupt lines to the Host INT0SRC and INT1SRC are controlled by the enabled interrupts. In addition each of the two interrupt lines can be enabled / disabled separately by programming bit ILE.EINT0/INT0SRC.SRE and ILE.EINT1/ INT1SRC.SRE.

The interrupt lines to the HostNDAT0SR and NDAT1SR are controlled by the enabled new data interrupts (NDIC1 to NDIC4). In addition each of the two interrupt lines can be enabled / disabled separately by programming bit NDAT0SRC.SRE and NDAT1SRC.SRE.

The interrupt lines to the Host MBSC0SR and MBSC1SR are controlled by the enabled message buffer status changed interrupts (MSIC1 to MSIC4). In addition each of the two interrupt lines can be enabled / disabled separately by programming bit MBSC0SRC.SRE and MBSC1SRC.SRE.

The two timer service requests generated by service request timer 0 and 1 are available on pins TINT0SR and TINT1SR. They can be configured through the Timer 0 and Timer 1 Configuration register. In addition each of the two interrupt lines can be enabled and disabled separately by programming bit TINT0SRC.SRE and TINT1SRC.SRE.

A stop watch event may be triggered through input pin STPWn.

The status of the data transfer between IBF and OBF and the Message RAM is signaled on signals IBUSY and OBUSY. When a transfer has completed bit SIR.TIBC or SIR.TOBC is set.

23.3.16 Restrictions

The following restrictions must be considered when programming the module. A violation of these restrictions may lead to an erroneous behavior.

23.3.16.1 Message buffers with the same frame ID

If two or more Message Buffers are configured with the same Frame ID, and if they have a matching cycle counter filter value for the same slot, then the Message Buffer with the lowest Message Buffer number is used.

Sharing of a static time slot through cycle counter filtering between different nodes of a FlexRay™ network is not allowed.

23.3.16.2 Data transfers between IBF, OBF and message RAM

The time required to transfer the contents of a Message Buffer between IBF, OBF and Message RAM depends on

- The setup time to start the first transfer
- The number of 4-byte words to be transferred
- The number of concurrent tasks to be managed by the Message Handler

The number of 4-byte words varies from 4 (Header Section only) to 68 (Header + maximum Data Section) while the number of concurrent task varies from one to three.

The following concurrent tasks are executed under control of the Message Handler:

- Data transfer between IBF or OBF and Message RAM
- Data transfer between TBF1 and Message RAM, search next TX/RX Message Buffer CHA
- Data transfer between TBF2 and Message RAM, search next TX/RX Message Buffer CHB

Transfers between IBF and Message RAM respectively Message RAM and OBF can only be handled one after another. In case that for example a transfer between IBF and Message RAM has been started shortly before a transfer between Message RAM and OBF is requested, the OBF transfer has to wait until the IBF transfer has completed.

The relative time is measured in f_{CLC_ERAY} cycles. Absolute time depends on the actual f_{CLC_ERAY} cycle period.

- $\text{cyclestrans} = (\text{remaining cyclesof transfer running}) + (\text{cycles of requested transfer})$,
- $\text{cyclestrans} = \text{cyclesrem} + \text{cyclesreq}$,
- $\text{cyclesrem} = (\text{number ofconcurrent tasks}) \times (\text{setuptime} + (\text{number of 4_byte words})\text{rem})$,
- $\text{cyclesreq} = (\text{number of concurrent tasks}) \times (\text{setup time} + (\text{number of 4_byte words})\text{req})$,
- setup time = 2 f_{CLC_ERAY} cycles

Under worst case conditions a transfer is requested directly after the previous transfer started:

Maximum number of f_{CLC_ERAY} cycles:

- $\text{cyclestrans} = (3 \times (2 + 68)) + (3 \times (2 + 68)) = 420$

Worst case timing:

- $\text{timetrans}(40\ \text{MHz}) = 420 \times 25\ \text{ns} = 10.5\ \mu s$

23.4 Registers

23.4.1 Register overview - access mode glossary

Table 1094 Register overview - access mode glossary

Keyword	Description
E	Access protection using PROT register PROTE.
SE	Access protection using PROT register PROTSE.
APU-P	Protection group consisting of registers ACCEN_WRA, ACCEN_WRB, ACCEN_RDA, ACCEN_RDB, ACCEN_VM, ACCEN_PRS.
P	Access protection using *APU-P* registers.
BE	Always returns a Bus Error.
nBE	Indicates that no Bus Error is generated when accessing this address range, even though it is either an access to an undefined address or the access does not follow the given rules.
U	No access restrictions.
PROT	Access restrictions as defined in the PROT register access rules.

23.4.2 Registers overview - ERAY (ascending offset address)

Table 1095 Registers overview - ERAY (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CLC	Clock Control Register	0000_H	P	P, SV, E	Application Reset	4488
CUST1	Busy and Input Buffer Control Register	0004_H	P	P	Kernel Reset	4488
ID	Module Identification Register	0008_H	P	BE	PowerOn Reset	4490
CUST3	Customer Interface Timeout Counter Register	$000C_H$	P	P	Kernel Reset	4491
TEST1	Test Register 1	0010_H	P	P	Kernel Reset	4492
TEST2	Test Register 2	0014_H	P	P	Kernel Reset	4496
LCK	Lock Register	$001C_H$	P	P	Kernel Reset	4498
EIR	Error Service Request Select Register	0020_H	P	P	Kernel Reset	4499
SIR	Status Service Request Register	0024_H	P	P	Kernel Reset	4502
EILS	Error Service Request Line Select	0028_H	P	P	Kernel Reset	4506
SILS	Status Service Request Line Select	$002C_H$	P	P	Kernel Reset	4508
EIES	Error Service Request Enable Set	0030_H	P	P	Kernel Reset	4510

(table continues...)

Table 1095 (continued) Registers overview - ERAY (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
EIER	Error Service Request Enable Reset	0034_H	P	P	Kernel Reset	4513
SIES	Status Service Request Enable Set	0038_H	P	P	Kernel Reset	4516
SIER	Status Service Request Enable Reset	$003C_H$	P	P	Kernel Reset	4518
ILE	Service Request Line Enable	0040_H	P	P	Kernel Reset	4521
T0C	Timer 0 Configuration	0044_H	P	P	Kernel Reset	4522
T1C	Timer 1 Configuration	0048_H	P	P	Kernel Reset	4523
STPW1	Stop Watch Register 1	$004C_H$	P	P	Kernel Reset	4524
STPW2	Stop Watch Register 2	0050_H	P	P	Kernel Reset	4525
SUCC1	SUC Configuration Register 1	0080_H	P	P	Kernel Reset	4526
SUCC2	SUC Configuration Register 2	0084_H	P	P	Kernel Reset	4531
SUCC3	SUC Configuration Register 3	0088_H	P	P	Kernel Reset	4532
NEMC	NEM Configuration Register	$008C_H$	P	P	Kernel Reset	4533
PRTC1	PRT Configuration Register 1	0090_H	P	P	Kernel Reset	4533
PRTC2	PRT Configuration Register 2	0094_H	P	P	Kernel Reset	4534
MHDC	MHD Configuration Register	0098_H	P	P	Kernel Reset	4535
GTUC01	GTU Configuration Register 1	$00A0_H$	P	P	Kernel Reset	4536
GTUC02	GTU Configuration Register 2	$00A4_H$	P	P	Kernel Reset	4536
GTUC03	GTU Configuration Register 3	$00A8_H$	P	P	Kernel Reset	4537
GTUC04	GTU Configuration Register 4	$00AC_H$	P	P	Kernel Reset	4538
GTUC05	GTU Configuration Register 5	$00B0_H$	P	P	Kernel Reset	4539
GTUC06	GTU Configuration Register 6	$00B4_H$	P	P	Kernel Reset	4540
GTUC07	GTU Configuration Register 7	$00B8_H$	P	P	Kernel Reset	4540
GTUC08	GTU Configuration Register 8	$00BC_H$	P	P	Kernel Reset	4541

(table continues...)

Table 1095 (continued) Registers overview - ERAY (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
GTUC09	GTU Configuration Register 9	$00C0_H$	P	P	Kernel Reset	4541
GTUC10	GTU Configuration Register 10	$00C4_H$	P	P	Kernel Reset	4542
GTUC11	GTU Configuration Register 11	$00C8_H$	P	P	Kernel Reset	4543
CCSV	Communication Controller Status Vector	0100_H	P	BE	Kernel Reset	4544
CCEV	Communication Controller Error Vector	0104_H	P	BE	Kernel Reset	4547
SCV	Slot Counter Value	0110_H	P	BE	Kernel Reset	4548
MTCCV	Macrotick and Cycle Counter Value	0114_H	P	BE	Kernel Reset	4549
RCV	Rate Correction Value	0118_H	P	BE	Kernel Reset	4549
OCV	Offset Correction Value	$011C_H$	P	BE	Kernel Reset	4550
SFS	SYNC Frame Status	0120_H	P	BE	Kernel Reset	4550
SWNIT	Symbol Window and Network Idle Time Status	0124_H	P	BE	Kernel Reset	4552
ACS	Aggregated Channel Status	0128_H	P	P	Kernel Reset	4554
ESIDn	Even Sync ID Symbol Window n	0130_H+ (n-1)*4	P	BE	Kernel Reset	4556
OSIDn	Odd Sync ID Symbol Window n	0170_H+ (n-1)*4	P	BE	Kernel Reset	4557
NMVx	Network Management Vector x	$01B0_H$+ (x-1)*4	P	BE	Kernel Reset	4558
MRC	Message RAM Configuration	0300_H	P	P	Kernel Reset	4559
FRF	FIFO Rejection Filter	0304_H	P	P	Kernel Reset	4561
FRFM	FIFO Rejection Filter Mask	0308_H	P	P	Kernel Reset	4562
FCL	FIFO Critical Level	$030C_H$	P	P	Kernel Reset	4563
MHDS	Message Handler Status	0310_H	P	P	Kernel Reset	4564
LDTS	Last Dynamic Transmit Slot	0314_H	P	P	Kernel Reset	4565
FSR	FIFO Status Register	0318_H	P	P	Kernel Reset	4566
MHDF	Message Handler Constraints Flags	$031C_H$	P	P	Kernel Reset	4567
TXRQ1	Transmission Request Register 1	0320_H	P	BE	Kernel Reset	4569
TXRQ2	Transmission Request Register 2	0324_H	P	BE	Kernel Reset	4570

(table continues...)

Table 1095 (continued) Registers overview - ERAY (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
TXRQ3	Transmission Request Register 3	0328_H	P	BE	Kernel Reset	4570
TXRQ4	Transmission Request Register 4	$032C_H$	P	BE	Kernel Reset	4571
NDAT1	New Data Register 1	0330_H	P	BE	Kernel Reset	4571
NDAT2	New Data Register 2	0334_H	P	BE	Kernel Reset	4572
NDAT3	New Data Register 3	0338_H	P	BE	Kernel Reset	4572
NDAT4	New Data Register 4	$033C_H$	P	BE	Kernel Reset	4573
MBSC1	Message Buffer Status Changed 1	0340_H	P	BE	Kernel Reset	4574
MBSC2	Message Buffer Status Changed 2	0344_H	P	BE	Kernel Reset	4574
MBSC3	Message Buffer Status Changed 3	0348_H	P	BE	Kernel Reset	4575
MBSC4	Message Buffer Status Changed 4	$034C_H$	P	BE	Kernel Reset	4575
NDIC1	New Data Interrupt Control 1	$03A8_H$	P	P	Kernel Reset	4576
NDIC2	New Data Interrupt Control 2	$03AC_H$	P	P	Kernel Reset	4577
NDIC3	New Data Interrupt Control 3	$03B0_H$	P	P	Kernel Reset	4577
NDIC4	New Data Interrupt Control 4	$03B4_H$	P	P	Kernel Reset	4578
MSIC1	Message Buffer Status Changed Interrupt Control 1	$03B8_H$	P	P	Kernel Reset	4578
MSIC2	Message Buffer Status Changed Interrupt Control 2	$03BC_H$	P	P	Kernel Reset	4579
MSIC3	Message Buffer Status Changed Interrupt Control 3	$03C0_H$	P	P	Kernel Reset	4580
MSIC4	Message Buffer Status Changed Interrupt Control 4	$03C4_H$	P	P	Kernel Reset	4580
CREL	Core Release Register	$03F0_H$	P	nBE	Kernel Reset	4581
ENDN	Endian Register	$03F4_H$	P	nBE	Kernel Reset	4582
WRDSn	Write Data Section n	0400_H+ (n-1)*4	P	P	Kernel Reset	4583
WRHS1	Write Header Section 1	0500_H	P	P	Kernel Reset	4584

(table continues...)

Table 1095 (continued) Registers overview - ERAY (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
WRHS2	Write Header Section 2	0504_H	P	P	Kernel Reset	4585
WRHS3	Write Header Section 3	0508_H	P	P	Kernel Reset	4586
IBCM	Input Buffer Command Mask	0510_H	P	P	Kernel Reset	4587
IBCR	Input Buffer Command Request	0514_H	P	P	Kernel Reset	4588
RDDSn	Read Data Section n	0600_H+ (n-1)*4	P	P	Kernel Reset	4589
RDHS1	Read Header Section 1	0700_H	P	P	Kernel Reset	4590
RDHS2	Read Header Section 2	0704_H	P	P	Kernel Reset	4591
RDHS3	Read Header Section 3	0708_H	P	P	Kernel Reset	4592
MBS	Message Buffer Status	$070C_H$	P	P	Kernel Reset	4593
OBCM	Output Buffer Command Mask	0710_H	P	P	Kernel Reset	4598
OBCR	Output Buffer Command Request	0714_H	P	P	Kernel Reset	4599
OTSS	OCDS Trigger Set Select	0870_H	P	SV, E, P	Debug Reset	4600
OCS	OCDS Control and Status Register	$08E8_H$	P	SV, P	Debug Reset	4601
RST_CTRLA	Reset Control Register A	$08EC_H$	P	P, SV, E	Application Reset	4601
RST_CTRLB	Reset Control Register B	$08F0_H$	P	P, SV, E	Application Reset	4602
RST_STAT	Reset Status Register	$08F4_H$	P	BE	Application Reset	4603
PROTE	PROT Register Endinit	0900_H	U	SV, PROT	Application Reset	4603
PROTSE	PROT Register Safe Endinit	0904_H	U	SV, PROT	Application Reset	4605
ACCEN_WRA	Write access enable register A	0920_H	U	SE, SV	Application Reset	4607
ACCEN_WRB	Write access enable register B	0924_H	U	SE, SV	Application Reset	4607
ACCEN_RDA	Read access enable register A	0928_H	U	SE, SV	Application Reset	4608
ACCEN_RDB	Read access enable register B	$092C_H$	U	SE, SV	Application Reset	4608
ACCEN_VM	VM access enable register	0930_H	U	SE, SV	Application Reset	4609
ACCEN_PRS	PRS access enable register	0934_H	U	SE, SV	Application Reset	4609

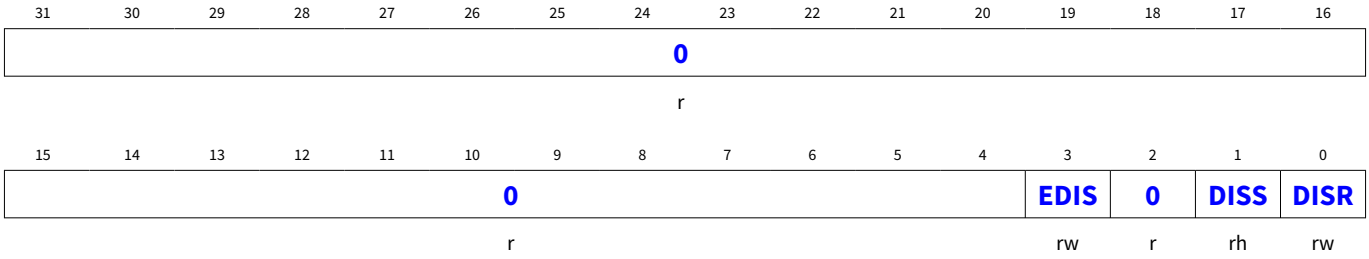
23.4.3 Clock Control Register

The Clock Control Register CLC allows the programmer to adapt the functionality and power consumption of the module to the requirements of the application.

Register CLC controls the module clock signal and the reactivity to the sleep signal.

CLC Offset address: 0000_H

Clock Control Register Application Reset value: $0000\ 0003_H$



Field	Bits	Type	Description
DISR	0	rw	**Module Disable Request Bit** Used for enable/disable control of the module. 0_B On request: enable the module clock 1_B Off request: stop the module clock
DISS	1	rh	**Module Disable Status Bit** 0_B Module clock is enabled 1_B Off: module is not clocked
EDIS	3	rw	**Sleep Mode Enable Control** Used to control the module's reaction to sleep mode. 0_B Sleep mode request is enabled and functional 1_B Module disregards the sleep mode control signal
0	2, 31:4	r	**Reserved** Read as 0; should be written with 0.

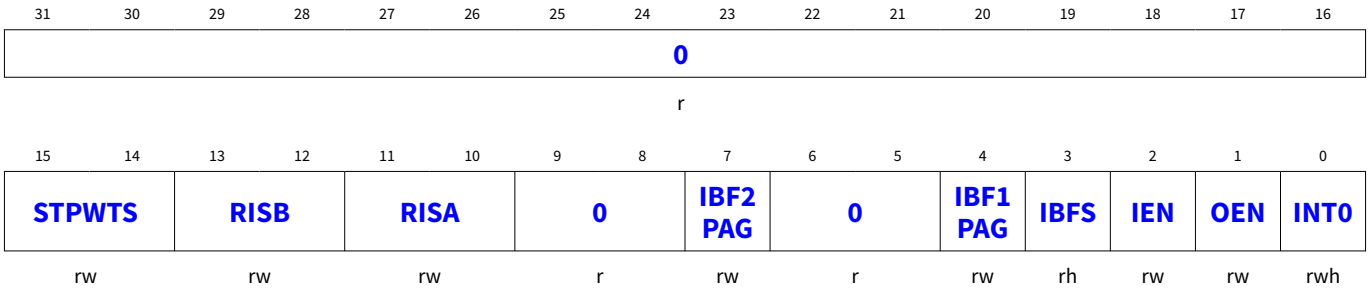
23.4.4 Busy and Input Buffer Control Register

The Busy Control Register enables the automatic delay scheme. Furthermore it signals a time-out service request for the automatic delay scheme. IBFS changes 2 clock cycles after any reset to '1'. Thus the application will read this bit as '1' already at first access after any reset.

***Note**: LDMST or SWAPMSK.W instructions should be used only with bit mask enabled for all ´rwh´ bits in this register.*

CUST1 Offset address: 0004_H

Busy and Input Buffer Control Register Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
INT0	0	rwh	**CIF Timeout Service Request Status** INT0 will be set if a timeout has occurred during the auto delay scheme and must be reset by writing zero to INT0. Software can also set this bit field. ***Note**: In case hardware sets INT0 and at the same point of time software clears INT0, INT0 is cleared.*
OEN	1	rw	**Enable auto delay scheme for Output Buffer Control Register (OBCR)** This control bit controls the delay scheme for Output Buffer Control Register (OBCR) read accesses. 0_B Disable auto delay scheme for Output Buffer Control Register (OBCR) 1_B Enable auto delay scheme for Output Buffer Control Register (OBCR)
IEN	2	rw	**Enable auto delay scheme for Input Buffer Control Register (IBCR)** This control bit controls the auto delay scheme for Input Buffer Control Register (IBCR) read accesses. 0_B Disable auto delay scheme for Input Buffer Control Register (IBCR) 1_B Enable auto delay scheme for Input Buffer Control Register (IBCR)
IBFS	3	rh	**Input Buffer Status Register** This status bit indicates which of the two Input Buffer RAMs (IBF) is accessible by the host (through CIF) as Input Buffer. The other non accessible buffer RAM is currently used as shadow buffer RAM by the ERAY Message Handler and therefore not accessible by the host.After reset, it is set by hardware. 0_B Input Buffer RAM 2 (IBF2) is accessible as Input Buffer by the host (CIF) 1_B Input Buffer RAM 1 (IBF1) is accessible as Input Buffer by the host (CIF)

(table continues...)

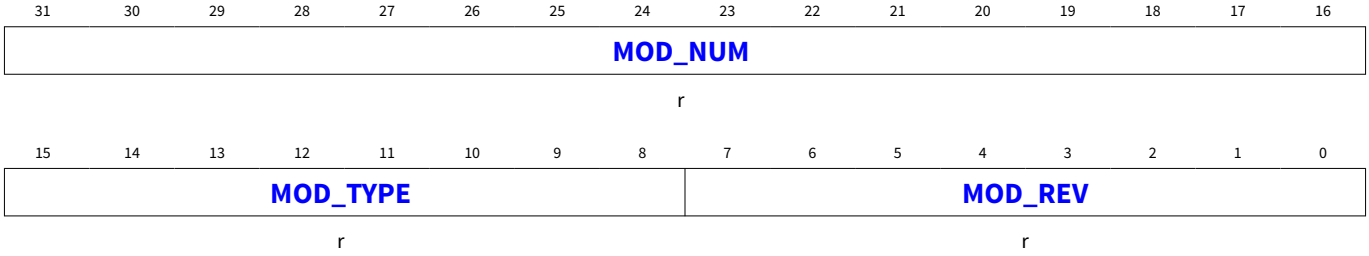
(continued)

Field	Bits	Type	Description
IBF1PAG	4	rw	**Input Buffer 1 Page Select Register** This control bit selects if the upper page or lower page of Input Buffer 1 (IBF1) currently active. ***Note**: Write is only possible, if Input Buffer RAM 1 is currently accessible by the host (through CIF) and therefore IBFS set.* 0_B Read: Lower Page (256 Bytes) of Input Buffer RAM 1 selected Write: Select Lower Page (256 Bytes) of Input Buffer RAM 1 1_B Read: Upper Page (256 Bytes) of Input Buffer RAM 1 selected Write: Select Upper Page (256 Bytes) of Input Buffer RAM 1
IBF2PAG	7	rw	**Input Buffer 2 Page Select Register** This control bit selects if the upper page or lower page of Input Buffer 2 (IBF2) currently active. ***Note**: Write is only possible, if Input Buffer RAM 2 is currently accessible by the host (through CIF) and therefore IBFS cleared.* 0_B Read: Lower Page (256 Bytes) of Input Buffer RAM 2 selected Write: Select Lower Page (256 Bytes) of Input Buffer RAM 2 1_B Read: Upper Page (256 Bytes) of Input Buffer RAM 2 selected Write: Select Upper Page (256 Byte) of Input Buffer RAM 2
RISA	11:10	rw	**Receive Input Select Channel A** 00_B Channel A receiver input RXDA0 selected 01_B Channel A receiver input RXDA1 selected 10_B Channel A receiver input RXDA2 selected 11_B Channel A receiver input RXDA3 selected
RISB	13:12	rw	**Receive Input Select Channel B** 00_B Channel B receiver input RXDB0 selected 01_B Channel B receiver input RXDB1 selected 10_B Channel B receiver input RXDB2 selected 11_B Channel B receiver input RXDB3 selected
STPWTS	15:14	rw	**Stop Watch Trigger Input Select** 00_B Stop Watch Trigger input STPWT0 selected 01_B Stop Watch Trigger input STPWT1 selected 10_B Stop Watch Trigger input STPWT2 selected 11_B Stop Watch Trigger input STPWT3 selected
0	6:5, 9:8, 31:16	r	**Reserved** Read as 0; should be written with 0.

23.4.5 Module Identification Register

ID Offset address: 0008_H

Module Identification Register PowerOn Reset value: $0044\ C009_H$



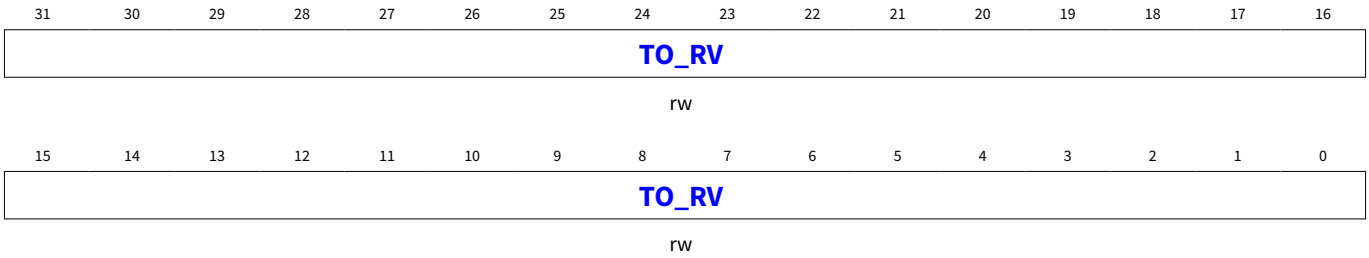
Field	Bits	Type	Description
MOD_REV	7:0	r	**Module Revision** Indicates the revision number of the implementation.
MOD_TYPE	15:8	r	**Module Type** This internal marker is fixed to $C0_H$.
MOD_NUM	31:16	r	**Module Number** Indicates the module identification number.

23.4.6 Customer Interface Timeout Counter Register

The Time-out Counter Register is realizing the time-out counter reload (startup) value for the automatic delay scheme (not the time-out down counter itself).

CUST3 Offset address: $000C_H$

Customer Interface Timeout Counter Register Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
TO_RV	31:0	rw	**CIF Timeout Reload Value - TO** The 32-bit down counter reload (start-up) value must be setup for the automatic delay scheme.

23.4.7 Test Register 1

The Test Register 1 holds the control bits to configure the test modes of the ERAY module. Write access to these bits is only possible if bit TEST1.WRTEN is set.

The Test Register 1 bits therefore can be used to test the interface to the physical layer (connectivity test) by driving / reading the respective pins.

When the ERAY IP is operated in one of its test modes that requires TEST1.WRTEN to be set (RAM Test Mode, I/O Test Mode, Asynchronous Transmit Mode, and Loop Back Mode) only the selected test mode functionality is available.

The test functions are not available in addition to the normal operational mode functions, they change the functions of parts of the ERAY module. Therefore normal operation as specified outside this chapter and as required by the FlexRay™ protocol specification and the FlexRay™ conformance test is not possible. Test mode functions may not be combined with each other or with FlexRay™ protocol functions.

The test mode features are intended for hardware testing or for FlexRay™ bus analyzer tools. They are not intended to be used in FlexRay™ applications

***Note**: LDMST or SWAPMSK.W instructions should be used only with bit mask enabled for all ´rwh´ bits in this register. In CERA and CERB, coding errors are also signalled when the Communication Controller is in "MONITOR_MODE". The error codes regarding CAS / MTS symbols concern only the monitored bit pattern, irrelevant whether those bit patterns are seen in the symbol window or elsewhere.*

TEST1 Offset address: 0010_H

Test Register 1 Kernel Reset value: $0000\ 0300_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CERB				CERA				0		TXENB	TXENA	TXB	TXA	RXB	RXA
rh				rh				r		rwh	rwh	rwh	rwh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0						AOB	AOA	0		TMC		0		ELBE	WRTEN
r						rh	rh	r		rw		r		rw	rw

Field	Bits	Type	Description
WRTEN	0	rw	**Write Test Register Enable** Enables write access to the test registers. To set the bit from 0 to 1 the test mode key has to be written as defined on “Lock Register”. The unlock sequence is not required when TEST1.WRTEN is kept at 1 while other bits of the register are changed. The bit can be reset to 0 at any time. 0_B Write access to test registers disabled. 1_B Write access to test registers enabled.

(table continues...)

(continued)

Field	Bits	Type	Description
ELBE	1	rw	**External Loop Back Enable** There are two possibilities to perform a loop back test. External loop back through physical layer or internal loop back for in-system self-test (default). In case of an internal loop back pins $\overline{\text{TXENA}}$ and $\overline{\text{TXENB}}$ are in their inactive state, pins TXDA and TXDB are set to HIGH, pins RXDA and RXDB are not evaluated. Bit ELBE is evaluated only when POC is in loop back mode and test multiplexer control is in non multiplexed mode TMC = 00. 0_B Internal loop back (default) 1_B External loop back
TMC	5:4	rw	**Test Multiplexer Control** 00_B Normal signal path (default). 01_B RAM Test Mode: Internal busses are multiplexed to make all RAM blocks of the ERAY module directly accessible by the Host. This mode is intended to enable testing of the embedded RAM blocks during production testing. 10_B I/O Test Mode: Output pins are driven to the values defined by bits TXA, TXB, $\overline{\text{TXENA}}$, $\overline{\text{TXENB}}$. The values applied to the input pins can be read from register bits RXA and RXB. 11_B Reserved; should not be used.
AOA	8	rh	**Activity on A** The channel idle condition is specified in the FlexRay™ protocol spec v2.1, chapter 3, BITSTRB process (zChannelIdle). 0_B No activity detected, channel A idle 1_B Activity detected, channel A not idle
AOB	9	rh	**Activity on B** The channel idle condition is specified in the FlexRay™ protocol spec v2.1, chapter 3, BITSTRB process (zChannelIdle). 0_B No activity detected, channel B idle 1_B Activity detected, channel B not idle
RXA	16	rh	**Read Channel A Receive Pin** This bit field shows the current logic state of RXDA. 0_B RXDA = 0 1_B RXDA = 1
RXB	17	rh	**Read Channel B Receive Pin** This bit field shows the current logic state of RXDB. 0_B RXDB = 0 1_B RXDB = 1
TXA	18	rwh	**Read or Write to Channel A Transmit Pin** A write to this bit field sets the TXDA to corresponding logic state. A read from this bit field shows the current logic state of TXDA. 0_B TXDA = 0 1_B TXDA = 1

(table continues...)

(continued)

Field	Bits	Type	Description
TXB	19	rwh	**Read or Write to Channel B Transmit Pin** A write to this bit field sets the TXDB to corresponding logic state. A read from this bit field shows the current logic state of TXDB. 0_B TXDB = 0 1_B TXDB = 1
TXENA	20	rwh	**Read or Write to Channel A Transmit Enable Pin** A write to this bit field sets the TXENA to corresponding logic state. A read from this bit field shows the current logic state of TXENA. 0_B $\overline{\text{TXENA}}$= 0 1_B $\overline{\text{TXENA}}$= 1
TXENB	21	rwh	**Read or Write to Channel B Transmit Enable Pin** A write to this bit field sets the TXENB to corresponding logic state. A read from this bit field shows the current logic state of TXENB. 0_B $\overline{\text{TXENB}}$ = 0 1_B $\overline{\text{TXENB}}$= 1
CERA	27:24	rh	**Coding Error Report Channel A** Set when a coding error is detected on channel A. Reset to zero when register TEST1 is read or written. Once the CERA is set it will remain unchanged until the Host accesses the TEST1 register. Other combinations are reserved. 0_H No coding error detected 1_H Header CRC error detected 2_H Frame CRC error detected 3_H Frame Start Sequence FSS too long 4_H First bit of Byte Start Sequence BSS seen LOW 5_H Second bit of Byte Start Sequence BSS seen HIGH 6_H First bit of Frame End Sequence FES seen HIGH 7_H Second bit of Frame End Sequence FES seen LOW 8_H CAS / MTS symbol seen too short 9_H CAS / MTS symbol seen too long

(table continues...)

(continued)

Field	Bits	Type	Description
CERB	31:28	rh	**Coding Error Report Channel B** Set when a coding error is detected on channel B. Reset to zero when register TEST1 is read or written. Once the CERB is set it will remain unchanged until the Host accesses the TEST1 register. Other combinations are reserved. 0_H No coding error detected 1_H Header CRC error detected 2_H Frame CRC error detected 3_H Frame Start Sequence FSS too long 4_H First bit of Byte Start Sequence BSS seen LOW 5_H Second bit of Byte Start Sequence BSS seen HIGH 6_H First bit of Frame End Sequence FES seen HIGH 7_H Second bit of Frame End Sequence FES seen LOW 8_H CAS / MTS symbol seen too short 9_H CAS / MTS symbol seen too long
0	3:2, 7:6, 15:10, 23:22	r	**Reserved** Read as 0; should be written with 0.

Asynchronous Transmit Mode (ATM)

The asynchronous transmit mode is entered by writing 1110_B to the CHI Command Vector SUCC1.CMD in the SUC Configuration Register 1 (CHI command: ATM) while the Communication Controller is in “CONFIG” state and bit TEST1.WRTEN in the Test Register 1 is set. This write operation has to be directly preceded by two consecutive write accesses to the Configuration Lock Key (unlock sequence). When called in any other state or when bit TEST1.WRTEN is not set, SUCC1.CMD will be reset to 0000_B = “COMMAND_NOT_ACCEPTED”. CCSV.POCS in the Communication Controller Status Vector will return 1110_B while the ERAY module is in ATM mode. Asynchronous Transmit mode can be left by writing 0001_B (CHI command: “CONFIG”) to the CHI Command Vector SUCC1.CMD in the SUC Configuration Register 1.

In ATM mode transmission of a FlexRay™ Frame is triggered by writing the number of the respective Message Buffer to the Input Buffer Command Request register (IBCR.IBRH) while bit IBCM.STXRS in the Input Buffer Command Mask register is set to 1. In this mode wake-up, startup, and clock synchronization are bypassed. The CHI command SEND_MTS results in the immediate transmission of an MTS symbol.

The cycle counter value of Frames send in ATM mode can be programmed in MTCCV.CCV (writable in ATM and loop back mode only).

Loop Back Mode

The loop back mode is entered by writing 1111_B to the CHI Command Vector SUCC1.CMD in the SUC Configuration Register 1 (CHI command: LOOP_BACK) while the Communication Controller is in “CONFIG” state and bit TEST1.WRTEN in the Test Register 1 is set. This write operation has to be directly preceded by two consecutive write accesses to the Configuration Lock Key (unlock sequence). When called in any other state or when bit TEST1.WRTEN is not set, SUCC1.CMD will be reset to 0000_B = “COMMAND_NOT_ACCEPTED”. CCSV.POCS in the Communication Controller Status Vector will show $0000\ 1101_H$ while the ERAY module is in loop back mode.

Loop Back mode can be left by writing 0001_B (CHI command: “CONFIG”) to the CHI Command Vector SUCC1.CMD in the SUC Configuration Register 1.

The loop back test mode is intended to check the module’s internal data paths. Normal, time triggered operation is not possible in loop back mode.

There are two possibilities to perform a loop back test. External loop back through physical layer (TEST1.ELBE = 1) or internal loop back for in-system self-test (TEST1.ELBE = 0). In case of an internal loop back pins $\overline{\text{TXENA}}$,

$\overline{\text{TXENB}}$ are in their inactive state, pins TXDA and TXDB are set to HIGH, pins RXDAn and RXDBn are not evaluated.

When the Communication Controller is in loop back mode, a loop back test is started by the Host writing a message to the Input Buffer and requesting the transmission by writing to the Input Buffer Command Request register IBCR. The Message Handler will transfer the message into the Message RAM and then into the Transient Buffer of the selected channel. The Channel Protocol Controller (PRT) will read (in 32-bit words) the message from the transmit part of the Transient Buffer and load it into its Rx / Tx shift register. The serial transmission is looped back into the shift register; its content is written into the receive part of the channels's Transient Buffer before the next word is loaded.

The PRT and the Message Handler will then treat this transmitted message like a received message, perform an acceptance filtering on Frame ID and receive channel, and store the message into the Message RAM if it passed acceptance filtering. The loop back test ends with the Host requesting this received message from the Message RAM and then checking the contents of the Output Buffer.

Each FlexRay™ channel is tested separately. The ERAY cannot receive messages from the FlexRay™ bus while it is in the loop back mode.

The cycle counter value of Frames used in loop back mode can be programmed in MTCCV.CCV (writable in ATM and loop back mode only).

Note that in case of an odd payload the last two bytes of the looped-back payload will be shifted by 16 bits to the right inside the last 32-bit data word.

23.4.8 Test Register 2

The Test Register 2 holds all bits required for the RAM test of the seven embedded RAM blocks of the ERAY module. Write access to this register is only possible when TEST1.WRTEN in the Test Register 1 is set to 1.

TEST2 Offset address: 0014_H

Test Register 2 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0								SSEL			0	RS		
r	r								rw			r	rw		

Field	Bits	Type	Description
RS	2:0	rw	**RAM Select** In RAM Test mode the RAM blocks selected by RS are mapped to module address $0000\ 0400_H$ to $0000\ 07FF_H$ (1024 byte addresses). 000_B Input Buffer RAM 1 (IBF1) 001_B Input Buffer RAM 2 (IBF2) 010_B Output Buffer RAM 1 (OBF1) 011_B Output Buffer RAM 2 (OBF2) 100_B Transient Buffer RAM A (TBF1) 101_B Transient Buffer RAM B (TBF2) 110_B Message RAM (MBF) 111_B Reserved; should not be used.

(table continues...)

(continued)

Field	Bits	Type	Description
SSEL	6:4	rw	**Segment Select** To enable access to the complete Message RAM (8192 byte addresses) the Message RAM is segmented. 000_B Access to RAM byte 0000_H to $03FF_H$ enabled 001_B Access to RAM byte 0400_H to $07FF_H$ enabled 010_B Access to RAM byte 0800_H to $0BFF_H$ enabled 011_B Access to RAM byte $0C00_H$ to $0FFF_H$ enabled 100_B Access to RAM byte 1000_H to $13FF_H$ enabled 101_B Access to RAM byte 1400_H to $17FF_H$ enabled 110_B Access to RAM byte 1800_H to $1BFF_H$ enabled 111_B Access to RAM byte $1C00_H$ to $1FFF_H$ enabled
0	3, 14:7, 15, 31:16	r	**Reserved** Read as 0; should be written with 0.

RAM Test Mode

In RAM test mode (TEST1.TMC = 1), one of the seven RAM blocks can be selected for direct RD/WR access by programming TEST2.RS.

For external access the selected RAM block is mapped to address space 400_H to $7FF_H$ (1024 byte addresses or 256 word addresses).

Because the length of the Message RAM exceeds the available address space, the Message RAM is segmented into segments of 1024 byte. The segments can be selected by programming TEST2.SSEL in the Test Register 2.

In RAM test mode the memory must be accessed with 32Bit accesses only.

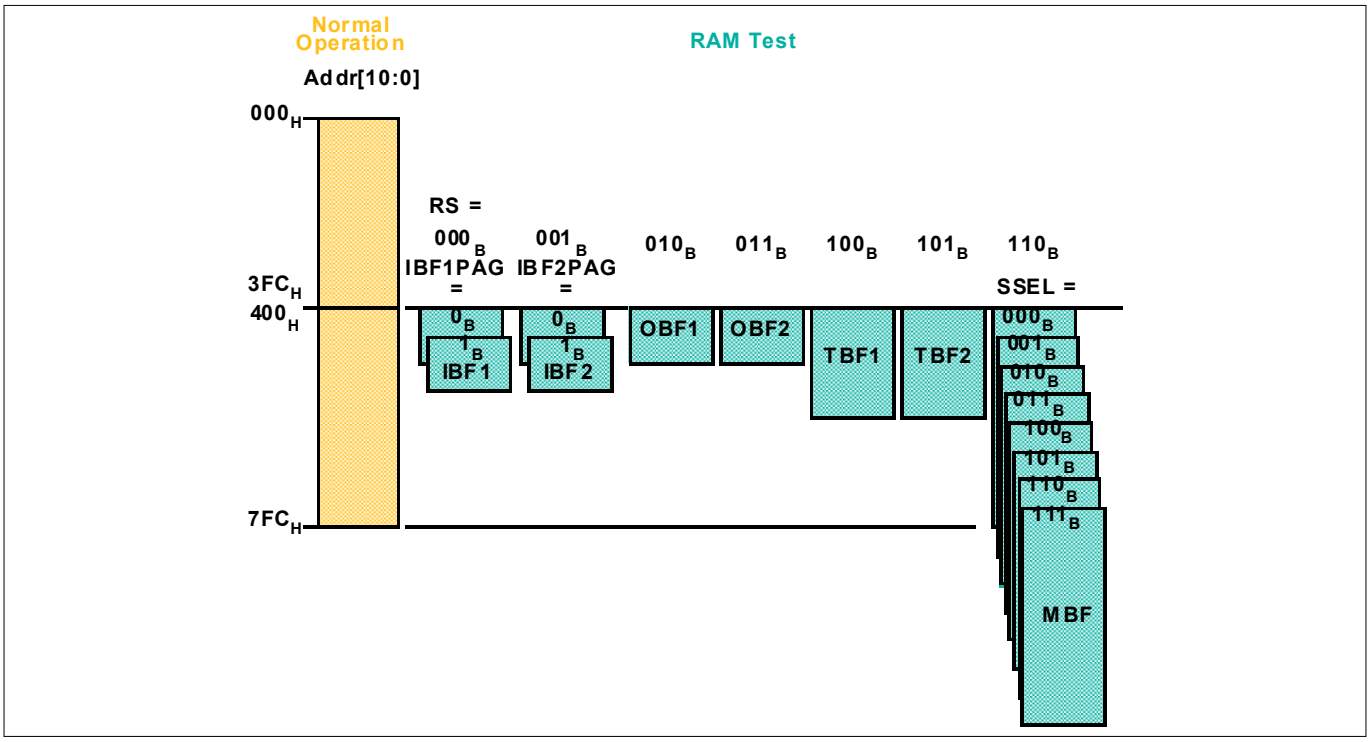


Figure 454 RAM test mode Access to ERAY RAM Blocks

23.4.9 Lock Register

The Lock Register is write-only. Reading the register will return 0000 0000$_H$.

***Note**: In case the Host uses 8/16-bit accesses to write the listed bit fields, the programmer has to ensure that no "dummy accesses" e.g. to the remaining register bytes / words are inserted by the compiler.*

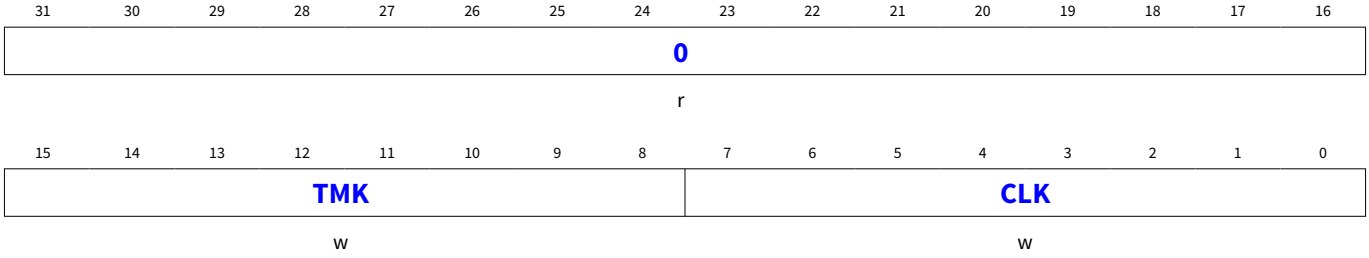
To exit "CONFIG" state by writing to SUCC1.CMD in the SUC Configuration Register 1, the write operation has to be directly preceded by two consecutive write accesses to the Configuration Lock Key. If this write sequence is service requested by read accesses or write accesses to other locations, the Communication Controller remains in "CONFIG" state and the sequence has to be repeated.

First write: LCK.CLK = CE$_H$ = 1100 1110$_B$

Second write: LCK.CLK = 31$_H$ = 0011 0001$_B$

LCK Offset address: 001C$_H$

Lock Register Kernel Reset value: 0000 0000$_H$



Field	Bits	Type	Description
CLK	7:0	w	**Configuration Lock Key** To leave "CONFIG" state by writing to SUCC1.CMD (commands READY, MONITOR_MODE, ATM, LOOP_BACK) in the SUC Configuration Register 1, the write operation has to be directly preceded by two consecutive write accesses to the Configuration Lock Key (unlock sequence). If the write sequence below is interrupted by other write accesses between the second write to the Configuration Lock Key and the write access to the SUCC1 register, the Communication Controller remains in "CONFIG" state and the sequence has to be repeated. First write: LCK.CLK = CE$_H$ = 1100 1110$_B$ Second write: LCK.CLK = 31$_H$ = 0011 0001$_B$ Third write: SUCC1.CMD Returns 0 if read
TMK	15:8	w	**Test Mode Key** To set bit TEST1.WRTEN the write operation has to be directly preceded by two consecutive write accesses to the Test Mode Key. If the write sequence is interrupted by other write accesses between the second write to the Test Mode Key and the write access to the Test1 register, bit TEST1.WRTEN is not set to 1 and the sequence has to be repeated. First write: LCK.TMK = 75$_H$ = 0111 0101$_B$ Second write: LCK.TMK = 8A$_H$ = 1000 1010$_B$ Third write: TEST1.WRTEN = 1 Returns 0 if read
0	31:16	r	**Reserved** Read as 0; should be written with 0.

23.4.10 Error Service Request Select Register

The flags are set when the Communication Controller detects one of the listed error conditions. They remain set until the Host clears them. A flag is cleared by writing a 1 to the corresponding bit position. Writing a 0 has no effect on the flag. An application reset will also clear the register.

***Note**: LDMST or SWAPMSK.W instructions should be used only with bit mask enabled for all ´rwh´ bits in this register.*

EIR Offset address: 0020_H

Error Service Request Select Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0					TABB	LTVB	EDB	0					TABA	LTVA	EDA
r					rwh	rwh	rwh	r					rwh	rwh	rwh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				MHF	IOBA	IIBA	EFA	RFO	EERR	CCL	CCF	SFO	SFBM	CNA	PEMC
r				rwh	rwh	rwh	rwh	rh	rh	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
PEMC	0	rwh	**POC Error Mode Changed** This flag is set whenever the error mode signalled by CCEV.ERRM in the Communication Controller Error Vector register has changed. This flag is cleared by writing a 1. 0_B Error mode has not changed 1_B Error mode has changed
CNA	1	rwh	**Command Not Accepted** The flag signals that the write access to the CHI command vector SUCC1.CMD in the SUC Configuration Register 1 was not successful because the requested command was not valid in the actual POC state, or because the CHI command was locked (CCL = 1). This flag is cleared by writing a 1. 0_B CHI command accepted 1_B CHI command not accepted
SFBM	2	rwh	**SYNC Frames Below Minimum** This flag signals that the number of SYNC Frames received during the last communication cycle was below the limit required by the FlexRay™ protocol. May be set during startup and therefore should be cleared by the Host after the Communication Controller entered “NORMAL_ACTIVE” state. This flag is cleared by writing a 1. 0_B Sync node: 1 or more SYNC Frames received Non-sync node: 2 or more SYNC Frames received 1_B Less than the required minimum of SYNC Frames received

(table continues...)

(continued)

Field	Bits	Type	Description
SFO	3	rwh	**SYNC Frame Overflow** Set when either the number of SYNC Frames received during the last communication cycle or the total number of SYNC Frames received during the last double cycle exceeds the maximum number of SYNC Frames as defined by GTUC02.SNM in the GTU Configuration Register 2. This flag is cleared by writing a 1. 0_B Number of received SYNC Frames ≤ GTUC02.SNM 1_B More SYNC Frames received than configured by GTUC02.SNM
CCF	4	rwh	**Clock Correction Failure** This flag is set at the end of the cycle whenever one of the following errors occurred: • Missing offset and / or rate correction • Clock Correction limit reached The clock correction status is monitored in registers CCEV and SFS. A failure may occur during startup, therefore bit CCF should be cleared by the Host after the Communication Controller entered “NORMAL_ACTIVE” state. This flag is cleared by writing a 1. 0_B Clock correction successful so far 1_B Clock correction failed
CCL	5	rwh	**CHI Command Locked** The flag signals that the write access to the CHI command vector SUCC1.CMD was not successful because the execution of the previous CHI command has not yet completed. In this case bit EIR.CNA is also set to 1. This flag is cleared by writing a 1. 0_B CHI command accepted 1_B CHI command not accepted
EERR	6	rh	**ECC Error** The flag signals an ECC error to the Host. It is set whenever one of the flags MHDS.EIBF, MHDS.EOBF, MHDS.EMR, MHDS.ETBF1, MHDS.ETBF2 changes from 0 to 1. See also “Message Handler Status”. This bit must be cleared at initialization of the module. 0_B No error detected 1_B Error detected
RFO	7	rh	**Receive FIFO Overrun** The flag is set by the Communication Controller when a receive FIFO overrun is detected. When a receive FIFO overrun occurs, the oldest message is overwritten with the actual received message. The actual state of the FIFO is monitored in register FSR. 0_B No receive FIFO overrun detected 1_B A receive FIFO overrun has been detected

(table continues...)

(continued)

Field	Bits	Type	Description
EFA	8	rwh	**Empty FIFO Access** This flag is set by the Communication Controller when the Host requests the transfer of a message from the receive FIFO through Output Buffer while the receive FIFO is empty. 0_B No Host access to empty FIFO occurred 1_B Host access to empty FIFO occurred
IIBA	9	rwh	**Illegal Input Buffer Access** This flag is set by the Communication Controller when the Host wants to modify a Message Buffer through Input Buffer while the Communication Controller is not in “CONFIG” or “DEFAULT_CONFIG” state and one of the following conditions applies: **1.** The Host writes to the Input Buffer Command Request register to modify the: • Header Section of Message Buffer 0, 1 if configured for transmission in key slot • Header Section of static Message Buffers with buffer number < MRC.FDB while MRC.SEC = 01_B • Header Section of any static or dynamic Message Buffer while MRC.SEC = $1x_B$ • Header and / or Data Section of any message buffer belonging to the receive FIFO **2.** The Host writes to any register of the Input Buffer while IBCR.IBSYS is set. 0_B No illegal Host access to Input Buffer occurred 1_B Illegal Host access to Input Buffer occurred
IOBA	10	rwh	**Illegal Output Buffer Access** This flag is set by the Communication Controller when the Host requests the transfer of a Message Buffer from the Message RAM to the Output Buffer while OBCR.OBSYS is set to 1. 0_B No illegal Host access to Output Buffer occurred 1_B Illegal Host access to Output Buffer occurred
MHF	11	rwh	**Message Handler Constraints Flag** The flag signals a Message Handler constraints violation condition. It is set whenever one of the flags MHDF.SNUA, MHDF.SNUB, MHDF.FNFA, MHDF.FNFB, MHDF.TBFA, MHDF.TBFB, MHDF.TNSA, MHDF.TNSB, MHDF.WAHP changes from 0 to 1. 0_B No Message Handler failure detected 1_B Message Handler failure detected
EDA	16	rwh	**Error Detected on Channel A** This bit is set whenever one of the flags ACS.SEDA, ACS.CEDA, ACS.CIA, ACS.SBVA changes from 0 to 1. This flag is cleared by writing a 1. 0_B No error detected on channel A 1_B Error detected on channel A

(table continues...)

(continued)

Field	Bits	Type	Description
LTVA	17	rwh	**Latest Transmit Violation Channel A** The flag signals a latest transmit violation on channel A to the Host. This flag is cleared by writing a 1. 0_B No latest transmit violation detected on channel A 1_B Latest transmit violation detected on channel A
TABA	18	rwh	**Transmission Across Boundary Channel A** The flag signals to the Host that a transmission across a slot boundary occurred for channel A. This flag is cleared by writing a 1. 0_B No transmission across slot boundary detected on channel A 1_B Transmission across slot boundary detected on channel A
EDB	24	rwh	**Error Detected on Channel B** This bit is set whenever one of the flags ACS.SEDB, ACS.CEDB, ACS.CIB, ACS.SBVB changes from 0 to 1. This flag is cleared by writing a 1. 0_B No error detected on channel B 1_B Error detected on channel B
LTVB	25	rwh	**Latest Transmit Violation Channel B** The flag signals a latest transmit violation on channel B to the Host. This flag is cleared by writing a 1. 0_B No latest transmit violation detected on channel B 1_B Latest transmit violation detected on channel B
TABB	26	rwh	**Transmission Across Boundary Channel B** The flag signals to the Host that a transmission across a slot boundary occurred for channel B. This flag is cleared by writing a 1. 0_B No transmission across slot boundary detected on channel B 1_B Transmission across slot boundary detected on channel B
0	15:12, 23:19, 31:27	r	**Reserved** Read as 0; should be written with 0.

23.4.11 Status Service Request Register

The flags are set whenever the Communication Controller detects one of the listed events. The flags remain set until the Host clears them. A flag is cleared by writing a 1 to the corresponding bit position. Writing a 0 has no effect on the flag. An application reset will also clear the register.

***Note**: LDMST or SWAPMSK.W instructions should be used only with bit mask enabled for all ´rwh´ bits in this register.*

SIR Offset address: 0024_H

Status Service Request Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0						MTSB	WUPB	0						MTSA	WUPA
r						rwh	rwh	r						rwh	rwh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SDS	MBSI	SUCS	SWE	TOBC	TIBC	TI1	TI0	NMVC	RFCL	RFNE	RXI	TXI	CYCS	CAS	WST
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rh	rh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
WST	0	rwh	**Wakeup Status** This flag is set when the wakeup status vector CCSV.WSV in the Communication Controller Status Vector register changes to a value other than UNDEFINED. This flag is cleared by writing a 1. 0_B Wake-up status unmodified 1_B Wake-up status modified (and not UNDEFINED)
CAS	1	rwh	**Collision Avoidance Symbol** This flag is set by the Communication Controller during STARTUP state when a CAS or potential CAS was received. This flag is cleared by writing a 1. 0_B No bit pattern matching the CAS symbol received 1_B Bit pattern matching the CAS symbol received
CYCS	2	rwh	**Cycle Start Service Request** This flag is set by the Communication Controller when a communication cycle starts This flag is cleared by writing a 1. 0_B No communication cycle started 1_B Communication cycle started
TXI	3	rwh	**Transmit Service Request** This flag is set by the Communication Controller at the end of Frame transmission if bit WRHS1.MBI in the respective Message Buffer is set. This flag is cleared by writing a 1. 0_B No Frame transmitted from a transmit buffer with WRHS1.MBI = 1 1_B At least one Frame was transmitted from a transmit buffer with WRHS1.MBI = 1
RXI	4	rwh	**Receive Service Request** This flag is set by the Communication Controller whenever the set condition of a Message Buffer ND flag is fulfilled and if bit WRHS1.MBI of that Message Buffer is set to 1. This flag is cleared by writing a 1. 0_B No ND flag of a receive buffer with WRHS1.MBI = 1 has been set to 1 1_B At least one ND flag of a receive buffer with WRHS1.MBI = 1 has been set to 1

(table continues...)

(continued)

Field	Bits	Type	Description
RFNE	5	rh	**Receive FIFO Not Empty** This flag is set by the Communication Controller when a received valid Frame was stored into the empty receive FIFO.m The actual state of the receive FIFO is monitored in register FSR 0_B Receive FIFO is empty 1_B Receive FIFO is not empty
RFCL	6	rh	**Receive FIFO Critical Level** This flag is set when a valid receive FIFO fill level FSR.RFFL is equal or greater than the critical level as configured by FCL.CL. 0_B Receive FIFO below critical level 1_B Receive FIFO critical level reached
NMVC	7	rwh	**Network Management Vector Changed** This service request flag signals a change in the Network Management Vector visible to the Host. This flag is cleared by writing a 1. 0_B No change in the Network Management vector 1_B Network Management vector changed
TI0	8	rwh	**Timer Service Request 0** This flag is set whenever timer 0 matches the conditions configured in the Timer Service Request 0 Configuration Register T0C. A Timer Service Request 0 is also signalled by TINT0SRC. This flag is cleared by writing a 1. 0_B No Timer Service Request 0 1_B Timer Service Request 0 occurred
TI1	9	rwh	**Timer Service Request 1** This flag is set whenever the conditions programmed in the Timer Service Request 1 Configuration Register T1C are met. A Timer Service Request 1 is also signalled by TINT1SRC. This flag is cleared by writing a 1. 0_B No Timer Service Request 1 1_B Timer Service Request 1 occurred
TIBC	10	rwh	**Transfer Input Buffer Completed** This flag is set whenever a transfer from Input Buffer to the Message RAM has completed and bit IBCR.IBSYS in the Input Buffer Command Request register has been reset by the Message Handler. This flag is cleared by writing a 1. 0_B No transfer completed 1_B Transfer between Input Buffer and Message RAM completed

(table continues...)

(continued)

Field	Bits	Type	Description
TOBC	11	rwh	**Transfer Output Buffer Completed** This flag is set whenever a transfer from Message RAM to the Output Buffer has completed and bit OBCR.OBSYS in the Output Buffer Command Request register has been reset by the Message Handler. This flag is cleared by writing a 1. 0_B No transfer completed 1_B Transfer between Message RAM and the Output Buffer completed
SWE	12	rwh	**Stop Watch Event** This flag is set after a stop watch activation when the current cycle counter and Macrotick value are stored in the Stop Watch Register 1 (STPW1). This flag is cleared by writing a 1. 0_B No Stop Watch Event 1_B Stop Watch Event occurred
SUCS	13	rwh	**Startup Completed Successfully** This flag is set whenever a startup completed successfully and the Communication Controller entered “NORMAL_ACTIVE” state. This flag is cleared by writing a 1. 0_B No startup completed successfully 1_B Startup completed successfully
MBSI	14	rwh	**Message Buffer Status Service Request** This flag is set by the Communication Controller when the Message Buffer status MBS has changed and if bit RDHS1.MBI of that Message Buffer is set. This flag is cleared by writing a 1. 0_B No Message Buffer status change of Message Buffer with RDHS1.MBI= 1 has changed 1_B Message Buffer status of at least one Message Buffer with RDHS1.MBI= 1 has changed
SDS	15	rwh	**Start of Dynamic Segment** This flag is set by the Communication Controller when the dynamic segment starts. 0_B Dynamic segment not yet started 1_B Dynamic segment started
WUPA	16	rwh	**Wakeup Pattern Channel A** This flag is set by the Communication Controller when a wakeup pattern was received on channel A. Only set when the Communication Controller is in “WAKEUP”, “READY”, or “STARTUP” state, or when in Monitor mode. This flag is cleared by writing a 1. 0_B No wake-up pattern received on channel A 1_B Wake-up pattern received on channel A

(table continues...)

(continued)

Field	Bits	Type	Description
MTSA	17	rwh	**MTS Received on Channel A (vSS!ValidMTSA)** Media Access Test symbol received on channel A during the proceeding symbol window. Updated by the Communication Controller for each channel at the end of the symbol window. This flag is cleared by writing a 1. 0_B No MTS symbol received on channel A 1_B MTS symbol received on channel A
WUPB	24	rwh	**Wakeup Pattern Channel B** This flag is set by the Communication Controller when a wakeup pattern was received on channel B. Only set when the Communication Controller is in “WAKEUP”, “READY”, or “STARTUP” state, or when in Monitor mode. This flag is cleared by writing a 1. 0_B No wake-up pattern received on channel B 1_B Wake-up pattern received on channel B
MTSB	25	rwh	**MTS Received on Channel B** Media Access Test symbol received on channel B during the proceeding symbol window. Updated by the Communication Controller for each channel at the end of the symbol window. This flag is cleared by writing a 1. 0_B No MTS symbol received on channel B 1_B MTS symbol received on channel B
0	23:18, 31:26	r	**Reserved** Read as 0; should be written with 0.

23.4.12 Error Service Request Line Select

The Error Service Request Line Select register assigns a service request generated by a specific error service request flag from register EIR to one of the two module service request lines INT0SRC or INT1SRC:

0 = Interrupt assigned to interrupt line (INT0SRC)

1 = Interrupt assigned to interrupt line (INT1SRC)

EILS Offset address: 0028_H

Error Service Request Line Select Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0					TABBL	LTVBL	EDBL	0					TABAL	LTVAL	EDAL
r					rw	rw	rw	r					rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				MHFL	IOBAL	IIBAL	EFAL	RFOL	EERRL	CCLL	CCFL	SFOL	SFBML	CNAL	PEMCL
r				rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
PEMCL	0	rw	**POC Error Mode Changed Service Request Line** 0_B Service Request assigned to service request line INT0SRC 1_B Service Request assigned to service request line INT1SRC
CNAL	1	rw	**Command Not Accepted Service Request Line** 0_B Service Request assigned to service request line INT0SRC 1_B Service Request assigned to service request line INT1SRC
SFBML	2	rw	**SYNC Frames Below Minimum Service Request Line** 0_B Service Request assigned to service request line INT0SRC 1_B Service Request assigned to service request line INT1SRC
SFOL	3	rw	**SYNC Frame Overflow Service Request Line** 0_B Service Request assigned to service request line INT0SRC 1_B Service Request assigned to service request line INT1SRC
CCFL	4	rw	**Clock Correction Failure Service Request Line** 0_B Service Request assigned to service request line INT0SRC 1_B Service Request assigned to service request line INT1SRC
CCLL	5	rw	**CHI Command Locked Service Request Line** 0_B Service Request assigned to service request line INT0SRC 1_B Service Request assigned to service request line INT1SRC
EERRL	6	rw	**ECC Error Service Request Line** 0_B Service Request assigned to service request line INT0SRC 1_B Service Request assigned to service request line INT1SRC
RFOL	7	rw	**Receive FIFO Overrun Service Request Line** 0_B Service Request assigned to service request line INT0SRC 1_B Service Request assigned to service request line INT1SRC
EFAL	8	rw	**Empty FIFO Access Service Request Line** 0_B Service Request assigned to service request line INT0SRC 1_B Service Request assigned to service request line INT1SRC
IIBAL	9	rw	**Illegal Input Buffer Access Service Request Line** 0_B Service Request assigned to service request line INT0SRC 1_B Service Request assigned to service request line INT1SRC
IOBAL	10	rw	**Illegal Output Buffer Access Service Request Line** 0_B Service Request assigned to service request line INT0SRC 1_B Service Request assigned to service request line INT1SRC
MHFL	11	rw	**Message Handler Constrains Flag Service Request Line** 0_B Service Request assigned to service request line INT0SRC 1_B Service Request assigned to service request line INT1SRC
EDAL	16	rw	**Error Detected on Channel A Service Request Line** 0_B Service Request assigned to service request line INT0SRC 1_B Service Request assigned to service request line INT1SRC

(table continues...)

(continued)

Field	Bits	Type	Description
LTVAL	17	rw	**Latest Transmit Violation Channel A Service Request Line** 0_B Service Request assigned to service request line INT0SRC 1_B Service Request assigned to service request line INT1SRC
TABAL	18	rw	**Transmission Across Boundary Channel A Service Request Line** 0_B Service Request assigned to service request line INT0SRC 1_B Service Request assigned to service request line INT1SRC
EDBL	24	rw	**Error Detected on Channel B Service Request Line** 0_B Service Request assigned to service request line INT0SRC 1_B Service Request assigned to service request line INT1SRC
LTVBL	25	rw	**Latest Transmit Violation Channel B Service Request Line** 0_B Service Request assigned to service request line INT0SRC 1_B Service Request assigned to service request line INT1SRC
TABBL	26	rw	**Transmission Across Boundary Channel B Service Request Line** 0_B Service Request assigned to service request line INT0SRC 1_B Service Request assigned to service request line INT1SRC
0	15:12, 23:19, 31:27	r	**Reserved** Read as 0; should be written with 0.

23.4.13 Status Service Request Line Select

The Status Service Request Line Select register assign an service request generated by a specific status service request flag from register SIR to one of the two module service request lines INT0SRC or INT1SRC:

0 = Interrupt assigned to interrupt line INT0SRC

1 = Interrupt assigned to interrupt line INT1SRC

SILS Offset address: $002C_H$

Status Service Request Line Select Kernel Reset value: $0303\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0						MTSBL	WUPBL	0						MTSAL	WUPAL
r						rw	rw	r						rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SDSL	MBSIL	SUCSL	SWEL	TOBCL	TIBCL	TI1L	TI0L	NMVCL	RFCLL	RFNEL	RXIL	TXIL	CYCSL	CASL	WSTL
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
WSTL	0	rw	**Wakeup Status Service Request Line** 0_B Service Request assigned to service request line INT0SRC 1_B Service Request assigned to service request line INT1SRC

(table continues...)

(continued)

Field	Bits	Type	Description
CASL	1	rw	**Collision Avoidance Symbol Service Request Line** 0_B Service Request assigned to service request line INT0SRC 1_B Service Request assigned to service request line INT1SRC
CYCSL	2	rw	**Cycle Start Service Request Line** 0_B Service Request assigned to service request line INT0SRC 1_B Service Request assigned to service request line INT1SRC
TXIL	3	rw	**Transmit Service Request Line** 0_B Service Request assigned to service request line INT0SRC 1_B Service Request assigned to service request line INT1SRC
RXIL	4	rw	**Receive Service Request Line** 0_B Service Request assigned to service request line INT0SRC 1_B Service Request assigned to service request line INT1SRC
RFNEL	5	rw	**Receive FIFO Not Empty Service Request Line** 0_B Service Request assigned to service request line INT0SRC 1_B Service Request assigned to service request line INT1SRC
RFCLL	6	rw	**Receive FIFO Critical Level Service Request Line** 0_B Service Request assigned to service request line INT0SRC 1_B Service Request assigned to service request line INT1SRC
NMVCL	7	rw	**Network Management Vector Changed Service Request Line** 0_B Service Request assigned to service request line INT0SRC 1_B Service Request assigned to service request line INT1SRC
TI0L	8	rw	**Timer Service Request 0 Line** 0_B Service Request assigned to service request line INT0SRC 1_B Service Request assigned to service request line INT1SRC
TI1L	9	rw	**Timer Service Request 1 Line** 0_B Service Request assigned to service request line INT0SRC 1_B Service Request assigned to service request line INT1SRC
TIBCL	10	rw	**Transfer Input Buffer Completed Service Request Line** 0_B Service Request assigned to service request line INT0SRC 1_B Service Request assigned to service request line INT1SRC
TOBCL	11	rw	**Transfer Output Buffer Completed Service Request Line** 0_B Service Request assigned to service request line INT0SRC 1_B Service Request assigned to service request line INT1SRC
SWEL	12	rw	**Stop Watch Event Service Request Line** 0_B Service Request assigned to service request line INT0SRC 1_B Service Request assigned to service request line INT1SRC
SUCSL	13	rw	**Startup Completed Successfully Service Request Line** 0_B Service Request assigned to service request line INT0SRC 1_B Service Request assigned to service request line INT1SRC

(table continues...)

(continued)

Field	Bits	Type	Description
MBSIL	14	rw	**Message Buffer Status Service Request Line** 0_B Service Request assigned to service request line INT0SRC 1_B Service Request assigned to service request line INT1SRC
SDSL	15	rw	**Start of Dynamic Segment Service Request Line** 0_B Service Request assigned to service request line INT0SRC 1_B Service Request assigned to service request line INT1SRC
WUPAL	16	rw	**Wakeup Pattern Channel A Service Request Line** 0_B Service Request assigned to service request line INT0SRC 1_B Service Request assigned to service request line INT1SRC
MTSAL	17	rw	**Media Access Test Symbol Channel A Service Request Line** 0_B Service Request assigned to service request line INT0SRC 1_B Service Request assigned to service request line INT1SRC
WUPBL	24	rw	**Wakeup Pattern Channel B Service Request Line** 0_B Service Request assigned to service request line INT0SRC 1_B Service Request assigned to service request line INT1SRC
MTSBL	25	rw	**Media Access Test Symbol Channel B Service Request Line** 0_B Service Request assigned to service request line INT0SRC 1_B Service Request assigned to service request line INT1SRC
0	23:18, 31:26	r	**Reserved** Read as 0; should be written with 0.

23.4.14 Error Service Request Enable Set

The settings in the Error Service Request Enable register determine which status changes in the Error Service Request Register will result in an service request. The enable bits are set by writing to EIES and reset by writing to EIER. Writing a 1 sets the specific enable bit, a 0 has no effect.

***Note**: LDMST or SWAPMSK.W instructions should be used only with bit mask enabled for all ´rwh´ bits in this register.*

EIES Offset address: 0030_H

Error Service Request Enable Set Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0					TABBE	LTVBE	EDBE	0					TABAE	LTVAE	EDAE
r					rwh	rwh	rwh	r					rwh	rwh	rwh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				MHFE	IOBAE	IIBAE	EFAE	RFOE	EERRE	CCLE	CCFE	SFOE	SFBME	CNAE	PEMCE
r				rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
PEMCE	0	rwh	**POC Error Mode Changed Service Request Enable** 0_B Read: Protocol Error Mode Changed Service Request disabled Write: Unchanged 1_B Read: Protocol Error Mode Changed Service Request enabled Write: Enable Protocol Error Mode Changed Service Request
CNAE	1	rwh	**Command Not Accepted Service Request Enable** 0_B Read: Command Not Valid Service Request disabled Write: Unchanged 1_B Read: Command Not Valid Service Request enabled Write: Enable Command Not Valid Service Request
SFBME	2	rwh	**SYNC Frames Below Minimum Service Request Enable** 0_B Read: SYNC Frames Below Minimum Service Request disabled Write: Unchanged 1_B Read: SYNC Frames Below Minimum Service Request enabled Write: Enable SYNC Frames Below Minimum Service Request
SFOE	3	rwh	**SYNC Frame Overflow Service Request Enable** 0_B Read: SYNC Frame Overflow Service Request disabled Write: Unchanged 1_B Read: SYNC Frame Overflow Service Request enabled Write: Enable Protocol Error Mode Changed Service Request
CCFE	4	rwh	**Clock Correction Failure Service Request Enable** 0_B Read: Clock Correction Failure Service Request disabled Write: Unchanged 1_B Read: Clock Correction Failure Service Request enabled Write: Enable Clock Correction Failure Service Request
CCLE	5	rwh	**CHI Command Locked Service Request Enable** 0_B Read: CHI Command Locked Service Request disabled Write: Unchanged 1_B Read: CHI Command Locked Service Request enabled Write: Enable CHI Command Locked Service Request
EERRE	6	rwh	**ECC Error Service Request Enable** 0_B Read: ECC Error Service Request disabled Write: Unchanged 1_B Read: ECC Error Service Request enabled Enable ECC Error Service Request Write: Unchanged
RFOE	7	rwh	**Receive FIFO Overrun Service Request Enable** 0_B Read: Receive FIFO Overrun Service Request disabled Write: Unchanged 1_B Read: Receive FIFO Overrun Service Request enabled Write: Enable Receive FIFO Overrun Service Request
EFAE	8	rwh	**Empty FIFO Access Service Request Enable** 0_B Read: Empty FIFO Access Service Request disabled Write: Unchanged 1_B Read: Empty FIFO Access Service Request enabled Write: Enable Empty FIFO Access Service Request

(table continues...)

(continued)

Field	Bits	Type	Description
IIBAE	9	rwh	**Illegal Input Buffer Access Service Request Enable** 0_B Read: Illegal Input Buffer Access Service Request disabled Write: Unchanged 1_B Read: Illegal Input Buffer Access Service Request enabled Write: Enable Illegal Input Buffer Access Service Request
IOBAE	10	rwh	**Illegal Output Buffer Access Service Request Enable** 0_B Read: Illegal Output Buffer Access Service Request disabled Write: Unchanged 1_B Read: Illegal Output Buffer Access Service Request enabled Write: Enable Illegal Output Buffer Access Service Request
MHFE	11	rwh	**Message Handler Constraints Flag Service Request Enable** 0_B Read: Message Handler Constraints Flag Service Request disabled Write: Unchanged 1_B Read: Message Handler Constraints Flag Service Request enabled Write: Enable Message Handler Constraints Flag Service Request
EDAE	16	rwh	**Error Detected on Channel A Service Request Enable** 0_B Read: Error Detected on Channel A Service Request disabled Write: Unchanged 1_B Read: Error Detected on Channel A Service Request enabled Write: Enable Error Detected on Channel A Service Request
LTVAE	17	rwh	**Latest Transmit Violation Channel A Service Request Enable** 0_B Read: Latest Transmit Violation Channel A Service Request disabled Write: Unchanged 1_B Read: Latest Transmit Violation Channel A Service Request enabled Write: Enable Latest Transmit Violation Channel A Service Request
TABAE	18	rwh	**Transmission Across Boundary Channel A Service Request Enable** 0_B Read: Transmission Across Boundary Channel A Service Request disabled Write: Unchanged 1_B Read: Transmission Across Boundary Channel A Service Request enabled Write: Enable Transmission Across Boundary Channel A Service Request
EDBE	24	rwh	**Error Detected on Channel B Service Request Enable** 0_B Read: Error Detected on Channel B Service Request disabled Write: Unchanged 1_B Read: Error Detected on Channel B Service Request enabled Write: Enable Error Detected on Channel B Service Request
LTVBE	25	rwh	**Latest Transmit Violation Channel B Service Request Enable** 0_B Read: Latest Transmit Violation Channel B Service Request disabled Write: Unchanged 1_B Read: Latest Transmit Violation Channel B Service Request enabled Write: Enable Latest Transmit Violation Channel B Service Request

(table continues...)

(continued)

Field	Bits	Type	Description
TABBE	26	rwh	**Transmission Across Boundary Channel B Service Request Enable** 0_B Read: Transmission Across Boundary Channel B Service Request disabled Write: Enable Transmission Across Boundary Channel B Service Request 1_B Read: Transmission Across Boundary Channel B Service Request enabled Write: Enable Transmission Across Boundary Channel B Service Request
0	15:12, 23:19, 31:27	r	**Reserved** Read as 0; should be written with 0.

23.4.15 Error Service Request Enable Reset

The settings in the Error Service Request Enable register determine which status changes in the Error Service Request Register will result in a service request. The enable bits are set by writing to EIES and reset by writing to EIER. Writing a 1 resets the specific enable bit, a 0 has no effect.

***Note**: LDMST or SWAPMSK.W instructions should be used only with bit mask enabled for all ´rwh´ bits in this register.*

EIER Offset address: 0034_H

Error Service Request Enable Reset Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0					TABBE	LTVBE	EDBE	0					TABAE	LTVAE	EDAE
r					rwh	rwh	rwh	r					rwh	rwh	rwh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				MHFE	IOBAE	IIBAE	EFAE	RFOE	EERRE	CCLE	CCFE	SFOE	SFBME	CNAE	PEMCE
r				rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
PEMCE	0	rwh	**POC Error Mode Changed Service Request Enable** 0_B Read: Protocol Error Mode Changed Service Request disabled Write: Unchanged 1_B Read: Protocol Error Mode Changed Service Request enabled Write: Disable Protocol Error Mode Changed Service Request
CNAE	1	rwh	**Command Not Accepted Service Request Enable** 0_B Read: Command Not Accepted Service Request disabled Write: Unchanged 1_B Read: Command Not Accepted Service Request enabled Write: Disable Command Not Accepted Service Request

(table continues...)

(continued)

Field	Bits	Type	Description
SFBME	2	rwh	**SYNC Frames Below Minimum Service Request Enable** 0_B Read: SYNC Frames Below Minimum Service Request disabled Write: Unchanged 1_B Read: SYNC Frames Below Minimum Service Request enabled Write: Disable SYNC Frames Below Minimum Service Request
SFOE	3	rwh	**SYNC Frame Overflow Service Request Enable** 0_B Read: SYNC Frame Overflow Service Request disabled Write: Unchanged 1_B Read: SYNC Frame Overflow Service Request enabled Write: Disable Protocol Error Mode Changed Service Request
CCFE	4	rwh	**Clock Correction Failure Service Request Enable** 0_B Read: Clock Correction Failure Service Request disabled Write: Unchanged 1_B Read: Clock Correction Failure Service Request enabled Write: Disable Clock Correction Failure Service Request
CCLE	5	rwh	**CHI Command Locked Service Request Enable** 0_B Read: CHI Command Locked Service Request disabled Write: Unchanged 1_B Read: CHI Command Locked Service Request enabled Write: Disable CHI Command Locked Service Request
EERRE	6	rwh	**ECC Error Service Request Enable** 0_B Read: ECC Error Service Request disabled Write: Unchanged 1_B ERead: CC Error Service Request enabled Write: Disable ECC Error Service Request
RFOE	7	rwh	**Receive FIFO Overrun Service Request Enable** 0_B Read: Receive FIFO Overrun Service Request disabled Write: Unchanged 1_B Read: Receive FIFO Overrun Service Request enabled Write: Disable Receive FIFO Overrun Service Request
EFAE	8	rwh	**Empty FIFO Access Service Request Enable** 0_B Read: Empty FIFO Access Service Request disabled Write: Unchanged 1_B Read: Empty FIFO Access Service Request enabled Write: Disable Empty FIFO Access Service Request
IIBAE	9	rwh	**Illegal Input Buffer Access Service Request Enable** 0_B Read: Illegal Input Buffer Access Service Request disabled Write: Unchanged 1_B Read: Illegal Input Buffer Access Service Request enabled Write: Disable Illegal Input Buffer Access Service Request
IOBAE	10	rwh	**Illegal Output Buffer Access Service Request Enable** 0_B Read: Illegal Output Buffer Access Service Request disabled Write: Unchanged 1_B Read: Illegal Output Buffer Access Service Request enabled Write: Disable Illegal Output Buffer Access Service Request

(table continues...)

(continued)

Field	Bits	Type	Description
MHFE	11	rwh	**Message Handler Constraints Flag Service Request Enable** 0_B Read: Message Handler Constraints Flag Service Request disabled Write: Unchanged 1_B Read: Message Handler Constraints Flag Service Request enabled Write: Disable Message Handler Constraints Flag Service Request
EDAE	16	rwh	**Error Detected on Channel A Service Request Enable** 0_B Read: Error Detected on Channel A Service Request disabled Write: Unchanged 1_B Read: Error Detected on Channel A Service Request enabled Write: Disable Error Detected on Channel A Service Request
LTVAE	17	rwh	**Latest Transmit Violation Channel A Service Request Enable** 0_B Read: Latest Transmit Violation Channel A Service Request disabled Write: Unchanged 1_B Read: Latest Transmit Violation Channel A Service Request enabled Write: Disable Latest Transmit Violation Channel A Service Request
TABAE	18	rwh	**Transmission Across Boundary Channel A Service Request Enable** 0_B Read: Transmission Across Boundary Channel A Service Request disabled Write: Unchanged 1_B Read: Transmission Across Boundary Channel A Service Request enabled Write: Enable Transmission Across Boundary Channel A Service Request
EDBE	24	rwh	**Error Detected on Channel B Service Request Enable** 0_B Read: Error Detected on Channel B Service Request disabled Write: Unchange 1_B Read: Error Detected on Channel B Service Request enabled Write: Disable Error Detected on Channel B Service Request
LTVBE	25	rwh	**Latest Transmit Violation Channel B Service Request Enable** 0_B Read: Latest Transmit Violation Channel B Service Request disabled Write: Unchanged 1_B Read: Latest Transmit Violation Channel B Service Request enabled Write: Disable Latest Transmit Violation Channel B Service Request
TABBE	26	rwh	**Transmission Across Boundary Channel B Service Request Enable** 0_B Read: Transmission Across Boundary Channel B Service Request disabled Write: Unchanged 1_B Read: Transmission Across Boundary Channel B Service Request enabled Write: Disable Transmission Across Boundary Channel B Service Request
0	15:12, 23:19, 31:27	r	**Reserved** Read as 0; should be written with 0.

23.4.16 Status Service Request Enable Set

The settings in the Status Service Request Enable Set register determine which status changes in the Status Service Request Register will result in a service request. The enable bits are set by writing to SIES and reset by writing to SIER. Writing a 1 sets the specific enable bit, a 0 has no effect.

***Note**: LDMST or SWAPMSK.W instructions should be used only with bit mask enabled for all ´rwh´ bits in this register.*

SIES Offset address: 0038_H

Status Service Request Enable Set Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0						MTSBE	WUPBE	0						MTSAE	WUPAE
r						rwh	rwh	r						rwh	rwh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SDSE	MBSIE	SUCSE	SWEE	TOBCE	TIBCE	TI1E	TI0E	NMVCE	RFCLE	RFNEE	RXIE	TXIE	CYCSE	CASE	WSTE
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
WSTE	0	rwh	**Wakeup Status Service Request Enable** 0_B Read: Wake-up Status Service Request disabled Write: Unchanged 1_B Read: Wake-up Status Service Request enabled Write: Enable Wakeup Status Service Request
CASE	1	rwh	**Collision Avoidance Symbol Service Request Enable** 0_B Read: Collision Avoidance Symbol Service Request disabled Write: Unchanged 1_B Read: Collision Avoidance Symbol Service Request enabled Write: Enable Collision Avoidance Symbol Service Request
CYCSE	2	rwh	**Cycle Start Service Request Enable** 0_B Read: Cycle Start Service Request disabled Write: Unchanged 1_B Read: Cycle Start Service Request enabled Write: Enable Cycle Start Service Request
TXIE	3	rwh	**Transmit Service Request Enable** 0_B Read: Transmit Service Request disabled Write: Unchanged 1_B Transmit Service Request enabled Write: Enable Transmit Service Request
RXIE	4	rwh	**Receive Service Request Enable** 0_B Read: Receive Service Request disabled Write: Unchanged 1_B Read: Receive Service Request enabled Write: Enable Receive Service Request

(table continues...)

(continued)

Field	Bits	Type	Description
RFNEE	5	rwh	**Receive FIFO Not Empty Service Request Enable** 0_B Read: Receive FIFO Not Empty Service Request disabled Write: Unchanged 1_B Read: Receive FIFO Not Empty Service Request enabled Write: Enable Receive FIFO Not Empty Service Request
RFCLE	6	rwh	**Receive FIFO Critical Level Service Request Enable** 0_B Read: Receive FIFO Critical Level Service Request disabled Write: Unchanged 1_B Read: Receive FIFO Critical Level Service Request enabled Write: Enable Receive FIFO Critical Level Service Request
NMVCE	7	rwh	**Network Management Vector Changed Service Request Enable** 0_B Read: Network Management Vector Changed Service Request disabled Write: Unchanged 1_B Read: Network Management Vector Changed Service Request enabled Write: Enable Network Management Vector Changed Service Request
TI0E	8	rwh	**Timer Service Request 0 Enable** 0_B Read: Timer Service Request 0 disabled Write: Unchanged 1_B Read: Timer Service Request 0 enabled Write: Enable Timer Service Request 0
TI1E	9	rwh	**Timer Service Request 1 Enable** 0_B Read: Timer Service Request 1 disabled Write: Unchanged 1_B Read: Timer Service Request 1 enabled Write: Enable Timer Service Request 1
TIBCE	10	rwh	**Transfer Input Buffer Completed Service Request Enable** 0_B Read: Wakeup Status Service Request disabled Write: Unchanged 1_B Read: Wakeup Status Service Request enabled Write: Enable Wakeup Status Service Request
TOBCE	11	rwh	**Transfer Output Buffer Completed Service Request Enable** 0_B Read: Transfer Input Buffer Completed Service Request disabled Write: Unchanged 1_B Read: Transfer Input Buffer Completed Service Request enabled Write: Enable Transfer Input Buffer Completed Service Request
SWEE	12	rwh	**Stop Watch Event Service Request Enable** 0_B Read: Stop Watch Event Service Request disabled Write: Unchanged 1_B Read: Stop Watch Event Service Request enabled Write: Enable Stop Watch Event Service Request
SUCSE	13	rwh	**Startup Completed Successfully Service Request Enable** 0_B Read: Startup Completed Successfully Service Request disabled Write: Unchanged 1_B Read: Startup Completed Successfully Service Request enabled Write: Enable Startup Completed Successfully Service Request

(table continues...)

(continued)

Field	Bits	Type	Description
MBSIE	14	rwh	**Message Buffer Status Service Request Enable** 0_B Read: Message Buffer Status Service Request disabled Write: Unchanged 1_B Read: Message Buffer Status Service Request enabled Write: Enable Message Buffer Status Service Request
SDSE	15	rwh	**Start of Dynamic Segment Service Request Enable** 0_B Read: Start of Dynamic Service Request disabled Write: Unchanged 1_B Read: Start of Dynamic Service Request enabled Write: Enable Start of Dynamic Service Request
WUPAE	16	rwh	**Wakeup Pattern Channel A Service Request Enable** 0_B Read: Wakeup Pattern Channel A Service Request disabled Write: Unchanged 1_B Read: Wakeup Pattern Channel A Service Request enabled Write: Enable Wakeup Pattern Channel A Service Request
MTSAE	17	rwh	**Media Access Test Symbol Channel A Service Request Enable** 0_B Read: Media Access Test Symbol Channel A Service Request disabled Write: Unchanged 1_B Read: Media Access Test Symbol Channel A Service Request enabled Write: Enable Media Access Test Symbol Channel A Service Request
WUPBE	24	rwh	**Wakeup Pattern Channel B Service Request Enable** 0_B Read: Wakeup Pattern Channel B Service Request disabled Write: Unchanged 1_B Read: Wakeup Pattern Channel B Service Request enabled Write: Enable Wakeup Pattern Channel B Service Request
MTSBE	25	rwh	**Media Access Test Symbol Channel B Service Request Enable** 0_B Read: Media Access Test Symbol Channel B Service Request disabled Write: Unchanged 1_B Read: Media Access Test Symbol Channel B Service Request enabled Write: Enable Media Access Test Symbol Channel B Service Request
0	23:18, 31:26	r	**Reserved** Read as 0; should be written with 0.

23.4.17 Status Service Request Enable Reset

The settings in the Status Service Request Enable Reset register determine which status changes in the Status Service Request Register will result in a service request. The enable bits are set by writing to SIES and reset by writing to SIER. Writing a 1 resets the specific enable bit, a 0 has no effect.

***Note**: LDMST or SWAPMSK.W instructions should be used only with bit mask enabled for all ´rwh´ bits in this register.*

SIER Offset address: $003C_H$

Status Service Request Enable Reset Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0						MTSBE	WUPBE	0						MTSAE	WUPAE
r						rwh	rwh	r						rwh	rwh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SDSE	MBSIE	SUCSE	SWEE	TOBCE	TIBCE	TI1E	TI0E	NMVCE	RFCLE	RFNEE	RXIE	TXIE	CYCSE	CASE	WSTE
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
WSTE	0	rwh	**Wakeup Status Service Request Enable** 0_B Read: Wakeup Status Service Request disabled Write: Unchanged 1_B Read: Wakeup Status Service Request enabled Write: Disable Wakeup Status Service Request
CASE	1	rwh	**Collision Avoidance Symbol Service Request Enable** 0_B Read: Collision Avoidance Symbol Service Request disabled Write: Unchanged 1_B Read: Collision Avoidance Symbol Service Request enabled Write: Disable Collision Avoidance Symbol Service Request
CYCSE	2	rwh	**Cycle Start Service Request Enable** 0_B Read: Cycle Start Service Request disabled Write: Unchanged 1_B Read: Cycle Start Service Request enabled Write: Disable Cycle Start Service Request
TXIE	3	rwh	**Transmit Service Request Enable** 0_B Read: Transmit Service Request disabled Write: Unchanged 1_B Read: Transmit Service Request enabled Write: Disable Transmit Service Request
RXIE	4	rwh	**Receive Service Request Enable** 0_B Read: Receive Service Request disabled Write: Unchanged 1_B Read: Receive Service Request enabled Write: Disable Receive Service Request
RFNEE	5	rwh	**Receive FIFO Not Empty Service Request Enable** 0_B Read: Receive FIFO Not Empty Service Request disabled Write: Unchanged 1_B Read: Receive FIFO Not Empty Service Request enabled Write: Disable Receive FIFO Not Empty Service Request
RFCLE	6	rwh	**Receive FIFO Critical Level Service Request Enable** 0_B Read: Service Request disabled Write: Unchanged 1_B Read: Receive FIFO Critical Level Service Request enabled Write: Disable Receive FIFO Critical Level Service Request

(table continues...)

(continued)

Field	Bits	Type	Description
NMVCE	7	rwh	**Network Management Vector Changed Service Request Enable** 0_B Read: Network Management Vector Changed Service Request disabled Write: Unchanged 1_B Read: Network Management Vector Changed Service Request enabled Write: Disable Network Management Vector Changed Service Request
TI0E	8	rwh	**Timer Service Request 0 Enable** 0_B Read: Timer Service Request 0 disabled Write: Unchanged 1_B Read: Timer Service Request 0 enabled Write: Disable Service Request 0
TI1E	9	rwh	**Timer Service Request 1 Enable** 0_B Read: Timer Service Request 1 disabled Write: Unchanged 1_B Read: Timer Service Request 1 enabled Write: Disable Timer Service Request 1
TIBCE	10	rwh	**Transfer Input Buffer Completed Service Request Enable** 0_B Read: Wakeup Status Service Request disabled Write: Unchanged 1_B Read: Wakeup Status Service Request enabled Write: Disable Wakeup Status Service Request
TOBCE	11	rwh	**Transfer Output Buffer Completed Service Request Enable** 0_B Read: Transfer Input Buffer Completed Service Request disabled Write: Unchanged 1_B Read: Transfer Input Buffer Completed Service Request enabled Write: Disable Transfer Input Buffer Completed Service Request
SWEE	12	rwh	**Stop Watch Event Service Request Enable** 0_B Read: Stop Watch Event Service Request disabled Write: Unchanged 1_B Read: Stop Watch Event Service Request enabled Write: Disable Stop Watch Event Service Request
SUCSE	13	rwh	**Startup Completed Successfully Service Request Enable** 0_B Read: Startup Completed Successfully Service Request disabled Write: Unchanged 1_B Read: Startup Completed Successfully Service Request enabled Write: Disable Startup Completed Successfully Service Request
MBSIE	14	rwh	**Message Buffer Status Service Request Enable** 0_B Read: Message Buffer Status Service Request disabled Write: Unchanged 1_B Read: Message Buffer Status Service Request enabled Write: Disable Message Buffer Status Service Request
SDSE	15	rwh	**Start of Dynamic Segment Service Request Enable** 0_B Read: Start of Dynamic Service Request disabled Write: Unchanged 1_B Read: Start of Dynamic Service Request enabled Write: Disable Start of Dynamic Service Reques

(table continues...)

(continued)

Field	Bits	Type	Description
WUPAE	16	rwh	**Wakeup Pattern Channel A Service Request Enable** 0_B Read: Wakeup Pattern Channel A Service Request disabled Write: Unchanged 1_B Read: Wakeup Pattern Channel A Service Request enabled Write: Disable Wakeup Pattern Channel A Service Request
MTSAE	17	rwh	**Media Access Test Symbol Channel A Service Request Enable** 0_B Read: Media Access Test Symbol Channel A Service Request disabled Write: Unchanged 1_B Read: Media Access Test Symbol Channel A Service Request enabled Write: Disable Media Access Test Symbol Channel A Service Request
WUPBE	24	rwh	**Wakeup Pattern Channel B Service Request Enable** 0_B Read: Wakeup Pattern Channel B Service Request disabled Write: Unchanged 1_B Read: Wakeup Pattern Channel B Service Request enabled Write: Disable Wakeup Pattern Channel B Service Request
MTSBE	25	rwh	**Media Access Test Symbol Channel B Service Request Enable** 0_B Read: Media Access Test Symbol Channel B Service Request disabled Write: Unchanged 1_B Read: Media Access Test Symbol Channel B Service Request enabled Write: Disable Media Access Test Symbol Channel B Service Request
0	23:18, 31:26	r	**Reserved** Read as 0; should be written with 0.

23.4.18 Service Request Line Enable

Each of the two service request lines to the Host INT0SRC, INT1SRC can be enabled / disabled separately by programming bit EINT0 and EINT1.

ILE Offset address: 0040_H

Service Request Line Enable Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0														EINT1	EINT0
r														rw	rw

Field	Bits	Type	Description
EINT0	0	rw	**Enable Service Request Line 0 (INT0SRC)** 0_B Service Request line disabled 1_B Service Request line enabled
EINT1	1	rw	**Enable Service Request Line 1 (INT1SRC)** 0_B Service Request line disabled 1_B Service Request line enabled
0	31:2	r	**Reserved** Read as 0; should be written with 0.

23.4.19 Timer 0 Configuration

Absolute timer. Specifies in terms of cycle count and Macrotick the point in time when the timer 0 service request occurs. When the timer 0 service request is asserted, output signal TINT0SR is set to 1 for the duration of one Macrotick and SIR.TI0 is set to 1.

Timer 0 can be activated as long as the POC is either in “NORMAL_ACTIVE” state or in “NORMAL_PASSIVE” state. Timer 0 is deactivated when leaving “NORMAL_ACTIVE” state or “NORMAL_PASSIVE” state except for transitions between the two states.

Before reconfiguration of the timer, the timer has to be halted first by writing 0 to bit T0RC.

***Note**: The configuration of timer 0 is compared against the Macrotick counter value, there is no separate counter for timer 0. In case the Communication Controller leaves “NORMAL_ACTIVE” or “NORMAL_PASSIVE” state, or if timer 0 is halted by Host command, output signal TINT0SR is reset to 0 immediately.*

T0C Offset address: 0044_H

Timer 0 Configuration Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		T0MO													
r		rw													

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	T0CC							0						T0MS	T0RC
r	rw							r						rw	rwh

Field	Bits	Type	Description
T0RC	0	rwh	**Timer 0 Run Control** 0_B Timer 0 halted 1_B Timer 0 running
T0MS	1	rw	**Timer 0 Mode Select** 0_B Single-shot mode 1_B Continuous mode

(table continues...)

(continued)

Field	Bits	Type	Description
T0CC	14:8	rw	**Timer 0 Cycle Code** The 7-bit timer 0 cycle code determines the cycle set used for generation of the timer 0 service request. For details about the configuration of the cycle code see “Cycle Counter Filtering”.
T0MO	29:16	rw	**Timer 0 Macrotick Offset** Configures the Macrotick offset from the beginning of the cycle where the service request is to occur. The Timer 0 Service Request occurs at this offset for each cycle of the cycle set.
0	7:2, 15, 31:30	r	**Reserved** Read as 0; should be written with 0.

23.4.20 Timer 1 Configuration

Relative timer. After the specified number of Macroticks has expired, the timer 1 service request is asserted, output signal TINT1SR is set to 1 for the duration of one Macrotick and SIR.TI1 is set to 1.

Timer 1 can be activated as long as the POC is either in “NORMAL_ACTIVE” state or in “NORMAL_PASSIVE” state. Timer 1 is deactivated when leaving “NORMAL_ACTIVE” state or “NORMAL_PASSIVE” state except for transitions between the two states.

Before reconfiguration of the timer, the timer has to be halted first by resetting bit T1RC to 0.

***Note**: In case the Communication Controller leaves “NORMAL_ACTIVE” or “NORMAL_PASSIVE” state, or if timer 1 is halted by Host command, output signal TINT1SR is reset to 0 immediately.*

T1C Offset address: 0048_H

Timer 1 Configuration Kernel Reset value: $0002\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		T1MC													
r		rw													

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0														T1MS	T1RC
r														rw	rwh

Field	Bits	Type	Description
T1RC	0	rwh	**Timer 1 Run Control** 0_B Timer 1 halted 1_B Timer 1 running
T1MS	1	rw	**Timer 1 Mode Select** 0_B Single-shot mode 1_B Continuous mode

(table continues...)

(continued)

Field	Bits	Type	Description
T1MC	29:16	rw	**Timer 1 Macrotick Count** When the configured Macrotick count is reached the timer 1 service request is generated. Valid values are: 2_H …$3FFF_H$ Macroticks in continuous mode 1_H …$3FFF_H$ Macroticks in single-shot mode
0	15:2, 31:30	r	**Reserved** Read as 0; should be written with 0.

23.4.21 Stop Watch Register 1

The stop watch is activated by a rising or falling edge on signal STPW, by a service request 0 or 1 event (rising edge on signal INT0SR or INT1SR) or by the Host by writing bit STPW1.SSWT to 1. With the Macrotick counter increment following next to the stop watch activation the actual cycle counter and Macrotick value are captured in the Stop Watch Register 1 STPW1 while the slot counter values for channel A and B are captured in the Stop Watch Register 2 STPW2.

***Note**: Bits ESWT and SSWT cannot be set to 1 simultaneously. In this case the write access is ignored, and both bits keep their previous values. Therefore either the external stop watch triggers or the software stop watch trigger may be used.*

STPW1 Offset address: $004C_H$

Stop Watch Register 1 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		SMTV													
r		rh													

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		SCCV						0	EINT1	EINT0	EETP	SSWT	EDGE	SWMS	ESWT
r		rh						r	rw	rw	rw	rwh	rw	rw	rwh

Field	Bits	Type	Description
ESWT	0	rwh	**Enable Stop Watch Trigger** If enabled, an edge on input STPW or a service request 0 or 1 event (rising edge on signal INT0SR or INT1SR)activates the stop watch. In single-shot mode this bit is reset to 0 after the actual cycle counter and Macrotick value are stored in the Stop Watch register. 0_B Stop watch trigger disabled 1_B Stop watch trigger enabled
SWMS	1	rw	**Stop Watch Mode Select** It is not possible to change the Stop Watch Mode during enabled stop watch trigger (STPW1.ESWT) 0_B Single-shot mode 1_B Continuous mode

(table continues...)

(continued)

Field	Bits	Type	Description
EDGE	2	rw	**Stop Watch Trigger Edge Select** 0_B Falling Edge 1_B Rising Edge
SSWT	3	rwh	**Software Stop Watch Trigger** When the Host writes this bit to 1 the stop watch is activated. After the actual cycle counter and Macrotick value are stored in the Stop Watch register this bit is reset to 0. The bit is only writeable while ESWT = 0. 0_B Software trigger reset 1_B Stop watch activated by software trigger
EETP	4	rw	**Enable External Trigger Pin** Enables stop watch trigger event through signal STPW if ESWT = 1. 0_B Stop watch trigger through signal STPW disabled 1_B Edge on signal STPW triggers stop watch
EINT0	5	rw	**Enable Service Request 0 Trigger** Enables stop watch trigger by service request 0 event if ESWT = 1. 0_B Stop watch trigger by service request 0 disabled 1_B Service Request 0 event triggers stop watch
EINT1	6	rw	**Enable Service Request 1 Trigger** Enables stop watch trigger by service request 1 event if ESWT = 1. 0_B Stop watch trigger by service request 1 disabled 1_B Service Request 1 event triggers stop watch
SCCV	13:8	rh	**Stopped Cycle Counter Value** State of the cycle counter when the stop watch event occurred. Valid values are: $0…3F_H$ Valid Values
SMTV	29:16	rh	**Stopped Macrotick Value** State of the Macrotick counter when the stop watch event occurred. Valid values are: $0…3F_H$ Valid Values
0	7, 15:14, 31:30	r	**Reserved** Read as 0; should be written with 0.

23.4.22 Stop Watch Register 2

STPW2 Offset address: 0050_H

Stop Watch Register 2 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0					SSCVB										
r					rh										

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0					SSCVA										
r					rh										

Field	Bits	Type	Description
SSCVA	10:0	rh	**Stop Watch Captured Slot Counter Value Channel A** State of the slot counter for channel A when the stop watch event occurred. Valid values are 0 to 2047 (0_H to $7FF_H$).
SSCVB	26:16	rh	**Stop Watch Captured Slot Counter Value Channel B** State of the slot counter for channel B when the stop watch event occurred. Valid values are 0 to 2047 (0_H to $7FF_H$).
0	15:11, 31:27	r	**Reserved** Read as 0; should be written with 0.

23.4.23 SUC Configuration Register 1

***Note**: The protocol requires that both bits TXST and TXSY are set for coldstart nodes (For bits TXST, TXSY, MTSB).*

MTSA and MTSB may also be changed outside "DEFAULT_CONFIG" or "CONFIG" state when the write to SUCC1 register is directly preceded by the unlock sequence as described in Lock Register (LCK). This may be combined with CHI command "SEND_MTS". If both bits MTSA and MTSB are set to 1, an MTS symbol will be transmitted on both channels when requested by writing SUCC1.CMD = 1000B (For MTSA).

SUCC1 Offset address: 0080_H

SUC Configuration Register 1 Kernel Reset value: $0C40\ 1000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0				CCHB	CCHA	MTSB	MTSA	HCSE	TSM	WUCS	PTA				
r				rw	rw	rw	rw	rw	rw	rw	rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CSA					0	TXSY	TXST	PBSY	0			CMD			
rw					r	rw	rw	rh	r			rwh			

Field	Bits	Type	Description
CMD	3:0	rwh	**CHI Command Vector** The host may write any CHI command at any time, but certain commands are only enabled in specific POC states. A disabled command will not be executed, the CHI command vector CMD will be reset to 0000_B = “COMMAND_NOT_ACCEPTED”, and flag EIR.CNA in the Error Service Request register will be set to 1. In case the previous CHI command has not yet completed, EIR.CCL is set to 1 together with EIR.CNA; the CHI command needs to be repeated. Except for HALT state, POC state change command applied while the Communication Controller is already in the requested POC state will be ignored. Reading SUCC1.CMD shows whether the last CHI command was accepted. CCSV.POCS monitors the actual POC state. The reserved CHI commands code hardware test functions. 0_H COMMAND_NOT_ACCEPTED 1_H CONFIG 2_H READY 3_H WAKEUP 4_H RUN 5_H ALL_SLOTS 6_H HALT 7_H FREEZE 8_H SEND_MTS 9_H ALLOW_COLDSTART A_H RESET_STATUS_INDICATORS B_H MONITOR_MODE C_H CLEAR_RAMS D_H Reserved … F_H Reserved
PBSY	7	rh	**POC Busy** Signals that the POC is busy and cannot accept a command from the Host. SUCC1.CMD is locked against write accesses. Set to 1 after hard reset during initialization of internal RAM blocks. 0_B POC not busy, SUCC1.CMD writable 1_B POC is busy, SUCC1.CMD locked
TXST	8	rw	**Transmit Startup Frame in Key Slot (pKeySlotUsedForStartup)** Defines whether the key slot is used to transmit startup Frames. The bit can be modified in “DEFAULT_CONFIG” or “CONFIG” state only. 0_B No Startup Frame transmission in key slot, node is non-coldstarter 1_B Key slot used to transmit startup Frame, node is leading or following coldstarter

(table continues...)

(continued)

Field	Bits	Type	Description
TXSY	9	rw	**Transmit SYNC Frame in Key Slot (pKeySlotUsedForSync)** Defines whether the key slot is used to transmit SYNC Frames. The bit can be modified in “DEFAULT_CONFIG” or “CONFIG” state only. 0_B No SYNC Frame transmission in key slot, node is neither sync nor coldstart node 1_B Key slot used to transmit SYNC Frames, node is sync node
CSA	15:11	rw	**Cold Start Attempts (gColdStartAttempts)** Configures the maximum number of attempts that a cold starting node is permitted to try to start up the network without receiving any valid response from another node. It can be modified in “DEFAULT_CONFIG” or “CONFIG” state only. Must be identical in all nodes of a cluster. Valid values are 2 to 31.
PTA	20:16	rw	**Passive to Active (pAllowPassiveToActive)** Defines the number of consecutive even / odd cycle pairs that must have valid clock correction terms before the Communication Controller is allowed to transit from “NORMAL_PASSIVE” to “NORMAL_ACTIVE” state. If set to 00000_B the Communication Controller is not allowed to transit from “NORMAL_PASSIVE” to “NORMAL_ACTIVE” state. It can be modified in “DEFAULT_CONFIG” or “CONFIG” state only. Valid values are 0 to 31 even / odd cycle pairs.
WUCS	21	rw	**Wakeup Channel Select (pWakeupChannel)** With this bit the Host selects the channel on which the Communication Controller sends the Wakeup pattern. The Communication Controller ignores any attempt to change the status of this bit when not in “DEFAULT_CONFIG” or “CONFIG” state. 0_B Send wakeup pattern on channel A 1_B Send wakeup pattern on channel B
TSM	22	rw	**Transmission Slot Mode (pSingleSlotEnabled)** Selects the initial transmission slot mode. In SINGLE slot mode the Communication Controller may only transmit in the preconfigured key slot. The key slot ID is configured in the Header Section of Message Buffer 0 respectively Message Buffers 0 and 1 depending on bit MRC.SPLM. In case SUCC1.TSM = 1, Message Buffer 0 respectively Message Buffers 0, 1 can be (re)configured in “DEFAULT_CONFIG” or “CONFIG” state only. In ALL slot mode the Communication Controller may transmit in all slots. The bit can be written in “DEFAULT_CONFIG” or “CONFIG” state only. The communication controller changes to ALL slot mode when the Host successfully applied the ALL_SLOTS command by writing SUCC1.CMD = 0101_B in POC states “NORMAL_ACTIVE” or “NORMAL_PASSIVE”. The actual slot mode is monitored by CCSV.SLM. 0_B ALL Slot Mode 1_B SINGLE Slot Mode (default after application reset)

(table continues...)

(continued)

Field	Bits	Type	Description
HCSE	23	rw	**Halt due to Clock Sync Error (pAllowHaltDueToClock)** Controls the transition to “HALT” state due to a clock synchronization error. The bit can be modified in “DEFAULT_CONFIG” or “CONFIG” state only. 0_B Communication Controller will enter / remain in “NORMAL_PASSIVE” 1_B Communication Controller will enter “HALT” state
MTSA	24	rw	**Select Channel A for MTS Transmission** The bit selects channel A for MTS symbol transmission. The flag is reset by default and may be modified only in “DEFAULT_CONFIG” or “CONFIG” state. 0_B Channel A disabled for MTS transmission 1_B Channel A selected for MTS transmission
MTSB	25	rw	**Select Channel B for MTS Transmission** The bit selects channel B for MTS symbol transmission. The flag is reset by default and may be modified only in “DEFAULT_CONFIG” or “CONFIG” state. [2)] 0_B Channel B disabled for MTS transmission 1_B Channel B selected for MTS transmission
CCHA	26	rw	**Connected to Channel A (pChannels)** Configures whether the node is connected to channel A. This bit can be modified in "DEFAULT_CONFIG" or "CONFIG" state only. 0_B Not connected to channel A 1_B Node connected to channel A (default after application reset)
CCHB	27	rw	**Connected to Channel B (pChannels)** Configures whether the node is connected to channel B. This bit can be modified in "DEFAULT_CONFIG" or "CONFIG" state only. 0_B Not connected to channel B 1_B Node connected to channel B (default after application reset)
0	6:4, 10, 31:28	r	**Reserved** Read as 0; should be written with 0.

COMMAND_NOT_ACCEPTED

SUCC1.CMD is reset to 0000_B due to one of the following conditions:

- Illegal command applied by the Host
- Host writes “COMMAND_NOT_ACCEPTED”
- Host applied new command while execution of the previous Host command has not completed

When SUCC1.CMD is reset to 0000_B, bit EIR.CNA in the Error Service Request register is set, and - if enabled - an service request is generated. Commands which are not accepted are not executed.

CONFIG

Go to POC state “CONFIG” when called in POC states “DEFAULT_CONFIG“, “READY“, or in “MONITOR_MODE“. When called in “HALT” state transits to POC state “DEFAULT_CONFIG“. When called in any other state, SUCC1.CMD will be reset to 0000_B = “COMMAND_NOT_ACCEPTED”.

READY

Go to POC state “READY” when called in POC states “CONFIG”, “NORMAL_ACTIVE”, “NORMAL_PASSIVE”, “STARTUP”, or “WAKEUP”. When called in any other state, SUCC1.CMD will be reset to 0000_B = “COMMAND_NOT_ACCEPTED”.

WAKEUP

Go to POC state WAKEUP when called in POC state “READY”. When called in any other state, SUCC1.CMD will be reset to 0000_B = “COMMAND_NOT_ACCEPTED”.

RUN

Go to POC state “STARTUP” when called in POC state “READY”. When called in any other state, SUCC1.CMD will be reset to 0000_B = “COMMAND_NOT_ACCEPTED”.

ALL_SLOTS

Leave SINGLE slot mode after successful startup / integration at the next end of cycle when called in POC states “NORMAL_ACTIVE” or “NORMAL_PASSIVE”. When called in any other state, SUCC1.CMD will be reset to 0000_B = “COMMAND_NOT_ACCEPTED”.

HALT

Set the halt request CCSV.HRQ bit in the Communication Controller Status Vector register and go to POC state “HALT” at the next end of cycle when called in POC states “NORMAL_ACTIVE” or “NORMAL_PASSIVE“. When called in any other state, SUCC1.CMD will be reset to 0000_B = “COMMAND_NOT_ACCEPTED“.

FREEZE

Set the freeze status indicator CCSV.FSI and go to POC state “HALT” immediately. Can be called from any state.

SEND_MTS

Send single MTS symbol during the next following symbol window on the channel configured by SUCC1.MTSA, SUCC1.MTSB, when called in POC state “NORMAL_ACTIVE”. When called in any other state, SUCC1.CMD will be reset to 0000_B = “COMMAND_NOT_ACCEPTED”.

ALLOW_COLDSTART

The command resets bit CCSV.CSI to enable the node to become cold starter. When called in states “DEFAULT_CONFIG“, “CONFIG”, “HALT”, or “MONITOR_MODE“. SUCC1.CMD will be reset to 0000_B = “COMMAND_NOT_ACCEPTED”. To become leading coldstarter it is also required that both TXST and TXSY are set.

RESET_STATUS_INDICATORS

Resets status flags CCSV.CSNI, CCSV.CSAI, CCSV.WSV to their default values. May be called in POC state READY. When called in any other state, SUCC1.CMD will be reset to 0000_B = “COMMAND_NOT_ACCEPTED”.

MONITOR_MODE

Enter MONITOR_MODE when called in POC state CONFIG. In this mode the Communication Controller is able to receive FlexRay™ Frames and wakeup pattern. It is also able to detect coding errors. The temporal integrity of received Frames is not checked. This mode can be used for debugging purposes, e.g. in case that the startup of a FlexRay™ network fails. When called in any other state, SUCC1.CMD will be reset to 0000_B = “COMMAND_NOT_ACCEPTED”. For details see “MONITOR_MODE”.

CLEAR_RAMS

Sets bit MHDS.CRAM in the Message Handler Status register when called in “DEFAULT_CONFIG” or “CONFIG” state. When called in any other state, SUCC1.CMD will be reset to 0000_B = “COMMAND_NOT_ACCEPTED”. By setting MHDS.CRAM all internal RAM blocks are initialized to zero. Note that only the currently active IBF bank is cleared. To clear the 2nd bank as well, CUST1.IBF1PAG and CUST1.IBF2PAG need to be set and command CLEAR_RAMS need to be issued again. This is required in particular after an application reset. If the 2nd bank of IBF is left unused, this procedure is not required. During the initialization of the RAMs, SUCC1.PBSY will show POC busy. Access to the configuration and status registers is possible during execution of CHI command CLEAR_RAMS.

The initialization of the ERAY internal RAM blocks requires 2048 f_{CLC_ERAY} cycles. There should be no Host access to IBF or OBF during initialization of the internal RAM blocks after application reset or after assertion of CHI command CLEAR_RAMS. Before asserting CHI command CLEAR_RAMS the Host should make sure that no transfer between Message RAM and IBF / OBF or the Transient Buffer RAMs is ongoing. This command also resets the Message Buffer Status registers MHDS, LDTS, FSR, MHDF, TXRQ1, TXRQ2, TXRQ3, TXRQ4, NDAT1, NDAT2, NDAT3, NDAT4, MBSC1, MBSC2, MBSC3, and MBSC4.

Table below references the CHI commands from the FlexRay™ Protocol Specification v2.1 (section 2.2.1.1, Table 2-2) to the ERAY CHI command vector CMD.

Table 1096 Reference to CHI Host command summary from FlexRay™ protocol specification

CHI Command	Where processed (POC State)	CHI Command Vector CMD
ALL_SLOT	POC:NORMAL_ACTIVE, POC:NORMAL_PASSIVE	ALL_SLOTS
ALLOW_COLDSTART	All except POC:DEFAULT_CONFIG, POC:CONFIG, POC:HALT	ALLOW_COLDSTART
CONFIG	POC:DEFAULT_CONFIG, POC:READY	CONFIG
CONFIG_COMPLETE	POC:CONFIG	Unlock sequence & READY
DEFAULT_CONFIG	POC:HALT	CONFIG
FREEZE	All	FREEZE
HALT	POC:NORMAL_ACTIVE, POC:NORMAL_PASSIVE	HALT
READY	All except POC:DEFAULT_CONFIG, POC:CONFIG, POC:READY, POC:HALT	READY
RUN	POC:READY	RUN
WAKEUP	POC:READY	WAKEUP

23.4.24 SUC Configuration Register 2

The Communication Controller accepts modifications of the register in “DEFAULT_CONFIG” or “CONFIG” state only.

***Note**: The wakeup / startup noise time-out is calculated as follows: The wakeup / startup noise time-out = pdListenTimeout • gListenNoise = LT • (LTN + 1)*

SUCC2 Offset address: 0084_H

SUC Configuration Register 2 Kernel Reset value: $0100\ 0504_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0				LTN				0			LT				
r				rw				r			rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LT															
rw															

Field	Bits	Type	Description
LT	20:0	rw	**Listen Timeout (pdListenTimeout)** Configures wakeup / startup listen timeout in Microticks. The range for wakeup / startup listen timeout (pdListenTimeout) is 1284 to 1283846 (504_H to 139706_H) Microticks

(table continues...)

(continued)

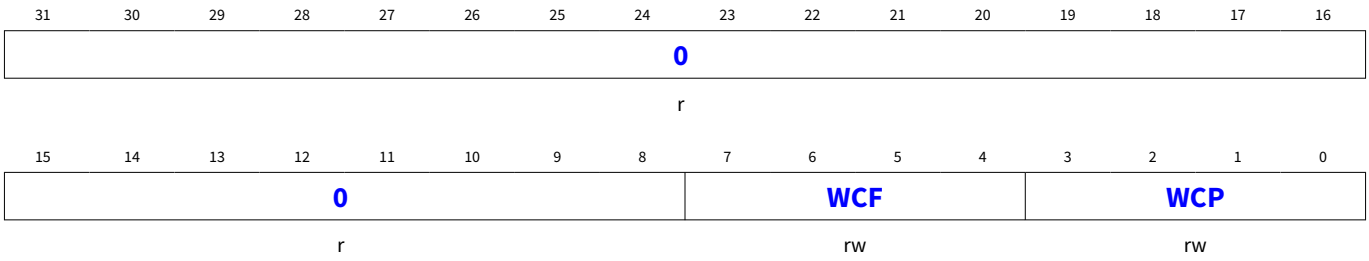
Field	Bits	Type	Description
LTN	27:24	rw	**Listen Time-out Noise (gListenNoise - 1)** Configures the upper limit for startup and wakeup listen timeout in the presence of noise expressed as a multiple of the cluster constant pdListenTimeout. The range of pdListenTimeout 2 to 16. LTN must be configured identical in all nodes of a cluster. This bit can be updated in "DEFAULT_CONFIG" or "CONFIG" state only. 1_H Listen Time-out Noise is equal 2 2_H Listen Time-out Noise is equal 3 3_H … E_H Listen Time-out Noise is equal 3 … 15 F_H Listen Time-out Noise is equal 16
0	23:21, 31:28	r	**Reserved** Read as 0; should be written with 0.

23.4.25 SUC Configuration Register 3

The Communication Controller accepts modifications of the register in “DEFAULT_CONFIG” or “CONFIG” state only.

SUCC3 Offset address: 0088_H

SUC Configuration Register 3 Kernel Reset value: $0000\ 0011_H$



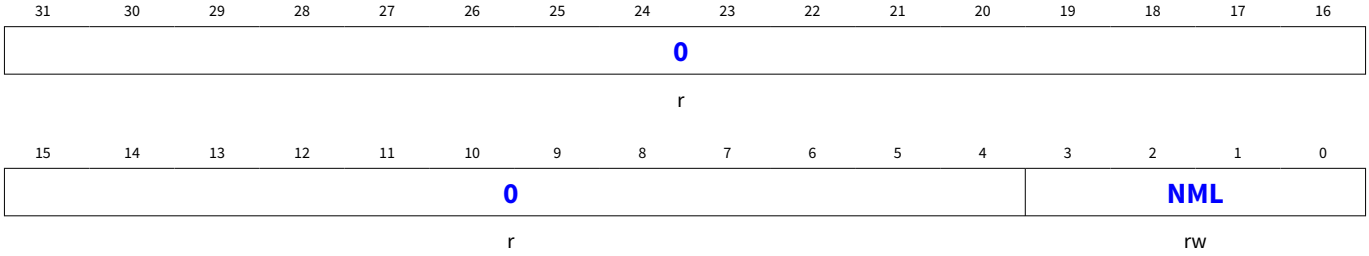
Field	Bits	Type	Description
WCP	3:0	rw	**Maximum Without Clock Correction Passive** Defines the number of consecutive even / odd cycle pairs with missing clock correction terms that will cause a transition from “NORMAL_ACTIVE” to “NORMAL_PASSIVE” state. Must be identical in all nodes of a cluster. Valid values are 1 to 15 (1_H to F_H) cycle pairs. (gMaxWithoutClockCorrectionPassive) - WCP
WCF	7:4	rw	**Maximum Without Clock Correction Fatal (gMaxWithoutClockCorrecti on Fatal)** Defines the number of consecutive even / odd cycle pairs with missing clock correction terms that will cause a transition from “NORMAL_ACTIVE” or “NORMAL_PASSIVE” to “HALT” state. Must be identical in all nodes of a cluster. Valid values are 1 to 15 (1_H to F_H) cycle pairs.
0	31:8	r	**Reserved** Read as 0; should be written with 0.

23.4.26 NEM Configuration Register

The Communication Controller accepts modifications of the register in “DEFAULT_CONFIG” or “CONFIG” state only.

NEMC Offset address: $008C_H$

NEM Configuration Register Kernel Reset value: $0000\ 0000_H$



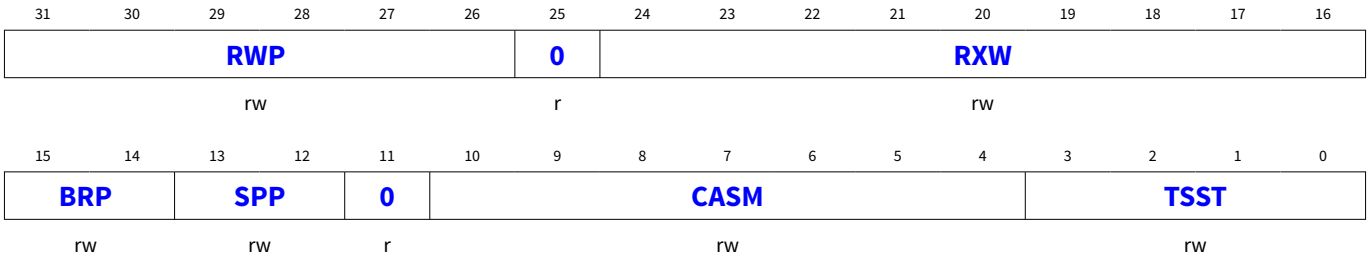
Field	Bits	Type	Description
NML	3:0	rw	**Network Management Vector Length (gNetworkManagementVectorLength)** These bits configure the length of the NM Vector. The configured length must be identical in all nodes of a cluster. Valid values are 0 to 12 (0_H to C_H) bytes.
0	31:4	r	**Reserved** Read as 0; should be written with 0.

23.4.27 PRT Configuration Register 1

The Communication Controller accepts modifications of the register in “DEFAULT_CONFIG” or “CONFIG” state only.

PRTC1 Offset address: 0090_H

PRT Configuration Register 1 Kernel Reset value: $084C\ 0633_H$



Field	Bits	Type	Description
TSST	3:0	rw	**Transmission Start Sequence Transmitter (gdTSSTransmitter)** Configures the duration of the Transmission Start Sequence (TSS) in terms of Bit Times (1 bit time = 4 Microticks = 100 ns at 10 Mbps). Must be identical in all nodes of a cluster. Valid values are 3 to 15 (3_H to F_H) Bit Times.

(table continues...)

(continued)

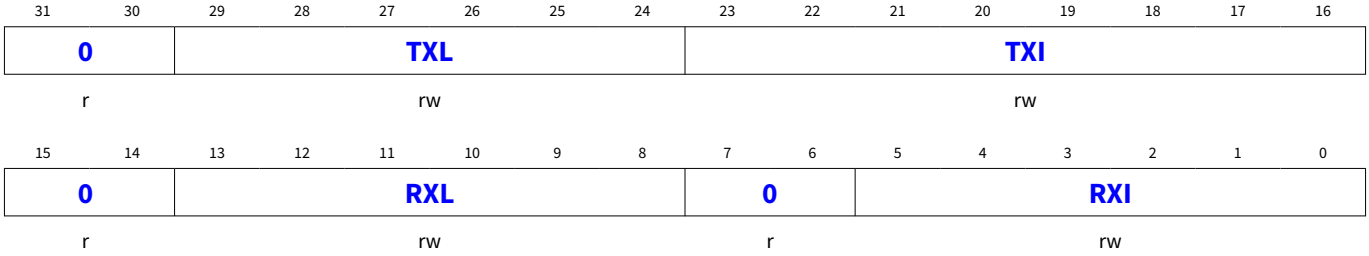
Field	Bits	Type	Description
CASM	10:4	rw	**Collision Avoidance Symbol Maximum (gdCASRxLowMax)** Configures the upper limit of the acceptance window for a collision avoidance symbol (CAS). Valid values are 67 to 99 (43_H to 63_H). Most significant bit of CASM is hard wired to 1 and can not be modified.
SPP	13:12	rw	**Strobe Point Position** Defines the sample count value for strobing. The strobed bit value is set to the voted value when the sample count is incremented to the value configured by SPP. ***Note**: The current revision 2.1 of the FlexRay™ protocol requires that SPP = 00_B. The alternate strobe point positions could be used to compensate for asymmetries in the physical layer.* 00_B Sample 5 (default) 01_B Sample 4 10_B Sample 6 11_B Reserved; should not be used.
BRP	15:14	rw	**Baud Rate Prescaler (gdSampleClockPeriod, pSamplePerMicrotick)** The baud rate prescaler configures the baud rate on the FlexRay™ bus. The baud rates listed below are valid with a sample clock f_{SCLK} = 80 MHz. One bit time always consists of 8 samples independent of the configured baud rate. 00_B 10 Mbit/s (1 Microtick= 25 ns; twice sampled with f_{SCLK}) gdSampleClockPeriod = 12.5 ns = 1 / f_{SCLK} pSamplesPerMicrotick = 2 01_B 5 Mbit/s (1 Microtick= 25 ns; single sampled with f_{SCLK} / 2) gdSampleClockPeriod = 25 ns = 2 / f_{SCLK} pSamplesPerMicrotick = 1 10_B 2.5 Mbit/s (1 Microtick = 50 ns; single sampled with f_{SCLK} / 4) gdSampleClockPeriod = 50 ns = 4 / f_{SCLK} pSamplesPerMicrotick = 1 11_B Reserved; should not be used (2.5 Mbit/s (1 Microtick = 50 ns; single sampled with f_{SCLK} / 4) gdSampleClockPeriod = 50 ns = 4 / f_{SCLK} pSamplesPerMicrotick = 1
RXW	24:16	rw	**Wakeup Symbol Receive Window Length (gdWakeupSymbolRxWindow)** Configures the number of Bit Times used by the node to test the duration of the received wakeup pattern. Must be identical in all nodes of a cluster. Valid values are 76 to 301 ($4C_H$ to $12D_H$) Bit Times.
RWP	31:26	rw	**Repetitions of Tx Wakeup Pattern (pWakeupPattern)** Configures the number of repetitions (sequences) of the Tx wakeup symbol. Valid values are 2 to 63 (2_H to $3F_H$).
0	11, 25	r	**Reserved** Read as 0; should be written with 0.

23.4.28 PRT Configuration Register 2

The Communication Controller accepts modifications of the register in “DEFAULT_CONFIG” or “CONFIG” state only.

PRTC2 Offset address: 0094_H

PRT Configuration Register 2 Kernel Reset value: $0F2D\ 0A0E_H$



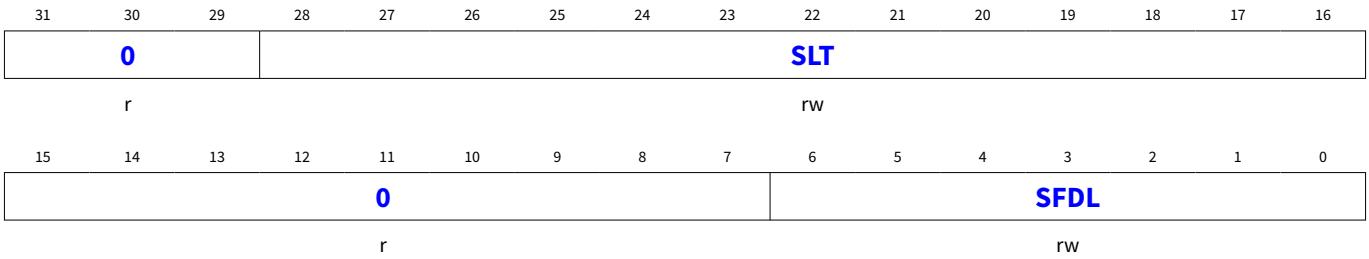
Field	Bits	Type	Description
RXI	5:0	rw	**Wakeup Symbol Receive Idle (gdWakeupSymbolRxIdle)** Configures the number of Bit Times used by the node to test the duration of the idle phase of the received wakeup symbol. Must be identical in all nodes of a cluster. Valid values are 14 to 59 (E_H to $3B_H$) Bit Times.
RXL	13:8	rw	**Wakeup Symbol Receive Low (gdWakeupSymbolRxLow)** Configures the number of Bit Times used by the node to test the duration of the low phase of the received wakeup symbol. Must be identical in all nodes of a cluster. Valid values are 10 to 55 (A_H to 37_H) Bit Times.
TXI	23:16	rw	**Wakeup Symbol Transmit Idle (gdWakeupSymbolTxIdle)** Configures the number of Bit Times used by the node to transmit the idle phase of the wakeup symbol. Must be identical in all nodes of a cluster. Valid values are 45 to 180 ($2D_H$ to $B4_H$) Bit Times.
TXL	29:24	rw	**Wakeup Symbol Transmit Low (gdWakeupSymbolTxLow)** Configures the number of Bit Times used by the node to transmit the low phase of the wakeup symbol. Must be identical in all nodes of a cluster. Valid values are 15 to 60 (F_H to $3C_H$) Bit Times.
0	7:6, 15:14, 31:30	r	**Reserved** Read as 0; should be written with 0.

23.4.29 MHD Configuration Register

The Communication Controller accepts modifications of the register in “DEFAULT_CONFIG” or “CONFIG” state only.

MHDC Offset address: 0098_H

MHD Configuration Register Kernel Reset value: $0000\ 0000_H$



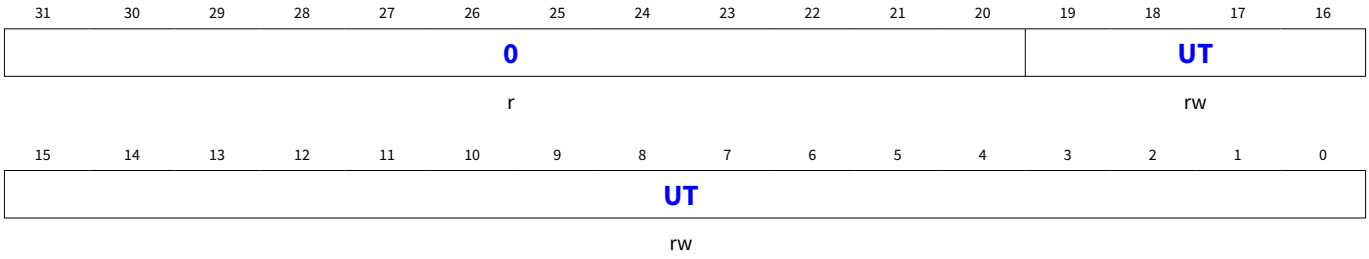
Field	Bits	Type	Description
SFDL	6:0	rw	**Static Frame Data Length (gPayloadLengthStatic)** Configures the cluster-wide payload length for all Frames sent in the static segment in double byte. The payload length must be identical in all nodes of a cluster. Valid values are 0 to 127 (0 to $7F_H$).
SLT	28:16	rw	**Start of Latest Transmit (pLatestTx)** Configures the maximum minislot value allowed before inhibiting Frame transmission in the dynamic segment of the cycle. There is no transmission dynamic segment if SLT is reset to zero. Valid values are 0 to 7981 (0 to $1F2D_H$) minislots.
0	15:7, 31:29	r	**Reserved** Read as 0; should be written with 0.

23.4.30 GTU Configuration Register 1

The Communication Controller accepts modifications of the register in “DEFAULT_CONFIG” or “CONFIG” state only.

GTUC01 Offset address: $00A0_H$

GTU Configuration Register 1 Kernel Reset value: $0000\ 0280_H$



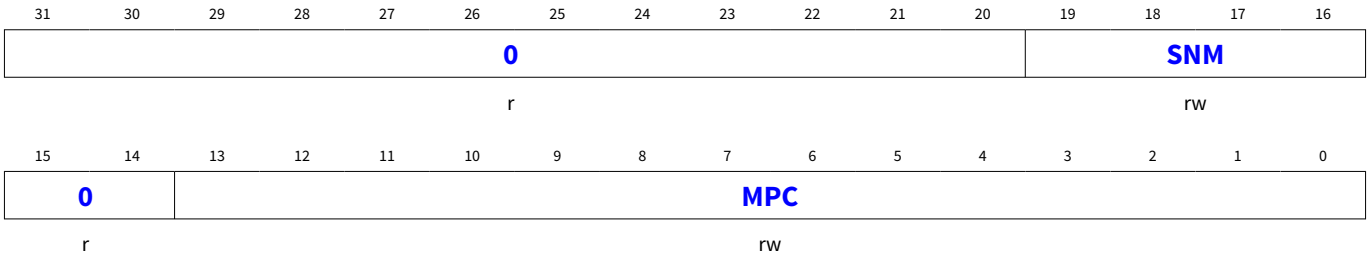
Field	Bits	Type	Description
UT	19:0	rw	**Microtick per Cycle (pMicroPerCycle)** Configures the duration of the communication cycle in Microticks. Valid values are 640 to 640000 (280_H to $9C400_H$) Microticks.
0	31:20	r	**Reserved** Read as 0; should be written with 0.

23.4.31 GTU Configuration Register 2

The Communication Controller accepts modifications of the register in “DEFAULT_CONFIG” or “CONFIG” state only.

GTUC02 Offset address: $00A4_H$

GTU Configuration Register 2 Kernel Reset value: $0002\ 000A_H$



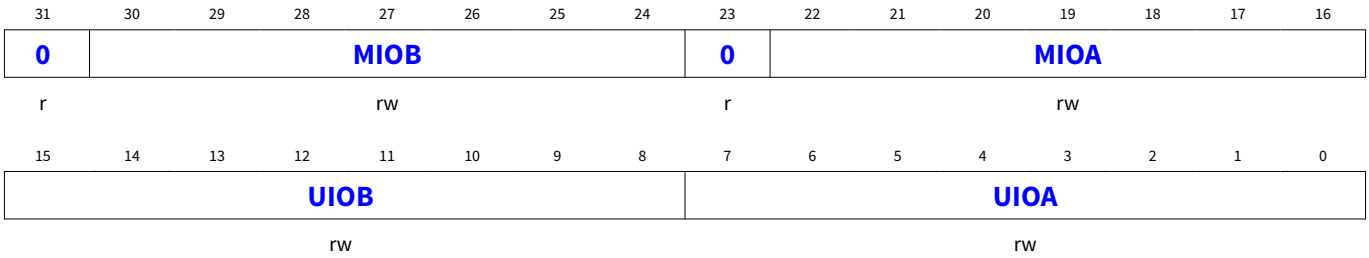
Field	Bits	Type	Description
MPC	13:0	rw	**Macrotick Per Cycle (gMacroPerCycle)** Configures the duration of one communication cycle in Macroticks. The cycle length must be identical in all nodes of a cluster. Valid values are 10 to 16000 (A_H to $3E80_H$) Macroticks.
SNM	19:16	rw	**Sync Node Max (gSyncNodeMax)** Maximum number of Frames within a cluster with SYNC Frame indicator bit SYN set to 1. Must be identical in all nodes of a cluster. Valid values are 2 to 15 (2_H to F_H).
0	15:14, 31:20	r	**Reserved** Read as 0; should be written with 0.

23.4.32 GTU Configuration Register 3

The Communication Controller accepts modifications of the register in “DEFAULT_CONFIG” or “CONFIG” state only.

GTUC03 Offset address: $00A8_H$

GTU Configuration Register 3 Kernel Reset value: $0202\ 0000_H$



Field	Bits	Type	Description
UIOA	7:0	rw	**Microtick Initial Offset Channel A (pMicroInitialOffset[A])** Configures the number of Microticks between the actual time reference point on channel A and the subsequent Macrotick boundary of the secondary time reference point. The parameter depends on pDelayCompensation[A] and therefore has to be set for each channel independently. Valid values are 0 to 240 (0_H to $F0_H$) Microticks.

(table continues...)

(continued)

Field	Bits	Type	Description
UIOB	15:8	rw	**Microtick Initial Offset Channel B (pMicroInitialOffset[B])** Configures the number of Microticks between the actual time reference point on channel B and the subsequent Macrotick boundary of the secondary time reference point. The parameter depends on pDelayCompensation[B] and therefore has to be set for each channel independently. Valid values are 0 to 240 (0_H to $F0_H$) Microticks.
MIOA	22:16	rw	**Macrotick Initial Offset Channel A (gMacroInitialOffset[A])** Configures the number of Macroticks between the static slot boundary and the subsequent Macrotick boundary of the secondary time reference point based on the nominal Macrotick duration. Must be identical in all nodes of a cluster. Valid values are 2 to 72 (2_H to 48_H) Macroticks.
MIOB	30:24	rw	**Macrotick Initial Offset Channel B (gMacroInitialOffset[B])** Configures the number of Macroticks between the static slot boundary and the subsequent Macrotick boundary of the secondary time reference point based on the nominal Macrotick duration. Must be identical in all nodes of a cluster. Valid values are 2 to 72 (2_H to 48_H) Macroticks.
0	23, 31	r	**Reserved** Read as 0; should be written with 0.

23.4.33 GTU Configuration Register 4

The Communication Controller accepts modifications of the register in “DEFAULT_CONFIG” or “CONFIG” state only. For details about configuration of NIT and OCS see “Configuration of Network Idle Time (NIT) Start and Offset Correction Start”.

GTUC04 Offset address: $00AC_H$

GTU Configuration Register 4 Kernel Reset value: $0008\ 0007_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		OCS													
r		rw													

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		NIT													
r		rw													

Field	Bits	Type	Description
NIT	13:0	rw	**Network Idle Time Start (gMacroPerCycle - gdNIT - 1)** Configures the starting point of the Network Idle Time (NIT) at the end of the communication cycle expressed in terms of Macroticks from the beginning of the cycle. The start of network idle time (NIT) is recognized if Macrotick = gMacroPerCycle - gdNIT -1 and the increment pulse of Macrotick is set. Must be identical in all nodes of a cluster. Valid values are 7 to 15997 (7_H to $3E7D_H$) Macroticks.

(table continues...)

(continued)

Field	Bits	Type	Description
OCS	29:16	rw	**Offset Correction Start (gOffsetCorrectionStart - 1)** Determines the start of the offset correction within the network idle time (NIT) phase, calculated from start of cycle. Must be identical in all nodes of a cluster. For cluster consisting of ERAY implementations only, it is sufficient to program OCS = NIT + 1. Valid values are 8 to 15998 (8_H to $3E7E_H$) Macroticks.
0	15:14, 31:30	r	**Reserved** Read as 0; should be written with 0.

23.4.34 GTU Configuration Register 5

The Communication Controller accepts modifications of the register in “DEFAULT_CONFIG” or “CONFIG” state only.

GTUC05 Offset address: $00B0_H$

GTU Configuration Register 5 Kernel Reset value: $0E00\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DEC								0			CDD				
rw								r			rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DCB								DCA							
rw								rw							

Field	Bits	Type	Description
DCA	7:0	rw	**Delay Compensation Channel A (pDelayCompensation[A])** Used to compensate for reception delays on channel A. This covers assumed propagation delay up to cPropagationDelayMax for Microticks in the range of 0.0125 µs to 0.05 µs. In practice, the minimum of the propagation delays of all sync nodes should be applied. Valid values are 0 to 200 (0_H to $C8_H$) Microticks.
DCB	15:8	rw	**Delay Compensation Channel B (pDelayCompensation[B])** Used to compensate for reception delays on channel B. This covers assumed propagation delay up to cPropagationDelayMax for Microticks in the range of 0.0125 to 0.05 µs. In practice, the minimum of the propagation delays of all sync nodes should be applied. Valid values are 0 to 200 (0_H to $C8_H$) Microticks.
CDD	20:16	rw	**Cluster Drift Damping (pClusterDriftDamping)** Configures the cluster drift damping value used in clock synchronization to minimize accumulation of rounding errors. Valid values are 0 to 20 (0_H to 14_H) Microticks.
DEC	31:24	rw	**Decoding Correction (pDecodingCorrection)** Configures the decoding correction value used to determine the primary time reference point. Valid values are 14 to 143 (E_H to $8F_H$) Microticks.

(table continues...)

(continued)

Field	Bits	Type	Description
0	23:21	r	**Reserved** Read as 0; should be written with 0.

23.4.35 GTU Configuration Register 6

The Communication Controller accepts modifications of the register in “DEFAULT_CONFIG” or “CONFIG” state only.

GTUC06 Offset address: $00B4_H$

GTU Configuration Register 6 Kernel Reset value: $0002\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0					MOD										
r					rw										

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0					ASR										
r					rw										

Field	Bits	Type	Description
ASR	10:0	rw	**Accepted Startup Range (pdAcceptedStartupRange)** Number of Microticks constituting the expanded range of measured deviation for startup Frames during integration. Valid values are 0 to 1875 (0_H to 753_H) Microticks.
MOD	26:16	rw	**Maximum Oscillator Drift (pdMaxDrift)** Maximum drift offset between two nodes that operate with unsynchronized clocks over one communication cycle in Microticks. Valid values are 2 to 1923 (2_H to 783_H) Microticks.
0	15:11, 31:27	r	**Reserved** Read as 0; should be written with 0.

23.4.36 GTU Configuration Register 7

The Communication Controller accepts modifications of the register in “DEFAULT_CONFIG” or “CONFIG” state only.

GTUC07 Offset address: $00B8_H$

GTU Configuration Register 7 Kernel Reset value: $0002\ 0004_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0						NSS									
r						rw									

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0						SSL									
r						rw									

Field	Bits	Type	Description
SSL	9:0	rw	**Static Slot Length (gdStaticSlot)** Configures the duration of a static slot in Macroticks. The static slot length must be identical in all nodes of a cluster. Valid values are 4 to 659 (4_H to 293_H) Macroticks.
NSS	25:16	rw	**Number of Static Slots (gNumberOfStaticSlots)** Configures the number of static slots in a cycle. At least 2 coldstart nodes must be configured to startup a FlexRay™ network. The number of static slots must be identical in all nodes of a cluster. Valid values are 2 to 1023 (2_H to $3FF_H$).
0	15:10, 31:26	r	**Reserved** Read as 0; should be written with 0.

23.4.37 GTU Configuration Register 8

The Communication Controller accepts modifications of the register in “DEFAULT_CONFIG” or “CONFIG” state only.

GTUC08 Offset address: $00BC_H$

GTU Configuration Register 8 Kernel Reset value: $0000\ 0002_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0			NMS												
r			rw												

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0										MSL					
r										rw					

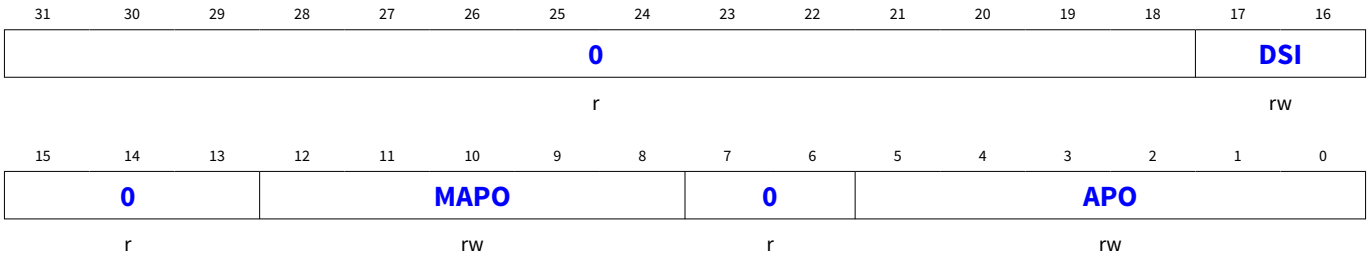
Field	Bits	Type	Description
MSL	5:0	rw	**Minislot Length (gdMinislot)** Configures the duration of a minislot in Macroticks. The minislot length must be identical in all nodes of a cluster. Valid values are 2 to 63 (2_H to $3F_H$) Macroticks.
NMS	28:16	rw	**Number of Minislots (gNumberOfMinislots)** Configures the number of minislots within the dynamic segment of a cycle. The number of minislots must be identical in all nodes of a cluster. Valid values are 0 to 7986 (0_H to $1F32_H$).
0	15:6, 31:29	r	**Reserved** Read as 0; should be written with 0.

23.4.38 GTU Configuration Register 9

The Communication Controller accepts modifications of the register in “DEFAULT_CONFIG” or “CONFIG” state only.

GTUC09 Offset address: $00C0_H$

GTU Configuration Register 9 Kernel Reset value: $0000\ 0101_H$



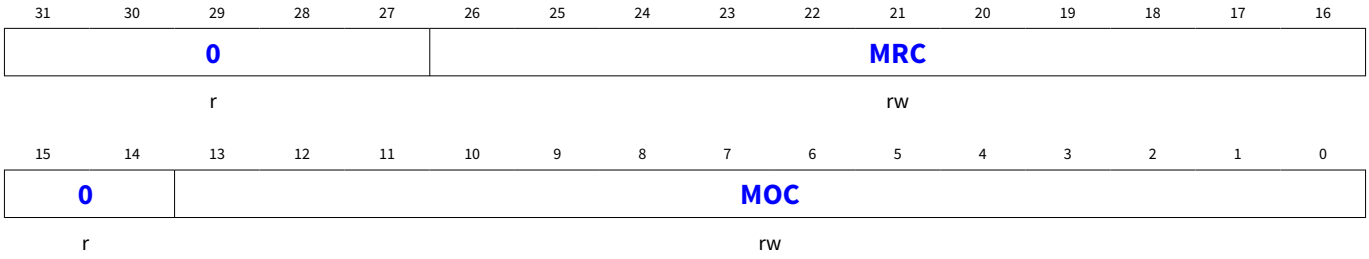
Field	Bits	Type	Description
APO	5:0	rw	**Action Point Offset (gdActionPointOffset)** Configures the action point offset in Macroticks within static slots and symbol window. Must be identical in all nodes of a cluster. Valid values are 1 to 63 (1_H to $3F_H$) Macroticks.
MAPO	12:8	rw	**Minislot Action Point Offset (gdMinislot Action Point Offset)** Configures the action point offset in Macroticks within the minislots of the dynamic segment. Must be identical in all nodes of a cluster. Valid values are 1 to 31 (1_H to $1F_H$) Macroticks.
DSI	17:16	rw	**Dynamic Slot Idle Phase (gdDynamicSlotIdlePhase)** The duration of the dynamic slot idle phase has to be greater or equal than the idle detection time. Must be identical in all nodes of a cluster. Valid values are 0 to 2 Minislot.
0	7:6, 15:13, 31:18	r	**Reserved** Read as 0; should be written with 0.

23.4.39 GTU Configuration Register 10

The Communication Controller accepts modifications of the register in “DEFAULT_CONFIG” or “CONFIG” state only.

GTUC10 Offset address: $00C4_H$

GTU Configuration Register 10 Kernel Reset value: $0002\ 0005_H$



Field	Bits	Type	Description
MOC	13:0	rw	**Maximum Offset Correction (pOffsetCorrectionOut)** Holds the maximum permitted offset correction value to be applied by the internal clock synchronization algorithm (absolute value). The Communication Controller checks only the internal offset correction value against the maximum offset correction value. Valid values are 5 to 15266 (5_H to $3BA2_H$) Microticks.
MRC	26:16	rw	**Maximum Rate Correction (pRateCorrectionOut)** Holds the maximum permitted rate correction value to be applied by the internal clock synchronization algorithm. The Communication Controller checks only the internal rate correction value against the maximum rate correction value (absolute value). Valid values are 2 to 1923 (2_H to 783_H) Microticks.
0	15:14, 31:27	r	**Reserved** Read as 0; should be written with 0.

23.4.40 GTU Configuration Register 11

GTUC11 Offset address: $00C8_H$

GTU Configuration Register 11 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0					ERC			0					EOC		
r					rw			r					rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0						ERCC		0						EOCC	
r						rw		r						rw	

Field	Bits	Type	Description
EOCC	1:0	rw	**External Offset Correction Control (pExternOffsetControl)** By writing to EOCC the external offset correction is enabled as specified below. Should be modified only outside network idle time (NIT). 00_B No external offset correction 01_B No external offset correction 10_B External offset correction value subtracted from calculated offset correction value 11_B External offset correction value added to calculated offset correction value

(table continues...)

(continued)

Field	Bits	Type	Description
ERCC	9:8	rw	**External Rate Correction Control (pExternRateControl)** By writing to ERCC the external rate correction is enabled as specified below. Should be modified only outside network idle time (NIT). 00_B No external rate correction 01_B No external rate correction 10_B External rate correction value subtracted from calculated rate correction value 11_B External rate correction value added to calculated rate correction value
EOC	18:16	rw	**External Offset Correction (pExternOffsetCorrection)** Holds the external clock offset correction value in Microticks to be applied by the internal synchronization algorithm. The value is subtracted / added from / to the calculated offset correction value. The value is applied during network idle time (NIT). May be modified in “DEFAULT_CONFIG” or “CONFIG” state only. Valid values are 0 to 7 Microticks.
ERC	26:24	rw	**External Rate Correction (pExternRateCorrection)** Holds the external rate correction value in Microticks to be applied by the internal clock synchronization algorithm. The value is subtracted / added from / to the calculated rate correction value. The value is applied during network idle time (NIT). Can be modified in “DEFAULT_CONFIG” or “CONFIG” state only. Valid values are 0 to 7 Microticks.
0	7:2, 15:10, 23:19, 31:27	r	**Reserved** Read as 0; should be written with 0.

23.4.41 Communication Controller Status Vector

CCSV Offset address: 0100_H

Communication Controller Status Vector Kernel Reset value: $0010\ 4000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		PSL						RCA					WSV		
r		rh						rh					rh		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	CSI	CSAI	CSNI	0		SLM		HRQ	FSI	POCS					
r	rh	rh	rh	r		rh		rh	rh	rh					

Field	Bits	Type	Description
POCS	5:0	rh	**Protocol Operation Control Status** 00_H ... 05_H, $0F_H$ Indicates the actual state of operation of the Communication Controller Protocol Operation Control 06_H … 14_H are reserved. 10_H ... 13_H Indicates the actual state of operation of the POC in the wakeup path 14_H … $1F_H$ are reserved. 20_H ... $2B_H$ Indicates the actual state of operation of the POC in the startup path $2C_H$ … $3F_H$ are reserved. 00_H “DEFAULT_CONFIG” state 01_H “READY” state 02_H “NORMAL_ACTIVE” state 03_H “NORMAL_PASSIVE” state 04_H “HALT” state 05_H “MONITOR_MODE” state $0F_H$ “CONFIG” state 10_H WAKEUP_STANDBY state 11_H “WAKEUP_LISTEN” state 12_H “WAKEUP_SEND” state 13_H “WAKEUP_DETECT” state 20_H “STARTUP_PREPARE” state 21_H “COLDSTART_LISTEN” state 22_H “COLDSTART_COLLISION_RESOLUTION state 23_H “COLDSTART_CONSISTENCY_CHECK” state 24_H “COLDSTART_GAP state 25_H “COLDSTART_JOIN” State 26_H “INTEGRATION_COLDSTART_CHECK” state 27_H “INTEGRATION_LISTEN” state 28_H “INTEGRATION_CONSISTENCY_CHECK” state 29_H “INITIALIZE_SCHEDULE” state $2A_H$ “ABORT_STARTUP” state $2B_H$ “STARTUP_SUCCESS” state
FSI	6	rh	**Freeze Status Indicator (vPOC!Freeze)** Indicates that the POC has entered the “HALT” state due to CHI command “FREEZE” or due to an error condition requiring an immediate POC halt. Reset by transition from “HALT” to “DEFAULT_CONFIG” state.
HRQ	7	rh	**Halt Request (vPOC!CHIHaltRequest)** Indicates that a request from the Host has been received to halt the POC at the end of the communication cycle. Reset by transition from “HALT” to “DEFAULT_CONFIG” state or when entering “READY” state.

(table continues...)

(continued)

Field	Bits	Type	Description
SLM	9:8	rh	**Slot Mode (vPOC!SlotMode)** Indicates the actual slot mode of the POC in states READY, WAKEUP, STARTUP, NORMAL_ACTIVE, and NORMAL_PASSIVE. Default is “SINGLE”. Changes to “ALL”, depending on configuration bit SUCC1.TSM. In “NORMAL_ACTIVE” or “NORMAL_PASSIVE” state the CHI command “ALL_SLOTS” will change the slot mode from “SINGLE” over “ALL_PENDING” to “ALL”. Set to SINGLE in all other states. 00_B SINGLE 01_B Reserved 10_B ALL_PENDING 11_B ALL
CSNI	12	rh	**Coldstart Noise Indicator (vPOC!ColdstartNoise)** Indicates that the cold start procedure occurred under noisy conditions. Reset by CHI command “RESET_STATUS_INDICATORS” or by transition from “HALT” to “DEFAULT_CONFIG” state or from “READY” to “STARTUP” state.
CSAI	13	rh	**Coldstart Abort Indicator** Coldstart aborted. Reset by CHI command “RESET_STATUS_INDICATORS” or by transition from “HALT” to “DEFAULT_CONFIG” state or from “READY” to “STARTUP” state.
CSI	14	rh	**Cold Start Inhibit (vColdStartInhibit)** Indicates that the node is disabled from cold starting. The flag is set whenever the POC enters “READY” state due to CHI command “READY”. The flag has to be reset under control of the Host by CHI command “ALLOW_COLDSTART” (SUCC1.CMD = 1001_B). 0_B Cold starting of node enabled 1_B Cold starting of node disabled

(table continues...)

(continued)

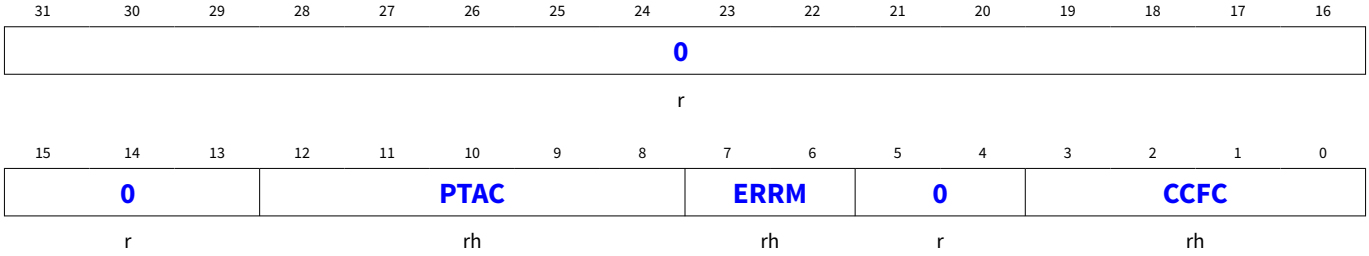
Field	Bits	Type	Description
WSV	18:16	rh	**Wakeup Status (vPOC!WakeupStatus)** Indicates the status of the current wakeup attempt. Reset by CHI command “RESET_STATUS_INDICATORS” or by transition from “HALT” to “DEFAULT_CONFIG” state. 000_B UNDEFINED. Wakeup not yet executed by the Communication Controller. 001_B RECEIVED_HEADER. Set when the Communication Controller finishes wakeup due to the reception of a Frame Header without coding violation on either channel in “WAKEUP_LISTEN” state. 010_B RECEIVED_WUP. Set when the Communication Controller finishes wakeup due to the reception of a valid wakeup pattern on the configured wakeup channel in “WAKEUP_LISTEN” state. 011_B COLLISION_HEADER. Set when the Communication Controller stops wakeup due to a detected collision during wakeup pattern transmission by receiving a valid Header on either channel. 100_B COLLISION_WUP. Set when the Communication Controller stops wakeup due to a detected collision during wakeup pattern transmission by receiving a valid wakeup pattern on the configured wakeup channel. 101_B COLLISION_UNKNOWN. Set when the Communication Controller stops wakeup by leaving “WAKEUP_DETECT” state after expiration of the wakeup timer without receiving a valid wakeup pattern or a valid Frame Header. 110_B TRANSMITTED. Set when the Communication Controller has successfully completed the transmission of the wakeup pattern. 111_B Reserved
RCA	23:19	rh	**Remaining Coldstart Attempts (vRemainingColdstartAttempts)** Indicates the number of remaining coldstart attempts. The RUN command resets this counter to the maximum number of coldstart attempts as configured by SUCC1.CSA.
PSL	29:24	rh	**POC Status Log** Status of CCSV.POCS immediately before entering “HALT” state. Set when entering “HALT” state. Set to “HALT” when FREEZE command is applied during “HALT” state. Reset to 000000_B when leaving “HALT” state.
0	11:10, 15, 31:30	r	**Reserved** Read as 0; should be written with 0.

23.4.42 Communication Controller Error Vector

Reset by transition from “HALT” to “DEFAULT_CONFIG” state or when entering “READY” state.

CCEV — Offset address: 0104_H

Communication Controller Error Vector — Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
CCFC	3:0	rh	**Clock Correction Failed Counter (vClockCorrectionFailed)** The Clock Correction Failed Counter is incremented by one at the end of any odd communication cycle where either the missing offset correction error or missing rate correction error are active. The Clock Correction Failed Counter is reset to 0 at the end of an odd communication cycle if neither the offset correction failed nor the rate correction failed errors are active. The Clock Correction Failed Counter stops at 15.
ERRM	7:6	rh	**Error Mode (vPOC!ErrorMode)** Indicates the actual error mode of the POC. 00_B “ACTIVE” 01_B “PASSIVE” 10_B “COMM_HALT” 11_B Reserved
PTAC	12:8	rh	**Passive to Active Count (vAllowPassiveToActive)** Indicates the number of consecutive even / odd cycle pairs that have passed with valid rate and offset correction terms, while the node is waiting to transit from “NORMAL_PASSIVE” state to “NORMAL_ACTIVE” state. The transition takes place when PTAC equals SUCC1.PTA - 1.
0	5:4, 31:13	r	**Reserved** Read as 0; should be written with 0.

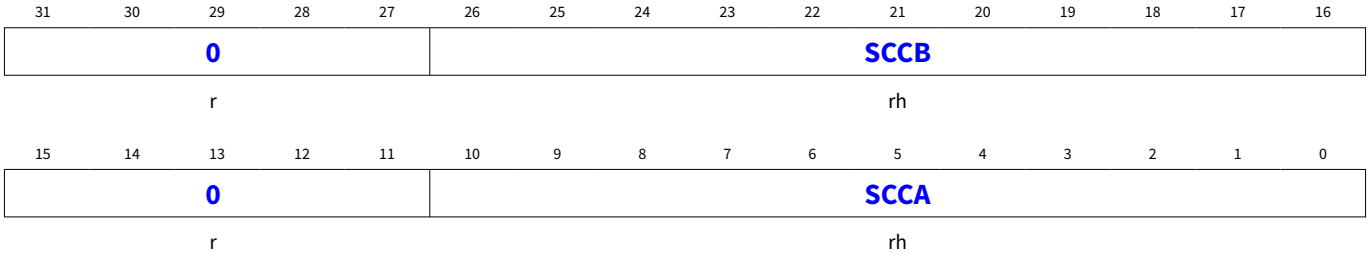
23.4.43 Slot Counter Value

SCV

Slot Counter Value

Offset address: 0110_H

Kernel Reset value: $0000\ 0000_H$

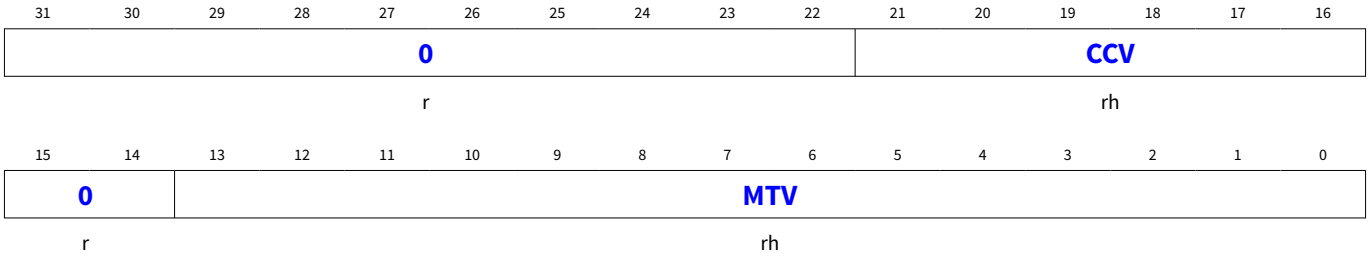


Field	Bits	Type	Description
SCCA	10:0	rh	**Slot Counter Channel A (vSlotCounter[A])** Current slot counter value on channel A. The value is incremented by the Communication Controller and reset at the start of a communication cycle. Valid values are 0 to 2047 (0_H to $7FF_H$).
SCCB	26:16	rh	**Slot Counter Channel B (vSlotCounter[B])** Current slot counter value on channel B. The value is incremented by the Communication Controller and reset at the start of a communication cycle. Valid values are 0 to 2047 (0_H to $7FF_H$).
0	15:11, 31:27	r	**Reserved** Read as 0; should be written with 0.

23.4.44 Macrotick and Cycle Counter Value

MTCCV Offset address: 0114_H

Macrotick and Cycle Counter Value Kernel Reset value: $0000\ 0000_H$

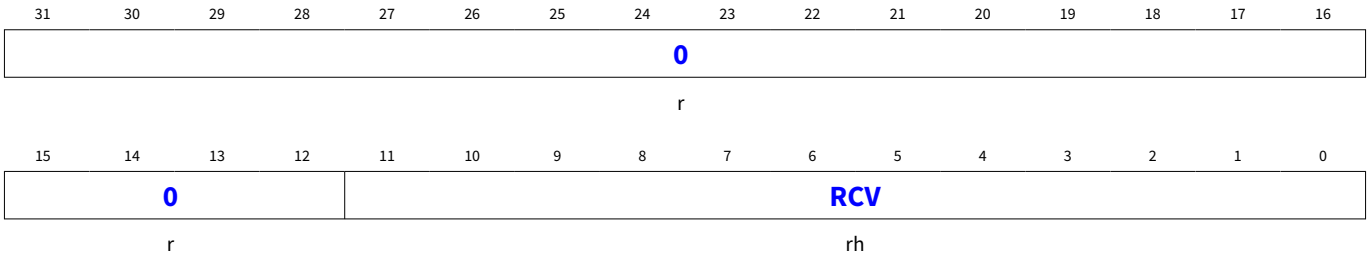


Field	Bits	Type	Description
MTV	13:0	rh	**Macrotick Value (vMacrotick)** Current Macrotick value. The value is incremented by the Communication Controller and reset at the start of a communication cycle. Valid values are 0 to 16000 (0_H to $3E80_H$).
CCV	21:16	rh	**Cycle Counter Value (vCycleCounter)** Current cycle counter value. The value is incremented by the Communication Controller at the start of a communication cycle. Valid values are 0 to 63 (0_H to $3F_H$).
0	15:14, 31:22	r	**Reserved** Read as 0; should be written with 0.

23.4.45 Rate Correction Value

RCV Offset address: 0118_H

Rate Correction Value Kernel Reset value: $0000\ 0000_H$



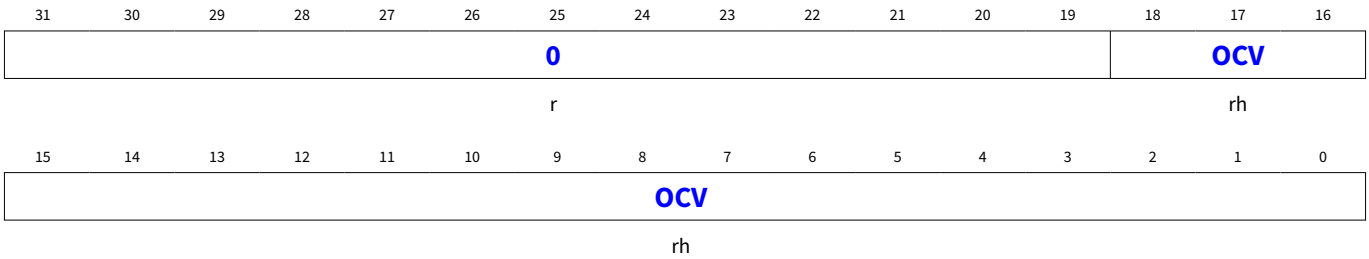
Field	Bits	Type	Description
RCV	11:0	rh	**Rate Correction Value (vRateCorrection)** Rate correction value (two's complement). Calculated internal rate correction value before limitation. If the RCV value exceeds the limits defined by GTUC10.MRC, flag SFS.RCLR is set to 1.
0	31:12	r	**Reserved** Read as 0; should be written with 0.

23.4.46 Offset Correction Value

***Note**: The external rate / offset correction value is added to the limited rate / offset correction value.*

OCV Offset address: $011C_H$

Offset Correction Value Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
OCV	18:0	rh	**Offset Correction Value (vOffsetCorrection)** Offset correction value (two's complement). Calculated internal offset correction value before limitation. If the OCV value exceeds the limits defined by GTUC10.MOC flag SFS.OCLR is set to 1.
0	31:19	r	**Reserved** Read as 0; should be written with 0.

23.4.47 SYNC Frame Status

The maximum number of valid SYNC Frames in a communication cycle is 15.

SFS Offset address: 0120_H

SYNC Frame Status Kernel Reset value: $0000\ 0000_H$

Field	Bits	Type	Description
VSAE	3:0	rh	**Valid SYNC Frames Channel A, even communication cycle** Holds the number of valid SYNC Frames received on channel A in the even communication cycle. If transmission of SYNC Frames is enabled by SUCC1.TXSY, the value is incremented by one. The value is updated during the network idle time (NIT) of each even communication cycle. This bit field is only valid if the channel A is assigned to the Communication Controller by SUCC1.CCHA.
VSAO	7:4	rh	**Valid SYNC Frames Channel A, odd communication cycle** Holds the number of valid SYNC Frames received on channel A in the odd communication cycle. If transmission of SYNC Frames is enabled by SUCC1.TXSY, the value is incremented by one. The value is updated during the network idle time (NIT) of each odd communication cycle. This bit field is only valid if the channel A is assigned to the Communication Controller by SUCC1.CCHA.
VSBE	11:8	rh	**Valid SYNC Frames Channel B, even communication cycle** Holds the number of valid SYNC Frames received on channel B in the even communication cycle. If transmission of SYNC Frames is enabled by SUCC1.TXSY, the value is incremented by one. The value is updated during the network idle time (NIT) of each even communication cycle. This bit field is only valid if the channel B is assigned to the Communication Controller by SUCC1.CCHB.
VSBO	15:12	rh	**Valid SYNC Frames Channel B, odd communication cycle** Holds the number of valid SYNC Frames received on channel B in the odd communication cycle. If transmission of SYNC Frames is enabled by SUCC1.TXSY, the value is incremented by one. The value is updated during the network idle time (NIT) of each odd communication cycle. This bit field is only valid if the channel B is assigned to the Communication Controller by SUCC1.CCHB.
MOCS	16	rh	**Missing Offset Correction Signal** The Missing Offset Correction flag signals to the Host, that no offset correction calculation can be performed because no SYNC Frames were received. The flag is updated by the Communication Controller at start of offset correction phase. 0_B Offset correction signal valid 1_B Missing offset correction signal

(table continues...)

(continued)

Field	Bits	Type	Description
OCLR	17	rh	**Offset Correction Limit Reached** The Offset Correction Limit Reached flag signals to the Host, that the offset correction value has exceeded its limit as defined by GTUC10.MOC. The flag is updated by the Communication Controller at start of offset correction phase. 0_B Offset correction below limit 1_B Offset correction limit reached
MRCS	18	rh	**Missing Rate Correction Signal** The Missing Rate Correction Flag signals to the Host, that no rate correction calculation can be performed because no pairs of even / odd SYNC Frames were received. The flag is updated by the Communication Controller at start of offset correction phase. 0_B Rate correction signal valid 1_B Missing rate correction signal
RCLR	19	rh	**Rate Correction Limit Reached** The Rate Correction Limit Reached flag signals to the Host, that the rate correction value has exceeded its limit as defined by GTUC10.MRC. The flag is updated by the Communication Controller at start of offset correction phase. 0_B Rate correction below limit 1_B Rate correction limit reached
0	31:20	r	**Reserved** Read as 0; should be written with 0.

23.4.48 Symbol Window and Network Idle Time Status

Symbol window related status information. Updated by the Communication Controller at the end of the symbol window for each channel. During startup the status data is not updated.

***Note**: MTSA and MTSB may be changed outside "DEFAULT_CONFIG" or "CONFIG" state when the write to SUC Configuration Register 1 (SUCC1) register is directly preceded by the unlock sequence as described in "Lock Register". This may be combined with CHI command SEND_MTS. If both bits MTSA and MTSB are set to 1, an MTS symbol will be transmitted on both channels when requested by writing SUCC1.CMD = 1000_B*

SWNIT Offset address: 0124_H

Symbol Window and Network Idle Time Status Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				SBNB	SENB	SBNA	SENA	MTSB	MTSA	TCSB	SBSB	SESB	TCSA	SBSA	SESA
r				rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
SESA	0	rh	**Syntax Error in Symbol Window Channel A (vSS!SyntaxErrorA)** 0_B No syntax error detected 1_B Syntax error during symbol window detected on channel A
SBSA	1	rh	**Slot Boundary Violation in Symbol Window Channel A (vSS! BViolationA)** 0_B No slot boundary violation detected 1_B Slot boundary violation during symbol window detected on channel A
TCSA	2	rh	**Transmission Conflict in Symbol Window Channel A (vSS! TxConflictA)** 0_B No transmission conflict detected 1_B Transmission conflict in symbol window detected on channel A
SESB	3	rh	**Syntax Error in Symbol Window Channel B (vSS!SyntaxErrorB)** 0_B No syntax error detected 1_B Syntax error during symbol window detected on channel B
SBSB	4	rh	**Slot Boundary Violation in Symbol Window Channel B (vSS! BViolationB)** 0_B No slot boundary violation detected 1_B Slot boundary violation during symbol window detected on channel B
TCSB	5	rh	**Transmission Conflict in Symbol Window Channel B (vSS! TxConflictB)** 0_B No transmission conflict detected 1_B Transmission conflict in symbol window detected on channel B
MTSA	6	rh	**MTS Received on Channel A (vSS!ValidMTSA)** Media Access Test symbol received on channel A during the proceeding symbol window. Updated by the Communication Controller for each channel at the end of the symbol window. When this bit is set to 1, also interrupt flag SIR.MTSA is set to 1. 0_B No MTS symbol received on channel A 1_B MTS symbol received on channel A
MTSB	7	rh	**MTS Received on Channel B (vSS!ValidMTSB)** Media Access Test symbol received on channel B during the proceeding symbol window. Updated by the Communication Controller for each channel at the end of the symbol window. When this bit is set to 1, also interrupt flag SIR.MTSB is set to 1. 0_B No MTS symbol received on channel B 1_B MTS symbol received on channel B
SENA	8	rh	**Syntax Error during network idle time (NIT) Channel A (vSS! SyntaxErrorA)** Updated by the Communication Controller channel A at the end of the NIT. 0_B No syntax error detected 1_B Syntax error during network idle time (NIT) detected on channel A

(table continues...)

(continued)

Field	Bits	Type	Description
SBNA	9	rh	**Slot Boundary Violation during network idle time (NIT) CH-A** Updated by the Communication Controller channel A at the end of the NIT. (vSS!BViolationA) - SBNA 0_B No slot boundary violation detected 1_B Slot boundary violation during network idle time (NIT) detected on channel A
SENB	10	rh	**Syntax Error during network idle time (NIT) Channel B (vSS! SyntaxErrorB)** Updated by the Communication Controller channel B at the end of the NIT. 0_B No syntax error detected 1_B Syntax error during network idle time (NIT) detected on channel B
SBNB	11	rh	**Slot Boundary Violation during network idle time (NIT) CH-B** Updated by the Communication Controller channel B at the end of the NIT. (vSS!BViolationB) - SBNB 0_B No slot boundary violation detected 1_B Slot boundary violation during network idle time (NIT) detected on channel B
0	31:12	r	**Reserved** Read as 0; should be written with 0.

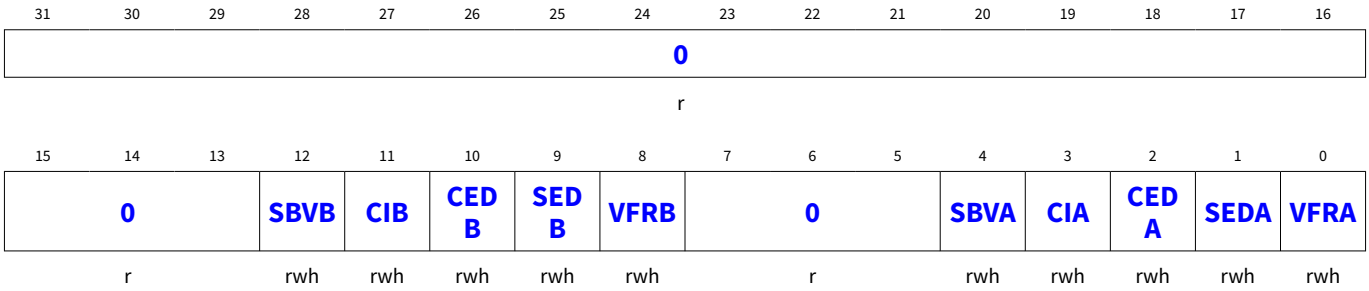
23.4.49 Aggregated Channel Status

The aggregated channel status provides the Host with an accrued status of channel activity for all communication slots regardless of whether they are assigned for transmission or subscribed for reception. The aggregated channel status also includes status data from the symbol window and the network idle time. The status data is updated (set) after each slot and aggregated until it is reset by the Host. During startup the status data is not updated. A flag is cleared by writing a 1 to the corresponding bit position. Writing a 0 has no effect on the flag. An application reset will also clear the register.

***Note**: The set condition of flags CIA and CIB is also fulfilled if there is only one single Frame in the slot and the slot boundary at the end of the slot is reached during the Frames channel idle recognition phase. When one of the flags SEDB, CEDB, CIB, SBVB changes from 0 to 1, service request flag EIR.EDB is set to 1.When one of the flags SEDA, CEDA, CIA, SBVA changes from 0 to 1, service request flag EIR.EDA is set to 1. LDMST or SWAPMSK.W instructions should be used only with bit mask enabled for all rwh bits in this register.*

ACS Offset address: 0128_H

Aggregated Channel Status Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
VFRA	0	rwh	**Valid Frame Received on Channel A (vSS!ValidFrameA)** One or more valid Frames were received on channel A in any static or dynamic slot during the observation period. 0_B No valid Frame received 1_B Valid Frame(s) received on channel A
SEDA	1	rwh	**Syntax Error Detected on Channel A (vSS!SyntaxErrorA)** One or more syntax errors in static or dynamic slots, symbol window, and network idle time (NIT) were observed on channel A. 0_B No syntax error observed 1_B Syntax error(s) observed on channel A
CEDA	2	rwh	**Content Error Detected on Channel A (vSS!ContentErrorA)** One or more Frames with a content error were received on channel A in any static or dynamic slot during the observation period. 0_B No Frame with content error received 1_B Frame(s) with content error received on channel A
CIA	3	rwh	**Communication Indicator Channel A** One or more valid Frames were received on channel A in slots that also contained any additional communication during the observation period, for example one or more slots received a valid Frame AND had any combination of either syntax error OR content error OR slot boundary violation. 0_B No valid Frame(s) received in slots containing any additional communication 1_B Valid Frame(s) received on channel A in slots containing any additional communication
SBVA	4	rwh	**Slot Boundary Violation on Channel A (vSS!BViolationA)** One or more slot boundary violations were observed on channel A at any time during the observation period (static or dynamic slots, symbol window, and network idle time NIT). 0_B No slot boundary violation observed 1_B Slot boundary violation(s) observed on channel A
VFRB	8	rwh	**Valid Frame Received on Channel B (vSS!ValidFrameB)** One or more valid Frames were received on channel B in any static or dynamic slot during the observation period. 0_B No valid Frame received 1_B Valid Frame(s) received on channel B

(table continues...)

(continued)

Field	Bits	Type	Description
SEDB	9	rwh	**Syntax Error Detected on Channel B (vSS!SyntaxErrorB)** One or more syntax errors in static or dynamic slots, symbol window, and network idle time (NIT) were observed on channel B. 0_B No syntax error observed 1_B Syntax error(s) observed on channel B
CEDB	10	rwh	**Content Error Detected on Channel B (vSS!ContentErrorB)** One or more Frames with a content error were received on channel B in any static or dynamic slot during the observation period. 0_B No Frame with content error received 1_B Frame(s) with content error received on channel B
CIB	11	rwh	**Communication Indicator Channel B** One or more valid Frames were received on channel B in slots that also contained any additional communication during the observation period, for example one or more slots received a valid Frame AND had any combination of either syntax error OR content error OR slot boundary violation. 0_B No valid Frame(s) received in slots containing any additional communication 1_B Valid Frame(s) received on channel B in slots containing any additional communication
SBVB	12	rwh	**Slot Boundary Violation on Channel B (vSS!BViolationB)** One or more slot boundary violations were observed on channel B at any time during the observation period (static or dynamic slots, symbol window, and network idle time NIT). 0_B No slot boundary violation observed 1_B Slot boundary violation(s) observed on channel B
0	7:5, 31:13	r	**Reserved** Read as 0; should be written with 0.

23.4.50 Even Sync ID Symbol Window n

Registers Even Sync ID n (ESIDn, n=01-15) hold the Frame IDs of the SYNC Frames received in **even** communication cycles, sorted in ascending order, with register ESID01 holding the lowest received SYNC Frame ID. If the node itself transmits a SYNC Frame in an even communication cycle, register ESID01 holds the respective SYNC Frame ID as configured in Message Buffer 0 and the flags RXEA, RXEB are set. The value is updated during the network idle time (NIT) of each even communication cycle.

ESIDn (n=01-15) Offset address: 0130_H+(n-1)*4

Even Sync ID Symbol Window n Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXEB	RXEA	0				EID									
rh	rh	r				rh									

Field	Bits	Type	Description
EID	9:0	rh	**Even Sync ID (vsSyncIDListA,B even)** SYNC Frame ID even communication cycle.
RXEA	14	rh	**Received/Configured Even Sync ID on Channel A** Signals that a SYNC Frame corresponding to the stored even sync ID was received on channel A or that the node is configured to be a sync node with key slot = EID (ESID1 only). 0_B SYNC Frame not received on channel A / node configured to transmit SYNC Frames 1_B SYNC Frame received on channel A/ node not configured to transmit SYNC Frames
RXEB	15	rh	**Received/Configured Even Sync ID on Channel B** Signals that a SYNC Frame corresponding to the stored even sync ID was received on channel B or that the node is configured to be a sync node with key slot = EID (ESID1 only). 0_B SYNC Frame not received on channel B / node configured to transmit SYNC Frames 1_B SYNC Frame received on channel B / node not configured to transmit SYNC Frames
0	13:10, 31:16	r	**Reserved** Read as 0; should be written with 0.

23.4.51 Odd Sync ID Symbol Window n

Registers Odd Sync ID n (OSIDn, n=01-15) hold the Frame IDs of the SYNC Frames received in **odd** communication cycles, sorted in ascending order, with register OSID01 holding the lowest received SYNC Frame ID. If the node itself transmits a SYNC Frame in an odd communication cycle, register OSID01 holds the respective SYNC Frame ID as configured in Message Buffer 0 and flags RXOA, RXOB are set. The value is updated during the network idle time (NIT) of each odd communication cycle.

OSIDn (n=01-15) Offset address: $0170_H+(n-1)*4$

Odd Sync ID Symbol Window n Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXOB	RXOA	0				OID									
rh	rh	r				rh									

Field	Bits	Type	Description
OID	9:0	rh	**Odd Sync ID(vsSyncIDListA,B odd)** SYNC Frame ID even communication cycle.

(table continues...)

(continued)

Field	Bits	Type	Description
RXOA	14	rh	**Received Odd Sync ID on Channel A** Signals that a SYNC Frame corresponding to the stored odd sync ID was received on channel A or that the node is configured to be a sync node with key slot = OID (OSID1 only). 0_B SYNC Frame not received on channel A / node configured to transmit SYNC Frames 1_B SYNC Frame received on channel A / node not configured to transmit SYNC Frames
RXOB	15	rh	**Received Odd Sync ID on Channel B** Signals that a SYNC Frame corresponding to the stored odd sync ID was received on channel B or that the node is configured to be a sync node with key slot = OID (OSID1 only) 0_B SYNC Frame not received on channel B / node configured to transmit SYNC Frames 1_B SYNC Frame received on channel B/ node not configured to transmit SYNC Frames
0	13:10, 31:16	r	**Reserved** Read as 0; should be written with 0.

23.4.52 Network Management Vector x

The three Network Management Vectors n (NMVx, x=1-3) registers hold the accrued Network Management (NM) vector (configurable 0 to 12 byte). The accrued Network Management (NM) vector is generated by the Communication Controller by bit-wise ORing each Network Management (NM) vector received (valid static Frames with PPI = 1) on each channel (see “Network Management”). The Communication Controller updates the Network Management (NM) vector at the end of each communication cycle as long as the Communication Controller is either in “NORMAL_ACTIVE” or “NORMAL_PASSIVE” state. NMVx-bytes exceeding the configured Network Management (NM) vector length are not valid.

NMVx (x=1-3) Offset address: $01B0_H+(x-1)*4$

Network Management Vector x Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
NM															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NM															
rh															

Field	Bits	Type	Description
NM	31:0	rh	**Network Management Vector**

Table below shows the assignment of the received payload’s data byte to the Network Management vector.

Table 1097 Assignment of data bytes to network management vector

Bit Word	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NM1	Data3								Data2								Data1								Data0							
NM2	Data7								Data6								Data5								Data4							
NM3	Data11								Data10								Data9								Data8							

23.4.53 Message RAM Configuration

The Message RAM Configuration register defines the number of Message Buffers assigned to the static segment, dynamic segment, and FIFO. The register can be written during “DEFAULT_CONFIG” or “CONFIG” state only.

MRC Offset address: 0300_H

Message RAM Configuration Kernel Reset value: $0180\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0					SPLM	SEC		LCB							
r					rw	rw		rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FFB								FDB							
rw								rw							

Field	Bits	Type	Description
FDB	7:0	rw	**First Dynamic Buffer** May be modified in “DEFAULT_CONFIG” or “CONFIG” state only. 00_H No group of Message Buffers exclusively for the static segment configured 01_H …$7F_H$ Message Buffers 0 to FDB-1 reserved for static segment 80_H …FF_H No dynamic Message Buffers configured
FFB	15:8	rw	**First Buffer of FIFO** May be modified in “DEFAULT_CONFIG” or “CONFIG” state only. 00_H …$7E_H$ Message Buffers from FFB to LCB assigned to the FIFO $7F_H$ All Message Buffers assigned to the FIFO 80_H …FF_H No Message Buffers assigned to the FIFO
LCB	23:16	rw	**Last Configured Buffer** May be only modified in “DEFAULT_CONFIG” or “CONFIG” state. 01_H …$7F_H$ Number of Message Buffers is LCB + 1 80_H …FF_H No Message Buffer configured

(table continues...)

(continued)

Field	Bits	Type	Description
SEC	25:24	rw	**Secure Buffers** Not evaluated when the Communication Controller is in “DEFAULT_CONFIG” or “CONFIG” state. For temporary unlocking see “Host Handling of Errors”. ***Note**: In nodes configured for SYNC Frame transmission or for single slot mode operation Message Buffer 0 (and if SPLM = 1, also Message Buffer 1) Reconfiguration of all Message Buffers is always locked* 00_B Reconfiguration of Message Buffers enabled with numbers < FFB enabled. 01_B Reconfiguration of Message Buffers with numbers < FDB and with numbers ≥ FFB locked and transmission of Message Buffers for static segment with numbers ≥ FDB disabled 10_B Reconfiguration of all Message Buffers locked 11_B Reconfiguration of all Message Buffers locked and transmission of Message Buffers for static segment with numbers ≥ FDB disabled
SPLM	26	rw	**SYNC Frame Payload Multiplex** This bit is only evaluated if the node is configured as sync node (SUCC1.TXSY = 1) or for single slot mode operation (SUCC1.TSM = 1). When this bit is set to 1 Message Buffers 0 and 1 are dedicated for SYNC Frame transmission with different payload data on channel A and B. When this bit is reset to 0, SYNC Frames are transmitted from Message Buffer 0 with the same payload data on both channels. Note that the channel filter configuration for Message Buffer 0 resp. Message Buffer 1 has to be chosen accordingly. 0_B Only Message Buffer 0 locked against reconfiguration 1_B Both Message Buffers 0 and 1 are locked against reconfiguration
0	31:27	r	**Reserved** Read as 0; should be written with 0.

***Note**: In case the node is configured as sync node (SUCC1.TXSY = 1) or for single slot mode operation (SUCC1.TSM = 1), Message Buffer 0 resp. 1 is reserved for SYNC Frames or single slot Frames and have to be configured with the node-specific key slot ID. In case the node is neither configured as sync node nor for single slot operation Message Buffer 0 resp. 1 is treated like all other Message Buffers.*

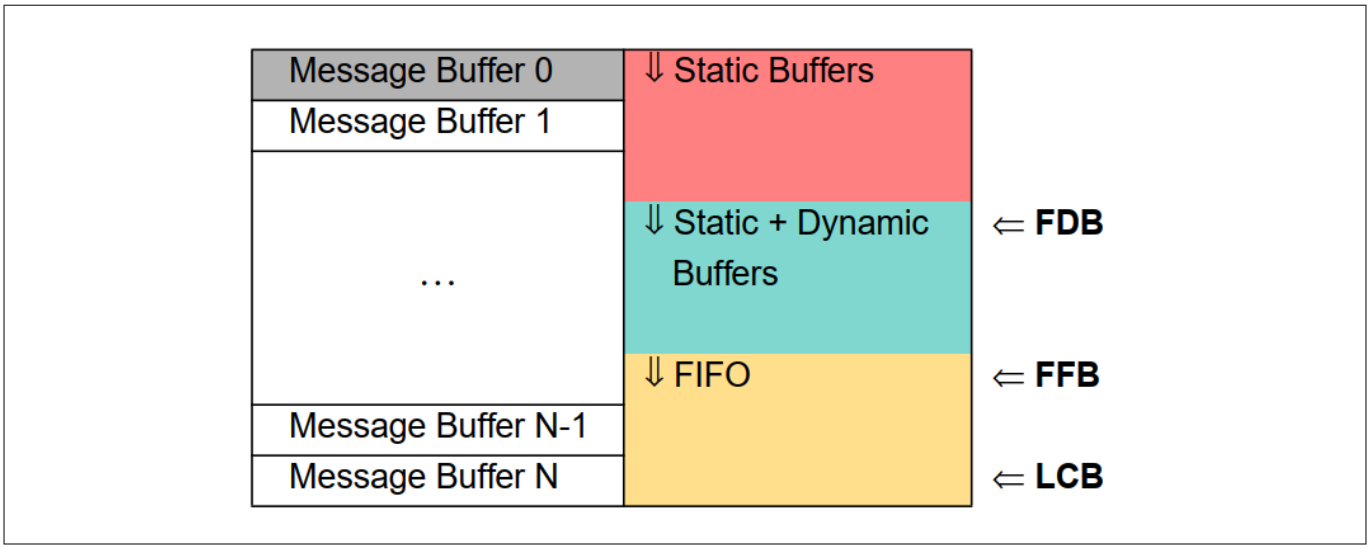


Figure 455 Assignment of message buffers

The programmer has to ensure that the configuration defined by FDB, FFB, and LCB is valid. **The Communication Controller does not check for erroneous configurations.**

The maximum number of Header Sections is 128. This means a maximum of 128 Message Buffer can be configured. The maximum length of a Data Section is 254 byte. The length of the Data Section may be configured differently for each Message Buffer. For details see “Message RAM”. In case two or more Message Buffers are assigned to slot 1 by use of cycle filtering, all of them must be located either in the “Static Buffers” or at the beginning of the “Static + Dynamic Buffers” section. The payload length configured and the length of the Data Section need to be configured identically for all Message Buffers belonging to the FIFO via WRHS2.PLC and WRHS3.DP. When the Communication Controller is not in “DEFAULT_CONFIG” or “CONFIG” state reconfiguration of Message Buffers belonging to the FIFO is locked.

23.4.54 FIFO Rejection Filter

The FIFO Rejection Filter defines a user specified sequence of bits to which channel, Frame ID, and cycle count of the incoming Frames are compared. Together with the FIFO Rejection Filter Mask this register determines whether a message is rejected by the FIFO. The FRF register can be written during “DEFAULT_CONFIG” or “CONFIG” state only.

FRF Offset address: 0304_H

FIFO Rejection Filter Kernel Reset value: $0180\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0							RNF	RSS	CYF						
r							rw	rw	rw						

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			FID											CH	
r			rw											rw	

Field	Bits	Type	Description
CH	1:0	rw	**Channel Filter** May be modified in “DEFAULT_CONFIG” or “CONFIG” state only. 00_B receive on both channels[1)] 01_B receive only on channel B 10_B receive only on channel A 11_B no reception
FID	12:2	rw	**Frame ID Filter** Determines the Frame ID to be rejected by the FIFO. With the additional configuration of register FRFM, the corresponding Frame ID filter bits are ignored, which results in further rejected Frame IDs. When FRFM.MFID is zero, a Frame ID filter value of zero means that no Frame ID is rejected. 000_H …$7FF_H$ Frame ID filter values
CYF	22:16	rw	**Cycle Counter Filter** The 7-bit cycle counter filter determines the cycle set to which Frame ID and channel rejection filter are applied. In cycles **not** belonging to the cycle set specified by CYF, all Frames are rejected. For details about the configuration of the cycle counter filter see “Cycle Counter Filtering”. May be modified in “DEFAULT_CONFIG” or “CONFIG” state only.
RSS	23	rw	**Reject in Static Segment** If this bit is set, the FIFO is used only for the dynamic segment. May be modified in “DEFAULT_CONFIG” or “CONFIG” state only. 0_B FIFO also used in static segment 1_B Reject messages for static segment
RNF	24	rw	**Reject NULL Frames** If this bit is set, received NULL Frames are not stored in the FIFO. May be modified in “DEFAULT_CONFIG” or “CONFIG” state only. 0_B NULL Frames are stored in the FIFO 1_B Reject all NULL Frames
0	15:13, 31:25	r	**Reserved** Read as 0; should be written with 0.

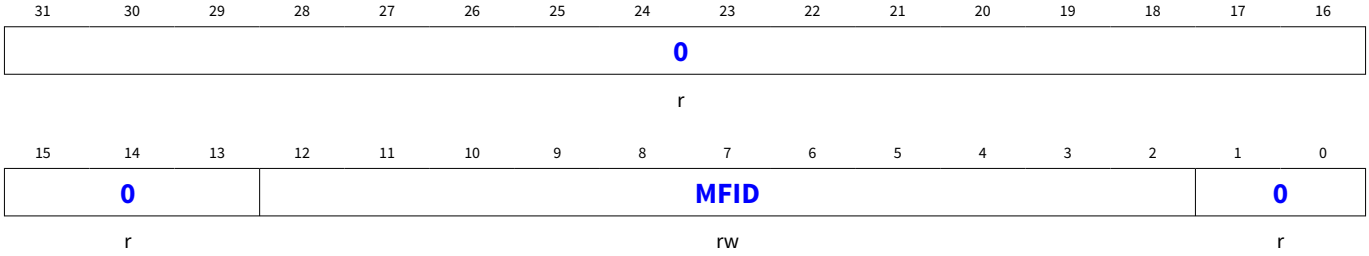
1) If reception on both channels is configured, also in static segment always both Frames (from channel A and B) are stored in the FIFO, even if they are identical.

23.4.55 FIFO Rejection Filter Mask

The FIFO Rejection Filter Mask specifies which of the corresponding Frame ID filter bits are relevant for rejection filtering. If a bit is set, it indicates that the corresponding bit in the FRF register will not be considered for rejection filtering. The FRFM register can be written during “DEFAULT_CONFIG” or “CONFIG” state only.

FRFM Offset address: 0308_H

FIFO Rejection Filter Mask Kernel Reset value: $0000\ 0000_H$



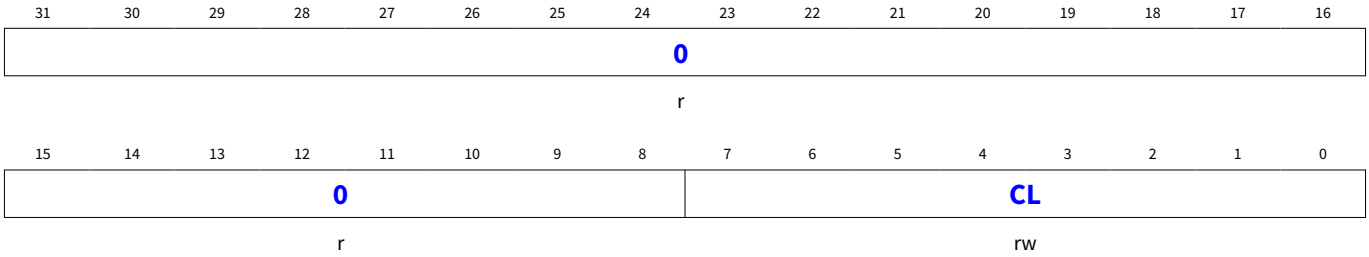
Field	Bits	Type	Description
MFID	12:2	rw	**Mask Frame ID Filter** May be modified in “DEFAULT_CONFIG” or “CONFIG” state only. When '0' written in a bit position, the corresponding Frame ID filter bit is used for rejection filtering. When '1' written in a bit position, the corresponding Frame ID filter bit is ignored. Valid values are from 0x000 - 0x3FF.
0	1:0, 31:13	r	**Reserved** Read as 0; should be written with 0.

23.4.56 FIFO Critical Level

The Communication Controller accepts modifications of the register in “DEFAULT_CONFIG” or “CONFIG” state only.

FCL Offset address: $030C_H$

FIFO Critical Level Kernel Reset value: $0000\ 0080_H$



Field	Bits	Type	Description
CL	7:0	rw	**Critical Level** When the receive FIFO fill level FSR.RFFL is equal or greater than the critical level configured by CL, the receive FIFO critical level flag FSR.RFCL is set. If CL is programmed to values > 128, bit FSR.RFCL is never set. When FSR.RFCL changes from 0 to 1, bit SIR.RFCL is set to 1, and if enabled, a service request is generated.
0	31:8	r	**Reserved** Read as 0; should be written with 0.

23.4.57 Message Handler Status

The Message Handler Status register gives the Host access to the current state of the Message Handler. A flag is cleared by writing a 1 to the corresponding bit position. Writing a 0 has no effect on the flag. An application reset will also clear the register. If one of the flags MHDS.EIBF, MHDS.EOBF, MHDS.EMR, MHDS.ETBF1, MHDS.ETBF2 changes from 0 to 1 EIR.EERR is set.

***Note**: LDMST or SWAPMSK.W instructions should be used only with bit mask enabled for all ´rwh´ bits in this register.*

MHDS Offset address: 0310_H

Message Handler Status Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	MBU							0	MBT						
r	rh							r	rh						

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	FMB							CRAM	MFMB	FMBD	ETBF2	ETBF1	EMR	EOBF	EIBF
r	rh							rh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
EIBF	0	rwh	**ECC Error Input Buffer RAM 1,2** 0_B No error 1_B Error occurred when reading Input Buffer RAM 1 or Input Buffer RAM 2
EOBF	1	rwh	**ECC Error Output Buffer RAM 1,2** 0_B No error 1_B Error occurred when reading Output Buffer RAM 1 or Output Buffer RAM 2
EMR	2	rwh	**ECC Error Message RAM** 0_B No error 1_B Error occurred when reading the Message RAM
ETBF1	3	rwh	**ECC Error Transient Buffer RAM A** 0_B No error 1_B Error occurred when reading Transient Buffer RAM A
ETBF2	4	rwh	**ECC Error Transient Buffer RAM B** 0_B No error 1_B Error occurred when reading Transient Buffer RAM B
FMBD	5	rwh	**Faulty Message Buffer Detected** 0_B No faulty Message Buffer 1_B Message Buffer referenced by MHDS.FMB holds faulty data due to a ECC error

(table continues...)

(continued)

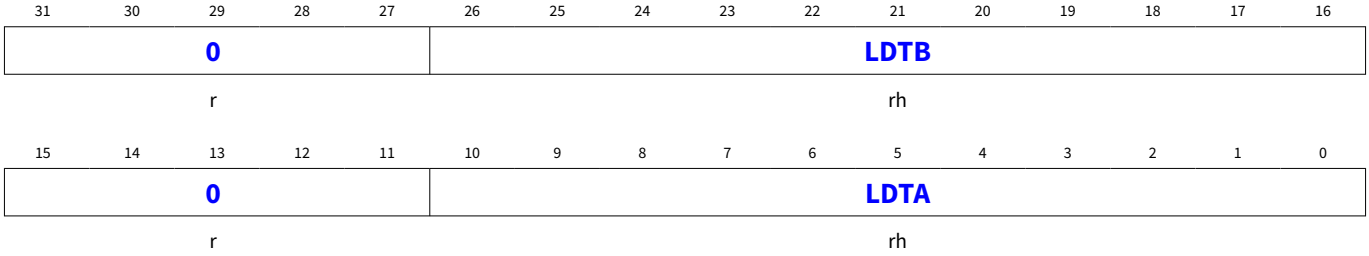
Field	Bits	Type	Description
MFMB	6	rwh	**Multiple Faulty Message Buffers detected** 0_B No additional faulty Message Buffer 1_B Another faulty Message Buffer was detected while flag MHDS.FMBD is set
CRAM	7	rh	**Clear all internal RAM's** Signals that execution of the CHI command CLEAR_RAMS is ongoing (all bits of all internal RAM blocks are written to 0). The bit is set by CHI command CLEAR_RAMS. 0_B No execution of the CHI command CLEAR_RAMS 1_B Execution of the CHI command CLEAR_RAMS ongoing
FMB	14:8	rh	**Faulty Message Buffer** ECC error occurred when reading from the Message Buffer or when transferring data from Input Buffer or Transient Buffer A or Transient Buffer B to the Message Buffer referenced by MHDS.FMB. Value only valid when one of the flags MHDS.EIBF, MHDS.EMR, MHDS.ETBF1, MHDS.ETBF2, and flag MHDS.FMBD is set. Updated only after the Host has reset flag MHDS.FMBD.
MBT	22:16	rh	**Message Buffer Transmitted** Number of last successfully transmitted Message Buffer. If the Message Buffer is configured for single-shot mode, the respective TXR flag in the Transmission Request Registers TXRQ1 to TXRQ4 was reset. MBT is reset when the Communication Controller leaves "CONFIG" state or enters "STARTUP" state.
MBU	30:24	rh	**Message Buffer Updated** Number of Message Buffer that was updated last. For this Message Buffer the respective NDn (n = 0-31) to NDn (n = 96-127)and / or MBCn (n = 0-31) to MBCn (n = 96-127) flag in the New Data Registers NDAT1 to NDAT4 and the Message Buffer Status Changed MBSC1 to MBSC4 registers are also set. MBU is reset when the Communication Controller leaves "CONFIG" state or enters "STARTUP" state.
0	15, 23, 31	r	**Reserved** Read as 0; should be written with 0.

23.4.58 Last Dynamic Transmit Slot

The Last Dynamic Transmit Slot Register stores the Slot Counter value at the time of the last Frame transmission in the dynamic segment. This register is reset when the Communication Controller leaves "CONFIG" state or enters "STARTUP" state.

LDTS Offset address: 0314_H

Last Dynamic Transmit Slot Kernel Reset value: $0000\ 0000_H$



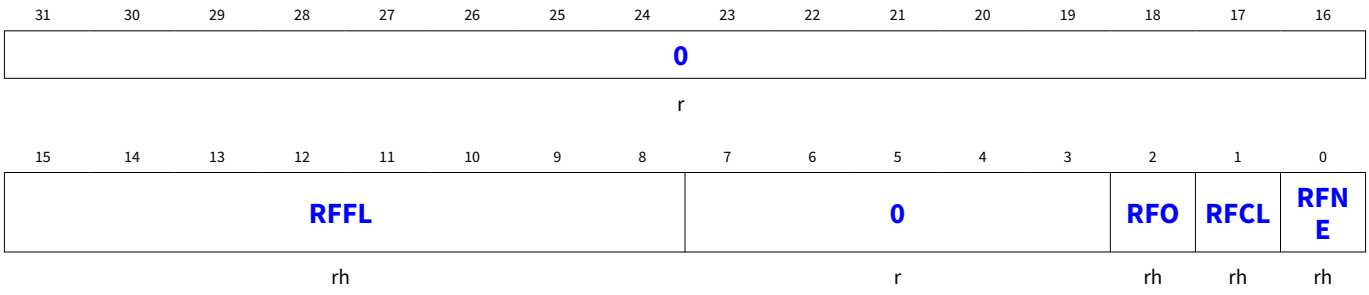
Field	Bits	Type	Description
LDTA	10:0	rh	**Last Dynamic Transmission Channel A** Value of (vSlotCounter[A]) at the time of the last Frame transmission on channel A in the dynamic segment of this node. It is updated at the end of the dynamic segment and is reset to zero if no Frame was transmitted during the dynamic segment.
LDTB	26:16	rh	**Last Dynamic Transmission Channel B** Value of (vSlotCounter[B]) at the time of the last Frame transmission on channel B in the dynamic segment of this node. It is updated at the end of the dynamic segment and is reset to zero if no Frame was transmitted during the dynamic segment.
0	15:11, 31:27	r	**Reserved** Read as 0; should be written with 0.

23.4.59 FIFO Status Register

The register is reset when the Communication Controller leaves “CONFIG” state or enters “STARTUP” state.

FSR Offset address: 0318_H

FIFO Status Register Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
RFNE	0	rh	**Receive FIFO Not Empty** This flag is set by the Communication Controller when a received valid Frame (data or NULL Frame depending on rejection mask) was stored in the FIFO. In addition, service request flag SIR.RFNE is set. The bit is reset after the Host has read all message from the FIFO. 0_B Receive FIFO is empty 1_B Receive FIFO is not empty

(table continues...)

(continued)

Field	Bits	Type	Description
RFCL	1	rh	**Receive FIFO Critical Level** This flag is set when the receive FIFO fill level RFFL is equal or greater than the critical level as configured by FCL.CL. The flag is cleared by the Communication Controller as soon as RFFL drops below FCL.CL. When RFCL changes from 0 to 1, bit SIR.RFCL is set to 1, and if enabled, an service request is generated. 0_B Receive FIFO below critical level 1_B Receive FIFO critical level reached
RFO	2	rh	**Receive FIFO Overrun** The flag is set by the Communication Controller when a receive FIFO overrun is detected. When a receive FIFO overrun occurs, the oldest message is overwritten with the actual received message. In addition, service request flag EIR.RFO is set. The flag is cleared by the next FIFO read access issued by the Host. 0_B No receive FIFO overrun detected 1_B A receive FIFO overrun has been detected
RFFL	15:8	rh	**Receive FIFO Fill Level** Number of FIFO buffers filled up with new data not yet read by the Host. Maximum value is 128.
0	7:3, 31:16	r	**Reserved** Read as 0; should be written with 0.

23.4.60 Message Handler Constraints Flags

Some constraints exist for the Message Handler regarding f_{CLC_ERAY} frequency, Message RAM configuration, and FlexRay™ bus traffic. To simplify software development, constraints violations are reported by setting flags in the MHDF. The register is reset when the Communication Controller leaves “CONFIG” state or enters “STARTUP” state. A flag is cleared by setting the corresponding bit position. Clearing has no effect on the flag. Software write of '1' when a flag is already cleared has no effect. If any flag in MHDFL is set, interrupt flag EIR.MHF is set.

***Note**: LDMST or SWAPMSK.W instructions should be used only with bit mask enabled for all ´rwh´ bits in this register.*

MHDF Offset address: $031C_H$

Message Handler Constraints Flags Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0							WAHP	TNSB	TNSA	TBFB	TBFA	FNFB	FNFA	SNUB	SNUA
r							rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
SNUA	0	rwh	**Status Not Updated Channel A** This flag is set by the Communication Controller when the Message Handler, due to overload condition, was not able to update a Message Buffer's status MBS with respect to channel A. 0_B No overload condition occurred when updating MBS for channel A 1_B MBS for channel A not updated
SNUB	1	rwh	**Status Not Updated Channel B** This flag is set by the Communication Controller when the Message Handler, due to overload condition, was not able to update a Message Buffer's status MBS with respect to channel B. 0_B No overload condition occurred when updating MBS for channel B 1_B MBS for channel B not updated
FNFA	2	rwh	**Find Sequence Not Finished Channel A** This flag is set by the Communication Controller when the Message Handler, due to overload condition, was not able to finish a find sequence (scan of Message RAM for matching Message Buffer) with respect to channel A. 0_B No find sequence not finished for channel A 1_B Find sequence not finished for channel A
FNFB	3	rwh	**Find Sequence Not Finished Channel B** This flag is set by the Communication Controller when the Message Handler, due to overload condition, was not able to finish a find sequence (scan of Message RAM for matching Message Buffer) with respect to channel B. 0_B No find sequence not finished for channel B 1_B Find sequence not finished for channel B
TBFA	4	rwh	**Transient Buffer Access Failure A** This flag is set by the Communication Controller when a read or write access to Transient Buffer A requested by PRT A could not complete within the available time. 0_B No TBF A access failure 1_B TBF A access failure
TBFB	5	rwh	**Transient Buffer Access Failure B** This flag is set by the Communication Controller when a read or write access to Transient Buffer B requested by PRT B could not complete within the available time. 0_B No Transient Buffer B access failure 1_B Transient Buffer B access failure
TNSA	6	rwh	**Transmission Not Started Channel A** This flag is set by the CC when the Message Handler was not ready to start a scheduled transmission on channel A at the action point of the configured slot. 0_B No transmission not started on channel A 1_B Transmission not started on channel A

(table continues...)

(continued)

Field	Bits	Type	Description
TNSB	7	rwh	**Transmission Not Started Channel B** This flag is set by the CC when the Message Handler was not ready to start a scheduled transmission on channel B at the action point of the configured slot. 0_B No transmission not started on channel B 1_B Transmission not started on channel B
WAHP	8	rwh	**Write Attempt to Header Partition** Outside “DEFAULT_CONFIG” and “CONFIG” state this flag is set by the Communication Controller when the Message Handler tries to write message data into the Header Partition of the Message RAM due to faulty configuration of a Message Buffer. The write attempt is not executed, to protect the Header Partition from unintended write accesses. 0_B No write attempt to Header Partition 1_B Write attempt to Header Partition
0	31:9	r	**Reserved** Read as 0; should be written with 0.

23.4.61 Transmission Request Register 1

This register reflect the state of the TXR flags of the configured Message Buffers 0 to 31. The flags are evaluated for transmit buffers only. If the number of configured Message Buffers is less than 31, the remaining TXRn flags have no meaning and are read as 0.

TXRQ1 Offset address: 0320_H

Transmission Request Register 1 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TXR31	TXR30	TXR29	TXR28	TXR27	TXR26	TXR25	TXR24	TXR23	TXR22	TXR21	TXR20	TXR19	TXR18	TXR17	TXR16
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXR15	TXR14	TXR13	TXR12	TXR11	TXR10	TXR9	TXR8	TXR7	TXR6	TXR5	TXR4	TXR3	TXR2	TXR1	TXR0
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
TXRn (n=0-31)	n	rh	**Transmission Request n (n = 0-31)** If the flag is set, the respective Message Buffer 0 to 31 is ready for transmission respectively transmission of this Message Buffer is in progress. In single-shot mode the flags are reset after transmission has completed.

23.4.62 Transmission Request Register 2

This register reflect the state of the TXR flags of the configured Message Buffers 32 to 63. The flags are evaluated for transmit buffers only. If the number of configured Message Buffers is less than 63, the remaining TXRn flags have no meaning and are read as 0.

TXRQ2 Offset address: 0324_H

Transmission Request Register 2 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TXR63	TXR62	TXR61	TXR60	TXR59	TXR58	TXR57	TXR56	TXR55	TXR54	TXR53	TXR52	TXR51	TXR50	TXR49	TXR48
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXR47	TXR46	TXR45	TXR44	TXR43	TXR42	TXR41	TXR40	TXR39	TXR38	TXR37	TXR36	TXR35	TXR34	TXR33	TXR32
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
TXRn (n=32-63)	n-32	rh	**Transmission Request n (n = 32-63)** If the flag is set, the respective Message Buffer 32 to 63 is ready for transmission respectively transmission of this Message Buffer is in progress. In single-shot mode the flags are reset after transmission has completed.

23.4.63 Transmission Request Register 3

This register reflect the state of the TXR flags of the configured Message Buffers 64 to 95. The flags are evaluated for transmit buffers only. If the number of configured Message Buffers is less than 95, the remaining TXRn flags have no meaning and are read as 0.

TXRQ3 Offset address: 0328_H

Transmission Request Register 3 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TXR95	TXR94	TXR93	TXR92	TXR91	TXR90	TXR89	TXR88	TXR87	TXR86	TXR85	TXR84	TXR83	TXR82	TXR81	TXR80
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXR79	TXR78	TXR77	TXR76	TXR75	TXR74	TXR73	TXR72	TXR71	TXR70	TXR69	TXR68	TXR67	TXR66	TXR65	TXR64
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
TXRn (n=64-95)	n-64	rh	**Transmission Request n (n = 64-95)** If the flag is set, the respective Message Buffer 64 to 95 is ready for transmission respectively transmission of this Message Buffer is in progress. In single-shot mode the flags are reset after transmission has completed.

23.4.64 Transmission Request Register 4

This register reflect the state of the TXR flags of the configured Message Buffers 96 to 127. The flags are evaluated for transmit buffers only. If the number of configured Message Buffers is less than 127, the remaining TXRn flags have no meaning and are read as 0.

TXRQ4 Offset address: $032C_H$

Transmission Request Register 4 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TXR127	TXR126	TXR125	TXR124	TXR123	TXR122	TXR121	TXR120	TXR119	TXR118	TXR117	TXR116	TXR115	TXR114	TXR113	TXR112
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXR111	TXR110	TXR109	TXR108	TXR107	TXR106	TXR105	TXR104	TXR103	TXR102	TXR101	TXR100	TXR99	TXR98	TXR97	TXR96
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
TXRn (n=96-127)	n-96	rh	**Transmission Request n (n = 96-127)** If the flag is set, the respective Message Buffer 96 to 127 is ready for transmission respectively transmission of this Message Buffer is in progress. In single-shot mode the flags are reset after transmission has completed.

23.4.65 New Data Register 1

This register reflect the state of the ND flags of all configured Message Buffers 0 to 31. ND flags assigned to transmit buffers are meaningless. If the number of configured Message Buffers is less than 31, the remaining NDn flags have no meaning. The registers are reset when the Communication Controller leaves “CONFIG” state or enters “STARTUP” state.

NDAT1 Offset address: 0330_H

New Data Register 1 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ND31	ND30	ND29	ND28	ND27	ND26	ND25	ND24	ND23	ND22	ND21	ND20	ND19	ND18	ND17	ND16
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ND15	ND14	ND13	ND12	ND11	ND10	ND9	ND8	ND7	ND6	ND5	ND4	ND3	ND2	ND1	ND0
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
NDn (n=0-31)	n	rh	**New Data n (n = 0-31)** The flags are set when a valid received Data Frame matches the Message Buffer's filter configuration, independent of the payload length received or the payload length configured for that Message Buffer. The flags are not set after reception of NULL Frames except for Message Buffers belonging to the receive FIFO. An ND flag is reset when the Header Section of the corresponding Message Buffer is reconfigured or when the Data Section has been transferred to the Output Buffer.

23.4.66 New Data Register 2

This register reflect the state of the ND flags of all configured Message Buffers 32 to 63. ND flags assigned to transmit buffers are meaningless. If the number of configured Message Buffers is less than 63, the remaining NDn flags have no meaning. The registers are reset when the Communication Controller leaves "CONFIG" state or enters "STARTUP" state.

NDAT2 Offset address: 0334_H

New Data Register 2 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ND63	ND62	ND61	ND60	ND59	ND58	ND57	ND56	ND55	ND54	ND53	ND52	ND51	ND50	ND49	ND48
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ND47	ND46	ND45	ND44	ND43	ND42	ND41	ND40	ND39	ND38	ND37	ND36	ND35	ND34	ND33	ND32
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
NDn (n=32-63)	n-32	rh	**New Data n (n = 32-63)** The flags are set when a valid received Data Frame matches the Message Buffer's filter configuration, independent of the payload length received or the payload length configured for that Message Buffer. The flags are not set after reception of NULL Frames except for Message Buffers belonging to the receive FIFO. An ND flag is reset when the Header Section of the corresponding Message Buffer is reconfigured or when the Data Section has been transferred to the Output Buffer.

23.4.67 New Data Register 3

This register reflect the state of the ND flags of all configured Message Buffers 64 to 95. ND flags assigned to transmit buffers are meaningless. If the number of configured Message Buffers is less than 95, the remaining NDn flags have no meaning. The registers are reset when the Communication Controller leaves "CONFIG" state or enters "STARTUP" state.

NDAT3 Offset address: 0338_H

New Data Register 3 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ND95	ND94	ND93	ND92	ND91	ND90	ND89	ND88	ND87	ND86	ND85	ND84	ND83	ND82	ND81	ND80
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ND79	ND78	ND77	ND76	ND75	ND74	ND73	ND72	ND71	ND70	ND69	ND68	ND67	ND66	ND65	ND64
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
NDn (n=64-95)	n-64	rh	**New Data n (n = 64-95)** The flags are set when a valid received Data Frame matches the Message Buffer's filter configuration, independent of the payload length received or the payload length configured for that Message Buffer. The flags are not set after reception of NULL Frames except for Message Buffers belonging to the receive FIFO. An ND flag is reset when the Header Section of the corresponding Message Buffer is reconfigured or when the Data Section has been transferred to the Output Buffer.

23.4.68 New Data Register 4

This register reflect the state of the ND flags of all configured Message Buffers 96 to 127. ND flags assigned to transmit buffers are meaningless. If the number of configured Message Buffers is less than 127, the remaining NDn flags have no meaning. The registers are reset when the Communication Controller leaves "CONFIG" state or enters "STARTUP" state.

NDAT4 Offset address: $033C_H$

New Data Register 4 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ND127	ND126	ND125	ND124	ND123	ND122	ND121	ND120	ND119	ND118	ND117	ND116	ND115	ND114	ND113	ND112
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ND111	ND110	ND109	ND108	ND107	ND106	ND105	ND104	ND103	ND102	ND101	ND100	ND99	ND98	ND97	ND96
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
NDn (n=96-127)	n-96	rh	**New Data n (n = 96-127)** The flags are set when a valid received Data Frame matches the Message Buffer's filter configuration, independent of the payload length received or the payload length configured for that Message Buffer. The flags are not set after reception of NULL Frames except for Message Buffers belonging to the receive FIFO. An ND flag is reset when the Header Section of the corresponding Message Buffer is reconfigured or when the Data Section has been transferred to the Output Buffer.

23.4.69 Message Buffer Status Changed 1

This register reflect the state of the MBC flags of all configured Message Buffers. If the number of configured Message Buffers is less than 31, the remaining MBCn flags have no meaning. The register is reset when the Communication Controller leaves “CONFIG” state or enters “STARTUP” state.

MBSC1 Offset address: 0340_H

Message Buffer Status Changed 1 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MBC31	MBC30	MBC29	MBC28	MBC27	MBC26	MBC25	MBC24	MBC23	MBC22	MBC21	MBC20	MBC19	MBC18	MBC17	MBC16
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MBC15	MBC14	MBC13	MBC12	MBC11	MBC10	MBC9	MBC8	MBC7	MBC6	MBC5	MBC4	MBC3	MBC2	MBC1	MBC0
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
MBCn (n=0-31)	n	rh	**Message Buffer Status Changed n (n = 0-31)** An MBC flags is set whenever the Message Handler changes on of the status flags VFRA, VFRB, SEOA, SEOB, CEOA, CEOB, SVOA, SVOB, TCIA, TCIB, ESA, ESB, MLST, FTA, FTB in the Header Section (see “Message Buffer Status”) of the respective Message Buffer 0 to Message Buffer 31. The flags are reset when the Header Section of the Message Buffer is reconfigured or when it has been transferred to the Output Buffer.

23.4.70 Message Buffer Status Changed 2

This register reflect the state of the MBC flags of all configured Message Buffers. If the number of configured Message Buffers is less than 63, the remaining MBCn flags have no meaning. The register is reset when the Communication Controller leaves “CONFIG” state or enters “STARTUP” state.

MBSC2 Offset address: 0344_H

Message Buffer Status Changed 2 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MBC63	MBC62	MBC61	MBC60	MBC59	MBC58	MBC57	MBC56	MBC55	MBC54	MBC53	MBC52	MBC51	MBC50	MBC49	MBC48
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MBC47	MBC46	MBC45	MBC44	MBC43	MBC42	MBC41	MBC40	MBC39	MBC38	MBC37	MBC36	MBC35	MBC34	MBC33	MBC32
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
MBCn (n=32-63)	n-32	rh	**Message Buffer Status Changed n (n = 32-63)** An MBC flags is set whenever the Message Handler changes on of the status flags VFRA, VFRB, SEOA, SEOB, CEOA, CEOB, SVOA, SVOB, TCIA, TCIB, ESA, ESB, MLST, FTA, FTB in the Header Section (see “Message Buffer Status”) of the respective Message Buffer 32 to Message Buffer 63. The flags are reset when the Header Section of the Message Buffer is reconfigured or when it has been transferred to the Output Buffer.

23.4.71 Message Buffer Status Changed 3

This register reflect the state of the MBC flags of all configured Message Buffers. If the number of configured Message Buffers is less than 95, the remaining MBCn flags have no meaning. The register is reset when the Communication Controller leaves “CONFIG” state or enters “STARTUP” state.

MBSC3 Offset address: 0348_H

Message Buffer Status Changed 3 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MBC95	MBC94	MBC93	MBC92	MBC91	MBC90	MBC89	MBC88	MBC87	MBC86	MBC85	MBC84	MBC83	MBC82	MBC81	MBC80
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MBC79	MBC78	MBC77	MBC76	MBC75	MBC74	MBC73	MBC72	MBC71	MBC70	MBC69	MBC68	MBC67	MBC66	MBC65	MBC64
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
MBCn (n=64-95)	n-64	rh	**Message Buffer Status Changed n (n = 64-95)** An MBC flags is set whenever the Message Handler changes on of the status flags VFRA, VFRB, SEOA, SEOB, CEOA, CEOB, SVOA, SVOB, TCIA, TCIB, ESA, ESB, MLST, FTA, FTB in the Header Section (see “Message Buffer Status”) of the respective Message Buffer 64 to Message Buffer 95. The flags are reset when the Header Section of the Message Buffer is reconfigured or when it has been transferred to the Output Buffer.

23.4.72 Message Buffer Status Changed 4

This register reflect the state of the MBC flags of all configured Message Buffers. If the number of configured Message Buffers is less than 127, the remaining MBCn flags have no meaning. The register is reset when the Communication Controller leaves “CONFIG” state or enters “STARTUP” state.

MBSC4 Offset address: $034C_H$

Message Buffer Status Changed 4 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MBC127	MBC126	MBC125	MBC124	MBC123	MBC122	MBC121	MBC120	MBC119	MBC118	MBC117	MBC116	MBC115	MBC114	MBC113	MBC112
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MBC111	MBC110	MBC109	MBC108	MBC107	MBC106	MBC105	MBC104	MBC103	MBC102	MBC101	MBC100	MBC99	MBC98	MBC97	MBC96
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
MBCn (n=96-127)	n-96	rh	**Message Buffer Status Changed n (n = 96-127)** An MBC flags is set whenever the Message Handler changes on of the status flags VFRA, VFRB, SEOA, SEOB, CEOA, CEOB, SVOA, SVOB, TCIA, TCIB, ESA, ESB, MLST, FTA, FTB in the Header Section (see “Message Buffer Status”) of the respective Message Buffer 96 to Message Buffer 127. The flags are reset when the Header Section of the Message Buffer is reconfigured or when it has been transferred to the Output Buffer.

23.4.73 New Data Interrupt Control 1

This New Data Interrupt Control register controls the interrupt that becomes active (NDAT0SRC or NDAT1SRC) on a ND flag turning active of all configured Message Buffers 0 to Message Buffers 31.

NDIC1 Offset address: $03A8_H$

New Data Interrupt Control 1 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
NDIP31	NDIP30	NDIP29	NDIP28	NDIP27	NDIP26	NDIP25	NDIP24	NDIP23	NDIP22	NDIP21	NDIP20	NDIP19	NDIP18	NDIP17	NDIP16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NDIP15	NDIP14	NDIP13	NDIP12	NDIP11	NDIP10	NDIP9	NDIP8	NDIP7	NDIP6	NDIP5	NDIP4	NDIP3	NDIP2	NDIP1	NDIP0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
NDIPn (n=0-31)	n	rw	**New Data Interrupt Pointer n (n = 0-31)** NDIPn determines the interrupt (NDAT0SRC or NDAT1SRC) of the service request output that becomes active on a New Data Flag becoming active. 0_B NDAT0SRC selected for New Data Service Request 1_B NDAT1SRC selected for New Data Service Request

23.4.74 New Data Interrupt Control 2

This New Data Interrupt Control register controls the interrupt that becomes active (NDAT0SRC or NDAT1SRC) on a ND flag turning active of all configured Message Buffers 32 to Message Buffers 63.

NDIC2 Offset address: $03AC_H$

New Data Interrupt Control 2 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
NDIP 63	NDIP 62	NDIP 61	NDIP 60	NDIP 59	NDIP 58	NDIP 57	NDIP 56	NDIP 55	NDIP 54	NDIP 53	NDIP 52	NDIP 51	NDIP 50	NDIP 49	NDIP 48
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NDIP 47	NDIP 46	NDIP 45	NDIP 44	NDIP 43	NDIP 42	NDIP 41	NDIP 40	NDIP 39	NDIP 38	NDIP 37	NDIP 36	NDIP 35	NDIP 34	NDIP 33	NDIP 32
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
NDIPn (n=32-63)	n-32	rw	**New Data Interrupt Pointer n (n = 32-63)** NDIPn determines the interrupt (NDAT0SRC or NDAT1SRC) of the service request output that becomes active on a New Data Flag becoming active. 0_B NDAT0SRC selected for New Data Service Request 1_B NDAT1SRC selected for New Data Service Request

23.4.75 New Data Interrupt Control 3

This New Data Interrupt Control register controls the interrupt that becomes active (NDAT0SRC or NDAT1SRC) on a ND flag turning active of all configured Message Buffers 64 to Message Buffers 95.

NDIC3 Offset address: $03B0_H$

New Data Interrupt Control 3 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
NDIP 95	NDIP 94	NDIP 93	NDIP 92	NDIP 91	NDIP 90	NDIP 89	NDIP 88	NDIP 87	NDIP 86	NDIP 85	NDIP 84	NDIP 83	NDIP 82	NDIP 81	NDIP 80
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NDIP 79	NDIP 78	NDIP 77	NDIP 76	NDIP 75	NDIP 74	NDIP 73	NDIP 72	NDIP 71	NDIP 70	NDIP 69	NDIP 68	NDIP 67	NDIP 66	NDIP 65	NDIP 64
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
NDIPn (n=64-95)	n-64	rw	**New Data Interrupt Pointer n (n = 64-95)** NDIPn determines the interrupt (NDAT0SRC or NDAT1SRC) of the service request output that becomes active on a New Data Flag becoming active. 0_B NDAT0SRC selected for New Data Service Request 1_B NDAT1SRC selected for New Data Service Request

23.4.76 New Data Interrupt Control 4

This New Data Interrupt Control register controls the interrupt that becomes active (NDAT0SRC or NDAT1SRC) on a ND flag turning active of all configured Message Buffers 96 to Message Buffers 127.

NDIC4 Offset address: $03B4_H$

New Data Interrupt Control 4 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
NDIP 127	NDIP 126	NDIP 125	NDIP 124	NDIP 123	NDIP 122	NDIP 121	NDIP 120	NDIP 119	NDIP 118	NDIP 117	NDIP 116	NDIP 115	NDIP 114	NDIP 113	NDIP 112
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NDIP 111	NDIP 110	NDIP 109	NDIP 108	NDIP 107	NDIP 106	NDIP 105	NDIP 104	NDIP 103	NDIP 102	NDIP 101	NDIP 100	NDIP 99	NDIP 98	NDIP 97	NDIP 96
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
NDIPn (n=96-127)	n-96	rw	**New Data Interrupt Pointer n (n = 96-127)** NDIPn determines the interrupt (NDAT0SRC or NDAT1SRC) of the service request output that becomes active on a New Data Flag becoming active. 0_B NDAT0SRC selected for New Data Service Request 1_B NDAT1SRC selected for New Data Service Request

23.4.77 Message Buffer Status Changed Interrupt Control 1

This Message Buffer Status Change Interrupt Control register controls the interrupt that becomes active (MBSC0SRC or MBSC1SRC) on a MBC flag of all configured Message Buffer 0 to Message Buffer 31 turning active.

MSIC1 Offset address: $03B8_H$

Message Buffer Status Changed Interrupt Control 1 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MSIP 31	MSIP 30	MSIP 29	MSIP 28	MSIP 27	MSIP 26	MSIP 25	MSIP 24	MSIP 23	MSIP 22	MSIP 21	MSIP 20	MSIP 19	MSIP 18	MSIP 17	MSIP 16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MSIP 15	MSIP 14	MSIP 13	MSIP 12	MSIP 11	MSIP 10	MSIP 9	MSIP 8	MSIP 7	MSIP 6	MSIP 5	MSIP 4	MSIP 3	MSIP 2	MSIP 1	MSIP 0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
MSIPn (n=0-31)	n	rw	**Message Buffer Status Changed Interrupt Pointer n (n = 0-31)** MSIPn determines the interrupt (MBSC0SRC or MBSC1SRC) of the service request output that becomes active on a Message Buffer Status Changed Flag becoming active. 0_B MBSC0SRC selected for Message Buffer Status Changed Service Request 1_B MBSC1SRC selected for Message Buffer Status Changed Service Request

23.4.78 Message Buffer Status Changed Interrupt Control 2

This Message Buffer Status Change Interrupt Control register controls the interrupt that becomes active (MBSC0SRC or MBSC1SRC) on a MBC flag of all configured Message Buffer 32 to Message Buffer 63 turning active.

MSIC2 Offset address: $03BC_H$

Message Buffer Status Changed Interrupt Control 2 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MSIP 63	MSIP 62	MSIP 61	MSIP 60	MSIP 59	MSIP 58	MSIP 57	MSIP 56	MSIP 55	MSIP 54	MSIP 53	MSIP 52	MSIP 51	MSIP 50	MSIP 49	MSIP 48
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MSIP 47	MSIP 46	MSIP 45	MSIP 44	MSIP 43	MSIP 42	MSIP 41	MSIP 40	MSIP 39	MSIP 38	MSIP 37	MSIP 36	MSIP 35	MSIP 34	MSIP 33	MSIP 32
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
MSIPn (n=32-63)	n-32	rw	**Message Buffer Status Changed Interrupt Pointer n (n = 32-63)** MSIPn determines the interrupt (MBSC0SRC or MBSC1SRC) of the service request output that becomes active on a Message Buffer Status Changed Flag becoming active. 0_B MBSC0SRC selected for Message Buffer Status Changed Service Request 1_B MBSC1SRC selected for Message Buffer Status Changed Service Request

23.4.79 Message Buffer Status Changed Interrupt Control 3

This Message Buffer Status Change Interrupt Control register controls the interrupt that becomes active (MBSC0SRC or MBSC1SRC) on a MBC flag of all configured Message Buffer 64 to Message Buffer 95 turning active.

MSIC3 Offset address: $03C0_H$

Message Buffer Status Changed Interrupt Control 3 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MSIP 95	MSIP 94	MSIP 93	MSIP 92	MSIP 91	MSIP 90	MSIP 89	MSIP 88	MSIP 87	MSIP 86	MSIP 85	MSIP 84	MSIP 83	MSIP 82	MSIP 81	MSIP 80
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MSIP 79	MSIP 78	MSIP 77	MSIP 76	MSIP 75	MSIP 74	MSIP 73	MSIP 72	MSIP 71	MSIP 70	MSIP 69	MSIP 68	MSIP 67	MSIP 66	MSIP 65	MSIP 64
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
MSIPn (n=64-95)	n-64	rw	**Message Buffer Status Changed Interrupt Pointer n (n = 64-95)** MSIPn determines the interrupt (MBSC0SRC or MBSC1SRC) of the service request output that becomes active on a Message Buffer Status Changed Flag becoming active. 0_B MBSC0SRC selected for Message Buffer Status Changed Service Request 1_B MBSC1SRC selected for Message Buffer Status Changed Service Request

23.4.80 Message Buffer Status Changed Interrupt Control 4

This Message Buffer Status Change Interrupt Control register controls the interrupt that becomes active (MBSC0SRC or MBSC1SRC) on a MBC flag of all configured Message Buffer 96 to Message Buffer 127 turning active.

MSIC4 Offset address: $03C4_H$

Message Buffer Status Changed Interrupt Control 4 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MSIP 127	MSIP 126	MSIP 125	MSIP 124	MSIP 123	MSIP 122	MSIP 121	MSIP 120	MSIP 119	MSIP 118	MSIP 117	MSIP 116	MSIP 115	MSIP 114	MSIP 113	MSIP 112
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MSIP 111	MSIP 110	MSIP 109	MSIP 108	MSIP 107	MSIP 106	MSIP 105	MSIP 104	MSIP 103	MSIP 102	MSIP 101	MSIP 100	MSIP 99	MSIP 98	MSIP 97	MSIP 96
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
MSIPn (n=96-127)	n-96	rw	**Message Buffer Status Changed Interrupt Pointer n (n = 96-127)** MSIPn determines the interrupt (MBSC0SRC or MBSC1SRC) of the service request output that becomes active on a Message Buffer Status Changed Flag becoming active. 0_B MBSC0SRC selected for Message Buffer Status Changed Service Request 1_B MBSC1SRC selected for Message Buffer Status Changed Service Request

23.4.81 Core Release Register

This register contains bit fields about the ERAY module identification. It is read only.

CREL Offset address: $03F0_H$

Core Release Register Kernel Reset value: $1027\ 1031_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
REL				STEP				SUBSTEP				YEAR			
r				r				r				r			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MON								DAY							
r								r							

Field	Bits	Type	Description
DAY	7:0	r	**Design Time Stamp, Day** Two digits, BCD-coded.
MON	15:8	r	**Design Time Stamp, Month** Two digits, BCD-coded.
YEAR	19:16	r	**Design Time Stamp, Year** One digit, BCD-coded.
SUBSTEP	23:20	r	**Sub-Step of Core Release** One digits, BCD-coded. 0_H Alpha, pre-Beta, pre-Beta-update, pre-Beta2, pre-Beta2-update, Beta, Beta2, Revision 1.0.0 1_H Beta_ct, Beta-ct-fix1, Revision 1.0.1 2_H Revision1.0RC1, Beta-ct-fix2, REVISION 1.0RC1

(table continues...)

(continued)

Field	Bits	Type	Description
STEP	27:24	r	**Step of Core Release** One digits, BCD-coded. 0_H Revision 1.0.0 1_H Alpha 2_H pre-Beta 3_H pre-Beta-update 4_H pre-Beta2 5_H pre-Beta2-update 6_H Beta 7_H Beta2
REL	31:28	r	**Core Release** One digit, BCD-coded. 0_H alpha…beta2ct 1_H Revision 1.0

Table 1098 Coding of releases

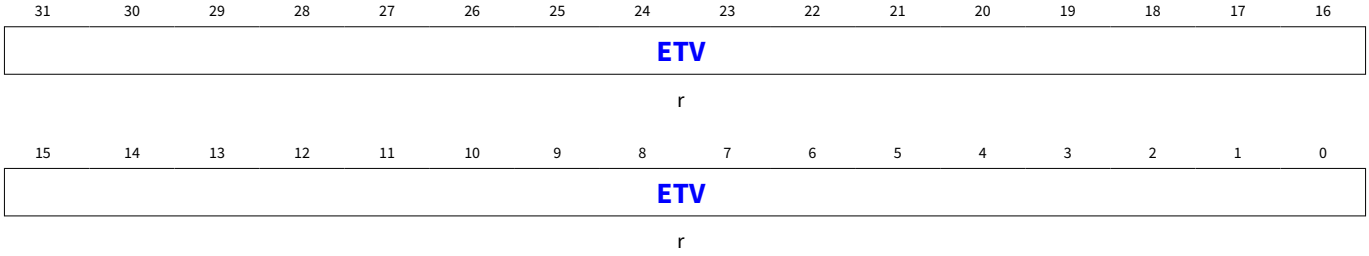
Release	Step	Sub-Step	Name	Release Date
0	1	0	Alpha	
0	2	0	pre-Beta	
0	3	0	pre-Beta-update	
0	4	0	pre-Beta2	
0	5	0	pre-Beta2-update	
0	6	0	Beta	
0	6	1	Beta-ct-fix1	14.10.2005
0	6	2	Beta-ct-fix2	14.12.2005
0	7	0	Beta2	03.02.2006
0	7	1	Beta2ct	24.03.2006
0	7	2	Revision 1.0RC1	07.04.2006
1	0	0	Release 1.0.0	19.05.2006
1	0	1	Release 1.0.1	2006
1	0	2	Release 1.0.2	31.10.2007

23.4.82 Endian Register

This register may be used to check, if the data of the ERAY is handled by a host with the correct endian format. It is read only.

ENDN Offset address: $03F4_H$

Endian Register Kernel Reset value: $8765\ 4321_H$



Field	Bits	Type	Description
ETV	31:0	r	**Endianness Test Value** The endianness test value.

23.4.83 Write Data Section n

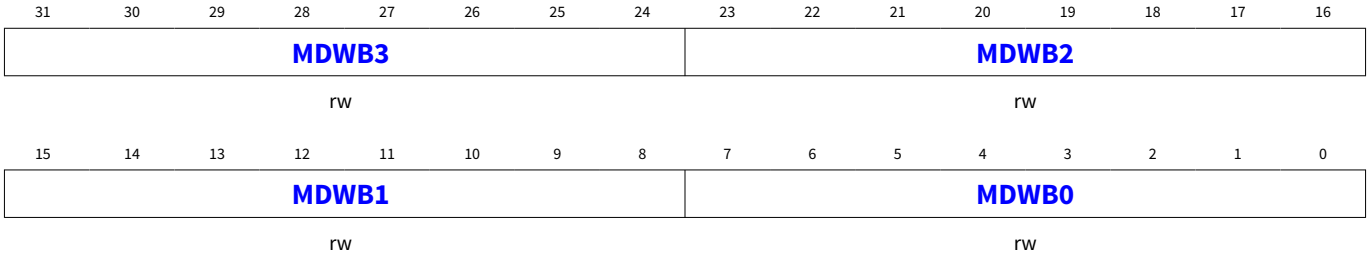
The Write Data Section (WRDSn, n = 01-64) holds the data words to be transferred to the Data Section of the addressed Message Buffer. The data words (DW_n) are written to the Message RAM in transmission order from DW1 (byte0, byte1) to DW_{PL} (PL = number of data words as defined by the payload length configured by WRHS2.PLC).

***Note**: 16-bit Word 127 is located on WRDS64.MDW. In this case WRDS64.MDW is unused (no valid data). The Input Buffer RAMs are initialized to zero by CHI command CLEAR_RAMS.*

When writing to the WRDSn (n = 01-64), each 32-bit word has to be filled up by one 32-bit access OR two consecutive 16-bit accesses OR four consecutive 8-bit accesses before the transfer from the Input Buffer to the Message RAM is started by writing the number of the target Message Buffer in the Message RAM to the Input Buffer Command Request register. If a 32-bit word of the Input Buffer has been filled with less then two consecutive 16-bit accesses OR four consecutive 8-bit accesses (less then 32-bit), random data is transferred into the Input buffer for every not written 16-bit or 8-bit of a 32-bit word.

WRDSn (n=01-64) Offset address: 0400_H+(n-1)*4

Write Data Section n Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
MDWB0	7:0	rw	**32-Bit Word n, Byte 0**
MDWB1	15:8	rw	**32-Bit Word n, Byte 1**
MDWB2	23:16	rw	**32-Bit Word n, Byte 2**
MDWB3	31:24	rw	**32-Bit Word n, Byte 3**

23.4.84 Write Header Section 1

WRHS1

Write Header Section 1

Offset address: 0500_H

Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		MBI	TXM	PPIT	CFG	CHB	CHA	0	CYC						
r		rw	rw	rw	rw	rw	rw	r	rw						

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0					FID										
r					rw										

Field	Bits	Type	Description
FID	10:0	rw	**Frame ID** Frame ID of the selected Message Buffer. The Frame ID defines the slot number for transmission / reception of the respective message. Message Buffers with Frame ID = 0 are considered as not valid.
CYC	22:16	rw	**Cycle Code** The 7-bit cycle code determines the cycle set used for cycle counter filtering. For details about the configuration of the cycle code see Section "Cycle Counter Filtering".
CHA	24	rw	**Channel Filter Control A** The channel filtering field A associated with the buffer serves of channel A as a filter for receive buffers, and as a control field for transmit buffers
CHB	25	rw	**Channel Filter Control B** The channel filtering field B associated with the buffer serves of channel B as a filter for receive buffers, and as a control field for transmit buffers
CFG	26	rw	**Message Buffer Direction Configuration Bit** This bit is used to configure the corresponding buffer as a transmit buffer or as a receive buffer. For Message Buffers belonging to the receive FIFO the bit is not evaluated. 0_B The corresponding buffer is configured as Receive Buffer 1_B The corresponding buffer is configured as Transmit Buffer
PPIT	27	rw	**Payload Preamble Indicator Transmit** This bit is used to control the state of the Payload Preamble Indicator in transmit Frames. If the bit is set in a static Message Buffer, the respective Message Buffer holds Network Management information. If the bit is set in a dynamic Message Buffer, the first two byte of the Payload Segment may be used for message ID filtering by the receiver. Message ID filtering of received FlexRay™ Frames is not supported by the ERAY module, but can be done by the Host. 0_B Payload Preamble Indicator not set 1_B Payload Preamble Indicator set

(table continues...)

(continued)

Field	Bits	Type	Description
TXM	28	rw	**Transmission Mode** This bit is used to select the transmission mode (see “Transmit Buffers”). 0_B Continuous mode 1_B Single-shot mode
MBI	29	rw	**Message Buffer Service Request** This bit enables the receive / transmit service request for the corresponding Message Buffer. After a dedicated receive buffer has been updated by the Message Handler, flag SIR.RXI and/or SIR.MBSI in the Status Service Request register are set. After a transmission has completed flag SIR.TXI is set. 0_B The corresponding Message Buffer service request is disabled 1_B The corresponding Message Buffer service request is enabled
0	15:11, 23, 31:30	r	**Reserved** Read as 0; should be written with 0.

***Note**: The Input Buffer RAMs are initialized to zero by CHI command CLEAR_RAMS. Note that only the currently active IBF bank is cleared. To clear the 2nd bank as well, CUST1.IBF1PAG and CUST1.IBF2PAG need to be set and command CLEAR_RAMS need to be issued again. This is required in particular after an application reset. If the 2nd bank of IBF is left unused, this procedure is not required.*

Table 1099 Channel Filter Control Bits

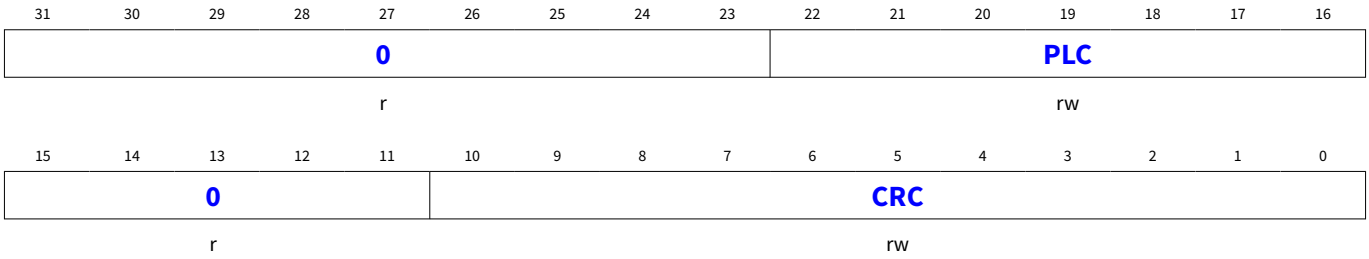
CHA	CHB	Transmit Buffertransmit Frame on	Receive Bufferstore Frame received from
1	1	Both Channels (static segment only)	Channel A or B (store first semantically valid Frame, static segment only)
1	0	Channel A	Channel A
0	1	Channel B	Channel B
0	0	No Transmission	Ignore Frame

***Note**: If a Message Buffer is configured for the dynamic segment and both bits of the channel filtering field are set to 1, no Frames are transmitted respectively received Frames are ignored (same function as CHA = CHB = 0)*

23.4.85 Write Header Section 2

WRHS2 Offset address: 0504_H

Write Header Section 2 Kernel Reset value: $0000\ 0000_H$

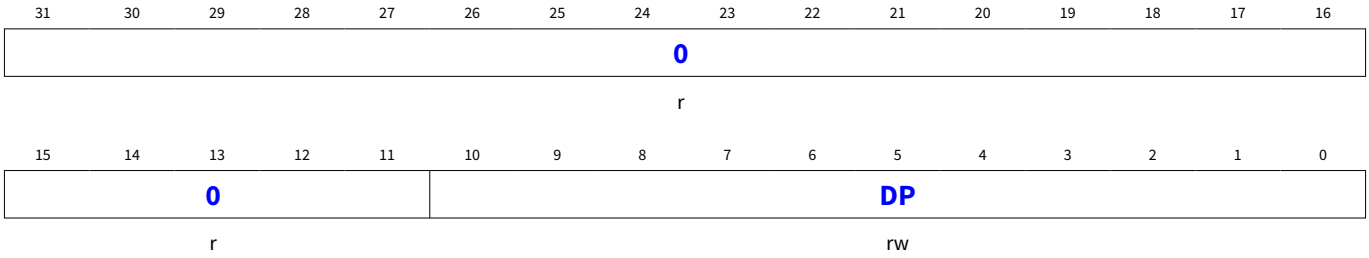


Field	Bits	Type	Description
CRC	10:0	rw	**Header CRC (vRF!Header!HeaderCRC)** Receive Buffer: Configuration not required Transmit Buffer: Header CRC calculated and configured by the Host. For calculation of the Header CRC the payload length of the Frame send on the bus has to be considered. In static segment the payload length of all Frames is configured by MHDC.SFDL.
PLC	22:16	rw	**Payload Length Configured** Length of Data Section (number of 2-byte words) as configured by the Host. During static segment the static Frame payload length as configured by MHDC.SFDL in the MHD Configuration Register defines the payload length for all static Frames. If the payload length configured by PLC is shorter than this value, padding byte are inserted to ensure that Frames have proper physical length. The padding pattern is logical zero.
0	15:11, 31:23	r	**Reserved** Read as 0; should be written with 0.

23.4.86 Write Header Section 3

WRHS3 Offset address: 0508_H

Write Header Section 3 Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
DP	10:0	rw	**Data Pointer** Pointer to the first 32-bit word of the Data Section of the addressed Message Buffer in the Message RAM.
0	31:11	r	**Reserved** Read as 0; should be written with 0.

23.4.87 Input Buffer Command Mask

Configures how the Message Buffer in the Message RAM selected by the Input Buffer Command Request register IBCR is updated. If IBF Host and IBF Shadow are swapped, also masked bits IBCM.LHSH, IBCM.LDSH, and IBCM.STXRH are swapped with bits IBCM.LHSS, IBCM.LDSS, and IBCM.STXRS to keep them attached to the respective Input Buffer transfer.

IBCM Offset address: 0510_H

Input Buffer Command Mask Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0													STXRS	LDSS	LHSS
r													rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0													STXRH	LDSH	LHSH
r													rwh	rwh	rwh

Field	Bits	Type	Description
LHSH	0	rwh	**Load Header Section Host** 0_B Header Section is not updated 1_B Header Section selected for transfer from Input Buffer to the Message RAM
LDSH	1	rwh	**Load Data Section Host** 0_B Data Section is not updated 1_B Data Section selected for transfer from Input Buffer to the Message RAM
STXRH	2	rwh	**Set Transmission Request Host** If this bit is set to 1, the Transmission Request flag TXRQ1.TXRn (n = 0-31) to TXRQ4.TXRn (n = 96-127) for the selected Message Buffer is set in the Transmission Request Registers to release the Message Buffer for transmission. In single-shot mode the flag is cleared by the Communication Controller after transmission has completed. TXRQ1.TXRn (n = 0-31) to TXRQ4.TXRn (n = 96-127) are evaluated for transmit buffer only. 0_B Reset Transmission Request flag 1_B Set Transmission Request flag, transmit buffer released for transmission
LHSS	16	rh	**Load Header Section Shadow** 0_B Header Section is not updated 1_B Header Section selected for transfer from Input Buffer to the Message RAM (transfer is ongoing or finalized)
LDSS	17	rh	**Load Data Section Shadow** 0_B Data Section is not updated 1_B Data Section selected for transfer from Input Buffer to the Message RAM (transfer is ongoing or finalized)

(table continues...)

(continued)

Field	Bits	Type	Description
STXRS	18	rh	**Transmission Request Shadow** If this bit is set to 1, the Transmission Request flag TXRQ1.TXRn (n = 0-31) to TXRQ4.TXRn (n = 96-127) for the selected Message Buffer is set in the Transmission Request Registers to release the Message Buffer for transmission. In single-shot mode the flag is cleared by the Communication Controller after transmission has completed. TXRQ1.TXRn (n = 0-31) to TXRQ4.TXRn (n = 96-127) are evaluated for transmit buffer only. 0_B Reset Transmission Request flag 1_B Set Transmission Request flag, transmit buffer released for transmission (operation is ongoing or finalized)
0	15:3, 31:19	r	**Reserved** Read as 0; should be written with 0.

23.4.88 Input Buffer Command Request

When the Host writes the number of the target Message Buffer in the Message RAM to IBRH in the Input Buffer Command Request register, IBF Host and IBF Shadow are swapped. In addition the Message Buffer numbers stored under IBRH and IBRS are also swapped (see also “Data Transfer from Input Buffer to Message RAM”).

With this write operation the IBSYS bit in the Input Buffer Command Request register is set to 1. The Message Handler then starts to transfer the contents of IBF Shadow to the Message Buffer in the Message RAM selected by IBRS.

While the Message Handler transfers the data from IBF Shadow to the target Message Buffer in the Message RAM, the Host may write the next message into the IBF Host. After the transfer between IBF Shadow and the Message RAM has completed, the IBSYS bit is set back to 0 and the next transfer to the Message RAM may be started by the Host by writing the respective target Message Buffer number to IBRH.

If a write access to IBRH occurs while IBSYS is 1, IBSYH is set to 1. After completion of the ongoing data transfer from IBF Shadow to the Message RAM, IBF Host and IBF Shadow are swapped, IBSYH is reset to 0. IBSYS remains set to 1, and the next transfer to the Message RAM is started. In addition the Message Buffer numbers stored under IBRH and IBRS are also swapped. Any write access to an Input Buffer register while both IBSYS and IBSYH are set will cause the error flag EIR.IIBA to be set. In this case the Input Buffer will not be changed.

IBCR Offset address: 0514_H

Input Buffer Command Request Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
IBSYS	0								IBRS						
rh	r								rh						

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IBSYH	0								IBRH						
rh	r								rwh						

Field	Bits	Type	Description
IBRH	6:0	rwh	**Input Buffer Request Host** Selects the target Message Buffer in the Message RAM for data transfer from Input Buffer. Valid values are 00_H to $7F_H$ (0…127).

(table continues...)

(continued)

Field	Bits	Type	Description
IBSYH	15	rh	**Input Buffer Busy Host** Set to 1 by writing IBRH while IBSYS is still 1. After the ongoing transfer between IBF Shadow and the Message RAM has completed, the IBSYH is set back to 0. 0_B No request pending 1_B Request while transfer between IBF Shadow and Message RAM in progress
IBRS	22:16	rh	**Input Buffer Request Shadow** Number of the target Message Buffer actually updated/lately updated. Valid values are 00_H to $7F_H$ (0…127).
IBSYS	31	rh	**Input Buffer Busy Shadow** Set to 1 after writing IBRH. When the transfer between IBF Shadow and the Message RAM has completed, IBSYS is set back to 0. 0_B Transfer between IBF Shadow and Message RAM completed 1_B Transfer between IBF Shadow and Message RAM in progress
0	14:7, 30:23	r	**Reserved** Read as 0; should be written with 0.

23.4.89 Read Data Section n

The Read Data Section n (RDDSn, n = 01-64) holds the data words read from the Data Section of the addressed Message Buffer. The data words are read from the Message RAM in reception order from DW1 (byte0, byte1) to DW_{PL} (PL = number of data words as defined by the Payload Length).

***Note**: DW127 is located on RDDS64.MDW. In this case RDDS64.MDW is unused (no valid data). The Output Buffer RAMs are initialized to zero by CHI command CLEAR_RAMS.*

RDDSn (n=01-64) Offset address: 0600_H+(n-1)*4

Read Data Section n Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MDRB3								MDRB2							
rh								rh							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MDRB1								MDRB0							
rh								rh							

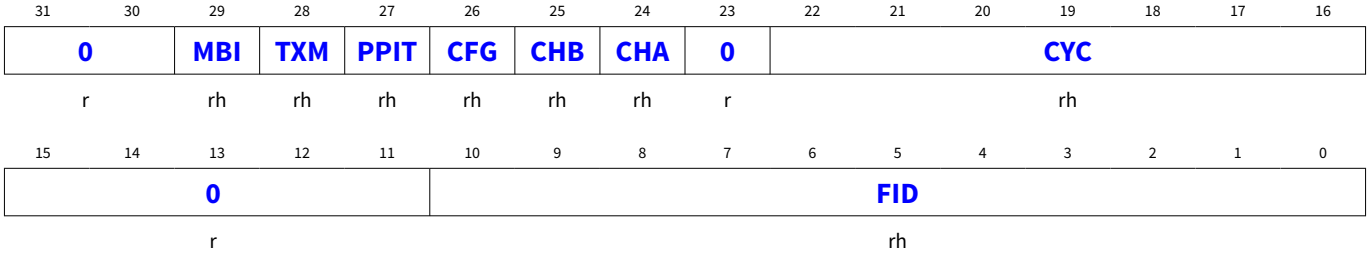
Field	Bits	Type	Description
MDRB0	7:0	rh	**32-Bit Word n, Byte 0**
MDRB1	15:8	rh	**32-Bit Word n, Byte 1**
MDRB2	23:16	rh	**32-Bit Word n, Byte 2**
MDRB3	31:24	rh	**32-Bit Word n, Byte 3**

23.4.90 Read Header Section 1

Values as configured by the Host using WRHS1 Register:

RDHS1 Offset address: 0700_H

Read Header Section 1 Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
FID	10:0	rh	**Frame ID**
CYC	22:16	rh	**Cycle Code**
CHA	24	rh	**Channel Filter Control A**
CHB	25	rh	**Channel Filter Control B**
CFG	26	rh	**Message Buffer Direction Configuration Bit**
PPIT	27	rh	**Payload Preamble Indicator Transmit**
TXM	28	rh	**Transmission Mode**
MBI	29	rh	**Message Buffer Service Request**
0	15:11, 23, 31:30	r	**Reserved** Read as 0; should be written with 0.

***Note**: In case that the Message Buffer read from the Message RAM belongs to the receive FIFO, FID holds the received Frame ID, while CYC, CHA, CHB, CFG, PPIT, TXM, and MBI are reset to zero.*

Table 1100 Channel Filter Control Bits

CHA	CHB	Transmit Buffertransmit Frame on	Receive Bufferstore Frame received from
1[1)]	1[1)]	Both Channels (static segment only)	Channel A or B (store first semantically valid Frame, static segment only)
1	0	Channel A	Channel A
0	1	Channel B	Channel B
0	0	No Transmission	Ignore Frame

***Note**: If a Message Buffer is configured for the dynamic segment and both bits of the channel filtering field are set to 1, no Frames are transmitted respecitvely received Frames are ignored (same function as CHA = CHB = 0)*

23.4.91 Read Header Section 2

***Note**: The Message RAM is organized in 4-byte words. When received data is stored into a Message Buffer's Data Section, the number of 2-byte data words written into the Message Buffer is PLC rounded to the next even value. PLC should be configured identical for all Message Buffers belonging to the receive FIFO. Header 2 is updated from Data Frames only.*

RDHS2 Offset address: 0704_H

Read Header Section 2 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	PLR							0	PLC						
r	rh							r	rh						

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0					CRC										
r					rh										

Field	Bits	Type	Description
CRC	10:0	rh	**Header CRC (vRF!Header!HeaderCRC)** Receive Buffer: Configuration not required. Header CRC updated from receive Data Frames. Transmit Buffer: Header CRC calculated and configured by the Host
PLC	22:16	rh	**Payload Length Configured** Length of Data Section (number of 2-byte words) as configured by the Host.
PLR	30:24	rh	**Payload Length Received (vRF!Header!Length)** Payload length value updated from received Data Frame (exception: if Message Buffer belongs to the receive FIFO, PLR is also updated from received NULL Frames). When a message is stored into a Message Buffer the following behavior with respect to payload length received and payload length configured is implemented: • **PLR > PLC:** The payload data stored in the Message Buffer is truncated to the payload length configured for even PLC or else truncated to PLC + 1. • **PLR ≤ PLC:** The received payload data is stored into the Message Buffers Data Section. The remaining data bytes of the Data Section as configured by PLC are filled with undefined data. • **PLR = 0:** The Message Buffer's Data Section is filled with undefined data. • **PLC = 0:** Message Buffer has no Data Section configured. No data is stored into the Message Buffer's Data Section.
0	15:11, 23, 31	r	**Reserved** Read as 0; should be written with 0.

23.4.92 Read Header Section 3

***Note**: Header 3 is updated from Data Frames only.*

RDHS3 Offset address: 0708_H

Read Header Section 3 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		RES	PPI	NFI	SYN	SFI	RCI	0		RCC					
r		rh	rh	rh	rh	rh	rh	r		rh					

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0					DP										
r					rh										

Field	Bits	Type	Description
DP	10:0	rh	**Data Pointer** Pointer to the first 32-bit word of the Data Section of the addressed Message Buffer in the Message RAM.
RCC	21:16	rh	**Receive Cycle Count (vRF!Header!CycleCount)** Cycle counter value updated from received Data Frame.
RCI	24	rh	**Received on Channel Indicator (vSS!Channel)** Indicates the channel from which the received Data Frame was taken to update the respective receive buffer. 0_B Frame received on channel B 1_B Frame received on channel A
SFI	25	rh	**Startup Frame Indicator (vRF!Header!SuFIndicator)** A Startup Frame is marked by the Startup Frame indicator. 0_B The received Frame is not a startup Frame 1_B The received Frame is a startup Frame
SYN	26	rh	**SYNC Frame Indicator (vRF!Header!SyFIndicator)** A SYNC Frame is marked by the SYNC Frame indicator. 0_B The received Frame is not a SYNC Frame 1_B The received Frame is a SYNC Frame
NFI	27	rh	**NULL Frame Indicator (vRF!Header!NFIndicator)** Is set to 1 after storage of the first received Data Frame. 0_B Up to now no Data Frame has been stored into the respective Message Buffer 1_B At least one Data Frame has been stored into the respective Message Buffer

(table continues...)

(continued)

Field	Bits	Type	Description
PPI	28	rh	**Payload Preamble Indicator (vRF!Header!PPIndicator)** The payload preamble indicator defines whether a Network Management vector or message ID is contained within the Payload Segment of the received Frame. 0_B The Payload Segment of the received Frame does not contain a Network Management vector nor a message ID 1_B Static segment: Network Management vector in the first part of the payload Dynamic segment: Message ID in the first part of the payload
RES	29	rh	**Reserved (vRF!Header!Reserved)** Reflects the state of the received reserved bit. The reserved bit is transmitted as 0.
0	15:11, 23:22, 31:30	r	**Reserved** Read as 0; should be written with 0.

23.4.93 Message Buffer Status

The Message Buffer status is updated by the Communication Controller with respect to the assigned channel(s) latest at the end of the slot following the slot assigned to the Message Buffer. The flags are updated only when the Communication Controller is in “NORMAL_ACTIVE” or “NORMAL_PASSIVE” state. If only one channel (A or B) is assigned to a Message Buffer, the channel-specific status flags of the other channel are written to zero. If both channels are assigned to a Message Buffer, the channel-specific status flags of both channels are updated. The Message Buffer status is updated only when the slot counter reached the configured Frame ID and when the cycle counter filter matched. When the Host updates a Message Buffer through Input Buffer, all MBS flags are reset to zero independent of which IBCM bits are set or not. For details about receive / transmit filtering see “Filtering and Masking”, “Transmit Process”, and “Receive Process”.

Whenever the Message Handler changes one of the flags VFRA, VFRB, SEOA, SEOB, CEOA, CEOB, SVOA, SVOB, TCIA, TCIB, ESA, ESB, MLST, FTA, FTB the respective Message Buffer’s MBC flag in registers MBSC1 to MBSC4 is set

MBS Offset address: $070C_H$

Message Buffer Status Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		RESERVED	PPIS	NFIS	SYNS	SFIS	RCIS	0		CCS					
r		rh	rh	rh	rh	rh	rh	r		rh					

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FTB	FTA	0	MLST	ESB	ESA	TCIB	TCIA	SVOB	SVOA	CEOB	CEOA	SEOB	SEOA	VFRB	VFRA
rh	rh	r	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
VFRA	0	rh	**Valid Frame Received on Channel A (vSS!ValidFrameA)** A valid Frame indication is set if a valid Frame was received on channel A. 0_B No valid Frame received on channel A 1_B Valid Frame received on channel A
VFRB	1	rh	**Valid Frame Received on Channel B (vSS!ValidFrameB)** A valid Frame indication is set if a valid Frame was received on channel B. 0_B No valid Frame received on channel B 1_B Valid Frame received on channel B
SEOA	2	rh	**Syntax Error Observed on Channel A (vSS!SyntaxErrorA)** A syntax error was observed in the assigned slot on channel A. 0_B No syntax error observed on channel A 1_B Syntax error observed on channel A
SEOB	3	rh	**Syntax Error Observed on Channel B (vSS!SyntaxErrorB)** A syntax error was observed in the assigned slot on channel B. 0_B No syntax error observed on channel B 1_B Syntax error observed on channel B
CEOA	4	rh	**Content Error Observed on Channel A (vSS!ContentErrorA)** A content error was observed in the assigned slot on channel A. 0_B No content error observed on channel A 1_B Content error observed on channel A
CEOB	5	rh	**Content Error Observed on Channel B (vSS!ContentErrorB)** A content error was observed in the assigned slot on channel B. 0_B No content error observed on channel B 1_B Content error observed on channel B
SVOA	6	rh	**Slot Boundary Violation Observed on Channel A (vSS!BViolationA)** A slot boundary violation (channel active at the start or at the end of the assigned slot) was observed on channel A. 0_B No slot boundary violation observed on channel A 1_B Slot boundary violation observed on channel A
SVOB	7	rh	**Slot Boundary Violation Observed on Channel B (vSS!BViolationB)** A slot boundary violation (channel active at the start or at the end of the assigned slot) was observed on channel B. 0_B No slot boundary violation observed on channel B 1_B Slot boundary violation observed on channel B
TCIA	8	rh	**Transmission Conflict Indication Channel A (vSS!TxConflictA)** A transmission conflict indication is set if a transmission conflict has occurred on channel A. 0_B No transmission conflict occurred on channel A 1_B Transmission conflict occurred on channel A

(table continues...)

(continued)

Field	Bits	Type	Description
TCIB	9	rh	**Transmission Conflict Indication Channel B (vSS!TxConflictB)** A transmission conflict indication is set if a transmission conflict has occurred on channel B. 0_B No transmission conflict occurred on channel B 1_B Transmission conflict occurred on channel B
ESA	10	rh	**Empty Slot Channel A** In an empty slot there is no activity detected on the bus. The condition is checked in static and dynamic slots. 0_B Bus activity detected in the assigned slot on channel A 1_B No bus activity detected in the assigned slot on channel A
ESB	11	rh	**Empty Slot Channel B** In an empty slot there is no activity detected on the bus. The condition is checked in static and dynamic slots. 0_B Bus activity detected in the assigned slot on channel B 1_B No bus activity detected in the assigned slot on channel B
MLST	12	rh	**Message Lost** The flag is set in case the Host did not read the message before the Message Buffer was updated from a received Data Frame. Not affected by reception of NULL Frames except for Message Buffers belonging to the receive FIFO. The flag is reset by a Host write to the Message Buffer through IBF or when a new message is stored into the Message Buffer after the Message Buffers ND flag was reset by reading out the Message Buffer through OBF. 0_B No message lost 1_B Unprocessed message was overwritten
FTA	14	rh	**Frame Transmitted on Channel A** Indicates that this node has transmitted a Data Frame in the assigned slot on channel A. ***Note**: The FlexRay™ protocol specification requires that FTA can only be reset by the Host. Therefore the Cycle Count Status CCS for these bits is only valid for the cycle where the bits are set to 1* 0_B No transmission transmitted on channel A 1_B Data Frame transmitted on channel A in cycle defined by CCS bit field

(table continues...)

(continued)

Field	Bits	Type	Description
FTB	15	rh	**Frame Transmitted on Channel B** Indicates that this node has transmitted a Data Frame in the assigned slot on channel B. ***Note:*** *The FlexRay™ protocol specification requires that FTB can only be reset by the Host. Therefore the Cycle Count Status CCS for these bits is only valid for the cycle where the bits are set to 1* 0_B No transmission transmitted on channel B 1_B Data Frame transmitted on channel B in cycle defined by CCS bit field
CCS	21:16	rh	**Cycle Count Status** Cycle Count when status (MBS register) has been updated.
RCIS	24	rh	**Received on Channel Indicator Status (vSS!Channel)** Indicates the channel on which the Frame was received. ***Note:*** *For receive buffers (CFG = 0) the RCIS is updated from both valid data and NULL Frames. If no valid Frame was received, the previous value is maintained. For transmit buffers the flags have no meaning and should be ignored.* 0_B Frame received on channel B 1_B Frame received on channel A
SFIS	25	rh	**Startup Frame Indicator Status (vRF!Header!SuFIndicator)** A Startup Frame is marked by the Startup Frame indicator. ***Note:*** *For receive buffers (CFG = 0) the SFIS is updated from both valid data and NULL Frames. If no valid Frame was received, the previous value is maintained. For transmit buffers the flags have no meaning and should be ignored.* 0_B No Startup Frame received 1_B The received Frame is a Startup Frame
SYNS	26	rh	**SYNC Frame Indicator Status (vRF!Header!SyFIndicator)** A SYNC Frame is marked by the SYNC Frame indicator. ***Note:*** *For receive buffers (CFG = 0) the SYNS is updated from both valid data and NULL Frames. If no valid Frame was received, the previous value is maintained. For transmit buffers the flags have no meaning and should be ignored.* 0_B No SYNC Frame received 1_B The received Frame is a SYNC Frame

(table continues...)

(continued)

Field	Bits	Type	Description
NFIS	27	rh	**NULL Frame Indicator Status (vRF!Header!NFIndicator)** If reset to 0, the Payload Segment of the received Frame contains no usable data. ***Note**: For receive buffers (CFG = 0) the NFIS is updated from both valid data and NULL Frames. If no valid Frame was received, the previous value is maintained. For transmit buffers the flags have no meaning and should be ignored.* 0_B Received Frame is a NULL Frame 1_B Received Frame is not a NULL Frame
PPIS	28	rh	**Payload Preamble Indictor Status (vRF!Header!PPIndicator)** The payload preamble indicator defines whether a Network Management vector or message ID is contained within the Payload Segment of the received Frame. ***Note**: For receive buffers (CFG = 0) the PPIS is updated from both valid data and NULL Frames. If no valid Frame was received, the previous value is maintained. For transmit buffers the flags have no meaning and should be ignored.* 0_B The Payload Segment of the received Frame does not contain a Network Management vector or a message ID 1_B Static Segment: Network Management vector at the beginning of the payload Dynamic Segment: Message ID at the beginning of the payload
RESERVED	29	rh	**Reserved (vRF!Header!Reserved) - RESS** Reflects the state of the received reserved bit. The reserved bit is transmitted as 0. ***Note**: For receive buffers (CFG = 0) the RESS is updated from both valid data and NULL Frames. If no valid Frame was received, the previous value is maintained. For transmit buffers the flags have no meaning and should be ignored.*
0	13, 23:22, 31:30	r	**Reserved** Read as 0; should be written with 0.

23.4.94 Output Buffer Command Mask

Configures how the Output Buffer is updated from the Message Buffer in the Message RAM selected by the Output Buffer Command Request register. If OBF Host and OBF Shadow are swapped, also mask bits OBCM.RDSH and OBCM.RHSH are swapped with bits OBCM.RDSS and OBCM.RHSS to keep them attached to the respective Output Buffer transfer.

***Note**: After the transfer of the Header Section from the Message RAM to OBF Shadow has completed, the Message Buffer status changed flag MBCn (n = 0-31) to MBCn (n = 96-127) of the selected Message Buffer in the Message Buffer Changed MBSC1 to MBSC4 registers is cleared. After the transfer of the Data Section from the Message RAM to OBF Shadow has completed, the New Data flag NDn (n = 0-31) to NDn (n = 96-127) of the selected Message Buffer in the New Data NDAT1 to NDAT4 registers is cleared.*

OBCM Offset address: 0710_H

Output Buffer Command Mask Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0														RDSH	RHSH
r														rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0														RDSS	RHSS
r														rwh	rwh

Field	Bits	Type	Description
RHSS	0	rwh	**Read Header Section Shadow** 0_B Header Section is not read 1_B Header Section selected for transfer from Message RAM to Output Buffer
RDSS	1	rwh	**Read Data Section Shadow** 0_B Data Section is not read 1_B Data Section selected for transfer from Message RAM to Output Buffer
RHSH	16	rh	**Read Header Section Host** 0_B Header Section is not read 1_B Header Section selected for transfer from Message RAM to Output Buffer
RDSH	17	rh	**Read Data Section Host** 0_B Data Section is not read 1_B Data Section selected for transfer from Message RAM to Output Buffer
0	15:2, 31:18	r	**Reserved** Read as 0; should be written with 0.

23.4.95 Output Buffer Command Request

The Message Buffer selected by OBCR.OBRS is transferred from the Message RAM to the Output Buffer as soon as the Host has set OBCR.REQ. Bit OBCR.REQ can only be set while OBCR.OBSYS is 0 (see also “Data Transfer from Message RAM to Output Buffer”).

After setting OBCR.REQ, OBCR.OBSYS is automatically set, and the transfer of the Message Buffer selected by OBCR.OBRS from the Message RAM to Output Buffer Shadow is started. When the transfer between the Message RAM and OBF Shadow has completed, this is signalled by clearing OBCR.OBSYS. By setting OBCR.VIEW while OBCR.OBSYS is 0, OBF Host and OBF Shadow are swapped. When Output Buffer Host and Output Buffer Shadow are swapped, also mask bits OBCM.RDSH and OBCM.RHSH are swapped with bits OBCM.RDSS and OBCM.RHSS to keep them attached to the respective Output Buffer transfer. Now the Host can read the transferred Message Buffer from OBF Host. In parallel the Message Handler may transfer the next message from the Message RAM to OBF Shadow if OBCR.VIEW and OBCR.REQ are set at the same time.

Any write access to an Output Buffer register while OBCR.OBSYS is set will cause the error flag EIR.IOBA to be set. In this case the Output Buffer will not be changed.

OBCR Offset address: 0714_H

Output Buffer Command Request Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0									OBRH						
r									rh						

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OBSYS	0					REQ	VIEW	0	OBRS						
rh	r					rw	rw	r	rw						

Field	Bits	Type	Description
OBRS	6:0	rw	**Output Buffer Request Shadow** Number of source Message Buffer to be transferred from the Message RAM to OBF Shadow. Valid values are 00_H to $7F_H$ (0 to 127). If the number of the first Message Buffer of the receive FIFO is written to this register the Message Handler transfers the Message Buffer addressed by the GET Index Register (GIDX, “FIFO Function”) to OBF Shadow.
VIEW	8	rw	**View Shadow Buffer** Toggles between OBF Shadow and OBF Host. Only writeable while OBCR.OBSYS = 0. 0_B No action 1_B Swap OBF Shadow and OBF Host
REQ	9	rw	**Request Message RAM Transfer** Requests transfer of Message Buffer addressed by OBCR.OBRS from Message RAM to OBF Shadow. Only writeable while OBCR.OBSYS = 0. 0_B No request 1_B Transfer to OBF Shadow requested

(table continues...)

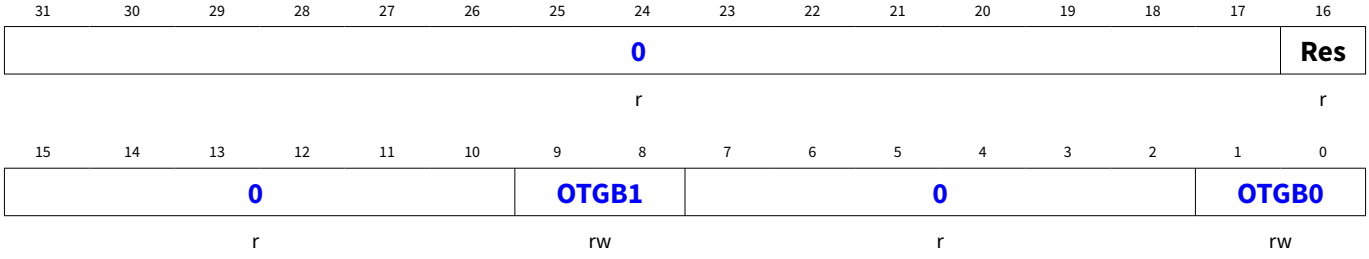
(continued)

Field	Bits	Type	Description
OBSYS	15	rh	**Output Buffer Busy Shadow** Set to 1 after setting bit OBCR.REQ. When the transfer between the Message RAM and OBF Shadow has completed, OBCR.OBSYS is cleared again. 0_B No transfer in progress 1_B Transfer between Message RAM and OBF Shadow in progress
OBRH	22:16	rh	**Output Buffer Request Host** Number of Message Buffer currently accessible by the Host through RDHS1 to RDHS3, MBS, and RDDSn (n = 01-64). By setting OBCR.VIEW OBF Shadow and OBF Host are swapped and the transferred Message Buffer is accessible by the Host. Valid values are 00_H to $7F_H$ (01 to 127).
0	7, 14:10, 31:23	r	**Reserved** Read as 0; should be written with 0.

23.4.96 OCDS Trigger Set Select

OTSS Offset address: 0870_H

OCDS Trigger Set Select Debug Reset value: $0000\ 0000_H$



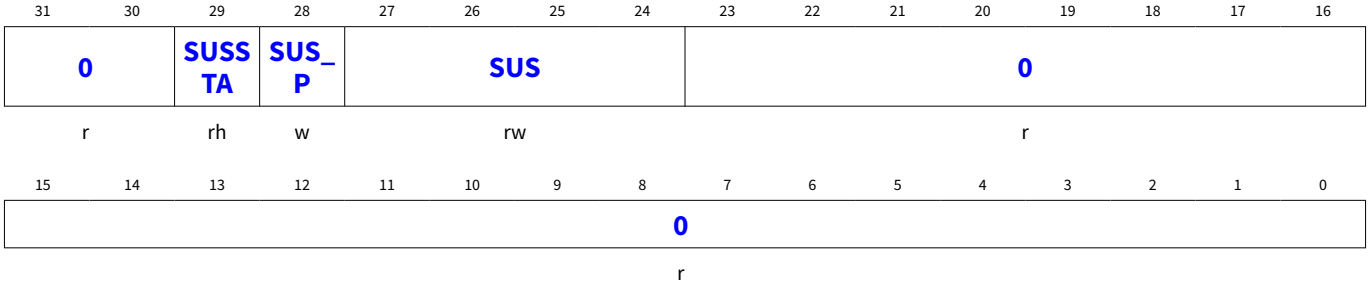
Field	Bits	Type	Description
OTGB0	1:0	rw	**Trigger Set for OTGB0** 00_B No Trigger Set selected 01_B Trigger Set TS16_SEP 10_B Trigger Set TS16_MC 11_B Reserved
OTGB1	9:8	rw	**Trigger Set for OTGB1** 00_B No Trigger Set selected 01_B Trigger Set TS16_SEP 10_B Trigger Set TS16_MC 11_B Reserved
0	7:2, 15:10, 31:17	r	**Reserved** Read as 0; should be written with 0.

23.4.97 OCDS Control and Status Register

The OCDS Control and Status register OCS controls the debug and trace behavior by selecting suspend modes and OTGB trigger sets. When OCDS is disabled, the suspend control is ineffective.

OCS Offset address: $08E8_H$

OCDS Control and Status Register Debug Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
SUS	27:24	rw	**OCDS Suspend Control** Controls the sensitivity to the suspend signal coming from the OCDS Trigger Switch (OTGS) Not listed combinations are reserved. 0_H Will not suspend 1_H Hard suspend: Clock f_{CLC_ERAY} and the sampling clock f_{SCLK} are switched off immediately. Read or write access to registers is possible, it enables the clock for few cycles. 2_H Reserved 3_H Soft suspend mode: Soft suspend request forces ERAY into FREEZE/HALT state (clock remains switched on).
SUS_P	28	w	**SUS Write Protection** SUS is only written when SUS_P is 1, otherwise unchanged. Read as 0.
SUSSTA	29	rh	**Suspend State** 0_B Module is not (yet) suspended 1_B Module is suspended
0	23:0, 31:30	r	**Reserved**

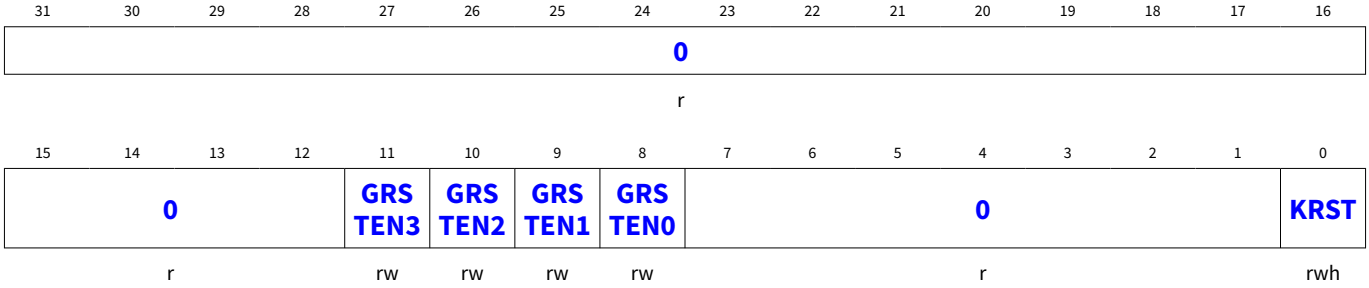
Table 1101 Access mode restrictions of OCS sorted by descending priority

Mode name	Access mode		Description
write 1 to .**SUS_P**	rw	SUS	Set SUS_P during write access
(default)	r	SUS	

23.4.98 Reset Control Register A

RST_CTRLA Offset address: $08EC_H$

Reset Control Register A Application Reset value: $0000\ 0000_H$

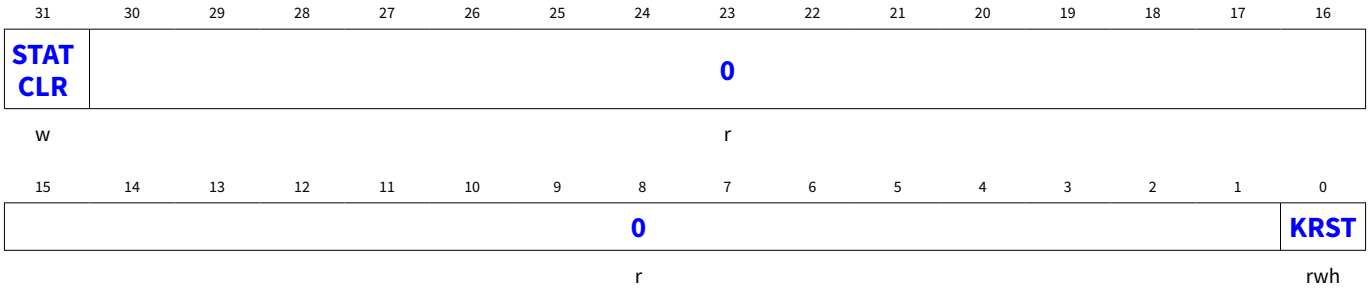


Field	Bits	Type	Description
KRST	0	rwh	**Kernel Reset** Request a kernel reset. The requested reset is executed if CTRLA.KRST=1 and CTRLB.KRST=1. KRST is cleared after the kernel reset was executed. 0_B No action 1_B A kernel reset was requested
GRSTENx (x=0-3)	x+8	rw	**Enable for Global Module Reset Group x** 0_B Global module reset group x does not have any effect 1_B Global module reset group x results in a kernel reset
0	7:1, 31:12	r	**Reserved** Read as 0; should be written with 0.

23.4.99 Reset Control Register B

RST_CTRLB Offset address: $08F0_H$

Reset Control Register B Application Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
KRST	0	rwh	**Kernel Reset** Request a kernel reset. The requested reset is executed if CTRLA.KRST=1 and CTRLB.KRST=1. KRST is cleared after the kernel reset was executed. 0_B No action 1_B A kernel reset was requested

(table continues...)

(continued)

Field	Bits	Type	Description
STATCLR	31	w	**Kernel Reset Status Clear** Clears all status bits in RST_STAT when 1 is written. Read will return 0. 0_B No action 1_B Write with ´1´ clears bits STAT.GRSTx and bit STAT.KRST .
0	30:1	r	**Reserved** Read as 0; should be written with 0.

23.4.100 Reset Status Register

The reset status register contains the status bits for kernel reset (KRST) and the global module reset groups (GRSTx).

RST_STAT Offset address: $08F4_H$

Reset Status Register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				GRST3	GRST2	GRST1	GRST0	0							KRST
r				rh	rh	rh	rh	r							rh

Field	Bits	Type	Description
KRST	0	rh	**Kernel Reset Status** Indicates an executed kernel reset. RST_STAT.KRST is set after the execution of a kernel reset in the same clock cycle in which the reset bits are cleared. Clear KRST by setting bit STATCLR in register RST_CTRLB.
GRSTx (x=0-3)	x+8	rh	**Status for Global Module Reset Group x** 0_B Reset was not triggered by Global Reset Group x 1_B Reset was triggered by Global Reset Group x
0	7:1, 31:12	r	**Reserved** Read as 0; should be written with 0.

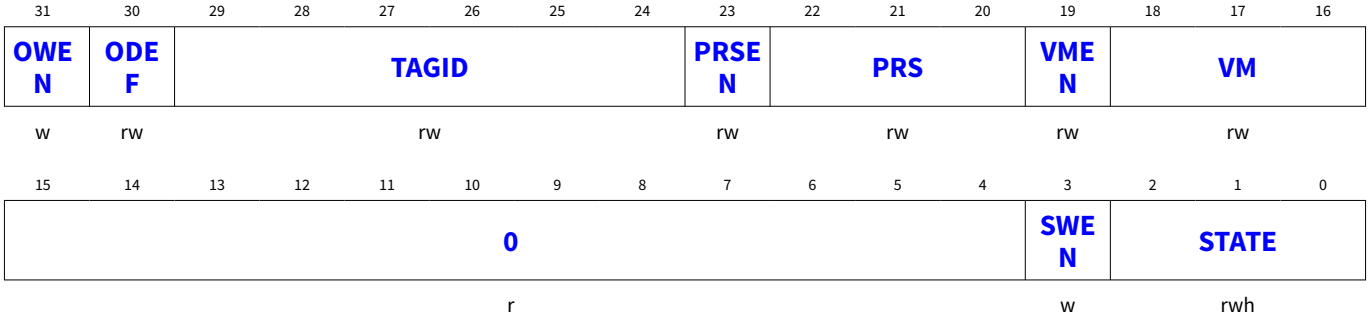
23.4.101 PROT Register Endinit

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner and in some PROT states, the secure master to update the PROT state as described in the PROT mechanism.

The PROTE register controls lock / unlock of the local Endinit (E) protected control registers.

PROTE Offset address: 0900_H

PROT Register Endinit Application Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
VM	18:16	rw	**Virtual machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual machine definition enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.
PRSEN	23	rw	**Protection set definition enable for PROT owner** 0_B **Disable**: PRS is not part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG-ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner

(table continues...)

(continued)

Field	Bits	Type	Description
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; should be written with 0.

Table 1102 Access mode restrictions of PROTE sorted by descending priority

Mode name	Access mode		Description
write 1 to .**SWEN**	rwh	STATE	
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

23.4.102 PROT Register Safe Endinit

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner and in some PROT states, the secure master to update the PROT state as described in the PROT mechanism.

The PROTSE register controls lock / unlock of the local Safe Endinit (SE) protected control registers.

PROTSE Offset address: 0904_H

PROT Register Safe Endinit Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OWEN	**ODEF**	**TAGID**						**PRSEN**	**PRS**			**VMEN**	**VM**		
w	rw	rw						rw	rw			rw	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												**SWEN**	**STATE**		
r												w	rwh		

Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
VM	18:16	rw	**Virtual machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual machine definition enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.
PRSEN	23	rw	**Protection set definition enable for PROT owner** 0_B **Disable**: PRS is not part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG-ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; should be written with 0.

Table 1103 Access mode restrictions of PROTSE sorted by descending priority

Mode name	Access mode		Description
write 1 to .**SWEN**	rwh	STATE	
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

23.4.103 Write access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

ACCEN_WRA Offset address: 0920_H

Write access enable register A Application Reset value: $1000\ 0003_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	**EN30**	**EN29**	**EN28**	**EN27**	**EN26**	**EN25**	**EN24**	**EN23**	**EN22**	**EN21**	**EN20**	**EN19**	**EN18**	**EN17**	**EN16**
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	**EN14**	**EN13**	**EN12**	**EN11**	**EN10**	**EN09**	**EN08**	**EN07**	**EN06**	**EN05**	**EN04**	**EN03**	**EN02**	**EN01**	**EN00**
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access

23.4.104 Write access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

ACCEN_WRB Offset address: 0924_H

Write access enable register B Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								**EN39**	**EN38**	**EN37**	**EN36**	**EN35**	**EN34**	**EN33**	**EN32**
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access
0	31:8	r	**Reserved** Read as 0; should be written with 0.

23.4.105 Read access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

ACCEN_RDA Offset address: 0928_H

Read access enable register A Application Reset value: $FFFF\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access

23.4.106 Read access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

ACCEN_RDB Offset address: $092C_H$

Read access enable register B Application Reset value: $0000\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access
0	31:8	r	**Reserved** Read as 0; should be written with 0.

23.4.107 VM access enable register

This register defines which virtual machine encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each VMx.

ACCEN_VM Offset address: 0930_H

VM access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for VM ID q** This bit enables read access to the access protected region for transactions with the VM ID q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for VM ID q** This bit enables write access to the access protected region for transactions with the VM ID q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

23.4.108 PRS access enable register

This register defines which protection register set encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each PRSx.

ACCEN_PRS Offset address: 0934_H

PRS access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for PRS q** This bit enables read access to the access protected region for transactions with the PRS q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for PRS q** This bit enables write access to the access protected region for transactions with the PRS q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

23.5 Debug information

ERAY debug information.

23.5.1 OCDS trigger bus (OTGB) interface

The ERAY module has two 16-bit trigger sets (Table 1104) which are selected with the OTSS register.

Table 1104 ERAY trigger sets

Trigger set	Details
TS16_SEP Service Requests, Errors and POC State	Table 1106
TS16_MC Macrotick Counter	Table 1107

Table 1105 shows all possible trigger set mapping options. If OTGB0 and/or OTGB1 is not used for ERAY, trigger sets of other sources can be added in the OTGM module.

Table 1105 Trigger set mapping options

Width	OTGB0	OTGB1
32-bit	TS16_SEP/MC	
		TS16_SEP/MC
	TS16_SEP/MC	TS16_SEP/MC
64-bit		
	TS16_SEP/MC	
		TS16_SEP/MC
	TS16_SEP/MC	TS16_SEP/M

Table 1106 TS16_SEP service requests, errors and POC state

Bits	Description
0	Interrupt 0 Service Request (INT0SRC)
1	Interrupt 1 Service Request (INT1SRC)
2	Timer Interrupt 0 Service Request (TINT0SRC)
3	Timer Interrupt 1 Service Request (TINT1SRC)
4	New Data 0 Service Request (NDAT0SRC)
5	New Data 1 Service Request (NDAT1SRC)
6	Message Buffer Status Changed 0 Service Request (MBSC0SRC)
7	Message Buffer Status Changed 1 Service Request (MBSC1SRC)
8	Output Buffer Busy Service Request (OBUSYSRC)
9	Input Buffer Busy Service Request (IBUSYSRC)
10	Reserved
11	Error on Channel A (EIR.EDA)
12	Error on Channel B (EIR.EDB)

(table continues...)

Table 1106 (continued) TS16_SEP service requests, errors and POC state

Bits	Description
[15:13]	POC State bits[2:0] (CCSV.POCS)

Table 1107 TS16_MC macrotick counter

Bits	Description
[13:0]	Macrotick Value (MTCCV.MTV)
[15:14]	Reserved

23.5.2 Destructive debug

On destructive debug entry ie., when CSRM Debug configuration GP_CSRM_DBGCFG.DESTDBG= 11b and CSRM Debug mode GP_CSRM_DBGMODE.EDDM = 11b, the ERAY Transmit Data and transmit enable for Channel A and Channel B (TXDA, TXDB, TXENA and TXENB) are tied to logic ‘1b’. This blocks any further ERAY transmission when the device has entered the Destructive Debug state.

23.6 References

1. FlexRay Consortium, FlexRay™ Communications System Protocol Specification Version 2.1, Decemeber 2005

23.7 ERAY revision history

Initial release of the chapter.

23.8 TC4Dx ERAY information

23.8.1 TC4Dx ERAY configuration

TC4Dx has 2 ERAY instances: ERAY0 and ERAY1. Each of these instances has 2 channels: Channel A and Channel B.

23.8.2 TC4Dx ERAY features

There are no deviations from the generic specification.

23.8.3 TC4Dx ERAY functional description

In TC4Dx device, the ERAY modules are connected to ComPB bus interface. The ERAY chapter refers to it as FPI bus.

23.8.4 TC4Dx ERAY registers

23.8.4.1 Register address space - ERAY

Table 1108 Registers address space - ERAY

Module	Base address	End address	Note
ERAY0	$F441C000_H$	$F441CFFF_H$	FPI slave interface
ERAY1	$F441D000_H$	$F441DFFF_H$	FPI slave interface

23.8.4.2 Register overview - access mode glossary

Table 1109 Register overview - access mode glossary

Keyword	Description
E	Access protection using PROT register ERAY0_PROTE or ERAY1_PROTE .
SE	Access protection using PROT register ERAY0_PROTSE or ERAY1_PROTSE .
APU-P	Protection group consisting of registers ERAY0_ACCEN_WRA , ERAY0_ACCEN_WRB , ERAY0_ACCEN_RDA , ERAY0_ACCEN_RDB , ERAY0_ACCEN_VM , ERAY0_ACCEN_PRS or ERAY1_ACCEN_WRA , ERAY1_ACCEN_WRB , ERAY1_ACCEN_RDA , ERAY1_ACCEN_RDB , ERAY1_ACCEN_VM , ERAY1_ACCEN_PRS .
P	Access protection using *APU-P* registers.
BE	Always returns a Bus Error.
nBE	Indicates that no Bus Error is generated when accessing this address range, even though it is either an access to an undefined address or the access does not follow the given rules.
U	No access restrictions.
PROT	Access restrictions as defined in the PROT register access rules.

23.8.4.3 Register overview - ERAY0 (ascending offset address)

Table 1110 Register overview - ERAY0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ERAY0_CLC	Clock Control Register	0000_H	P	P, SV, E	Application Reset	4488
ERAY0_CUST1	Busy and Input Buffer Control Register	0004_H	P	P	Kernel Reset	4488
ERAY0_ID	Module Identification Register	0008_H	P	BE	PowerOn Reset	4490
ERAY0_CUST3	Customer Interface Timeout Counter Register	$000C_H$	P	P	Kernel Reset	4491
ERAY0_TEST1	Test Register 1	0010_H	P	P	Kernel Reset	4492
ERAY0_TEST2	Test Register 2	0014_H	P	P	Kernel Reset	4496
ERAY0_LCK	Lock Register	$001C_H$	P	P	Kernel Reset	4498

(table continues...)

Table 1110 (continued) Register overview - ERAY0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ERAY0_EIR	Error Service Request Select Register	0020_H	P	P	Kernel Reset	4499
ERAY0_SIR	Status Service Request Register	0024_H	P	P	Kernel Reset	4502
ERAY0_EILS	Error Service Request Line Select	0028_H	P	P	Kernel Reset	4506
ERAY0_SILS	Status Service Request Line Select	$002C_H$	P	P	Kernel Reset	4508
ERAY0_EIES	Error Service Request Enable Set	0030_H	P	P	Kernel Reset	4510
ERAY0_EIER	Error Service Request Enable Reset	0034_H	P	P	Kernel Reset	4513
ERAY0_SIES	Status Service Request Enable Set	0038_H	P	P	Kernel Reset	4516
ERAY0_SIER	Status Service Request Enable Reset	$003C_H$	P	P	Kernel Reset	4518
ERAY0_ILE	Service Request Line Enable	0040_H	P	P	Kernel Reset	4521
ERAY0_T0C	Timer 0 Configuration	0044_H	P	P	Kernel Reset	4522
ERAY0_T1C	Timer 1 Configuration	0048_H	P	P	Kernel Reset	4523
ERAY0_STPW1	Stop Watch Register 1	$004C_H$	P	P	Kernel Reset	4524
ERAY0_STPW2	Stop Watch Register 2	0050_H	P	P	Kernel Reset	4525
ERAY0_SUCC1	SUC Configuration Register 1	0080_H	P	P	Kernel Reset	4526
ERAY0_SUCC2	SUC Configuration Register 2	0084_H	P	P	Kernel Reset	4531
ERAY0_SUCC3	SUC Configuration Register 3	0088_H	P	P	Kernel Reset	4532
ERAY0_NEMC	NEM Configuration Register	$008C_H$	P	P	Kernel Reset	4533
ERAY0_PRTC1	PRT Configuration Register 1	0090_H	P	P	Kernel Reset	4533
ERAY0_PRTC2	PRT Configuration Register 2	0094_H	P	P	Kernel Reset	4534
ERAY0_MHDC	MHD Configuration Register	0098_H	P	P	Kernel Reset	4535
ERAY0_GTUC01	GTU Configuration Register 1	$00A0_H$	P	P	Kernel Reset	4536
ERAY0_GTUC02	GTU Configuration Register 2	$00A4_H$	P	P	Kernel Reset	4536
ERAY0_GTUC03	GTU Configuration Register 3	$00A8_H$	P	P	Kernel Reset	4537

(table continues...)

Table 1110 (continued) Register overview - ERAY0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ERAY0_GTUC04	GTU Configuration Register 4	$00AC_H$	P	P	Kernel Reset	4538
ERAY0_GTUC05	GTU Configuration Register 5	$00B0_H$	P	P	Kernel Reset	4539
ERAY0_GTUC06	GTU Configuration Register 6	$00B4_H$	P	P	Kernel Reset	4540
ERAY0_GTUC07	GTU Configuration Register 7	$00B8_H$	P	P	Kernel Reset	4540
ERAY0_GTUC08	GTU Configuration Register 8	$00BC_H$	P	P	Kernel Reset	4541
ERAY0_GTUC09	GTU Configuration Register 9	$00C0_H$	P	P	Kernel Reset	4541
ERAY0_GTUC10	GTU Configuration Register 10	$00C4_H$	P	P	Kernel Reset	4542
ERAY0_GTUC11	GTU Configuration Register 11	$00C8_H$	P	P	Kernel Reset	4543
ERAY0_CCSV	Communication Controller Status Vector	0100_H	P	BE	Kernel Reset	4544
ERAY0_CCEV	Communication Controller Error Vector	0104_H	P	BE	Kernel Reset	4547
ERAY0_SCV	Slot Counter Value	0110_H	P	BE	Kernel Reset	4548
ERAY0_MTCCV	Macrotick and Cycle Counter Value	0114_H	P	BE	Kernel Reset	4549
ERAY0_RCV	Rate Correction Value	0118_H	P	BE	Kernel Reset	4549
ERAY0_OCV	Offset Correction Value	$011C_H$	P	BE	Kernel Reset	4550
ERAY0_SFS	SYNC Frame Status	0120_H	P	BE	Kernel Reset	4550
ERAY0_SWNIT	Symbol Window and Network Idle Time Status	0124_H	P	BE	Kernel Reset	4552
ERAY0_ACS	Aggregated Channel Status	0128_H	P	P	Kernel Reset	4554
ERAY0_ESIDn (n=01-15)	Even Sync ID Symbol Window n	0130_H+ (n-1)*4	P	BE	Kernel Reset	4556
ERAY0_OSIDn (n=01-15)	Odd Sync ID Symbol Window n	0170_H+ (n-1)*4	P	BE	Kernel Reset	4557
ERAY0_NMVx (x=1-3)	Network Management Vector x	$01B0_H$+ (x-1)*4	P	BE	Kernel Reset	4558
ERAY0_MRC	Message RAM Configuration	0300_H	P	P	Kernel Reset	4559
ERAY0_FRF	FIFO Rejection Filter	0304_H	P	P	Kernel Reset	4561
ERAY0_FRFM	FIFO Rejection Filter Mask	0308_H	P	P	Kernel Reset	4562
ERAY0_FCL	FIFO Critical Level	$030C_H$	P	P	Kernel Reset	4563

(table continues...)

Table 1110 (continued) Register overview - ERAY0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ERAY0_MHDS	Message Handler Status	0310_H	P	P	Kernel Reset	4564
ERAY0_LDTS	Last Dynamic Transmit Slot	0314_H	P	P	Kernel Reset	4565
ERAY0_FSR	FIFO Status Register	0318_H	P	P	Kernel Reset	4566
ERAY0_MHDF	Message Handler Constraints Flags	$031C_H$	P	P	Kernel Reset	4567
ERAY0_TXRQ1	Transmission Request Register 1	0320_H	P	BE	Kernel Reset	4569
ERAY0_TXRQ2	Transmission Request Register 2	0324_H	P	BE	Kernel Reset	4570
ERAY0_TXRQ3	Transmission Request Register 3	0328_H	P	BE	Kernel Reset	4570
ERAY0_TXRQ4	Transmission Request Register 4	$032C_H$	P	BE	Kernel Reset	4571
ERAY0_NDAT1	New Data Register 1	0330_H	P	BE	Kernel Reset	4571
ERAY0_NDAT2	New Data Register 2	0334_H	P	BE	Kernel Reset	4572
ERAY0_NDAT3	New Data Register 3	0338_H	P	BE	Kernel Reset	4572
ERAY0_NDAT4	New Data Register 4	$033C_H$	P	BE	Kernel Reset	4573
ERAY0_MBSC1	Message Buffer Status Changed 1	0340_H	P	BE	Kernel Reset	4574
ERAY0_MBSC2	Message Buffer Status Changed 2	0344_H	P	BE	Kernel Reset	4574
ERAY0_MBSC3	Message Buffer Status Changed 3	0348_H	P	BE	Kernel Reset	4575
ERAY0_MBSC4	Message Buffer Status Changed 4	$034C_H$	P	BE	Kernel Reset	4575
ERAY0_NDIC1	New Data Interrupt Control 1	$03A8_H$	P	P	Kernel Reset	4576
ERAY0_NDIC2	New Data Interrupt Control 2	$03AC_H$	P	P	Kernel Reset	4577
ERAY0_NDIC3	New Data Interrupt Control 3	$03B0_H$	P	P	Kernel Reset	4577
ERAY0_NDIC4	New Data Interrupt Control 4	$03B4_H$	P	P	Kernel Reset	4578
ERAY0_MSIC1	Message Buffer Status Changed Interrupt Control 1	$03B8_H$	P	P	Kernel Reset	4578
ERAY0_MSIC2	Message Buffer Status Changed Interrupt Control 2	$03BC_H$	P	P	Kernel Reset	4579

(table continues...)

Table 1110 (continued) Register overview - ERAY0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ERAY0_MSIC3	Message Buffer Status Changed Interrupt Control 3	$03C0_H$	P	P	Kernel Reset	4580
ERAY0_MSIC4	Message Buffer Status Changed Interrupt Control 4	$03C4_H$	P	P	Kernel Reset	4580
ERAY0_CREL	Core Release Register	$03F0_H$	P	nBE	Kernel Reset	4581
ERAY0_ENDN	Endian Register	$03F4_H$	P	nBE	Kernel Reset	4582
ERAY0_WRDSn (n=01-64)	Write Data Section n	0400_H+ (n-1)*4	P	P	Kernel Reset	4583
ERAY0_WRHS1	Write Header Section 1	0500_H	P	P	Kernel Reset	4584
ERAY0_WRHS2	Write Header Section 2	0504_H	P	P	Kernel Reset	4585
ERAY0_WRHS3	Write Header Section 3	0508_H	P	P	Kernel Reset	4586
ERAY0_IBCM	Input Buffer Command Mask	0510_H	P	P	Kernel Reset	4587
ERAY0_IBCR	Input Buffer Command Request	0514_H	P	P	Kernel Reset	4588
ERAY0_RDDSn (n=01-64)	Read Data Section n	0600_H+ (n-1)*4	P	P	Kernel Reset	4589
ERAY0_RDHS1	Read Header Section 1	0700_H	P	P	Kernel Reset	4590
ERAY0_RDHS2	Read Header Section 2	0704_H	P	P	Kernel Reset	4591
ERAY0_RDHS3	Read Header Section 3	0708_H	P	P	Kernel Reset	4592
ERAY0_MBS	Message Buffer Status	$070C_H$	P	P	Kernel Reset	4593
ERAY0_OBCM	Output Buffer Command Mask	0710_H	P	P	Kernel Reset	4598
ERAY0_OBCR	Output Buffer Command Request	0714_H	P	P	Kernel Reset	4599
ERAY0_OTSS	OCDS Trigger Set Select	0870_H	P	SV, E, P	Debug Reset	4600
ERAY0_OCS	OCDS Control and Status Register	$08E8_H$	P	SV, P	Debug Reset	4601
ERAY0_RST_CTRLA	Reset Control Register A	$08EC_H$	P	P, SV, E	Application Reset	4601
ERAY0_RST_CTRLB	Reset Control Register B	$08F0_H$	P	P, SV, E	Application Reset	4602
ERAY0_RST_STAT	Reset Status Register	$08F4_H$	P	BE	Application Reset	4603
ERAY0_PROTE	PROT Register Endinit	0900_H	U	SV, PROT	Application Reset	4603
ERAY0_PROTSE	PROT Register Safe Endinit	0904_H	U	SV, PROT	Application Reset	4605

(table continues...)

Table 1110 (continued) Register overview - ERAY0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ERAY0_ACCEN_WRA	Write access enable register A	0920_H	U	SE, SV	Application Reset	4607
ERAY0_ACCEN_WRB	Write access enable register B	0924_H	U	SE, SV	Application Reset	4607
ERAY0_ACCEN_RDA	Read access enable register A	0928_H	U	SE, SV	Application Reset	4608
ERAY0_ACCEN_RDB	Read access enable register B	$092C_H$	U	SE, SV	Application Reset	4608
ERAY0_ACCEN_VM	VM access enable register	0930_H	U	SE, SV	Application Reset	4609
ERAY0_ACCEN_PRS	PRS access enable register	0934_H	U	SE, SV	Application Reset	4609

23.8.4.4 Register overview - ERAY1 (ascending offset address)

Table 1111 Register overview - ERAY1 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ERAY1_CLC	Clock Control Register	0000_H	P	P, SV, E	Application Reset	4488
ERAY1_CUST1	Busy and Input Buffer Control Register	0004_H	P	P	Kernel Reset	4488
ERAY1_ID	Module Identification Register	0008_H	P	BE	PowerOn Reset	4490
ERAY1_CUST3	Customer Interface Timeout Counter Register	$000C_H$	P	P	Kernel Reset	4491
ERAY1_TEST1	Test Register 1	0010_H	P	P	Kernel Reset	4492
ERAY1_TEST2	Test Register 2	0014_H	P	P	Kernel Reset	4496
ERAY1_LCK	Lock Register	$001C_H$	P	P	Kernel Reset	4498
ERAY1_EIR	Error Service Request Select Register	0020_H	P	P	Kernel Reset	4499
ERAY1_SIR	Status Service Request Register	0024_H	P	P	Kernel Reset	4502
ERAY1_EILS	Error Service Request Line Select	0028_H	P	P	Kernel Reset	4506
ERAY1_SILS	Status Service Request Line Select	$002C_H$	P	P	Kernel Reset	4508
ERAY1_EIES	Error Service Request Enable Set	0030_H	P	P	Kernel Reset	4510
ERAY1_EIER	Error Service Request Enable Reset	0034_H	P	P	Kernel Reset	4513

(table continues...)

Table 1111 (continued) Register overview - ERAY1 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ERAY1_SIES	Status Service Request Enable Set	0038_H	P	P	Kernel Reset	4516
ERAY1_SIER	Status Service Request Enable Reset	$003C_H$	P	P	Kernel Reset	4518
ERAY1_ILE	Service Request Line Enable	0040_H	P	P	Kernel Reset	4521
ERAY1_T0C	Timer 0 Configuration	0044_H	P	P	Kernel Reset	4522
ERAY1_T1C	Timer 1 Configuration	0048_H	P	P	Kernel Reset	4523
ERAY1_STPW1	Stop Watch Register 1	$004C_H$	P	P	Kernel Reset	4524
ERAY1_STPW2	Stop Watch Register 2	0050_H	P	P	Kernel Reset	4525
ERAY1_SUCC1	SUC Configuration Register 1	0080_H	P	P	Kernel Reset	4526
ERAY1_SUCC2	SUC Configuration Register 2	0084_H	P	P	Kernel Reset	4531
ERAY1_SUCC3	SUC Configuration Register 3	0088_H	P	P	Kernel Reset	4532
ERAY1_NEMC	NEM Configuration Register	$008C_H$	P	P	Kernel Reset	4533
ERAY1_PRTC1	PRT Configuration Register 1	0090_H	P	P	Kernel Reset	4533
ERAY1_PRTC2	PRT Configuration Register 2	0094_H	P	P	Kernel Reset	4534
ERAY1_MHDC	MHD Configuration Register	0098_H	P	P	Kernel Reset	4535
ERAY1_GTUC01	GTU Configuration Register 1	$00A0_H$	P	P	Kernel Reset	4536
ERAY1_GTUC02	GTU Configuration Register 2	$00A4_H$	P	P	Kernel Reset	4536
ERAY1_GTUC03	GTU Configuration Register 3	$00A8_H$	P	P	Kernel Reset	4537
ERAY1_GTUC04	GTU Configuration Register 4	$00AC_H$	P	P	Kernel Reset	4538
ERAY1_GTUC05	GTU Configuration Register 5	$00B0_H$	P	P	Kernel Reset	4539
ERAY1_GTUC06	GTU Configuration Register 6	$00B4_H$	P	P	Kernel Reset	4540
ERAY1_GTUC07	GTU Configuration Register 7	$00B8_H$	P	P	Kernel Reset	4540
ERAY1_GTUC08	GTU Configuration Register 8	$00BC_H$	P	P	Kernel Reset	4541
ERAY1_GTUC09	GTU Configuration Register 9	$00C0_H$	P	P	Kernel Reset	4541

(table continues...)

Table 1111 (continued) Register overview - ERAY1 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ERAY1_GTUC10	GTU Configuration Register 10	$00C4_H$	P	P	Kernel Reset	4542
ERAY1_GTUC11	GTU Configuration Register 11	$00C8_H$	P	P	Kernel Reset	4543
ERAY1_CCSV	Communication Controller Status Vector	0100_H	P	BE	Kernel Reset	4544
ERAY1_CCEV	Communication Controller Error Vector	0104_H	P	BE	Kernel Reset	4547
ERAY1_SCV	Slot Counter Value	0110_H	P	BE	Kernel Reset	4548
ERAY1_MTCCV	Macrotick and Cycle Counter Value	0114_H	P	BE	Kernel Reset	4549
ERAY1_RCV	Rate Correction Value	0118_H	P	BE	Kernel Reset	4549
ERAY1_OCV	Offset Correction Value	$011C_H$	P	BE	Kernel Reset	4550
ERAY1_SFS	SYNC Frame Status	0120_H	P	BE	Kernel Reset	4550
ERAY1_SWNIT	Symbol Window and Network Idle Time Status	0124_H	P	BE	Kernel Reset	4552
ERAY1_ACS	Aggregated Channel Status	0128_H	P	P	Kernel Reset	4554
ERAY1_ESIDn (n=01-15)	Even Sync ID Symbol Window n	0130_H+ (n-1)*4	P	BE	Kernel Reset	4556
ERAY1_OSIDn (n=01-15)	Odd Sync ID Symbol Window n	0170_H+ (n-1)*4	P	BE	Kernel Reset	4557
ERAY1_NMVx (x=1-3)	Network Management Vector x	$01B0_H$+ (x-1)*4	P	BE	Kernel Reset	4558
ERAY1_MRC	Message RAM Configuration	0300_H	P	P	Kernel Reset	4559
ERAY1_FRF	FIFO Rejection Filter	0304_H	P	P	Kernel Reset	4561
ERAY1_FRFM	FIFO Rejection Filter Mask	0308_H	P	P	Kernel Reset	4562
ERAY1_FCL	FIFO Critical Level	$030C_H$	P	P	Kernel Reset	4563
ERAY1_MHDS	Message Handler Status	0310_H	P	P	Kernel Reset	4564
ERAY1_LDTS	Last Dynamic Transmit Slot	0314_H	P	P	Kernel Reset	4565
ERAY1_FSR	FIFO Status Register	0318_H	P	P	Kernel Reset	4566
ERAY1_MHDF	Message Handler Constraints Flags	$031C_H$	P	P	Kernel Reset	4567
ERAY1_TXRQ1	Transmission Request Register 1	0320_H	P	BE	Kernel Reset	4569
ERAY1_TXRQ2	Transmission Request Register 2	0324_H	P	BE	Kernel Reset	4570
ERAY1_TXRQ3	Transmission Request Register 3	0328_H	P	BE	Kernel Reset	4570

(table continues...)

Table 1111 (continued) Register overview - ERAY1 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ERAY1_TXRQ4	Transmission Request Register 4	$032C_H$	P	BE	Kernel Reset	4571
ERAY1_NDAT1	New Data Register 1	0330_H	P	BE	Kernel Reset	4571
ERAY1_NDAT2	New Data Register 2	0334_H	P	BE	Kernel Reset	4572
ERAY1_NDAT3	New Data Register 3	0338_H	P	BE	Kernel Reset	4572
ERAY1_NDAT4	New Data Register 4	$033C_H$	P	BE	Kernel Reset	4573
ERAY1_MBSC1	Message Buffer Status Changed 1	0340_H	P	BE	Kernel Reset	4574
ERAY1_MBSC2	Message Buffer Status Changed 2	0344_H	P	BE	Kernel Reset	4574
ERAY1_MBSC3	Message Buffer Status Changed 3	0348_H	P	BE	Kernel Reset	4575
ERAY1_MBSC4	Message Buffer Status Changed 4	$034C_H$	P	BE	Kernel Reset	4575
ERAY1_NDIC1	New Data Interrupt Control 1	$03A8_H$	P	P	Kernel Reset	4576
ERAY1_NDIC2	New Data Interrupt Control 2	$03AC_H$	P	P	Kernel Reset	4577
ERAY1_NDIC3	New Data Interrupt Control 3	$03B0_H$	P	P	Kernel Reset	4577
ERAY1_NDIC4	New Data Interrupt Control 4	$03B4_H$	P	P	Kernel Reset	4578
ERAY1_MSIC1	Message Buffer Status Changed Interrupt Control 1	$03B8_H$	P	P	Kernel Reset	4578
ERAY1_MSIC2	Message Buffer Status Changed Interrupt Control 2	$03BC_H$	P	P	Kernel Reset	4579
ERAY1_MSIC3	Message Buffer Status Changed Interrupt Control 3	$03C0_H$	P	P	Kernel Reset	4580
ERAY1_MSIC4	Message Buffer Status Changed Interrupt Control 4	$03C4_H$	P	P	Kernel Reset	4580
ERAY1_CREL	Core Release Register	$03F0_H$	P	nBE	Kernel Reset	4581
ERAY1_ENDN	Endian Register	$03F4_H$	P	nBE	Kernel Reset	4582
ERAY1_WRDSn (n=01-64)	Write Data Section n	0400_H+ (n-1)*4	P	P	Kernel Reset	4583
ERAY1_WRHS1	Write Header Section 1	0500_H	P	P	Kernel Reset	4584
ERAY1_WRHS2	Write Header Section 2	0504_H	P	P	Kernel Reset	4585
ERAY1_WRHS3	Write Header Section 3	0508_H	P	P	Kernel Reset	4586

(table continues...)

Table 1111 (continued) Register overview - ERAY1 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ERAY1_IBCM	Input Buffer Command Mask	0510_H	P	P	Kernel Reset	4587
ERAY1_IBCR	Input Buffer Command Request	0514_H	P	P	Kernel Reset	4588
ERAY1_RDDSn (n=01-64)	Read Data Section n	0600_H+ (n-1)*4	P	P	Kernel Reset	4589
ERAY1_RDHS1	Read Header Section 1	0700_H	P	P	Kernel Reset	4590
ERAY1_RDHS2	Read Header Section 2	0704_H	P	P	Kernel Reset	4591
ERAY1_RDHS3	Read Header Section 3	0708_H	P	P	Kernel Reset	4592
ERAY1_MBS	Message Buffer Status	$070C_H$	P	P	Kernel Reset	4593
ERAY1_OBCM	Output Buffer Command Mask	0710_H	P	P	Kernel Reset	4598
ERAY1_OBCR	Output Buffer Command Request	0714_H	P	P	Kernel Reset	4599
ERAY1_OTSS	OCDS Trigger Set Select	0870_H	P	SV, E, P	Debug Reset	4600
ERAY1_OCS	OCDS Control and Status Register	$08E8_H$	P	SV, P	Debug Reset	4601
ERAY1_RST_CTRLA	Reset Control Register A	$08EC_H$	P	P, SV, E	Application Reset	4601
ERAY1_RST_CTRLB	Reset Control Register B	$08F0_H$	P	P, SV, E	Application Reset	4602
ERAY1_RST_STAT	Reset Status Register	$08F4_H$	P	BE	Application Reset	4603
ERAY1_PROTE	PROT Register Endinit	0900_H	U	SV, PROT	Application Reset	4603
ERAY1_PROTSE	PROT Register Safe Endinit	0904_H	U	SV, PROT	Application Reset	4605
ERAY1_ACCEN_WRA	Write access enable register A	0920_H	U	SE, SV	Application Reset	4607
ERAY1_ACCEN_WRB	Write access enable register B	0924_H	U	SE, SV	Application Reset	4607
ERAY1_ACCEN_RDA	Read access enable register A	0928_H	U	SE, SV	Application Reset	4608
ERAY1_ACCEN_RDB	Read access enable register B	$092C_H$	U	SE, SV	Application Reset	4608
ERAY1_ACCEN_VM	VM access enable register	0930_H	U	SE, SV	Application Reset	4609
ERAY1_ACCEN_PRS	PRS access enable register	0934_H	U	SE, SV	Application Reset	4609

23.8.4.5 Device specific registers

There are no device specific register changes.

23.8.5 TC4Dx ERAY connectivity

The table below lists interfaces to and from this functional block to other blocks in the device.

Table 1112 List of ERAY interface signals

Interface signals	I/O	Description
CLOCK_ERAY_fSPB	In	Synchronous clock input
CLOCK_ERAY_fERAY	In	Asynchronous clock input
FPI_ERAY_SIF	In	Slave interface
ERAY0_IR_INT0	Out	Interrupt 0 Service Request
ERAY1_IR_INT0	Out	Interrupt 0 Service Request
ERAY0_IR_INT1	Out	Interrupt 1 Service Request
ERAY1_IR_INT1	Out	Interrupt 1 Service Request
ERAY0_IR_TINT0	Out	Timer Interrupt 0 Service Request
ERAY1_IR_TINT0	Out	Timer Interrupt 0 Service Request
ERAY0_IR_TINT1	Out	Timer Interrupt 1 Service Request
ERAY1_IR_TINT1	Out	Timer Interrupt 1 Service Request
ERAY0_IR_OBUSY	Out	Output Buffer Busy Service Request
ERAY1_IR_OBUSY	Out	Output Buffer Busy Service Request
ERAY0_IR_IBUSY	Out	Input Buffer Busy Service Request
ERAY1_IR_IBUSY	Out	Input Buffer Busy Service Request
ERAY0_IR_NDAT0	Out	New Data 0 Service Request
ERAY1_IR_NDAT0	Out	New Data 0 Service Request
ERAY0_IR_NDAT1	Out	New Data 1 Service Request
ERAY1_IR_NDAT1	Out	New Data 1 Service Request
ERAY0_IR_MBSC0	Out	Message Buffer Status Changed 0 Service Request
ERAY1_IR_MBSC0	Out	Message Buffer Status Changed 0 Service Request
ERAY0_IR_MBSC1	Out	Message Buffer Status Changed 1 Service Request
ERAY1_IR_MBSC1	Out	Message Buffer Status Changed 1 Service Request
PORTS_ERAY0_RXDA0	In	Receive Channel A0
PORTS_ERAY1_RXDA0	In	Receive Channel A0
PORTS_ERAY0_RXDA1	In	Receive Channel A1
PORTS_ERAY1_RXDA1	In	Receive Channel A1
PORTS_ERAY0_RXDA2	In	Receive Channel A2
PORTS_ERAY1_RXDA2	In	Receive Channel A2

(table continues...)

Table 1112 (continued) List of ERAY interface signals

Interface signals	I/O	Description
PORTS_ERAY0_RXDA3	In	Receive Channel A3
PORTS_ERAY1_RXDA3	In	Receive Channel A3
PORTS_ERAY0_RXDB0	In	Receive Channel B0
PORTS_ERAY1_RXDB0	In	Receive Channel B0
PORTS_ERAY0_RXDB1	In	Receive Channel B1
PORTS_ERAY1_RXDB1	In	Receive Channel B1
PORTS_ERAY0_RXDB2	In	Receive Channel B2
PORTS_ERAY1_RXDB2	In	Receive Channel B2
PORTS_ERAY0_RXDB3	In	Receive Channel B3
PORTS_ERAY1_RXDB3	In	Receive Channel B3
ERAY0_PORTS_TXDA	Out	Transmit Channel A
ERAY1_PORTS_TXDA	Out	Transmit Channel A
ERAY0_PORTS_TXENA	Out	Transmit Enable Channel A
ERAY1_PORTS_TXENA	Out	Transmit Enable Channel A
ERAY0_PORTS_TXDB	Out	Transmit Channel B
ERAY1_PORTS_TXDB	Out	Transmit Channel B
ERAY0_PORTS_TXENB	Out	Transmit Enable Channel B
ERAY1_PORTS_TXENB	Out	Transmit Enable Channel B
ERAY0_CLOCK_fMT	Out	Macrotick-clock from CC (External Clock Output synchronous to SPB clock)
ERAY1_CLOCK_fMT	Out	Macrotick-clock from CC (External Clock Output synchronous to SPB clock)
ERAY0_SCU_MACROTICK	Out	Macrotick-clock from CC (External Clock Output synchronous to SPB clock)
ERAY1_SCU_MACROTICK	Out	Macrotick-clock from CC (External Clock Output synchronous to SPB clock)
ERAY0_EGTM_MACROTICK	Out	Macrotick-clock from CC (External Clock Output synchronous to SPB clock)
ERAY1_EGTM_MACROTICK	Out	Macrotick-clock from CC (External Clock Output synchronous to SPB clock)
SCU_ERAY0_STPWT[3:0]	In	Timer input for capture, Stop Watch Trigger Signal (External Request Unit)
SCU_ERAY1_STPWT[3:0]	In	Timer input for capture, Stop Watch Trigger Signal (External Request Unit)

23.8.6 TC4Dx ERAY revision history

Initial release of the chapter.

24 Asynchronous/Synchronous Interface (ASCLIN)

The ASCLIN module provides asynchronous serial communication with external devices using only data-in and data-out signals. The focus of the module is for fast and flexible communication, either as fast point-to-point using ASC protocol or master-to-many slaves communication using the LIN protocol. ASCLIN supports also the synchronous SPI communication.

24.1 Feature list

General features

- Transmit ASC, LIN, and SPI frames to external devices and handle all protocol relevant topics
- Receive ASC, LIN, and SPI frames from external devices and handle all protocol relevant topics
- Data storage of 16-bytes TxFIFO (Transmit first in first out) and 16-bytes RxFIFO (Receive first in first out)
- Pack and unpack capabilities of the TxFIFO and RxFIFO
- Interrupt generation:
  - On a configurable TxFIFO level
  - On a configurable RxFIFO level
  - On an error condition (frame, parity, overrun error)
  - On various internal events (end of ASC or SPI frame, or LIN events)
- Interrupt signals capable of triggering either a CPU or a DMA
- Flexible functional configuration for software:
  - Programmable oversampling of 4 to 16 times per bit
  - Programmable sampling point position
  - Programmable digital glitch filter and median filter for incoming bit stream
  - Shift direction LSB first for LIN mode and programmable LSB or MSB first for SPI and ASC mode
- Internal loop-back mode

Standard ASC features

- Full-duplex asynchronous operating mode:
  - 7-bit, 8-bit, or 9-bit (or up to 16-bit) data frames, LSB first
  - Parity-bit generation and checking
  - One or two stop bits
  - Maximum baud rate $\frac{f_A}{16}$ (6.25 MBaud at 100 MHz f_A module clock)
  - Minimum baud rate $\frac{f_A}{268\,435\,456}$ (0.37 Baud at 100 MHz f_A module clock)
- Frame transmit on hardware or software trigger
- Circular buffer

Extended ASC features

- Programmable oversampling of 4 to 16 times per bit:
  - Module capability of up to 4 times higher baud rates ($\frac{f_A}{4}$) than the standard ASC
  - System considerations such as pad type, incoming signal quality, and accumulated PLL jitter can lead to constraints regarding the usable oversampling ratios (for example $\frac{f_A}{8} = \frac{200\,\text{MHz}}{8} = 25\,\text{MBaud}$)
- Optional RTS/CTS handshaking

LIN features

- Support of:
  - LIN version 1.3, 2.0, 2.1, and 2.2
  - J2602
- Break detection
- Break injection
- Sync field generation
- Auto baud detection based on Sync Field measurement
- Collision detection optionally (LIN2.1 mandatory)
- LIN Watchdogs:
  - Header time-out
  - Frame or Response time-out
- Stuck at zero or one monitoring
- Bus idle time monitoring
- Energy saving features (Wake-Up, Sleep)
- Minimum CPU load in master mode:
  - Single interrupt indicating the end of the frame
- Minimum CPU load in slave mode:
  - Single interrupt at the end of the header reception
  - Single interrupt at the end of the response or end of frame
- Standard operation with one interrupt per transmitted or received byte supported

SPI features

- SPI master modes (slave mode not supported):
  - Four-wire or three-wire (with or without slave select output signal)
- Up to 16-bit data width
- Full-duplex and half-duplex:
  - Maximum baud rate $\frac{f_A}{4}$ (= 25 MBaud at 100 MHz f_A module clock)
  - Minimum baud rate $\frac{f_A}{268\,435\,456}$ (= 0.37 Baud at 100 MHz f_A module clock)
- Programmable leading and trailing delays

Related information

References on page 4715

24.2 Functional overview

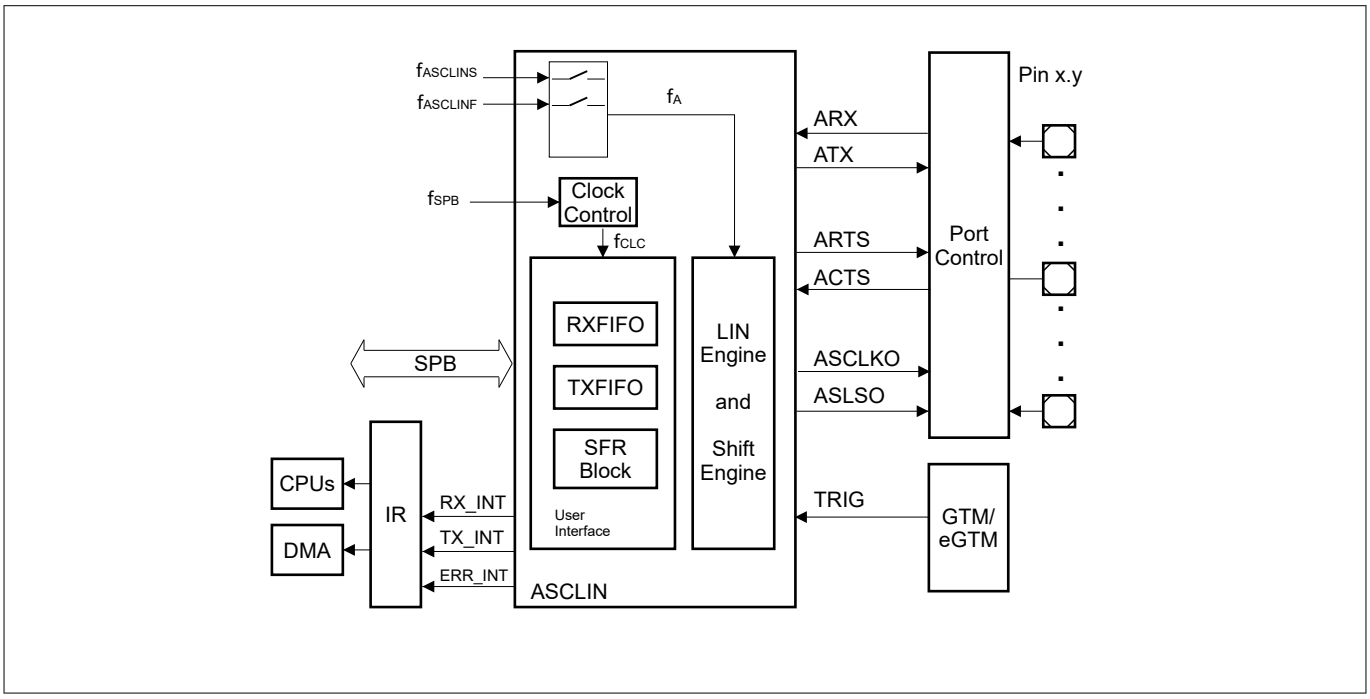


Figure 456 ASCLIN block diagram

The Shift Engine is responsible for frame transmission and reception, depending on the configured settings. For LIN communication the LIN Engine controls frame transmission and reception according to the protocol specification. This block runs on asynchronous clock.

The User Interface block is for configuration of protocol settings as well as the RxFIFO and TxFIFO and their management. The user can access the FIFOs using the registers. This block runs on synchronous clock.

Related information

TC4Dx SMU alarm mapping tables on page 7233

24.3 Functional description

This chapter provides the description of the module and it's configuration options for functional behavior.

24.3.1 Architecture overview

The essential sub-blocks and their interactions are shown in the following image.

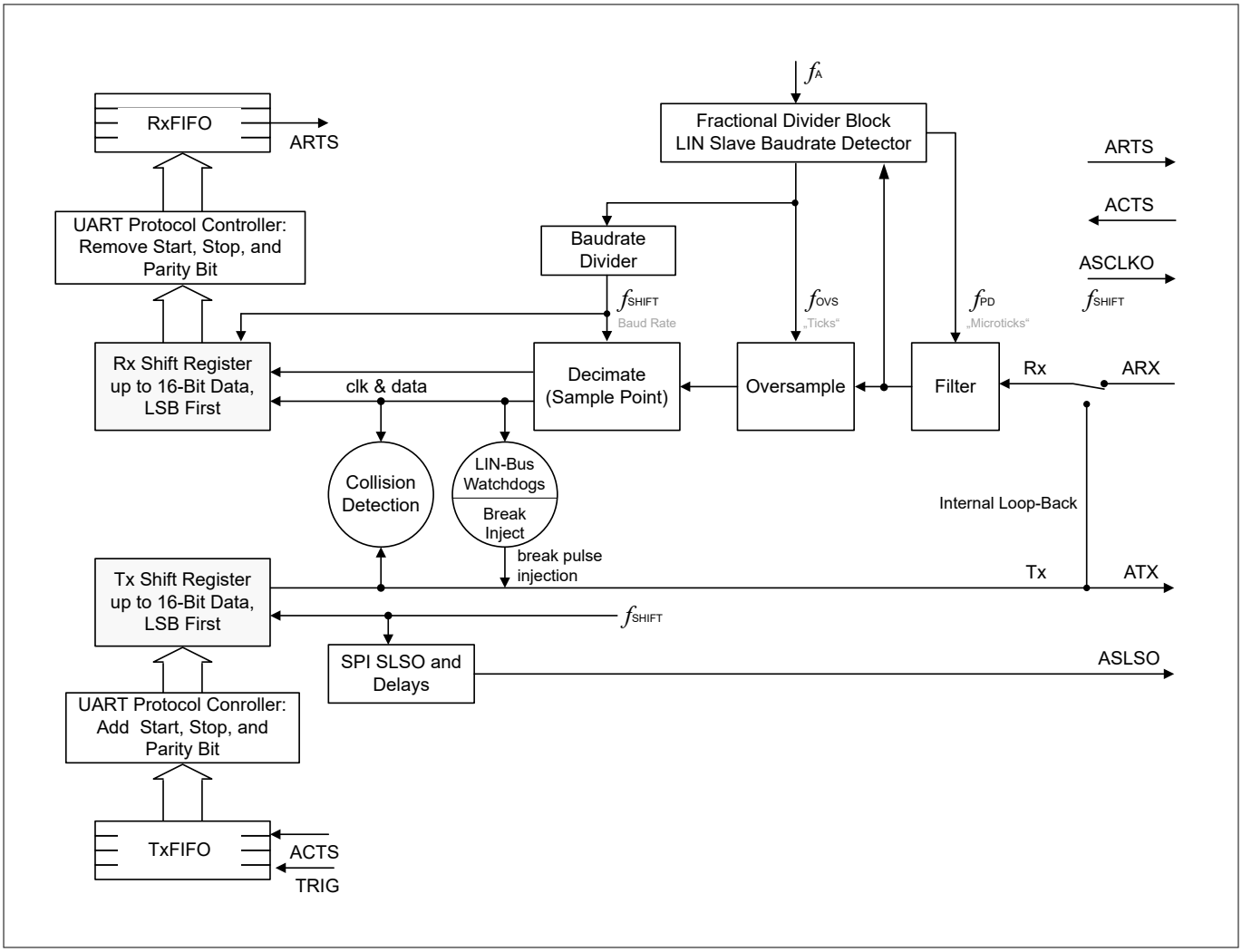


Figure 457 Architecture overview

Clocking blocks

- The Fractional Divider block and the Baudrate Divider block are used to provide the incoming and outgoing data baudrate
- The Filter, Oversample, and Sample Point blocks are used for optimizing the sampling of the incoming bit stream
- If enabled, the clocking blocks provide automatic baudrate detection in LIN slave mode

Timeouts and control blocks

- The LIN Bus Watchdogs and Break Inject are used for detecting the delays and translating them for the respective protocol
- The Collision Detection block is used to monitor the bus and report if there is any inconsistency with the outgoing bit-stream

Data receive or transmit blocks

- The RxFIFO and TxFIFO provide the user interface where the data is stored
- The protocol control machine is used to add or check protocol specific bits, for example parity bit, start bit, stop bit and shift the data

24.3.2 Supported modes

The ASCLIN module provides a number of external signals:

- ASCLKO - Serial clock output
- ARX - Receive data input (Master Receive MR in SPI mode)
- ATX - Transmit data output (Master Transmit MT in SPI mode)
- ASLSO - Slave select signal output
- ARTS - Request to send handshake output
- ACTS - Clear to send handshake input

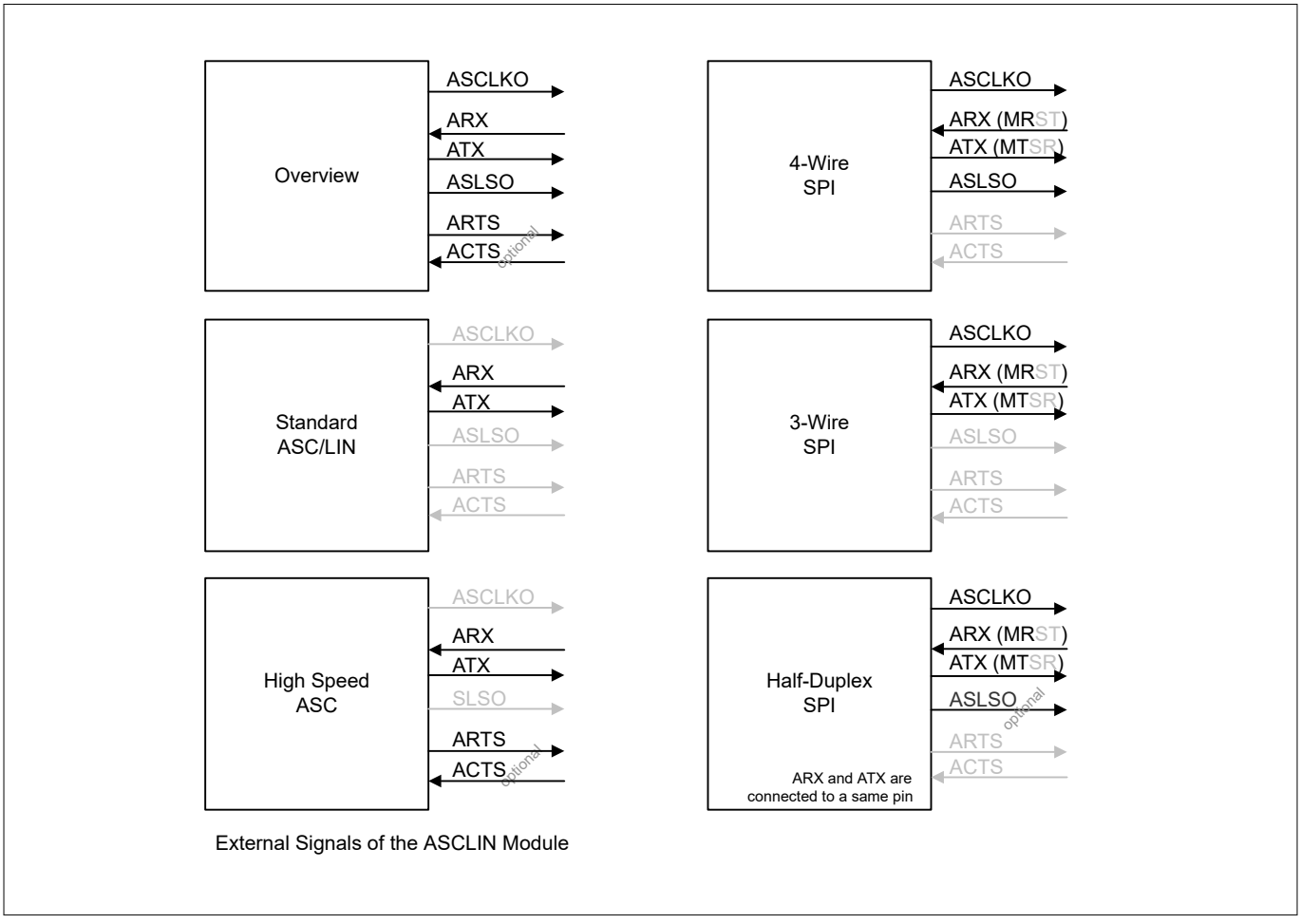


Figure 458 External signals of the ASCLIN in different modes

24.3.3 User interface

The user interface contains a TxFIFO and a RxFIFO that provide the services listed:

- Takes data packets with width optimal for the SPB bus and packs them to serial frames
- Buffers the SPB bus data
- Manages handshaking

These features are designed to optimize the use of the module in LIN, high-speed ASC, standard mode ASC, and SPI mode.

24.3.3.1 TxFIFO

The TxFIFO has the ability to pack:

- 16-bit writes from the SPB bus to two up-to-8-bit frames or one up-to-16-bit frame
- 32-bit writes from the SPB bus to four up-to-8-bit or two up-to-16-bit frames

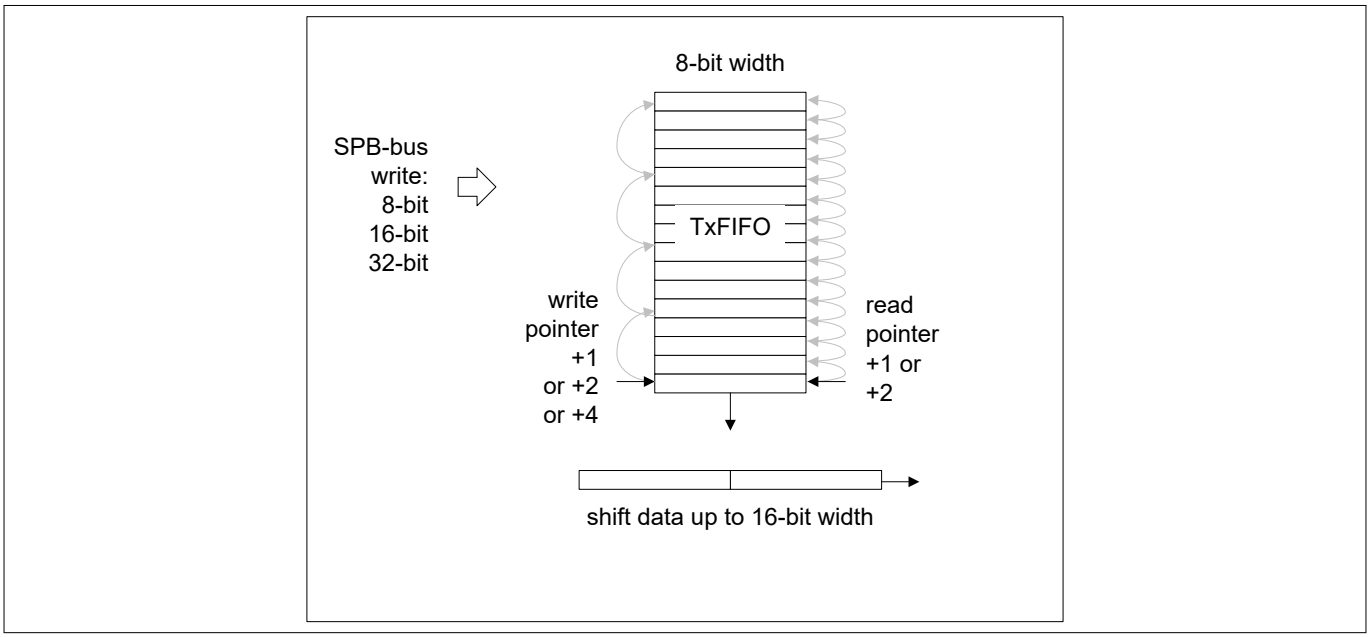


Figure 459 TxFIFO overview

The number of bytes written to the TxFIFO is always defined with the TXFIFOCON.INW register, here after referred to as Tx_Inlet_Width, and does not depend on the SPB write width. For proper functionality the SPB write width should be equal or shorter than Tx_Inlet_Width. If the SPB write width is shorter than Tx_Inlet_Width, then the missing part is padded with zeros. If the SPB-bus write width is longer then Tx_Inlet_Width, then the excessive part is lost.

On the shift register side, the TxFIFO is always emptied by a number of bytes as needed for the data width field DATCON.DATLEN register, referred to as Tx_Outlet_Width.

The table below describes all the possible write (fill) combinations. The bytes which will be filled into the TxFIFO are labeled as A, B, C, D, and 0.

Table 1113 Inlet width versus SPB write width

TxFIFO contents		SPB bus write width		
		8-bit A	**16-bit BA**	**32-bit DCBA**
Tx_Inlet_Width	8-bit	A_H	A_H	A_H
	16-bit	$0A_H$	BA_H	BA_H
	32-bit	$000A_H$	$00BA_H$	$DCBA_H$

24.3.3.1.1 Using the TxFIFO

The transmit data is written to the address TXDATA registers. Writing to any of these registers has the same effect as writing to TXDATA. The eight consecutive TXDATA registers are provided to support burst access.

Standard ASC mode

The most standard usage of the ASC module is to transmit 7 or 8-bit values, and to fill the FIFO with 8-bit wide SPB-bus accesses. The SPB bus access is 8-bit wide, and there is only BS0 (Byte Select 0) signal active.

The Tx_Inlet_Width is 8-bit and the Tx_Outlet_Width is 8-bit.

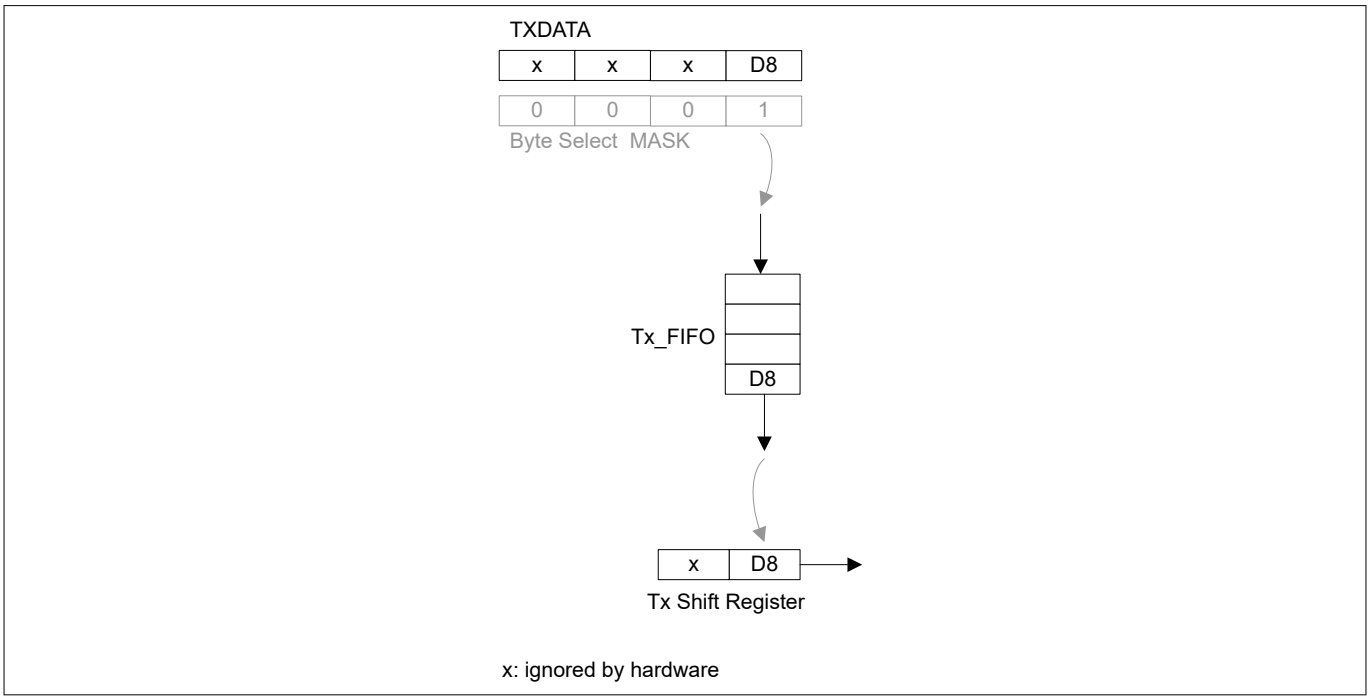


Figure 460 FIFO operation in 7- or 8-Bit ASC mode, with or without parity

A less common mode of operation of the ASC module is to transmit 9-bit values, containing either 9-bit data or 8-bit data. The SPB bus access should be 16-bit wide, and there are BS0 and BS1 (Byte Select 0 and 1) signals active. The shift register is filled with two 8-bit values at the location "read pointer" and "read pointer + 1".

The Tx_Inlet_Width is 16-bit and the Tx_Outlet_Width is 16-bit.

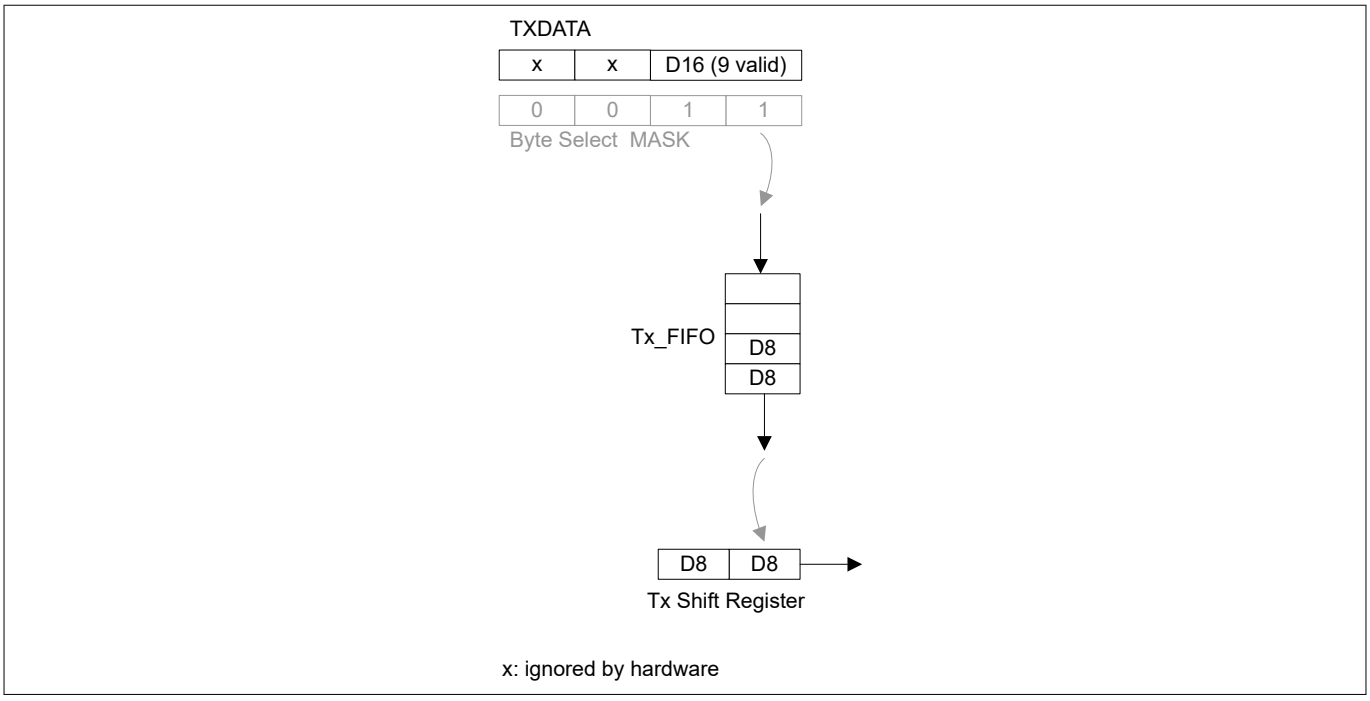


Figure 461 FIFO operation in 9-Bit ASC mode

A mode exclusive to the ASC is the circular buffer, which is selected from the TXFIFOCON.FM. The configuration settings, with the Tx_Inlet_Width and the Tx_Outlet_Width are similar to the previous description. The circulating level is defined in the configuration of the FIFO. The shifted data stays in the FIFO, only the "get-pointer" is updated. After all data, up to the circulating level, is shifted out, the pointer is reset to the first data in FIFO, thus making it a circulating FIFO. This mode is only compatible with hardware transmit trigger or software transmit trigger, but is not compatible with the traditional transmit mode.

High-speed ASC mode

In order to reduce the SPB bus load by not using a 32-bit move for 7 or 8-bit values, one option is to fill the TxFIFO with 32-bit wide accesses containing four 8-bit wide values. All Byte Select signals BS[3:0] are active, and all four bytes are written in the TxFIFO in one cycle. At the same time, the "write pointer" of the TxFIFO is incremented by four.

The Tx_Inlet_Width is 32-bit and the Tx_Outlet_Width is 8-bit.

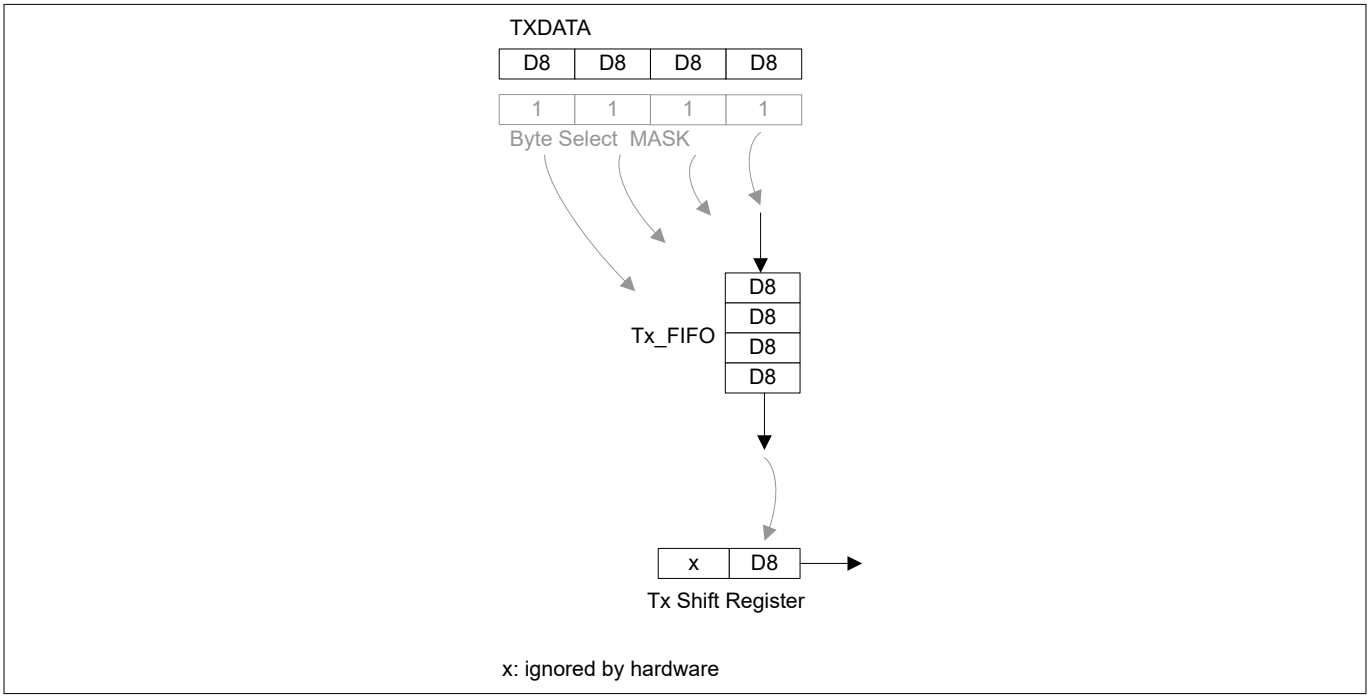


Figure 462 High-speed communication scenario

LIN mode

In LIN mode, several 8-bit frames are preceded by a low break pulse. The generation and detection of the break pulse is programmable. For more details please refer to LIN support.

The Tx_Inlet_Width can be 8-bit (or 16-bit, or 32-bit) and the Tx_Outlet_Width is 8-bit.

SPI mode

SPI mode is used most often to send or receive data of 8 up to 16-bits length.

The Tx_Inlet_Width is up to 32-bit and the Tx_Outlet_Width is up to 16-bit.

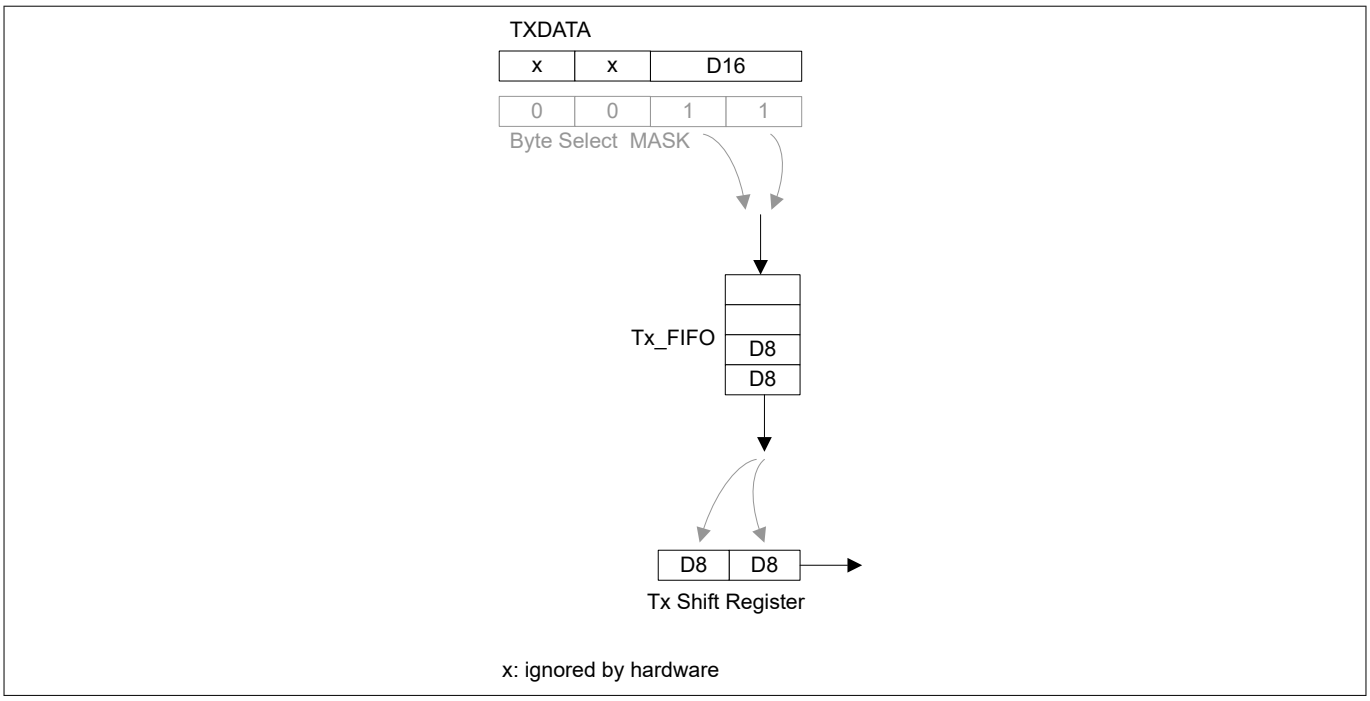


Figure 463 TxFIFO operation in SPI mode

24.3.3.1.2 TxFIFO interrupt generation

The TxFIFO provides four interrupt generation modes, selected by the bit-field TXFIFOCON.FM:

- Single Move Mode
- Batch Move Mode
- Combined Mode
- Circular Buffer Mode

The RxFIFO and the TxFIFO can use the interrupt generation modes independently of each other.

The TxFIFO interrupt flag is FLAGS.TFL. In case of an event, if the flag interrupt is enabled in FLAGSENABLE.TFLE an interrupt is generated.

Single move mode

The purpose of the Single Move Mode is to keep the TxFIFO as full as possible, refilling the TxFIFO by writing to it as soon as there is a free element.

The single move mode supports primarily a DMA operation using single move per TxFIFO interrupt. In this mode the DMA keeps the TxFIFO full. A DMA request is triggered each time a write to the TxFIFO is performed, and there is at least one empty element available.

If the write:

- Makes the TxFIFO full (no more empty elements), or
- The next write would make the TxFIFO overflow taking TXFIFOCON.INW into account

an interrupt is not generated in order to prevent overflow. It is generated later when the shift register reads one element from the TxFIFO, making it sufficiently empty to accept one Tx_Inlet_Width wide element. If TxFIFO can handle new data, it generates an interrupt but expects just one data of the defined frame width. The DMA or the user should not write multiple data at once to avoid unexpected behavior.

The following figure shows the filling levels of the TxFIFO, and the events associated with each filling level triggering an interrupt (FLAGS.TFL).

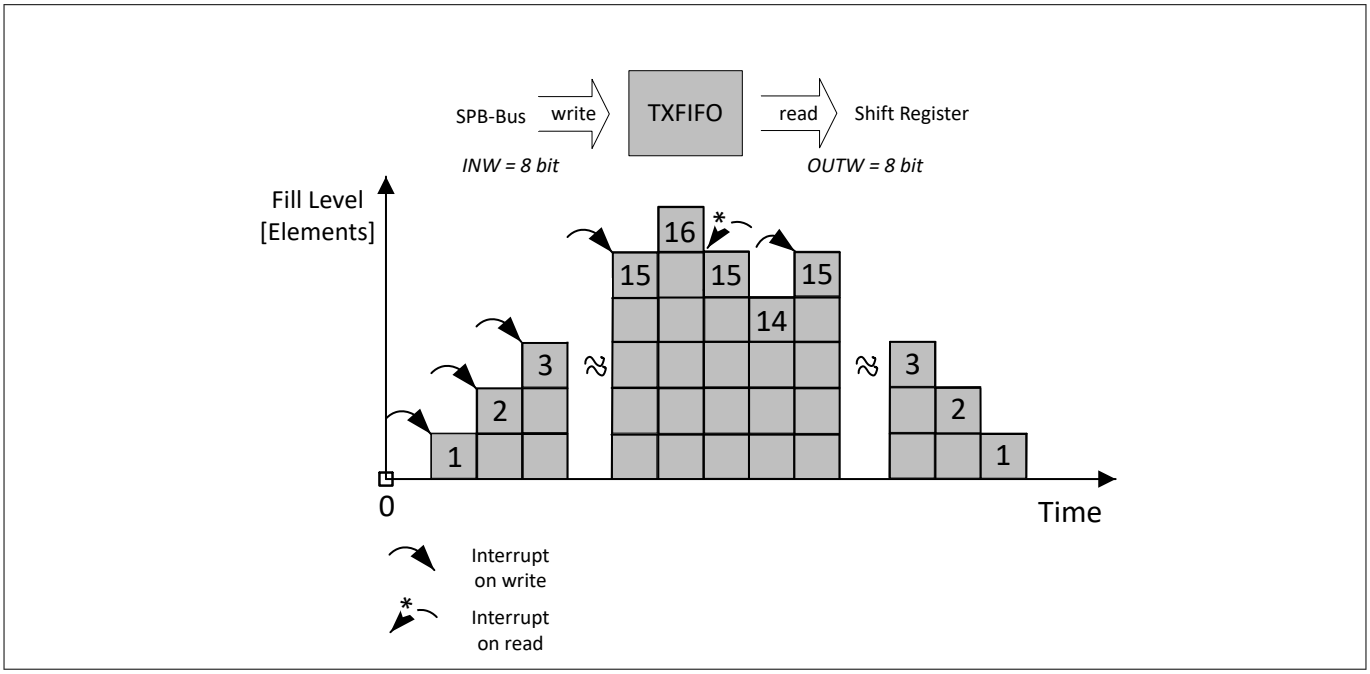


Figure 464 Interrupt generation in the single move mode

In order to initiate the very first interrupt in the chain of the refill interrupts after power-on reset, it is necessary that the software either performs one write to the TxFIFO or by setting the interrupt flag in the TxFIFO interrupt node. Afterwards, the (DMA) interrupt-chain is self sustaining until the whole transaction is over.

In Single Move Mode, when the TxFIFO is almost full (the next write operation causes the TxFIFO full) or when the TxFIFO is full, a flush on TxFIFO will trigger an interrupt (FLAGS.TFL).

Attention: ***In Single Move Mode multiple software writes or block DMA moves would lead to multiple interrupts and (false) transaction lost events. Therefore they should be avoided - only single moves should be used.***

Batch move mode

Batch Move Mode supports primarily the CPU servicing the TxFIFO use case.

The purpose of the Batch Move Mode is to reduce the number of interrupts by triggering an interrupt when more then one TxFIFO elements are empty. For example, a CPU can be interrupted less frequently and it can perform more than one moves per interrupt.

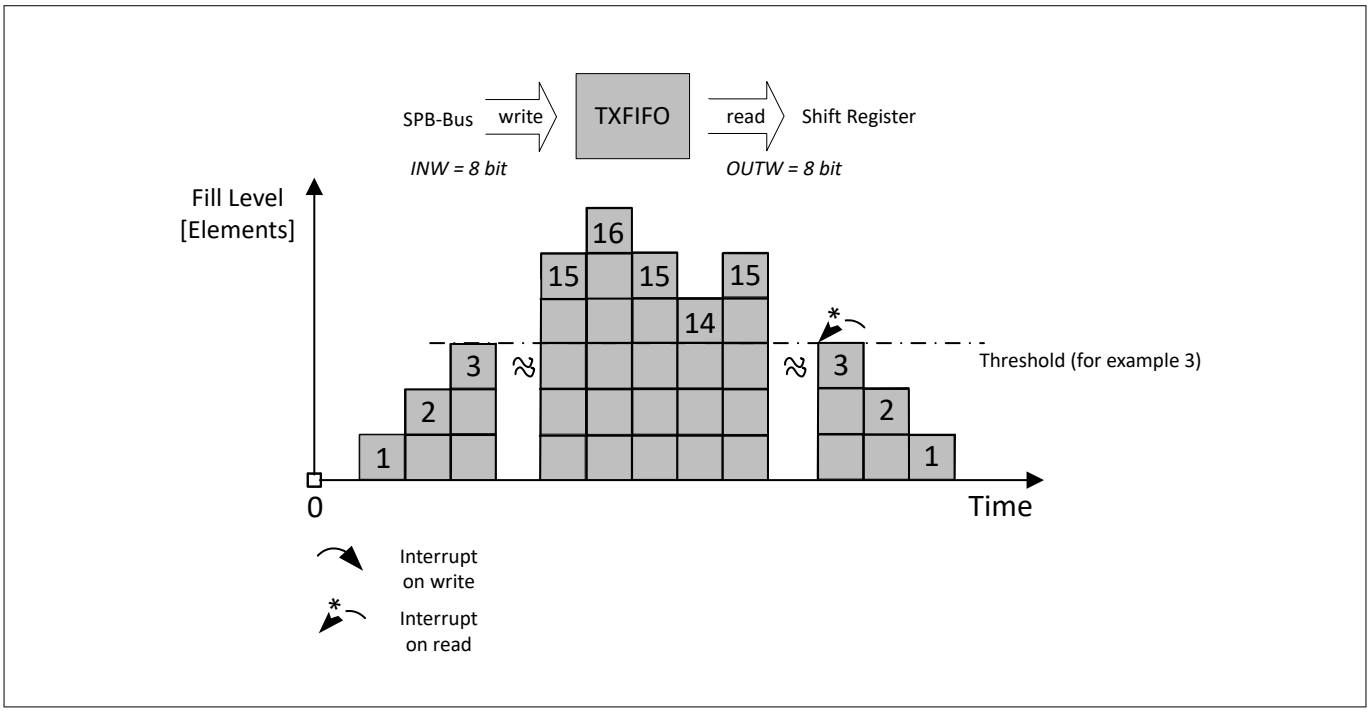


Figure 465 TxFIFO - interrupt generation in the batch move mode

In Batch Move Mode, an interrupt is generated only at one point in the TxFIFO, when the filling level falls to or below the programmed threshold, as configured in TXFIFOCON.INTLEVEL, implying that there are at least the predefined number of empty TxFIFO elements available.

Attention: ***The TxFIFO must be refilled until the filling level has risen above the interrupt threshold (ensured by polling the filling level), in order to guarantee that the next interrupt will occur.***

At high baud rates and short frames, it could be possible that a couple of frames have been transmitted until the moment the TxFIFO is refilled. For example, if the threshold is set to three full elements, then the interrupt will be set when the filling level falls below three, making two elements free. If the refill moment of the TxFIFO is delayed so long that two frames have been transmitted, the TxFIFO will become empty, and refill sequence of two will not reach the level of three. So, the next interrupt would never come unless the filling level is being polled and the CPU keeps refilling in a loop until the threshold has been passed (ideally the TxFIFO would be filled completely with each batch of moves).

Combined (compatibility) mode

An interrupt is generated when the filling level falls to INTLEVEL or below, each time when a data byte is taken out of the FIFO, as defined in the TXFIFOCON.INTLEVEL. This is the mode which is compatible with the previous versions of the ASCLIN module and described in the TXFIFOCON.INTLEVEL bit-field.

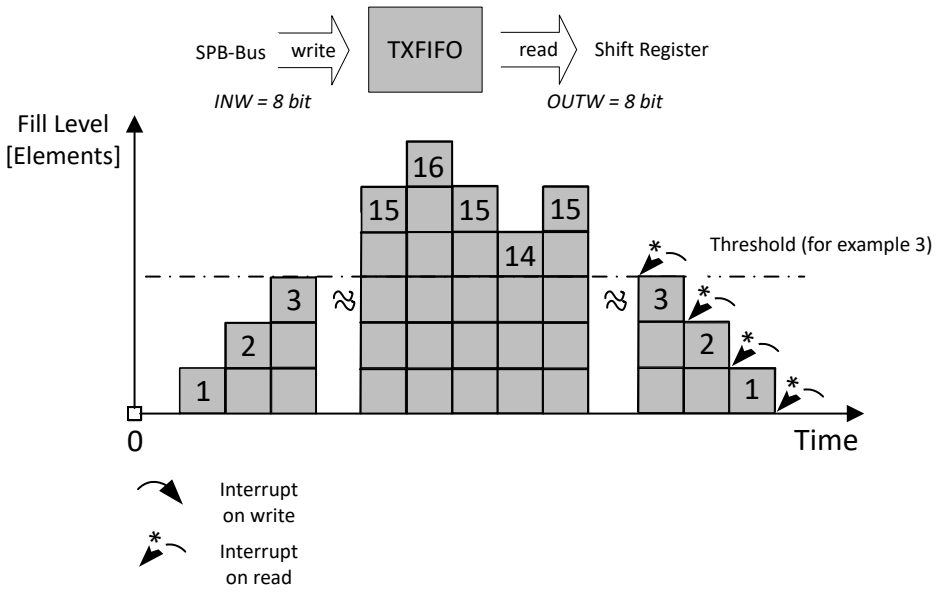


Figure 466 TxFIFO - interrupt generation in the combined mode

Circular buffer mode

This is a special mode used only when the ASCLIN is configured to run in ASC mode and the FIFO mode is circular buffer. This mode is only compatible with the hardware or software trigger transmit mode, thus an interrupt is not needed for each transmitted frame. In this mode, if enabled from FLAGSENABLE register, an interrupt in FLAGS.TFL is generated each time a full circulation is done. A full circulation means transferring of FIFO elements from the first one up to the circulation level.

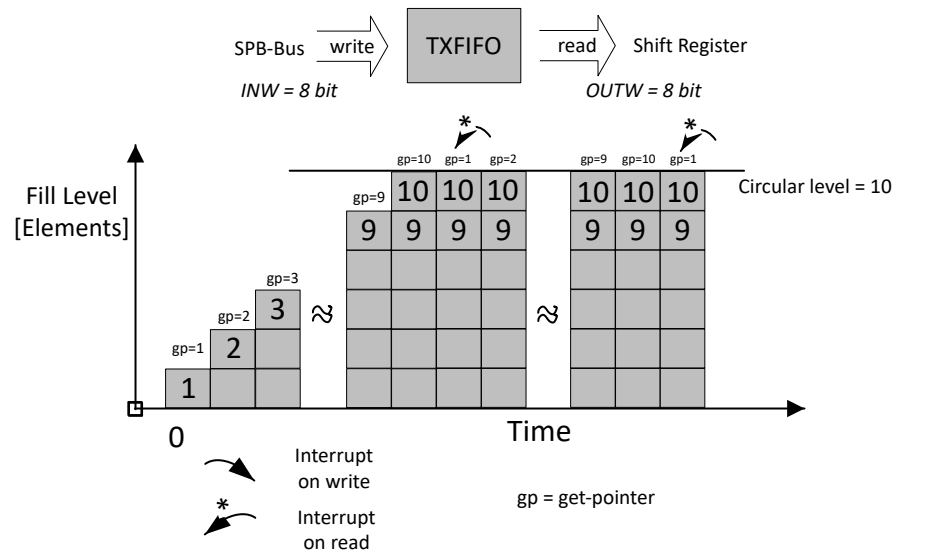


Figure 467 TxFIFO - interrupt generation in the circular buffer mode

24.3.3.2 RxFIFO

The RxFIFO has the ability to pack:

- Two up-to-8-bit frames or one up-to-16-bit frame to one 16-bit write to the SPB bus
- Four up-to-8-bit or two up-to-16-bit frames to one 32-bit write to the SPB bus

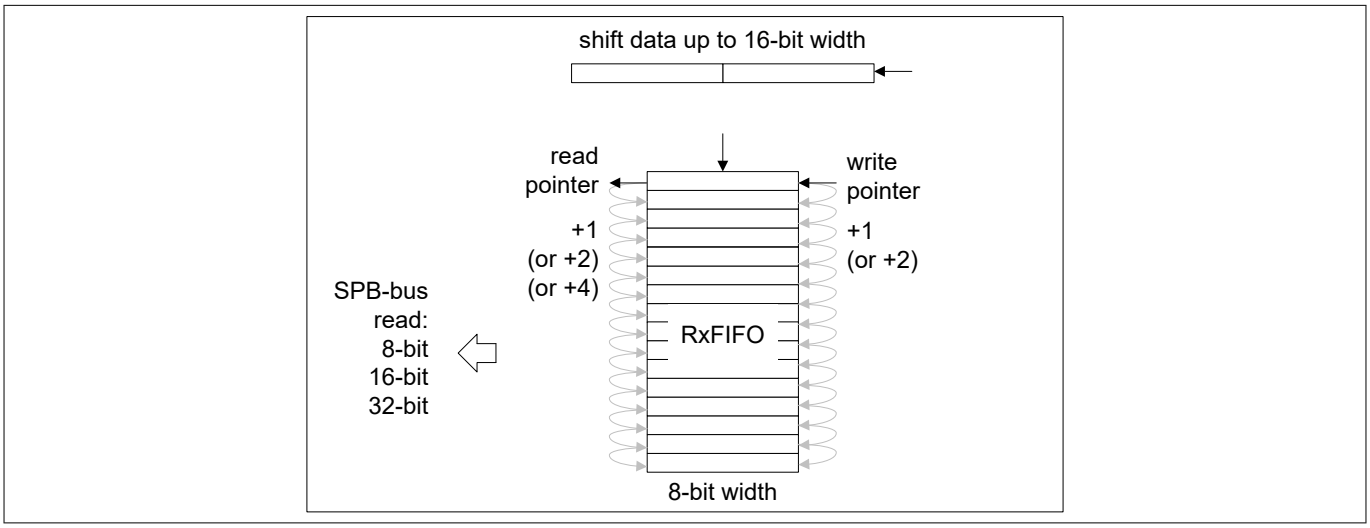


Figure 468 RxFIFO overview

The number of bytes taken out of the RxFIFO is always RXFIFOCON.OUTW, hereinafter referred as Rx_Outlet_Width and it does not depend on the SPB bus read width. For proper functionality the SPB read width should be equal or shorter than Rx_Outlet_Width.

If the SPB read width is shorter than the Rx_Outlet_Width, then the excessive part is lost.

If the SPB read width is longer then Rx_Outlet_Width, then the missing part is padded with zeros.

On the shift register side, the RxFIFO is always filled with a number of bytes as needed for the data width field DATCON.DATLEN, hereinafter referred as Rx_Inlet_Width.

The table below describes all the possible combinations. The bytes in the RxFIFO are named A, B, C and D. A being the first one to take out. The padding byte is 0.

***Note**: In case of an underflow, that is when a read access is performed to RXDATA and the number of bytes to be taken out of the FIFO as given by the setting of RXFIFOCON.OUTW exceeds the number of bytes that are actually stored in the FIFO, as indicated by RXFIFOCON.FILL, then the RxFIFO delivers the available data padded with zeros up to the read access width. However, the data remain in the FIFO and the filling level RXFIFOCON.FILL is not changed. To remove the data, the user software must flush the RxFIFO. The bit FLAGS.RFU is set.*

Table 1114 Outlet width versus SPB read width

SPB-read data, taken from the RxFIFO		SPB bus read width		
		8-bit	**16-bit**	**32-bit**
Rx_Outlet_Width	8-bit	A_H	$0A_H$	$000A_H$
	16-bit	A_H	BA_H	$00BA_H$
	32-bit	A_H	BA_H	$DCBA_H$

24.3.3.2.1 Using the RxFIFO

The received data is read from the address RXDATA registers. Reading from any of these registers has the same effect. The eight consecutive RX registers are provided to support burst access.

Standard ASC mode

In case of data width of 7 or 8 bits, one read on this address delivers one byte and empties the FIFO for one element, if the Rx_Outlet_Width is 8-bit. The Rx_Inlet_Width is 8-bit.

In case of data width of 9 bits, one read on this address delivers two bytes and empties the FIFO for two elements, if the Rx_Outlet_Width is 16-bit. The Rx_Inlet_Width is 16-bit.

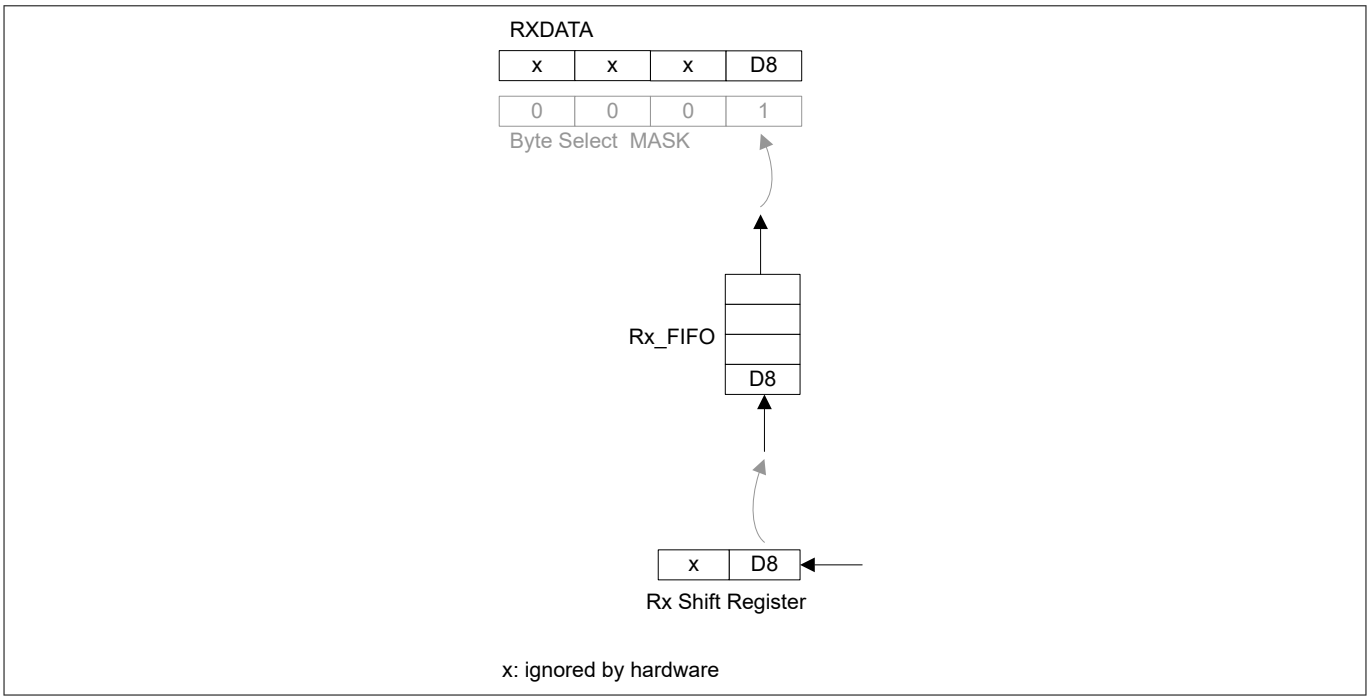


Figure 469 RxFIFO operation in 7- or 8-Bit ASC mode, with or without parity

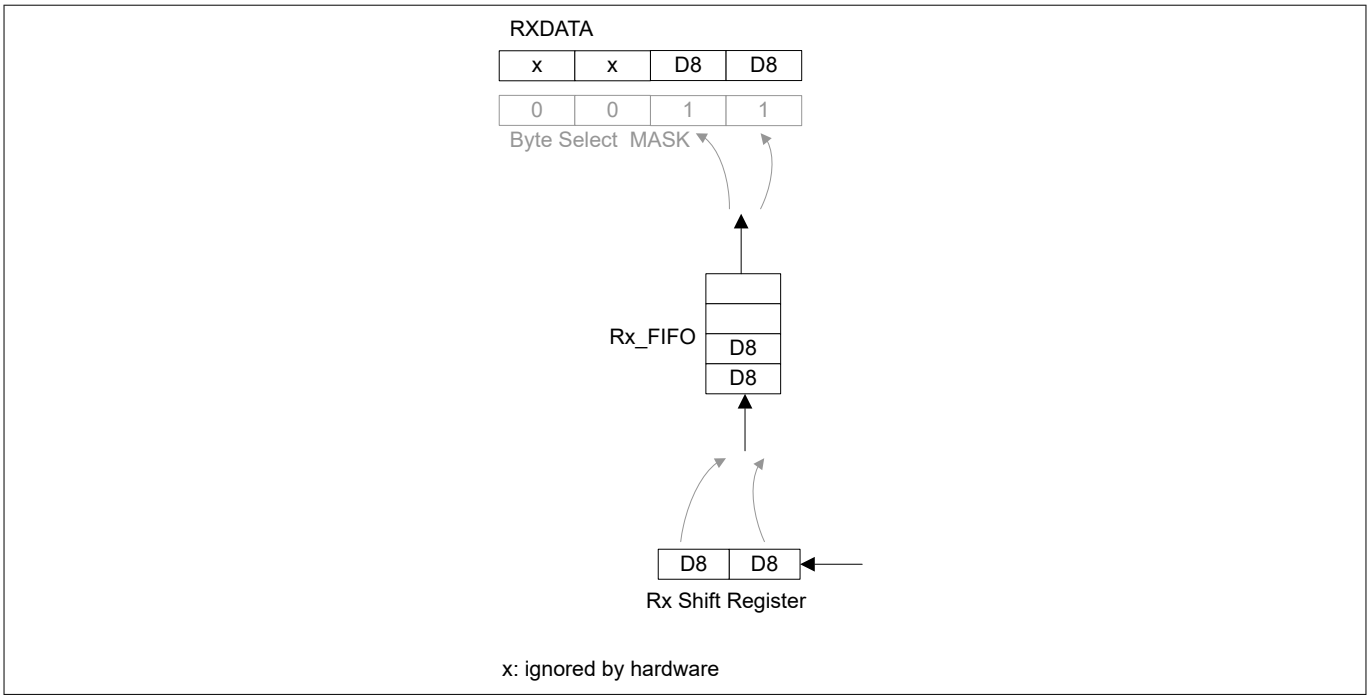


Figure 470 RxFIFO operation in 9-Bit ASC mode

High speed ASC mode

In case of data width of 7 or 8 bits, one read on this address can deliver (one or two or) four bytes and empties the FIFO by four elements, if the Rx_Outlet_Width is 32-bit. The Rx_Inlet_Width is 8-bit.

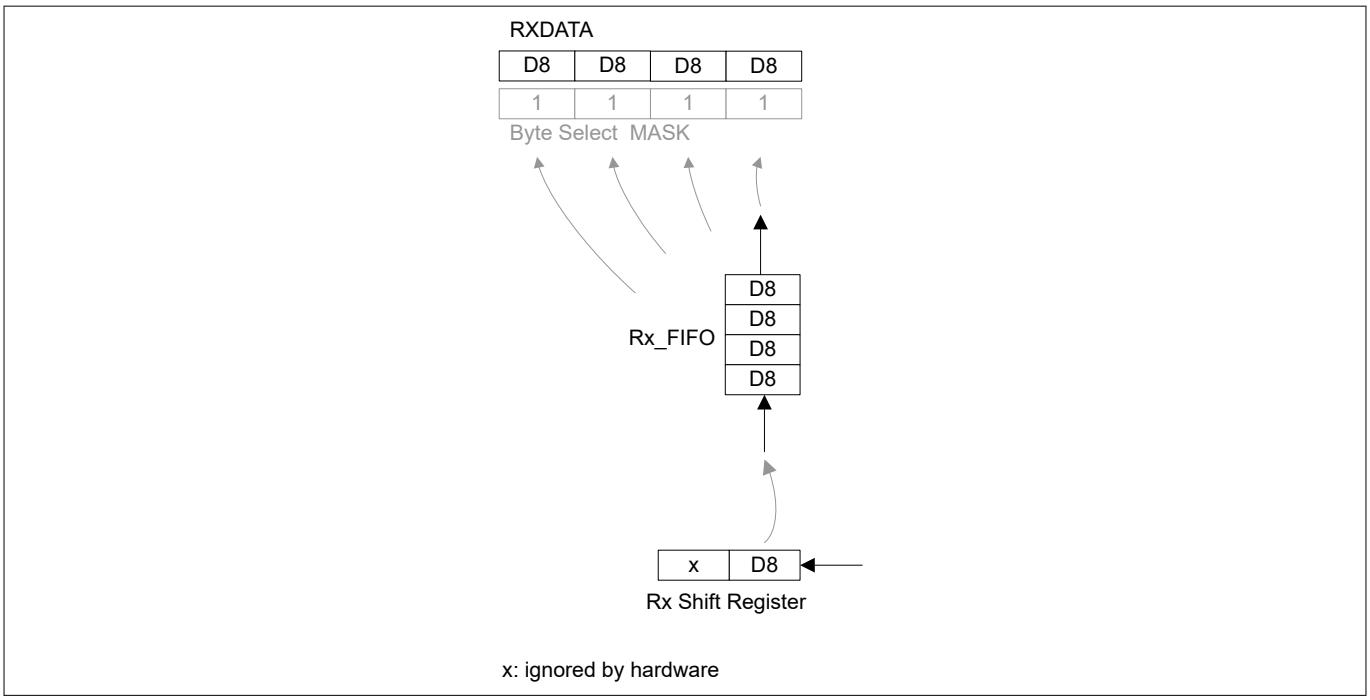


Figure 471 RxFIFO operation in high speed ASC mode

LIN mode

The read data width is 8 bits (or 16-bit or 32-bit), one read on this address delivers one byte and empties the FIFO by one element, if the Rx_Outlet_Width is 8-bit. The Rx_Inlet_Width is 8-bit.

The RxFIFO operation in LIN mode is different when using it in the slave or master mode. In slave mode first the sync break is detected. After the sync field reception the ENI bit is set by hardware to enable the transfer of the ID byte into FIFO. If response is expected, selectable from HO, then also the response is transferred into FIFO. Last field is checksum, if enabled in CSI field, this byte is transferred into FIFO.

In master mode the ENI is set by software and if enabled the response is expected from slave and it allows its transfer into FIFO. The checksum field is stored similarly to the slave mode.

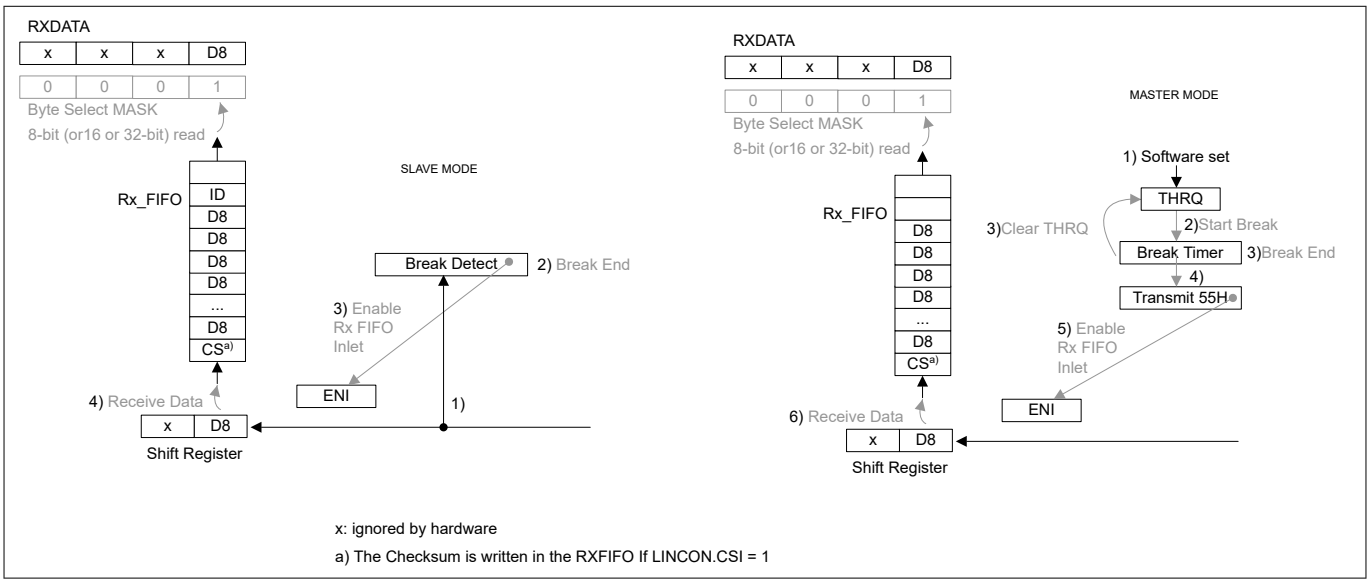


Figure 472 RxFIFO operation in LIN mode

SPI mode

In case of data width of 16 bits, one read on this address delivers two bytes and empties the FIFO by two elements, if the Rx_Outlet_Width is 16-bit. The Rx_Inlet_Width is 16-bit.

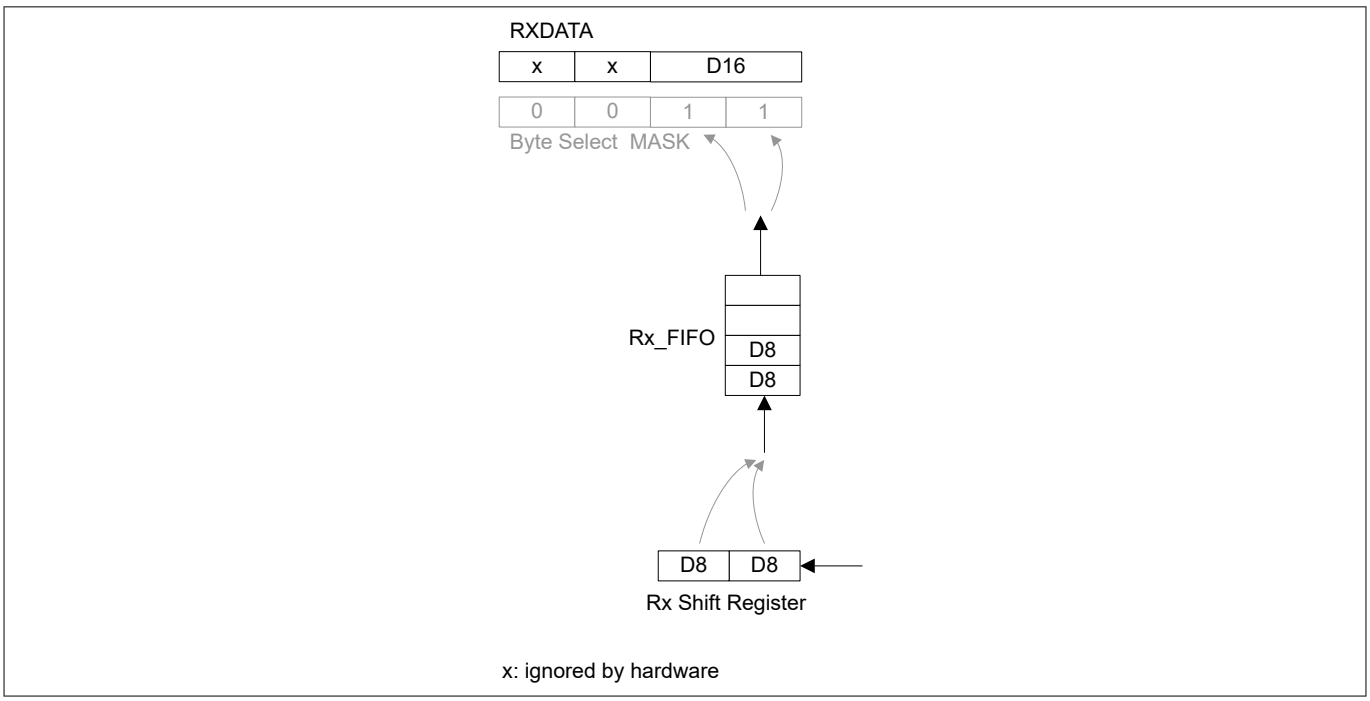


Figure 473 RxFIFO operation in SPI mode

24.3.3.2.2 RxFIFO interrupt generation

The RxFIFO provides three interrupt generation modes, selected by the bit-field RXFIFOCON.FM:

- Single Move Mode
- Batch Move Mode
- Combined (Compatibility) Mode

The RxFIFO interrupt flag is FLAGS.RFL. In case of an event (irrelevant if the event source is from read or write to the FIFO), if the flag interrupt is enabled in FLAGSENABLE.RFLE, an interrupt is generated.

Single move mode

The purpose of the Single Move Mode is to keep the RxFIFO as empty as possible, by fetching the received elements one by one as soon as possible.

The single move mode supports primarily a DMA operation using single move per RxFIFO interrupt. In this mode the DMA keeps the RxFIFO as empty as possible. A DMA request is triggered each time a read from the RxFIFO is performed, and the RxFIFO is afterwards still not empty.

If the read:

- Makes the RxFIFO empty, or
- The next read would make the RxFIFO underflow taking RXFIFOCON.OUTW into account

an interrupt is not generated in order to prevent underflow. It is generated later when the shift register writes one element to the RxFIFO, making it sufficiently full to provide one Rx_Outlet_Width wide element. If RxFIFO can handle new data, it generates an interrupt but fetches just one data of the defined frame width. The DMA or the user should not read multiple data at once to avoid unexpected behavior.

The figure below shows the filling levels of the RxFIFO, and the events associated with each filling level triggering a RxFIFO refill interrupt.

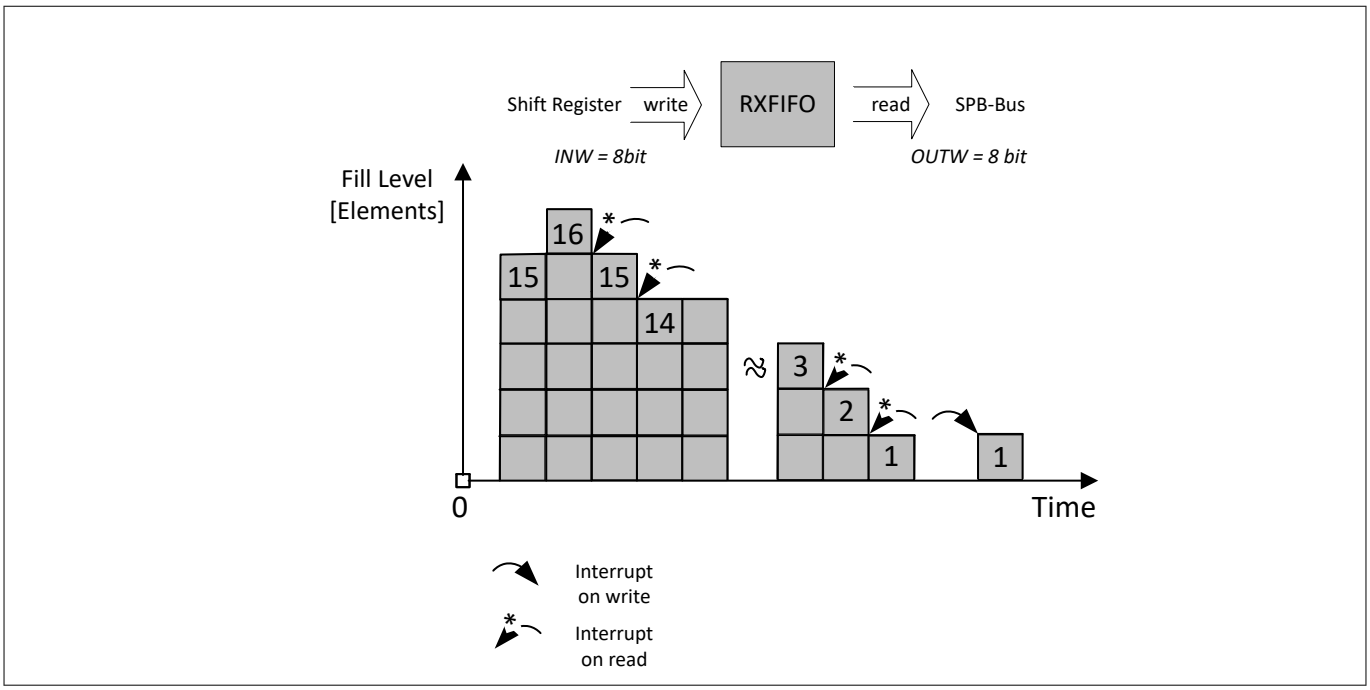


Figure 474 RxFIFO - interrupt triggering in the single move mode

The initial RxFIFO interrupt is triggered by the Shift Register, after it delivers the first received element. Afterwards, the DMA trigger-chain of re-fetch interrupts is self sustaining until the whole transaction is over. At the end of the DMA transaction, there is no service request remaining active in the service request node, due to the fact that the read of the last element in the RxFIFO does not trigger an interrupt.

Attention**: **In Single Move Mode multiple software reads or block DMA moves lead to multiple interrupts and (false) transaction lost events. Therefore they should be avoided - only single moves should be used.

Batch move mode

The purpose of the Batch Move Mode is to reduce the number of interrupts by triggering an interrupt when more then one RxFIFO elements are full. For example, a CPU can be interrupted less frequently and it can perform more than one move per interrupt.

Batch Move Mode supports primarily the CPU servicing the RxFIFO use case.

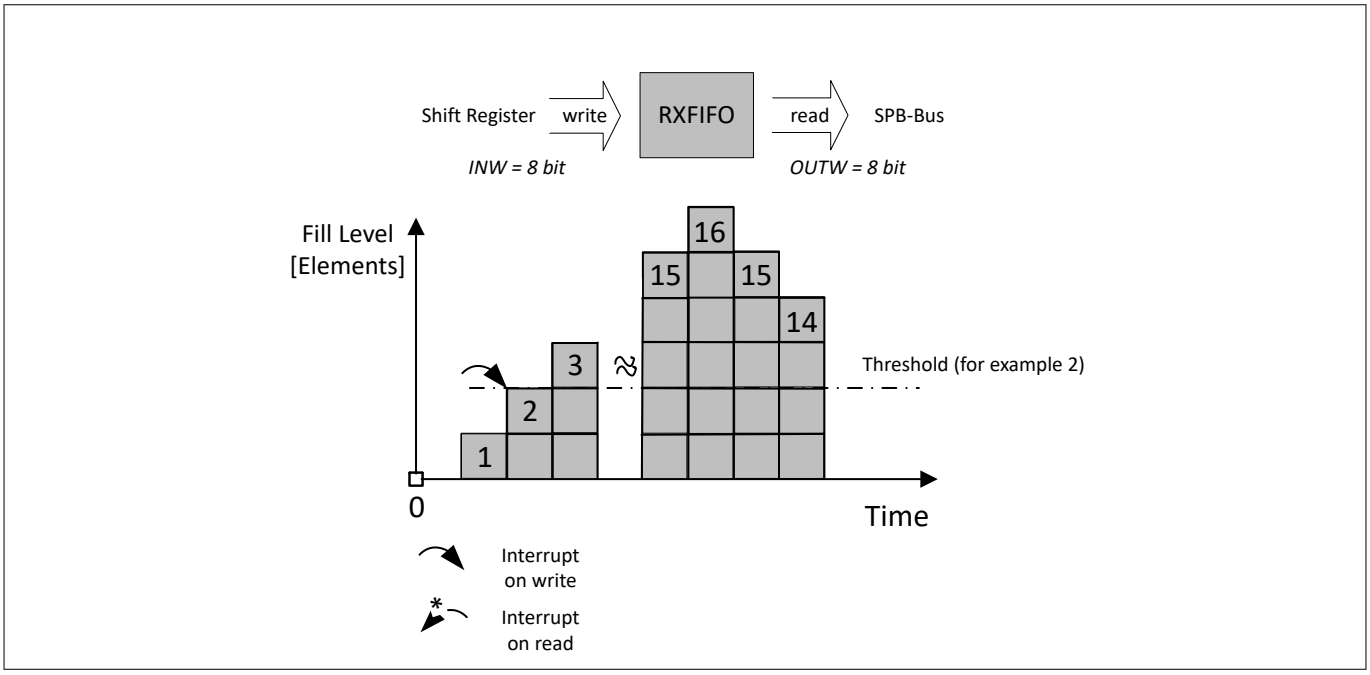


Figure 475 RxFIFO - interrupt triggering in the batch move mode

In Batch Move Mode, an interrupt is generated only at one point in the RxFIFO, when the filling level rises to or above the programmed threshold, implying that there are at least the predefined number of full RxFIFO elements available. The description refers to the interpreted value of RXFIFOCON.INTLEVEL (register value +1).

Attention**: **It must be guaranteed that the CPU keeps emptying the RxFIFO and polling the RxFIFO filling level until the filling level has fallen below the interrupt threshold (or the RxFIFO is empty), in order to guarantee that next interrupt will occur.

At high baud rates and short frames, it could be possible that more than one frames have been received until the moment the RxFIFO is re-emptied. For example, if the threshold is set to two full elements, then the interrupt will be set when the filling level rises above two. If the emptying moment of the RxFIFO is delayed so long that two additional frames have been received, the RxFIFO will become full, and re-empty sequence of two will not reach the level of two. So, the next interrupt would never come. This effect can not occur with threshold levels of 3 and 4. The threshold level of 1 is possible, but does not make much sense (use single move mode instead).

Combined (compatibility) mode

An interrupt is generated when the filling level rises to INTLEVEL or beyond, each time when a data byte is delivered to the FIFO, as defined in the RXFIFOCON.INTLEVEL. The description refers to the interpreted value of RXFIFOCON.INTLEVEL (register value +1). This is the mode which is compatible with the previous versions of the module and described in the RXFIFOCON.INTLEVEL bit-field.

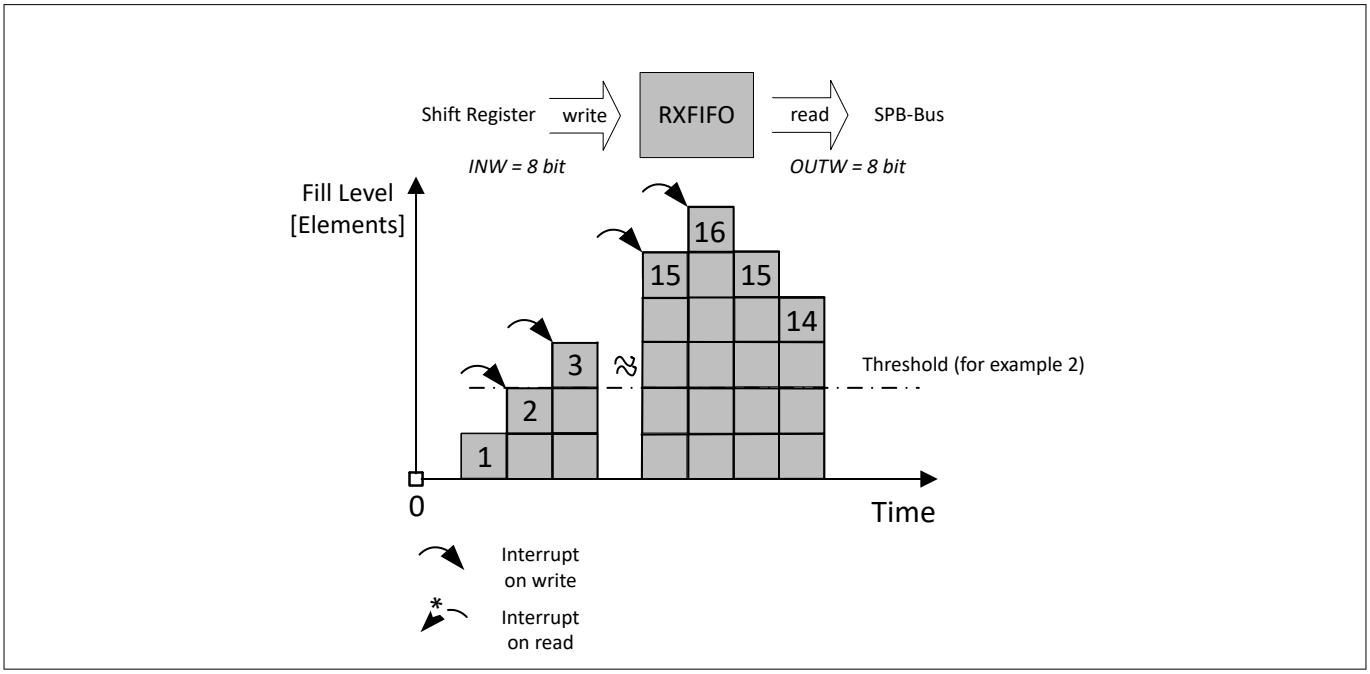


Figure 476 RXFIFO - interrupt triggering in the combined mode

24.3.3.3 RTS/CTS handshaking

The handshaking is a hardware mechanism to control the flow of communication in the ASC and High Speed ASC modes. It uses two additional signals RTS (Request to Send) and CTS (Clear to Send).

The receiver deactivates the RTS output signal when the RxFIFO is almost full, in order to avoid its overload. When the emptying of the RxFIFO starts, for example by a DMA, and the filling level falls below the threshold level, the RTS output is activated again. The transmitter receives the RTS output of the receiver on its CTS input and accordingly pauses and resumes the transmission.

The Handshaking is enabled by setting the IOCR.CTSEN bit. The threshold level is fixed to RxFIFO size minus four, in order to support byte, two bytes and four bytes DMA transfers.

ASCLIN connection with itself

The following connection is defined for each ASCLIN instance:

ARTS -> ACTSD

This connection can be useful in the SPI mode, where frequently the transmission and the reception of data are performed in parallel. If this connection is selected by using IOCR.CTS register, the TXFIFO will deliver data for transmission only if there is a free space in the RXFIFO. If not, the TXFIFO will wait for emptying of the RXFIFO by software and then automatically continue the transmission.

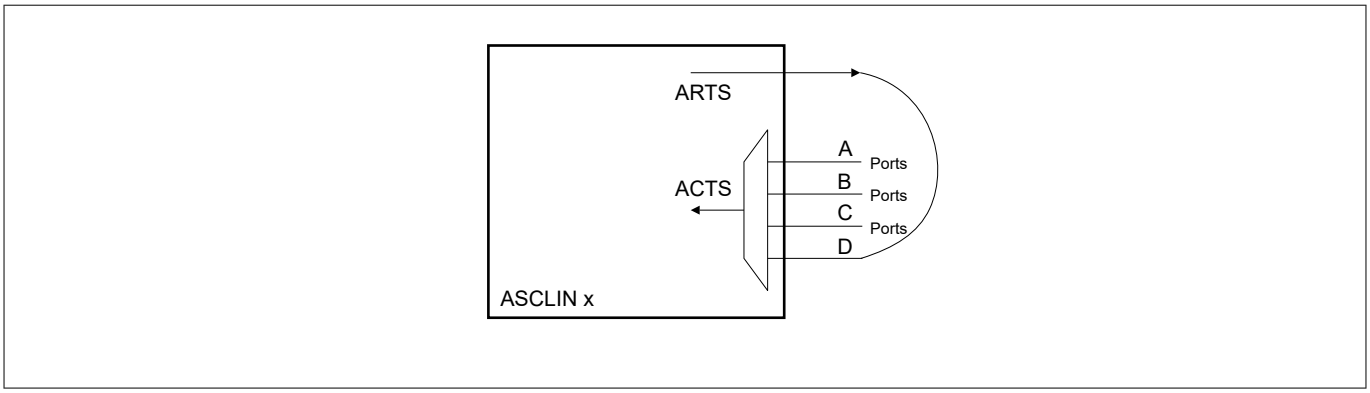


Figure 477 ASCLIN ARTS to ACTS Connection

24.3.3.4 Transmit trigger

This feature is only applicable when the ASCLIN module is configured in ASC mode.

In addition to the traditional mode, to transmit the frame once there is a full frame in TXFIFO, the ASCLIN can send frames in two additional advanced modes. The new modes are:

- Software-triggered frame transmission. It is enabled in DATCON.TRGSRC. To transmit a frame, the FLAGSSET.SWTRG bit shall be set
- Hardware-triggered frame transmission. It is enabled in the DATCON.TRGSRC bit-field, where the user selects thetimer source: GTM or eGTM for TRIG trigger input. The channel and frequency selection is configured in GTM or eGTM module
  - This mode offers transmission of frames in synchronous fashion

In case a Transmit Trigger is received but there is no data or not enough data for a full frame in the TxFIFO, the FLAGS.TFE flag is set, and an interrupt is generated, if enabled.

In case during ongoing transmission another trigger happens, this trigger is ignored and the overflow of multiple triggers (FLAGS.OMT) flag is set. The ongoing transmission is finished.

24.3.4 Clock system

The clock system generates all the clocks needed for the proper operation of the module: digital filter clock, oversampling clock, bit time and serial SPI clock.

The clock used for the clock system f_A is independent from the SPB bus clock and remains constant if the SPB bus clock changes, for example in power saving scenarios. The bit-field CSR.CLKSEL selects the clock source for the f_A frequency. There are two sources for f_A clock namely, $f_{ASCLINF}$ and $f_{ASCLINS}$. $f_{ASCLINF}$ is a fast clock typically used for SPI and extended ASC mode and $f_{ASCLINS}$ is slow clock, normally used for normal ASC mode. LIN baud rates can be derived from both clocks.

The SPB clock f_{SPB} feeds the f_{CLC} and it is enabled in CLC register.

24.3.4.1 Baud rate generation

The baud rate generation overview is shown in the figure and formula is given as follows:

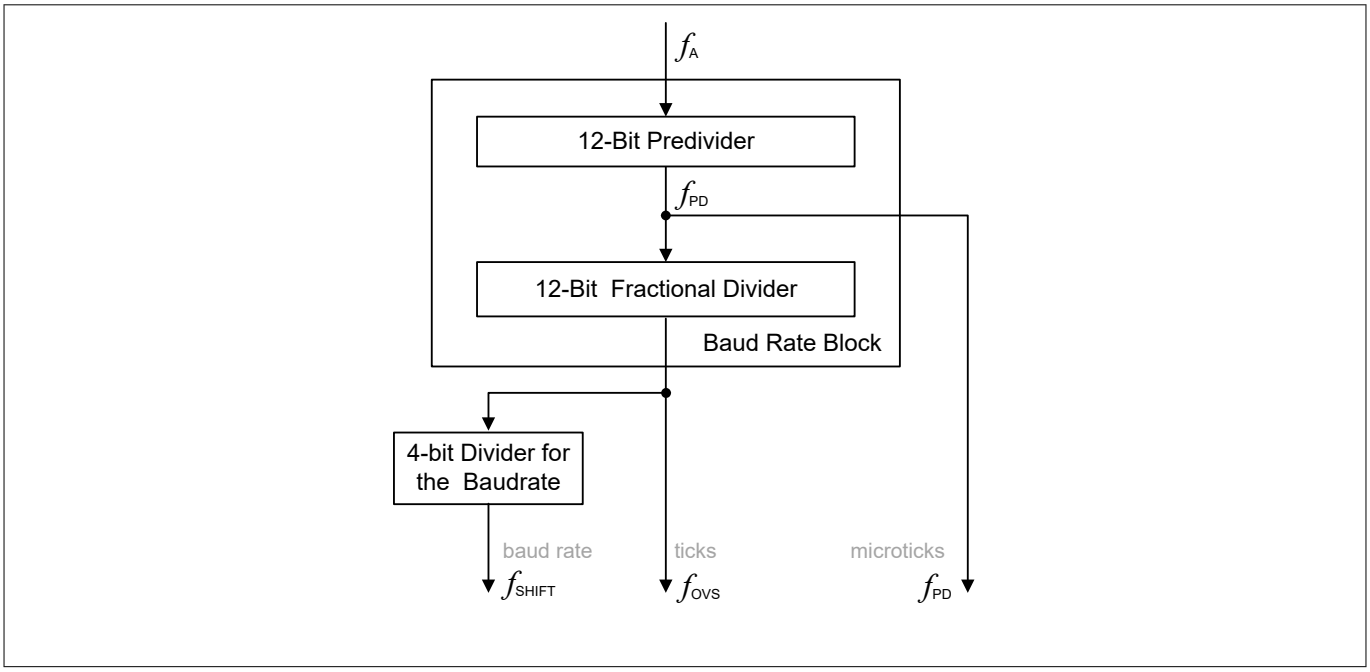


Figure 478 Baud rate generation

The chain of the generated frequencies from f_A down to the f_{SHIFT} (the baudrate) is calculated as follows:

$$f_{PD} = \frac{f_A}{\text{BITCON.PRESCALER} + 1}$$

$$f_{OVS} = f_{PD} \times \frac{\text{BRG.NUMERATOR}}{\text{BRG.DENOMINATOR}}$$

$$f_{SHIFT} = \frac{f_{OVS}}{\text{BITCON.OVERSAMPLING} + 1}$$

The following bit-fields are available for configuring of the 28-bit baud rate divider chain:

- BITCON.PRESCALER - the division ratio of the pre-divider
- BRG.NUMERATOR - the numerator of the fractional divider
- BRG.DENOMINATOR - the denominator of the fractional divider
- BITCON.OVERSAMPLING - the division ratio of the baudrate post divider

***Note**:*
- *Fractional division requires that* BRG*.NUMERATOR is less or equal than the* BRG*.DENOMINATOR*
- *Bit time is equal to the time period of* f_{SHIFT}

24.3.4.2 Bit timing properties

The ASCLIN module provides flexible programming of the bit oversampling, sampling point, digital median filter and input signal de-glitch filtering properties.

The oversampling factor for the incoming bit-stream is configurable from 4 to 16 ticks per bit.

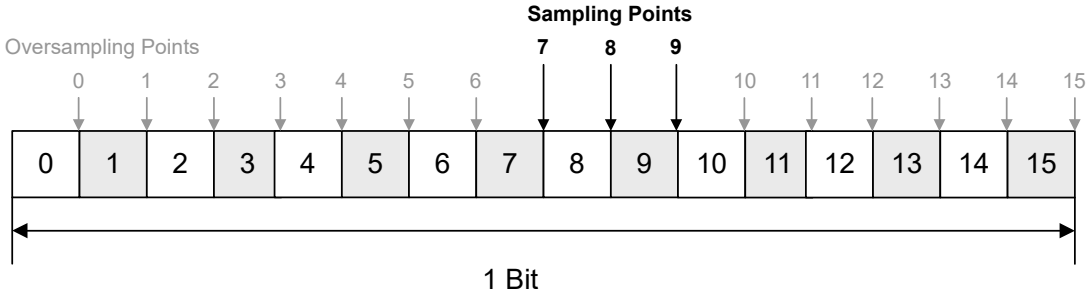
Additionally, using the oversampling frequency, a digital median filter can be enabled to filter the incoming bit stream. The filtering uses the standard majority out of three procedure. If the filter is disabled, then each bit is sampled only once.

The sampling point is also configurable and should be used in conjunction with the oversampling factor. One standard setting is 16x oversampling and using the samples 7, 8 and 9 for the data. Another possible setting could be 8x oversampling and using the samples 3, 4, and 5 for data.

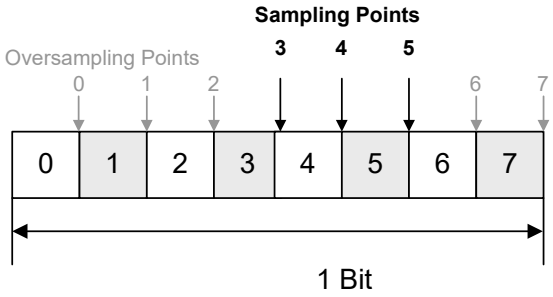
The following bit-fields are available for configuring the bit properties:

- BITCON.PRESCALER - the twelve bit integer divider defines the microtick used by the fractional divider to generate the baud rate and by the digital glitch filter for the de-glitching of the RX input signal
- BITCON.SAMPLEPOINT - the bit-field defines the sampling point position and the duty cycle in the SPI mode
- BITCON.SM - the bit enables the digital median filter (majority out of three): 1 or 3 samples per bit
- IOCR.DEPTH - the bit-field defines the floating average digital glitch filter depth: Off or 1 to 63 microticks
- BITCON.OVERSAMPLING - the bit-field defines the number of ticks per bit, in the range of 4 to 16. This is a post-divider located after the fractional divider

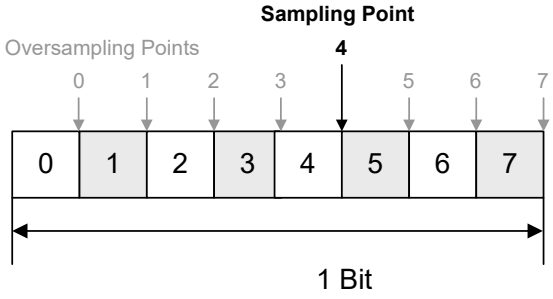
16x Oversampling, 3 sample points, revelant sample points = 7, 8, 9
(BITCON.OVERSAMPLING = 16, BITCON.SM = 1, BITCON.SAMPLEPOINT = 9)



8x Oversampling, 3 sample points, relevant sample points = 3, 4, 5
(BITCON.OVERSAMPLING = 8, BITCON.SM = 1, BITCON.SAMPLEPOINT = 5)



8x Oversampling, 1 sample point, sample point = 4
(BITCON.OVERSAMPLING = 8, BITCON.SM = 0, BITCON.SAMPLEPOINT = 4)



16x Oversampling, 1 sample point, sample point = 12
(BITCON.OVERSAMPLING = 16, BITCON.SM = 0, BITCON.SAMPLEPOINT = 12)

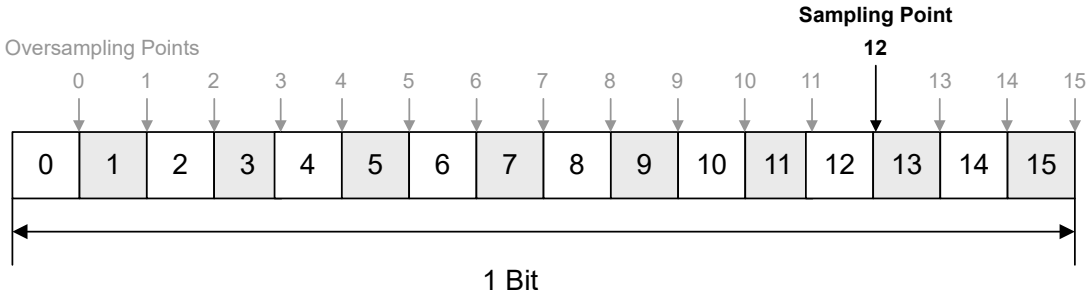


Figure 479 ASCLIN bit structure

Generally, the sampling point should be placed in the middle of the bit. This position is optimal in case the baud rate (the oscillator frequency) is not very precise and stable. However, different combinations of oscillator

precision, asymmetry of the edges, and loop delays for collision detection result in different optimal positions of the sampling point.

One example is shown in the figure below, when two microcontroller driven by quartz oscillator communicate, the oscillator precision is very high. In case the open drain half duplex connection is used, the signal edges are asymmetrical. Open drain connection usually causes the “0” bits to be longer than the “1” bits. In this case it can be of advantage to optimize the sample point by placing it in the middle of the shorter “1” bit.

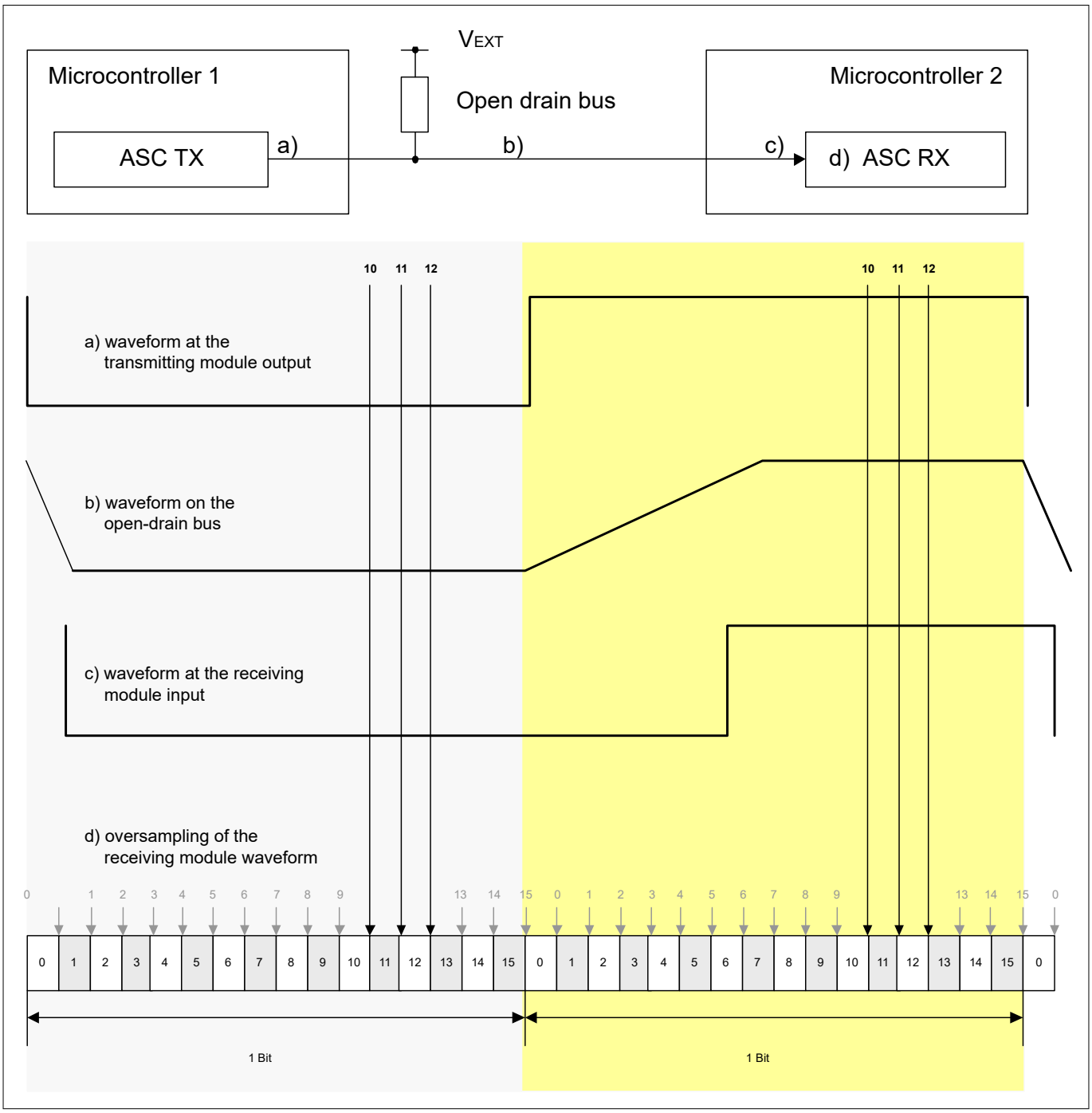


Figure 480 Bit length asymmetry and the sampling point

24.3.4.3 Auto baud rate detection

Auto baud rate detection feature is available when using the module in LIN slave mode. If the auto baud rate usage is enabled LINCON.ABD = 1, this feature is active during the reception of sync field where it measures the longest time interval between two falling edges in the sync field (55_H). The measured value (t_{AB}), is loaded in the denominator of the fractional divider and used afterward for generating the baudrate for the reminder of this frame.

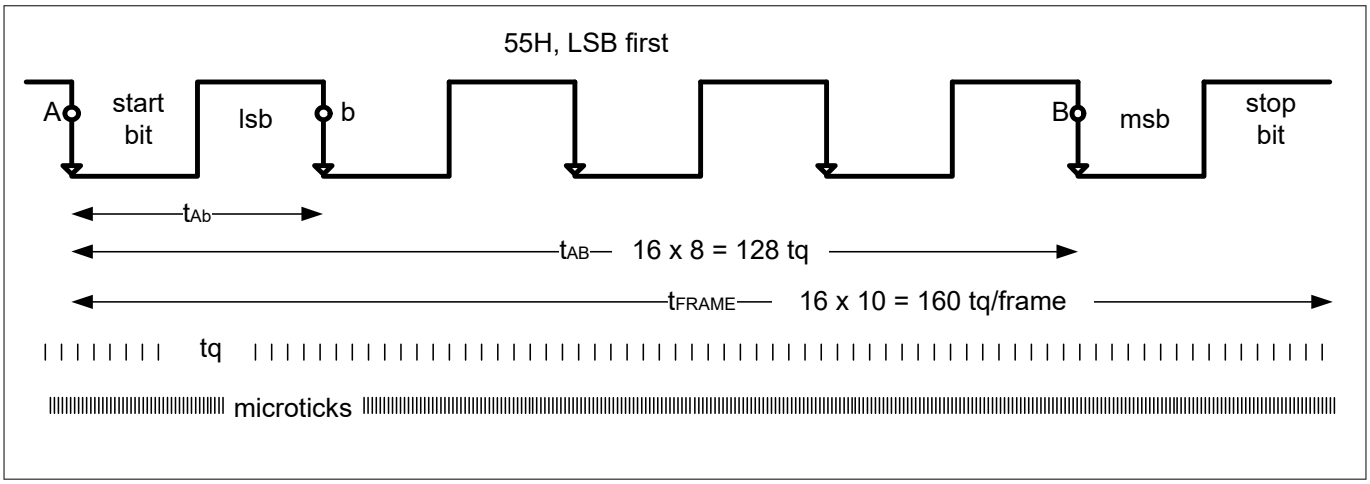


Figure 481 Measurement of the sync field

The following bitfields are available for monitoring the autobaud detection:

- BRD.MEASURED - the measured time interval between the first and the fifth falling edge of the sync byte

  ***Note**: When ABD = 1, the selected PRESCALER shall ensure, that the MEASURED (which is the measured DENOMINATOR) fits into 12 bits for the desired incoming baudate from the master.*

- BRD.UPPERLIMIT - the configured upper limit of the measured time interval
- BRD.LOWERLIMIT - the configured lower limit of the measured time interval

Auto baud rate operation

The BRD.UPPERLIMIT time interval defines the maximum allowed duration for 8 bits in microticks, in other words it defines the minimum allowed baud rate. In case the LIN autobaud detection measures a baud rate 14% lower than the nominal one, meaning it measures a time interval longer than the BRD.UPPERLIMIT, a baudrate event is triggered that, if enabled, generates an interrupt. To define 14% lower baud rate condition, the time interval duration of 16% longer than the nominal (8 bits microticks) must be entered in BRD.UPPERLIMIT bit-field.

The BRD.LOWERLIMIT defines the minimum allowed duration for 8 bits in microticks, and in other words it defines the maximum allowed baud rate. In case the LIN autobaud detection measures a baud rate 14% higher than the nominal one, meaning it measures a time interval shorter than the BRD.LOWERLIMIT, an baudrate event is triggered that, if enabled, generates an interrupt. To define 14% higher baud rate condition, the time interval duration of 12% shorter than the nominal (8 bits microticks) must be entered in BRD.LOWERLIMIT bit-field.

If the auto baud rate detection is activated, the fractional divider uses the numerator value of $8 \times (\text{BITCON.OVERSAMPLING} + 1)$ (BITCON) and ignores the bit-field BRG.NUMERATOR. For example, for the standard LIN protocol where $\text{BITCON.OVERSAMPLING} = 15$, the numerator value used is $8 \times (15 + 1) = 128$. Programming 128 in BRG.NUMERATOR may increase the clarity of the software.

Regarding the denominator of the fractional divider, defined in BRG.DENOMINATOR, its initial value is the nominal value and as set by the application software. During the operation of the module, the BRD.MEASURED value is automatically loaded into the denominator of the fractional divider, as long as it is within the limits.

***Note**: It has to be ensured that the numerator value that is, $8 \times (\text{OVERSAMPLING} + 1)$) is less than or equal to the denominator value (BRD.MEASURED) which is calculated by the Auto Baud Detection mechanism. There is no check implemented by the hardware.*

24.3.5 Data frame configuration

The following bit-fields are available for configuring the data frame, applicable for both transmitter or receiver:

- FRAMECON.ODD - the parity scheme configuration
- DATCON.DATLEN - the data length in the bit-field. Data length shall only be changed after flushing both Tx and Rx FIFOs.
- FRAMECON.STOP - the stop bits

24.3.6 Miscellaneous configuration

Loop-back mode

The loop-back mode is used for test purposes. It is enabled by setting IOCR.LB bit. It connects internally the receive signal ARX with the transmit signal ATX.

Related information

Architecture overview on page 4631

24.3.7 LIN support

The LIN (Local Interconnect Network) is a serial network communication protocol. It is a bus communication made of one master node and several slaves nodes.

A LIN frame consists of the header sent from the master and a response, send either from the master or from the slave. An overview of a LIN frame is shown in the figure below.

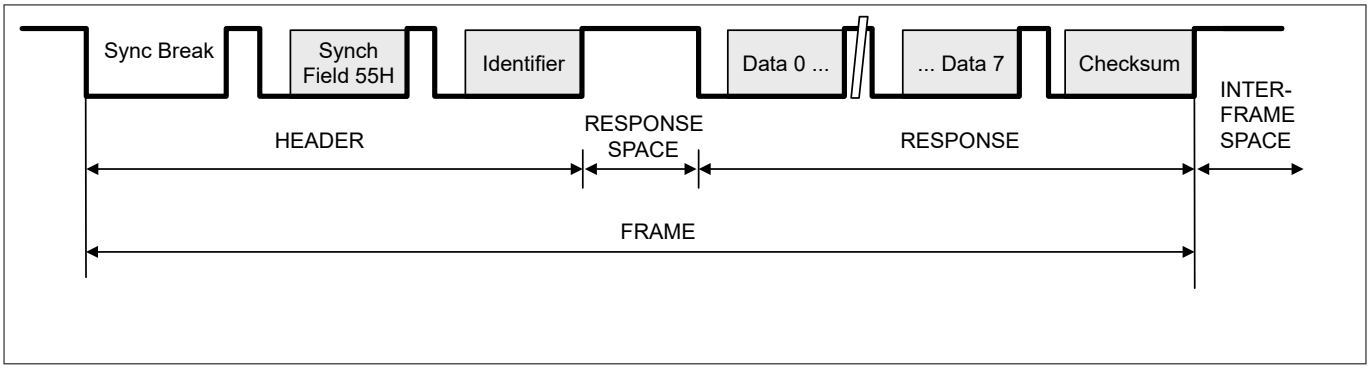


Figure 482 LIN frame format

The ASCLIN module provides hardware support for the LIN protocol. The central LIN protocol state machine is connected to the receiver and the transmitter shift registers, the TxFIFO and RxFIFO, the checksum logic and the watchdog timers. The machine takes care of generating the sync byte of 55_H, if used in master mode, and the automatic handling of the checksum. It supports all LIN use cases, sending and receiving headers and responses as LIN master or LIN slave mode.

It supports all four elementary LIN transactions:

- TxH - Transmission of Header
- TxR - Transmission of Response

- RxH - Reception of Header
- RxR - Reception of Response

A LIN master is engaged in the transactions: TxH, TxR, RxR.

A LIN slave is engaged in the transactions: RxH, TxR, RxR.

Each elementary transaction needs some hardware resources in order to be completed with minimum CPU intervention. Here is a list of tasks per transaction, supported by hardware, and the required hardware resources:

- TxH:
  - Break generation: 6 bit-field defining the break length in units of bits
  - Sync-field generation: Hard coded 55_H byte
  - ID transmission with interrupt generation
- TxR:
  - Number of bytes parameter: bit-field of length 4
  - Checksum generation: hardware engine, supporting classic and enhanced checksum, which can be enabled or disabled
- RxH:
  - Optional auto-baud detection: fractional divider with programmable nominator and denominator
  - Number of bytes parameter: bit-field of length 4
  - Interrupt at end of header (necessary to set the number of bytes for the RxR phase)
  - Timeout on overflow: 8-bit timer
  - Break detection: 8-bit timer with programmable threshold in units of bits
- RxR:
  - Number of bytes parameter: bit-field of length 4
  - Checksum detection: hardware that supports classic and enhanced checksum, which can be enabled or disabled; If enabled, a checksum error is flagged and an error interrupt can be triggered
  - Checksum: the received checksum is optionally delivered in the RXFIFO
  - Timeout on overflow: 8-bit timer

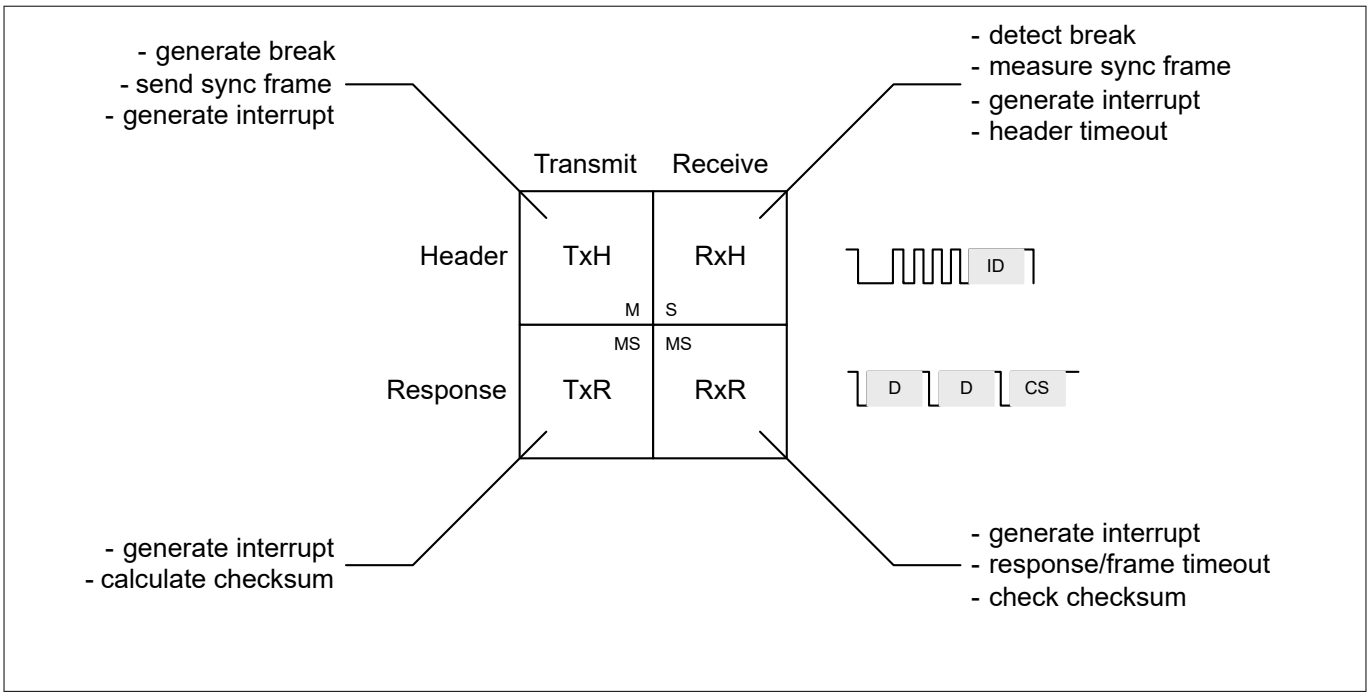


Figure 483 Overview of the elementary LIN transactions

The break detection feature is always active. Wake-up signal generation in both slave and master mode. The configuration parameters are:

- Break length
- Data width
- Break threshold
- Wake-up length
- Header, frame and response timeout
- Idle time

***Note**: For many LIN configuration parameters, the hardware of the ASCLIN module provides wider ranges than the LIN standard parameters. Therefore the application software shall take care that appropriate LIN standard values are used to configure the module.*

24.3.7.1 Using the ENI and HO Bits

The ENI (Enable Input) and HO (Header Only) bits affect the reception of the various byte types in the LIN frame.

The HO bit is normally used in cases where the response part of a frame should be ignored; the module waits for the next break signal. The ENI bit can be set by the user software, but is also set by the hardware in slave mode after the sync byte reception, in order to enable the transfer of the ID byte of the header in the RXFIFO.

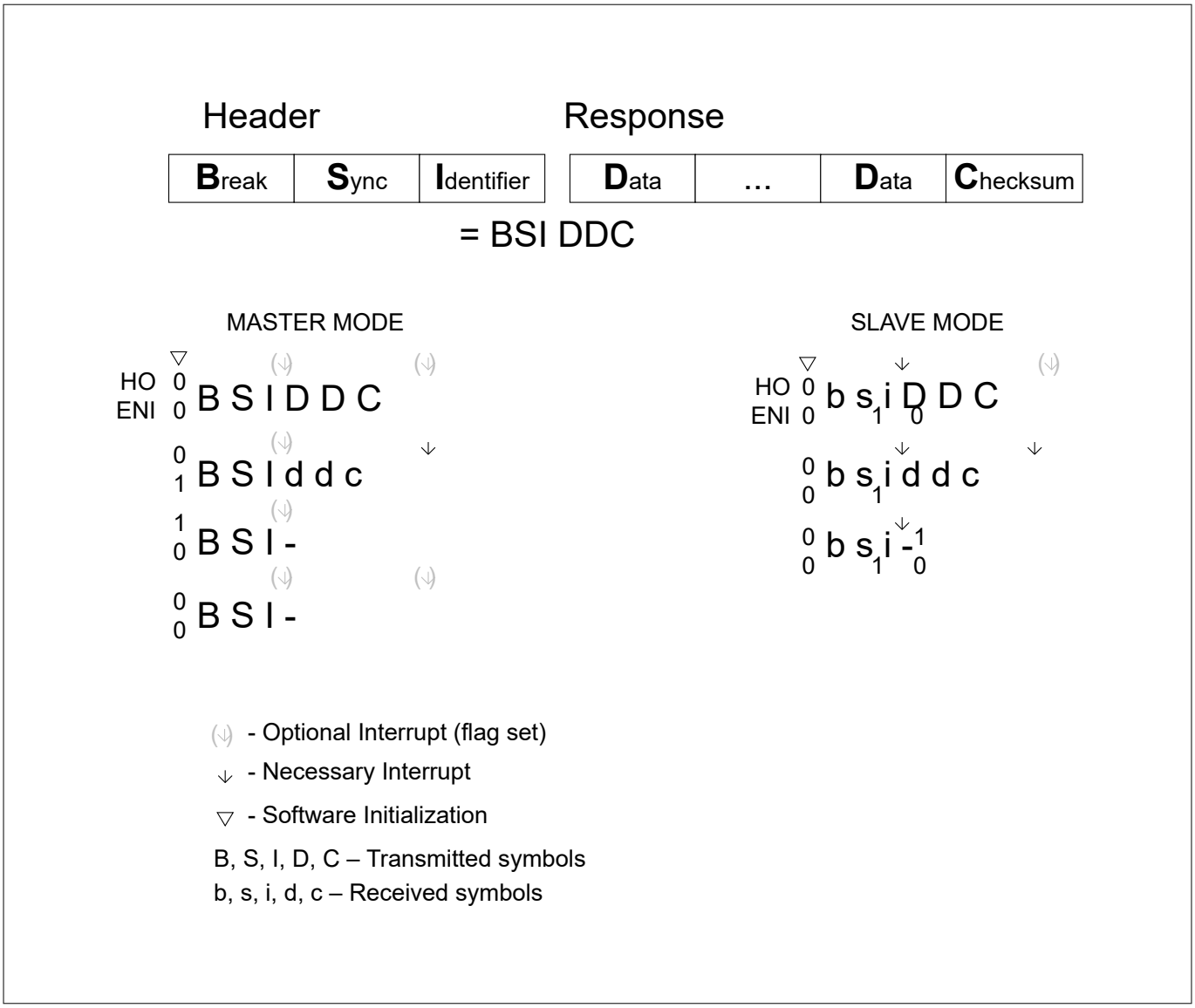


Figure 484 ENI and HO bits

24.3.7.2 LIN watchdog

The LIN watchdog monitors the duration of the header, the frame or the response, and appearance of break pulses. It checks against the pre-defined time limits. If the limits are violated, timeout interrupts are generated.

The LIN watchdog is necessary in slave mode.

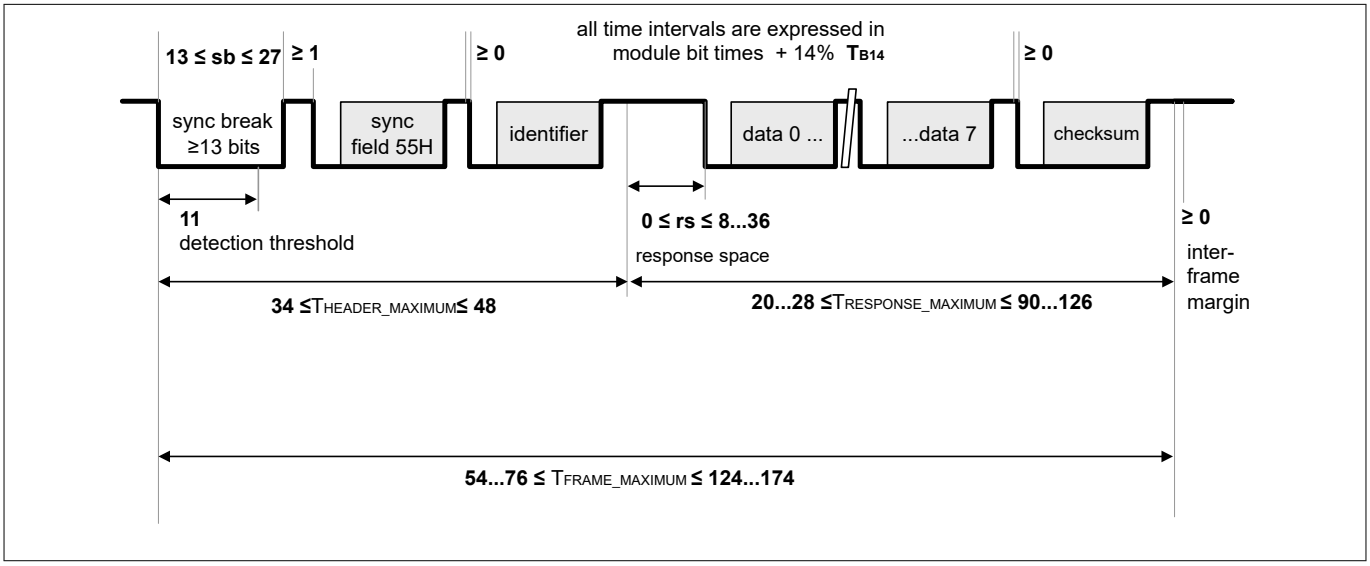


Figure 485 Duration of the elements of a LIN frame

This section describes the following watchdog monitors:

- Monitoring of the LIN Bus
- Break and Wake-up pulse detection
- Break and Wake-up pulse generation
- Header and response timeouts

Monitoring the bus

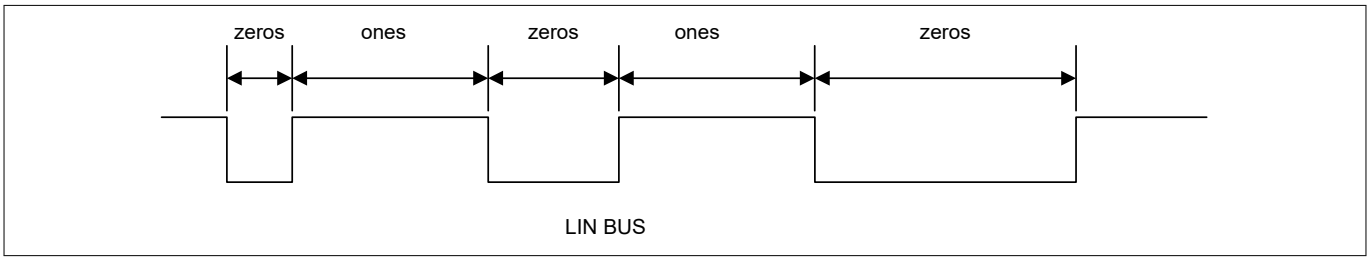


Figure 486 LIN bus view as sequence of sequences of zeros and ones

Monitoring the bus for long idle periods (or stuck periods) is supported by providing the bit-fields FLAGS.RED and FLAGS.FED. Polling these fields periodically within some defined time period by the operating system or timer for example each 1 ms or each 10 ms can be used for detecting very long inactive periods in the range, for example, of 150 ms to 10 s. Additionally, an interrupt on edge can be enabled by using the corresponding enable bits in the FLAGSENABLE registers. Setting and clearing the flags can be done by software by using the FLAGSSET and FLAGSCLEAR register.

Break and wake pulse detection

The detection of the break pulse, which is defined as low pulse with a minimum duration of 13-bit times, is performed using a threshold of either 10 or 11-bit times. The break detection threshold is set using the bit-field LINBTIMER.BREAK.

Detecting a break pulse anywhere in the frame resets the LIN state machine and the whole sequence including the watchdog timeout counting starts from the beginning. A wake-up pulse, which is also a low pulse with a duration longer than 150 µs wakes up a sleeping module. A slave which has been woken up shall be capable of receiving LIN headers after a wake-up time of maximum 100 ms.

***Note**: Detecting a break pulse anywhere in the frame could result in setting the Frame Error Flag (FE)*

Wake-up pulse detection is done by monitoring the bus for a falling edge in sleep mode. If an early wake-up shall be suppressed, the parameter IOCR.DEPTH (glitch filter) may be used.

Break and wake-up pulse generation

The generated break pulse duration is defined by the bit-field LINBTIMER.BREAK in the range of 1 to 64 bits.

Wake-up pulse is nominal 5-bits long low pulse, but can be set to any value between 1 and 9 by writing the appropriate character in the TXFIFO and requesting its transmission as the wake pulse by setting FLAGS.TWRQ (by writing 1_B to FLAGSSET.TWRQS). For a wake-up pulse of n-bit width, the appropriate character should contain "(n-1)" 0 values considering. Below are some examples for wake-up pulses of different width at a speed of 20 kBaud/s (example configurations: fA = 100 MHz, prescaler is 99, oversampling is 4, numerator is 1, denominator is 10):

- To generate the wake up pulse 9-bit long (450 µs), character written to the TXFIFO shall be 00000000_B or 00_H
- To generate the wake-up pulse 5-bit long (250 µs), character written to the TXFIFO shall be 11110000_B or $F0_H$

A master node which has received a wake-up pulse can start polling the slaves, it can start sending break pulses and headers, and send or receive the corresponding responses.

A slave issuing a wake-up pulse expects to receive a header within 150 ms to 250 ms after the end of the wake-up pulse. If the header does not come, the slave issues a wake-up pulse again.

***Note**: Injecting the low pulse disables the reception with the shift register, so that this low pulse is not detected or treated as a normal ASC frame.*

LIN header and response timers

The LIN header and response times can be monitored separately.

- Header duration measurement:
  - In master mode, the starting point of the time measurement is the falling edge of the break pulse
  - In slave mode, the starting point of the time measurement is the moment of detecting that there is a break pulse going on, that is the moment of detection of a low pulse longer than either 10 or 11-bit times, as configured in the LINBTIMER register
- Response duration measurement:
  - In master and slave mode the response duration measurement starts with the end of the header and ends with the end of the response

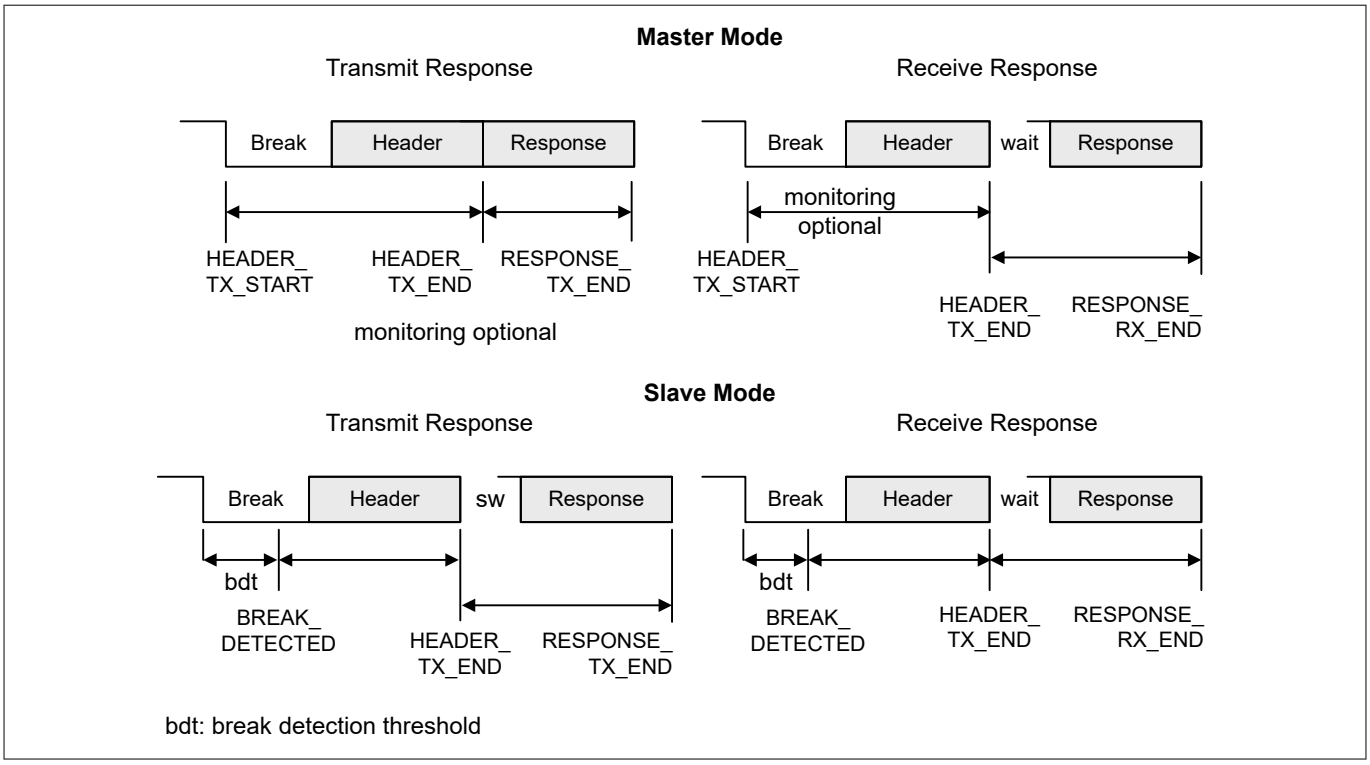


Figure 487 Duration measurement of the header and response

***Note**: Transmitting master node can optionally monitor its own header or response timeouts, in order to detect some error conditions, like TxFIFO not containing the ID or data to be transmitted. However, these error conditions can be detected also in other ways.*

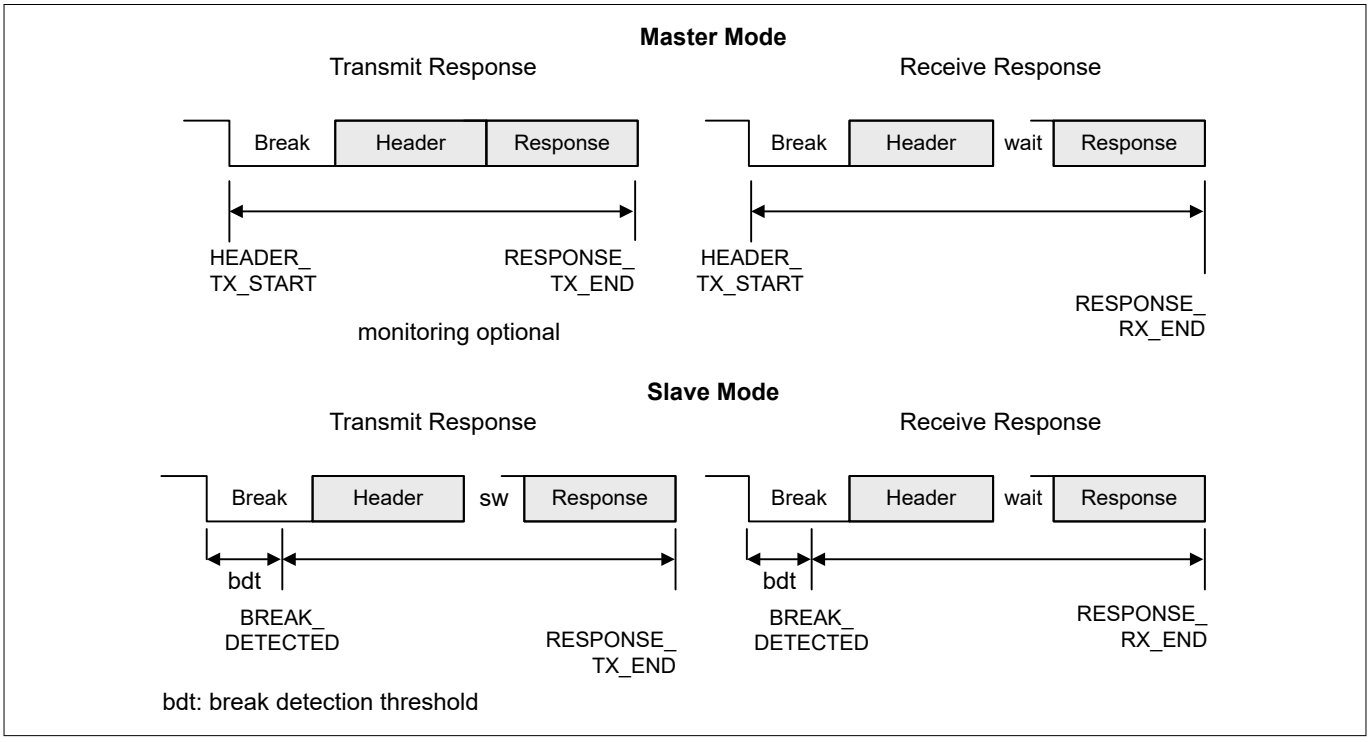


Figure 488 Duration measurement of the response or frame

According to the setting of the bit DATCON.RM, the DATCON.RESPONSE bit-field defines response or frame duration threshold.

LIN version 1.x specified the Frame Timeout Level, which includes header where as in LIN version 2.x the Response Timeout Level counter starts after the header, and does not include Header Timeout Level.

24.3.7.3 LIN checksum

In ASCLIN both the classic LIN V1.3 and the enhanced LIN V2.0/V2.1 checksum are supported. The hardware checksum feature is switchable (enable or disable), and the choice between using the classical and the enhanced checksum is done by software with the DATCON.CSM bit on a frame by frame basis. As can be seen in the figure below, for the LIN version 2.0 and 2.1, the enhanced checksum is calculated for the identifiers 0…59, and the classical checksum for the identifiers 60…63.

Additionally, the parity of the ID field is generated in master mode, and checked in slave mode. In slave mode, in case of a mismatch between the received and the calculated parity, an error interrupt is raised.

In receive case, if LINCON.CSI = 1, the received checksum byte is written to the RxFIFO after the last data byte.

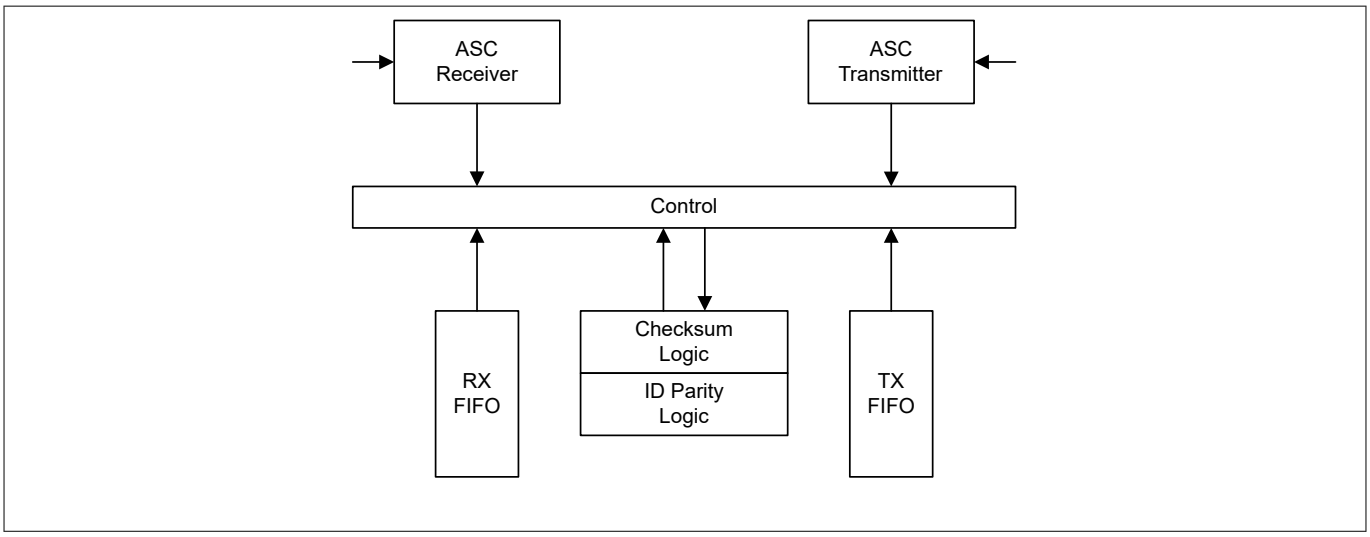


Figure 489 Block diagram of the LIN control subsystem

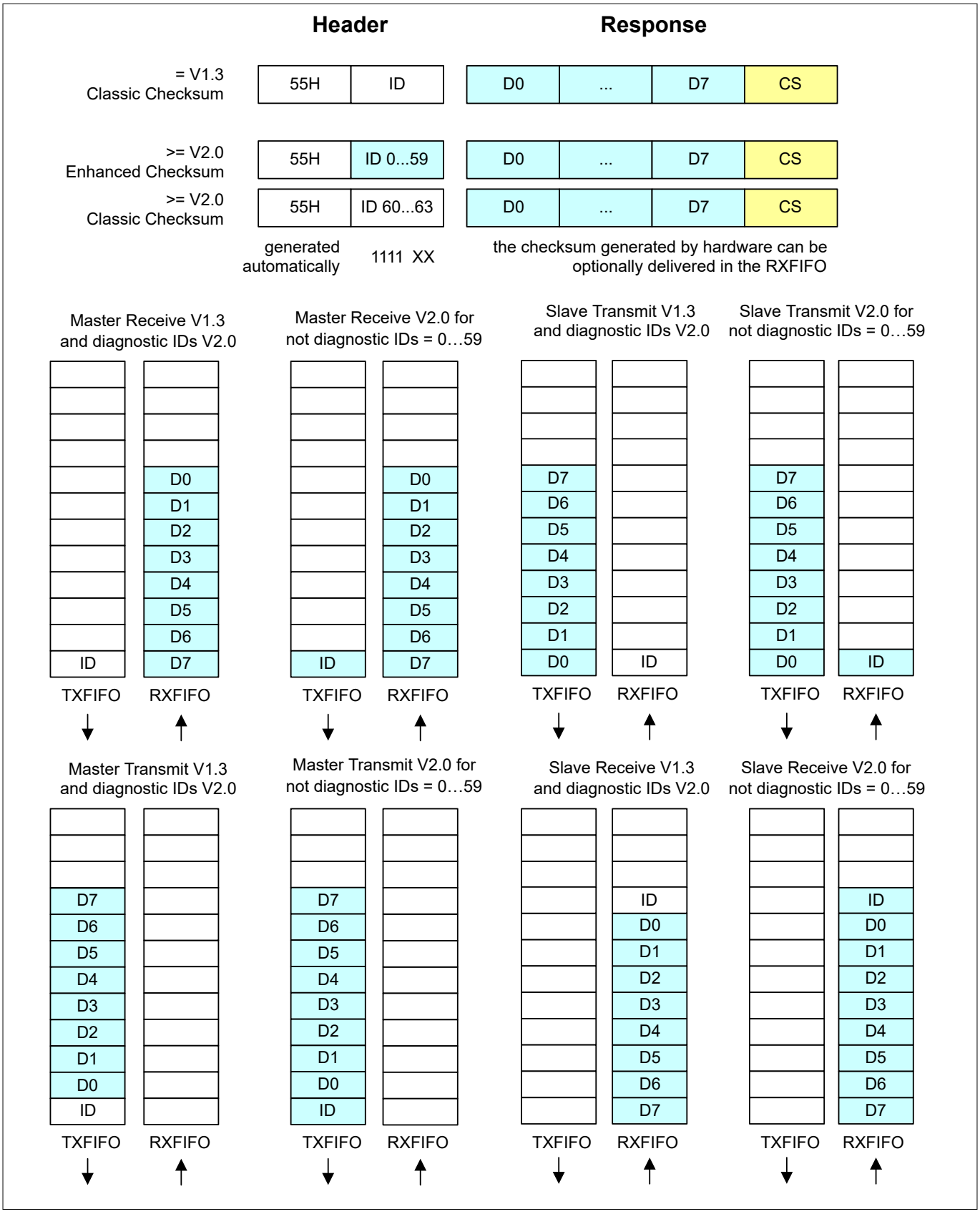


Figure 490 Coverage of the checksum in different use cases

24.3.7.4 LIN error recovery

This section describes the behavior of the module in case of errors detected during reception and during transmission of a LIN frame.

Reception related errors

The following errors are possible during the reception:

- ID Parity error (FLAGS.LP)
- Checksum error (FLAGS.LC)
- Timeout error (FLAGS.HT and FLAGS.RT)
- Framing error (FLAGS.FE)
- Baud Rate error (FLAGS.LA)

In case one of the above mentioned errors is encountered, the receive state machine goes into the state “waiting for break”, and the corresponding interrupt is triggered. In case of ID Parity or Baudrate error, data is not stored in RxFIFO. In all other cases, data received until the error condition happened is stored in RxFIFO.

Transmission related errors

During transmission it is possible to encounter collision error. If the collision detection mechanism is enabled via FLAGSSET.CES (in LIN2.1 mandatory to enable) FRAMECON.CEN, the frame will be aborted and the transmitter state machine goes to the idle state. The aborted frame is lost no longer stored in TxFIFO.

24.3.8 SPI support

The ASCLIN supports the SPI master communication in synchronous mode. The SPI setting of shift edge first, than the latch edge, see figure below. The module is set in synchronous mode by using the bit-field FRAMECON.MODE.

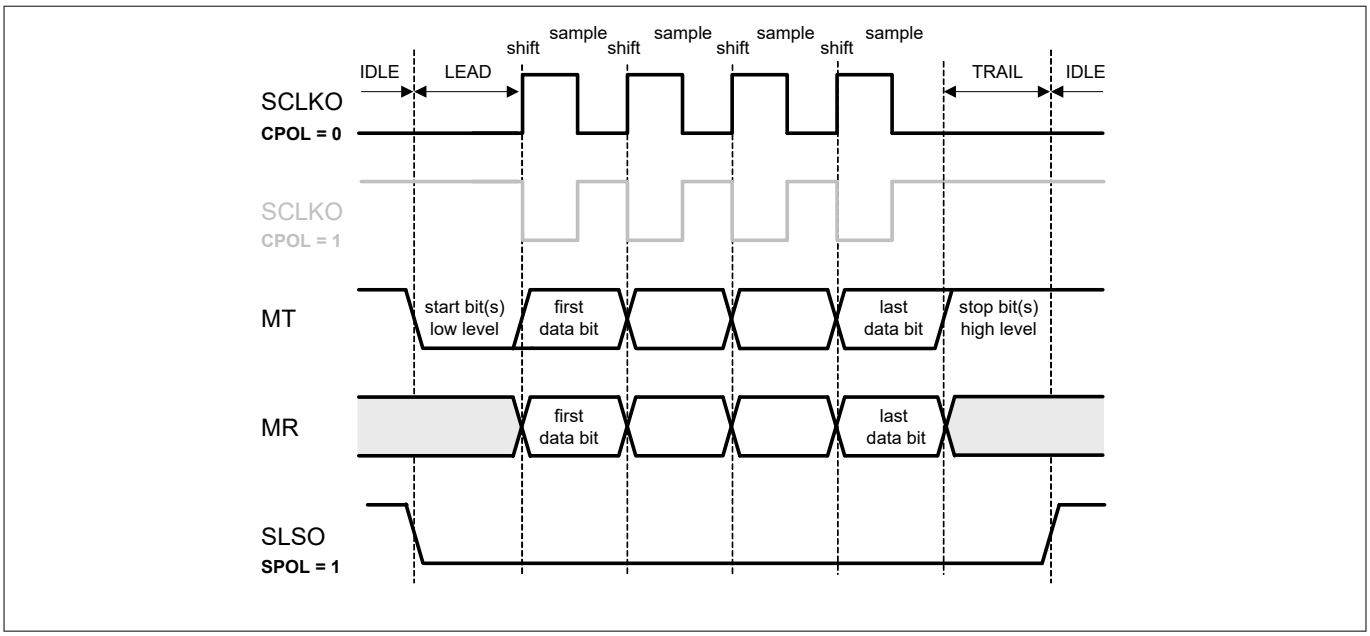


Figure 491 SPI timing

24.3.8.1 Baud rate and clock generation

The baud rate and clock generation in the synchronous mode uses generally the same counters and principles as in the asynchronous mode. It uses the same prescaler, fractional divider, and oversampling divider with configurable sample point. The only difference is that the shift clock is driven as output signal at the pin SCLKO.

If a symmetrical shift clock is required, the oversampling ratio should be an even number, and the sampling point should be set to one half of the oversampling ratio (see BITCON register).

The clock polarity is configured in the bit IOCR.CPOL.

24.3.8.2 Data frame configuration

The leading and trailing delays are configured in the bit-fields FRAMECON.LEAD and FRAMECON.STOP. The IDLE phase duration is configured using the bit-field FRAMECON.IDLE. The data length of 2 to 16-bit is defined in the bit-field DATCON.DATLEN.

24.3.8.3 Slave selects configuration

The SPI master activates automatically the slave select output signal for each data word. The polarity of the slave select can be configured by using IOCR.SPOL.

24.3.9 Collision detection

Collision detection monitors the consistency of transmitted and the echoed received bytes in LIN mode and half duplex SPI modes. Collision detection is enabled in FRAMECON.CEN bit-field.

24.3.10 Interrupts

The ASCLIN module generates three interrupts:

- TX_INT - Transmit Interrupt
  - Signals the TxFIFO level event (FLAGS.TFL)
- RX_INT - Receive Interrupt
  - Signals the RxFIFO level event (FLAGS.RFL)
- ERR_INT - Extended Error Interrupt
  - Signals every error (FLAGS.PE, FE, CE, RFO, RFU, TFO,TFE,OMT) and
  - Additional events (FLAGS.FED, RED)

The LIN events related to transmission and reception of header and response are mapped to the corresponding transmit, receive and extended error interrupts.

- TX_INT:
  - Transmission of the header in master mode completed (FLAGS.TH)
  - Transmission of a response completed (FLAGS.TR)
- RX_INT:
  - Reception of the header in a slave mode completed (FLAGS.RH)
  - Reception of the response completed (FLAGS.RR)
- ERR_INT:
  - LIN protocol (FLAGS. BD, TC) and
  - LIN error events (FLAGS.HT, RT, LP, LA, LC)

In order to determine which event is the source of a LIN interrupt, the FLAGS register must be polled.

Triggering a DMA

The interrupt signals are used also as DMA trigger signals. The interrupt signals are connected to the Interrupt Router Module, which routes the interrupts either to a CPU or to a DMA. There are no separate DMA trigger signals.

Service request and the events

There are several events mapped to each service request node.

Each service request (interrupt) node contains an SRC (Service Request Control) register containing a sticky flag bit and the associated set, clear and enable bits. The SRC registers are located in the IR (Interrupt Router) module.

One interrupt node may be triggered by more than one events. Each event which causes an interrupt also has a sticky flag bit and associated set, clear and enable bits. These bits are distributed in four registers: FLAGS, FLAGSSET, FLAGSCLEAR, and FLAGSENABLE located in the ASCLIN module. Each set of four bits associated to one event builds a virtual “event node”, see figure below.

The event flags are set by hardware and if enabled, trigger an interrupt. They are not cleared by hardware. They are cleared only by software in the corresponding interrupt service routine, which usually polls the flags to find the cause for the interrupts. The event flags can be also set by software for test purposes.

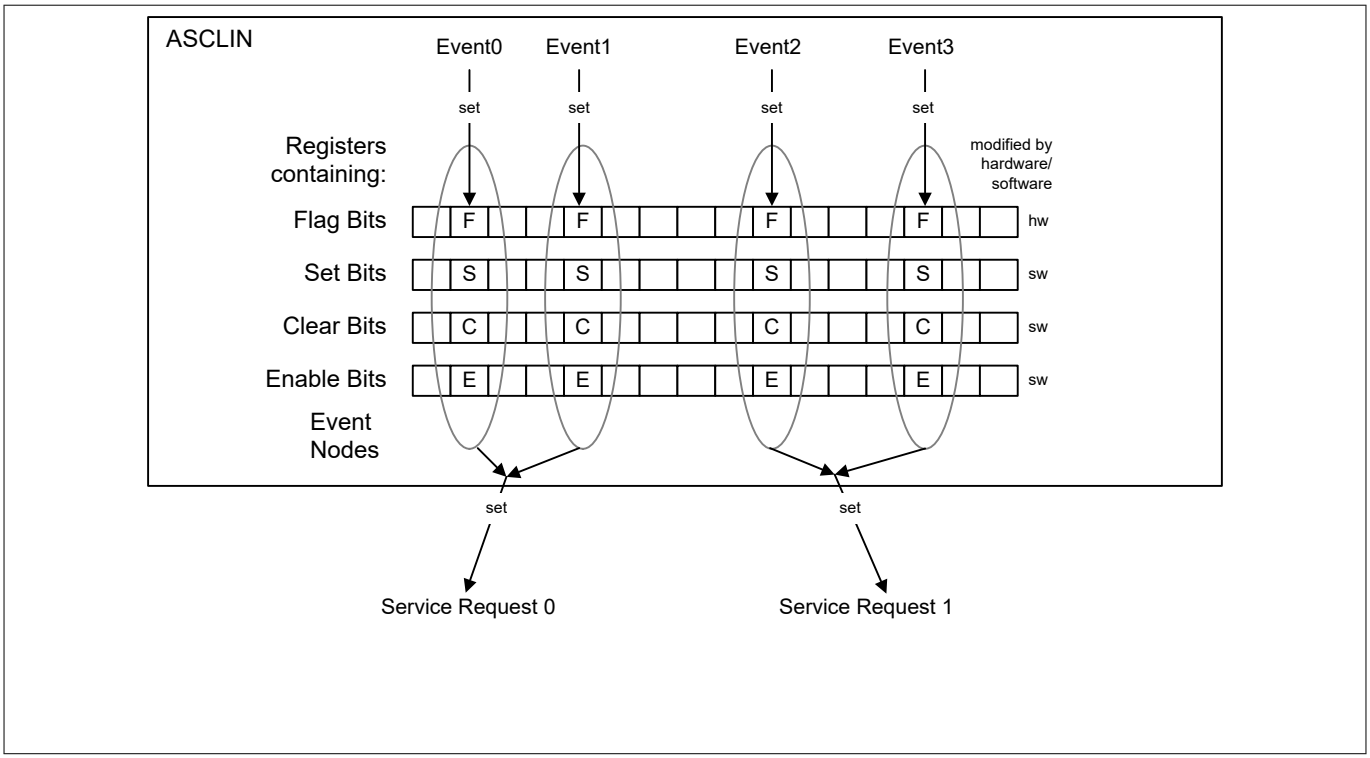


Figure 492 Relationship between event nodes and service request nodes

Some flags do not generate interrupts and do not have enable bits. They are used either for issuing requests, which are subsequently acknowledged by hardware, or only for polling by software, see figure below.

There are no pure polled only flags in the ASCLIN module, although by letting the enable bits disabled, all interrupt event flags can be used as polled only (for example idle and stuck monitoring by using FLAGS.FED, RED. Setting such bits by software in such a case is meaningless, except for test purposes).

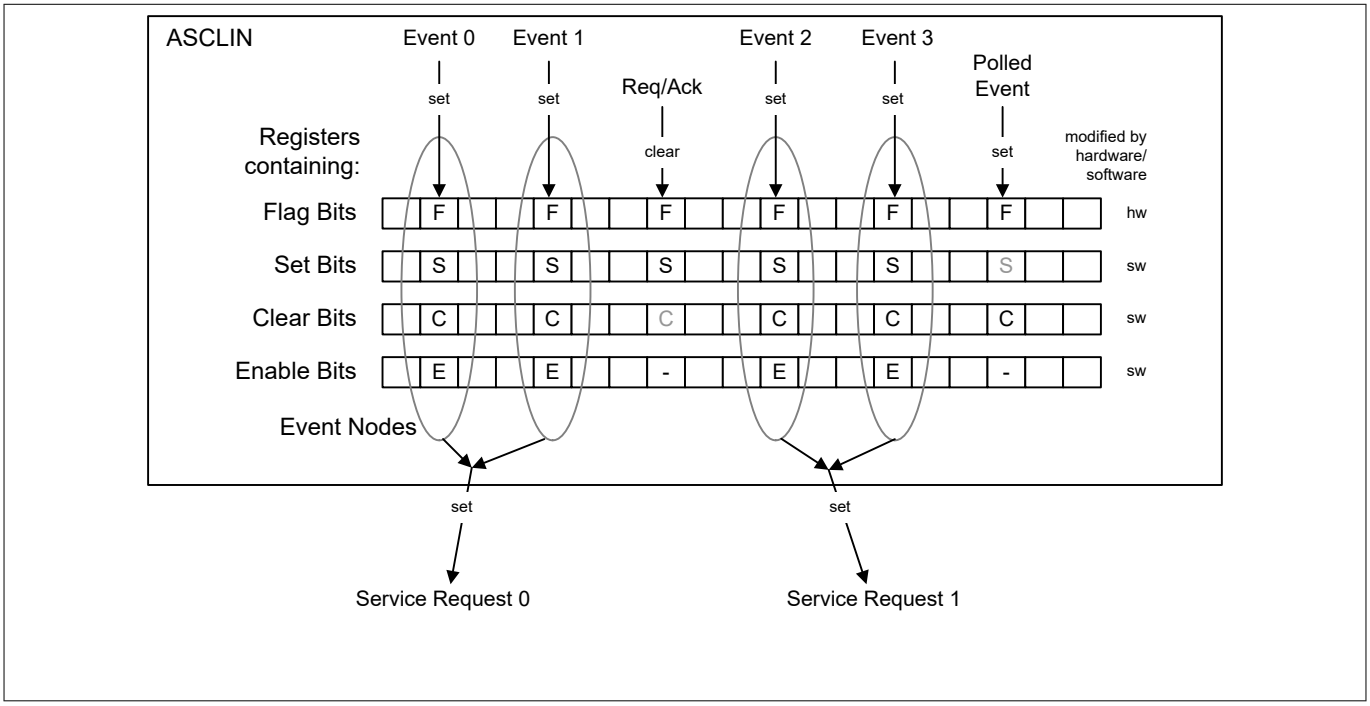


Figure 493 Acknowledge flags and polled flags

24.3.11 Digital glitch filter

The digital glitch filter removes short glitches from the input data signal by using an increment/decrement counter with programmable threshold. On the other hand, the filter introduces a delay into the signal path, depending on the digital filter sampling frequency f_{PD} and the threshold programmed in the bit-field IOCR.DEPTH. Hence the glitch filter should not be enabled in SPI mode.

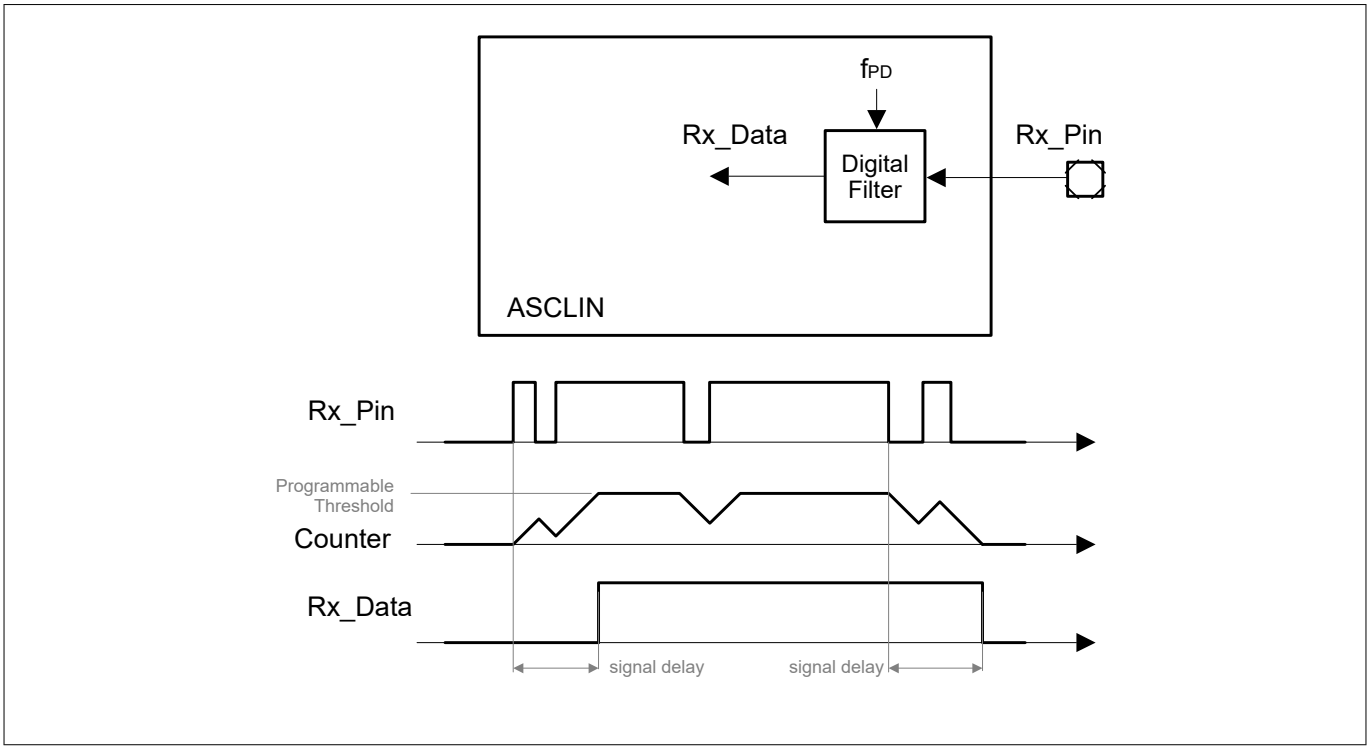


Figure 494 Digital filter

24.3.12 Sleep and power-off behavior

The ASCLIN module transmits either ASC frames (shorter frames less than 16 bits, but mostly 8 bits long) or LIN frames. Freezing the module in the middle of an asynchronous ASC frame generally results in an erroneous data at the receiver side, depending on the data content.

Freezing the module inside the LIN frame transmission, but between the transmission of single bytes, results in a timeout error at the receiver side.

If the ASCLIN module gets a request for switching off the clock, it immediately disables all request lines towards the DMA in order to stop further DMA transfers.

In order to power-down the module in a well defined way, the user software must take care that:

- The power-down request is issued when there is no ongoing single frame or LIN frame transfer
- Both FIFOs are empty

24.3.12.1 Sleep mode

Going into sleep mode implies that the ASCLIN module has finished all activities and that the TxFIFO and RxFIFO are empty. It is the responsibility of the user software to issue a sleep request in a safe time interval, where no race conditions or pipeline effects between the ASCLIN module and the DMA can occur.

The sleep mode is controlled using the CLC.EDIS register bit.

The ASCLIN module in Sleep Mode behaves the same way as in case of Disable Request.

24.3.12.2 Disable request (power-off)

Going into power off implies that the ASCLIN module has finished all activities and that the TX and RX FIFOs are empty. It is the responsibility of the user software to issue a power-off request in a safe time interval, where no race conditions or pipeline effects between the ASCLIN module and the DMA can occur. This is no issue if the power-off procedure is one-way that is no continuation of operation without re-initialization of the module is expected.

The Disable request is controller using the CLC.DISR register bit and the status can also be observed using the CLC.DISS register bit.

The ASCLIN module sets immediately all outputs to inactive, stops reacting to inputs, the internal state machines go to initial state, the registers remain unchanged, and then the module acknowledges the request. All clocks to the module are switched off.

24.4 Registers

24.4.1 Register overview - access mode glossary

Table 1115 Register overview - access mode glossary

Keyword	Description
E	Access protection using PROT register PROTE.
SE	Access protection using PROT register PROTSE.
APU-P	Protection group consisting of registers ACCEN_WRA, ACCEN_WRB, ACCEN_RDA, ACCEN_RDB, ACCEN_VM, ACCEN_PRS.
P	Access protection using *APU-P* registers.
BE	Always returns a Bus Error.

(table continues...)

Table 1115 (continued) Register overview - access mode glossary

Keyword	Description
U	No access restrictions.
PROT	Access restrictions as defined in the PROT register access rules.

24.4.2 Registers overview - ASCLIN (ascending offset address)

Table 1116 Registers overview - ASCLIN (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CLC	Clock control register	000_H	P	P, SV, E	See 4670	4670
OCS	OCDS control and status register	004_H	P	SV, P	Debug Reset	4671
ID	Module identification register	008_H	P	BE	PowerOn Reset	4672
RST_CTRLA	Reset control register A	$00C_H$	P	P, SV, E	Application Reset	4673
RST_CTRLB	Reset control register B	010_H	P	P, SV, E	Application Reset	4673
RST_STAT	Reset status register	014_H	P	BE	Application Reset	4674
PROTE	PROT register endinit	020_H	U	SV, PROT	Application Reset	4675
PROTSE	PROT register safe endinit	024_H	U	SV, PROT	Application Reset	4676
ACCEN_WRA	Write access enable register A	040_H	U	SE, SV	Application Reset	4678
ACCEN_WRB	Write access enable register B	044_H	U	SE, SV	Application Reset	4678
ACCEN_RDA	Read access enable register A	048_H	U	SE, SV	Application Reset	4679
ACCEN_RDB	Read access enable register B	$04C_H$	U	SE, SV	Application Reset	4679
ACCEN_VM	VM access enable register	050_H	U	SE, SV	Application Reset	4680
ACCEN_PRS	PRS access enable register	054_H	U	SE, SV	Application Reset	4680
IOCR	Input and output control register	100_H	P	P	See 4681	4681
TXFIFOCON	TxFIFO configuration register	104_H	P	P	See 4683	4683
RXFIFOCON	RxFIFO configuration register	108_H	P	P	See 4685	4685
BITCON	Bit configuration register	$10C_H$	P	P	See 4687	4687

(table continues...)

Table 1116 (continued) Registers overview - ASCLIN (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
FRAMECON	Frame control register	110_H	P	P	See 4689	4689
DATCON	Data configuration register	114_H	P	P	Kernel Reset	4691
BRG	Baud rate generation register	118_H	P	P	See 4692	4692
BRD	Baud rate detection register	$11C_H$	P	P	Kernel Reset	4693
LINCON	LIN control register	120_H	P	P	Kernel Reset	4694
LINBTIMER	LIN break timer register	124_H	P	P	Kernel Reset	4695
LINHTIMER	LIN header timer register	128_H	P	P	Kernel Reset	4696
FLAGS	Flags register	$12C_H$	P	BE	Kernel Reset	4696
FLAGSSET	Flags set register	130_H	P	P	Kernel Reset	4701
FLAGSCLEAR	Flags clear register	134_H	P	P	Kernel Reset	4704
FLAGSENABLE	Flags enable register	138_H	P	P	Kernel Reset	4708
CSR	Clock selection register	$13C_H$	P	P	See 4712	4712
TXDATA	Transmit data register	140_H	P	P	Kernel Reset	4713
TXDATA# (#=0-7)	Transmit data register mirror #	140_H+#*4	P	P	Kernel Reset	4713
RXDATA	Receive data register	160_H	P	BE	Kernel Reset	4713
RXDATA# (#=0-7)	Receive data register mirror #	160_H+#*4	P	BE	Kernel Reset	4713
RXDATAD	Receive data debug register	180_H	P	BE	Kernel Reset	4714

24.4.3 Clock control register

The Clock Control Register CLC allows the programmer to adapt the functionality and power consumption of the module to the requirements of the application.

Register CLC controls the module clock signal and the reactivity to the sleep signal.

CLC Offset address: 000_H

Clock control register Reset values see: Table 1117

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												**EDIS**	**0**	**DISS**	**DISR**
r												rw	r	rh	rw

Field	Bits	Type	Description
DISR	0	rw	**Module Disable Request Bit** Used for enable/disable control of the module. 0_B On request: enable the module clock 1_B Off request: stop the module clock
DISS	1	rh	**Module Disable Status Bit** 0_B Module clock is enabled (On) 1_B module is not clocked (Off)
EDIS	3	rw	**Sleep Mode Enable Control** Used to control the module's reaction to sleep mode. 0_B Sleep mode request is enabled and functional 1_B Module disregards the sleep mode control signal
0	2, 31:4	r	**Reserved** Read as 0; should be written with 0.

Table 1117 Reset values of CLC

Reset type	Reset value	Note
Application Reset	$0000\ 0003_H$	
After Boot-FW Value	$0000\ 0000_H$	After ASC BSL execution

24.4.4 OCDS control and status register

The OCDS Control and Status register OCS controls the debug and trace behavior by selecting suspend modes. When OCDS is disabled the suspend control is ineffective.

OCS Offset address: 004_H

OCDS control and status register Debug Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		SUSSTA	SUS_P	SUS				0							
r		rh	w	rw				r							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															
r															

Field	Bits	Type	Description
SUS	27:24	rw	**OCDS Suspend Control** Controls the sensitivity to the suspend signal coming from the OCDS Trigger Switch (OTGS) Not listed combinations are reserved. 0_H Will not suspend 1_H Hard suspend: Clocks will be disabled immediately 2_H Soft suspend mode: Clock will be disabled after acknowledge. 3_H Reserved: not used
SUS_P	28	w	**SUS Write Protection** SUS is only written when SUS_P is 1, otherwise unchanged. Read as 0.
SUSSTA	29	rh	**Suspend State** 0_B Module is not (yet) suspended 1_B Module is suspended
0	23:0, 31:30	r	**Reserved**

Table 1118 Access mode restrictions of OCS sorted by descending priority

Mode name	Access mode		Description
write 1 to .**SUS_P**	rw	SUS	Set SUS_P during write access
(default)	r	SUS	

24.4.5 Module identification register

ID Offset address: 008_H

Module identification register PowerOn Reset value: $00C1\ C005_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MOD_NUM															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MOD_TYPE								MOD_REV							
r								r							

Field	Bits	Type	Description
MOD_REV	7:0	r	**Module Revision** Indicates the revision number of the implementation.
MOD_TYPE	15:8	r	**Module Type** This internal marker is fixed to $C0_H$.
MOD_NUM	31:16	r	**Module Number** Indicates the module identification number.

24.4.6 Reset control register A

RST_CTRLA Offset address: $00C_H$

Reset control register A Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				GRSTEN3	GRSTEN2	GRSTEN1	GRSTEN0	0							KRST
r				rw	rw	rw	rw	r							rwh

Field	Bits	Type	Description
KRST	0	rwh	**Kernel Reset** Request a kernel reset. The requested reset is executed if CTRLA.KRST=1 and CTRLB.KRST=1. KRST is cleared after the kernel reset was executed. 0_B No action 1_B A kernel reset was requested
GRSTENx (x=0-3)	x+8	rw	**Enable for Global Module Reset Group x** 0_B Global module reset group x does not have any effect 1_B Global module reset group x results in a kernel reset
0	7:1, 31:12	r	**Reserved** Read as 0; should be written with 0.

24.4.7 Reset control register B

RST_CTRLB Offset address: 010_H

Reset control register B Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
STATCLR								0							
w								r							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															KRST
r															rwh

Field	Bits	Type	Description
KRST	0	rwh	**Kernel Reset** Request a kernel reset. The requested reset is executed if CTRLA.KRST=1 and CTRLB.KRST=1. KRST is cleared after the kernel reset was executed. 0_B No action 1_B A kernel reset was requested
STATCLR	31	w	**Kernel Reset Status Clear** Clears all status bits in RST_STAT when 1 is written. Read will return 0. 0_B No action 1_B Write with ´1´ clears bits STAT.GRSTx and bit STAT.KRST .
0	30:1	r	**Reserved** Read as 0; should be written with 0.

24.4.8 Reset status register

The reset status register contains the status bits for kernel reset (KRST) and the global module reset groups (GRSTx).

RST_STAT Offset address: 014_H

Reset status register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				GRST3	GRST2	GRST1	GRST0	0							KRST
r				rh	rh	rh	rh	r							rh

Field	Bits	Type	Description
KRST	0	rh	**Kernel Reset Status** Indicates an executed kernel reset. RST_STAT.KRST is set after the execution of a kernel reset in the same clock cycle in which the reset bits are cleared. Clear KRST by setting bit STATCLR in register RST_CTRLB.
GRSTx (x=0-3)	x+8	rh	**Status for Global Module Reset Group x** 0_B Reset was not triggered by Global Reset Group x 1_B Reset was triggered by Global Reset Group x
0	7:1, 31:12	r	**Reserved** Read as 0; should be written with 0.

24.4.9 PROT register endinit

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner and in some PROT states, the secure master to update the PROT state as described in the PROT mechanism.

The PROTE register controls lock / unlock of the local Endinit (E) protected control registers.

PROTE Offset address: 020_H

PROT register endinit Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OWEN	ODEF	TAGID						PRSEN	PRS			VMEN	VM		
w	rw	rw						rw	rw			rw	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												SWEN	STATE		
r												w	rwh		

Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
VM	18:16	rw	**Virtual machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual machine definition enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.

(table continues...)

(continued)

Field	Bits	Type	Description
PRSEN	23	rw	**Protection set definition enable for PROT owner** 0_B **Disable**: PRS is not part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG-ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; should be written with 0.

Table 1119 Access mode restrictions of PROTE sorted by descending priority

Mode name	Access mode		Description
write 1 to .**SWEN**	rwh	STATE	
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

24.4.10 PROT register safe endinit

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner and in some PROT states, the secure master to update the PROT state as described in the PROT mechanism.

The PROTSE register controls lock / unlock of the local Safe Endinit (SE) protected control registers.

PROTSE Offset address: 024_H

PROT register safe endinit Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OWEN	ODEF	TAGID						PRSEN	PRS			VMEN	VM		
w	rw	rw						rw	rw			rw	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												SWEN	STATE		
r												w	rwh		

Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
VM	18:16	rw	**Virtual machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual machine definition enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.
PRSEN	23	rw	**Protection set definition enable for PROT owner** 0_B **Disable**: PRS is not part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG-ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; should be written with 0.

Table 1120 Access mode restrictions of PROTSE sorted by descending priority

Mode name	Access mode		Description
write 1 to .**SWEN**	rwh	STATE	
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

24.4.11 Write access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

ACCEN_WRA Offset address: 040_H

Write access enable register A Application Reset value: $1000\ 0003_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	**EN30**	**EN29**	**EN28**	**EN27**	**EN26**	**EN25**	**EN24**	**EN23**	**EN22**	**EN21**	**EN20**	**EN19**	**EN18**	**EN17**	**EN16**
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	**EN14**	**EN13**	**EN12**	**EN11**	**EN10**	**EN09**	**EN08**	**EN07**	**EN06**	**EN05**	**EN04**	**EN03**	**EN02**	**EN01**	**EN00**
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access

24.4.12 Write access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

ACCEN_WRB Offset address: 044_H

Write access enable register B Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								**EN39**	**EN38**	**EN37**	**EN36**	**EN35**	**EN34**	**EN33**	**EN32**
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access
0	31:8	r	**Reserved** Read as 0; should be written with 0.

24.4.13 Read access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

ACCEN_RDA Offset address: 048_H

Read access enable register A Application Reset value: $FFFF\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access

24.4.14 Read access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

ACCEN_RDB Offset address: $04C_H$

Read access enable register B Application Reset value: $0000\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access
0	31:8	r	**Reserved** Read as 0; should be written with 0.

24.4.15 VM access enable register

This register defines which virtual machine encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each VMx.

ACCEN_VM Offset address: 050_H

VM access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for VM ID q** This bit enables read access to the access protected region for transactions with the VM ID q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for VM ID q** This bit enables write access to the access protected region for transactions with the VM ID q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

24.4.16 PRS access enable register

This register defines which protection register set encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each PRSx.

ACCEN_PRS Offset address: 054_H

PRS access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for PRS q** This bit enables read access to the access protected region for transactions with the PRS q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for PRS q** This bit enables write access to the access protected region for transactions with the PRS q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

24.4.17 Input and output control register

The Input and Output Control Register IOCRx determines several properties of the RX and TX signal path:
For the RX signal:

- The alternate input
- The filter depth

For the TX signal:

- The trigger source

Note: IOCR is only writable, if CSR.CLKSEL=0.

IOCR Offset address: 100_H

Input and output control register Reset values see: Table 1121

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TXM	RXM	CTSEN	LB	SPOL	CPOL	RCPOL	0							CTS	
rh	rh	rw	rw	rw	rw	rw	r							rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0						DEPTH						0	ALTI		
r						rw						r	rw		

Field	Bits	Type	Description
ALTI	2:0	rw	**Alternate Input Select** Selects the alternate input for the RX signal: 000_B Alternate Input A selected 001_B Alternate Input B selected 010_B Alternate Input C selected 011_B Alternate Input D selected 100_B Alternate Input E selected 101_B Alternate Input F selected 110_B Alternate Input G selected 111_B Alternate Input H selected
DEPTH	9:4	rw	**Digital Glitch Filter Depth** DEPTH determines the number of port input samples clocked with microticks that are taken into account for the calculation of the floating average. The higher the DEPTH is chosen to be, the longer the glitches that are suppressed and the longer the delay of the input signal introduced by this filter. 00_H off, default 01_H 1 … $3F_H$ 63
CTS	17:16	rw	**CTS Select** Selects the CTS input pin out of maximum four possible. 00_B ACTSA 01_B ACTSB 10_B ACTSC 11_B ACTSD
RCPOL	25	rw	**RTS CTS Polarity** RCPOL defines the active level or the RTS and CTS signals. Active means ready/clear to send. 0_B Active high 1_B Active low
CPOL	26	rw	**Clock Polarity in Synchronous Mode** CPOL defines the idle level of the clock signal if the module is set in the SPI mode. The idle level is the level outside the data transmission time intervals. Default is low level. 0_B Idle low 1_B Idle high
SPOL	27	rw	**Slave Polarity in Synchronous Mode** Defines the idle level of the SLSO signal, which is the level outside the data transmission, leading and trailing time intervals. 0_B Idle low 1_B Idle high

(table continues...)

(continued)

Field	Bits	Type	Description
LB	28	rw	**Loop Back Mode** Enables the in module connection of the transmit signal to receive signal. If Loop-back is enabled, the module can be run and tested without an external connection, in ASC and SPI modes. In LIN mode, loopback should not be used, because the module can be either master or slave. 0_B Disabled 1_B Enabled
CTSEN	29	rw	**Input Signal CTS Enable** Enables the sensitivity of the module to the external CTS signal. If disabled, the CTS signal is considered being permanently active. 0_B Disabled 1_B Enabled
RXM	30	rh	**Receive Monitor** Shows the status of the receive signal. 0_B Current signal is low. 1_B Current signal is high.
TXM	31	rh	**Transmit Monitor** Shows the status of the transmit signal. 0_B Current signal is low. 1_B Current signal is high.
0	3, 15:10, 24:18	r	**Reserved** Read as 0; should be written with 0.

Table 1121 Reset values of IOCR

Reset type	Reset value	Note
After Boot-FW Value	––00 0000 0000 0000 0000 0000 0000 0001_B	Values of RXM and TXM depend on RX and TX signal value. After ASC BSL execution.

24.4.18 TxFIFO configuration register

TXFIFOCON Offset address: 104_H

TxFIFO configuration register Reset values see: Table 1122

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0			CIRCLEVEL					0			FILL				
r			rw					r			rh				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				INTLEVEL				INW		FM		0		ENO	FLUSH
r				rw				rw		rw		r		rw	w

Field	Bits	Type	Description
FLUSH	0	w	**Flush the transmit FIFO** Write of 1 brings the TxFIFO in empty state. Write of 0 has no effect. 0_B No effect / This value will be always read back from the Bit. 1_B Empty the Tx FIFO
ENO	1	rw	**Transmit FIFO Outlet Enable** Enables the TxFIFO outlet. 0_B Disabled.In ASC and SPI, no data is transmitted.In LIN case, no data is moved to the shift register even if the protocol engine tries to fetch data. 1_B Enabled.In SPI and ASC, data transmission starts immediately when the data is available. Except in ASC mode when trigger is enabled, a frame is transmitted for each trigger. In LIN case, no data is moved to the shift register until it is fetched by the protocol engine.
FM	5:4	rw	**TxFIFO Mode** Selects between the TxFIFO Modes. 00_B Combined Move Mode 01_B Single Move Mode 10_B Batch Move Mode 11_B Circular Mode
INW	7:6	rw	**Transmit FIFO Inlet Width** Defines the number of bytes written to the TxFIFO with one SPB bus write. 00_B 0 01_B 1 10_B 2 11_B 4
INTLEVEL	11:8	rw	**FIFO Interrupt Level** Defines the filling level that triggers a re-fill interrupt or DMA access. An interrupt is generated when the filling level falls to INTLEVEL or below, each time when a data byte is taken out of the FIFO. This behavior corresponds to the Combined/Compatibility Mode of interrupt generation. See also Single Move Mode and Batch Move Mode for two additional modes. Not used with circular buffer mode. 0_H 0 … F_H 15

(table continues...)

(continued)

Field	Bits	Type	Description
FILL	20:16	rh	**FIFO Filling Level** Read only bit-field containing the current filling level of the FIFO in bytes. 00_H 0 … 10_H 16 11_H Reserved … $1F_H$ Reserved
CIRCLEVEL	28:24	rw	**FIFO Circulate Level** Defines the circulating level in data bytes. When this level is reached, the next Transmit Trigger transmits the first element of FIFO. An interrupt is generated each time one full circle of data is transmitted. This mode is used only with the hardware or software Transmit Trigger, if enabled. Note: User should make sure that CIRCLEVEL is equal with FIFO fill level. 00_H Not allowed Note: If this is selected, no trasmission is possible. 01_H 1 … 10_H 16 **others,** Reserved Note: In case this is selected the circular buffer is assumed for maximum FIFO storage (16).
0	3:2, 15:12, 23:21, 31:29	r	**Reserved** Read as 0; should be written with 0.

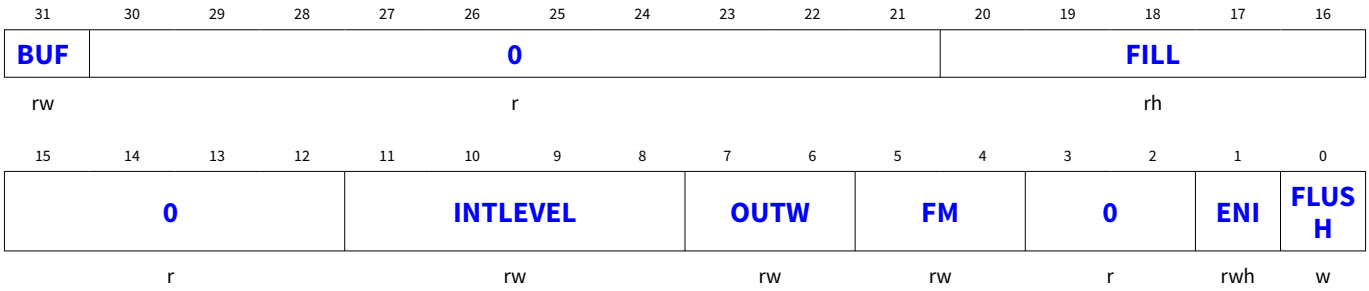
Table 1122 Reset values of TXFIFOCON

Reset type	Reset value	Note
Kernel Reset	$0000\ 0000_H$	
After Boot-FW Value	$0000\ 0142_H$	After ASC BSL execution

24.4.19 RxFIFO configuration register

RXFIFOCON Offset address: 108_H

RxFIFO configuration register Reset values see: Table 1123



Field	Bits	Type	Description
FLUSH	0	w	**Flush the receive FIFO** Write of 1 brings the RxFIFO in empty state. Write of 0 has no effect. 0_B No effect / This value will be always read back from the module. 1_B Empty the Rx FIFO
ENI	1	rwh	**Receive FIFO Inlet Enable** Enables the receiver and the filling of the RxFIFO through the shift register. In LIN slave mode, this bit is set by hardware after the correct reception of the sync byte. The software can clear this bit after reception of an foreign ID in order to suppress the reception of the following response. 0_B Disabled 1_B Enabled
FM	5:4	rw	**RxFIFO Mode** Selects between the RxFIFO Modes. 00_B Combined Move Mode 01_B Single Move Mode 10_B Batch Move Mode 11_B reserved
OUTW	7:6	rw	**Receive FIFO Outlet Width** Defines the number of bytes read to the RxFIFO with one SPB bus read. Note: If RXFIFO is configured as receive buffer mode (RXFIFOCON.BUF=1), then OUTW setting is ignored while reading data. 00_B 0 01_B 1 10_B 2 11_B 4
INTLEVEL	11:8	rw	**FIFO Interrupt Level** Defines the filling level that triggers a drain interrupt or DMA access. An interrupt is generated when the filling level rises to INTLEVEL or beyond, each time when a data byte is delivered to the FIFO. This behavior corresponds to the Combined/Compatibility Mode of interrupt generation. See also Single Move Mode and Batch Move Mode for two additional modes. 0_H 1 … F_H 16

(table continues...)

(continued)

Field	Bits	Type	Description
FILL	20:16	rh	**FIFO Filling Level** Read only bit-field containing the current filling level of the FIFO in bytes. 00_H 0 … 10_H 16 11_H Reserved … $1F_H$ Reserved
BUF	31	rw	**Receive Buffer Mode** If this bit is zero, then the RXFIFO behaves normally as described in this document. If this bit is set, the RXFIFO behaves as simple 32-bit one stage RX buffer, which is overwritten with each new received data. The received bits appear in the RXDATA register on the lowest bit locations. The upper locations are padded with zeros. 0_B RXFIFO 1_B Single Stage RX Buffer
0	3:2, 15:12, 30:21	r	**Reserved** Read as 0; should be written with 0.

Table 1123 Reset values of RXFIFOCON

Reset type	Reset value	Note
Kernel Reset	$0000\ 0000_H$	
After Boot-FW Value	$8000\ 0002_H$	After ASC BSL execution

24.4.20 Bit configuration register

The BITCON Register defines the integer timer parameters in the baud rate generation block.

Note: The register is only writable, if CSR.CLKSEL=0.

BITCON Offset address: $10C_H$

Bit configuration register Reset values see: Table 1124

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SM	0			SAMPLEPOINT				0				OVERSAMPLING			
rw	r			rw				r				rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				PRESCALER											
r				rw											

Field	Bits	Type	Description
PRESCALER	11:0	rw	**Prescaling of the Fractional Divider** Prescaler bit-field with values in the range of 0 to 4095, defining division ratios from 1 to 4096. Used also as a microtick generator for the input digital filter.
OVERSAMPLING	19:16	rw	**Oversampling Factor** Defines the bit length in ticks in the range of 1 to 16. The lengths of 1 to 3 are not allowed. The position of the sampling points is shown in Bit Timing Properties chapter. 0_H 1 (not allowed) 1_H 2 (not allowed) 2_H 3 (not allowed) 3_H 4 … F_H 16
SAMPLEPOINT	27:24	rw	**Sample Point Position** Programmable in the range of 0 to 15, see the Bit Timing Properties chapter for more details. For example, if three sample points at position 7, 8, 9 are required, this bit field would contain 9. In SPI mode, this bit field + 1 defines the length of the first SCLK half period in ticks. Values equal or higher then the OVERSAMPLING value are forbidden. 0_H 0 (not allowed) 1_H 1 … F_H 15
SM	31	rw	**Sample Mode** Number of samples per bit. 0_B 1 1_B 3
0	15:12, 23:20, 30:28	r	**Reserved** Read as 0; should be written with 0.

Table 1124 Reset values of BITCON

Reset type	Reset value	Note
Kernel Reset	$0000\ 0000_H$	
After Boot-FW Value	$4070\ 0000_H$	After ASC BSL execution

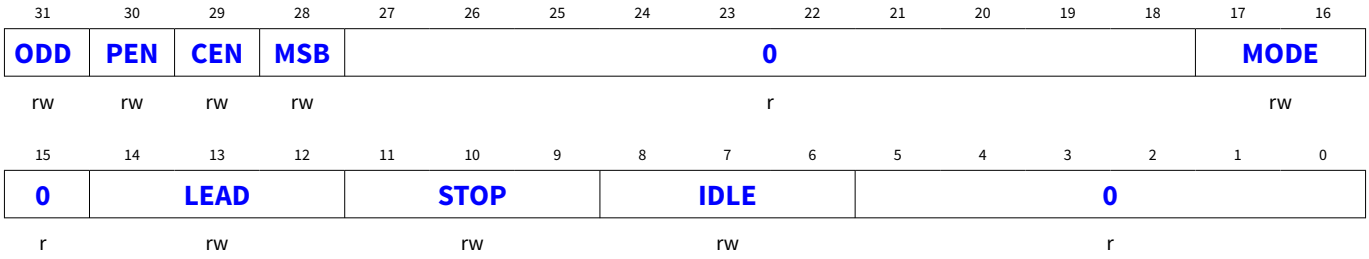
24.4.21 Frame control register

The parameters regarding the properties of the message frame of the ASC communication are controlled by the Frame Control Register FRAMECON.

Note: The register is only writable, if CSR.CLKSEL=0.

FRAMECON Offset address: 110_H

Frame control register Reset values see: Table 1125



Field	Bits	Type	Description
IDLE	8:6	rw	**Duration of the IDLE delay** Defines the duration of the IDLE delay in bit times. If more characters are available in the TxFIFO, this is the pause inserted between the characters. In the SPI mode, this is the idle time between the frames. In the ASC and LIN mode, this is the pause inserted between transmission of bytes. Idle also applies to the pause between the header and the response (response space). ***Note**: The collision detection runs in parallel to the idle delay and in LIN master mode it may extend the time between two bytes for one bit length. This effect may occur if the round trip delay including the digital filter delay is longer than the idle delay. For LIN slave task use IDLE=0.* 000_B 0 … 111_B 7
STOP	11:9	rw	**Number of Stop Bits** Defines the number of stop bits in ASC and LIN mode, or the trailing delay in SPI mode. In ASC mode, standard values are 1 and 2. In LIN mode, standard value is 1. In SPI mode there is no standard value. Nevertheless, all settings are possible in all modes. 000_B 0 (not allowed in ASC and LIN modes) 001_B 1 … 111_B 7

(table continues...)

(continued)

Field	Bits	Type	Description
LEAD	14:12	rw	**Duration of the Leading Delay** Defines the leading delay in bit times in SPI mode. Has no meaning in the ASC mode. In LIN mode, this is a delay inserted between the end of the break and the start of the sync character. 000_B 0 (not allowed in LIN mode and 4-Wire SPI mode) 001_B 1 … 111_B 7
MODE	17:16	rw	**Mode Selection** This bit field defines the basic operating mode of the module. In INIT mode, all outputs are at inactive level, and the module does not respond to the input signals. Changing the mode of the module must be done by switching first to INIT mode, and then to the other mode. The SCLK signal generated by the module is active only in the SPI mode. The CTS output generated by the module is active only in the ASC mode. 00_B INIT mode 01_B ASC mode 10_B SPI mode 11_B LIN mode
MSB	28	rw	**Shift Direction** Defines the shift direction of the shift register. Relevant for the SPI and ASC mode. In LIN mode, should be set to zero. Parity bit is shifted out last independently of the shift direction. 0_B LSB first 1_B MSB first
CEN	29	rw	**Collision Detection Enable** Enables the collision detection mechanism. 0_B Disabled 1_B Enabled
PEN	30	rw	**Parity Enable** Enables the parity bit attached to the data bits. Parity bit can be used for ASC and SPI protocols. The standard LIN bytes do not use this parity bit. 0_B Disabled 1_B Enabled
ODD	31	rw	**Parity Type** Defines the type of parity bit attached to the data bits. This setting is valid for all modes of operation (ASC, LIN, SPI). 0_B Even 1_B Odd

(table continues...)

(continued)

Field	Bits	Type	Description
0	5:0, 15, 27:18	r	**Reserved** Read as 0; should be written with 0.

Table 1125 Reset values of FRAMECON

Reset type	Reset value	Note
Kernel Reset	0000 0000$_H$	
After Boot-FW Value	0001 0200$_H$	After ASC BSL execution

24.4.22 Data configuration register

DATCON register defines the number of bits in the ASC (affecting the LIN protocol, if used) and SPI frames, data length or the number of data bytes in a LIN response (if any) and the checksum mode. It additionally defines the time window for the LIN response.

In case the whole LIN response does not fit in this time window, an error interrupt is generated. The measurement starts at the end of the last bit of the header, and ends at the end of the last bit of the response.

DATCON Offset address: 114$_H$

Data configuration register Kernel Reset value: 0000 0000$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								RESPONSE							
r								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CSM	RM	HO	0			TRGSRC		0				DATLEN			
rw	rw	rw	r			rw		r				rw			

Field	Bits	Type	Description
DATLEN	3:0	rw	**Data Length** Defines the number of bits in a character. In the ASC mode, standard length is 7, 8, or 9 bits. In the SPI mode, there is no standard length. In ASC and SPI modes, any length from 2 to 16 bits is possible, although not standard for some protocols. In LIN mode, standard length is 8 bits per character. Therefore, this field defines the number of data bytes of the response. Allowed configuration is 0 to 7 to be compliant with LIN specification. 0_H 1 … F_H 16

(table continues...)

(continued)

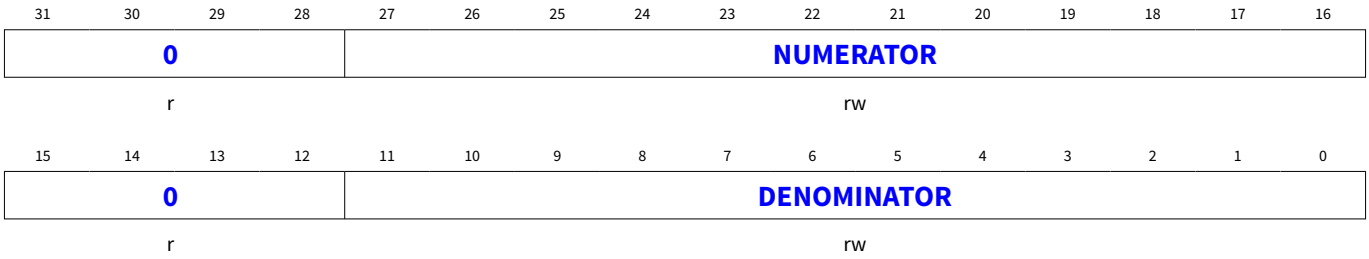
Field	Bits	Type	Description
TRGSRC	9:8	rw	**Trigger Source** This bit selects the trigger source for the triggered transmit in ASC mode. 00_B Trigger disabled. 01_B GTM trigger 10_B eGTM trigger 11_B Software trigger
HO	13	rw	**Header Only** Defines if the LIN frame shall consist of a header and response or of a header only. 0_B Header and response expected 1_B Header only expected, response ignored
RM	14	rw	**Response Mode** Defines if the RESPONSE bit field defines a LIN Frame or LIN Response timeout threshold. 0_B Frame 1_B Response
CSM	15	rw	**Checksum Mode** Defines if the classic or the enhanced checksum will be calculated by the checksum block. 0_B Classic 1_B Enhanced
RESPONSE	23:16	rw	**Response Timeout Threshold Value** Defines the timer limit in the range of 1 to 256 bit times. 00_H 1 … FF_H 256
0	7:4, 12:10, 31:24	r	**Reserved** Read as 0; should be written with 0.

24.4.23 Baud rate generation register

Configures the numerator and the denominator of the fractional divider in the baud rate generation block.
Note: The register is only writable, if CSR.CLKSEL=0.

BRG Offset address: 118_H

Baud rate generation register Reset values see: Table 1126



Field	Bits	Type	Description
DENOMINATOR	11:0	rw	**Denominator** Programmed by software, in a range of 0 to 4095. The setting of 0 is not allowed If the module is used as ASC, SPI, LIN master and LIN slave without autobaud detection, this value determines the baud rate. In slave mode with autobaud detection, it contains the nominal value. For the value measured by the autobaud detection hardware, see the BRD register.
NUMERATOR	27:16	rw	**Numerator** Defines the numerator of the fractional divider in a range of 0 to 4095. Programmed by software. The setting of 0 is not allowed.
0	15:12, 31:28	r	**Reserved** Read as 0; should be written with 0.

Table 1126 Reset values of BRG

Reset type	Reset value	Note
Kernel Reset	0000 0000$_H$	
After Boot-FW Value	0048 0–––$_H$	After ASC BSL execution

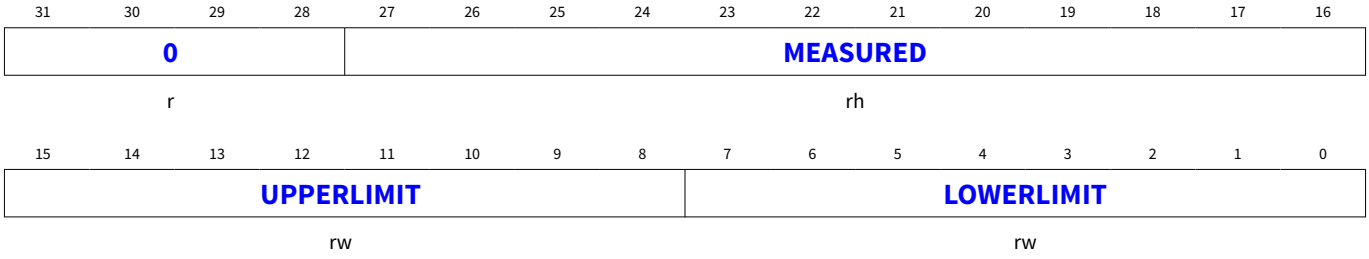
24.4.24 Baud rate detection register

The BRD defines the properties specific for the automatic baud rate detection when the module operates as a LIN slave.

Note: The register is only writable, if CSR.CLKSEL=0.

BRD Offset address: 11C$_H$

Baud rate detection register Kernel Reset value: 0000 0000$_H$



Field	Bits	Type	Description
LOWERLIMIT	7:0	rw	**Lower Limit** This field defines the 8 most significant bits of the 12 bit compare value. The lower four bits are 1000_B
UPPERLIMIT	15:8	rw	**Upper Limit** This field defines the 8 most significant bits of the 12 bit compare value. The lower four bits are 1000_B.
MEASURED	27:16	rh	**Measured Value of 8-bits from Sync Field** This bit field contains the measured value of the duration of 8-bits form the sync field of the LIN header in microticks. It is automatically loaded in the denominator of the fractional divider, in case of LIN slave operation with autobaud detection.
0	31:28	r	**Reserved** Read as 0; should be written with 0.

24.4.25 LIN control register

LINCON contains bits that control LIN specific features of the module.

Note: The register is only writable, if CSR.CLKSEL=0.

LINCON Offset address: 120_H

LIN control register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0				ABD	MS	CSEN	0	CSI	0						
r				rw	rw	rw	r	rw	r						

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															
r															

Field	Bits	Type	Description
CSI	23	rw	**Checksum Injection** Defines if the received checksum byte is written into the RXFIFO or not. 0_B Not written 1_B Written
CSEN	25	rw	**Hardware Checksum Enable** Enables the hardware checksum generation and checking. 0_B Disabled 1_B Enabled
MS	26	rw	**Master Slave Mode** Configures if the module in LIN mode operates as master or as slave. 0_B Slave 1_B Master

(table continues...)

(continued)

Field	Bits	Type	Description
ABD	27	rw	**Autobaud Detection** Enables the autobaud detection feature in LIN slave mode. In all other operating modes of the module (LIN master, ASC, SPI) not effective. If the autobaud detection is disabled (the oscillator precision of the slave is sufficient), the sync field (byte field and stop bit) is checked if correct. If not correct, a framing error is triggered. 0_B Disabled 1_B Enabled
0	22:0, 24, 31:28	r	**Reserved** Read as 0; should be written with 0.

24.4.26 LIN break timer register

This register defines

- break detection limit if the module operates as LIN slave or
- length of the generated break pulse if the module operates as LIN master.

The break timer, if enabled, monitors the bus continuously.

Note: The register is only writable, if CSR.CLKSEL=0.

LINBTIMER Offset address: 124_H

LIN break timer register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0										BREAK					
r										rw					

Field	Bits	Type	Description
BREAK	5:0	rw	**Break Pulse Generation and Detection** In LIN slave mode, this bit field defines the duration of the detection threshold for the break pulse. In LIN master mode, this bit field defines the duration of the transmitted break pulse. The time unit is bit time.
0	31:6	r	**Reserved** Read as 0; should be written with 0.

24.4.27 LIN header timer register

LINHTIMER register defines the time windows for the header of a LIN frame.

In master mode, the timer starts counting at the falling edge of the break pulse, and stops counting when the last bit of the header (including the stop bits) has been transmitted. If the predefined time in the HEADER bit field is violated, an error interrupt is generated (if enabled).

In slave mode, the timer starts counting when the break pulse has been detected, after a low time of 10 or 11 bit times, and stops counting when the last bit of the header has been received. If the predefined time in the HEADER bit field is violated, an error interrupt is generated (if enabled).

Note: The register is only writable, if CSR.CLKSEL=0.

LINHTIMER Offset address: 128_H

LIN header timer register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								HEADER							
r								rw							

Field	Bits	Type	Description
HEADER	7:0	rw	**Header Timeout Threshold Value** Defines the timer limit in the range of 0 to 255 bit times.
0	31:8	r	**Reserved** Read as 0; should be written with 0.

24.4.28 Flags register

The FLAGS register contains all the flag bits of the ASCLIN module: the LIN phase flags (header and response transmit and receive), the overflow flags of the LIN timers, and the standard ASC error flags.

A corresponding interrupt triggering can be enabled using the FLAGSENABLE register.

FLAGS Offset address: $12C_H$

Flags register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TFL	TFO	0	RFL	RFU	RFO	CE	LC	LA	LP	BD	RT	HT	FE	TC	PE
rh	rh	r	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TRRQ	THRQ	TWRQ	SWTRG	OMT	TFE	0			RED	FED	0	RR	RH	TR	TH
rh	rh	rh	rh	rh	rh	r			rh	rh	r	rh	rh	rh	rh

Field	Bits	Type	Description
TH	0	rh	**Transmit Header End Flag** Signals the HEADER_TX_END event. Set by hardware, clear by software. If enabled, a transmit interrupt is triggered. 0_B No HEADER_TX_END event since the last clear by software 1_B New HEADER_TX_END event since the last clear by software
TR	1	rh	**Transmit Response End Flag** Signals that RESPONSE_TX_END event. Set by hardware, clear by software. If enabled, a transmit interrupt is triggered. 0_B No RESPONSE_TX_END event since the last clear by software 1_B New RESPONSE_TX_END event since the last clear by software
RH	2	rh	**Receive Header End Flag** Signals that HEADER_RX_END event. Set by hardware, clear by software. If enabled, a receive interrupt is triggered. 0_B No HEADER_RX_END event since the last clear by software 1_B New HEADER_RX_END event since the last clear by software
RR	3	rh	**Receive Response End Flag** Signals that RESPONSE_RX_END event. Set by hardware, clear by software. If enabled, a receive interrupt is triggered. 0_B No RESPONSE_RX_END event since the last clear by software 1_B New RX_RESPONSE_END event since the last clear by software
FED	5	rh	**Falling Edge from Level 1 to Level 0 Detected** This bit is set by hardware when a falling edge is detected on the RX line. 0_B No falling edge detected 1_B Falling edge detected
RED	6	rh	**Rising Edge from Level 0 to Level 1 Detected** This bit is set by hardware when a rising edge is detected on the RX line. 0_B No rising edge detected 1_B Rising edge detected
TFE	10	rh	**Transmit FIFO Empty Flag** Signals an error whenever TxFIFO is empty or not enough data for one frame is available and a transmit request was triggered by either hardware or software. If enabled, an error interrupt is triggered. 0_B No empty error occurred 1_B TxFIFO empty error occurred

(table continues...)

(continued)

Field	Bits	Type	Description
OMT	11	rh	**Overload of Multiple Triggers** Signals that if during ongoing transmission there is one or more hardware or software trigger requests. If enabled, an error interrupt is triggered. 0_B No overload of multiple triggers 1_B Multiple trigger requested during ongoing transmission
SWTRG	12	rh	**Software Trigger Transmit Request Flag** Signals that a software trigger frame transmit has been requested. As soon as the frame transmission starts, the bit is cleared by the hardware. No interrupt is triggered. 0_B No Frame Transmit Request pending 1_B Frame Transmit Request pending.
TWRQ	13	rh	**Transmit Wake Request Flag** Signals that transmission of wake has been requested. No interrupt triggered. As soon as the wake pulse transmission starts, the bit is cleared by the hardware. 0_B No Transmit Wake Request pending 1_B Transmit Wake Request pending.
THRQ	14	rh	**Transmit Header Request Flag** Signals that transmission of header has been requested. No interrupt triggered. As soon as the header transmission starts, the bit is cleared by the hardware. 0_B No Transmit Header Request pending 1_B Transmit Header Request pending.
TRRQ	15	rh	**Transmit Response Request Flag** Signals that transmission of response has been requested. No interrupt triggered. As soon as the response transmission starts, the bit is cleared by the hardware. 0_B No Transmit Response Request pending 1_B Transmit Response Request pending.
PE	16	rh	**Parity Error Flag** Signals parity error. If enabled, an error interrupt is triggered. Parity error occurs if the internally calculated parity bit is not equal to the received parity bit. Note: This bit is only used with ASC and SPI mode. 0_B Last message received error free 1_B Last message received with parity error

(table continues...)

(continued)

Field	Bits	Type	Description
TC	17	rh	**Transmission Completed Flag** Signals an end of an ASC or SPI frame. This bit is set after the last stop bit transmission in ASC mode, or after the trailing delay in case of SPI mode. In LIN mode, if the node is transmitting a header this flag is set after each transmission break (incl lead) field, Sync field or PID field. If the node is transmitting a response, this flag is set after each byte is transmitted. If enabled, an ERR_INT interrupt is triggered. Should be cleared by software. 0_B No end of frame event occurred 1_B End of frame event occurred
FE	18	rh	**Framing Error Flag** Signals framing error. If enabled, an error interrupt is triggered. Framing error occurs if “0” is received at a stop bit position. If autobaud detection is deactivated, then the sync field is checked for framing error. 0_B Last message received error free 1_B Last message received with error
HT	19	rh	**Header Timeout Flag** Signals violation of the header duration limit. If enabled, an error interrupt is triggered. 0_B No HEADER_OVERFLOW event since the last clear by software 1_B New HEADER_OVERFLOW event since the last clear by software
RT	20	rh	**Response Timeout Flag** Signals violation of the response or frame duration limit as defined in DATCON.RM bit. If enabled, an error interrupt is triggered. 0_B No timeout event since the last clear by software 1_B New timeout event since the last clear by software
BD	21	rh	**Break Detected Flag** Signals a detection of a break pulse. If enabled, an error interrupt is triggered. Slave mode only. 0_B No Break/Wake/Stuck event since the last clear by software 1_B New Break/Wake/Stuck event since the last clear by software
LP	22	rh	**LIN Parity Error Flag** Signals parity error in the LIN identifier. If enabled, an error interrupt is triggered. Applies to LIN mode only. LIN parity error occurs if the internally calculated parity bits are not equal to the received parity bits. 0_B Last ID received error free 1_B Last ID received with parity error

(table continues...)

(continued)

Field	Bits	Type	Description
LA	23	rh	**LIN Autobaud Detection Error Flag** Signals baudrate outside the range defined by BRD.LOWERLIMIT and BRD.UPPERLIMIT. 0_B No autobaud detection error 1_B Autobaud detection error
LC	24	rh	**LIN Checksum Error Flag** Signals checksum error when receiving response, if the internally calculated checksum is different than the received checksum. If enabled, an error interrupt is triggered. 0_B Last checksum error free 1_B Last checksum shows an error
CE	25	rh	**Collision Detection Error Flag** When transmitting, signals if the transmitted data differs from the received data. If enabled, an error interrupt is triggered in case of a mismatch. Collision detection is mandatory only when supporting LIN version 2.1. 0_B No mismatch 1_B Mismatch detected
RFO	26	rh	**Receive FIFO Overflow Flag** Signals an overflow error. If enabled, an error interrupt is triggered. 0_B No overflow error occurred 1_B Overflow error occurred
RFU	27	rh	**Receive FIFO Underflow Flag** Signals an underflow error. If enabled, an error interrupt is triggered. See also RxFIFO. 0_B No underflow error occurred 1_B Underflow error occurred
RFL	28	rh	**Receive FIFO Level Flag** This flag signals whenever a RXFIFO fill interrupt is generated based on RXFIFOCON.FM mode. 0_B No receive interrupt occurred 1_B Receive interrupt occurred
TFO	30	rh	**Transmit FIFO Overflow Flag** Signals an overflow error. If enabled, an error interrupt is triggered. 0_B No overflow error occurred 1_B Overflow error occurred
TFL	31	rh	**Transmit FIFO Level Flag** This flag signals whenever a TxFIFO refill interrupt (if enabled) is generated based on TXFIFOCON.FM mode. 0_B No transmit interrupt occurred 1_B Transmit interrupt occurred

(table continues...)

(continued)

Field	Bits	Type	Description
0	4, 9:7, 29	r	**Reserved** Read as 0; should be written with 0.

24.4.29 Flags set register

The FLAGSSET register contains the write only bits used to set the corresponding bits in the FLAGS register by software. Setting a flag bit triggers an interrupt, if the corresponding interrupt enable bit is set. A read access to the register always results in 0.

FLAGSSET Offset address: 130_H

Flags set register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TFLS	TFOS	0	RFLS	RFUS	RFOS	CES	LCS	LAS	LPS	BDS	RTS	HTS	FES	TCS	PES
w	w	r	w	w	w	w	w	w	w	w	w	w	w	w	w

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TRRQS	THRQS	TWRQS	SWTRGS	OMTS	TFES	0			REDS	FEDS	0	RRS	RHS	TRS	THS
w	w	w	w	w	w	r			w	w	r	w	w	w	w

Field	Bits	Type	Description
THS	0	w	**Flag Set Bit** Write of “1” in this bit sets the bit at the corresponding position in the FLAGS register. 0_B No action / This value will be always read back from the Bit. 1_B Set the corresponding flag
TRS	1	w	**Flag Set Bit** Write of “1” in this bit sets the bit at the corresponding position in the FLAGS register. 0_B No action / This value will be always read back from the Bit. 1_B Set the corresponding flag
RHS	2	w	**Flag Set Bit** Write of “1” in this bit sets the bit at the corresponding position in the FLAGS register. 0_B No action / This value will be always read back from the Bit. 1_B Set the corresponding flag
RRS	3	w	**Flag Set Bit** Write of “1” in this bit sets the bit at the corresponding position in the FLAGS register. 0_B No action / This value will be always read back from the Bit. 1_B Set the corresponding flag

(table continues...)

(continued)

Field	Bits	Type	Description
FEDS	5	w	**Flag Set Bit** Write of “1” in this bit sets the bit at the corresponding position in the FLAGS register. 0_B No action / This value will be always read back from the Bit. 1_B Set the corresponding flag
REDS	6	w	**Flag Set Bit** Write of “1” in this bit sets the bit at the corresponding position in the FLAGS register. 0_B No action / This value will be always read back from the Bit. 1_B Set the corresponding flag
TFES	10	w	**Flag Set Bit** Write of “1” in this bit sets the bit at the corresponding position in the FLAGS register. 0_B No action / This value will be always read back from the Bit. 1_B Set the corresponding flag
OMTS	11	w	**Flag Set Bit** Write of “1” in this bit sets the bit at the corresponding position in the FLAGS register. 0_B No action / This value will be always read back from the Bit. 1_B Set the corresponding flag
SWTRGS	12	w	**Flag Set Bit** If software trigger is enabled, write of “1” in this bit sets the bit at the corresponding position in the FLAGS register. In all other cases, write of this bit has no action. 0_B No action / This value will be always read back from the Bit. 1_B Set the corresponding flag
TWRQS	13	w	**Flag Set Bit** Write of “1” in this bit sets the bit at the corresponding position in the FLAGS register. 0_B No action / This value will be always read back from the Bit. 1_B Set the corresponding flag
THRQS	14	w	**Flag Set Bit** Write of “1” in this bit sets the bit at the corresponding position in the FLAGS register. 0_B No action / This value will be always read back from the Bit. 1_B Set the corresponding flag
TRRQS	15	w	**Flag Set Bit** Write of “1” in this bit sets the bit at the corresponding position in the FLAGS register. 0_B No action / This value will be always read back from the Bit. 1_B Set the corresponding flag

(table continues...)

(continued)

Field	Bits	Type	Description
PES	16	w	**Flag Set Bit** Write of “1” in this bit sets the bit at the corresponding position in the FLAGS register. 0_B No action / This value will be always read back from the Bit. 1_B Set the corresponding flag
TCS	17	w	**Flag Set Bit** Write of “1” in this bit sets the bit at the corresponding position in the FLAGS register. 0_B No action / This value will be always read back from the Bit. 1_B Set the corresponding flag
FES	18	w	**Flag Set Bit** Write of “1” in this bit sets the bit at the corresponding position in the FLAGS register. 0_B No action / This value will be always read back from the Bit. 1_B Set the corresponding flag
HTS	19	w	**Flag Set Bit** Write of “1” in this bit sets the bit at the corresponding position in the FLAGS register. 0_B No action / This value will be always read back from the Bit. 1_B Set the corresponding flag
RTS	20	w	**Flag Set Bit** Write of “1” in this bit sets the bit at the corresponding position in the FLAGS register. 0_B No action / This value will be always read back from the Bit. 1_B Set the corresponding flag
BDS	21	w	**Flag Set Bit** Write of “1” in this bit sets the bit at the corresponding position in the FLAGS register. 0_B No action / This value will be always read back from the Bit. 1_B Set the corresponding flag
LPS	22	w	**Flag Set Bit** Write of “1” in this bit sets the bit at the corresponding position in the FLAGS register. 0_B No action / This value will be always read back from the Bit. 1_B Set the corresponding flag
LAS	23	w	**Flag Set Bit** Write of “1” in this bit sets the bit at the corresponding position in the FLAGS register. 0_B No action / This value will be always read back from the Bit. 1_B Set the corresponding flag

(table continues...)

(continued)

Field	Bits	Type	Description
LCS	24	w	**Flag Set Bit** Write of "1" in this bit sets the bit at the corresponding position in the FLAGS register. 0_B No action / This value will be always read back from the Bit. 1_B Set the corresponding flag
CES	25	w	**Flag Set Bit** Write of "1" in this bit sets the bit at the corresponding position in the FLAGS register. 0_B No action / This value will be always read back from the Bit. 1_B Set the corresponding flag
RFOS	26	w	**Flag Set Bit** Write of "1" in this bit sets the bit at the corresponding position in the FLAGS register. 0_B No action / This value will be always read back from the Bit. 1_B Set the corresponding flag
RFUS	27	w	**Flag Set Bit** Write of "1" in this bit sets the bit at the corresponding position in the FLAGS register. 0_B No action / This value will be always read back from the Bit. 1_B Set the corresponding flag
RFLS	28	w	**Flag Set Bit** Write of "1" in this bit sets the bit at the corresponding position in the FLAGS register. 0_B No action / This value will be always read back from the Bit. 1_B Set the corresponding flag
TFOS	30	w	**Flag Set Bit** Write of "1" in this bit sets the bit at the corresponding position in the FLAGS register. 0_B No action / This value will be always read back from the Bit. 1_B Set the corresponding flag
TFLS	31	w	**Flag Set Bit** Write of "1" in this bit sets the bit at the corresponding position in the FLAGS register. 0_B No action / This value will be always read back from the Bit. 1_B Set the corresponding flag
0	4, 9:7, 29	r	**Reserved** Read as 0; should be written with 0.

24.4.30 Flags clear register

The FLAGSCLEAR register contains the write only bits used to clear the corresponding bits in the FLAGS register.

FLAGSCLEAR Offset address: 134_H

Flags clear register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TFLC	TFOC	0	RFLC	RFUC	RFOC	CEC	LCC	LAC	LPC	BDC	RTC	HTC	FEC	TCC	PEC
w	w	r	w	w	w	w	w	w	w	w	w	w	w	w	w

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TRRQC	THRQC	TWRQC	SWTRGC	OMTC	TFEC	0			REDC	FEDC	0	RRC	RHC	TRC	THC
w	w	w	w	w	w	r			w	w	r	w	w	w	w

Field	Bits	Type	Description
THC	0	w	**Flag Clear Bit** Write of “1” in this bit clears the bit at the corresponding position in the FLAGS register. 0_B No action / This value will be always read back from the Bit. 1_B Clear the corresponding flag
TRC	1	w	**Flag Clear Bit** Write of “1” in this bit clears the bit at the corresponding position in the FLAGS register. 0_B No action / This value will be always read back from the Bit. 1_B Clear the corresponding flag
RHC	2	w	**Flag Clear Bit** Write of “1” in this bit clears the bit at the corresponding position in the FLAGS register. 0_B No action / This value will be always read back from the Bit. 1_B Clear the corresponding flag
RRC	3	w	**Flag Clear Bit** Write of “1” in this bit clears the bit at the corresponding position in the FLAGS register. 0_B No action / This value will be always read back from the Bit. 1_B Clear the corresponding flag
FEDC	5	w	**Flag Clear Bit** Write of “1” in this bit clears the bit at the corresponding position in the FLAGS register. 0_B No action / This value will be always read back from the Bit. 1_B Clear the corresponding flag
REDC	6	w	**Flag Clear Bit** Write of “1” in this bit clears the bit at the corresponding position in the FLAGS register. 0_B No action / This value will be always read back from the Bit. 1_B Clear the corresponding flag

(table continues...)

(continued)

Field	Bits	Type	Description
TFEC	10	w	**Flag Clear Bit** Write of “1” in this bit clears the bit at the corresponding position in the FLAGS register. 0_B No action / This value will be always read back from the Bit. 1_B Clears the corresponding flag
OMTC	11	w	**Flag Clear Bit** Write of “1” in this bit clears the bit at the corresponding position in the FLAGS register. 0_B No action / This value will be always read back from the Bit. 1_B Clears the corresponding flag
SWTRGC	12	w	**Flag Clear Bit** Write of “1” in this bit clears the bit at the corresponding position in the FLAGS register. 0_B No action / This value will be always read back from the Bit. 1_B Clear the corresponding flag
TWRQC	13	w	**Flag Clear Bit** Write of “1” in this bit clears the bit at the corresponding position in the FLAGS register. 0_B No action / This value will be always read back from the Bit. 1_B Clear the corresponding flag
THRQC	14	w	**Flag Clear Bit** Write of “1” in this bit clears the bit at the corresponding position in the FLAGS register. 0_B No action / This value will be always read back from the Bit. 1_B Clear the corresponding flag
TRRQC	15	w	**Flag Clear Bit** Write of “1” in this bit clears the bit at the corresponding position in the FLAGS register. 0_B No action / This value will be always read back from the Bit. 1_B Clear the corresponding flag
PEC	16	w	**Flag Clear Bit** Write of “1” in this bit clears the bit at the corresponding position in the FLAGS register. 0_B No action / This value will be always read back from the Bit. 1_B Clears the corresponding flag
TCC	17	w	**Flag Clear Bit** Write of “1” in this bit clears the bit at the corresponding position in the FLAGS register. 0_B No action / This value will be always read back from the Bit. 1_B Clears the corresponding flag

(table continues...)

(continued)

Field	Bits	Type	Description
FEC	18	w	**Flag Clear Bit** Write of “1” in this bit clears the bit at the corresponding position in the FLAGS register. 0_B No action / This value will be always read back from the Bit. 1_B Clears the corresponding flag
HTC	19	w	**Flag Clear Bit** Write of “1” in this bit clears the bit at the corresponding position in the FLAGS register. 0_B No action / This value will be always read back from the Bit. 1_B Clears the corresponding flag
RTC	20	w	**Flag Clear Bit** Write of “1” in this bit clears the bit at the corresponding position in the FLAGS register. 0_B No action / This value will be always read back from the Bit. 1_B Clears the corresponding flag
BDC	21	w	**Flag Clear Bit** Write of “1” in this bit clears the bit at the corresponding position in the FLAGS register. 0_B No action / This value will be always read back from the Bit. 1_B Clears the corresponding flag
LPC	22	w	**Flag Clear Bit** Write of “1” in this bit clears the bit at the corresponding position in the FLAGS register. 0_B No action / This value will be always read back from the Bit. 1_B Clears the corresponding flag
LAC	23	w	**Flag Clear Bit** Write of “1” in this bit clears the bit at the corresponding position in the FLAGS register. 0_B No action / This value will be always read back from the Bit. 1_B Clears the corresponding flag
LCC	24	w	**Flag Clear Bit** Write of “1” in this bit clears the bit at the corresponding position in the FLAGS register. 0_B No action / This value will be always read back from the Bit. 1_B Clears the corresponding flag
CEC	25	w	**Flag Clear Bit** Write of “1” in this bit clears the bit at the corresponding position in the FLAGS register. 0_B No action / This value will be always read back from the Bit. 1_B Clears the corresponding flag

(table continues...)

(continued)

Field	Bits	Type	Description
RFOC	26	w	**Flag Clear Bit** Write of “1” in this bit clears the bit at the corresponding position in the FLAGS register. 0_B No action / This value will be always read back from the Bit. 1_B Clears the corresponding flag
RFUC	27	w	**Flag Clear Bit** Write of “1” in this bit clears the bit at the corresponding position in the FLAGS register. 0_B No action / This value will be always read back from the Bit. 1_B Clears the corresponding flag
RFLC	28	w	**Flag Clear Bit** Write of “1” in this bit clears the bit at the corresponding position in the FLAGS register. 0_B No action / This value will be always read back from the Bit. 1_B Clears the corresponding flag
TFOC	30	w	**Flag Clear Bit** Write of “1” in this bit clears the bit at the corresponding position in the FLAGS register. 0_B No action / This value will be always read back from the Bit. 1_B Clears the corresponding flag
TFLC	31	w	**Flag Clear Bit** Write of “1” in this bit clears the bit at the corresponding position in the FLAGS register. 0_B No action / This value will be always read back from the Bit. 1_B Clears the corresponding flag
0	4, 9:7, 29	r	**Reserved** Read as 0; should be written with 0.

24.4.31 Flags enable register

The FLAGSENABLE register contains the read write bits that enable the corresponding interrupt in case the corresponding event has occurred.

FLAGSENABLE Offset address: 138_H

Flags enable register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TFLE	TFOE	0	RFLE	RFUE	RFOE	CEE	LCE	LAE	LPE	BDE	RTE	HTE	FEE	TCE	PEE
rw	rw	r	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				OMTE	TFEE	0			REDE	FEDE	0	RRE	RHE	TRE	THE
r				rw	rw	r			rw	rw	r	rw	rw	rw	rw

Field	Bits	Type	Description
THE	0	rw	**Flag Enable Bit** This bit enables the interrupt for the flag at the corresponding position in the FLAGS register. 0_B Disabled 1_B Enabled
TRE	1	rw	**Flag Enable Bit** This bit enables the interrupt for the flag at the corresponding position in the FLAGS register. 0_B Disabled 1_B Enabled
RHE	2	rw	**Flag Enable Bit** This bit enables the interrupt for the flag at the corresponding position in the FLAGS register. 0_B Disabled 1_B Enabled
RRE	3	rw	**Flag Enable Bit** This bit enables the interrupt for the flag at the corresponding position in the FLAGS register. 0_B Disabled 1_B Enabled
FEDE	5	rw	**Flag Enable Bit** This bit enables the interrupt for the flag at the corresponding position in the FLAGS register. 0_B Disabled 1_B Enabled
REDE	6	rw	**Flag Enable Bit** This bit enables the interrupt for the flag at the corresponding position in the FLAGS register. 0_B Disabled 1_B Enabled
TFEE	10	rw	**Flag Enable Bit** This bit enables the interrupt for the flag at the corresponding position in the FLAGS register. 0_B Disabled 1_B Enabled
OMTE	11	rw	**Flag Enable Bit** This bit enables the interrupt for the flag at the corresponding position in the FLAGS register. 0_B Disabled 1_B Enabled

(table continues...)

(continued)

Field	Bits	Type	Description
PEE	16	rw	**Flag Enable Bit** This bit enables the interrupt for the flag at the corresponding position in the FLAGS register. 0_B Disabled 1_B Enabled
TCE	17	rw	**Flag Enable Bit** This bit enables the interrupt for the flag at the corresponding position in the FLAGS register. 0_B Disabled 1_B Enabled
FEE	18	rw	**Flag Enable Bit** This bit enables the interrupt for the flag at the corresponding position in the FLAGS register. 0_B Disabled 1_B Enabled
HTE	19	rw	**Flag Enable Bit** This bit enables the interrupt for the flag at the corresponding position in the FLAGS register. 0_B Disabled 1_B Enabled
RTE	20	rw	**Flag Enable Bit** This bit enables the interrupt for the flag at the corresponding position in the FLAGS register. 0_B Disabled 1_B Enabled
BDE	21	rw	**Flag Enable Bit** This bit enables the interrupt for the flag at the corresponding position in the FLAGS register. 0_B Disabled 1_B Enabled
LPE	22	rw	**Flag Enable Bit** This bit enables the interrupt for the flag at the corresponding position in the FLAGS register. 0_B Disabled 1_B Enabled
LAE	23	rw	**Flag Enable Bit** This bit enables the interrupt for the flag at the corresponding position in the FLAGS register. 0_B Disabled 1_B Enabled

(table continues...)

(continued)

Field	Bits	Type	Description
LCE	24	rw	**Flag Enable Bit** This bit enables the interrupt for the flag at the corresponding position in the FLAGS register. 0_B Disabled 1_B Enabled
CEE	25	rw	**Flag Enable Bit** This bit enables the interrupt for the flag at the corresponding position in the FLAGS register. 0_B Disabled 1_B Enabled
RFOE	26	rw	**Flag Enable Bit** This bit enables the interrupt for the flag at the corresponding position in the FLAGS register. 0_B Disabled 1_B Enabled
RFUE	27	rw	**Flag Enable Bit** This bit enables the interrupt for the flag at the corresponding position in the FLAGS register. 0_B Disabled 1_B Enabled
RFLE	28	rw	**Flag Enable Bit** This bit enables the interrupt for the flag at the corresponding position in the FLAGS register. 0_B Disabled 1_B Enabled
TFOE	30	rw	**Flag Enable Bit** This bit enables the interrupt for the flag at the corresponding position in the FLAGS register. 0_B Disabled 1_B Enabled
TFLE	31	rw	**Flag Enable Bit** This bit enables the interrupt for the flag at the corresponding position in the FLAGS register. 0_B Disabled 1_B Enabled
0	4, 9:7, 15:12, 29	r	**Reserved** Read as 0; should be written with 0.

24.4.32 Clock selection register

This register is used to select the clock source for the baud rate generation, detection, timeouts and the SPI delays.

The reconfiguration of the clock source has to be done using two writes: first a write of zero to the CLKSEL bit-field, and then a second write defining the new clock source. Between the first and the second write a delay of minimum T_W [3] 4*(1/fA)+2*(1/fCLC) must be inserted by software, where f_A is the frequency being switched off with the first write.

Additionally, always activate the asynchronous clock for at least two f_A cycles:

- after entering the INIT state and after device reset
- after device reset

This is an example of a correct sequence:

1. CSR.CLKSEL = No_Clock (equals the reset state)
2. Wait T_W or poll for CSR.CON = 0
3. FRAMECON.MODE = INIT
4. CSR.CLKSEL = Clock_On
5. Wait T_W or poll for CSR.CON = 1
6. CSR.CLKSEL = No_Clock
7. Wait T_W or poll for CSR.CON = 0
8. FRAMECON.MODE = ASC (or SPI or LIN)
9. CSR.CLKSEL = Clock_On
10. Wait T_W or poll for CSR.CON = 1

CSR Offset address: $13C_H$

Clock selection register Reset values see: Table 1127

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CON								**0**							
rh								r							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0											**CLKSEL**				
r											rw				

Field	Bits	Type	Description
CLKSEL	4:0	rw	**Baud Rate Logic Clock Select** 00_H No clock supplied 02_H $f_{ASCLINF}$ 04_H $f_{ASCLINS}$ (and f_{OSC0} , see CLOCK chapter) **others,** not allowed
CON	31	rh	**Clock On Flag** Shows if the clock in the bit time domain is switched on or off. Many configuration registers can be written only if this bit shows 0 (see header of registers IOCR, BITCON, FRAMECON, BRG, BRD, LINCON, LINBTIMER, LINHTIMER). 0_B Clock is off 1_B Clock is on
0	30:5	r	**Reserved** Read as 0; should be written with 0.

Table 1127 Reset values of CSR

Reset type	Reset value	Note
Kernel Reset	0000 0000$_H$	
After Boot-FW Value	8000 0002$_H$	After ASC BSL execution

24.4.33 Transmit data register

Writing data to this register enters the data to the TxFIFO.

TXDATA Offset address: 140$_H$

Transmit data register Kernel Reset value: 0000 0000$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DATA															
w															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA															
w															

Field	Bits	Type	Description
DATA	31:0	w	**Data** Writing to this bit field writes the content to the TxFIFO, depending on the write width - 8, 16 or 32 bit (configured in TXFIFOCON.INW). A read access to this register returns 0x0.

Table 1128 Mirrors of TXDATA

Short name	Long name	Offset address
TXDATA# (#=0-7)	Transmit data register mirror #	140$_H$+#*4

24.4.34 Receive data register

Reading data from this register takes data from the RXFIFO.

RXDATA Offset address: 160$_H$

Receive data register Kernel Reset value: 0000 0000$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DATA															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA															
rh															

Field	Bits	Type	Description
DATA	31:0	rh	**Data** Reading from this bit field takes content from the RXFIFO, depending on the read width - 8, 16 or 32 bit (configured in RXFIFOCON.OUTW). A write access to this register has no effect.

Table 1129 Mirrors of RXDATA

Short name	Long name	Offset address
RXDATA# (#=0-7)	Receive data register mirror #	160_H+#*4

24.4.35 Receive data debug register

Reading data from this register takes data from the RXFIFO, but does not influence the read pointer. This virtual register provides non-destructive read to the RXFIFO.

RXDATAD Offset address: 180_H

Receive data debug register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DATA															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA															
rh															

Field	Bits	Type	Description
DATA	31:0	rh	**Data** Reading from this bit field takes content from the RXFIFO, depending on the read width - 8, 16 or 32 bit (see RXFIFOCON.OUTW), but does not influence the read pointer of the RXFIFO. A write access to this register has no effect.

24.5 Debug information

OCDS Suspend

OCDS soft suspend request suspends the module activity at the end of the current transaction. In ASC and SPI cases, this is the end of the current frame. In LIN case, this is the end of the current LIN frame (regular end of response, header timeout, or response timeout).

OCDS hard suspend request, for debugging purposes, immediately freezes the ASCLIN module in the current register state. The SPB clock feeding the kernel clock f_{CLC} is immediately switched off and the asynchronous clock f_A remains switched on. Reading and writing of registers is possible but will enable the kernel clock f_{CLC} for a few cycles.

Attention**:* ***Register accesses with clocking in Hard Suspend Mode can have unintended side effects like signals becoming and staying active. This can affect also other modules, so an ASCLIN kernel reset might not be sufficient to bring the system into a defined state.

***Note**:* *In SPI half duplex or LIN mode, when a transmission is triggered, and if soft suspend is requested before the transmission has started (first bit sent out), the CE flag might be set (if collision detection is enabled by setting bit-field FRAMECON.CEN). This flag should be ignored in this particular use-case.*

24.6 References

1. LIN Consortium, LIN Specification Package Revision 1.3, 2.0, 2.1, 2.2

2. Society of Automotive Engineer, LIN Network for Vehicle Applications J2602, 2012

24.7 ASCLIN revision history

Reference	Description of change(s)
Date range: 2024-08-17 to 2024-09-18	
Debug information	Added another note concerning soft suspend behaviour

24.8 TC4Dx ASCLIN information

24.8.1 TC4Dx ASCLIN configuration

There are no deviations from the generic specification.

24.8.2 TC4Dx ASCLIN features

There are no deviations from the generic specification.

24.8.3 TC4Dx ASCLIN functional description

- In TC4Dx device, the ASCLIN module is connected to ComPB bus interface.
- In TC4Dx device hardware-triggered frame transmission is limited just to the eGTM module as GTM module is not present.

Related information

Transmit trigger on page 4647

24.8.4 TC4Dx ASCLIN registers

There are no deviations from the generic specification.

24.8.4.1 Register address space - ASCLIN

Table 1130 Registers address space - ASCLIN

Module	Base address	End address	Note
ASCLIN0	$F46C0000_H$	$F46C01FF_H$	SPB slave interface
ASCLIN1	$F46C0200_H$	$F46C03FF_H$	SPB slave interface
ASCLIN2	$F46C0400_H$	$F46C05FF_H$	SPB slave interface
ASCLIN3	$F46C0600_H$	$F46C07FF_H$	SPB slave interface
ASCLIN4	$F46C0800_H$	$F46C09FF_H$	SPB slave interface
ASCLIN5	$F46C0A00_H$	$F46C0BFF_H$	SPB slave interface
ASCLIN6	$F46C0C00_H$	$F46C0DFF_H$	SPB slave interface
ASCLIN7	$F46C0E00_H$	$F46C0FFF_H$	SPB slave interface
ASCLIN8	$F46C1000_H$	$F46C11FF_H$	SPB slave interface
ASCLIN9	$F46C1200_H$	$F46C13FF_H$	SPB slave interface
ASCLIN10	$F46C1400_H$	$F46C15FF_H$	SPB slave interface
ASCLIN11	$F46C1600_H$	$F46C17FF_H$	SPB slave interface
ASCLIN12	$F46C1800_H$	$F46C19FF_H$	SPB slave interface
ASCLIN13	$F46C1A00_H$	$F46C1BFF_H$	SPB slave interface
ASCLIN14	$F46C1C00_H$	$F46C1DFF_H$	SPB slave interface
ASCLIN15	$F46C1E00_H$	$F46C1FFF_H$	SPB slave interface
ASCLIN16	$F46C2000_H$	$F46C21FF_H$	SPB slave interface
ASCLIN17	$F46C2200_H$	$F46C23FF_H$	SPB slave interface
ASCLIN18	$F46C2400_H$	$F46C25FF_H$	SPB slave interface
ASCLIN19	$F46C2600_H$	$F46C27FF_H$	SPB slave interface
ASCLIN20	$F46C2800_H$	$F46C29FF_H$	SPB slave interface
ASCLIN21	$F46C2A00_H$	$F46C2BFF_H$	SPB slave interface
ASCLIN22	$F46C2C00_H$	$F46C2DFF_H$	SPB slave interface
ASCLIN23	$F46C2E00_H$	$F46C2FFF_H$	SPB slave interface
ASCLIN24	$F46C3000_H$	$F46C31FF_H$	SPB slave interface
ASCLIN25	$F46C3200_H$	$F46C33FF_H$	SPB slave interface
ASCLIN26	$F46C3400_H$	$F46C35FF_H$	SPB slave interface
ASCLIN27	$F46C3600_H$	$F46C37FF_H$	SPB slave interface

24.8.4.2 Register overview - access mode glossary

Table 1131 Register overview - access mode glossary

Keyword	Description
E	Access protection using PROT register ASCLIN0_PROTE or ASCLIN1_PROTE or ASCLIN2_PROTE or ASCLIN3_PROTE or ASCLIN4_PROTE or ASCLIN5_PROTE or ASCLIN6_PROTE or ASCLIN7_PROTE or ASCLIN8_PROTE or ASCLIN9_PROTE or ASCLIN10_PROTE or ASCLIN11_PROTE or ASCLIN12_PROTE or ASCLIN13_PROTE or ASCLIN14_PROTE or ASCLIN15_PROTE or ASCLIN16_PROTE or ASCLIN17_PROTE or ASCLIN18_PROTE or ASCLIN19_PROTE or ASCLIN20_PROTE or ASCLIN21_PROTE or ASCLIN22_PROTE or ASCLIN23_PROTE or ASCLIN24_PROTE or ASCLIN25_PROTE or ASCLIN26_PROTE or ASCLIN27_PROTE .
SE	Access protection using PROT register ASCLIN0_PROTSE or ASCLIN1_PROTSE or ASCLIN2_PROTSE or ASCLIN3_PROTSE or ASCLIN4_PROTSE or ASCLIN5_PROTSE or ASCLIN6_PROTSE or ASCLIN7_PROTSE or ASCLIN8_PROTSE or ASCLIN9_PROTSE or ASCLIN10_PROTSE or ASCLIN11_PROTSE or ASCLIN12_PROTSE or ASCLIN13_PROTSE or ASCLIN14_PROTSE or ASCLIN15_PROTSE or ASCLIN16_PROTSE or ASCLIN17_PROTSE or ASCLIN18_PROTSE or ASCLIN19_PROTSE or ASCLIN20_PROTSE or ASCLIN21_PROTSE or ASCLIN22_PROTSE or ASCLIN23_PROTSE or ASCLIN24_PROTSE or ASCLIN25_PROTSE or ASCLIN26_PROTSE or ASCLIN27_PROTSE .

(table continues...)

Table 1131 (continued) Register overview - access mode glossary

Keyword	Description
APU-P	Protection group consisting of registers ASCLIN0_ACCEN_WRA , ASCLIN0_ACCEN_WRB , ASCLIN0_ACCEN_RDA , ASCLIN0_ACCEN_RDB , ASCLIN0_ACCEN_VM , ASCLIN0_ACCEN_PRS or ASCLIN1_ACCEN_WRA , ASCLIN1_ACCEN_WRB , ASCLIN1_ACCEN_RDA , ASCLIN1_ACCEN_RDB , ASCLIN1_ACCEN_VM , ASCLIN1_ACCEN_PRS or ASCLIN2_ACCEN_WRA , ASCLIN2_ACCEN_WRB , ASCLIN2_ACCEN_RDA , ASCLIN2_ACCEN_RDB , ASCLIN2_ACCEN_VM , ASCLIN2_ACCEN_PRS or ASCLIN3_ACCEN_WRA , ASCLIN3_ACCEN_WRB , ASCLIN3_ACCEN_RDA , ASCLIN3_ACCEN_RDB , ASCLIN3_ACCEN_VM , ASCLIN3_ACCEN_PRS or ASCLIN4_ACCEN_WRA , ASCLIN4_ACCEN_WRB , ASCLIN4_ACCEN_RDA , ASCLIN4_ACCEN_RDB , ASCLIN4_ACCEN_VM , ASCLIN4_ACCEN_PRS or ASCLIN5_ACCEN_WRA , ASCLIN5_ACCEN_WRB , ASCLIN5_ACCEN_RDA , ASCLIN5_ACCEN_RDB , ASCLIN5_ACCEN_VM , ASCLIN5_ACCEN_PRS or ASCLIN6_ACCEN_WRA , ASCLIN6_ACCEN_WRB , ASCLIN6_ACCEN_RDA , ASCLIN6_ACCEN_RDB , ASCLIN6_ACCEN_VM , ASCLIN6_ACCEN_PRS or ASCLIN7_ACCEN_WRA , ASCLIN7_ACCEN_WRB , ASCLIN7_ACCEN_RDA , ASCLIN7_ACCEN_RDB , ASCLIN7_ACCEN_VM , ASCLIN7_ACCEN_PRS or ASCLIN8_ACCEN_WRA , ASCLIN8_ACCEN_WRB , ASCLIN8_ACCEN_RDA , ASCLIN8_ACCEN_RDB , ASCLIN8_ACCEN_VM , ASCLIN8_ACCEN_PRS or ASCLIN9_ACCEN_WRA , ASCLIN9_ACCEN_WRB , ASCLIN9_ACCEN_RDA , ASCLIN9_ACCEN_RDB , ASCLIN9_ACCEN_VM , ASCLIN9_ACCEN_PRS or ASCLIN10_ACCEN_WRA , ASCLIN10_ACCEN_WRB , ASCLIN10_ACCEN_RDA , ASCLIN10_ACCEN_RDB , ASCLIN10_ACCEN_VM , ASCLIN10_ACCEN_PRS or ASCLIN11_ACCEN_WRA , ASCLIN11_ACCEN_WRB , ASCLIN11_ACCEN_RDA , ASCLIN11_ACCEN_RDB , ASCLIN11_ACCEN_VM , ASCLIN11_ACCEN_PRS or ASCLIN12_ACCEN_WRA , ASCLIN12_ACCEN_WRB , ASCLIN12_ACCEN_RDA , ASCLIN12_ACCEN_RDB , ASCLIN12_ACCEN_VM , ASCLIN12_ACCEN_PRS or ASCLIN13_ACCEN_WRA , ASCLIN13_ACCEN_WRB , ASCLIN13_ACCEN_RDA , ASCLIN13_ACCEN_RDB , ASCLIN13_ACCEN_VM , ASCLIN13_ACCEN_PRS or ASCLIN14_ACCEN_WRA , ASCLIN14_ACCEN_WRB , ASCLIN14_ACCEN_RDA , ASCLIN14_ACCEN_RDB , ASCLIN14_ACCEN_VM , ASCLIN14_ACCEN_PRS or ASCLIN15_ACCEN_WRA , ASCLIN15_ACCEN_WRB , ASCLIN15_ACCEN_RDA , ASCLIN15_ACCEN_RDB , ASCLIN15_ACCEN_VM , ASCLIN15_ACCEN_PRS or ASCLIN16_ACCEN_WRA , ASCLIN16_ACCEN_WRB , ASCLIN16_ACCEN_RDA , ASCLIN16_ACCEN_RDB , ASCLIN16_ACCEN_VM , ASCLIN16_ACCEN_PRS or ASCLIN17_ACCEN_WRA , ASCLIN17_ACCEN_WRB , ASCLIN17_ACCEN_RDA , ASCLIN17_ACCEN_RDB , ASCLIN17_ACCEN_VM , ASCLIN17_ACCEN_PRS or ASCLIN18_ACCEN_WRA , ASCLIN18_ACCEN_WRB , ASCLIN18_ACCEN_RDA , ASCLIN18_ACCEN_RDB , ASCLIN18_ACCEN_VM , ASCLIN18_ACCEN_PRS or ASCLIN19_ACCEN_WRA , ASCLIN19_ACCEN_WRB , ASCLIN19_ACCEN_RDA , ASCLIN19_ACCEN_RDB , ASCLIN19_ACCEN_VM , ASCLIN19_ACCEN_PRS or ASCLIN20_ACCEN_WRA , ASCLIN20_ACCEN_WRB , ASCLIN20_ACCEN_RDA , ASCLIN20_ACCEN_RDB , ASCLIN20_ACCEN_VM , ASCLIN20_ACCEN_PRS

(table continues...)

Table 1131 (continued) Register overview - access mode glossary

Keyword	Description
	or ASCLIN21_ACCEN_WRA , ASCLIN21_ACCEN_WRB , ASCLIN21_ACCEN_RDA , ASCLIN21_ACCEN_RDB , ASCLIN21_ACCEN_VM , ASCLIN21_ACCEN_PRS or ASCLIN22_ACCEN_WRA , ASCLIN22_ACCEN_WRB , ASCLIN22_ACCEN_RDA , ASCLIN22_ACCEN_RDB , ASCLIN22_ACCEN_VM , ASCLIN22_ACCEN_PRS or ASCLIN23_ACCEN_WRA , ASCLIN23_ACCEN_WRB , ASCLIN23_ACCEN_RDA , ASCLIN23_ACCEN_RDB , ASCLIN23_ACCEN_VM , ASCLIN23_ACCEN_PRS or ASCLIN24_ACCEN_WRA , ASCLIN24_ACCEN_WRB , ASCLIN24_ACCEN_RDA , ASCLIN24_ACCEN_RDB , ASCLIN24_ACCEN_VM , ASCLIN24_ACCEN_PRS or ASCLIN25_ACCEN_WRA , ASCLIN25_ACCEN_WRB , ASCLIN25_ACCEN_RDA , ASCLIN25_ACCEN_RDB , ASCLIN25_ACCEN_VM , ASCLIN25_ACCEN_PRS or ASCLIN26_ACCEN_WRA , ASCLIN26_ACCEN_WRB , ASCLIN26_ACCEN_RDA , ASCLIN26_ACCEN_RDB , ASCLIN26_ACCEN_VM , ASCLIN26_ACCEN_PRS or ASCLIN27_ACCEN_WRA , ASCLIN27_ACCEN_WRB , ASCLIN27_ACCEN_RDA , ASCLIN27_ACCEN_RDB , ASCLIN27_ACCEN_VM , ASCLIN27_ACCEN_PRS .
P	Access protection using *APU-P* registers.
BE	Always returns a Bus Error.
U	No access restrictions.
PROT	Access restrictions as defined in the PROT register access rules.

24.8.4.3 Register overview - ASCLIN0 (ascending offset address)

Table 1132 Register overview - ASCLIN0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ASCLIN0_CLC	Clock control register	000_H	P	P, SV, E	4670	4670
ASCLIN0_OCS	OCDS control and status register	004_H	P	SV, P	Debug Reset	4671
ASCLIN0_ID	Module identification register	008_H	P	BE	PowerOn Reset	4672
ASCLIN0_RST_CTRLA	Reset control register A	$00C_H$	P	P, SV, E	Application Reset	4673
ASCLIN0_RST_CTRLB	Reset control register B	010_H	P	P, SV, E	Application Reset	4673
ASCLIN0_RST_STAT	Reset status register	014_H	P	BE	Application Reset	4674
ASCLIN0_PROTE	PROT register endinit	020_H	U	SV, PROT	Application Reset	4675
ASCLIN0_PROTSE	PROT register safe endinit	024_H	U	SV, PROT	Application Reset	4676
ASCLIN0_ACCEN_WRA	Write access enable register A	040_H	U	SE, SV	Application Reset	4678

(table continues...)

Table 1132 (continued) Register overview - ASCLIN0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ASCLIN0_ACCEN_WRB	Write access enable register B	044_H	U	SE, SV	Application Reset	4678
ASCLIN0_ACCEN_RDA	Read access enable register A	048_H	U	SE, SV	Application Reset	4679
ASCLIN0_ACCEN_RDB	Read access enable register B	$04C_H$	U	SE, SV	Application Reset	4679
ASCLIN0_ACCEN_VM	VM access enable register	050_H	U	SE, SV	Application Reset	4680
ASCLIN0_ACCEN_PRS	PRS access enable register	054_H	U	SE, SV	Application Reset	4680
ASCLIN0_IOCR	Input and output control register	100_H	P	P	4681	4681
ASCLIN0_TXFIFOCON	TxFIFO configuration register	104_H	P	P	4683	4683
ASCLIN0_RXFIFOCON	RxFIFO configuration register	108_H	P	P	4685	4685
ASCLIN0_BITCON	Bit configuration register	$10C_H$	P	P	4687	4687
ASCLIN0_FRAMECON	Frame control register	110_H	P	P	4689	4689
ASCLIN0_DATCON	Data configuration register	114_H	P	P	Kernel Reset	4691
ASCLIN0_BRG	Baud rate generation register	118_H	P	P	4692	4692
ASCLIN0_BRD	Baud rate detection register	$11C_H$	P	P	Kernel Reset	4693
ASCLIN0_LINCON	LIN control register	120_H	P	P	Kernel Reset	4694
ASCLIN0_LINBTIMER	LIN break timer register	124_H	P	P	Kernel Reset	4695
ASCLIN0_LINHTIMER	LIN header timer register	128_H	P	P	Kernel Reset	4696
ASCLIN0_FLAGS	Flags register	$12C_H$	P	BE	Kernel Reset	4696
ASCLIN0_FLAGSSET	Flags set register	130_H	P	P	Kernel Reset	4701
ASCLIN0_FLAGSCLEAR	Flags clear register	134_H	P	P	Kernel Reset	4704
ASCLIN0_FLAGSENABLE	Flags enable register	138_H	P	P	Kernel Reset	4708
ASCLIN0_CSR	Clock selection register	$13C_H$	P	P	4712	4712
ASCLIN0_TXDATA	Transmit data register	140_H	P	P	Kernel Reset	4713
ASCLIN0_TXDATA # (#=0-7)	Transmit data register mirror #	140_H+#*4	P	P	Kernel Reset	4713

(table continues...)

Table 1132 (continued) Register overview - ASCLIN0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ASCLIN0_RXDATA	Receive data register	160_H	P	BE	Kernel Reset	4713
ASCLIN0_RXDATA# (#=0-7)	Receive data register mirror #	160_H+#*4	P	BE	Kernel Reset	4713
ASCLIN0_RXDATAD	Receive data debug register	180_H	P	BE	Kernel Reset	4714

24.8.4.4 Register overview - ASCLIN1 (ascending offset address)

Table 1133 Register overview - ASCLIN1 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ASCLIN1_CLC	Clock control register	000_H	P	P, SV, E	4670	4670
ASCLIN1_OCS	OCDS control and status register	004_H	P	SV, P	Debug Reset	4671
ASCLIN1_ID	Module identification register	008_H	P	BE	PowerOn Reset	4672
ASCLIN1_RST_CTRLA	Reset control register A	$00C_H$	P	P, SV, E	Application Reset	4673
ASCLIN1_RST_CTRLB	Reset control register B	010_H	P	P, SV, E	Application Reset	4673
ASCLIN1_RST_STAT	Reset status register	014_H	P	BE	Application Reset	4674
ASCLIN1_PROTE	PROT register endinit	020_H	U	SV, PROT	Application Reset	4675
ASCLIN1_PROTSE	PROT register safe endinit	024_H	U	SV, PROT	Application Reset	4676
ASCLIN1_ACCEN_WRA	Write access enable register A	040_H	U	SE, SV	Application Reset	4678
ASCLIN1_ACCEN_WRB	Write access enable register B	044_H	U	SE, SV	Application Reset	4678
ASCLIN1_ACCEN_RDA	Read access enable register A	048_H	U	SE, SV	Application Reset	4679
ASCLIN1_ACCEN_RDB	Read access enable register B	$04C_H$	U	SE, SV	Application Reset	4679
ASCLIN1_ACCEN_VM	VM access enable register	050_H	U	SE, SV	Application Reset	4680
ASCLIN1_ACCEN_PRS	PRS access enable register	054_H	U	SE, SV	Application Reset	4680
ASCLIN1_IOCR	Input and output control register	100_H	P	P	4681	4681

(table continues...)

Table 1133 (continued) Register overview - ASCLIN1 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ASCLIN1_TXFIFOCON	TxFIFO configuration register	104_H	P	P	4683	4683
ASCLIN1_RXFIFOCON	RxFIFO configuration register	108_H	P	P	4685	4685
ASCLIN1_BITCON	Bit configuration register	$10C_H$	P	P	4687	4687
ASCLIN1_FRAMECON	Frame control register	110_H	P	P	4689	4689
ASCLIN1_DATCON	Data configuration register	114_H	P	P	Kernel Reset	4691
ASCLIN1_BRG	Baud rate generation register	118_H	P	P	4692	4692
ASCLIN1_BRD	Baud rate detection register	$11C_H$	P	P	Kernel Reset	4693
ASCLIN1_LINCON	LIN control register	120_H	P	P	Kernel Reset	4694
ASCLIN1_LINBTIMER	LIN break timer register	124_H	P	P	Kernel Reset	4695
ASCLIN1_LINHTIMER	LIN header timer register	128_H	P	P	Kernel Reset	4696
ASCLIN1_FLAGS	Flags register	$12C_H$	P	BE	Kernel Reset	4696
ASCLIN1_FLAGSSET	Flags set register	130_H	P	P	Kernel Reset	4701
ASCLIN1_FLAGSCLEAR	Flags clear register	134_H	P	P	Kernel Reset	4704
ASCLIN1_FLAGSENABLE	Flags enable register	138_H	P	P	Kernel Reset	4708
ASCLIN1_CSR	Clock selection register	$13C_H$	P	P	4712	4712
ASCLIN1_TXDATA	Transmit data register	140_H	P	P	Kernel Reset	4713
ASCLIN1_TXDATA# (#=0-7)	Transmit data register mirror #	140_H+#*4	P	P	Kernel Reset	4713
ASCLIN1_RXDATA	Receive data register	160_H	P	BE	Kernel Reset	4713
ASCLIN1_RXDATA# (#=0-7)	Receive data register mirror #	160_H+#*4	P	BE	Kernel Reset	4713
ASCLIN1_RXDATAD	Receive data debug register	180_H	P	BE	Kernel Reset	4714

24.8.4.5 Register overview - ASCLIN2 (ascending offset address)

Table 1134 Register overview - ASCLIN2 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ASCLIN2_CLC	Clock control register	000_H	P	P, SV, E	4670	4670
ASCLIN2_OCS	OCDS control and status register	004_H	P	SV, P	Debug Reset	4671
ASCLIN2_ID	Module identification register	008_H	P	BE	PowerOn Reset	4672
ASCLIN2_RST_CTRLA	Reset control register A	$00C_H$	P	P, SV, E	Application Reset	4673
ASCLIN2_RST_CTRLB	Reset control register B	010_H	P	P, SV, E	Application Reset	4673
ASCLIN2_RST_STAT	Reset status register	014_H	P	BE	Application Reset	4674
ASCLIN2_PROTE	PROT register endinit	020_H	U	SV, PROT	Application Reset	4675
ASCLIN2_PROTSE	PROT register safe endinit	024_H	U	SV, PROT	Application Reset	4676
ASCLIN2_ACCEN_WRA	Write access enable register A	040_H	U	SE, SV	Application Reset	4678
ASCLIN2_ACCEN_WRB	Write access enable register B	044_H	U	SE, SV	Application Reset	4678
ASCLIN2_ACCEN_RDA	Read access enable register A	048_H	U	SE, SV	Application Reset	4679
ASCLIN2_ACCEN_RDB	Read access enable register B	$04C_H$	U	SE, SV	Application Reset	4679
ASCLIN2_ACCEN_VM	VM access enable register	050_H	U	SE, SV	Application Reset	4680
ASCLIN2_ACCEN_PRS	PRS access enable register	054_H	U	SE, SV	Application Reset	4680
ASCLIN2_IOCR	Input and output control register	100_H	P	P	4681	4681
ASCLIN2_TXFIFOCON	TxFIFO configuration register	104_H	P	P	4683	4683
ASCLIN2_RXFIFOCON	RxFIFO configuration register	108_H	P	P	4685	4685
ASCLIN2_BITCON	Bit configuration register	$10C_H$	P	P	4687	4687
ASCLIN2_FRAMECON	Frame control register	110_H	P	P	4689	4689
ASCLIN2_DATCON	Data configuration register	114_H	P	P	Kernel Reset	4691
ASCLIN2_BRG	Baud rate generation register	118_H	P	P	4692	4692

(table continues...)

Table 1134 (continued) Register overview - ASCLIN2 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ASCLIN2_BRD	Baud rate detection register	$11C_H$	P	P	Kernel Reset	4693
ASCLIN2_LINCON	LIN control register	120_H	P	P	Kernel Reset	4694
ASCLIN2_LINBTIMER	LIN break timer register	124_H	P	P	Kernel Reset	4695
ASCLIN2_LINHTIMER	LIN header timer register	128_H	P	P	Kernel Reset	4696
ASCLIN2_FLAGS	Flags register	$12C_H$	P	BE	Kernel Reset	4696
ASCLIN2_FLAGSSET	Flags set register	130_H	P	P	Kernel Reset	4701
ASCLIN2_FLAGSCLEAR	Flags clear register	134_H	P	P	Kernel Reset	4704
ASCLIN2_FLAGSENABLE	Flags enable register	138_H	P	P	Kernel Reset	4708
ASCLIN2_CSR	Clock selection register	$13C_H$	P	P	4712	4712
ASCLIN2_TXDATA	Transmit data register	140_H	P	P	Kernel Reset	4713
ASCLIN2_TXDATA# (#=0-7)	Transmit data register mirror #	140_H+#*4	P	P	Kernel Reset	4713
ASCLIN2_RXDATA	Receive data register	160_H	P	BE	Kernel Reset	4713
ASCLIN2_RXDATA# (#=0-7)	Receive data register mirror #	160_H+#*4	P	BE	Kernel Reset	4713
ASCLIN2_RXDATAD	Receive data debug register	180_H	P	BE	Kernel Reset	4714

24.8.4.6 Register overview - ASCLIN3 (ascending offset address)

Table 1135 Register overview - ASCLIN3 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ASCLIN3_CLC	Clock control register	000_H	P	P, SV, E	4670	4670
ASCLIN3_OCS	OCDS control and status register	004_H	P	SV, P	Debug Reset	4671
ASCLIN3_ID	Module identification register	008_H	P	BE	PowerOn Reset	4672
ASCLIN3_RST_CTRLA	Reset control register A	$00C_H$	P	P, SV, E	Application Reset	4673
ASCLIN3_RST_CTRLB	Reset control register B	010_H	P	P, SV, E	Application Reset	4673
ASCLIN3_RST_STAT	Reset status register	014_H	P	BE	Application Reset	4674

(table continues...)

Table 1135 (continued) Register overview - ASCLIN3 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ASCLIN3_PROTE	PROT register endinit	020_H	U	SV, PROT	Application Reset	4675
ASCLIN3_PROTSE	PROT register safe endinit	024_H	U	SV, PROT	Application Reset	4676
ASCLIN3_ACCEN_WRA	Write access enable register A	040_H	U	SE, SV	Application Reset	4678
ASCLIN3_ACCEN_WRB	Write access enable register B	044_H	U	SE, SV	Application Reset	4678
ASCLIN3_ACCEN_RDA	Read access enable register A	048_H	U	SE, SV	Application Reset	4679
ASCLIN3_ACCEN_RDB	Read access enable register B	$04C_H$	U	SE, SV	Application Reset	4679
ASCLIN3_ACCEN_VM	VM access enable register	050_H	U	SE, SV	Application Reset	4680
ASCLIN3_ACCEN_PRS	PRS access enable register	054_H	U	SE, SV	Application Reset	4680
ASCLIN3_IOCR	Input and output control register	100_H	P	P	4681	4681
ASCLIN3_TXFIFOCON	TxFIFO configuration register	104_H	P	P	4683	4683
ASCLIN3_RXFIFOCON	RxFIFO configuration register	108_H	P	P	4685	4685
ASCLIN3_BITCON	Bit configuration register	$10C_H$	P	P	4687	4687
ASCLIN3_FRAMECON	Frame control register	110_H	P	P	4689	4689
ASCLIN3_DATCON	Data configuration register	114_H	P	P	Kernel Reset	4691
ASCLIN3_BRG	Baud rate generation register	118_H	P	P	4692	4692
ASCLIN3_BRD	Baud rate detection register	$11C_H$	P	P	Kernel Reset	4693
ASCLIN3_LINCON	LIN control register	120_H	P	P	Kernel Reset	4694
ASCLIN3_LINBTIMER	LIN break timer register	124_H	P	P	Kernel Reset	4695
ASCLIN3_LINHTIMER	LIN header timer register	128_H	P	P	Kernel Reset	4696
ASCLIN3_FLAGS	Flags register	$12C_H$	P	BE	Kernel Reset	4696
ASCLIN3_FLAGSSET	Flags set register	130_H	P	P	Kernel Reset	4701
ASCLIN3_FLAGSCLEAR	Flags clear register	134_H	P	P	Kernel Reset	4704

(table continues...)

Table 1135 (continued) Register overview - ASCLIN3 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ASCLIN3_FLAGSENABLE	Flags enable register	138_H	P	P	Kernel Reset	4708
ASCLIN3_CSR	Clock selection register	$13C_H$	P	P	4712	4712
ASCLIN3_TXDATA	Transmit data register	140_H	P	P	Kernel Reset	4713
ASCLIN3_TXDATA # (#=0-7)	Transmit data register mirror #	140_H+#* 4	P	P	Kernel Reset	4713
ASCLIN3_RXDATA	Receive data register	160_H	P	BE	Kernel Reset	4713
ASCLIN3_RXDATA # (#=0-7)	Receive data register mirror #	160_H+#* 4	P	BE	Kernel Reset	4713
ASCLIN3_RXDATAD	Receive data debug register	180_H	P	BE	Kernel Reset	4714

24.8.4.7 Register overview - ASCLIN4 (ascending offset address)

Table 1136 Register overview - ASCLIN4 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ASCLIN4_CLC	Clock control register	000_H	P	P, SV, E	4670	4670
ASCLIN4_OCS	OCDS control and status register	004_H	P	SV, P	Debug Reset	4671
ASCLIN4_ID	Module identification register	008_H	P	BE	PowerOn Reset	4672
ASCLIN4_RST_CTRLA	Reset control register A	$00C_H$	P	P, SV, E	Application Reset	4673
ASCLIN4_RST_CTRLB	Reset control register B	010_H	P	P, SV, E	Application Reset	4673
ASCLIN4_RST_STAT	Reset status register	014_H	P	BE	Application Reset	4674
ASCLIN4_PROTE	PROT register endinit	020_H	U	SV, PROT	Application Reset	4675
ASCLIN4_PROTSE	PROT register safe endinit	024_H	U	SV, PROT	Application Reset	4676
ASCLIN4_ACCEN_WRA	Write access enable register A	040_H	U	SE, SV	Application Reset	4678
ASCLIN4_ACCEN_WRB	Write access enable register B	044_H	U	SE, SV	Application Reset	4678
ASCLIN4_ACCEN_RDA	Read access enable register A	048_H	U	SE, SV	Application Reset	4679
ASCLIN4_ACCEN_RDB	Read access enable register B	$04C_H$	U	SE, SV	Application Reset	4679

(table continues...)

Table 1136 (continued) Register overview - ASCLIN4 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ASCLIN4_ACCEN_VM	VM access enable register	050_H	U	SE, SV	Application Reset	4680
ASCLIN4_ACCEN_PRS	PRS access enable register	054_H	U	SE, SV	Application Reset	4680
ASCLIN4_IOCR	Input and output control register	100_H	P	P	4681	4681
ASCLIN4_TXFIFOCON	TxFIFO configuration register	104_H	P	P	4683	4683
ASCLIN4_RXFIFOCON	RxFIFO configuration register	108_H	P	P	4685	4685
ASCLIN4_BITCON	Bit configuration register	$10C_H$	P	P	4687	4687
ASCLIN4_FRAMECON	Frame control register	110_H	P	P	4689	4689
ASCLIN4_DATCON	Data configuration register	114_H	P	P	Kernel Reset	4691
ASCLIN4_BRG	Baud rate generation register	118_H	P	P	4692	4692
ASCLIN4_BRD	Baud rate detection register	$11C_H$	P	P	Kernel Reset	4693
ASCLIN4_LINCON	LIN control register	120_H	P	P	Kernel Reset	4694
ASCLIN4_LINBTIMER	LIN break timer register	124_H	P	P	Kernel Reset	4695
ASCLIN4_LINHTIMER	LIN header timer register	128_H	P	P	Kernel Reset	4696
ASCLIN4_FLAGS	Flags register	$12C_H$	P	BE	Kernel Reset	4696
ASCLIN4_FLAGSSET	Flags set register	130_H	P	P	Kernel Reset	4701
ASCLIN4_FLAGSCLEAR	Flags clear register	134_H	P	P	Kernel Reset	4704
ASCLIN4_FLAGSENABLE	Flags enable register	138_H	P	P	Kernel Reset	4708
ASCLIN4_CSR	Clock selection register	$13C_H$	P	P	4712	4712
ASCLIN4_TXDATA	Transmit data register	140_H	P	P	Kernel Reset	4713
ASCLIN4_TXDATA# (#=0-7)	Transmit data register mirror #	140_H+#*4	P	P	Kernel Reset	4713
ASCLIN4_RXDATA	Receive data register	160_H	P	BE	Kernel Reset	4713
ASCLIN4_RXDATA# (#=0-7)	Receive data register mirror #	160_H+#*4	P	BE	Kernel Reset	4713
ASCLIN4_RXDATAD	Receive data debug register	180_H	P	BE	Kernel Reset	4714

24.8.4.8 Register overview - ASCLIN5 (ascending offset address)

Table 1137 Register overview - ASCLIN5 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ASCLIN5_CLC	Clock control register	000_H	P	P, SV, E	4670	4670
ASCLIN5_OCS	OCDS control and status register	004_H	P	SV, P	Debug Reset	4671
ASCLIN5_ID	Module identification register	008_H	P	BE	PowerOn Reset	4672
ASCLIN5_RST_CTRLA	Reset control register A	$00C_H$	P	P, SV, E	Application Reset	4673
ASCLIN5_RST_CTRLB	Reset control register B	010_H	P	P, SV, E	Application Reset	4673
ASCLIN5_RST_STAT	Reset status register	014_H	P	BE	Application Reset	4674
ASCLIN5_PROTE	PROT register endinit	020_H	U	SV, PROT	Application Reset	4675
ASCLIN5_PROTSE	PROT register safe endinit	024_H	U	SV, PROT	Application Reset	4676
ASCLIN5_ACCEN_WRA	Write access enable register A	040_H	U	SE, SV	Application Reset	4678
ASCLIN5_ACCEN_WRB	Write access enable register B	044_H	U	SE, SV	Application Reset	4678
ASCLIN5_ACCEN_RDA	Read access enable register A	048_H	U	SE, SV	Application Reset	4679
ASCLIN5_ACCEN_RDB	Read access enable register B	$04C_H$	U	SE, SV	Application Reset	4679
ASCLIN5_ACCEN_VM	VM access enable register	050_H	U	SE, SV	Application Reset	4680
ASCLIN5_ACCEN_PRS	PRS access enable register	054_H	U	SE, SV	Application Reset	4680
ASCLIN5_IOCR	Input and output control register	100_H	P	P	4681	4681
ASCLIN5_TXFIFOCON	TxFIFO configuration register	104_H	P	P	4683	4683
ASCLIN5_RXFIFOCON	RxFIFO configuration register	108_H	P	P	4685	4685
ASCLIN5_BITCON	Bit configuration register	$10C_H$	P	P	4687	4687
ASCLIN5_FRAMECON	Frame control register	110_H	P	P	4689	4689
ASCLIN5_DATCON	Data configuration register	114_H	P	P	Kernel Reset	4691
ASCLIN5_BRG	Baud rate generation register	118_H	P	P	4692	4692

(table continues...)

Table 1137 (continued) Register overview - ASCLIN5 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ASCLIN5_BRD	Baud rate detection register	$11C_H$	P	P	Kernel Reset	4693
ASCLIN5_LINCON	LIN control register	120_H	P	P	Kernel Reset	4694
ASCLIN5_LINBTIMER	LIN break timer register	124_H	P	P	Kernel Reset	4695
ASCLIN5_LINHTIMER	LIN header timer register	128_H	P	P	Kernel Reset	4696
ASCLIN5_FLAGS	Flags register	$12C_H$	P	BE	Kernel Reset	4696
ASCLIN5_FLAGSSET	Flags set register	130_H	P	P	Kernel Reset	4701
ASCLIN5_FLAGSCLEAR	Flags clear register	134_H	P	P	Kernel Reset	4704
ASCLIN5_FLAGSENABLE	Flags enable register	138_H	P	P	Kernel Reset	4708
ASCLIN5_CSR	Clock selection register	$13C_H$	P	P	4712	4712
ASCLIN5_TXDATA	Transmit data register	140_H	P	P	Kernel Reset	4713
ASCLIN5_TXDATA# (#=0-7)	Transmit data register mirror #	140_H+#*4	P	P	Kernel Reset	4713
ASCLIN5_RXDATA	Receive data register	160_H	P	BE	Kernel Reset	4713
ASCLIN5_RXDATA# (#=0-7)	Receive data register mirror #	160_H+#*4	P	BE	Kernel Reset	4713
ASCLIN5_RXDATAD	Receive data debug register	180_H	P	BE	Kernel Reset	4714

24.8.4.9 Register overview - ASCLIN6 (ascending offset address)

Table 1138 Register overview - ASCLIN6 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ASCLIN6_CLC	Clock control register	000_H	P	P, SV, E	4670	4670
ASCLIN6_OCS	OCDS control and status register	004_H	P	SV, P	Debug Reset	4671
ASCLIN6_ID	Module identification register	008_H	P	BE	PowerOn Reset	4672
ASCLIN6_RST_CTRLA	Reset control register A	$00C_H$	P	P, SV, E	Application Reset	4673
ASCLIN6_RST_CTRLB	Reset control register B	010_H	P	P, SV, E	Application Reset	4673
ASCLIN6_RST_STAT	Reset status register	014_H	P	BE	Application Reset	4674

(table continues...)

Table 1138 (continued) Register overview - ASCLIN6 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ASCLIN6_PROTE	PROT register endinit	020_H	U	SV, PROT	Application Reset	4675
ASCLIN6_PROTSE	PROT register safe endinit	024_H	U	SV, PROT	Application Reset	4676
ASCLIN6_ACCEN_WRA	Write access enable register A	040_H	U	SE, SV	Application Reset	4678
ASCLIN6_ACCEN_WRB	Write access enable register B	044_H	U	SE, SV	Application Reset	4678
ASCLIN6_ACCEN_RDA	Read access enable register A	048_H	U	SE, SV	Application Reset	4679
ASCLIN6_ACCEN_RDB	Read access enable register B	$04C_H$	U	SE, SV	Application Reset	4679
ASCLIN6_ACCEN_VM	VM access enable register	050_H	U	SE, SV	Application Reset	4680
ASCLIN6_ACCEN_PRS	PRS access enable register	054_H	U	SE, SV	Application Reset	4680
ASCLIN6_IOCR	Input and output control register	100_H	P	P	4681	4681
ASCLIN6_TXFIFOCON	TxFIFO configuration register	104_H	P	P	4683	4683
ASCLIN6_RXFIFOCON	RxFIFO configuration register	108_H	P	P	4685	4685
ASCLIN6_BITCON	Bit configuration register	$10C_H$	P	P	4687	4687
ASCLIN6_FRAMECON	Frame control register	110_H	P	P	4689	4689
ASCLIN6_DATCON	Data configuration register	114_H	P	P	Kernel Reset	4691
ASCLIN6_BRG	Baud rate generation register	118_H	P	P	4692	4692
ASCLIN6_BRD	Baud rate detection register	$11C_H$	P	P	Kernel Reset	4693
ASCLIN6_LINCON	LIN control register	120_H	P	P	Kernel Reset	4694
ASCLIN6_LINBTIMER	LIN break timer register	124_H	P	P	Kernel Reset	4695
ASCLIN6_LINHTIMER	LIN header timer register	128_H	P	P	Kernel Reset	4696
ASCLIN6_FLAGS	Flags register	$12C_H$	P	BE	Kernel Reset	4696
ASCLIN6_FLAGSSET	Flags set register	130_H	P	P	Kernel Reset	4701
ASCLIN6_FLAGSCLEAR	Flags clear register	134_H	P	P	Kernel Reset	4704

(table continues...)

Table 1138 (continued) Register overview - ASCLIN6 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ASCLIN6_FLAGSENABLE	Flags enable register	138_H	P	P	Kernel Reset	4708
ASCLIN6_CSR	Clock selection register	$13C_H$	P	P	4712	4712
ASCLIN6_TXDATA	Transmit data register	140_H	P	P	Kernel Reset	4713
ASCLIN6_TXDATA # (#=0-7)	Transmit data register mirror #	140_H+#*4	P	P	Kernel Reset	4713
ASCLIN6_RXDATA	Receive data register	160_H	P	BE	Kernel Reset	4713
ASCLIN6_RXDATA # (#=0-7)	Receive data register mirror #	160_H+#*4	P	BE	Kernel Reset	4713
ASCLIN6_RXDATAD	Receive data debug register	180_H	P	BE	Kernel Reset	4714

24.8.4.10 Register overview - ASCLIN7 (ascending offset address)

Table 1139 Register overview - ASCLIN7 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ASCLIN7_CLC	Clock control register	000_H	P	P, SV, E	4670	4670
ASCLIN7_OCS	OCDS control and status register	004_H	P	SV, P	Debug Reset	4671
ASCLIN7_ID	Module identification register	008_H	P	BE	PowerOn Reset	4672
ASCLIN7_RST_CTRLA	Reset control register A	$00C_H$	P	P, SV, E	Application Reset	4673
ASCLIN7_RST_CTRLB	Reset control register B	010_H	P	P, SV, E	Application Reset	4673
ASCLIN7_RST_STAT	Reset status register	014_H	P	BE	Application Reset	4674
ASCLIN7_PROTE	PROT register endinit	020_H	U	SV, PROT	Application Reset	4675
ASCLIN7_PROTSE	PROT register safe endinit	024_H	U	SV, PROT	Application Reset	4676
ASCLIN7_ACCEN_WRA	Write access enable register A	040_H	U	SE, SV	Application Reset	4678
ASCLIN7_ACCEN_WRB	Write access enable register B	044_H	U	SE, SV	Application Reset	4678
ASCLIN7_ACCEN_RDA	Read access enable register A	048_H	U	SE, SV	Application Reset	4679
ASCLIN7_ACCEN_RDB	Read access enable register B	$04C_H$	U	SE, SV	Application Reset	4679

(table continues...)

Table 1139 (continued) Register overview - ASCLIN7 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ASCLIN7_ACCEN_VM	VM access enable register	050_H	U	SE, SV	Application Reset	4680
ASCLIN7_ACCEN_PRS	PRS access enable register	054_H	U	SE, SV	Application Reset	4680
ASCLIN7_IOCR	Input and output control register	100_H	P	P	4681	4681
ASCLIN7_TXFIFOCON	TxFIFO configuration register	104_H	P	P	4683	4683
ASCLIN7_RXFIFOCON	RxFIFO configuration register	108_H	P	P	4685	4685
ASCLIN7_BITCON	Bit configuration register	$10C_H$	P	P	4687	4687
ASCLIN7_FRAMECON	Frame control register	110_H	P	P	4689	4689
ASCLIN7_DATCON	Data configuration register	114_H	P	P	Kernel Reset	4691
ASCLIN7_BRG	Baud rate generation register	118_H	P	P	4692	4692
ASCLIN7_BRD	Baud rate detection register	$11C_H$	P	P	Kernel Reset	4693
ASCLIN7_LINCON	LIN control register	120_H	P	P	Kernel Reset	4694
ASCLIN7_LINBTIMER	LIN break timer register	124_H	P	P	Kernel Reset	4695
ASCLIN7_LINHTIMER	LIN header timer register	128_H	P	P	Kernel Reset	4696
ASCLIN7_FLAGS	Flags register	$12C_H$	P	BE	Kernel Reset	4696
ASCLIN7_FLAGSSET	Flags set register	130_H	P	P	Kernel Reset	4701
ASCLIN7_FLAGSCLEAR	Flags clear register	134_H	P	P	Kernel Reset	4704
ASCLIN7_FLAGSENABLE	Flags enable register	138_H	P	P	Kernel Reset	4708
ASCLIN7_CSR	Clock selection register	$13C_H$	P	P	4712	4712
ASCLIN7_TXDATA	Transmit data register	140_H	P	P	Kernel Reset	4713
ASCLIN7_TXDATA# (#=0-7)	Transmit data register mirror #	140_H+#*4	P	P	Kernel Reset	4713
ASCLIN7_RXDATA	Receive data register	160_H	P	BE	Kernel Reset	4713
ASCLIN7_RXDATA# (#=0-7)	Receive data register mirror #	160_H+#*4	P	BE	Kernel Reset	4713
ASCLIN7_RXDATAD	Receive data debug register	180_H	P	BE	Kernel Reset	4714

24.8.4.11 Register overview - ASCLIN8 (ascending offset address)

Table 1140 Register overview - ASCLIN8 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ASCLIN8_CLC	Clock control register	000_H	P	P, SV, E	4670	4670
ASCLIN8_OCS	OCDS control and status register	004_H	P	SV, P	Debug Reset	4671
ASCLIN8_ID	Module identification register	008_H	P	BE	PowerOn Reset	4672
ASCLIN8_RST_CTRLA	Reset control register A	$00C_H$	P	P, SV, E	Application Reset	4673
ASCLIN8_RST_CTRLB	Reset control register B	010_H	P	P, SV, E	Application Reset	4673
ASCLIN8_RST_STAT	Reset status register	014_H	P	BE	Application Reset	4674
ASCLIN8_PROTE	PROT register endinit	020_H	U	SV, PROT	Application Reset	4675
ASCLIN8_PROTSE	PROT register safe endinit	024_H	U	SV, PROT	Application Reset	4676
ASCLIN8_ACCEN_WRA	Write access enable register A	040_H	U	SE, SV	Application Reset	4678
ASCLIN8_ACCEN_WRB	Write access enable register B	044_H	U	SE, SV	Application Reset	4678
ASCLIN8_ACCEN_RDA	Read access enable register A	048_H	U	SE, SV	Application Reset	4679
ASCLIN8_ACCEN_RDB	Read access enable register B	$04C_H$	U	SE, SV	Application Reset	4679
ASCLIN8_ACCEN_VM	VM access enable register	050_H	U	SE, SV	Application Reset	4680
ASCLIN8_ACCEN_PRS	PRS access enable register	054_H	U	SE, SV	Application Reset	4680
ASCLIN8_IOCR	Input and output control register	100_H	P	P	4681	4681
ASCLIN8_TXFIFOCON	TxFIFO configuration register	104_H	P	P	4683	4683
ASCLIN8_RXFIFOCON	RxFIFO configuration register	108_H	P	P	4685	4685
ASCLIN8_BITCON	Bit configuration register	$10C_H$	P	P	4687	4687
ASCLIN8_FRAMECON	Frame control register	110_H	P	P	4689	4689
ASCLIN8_DATCON	Data configuration register	114_H	P	P	Kernel Reset	4691
ASCLIN8_BRG	Baud rate generation register	118_H	P	P	4692	4692

(table continues...)

Table 1140 (continued) Register overview - ASCLIN8 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ASCLIN8_BRD	Baud rate detection register	$11C_H$	P	P	Kernel Reset	4693
ASCLIN8_LINCON	LIN control register	120_H	P	P	Kernel Reset	4694
ASCLIN8_LINBTIMER	LIN break timer register	124_H	P	P	Kernel Reset	4695
ASCLIN8_LINHTIMER	LIN header timer register	128_H	P	P	Kernel Reset	4696
ASCLIN8_FLAGS	Flags register	$12C_H$	P	BE	Kernel Reset	4696
ASCLIN8_FLAGSSET	Flags set register	130_H	P	P	Kernel Reset	4701
ASCLIN8_FLAGSCLEAR	Flags clear register	134_H	P	P	Kernel Reset	4704
ASCLIN8_FLAGSENABLE	Flags enable register	138_H	P	P	Kernel Reset	4708
ASCLIN8_CSR	Clock selection register	$13C_H$	P	P	4712	4712
ASCLIN8_TXDATA	Transmit data register	140_H	P	P	Kernel Reset	4713
ASCLIN8_TXDATA# (#=0-7)	Transmit data register mirror #	140_H+#*4	P	P	Kernel Reset	4713
ASCLIN8_RXDATA	Receive data register	160_H	P	BE	Kernel Reset	4713
ASCLIN8_RXDATA# (#=0-7)	Receive data register mirror #	160_H+#*4	P	BE	Kernel Reset	4713
ASCLIN8_RXDATAD	Receive data debug register	180_H	P	BE	Kernel Reset	4714

24.8.4.12 Register overview - ASCLIN9 (ascending offset address)

Table 1141 Register overview - ASCLIN9 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ASCLIN9_CLC	Clock control register	000_H	P	P, SV, E	4670	4670
ASCLIN9_OCS	OCDS control and status register	004_H	P	SV, P	Debug Reset	4671
ASCLIN9_ID	Module identification register	008_H	P	BE	PowerOn Reset	4672
ASCLIN9_RST_CTRLA	Reset control register A	$00C_H$	P	P, SV, E	Application Reset	4673
ASCLIN9_RST_CTRLB	Reset control register B	010_H	P	P, SV, E	Application Reset	4673
ASCLIN9_RST_STAT	Reset status register	014_H	P	BE	Application Reset	4674

(table continues...)

Table 1141 (continued) Register overview - ASCLIN9 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ASCLIN9_PROTE	PROT register endinit	020_H	U	SV, PROT	Application Reset	4675
ASCLIN9_PROTSE	PROT register safe endinit	024_H	U	SV, PROT	Application Reset	4676
ASCLIN9_ACCEN_WRA	Write access enable register A	040_H	U	SE, SV	Application Reset	4678
ASCLIN9_ACCEN_WRB	Write access enable register B	044_H	U	SE, SV	Application Reset	4678
ASCLIN9_ACCEN_RDA	Read access enable register A	048_H	U	SE, SV	Application Reset	4679
ASCLIN9_ACCEN_RDB	Read access enable register B	$04C_H$	U	SE, SV	Application Reset	4679
ASCLIN9_ACCEN_VM	VM access enable register	050_H	U	SE, SV	Application Reset	4680
ASCLIN9_ACCEN_PRS	PRS access enable register	054_H	U	SE, SV	Application Reset	4680
ASCLIN9_IOCR	Input and output control register	100_H	P	P	4681	4681
ASCLIN9_TXFIFOCON	TxFIFO configuration register	104_H	P	P	4683	4683
ASCLIN9_RXFIFOCON	RxFIFO configuration register	108_H	P	P	4685	4685
ASCLIN9_BITCON	Bit configuration register	$10C_H$	P	P	4687	4687
ASCLIN9_FRAMECON	Frame control register	110_H	P	P	4689	4689
ASCLIN9_DATCON	Data configuration register	114_H	P	P	Kernel Reset	4691
ASCLIN9_BRG	Baud rate generation register	118_H	P	P	4692	4692
ASCLIN9_BRD	Baud rate detection register	$11C_H$	P	P	Kernel Reset	4693
ASCLIN9_LINCON	LIN control register	120_H	P	P	Kernel Reset	4694
ASCLIN9_LINBTIMER	LIN break timer register	124_H	P	P	Kernel Reset	4695
ASCLIN9_LINHTIMER	LIN header timer register	128_H	P	P	Kernel Reset	4696
ASCLIN9_FLAGS	Flags register	$12C_H$	P	BE	Kernel Reset	4696
ASCLIN9_FLAGSSET	Flags set register	130_H	P	P	Kernel Reset	4701
ASCLIN9_FLAGSCLEAR	Flags clear register	134_H	P	P	Kernel Reset	4704

(table continues...)

Table 1141 (continued) Register overview - ASCLIN9 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ASCLIN9_FLAGSENABLE	Flags enable register	138_H	P	P	Kernel Reset	4708
ASCLIN9_CSR	Clock selection register	$13C_H$	P	P	4712	4712
ASCLIN9_TXDATA	Transmit data register	140_H	P	P	Kernel Reset	4713
ASCLIN9_TXDATA # (#=0-7)	Transmit data register mirror #	140_H+#* 4	P	P	Kernel Reset	4713
ASCLIN9_RXDATA	Receive data register	160_H	P	BE	Kernel Reset	4713
ASCLIN9_RXDATA # (#=0-7)	Receive data register mirror #	160_H+#* 4	P	BE	Kernel Reset	4713
ASCLIN9_RXDATAD	Receive data debug register	180_H	P	BE	Kernel Reset	4714

24.8.4.13 Register overview - ASCLIN10 (ascending offset address)

Table 1142 Register overview - ASCLIN10 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ASCLIN10_CLC	Clock control register	000_H	P	P, SV, E	4670	4670
ASCLIN10_OCS	OCDS control and status register	004_H	P	SV, P	Debug Reset	4671
ASCLIN10_ID	Module identification register	008_H	P	BE	PowerOn Reset	4672
ASCLIN10_RST_CTRLA	Reset control register A	$00C_H$	P	P, SV, E	Application Reset	4673
ASCLIN10_RST_CTRLB	Reset control register B	010_H	P	P, SV, E	Application Reset	4673
ASCLIN10_RST_STAT	Reset status register	014_H	P	BE	Application Reset	4674
ASCLIN10_PROTE	PROT register endinit	020_H	U	SV, PROT	Application Reset	4675
ASCLIN10_PROTSE	PROT register safe endinit	024_H	U	SV, PROT	Application Reset	4676
ASCLIN10_ACCEN_WRA	Write access enable register A	040_H	U	SE, SV	Application Reset	4678
ASCLIN10_ACCEN_WRB	Write access enable register B	044_H	U	SE, SV	Application Reset	4678
ASCLIN10_ACCEN_RDA	Read access enable register A	048_H	U	SE, SV	Application Reset	4679
ASCLIN10_ACCEN_RDB	Read access enable register B	$04C_H$	U	SE, SV	Application Reset	4679

(table continues...)

Table 1142 (continued) Register overview - ASCLIN10 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ASCLIN10_ACCEN_VM	VM access enable register	050_H	U	SE, SV	Application Reset	4680
ASCLIN10_ACCEN_PRS	PRS access enable register	054_H	U	SE, SV	Application Reset	4680
ASCLIN10_IOCR	Input and output control register	100_H	P	P	4681	4681
ASCLIN10_TXFIFOCON	TxFIFO configuration register	104_H	P	P	4683	4683
ASCLIN10_RXFIFOCON	RxFIFO configuration register	108_H	P	P	4685	4685
ASCLIN10_BITCON	Bit configuration register	$10C_H$	P	P	4687	4687
ASCLIN10_FRAMECON	Frame control register	110_H	P	P	4689	4689
ASCLIN10_DATCON	Data configuration register	114_H	P	P	Kernel Reset	4691
ASCLIN10_BRG	Baud rate generation register	118_H	P	P	4692	4692
ASCLIN10_BRD	Baud rate detection register	$11C_H$	P	P	Kernel Reset	4693
ASCLIN10_LINCON	LIN control register	120_H	P	P	Kernel Reset	4694
ASCLIN10_LINBTIMER	LIN break timer register	124_H	P	P	Kernel Reset	4695
ASCLIN10_LINHTIMER	LIN header timer register	128_H	P	P	Kernel Reset	4696
ASCLIN10_FLAGS	Flags register	$12C_H$	P	BE	Kernel Reset	4696
ASCLIN10_FLAGSSET	Flags set register	130_H	P	P	Kernel Reset	4701
ASCLIN10_FLAGSCLEAR	Flags clear register	134_H	P	P	Kernel Reset	4704
ASCLIN10_FLAGSENABLE	Flags enable register	138_H	P	P	Kernel Reset	4708
ASCLIN10_CSR	Clock selection register	$13C_H$	P	P	4712	4712
ASCLIN10_TXDATA	Transmit data register	140_H	P	P	Kernel Reset	4713
ASCLIN10_TXDATA# (#=0-7)	Transmit data register mirror #	140_H+#*4	P	P	Kernel Reset	4713
ASCLIN10_RXDATA	Receive data register	160_H	P	BE	Kernel Reset	4713
ASCLIN10_RXDATA# (#=0-7)	Receive data register mirror #	160_H+#*4	P	BE	Kernel Reset	4713

(table continues...)

Table 1142 (continued) Register overview - ASCLIN10 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ASCLIN10_RXDATAD	Receive data debug register	180_H	P	BE	Kernel Reset	4714

24.8.4.14 Register overview - ASCLIN11 (ascending offset address)

Table 1143 Register overview - ASCLIN11 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ASCLIN11_CLC	Clock control register	000_H	P	P, SV, E	4670	4670
ASCLIN11_OCS	OCDS control and status register	004_H	P	SV, P	Debug Reset	4671
ASCLIN11_ID	Module identification register	008_H	P	BE	PowerOn Reset	4672
ASCLIN11_RST_CTRLA	Reset control register A	$00C_H$	P	P, SV, E	Application Reset	4673
ASCLIN11_RST_CTRLB	Reset control register B	010_H	P	P, SV, E	Application Reset	4673
ASCLIN11_RST_STAT	Reset status register	014_H	P	BE	Application Reset	4674
ASCLIN11_PROTE	PROT register endinit	020_H	U	SV, PROT	Application Reset	4675
ASCLIN11_PROTSE	PROT register safe endinit	024_H	U	SV, PROT	Application Reset	4676
ASCLIN11_ACCEN_WRA	Write access enable register A	040_H	U	SE, SV	Application Reset	4678
ASCLIN11_ACCEN_WRB	Write access enable register B	044_H	U	SE, SV	Application Reset	4678
ASCLIN11_ACCEN_RDA	Read access enable register A	048_H	U	SE, SV	Application Reset	4679
ASCLIN11_ACCEN_RDB	Read access enable register B	$04C_H$	U	SE, SV	Application Reset	4679
ASCLIN11_ACCEN_VM	VM access enable register	050_H	U	SE, SV	Application Reset	4680
ASCLIN11_ACCEN_PRS	PRS access enable register	054_H	U	SE, SV	Application Reset	4680
ASCLIN11_IOCR	Input and output control register	100_H	P	P	4681	4681
ASCLIN11_TXFIFOCON	TxFIFO configuration register	104_H	P	P	4683	4683
ASCLIN11_RXFIFOCON	RxFIFO configuration register	108_H	P	P	4685	4685

(table continues...)

Table 1143 (continued) Register overview - ASCLIN11 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ASCLIN11_BITCON	Bit configuration register	$10C_H$	P	P	4687	4687
ASCLIN11_FRAMECON	Frame control register	110_H	P	P	4689	4689
ASCLIN11_DATCON	Data configuration register	114_H	P	P	Kernel Reset	4691
ASCLIN11_BRG	Baud rate generation register	118_H	P	P	4692	4692
ASCLIN11_BRD	Baud rate detection register	$11C_H$	P	P	Kernel Reset	4693
ASCLIN11_LINCON	LIN control register	120_H	P	P	Kernel Reset	4694
ASCLIN11_LINBTIMER	LIN break timer register	124_H	P	P	Kernel Reset	4695
ASCLIN11_LINHTIMER	LIN header timer register	128_H	P	P	Kernel Reset	4696
ASCLIN11_FLAGS	Flags register	$12C_H$	P	BE	Kernel Reset	4696
ASCLIN11_FLAGSSET	Flags set register	130_H	P	P	Kernel Reset	4701
ASCLIN11_FLAGSCLEAR	Flags clear register	134_H	P	P	Kernel Reset	4704
ASCLIN11_FLAGSENABLE	Flags enable register	138_H	P	P	Kernel Reset	4708
ASCLIN11_CSR	Clock selection register	$13C_H$	P	P	4712	4712
ASCLIN11_TXDATA	Transmit data register	140_H	P	P	Kernel Reset	4713
ASCLIN11_TXDATA# (#=0-7)	Transmit data register mirror #	140_H+#*4	P	P	Kernel Reset	4713
ASCLIN11_RXDATA	Receive data register	160_H	P	BE	Kernel Reset	4713
ASCLIN11_RXDATA# (#=0-7)	Receive data register mirror #	160_H+#*4	P	BE	Kernel Reset	4713
ASCLIN11_RXDATAD	Receive data debug register	180_H	P	BE	Kernel Reset	4714

24.8.4.15 Register overview - ASCLIN12 (ascending offset address)

Table 1144 Register overview - ASCLIN12 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ASCLIN12_CLC	Clock control register	000_H	P	P, SV, E	4670	4670

(table continues...)

Table 1144 (continued) Register overview - ASCLIN12 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ASCLIN12_OCS	OCDS control and status register	004_H	P	SV, P	Debug Reset	4671
ASCLIN12_ID	Module identification register	008_H	P	BE	PowerOn Reset	4672
ASCLIN12_RST_CTRLA	Reset control register A	$00C_H$	P	P, SV, E	Application Reset	4673
ASCLIN12_RST_CTRLB	Reset control register B	010_H	P	P, SV, E	Application Reset	4673
ASCLIN12_RST_STAT	Reset status register	014_H	P	BE	Application Reset	4674
ASCLIN12_PROTE	PROT register endinit	020_H	U	SV, PROT	Application Reset	4675
ASCLIN12_PROTSE	PROT register safe endinit	024_H	U	SV, PROT	Application Reset	4676
ASCLIN12_ACCEN_WRA	Write access enable register A	040_H	U	SE, SV	Application Reset	4678
ASCLIN12_ACCEN_WRB	Write access enable register B	044_H	U	SE, SV	Application Reset	4678
ASCLIN12_ACCEN_RDA	Read access enable register A	048_H	U	SE, SV	Application Reset	4679
ASCLIN12_ACCEN_RDB	Read access enable register B	$04C_H$	U	SE, SV	Application Reset	4679
ASCLIN12_ACCEN_VM	VM access enable register	050_H	U	SE, SV	Application Reset	4680
ASCLIN12_ACCEN_PRS	PRS access enable register	054_H	U	SE, SV	Application Reset	4680
ASCLIN12_IOCR	Input and output control register	100_H	P	P	4681	4681
ASCLIN12_TXFIFOCON	TxFIFO configuration register	104_H	P	P	4683	4683
ASCLIN12_RXFIFOCON	RxFIFO configuration register	108_H	P	P	4685	4685
ASCLIN12_BITCON	Bit configuration register	$10C_H$	P	P	4687	4687
ASCLIN12_FRAMECON	Frame control register	110_H	P	P	4689	4689
ASCLIN12_DATCON	Data configuration register	114_H	P	P	Kernel Reset	4691
ASCLIN12_BRG	Baud rate generation register	118_H	P	P	4692	4692
ASCLIN12_BRD	Baud rate detection register	$11C_H$	P	P	Kernel Reset	4693

(table continues...)

Table 1144 (continued) Register overview - ASCLIN12 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ASCLIN12_LINCON	LIN control register	120_H	P	P	Kernel Reset	4694
ASCLIN12_LINBTIMER	LIN break timer register	124_H	P	P	Kernel Reset	4695
ASCLIN12_LINHTIMER	LIN header timer register	128_H	P	P	Kernel Reset	4696
ASCLIN12_FLAGS	Flags register	$12C_H$	P	BE	Kernel Reset	4696
ASCLIN12_FLAGSSET	Flags set register	130_H	P	P	Kernel Reset	4701
ASCLIN12_FLAGSCLEAR	Flags clear register	134_H	P	P	Kernel Reset	4704
ASCLIN12_FLAGSENABLE	Flags enable register	138_H	P	P	Kernel Reset	4708
ASCLIN12_CSR	Clock selection register	$13C_H$	P	P	4712	4712
ASCLIN12_TXDATA	Transmit data register	140_H	P	P	Kernel Reset	4713
ASCLIN12_TXDATA# (#=0-7)	Transmit data register mirror #	140_H+#*4	P	P	Kernel Reset	4713
ASCLIN12_RXDATA	Receive data register	160_H	P	BE	Kernel Reset	4713
ASCLIN12_RXDATA# (#=0-7)	Receive data register mirror #	160_H+#*4	P	BE	Kernel Reset	4713
ASCLIN12_RXDATAD	Receive data debug register	180_H	P	BE	Kernel Reset	4714

24.8.4.16 Register overview - ASCLIN13 (ascending offset address)

Table 1145 Register overview - ASCLIN13 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ASCLIN13_CLC	Clock control register	000_H	P	P, SV, E	4670	4670
ASCLIN13_OCS	OCDS control and status register	004_H	P	SV, P	Debug Reset	4671
ASCLIN13_ID	Module identification register	008_H	P	BE	PowerOn Reset	4672
ASCLIN13_RST_CTRLA	Reset control register A	$00C_H$	P	P, SV, E	Application Reset	4673
ASCLIN13_RST_CTRLB	Reset control register B	010_H	P	P, SV, E	Application Reset	4673
ASCLIN13_RST_STAT	Reset status register	014_H	P	BE	Application Reset	4674

(table continues...)

Table 1145 (continued) Register overview - ASCLIN13 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ASCLIN13_PROTE	PROT register endinit	020_H	U	SV, PROT	Application Reset	4675
ASCLIN13_PROTSE	PROT register safe endinit	024_H	U	SV, PROT	Application Reset	4676
ASCLIN13_ACCEN_WRA	Write access enable register A	040_H	U	SE, SV	Application Reset	4678
ASCLIN13_ACCEN_WRB	Write access enable register B	044_H	U	SE, SV	Application Reset	4678
ASCLIN13_ACCEN_RDA	Read access enable register A	048_H	U	SE, SV	Application Reset	4679
ASCLIN13_ACCEN_RDB	Read access enable register B	$04C_H$	U	SE, SV	Application Reset	4679
ASCLIN13_ACCEN_VM	VM access enable register	050_H	U	SE, SV	Application Reset	4680
ASCLIN13_ACCEN_PRS	PRS access enable register	054_H	U	SE, SV	Application Reset	4680
ASCLIN13_IOCR	Input and output control register	100_H	P	P	4681	4681
ASCLIN13_TXFIFOCON	TxFIFO configuration register	104_H	P	P	4683	4683
ASCLIN13_RXFIFOCON	RxFIFO configuration register	108_H	P	P	4685	4685
ASCLIN13_BITCON	Bit configuration register	$10C_H$	P	P	4687	4687
ASCLIN13_FRAMECON	Frame control register	110_H	P	P	4689	4689
ASCLIN13_DATCON	Data configuration register	114_H	P	P	Kernel Reset	4691
ASCLIN13_BRG	Baud rate generation register	118_H	P	P	4692	4692
ASCLIN13_BRD	Baud rate detection register	$11C_H$	P	P	Kernel Reset	4693
ASCLIN13_LINCON	LIN control register	120_H	P	P	Kernel Reset	4694
ASCLIN13_LINBTIMER	LIN break timer register	124_H	P	P	Kernel Reset	4695
ASCLIN13_LINHTIMER	LIN header timer register	128_H	P	P	Kernel Reset	4696
ASCLIN13_FLAGS	Flags register	$12C_H$	P	BE	Kernel Reset	4696
ASCLIN13_FLAGSSET	Flags set register	130_H	P	P	Kernel Reset	4701

(table continues...)

Table 1145 (continued) Register overview - ASCLIN13 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ASCLIN13_FLAGSCLEAR	Flags clear register	134_H	P	P	Kernel Reset	4704
ASCLIN13_FLAGSENABLE	Flags enable register	138_H	P	P	Kernel Reset	4708
ASCLIN13_CSR	Clock selection register	$13C_H$	P	P	4712	4712
ASCLIN13_TXDATA	Transmit data register	140_H	P	P	Kernel Reset	4713
ASCLIN13_TXDATA# (#=0-7)	Transmit data register mirror #	140_H+#*4	P	P	Kernel Reset	4713
ASCLIN13_RXDATA	Receive data register	160_H	P	BE	Kernel Reset	4713
ASCLIN13_RXDATA# (#=0-7)	Receive data register mirror #	160_H+#*4	P	BE	Kernel Reset	4713
ASCLIN13_RXDATAD	Receive data debug register	180_H	P	BE	Kernel Reset	4714

24.8.4.17 Register overview - ASCLIN14 (ascending offset address)

Table 1146 Register overview - ASCLIN14 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ASCLIN14_CLC	Clock control register	000_H	P	P, SV, E	4670	4670
ASCLIN14_OCS	OCDS control and status register	004_H	P	SV, P	Debug Reset	4671
ASCLIN14_ID	Module identification register	008_H	P	BE	PowerOn Reset	4672
ASCLIN14_RST_CTRLA	Reset control register A	$00C_H$	P	P, SV, E	Application Reset	4673
ASCLIN14_RST_CTRLB	Reset control register B	010_H	P	P, SV, E	Application Reset	4673
ASCLIN14_RST_STAT	Reset status register	014_H	P	BE	Application Reset	4674
ASCLIN14_PROTE	PROT register endinit	020_H	U	SV, PROT	Application Reset	4675
ASCLIN14_PROTSE	PROT register safe endinit	024_H	U	SV, PROT	Application Reset	4676
ASCLIN14_ACCEN_WRA	Write access enable register A	040_H	U	SE, SV	Application Reset	4678
ASCLIN14_ACCEN_WRB	Write access enable register B	044_H	U	SE, SV	Application Reset	4678

(table continues...)

Table 1146 (continued) Register overview - ASCLIN14 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ASCLIN14_ACCEN_RDA	Read access enable register A	048_H	U	SE, SV	Application Reset	4679
ASCLIN14_ACCEN_RDB	Read access enable register B	$04C_H$	U	SE, SV	Application Reset	4679
ASCLIN14_ACCEN_VM	VM access enable register	050_H	U	SE, SV	Application Reset	4680
ASCLIN14_ACCEN_PRS	PRS access enable register	054_H	U	SE, SV	Application Reset	4680
ASCLIN14_IOCR	Input and output control register	100_H	P	P	4681	4681
ASCLIN14_TXFIFOCON	TxFIFO configuration register	104_H	P	P	4683	4683
ASCLIN14_RXFIFOCON	RxFIFO configuration register	108_H	P	P	4685	4685
ASCLIN14_BITCON	Bit configuration register	$10C_H$	P	P	4687	4687
ASCLIN14_FRAMECON	Frame control register	110_H	P	P	4689	4689
ASCLIN14_DATCON	Data configuration register	114_H	P	P	Kernel Reset	4691
ASCLIN14_BRG	Baud rate generation register	118_H	P	P	4692	4692
ASCLIN14_BRD	Baud rate detection register	$11C_H$	P	P	Kernel Reset	4693
ASCLIN14_LINCON	LIN control register	120_H	P	P	Kernel Reset	4694
ASCLIN14_LINBTIMER	LIN break timer register	124_H	P	P	Kernel Reset	4695
ASCLIN14_LINHTIMER	LIN header timer register	128_H	P	P	Kernel Reset	4696
ASCLIN14_FLAGS	Flags register	$12C_H$	P	BE	Kernel Reset	4696
ASCLIN14_FLAGSSET	Flags set register	130_H	P	P	Kernel Reset	4701
ASCLIN14_FLAGSCLEAR	Flags clear register	134_H	P	P	Kernel Reset	4704
ASCLIN14_FLAGSENABLE	Flags enable register	138_H	P	P	Kernel Reset	4708
ASCLIN14_CSR	Clock selection register	$13C_H$	P	P	4712	4712
ASCLIN14_TXDATA	Transmit data register	140_H	P	P	Kernel Reset	4713
ASCLIN14_TXDATA# (#=0-7)	Transmit data register mirror #	140_H+#*4	P	P	Kernel Reset	4713

(table continues...)

Table 1146 (continued) Register overview - ASCLIN14 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ASCLIN14_RXDATA	Receive data register	160_H	P	BE	Kernel Reset	4713
ASCLIN14_RXDATA# (#=0-7)	Receive data register mirror #	160_H+#*4	P	BE	Kernel Reset	4713
ASCLIN14_RXDATAD	Receive data debug register	180_H	P	BE	Kernel Reset	4714

24.8.4.18 Register overview - ASCLIN15 (ascending offset address)

Table 1147 Register overview - ASCLIN15 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ASCLIN15_CLC	Clock control register	000_H	P	P, SV, E	4670	4670
ASCLIN15_OCS	OCDS control and status register	004_H	P	SV, P	Debug Reset	4671
ASCLIN15_ID	Module identification register	008_H	P	BE	PowerOn Reset	4672
ASCLIN15_RST_CTRLA	Reset control register A	$00C_H$	P	P, SV, E	Application Reset	4673
ASCLIN15_RST_CTRLB	Reset control register B	010_H	P	P, SV, E	Application Reset	4673
ASCLIN15_RST_STAT	Reset status register	014_H	P	BE	Application Reset	4674
ASCLIN15_PROTE	PROT register endinit	020_H	U	SV, PROT	Application Reset	4675
ASCLIN15_PROTSE	PROT register safe endinit	024_H	U	SV, PROT	Application Reset	4676
ASCLIN15_ACCEN_WRA	Write access enable register A	040_H	U	SE, SV	Application Reset	4678
ASCLIN15_ACCEN_WRB	Write access enable register B	044_H	U	SE, SV	Application Reset	4678
ASCLIN15_ACCEN_RDA	Read access enable register A	048_H	U	SE, SV	Application Reset	4679
ASCLIN15_ACCEN_RDB	Read access enable register B	$04C_H$	U	SE, SV	Application Reset	4679
ASCLIN15_ACCEN_VM	VM access enable register	050_H	U	SE, SV	Application Reset	4680
ASCLIN15_ACCEN_PRS	PRS access enable register	054_H	U	SE, SV	Application Reset	4680
ASCLIN15_IOCR	Input and output control register	100_H	P	P	4681	4681

(table continues...)

Table 1147 (continued) Register overview - ASCLIN15 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ASCLIN15_TXFIFOCON	TxFIFO configuration register	104_H	P	P	4683	4683
ASCLIN15_RXFIFOCON	RxFIFO configuration register	108_H	P	P	4685	4685
ASCLIN15_BITCON	Bit configuration register	$10C_H$	P	P	4687	4687
ASCLIN15_FRAMECON	Frame control register	110_H	P	P	4689	4689
ASCLIN15_DATCON	Data configuration register	114_H	P	P	Kernel Reset	4691
ASCLIN15_BRG	Baud rate generation register	118_H	P	P	4692	4692
ASCLIN15_BRD	Baud rate detection register	$11C_H$	P	P	Kernel Reset	4693
ASCLIN15_LINCON	LIN control register	120_H	P	P	Kernel Reset	4694
ASCLIN15_LINBTIMER	LIN break timer register	124_H	P	P	Kernel Reset	4695
ASCLIN15_LINHTIMER	LIN header timer register	128_H	P	P	Kernel Reset	4696
ASCLIN15_FLAGS	Flags register	$12C_H$	P	BE	Kernel Reset	4696
ASCLIN15_FLAGSSET	Flags set register	130_H	P	P	Kernel Reset	4701
ASCLIN15_FLAGSCLEAR	Flags clear register	134_H	P	P	Kernel Reset	4704
ASCLIN15_FLAGSENABLE	Flags enable register	138_H	P	P	Kernel Reset	4708
ASCLIN15_CSR	Clock selection register	$13C_H$	P	P	4712	4712
ASCLIN15_TXDATA	Transmit data register	140_H	P	P	Kernel Reset	4713
ASCLIN15_TXDATA# (#=0-7)	Transmit data register mirror #	140_H+#*4	P	P	Kernel Reset	4713
ASCLIN15_RXDATA	Receive data register	160_H	P	BE	Kernel Reset	4713
ASCLIN15_RXDATA# (#=0-7)	Receive data register mirror #	160_H+#*4	P	BE	Kernel Reset	4713
ASCLIN15_RXDATAD	Receive data debug register	180_H	P	BE	Kernel Reset	4714

24.8.4.19 Register overview - ASCLIN16 (ascending offset address)

Table 1148 Register overview - ASCLIN16 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ASCLIN16_CLC	Clock control register	000_H	P	P, SV, E	4670	4670
ASCLIN16_OCS	OCDS control and status register	004_H	P	SV, P	Debug Reset	4671
ASCLIN16_ID	Module identification register	008_H	P	BE	PowerOn Reset	4672
ASCLIN16_RST_CTRLA	Reset control register A	$00C_H$	P	P, SV, E	Application Reset	4673
ASCLIN16_RST_CTRLB	Reset control register B	010_H	P	P, SV, E	Application Reset	4673
ASCLIN16_RST_STAT	Reset status register	014_H	P	BE	Application Reset	4674
ASCLIN16_PROTE	PROT register endinit	020_H	U	SV, PROT	Application Reset	4675
ASCLIN16_PROTSE	PROT register safe endinit	024_H	U	SV, PROT	Application Reset	4676
ASCLIN16_ACCEN_WRA	Write access enable register A	040_H	U	SE, SV	Application Reset	4678
ASCLIN16_ACCEN_WRB	Write access enable register B	044_H	U	SE, SV	Application Reset	4678
ASCLIN16_ACCEN_RDA	Read access enable register A	048_H	U	SE, SV	Application Reset	4679
ASCLIN16_ACCEN_RDB	Read access enable register B	$04C_H$	U	SE, SV	Application Reset	4679
ASCLIN16_ACCEN_VM	VM access enable register	050_H	U	SE, SV	Application Reset	4680
ASCLIN16_ACCEN_PRS	PRS access enable register	054_H	U	SE, SV	Application Reset	4680
ASCLIN16_IOCR	Input and output control register	100_H	P	P	4681	4681
ASCLIN16_TXFIFOCON	TxFIFO configuration register	104_H	P	P	4683	4683
ASCLIN16_RXFIFOCON	RxFIFO configuration register	108_H	P	P	4685	4685
ASCLIN16_BITCON	Bit configuration register	$10C_H$	P	P	4687	4687
ASCLIN16_FRAMECON	Frame control register	110_H	P	P	4689	4689
ASCLIN16_DATCON	Data configuration register	114_H	P	P	Kernel Reset	4691

(table continues...)

Table 1148 (continued) Register overview - ASCLIN16 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ASCLIN16_BRG	Baud rate generation register	118_H	P	P	4692	4692
ASCLIN16_BRD	Baud rate detection register	$11C_H$	P	P	Kernel Reset	4693
ASCLIN16_LINCON	LIN control register	120_H	P	P	Kernel Reset	4694
ASCLIN16_LINBTIMER	LIN break timer register	124_H	P	P	Kernel Reset	4695
ASCLIN16_LINHTIMER	LIN header timer register	128_H	P	P	Kernel Reset	4696
ASCLIN16_FLAGS	Flags register	$12C_H$	P	BE	Kernel Reset	4696
ASCLIN16_FLAGSSET	Flags set register	130_H	P	P	Kernel Reset	4701
ASCLIN16_FLAGSCLEAR	Flags clear register	134_H	P	P	Kernel Reset	4704
ASCLIN16_FLAGSENABLE	Flags enable register	138_H	P	P	Kernel Reset	4708
ASCLIN16_CSR	Clock selection register	$13C_H$	P	P	4712	4712
ASCLIN16_TXDATA	Transmit data register	140_H	P	P	Kernel Reset	4713
ASCLIN16_TXDATA# (#=0-7)	Transmit data register mirror #	140_H+#*4	P	P	Kernel Reset	4713
ASCLIN16_RXDATA	Receive data register	160_H	P	BE	Kernel Reset	4713
ASCLIN16_RXDATA# (#=0-7)	Receive data register mirror #	160_H+#*4	P	BE	Kernel Reset	4713
ASCLIN16_RXDATAD	Receive data debug register	180_H	P	BE	Kernel Reset	4714

24.8.4.20 Register overview - ASCLIN17 (ascending offset address)

Table 1149 Register overview - ASCLIN17 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ASCLIN17_CLC	Clock control register	000_H	P	P, SV, E	4670	4670
ASCLIN17_OCS	OCDS control and status register	004_H	P	SV, P	Debug Reset	4671
ASCLIN17_ID	Module identification register	008_H	P	BE	PowerOn Reset	4672
ASCLIN17_RST_CTRLA	Reset control register A	$00C_H$	P	P, SV, E	Application Reset	4673

(table continues...)

Table 1149 (continued) Register overview - ASCLIN17 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ASCLIN17_RST_CTRLB	Reset control register B	010_H	P	P, SV, E	Application Reset	4673
ASCLIN17_RST_STAT	Reset status register	014_H	P	BE	Application Reset	4674
ASCLIN17_PROTE	PROT register endinit	020_H	U	SV, PROT	Application Reset	4675
ASCLIN17_PROTSE	PROT register safe endinit	024_H	U	SV, PROT	Application Reset	4676
ASCLIN17_ACCEN_WRA	Write access enable register A	040_H	U	SE, SV	Application Reset	4678
ASCLIN17_ACCEN_WRB	Write access enable register B	044_H	U	SE, SV	Application Reset	4678
ASCLIN17_ACCEN_RDA	Read access enable register A	048_H	U	SE, SV	Application Reset	4679
ASCLIN17_ACCEN_RDB	Read access enable register B	$04C_H$	U	SE, SV	Application Reset	4679
ASCLIN17_ACCEN_VM	VM access enable register	050_H	U	SE, SV	Application Reset	4680
ASCLIN17_ACCEN_PRS	PRS access enable register	054_H	U	SE, SV	Application Reset	4680
ASCLIN17_IOCR	Input and output control register	100_H	P	P	4681	4681
ASCLIN17_TXFIFOCON	TxFIFO configuration register	104_H	P	P	4683	4683
ASCLIN17_RXFIFOCON	RxFIFO configuration register	108_H	P	P	4685	4685
ASCLIN17_BITCON	Bit configuration register	$10C_H$	P	P	4687	4687
ASCLIN17_FRAMECON	Frame control register	110_H	P	P	4689	4689
ASCLIN17_DATCON	Data configuration register	114_H	P	P	Kernel Reset	4691
ASCLIN17_BRG	Baud rate generation register	118_H	P	P	4692	4692
ASCLIN17_BRD	Baud rate detection register	$11C_H$	P	P	Kernel Reset	4693
ASCLIN17_LINCON	LIN control register	120_H	P	P	Kernel Reset	4694
ASCLIN17_LINBTIMER	LIN break timer register	124_H	P	P	Kernel Reset	4695
ASCLIN17_LINHTIMER	LIN header timer register	128_H	P	P	Kernel Reset	4696

(table continues...)

Table 1149 (continued) Register overview - ASCLIN17 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ASCLIN17_FLAGS	Flags register	$12C_H$	P	BE	Kernel Reset	4696
ASCLIN17_FLAGSSET	Flags set register	130_H	P	P	Kernel Reset	4701
ASCLIN17_FLAGSCLEAR	Flags clear register	134_H	P	P	Kernel Reset	4704
ASCLIN17_FLAGSENABLE	Flags enable register	138_H	P	P	Kernel Reset	4708
ASCLIN17_CSR	Clock selection register	$13C_H$	P	P	4712	4712
ASCLIN17_TXDATA	Transmit data register	140_H	P	P	Kernel Reset	4713
ASCLIN17_TXDATA# (#=0-7)	Transmit data register mirror #	140_H+#*4	P	P	Kernel Reset	4713
ASCLIN17_RXDATA	Receive data register	160_H	P	BE	Kernel Reset	4713
ASCLIN17_RXDATA# (#=0-7)	Receive data register mirror #	160_H+#*4	P	BE	Kernel Reset	4713
ASCLIN17_RXDATAD	Receive data debug register	180_H	P	BE	Kernel Reset	4714

24.8.4.21 Register overview - ASCLIN18 (ascending offset address)

Table 1150 Register overview - ASCLIN18 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ASCLIN18_CLC	Clock control register	000_H	P	P, SV, E	4670	4670
ASCLIN18_OCS	OCDS control and status register	004_H	P	SV, P	Debug Reset	4671
ASCLIN18_ID	Module identification register	008_H	P	BE	PowerOn Reset	4672
ASCLIN18_RST_CTRLA	Reset control register A	$00C_H$	P	P, SV, E	Application Reset	4673
ASCLIN18_RST_CTRLB	Reset control register B	010_H	P	P, SV, E	Application Reset	4673
ASCLIN18_RST_STAT	Reset status register	014_H	P	BE	Application Reset	4674
ASCLIN18_PROTE	PROT register endinit	020_H	U	SV, PROT	Application Reset	4675
ASCLIN18_PROTSE	PROT register safe endinit	024_H	U	SV, PROT	Application Reset	4676
ASCLIN18_ACCEN_WRA	Write access enable register A	040_H	U	SE, SV	Application Reset	4678

(table continues...)

Table 1150 (continued) Register overview - ASCLIN18 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ASCLIN18_ACCEN_WRB	Write access enable register B	044_H	U	SE, SV	Application Reset	4678
ASCLIN18_ACCEN_RDA	Read access enable register A	048_H	U	SE, SV	Application Reset	4679
ASCLIN18_ACCEN_RDB	Read access enable register B	$04C_H$	U	SE, SV	Application Reset	4679
ASCLIN18_ACCEN_VM	VM access enable register	050_H	U	SE, SV	Application Reset	4680
ASCLIN18_ACCEN_PRS	PRS access enable register	054_H	U	SE, SV	Application Reset	4680
ASCLIN18_IOCR	Input and output control register	100_H	P	P	4681	4681
ASCLIN18_TXFIFOCON	TxFIFO configuration register	104_H	P	P	4683	4683
ASCLIN18_RXFIFOCON	RxFIFO configuration register	108_H	P	P	4685	4685
ASCLIN18_BITCON	Bit configuration register	$10C_H$	P	P	4687	4687
ASCLIN18_FRAMECON	Frame control register	110_H	P	P	4689	4689
ASCLIN18_DATCON	Data configuration register	114_H	P	P	Kernel Reset	4691
ASCLIN18_BRG	Baud rate generation register	118_H	P	P	4692	4692
ASCLIN18_BRD	Baud rate detection register	$11C_H$	P	P	Kernel Reset	4693
ASCLIN18_LINCON	LIN control register	120_H	P	P	Kernel Reset	4694
ASCLIN18_LINBTIMER	LIN break timer register	124_H	P	P	Kernel Reset	4695
ASCLIN18_LINHTIMER	LIN header timer register	128_H	P	P	Kernel Reset	4696
ASCLIN18_FLAGS	Flags register	$12C_H$	P	BE	Kernel Reset	4696
ASCLIN18_FLAGSSET	Flags set register	130_H	P	P	Kernel Reset	4701
ASCLIN18_FLAGSCLEAR	Flags clear register	134_H	P	P	Kernel Reset	4704
ASCLIN18_FLAGSENABLE	Flags enable register	138_H	P	P	Kernel Reset	4708
ASCLIN18_CSR	Clock selection register	$13C_H$	P	P	4712	4712
ASCLIN18_TXDATA	Transmit data register	140_H	P	P	Kernel Reset	4713

(table continues...)

Table 1150 (continued) Register overview - ASCLIN18 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ASCLIN18_TXDATA# (#=0-7)	Transmit data register mirror #	140_H+#*4	P	P	Kernel Reset	4713
ASCLIN18_RXDATA	Receive data register	160_H	P	BE	Kernel Reset	4713
ASCLIN18_RXDATA# (#=0-7)	Receive data register mirror #	160_H+#*4	P	BE	Kernel Reset	4713
ASCLIN18_RXDATAD	Receive data debug register	180_H	P	BE	Kernel Reset	4714

24.8.4.22 Register overview - ASCLIN19 (ascending offset address)

Table 1151 Register overview - ASCLIN19 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ASCLIN19_CLC	Clock control register	000_H	P	P, SV, E	4670	4670
ASCLIN19_OCS	OCDS control and status register	004_H	P	SV, P	Debug Reset	4671
ASCLIN19_ID	Module identification register	008_H	P	BE	PowerOn Reset	4672
ASCLIN19_RST_CTRLA	Reset control register A	$00C_H$	P	P, SV, E	Application Reset	4673
ASCLIN19_RST_CTRLB	Reset control register B	010_H	P	P, SV, E	Application Reset	4673
ASCLIN19_RST_STAT	Reset status register	014_H	P	BE	Application Reset	4674
ASCLIN19_PROTE	PROT register endinit	020_H	U	SV, PROT	Application Reset	4675
ASCLIN19_PROTSE	PROT register safe endinit	024_H	U	SV, PROT	Application Reset	4676
ASCLIN19_ACCEN_WRA	Write access enable register A	040_H	U	SE, SV	Application Reset	4678
ASCLIN19_ACCEN_WRB	Write access enable register B	044_H	U	SE, SV	Application Reset	4678
ASCLIN19_ACCEN_RDA	Read access enable register A	048_H	U	SE, SV	Application Reset	4679
ASCLIN19_ACCEN_RDB	Read access enable register B	$04C_H$	U	SE, SV	Application Reset	4679
ASCLIN19_ACCEN_VM	VM access enable register	050_H	U	SE, SV	Application Reset	4680
ASCLIN19_ACCEN_PRS	PRS access enable register	054_H	U	SE, SV	Application Reset	4680

(table continues...)

Table 1151 (continued) Register overview - ASCLIN19 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ASCLIN19_IOCR	Input and output control register	100_H	P	P	4681	4681
ASCLIN19_TXFIFOCON	TxFIFO configuration register	104_H	P	P	4683	4683
ASCLIN19_RXFIFOCON	RxFIFO configuration register	108_H	P	P	4685	4685
ASCLIN19_BITCON	Bit configuration register	$10C_H$	P	P	4687	4687
ASCLIN19_FRAMECON	Frame control register	110_H	P	P	4689	4689
ASCLIN19_DATCON	Data configuration register	114_H	P	P	Kernel Reset	4691
ASCLIN19_BRG	Baud rate generation register	118_H	P	P	4692	4692
ASCLIN19_BRD	Baud rate detection register	$11C_H$	P	P	Kernel Reset	4693
ASCLIN19_LINCON	LIN control register	120_H	P	P	Kernel Reset	4694
ASCLIN19_LINBTIMER	LIN break timer register	124_H	P	P	Kernel Reset	4695
ASCLIN19_LINHTIMER	LIN header timer register	128_H	P	P	Kernel Reset	4696
ASCLIN19_FLAGS	Flags register	$12C_H$	P	BE	Kernel Reset	4696
ASCLIN19_FLAGSSET	Flags set register	130_H	P	P	Kernel Reset	4701
ASCLIN19_FLAGSCLEAR	Flags clear register	134_H	P	P	Kernel Reset	4704
ASCLIN19_FLAGSENABLE	Flags enable register	138_H	P	P	Kernel Reset	4708
ASCLIN19_CSR	Clock selection register	$13C_H$	P	P	4712	4712
ASCLIN19_TXDATA	Transmit data register	140_H	P	P	Kernel Reset	4713
ASCLIN19_TXDATA# (#=0-7)	Transmit data register mirror #	140_H+#*4	P	P	Kernel Reset	4713
ASCLIN19_RXDATA	Receive data register	160_H	P	BE	Kernel Reset	4713
ASCLIN19_RXDATA# (#=0-7)	Receive data register mirror #	160_H+#*4	P	BE	Kernel Reset	4713
ASCLIN19_RXDATAD	Receive data debug register	180_H	P	BE	Kernel Reset	4714

24.8.4.23 Register overview - ASCLIN20 (ascending offset address)

Table 1152 Register overview - ASCLIN20 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ASCLIN20_CLC	Clock control register	000_H	P	P, SV, E	4670	4670
ASCLIN20_OCS	OCDS control and status register	004_H	P	SV, P	Debug Reset	4671
ASCLIN20_ID	Module identification register	008_H	P	BE	PowerOn Reset	4672
ASCLIN20_RST_CTRLA	Reset control register A	$00C_H$	P	P, SV, E	Application Reset	4673
ASCLIN20_RST_CTRLB	Reset control register B	010_H	P	P, SV, E	Application Reset	4673
ASCLIN20_RST_STAT	Reset status register	014_H	P	BE	Application Reset	4674
ASCLIN20_PROTE	PROT register endinit	020_H	U	SV, PROT	Application Reset	4675
ASCLIN20_PROTSE	PROT register safe endinit	024_H	U	SV, PROT	Application Reset	4676
ASCLIN20_ACCEN_WRA	Write access enable register A	040_H	U	SE, SV	Application Reset	4678
ASCLIN20_ACCEN_WRB	Write access enable register B	044_H	U	SE, SV	Application Reset	4678
ASCLIN20_ACCEN_RDA	Read access enable register A	048_H	U	SE, SV	Application Reset	4679
ASCLIN20_ACCEN_RDB	Read access enable register B	$04C_H$	U	SE, SV	Application Reset	4679
ASCLIN20_ACCEN_VM	VM access enable register	050_H	U	SE, SV	Application Reset	4680
ASCLIN20_ACCEN_PRS	PRS access enable register	054_H	U	SE, SV	Application Reset	4680
ASCLIN20_IOCR	Input and output control register	100_H	P	P	4681	4681
ASCLIN20_TXFIFOCON	TxFIFO configuration register	104_H	P	P	4683	4683
ASCLIN20_RXFIFOCON	RxFIFO configuration register	108_H	P	P	4685	4685
ASCLIN20_BITCON	Bit configuration register	$10C_H$	P	P	4687	4687
ASCLIN20_FRAMECON	Frame control register	110_H	P	P	4689	4689
ASCLIN20_DATCON	Data configuration register	114_H	P	P	Kernel Reset	4691

(table continues...)

Table 1152 (continued) Register overview - ASCLIN20 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ASCLIN20_BRG	Baud rate generation register	118_H	P	P	4692	4692
ASCLIN20_BRD	Baud rate detection register	$11C_H$	P	P	Kernel Reset	4693
ASCLIN20_LINCON	LIN control register	120_H	P	P	Kernel Reset	4694
ASCLIN20_LINBTIMER	LIN break timer register	124_H	P	P	Kernel Reset	4695
ASCLIN20_LINHTIMER	LIN header timer register	128_H	P	P	Kernel Reset	4696
ASCLIN20_FLAGS	Flags register	$12C_H$	P	BE	Kernel Reset	4696
ASCLIN20_FLAGSSET	Flags set register	130_H	P	P	Kernel Reset	4701
ASCLIN20_FLAGSCLEAR	Flags clear register	134_H	P	P	Kernel Reset	4704
ASCLIN20_FLAGSENABLE	Flags enable register	138_H	P	P	Kernel Reset	4708
ASCLIN20_CSR	Clock selection register	$13C_H$	P	P	4712	4712
ASCLIN20_TXDATA	Transmit data register	140_H	P	P	Kernel Reset	4713
ASCLIN20_TXDATA# (#=0-7)	Transmit data register mirror #	140_H+#*4	P	P	Kernel Reset	4713
ASCLIN20_RXDATA	Receive data register	160_H	P	BE	Kernel Reset	4713
ASCLIN20_RXDATA# (#=0-7)	Receive data register mirror #	160_H+#*4	P	BE	Kernel Reset	4713
ASCLIN20_RXDATAD	Receive data debug register	180_H	P	BE	Kernel Reset	4714

24.8.4.24 Register overview - ASCLIN21 (ascending offset address)

Table 1153 Register overview - ASCLIN21 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ASCLIN21_CLC	Clock control register	000_H	P	P, SV, E	4670	4670
ASCLIN21_OCS	OCDS control and status register	004_H	P	SV, P	Debug Reset	4671
ASCLIN21_ID	Module identification register	008_H	P	BE	PowerOn Reset	4672
ASCLIN21_RST_CTRLA	Reset control register A	$00C_H$	P	P, SV, E	Application Reset	4673

(table continues...)

Table 1153 (continued) Register overview - ASCLIN21 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ASCLIN21_RST_CTRLB	Reset control register B	010_H	P	P, SV, E	Application Reset	4673
ASCLIN21_RST_STAT	Reset status register	014_H	P	BE	Application Reset	4674
ASCLIN21_PROTE	PROT register endinit	020_H	U	SV, PROT	Application Reset	4675
ASCLIN21_PROTSE	PROT register safe endinit	024_H	U	SV, PROT	Application Reset	4676
ASCLIN21_ACCEN_WRA	Write access enable register A	040_H	U	SE, SV	Application Reset	4678
ASCLIN21_ACCEN_WRB	Write access enable register B	044_H	U	SE, SV	Application Reset	4678
ASCLIN21_ACCEN_RDA	Read access enable register A	048_H	U	SE, SV	Application Reset	4679
ASCLIN21_ACCEN_RDB	Read access enable register B	$04C_H$	U	SE, SV	Application Reset	4679
ASCLIN21_ACCEN_VM	VM access enable register	050_H	U	SE, SV	Application Reset	4680
ASCLIN21_ACCEN_PRS	PRS access enable register	054_H	U	SE, SV	Application Reset	4680
ASCLIN21_IOCR	Input and output control register	100_H	P	P	4681	4681
ASCLIN21_TXFIFOCON	TxFIFO configuration register	104_H	P	P	4683	4683
ASCLIN21_RXFIFOCON	RxFIFO configuration register	108_H	P	P	4685	4685
ASCLIN21_BITCON	Bit configuration register	$10C_H$	P	P	4687	4687
ASCLIN21_FRAMECON	Frame control register	110_H	P	P	4689	4689
ASCLIN21_DATCON	Data configuration register	114_H	P	P	Kernel Reset	4691
ASCLIN21_BRG	Baud rate generation register	118_H	P	P	4692	4692
ASCLIN21_BRD	Baud rate detection register	$11C_H$	P	P	Kernel Reset	4693
ASCLIN21_LINCON	LIN control register	120_H	P	P	Kernel Reset	4694
ASCLIN21_LINBTIMER	LIN break timer register	124_H	P	P	Kernel Reset	4695
ASCLIN21_LINHTIMER	LIN header timer register	128_H	P	P	Kernel Reset	4696

(table continues...)

Table 1153 (continued) Register overview - ASCLIN21 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ASCLIN21_FLAGS	Flags register	$12C_H$	P	BE	Kernel Reset	4696
ASCLIN21_FLAGSSET	Flags set register	130_H	P	P	Kernel Reset	4701
ASCLIN21_FLAGSCLEAR	Flags clear register	134_H	P	P	Kernel Reset	4704
ASCLIN21_FLAGSENABLE	Flags enable register	138_H	P	P	Kernel Reset	4708
ASCLIN21_CSR	Clock selection register	$13C_H$	P	P	4712	4712
ASCLIN21_TXDATA	Transmit data register	140_H	P	P	Kernel Reset	4713
ASCLIN21_TXDATA# (#=0-7)	Transmit data register mirror #	140_H+#*4	P	P	Kernel Reset	4713
ASCLIN21_RXDATA	Receive data register	160_H	P	BE	Kernel Reset	4713
ASCLIN21_RXDATA# (#=0-7)	Receive data register mirror #	160_H+#*4	P	BE	Kernel Reset	4713
ASCLIN21_RXDATAD	Receive data debug register	180_H	P	BE	Kernel Reset	4714

24.8.4.25 Register overview - ASCLIN22 (ascending offset address)

Table 1154 Register overview - ASCLIN22 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ASCLIN22_CLC	Clock control register	000_H	P	P, SV, E	4670	4670
ASCLIN22_OCS	OCDS control and status register	004_H	P	SV, P	Debug Reset	4671
ASCLIN22_ID	Module identification register	008_H	P	BE	PowerOn Reset	4672
ASCLIN22_RST_CTRLA	Reset control register A	$00C_H$	P	P, SV, E	Application Reset	4673
ASCLIN22_RST_CTRLB	Reset control register B	010_H	P	P, SV, E	Application Reset	4673
ASCLIN22_RST_STAT	Reset status register	014_H	P	BE	Application Reset	4674
ASCLIN22_PROTE	PROT register endinit	020_H	U	SV, PROT	Application Reset	4675
ASCLIN22_PROTSE	PROT register safe endinit	024_H	U	SV, PROT	Application Reset	4676
ASCLIN22_ACCEN_WRA	Write access enable register A	040_H	U	SE, SV	Application Reset	4678

(table continues...)

Table 1154 (continued) Register overview - ASCLIN22 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ASCLIN22_ACCEN_WRB	Write access enable register B	044_H	U	SE, SV	Application Reset	4678
ASCLIN22_ACCEN_RDA	Read access enable register A	048_H	U	SE, SV	Application Reset	4679
ASCLIN22_ACCEN_RDB	Read access enable register B	$04C_H$	U	SE, SV	Application Reset	4679
ASCLIN22_ACCEN_VM	VM access enable register	050_H	U	SE, SV	Application Reset	4680
ASCLIN22_ACCEN_PRS	PRS access enable register	054_H	U	SE, SV	Application Reset	4680
ASCLIN22_IOCR	Input and output control register	100_H	P	P	4681	4681
ASCLIN22_TXFIFOCON	TxFIFO configuration register	104_H	P	P	4683	4683
ASCLIN22_RXFIFOCON	RxFIFO configuration register	108_H	P	P	4685	4685
ASCLIN22_BITCON	Bit configuration register	$10C_H$	P	P	4687	4687
ASCLIN22_FRAMECON	Frame control register	110_H	P	P	4689	4689
ASCLIN22_DATCON	Data configuration register	114_H	P	P	Kernel Reset	4691
ASCLIN22_BRG	Baud rate generation register	118_H	P	P	4692	4692
ASCLIN22_BRD	Baud rate detection register	$11C_H$	P	P	Kernel Reset	4693
ASCLIN22_LINCON	LIN control register	120_H	P	P	Kernel Reset	4694
ASCLIN22_LINBTIMER	LIN break timer register	124_H	P	P	Kernel Reset	4695
ASCLIN22_LINHTIMER	LIN header timer register	128_H	P	P	Kernel Reset	4696
ASCLIN22_FLAGS	Flags register	$12C_H$	P	BE	Kernel Reset	4696
ASCLIN22_FLAGSSET	Flags set register	130_H	P	P	Kernel Reset	4701
ASCLIN22_FLAGSCLEAR	Flags clear register	134_H	P	P	Kernel Reset	4704
ASCLIN22_FLAGSENABLE	Flags enable register	138_H	P	P	Kernel Reset	4708
ASCLIN22_CSR	Clock selection register	$13C_H$	P	P	4712	4712
ASCLIN22_TXDATA	Transmit data register	140_H	P	P	Kernel Reset	4713

(table continues...)

Table 1154 (continued) Register overview - ASCLIN22 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ASCLIN22_TXDATA# (#=0-7)	Transmit data register mirror #	140_H+#*4	P	P	Kernel Reset	4713
ASCLIN22_RXDATA	Receive data register	160_H	P	BE	Kernel Reset	4713
ASCLIN22_RXDATA# (#=0-7)	Receive data register mirror #	160_H+#*4	P	BE	Kernel Reset	4713
ASCLIN22_RXDATAD	Receive data debug register	180_H	P	BE	Kernel Reset	4714

24.8.4.26 Register overview - ASCLIN23 (ascending offset address)

Table 1155 Register overview - ASCLIN23 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ASCLIN23_CLC	Clock control register	000_H	P	P, SV, E	4670	4670
ASCLIN23_OCS	OCDS control and status register	004_H	P	SV, P	Debug Reset	4671
ASCLIN23_ID	Module identification register	008_H	P	BE	PowerOn Reset	4672
ASCLIN23_RST_CTRLA	Reset control register A	$00C_H$	P	P, SV, E	Application Reset	4673
ASCLIN23_RST_CTRLB	Reset control register B	010_H	P	P, SV, E	Application Reset	4673
ASCLIN23_RST_STAT	Reset status register	014_H	P	BE	Application Reset	4674
ASCLIN23_PROTE	PROT register endinit	020_H	U	SV, PROT	Application Reset	4675
ASCLIN23_PROTSE	PROT register safe endinit	024_H	U	SV, PROT	Application Reset	4676
ASCLIN23_ACCEN_WRA	Write access enable register A	040_H	U	SE, SV	Application Reset	4678
ASCLIN23_ACCEN_WRB	Write access enable register B	044_H	U	SE, SV	Application Reset	4678
ASCLIN23_ACCEN_RDA	Read access enable register A	048_H	U	SE, SV	Application Reset	4679
ASCLIN23_ACCEN_RDB	Read access enable register B	$04C_H$	U	SE, SV	Application Reset	4679
ASCLIN23_ACCEN_VM	VM access enable register	050_H	U	SE, SV	Application Reset	4680
ASCLIN23_ACCEN_PRS	PRS access enable register	054_H	U	SE, SV	Application Reset	4680

(table continues...)

Table 1155 (continued) Register overview - ASCLIN23 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ASCLIN23_IOCR	Input and output control register	100_H	P	P	4681	4681
ASCLIN23_TXFIFOCON	TxFIFO configuration register	104_H	P	P	4683	4683
ASCLIN23_RXFIFOCON	RxFIFO configuration register	108_H	P	P	4685	4685
ASCLIN23_BITCON	Bit configuration register	$10C_H$	P	P	4687	4687
ASCLIN23_FRAMECON	Frame control register	110_H	P	P	4689	4689
ASCLIN23_DATCON	Data configuration register	114_H	P	P	Kernel Reset	4691
ASCLIN23_BRG	Baud rate generation register	118_H	P	P	4692	4692
ASCLIN23_BRD	Baud rate detection register	$11C_H$	P	P	Kernel Reset	4693
ASCLIN23_LINCON	LIN control register	120_H	P	P	Kernel Reset	4694
ASCLIN23_LINBTIMER	LIN break timer register	124_H	P	P	Kernel Reset	4695
ASCLIN23_LINHTIMER	LIN header timer register	128_H	P	P	Kernel Reset	4696
ASCLIN23_FLAGS	Flags register	$12C_H$	P	BE	Kernel Reset	4696
ASCLIN23_FLAGSSET	Flags set register	130_H	P	P	Kernel Reset	4701
ASCLIN23_FLAGSCLEAR	Flags clear register	134_H	P	P	Kernel Reset	4704
ASCLIN23_FLAGSENABLE	Flags enable register	138_H	P	P	Kernel Reset	4708
ASCLIN23_CSR	Clock selection register	$13C_H$	P	P	4712	4712
ASCLIN23_TXDATA	Transmit data register	140_H	P	P	Kernel Reset	4713
ASCLIN23_TXDATA# (#=0-7)	Transmit data register mirror #	140_H+#*4	P	P	Kernel Reset	4713
ASCLIN23_RXDATA	Receive data register	160_H	P	BE	Kernel Reset	4713
ASCLIN23_RXDATA# (#=0-7)	Receive data register mirror #	160_H+#*4	P	BE	Kernel Reset	4713
ASCLIN23_RXDATAD	Receive data debug register	180_H	P	BE	Kernel Reset	4714

24.8.4.27 Register overview - ASCLIN24 (ascending offset address)

Table 1156 Register overview - ASCLIN24 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ASCLIN24_CLC	Clock control register	000_H	P	P, SV, E	4670	4670
ASCLIN24_OCS	OCDS control and status register	004_H	P	SV, P	Debug Reset	4671
ASCLIN24_ID	Module identification register	008_H	P	BE	PowerOn Reset	4672
ASCLIN24_RST_CTRLA	Reset control register A	$00C_H$	P	P, SV, E	Application Reset	4673
ASCLIN24_RST_CTRLB	Reset control register B	010_H	P	P, SV, E	Application Reset	4673
ASCLIN24_RST_STAT	Reset status register	014_H	P	BE	Application Reset	4674
ASCLIN24_PROTE	PROT register endinit	020_H	U	SV, PROT	Application Reset	4675
ASCLIN24_PROTSE	PROT register safe endinit	024_H	U	SV, PROT	Application Reset	4676
ASCLIN24_ACCEN_WRA	Write access enable register A	040_H	U	SE, SV	Application Reset	4678
ASCLIN24_ACCEN_WRB	Write access enable register B	044_H	U	SE, SV	Application Reset	4678
ASCLIN24_ACCEN_RDA	Read access enable register A	048_H	U	SE, SV	Application Reset	4679
ASCLIN24_ACCEN_RDB	Read access enable register B	$04C_H$	U	SE, SV	Application Reset	4679
ASCLIN24_ACCEN_VM	VM access enable register	050_H	U	SE, SV	Application Reset	4680
ASCLIN24_ACCEN_PRS	PRS access enable register	054_H	U	SE, SV	Application Reset	4680
ASCLIN24_IOCR	Input and output control register	100_H	P	P	4681	4681
ASCLIN24_TXFIFOCON	TxFIFO configuration register	104_H	P	P	4683	4683
ASCLIN24_RXFIFOCON	RxFIFO configuration register	108_H	P	P	4685	4685
ASCLIN24_BITCON	Bit configuration register	$10C_H$	P	P	4687	4687
ASCLIN24_FRAMECON	Frame control register	110_H	P	P	4689	4689
ASCLIN24_DATCON	Data configuration register	114_H	P	P	Kernel Reset	4691

(table continues...)

Table 1156 (continued) Register overview - ASCLIN24 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ASCLIN24_BRG	Baud rate generation register	118_H	P	P	4692	4692
ASCLIN24_BRD	Baud rate detection register	$11C_H$	P	P	Kernel Reset	4693
ASCLIN24_LINCON	LIN control register	120_H	P	P	Kernel Reset	4694
ASCLIN24_LINBTIMER	LIN break timer register	124_H	P	P	Kernel Reset	4695
ASCLIN24_LINHTIMER	LIN header timer register	128_H	P	P	Kernel Reset	4696
ASCLIN24_FLAGS	Flags register	$12C_H$	P	BE	Kernel Reset	4696
ASCLIN24_FLAGSSET	Flags set register	130_H	P	P	Kernel Reset	4701
ASCLIN24_FLAGSCLEAR	Flags clear register	134_H	P	P	Kernel Reset	4704
ASCLIN24_FLAGSENABLE	Flags enable register	138_H	P	P	Kernel Reset	4708
ASCLIN24_CSR	Clock selection register	$13C_H$	P	P	4712	4712
ASCLIN24_TXDATA	Transmit data register	140_H	P	P	Kernel Reset	4713
ASCLIN24_TXDATA# (#=0-7)	Transmit data register mirror #	140_H+#*4	P	P	Kernel Reset	4713
ASCLIN24_RXDATA	Receive data register	160_H	P	BE	Kernel Reset	4713
ASCLIN24_RXDATA# (#=0-7)	Receive data register mirror #	160_H+#*4	P	BE	Kernel Reset	4713
ASCLIN24_RXDATAD	Receive data debug register	180_H	P	BE	Kernel Reset	4714

24.8.4.28 Register overview - ASCLIN25 (ascending offset address)

Table 1157 Register overview - ASCLIN25 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ASCLIN25_CLC	Clock control register	000_H	P	P, SV, E	4670	4670
ASCLIN25_OCS	OCDS control and status register	004_H	P	SV, P	Debug Reset	4671
ASCLIN25_ID	Module identification register	008_H	P	BE	PowerOn Reset	4672
ASCLIN25_RST_CTRLA	Reset control register A	$00C_H$	P	P, SV, E	Application Reset	4673

(table continues...)

Table 1157 (continued) Register overview - ASCLIN25 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ASCLIN25_RST_CTRLB	Reset control register B	010_H	P	P, SV, E	Application Reset	4673
ASCLIN25_RST_STAT	Reset status register	014_H	P	BE	Application Reset	4674
ASCLIN25_PROTE	PROT register endinit	020_H	U	SV, PROT	Application Reset	4675
ASCLIN25_PROTSE	PROT register safe endinit	024_H	U	SV, PROT	Application Reset	4676
ASCLIN25_ACCEN_WRA	Write access enable register A	040_H	U	SE, SV	Application Reset	4678
ASCLIN25_ACCEN_WRB	Write access enable register B	044_H	U	SE, SV	Application Reset	4678
ASCLIN25_ACCEN_RDA	Read access enable register A	048_H	U	SE, SV	Application Reset	4679
ASCLIN25_ACCEN_RDB	Read access enable register B	$04C_H$	U	SE, SV	Application Reset	4679
ASCLIN25_ACCEN_VM	VM access enable register	050_H	U	SE, SV	Application Reset	4680
ASCLIN25_ACCEN_PRS	PRS access enable register	054_H	U	SE, SV	Application Reset	4680
ASCLIN25_IOCR	Input and output control register	100_H	P	P	4681	4681
ASCLIN25_TXFIFOCON	TxFIFO configuration register	104_H	P	P	4683	4683
ASCLIN25_RXFIFOCON	RxFIFO configuration register	108_H	P	P	4685	4685
ASCLIN25_BITCON	Bit configuration register	$10C_H$	P	P	4687	4687
ASCLIN25_FRAMECON	Frame control register	110_H	P	P	4689	4689
ASCLIN25_DATCON	Data configuration register	114_H	P	P	Kernel Reset	4691
ASCLIN25_BRG	Baud rate generation register	118_H	P	P	4692	4692
ASCLIN25_BRD	Baud rate detection register	$11C_H$	P	P	Kernel Reset	4693
ASCLIN25_LINCON	LIN control register	120_H	P	P	Kernel Reset	4694
ASCLIN25_LINBTIMER	LIN break timer register	124_H	P	P	Kernel Reset	4695
ASCLIN25_LINHTIMER	LIN header timer register	128_H	P	P	Kernel Reset	4696

(table continues...)

Table 1157 (continued) Register overview - ASCLIN25 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ASCLIN25_FLAGS	Flags register	$12C_H$	P	BE	Kernel Reset	4696
ASCLIN25_FLAGSSET	Flags set register	130_H	P	P	Kernel Reset	4701
ASCLIN25_FLAGSCLEAR	Flags clear register	134_H	P	P	Kernel Reset	4704
ASCLIN25_FLAGSENABLE	Flags enable register	138_H	P	P	Kernel Reset	4708
ASCLIN25_CSR	Clock selection register	$13C_H$	P	P	4712	4712
ASCLIN25_TXDATA	Transmit data register	140_H	P	P	Kernel Reset	4713
ASCLIN25_TXDATA# (#=0-7)	Transmit data register mirror #	140_H+#*4	P	P	Kernel Reset	4713
ASCLIN25_RXDATA	Receive data register	160_H	P	BE	Kernel Reset	4713
ASCLIN25_RXDATA# (#=0-7)	Receive data register mirror #	160_H+#*4	P	BE	Kernel Reset	4713
ASCLIN25_RXDATAD	Receive data debug register	180_H	P	BE	Kernel Reset	4714

24.8.4.29 Register overview - ASCLIN26 (ascending offset address)

Table 1158 Register overview - ASCLIN26 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ASCLIN26_CLC	Clock control register	000_H	P	P, SV, E	4670	4670
ASCLIN26_OCS	OCDS control and status register	004_H	P	SV, P	Debug Reset	4671
ASCLIN26_ID	Module identification register	008_H	P	BE	PowerOn Reset	4672
ASCLIN26_RST_CTRLA	Reset control register A	$00C_H$	P	P, SV, E	Application Reset	4673
ASCLIN26_RST_CTRLB	Reset control register B	010_H	P	P, SV, E	Application Reset	4673
ASCLIN26_RST_STAT	Reset status register	014_H	P	BE	Application Reset	4674
ASCLIN26_PROTE	PROT register endinit	020_H	U	SV, PROT	Application Reset	4675
ASCLIN26_PROTSE	PROT register safe endinit	024_H	U	SV, PROT	Application Reset	4676
ASCLIN26_ACCEN_WRA	Write access enable register A	040_H	U	SE, SV	Application Reset	4678

(table continues...)

Table 1158 (continued) Register overview - ASCLIN26 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ASCLIN26_ACCEN_WRB	Write access enable register B	044_H	U	SE, SV	Application Reset	4678
ASCLIN26_ACCEN_RDA	Read access enable register A	048_H	U	SE, SV	Application Reset	4679
ASCLIN26_ACCEN_RDB	Read access enable register B	$04C_H$	U	SE, SV	Application Reset	4679
ASCLIN26_ACCEN_VM	VM access enable register	050_H	U	SE, SV	Application Reset	4680
ASCLIN26_ACCEN_PRS	PRS access enable register	054_H	U	SE, SV	Application Reset	4680
ASCLIN26_IOCR	Input and output control register	100_H	P	P	4681	4681
ASCLIN26_TXFIFOCON	TxFIFO configuration register	104_H	P	P	4683	4683
ASCLIN26_RXFIFOCON	RxFIFO configuration register	108_H	P	P	4685	4685
ASCLIN26_BITCON	Bit configuration register	$10C_H$	P	P	4687	4687
ASCLIN26_FRAMECON	Frame control register	110_H	P	P	4689	4689
ASCLIN26_DATCON	Data configuration register	114_H	P	P	Kernel Reset	4691
ASCLIN26_BRG	Baud rate generation register	118_H	P	P	4692	4692
ASCLIN26_BRD	Baud rate detection register	$11C_H$	P	P	Kernel Reset	4693
ASCLIN26_LINCON	LIN control register	120_H	P	P	Kernel Reset	4694
ASCLIN26_LINBTIMER	LIN break timer register	124_H	P	P	Kernel Reset	4695
ASCLIN26_LINHTIMER	LIN header timer register	128_H	P	P	Kernel Reset	4696
ASCLIN26_FLAGS	Flags register	$12C_H$	P	BE	Kernel Reset	4696
ASCLIN26_FLAGSSET	Flags set register	130_H	P	P	Kernel Reset	4701
ASCLIN26_FLAGSCLEAR	Flags clear register	134_H	P	P	Kernel Reset	4704
ASCLIN26_FLAGSENABLE	Flags enable register	138_H	P	P	Kernel Reset	4708
ASCLIN26_CSR	Clock selection register	$13C_H$	P	P	4712	4712
ASCLIN26_TXDATA	Transmit data register	140_H	P	P	Kernel Reset	4713

(table continues...)

Table 1158 (continued) Register overview - ASCLIN26 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ASCLIN26_TXDATA# (#=0-7)	Transmit data register mirror #	140_H+#*4	P	P	Kernel Reset	4713
ASCLIN26_RXDATA	Receive data register	160_H	P	BE	Kernel Reset	4713
ASCLIN26_RXDATA# (#=0-7)	Receive data register mirror #	160_H+#*4	P	BE	Kernel Reset	4713
ASCLIN26_RXDATAD	Receive data debug register	180_H	P	BE	Kernel Reset	4714

24.8.4.30 Register overview - ASCLIN27 (ascending offset address)

Table 1159 Register overview - ASCLIN27 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ASCLIN27_CLC	Clock control register	000_H	P	P, SV, E	4670	4670
ASCLIN27_OCS	OCDS control and status register	004_H	P	SV, P	Debug Reset	4671
ASCLIN27_ID	Module identification register	008_H	P	BE	PowerOn Reset	4672
ASCLIN27_RST_CTRLA	Reset control register A	$00C_H$	P	P, SV, E	Application Reset	4673
ASCLIN27_RST_CTRLB	Reset control register B	010_H	P	P, SV, E	Application Reset	4673
ASCLIN27_RST_STAT	Reset status register	014_H	P	BE	Application Reset	4674
ASCLIN27_PROTE	PROT register endinit	020_H	U	SV, PROT	Application Reset	4675
ASCLIN27_PROTSE	PROT register safe endinit	024_H	U	SV, PROT	Application Reset	4676
ASCLIN27_ACCEN_WRA	Write access enable register A	040_H	U	SE, SV	Application Reset	4678
ASCLIN27_ACCEN_WRB	Write access enable register B	044_H	U	SE, SV	Application Reset	4678
ASCLIN27_ACCEN_RDA	Read access enable register A	048_H	U	SE, SV	Application Reset	4679
ASCLIN27_ACCEN_RDB	Read access enable register B	$04C_H$	U	SE, SV	Application Reset	4679
ASCLIN27_ACCEN_VM	VM access enable register	050_H	U	SE, SV	Application Reset	4680
ASCLIN27_ACCEN_PRS	PRS access enable register	054_H	U	SE, SV	Application Reset	4680

(table continues...)

Table 1159 (continued) Register overview - ASCLIN27 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ASCLIN27_IOCR	Input and output control register	100_H	P	P	4681	4681
ASCLIN27_TXFIFOCON	TxFIFO configuration register	104_H	P	P	4683	4683
ASCLIN27_RXFIFOCON	RxFIFO configuration register	108_H	P	P	4685	4685
ASCLIN27_BITCON	Bit configuration register	$10C_H$	P	P	4687	4687
ASCLIN27_FRAMECON	Frame control register	110_H	P	P	4689	4689
ASCLIN27_DATCON	Data configuration register	114_H	P	P	Kernel Reset	4691
ASCLIN27_BRG	Baud rate generation register	118_H	P	P	4692	4692
ASCLIN27_BRD	Baud rate detection register	$11C_H$	P	P	Kernel Reset	4693
ASCLIN27_LINCON	LIN control register	120_H	P	P	Kernel Reset	4694
ASCLIN27_LINBTIMER	LIN break timer register	124_H	P	P	Kernel Reset	4695
ASCLIN27_LINHTIMER	LIN header timer register	128_H	P	P	Kernel Reset	4696
ASCLIN27_FLAGS	Flags register	$12C_H$	P	BE	Kernel Reset	4696
ASCLIN27_FLAGSSET	Flags set register	130_H	P	P	Kernel Reset	4701
ASCLIN27_FLAGSCLEAR	Flags clear register	134_H	P	P	Kernel Reset	4704
ASCLIN27_FLAGSENABLE	Flags enable register	138_H	P	P	Kernel Reset	4708
ASCLIN27_CSR	Clock selection register	$13C_H$	P	P	4712	4712
ASCLIN27_TXDATA	Transmit data register	140_H	P	P	Kernel Reset	4713
ASCLIN27_TXDATA# (#=0-7)	Transmit data register mirror #	140_H+#*4	P	P	Kernel Reset	4713
ASCLIN27_RXDATA	Receive data register	160_H	P	BE	Kernel Reset	4713
ASCLIN27_RXDATA# (#=0-7)	Receive data register mirror #	160_H+#*4	P	BE	Kernel Reset	4713
ASCLIN27_RXDATAD	Receive data debug register	180_H	P	BE	Kernel Reset	4714

24.8.4.31 Device specific registers

There are no device specific register changes.

24.8.5 TC4Dx ASCLIN connectivity

The table below lists interfaces to and from this functional block to other blocks in the device.

Table 1160 List of ASCLIN interface signals

Interface signals	I/O	Description
CLOCK_ASCLIN_fSPB	In	SPB clock input
CLOCK_ASCLIN_fASCLINF	In	ASCLINF clock input
CLOCK_ASCLIN_fASCLINS	In	ASCLINS clock input
FPI_ASCLIN_SIF	In	ASCLIN SPB slave interface
ASCLIN0_IR_RX_INT	Out	ASCLIN0 Receive Service Request
ASCLIN1_IR_RX_INT	Out	ASCLIN1 Receive Service Request
ASCLIN2_IR_RX_INT	Out	ASCLIN2 Receive Service Request
ASCLIN3_IR_RX_INT	Out	ASCLIN3 Receive Service Request
ASCLIN4_IR_RX_INT	Out	ASCLIN4 Receive Service Request
ASCLIN5_IR_RX_INT	Out	ASCLIN5 Receive Service Request
ASCLIN6_IR_RX_INT	Out	ASCLIN6 Receive Service Request
ASCLIN7_IR_RX_INT	Out	ASCLIN7 Receive Service Request
ASCLIN8_IR_RX_INT	Out	ASCLIN8 Receive Service Request
ASCLIN9_IR_RX_INT	Out	ASCLIN9 Receive Service Request
ASCLIN10_IR_RX_INT	Out	ASCLIN10 Receive Service Request
ASCLIN11_IR_RX_INT	Out	ASCLIN11 Receive Service Request
ASCLIN12_IR_RX_INT	Out	ASCLIN12 Receive Service Request
ASCLIN13_IR_RX_INT	Out	ASCLIN13 Receive Service Request
ASCLIN14_IR_RX_INT	Out	ASCLIN14 Receive Service Request
ASCLIN15_IR_RX_INT	Out	ASCLIN15 Receive Service Request
ASCLIN16_IR_RX_INT	Out	ASCLIN16 Receive Service Request
ASCLIN17_IR_RX_INT	Out	ASCLIN17 Receive Service Request
ASCLIN18_IR_RX_INT	Out	ASCLIN18 Receive Service Request
ASCLIN19_IR_RX_INT	Out	ASCLIN19 Receive Service Request
ASCLIN20_IR_RX_INT	Out	ASCLIN20 Receive Service Request
ASCLIN21_IR_RX_INT	Out	ASCLIN21 Receive Service Request
ASCLIN22_IR_RX_INT	Out	ASCLIN22 Receive Service Request
ASCLIN23_IR_RX_INT	Out	ASCLIN23 Receive Service Request
ASCLIN24_IR_RX_INT	Out	ASCLIN24 Receive Service Request

(table continues...)

Table 1160 (continued) List of ASCLIN interface signals

Interface signals	I/O	Description
ASCLIN25_IR_RX_INT	Out	ASCLIN25 Receive Service Request
ASCLIN26_IR_RX_INT	Out	ASCLIN26 Receive Service Request
ASCLIN27_IR_RX_INT	Out	ASCLIN27 Receive Service Request
ASCLIN0_IR_TX_INT	Out	ASCLIN0 Transmit Service Request
ASCLIN1_IR_TX_INT	Out	ASCLIN1 Transmit Service Request
ASCLIN2_IR_TX_INT	Out	ASCLIN2 Transmit Service Request
ASCLIN3_IR_TX_INT	Out	ASCLIN3 Transmit Service Request
ASCLIN4_IR_TX_INT	Out	ASCLIN4 Transmit Service Request
ASCLIN5_IR_TX_INT	Out	ASCLIN5 Transmit Service Request
ASCLIN6_IR_TX_INT	Out	ASCLIN6 Transmit Service Request
ASCLIN7_IR_TX_INT	Out	ASCLIN7 Transmit Service Request
ASCLIN8_IR_TX_INT	Out	ASCLIN8 Transmit Service Request
ASCLIN9_IR_TX_INT	Out	ASCLIN9 Transmit Service Request
ASCLIN10_IR_TX_INT	Out	ASCLIN10 Transmit Service Request
ASCLIN11_IR_TX_INT	Out	ASCLIN11 Transmit Service Request
ASCLIN12_IR_TX_INT	Out	ASCLIN12 Transmit Service Request
ASCLIN13_IR_TX_INT	Out	ASCLIN13 Transmit Service Request
ASCLIN14_IR_TX_INT	Out	ASCLIN14 Transmit Service Request
ASCLIN15_IR_TX_INT	Out	ASCLIN15 Transmit Service Request
ASCLIN16_IR_TX_INT	Out	ASCLIN16 Transmit Service Request
ASCLIN17_IR_TX_INT	Out	ASCLIN17 Transmit Service Request
ASCLIN18_IR_TX_INT	Out	ASCLIN18 Transmit Service Request
ASCLIN19_IR_TX_INT	Out	ASCLIN19 Transmit Service Request
ASCLIN20_IR_TX_INT	Out	ASCLIN20 Transmit Service Request
ASCLIN21_IR_TX_INT	Out	ASCLIN21 Transmit Service Request
ASCLIN22_IR_TX_INT	Out	ASCLIN22 Transmit Service Request
ASCLIN23_IR_TX_INT	Out	ASCLIN23 Transmit Service Request
ASCLIN24_IR_TX_INT	Out	ASCLIN24 Transmit Service Request
ASCLIN25_IR_TX_INT	Out	ASCLIN25 Transmit Service Request
ASCLIN26_IR_TX_INT	Out	ASCLIN26 Transmit Service Request
ASCLIN27_IR_TX_INT	Out	ASCLIN27 Transmit Service Request
ASCLIN0_IR_ERR_INT	Out	ASCLIN0 Error Service Request
ASCLIN1_IR_ERR_INT	Out	ASCLIN1 Error Service Request

(table continues...)

Table 1160 (continued) List of ASCLIN interface signals

Interface signals	I/O	Description
ASCLIN2_IR_ERR_INT	Out	ASCLIN2 Error Service Request
ASCLIN3_IR_ERR_INT	Out	ASCLIN3 Error Service Request
ASCLIN4_IR_ERR_INT	Out	ASCLIN4 Error Service Request
ASCLIN5_IR_ERR_INT	Out	ASCLIN5 Error Service Request
ASCLIN6_IR_ERR_INT	Out	ASCLIN6 Error Service Request
ASCLIN7_IR_ERR_INT	Out	ASCLIN7 Error Service Request
ASCLIN8_IR_ERR_INT	Out	ASCLIN8 Error Service Request
ASCLIN9_IR_ERR_INT	Out	ASCLIN9 Error Service Request
ASCLIN10_IR_ERR_INT	Out	ASCLIN10 Error Service Request
ASCLIN11_IR_ERR_INT	Out	ASCLIN11 Error Service Request
ASCLIN12_IR_ERR_INT	Out	ASCLIN12 Error Service Request
ASCLIN13_IR_ERR_INT	Out	ASCLIN13 Error Service Request
ASCLIN14_IR_ERR_INT	Out	ASCLIN14 Error Service Request
ASCLIN15_IR_ERR_INT	Out	ASCLIN15 Error Service Request
ASCLIN16_IR_ERR_INT	Out	ASCLIN16 Error Service Request
ASCLIN17_IR_ERR_INT	Out	ASCLIN17 Error Service Request
ASCLIN18_IR_ERR_INT	Out	ASCLIN18 Error Service Request
ASCLIN19_IR_ERR_INT	Out	ASCLIN19 Error Service Request
ASCLIN20_IR_ERR_INT	Out	ASCLIN20 Error Service Request
ASCLIN21_IR_ERR_INT	Out	ASCLIN21 Error Service Request
ASCLIN22_IR_ERR_INT	Out	ASCLIN22 Error Service Request
ASCLIN23_IR_ERR_INT	Out	ASCLIN23 Error Service Request
ASCLIN24_IR_ERR_INT	Out	ASCLIN24 Error Service Request
ASCLIN25_IR_ERR_INT	Out	ASCLIN25 Error Service Request
ASCLIN26_IR_ERR_INT	Out	ASCLIN26 Error Service Request
ASCLIN27_IR_ERR_INT	Out	ASCLIN27 Error Service Request
EGTM_ASCLIN0_TRIG	In	eGTM Trigger Input
EGTM_ASCLIN1_TRIG	In	eGTM Trigger Input
EGTM_ASCLIN2_TRIG	In	eGTM Trigger Input
EGTM_ASCLIN3_TRIG	In	eGTM Trigger Input
EGTM_ASCLIN4_TRIG	In	eGTM Trigger Input
EGTM_ASCLIN5_TRIG	In	eGTM Trigger Input
EGTM_ASCLIN6_TRIG	In	eGTM Trigger Input

(table continues...)

Table 1160 (continued) List of ASCLIN interface signals

Interface signals	I/O	Description
EGTM_ASCLIN7_TRIG	In	eGTM Trigger Input
EGTM_ASCLIN8_TRIG	In	eGTM Trigger Input
EGTM_ASCLIN9_TRIG	In	eGTM Trigger Input
EGTM_ASCLIN10_TRIG	In	eGTM Trigger Input
EGTM_ASCLIN11_TRIG	In	eGTM Trigger Input
EGTM_ASCLIN12_TRIG	In	eGTM Trigger Input
EGTM_ASCLIN13_TRIG	In	eGTM Trigger Input
EGTM_ASCLIN14_TRIG	In	eGTM Trigger Input
EGTM_ASCLIN15_TRIG	In	eGTM Trigger Input
EGTM_ASCLIN16_TRIG	In	eGTM Trigger Input
EGTM_ASCLIN17_TRIG	In	eGTM Trigger Input
EGTM_ASCLIN18_TRIG	In	eGTM Trigger Input
EGTM_ASCLIN19_TRIG	In	eGTM Trigger Input
EGTM_ASCLIN20_TRIG	In	eGTM Trigger Input
EGTM_ASCLIN21_TRIG	In	eGTM Trigger Input
EGTM_ASCLIN22_TRIG	In	eGTM Trigger Input
EGTM_ASCLIN23_TRIG	In	eGTM Trigger Input
EGTM_ASCLIN24_TRIG	In	eGTM Trigger Input
EGTM_ASCLIN25_TRIG	In	eGTM Trigger Input
EGTM_ASCLIN26_TRIG	In	eGTM Trigger Input
EGTM_ASCLIN27_TRIG	In	eGTM Trigger Input
PORTS_ASCLIN0_ARXA	In	Receive Input
PORTS_ASCLIN1_ARXA	In	Receive Input
PORTS_ASCLIN2_ARXA	In	Receive Input
PORTS_ASCLIN3_ARXA	In	Receive Input
PORTS_ASCLIN4_ARXA	In	Receive Input
PORTS_ASCLIN5_ARXA	In	Receive Input
PORTS_ASCLIN6_ARXA	In	Receive Input
PORTS_ASCLIN7_ARXA	In	Receive Input
PORTS_ASCLIN8_ARXA	In	Receive Input
PORTS_ASCLIN9_ARXA	In	Receive Input
PORTS_ASCLIN10_ARXA	In	Receive Input
PORTS_ASCLIN11_ARXA	In	Receive Input

(table continues...)

Table 1160 (continued) List of ASCLIN interface signals

Interface signals	I/O	Description
PORTS_ASCLIN12_ARXA	In	Receive Input
PORTS_ASCLIN13_ARXA	In	Receive Input
PORTS_ASCLIN14_ARXA	In	Receive Input
PORTS_ASCLIN15_ARXA	In	Receive Input
PORTS_ASCLIN16_ARXA	In	Receive Input
PORTS_ASCLIN17_ARXA	In	Receive Input
PORTS_ASCLIN18_ARXA	In	Receive Input
PORTS_ASCLIN19_ARXA	In	Receive Input
PORTS_ASCLIN20_ARXA	In	Receive Input
PORTS_ASCLIN21_ARXA	In	Receive Input
PORTS_ASCLIN22_ARXA	In	Receive Input
PORTS_ASCLIN23_ARXA	In	Receive Input
PORTS_ASCLIN24_ARXA	In	Receive Input
PORTS_ASCLIN25_ARXA	In	Receive Input
PORTS_ASCLIN26_ARXA	In	Receive Input
PORTS_ASCLIN27_ARXA	In	Receive Input
PORTS_ASCLIN0_ARXB	In	Receive Input
PORTS_ASCLIN1_ARXB	In	Receive Input
PORTS_ASCLIN2_ARXB	In	Receive Input
PORTS_ASCLIN3_ARXB	In	Receive Input
PORTS_ASCLIN4_ARXB	In	Receive Input
PORTS_ASCLIN5_ARXB	In	Receive Input
PORTS_ASCLIN6_ARXB	In	Receive Input
PORTS_ASCLIN7_ARXB	In	Receive Input
PORTS_ASCLIN8_ARXB	In	Receive Input
PORTS_ASCLIN9_ARXB	In	Receive Input
PORTS_ASCLIN10_ARXB	In	Receive Input
PORTS_ASCLIN11_ARXB	In	Receive Input
PORTS_ASCLIN12_ARXB	In	Receive Input
PORTS_ASCLIN13_ARXB	In	Receive Input
PORTS_ASCLIN14_ARXB	In	Receive Input
PORTS_ASCLIN15_ARXB	In	Receive Input
PORTS_ASCLIN16_ARXB	In	Receive Input

(table continues...)

Table 1160 (continued) List of ASCLIN interface signals

Interface signals	I/O	Description
PORTS_ASCLIN17_ARXB	In	Receive Input
PORTS_ASCLIN18_ARXB	In	Receive Input
PORTS_ASCLIN19_ARXB	In	Receive Input
PORTS_ASCLIN20_ARXB	In	Receive Input
PORTS_ASCLIN21_ARXB	In	Receive Input
PORTS_ASCLIN22_ARXB	In	Receive Input
PORTS_ASCLIN23_ARXB	In	Receive Input
PORTS_ASCLIN24_ARXB	In	Receive Input
PORTS_ASCLIN25_ARXB	In	Receive Input
PORTS_ASCLIN26_ARXB	In	Receive Input
PORTS_ASCLIN27_ARXB	In	Receive Input
PORTS_ASCLIN0_ARXC	In	Receive Input
PORTS_ASCLIN1_ARXC	In	Receive Input
PORTS_ASCLIN2_ARXC	In	Receive Input
PORTS_ASCLIN3_ARXC	In	Receive Input
PORTS_ASCLIN4_ARXC	In	Receive Input
PORTS_ASCLIN5_ARXC	In	Receive Input
PORTS_ASCLIN6_ARXC	In	Receive Input
PORTS_ASCLIN7_ARXC	In	Receive Input
PORTS_ASCLIN8_ARXC	In	Receive Input
PORTS_ASCLIN9_ARXC	In	Receive Input
PORTS_ASCLIN10_ARXC	In	Receive Input
PORTS_ASCLIN11_ARXC	In	Receive Input
PORTS_ASCLIN12_ARXC	In	Receive Input
PORTS_ASCLIN13_ARXC	In	Receive Input
PORTS_ASCLIN14_ARXC	In	Receive Input
PORTS_ASCLIN15_ARXC	In	Receive Input
PORTS_ASCLIN16_ARXC	In	Receive Input
PORTS_ASCLIN17_ARXC	In	Receive Input
PORTS_ASCLIN18_ARXC	In	Receive Input
PORTS_ASCLIN19_ARXC	In	Receive Input
PORTS_ASCLIN20_ARXC	In	Receive Input
PORTS_ASCLIN21_ARXC	In	Receive Input

(table continues...)

Table 1160 (continued) List of ASCLIN interface signals

Interface signals	I/O	Description
PORTS_ASCLIN22_ARXC	In	Receive Input
PORTS_ASCLIN23_ARXC	In	Receive Input
PORTS_ASCLIN24_ARXC	In	Receive Input
PORTS_ASCLIN25_ARXC	In	Receive Input
PORTS_ASCLIN26_ARXC	In	Receive Input
PORTS_ASCLIN27_ARXC	In	Receive Input
PORTS_ASCLIN0_ARXD	In	Receive Input
PORTS_ASCLIN1_ARXD	In	Receive Input
PORTS_ASCLIN2_ARXD	In	Receive Input
PORTS_ASCLIN3_ARXD	In	Receive Input
PORTS_ASCLIN4_ARXD	In	Receive Input
PORTS_ASCLIN5_ARXD	In	Receive Input
PORTS_ASCLIN6_ARXD	In	Receive Input
PORTS_ASCLIN7_ARXD	In	Receive Input
PORTS_ASCLIN8_ARXD	In	Receive Input
PORTS_ASCLIN9_ARXD	In	Receive Input
PORTS_ASCLIN10_ARXD	In	Receive Input
PORTS_ASCLIN11_ARXD	In	Receive Input
PORTS_ASCLIN12_ARXD	In	Receive Input
PORTS_ASCLIN13_ARXD	In	Receive Input
PORTS_ASCLIN14_ARXD	In	Receive Input
PORTS_ASCLIN15_ARXD	In	Receive Input
PORTS_ASCLIN16_ARXD	In	Receive Input
PORTS_ASCLIN17_ARXD	In	Receive Input
PORTS_ASCLIN18_ARXD	In	Receive Input
PORTS_ASCLIN19_ARXD	In	Receive Input
PORTS_ASCLIN20_ARXD	In	Receive Input
PORTS_ASCLIN21_ARXD	In	Receive Input
PORTS_ASCLIN22_ARXD	In	Receive Input
PORTS_ASCLIN23_ARXD	In	Receive Input
PORTS_ASCLIN24_ARXD	In	Receive Input
PORTS_ASCLIN25_ARXD	In	Receive Input
PORTS_ASCLIN26_ARXD	In	Receive Input

(table continues...)

Table 1160 (continued) List of ASCLIN interface signals

Interface signals	I/O	Description
PORTS_ASCLIN27_ARXD	In	Receive Input
PORTS_ASCLIN0_ARXE	In	Receive Input
PORTS_ASCLIN1_ARXE	In	Receive Input
PORTS_ASCLIN2_ARXE	In	Receive Input
PORTS_ASCLIN3_ARXE	In	Receive Input
PORTS_ASCLIN4_ARXE	In	Receive Input
PORTS_ASCLIN5_ARXE	In	Receive Input
PORTS_ASCLIN6_ARXE	In	Receive Input
PORTS_ASCLIN7_ARXE	In	Receive Input
PORTS_ASCLIN8_ARXE	In	Receive Input
PORTS_ASCLIN9_ARXE	In	Receive Input
PORTS_ASCLIN10_ARXE	In	Receive Input
PORTS_ASCLIN11_ARXE	In	Receive Input
PORTS_ASCLIN12_ARXE	In	Receive Input
PORTS_ASCLIN13_ARXE	In	Receive Input
PORTS_ASCLIN14_ARXE	In	Receive Input
PORTS_ASCLIN15_ARXE	In	Receive Input
PORTS_ASCLIN16_ARXE	In	Receive Input
PORTS_ASCLIN17_ARXE	In	Receive Input
PORTS_ASCLIN18_ARXE	In	Receive Input
PORTS_ASCLIN19_ARXE	In	Receive Input
PORTS_ASCLIN20_ARXE	In	Receive Input
PORTS_ASCLIN21_ARXE	In	Receive Input
PORTS_ASCLIN22_ARXE	In	Receive Input
PORTS_ASCLIN23_ARXE	In	Receive Input
PORTS_ASCLIN24_ARXE	In	Receive Input
PORTS_ASCLIN25_ARXE	In	Receive Input
PORTS_ASCLIN26_ARXE	In	Receive Input
PORTS_ASCLIN27_ARXE	In	Receive Input
PORTS_ASCLIN0_ARXF	In	Receive Input
PORTS_ASCLIN1_ARXF	In	Receive Input
PORTS_ASCLIN2_ARXF	In	Receive Input
PORTS_ASCLIN3_ARXF	In	Receive Input

(table continues...)

Table 1160 (continued) List of ASCLIN interface signals

Interface signals	I/O	Description
PORTS_ASCLIN4_ARXF	In	Receive Input
PORTS_ASCLIN5_ARXF	In	Receive Input
PORTS_ASCLIN6_ARXF	In	Receive Input
PORTS_ASCLIN7_ARXF	In	Receive Input
PORTS_ASCLIN8_ARXF	In	Receive Input
PORTS_ASCLIN9_ARXF	In	Receive Input
PORTS_ASCLIN10_ARXF	In	Receive Input
PORTS_ASCLIN11_ARXF	In	Receive Input
PORTS_ASCLIN12_ARXF	In	Receive Input
PORTS_ASCLIN13_ARXF	In	Receive Input
PORTS_ASCLIN14_ARXF	In	Receive Input
PORTS_ASCLIN15_ARXF	In	Receive Input
PORTS_ASCLIN16_ARXF	In	Receive Input
PORTS_ASCLIN17_ARXF	In	Receive Input
PORTS_ASCLIN18_ARXF	In	Receive Input
PORTS_ASCLIN19_ARXF	In	Receive Input
PORTS_ASCLIN20_ARXF	In	Receive Input
PORTS_ASCLIN21_ARXF	In	Receive Input
PORTS_ASCLIN22_ARXF	In	Receive Input
PORTS_ASCLIN23_ARXF	In	Receive Input
PORTS_ASCLIN24_ARXF	In	Receive Input
PORTS_ASCLIN25_ARXF	In	Receive Input
PORTS_ASCLIN26_ARXF	In	Receive Input
PORTS_ASCLIN27_ARXF	In	Receive Input
PORTS_ASCLIN0_ARXG	In	Receive Input
PORTS_ASCLIN1_ARXG	In	Receive Input
PORTS_ASCLIN2_ARXG	In	Receive Input
PORTS_ASCLIN3_ARXG	In	Receive Input
PORTS_ASCLIN4_ARXG	In	Receive Input
PORTS_ASCLIN5_ARXG	In	Receive Input
PORTS_ASCLIN6_ARXG	In	Receive Input
PORTS_ASCLIN7_ARXG	In	Receive Input
PORTS_ASCLIN8_ARXG	In	Receive Input

(table continues...)

Table 1160 (continued) List of ASCLIN interface signals

Interface signals	I/O	Description
PORTS_ASCLIN9_ARXG	In	Receive Input
PORTS_ASCLIN10_ARXG	In	Receive Input
PORTS_ASCLIN11_ARXG	In	Receive Input
PORTS_ASCLIN12_ARXG	In	Receive Input
PORTS_ASCLIN13_ARXG	In	Receive Input
PORTS_ASCLIN14_ARXG	In	Receive Input
PORTS_ASCLIN15_ARXG	In	Receive Input
PORTS_ASCLIN16_ARXG	In	Receive Input
PORTS_ASCLIN17_ARXG	In	Receive Input
PORTS_ASCLIN18_ARXG	In	Receive Input
PORTS_ASCLIN19_ARXG	In	Receive Input
PORTS_ASCLIN20_ARXG	In	Receive Input
PORTS_ASCLIN21_ARXG	In	Receive Input
PORTS_ASCLIN22_ARXG	In	Receive Input
PORTS_ASCLIN23_ARXG	In	Receive Input
PORTS_ASCLIN24_ARXG	In	Receive Input
PORTS_ASCLIN25_ARXG	In	Receive Input
PORTS_ASCLIN26_ARXG	In	Receive Input
PORTS_ASCLIN27_ARXG	In	Receive Input
PORTS_ASCLIN0_ARXH	In	Receive Input
PORTS_ASCLIN1_ARXH	In	Receive Input
PORTS_ASCLIN2_ARXH	In	Receive Input
PORTS_ASCLIN3_ARXH	In	Receive Input
PORTS_ASCLIN4_ARXH	In	Receive Input
PORTS_ASCLIN5_ARXH	In	Receive Input
PORTS_ASCLIN6_ARXH	In	Receive Input
PORTS_ASCLIN7_ARXH	In	Receive Input
PORTS_ASCLIN8_ARXH	In	Receive Input
PORTS_ASCLIN9_ARXH	In	Receive Input
PORTS_ASCLIN10_ARXH	In	Receive Input
PORTS_ASCLIN11_ARXH	In	Receive Input
PORTS_ASCLIN12_ARXH	In	Receive Input
PORTS_ASCLIN13_ARXH	In	Receive Input

(table continues...)

Table 1160 (continued) List of ASCLIN interface signals

Interface signals	I/O	Description
PORTS_ASCLIN14_ARXH	In	Receive Input
PORTS_ASCLIN15_ARXH	In	Receive Input
PORTS_ASCLIN16_ARXH	In	Receive Input
PORTS_ASCLIN17_ARXH	In	Receive Input
PORTS_ASCLIN18_ARXH	In	Receive Input
PORTS_ASCLIN19_ARXH	In	Receive Input
PORTS_ASCLIN20_ARXH	In	Receive Input
PORTS_ASCLIN21_ARXH	In	Receive Input
PORTS_ASCLIN22_ARXH	In	Receive Input
PORTS_ASCLIN23_ARXH	In	Receive Input
PORTS_ASCLIN24_ARXH	In	Receive Input
PORTS_ASCLIN25_ARXH	In	Receive Input
PORTS_ASCLIN26_ARXH	In	Receive Input
PORTS_ASCLIN27_ARXH	In	Receive Input
PORTS_ASCLIN0_ACTSA	In	Clear to Send Input
PORTS_ASCLIN1_ACTSA	In	Clear to Send Input
PORTS_ASCLIN2_ACTSA	In	Clear to Send Input
PORTS_ASCLIN3_ACTSA	In	Clear to Send Input
PORTS_ASCLIN4_ACTSA	In	Clear to Send Input
PORTS_ASCLIN5_ACTSA	In	Clear to Send Input
PORTS_ASCLIN6_ACTSA	In	Clear to Send Input
PORTS_ASCLIN7_ACTSA	In	Clear to Send Input
PORTS_ASCLIN8_ACTSA	In	Clear to Send Input
PORTS_ASCLIN9_ACTSA	In	Clear to Send Input
PORTS_ASCLIN10_ACTSA	In	Clear to Send Input
PORTS_ASCLIN11_ACTSA	In	Clear to Send Input
PORTS_ASCLIN12_ACTSA	In	Clear to Send Input
PORTS_ASCLIN13_ACTSA	In	Clear to Send Input
PORTS_ASCLIN14_ACTSA	In	Clear to Send Input
PORTS_ASCLIN15_ACTSA	In	Clear to Send Input
PORTS_ASCLIN16_ACTSA	In	Clear to Send Input
PORTS_ASCLIN17_ACTSA	In	Clear to Send Input
PORTS_ASCLIN18_ACTSA	In	Clear to Send Input

(table continues...)

Table 1160 (continued) List of ASCLIN interface signals

Interface signals	I/O	Description
PORTS_ASCLIN19_ACTSA	In	Clear to Send Input
PORTS_ASCLIN20_ACTSA	In	Clear to Send Input
PORTS_ASCLIN21_ACTSA	In	Clear to Send Input
PORTS_ASCLIN22_ACTSA	In	Clear to Send Input
PORTS_ASCLIN23_ACTSA	In	Clear to Send Input
PORTS_ASCLIN24_ACTSA	In	Clear to Send Input
PORTS_ASCLIN25_ACTSA	In	Clear to Send Input
PORTS_ASCLIN26_ACTSA	In	Clear to Send Input
PORTS_ASCLIN27_ACTSA	In	Clear to Send Input
PORTS_ASCLIN0_ACTSB	In	Clear to Send Input
PORTS_ASCLIN1_ACTSB	In	Clear to Send Input
PORTS_ASCLIN2_ACTSB	In	Clear to Send Input
PORTS_ASCLIN3_ACTSB	In	Clear to Send Input
PORTS_ASCLIN4_ACTSB	In	Clear to Send Input
PORTS_ASCLIN5_ACTSB	In	Clear to Send Input
PORTS_ASCLIN6_ACTSB	In	Clear to Send Input
PORTS_ASCLIN7_ACTSB	In	Clear to Send Input
PORTS_ASCLIN8_ACTSB	In	Clear to Send Input
PORTS_ASCLIN9_ACTSB	In	Clear to Send Input
PORTS_ASCLIN10_ACTSB	In	Clear to Send Input
PORTS_ASCLIN11_ACTSB	In	Clear to Send Input
PORTS_ASCLIN12_ACTSB	In	Clear to Send Input
PORTS_ASCLIN13_ACTSB	In	Clear to Send Input
PORTS_ASCLIN14_ACTSB	In	Clear to Send Input
PORTS_ASCLIN15_ACTSB	In	Clear to Send Input
PORTS_ASCLIN16_ACTSB	In	Clear to Send Input
PORTS_ASCLIN17_ACTSB	In	Clear to Send Input
PORTS_ASCLIN18_ACTSB	In	Clear to Send Input
PORTS_ASCLIN19_ACTSB	In	Clear to Send Input
PORTS_ASCLIN20_ACTSB	In	Clear to Send Input
PORTS_ASCLIN21_ACTSB	In	Clear to Send Input
PORTS_ASCLIN22_ACTSB	In	Clear to Send Input
PORTS_ASCLIN23_ACTSB	In	Clear to Send Input

(table continues...)

Table 1160 (continued) List of ASCLIN interface signals

Interface signals	I/O	Description
PORTS_ASCLIN24_ACTSB	In	Clear to Send Input
PORTS_ASCLIN25_ACTSB	In	Clear to Send Input
PORTS_ASCLIN26_ACTSB	In	Clear to Send Input
PORTS_ASCLIN27_ACTSB	In	Clear to Send Input
PORTS_ASCLIN0_ACTSC	In	Clear to Send Input
PORTS_ASCLIN1_ACTSC	In	Clear to Send Input
PORTS_ASCLIN2_ACTSC	In	Clear to Send Input
PORTS_ASCLIN3_ACTSC	In	Clear to Send Input
PORTS_ASCLIN4_ACTSC	In	Clear to Send Input
PORTS_ASCLIN5_ACTSC	In	Clear to Send Input
PORTS_ASCLIN6_ACTSC	In	Clear to Send Input
PORTS_ASCLIN7_ACTSC	In	Clear to Send Input
PORTS_ASCLIN8_ACTSC	In	Clear to Send Input
PORTS_ASCLIN9_ACTSC	In	Clear to Send Input
PORTS_ASCLIN10_ACTSC	In	Clear to Send Input
PORTS_ASCLIN11_ACTSC	In	Clear to Send Input
PORTS_ASCLIN12_ACTSC	In	Clear to Send Input
PORTS_ASCLIN13_ACTSC	In	Clear to Send Input
PORTS_ASCLIN14_ACTSC	In	Clear to Send Input
PORTS_ASCLIN15_ACTSC	In	Clear to Send Input
PORTS_ASCLIN16_ACTSC	In	Clear to Send Input
PORTS_ASCLIN17_ACTSC	In	Clear to Send Input
PORTS_ASCLIN18_ACTSC	In	Clear to Send Input
PORTS_ASCLIN19_ACTSC	In	Clear to Send Input
PORTS_ASCLIN20_ACTSC	In	Clear to Send Input
PORTS_ASCLIN21_ACTSC	In	Clear to Send Input
PORTS_ASCLIN22_ACTSC	In	Clear to Send Input
PORTS_ASCLIN23_ACTSC	In	Clear to Send Input
PORTS_ASCLIN24_ACTSC	In	Clear to Send Input
PORTS_ASCLIN25_ACTSC	In	Clear to Send Input
PORTS_ASCLIN26_ACTSC	In	Clear to Send Input
PORTS_ASCLIN27_ACTSC	In	Clear to Send Input
ASCLIN0_PORTS_ATX	Out	Transmit Output

(table continues...)

Table 1160 (continued) List of ASCLIN interface signals

Interface signals	I/O	Description
ASCLIN1_PORTS_ATX	Out	Transmit Output
ASCLIN2_PORTS_ATX	Out	Transmit Output
ASCLIN3_PORTS_ATX	Out	Transmit Output
ASCLIN4_PORTS_ATX	Out	Transmit Output
ASCLIN5_PORTS_ATX	Out	Transmit Output
ASCLIN6_PORTS_ATX	Out	Transmit Output
ASCLIN7_PORTS_ATX	Out	Transmit Output
ASCLIN8_PORTS_ATX	Out	Transmit Output
ASCLIN9_PORTS_ATX	Out	Transmit Output
ASCLIN10_PORTS_ATX	Out	Transmit Output
ASCLIN11_PORTS_ATX	Out	Transmit Output
ASCLIN12_PORTS_ATX	Out	Transmit Output
ASCLIN13_PORTS_ATX	Out	Transmit Output
ASCLIN14_PORTS_ATX	Out	Transmit Output
ASCLIN15_PORTS_ATX	Out	Transmit Output
ASCLIN16_PORTS_ATX	Out	Transmit Output
ASCLIN17_PORTS_ATX	Out	Transmit Output
ASCLIN18_PORTS_ATX	Out	Transmit Output
ASCLIN19_PORTS_ATX	Out	Transmit Output
ASCLIN20_PORTS_ATX	Out	Transmit Output
ASCLIN21_PORTS_ATX	Out	Transmit Output
ASCLIN22_PORTS_ATX	Out	Transmit Output
ASCLIN23_PORTS_ATX	Out	Transmit Output
ASCLIN24_PORTS_ATX	Out	Transmit Output
ASCLIN25_PORTS_ATX	Out	Transmit Output
ASCLIN26_PORTS_ATX	Out	Transmit Output
ASCLIN27_PORTS_ATX	Out	Transmit Output
ASCLIN0_PORTS_ARTS	Out	Ready to Send Output
ASCLIN1_PORTS_ARTS	Out	Ready to Send Output
ASCLIN2_PORTS_ARTS	Out	Ready to Send Output
ASCLIN3_PORTS_ARTS	Out	Ready to Send Output
ASCLIN4_PORTS_ARTS	Out	Ready to Send Output
ASCLIN5_PORTS_ARTS	Out	Ready to Send Output

(table continues...)

Table 1160 (continued) List of ASCLIN interface signals

Interface signals	I/O	Description
ASCLIN6_PORTS_ARTS	Out	Ready to Send Output
ASCLIN7_PORTS_ARTS	Out	Ready to Send Output
ASCLIN8_PORTS_ARTS	Out	Ready to Send Output
ASCLIN9_PORTS_ARTS	Out	Ready to Send Output
ASCLIN10_PORTS_ARTS	Out	Ready to Send Output
ASCLIN11_PORTS_ARTS	Out	Ready to Send Output
ASCLIN12_PORTS_ARTS	Out	Ready to Send Output
ASCLIN13_PORTS_ARTS	Out	Ready to Send Output
ASCLIN14_PORTS_ARTS	Out	Ready to Send Output
ASCLIN15_PORTS_ARTS	Out	Ready to Send Output
ASCLIN16_PORTS_ARTS	Out	Ready to Send Output
ASCLIN17_PORTS_ARTS	Out	Ready to Send Output
ASCLIN18_PORTS_ARTS	Out	Ready to Send Output
ASCLIN19_PORTS_ARTS	Out	Ready to Send Output
ASCLIN20_PORTS_ARTS	Out	Ready to Send Output
ASCLIN21_PORTS_ARTS	Out	Ready to Send Output
ASCLIN22_PORTS_ARTS	Out	Ready to Send Output
ASCLIN23_PORTS_ARTS	Out	Ready to Send Output
ASCLIN24_PORTS_ARTS	Out	Ready to Send Output
ASCLIN25_PORTS_ARTS	Out	Ready to Send Output
ASCLIN26_PORTS_ARTS	Out	Ready to Send Output
ASCLIN27_PORTS_ARTS	Out	Ready to Send Output
ASCLIN0_PORTS_ASLSO	Out	Slave Select Signal Output
ASCLIN1_PORTS_ASLSO	Out	Slave Select Signal Output
ASCLIN2_PORTS_ASLSO	Out	Slave Select Signal Output
ASCLIN3_PORTS_ASLSO	Out	Slave Select Signal Output
ASCLIN4_PORTS_ASLSO	Out	Slave Select Signal Output
ASCLIN5_PORTS_ASLSO	Out	Slave Select Signal Output
ASCLIN6_PORTS_ASLSO	Out	Slave Select Signal Output
ASCLIN7_PORTS_ASLSO	Out	Slave Select Signal Output
ASCLIN8_PORTS_ASLSO	Out	Slave Select Signal Output
ASCLIN9_PORTS_ASLSO	Out	Slave Select Signal Output
ASCLIN10_PORTS_ASLSO	Out	Slave Select Signal Output

(table continues...)

Table 1160 (continued) List of ASCLIN interface signals

Interface signals	I/O	Description
ASCLIN11_PORTS_ASLSO	Out	Slave Select Signal Output
ASCLIN12_PORTS_ASLSO	Out	Slave Select Signal Output
ASCLIN13_PORTS_ASLSO	Out	Slave Select Signal Output
ASCLIN14_PORTS_ASLSO	Out	Slave Select Signal Output
ASCLIN15_PORTS_ASLSO	Out	Slave Select Signal Output
ASCLIN16_PORTS_ASLSO	Out	Slave Select Signal Output
ASCLIN17_PORTS_ASLSO	Out	Slave Select Signal Output
ASCLIN18_PORTS_ASLSO	Out	Slave Select Signal Output
ASCLIN19_PORTS_ASLSO	Out	Slave Select Signal Output
ASCLIN20_PORTS_ASLSO	Out	Slave Select Signal Output
ASCLIN21_PORTS_ASLSO	Out	Slave Select Signal Output
ASCLIN22_PORTS_ASLSO	Out	Slave Select Signal Output
ASCLIN23_PORTS_ASLSO	Out	Slave Select Signal Output
ASCLIN24_PORTS_ASLSO	Out	Slave Select Signal Output
ASCLIN25_PORTS_ASLSO	Out	Slave Select Signal Output
ASCLIN26_PORTS_ASLSO	Out	Slave Select Signal Output
ASCLIN27_PORTS_ASLSO	Out	Slave Select Signal Output
ASCLIN0_PORTS_ASCLK	Out	Shift Clock Output
ASCLIN1_PORTS_ASCLK	Out	Shift Clock Output
ASCLIN2_PORTS_ASCLK	Out	Shift Clock Output
ASCLIN3_PORTS_ASCLK	Out	Shift Clock Output
ASCLIN4_PORTS_ASCLK	Out	Shift Clock Output
ASCLIN5_PORTS_ASCLK	Out	Shift Clock Output
ASCLIN6_PORTS_ASCLK	Out	Shift Clock Output
ASCLIN7_PORTS_ASCLK	Out	Shift Clock Output
ASCLIN8_PORTS_ASCLK	Out	Shift Clock Output
ASCLIN9_PORTS_ASCLK	Out	Shift Clock Output
ASCLIN10_PORTS_ASCLK	Out	Shift Clock Output
ASCLIN11_PORTS_ASCLK	Out	Shift Clock Output
ASCLIN12_PORTS_ASCLK	Out	Shift Clock Output
ASCLIN13_PORTS_ASCLK	Out	Shift Clock Output
ASCLIN14_PORTS_ASCLK	Out	Shift Clock Output
ASCLIN15_PORTS_ASCLK	Out	Shift Clock Output

(table continues...)

Table 1160 (continued) List of ASCLIN interface signals

Interface signals	I/O	Description
ASCLIN16_PORTS_ASCLK	Out	Shift Clock Output
ASCLIN17_PORTS_ASCLK	Out	Shift Clock Output
ASCLIN18_PORTS_ASCLK	Out	Shift Clock Output
ASCLIN19_PORTS_ASCLK	Out	Shift Clock Output
ASCLIN20_PORTS_ASCLK	Out	Shift Clock Output
ASCLIN21_PORTS_ASCLK	Out	Shift Clock Output
ASCLIN22_PORTS_ASCLK	Out	Shift Clock Output
ASCLIN23_PORTS_ASCLK	Out	Shift Clock Output
ASCLIN24_PORTS_ASCLK	Out	Shift Clock Output
ASCLIN25_PORTS_ASCLK	Out	Shift Clock Output
ASCLIN26_PORTS_ASCLK	Out	Shift Clock Output
ASCLIN27_PORTS_ASCLK	Out	Shift Clock Output

24.8.6 TC4Dx ASCLIN revision history

Initial release of the chapter.

25 Single Edge Nibble Transmission (SENT)

The SENT interface provides a serial communication link typically used to connect sensors or other peripheral devices.

25.1 Feature list

- Conformance with SENT specification J2716 201604
- Flexible functional configuration for software
  - Baud rate generation: Data rates of up to 65.8 kbit/s at 3 µs tick length and 6 data nibbles on each channel
  - Support of standard tick times (3 µs through 90 µs)
  - Message tick time programmable between 0.2 µs and 1024 µs
  - Configurable frame length (default is 24-bit) and maximum data size of 32 bits
  - Digital Glitch filter suppressing noise
  - Optional output inversion for use of external open drain transistor
  - Optional input inversion for use of external open transistor for level shifting
- Receive SENT frames from external devices and handle all protocol relevant topics
  - 15 SENT channels per instance implemented, all working independently in parallel
  - Programmable nibble sorting to relieve CPU
  - Transparent mode (nibble CRCs are written to the receive control register for software processing)
  - Status nibble optionally included in the checksum
  - Optional serial data processing
  - Option to enable or disable the check of the next calibration pulse before validation of received data
  - Support frequency drift analysis based on frame length for frames with pause pulse
  - Indication of system status: STOP, INITIALIZED, RUNNING, SYNCHRONIZED
  - Time stamp generation
  - Watchdog on incoming frames
  - Support of trailing pause nibble of any length (even longer than 70 ticks)
- Support of SPC (Short PWM Code)
- Flexible interrupt structure generation available
  - The receiver module will monitor the message for the following error conditions:
    - Calibration pulse length deviates more than +/-25% from the nominal 56 ticks
    - Too many or too few nibbles between calibration pulses
    - Checksum error
    - Successive calibration pulse differ by more than 1.5625%
    - Any nibble data values measured as < 0 or > 15
  - Sticky interrupt flags, error interrupt optional
  - Interrupt on status nibble violation
  - Buffer overrun detection

25.2 Functional overview

The ERAY module communicates with the external world through one IO line for each channel. The SENT Input lines are the receive data input signals. They can overlay ADC inputs. If the optional SPC mode is used, they can be used on a port configured with an open drain transistor. In this way the optional SPC data can be transmitted and the line is used bi-directionally. For an external transceiver, the receive and transmit path can be routed to two different ports.

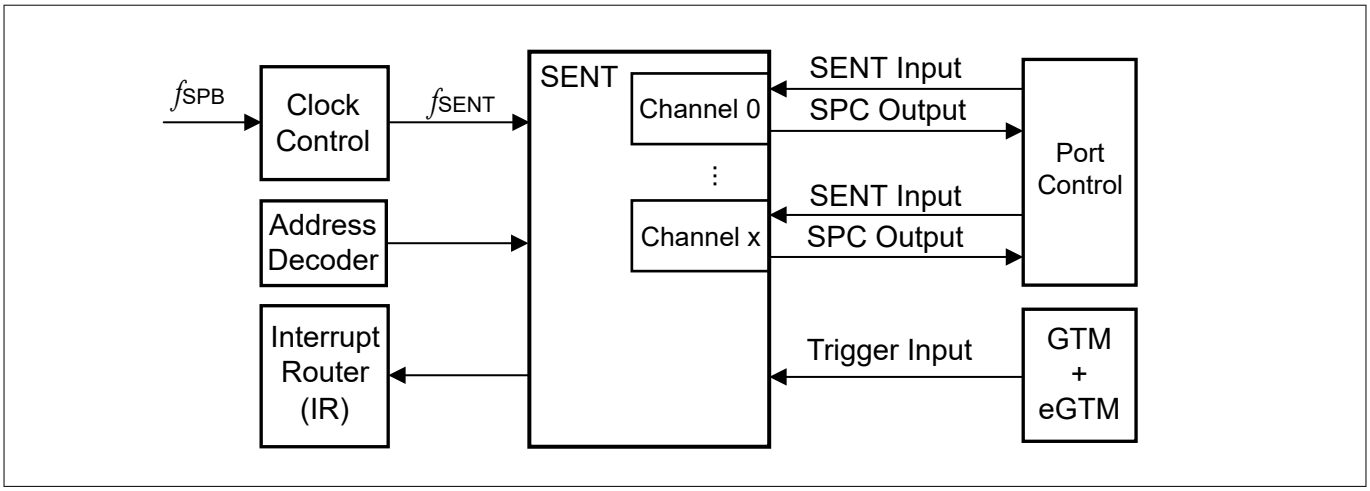


Figure 495 SENT block diagram

The ERAY interface provides a serial communication link typically used to connect sensors or other peripheral devices.

Clock control, address decoding, and service request control are managed by the ERAY module kernel.

The ERAY IP-module performs communication according to the v2.1. While staying compliant to this standard, it is able to cover as well the Short PWM Code (SPC) protocol extensions. This enhances the standardized ERAY protocol defined by v2.1. SPC enables the use of enhanced protocol functionality like “synchronous”, “range selection” and “ID selection” protocol mode.

Receive data on a ERAY channel can be set up according to the underlying application. In particular the number of nibbles forming one value is configurable.

The message storage consists of two 32-bit registers for each channel, representing a flexible double buffer system.

In SPC mode, maintaining the sample and transmission schedule as well as providing message status information is supported.

The register set of the ERAY module can be accessed directly by the CPU for configuration, GTM and eGTM for SPC pulse configuration (Trigger Input), data read out and status query.

Related information

TC4Dx SMU alarm mapping tables on page 7233

25.3 Functional description

This section describes:

- General operation
- Standard SENT operation
- SPC operation
- Baud rate generation
- Error detection capabilities
- Support for frequency drift analysis in frames with pause pulse
- Digital glitch filter
- Interrupts
- Interface connections of the module
- Port control

25.3.1 General operation

The Single Edge Nibble Transmission encoding scheme (SENT) is intended for use in applications where high resolution sensor data needs to be communicated from a sensor to an Engine Control Unit (ECU). It is intended as a replacement for the lower resolution methods of 10-bit A/D converters and PWM and as a simpler low cost alternative to CAN or LIN. The implementation assumes that the sensor is a smart sensor containing a microprocessor or dedicated logic device (ASIC) to create the signal.

The following figure shows a typical application in which a SENT interface reads a sensor device.

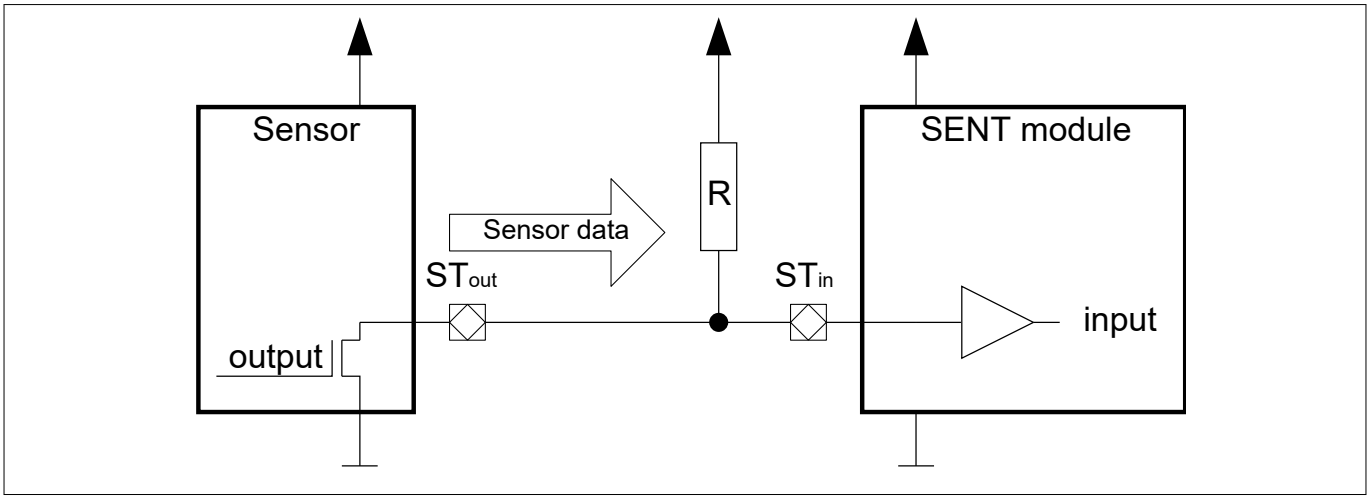


Figure 496 SENT to external device connection

SENT communication is unidirectional from sensor to controller without any synchronization. The sensor signal is transmitted as a series of PWM blocks measured as falling to falling edge times.

The Short PWM Code (SPC) extension overcomes the drawback of uni-directionality while keeping conformance to the standard.

The following figure shows a typical application in which a SPC enabled SENT ECU reads a SPC enabled SENT sensor device.

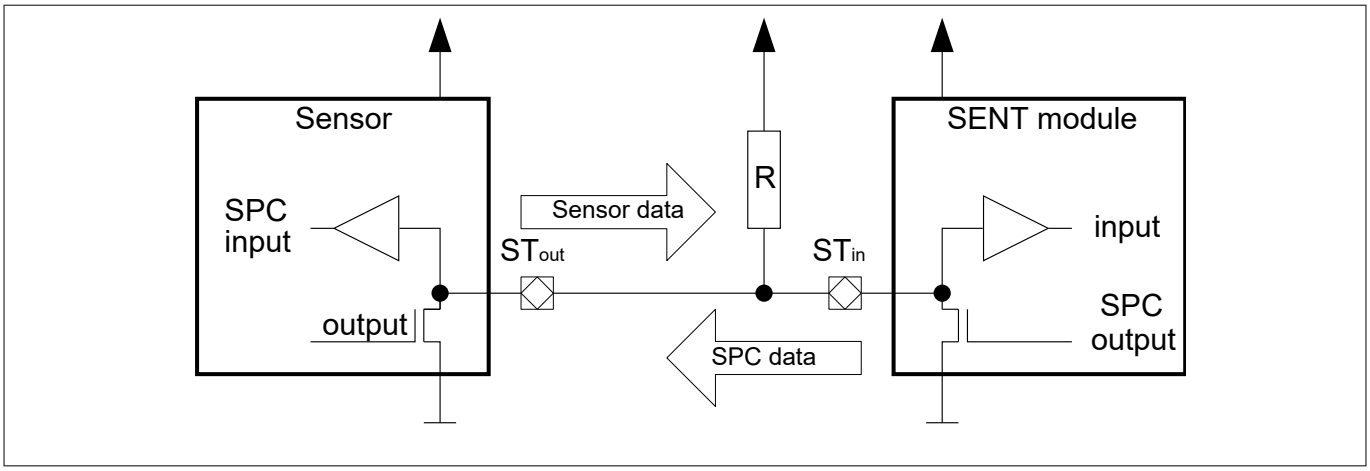


Figure 497 SENT SPC to external device connection

Some applications are:

- Read-out of the external throttle position sensor (for example: SENT enabled linear hall sensor)
- Receiving pedal position
- Synchronize sampling and read-out of up to four sensors on one single line (SPC)
- Selection of 1 out of 4 sensors connected to a single SENT channel
- Serial connections to other peripheral devices

25.3.2 Standard SENT operation

Standard SENT mode supports communication fully compliant to SENT specification J2716 201604, in which only the external device is sending and the ECU is receiving. In this unidirectional communication, both transmitter and receiver use the same data frame format and have the same baud rate. These settings are defined at system integration time and do not change for a given application. Data is received on pin ST. The following figure shows the block diagram of the SENT module when operating in standard SENT mode.

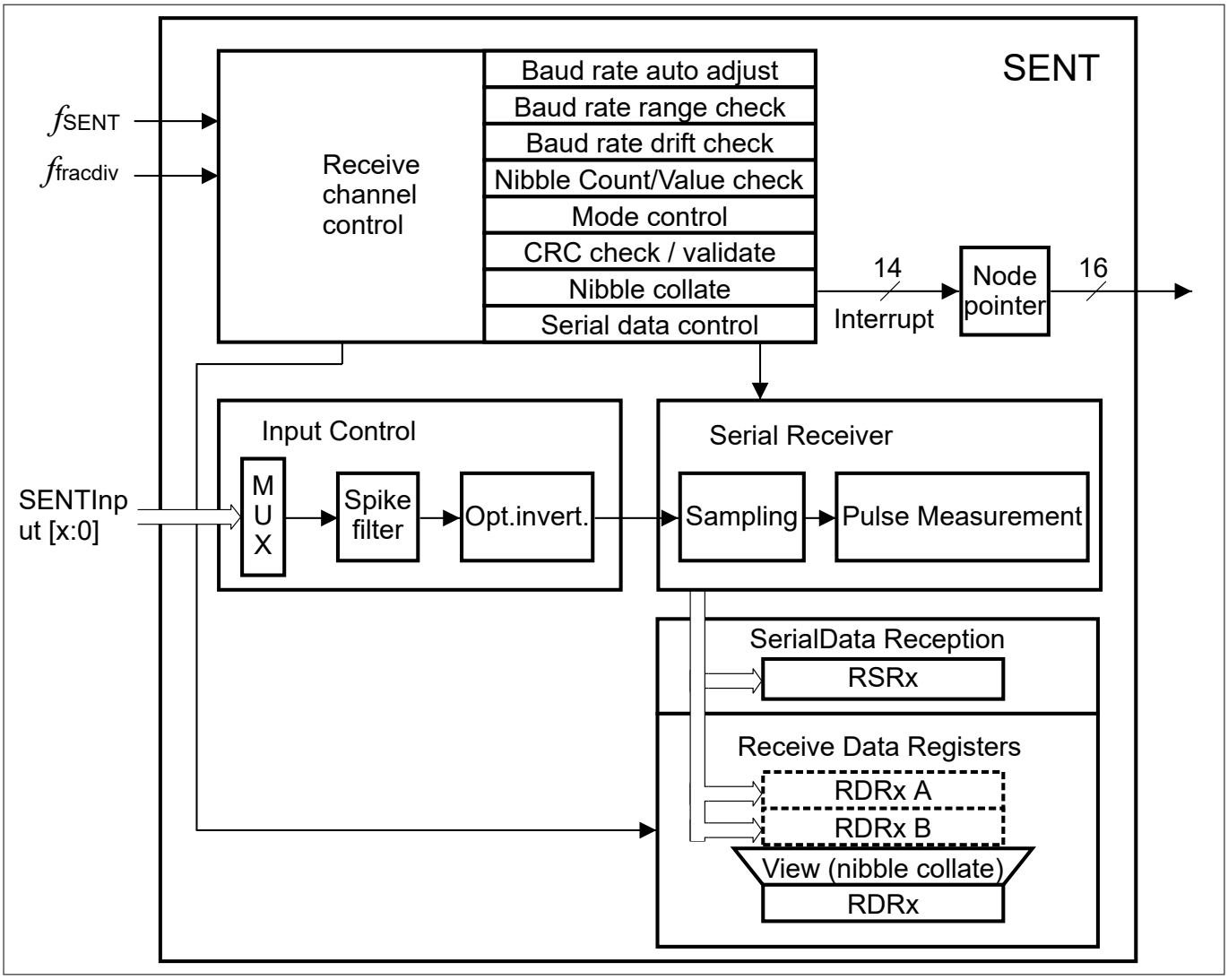


Figure 498 Standard SENT mode operation

25.3.2.1 Frame formats and definitions

This section describes the frame formats and definitions of the SENT protocol.

Basic definitions

The following figure shows the layout and definitions of a standard SENT frame. Note that the SENT standard does not specify whether the most significant nibble or the least significant nibble is sent out first.

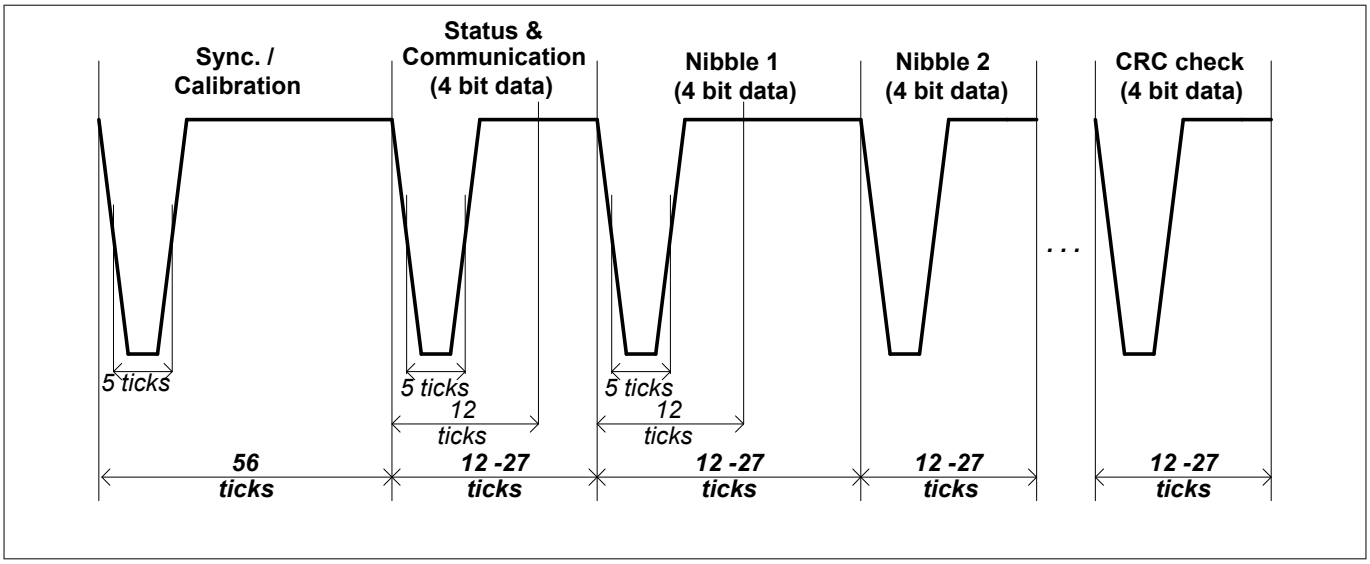


Figure 499 Standard encoding frame

Serial data encoding in status and communication nibble

The Status and Communication nibble is used by sensors that transmit extra serial data such as part numbers or diagnostic information. Data bits from this nibble are constructed across multiple SENT messages to form a Short Serial Message or an Enhanced Serial Message. These messages are alternately referred to as “Slow Channel” data in the SENT specification J2716 201604.

Table 1161 Short serial data encoding

Bit number	Bit function
0	Reserved for specific application
1	Reserved for specific application
2	Serial Data message bits
3	Message start = 1, otherwise = 0

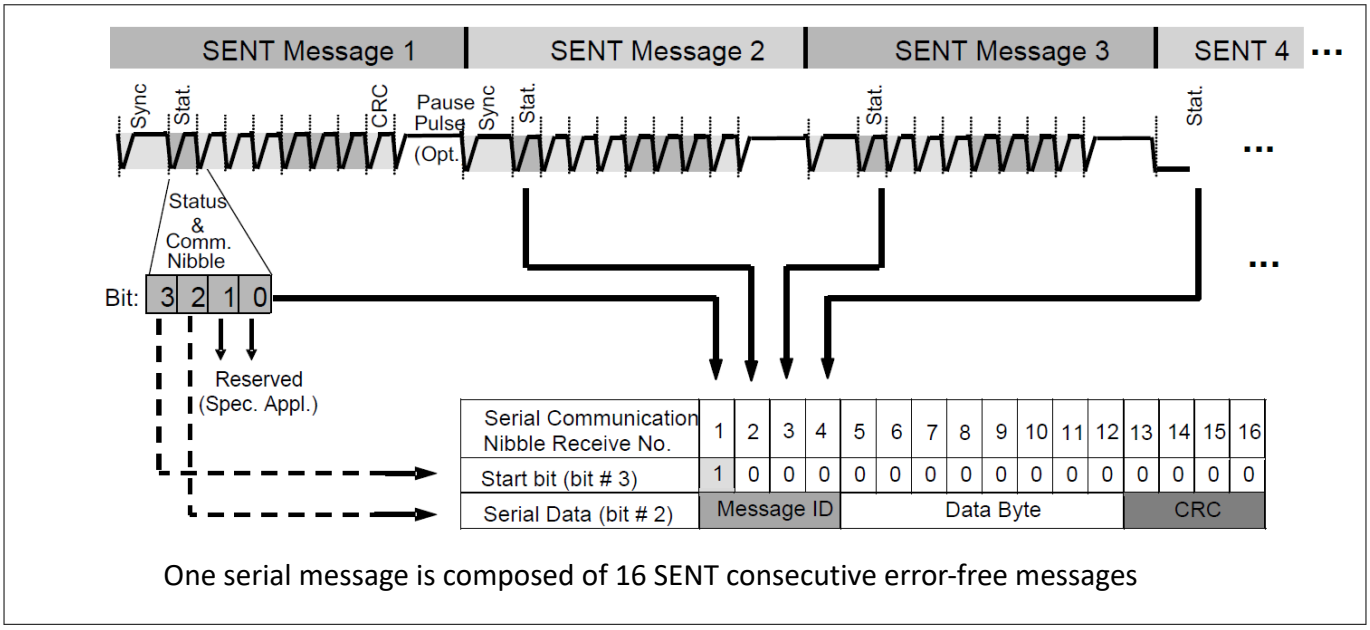


Figure 500 Short serial message, serial data encoding over 16 messages

Short serial message format

The SENT kernel supports the Short Serial Message Format defined in the SENT specification J2716 201604. The Short Serial Message Format provides 16 bits of data, constructed across 16 SENT messages. Each of the 16 messages contains one bit of Short Serial Message data in Status and Communication nibble bit 2.

Serial data is communicated in a 16-bit sequence as shown in the table below. The start of a Short Serial Message is indicated by a 1 in bit 3 of the Status and Communication nibble. That SENT message and the next 15 must be successfully received (no errors) for the Short Serial Message data to be received. The CRC generation is identical to the CRC generation on the data nibbles. Short Serial Message data is provided in the SDSx register.

Table 1162 Short serial message format

Serial communication nibble receive no.	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Startbit (bit # 3)	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Serial data (bit # 2)	Message ID				Data Byte								CRC			

CRC calculation

SENT signal data and Short Serial Message data is secured with a 4-bit CRC. The standard CRC calculation method is defined in the SENT specification J2716 201604.

RCR.CRZ defines, if a zero nibble is added for calculation or not.

The alternative checksum nibble is a 4-bit CRC of the data nibbles (including the status nibble if RCR.SNI is set, as used in sensor TLE4998). The CRC is calculated using a polynomial $x^4 + x^3 + x^2 + 1$ with a seed value of 0101.

In order to facilitate CRC implementations and to avoid ambiguities, an example implementation of the alternative 4-bit CRC is given below. This is used for example in the sensor TLE4998 for the signal data. See the following figure for details.

```
//Fast way for any µC with low memory and compute capabilities
// contains input data (status nibble, 6 data nibble, CRC)
char Data[8] = {…};
// required variables and LUT
char CheckSum, i;
char CrcLookup[16] = {0, 13, 7, 10, 14, 3, 9, 4, 1, 12, 6, 11, 15, 2, 8, 5};
CheckSum= 5; // initialize checksum with seed “0101”
for (i=0; i<7; i++) {
 CheckSum = CheckSum ^ Data[i];
CheckSum = CrcLookup[CheckSum];
};
// finally check if Data[7] is equal to CheckSum
```

For the “Enhanced Serial Message Frame Format”, an own 6-bit CRC calculation method is defined by the standard.

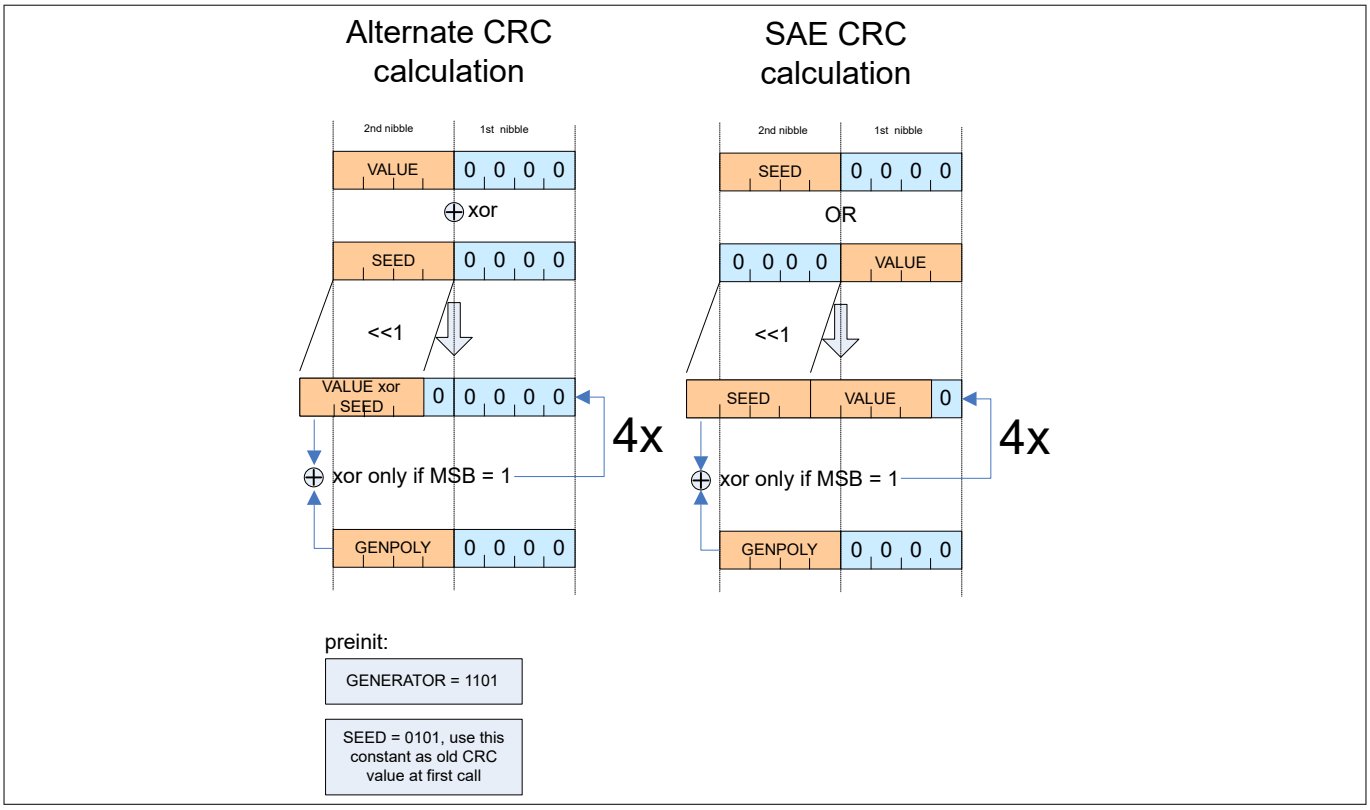


Figure 501 Alternate versus SAE CRC calculation

Enhanced serial message format

The SENT kernel supports the Enhanced Serial Message Format defined in the SENT specification J2716 201604. The Enhanced Serial Message Format provides 21 bits of message data and a 6-bit CRC, constructed across 18 SENT messages. The serial data is transmitted bit wise per frame in bits [3:2] of the Status and Communication nibbles of 18 consecutive messages from the transmitter.

Table 1163 Enhanced serial data encoding

Bit number	Bit function
0	Reserved for specific application
1	Reserved for specific application
2	Serial data message bits
3	Message start =0 1 1 1 1 1 1 0

Enhanced serial message frame

Serial data is communicated in an 18 frame sequence as shown in Table 1164. The start of an Enhanced Serial Message is indicated by the unique pattern “01111110” in bit #3 of the status and communication nibble, constructed across 8 SENT messages (SENT message 18, 1 to 7). The first “1” in a series of six ones in message 1 to 6 indicates the start of a serial message frame. In message 7, 13 and 18 they are set to “0”.

***Note**: In the start pattern of an enhanced serial message “01111110”, the "0" in SENT message 18 (first bit in the pattern) could belong either to a normal SENT frame with no enhanced serial message data or to the final message in a series of 18 consecutive SENT messages constructing the serial message.*

18 consecutive SENT messages must be successfully received (no errors) for the Enhanced Serial Message data to be received. If an error occurs in any of these 18 consecutive frames, then the SENT discard the current serial message and waits for a valid sequence of a new serial message. The 6-bit Enhanced Serial Message CRC is

different from the 4-bit CRC on SENT (fast channel) messages. (More details can be found in the SENT specification J2716 201604)

An Enhanced Serial Message contains 21 bits of message data and a 6-bit frame-check sequence. The configuration bit (serial data bit #3 of the 8th SENT message's Serial Communication Nibble) determines how the SENT module interprets the serial data.

- Configuration bit = 0: 12-bit data and 8-bit message ID
- Configuration bit = 1: 16-bit data and 4-bit message ID

Enhanced Serial Message data is provided in the SDSx register.

Table 1164 Enhanced serial message frame

Serial communication nibble receive no.	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
Serial data (bit # 3)	1	1	1	1	1	1	0	C	8-bit ID (7-4)				0	8-bit ID (3-0) or 4-bit data				0
Serial data (bit # 2)	6-bit CRC						12-bit data field											

***Note**: Messages 17-14 Status Nibble bit # 3 contain either the lower 4 bits (Bit 3-0) of 8-bit ID or 4 additional data bits.*

The following table shows in detail the frame structure for Enhanced Serial Data Frames with Configuration Bit = 0.

Table 1165 Enhanced serial message configuration - Bit = 0

Serial communication nibble receive no.	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
Serial data (bit # 3)	1	1	1	1	1	1	0	0	8-bit ID (7-4)				0	8-bit ID (3-0)				0
Serial data (bit # 2)	6-bit CRC						12-bit data field											

The following table shows in detail the frame structure for Enhanced Serial Data Frames with Configuration Bit = 1.

Table 1166 Enhanced serial message configuration Bit = 1

Serial communication nibble receive no.	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
Serial data (bit # 3)	1	1	1	1	1	1	0	1	4-bit ID (3-0)				0	4-bit data (15-12)				0
Serial data (bit # 2)	6-bit CRC						12-bit data field											

25.3.3 SPC operation

The module supports the SPC (Short PWM Code) protocol, which enhances the standardized SENT protocol (Single Edge Nibble Transmission) defined by SENT specification J2716 201604. SPC enables the use of enhanced protocol functionality due to the ability to select between “synchronous”, “range selection” and “ID selection” protocol mode or even “bidirectional transmit mode”.

25.3.3.1 Synchronous transmission

In the “synchronous” mode, the sensor (slave) starts to transfer a complete data frame only after a low pulse is forced by the master on the OUT pin (SPC Output). This means that the data line is bidirectional - an open drain output of the micro controller (master) sends the trigger pulse. The sensor then initiates a sync pulse and starts to calculate the new output data value. After the synchronization period, the data follows in form of a standard SENT frame, starting with the status, data and CRC nibbles. At the end, an end pulse allows the CRC nibble

decoding and indicates that the data line is idle again. The timing diagram below visualizes a synchronous transmission

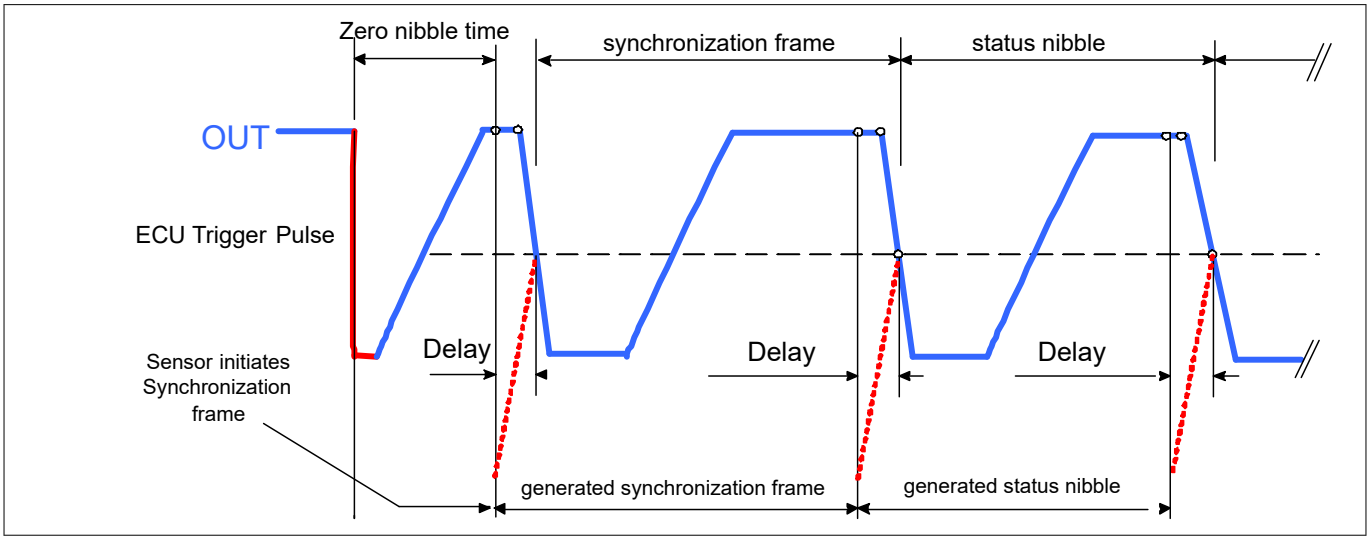


Figure 502 Synchronous transmission (SPC)

25.3.3.2 Range selection

The low time duration of the master can be used to select the range of the sensor in SPC dynamic range selection mode.

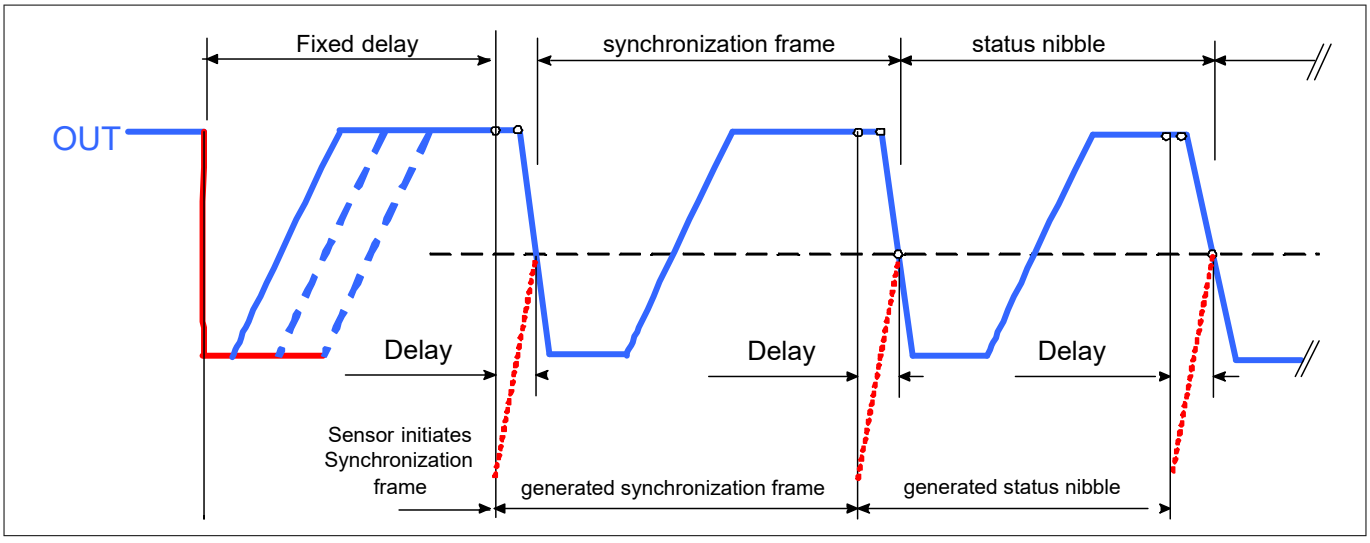


Figure 503 Range (SPC)

25.3.3.3 ID selection

This functionality is similar to the previous mode, but instead of switching the range of one sensor, one of up to four sensors are selectable on a bus (bus mode, 1 master with up to 4 slaves). This allows parallel connection of up to 4 sensors using only three lines (VDD, GND, OUT), as illustrated in the following figure.

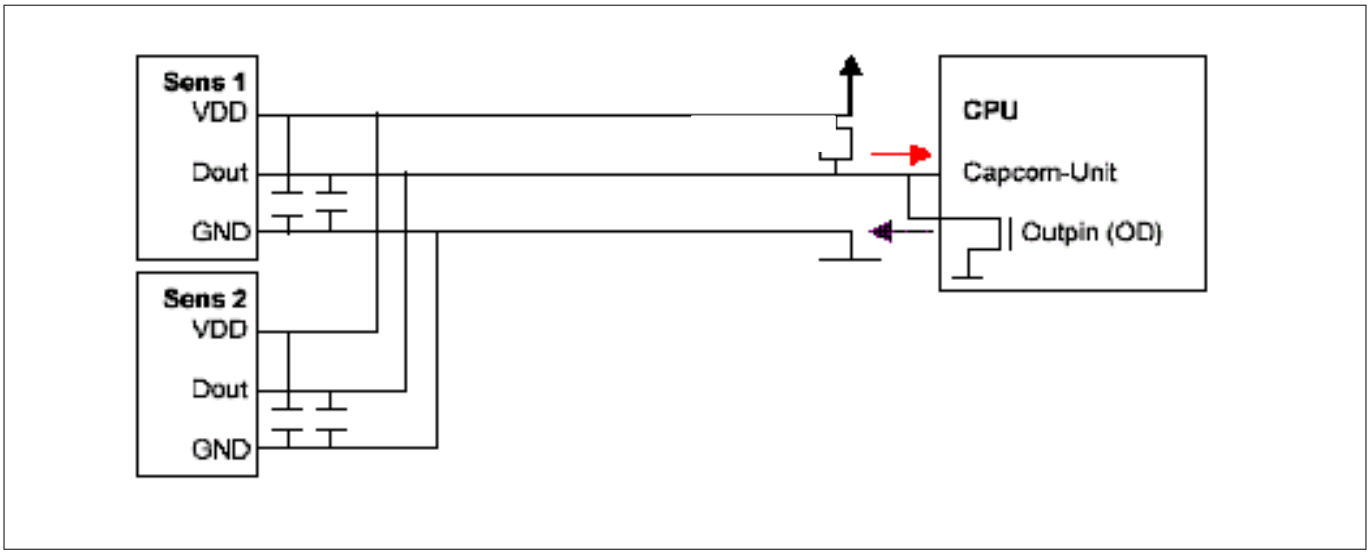


Figure 504 ID selection (SPC)

25.3.3.4 Bidirectional transmit mode

After addressing the sensor the ECU interrupts the sensor sync pulse by pulling down the line. The ECU starts to transmit own nibbles on the time base of the selected sensor. The ECU has to adapt the sensors timing from an earlier received cycle.

The sensor detects the falling edge that infringes the protocol and switches to receive mode as illustrated in the following figure.

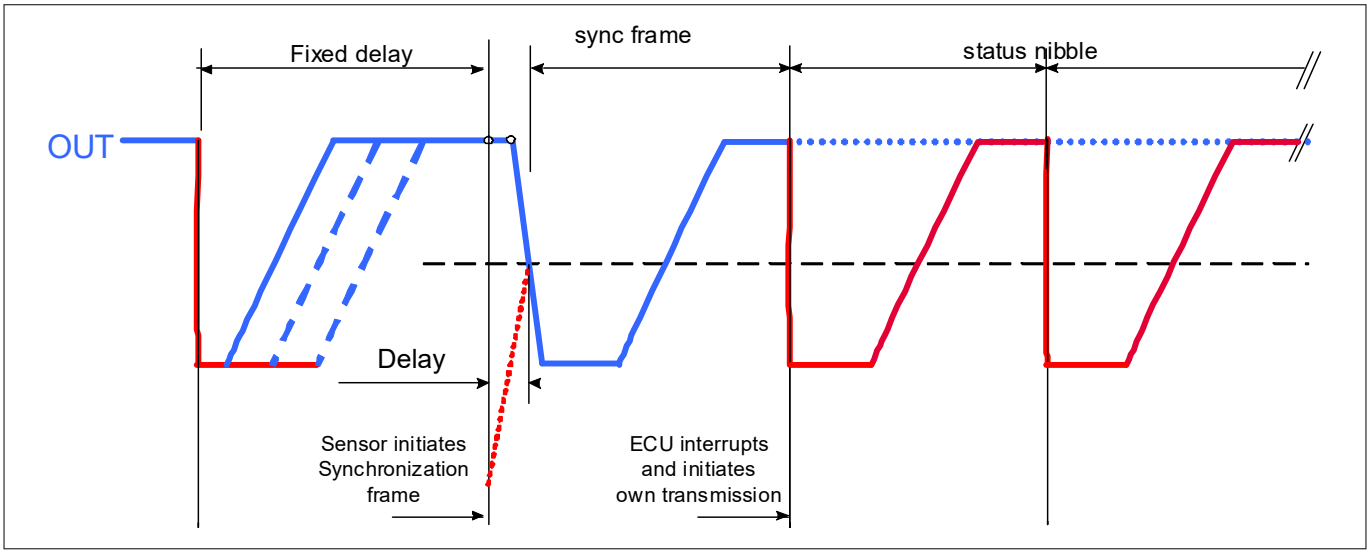


Figure 505 Bidirectional transmit mode (SPC)

25.3.3.5 SPC timing

SPC transmission is initiated by a Master pulse on the OUT pin. To detect a low level on the OUT pin, the voltage must be below a threshold V_{thf}. The sensor detects that the OUT line has been released as soon as V_{thr} is crossed. The following figure shows the timing definitions for the master pulse. The master low time t_{mlow} as well as the total trigger time t_{mtr} are individual for the different SPC modes and are given in the sensors

specifications. It is recommended to chose the typical master low time exactly between the minimum and the maximum possible time given by the connected sensor: $t_{mlow,\,typ} = \frac{(t_{mlow,\,min} + t_{mlow,\,max})}{2}$.

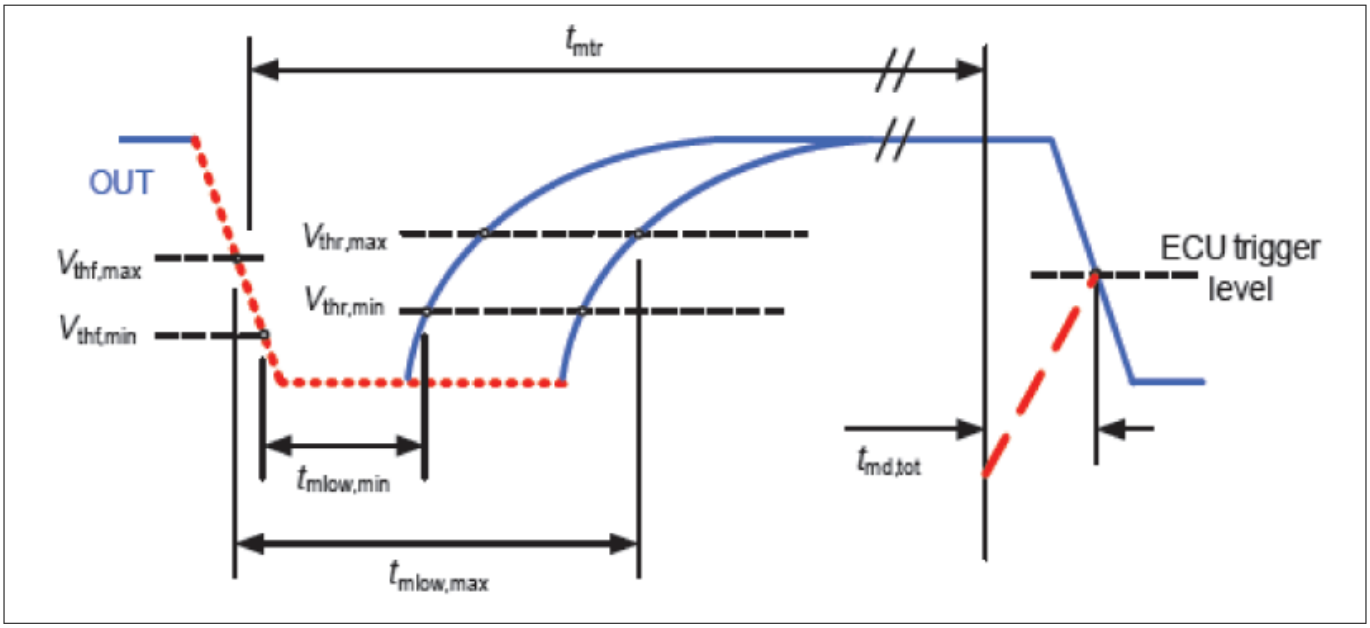


Figure 506 Timing (SPC)

25.3.3.6 Abort of frames

Only a reset condition of the device, a hard suspend or choosing Mode 0 by clearing bit-field SCRx.TRIG can abort a current SPC transmission immediately. During hard suspend, the port output drives its current value until suspend is terminated. If the receive channel becomes disabled, the soft suspend mode is requested, or the sleep mode is entered, the SENT module still does start a new frame transmission. If one of these three conditions becomes active during a running frame transmission, the frame transmission is completely finished before the requested abort state is entered. Note that in this case no frame finished interrupt is generated any more.

Received frames are aborted immediately on a disable, suspend or sleep request.

25.3.4 Baud rate generation

The nominal baud rate of each channel is individually adjustable. This is required as it is depending on the connected sensor and its current deviation from nominal frequency.

The actual baud rate of the channel follows automatically the baud rate of the connected device and its current deviation from nominal frequency. The adaptation range is +/- 25% as specified in SENT specification J2716 201604.

This chapter shows in detail how the baud rate for each channel is adjusted. $\frac{1}{f_{tick_x}}$ defines the resulting tick length.

In the first stage, a global fractional divider serves as pre divider. Its intermediate frequency $f_{fracdiv}$ can be set up as input frequency for the different SENT channels. The clock signal f_{pdiv} of a channel must always be at least 20 times the nominal tick frequency of the channel. This is to allow for the highly flexible and big tolerance of +/-25% versus the sending device. In addition the module must be able to detect a deviation in the length of 2 consecutive Synchronization and Calibration pulses of 1.5625% ($\frac{1}{64}$) and adapt the own baud rate. The tick

time is typically 3 µs but can vary by standard and take values up to 90 µs. To achieve this each channel has its own fractional divider. This allows to downscale the frequency of f_{tick} precisely to the required tick frequency.

The SENT module provides two clock signals to the channels (See the figure below):

- f_{SENT}
  This is the module clock that is used inside the SENT kernel for control purposes such as clocking of control logic and register operations. The frequency of f_{SENT} is always identical to the system clock frequency f_{SPB}. The clock control register SENT_CLC makes it possible to enable or disable f_{SENT} under certain conditions
- $f_{fracdiv}$
  This clock is the module clock that is used inside the SENT kernel for baud rate generation of the serial channels. The fractional divider register SENT_FDR controls the frequency of $f_{fracdiv}$. DM bit-field is usually not required and set to 1

The channels generate two local clock signals:

- f_{pdiv_x}
  This clock is the channel clock that is used inside the SENT channel x for baud rate generation of the serial channel. The divider register SENT_CPDRx (x=0-14) controls the frequency of f_{pdiv_x}
- f_{tick_x}
  This clock is the channel clock that is used inside the SENT channel as the baud rate frequency. The fractional divider register SENT_CFDRx (x=0-14) controls the frequency of f_{tick_x}. The higher the value of bit-field DIV the higher the precision with which the synchronization and calibration pulse and the deviation in the length of two consecutive synchronization and calibration pulses (drift) can be measured. The bit-field DIV must be set in the range of [2200, 52428]

Each time a Synchronization and Calibration pulse starts bit-field DIVM is measured. DIVM is the result of measuring the calibration pulse duration with f_{pdiv_x}. At the end of the Synchronization and Calibration pulse this value is taken as new divider of the register CFDR.

Each time a pulse starts, the internal accumulator of the CFDR is preset with $\frac{DIVM}{2}$. This is required to center the data eye and to make the margin symmetrical.

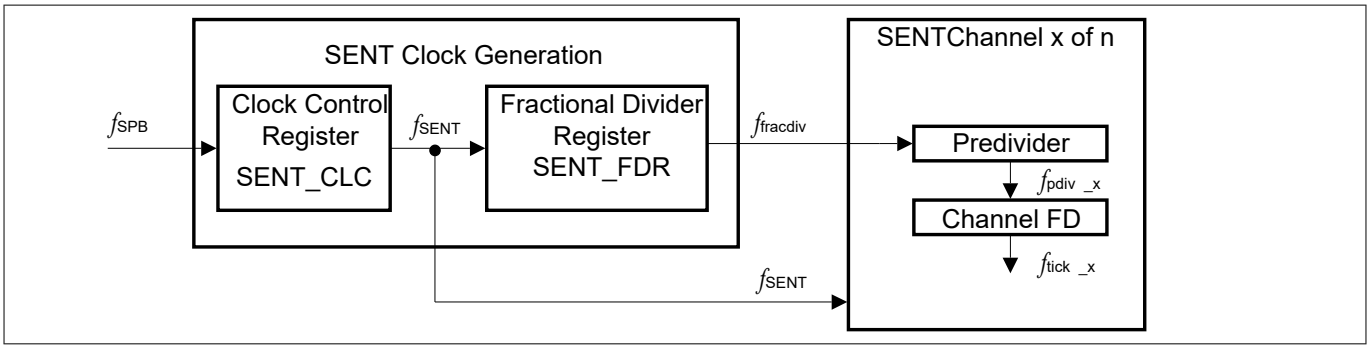


Figure 507 SENT module clock generation

The following two equations define the frequency of $f_{fracdiv}$:

$$f_{fracdiv} = f_{SENT} \div (1024 - SENT_FDR.STEP) when FDR.DM = 01_B \quad (11)$$

$$f_{fracdiv} = f_{SENT} \times SENT_FDR.STEP \div 1024 \text{ with } STEP = 0...1023 \text{ when} FDR.DM = 10_B \quad (12)$$

The following formula defines the frequency of f_{pdiv_x}:

$$f_{pdiv_x} = f_{fracdiv} \div (SENT_CPDRx.PDIV + 1) \quad (13)$$

The following formula defines the nominal frequency of f_{tick_x}. For the actual frequency of f_{tick_x} DIV is replaced by DIVM after the current sensor frequency was validly measured.

$$f_{tick_x} = \left(f_{pdiv_x} \times 56\right) \div SENT_CFDRx.DIV \text{ with } DIV = \text{from } 2200 \text{ to } 52428 \quad (14)$$

25.3.4.1 Operation outside the granted standard limits

The SENT module is granted to comply SENT specification J2716 201604. The tolerances given in this standard are valid for tick times between 3 µs and 90 µs. Nevertheless, the SENT module can operate outside this standard. A minimum tick time of 0.2 µs is programmable. In these use cases outside the standard it is in the responsibility of the system integrator to do proper jitter calculation and system design. Same holds true for tick times of greater than 90 µs. Note that the low pass filter as given in SENT specification J2716 201604 needs to be replaced by a circuit fit for the intended use case as well. Such use is not recommended, as a standard is missing for the system.

25.3.5 Error detection capabilities

Each SENT channel can detect and signal the following error conditions:

Protocol level:

- Calibration pulse length deviates more than +/-25% from the nominal 56 ticks
- Too many or too few nibbles between calibration pulses
- Checksum error
- Successive calibration pulse differ by more than 1.5625%
- Any nibble data values measured as < 0 or >15
- Wrong status and communication nibbles
- Serial communication CRC error

Any of those errors shall cause the receiver to begin searching for a valid calibration pulse to resynchronize.

Transfer Management Level:

- Receive data buffer overrun
- SPC data buffer underrun

25.3.6 Support for frequency drift analysis in frames with pause pulse (FDFL mode)

Each SENT channel supports FDFL mode, that is monitoring the following condition for messages with pause pulse and fixed message length:

Ratio of calibration pulse to message length varies by $> \frac{1}{64}$ or $< -\frac{1}{64}$ from one message to another.

The FDFL mode is entered if RCRx (x=0-14).FDFL, RCRx (x=0-14).IEP and RCRx (x=0-14).IDE are set. RCRx (x=0-14).CFC is ignored in this mode.

There is no automatic calculation or check done by hardware in this mode nor is FDI issued. Frames that have an invalid frequency drift and would cause an FDI are still accepted and not dropped by hardware. FRI is not affected by this mode. This means that the hardware still checks the sensor frequency range. The calculation of

the ratio and the comparison with the ratio of the preceding frame needs to be done in software. The software needs to decide whether or not to accept the frame based on this calculation.

The value for the calibration pulse length can be read by software from CFDRx (x=0-14).DIVM.

The value for the message length including pause pulse can be read by software from RCRx (x=0-14).FRLEN only if RCRx (x=0-14).FDFL, RCRx (x=0-14).IEP and RCRx (x=0-14).IDE are set.Both values are valid after RSI. RSI indicates the Frame Buffer valid immediately after the Pause Pulse belonging to the current frame is received if no other error occurs.

FRLEN is a 16-bit counter with saturation (stops at 0xFFFF/65.535) and is driven by f_{pdiv}. These factors limit the maximum supported length of frames including pause pulses.

For the typical use case $\frac{1}{f_{tick}} = 3\,\mu s$, a good choice for f_{pdiv} is:

16.6 MHz (PDIV = 5; f_{SENT} = 100 MHz)

The frame length can be counted up to 3.93 ms.

Note that the nominal frame length must be 20% shorter to allow the system to compensate a 20% frequency drift. This is just an example calculation and not all products may support all frequencies.

The following figure shows the two times that are measured and that need to be set in relation to each other:

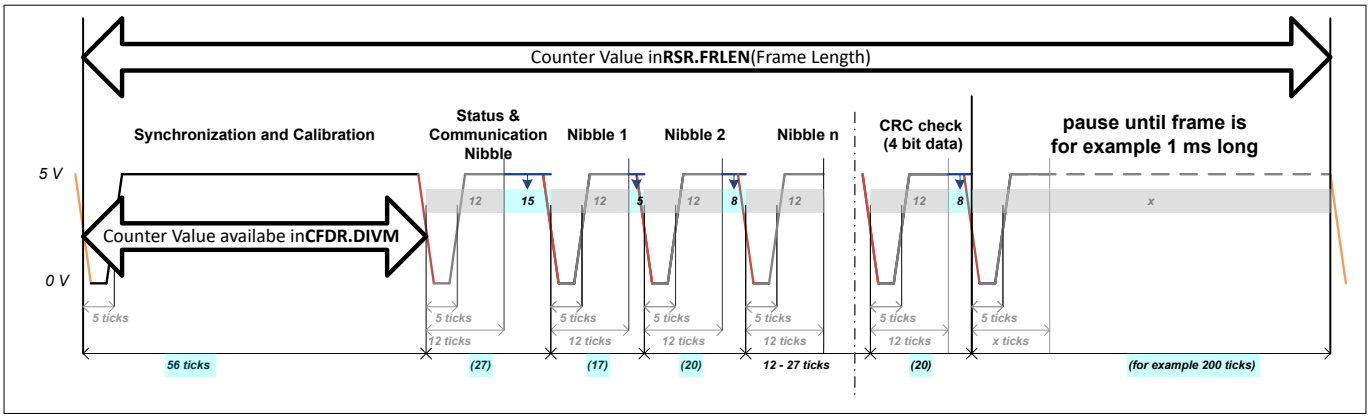


Figure 508 Frequency drift analysis in frames with pause pulse

25.3.7 Digital glitch filter

Very slow slopes and signal noise can lead to fast transitions of the input signal. These unwanted transitions are suppressed by a digital glitch filter similar to a Filter and Prescaler Cell (FPC) in Delayed Debounce Filter Mode with up and down (no reset).

It is built for filtering very fast transitions only. The filter calculates the integral of the signal. If the integral reaches a programmable saturation point, the signal change is notified to the pulse measurement unit. So it helps to find the exact pulse length for slow slopes in noisy environment.

The glitch filter is clocked with f_{pdiv}. The following conditions may apply:

- If the state of the input sample differs from the current output signal value, the internal counter is incremented by one
- If the state of the input sample matches the current output signal value and the timer is not in idle, the timer is decremented by one
- If the timer matches the compare value stored in IOCRx (x=0-14).DEPTH, the level of the output signal line is inverted. The timer will be reset and set to idle state again

The depth of the filter can be programmed to a value between 1 and 15. Typically a depth of 3 to 5 T_{pdiv} is sufficient. By default it is cleared. If IOCRx (x=0-14).DEPTH is cleared the filter is inactive. Nevertheless, the input signal is sampled with f_{pdiv}.

The internal signal after the filter will change value not before the new value was sampled DEPTH times. If during this period a spike occurs, it takes $2 \times T_{\text{pdiv}}$ times longer for the internal signal to change value. The filters principal implies a delay of the signal by ($\text{DEPTH} \times T_{\text{pdiv}}$).

Upon detection of glitch during rising or falling edge, IOCRx (x=0-14).REG or IOCRx (x=0-14).FEG is set. The rising and falling edge glitch flags must be reset by software.

The following figure shows the digital glitch filter:

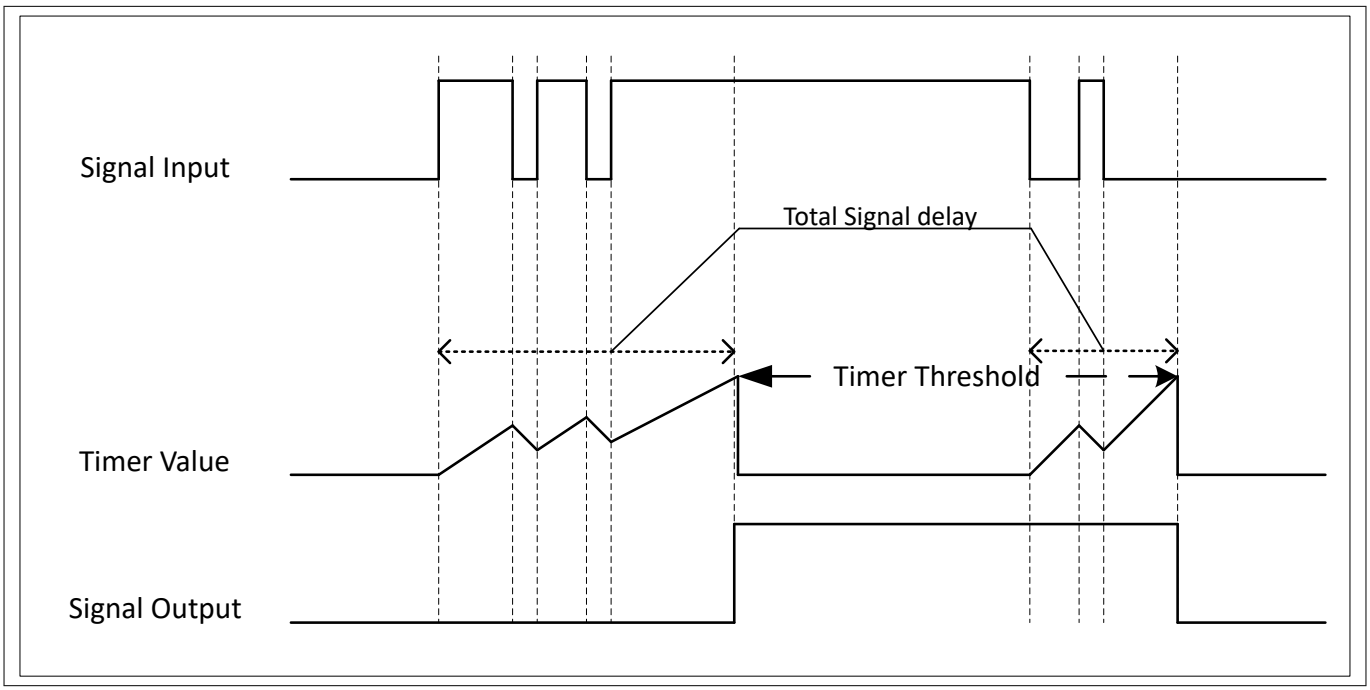


Figure 509 Digital glitch filter

25.3.8 Interrupts

For the SENT module, each channel has a total of 14 interrupts:

- RDI: Receive data interrupt is activated when a received frame is moved to the Receive Data Register RDRx (x=0-14)
- RSI: Receive frame success interrupt, that is the CRC was successful

***Note**: Both RDI and RSI will be issued together in normal use cases where the frame size is not bigger than 8 nibbles and CRC is correct. For more than 8 data nibbles, please refer to RCRx (x=0-14).FRL description.*

- RBI: Receive buffer overrun interrupt is activated when a new frame is transferred to a Receive Data Register RDRx (x=0-14) while the old value was still not read by the host (“overwrite”), that is the kernel wants to set any of the two interrupts RSI and RDI and finds any of these two interrupts already set
- TDI: Transmit interrupt is activated when data is moved from a SCR to a transmit shift register
- TBI: Transfer buffer under run interrupt is set after data has been completely transferred (PLEN exceeded) and no new data was written to SCRx (x=0-14)
- FRI, FDI, NNI, NVI, CRCI: Protocol error interrupts, if one of the protocol error interrupts is activated, data is to be treated as invalid according to SENT specification J2716 201604
- WSI, SDI, SCRI treat the interrupts referring to the Status and Communication nibble
- WDI: Watch Dog Error Interrupt is issued if the time between two frames is too long

For acceleration of the interrupt service routine, a Register INTOV is implemented that holds a flag for each channel. This flag is automatically set if there is an interrupt pending for the channel which is enabled. It is automatically reset, if no more enabled interrupt is pending for this channel.

The interrupt request or the corresponding interrupt set bit (in register INTSETx (x=0-14)) can trigger the interrupt generation at the selected interrupt node. The service request pulse is generated independently from the interrupt flag in registerINSTATx (x=0-14). The interrupt flag can be cleared by software by writing to the corresponding bit in register INTCLRx (x=0-14). If more than one interrupt source is connected to the same interrupt node pointer (in register INPx (x=0-14)), the requests are combined to one common line.

***Note**: Only 10 Interrupt request lines are available for the SENT module to interface with the Interrupt Router.*

25.3.9 Interface connections of the SENT module

The following figure shows a more specific implementation details and interconnections of the SENT module.

The SENT module is supplied with a separate clock control, address decoding, and interrupt control logic. Outputs of GTM timer are connected to the SENT Trigger inputs.

The inputs of the receive channels of the SENT module as well as the SPC outputs (SPCn) are connected to GPIO lines. If SPC outputs are mapped to the same port pin like the referring SENT data input line on a specific product, they can be used in a quasi bidirectional mode (that is open drain out and input on the same pin).

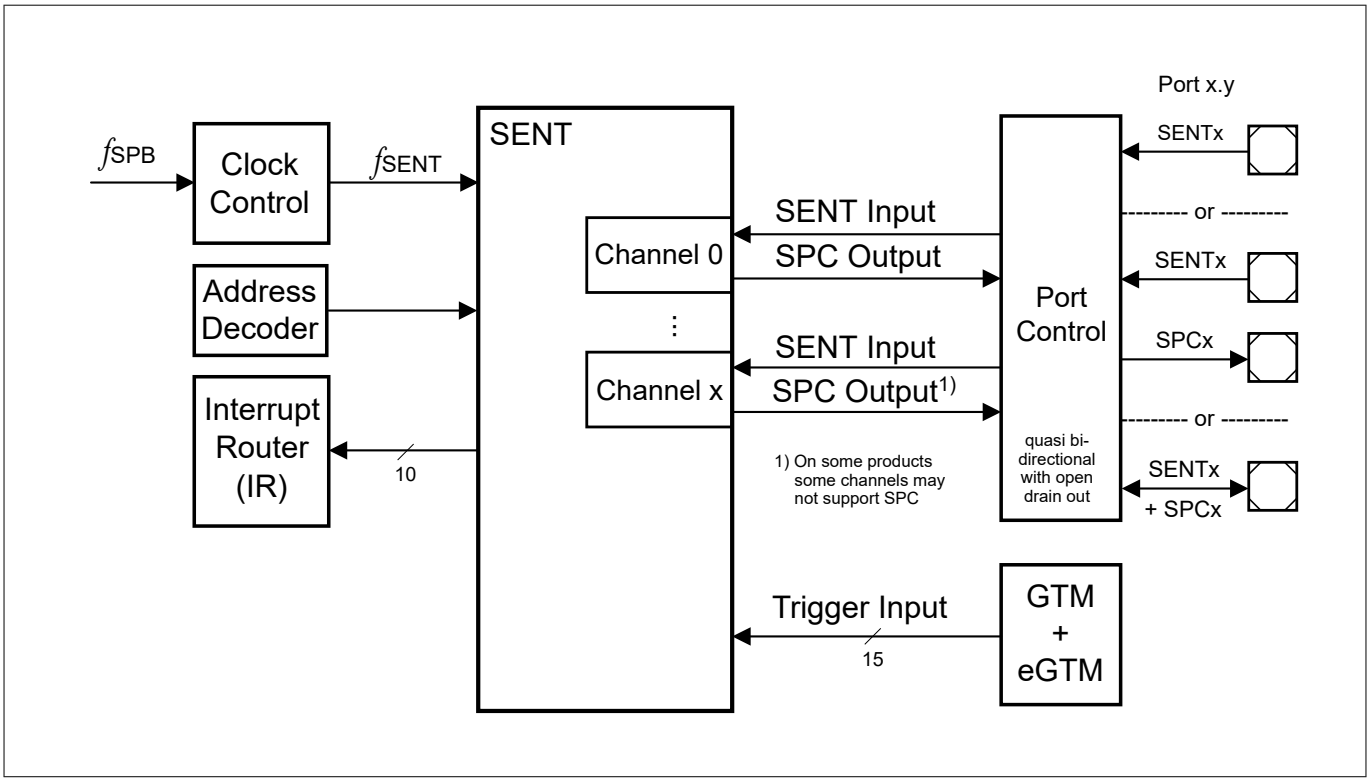


Figure 510 SENT module implementation and interconnections

25.3.9.1 Trigger inputs

The module has 15 SENT channels and the same number of trigger inputs. They can be randomly chosen by programming IOCRx (x=0-14).ETS. The trigger inputs (TRIG[14:0]) of the SENT module are connected to GTM and eGTM as shown in the SENT connectivity chapter.

25.3.10 Port control

The SENT input channels are overlain with standard ADC channels as they are replacing former analog signals. In addition each channel is connected to general purpose I/O pads as well. The selection from one of up to 4 alternate inputs per channel is done by application software by programming IOCRx (x=0-14).ALTI.

Depending on the implementation some products support SPC output as well on some of the channels.

The following port control operations must be executed for the SPC output I/O pads:

- Input/output function selection (Port IOCR registers)
- Pad driver characteristics selection for the outputs (Port PDR registers)

25.3.10.1 Input/output function selection

SENT can be used in three different ways:

- ADC input (dedicated for ADC) overlain with SENT digital input and no output option on this pad
- Standard general purpose input/output function on two different I/O pads per channel. This means that it is an input on ADC or on I/O pad while the output is chosen to be on a different I/O pad. This allows the use of external transceivers
- SPC open drain output on a digital I/O pad and SENT input from same digital I/O pad (“quasi bidirectional SENT/SPC mode”)

The SENT module overlays dedicated analog to digital converter (ADC) pads as digital input pad.

SENT physical layer uses 5 V signal voltage levels. All SENT inputs are mapped to 5 V compatible ADC pads. V_{ddm} must be supplied with 5 V for SENT operation.

The port input/output control registers contain the bit-fields that select the digital output and input driver characteristics such as port direction (input/output), alternate output selection, pull-up and pull-down devices (for pure inputs only), and open-drain selections. The I/O pads for the SENT module are controlled by the Port input/output control registers. Please see Port chapter to make conscious decisions here.

Quasi bidirectional settings are possible only, where an SPC output and a SENT input of the same channel are connected to the same pad. This configuration is not available on all SENT input pads and varies by product.

For some channels SPC outputs and SENT inputs are connected to the same I/O pad. This allows for quasi bidirectional operation: That is the output is configured as open drain (for example P00_IOCR0.PC1=11110_B) while the same pad is used as input (for example P00_IOCR0.PC1=$0XXXX_B$). The port in the example may not be present in all products.

Note that in the examples, figures and tables of this chapter, no quasi bidirectional operation is shown. Only pure inputs and pure outputs are shown for the sake of simplicity.

Alternate inputs that are not connected to any pin are connected to GND internally. For example the input selected by IOCRx (x=0-14).ALTI=0011_B (also known as: “input D”) is not present in all products.

25.3.11 Definitions

Table 1167 Definitions

Name	Description
SENT	Single Edge Nibble Transmission
Nibble	Four Bit value between 0 and 15 = half a Byte = one character in hex (0 to F)
SPC	Short PWM Code
PWM	Pulse Width Modulation

(table continues...)

Table 1167 (continued) Definitions

Name	Description
ASIC	Application Specific Integrated Circuit
CAN	Controller Area Network
LIN	Local Interconnect Network
ISO	International Organization for Standardization
ECU	Electronic Control Unit
FSM	Finite State Machine

25.4 Registers

25.4.1 Register overview - access mode glossary

Table 1168 Register overview - access mode glossary

Keyword	Description
E	Access protection using PROT register PROTE.
SE	Access protection using PROT register PROTSE.
APU-P	Protection group consisting of registers ACCEN_WRA, ACCEN_WRB, ACCEN_RDA, ACCEN_RDB, ACCEN_VM, ACCEN_PRS.
P	Access protection using *APU-P* registers.
BE	Always returns a Bus Error.
U	No access restrictions.
PROT	Access restrictions as defined in the PROT register access rules.

25.4.2 Registers overview - SENT (ascending offset address)

Table 1169 Registers overview - SENT (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CLC	Clock control register	000_H	P	P, SV, E	Application Reset	4807
OCS	OCDS control and status register	004_H	P	SV, P	Debug Reset	4808
ID	Module identification register	008_H	P	BE	PowerOn Reset	4809
RST_CTRLA	Reset control register A	$00C_H$	P	P, SV, E	Application Reset	4809
RST_CTRLB	Reset control register B	010_H	P	P, SV, E	Application Reset	4810
RST_STAT	Reset status register	014_H	P	BE	Application Reset	4811

(table continues...)

Table 1169 (continued) Registers overview - SENT (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PROTE	PROT register endinit	018_H	U	SV, PROT	Application Reset	4811
PROTSE	PROT register safe endinit	$01C_H$	U	SV, PROT	Application Reset	4813
ACCEN_WRA	Write access enable register A	020_H	U	SE, SV	Application Reset	4814
ACCEN_WRB	Write access enable register B	024_H	U	SE, SV	Application Reset	4815
ACCEN_RDA	Read access enable register A	028_H	U	SE, SV	Application Reset	4815
ACCEN_RDB	Read access enable register B	$02C_H$	U	SE, SV	Application Reset	4816
ACCEN_VM	VM access enable register	030_H	U	SE, SV	Application Reset	4816
ACCEN_PRS	PRS access enable register	034_H	U	SE, SV	Application Reset	4817
FDR	SENT Fractional divider register	040_H	P	E, P	Kernel Reset	4818
INTOV	Interrupt overview register	044_H	P	BE	Kernel Reset	4818
TSR	Module time stamp register	048_H	P	BE	Kernel Reset	4819
TPD	Time stamp predivider register	$04C_H$	P	E, P	Kernel Reset	4819
RDRx	Receive data register x	080_H+x*4	P	BE	Kernel Reset	4820
CPDRx	Channel pre divider register x	100_H+x*40_H	P	P	Kernel Reset	4820
CFDRx	Channel fractional divider register x	104_H+x*40_H	P	P	Kernel Reset	4821
RCRx	Receiver control register x	108_H+x*40_H	P	P	See 4822	4822
RSRx	Receive status register x	$10C_H$+x*40_H	P	BE	Kernel Reset	4826
SDSx	Serial data and status register x	110_H+x*40_H	P	BE	Kernel Reset	4827
IOCRx	Input and output control register x	114_H+x*40_H	P	P	Kernel Reset	4828
SCRx	SPC control register x	118_H+x*40_H	P	P	Kernel Reset	4831
VIEWx	Receive data view register x	$11C_H$+x*40_H	P	P	Kernel Reset	4833

(table continues...)

Table 1169 (continued) Registers overview - SENT (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
INTSTATx	Interrupt status register x	120_H+x* 40_H	P	BE	Kernel Reset	4834
INTSETx	Interrupt set register x	124_H+x* 40_H	P	P	Kernel Reset	4837
INTCLRx	Interrupt clear register x	128_H+x* 40_H	P	P	Kernel Reset	4839
INTENx	Interrupt enable register x	$12C_H$+x* 40_H	P	P	Kernel Reset	4841
INPx	Interrupt node pointer register x	130_H+x* 40_H	P	P	Kernel Reset	4843
WDTx	Watch dog timer register x	134_H+x* 40_H	P	P	Kernel Reset	4844
RTSx	Receive time stamp register x	$A80_H$+x* 4	P	BE	Kernel Reset	4845

25.4.3 Clock control register

The Clock control register CLC allows the programmer to adapt the functionality and power consumption of the module to the requirements of the application.

Register CLC controls the module clock signal and the reactivity to the sleep signal.

CLC Offset address: 000_H

Clock control register Application Reset value: $0000\ 0003_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												EDIS	0	DISS	DISR
r												rw	r	rh	rw

Field	Bits	Type	Description
DISR	0	rw	**Module Disable Request Bit** Used for enable/disable control of the module. Also the analog section is disabled by clearing ANONS. 0_B On request: enable the module clock 1_B Off request: stop the module clock
DISS	1	rh	**Module Disable Status Bit** 0_B Module clock is enabled 1_B Off: module is not clocked

(table continues...)

(continued)

Field	Bits	Type	Description
EDIS	3	rw	**Sleep Mode Enable Control** Used to control the module's reaction to sleep mode. 0_B Sleep mode request is enabled and functional 1_B Module disregards the sleep mode control signal
0	2, 31:4	r	**Reserved** Read as 0; should be written with 0.

25.4.4 OCDS control and status register

The OCDS control and status register OCS controls the module's behavior in suspend mode (used for debugging).

Register OCS is cleared by Debug Reset.

If OCDS is being disabled, the OCS register value will not change. When OCDS is disabled the OCS suspend control is ineffective.

OCS Offset address: 004_H

OCDS control and status register Debug Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		SUSSTA	SUS_P	SUS				0							
r		rh	w	rw				r							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															
r															

Field	Bits	Type	Description
SUS	27:24	rw	**OCDS Suspend Control** Controls the sensitivity to the suspend signal coming from the OCDS Trigger Switch (OTGS) Not listed combinations are reserved. 0_H Will not suspend 1_H Hard suspend: Clocks will be disabled immediately 2_H Soft suspend mode Clock will be disabled after kernel acknowledge. 3_H Soft suspend mode Kernel clock will not be disabled after acknowledge
SUS_P	28	w	**SUS Write Protection** SUS is only written when SUS_P is 1, otherwise unchanged. Read as 0.
SUSSTA	29	rh	**Suspend State** 0_B Module is not (yet) suspended 1_B Module is suspended
0	23:0, 31:30	r	**Reserved, write 0, read as 0**

Table 1170 Access mode restrictions of OCS sorted by descending priority

Mode name	Access mode		Description
write 1 to .**SUS_P**	rw	SUS	Set SUS_P during write access
(default)	r	SUS	

25.4.5 Module identification register

The SENT Module identification register ID contains read-only information about the module version.

ID Offset address: 008_H

Module identification register PowerOn Reset value: $0080\ C013_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MOD_NUM															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MOD_TYPE								MOD_REV							
r								r							

Field	Bits	Type	Description
MOD_REV	7:0	r	**Module Revision** Indicates the revision number of the implementation.
MOD_TYPE	15:8	r	**Module Type** This internal marker is fixed to $C0_H$.
MOD_NUM	31:16	r	**Module Number** Indicates the module identification number (0080_H = SENT).

25.4.6 Reset control register A

RST_CTRLA Offset address: $00C_H$

Reset control register A Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				GRSTEN3	GRSTEN2	GRSTEN1	GRSTEN0	0							KRST
r				rw	rw	rw	rw	r							rwh

Field	Bits	Type	Description
KRST	0	rwh	**Kernel Reset** Request a kernel reset. The requested reset is executed if CTRLA.KRST=1 and CTRLB.KRST=1. KRST is cleared after the kernel reset was executed. 0_B No action 1_B A kernel reset was requested
GRSTENx (x=0-3)	x+8	rw	**Enable for Global Module Reset Group x** 0_B Global module reset group x does not have any effect 1_B Global module reset group x results in a kernel reset
0	7:1, 31:12	r	**Reserved** Read as 0; should be written with 0.

25.4.7 Reset control register B

RST_CTRLB Offset address: 010_H

Reset control register B Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
STATCLR								0							
w								r							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															KRST
r															rwh

Field	Bits	Type	Description
KRST	0	rwh	**Kernel Reset** Request a kernel reset. The requested reset is executed if CTRLA.KRST=1 and CTRLB.KRST=1. KRST is cleared after the kernel reset was executed. 0_B No action 1_B A kernel reset was requested
STATCLR	31	w	**Kernel Reset Status Clear** Clears all status bits in RST_STAT when 1 is written. Read will return 0. 0_B No action 1_B Write with ´1´ clears bits STAT.GRSTx and bit STAT.KRST .
0	30:1	r	**Reserved** Read as 0; should be written with 0.

25.4.8 Reset status register

The reset status register contains the status bits for kernel reset (KRST) and the global module reset groups (GRSTx).

RST_STAT Offset address: 014_H

Reset status register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				GRST3	GRST2	GRST1	GRST0	0							KRST
r				rh	rh	rh	rh	r							rh

Field	Bits	Type	Description
KRST	0	rh	**Kernel Reset Status** Indicates an executed kernel reset. RST_STAT.KRST is set after the execution of a kernel reset in the same clock cycle in which the reset bits are cleared. Clear KRST by setting bit STATCLR in register RST_CTRLB.
GRSTx (x=0-3)	x+8	rh	**Status for Global Module Reset Group x** 0_B Reset was not triggered by Global Reset Group x 1_B Reset was triggered by Global Reset Group x
0	7:1, 31:12	r	**Reserved** Read as 0.

25.4.9 PROT register endinit

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner and in some PROT states, the secure master to update the PROT state as described in the PROT mechanism.

The PROTE register controls lock / unlock of the local Endinit (E) protected control registers.

PROTE Offset address: 018_H

PROT register endinit Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OWEN	ODEF	TAGID						PRSEN	PRS			VMEN	VM		
w	rw	rw						rw	rw			rw	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												SWEN	STATE		
r												w	rwh		

Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
VM	18:16	rw	**Virtual machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual machine definition enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.
PRSEN	23	rw	**Protection set definition enable for PROT owner** 0_B **Disable**: PRS is not part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG-ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; should be written with 0.

Table 1171 Access mode restrictions of PROTE sorted by descending priority

Mode name	Access mode		Description
write 1 to .**SWEN**	rwh	STATE	
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

25.4.10 PROT register safe endinit

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner and in some PROT states, the secure master to update the PROT state as described in the PROT mechanism.

The PROTSE register controls lock / unlock of the local Safe Endinit (SE) protected control registers.

PROTSE Offset address: $01C_H$

PROT register safe endinit Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OWEN	**ODEF**	**TAGID**						**PRSEN**	**PRS**			**VMEN**	**VM**		
w	rw	rw						rw	rw			rw	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												**SWEN**	**STATE**		
r												w	rwh		

Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write

(table continues...)

(continued)

Field	Bits	Type	Description
VM	18:16	rw	**Virtual machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual machine definition enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.
PRSEN	23	rw	**Protection set definition enable for PROT owner** 0_B **Disable**: PRS is not part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG-ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; should be written with 0.

Table 1172 Access mode restrictions of PROTSE sorted by descending priority

Mode name	Access mode		Description
write 1 to .**SWEN**	rwh	STATE	
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

25.4.11 Write access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

ACCEN_WRA Offset address: 020_H

Write access enable register A Application Reset value: $1000\ 0003_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access

25.4.12 Write access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

ACCEN_WRB Offset address: 024_H

Write access enable register B Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access
0	31:8	r	**Reserved** Read as 0; should be written with 0.

25.4.13 Read access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

ACCEN_RDA Offset address: 028_H

Read access enable register A Application Reset value: $FFFF\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access

25.4.14 Read access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

ACCEN_RDB Offset address: $02C_H$

Read access enable register B Application Reset value: $0000\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access
0	31:8	r	**Reserved** Read as 0; should be written with 0.

25.4.15 VM access enable register

This register defines which virtual machine encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each VMx.

ACCEN_VM Offset address: 030_H

VM access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for VM ID q** This bit enables read access to the access protected region for transactions with the VM ID q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for VM ID q** This bit enables write access to the access protected region for transactions with the VM ID q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

25.4.16 PRS access enable register

This register defines which protection register set encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each PRSx.

ACCEN_PRS Offset address: 034_H

PRS access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for PRS q** This bit enables read access to the access protected region for transactions with the PRS q. 0_B Disabled for read access 1_B Enabled for read access

(table continues...)

(continued)

Field	Bits	Type	Description
WRq (q=00-07)	q+16	rw	**Write access enable for PRS q** This bit enables write access to the access protected region for transactions with the PRS q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

25.4.17 SENT Fractional divider register

The SENT Fractional divider register controls the input clock $f_{fracdiv}$ of all SENT channels.

FDR Offset address: 040_H

SENT Fractional divider register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0						RESULT									
r						rh									

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DM		0				STEP									
rw		r				rw									

Field	Bits	Type	Description
STEP	9:0	rw	**Step Value** Reload or addition value for RESULT.
DM	15:14	rw	**Divider Mode** DM selects normal or fractional divider mode. 00_B Fractional divider is switched off; no output clock is generated. The Reset External Divider signal is 1. RESULT is not updated (default after System Reset). 01_B Normal Divider Mode selected. 10_B Fractional Divider Mode selected. 11_B Fractional divider is switched off; no output clock is generated. RESULT is not updated.
RESULT	25:16	rh	**Result Value** Bit field for the addition result.
0	13:10, 31:26	r	**Reserved** Read as 0; should be written with 0.

25.4.18 Interrupt overview register

INTOV Offset address: 044_H

Interrupt overview register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	IPC14	IPC13	IPC12	IPC11	IPC10	IPC9	IPC8	IPC7	IPC6	IPC5	IPC4	IPC3	IPC2	IPC1	IPC0
r	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
IPCy (y=0-14)	y	rh	**Interrupt Pending on Channel y** If any interrupt requested flag is set for channel y in register INTSTATy AND the referring interrupt is enabled in INTENx then IPCy is set. It is automatically reset if all flags in INTSTATy are cleared for which the referring interrupt is enabled in INTENx.
0	31:15	r	**Reserved** Read as 0.

25.4.19 Module time stamp register

The SENT Module time stamp register contains read-only information about the current time given in clock cycles generated from TPD since last reset while module was clocked. Writing to TPD clears this register.

TSR Offset address: 048_H

Module time stamp register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CTS															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CTS															
rh															

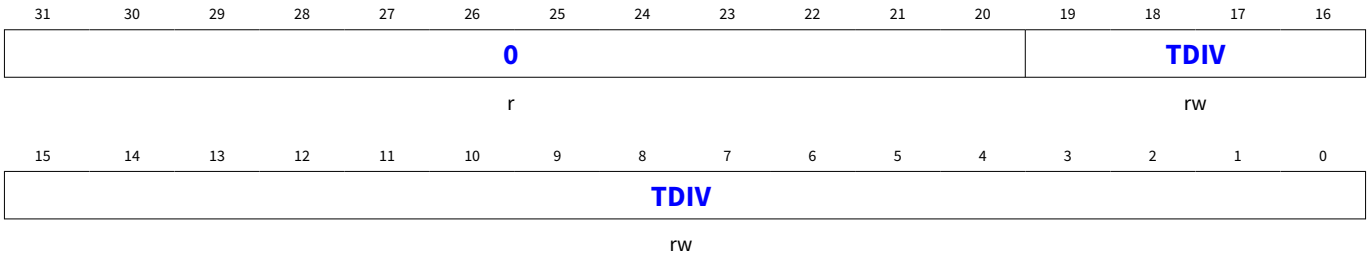
Field	Bits	Type	Description
CTS	31:0	rh	**Current Time Stamp for the Module** This bit field shows the current time stamp.

25.4.20 Time stamp predivider register

The SENT Module Time stamp predivider register contains the pre divider that defines the time resolution of TSR. Writing to TPD clears register TSR.

TPD Offset address: $04C_H$

Time stamp predivider register Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
TDIV	19:0	rw	**Divider Factor of Pre Divider for TSR** Divides $f_{fracdiv}$ by (TDIV + 1) and drives TSR.
0	31:20	r	**Reserved** Read as 0; should be written with 0.

25.4.21 Receive data register x

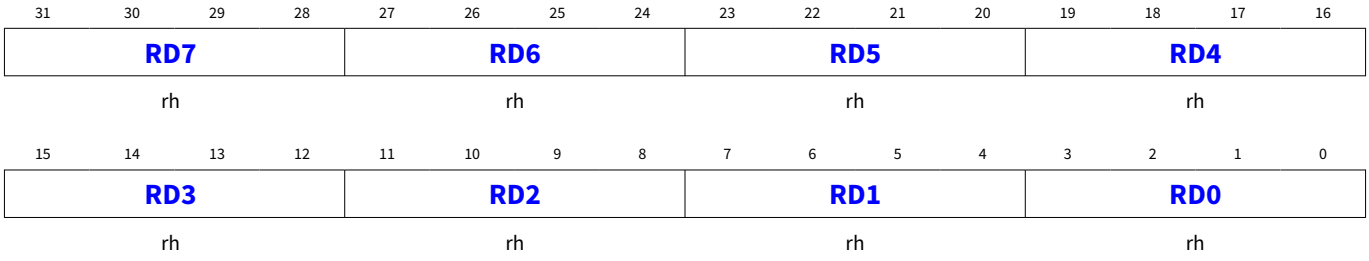
The Receive data registers RDRx for channel x shows the data content of a received data frame. Register VIEWx is used to sort the nibbles.

Register VIEW must be set up correctly to see all data nibbles of the frame. By default the application software set VIEW to 7654 3210_H.

The internal receive buffer is always cleared (0x0000 0000_H) at each frame start. Thus unused nibbles are always read as zero.

RDRx (x=0-14) Offset address: 080_H+x*4

Receive data register x Kernel Reset value: 0000 0000_H



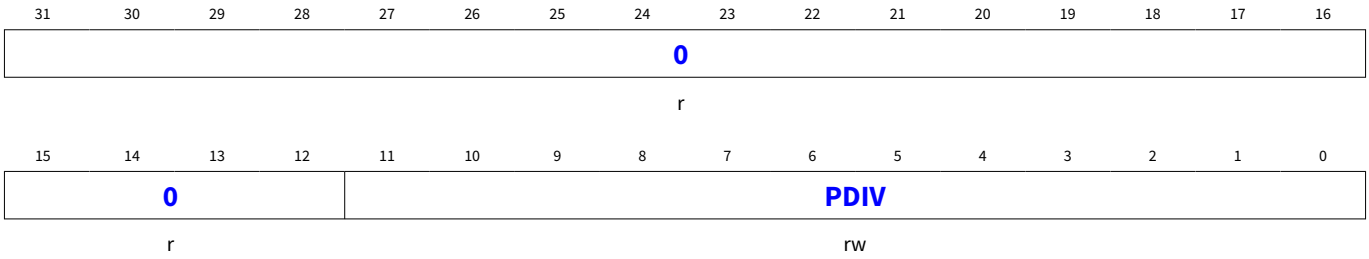
Field	Bits	Type	Description
RDy (y=0-7)	4*y+3:4*y	rh	**Receive Data Nibble y** RDy shows the nibble from the received frame that is sorted to this position. It can be selected by any of VIEWx.RDNPy (y = 0-7). By default all nibbles are sorted to RD0 as the reset value of VIEW is 0x0000 0000H. I.e. at the end of frame reception RD0 contains the last data nibble of the frame.

25.4.22 Channel pre divider register x

The Channel pre divider register CPDRx contains the pre divider that is related to the SENT channel baud rate. See

CPDRx (x=0-14) Offset address: 100_H+x*40_H

Channel pre divider register x Kernel Reset value: 0000 0000_H



Field	Bits	Type	Description
PDIV	11:0	rw	**Divider Factor of Pre Divider for Channel x** Divides $f_{fracdiv}$ by (PDIV + 1) and delivers f_{pdiv_x} to the Channel Fractional Divider. RCR.CEN must be cleared before changing CPDR.PDIV or CFDR.DIV.
0	31:12	r	**Reserved** Read as 0; should be written with 0.

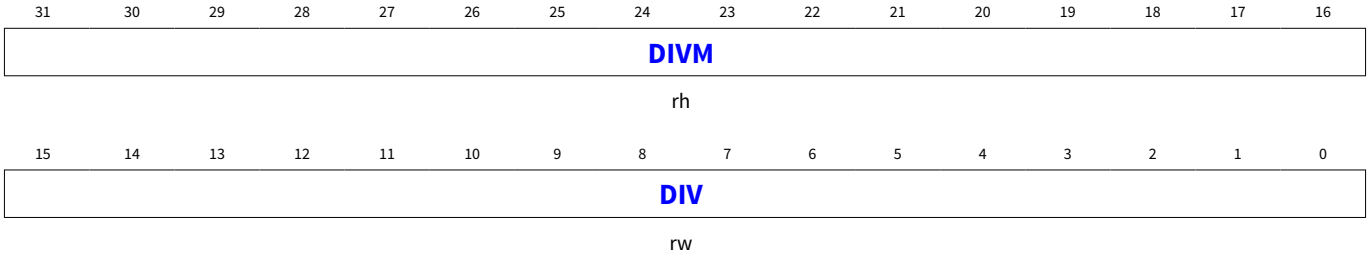
25.4.23 Channel fractional divider register x

The Channel fractional divider register CFDRx contains control bits/bit fields that are related to the SENT channel baud rate.

See in for a detailed description.

CFDRx (x=0-14) Offset address: 104_H+x*40_H

Channel fractional divider register x Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
DIV	15:0	rw	**Divider Value** Initial and reference divider value for the CFDR. DIV must be programmed > 0. If cleared, DIV becomes 1. If written, DIVM is updated automatically with the same value.RCR.CEN must be cleared before changing CPDR.PDIV or CFDR.DIV. DIV must be set in the range of [2200, 52428].
DIVM	31:16	rh	**Measured Divider Value** DIVM is automatically updated by HW to adjust the receiver frequency to the current sender frequency. This value is kept automatically in the range of 75% DIV < DIVM < 125% DIV Write data is ignored.

25.4.24 Receiver control register x

The Receiver control register RCRx contains control bits/bit fields that are related to the SENT receiver operation.

RCRx (x=0-14) Offset address: 108_H+x*40_H

Receiver control register x Reset values see: Table 1173

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0											FDFL	SUSEN	IDE	ESF	CRZ
r											rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FRL								CFC	CDIS	SCDIS	SDP	SNI	ACE	IEP	CEN
rw								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
CEN	0	rw	**Channel Enable** When CEN is set, the receiver of channel x is enabled. The internal receiver state machine can be initialized by switching the channel off and on. This does not change the current register content. 0_B channel x disabled (default) 1_B channel x enabled
IEP	1	rw	**Ignore End Pulse** When IEP is set, an end pulse is ignored. An end pulse can be generated in SPC mode or as pause pulse. For systems with an end pulse, during synchronize or re-synchronize of reception, if calibration pulses are detected one immediately following the other, the first calibration pulse shall be ignored as it may be a pause pulse with duration matching the calibration pulse range. 0_B End Pulse not ignored (default) 1_B End Pulse ignored
ACE	2	rw	**Alternative CRC Mode Enable** When ACE is set, the CRC is calculated in an alternative way for both: fast (signal) and slow (serial message) data path. If ESF is set, the standard 6 bit CRC is always used for the serial message and ACE is ignored. 0_B Serial CRC calculation as specified in SENT specification J2716 201604 (default) 1_B Alternative: 4 bit in parallel CRC calculation as used e.g. in hall sensor TLE4998C.
SNI	3	rw	**Status Nibble Included in CRC** When SNI is set, the status Nibble is included in (signal data) CRC. 0_B Status Nibble not included in CRC (default) 1_B Status Nibble included in CRC (as used e.g. in hall sensor TLE4998C).

(table continues...)

(continued)

Field	Bits	Type	Description
SDP	4	rw	**Serial Data Processing Mode** This bit switches automatic serial data processing on. 0_B Automatic Serial Data Processing is disabled. Status and Communication nibble can be read from RSRx for SW processing. (default) 1_B Automatic Serial Data Processing is enabled. Status and Communication nibble can be read from RSRx; Message ID, Serial Data and SCRC can be read from SDSx after serial data interrupt SDI is activated.
SCDIS	5	rw	**CRC for Serial Data Disabled Mode** This bit selects the CRC disabled mode. 0_B CRC is enabled (default) 1_B CRC is disabled CRC nibble can be read from SDSx. The CPU must perform the CRC on the current data by SW.
CDIS	6	rw	**CRC Disabled Mode** This bit selects the CRC disabled mode. 0_B CRC is enabled (default) 1_B CRC is disabled CRC nibble can be read from RSRx. The CPU must perform the CRC on the current data by SW.

(table continues...)

(continued)

Field	Bits	Type	Description
CFC	7	rw	**Consecutive Frame Check** This bit determines the way the most recently received frame buffer is indicated as valid. Note: If FDFL is set, CFC is ignored and the checks described here are not executed. 0_B Check against Past Sync Pulse The current Synchronization / Calibration Pulse is compared to the Synchronization / Calibration Pulse received immediately before. The whole frame is invalid if the Synchronization / Calibration Pulse length differs from the length of the Synchronization / Calibration Pulse before by more than 1.5625%. In this case of error, its length is not used as new reference. In case the check passes and no other error occurs the Frame Buffer is indicated valid (RSI) immediately after CRC calculation result is correct. Resynchronization: On the third successive calibration pulse error, the current calibration pulse value is considered as valid and the message accepted unless the message frame contains other errors. In both cases a receive data interrupt (RDI), or a referring error interrupt is issued. 1_B Check against Future Sync Pulse The current Synchronization / Calibration Pulse is compared with the Synchronization / Calibration Pulse received immediately after the current frame. The whole frame is invalid if the Synchronization / Calibration Pulse length differs from the length of the following Synchronization / Calibration Pulse by more than 1.5625%. In case the check passes and no other error occurs the Frame Buffer is indicated valid (RSI) immediately after the Synchronization / Calibration Pulse following the current frame. Resynchronization: In this case of error, the current length is used as new reference. Note: The whole frame can be indicated valid only after additionally measuring the Synchronization / Calibration Pulse of the successive frame.

(table continues...)

(continued)

Field	Bits	Type	Description
FRL	15:8	rw	**Frame Length** FRL determines the number of data nibbles per frame that the SENT channel x is setup for. Note that FRL does not include the Synchronization / Calibration Pulse, the Status and Communication nibble, the CRC nibble nor the additional zero length nibble that might be introduced by use of SPC If more than 8 nibbles are configured, please note: In addition to the receive success interrupt RSI at the successfully received end of a frame, a receive data interrupt RDI is issued each time 8 nibbles have been transferred to the Receive Data Register RDRx. At the end of a frame, RDI is issued if RSI is issued. If an error occurred, RDI is not set at the end of a frame. If no CRC has been received at the point in time where RDI is issued, the receive data interrupt is no indication whether or not the transfer was successful so far. A CRC Error Interrupt is issued at the end of the frame if Automatic CRC check is enabled and the CRC is wrong. 00_H No data nibble 01_H 1 data nibble 02_H 2 data nibbles 03_H 3 data nibbles … 07_H 7 data nibbles 08_H 8 data nibbles Maximum in normal length mode 09_H 9 data nibbles … FF_H 255 data nibbles
CRZ	16	rw	**CRC with Zero Nibble for Serial Data** This bit selects the CRC method. If CRZ is cleared, augmentation is selected, (i.e a ZERO NIBBLE is added at the end of CRC calculation (only in calculation)). E.g. as 7th nibble (in case of 6 data nibbles) The enhanced serial message (18 Frames, 6-bit CRC) is not controlled by CRZ but the 6-bit CRC is always augmented according to standard. 0_B Augmentation is selected for both 4-bit message CRC and the 4-bit CRC of the serial messages (legacy, 16 frames) (default) 1_B Augmentation is switched off for both 4-bit message CRC and the 4-bit CRC of the serial messages (legacy, 16 frames).
ESF	17	rw	**Enhanced Serial Frame Mode** This bit selects the serial frame structure. 0_B short (16 frames, 4 bit ID, 8 bit data, 4 bit CRC) 1_B enhanced (18 frames, 4 or 8 bit ID, 12 or 16 bit data, 6 bit CRC)
IDE	18	rw	**Ignore Drift Error Mode** This bit selects if drift errors lead to frame rejection and if an interrupt (INTSTAT.FDI) is generated. Used, if sensors are triggered by SPC. During a long pause period the accumulated drift could be more than 1.5625%. In this special case setting IDE is useful. 0_B Drift Errors enabled (default) 1_B Drift Errors disabled

(table continues...)

(continued)

Field	Bits	Type	Description
SUSEN	19	rw	**Suspend Enable** This bit makes it possible to set the SENT channel into Suspend Mode via OCDS (on chip debug support): Bit SUSEN is reset via OCDS Reset. 0_B An OCDS suspend trigger is ignored by this SENT channel. 1_B An OCDS suspend trigger disables the SENT channel: As soon as the SPC sender logic of the SENT channel becomes idle, the module is stopped while all registers of the channel stay readable. The receiver is stopped unconditionally. A partly received frame is discarded.
FDFL	20	rw	**Frequency Drift Check based on Frame Length** This bit is used for frames with pause pulse only. If set the drift error is not checked by HW. Instead, counter values DIVM and FRLEN are provided to the SW for checking the frequency drift. IEP (Pause Pulse expected) and IDE (no HW check of drift error) must always be set (=1) together with FDFL. Note: If FDFL is set, RCR.CFC is ignored and the checks described there are not executed. 0_B FDFL mode deactivated, RCR.CFC is valid (if IDE is cleared). 1_B FDFL mode is activated, IEP and IDE must be set too for proper function! RCR.CFC is ignored. If no other error occurs the Frame Buffer is indicated valid (RSI) immediately after the Pause Pulse belonging to the current frame.
0	31:21	r	**Reserved** Read as 0; should be written with 0.

Table 1173 Reset values of RCRx (x=0-14)

Reset type	Reset value	Note
Kernel Reset	0000 0000 0000 X000 0000 0000 0000 0000_B	
Debug Reset	XXXX XXXX XXXX 0XXX XXXX XXXX XXXX $XXXX_B$	

25.4.25 Receive status register x

The Receive status register provides the status information of channel x.

RSRx (x=0-14) Offset address: $10C_H+x*40_H$

Receive status register x Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FRLEN															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				SCN				0		CST		CRC			
r				r				r		r		r			

Field	Bits	Type	Description
CRC	3:0	r	**CRC** of last frame. CRC0 is on bit position 0.
CST	5:4	r	**Channel Status** CST shows the current status of channel x. 00_B STOP Channel is disabled and can be configured 01_B INITIALIZED Channel is configured and enabled and no Synchronization / Calibration Pulse was received since last enable. 10_B RUNNING one or more Synchronization / Calibration Pulses were received and Frequency Range or Frequency Drift not or no longer in range. Fallback status from SYNCHRONIZED. 11_B SYNCHRONIZED Frequency Range and Frequency Drift in range
SCN	11:8	r	**Status and Communication Nibble** of last frame. SCN0 is on bit position 8.
FRLEN	31:16	r	**Frame Length including Pause Nibble** of last frame. Bit FRLEN0 is on bit position 16. FRLEN is a 16 bit counter with saturation (stops at 0xFFFF / 65.535) and is driven by f_{pdiv}. This counter supports monitoring the following condition for messages with pause pulse and fixed message length: Ratio of calibration pulse to message length varies by > 1/64 or < -1/64 from one message to another. FRLEN is valid after RSI. See chapter “Support for Frequency Drift Analysis in Frames with Pause Pulse”.
0	7:6, 15:12	r	**Reserved** Read as 0; should be written with 0.

25.4.26 Serial data and status register x

The Serial (receive) data and status register provides the data and status information of channel x.

SDSx (x=0-14) Offset address: 110_H+x*40_H

Serial data and status register x Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CON	0	SCRC						MID							
r	r	r						r							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SD															
r															

Field	Bits	Type	Description
SD	15:0	r	**Serial Data** of last serial data frame. SD0 is on bit position 0. Usually all 16 data bits are used. If RCR.ESF is cleared 8 bits of data are available and bits [15:8] are zero. If RCR.ESF is set and if SDS.CON is cleared 12 bits of data are available and bits [15:12] are zero.

(table continues...)

(continued)

Field	Bits	Type	Description
MID	23:16	r	**Message ID** of last serial data frame. ID0 is on bit position 16. If RCR.ESF is cleared, or if SDS.CON is set, bits [23:20] are zero.
SCRC	29:24	r	**SCRC** CRC of last serial data frame. CRC0 is on position 24. If RCR.ESF is cleared, bits [29:28] are always zero.
CON	31	r	**Configuration bit** of last serial frame. 0_B 12-bit data and 8-bit message ID 1_B 16-bit data and 4-bit message ID
0	30	r	**Reserved** Read as 0; should be written with 0.

25.4.27 Input and output control register x

The Input and output control register IOCRx determines for the SENT channel x:
for the receiver:
- the alternate input
- the filter depth
- the input signal polarity

for the SPC Unit
- the trigger source
- the output signal polarity

IOCRx (x=0-14) Offset address: 114_H+x*40_H

Input and output control register x Reset values see: Table 1174

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TXM	RXM	TRM	CTR	EC								ETS			
rh	rh	rh	rw	rh								rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CFEG	CREG	FEG	REG	0	CEC	IIE	OIE	DEPTH				0		ALTI	
rw	rw	rh	rh	r	w	rw	rw	rw				r		rw	

Field	Bits	Type	Description
ALTI	1:0	rw	**Alternate Input Select** Selects the alternate input for channel y: 00_B Alternate Input 0 selected 01_B Alternate Input 1 selected 10_B Alternate Input 2 selected 11_B Alternate Input 3 selected

(table continues...)

(continued)

Field	Bits	Type	Description
DEPTH	7:4	rw	**Digital Glitch Filter Depth** DEPTH determines the number of port input samples clocked with f_{pdiv} that are taken into account for the calculation of the floating average. The higher DEPTH is chosen to be, the longer the glitches that are suppressed and the longer the delay of the input signal introduced by this filter. 0_H off, default 1_H 1 T_{pdiv} 2_H 2 … F_H 15
OIE	8	rw	**Output Inverter Enable Channel x** Selects the Pulse Polarity of the output of channel x 0_B Pulse polarity is active low 1_B Pulse polarity is active high
IIE	9	rw	**Input Inverter Enable Channel x** Selects the Pulse Polarity of the input of channel x 0_B Pulse polarity is active low 1_B Pulse polarity is active high
CEC	10	w	**Clear Edge Counter** If this bit is set, IOCR.EC is cleared. Always reads back as '0'.
REG	12	rh	**Rising Edge Glitch Flag for Channel x** Shows the status of the glitch detection of channel x REG is cleared by setting CREG. 0_B No Glitch detected on rising edge 1_B Glitch detected on rising edge
FEG	13	rh	**Falling Edge Glitch Flag for Channel x** Shows the status of the glitch detection of channel x FEG is cleared by setting CFEG. 0_B No Glitch detected on falling edge 1_B Glitch detected on falling edge
CREG	14	rw	**Clear Rising Edge Glitch Flag for Channel x** Clears the status flag REG CREG always read zero. 0_B REG is not cleared 1_B REG is cleared
CFEG	15	rw	**Clear Falling Edge Glitch Flag for Channel x** Clears the status flag FEG CFEG always read zero. 0_B FEG is not cleared 1_B FEG is cleared

(table continues...)

(continued)

Field	Bits	Type	Description
ETS	19:16	rw	**External Trigger Select** Selects the external trigger line if SCRx.TRIG is programmed to 11B. In some products, not all inputs are connected. Refer to device specific connectivity chapter. 0_H TRIG0 … F_H TRIG15
EC	27:20	rh	**Edge Counter** This bit field contains a counter with saturation (stops at 0xFF). It is incremented with any falling edge that appears on the input pin selected by IOCR.ALTI. Note that this holds true in all states (STOP, INITIALIZED, RUNNING, SYNCHRONIZED). It is intended for debugging, in particular to find a babbling idiot that sends before enabling the module.
CTR	28	rw	**Clear Trigger Monitor Flag for Channel x** Clears the status flag TRM CTR always read zero. Reset value of CTR is 0. 0_B TRM is not cleared 1_B TRM is cleared
TRM	29	rh	**Trigger Monitor Flag for Channel x** Shows the status of the trigger detection of channel x TRM is cleared by setting CTR. Reset value of TRM is 0. 0_B No Trigger detected 1_B Trigger detected (one or several)
RXM	30	rh	**Receive Monitor for Channel x** Shows the status of the receive signal of channel x after glitch filtering and inverted as specified by IIE. Reset value of RXM is X. 0_B Current signal is low. 1_B Current signal is high.
TXM	31	rh	**Transmit Monitor for Channel x** Shows the status of the transmit signal of channel x inverted as specified by OIE. Reset value of TXM is X. 0_B Current signal is low. 1_B Current signal is high.
0	3:2, 11	r	**Reserved** Read as 0; should be written with 0.

Table 1174 Reset values of IOCRx (x=0-14)

Reset type	Reset value	Note
Kernel Reset	X000 0000 0000 0000 0000 0000 0000 0000_B	

25.4.28 SPC control register x

The SPC control register SCR contains data to be transmitted during the sync pulse of the data frames. It contains as well the trigger control bits required for sending nibbles from the SENT module to the sensor / external SENT device.

The SPC Control Register is used to control the trigger mode and time base of the SPC channel transmission.

Data and the control bits are collected in this single register to ease transfer of multiple pulses in cases where dynamic switching of the trigger condition is required.

The SPC Control Register is used to control single pulse transfers only.

Thus it is possible to change the control settings of an individual channel from pulse to pulse as it is required in SPC mode “Bidirectional transmit”. Here it might be considered useful to change trigger mode between Mode 1 (immediately) and Mode 2 (falling edge of next Synchronization / Calibration Pulse with programmable delay).

In addition repeating transfers with the same control settings are supported. For a pulse transfer to be initiated a synchronization signal is sufficient and no further SW intervention is required. Here the data can be changed (“ID-Selection” Mode) or simply left constant (“Sync” Mode). Only a HW trigger needs to be set up.

In Mode 0, SPC is deactivated for this channel. A write access to SCRx does not initiate an SPC pulse transmission. All state transmitter machines are initialized.

In Mode 1, an SPC pulse is sent, each time SPC Control Register SCRx is written to. If a transfer is ongoing, the channel waits automatically until the internal transmission register is ready. Transmit Buffer Underflow bit TBIx is set after data has been completely transferred (PLEN exceeded) and no new data was written to SCRx. After the data was transferred to the internal transmission register, interrupt TDIx signals that a new value can be written. INTSTATx.TDIx must be cleared by SW. Independently from this interrupt pending bit, a new interrupt pulse is generated on each transfer of an SPC pulse. This mode is important for back to back transfers of several nibbles as in bidirectional SPC mode.

In Mode 2, an SPC pulse is sent, each time the first falling edge of any Synchronization / Calibration Pulse is received. In this mode, the programmable delay DEL is most useful. In SPC mode “Bidirectional Transmit” this mode is useful to synchronize the transmission. TDIx and TBIx work as in Mode 1.

In Mode 3, an SPC pulse is sent (with current DEL and PLEN) after each external trigger event (defined by IOCRx.ETS). This is most useful in SPC “sync” mode. TDIx and TBIx work as in Mode 1.

SCRx (x=0-14) Offset address: 118_H+x*40_H

SPC control register x Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TRQ	BASE	DEL						TRIG		PLEN					
r	rw	rw						rw		rw					

Field	Bits	Type	Description
PLEN	5:0	rw	**Pulse Length** Defines the length of the pulse in tick times. The time base is the measured tick time of the latest received frame if selected so by BASE. In case this measured tick time was invalid or not already available after enable of the channel, the nominal time base of the module is used. 00_H Pulse length is 0 ticks … $3F_H$ Pulse length is 63 ticks
TRIG	7:6	rw	**Trigger Source and Mode Selection** Selects the Trigger Source and Mode. The internal sender state machine can be initialized by switching the channel off (TRIG is cleared) and on. This does not change the current register content. 00_B No Pulse is generated, OFF When cleared, an ongoing transfer is stopped immediately and the transmit output is driven recessive. 01_B Pulse starts immediately (no auto repetition) 10_B Pulse starts each time the first falling edge of any Synchronization / Calibration Pulse is received (auto repetition on next Sync. / Cal. Pulses) 11_B Pulse starts after each external trigger event. (auto repetition on next trigger) IOCRx.ETS selects the source of this event.
DEL	13:8	rw	**Delay Length** Selects how long the SPC pulse is delayed after the trigger condition. The time base is the measured tick time of the latest received frame if selected so by BASE. In case this measured tick time was invalid or not already available after enable of the channel, the nominal time base of the module is used. 00_H Pulse is not delayed 01_H Pulse is delayed by 1 tick 02_H Pulse is delayed by 2 ticks … $3F_H$ Pulse is delayed by 63 ticks
BASE	14	rw	**Time Base** Selects the Pulse Time Base 0_B Pulse is based on measured frequency of last Synchronization/ Calibration Pulse 1_B Pulse is based on nominal frequency
TRQ	15	r	**Transfer Request in Progress** While an SPC Pulse is being sent this bit is set. Write access is ignored.
0	31:16	r	**Reserved** Read as 0; should be written with 0.

25.4.29 Receive data view register x

The Receive data view registers VIEWx stores the nibble pointers. They determine the sequence in which the received data nibbles are presented to the host. This reduces the SW effort to sort the nibbles. The data nibble that is received first in the received frame is moved to the location given in register VIEW.RDNP0, the second to VIEW.RDNP1 and so on until VIEW.RDNP7. If more than one VIEW.RDNPx point to a certain location in RDR, the last one will overwrite the previous ones. Example: two 12 bit values are transmitted. One with most significant nibble first and one with least significant nibble first. The frame looks like this: 456321_H. Note that the 1 is received as first data nibble and the 4 comes in as last data nibble. The actual signal values are $0x123_H$ and $0x456_H$. By using VIEWx this can be sorted out into two 16bit values by HW. In the example VIEWx would be set to: $73\ 654\ 012_H$. 73 is a dummy value and is not regarded if not more than 6 nibbles are received in a frame. The Register RDRx looks like this: $0x0456\ 0123_H$ to the host.

In the example RDR nibbles 3 and 7 contain $0x0000_B$ as the Receive Buffer is always cleared ($0x0000\ 0000_H$) before new data is received.

If a frame contains more than eight nibbles and the sorting can not be specified statically, VIEW can be set to e.g. $7654\ 3210_H$.

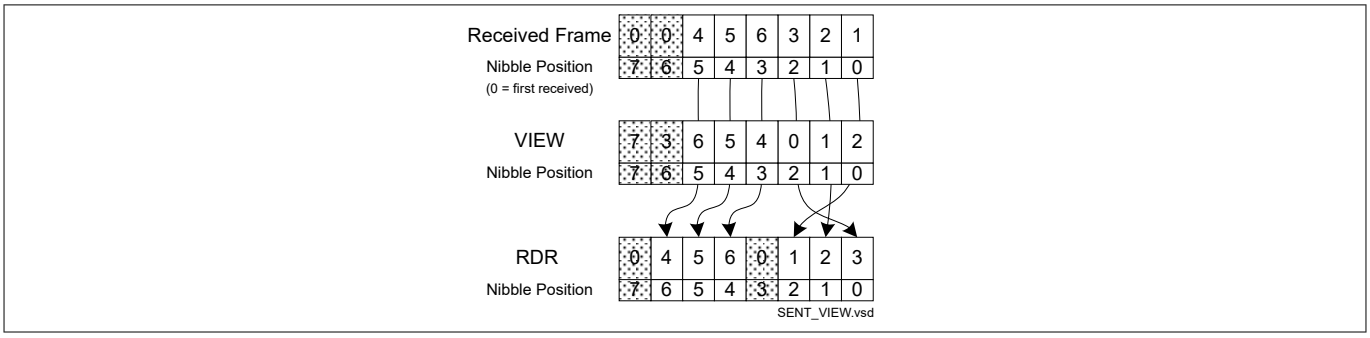


Figure 511 Functionality of VIEW Register

VIEWx (x=0-14) Offset address: $11C_H+x*40_H$

Receive data view register x Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	RDNP7			0	RDNP6			0	RDNP5			0	RDNP4		
r	rw			r	rw			r	rw			r	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	RDNP3			0	RDNP2			0	RDNP1			0	RDNP0		
r	rw			r	rw			r	rw			r	rw		

Field	Bits	Type	Description
RDNPy (y=0-7)	4*y+2:4*y	rw	**Receive Data Target Nibble Pointer y** RDNPy points to the Nibble in Receive Data Register RDRx where the nibble y from the received frame is sorted to. Nibble 0 is the first data nibble in the frame. It gets moved to the position defined in RDNP0. And on. RDNPy must be written before first frame reception. All RDNPy must have different values. (Higher RDNPy overwrite lower RDNPy.) 000_B Nibble 0 selected ... 111_B Nibble 7 selected

(table continues...)

(continued)

Field	Bits	Type	Description
0	3, 7, 11, 15, 19, 23, 27, 31	r	**Reserved** Read as 0; should be written with 0.

25.4.30 Interrupt status register x

The Interrupt status register INTSTATx contains status bits that show the status of any interrupt of SENT channel x.

The bits are set independently from the referring Interrupt Enable in Register INTENx. Thus they can be used as status bits as well e.g. by a SW based on polling.

On occurrence of FRI, FDI, CRCI, NVI and NNI error interrupts, the SENT state machine is reset and all corresponding received data until then is flushed and the state machine waits for a valid sync. Pulse in the input. Additionally, in case of CRCI, the RSI/RDI interrupts are skipped.

INTSTATx (x=0-14) Offset address: 120_H+x*40_H

Interrupt status register x Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		WDI	SCRI	SDI	WSI	CRCI	NVI	NNI	FDI	FRI	TBI	TDI	RBI	RDI	RSI
r		rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
RSI	0	rh	**Receive Success Interrupt Request Flag** This bit is set at the successfully received end of a frame. Depending on bit RCRx.CDIS this indicates a successful check of the CRC. This bit can be cleared by bit INTCLRx.RSI. This bit can be set by bit INTSETx.RSI. This bit is set independently from INTENx. 0_B No interrupt was requested since this bit was cleared the last time 1_B An interrupt was requested since this bit was cleared the last time

(table continues...)

(continued)

Field	Bits	Type	Description
RDI	1	rh	**Receive Data Interrupt Request Flag** RDI is activated when a received frame is moved to a Receive Data Register RDR. Both RDI and RSI will be issued together in normal use cases where the frame size is not bigger than 8 nibbles and CRC is correct or not checked (if RCRx.CDIS is cleared). For frames with more than 8 nibbles RDI is issued after 8 nibbles. After the last nibbles of the frame are received, a last RDI and an RSI are issued together. Note that this is true independently from RCR.FDFL. This bit can be cleared by bit INTCLRx.RDI. This bit can be set by bit INTSETx.RDI. This bit is set independently from INTENx. 0_B No interrupt was requested since this bit was cleared the last time 1_B An interrupt was requested since this bit was cleared the last time
RBI	2	rh	**Receive Buffer Overflow Interrupt Request Flag** This bit is set after a frame has been received while the old one was not read from RDRx. I.e. the kernel wants to set any of the two interrupts RSI and RDI and finds any of these two interrupts already set.The old data is overwritten by the new data. This bit is NOT cleared by reading RDRx. This bit can be cleared by bit INTCLRx.RBI. This bit can be set by bit INTSETx.RBI. This bit is set independently from INTENx. 0_B No interrupt was requested since this bit was cleared the last time 1_B An interrupt was requested since this bit was cleared the last time
TDI	3	rh	**Transfer Data Interrupt Request Flag** This bit is set after the trigger condition was detected. Data to be transferred has been moved internally. Thus a new value can be written to SCRx. This can be used for back to back transfers. This bit is automatically cleared by writing SCRx. This bit can be cleared by bit INTCLRx.TDI. This bit can be set by bit INTSETx.TDI. This bit is set independently from INTENx. 0_B No interrupt was requested since this bit was cleared the last time 1_B An interrupt was requested since this bit was cleared the last time
TBI	4	rh	**Transmit Buffer Underflow Interrupt Request Flag** This bit is set after data has been completely transferred (PLEN exceeded) and no new data was written to SCRx. This bit is NOT cleared by writing SCRx. This bit can be cleared by bit INTCLRx.TBI. This bit can be set by bit INTSETx.TBI. This bit is set independently from INTENx. 0_B No interrupt was requested since this bit was cleared the last time 1_B An interrupt was requested since this bit was cleared the last time

(table continues...)

(continued)

Field	Bits	Type	Description
FRI	5	rh	**Frequency Range Interrupt Request Flag** This bit is set after a Synchronization / Calibration pulse was received that deviates more than +/- 25% from the nominal value. The referring data is ignored. This bit can be cleared by bit INTCLRx.FRI. This bit can be set by bit INTSETx.FRI. This bit is set independently from INTENx. 0_B No interrupt was requested since this bit was cleared the last time 1_B An interrupt was requested since this bit was cleared the last time
FDI	6	rh	**Frequency Drift Interrupt Request Flag** This bit is set after a subsequent Synchronization / Calibration pulse was received that deviates more than 1.5625% (1/64) from its predecessor. (See RCR.CFC) This bit can be cleared by bit INTCLRx.FDI. This bit can be set by bit INTSETx.FDI. This bit is set independently from INTENx. 0_B No interrupt was requested since this bit was cleared the last time 1_B An interrupt was requested since this bit was cleared the last time
NNI	7	rh	**Number of Nibbles Wrong Request Flag** This bit is set after more nibbles have been received than expected or a Synchronization / Calibration Pulse is received too early thus too few nibbles have been received. This bit can be cleared by bit INTCLRx.NNI. This bit can be set by bit INTSETx.NNI. This bit is set independently from INTENx. 0_B No interrupt was requested since this bit was cleared the last time 1_B An interrupt was requested since this bit was cleared the last time
NVI	8	rh	**Nibbles Value out of Range Request Flag** This bit is set after a too long or too short nibble pulse has been received. I.e. value < 0 or value > 15. This bit can be cleared by bit INTCLRx.NVI. This bit can be set by bit INTSETx.NVI. This bit is set independently from INTENx. 0_B No interrupt was requested since this bit was cleared the last time 1_B An interrupt was requested since this bit was cleared the last time
CRCI	9	rh	**CRC Error Request Flag** This bit is set if the CRC fails. This bit can be cleared by bit INTCLRx.CRCI. This bit can be set by bit INTSETx.CRCI. This bit is set independently from INTENx. 0_B No interrupt was requested since this bit was cleared the last time 1_B An interrupt was requested since this bit was cleared the last time

(table continues...)

(continued)

Field	Bits	Type	Description
WSI	10	rh	**Wrong Status and Communication Nibble Error Request Flag** In Short Serial Frame Mode (RCR.ESF is cleared), this bit is set if the Status and Communication nibble shows a start bit in a frame other than frame number n x 16. In Enhanced Serial Frame Mode this bit is without function. This bit can be cleared by bit INTCLRx.WSI. This bit can be set by bit INTSETx.WSI. This bit is set independently from INTENx. 0_B No interrupt was requested since this bit was cleared the last time 1_B An interrupt was requested since this bit was cleared the last time
SDI	11	rh	**Serial Data Receive Interrupt Request Flag** This bit is set after all serial data bits have been received via the Status and Communication nibble. Depending on bit RCRx.SCDIS this indicates a successful check of the CRC. This bit can be cleared by bit INTCLRx.SDI. This bit can be set by bit INTSETx.SDI. This bit is set independently from INTENx. 0_B No interrupt was requested since this bit was cleared the last time 1_B An interrupt was requested since this bit was cleared the last time
SCRI	12	rh	**Serial Data CRC Error Request Flag** This bit is set if the CRC of the serial message fails. In Enhanced Serial Message Format, this includes a check of the Serial Communication Nibble for correct 0 values of bit 3 in frames 7, 13 and 18. This bit can be cleared by bit INTCLRx.SCRI. This bit can be set by bit INTSETx.SCRI. This bit is set independently from INTENx. 0_B No interrupt was requested since this bit was cleared the last time 1_B An interrupt was requested since this bit was cleared the last time
WDI	13	rh	**Watch Dog Error Request Flag** This bit is set if the Watch Dog Timer of the channel x expires. This bit can be cleared by bit INTCLRx.WDI. This bit can be set by bit INTSETx.WDI. This bit is set independently from INTENx. 0_B No interrupt was requested since this bit was cleared the last time 1_B An interrupt was requested since this bit was cleared the last time
0	31:14	r	**Reserved** Read as 0.

25.4.31 Interrupt set register x

The Interrupt set register INTSETx contains control bits that trigger an interrupt pulse for any interrupt of SENT channel x.

INTSETx (x=0-14) Offset address: 124_H+x*40_H

Interrupt set register x Kernel Reset value: $0000\ 0000_H$

25 Single Edge Nibble Transmission (SENT)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		WDI	SCRI	SDI	WSI	CRCI	NVI	NNI	FDI	FRI	TBI	TDI	RBI	RDI	RSI
r		w	w	w	w	w	w	w	w	w	w	w	w	w	w

Field	Bits	Type	Description
RSI	0	w	**Set Interrupt Request Flag RSI** Setting this bit set bit INTSTATx.RSI. Clearing this bit has no effect. Reading this bit returns always zero.
RDI	1	w	**Set Interrupt Request Flag RDI** Setting this bit set bit INTSTATx.RDI. Clearing this bit has no effect. Reading this bit returns always zero.
RBI	2	w	**Set Interrupt Request Flag RBI** Setting this bit set bit INTSTATx.RBI. Clearing this bit has no effect. Reading this bit returns always zero.
TDI	3	w	**Set Interrupt Request Flag TDI** Setting this bit set bit INTSTATx.TDI. Clearing this bit has no effect. Reading this bit returns always zero.
TBI	4	w	**Set Interrupt Request Flag TBI** Setting this bit set bit INTSTATx.TBI. Clearing this bit has no effect. Reading this bit returns always zero.
FRI	5	w	**Set Interrupt Request Flag FRI** Setting this bit set bit INTSTATx.FRI. Clearing this bit has no effect. Reading this bit returns always zero.
FDI	6	w	**Set Interrupt Request Flag FDI** Setting this bit set bit INTSTATx.FDI. Clearing this bit has no effect. Reading this bit returns always zero.
NNI	7	w	**Set Interrupt Request Flag NNI** Setting this bit set bit INTSTATx.NNI. Clearing this bit has no effect. Reading this bit returns always zero.

(table continues...)

(continued)

Field	Bits	Type	Description
NVI	8	w	**Set Interrupt Request Flag NVI** Setting this bit set bit INTSTATx.NVI. Clearing this bit has no effect. Reading this bit returns always zero.
CRCI	9	w	**Set Interrupt Request Flag CRCI** Setting this bit set bit INTSTATx.CRCI. Clearing this bit has no effect. Reading this bit returns always zero.
WSI	10	w	**Set Interrupt Request Flag WSI** Setting this bit set bit INTSTATx.WSI. Clearing this bit has no effect. Reading this bit returns always zero.
SDI	11	w	**Set Interrupt Request Flag SDI** Setting this bit set bit INTSTATx.SDI. Clearing this bit has no effect. Reading this bit returns always zero.
SCRI	12	w	**Set Interrupt Request Flag SCRI** Setting this bit set bit INTSTATx.SCRI. Clearing this bit has no effect. Reading this bit returns always zero.
WDI	13	w	**Set Interrupt Request Flag WDI** Setting this bit set bit INTSTATx.WDI. Clearing this bit has no effect. Reading this bit returns always zero.
0	31:14	r	**Reserved** Read as 0; should be written with 0.

25.4.32 Interrupt clear register x

The Interrupt clear register INTCLRx contains control bits that clear the status of any interrupt of SENT channel x.

INTCLRx (x=0-14) Offset address: 128_H+x*40_H

Interrupt clear register x Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		WDI	SCRI	SDI	WSI	CRCI	NVI	NNI	FDI	FRI	TBI	TDI	RBI	RDI	RSI
r		w	w	w	w	w	w	w	w	w	w	w	w	w	w

Field	Bits	Type	Description
RSI	0	w	**Clear Interrupt Request Flag RSI** Setting this bit clears bit INTSTATx.RSI. Clearing this bit has no effect. Reading this bit returns always zero.
RDI	1	w	**Clear Interrupt Request Flag RDI** Setting this bit clears bit INTSTATx.RDI. Clearing this bit has no effect. Reading this bit returns always zero.
RBI	2	w	**Clear Interrupt Request Flag RBI** Setting this bit clears bit INTSTATx.RBI. Clearing this bit has no effect. Reading this bit returns always zero.
TDI	3	w	**Clear Interrupt Request Flag TDI** Setting this bit clears bit INTSTATx.TDI. Clearing this bit has no effect. Reading this bit returns always zero.
TBI	4	w	**Clear Interrupt Request Flag TBI** Setting this bit clears bit INTSTATx.TBI. Clearing this bit has no effect. Reading this bit returns always zero.
FRI	5	w	**Clear Interrupt Request Flag FRI** Setting this bit clears bit INTSTATx.FRI. Clearing this bit has no effect. Reading this bit returns always zero.
FDI	6	w	**Clear Interrupt Request Flag FDI** Setting this bit clears bit INTSTATx.FDI. Clearing this bit has no effect. Reading this bit returns always zero.
NNI	7	w	**Clear Interrupt Request Flag NNI** Setting this bit clears bit INTSTATx.NNI. Clearing this bit has no effect. Reading this bit returns always zero.
NVI	8	w	**Clear Interrupt Request Flag NVI** Setting this bit clears bit INTSTATx.NVI. Clearing this bit has no effect. Reading this bit returns always zero.
CRCI	9	w	**Clear Interrupt Request Flag CRCI** Setting this bit clears bit INTSTATx.CRCI. Clearing this bit has no effect. Reading this bit returns always zero.

(table continues...)

(continued)

Field	Bits	Type	Description
WSI	10	w	**Clear Interrupt Request Flag WSI** Setting this bit clears bit INTSTATx.WSI. Clearing this bit has no effect. Reading this bit returns always zero.
SDI	11	w	**Clear Interrupt Request Flag SDI** Setting this bit clears bit INTSTATx.SDI. Clearing this bit has no effect. Reading this bit returns always zero.
SCRI	12	w	**Clear Interrupt Request Flag SCRI** Setting this bit clears bit INTSTATx.SCRI. Clearing this bit has no effect. Reading this bit returns always zero.
WDI	13	w	**Clear Interrupt Request Flag WDI** Setting this bit clears bit INTSTATx.WDI. Clearing this bit has no effect. Reading this bit returns always zero.
0	31:14	r	**Reserved** Read as 0; should be written with 0.

25.4.33 Interrupt enable register x

The Interrupt enable register INTENx contains control bits that enable the interrupt source of any interrupt of SENT channel x.

The Interrupt Status bits in register INTSTATx are set independently from the Interrupt Enable in Register INTENx.

INTENx (x=0-14) Offset address: $12C_H+x*40_H$

Interrupt enable register x Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		WDI	SCRI	SDI	WSI	CRCI	NVI	NNI	FDI	FRI	TBI	TDI	RBI	RDI	RSI
r		rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RSI	0	rw	**Enable Interrupt Request RSI** 0_B No interrupt request can be generated for this source 1_B An interrupt request can be generated for this source

(table continues...)

(continued)

Field	Bits	Type	Description
RDI	1	rw	**Enable Interrupt Request RDI** 0_B No interrupt request can be generated for this source 1_B An interrupt request can be generated for this source
RBI	2	rw	**Enable Interrupt Request RBI** 0_B No interrupt request can be generated for this source 1_B An interrupt request can be generated for this source
TDI	3	rw	**Enable Interrupt Request TDI** 0_B No interrupt request can be generated for this source 1_B An interrupt request can be generated for this source
TBI	4	rw	**Enable Interrupt Request TBI** 0_B No interrupt request can be generated for this source 1_B An interrupt request can be generated for this source
FRI	5	rw	**Enable Interrupt Request FRI** 0_B No interrupt request can be generated for this source 1_B An interrupt request can be generated for this source
FDI	6	rw	**Enable Interrupt Request FDI** 0_B No interrupt request can be generated for this source 1_B An interrupt request can be generated for this source
NNI	7	rw	**Enable Interrupt Request NNI** 0_B No interrupt request can be generated for this source 1_B An interrupt request can be generated for this source
NVI	8	rw	**Enable Interrupt Request NVI** 0_B No interrupt request can be generated for this source 1_B An interrupt request can be generated for this source
CRCI	9	rw	**Enable Interrupt Request CRCI** 0_B No interrupt request can be generated for this source 1_B An interrupt request can be generated for this source
WSI	10	rw	**Enable Interrupt Request WSI** 0_B No interrupt request can be generated for this source 1_B An interrupt request can be generated for this source
SDI	11	rw	**Enable Interrupt Request SDI** 0_B No interrupt request can be generated for this source 1_B An interrupt request can be generated for this source
SCRI	12	rw	**Enable Interrupt Request SCRI** 0_B No interrupt request can be generated for this source 1_B An interrupt request can be generated for this source
WDI	13	rw	**Enable Interrupt Request WDI** 0_B No interrupt request can be generated for this source 1_B An interrupt request can be generated for this source

(table continues...)

(continued)

Field	Bits	Type	Description
0	31:14	r	**Reserved** Read as 0; should be written with 0.

25.4.34 Interrupt node pointer register x

The Interrupt node pointer register INPx contains the node pointers of SENT channel x.
Node Pointer ERRI is one single node pointer for the following error interrupts:

- FRI
- FDI
- NNI
- NVI
- CRCI
- WSI
- SCRI

INPx (x=0-14) Offset address: 130_H+x*40_H

Interrupt node pointer register x Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
WDI				SDI				ERRI				TBI			
rw				rw				rw				rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TDI				RBI				RDI				RSI			
rw				rw				rw				rw			

Field	Bits	Type	Description
RSI	3:0	rw	**Interrupt Node Pointer for Interrupt RSI** This bit field defines the interrupt node, that is requested due to the set condition for bit INTSTATx.RSI (if enabled by bit INTENx.RSI). 0_H Trigger Output TRIGO0 is selected … F_H Trigger Output TRIGO15 is selected
RDI	7:4	rw	**Interrupt Node Pointer for Interrupt RDI** This bit field defines the interrupt node, that is requested due to the set condition for bit INTSTATx.RDI (if enabled by bit INTENx.RDI). For bit field definition, see RSI.
RBI	11:8	rw	**Interrupt Node Pointer for Interrupt RBI** This bit field defines the interrupt node, that is requested due to the set condition for bit INTSTATx.RBI (if enabled by bit INTENx.RBI). For bit field definition, see RSI.

(table continues...)

(continued)

Field	Bits	Type	Description
TDI	15:12	rw	**Interrupt Node Pointer for Interrupt TDI** This bit field defines the interrupt node, that is requested due to the set condition for bit INTSTATx.TDI (if enabled by bit INTENx.TDI). For bit field definition, see RSI.
TBI	19:16	rw	**Interrupt Node Pointer for Interrupt TBI** This bit field defines the interrupt node, that is requested due to the set condition for bit INTSTATx.TBI (if enabled by bit INTENx.TBI). For bit field definition, see RSI.
ERRI	23:20	rw	**Interrupt Node Pointer for Interrupt FRI, FDI, NNI, NVI, CRCI, WSI, SCRI** This bit field defines the interrupt node, that is requested due to the set condition for bit INTSTATx.FRI (if enabled by bit INTENx.FRI) or INTSTATx.FDI (if enabled by bit INTENx.FDI) or INTSTATx.NNI (if enabled by bit INTENx.NNI) or INTSTATx.NVI (if enabled by bit INTENx.NVI) or INTSTATx.CRCI (if enabled by bit INTENx.CRCI) or INTSTATx.WSI (if enabled by bit INTENx.WSI) or INTSTATx.SCRI (if enabled by bit INTENx.SCRI) For bit field definition, see RSI.
SDI	27:24	rw	**Interrupt Node Pointer for Interrupt SDI** This bit field defines the interrupt node, that is requested due to the set condition for bit INTSTATx.SDI (if enabled by bit INTENx.SDI). For bit field definition, see RSI.
WDI	31:28	rw	**Interrupt Node Pointer for Interrupt WDI** This bit field defines the interrupt node, that is requested due to the set condition for bit INTSTATx.WDI (if enabled by bit INTENx.WDI). For bit field definition, see RSI.

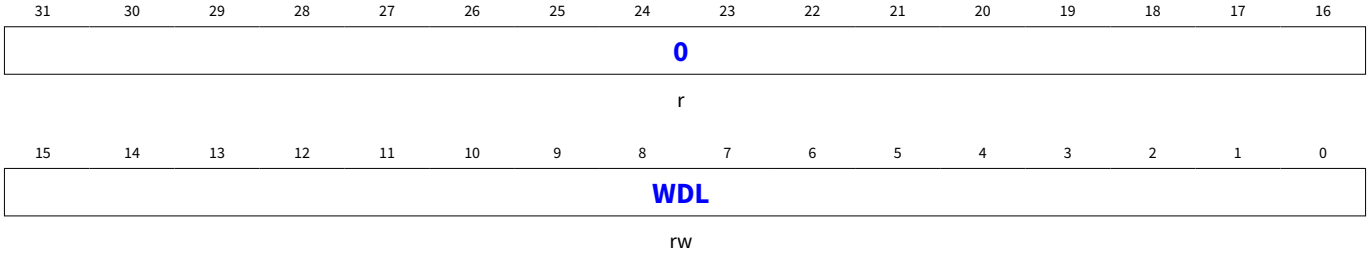
25.4.35 Watch dog timer register x

The Watch dog timer register WDTx contains the time out value for channel x. The internal Watch Dog timer is cleared and started automatically each time an RDI (Receive Data Interrupt Request Flag) is set in the INTSTAT of the referring channel and also on any write to WDL that sets WDL>0. Normally the WDTx.WDL is configured only once at module setup. However, if WDTx.WDL is written again with a value larger than 0 during frame reception, the internal Watch Dog timer is NOT cleared and restarted. Instead, the Watch Dog Timer keeps on counting and comparing to the new value. This is not recommended but offers flexibility in exceptional cases. Still, if WDTx.WDL is written with 0 during frame reception, the internal Watch Dog timer is cleared and stopped.

The value entered here defines the time in multiples of Ttick_x (defined in CFDR) from the last RDI on channel x. The internal Watch Dog Timer is compared to WDLx. A match triggers interrupt WDI and stops the internal Watch Dog Counter x. After such a match, the Watch Dog Timer will be cleared and started automatically with clearing bit WDI (by setting INTCLR.WDI). I.e. the Watch Dog Counter is running only if interrupt flag WDI is cleared. If WDL is cleared, the WDTx is stopped/disabled.

WDTx (x=0-14) Offset address: 134_H+x*40_H

Watch dog timer register x Kernel Reset value: $0000\ 0000_H$



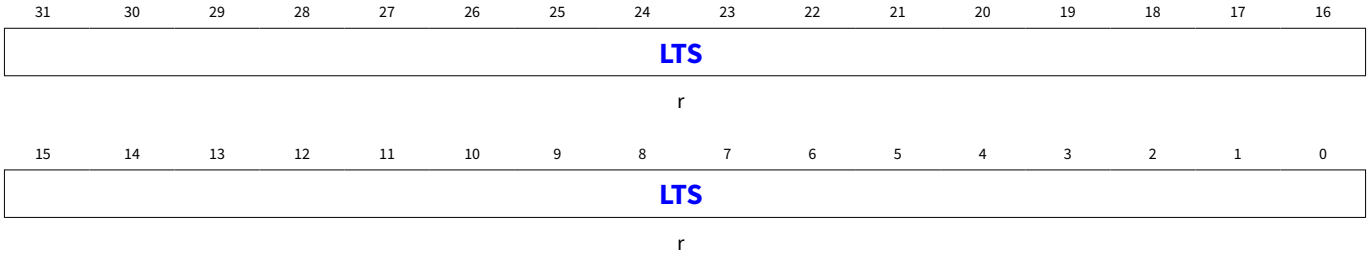
Field	Bits	Type	Description
WDL	15:0	rw	**Watch Dog Timer Limit** for channel x .
0	31:16	r	**Reserved** Read as 0; should be written with 0.

25.4.36 Receive time stamp register x

The SENT Channel x Receive time stamp register contains read-only information about the time the last frame for channel x was received. Time is captured with the second falling edge, i.e. at the Status/Communication data pulse.

RTSx (x=0-14) Offset address: $A80_H+x*4$

Receive time stamp register x Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
LTS	31:0	r	**Last Receive Time Stamp for Channel x** This bit field shows the time stamp of the last frame on channel x.

25.5 Debug information

This section is not applicable for the module being described.

25.6 References

SAE Standard “SENT specification J2716 201604”.

25.7 SENT revision history

Reference	Description of change(s)
Date range: 2024-08-17 to 2025-01-08	
–	• No visible changes

25.8 TC4Dx SENT information

25.8.1 TC4Dx SENT configuration

There are no deviations from the generic specification.

25.8.2 TC4Dx SENT features

Table 1175 Feature support in TC4Dx

Parameter	Number of instances	Comment
GTM trigger input	0	No GTM available in TC4Dx
eGTM trigger input	3	

25.8.3 TC4Dx SENT functional description

In TC4Dx device, the SENT module is connected to ComPB bus interface.

25.8.4 TC4Dx SENT registers

25.8.4.1 Register address space - SENT

Table 1176 Registers address space - SENT

Module	Base address	End address	Note
SENT0	$F4403000_H$	$F4403BFF_H$	FPI slave interface
SENT1	$F4404000_H$	$F4404BFF_H$	FPI slave interface

25.8.4.2 Register overview - access mode glossary

Table 1177 Register overview - access mode glossary

Keyword	Description
E	Access protection using PROT register SENT0_PROTE or SENT1_PROTE .
SE	Access protection using PROT register SENT0_PROTSE or SENT1_PROTSE .
APU-P	Protection group consisting of registers SENT0_ACCEN_WRA , SENT0_ACCEN_WRB , SENT0_ACCEN_RDA , SENT0_ACCEN_RDB , SENT0_ACCEN_VM , SENT0_ACCEN_PRS or SENT1_ACCEN_WRA , SENT1_ACCEN_WRB , SENT1_ACCEN_RDA , SENT1_ACCEN_RDB , SENT1_ACCEN_VM , SENT1_ACCEN_PRS .
P	Access protection using *APU-P* registers.
BE	Always returns a Bus Error.
U	No access restrictions.
PROT	Access restrictions as defined in the PROT register access rules.

25.8.4.3 Register overview - SENT0 (ascending offset address)

Table 1178 Register overview - SENT0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
SENT0_CLC	Clock control register	000_H	P	P, SV, E	Application Reset	4807
SENT0_OCS	OCDS control and status register	004_H	P	SV, P	Debug Reset	4808
SENT0_ID	Module identification register	008_H	P	BE	PowerOn Reset	4809
SENT0_RST_CTRLA	Reset control register A	$00C_H$	P	P, SV, E	Application Reset	4809
SENT0_RST_CTRLB	Reset control register B	010_H	P	P, SV, E	Application Reset	4810
SENT0_RST_STAT	Reset status register	014_H	P	BE	Application Reset	4811
SENT0_PROTE	PROT register endinit	018_H	U	SV, PROT	Application Reset	4811

(table continues...)

Table 1178 (continued) Register overview - SENT0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
SENT0_PROTSE	PROT register safe endinit	$01C_H$	U	SV, PROT	Application Reset	4813
SENT0_ACCEN_WRA	Write access enable register A	020_H	U	SE, SV	Application Reset	4814
SENT0_ACCEN_WRB	Write access enable register B	024_H	U	SE, SV	Application Reset	4815
SENT0_ACCEN_RDA	Read access enable register A	028_H	U	SE, SV	Application Reset	4815
SENT0_ACCEN_RDB	Read access enable register B	$02C_H$	U	SE, SV	Application Reset	4816
SENT0_ACCEN_VM	VM access enable register	030_H	U	SE, SV	Application Reset	4816
SENT0_ACCEN_PRS	PRS access enable register	034_H	U	SE, SV	Application Reset	4817
SENT0_FDR	SENT Fractional divider register	040_H	P	E, P	Kernel Reset	4818
SENT0_INTOV	Interrupt overview register	044_H	P	BE	Kernel Reset	4818
SENT0_TSR	Module time stamp register	048_H	P	BE	Kernel Reset	4819
SENT0_TPD	Time stamp predivider register	$04C_H$	P	E, P	Kernel Reset	4819
SENT0_RDRx (x=0-14)	Receive data register x	080_H+x*4	P	BE	Kernel Reset	4820
SENT0_CPDRx (x=0-14)	Channel pre divider register x	100_H+x*40_H	P	P	Kernel Reset	4820
SENT0_CFDRx (x=0-14)	Channel fractional divider register x	104_H+x*40_H	P	P	Kernel Reset	4821
SENT0_RCRx (x=0-14)	Receiver control register x	108_H+x*40_H	P	P	4822	4822
SENT0_RSRx (x=0-14)	Receive status register x	$10C_H$+x*40_H	P	BE	Kernel Reset	4826
SENT0_SDSx (x=0-14)	Serial data and status register x	110_H+x*40_H	P	BE	Kernel Reset	4827
SENT0_IOCRx (x=0-14)	Input and output control register x	114_H+x*40_H	P	P	Kernel Reset	4828
SENT0_SCRx (x=0-14)	SPC control register x	118_H+x*40_H	P	P	Kernel Reset	4831
SENT0_VIEWx (x=0-14)	Receive data view register x	$11C_H$+x*40_H	P	P	Kernel Reset	4833

(table continues...)

Table 1178 (continued) Register overview - SENT0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
SENT0_INTSTATx (x=0-14)	Interrupt status register x	120_H+x* 40_H	P	BE	Kernel Reset	4834
SENT0_INTSETx (x=0-14)	Interrupt set register x	124_H+x* 40_H	P	P	Kernel Reset	4837
SENT0_INTCLRx (x=0-14)	Interrupt clear register x	128_H+x* 40_H	P	P	Kernel Reset	4839
SENT0_INTENx (x=0-14)	Interrupt enable register x	$12C_H$+x* 40_H	P	P	Kernel Reset	4841
SENT0_INPx (x=0-14)	Interrupt node pointer register x	130_H+x* 40_H	P	P	Kernel Reset	4843
SENT0_WDTx (x=0-14)	Watch dog timer register x	134_H+x* 40_H	P	P	Kernel Reset	4844
SENT0_RTSx (x=0-14)	Receive time stamp register x	$A80_H$+x* 4	P	BE	Kernel Reset	4845

25.8.4.4 Register overview - SENT1 (ascending offset address)

Table 1179 Register overview - SENT1 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
SENT1_CLC	Clock control register	000_H	P	P, SV, E	Application Reset	4807
SENT1_OCS	OCDS control and status register	004_H	P	SV, P	Debug Reset	4808
SENT1_ID	Module identification register	008_H	P	BE	PowerOn Reset	4809
SENT1_RST_CTRLA	Reset control register A	$00C_H$	P	P, SV, E	Application Reset	4809
SENT1_RST_CTRLB	Reset control register B	010_H	P	P, SV, E	Application Reset	4810
SENT1_RST_STAT	Reset status register	014_H	P	BE	Application Reset	4811
SENT1_PROTE	PROT register endinit	018_H	U	SV, PROT	Application Reset	4811
SENT1_PROTSE	PROT register safe endinit	$01C_H$	U	SV, PROT	Application Reset	4813
SENT1_ACCEN_WRA	Write access enable register A	020_H	U	SE, SV	Application Reset	4814
SENT1_ACCEN_WRB	Write access enable register B	024_H	U	SE, SV	Application Reset	4815

(table continues...)

Table 1179 (continued) Register overview - SENT1 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
SENT1_ACCEN_RDA	Read access enable register A	028_H	U	SE, SV	Application Reset	4815
SENT1_ACCEN_RDB	Read access enable register B	$02C_H$	U	SE, SV	Application Reset	4816
SENT1_ACCEN_VM	VM access enable register	030_H	U	SE, SV	Application Reset	4816
SENT1_ACCEN_PRS	PRS access enable register	034_H	U	SE, SV	Application Reset	4817
SENT1_FDR	SENT Fractional divider register	040_H	P	E, P	Kernel Reset	4818
SENT1_INTOV	Interrupt overview register	044_H	P	BE	Kernel Reset	4818
SENT1_TSR	Module time stamp register	048_H	P	BE	Kernel Reset	4819
SENT1_TPD	Time stamp predivider register	$04C_H$	P	E, P	Kernel Reset	4819
SENT1_RDRx (x=0-14)	Receive data register x	080_H+x*4	P	BE	Kernel Reset	4820
SENT1_CPDRx (x=0-14)	Channel pre divider register x	100_H+x*40_H	P	P	Kernel Reset	4820
SENT1_CFDRx (x=0-14)	Channel fractional divider register x	104_H+x*40_H	P	P	Kernel Reset	4821
SENT1_RCRx (x=0-14)	Receiver control register x	108_H+x*40_H	P	P	4822	4822
SENT1_RSRx (x=0-14)	Receive status register x	$10C_H$+x*40_H	P	BE	Kernel Reset	4826
SENT1_SDSx (x=0-14)	Serial data and status register x	110_H+x*40_H	P	BE	Kernel Reset	4827
SENT1_IOCRx (x=0-14)	Input and output control register x	114_H+x*40_H	P	P	Kernel Reset	4828
SENT1_SCRx (x=0-14)	SPC control register x	118_H+x*40_H	P	P	Kernel Reset	4831
SENT1_VIEWx (x=0-14)	Receive data view register x	$11C_H$+x*40_H	P	P	Kernel Reset	4833
SENT1_INTSTATx (x=0-14)	Interrupt status register x	120_H+x*40_H	P	BE	Kernel Reset	4834
SENT1_INTSETx (x=0-14)	Interrupt set register x	124_H+x*40_H	P	P	Kernel Reset	4837
SENT1_INTCLRx (x=0-14)	Interrupt clear register x	128_H+x*40_H	P	P	Kernel Reset	4839

(table continues...)

Table 1179 (continued) Register overview - SENT1 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
SENT1_INTENx (x=0-14)	Interrupt enable register x	$12C_H$+x*40_H	P	P	Kernel Reset	4841
SENT1_INPx (x=0-14)	Interrupt node pointer register x	130_H+x*40_H	P	P	Kernel Reset	4843
SENT1_WDTx (x=0-14)	Watch dog timer register x	134_H+x*40_H	P	P	Kernel Reset	4844
SENT1_RTSx (x=0-14)	Receive time stamp register x	$A80_H$+x*4	P	BE	Kernel Reset	4845

25.8.4.5 Device specific registers

There are no device specific register changes.

25.8.5 TC4Dx SENT connectivity

The table below lists interfaces to and from this functional block to other blocks in the device.

Table 1180 List of SENT interface signals

Interface signals	I/O	Description
CLOCK_SENT_fSPB	In	SPB clock input
SENT0_IR_SR[9:0]	Out	Service Request
SENT1_IR_SR[9:0]	Out	Service Request
EGTM_SENT0_TRIG[2:0]	In	eGTM timer output vector
EGTM_SENT1_TRIG[2:0]	In	eGTM timer output vector
PORTS_SENT0_Input[14:0]	In	Input signal. The number indicates the channel.
PORTS_SENT1_Input[14:0]	In	Input signal. The number indicates the channel.
SENT0_PORTS_SPC[14:0]	Out	Transmit output. The number indicates the channel.
SENT1_PORTS_SPC[14:0]	Out	Transmit output. The number indicates the channel.

25.8.6 TC4Dx SENT revision history

Reference	Description of change(s)
Date range: 2024-08-17 to 2025-04-14	
TC4Dx SENT registers	• No deviation statement removed (no editorial change)

26 Peripheral Sensor Interface (PSI5)

The PSI5 module communicates with the external world through one I/O line for each channel. The PSI5RXx lines are the receive data input signals. The transmit lines PSI5TXx are on separate GPIO ports. The receive and transmit paths are always routed to two different ports.

26.1 Feature list

- According to PSI5 protocol specification V2.1 and extended powertrain substandard
- Flexibility in configuration and monitoring for software
  - Supports both Asynchronous and Synchronous data transmission modes (controlled by microcontroller in Synchronous mode)
  - Support of Enhanced Serial Messages according to SENT SAE J2716 JAN 2010
  - Three 64-bit downstream data registers for data input, data preparation and data output
  - Sticky interrupt flags, error interrupt optional (default disabled)
  - One sum interrupt for general events
  - One sum interrupt on selectable errors
  - Optional output inversion for use of external open drain transistor
  - Optional input inversion for use of external open transistor for level shifting
- Receive PSI5 frames based on the protocol standard
  - Data rates of 125 kbit/s and 189 kbit/s are supported
  - Independent and parallel PSI5 channels implemented
  - Supports 6 sensor slots per channel
  - Decoding Manchester protocol
  - Error recognition in Manchester code
  - CRC check of received sensor data implemented but CRC code transparent
  - 24-bit time stamp (resolution: 1 µs)
- Transmit PSI5 frames based on the protocol standard
  - Support of ECU to Sensor communication
  - Configurable data word length 8, 10, 16, 20, 24 bits according to standard (PSI5 V1.3)
  - Support of non-standard V1.3 frame length: 11 to 33 bits
  - Generation of 3 or 6 CRC bits for downstream data
  - Start sequence and CRC generator for downstream data
- Message Storage and FIFO access
  - Storage of up to 32 frames per channel with time stamp
  - FIFO access and management
  - Buffer overrun detection
  - Buffer memory status overview registers
- Sending sync pulses
  - Downstream data transmission by variation of pulse length (PSI5 V2.1)
  - Downstream data transmission by leaving out sync pulses (PSI5 V1.3)
  - Staggering sync pulses of all channels to avoid too many channels sending sync pulses in parallel

26.2 Functional overview

The PSI5 IP-module performs communication according to the PSI5 specification V2.1 and also supports the extended powertrain substandard. PSI5 provides high speed and high reliability data transfer.

The PSI5 kernel consists of 2 independent and parallel PSI5 channels each of which support 6 sensor slots. The IP also consists of a Manchester Decoding block that decodes the received PSI5 data frames.

The Buffer Memory holds up to 32 frames per channel along with the timestamp.

The Sync + Data Pulse Generation block generates sync pulses to trigger a data frame for data acquisition from a sensor or for ECU to sensor communication.

The Timestamp Generation block offers three time bases to select from for timestamping.

PSI5 acts as a slave on the FPI bus. The Clock Control block is driven by the system clock frequency f_{SPB}. The Clock Control block produces a total of 5 clock signals.

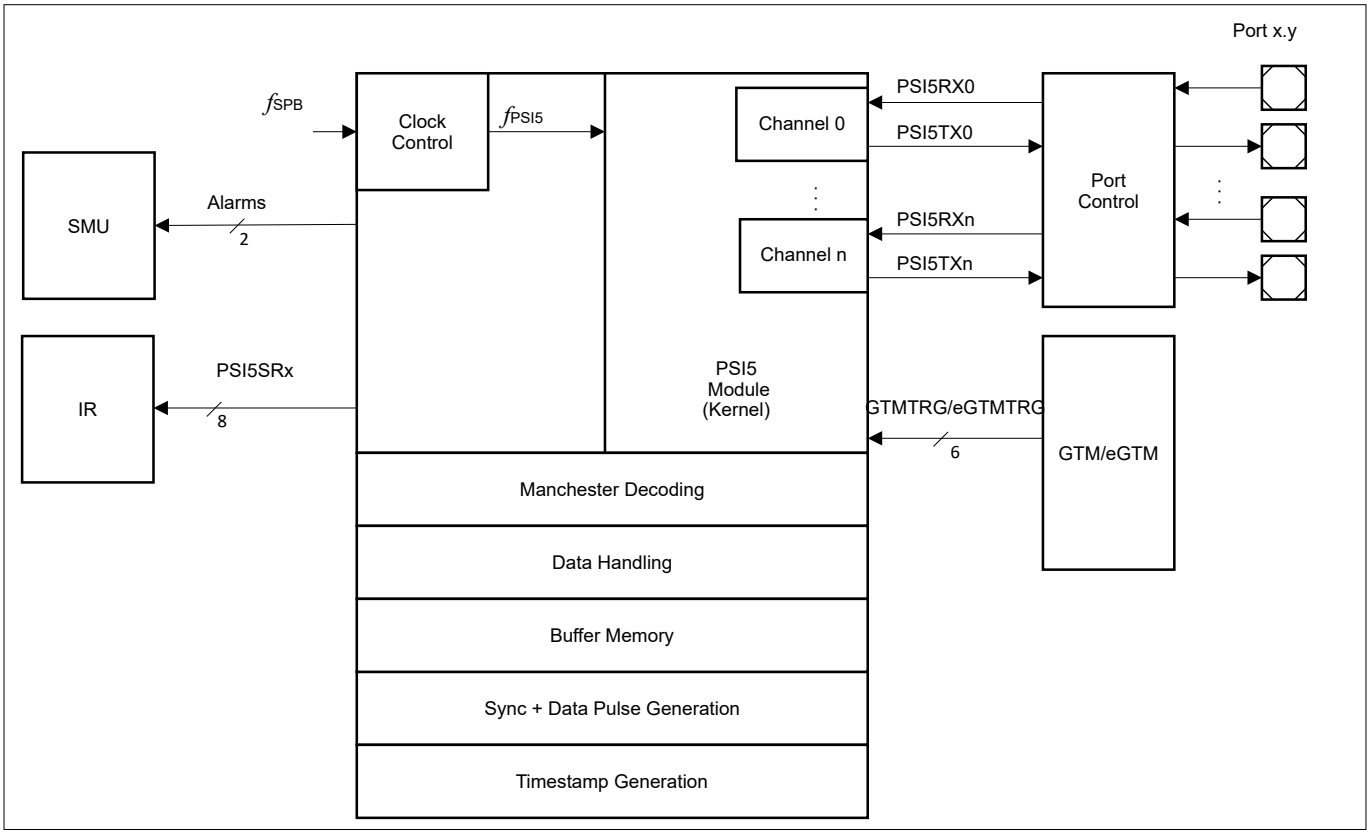


Figure 512 PSI5 block diagram

For the alarm description please refer to the alarm mapping tables in the SMU functional block and the Safety Manual.

The Peripheral Sensor Interface is an interface for automotive sensor applications. PSI5 is an open standard based on existing sensor interfaces for peripheral airbag sensors, already proven in millions of airbag systems.

The PSI5 interface provides a current loop-based serial communication link typically used to connect airbag sensors or other peripheral devices.

While the physical layer is done externally, this module manages protocol handling and data representation to the application. Note that there is no on-chip PHY. The current to voltage and voltage to sync pulse translation is done externally as shown in the figure below.

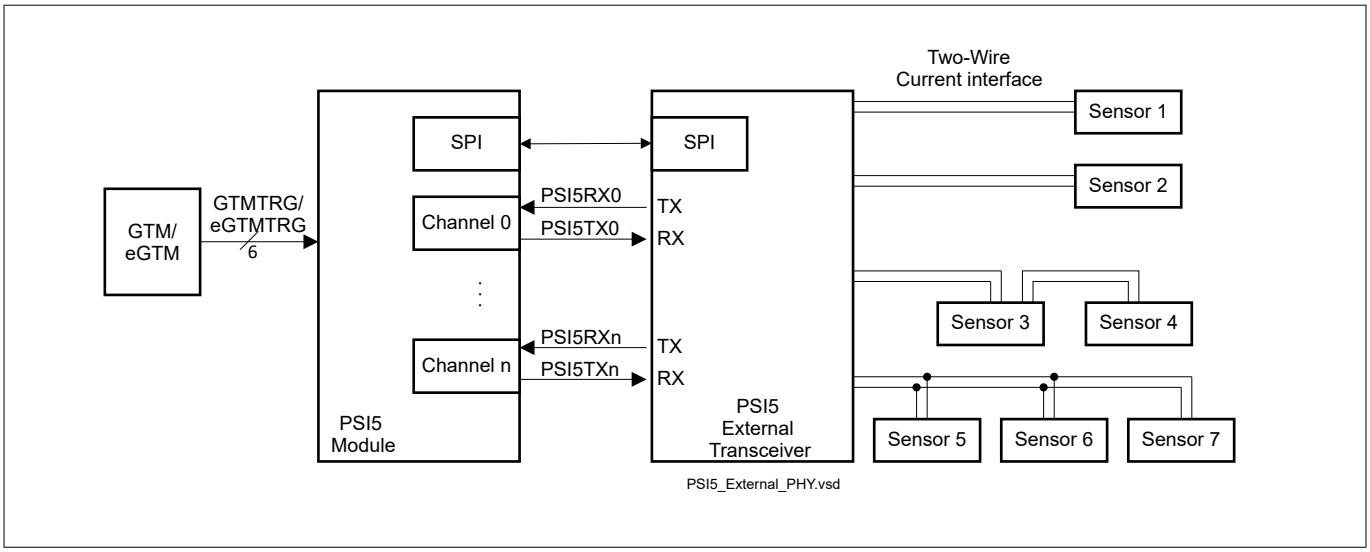


Figure 513 PSI5 module connected to external PHY

Receive data on a PSI5 channel can be set up according to the underlying application. In particular the number of bits is configurable.

The message storage consists of a 32-bit buffer register for each channel and an additional 32-bit register containing the 24-bit time stamp and additional status bits. These 64 bits per frame are additionally stored in a buffer of 32 lines per channel.

In ECU to sensor communication mode, the module can provide regular sync pulses and select between internal and external timer or trigger resources. It also supports setting up the sensor and retrieving sensor status information.

The register set of the PSI5 module can be accessed directly by the CPU for configuration, data read out and status query.

Related information

TC4Dx SMU alarm mapping tables on page 7233

26.3 Functional description

The following section explains the different PSI5 modes of operation and frame formats.

26.3.1 General operation

PSI5 is intended for use in applications where high resolution sensor data needs to be communicated from a sensor to an Engine Control Unit (ECU). It is intended as a replacement for the lower resolution methods of 10-bit A/D converters and PWM and as a simpler low cost alternative to CAN or LIN. The implementation assumes that the sensor is a smart sensor containing a microprocessor or dedicated logic device (ASIC) to create the signal.

The following figure shows a typical device application in which a PSI5 interface reads a sensor device.

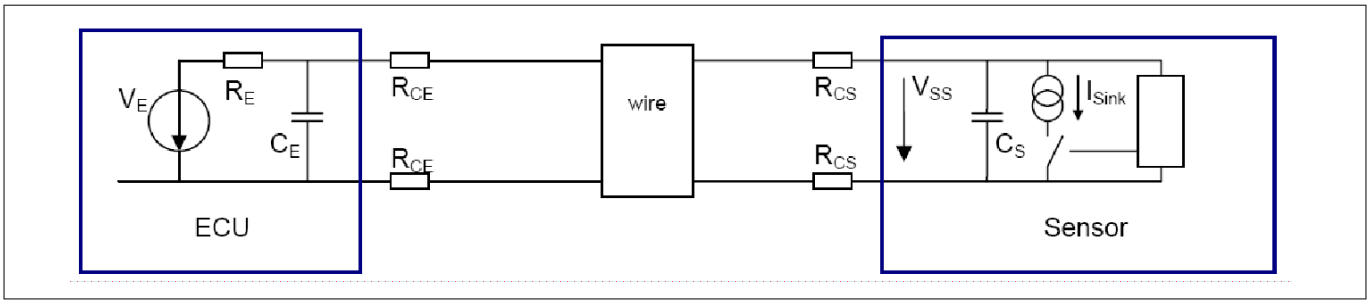


Figure 514 PSI5 to external device connection

PSI5 communication is:

- Asynchronous, unidirectional from sensor to controller without any synchronization
- Synchronous, bidirectional with a sync pulse from the ECU triggering messages
- Synchronous, bidirectional, sync pulses additionally coding data from ECU to sensor

The sensor signal is transmitted as a series of current pulses in Manchester coding.

The sync pulses are transmitted by increasing the voltage.

The following figure shows a typical device application in which a PSI5 ECU reads multiple PSI5 sensor devices on a bus.

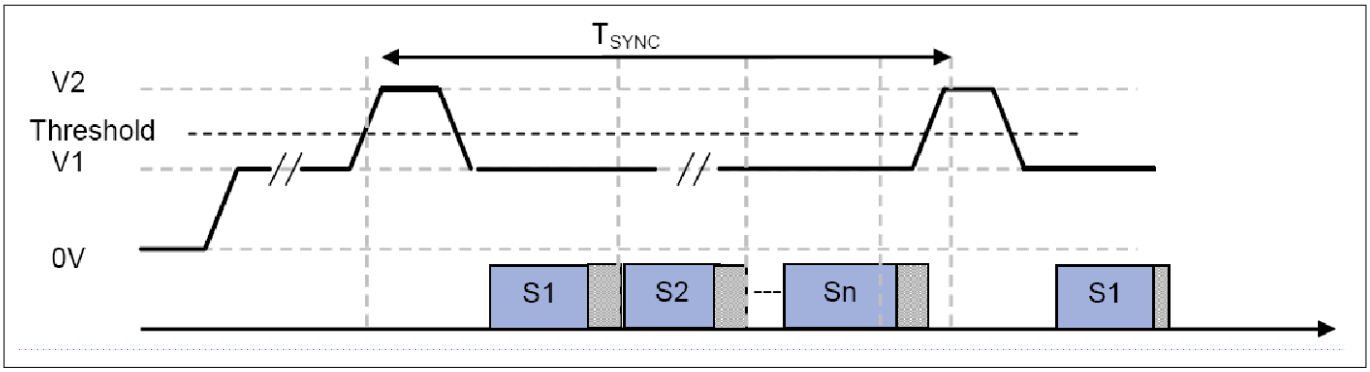


Figure 515 PSI5 frames with sync pulses

26.3.2 Definitions

Acronym	Definition
ASIC	Application specific integrated circuit
ECU	Electronic control unit

Acronym	Definition
SOF	Start of frame
EOF	End of frame
STS	Start sequence
AD	Address data (for example of ECU to Sensor frame)
SD	Send data (Payload of ECU to Sensor frame)

26.3.3 PSI5 operation

The Module supports two modes:

- Standard PSI5 mode supports communication fully compliant to V1.3
- Extended PSI5 mode supporting additional bit-fields and longer frames compliant to V2.1 (see below)

In Standard V1.3 mode, a specific start condition and a start sequence switch the sensor in ECU to Sensor Communication mode. In this mode, the sync pulses are no longer used as triggers for data frames but are interpreted as data. A pulse represents a 1 and a missing pulse represents a 0. This requires isochronous pulses.

In Extended mode (powertrain substandard), the pulse width is modulated. A standard pulse represents a 0, a longer pulse a 1. This way continuous data transmission can be achieved.

26.3.4 Frame formats and definitions

This section describes the frame formats and definitions of the PSI5 protocol.

26.3.4.1 PSI5 V1.3 frame

The following figure shows the layout and definitions of a standard PSI5 frame. Note that the PSI5 standard specifies that the least significant bit is sent out first. See standard for CRC and parity definition.

Example: 16 Bit Data Word „0xAD2C“ with 3-Bit CRC

S1	S2	D0	D1	D2	D3	D4	D5	D6	D7	D8	D9	D10	D11	D12	D13	D14	D15	C2	C1	C0
0	0	0	0	1	1	0	1	0	0	1	0	1	1	0	1	0	1	1	0	0
		C = „1100“				2 = „0010“				D = „1101“				A = „1010“						

Figure 516 Standard PSI5 frame

26.3.4.2 Extended PSI5 frame

The following figure shows the layout and definitions of an extended PSI5 frame as defined in V2.1 for automotive powertrain application. Note that the PSI5 standard specifies that the least significant bit is sent out first.

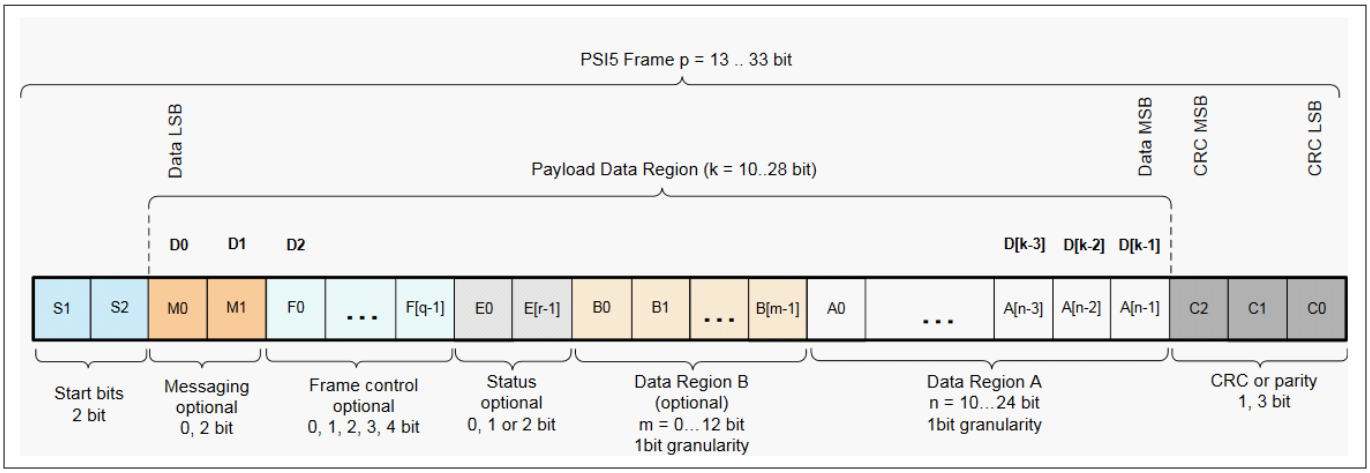


Figure 517 Extended PSI5 frame

Extended PSI5 sensor to ECU message CRC

The transmission of PSI5 data is checked by the following protection modes. The transmission error detection must be selectable:

1. 1-bit even parity
2. 3-bit CRC

The applied generator polynomial of the CRC is $g(x) = 1 + x + x^3$ with a binary start value (seed) “111”. The transmitter shall extend the data bits by three zeros (as MSBs). This augmented data word shall be fed (LSB first) into the shift registers of the CRC check. Start bits shall be ignored in this check. When the last zero of the augmentation is pending on the input adder, the shift registers contain the CRC checksum. These three check bits shall be transmitted in reverse order (MSB first: C2, C1, C0).

In this implementation the number of elements is configurable no longer or shorter than 8..28 bits.

26.3.4.3 Enhanced serial message (“slow channel”)

The serial data is transmitted bit-wise per frame in bits [1:0] of the Messaging bit-field of 18 consecutive messages from the transmitter.

26.3.4.3.1 Enhanced serial data frame

Serial data (“slow channel”) is communicated in an 18 frame sequence as shown in the following figure. The start of an Enhanced Serial Message is indicated by the unique pattern “1111110” in bit M1 constructed across 7 consecutive PSI5 messages. In message 1 - 6 they are set to “1”. In message 7, 13 and 18 they are set to “0”. 18 consecutive PSI5 messages must be successfully received (no errors) for the Enhanced Serial Message data to be received. The CRC generation is different from the CRC generation on the data nibbles. (See SENT SAE J2716 JAN 2010 [3])

The serial message frame contains 21 bits of message data and a 6-bit frame-check sequence. Two different configurations can be chosen:

- 12-bit data and 8-bit message ID
- 16-bit data and 4-bit message ID

A configuration bit (serial data bit M1, serial communication Messaging bits of frame No. 8) determines the configuration of the Enhanced Serial Message frame. It determines how the PSI5 module automatically interprets the serial data.

- Configuration bit = 0: 12-bit data and 8-bit message ID
- Configuration bit = 1: 16-bit data and 4-bit message ID

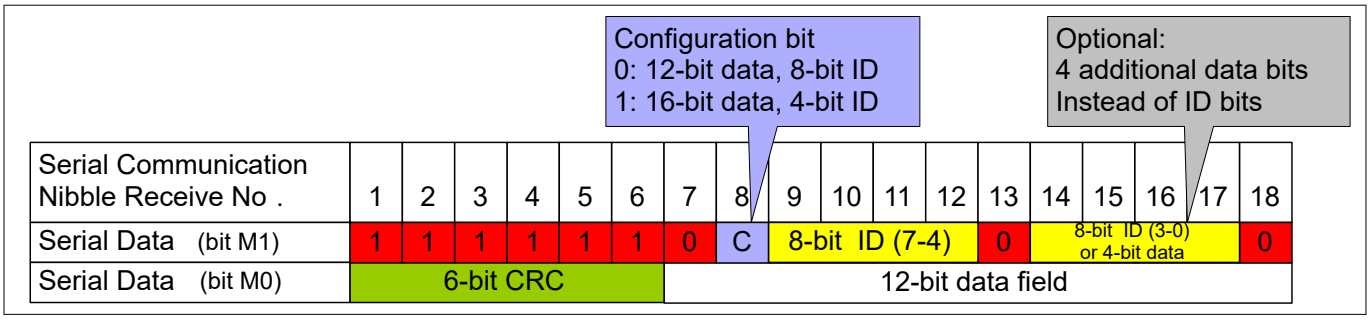


Figure 518 Enhanced serial data frame

The following figure shows in detail the frame structure for Enhanced Serial Data frames with configuration bit = 0.

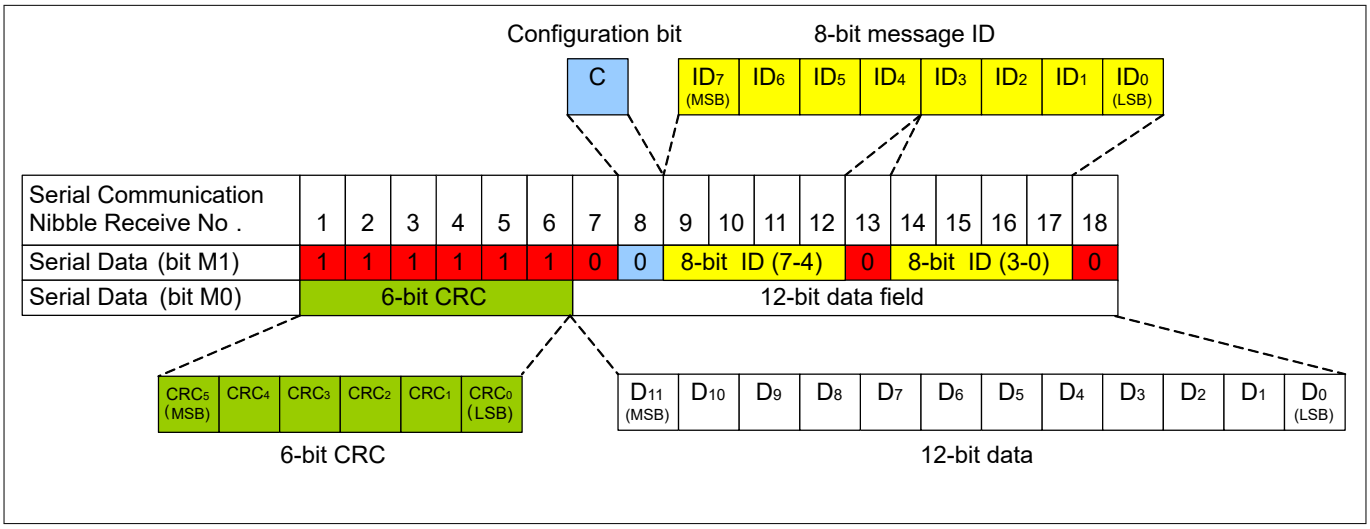


Figure 519 Configuration bit = 0

The following figure shows in detail the frame structure for Enhanced Serial Data frames with configuration bit = 1.

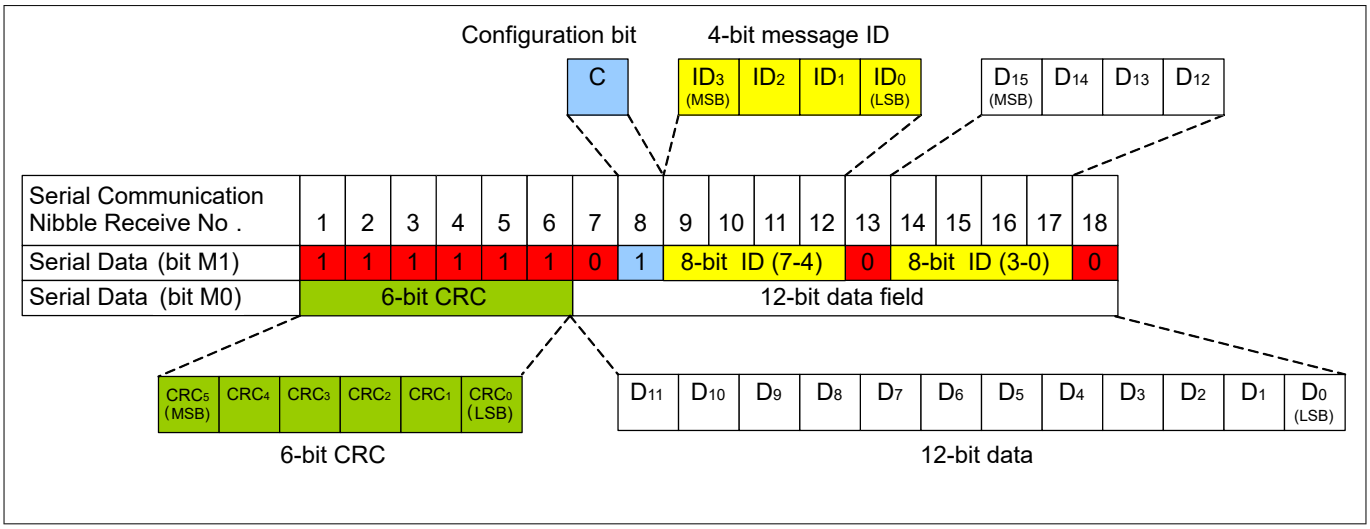


Figure 520 Configuration bit = 1

26.3.5 Sync pulses

Sync pulses are used for two different purposes:

- Triggering a data frame for data acquisition from a sensor or
- ECU to sensor communication

26.3.5.1 Synchronous transmission

In the Synchronous mode, the sensor (slave) starts to transfer a complete data frame only after a sync pulse is forced by the master on the output PSI5TXx pin. An external PHY translates this pulse into a voltage increase. The sensor then initiates a measurement and starts to calculate the new output data value. The data follows in a standard PSI5 frame, starting with the 2 start bits, data and parity or CRC. The timing diagram in the following figure visualizes a synchronous transmission.

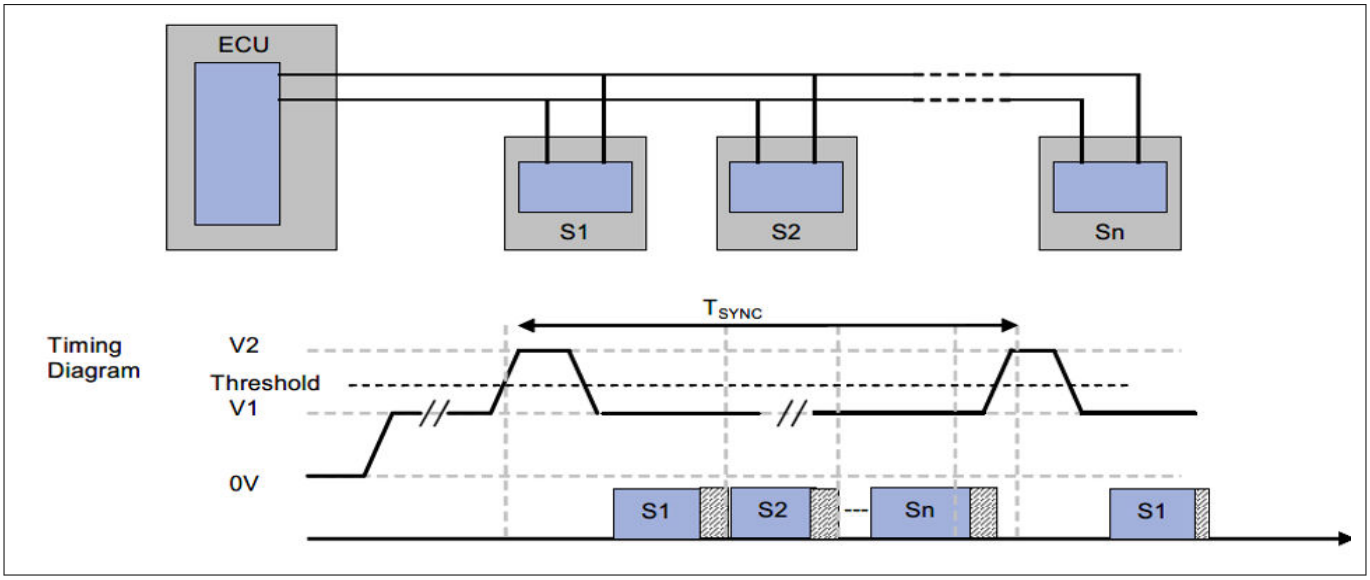


Figure 521 Synchronous transmission

26.3.5.2 ECU to sensor communication

Two versions are supported:

- Legacy PSI5 V1.3 and
- Extended powertrain substandard

Legacy version

The legacy version uses sync pulses as start condition and start bits to signal the start of the ECU to sensor communication. From this start on, the sync pulses are interpreted by the sensor as data. See standard for CRC and parity definition.

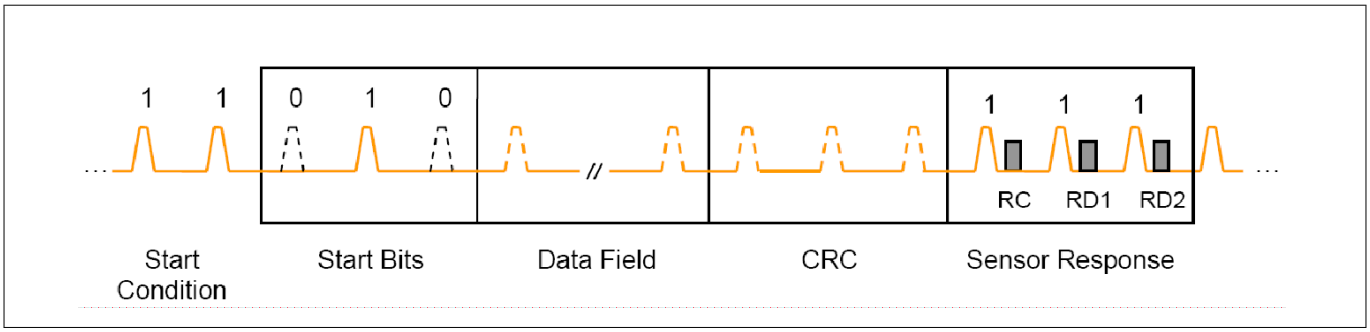


Figure 522 ECU to sensor (legacy)

Extended powertrain substandard

The revised version is the extended powertrain substandard that allows for modulation of the sync pulse duration. This way bidirectional communication is possible continuously. Every ECU to sensor frame starts with a nine-bit frame-start-sequence [0 1 1 1 1 1 1 1 0]. Synchronization is possible at the beginning of each frame.

Sync bits

A consecutive train of seven ones within the transmitted data could be misinterpreted as a frame start condition. Therefore a sync-bit (logical 0) is inserted at every seventh bit position (after every six bits of the sensor address and the payload data). With respect to the data transmission, these sync bits are treated like stuffing bits, which are removed at the receiver side.

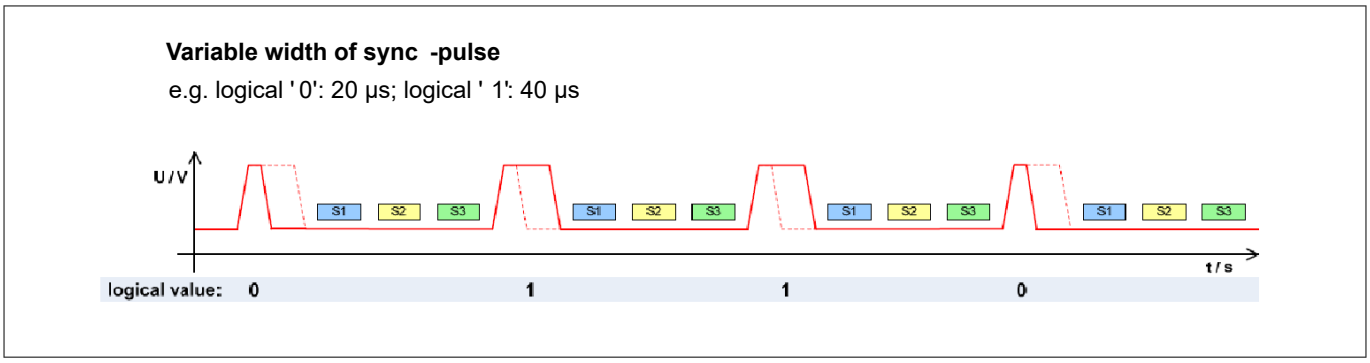


Figure 523 ECU to sensor, bit transmission (extended)

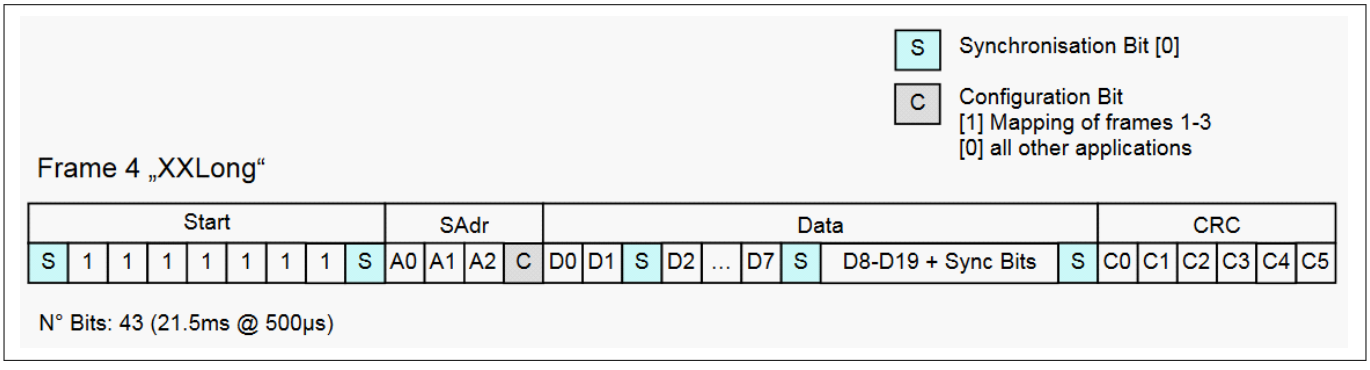


Figure 524 ECU to sensor, frame format (extended)

Extended ECU to sensor message CRC

The ECU to sensor frame has a 6-bit CRC.

The generator polynomial of the CRC is $g(x) = x^6 + x^4 + x^3 + 1$ with a binary CRC initialization value “010101”. The transmitter extends the data bits by six zeros (as MSBs). This augmented data word shall be fed (LSB first) into the shift registers of the CRC check. Start bits and sync bits are ignored in this check. When the last zero of the augmentation is pending on the input adder, the shift registers contain the CRC checksum. These six check

bits shall be transmitted LSB first [C0, C1…C5]. In this implementation the number of elements is configurable and no longer than 24 bits (standard is 4 address and 20 data bits)

26.3.6 Manchester decoding

Decoding of the Manchester protocol:

- The Manchester decoder uses a nominal 4 MHz/6.048 MHz sampling clock (32 samples per bit)
- A sampling counter, running at the sampling clock rate, controls the sampling of the incoming Manchester encoded data
- The Manchester signal transition occurring in the middle of the bit time resets the counter, making the counter count from 0-31 (nominally) between successive mid-bit times
- Detection of the mid-bit transitions uses three consecutive samples (011 = rising edge, 100 = falling edge)
- The maximum value of the 6-bit counter shall be 41, at which point a Manchester error is detected due to lack of a transition, and the counter rolls over to 0 (“idle time”)
- The value of a Manchester encoded input data bit is determined by proper logic levels before and after a transition and proper timing of the transition
- Bit-value determination nominally takes place at the $\frac{1}{4}$ and $\frac{3}{4}$ bit times, with the mid-bit transition occurring at the $\frac{1}{2}$ bit time. This is nominally at sampling counter values of 8 ($\frac{1}{4}$ bit time) and 24 ($\frac{3}{4}$ bit time)
- Logic levels are determined using a 2 out of 3 majority vote of consecutive samples. (samples 23, 24, 25 and 7, 8, 9)
  - The combinations 111, 011, 101, 110 indicate a 1
  - The combinations 000, 001, 010, 100 indicate a 0

Manchester-2 is used by the PSI5 standard:

- Start bits = 0 0
- Falling edge corresponds to logical 1
- Rising edge corresponds to logical 0
- Least significant bit (LSB) is sent out first

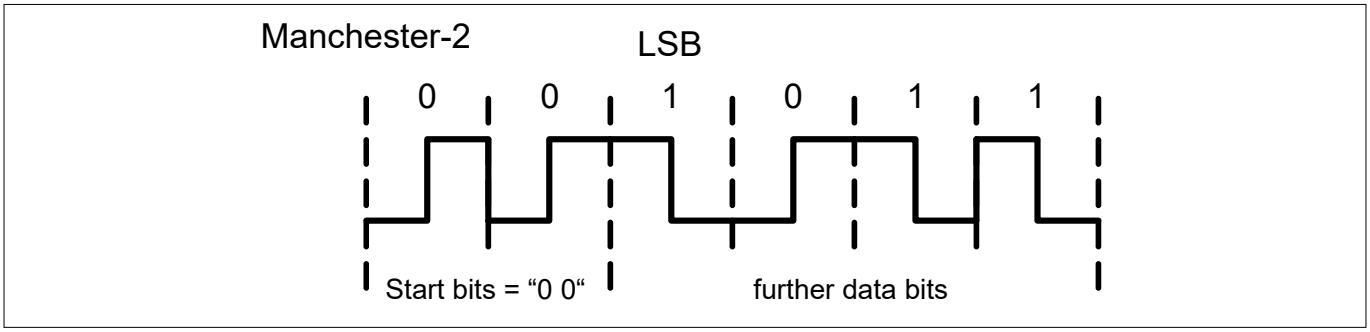


Figure 525 Manchester-2

Sampling details

- Recognition of frame start
  - Manchester decoding requires a deterministic start of frame
  - Frame separation: An idle time of at least one bit time is required
  - Idle time is the time without edges on the line that is required to safely determine that no frame is being transferred (1 bit time + tolerance = > 41 samples)

The idle counter starts counting on a falling edge, after sync pulse or after blank out time, if the channel is switched on. If the line is low, it counts up to 41 and stops and activates the idle flag until the line shows a rising edge. This clears the idle counter
  - After this idle time, a determined start bit sequence is required for synchronization of the phase of receiver and transmitter (0 0 in case of Manchester encoding)
  - If no idle time occurs at the end of a frame, bits after the configured length are dropped
  - If a second frame comes too early (collision = no idle time), the second frame is dropped. The decoder waits for an idle time for resynchronization
  - Idle time is particularly required in case of Manchester errors and frame collisions for a safe resynchronization. Otherwise, two successive zeros in a damaged frame/a second colliding frame could lead to reception of wrong data. Note that back to back transfers of correct frames are forbidden by standard (T_gap required)
  - This idle time is required as well after a regular end Of frame, sync pulse or a blank out time to avoid inadvertent synchronization inside frames under transmission
- Bit sampling
  - After the start of the protocol is detected and an edge on the line signal is detected and the sample counter is greater than 25 the sample counter value is set to 0
  - If the sample counter is 41, the counter value is set to 0 and stops
  - The sample counter will be incremented each clock cycle
  - When the sample counter has the value 9, the level is stored in cnt9_val
  - When the sample counter has the value 25, the level is stored in cnt25_val
  - cnt9_val and cnt25_val are reset if no frame is detected. Otherwise, a wrong start bit detection could occur
- Bit evaluation
  - When the sample counter value is 16, the decision is done for the new sample value:
  - Check if the last edge was detected correctly (edge detected between sample counter value 25 and sample counter value 41)
  - If true:
  - Cnt_25_lev = 0 and cnt_9_lev = 1 => new sample = 0
  - Cnt_25_lev = 1 and cnt_9_lev = 0 => new sample = 1
  - If the edge check fails or cnt_25_lev = cnt_9_lev, Manchester Error is assumed

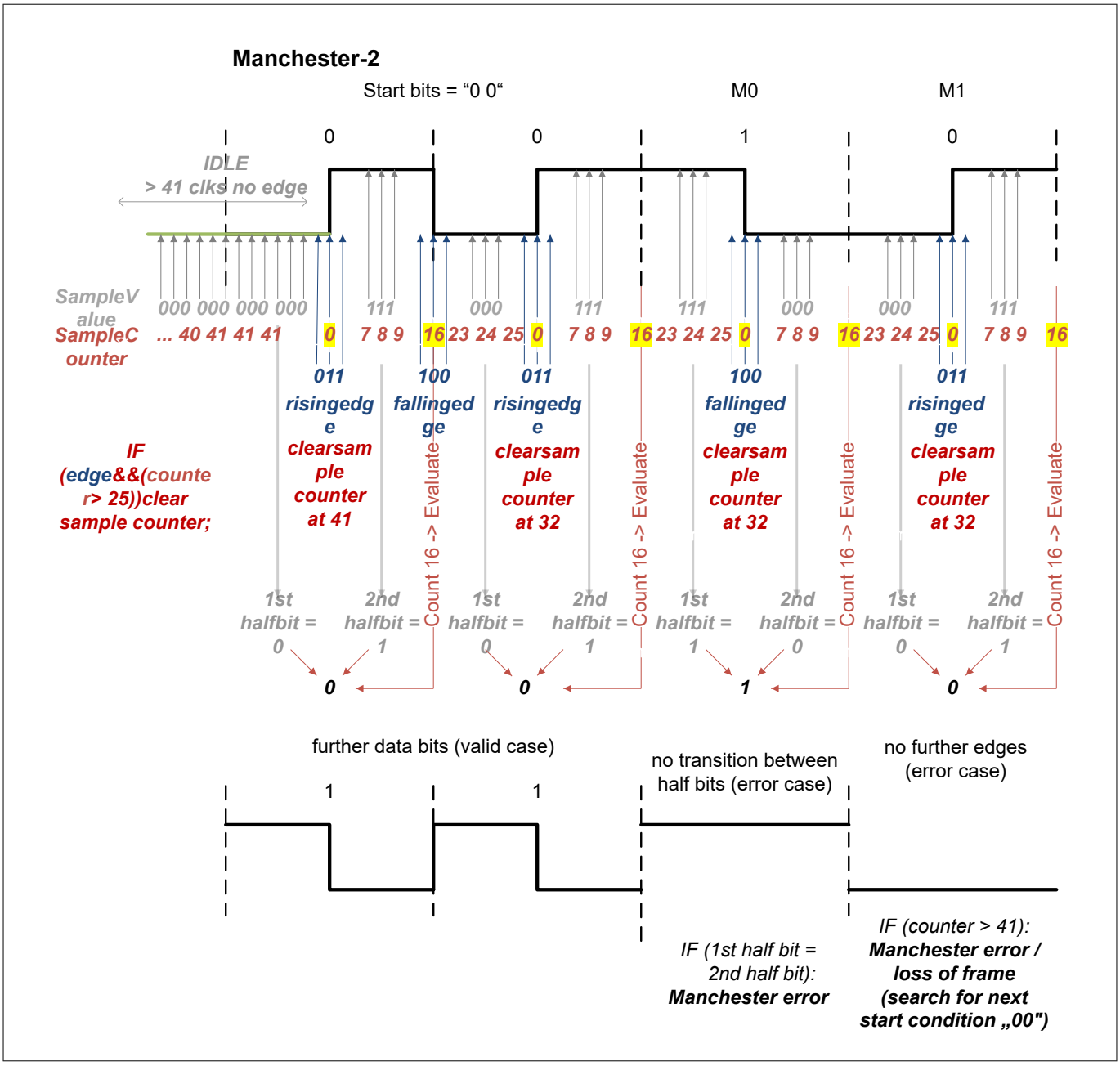


Figure 526 Manchester decoder

Definition of Manchester error (ME)

- Two half bits have the same value => ME, the decoder waits for an idle time
- > 41 sample times no edge => ME
- The ME has no dedicated interrupt but will lead to various error responses depending on where it occurs. See the following table

Table 1181 Error responses due to ME

Manchester error (ME) case	Response	Comment
First ME in start bits	Discard frame	NFI (if RCRBx.FEC is set) and empty frame stored (if RCRBx.FEC and RCRBx.VBS is set), Timestamp = end of slot where SOF happened. This is true even if these start bits should start before WDL and end after WDL
Start bits cut off by blank out time (BOT) or sync pulse (frame starts without leading idle time)	Discard frame	NFI (if RCRBx.FEC is set) and empty frame stored (if RCRBx.FEC and RCRBx.VBS is set)
First ME in Messaging bits	If MSG is set: MEI	Frame is stored, bits after error are lost (“0”)
	If MSG cleared: NBI	
First ME in data, parity or CRC / Frame ends too early	NBI	Frame is stored, bits after error are lost (“0”)
Frame ends too late (longer than PDL)	No NBI	Bits after PDL are cut off and EOF is detected. Frame is stored
Collision (idle time missing)	Discard 2nd frame	If the 1st frame was too late and running over WDL or If the 2nd frame was too early and running over WDL then for 2nd frame, NFI (if RCRBx.FEC is set) and empty frame stored (if RCRBx.FEC and RCRBx.VBS is set), Timestamp = end of 2nd slot
In all cases, the receiver waits for an idle time followed by the start bits “0 0” before a new frame is detected		

26.3.7 Bit rate generation

The bit rate of each channel is individually selectable from the 2 standard bit rates 125 kHz and 189 kHz. The two referring over sampling frequencies are generated centrally.

This chapter shows in detail how the bit rate for each channel is adjusted.

In the first stage, a global fractional divider serves as pre-divider. Its intermediate frequency $f_{fracdiv}$ can be set up so that it is handy as input frequency for the different PSI5 channels. Usually this can be left unused.

For each of the two bit rates there is an own fractional divider. Each channel can select one out of the two clock signals f_{125} and f_{189}.

The PSI5 module provides 5 clock signals centrally (see following figure):

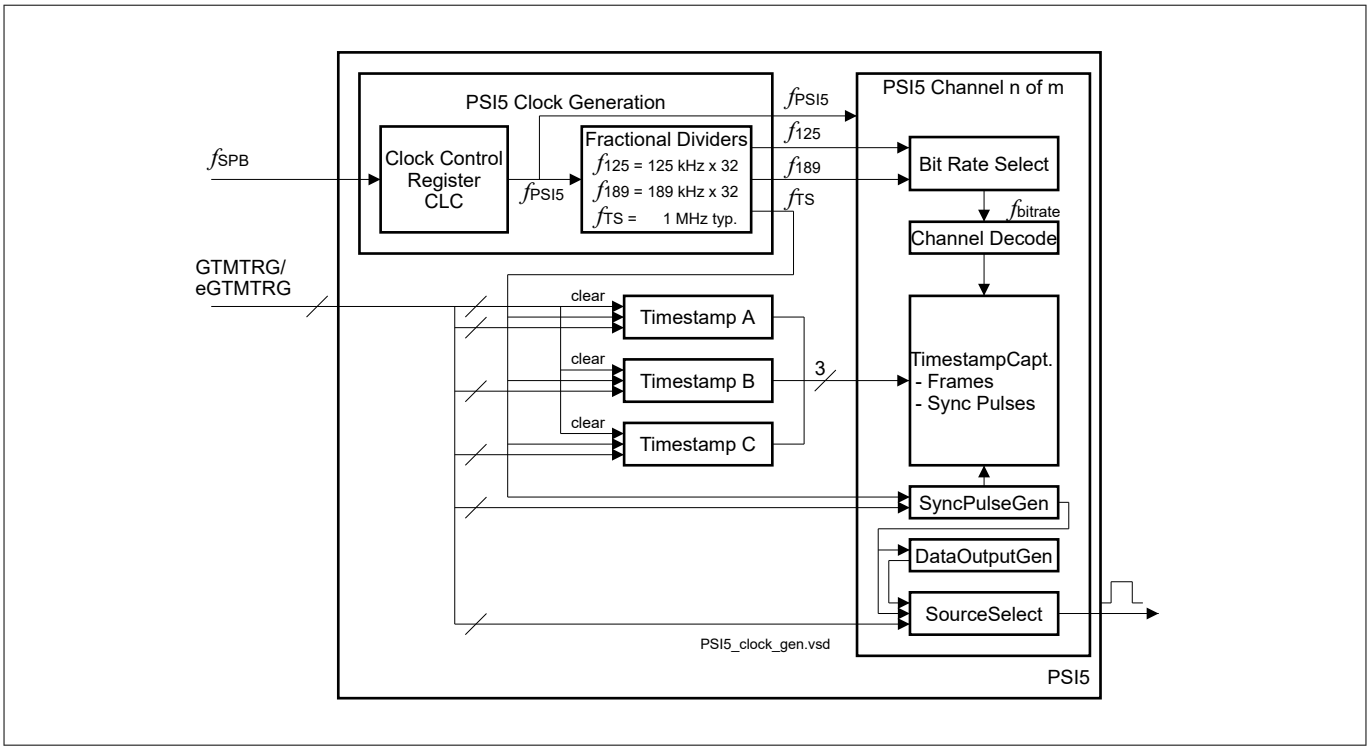


Figure 527 PSI5 module clock generation

- f_{PSI5}
  This is the module clock that is used inside the PSI5 kernel for control purposes such as clocking of control logic and register operations. The frequency of f_{PSI5} is always identical to the system clock frequency f_{SPB}. The Clock Control register CLC makes it possible to enable/disable f_{PSI5} under certain conditions
- $f_{fracdiv}$
  This clock is the module clock that is used inside the PSI5 kernel for bit rate generation of the serial channels. The Fractional Divider register FDR controls the frequency of $f_{fracdiv}$. Usually not required and set to 1
- f_{125}
  This clock is the oversampling clock that can be selected as the bit rate frequency inside a PSI5 channel. The Fractional Divider register for Lower Bit Rate FDRL controls the frequency of f_{125}. The clock drives the Manchester decoder and needs to be 32 times faster than the nominal bus frequency
- f_{189}
  This clock is the oversampling clock that can be selected as the bit rate frequency inside a PSI5 channel. The Fractional Divider register for Higher Bit Rate FDRH controls the frequency of f_{189}. The clock drives the Manchester decoder and needs to be 32 times faster than the nominal bus frequency
- f_{TS}
  This clock is used to drive the central timestamp counter. It can be selected to drive the global sync pulse time base and inside the PSI5 channels to drive the local Watchdog timers. The Fractional Divider register for Timestamp FDRT controls the frequency of f_{TS}. This is usually adjusted to 1 MHz/1 µs period time

The following two formulas define the frequency of $f_{fracdiv}$:

$$f_{\text{fracdiv}} = f_{\text{PSI5}} \div (1024 - \text{FDR.STEP});\ \text{FDR.DM} = 01_{\text{B}} \tag{15}$$

$$f_{\text{fracdiv}} = f_{\text{PSI5}} \times \text{FDR.STEP} \div 1024 \text{ with } \text{STEP} = 0...1023; \text{FDR.DM} = 10_{\text{B}} \quad (16)$$

The following two formulas define the frequency of f_{125}:

$$f_{125} = f_{\text{fracdiv}} \div (1024 - \text{FDRL.STEP}); \text{FDRL.DM} = 01_{\text{B}} \quad (17)$$

$$f_{125} = f_{\text{fracdiv}} \times \text{FDRL.STEP} \div 1024 \text{ with } \text{STEP} = 0...1023; \text{FDRL.DM} = 10_{\text{B}} \quad (18)$$

The following two formulas define the frequency of f_{189}:

$$f_{189} = f_{\text{fracdiv}} \div (1024 - \text{FDRH.STEP}); \text{FDRH.DM} = 01_{\text{B}} \quad (19)$$

$$f_{189} = f_{\text{fracdiv}} \times \text{FDRH.STEP} \div 1024 \text{ with } \text{STEP} = 0...1023; \text{FDRH.DM} = 10_{\text{B}} \quad (20)$$

The following two formulas define the frequency of f_{TS}:

$$f_{\text{TS}} = f_{\text{PSI5}} \div (1024 - \text{FDRT.STEP}); \text{FDRT.DM} = 01_{\text{B}} \quad (21)$$

$$f_{\text{TS}} = f_{\text{PSI5}} \times \text{FDRT.STEP} \div 1024 \text{ with } \text{STEP} = 0...1023; \text{FDRT.DM} = 10_{\text{B}} \quad (22)$$

26.3.8 Digital glitch filter

Very slow slopes and signal noise can lead to fast transitions of the input signal. These unwanted transitions are suppressed by a digital glitch filter.

It is built for filtering very fast transitions only. The filter calculates the integral of the signal. If the integral reaches a programmable saturation point, the signal change is notified to the pulse measurement unit. So it helps to find the exact pulse length for said slow slopes in noisy environment.

The glitch filter is clocked with f_{PSI5}. If the state of the input sample differs from the current output signal value, the internal counter is incremented by one. When the state of the input sample matches the current output signal value and the timer is not in idle, the timer is decremented by one. When the timer matches the compare value stored in IOCRx.DEPTH, the level of the output signal line is inverted. The timer will be reset and set to idle state again.

The depth of the filter can be programmed to a value between 1 and 15. Typically a depth of 3 to 5 T_{PSI5} is sufficient. By default it is cleared. If IOCRx.DEPTH is cleared, the filter is inactive/bypassed. Nevertheless, the input signal is sampled with f_{PSI5}.

The internal signal after the filter will change value not before the new value was sampled IOCRx.DEPTH times. If during this period a spike occurs, it takes $2 \times T_{\text{PSI5}}$ times longer for the internal signal to change value. The filters principal implies a delay of the signal by (IOCRx.DEPTH x T_{PSI5}).

Upon detection of glitch during rising or falling edge, IOCRx.REG or IOCRx.FEG is set. The rising / falling edge glitch flags must be reset by software.

The following figure shows the digital glitch filter:

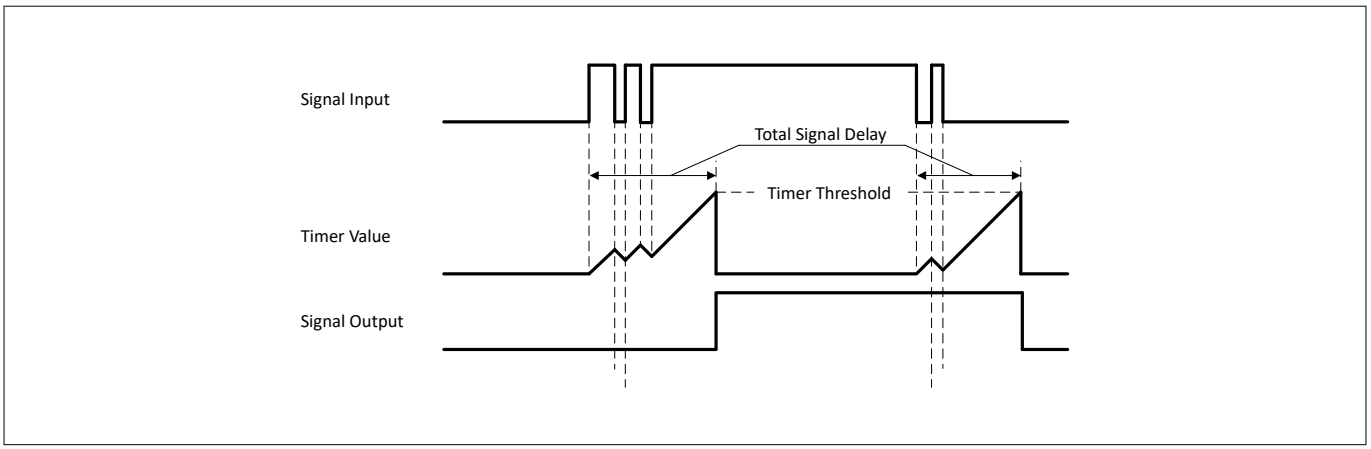


Figure 528 Digital glitch filter

26.3.9 Timestamp generation

Two time bases can be selected for timestamping:

- Internal time base (bit TBS = 0)
- External time base (bit TBS = 1) GTM/eGTM. Bit TSRA/B/C.TRIGSEL is used to select between GTM or eGTM for each of the timestamp counters. Bit TSRA/B/C.ETB is used to select the external clock line for counters CTS. ECS, if enabled by ECEA/ECEB/ECEC, selects the external trigger line that clears the timestamp counters CTS

Three timestamp counters can be selected for timestamping:

Any of the registers TSRA, TSRB and TSRC provide one timestamp counter. Each of them can be configured to be driven by either f_{TS} or by an external timer, that is, one of the GTM or eGTM inputs based on bit TRIGSEL. The frequency of f_{TS} is controllable by the fractional divider in register FDRT.

All of them can be cleared by a selectable GTM signal or by software, individually or synchronously.

Two time instances are captured for each channel:

1. The last sync pulse sent per channel is stored in the Start of Pulse Timestamp Capture register SPTSCx. Bit RCRCx.TSP is used to select from the three timestamp counters for sync pulses
2. The last start of frame (SOF) on this channel is stored in the Start of Frame (SOF) Timestamp Capture register SFTSCx. Bit RCRCx.TSF is used to select from the three timestamp counters for SOF

For each channel the capture time (SPTSC or SFTSC) can be selected using RCRCx.TSR. The selected time would be stored in the Receive Data Register (RDRLx, RDRHx) and the Receive Data Memory (RDMLxy, RDMHxy).

The following figure shows the timestamp generation:

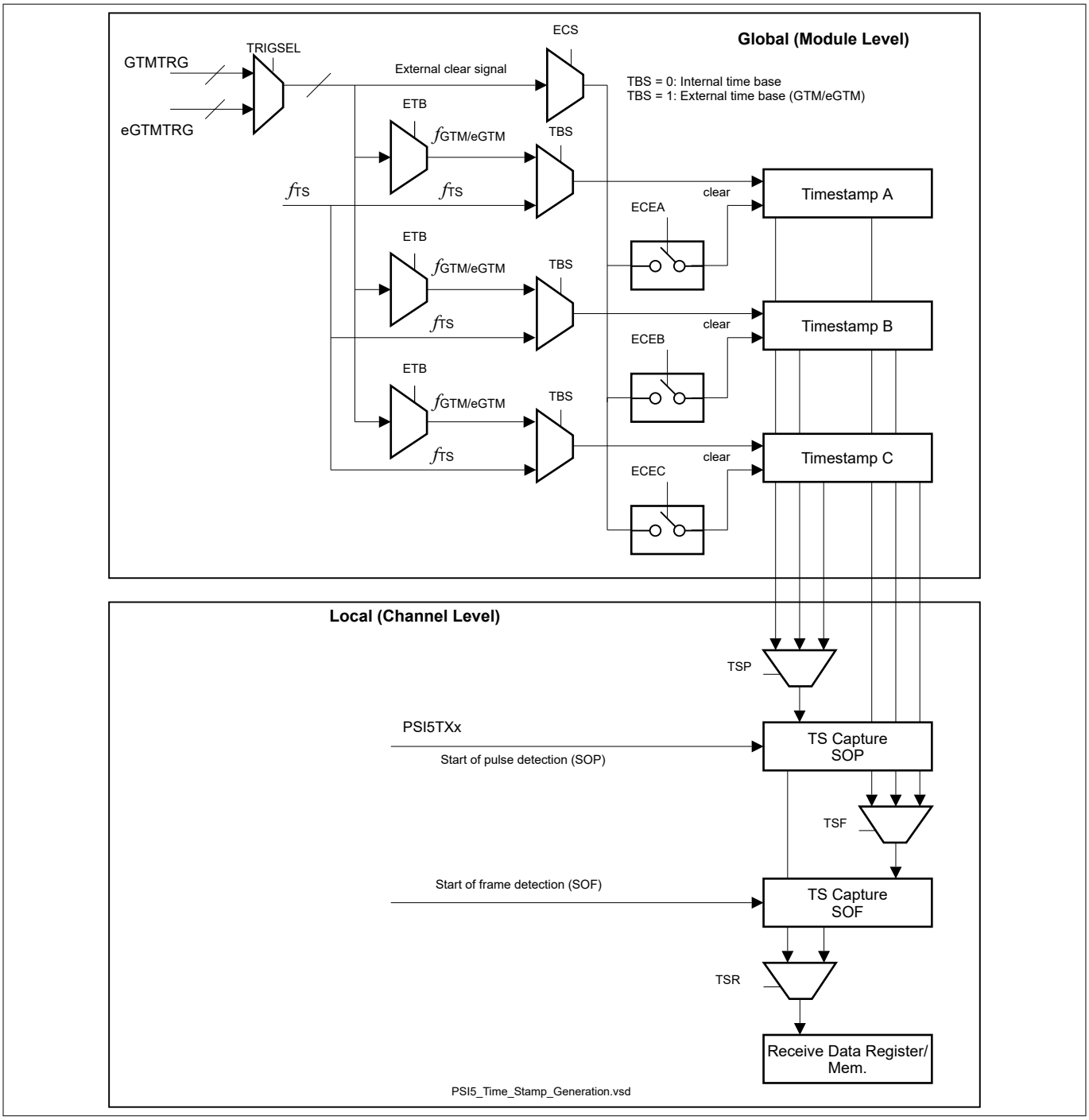


Figure 529 Timestamp generation

26.3.10 Watchdog timer

This section describes the behavior of the Watchdog timer in both Synchronous and Asynchronous modes of operation.

26.3.10.1 Synchronous mode

Selected with RCRAx.ASYN = 0.

The Watchdog Timer Registers WDTxw contain 7 time limits defining 6 frame slots per channel. The value entered here defines the time in multiples of TTS (defined in FDRT) from the last start of a sync pulse on channel x. The internal watchdog timer is started automatically at the start of a sync pulse.

The internal watchdog timer is compared to WDTxw.WDL. A match triggers:

- The automatic incrementing of slot counter RDRHx.SC
- Starting from WDTx1.WDL (channel x limit 1) on: Check if a frame with the corresponding SC y (y = slot 1 to 6) has been received or if such a frame is still being received since last WDTxw.WDL expired. If this is not the case but a frame was expected in this slot y (configured so in RCRBx.FECy (y=0-5)), “No Frame Received” interrupt NFI is issued. If configured so (in RCRBx.VBSy (y = 0-5)), an empty frame (all data fields are 0) is stored in RDRL/Hx and the referring RDML/H
- Frames that start before a slot boundary and end after this boundary (but are too short and so discarded) will result in an NFI / empty frame as well (if configured). In this scenario, Timestamps assume that RCRCx.TSR selects SFTSCx
  - If no frame was received before the SOF in this slot:

    SC = slot where SOF happened.

    Timestamp = end of slot where SOF happened
  - If a valid frame was received in this slot before a second SOF crosses WDL value, the parameters for the next slot are taken over already and if no valid frame is received in this next slot:

    SC = next slot (SC where SOF happened + 1)

    Timestamp = end of next slot (SC where SOF happened + 1)
  - If a valid frame was received in this slot before a second SOF crosses WDL value, the parameters for the next slot are taken over already and if a valid frame is received in this next slot, no NFI is issued
- A frame starting before a slot y (y = 1 to 6) and ending in slot y will not prevent NFI for slot y (y = 1 to 6)

Detection of frame boundaries:

- SOF is the first rising edge of the first start bit after an idle time, a timestamp is triggered and SC is captured for the frame at this point in time
- EOF is determined based on the configured frame length (RCRAx.PDLy and RCRCx.CRCy). An unexpected idle time or Manchester error will result in earlier EOF detection
- For each frame, the receiver checks if slot counter SC at SOF is equal to SC at EOF. If not, the interrupt TEI is issued. FEC is not relevant for TEI

If WDTxw.WDL is cleared, no check is performed but RDRH.SC is still incremented when it is its turn in the sequence of the WDLs. So such unconfigured WDTxw.WDL must appear only at the end. If an unconfigured WDTxw.WDL appears between configured WDTxw.WDL, the slot count will be incorrect

If a frame is received in a slot where no frame is expected, no dedicated flag is set but the frame is received with the parameters (PDL, SC and on) given for this slot. SW should detect this as SC for this frame is wrong.

26.3.10.2 Asynchronous mode

Selected with RCRAx.ASYN = 1.

The watchdog timers for the PSI5 channel x, limits WDL1 to WDL6 are disabled. If WDL0 (limit 0 of WDL) is not cleared, TEI is issued if the distance between two frames is longer than specified value in WDL0. Slot Counter SC is incremented with each received frame (works as frame counter) with roll over after 6. If bit RCRAx.AVBS (verbose mode) is set, an empty frame is stored in RDRL/Hx and the referring RDML/H.

Channel configuration at slot boundaries/take over of slot parameters

The receiver takes over the parameters for a frame at the beginning of a slot with 3 exceptions:

1. During SLOT0 (this is the slot after the last configured slot), after the sync pulse, the configuration of SLOT1 is already used. An internal shadow slot count value makes sure the messaging bits are routed to

slow channel for SLOT1. This way, a frame for SLOT1 that is too early will be received correctly, the messaging bits are sorted correctly. The time violation is still visible by TEI = 1 as such a frame would still start in a different slot (SLOT0) than it ends (SLOT1). SC = 1 will be stored in the receive buffer/memory

2. Frames that come too early could destroy the slow channel of the preceding slot. To prevent this, during SLOT1-SLOT5, the configuration of the next slot is loaded immediately after EOF was received. Note that an SOF must be seen first in the same slot before an EOF is looked for. This way, the hardware prevents the destruction of a good frame by a preceding frame that is too late. An internal shadow slot count value makes sure the messaging bits are routed to the slow channel for the following SLOT. This way, a frame for example on SLOT3 that is too early will be received correctly, the messaging bits are sorted correctly in SLOT3 and not in SLOT2 where the SOF of this frame is. The time violation is still visible by TEI = 1 as such a frame would still start in a different slot (SLOT2) than it ends (SLOT3). SC = 3 will be stored in the receive buffer/memory
3. If a frame does not end at the configured end of a slot, the new parameters of the following slot are taken over by the receiver only after the EOF was detected. This allows to receive a frame properly that is too late

***Note**: Orphan frames that come with a shift of one or more slots cannot be detected. They will be stored with the SC as they appear.*

Frames coming too early

A frame may come too early and will consequently start in a slot that is configured for no frame (FEC = 0). If the user wants to make sure that this frame is stored in RDR/RDM, the slot with FEC = 0 must be configured in the same way as its follower. At least, the length field PDL must be set to its maximum value (28) to allow for reception of all frame lengths. This allows to capture such a frame with referring error bits. (TEI, NBI, CRC in most cases). If this is not done and PDL is (by default) = 0, the following happens:

CASE1: SOF is too early. In the slot with FEC = 0, EOF is detected in the slot with FEC = 0.

The frame is stored in RDR/RDM with timestamp, 0 bits of data (as configured in PDL) and the slot count of the too early slot (with FEC = 0). CRCI is generated as the state machine assumes that there is no valid CRC that can be checked as there is no valid data, NBI is issued.

CASE2: SOF is too early. In the slot with FEC = 0, EOF is detected in the slot following the slot with FEC = 0 (because CRC transmission ends here. Meaning, the SOF was very close to the end of the slot)

The frame is stored in RDR/RDM with timestamp, 0 bits of data (as configured in PDL) and the slot count of the too early slot (with FEC = 0). CRCI is generated as the state machine assumes that there is no valid CRC that can be checked as there is no valid data, NBI and TEI bits are set.

Usage of the watchdog timer limits

The example below shows some use cases.

F1: “Frame too early”: Slot Count SC at SOF is 0 and 1 at EOF. Timing Error is issued (TEI). F1 is stored with configuration of slot 1 and SC = 1.

WDL2: Slot Count SC incremented by hardware, no “No Receive Frame” interrupt as no frame expected (FEC = 0)

F3: “Frame too late”: Slot Count SC at SOF is 3 and 4 at EOF. Timing Error is issued (TEI). The configuration for slot 3 is kept until the end of F3 so that it is not destroyed. Note that the EOF of frame 3 does not lead to take over the configuration of slot 5. But the EOF of F4 does. This way a F5 coming too early would be received correctly.

WDL5: “Frame missing”: No SOF or EOF received since WDL4. A frame was expected (FEC = 1), No Receive Frame interrupt is issued.

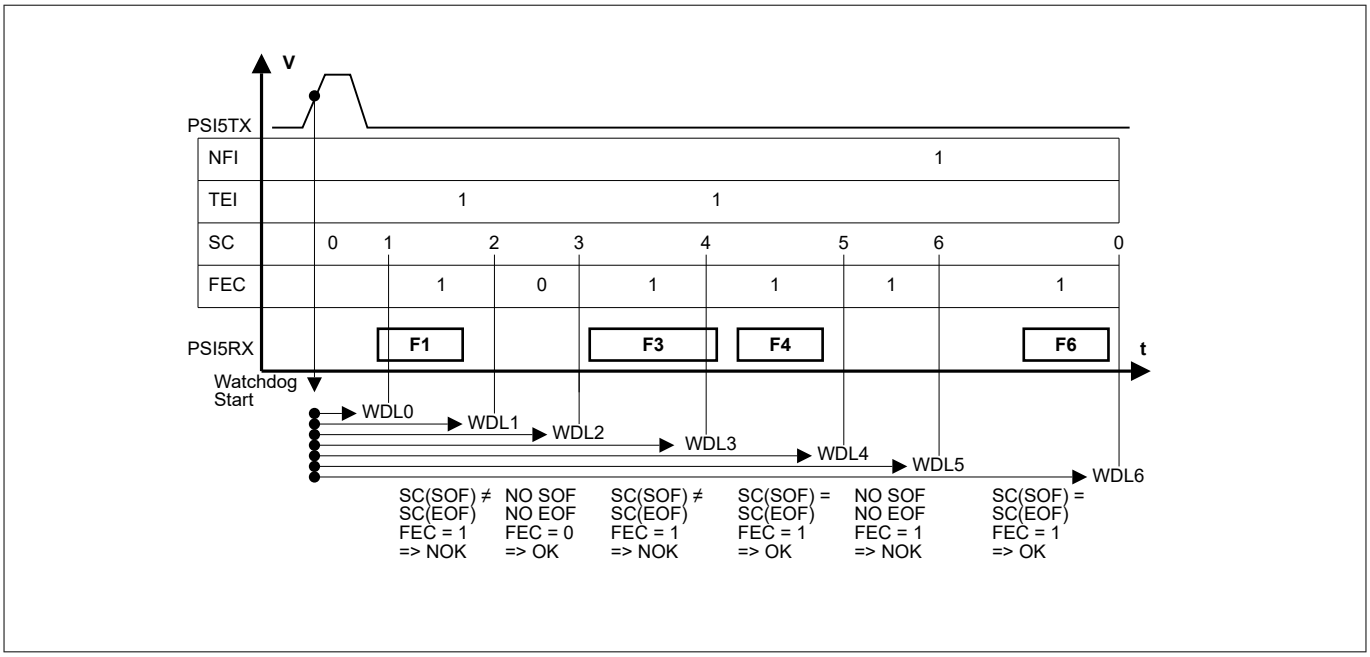


Figure 530 Watchdog timer limits

26.3.11 Receive data memory

The module design allows for usage in 2 ways without mode selection:

Ring buffer usage

The Receive Data Memory x is built as a ring buffer. That is, the first frame is stored in RDML/Hx0, the next in RDML/Hx1 and so on. At buffer 31, a wrap around is performed automatically. That is, the next frame is stored in buffer 0. RFCx.WRP shows the last location written and is automatically incremented after write.

Receive Buffer Overrun Interrupt Flag RBI is set, if a buffer is written before the SW cleared Receive Data Interrupt Overview Flag RDIOVx (unchecked or erroneous frame received or No Frame Received (NFI) message to be stored).

FIFO usage

Writing to the Receive Data Memory by the module moves the write pointer after writing.

Reading from RDFx 2 times with a 32 bit word access moves the read pointer to the next available buffer automatically after the read access. It clears as well all interrupt overview bits for this buffer line in NBIOVx, CRCIOVx, TEIOVx, RMIOVx, RSIOVx, RDIOVx, NFIOVx, MEIOVx. This allows for fully automatic DMA transfers without CPU intervention. The DMA simply needs to be set up in a way that it reads exactly the number of buffers configured with the FIFO warning level RFCx. The application SW configures “FIFO Warning Level Request Interrupt” FWI so that it is routed to the referring DMA channel as trigger source. Note that FWI does not need to be cleared to repetitively trigger the DMA.

If RFCx.WRAP is cleared and the read pointer (after the auto increment said above) equals the write pointer while reading from RDFx, a Receive Memory Underrun Interrupt INTSTATAx.RUI is flagged and the read pointer is not moved on automatically even if RDFx is read.

If RFCx.FRQ is set, the FIFO needs to be flushed by RFCx.FLU.

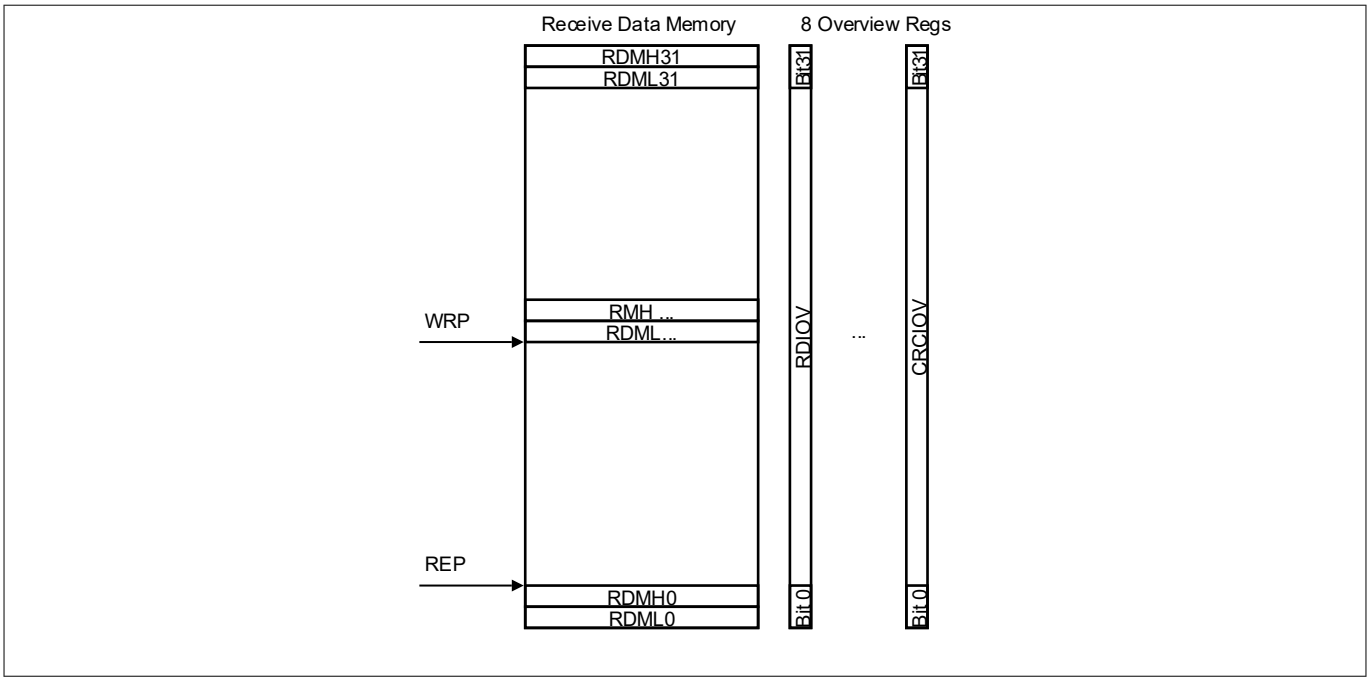


Figure 531 Receive data memory

26.3.12 Sync pulse control

The sync pulse generation provides the following functionality:

- Periodic sync pulses, driven by a common module time base (PGCx.PTE is set)
- Angle synchronous sync pulses, driven by GTM or eGTM based on PGCx.TRIGSEL (PGCx.ETE is set)
- Sync pulses for ECU to sensor communication (no pulse length extension) (SCRx.EPS is cleared)
- Sync pulses for ECU to sensor communication (with pulse length extension) (SCRx.EPS is set)
- Sync pulses provided by GTM or eGTM based on PGCx.TRIGSEL (Bypass PGCx.BYP set)
- Blank out time (switch off receiver during and after the sync pulse)

It is possible to switch on and off the periodic trigger or the angle synchronous trigger at any time. If data is available (SCRx.TOF is set), the pulses will be modulated according to the selected standard. SCRx.EPS selects between extended standard with pulse length modulation and legacy mode according to PSI5 standard V1.3 with pulse dropping (“tooth gap”). The minimum pulse length is defined by PGC.PLEN for both standards. For the extended standard, the pulses are prolonged by PGCx.DEL if a one is coded. Exception is the bypass path (BYP is set). These pulses are transmitted without any modulation.

Any combination of PTE, ETE and BYP can produce too long pulses.

An ECU to Sensor communication can be stopped at any point in time by setting the flush bit. This will clear the send register and flag TOF.

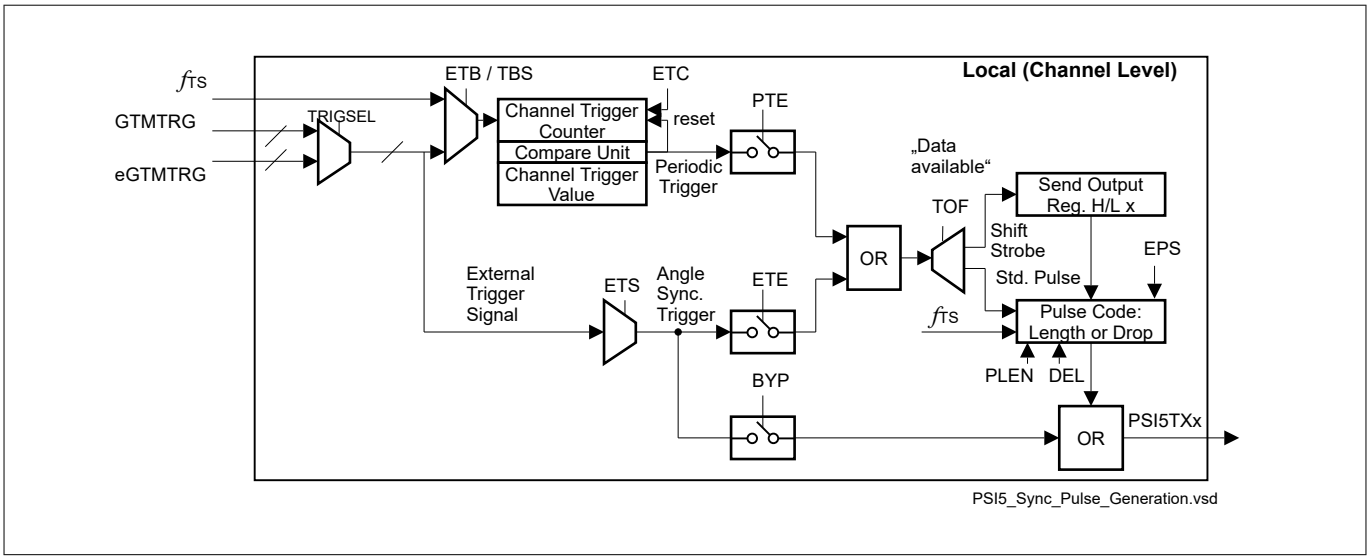


Figure 532 PSI5 sync pulse generation

Blank out time

BOT defines the length of the blank out time in TTS times (defined by FDRT) starting from the end of a sync pulse. The receiver is always switched off starting from the beginning of a sync pulse. BOT keeps the receiver switched off additionally after the sync pulse. This suppresses potential noise on the receive line after the pulse and so eases the design of the transceiver.

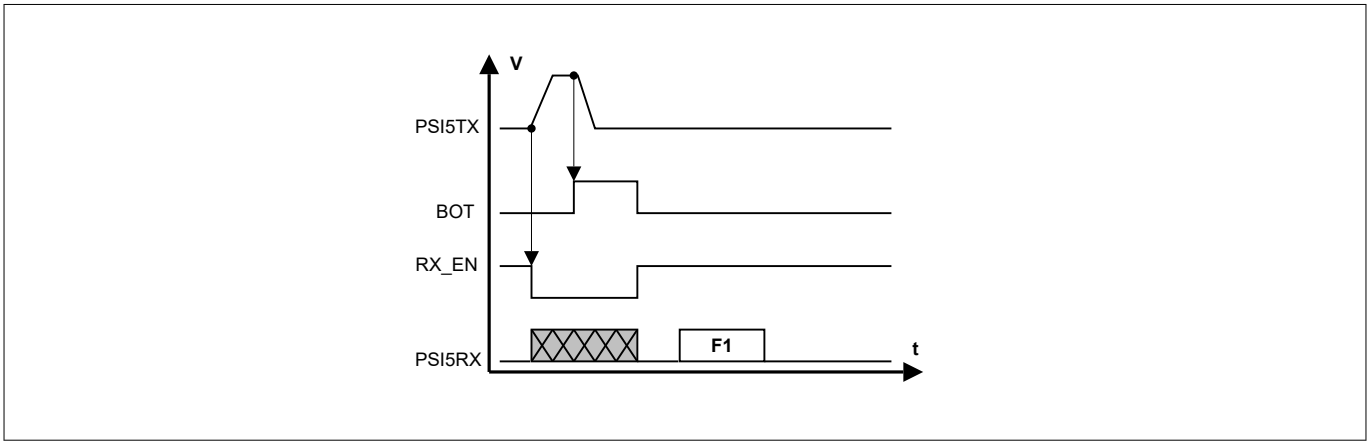


Figure 533 PSI5 blank out time

26.3.13 Channel trigger

The Channel Trigger Value Register CTVx contains a two cell reset counter CTC and CTV. CTV is the compare value of the channel trigger CTC. It allows for setting up periodic sync pulses for channel x.

The compare on equality allows to preset CTC and also configure an offset between the CTVx.CTCs. This allows for staggering periodic sync pulses that must have frequencies with common multiples (for example same frequency, double etc.). The Channel Trigger Value CTV is the compare value which clears the counter CTC on equality. At exactly this time, a sync pulse is triggered.

In this document, OFFSET = 65535 - “CTC preset value”

The Channel Trigger Value CTV is the compare value which clears the counter CTC. That is, when CTC preset is greater than CTV, the time to roll over is the offset.

When CTC preset is lesser than CTV, CTC has an advance.

Note that CTC can be written only if it is disabled by clearing GCR.ETCx.

Staggering the pulses might be required if the current source for the external PHYs can provide only limited current so that not all channels can be provided with the additional current that is required to apply the voltage increase representing a proper sync pulse.

For a synchronization of all counters CTVx.CTC, bits GCR.ETC0 can be used.

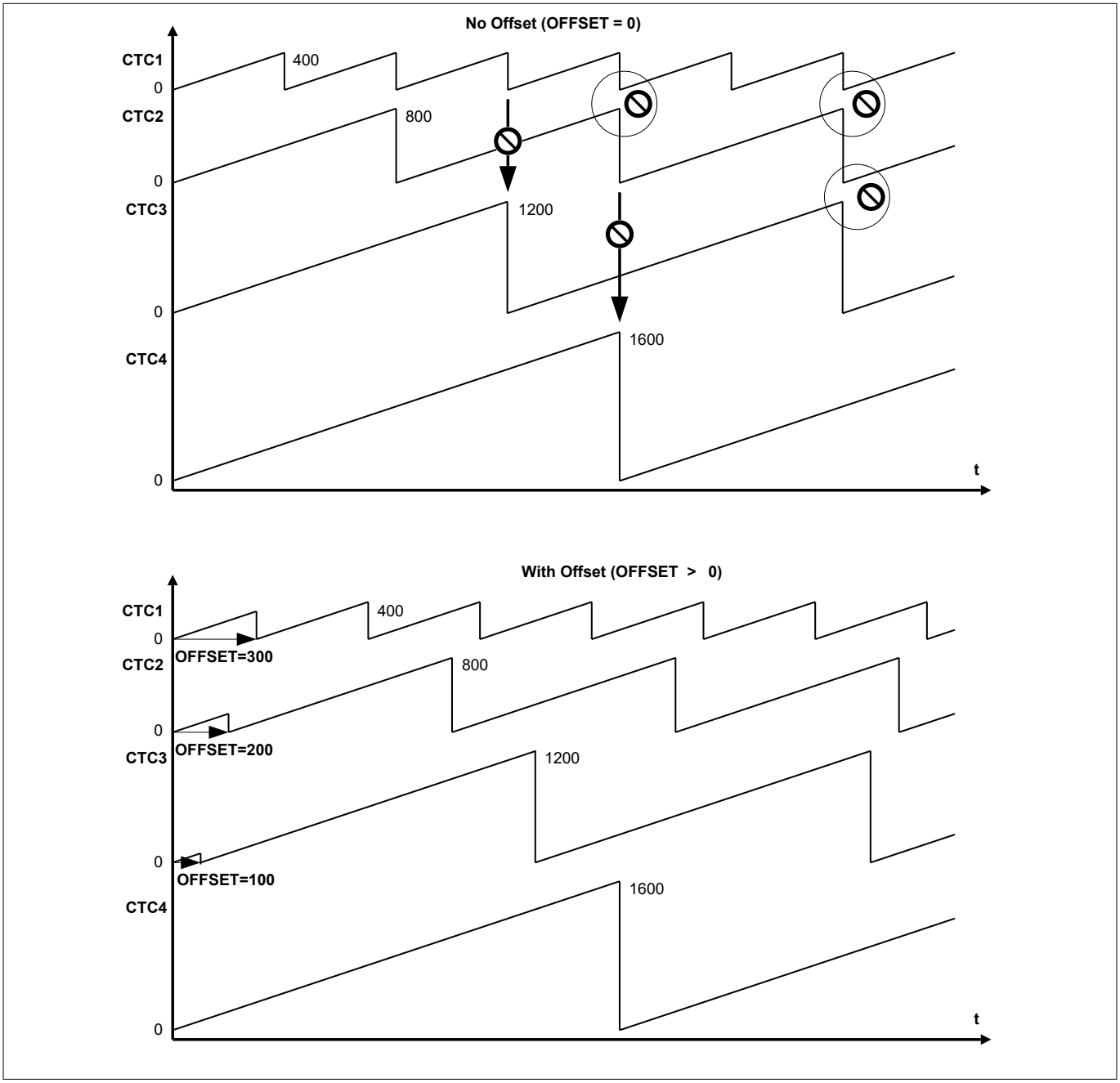


Figure 534 PSI5 pulse staggering

26.3.14 Send control

The Send Control Register SCRx contains control data bits required for sending data from the PSI5 module to the sensor / external PSI5 device.

Data is collected in SDRL/Hx (SDRLx and SDRHx) per channel and the control bits are collected in SCRx.

Send data preparation

The send control unit is built to transmit complete ECU to sensor frames. The figure below shows the send data preparation. Bit SCRx.TPF is set when send data preparation in the SDRL/Hx is in progress. Transmit Preparation Overflow Interrupt flag INTSTATAx.TPOI is set if the SDRL/Hx registers are written when the TPF bit is set. The old data is not overwritten. Send data is moved from SDRL/Hx to the Send Shift Registers. The bit INTSTATAx.TPI is set and TPF is cleared after data to be transferred has been moved from SDRL/H to SSRL/Hx. A new value can now be written to SDRL/Hx .

The Send Shift Registers SSRLX and SSRHx for channel x shows the updated send data after CRC, Stuffing and Start Sequence Insertion. Automatic CRC generation is controlled by SCRx.CRC. Bit SCRx.EPS controls the pulse length expansion. The automatic bit stuffing, depending on the EPS, is controlled by bit SCRx.BSC. When bit SCRx.STA is 0, there is no start sequence generated. Which means the payload starts from bit 0. When STA is set, there is an automatic start sequence generated. Transmit Shift Flag SCRx.TSF is set when the data in SSRL/Hx is waiting to be shifted to Send Output Registers.

Transmit Shift Overflow Interrupt flag INTSTATx.TSOI is set if SSRL/Hx registers are written when the TSF bit is set. The old data is not overwritten. The bit INTSTATAx.TSI is set and TSF is cleared after data to be transferred has been moved from SSRL/Hx to SORL/Hx. A new value can now be written to SSRL/Hx .

Send Output Registers SORLx and SORHx show the data that will be shifted to the pulse generator. Transmit Output flag SCRx.TOF is set when the data in the SORL/Hx registers are waiting to be transmitted to the sensor. Transmit Shift Overflow Interrupt flag INTSTATAx.TOOI is set if SORL/Hx registers are written when the TOF bit is set. The old data is not overwritten. The bit INTSTATAx.TOI is set and TOF is cleared after last bit to be transferred has been moved from SORL/H to the pulse generator. A new value can now be written to SORL/Hx.

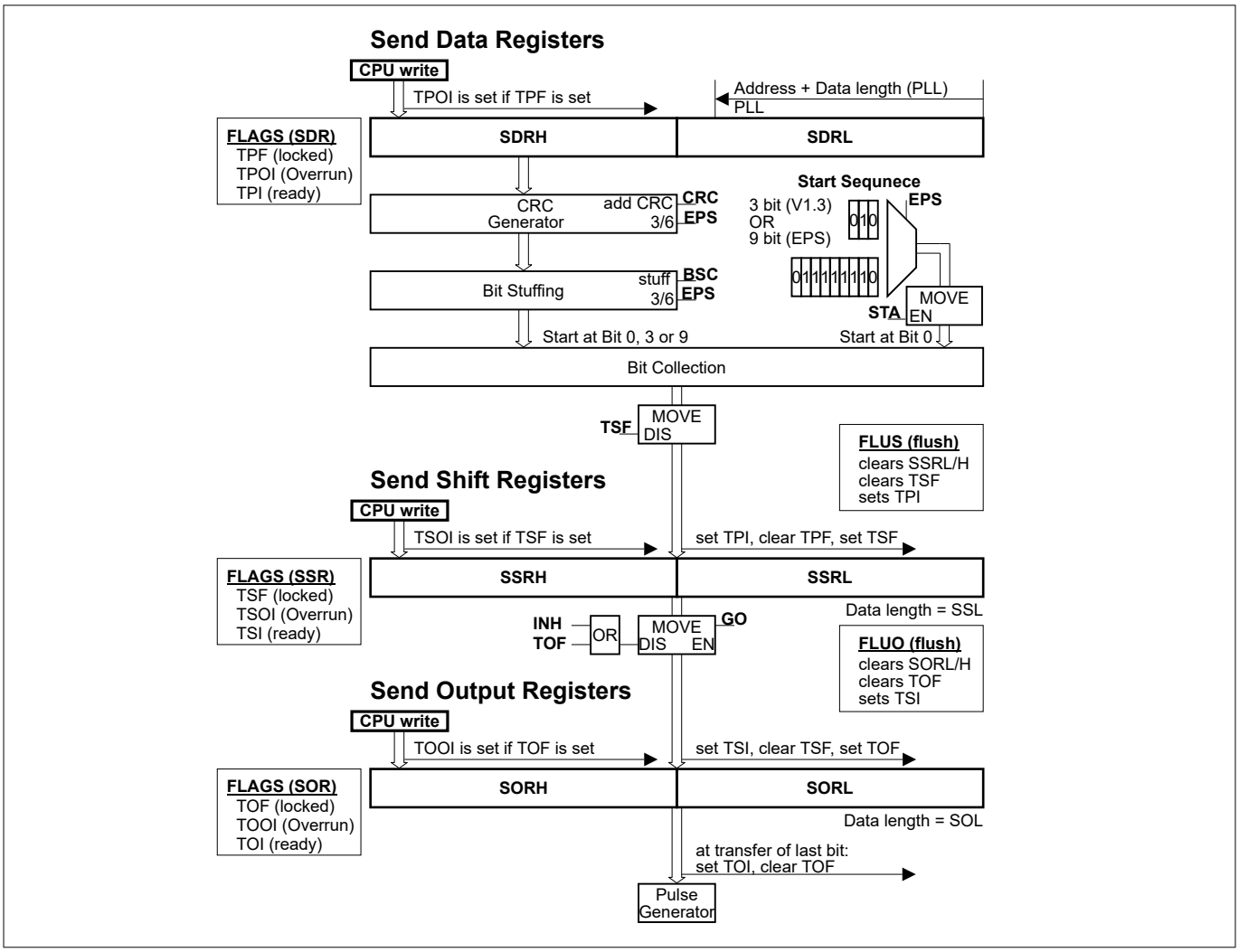


Figure 535 PSI5 send data preparation

26.3.15 Error detection capabilities

Each PSI5 channel can detect and signal the following error conditions:

Protocol level:

- No frame received/Manchester coding error in start bits (NFI)
- Messaging bits with Manchester coding error (MEI)
- Number of bits too small/Manchester coding error but messaging bits OK (NBI)
- Checksum error (CRCI)
- Frame not sent in expected time slot (TEI)
- Serial Communication: Wrong Status and Communication bit-field (WSI)
- Serial Communication: CRC error (SCRI)

Transfer Management Level:

- Receive Data Buffer Overrun (RBI)
- Receive Data Memory Overrun (RMI)
- FIFO Warning Level (FWI)
- Receive Memory Underrun (RUI)
- ECU to Sensor Data Buffer Underrun (TBI)

26.3.16 Interrupts

8 interrupt sources are available for the PSI5 module. For each trigger source of a channel x one interrupt can be selected in register INPx.

1. INPx.RDI indicates a receive data interrupt. It is activated when a received frame is moved to a Receive Data Register RDR. This happens if at least the start bits were received without Manchester coding error
2. INPx.SDI indicates a serial data receive interrupt when serial data communication is enabled
3. INPx.RSI indicates a receive frame success interrupt if none of the interrupts NFI, MEI, NBI, CRCI and TEI were detected. The referring interrupt must be enabled in GCR to be considered for RSI. This allows, for example, to use only the Receive Data Register RDRL/Hx and ignore the Receive Data Memory RDML/Hx. Both RDI and RSI will be issued together in normal use cases where reception is correct
4. INPx.RBI indicates a receive buffer overrun interrupt. It is activated when a new frame is transferred to a Receive Data Register RDR while the old value was still not read by the host (overwrite). Meaning the kernel wants to set any of the two interrupts RSI and RDI and finds any of these two interrupts already set
5. INPx.FWI is set if the configured warning level of the FIFOx was reached (see RFCx)
6. INPx.TBI is a single node pointer for the overflow interrupts TPOI, TSOI, TOOI seen in the send control. TPOI, TSOI and TOOI indicate a transfer (send) register overrun interrupt. They are set if data is written to one of the registers in the transmission chain while the referring register is locked (occupied). The data written is ignored in this case
7. INPx.TDI is a single node pointer for TPI, TSI and TOI interrupts. TPI, TSI and TOI indicate a transmit interrupt. They are activated when data is moved from a SDR to SSR (TPI), from SSR to SOR (TSI) and from SOR to the pulse generator (TOI)
8. INPx.ERRI is a single node pointer for TEI, NFI, NBI, MEI, CRCI, RUI, RMI, WSI, SOI, SCRI interrupts
   a. RMI is issued if no memory buffer is available. RMI is set after a frame has been received or a No Frame Received (NFI) message is to be stored while RDIOVx.RDI[RFCx.WRP] is set. That means the kernel wants to set flag INTSTATAx.RDI[RFCx.WRP] and finds this flag already set. The old data is overwritten by the new data in the buffer memory
   b. TEI, NBI, MEI and CRCI are protocol error interrupts. If one of the protocol interrupts is activated, data is to be treated as invalid according to the PSI5 standard. The payload data received should

match the configured payload size RCRAx.PDLy (y=0-5) for channel x and slot y. Note that the wrong data is still available in all buffers. In particular, on MEI the wrong data is still fed into the serial data analyzer (slow channel)

c. NFI is set, if a frame is missing in a slot configured by the watchdog timer. The module can be configured to store a complete message anyhow with flag NFI set. The slot count and the timestamp can be helpful for debugging. This provides a method for easy data handling, for example by using DMA transfers only without checking single interrupt bits

d. WSI, SOI, SCRI treat the interrupts referring to the Status and Communication nibble for serial communication

e. RUI is issued when there is a read operation of the Receive Data FIFO while no data is available

For acceleration of the interrupt service routine, a register INTOV is implemented that shows if an interrupt class (RSI, RDI, RBI, TDI, TBI, ERRI, SDI, FWI) is active on any of the channels. It holds a flag for each class which combines the states from all channels of each node pointer in interrupt node pointer registers INPx. This flag is automatically set if there is an interrupt pending for the node pointer which is enabled. It is automatically reset, if no more enabled interrupt is pending for interrupt class.

In addition, there are 8 overview registers per channel holding the flags NBI, CRCI, TEI, RMI, RSI, RDI, NFI, MEI for each buffer line: NBIOVx, CRCIOVx, TEIOVx, RMIOVx, RSIOVx, RDIOVx, NFIOVx, MEIOVx; x = [0 - No. of PSI5 channels-1]. For example, NBIOV0 holds 32 bits, one for each buffer line of channel 0 reflecting if NBI is set in this buffer line.

The interrupt request or the corresponding interrupt set bit (in register INTSETAx or INTSETBx) can trigger the interrupt generation at the selected interrupt node. The service request pulse is generated independently from the interrupt flag in register INTSTATAx or INTSTATBx. The interrupt flag can be cleared by software by writing to the corresponding bit in register INTCLRAx or INTCLRBx.

If more than one interrupt source is connected to the same interrupt node (in register INPx), the requests are combined to one common line.

26.4 Registers

26.4.1 Register overview - access mode glossary

Table 1182 Register overview - access mode glossary

Keyword	Description
E	Access protection using PROT register PROTE.
SE	Access protection using PROT register PROTSE.
APU-P	Protection group consisting of registers ACCEN_WRA, ACCEN_WRB, ACCEN_RDA, ACCEN_RDB, ACCEN_VM, ACCEN_PRS.
P	Access protection using *APU-P* registers.
BE	Always returns a Bus Error.
U	No access restrictions.
PROT	Access restrictions as defined in the PROT register access rules.

26.4.2 Registers overview - PSI5 (ascending offset address)

Table 1183 Registers overview - PSI5 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CLC	Clock Control Register	000_H	P	P, SV, E	Application Reset	4885
OCS	OCDS Control and Status Register	004_H	P	SV, P	Debug Reset	4886
ID	Module Identification Register	008_H	P	BE	PowerOn Reset	4886
RST_CTRLA	Reset Control Register A	$00C_H$	P	P, SV, E	Application Reset	4887
RST_CTRLB	Reset Control Register B	010_H	P	P, SV, E	Application Reset	4888
RST_STAT	Reset Status Register	014_H	P	BE	Application Reset	4888
PROTE	PROT Register Endinit	020_H	U	SV, PROT	Application Reset	4889
PROTSE	PROT Register Safe Endinit	024_H	U	SV, PROT	Application Reset	4891
ACCEN_WRA	Write access enable register A	040_H	U	SE, SV	Application Reset	4892
ACCEN_WRB	Write access enable register B	044_H	U	SE, SV	Application Reset	4893
ACCEN_RDA	Read access enable register A	048_H	U	SE, SV	Application Reset	4893

(table continues...)

Table 1183 (continued) Registers overview - PSI5 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ACCEN_RDB	Read access enable register B	$04C_H$	U	SE, SV	Application Reset	4894
ACCEN_VM	VM access enable register	050_H	U	SE, SV	Application Reset	4894
ACCEN_PRS	PRS access enable register	054_H	U	SE, SV	Application Reset	4895
FDR	Fractional Divider Register	060_H	P	SV, E, P	Kernel Reset	4896
FDRL	Fractional Divider Register for Lower Bit Rate	064_H	P	SV, E, P	Kernel Reset	4896
FDRH	Fractional Divider Register for Higher Bit Rate	068_H	P	SV, E, P	Kernel Reset	4897
FDRT	Fractional Divider Register for Timestamp	$06C_H$	P	SV, E, P	Kernel Reset	4898
TSRA	Timestamp Register A	070_H	P	SV, E, P	Kernel Reset	4899
TSRB	Timestamp Register B	074_H	P	SV, E, P	Kernel Reset	4901
TSRC	Timestamp Register C	078_H	P	SV, E, P	Kernel Reset	4902
GCR	Global Control Register	080_H	P	SV, E, P	Kernel Reset	4903
IOCRx	Input and Output Control Register x	084_H+x*90_H	P	SV, E, P	Kernel Reset	4904
RCRAx	Receiver Control Register A x	088_H+x*90_H	P	SV, E, P	Kernel Reset	4906
RCRBx	Receiver Control Register B x	$08C_H$+x*90_H	P	SV, E, P	Kernel Reset	4907
RCRCx	Receiver Control Register C x	090_H+x*90_H	P	SV, E, P	Kernel Reset	4908
WDTxw	Watchdog Timer Register xw	094_H+x*90_H+w*4	P	SV, E, P	Kernel Reset	4909
RSRx	Receive Status Register x	$0B0_H$+x*90_H	P	BE	Kernel Reset	4910
SDSxz	Serial Data and Status Register xz	$0B4_H$+x*90_H+z*4	P	BE	Kernel Reset	4910
SPTSCx	Start of Pulse Timestamp Capture Register x	$0CC_H$+x*90_H	P	BE	Kernel Reset	4911
SFTSCx	Start of Frame Timestamp Capture Register x	$0D0_H$+x*90_H	P	BE	Kernel Reset	4912
RDRLx	Receive Data Register Low x	$0D4_H$+x*90_H	P	BE	Kernel Reset	4912
RDRHx	Receive Data Register High x	$0D8_H$+x*90_H	P	BE	Kernel Reset	4913
PGCx	Pulse Generation Control Register x	$0DC_H$+x*90_H	P	SV, E, P	Kernel Reset	4915

(table continues...)

Table 1183 (continued) Registers overview - PSI5 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CTVx	Channel Trigger Value Register x	$0E0_H$+x*90_H	P	SV, E, P	Kernel Reset	4917
SCRx	Send Control Register x	$0E4_H$+x*90_H	P	SV, E, P	Kernel Reset	4917
SDRLx	Send Data Register Low x	$0E8_H$+x*90_H	P	P	Kernel Reset	4920
SDRHx	Send Data Register High x	$0EC_H$+x*90_H	P	P	Kernel Reset	4921
SSRLx	Send Shift Register Low x	$0F0_H$+x*90_H	P	P	Kernel Reset	4921
SSRHx	Send Shift Register High x	$0F4_H$+x*90_H	P	P	Kernel Reset	4922
SORLx	Send Output Register Low x	$0F8_H$+x*90_H	P	P	Kernel Reset	4922
SORHx	Send Output Register High x	$0FC_H$+x*90_H	P	P	Kernel Reset	4923
INTOV	Interrupt Overview Register	$2C4_H$	P	BE	Kernel Reset	4923
INPx	Interrupt Node Pointer Register x	$2C8_H$+x*4	P	SV, E, P	Kernel Reset	4925
INTSTATAx	Interrupt Status Register A x	$2DC_H$+x*4	P	BE	Kernel Reset	4927
INTSTATBx	Interrupt Status Register B x	$2EC_H$+x*4	P	BE	Kernel Reset	4931
INTSETAx	Interrupt Set Register A x	$2FC_H$+x*4	P	SV, E, P	Kernel Reset	4933
INTSETBx	Interrupt Set Register B x	$30C_H$+x*4	P	SV, E, P	Kernel Reset	4935
INTCLRAx	Interrupt Clear Register A x	$31C_H$+x*4	P	P	Kernel Reset	4935
INTCLRBx	Interrupt Clear Register B x	$32C_H$+x*4	P	P	Kernel Reset	4938
INTENAx	Interrupt Enable Register A x	$33C_H$+x*4	P	SV, E, P	Kernel Reset	4938
INTENBx	Interrupt Enable Register B x	$34C_H$+x*4	P	SV, E, P	Kernel Reset	4940
RFCx	Receive FIFO Control Register x	378_H+x*4	P	P	Kernel Reset	4941
RDFx	Receive Data FIFO x	388_H+x*4	P	BE	Kernel Reset	4942
RSIOVx	RSI Overview Register x	398_H+x*4	P	BE	Kernel Reset	4943

(table continues...)

Table 1183 (continued) Registers overview - PSI5 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
RMIOVx	RMI Overview Register x	$3A8_H$+x*4	P	BE	Kernel Reset	4943
NBIOVx	NBI Overview Register x	$3B8_H$+x*4	P	BE	Kernel Reset	4944
TEIOVx	TEI Overview Register x	$3C8_H$+x*4	P	BE	Kernel Reset	4944
CRCIOVx	CRCI Overview Register x	$3D8_H$+x*4	P	BE	Kernel Reset	4945
RDIOVx	RDI Overview Register x	$3E8_H$+x*4	P	BE	Kernel Reset	4945
NFIOVx	NFI Overview Register x	$3F8_H$+x*4	P	BE	Kernel Reset	4946
MEIOVx	MEI Overview Register x	408_H+x*4	P	BE	Kernel Reset	4946
RSISETx	RSI Overview Set Register x	418_H+x*4	P	P	Kernel Reset	4947
RMISETx	RMI Overview Set Register x	428_H+x*4	P	P	Kernel Reset	4947
NBISETx	NBI Overview Set Register x	438_H+x*4	P	P	Kernel Reset	4948
TEISETx	TEI Overview Set Register x	448_H+x*4	P	P	Kernel Reset	4948
CRCISETx	CRCI Overview Set Register x	458_H+x*4	P	P	Kernel Reset	4949
RDISETx	RDI Overview Set Register x	468_H+x*4	P	P	Kernel Reset	4949
NFISETx	NFI Overview Set Register x	478_H+x*4	P	P	Kernel Reset	4950
MEISETx	MEI Overview Set Register x	488_H+x*4	P	P	Kernel Reset	4950
RSICLRx	RSI Overview Clear Register x	498_H+x*4	P	P	Kernel Reset	4951
RMICLRx	RMI Overview Clear Register x	$4A8_H$+x*4	P	P	Kernel Reset	4951
NBICLRx	NBI Overview Clear Register x	$4B8_H$+x*4	P	P	Kernel Reset	4952
TEICLRx	TEI Overview Clear Register x	$4C8_H$+x*4	P	P	Kernel Reset	4952
CRCICLRx	CRCI Overview Clear Register x	$4D8_H$+x*4	P	P	Kernel Reset	4953
RDICLRx	RDI Overview Clear Register x	$4E8_H$+x*4	P	P	Kernel Reset	4953

(table continues...)

Table 1183 (continued) Registers overview - PSI5 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
NFICLRx	NFI Overview Clear Register x	$4F8_H$+x*4	P	P	Kernel Reset	4954
MEICLRx	MEI Overview Clear Register x	508_H+x*4	P	P	Kernel Reset	4954
RDMLxy	Receive Data Memory Low xy	600_H+x*100_H+y*8	P	BE	Kernel Reset	4955
RDMHxy	Receive Data Memory High xy	604_H+x*100_H+y*8	P	BE	Kernel Reset	4955

26.4.3 Clock Control Register

The Clock Control Register CLC allows the programmer to adapt the functionality and power consumption of the module to the requirements of the application.

Register CLC controls the module clock signal and the reactivity to the sleep signal.

CLC Offset address: 000_H

Clock Control Register Application Reset value: $0000\ 0003_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												EDIS	0	DISS	DISR
r												rw	r	rh	rw

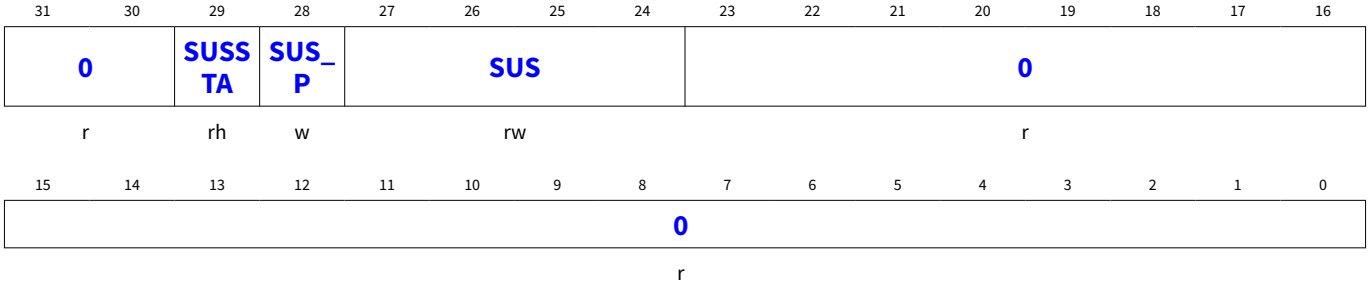
Field	Bits	Type	Description
DISR	0	rw	**Module Disable Request Bit** Used for enable/disable control of the module. 0_B On request: enable the module clock 1_B Off request: stop the module clock
DISS	1	rh	**Module Disable Status Bit** 0_B Module clock is enabled 1_B Off: module is not clocked
EDIS	3	rw	**Sleep mode Enable Control** Used to control the module's reaction to sleep mode. 0_B Sleep mode request is enabled and functional 1_B Module disregards the sleep mode control signal
0	2, 31:4	r	**Reserved** Read as 0; should be written with 0.

26.4.4 OCDS Control and Status Register

The OCDS Control and Status register OCS controls the debug and trace behavior by selecting suspend modes and OTGB trigger sets. When OCDS is disabled the suspend control is ineffective

OCS Offset address: 004_H

OCDS Control and Status Register Debug Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
SUS	27:24	rw	**OCDS Suspend Control** Controls the sensitivity to the suspend signal coming from the OCDS Trigger Switch (OTGS) Not listed combinations are reserved. 0_H Will not suspend 1_H Hard suspend: Clocks will be disabled immediately 2_H Reserved 3_H Soft suspend mode Clocks will be disabled after kernel acknowledge.
SUS_P	28	w	**SUS Write Protection** SUS is only written when SUS_P is 1, otherwise unchanged. Read as 0.
SUSSTA	29	rh	**Suspend State** 0_B Module is not (yet) suspended 1_B Module is suspended
0	23:0, 31:30	r	**Reserved** Read as 0; should be written with 0.

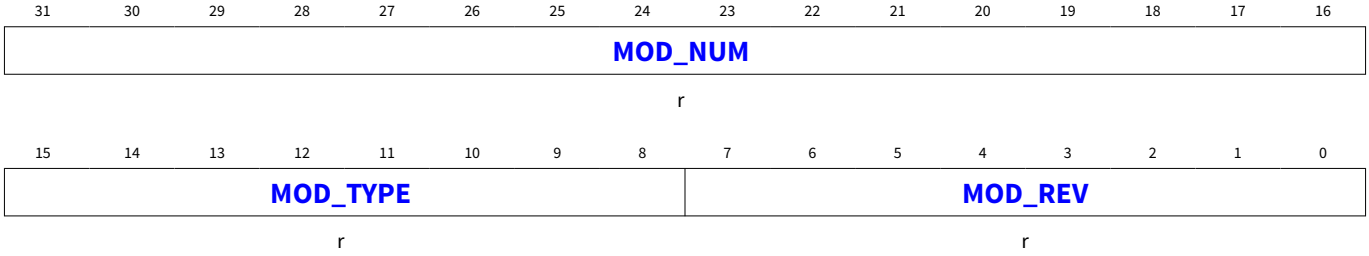
Table 1184 Access mode restrictions of OCS sorted by descending priority

Mode name	Access mode		Description
write 1 to .**SUS_P**	rw	SUS	Set SUS_P during write access
(default)	r	SUS	

26.4.5 Module Identification Register

ID Offset address: 008_H

Module Identification Register PowerOn Reset value: $00C3\ C001_H$

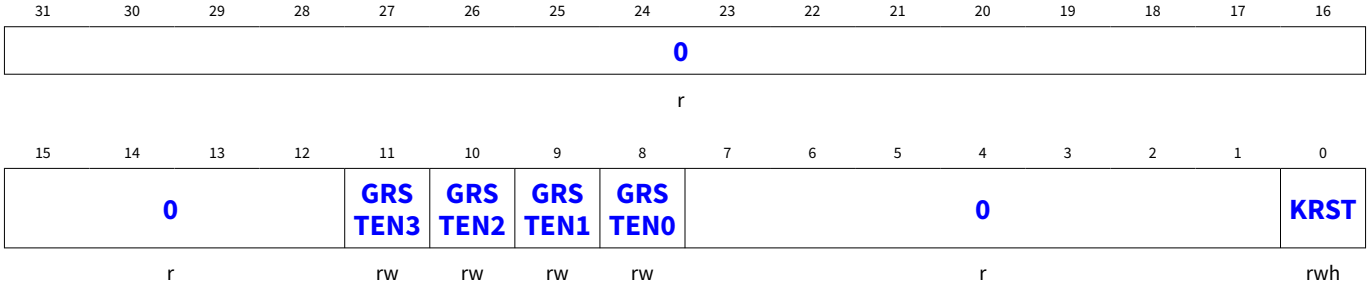


Field	Bits	Type	Description
MOD_REV	7:0	r	**Module Revision** Indicates the revision number of the implementation.
MOD_TYPE	15:8	r	**Module Type** This internal marker is fixed to $C0_H$.
MOD_NUM	31:16	r	**Module Number** Indicates the module identification number.

26.4.6 Reset Control Register A

RST_CTRLA Offset address: $00C_H$

Reset Control Register A Application Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
KRST	0	rwh	**Kernel Reset** Request a kernel reset. The requested reset is executed if CTRLA.KRST=1 and CTRLB.KRST=1. KRST is cleared after the kernel reset was executed. 0_B No action 1_B A kernel reset was requested
GRSTENx (x=0-3)	x+8	rw	**Enable for Global Module Reset Group x** 0_B Global module reset group x does not have any effect 1_B Global module reset group x results in a kernel reset
0	7:1, 31:12	r	**Reserved** Read as 0; should be written with 0.

26.4.7 Reset Control Register B

RST_CTRLB Offset address: 010_H

Reset Control Register B Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
STATCLR								0							
w								r							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															KRST
r															rwh

Field	Bits	Type	Description
KRST	0	rwh	**Kernel Reset** Request a kernel reset. The requested reset is executed if CTRLA.KRST=1 and CTRLB.KRST=1. KRST is cleared after the kernel reset was executed. 0_B No action 1_B A kernel reset was requested
STATCLR	31	w	**Kernel Reset Status Clear** Clears all status bits in RST_STAT when 1 is written. Read will return 0. 0_B No action 1_B Write with ´1´ clears bits STAT.GRSTx and bit STAT.KRST .
0	30:1	r	**Reserved** Read as 0; should be written with 0.

26.4.8 Reset Status Register

The reset status register contains the status bits for kernel reset (KRST) and the global module reset groups (GRSTx).

RST_STAT Offset address: 014_H

Reset Status Register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				GRST3	GRST2	GRST1	GRST0	0							KRST
r				rh	rh	rh	rh	r							rh

Field	Bits	Type	Description
KRST	0	rh	**Kernel Reset Status** Indicates an executed kernel reset. RST_STAT.KRST is set after the execution of a kernel reset in the same clock cycle in which the reset bits are cleared. Clear KRST by setting bit STATCLR in register RST_CTRLB.
GRSTx (x=0-3)	x+8	rh	**Status for Global Module Reset Group x** 0_B Reset was not triggered by Global Reset Group x 1_B Reset was triggered by Global Reset Group x
0	7:1, 31:12	r	**Reserved** Read as 0.

26.4.9 PROT Register Endinit

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner and in some PROT states, the secure master to update the PROT state as described in the PROT mechanism.

The PROTE register controls lock / unlock of the local Endinit (E) protected control registers.

PROTE Offset address: 020_H

PROT Register Endinit Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OWEN	ODEF	TAGID						PRSEN	PRS			VMEN	VM		
w	rw	rw						rw	rw			rw	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												SWEN	STATE		
r												w	rwh		

Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)

(table continues...)

(continued)

Field	Bits	Type	Description
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
VM	18:16	rw	**Virtual machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual machine definition enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.
PRSEN	23	rw	**Protection set definition enable for PROT owner** 0_B **Disable**: PRS is not part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG-ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; should be written with 0.

Table 1185 Access mode restrictions of PROTE sorted by descending priority

Mode name	Access mode		Description
write 1 to .**SWEN**	rwh	STATE	
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

26.4.10 PROT Register Safe Endinit

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner and in some PROT states, the secure master to update the PROT state as described in the PROT mechanism.

The PROTSE register controls lock / unlock of the local Safe Endinit (SE) protected control registers.

PROTSE Offset address: 024_H

PROT Register Safe Endinit Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OWEN	ODEF	TAGID						PRSEN	PRS			VMEN	VM		
w	rw	rw						rw	rw			rw	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												SWEN	STATE		
r												w	rwh		

Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
VM	18:16	rw	**Virtual machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual machine definition enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.

(table continues...)

(continued)

Field	Bits	Type	Description
PRSEN	23	rw	**Protection set definition enable for PROT owner** 0_B **Disable**: PRS is not part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG-ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; should be written with 0.

Table 1186 Access mode restrictions of PROTSE sorted by descending priority

Mode name	Access mode		Description
write 1 to .**SWEN**	rwh	STATE	
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

26.4.11 Write access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

ACCEN_WRA Offset address: 040_H

Write access enable register A Application Reset value: $1000\ 0003_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access

26.4.12 Write access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

ACCEN_WRB Offset address: 044_H

Write access enable register B Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access
0	31:8	r	**Reserved** Read as 0; should be written with 0.

26.4.13 Read access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

ACCEN_RDA Offset address: 048_H

Read access enable register A Application Reset value: $FFFF\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access

26.4.14 Read access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

ACCEN_RDB Offset address: $04C_H$

Read access enable register B Application Reset value: $0000\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access
0	31:8	r	**Reserved** Read as 0; should be written with 0.

26.4.15 VM access enable register

This register defines which virtual machine encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each VMx.

ACCEN_VM Offset address: 050_H

VM access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for VM ID q** This bit enables read access to the access protected region for transactions with the VM ID q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for VM ID q** This bit enables write access to the access protected region for transactions with the VM ID q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

26.4.16 PRS access enable register

This register defines which protection register set encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each PRSx.

ACCEN_PRS Offset address: 054_H

PRS access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

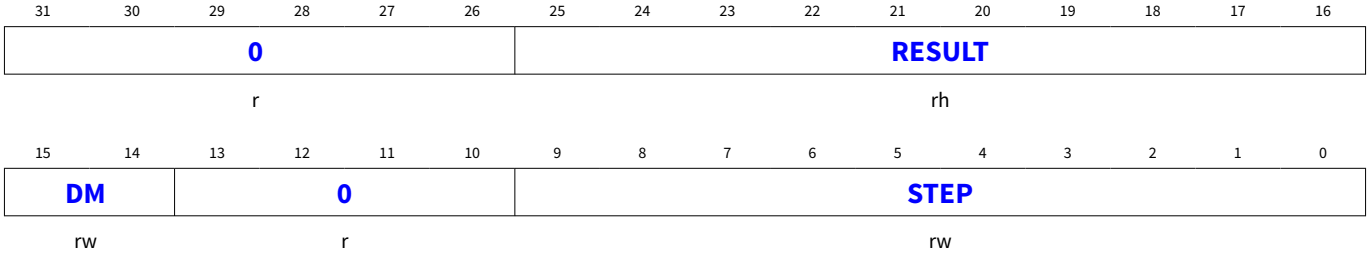
Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for PRS q** This bit enables read access to the access protected region for transactions with the PRS q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for PRS q** This bit enables write access to the access protected region for transactions with the PRS q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

26.4.17 Fractional Divider Register

The Fractional Divider Register controls the input clock $f_{fracdiv}$.

FDR Offset address: 060_H

Fractional Divider Register Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
STEP	9:0	rw	**Step Value** Reload or addition value for RESULT.
DM	15:14	rw	**Divider mode** DM selects normal or fractional divider mode. 00_B Fractional divider is switched off; no output clock is generated. The Reset External Divider signal is 1. RESULT is not updated 01_B Normal Divider mode selected. 10_B Fractional Divider mode selected. 11_B Fractional divider is switched off; no output clock is generated. RESULT is not updated.
RESULT	25:16	rh	**Result Value** Bit-field for the addition result.
0	13:10, 31:26	r	**Reserved** Read as 0; should be written with 0.

Related information

Bit rate generation on page 4867

26.4.18 Fractional Divider Register for Lower Bit Rate

The Fractional Divider Register for Lower Bit Rate contains the divider that defines the time resolution of the f_{125} . It divides $f_{fracdiv}$ by a factor and provides f_{125} to all channels.

FDRL Offset address: 064_H

Fractional Divider Register for Lower Bit Rate Kernel Reset value: $0000\ 0000_H$

Field	Bits	Type	Description
STEP	9:0	rw	**Step Value** Reload or addition value for RESULT.
DM	15:14	rw	**Divider mode** DM selects normal or fractional divider mode. 00_B Fractional divider is switched off; no output clock is generated. 01_B Normal Divider mode selected. 10_B Fractional Divider mode selected. 11_B Fractional divider is switched off; no output clock is generated. RESULT is not updated.
RESULT	25:16	rh	**Result Value** Bit-field for the addition result.
0	13:10, 31:26	r	**Reserved** Read as 0; should be written with 0.

Related information

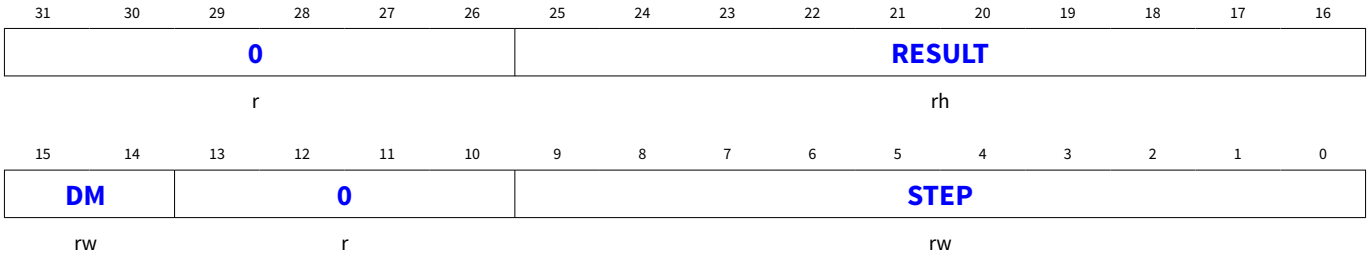
Bit rate generation on page 4867

26.4.19 Fractional Divider Register for Higher Bit Rate

The Fractional Divider Register for Higher Bit Rate contains the divider that defines the time resolution of f_{189} . It divides $f_{fracdiv}$ by a factor and provides f_{189} to all channels.

FDRH Offset address: 068_H

Fractional Divider Register for Higher Bit Rate Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
STEP	9:0	rw	**Step Value** Reload or addition value for RESULT.
DM	15:14	rw	**Divider mode** DM selects normal or fractional divider mode. 00_B Fractional divider is switched off; no output clock is generated. 01_B Normal Divider mode selected. 10_B Fractional Divider mode selected. 11_B Fractional divider is switched off; no output clock is generated. RESULT is not updated.

(table continues...)

(continued)

Field	Bits	Type	Description
RESULT	25:16	rh	**Result Value** Bit-field for the addition result.
0	13:10, 31:26	r	**Reserved** Read as 0; should be written with 0.

Related information

Bit rate generation on page 4867

26.4.20 Fractional Divider Register for Timestamp

The Fractional Divider Register for Timestamp contains the divider that defines the time resolution of the Timestamp Registers TSRA/B/C and the sync pulse time base. It divides f_{PSI5} by a factor. It contains as well the bits for reset control of the timestamp counters.

FDRT Offset address: $06C_H$

Fractional Divider Register for Timestamp Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECEC	ECEB	ECEA	ECS			RESULT									
rw	rw	rw	rw			rh									

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DM		0			TRIG SEL	STEP									
rw		r			rw	rw									

Field	Bits	Type	Description
STEP	9:0	rw	**Step Value** Reload or addition value for RESULT.
TRIGSEL	10	rw	**Trigger select** Trigger source selection bit 0_B **GTMTRG**: GTM GTM trigger 1_B **eGTMTRG**: eGTM eGTM trigger
DM	15:14	rw	**Divider mode** DM selects normal or fractional divider mode. 00_B Fractional divider is switched off; no output clock is generated. 01_B Normal Divider mode selected. 10_B Fractional Divider mode selected. 11_B Fractional divider is switched off; no output clock is generated. RESULT is not updated.
RESULT	25:16	rh	**Result Value** Bit-field for the addition result.

(table continues...)

(continued)

Field	Bits	Type	Description
ECS	28:26	rw	**External Timestamp Clear Source Select** Selects the external trigger line that clears the global timestamp counters TSRA/B/C.CTS if this is enabled by ECEA/ECEB/ECEC. Channel must be disabled if changed (GCR.CEN = 0). 000_B TRIG0 001_B TRIG1 … 101_B TRIG5 110_B Reserved, no trigger selected 111_B Reserved, no trigger selected
ECEA	29	rw	**External Timestamp Clear Enable A** Enables the external trigger line selected by ECS to clear the global timestamp counter TSRA.CTS on rising edge of the external trigger. 0_B Disabled 1_B Enabled
ECEB	30	rw	**External Timestamp Clear Enable B** Enables the external trigger line selected by ECS to clear the global timestamp counter TSRB.CTS on rising edge of the external trigger. 0_B Disabled 1_B Enabled
ECEC	31	rw	**External Timestamp Clear Enable C** Enables the external trigger line selected by ECS to clear the global timestamp counter TSRC.CTS on rising edge of the external trigger. 0_B Disabled 1_B Enabled
0	13:11	r	**Reserved** Read as 0; should be written with 0.

Related information

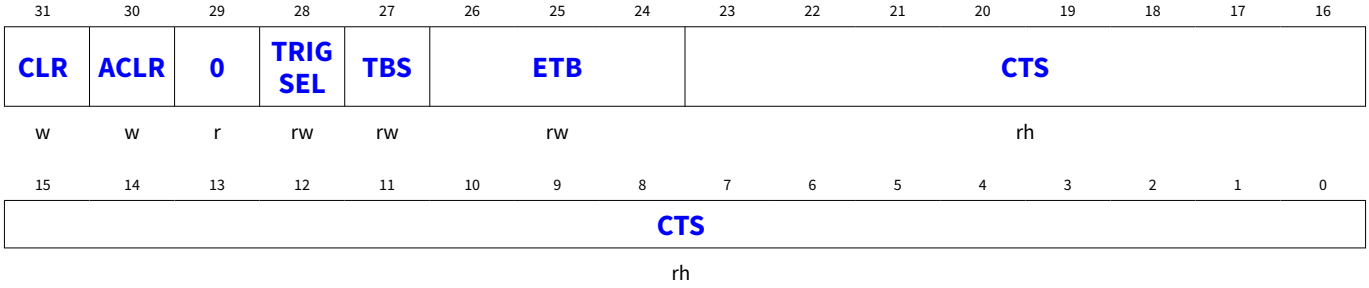
Bit rate generation on page 4867

26.4.21 Timestamp Register A

This Timestamp Register contains read-only information about the current time given in clock cycles of f_{TS} or $f_{GTM/eGTM}$ since last reset while module was clocked. It also contains the control bits for selection of input clock and reset of this counter.

TSRA — Offset address: 070_H

Timestamp Register A — Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
CTS	23:0	rh	**Current Timestamp for the Module** This bit-field shows the current timestamp.
ETB	26:24	rw	**External Time Base Select** Selects the external clock line for counter CTS. Channel must be disabled if changed (GCR.CEN = 0). 000_B TRIG0 001_B TRIG1 … 101_B TRIG5 110_B Reserved, no trigger selected 111_B Reserved, no trigger selected
TBS	27	rw	**Time Base Select** This bit selects the clock source for CTS 0_B Internal, CTS counts in clock cycles of f_{TS} 1_B External, CTS counts in clock cycles of f_{GTM} , eGTM
TRIGSEL	28	rw	**Trigger select** Trigger source selection bit 0_B **GTMTRG**: GTM GTM trigger 1_B **eGTMTRG**: eGTM eGTM trigger
ACLR	30	w	**Clear All Current Timestamp Counters** If set, this bit clears TSRA/B/C.CTS. TSRA/B/C.CTS count on, starting from 0. This bit is automatically cleared by HW and will always be read as zero.
CLR	31	w	**Clear Current Timestamp for the Module** If set, this bit clears CTS. CTS counts on, starting from 0. This bit is automatically cleared by HW and will always be read as zero.
0	29	r	**Reserved** Read as 0; should be written with 0

26.4.22 Timestamp Register B

This Timestamp Register contains read-only information about the current time given in clock cycles of f_{TS} or $f_{GTM/eGTM}$ since last reset while module was clocked. It also contains the control bits for selection of input clock and reset of this counter.

TSRB Offset address: 074_H

Timestamp Register B Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CLR	**ACLR**	**0**	**TRIG SEL**	**TBS**	**ETB**			**CTS**							
w	w	r	rw	rw	rw			rh							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CTS															
rh															

Field	Bits	Type	Description
CTS	23:0	rh	**Current Timestamp for the Module** This bit-field shows the current timestamp.
ETB	26:24	rw	**External Time Base Select** Selects the external clock line for counter CTS. Channel must be disabled if changed (GCR.CEN = 0). 000_B TRIG0 001_B TRIG1 … 101_B TRIG5 110_B Reserved, no trigger selected 111_B Reserved, no trigger selected
TBS	27	rw	**Time Base Select** This bit selects the clock source for CTS 0_B Internal, CTS counts in clock cycles of f_{TS} 1_B External, CTS counts in clock cycles of f_{GTM} , $_{eGTM}$
TRIGSEL	28	rw	**Trigger select** Trigger source selection bit 0_B **GTMTRG**: GTM GTM trigger 1_B **eGTMTRG**: eGTM eGTM trigger
ACLR	30	w	**Clear All Current Timestamp Counters** If set, this bit clears TSRA/B/C.CTS. TSRA/B/C.CTS count on, starting from 0. This bit is automatically cleared by HW and will always be read as zero.
CLR	31	w	**Clear Current Timestamp for the Module** If set, this bit clears CTS. CTS counts on, starting from 0. This bit is automatically cleared by HW and will always be read as zero.

(table continues...)

(continued)

Field	Bits	Type	Description
0	29	r	**Reserved** Read as 0; should be written with 0

26.4.23 Timestamp Register C

This Timestamp Register contains read-only information about the current time given in clock cycles of f_{TS} or $f_{GTM/eGTM}$ since last reset while module was clocked. It also contains the control bits for selection of input clock and reset of this counter.

TSRC Offset address: 078_H

Timestamp Register C Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CLR	ACLR	0	TRIG SEL	TBS	ETB			CTS							
w	w	r	rw	rw	rw			rh							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CTS															
rh															

Field	Bits	Type	Description
CTS	23:0	rh	**Current Timestamp for the Module** This bit-field shows the current timestamp.
ETB	26:24	rw	**External Time Base Select** Selects the external clock line for counter CTS. Channel must be disabled if changed (GCR.CEN = 0). 000_B TRIG0 001_B TRIG1 ... 101_B TRIG5 110_B Reserved, no trigger selected 111_B Reserved, no trigger selected
TBS	27	rw	**Time Base Select** This bit selects the clock source for CTS 0_B Internal, CTS counts in clock cycles of f_{TS} 1_B External, CTS counts in clock cycles of f_{GTM} , eGTM
TRIGSEL	28	rw	**Trigger select** Trigger source selection bit 0_B **GTMTRG**: GTM GTM trigger 1_B **eGTMTRG**: eGTM eGTM trigger

(table continues...)

(continued)

Field	Bits	Type	Description
ACLR	30	w	**Clear All Current Timestamp Counters** If set, this bit clears TSRA/B/C.CTS. TSRA/B/C.CTS count on, starting from 0. This bit is automatically cleared by HW and will always be read as zero.
CLR	31	w	**Clear Current Timestamp for the Module** If set, this bit clears CTS. CTS counts on, starting from 0. This bit is automatically cleared by HW and will always be read as zero.
0	29	r	**Reserved** Read as 0; should be written with 0

26.4.24 Global Control Register

The Global Control Register defines module wide settings.

The first 5 bits define which error flags are considered by RSI interrupt. RSI indicates that the referring frame is free of the selected errors. Each of these errors can be selected: NFI, TEI, NBI, MEI, CRCI.

GCR Offset address: 080_H

Global Control Register Kernel Reset value: $0000\ 001F_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0											CEN4	CEN3	CEN2	CEN1	CEN0
r											rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			ETC4	ETC3	ETC2	ETC1	ETC0	0			TEI	NFI	MEI	NBI	CRCI
r			rw	rw	rw	rw	rw	r			rw	rw	rw	rw	rw

Field	Bits	Type	Description
CRCI	0	rw	**CRCI** If bit is set, CRCI is selected to be checked for triggering RSI interrupt. No RSI is triggered in case of CRCI error
NBI	1	rw	**NBI** If bit is set, NBI is selected to be checked for triggering RSI interrupt. No RSI is triggered in case of NBI error
MEI	2	rw	**MEI** If bit is set, MEI is selected to be checked for triggering RSI interrupt. No RSI is triggered in case of MEI error
NFI	3	rw	**NFI** If bit is set, NFI is selected to be checked for triggering RSI interrupt. No RSI is triggered in case of NFI error
TEI	4	rw	**TEI** If bit is set, TEI is selected to be checked for triggering RSI interrupt. No RSI is triggered in case of TEI error

(table continues...)

(continued)

Field	Bits	Type	Description
ETCx (x=0-4)	x+8	rw	**Enable Channel Trigger Counter CTVx.CTC** This bit enables CTVx.CTC. The bits ETC0-ETCx can be set with one write access to synchronously start all counters. This is required for proper sync pulse staggering. If set, CTVx.CTC counts on, starting from its current value. CTVx.CTC can be written only if ETCx is cleared (stopped). In the device only ETC0-3 are available. All other must be written with 0, and always read 0.
CENx (x=0-4)	x+16	rw	**Enable Channel** This bit enables PSI5 Channel x. If cleared, all internal state machines of the receiver and the sender are forced to default idle state while all registers can be read and written. Used for configuration of a channel. In the device only CEN0…3 are available. All other must be written with 0, and always read 0.
0	7:5, 15:13, 31:21	r	**Reserved** Read as 0; should be written with 0.

26.4.25 Input and Output Control Register x

The Input and Output Control Register IOCRx determines for the PSI5 channel x:
for the receiver:

- the alternate input
- the filter depth
- the input signal polarity

for the transmitter

- the output signal polarity

IOCRx (x=0-1) Offset address: 084_H+x*90_H

Input and Output Control Register x Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TXM	RXM	0													
rh	rh	r													

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CFEG	CREG	FEG	REG	0		IIE	OIE	DEPTH				0		ALTI	
w	w	rh	rh	r		rw	rw	rw				r		rw	

Field	Bits	Type	Description
ALTI	1:0	rw	**Alternate Input Select** Selects the alternate input for channel x: 00_B Alternate Input 0 selected 01_B Alternate Input 1 selected 10_B Alternate Input 2 selected 11_B Alternate Input 3 selected
DEPTH	7:4	rw	**Digital Glitch Filter Depth** DEPTH determines the number of port input samples clocked with *f*PSI5 that are taken into account for the calculation of the floating average. The higher DEPTH is chosen to be, the longer the glitches that are suppressed and the longer the delay of the input signal introduced by this filter. 0_H Off 1_H 1 T_{PSI5} 2_H 2 … F_H 15
OIE	8	rw	**Output Inverter Enable Channel x** Selects the pulse polarity of the output of channel x 0_B Pulse polarity is not inverted 1_B Pulse polarity is inverted
IIE	9	rw	**Input Inverter Enable Channel x** Selects the pulse polarity of the input of channel x 0_B Pulse polarity is not inverted 1_B Pulse polarity is inverted
REG	12	rh	**Rising Edge Glitch Flag for Channel x** Shows the status of the glitch detection of channel x REG is cleared by setting CREG. 0_B No glitch detected on rising edge 1_B Glitch detected on rising edge
FEG	13	rh	**Falling Edge Glitch Flag for Channel x** Shows the status of the glitch detection of channel x FEG is cleared by setting CFEG. 0_B No glitch detected on falling edge 1_B Glitch detected on falling edge
CREG	14	w	**Clear Rising Edge Glitch Flag for Channel x** Clears the status flag REG CREG always read zero. 0_B REG is not cleared 1_B REG is cleared

(table continues...)

(continued)

Field	Bits	Type	Description
CFEG	15	w	**Clear Falling Edge Glitch Flag for Channel x** Clears the status flag FEG CFEG always read zero. 0_B FEG is not cleared 1_B FEG is cleared
RXM	30	rh	**Receive Monitor for Channel x** Shows the status of the receive signal of channel x after glitch filtering and inverted as specified by IIE. 0_B Current signal is low. 1_B Current signal is high.
TXM	31	rh	**Transmit Monitor for Channel x** Shows the status of the transmit signal of channel x inverted as specified by OIE. 0_B Current signal is low. 1_B Current signal is high.
0	3:2, 11:10, 29:16	r	**Reserved** Read as 0; should be written with 0.

26.4.26 Receiver Control Register A x

The Receiver Control Registers RCRA contains control bits/bit-fields that are related to the PSI5 receiver operation. It configures the payload length of the up to 6 frames in the up to 6 slots.

RCRAx (x=0-1) Offset address: 088_H+x*90_H

Receiver Control Register A x Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AVBS	ASYN	PDL5					PDL4					PDL3			
rw	rw	rw					rw					rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PDL3	PDL2					PDL1					PDL0				
rw	rw					rw					rw				

Field	Bits	Type	Description
PDLy (y=0-5)	5*y+4:5*y	rw	**Payload Data Length y** PDL determines the number of data bits per frame that the PSI5 channel x is setup for in slot y. PDL does not include the start bits and the Parity/CRC bits. PDL includes the Messaging bits, the Frame Control bits and the Status bits. 00_H 0 bit data 01_H 1 bit data 02_H 2 bit data 03_H 3 bit data … $1C_H$ 28 bit data $1D_H$ 0 bit data for 29 … $1F_H$ 0 bit data for 31
ASYN	30	rw	**Asynchronous mode** If set, • The watchdog timers for the PSI5 channel x, slot WDL1 - 6 are disabled • If WDL0 is not cleared TEI is issued if the distance between two frames is longer than specified in WDL0 • Slot Counter SC is incremented with each received frame (works as frame counter) with roll over after 6. • No sync pulses generated, TX path is inactive 0_B Off 1_B On
AVBS	31	rw	**Verbose mode for Asynchronous mode** If set, and ASYN is set and WDL0 is > 0, an empty frame is stored in RDRL/Hx and the referring RDML/H each time, the watchdog timer expires for the PSI5 channel x without reception of a frame. 0_B Off 1_B On

26.4.27 Receiver Control Register B x

The Receiver Control Registers RCRBx contains control bits/bit-fields that are related to the PSI5 receiver operation. It configures up to 6 frame types in the up to 6 slots.

RCRBx (x=0-1) Offset address: $08C_H+x*90_H$

Receiver Control Register B x Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								VBS5	FEC5	CRC5	MSG 5	VBS4	FEC4	CRC4	MSG 4
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VBS3	FEC3	CRC3	MSG 3	VBS2	FEC2	CRC2	MSG 2	VBS1	FEC1	CRC1	MSG 1	VBS0	FEC0	CRC0	MSG 0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

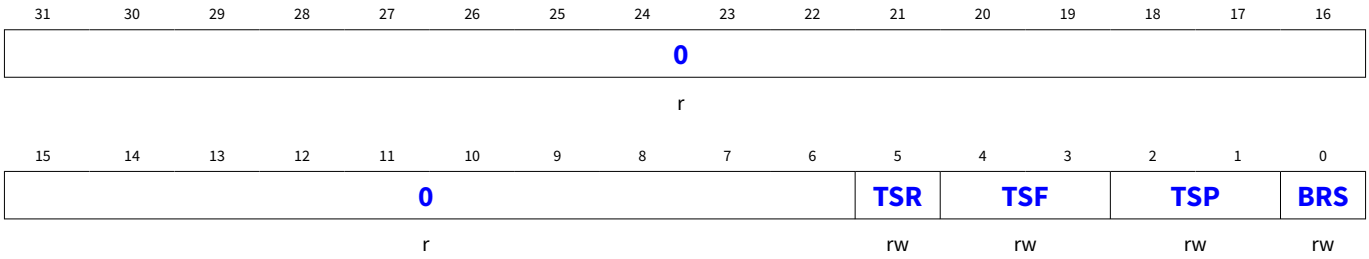
Field	Bits	Type	Description
MSGy (y=0-5)	4*y	rw	**Messaging Bits** If set, the 2 Messaging bits are configured for the PSI5 channel x in slot y. If set, Enhanced Serial Message processing on bits D[1:0] is activated. (18 frames, 4 or 8-bit ID, 12 or 16-bit data, 6-bit CRC) If Messaging bits are transmitted, they are always presented in RDRx independently from status of MSGy. That is, RDRx always stores exactly the data that was received. 0_B No Messaging bits 1_B 2 Messaging bits
CRCy (y=0-5)	4*y+1	rw	**CRC or Parity Selection** If set, a 3-bit CRC checksum is expected for the PSI5 channel x in slot y. Else, 1-bit Parity is assumed. 0_B 1 Parity Bit is configured 1_B 3 CRC bits are configured
FECy (y=0-5)	4*y+2	rw	**Frame Expectation Control** If set, a frame is expected for the PSI5 channel x in slot y. A No Frame Received error interrupt NFI is issued each time, the watchdog timer for the PSI5 channel x, slot y expires without reception of a SOF or EOF. 0_B No frame expected 1_B A frame is expected, SC is checked
VBSy (y=0-5)	4*y+3	rw	**Verbose mode** If set, and a message is expected for the PSI5 channel x in slot y (FECy) an empty frame is stored in RDRL/Hx and the referring RDML/H each time, the watchdog timer for the PSI5 channel x, slot y expires without reception of a frame. 0_B Off 1_B On
0	31:24	r	**Reserved** Read as 0; should be written with 0.

26.4.28 Receiver Control Register C x

The Receiver Control Registers RCRCx contains control bits/bit-fields that are related to the PSI5 receiver operation. Bit rate and timestamp sources are configured here.

RCRCx (x=0-1) Offset address: 090_H+x*90_H

Receiver Control Register C x Kernel Reset value: $0000\ 0000_H$



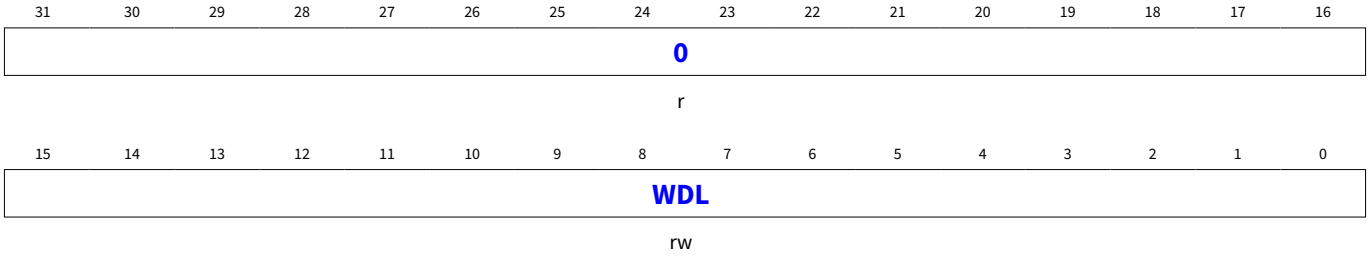
Field	Bits	Type	Description
BRS	0	rw	**Bit Rate Select** 0_B f_{125} selected for channel x 1_B f_{189} selected for channel x
TSP	2:1	rw	**Timestamp Select for Pulses** 00_B TSRA.CTS is captured in SPTSC 01_B TSRB.CTS is captured in SPTSC 10_B TSRC.CTS is captured in SPTSC 11_B not defined. Do not use
TSF	4:3	rw	**Timestamp Select for Start of Frame (SOF)** 00_B TSRA.CTS is captured in SFTSC 01_B TSRB.CTS is captured in SFTSC 10_B TSRC.CTS is captured in SFTSC 11_B not defined. Do not use
TSR	5	rw	**Timestamp Select for Receive Data Registers** 0_B SPTSC is stored in RDRHx 1_B SFTSC is stored in RDRHx
0	31:6	r	**Reserved** Read as 0; should be written with 0.

26.4.29 Watchdog Timer Register xw

For details, see chapter "Watchdog Timer".

WDTxw (w=0-6;x=0-1) Offset address: 094_H+x*90_H+w*4

Watchdog Timer Register xw Kernel Reset value: $0000\ 0000_H$



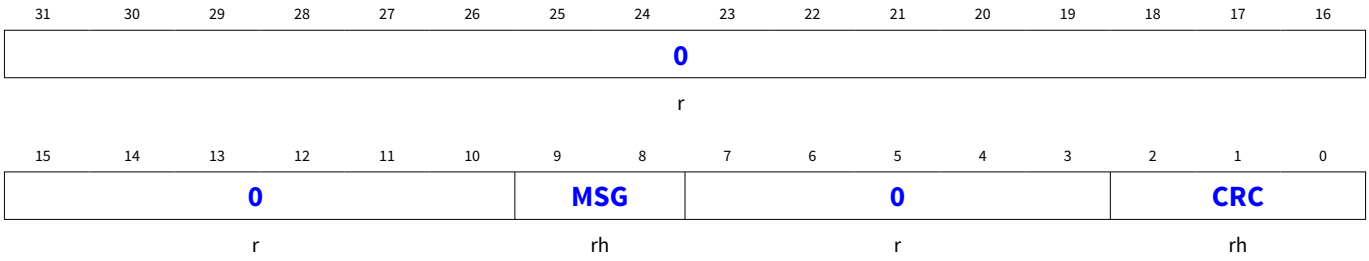
Field	Bits	Type	Description
WDL	15:0	rw	**Watchdog Timer Limit** for channel x limit w.
0	31:16	r	**Reserved** Read as 0; should be written with 0.

26.4.30 Receive Status Register x

The Receive Status Register provides the status information of channel x.

RSRx (x=0-1) Offset address: $0B0_H+x*90_H$

Receive Status Register x Kernel Reset value: $0000\ 0000_H$



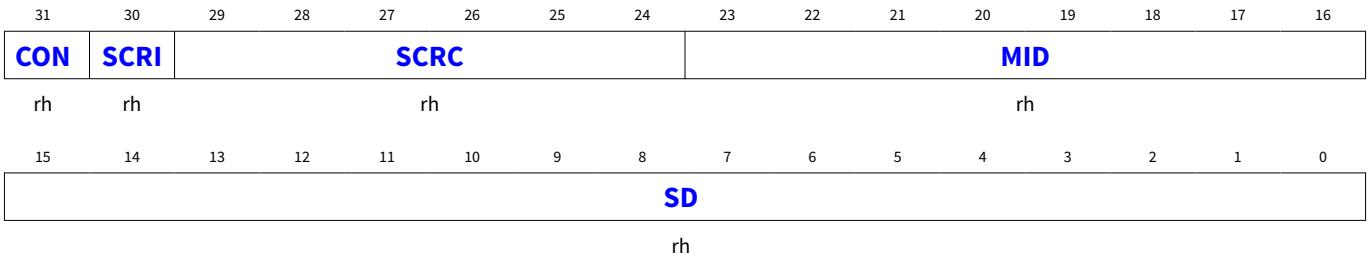
Field	Bits	Type	Description
CRC	2:0	rh	**CRC** of last frame. CRC0 is on bit position 0.
MSG	9:8	rh	**Messaging Bits** of last frame. MSG0 is on bit position 8.
0	7:3, 31:10	r	**Reserved** Read as 0; should be written with 0.

26.4.31 Serial Data and Status Register xz

The Serial (Receive) Data and Status Register provides the data and status information of channel x slot z. The data is presented in bit-field SD independent from the SCRC check result in SCRI.

SDSxz (x=0-1;z=0-5) Offset address: $0B4_H+x*90_H+z*4$

Serial Data and Status Register xz Kernel Reset value: $0000\ 0000_H$



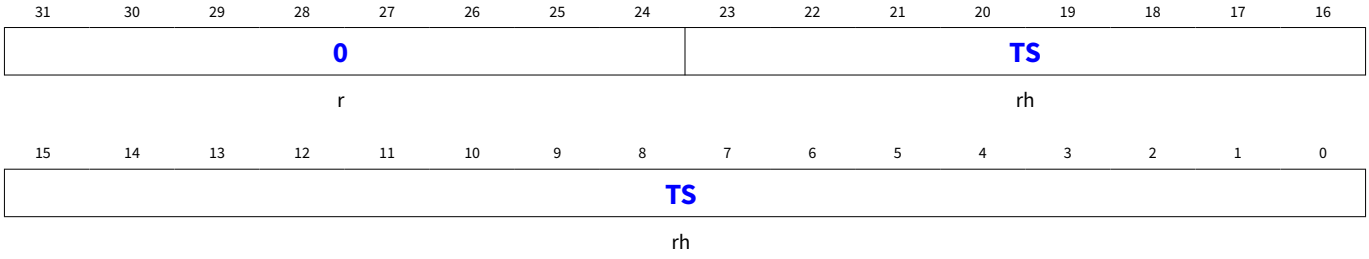
Field	Bits	Type	Description
SD	15:0	rh	**Serial Data** of last serial data frame on channel x slot z. SD0 is on bit position 0. If SDSxz.CON is not set, bits [15:12] are zero.
MID	23:16	rh	**Message ID** of last serial data frame. ID0 is on bit position 16. If SDSxz.CON is set, bits [23:20] are zero.
SCRC	29:24	rh	**SCRC** CRC of last serial data frame. SCRC0 is on position 24.
SCRI	30	rh	**CRC of Serial Message failed Interrupt Flag.** This bit is set if the CRC of the serial message fails. This includes a check of the Messaging bit-field for correct 0 values of bit 1 in frames 7, 13 and 18. See INTSTATAx.
CON	31	rh	**Configuration bit** of last serial frame. 0_B 12-bit data and 8-bit message ID 1_B 16-bit data and 4-bit message ID

26.4.32 Start of Pulse Timestamp Capture Register x

Each time a sync pulse is sent on channel x, the timestamp is captured in this register. RCRC.TSP selects if TSRA, B or C is used. RCRC.TSR determines if SPTSCx or SFTSCx is stored in RDRHx.

SPTSCx (x=0-1) Offset address: $0CC_H+x*90_H$

Start of Pulse Timestamp Capture Register x Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
TS	23:0	rh	**Timestamp** of the last sync pulse sent on channel x. The Timestamp is taken at the rising edge of the pulse.
0	31:24	r	**Reserved** Read as 0; should be written with 0.

26.4.33 Start of Frame Timestamp Capture Register x

Each time a start of frame is received on channel x, the timestamp is captured in this register. The first rising edge of the first start bit is captured. Note that a preceding “Idle Time” is required for start bit detection. In case the two start bits should not be received successfully (that is, without Manchester coding errors) the frame is completely dropped and the latest captured value is overwritten with the next SOF or at the referring WDTxw.WDL for “empty messages” if configured in next SOF or at the referring WDTxw.WDL for “empty messages” if configured in RCRBx.VBSy. RCRC.TSF selects if TSRA, B or C is used. RCRCx.TSR determines if SPTSCx or SFTSCx is stored in RDRHx.

SFTSCx (x=0-1) Offset address: $0D0_H+x*90_H$

Start of Frame Timestamp Capture Register x Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								TS							
r								rh							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TS															
rh															

Field	Bits	Type	Description
TS	23:0	rh	**Timestamp** of the last frame received on channel x. The timestamp is taken at the rising edge of the first start bit S1 that was qualified by the receiver.
0	31:24	r	**Reserved** Read as 0; should be written with 0.

26.4.34 Receive Data Register Low x

The Receive Data Register RDRLx for channel x shows the data content of a received data frame.

The internal receive buffer is always cleared ($0x0000\ 0000_H$) at each frame start. Thus unused nibbles are always read as zero.

The bit CRCI reflect, wether the conditions are met to issue an interrupt signal and set the referring interrupt status bit in INTSTATA for the latest frame. Thus it is independent from the old status of the referring sticky bit in INTSTATA before latest frame reception.

RDRLx (x=0-1) Offset address: $0D4_H+x*90_H$

Receive Data Register Low x Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RD															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RD												CRC			CRCI
rh												rh			rh

Field	Bits	Type	Description
CRCI	0	rh	**CRC Error Flag** This bit is set if the CRC or parity check fails. Set as well if CRC can not be calculated: PDL = 0, CRC cut off by Manchester Error, too few bits, BOT, Sync Pulse) 0_B correct CRC/parity 1_B wrong CRC/parity
CRC	3:1	rh	**CRC** CRC/parity bit of last PSI5 frame. CRC0 or parity bit is on bit position 1. In case of NFI, this field is cleared.
RD	31:4	rh	**RD** Receive data of last PSI5 frame. RD0 is on bit position 4. In case of NFI, this field is cleared.

26.4.35 Receive Data Register High x

The Receive Data Register RDRHx for channel x shows the data content of a received data frame.

The internal receive buffer is always cleared (0x0000 0000_H) at each frame start. Thus unused bits are always read as zero.

The bits RBI, RMI, RSI, NBI and TEI reflect wether the conditions are met to issue an interrupt signal and set the referring interrupt status bit in INTSTATA for the latest frame. Thus it is independent from the old status of the referring sticky bit in INTSTATA before latest frame reception.

RDRHx (x=0-1) Offset address: $0D8_H+x*90_H$

Receive Data Register High x Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RBI	NFI	MEI	NBI	TEI	SC			TS							
rh	rh	rh	rh	rh	rh			rh							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TS															
rh															

Field	Bits	Type	Description
TS	23:0	rh	**Timestamp** of the last received PSI5 frame. RCRC.TSR determines if SPTSCx or SFTSCx is stored in RDRHx. (It can be selected if the timestamp of the last sync pulse is stored or the timestamp for the start of frame).

(table continues...)

(continued)

Field	Bits	Type	Description
SC	26:24	rh	**Slot Counter** In Synchronous mode (RCRAx.ASYN=0): Number of the slot the SOF of the frame was received in. In Asynchronous mode (RCRAx.ASYN=1): revolving frame number between 1 and 6. 000_B not valid (SOF too early) 001_B Slot 1 … 110_B Slot 6 111_B not valid
TEI	27	rh	**Time Slot Error Flag** In Synchronous mode (RCRAx.ASYN = 0), this bit is set if the SOF and the EOF of a frame are received in different time slots. The slots are constraint by the watchdog timer (see WDTxw). In Asynchronous mode (RCRAx.ASYN = 1), this bit is set if the distance between two frames is longer than specified in WDL0. 0_B no error 1_B error
NBI	28	rh	**Number of bits Error Flag** This bit is set if the last frame received less bits than expected but no Manchester coding error occurred until S1, S2, M0 and M1 were received. Note that the frame after the error might be completely wrong. 0_B all bits received 1_B not all bits received
MEI	29	rh	**Error in Message Bits Flag** This bit is set if a Manchester error occurred in Bit M0 or M1. (Only if configured in RCRBx.MSG) Note that the frame after the error might be completely wrong. 0_B frame is correct 1_B frame is not correct
NFI	30	rh	**No Frame Received Flag** This bit is set when the NFI condition as described in the first paragraph of INTSTATA.NFI occurs. If only the two start bits are received and no further data, this “SOF” is discarded completely and treated like no frame received as well. If RCRBx.VBS is set, an empty frame is stored anyhow and RDIx is issued as well. The content of RDRL/Hx / RDML/Hx is valid for SC and for this bit (NFI). bit-fields RD and CRC are cleared (all zero). If RCRC.VBSx is 0 then TS is cleared. If RCRC.VBSx is 1 then TS is set to either SPTSCx.TS or SFTSCx.TS depending on the value of RCRC.TSR. 0_B no error 1_B error

(table continues...)

(continued)

Field	Bits	Type	Description
RBI	31	rh	**Receive Buffer Overflow Flag** This bit is set after a frame has been received while the old one was not read from RDRHx. That is, the kernel wants to set any of the two interrupts RSI and RDI and finds any of these two interrupts already set. The old data is overwritten by the new data. 0_B No overflow 1_B Overflow

26.4.36 Pulse Generation Control Register x

The Pulse Generation Control Register PGC contains control data for the sync pulse generation. It contains as well the trigger control bits required for sending data from the PSI5 module to the sensor / external PSI5 device.

PGCx (x=0-1) Offset address: $0DC_H+x*90_H$

Pulse Generation Control Register x Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BOT							BYP	ETE	ETS			PTE	ETB		
rw							rw	rw	rw			rw	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TBS	TRIG SEL	DEL						0		PLEN					
rw	rw	rw						r		rw					

Field	Bits	Type	Description
PLEN	5:0	rw	**Pulse Length** Defines the length of the pulse in TTS times (defined by FDRT). This is the standard pulse width without data coding into the pulse width or for coding a ‘0’. 00_H Pulse length is 0 TTS 01_H Pulse length is 1 TTS … $3F_H$ Pulse length is 63 TTS
DEL	13:8	rw	**Delay Length** In case data is coded into the pulse length, this defines the ADDITIONAL length of the pulse in TTS times. The resulting sum length is the pulse width for coding a ‘1’ into the pulse width. 00_H Pulse length is 0 TTS 01_H Pulse length is 1 TTS … $3F_H$ Pulse length is 63 TTS

(table continues...)

(continued)

Field	Bits	Type	Description
TRIGSEL	14	rw	**Trigger select** Trigger source selection bit 0_B **GTMTRG**: GTM GTM trigger 1_B **eGTMTRG**: eGTM eGTM trigger
TBS	15	rw	**Time Base Select** This bit selects the clock source for CTVx 0_B Internal, CTV counts in clock cycles of f_{TS} 1_B External, CTV counts in clock cycles of f_{GTM} , eGTM
ETB	18:16	rw	**External Time Base Select** Selects the external clock line for counter CTVx. 000_B TRIG0 001_B TRIG1 … 101_B TRIG5 110_B Reserved, no trigger selected 111_B Reserved, no trigger selected
PTE	19	rw	**Periodic Trigger Enable** Periodic trigger is defined by CTVx. Should be 0 if ETE or BYP is set. 0_B Disabled 1_B Enabled
ETS	22:20	rw	**External Trigger Select** Selects the external trigger line for pulse generation (e.g. angle synchronous). 000_B TRIG0 001_B TRIG1 … 101_B TRIG5 110_B Reserved, no trigger selected 111_B Reserved, no trigger selected
ETE	23	rw	**External Trigger Enable** “Angle sync. trigger”, external line is selected by ETS. Should be 0 if PTE or BYP is set. 0_B Disabled 1_B Enabled
BYP	24	rw	**Bypass Enable** An external signal, selected by ETS directly drives the output of channel x. Should be 0 if PTE or ETE is set. 0_B Disabled 1_B Enabled

(table continues...)

(continued)

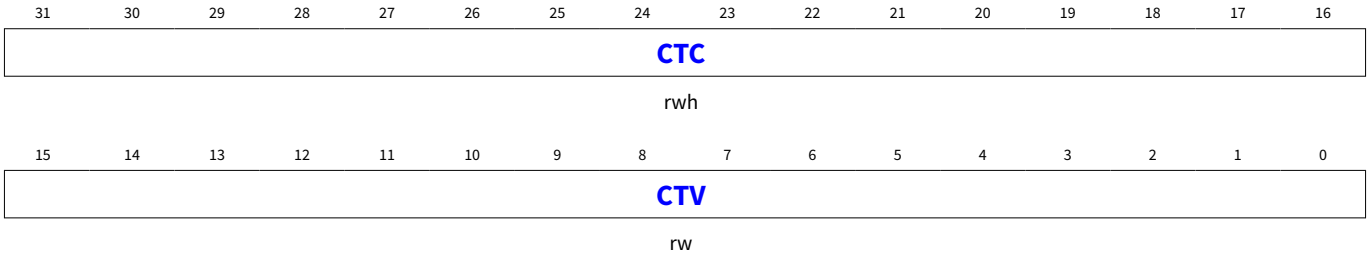
Field	Bits	Type	Description
BOT	31:25	rw	**Blank Out Time** BOT defines the length of the blank out time in TTS times (defined by FDRT) starting from the end of a Sync Pulse. The receiver is always switched off starting from the beginning of a Sync Pulse. BOT keeps the receiver switched off additionally after the Sync Pulse. This suppresses potential noise on the receive line after the pulse and thus eases the design of the transceiver. 00_H Blank Out Time is 0 TTS 01_H Blank Out Time is 1 TTS … $7F_H$ Blank Out Time is 127 TTS
0	7:6	r	**Reserved** Read as 0; should be written with 0.

26.4.37 Channel Trigger Value Register x

The Channel Trigger Value Register x contains a two cell reset counter. It allows for setting up periodic sync pulses for channel x.

CTVx (x=0-1) Offset address: $0E0_H+x*90_H$

Channel Trigger Value Register x Kernel Reset value: $0000\ 0000_H$



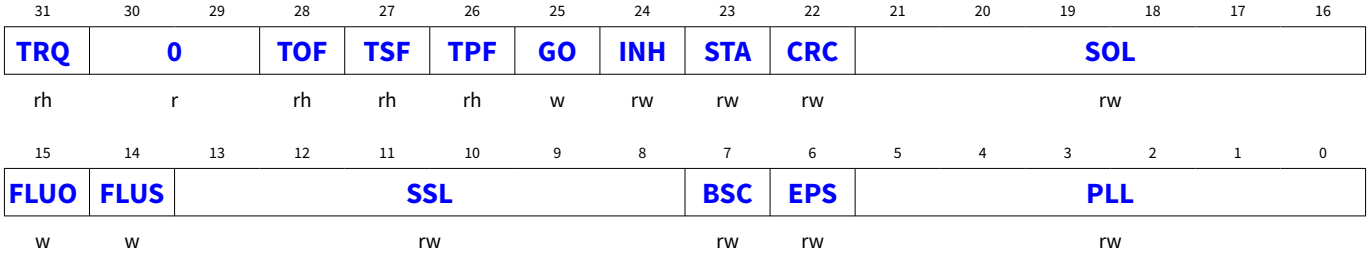
Field	Bits	Type	Description
CTV	15:0	rw	**Channel Trigger Value CTV** Contains the compare value (exact match) of Channel Trigger Counter (CTC) at which a sync pulse is triggered for channel x and the CTC value is cleared. If cleared, CTC is stopped and no pulse is generated.
CTC	31:16	rwh	**Channel Trigger Counter** This bit-field allows to read the current counter value of the reset timer cell CTVx. If GCR.ETCx is cleared, CTC can be written.

26.4.38 Send Control Register x

The Send Control Register SCR contains control data bits required for sending data from the PSI5 module to the sensor / external PSI5 device.

SCRx (x=0-1) Offset address: $0E4_H+x*90_H$

Send Control Register x Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
PLL	5:0	rw	**Payload Length of Registers SDRL/H** If PLL > 31, SDRH needs to be written to trigger the HW for automatic STA, BSC or CRC generation or just moving data to SSRL/H. Else writing to SDRL is sufficient. PLL needs to written before SDRL/H is used for proper operation. If insertion of STA, BSC and CRC results in more than 64 bits, the MSBs are truncated in SSR. Defines the length that is taken into account: 00_H length is 1 01_H length is 2 … $3F_H$ length is 64
EPS	6	rw	**Enhanced Protocol Selection** 0_B PSI5 V1.3. No Pulse length extension 1_B Enhanced Protocol with Pulse length extension.
BSC	7	rw	**Bit Stuffing Control** Depending from bit EPS after 3 bits a ‘1’ is inserted (V1.3) or after 6 bits a ‘0’ is inserted (enhanced Power Train mode) Note that this makes the frame longer. 0_B No automatic bit stuffing 1_B Automatic bit stuffing is enabled.
SSL	13:8	rw	**Payload Length of Registers SSRL/H** If SSL > 31, SSRH needs to be written to trigger the HW for automatic moving data to SORL/H. Else writing to SSRL is sufficient. SSL needs to be written before SSRL/H is used for proper operation. Defines the length that is taken into account. Start Sequence, BSC and CRC need to be added to PLL by SW if SSL is calculated based on PLL. 00_H length is 1 01_H length is 2 … $3F_H$ length is 64
FLUS	14	w	**Flush SSRH/Lx** Setting this bit stops the shifting process and flushes (clears) SSRH/Lx and TSF. INTSTATAx.TPI is issued at the end of successful flushing. Reads always as zero.

(table continues...)

(continued)

Field	Bits	Type	Description
FLUO	15	w	**Flush SORH/Lx** Setting this bit stops the sending process and flushes (clears) SORH/Lx and TOF. INTSTATAx.TSI is issued at the end of successful flushing. Reads always as zero.
SOL	21:16	rw	**Payload Length of Registers SORL/H** If SOL > 31, SORH needs to be written to trigger the HW for automatic moving data to the pulse generator. Else writing to SORL is sufficient. SOL needs to written before SORL/H is used for proper operation. Defines the length that is taken into account. Start Sequence, CRC and Stuffing needs to be added by SW to PLL if SOL is calculated based on PLL. 00_H length is 1 01_H length is 2 … $3F_H$ length is 64
CRC	22	rw	**CRC Generation Control** 0_B CRC is not generated automatically. 1_B CRC is automatically generated and added to the frame by HW (according to EPS). Note that this makes the frame longer. CRC calculation is not performed, if at least 1 CRC bit does not fit into 64 bit, e.g. due to STA or BSC. In this case, one or two zero bits are inserted.
STA	23	rw	**Start Sequence Generation Control** 0_B no start sequence generated, payload starts at bit 0 1_B automatically generated and added to the frame by HW (according to EPS) payload starts at bit 3/9 (EPS = 0/1). Note that this makes the frame longer.
INH	24	rw	**Inhibit Transfer** This inhibits the automatic transfer from the shift registers SSRL/H to SORL/H after the preparation of all automatic bit-fields (STA, BSC, CRC). After a write access to SDRL/Hx, the automatic preparation is done by HW if selected. If automatic preparation of the start sequence is not selected, data is moved by HW without change to SSRL/H. When the HW is done with the transfer to the internal shift registers, interrupt TPI is activated, flag TSF is set and flag TPF is cleared automatically. Depending on INH, GO must be set after TPI before transfer to SORL/H takes place: After the transfer to SORL/H, TSI is activated, TOF is set and and TSF is cleared automatically. . INH defines the start mode: 0_B automatic transfer 1_B GO bit must be set before transfer
GO	25	w	**Release prepared Send data** This is only relevant if INH is set It allows for manual control of the transfer from SSRH/L to SORL/H. It is always read as zero and is automatically cleared. 0_B no transfer 1_B start transfer

(table continues...)

(continued)

Field	Bits	Type	Description
TPF	26	rh	**Transmit Preparation Flag** If set, preparation is in progress: start sequence and/ or CRC and/or stuffing and/or at least copying data from SDRL/H. If set, write access to SDRL/H will not change any data and issue TPOI. It is cleared automatically after preparation is finished.
TSF	27	rh	**Transmit Shift Flag** If set, data in SSRL/H is waiting to be shifted to SORL/H. This can be caused by an automatic transfer from SDRL/H or from a CPU write access to SSRL/H. If SSL < 32 it is sufficient to write SSRL. Else, only writing SSRH will set TSF. If set, write access to SSRL/H will not change any data and issue TSOI. It is cleared automatically after data was shifted to SORL/H or after flushing the register by setting bit FLUS.
TOF	28	rh	**Transmit Output Flag** If set, data in SORL/H is waiting to be transmitted to the sensor. This can be caused by an automatic transfer from SSRL/H or from a CPU write access to SORL/H. If SOL < 32 it is sufficient to write SORL. Else, only writing SDRH will set TSF. If set, write access to SORL/H will not change any data and issue TOOI. It is cleared automatically after data was transmitted to the sensor or after flushing the register by setting bit FLUO.
TRQ	31	rh	**Transfer Request in Progress** While a transfer is being sent this bit is set. Write access is ignored.
0	30:29	r	**Reserved** Read as 0; should be written with 0.

26.4.39 Send Data Register Low x

The Send Data Register SDRLx for channel x shows the data content of a data frame to be sent.

Data to be sent must be written while the channel is enabled (GCR.CEN is set). While CEN is cleared the HW can not detect the write access as all state machines are stopped.

SDRLx (x=0-1) Offset address: $0E8_H+x*90_H$

Send Data Register Low x Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SD31	SD30	SD29	SD28	SD27	SD26	SD25	SD24	SD23	SD22	SD21	SD20	SD19	SD18	SD17	SD16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SD15	SD14	SD13	SD12	SD11	SD10	SD9	SD8	SD7	SD6	SD5	SD4	SD3	SD2	SD1	SD0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
SDy (y=0-31)	y	rw	**SDy** Send data of next ECU to Sensor frame. The unused MSBs (bit position SCRx.PLL and higher) must be written with '0'. If PLL is < 32, SDRHx must not be written.

26.4.40 Send Data Register High x

The Send Data Register SDRHx for channel x shows the data content of a data frame to be sent.

SDRHx (x=0-1) Offset address: $0EC_H+x*90_H$

Send Data Register High x Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SD63	SD62	SD61	SD60	SD59	SD58	SD57	SD56	SD55	SD54	SD53	SD52	SD51	SD50	SD49	SD48
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SD47	SD46	SD45	SD44	SD43	SD42	SD41	SD40	SD39	SD38	SD37	SD36	SD35	SD34	SD33	SD32
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
SDy (y=32-63)	y-32	rw	**SDy** Send data of next ECU to Sensor frame. The unused MSBs (bit position SCRx.PLL and higher) must be written with '0'. If PLL is < 32, SDRHx must not be written.

26.4.41 Send Shift Register Low x

The Send Shift Register SSRLx for channel x shows the data that will be shifted to the Send Output Register SORL. It contains as well the stuff bits as specified in PSI5 V1.3 or in PSI5 V2.1 frame formats 1-3. During transmission, all bits are shifted to LSB with each sync pulse. MSB is filled up with zero.

SSRLx (x=0-1) Offset address: $0F0_H+x*90_H$

Send Shift Register Low x Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SD31	SD30	SD29	SD28	SD27	SD26	SD25	SD24	SD23	SD22	SD21	SD20	SD19	SD18	SD17	SD16
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SD15	SD14	SD13	SD12	SD11	SD10	SD9	SD8	SD7	SD6	SD5	SD4	SD3	SD2	SD1	SD0
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
SDy (y=0-31)	y	rwh	**SDy** Send data of next ECU to Sensor frame. The sequence is: Start sequence (STS), Address data (AD), Send data (SD), Cyclic redundancy check (CRC). Depending on individual field length, the bit stream is continued in SSRHx.

26.4.42 Send Shift Register High x

Continues the bit stream as described in SSRLx.

SSRHx (x=0-1) Offset address: $0F4_H+x*90_H$

Send Shift Register High x Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SD63	SD62	SD61	SD60	SD59	SD58	SD57	SD56	SD55	SD54	SD53	SD52	SD51	SD50	SD49	SD48
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SD47	SD46	SD45	SD44	SD43	SD42	SD41	SD40	SD39	SD38	SD37	SD36	SD35	SD34	SD33	SD32
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
SDy (y=32-63)	y-32	rwh	**SDy** Continues the bit stream as described in SSRLx.

26.4.43 Send Output Register Low x

The Send Output Register SORLx for channel x shows the data that will be shifted to the pulse generator. During transmission, all bits are shifted to LSB with each sync pulse. MSB is filled up from SORH bit 0, while the MSB of SORH is filled up with zeros.

SORLx (x=0-1) Offset address: $0F8_H+x*90_H$

Send Output Register Low x Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SD31	SD30	SD29	SD28	SD27	SD26	SD25	SD24	SD23	SD22	SD21	SD20	SD19	SD18	SD17	SD16
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SD15	SD14	SD13	SD12	SD11	SD10	SD9	SD8	SD7	SD6	SD5	SD4	SD3	SD2	SD1	SD0
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
SDy (y=0-31)	y	rwh	**SDy** Send data of next ECU to Sensor frame. The sequence is: Start sequence (STS), Address data (AD), Send data (SD), Cyclic redundancy check (CRC). Depending on individual field length, the bit stream is continued in SORHx.

26.4.44 Send Output Register High x

Continues the bit stream as described in SORLx.

SORHx (x=0-1) Offset address: $0FC_H+x*90_H$

Send Output Register High x Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SD63	SD62	SD61	SD60	SD59	SD58	SD57	SD56	SD55	SD54	SD53	SD52	SD51	SD50	SD49	SD48
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SD47	SD46	SD45	SD44	SD43	SD42	SD41	SD40	SD39	SD38	SD37	SD36	SD35	SD34	SD33	SD32
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
SDy (y=32-63)	y-32	rwh	**SDy** Continues the bit stream as described in SORLx.

26.4.45 Interrupt Overview Register

INTOV Offset address: $2C4_H$

Interrupt Overview Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								FWI	SDI	ERRI	TBI	TDI	RBI	RDI	RSI
r								rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
RSI	0	rh	**Interrupt Pending on any Node Pointer RSI** If any interrupt requested flag is set for any Node Pointer RSI in register INTSTATAx and the referring interrupt is enabled in corresponding INTEN register (INTENA or INTENB) then INTOV.RSI is set. It is automatically reset if all flags in INTSTATAx are cleared for which the referring interrupt is enabled in corresponding INTEN register (INTENA or INTENB).
RDI	1	rh	**Interrupt Pending on any Node Pointer RDI** If any interrupt requested flag is set for any Node Pointer RDI in register INTSTATAx and the referring interrupt is enabled in corresponding INTEN register (INTENA or INTENB) then INTOV.RDI is set. It is automatically reset if all flags in INTSTATAx are cleared for which the referring interrupt is enabled in corresponding INTEN register (INTENA or INTENB).
RBI	2	rh	**Interrupt Pending on any Node Pointer RBI** If any interrupt requested flag is set for any Node Pointer RBI in register INTSTATAx and the referring interrupt is enabled in corresponding INTEN register (INTENA or INTENB) then INTOV.RBI is set. It is automatically reset if all flags in INTSTATAx are cleared for which the referring interrupt is enabled in corresponding INTEN register (INTENA or INTENB).
TDI	3	rh	**Interrupt Pending on any Node Pointer TDI** If any interrupt requested flag is set for any Node Pointer TDI in register INTSTATAx and the referring interrupt is enabled in corresponding INTEN register (INTENA or INTENB) then INTOV.TDI is set. It is automatically reset if all flags in INTSTATAx are cleared for which the referring interrupt is enabled in corresponding INTEN register (INTENA or INTENB).
TBI	4	rh	**Interrupt Pending on any Node Pointer TBI** If any interrupt requested flag is set for any Node Pointer TBI in register INTSTATAx and the referring interrupt is enabled in corresponding INTEN register (INTENA or INTENB) then INTOV.TBI is set. It is automatically reset if all flags in INTSTATAx are cleared for which the referring interrupt is enabled in corresponding INTEN register (INTENA or INTENB).
ERRI	5	rh	**Interrupt Pending on any Node Pointer ERRI** If any interrupt requested flag is set for any Node Pointer ERRI in register INTSTATAx or INTSTATBx and the referring interrupt is enabled in corresponding INTEN register (INTENA or INTENB) then INTOV.ERRI is set. It is automatically reset if all flags in INTSTATAx and INTSTATBx are cleared for which the referring interrupt is enabled in corresponding INTEN register (INTENA or INTENB).
SDI	6	rh	**Interrupt Pending on any Node Pointer SDI** If any interrupt requested flag is set for any Node Pointer SDI in register INTSTATBx and the referring interrupt is enabled in corresponding INTEN register (INTENA or INTENB) then INTOV.SDI is set. It is automatically reset if all flags in INTSTATBx are cleared for which the referring interrupt is enabled in corresponding INTEN register (INTENA or INTENB).

(table continues...)

(continued)

Field	Bits	Type	Description
FWI	7	rh	**Interrupt Pending on any Node Pointer FWI** If any interrupt requested flag is set for any Node Pointer FWI in register INTSTATAx and the referring interrupt is enabled in corresponding INTEN register (INTENA or INTENB) then INTOV.FWI is set. It is automatically reset if all flags in INTSTATAx are cleared for which the referring interrupt is enabled in corresponding INTEN register (INTENA or INTENB).
0	31:8	r	**Reserved** Read as 0.

26.4.46 Interrupt Node Pointer Register x

The Interrupt Node Pointer Register INPx contains the node pointers of PSI5 channel x.

Node Pointer ERRI is one single node pointer for the following interrupts:

- TEI
- NFI
- NBI
- MEI
- CRCI
- RUI
- RMI
- WSIy
- SOIy
- SCRIy

Node Pointer TBI is one single node pointer for the following interrupts:

- TPOI
- TSOI
- TOOI

Node Pointer TDI is one single node pointer for the following interrupts:

- TPI
- TSI
- TOI

INPx (x=0-1) Offset address: $2C8_H+x*4$

Interrupt Node Pointer Register x Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FWI				SDI				ERRI				TBI			
rw				rw				rw				rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TDI				RBI				RDI				RSI			
rw				rw				rw				rw			

Field	Bits	Type	Description
RSI	3:0	rw	**Interrupt Node Pointer for Interrupt RSI** This bit-field defines the interrupt node, that is requested due to the set condition for bit INTSTATAx.RSI (if enabled by bit INTENAx.RSI). 0_H Interrupt trigger PSI5SRx 0 is selected 1_H Interrupt trigger PSI5SRx 1 is selected … 7_H Interrupt trigger PSI5SRx 7 is selected 8_H reserved … F_H reserved
RDI	7:4	rw	**Interrupt Node Pointer for Interrupt RDI** This bit-field defines the interrupt node, that is requested due to the set condition for bit INTSTATAx.RDI (if enabled by bit INTENAx.RDI). For bit-field definition, see RSI.
RBI	11:8	rw	**Interrupt Node Pointer for Interrupt RBI** This bit-field defines the interrupt node, that is requested due to the set condition for bit INTSTATAx.RBI (if enabled by bit INTENAx.RBI). For bit-field definition, see RSI.
TDI	15:12	rw	**Interrupt Node Pointer for Interrupt TDI** This bit-field defines the interrupt node, that is requested due to the set condition for bit INTSTATAx.TPI, TSI, TOI (if enabled by bit INTENAx.TPI, TSI, TOI). For bit-field definition, see RSI.
TBI	19:16	rw	**Interrupt Node Pointer for Interrupt TBI** This bit-field defines the interrupt node, that is requested due to the set condition for bit INTSTATAx.TPOI, TSOI, TOOI (if enabled by bit INTENAx.TPOI, TSOI, TOOI). For bit-field definition, see RSI.
ERRI	23:20	rw	**Interrupt Node Pointer for Interrupt ERRI** This bit-field defines the interrupt node, that is requested due to the set condition for bit TEI, NFI, NBI, MEI, CRCI, RUI, RMI, CRCIy, WSIy, SOIy, SCRIy. For bit-field definition, see RSI.
SDI	27:24	rw	**Interrupt Node Pointer for Interrupt SDI** This bit-field defines the interrupt node, that is requested due to the set condition for bit INTSTATBx.SDI (if enabled by bit corresponding INTEN register (INTENA or INTENB).SDI). For bit-field definition, see RSI.
FWI	31:28	rw	**Interrupt Node Pointer for FWI** This bit-field defines the interrupt node, that is requested due to the set condition for bit INTSTATAx.FWI. For bit-field definition, see RSI.

26.4.47 Interrupt Status Register A x

The Interrupt Status Registers INTSTATA/Bx contains status bits that show the status of any interrupt of PSI5 channel x.

The bits are set independently from the referring Interrupt Enable in Register corresponding INTEN register (INTENA or INTENB). Thus they can be used as status bits as well e.g. by a SW based on polling.

INTSTATAx (x=0-1) Offset address: $2DC_H+x*4$

Interrupt Status Register A x Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															NFI
r															rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TOOI	TOI	TSOI	TSI	TPOI	TPI	RMI	RUI	FWI	CRCI	MEI	NBI	TEI	RBI	RDI	RSI
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
RSI	0	rh	**Receive Success Interrupt Request Flag** This bit is set at the successfully received end of a frame. It indicates that this frame is free of the errors NFI, TEI, NBI, MEI, CRCI if selected to be taken into account in register GCR. This bit can be cleared by bit INTCLRAx.RSI. This bit can be set by bit INTSETAx.RSI. This bit is set independently from INTENAx. 0_B No interrupt was requested since this bit was cleared the last time 1_B An interrupt was requested since this bit was cleared the last time
RDI	1	rh	**Receive Data Interrupt Request Flag** RDI is activated when a received frame is moved to a Receive Data Register RDRL/Hx. Both RDI and RSI will be issued together at correct reception. This bit can be cleared by bit INTCLRAx.RDI. This bit can be set by bit INTSETAx.RDI. This bit is set independently from INTENAx. 0_B No interrupt was requested since this bit was cleared the last time 1_B An interrupt was requested since this bit was cleared the last time

(table continues...)

(continued)

Field	Bits	Type	Description
RBI	2	rh	**Receive Buffer Overflow Interrupt Request Flag** This bit is set after a frame has been received while the old one was not read from RDRHx. That is, the kernel wants to set any of the two interrupts RSI and RDI and finds any of these two interrupts already set. The old data is overwritten by the new data. Thus, RBI can be ignored if the receive memory is used. This bit is NOT cleared by reading RDRx. This bit can be cleared by bit INTCLRAx.RBI. This bit can be set by bit INTSETAx.RBI. This bit is set independently from INTENAx. 0_B No interrupt was requested since this bit was cleared the last time 1_B An interrupt was requested since this bit was cleared the last time
TEI	3	rh	**Time Slot Error Interrupt Request Flag** In Synchronous mode (RCRAx.ASYN = 0), this bit is set if the SOF and the EOF of a frame are received in different time slots. The slots are constrained by the watchdog timer (see WDTxw). In Asynchronous mode (RCRAx.ASYN = 1), this bit is set if the distance between two frames is longer than specified in WDL0. This bit can be cleared by bit INTCLRAx.TEI. This bit can be set by bit INTSETAx.TEI. This bit is set independently from INTENAx. 0_B No interrupt was requested since this bit was cleared the last time 1_B An interrupt was requested since this bit was cleared the last time
NBI	4	rh	**Number of Bits Wrong Request Flag** This bit is set if the last frame received less bits than expected but no Manchester coding error occurred until S1, S2, M0 and M1 where received. Note that the frame after the error might be completely wrong. This bit can be cleared by bit INTCLRAx.NBI. This bit can be set by bit INTSETAx.NBI. This bit is set independently from INTENAx. 0_B No interrupt was requested since this bit was cleared the last time 1_B An interrupt was requested since this bit was cleared the last time
MEI	5	rh	**Error in Message Bits Flag** This bit is set if a Manchester error occurred in bit M0 or M1 (Only if configured in RCRBx.MSG). Note that the frame after the error might be completely wrong. This bit can be cleared by bit INTCLRAx.MEI. This bit can be set by bit INTSETAx.MEI. This bit is set independently from INTENAx. 0_B No interrupt was requested since this bit was cleared the last time 1_B An interrupt was requested since this bit was cleared the last time

(table continues...)

(continued)

Field	Bits	Type	Description
CRCI	6	rh	**CRC Error Request Flag** This bit is set if the CRC fails. Set as well if CRC can not be calculated: PDL = 0, CRC cut off by Manchester Error, too few bits, Blank out time, Sync Pulse) This bit can be cleared by bit INTCLRAx.CRCI. This bit can be set by bit INTSETAx.CRCI. This bit is set independently from INTENAx. 0_B No interrupt was requested since this bit was cleared the last time 1_B An interrupt was requested since this bit was cleared the last time
FWI	7	rh	**FIFO Warning Level Request Flag** This bit is set after if the configured warning level of the FIFOx was reached. (See RFCx) This bit can be cleared by bit INTCLRAx.FWI. This bit can be set by bit INTSETAx.FWI. This bit is set independently from INTENAx. 0_B No interrupt was requested since this bit was cleared the last time 1_B An interrupt was requested since this bit was cleared the last time
RUI	8	rh	**Receive Memory Underrun Interrupt Request Flag** This bit is set after a FPI master reads from FIFO while no new data was available. (See RFCx) This bit can be cleared by bit INTCLRAx.RUI. This bit can be set by bit INTSETAx.RUI. This bit is set independently from INTENAx. 0_B No interrupt was requested since this bit was cleared the last time 1_B An interrupt was requested since this bit was cleared the last time
RMI	9	rh	**Receive Memory Overflow Flag** This bit is set after a frame has been received while RDIOVx.RDI[RFCx.WRP] is set. That is, the kernel wants to set flag RFCx.WRP and finds this flag already set. The old data is overwritten by the new data in the buffer memory. This bit can be cleared by bit INTCLRAx.RMIx. This bit can be set by bit INTSETAx.RMIx. This bit is set independently from INTENAx. 0_B No overflow 1_B Overflow

(table continues...)

(continued)

Field	Bits	Type	Description
TPI	10	rh	**Transfer Preparation Interrupt Request Flag** This bit is set after data to be transferred has been moved from SDRL/H to SSRL/H. Thus a new value can be written to SDRL/Hx and the prepared data can be checked and modified by SW. This bit is automatically cleared by writing SDRx. This is an exception. It allows for DMA transfers without CPU activity. This bit can be cleared by bit INTCLRAx.TDI. This bit can be set by bit INTSETAx.TDI. This bit is set independently from INTENAx. 0_B No interrupt was requested since this bit was cleared the last time 1_B An interrupt was requested since this bit was cleared the last time
TPOI	11	rh	**Transmit Preparation Overflow Interrupt Request Flag** This bit is set after if SDRL/H is written while TPF is set. The old data is NOT overwritten. This bit can be cleared by bit INTCLRAx.TPOI. This bit can be set by bit INTSETAx.TPOI. This bit is set independently from INTENAx. 0_B No interrupt was requested since this bit was cleared the last time 1_B An interrupt was requested since this bit was cleared the last time
TSI	12	rh	**Transfer Shift Interrupt Request Flag** This bit is set after data to be transferred has been moved from SSRL/H to SORL/H. Thus a new value can be written to SSRL/Hx. This bit is NOT cleared by writing SSRL/Hx. This bit can be cleared by bit INTCLRAx.TSI. This bit can be set by bit INTSETAx.TSI. This bit is set independently from INTENAx. 0_B No interrupt was requested since this bit was cleared the last time 1_B An interrupt was requested since this bit was cleared the last time
TSOI	13	rh	**Transmit Shift Overflow Interrupt Request Flag** This bit is set after if SSRL/H is written while TSF is set. The old data is NOT overwritten. This bit can be cleared by bit INTCLRAx.TSOI. This bit can be set by bit INTSETAx.TSOI. This bit is set independently from INTENAx. 0_B No interrupt was requested since this bit was cleared the last time 1_B An interrupt was requested since this bit was cleared the last time

(table continues...)

(continued)

Field	Bits	Type	Description
TOI	14	rh	**Transfer Output Interrupt Request Flag** This bit is set after last bit to be transferred has been moved from SORL/H to the pulse generator. Thus a new value can be written to SORL/Hx. This bit is NOT cleared by writing SORL/Hx. This bit can be cleared by bit INTCLRAx.TOI. This bit can be set by bit INTSETAx.TOI. This bit is set independently from INTENAx. 0_B No interrupt was requested since this bit was cleared the last time 1_B An interrupt was requested since this bit was cleared the last time
TOOI	15	rh	**Transmit Shift Overflow Interrupt Request Flag** This bit is set after SORL/H is written while TOF is set. The old data is NOT overwritten. This bit can be cleared by bit INTCLRAx.TOOI. This bit can be set by bit INTSETAx.TOOI. This bit is set independently from INTENAx. 0_B No interrupt was requested since this bit was cleared the last time 1_B An interrupt was requested since this bit was cleared the last time
NFI	16	rh	**No Frame Received Interrupt Flag** This bit is set at the end of a slot if a frame is expected in the slot and SOF was not received in the slot. A special case is where an SOF is received in slot 0. In this scenario, the NFI bit is only set at the end of slot 1 if a frame is expected in slot 1 and an EOF was not received in slot 1. If only the two start bits are received and no further data, this SOF is discarded completely and treated like no frame was received as well. If the SOF only frame crosses the watchdog timer limit, NFI occurs after watchdog timer limit and EOF detection. If RCRBx.VBS is set, an empty frame is stored anyhow. In this case, RDIx is issued as well. The only valid information is SC, Timestamp (captured at the end of the slot) and this bit (NFI). This bit can be cleared by bit INTCLRAx.NFI. This bit can be set by bit INTSETAx.NFI. This bit is set independently from INTENAx. 0_B No interrupt was requested since this bit was cleared the last time 1_B An interrupt was requested since this bit was cleared the last time
0	31:17	r	**Reserved** Read as 0.

26.4.48 Interrupt Status Register B x

INTSTATBx (x=0-1) Offset address: $2EC_H+x*4$

Interrupt Status Register B x Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								SCRI5	SCRI4	SCRI3	SCRI2	SCRI1	SCRI0	SOI5	SOI4
r								rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SOI3	SOI2	SOI1	SOI0	SDI5	SDI4	SDI3	SDI2	SDI1	SDI0	WSI5	WSI4	WSI3	WSI2	WSI1	WSI0
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
WSIy (y=0-5)	y	rh	**Wrong Serial Protocol Error Request Flag** This bit is set if the Messaging bits are configured and do not show a start sequence for more than 18 frames of slot y. This bit can be cleared by bit INTCLRBx.WSIy. This bit can be set by bit INTSETBx.WSIy. This bit is set independently from INTENBx. 0_B No interrupt was requested since this bit was cleared the last time 1_B An interrupt was requested since this bit was cleared the last time
SDIy (y=0-5)	y+6	rh	**Serial Data Receive Interrupt Request Flag** This bit is set after all serial data bits have been received over the Messaging bit-field of slot y. This does NOT indicates a successful check of the CRC. This bit can be cleared by bit INTCLRBx.SDIy. This bit can be set by bit INTSETBx.SDIy. This bit is set independently from INTENBx. 0_B No interrupt was requested since this bit was cleared the last time 1_B An interrupt was requested since this bit was cleared the last time
SOIy (y=0-5)	y+12	rh	**Serial Data Buffer Overrun Interrupt Request Flag** This bit is set if the HW wants to set SDIy while SDIy is still set. This bit can be cleared by bit INTCLRBx.SOIy. This bit can be set by bit INTSETBx.SOIy. This bit is set independently from INTENBx. 0_B No interrupt was requested since this bit was cleared the last time 1_B An interrupt was requested since this bit was cleared the last time
SCRIy (y=0-5)	y+18	rh	**Serial Data CRC Error Request Flag** This bit is set if the CRC of the serial message fails. This includes a check of the Messaging bit-field for correct 0 values of bit 1 in frames 7, 13 and 18. This bit can be cleared by bit INTCLRBx.SCRIy. This bit can be set by bit INTSETx.SCRIy. This bit is set independently from INTENBx. 0_B No interrupt was requested since this bit was cleared the last time 1_B An interrupt was requested since this bit was cleared the last time
0	31:24	r	**Reserved** Read as 0.

26.4.49 Interrupt Set Register A x

The Interrupt Set Register INTSETA/Bx contain control bits that trigger an interrupt pulse for any interrupt of PSI5 channel x.

INTSETAx (x=0-1) Offset address: $2FC_H+x*4$

Interrupt Set Register A x Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															NFI
r															w

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TOOI	TOI	TSOI	TSI	TPOI	TPI	RMI	RUI	FWI	CRCI	MEI	NBI	TEI	RBI	RDI	RSI
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

Field	Bits	Type	Description
RSI	0	w	**Set Interrupt Request Flag RSI** Setting this bit sets bit INTSTATAx.RSI. Clearing this bit has no effect. Reading this bit returns always zero.
RDI	1	w	**Set Interrupt Request Flag RDI** Setting this bit sets bit INTSTATAx.RDI. Clearing this bit has no effect. Reading this bit returns always zero.
RBI	2	w	**Set Interrupt Request Flag RBI** Setting this bit sets bit INTSTATAx.RBI. Clearing this bit has no effect. Reading this bit returns always zero.
TEI	3	w	**Set Interrupt Request Flag TEI** Setting this bit sets bit INTSTATAx.TEI. Clearing this bit has no effect. Reading this bit returns always zero.
NBI	4	w	**Set Interrupt Request Flag NBI** Setting this bit sets bit INTSTATAx.NBI. Clearing this bit has no effect. Reading this bit returns always zero.
MEI	5	w	**Set Interrupt Request Flag MEI** Setting this bit sets bit INTSTATAx.MEI. Clearing this bit has no effect. Reading this bit returns always zero.
CRCI	6	w	**Set Interrupt Request Flag CRCI** Setting this bit sets bit INTSTATAx.CRCI. Clearing this bit has no effect. Reading this bit returns always zero.

(table continues...)

(continued)

Field	Bits	Type	Description
FWI	7	w	**Set Interrupt Request Flag FWI** Setting this bit sets bit INTSTATAx.FWI. Clearing this bit has no effect. Reading this bit returns always zero.
RUI	8	w	**Set Interrupt Request Flag RUI** Setting this bit sets bit INTSTATAx.RUI. Clearing this bit has no effect. Reading this bit returns always zero.
RMI	9	w	**Set Interrupt Request Flag RMI** Setting this bit sets bit INTSTATAx.RMI. Clearing this bit has no effect. Reading this bit returns always zero.
TPI	10	w	**Set Interrupt Request Flag TPI** Setting this bit sets bit INTSTATAx.TPI. Clearing this bit has no effect. Reading this bit returns always zero.
TPOI	11	w	**Set Interrupt Request Flag TPOI** Setting this bit sets bit INTSTATAx.TPOI. Clearing this bit has no effect. Reading this bit returns always zero.
TSI	12	w	**Set Interrupt Request Flag TSI** Setting this bit sets bit INTSTATAx.TSI. Clearing this bit has no effect. Reading this bit returns always zero.
TSOI	13	w	**Set Interrupt Request Flag TSOI** Setting this bit sets bit INTSTATAx.TSOI. Clearing this bit has no effect. Reading this bit returns always zero.
TOI	14	w	**Set Interrupt Request Flag TOI** Setting this bit sets bit INTSTATAx.TOI. Clearing this bit has no effect. Reading this bit returns always zero.
TOOI	15	w	**Set Interrupt Request Flag TOOI** Setting this bit sets bit INTSTATAx.TOOI. Clearing this bit has no effect. Reading this bit returns always zero.
NFI	16	w	**Set Interrupt Request Flag NFI** Setting this bit sets bit INTSTATAx.NFI. Clearing this bit has no effect. Reading this bit returns always zero.

(table continues...)

(continued)

Field	Bits	Type	Description
0	31:17	r	**Reserved** Read as 0; should be written with 0.

26.4.50 Interrupt Set Register B x

INTSETBx (x=0-1) Offset address: $30C_H+x*4$

Interrupt Set Register B x Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								SCRI5	SCRI4	SCRI3	SCRI2	SCRI1	SCRI0	SOI5	SOI4
r								w	w	w	w	w	w	w	w

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SOI3	SOI2	SOI1	SOI0	SDI5	SDI4	SDI3	SDI2	SDI1	SDI0	WSI5	WSI4	WSI3	WSI2	WSI1	WSI0
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

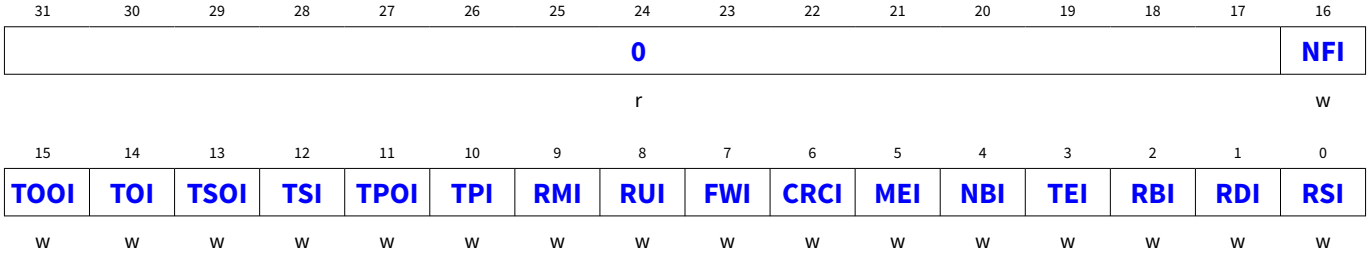
Field	Bits	Type	Description
WSIy (y=0-5)	y	w	**Set Interrupt Request Flag WSIy** Setting this bit sets bit INTSTATBx.WSIy. Clearing this bit has no effect. Reading this bit returns always zero.
SDIy (y=0-5)	y+6	w	**Set Interrupt Request Flag SDIy** Setting this bit sets bit INTSTATBx.SDIy. Clearing this bit has no effect. Reading this bit returns always zero.
SOIy (y=0-5)	y+12	w	**Set Interrupt Request Flag SOIy** Setting this bit sets bit INTSTATBx.SOIy. Clearing this bit has no effect. Reading this bit returns always zero.
SCRIy (y=0-5)	y+18	w	**Set Interrupt Request Flag SCRIy** Setting this bit sets bit INTSTATBx.SCRIy. Clearing this bit has no effect. Reading this bit returns always zero.
0	31:24	r	**Reserved** Read as 0; should be written with 0.

26.4.51 Interrupt Clear Register A x

The Interrupt Clear Register INTCLRA/Bx contain control bits that clear the status of any interrupt of PSI5 channel x.

INTCLRAx (x=0-1) Offset address: $31C_H+x*4$

Interrupt Clear Register A x Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
RSI	0	w	**Clear Interrupt Request Flag RSI** Setting this bit clears bit INTSTATAx.RSI. Clearing this bit has no effect. Reading this bit returns always zero.
RDI	1	w	**Clear Interrupt Request Flag RDI** Setting this bit clears bit INTSTATAx.RDI. Clearing this bit has no effect. Reading this bit returns always zero.
RBI	2	w	**Clear Interrupt Request Flag RBI** Setting this bit clears bit INTSTATAx.RBI. Clearing this bit has no effect. Reading this bit returns always zero.
TEI	3	w	**Clear Interrupt Request Flag TEI** Setting this bit clears bit INTSTATAx.TEI. Clearing this bit has no effect. Reading this bit returns always zero.
NBI	4	w	**Clear Interrupt Request Flag NBI** Setting this bit clears bit INTSTATAx.NBI. Clearing this bit has no effect. Reading this bit returns always zero.
MEI	5	w	**Clear Interrupt Request Flag MEI** Setting this bit clears bit INTSTATAx.MEI. Clearing this bit has no effect. Reading this bit returns always zero.
CRCI	6	w	**Clear Interrupt Request Flag CRCI** Setting this bit clears bit INTSTATAx.CRCI. Clearing this bit has no effect. Reading this bit returns always zero.
FWI	7	w	**Clear Interrupt Request Flag FWI** Setting this bit clears bit INTSTATAx.FWI. Clearing this bit has no effect. Reading this bit returns always zero.

(table continues...)

(continued)

Field	Bits	Type	Description
RUI	8	w	**Clear Interrupt Request Flag RUI** Setting this bit clears bit INTSTATAx.RUI. Clearing this bit has no effect. Reading this bit returns always zero.
RMI	9	w	**Clear Interrupt Request Flag RMI** Setting this bit clears bit INTSTATAx.RMI. Clearing this bit has no effect. Reading this bit returns always zero.
TPI	10	w	**Clear Interrupt Request Flag TPI** Setting this bit clears bit INTSTATAx.TPI. Clearing this bit has no effect. Reading this bit returns always zero.
TPOI	11	w	**Clear Interrupt Request Flag TPOI** Setting this bit clears bit INTSTATAx.TPOI. Clearing this bit has no effect. Reading this bit returns always zero.
TSI	12	w	**Clear Interrupt Request Flag TSI** Setting this bit clears bit INTSTATAx.TSI. Clearing this bit has no effect. Reading this bit returns always zero.
TSOI	13	w	**Clear Interrupt Request Flag TSOI** Setting this bit clears bit INTSTATAx.TSOI. Clearing this bit has no effect. Reading this bit returns always zero.
TOI	14	w	**Clear Interrupt Request Flag TOI** Setting this bit clears bit INTSTATAx.TOI. Clearing this bit has no effect. Reading this bit returns always zero.
TOOI	15	w	**Clear Interrupt Request Flag TOOI** Setting this bit clears bit INTSTATAx.TOOI. Clearing this bit has no effect. Reading this bit returns always zero.
NFI	16	w	**Clear Interrupt Request Flag NFI** Setting this bit clears bit INTSTATAx.NFI. Clearing this bit has no effect. Reading this bit returns always zero.
0	31:17	r	**Reserved** Read as 0; should be written with 0.

26.4.52 Interrupt Clear Register B x

INTCLRBx (x=0-1) Offset address: $32C_H+x*4$

Interrupt Clear Register B x Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								SCRI5	SCRI4	SCRI3	SCRI2	SCRI1	SCRI0	SOI5	SOI4
r								w	w	w	w	w	w	w	w

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SOI3	SOI2	SOI1	SOI0	SDI5	SDI4	SDI3	SDI2	SDI1	SDI0	WSI5	WSI4	WSI3	WSI2	WSI1	WSI0
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

Field	Bits	Type	Description
WSIy (y=0-5)	y	w	**Clear Interrupt Request Flag WSIy** Setting this bit clears bit INTSTATBx.WSIy. Clearing this bit has no effect. Reading this bit returns always zero.
SDIy (y=0-5)	y+6	w	**Clear Interrupt Request Flag SDIy** Setting this bit clears bit INTSTATBx.SDIy. Clearing this bit has no effect. Reading this bit returns always zero.
SOIy (y=0-5)	y+12	w	**Clear Interrupt Request Flag SOIy** Setting this bit clears bit INTSTATBx.SOIy. Clearing this bit has no effect. Reading this bit returns always zero.
SCRIy (y=0-5)	y+18	w	**Clear Interrupt Request Flag SCRIy** Setting this bit clears bit INTSTATBx.SCRIy. Clearing this bit has no effect. Reading this bit returns always zero.
0	31:24	r	**Reserved** Read as 0; should be written with 0.

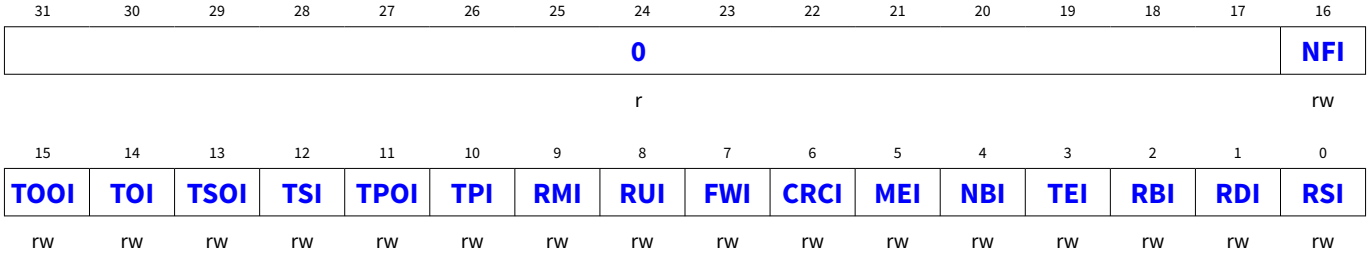
26.4.53 Interrupt Enable Register A x

The Interrupt Enable Register INTENA/B x contain control bits that enable the interrupt source of any interrupt of PSI5 channel x.

The Interrupt Status bits in register INTSTATA/Bx are set independently from the Interrupt Enable in Register INTENA/Bx.

INTENAx (x=0-1) Offset address: $33C_H+x*4$

Interrupt Enable Register A x Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
RSI	0	rw	**Enable Interrupt Request RSI** 0_B No interrupt request can be generated for this source 1_B An interrupt request can be generated for this source
RDI	1	rw	**Enable Interrupt Request RDI** 0_B No interrupt request can be generated for this source 1_B An interrupt request can be generated for this source
RBI	2	rw	**Enable Interrupt Request RBI** 0_B No interrupt request can be generated for this source 1_B An interrupt request can be generated for this source
TEI	3	rw	**Enable Interrupt Request TEI** 0_B No interrupt request can be generated for this source 1_B An interrupt request can be generated for this source
NBI	4	rw	**Enable Interrupt Request NBI** 0_B No interrupt request can be generated for this source 1_B An interrupt request can be generated for this source
MEI	5	rw	**Enable Interrupt Request MEI** 0_B No interrupt request can be generated for this source 1_B An interrupt request can be generated for this source
CRCI	6	rw	**Enable Interrupt Request CRCI** 0_B No interrupt request can be generated for this source 1_B An interrupt request can be generated for this source
FWI	7	rw	**Enable Interrupt Request FWI** 0_B No interrupt request can be generated for this source 1_B An interrupt request can be generated for this source
RUI	8	rw	**Enable Interrupt Request RUI** 0_B No interrupt request can be generated for this source 1_B An interrupt request can be generated for this source
RMI	9	rw	**Enable Interrupt Request RMI** 0_B No interrupt request can be generated for this source 1_B An interrupt request can be generated for this source
TPI	10	rw	**Enable Interrupt Request TPI** 0_B No interrupt request can be generated for this source 1_B An interrupt request can be generated for this source

(table continues...)

(continued)

Field	Bits	Type	Description
TPOI	11	rw	**Enable Interrupt Request TPOI** 0_B No interrupt request can be generated for this source 1_B An interrupt request can be generated for this source
TSI	12	rw	**Enable Interrupt Request TSI** 0_B No interrupt request can be generated for this source 1_B An interrupt request can be generated for this source
TSOI	13	rw	**Enable Interrupt Request TSOI** 0_B No interrupt request can be generated for this source 1_B An interrupt request can be generated for this source
TOI	14	rw	**Enable Interrupt Request TOI** 0_B No interrupt request can be generated for this source 1_B An interrupt request can be generated for this source
TOOI	15	rw	**Enable Interrupt Request TOOI** 0_B No interrupt request can be generated for this source 1_B An interrupt request can be generated for this source
NFI	16	rw	**Enable Interrupt Request NFI** 0_B No interrupt request can be generated for this source 1_B An interrupt request can be generated for this source
0	31:17	r	**Reserved** Read as 0; should be written with 0.

26.4.54 Interrupt Enable Register B x

INTENBx (x=0-1) Offset address: $34C_H+x*4$

Interrupt Enable Register B x Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								SCRI5	SCRI4	SCRI3	SCRI2	SCRI1	SCRI0	SOI5	SOI4
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SOI3	SOI2	SOI1	SOI0	SDI5	SDI4	SDI3	SDI2	SDI1	SDI0	WSI5	WSI4	WSI3	WSI2	WSI1	WSI0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
WSIy (y=0-5)	y	rw	**Enable Interrupt Request WSIy** 0_B No interrupt request can be generated for this source 1_B An interrupt request can be generated for this source

(table continues...)

(continued)

Field	Bits	Type	Description
SDIy (y=0-5)	y+6	rw	**Enable Interrupt Request SDIy** 0_B No interrupt request can be generated for this source 1_B An interrupt request can be generated for this source
SOIy (y=0-5)	y+12	rw	**Enable Interrupt Request SOIy** 0_B No interrupt request can be generated for this source 1_B An interrupt request can be generated for this source
SCRIy (y=0-5)	y+18	rw	**Enable Interrupt Request SCRIy** 0_B No interrupt request can be generated for this source 1_B An interrupt request can be generated for this source
0	31:24	r	**Reserved** Read as 0; should be written with 0.

26.4.55 Receive FIFO Control Register x

RFCx contains control bits for the FIFO and ring buffer operation.

RFCx (x=0-1) Offset address: 378_H+x*4

Receive FIFO Control Register x Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLU	FRQ	WRAP	0								FWL				
w	rh	rh	r								rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		WRP						0		REP					
r		rh						r		rh					

Field	Bits	Type	Description
REP	5:0	rh	**FIFO Read Pointer** points to the buffer to be read next. Incremented after read. The LSB bit indicates if RDMxH or RDMxL is read. This LSB is ignored for FWL comparison WRP - REP collision (RBI, RUI)
WRP	13:8	rh	**FIFO/Ring Buffer Write Pointer** points to the buffer written last. Incremented after write. The LSB indicates if RDMxH or RDMxL is written. This LSB is ignored for FWL comparison WRP - REP collision (RBI, RUI).
FWL	20:16	rw	**FIFO Warning Level** Number of new entries at which the FIFO Warning Interrupt FWIx is set automatically. For calculation WRP and REP are used.Which means, only if the memory is read from RDFx, this works properly.

(table continues...)

(continued)

Field	Bits	Type	Description
WRAP	29	rh	**Write Pointer WRAP Indicator** If set, the Write Pointer is one wrap around ahead of the Read Pointer. It is cleared, if the Read Pointer wraps around too or when FLU is set by SW.
FRQ	30	rh	**Flush Request** If set, FIFO read and write pointers are not incremented any more due to excessive RMI (too many overrun conditions). This bit is cleared when FLU is set.
FLU	31	w	**Flush** If set, FIFO read and write pointers are cleared. Bit WRAP is cleared. SW needs to clear interrupts as needed. This bit can only be set and is cleared by HW. Always read as 0.
0	7:6, 15:14, 28:21	r	**Reserved** Read as 0; should be written with 0.

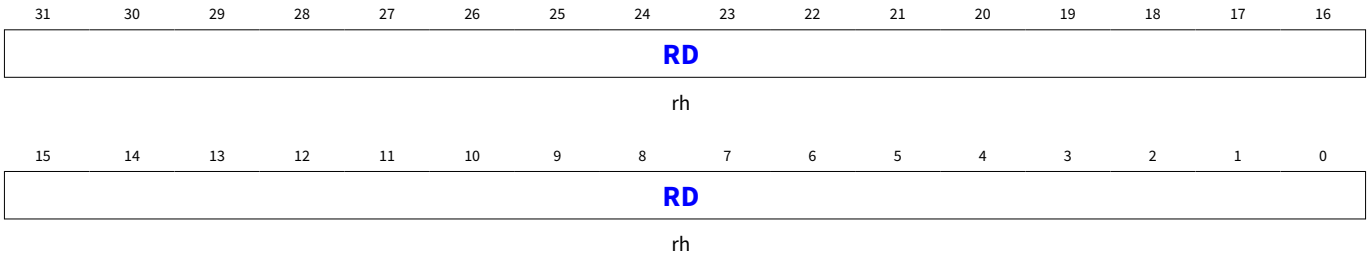
26.4.56 Receive Data FIFO x

The Receive Data FIFO RDFx for channel x shows the oldest data content of up to 32 received data frames. That is, RFCx.REP is pointing at the buffer presented.

Do not set a debugger watch point on this register as any read access will issue a hardware request for new data (move read pointer REP and present referring memory content in RDFx). A debugger watch point can be set on the RDML/Hx and RFCx to monitor the FIFO behavior.

RDFx (x=0-1) Offset address: 388_H+x*4

Receive Data FIFO x Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
RD	31:0	rh	**RD** Shows the content of RDML/Hxy; y = RFCx.REP[5:1]; L/H = RFCx.REP[0]. Reading this register triggers incrementation of REP and presentation of next FIFO value at next read access. Once a complete buffer is read (64 bit, 2 accesses to RDF) all overview flags for this very buffer are cleared automatically.

26.4.57 RSI Overview Register x

The RSI Overview Register RSIOVx contains flags to manage the receive memory of PSI5 channel x.

RSIOVx (x=0-1) Offset address: 398_H+x*4

RSI Overview Register x Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSI31	RSI30	RSI29	RSI28	RSI27	RSI26	RSI25	RSI24	RSI23	RSI22	RSI21	RSI20	RSI19	RSI18	RSI17	RSI16
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSI15	RSI14	RSI13	RSI12	RSI11	RSI10	RSI9	RSI8	RSI7	RSI6	RSI5	RSI4	RSI3	RSI2	RSI1	RSI0
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
RSIy (y=0-31)	y	rh	**RSI Flag of Buffer y** Copied from INTSTATAx at frame end while RFCx.WRP pointed to buffer y. This bit can be cleared by bit RSICLRx.RSIy. This bit can be set by bit RSISETx.RSIy.

26.4.58 RMI Overview Register x

The RMI Overview Register RMIOVx contains flags to manage the receive memory of PSI5 channel x.

RMIOVx (x=0-1) Offset address: $3A8_H+x*4$

RMI Overview Register x Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RMI31	RMI30	RMI29	RMI28	RMI27	RMI26	RMI25	RMI24	RMI23	RMI22	RMI21	RMI20	RMI19	RMI18	RMI17	RMI16
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RMI15	RMI14	RMI13	RMI12	RMI11	RMI10	RMI9	RMI8	RMI7	RMI6	RMI5	RMI4	RMI3	RMI2	RMI1	RMI0
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
RMIy (y=0-31)	y	rh	**RMI Flag of Buffer y** Copied from INTSTATAx at frame end while RFCx.WRP pointed to buffer y. This bit can be cleared by bit RMICLRx.RMIy. This bit can be set by bit RMISETx.RMIy.

26.4.59 NBI Overview Register x

The NBI Overview Register NBIOVx contains flags to manage the receive memory of PSI5 channel x.

NBIOVx (x=0-1) Offset address: $3B8_H+x*4$

NBI Overview Register x Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
NBI31	NBI30	NBI29	NBI28	NBI27	NBI26	NBI25	NBI24	NBI23	NBI22	NBI21	NBI20	NBI19	NBI18	NBI17	NBI16
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NBI15	NBI14	NBI13	NBI12	NBI11	NBI10	NBI9	NBI8	NBI7	NBI6	NBI5	NBI4	NBI3	NBI2	NBI1	NBI0
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
NBIy (y=0-31)	y	rh	**NBI Flag of Buffer y** Copied from INTSTATAx at frame end while RFCx.WRP pointed to buffer y. This bit can be cleared by bit NBICLRx.NBIy. This bit can be set by bit NBISETx.NBIy.

26.4.60 TEI Overview Register x

The TEI Overview Register TEIOVx contains flags to manage the receive memory of PSI5 channel x.

TEIOVx (x=0-1) Offset address: $3C8_H+x*4$

TEI Overview Register x Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TEI31	TEI30	TEI29	TEI28	TEI27	TEI26	TEI25	TEI24	TEI23	TEI22	TEI21	TEI20	TEI19	TEI18	TEI17	TEI16
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TEI15	TEI14	TEI13	TEI12	TEI11	TEI10	TEI9	TEI8	TEI7	TEI6	TEI5	TEI4	TEI3	TEI2	TEI1	TEI0
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
TEIy (y=0-31)	y	rh	**TEI Flag of Buffer y** Copied from INTSTATAx at frame end while RFCx.WRP pointed to buffer y. This bit can be cleared by bit TEICLRx.TEIy. This bit can be set by bit TEISETx.TEIy.

26.4.61 CRCI Overview Register x

The CRCI Overview Register CRCIOVx contains flags to manage the receive memory of PSI5 channel x.

CRCIOVx (x=0-1) Offset address: $3D8_H+x*4$

CRCI Overview Register x Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CRCI 31	CRCI 30	CRCI 29	CRCI 28	CRCI 27	CRCI 26	CRCI 25	CRCI 24	CRCI 23	CRCI 22	CRCI 21	CRCI 20	CRCI 19	CRCI 18	CRCI 17	CRCI 16
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CRCI 15	CRCI 14	CRCI 13	CRCI 12	CRCI 11	CRCI 10	CRCI 9	CRCI 8	CRCI 7	CRCI 6	CRCI 5	CRCI 4	CRCI 3	CRCI 2	CRCI 1	CRCI 0
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
CRCIy (y=0-31)	y	rh	**CRCI Flag of Buffer y** Copied from INTSTATAx at frame end while RFCx.WRP pointed to buffer y. This bit can be cleared by bit CRCICLRx.CRCIy. This bit can be set by bit CRCISETx.CRCIy.

26.4.62 RDI Overview Register x

The RDI Overview Register RDIOVx contains flags to manage the receive memory of PSI5 channel x.

RDIOVx (x=0-1) Offset address: $3E8_H+x*4$

RDI Overview Register x Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RDI31	RDI30	RDI29	RDI28	RDI27	RDI26	RDI25	RDI24	RDI23	RDI22	RDI21	RDI20	RDI19	RDI18	RDI17	RDI16
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RDI15	RDI14	RDI13	RDI12	RDI11	RDI10	RDI9	RDI8	RDI7	RDI6	RDI5	RDI4	RDI3	RDI2	RDI1	RDI0
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
RDIy (y=0-31)	y	rh	**RDI Flag of Buffer y** Copied from INTSTATAx at frame end while RFCx.WRP pointed to buffer y. This bit can be cleared by bit RDICLRx.RDIy. This bit can be set by bit RDISETx.RDIy.

26.4.63 NFI Overview Register x

The NFI Overview Register NFIOVx contains flags to manage the receive memory of PSI5 channel x.

NFIOVx (x=0-1) Offset address: $3F8_H+x*4$

NFI Overview Register x Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
NFI31	NFI30	NFI29	NFI28	NFI27	NFI26	NFI25	NFI24	NFI23	NFI22	NFI21	NFI20	NFI19	NFI18	NFI17	NFI16
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NFI15	NFI14	NFI13	NFI12	NFI11	NFI10	NFI9	NFI8	NFI7	NFI6	NFI5	NFI4	NFI3	NFI2	NFI1	NFI0
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
NFIy (y=0-31)	y	rh	**NFI Flag of Buffer y** Copied from INTSTATAx at frame end while RFCx.WRP pointed to buffer y. This bit can be cleared by bit NFICLRx.NFIy. This bit can be set by bit NFISETx.NFIy.

26.4.64 MEI Overview Register x

The MEI Overview Register MEIOVx contains flags to manage the receive memory of PSI5 channel x.

MEIOVx (x=0-1) Offset address: 408_H+x*4

MEI Overview Register x Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MEI31	MEI30	MEI29	MEI28	MEI27	MEI26	MEI25	MEI24	MEI23	MEI22	MEI21	MEI20	MEI19	MEI18	MEI17	MEI16
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MEI15	MEI14	MEI13	MEI12	MEI11	MEI10	MEI9	MEI8	MEI7	MEI6	MEI5	MEI4	MEI3	MEI2	MEI1	MEI0
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
MEIy (y=0-31)	y	rh	**MEI Flag of Buffer y** Copied from INTSTATAx at frame end while RFCx.WRP pointed to buffer y. This bit can be cleared by bit MEICLRx.MEIy. This bit can be set by bit MEISETx.MEIy.

26.4.65 RSI Overview Set Register x

The RSI Overview Set Register RSISETx contains bits to set the status flags of RSIOVx.

RSISETx (x=0-1) Offset address: 418_H+x*4

RSI Overview Set Register x Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSI31	RSI30	RSI29	RSI28	RSI27	RSI26	RSI25	RSI24	RSI23	RSI22	RSI21	RSI20	RSI19	RSI18	RSI17	RSI16
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSI15	RSI14	RSI13	RSI12	RSI11	RSI10	RSI9	RSI8	RSI7	RSI6	RSI5	RSI4	RSI3	RSI2	RSI1	RSI0
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

Field	Bits	Type	Description
RSIy (y=0-31)	y	w	**Set RSI Flag of Buffer y** Setting this bit sets bit RSIOVx.RSIy Clearing this bit has no effect. Reading this bit returns always zero.

26.4.66 RMI Overview Set Register x

The RMI Overview Set Register RMISETx contains bits to set the status flags of RMIOVx.

RMISETx (x=0-1) Offset address: 428_H+x*4

RMI Overview Set Register x Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RMI31	RMI30	RMI29	RMI28	RMI27	RMI26	RMI25	RMI24	RMI23	RMI22	RMI21	RMI20	RMI19	RMI18	RMI17	RMI16
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RMI15	RMI14	RMI13	RMI12	RMI11	RMI10	RMI9	RMI8	RMI7	RMI6	RMI5	RMI4	RMI3	RMI2	RMI1	RMI0
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

Field	Bits	Type	Description
RMIy (y=0-31)	y	w	**Set RMI Flag of Buffer y** Setting this bit sets bit RMIOVx.RMIy Clearing this bit has no effect. Reading this bit returns always zero.

26.4.67 NBI Overview Set Register x

The NBI Overview Set Register NBISETx contains bits to set the status flags of NBIOVx.

NBISETx (x=0-1) Offset address: 438_H+x*4

NBI Overview Set Register x Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
NBI31	NBI30	NBI29	NBI28	NBI27	NBI26	NBI25	NBI24	NBI23	NBI22	NBI21	NBI20	NBI19	NBI18	NBI17	NBI16
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NBI15	NBI14	NBI13	NBI12	NBI11	NBI10	NBI9	NBI8	NBI7	NBI6	NBI5	NBI4	NBI3	NBI2	NBI1	NBI0
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

Field	Bits	Type	Description
NBIy (y=0-31)	y	w	**Set NBI Flag of Buffer y** Setting this bit sets bit NBIOVx.NBIy Clearing this bit has no effect. Reading this bit returns always zero.

26.4.68 TEI Overview Set Register x

The TEI Overview Set Register TEISETx contains bits to set the status flags of TEIOVx.

TEISETx (x=0-1) Offset address: 448_H+x*4

TEI Overview Set Register x Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TEI31	TEI30	TEI29	TEI28	TEI27	TEI26	TEI25	TEI24	TEI23	TEI22	TEI21	TEI20	TEI19	TEI18	TEI17	TEI16
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TEI15	TEI14	TEI13	TEI12	TEI11	TEI10	TEI9	TEI8	TEI7	TEI6	TEI5	TEI4	TEI3	TEI2	TEI1	TEI0
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

Field	Bits	Type	Description
TEIy (y=0-31)	y	w	**Set TEI Flag of Buffer y** Setting this bit sets bit TEIOVx.TEIy Clearing this bit has no effect. Reading this bit returns always zero.

26.4.69 CRCI Overview Set Register x

The CRCI Overview Set Register CRCISETx contains bits to set the status flags of CRCIOVx.

CRCISETx (x=0-1) Offset address: 458_H+x*4

CRCI Overview Set Register x Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CRCI 31	CRCI 30	CRCI 29	CRCI 28	CRCI 27	CRCI 26	CRCI 25	CRCI 24	CRCI 23	CRCI 22	CRCI 21	CRCI 20	CRCI 19	CRCI 18	CRCI 17	CRCI 16
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CRCI 15	CRCI 14	CRCI 13	CRCI 12	CRCI 11	CRCI 10	CRCI 9	CRCI 8	CRCI 7	CRCI 6	CRCI 5	CRCI 4	CRCI 3	CRCI 2	CRCI 1	CRCI 0
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

Field	Bits	Type	Description
CRCIy (y=0-31)	y	w	**Set CRCI Flag of Buffer y** Setting this bit sets bit CRCIOVx.CRCIy Clearing this bit has no effect. Reading this bit returns always zero.

26.4.70 RDI Overview Set Register x

The RDI Overview Set Register RDISETx contains bits to set the status flags of RDIOVx.

RDISETx (x=0-1) Offset address: 468_H+x*4

RDI Overview Set Register x Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RDI31	RDI30	RDI29	RDI28	RDI27	RDI26	RDI25	RDI24	RDI23	RDI22	RDI21	RDI20	RDI19	RDI18	RDI17	RDI16
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RDI15	RDI14	RDI13	RDI12	RDI11	RDI10	RDI9	RDI8	RDI7	RDI6	RDI5	RDI4	RDI3	RDI2	RDI1	RDI0
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

Field	Bits	Type	Description
RDIy (y=0-31)	y	w	**Set RDI Flag of Buffer y** Setting this bit sets bit RDIOVx.RDIy Clearing this bit has no effect. Reading this bit returns always zero.

26.4.71 NFI Overview Set Register x

The NFI Overview Set Register NFISETx contains bits to set the status flags of NFIOVx.

NFISETx (x=0-1) Offset address: 478_H+x*4

NFI Overview Set Register x Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
NFI31	NFI30	NFI29	NFI28	NFI27	NFI26	NFI25	NFI24	NFI23	NFI22	NFI21	NFI20	NFI19	NFI18	NFI17	NFI16
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NFI15	NFI14	NFI13	NFI12	NFI11	NFI10	NFI9	NFI8	NFI7	NFI6	NFI5	NFI4	NFI3	NFI2	NFI1	NFI0
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

Field	Bits	Type	Description
NFIy (y=0-31)	y	w	**Set NFI Flag of Buffer y** Setting this bit sets bit NFIOVx.NFIy Clearing this bit has no effect. Reading this bit returns always zero.

26.4.72 MEI Overview Set Register x

The MEI Overview Set Register MEISETx contains bits to set the status flags of MEIOVx.

MEISETx (x=0-1) Offset address: 488_H+x*4

MEI Overview Set Register x Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MEI31	MEI30	MEI29	MEI28	MEI27	MEI26	MEI25	MEI24	MEI23	MEI22	MEI21	MEI20	MEI19	MEI18	MEI17	MEI16
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MEI15	MEI14	MEI13	MEI12	MEI11	MEI10	MEI9	MEI8	MEI7	MEI6	MEI5	MEI4	MEI3	MEI2	MEI1	MEI0
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

Field	Bits	Type	Description
MEIy (y=0-31)	y	w	**Set MEI Flag of Buffer y** Setting this bit sets bit MEIOVx.MEIy Clearing this bit has no effect. Reading this bit returns always zero.

26.4.73 RSI Overview Clear Register x

The RSI Overview Clear Register RSICLRx contains bits to clear the status flags of RSIOVx.

RSICLRx (x=0-1) Offset address: 498_H+x*4

RSI Overview Clear Register x Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSI31	RSI30	RSI29	RSI28	RSI27	RSI26	RSI25	RSI24	RSI23	RSI22	RSI21	RSI20	RSI19	RSI18	RSI17	RSI16
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSI15	RSI14	RSI13	RSI12	RSI11	RSI10	RSI9	RSI8	RSI7	RSI6	RSI5	RSI4	RSI3	RSI2	RSI1	RSI0
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

Field	Bits	Type	Description
RSIy (y=0-31)	y	w	**Clear RSI Flag of Buffer y** Setting this bit clears bit RSIOVx.RSIy Clearing this bit has no effect. Reading this bit returns always zero.

26.4.74 RMI Overview Clear Register x

The RMI Overview Clear Register RMICLRx contains bits to clear the status flags of RMIOVx.

RMICLRx (x=0-1) Offset address: $4A8_H+x*4$

RMI Overview Clear Register x Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RMI31	RMI30	RMI29	RMI28	RMI27	RMI26	RMI25	RMI24	RMI23	RMI22	RMI21	RMI20	RMI19	RMI18	RMI17	RMI16
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RMI15	RMI14	RMI13	RMI12	RMI11	RMI10	RMI9	RMI8	RMI7	RMI6	RMI5	RMI4	RMI3	RMI2	RMI1	RMI0
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

Field	Bits	Type	Description
RMIy (y=0-31)	y	w	**Clear RMI Flag of Buffer y** Setting this bit clears bit RMIOVx.RMIy Clearing this bit has no effect. Reading this bit returns always zero.

26.4.75 NBI Overview Clear Register x

The NBI Overview Clear Register NBICLRx contains bits to clear the status flags of NBIOVx.

NBICLRx (x=0-1) Offset address: $4B8_H+x*4$

NBI Overview Clear Register x Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
NBI31	NBI30	NBI29	NBI28	NBI27	NBI26	NBI25	NBI24	NBI23	NBI22	NBI21	NBI20	NBI19	NBI18	NBI17	NBI16
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NBI15	NBI14	NBI13	NBI12	NBI11	NBI10	NBI9	NBI8	NBI7	NBI6	NBI5	NBI4	NBI3	NBI2	NBI1	NBI0
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

Field	Bits	Type	Description
NBIy (y=0-31)	y	w	**Clear NBI Flag of Buffer y** Setting this bit clears bit NBIOVx.NBIy Clearing this bit has no effect. Reading this bit returns always zero.

26.4.76 TEI Overview Clear Register x

The TEI Overview Clear Register TEICLRx contains bits to clear the status flags of TEIOVx.

TEICLRx (x=0-1) Offset address: $4C8_H+x*4$

TEI Overview Clear Register x Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TEI31	TEI30	TEI29	TEI28	TEI27	TEI26	TEI25	TEI24	TEI23	TEI22	TEI21	TEI20	TEI19	TEI18	TEI17	TEI16
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TEI15	TEI14	TEI13	TEI12	TEI11	TEI10	TEI9	TEI8	TEI7	TEI6	TEI5	TEI4	TEI3	TEI2	TEI1	TEI0
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

Field	Bits	Type	Description
TEIy (y=0-31)	y	w	**Clear TEI Flag of Buffer y** Setting this bit clears bit TEIOVx.TEIy Clearing this bit has no effect. Reading this bit returns always zero.

26.4.77 CRCI Overview Clear Register x

The CRCI Overview Clear Register CRCICLRx contains bits to clear the status flags of CRCIOVx.

CRCICLRx (x=0-1) Offset address: $4D8_H+x*4$

CRCI Overview Clear Register x Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CRCI 31	CRCI 30	CRCI 29	CRCI 28	CRCI 27	CRCI 26	CRCI 25	CRCI 24	CRCI 23	CRCI 22	CRCI 21	CRCI 20	CRCI 19	CRCI 18	CRCI 17	CRCI 16
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CRCI 15	CRCI 14	CRCI 13	CRCI 12	CRCI 11	CRCI 10	CRCI 9	CRCI 8	CRCI 7	CRCI 6	CRCI 5	CRCI 4	CRCI 3	CRCI 2	CRCI 1	CRCI 0
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

Field	Bits	Type	Description
CRCIy (y=0-31)	y	w	**Clear CRCI Flag of Buffer y** Setting this bit clears bit CRCIOVx.CRCIy Clearing this bit has no effect. Reading this bit returns always zero.

26.4.78 RDI Overview Clear Register x

The RDI Overview Clear Register RDICLRx contains bits to clear the status flags of RDIOVx.

RDICLRx (x=0-1) Offset address: $4E8_H+x*4$

RDI Overview Clear Register x Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RDI31	RDI30	RDI29	RDI28	RDI27	RDI26	RDI25	RDI24	RDI23	RDI22	RDI21	RDI20	RDI19	RDI18	RDI17	RDI16
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RDI15	RDI14	RDI13	RDI12	RDI11	RDI10	RDI9	RDI8	RDI7	RDI6	RDI5	RDI4	RDI3	RDI2	RDI1	RDI0
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

Field	Bits	Type	Description
RDIy (y=0-31)	y	w	**Clear RDI Flag of Buffer y** Setting this bit clears bit RDIOVx.RDIy Clearing this bit has no effect. Reading this bit returns always zero.

26.4.79 NFI Overview Clear Register x

The NFI Overview Clear Register NFICLRx contains bits to clear the status flags of NFIOVx.

NFICLRx (x=0-1) Offset address: $4F8_H$+x*4

NFI Overview Clear Register x Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
NFI31	NFI30	NFI29	NFI28	NFI27	NFI26	NFI25	NFI24	NFI23	NFI22	NFI21	NFI20	NFI19	NFI18	NFI17	NFI16
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NFI15	NFI14	NFI13	NFI12	NFI11	NFI10	NFI9	NFI8	NFI7	NFI6	NFI5	NFI4	NFI3	NFI2	NFI1	NFI0
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

Field	Bits	Type	Description
NFIy (y=0-31)	y	w	**Clear NFI Flag of Buffer y** Setting this bit clears bit NFIOVx.NFIy Clearing this bit has no effect. Reading this bit returns always zero.

26.4.80 MEI Overview Clear Register x

The MEI Overview Clear Register MEICLRx contains bits to clear the status flags of MEIOVx.

MEICLRx (x=0-1) Offset address: 508_H+x*4

MEI Overview Clear Register x Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MEI31	MEI30	MEI29	MEI28	MEI27	MEI26	MEI25	MEI24	MEI23	MEI22	MEI21	MEI20	MEI19	MEI18	MEI17	MEI16
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MEI15	MEI14	MEI13	MEI12	MEI11	MEI10	MEI9	MEI8	MEI7	MEI6	MEI5	MEI4	MEI3	MEI2	MEI1	MEI0
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

Field	Bits	Type	Description
MEIy (y=0-31)	y	w	**Clear MEI Flag of Buffer y** Setting this bit clears bit MEIOVx.MEIy Clearing this bit has no effect. Reading this bit returns always zero.

26.4.81 Receive Data Memory Low xy

The Receive Data Memory RDMLx for channel x shows the data content of up to 32 received data frames.

The internal receive buffer is always cleared (0x0000 0000$_H$) at each frame start. Thus unused nibbles are always read as zero.

RDMLxy (x=0-1;y=0-31) Offset address: $600_H+x*100_H+y*8$

Receive Data Memory Low xy Kernel Reset value: 0000 0000$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RD															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RD												CRC			CRCI
rh												rh			rh

Field	Bits	Type	Description
CRCI	0	rh	**CRC Error Flag** Copied from RDRLx at frame end.
CRC	3:1	rh	**CRC** Copied from RDRLx at frame end.
RD	31:4	rh	**RD** Copied from RDRLx at frame end.

26.4.82 Receive Data Memory High xy

The Receive Data Memory RDMHx for channel x shows the data content of up to 32 received data frames.

The internal receive buffer is always cleared (0x0000 0000$_H$) at each frame start. Thus unused nibbles are always read as zero.

RDMHxy (x=0-1;y=0-31) Offset address: $604_H+x*100_H+y*8$

Receive Data Memory High xy Kernel Reset value: 0000 0000$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RMI	NFI	MEI	NBI	TEI	SC			TS							
rh	rh	rh	rh	rh	rh			rh							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TS															
rh															

Field	Bits	Type	Description
TS	23:0	rh	**Timestamp** Copied from RDRHx at frame end.
SC	26:24	rh	**Slot Counter** Copied from RDRHx at frame end.

(table continues...)

(continued)

Field	Bits	Type	Description
TEI	27	rh	**Time Slot Error Flag** Copied from RDRHx at frame end.
NBI	28	rh	**Number of bits Error Flag** Copied from RDRHx at frame end.
MEI	29	rh	**Error in Messaging Bits Flag** Copied from RDRHx at frame end.
NFI	30	rh	**No Frame Received Flag** Copied from RDRHx at frame end.
RMI	31	rh	**Receive Memory Overflow Flag** Copied from INTSTATAx at frame end.

26.5 Debug information

This section is not applicable for the module being described.

26.6 References

1. PSI5 Peripheral Sensor Interface V2.1 Technical Specification
2. PSI5 Peripheral Sensor Interface V1.3 Technical Specification
3. SENT—Single Edge Nibble Transmission for Automotive Applications J2716_201001

26.7 PSI5 revision history

Initial release of the chapter.

26.8 TC4Dx PSI5 information

26.8.1 TC4Dx PSI5 configuration

Table 1187 TC4Dx PSI5 configuration

Parameters	PSI5
Number of channels	2
RAM size (in Kbytes)	0.5

26.8.2 TC4Dx PSI5 features

There are no deviations from the generic specification.

26.8.3 TC4Dx PSI5 functional description

1. PSI5 acts as a slave on the ComPB bus. The Clock Control block is driven by the system clock frequency f_{SPB} as in TC4Dx, the ComPB also uses this clock
2. There is no GTM trigger available. Only eGTM trigger signals are used for Time stamp generation and Sync pulse control

26.8.4 TC4Dx PSI5 registers

There are no deviations from the generic specification.

26.8.4.1 Register address space - PSI5

Table 1188 Registers address space - PSI5

Module	Base address	End address	Note
PSI5	$F4405000_H$	$F4405BFF_H$	FPI slave interface

26.8.4.2 Register overview - access mode glossary

Table 1189 Register overview - access mode glossary

Keyword	Description
E	Access protection using PROT register PSI5_PROTE .
SE	Access protection using PROT register PSI5_PROTSE .
APU-P	Protection group consisting of registers PSI5_ACCEN_WRA , PSI5_ACCEN_WRB , PSI5_ACCEN_RDA , PSI5_ACCEN_RDB , PSI5_ACCEN_VM , PSI5_ACCEN_PRS .
P	Access protection using *APU-P* registers.
BE	Always returns a Bus Error.
U	No access restrictions.
PROT	Access restrictions as defined in the PROT register access rules.

26.8.4.3 Registers overview - PSI5 (ascending offset address)

Table 1190 Registers overview - PSI5 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PSI5_CLC	Clock Control Register	000_H	P	P, SV, E	Application Reset	4885
PSI5_OCS	OCDS Control and Status Register	004_H	P	SV, P	Debug Reset	4886
PSI5_ID	Module Identification Register	008_H	P	BE	PowerOn Reset	4886
PSI5_RST_CTRLA	Reset Control Register A	$00C_H$	P	P, SV, E	Application Reset	4887
PSI5_RST_CTRLB	Reset Control Register B	010_H	P	P, SV, E	Application Reset	4888
PSI5_RST_STAT	Reset Status Register	014_H	P	BE	Application Reset	4888
PSI5_PROTE	PROT Register Endinit	020_H	U	SV, PROT	Application Reset	4889
PSI5_PROTSE	PROT Register Safe Endinit	024_H	U	SV, PROT	Application Reset	4891

(table continues...)

Table 1190 (continued) Registers overview - PSI5 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PSI5_ACCEN_WRA	Write access enable register A	040_H	U	SE, SV	Application Reset	4892
PSI5_ACCEN_WRB	Write access enable register B	044_H	U	SE, SV	Application Reset	4893
PSI5_ACCEN_RDA	Read access enable register A	048_H	U	SE, SV	Application Reset	4893
PSI5_ACCEN_RDB	Read access enable register B	$04C_H$	U	SE, SV	Application Reset	4894
PSI5_ACCEN_VM	VM access enable register	050_H	U	SE, SV	Application Reset	4894
PSI5_ACCEN_PRS	PRS access enable register	054_H	U	SE, SV	Application Reset	4895
PSI5_FDR	Fractional Divider Register	060_H	P	SV, E, P	Kernel Reset	4896
PSI5_FDRL	Fractional Divider Register for Lower Bit Rate	064_H	P	SV, E, P	Kernel Reset	4896
PSI5_FDRH	Fractional Divider Register for Higher Bit Rate	068_H	P	SV, E, P	Kernel Reset	4897
PSI5_FDRT	Fractional Divider Register for Timestamp	$06C_H$	P	SV, E, P	Kernel Reset	4898
PSI5_TSRA	Timestamp Register A	070_H	P	SV, E, P	Kernel Reset	4899
PSI5_TSRB	Timestamp Register B	074_H	P	SV, E, P	Kernel Reset	4901
PSI5_TSRC	Timestamp Register C	078_H	P	SV, E, P	Kernel Reset	4902
PSI5_GCR	Global Control Register	080_H	P	SV, E, P	Kernel Reset	4903
PSI5_IOCRx (x=0-1)	Input and Output Control Register x	084_H+x*90_H	P	SV, E, P	Kernel Reset	4904
PSI5_RCRAx (x=0-1)	Receiver Control Register A x	088_H+x*90_H	P	SV, E, P	Kernel Reset	4906
PSI5_RCRBx (x=0-1)	Receiver Control Register B x	$08C_H$+x*90_H	P	SV, E, P	Kernel Reset	4907
PSI5_RCRCx (x=0-1)	Receiver Control Register C x	090_H+x*90_H	P	SV, E, P	Kernel Reset	4908
PSI5_WDTxw (w=0-6;x=0-1)	Watchdog Timer Register xw	094_H+x*90_H+w*4	P	SV, E, P	Kernel Reset	4909
PSI5_RSRx (x=0-1)	Receive Status Register x	$0B0_H$+x*90_H	P	BE	Kernel Reset	4910
PSI5_SDSxz (x=0-1;z=0-5)	Serial Data and Status Register xz	$0B4_H$+x*90_H+z*4	P	BE	Kernel Reset	4910
PSI5_SPTSCx (x=0-1)	Start of Pulse Timestamp Capture Register x	$0CC_H$+x*90_H	P	BE	Kernel Reset	4911

(table continues...)

Table 1190 (continued) Registers overview - PSI5 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PSI5_SFTSCx (x=0-1)	Start of Frame Timestamp Capture Register x	$0D0_H$+x*90_H	P	BE	Kernel Reset	4912
PSI5_RDRLx (x=0-1)	Receive Data Register Low x	$0D4_H$+x*90_H	P	BE	Kernel Reset	4912
PSI5_RDRHx (x=0-1)	Receive Data Register High x	$0D8_H$+x*90_H	P	BE	Kernel Reset	4913
PSI5_PGCx (x=0-1)	Pulse Generation Control Register x	$0DC_H$+x*90_H	P	SV, E, P	Kernel Reset	4915
PSI5_CTVx (x=0-1)	Channel Trigger Value Register x	$0E0_H$+x*90_H	P	SV, E, P	Kernel Reset	4917
PSI5_SCRx (x=0-1)	Send Control Register x	$0E4_H$+x*90_H	P	SV, E, P	Kernel Reset	4917
PSI5_SDRLx (x=0-1)	Send Data Register Low x	$0E8_H$+x*90_H	P	P	Kernel Reset	4920
PSI5_SDRHx (x=0-1)	Send Data Register High x	$0EC_H$+x*90_H	P	P	Kernel Reset	4921
PSI5_SSRLx (x=0-1)	Send Shift Register Low x	$0F0_H$+x*90_H	P	P	Kernel Reset	4921
PSI5_SSRHx (x=0-1)	Send Shift Register High x	$0F4_H$+x*90_H	P	P	Kernel Reset	4922
PSI5_SORLx (x=0-1)	Send Output Register Low x	$0F8_H$+x*90_H	P	P	Kernel Reset	4922
PSI5_SORHx (x=0-1)	Send Output Register High x	$0FC_H$+x*90_H	P	P	Kernel Reset	4923
PSI5_INTOV	Interrupt Overview Register	$2C4_H$	P	BE	Kernel Reset	4923
PSI5_INPx (x=0-1)	Interrupt Node Pointer Register x	$2C8_H$+x*4	P	SV, E, P	Kernel Reset	4925
PSI5_INTSTATAx (x=0-1)	Interrupt Status Register A x	$2DC_H$+x*4	P	BE	Kernel Reset	4927
PSI5_INTSTATBx (x=0-1)	Interrupt Status Register B x	$2EC_H$+x*4	P	BE	Kernel Reset	4931
PSI5_INTSETAx (x=0-1)	Interrupt Set Register A x	$2FC_H$+x*4	P	SV, E, P	Kernel Reset	4933
PSI5_INTSETBx (x=0-1)	Interrupt Set Register B x	$30C_H$+x*4	P	SV, E, P	Kernel Reset	4935
PSI5_INTCLRAx (x=0-1)	Interrupt Clear Register A x	$31C_H$+x*4	P	P	Kernel Reset	4935

(table continues...)

Table 1190 (continued) Registers overview - PSI5 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PSI5_INTCLRBx (x=0-1)	Interrupt Clear Register B x	$32C_H$+x*4	P	P	Kernel Reset	4938
PSI5_INTENAx (x=0-1)	Interrupt Enable Register A x	$33C_H$+x*4	P	SV, E, P	Kernel Reset	4938
PSI5_INTENBx (x=0-1)	Interrupt Enable Register B x	$34C_H$+x*4	P	SV, E, P	Kernel Reset	4940
PSI5_RFCx (x=0-1)	Receive FIFO Control Register x	378_H+x*4	P	P	Kernel Reset	4941
PSI5_RDFx (x=0-1)	Receive Data FIFO x	388_H+x*4	P	BE	Kernel Reset	4942
PSI5_RSIOVx (x=0-1)	RSI Overview Register x	398_H+x*4	P	BE	Kernel Reset	4943
PSI5_RMIOVx (x=0-1)	RMI Overview Register x	$3A8_H$+x*4	P	BE	Kernel Reset	4943
PSI5_NBIOVx (x=0-1)	NBI Overview Register x	$3B8_H$+x*4	P	BE	Kernel Reset	4944
PSI5_TEIOVx (x=0-1)	TEI Overview Register x	$3C8_H$+x*4	P	BE	Kernel Reset	4944
PSI5_CRCIOVx (x=0-1)	CRCI Overview Register x	$3D8_H$+x*4	P	BE	Kernel Reset	4945
PSI5_RDIOVx (x=0-1)	RDI Overview Register x	$3E8_H$+x*4	P	BE	Kernel Reset	4945
PSI5_NFIOVx (x=0-1)	NFI Overview Register x	$3F8_H$+x*4	P	BE	Kernel Reset	4946
PSI5_MEIOVx (x=0-1)	MEI Overview Register x	408_H+x*4	P	BE	Kernel Reset	4946
PSI5_RSISETx (x=0-1)	RSI Overview Set Register x	418_H+x*4	P	P	Kernel Reset	4947
PSI5_RMISETx (x=0-1)	RMI Overview Set Register x	428_H+x*4	P	P	Kernel Reset	4947
PSI5_NBISETx (x=0-1)	NBI Overview Set Register x	438_H+x*4	P	P	Kernel Reset	4948
PSI5_TEISETx (x=0-1)	TEI Overview Set Register x	448_H+x*4	P	P	Kernel Reset	4948
PSI5_CRCISETx (x=0-1)	CRCI Overview Set Register x	458_H+x*4	P	P	Kernel Reset	4949
PSI5_RDISETx (x=0-1)	RDI Overview Set Register x	468_H+x*4	P	P	Kernel Reset	4949

(table continues...)

Table 1190 (continued) Registers overview - PSI5 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PSI5_NFISETx (x=0-1)	NFI Overview Set Register x	478_H+x*4	P	P	Kernel Reset	4950
PSI5_MEISETx (x=0-1)	MEI Overview Set Register x	488_H+x*4	P	P	Kernel Reset	4950
PSI5_RSICLRx (x=0-1)	RSI Overview Clear Register x	498_H+x*4	P	P	Kernel Reset	4951
PSI5_RMICLRx (x=0-1)	RMI Overview Clear Register x	$4A8_H$+x*4	P	P	Kernel Reset	4951
PSI5_NBICLRx (x=0-1)	NBI Overview Clear Register x	$4B8_H$+x*4	P	P	Kernel Reset	4952
PSI5_TEICLRx (x=0-1)	TEI Overview Clear Register x	$4C8_H$+x*4	P	P	Kernel Reset	4952
PSI5_CRCICLRx (x=0-1)	CRCI Overview Clear Register x	$4D8_H$+x*4	P	P	Kernel Reset	4953
PSI5_RDICLRx (x=0-1)	RDI Overview Clear Register x	$4E8_H$+x*4	P	P	Kernel Reset	4953
PSI5_NFICLRx (x=0-1)	NFI Overview Clear Register x	$4F8_H$+x*4	P	P	Kernel Reset	4954
PSI5_MEICLRx (x=0-1)	MEI Overview Clear Register x	508_H+x*4	P	P	Kernel Reset	4954
PSI5_RDMLxy (x=0-1;y=0-31)	Receive Data Memory Low xy	600_H+x*100_H+y*8	P	BE	Kernel Reset	4955
PSI5_RDMHxy (x=0-1;y=0-31)	Receive Data Memory High xy	604_H+x*100_H+y*8	P	BE	Kernel Reset	4955

26.8.4.4 Device specific registers

There are no device specific register changes.

26.8.5 TC4Dx PSI5 connectivity

The table below lists interfaces to and from this functional block to other blocks in the device.

Table 1191 List of PSI5 interface signals

Interface signals	I/O	Description
CLOCK_PSI5_fSPB	In	FPI clock input
PSI5_IR_PSI5SRx[7:0]	Out	Interrupt trigger output
EGTM_PSI5_TRIG[5:0]	In	eGTM trigger
PORTS_PSI5_PSI5RX[1:0]	In	Receive data per PSI5 module
PSI5_PORTS_PSI5TX[1:0]	Out	Transmit data per PSI5 module

26.8.6 TC4Dx PSI5 revision history

Initial release of the chapter.

27 Peripheral Sensor Interface with Serial Interface to PHY (PSI5-S)

The PSI5-S module performs communication according to the PSI5 specification. It communicates with the external world using one ASC interface for all channels. This ASC interface is built into the PSI5-S module. If the optional bidirectional mode is used, the commands to the external Physical Layer Interface (PHY) are transmitted through this interface.

27.1 Feature list

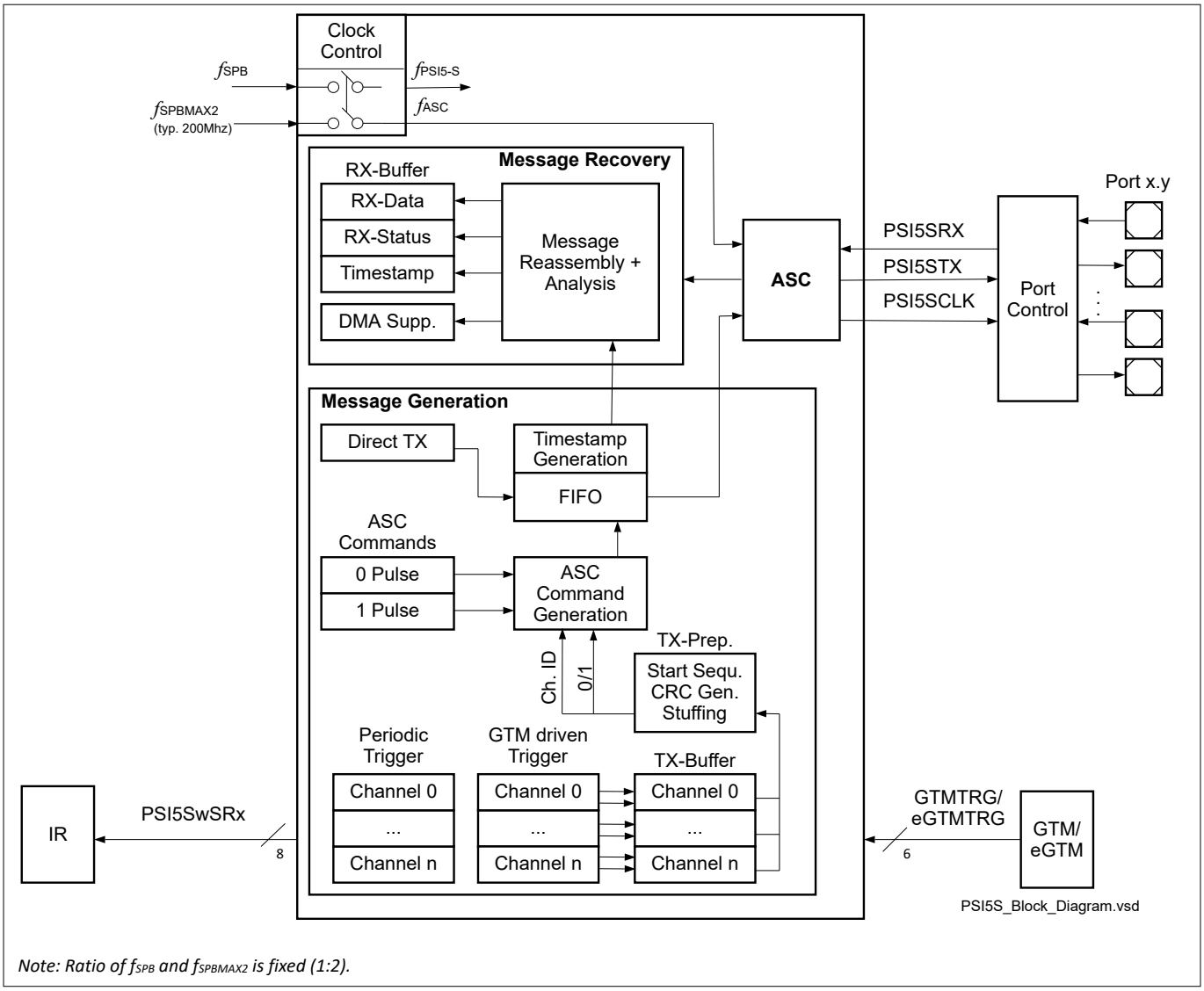
- According to PSI5 protocol specification v2.1
- Flexible configuration and monitoring using software
  - Asynchronous/synchronous serial communication (ASC interface) with compatible PSI5-S physical-layer (PHY)
  - Baud rate generator up to 12,5 MBaud at 200 MHz ASC sub-module clock
  - Clock out generator to supply external PHY
  - Data rates of 125 kbit/s and 189 kbit/s supported
  - 8 PSI5-S channels sharing one common ASC interface (channel 0 frame 1 special use)
  - Asynchronous and Synchronous PSI5 data transmission modes
  - Supports 6 sensor slots per channel
  - 8 interrupts or direct memory access (DMA) triggers
- Receive data processing using Message Recovery block:
  - 10-bit ASC format: 1 Start bit, 8 Data bits, 1 Stop bit (upstream)
  - Each PSI5 frame is transported in a packet frame consisting of 3 to 6 UART frames transmitted back to back with exactly one Stop bit. There is no additional delay
  - Packet frames are separated by a programmable idle time (with 1 to 16 idle bits)
  - Assembly and decoding of packet frames (envelope)
  - Routing of PSI5 frames to their referring channel receive buffer in system memory by hardware DMA support
  - Checking CRC of packet frames (XCRC)
  - Frames with XCRC error will be routed to channel 0 frame 1
  - The external PHY sends all frames of an asynchronous channel with frame ID (FID) = 1. All frames of this channel have the same length and structure as defined for FID 1. Therefore it is sufficient to configure frame 1 for asynchronous channels
  - Configurable data length from 8 to 28 bits and 3-bit PSI5 CRC or 1-bit PSI5 parity
  - CRC check of received PSI5 sensor data, CRC code still transparent
  - No hardware support of extended serial data messaging according to SENT SAE J2716 JAN 2010. Messaging bits are transported fully transparent for software decoding
  - 24-bit timestamp on sync pulse request through ASC (resolution: 1 µs)
  - Two independent time bases for timestamp: clocked by GTM or eGTM (1 out of n GTM or eGTM signals is selectable) or internal periodic trigger generator
  - PHY answers to CPU commands with standard frames of fixed predefined length in channel 0 frame 0
  - Error bits in packet frame are presented fully transparent and an interrupt is issued
  - One interrupt for error free packet frame. Relevant errors are selectable
  - One interrupt for packet frame reception disregarding any error
- Transmit message data preparation using Message Generation block
  - Support of ECU to sensor communication
  - One 32-bit send data register per channel for downstream data input
  - Data is shifted out LSB first. MSB is filled with ‘0’ or ‘1’ with each shift, depending on the bit coding method: Tooth Gap or Pulse Width

- 11-bit ASC format: 1 Start bit, 8 Data bits, 1 Parity, 1 Stop bit (downstream)
- Downstream commands of fixed format (3 MSB = Channel ID. 5 LSB = Command)
- All commands are randomly programmable and are therefore not interpreted by PSI5-S
- Downstream data transmission by 2 different ASC commands (short/no, and long pulse)
- 8 common lines of FIFO for all channels, writable by:
  - PSI5 message generation
  - Sync pulse generation. Pre-programmed bytes for 0 and 1
  - CPU commands:
    - CPU direct write registers (CDW) allow for insertion of commands into the FIFO for configuration and debugging
- 8 GTM or eGTM inputs for sync pulse triggering
- Software configurable staggering of sync pulse generation
- Generation of 3 or 6-bit CRC for downstream data
- Start sequence generator for downstream data. This can also be switched off
- Bit stuffing generator for downstream data. This can also be switched off
- CRC generator for downstream data. This can also be switched off
- Internal Loop-Back mode for a check of the CRC generator and for software development

27.2 Functional overview

The Peripheral Sensor Interface is an interface for automotive sensor applications. PSI5 is an open standard based on existing sensor interfaces for peripheral airbag sensors, already proven in millions of airbag systems.

The following block diagram shows a global view of the PSI5-S interface.



Note: Ratio of f_{SPB} and $f_{SPBMAX2}$ is fixed (1:2).

Figure 536 PSI5-S block diagram

PSI5 defines a current loop based serial communication link typically used to connect airbag sensors or other peripheral devices. PSI5 provides high speed and high reliability data transfer. The PSI5-S acts as a slave on the SPB bus. The PSI5-S kernel is composed of the clock control block that is driven by the system clock frequency f_{SPB}.

The PSI5-S consists of three main blocks that are shared between 8 channels :

- Message Generation block that is used for preparing the transmit data
- Message Recovery block that processes the received messages
- Common ASC module

While the physical layer is done externally, this module manages communication with this PHY and the presentation of the data to the application. Note that there is no on chip PHY, therefore the current to voltage and voltage to sync pulse translation is done externally.

Receive data on a PSI5-S channel can be set up according to the underlying application. In particular the number of bits is configurable.

The message storage consists of a set of 3 32-bit registers for all channels: one receive data register containing the received sensor data, one timestamp register containing the 24-bit value and one additional register containing status bits.

For Synchronous Communication mode, the module can consume sync pulses from the external GTM/eGTM or generate periodic sync pulses by itself. It supports as well ECU to sensor communication by offering a 32-bit

register, where messages can be set up by the CPU. All kinds of sync pulses are translated into a referring, programmable byte to be sent through ASC.

The register set of the PSI5-S module can be accessed directly by the CPU for configuration, data read out and status query.

Related information

TC4Dx SMU alarm mapping tables on page 7233

27.3 Functional description

The following section explains the different PSI5-S modes of operation and frame formats

27.3.1 PSI5 ECU to sensor operation

ECU to sensor communication is defined in the PSI5 protocol specification v2.1. Data transmission and configuration of the sensor is made by modulation of the sync pulses.

Two modes are supported:

- Tooth Gap method
  - With the Tooth Gap method, a specific start condition and a start sequence switch the sensor in ECU to sensor communication mode. In this mode, the sync pulses are no longer used as triggers for PSI5 frames but are interpreted as data for the sensor. A pulse represents a 1 and a missing pulse represents a 0. This requires isochronous pulses.
- Pulse Width method
  - With the Pulse Width method a standard length pulse represents a 0, and a longer pulse a 1. In this way, continuous data transmission can be achieved. The pulses can be non-isochronous. They can for example, be dependent from the angle position of a rotating axis.

Related information

References on page 5071

27.3.2 Frame formats and definitions

This section describes the frame formats and definitions of the PSI5-S protocol.

27.3.2.1 Communication between PSI5-S and PHY through UART

This chapter covers the communication on the UART connection.

27.3.2.1.1 Packet frames received from PHY

Packet frames are 3 to 6 UART frames carrying one PSI5 frame. The format accepted by PSI5-S is explained here.

A UART frame is one transmission unit over the ASC interface. On the receive path it consists of one Start bit, 8 Data bits, and one Stop bit. Parity can be optionally chosen, for loop-back tests for example.

The figures show the minimum and maximum packet frame population with data. Where the figures show ‘Res0’ bits, these bits can be crowded by more data bits if configured.

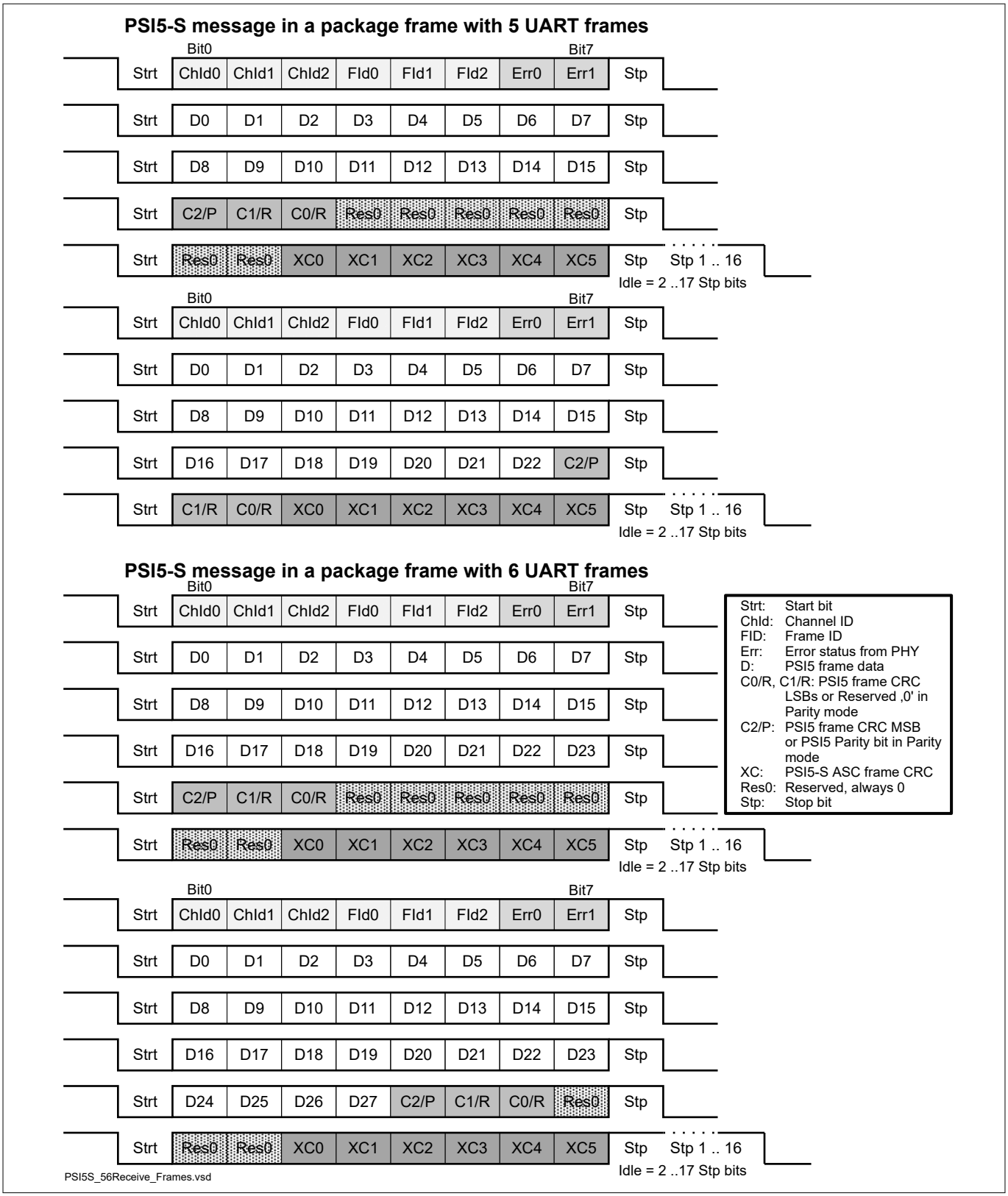


Figure 537 PSI5-S 5 and 6 UART frames per package frame received from PHY

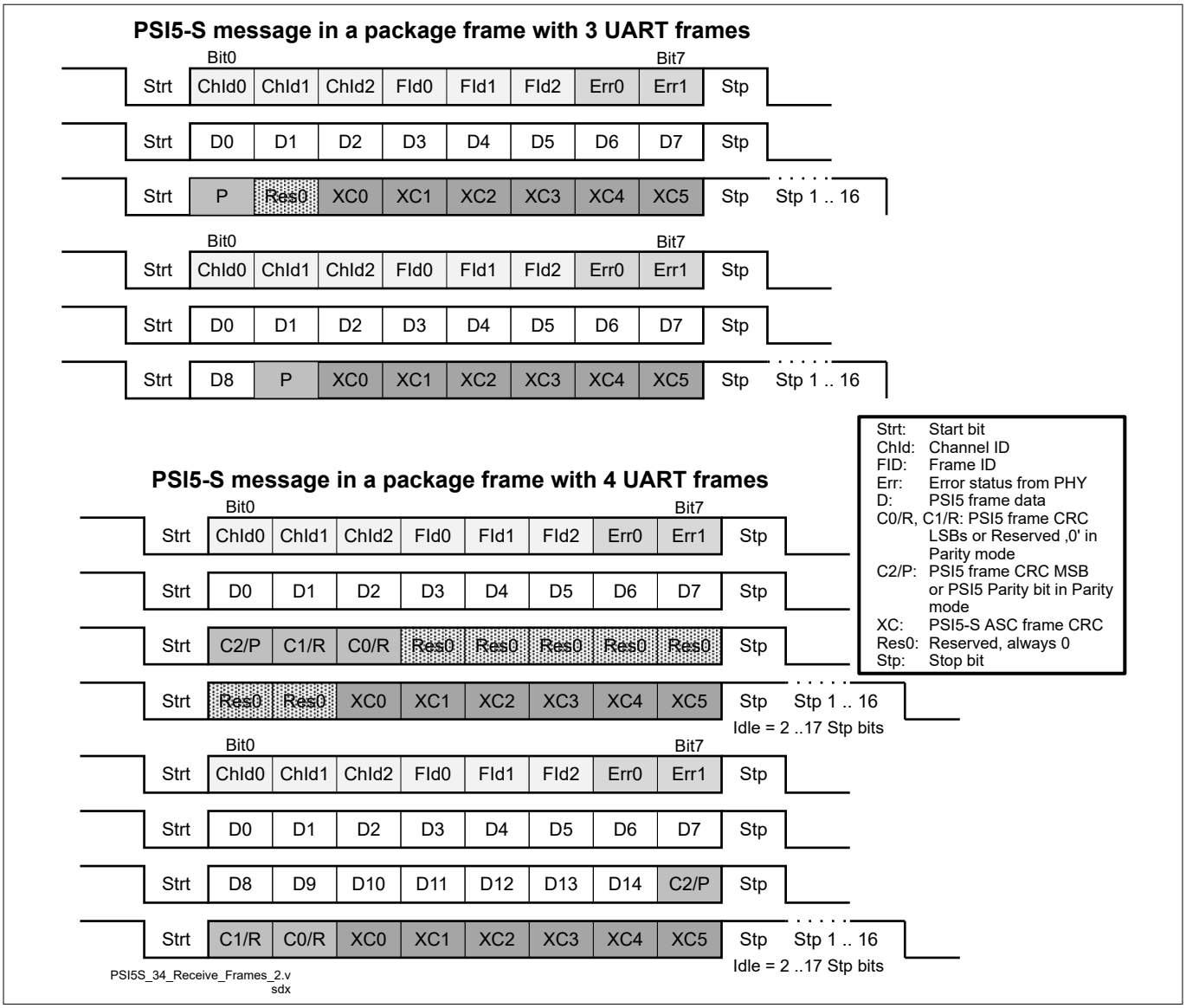


Figure 538 PSI5-S 3 and 4 UART frames per package frame received from PHY

Detection of frame boundaries and packet frame processing

- SOF (start of frame) is defined by a Start bit detected after idle on ASC interface
- Idle equals at least two Stop bits:
  - The required idle time can be configured between 1 and 16 additional Stop bits
  - An idle state signal is provided internally by the ASC module to the reassembly unit
  - The idle signal starts on detection of the first Stop bit and ends when the first slope of the next Start bit has passed the two out of three majority vote input filter
- If idle is detected before the minimum of 3 UART frames were received, these UART frames are copied to channel 0 frame 1 (non-recoverable frames)
- Based on the RDS.ChID and RDS.FID received in the first UART frame, the module checks:
  - If the referring channel is enabled
  - If the FID is valid and configured (PDLy > 0)
  - The expected number of UART frames configured in RCRAx.UFCy
  - If at least this number of UART frames is received without idle
  - If the XCRC is correct

- If these checks pass:
  - EOF is detected after the correct number of UART frames
  - Data is copied to RDR, interrupts and the status are updated in RDS. In particular CRC/parity check is executed. Timestamp (TSM) and target address (TAR) are updated. Depending from bit RCRAx.FIDS, the FID might be manipulated
  - The message recovery unit does not wait for idle to execute the steps above
- If these checks fail (non-recoverable frames):
  - EOF is detected after idle or a maximum of 6 UART frames
  - The 1 to 6 UART frames are stored in channel 0 frame 1
- If more UART frames follow a passing or failing packet frame without idle:
  - EOF is detected after idle or a maximum of 6 UART frames
  - The 1 to 6 UART frames are stored in channel 0 frame 1
  - Reception is stopped
  - The next UART frames following without idle time are ignored
  - Reception starts only after the idle time programmed in GCR.IDT is detected (so-called babbling-idiot protection)

Non-recoverable frame presentation in RDR and RDS

- RDS.FID and ChID are derived from the first UART frame received
- RDR.RD[27:0] contains all bits of each UART frame received after the header. That is:
  - 0 bits for 1 UART frame
  - 8 bits for 2 UART frames
  - 16 bits for 3 UART frames
  - 24 bits for 4 UART frames
  - 28 bits for 5 and 6 UART frames
- RDS.CRC is determined as follows:
  - If 6 UART frames were received: bits [6:4] of the 5th UART frame
  - If 1 to 5 UART frames where received: determined by RCRBx.PDLy
  - RCRBx.PDLy may not be reliable, but this way all data is covered either in the data field or in the XCRC field. In case RCRBx.PDLy was correct, CRC is shown correctly
- RDS.XCRC contains always the bits [7:2] of the last UART frame received. Also if only 1 UART frame was received
- Note that XCRCI is always set as either XCRC was wrong, or the number of UART frames received was wrong, or it could not be determined

Loop-back support

- Parity and number of Stop bits need to be configured the same for sender and receiver in Loop-Back mode. Parity control and polarity of parity can be configured separately each for TX and RX. This can be done before switching on Loop-Back mode. Alternatively, the parity control can be adjusted as well in one single register access together with switching the ASC part to Loop-Back mode
- Both ways are supported: Add parity bit to the receiver, or use the transmitter without parity. The standard use case is to use transmitter with parity
- Data to be transmitted can be written directly to the CDW register
- In this way a synthetic Packet frame can be sent to the reassembly unit for test purposes
- The received loop-back data is treated in the same way as the normal receive data. Good frames are received in their referring channel and slot. Frames with the wrong XCRC, idle time, or RCRAx.UFCy, are stored in channel 0 frame 1

XCRC content

XCRC is always sent at the end of the last UART Packet. For XCRC calculation, all bits of the preceding UART frames that belong to the packet frame are relevant, including potential stuffing bits (Res0). UART Start or Stop bits are excluded.

In detail the XCRC shall contain the header (ChID, FID, Err0, Err1), PSI5 frame data (D0 … Dx, PSI5 frame parity bit or CRC bits) and potential stuffing bits (Res0) between P/CRC and XCRC. Not included are Start and Stop bits of the UART packets and the 2 Start bits (S0, S1) of the PSI5 frame.

The LSB of the first byte (ChId0) is fed first into the XCRC polynomial.

XCRC calculation method

XCRC equals the CRC defined in the PSI5 standard v2.1 “Frame format 4 ECU to Sensor Message CRC”.

Related information

ECU to sensor communication on page 4976

27.3.2.1.2 PSI5-S UART frames transmitted to PHY

The following figure shows the layout and definitions of the UART frames sent by PSI5-S.

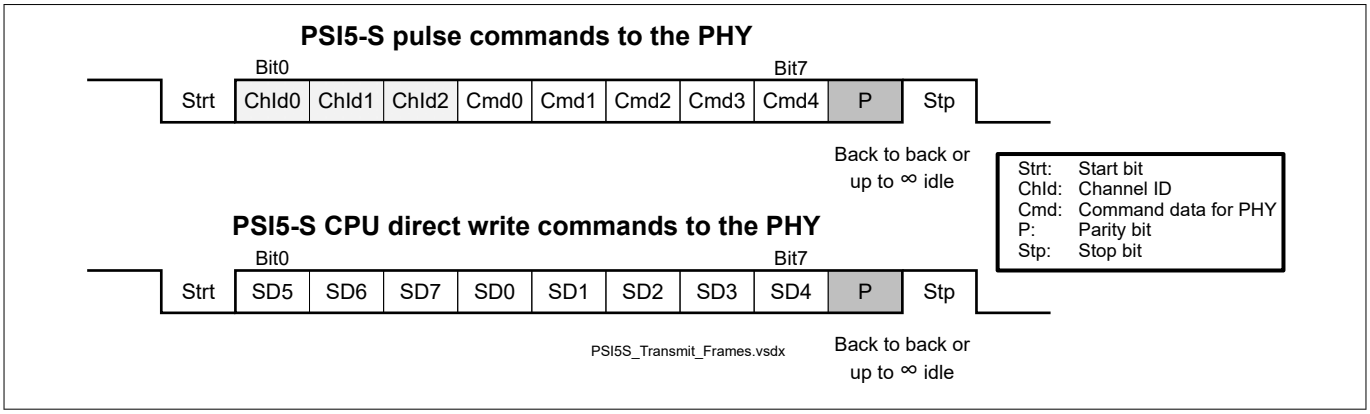


Figure 539 PSI5-S UART frames transmitted to the PHY

27.3.2.2 Communication between PHY and sensor (PSI5 standard)

This chapter treats the frame formats on the PSI5 bus as short reference.

27.3.2.2.1 PSI5 standard frame format

The following figure shows the principal layout and definitions of a PSI5 frame. Note that the PSI5 standard specifies that the least significant bit is sent out first. See standard for CRC and parity definition.

Example: 16 Bit Data Word „0xAD2C“ with 3-Bit CRC

S1	S2	D0	D1	D2	D3	D4	D5	D6	D7	D8	D9	D10	D11	D12	D13	D14	D15	C2	C1	C0
0	0	0	0	1	1	0	1	0	0	1	0	1	1	0	1	0	1	1	0	0
		C = „1100“				2 = „0010“				D = „1101“				A = „1010“						

Figure 540 Standard PSI5 frame format

27.3.2.2.2 PSI5 extended frame format

The following figure shows the layout and definitions of an Extended PSI5 Frame. Note that the PSI5 standard specifies that the least significant bit is sent out first.

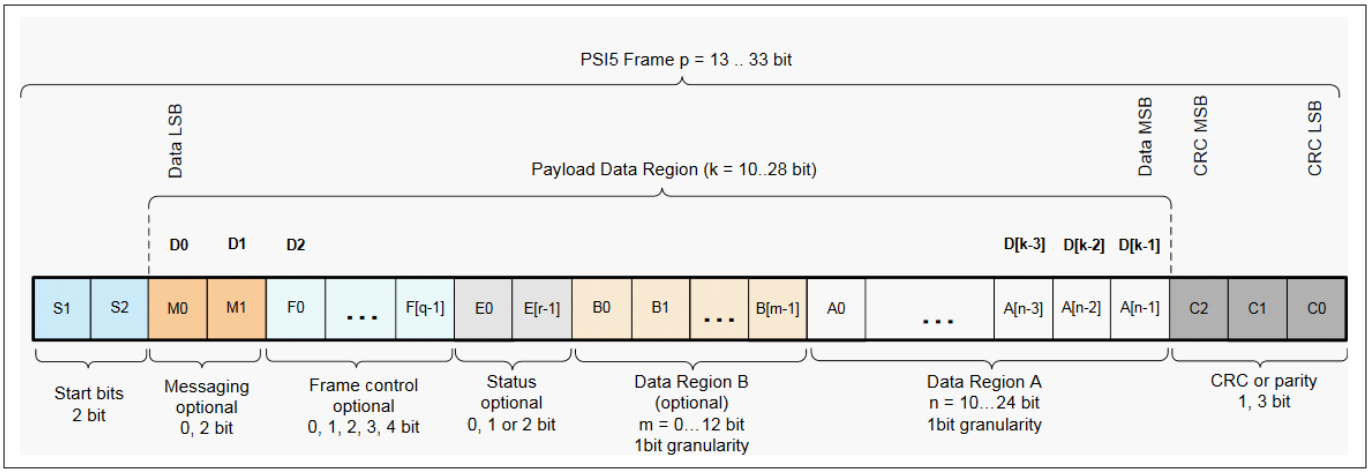


Figure 541 PSI5 extended frame format

PSI5 Sensor to ECU Message CRC

The transmission of PSI5 data is checked by the following protection modes. The transmission error detection must be selectable:

1. 1-bit even parity
2. 3-bit CRC

The applied generator polynomial of the CRC is $g(x) = 1 + x + x^3$ with a binary start value (seed) “111”. The transmitter shall extend the data bits by three zeros (as MSBs). This augmented data word shall be fed (LSB first) into the shift registers of the CRC check. Start bits shall be ignored in this check. When the last zero of the augmentation is pending on the input adder, the shift registers contain the CRC checksum. These three check bits shall be transmitted in reverse order (MSB first: C2, C1, C0).

27.3.2.2.3 Sync pulses

Sync pulses are used for two different purposes:

- Triggering a data frame for data acquisition from a sensor or
- ECU to sensor communication

Synchronous transmission

In the “Synchronous” mode, the sensor (slave) starts to transfer a complete data frame only after a sync pulse is transmitted through ASC to the external PHY. The external PHY translates this ASC command into a voltage increase. The sensor then initiates a measurement and starts to calculate the new output data value. The data follows in a standard PSI5 frame, starting with the 2 Start bits, data and parity or CRC. The timing diagram in the following figure visualizes a synchronous transmission

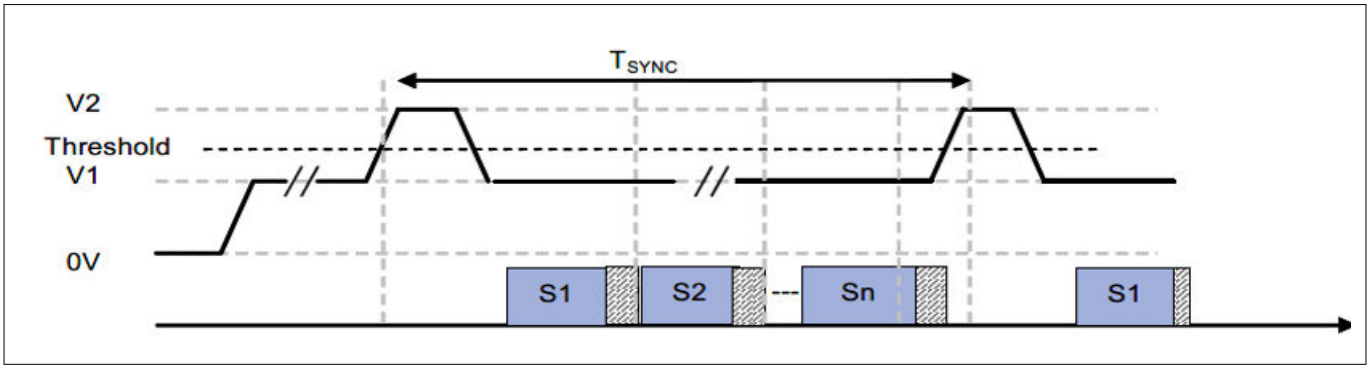


Figure 542 Synchronous transmission

ECU to sensor communication

PSI5 v2.1 defines 4 principle frame formats. Please refer to this standard document for more details. As the main target application is powertrain, only frame format 4 is cited here. This is the format for the powertrain substandard.

Two methods for the ECU to sensor communication are supported:

- Tooth Gap method
- Pulse Width method

Tooth gap method

Tooth gap method uses usually 31 sync pulses as start condition and start bits (0 1 0) to signal the start of the ECU to sensor communication. From this start on, the sync pulses are interpreted by the sensor as data. See standard for CRC and parity definition.

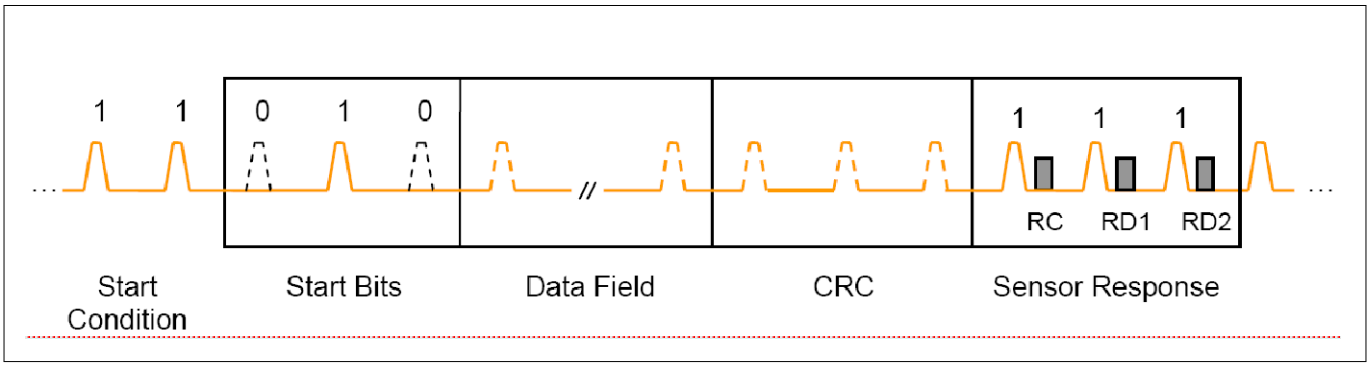


Figure 543 ECU to sensor (tooth gap method)

Pulse width method

Pulse width method allows for modulation of the sync pulse duration. This way bidirectional communication is possible continuously.

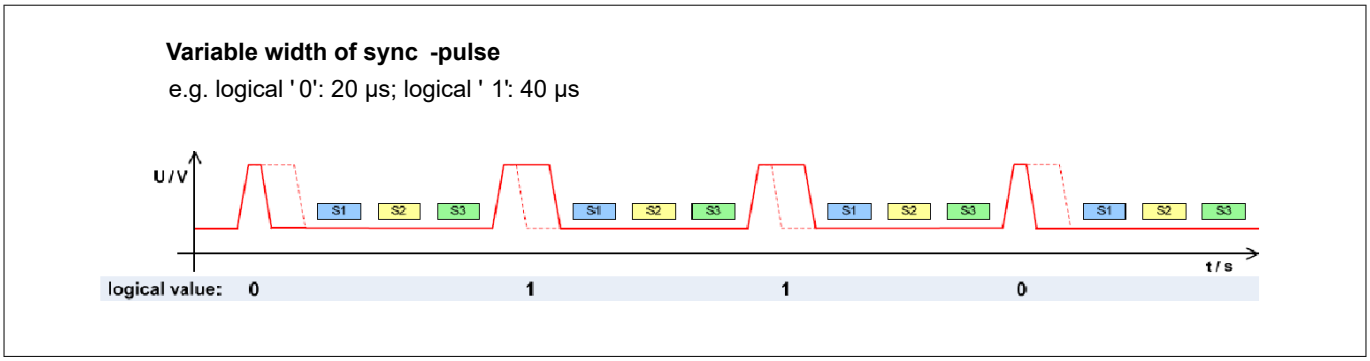


Figure 544 ECU to sensor, bit transmission (pulse width method)

The extended powertrain substandard frame format 4 "XXLong" follows the Pulse Width method.

Frame format 4 (power train)

Sync bits

Seven consecutive ones within the transmitted data could be misinterpreted as a frame start condition. Therefore a “sync bit” (logical 0) is inserted after every six bits of the sensor address and the payload data. With respect to the data transmission, these sync bits are treated like stuffing bits, which are removed at the receiver side.

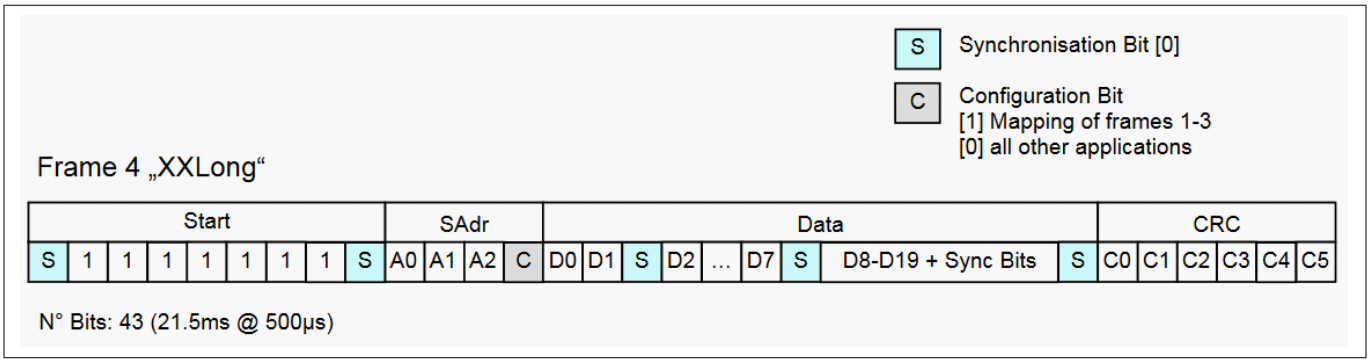


Figure 545 ECU to Sensor, frame format 4 (power train)

ECU to sensor message CRC

The ECU-to-Sensor frame has a 6-bit CRC. The generator polynomial of the CRC is $g(x) = x^6 + x^4 + x^3 + 1$ with a binary CRC initialization value “010101”.

The transmitter extends the data bits by six zeros (as MSBs). This augmented data word shall be fed (LSB first) into the shift registers of the CRC check. Start bits and sync bits are ignored in this check. When the last zero of the augmentation is pending on the input adder, the shift registers contain the CRC checksum. These six check bits shall be transmitted LSB first [C0, C1 … C5]. This CRC value is computed as a function of the contents of all Address and Data bits and sync bits (“0” bits). For purposes of the CRC calculation, the bits shall be ordered: m = [A0 A1 … A3 D0 D1 0 D2 … D19 0]. In this implementation the number of elements is configurable. While the standard is 4 Address and 20 Data bits, the module allows longer or shorter messages too.

27.3.3 Clock generation

The PSI5-S module provides 2 Timestamp Counters (TSCNTA and TSCNTB) and the timestamp clock f_{TS} to each channel as shown in the following figure. The clock of each timestamp counter is individually selectable: either f_{TS} or any of the GTM/eGTM inputs can be selected to drive them.

The PSI5-S module provides 2 internal clock signals

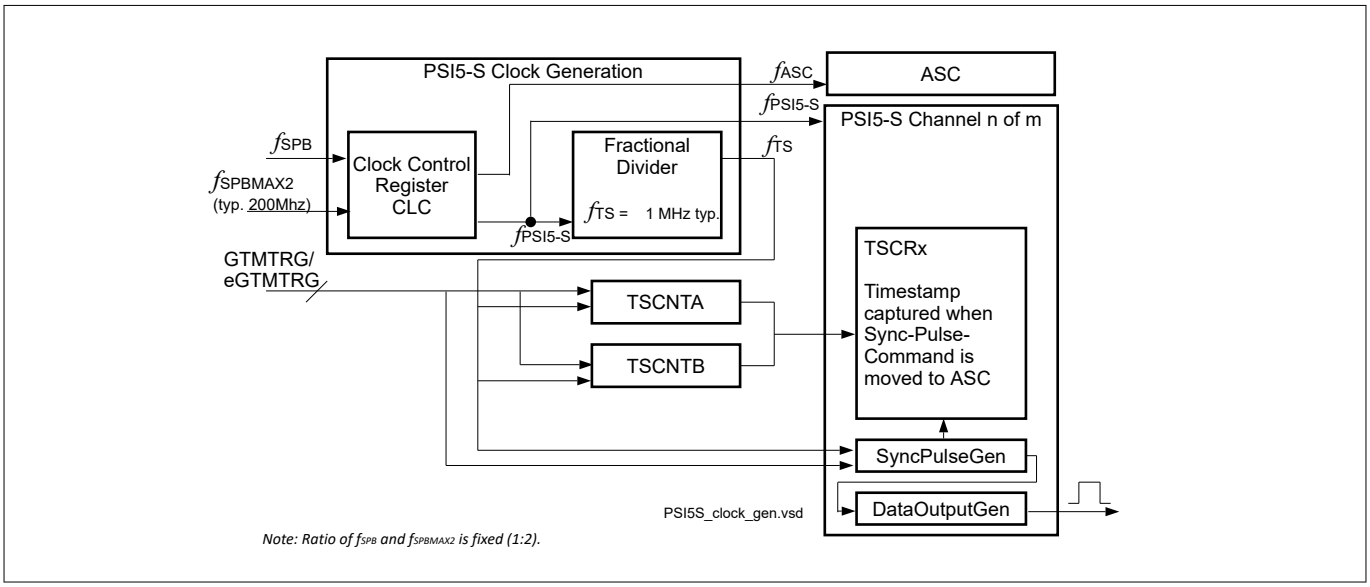


Figure 546 PSI5-S module clock generation

- f_{PSI5-S}
  This is the module clock that is used inside the PSI5-S kernel for control purposes such as clocking of control logic and register operations. The frequency of f_{PSI5-S} is controlled by register FDR and derived from the system clock frequency f_{SPB}. The Clock Control register CLC makes it possible to enable/disable f_{PSI5-S} under certain conditions.
  The following two formulas define the frequency of f_{PSI5-S}:

$$f_{PSI5-S} = f_{SPB} \div (1024 - FDR.STEP); FDR.DM = 01_B \quad (23)$$

$$f_{PSI5-S} = f_{SPB} \times FDR.STEP \div 1024 \text{ with } STEP = 0...1023; FDR.DM = 10_B \quad (24)$$

- f_{TS}
  This clock is usually used to drive the timestamp counters. It can be selected to drive the global sync pulse time base and inside the PSI5-S channels to drive the local sync pulse generators and watchdog timers. The Fractional Divider register FDRT controls the frequency of f_{TS}. This is usually adjusted to 1 MHz/1 µs period time.
  The following two formulas define the frequency of f_{TS}:

$$f_{TS} = f_{PSI5-S} \div (1024 - FDRT.STEP); FDRT.DM = 01_B \quad (25)$$

$$f_{TS} = f_{PSI5-S} \times FDRT.STEP \div 1024 \text{ with } STEP = 0...1023; FDRT.DM = 10_B \quad (26)$$

27.3.3.1 Overview on clocks in the system

To reduce system cost, the external PHY can be supplied with a clock from this module by PSI5SCLK pin. The range is limited to 20-25 MHz for proper clock transmission. The PHY needs to provide 3 clocks from this: sample clock for 189 kHz, 125 kHz and the ASC oversampling clock.

$f_{PSISCLK}$

This clock is used to drive the external PHY as shown in the following figure.

The Fractional Divider register FDO controls the frequency of $f_{PSISCLK}$.

The following formula defines the frequency of $f_{PSISCLK}$:

$$f_{PSISCLK} = f_{ASC} \div (2048 - FDO.STEP) \div 2; DM = 01_B \quad (27)$$

$$f_{PSISCLK} = f_{ASC} \times FDO.STEP \div 2048 \div 2 \text{ with } STEP = 0...2047; DM = 10_B \quad (28)$$

In order to relax the requirements with respect to ASC communication while allowing for high bit rates, some additional provisions have been made. The ASC submodule is supplied with a higher frequency (f_{ASC}) to allow for very high precision at high baud rates (4 - 12,5 MBaud). As a second measure to achieve this, the fractional divider in ASC submodule is longer than in the standard module. The following figure illustrates the situation. Refer to the ASC subchapter for baud rate calculations.

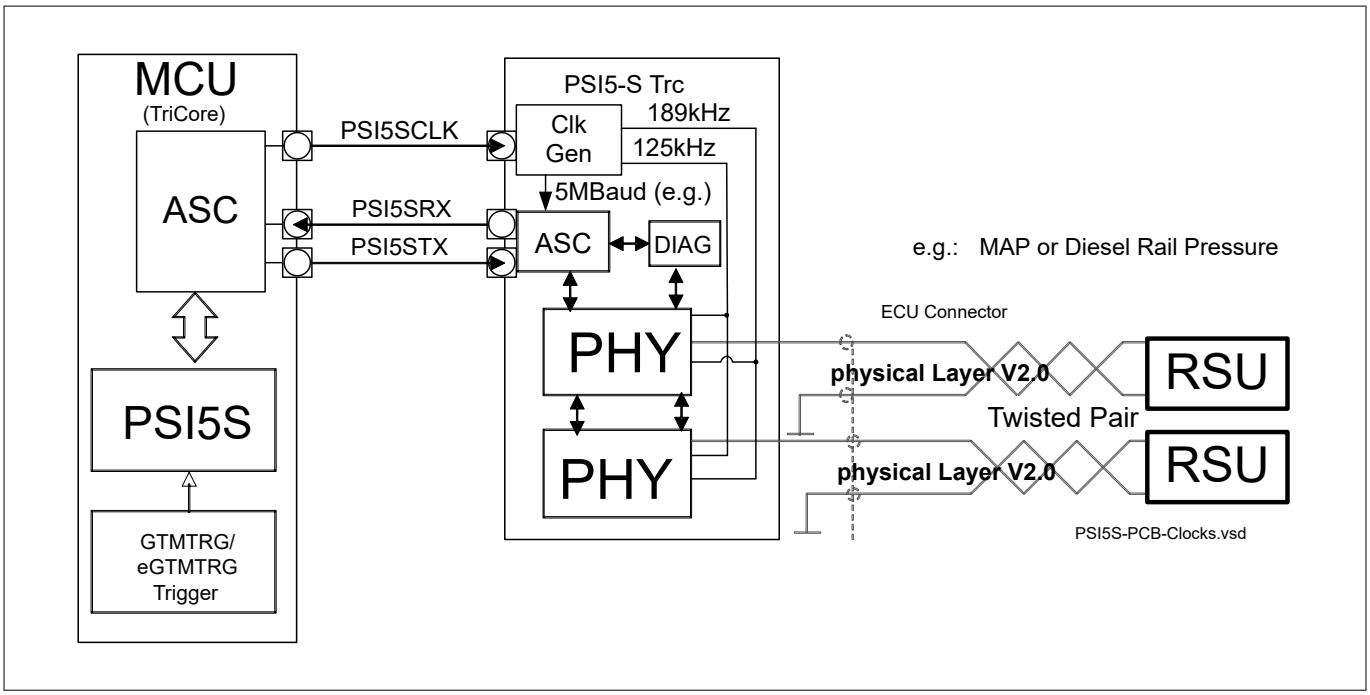


Figure 547 PSI5-S clock requirements of PHY

27.3.4 Timestamp generation

Two time bases can be selected for timestamping:

- Internal time base (Bit TSCNTA.TBS = 0 or TSCNTB.TBS = 0)
- External time base (Bit TSCNTA.TBS = 1 or TSCNTB.TBS = 1) GTM or eGTM
  - Bit TSCNTB.TRIGSELA and TSCNTB.TRIGSELB are used to select between GTM or eGTM for TSCNTA and TSCNTB respectively. Bit TSCNTA.ETB or TSCNTB.ETB are used to select the external clock line for counters CTS. Bit ECS, if enabled by FDRT.ECEA or ECEB, selects the external trigger line that clears the timestamp counters CTS

There are two Timestamp Counters TSCNTA.CTS and TSCNTB.CTS . Bits TSCNTA.TBEA and TSCNTA.TBEB start or stop the counters. Bit RCRAx.TSP is used to select between the two timestamp counters. Each of them can be

configured to be driven by either f_{TS} or by an external timer(GTM or eGTM). The frequency of f_{TS} is controllable by the fractional divider in the FDRT register.

All of the counters can be cleared by software synchronously.

The last sync pulse sent on a channel is stored in the sync pulse Timestamp Capture register TSCRx. Note that if RCRAx.TSTS is set, the sync pulses do not trigger the timestamp capture. In this case, the packet frame reception triggers the capture. Non-recoverable packet frames will be stored with the current value of TSCNTA or B copied directly to the TSM register. See RCRAx for more details.

For each channel the capture time (TSCRx) is stored.

The following figure shows the timestamp generation:

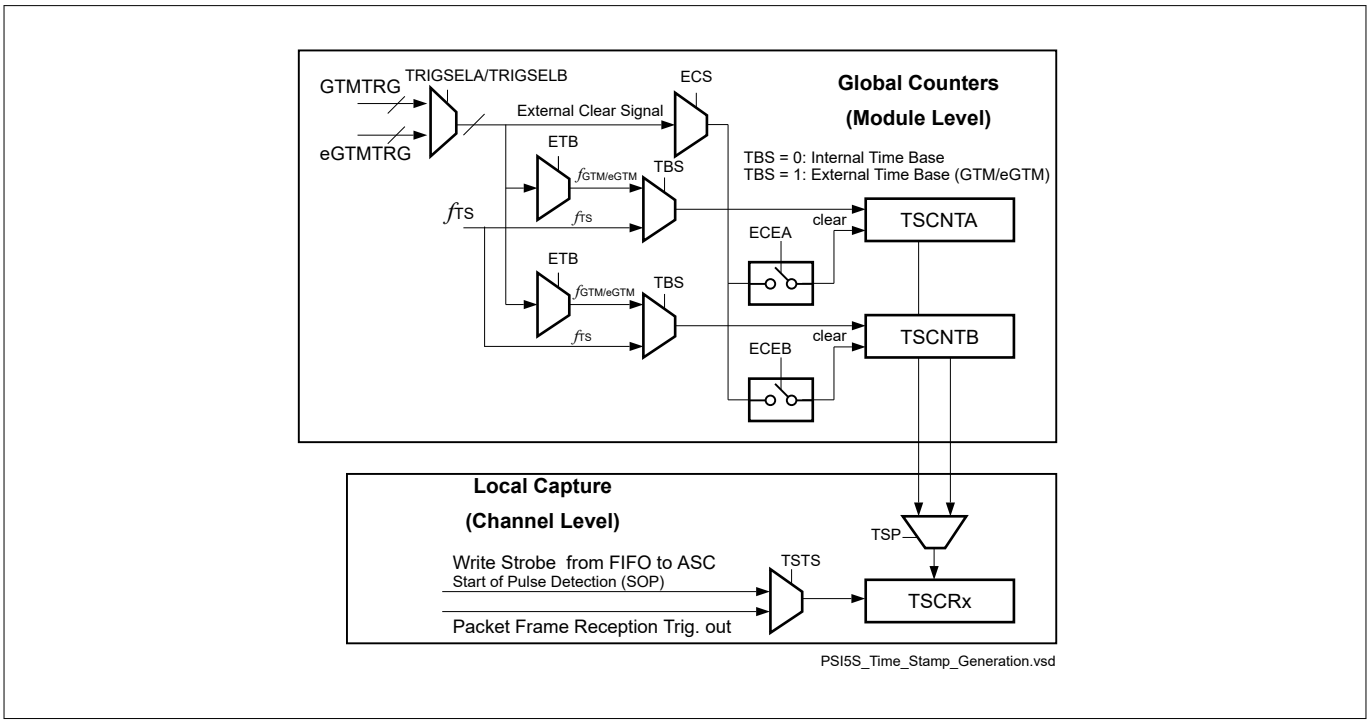


Figure 548 Timestamp generation

27.3.5 Watchdog timers

The value entered in WDTx.WDL defines the time in multiples of TTS (time resolution defined in FDRT) until expiry of the watchdog timer.

There are two modes of operation that can be selected with bit RCRAx.WDMS.

In Asynchronous mode an interrupt (TEI) is issued if the time between two RDIs is longer than specified in WDL. In this mode, the internal watchdog timer is started automatically when the channel is enabled.

In Synchronous mode, TEI is issued if the watchdog timer expires without reception of interrupt CHCI in time. Which means that the time from issuing the sync pulse to reception of the last expected frame configured in NFC.NFx was too long.

In both modes, only completely and correctly received packet frames with no XCRC error on channel x can qualify. Otherwise, the channel number cannot be determined as the referring WDTx.WDL will not be selected.

The internal watchdog timer is compared to WDTx.WDL. A match:

- Triggers a TEI
- Restarts the watchdog timer. It clears the internal watchdog timer and it counts on from zero. TEI is repeated if non-recoverable frames are received

If no watchdog timer is needed, WDL is cleared, the internal watchdog timer is stopped and no check is performed.

Asynchronous mode of the watchdog timers

If RCRAx.WDMS is cleared, the internal watchdog timer is restarted on reception of a new recoverable frame (RDIx is set) for the referring channel, as shown in the following figure. The frame is considered to be valid when the timing between two consecutive received frames (RDIx is set) is less than WDL. Otherwise, a TEI interrupt is triggered. It is not restarted by a sync pulse and not stopped by a certain number of received packet frames.

The example below shows the usage of the watchdog timer. The sync pulse is completely ignored by the watchdog timer system.

F2 is the only frame coming too late. All others start before the expiry of the watchdog.

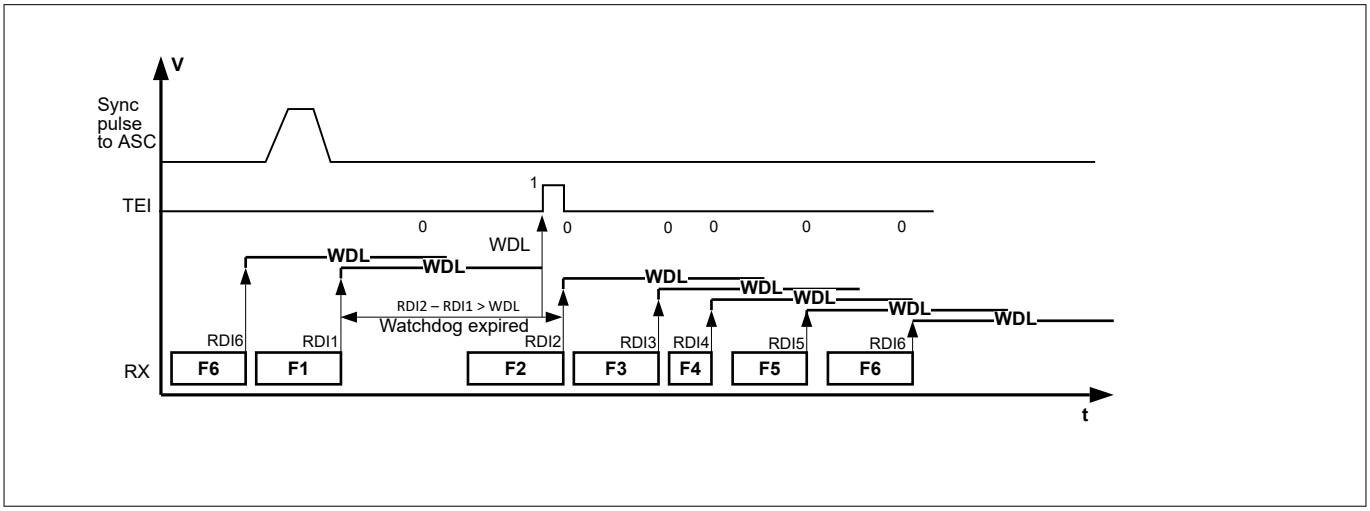


Figure 549 Watchdog timers in asynchronous mode

Synchronous mode of the watchdog timers

If RCRAx.WDMS is set, the watchdog timer is restarted on sync pulse and stopped at reception of the last frame configured in NFC.NFx. Register NFC contains the bit fields NFx which configure the number of packet frames expected after the sync pulse.

The watchdog timer is restarted (cleared and set to run), each time a sync pulse is transferred for the referring channel from FIFO to ASC. The watchdog timer of this channel is forced to stop if the number of a received packet frames (counted in FCNT.FCx) matches the number configured in NFC.NFx. This supports in particular angle synchronous channels where the distance of packets can vary strongly.

The watchdog timer cannot stop in time if a non-recoverable (and thus not counted) packet frame is received. If more than the configured number of packet frames are received, no dedicated interrupt is issued for this but the frames are still received correctly.

RCRAx.WDMS should be 0 on asynchronous channels. Otherwise, the internal watchdog timer will be stopped after reception of the number of configured frames but never be restarted due to the lack of a sync pulse.

If a sensor is not present in its configured slot, NFC.NFx must be reduced accordingly. For example, if out of F1-F6 the F2 is missing, only 5 frames are configured.

Please refer to register NFC on page 5033 for detailed description.

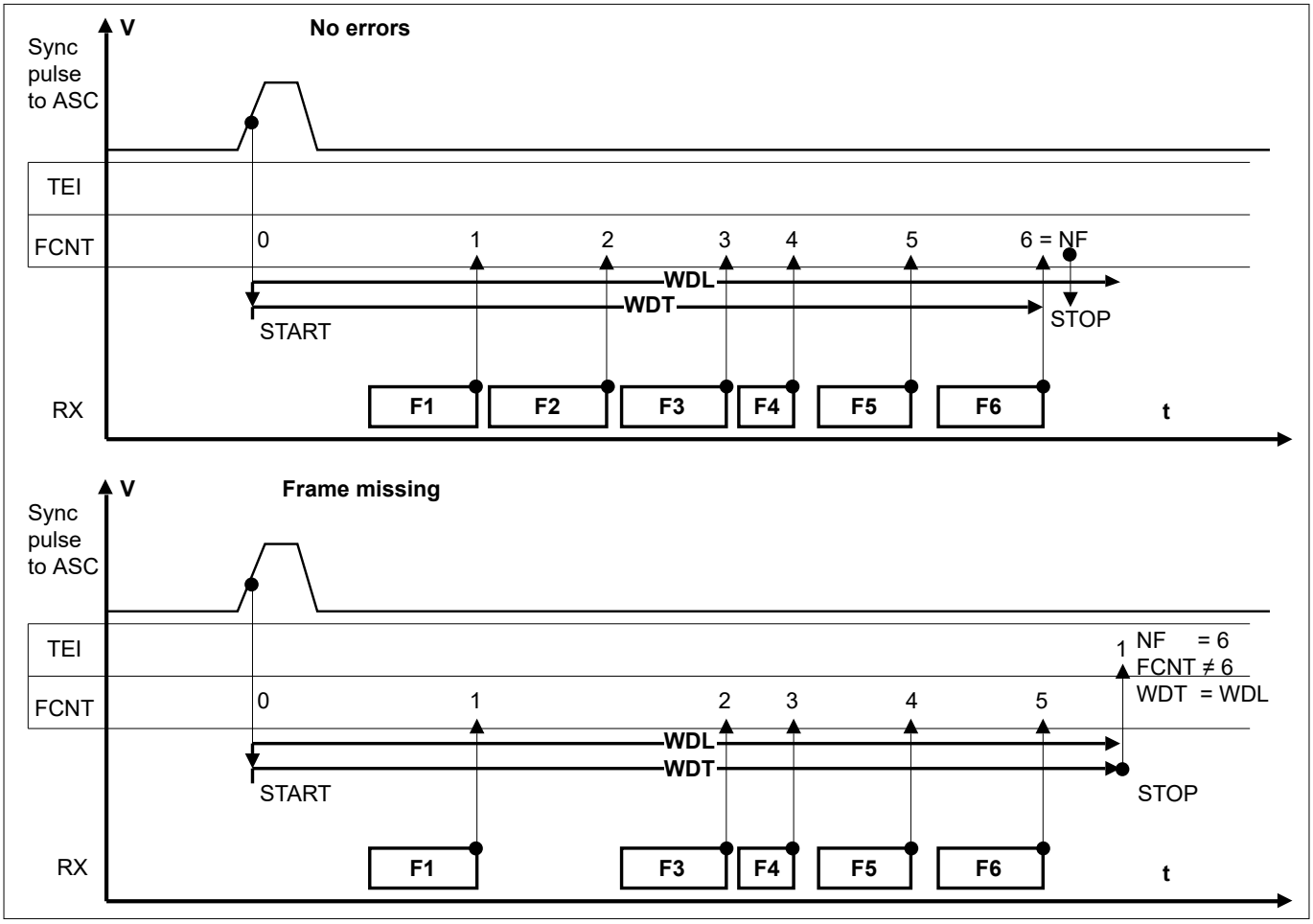


Figure 550 Watchdog timers in synchronous mode

27.3.6 Send data

This chapter contains the details on how data to be sent is prepared. The scope extends to:

- Periodic triggers
- Angle synchronous triggers
- ECU to sensor communication

27.3.6.1 Channel trigger

The channel trigger value register CTVx contains a reset counter. It allows for setting up periodic sync pulses for channel x.

The channel trigger value CTV is the compare value which restarts the counter CTC on equality. At exactly this time, a sync pulse is triggered.

The compare on equality allows to preset CTC and thus configure an offset between the CTVx.CTCs. This allows for staggering periodic sync pulses that must have frequencies with common multiples (for example same frequency, double etc.).

The PSI5-S sync pulse staggering is done based on the offset.

$$\text{Offset} = 65535 - \text{CTC preset value}$$

Which means, if the CTC preset is greater than CTV and the time to roll over is the offset. When the CTC preset is lesser than CTV, the CTC has some lead. This is shown in the following figure.

Note that CTC can be written only if it is disabled by clearing GCR.ETCx.

Staggering the pulses might be required if the current source for the external PHYs can provide only limited current so that not all channels can be provided with the additional current that is required to apply the voltage increase representing a proper sync pulse.

For a synchronization of all counters CTVx.CTC, bits GCR.ETCx can be used.

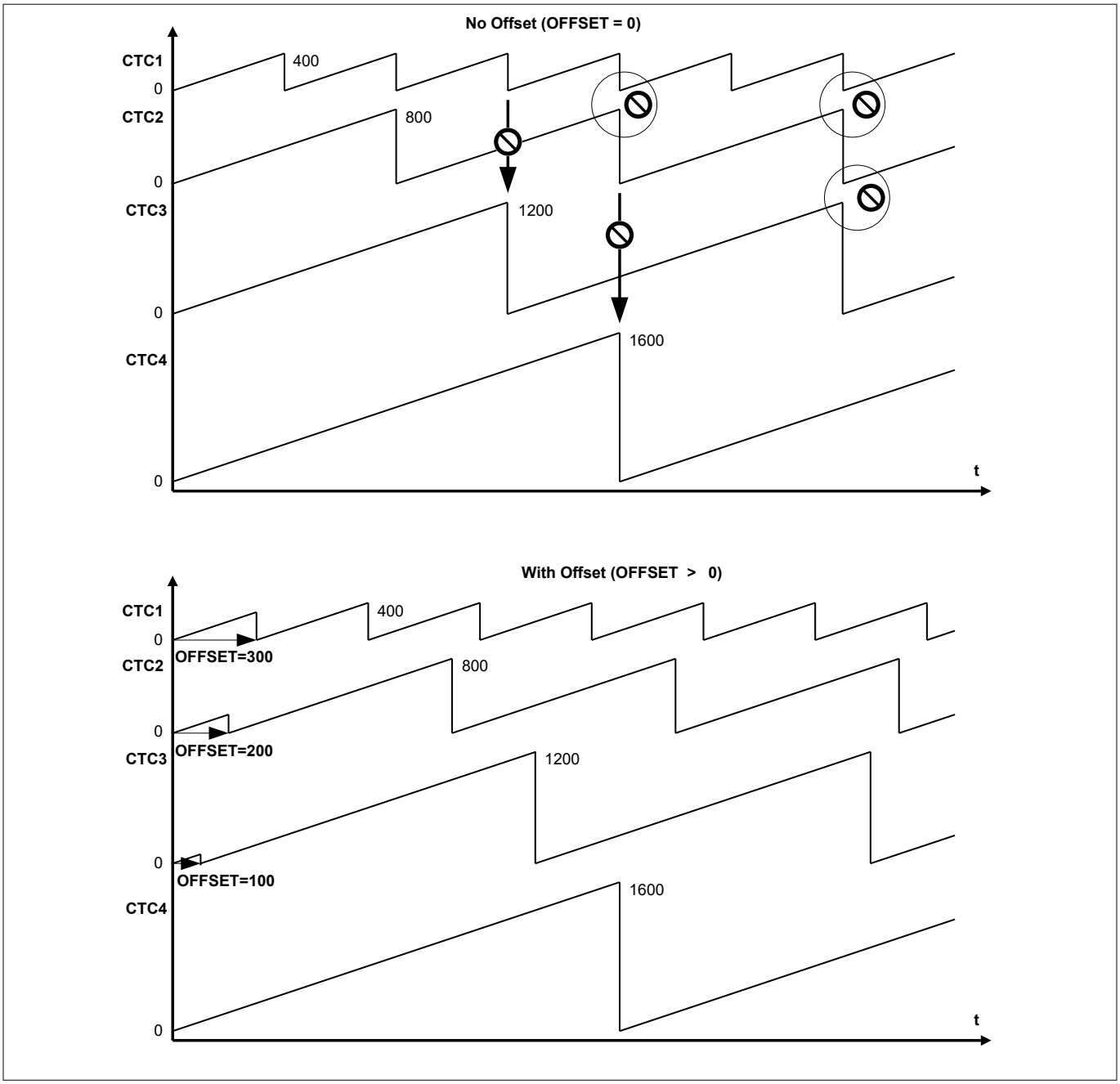


Figure 551 PSI5-S pulse staggering

27.3.6.2 Sync pulse control

The sync pulse generation provides the following functionality:

- Periodic sync pulses, driven by a common module time base (PGCx.PTE is set)
- Angle synchronous sync pulses, driven by GTM or eGTM based on bit FDRT.TRIGSEL (PGCx.ETE is set)
- Sync pulses for ECU to sensor communication (Tooth Gap method, SCRx.EPS[0]=0)
- Sync pulses for ECU to sensor communication (Pulse Width method, SCRx.EPS[0]=1)

It is possible to switch on and off the periodic trigger or the angle synchronous trigger at any time. The pulses will be modulated according to the selected standard. EPS selects between Pulse Width method and Tooth Gap method with pulse dropping.

An ECU to sensor communication can be stopped at any point in time by setting the flush bit SCRx.FLUS. This will clear the send register and flag SCRx.TPF.

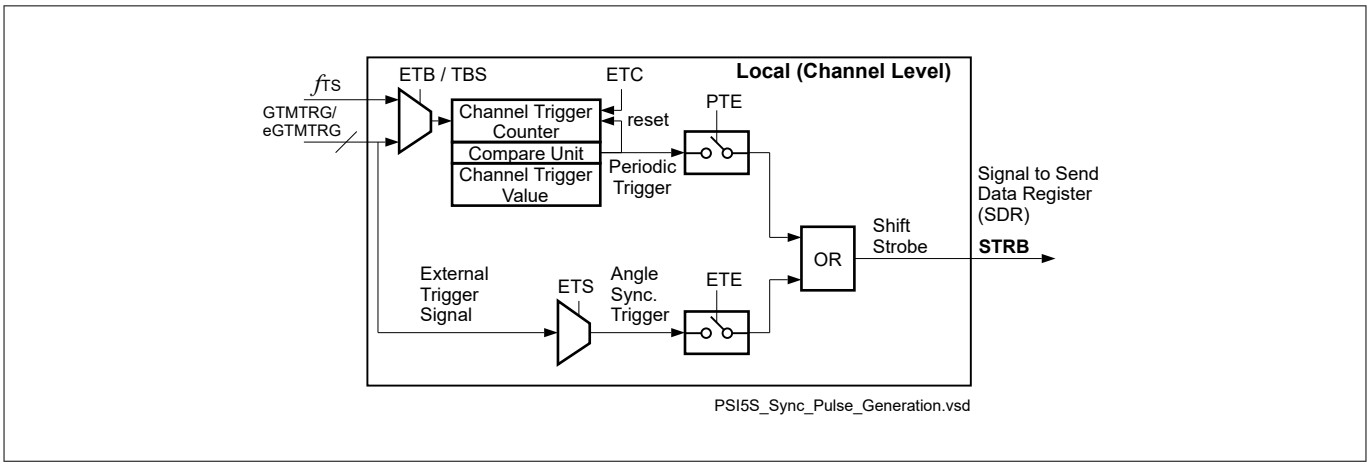


Figure 552 PSI5-S sync pulse generation

27.3.6.3 Send data preparation

The data frame to be sent is prepared in the SDRx register. The Send Data Register can be cleared and initialized by setting the SCRx.FLUS bit. The Send Control Register SCRx contains control bits that are required to send the data from the PSI5-S module to an external PHY. The following figure shows the mechanisms for how ECU to sensor communication data is treated:

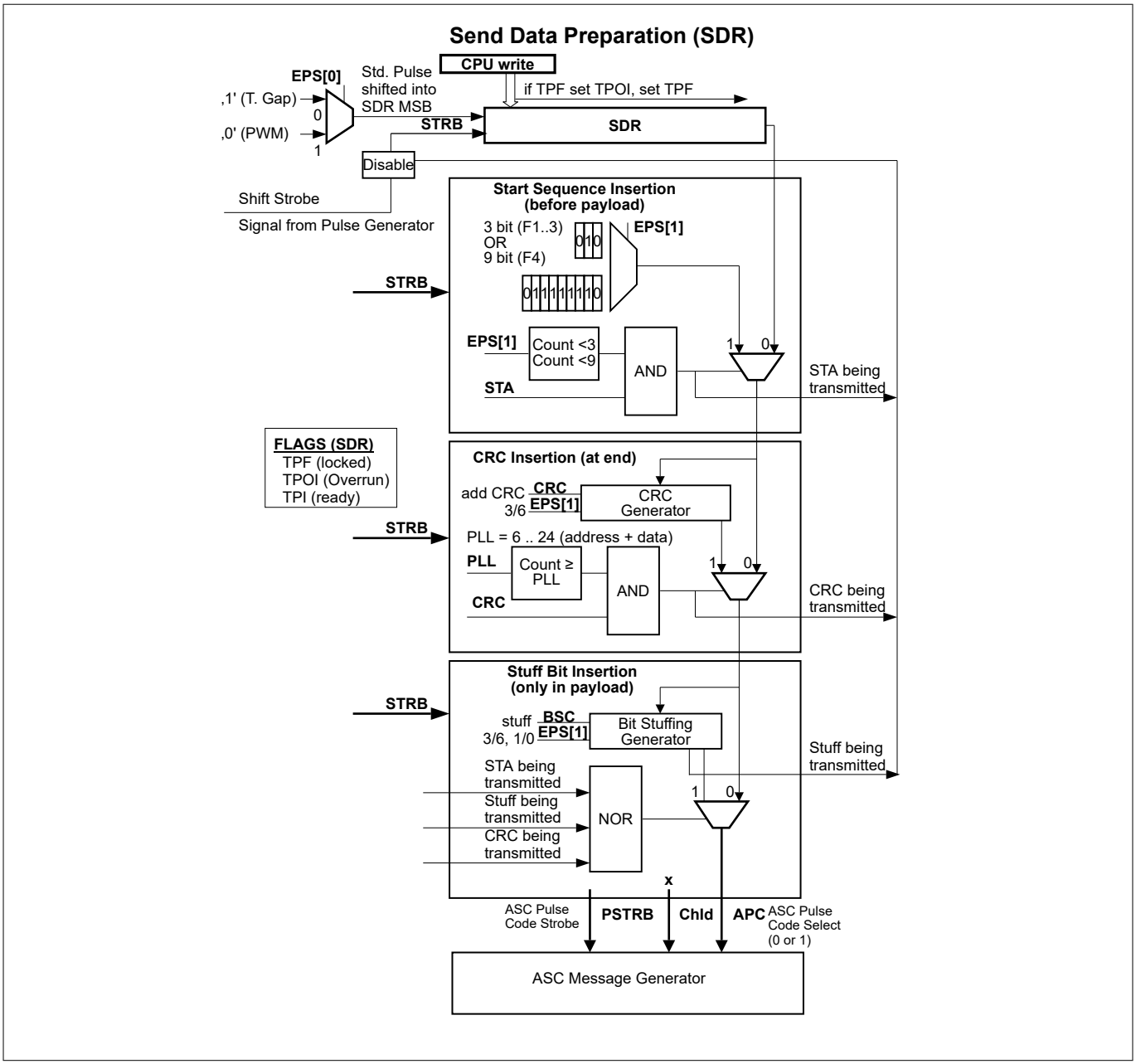


Figure 553 PSI5-S send data preparation

Bit EPS is used to select between the two methods of ECU to sensor communication. The bit is also used to set the frame format (format 1-3 or format 4). SCRx.PLL defines the payload length that is taken into account. SCRx.PLL needs to be configured before SDRx is issued for send data preparation. The send data preparation involves:

- Start sequence insertion (before the payload)
- CRC insertion (at the end)
- Stuff bit insertion (in the payload)

All three processes are driven by the shift strobe signal from pulse generator STRB. The start sequence insertion is controlled by the SCRx.STA bit. When STA is cleared, no start sequence is generated. When STA is set, the start sequence is automatically generated by the hardware. Similarly, the CRC insertion is controlled by the SCRx.CRC bit. When CRC is cleared, there is no automatic generation of CRC. It can still be written by software along with the data. When CRC bit is set, the CRC is automatically generated by the hardware. Bit BSC controls the stuff bit insertion. The SCRx.TPF flag is set when there is a data preparation in progress. At this moment, when the SDRx register is written, the old data is not overwritten and TPOI interrupt request flag is set. Once the send data has been moved completely and when the SDRx register is ready to be written with a new value, TPI interrupt request flag is set.

27.3.7 Message generation

The following figure shows the message generation logic.

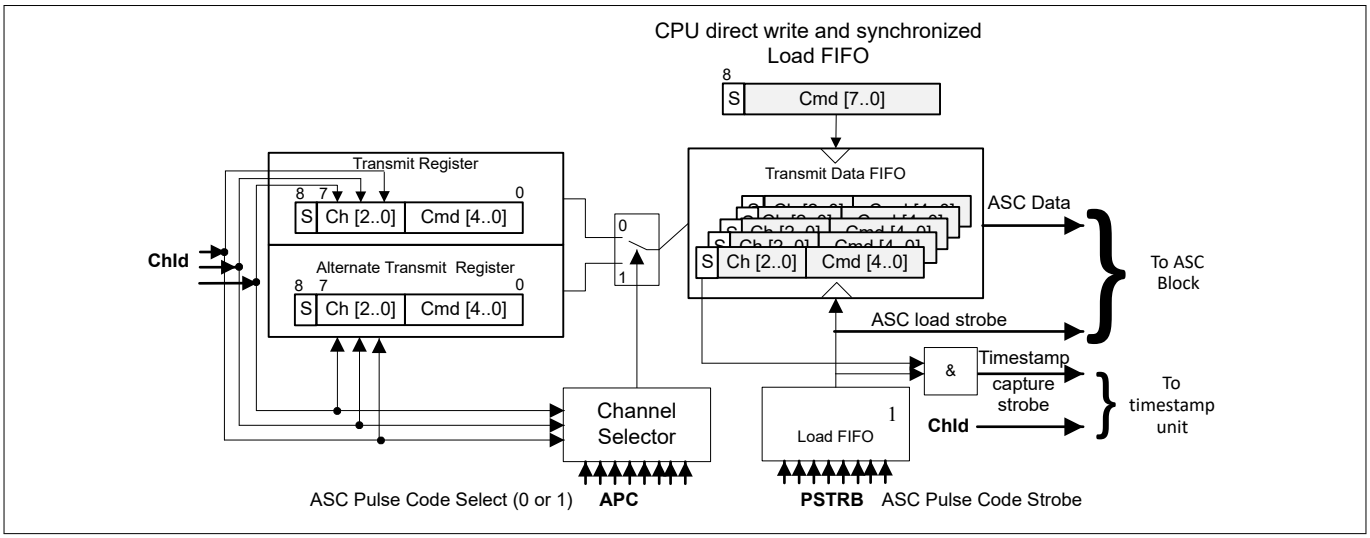


Figure 554 PSI5-S message generation logic

ASC Pulse Code Strobe PSTRB, channel ID ChId, and ASC Pulse Code Select APC are the signals from the send data preparation that are used for the ASC message generation (UART). The generated ASC commands are sorted into an internal FIFO of 8 entries of 9 bits width. The MSB stores the information, if a command is generated from the sync pulse generation (ASC Pulse Code Strobe PSTRB issued the write into FIFO) or if the CPU wrote a command.

The APC Pulse Code Select signal from the send data preparation, selects between the pulse code 0 or 1. This is important for the timestamp generation. Usually the timestamp is captured when a sync pulse command leaves the FIFO. That is, it is copied to the ASC TX buffer. Commands stored in the FIFO can only trigger a timestamp capture if they are marked as sync pulse in bit 8. Nevertheless the CPU can set bit CDW.TSI if the command is to be treated like a sync pulse command and a timestamp should be captured. Note that if RCRAx.TSTS is set, the sync pulses do not trigger the timestamp capture. In this case, the packet frame reception triggers the capture and bit 9 is ignored.

There is no further arbitration of the CPU and Message Generation Unit. The PHY answers CPU commands with standard frames of fixed predefined length in channel 0 frame 0.

Interrupt FOI is issued if a transfer to the FIFO was generated by the Message Generation Unit or by CPU direct write register (CDW) and if the FIFO was full. In a correct setup this will never be the case as the bandwidth of the ASC is assumed to be by far higher than the write bandwidth in order to have short delays. In most cases too many write accesses to CDW will be the root cause. This exceptional condition can be handled either by waiting until the FIFO has transferred at least one command or by a module reset by software. It should not occur in the application but FOI should help software generation and debugging.

The transfer path of a command to the ASC block could add jitter to sync pulses.

In case several channels request a pulse at exactly the same time, as shown in the figure above by two transmit registers (Transmit and Alternate Transmit), the channel with the lower channel number is served first and CDW has highest priority.

27.3.8 DMA support

The received frames come in sequentially and are XCRC checked. The complete re-assembled frame including channel Id (ChId), frame ID (FId), and status are available at RDR and RDS. TSM shows the latest timestamp of the currently received frame.

RDR, RDS, and TSM are located in 3 sequentially addressed registers. In this way they are easily readable by DMA.

Related information

Dedicated DMA channels on page 4989

27.3.8.1 Dedicated DMA channels

With one single DMA channel, one common frame buffer in the system memory can be built up. All frames can be collected from RDR, RDS and TSM registers. The DMA source address is programmed to be always RDS, block transfer is set to 3 words with auto address increment. The target is set to be automatically incremented. This results in one buffer in system memory with all frames from all channels.

Individual buffers for each channel can be built up if more DMA channels are used. Each channel can generate its own receive triggers RDI and RSI. Thus individual DMAs can be set up for each of the 8 channels similarly to the single DMA use case but with individual target addresses for each channel.

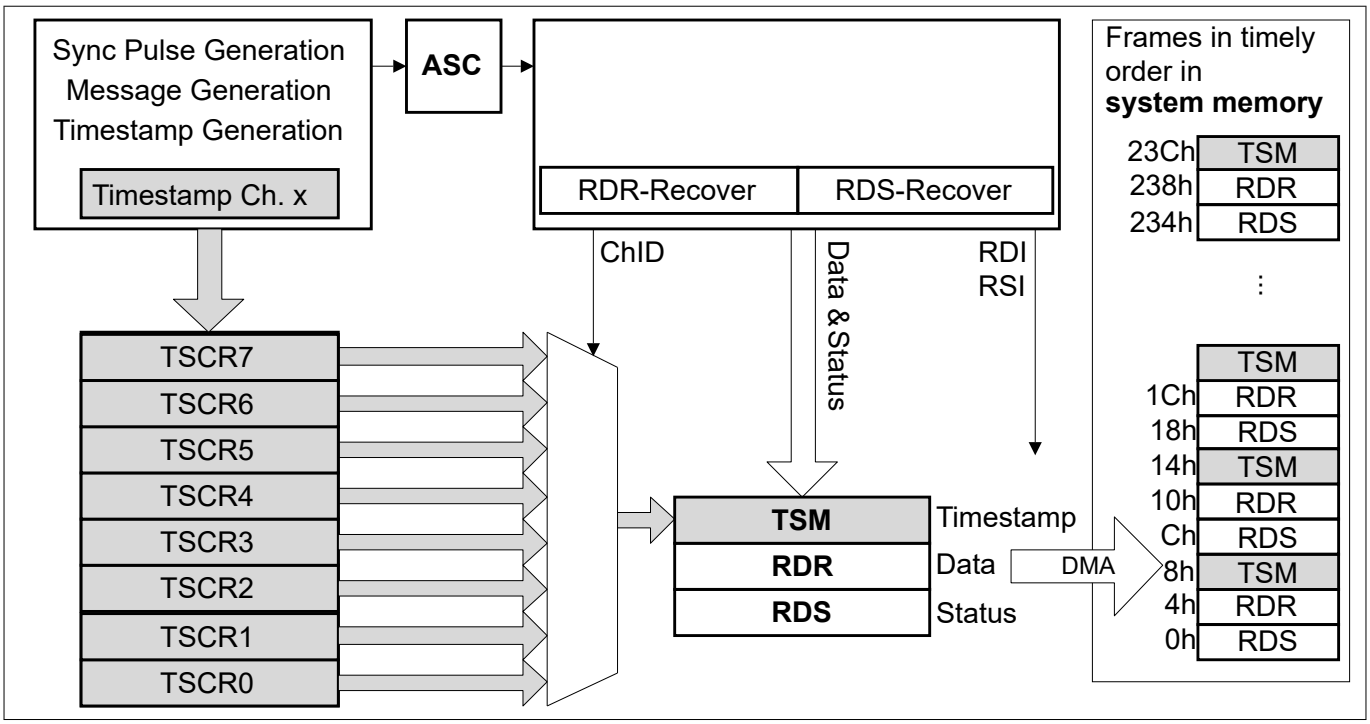


Figure 555 PSI5-S single DMA channel

27.3.8.2 Two daisy-chained DMAs

The PSI5-S supports sorting by channel with only two DMA channels. The register TAR contains the target address that is to be copied by the first DMA channel into the target address register of the second DMA channel. Source and target of the first DMA are fixed (DMA register SADR and DADR). The target address offered in TAR is calculated by PSI5-S by adding an offset value to the programmable base address defined in register BAR. This allows for locating the channel buffers at any location in system memory.

The resulting base + offset is presented in the register TAR. The target address depends on:

- Base address defined in the BAR register
- ChID: Offset of channel number currently received (6 buffer lines, 3 words per frame and 4 bytes per word gives 72 bytes times channel number). This allows to receive 6 frames per channel which equals the maximum number of PSI5 frames per sync pulse
- FID: Offset of frame ID currently received (3 words or 12 bytes per frame)

  Note that the frame ID used for this sorting depends on RCRAx.FIDS. Particularly in Asynchronous mode, RCRAx.FIDS needs to be set so that the FID from the packet frame is replaced by a rolling number 0 to 5.
- Note that the Message Reassembly unit forces non-recoverable messages to be stored at the address for ChID = 0, FID = 1, but with original ChID and FID. That is, RDS contains the ChID and FID as received in the first UART frame for better debugging
- The source address of the second DMA channel needs to be programmed to be fixed to the address of RDS. Use shadow control to “wrap around” the source address. (MEmADICR.SHCT must be set to 0x0101b).

  Block transfer with address auto incrementing must be selected and the number of transactions programmed to 3 (see the SDMA chapter, MEmCHCR.BLKM) 32-bit (the data width is defined in MEmCHCR.CHDW) accesses.

  Use shadow control to wrap around the source address. (MEmADICRSHCT must be set to 0x0101b)

The second DMA must be of the next lower priority than the first. Selecting hardware request input through CHCRxz.PRSEL allows for daisy chaining the two DMA channels. See following figure.

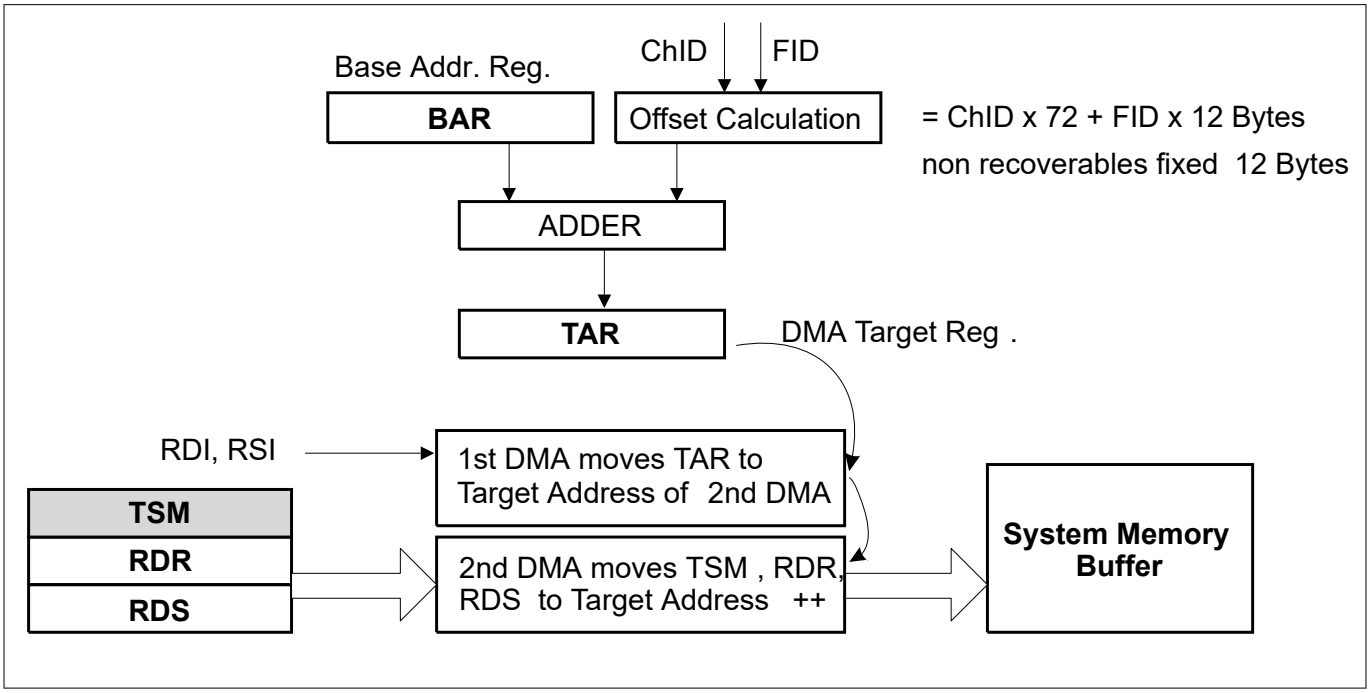


Figure 556 PSI5-S dual DMA support (daisy chained)

The following table shows the detailed mapping of the received frames resulting from the described calculation rules:

Table 1192 Frame mapping

Offset (dec)	Offset (hex)	Channel/frame	Content
0	0_H	Channel 0 frame 0	Status (Reserved for transceiver messages)
4	4_H	Channel 0 frame 0	Data (Reserved for transceiver messages)
8	8_H	Channel 0 frame 0	Timestamp (Reserved for transceiver messages)
12	C_H	Channel 0 frame 1	Status (Reserved for non-recoverable messages)
16	10_H	Channel 0 frame 1	Data (Reserved for non-recoverable messages)
20	14_H	Channel 0 frame 1	Timestamp (Reserved for non-recoverable messages)
24	18_H	Channel 0 frame 2	Status
28	$1C_H$	Channel 0 frame 2	Data
32	20_H	Channel 0 frame 2	Timestamp
36	24_H	Channel 0 frame 3	Status

(table continues...)

Table 1192 (continued) Frame mapping

Offset (dec)	Offset (hex)	Channel/frame	Content
40	28_H	Channel 0 frame 3	Data
44	$2C_H$	Channel 0 frame 3	Timestamp
48	30_H	Channel 0 frame 4	Status
52	34_H	Channel 0 frame 4	Data
56	38_H	Channel 0 frame 4	Timestamp
60	$3C_H$	Channel 0 frame 5	Status
64	40_H	Channel 0 frame 5	Data
68	44_H	Channel 0 frame 5	Timestamp
72	48_H	Channel 1 frame 0	Status
76	$4C_H$	Channel 1 frame 0	Data
80	50_H	Channel 1 frame 0	Timestamp
84	54_H	Channel 1 frame 1	Status
88	58_H	Channel 1 frame 1	Data
92	$5C_H$	Channel 1 frame 1	Timestamp
96	60_H	Channel 1 frame 2	Status
100	64_H	Channel 1 frame 2	Data
104	68_H	Channel 1 frame 2	Timestamp
108	$6C_H$	Channel 1 frame 3	Status
112	70_H	Channel 1 frame 3	Data
116	74_H	Channel 1 frame 3	Timestamp
120	78_H	Channel 1 frame 4	Status
124	$7C_H$	Channel 1 frame 4	Data
128	80_H	Channel 1 frame 4	Timestamp
132	84_H	Channel 1 frame 5	Status
136	88_H	Channel 1 frame 5	Data
140	$8C_H$	Channel 1 frame 5	Timestamp
144	90_H	Channel 2 frame 0	Status
148	94_H	Channel 2 frame 0	Data
152	98_H	Channel 2 frame 0	Timestamp
156	$9C_H$	Channel 2 frame 1	Status
160	$A0_H$	Channel 2 frame 1	Data
164	$A4_H$	Channel 2 frame 1	Timestamp
168	$A8_H$	Channel 2 frame 2	Status

(table continues...)

Table 1192 (continued) Frame mapping

Offset (dec)	Offset (hex)	Channel/frame	Content
172	AC_H	Channel 2 frame 2	Data
176	$B0_H$	Channel 2 frame 2	Timestamp
180	$B4_H$	Channel 2 frame 3	Status
184	$B8_H$	Channel 2 frame 3	Data
188	BC_H	Channel 2 frame 3	Timestamp
192	$C0_H$	Channel 2 frame 4	Status
196	$C4_H$	Channel 2 frame 4	Data
200	$C8_H$	Channel 2 frame 4	Timestamp
204	CC_H	Channel 2 frame 5	Status
208	$D0_H$	Channel 2 frame 5	Data
212	$D4_H$	Channel 2 frame 5	Timestamp
216	$D8_H$	Channel 3 frame 0	Status
220	DC_H	Channel 3 frame 0	Data
224	$E0_H$	Channel 3 frame 0	Timestamp
228	$E4_H$	Channel 3 frame 1	Status
232	$E8_H$	Channel 3 frame 1	Data
236	EC_H	Channel 3 frame 1	Timestamp
240	$F0_H$	Channel 3 frame 2	Status
244	$F4_H$	Channel 3 frame 2	Data
248	$F8_H$	Channel 3 frame 2	Timestamp
252	FC_H	Channel 3 frame 3	Status
256	100_H	Channel 3 frame 3	Data
260	104_H	Channel 3 frame 3	Timestamp
264	108_H	Channel 3 frame 4	Status
268	$10C_H$	Channel 3 frame 4	Data
272	110_H	Channel 3 frame 4	Timestamp
276	114_H	Channel 3 frame 5	Status
280	118_H	Channel 3 frame 5	Data
284	$11C_H$	Channel 3 frame 5	Timestamp
288	120_H	Channel 4 frame 0	Status
292	124_H	Channel 4 frame 0	Data
296	128_H	Channel 4 frame 0	Timestamp
300	$12C_H$	Channel 4 frame 1	Status

(table continues...)

Table 1192 (continued) Frame mapping

Offset (dec)	Offset (hex)	Channel/frame	Content
304	130_H	Channel 4 frame 1	Data
308	134_H	Channel 4 frame 1	Timestamp
312	138_H	Channel 4 frame 2	Status
316	$13C_H$	Channel 4 frame 2	Data
320	140_H	Channel 4 frame 2	Timestamp
324	144_H	Channel 4 frame 3	Status
328	148_H	Channel 4 frame 3	Data
332	$14C_H$	Channel 4 frame 3	Timestamp
336	150_H	Channel 4 frame 4	Status
340	154_H	Channel 4 frame 4	Data
344	158_H	Channel 4 frame 4	Timestamp
348	$15C_H$	Channel 4 frame 5	Status
352	160_H	Channel 4 frame 5	Data
356	164_H	Channel 4 frame 5	Timestamp
360	168_H	Channel 5 frame 0	Status
364	$16C_H$	Channel 5 frame 0	Data
368	170_H	Channel 5 frame 0	Timestamp
372	174_H	Channel 5 frame 1	Status
376	178_H	Channel 5 frame 1	Data
380	$17C_H$	Channel 5 frame 1	Timestamp
384	180_H	Channel 5 frame 2	Status
388	184_H	Channel 5 frame 2	Data
392	188_H	Channel 5 frame 2	Timestamp
396	$18C_H$	Channel 5 frame 3	Status
400	190_H	Channel 5 frame 3	Data
404	194_H	Channel 5 frame 3	Timestamp
408	198_H	Channel 5 frame 4	Status
412	$19C_H$	Channel 5 frame 4	Data
416	$1A0_H$	Channel 5 frame 4	Timestamp
420	$1A4_H$	Channel 5 frame 5	Status
424	$1A8_H$	Channel 5 frame 5	Data
428	$1AC_H$	Channel 5 frame 5	Timestamp
432	$1B0_H$	Channel 6 frame 0	Status

(table continues...)

Table 1192 (continued) Frame mapping

Offset (dec)	Offset (hex)	Channel/frame	Content
436	$1B4_H$	Channel 6 frame 0	Data
440	$1B8_H$	Channel 6 frame 0	Timestamp
444	$1BC_H$	Channel 6 frame 1	Status
448	$1C0_H$	Channel 6 frame 1	Data
452	$1C4_H$	Channel 6 frame 1	Timestamp
456	$1C8_H$	Channel 6 frame 2	Status
460	$1CC_H$	Channel 6 frame 2	Data
464	$1D0_H$	Channel 6 frame 2	Timestamp
468	$1D4_H$	Channel 6 frame 3	Status
472	$1D8_H$	Channel 6 frame 3	Data
476	$1DC_H$	Channel 6 frame 3	Timestamp
480	$1E0_H$	Channel 6 frame 4	Status
484	$1E4_H$	Channel 6 frame 4	Data
488	$1E8_H$	Channel 6 frame 4	Timestamp
492	$1EC_H$	Channel 6 frame 5	Status
496	$1F0_H$	Channel 6 frame 5	Data
500	$1F4_H$	Channel 6 frame 5	Timestamp
504	$1F8_H$	Channel 7 frame 0	Status
508	$1FC_H$	Channel 7 frame 0	Data
512	200_H	Channel 7 frame 0	Timestamp
516	204_H	Channel 7 frame 1	Status
520	208_H	Channel 7 frame 1	Data
524	$20C_H$	Channel 7 frame 1	Timestamp
528	210_H	Channel 7 frame 2	Status
532	214_H	Channel 7 frame 2	Data
536	218_H	Channel 7 frame 2	Timestamp
540	$21C_H$	Channel 7 frame 3	Status
544	220_H	Channel 7 frame 3	Data
548	224_H	Channel 7 frame 3	Timestamp
552	228_H	Channel 7 frame 4	Status
556	$22C_H$	Channel 7 frame 4	Data
560	230_H	Channel 7 frame 4	Timestamp
564	234_H	Channel 7 frame 5	Status

(table continues...)

Table 1192 (continued) Frame mapping

Offset (dec)	Offset (hex)	Channel/frame	Content
568	238_H	Channel 7 frame 5	Data
572	$23C_H$	Channel 7 frame 5	Timestamp

27.3.8.3 Interrupts for DMA support

Interrupts for single DMA channel usage.

Interrupts RDI or RSI can be used as trigger for the DMA transfer from TSM/RDS/RDR.

Interrupts for two daisy chained DMA usage.

If one of the buffers is full, this is signaled to the software by the interrupt INTSTATx.CHCI. The “status buffer full” is defined as follows: current number of received frames counted in FCNT.FCx equals the number of expected frames configured in NFC.NFx. In Synchronous mode this is the case if all expected PSI5 frames after a sync pulse are received by PSI5-S. The number of expected frames is looked up in the register NFC.NFx for channel x. In Asynchronous mode, NFC.NFx should be programmed to 6 while RCRAx.FIDS is set. This way the FID for an asynchronous channel is a rolling number 0 to 5 and the 6 buffer locations are used completely before INTSTATx.CHCI is issued. See following figure.

There is a delay between INTSTATx.CHCI and the completion of the DMA transfer of the last set of RDS, RDR and TSM to the system memory. The software needs special checks to grant data consistency, for example evaluating the DMA interrupts or comparison of the timestamps in the system memory.

CHCI is not granting that all data is completely moved to system memory by the DMA.

Improved data consistency with packet frame count (PFC)

For data consistency, RDR, RDS and TSM are tagged with a packet frame count value. For each channel there is a separate internal packet counter. This allows to read a triplet of RDR, RDS and TSM from the system memory without looking at interrupts. If the PFC is identical all three values belong together. Otherwise, they need to be read again until PFC is identical.

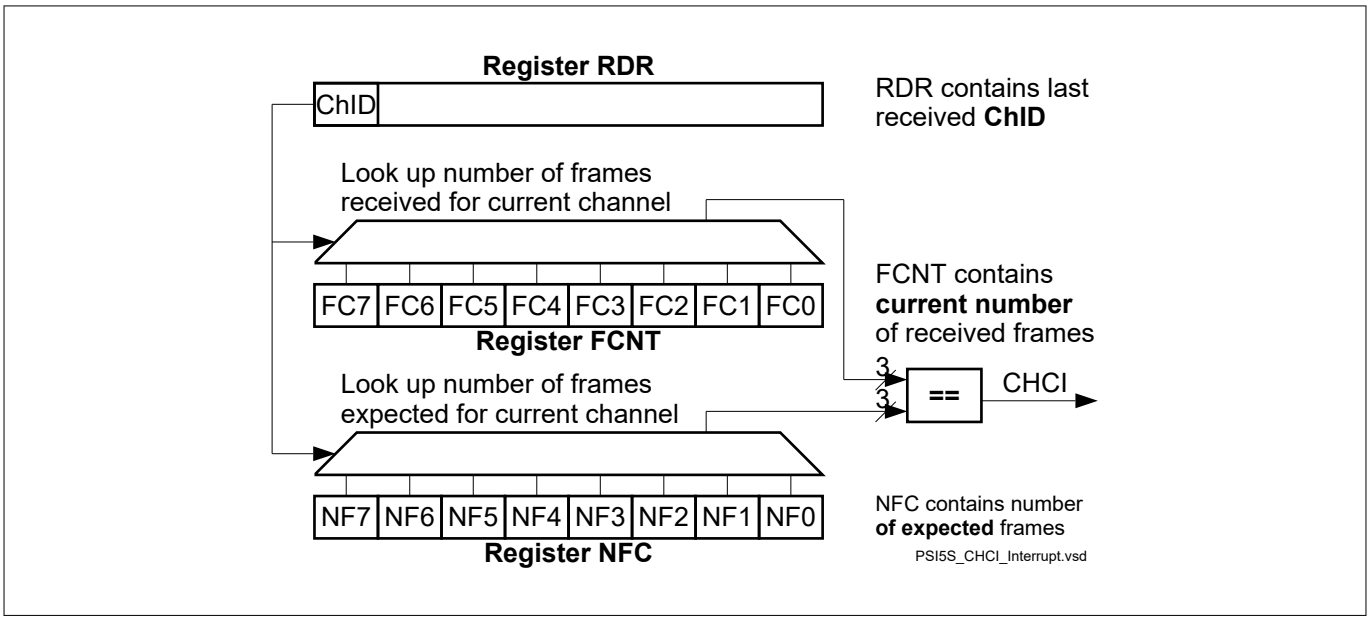


Figure 557 PSI5-S generation of interrupt CHCI

27.3.9 Error detection capabilities

Each PSI5-S channel can detect and signal the following error conditions:

Protocol Level:

- Packet frame CRC error (XCRCI)
- PSI5 frame checksum error (CRCI)
- Frame not sent in time/UART frames missing (TEI)
- Error bits set in packet frame (HDI)
- Errors signaled by ASC sub module (that is PE, FE, OE)

Transfer management level:

- Receive data buffer overrun (RBI)
- ECU to sensor data buffer underrun (TBIR)

27.3.10 Special use of channel 0

Note that data that is received in incomplete or too long packet frames is always copied - as is - to channel 0 frame ID 1. This can happen, if UART frames are lost, the PHY sends non allowed data or idle before the actual end of the packet frame. See related information.

This is the only actual reservation/limitation in functionality of channel 0. I.e. all frames can be received and sorted correctly in channel 0 as long as the packet frame is configured with: ChID = 0. It is recommended not to use ChID = 0 and FID = 1 for normal sensor frames but to keep FID = {2, 3 ,4 ,5}.

The target application assumes, that the PHY sends messages for its own house keeping with ChID = 0, FID = 0. This is assumed in this document and for example shown in the table in the Two daisy-chained DMAs chapter. Nevertheless, a different application may not have this functionality or may use a different ChID/FID configuration.

Related information

Packet frames received from PHY on page 4970

Two daisy-chained DMAs on page 4990

27.3.11 ASC kernel description

The following figure shows a global view of the ASC interface. The ASC interface is fully integrated into the PSI5-S module. It can be switched to dedicated internal use of the PSI5-S module or to ASC-only mode where all PSI5-S functionality is disabled. In the last case, the whole PSI5-S module represents one additional ASC interface in the system.

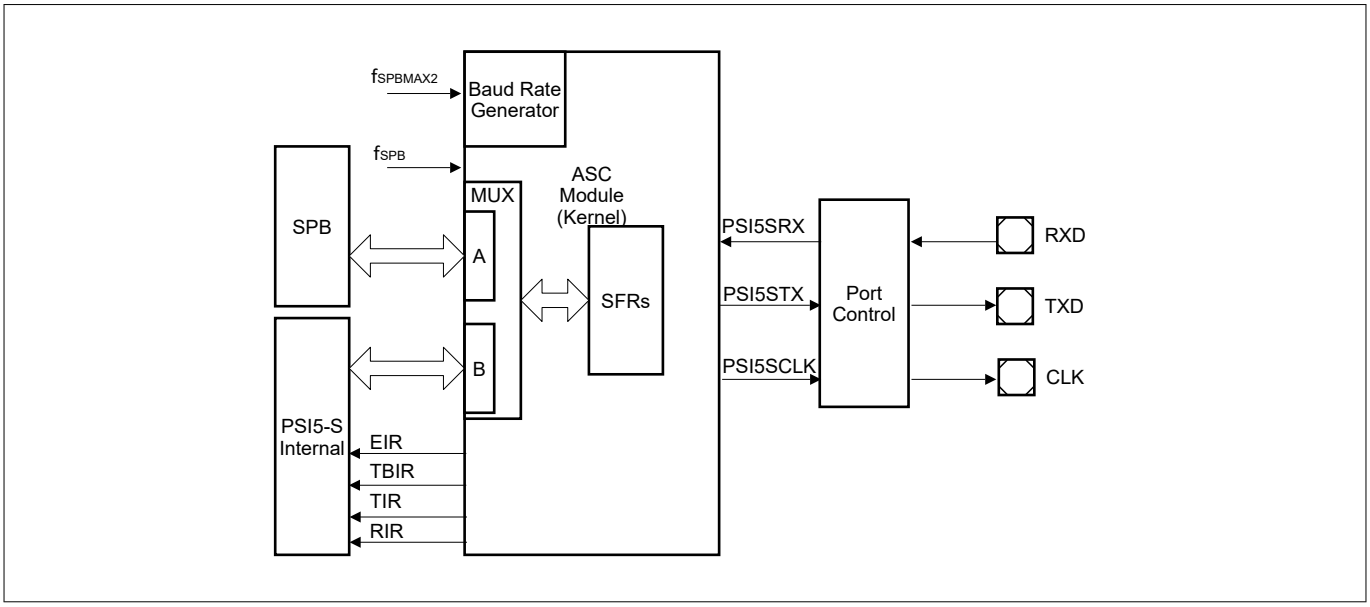


Figure 558 General block diagram of the ASC interface

The ASC module communicates with the external world through two I/O lines. The RXD line is the receive data input signal (and also output signal in Synchronous mode), and TXD is the transmit output signal.

Clock control, address decoding and interrupt service request control are managed outside the ASC module kernel.

27.3.11.1 ASC kernel overview

The ASC provides serial communication between this device and other micro controllers, microprocessors, or external peripherals.

The ASC supports full-duplex asynchronous communication and half-duplex synchronous communication. In Synchronous mode, data is transmitted or received synchronous to a shift clock that is generated by the ASC internally. In Asynchronous mode, 8-bit or 9-bit data transfer, parity generation, and the number of Stop bits can be selected. Parity, framing, and overrun error detection are provided to increase the reliability of data transfers. Transmission and reception of data is double-buffered in ASC-only mode. During internal PSI5-S mode, double-buffering is switched off for the transmitter. The receiver stays double buffered. For multiprocessor communication, a mechanism is included to distinguish address bytes from data bytes. Testing is supported by a Loop-Back option. A 13-bit baud rate generator provides the ASC with a separate serial clock signal, which can be accurately adjusted by a prescaler implemented as fractional divider.

Features

- Full-duplex asynchronous operating modes
  - 8-bit or 9-bit data frames, LSB first
  - Parity-bit generation/checking
  - Separate parity control for receive and transmit
  - One or two Stop bits
  - Baud rate from 12,5 Mbit/s to 3 bit/s (at 200 MHz module clock)

  - Multiprocessor mode for automatic address/data byte detection
  - Loop-back capability
- Half-duplex 8-bit synchronous operating mode
  - Baud rate from 25 Mbit/s to 2034,5 bit/s (at 200 MHz module clock)
- Double-buffered transmitter (in ASC-only mode)/receiver
- Interrupt generation
  - On a transmit buffer empty condition
  - On a transmit last bit of a frame condition
  - On a receive buffer full condition
  - On an error condition (frame, parity, overrun error)
- Implementation features
  - Connections to DMA controller
  - Connections from GTM

Related information

Packet frames received from PHY on page 4970

27.3.11.2 ASC kernel general operation

The ASC supports full-duplex asynchronous communication and half-duplex synchronous communication. In Synchronous mode, data is transmitted or received synchronously to a shift clock generated by the microcontroller. In Asynchronous mode, 8-bit or 9-bit data transfer, parity generation, and the number of Stop bits can be selected. Parity, framing, and overrun error detection are provided to increase the reliability of data transfers. Transmission and reception of data are double-buffered. For multiprocessor communication, a mechanism is included to distinguish address bytes from data bytes. Testing is supported by a Loop-Back option. A 13-bit baud rate generator provides the ASC with a separate serial clock signal, which can be accurately adjusted by a prescaler implemented as fractional divider.

A transmission is started by writing to the Transmit Buffer Register, TBUF. Only the number of data bits determined by the selected operating mode will actually be transmitted; that is, bits written to positions 9 through 15 of register TBUF are always insignificant. Data transmission is double-buffered, so a new character may be written to TBUF before the transmission of the previous character is complete. This allows a back-to-back transmission of characters to take place without gaps.

Data reception is enabled by the receiver enable bit CON.REN. After a reception has been completed, the received data and, if provided by the selected operating mode, the received parity bit can be read from the (read-only) Receive Buffer Register RBUF. Unused bits in the upper half of RBUF that are not required in the selected operating mode will be read as zeros.

Data reception is double-buffered, so that reception of a second character may already begin before the previously received character has been read out of the receive buffer register. During the exclusive use by the PSI5-S, double buffering is switched off. The PSI5-S internal output FIFO is used instead. As commands for sync pulses are timestamped inside PSI5-S as they leave the internal FIFO, the double buffering would add unwanted jitter. In all modes, receive buffer overrun error detection can be selected through bit CON.OEN. When enabled, the overrun error status flag CON.OE and the error interrupt request line INTOV.EIR will be activated when the receive buffer register has not been read by the time reception of a second character is complete. In this case, the previously received character in the receive buffer is overwritten.

The Loop-Back option (selected by bit CON.LB) allows the data currently being transmitted to be received simultaneously in the receive buffer. This may be used to test serial communication routines at an early stage without having to provide an external network. In Loop-Back mode, the alternate input/output function of port pins is not required.

27.3.11.3 Asynchronous operation

Asynchronous mode supports full-duplex communication, in which both transmitter and receiver use the same data frame format and have the same baud rate. Data is transmitted on pin TXD and received on pin RXD. The following figure shows the block diagram of the ASC when operating in Asynchronous mode.

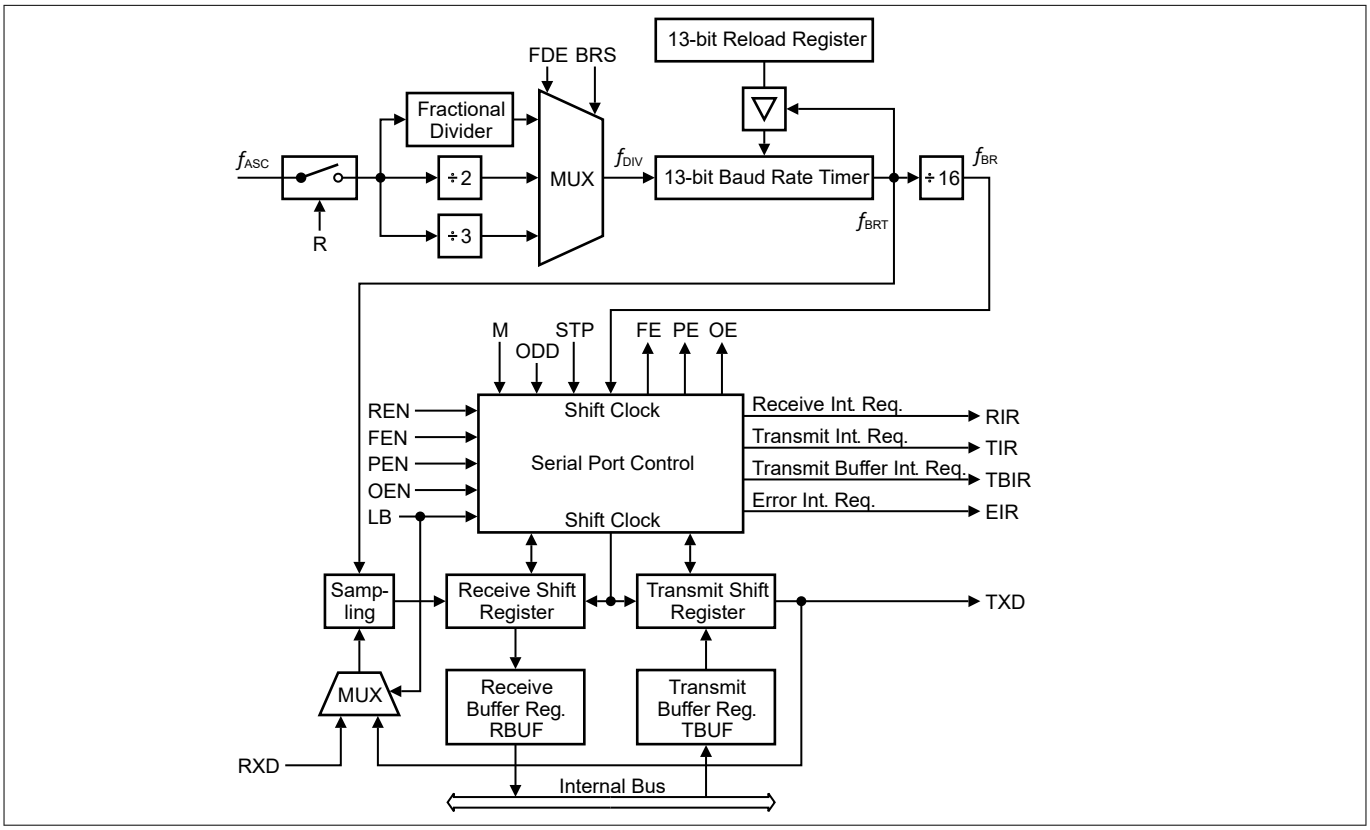


Figure 559 Asynchronous mode of the ASC

27.3.11.3.1 Asynchronous data frames

Asynchronous data frames can consist of 8-bit or 9-bit data frames.

8-bit data frames

The 8-bit data frames consist of either eight Data bits D7…D0 (CON.M=001_B), or of seven Data bits D6…D0 plus an automatically generated Parity bit (CON.M=011_B). Parity may be odd or even, depending on bit CON.ODD. An even Parity bit will be set if the modulo-2 sum of the seven data bits is 1. An odd Parity bit will be cleared in this case. Parity checking is enabled by bit CON.PEN (always OFF in 8-bit data mode). The parity error flag CON.PE will be set, along with the error interrupt request flag, if a wrong Parity bit is received. The received Parity bit itself will be stored in RBUF too.

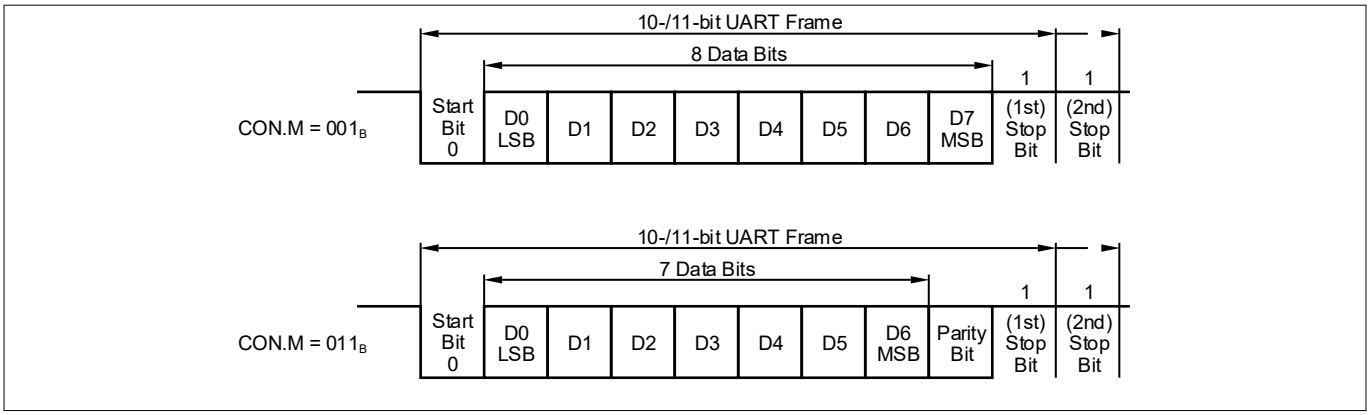


Figure 560 Asynchronous 8-bit frames

9-bit data frames

The 9-bit data frames consist of nine Data bits D8…D0 (CON.M=100_B), or of eight Data bits D7…D0 plus an automatically generated Parity bit (CON.M=111_B) or of eight Data bits D7…D0 plus Wake-up bit (CON.M=101_B). Parity may be odd or even, depending on bit CON.ODD. An even Parity bit will be set if the modulo-2-sum of the eight data bits is 1. An odd Parity bit will be cleared in this case. Parity checking is enabled by bit CON.PEN (always OFF in 9-bit data and Wake-up mode). The parity error flag CON.PE will be set along with the error interrupt request flag if a wrong parity bit is received. The received Parity bit itself will be stored in RBUF too.

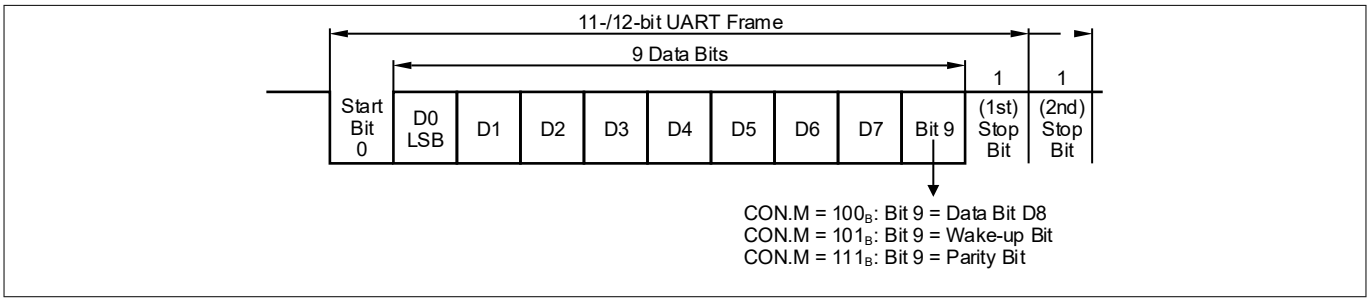


Figure 561 Asynchronous 9-bit frames

In Wake-up mode (CON.M=101_B), received frames are transferred to the receive buffer register only if the 9^{th} bit (the Wake-up bit) of the frame is 1. If this bit is 0, no receive interrupt request will be activated and no data will be transferred.

This feature can be used to control communication in multi-processor systems, for example:

When the master processor aims to transmit a block of data to one of several slaves, it first sends out an address ‘Byte’ (in this case, a ‘Byte’ consists of nine bits) that identifies the target slave. An address ‘Byte’ differs from a data ‘Byte’ in that the additional 9^{th} bit is a 1 for an address ‘Byte’ but is a 0 for a data ‘Byte’, so, no slave will be interrupted by a data ‘Byte’. An address ‘Byte’ will interrupt all slaves (operating in 8-bit data + Wake-up bit mode), so each slave can examine the eight LSBs of the received character (the address). The addressed slave will switch to 9-bit data mode (for example, by clearing bit CON.M[0]), which enables it to also receive the data 'Bytes' that will be coming (having the Wake-up bit cleared). The slaves that were not being addressed remain in 8-bit data + Wake-up bit mode, ignoring the following data ‘Bytes’.

27.3.11.3.2 Asynchronous transmission

Asynchronous transmission begins when the next overflow of the divide-by-16 baud rate timer (transition of the baud rate clock f_{BR}) occurs, if bit CON.R is set and data has been loaded into TBUF. The transmitted data frame consists of three elements:

1. The Start bit
2. The data field (8 or 9 bits, LSB first, including a Parity bit, if selected)
3. The delimiter (1 or 2 Stop bits)

Data transmission is double-buffered. When the transmitter is idle, the transmit data loaded into TBUF is immediately moved to the transmit shift register; thus, freeing TBUF for the next transmit data to be loaded. This is indicated by the transmit buffer interrupt request line INTOV.TBIR being activated. TBUF may then be loaded with the next transmit data while transmission of the previous one continues.

The transmit interrupt request line INTOV.TIR will be activated before the last bit of a frame is transmitted, that is, before the first or the second Stop bit is shifted out of the transmit shift register.

***Note**: A dedicated GPIO device pin which is connected to the module output pin TXD must be configured by software as alternate data output for asynchronous transmission.*

27.3.11.3.3 Asynchronous reception

Asynchronous reception is initiated by a falling edge (1-to-0 transition) on pin RXD, on the condition that bits CON.R and CON.REN are set. The receive data input pin RXD is sampled at sixteen times the rate of the selected baud rate. A majority decision of the 7^{th}, 8^{th} and 9^{th} sample determines the effective sampled bit value. This avoids erroneous results that may be caused by noise.

If the detected value is not a 0 when the Start bit is sampled, the receive circuit is reset and waits for the next 1-to-0 transition at pin RXD. If the Start bit proves valid, the receive circuit continues sampling and shifts the incoming data frame into the receive shift register.

When the last Stop bit has been received, the contents of the receive shift register are transferred to the Receive Data Buffer Register RBUF. Simultaneously, the receive interrupt request line INTOV.RIR is activated after the 9^{th} sample in the last stop bit time-slot (as programmed), regardless whether valid Stop bits have been received or not. The receive circuit then waits for the next Start bit (1-to-0 transition) at the receive data input line.

***Note**: A dedicated GPIO pin that is connected to the module input pin RXD must be configured by software as input for asynchronous reception.*

Asynchronous reception is stopped by clearing bit CON.REN. A currently received frame is completed including generation of the receive interrupt request and an error interrupt request, if appropriate. Start bits that follow this frame will not be recognized.

***Note**: In Wake-up mode, received frames are transferred to the receive buffer register only if the 9^{th} bit (the Wake-up bit) is 1. If this bit is 0, no receive interrupt request will be activated and no data will be transferred.*

27.3.11.3.4 RXD/TXD data path selection in asynchronous modes

The data paths for the serial input and output data in Asynchronous modes are affected by control bit CON.LB (loop-back) as shown in the following figure.

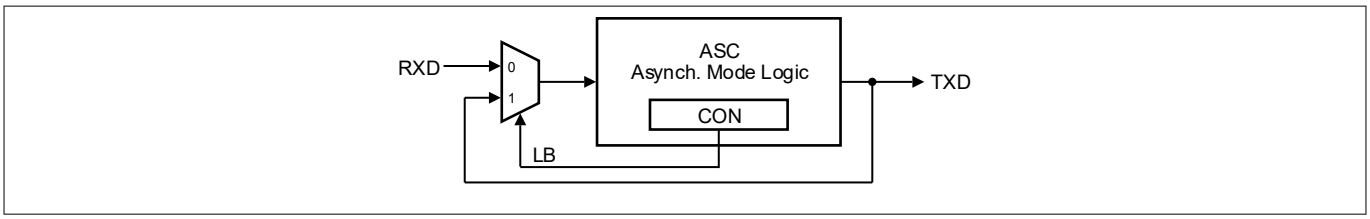


Figure 562 RXD/TXD data path selection in asynchronous modes

27.3.11.4 Synchronous operation

Synchronous mode supports half-duplex communication, usable for simple I/O expansion by shift registers. Data is transmitted and received by pin RXD while pin TXD outputs the shift clock. These signals are typically connected as alternate functions with GPIO port pins. Synchronous mode is selected with CON.M=000_B.

Eight data bits are transmitted or received synchronous to a shift clock generated by the internal baud rate generator. The shift clock is active only as long as data bits are transmitted or received.

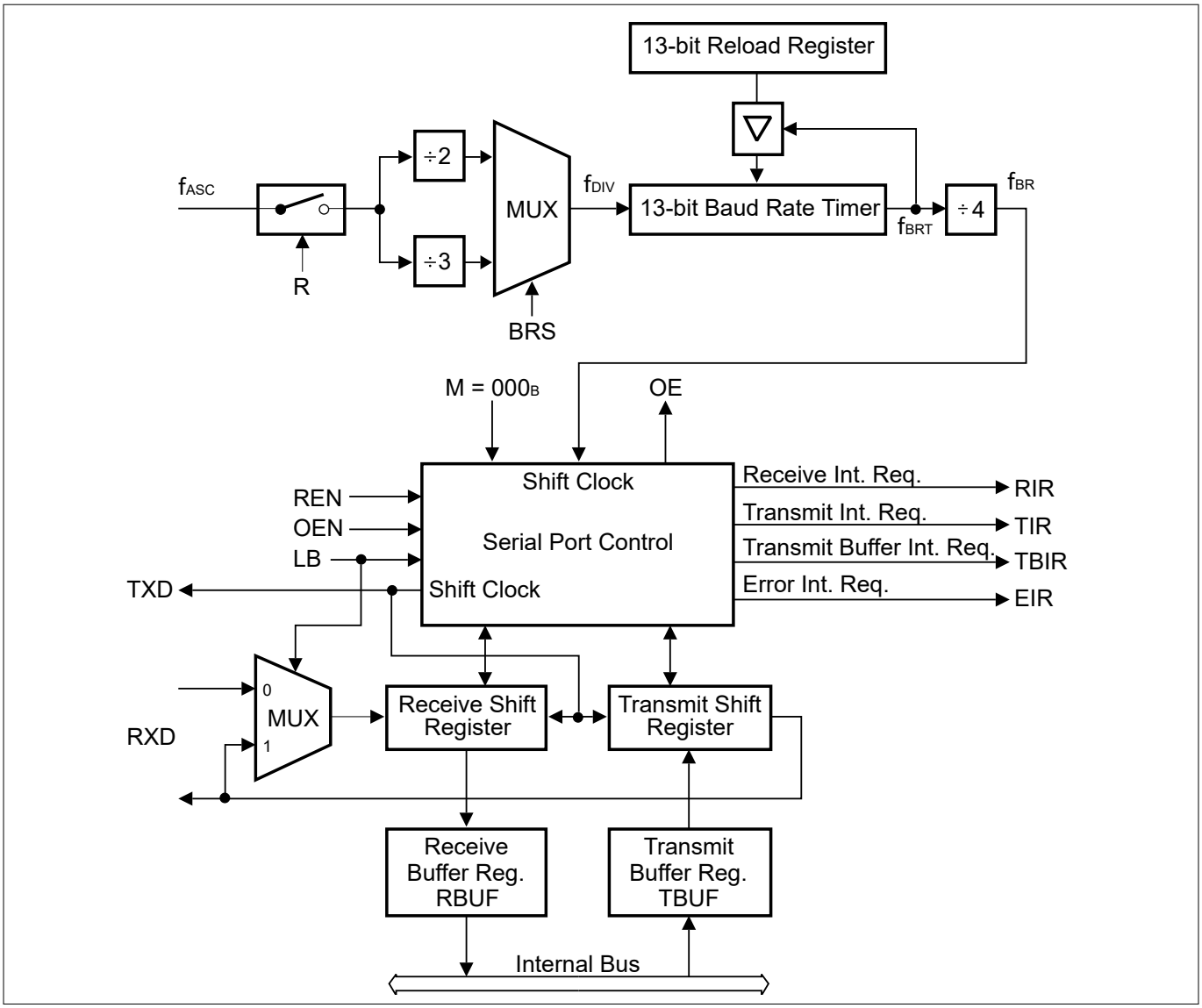


Figure 563 Synchronous mode of serial channel ASC

27.3.11.4.1 Synchronous transmission

Synchronous transmission begins after data has been loaded into TBUF, provided that CON.R is set and CON.REN=0 (half-duplex, no reception), with one exception: in asynchronous Loop-Back mode (bit CON.LB set), CON.REN must be set for reception of the transmitted byte. In synchronous Loop-Back mode, the transmitted byte can still be received if CON.REN=0. Data transmission is double-buffered. When the transmitter is idle, the transmit data loaded into TBUF is immediately moved to the transmit shift register, thus freeing TBUF for the next data to be sent. This is indicated by the transmit buffer interrupt request line INTOV.TBIR being activated. TBUF may now be loaded with the next data, while transmission of the previous one continues. The data bits are transmitted synchronously with the shift clock. After the bit time for the 8th data bit, both TXD and RXD will be set to high level, the transmit interrupt request line INTOV.TIR is activated, and serial data transmission stops.

***Note**: The dedicated GPIO device pins that are connected to TXD and RXD must be configured by software as alternate data outputs in order to provide the shift clock and the output data during synchronous transmission.*

27.3.11.4.2 Synchronous reception

Synchronous reception is initiated by setting bit CON.REN=1. If bit CON.R=1, the data applied at RXD is clocked into the receive shift register synchronously to the clock which is output at TXD. After the 8th bit has been shifted in, the contents of the receive shift register are transferred to the receive data buffer RBUF, the receive interrupt request line INTOV.RIR is activated, the receiver enable bit CON.REN is reset, and serial data reception stops.

Synchronous reception is stopped by clearing bit CON.REN. Any byte that is currently being received is completed, including the generation of the receive interrupt request and an error interrupt request, if appropriate. Writing to the transmit buffer register while a reception is in progress has no effect on reception and will not start a transmission.

If a previously received byte has not been read out of the receive buffer register by the time the reception of the next byte is complete, both the error interrupt request line INTOV.EIR and the overrun error status flag CON.OE will be activated/set, provided that the overrun check has been enabled by bit CON.OEN.

***Note**: The dedicated GPIO device pin that is connected to TXD must be configured by software as alternate data output in order to provide the shift clock. The dedicated GPIO device pin that is connected to RXD must be configured by software as input during synchronous reception.*

27.3.11.4.3 Synchronous timing

The following figure shows timing diagrams of the ASC Synchronous mode data reception and data transmission. In idle state, the shift clock is at high level. With the beginning of a synchronous transmission of a data byte, the data is shifted out at RXD with the falling edge of the shift clock. If a data byte is received through RXD, data is latched with the rising edge of the shift clock.

One shift clock cycle (f_{BR}) delay is inserted between two consecutive receive or transmit data bytes.

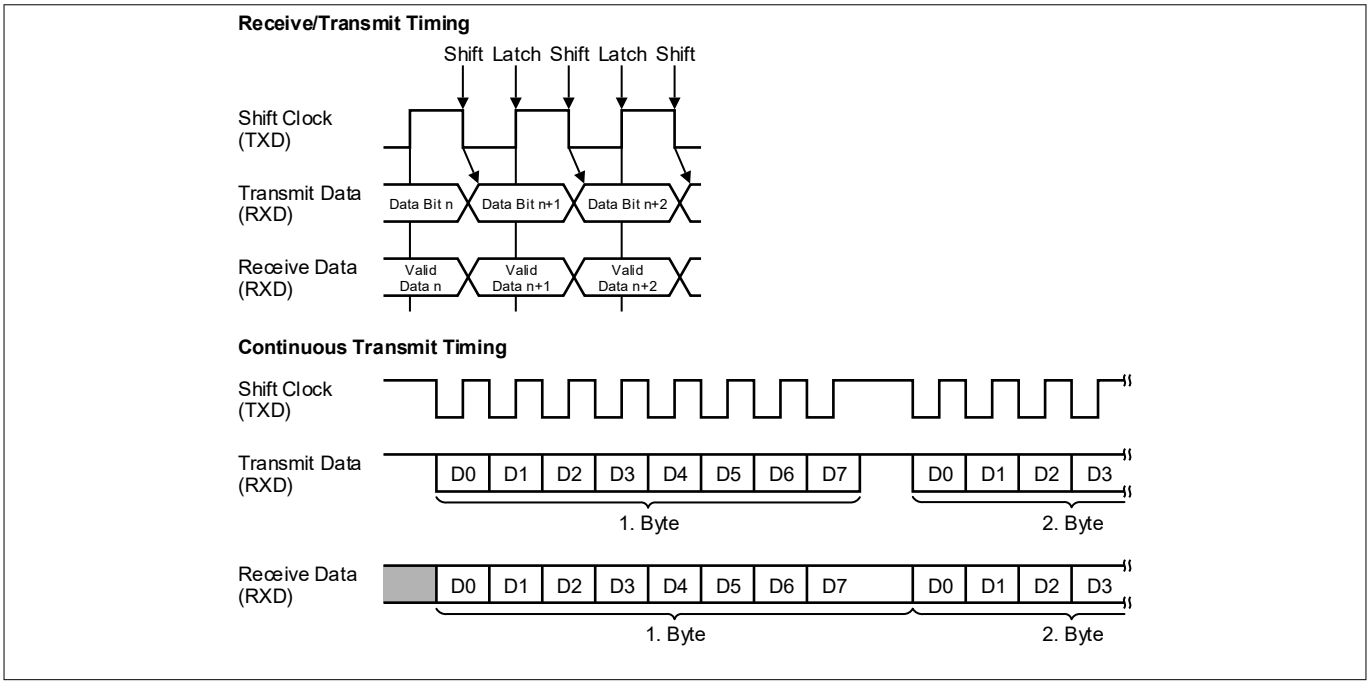


Figure 564 ASC synchronous mode waveforms

27.3.11.5 Baud rate generation

The ASC has its own dedicated 13-bit baud rate generator with 13-bit reload capability, allowing baud rate generation independent of other timers.

The baud rate generator is clocked with a clock (f_{DIV}) which is derived by a prescaler from the ASC module clock f_{ASC}. The baud rate timer is counting downwards and can be started or stopped through the baud rate generator run bit CON.R. Each underflow of the timer generates one clock pulse to the serial channel. The timer is reloaded with the value stored in its 13-bit reload register at each underflow. The resulting clock f_{BRT} is again divided by a factor for the baud rate clock ($\frac{f_{BRT}}{16}$ in Asynchronous operating modes and $\frac{f_{BRT}}{4}$ in Synchronous operating mode). The prescaler is selected by the bits CON.BRS and CON.FDE. In the Asynchronous operating modes, a fractional divider prescaler unit is available (in addition to the two fixed dividers) that allows selection of prescaler divider ratios of $\frac{n}{2048}$ with n=0-2047. Therefore, the baud rate of ASC is determined by the module clock, the content of register FDV, the reload value in register BG, and the operating mode (Asynchronous or Aynchronous).

Register BG is the dual-function baud rate generator/reload register. Reading BG returns the contents of the timer in bit field BR_VALUE (bits 31:13 return zero), while writing to BG always updates the reload register (bits 31:13 are insignificant).

An auto-reload of the timer with the contents of the reload register is performed each time BG is written to. However, if CON.R=0 at the time the write operation to BG is performed, the timer will not be reloaded until the first instruction cycle after CON.R=1. For a clean baud rate initialization, BG should only be written if CON.R=0. If BG is written with CON.R=1, an unpredictable behavior of the ASC may occur during running transmit or receive operations.

27.3.11.5.1 Baud rates in asynchronous mode

For asynchronous operation, the baud rate generator provides a clock f_{BRT} with sixteen times the rate of the established baud rate. Every received bit is sampled on the 7th, 8th and 9th cycle of this clock. The clock divider circuitry, which generates the input clock f_{DIV} for the 13-bit baud rate timer, is extended by a fractional divider circuitry that allows the adjustment of more accurate baud rates and the extension of the baud rate range.

The baud rate of the baud rate generator depends on the settings of the following bits and register values:

- Input clock f_{ASC}
- Selection of the baud rate timer input clock f_{DIV} by bits CON.FDE and CON.BRS
- If bit CON.FDE=1 (fractional divider): value of register FDV
- Value of the 13-bit reload register BG

The output clock of the baud rate timer with the reload register is the sample clock in the Asynchronous operating modes of the ASC. For baud rate calculations, this baud rate clock f_{BR} is derived from the sample clock f_{BRT} by a division of sixteen.

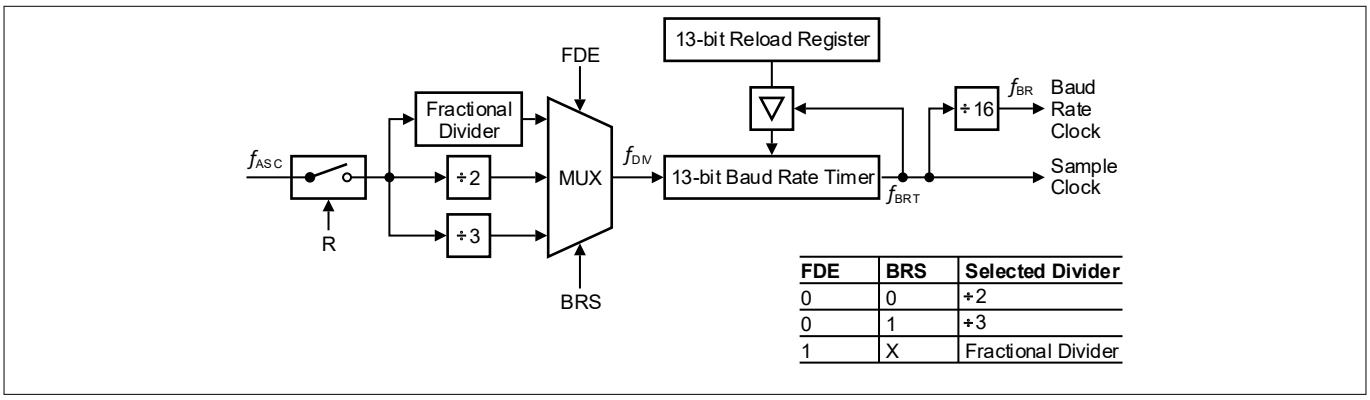


Figure 565 ASC baud rate generator circuitry in asynchronous modes

Using the fixed input clock divider

The baud rate for asynchronous operation of the serial channel ASC when using the fixed input clock divider ratios (CON.FDE=0) and the required BG reload value for a given baud rate can be determined by the following formulas:

Table 1193 Asynchronous baud rate formulas using the fixed input clock dividers

FDE	BRS	BG	Formula
0	0	0 … 8191	$\text{Baud rate} = \frac{f_{ASC}}{(32 \times (BG + 1))}$ $BG = \frac{f_{ASC}}{(32 \times \text{Baud rate}) - 1}$
	1		$\text{Baud rate} = \frac{f_{ASC}}{(48 \times (BG + 1))}$ $BG = \frac{f_{ASC}}{(48 \times \text{Baud rate}) - 1}$

BG represents the content of the reload register bit field BG.BR_VALUE, taken as an unsigned 13-bit integer.

The maximum baud rate that can be achieved for the Asynchronous operating modes when using the two fixed clock dividers and a module clock of 200 MHz is 6.25 Mbit/s.

Using the fractional divider

When the fractional divider is selected, the input clock f_{DIV} for the baud rate timer is derived from the module clock f_{ASC} by a programmable fractional divider. If CON.FDE=1, the fractional divider is activated. It divides f_{ASC} by a fraction of $\frac{n}{2048}$ for any value of n from 0 to 2047. If n=0, the divider ratio is 1, which means that f_{DIV}=f_{ASC}. In general, the fractional divider allows the baud rate to be programmed with much better accuracy than with the two fixed prescaler divider stages.

***Note**: In fractional divider mode, the clock f_{DIV} can have a maximum period jitter of one f_{ASC} clock period.*

Table 1194 Asynchronous baud rate formulas using the fractional input clock divider

FDE	BRS	BG	FDV	Formula
1	–	0 … 8191	1 … 2047	$\text{Baud rate} = \frac{\text{FDV}}{2048} \times \frac{f_{\text{ASC}}}{16 \times (\text{BG} + 1)}$
			0	$\text{Baud rate} = \frac{f_{\text{ASC}}}{16 \times (\text{BG} + 1)}$

BG represents the content of the reload register bit field BG.BR_VALUE, taken as an unsigned 13-bit integer. FDV represents the contents of the fractional divider register bit field FDV.FD_VALUE, taken as an unsigned 11-bit integer.

27.3.11.5.2 Baud rates in synchronous mode

For synchronous operation, the baud rate generator provides a clock f_{BRT} that runs with four times the established baud rate, see following figure.

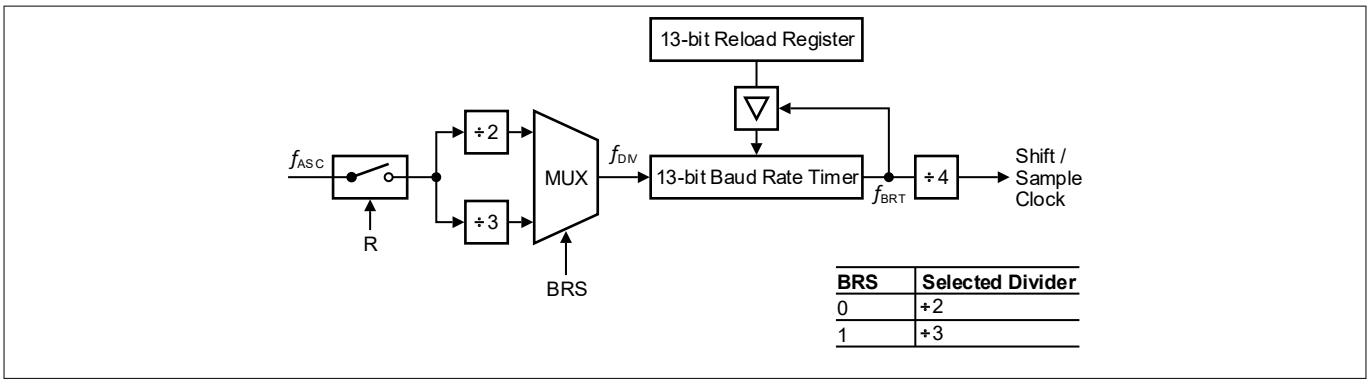


Figure 566 ASC baud rate generator circuitry in synchronous mode

The baud rate for synchronous operation of the serial channel ASC can be determined by the formulas as shown in the following table.

Table 1195 Synchronous baud rate formulas

BRS	BG	Formula
0	0 … 8191	$\text{Baud rate} = \frac{f_{ASC}}{8 \times (BG + 1)}$ $BG = \frac{f_{ASC}}{(8 \times \text{Baud rate}) - 1}$
1		$\text{Baud rate} = \frac{f_{ASC}}{12 \times (BG + 1)}$ $BG = \frac{f_{ASC}}{(12 \times \text{Baud rate}) - 1}$

BG represents the content of the reload register bit field BG.BR_VALUE, taken as an unsigned 13-bit integer.

27.3.11.6 Hardware error detection capabilities

To improve the reliability of serial data exchange, the serial channel ASC provides an error interrupt request flag that indicates the presence of an error and three (selectable) error status flags in register CON that indicate which error has been detected during reception. Upon completion of a reception, the error interrupt request line INTOV.EIR will be activated simultaneously with the receive interrupt request line INTOV.RIR, if one or more of the following conditions are met:

- If the framing error detection enable bit CON.FEN is set and any of the expected Stop bits is not high, the framing error flag CON.FE is set, indicating that the error interrupt request is due to a framing error (Asynchronous operating modes only)
- If the parity error detection enable bit CON.PEN is set in the modes where a Parity bit is received and the parity check on the received data bits proves false, the parity error flag CON.PE is set, indicating that the error interrupt request is due to a parity error (asynchronous operating modes only)
- If the overrun error detection enable bit CON.OEN is set and the last character received was not read out of the receive buffer by software or DMA transfer is complete at the time the reception of a new frame, the overrun error flag CON.OE is set, indicating that the error interrupt request is due to an overrun error (Asynchronous and Synchronous modes)

27.3.11.7 ASC Interrupts

Four interrupt sources are provided for serial channel ASC.

- INTOV.TIR indicates a transmit interrupt
- INTOV.TBIR indicates a transmit buffer interrupt
- INTOV.RIR indicates a receive interrupt
- INTOV.EIR indicates an error interrupt of the serial channel

The interrupt output lines INTOV.TBIR, INTOV.TIR, INTOV.RIR, and INTOV.EIR are synchronized to the PSI5-S kernel and routed by the INPx to the trigger output lines.

The cause of an error interrupt request EIR (framing, parity, overrun error) can be identified by the error status flags CON.FE, CON.PE, and CON.OE.

***Note**: By contrast to the error interrupt request line EIR, the error status flags CON.FE/CON.PE/CON.OE are not reset automatically and must be cleared by software.*

For normal operation (that is, other than error interrupt), the ASC provides three interrupt requests to control data exchange by this serial channel:

- TBIR is activated when data is moved from TBUF to the transmit shift register
- TIR is activated before the last bit of an asynchronous frame is transmitted or after the last bit of a synchronous frame has been transmitted
- RIR is activated when the received frame is moved to RBUF

While the task of the receive interrupt handler is quite clear, the transmitter is serviced by two interrupt handlers. This provides advantages for the servicing software.

For single transfers, it is sufficient to use the transmit interrupt (TIR), which indicates that the previously loaded data has been transmitted, except for the last bit of an asynchronous frame.

For multiple back-to-back transfers, it is necessary to load the following piece of data at least before the last bit of the previous frame has been transmitted. In Asynchronous mode, this leaves just one bit-time for the handler to respond to the transmitter interrupt request; in Synchronous mode, it is entirely impossible.

Using the transmit buffer interrupt (TBIR) to reload transmit data provides the time necessary to transmit a complete frame for the service routine, as TBUF may be reloaded while the previous data is still being transmitted.

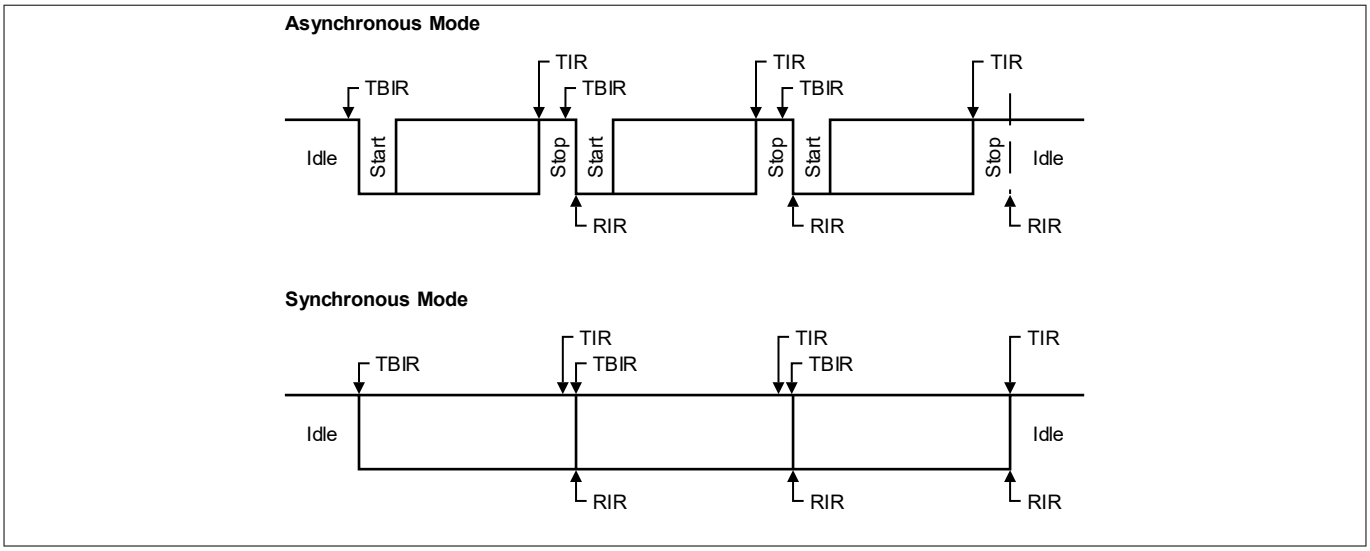


Figure 567 ASC interrupt generation

As shown in the figure above, TBIR is an early trigger for the reload routine, while TIR indicates the completed transmission. Software using handshake should, therefore, rely on TIR at the end of a data block to ensure that all data has been transmitted.

27.3.12 PSI5-S interrupts

8 Interrupt sources are available for the PSI5-S module. For each trigger source of a channel x one interrupt can be selected in register INPx.

1. INTSTATx.RDI indicates a receive data interrupt. It is activated when a received frame is moved to Receive Data Register RDR
2. INTSTATx.RSI indicates a receive frame success interrupt, i.e. non of the interrupts/errors XCRCI, CRCI, TEI, PE, FE, OE, RBI or HDI was detected. The referring interrupt must be enabled in GCR to be considered for RSI. Both RDI and RSI will be issued together in normal use cases where reception is correct
3. INTSTATx.RBI indicates a receive buffer overrun interrupt. It is activated when a new frame is transferred to a Receive Data Register RDR while the old value was still not read by the host (“overwrite”), i.e. the kernel wants to set interrupt RDI and finds RDI already set
4. INTSTATx.TPI indicates a transmit interrupt. It is activated when data is completely moved from a SDR to ASC (INTSTATx.TPI). Thus it can be used to trigger the next write access to SDRx
5. INTSTATx.TPOI indicates a transfer (send) register (SDRx) overrun interrupt. It is set if data is written to SDRx while the referring register is locked (occupied), signaled by SCRx.TPF. The data written is ignored in this case. INTOV.FOI indicates that the output FIFO was overrun

   In addition the protocol error interrupts are available:
6. INTSTATx.CRCI, XCRC. If CRC interrupt is activated, data is to be treated as invalid according to standard v2.1. Note that the wrong data is still available in all buffers. For XCRCI, these frames are always stored in channel 0 frame 1
7. INTSTATx.TEI is issued by the watchdog timer if the timely distance between two frames is too long
8. INTSTATx.CHCI indicates that the last frame expected for a channel is received

The interrupt request or the corresponding interrupt set bit (in register INTSET) can trigger the interrupt generation at the selected interrupt node. The service request pulse is generated independently from the interrupt flag in register INTSTATx. The interrupt flag can be cleared by software by writing to the corresponding bit in register INTCLR.

If more than one interrupt source is connected to the same interrupt node pointer (in register INPx all node pointer point to the same line), the requests are combined to one common line.

27.4 Registers

27.4.1 Register overview - access mode glossary

Table 1196 Register overview - access mode glossary

Keyword	Description
E	Access protection using PROT register PROTE.
SE	Access protection using PROT register PROTSE.
APU-P	Protection group consisting of registers ACCEN_WRA, ACCEN_WRB, ACCEN_RDA, ACCEN_RDB, ACCEN_VM, ACCEN_PRS.
P	Access protection using *APU-P* registers.
BE	Always returns a Bus Error.
U	No access restrictions.
PROT	Access restrictions as defined in the PROT register access rules.

27.4.2 Registers overview - PSI5S (ascending offset address)

Table 1197 Registers overview - PSI5S (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CLC	Clock Control Register	0000_H	P	P, SV, E	Application Reset	5016
OCS	OCDS Control and Status Register	0004_H	P	SV, P	Debug Reset	5017
ID	Module Identification Register	0008_H	P	BE	PowerOn Reset	5018
RST_CTRLA	Reset Control Register A	$000C_H$	P	P, SV, E	Application Reset	5019
RST_CTRLB	Reset Control Register B	0010_H	P	P, SV, E	Application Reset	5019
RST_STAT	Reset Status Register	0014_H	P	BE	Application Reset	5020
PROTE	PROT Register Endinit	0020_H	U	SV, PROT	Application Reset	5021
PROTSE	PROT Register Safe Endinit	0024_H	U	SV, PROT	Application Reset	5022
ACCEN_WRA	Write access enable register A	0040_H	U	SE, SV	Application Reset	5024
ACCEN_WRB	Write access enable register B	0044_H	U	SE, SV	Application Reset	5024
ACCEN_RDA	Read access enable register A	0048_H	U	SE, SV	Application Reset	5025

(table continues...)

Table 1197 (continued) Registers overview - PSI5S (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ACCEN_RDB	Read access enable register B	$004C_H$	U	SE, SV	Application Reset	5025
ACCEN_VM	VM access enable register	0050_H	U	SE, SV	Application Reset	5026
ACCEN_PRS	PRS access enable register	0054_H	U	SE, SV	Application Reset	5026
FDR	Fractional Divider Register	0060_H	P	P	Kernel Reset	5027
FDRT	Fractional Divider Register for Timestamp	0064_H	P	P	Kernel Reset	5028
TSCNTA	Timestamp Count Register A	0068_H	P	P	Kernel Reset	5029
TSCNTB	Timestamp Count Register B	$006C_H$	P	P	Kernel Reset	5030
GCR	Global Control Register	0070_H	P	P	Kernel Reset	5031
NFC	Number of Frames Control Register	0074_H	P	P	Kernel Reset	5033
FCNT	Frame Counter Register	0078_H	P	P	Kernel Reset	5034
IOCR	Input and Output Control Register	$007C_H$	P	P	Kernel Reset	5034
RCRAx	Receiver Control Register Ax	0080_H+x *4	P	P	Kernel Reset	5035
RCRBx	Receiver Control Register Bx	$00A0_H$+x *4	P	P	Kernel Reset	5037
WDTx	Watchdog Timer Register x	$00C0_H$+x *4	P	P	Kernel Reset	5037
TSCRx	Timestamp Capture Register x	$00E0_H$+x *4	P	BE	Kernel Reset	5038
RDS	Receive Status Register	0100_H	P	BE	Kernel Reset	5039
RDR	Receive Data Register	0104_H	P	BE	Kernel Reset	5041
TSM	Timestamp Mirror Register	0108_H	P	BE	Kernel Reset	5042
TAR	Target Address Register	$010C_H$	P	BE	Kernel Reset	5042
BAR	Base Address Register	0110_H	P	P	Kernel Reset	5043
PGCx	Pulse Generation Control Register x	0114_H+x *4	P	P	Kernel Reset	5043
CTVx	Channel Trigger Value Register x	0134_H+x *4	P	P	Kernel Reset	5045
SCRx	Send Control Register x	0154_H+x *4	P	P	Kernel Reset	5045
SDRx	Send Data Register x	0174_H+x *4	P	P	Kernel Reset	5047

(table continues...)

Table 1197 (continued) Registers overview - PSI5S (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CDW	CPU Direct Write Register	0194_H	P	P	Kernel Reset	5048
CON	Control Register	0198_H	P	P	Kernel Reset	5049
BG	Baud Rate Timer/Reload Register	$019C_H$	P	P	Kernel Reset	5051
FDV	Fractional Divider Register	$01A0_H$	P	P	Kernel Reset	5051
FDO	Fractional Divider for Output CLK Register	$01A4_H$	P	P	Kernel Reset	5052
TBUF	Transmit Buffer Register	$01A8_H$	P	P	Kernel Reset	5053
RBUF	Receive Buffer Register	$01AC_H$	P	BE	Kernel Reset	5053
WHBCON	Write Hardware Bits Control Register	$01B0_H$	P	P	Kernel Reset	5054
INTSTATx	Interrupt Status Register x	$01B4_H$+x *4	P	BE	Kernel Reset	5055
INTSETx	Interrupt Set Register x	$01D4_H$+x *4	P	P	Kernel Reset	5058
INTCLRx	Interrupt Clear Register x	$01F4_H$+x *4	P	P	Kernel Reset	5059
INTENx	Interrupt Enable Register x	0214_H+x *4	P	P	Kernel Reset	5060
INPx	Interrupt Node Pointer Register x	0234_H+x *4	P	P	Kernel Reset	5061
INTOV	Interrupt Overview Register	0254_H	P	BE	Kernel Reset	5063
INTSTATG	Interrupt Status Register Global	0258_H	P	BE	Kernel Reset	5064
INTSETG	Interrupt Set Register Global	$025C_H$	P	P	Kernel Reset	5066
INTCLRG	Interrupt Clear Register Global	0260_H	P	P	Kernel Reset	5067
INTENG	Interrupt Enable Register Global	0264_H	P	P	Kernel Reset	5068
INPG	Interrupt Node Pointer Register Global	0268_H	P	P	Kernel Reset	5069

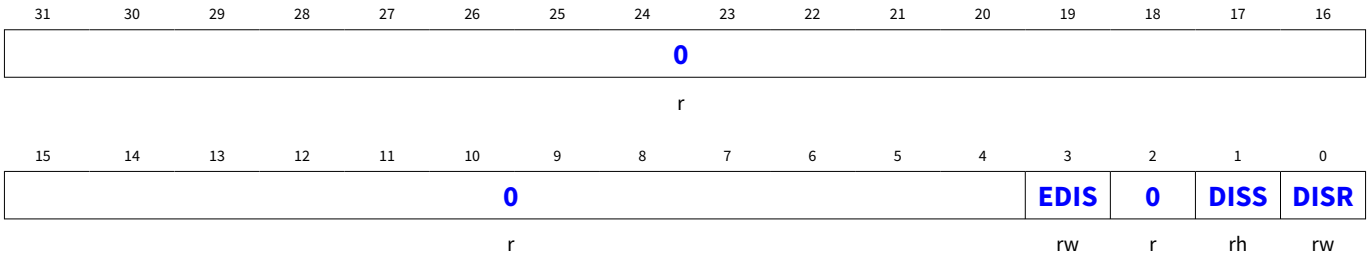
27.4.3 Clock Control Register

The Clock Control Register CLC allows the programmer to adapt the functionality and power consumption of the module to the requirements of the application.

Register CLC controls the module clock signal and the reactivity to the sleep signal.

CLC Offset address: 0000_H

Clock Control Register Application Reset value: $0000\ 0003_H$



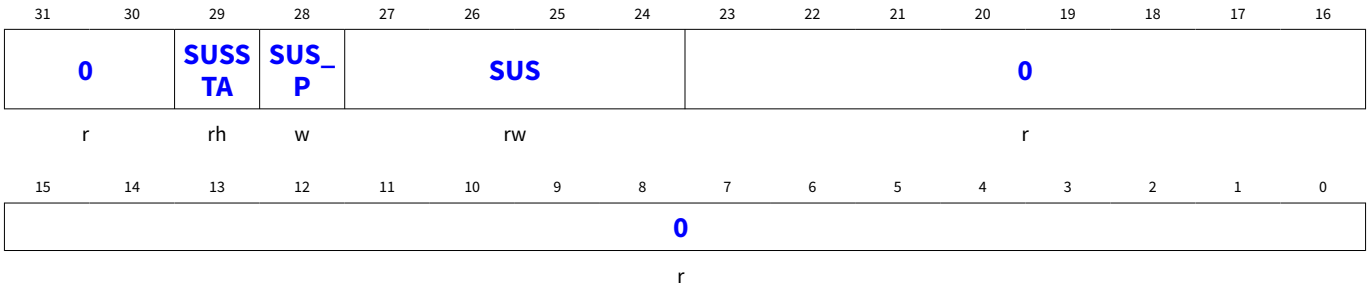
Field	Bits	Type	Description
DISR	0	rw	**Module Disable Request Bit** Used for enable/disable control of the module. 0_B On request: enable the module clock 1_B Off request: stop the module clock
DISS	1	rh	**Module Disable Status Bit** 0_B On: Module clock is enabled 1_B Off: module is not clocked
EDIS	3	rw	**Sleep Mode Enable Control** Used to control the module's reaction to sleep mode. 0_B Sleep mode request is enabled and functional 1_B Module disregards the sleep mode control signal
0	2, 31:4	r	**Reserved** Read as 0; should be written with 0.

27.4.4 OCDS Control and Status Register

The OCDS Control and Status register OCS controls the debug and trace behavior by selecting suspend modes and OTGB trigger sets. When OCDS is disabled the suspend control is ineffective

OCS Offset address: 0004_H

OCDS Control and Status Register Debug Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
SUS	27:24	rw	**OCDS Suspend Control** Controls the sensitivity to the suspend signal coming from the OCDS Trigger Switch (OTGS) Not listed combinations are reserved. 0_H Will not suspend 1_H Hard suspend: Clocks will be disabled immediately 2_H Soft suspend mode Clock will be disabled after kernel acknowledge 3_H Soft suspend mode Kernel clock will not be disabled after acknowledge
SUS_P	28	w	**SUS Write Protection** SUS is only written when SUS_P is 1, otherwise unchanged. Read as 0.
SUSSTA	29	rh	**Suspend State** 0_B Module is not (yet) suspended 1_B Module is suspended
0	23:0, 31:30	r	**Reserved** Read as 0; should be written with 0.

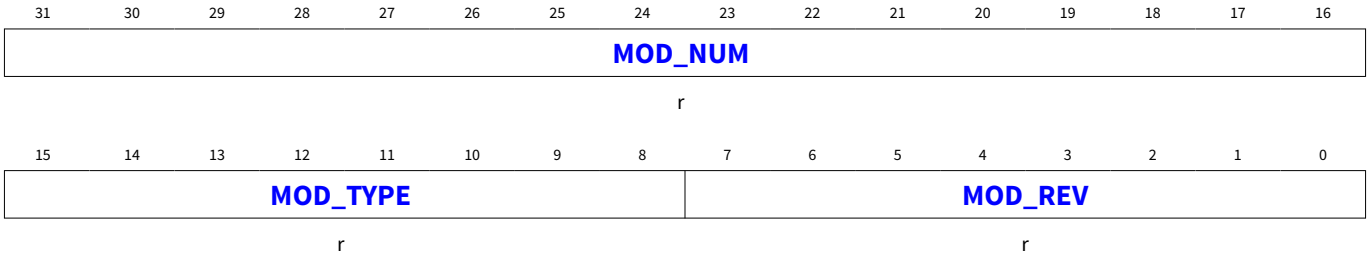
Table 1198 Access mode restrictions of OCS sorted by descending priority

Mode name	Access mode		Description
write 1 to .**SUS_P**	rw	SUS	Set SUS_P during write access
(default)	r	SUS	

27.4.5 Module Identification Register

ID Offset address: 0008_H

Module Identification Register PowerOn Reset value: $00D3\ C001_H$



Field	Bits	Type	Description
MOD_REV	7:0	r	**Module Revision** Indicates the revision number of the implementation.
MOD_TYPE	15:8	r	**Module Type** This internal marker is fixed to $C0_H$.
MOD_NUM	31:16	r	**Module Number** Indicates the module identification number.

27.4.6 Reset Control Register A

RST_CTRLA Offset address: $000C_H$

Reset Control Register A Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				GRSTEN3	GRSTEN2	GRSTEN1	GRSTEN0	0							KRST
r				rw	rw	rw	rw	r							rwh

Field	Bits	Type	Description
KRST	0	rwh	**Kernel Reset** Request a kernel reset. The requested reset is executed if CTRLA.KRST=1 and CTRLB.KRST=1. KRST is cleared after the kernel reset was executed. 0_B No action 1_B A kernel reset was requested
GRSTENx (x=0-3)	x+8	rw	**Enable for Global Module Reset Group x** 0_B Global module reset group x does not have any effect 1_B Global module reset group x results in a kernel reset
0	7:1, 31:12	r	**Reserved** Read as 0; should be written with 0.

27.4.7 Reset Control Register B

RST_CTRLB Offset address: 0010_H

Reset Control Register B Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
STATCLR								0							
w								r							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															KRST
r															rwh

Field	Bits	Type	Description
KRST	0	rwh	**Kernel Reset** Request a kernel reset. The requested reset is executed if CTRLA.KRST=1 and CTRLB.KRST=1. KRST is cleared after the kernel reset was executed. 0_B No action 1_B A kernel reset was requested
STATCLR	31	w	**Kernel Reset Status Clear** Clears all status bits in RST_STAT when 1 is written. Read will return 0. 0_B No action 1_B Write with ´1´ clears bits STAT.GRSTx and bit STAT.KRST .
0	30:1	r	**Reserved** Read as 0; should be written with 0.

27.4.8 Reset Status Register

The reset status register contains the status bits for kernel reset (KRST) and the global module reset groups (GRSTx).

RST_STAT Offset address: 0014_H

Reset Status Register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				GRST3	GRST2	GRST1	GRST0	0							KRST
r				rh	rh	rh	rh	r							rh

Field	Bits	Type	Description
KRST	0	rh	**Kernel Reset Status** Indicates an executed kernel reset. RST_STAT.KRST is set after the execution of a kernel reset in the same clock cycle in which the reset bits are cleared. Clear KRST by setting bit STATCLR in register RST_CTRLB.
GRSTx (x=0-3)	x+8	rh	**Status for Global Module Reset Group x** 0_B Reset was not triggered by Global Reset Group x 1_B Reset was triggered by Global Reset Group x
0	7:1, 31:12	r	**Reserved** Read as 0.

27.4.9 PROT Register Endinit

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner and in some PROT states, the secure master to update the PROT state as described in the PROT mechanism.

The PROTE register controls lock / unlock of the local Endinit (E) protected control registers.

PROTE Offset address: 0020_H

PROT Register Endinit Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OWEN	ODEF	TAGID						PRSEN	PRS			VMEN	VM		
w	rw	rw						rw	rw			rw	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												SWEN	STATE		
r												w	rwh		

Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
VM	18:16	rw	**Virtual machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual machine definition enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.

(table continues...)

(continued)

Field	Bits	Type	Description
PRSEN	23	rw	**Protection set definition enable for PROT owner** 0_B **Disable**: PRS is not part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG-ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; should be written with 0.

Table 1199 Access mode restrictions of PROTE sorted by descending priority

Mode name	Access mode		Description
write 1 to .**SWEN**	rwh	STATE	
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

27.4.10 PROT Register Safe Endinit

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner and in some PROT states, the secure master to update the PROT state as described in the PROT mechanism.

The PROTSE register controls lock / unlock of the local Safe Endinit (SE) protected control registers.

PROTSE Offset address: 0024_H

PROT Register Safe Endinit Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OWEN	ODEF	TAGID						PRSEN	PRS			VMEN	VM		
w	rw	rw						rw	rw			rw	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												SWEN	STATE		
r												w	rwh		

Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
VM	18:16	rw	**Virtual machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual machine definition enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.
PRSEN	23	rw	**Protection set definition enable for PROT owner** 0_B **Disable**: PRS is not part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG-ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; should be written with 0.

Table 1200 Access mode restrictions of PROTSE sorted by descending priority

Mode name	Access mode		Description
write 1 to .**SWEN**	rwh	STATE	
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

27.4.11 Write access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

ACCEN_WRA Offset address: 0040_H

Write access enable register A Application Reset value: $1000\ 0003_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	**EN30**	**EN29**	**EN28**	**EN27**	**EN26**	**EN25**	**EN24**	**EN23**	**EN22**	**EN21**	**EN20**	**EN19**	**EN18**	**EN17**	**EN16**
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	**EN14**	**EN13**	**EN12**	**EN11**	**EN10**	**EN09**	**EN08**	**EN07**	**EN06**	**EN05**	**EN04**	**EN03**	**EN02**	**EN01**	**EN00**
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access

27.4.12 Write access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

ACCEN_WRB Offset address: 0044_H

Write access enable register B Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								**EN39**	**EN38**	**EN37**	**EN36**	**EN35**	**EN34**	**EN33**	**EN32**
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access
0	31:8	r	**Reserved** Read as 0; should be written with 0.

27.4.13 Read access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

ACCEN_RDA Offset address: 0048_H

Read access enable register A Application Reset value: FFFF $FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access

27.4.14 Read access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

ACCEN_RDB Offset address: $004C_H$

Read access enable register B Application Reset value: 0000 $00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access
0	31:8	r	**Reserved** Read as 0; should be written with 0.

27.4.15 VM access enable register

This register defines which virtual machine encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each VMx.

ACCEN_VM Offset address: 0050_H

VM access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for VM ID q** This bit enables read access to the access protected region for transactions with the VM ID q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for VM ID q** This bit enables write access to the access protected region for transactions with the VM ID q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

27.4.16 PRS access enable register

This register defines which protection register set encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each PRSx.

ACCEN_PRS Offset address: 0054_H

PRS access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for PRS q** This bit enables read access to the access protected region for transactions with the PRS q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for PRS q** This bit enables write access to the access protected region for transactions with the PRS q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

27.4.17 Fractional Divider Register

The Fractional Divider Register controls the clock $f_{PS5\text{-}S}$.
This bit field only controls the kernel clock $f_{PSI5\text{-}S}$ and not the sampling clock f_{ASC} .

FDR Offset address: 0060_H

Fractional Divider Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0						RESULT									
r						rh									

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DM		0				STEP									
rw		r				rw									

Field	Bits	Type	Description
STEP	9:0	rw	**Step Value** Reload or addition value for RESULT.

(table continues...)

(continued)

Field	Bits	Type	Description
DM	15:14	rw	**Divider Mode** DM selects normal or fractional divider mode. 00_B Fractional divider is switched off; no output clock is generated. The Reset External Divider signal is 1. RESULT is not updated. 01_B Normal Divider Mode selected. 10_B Fractional Divider Mode selected. 11_B Fractional divider is switched off; no output clock is generated. RESULT is not updated.
RESULT	25:16	rh	**Result Value** Bit field for the addition result.
0	13:10, 31:26	r	**Reserved** Read as 0; should be written with 0.

Related information

Clock generation on page 4977

27.4.18 Fractional Divider Register for Timestamp

The Fractional Divider Register for Timestamp contains the pre divider that defines the time resolution of the Timestamp Registers TSCNTA/B and the sync pulse Time Base. It divides $f_{PSI5\text{-}S}$ by a factor. It contains as well the bits for reset control of the Timestamp Counters.

FDRT Offset address: 0064_H

Fractional Divider Register for Timestamp Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	ECEB	ECEA	ECS			RESULT									
r	rw	rw	rw			rh									

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DM		0			TRIG SEL	STEP									
rw		r			rw	rw									

Field	Bits	Type	Description
STEP	9:0	rw	**Step Value** Reload or addition value for RESULT.
TRIGSEL	10	rw	**Trigger select** Trigger source selection bit 0_B **GTMTRG**: GTM GTM trigger 1_B **eGTMTRG**: eGTM eGTM trigger

(table continues...)

(continued)

Field	Bits	Type	Description
DM	15:14	rw	**Divider Mode** DM selects normal or fractional divider mode. 00_B Fractional divider is switched off; no output clock is generated. 01_B Normal Divider Mode selected. 10_B Fractional Divider Mode selected. 11_B Fractional divider is switched off; no output clock is generated. RESULT is not updated.
RESULT	25:16	rh	**Result Value** Bit field for the addition result.
ECS	28:26	rw	**External Timestamp Clear Source Select** Selects the external trigger line that clears the global Timestamp Counters TSCNTA/B.CTS if this is enabled by ECEA/ECEB. 000_B TRIG0 001_B TRIG1 … 111_B TRIG7
ECEA	29	rw	**External Timestamp Clear Enable A** Enables the external trigger line selected by ECS to clear the global Timestamp Counter TSCNTA.CTS on rising edge of the external trigger. 0_B Disabled 1_B Enabled
ECEB	30	rw	**External Timestamp Clear Enable B** Enables the external trigger line selected by ECS to clear the global Timestamp Counter TSCNTB.CTS on rising edge of the external trigger. 0_B Disabled 1_B Enabled
0	13:11, 31	r	**Reserved** Read as 0; should be written with 0.

Related information

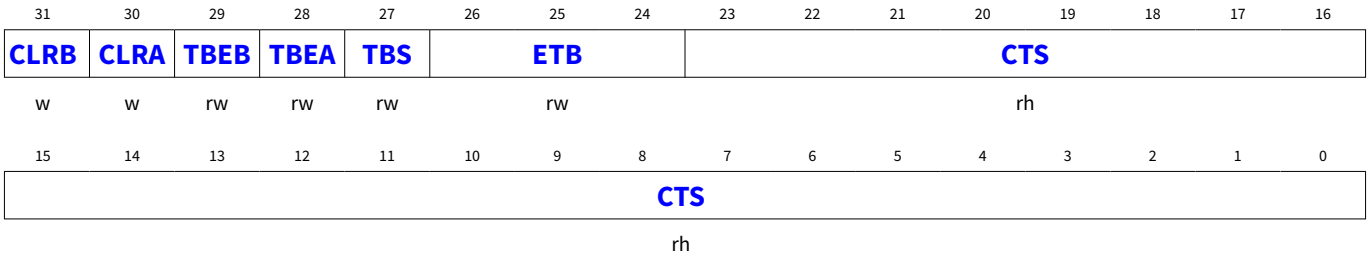
Clock generation on page 4977

27.4.19 Timestamp Count Register A

The Timestamp Counter Register contains read-only information about the current time given in clock cycles of f_{TS} or $f_{GTM/eGTM}$ since last reset while module was clocked. It also contains the control bits for selection of input clock and reset of this counter.

TSCNTA Offset address: 0068_H

Timestamp Count Register A Kernel Reset value: $0000\ 0000_H$



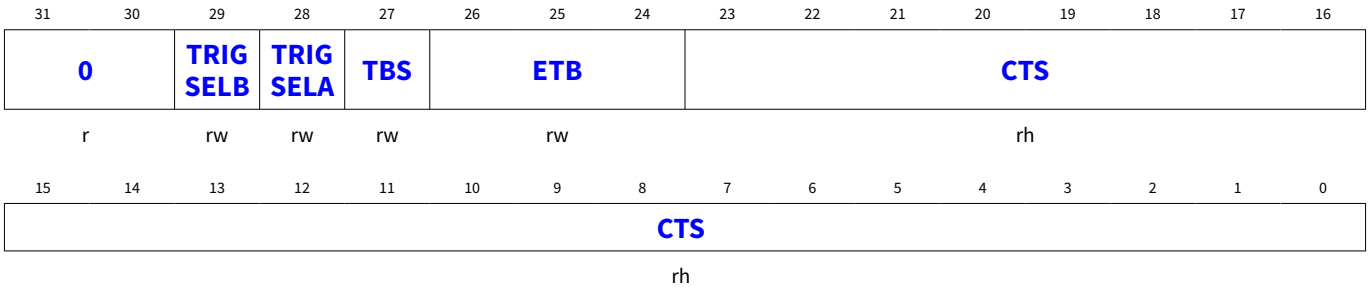
Field	Bits	Type	Description
CTS	23:0	rh	**Current Timestamp for the Module** This bit field shows the current timestamp.
ETB	26:24	rw	**External Time Base Select** Selects the external clock line for counter CTS if TBS is set. 000_B TRIG0 001_B TRIG1 … 111_B TRIG7
TBS	27	rw	**Time Base Select** This bit selects the clock source for CTS 0_B Internal, CTS counts in clock cycles of f_{TS} 1_B External, CTS counts in clock cycles received on external clock line selected by ETB
TBEA	28	rw	**Time Base Enable TSCNTA** This bit starts/stops TSCNTA.CTS 0_B CTS stopped (without clear of CTS) 1_B CTS started (without clear of CTS)
TBEB	29	rw	**Time Base Enable TSCNTB** This bit starts/stops TSCNTB.CTS 0_B CTS stopped (w/o clear of CTS) 1_B CTS started (w/o clear of CTS)
CLRA	30	w	**Clear Timestamp Counter A** This bit clears TSCNTA.CTS. TSCNTA.CTS counts on, starting from 0.
CLRB	31	w	**Clear Timestamp Counter B** This bit clears TSCNTB.CTS. TSCNTB.CTS counts on, starting from 0.

27.4.20 Timestamp Count Register B

The Timestamp Counter Register contains read-only information about the current time given in clock cycles of f_{TS} or $f_{GTM/eGTM}$ since last reset while module was clocked. It also contains the control bits for selection of input clock and reset of this counter.

TSCNTB Offset address: $006C_H$

Timestamp Count Register B Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
CTS	23:0	rh	**Current Timestamp for the Module** This bit field shows the current timestamp.
ETB	26:24	rw	**External Time Base Select** Selects the external clock line for counter CTS. 000_B TRIG0 001_B TRIG1 … 111_B TRIG7
TBS	27	rw	**Time Base Select** This bit selects the clock source for CTS 0_B Internal, CTS counts in clock cycles of f_{TS} 1_B External, CTS counts in clock cycles received on external clock line selected by ETB
TRIGSELA	28	rw	**Trigger select** Trigger source selection bit for TSCNTA 0_B **GTMTRG**: GTM GTM trigger 1_B **eGTMTRG**: eGTM eGTM trigger
TRIGSELB	29	rw	**Trigger select** Trigger source selection bit for TSCNTB 0_B **GTMTRG**: GTM GTM trigger 1_B **eGTMTRG**: eGTM eGTM trigger
0	31:30	r	**Reserved** Read as 0; should be written with 0.

27.4.21 Global Control Register

The Global Control Register defines module wide settings.

The first 8 bits define, which error flags are regarded at RSI. RSI indicates that the referring frame is free of the selected errors. Each of these errors can be selected: CRCI, XCRCI, TEI, PE, FE, OE, RBI, HDI.

GCR Offset address: 0070_H

Global Control Register Kernel Reset value: $0000\ 001F_H$

27 Peripheral Sensor Interface with Serial Interface to PHY (PSI5-S)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ASC	0			IDT				CEN7	CEN6	CEN5	CEN4	CEN3	CEN2	CEN1	CEN0
rw	r			rw				rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ETC7	ETC6	ETC5	ETC4	ETC3	ETC2	ETC1	ETC0	HDI	RBI	OE	FE	PE	TEI	XCRCI	CRCI
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
CRCI	0	rw	**CRCI** If bit is set, CRCI is selected to be checked for triggering RSI interrupt. No RSI is triggered in case of CRCI error
XCRCI	1	rw	**XCRCI** If bit is set, XCRCI is selected to be checked for triggering RSI interrupt. No RSI is triggered in case of XCRCI error
TEI	2	rw	**TEI** If bit is set, TEI is selected to be checked for triggering RSI interrupt. No RSI is triggered in case of TEI error
PE	3	rw	**PE** If bit is set, PE is selected to be checked for triggering RSI interrupt. No RSI is triggered in case of PE error
FE	4	rw	**FE** If bit is set, FE is selected to be checked for triggering RSI interrupt. No RSI is triggered in case of FE error
OE	5	rw	**OE** If bit is set, OE is selected to be checked for triggering RSI interrupt. No RSI is triggered in case of OE error
RBI	6	rw	**RBI** If bit is set, RBI is selected to be checked for triggering RSI interrupt. No RSI is triggered in case of RBI error
HDI	7	rw	**HDI** If bit is set, HDI is selected to be checked for triggering RSI interrupt. No RSI is triggered in case of HDI error
ETCx (x=0-7)	x+8	rw	**Enable Channel Trigger Counter CTVx.CTC** This bit enables CTVx.CTC. The bits ETC0 - x can be set with one write access to synchronously start all counters. This is required for proper sync pulse staggering. If set, CTVx.CTC counts on, starting from its current value. CTVx.CTC can be written only if ETCx is cleared (stopped).
CENx (x=0-7)	x+16	rw	**Enable Channel x** This bit enables PSI5-S channel x. If cleared, all internal state machines of the receiver and the sender are forced to default idle state while all registers can be read and written. Used for configuration of a channel. Frames received for a disabled channel are copied to ChID 0, FID 1 with original IDs.

(table continues...)

(continued)

Field	Bits	Type	Description
IDT	27:24	rw	**Idle Time (GLOBAL VALUE FOR ALL CHANNELS)** determines the number of stop bits in addition to the stop bit of the last UART frame that is required for SOF detection. (IDT-1) idle bit times are allowed/tolerated within one Packet frame. Default is IDT = 0, that is, back to back transfer. 0_H 1 bit time idle … F_H 16 bit time idle
ASC	31	rw	**ASC only Mode** is selected if bit is set. The ASC registers are fully controllable by software via SPB bus. If cleared, the ASC is controlled by the message reassembly unit and the message generation unit. RBUF and TBUF are no longer writable by software and interrupts are handled by the message reassembly block automatically.
0	30:28	r	**Reserved** Read as 0; should be written with 0.

27.4.22 Number of Frames Control Register

This register contains the number of Packet frames expected after a Sync Pulse.
It stores the compare values for the counters in register FCNT.
See FCNT.

NFC Offset address: 0074_H

Number of Frames Control Register Kernel Reset value: $0024\ 9249_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								NF7			NF6			NF5	
r								rw			rw			rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NF5		NF4		NF3			NF2			NF1			NF0		
rw		rw		rw			rw			rw			rw		

Field	Bits	Type	Description
NFx (x=0-7)	3*x+2:3*x	rw	**Number of expected frames on Channel x** 000_B reserved, do not use. Reads back 1. 001_B 1 … 110_B 6 111_B reserved, do not use. Reads back 6.
0	31:24	r	**Reserved** Read as 0; should be written with 0.

27.4.23 Frame Counter Register

This register contains the number of Packet frames actually received after the last sync pulse. A sync pulse on channel x will clear FCx.

If FCNT.FCx equals NFC.NFx, interrupt CHCI is issued and FCNT.FCx will roll over to '1' with the reception of the next recoverable message on channel x. The roll over happens in any case, even if no sync pulse clears FCx after CHCI.

If RCRAx.WDMS is set, NFC and FCNT affect the operation of the watchdog timer. See Watchdog Timer chapter.

NFC and FCNT also affect the generation of interrupt CHCI. See Interrupts for DMA support chapter.

FCNT Offset address: 0078_H

Frame Counter Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
NFCLR7	NFCLR6	NFCLR5	NFCLR4	NFCLR3	NFCLR2	NFCLR1	NFCLR0	FC7			FC6			FC5	
w	w	w	w	w	w	w	w	rh			rh			rh	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FC5	FC4			FC3			FC2			FC1			FC0		
rh	rh			rh			rh			rh			rh		

Field	Bits	Type	Description
FCx (x=0-7)	3*x+2:3*x	rh	**Frame Counter for Channel x** Contains the number of received frames on Channel x. Copied to RDR.FID if RCRAx.FIDS is set. 000_B 0 (after reset, sync pulse(if WDMS is set only) or setting NFCLRx) 001_B 1 ... 110_B 6 111_B not valid
NFCLRx (x=0-7)	x+24	w	**Clear Number of Frame Counter for Channel x** Clears the referring counter FCNT.FCx. Intended for use during recovery from TEI. If set while a frame is being received, this action results in FCx = '1'. Thus FCx will never be '0' when RDI/RSI signal a new receive frame. Use with care.

Related information

Watchdog timers on page 4980

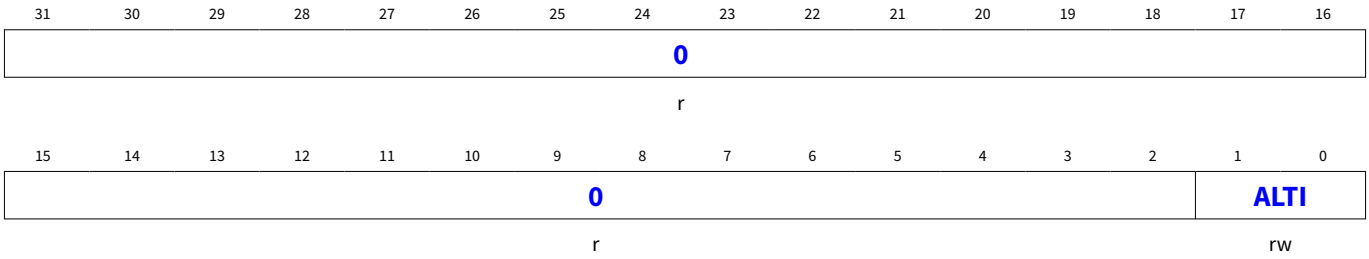
Interrupts for DMA support on page 4996

27.4.24 Input and Output Control Register

The Input and Output Control Register IOCR determines for the PSI5-S, the alternate input for the receiver

IOCR Offset address: $007C_H$

Input and Output Control Register Kernel Reset value: $0000\ 0000_H$



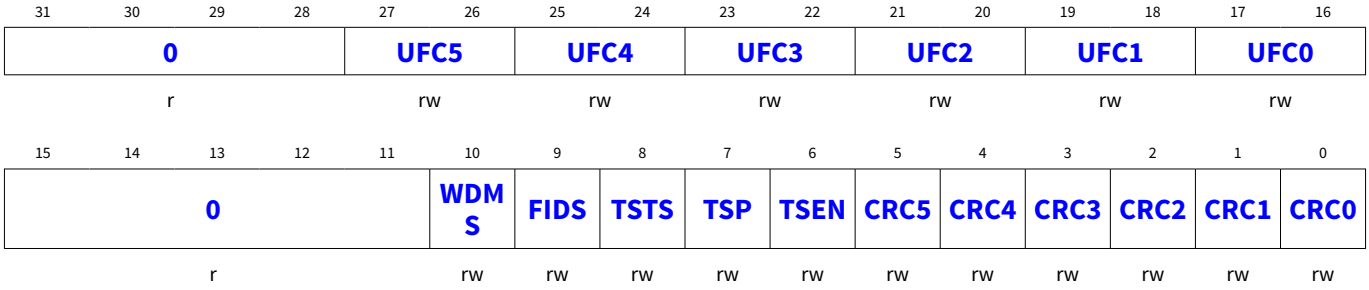
Field	Bits	Type	Description
ALTI	1:0	rw	**Alternate Input Select** Selects the alternate input for RX of the ASC: 00_B Alternate Input 0 selected … 11_B Alternate Input 3 selected
0	31:2	r	**Reserved** Read as 0; should be written with 0.

27.4.25 Receiver Control Register Ax

The Receiver Control Registers RCRAx contain control bits/bit fields that are related to the PSI5-S receiver operation. It enables the channel and determines the use of CRC or parity. GCR.CENx must be clear for write access.

RCRAx (x=0-7) Offset address: 0080_H+x*4

Receiver Control Register Ax Kernel Reset value: 0000 0000_H



Field	Bits	Type	Description
CRCy (y=0-5)	y	rw	**CRC or Parity Selection y** If set, a 3 bit CRC checksum is expected for the PSI5-S channel x in slot y. Else, 1 bit Parity is assumed. This bit field is looked up before potential modification of FID according to FIDS. 0_B 1 Parity Bit is configured 1_B 3 CRC bits are configured
TSEN	6	rw	**Timestamp Enable** Enables the timestamping for channel x 0_B Off TSCRx and thus TSM are forced to 0x0000 0000 1_B On, see TSP and TSTS bit fields

(table continues...)

(continued)

Field	Bits	Type	Description
TSP	7	rw	**Timestamp Select** For non-recoverable Packets (stored in ChID ‘0’, FID ’1’ with original IDs); TSCNTA is captured in TSM and not in TSCR0 (independent from TSP). 0_B TSCNTA.CTS is captured in TSCRx 1_B TSCNTB.CTS is captured in TSCRx
TSTS	8	rw	**Timestamp Trigger Select** 0_B On sync pulse. TSCRx is updated on sync pulse. TSM is updated from TSCRx on Packet Frame reception after ChID extraction. For non-recoverable Packets (stored in ChID ‘0’, FID ’1’ with original IDs); TSCR0 is not updated and TSM is updated with the current value of TSCNTA (independent from TSP). This happens at the time the FSM assumes the Packet Frame to be non-recoverable. This allows following good packets to be timestamped correctly. 1_B On any Frame. TSCRx and TSM are updated simultaneously with the current value of TSCNTA/B depending from RCRAx.TSP on Packet Frame reception after ChID extraction. (RDI) This allows software to read the time of reception on the referring channel from TSCRx. For non-recoverable Packets (stored in ChID ‘0’, FID ’1’ with original IDs), TSM is updated with the current value of TSCNTA (independent from TSP and TSEN). This happens at the time the FSM assumes the Packet Frame to be non-recoverable. TSCR0 is updated only if RCRA0.TSTS is set (‘1’)
FIDS	9	rw	**Frame ID Select** 0_B Frame ID is updated from Packet Frame Header (sync mode). Channel 0 should have FIDS cleared. This avoids that the module overwrites non-recoverable messages with transceiver messages. Note that non-recoverable messages are always stored in ChID ‘0’, FID ’1’ with original IDs. 1_B Frame ID is a rolling number 0-5 copied from FCNT.FC(x-1) (async mode) non-recoverable messages are still stored in ChID ‘0’, FID ’1’ with original IDs.
WDMS	10	rw	**Watchdog Timer Mode Select** 0_B Watchdog Timer is restarted on reception of each recoverable frame on Channel x (async mode). 1_B Watchdog Timer is restarted on sync pulse and stopped at reception of the last frame configured in NFC.NFx.(sync mode)
UFCy (y=0-5)	2*y+17:2*y +16	rw	**UART frame Count per Packet Frame in Slot y** This bit field defines the number of UART frames per Packet frame that are expected for Slot y. This bit field is looked up before potential modification of FID according to FIDS. 00_B 3 UART frames 01_B 4 UART frames 10_B 5 UART frames 11_B 6 UART frames
0	15:11, 31:28	r	**Reserved** Read as 0; should be written with 0.

27.4.26 Receiver Control Register Bx

The Receiver Control Registers RCRBx configures the number of payload bits and implicitly the number of UART frames (Bytes) each of the up to 6 slots in channel x. GCR.CENx must be clear for write access.

RCRBx (x=0-7) Offset address: $00A0_H$+x*4

Receiver Control Register Bx Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		PDL5					PDL4					PDL3			
r		rw					rw					rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PDL3		PDL2				PDL1					PDL0				
rw		rw				rw					rw				

Field	Bits	Type	Description
PDLy (y=0-5)	5*y+4:5*y	rw	**Payload Data Length y** PDL determines the number of bits in a Packet frame for Slot y. It also determines the position of the CRC/Parity bit. For exmaple, 8 defines 8 data bits on position [7:0]. On bit position 8 the CRC/Parity is located. See Packet frames received from PHY chapter. If PDLy is cleared (‘0’) no frame is expected for this slot. Packet frames received for a slot with PDL = ‘0’ are copied to ChID 0, FID 1 with original IDs without further processing. This bit field is looked up before potential modification of FID according to RCRAx.FIDS 00_H No Frame expected 01_H 8 bits for 1 … 08_H 8 bits for 8 09_H 9 bits … $1B_H$ 27 bits $1C_H$ 28 bits for 28 … $1F_H$ 28 bits for 31
0	31:30	r	**Reserved** Read as 0; should be written with 0.

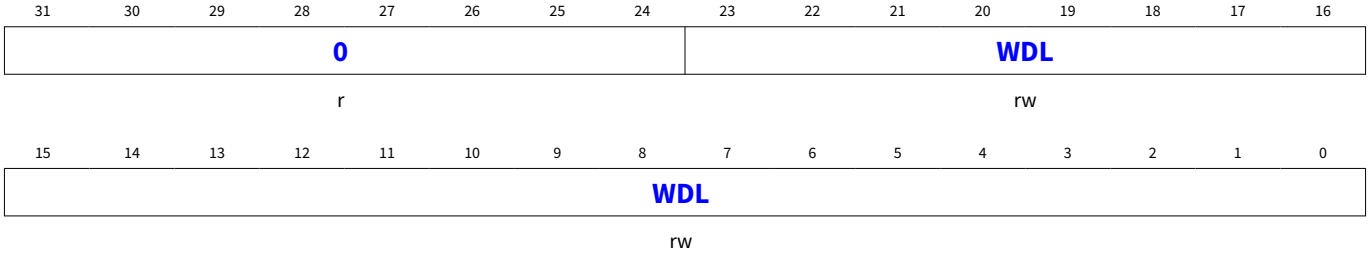
Related information

Packet frames received from PHY on page 4970

27.4.27 Watchdog Timer Register x

WDTx (x=0-7) Offset address: $00C0_H$+x*4

Watchdog Timer Register x Kernel Reset value: $0000\ 0000_H$



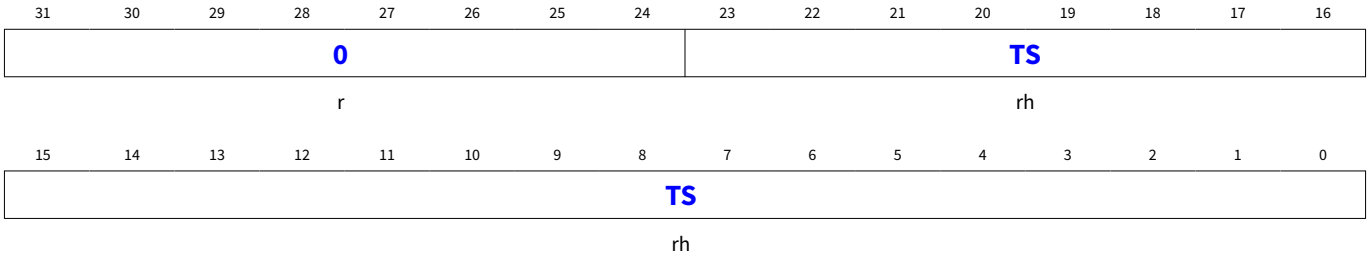
Field	Bits	Type	Description
WDL	23:0	rw	**Watchdog Timer Limit** for channel x. If no watchdog is needed, WDL is cleared, the internal watchdog timer is stopped and no check is performed. GCR.CENx must be clear for write access.
0	31:24	r	**Reserved** Read as 0; should be written with 0.

27.4.28 Timestamp Capture Register x

If RCRAx.TSEN is set, the timestamp is captured in this register each time a sync pulse is sent from the ASC FIFO to the ASC TX buffer for channel x. RCRAx.TSP selects, if TSCNTA or B is used.

TSCRx (x=0-7) Offset address: $00E0_H+x*4$

Timestamp Capture Register x Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
TS	23:0	rh	**Timestamp** of the last sync pulse sent for channel x.
0	31:24	r	**Reserved** Read as 0; should be written with 0.

27.4.29 Receive Status Register

The Receive Status Register RDS shows the status of a received data frame.

The internal receive buffer is always cleared (0x0000 0000$_H$) at each frame start. Thus unused bits are always read as zero.

The bits CRCI, XCRCI, HDI, PE, FE, OE, TEI and RBI reflect, wether the conditions are met to issue an interrupt signal and set the referring interrupt status bit in INTSTAT or CON (ASC sub module) for the latest frame. Thus it is independent from the old status of the referring sticky bit in INTSTAT or CON before latest frame reception.

RDS Offset address: 0100$_H$

Receive Status Register Kernel Reset value: 0000 0000$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PFC				AFC			CID			FID			RBI	TEI	OE
rh				rh			rh			rh			rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FE	PE	HDI	ERR1	ERR0	CRCI	CRC2	CRC1	CRC0	XCRCI	XCRC5	XCRC4	XCRC3	XCRC2	XCRC1	XCRC0
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
XCRCy (y=0-5)	y	rh	**XCRC** CRC of last Packet frame. XCRC0 is on bit position 0.
XCRCI	6	rh	**XCRC Error Flag** This bit is set if the CRC check on the enveloping Packet frame fails including the case where XCRC check can not be performed (non-recoverable frames). Refer to Packet Frames received from PHY chapter 0_B correct XCRC 1_B wrong XCRC
CRCy (y=0-2)	y+7	rh	**CRC** of last frame. CRC0 / Parity is on bit position 7. If Parity is used, CRC1/2 are always 0.
CRCI	10	rh	**CRC Error Flag** This bit is set if the CRC or Parity check on the transported PSI5 frame fails. 0_B correct CRC/Parity 1_B wrong CRC/Parity
ERR0	11	rh	**Error signalling Flag 0** This bit represents the status of the error signalling flag Err0 in the enveloping Packet frame.
ERR1	12	rh	**Error signalling Flag 1** This bit represents the status of the error signalling flag Err1 in the enveloping Packet frame.

(table continues...)

(continued)

Field	Bits	Type	Description
HDI	13	rh	**Header Error Signalled Flag** This bit is set if at least one of the error signalling flags in the enveloping Packet Frame Err0 and Err1 is set. 0_B (Err0 OR Err1) = false (0) 1_B (Err0 OR Err1) = true (1)
PE	14	rh	**ASC Parity Error Flag** This bit is set if the error flag signalling a parity error was set during reception of one of the ASC bytes transporting this PSI5 frame.
FE	15	rh	**ASC Framing Error Flag** This bit is set if the error flag signaling a framing error was set during reception of one of the ASC bytes transporting this PSI5 frame.
OE	16	rh	**ASC Overrun Error Flag** This bit is set if the error flag signaling an overrun error was set during reception of one of the ASC bytes transporting this PSI5 frame.
TEI	17	rh	**Time Error Flag** This bit is set if the watchdog timer expired. Depending from RCRAx.WDMS either the distance between two RDIs is longer than specified in WDL or it expired without reception of CHCI in time. That is, the time from issuing the sync pulse to reception of the last expected frame configured in NFC.NFx was too long. If WDMS is in synchronous mode, all frames after TEI have TEI set in RDS until either CHCI is issued or INSTATx.TEI is cleared by software. RDS.TEI flag is independently from INTENx.TEI. 0_B no error 1_B error
RBI	18	rh	**Receive Buffer Overflow Flag** This bit is set after a frame has been received while the old one was not read from RDR. I.e. the kernel wants to set interrupt RDI and finds RDI already set. The old data is overwritten by the new data. 0_B No overflow 1_B Overflow
FID	21:19	rh	**Frame ID (Frame Number)** See bit RCRAx.FIDS for actual content. 000_B Slot0 … 101_B Slot5 110_B not valid, frame is copied to ChID 0, FID 1 with original IDs 111_B not valid, frame is copied to ChID 0, FID 1 with original IDs
CID	24:22	rh	**Channel ID (Channel Number)** 000_B channel 0 001_B channel 1 … 111_B channel 7

(table continues...)

(continued)

Field	Bits	Type	Description
AFC	27:25	rh	**Actual UART frame Count** This bit field shows the number of UART frames actually received. This is used to further analyze non-recoverable frames by software or during debugging.
PFC	31:28	rh	**Packet Frame Count** For data consistency, RDR, RDS and TSM are tagged with a Packet Frame count value. For each channel there is a separate internal packet counter. This allows to read a triplet of RDR, RDS and TSM from the system memory without looking at interrupts. If the PFC is identical all three values belong together. Otherwise they need to be read again until PFC is identical.

Related information

Packet frames received from PHY on page 4970

27.4.30 Receive Data Register

The Receive Data Register RDR shows the data content of a received data frame. If Messaging bits are used, they are located on bit position [1:0].

The internal receive buffer is always cleared (0x0000 0000$_H$) at each frame start. Thus unused bits are always read as zero.

RDR Offset address: 0104$_H$

Receive Data Register Kernel Reset value: 0000 0000$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PFC				RD27	RD26	RD25	RD24	RD23	RD22	RD21	RD20	RD19	RD18	RD17	RD16
rh				rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RD15	RD14	RD13	RD12	RD11	RD10	RD9	RD8	RD7	RD6	RD5	RD4	RD3	RD2	RD1	RD0
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

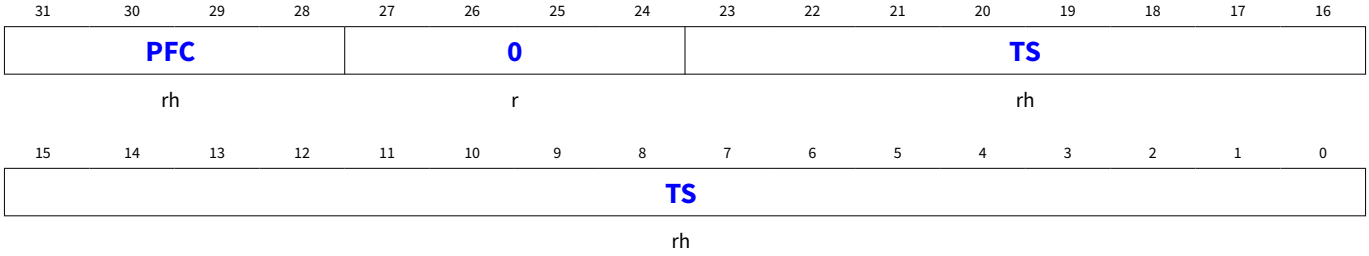
Field	Bits	Type	Description
RDy (y=0-27)	y	rh	**PSI5 Receive Data** of last frame. D0 is on bit position 0.
PFC	31:28	rh	**Packet Frame Count** For data consistency, RDR, RDS and TSM are tagged with a Packet Frame count value. For each channel there is a separate internal packet counter. This allows to read a triplet of RDR, RDS and TSM from the system memory without looking at interrupts. If the PFC is identical all three values belong together. Otherwise they need to be read again until PFC is identical.

27.4.31 Timestamp Mirror Register

Each time a new frame is received, the value from TSCRx for the referring channel is mirrored here. This register is updated, after XCRC was checked ok, as the referring channel number can not be securely determined before.

TSM Offset address: 0108_H

Timestamp Mirror Register Kernel Reset value: $0000\ 0000_H$



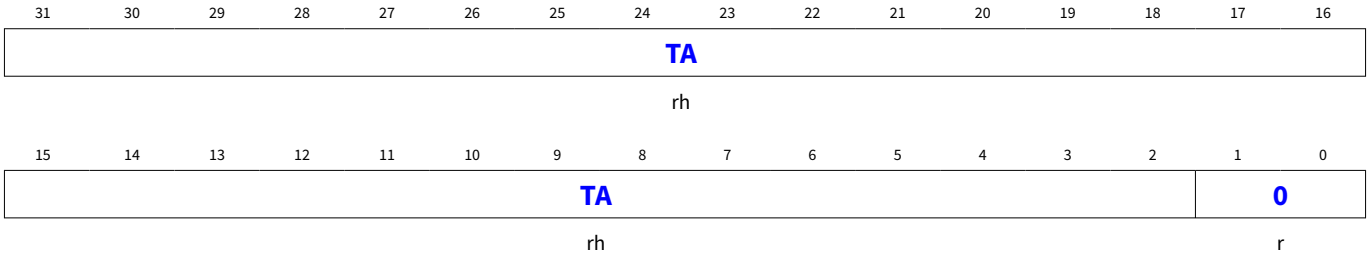
Field	Bits	Type	Description
TS	23:0	rh	**Timestamp** of the last sync pulse sent on channel x.
PFC	31:28	rh	**Packet Frame Count** For data consistency, RDR, RDS and TSM are tagged with a Packet Frame count value. For each channel there is a separate internal packet counter. This allows to read a triplet of RDR, RDS and TSM from the system memory without looking at interrupts. If the PFC is identical all three values belong together. Otherwise they need to be read again until PFC is identical.
0	27:24	r	**Reserved** Read as 0; should be written with 0.

27.4.32 Target Address Register

TAR contains the address that is to be copied by the first DMA into the target address register of the second DMA. This is required for the use case with 2 DMAs. See DMA Support chapter.

TAR Offset address: $010C_H$

Target Address Register Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
TA	31:2	rh	**Target Address** Contains the upper 30 bit of the target address for the next DMA transfer. The 32 bit target address must be word aligned. Thus the 2 LSBs are fixed to 0. It is updated each time a new Packet Frame is received completely.
0	1:0	r	**Reserved** Read as 0; should be written with 0.

Related information

DMA support on page 4989

27.4.33 Base Address Register

BAR contains the address in the system memory where the PSI5-S buffer for all channels is built up.

BAR must be configured by the application for an address space that is big enough for all channels that are enabled. TAR will wrap around after the upper limit (0xFFFF FFFF) if adding the offsets (based on ChID and FID) exceeds this limit.

BAR Offset address: 0110_H

Base Address Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BA															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BA														0	
rw														r	

Field	Bits	Type	Description
BA	31:2	rw	**Base Address** Contains the upper 30 bits of the base address for the DMA transfers. The 32 bit base address must be word aligned. Thus the 2 LSBs are fixed to 0.
0	1:0	r	**Reserved** Read as 0; should be written with 0.

27.4.34 Pulse Generation Control Register x

The Pulse Generation Control Register PGC contains control data for the sync pulse generation. It contains as well the trigger control bits required for sending data from the PSI5-S module to the sensor / external PHY.

PGCx (x=0-7) Offset address: 0114_H+x*4

Pulse Generation Control Register x Kernel Reset value: $0000\ 0000_H$

27 Peripheral Sensor Interface with Serial Interface to PHY (PSI5-S)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								ETE	ETS			PTE	ETB		
r								rw	rw			rw	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TBS	0	TRIG SEL	ATXCMD					0			TXCMD				
rw	r	rw	rw					r			rw				

Field	Bits	Type	Description
TXCMD	4:0	rw	**TX Command** Defines the value that is copied to the ASC FIFO for coding a ‘0’.
ATXCMD	12:8	rw	**Alternate TX Command** Defines the value that is copied to the ASC FIFO for the alternate pulse width, that is, for coding a ‘1’.
TRIGSEL	13	rw	**Trigger select** Trigger source selection bit 0_B **GTMTRG**: GTM GTM trigger 1_B **eGTMTRG**: eGTM eGTM trigger
TBS	15	rw	**Time Base Select** This bit selects the clock source for CTVx 0_B Internal, CTVx.CTC counts in clock cycles of f_{TS} 1_B External, CTVx.CTC counts in clock cycles of $f_{GTM/eGTM}$ according to the setting of bit ETB.
ETB	18:16	rw	**External Time Base Select** Selects the external clock line for counter CTVx.CTC. 000_B TRIG0 001_B TRIG1 … 111_B TRIG7
PTE	19	rw	**Periodic Trigger Enable** Periodic trigger is defined by CTVx.CTC. Should be 0 if ETE is set. 0_B Disabled 1_B Enabled
ETS	22:20	rw	**External Trigger Select** Selects the external trigger line for pulse generation (for example: angle synchronous). 000_B TRIG0 001_B TRIG1 … 111_B TRIG7

(table continues...)

(continued)

Field	Bits	Type	Description
ETE	23	rw	**External Trigger Enable** Angle synchronous trigger, external line is selected by ETS. Should be 0 if PTE is set. 0_B Disabled 1_B Enabled
0	7:5, 14, 31:24	r	**Reserved** Read as 0; should be written with 0.

27.4.35 Channel Trigger Value Register x

CTV contains the value that determines the period of periodic triggers for each channel. It contains as well the counter that can be initialized with an offset so that the phase of harmonic trigger frequencies on different channels can be staggered.

CTVx (x=0-7) Offset address: 0134_H+x*4

Channel Trigger Value Register x Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CTC															
rwh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CTV															
rw															

Field	Bits	Type	Description
CTV	15:0	rw	**Channel Trigger Value CTV** Contains the compare value (exact match) of Channel Trigger Counter CTC at which a sync pulse is triggered for channel x and the counter CTC is cleared. If cleared, CTC is stopped and no pulse triggered.
CTC	31:16	rwh	**Channel Trigger Counter** This bit field allows to read the current counter value of the reset timer cell CTVx. If GCR.ETCx is cleared, CTC can be written.

27.4.36 Send Control Register x

The Send Control Register SCRx contains control data bits required for sending data from the PSI5-S module to the sensor / external PHY.

Data is collected in SDRx and the control bits are collected in SCRx.

The send control unit is build to transmit complete ECU to sensor frames.

SCRx (x=0-7) Offset address: 0154_H+x*4

Send Control Register x Kernel Reset value: $0000\ 0000_H$

27 Peripheral Sensor Interface with Serial Interface to PHY (PSI5-S)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0					TPF	0		STA	CRC	0					
r					r	r		rw	rw	r					

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	FLUS	0					BSC	EPS		0	PLL				
r	w	r					rw	rw		r	rw				

Field	Bits	Type	Description
PLL	4:0	rw	**Payload Length of Registers SDRx** Defines the length that is taken into account. PLL needs to written before SDRx is used for proper operation. 00_H length is 6 for 0 … 06_H length is 6 for 6 07_H length is 7 … 17_H length is 23 18_H length is 24 for 24 … $1F_H$ length is 24 for 31
EPS	7:6	rw	**Enhanced Protocol Selection** EPS[0] controls the Bit Format and MSB Fill bit. (0: Tooth Gap, MSB Fill 1) (1: Pulse Width method, MSB Fill 0) EPS[1] controls the Frame Format (0: Frame Format 1 - 3) (1: Frame Format 4) 00_B Tooth Gap method, ‘1’ for filling the shift register from MSB; Frame format 1 -3 (3 bit start sequence, 3 bit stuffing distance, ‘1’ for stuffing, 3 bit CRC). 01_B Pulse Width method, ‘0’ for filling the shift register from MSB; Frame format 1 -3 (3 bit start sequence, 3 bit stuffing distance, ‘1’ for stuffing, 3 bit CRC). 10_B Reserved, do not use 11_B Pulse Width method, ‘0’ for filling the shift register from MSB; Frame Format 4 (9 bit start sequence, 6 bit stuffing distance, ‘0’ for stuffing, 6 bit CRC).
BSC	8	rw	**Bit Stuffing Control** Depending from bit EPS[1] after 3 bits a ‘1’ is inserted (Frame Format 1 -3) or after 6 bits a ‘0’ is inserted (Frame Format 4) 0_B No automatic bit stuffing 1_B Automatic bit stuffing is enabled.
FLUS	14	w	**Flush SDRx** Setting this bit stops shifting out the data in SDRx, the start sequence or CRC if any and clears the referring FSMs and counters. If EPS[0] is 0, SDRx is flushed by setting all bits of EPS[0] to 1 and clears TPF. TPIx is issued at the end of successful flushing. Reads always as zero.

(table continues...)

(continued)

Field	Bits	Type	Description
CRC	22	rw	**CRC Generation Control** 0_B CRC is not generated automatically, it still can be written by software together with the data (for example, to test the remote CRC) 1_B CRC is automatically generated by hardware (according to EPS[1])
STA	23	rw	**Start Sequence Generation Control** 0_B no start sequence generated, shifting out payload starts at bit 0! 1_B automatically generated by hardware (according to EPS[1]) shifting out payload starts after 3/9 bits (EPS[1] = 0/1)
TPF	26	r	**Transmit in Progress Flag** If set, data preparation and transmission is in progress: start sequence or CRC or stuffing bits or data from SDRx are being transferred. It is cleared automatically after preparation and transmitting is finished. If set, write access to SDRx will not change any data and issue TPOI.
0	5, 13:9, 21:15, 25:24, 31:27	r	**Reserved** Read as 0; should be written with 0.

27.4.37 Send Data Register x

The Send Data Register SDRx shows the data content of a data frame to be sent.
GCR.CENx must be set for write access. For initialization SCRx.FLUS can be used.

SDRx (x=0-7) Offset address: 0174_H+x*4

Send Data Register x Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								SD23	SD22	SD21	SD20	SD19	SD18	SD17	SD16
r								rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SD15	SD14	SD13	SD12	SD11	SD10	SD9	SD8	SD7	SD6	SD5	SD4	SD3	SD2	SD1	SD0
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
SDy (y=0-23)	y	rwh	**SDy** Send data of next ECU to Sensor frame. Each time a bit is shifted out, the whole content is shifted right by 1 position and the MSB is filled with '1' or '0' depending from EPS[0]. This allows to read back SDR and determine the status of the shift process by software. The unused MSBs (bit position SCRx.PLL and higher) must be written with '0' if EPS[0] is set (Pulse width method) and with '1' if EPS[0] is cleared (Tooth Gap method). This is required, as the respective standard value is shifted into the register from MSB during shift out operation. SDRx will be filled with this value after shift out process.
0	31:24	r	**Reserved** Read as 0.

27.4.38 CPU Direct Write Register

The CPU Direct Write Register CDW allows the CPU to insert a command into the ASC FIFO. Note that Bits SD[7:5] select the channel for which the timestamp is captured. These bits are copied to the UART frame Bit positions [2:0] (ChID) while SD[4:0] are copied to UART frame Bit positions [7:3] (Command). See Figure PSI5-S UART frames transmitted to the PHY .

CDW Offset address: 0194_H

CPU Direct Write Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0							TSI	SD7	SD6	SD5	SD4	SD3	SD2	SD1	SD0
r							rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
SDy (y=0-7)	y	rw	**SDy** Send data of next ECU to Sensor frame.
TSI	8	rw	**Trigger Pulse Indicator** If this bit is set, a sync pulse is assumed and thus a timestamp is captured when the command leaves the FIFO and is written to ASC TX Register. Bits SD[7:5] select the channel for which the timestamp is captured. RCRAx.TSTS must be cleared. If set, the timestamp is captured on Packet frame reception only.
0	31:9	r	**Reserved** Read as 0.

Related information

PSI5-S UART frames transmitted to PHY on page 4974

27.4.39 Control Register

The serial operating modes of the ASC module are controlled by its Control Register CON. This register contains control bits for mode and error check selection, and status flags for error identification.

CON Offset address: 0198_H

Control Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0			ODDTX	0										MTX	
r			rw	r										rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	LB	BRS	ODD	FDE	OE	FE	PE	OEN	FEN	PEN	REN	STP	M		
rw	rw	rw	rw	rw	rh	rh	rh	rw	rw	rw	rh	rw	rw		

Field	Bits	Type	Description
M	2:0	rw	**Mode Selection** 000_B 8-bit data Synchronous Mode. MTX needs to be cleared as well for proper operation. 001_B 8-bit data Asynchronous Mode 010_B Reserved. Do not use this combination. 011_B 7-bit data + parity Asynchronous Mode 100_B 9-bit data Asynchronous Mode 101_B 8-bit data + wake up bit Asynchronous Mode 110_B Reserved. Do not use this combination. 111_B 8-bit data + parity Asynchronous Mode
STP	3	rw	**Number of Stop Bit Selection** 0_B One stop bit 1_B Two stop bits
REN	4	rh	**Receiver Enable Control** Bit is reset by hardware after reception of a Byte in Synchronous Mode. 0_B Receiver disabled 1_B Receiver enabled
PEN	5	rw	**Parity Check Enable (asynchronous mode only)** 0_B Ignore parity 1_B Check parity
FEN	6	rw	**Framing Check Enable (asynchronous mode only)** 0_B Ignore framing errors 1_B Check framing errors
OEN	7	rw	**Overrun Check Enable** 0_B Ignore overrun errors 1_B Check overrun errors
PE	8	rh	**ASC Parity Error Flag** Set by hardware on a parity error (PEN = 1). Must be reset by software.

(table continues...)

(continued)

Field	Bits	Type	Description
FE	9	rh	**ASC Framing Error Flag** Set by hardware on a framing error (FEN = 1). Must be reset by software.
OE	10	rh	**ASC Overrun Error Flag** Set by hardware on an overrun error (OEN = 1). Must be reset by software.
FDE	11	rw	**Fractional Divider Enable** FDE is don't care and assumed '0' in Synchr. Mode. 0_B Fractional divider disabled 1_B Fractional divider is enabled and used as prescaler for baud rate timer (bit BRS is don't care)
ODD	12	rw	**Parity Selection** Controls the parity selection for Rx path. (Refer ODDTX for Tx path) 0_B Even parity selected (parity bit = 1 on odd number of 1s in data, parity bit = 0 on even number of 1s in data) 1_B Odd parity selected (parity bit = 1 on even number of 1s in data, parity bit = 0 on odd number of 1s in data)
BRS	13	rw	**Baud Rate Selection** BRS is don't care if FDE = 1 (fractional divider enabled) FDE is don't care and assumed '0' in Synchr. Mode. 0_B Baud rate timer prescaler divide-by-2 selected 1_B Baud rate timer prescaler divide-by-3 selected
LB	14	rw	**Loop-back Mode Enable** 0_B Loop-Back mode disabled 1_B Loop-Back mode enabled
R	15	rw	**Baud Rate Generator Run Control** Register BG should only be written if R = 0. 0_B Baud rate generator disabled (ASC inactive) 1_B Baud rate generator enabled
MTX	18:16	rw	**Mode Selection TX direction** While bit field M controls the RX path, MTX controls the mode for the TX path. 000_B 8-bit data Synchronous Mode. M needs to be cleared as well for proper operation. 001_B 8-bit data Asynchronous Mode 010_B Reserved. Do not use this combination. 011_B 7-bit data + parity Asynchronous Mode 100_B 9-bit data Asynchronous Mode 101_B 8-bit data + wake up bit Asynchronous Mode 110_B Reserved. Do not use this combination. 111_B 8-bit data + parity Asynchronous Mode

(table continues...)

(continued)

Field	Bits	Type	Description
ODDTX	28	rw	**Parity Selection TX direction** While bit field ODD controls the RX path, ODDTX controls the mode for the TX path. 0_B Even parity selected (parity bit = 1 on odd number of 1s in data, parity bit = 0 on even number of 1s in data) 1_B Odd parity selected (parity bit = 1 on even number of 1s in data, parity bit = 0 on odd number of 1s in data)
0	27:19, 31:29	r	**Reserved** Read as 0; should be written with 0.

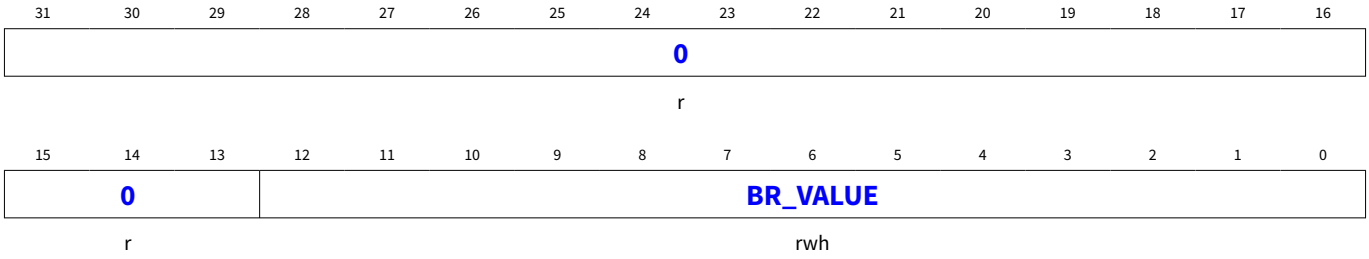
Serial data transmission or reception is possible only when the run bit CON.R is set to 1. Otherwise, the serial interface is idle. To avoid unpredictable behavior of the serial interface, do not program the mode control field CON.M to one of the reserved combinations.

27.4.40 Baud Rate Timer/Reload Register

The Baud Rate Timer Reload Register BG of the ASC module contains the 13-bit reload value for the baud rate timer in Asynchronous and Synchronous Modes.

BG Offset address: $019C_H$

Baud Rate Timer/Reload Register Kernel Reset value: $0000\ 0000_H$



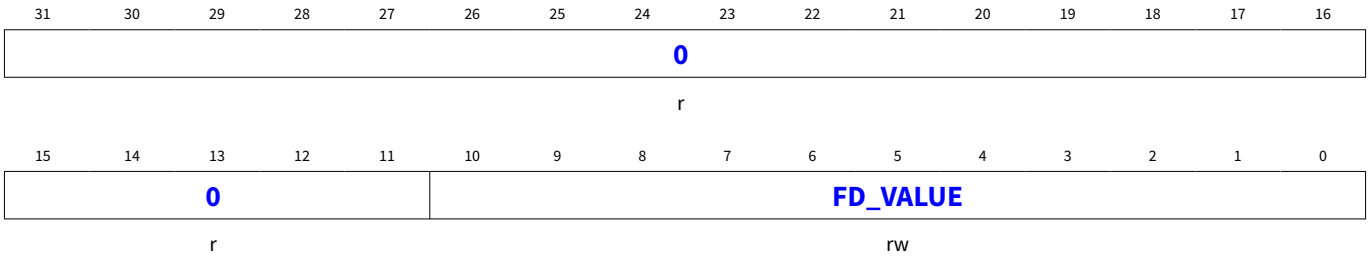
Field	Bits	Type	Description
BR_VALUE	12:0	rwh	**Baud Rate Timer/Reload Register Value** Reading BR_VALUE returns the 13-bit content of the baud rate timer. Writing BR_VALUE loads the baud rate timer reload register. BG should only be written if CON.R = 0.
0	31:13	r	**Reserved** Read as 0; should be written with 0.

27.4.41 Fractional Divider Register

The Fractional Divider Register FDV of the ASC module contains the 11-bit divider value for the fractional divider (asynchronous mode only).

FDV Offset address: $01A0_H$

Fractional Divider Register Kernel Reset value: $0000\ 0000_H$



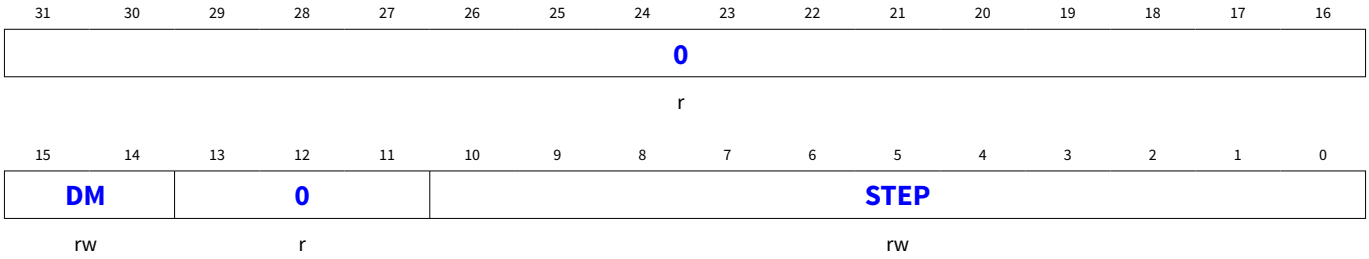
Field	Bits	Type	Description
FD_VALUE	10:0	rw	**Fractional Divider Register Value** FD_VALUE contains the 11-bit value n of the fractional divider which determines the fractional divider ratio n/2048 (n = 0-2047). With n = 0, the fractional divider is switched off (divider ratio = 1).
0	31:11	r	**Reserved** Read as 0; should be written with 0.

27.4.42 Fractional Divider for Output CLK Register

The Fractional Divider Register FDO contains the 11-bit divider value for the fractional divider that is generating $f_{PSISCLK}$ from f_{ASC} . PSISCLK is a clock that can be used on a pin to drive the external PHY.

FDO Offset address: $01A4_H$

Fractional Divider for Output CLK Register Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
STEP	10:0	rw	**Step Value** Reload or addition value for internal accumulator.
DM	15:14	rw	**Divider Mode** DM selects normal or fractional divider mode. 00_B Fractional divider is switched off; no output clock is generated. The Reset External Divider signal is 1. 01_B Normal Divider Mode selected. 10_B Fractional Divider Mode selected. 11_B Fractional divider is switched off; no output clock is generated.
0	13:11, 31:16	r	**Reserved** Read as 0; should be written with 0.

Related information

Clock generation on page 4977

27.4.43 Transmit Buffer Register

The Transmit Buffer Register TBUF of the ASC module contains the transmit data value in Asynchronous And Synchronous Modes. If GCR.ASC is cleared, TBUF is no longer writable by software and interrupts are handled by the message reassembly block automatically.

TBUF Offset address: $01A8_H$

Transmit Buffer Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0							TD_VALUE								
r							rwh								

Field	Bits	Type	Description
TD_VALUE	8:0	rwh	**Transmit Data Register Value** TBUF contains the data to be transmitted in the asynchronous and synchronous operating modes of the ASC. Data transmission is double-buffered; therefore, a new value can be written to TBUF before the transmission of the previous value is complete.
0	31:9	r	**Reserved** Read as 0; should be written with 0.

27.4.44 Receive Buffer Register

The receive buffer register RBUF of the ASC module contains the receive data value in Asynchronous and Synchronous Modes.

RBUF Offset address: $01AC_H$

Receive Buffer Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0							RD_VALUE								
r							rh								

Field	Bits	Type	Description
RD_VALUE	8:0	rh	**Receive Data Register Value** RD_VALUE contains the received data bits and, depending on the selected mode, the parity bit in the asynchronous and synchronous operating modes of the ASC. In Asynchronous Mode, with CON.M = 011_B (7-bit data + parity), the received parity bit is written into RD_VALUE[7]. In Asynchronous Mode, with CON.M = 111_B (8-bit data + parity), the received parity bit is written into RD_VALUE[8].
0	31:9	r	**Reserved** Read as 0.

27.4.45 Write Hardware Bits Control Register

The three error flags in register CON and the REN bit can be set or cleared by software by register WHBCON. WHBCON is a write-only register. Reading WHBCON always returns 0000 0000_H.

***Note**: When the set and clear bits for an error flag are set at the same time during a WHBCON write operation (for example, SETPE = CLRPE = 1), the error flag in CON is not affected.*

WHBCON Offset address: $01B0_H$

Write Hardware Bits Control Register Kernel Reset value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		SETOE	SETFE	SETPE	CLROE	CLRFE	CLRPE	0		SETREN	CLRREN	0			
r		w	w	w	w	w	w	r		w	w	r			

Field	Bits	Type	Description
CLRREN	4	w	**Clear Receiver Enable Bit** Bit is always read as 0. 0_B No effect 1_B Bit CON.REN is cleared.
SETREN	5	w	**Set Receiver Enable Bit** Bit is always read as 0. 0_B No effect 1_B Bit CON.REN is set.
CLRPE	8	w	**Clear Parity Error Flag** Bit is always read as 0. 0_B No effect 1_B Bit CON.PE is cleared.

(table continues...)

(continued)

Field	Bits	Type	Description
CLRFE	9	w	**Clear Framing Error Flag** Bit is always read as 0. 0_B No effect 1_B Bit CON.FE is cleared.
CLROE	10	w	**Clear Overrun Error Flag** Bit is always read as 0. 0_B No effect 1_B Bit CON.OE is cleared.
SETPE	11	w	**Set Parity Error Flag** Bit is always read as 0. 0_B No effect 1_B Bit CON.PE is set.
SETFE	12	w	**Set Framing Error Flag** Bit is always read as 0. 0_B No effect 1_B Bit CON.FE is set.
SETOE	13	w	**Set Overrun Error Flag** Bit is always read as 0. 0_B No effect 1_B Bit CON.OE is set.
0	3:0, 7:6, 31:14	r	**Reserved** Read as 0; should be written with 0.

27.4.46 Interrupt Status Register x

The Interrupt Status Registers INTSTATx contains status bits that show the status of any interrupt of PSI5-S channel x.

The bits are set independently from the referring Interrupt Enable in Register INTENx. Thus they can be used as status bits as well. For example, by a software based on polling.

INTSTATx (x=0-7) Offset address: $01B4_H+x*4$

Interrupt Status Register x Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0							HDI	TPOI	TPI	CRCI	CHCI	TEI	RBI	RDI	RSI
r							rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
RSI	0	rh	**Receive Success Interrupt Request Flag** This bit is set at the successfully received end of a frame. It indicates that this frame is free of the errors CRCI, XCRCI, TEI, PE, FE, OE, RBI, HDI if selected to be taken into account in register GCR. This bit can be cleared by bit INTCLRx.RSI. This bit can be set by bit INTSETx.RSI. This bit is set independently from INTENx. 0_B No interrupt was requested since this bit was cleared the last time 1_B An interrupt was requested since this bit was cleared the last time
RDI	1	rh	**Receive Data Interrupt Request Flag** RDI is activated when a received frame is moved to a Receive Data Register RDR. Both RDI and RSI will be issued together at correct reception. This bit can be cleared by bit INTCLRx.RDI. This bit can be set by bit INTSETx.RDI. This bit is set independently from INTENx. 0_B No interrupt was requested since this bit was cleared the last time 1_B An interrupt was requested since this bit was cleared the last time
RBI	2	rh	**Receive Buffer Overflow Interrupt Request Flag** This bit is set after a frame has been received while the old one was not read from RDR. That is, the kernel wants to set interrupt RDI and finds this interrupt already set. The old data is overwritten by the new data. This bit is NOT cleared by reading RDR. This bit can be cleared by bit INTCLRx.RBI. This bit can be set by bit INTSETx.RBI. This bit is set independently from INTENx. 0_B No interrupt was requested since this bit was cleared the last time 1_B An interrupt was requested since this bit was cleared the last time
TEI	3	rh	**Timing Error Interrupt Request Flag** This bit is set if the watchdog timer expired. Depending from RCRAx.WDMS either the distance between two RDIs is longer than specified in WDL or it expired without reception of CHCI in time. That is, the time from issuing the sync pulse to reception of the last expected frame configured in NFC.NFx was too long. Note that the root cause might be a non-recoverable frame! This bit can be cleared by bit INTCLRx.TEI. This bit can be set by bit INTSETx.TEI. This bit is set independently from INTENx. 0_B No interrupt was requested since this bit was cleared the last time 1_B An interrupt was requested since this bit was cleared the last time

(table continues...)

(continued)

Field	Bits	Type	Description
CHCI	4	rh	**Channel Completed Interrupt Request Flag** This bit is set if FCNT.FCx equals NFC.NFx. This bit can be cleared by bit INTCLRx.CHCI. This bit can be set by bit INTSETx.CHCI. This bit is set independently from INTENx. 0_B No interrupt was requested since this bit was cleared the last time 1_B An interrupt was requested since this bit was cleared the last time
CRCI	5	rh	**CRC Error Request Flag** This bit is set if the CRC fails. This bit can be cleared by bit INTCLRx.CRCI. This bit can be set by bit INTSETx.CRCI. This bit is set independently from INTENx. 0_B No interrupt was requested since this bit was cleared the last time 1_B An interrupt was requested since this bit was cleared the last time
TPI	6	rh	**Transfer Preparation Interrupt Request Flag** This bit is set after data to be transferred has been moved completely. Thus a new value can be written to SDRx. This bit can be cleared by bit INTCLRx.TPI. This bit can be set by bit INTSETx.TPI. This bit is set independently from INTENx. 0_B No interrupt was requested since this bit was cleared the last time 1_B An interrupt was requested since this bit was cleared the last time
TPOI	7	rh	**Transmit Preparation Overflow Interrupt Request Flag** This bit is set if SDR is written while TPF is set. The old data is NOT overwritten. This bit can be cleared by bit INTCLRx.TPOI. This bit can be set by bit INTSETx.TPOI. This bit is set independently from INTENx. 0_B No interrupt was requested since this bit was cleared the last time 1_B An interrupt was requested since this bit was cleared the last time
HDI	8	rh	**Header Error Signalled Flag** This bit is set if at least one of the error signalling flags in the enveloping Packet frame Err0 and Err1 is set. 0_B (Err0 OR Err1) = false (0) 1_B (Err0 OR Err1) = true (1)
0	31:9	r	**Reserved** Read as 0.

27.4.47 Interrupt Set Register x

The Interrupt Set Registers INTSETx contain control bits that trigger an interrupt pulse for any interrupt of PSI5-S channel x.

INTSETx (x=0-7) Offset address: $01D4_H+x*4$

Interrupt Set Register x Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0							HDI	TPOI	TPI	CRCI	CHCI	TEI	RBI	RDI	RSI
r							w	w	w	w	w	w	w	w	w

Field	Bits	Type	Description
RSI	0	w	**Set Interrupt Request Flag RSI** Setting this bit sets bit INTSTATx.RSI. Clearing this bit has no effect. Reading this bit returns always zero.
RDI	1	w	**Set Interrupt Request Flag RDI** Setting this bit sets bit INTSTATx.RDI. Clearing this bit has no effect. Reading this bit returns always zero.
RBI	2	w	**Set Interrupt Request Flag RBI** Setting this bit sets bit INTSTATx.RBI. Clearing this bit has no effect. Reading this bit returns always zero.
TEI	3	w	**Set Interrupt Request Flag TEI** Setting this bit sets bit INTSTATx.TEI. Clearing this bit has no effect. Reading this bit returns always zero.
CHCI	4	w	**Set Interrupt Request Flag CHCI** Setting this bit sets bit INTSTATx.CHCI. Clearing this bit has no effect. Reading this bit returns always zero.
CRCI	5	w	**Set Interrupt Request Flag CRCI** Setting this bit sets bit INTSTATx.CRCI. Clearing this bit has no effect. Reading this bit returns always zero.
TPI	6	w	**Set Interrupt Request Flag TPI** Setting this bit sets bit INTSTATx.TPI. Clearing this bit has no effect. Reading this bit returns always zero.

(table continues...)

(continued)

Field	Bits	Type	Description
TPOI	7	w	**Set Interrupt Request Flag TPOI** Setting this bit sets bit INTSTATx.TPOI. Clearing this bit has no effect. Reading this bit returns always zero.
HDI	8	w	**Set Interrupt Request Flag HDI** Setting this bit sets bit INTSTATx.HDI. Clearing this bit has no effect. Reading this bit returns always zero.
0	31:9	r	**Reserved** Read as 0; should be written with 0.

27.4.48 Interrupt Clear Register x

The Interrupt Clear Register INTCLRx contain control bits that clear the status of any interrupt of PSI5-S channel x.

INTCLRx (x=0-7) Offset address: $01F4_H+x*4$

Interrupt Clear Register x Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0							HDI	TPOI	TPI	CRCI	CHCI	TEI	RBI	RDI	RSI
r							w	w	w	w	w	w	w	w	w

Field	Bits	Type	Description
RSI	0	w	**Clear Interrupt Request Flag RSI** Setting this bit clears bit INTSTATx.RSI. Clearing this bit has no effect. Reading this bit returns always zero.
RDI	1	w	**Clear Interrupt Request Flag RDI** Setting this bit clears bit INTSTATx.RDI. Clearing this bit has no effect. Reading this bit returns always zero.
RBI	2	w	**Clear Interrupt Request Flag RBI** Setting this bit clears bit INTSTATx.RBI. Clearing this bit has no effect. Reading this bit returns always zero.

(table continues...)

(continued)

Field	Bits	Type	Description
TEI	3	w	**Clear Interrupt Request Flag TEI** Setting this bit clears bit INTSTATx.TEI. Clearing this bit has no effect. Reading this bit returns always zero.
CHCI	4	w	**Clear Interrupt Request Flag CHCI** Setting this bit clears bit INTSTATx.CHCI. Clearing this bit has no effect. Reading this bit returns always zero.
CRCI	5	w	**Clear Interrupt Request Flag CRCI** Setting this bit clears bit INTSTATx.CRCI. Clearing this bit has no effect. Reading this bit returns always zero.
TPI	6	w	**Clear Interrupt Request Flag TPI** Setting this bit clears bit INTSTATx.TPI. Clearing this bit has no effect. Reading this bit returns always zero.
TPOI	7	w	**Clear Interrupt Request Flag TPOI** Setting this bit clears bit INTSTATx.TPOI. Clearing this bit has no effect. Reading this bit returns always zero.
HDI	8	w	**Clear Interrupt Request Flag HDI** Setting this bit clears bit INTSTATx.HDI. Clearing this bit has no effect. Reading this bit returns always zero.
0	31:9	r	**Reserved** Read as 0; should be written with 0.

27.4.49 Interrupt Enable Register x

The Interrupt Enable Register INTENx contain control bits that enable the interrupt source of any interrupt of PSI5-S channel x.

The Interrupt Status bits in register INTSTATx are set independently from the Interrupt Enable in Register INTENx.

INTENx (x=0-7) Offset address: 0214_H+x*4

Interrupt Enable Register x Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0							HDI	TPOI	TPI	CRCI	CHCI	TEI	RBI	RDI	RSI
r							rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RSI	0	rw	**Enable Interrupt Request RSI** 0_B No interrupt request can be generated for this source 1_B An interrupt request can be generated for this source
RDI	1	rw	**Enable Interrupt Request RDI** 0_B No interrupt request can be generated for this source 1_B An interrupt request can be generated for this source
RBI	2	rw	**Enable Interrupt Request RBI** 0_B No interrupt request can be generated for this source 1_B An interrupt request can be generated for this source
TEI	3	rw	**Enable Interrupt Request TEI** 0_B No interrupt request can be generated for this source 1_B An interrupt request can be generated for this source
CHCI	4	rw	**Enable Interrupt Request CHCI** 0_B No interrupt request can be generated for this source 1_B An interrupt request can be generated for this source
CRCI	5	rw	**Enable Interrupt Request CRCI** 0_B No interrupt request can be generated for this source 1_B An interrupt request can be generated for this source
TPI	6	rw	**Enable Interrupt Request TPI** 0_B No interrupt request can be generated for this source 1_B An interrupt request can be generated for this source
TPOI	7	rw	**Enable Interrupt Request TPOI** 0_B No interrupt request can be generated for this source 1_B An interrupt request can be generated for this source
HDI	8	rw	**Enable Interrupt Request HDI** 0_B No interrupt request can be generated for this source 1_B An interrupt request can be generated for this source
0	31:9	r	**Reserved** Read as 0; should be written with 0.

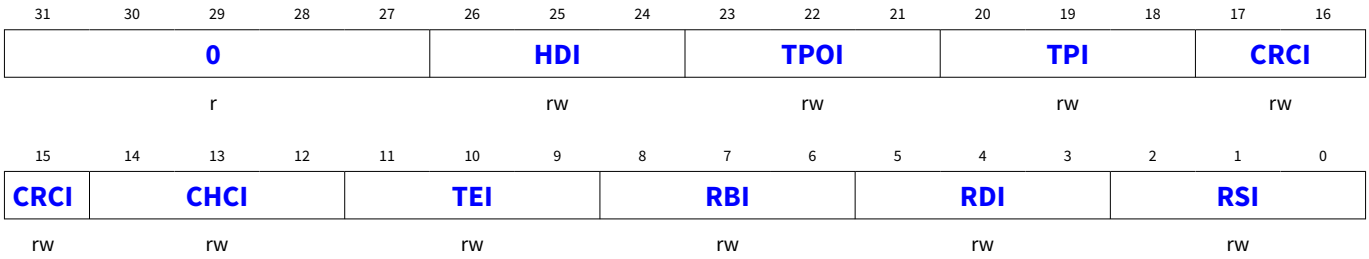
27.4.50 Interrupt Node Pointer Register x

The Interrupt Node Pointer Register INPx contains the node pointers of PSI5-S channel x.

INPx (x=0-7) Offset address: 0234_H+x*4

Interrupt Node Pointer Register x Kernel Reset value: $0000\ 0000_H$

27 Peripheral Sensor Interface with Serial Interface to PHY (PSI5-S)



Field	Bits	Type	Description
RSI	2:0	rw	**Interrupt Node Pointer for Interrupt RSI** This bit field defines the interrupt node that is requested due to the set condition for bit INTSTATx.RSI (if enabled by bit INTENx.RSI). 000_B Trigger Output PSI5SwSRx 0 is selected ... 111_B Trigger Output PSI5SwSRx 7 is selected
RDI	5:3	rw	**Interrupt Node Pointer for Interrupt RDI** This bit field defines the interrupt node that is requested due to the set condition for bit INTSTATx.RDI (if enabled by bit INTENx.RDI). For bit field definition, see INPx.RSI
RBI	8:6	rw	**Interrupt Node Pointer for Interrupt RBI** This bit field defines the interrupt node that is requested due to the set condition for bit INTSTATx.RBI (if enabled by bit INTENx.RBI). For bit field definition, see INPx.RSI
TEI	11:9	rw	**Interrupt Node Pointer for Interrupt TEI** This bit field defines the interrupt node that is requested due to the set condition for bit INTSTATx.TEI (if enabled by bit INTENx.TEI). For bit field definition, see INPx.RSI
CHCI	14:12	rw	**Interrupt Node Pointer for Interrupt CHCI** This bit field defines the interrupt node that is requested due to the set condition for bit INTSTATx.CHCI (if enabled by bit INTENx.CHCI). For bit field definition, see INPx.RSI
CRCI	17:15	rw	**Interrupt Node Pointer for Interrupt CRCI** This bit field defines the interrupt node that is requested due to the set condition for bit INTSTATx.CRCI. For bit field definition, see INPx.RSI
TPI	20:18	rw	**Interrupt Node Pointer for Interrupt TOI** This bit field defines the interrupt node that is requested due to the set condition for bit INTSTATx.TPI (if enabled by bit INTENx.TPI). For bit field definition, see INPx.RSI
TPOI	23:21	rw	**Interrupt Node Pointer for TPOI** This bit field defines the interrupt node that is requested due to the set condition for bit INTSTATx.TPOI. For bit field definition, see INPx.RSI

(table continues...)

(continued)

Field	Bits	Type	Description
HDI	26:24	rw	**Interrupt Node Pointer for HDI** This bit field defines the interrupt node that is requested due to the set condition for bit INTSTATx.HDI. For bit field definition, see INPx.RSI
0	31:27	r	**Reserved** Read as 0; should be written with 0.

27.4.51 Interrupt Overview Register

INTOV Offset address: 0254_H

Interrupt Overview Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	FOI	XCRCI	TBIR	EIR	RIR	TIR	HDI	TPOI	TPI	CRCI	CHCI	TEI	RBI	RDI	RSI
r	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
RSI	0	rh	**Interrupt Pending on Node Pointer RSI** If any interrupt requested flag is set for this node pointer in register (INTSTATx or INTSTATG) and the referring interrupt is enabled in (INTENx or INTENG), then this bit is set. It is automatically reset if all flags in INTSTATx/G are cleared for which the referring interrupt is enabled in INTENx/G.
RDI	1	rh	**Interrupt Pending on Node Pointer RDI** See details of INTOV.RSI.
RBI	2	rh	**Interrupt Pending on Node Pointer RBI** See details of INTOV.RSI.
TEI	3	rh	**Interrupt Pending on Node Pointer TEI** See details of INTOV.RSI.
CHCI	4	rh	**Interrupt Pending on Node Pointer CHCI** See details of INTOV.RSI.
CRCI	5	rh	**Interrupt Pending on Node Pointer CRCI** See details of INTOV.RSI.
TPI	6	rh	**Interrupt Pending on Node Pointer TPI** See details of INTOV.RSI.
TPOI	7	rh	**Interrupt Pending on Node Pointer TPOI** See details of INTOV.RSI.

(table continues...)

(continued)

Field	Bits	Type	Description
HDI	8	rh	**Interrupt Pending on Node Pointer HDI** See details of INTOV.HDI.
TIR	9	rh	**Interrupt Pending on Node Pointer TIR** If any interrupt requested flag is set for this node pointer in register INTSTATG and the referring interrupt is enabled in INTENG, then this bit is set. It is automatically reset if all flags in INTSTATG are cleared for which the referring interrupt is enabled in INTENG.
RIR	10	rh	**Interrupt Pending on Node Pointer RIR** See details of INTOV.TIR.
EIR	11	rh	**Interrupt Pending on Node Pointer EIR** See details of INTOV.TIR.
TBIR	12	rh	**Interrupt Pending on Node Pointer TBIR** See details of INTOV.TIR.
XCRCI	13	rh	**Interrupt Pending on Node Pointer XCRCI** See details of INTOV.TIR.
FOI	14	rh	**Interrupt Pending on Node Pointer FOI** See details of INTOV.TIR.
0	31:15	r	**Reserved** Read as 0.

27.4.52 Interrupt Status Register Global

The Interrupt Status Registers Global INTSTATG contains status bits that show the status of any global interrupt of the ASC inside PSI5-S and XCRCI indicating a non-recoverable message is received. On XCRCI, the non-recoverable message is stored in ChID ‘0’, FID ’1’ with original IDs. INTSTATG contains as well FOI indicating a FIFO overrun condition. In a correct setup, this will never be set as the bandwidth of the ASC is assumed to be by far higher than the write bandwidth in order to have short delays.

The bits are set independently from the referring Interrupt Enable in Register INTENG. Thus they can be used as status bits as well. For example, by a software based on polling.

INTSTATG Offset address: 0258_H

Interrupt Status Register Global Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0										FOI	XCRCI	TBIR	EIR	RIR	TIR
r										rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
TIR	0	rh	**Transmit Interrupt Request Flag** This bit can be cleared by bit INTCLRG.TIR. This bit can be set by bit INTSETG.TIR. This bit is set independently from INTENG. 0_B No interrupt was requested since this bit was cleared the last time 1_B An interrupt was requested since this bit was cleared the last time
RIR	1	rh	**Receive Interrupt Request Flag** This bit can be cleared by bit INTCLRG.RIR. This bit can be set by bit INTSETG.RIR. This bit is set independently from INTENG. 0_B No interrupt was requested since this bit was cleared the last time 1_B An interrupt was requested since this bit was cleared the last time
EIR	2	rh	**Error Interrupt Request Flag** The cause of an error interrupt request EIR (framing, parity, overrun error) can be identified by the error status flags CON.FE, CON.PE, and CON.OE, This bit can be cleared by bit INTCLRG.EIR. This bit can be set by bit INTSETG.EIR. This bit is set independently from INTENG. 0_B No interrupt was requested since this bit was cleared the last time 1_B An interrupt was requested since this bit was cleared the last time
TBIR	3	rh	**Transmit Buffer Interrupt Request Flag** This bit can be cleared by bit INTCLRG.TBIR. This bit can be set by bit INTSETG.TBIR. This bit is set independently from INTENG. 0_B No interrupt was requested since this bit was cleared the last time 1_B An interrupt was requested since this bit was cleared the last time
XCRCI	4	rh	**XCRC Error Request Flag** This bit is set if the CRC check on the enveloping Packet frame fails including the case where XCRC check can not be performed (non-recoverable frames). Refer Packet frames received from PHY chapter. The received data is not reliable and stored in ChID ‘0’, FID ’1’ with original IDs. This bit can be cleared by bit INTCLRG.XCRCI. This bit can be set by bit INTSETG.XCRCI. This bit is set independently from INTENG. 0_B No interrupt was requested since this bit was cleared the last time 1_B An interrupt was requested since this bit was cleared the last time

(table continues...)

(continued)

Field	Bits	Type	Description
FOI	5	rh	**FIFO Error Request Flag** This bit is set if the Transmit FIFO of the message generation is overrun, that is, a transfer to the FIFO was generated by the message generation unit or by CDW (CPU Direct Write Register) while the FIFO was already full. This bit can be cleared by bit INTCLRG.FOI. This bit can be set by bit INTSETG.FOI. This bit is set independently from INTENG. 0_B No interrupt was requested since this bit was cleared the last time 1_B An interrupt was requested since this bit was cleared the last time
0	31:6	r	**Reserved** Read as 0.

Related information

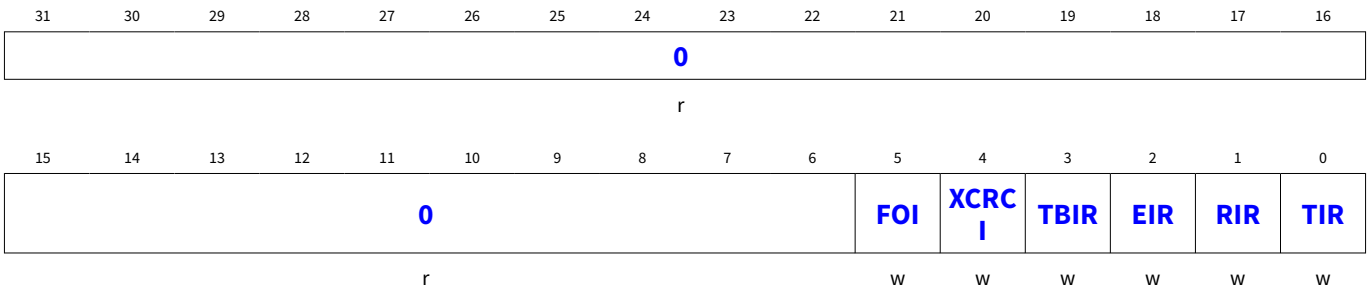
Packet frames received from PHY on page 4970

27.4.53 Interrupt Set Register Global

The Interrupt Set Register Global INTSETG contains control bits that trigger an interrupt pulse for any interrupt of the ASC integrated in PSI5-S and for XCRCI and FOI.

INTSETG Offset address: $025C_H$

Interrupt Set Register Global Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
TIR	0	w	**Set Interrupt Request Flag TIR** Setting this bit sets bit INTSTATG.TIR. Clearing this bit has no effect. Reading this bit returns always zero.
RIR	1	w	**Set Interrupt Request Flag RIR** Setting this bit sets bit INTSTATG.RIR. Clearing this bit has no effect. Reading this bit returns always zero.

(table continues...)

(continued)

Field	Bits	Type	Description
EIR	2	w	**Set Interrupt Request Flag EIR** Setting this bit sets bit INTSTATG.EIR. Clearing this bit has no effect. Reading this bit returns always zero.
TBIR	3	w	**Set Interrupt Request Flag TBIR** Setting this bit sets bit INTSTATG.TBIR. Clearing this bit has no effect. Reading this bit returns always zero.
XCRCI	4	w	**Set Interrupt Request Flag XCRCI** Setting this bit sets bit INTSTATG.XCRCI. Clearing this bit has no effect. Reading this bit returns always zero.
FOI	5	w	**Set Interrupt Request Flag FOI** Setting this bit sets bit INTSTATG.FOI. Clearing this bit has no effect. Reading this bit returns always zero.
0	31:6	r	**Reserved** Read as 0; should be written with 0.

27.4.54 Interrupt Clear Register Global

The Interrupt Clear Register Global INTCLRG contain control bits that clear the status of any interrupt of the ASC integrated in PSI5-S and for XCRCI and FOI.

INTCLRG Offset address: 0260_H

Interrupt Clear Register Global Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0										FOI	XCRCI	TBIR	EIR	RIR	TIR
r										w	w	w	w	w	w

Field	Bits	Type	Description
TIR	0	w	**Clear Interrupt Request Flag TIR** Setting this bit clears bit INTSTATG.TIR. Clearing this bit has no effect. Reading this bit returns always zero.

(table continues...)

(continued)

Field	Bits	Type	Description
RIR	1	w	**Clear Interrupt Request Flag RIR** Setting this bit clears bit INTSTATG.RIR. Clearing this bit has no effect. Reading this bit returns always zero.
EIR	2	w	**Clear Interrupt Request Flag EIR** Setting this bit clears bit INTSTATG.EIR. Clearing this bit has no effect. Reading this bit returns always zero.
TBIR	3	w	**Clear Interrupt Request Flag TBIR** Setting this bit clears bit INTSTATG.TBIR. Clearing this bit has no effect. Reading this bit returns always zero.
XCRCI	4	w	**Clear Interrupt Request Flag XCRCI** Setting this bit clears bit INTSTATG.XCRCI. Clearing this bit has no effect. Reading this bit returns always zero.
FOI	5	w	**Clear Interrupt Request Flag FOI** Setting this bit clears bit INTSTATG.FOI. Clearing this bit has no effect. Reading this bit returns always zero.
0	31:6	r	**Reserved** Read as 0; should be written with 0.

27.4.55 Interrupt Enable Register Global

The Interrupt Enable Register Global INTENG contain control bits that enable the interrupt source of any interrupt of the ASC integrated in PSI5-S and for XCRCI and FOI.

The Interrupt Status bits in register INTSTATG are set independently from the Interrupt Enable in Register INTENG.

INTENG Offset address: 0264_H

Interrupt Enable Register Global Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0										FOI	XCRCI	TBIR	EIR	RIR	TIR
r										rw	rw	rw	rw	rw	rw

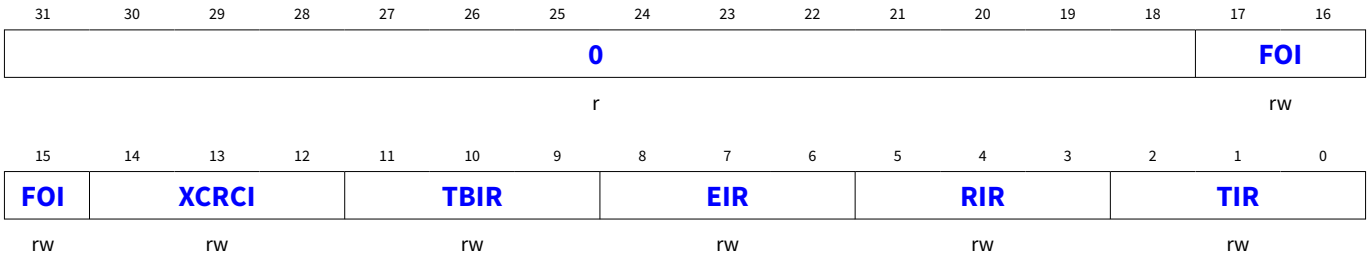
Field	Bits	Type	Description
TIR	0	rw	**Enable Interrupt Request TIR** 0_B No interrupt request can be generated for this source 1_B An interrupt request can be generated for this source
RIR	1	rw	**Enable Interrupt Request RIR** 0_B No interrupt request can be generated for this source 1_B An interrupt request can be generated for this source
EIR	2	rw	**Enable Interrupt Request EIR** 0_B No interrupt request can be generated for this source 1_B An interrupt request can be generated for this source
TBIR	3	rw	**Enable Interrupt Request TBIR** 0_B No interrupt request can be generated for this source 1_B An interrupt request can be generated for this source
XCRCI	4	rw	**Enable Interrupt Request XCRCI** 0_B No interrupt request can be generated for this source 1_B An interrupt request can be generated for this source
FOI	5	rw	**Enable Interrupt Request FOI** 0_B No interrupt request can be generated for this source 1_B An interrupt request can be generated for this source
0	31:6	r	**Reserved** Read as 0; should be written with 0.

27.4.56 Interrupt Node Pointer Register Global

The Interrupt Node Pointer Register Global INPG contains the node pointers of PSI5-S ASC and for XCRCI and FOI.

INPG Offset address: 0268_H

Interrupt Node Pointer Register Global Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
TIR	2:0	rw	**Interrupt Node Pointer for Interrupt TIR** This bit field defines the interrupt node that is requested due to the set condition for bit INTSTATG.TIR (if enabled by bit INTENG.TIR). 000_B Trigger Output PSI5SwSRx 0 is selected ... 111_B Trigger Output PSI5SwSRx 7 is selected
RIR	5:3	rw	**Interrupt Node Pointer for Interrupt RIR** This bit field defines the interrupt node that is requested due to the set condition for bit INTSTATG.RIR (if enabled by bit INTENG.RIR). For bit field definition, see INPGx.TIR.
EIR	8:6	rw	**Interrupt Node Pointer for Interrupt EIR** This bit field defines the interrupt node that is requested due to the set condition for bit INTSTATG.EIR (if enabled by bit INTENG.EIR). For bit field definition, see INPGx.TIR.
TBIR	11:9	rw	**Interrupt Node Pointer for Interrupt TBIR** This bit field defines the interrupt node that is requested due to the set condition for bit INTSTATG.TBIR (if enabled by bit INTENG.TBIR). For bit field definition, see INPGx.TIR.
XCRCI	14:12	rw	**Interrupt Node Pointer for Interrupt XCRCI** This bit field defines the interrupt node that is requested due to the set condition for bit INTSTATG.XCRCI (if enabled by bit INTENG.XCRCI). For bit field definition, see INPGx.TIR.
FOI	17:15	rw	**Interrupt Node Pointer for Interrupt FOI** This bit field defines the interrupt node that is requested due to the set condition for bit INTSTATG.FOI (if enabled by bit INTENG.FOI). For bit field definition, see INPGx.TIR.
0	31:18	r	**Reserved** Read as 0; should be written with 0.

27.5 Debug information

This section is not applicable for the module being described.

27.6 References

1. PSI5 Peripheral Sensor Interface V2.1 Technical Specification
2. SENT—Single Edge Nibble Transmission for Automotive Applications J2716_201001

27.7 PSI5-S revision history

Initial release of the chapter.

27.8 TC4Dx PSI5-S information

27.8.1 TC4Dx PSI5-S configuration

Table 1201 TC4Dx PSI5 configuration

Parameters	PSI5-S
Number of instances	1

27.8.2 TC4Dx PSI5-S features

There are no deviations from the generic specification.

27.8.3 TC4Dx PSI5-S functional description

1. PSI5-S acts as a slave on the ComPB bus. The Clock Control block is driven by the system clock frequency f_{SPB} as in TC4Dx, the ComPB also uses this clock
2. There is no GTM trigger available. Only eGTM trigger signals are used for timestamp generation and sync pulse control

27.8.4 TC4Dx PSI5-S registers

There are no deviations from the generic specification.

27.8.4.1 Register address space - PSI5S

Table 1202 Registers address space - PSI5S

Module	Base address	End address	Note
PSI5S0	$F4407000_H$	$F4407FFF_H$	FPI slave interface

27.8.4.2 Register overview - access mode glossary

Table 1203 Register overview - access mode glossary

Keyword	Description
E	Access protection using PROT register PSI5S0_PROTE .
SE	Access protection using PROT register PSI5S0_PROTSE .
APU-P	Protection group consisting of registers PSI5S0_ACCEN_WRA , PSI5S0_ACCEN_WRB , PSI5S0_ACCEN_RDA , PSI5S0_ACCEN_RDB , PSI5S0_ACCEN_VM , PSI5S0_ACCEN_PRS .
P	Access protection using *APU-P* registers.
BE	Always returns a Bus Error.
U	No access restrictions.
PROT	Access restrictions as defined in the PROT register access rules.

27.8.4.3 Registers overview - PSI5S0 (ascending offset address)

Table 1204 Registers overview - PSI5S0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PSI5S0_CLC	Clock Control Register	0000_H	P	P, SV, E	Application Reset	5016
PSI5S0_OCS	OCDS Control and Status Register	0004_H	P	SV, P	Debug Reset	5017
PSI5S0_ID	Module Identification Register	0008_H	P	BE	PowerOn Reset	5018
PSI5S0_RST_CTRLA	Reset Control Register A	$000C_H$	P	P, SV, E	Application Reset	5019
PSI5S0_RST_CTRLB	Reset Control Register B	0010_H	P	P, SV, E	Application Reset	5019
PSI5S0_RST_STAT	Reset Status Register	0014_H	P	BE	Application Reset	5020
PSI5S0_PROTE	PROT Register Endinit	0020_H	U	SV, PROT	Application Reset	5021
PSI5S0_PROTSE	PROT Register Safe Endinit	0024_H	U	SV, PROT	Application Reset	5022

(table continues...)

Table 1204 (continued) Registers overview - PSI5S0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PSI5S0_ACCEN_WRA	Write access enable register A	0040_H	U	SE, SV	Application Reset	5024
PSI5S0_ACCEN_WRB	Write access enable register B	0044_H	U	SE, SV	Application Reset	5024
PSI5S0_ACCEN_RDA	Read access enable register A	0048_H	U	SE, SV	Application Reset	5025
PSI5S0_ACCEN_RDB	Read access enable register B	$004C_H$	U	SE, SV	Application Reset	5025
PSI5S0_ACCEN_VM	VM access enable register	0050_H	U	SE, SV	Application Reset	5026
PSI5S0_ACCEN_PRS	PRS access enable register	0054_H	U	SE, SV	Application Reset	5026
PSI5S0_FDR	Fractional Divider Register	0060_H	P	P	Kernel Reset	5027
PSI5S0_FDRT	Fractional Divider Register for Timestamp	0064_H	P	P	Kernel Reset	5028
PSI5S0_TSCNTA	Timestamp Count Register A	0068_H	P	P	Kernel Reset	5029
PSI5S0_TSCNTB	Timestamp Count Register B	$006C_H$	P	P	Kernel Reset	5030
PSI5S0_GCR	Global Control Register	0070_H	P	P	Kernel Reset	5031
PSI5S0_NFC	Number of Frames Control Register	0074_H	P	P	Kernel Reset	5033
PSI5S0_FCNT	Frame Counter Register	0078_H	P	P	Kernel Reset	5034
PSI5S0_IOCR	Input and Output Control Register	$007C_H$	P	P	Kernel Reset	5034
PSI5S0_RCRAx (x=0-7)	Receiver Control Register Ax	0080_H+x *4	P	P	Kernel Reset	5035
PSI5S0_RCRBx (x=0-7)	Receiver Control Register Bx	$00A0_H$+x *4	P	P	Kernel Reset	5037
PSI5S0_WDTx (x=0-7)	Watchdog Timer Register x	$00C0_H$+x *4	P	P	Kernel Reset	5037
PSI5S0_TSCRx (x=0-7)	Timestamp Capture Register x	$00E0_H$+x *4	P	BE	Kernel Reset	5038
PSI5S0_RDS	Receive Status Register	0100_H	P	BE	Kernel Reset	5039
PSI5S0_RDR	Receive Data Register	0104_H	P	BE	Kernel Reset	5041
PSI5S0_TSM	Timestamp Mirror Register	0108_H	P	BE	Kernel Reset	5042
PSI5S0_TAR	Target Address Register	$010C_H$	P	BE	Kernel Reset	5042
PSI5S0_BAR	Base Address Register	0110_H	P	P	Kernel Reset	5043

(table continues...)

Table 1204 (continued) Registers overview - PSI5S0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PSI5S0_PGCx (x=0-7)	Pulse Generation Control Register x	0114_H+x *4	P	P	Kernel Reset	5043
PSI5S0_CTVx (x=0-7)	Channel Trigger Value Register x	0134_H+x *4	P	P	Kernel Reset	5045
PSI5S0_SCRx (x=0-7)	Send Control Register x	0154_H+x *4	P	P	Kernel Reset	5045
PSI5S0_SDRx (x=0-7)	Send Data Register x	0174_H+x *4	P	P	Kernel Reset	5047
PSI5S0_CDW	CPU Direct Write Register	0194_H	P	P	Kernel Reset	5048
PSI5S0_CON	Control Register	0198_H	P	P	Kernel Reset	5049
PSI5S0_BG	Baud Rate Timer/Reload Register	$019C_H$	P	P	Kernel Reset	5051
PSI5S0_FDV	Fractional Divider Register	$01A0_H$	P	P	Kernel Reset	5051
PSI5S0_FDO	Fractional Divider for Output CLK Register	$01A4_H$	P	P	Kernel Reset	5052
PSI5S0_TBUF	Transmit Buffer Register	$01A8_H$	P	P	Kernel Reset	5053
PSI5S0_RBUF	Receive Buffer Register	$01AC_H$	P	BE	Kernel Reset	5053
PSI5S0_WHBCON	Write Hardware Bits Control Register	$01B0_H$	P	P	Kernel Reset	5054
PSI5S0_INTSTATx (x=0-7)	Interrupt Status Register x	$01B4_H$+x *4	P	BE	Kernel Reset	5055
PSI5S0_INTSETx (x=0-7)	Interrupt Set Register x	$01D4_H$+x *4	P	P	Kernel Reset	5058
PSI5S0_INTCLRx (x=0-7)	Interrupt Clear Register x	$01F4_H$+x *4	P	P	Kernel Reset	5059
PSI5S0_INTENx (x=0-7)	Interrupt Enable Register x	0214_H+x *4	P	P	Kernel Reset	5060
PSI5S0_INPx (x=0-7)	Interrupt Node Pointer Register x	0234_H+x *4	P	P	Kernel Reset	5061
PSI5S0_INTOV	Interrupt Overview Register	0254_H	P	BE	Kernel Reset	5063
PSI5S0_INTSTATG	Interrupt Status Register Global	0258_H	P	BE	Kernel Reset	5064
PSI5S0_INTSETG	Interrupt Set Register Global	$025C_H$	P	P	Kernel Reset	5066
PSI5S0_INTCLRG	Interrupt Clear Register Global	0260_H	P	P	Kernel Reset	5067
PSI5S0_INTENG	Interrupt Enable Register Global	0264_H	P	P	Kernel Reset	5068
PSI5S0_INPG	Interrupt Node Pointer Register Global	0268_H	P	P	Kernel Reset	5069

27.8.4.4 Device specific registers

There are no device specific register changes.

27.8.5 TC4Dx PSI5S connectivity

The table below lists interfaces to and from this functional block to other blocks in the device.

Table 1205 List of PSI5S interface signals

Interface signals	I/O	Description
CLOCK_PSI5S_fSPB	In	FPI clock input
PSI5S0_IR_PSI5SwSRx[7:0]	Out	Interrupt trigger output
EGTM_PSI5S0_TRIG[7:0]	In	eGTM trigger
PORTS_PSI5S0_PSI5SRX[3:0]	In	Receive data per PSI5-S module
PSI5S0_PORTS_PSI5STX	Out	Transmit data per PSI5-S module
PSI5S0_PORTS_PSI5SCLK	Out	Clock to drive external PHY

27.8.6 TC4Dx PSI5-S revision history

Initial release of the chapter.

28 Micro Second Channel (MSC)

The MSC is a serial interface that is especially designed to connect external power switch devices. The serial data transmission capability minimizes the number of pins required to connect such external power switch devices. Parallel data information (coming from the timer units) or command information is sent out to the power switch device using a high-speed synchronous serial data stream (downstream channel). The MSC receives data and status back from the power switch device through a low-speed asynchronous serial data stream (upstream channel).

Parallel requests from on chip bus masters to a module will be executed sequentially through the on-chip bus system. A read-modify-write feature provides an atomic read/write sequence where no other master can access the module in between. Module hardware semaphores are not supported.

The following figure shows a typical application in which an MSC interface controls two power switch devices. Output data is provided by the module.

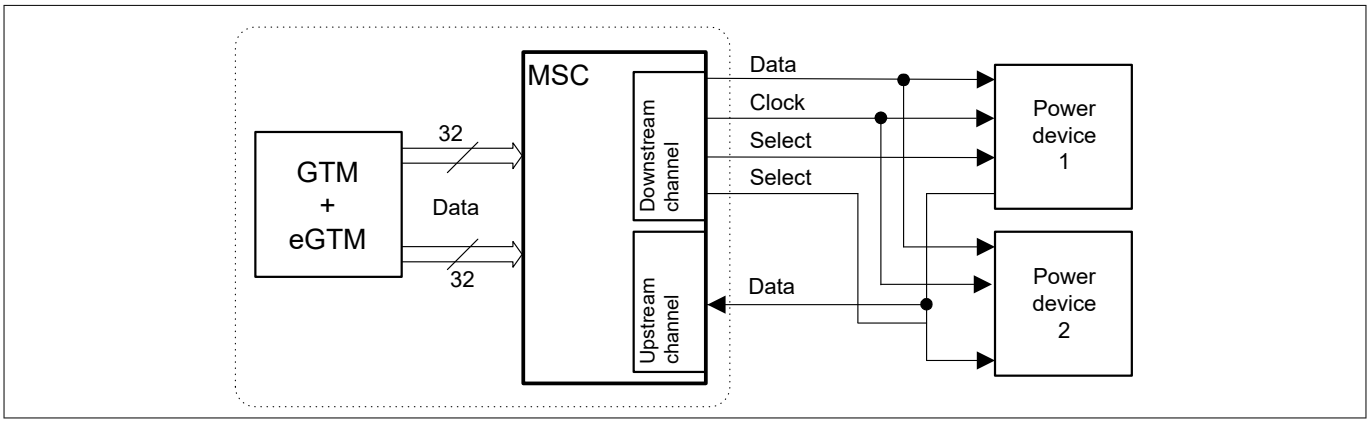


Figure 568 MSC to external power device connection

Some MSC applications are:

- Control of the external power switching unit through the downstream channel
- Receiving information back from power switching unit
- Serial connections to other peripheral devices

28.1 Feature list

- Flexible functional configuration for software
  - Baud rate configuration
- Transmit MSC frames to external devices and handle all protocol relevant topics
  - Supports 64-bit data frames and 32-bit data frames transmission
  - Asynchronous baud rate adjustment block
  - Supports command frame transmission through slow channel
  - Selectable MSC Manchester code transmission: combination of downstream data, clock and enable signals transmitted in the same line
- Receive MSC frames from external devices and handle all protocol relevant topics
  - Programmable upstream data frame length (16 or 12 bits)
  - Programmable delay of the receive interrupt after the last stop bit (0 or 1-bit time)
- Flexible interrupt structure generation available
- Selectable pin types of downstream channel interface: four low-voltage differential signaling (LVDS) output drivers or four digital general purpose input output (GPIO) pins
- Additional clock input selection is available, f_{SRI} = 500 MHz. This clock selection with an additional clock divider offers the possibility to provide a MSC baud rate configuration up to 50 MHz (and frequencies

divided as describe in the Baud rate configuration section) without asynchronous baud rate adjustment (ABRA) usage

28.2 Functional overview

The MSC interface provides a serial communication link typically used to connect power switches or other peripheral devices. The serial communication link includes a fast synchronous downstream channel and a slow asynchronous upstream channel.

The following block diagram of the MSC interface shows a global view of the MSC interface signals.

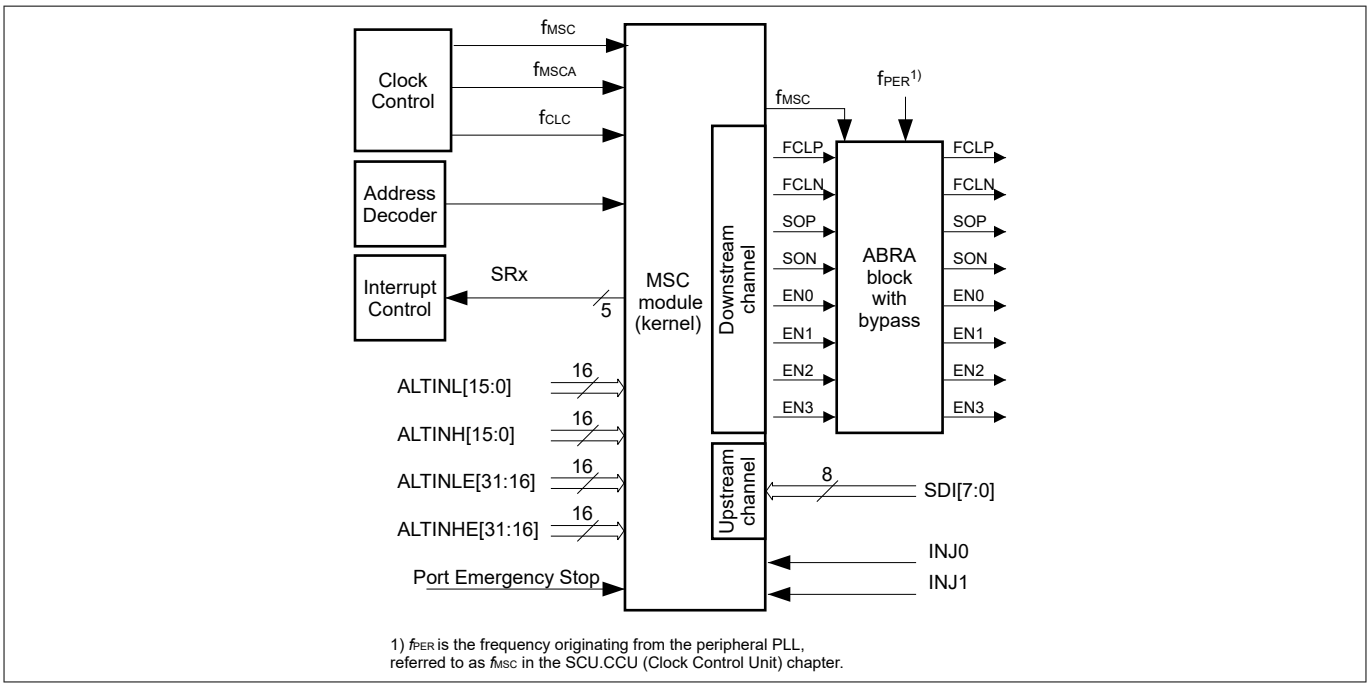


Figure 569 MSC block diagram

The MSC module features functional extensions, but is backward compatible with the previous microcontroller generations.

The downstream and upstream channels of the MSC module communicate with the external world through nine I/O lines. Eight output lines are required for the serial communication of the downstream channel (clock, data and enable signals). One out of the eight input lines SDI[7:0] is used as serial data input signal for the upstream channel. The source of the serial data to be transmitted by the downstream channel can be MSC register contents or data that is provided at the ALTINL/ALTINH input lines. These input lines are typically connected to other on-chip peripheral units (for example with a timer unit like the GTM). A Port Emergency Stop input signal makes it possible to set bits of the serial data stream to dedicated values in case of emergency.

Clock control, address decoding and interrupt service request control are managed outside the MSC module kernel. Service request outputs are able to trigger an interrupt or a DMA request.

In MSC+ mode 2 lines are required for the serial communication of the downstream channel (since clock, data and enable are embedded in one pair line).

Related information

TC4Dx SMU alarm mapping tables on page 7233

28.3 Functional description

This section describes:

- MSC kernel
- The extension mode
- Asynchronous baud rate adjustment block
- Slow channel
- MSC+ transmission
- Module implementation

28.3.1 MSC kernel description

This section describes the functionality of the MSC kernel.

***Note**: The configuration should not be changed on run-time.*

28.3.1.1 Downstream channel

The downstream channel performs a high-speed synchronous serial transmission of data to external devices. Its 64-bit shift register is divided into two 32-bit parts, SRL and SRH.

Each bit of SRL and SRH can be selected to be delivered by the Downstream Data Register DD, by the Downstream Command Register DC or by two 16-bit wide input signal buses, ALTINL and ALTINH.

In the 64-bit extended mode, set with the bit DSCE.EXEN=1, the MSC module provides two 16-bit extension buses, ALTINLE and ALTINHE.

The figure below shows a diagram of the MSC downstream channel.

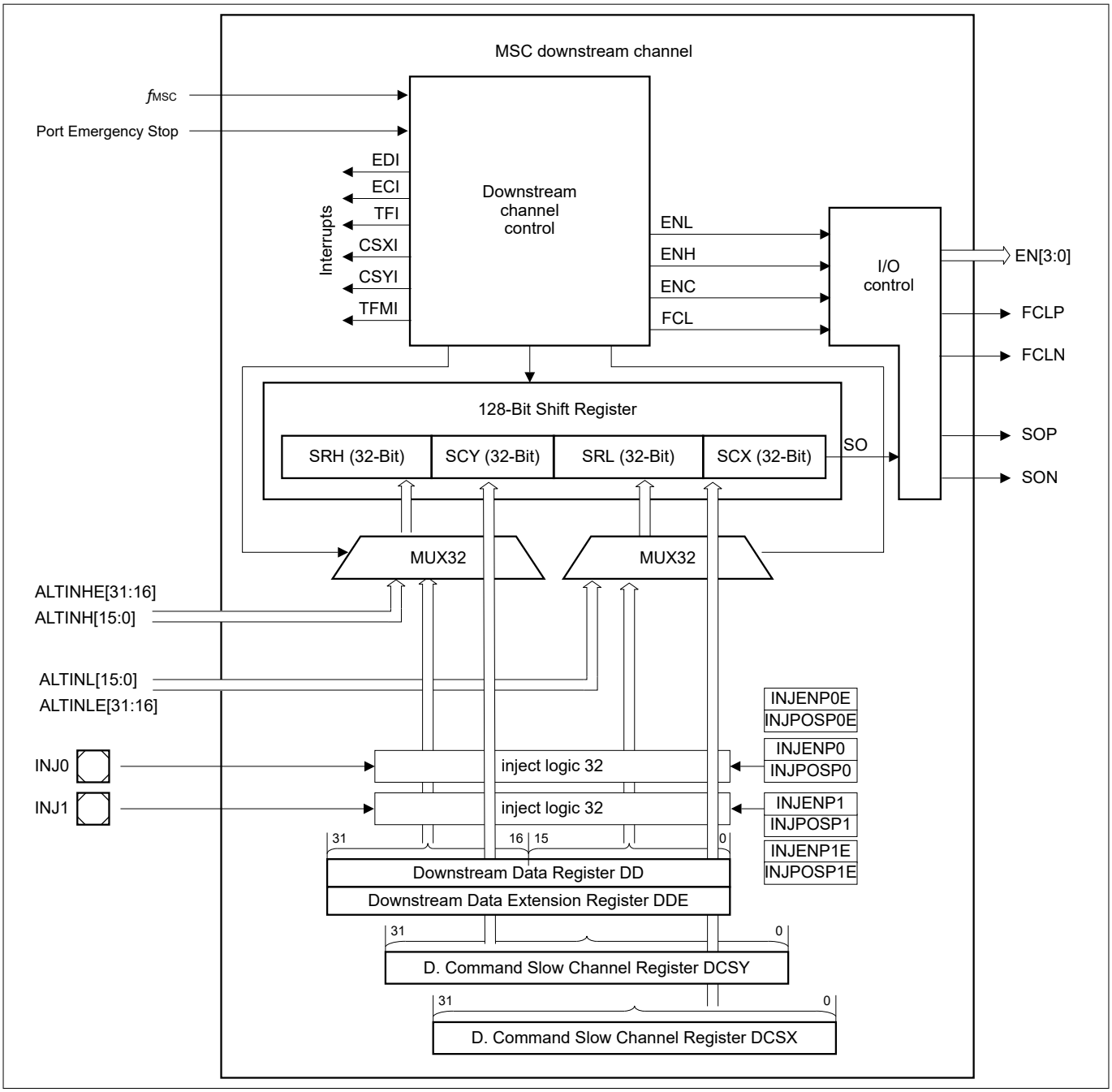


Figure 570 Downstream channel block diagram

***Note**: Refer to the block diagram in the Downstream channel with slow channel feature chapter when operating with slow channel.*

The enable signals ENL, ENH and ENC indicate certain phases of the serial transmission in relation to the serial clock FCL. In the I/O control logic, these signals can be combined to four enable/select outputs EN[3:0]. For supporting differential output drivers, the serial clock output FCL and the serial data output SO are available in both polarities, indicated by the signal name suffix “P” and “N”.

The Port Emergency Stop input is used to indicate an emergency stop condition of a power switch device. In an emergency case, shift register bits can be loaded bit-wise from the downstream data register instead from the ALTINL, ALTINH, ALTINLE and ALTINHE buses.

Related information

Downstream channel with slow channel feature on page 5110

28.3.1.1.1 Frame formats and definitions

This section describes the frame formats and definitions of the MSC.

Basic definitions

A downstream frame is composed of:

- An active phase
  - During the active phase data transmission takes place
- A passive phase
  - During the passive phase no data is transmitted at SO

The active phase is split into two parts:

- The SRL active phase in which the content of the shift register low part SRL is transmitted
- The SRH active phase in which the content of the shift register high part SRH is transmitted

At the beginning of the SRL and SRH active phase, a Selection bit (SELL) can be optionally inserted into the serial data stream. In the frame shown in the figure below, SELL is generated at the beginning of the SRL active phase (not for the SRH active phase). The least significant bits of SRL and SRH are sent out first.

The figure below shows the layout and definitions of a downstream frame.

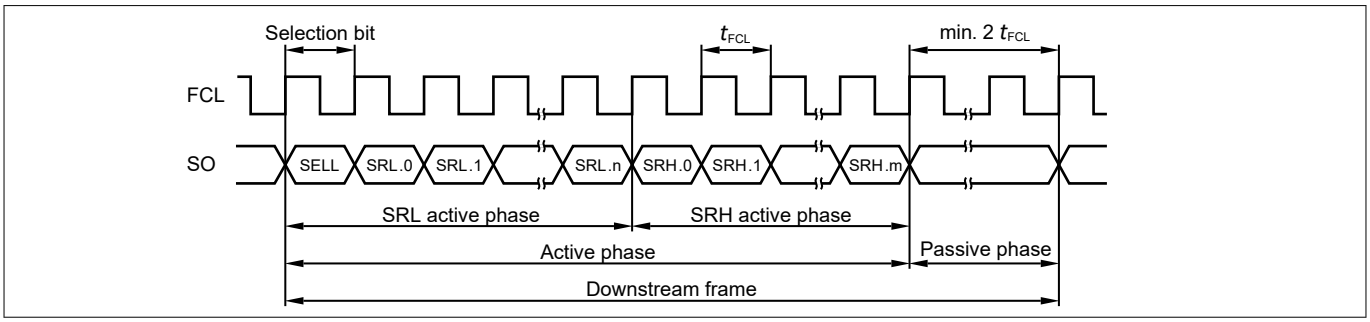


Figure 571 Downstream channel frame

The MSC downstream channel uses three types of frame formats for operation:

- Command frames, indicated by SELL = 1
- Data frames, indicated by SELL = 0 or SELL bit insertion disabled
- Passive time frame, indicated by ENL=ENH = 0

Command frames

A command frame has two active phase parts:

- SRL active phase
- SRH active phase

The command frame always starts with a High-level Selection bit, independently of whether the Selection bit insertion (as defined by bit DSC.ENSELL) is enabled or not. The number of bits transmitted during SRL and SRH active phases (except the Selection bit) is defined by bit-field DSC.NBC. SRL and SRH bits combined length can be selected from 0 upto 32 bits. SRH bits transmission starts only after SRL bits transmission is completed. In Command Extension mode, activated with the DSTE.CX bit, the commands can be up to 64-bit long (see CX (command extension) mode section).

During the active phase of a command frame the enable output signal ENC becomes active. The enable output signals ENL and ENH remain inactive.

The passive phase of a command frame always has a fixed length of $2 \times t_{FCL}$. The diagram shown in the figure below assumes that the FCL clock is only generated during the active phase of the command frame (OCR.CLKCTRL=0).

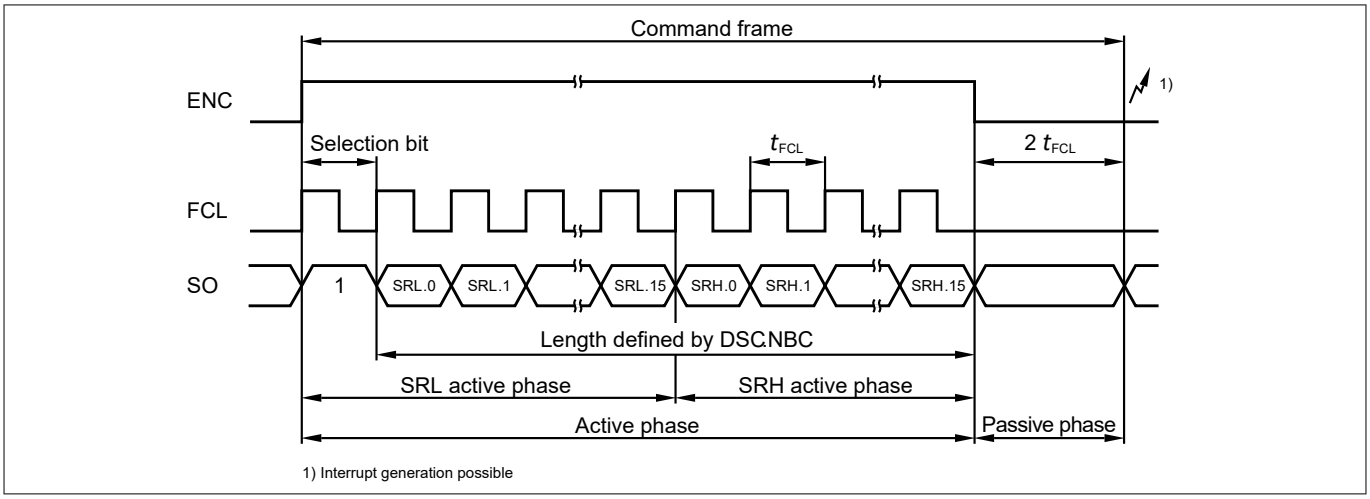


Figure 572 Command frame layout

The following table shows the programming of the bits to be transmitted and the resulting length of the complete command frame.

Table 1206 Command frame length

DSC.NBC	SRL/SRH bits transmitted in active phase	Command frame length in t_{FCL} periods
000000_B	No bit shifted out	1+0+2=3
000001_B	SRL[0] shifted out	1+1+2=4
000010_B	SRL[1:0] shifted out	1+2+2=5
000011_B	SRL[2:0] shifted out	1+3+2=6
…	…	…
001111_B	SRL[14:0] shifted out	1+15+2=18
010000_B	SRL[15:0] shifted out	1+16+2=19
010001_B	SRL[15:0] and SRH[0] shifted out	1+17+2=20
010010_B	SRL[15:0] and SRH[1:0] shifted out	1+18+2=21
010011_B	SRL[15:0] and SRH[2:0] shifted out	1+19+2=22
…	…	…
011111_B	SRL[15:0] and SRH[14:0] shifted out	1+31+2=34
100000_B	SRL[15:0] and SRH[15:0] shifted out	1+32+2=35
Other NBC combinations	Reserved; do not use these bit combinations	

Data frames

A data frame has two active phase parts:

- SRL active phase
- SRH active phase

The number of bits that are transmitted can be programmed separately for each of these two phases.

- Bit-field DSC.NDBL determines the number of SRL bits that are transmitted during the SRL active phase
- Bit-field DSC.NDBH determines the number of SRH bits that are transmitted during the SRH active phase

SRL and SRH active phases can start with a Low-level Selection bit when enabled by bits DSC.ENSELL or DSC.ENSELH.

During the SRL active phase of a data frame the enable output signal ENL becomes active and during the SRH active phase of a data frame, the enable output signal ENH becomes active. The enable output signal ENC remains inactive.

The length of the data frame's passive phase is variable and is defined by the bit-field DSC.PPD. It can be within a range of $2 \times t_{FCL}$ up to $31 \times t_{FCL}$. The diagram shown in the figure below assumes that the FCL clock is only generated during the active phase of the data frame (OCR.CLKCTRL=0).

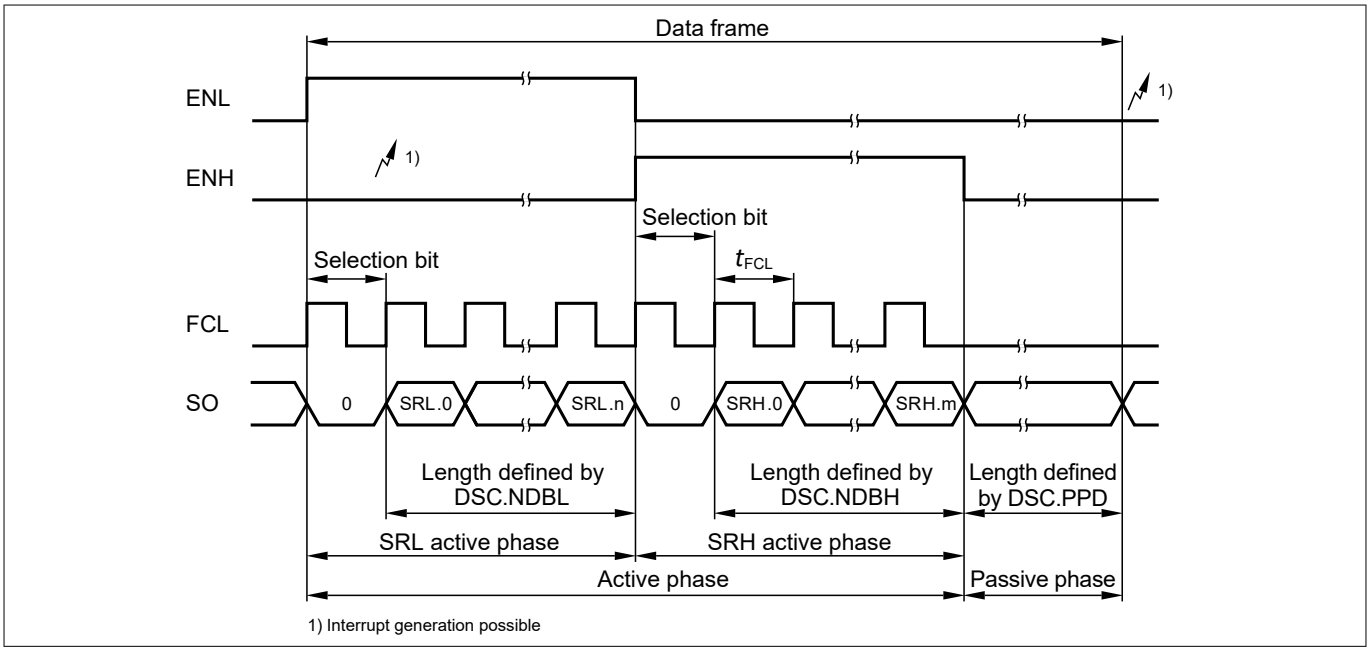


Figure 573 Data frame layout

The following tables show the definitions of the five data frame parameters that determine the layout of the data frame.

Table 1207 Data frame selection bit parameters

DSC.ENSELL	Selection bit	DSC.ENSELH	Selection bit
0	No Selection bit inserted at the beginning of the SRL active phase	0	No Selection bit inserted at the beginning of the SRH active phase
1	A Low-level Selection bit is inserted at the beginning of the SRL active phase	1	A Low-level selection bit is inserted at the beginning of the SRH active phase

***Note**: The MSC module can operate in two modes: Standard (up to 32 data bits) or Extended (up to 64 data bits). The mode is selected by using the bit-field DSCE.EXEN.*

Table 1208 Data frame SRL/SRH length parameters, Standard mode

DSCE.NDBLE=0 and DSC.NDBL	SRL bits transmitted in SRL active phase	DSCE.NDBHE=0 and DSC.NDBH	SRH bits transmitted in SRH active phase
$0\ 00000_B$	No SRL bit transmitted	$0\ 00000_B$	No SRH bit transmitted
$0\ 00001_B$	SRL[0]	$0\ 00001_B$	SRH[0]

(table continues...)

Table 1208 (continued) Data frame SRL/SRH length parameters, Standard mode

DSCE.NDBLE=0 and DSC.NDBL	SRL bits transmitted in SRL active phase	DSCE.NDBHE=0 and DSC.NDBH	SRH bits transmitted in SRH active phase
$0\ 00010_B$	SRL[1:0]	$0\ 00010_B$	SRH[1:0]
$0\ 00011_B$	SRL[2:0]	$0\ 00011_B$	SRH[2:0]
…	…	…	…
$0\ 01111_B$	SRL[14:0]	$0\ 01111_B$	SRH[14:0]
$0\ 10000_B$	SRL[15:0]	$0\ 10000_B$	SRH[15:0]
Other bit combinations	Reserved; do not use these bit combinations	Other bit combinations	Reserved; do not use these bit combinations

Table 1209 Data frame SRL/SRH length parameters, Extended mode

DSCE.NDBLE=1 and DSC.NDBL	SRL bits transmitted in SRL active phase	DSCE.NDBHE=1 and DSC.NDBH	SRH bits transmitted in SRH active phase
$1\ 00000_B$	No SRL bit transmitted	$1\ 00000_B$	No SRH bit transmitted
$1\ 00001_B$	SRL[16:0]	$1\ 00001_B$	SRH[16:0]
$1\ 00010_B$	SRL[17:0]	$1\ 00010_B$	SRH[17:0]
$1\ 00011_B$	SRL[18:0]	$1\ 00011_B$	SRH[18:0]
…	…	…	…
$1\ 01111_B$	SRL[30:0]	$1\ 01111_B$	SRH[30:0]
$1\ 10000_B$	SRL[31:0]	$1\ 10000_B$	SRH[31:0]
Other bit combinations	Reserved; do not use these bit combinations	Other bit combinations	Reserved; do not use these bit combinations

Table 1210 Data frame passive phase length

DSC.PPD	Passive phase length
00000_B	$2 \times t_{FCL}$
00001_B	$2 \times t_{FCL}$
00010_B	$2 \times t_{FCL}$
00011_B	$3 \times t_{FCL}$
…	…
11110_B	$30 \times t_{FCL}$
11111_B	$31 \times t_{FCL}$

The following formula determines the number of t_{FCL} cycles of data frame up to 32-bit in size (all parameters, bits and bit-fields, are located in register DSC):

$$N_{CYCLES} = \text{ENSELL} + \text{NDBL} + \text{ENSELH} + \text{NDBH} + \text{PPD} \quad (29)$$

***Note**: In the formula above, PPD must be set to 2 when DSC.PPD ≤ 00010_B.*

The following formula determines the number of t_{FCL} cycles of an extended, up to 64-bit data frame, in case DSC.NDBL and DSC.NDBH are not equal to zero:

$$N_{CYCLES}=ENSELL + NDBL + NDBLE \times EXEN \times 16 + ENSELH + NDBH + NDBHE \times EXEN \times 16 + PPD + PPDE \times EXEN \times 32 \quad (30)$$

Otherwise, the corresponding factor $NDBLE \times EXEN \times 16$ or $NDBHE \times EXEN \times 16$ must be taken as zero.

***Note**: In the formula above, PPD must be set to 2 when DSC.PPD ≤ 00010_B.*

The parameters (bits and bit-fields) are located in the registers DSC and DSCE.

Attention**: **In order to calculate the time between two consecutive TRPs (time reference points), always use ENSELL=1 in the formulas above. ENSELL does not influence the time between two consecutive TRPs, but only the length of a data frame. The difference in the starting point of a frame with ENSELL=0 and 1 is shown in related information.

***Note**: Refer to the Slow channel or Manchester code transmission mode sections when operating in these modes. The equation to calculate the time reference point differs from the one operating in legacy mode.*

Passive time frames

A passive time frame has the length defined by the five data frame parameters according to the equations above. They are generated only in Data Repetition mode. Under special conditions (command frame insertion), passive time frames can be shortened.

During passive time frames, the data output SO has to be considered as invalid at the receiving device and the clock output FCL may toggle or not (as selected by bit OCR.CLKCTRL). The ENL and ENH enable signals remain at low level during a passive time frame.

***Note**: A passive time frame and a passive phase are different from each other. A passive phase is part of a downstream frame whereas a passive time frame(PTF) is a type of frame which is transmitted after the transmission of downstream frame is finished.*

Related information

Downstream counter and enable signals on page 5095
CX (command extension) mode on page 5103
Slow channel on page 5107
Manchester code transmission mode on page 5113

28.3.1.1.2 Transmission modes

The downstream channel of the MSC makes it possible to select between two transmission modes:

- Triggered mode, selected by DSC.TM=0 or
- Data Repetition mode, selected by DSC.TM=1

  ***Note**: The behavior is undefined when the configuration is changed in between an ongoing transaction*

Triggered mode

In Triggered mode, command frames or data frames are sent out as a result of a software event. When a frame transmission has been finished and no further frame transmission has been requested, the downstream channel returns to idle state and waits for the next frame transmission to be triggered by software.

When the Downstream Command Register DC is written, the command pending bit DSC.CP becomes set and a command frame will be immediately started and sent out if the downstream channel is idle. If a data or

command frame is currently processed and output, the command frame transmission is delayed and started when the active downstream frame has been finished. The command pending bit DSC.CP will be cleared by hardware.

If the downstream channel is idle and the data pending bit DSC.DP is set by writing bit ISC.SDP with 1, a data frame will be immediately started and sent out if the downstream channel is idle. If a data frame or a command frame is currently processed and output, the data frame transmission is delayed and started when the active downstream frame has been finished. The data pending bit DSC.DP will be cleared by hardware.

If the MSC configuration is set for command frame prioritization over data frame (DSCE.DFP = 0) then both frame pending bits are set (DSC.DP=DSC.CP=1), the command frame will always be sent first. Therefore, a pending data frame transmission will be delayed as long as no further command frame transmission is running or requested.

The following figure is a flow diagram of the Triggered mode. This diagram especially shows the behavior of the data and command pending bits DSC.DP and DSC.CP. If both frame pending bits are set (DSC.DP=DSC.CP=1), the command frame will always (if DSCE.DFP = 0) be sent first, followed by the data frame (assuming no further command frame has been requested).

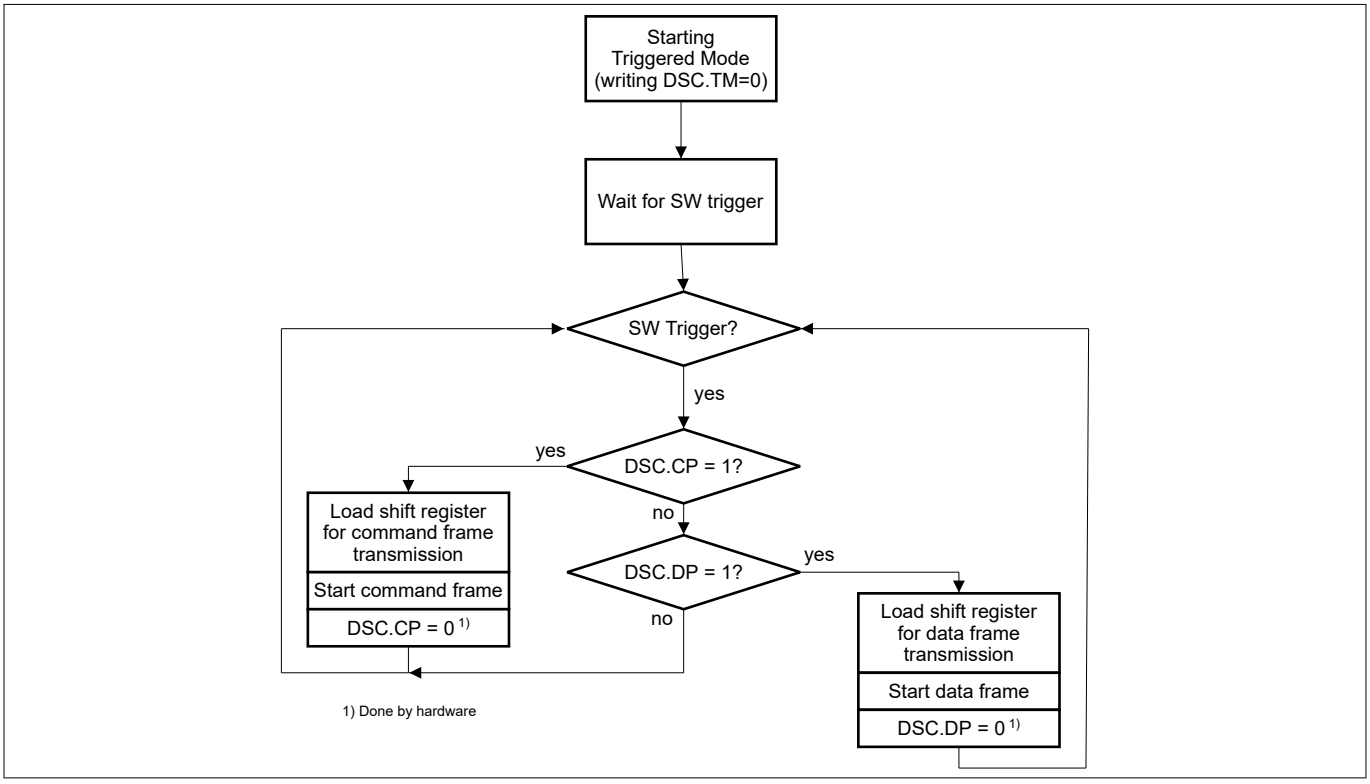


Figure 574 Triggered mode flow diagram

The type of the active frame that is currently processed and output is indicated by two status flags: DSS.DFA is set during a data frame transmission and DSS.CFA is set during a command frame transmission. Further, the downstream counter DSS.DC indicates the number of shift clock periods that have been elapsed since the start of the current frame.

In Triggered mode, the shift register loading event, as described in the SRL and SRH shift register loading chapter, occurs just before a command or data frame transmission is started.

Data repetition mode

Setting the DSC.TM bit selects Data Repetition mode. In Data Repetition mode, data frames are sent out continuously without any software interaction. In the time gap between two consecutive data frames, passive time frames can be inserted. The number of passive time frames to be inserted (0 to 15) is defined by bit-field DSS.NPTF. The duration of data frame (t_{DF}) and passive time frames (t_{PTF}) is determined by the data frame parameters as described in the previous chapter. These parameters determine the time reference points (TRP)

at which a data, command or passive time frame is started (see diagram A in the following figure and the Attention note in the Frame formats and definitions chapter).

The automatic data frame generation is controlled by the data pending bit DSC.DP. This bit is set at the end of the last passive time frame if a command frame is running or starts at this TRP, overruling a data frame. DSC.DP is cleared by hardware when the pending data frame starts with transmission at the next TRP. Data frames are always aligned to time reference points. This means they always start at a TRP. Passive time frames can be shortened. This is especially the case when command frames are inserted.

Continuous data frame transmission can be interrupted by insertion of command frames. Command frames are initiated by software. When the downstream command register DC is written, the command pending bit DSC.CP is set by hardware. CP=1 indicates that the MSC starts a command frame at the next TRP, independently of whether a data frame (indicated by DSC.DP=1) or passive time frame should be started with the next TRP. This means also that command frames are always aligned to time reference points.

The bit-field DSCE.CDCM provides an option for automatic insertion of data frame between two consecutive command frames.

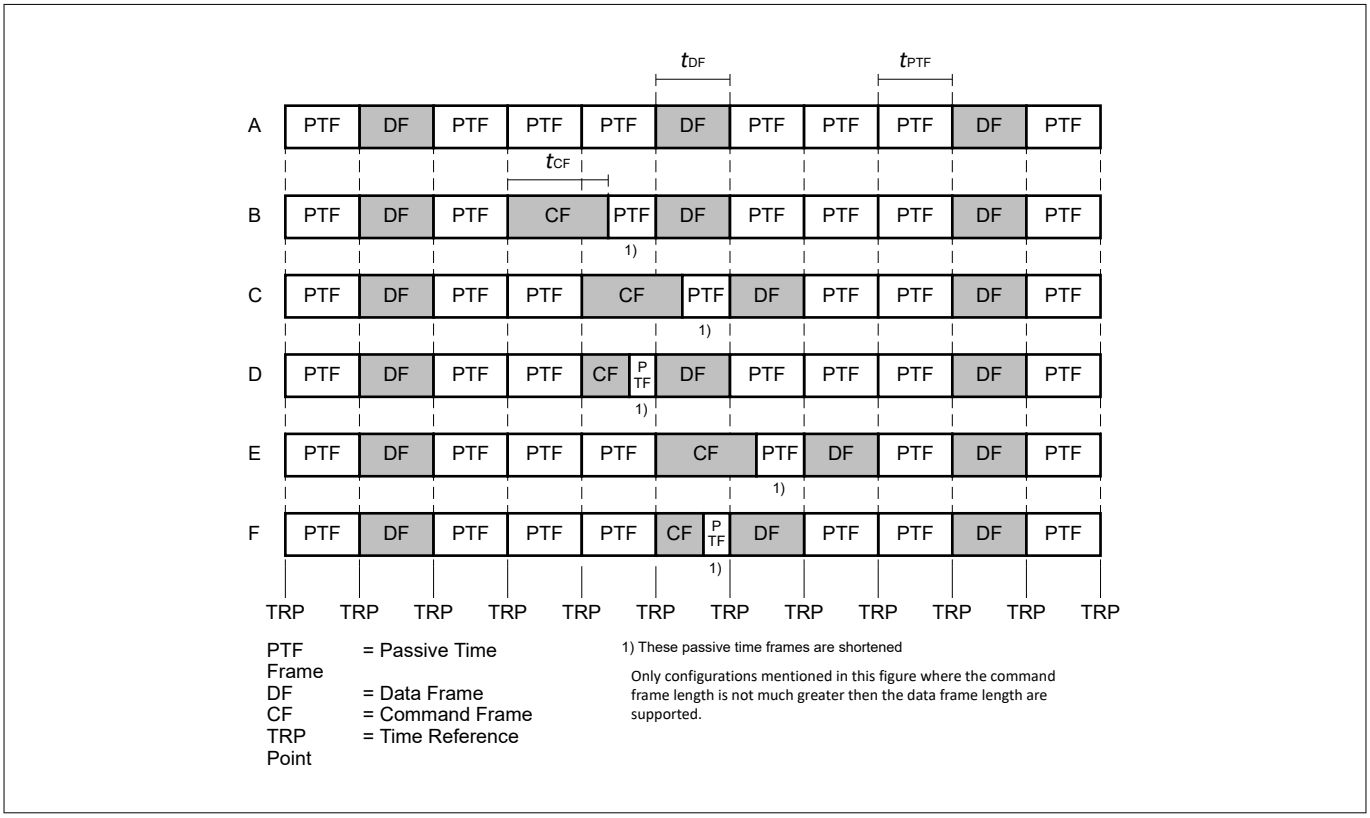


Figure 575 Data repetition mode frame examples with DSS.NPTF=0011$_B$

Diagrams B to F in the figure above show the command frame insertion in Data Repetition mode.

In diagram B, a command frame has been requested during the first passive time frame after the data frame and is inserted at the next TRP. In diagrams C and D, a command frame has been requested during the second passive time frame and is inserted at the time reference point of the last nominal passive time frame.

When the command frame and data frame is not of the same length (this is the case in diagram B to F), a shortened passive time frame is inserted until the next TRP is reached. This ensures that the next data or normal passive time frame is again aligned to a TRP.

When the last passive frame is transmitted and in case of a conflict with command frame, DSC.DP becomes set by hardware. This triggers the start of a data frame when the next TRP is reached.

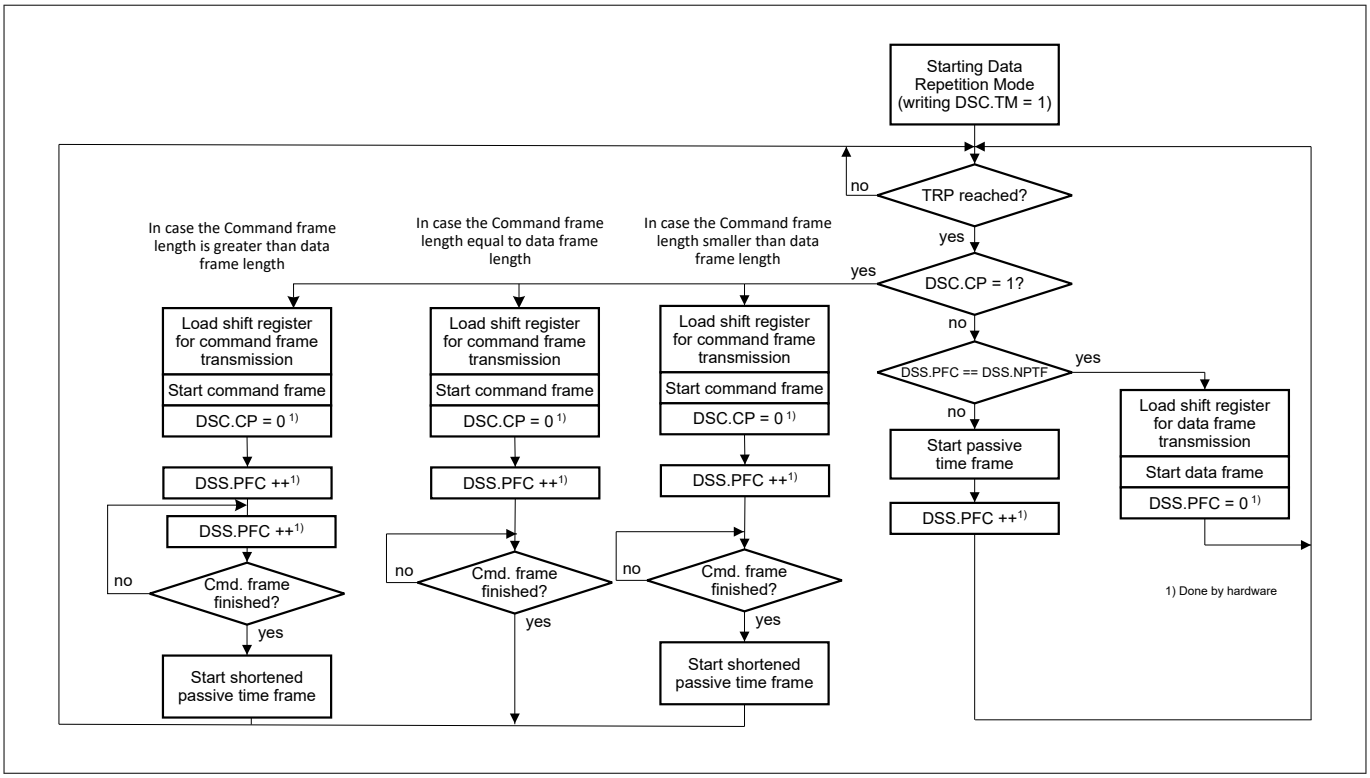


Figure 576 Data repetition mode flow diagram

The type of the active frame (data or command frame) that is currently processed and output is indicated by two status flags: DSS.DFA is set during a data frame transmission and DSS.CFA is set during a command frame transmission. Further, the downstream counter DSS.DC indicates the number of shift clock periods that have been elapsed since the start of the current data, command or passive time frame.

As in Triggered mode, the shift register loading event as described in related information occurs in Data Repetition mode just before a TRP, this means shortly before a command or data frame transmission is started.

Data frame prioritization over command frame

The bit-field DSCE.DFP provides an option to prioritize the data frame over the command frame.

This change is illustrated in diagram C, E and F by enabling this feature in bit field DSCE.DFP.

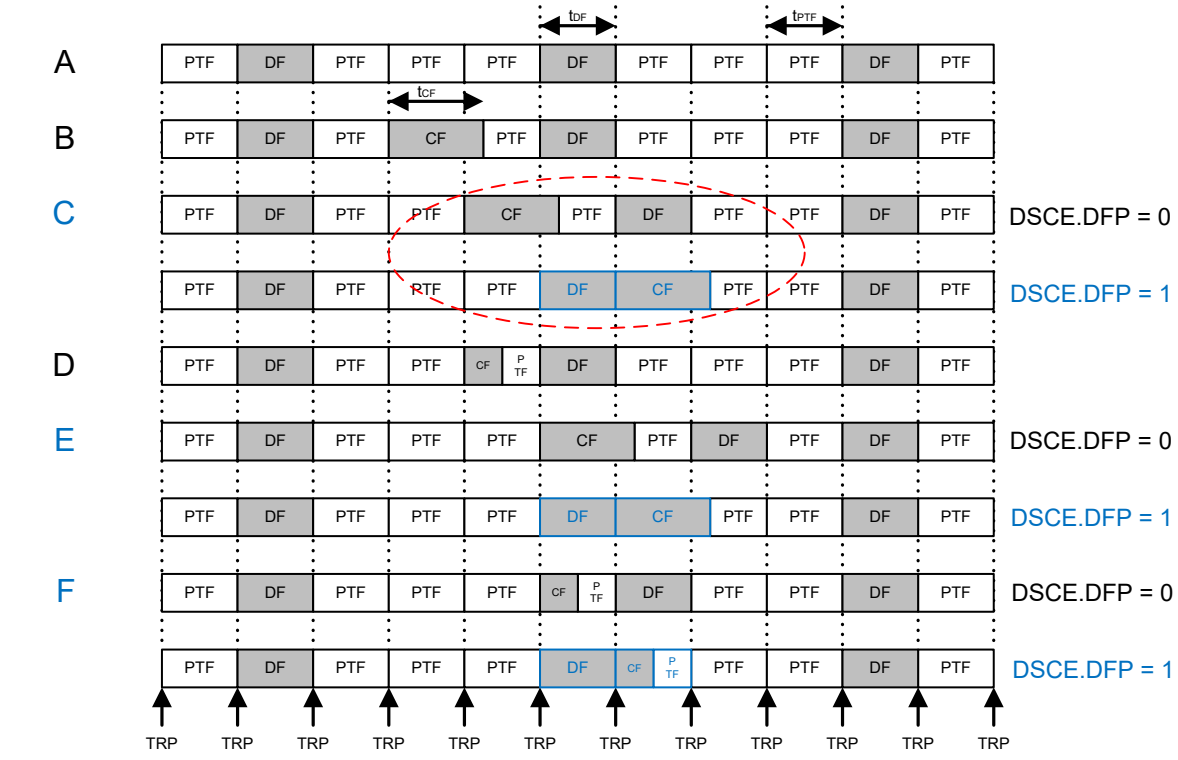


Figure 577 Data frame prioritization over command frame

If there is a request to send a command frame at a TRP where a data frame should be sent and DSCE.DFP is enabled, the data frame is prioritized over the command frame. This means the transmission of the command frame is delayed until the next TRP and the data frame is sent. This allows to guarantee equidistant data frames.

This feature, data frame prioritization over command frame can be used in both transmission modes: Triggered mode and Data Repetition mode.

The MSC does not provide protection against misconfiguration, when a command frame is longer than the longest passive time frame (PTF).

The following figure is a flow diagram of the Triggered mode when the data frame prioritization over command frame is active.

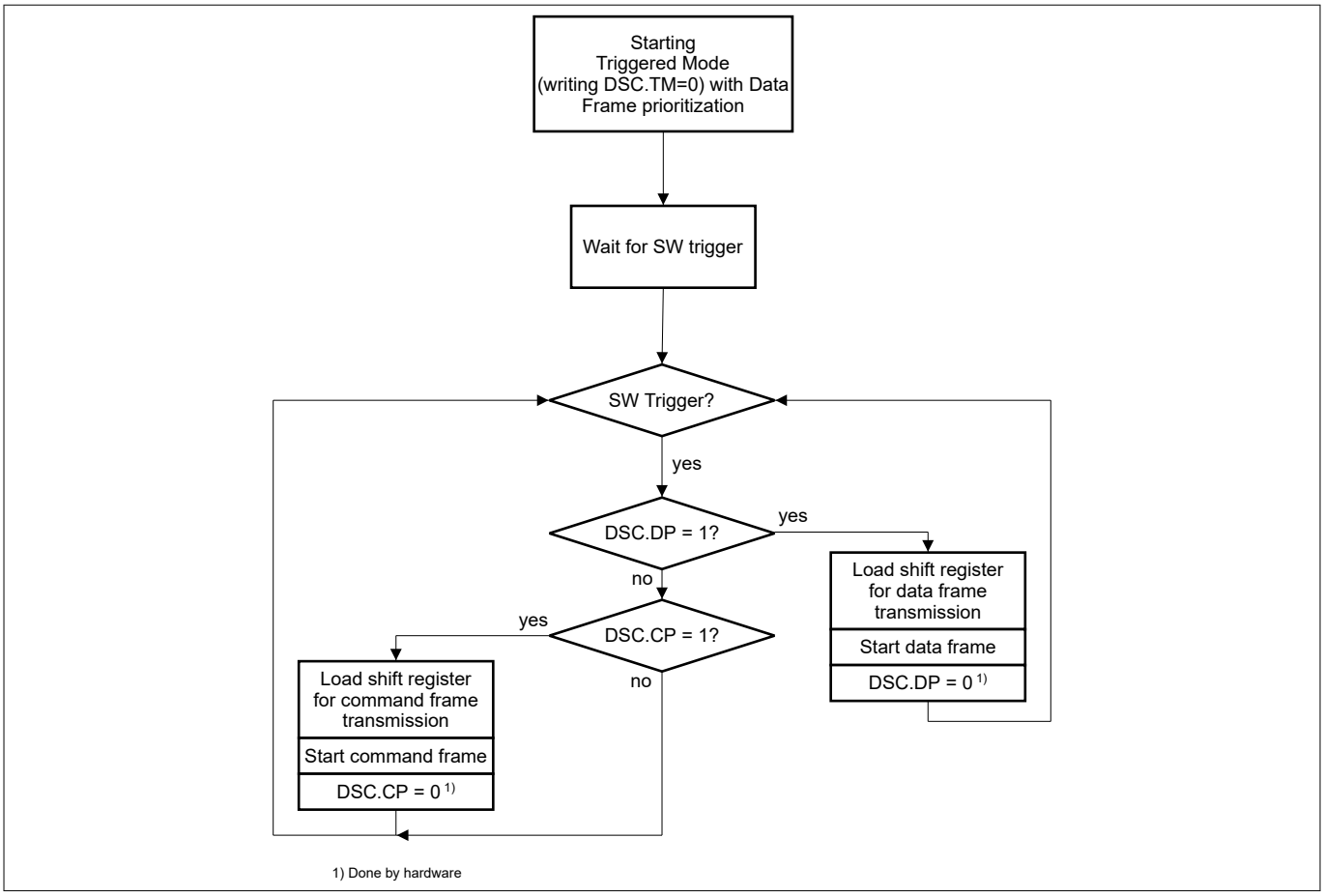


Figure 578 Triggered mode flow diagram with data frame prioritization

The following figure is a flow diagram of the Data Repetition mode when the data frame prioritization over command frame is active.

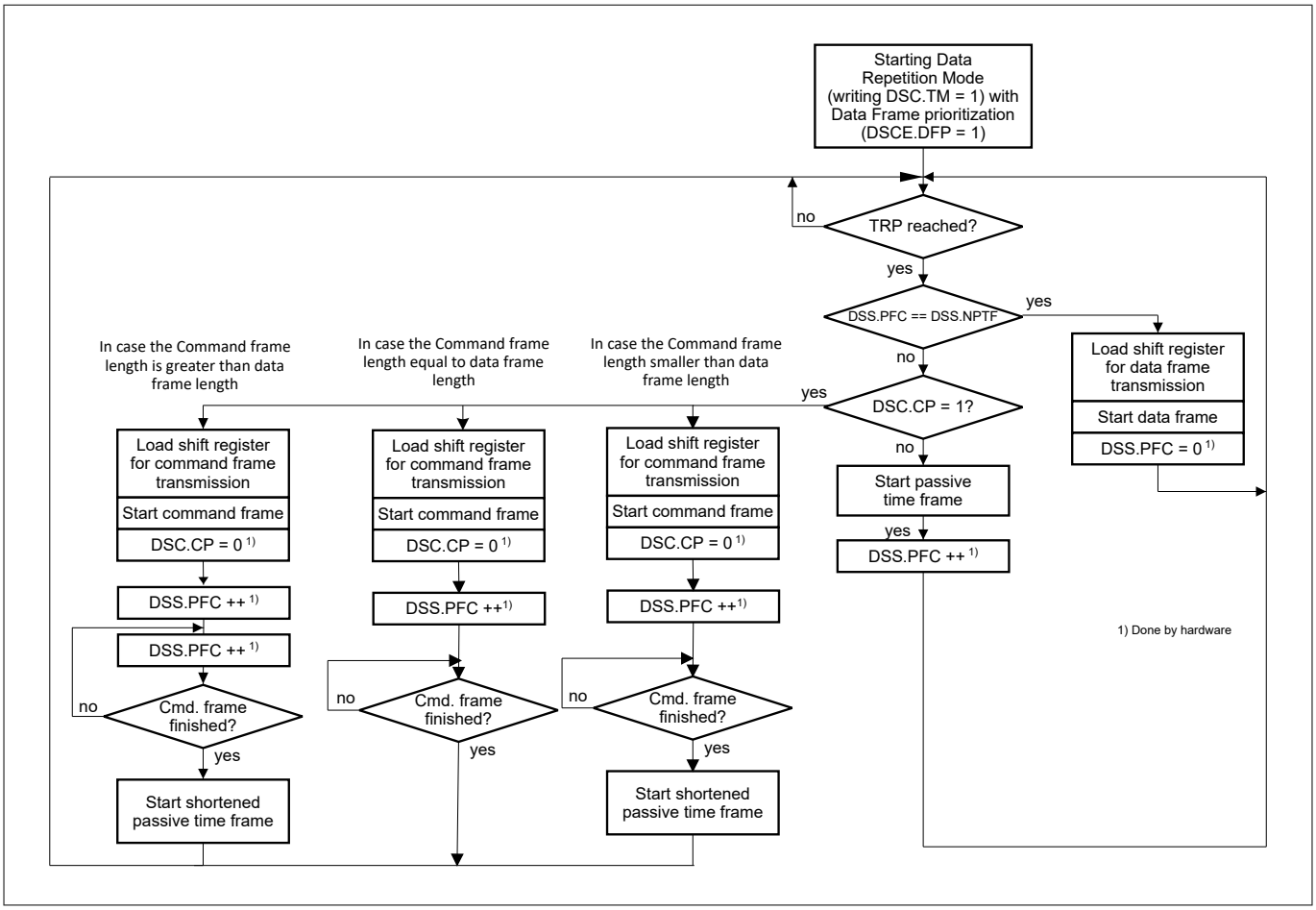


Figure 579 Data repetition mode flow diagram with data frame prioritization

Passive frame counter in data repetition mode

In Data Repetition mode, a passive time frame counter DSS.PFC indicates how many time frames have been already transmitted after the last regular data frame occurrence. The passive time frame counter counts up from 0000_B to the value which has been written into bit field DSS.NPTF (number of passive time frames). DSS.PFC=0000_B indicates that a data frame is requested for transmission.

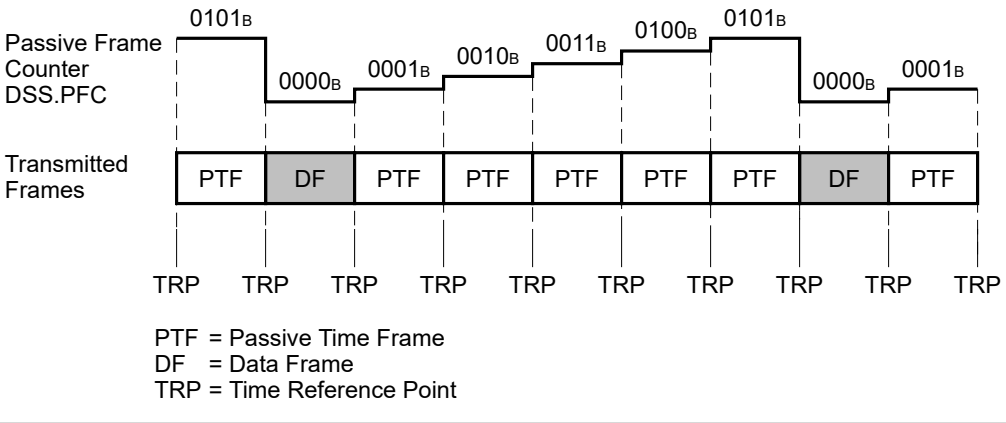


Figure 580 Passive frame counter operation (with DSS.NPTF=0101_B)

Related information

SRL and SRH shift register loading on page 5093

Frame formats and definitions on page 5082

28.3.1.1.3 SRL and SRH shift register loading

During the SRL and SRH shift register load operation at the beginning of each downstream frame transmission, several parameters determine which information is loaded into the bits of the shift register.

SRL shift register loading

The following figure shows the logic that is implemented for the SRL shift register loading operation. The logic for the SRH shift register loading operation is equivalent to the one for the SRL register. Its differences in data sources and register controls are described below.

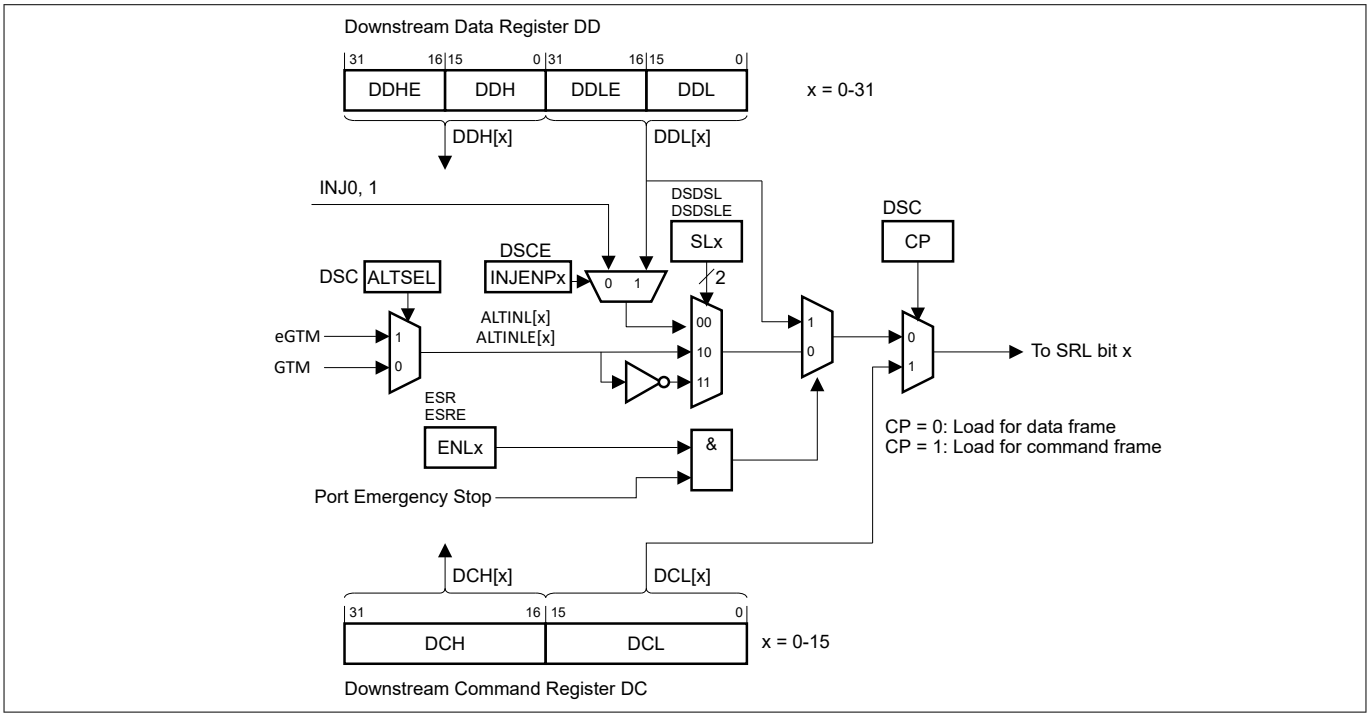


Figure 581 SRL shift register data loading control

Four data sources can be selected for each SRL bit by using several control bits and one control signal:

- ALTINL, ALTINLE input line (non-inverted)
- ALTINL, ALTINLE input line (inverted)
- Bit of DD.DDL, DDLE (Downstream Data Register)
- Bit of DC.DCL (Downstream Control Register)

GTM or eGTM can be selected (DSC.ALTSEL) as data source for ALTINL and ALTINLE.

When SRL is loaded for data frame transmission (DSC.CP=0), bit-fields DSDSL.SLx determine bit-wise which data is loaded into SRL bit x. The data source selection as controlled by DSDSL.SLx, will only be effective when Port Emergency Stop is inactive (at low level). When Port Emergency Stop=1 (active) during the load operation, all SRL[x] bits that are enabled for the emergency stop feature (bit ESR.ENLx=1) are loaded directly with the corresponding bit DDL[x] of the Downstream Data Register DD.

When SRL is loaded for command frame transmission (DSC.CP=1), always the lower 16-bit part DCL of the downstream control register is loaded completely into SRL.

The following table summarizes all SRL data source selection capabilities $(x = 0 - 31)$:

- In Standard mode, DSCE.EXEN = 0, the selection configuration of up to 16 bits $(x = 0 - 15)$ is taken into account
- In Extended mode, DSCE.EXEN = 1, the selection configuration of up to 32 bits $(x = 0 - 31)$ is taken into account

Table 1211 SRL data source selection capabilities

DSC. CP	DSDSL.SLx DSDSLE.SLx	ESR. ENLx	Port emergency stop	Selection
0	00_B	0	–	Bit DD.DDL[x] is loaded into SRL[x]
	01_B			Reserved
	10_B			State of ALTINL[x] input is loaded into SRL[x]
	11_B			Inverted state of ALTINL[x] input is loaded into SRL[x]
	XX_B	1	1	Bit DD.DDL[x] is loaded into SRL[x]
1	XX_B	X	X	Bit-fields DCL and DCH are completely loaded into SRL and SRH, respectively

***Note**: The data signals can be overruled by injected pin signals INJ0 and INJ1. See DSCE register.*

SRH shift register loading

The SRH operation is equivalent to the SRL shift register load operation, but the following differences must be taken into account for SRH shift register loading:

- Input lines ALTINH/ALTINHE are connected instead of ALTINL/ALTINLE input lines
- DSDSH register bits control data source selection instead of the DSDSL register
- Emergency stop is enabled by ESR.ENHx bits instead of ESR.ENLx bits
- Bits of the Downstream Data Register high part DDH are selected instead of DDL
- Downstream Control Register high part DCH is selected instead of DCL

28.3.1.1.4 External signal injection

Input signals from two pins can be independently injected at two data bit positions in a data frame. The injection takes place both in Normal and in Extended mode. The external signals are always injected at the defined position and if the data frame is shorter than this position, than these signals remain unused. For example, if DSCE.INJPOS0 defines injection at the position 14 of the SRL and the DSC.NBDL=12, then all the data bits from position 12 and up remain unused, including the injected signal.

In Normal mode the bits are counted from 0 to 31 from LSB to MSB and in Extended mode the bits are counted from 0 to 63 from LSB to MSB (see following figure). The injection position is always counted as if the frame has the maximum length. In the first step of the preparation for a transmission the injection takes place and then in the second step, if some fields are configured to be shorter or omitted (for example DDLE or DDH or any other), the unused bits are discarded and only the remaining valid bits are shifted out.

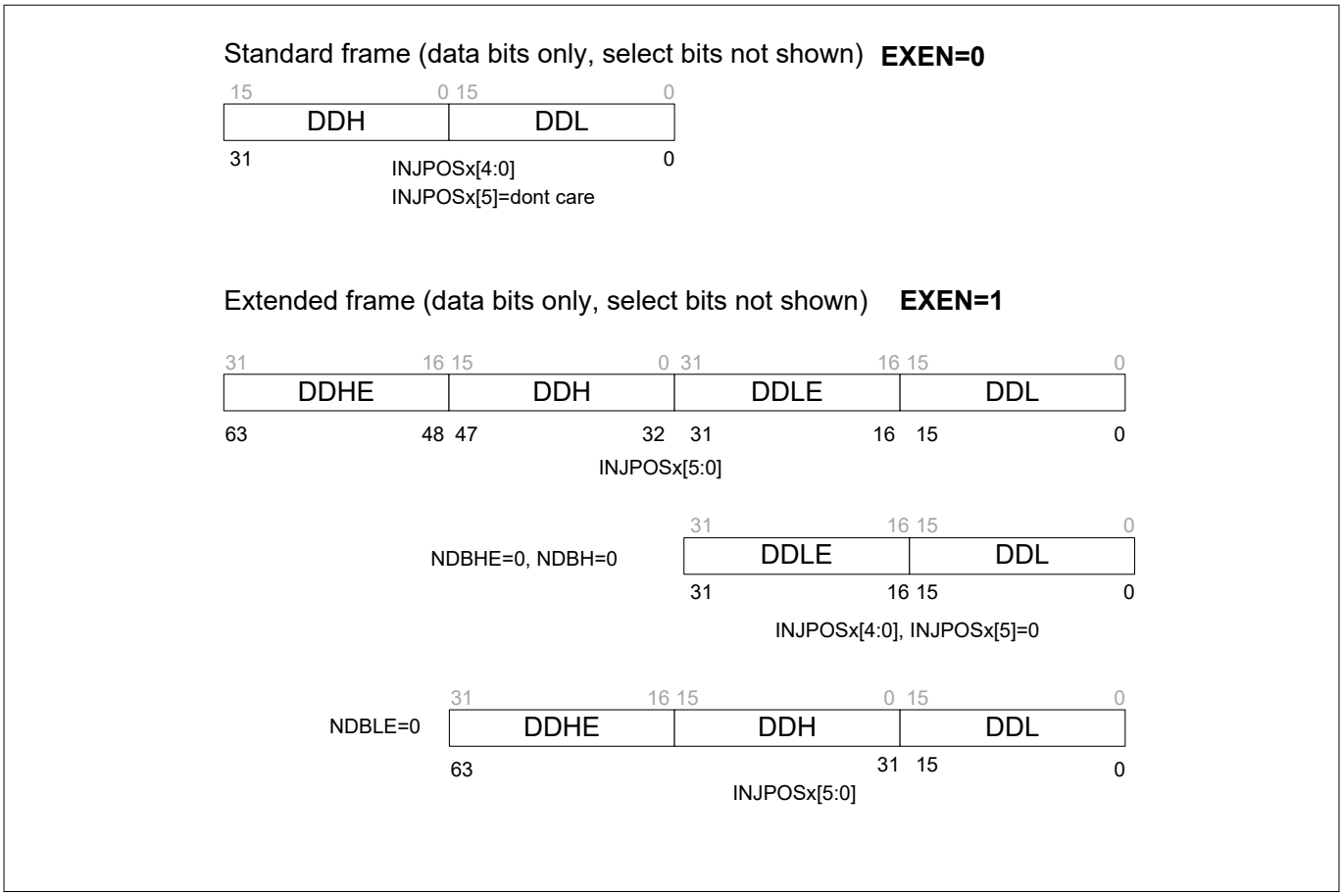


Figure 582 External signal injection

28.3.1.1.5 Downstream counter and enable signals

During downstream channel operation, a 8-bit downstream counter DSS.DC is counting FCL shift clock periods. With the loading of the shift register, the downstream counter is reset to 00_H and started for counting up to the end of the downstream frame (end of passive phase).

- Triggered mode
  - The downstream counter stops counting at the end of the passive phase and waits until a new downstream frame is started
- Data Repetition mode
  - The downstream counter does not stop at the end of the passive phase but is reset and starts counting up again with the next frame, independently of whether a data frame, command frame or passive time frame is started as the next frame

The following figure shows an example of downstream channel data frame transmission. In this example, the Selection bit for the SRL active frame is enabled (ENSELL=1) and the Selection bit for the SRH active frame is disabled (ENSELH=0). With loading of the shift register SRL/SRH, the downstream counter is reset and then starts counting up with each FCL clock until the end of the passive phase. ENL is set to high level at the beginning of the SRL active frame Selection bit.

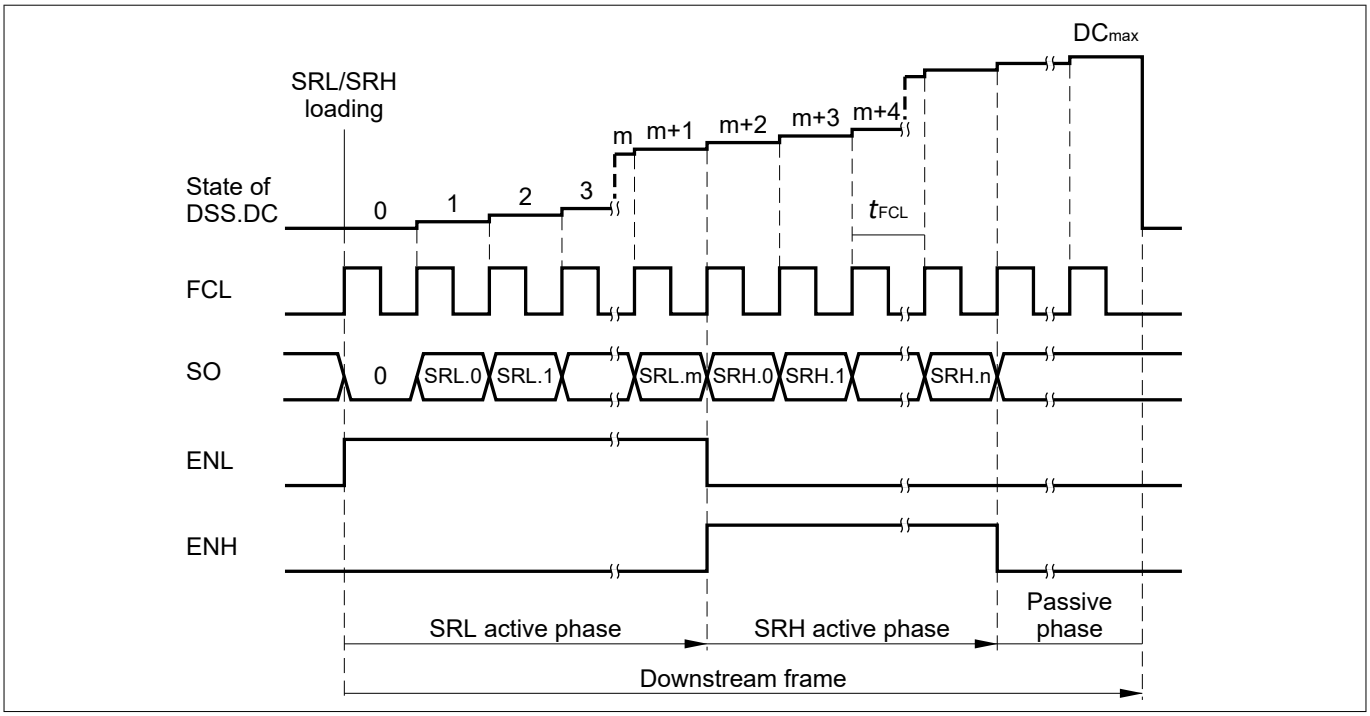


Figure 583 Shift clock counting: Data frame with ENSELL=1 and ENSELH=0

When the Selection bit for the SRL active frame is disabled (ENSELL=0, see following figure), the loading of the shift register SRL/SRH (and reset of the downstream counter) occurs one FCL clock cycle before the first data bit SRL.0 is output. ENL is set to high level with the beginning of the first data bit SRL.0.

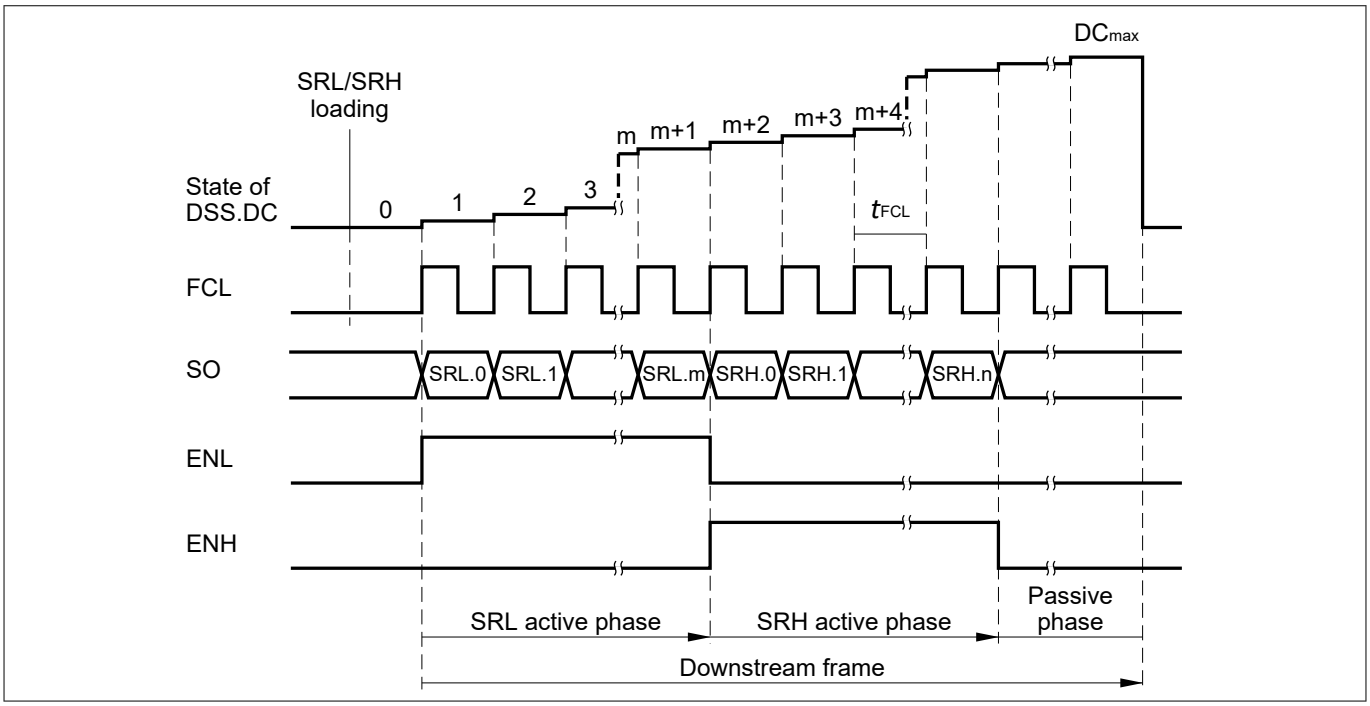


Figure 584 Shift clock counting: Data frame with ENSELL=0 and ENSELH=0

28.3.1.1.6 Downstream channel baud rate

The baud rate of the downstream channel's serial transmission is defined by the frequency of the serial clock output (FCL) and is always $\frac{f_{MSC}}{2}$.

The f_{MSC} generation is device specific and depends on the implementation of the MSC module.

Related information

Clock control on page 5132

28.3.1.1.7 Aborting frames

Only a reset condition of the device can abort a current transmission. The MSC module will not start a new frame transmission when:

- The downstream channel becomes disabled
- The Suspend mode is requested
- Sleep mode is entered

If one of these three conditions becomes active during a running frame transmission, the frame transmission is completely finished before the requested abort state is entered. Note that in this case no Time Frame Finished interrupt is generated.

***Note**: In case of a hard suspend, the abort state is entered immediately after being requested.*

28.3.1.2 Upstream channel

The MSC upstream channel is an asynchronous serial receiver based on the standard asynchronous data transfer protocol. It is dedicated to receive a serial data stream from a peripheral device through its serial data input SDI, using two specific data frame formats.

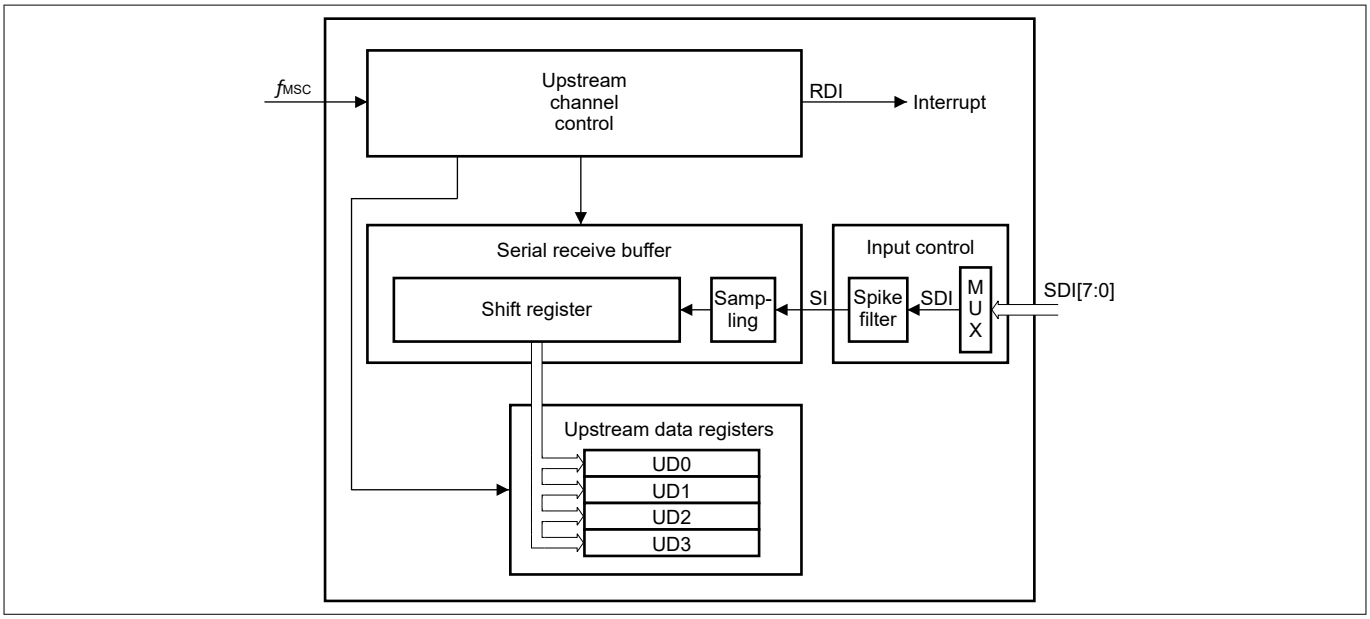


Figure 585 Upstream channel block diagram

The incoming data at SI is sampled after it has been filtered for spikes. The detected logic states of the serial input are clocked into a shift register.

After the complete reception of the serial data frame, the content of the shift register is transferred into one of the four data registers and an interrupt can be optionally generated.

The reception baud rate is directly coupled to the module clock f_{MSC} and can be within a range of $\frac{f_{MSC}}{4}$ down to $\frac{f_{MSC}}{256}$.

***Note**: If the ABRA block is used, a fractional divider with a fractional ratio generates the upstream sampling rate and in order to compensate the additional fractional divider jitter, the reception baud rate must be less than* $\frac{f_{MSC}}{10}$ *or* $\frac{f_{MSC}}{(8 \text{ to } 256)} \times \frac{n}{1024} < \frac{f_{MSC}}{10}$

28.3.1.2.1 Data frames

The asynchronous data frames used by the upstream channel include four basic parts:

1. One Start bit, always at low level
2. An optional 4-bit address field A[3:0] with LSB first
3. An 8-bit data field D[7:0] with LSB first
4. One Parity bit and two Stop bits that are always at high level

The receive interrupt is generated after the second Stop bit, but its generation can be delayed by one bit time by setting the bit-field USR.SRDC.

As shown in the following figure, the 16-bit upstream data frame includes an additional 4-bit address field. The upstream frame type is selected by bit-field USR.UFT.

- USR.UFT = 0:12-bit upstream data frame selected
- USR.UFT = 1:16-bit upstream data frame with 4-bit address field selected

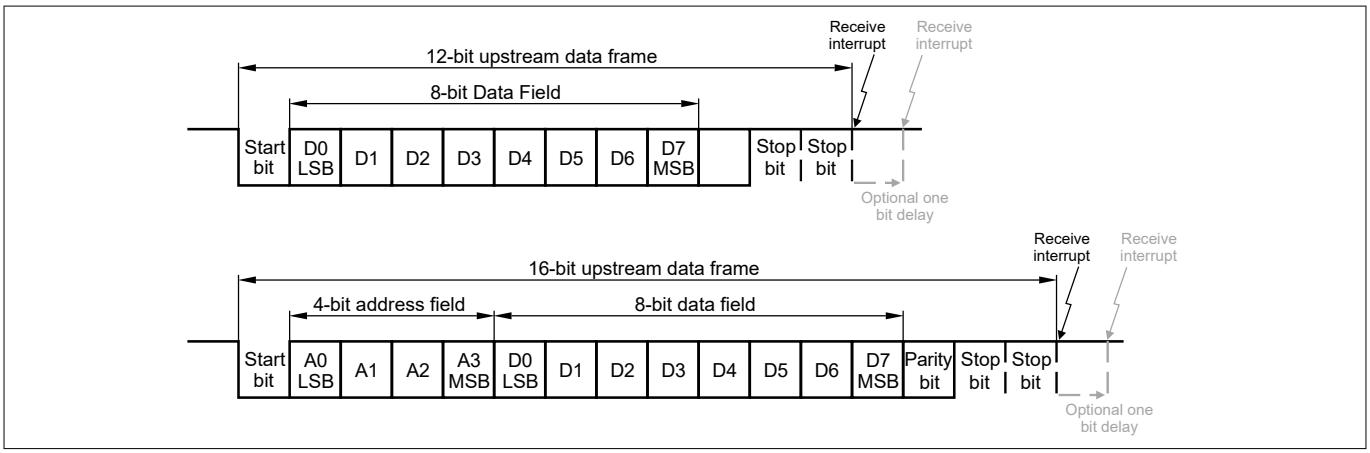


Figure 586 Upstream channel frame types

28.3.1.2.2 Parity checking

The incoming Parity bit of the data frames can be checked by the upstream channel. When a Parity error is detected, the Parity error flag PERR in the related Upstream Data Register UDx is set. Note that a setting of the Parity error flag PERR does not generate an interrupt. The PERR bits must be checked by software, data will be stored regardless of a detected Parity error. The UDx registers also store the Parity bit of the incoming data frame (UDx.P) and the Parity bit that is generated internally (UDx.IPF).

Bit USR.PCTR determines the Parity mode, even or odd, that is selected for parity checking.

- USR.PCTR = 0
  - Even Parity mode is selected. Even parity means that the Parity bit is set on an odd number of 1 s in the data field (12-bit upstream data frame) or in the address plus data field (16-bit upstream data frame)
- USR.PCTR = 1
  - Odd Parity mode is selected. In Odd Parity mode, the Parity bit is set on an even number of 1 s of the related data

The parity checking logic in the upstream channel also controls whether the Start bit and the two Stop bits of the upstream data frame are at the correct logic level. If the Start bit is not at a low level and the two Stop bits are not at a high level at the end of the frame reception, the Parity error flag UDx.PERR is set.

28.3.1.2.3 Data reception

The reception of the upstream frame is started with a falling edge (1-to-0 transition) on the SI line. When the Start bit is detected, serial reception is enabled and the receive circuit begins to sample the incoming serial data and to buffer it in the receive buffer.

After the second Stop bit has been detected, the content of the receive buffer is transferred to one of four upstream data registers UDx. The receive circuit then waits for the next Start bit (1-to-0 transition) at the SI line. When the content of the receive buffer has been transferred to UDx, the Valid bit UDx.V is set by hardware and a receive interrupt can be generated, independently of whether PERR occurred or not.

The data reception mechanism does not involve destructive read mechanisms. This means, reading the UDx registers does not change any bits automatically.

***Note**: The SI input line is the filtered non-inverted (OCR.ILP=0) or inverted (OCR.ILP=1) SDI input signal. The SI input signal selection is described in related information).*

Frame reception with address field

Frame reception for a 16-bit data frame is selected by USR.UFT=1. When the content of the receive buffer has been received completely, it is transferred to one of the four UDx registers. The two most significant address bits A[3:2] of the received 4-bit address field select the number x of register UDx in which the received frame content is stored.

Register UDx is loaded with:

- The two least significant address bits A0 and A1 (UDx.LABF)
- The 8-bit data (UDx.DATA)
- The received Parity bit (UDx.P)
- The calculated Parity bit (UDx.IPF)
- The parity checking result (UDx.PERR)

Finally, the Valid bit UDx.V is set to indicate that the UDx register contains valid data.

The current state of the frame reception is indicated by the content of an upstream counter that is readable through bit-field USR.UC. The upstream counter is a 5-bit counter that counts the upstream frame bits during reception. As shown in the following figure, the upstream counter is loaded with 10000_B at the detection of a Start bit. It counts down and is again at 00000_B when the second Stop bit has been detected and the frame reception is finished.

The state of the serial input data line SI is sampled in the middle of a bit cell and shifted into the receive buffer at the end of the bit cell. The frequency of the shift clock f_{BR} depends on the selected baud rate.

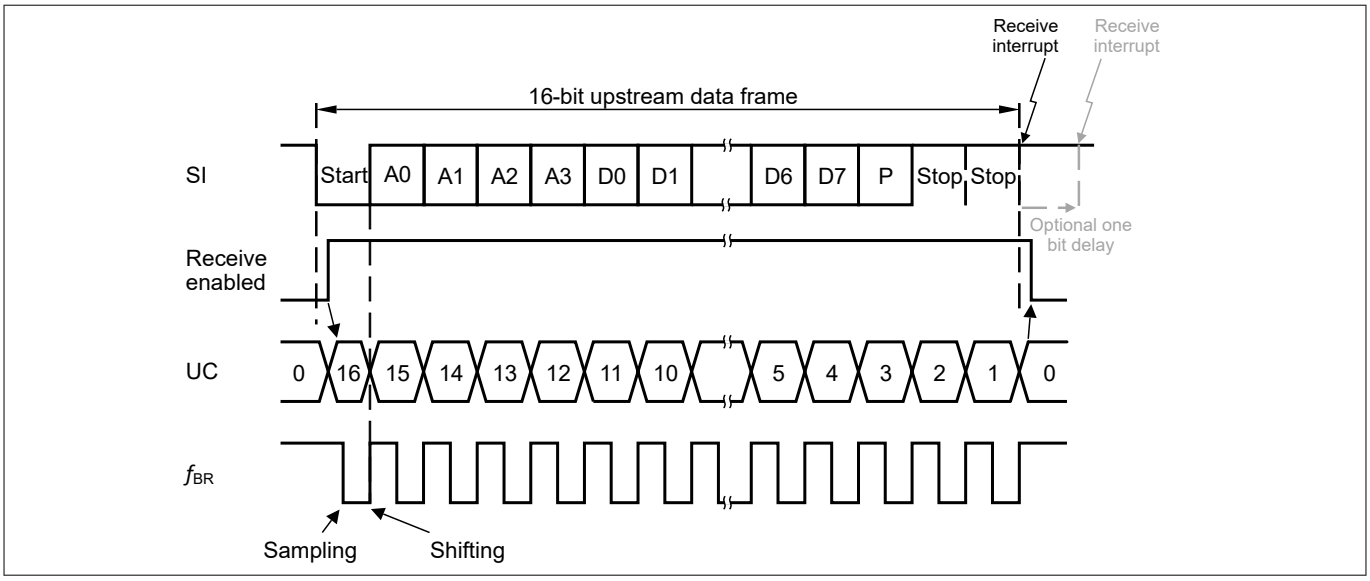


Figure 587 16-bit upstream reception

Data reception without address field

Frame reception for a 12-bit data frame is selected by USR.UFT=0.

The reception scheme is comparable with that of the 16-bit data frame reception but there are a few differences:

- The upstream counter is initially loaded with 01100_B
- The received frame content is always stored in register UD0
- Bit-field UD0.LABF is always loaded with 00_B when the frame is stored

Related information

Upstream channel baud rate on page 5100

Upstream channel input control on page 5107

28.3.1.2.4 Upstream channel baud rate

The baud rate of the upstream channel is derived from the MSC module clock f_{MSC}.

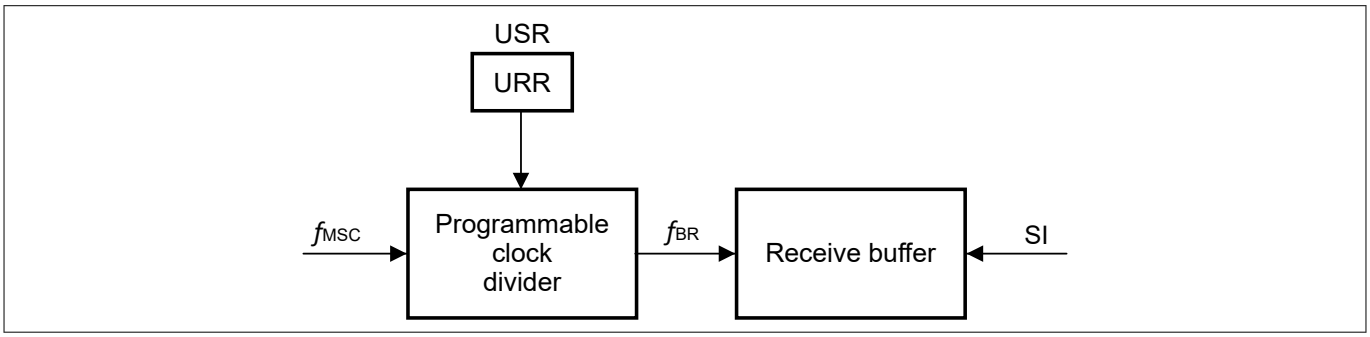


Figure 588 Upstream channel clock circuitry

The serial data input SI is evaluated with the baud rate clock f_{BR} in the middle of each bit element and latched in case of a data bit.

The baud rate clock f_{BR} is derived from f_{MSC} by a programmable clock divider. The frequency of f_{BR} determines the width of a received bit element and therefore the baud rate for the received data.

The content of bit-field USR.URR selects the baud rate according to the table below. The resulting baud rate formula is:

$$\text{Baud rate}_{\text{MSC Upstream Channel}} = \frac{f_{\text{MSC}}}{\text{DF}} \tag{31}$$

Table 1212 Upstream channel divide factor (DF) selection and baud rate

USR.URR	Divide factor (DF)	Baud rate
000_B	Reception disabled	–
001_B	4	$\frac{f_{\text{MSC}}}{4}$
010_B	8	$\frac{f_{\text{MSC}}}{8}$
011_B	16	$\frac{f_{\text{MSC}}}{16}$
100_B	32	$\frac{f_{\text{MSC}}}{32}$
101_B	64	$\frac{f_{\text{MSC}}}{64}$
110_B	128	$\frac{f_{\text{MSC}}}{128}$
111_B	256	$\frac{f_{\text{MSC}}}{256}$

***Note**: When USR.URR=000_B the upstream channel is disabled and data reception is not possible.*

The content of bit-field USR.URR determines the operation of an internal sampling reload counter that is clocked with f_{MSC}. The following figure shows the operation of the sampling counter at the beginning of an upstream frame with a divide factor DF of 8 (USR.URR=010_B is equal to DF=8) which means eight sampling clocks per each frame bit cell.

When the upstream channel is in idle state, it waits for a falling edge (1-to-0 transition) at SI. Therefore the sample counter starts counting up and is reset when the selected divide factor DF as shown in the table above is reached. In the middle of the sampling counter's count range, the logic state at SI is evaluated and, in case of a data bit, latched in the receive buffer's shift register. With the reload of the sampling counter, the shift register is shifted by one bit position.

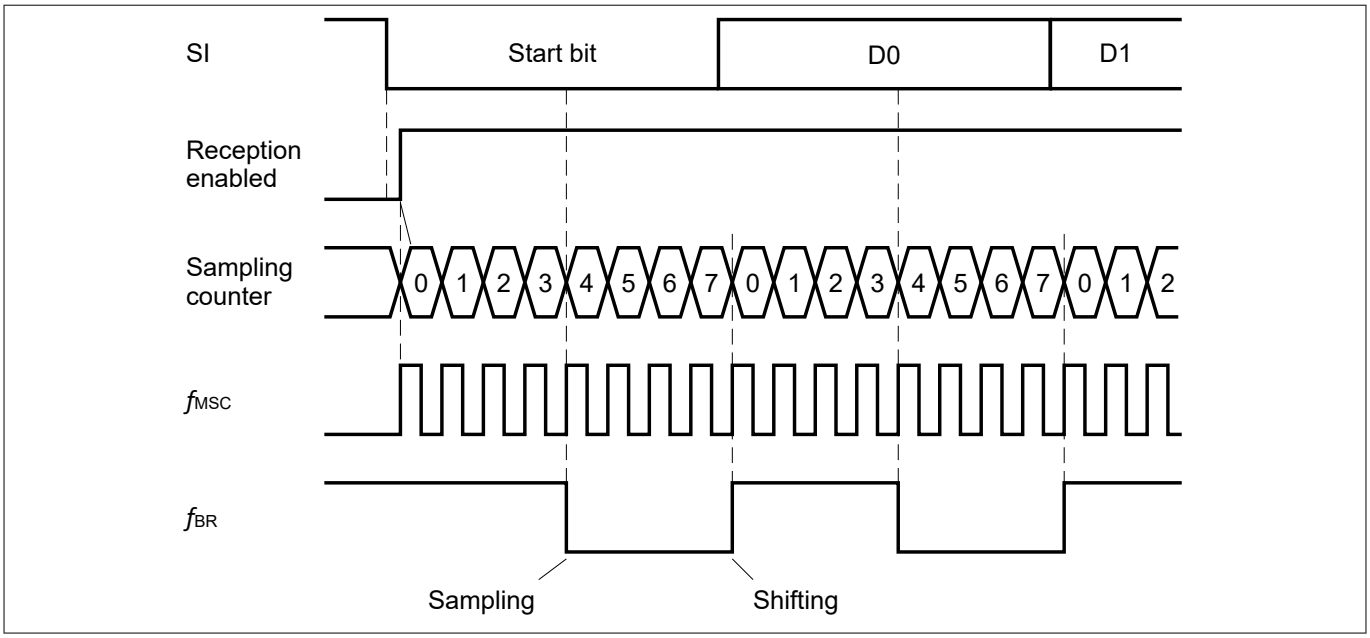


Figure 589 Upstream channel sampling with URR=010$_B$

28.3.1.2.5 Spike filter

The upstream channel input line SDI is sampled using a built-in spike filter with synchronization stage, both clocked with f_{MSC}. The spike filter is a chain of flip-flops with a majority decision logic (2 out of 3). A sampled value that is found at least twice in three samples is taken as data input value for SI.

28.3.1.2.6 Upstream timer

The upstream timer monitors the time interval between two subsequent upstream frames. The timer reloads its starting values from the register USCE and starts counting downwards. If the time interval between two frames is longer than allowed, the counter reaches the value of zero, raises an upstream timer interrupt and stops counting.

The counter reloads and starts counting on four events:

- Enabling of the MSC module after reset through CLC.DISR
- A module reset
- Writing the bit-fields USCE.USTOPRE or USCE.USTOVAL by software
- Detecting the falling edge of the start bit of a frame

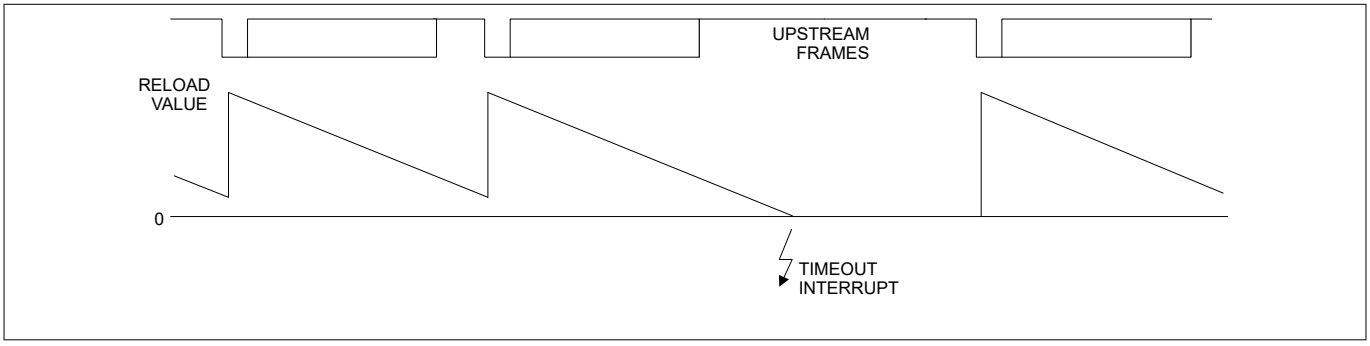


Figure 590 Upstream timer operation

The upstream watchdog timer counts bit times. The time window in which the next upstream frame must occur can be calculated according to the formula:

$$\text{Time-out} = \text{BitTime} \times 2^{(\text{USTOPRE}+1)} \times (\text{USTOVAL}+1) \quad (32)$$

28.3.1.3 CX (command extension) mode

Command Extension mode overrides the following standard features of the MSC protocol:

- Command frames:
  - Instead of 32 bits, command frames contain up to 2×32 bits and two select bits set to 1, one before each 32-bit block. One command frame now contains two commands, both with same length
  - The bit-field DSC.NBC is reused as it is and defines the length of both commands
  - The command chip select ENC is active during the whole active phase of the frame
  - The passive phase extension is extended to cover 2 to 127-bit times, see bit-field DSTE.PPCE and DSTE.PPCEM. This is valid only in ABRA active mode. When ABRA bypassed, the command passive phase is constant and fixed to two. The PPCEM bit is needed and extends the command passive phase only in the CX mode (CX = 1), where a command can be up to 64-bit long. If CX = 0, PPCEM bit is ignored and treated as always 0
- Data frames:
  - Instead of activating two enable signals ENL and ENH, data frames activate only one enable signal, ENL
  - It is the responsibility of the user to mandatory activate the select bits in the data frames with DSC.ENSELL and ENSELH in order to ensure the same frame structure of the command and the data frames
  - The existing data frame length bit-fields are reused as they are: DSC.NDBL, DSC.NDBH, DSCE.NDBLE and DSCE.NDBHE. It is the responsibility to the user to program them symmetrically if required and program the DSCE.EXEN, for example to set the same data frame length as the command frame length, which is the standard use case
  - The passive phase of the data frames is reused as it is
- Baud rate: It is possible to configure the MSC module to drive with maximal speed of 80 MBaud. The divider DSTE.NDD divides the module kernel frequency. When NDD = 0, the module runs with 100 MHz. When NDD = 2, the module runs with 33 MHz

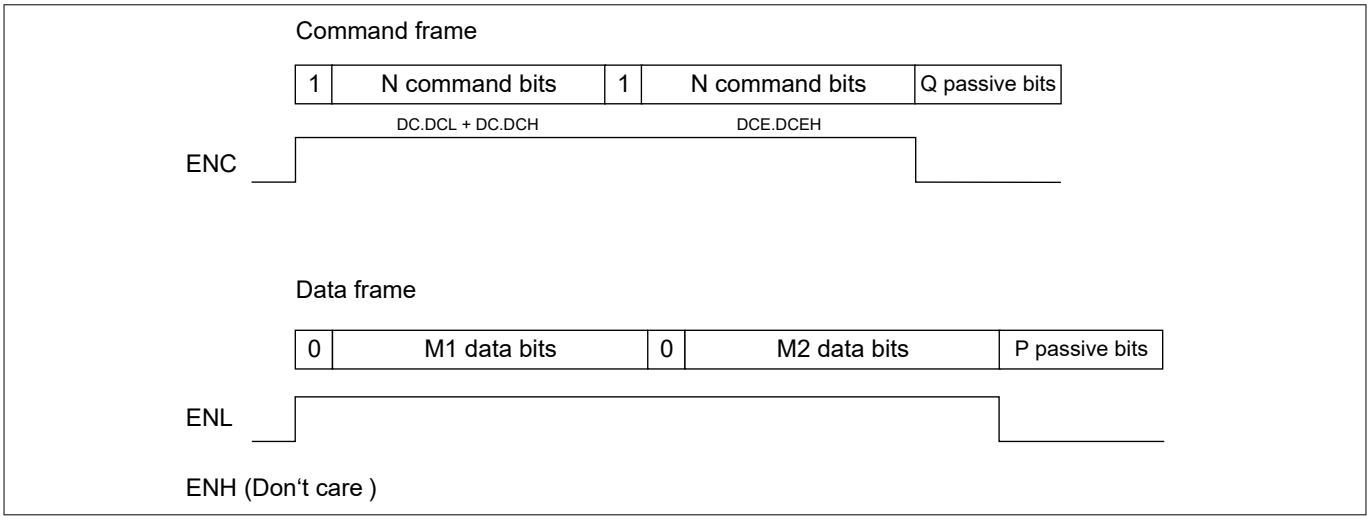


Figure 591 MSC frames in CX mode

This is the list of the additional bit-fields and registers correlated to CX mode.

- DSTE.CX bit, activating the CX mode

- DSTE.FM bit, activating the fast mode (up to 80 MBaud)
- DSTE.UL1 protection bit, for unlocking the CX and FM bits for one clock cycle
- DSTE.PPCE extended by one bit, in order to lengthen the passive phase of the command frame
- DCM, Downstream Command Mirror Register, providing convenient way to send more than 32-bit long commands by using DMA with memory address wrap around
- DCE, Downstream Command Extension Register, defining the second half of the command frame. In CX mode, always write first the DCE register and then DC or DCM, because write to the DC or DCM triggers the command transmission

28.3.1.4 I/O control

The types of I/O control logic for the MSC module I/O lines are shown in the following figure:

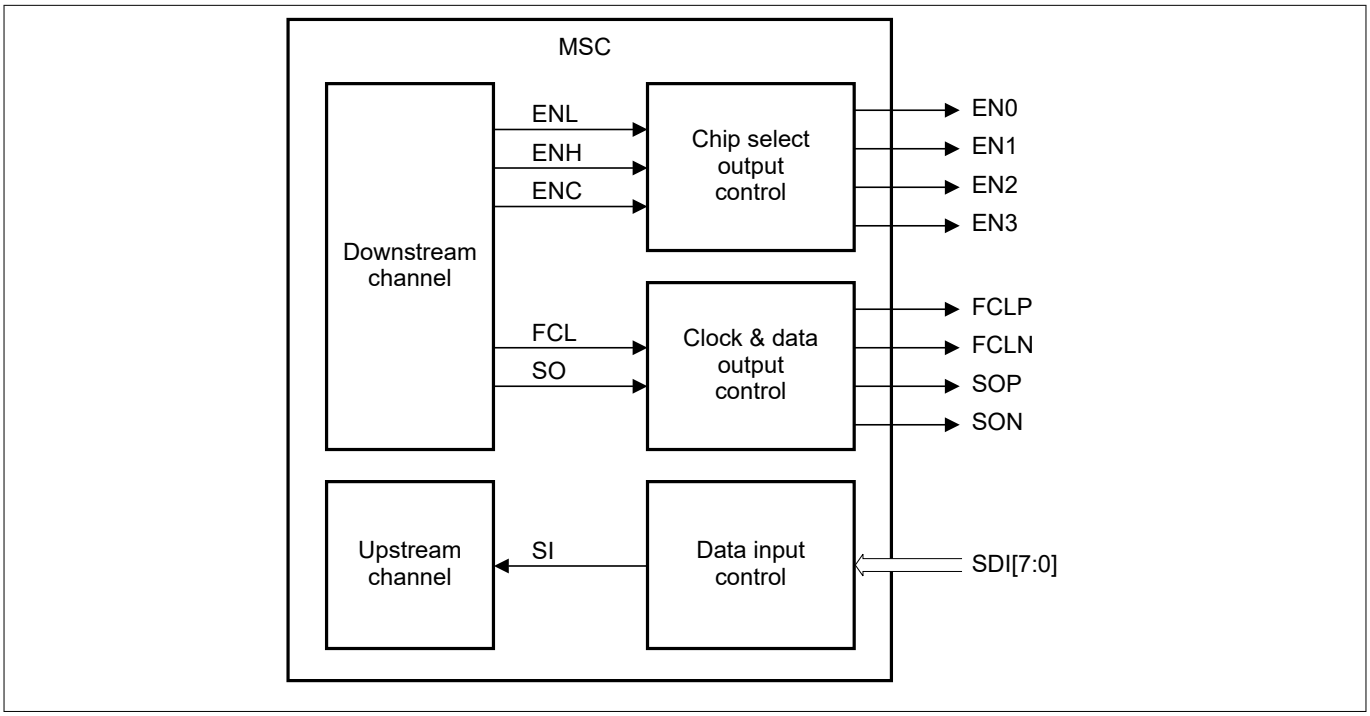


Figure 592 I/O control

The downstream channel generates five output signals that control eight MSC module outputs, split into four chip select outputs, two clock outputs and two serial data outputs.

The upstream channel has one input signal.

The MSC module I/O signals are controlled by bit-fields located in the Output Control Register (OCR).

28.3.1.4.1 Downstream channel output control

As described in the Frame and formats and definitions section, the active phases during downstream channel operation are indicated by three enable signals:

- ENL indicates the SRL active phase of a data frame
- ENH indicates the SRH active phase of a data frame
- ENC indicates the active phase of a command frame

The chip select output control logic of the MSC uses a signal compressing scheme (similar to the scheme described in the Interrupt request compressor section) that allows each of the three enable signals to be directed through a 2-bit selector to one of the four chip enable outputs EN[3:0]. This also makes it possible to connect more than one internal enable signal (ENL, ENH, ENC) to one chip enable output ENx. Three bit-fields

in the register OCR (CSL, CSH and CSC) determine which chip enable output becomes active on a valid internal enable signal.

In the MSC, enable signals are high-level active signals. If required in a specific application, all chip enable outputs ENx can be assigned for low-level active polarity by setting bit OCR.CSLP.

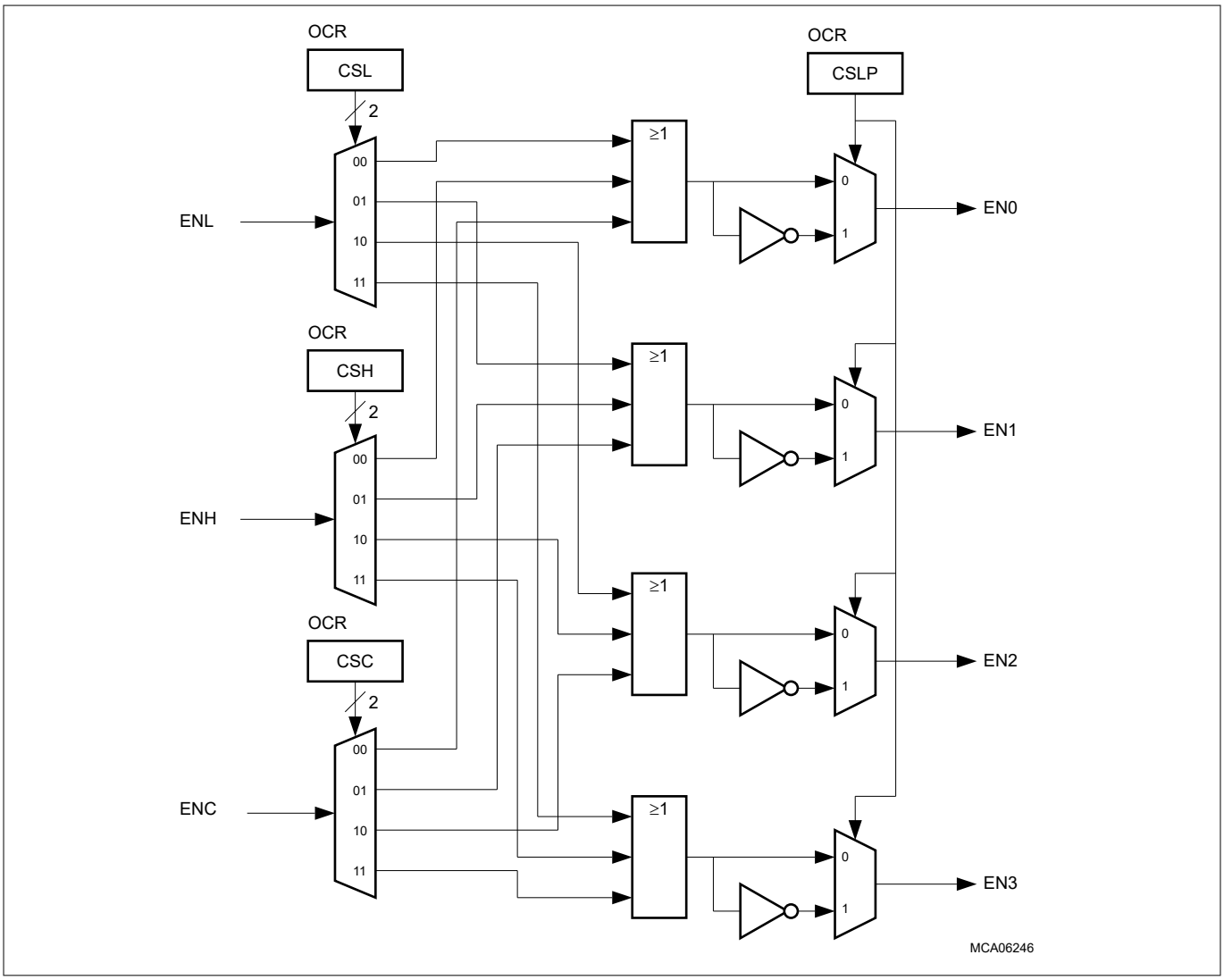


Figure 593 Downstream channel: Chip enable output control

At the MSC downstream channel, the internal serial clock output FCL and data output line SO are available outside the MSC module as two signal pairs with inverted signal polarity, FCLP/FCLN and SOP/SON. Both, clock and data outputs, are generated from the module internal signals FCL and SO according to the following figure.

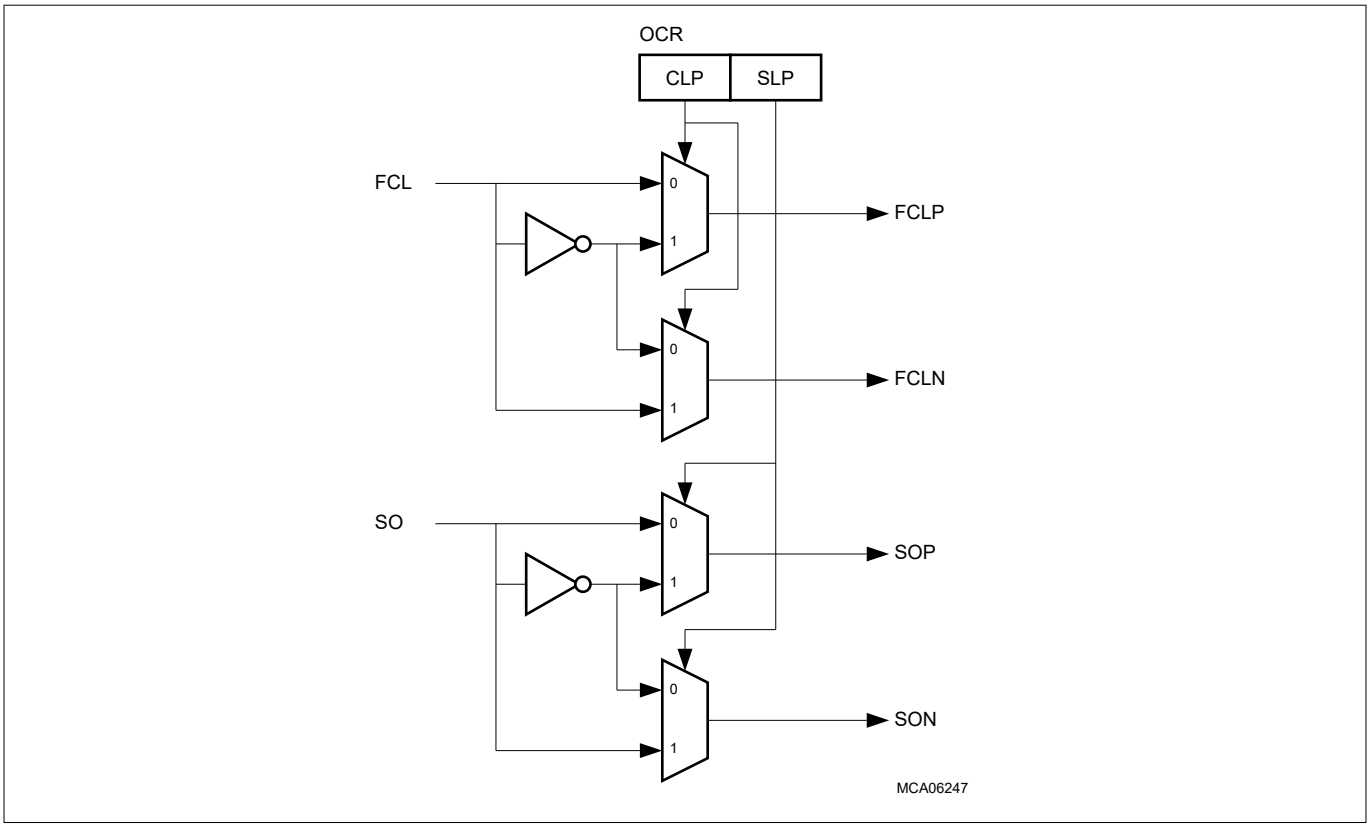


Figure 594 Downstream channel: Clock and data output control

With OCR.CLP=0, FCLP has identical and FCLN has inverted polarity compared to FCL. Setting OCR.CLP, exchanges the signal polarities of FCLP and FCLN. An equivalent control capability is available for the SOP and SON data outputs (controlled by OCR.SLP).

One additional control capability not shown in the previous figure is available for the FCL signal:

- OCR.CLKCTRL=1
  - The FCL clock signal will always be generated, independently of whether a downstream frame is currently transmitted or not
- OCR.CLKCTRL=0
  - FCL becomes active only during the active phases of data or command frames (not during passive time frames)

Enable Signal Time Adjustment

The enable signal can be adjusted in steps of ½ FCL clock time period.

Each enable signal has the possibility to be configured 0, 1 or 2 steps of ½ clock time-period adjustment at assertion and deassertion independently, as shown in the figure below. This functionality can be programmed in the OECR register.

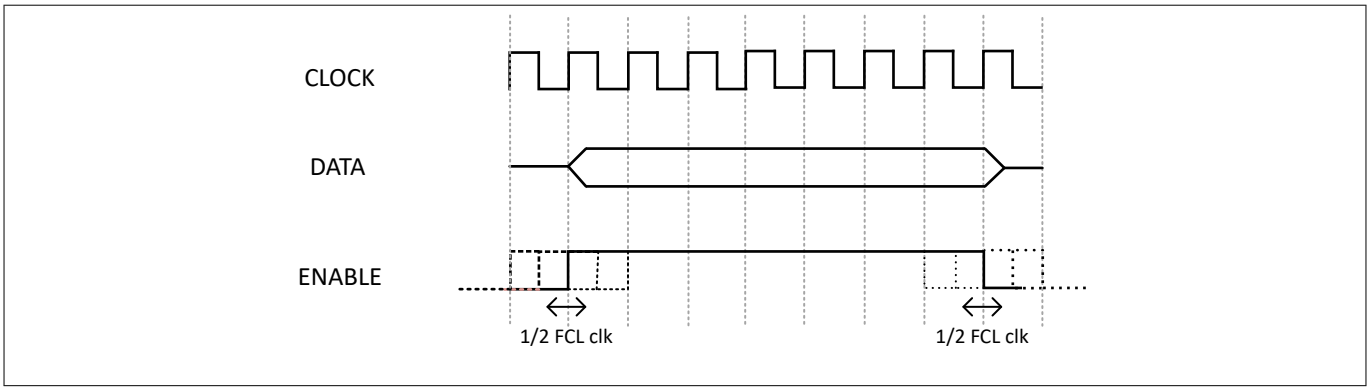


Figure 595 Halb bit enable signal adjustment

When using this option the passive phase needs to be taken into account. The passive phase is min. $2 \times t_{FCL}$. If a time adjustment is configured for the enable signal, the passive phase has to be adjusted accordingly (minimum $2 \times t_{FCL}$). The length of the passive phase can be programed accordingly in DSC.PPD for data frame and in DSTE.PPCE/DSTE. PPCEM for command frame.

Related information

Frame formats and definitions on page 5082
Interrupt request compressor on page 5130

28.3.1.4.2 Upstream channel input control

As shown in the figure below, the MSC upstream channel can be connected to up to eight SDI[7:0] serial inputs. Bit-field OCR.SDISEL selects one out of these input lines (input signal SDI).

- OCR.ILP = 0
  - SDI is directly connected to the serial receive buffer input SI
- OCR.ILP = 1
  - SDI is connected to input SI through an inverter

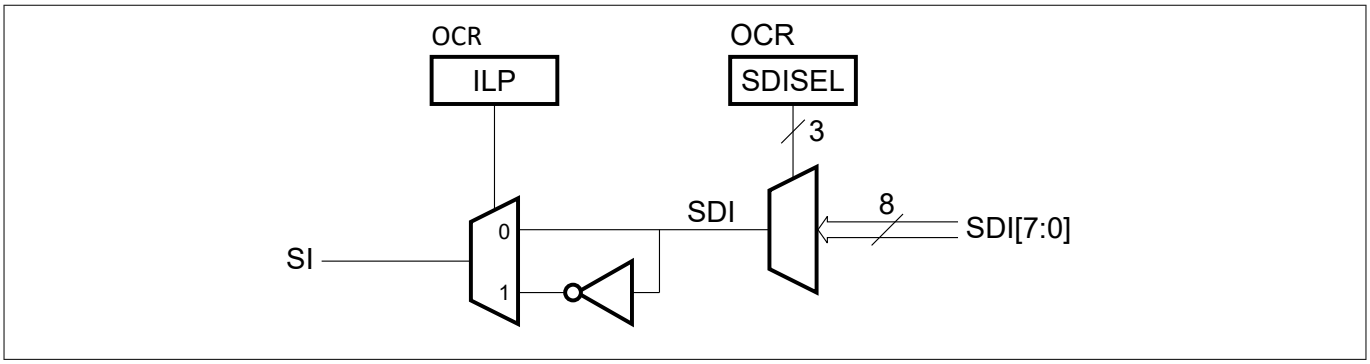


Figure 596 Upstream channel serial data input control

28.3.1.5 Slow channel

This section describes the Slow Channel feature for the MSC downstream transmission.

The main use case of the Slow Channel feature is the transmission of command frames without interrupting the continuous data frame traffic (for example equidistant update of data).

Slow Channel mode overrides the following standard features of the MSC protocol:

- No Selection bits are used

- The enable signals ENH and ENL are used as are used in legacy
- The enable signal ENC is not used
- There is no possibility to send command frames as in legacy, they can be only send embedded through the slow channel
- It is not compatible with CX (Command Extension) mode
- The registers DD and DDE are reused and contain the data to be transmitted in the data frames
- The existing data frame length bit fields are reused as they are: DSC.NDBL, DSC.NDBH, DSCE.NDBLE and DSCE.NDBHE. It is the user responsibility to program them symmetrically if required and program the DSCE.EXEN
- The passive phase of the data frames is reused as is
- Interrupts (see interrupt section)

This is the list of the additional bit-fields and registers correlated to the slow channel mode:

- The bit-field SCC.SC enables the Slow Channel mode
- The bit-fields SCC.SCLX and SCC.SCLY define the bit length of the slow channels
- The bit-fields SCC.CFLX and SCCCFLY define the length of the command frames
- The bit-fields SCC.PCX and SCC.PCY enable and select the parity to be used for the command frame transmission
- The registers DCSX and DCSY are used for defining the command frame data
- The bit-fields SCC.CFSTOPX and SCC.CFSTOPY define the Stop bits configuration for the command frame
- The bit-fields ISR.CSXI and ISR.CSYI for the interrupt status flag for the corresponding command frame slave X or slave Y and the related bit-fields in registers ICR and ISC

***Note**: The behavior is undefined when the configuration is changed in between an ongoing transaction*

28.3.1.5.1 Overview

The main difference to the standard downstream transmission is that at every MSC downstream frame some bits are reserved for including a slow channel for sending the command frame in small parts.

In the block diagram below the behavior of the downstream channel when enabling the Slow Channel feature is shown. The Slow Channel feature is set in bit-field SCC.SC.

One slow channel is supported per slave without CPU interaction in Data Repetition and Triggered mode by using this feature. A maximum of 2 slaves are supported.

The data source for the command frame are the Downstream Command Register DCSX for Slave X and the Downstream Command Register DCSY for Slave Y. The length of the command frame is configurable between 1 and 32 bits in bit-fields SCC.CFLX and SCC.CFLY.

The command frame is split into small parts and each of these parts is transferred as slow channel payload in the data frame. The bit width of the slow channel (number of slow channel bits) per data frame is configurable independently for each slave in the bit-fields SCC.SCLX and SCC.SCLY. The slow channel width possible configuration is: 0, 1, 2, 4, 8, 16 or 32-bits. The slow channel is always transmitted in the payload before the data frame of each slave.

The transmitted payload through the MSC interface is built as follows: The slow channel data is transmitted with the configured bit length and the data frame is sent directly after the slow channel as indicated in the picture below.

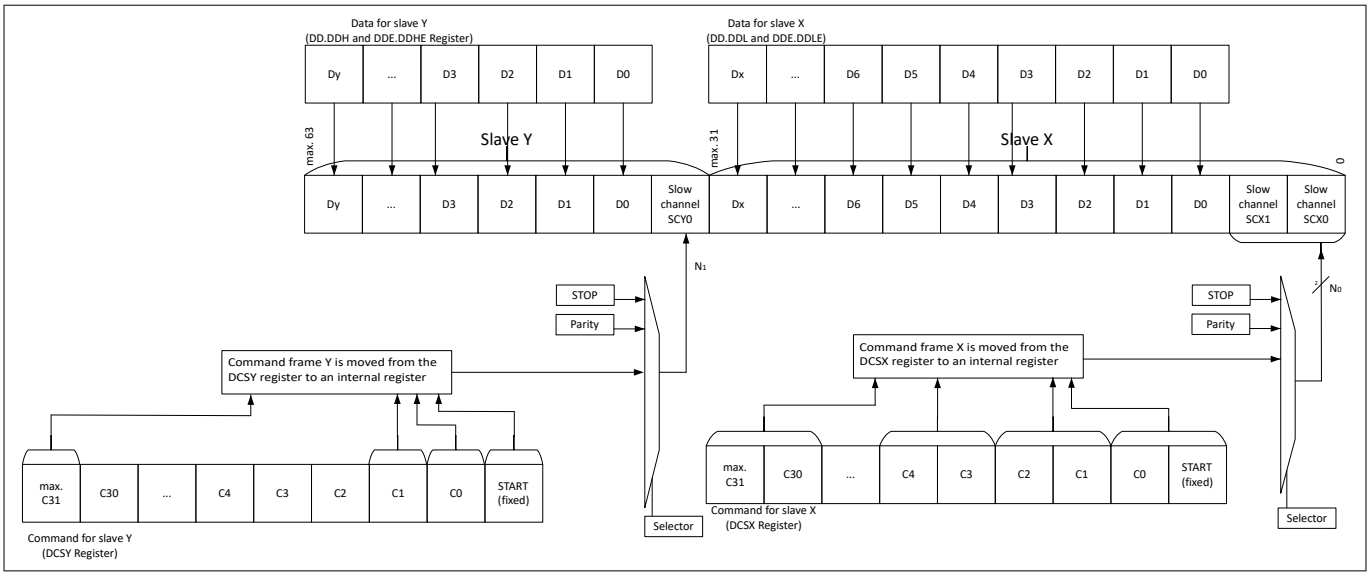


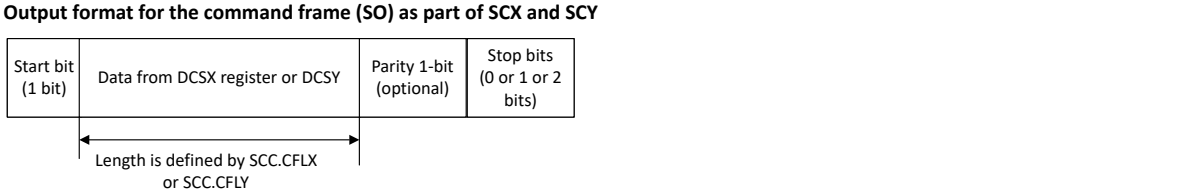
Figure 597 MSC transmitted payload with slow channel

The command frame transmission for both slaves is completely independent.

If there is no command data to be transmitted the slow channel will transmit Stop bits (logical 1).

If command data is being transmitted though the slow channel and a next command is pending, the next command frame is transmitted directly after the current one is finished.

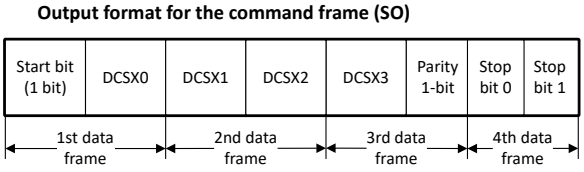
Command frame data is encoded in a UART fashion frame (Start, Data, Parity optional and configurable 1 or 2 Stop bits). The Start bit is fixed and coded with a logical 0. The parity for the command frame is optional and the type of parity can be configured in the bit-field SCC.PCX and SCC.PCY. The number of Stop bits to be used can be programmed in SCC.CFSTOPX and SCC.CFSTOPY bit-fields. 0, 1 or 2 Stop bits can be programmed. More Stop bits can be programmed by software as part of the command frame (in registers DCSX and DCSY). The Stop bit is coded with a logical 1. The exact format of the output command frame is shown in the figure below along with an example.



Whether the entire command frame is sent out or if it is sent out in parts depends on the ratio of the value configured for SCC.SCLX or SCC.SCLY and the complete number of bits in the command frame (Start bit, Data, optional Parity bit and optional Stop bits)

Figure 598 Output format for command frame

Example configuration
SCC.SCLX = 010 => 2 bits
SCC.CFLX = 4 bits
SCC.PCX = 01 => Even parity ON
SCC.CFSTOPX = 10 => 2 Stop bits



Write into the DCSX register triggers the next command frame to be sent with new data. Otherwise default bits are sent out

Figure 599 Example command frame

***Note**: No Selection bit is used in this mode. While using slow channel in legacy mode the Selection bit has to be disabled (DSC.ENSELL=0 and DSC.ENSELH=0).*

***Note**: The number of bits of the command frame to be transmitted through the slow channel are the width programmed in SCC.CFLX and SCC.CFLY.*

***Note**: Without any data frame to be transmitted, no command bits will be sent out in Slow Channel mode.*

28.3.1.5.2 Downstream channel with slow channel feature

The downstream channel performs a high-speed synchronous serial transmission of data to external devices. Its 128-bit shift register is divided into four 32-bit parts, SRH, SCY, SRL and SCX.

Each 32-bits of SRL and SRH can be selected to be delivered by the Downstream Data Register DD and DDE or by four 16-bit wide input signal buses, ALTINL, ALTINH.ALTINLE and ALTINHE if the Extended mode is activated (DSCE.EXEN=1).

The figure below is a diagram of the MSC downstream channel with the Slow Channel feature activated.

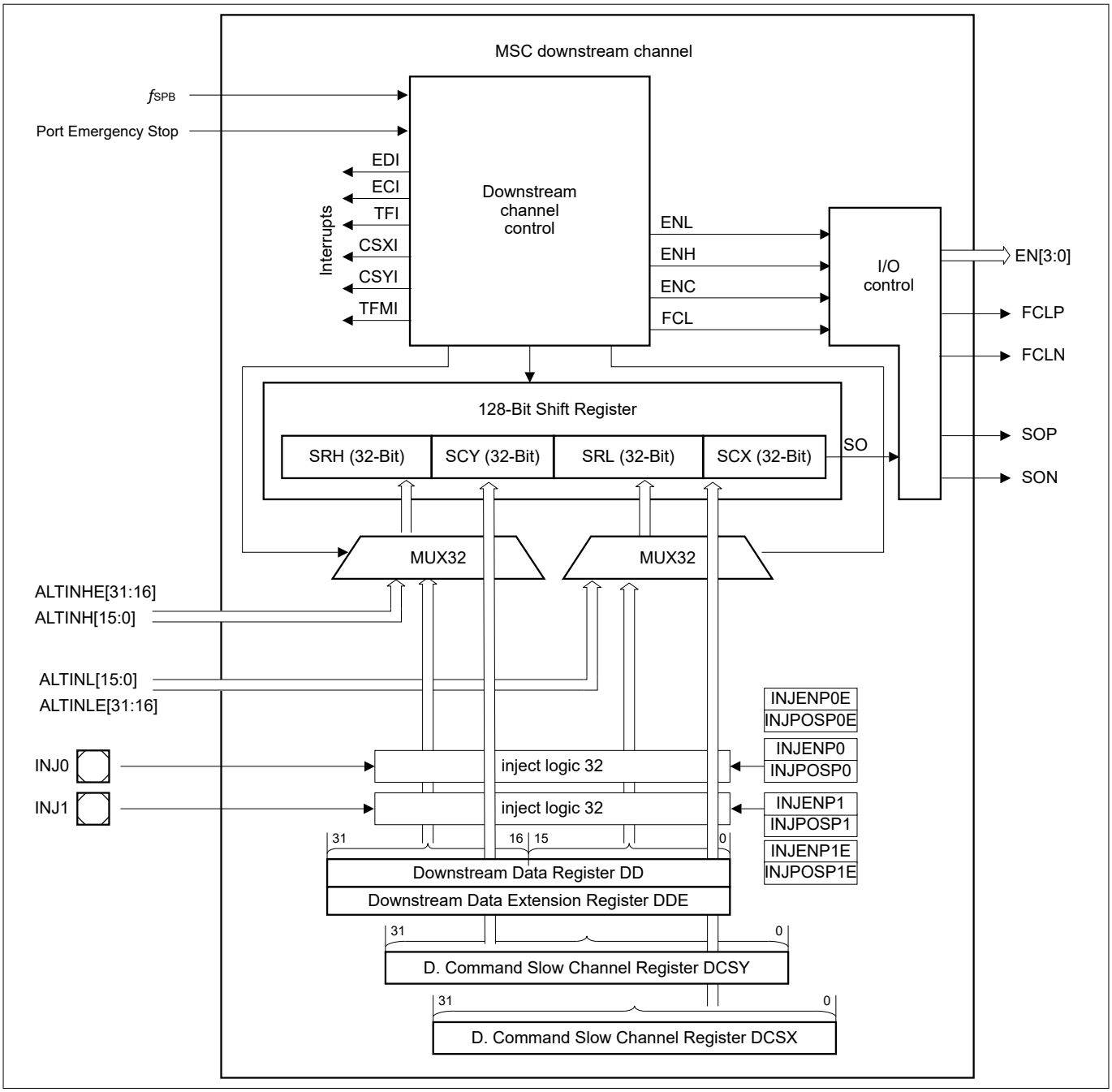


Figure 600 Downstream channel with slow channel feature

The enable signals ENL and ENH indicate the low and high phases of the serial transmission in relation to the serial FCL. The enable signal ENC is not used while Slow Channel feature is enabled.

The Port Emergency Stop input is used to indicate an emergency stop condition of a power device. In the case of an emergency, shift register bits can be loaded bit-wise from the downstream data register instead from the ALTINL, ALTINH, ALTINLE and ALTINHE buses. This behavior does not change when the slow channel is activated since these bits can be mapped only to data frame bits and the slow channel bits are extra bits in the MSC downstream payload.

28.3.1.5.3 Frame format using slow channel for command frame transmission

A downstream frame is composed of:

- An active phase
  - During the active phase data and slow channel transmission take place
- A passive phase
  - During the passive phase no data is transmitted at SO

The active phase is split into four parts:

- SRL active phase, which is further two parts:
  - The content of the shift register slow channel X and
  - Low part SRL is transmitted
- SRH active phase , which has further two parts:
  - The content of the shift register slow channel Y and
  - High part SRH is transmitted

The least significant bits of SRL, SRH, slow channel X and slow channel Y are sent out first.

The figure below shows the layout and definitions of the downstream frame using slow channel for the command frame transmission.

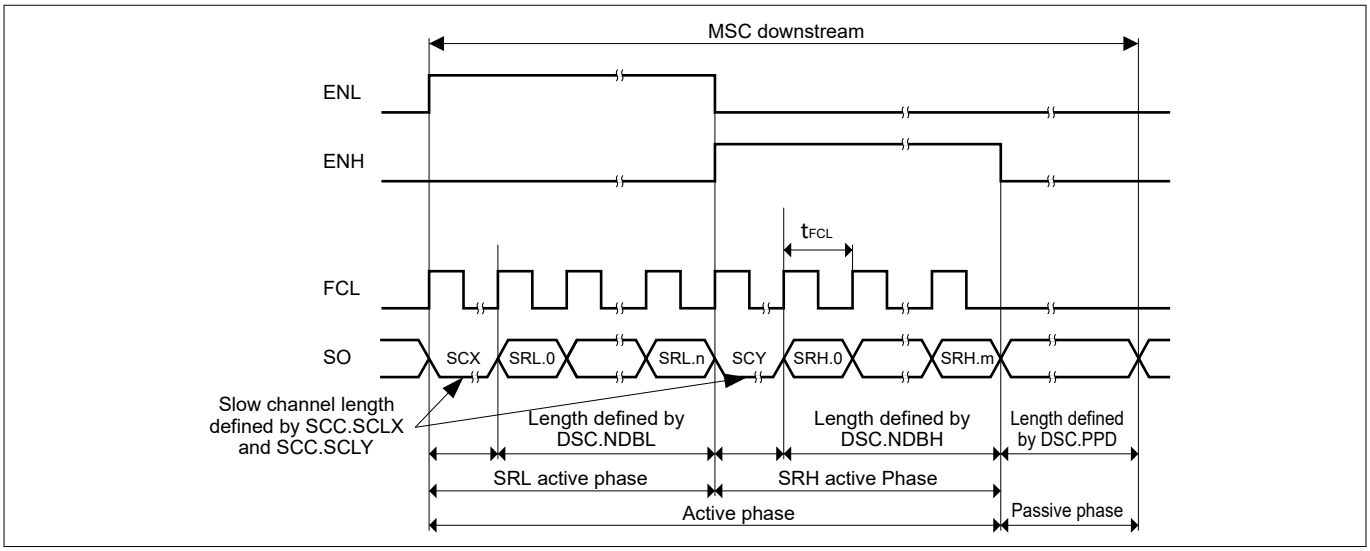


Figure 601 Downstream channel frame using slow channel

28.3.1.5.4 Interrupts in slow channel mode

The interrupts available in slow channel:

- Command frame Interrupts for slow channels X and Y (special implementation only for slow channel mode)

28.3.1.5.5 Time reference point in slow channel, data repetition mode

A passive time frame has the length defined by the parameters according to the following equation. The passive time frame is only generated in Data Repetition mode as shown in the picture below. Running in slow channel mode, there is no possibility to shorten the passive time frames.

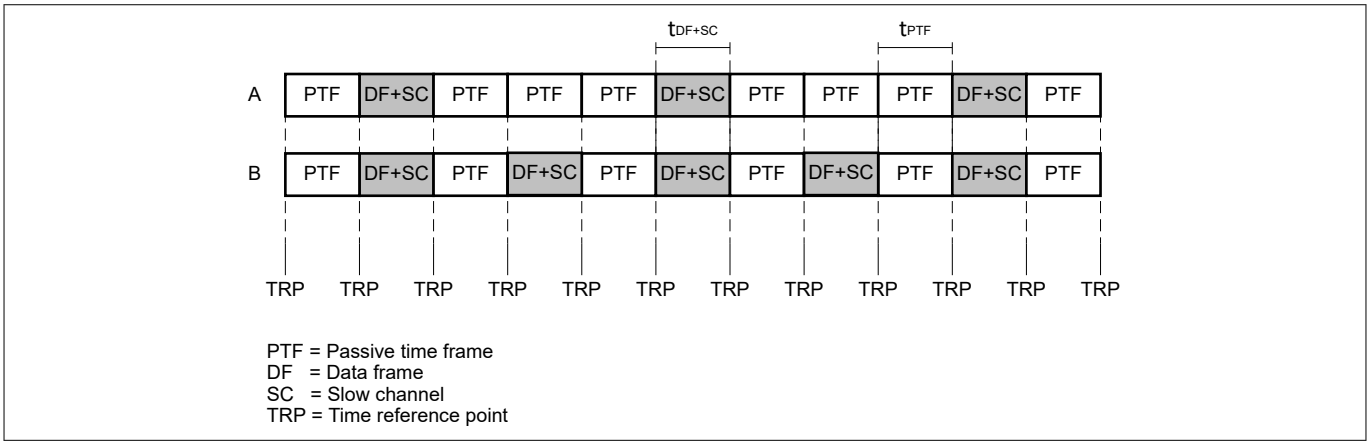


Figure 602 Passive time frame and time reference point using slow channel

The time reference point in slow channel mode is calculated with the following formula:

$$N_{CYCLES} = SCC.SCLX + NDBL + NDBLE \times EXEN \times 16 + SCC.SCLY + NDBH + NDBHE \times EXEN \times 16 + PPD + PPDE \times EXEN \times 32 \quad (33)$$

t_{PTF} calculation using slow channel

***Note**: In the formula above, PPD must be set to 2 when DSC.PPD<= 00010_B.*

28.3.1.5.6 Slow channel and ABRA block

While combining the ABRA block and the slow channel, the formulas explained in chapter Adjusting the passive phase of a frame can be used. ND needs to be calculated as follow:

$$N_D = SCC.SCLX + SCC.SCLY + DSC.NDBH + DSC.NDBL + DSCE.NDBHE + DSCE.NDBLE \quad (34)$$

N_D calculation

***Note**: While combining the slow channel and ABRA block, note that only 64 bit length downstream frames are considered to be used.*

28.3.1.6 Manchester code transmission mode

This section describes the Manchester Code transmission mode for the downstream channel. This transmission mode is only compatible with the Slow Channel feature, also when the slow channel is combined with ABRA block . No legacy downstream transmission is possible with Manchester code.

In order to reduce the pin count on the downstream channel all following signals are embedded in one:

- Enable signal
- Clock
- Data

This is the list of the additional bit-fields and registers correlated to Manchester Code transmission mode:

- The bit-field SCC.MCT enables the Manchester encoding transmission
- The bit-field ISR.TFMI flags for time frame Manchester code interrupt and corresponding bit-fields in ICR and ISC registers

The Manchester encoding transmission is enabled in bit-field SCC.MCT = 1.

The data and clock signals are sent through the MSC output line encoded in Manchester code, while the enable signals are represented by 2 idle bits without Manchester encoding. The level of these idle bits is used for addressing the slave. There can be either IDLE 0 or IDLE 1 between the Data frames.

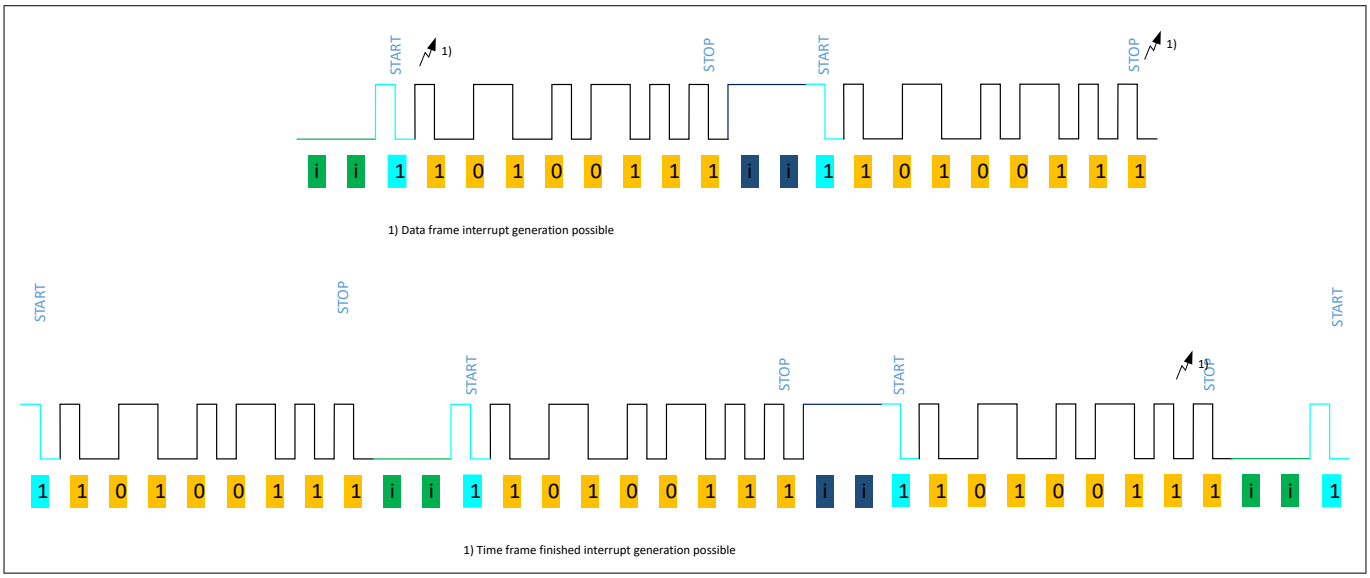


Figure 603 MSC frame Manchester encoded

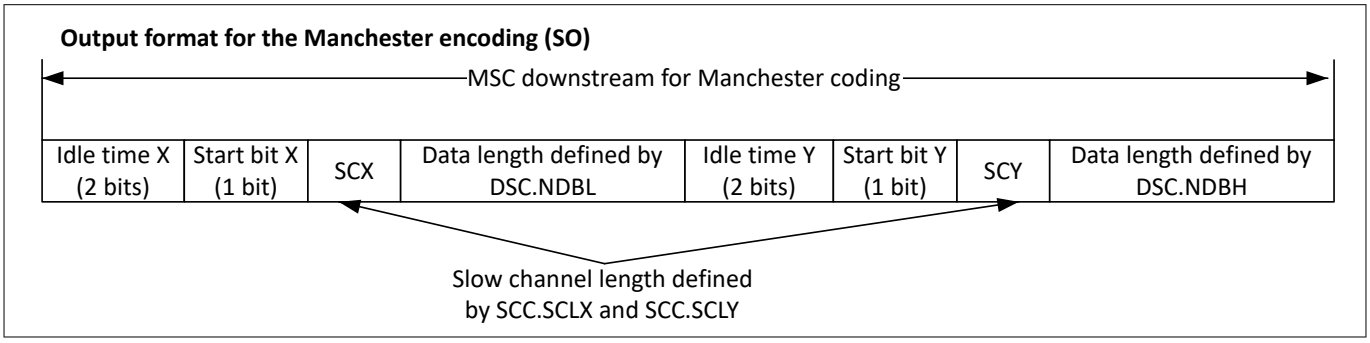


Figure 604 Output format for Manchester encoding (SO)

Both Manchester encoding are supported: IEEE and G.E. Thomas. This can be configured in bit-field OCR.SLP. Transmission in G.E. Thomas when OCR.SLP=0. To transmit in IEEE 802.3 the polarity of the SOP line has to be inverted, by setting OCR.SLP=1. Note that transmitting in IEEE encoding the slaves are swapped.

This transmission mode supports 2 slaves addressing. The slaves are addressed with the idle time:

- Idle time with level low, addresses slave X
- Idle time with level high, addresses slave Y

***Note**: This description and all figures in this document represent the G.E. Thomas Manchester encoding.*

28.3.1.6.1 Interrupts in Manchester code

The interrupts available in Manchester code are:

- Time frame finished interrupt in Manchester code mode (special for Manchester code - no passive phase available)

28.3.1.6.2 Time reference point in Manchester code - data repetition mode

A passive time frame has the length defined by the parameters according to the following equation in Manchester code. The passive time frame is only generated in Data Repetition mode (refer to figure "Passive time frame and time reference point using slow channel").

The time reference point in Manchester Code mode has to be calculated with the following formula:

$$N_{CYCLES} = 2(\text{idle time length}) + 1(\text{start bit}) + SCC.SCLX + NDBL + NDBLE \times EXEN \times 16 + 2(\text{idle time length}) + 1(\text{start bit}) + SCC.SCLY + NDBH + NDBHE \times EXEN \times 16 \quad (35)$$

t_{PTF} calculation in Manchester code

Related information

Time reference point in slow channel, data repetition mode on page 5113

28.3.2 Asynchronous baud rate adjustment (ABRA)

The ABRA block takes as input a serial MSC stream with one baud rate and outputs the same serial stream with another baud rate, asynchronous to the input one. The input signal bundle consists of serial clock, data and select signal. The output signal bundle consists of the same signals.

The output clock of an MSC without ABRA can only be set to 50% duty cycle, while with the ABRA block other duty cycles are also possible.

***Note**: The ABRA feature and Manchester encoding are not compatible with each other, therefore they can not be used together.*

28.3.2.1 Overview

The ABRA block is located between the MSC kernel and the output pins.

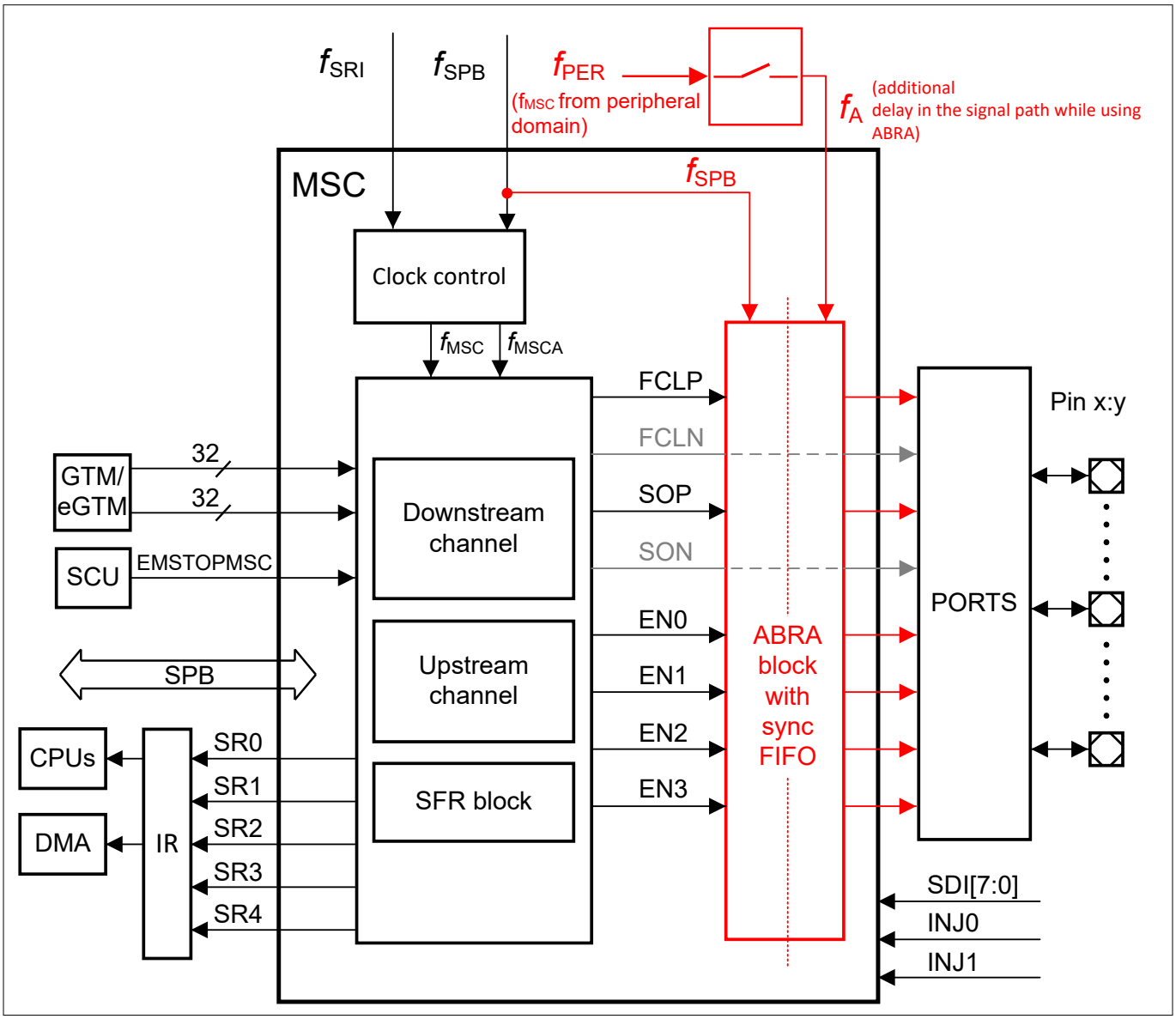


Figure 605 ABRA overview

The user has two possibilities: either to use the ABRA block or to bypass it. In case of using ABRA, additional delay in the signal path between the module and the pins of up to three f_A periods is to be taken into account. Additionally, some margin between the input frame length and the output frame length must be guaranteed by the customer, in order to avoid overflow of the ABRA block.

If the ABRA is not used, the behavior of the MSC module is identical with previous implementations. The user can also choose between using the 64-bit extension or not, which makes 4 combinations in total, as shown in the figure below.

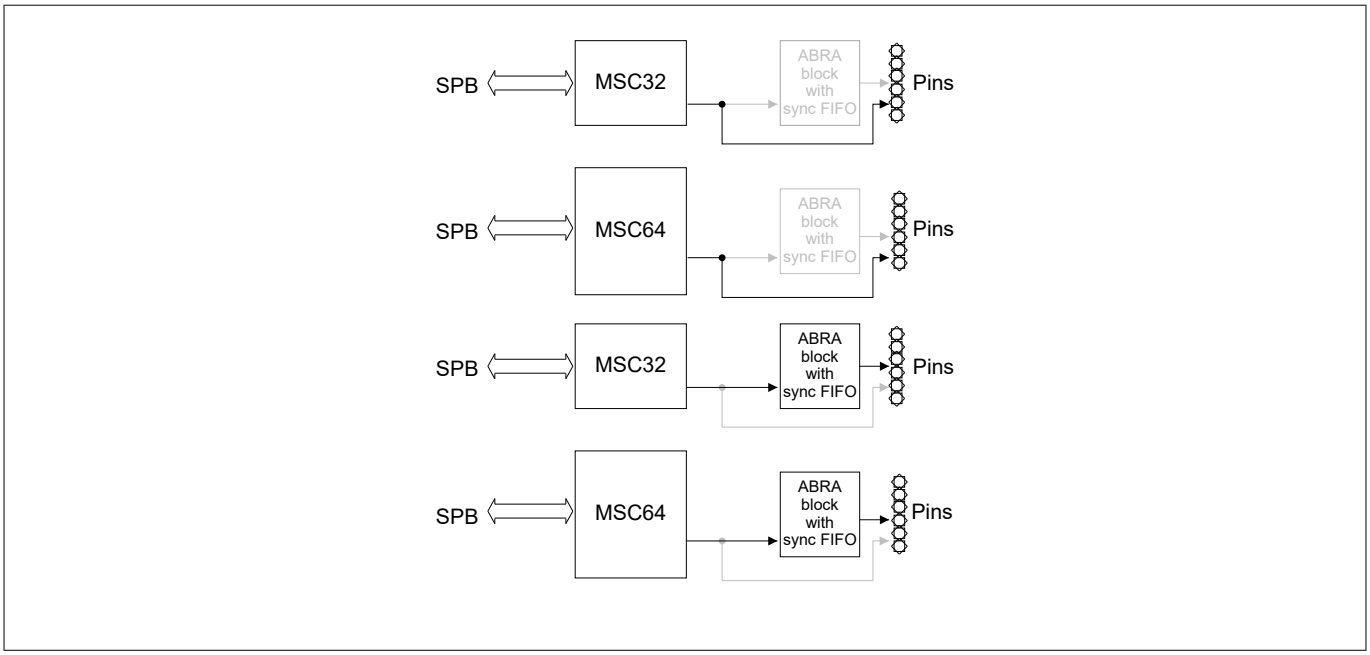


Figure 606 Use-case combinations

All configuration parameters related to the ABRA block are located in the ABC (Asynchronous Block Configuration) register.

28.3.2.2 Timing issues

The ABRA block introduces a delay in the signal path from the MSC module to the pins. However it does not significantly influence the timings between the signals themselves.

The output delays between the shift clock, data, and enable signals remain approximately the same as if the ABRA block is not used. These are covered in the timings of the MSC module as defined in the respective datasheet.

28.3.2.3 Adjusting the passive phase of a frame

The main purpose of the ABRA block is to transform a frame going in with a higher baud rate (for example 50 MBaud), to a frame going out with a lower baud rate (for example 40 MBaud). Therefore, the duration of the outgoing frame is longer then the duration of the incoming frame. In order not to overflow the ABRA block with too many incoming frames, there must be a pause between them, which is adjusted by configuring the passive phase of the frames. This pause plus the incoming frame duration must be longer or equal to the outgoing frame duration plus some margin.

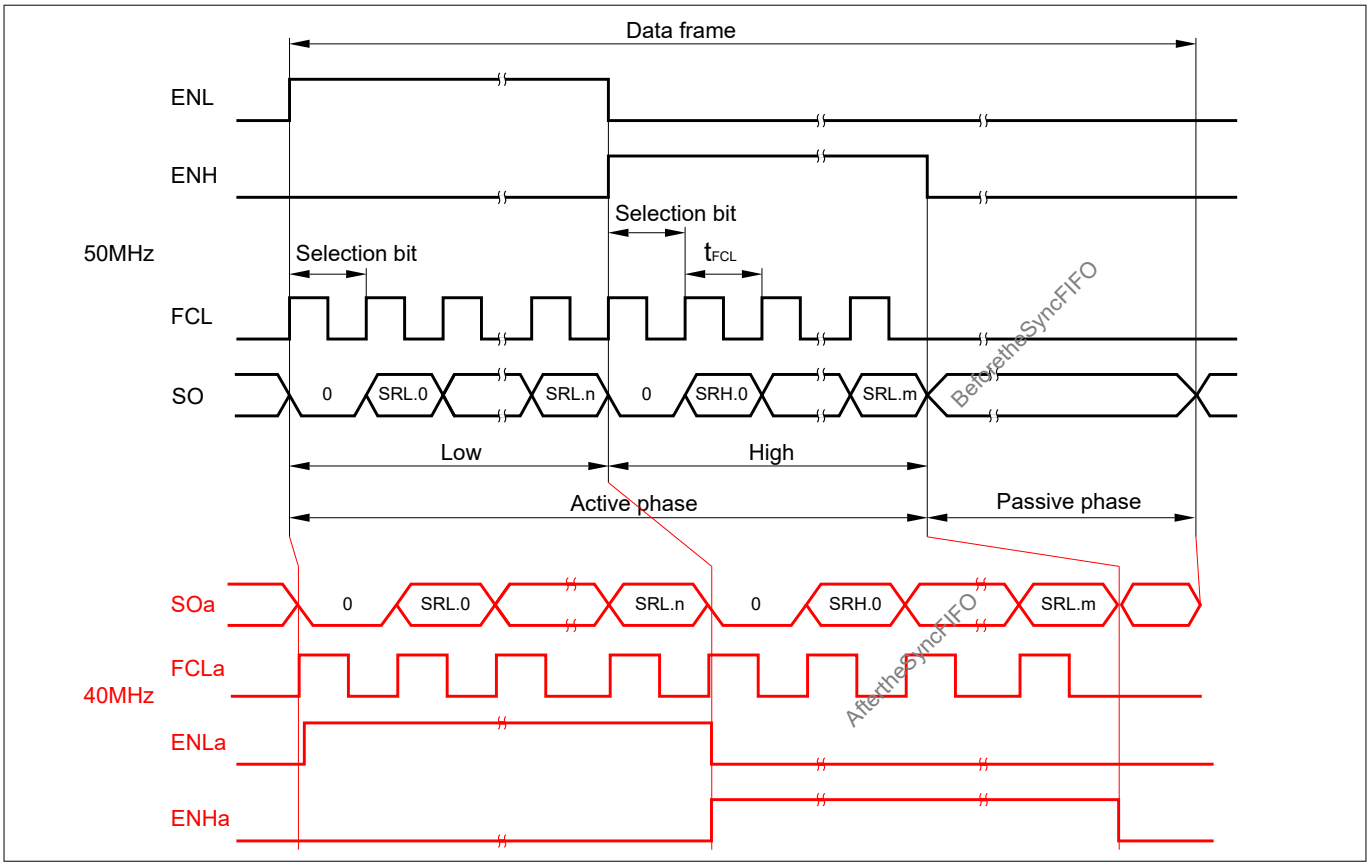


Figure 607 Passive phase adjustment

In order to allow for frame length adjustments:

- For command frames, the bit-field DSTE.PPCE defines a programmable passive phase in a range of 2 to 65 input frequency cycles and for data frames, the bit-field DSTE.PPDE extends the passive phase from 2 to 31 to 2 to 127-bit times

The length of the frame in the MSC domain (data and passive bits) must be longer than the length of the frame in the asynchronous domain, in order to prevent overload in case of back to back transmission:

$(N_D + N_P) \times T_{\mathrm{FCL}} > (N_D + N_{\mathrm{PA}}) \times T_{\mathrm{FCLA}}$

which results to the following inequality for setting the length of the passive phase in the MSC domain:

$N_P > \frac{(N_D + N_{\mathrm{PA}}) \times T_{\mathrm{FCLA}}}{T_{\mathrm{FCL}}} - N_D$ or equivalently $N_P > \frac{(N_D + N_{\mathrm{PA}}) \times f_{\mathrm{FCL}}}{f_{\mathrm{FCLA}}} - N_D$

for example if $f_{\mathrm{FCL}} = 50\ \mathrm{MBaud}$, $f_{\mathrm{FCLA}} = 40\ \mathrm{MBaud}$, $N_D = 32$ and $N_{\mathrm{PA}} = 3$, then $N_P > (32 + 3) \times 1.25 - 32$, $N_P > 12$

Where:

T_{FCL}: serial clock period in the MSC domain with activated ABRA: $T_{\mathrm{FCL}} = \frac{1}{\mathrm{BaudRate}_{\mathrm{MSCA}}}$,

$\mathrm{BaudRate}_{\mathrm{MSCA}} = \frac{f_{\mathrm{SPB}}}{(2 \times m)}$, see the Clock control when using the ABRA block section.

T_{FCLA}: serial clock period of the ABRA block $T_{\mathrm{FCLA}} = \frac{1}{\mathrm{BaudRate}_{\mathrm{ABRA}}}$,

$\mathrm{BaudRate}_{\mathrm{ABRA}} = \frac{f_A}{(\mathrm{ABC.NDA} + 1) \times (\mathrm{ABC.LOW} + 1 + \mathrm{ABC.HIGH} + 1)}$

N_{P}: number of the passive bits in the MSC domain (see DSTE.PPDE and DSC.PPD)

N_{PA}: number of the targeted passive bits in the asynchronous domain

N_D: number of data bits DSC.NDBH + NDBL + SELH + SELL + DSCE.NDBLE + NDBHE in both domains.

***Note**: If the passive phase settings in the MSC module are not correctly configured in such a way that two frames merge, that is, there is no passive phase on the pins at all, an overflow interrupt is triggered. If the passive phase is wrongly shorter, no error signal is generated.*

Input/output baud rate ratio

The ABRA output baud rate going to the pins must be lower than the input baud rate coming from the MSC module and higher than roughly half of the input baud rate (more precisely higher than 57% of the input baud rate in worst-case, due to the fact that in worst-case MSC data frame can contain 64 data bits plus additionally two select bits).

Related information

Clock control when using the ABRA block on page 5134

28.3.2.4 Jitter of the downstream frames

This section describes the jitter effects affecting the starting point of the downstream frames, depending on the setting of the serial clock - serial clock only during the active phase of a frame or continuous clock.

28.3.2.4.1 Jitter in active phase clock mode

The active phase clock mode is set when OCR.CLKCTRL = 0. In this mode the serial clock is generated only during the transmission of the data bits. The ABRA outgoing frame passive phase duration jitters for +- one f_A cycle.

The example in the following figure assumes 50 MBaud input and 40 MBaud output baud rate, f_A = 80 MHz, back to back frames.

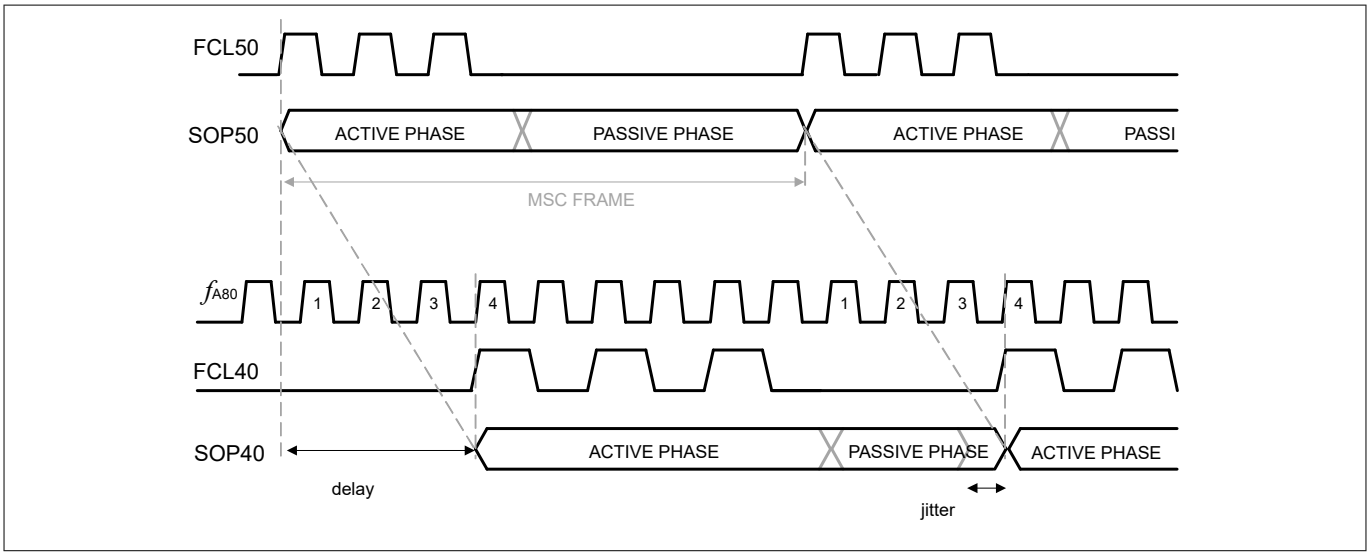


Figure 608 Frame jitter in active phase clock mode

28.3.2.4.2 Jitter in continuous clock mode

Continuous Clock mode is set when OCR.CLKCTRL = 1. In this mode the serial clock is always on. The jitter is one output bit time.

The average length of the output frames is equal to the length of the input frames. The ABRA block generates an averaging effect similar to that of a fractional divider. It introduces occasionally jitter of one bit time in the output frame length, in the passive phase of the frame, in order to adjust the average output time raster to the ideal input time raster.

The granularity of the input time raster is one input bit time (for example 20 ns at 50 MBaud).

The granularity of the output time raster is one output bit time (for example 25 ns at 40 MBaud).

The example in the following figure assumes 50 MBaud input baud rate and 40 MBaud output baud rate, f_A = 80 MHz, back to back frames.

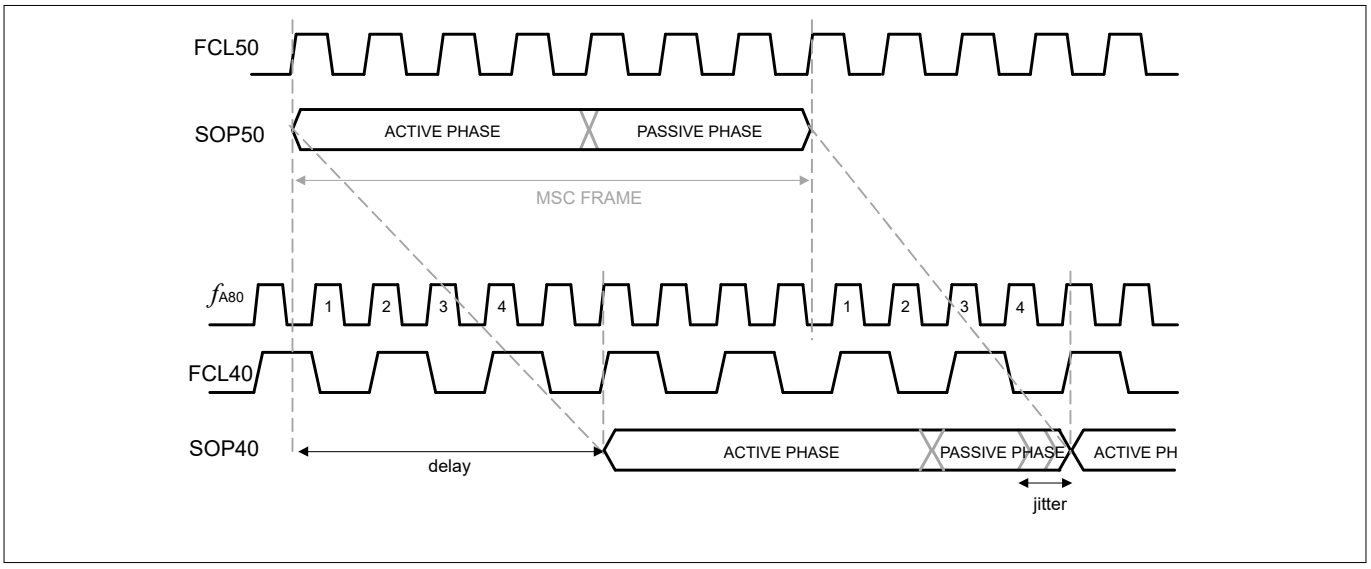


Figure 609 Frame jitter in continuous clock mode

28.3.2.5 Interrupt position with the ABRA block

The position of the MSC interrupts within a frame depends only on the internal time schedule of the MSC module in both use cases with and without the ABRA block. Therefore the interrupt position in the internal time raster is constant and is the same as in the previous versions of the MSC module and does not depend on the asynchronous baud rate which is seen on the pins.

This behavior affects only the downstream interrupts, which are:

- Data frame interrupt
- Command frame interrupt
- Time frame finished interrupt

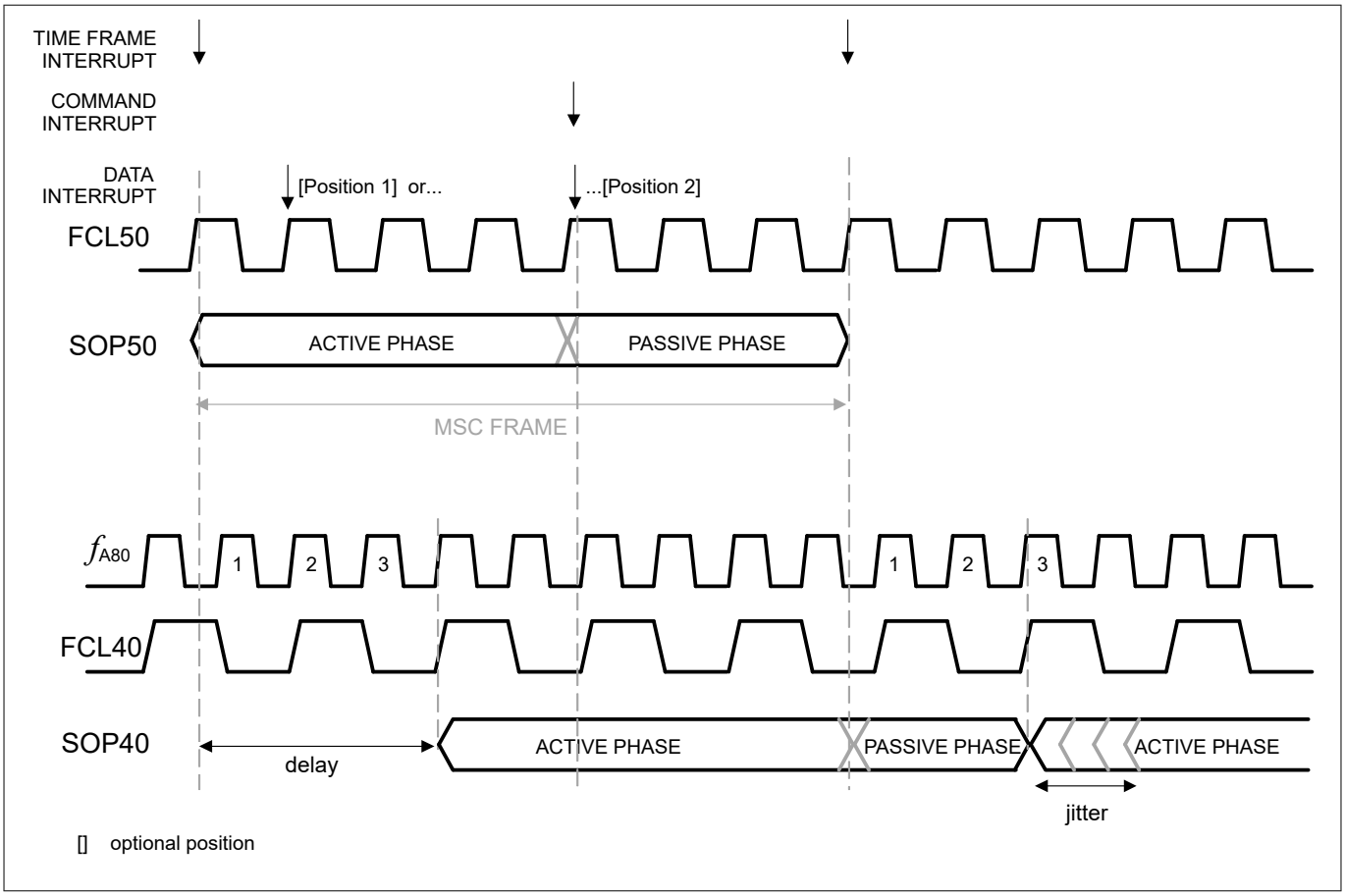


Figure 610 Interrupt position with the ABRA block

The downstream interrupts can be used for updates of the shadowed rw bit-fields for the next frame transmission. Such bit-fields are located in:

- Data register DD
- Command register DC
- Data control DSC
- Data status DSS

Due to the next-frame-update effect, the small jitter and delay effects of the frames as seen on the pins is not relevant, as well as the position of the interrupt in the asynchronous frame.

28.3.2.6 Configuring the ABRA block

This section describes all the configurable parameters of the ABRA block. They are located in the Asynchronous Block Configuration Register ABC.

Output baud rate

The ABRA block shifts out the data with a baud rate generated by dividing the clock f_A and using the bit-fields ABC.HIGH, ABC.LOW and ABC.NDA.

The high and the low time of the shift clock are generated by a separate 4-bit n-divider, each operating in the range of 1 to 16. Therefore the duty cycle can be finely configured to be 50% or any other ratio. This may provide some advantages when the communicating devices have asymmetrical timings (output delays or set-up/hold times). See the description of the bit-fields ABC.HIGH and ABC.LOW.

The output baud rate of the ABRA block is defined with the formula:

$$\text{BaudRate} = \frac{f_A}{(\text{ABC.NDA} + 1) \times (\text{ABC.LOW} + 1 + \text{ABC.HIGH} + 1)}$$

Committed baud rates

The following table defines the settings for the baud rates that are mandatory for the ABRA block for various devices. The ABRA ABC.LOW and HIGH are equal in the range of 0 to 7 and divide the f_A with 2, 4, 6, 8, 10, 12, 14, 16. The clock pre divider ABC.NDA=000_B.

The planned f_A frequencies for the various devices are the following:

160 MHz (80 MHz) and 200 MHz without FMPLL, 100 MHz with FMPLL.

This results in the following table (* baud rates above 40 MBaud are beyond the standard MSC specification):

Table 1213 Committed baud rates in MBaud

Division ratio	200 MHz	160 MHz	100 MHz	80 MHz
2	-	80,00[1)]	50,00*	40,00
4	50,00*	40,00	25,00[1)]	20,00
6	33,33	26,67[1)]	16,67	13,33
8	25,00[1)]	20,00	12,50	10,00
10	20,00	16,00	10,00	8,00
12	16,67	13,33	8,33	6,67
14	14,29	11,43	7,14	5,71
16	12,50	10,00	6,25	5,00

1) For baud rate of 80 MBaud, FM mode with NDD=0 must be used. For baud rates of 25 MBaud and 26,67 MBaud, it is recommended to use FM mode with NDD=2, so that the MSC module fills the ABRA FIFO with baud rate of 33.3 MBaud.

Continuous shift clock

As defined in the bit-field OCR.CLKCTRL, the shift clock signal can be active either:

- Only during the data transmission time
- Continuously

Overflow and underflow interrupts

If the input and the output timings of the ABRA block are not adjusted to each other, an overflow of the ABRA block will occur and an error interrupt will be triggered on line X.

Overflow occurs either if the input data rate is too high or the output data rate is too low.

The corresponding bit-fields are ABC.OVF, OFM, OIE.

Underflow occurs either if the input data rate is too low or the output data rate is too high.

The corresponding bit-fields are ABC.UNF, UFM, UIE

See also the Interrupt request compressor section.

This feature can not guarantee protection against software errors like reconfigure the baud rates in the middle of a frame.

The configuration of the ABRA block and the MSC kernel is static and the time behavior is deterministic and calculable. Therefore, an overflow/underflow event always signals some configuration error resulting in unadjusted input/output baud rate ratio (see the Adjusting the passive phase of a frame section). In normal operation and with correctly configured baud rates (clocks and clock dividers) these events do not occur.

In case of underflow/overflow event the output frame will be corrupt. The simplest solution to a sporadic event caused by a transient (noise, alpha particle) is to reset the module, reconfigure and restart the communication.

Bypass mode

If the ABRA block is not used, it is disabled and bypassed through the bit-field ABC.ABB.

Related information

Adjusting the passive phase of a frame on page 5117
Interrupt request compressor on page 5130

28.3.2.7 Implementation issues

The ABRA block operates using synchronization FIFO. The MSC module writes its serial stream to the synchronization FIFO with its serial shift clock and the same data is read out of the FIFO by using the ABRA serial shift clock, generating out of the f_A.

Three signals are being transformed: the serial shift clock, the serial data stream and the enable (chip select) signals.

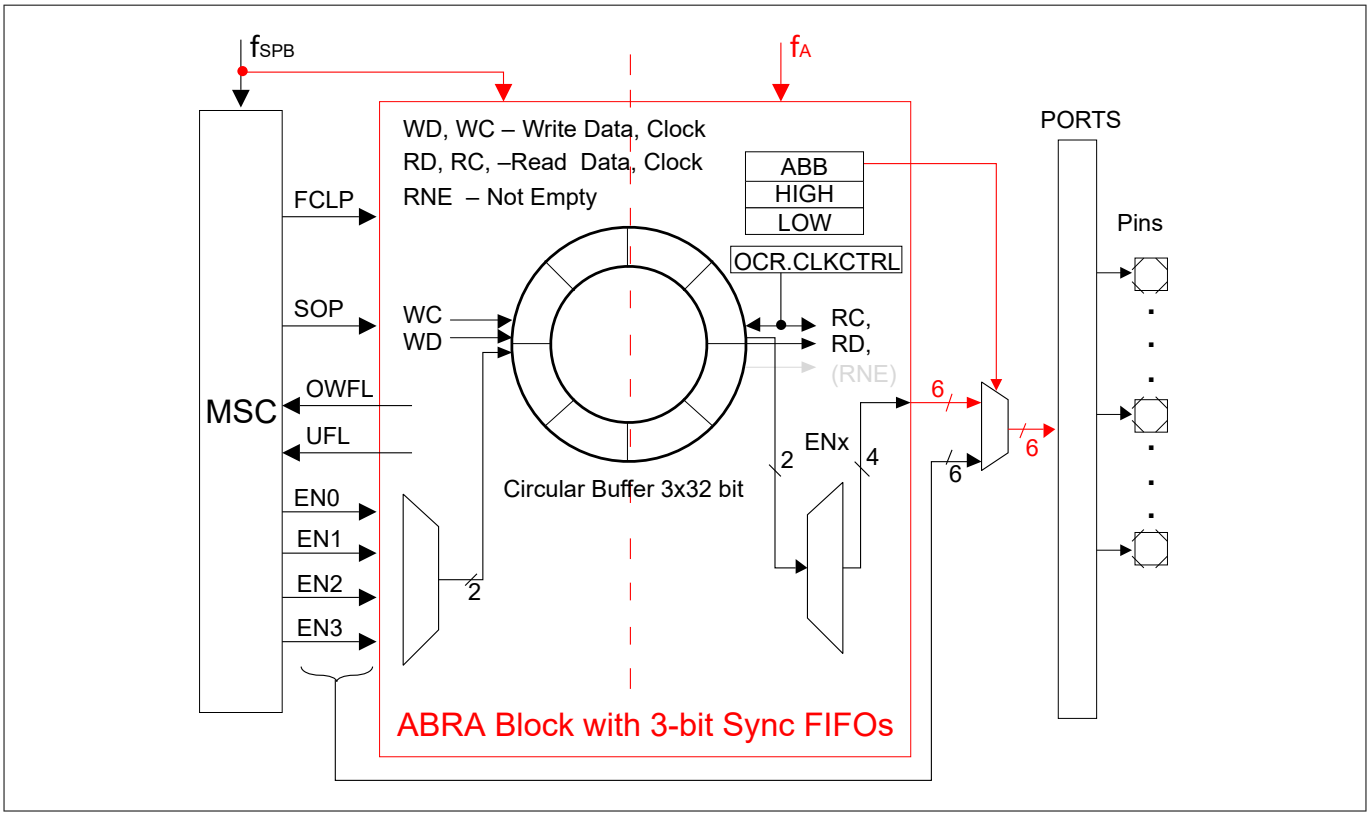


Figure 611 Implementation details of the ABRA block

28.3.2.8 ABRA disable and sleep

This section describes the behavior of the MSC module in case of:

- Switching the module on/off - disable behavior
- Power saving - Sleep mode

The main concept is to provide behavior as similar as the behavior of the MSC module without the ABRA block, as much as possible.

28.3.2.8.1 Disable and sleep behavior

There is no difference in the behavior of the module in case of entering/leaving the disable and sleep state. Disable state is controlled by a bit set/cleared by software, sleep state is controlled by a hardware signal which can be disabled or enabled.

On disable/sleep state request, the module finishes the possibly ongoing frames (downstream or upstream), then acknowledges the request and the module clock is switched off.

On leaving the disable/sleep state, the module continues with the operation at the same point where it entered the corresponding state.

Refer to the chapter Debug information for suspend behavior.

Related information

Debug information on page 5186

28.3.3 MSC interrupts

The MSC module has seven interrupt sources and five service request outputs. A service request output is able to generate interrupts or DMA requests(controlled by a service request control register). The service request output assignment, interrupt or DMA request, is specific for each microcontroller that is using the MSC. In this section, the term “interrupt request” has the meaning of “service request” that is able to handle interrupt or DMA requests.

Each interrupt source is provided with a status flag, enable bit(s) with software set/clear capability and an interrupt node pointer. An interrupt event, internally generated as a request pulse, is always stored in an interrupt status flag that is located in the Interrupt Status Register ISR. All interrupt status flags can be set or cleared individually by software through the interrupt Set Clear Register ISC. Software-controlled interrupt generation can be initiated by setting the interrupt status flag of the corresponding interrupt. Each interrupt source can be enabled or disabled individually. When an interrupt event is enabled, a 2-bit interrupt node pointer determines which of the service request outputs will be activated. See the Interrupt request compressor section.

The following table shows the seven MSC interrupt sources.

Table 1214 MSC interrupt sources

Interrupt type	Generated by
Data frame interrupt	Downstream channel
Command frame interrupt	
Time frame finished interrupt	
Receive data interrupt	Upstream channel
Upstream time-out Interrupt	
Sync FIFO overflow	ABRA block
Sync FIFO underflow	

Related information

Interrupt request compressor on page 5130

28.3.3.1 Data frame interrupt

A data frame interrupt can be generated when either the first or the last data bit of the downstream channel is shifted out and becomes available at the SO output line (see also related information). Bit ICR.EDIE selects which case is selected.

***Note**: If ICR.EDIE=10_B, an interrupt at the first data bit is only generated if DSC.NDBL is not equal 00000_B. This means, at least one SRL bit must be shifted out for the first data bit shifted interrupt to become active.*

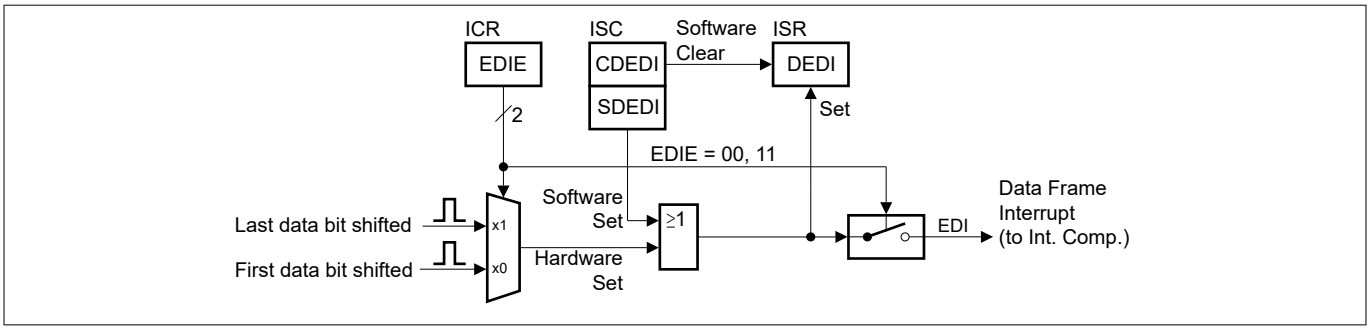


Figure 612 Data frame interrupt control

Slow Channel mode

This interrupt can also be used in Slow Channel mode.

In the figure below the behavior of the data frame interrupt in Slow Channel mode is represented.

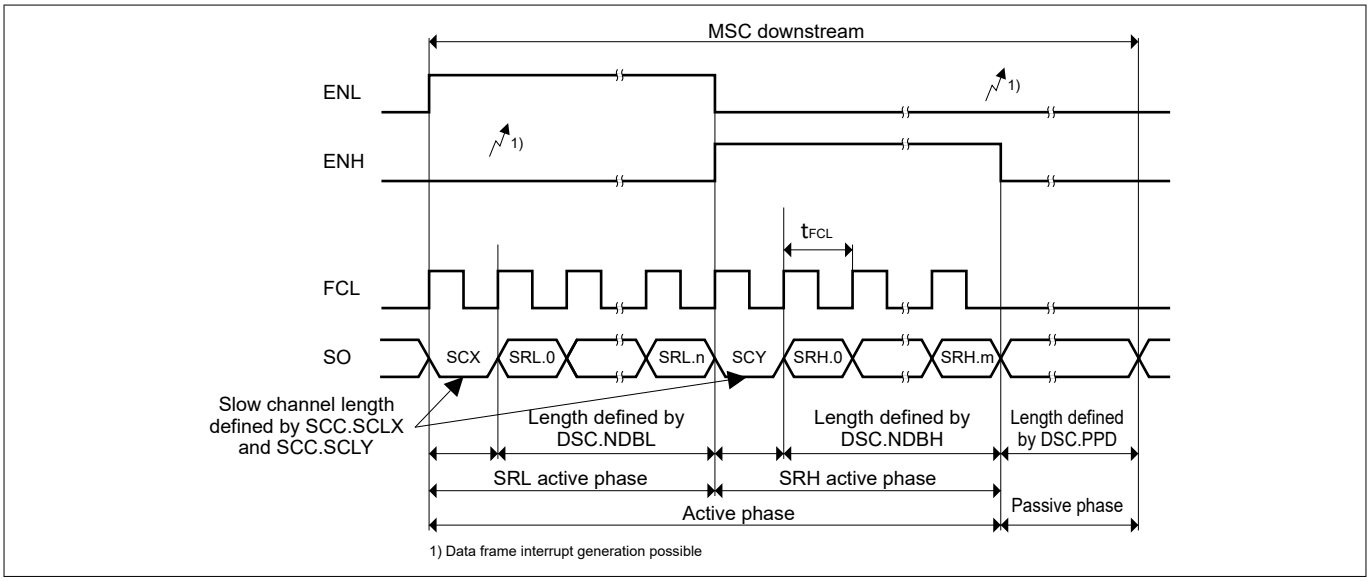


Figure 613 Data frame interrupt diagram (slow channel)

Manchester Code mode

This interrupt can also be used in Manchester Code mode.

In the figure below the behavior of the data frame interrupt in Manchester Code mode is explained. Note that Manchester Code mode can only be used in combination with slow channel.

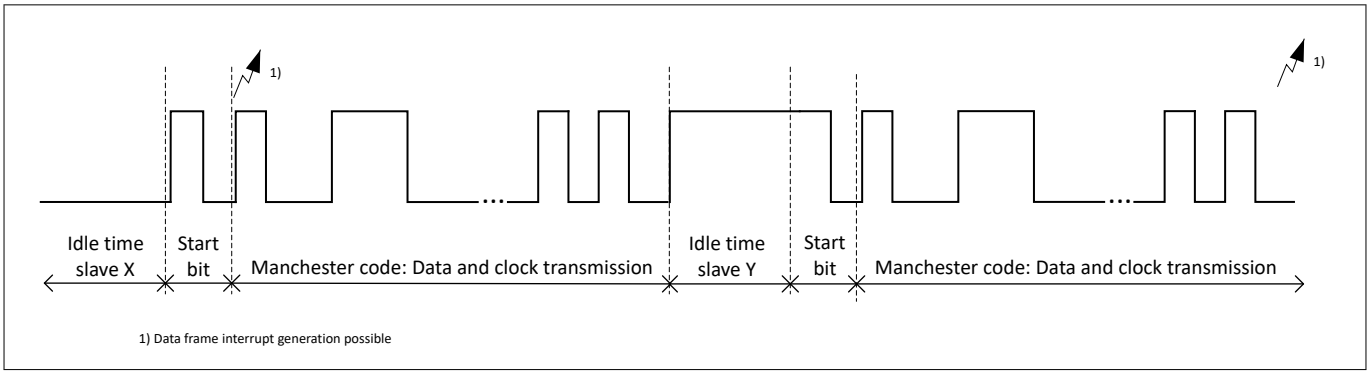


Figure 614 Data frame interrupt behavior in Manchester code mode

Related information

Frame formats and definitions on page 5082

28.3.3.2 Command frame interrupt

A command frame interrupt can be generated at the end of a downstream channel command frame, directly after the active phase, see also related information.

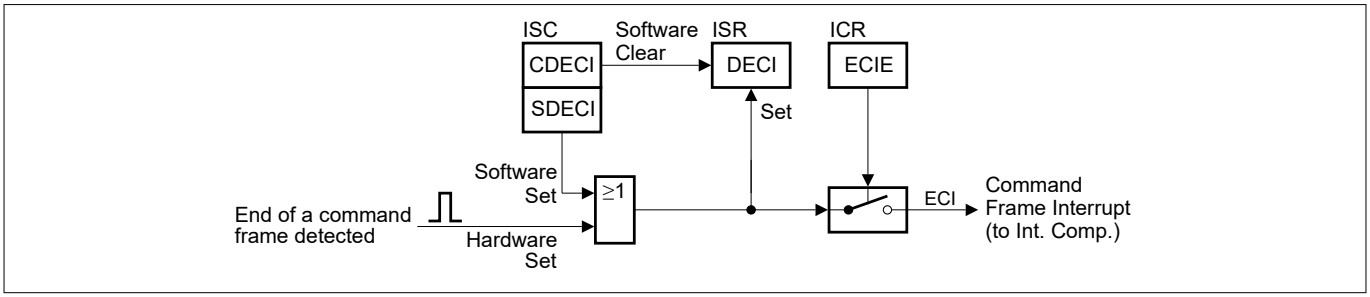


Figure 615 Command frame interrupt control

Command frame interrupts for slow channel X and Y

A command frame interrupt for slow channel X can be generated when the command frame X is moved from the DCSX register to an internal register to be shifted on the SO output and the DCSX register can be written again. This interrupt can be enabled in bit-field ICR.CSXIE. The same behavior applies to the command frame interrupt for slow channel Y, this interrupt is generated when the command frame Y is moved from the DCSY register to an internal register to be shifted on the SO output and the DCSY register can be written again. This interrupt can be enabled in bit-field ICR.CSYIE.

When the system is configured to run with only one slave, the interrupts can be configured accordingly.

In the diagrams below both interrupt generation are explained.

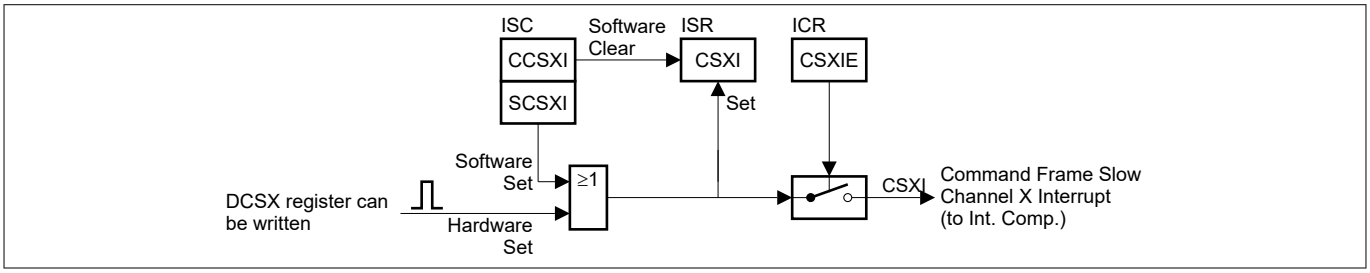


Figure 616 Command frame interrupt for slow channel X

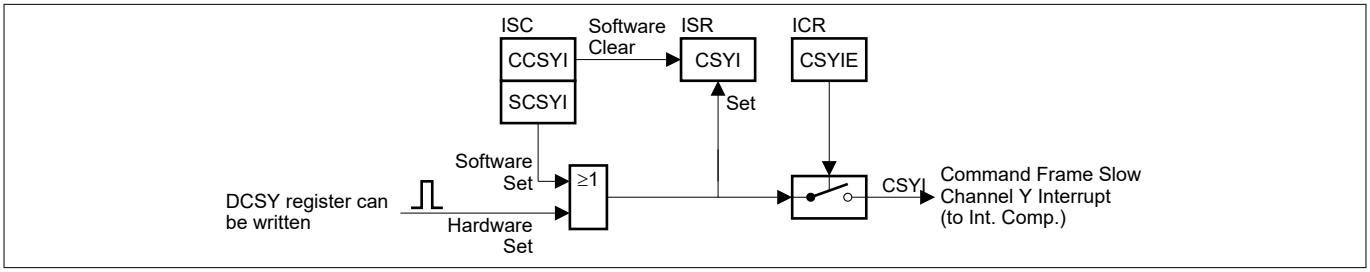


Figure 617 Command frame interrupt for slow channel Y

***Note**: This interrupt can be used in all modes that allow slow channel, in legacy with slow channel and Manchester code transmission.*

Related information

Frame formats and definitions on page 5082

28.3.3.3 Time frame finished interrupt

A time frame finished interrupt can be generated at the end of a downstream channel passive phase.

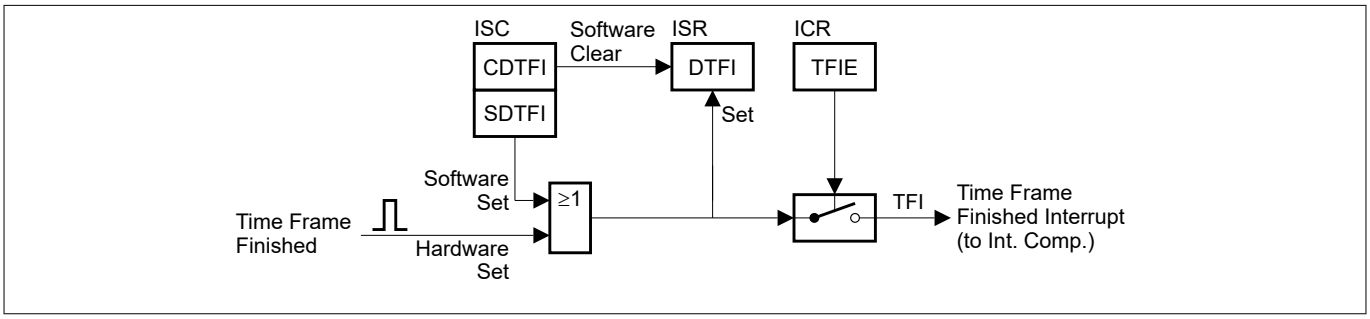


Figure 618 Time frame finished interrupt control

Slow channel mode

This interrupt can also be used in slow channel mode.

In the figure below the behavior of the time frame finished interrupt in Slow Channel mode is represented.

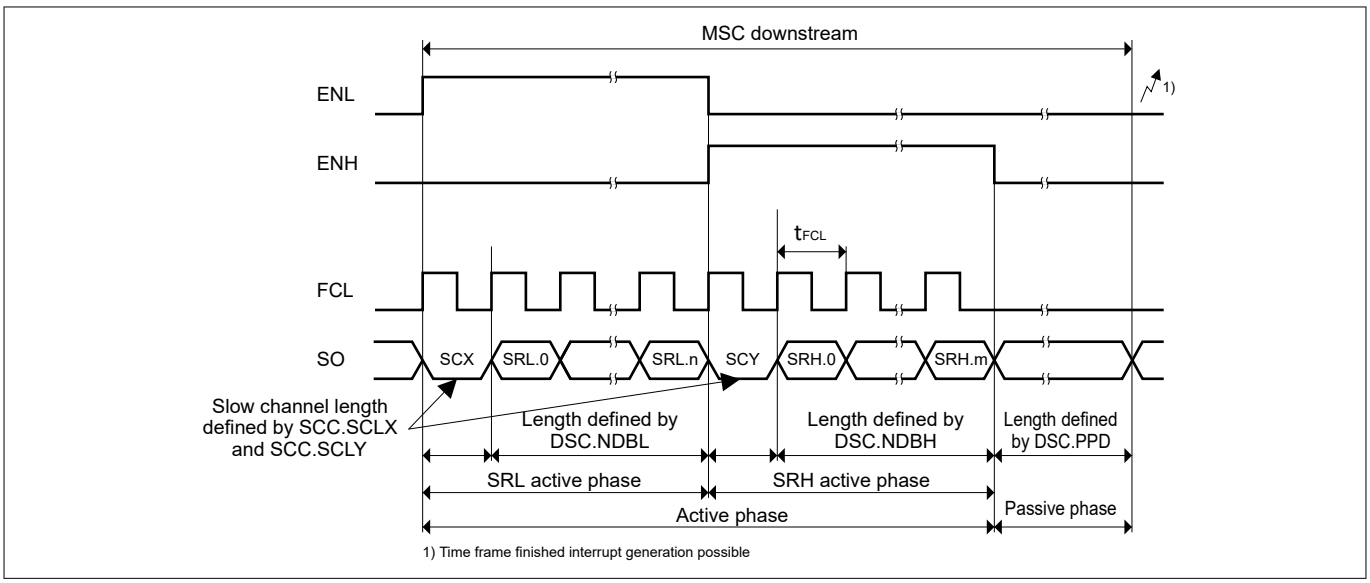


Figure 619 Time frame finished interrupt (slow channel)

Time frame finished interrupt in Manchester code mode

A time frame finished interrupt can be generated at the end of a downstream channel frame when the last bit of the downstream frame is shifted out and becomes available at the SO output.

Since the time frame finished interrupt in legacy mode is based on the passive phase and in Manchester Code mode there is no passive phase available, a new interrupt has been added for indicating the end of the frame in this mode, time frame finished Manchester code Interrupt (TFMI).

In the diagram below the interrupt generation is explained.

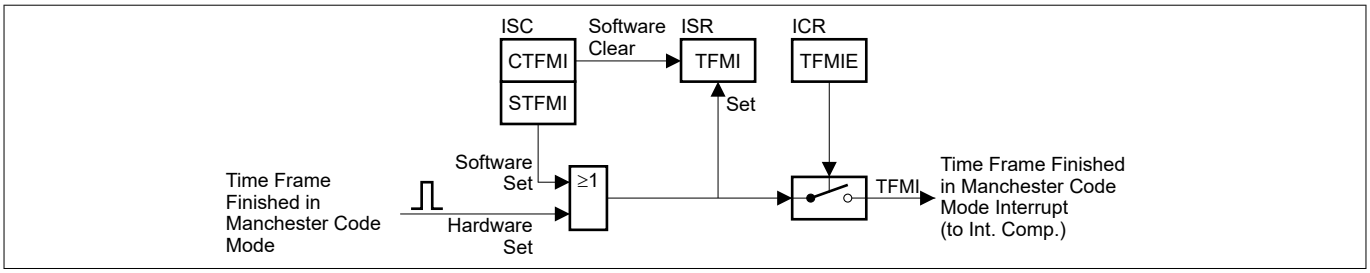


Figure 620 Time frame finished Manchester code interrupt

In the figure below the behavior of the Interrupt is explained.

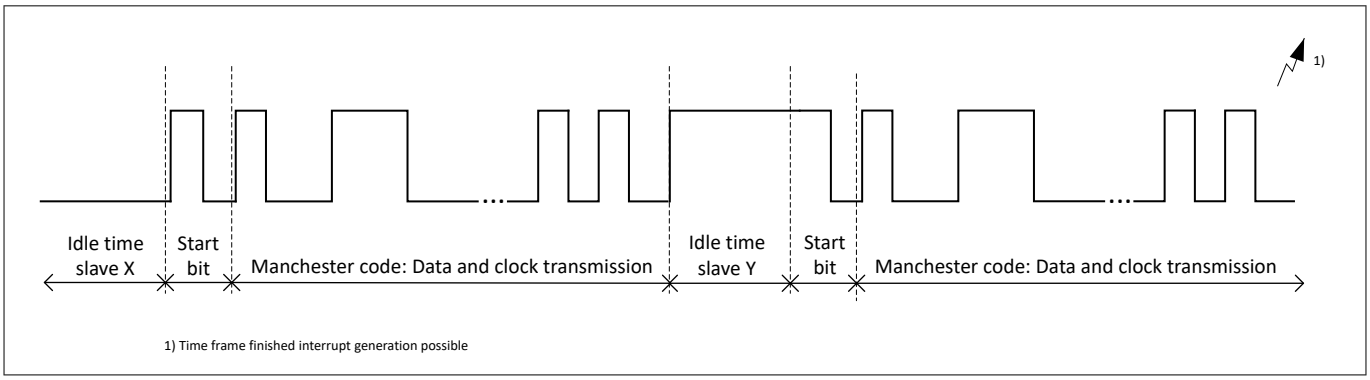


Figure 621 Time frame finished Manchester code interrupt behavior

28.3.3.4 Receive data interrupt

Whenever the upstream channel receives data in the UDx $(x = 0 - 3)$ registers, the MSC is able to generate an interrupt. Three interrupt generation conditions can be selected for the receive data interrupt:

- Each update of UDx $(x = 0 - 3)$ generates a receive data interrupt
- Each update of UDx $(x = 0 - 3)$ generates a receive data interrupt when the updated value is not equal 00_H
- Only an update of register UD3 generates a receive data interrupt

The selection of the interrupt generation condition is controlled by bit-field ICR.RDIE. Setting ICR.RDIE=0 disables the receive data interrupt in general. ISR.URDI is the interrupt status flag that can be set or cleared when writing bits ISC.SURDI or ISC.CURDI with a 1. If the fractional divider is used, then the software should clear the URDI flag by using ISC.CURDI with a delay of one bit time after the interrupt signal has been generated. The same applies to clearing the UDx.V flag via UDx.C. Otherwise, due to the latencies introduced by the fractional divider, the clear may fail.

The receive interrupt is generated either after the second Stop bit or delayed for one bit time by setting the USR.SRDC, as shown in related information.

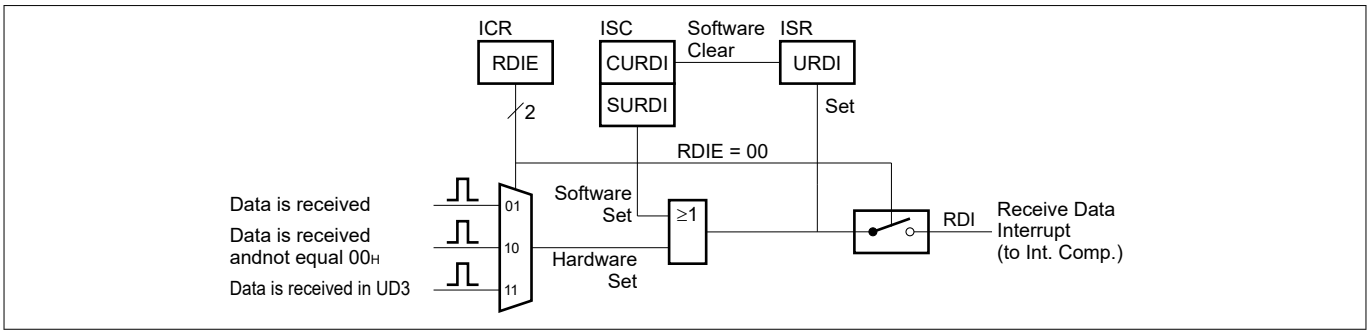


Figure 622 Receive data interrupt control

Related information

Data frames on page 5098

28.3.3.5 Interrupt request compressor

The interrupt control logic of the MSC uses an interrupt compressing scheme that allows high flexibility in interrupt processing. The seven interrupt sources can be directed to five service request outputs SR[4:0] as shown in the figure below. This makes it possible to connect more than one interrupt source to one interrupt output SRx.

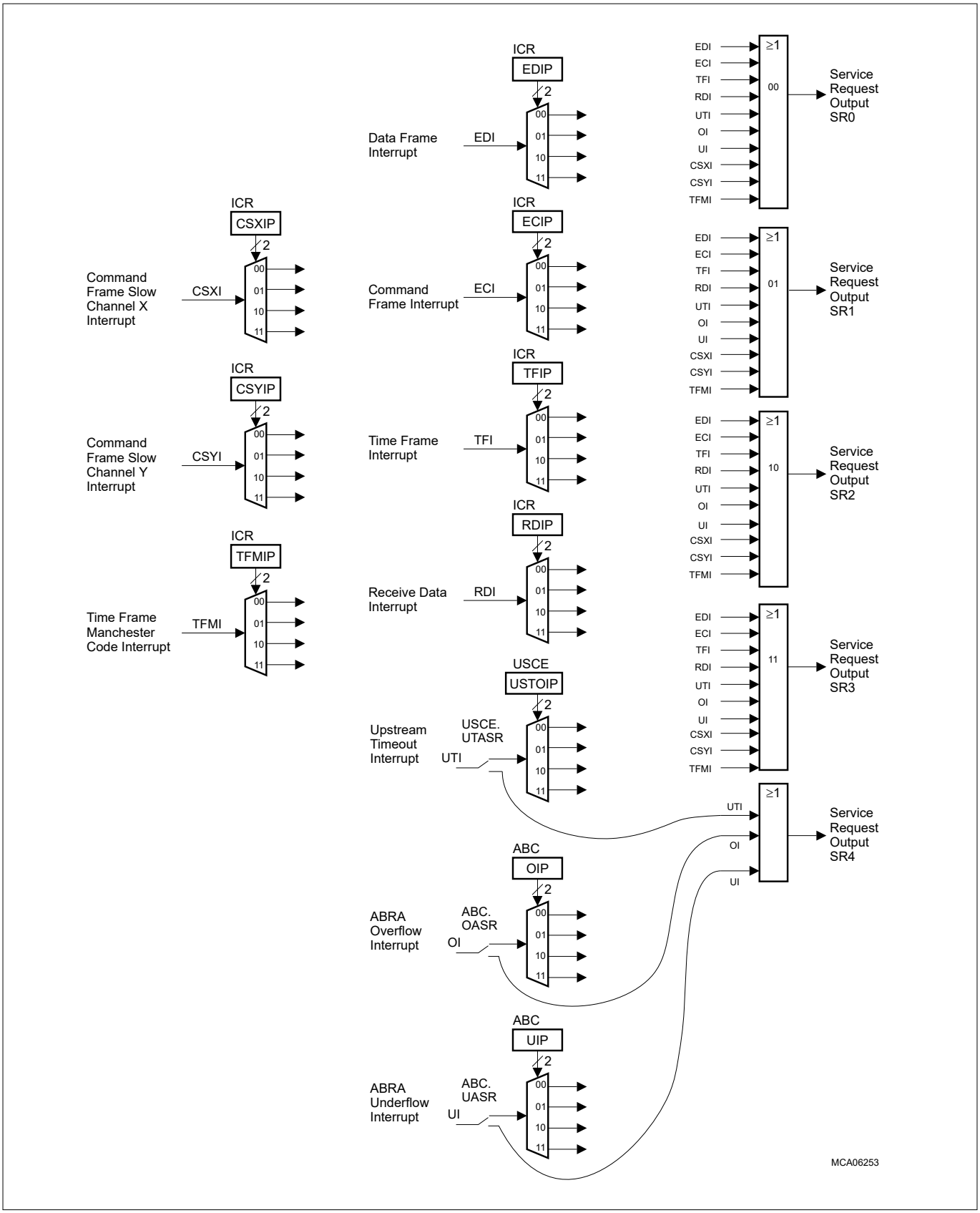


Figure 623 MSC interrupt request compressor

28.3.4 Clock control

The following section describes the clock control and the baud rate generation of the MSC module.

The MSC module is provided with two independent clock signals (see following figure):

- f_{CLC}
  This is the module clock that is used inside the MSC kernel for control purposes such as clocking of control logic and register operations. The frequency of f_{CLC} is always identical to the system clock frequency f_{SPB}. The clock control register MSC_CLC makes it possible to enable/disable f_{CLC} under certain conditions
- f_{MSC}
  This clock is the module clock that is used inside the MSC for baud rate generation of the serial upstream and downstream channel. The fractional divider register MSC_FDR controls the frequency of f_{MSC} and makes it possible to enable/disable it independent of f_{CLC}
- f_{MSCA}
  This clock is the downstream clock which is used together with the ABRA block. In this case, the upstream baud rate can be fine-tuned with the fractional divider and the downstream channel frequency is configured with an N-divider

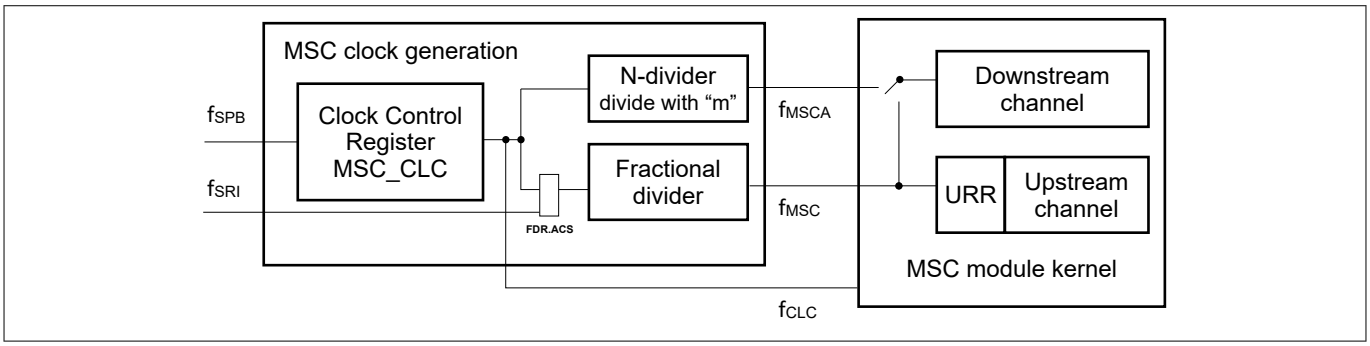


Figure 624 MSC module clock generation

28.3.4.1 Clock control without the ABRA block

Not using the ABRA block is the standard, compatibility use-case, which guarantees the full compatibility with the standard MSC module not containing the ABRA block. The clock generation is therefore identical with the standard MSC clock generation. The additional N-divider (dividing with the division factor named “m”) is not used. Only the fractional divider is used (using the factor named “n” in the formulas below).

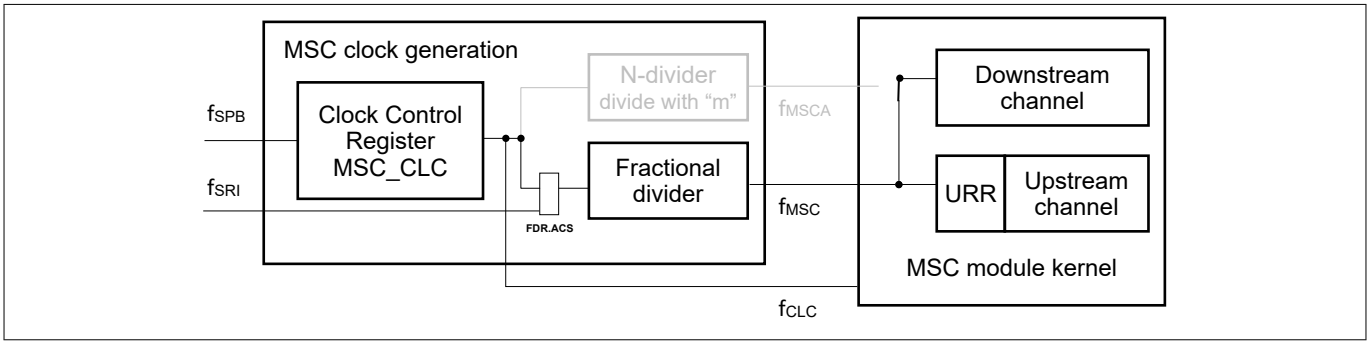


Figure 625 MSC module clock generation

To use the f_{SPB} clock source for the MSC output the bit-field FDR.ACS has to be set to 0 (default value). The following two formulas define the frequency of f_{MSC}:

$$f_{MSC} = f_{SPB} \times \frac{1}{n} \quad \text{with } n = 1024 - \text{MSC.FDR.STEP} \tag{36}$$

Valid for normal divider mode (FDR.DM=01_B)

$$f_{MSC} = f_{SPB} \times \frac{n}{1024} \quad \text{with } n = 0\text{-}1023 \tag{37}$$

Valid for fractional divider mode (FDR.DM=10_B)

The f_{SRI} clock source is synchronous to the f_{SPB} clock. To use the f_{SRI} clock source for the MSC output the bit-field FDR.ACS has to be activated and set to 1. The following two formulas define the frequency of f_{MSC}:

$$f_{MSC} = f_{SRI} \times \frac{1}{n} \quad \text{with } n = 1024 - \text{MSC.FDR.STEP} \tag{38}$$

Valid for normal divider mode (FDR.DM=01_B) and f_{SRI} clock source

$$f_{MSC} = f_{SRI} \times \frac{n}{1024} \quad \text{with } n = 0\text{-}1023 \tag{39}$$

Valid for fractional divider mode (FDR.DM=10_B) and f_{SRI} clock source

***Note**: The behavior is undefined when the configuration of FDR.ACS is changed in between an ongoing transaction*

Downstream channel baud rate

As the clock signal FCL of the synchronous downstream channel is always half the frequency of f_{MSC}, the resulting downstream channel baud rate is defined by:

$$\text{Baud rate}_{MSC} = f_{SPB} \times \frac{1}{2 \times (1024 - \text{MSC.FDR.STEP})} \tag{40}$$

Valid for normal divider mode (FDR.DM=01_B)

$$\text{Baud rate}_{MSC} = f_{SPB} \times \frac{\text{MSC.FDR.STEP}}{2 \times 1024} \tag{41}$$

Valid for fractional divider mode (FDR.DM=10_B)

$$\text{Baud rate}_{MSC} = f_{SRI} \times \frac{1}{2 \times (1024 - \text{MSC.FDR.STEP})} \tag{42}$$

Valid for normal divider mode (FDR.DM=01_B) and f_{SRI} clock source

$$\text{Baud rate}_{MSC} = f_{SRI} \times \frac{\text{MSC.FDR.STEP}}{2 \times 1024} \tag{43}$$

Valid for fractional divider mode (FDR.DM=10_B) and f_{SRI} clock source

Upstream channel baud rate

The baud rate of the asynchronous upstream channel is derived from the module clock f_{MSC} by a programmable clock divider selected by bit-field USR.URR (see also the equation in the Upstream channel baud rate section). The divide factor DF can be at minimum 4 and at maximum 256. The following formulas can be used for both configurations FDR.ACS set to 0 and FDR.ACS set to 1. In case FDR.ACS set to 1, f_{SRI} clock source is used as a substitute for f_{SPB}.

$$\text{Baud rate}_{MSC} = f_{SPB} \times \frac{1}{DF \times (1024 - \text{MSC.FDR.STEP})} \quad (44)$$

Valid for normal divider mode (FDR.DM=01_B)

$$\text{Baud rate}_{MSC} = f_{SPB} \times \frac{\text{MSC.FDR.STEP}}{DF \times 1024} \quad (45)$$

Valid for fractional divider mode (FDR.DM=10_B)

Related information

Upstream channel baud rate on page 5100

28.3.4.2 Clock control when using the ABRA block

Using the ABRA block makes additional considerations regarding the baud-rate adjustments necessary.

An additional N-divider DSTE.NDD is used for configuring the intermediate downstream baud rate the MSC module is using for writing to the ABRA block (dividing the f_{SPB} with the integer division factor named “m” in the formulas below). The final output downstream baud rate is configured in the ABRA block itself. The following restriction applies: the ABRA output downstream baud rate is between half and whole intermediate baud rate.

When the ABRA block is active, the fractional divider FDR.STEP is used only for adjusting the upstream baud rate (using the factor named “n” in the formulas below).

In case FDR.ACS set to 1, f_{SRI} clock source is used as a substitute for f_{SPB}.

Attention*: *The following restriction applies: the upstream baud rate must be less than one tenth of the f_{SPB}. See also related information.

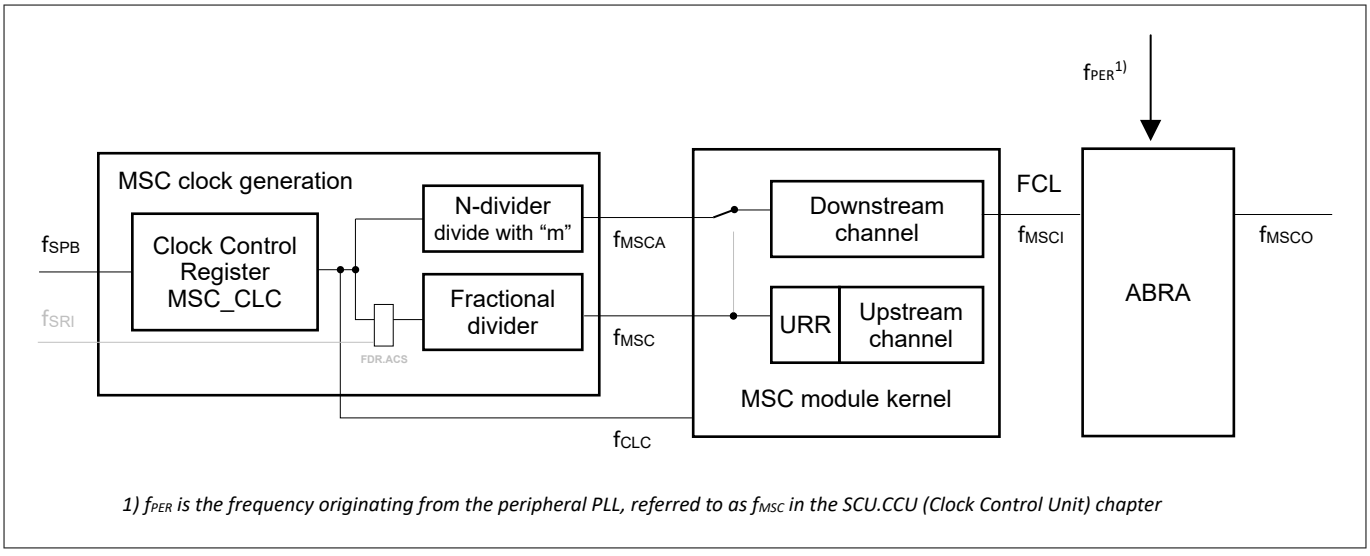


Figure 626 MSC module clock generation

The following two formulas define the frequency of f_{MSC}:

$$f_{MSC} = f_{SPB} \times \frac{1}{n} \quad \text{with } n = 1024 - \text{MSC.FDR.STEP} \tag{46}$$

Valid for normal divider mode (FDR.DM=01_B)

$$f_{MSC} = f_{SPB} \times \frac{n}{1024} \quad \text{with } n = 0\text{-}1023 \tag{47}$$

Valid for fractional divider mode (FDR.DM=10_B)

The following formula defines the frequency of f_{MSCA}:

$$f_{MSCA} = f_{SPB} \times \frac{1}{m} \quad \text{with } m = 1 - 16 \tag{48}$$

Intermediate downstream channel baud rate

When the ABRA block and the downstream N-divider are enabled with ABC.ABB, the intermediate clock signal FCL with frequency f_{MSCI} of the synchronous downstream channel going into the ABRA block is always half the frequency of f_{MSCA}:

$$\text{Baud rate}_{MSCI} = f_{SPB} \times \frac{1}{2 \times m} \tag{49}$$

The final output baud rate f_{MSCO} can be calculated as described in Output baud rate.

Upstream channel baud rate

The baud rate of the asynchronous upstream channel is derived from the module clock f_{MSC} by a programmable clock divider selected by bit-field USR.URR (see also equation 3 in related information, chapter upstream channel baud rate). The divide factor DF can be at minimum 4 and at maximum 256. Nevertheless, the minimum DF factor when using ABRA block should be 8. See the note in the chapter Upstream channel (related information).

$$\text{Baud rate}_{MSC} = f_{SPB} \times \frac{1}{\text{DF} \times (1024 - \text{MSC.FDR.STEP})} \tag{50}$$

Valid for normal divider mode (FDR.DM=01_B)

$$\text{Baud rate}_{MSC} = f_{SPB} \times \frac{\text{MSC.FDR.STEP}}{\text{DF} \times 1024} \tag{51}$$

Valid for fractional divider mode (FDR.DM=10_B)

Example for setting the fractional divider

This example shows how to receive upstream frames generated from 40 MHz downstream clock, when the module baud-rate frequency is $\frac{100\,\text{MHz}}{2} = 50\,\text{MHz}$. The fractional divider is used to adjust the FCL clock of 50 MHz to FCL clock of 40 MHz, that is, it multiplies with $\frac{4}{5}$. The best approximation of $\frac{4}{5}$ with $\frac{x}{1024}$ is for $x = 819$, giving an error of 0.025%. So in this case, the optimal setting for FDR.STEP is 819. In the next stage of

the divider cascade, the DF power-of-two-divider divides the clock further to the final baud rate, matching the baud rate generated by the MSC slave device.

Related information

Upstream channel on page 5097
Upstream channel baud rate on page 5100
Upstream channel input control on page 5107
Configuring the ABRA block on page 5121

28.3.4.3 Baud rate configuration

This section describes the available baud rates for the various devices.

The planned f_{MSC} frequencies are the following:

- f_{SPB} (The f_{SPB} depends on the system configuration and this is 100 MHz)
- f_{SRI}. (The f_{SRI} depends on the system configuration and this is 400 MHz or 500 MHz)

This results in the following table:

Table 1215 Baud rate configuration

Division ratio	f_{SRI} = 500 MHz	f_{SRI} = 400 MHz	f_{SPB} = 100 MHz
2	-	-	50.00
4	-	-	25.00
6	-	-	16.67
8	-	50	12.50
10	50	40	10.00
12	41.6	33.34	8.33
14	35.71	28.57	7.14
16	31.25	25	6.25

These baud rates are bypassing ABRA. The available baud rates using ABRA are already listed in the related chapter.

28.4 Registers

28.4.1 Register overview - access mode glossary

Table 1216 Register overview - access mode glossary

Keyword	Description
E	Access protection using PROT register PROTE.
SE	Access protection using PROT register PROTSE.
APU-P	Protection group consisting of registers ACCEN_WRA, ACCEN_WRB, ACCEN_RDA, ACCEN_RDB, ACCEN_VM, ACCEN_PRS.
P	Access protection using *APU-P* registers.
BE	Always returns a Bus Error.
U	No access restrictions.
PROT	Access restrictions as defined in the PROT register access rules.

28.4.2 Registers overview - MSC (ascending offset address)

Table 1217 Registers overview - MSC (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CLC	Clock control register	000_H	P	P, SV, E	Application Reset	5139
OCS	OCDS control and status register	004_H	P	SV, P	Debug Reset	5140
ID	Module identification register	008_H	P	BE	PowerOn Reset	5141
RST_CTRLA	Reset control register A	$00C_H$	P	P, SV, E	Application Reset	5141
RST_CTRLB	Reset control register B	010_H	P	P, SV, E	Application Reset	5142
RST_STAT	Reset status register	014_H	P	BE	Application Reset	5143
PROTE	PROT register endinit	018_H	U	SV, PROT	Application Reset	5143
PROTSE	PROT register safe endinit	$01C_H$	U	SV, PROT	Application Reset	5145
ACCEN_WRA	Write access enable register A	020_H	U	SE, SV	Application Reset	5146
ACCEN_WRB	Write access enable register B	024_H	U	SE, SV	Application Reset	5147
ACCEN_RDA	Read access enable register A	028_H	U	SE, SV	Application Reset	5147

(table continues...)

Table 1217 (continued) Registers overview - MSC (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ACCEN_RDB	Read access enable register B	$02C_H$	U	SE, SV	Application Reset	5148
ACCEN_VM	VM access enable register	030_H	U	SE, SV	Application Reset	5148
ACCEN_PRS	PRS access enable register	034_H	U	SE, SV	Application Reset	5149
FDR	Fractional divider register	040_H	P	SV, P, E	Kernel Reset	5150
USR	Upstream status register	044_H	P	P	Kernel Reset	5151
DSC	Downstream control register	048_H	P	P	Kernel Reset	5152
DSS	Downstream status register	$04C_H$	P	P	Kernel Reset	5154
DD	Downstream data register	050_H	P	P	Kernel Reset	5156
DC	Downstream command register	054_H	P	P	Kernel Reset	5156
DSDSL	Downstream select data source low register	058_H	P	P	Kernel Reset	5157
DSDSH	Downstream select data source high register	$05C_H$	P	P	Kernel Reset	5157
ESR	Emergency stop register	060_H	P	P	Kernel Reset	5158
UDx	Upstream data register x	070_H+x* 4	P	P	Kernel Reset	5158
ICR	Interrupt control register	080_H	P	P	Kernel Reset	5159
ISR	Interrupt status register	084_H	P	BE	Kernel Reset	5162
ISC	Interrupt set clear register	088_H	P	P	Kernel Reset	5163
OCR	Output control register	$08C_H$	P	P	Kernel Reset	5165
OECR	Output extended control register	090_H	P	P	Kernel Reset	5167
SCC	Slow channel configuration register	094_H	P	P	Kernel Reset	5170
DSCE	Downstream control enhanced register	098_H	P	P	Kernel Reset	5172
USCE	Upstream control enhanced register	$09C_H$	P	P	Kernel Reset	5174
DSDSLE	Downstream select data source low extension register	$0A0_H$	P	P	Kernel Reset	5176
DSDSHE	Downstream select data source high extension register	$0A4_H$	P	P	Kernel Reset	5176

(table continues...)

Table 1217 (continued) Registers overview - MSC (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ESRE	Emergency stop extension register	$0A8_H$	P	P	Kernel Reset	5177
DSTE	Downstream timing extension register	$0AC_H$	P	P	Kernel Reset	5178
DDM	Downstream data mirror register	$0B0_H$	P	P	Kernel Reset	5179
DDE	Downstream data extension register	$0B4_H$	P	P	Kernel Reset	5179
DCM	Downstream command mirror register	$0B8_H$	P	P	Kernel Reset	5180
DCE	Downstream command extension register	$0BC_H$	P	P	Kernel Reset	5181
ABC	Asynchronous block configuration register	$0C0_H$	P	P	Kernel Reset	5181
DCSX	Downstream command slow channel X register	$0C4_H$	P	P	Kernel Reset	5184
DCSY	Downstream command slow channel Y register	$0C8_H$	P	P	Kernel Reset	5184

28.4.3 Clock control register

Register CLC controls the module clock signal

CLC Offset address: 000_H

Clock control register Application Reset value: $0000\ 0003_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												EDIS	0	DISS	DISR
r												rw	r	rh	rw

Field	Bits	Type	Description
DISR	0	rw	**Module Disable Request Bit** Used for enable/disable control of the module. 0_B Enable the module clock 1_B Stop the module clock
DISS	1	rh	**Module Disable Status Bit** 0_B Module clock is enabled 1_B Module is not clocked

(table continues...)

(continued)

Field	Bits	Type	Description
EDIS	3	rw	**Sleep Mode Enable Control** Used to control the module's reaction to sleep mode. 0_B Sleep mode request is enabled and functional 1_B Module disregards the sleep mode control signal
0	2, 31:4	r	**Reserved**

28.4.4 OCDS control and status register

The OCDS Control and Status register OCS controls the module's behavior in suspend mode (used for debugging).

Register OCS is cleared by Debug Reset.

If OCDS is being disabled, the OCS register value will not change. When OCDS is disabled the OCS suspend control is ineffective.

OCS Offset address: 004_H

OCDS control and status register Debug Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		SUSSTA	SUS_P	SUS				0							
r		rh	w	rw				r							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															
r															

Field	Bits	Type	Description
SUS	27:24	rw	**OCDS Suspend Control** Controls the sensitivity to the suspend signal coming from the OCDS Trigger Switch (OTGS) Not listed combinations are reserved. 0_H Will not suspend 1_H Hard suspend: Clocks will be disabled immediately 2_H Soft suspend: Clock will be disabled after kernel acknowledge 3_H Reserved, do not use
SUS_P	28	w	**SUS Write Protection** SUS is only written when SUS_P is 1, otherwise unchanged. Read as 0.
SUSSTA	29	rh	**Suspend State** 0_B Module is not (yet) suspended 1_B Module is suspended
0	23:0, 31:30	r	**Reserved** Read as 0; Written with 0

Table 1218 Access mode restrictions of OCS sorted by descending priority

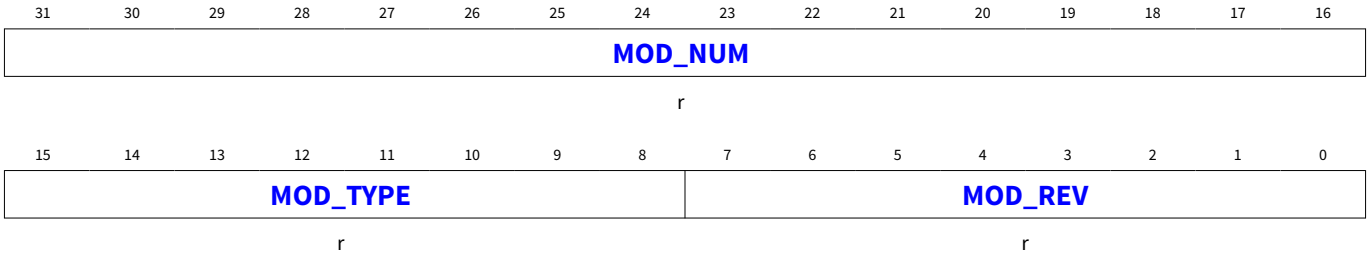
Mode name	Access mode		Description
write 1 to .**SUS_P**	rw	SUS	Set SUS_P during write access
(default)	r	SUS	

28.4.5 Module identification register

The MSC Module Identification Register ID contains read-only information about the module version.

ID Offset address: 008_H

Module identification register PowerOn Reset value: $0028\ C014_H$

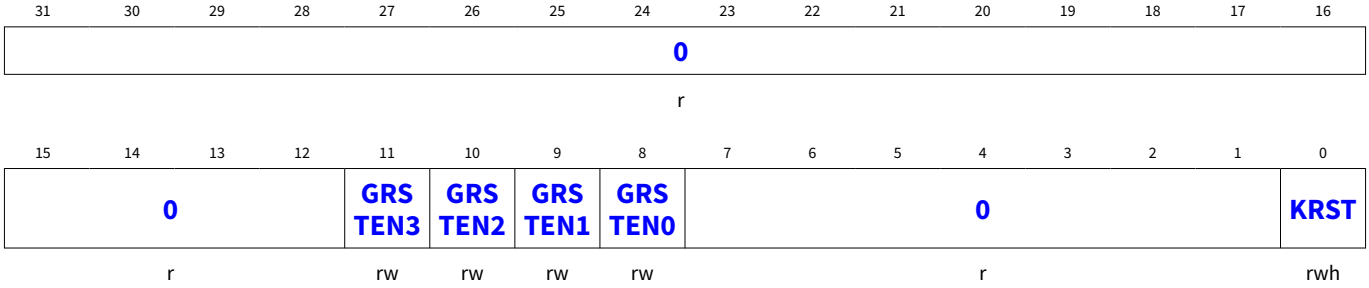


Field	Bits	Type	Description
MOD_REV	7:0	r	**Module Revision** Indicates the revision number of the implementation.
MOD_TYPE	15:8	r	**Module Type** This internal marker is fixed to $C0_H$.
MOD_NUM	31:16	r	**Module Number** Indicates the module identification number (0028_H = MSC).

28.4.6 Reset control register A

RST_CTRLA Offset address: $00C_H$

Reset control register A Application Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
KRST	0	rwh	**Kernel Reset** Request a kernel reset. The requested reset is executed if RST_CTRLA.KRST=1 and RST_CTRLB.KRST=1. KRST is cleared after the kernel reset was executed. 0_B No action 1_B A kernel reset was requested
GRSTENx (x=0-3)	x+8	rw	**Enable for Global Module Reset Group x** 0_B Global module reset group x does not have any effect 1_B Global module reset group x results in a kernel reset
0	7:1, 31:12	r	**Reserved** Read as 0; should be written with 0.

28.4.7 Reset control register B

RST_CTRLB Offset address: 010_H

Reset control register B Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
STATCLR								0							
w								r							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															KRST
r															rwh

Field	Bits	Type	Description
KRST	0	rwh	**Kernel Reset** Request a kernel reset. The requested reset is executed if RST_CTRLA.KRST=1 and RST_CTRLB.KRST=1. KRST is cleared after the kernel reset was executed. 0_B No action 1_B A kernel reset was requested
STATCLR	31	w	**Kernel Reset Status Clear** Clears all status bits in RST_STAT when 1 is written. Read will return 0. 0_B No action 1_B Write with ´1´ clears bits RST_STAT.GRSTx and bit RST_STAT.KRST .
0	30:1	r	**Reserved** Read as 0; should be written with 0.

28.4.8 Reset status register

The reset status register contains the status bits for kernel reset (KRST) and the global module reset groups (GRSTx).

RST_STAT Offset address: 014_H

Reset status register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				GRST3	GRST2	GRST1	GRST0	0							KRST
r				rh	rh	rh	rh	r							rh

Field	Bits	Type	Description
KRST	0	rh	**Kernel Reset Status** Indicates an executed kernel reset. RST_STAT.KRST is set after the execution of a kernel reset in the same clock cycle in which the reset bits are cleared. Clear KRST by setting bit STATCLR in register RST_CTRLB.
GRSTx (x=0-3)	x+8	rh	**Status for Global Module Reset Group x** 0_B Reset was not triggered by Global Reset Group x 1_B Reset was triggered by Global Reset Group x
0	7:1, 31:12	r	**Reserved** Read as 0.

28.4.9 PROT register endinit

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner and in some PROT states, the secure master to update the PROT state as described in the PROT mechanism.

The PROTE register controls lock / unlock of the local Endinit (E) protected control registers.

PROTE Offset address: 018_H

PROT register endinit Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OWEN	ODEF	TAGID						PRSEN	PRS			VMEN	VM		
w	rw	rw						rw	rw			rw	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												SWEN	STATE		
r												w	rwh		

Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
VM	18:16	rw	**Virtual machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual machine definition enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.
PRSEN	23	rw	**Protection set definition enable for PROT owner** 0_B **Disable**: PRS is not part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG-ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; should be written with 0.

Table 1219 Access mode restrictions of PROTE sorted by descending priority

Mode name	Access mode		Description
write 1 to .**SWEN**	rwh	STATE	
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

28.4.10 PROT register safe endinit

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner and in some PROT states, the secure master to update the PROT state as described in the PROT mechanism.

The PROTSE register controls lock / unlock of the local Safe Endinit (SE) protected control registers.

PROTSE Offset address: $01C_H$

PROT register safe endinit Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OWEN	**ODEF**	**TAGID**						**PRSEN**	**PRS**			**VMEN**	**VM**		
w	rw	rw						rw	rw			rw	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												**SWEN**	**STATE**		
r												w	rwh		

Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write

(table continues...)

(continued)

Field	Bits	Type	Description
VM	18:16	rw	**Virtual machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual machine definition enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.
PRSEN	23	rw	**Protection set definition enable for PROT owner** 0_B **Disable**: PRS is not part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG-ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; should be written with 0.

Table 1220 Access mode restrictions of PROTSE sorted by descending priority

Mode name	Access mode		Description
write 1 to .**SWEN**	rwh	STATE	
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

28.4.11 Write access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

ACCEN_WRA Offset address: 020_H

Write access enable register A Application Reset value: $1000\ 0003_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access

28.4.12 Write access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

ACCEN_WRB Offset address: 024_H

Write access enable register B Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access
0	31:8	r	**Reserved** Read as 0; should be written with 0.

28.4.13 Read access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

ACCEN_RDA Offset address: 028_H

Read access enable register A Application Reset value: $FFFF\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access

28.4.14 Read access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

ACCEN_RDB Offset address: $02C_H$

Read access enable register B Application Reset value: $0000\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access
0	31:8	r	**Reserved** Read as 0; should be written with 0.

28.4.15 VM access enable register

This register defines which virtual machine encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each VMx.

ACCEN_VM Offset address: 030_H

VM access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for VM ID q** This bit enables read access to the access protected region for transactions with the VM ID q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for VM ID q** This bit enables write access to the access protected region for transactions with the VM ID q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

28.4.16 PRS access enable register

This register defines which protection register set encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each PRSx.

ACCEN_PRS Offset address: 034_H

PRS access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for PRS q** This bit enables read access to the access protected region for transactions with the PRS q. 0_B Disabled for read access 1_B Enabled for read access

(table continues...)

(continued)

Field	Bits	Type	Description
WRq (q=00-07)	q+16	rw	**Write access enable for PRS q** This bit enables write access to the access protected region for transactions with the PRS q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

28.4.17 Fractional divider register

The Fractional Divider Register controls the clock rate of the shift clock f_{MSC} .

FDR Offset address: 040_H

Fractional divider register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DISCLK			0			RESULT									
rwh			r			rh									

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DM		0			ACS	STEP									
rw		r			rw	rw									

Field	Bits	Type	Description
STEP	9:0	rw	**Step Value** Reload or addition value for RESULT.
ACS	10	rw	**Alternate Clock Source** ACS selects the clock source for the MSC Output. 0_B Source clock for MSC Output f_{SPB}. 1_B Source clock for MSC Output f_{SRI}.
DM	15:14	rw	**Divider Mode** DM selects normal or fractional divider mode. 00_B Fractional divider is switched off; no output clock is generated. The Reset External Divider signal is 1. RESULT is not updated (default after System Reset). 01_B Normal divider mode 10_B Fractional divider mode 11_B Fractional divider is switched off; no output clock is generated. RESULT is not updated.
RESULT	25:16	rh	**Result Value** Bit field for the addition result.

(table continues...)

(continued)

Field	Bits	Type	Description
DISCLK	31	rwh	**Disable Clock** Hardware controlled disable for f_{MSC} signal. 0_B Enable 1_B Disable
0	13:11, 30:26	r	**Reserved** Read as 0; should be written with 0.

28.4.18 Upstream status register

The Upstream Status Register is used to configure the upstream channel data format, baud rate and parity type. It also provides the status information of the upstream counter (UC).

USR Offset address: 044_H

Upstream status register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0											UC				
r											rh				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0										SRDC	PCTR	URR			UFT
r										rw	rw	rw			rw

Field	Bits	Type	Description
UFT	0	rw	**Upstream Channel Frame Type** This bit determines the frame type used by the upstream channel for data reception. 0_B 12-bit upstream frame selected 1_B 16-bit upstream frame selected (with 4-bit address field)
URR	3:1	rw	**Upstream Channel Receiving Rate** This bit field determines the baud rate for the upstream channel. 000_B Upstream channel disabled; no reception is possible 001_B Baud rate = f_{MSC} /4 010_B Baud rate = f_{MSC} /8 011_B Baud rate = f_{MSC} /16 100_B Baud rate = f_{MSC} /32 101_B Baud rate = f_{MSC} /64 110_B Baud rate = f_{MSC} /128 111_B Baud rate = f_{MSC} /256

(table continues...)

(continued)

Field	Bits	Type	Description
PCTR	4	rw	**Parity Control** This bit determines the parity mode used by the upstream channel for data reception. 0_B Even parity mode is selected. A parity bit is set on an odd number of 1s in the serial address/data stream. 1_B Odd parity mode is selected. A parity bit is set on an even number of 1s in the serial address/data stream.
SRDC	5	rw	**Service Request Delay Control** This bit determines the additional delay inserted by the upstream channel when triggering the receive interrupt. Only the hardware generated interrupt can be delayed. The software generated interrupt by writing the ISC.SURDI bit is never delayed. 0_B No delay 1_B Delay of 1 bit time
UC	20:16	rh	**Upstream Counter** This bit field indicates the content of the upstream counter that counts the bits during upstream channel reception.
0	15:6, 31:21	r	**Reserved** Read as 0; should be written with 0.

28.4.19 Downstream control register

The Downstream Control Register is used to control the operation mode and frame layout of the downstream channel transmission. It also contains the two pending status bits.

The “rw” bits in the DSC register are buffered in a shadow buffer at the start of a corresponding frame transmission.

If the asynchronous mode (ABC.ABB=1) is enabled and no data-bit (zero data-bits) in a data frame is configured (DSC.NDBL=DSC.NDBH=00000), then the first select-bit must be set to active (DSC.ENSELL=1).

DSC Offset address: 048_H

Downstream control register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		ALTSEL	PPD					0		NBC					
r		rw	rw					r		rw					

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DSDIS	ENSELH	ENSELL	NDBH					NDBL					DP	CP	TM
rh	rw	rw	rw					rw					rh	rh	rw

Field	Bits	Type	Description
TM	0	rw	**Transmission Mode** This bit selects the transmission mode of the downstream channel. 0_B Triggered Mode selected 1_B Data Repetition Mode selected
CP	1	rh	**Command Pending** This bit is set when the downstream command register DC is written. CP is cleared when the first bit of the related command frame is sent out.
DP	2	rh	**Data Pending** In Triggered Mode, this bit is set when the set data pending bit ISC.SDP is set by software. In Data Repetition Mode, this bit is set by hardware at the last passive time frame. At the start of the data frame, DP is cleared by hardware.
NDBL	7:3	rw	**Number of SRL Bits Shifted at data frames** NDBL determines the number of shift register low part (SRL) bits that are shifted out on SO during a data frame. Other bit combinations are reserved; do not use these bit combinations. 00_H No SRL bit shifted 01_H SRL[0:0] shifted … 10_H SRL[15:0] shifted **others,** reserved
NDBH	12:8	rw	**Number of SRH Bits Shifted at data frames** NDBH determines the number of shift register high part (SRH) bits that are shifted out on SO during a data frame. Other bit combinations are reserved; do not use these bit combinations. 00_H No SRH bit shifted; no selection bit is generated, the SRH active phase is completely skipped. 01_H SRH[0:0] shifted … 10_H SRH[15:0] shifted **others,** reserved
ENSELL	13	rw	**Enable SRL Active Phase Selection Bit** This bit determines whether a low level selection bit is inserted at the beginning of a data frame's SRL active phase. 0_B No selection bit inserted. 1_B Low level selection bit inserted.
ENSELH	14	rw	**Enable SRH Active Phase Selection Bit** This bit determines whether a low level selection bit is inserted at the beginning of a data frame's SRH active phase. 0_B No selection bit inserted. 1_B Low level selection bit inserted.

(table continues...)

(continued)

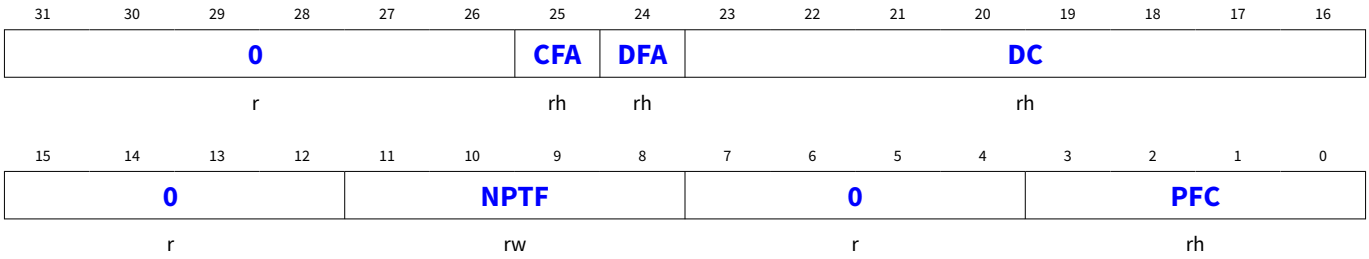
Field	Bits	Type	Description
DSDIS	15	rh	**Downstream Disable** This bit indicates the state of the downstream channel operation. 0_B The downstream channel is enabled. A frame transmission can take place (Triggered Mode) or takes place (Data Repetition Mode). 1_B Downstream Counter becomes disabled. No new frame transmission is started. A running frame transmission is always completed.
NBC	21:16	rw	**Number of Bits Shifted at command frames** This bit field determines how many bits of the SRL/SRH shift registers are shifted out during transmission of a command frame. 00_H No bit shifted 01_H SRL[0:0] shifted … 10_H SRL[15:0] shifted 11_H SRL[15:0] and SRH[0:0] shifted … 20_H SRL[15:0] and SRH[15:0] shifted **others,** reserved
PPD	28:24	rw	**Passive Phase Length at data frames** This bit field determines the length of the passive phase of a data frame. 00_H Passive phase length is 2 x t_FCL for value 0 … 02_H Passive phase length is 2 x t_FCL for value 2 03_H Passive phase length is 3 x t_FCL … $1F_H$ Passive phase length is 31 x t_FCL
ALTSEL	29	rw	**Alternative Input Timer Selection** This bit-field selects the input source, GTM or eGTM, for the ALTINL, ALTINH, ALTINLE and ALTINHE. 0_B GTM input selected. 1_B eGTM input selected.
0	23:22, 31:30	r	**Reserved** Read as 0; should be written with 0.

28.4.20 Downstream status register

The Downstream Status Register DSS contains counter bit fields, status bits and indicates the number of passive time frames.

DSS Offset address: $04C_H$

Downstream status register Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
PFC	3:0	rh	**Passive Time Frame Counter** In Data Repetition Mode, this bit field indicates the count of passive time frames that are currently transmitted. In Triggered Mode PFC remains at 0000_B. 0_H Data frame is transmitted. 1_H 1st passive time frame is transmitted. 2_H 2th passive time frame is transmitted. … F_H 15th passive time frame is transmitted.
NPTF	11:8	rw	**Number Of Passive Time Frames** This bit field indicates the number of passive time frames that are inserted in Data Repetition Mode between two data frames. 0_H No passive time frame inserted. 1_H 1 passive time frame inserted. 2_H 2 passive time frames inserted. … F_H 15 passive time frames inserted.
DC	23:16	rh	**Downstream Counter** This bit field indicates the number of downstream shift clock periods that have been elapsed since the start of the current frame. DC is reset at the end of a downstream frame. 00_H No shift clock elapsed (after counter reset). 01_H 1 shift clock elapsed. 02_H 2 shift clocks elapsed. … FF_H 255 shift clocks elapsed.
DFA	24	rh	**Data Frame Active** This bit indicates if a data frame is currently sent out, active phase only. 0_B No data frame is currently sent out. 1_B A data frame is currently sent out.
CFA	25	rh	**Command Frame Active** This bit indicates if a command frame is currently sent out, active phase only. 0_B No command frame is currently sent out. 1_B A command frame is currently sent out.

(table continues...)

(continued)

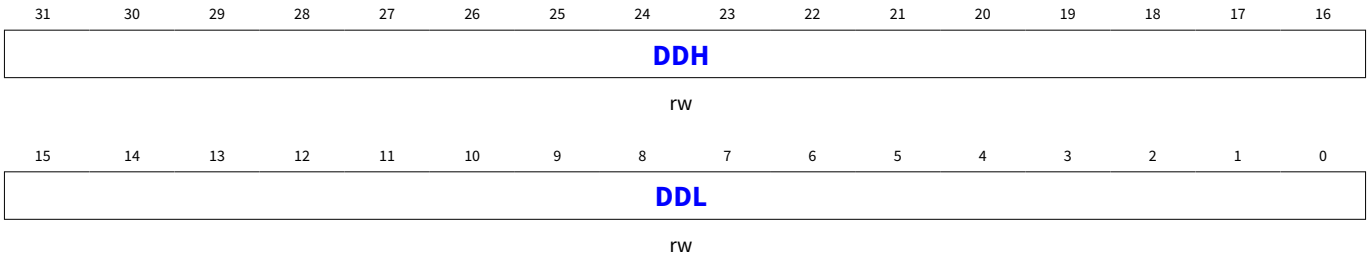
Field	Bits	Type	Description
0	7:4, 15:12, 31:26	r	**Reserved** Read as 0; shall be written with 0.

28.4.21 Downstream data register

The Downstream Data Register DD contains data to be transmitted during data frames.

DD Offset address: 050_H

Downstream data register Kernel Reset value: $0000\ 0000_H$



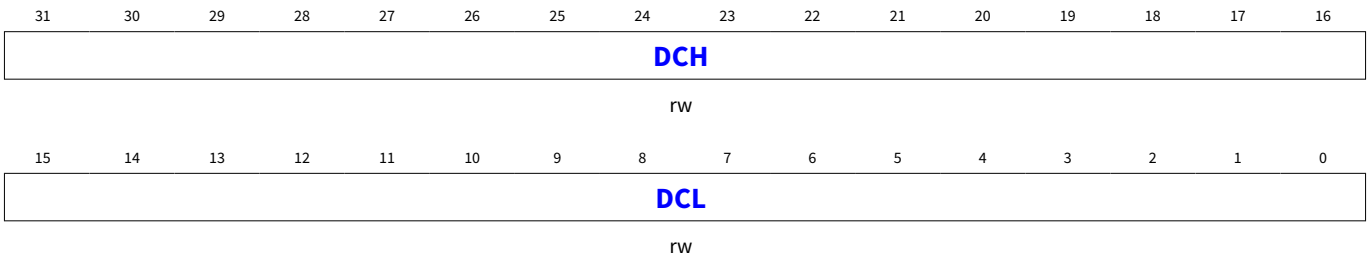
Field	Bits	Type	Description
DDL	15:0	rw	**Downstream Data for SRL Shift Register** Contains the data bits to be transmitted during the SRL active phase of a data frame.
DDH	31:16	rw	**Downstream Data for SRH Shift Register** Contains the data bits to be transmitted during the SRH active phase of a data frame.

28.4.22 Downstream command register

The Downstream Command Register DC contains command information to be transmitted during command frames. Every write access to the DC register triggers a transmission of a command frame.

DC Offset address: 054_H

Downstream command register Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
DCL	15:0	rw	**Downstream Command for SRL Shift Register** Contains the data bits to be transmitted during the SRL active phase of a command frame.
DCH	31:16	rw	**Downstream Command for SRH Shift Register** Contains the data bits to be transmitted during the SRH active phase of a command frame.

28.4.23 Downstream select data source low register

The bit fields of the Downstream Select Data Source Low Register DSDSL determine the data source for each bit in shift register SRL.

DSDSL Offset address: 058_H

Downstream select data source low register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SL15		SL14		SL13		SL12		SL11		SL10		SL9		SL8	
rw		rw		rw		rw		rw		rw		rw		rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SL7		SL6		SL5		SL4		SL3		SL2		SL1		SL0	
rw		rw		rw		rw		rw		rw		rw		rw	

Field	Bits	Type	Description
SLx (x=0-15)	2*x+1:2*x	rw	**Select Source for SRL** SLx determines which data source is used for the shift register bit SRL[x] for data frame transmission. 00_B SRL[x] is taken from data register DD.DDL[x]. 01_B Reserved. 10_B SRL[x] is taken from the ALTINL input line x. 11_B SRL[x] is taken from the ALTINL input line x in inverted state.

28.4.24 Downstream select data source high register

The bit fields of the Downstream Select Data Source High Register DSDSH determine the data source for each bit in shift register SRH.

DSDSH Offset address: $05C_H$

Downstream select data source high register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SH15		SH14		SH13		SH12		SH11		SH10		SH9		SH8	
rw		rw		rw		rw		rw		rw		rw		rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SH7		SH6		SH5		SH4		SH3		SH2		SH1		SH0	
rw		rw		rw		rw		rw		rw		rw		rw	

Field	Bits	Type	Description
SHx (x=0-15)	2*x+1:2*x	rw	**Select Source for SRH** SHx determines which data source is used for the shift register bit SRH[x] during data frame transmission. 00_B SRH[x] is taken from data register DD.DDH[x]. 01_B Reserved. 10_B SRH[x] is taken from the ALTINH input line x. 11_B SRH[x] is taken from the ALTINH input line x in inverted state.

28.4.25 Emergency stop register

The Emergency Stop Register ESR determines which bits of SRL and SRH are enabled for emergency operation.

ESR Offset address: 060_H

Emergency stop register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ENH15	ENH14	ENH13	ENH12	ENH11	ENH10	ENH9	ENH8	ENH7	ENH6	ENH5	ENH4	ENH3	ENH2	ENH1	ENH0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ENL15	ENL14	ENL13	ENL12	ENL11	ENL10	ENL9	ENL8	ENL7	ENL6	ENL5	ENL4	ENL3	ENL2	ENL1	ENL0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

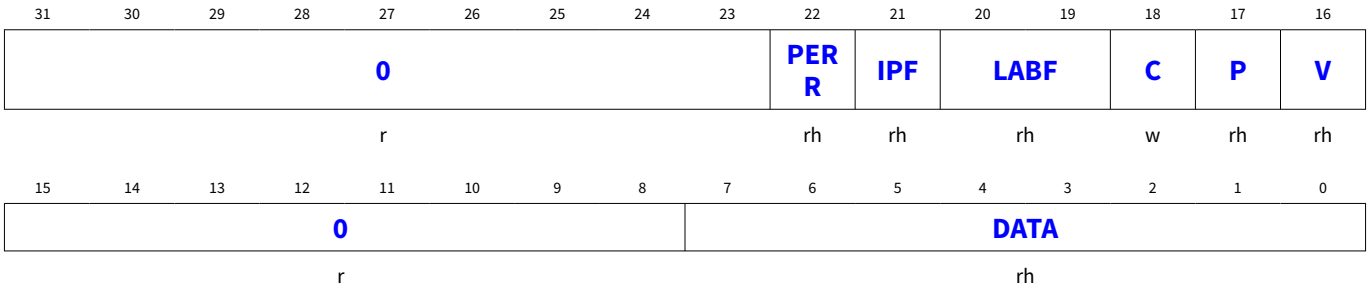
Field	Bits	Type	Description
ENLx (x=0-15)	x	rw	**Emergency Stop Enable for Bit x in SRL** This bit enables the emergency stop feature selectively for each SRL bit. If the emergency stop condition is met and enabled (ENLx = 1), the SRL[x] bit of the data register DD.DDL[x] is used for the shift register load operation. 0_B Emergency stop feature for bit SRL[x] is disabled. 1_B The emergency stop feature for bit SRL[x] is enabled.
ENHx (x=0-15)	x+16	rw	**Emergency Stop Enable for Bit x in SRH** This bit enables the emergency stop feature selectively for each SRH bit. If the emergency stop condition is met and enabled (ENHx = 1), the SRH[x] bit of the data register DD.DDH[x] is used for the shift register load operation. 0_B Emergency stop feature for bit SRH[x] is disabled. 1_B The emergency stop feature for bit SRH[x] is enabled.

28.4.26 Upstream data register x

The four Upstream Data Registers UDx store the content (data, addresses, received and calculated parity bit, parity error bit) of a received upstream channel data frame.

UDx (x=0-3) Offset address: 070_H+x*4

Upstream data register x Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
DATA	7:0	rh	**Received Data** This bit field contains the 8-bit receive data.
V	16	rh	**Valid Bit** This bit is set by hardware when the received data is written to UDx. Writing bit C = 1 clears V. If hardware setting and software clearing of the valid bit occur simultaneously, bit V will be cleared.
P	17	rh	**Parity Bit** This flag contains the parity bit that has been received with the data frame.
C	18	w	**Clear Bit** C is always read as 0. 0_B No operation. 1_B Bit V is cleared.
LABF	20:19	rh	**Lower Address Bit Field** This bit field contains the two address bits A[1:0] of the 4-bit address field (16-bit data frame). If 12-bit data frame is selected, LABF is always set to 00_B.
IPF	21	rh	**Internal Parity Flag** This bit contains the parity bit that has been calculated in the MSC during data frame reception.
PERR	22	rh	**Error bit- PERR** This bit indicates if a start bit error, parity error or stop bit error occurred during frame reception. 0_B No error detected. 1_B Error detected.
0	15:8, 31:23	r	**Reserved** Read as 0; should be written with 0.

28.4.27 Interrupt control register

The Interrupt Control Register ICR holds the interrupt enable bits and interrupt pointers of all seven MSC interrupts.

ICR Offset address: 080_H

Interrupt control register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0				TFMIE	0	TFMIP		CSYIE	0	CSYIP		CSXIE	0	CSXIP	
r				rw	r	rw		rw	r	rw		rw	r	rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RDIE		RDIP		TFIE	0	TFIP		ECIE	0	ECIP		EDIE		EDIP	
rw		rw		rw	r	rw		rw	r	rw		rw		rw	

Field	Bits	Type	Description
EDIP	1:0	rw	**Data Frame Interrupt Node Pointer** EDIP selects the service request output line SRn (n = 0-3) for the data frame interrupt. 00_B Service request output SR0 selected 01_B Service request output SR1 selected 10_B Service request output SR2 selected 11_B Service request output SR3 selected
EDIE	3:2	rw	**Data Frame Interrupt Enable** This bit field determines the enable conditions for the data frame interrupt. 00_B Interrupt generation disabled 01_B An interrupt is generated when the last data bit has been shifted out. 10_B An interrupt is generated when the first data bit has been shifted out, but only if DSC.NDBL is not equal 00000_B. This means, at least one SRL bit must be shifted out for the first data bit shifted interrupt to become active. 11_B Interrupt generation disabled
ECIP	5:4	rw	**Command Frame Interrupt Node Pointer** ECIP selects the service request output line SRn (n = 0-3) for the command frame interrupt. 00_B Service request output SR0 selected 01_B Service request output SR1 selected 10_B Service request output SR2 selected 11_B Service request output SR3 selected
ECIE	7	rw	**Command Frame Interrupt Enable** This bit enables the command frame interrupt. 0_B Interrupt generation disabled. 1_B Interrupt generation enabled.
TFIP	9:8	rw	**Time Frame Interrupt Pointer** TFIP selects the service request output line SRn (n = 0-3) for the time frame interrupt. 00_B Service request output SR0 selected 01_B Service request output SR1 selected 10_B Service request output SR2 selected 11_B Service request output SR3 selected

(table continues...)

(continued)

Field	Bits	Type	Description
TFIE	11	rw	**Time Frame Interrupt Enable** This bit enables the time frame interrupt. 0_B Interrupt generation disabled. 1_B Interrupt generation enabled.
RDIP	13:12	rw	**Receive Data Interrupt Pointer** RDIP selects the service request output line SRn (n = 0-3) for the receive data interrupt. 00_B Service request output SR0 selected 01_B Service request output SR1 selected 10_B Service request output SR2 selected 11_B Service request output SR3 selected
RDIE	15:14	rw	**Receive Data Interrupt Enable** This bit field determines the enable conditions for the receive data interrupt. 00_B Interrupt generation disabled. 01_B An interrupt is generated when data is received and written into the upstream data registers UDx (x = 0-3). 10_B An interrupt is generated as with RDIE = 01_B but only if the received data is not equal to 00_H. 11_B An interrupt is generated when data is received and written into register UD3.
CSXIP	17:16	rw	**Command Frame Slow Channel X Interrupt Node Pointer** CSXIP selects the service request output line SRn (n = 0-3) for the command frame interrupt. 00_B Service request output SR0 selected 01_B Service request output SR1 selected 10_B Service request output SR2 selected 11_B Service request output SR3 selected
CSXIE	19	rw	**Command Frame Slow Channel X Interrupt Enable** This bit enables the command frame interrupt on Slow Channel mode for Slave X. 0_B Interrupt generation disabled. 1_B Interrupt generation enabled.
CSYIP	21:20	rw	**Command Frame Slow Channel Y Interrupt Node Pointer** CSYIP selects the service request output line SRn (n = 0-3) for the command frame interrupt. 00_B Service request output SR0 selected 01_B Service request output SR1 selected 10_B Service request output SR2 selected 11_B Service request output SR3 selected

(table continues...)

(continued)

Field	Bits	Type	Description
CSYIE	23	rw	**Command Frame Slow Channel Y Interrupt Enable** This bit enables the command frame interrupt on Slow Channel mode for Slave Y. 0_B Interrupt generation disabled. 1_B Interrupt generation enabled.
TFMIP	25:24	rw	**Time Frame Manchester Code Interrupt Pointer** TFMIP selects the service request output line SRn (n = 0-3) for the command frame interrupt. 00_B Service request output SR0 selected 01_B Service request output SR1 selected 10_B Service request output SR2 selected 11_B Service request output SR3 selected
TFMIE	27	rw	**Time Frame Manchester Code Interrupt Enable** This bit enables the Time Frame interrupt in Manchester Code mode. 0_B Interrupt generation disabled. 1_B Interrupt generation enabled.
0	6, 10, 18, 22, 26, 31:28	r	**Reserved** Read as 0; should be written with 0.

28.4.28 Interrupt status register

The Interrupt Status Register ISR holds the interrupt status flags that indicate an interrupt occurrence in downstream and upstream channels.

ISR Offset address: 084_H

Interrupt status register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0									TFMI	CSYI	CSXI	URDI	DTFI	DECI	DEDI
r									rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
DEDI	0	rh	**Data Frame Interrupt Flag** This flag is always set by hardware when a downstream channel data frame interrupt is generated. DEDI can be set or cleared by software when writing to register ISC with the appropriate bits ISC.SDEDI or ISC.CDEDI set.

(table continues...)

(continued)

Field	Bits	Type	Description
DECI	1	rh	**Command Frame Interrupt Flag** This flag is always set by hardware when a downstream channel command frame interrupt is generated, whether or not it is enabled. DECI can be set or cleared by software when writing to register ISC with the appropriate bits SDECI or CDECI set.
DTFI	2	rh	**Time Frame Interrupt Flag** This flag is always set by hardware when a downstream channel time frame interrupt is generated, whether or not it is enabled. DTFI can be set or cleared by software when writing to register ISC with the appropriate bits SDTFI or CDTFI set.
URDI	3	rh	**Receive Data Interrupt Flag** This flag is always set by hardware when an upstream channel receive data interrupt is generated, whether or not it is enabled. URDI can be set or cleared by software when writing to register ISC with the appropriate bits SURDI or CURDI set.
CSXI	4	rh	**Command Frame Slow Channel X Interrupt Flag** This flag is always set by hardware when a downstream channel command frame interrupt in Slow Channel mode is generated. CSXI can be set or cleared by software when writing to register ISC with the appropriate bits ISC.SCSXI or ISC.CCSXI set.
CSYI	5	rh	**Command Frame Slow Channel Y Interrupt Flag** This flag is always set by hardware when a downstream channel command frame interrupt in Slow Channel mode is generated. CSYI can be set or cleared by software when writing to register ISC with the appropriate bits ISC.SCSYI or ISC.CCSYI set.
TFMI	6	rh	**Time Frame Manchester Code Interrupt Flag** This flag is always set by hardware when a downstream channel command frame interrupt in Slow Channel mode is generated. TFMI can be set or cleared by software when writing to register ISC with the appropriate bits ISC.STFMI or ISC.CTFMI set.
0	31:7	r	**Reserved** Read as 0.

28.4.29 Interrupt set clear register

The Interrupt Set Clear Register ISC is used to set or clear the MSC interrupt flags located in the Interrupt Status Register ISR. Reading ISC always returns 0000 0000_H.

***Note**: When the ISC register is written with both bits (set and clear bit) for a specific interrupt flag, the clear operation takes place and the set operation is ignored.*

ISC Offset address: 088_H

Interrupt set clear register Kernel Reset value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0						STFMI	SCSYI	SCSXI	SDDIS	SCP	SDP	SURDI	SDTFI	SDECI	SDEDI
r						w	w	w	w	w	w	w	w	w	w

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0						CTFMI	CCSYI	CCSXI	CDDIS	CCP	CDP	CURDI	CDTFI	CDECI	CDEDI
r						w	w	w	w	w	w	w	w	w	w

Field	Bits	Type	Description
CDEDI	0	w	**Clear DEDI Flag** 0_B No operation 1_B Bit ISR.DEDI is cleared.
CDECI	1	w	**Clear DECI Flag** 0_B No operation 1_B Bit ISR.DECI is cleared.
CDTFI	2	w	**Clear DTFI Flag** 0_B No operation 1_B Bit ISR.DTFI is cleared.
CURDI	3	w	**Clear URDI Flag** 0_B No operation 1_B Bit ISR.URDI is cleared.
CDP	4	w	**Clear DP Flag** 0_B No operation 1_B Bit DSC.DP is cleared.
CCP	5	w	**Clear CP Flag** 0_B No operation 1_B Bit DSC.CP is cleared.
CDDIS	6	w	**Clear DSDIS Flag** 0_B No operation 1_B Bit DSC.DSDIS is cleared.
CCSXI	7	w	**Clear CSXI Flag** 0_B No operation 1_B Bit ISR.CSXI is cleared.
CCSYI	8	w	**Clear CSYI Flag** 0_B No operation 1_B Bit ISR.CSYI is cleared.
CTFMI	9	w	**Clear TFMIFlag** 0_B No operation 1_B Bit ISR.TFMI is cleared.

(table continues...)

(continued)

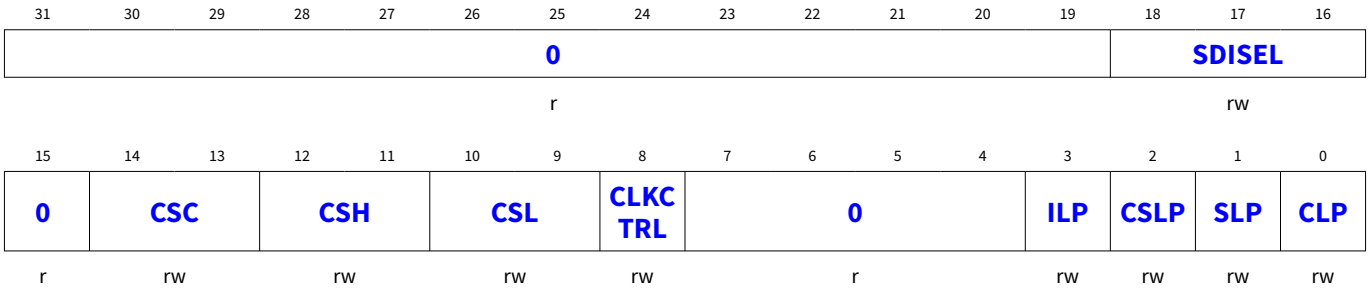
Field	Bits	Type	Description
SDEDI	16	w	**Set DEDI Flag** 0_B No operation 1_B Bit ISR.DEDI is set.
SDECI	17	w	**Set DECI Flag** 0_B No operation 1_B Bit ISR.DECI is set.
SDTFI	18	w	**Set DTFI Flag** 0_B No operation 1_B Bit ISR.DTFI is set.
SURDI	19	w	**Set URDI Flag** 0_B No operation 1_B Bit ISR.URDI is set.
SDP	20	w	**Set DP Bit** 0_B No effect 1_B Bit DSC.DP is set.
SCP	21	w	**Set CP Flag** 0_B No operation 1_B Bit DSC.CP is set.
SDDIS	22	w	**Set DSDIS Flag** 0_B No operation 1_B Bit DSC.DSDIS is set.
SCSXI	23	w	**Set CSXI Flag** 0_B No operation 1_B Bit ISR.CSXI is set.
SCSYI	24	w	**Set CSYI Flag** 0_B No operation 1_B Bit ISR.CSYI is set.
STFMI	25	w	**Set TFMIFlag** 0_B No operation 1_B Bit ISR.TFMI is set.
0	15:10, 31:26	r	**Reserved** Read as 0; should be written with 0.

28.4.30 Output control register

The Output Control Register OCR determines the MSC input/output signal polarities, the chip select output signal assignment and the serial output clock generation.

OCR Offset address: $08C_H$

Output control register Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
CLP	0	rw	**FCLP Line Polarity** 0_B FCLP and FCL signal polarity is identical. FCLN signal has inverted FCL signal polarity. 1_B FCLP signal has inverted FCL signal polarity. FCLN and FCL signal polarities are identical.
SLP	1	rw	**SOP Line Polarity** This bit determines the polarity configuration of SON, SOP and SO. Also determines the Manchester encoding method to be used. 0_B SOP and SO signal polarity is identical. SON signal has inverted SO signal polarity. G.E. Thomas manchester encoding is used. 1_B SOP signal has inverted SO signal polarity. SON and SO signal polarities are identical. IEEE enconding is used.
CSLP	2	rw	**Chip Selection Lines Polarity** Bit CSLP is buffered during a frame transmission. This means that any change of CSLP becomes valid first with the start of the next frame transmission. 0_B EN[3:0] and ENL, ENH, ENC signal polarities are identical (high active). 1_B EN[3:0] signal polarities are inverted (low active) to the ENL, ENH, ENC signal polarities.
ILP	3	rw	**SDI Line Polarity** 0_B SDI and SI signal polarities are identical. 1_B SDI and SI signal polarities are inverted.
CLKCTRL	8	rw	**Clock Control** This bit determines the activation of clock output FCL. 0_B FCL is activated only during the active phases of data or command frames (not during passive time frames). 1_B FCL is always active whether or not a downstream frame is currently transmitted.

(table continues...)

(continued)

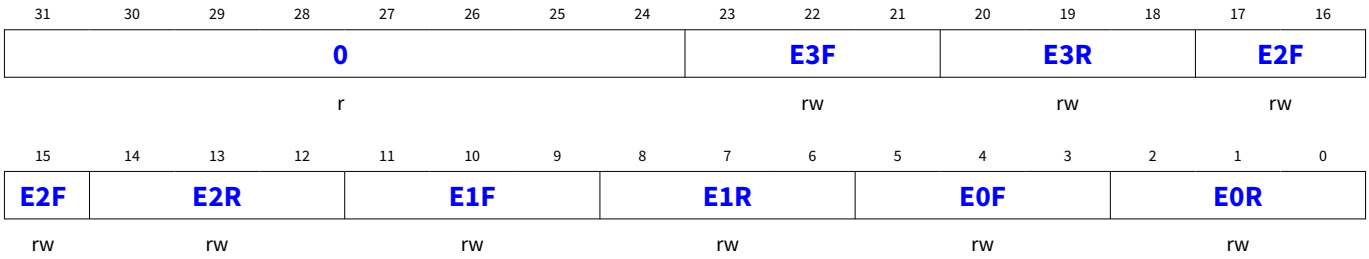
Field	Bits	Type	Description
CSL	10:9	rw	**Chip Enable Selection for ENL** This bit field selects the chip enable output ENx that becomes active during the SRL active phase (ENL = 1) of a data frame. The active level of ENx is defined by bit CSLP. 00_B EN0 line is selected for ENL. 01_B EN1 line is selected for ENL. 10_B EN2 line is selected for ENL. 11_B EN3 line is selected for ENL.
CSH	12:11	rw	**Chip Enable Selection for ENH** This bit field selects the chip enable output ENx that becomes active during the SRH active phase (ENH = 1) of a data frame. The active level of ENx is defined by bit CSLP. 00_B EN0 line is selected for ENH. 01_B EN1 line is selected for ENH. 10_B EN2 line is selected for ENH. 11_B EN3 line is selected for ENH.
CSC	14:13	rw	**Chip Enable Selection for ENC** This bit field selects the chip enable output ENx that becomes active during the active phase (ENC = 1) of a command frame. The active level of ENx is defined by bit CSLP. 00_B EN0 line is selected for ENC. 01_B EN1 line is selected for ENC. 10_B EN2 line is selected for ENC. 11_B EN3 line is selected for ENC.
SDISEL	18:16	rw	**Serial Data Input Selection** This bit field selects the source for the serial data input SDI of the upstream channel. 000_B SDI0 input is selected for SDI. 001_B SDI1 input is selected for SDI. 010_B SDI2 input is selected for SDI. 011_B SDI3 input is selected for SDI. 100_B SDI4 input is selected for SDI. 101_B SDI5 input is selected for SDI. 110_B SDI6 input is selected for SDI. 111_B SDI7 input is selected for SDI.
0	7:4, 15, 31:19	r	**Reserved** Read as 0; should be written with 0.

28.4.31 Output extended control register

The Output Extended Control Register OCER determines the MSC enable output signal adjustment.

OECR Offset address: 090_H

Output extended control register Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
E0R	2:0	rw	**Downstream Enable 1/2 Clock Time Adjustment Assertion (Rising Edge, EN0)** This bit field selects the configuration adjustment of 0, 1 or 2 steps of ½ clock time-period at assertion. 000_B 0 Step of ½ clock period before edge adjustment. 001_B 1 Step of ½ clock period before edge adjustment. 010_B 2 Step of ½ clock period before edge adjustment. 011_B 1 Step of ½ clock period after edge adjustment. 100_B 2 Steps of ½ clock period after edge adjustment. **others,** reserved
E0F	5:3	rw	**Downstream Enable 1/2 Clock Time Adjustment Deassertion (Falling Edge,EN0)** This bit field selects the configuration adjustment of 0, 1 or 2 steps of ½ clock time-period at deassertion. 000_B 0 Step of ½ clock period before edge adjustment. 001_B 1 Step of ½ clock period before edge adjustment. 010_B 2 Step of ½ clock period before edge adjustment. 011_B 1 Step of ½ clock period after edge adjustment. 100_B 2 Steps of ½ clock period after edge adjustment. **others,** reserved
E1R	8:6	rw	**Downstream Enable 1/2 Clock Time Adjustment Assertion (Rising Edge, EN1)** This bit field selects the configuration adjustment of 0, 1 or 2 steps of ½ clock time-period at assertion. 000_B 0 Step of ½ clock period before edge adjustment. 001_B 1 Step of ½ clock period before edge adjustment. 010_B 2 Step of ½ clock period before edge adjustment. 011_B 1 Step of ½ clock period after edge adjustment. 100_B 2 Steps of ½ clock period after edge adjustment. **others,** reserved

(table continues...)

(continued)

Field	Bits	Type	Description
E1F	11:9	rw	**Downstream Enable 1/2 Clock Time Adjustment Deassertion (Falling Edge,EN1)** This bit field selects the configuration adjustment of 0, 1 or 2 steps of ½ clock time-period at deassertion. 000_B 0 Step of ½ clock period before edge adjustment. 001_B 1 Step of ½ clock period before edge adjustment. 010_B 2 Step of ½ clock period before edge adjustment. 011_B 1 Step of ½ clock period after edge adjustment. 100_B 2 Steps of ½ clock period after edge adjustment. **others,** reserved
E2R	14:12	rw	**Downstream Enable 1/2 Clock Time Adjustment Assertion (Rising Edge, EN2)** This bit field selects the configuration adjustment of 0, 1 or 2 steps of ½ clock time-period at assertion. 000_B 0 Step of ½ clock period before edge adjustment. 001_B 1 Step of ½ clock period before edge adjustment. 010_B 2 Step of ½ clock period before edge adjustment. 011_B 1 Step of ½ clock period after edge adjustment. 100_B 2 Steps of ½ clock period after edge adjustment. **others,** reserved
E2F	17:15	rw	**Downstream Enable 1/2 Clock Time Adjustment Deassertion (Falling Edge,EN2)** This bit field selects the configuration adjustment of 0, 1 or 2 steps of ½ clock time-period at deassertion. 000_B 0 Step of ½ clock period before edge adjustment. 001_B 1 Step of ½ clock period before edge adjustment. 010_B 2 Step of ½ clock period before edge adjustment. 011_B 1 Step of ½ clock period after edge adjustment. 100_B 2 Steps of ½ clock period after edge adjustment. **others,** reserved
E3R	20:18	rw	**Downstream Enable 1/2 Clock Time Adjustment Assertion (Rising Edge, EN3)** This bit field selects the configuration adjustment of 0, 1 or 2 steps of ½ clock time-period at assertion. 000_B 0 Step of ½ clock period before edge adjustment. 001_B 1 Step of ½ clock period before edge adjustment. 010_B 2 Step of ½ clock period before edge adjustment. 011_B 1 Step of ½ clock period after edge adjustment. 100_B 2 Steps of ½ clock period after edge adjustment. **others,** reserved

(table continues...)

(continued)

Field	Bits	Type	Description
E3F	23:21	rw	**Downstream Enable 1/2 Clock Time Adjustment Deassertion (Falling Edge,EN3)** This bit field selects the configuration adjustment of 0, 1 or 2 steps of ½ clock time-period at deassertion. 000_B 0 Step of ½ clock period before edge adjustment. 001_B 1 Step of ½ clock period before edge adjustment. 010_B 2 Step of ½ clock period before edge adjustment. 011_B 1 Step of ½ clock period after edge adjustment. 100_B 2 Steps of ½ clock period after edge adjustment. **others,** reserved
0	31:24	r	**Reserved** Read as 0; should be written with 0.

28.4.32 Slow channel configuration register

The Slow Channel Configuration Register SCC is used to control the Slow Channel mode for the downstream transmission.

SCC Offset address: 094_H

Slow channel configuration register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0				CFSTOPY		CFSTOPX		PCY		PCX		CFLY			
r				rw		rw		rw		rw		rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CFLY				CFLX				SCLY				SCLX		MCT	SC
rw				rw				rw				rw		rw	rw

Field	Bits	Type	Description
SC	0	rw	**Slow Channel** This bit field enables the Slow Channel (embedded command frame into data frame) for 2 slaves. 0_B Slow Channel is disabled. 1_B Slow Channel is enabled for both slaves.
MCT	1	rw	**Manchester Code Transmission MSC+** This bits enables the Manchester Encoding transmission. Note that the Manchester code transmission can only be set in combination with an active Slow Channel. 0_B Manchester code transmission is disabled. 1_B Manchester code transmission is enabled.

(table continues...)

(continued)

Field	Bits	Type	Description
SCLX	4:2	rw	**Slow Channel Length for Slave X** This bit field determines the number of slow channel X bits which precede the data frame. This bit field has no effect if the Slow Channel is not enabled (SCC.SC). 000_B Slow Channel length is 0 bits. 001_B Slow Channel length is 1 bit. 010_B Slow Channel length is 2 bits. 011_B Slow Channel length is 4 bits. 100_B Slow Channel length is 8 bits. 101_B Slow Channel length is 16 bits. 110_B Slow Channel length is 32 bits. **others,** reserved
SCLY	7:5	rw	**Slow Channel Length for Slave Y** This bit field determines the number of slow channel Y bits which precede the data frame. This bit field has no effect if the Slow Channel is not enabled (SCC.SC). 000_B Slow Channel length is 0 bits. 001_B Slow Channel length is 1 bit. 010_B Slow Channel length is 2 bits. 011_B Slow Channel length is 4 bits. 100_B Slow Channel length is 8 bits. 101_B Slow Channel length is 16 bits. 110_B Slow Channel length is 32 bits. **others,** reserved
CFLX	13:8	rw	**Command Frame Length for Slave X** This bit field determines the length of the command frame data (excluding start, parity and stop bits). This bit field has no effect if the Slow Channel is not enabled (SCC.SC). 00_H Slow channel command frame length is 0 bits 01_H Slow channel command frame length is 1 bits … 20_H Slow channel command frame length is 32 bits **others,** reserved
CFLY	19:14	rw	**Command Frame Length for Slave Y** This bit field determines the length of the command frame data (excluding start, parity and stop bits). This bit field has no effect if the Slow Channel is not enabled (SCC.SC). 00_H Slow channel command frame length is 0 bits 01_H Slow channel command frame length is 1 bits … 20_H Slow channel command frame length is 32 bits **others,** reserved

(table continues...)

(continued)

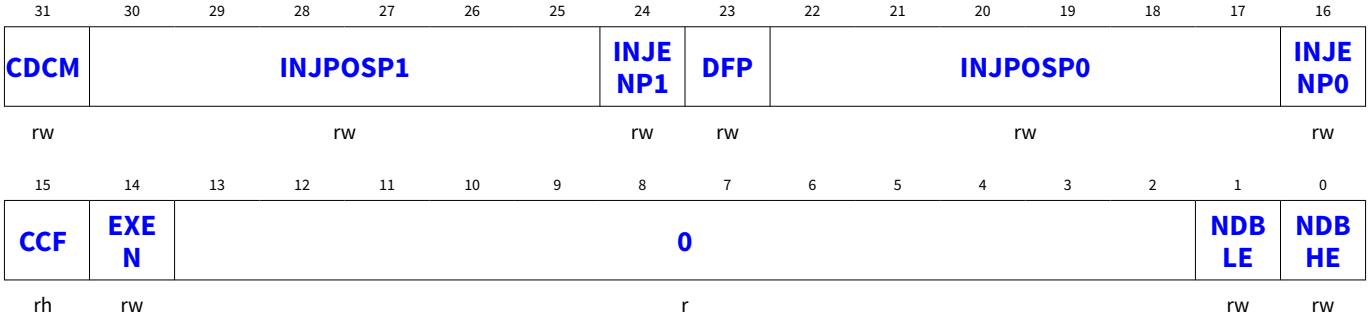
Field	Bits	Type	Description
PCX	21:20	rw	**Paritiy Control for command frame Slave X** This bit field enables the parity for the command frame transmission on Slave X when Slow Channel mode is enabled. This bit field has no effect if the Slow Channel is not enabled (SCC.SC). 00_B Parity OFF. 01_B Even parity ON. 10_B Odd parity ON. **others,** reserved
PCY	23:22	rw	**Paritiy Control for command frame Slave Y** This bit field enables the parity for the command frame transmission on Slave Y when Slow Channel mode is enabled. This bit field has no effect if the Slow Channel is not enabled (SCC.SC). 00_B Parity OFF. 01_B Even parity ON. 10_B Odd parity ON. **others,** reserved
CFSTOPX	25:24	rw	**Command Frame Stop Bits Slave X** This bit field enables and selects the stop bit options for the command frame transmission on Slave X. 00_B no stop bits 01_B 1 stop bit 10_B 2 stop bits **others,** reserved
CFSTOPY	27:26	rw	**Command Frame Stop Bits Slave Y** This bit field enables and selects the stop bit options for the command frame transmission on Slave Y. 00_B no stop bits 01_B 1 stop bit 10_B 2 stop bits **others,** reserved
0	31:28	r	**Reserved** Read as 0; should be written with 0.

28.4.33 Downstream control enhanced register

The Downstream Control Enhanced Register is used to control the enhanced operation mode and frame layout of the downstream channel transmission.

DSCE Offset address: 098_H

Downstream control enhanced register Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
NDBHE	0	rw	**Number of SRH Bits Shifted at data frames Extension** Additional MSB bit extension of the DSC.NDBH bit field. Adds 16 to the resulting DSC.NDBH value if set.
NDBLE	1	rw	**Number of SRL Bits Shifted at data frames Extension** Additional MSB bit extension of the DSC.NDBL bit field. Adds 16 to the resulting DSC.NDBL value if set.
EXEN	14	rw	**Extension Enable** Enables the extension bit fields. 0_B Disabled 1_B Enabled
CCF	15	rh	**Command-Command Flag** This bit flags that a second command frame has been written without data frame to be sent in between. It is active only if CDCM=1. Otherwise it is always low. 0_B no second command 1_B second command frame pending
INJENP0	16	rw	**Injection Enable of the Pin 0 Signal** This bit selects if an external signal is injected in a data frame. 0_B Disabled 1_B Enabled
INJPOSP0	22:17	rw	**Injection Position of the Pin 0 Signal** This bit selects the position of the injected external one bit signal, at the bit position in range of 0 to 63 in the data frame. If both INJPOSP0 and INJPOSP1 point to a same bit, INJPOSP0 has the higher priority, that is only INJ0 level will be visible in the data frame.
DFP	23	rw	**Data Frame over command frame Prioritization** This bits selects if the data frame has priority over command frame, in both modes: Triggered mode and Data Repetition mode. In Manchester Code transmission mode, while using the Slow Channel for transmiting the command frame this bit-filed has no effect. 0_B Data Frame prioritization over command frame disabled. 1_B Data Frame prioritization over command frame enabled.

(table continues...)

(continued)

Field	Bits	Type	Description
INJENP1	24	rw	**Injection Enable of the Pin 1 Signal** This bit selects if an external signal is injected in a data frame. 0_B Disabled 1_B Enabled
INJPOSP1	30:25	rw	**Injection Position of the Pin 1 Signal** This bit selects the position of the injected external one bit signal, at the bit position in range of 0 to 63 in the data frame. If both INJPOSP0 and INJPOSP1 point to a same bit, INJPOSP0 has the higher priority, that is INJ0 level will be visible in the data frame.
CDCM	31	rw	**Command-Data-Command in Data Repetition Mode** This bit selects if a data frame is automatically inserted between two consecutive command frame requests. 0_B Automatic data insertion disabled 1_B Automatic data insertion enabled
0	13:2	r	**Reserved** Read as 0; should be written with 0.

28.4.34 Upstream control enhanced register

The Upstream Control Enhanced Register is used to control the upstream channel time-out feature.

USCE Offset address: $09C_H$

Upstream control enhanced register Kernel Reset value: $0000\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
USTOIP		UTASR	0	USTS	USTC	USTF	USTOEN	USTOVAL				USTOPRE			
rw		rw	r	w	w	rh	rw	rw				rw			

Field	Bits	Type	Description
USTOPRE	3:0	rw	**Upstream Time-out Prescaler** Prescaler for the upstream time-out limit. Expressed in upstream bit times. 2^n divider in the range of 1 to 16384. Write to this bit field triggers new start of the watchdog timer. 0_H 1 … E_H 16384 F_H reserved

(table continues...)

(continued)

Field	Bits	Type	Description
USTOVAL	7:4	rw	**Upstream Time-out Value** Upstream time-out value for the N-divider in the range of 1 to 16, expressed in number of upstream bit time lengths (the upstream bit time is the reciprocal value of the upstream baud rate). Time-out = BitTime * $2^{(USTOPRE+1)}$ * (USTOVAL+1). Write to this bit field triggers new start of the watchdog timer.
USTOEN	8	rw	**Upstream Time-out Interrupt Enable** Enable bit for the upstream time-out interrupt. The Time-out counter runs continuously independently of the USTOEN bit. The USTOEN enables only the interrupt generation. 0_B Disabled 1_B Enabled
USTF	9	rh	**Upstream Time-out Flag** Signals an upstream timer event. If set by software, the interrupt is also triggered, if enabled. The watchdog timer runs continuously and sets the USTF flag at overflow, independently from the enable bit. Therefore this bit must be cleared with the same write access which enables the interrupt. 0_B Disabled 1_B Enabled
USTC	10	w	**Upstream Time-out Clear** Clears the USTF flag. Write only bit, reads as 0. 0_B No action 1_B Clear
USTS	11	w	**Upstream Time-out Set** Sets the USTF flag. Write only bit, reads as 0. 0_B No action 1_B Set
UTASR	13	rw	**Upstream Time-out Alternate Service Request** Selects if the interrupt signal is routed to the alternate service request node. Refer to interrupt request compressor. 0_B SR Multiplexer selected 1_B Alternate Service Request Node (SR4) selected
USTOIP	15:14	rw	**Upstream Time-out Interrupt Node Pointer** USTOIP selects the service request output line SRn (n = 0-3) for the time-out interrupt. 00_B Service request output SR0 selected 01_B Service request output SR1 selected 10_B Service request output SR2 selected 11_B Service request output SR3 selected
0	12, 31:16	r	**Reserved** Read as 0; should be written with 0.

28.4.35 Downstream select data source low extension register

The bit fields of the Downstream Select Data Source Low Extension Register DSDSLE determine the data source for the upper 16 bits in the shift register SRL.

DSDSLE Offset address: $0A0_H$

Downstream select data source low extension register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SL31		SL30		SL29		SL28		SL27		SL26		SL25		SL24	
rw		rw		rw		rw		rw		rw		rw		rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SL23		SL22		SL21		SL20		SL19		SL18		SL17		SL16	
rw		rw		rw		rw		rw		rw		rw		rw	

Field	Bits	Type	Description
SLx (x=16-31)	2*x-31:2*x-32	rw	**Select Source for SRL** SLx determines which data source is used for the shift register bit SRL[x] during data frame transmission. 00_B SRL[x] is taken from data register DDE.DDLE[x]. 01_B Reserved. 10_B SRL[x] is taken from the ALTINLE[15:0] input line x-16. 11_B SRL[x] is taken from the ALTINLE[15:0] input line x-16 in inverted state.

28.4.36 Downstream select data source high extension register

The bit fields of the Downstream Select Data Source High Extension Register DSDSHE determine the data source for the upper 16 bit in the shift register SRH.

DSDSHE Offset address: $0A4_H$

Downstream select data source high extension register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SH31		SH30		SH29		SH28		SH27		SH26		SH25		SH24	
rw		rw		rw		rw		rw		rw		rw		rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SH23		SH22		SH21		SH20		SH19		SH18		SH17		SH16	
rw		rw		rw		rw		rw		rw		rw		rw	

Field	Bits	Type	Description
SHx (x=16-31)	2*x-31:2*x-32	rw	**Select Source for SRH** SHx determines which data source is used for the shift register bit SRH[x] during data frame transmission. 00_B SRH[x] is taken from data register DDE.DDHE[x]. 01_B Reserved. 10_B SRH[x] is taken from the ALTINHE[15:0] input line x-16. 11_B SRH[x] is taken from the ALTINHE[15:0] input line x-16 in inverted state.

28.4.37 Emergency stop extension register

The Emergency Stop Extension Register ESRE determines which bits of SRL and SRH are enabled for emergency operation.

ESRE Offset address: $0A8_H$

Emergency stop extension register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ENH31	ENH30	ENH29	ENH28	ENH27	ENH26	ENH25	ENH24	ENH23	ENH22	ENH21	ENH20	ENH19	ENH18	ENH17	ENH16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ENL31	ENL30	ENL29	ENL28	ENL27	ENL26	ENL25	ENL24	ENL23	ENL22	ENL21	ENL20	ENL19	ENL18	ENL17	ENL16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

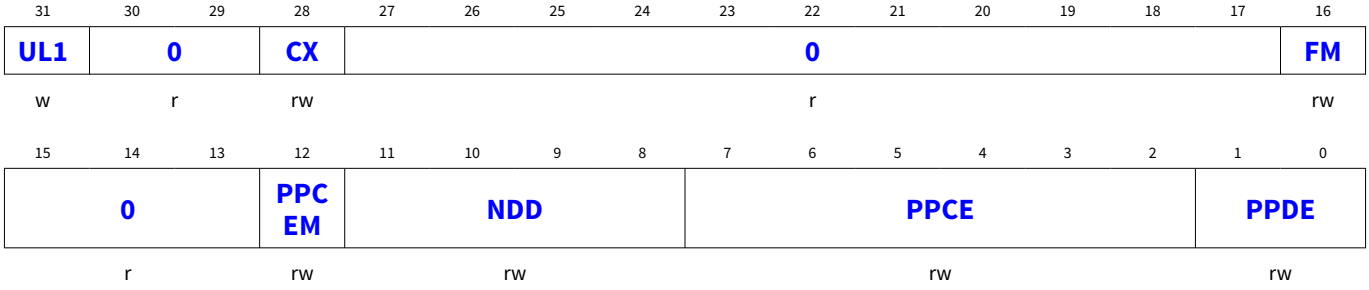
Field	Bits	Type	Description
ENLx (x=16-31)	x-16	rw	**Emergency Stop Enable for Bit x in SRL** This bit enables the emergency stop feature selectively for each SRL bit. If the emergency stop condition is met and enabled (ENLx = 1), the SRL[x] bit is of the data register DDE.DDLE[x] is used for the shift register load operation. 0_B Emergency stop feature for bit SRL[x] is disabled. 1_B The emergency stop feature for bit SRL[x] is enabled.
ENHx (x=16-31)	x	rw	**Emergency Stop Enable for Bit x in SRH** This bit enables the emergency stop feature selectively for each SRH bit. If the emergency stop condition is met and enabled (ENHx = 1), the SRH[x] bit of the data register DDE.DDHE[x] is used for the shift register load operation. 0_B Emergency stop feature for bit SRH[x] is disabled. 1_B The emergency stop feature for bit SRH[x] is enabled.

28.4.38 Downstream timing extension register

The Downstream Timing Extension Register is used to control the enhanced operation mode and frame layout of the downstream channel transmission.

DSTE Offset address: $0AC_H$

Downstream timing extension register Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
PPDE	1:0	rw	**Passive Phase Length at data frames Extension** Additional MSB bits extension of the DSC.PPD bit field.
PPCE	7:2	rw	**Passive Phase Length at Control Frames Extension** Additional MSB bits extension of the fixed length of 2 of the command frames passive phase. The final length is 2 + PPCE resulting in a range of 2 to 65. 00_H 2 … $3F_H$ 65
NDD	11:8	rw	**N Divider Downstream** Defines the division ratio in the range of 1 to 16. 0_H 1 … F_H 16
PPCEM	12	rw	**PPCE Extension Bit on the MSB Side** This bit is the MSB extension bit for the PPCE bit field, extending command frame passive phase to the value of 127. The values of 128 and 129 are not allowed. 0_B MSB value of 0 1_B MSB value of 1
FM	16	rw	**Fast Mode** Activates the fast mode and writing to the ABRA FIFO with 100 MBaud input baud rate, in order to provide 80MBaud output baud rate. It is also recommended for baud rate of 26,67 MBaud, as defined in the committed baud rates. FM=0 is the compatibility setting with the previous generations of the MSC. 0_B Fast Mode deactivated 1_B Fast Mode activated

(table continues...)

(continued)

Field	Bits	Type	Description
CX	28	rw	**Command Extension Mode** Activates 64 bit command frame feature. 0_B CX mode deactivated 1_B CX mode activated
UL1	31	w	**Unlock CX and FM for one write access** Write access with UL1 = 1 unlocks the CX and FM bit fields for the current write access. Write accesses to these bit fields with UL1 = 0 do not have any effect. Returns zero on read.
0	15:13, 27:17, 30:29	r	**Reserved** Read as 0; should be written with 0.

28.4.39 Downstream data mirror register

The Downstream Data Mirror Register DDM provides alternative address for the DD register. The purpose of this alternative address is to provide convenient way to write up to 64 bit data blocks (32 bitwise) to the MSC module by using DMA with automatic address increment and wrap around. The DMA writes first to the DDE register and then to the DDM register, which behaves in the same way as a write to the DD register.

DDM Offset address: $0B0_H$

Downstream data mirror register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DDHM															
w															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DDLM															
w															

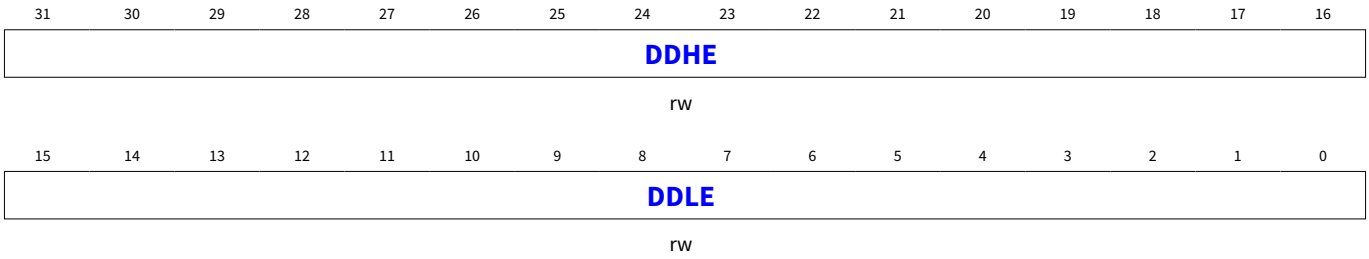
Field	Bits	Type	Description
DDLM	15:0	w	**Downstream Data Mirror for SRL Shift Register** Contains the data bits to be transmitted during the SRL active phase of a data frame.
DDHM	31:16	w	**Downstream Data Mirror for SRH Shift Register** Contains the data bits to be transmitted during the SRH active phase of a data frame.

28.4.40 Downstream data extension register

The Downstream Data Extension Register DDE contains data to be transmitted during data frames.

DDE Offset address: $0B4_H$

Downstream data extension register Kernel Reset value: $0000\ 0000_H$



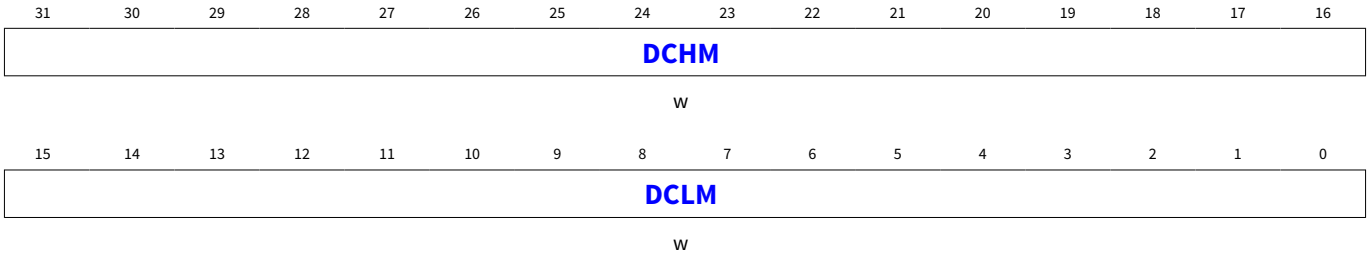
Field	Bits	Type	Description
DDLE	15:0	rw	**Downstream Data Extension for SRL Shift Register** Contains the data bits to be transmitted during the SRL active phase of a data frame.
DDHE	31:16	rw	**Downstream Data Extension for SRH Shift Register** Contains the data bits to be transmitted during the SRH active phase of a data frame.

28.4.41 Downstream command mirror register

The Downstream Command Mirror Register DCM provides alternative address for the DC register. This alternative address provides convenient way to write up to 64 bit data blocks (32 bitwise) to the MSC module by using DMA with automatic address increment and wrap around. The DMA writes first to the DCE register and then to the DCM register. Every write to DCE triggers a command transmission the same way as a write to the DC register.

DCM Offset address: $0B8_H$

Downstream command mirror register Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
DCLM	15:0	w	**Downstream Command Mirror of the DC.DCL Bit Field** Contains alternate write location for the command bits to be transmitted during the SRL active phase of a command frame.
DCHM	31:16	w	**Downstream Command Mirror of the DC.DCH Bit Field** Contains the alternate write location for the command bits to be transmitted during the SRH active phase of a command frame.

28.4.42 Downstream command extension register

The Downstream Command Extension Register DCE contains data to be transmitted during data frames, which are concatenated to the standard 32-bit commands, after the second select bit.

DCE Offset address: $0BC_H$

Downstream command extension register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DCEH															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DCEH															
rw															

Field	Bits	Type	Description
DCEH	31:0	rw	**Downstream Command Extension for SRH Shift Register** Contains the data bits to be transmitted during the second half of the command frame in CX mode.

28.4.43 Asynchronous block configuration register

The Asynchronous Block Configuration register ABC contains all the bit fields related to the ABRA block.

***Note**: Switching the peripheral clock off and on is done by using two writes: first a write of zero to the CLKSEL bitfield and then a second write switching on the clock source. Between the first and the second write a delay of minimum 4 * (1/f_{PER}) + 2 * (1/f_{CLC}) must be inserted by software, where f_{PER} is the frequency being switched off with the first write. This also applies if in-between the writes the clock frequency is changed in the CCU.*

ABC Offset address: $0C0_H$

Asynchronous block configuration register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ABB	0	CLKSEL			UIE	UFM		UNF	0	UASR	UIP		NDA		
rw	r	rw			rw	w		rh	r	rw	rw		rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OIE	OFM		OVF	0	OASR	OIP		HIGH					LOW		
rw	w		rh	r	rw	rw		rw					rw		

Field	Bits	Type	Description
LOW	3:0	rw	**Duration of the Low Phase of the Shift Clock** Defines the duration of the low phase of the shift clock in periods of f_A in the range of 1 to 16. 0_H 1 … F_H 16
HIGH	7:4	rw	**Duration of the High Phase of the Shift Clock** Defines the duration of the high phase of the shift clock in periods of f_A in the range of 1 to 16. 0_H 1 … F_H 16
OIP	9:8	rw	**Overflow Interrupt Node Pointer** OIP selects the service request output line SRn (n = 0-3) for the overflow interrupt. 00_B Service request output SR0 selected 01_B Service request output SR1 selected 10_B Service request output SR2 selected 11_B Service request output SR3 selected
OASR	10	rw	**Overflow Alternate Service Request** Selects if the interrupt signal is routed to the alternate service request node. Refer to interrupt request compressor. 0_B SR Multiplexer selected 1_B Alternate Service Request Node (SR4) selected
OVF	12	rh	**Overflow Flag** Indicates if overflow error has occurred. 0_B No overflow error 1_B Overflow error
OFM	14:13	w	**Overflow Flag Modify** Sets or clears the overflow flag OVF. 00_B No action 01_B Set (and triggers the overflow interrupt if enabled) 10_B Clear (additionally flushes the ABRA FIFO and resets the bit counter) 11_B No action
OIE	15	rw	**Overflow Interrupt Enable** Enables or disables the path of the interrupt signal towards the interrupt node. If enabled, an overflow event triggers an interrupt and if disabled then not. The OVF flag always indicates the occurrence of an overflow event, independently of OIE. 0_B Overflow interrupt disabled 1_B Overflow interrupt enabled

(table continues...)

(continued)

Field	Bits	Type	Description
NDA	18:16	rw	**N Divider ABRA** Defines the division ratio in the range of 1 to 8. 000_B 1 … 111_B 8
UIP	20:19	rw	**Underflow Interrupt Node Pointer** UIP selects the service request output line SRn (n = 0-3) for the underflow interrupt. 00_B Service request output SR0 selected 01_B Service request output SR1 selected 10_B Service request output SR2 selected 11_B Service request output SR3 selected
UASR	21	rw	**Underflow Alternate Service Request** Selects if the interrupt signal is routed to the alternate service request node. Refer to interrupt request compressor. 0_B SR Multiplexer selected 1_B Alternate Service Request Node (SR4) selected
UNF	23	rh	**Underflow Flag** Indicates if underflow error has occurred. 0_B No underflow error 1_B Underflow error
UFM	25:24	w	**Underflow Flag Modify** Sets or clears the underflow flag UNF. 00_B No action 01_B Set (also triggers the underflow interrupt if enabled) 10_B Clear (additionally flushes the ABRA FIFO and resets the bit counter) 11_B No action
UIE	26	rw	**Underflow Interrupt Enable** Enables or disables the path of the interrupt signal towards the interrupt node. If enabled, an underflow event triggers an interrupt and if disabled then not. The UNF flag always indicates the occurrence of an underflow event, independently of UIE. 0_B Underflow interrupt disabled 1_B Underflow interrupt enabled
CLKSEL	29:27	rw	**Clock Select** Selects the clock source for the ABRA block. 000_B no clock 001_B f_{PER} **others,** reserved (no clock)

(table continues...)

(continued)

Field	Bits	Type	Description
ABB	31	rw	**Asynchronous Block Bypass** Defines if the asynchronous block and the n-divider of the MSC downstream path (located parallel to the fractional divider) are used or not. If bypassed, then also disabled. 0_B Bypassed and disabled 1_B Not bypassed and active
0	11, 22, 30	r	**Reserved** Read as 0; should be written with 0.

28.4.44 Downstream command slow channel X register

The Downstream Command Slow Channel X Register contains data to be transmitted through the Slow Channel for Slave X.

DCSX Offset address: $0C4_H$

Downstream command slow channel X register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DCSXD															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DCSXD															
rw															

Field	Bits	Type	Description
DCSXD	31:0	rw	**Downstream Command Slow Channel X Data** Contains the command frame data bits to be transmitted through the Slow Channel in Slow Channel mode.

28.4.45 Downstream command slow channel Y register

The Downstream Command Slow Channel Y Register contains data to be transmitted through the Slow Channel for Slave Y.

DCSY Offset address: $0C8_H$

Downstream command slow channel Y register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DCSYD															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DCSYD															
rw															

Field	Bits	Type	Description
DCSYD	31:0	rw	**Downstream Command Slow Channel Y Data** Contains the command frame data bits to be transmitted through the Slow Channel in Slow Channel mode.

28.5 Debug information

28.5.1 OCDS suspend behavior

In case of hard suspend, the clock is switched off immediately. A running frame will not be finished.

In case of soft suspend, the state is entered after the following two conditions are met:

- Any ongoing upstream reception is completed and
- Any ongoing downstream frame transmission is completed, both from the MSC kernel and the ABRA block side

28.6 References

This section is not applicable for the module being described.

28.7 MSC revision history

Initial release of the chapter.

28.8 TC4Dx MSC information

28.8.1 TC4Dx MSC configuration

There are no deviations from the generic specification.

28.8.2 TC4Dx MSC features

There are no deviations from the generic specification.

28.8.3 TC4Dx MSC functional description

1. MSC acts as a slave on the ComPB bus. The Clock Control block is driven by the system clock frequency f_{SPB}. As in TC4Dx, the ComPB also uses this clock
2. There is no GTM trigger available. Only eGTM trigger signals are to be selected

28.8.4 TC4Dx MSC registers

There are no deviations from the generic specification.

28.8.4.1 Register address space - MSC

Table 1221 Registers address space - MSC

Module	Base address	End address	Note
MSC0	$F4400800_H$	$F44009FF_H$	FPI slave interface

28.8.4.2 Register overview - access mode glossary

Table 1222 Register overview - access mode glossary

Keyword	Description
E	Access protection using PROT register MSC0_PROTE .
SE	Access protection using PROT register MSC0_PROTSE .
APU-P	Protection group consisting of registers MSC0_ACCEN_WRA , MSC0_ACCEN_WRB , MSC0_ACCEN_RDA , MSC0_ACCEN_RDB , MSC0_ACCEN_VM , MSC0_ACCEN_PRS .
P	Access protection using *APU-P* registers.
BE	Always returns a Bus Error.
U	No access restrictions.
PROT	Access restrictions as defined in the PROT register access rules.

28.8.4.3 Registers overview - MSC0 (ascending offset address)

Table 1223 Registers overview - MSC0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
MSC0_CLC	Clock control register	000_H	P	P, SV, E	Application Reset	5139
MSC0_OCS	OCDS control and status register	004_H	P	SV, P	Debug Reset	5140
MSC0_ID	Module identification register	008_H	P	BE	PowerOn Reset	5141
MSC0_RST_CTRLA	Reset control register A	$00C_H$	P	P, SV, E	Application Reset	5141
MSC0_RST_CTRLB	Reset control register B	010_H	P	P, SV, E	Application Reset	5142
MSC0_RST_STAT	Reset status register	014_H	P	BE	Application Reset	5143
MSC0_PROTE	PROT register endinit	018_H	U	SV, PROT	Application Reset	5143
MSC0_PROTSE	PROT register safe endinit	$01C_H$	U	SV, PROT	Application Reset	5145

(table continues...)

Table 1223 (continued) Registers overview - MSC0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
MSC0_ACCEN_WRA	Write access enable register A	020_H	U	SE, SV	Application Reset	5146
MSC0_ACCEN_WRB	Write access enable register B	024_H	U	SE, SV	Application Reset	5147
MSC0_ACCEN_RDA	Read access enable register A	028_H	U	SE, SV	Application Reset	5147
MSC0_ACCEN_RDB	Read access enable register B	$02C_H$	U	SE, SV	Application Reset	5148
MSC0_ACCEN_VM	VM access enable register	030_H	U	SE, SV	Application Reset	5148
MSC0_ACCEN_PRS	PRS access enable register	034_H	U	SE, SV	Application Reset	5149
MSC0_FDR	Fractional divider register	040_H	P	SV, P, E	Kernel Reset	5150
MSC0_USR	Upstream status register	044_H	P	P	Kernel Reset	5151
MSC0_DSC	Downstream control register	048_H	P	P	Kernel Reset	5152
MSC0_DSS	Downstream status register	$04C_H$	P	P	Kernel Reset	5154
MSC0_DD	Downstream data register	050_H	P	P	Kernel Reset	5156
MSC0_DC	Downstream command register	054_H	P	P	Kernel Reset	5156
MSC0_DSDSL	Downstream select data source low register	058_H	P	P	Kernel Reset	5157
MSC0_DSDSH	Downstream select data source high register	$05C_H$	P	P	Kernel Reset	5157
MSC0_ESR	Emergency stop register	060_H	P	P	Kernel Reset	5158
MSC0_UDx (x=0-3)	Upstream data register x	070_H+x* 4	P	P	Kernel Reset	5158
MSC0_ICR	Interrupt control register	080_H	P	P	Kernel Reset	5159
MSC0_ISR	Interrupt status register	084_H	P	BE	Kernel Reset	5162
MSC0_ISC	Interrupt set clear register	088_H	P	P	Kernel Reset	5163
MSC0_OCR	Output control register	$08C_H$	P	P	Kernel Reset	5165
MSC0_OECR	Output extended control register	090_H	P	P	Kernel Reset	5167
MSC0_SCC	Slow channel configuration register	094_H	P	P	Kernel Reset	5170
MSC0_DSCE	Downstream control enhanced register	098_H	P	P	Kernel Reset	5172
MSC0_USCE	Upstream control enhanced register	$09C_H$	P	P	Kernel Reset	5174

(table continues...)

Table 1223 (continued) Registers overview - MSC0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
MSC0_DSDSLE	Downstream select data source low extension register	$0A0_H$	P	P	Kernel Reset	5176
MSC0_DSDSHE	Downstream select data source high extension register	$0A4_H$	P	P	Kernel Reset	5176
MSC0_ESRE	Emergency stop extension register	$0A8_H$	P	P	Kernel Reset	5177
MSC0_DSTE	Downstream timing extension register	$0AC_H$	P	P	Kernel Reset	5178
MSC0_DDM	Downstream data mirror register	$0B0_H$	P	P	Kernel Reset	5179
MSC0_DDE	Downstream data extension register	$0B4_H$	P	P	Kernel Reset	5179
MSC0_DCM	Downstream command mirror register	$0B8_H$	P	P	Kernel Reset	5180
MSC0_DCE	Downstream command extension register	$0BC_H$	P	P	Kernel Reset	5181
MSC0_ABC	Asynchronous block configuration register	$0C0_H$	P	P	Kernel Reset	5181
MSC0_DCSX	Downstream command slow channel X register	$0C4_H$	P	P	Kernel Reset	5184
MSC0_DCSY	Downstream command slow channel Y register	$0C8_H$	P	P	Kernel Reset	5184

28.8.4.4 Device specific registers

There are no device specific register changes.

28.8.5 TC4Dx MSC connectivity

The table below lists interfaces to and from this functional block to other blocks in the device.

Table 1224 List of MSC interface signals

Interface signals	I/O	Description
CLOCK_MSC_fSPB	In	SPB clock input
CLOCK_MSC_fSRI	In	SRI clock input
CLOCK_MSC_fMSC	In	Peripheral clock input
MSC0_IR_SR[4:0]	Out	Service Requests
EGTM_MSC0_MSCSELIN_INLCON	In	eGTM timer output vector [15:0]- low part
EGTM_MSC0_MSCSELIN_INLECON	In	eGTM timer output vector [15:0] - low extension part
EGTM_MSC0_MSCSELIN_INHCON	In	eGTM timer output vector [31:16]- high part
EGTM_MSC0_MSCSELIN_INHECON	In	eGTM timer output vector [31:16] - high extension part
PORTS_MSC0_SDI[7:0]	In	Upstream asynchronous input signal
PORTS_MSC0_INJ[1:0]	In	Injection signal from port
MSC0_PORTS_EN[3:0]	Out	Chip Select
SMU_MSC_PORTSEmergencyStop	In	Emergency stop signal from SMU
MSC0_PORTS_FCL[P,N]	Out	Shift-clock signal pairs with inverted signal polarities FCLP/FCLN
MSC0_PORTS_SO[P,N]	Out	Data output signal pairs with inverted signal polarities SOP/SON
MSC0_SCU_FCLP	Out	MSC0 FCLP output to ERU

28.8.6 TC4Dx MSC revision history

Initial release of the chapter.

29 Secure Digital Multi-Media Card (SDMMC)

The purpose of the SDMMC module is to enable communication to external managed NAND flashes and SDIO (Secure Digital Input Output) devices using the SD (Secure Digital) or eMMC(embedded Multi-Media Card) interface. The focus of the module is communication with a single eMMC memory device, SD-card or a SDIO device.

29.1 Feature list

- Communication to eMMC (embedded multi-media card) memories
  - Communication using 1, 4, or 8-data lines
  - Legacy MMC (multi-media Card) and High-speed SDR (single data rate) mode supported
- Communication to SD-cards
  - Communication using 1 or 4-data lines
  - Default- and High Speedmode supported
- Communication to SDIO cards
- Communication support using different levels of direct memory access (DMA) support
  - Programmed Input Output (PIO) - operation without DMA
  - Single operation DMA (SDMA)
  - Advanced DMA-2 (ADMA2)
  - Advanced DMA-3 (ADMA3)
- Provides 1 KB SRAM for buffering up to two 512-Byte blocks
- Compliant to "Embedded Multi-Media Card (eMMC) Standard (5.1)" with the following exceptions:
  - No dual voltage eMMC support, only high voltage (3.3 V)
  - No high clock frequencies for HS200/HS400, only 0-50 MHz
  - No large 4-kB sector device support, only 512-B sectors
  - No DDR (Double Data Rate) signaling only SDR (Single Data Rate) signaling
  - No data strobe support
  - No high speed DDR, HS200 or HS400 speed mode support, only Legacy MMC card and High Speed SDR
  - No “Interrupt” mode, only “device identification”, “data transfer” and “inactive” mode
  - No clock tuning support
  - No multi device support
  - No Command Queuing Engine (CQE)
- Compliant to “SD Specifications Part A2 SD Host Controller Standard Specification v4.20” with the following exceptions:
  - No Multi Slot Support , only single cards
  - No suspend and resume functionality
  - No Sampling Clock Tuning supported, only Default Speed mode and High Speed mode supported
  - No Suspend or Resume operation in an SDIO card
- Compliant to “SD Specifications Part 1 Physical Layer Specification v6.00” with the following exceptions:
  - No UHS-II (Ultra High Speed) voltages, only high voltage SD memory card
  - No UHS-I and UHS-II speed modes, only Default Speed mode and High Speed mode
  - No physical write protection switch support
  - No card insertion detection, only sensing of pull-up on DAT line
  - No SPI card support
- Compliant to SD Specifications Part E1 SDIO Specification Version 4.10

29.2 Functional overview

The SDMMC controller supports SD-card, eMMC and SDIO interfaces. It contains a DMA engine for data transfer, a registers unit, and a FIFO controller.

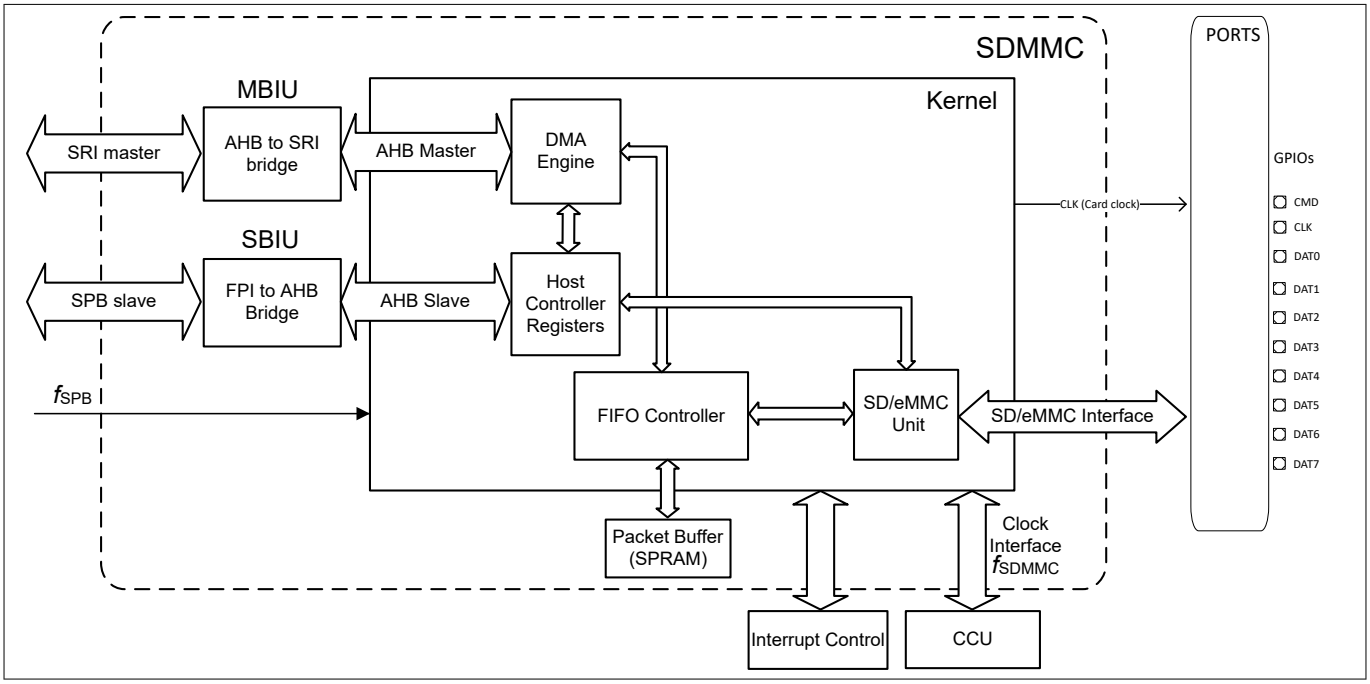


Figure 627 SDMMC block diagram

The figure above provides a simplified overview of SDMMC in the form of a block diagram.

The SDMMC kernel is connected to the SPB slave through the FPI to AHB bridge and SRI master via AHB to SRI bridge. Access to host control registers takes place through slave interface. Data transfers from or to external NAND memory to or from internal system memory space takes place through an internal Direct Memory Access(DMA) engine through the master interface. The slave and master clock domain runs at a frequency of f_{SPB} The card clock domain runs at the peripheral frequency of f_{SDMMC}. The Host controller accesses the external memory device over the GPIO pads belonging to the PORTS functional block. SDMMC has a single port RAM for packet buffering interfacing the FIFO controller.

Related information

TC4Dx SMU alarm mapping tables on page 7233

29.3 Functional description

The following sections describes briefly the protocol overview, SDMMC programming sequences and Error handling.

29.3.1 Architecture overview

The following are the various modules in the SDMMC functional block:

***Note**: Only a brief description of the blocks are given below, please refer to the standards and specifications listed in References section for more details.*

Master bus interface unit

The master bus interface unit (MBIU) transfers data to and from the system memory through the system resource interconnect (SRI) master.

Slave bus interface unit

The slave bus interface unit (SBIU) implements the logic to access the SDMMC registers through the system peripheral bus (SPB) slave.

DMA engine

The DMA engine unit handles data transfer between SDMMC and system memory. The key features of this unit are as follows:

- Supports: SDMA,ADMA2 and ADMA3 modes (for more details on DMA programming then please refer to Application layer data flow section)
- Write back the received data packets to the system memory
- Write data from system memory to the FIFO for SD or SDIO or eMMC transfer
- Support of SINGLE and burst (INCR4 and INCR8) transfers

  ***Note**: In order to achieve optimal performance with burst transfers, data should be aligned to 256-bit for INCR8 and 128-bit for INCR4. For example, in case of INCR8 and non 256-bit aligned data, the DMA engine will use SINGLE and INCR4 transfers until 256-bit alignment is achieved*

- Pre-fetches data for back-to-back eMMC write commands
- 32-bit addressing support
- 32-bit data width

SDMMC registers

The host controller register unit comprises of the standard SD host controller registers as specified in the SD Specifications Part A2 SD Host Controller Standard specification version 4.20a.

For the following registers the internal clock controlled by CLK_CTRL.INTERNAL_CLK_EN must be enabled before accessing them:

- BLOCKSIZE
- BLOCKCOUNT
- ARGUMENT
- XFER_MODE
- CMD

FIFO controller

The data FIFO controller interfaces the internal FIFO that transport data To/From Packet Buffer (External Memory) and the DMA Engine.

SD/eMMC unit

This unit is responsible for managing SD/eMMC interface protocols. The key features of this unit are to:

- Generate command and data packets
- Generate Cyclic redundancy Check (CRC) to check command and data packets
- Handle packet timeouts
- Support 1-bit DAT, 4-bit DAT, and 8-bit DAT modes

SD/eMMC interface signals

SD/eMMC interface signals to the pads are

- SDMMC_DAT0-7: data lines
- SDMMC_CMD: command line
- SDMMC_CLK: clock line

For the different data widths 1, 4, and 8-wire communication different enable signals for the corresponding data pins are required to activate and deactivate these lines according to the output width selection.

SPRAM (single port RAM)

SRAM that is internal to the SDMMC functional block is used as a packet buffer to store data packets while carrying out data transfer to and from the card. The size of the SRAM is 1 KB that supports buffering of two 512-bytes blocks.

In the unlikely scenario when the packet buffer may become full or empty during card read, the clock to the card is stopped in SD/eMMC mode to prevent card from sending the next data block.

The single port RAM is not initialized after reset. Reading the RAM should only be done after the uninitialized values got overwritten (either data was received by the card, or the RAM was written by application software to prepare a write to the card). This is to avoid Checksum errors (safety alerts) when the software accesses uninitialized RAM cells.

As write and read transfers to the cards do not occur simultaneously, a single shared buffer is used for read and write operations. During the data transfer command handshake, the read/write bit of the command register is sampled and stored. This read/write bit defines whether the SDMMC is in read or write mode. The figure below shows how data flows from the card interface to the AHB master interface unit through the packet buffer for a card read transfer.

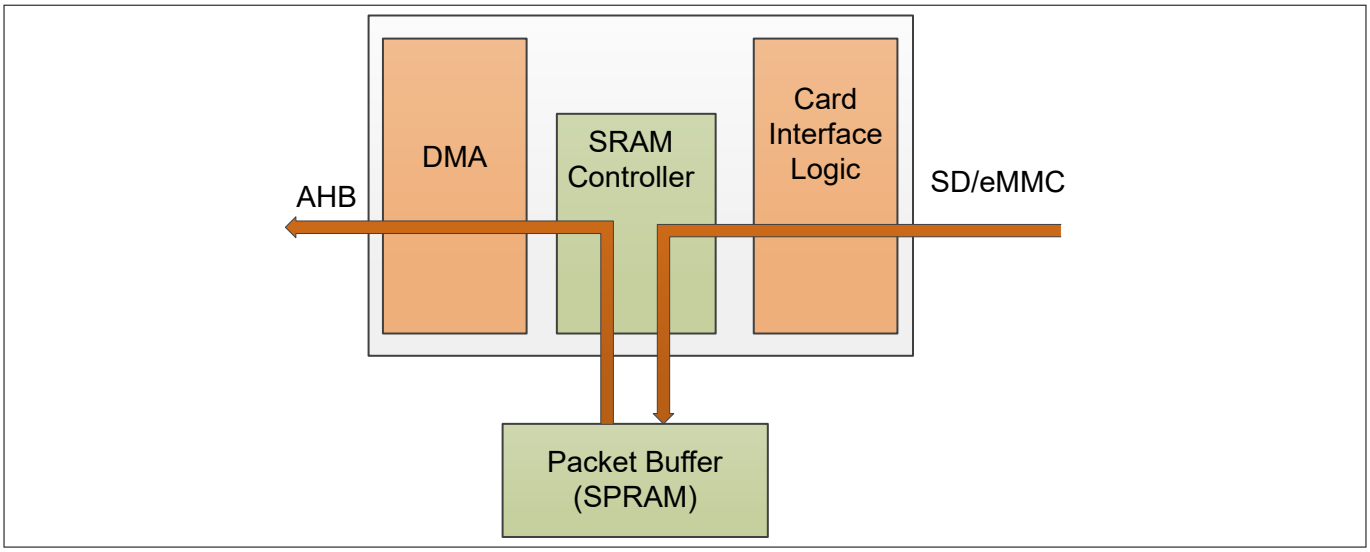


Figure 628 Data flow for card read

Received data from the card interface is written into the packet buffer. When one block of data (512 bytes) is received, DMA starts transmitting that data to the system memory by reading it from the packet buffer. For a card write transfer, data flows in the reverse direction. DMA writes data into a packet buffer that is subsequently read by the card interface logic. DMA and card interface logic can work simultaneously because read and write to packet buffer can be interleaved.

For card read, DMA can send out the previous block while card interface logic is receiving the current block. For card write, DMA can write the current block into packet buffer while card interface logic is sending out the previous block. When the packet buffer becomes full in card read, the clock to the card is stopped to prevent the card from sending the next data block. When packet buffer is empty, data block is not sent. In both cases, card interface logic is idle. SDMMC does not support SDIO Read Wait signaling through DAT[2]. Therefore, the I/O command (CMD52) cannot be performed during a multiple read cycle because the card clock is stopped.

Interrupts

The SDMMC supports the following interrupts:

- Normal Interrupt signal
  - Used as interrupt for different events during active mode. The Interrupt Status Registers (NORMAL_INT_STAT) indicate the events that caused the interrupt generation. Each event can be prevented from asserting the interrupt on the interrupt signal by setting the corresponding mask bits NORMAL_INT_SIGNAL_EN. When an interrupt event occurs, it will be captured in the Interrupt Status Registers (NORMAL_INT_STAT) only if the corresponding interrupt bit is enabled in the NORMAL_INT_STAT_EN register. The interrupt line gets active high only when the interrupt status in (NORMAL_INT_STAT) register and the corresponding interrupt signal enable in NORMAL_INT_SIGNAL_EN register are set to 1
- Wake-up Interrupt signal
  - During standby mode, Wake-up Interrupt Signal must be used to identify any wake-up event, such as an SD-card insertion and removal. The corresponding register is WUP_CTRL. As card insertion and removal is not applicable to an embedded device, wake-up interrupt should not be used in this case. However it can still be used for SDIO card interrupt
- Error Interrupts
  - Used as interrupt for errors arising from different events during active mode captured in register SDMMC_ERROR_INT_STAT and enabled bySDMMC_ERROR_INT_STAT_EN and ERROR_INT_SIGNAL_EN(similar to normal interrupts)

The interrupts are signaled to the Interrupt Router (IR) by using a single pulse using a pulse shaper.

SDMMC host driver is responsible for enabling wake-up signals and disabling interrupt signals when the host system enters its standby mode, and for disabling wake-up signals and enabling interrupt signals when host

system goes into active mode. The host driver must not enable both at same time. Interrupt signals are enabled using interrupt signal enable and wake-up signals are enabled using wake-up event enable.

Table 1225 Interrupt grouping towards IR

Interrupt grouping towards interrupt router(IR)	Interrupts in SDMMC
SRC_SDMMCERR	Normal interrupt signal
	Error interrupts
SRC_SDMMCDMA	Wake-up interrupt signal

Clock interface

The clock interface is the interface to the CCU (clock control unit) in order to derive a 100 MHz peripheral PLL clock f_{SDMMC}. The card clock 'SDCLK' is subsequently derived from f_{SDMMC} using the controls in the register CLK_CTRL.

29.3.2 Error handling

SDMMC is capable of detecting different types of error in SD and eMMC. An error can be detected in either the command or data portion of the transaction. When an error is detected, the error interrupt in the normal interrupt status register (NORMAL_INT_STAT) is set.

When an error occurs in the SD or eMMC mode, one of the bits is set in the error interrupt status register(ERROR_INT_STAT). To recover from an error detected during data transfer, predefined commands are used.

There are two more error status registers supported in both SD and eMMC modes

- AUTO_CMD_STAT
- ADMA_ERR_STAT

Table 1226 Error types and categories for SD or eMMC mode

Categories	Error type	Bit field
Command errors	• Command timeout error • Command CRC error • Command end bit error • Command index error • Response error	• ERROR_INT_STAT.CMD_TOUT_ERR • ERROR_INT_STAT.CMD_CRC_ERR • ERROR_INT_STAT.CMD_END_BIT_ERR • ERROR_INT_STAT.CMD_IDX_ERR • ERROR_INT_STAT.RESP_ERR

(table continues...)

Table 1226 (continued) Error types and categories for SD or eMMC mode

Categories	Error type	Bit field
Auto command errors	• Auto CMD12 Not Executed • Command not issued by auto CMD12 error • Auto command timeout error • Auto command CRC error • Auto command end bit error • Auto command index error • Auto CMD response error	• AUTO_CMD_STAT.AUTO_CMD12_NOT_EXEC • AUTO_CMD_STAT.CMD_NOT_ISSUED_AUTO_CMD12 • AUTO_CMD_STAT.AUTO_CMD_TOUT_ERR • AUTO_CMD_STAT.AUTO_CMD_CRC_ERR • AUTO_CMD_STAT.AUTO_CMD_EBIT_ERR • AUTO_CMD_STAT.AUTO_CMD_IDX_ERR • AUTO_CMD_STAT.AUTO_CMD_RESP_ERR
Data errors	• Data timeout error • Data CRC error • Data end bit error • ADMA error	• ERROR_INT_STAT.DATA_TOUT_ERR • ERROR_INT_STAT.DATA_CRC_ERR • ERROR_INT_STAT.DATA_END_BIT_ERR • ERROR_INT_STAT.ADMA_ERR
Other errors	• Current Limit Error • Boot Acknowledgment Error	• ERROR_INT_STAT.CUR_LMT_ERR • ERROR_INT_STAT.BOOT_ACK_ERR

29.3.3 Protocol overview

This section provides an overview of protocols that are supported in SDMMC. For detailed information about the protocols, refer to the applicable standard documents.

The SDMMC block operates in eMMC mode when the EMMC_CTRL.CARD_IS_EMMC bit is set; otherwise, it operates in SD/SDIO mode. The speed mode is configured through HOST_CTRL1 and HOST_CTRL2 registers as per following table.

Table 1227 Bus Speed Mode Configuration

Bus speed mode	HOST_CTRL1.**HIGH_SPEED_EN**	HOST_CTRL2.**SIGNALING_EN**	Preset register
SD default speed (DS)	0	0	PRESET_DS
SD high speed (HS)	1	0	PRESET_HS

29.3.3.1 SD bus protocol

The bus communication in SD mode is based on command and data bit streams each bounded by a start bit and a stop bit. A command is a token that starts an operation and is sent by the Host to the card. The command is transferred through the CMD line.

Response is a token that is sent by the addressed card to the host as a reply to the previous command. The response is also transferred serially through the CMD line.

Data can be set either by the Host (Write operation) or by the card (Read operation) and can be transferred through the data lines.

Data transfers in either direction are made in blocks. Each block has a CRC succeeding it. The transfers can either happen as a single or multiple block. Multiple block transmissions are terminated by the issue of a stop command on the command line.

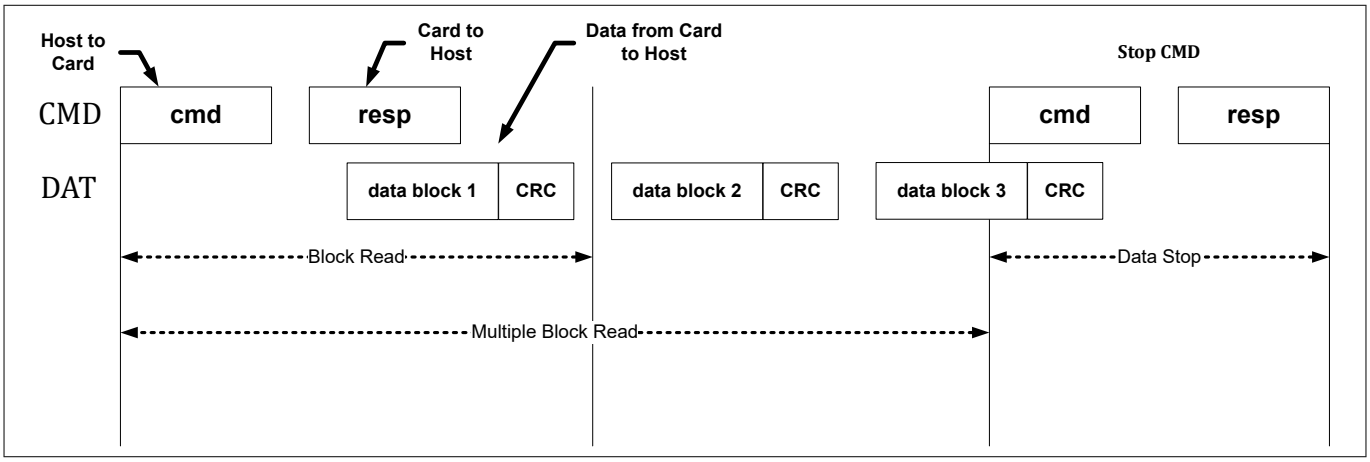


Figure 629 SD bus protocol; Command and data bit streams in the SD mode

29.3.3.2 SDIO bus protocol

An embedded SDIO device or card utilizes the SD bus and SD commands. Several commands required for SD Memory cards, example erase, are not supported by SDIO-only cards. In addition, there are several commands for SD cards that have different commands when used with the SDIO section of a card. [29)]

A host device controls the SDCLK (card clock) to stop the read data block output from a card executing a multiple read commands when the host cannot accept more data. The duration of time when the host has stopped the SDCLK, a CMD52 cannot be issued. This limitation causes a problem in that a host device cannot perform the I/O command during a multiple read cycle.

To overcome this limitation, the read wait feature is introduced to signal a card that is executing multiple read operations to temporarily pause data transfer while allowing the host to send commands to any other function within the card. Read wait feature is not supported by SDMMC. The read wait function is optional for SDIO cards.

SDIO Card Interrupt

An SD interface supports the interrupt function to allow an SDIO card to interrupt the host. This interrupt is used:

- To Inform the host about error status and also about normal information, such as function ready.
- As a source of wakeup event to inform the host to resume from standby mode

The card interrupt is cleared when the card/device stops asserting the interrupt, that is, when the application services the interrupt condition. For SD 1-bit mode, there are no timing constraints for interrupt. For 4-bit mode, the SDMMC samples the card interrupt only during the interrupt period that is defined as follows:

- Interrupt period ends from the end bit of a command that transfers the data block, 1 to 3 clocks for SDR50 mode
- Resume two clocks after the completion of last data block transfer, 2 to 4 for SDR50
- Within the data block gap, 2-clock Interrupt period begins 2 clocks after the end bit of data blocks. Block Gap card interrupt is disabled in SDR50

29 refer toSD Specifications Part E1 SDIO Specification Version document described in References section for more information about SDIO cards and differences between them and D-Card commands

- The SDMMC does not distinguish between synchronous and asynchronous interrupt periods
- In case of abort CMD, the interrupt period ends 2 clocks after the end bit of abort CMD and resumes 2 clocks after the end bit of the response

29.3.3.3 eMMC bus protocol

An eMMC bus data transfer contains a command, response and data block tokens. Each data transfer is considered as a bus operation that always contains a command and a response token. Additionally, certain bus operations also contain a data token.

Data sent on the data line can be transferred one bit (single data rate) or two bits (dual data rate, not supported) per clock cycle. Similar to the SD bus protocol, command and data bit streams are bound by a start bit and a stop bit.

Data transfer occurs in the form of blocks that are followed by CRC bits and data is transferred in single or multiple blocks. While a single or multiple read block is terminated by a stop command on the CMD line, a single or multiple write block is terminated by a busy signal on the data (DAT[0]) line.

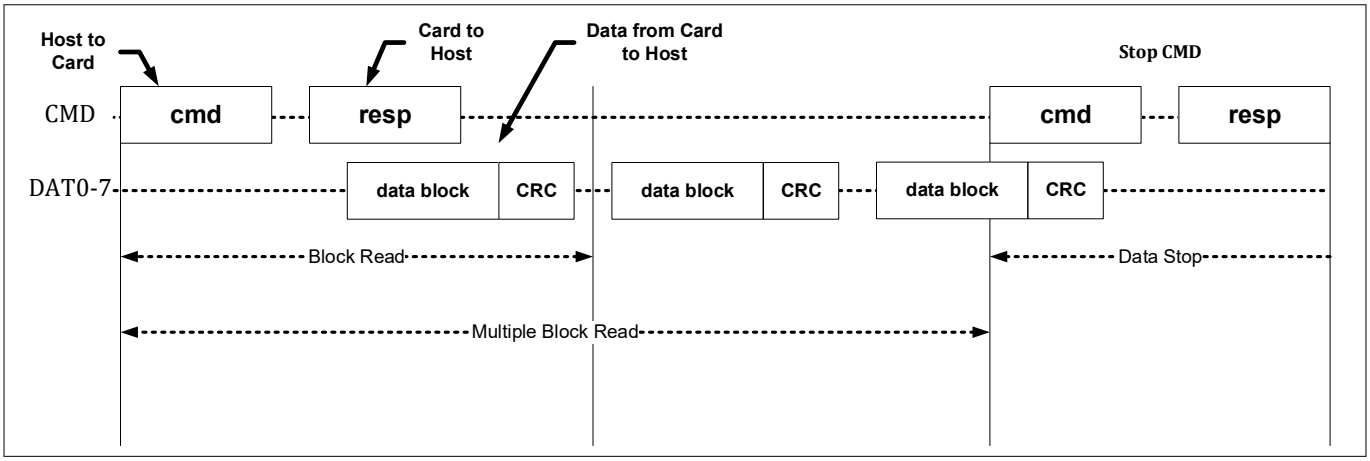


Figure 630 eMMC block read operation

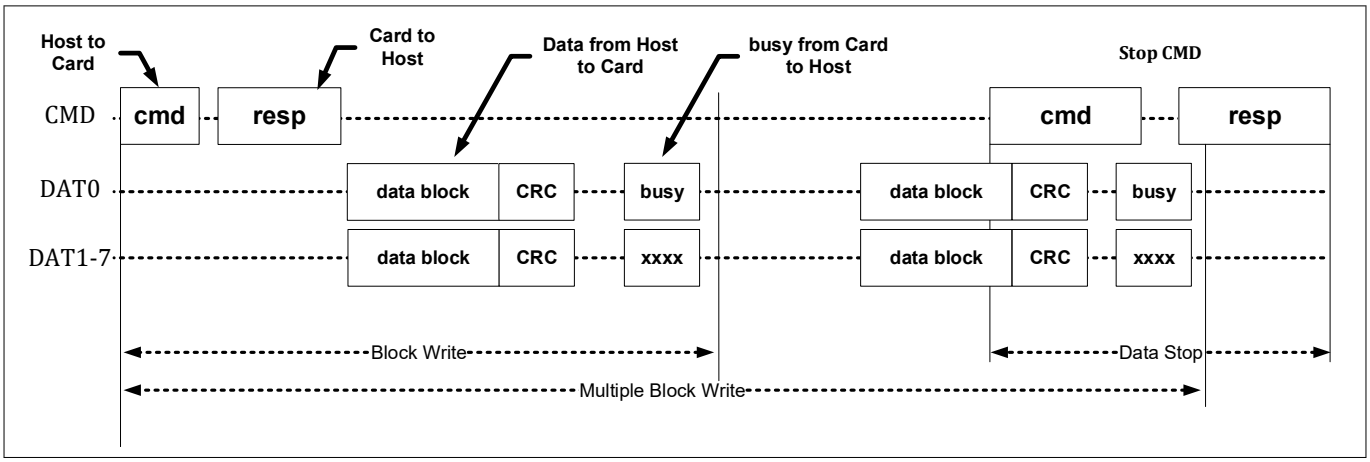


Figure 631 eMMC block write operation

29.3.4 Application layer data flow

The host controller communicates to the software driver using the system bus interface and to the card using the SD Bus interface.

The following figure shows the high level data flow through the Host controller.

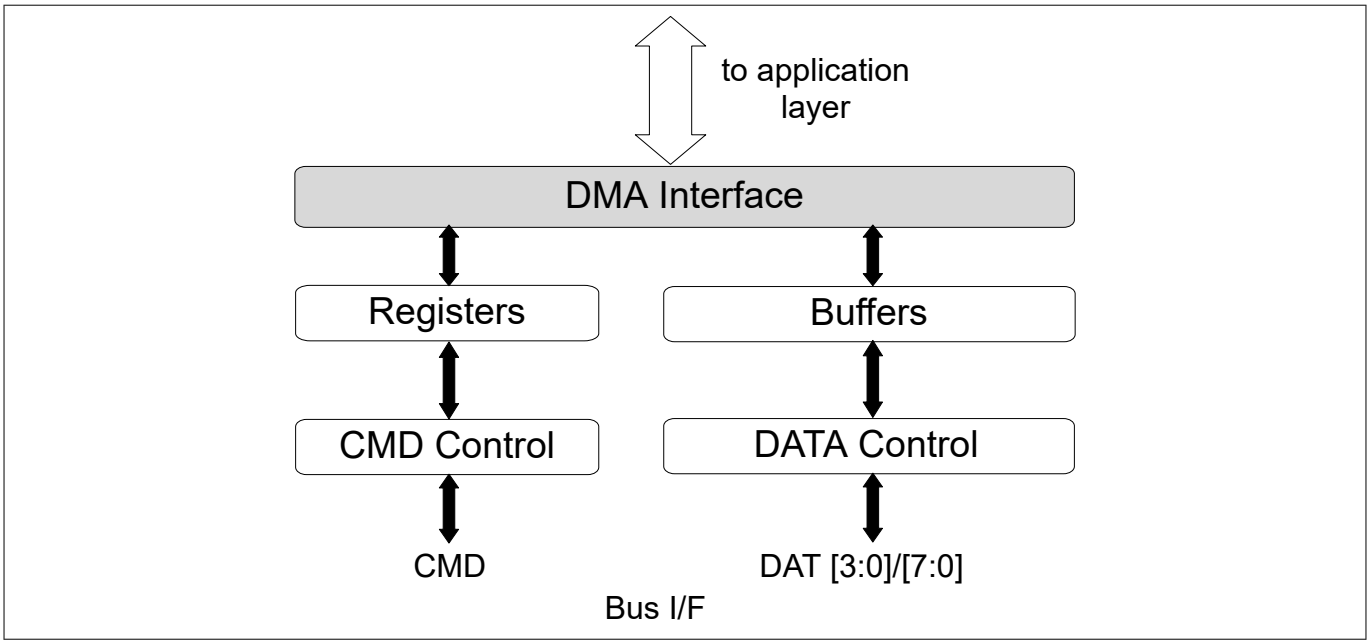


Figure 632 Host controller data flow

The host driver can transfer data using either a programmed I/O method or using any of the defined DMA methods.

Programmed Input/Output (PIO)

In the Programmed Input/Output (PIO) method of data transfer, data is transferred to the system memory by the host driver. It is done on the basis of block size, a maximum of 512 bytes. In a card write, the controller provides Buffer Write ready interrupt when it has space for one block. Based on this interrupt, the host driver must transfer one block of data from system memory to the SDMMC buffer through the Buffer data port register BUF_DATA. After that block is sent to the card by the controller, the same mechanism is used in the reverse direction in the case of a card read using the Buffer read ready interrupt. PIO mode is much slower and burdens the processor.

***Note**:*
- *It is not recommended to use the PIO mode for large transfers*
- *All bus accesses by the DMA engine are executed in user mode*

DMA

The following are the different DMA methods described in the SD standards:

1. SDMA: Single operation DMA is suitable for short data transfer because it requires address update at page boundary of system memory (256bit for Pflash and 64bit for Dflash). An interrupt is generated at every page boundary. Writing to the command register CMD triggers SDMA transfer. The Configuration registers for SDMA transfer are:
- System Memory Address or Block Count SDMASA.BLOCKCNT
- Data Block Size BLOCKSIZE.XFER_BLOCK_SIZE
- Size of Contiguous Buffer BLOCKSIZE.SDMA_BUF_BDARY

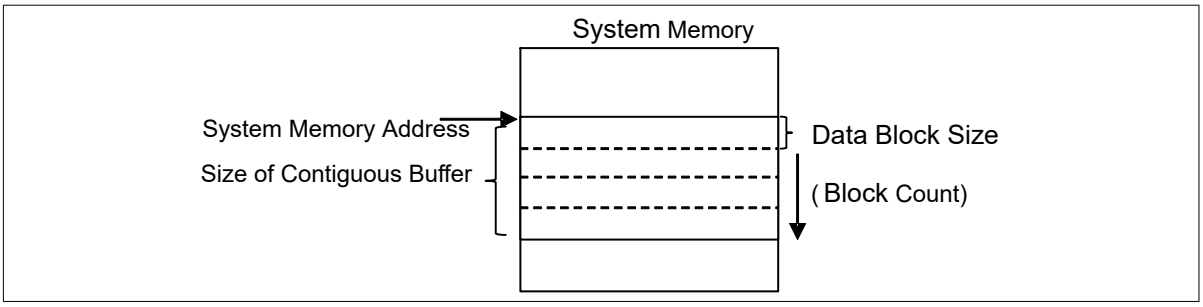


Figure 633 System Memory

2. ADMA2: Advanced DMA-2, supports multiple SDMA transfers allowing a single read or write operation across page boundaries. The configuration registers for ADMA2 transfers are:
   - Setting ADMA System Memory Address in ADMA system address register low for Descriptor table start address
   - Writing to the command register Command register triggers ADMA2 transfer
   - ADMA2 fetches one descriptor line and executes it in the Descriptor Table. This step is repeated until the end of the descriptor is found

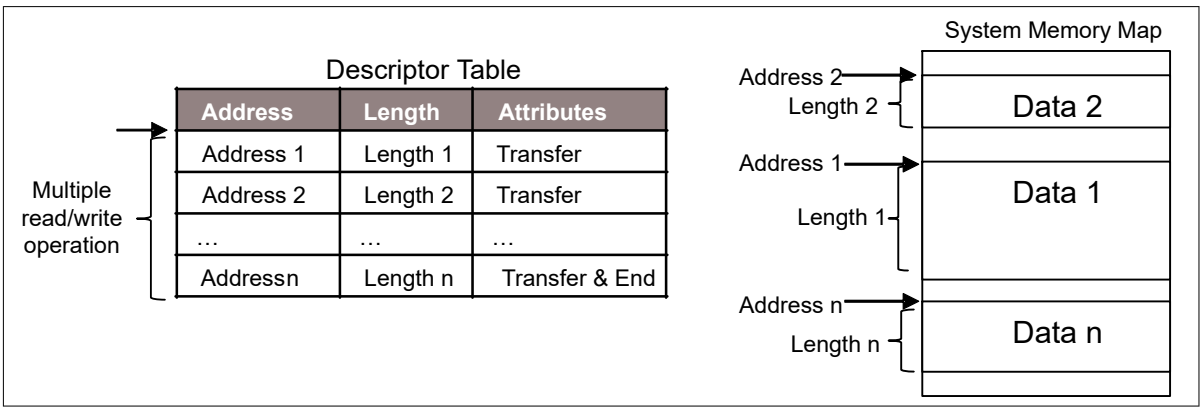


Figure 634 ADMA2 Descriptor table

3. ADMA3: Advanced DMA-3, supports multiple read or write commands to be programmed in the descriptor table. It performs multiple ADMA2 operations.
   a. Writing to the ADMA3 integrated descriptor address register ADMA_ID_LOW triggers the DMA operation as depicted in figure below

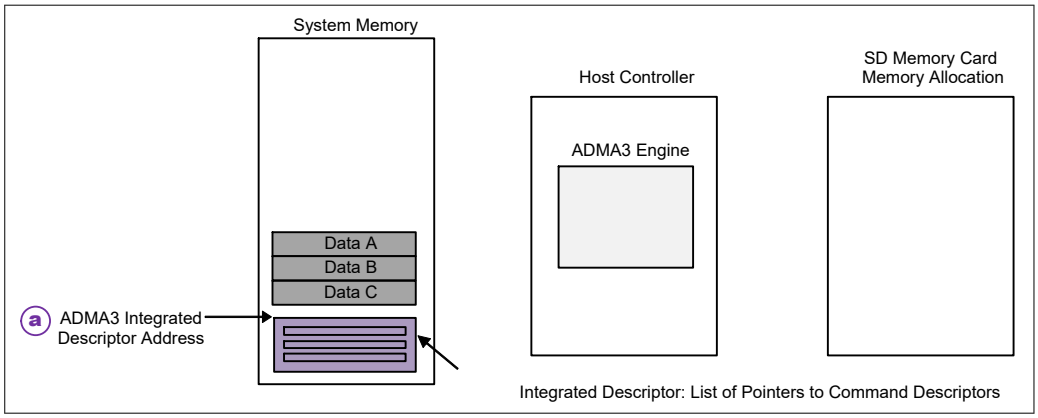


Figure 635 ADMA3 writing to Integrated Address Register

   b. ADMA3 fetches pointers one by one in the Integrated Descriptor and executes descriptors designated by the pointer as in figure below

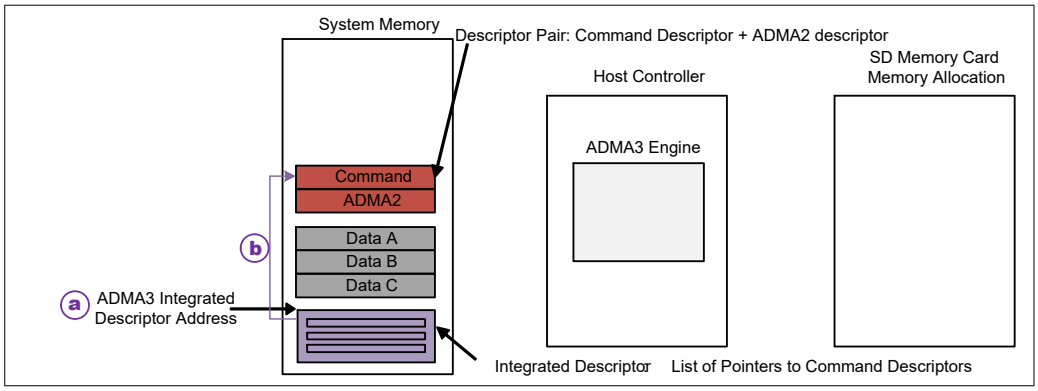


Figure 636 ADMA3 pointer fetch

c. ADMA3 writes the Command Descriptor contents to the Host Controller registers to issue a command and then executes the ADMA2 Descriptor

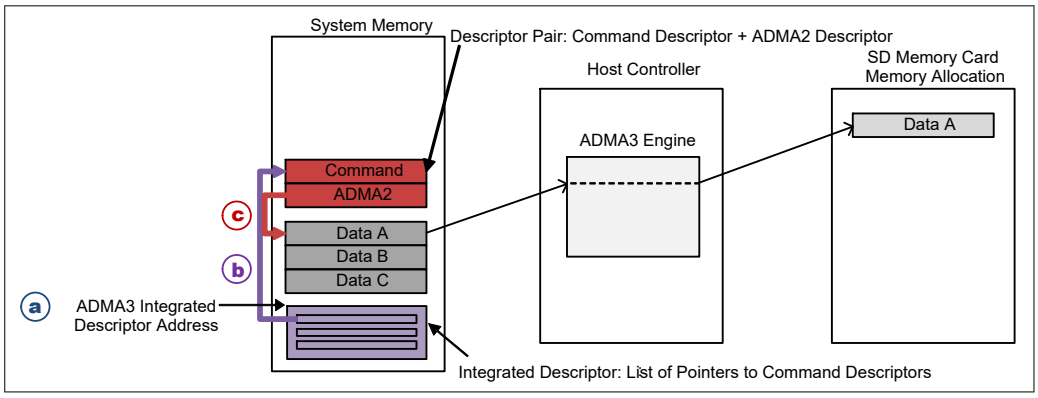


Figure 637 Issuing a Command

d. ADMA3 fetches the next pointer in the Integrated Descriptor and executes descriptors designated by the pointer

e. ADMA3 writes the next Command Descriptor contents to the Host Controller registers to issue a command and then executes the ADMA2 descriptor as shown below

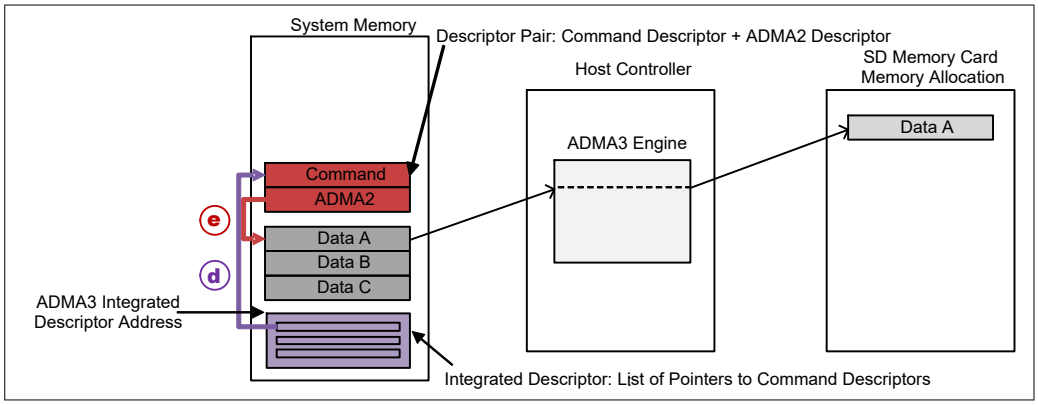


Figure 638 : Executing Next Descriptor

f. ADMA3 fetches the next pointer in the Integrated Descriptor and executes descriptors designated by the pointer until the end of the Command Descriptor

g. ADMA3 writes the end of the Command Descriptor contents to the Host Controller registers to issue a command and then executes the ADMA2 descriptor as shown below

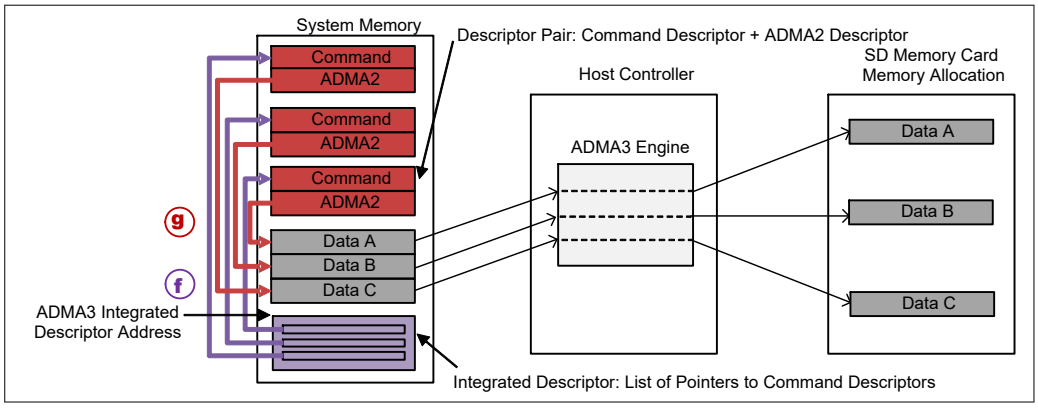


Figure 639 Executing until the end of Descriptor

***Note**: The control bits in the Block Gap Control(BGAP_CTRL) register are used to Stop and Restart a DMA operation*

***Note**: When HOST_CTRL2.HOST_VER4_EN = 1, SDMA uses ADMA_SA_LOW as system address register and SDMASA is reassigned to 32-bit block count*

29.3.5 SDMMC programming sequences

This section describes the SDMMC programming sequences.

29.3.5.1 Pin enabling

The direction control of the data and command pins(half duplex) are managed by the SDMMC peripheral through controlling the output enable signals. During start-up, the pins can be configured as inputs. The direction of the clock pin is set to 1 using the DIR bit in register PADCFGp_DRVCFG (p=00-15). Other pin configurations, such as pulls and driver strength, are also selected according to the application needs using the same register. For more details please refer to the Ports chapter.

29.3.5.2 Programming overview

A certain programming sequence is required for SDMMC and the card sub-system before starting data transfer with a connected card.

The following is the programming sequence required before starting the data transfer with the card:

1. Enable SDMMC peripheral clock,f_{SDMMC} using PERCCUCON1.SDMMCPERON (please refer to Clocking chapter)
2. Enabled clock for SDMMC module if not already enabled by configuring CLC.DISR after applying a module reset. Check if CLC.DISS!=0
3. Set up basic settings for SDMMC as described in Host Controller Setup Sequence. It involves settings such as the bus power voltage level for the card, timeout counter value, setting clock generation parameters. Different setup sequences for SD card (“Host controller setup sequence for an SD interface”) and eMMC card (“Host controller setup sequence for an eMMC interface”) are required
4. Enable other input clocks like the card clock that are available in SDMMC. Further details are as discussed in the programming sequence in Host controller clock setup sequence. At this stage, SDMMC is ready to communicate with the card
5. Identify the type of card that is connected to SDMMC as explained in the programming sequence in “SD Card Interface Detection”[30]. If SDMMC is connected to an eMMC card, then programming sequence described in “eMMC Card Interface Setup”[31] must be executed. This is a simplified sequence especially for an eMMC card as the type of card is always known. Both the programming sequences provide power and clock to the card
6. After powering up the card, the card is initialized. Appropriate sequence as shown below must to be executed based on type of the connected card

Table 1228 Card initialization sequence for different cards

Type of card	Card initialization sequence
SD	SD card initialization and identification[1]
eMMC	Initializing and identifying an eMMC card[2]

1) refer to SD Specifications Part A2 Host Controller document in References section for the sequence
2) refer to JEDEC eMMC 5.1 Specification - JESD84-B51 document in References section for the sequence

7. As both host controller and card are initialized, the host can start sending commands to the card to perform the data transfer. Control and data commands are two types of commands that can be issued to

30 refer to SD Specifications Part A2 Host Controller document in References section for the sequence
31 refer to SD Specifications Part A2 Host Controller document in References section for the sequence

the card. The control command is used to read or write any register in the card. The data command is used for writing or reading data to or from the card.

a. Data transfer can be aborted using abort command

b. If errors are detected during the transfer, recovery should be done using the Error Recovery sequence

Table 1229 Command sequence for SD and eMMC cards

Type of card	Control and data command sequences for SD and eMMC cards	
SD/eMMC	Control CMD	Issuing CMD without data transfer[1)]
	Data CMD	Issuing CMD with data transfer[2)]

1) refer to SD Specifications Part A2 Host Controller document in References section for the sequence

2) refer to SD Specifications Part A2 Host Controller document in References section for the sequence

The figure below shows the programming overview. Subsequent sections describe the steps in detail.

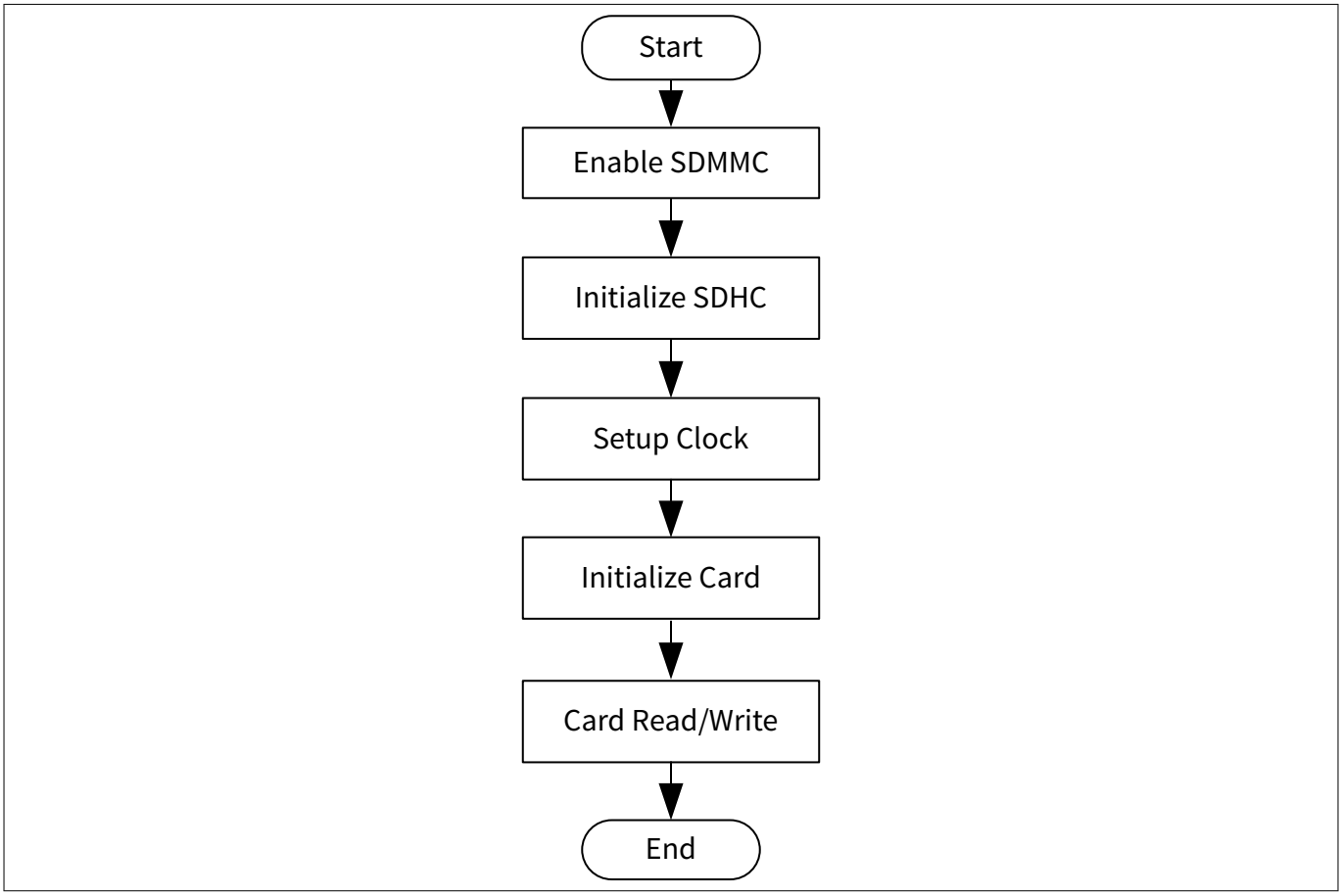


Figure 640 Programming sequence overview

Related information

Host controller setup sequence for an SD interface on page 5208
Host controller setup sequence for an eMMC interface on page 5208
References on page 5287

29.3.5.3 Host controller setup sequence

This section discusses the host controller setup sequence for SD device and an eMMC device.

29.3.5.3.1 Host controller setup sequence for an SD interface

The host controller setup sequence for an SD interface involves some basic settings for SDMMC, such as the bus power voltage level for the card, timeout counter value, and setting clock generation parameters.

The following figure shows the host controller setup sequence. The register parameters set in this sequence retain their values irrespective of bus speed mode changes.

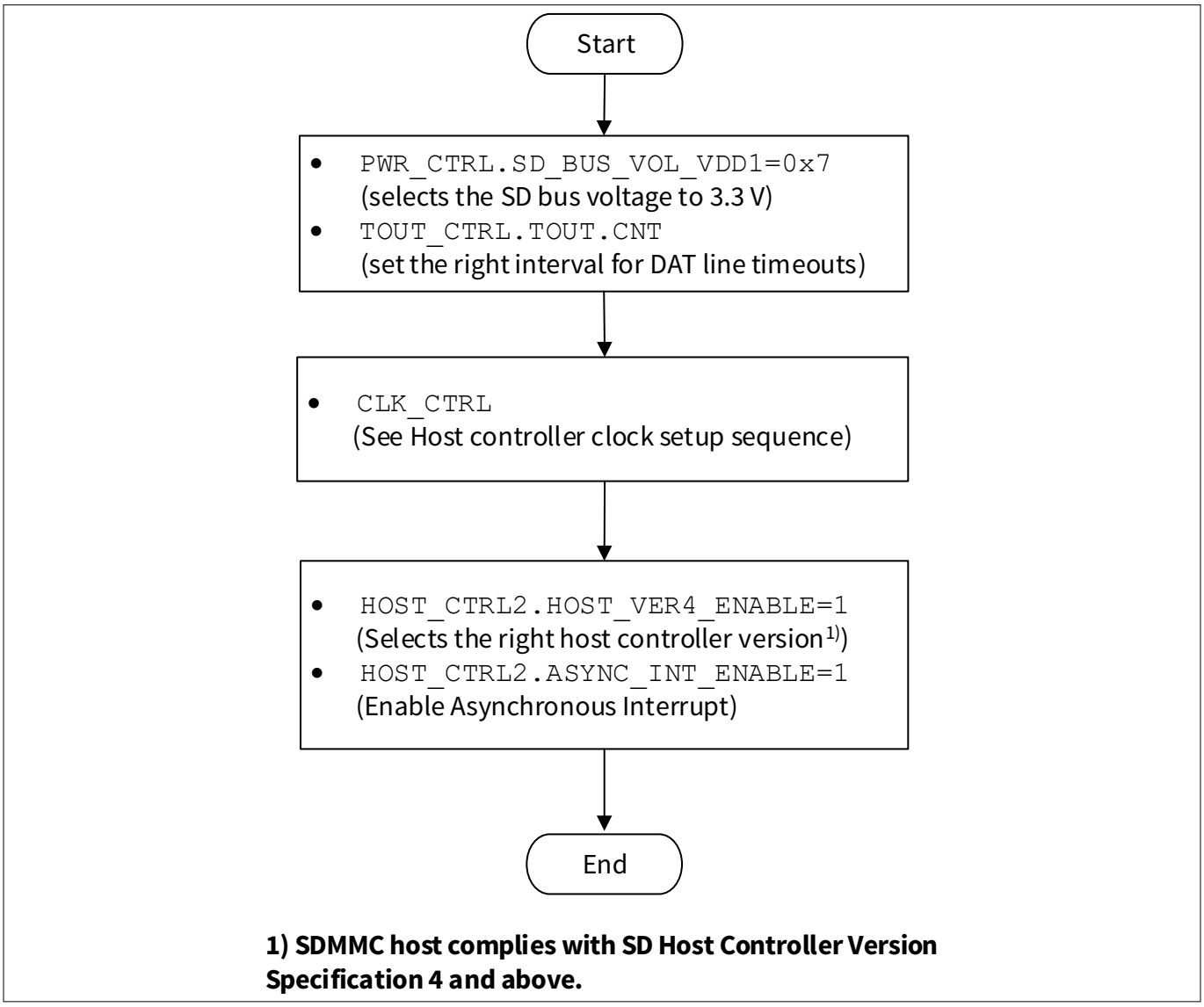


Figure 641 Host controller setup sequence for SD interface

29.3.5.3.2 Host controller setup sequence for an eMMC interface

The host controller setup sequence for an eMMC device requires some basic settings for SDMMC. The procedure involves setting certain parameters common to both version 3 and 4, clock control kernel register, and version 4 parameters.

The following figure shows the host controller setup sequence for an eMMC device. In addition to completing the programming sequence you must set the parameter EMMC_CRTL.CARD_IS_EMMC = 1.

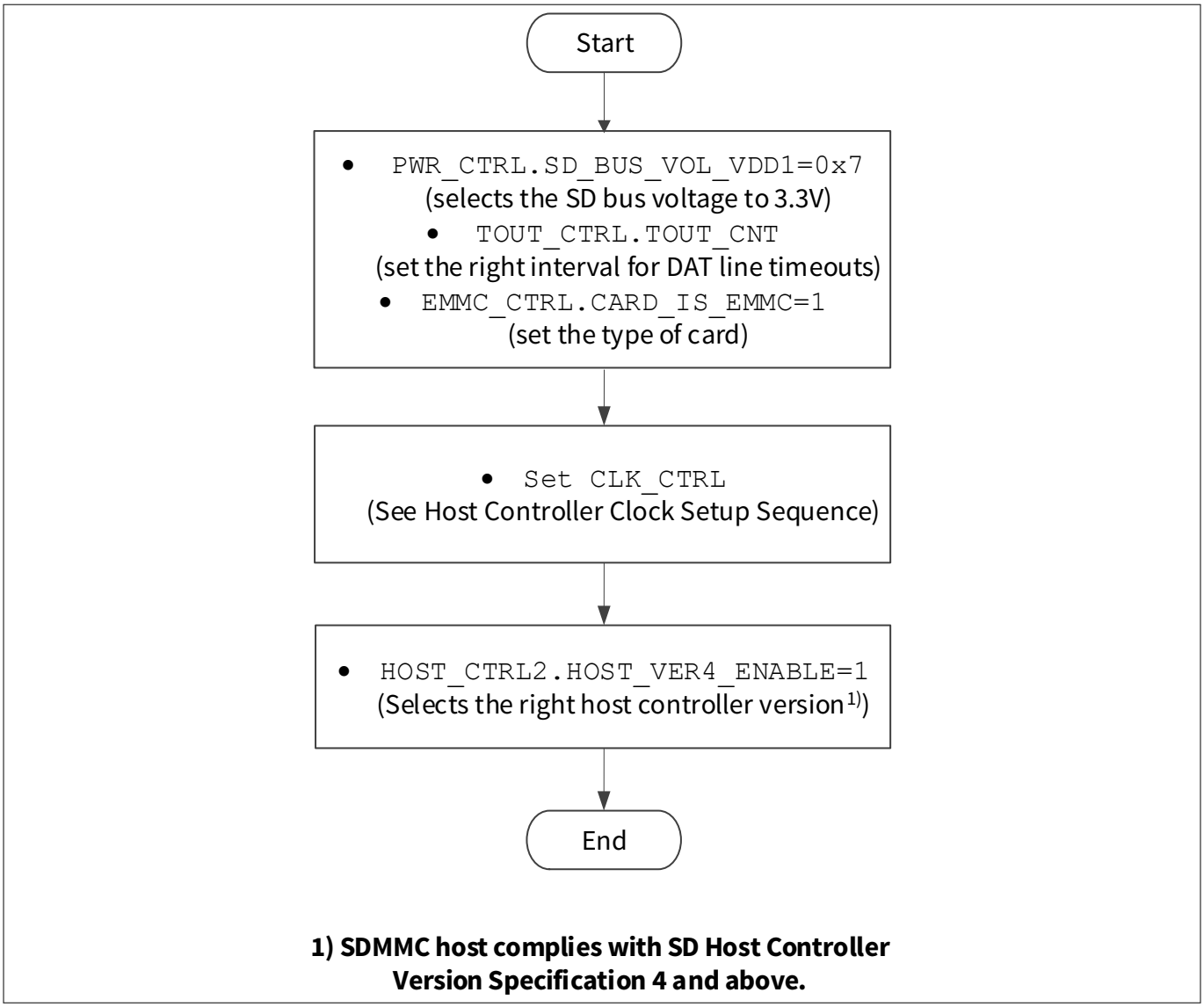


Figure 642 Host controller setup sequence for eMMC interface

29.3.5.4 Clock control

This section discusses the programming sequence for setting up the internal clocks and card clock provided to SDMMC, for supplying and setting up a clock provided to the card, and for changing the card clock frequency.

This section includes the following sequences:

- Host controller clock setup sequence
- Card clock supply and stop sequence

29.3.5.4.1 Host controller clock setup sequence

The SDMMC requires a master domain clock, slave domain clock and peripheral clock (f_{SDMMC}) from the system. All the input clocks are asynchronous to each other. Based on the controls available for enabling clocks required for SDMMC, they are categorized into two types, namely Internal clock and Card clock.

The following figure shows the sequence for setting up clocks for SDMMC.

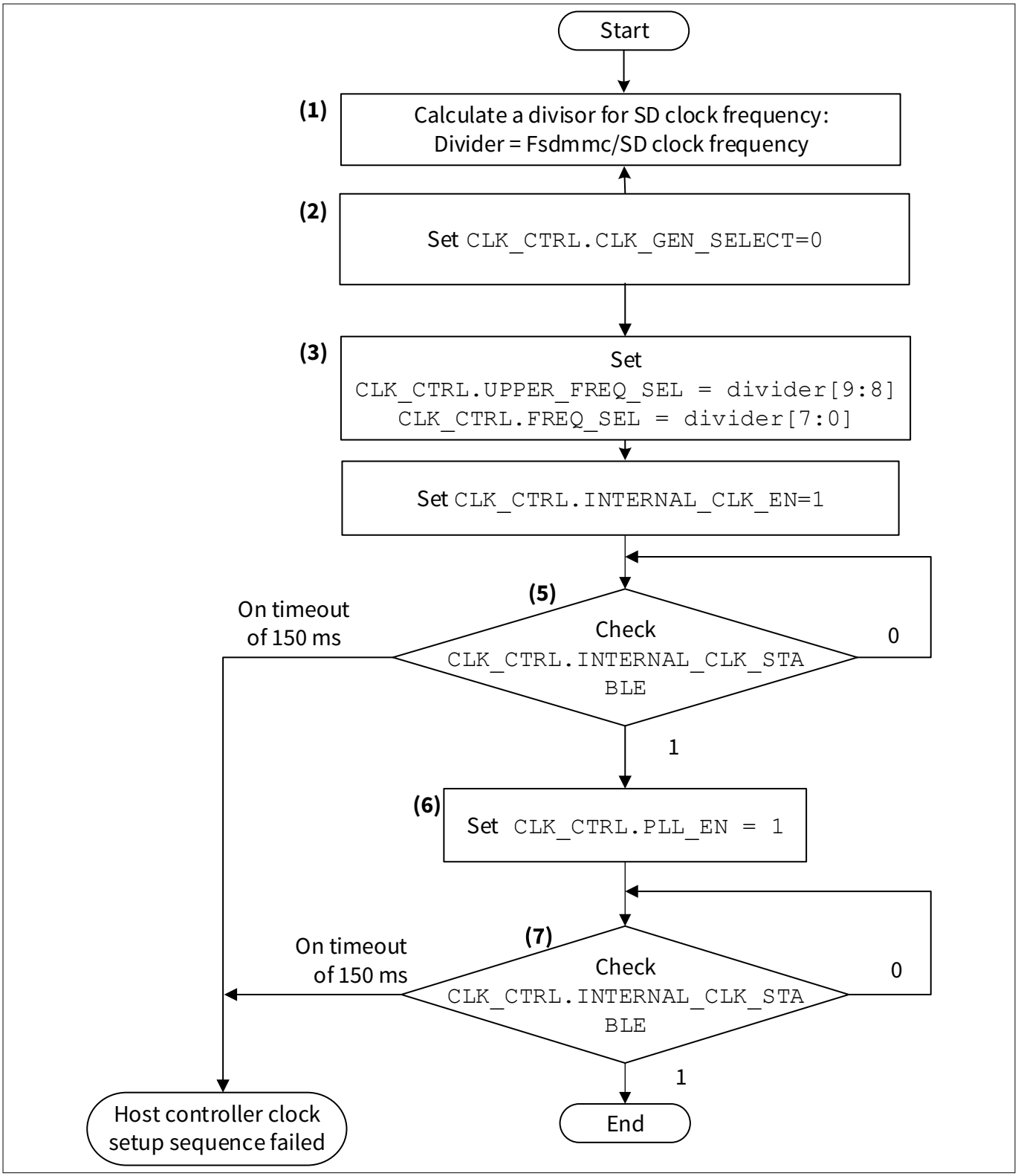


Figure 643 Host controller clock setup sequence

29.3.5.4.2 Card clock supply and stop sequence

The following figure shows the flow chart for stopping the clock to card. The procedure outlined in the figure applies to both SD and eMMC cards.

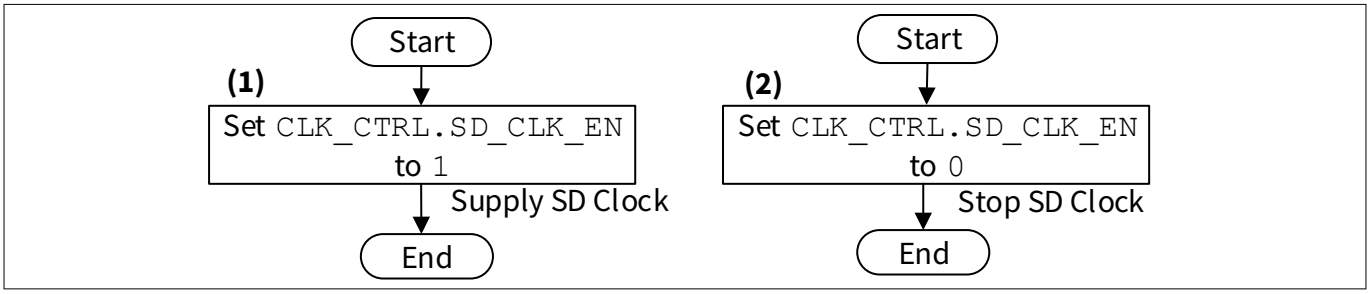


Figure 644 Card clock supply and stop sequence

29.4 Registers

29.4.1 Register overview - access mode glossary

Table 1230 Register overview - access mode glossary

Keyword	Description
E	Access protection using PROT register PROTE.
SE	Access protection using PROT register PROTSE.
APU-P	Protection group consisting of registers ACCEN_WRA, ACCEN_WRB, ACCEN_RDA, ACCEN_RDB, ACCEN_VM, ACCEN_PRS.
P	Access protection using *APU-P* registers.
BE	Always returns a Bus Error.
U	No access restrictions.
PROT	Access restrictions as defined in the PROT register access rules.

29.4.2 Registers overview - SDMMC (ascending offset address)

Table 1231 Registers overview - SDMMC (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CLC	Clock control register	000_H	P	P, SV, E	Application Reset	5214
OCS	OCDS control and status register	004_H	P	SV, P	Debug Reset	5215
ID	Module identification register	008_H	P	BE	PowerOn Reset	5215
RST_CTRLA	Reset control register A	$00C_H$	P	P, SV, E	Application Reset	5216
RST_CTRLB	Reset control register B	010_H	P	P, SV, E	Application Reset	5216

(table continues...)

Table 1231 (continued) Registers overview - SDMMC (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
RST_STAT	Reset status register	014_H	P	BE	Application Reset	5217
PROTE	PROT register endinit	020_H	U	SV, PROT	Application Reset	5218
PROTSE	PROT register safe endinit	024_H	U	SV, PROT	Application Reset	5219
ACCEN_WRA	Write access enable register A	040_H	U	SE, SV	Application Reset	5221
ACCEN_WRB	Write access enable register B	044_H	U	SE, SV	Application Reset	5221
ACCEN_RDA	Read access enable register A	048_H	U	SE, SV	Application Reset	5222
ACCEN_RDB	Read access enable register B	$04C_H$	U	SE, SV	Application Reset	5222
ACCEN_VM	VM access enable register	050_H	U	SE, SV	Application Reset	5223
ACCEN_PRS	PRS access enable register	054_H	U	SE, SV	Application Reset	5223
SDMASA	SDMA system address register	400_H	P	P	Kernel Reset	5224
BLOCKSIZE	Block size register	404_H	P	P	Kernel Reset	5225
BLOCKCOUNT	16-bit block count register	406_H	P	P	Kernel Reset	5226
ARGUMENT	Argument register	408_H	P	P	Kernel Reset	5227
XFER_MODE	Transfer mode register	$40C_H$	P	P	Kernel Reset	5227
CMD	Command register	$40E_H$	P	P	Kernel Reset	5230
RESP01	Response register 01	410_H	P	BE	Kernel Reset	5232
RESP23	Response register 23	414_H	P	BE	Kernel Reset	5232
RESP45	Response register 45	418_H	P	BE	Kernel Reset	5233
RESP67	Response register 67	$41C_H$	P	BE	Kernel Reset	5233
BUF_DATA	Buffer data port register	420_H	P	P	Kernel Reset	5234
PSTATE_REG	Present state register	424_H	P	BE	Kernel Reset	5234
HOST_CTRL1	Host control 1 register	428_H	P	P	Kernel Reset	5237
PWR_CTRL	Power control register	429_H	P	P	Kernel Reset	5239
BGAP_CTRL	Block gap control register	$42A_H$	P	P	Kernel Reset	5240
WUP_CTRL	Wakeup control register	$42B_H$	P	P	Kernel Reset	5241
CLK_CTRL	Clock control kernel register	$42C_H$	P	P	Kernel Reset	5242
TOUT_CTRL	Timeout control register	$42E_H$	P	P	Kernel Reset	5243

(table continues...)

Table 1231 (continued) Registers overview - SDMMC (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
SW_RST	Software reset register	$42F_H$	P	P	Kernel Reset	5244
NORMAL_INT_STAT	Normal interrupt status register	430_H	P	P	Kernel Reset	5246
ERROR_INT_STAT	Error interrupt status register	432_H	P	P	Kernel Reset	5248
NORMAL_INT_STAT_EN	Normal interrupt status enable register	434_H	P	P	Kernel Reset	5252
ERROR_INT_STAT_EN	Error interrupt status enable register	436_H	P	P	Kernel Reset	5254
NORMAL_INT_SIGNAL_EN	Normal interrupt signal enable register	438_H	P	P	Kernel Reset	5256
ERROR_INT_SIGNAL_EN	Error interrupt signal enable register	$43A_H$	P	P	Kernel Reset	5258
AUTO_CMD_STAT	Auto CMD status register	$43C_H$	P	P	Kernel Reset	5260
HOST_CTRL2	Host control 2 register	$43E_H$	P	P	Kernel Reset	5261
CAPABILITIES1	Capabilities 1 register 0 to 31	440_H	P	BE	Kernel Reset	5264
CAPABILITIES2	Capabilities register 32 to 63	444_H	P	BE	Kernel Reset	5267
CURR_CAPABILITIES1	Maximum current capabilities register 0 to 31	448_H	P	BE	Kernel Reset	5270
CURR_CAPABILITIES2	Maximum current capabilities register 32 to 63	$44C_H$	P	BE	Kernel Reset	5271
FORCE_AUTO_CMD_STAT	Force event register for auto CMD error status register	450_H	P	P	Kernel Reset	5271
FORCE_ERROR_INT_STAT	Force event register for error interrupt status	452_H	P	P	Kernel Reset	5273
ADMA_ERR_STAT	ADMA error status register	454_H	P	BE	Kernel Reset	5274
ADMA_SA_LOW	ADMA system address register low	458_H	P	P	Kernel Reset	5275
PRESET_INIT	Preset value for initialization	460_H	P	BE	Kernel Reset	5276
PRESET_DS	Preset value for default speed	462_H	P	BE	Kernel Reset	5277
PRESET_HS	Preset value for high speed	464_H	P	BE	Kernel Reset	5278
ADMA_ID_LOW	ADMA3 integrated descriptor address register - low	478_H	P	P	Kernel Reset	5278

(table continues...)

Table 1231 (continued) Registers overview - SDMMC (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
P_VENDOR_SPECIFIC_AREA	Pointer for vendor specific area 1	$4E8_H$	P	BE	Kernel Reset	5279
P_VENDOR2_SPECIFIC_AREA	Pointer for vendor specific area 2	$4EA_H$	P	BE	Kernel Reset	5279
SLOT_INTR_STATUS	Slot interrupt status register	$4FC_H$	P	BE	Kernel Reset	5280
HOST_CNTRL_VERS	Host controller version	$4FE_H$	P	BE	Kernel Reset	5280
MSHC_VER_ID	MSHC version	580_H	P	BE	Kernel Reset	5281
MSHC_VER_TYPE	MSHC version type	584_H	P	BE	Kernel Reset	5281
MBIU_CTRL	DMA burst control register	590_H	P	P	Kernel Reset	5282
EMMC_CTRL	eMMC control register	$5AC_H$	P	P	Kernel Reset	5283
BOOT_CTRL	eMMC boot control register	$5AE_H$	P	P	Kernel Reset	5284
EMBEDDED_CTRL	Embedded control register	680_H	P	P	Kernel Reset	5285

29.4.3 Clock control register

The Clock Control Register CLC allows the programmer to adapt the functionality and power consumption of the module to the requirements of the application.

Register CLC controls the module clock signal and the reactivity to the sleep signal.

CLC Offset address: 000_H

Clock control register Application Reset value: $0000\ 0003_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												EDIS	0	DISS	DISR
r												rw	r	rh	rw

Field	Bits	Type	Description
DISR	0	rw	**Module Disable Request Bit** Used for enable/disable control of the module. 0_B On request: enable the module clock 1_B Off request: stop the module clock
DISS	1	rh	**Module Disable Status Bit** 0_B Module clock is enabled 1_B Off: module is not clocked

(table continues...)

(continued)

Field	Bits	Type	Description
EDIS	3	rw	**Sleep Mode Enable Control** Used to control the module's reaction to sleep mode. 0_B Sleep mode request is enabled and functional 1_B Module disregards the sleep mode control signal
0	2, 31:4	r	**Reserved** Read as 0; should be written with 0.

29.4.4 OCDS control and status register

OCS Offset address: 004_H

OCDS control and status register Debug Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															
r															

Field	Bits	Type	Description
0	31:0	r	**Reserved** Read as 0; should be written with 0.

29.4.5 Module identification register

ID Offset address: 008_H

Module identification register PowerOn Reset value: $00EA\ C002_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MOD_NUM															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MOD_TYPE								MOD_REV							
r								r							

Field	Bits	Type	Description
MOD_REV	7:0	r	**Module Revision** Indicates the revision number of the implementation.
MOD_TYPE	15:8	r	**Module Type** This internal marker is fixed to $C0_H$.

(table continues...)

(continued)

Field	Bits	Type	Description
MOD_NUM	31:16	r	**Module Number** Indicates the module identification number

29.4.6 Reset control register A

RST_CTRLA Offset address: $00C_H$

Reset control register A Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				GRSTEN3	GRSTEN2	GRSTEN1	GRSTEN0	0							KRST
r				rw	rw	rw	rw	r							rwh

Field	Bits	Type	Description
KRST	0	rwh	**Kernel Reset** Request a kernel reset. The requested reset is executed if CTRLA.KRST=1 and CTRLB.KRST=1. KRST is cleared after the kernel reset was executed. 0_B No action 1_B A kernel reset was requested
GRSTENx (x=0-3)	x+8	rw	**Enable for Global Module Reset Group x** 0_B Global module reset group x does not have any effect 1_B Global module reset group x results in a kernel reset
0	7:1, 31:12	r	**Reserved** Read as 0; should be written with 0.

29.4.7 Reset control register B

RST_CTRLB Offset address: 010_H

Reset control register B Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
STATCLR								0							
w								r							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															KRST
r															rwh

Field	Bits	Type	Description
KRST	0	rwh	**Kernel Reset** Request a kernel reset. The requested reset is executed if CTRLA.KRST=1 and CTRLB.KRST=1. KRST is cleared after the kernel reset was executed. 0_B No action 1_B A kernel reset was requested
STATCLR	31	w	**Kernel Reset Status Clear** Clears all status bits in RST_STAT when 1 is written. Read will return 0. 0_B No action 1_B Write with ´1´ clears bits STAT.GRSTx and bit STAT.KRST .
0	30:1	r	**Reserved** Read as 0; should be written with 0.

29.4.8 Reset status register

The reset status register contains the status bits for kernel reset (KRST) and the global module reset groups (GRSTx).

RST_STAT Offset address: 014_H

Reset status register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				GRST3	GRST2	GRST1	GRST0	0							KRST
r				rh	rh	rh	rh	r							rh

Field	Bits	Type	Description
KRST	0	rh	**Kernel reset status** Indicates an executed kernel reset. RST_STAT.KRST is set after the execution of a kernel reset in the same clock cycle in which the reset bits are cleared. Clear KRST by setting bit STATCLR in register RST_CTRLB.
GRSTx (x=0-3)	x+8	rh	**Status for global module reset group x** 0_B Reset was not triggered by global reset group x 1_B Reset was triggered by global reset group x
0	7:1, 31:12	r	**Reserved** Read as 0.

29.4.9 PROT register endinit

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner or CS master to update the PROT state as described in the PROT mechanism.

The PROTE register controls lock / unlock of the local Endinit (E) protected control registers.

PROTE Offset address: 020_H

PROT register endinit Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OWEN	ODEF	TAGID						PRSEN	PRS			VMEN	VM		
w	rw	rw						rw	rw			rw	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												SWEN	STATE		
r												w	rwh		

Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
VM	18:16	rw	**Virtual machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual machine definition enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.

(table continues...)

(continued)

Field	Bits	Type	Description
PRSEN	23	rw	**Protection set definition enable for PROT owner** 0_B **Disable**: PRS is not part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG-ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; should be written with 0.

Table 1232 Access mode restrictions of PROTE sorted by descending priority

Mode name	Access mode		Description
write 1 to .**SWEN**	rwh	STATE	
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

29.4.10 PROT register safe endinit

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner or CS master to update the PROT state as described in the PROT mechanism.

The PROTSE register controls lock / unlock of the local Safe Endinit (SE) protected control registers.

PROTSE Offset address: 024_H

PROT register safe endinit Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OWEN	ODEF	TAGID						PRSEN	PRS			VMEN	VM		
w	rw	rw						rw	rw			rw	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												SWEN	STATE		
r												w	rwh		

Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
VM	18:16	rw	**Virtual machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual machine definition enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.
PRSEN	23	rw	**Protection set definition enable for PROT owner** 0_B **Disable**: PRS is not part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG-ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; should be written with 0.

Table 1233 Access mode restrictions of PROTSE sorted by descending priority

Mode name	Access mode		Description
write 1 to .**SWEN**	rwh	STATE	
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

29.4.11 Write access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

ACCEN_WRA Offset address: 040_H

Write access enable register A Application Reset value: $1000\ 0003_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	**EN30**	**EN29**	**EN28**	**EN27**	**EN26**	**EN25**	**EN24**	**EN23**	**EN22**	**EN21**	**EN20**	**EN19**	**EN18**	**EN17**	**EN16**
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	**EN14**	**EN13**	**EN12**	**EN11**	**EN10**	**EN09**	**EN08**	**EN07**	**EN06**	**EN05**	**EN04**	**EN03**	**EN02**	**EN01**	**EN00**
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access

29.4.12 Write access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

ACCEN_WRB Offset address: 044_H

Write access enable register B Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								**EN39**	**EN38**	**EN37**	**EN36**	**EN35**	**EN34**	**EN33**	**EN32**
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access
0	31:8	r	**Reserved** Read as 0; should be written with 0.

29.4.13 Read access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

ACCEN_RDA Offset address: 048_H

Read access enable register A Application Reset value: FFFF $FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access

29.4.14 Read access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

ACCEN_RDB Offset address: $04C_H$

Read access enable register B Application Reset value: 0000 $00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access
0	31:8	r	**Reserved** Read as 0; should be written with 0.

29.4.15 VM access enable register

This register defines which virtual machine encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each VMx.

ACCEN_VM Offset address: 050_H

VM access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for VM ID q** This bit enables read access to the access protected region for transactions with the VM ID q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for VM ID q** This bit enables write access to the access protected region for transactions with the VM ID q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

29.4.16 PRS access enable register

This register defines which protection register set encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each PRSx.

ACCEN_PRS Offset address: 054_H

PRS access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for PRS q** This bit enables read access to the access protected region for transactions with the PRS q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for PRS q** This bit enables write access to the access protected region for transactions with the PRS q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

29.4.17 SDMA system address register

This register is used to configure 32-bit Block Count or SDMA System Address based on the Host Version 4 Enable bit in the Host Control 2 register. This register is applicable for both SD and eMMC modes.

SDMASA Offset address: 400_H

SDMA system address register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BLOCKCNT_SDMASA															
rwh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BLOCKCNT_SDMASA															
rwh															

Field	Bits	Type	Description
BLOCKCNT_SDMASA	31:0	rwh	**32-bit Block Count (SDMA System Address)** • SDMA System Address (Host Version 4 Enable = 0) - This register contains the system memory address for SDMA transfer in the 32-bit addressing mode. When the Host Controller stops an SDMA transfer, this register shall point to the system address of the next contiguous data position. It can be accessed only if no transaction is executing. Reading this register during data transfers may return invalid value. • 32-bit Block Count (Host Version 4 Enable = 1) - From Host Controller Version 4.10, this register is redefined as 32-bit Block Count. The Host Controller decrements the block count of this register for every block transfer and the data transfer stops when the count reaches zero. This register should be accessed when no transaction is executing. Reading this register during data transfers may return invalid value. • FFFF_FFFFh - 4G - 1 block • ... - ... • 0000_0002h - 2 blocks • 0000_0001h - 1 block • 0000_0000h - Stop Count ***Note**: For Host Version 4 Enable = 0, Host driver shall not program system address in this register while operating in ADMA mode. It should be programmed in ADMA System Address register.* ***Note**: For Host Version 4 Enable = 0, Host driver shall program 32-bit block count in this register when Auto CMD23 is enabled for non-DMA and ADMA mode.*

29.4.18 Block size register

This register is used to configure SDMA buffer boundary and number of bytes in a data block. This register is applicable for both SD and eMMC modes.

BLOCKSIZE Offset address: 404_H

Block size register Kernel Reset value: 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD_BLOCKSIZE15	SDMA_BUF_BDARY			XFER_BLOCK_SIZE											
r	rw			rw											

Field	Bits	Type	Description
XFER_BLOCK_SIZE	11:0	rw	**Transfer Block Size** This register specifies the block size of data transfers. In case of memory, it shall be set up to 512 bytes. It can be accessed only if no transaction is executing. Read operations during transfers may return an invalid value, and write operations shall be ignored. 000_H No Data Transfer 001_H 1 Byte 002_H 2 Bytes … 200_H 512 Bytes 201_H reserved … FFF_H reserved
SDMA_BUF_BDARY	14:12	rw	**SDMA Buffer Boundary** These bits specify the size of contiguous buffer in system memory. The SDMA transfer shall wait at every boundary specified by these fields and the Host Controller generates the DMA interrupt to request the Host Driver to update the SDMA System Address register. 000_B 4K Bytes 001_B 8K Bytes 010_B 16K Bytes 011_B 32K Bytes 100_B 64K Bytes 101_B 128K Bytes 110_B 256K Bytes 111_B 512K Bytes
RSVD_BLOCKSIZE15	15	r	**Reserved** This bit is reserved. It will always return 0.

29.4.19 16-bit block count register

This register is used to configure the number of data blocks. This register is applicable for both SD and eMMC modes.

BLOCKCOUNT Offset address: 406_H

16-bit block count register Kernel Reset value: 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BLOCK_CNT															
rwh															

Field	Bits	Type	Description
BLOCK_CNT	15:0	rwh	**16-bit Block Count** • If Host Version 4 Enable is set 0 or 16-bit Block Count register is set to non-zero, 16-bit Block Count register is selected. • If Host Version 4 Enable is set 1 and 16-bit Block Count register is set to zero, 32-bit Block Count register is selected. ***Note**: For Host Version 4 Enable = 0, this register should be set to 0000h before programming 32-bit block count register when Auto CMD23 is enabled for non-DMA and ADMA mode.* 0000_H Stop Count 0001_H 1 Block 0002_H 2 Blocks … $FFFF_H$ 65535 Blocks

29.4.20 Argument register

This register is used to configure the SD/eMMC command argument.

ARGUMENT Offset address: 408_H

Argument register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ARGUMENT															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ARGUMENT															
rw															

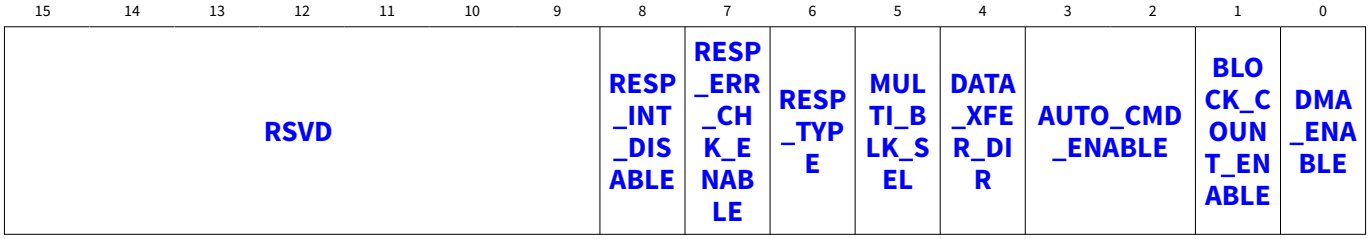
Field	Bits	Type	Description
ARGUMENT	31:0	rw	**Command Argument** The SD/eMMC command argument is specified as bit 39-8 of the command format.

29.4.21 Transfer mode register

This register is used to control the operation of data transfers for an SD/eMMC mode. The Host driver shall set this register before issuing a command that transfers data.

XFER_MODE Offset address: $40C_H$

Transfer mode register Kernel Reset value: 0000_H



Field	Bits	Type	Description
DMA_ENABLE	0	rw	**DMA Enable** This bit enables the DMA functionality. If this bit is set to 1, a DMA operation begins when Host Driver writes to Command register. You can select one of the DMA modes by using DMA Select in the Host Control 1 register. The DMA supports only transfers of 32bit data. The addresses have to be 32bit aligned for 32bit addressing. 0_B No data transfer or Non DMA data transfer 1_B DMA Data transfer
BLOCK_COUNT_ENABLE	1	rw	**Block Count Enable** This bit is used to enable the Block Count register, which is relevant for multiple block transfers. When this bit is 0, the Block Count register is disabled, which is useful in executing an infinite transfer. Host Driver should set this bit to 0 when ADMA is used. 0_B Disable 1_B Enable
AUTO_CMD_ENABLE	3:2	rw	**Auto Command Enable** This field determines use of auto command functions. ***Note**: In SDIO, this field should be set as 00b (Auto Command Disabled).* 00_B Auto Command Disabled 01_B Auto CMD12 Enable 10_B Auto CMD23 Enable 11_B Auto CMD Auto Select
DATA_XFER_DIR	4	rw	**Data Transfer Direction Select** This bit defines the direction of DAT line data transfers. This bit is set to 1 by the Host Driver to transfer data from the SD/eMMC card to the Host Controller and it is set to 0 for all other commands. 0_B Write (Host to Card) 1_B Read (Card to Host)

(table continues...)

(continued)

Field	Bits	Type	Description
MULTI_BLK_SEL	5	rw	**Multi/Single Block Select** This bit is set when issuing multiple-block transfer commands using DAT line. If this bit is set to 0, it is not necessary to set Block Count register. 0_B Write (Single Block) 1_B Read (Multiple Block)
RESP_TYPE	6	rw	**Response Type R1/R5** When response error check is enabled, this bit selects either R1 or R5 response types. Error statuses checked in R1: • OUT_OF_RANGE • ADDRESS_ERROR • BLOCK_LEN_ERROR • WP_VIOLATION • CARD_IS_LOCKED • COM_CRC_ERROR • CARD_ECC_FAILED • CC_ERROR • ERROR Response Flags checked in R5: • COM_CRC_ERROR • ERROR • FUNCTION_NUMBER • OUT_OF_RANGE 0_B R1 (Memory) 1_B R5 (SDIO)
RESP_ERR_CHK_ENABLE	7	rw	**Response Error Check Enable** Host Controller supports response check function to avoid overhead of response error check by Host driver. Only R1 and R5 can be checked by the controller. If Host Controller checks the response error, set this bit to 1 and set Response Interrupt Disable to 1. If an error is detected, Response Error interrupt is generated in the Error Interrupt Status register. ***Note**: Response error check should not be enabled for any reponse type other than R1 and R5.* ***Note**: Response check should not be enabled for tuning command.* 0_B Response Error Check is disabled 1_B Response Error Check is enabled

(table continues...)

(continued)

Field	Bits	Type	Description
RESP_INT_DISABLE	8	rw	**Response Interrupt Disable** Host Controller supports response check function to avoid overhead of response error check by Host driver. Only R1 and R5 can be checked by the controller. If Host Driver checks response error, set this bit to 0 and wait for Command Complete Interrupt and then check the response register. If Host Controller checks response error, set this bit to 1 and set Response Error Check Enable to 1. Command Complete Interrupt is disabled by this bit regardless of Command Complete Signal Enable. ***Note**: While carrying out tuning (When Execute Tuning bit in Host Control2 register is set), command complete interrupt is not generated irrespective of Response Interrupt Disable setting.* 0_B Response Interrupt enabled 1_B Response Interrupt is disabled
RSVD	15:9	r	**Reserved** These bits are reserved. They will always return 0.

29.4.22 Command register

This register is used to provide the information related to command and response packet. This register is applicable for an SD/eMMC mode.

CMD Offset address: $40E_H$

Command register Kernel Reset value: 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD		CMD_INDEX						CMD_TYPE		DATA_PRESENT_SEL	CMD_IDX_CHK_ENABLE	CMD_CRC_CHK_ENABLE	SUB_CMD_FLAG	RESP_TYPE_SELECT	
r		rw						rw		rw	rw	rw	rw	rw	

Field	Bits	Type	Description
RESP_TYPE_SELECT	1:0	rw	**Response Type Select** These bits indicate the type of response expected from the card. 00_B No Response 01_B Response Length 136 10_B Response Length 48 11_B Response Length 48 (check Busy after response)
SUB_CMD_FLAG	2	rw	**Sub Command Flag** This bit distinguishes a main command or a sub command. 0_B Main Command 1_B Sub Command

(table continues...)

(continued)

Field	Bits	Type	Description
CMD_CRC_CHK_ENABLE	3	rw	**Command CRC Check Enable** If this bit is set, Host Controller shall check the CRC field in the response. If an error is detected, it is reported as a Command CRC error. ***Note**: CRC Check enable should be set to 0 for the command with no response, R3 response and R4 response.* 0_B Disable 1_B Enable
CMD_IDX_CHK_ENABLE	4	rw	**Command Index Check Enable** If this bit is set, Host Controller shall check the index field in the response to verify if it has the same value as the command index. If it is not, it is reported as a Command Index error. ***Note**: Index Check enable should be set to 0 for the command with no response, R2 response, R3 response and R4 response.* 0_B Disable 1_B Enable
DATA_PRESENT_SEL	5	rw	**Data Present Select** This bit is set to 1 to indicate that data is present and shall be transferred using the DAT line. This bit is set to 0 in the following instances: • Command using CMD line • Command with no data transfer but using busy signal on DAT[0] line • Resume Command 0_B No Data Present 1_B Data Present
CMD_TYPE	7:6	rw	**Command Type** These bits indicate the command type. 00_B Normal Command 01_B Suspend Command 10_B Resume Command 11_B Abort Command
CMD_INDEX	13:8	rw	**Command Index** These bits shall be set to the command number that is specified in bits 45-40 of the Command Format.
RSVD	15:14	r	**Reserved** These bits are reserved. They will always return 0.

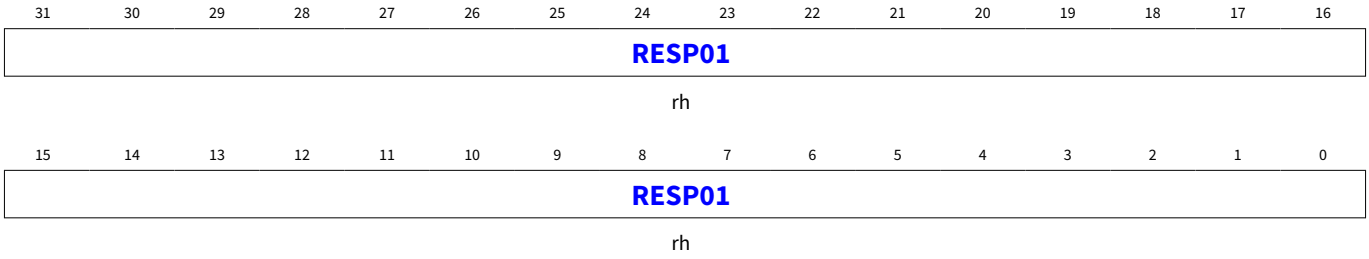
29.4.23 Response register 01

This register is used to store the response from the cards. The response can be a maximum of 128 bits. These 128 bits are segregated into four 32-bit registers: RESP01_R, RESP23_R, RESP45_R and RESP67_R.

RESP01_R stores 39-08 bits of Response Field for an SD/eMMC mode. Writing this register has no effect.

RESP01 Offset address: 410_H

Response register 01 Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
RESP01	31:0	rh	**Command Response** These bits reflect bits 39-8 of the Response Field. ***Note**: For Auto CMD, the 32-bit response (bits 39-8 of the Response Field) is updated in RESP67_R register.*

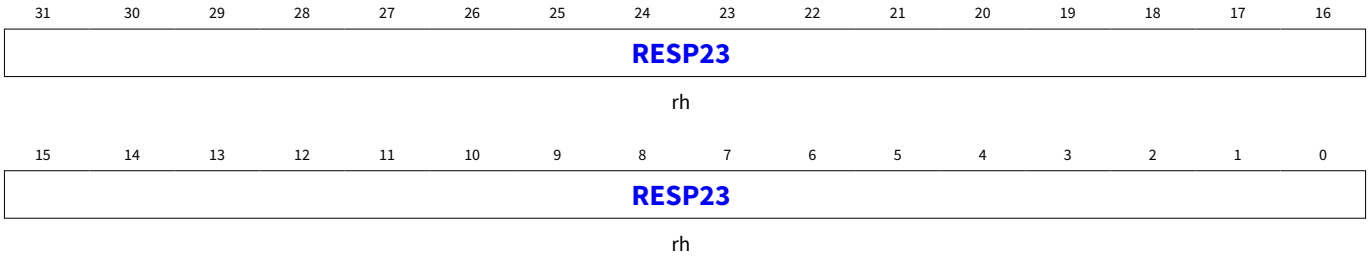
29.4.24 Response register 23

This register is used to store the response from the cards. The response can be of maximum 128 bits. These 128 bits are segregated into four 32-bit registers RESP01_R, RESP23_R, RESP45_R and RESP67_R.

RESP23_R stores 71-40 bits of Response Field for an SD/eMMC mode. Writing this register has no effect.

RESP23 Offset address: 414_H

Response register 23 Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
RESP23	31:0	rh	**Command Response** These bits reflect bits 71-40 of the Response Field.

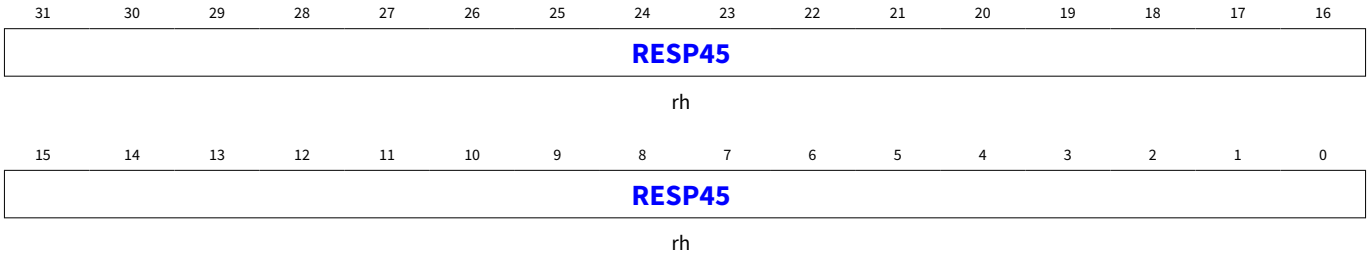
29.4.25 Response register 45

This register is used to store the response from the cards. The response can be of maximum 128 bits. These 128 bits are segregated into four 32-bit registers RESP01_R, RESP23_R, RESP45_R and RESP67_R.

RESP45_R stores 103-72 bits of Response Field for an SD/eMMC mode. Writing this register has no effect.

RESP45 Offset address: 418_H

Response register 45 Kernel Reset value: $0000\ 0000_H$



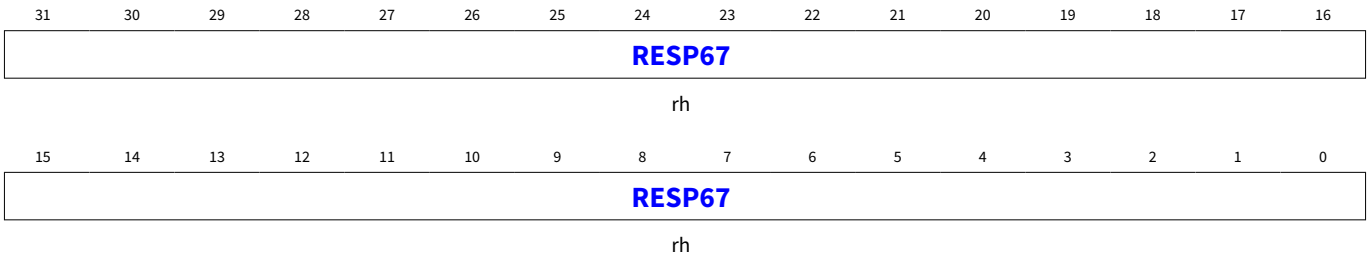
Field	Bits	Type	Description
RESP45	31:0	rh	**Command Response** These bits reflect bits 103-72 of the Response Field.

29.4.26 Response register 67

This register is used to store the response from the cards. The response can be of maximum 128 bits. These 128 bits are segregated into four 32-bit registers RESP01_R, RESP23_R, RESP45_R and RESP67_R. RESP67_R stores 135-104 bits of Response Field for an SD/eMMC mode. Writing this register has no effect.

RESP67 Offset address: $41C_H$

Response register 67 Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
RESP67	31:0	rh	**Command Response** These bits reflect bits 135-104 of the Response Field. ***Note**: For Auto CMD, this register also reflects the 32-bit response (bits 39-8 of the Response Field).*

29.4.27 Buffer data port register

This register is used to access the packet buffer. This register is applicable for an SD/eMMC mode.

BUF_DATA Offset address: 420_H

Buffer data port register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BUF_DATA															
rwh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BUF_DATA															
rwh															

Field	Bits	Type	Description
BUF_DATA	31:0	rwh	**Buffer Data** The Host Controller packet buffer can be accessed through this 32-bit Buffer Data Port register.

29.4.28 Present state register

This register indicates the present status of the Host Controller. This register is applicable for an SD/eMMC mode. Writing this register has no effect.

PSTATE_REG Offset address: 424_H

Present state register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UHS2_IF_DETECT	LANE_SYNC	IN_DORMANT_ST	SUB_CMD_STAT	CMD_ISSUE_ERR	RSVD_26	HOST_REG_VOL	CMD_LINE_LVL	DAT_3_0				WR_PROTECT_SW_LVL	CARD_DETECT_LEVEL	CARD_STABLE	CARD_INSERTED
rh	rh	rh	rh	rh	r	rh	rh	rh				rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD_15_12				BUF_RD_ENABLE	BUF_WR_ENABLE	RD_XFER_ACTIVE	WR_XFER_ACTIVE	DAT_7_4				RE_TUNE_REQ	DAT_LINE_ACTIVE	CMD_INHIBIT_DAT	CMD_INHIBIT
r				rh	rh	rh	rh	rh				rh	rh	rh	rh

Field	Bits	Type	Description
CMD_INHIBIT	0	rh	**Command Inhibit (CMD)** In an SD/eMMC mode, if this bit is 0, it indicates that the CMD line is not in use and the Host controller can issue an SD/eMMC command using the CMD line. This bit is set when the command register is written. This bit is cleared when the command response is received. This bit is not cleared by the response of auto CMD12/23 but cleared by the response of read/write command. 0_B Host Controller is not ready to issue a command 1_B Host Controller is ready to issue a command
CMD_INHIBIT_DAT	1	rh	**Command Inhibit (DAT)(SD/eMMC Mode only)** This status bit is generated if either DAT line active or Read transfer active is set to 1. If this bit is 0, it indicates that Host Controller can issue subsequent SD/eMMC commands. 0_B Can issue command which used DAT line 1_B Cannot issue command which used DAT line
DAT_LINE_ACTIVE	2	rh	**DAT Line Active (SD/eMMC Mode only)** This bit indicates whether one of the DAT lines on the SD/eMMC bus is in use. In the case of read transactions, this status indicates whether a read transfer is executing on the SD/eMMC bus. In the case of write transactions, this status indicates whether a write transfer is executing on the SD/eMMC bus. For command with busy, this status indicates whether command executing busy is executing on the SD/eMMC bus. 0_B DAT Line Inactive 1_B DAT Line Active
RE_TUNE_REQ	3	rh	**Re-Tuning Request** This bit can be ignored.
DAT_7_4	7:4	rh	**DAT[7:4] Line Signal Level (Embedded only)** This status is used to check the DAT line level to recover from errors and for debugging.
WR_XFER_ACTIVE	8	rh	**Write Transfer Active (SD/eMMC Mode only)** This status indicates whether a write transfer is active. 0_B No valid data 1_B Transferring data
RD_XFER_ACTIVE	9	rh	**Read Transfer Active (SD/eMMC Mode only)** This status indicates whether a read transfer is active. 0_B No valid data 1_B Transferring data

(table continues...)

(continued)

Field	Bits	Type	Description
BUF_WR_ENABLE	10	rh	**Buffer Write Enable** This status is used for non-DMA transfers. This bit is set if space is available for writing data. 0_B Write disable 1_B Write enable
BUF_RD_ENABLE	11	rh	**Buffer Read Enable** This status is used for non-DMA transfers. This bit is set if valid data exists in Host buffer. 0_B Read disable 1_B Read enable
RSVD_15_12	15:12	r	**Reserved** These bits are reserved. They will always return 0.
CARD_INSERTED	16	rh	**Card Inserted** This bit indicates whether a card has been inserted. The Host Controller shall debounce this signal so that Host Driver will not need to wait for it to stabilize. 0_B Reset or Debouncing or No card 1_B Card Inserted
CARD_STABLE	17	rh	**Card Stable** If this bit is set, it means the Card Detect Level is stable. No card is detected if this bit is set to 1 and card inserted is 0 0_B Reset or Debouncing 1_B No Card or Inserted
CARD_DETECT_LEVEL	18	rh	**Card Detect Pin Level - CARD_DETECT_PIN_LEVEL** This bit usually reflects the card detect stats. I.e, it reflects the inverse synchronized value of **card_detect_n** pin. But, this is not truly reflected in the bitfield because there is no pin connection. The user will need to use other means, i.e. by CMD timeout and checking CARD registers, to check whether a CARD is present and if it is write protected. 0_B No card present 1_B Card Present
WR_PROTECT_SW_LVL	19	rh	**Write Protect Switch Pin Level** The write Protect Switch is supported only for memory and combo card. This bit usually reflects the synchronized value of **card_write_prot**pin. But, this is not truly reflected in the bitfield because there is no pin connection. The user will need to use other means, i.e. by CMD timeout and checking CARD registers, to check whether a CARD is present and if it is write protected. 0_B Write protected 1_B Write enabled

(table continues...)

(continued)

Field	Bits	Type	Description
DAT_3_0	23:20	rh	**DAT[3:0] Line Signal Level (SD/eMMC Mode only)** This status is used to check the DAT line level to recover from errors and for debugging.
CMD_LINE_LVL	24	rh	**Command Line Signal Level (SD/eMMC Mode only)** This status is used to check the CMD line level to recover from errors and for debugging.
HOST_REG_VOL	25	rh	**Host Regulator Voltage Stable** This bit can be ignored.
RSVD_26	26	r	**Reserved** This bit is reserved. It will always return 0.
CMD_ISSUE_ERR	27	rh	**Command Not Issued by Error** This bit is set if a command cannot be issued after setting the command register due to error except Auto CMD12 error 0_B No error for issuing a command 1_B Command cannot be issued
SUB_CMD_STAT	28	rh	**Sub Command Status** This status is used to distinguish main command and sub command status. 0_B Main Command Status 1_B Sub Command Status
IN_DORMANT_ST	29	rh	**In Dormant Status** This bit can be ignored.
LANE_SYNC	30	rh	**Lane Synchronization** This bit can be ignored.
UHS2_IF_DETECT	31	rh	**UHS-II IF Detection** This bit can be ignored.

29.4.29 Host control 1 register

This register is used to control the operation of Host Controller. This register is applicable for an SD/eMMC mode.

HOST_CTRL1 Offset address: 428_H

Host control 1 register Kernel Reset value: 00_H

7	6	5	4	3	2	1	0
CARD_DETECT_SIG_LVL	CARD_DETECT_TEST_LVL	EXT_DAT_XFER	DMA_SEL		HIGH_SPEED_EN	DAT_XFER_WIDTH	LED_CTRL
rw	rw	rw	rw		rw	rw	rw

Field	Bits	Type	Description
LED_CTRL	0	rw	**LED Control** This bit can be ignored. Must be written with '0'. 0_B LED off 1_B LED on
DAT_XFER_WIDTH	1	rw	**Data Transfer Width (SD/eMMC Mode only)** This bit selects the data width of the Host Controller. The Host Driver shall set it to match the data width of the SD/eMMC card. 0_B 1-bit mode 1_B 4-bit mode
HIGH_SPEED_EN	2	rw	**High Speed Enable (SD/eMMC Mode only)** Before setting this bit, the Host Driver shall check the High Speed Support in the Capabilities register. This bit is used to determine the selection of preset value for High Speed mode. ***Note**: SDMMC always outputs the **sd_cmd_out and sd_dat_out** lines at the rising edge of card clock irrespective of this bit.* 0_B Normal Speed mode 1_B High Speed mode
DMA_SEL	4:3	rw	**DMA Select** This field is used to select the DMA type. When Host Version 4 Enable is 1 in Host Control 2 register: • 00b - SDMA is selected • 01b - Reserved • 10b - ADMA2 is selected • 11b - ADMA2 or ADMA3 is selected When Host Version 4 Enable is 0 in Host Control 2 register: • 00b - SDMA is selected • 01b - Reserved • 10b - 32-bit Address ADMA2 is selected • 11b - Reserved 00_B SDMA is selected 01_B Reserved 10_B ADMA2 is selected (+32 bit address if Host Version 4 enable is 1 in Host Control 2 register) 11_B ADMA2 or ADMA3 is selected
EXT_DAT_XFER	5	rw	**Extended Data Transfer Width (Embedded and an SD/eMMC Mode only)** This bit controls 8-bit bus width mode of embedded device. 0_B Bus Width is selected by Data Transfer Width 1_B 8-bit Bus Width

(table continues...)

(continued)

Field	Bits	Type	Description
CARD_DETECT_TEST_LVL	6	rw	**Card Detect Test Level** This bit is enabled while the Card Detect Signal Selection is set to 1 and it indicates card inserted or not 0_B No Card 1_B Card Inserted
CARD_DETECT_SIG_LVL	7	rw	**Card Detect Signal Selection** This bit selects source for card detection 0_B SDCD# is selected (for normal use) 1_B The Card Detect Test Level is selected (for test purpose)

29.4.30 Power control register

Control the bus power for the Card. This register is applicable for an SD/eMMC mode.

PWR_CTRL Offset address: 429_H

Power control register Kernel Reset value: 00_H

7	6	5	4	3	2	1	0
SD_BUS_VOL_VDD2			SD_BUS_PWR_VDD2	SD_BUS_VOL_VDD1			SD_BUS_PWR_VDD1
rw			rw	rw			rw

Field	Bits	Type	Description
SD_BUS_PWR_VDD1	0	rw	**SD Bus Power for VDD1** Before setting this bit, the SD Host Driver shall set SD Bus Voltage Select. If the Host Controller detects the No Card state, this bit shall be cleared. In SD mode, if this bit is cleared, the Host Controller shall stop SD Clock by clearing SD_CLK_IN bit in CLK_CTRL_R register. 0_B Power off 1_B Power on
SD_BUS_VOL_VDD1	3:1	rw	**SD Bus Voltage Select for VDD1/eMMC Bus Voltage Select for VDD.** By setting these bits, the Host Driver selects the voltage level for the SD/eMMC card. Before setting this register, the Host Driver shall check the Voltage Support bits in the capabilities register. If an unsupported voltage is selected, the Host System shall not supply SD Bus voltage. The value set in this field is available on the SDMMC output signal (sd_vdd1_sel), which will be used by the voltage switching circuitry. 000_B Reserved … 110_B Reserved 111_B SD 3.3 V (Typ.) / eMMC 3.0 V (Typ.)
SD_BUS_PWR_VDD2	4	rw	**SD Bus Power for VDD2.** This bit needs to be written with '1'

(table continues...)

(continued)

Field	Bits	Type	Description
SD_BUS_VOL_VDD2	7:5	rw	**SD Bus Voltage Select for VDD2.** This field needs to be written with 0b000 000_B VDD2 Not Supported 001_B Reserved … 101_B Reserved 110_B Not used 111_B Not used

29.4.31 Block gap control register

The host driver shall use this register to control any operation related to Block Gap. This register is applicable for an SD/eMMC mode.

BGAP_CTRL Offset address: $42A_H$

Block gap control register Kernel Reset value: 00_H

7	6	5	4	3	2	1	0
RSVD_7_4				INT_AT_BGAP	RD_WAIT_CTRL	CONTINUE_REQ	STOP_BG_REQ
r				rw	rw	rwh	rw

Field	Bits	Type	Description
STOP_BG_REQ	0	rw	**Stop At Block Gap Request** This bit is used to stop executing read and write transaction at the next block gap for non-DMA, SDMA and ADMA transfers. 0_B Transfer 1_B Stop
CONTINUE_REQ	1	rwh	**Continue Request** This bit is used to restart the transaction, which was stopped using the Stop At Block Gap Request. The Host Controller automatically clears this bit when the transaction re-starts. If stop at block gap request is set to 1, any write to this bit is ignored. 0_B Not Affected 1_B Restart
RD_WAIT_CTRL	2	rw	**Read Wait Control (SD Mode only)** This bit is used to enable the read wait protocol to stop read data using DAT[2] line if the card supports read wait. Otherwise, the Host Controller has to stop the card clock to hold the read data. Read wait feature is not supported by SDMMC. Software must set this bitfield to 0. 0_B Disable Read Wait Control 1_B Enable Read Wait Control

(table continues...)

(continued)

Field	Bits	Type	Description
INT_AT_BGAP	3	rw	**Interrupt At Block Gap (SD Mode only)** This bit is valid only in 4-bit mode of SDIO card and selects a sample point in the interrupt cycle. Setting to 1 enables interrupt detection at the block gap for a multiple block transfer. 0_B Disabled 1_B Enabled
RSVD_7_4	7:4	r	**Reserved** These bits of Block Gap Control register are reserved. They will always return 0.

29.4.32 Wakeup control register

This register is mandatory for the Host Controller, but wakeup functionality depends on the Host Controller system hardware and software. The Host Driver shall maintain voltage on the SD Bus, by setting SD Bus Power to 1 in the Power Control Register, when wakeup event through Card Interrupt is desired.

WUP_CTRL Offset address: $42B_H$

Wakeup control register Kernel Reset value: 00_H

7	6	5	4	3	2	1	0
RSVD_7_3					CARD_REMOVAL	CARD_INSERT	CARD_INT
r					rw	rw	rw

Field	Bits	Type	Description
CARD_INT	0	rw	**Wakeup Event Enable on Card Interrupt** This bit enables wakeup event through Card Interrupt assertion in the Normal Interrupt Status register. This bit can be set to 1 if FN_WUS (Wake Up Support) in CIS is set to 1. 0_B Disable 1_B Enable
CARD_INSERT	1	rw	**Wakeup Event Enable on SD Card Insertion** This bit enables wakeup event through Card Insertion assertion in the Normal Interrupt Status register. FN_WUS (Wake Up Support) in CIS does not affect this bit. 0_B Disable 1_B Enable
CARD_REMOVAL	2	rw	**Wakeup Event Enable on SD Card Removal** This bit enables wakeup event through Card Removal assertion in the Normal Interrupt Status register.FN_WUS (Wake Up Support) in CIS does not affect this bit. 0_B Disable 1_B Enable
RSVD_7_3	7:3	r	**Reserved** These bits are reserved. They will always return 0.

29.4.33 Clock control kernel register

This register controls SDCLK (card clock) in an SD/eMMC mode. This register is applicable for an SD/eMMC mode.

CLK_CTRL Offset address: $42C_H$

Clock control kernel register Kernel Reset value: 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FREQ_SEL								UPPER_FREQ_SEL		CLK_GEN_SELECT	RSVD_4	PLL_ENABLE	SD_CLK_EN	INTERNAL_CLK_STABLE	INTERNAL_CLK_EN
rwh								rwh		rwh	r	rw	rw	rh	rw

Field	Bits	Type	Description
INTERNAL_CLK_EN	0	rw	**Internal Clock Enable** This bit must be set to 1 to enable the internal clock before programming the SDMMC registers. It is set to 0 when the Host Driver is not using the Host Controller or the Host Controller awaits a wakeup interrupt. The Host Controller should stop its internal clock in-order to enter low power state. However, registers, belonging to hclk domain can still be capable of read and write. There are other registers belonging to bclk domain like BLOCKSIZE, BLOCKCOUNT, ARGUMENT, XFER_MODE_CMD, that cannot be read and written to when there is no internal clock 0_B Stop 1_B Oscillate
INTERNAL_CLK_STABLE	1	rh	**Internal Clock Stable** Host Driver checks clock stability by this status twice after Internal Clock Enable is set and after PLL Enable is set. 0_B Not Ready 1_B Ready
SD_CLK_EN	2	rw	**SD/eMMC Clock Enable** This bit stops the SDCLK or RCLK when this bit is set to 0. SDCLK/RCLK Frequency Select can be changed when this bit is 0. 0_B Disable 1_B Enable
PLL_ENABLE	3	rw	**PLL Enable** This bit can be ignored. Must be written with '0'. 0_B PLL is in low power mode 1_B PLL is enabled
RSVD_4	4	r	**Reserved** This bit is reserved. It will always return 0.

(table continues...)

(continued)

Field	Bits	Type	Description
CLK_GEN_SELECT	5	rwh	**Clock Generator Select** This bit is used to select the clock generator mode in SDCLK/RCLK Frequency Select. If Preset Value Enable = 0, this field is set by Host Driver. If Preset Value Enable = 1, this field is automatically set to a value specified in one of the Preset Value registers. 0_B Divided Clock Mode 1_B Programmable Clock Mode
UPPER_FREQ_SEL	7:6	rwh	**SDCLK/RCLK Upper Frequency Select -** These bits specify the upper 2 bits of 10-bit SDCLK/RCLK Frequency Select control. The value is reflected on the upper 2 bits of the card_clk_freq_sel signal
FREQ_SEL	15:8	rwh	**SDCLK/RCLK Frequency Select** These bits together with UPPER_FREQ_SEL are used to select the frequency of the SDCLK pin. This field depends on setting of Preset Value Enable in Host Control 2 register. If Preset Value Enable = 0, this field is set by Host Driver. If Preset Value Enable = 1, this field is automatically set to a value specified in one of the Preset Value register. frequency of the SDCLK pin is calculated with the formula =Fsdmmc * 1/((N+1)*2). Here Fsdmmc= 100MHz **10-bit Divided Clock Mode:** • 3FFh - 1/2048 Fsdmmc • ... - ... • N - 1/2(N+1) Fsdmmc • ... - ... • 002h - 1/6 Fsdmmc • 001h - 1/4 Fsdmmc • 000h - 1/2 Fsdmmc

29.4.34 Timeout control register

At the initialization of the Host Controller, the Host Driver shall set the Data Timeout Counter value for an SD/eMMC mode according to the timer clock defined in the Capabilities register.

TOUT_CTRL Offset address: $42E_H$

Timeout control register Kernel Reset value: 00_H

7	6	5	4	3	2	1	0
RSVD_7_4				TOUT_CNT			
r				rw			

Field	Bits	Type	Description
TOUT_CNT	3:0	rw	**Data Timeout Counter Value.** This value determines the interval by which DAT line timeouts are detected. The Timeout is based on the base clock TMCLK, which is fSPB devided by 32. Timeout clock frequency will be generated by dividing the base clock TMCLK value by this value. When setting this register, prevent inadvertent timeout events by clearing the Data Timeout Error Status Enable (in the Error Interrupt Status Enable register) ***Note**: During a boot operating in an eMMC mode, an application should configure the boot data timeout value (approximately 1 sec) in this field.* 0_H TMCLK * 2^13 1_H TMCLK * 2^14 2_H TMCLK * 2^15 … E_H TMCLK * 2^27 F_H Reserved
RSVD_7_4	7:4	r	**Reserved** These bits are reserved. They will always return 0.

29.4.35 Software reset register

A reset pulse is generated when writing 1 to each bit of this register. After completing the reset, the Host Controller shall clear each bit. As it takes some time to complete software reset, the Host Driver shall confirm that these bits are 0. This register is applicable for an SD/eMMC mode.

SW_RST Offset address: $42F_H$

Software reset register Kernel Reset value: 00_H

7	6	5	4	3	2	1	0
RSVD_7_3					SW_RST_DAT	SW_RST_CMD	SW_RST_ALL
r					rwh	rwh	rwh

Field	Bits	Type	Description
SW_RST_ALL	0	rwh	**Software Reset For All** This reset affects the entire Host Controller except for the card detection circuit. During its initialization, the Host Driver shall set this bit to 1 to reset the Host Controller. All registers are reset except the capabilities register. If this bit is set to 1, the Host Driver should issue reset command and reinitialize the SD/eMMC card. 0_B Work 1_B Reset

(table continues...)

(continued)

Field	Bits	Type	Description
SW_RST_CMD	1	rwh	**Software Reset For CMD line** Only part of command circuit is reset to be able to issue a command. This reset is effective only for a command issuing circuit (including response error statuses related to Command Inhibit (CMD) control) and does not affect the data transfer circuit. Host Controller can continue data transfer even after this reset is executed during handling of subcommand response errors. The following registers and bits are cleared by this bit: - Present State register Command Inhibit (CMD) - Normal Interrupt Status register Command Complete - Error Interrupt Status Response error statuses related to Command Inhibit (CMD) 0_B Work 1_B Reset
SW_RST_DAT	2	rwh	**Software Reset For DAT line (SD/eMMC Mode only)** This bit resets only a part of the data circuit and the DMA circuit is also reset. The following registers and bits are cleared by this bit: • Buffer Data Port register Buffer is cleared and initialized. - Buffer is cleared and initialized. • Present state register Buffer Read Enable Buffer Write Enable Read Transfer Active Write Transfer Active DAT Line Active Command Inhibit (DAT) - Buffer Read Enable - Buffer Write Enable - Read Transfer Active - Write Transfer Active - DAT Line Active - Command Inhibit (DAT) • Block Gap Control register Continue Request Stop At Block Gap Request - Continue Request - Stop At Block Gap Request • Normal Interrupt status register Buffer Read Ready Buffer Write Ready DMA Interrupt Block Gap Event Transfer Complete - Buffer Read Ready - Buffer Write Ready - DMA Interrupt - Block Gap Event - Transfer Complete 0_B Work 1_B Reset

(table continues...)

(continued)

Field	Bits	Type	Description
RSVD_7_3	7:3	r	**Reserved** These bits are reserved. They will always return 0.

29.4.36 Normal interrupt status register

This register reflects the status of Normal Interrupt. This register is applicable for an SD/eMMC mode.

NORMAL_INT_STAT Offset address: 430_H

Normal interrupt status register Kernel Reset value: 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ERR_INTERRUPT	CQE_EVENT	FX_EVENT	RE_TUNE_EVENT	INT_C	INT_B	INT_A	CARD_INTERRUPT	CARD_REMOVAL	CARD_INSERTION	BUF_RD_READY	BUF_WR_READY	DMA_INTERRUPT	BGAP_EVENT	XFER_COMPLETE	CMD_COMPLETE
rh	rwh	rh	r	rh	rh	rh	rh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
CMD_COMPLETE	0	rwh	**Command Complete** In an SD/eMMC Mode, this bit is set when the end bit of a response except for Auto CMD12 and Auto CMD23. This interrupt is not generated when the Response Interrupt Disable in Transfer Mode Register is set to 1. A write of 1 clears this register field. 0_B No command complete 1_B Command Complete
XFER_COMPLETE	1	rwh	**Transfer Complete** This bit is set when a read/write transfer and a command with status busy is completed. A write of 1 clears this register field. 0_B Not complete 1_B Command execution is completed
BGAP_EVENT	2	rwh	**Block Gap Event** This bit is set when both read/write transaction is stopped at block gap due to Stop at Block Gap Request. A write of 1 clears this register field. 0_B No Block Gap Event 1_B Transaction stopped

(table continues...)

(continued)

Field	Bits	Type	Description
DMA_INTERRUPT	3	rwh	**DMA Interrupt** This bit is set if the Host Controller detects the SDMA Buffer Boundary during transfer. In case of ADMA, by setting Int field in the descriptor table, Host controller generates this interrupt. This interrupt shall not be generated after Transfer Complete. A write of 1 clears this register field. 0_B No DMA Interrupt 1_B DMA Interrupt is generated
BUF_WR_READY	4	rwh	**Buffer Write Ready** This bit is set if the Buffer Write Enable changes from 0 to 1. A write of 1 clears this register field. 0_B Not ready to write buffer 1_B Ready to write buffer
BUF_RD_READY	5	rwh	**Buffer Read Ready** This bit is set if the Buffer Read Enable changes from 0 to 1. A write of 1 clears this register field. 0_B Not ready to read buffer 1_B Ready to read buffer
CARD_INSERTION	6	rwh	**Card Insertion** This bit is set if the Card Inserted in the Present State register changes from 0 to 1. A write of 1 clears this register field. 0_B Card state stable or Debouncing 1_B Card Inserted
CARD_REMOVAL	7	rwh	**Card Removal** This bit is set if the Card Inserted in the Present State register changes from 1 to 0. A write of 1 clears this register field. 0_B Card state stable or Debouncing 1_B Card Removed
CARD_INTERRUPT	8	rh	**Card Interrupt** This bit reflects the synchronized value of: DAT[1]. ***Note**: This reflection changes after the interrupt ocurred. Clearing the interrupt is only possible by clearing it inside the card while the CARD_INTERRUPT is disabled inside SDMMC.* 0_B No Card Interrupt 1_B Generate Card Interrupt

(table continues...)

(continued)

Field	Bits	Type	Description
INT_A	9	rh	**INT_A (Embedded)** This bit is set if INT_A is enabled. 0_B No Interrupt is detected 1_B INT_A is detected
INT_B	10	rh	**INT_B (Embedded)** This bit is set if INT_B is enabled. 0_B No Interrupt is detected 1_B INT_B is detected
INT_C	11	rh	**INT_C (Embedded)** This bit is set if INT_C is enabled. 0_B No Interrupt is detected 1_B INT_C is detected
RE_TUNE_EVENT	12	r	**Re-tuning Event** This bit can be ignored.
FX_EVENT	13	rh	**FX Event** This status is set when r[14] of response register is set to 1 and Response Type R1/R5 is set to 0 in Transfer Mode register. This interrupt is used with response check function. 0_B No Event 1_B FX Event is detected
CQE_EVENT	14	rwh	**Command Queuing Event** This bit can be ignored. 0_B No Event 1_B Command Queuing Event is detected
ERR_INTERRUPT	15	rh	**Error Interrupt** If any of the bits in the Error Interrupt Status register are set, then this bit is set. 0_B No Error 1_B Error

29.4.37 Error interrupt status register

Signals defined in this register can be enabled by the Error Interrupt Status Enable register, but not by the Error Interrupt Signal Enable register. The interrupt is generated when the Error Interrupt Status Enable is enabled and at least one of the statuses is set to 1. Writing to 1 clears the bit and writing to 0 keeps the bit unchanged. More than one status can be cleared with a single register write. This register is applicable for an SD/eMMC mode.

ERROR_INT_STAT Offset address: 432_H

Error interrupt status register Kernel Reset value: 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VENDOR_ERR3	VENDOR_ERR2	VENDOR_ERR1	BOOT_ACK_ERR	RESP_ERR	TUNING_ERR	ADMA_ERR	AUTO_CMD_ERR	CUR_LMT_ERR	DATA_END_BIT_ERR	DATA_CRC_ERR	DATA_TOUT_ERR	CMD_IDX_ERR	CMD_END_BIT_ERR	CMD_CRC_ERR	CMD_TOUT_ERR
rw	rw	rw	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
CMD_TOUT_ERR	0	rwh	**Command Timeout Error (SD/eMMC Mode only)** This bit is set only if no response is returned within 64 SD clock cycles from the end bit of the command. If the Host Controller detects a CMD line conflict, clock cycles. A write of 1 clears this register field. 0_B No Error 1_B Time out
CMD_CRC_ERR	1	rwh	**Command CRC Error (SD/eMMC Mode only)** Command CRC Error is generated in two cases. If a response is returned and the Command Timeout Error is set to 0 (indicating no timeout), this bit is set to 1 when detecting a CRC error in the command response. The Host Controller detects a CMD line conflict by monitoring the CMD line when a command is issued. If the Host Controller drives the CMD line to 1 level, but detects 0 level on the CMD line at the next SD clock edge, then the Host Controller shall abort the command (stop driving CMD line) and set this bit to 1. The Command Timeout Error shall also be set to 1 to distinguish CMD line conflict. A write of 1 clears this register field. 0_B No Error 1_B CRC Error Generated
CMD_END_BIT_ERR	2	rwh	**Command End Bit Error (SD/eMMC Mode only)** This bit is set when detecting that the end bit of a command response is 0. A write of 1 clears this register field. 0_B No Error 1_B End Bit Error Generated
CMD_IDX_ERR	3	rwh	**Command Index Error (SD/eMMC Mode only)** This bit is set if a Command Index error occurs in the command response. A write of 1 clears this register field. 0_B No Error 1_B Error

(table continues...)

(continued)

Field	Bits	Type	Description
DATA_TOUT_ERR	4	rwh	**Data Timeout Error (SD/eMMC Mode only)** This bit is set when detecting one of the following timeout conditions: - Busy timeout for R1b, R5b type - Busy timeout after Write CRC status - Write CRC Status timeout - Read Data timeout A write of 1 clears this register field. 0_B No Error 1_B Time out
DATA_CRC_ERR	5	rwh	**Data CRC Error (SD/eMMC Mode only).** This error occurs when detecting CRC error when transferring read data which uses the DAT line, when detecting the Write CRC status having a value of other than 010 or when write CRC status timeout. A write of 1 clears this register field. 0_B No Error 1_B Error
DATA_END_BIT_ERR	6	rwh	**Data End Bit Error (SD/eMMC Mode only)** This error occurs either when detecting 0 at the end bit position of read data that uses the DAT line or at the end bit position of the CRC status. A write of 1 clears this register field. 0_B No Error 1_B Error
CUR_LMT_ERR	7	rwh	**Current Limit Error** By setting the SD Bus Power bit in the Power Control register, the Host Controller is requested to supply power for the SD Bus. If the Host Controller supports the Current Limit function, it can be protected from an illegal card by stopping power supply to the card in which case this bit indicates a failure status. A reading of 1 for this bit means that the Host Controller is not supplying power to the SD card due to some failure. A reading of 0 for this bit means that the Host Controller is supplying power and no error has occurred. The Host Controller may require some sampling time to detect the current limit. If the Host Controller does not support this function, this bit shall always be set to 0. A write of 1 clears this register field. 0_B No Error 1_B Power Fail

(table continues...)

(continued)

Field	Bits	Type	Description
AUTO_CMD_ERR	8	rwh	**Auto CMD Error (SD/eMMC Mode only)** This error status is used by Auto CMD12 and Auto CMD23. This bit is set when detecting that any of the bits D00 to D05 in Auto CMD Error Status register has changed from 0 to 1. D07 is effective in case of Auto CMD12. Auto CMD Error Status register is valid while this bit is set to 1 and may be cleared by clearing of this bit (another implementation is also allowed). A write of 1 clears this register field. 0_B No Error 1_B Error
ADMA_ERR	9	rwh	**ADMA Error** This bit is set when the Host Controller detects error during ADMA-based data transfer. The error could be due to following reasons: • Error response received from System bus (Master I/F) • ADMA3,ADMA2 Descriptors invalid • CQE Task or Transfer descriptors invalid When the error occurs, the state of the ADMA is saved in the ADMA Error Status register. In eMMC CQE mode: The Host Controller generates this Interrupt when it detects an invalid descriptor data (Valid=0) at the ST_FDS state. ADMA Error State in the ADMA Error Status indicates that an error has occurred in ST_FDS state. The Host Driver may find that Valid bit is not set at the error descriptor. A write of 1 clears this register field. 0_B No Error 1_B Error
TUNING_ERR	10	rwh	**Tuning Error** This bit can be ignored and needs to be written to with '0'.
RESP_ERR	11	rwh	**Response Error (SD Mode only)** Host Controller Version 4.00 supports response error check function to avoid overhead of response error check by Host Driver during DMA execution. If Response Error Check Enable is set to 1 in the Transfer Mode register, Host Controller Checks R1 or R5 response. If an error is detected in a response, this bit is set to 1. A write of 1 clears this register field. 0_B No Error 1_B Error

(table continues...)

(continued)

Field	Bits	Type	Description
BOOT_ACK_ERR	12	rwh	**Boot Acknowledgement Error (eMMC Mode only)** This bit is set when there is a timeout for boot acknowledgement or when detecting boot ack status having a value other than 010. This is applicable only when boot acknowledgement is expected and boot feature in SDMMC is enabled with DWC_MSHC_EMMC_BOOT_EN = 1. A write of 1 clears this register field. 0_B No Error 1_B Error
VENDOR_ERR1	13	rw	**Reserved** This bit can be ignored. Must be written with '0'. 0_B No Error 1_B Error
VENDOR_ERR2	14	rw	**Reserved** This bit can be ignored. Must be written with '0'. 0_B No Error 1_B Error
VENDOR_ERR3	15	rw	**Reserved** This bit can be ignored. Must be written with '0'. 0_B No Error 1_B Error

29.4.38 Normal interrupt status enable register

This register enables the Interrupt Status for Normal Interrupt Status register (NORMAL_INT_STAT_R) when NORMAL_INT_STAT_R is set to 1. This register is applicable for an SD/eMMC mode.

NORMAL_INT_STAT_EN Offset address: 434_H

Normal interrupt status enable register Kernel Reset value: 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD_15	CQE_EVENT_STAT_EN	FX_EVENT_STAT_EN	RE_TUNE_EVENT_STAT_EN	INT_C_STAT_EN	INT_B_STAT_EN	INT_A_STAT_EN	CARD_INTERRUPT_STAT_EN	CARD_REMOVAL_STAT_EN	CARD_INSERTION_STAT_EN	BUF_RD_READY_STAT_EN	BUF_WR_READY_STAT_EN	DMA_INTERRUPT_STAT_EN	BGAP_EVENT_STAT_EN	XFER_COMPLETE_STAT_EN	CMD_COMPLETE_STAT_EN
r	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
CMD_COMPLETE_STAT_EN	0	rw	**Command Complete Status Enable** 0_B Masked 1_B Enabled

(table continues...)

(continued)

Field	Bits	Type	Description
XFER_COMPLETE_STAT_EN	1	rw	**Transfer Complete Status Enable** 0_B Masked 1_B Enabled
BGAP_EVENT_STAT_EN	2	rw	**Block Gap Event Status Enable** 0_B Masked 1_B Enabled
DMA_INTERRUPT_STAT_EN	3	rw	**DMA Interrupt Status Enable** 0_B Masked 1_B Enabled
BUF_WR_READY_STAT_EN	4	rw	**Buffer Write Ready Status Enable** 0_B Masked 1_B Enabled
BUF_RD_READY_STAT_EN	5	rw	**Buffer Read Ready Status Enable** 0_B Masked 1_B Enabled
CARD_INSERTION_STAT_EN	6	rw	**Card Insertion Status Enable** 0_B Masked 1_B Enabled
CARD_REMOVAL_STAT_EN	7	rw	**Card Removal Status Enable** 0_B Masked 1_B Enabled
CARD_INTERRUPT_STAT_EN	8	rw	**Card Interrupt Status Enable** If this bit is set to 0, the Host Controller shall clear the interrupt request to the System. The Card Interrupt detection is stopped when this bit is cleared and restarted when this bit is set to 1. The Host Driver may clear the Card Interrupt Status Enable before servicing the Card Interrupt and may set this bit again after all interrupt requests from the card are cleared to prevent inadvertent interrupts. By setting this bit to 0, interrupt input should be masked by implementation so that the interrupt input is not affected by external signal in any state (for example, floating). 0_B Masked 1_B Enabled
INT_A_STAT_EN	9	rw	**INT_A (Embedded) Status Enable** If this bit is set to 0, the Host Controller shall clear the interrupt request to the System. The Host Driver may clear this bit before servicing the INT_A and may set this bit again after all interrupt requests to INT_A pin are cleared to prevent inadvertent interrupts. 0_B Masked 1_B Enabled

(table continues...)

(continued)

Field	Bits	Type	Description
INT_B_STAT_EN	10	rw	**INT_B (Embedded) Status Enable.** If this bit is set to 0, the Host Controller shall clear the interrupt request to the System. The Host Driver may clear this bit before servicing the INT_B and may set this bit again after all interrupt requests to INT_B pin are cleared to prevent inadvertent interrupts. 0_B Masked 1_B Enabled
INT_C_STAT_EN	11	rw	**INT_C (Embedded) Status Enable.** If this bit is set to 0, the Host Controller shall clear the interrupt request to the System. The Host Driver may clear this bit before servicing the INT_C and may set this bit again after all interrupt requests to INT_C pin are cleared to prevent inadvertent interrupts. 0_B Masked 1_B Enabled
RE_TUNE_EVENT_STAT_EN	12	rw	**Re-Tuning Event Status Enable** This bit must always remain '0'.
FX_EVENT_STAT_EN	13	rw	**FX Event Status Enable.** This bit is added from Version 4.10. 0_B Masked 1_B Enabled
CQE_EVENT_STAT_EN	14	rw	**CQE Event Status Enable.** This bit must always remain '0'.
RSVD_15	15	r	**Reserved** This bit is reserved. It will always return 0.

29.4.39 Error interrupt status enable register

This register sets the Interrupt Status for Error Interrupt Status register (ERROR_INT_STAT_R), when ERROR_INT_STAT_EN_R is set to 1. This register is applicable for an SD/eMMC mode.

ERROR_INT_STAT_EN Offset address: 436_H

Error interrupt status enable register Kernel Reset value: 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VENDOR_ERR_STAT_EN3	VENDOR_ERR_STAT_EN2	VENDOR_ERR_STAT_EN1	BOOT_ACK_ERR_STAT_EN	RESP_ERR_STAT_EN	TUNING_ERR_STAT_EN	ADMA_ERR_STAT_EN	AUTO_CMD_ERR_STAT_EN	CUR_LMT_ERR_STAT_EN	DATA_END_BIT_ERR_STAT_EN	DATA_CRC_ERR_STAT_EN	DATA_TOUT_ERR_STAT_EN	CMD_IDX_ERR_STAT_EN	CMD_END_BIT_ERR_STAT_EN	CMD_CRC_ERR_STAT_EN	CMD_TOUT_ERR_STAT_EN
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
CMD_TOUT_ERR_STAT_EN	0	rw	**Command Timeout Error Status Enable (SD/eMMC Mode only).** 0_B Masked 1_B Enabled
CMD_CRC_ERR_STAT_EN	1	rw	**Command CRC Error Status Enable (SD/eMMC Mode only)** 0_B Masked 1_B Enabled
CMD_END_BIT_ERR_STAT_EN	2	rw	**Command End Bit Error Status Enable (SD/eMMC Mode only)** 0_B Masked 1_B Enabled
CMD_IDX_ERR_STAT_EN	3	rw	**Command Index Error Status Enable (SD/eMMC Mode only)** 0_B Masked 1_B Enabled
DATA_TOUT_ERR_STAT_EN	4	rw	**Data Timeout Error Status Enable (SD/eMMC Mode only)** 0_B Masked 1_B Enabled
DATA_CRC_ERR_STAT_EN	5	rw	**Data CRC Error Status Enable (SD/eMMC Mode only)** 0_B Masked 1_B Enabled
DATA_END_BIT_ERR_STAT_EN	6	rw	**Data End Bit Error Status Enable (SD/eMMC Mode only).** 0_B Masked 1_B Enabled
CUR_LMT_ERR_STAT_EN	7	rw	**Current Limit Error Status Enable** 0_B Masked 1_B Enabled
AUTO_CMD_ERR_STAT_EN	8	rw	**Auto CMD Error Status Enable (SD/eMMC Mode only).** 0_B Masked 1_B Enabled
ADMA_ERR_STAT_EN	9	rw	**ADMA Error Status Enable.** 0_B Masked 1_B Enabled
TUNING_ERR_STAT_EN	10	rw	**Tuning Error Status Enable.** This bit must always remain '0'.
RESP_ERR_STAT_EN	11	rw	**Response Error Status Enable (SD Mode only)** 0_B Masked 1_B Enabled

(table continues...)

(continued)

Field	Bits	Type	Description
BOOT_ACK_ERR_STAT_EN	12	rw	**Boot Acknowledgment Error (eMMC Mode only).** Setting this bit to 1 enables setting of Boot Acknowledgment Error in Error Interrupt Status register (ERROR_INT_STAT_R). 0_B Masked 1_B Enabled
VENDOR_ERR_STAT_EN1	13	rw	**Reserved** This bit can be ignored. Must be written with '0'. 0_B Masked 1_B Enabled
VENDOR_ERR_STAT_EN2	14	rw	**Reserved** This bit can be ignored. Must be written with '0'. 0_B Masked 1_B Enabled
VENDOR_ERR_STAT_EN3	15	rw	**Reserved** This bit can be ignored. Must be written with '0'. 0_B Masked 1_B Enabled

29.4.40 Normal interrupt signal enable register

This register is used to select the interrupt status that is indicated to the Host System as the interrupt. All these status bits share the same 1-bit interrupt line. Setting any of these bits to 1, enables interrupt generation. This register is applicable for an SD/eMMC mode.

NORMAL_INT_SIGNAL_EN Offset address: 438_H

Normal interrupt signal enable register Kernel Reset value: 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD_15	CQE_EVENT_SIGNAL_EN	FX_EVENT_SIGNAL_EN	RE_TUNE_EVENT_SIGNAL_EN	INT_C_SIGNAL_EN	INT_B_SIGNAL_EN	INT_A_SIGNAL_EN	CARD_INTERRUPT_SIGNAL_EN	CARD_REMOVAL_SIGNAL_EN	CARD_INSERTION_SIGNAL_EN	BUF_RD_READY_SIGNAL_EN	BUF_WR_READY_SIGNAL_EN	DMA_INTERRUPT_SIGNAL_EN	BGAP_EVENT_SIGNAL_EN	XFER_COMPLETE_SIGNAL_EN	CMD_COMPLETE_SIGNAL_EN
r	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
CMD_COMPLETE_SIGNAL_EN	0	rw	**Command Complete Signal Enable** 0_B Masked 1_B Enabled
XFER_COMPLETE_SIGNAL_EN	1	rw	**Transfer Complete Signal Enable** 0_B Masked 1_B Enabled

(table continues...)

(continued)

Field	Bits	Type	Description
BGAP_EVENT_SIGNAL_EN	2	rw	**Block Gap Event Signal Enable** 0_B Masked 1_B Enabled
DMA_INTERRUPT_SIGNAL_EN	3	rw	**DMA Interrupt Signal Enable** 0_B Masked 1_B Enabled
BUF_WR_READY_SIGNAL_EN	4	rw	**Buffer Write Ready Signal Enable** 0_B Masked 1_B Enabled
BUF_RD_READY_SIGNAL_EN	5	rw	**Buffer Read Ready Signal Enable** 0_B Masked 1_B Enabled
CARD_INSERTION_SIGNAL_EN	6	rw	**Card Insertion Signal Enable** 0_B Masked 1_B Enabled
CARD_REMOVAL_SIGNAL_EN	7	rw	**Card Removal Signal Enable** 0_B Masked 1_B Enabled
CARD_INTERRUPT_SIGNAL_EN	8	rw	**Card Interrupt Signal Enable** 0_B Masked 1_B Enabled
INT_A_SIGNAL_EN	9	rw	**INT_A (Embedded) Signal Enable** 0_B Masked 1_B Enabled
INT_B_SIGNAL_EN	10	rw	**INT_B (Embedded) Signal Enable** 0_B Masked 1_B Enabled
INT_C_SIGNAL_EN	11	rw	**INT_C (Embedded) Signal Enable** 0_B Masked 1_B Enabled
RE_TUNE_EVENT_SIGNAL_EN	12	rw	**Re-Tuning Event Signal Enable.** This bit must always remain '0'.
FX_EVENT_SIGNAL_EN	13	rw	**FX Event Signal Enable** 0_B Masked 1_B Enabled
CQE_EVENT_SIGNAL_EN	14	rw	**Command Queuing Engine Event Signal Enable** This bit must always remain '0'.

(table continues...)

(continued)

Field	Bits	Type	Description
RSVD_15	15	r	**Reserved** This bit is reserved. It will always return 0.

29.4.41 Error interrupt signal enable register

This register is used to select the interrupt status that is notified to the Host System as an interrupt. All these status bits share the same 1-bit interrupt line. Setting any of these bits to 1 enables interrupt generation. This register is applicable for an SD/eMMC mode.

ERROR_INT_SIGNAL_EN Offset address: $43A_H$

Error interrupt signal enable register Kernel Reset value: 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VENDOR_ERR_SIGNAL_EN3	VENDOR_ERR_SIGNAL_EN2	VENDOR_ERR_SIGNAL_EN1	BOOT_ACK_ERR_SIGNAL_EN	RESP_ERR_SIGNAL_EN	TUNING_ERR_SIGNAL_EN	ADMA_ERR_SIGNAL_EN	AUTO_CMD_ERR_SIGNAL_EN	CUR_LMT_ERR_SIGNAL_EN	DATA_END_BIT_ERR_SIGNAL_EN	DATA_CRC_ERR_SIGNAL_EN	DATA_TOUT_ERR_SIGNAL_EN	CMD_IDX_ERR_SIGNAL_EN	CMD_END_BIT_ERR_SIGNAL_EN	CMD_CRC_ERR_SIGNAL_EN	CMD_TOUT_ERR_SIGNAL_EN
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
CMD_TOUT_ERR_SIGNAL_EN	0	rw	**Command Timeout Error Signal Enable (SD/eMMC Mode only)** 0_B Masked 1_B Enabled
CMD_CRC_ERR_SIGNAL_EN	1	rw	**Command CRC Error Signal Enable (SD/eMMC Mode only)** 0_B Masked 1_B Enabled
CMD_END_BIT_ERR_SIGNAL_EN	2	rw	**Command End Bit Error Signal Enable (SD/eMMC Mode only)** 0_B Masked 1_B Enabled
CMD_IDX_ERR_SIGNAL_EN	3	rw	**Command Index Error Signal Enable (SD/eMMC Mode only)** 0_B Masked 1_B Enabled
DATA_TOUT_ERR_SIGNAL_EN	4	rw	**Data Timeout Error Signal Enable (SD/eMMC Mode only)** 0_B Masked 1_B Enabled
DATA_CRC_ERR_SIGNAL_EN	5	rw	**Data CRC Error Signal Enable (SD/eMMC Mode only)** 0_B Masked 1_B Enabled

(table continues...)

(continued)

Field	Bits	Type	Description
DATA_END_BIT_ERR_SIGNAL_EN	6	rw	**Data End Bit Error Signal Enable (SD/eMMC Mode only)** 0_B Masked 1_B Enabled
CUR_LMT_ERR_SIGNAL_EN	7	rw	**Current Limit Error Signal Enable** 0_B Masked 1_B Enabled
AUTO_CMD_ERR_SIGNAL_EN	8	rw	**Auto CMD Error Signal Enable (SD/eMMC Mode only)** 0_B Masked 1_B Enabled
ADMA_ERR_SIGNAL_EN	9	rw	**ADMA Error Signal Enable** 0_B Masked 1_B Enabled
TUNING_ERR_SIGNAL_EN	10	rw	**Tuning Error Signal Enable** This bit must always remain '0'.
RESP_ERR_SIGNAL_EN	11	rw	**Response Error Signal Enable (SD Mode only)** 0_B Masked 1_B Enabled
BOOT_ACK_ERR_SIGNAL_EN	12	rw	**Boot Acknowledgment Error (eMMC Mode only).** Setting this bit to 1 enables generating interrupt signal when Boot Acknowledgement Error in Error Interrupt Status register is set. 0_B Masked 1_B Enabled
VENDOR_ERR_SIGNAL_EN1	13	rw	**Reserved** This bit can be ignored. Must be written with '0'. 0_B Masked 1_B Enabled
VENDOR_ERR_SIGNAL_EN2	14	rw	**Reserved** This bit can be ignored. Must be written with '0'. 0_B Masked 1_B Enabled
VENDOR_ERR_SIGNAL_EN3	15	rw	**Reserved** This bit can be ignored. Must be written with '0'. 0_B Masked 1_B Enabled

29.4.42 Auto CMD status register

This register is used to indicate the CMD12 response error of Auto CMD12, and the CMD23 response error of Auto CMD23. The Host driver can determine the kind of Auto CMD12/CMD23 errors that can occur in this register. Auto CMD23 errors are indicated in bit 04-01. This register is valid only when Auto CMD Error is set. This register is applicable for an SD/eMMC mode. Writing this register has no effect.

AUTO_CMD_STAT Offset address: $43C_H$

Auto CMD status register Kernel Reset value: 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD_15_8								CMD_NOT_ISSUED_AUTO_CMD12	RSVD_6	AUTO_CMD_RESP_ERR	AUTO_CMD_IDX_ERR	AUTO_CMD_EBIT_ERR	AUTO_CMD_CRC_ERR	AUTO_CMD_TOUT_ER R	AUTO_CMD12_NOT_EXEC
r								rh	r	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
AUTO_CMD12_NOT_EXEC	0	rh	**Auto CMD12 Not Executed** If multiple memory block data transfer is not started due to a command error, this bit is not set because it is not necessary to issue an Auto CMD12. Setting this bit to 1 means that the Host Controller cannot issue Auto CMD12 to stop multiple memory block data transfer, due to some error. If this bit is set to 1, error status bits (D04-D01) is meaningless. This bit is set to 0 when Auto CMD Error is generated by Auto CMD23. 0_B Executed 1_B Not Executed
AUTO_CMD_TOUT_ERR	1	rh	**Auto CMD Timeout Error** This bit is set if no response is returned with 64 SDCLK cycles from the end bit of the command. If this bit is set to 1, error status bits (D04 - D01) are meaningless. 0_B No Error 1_B Time out
AUTO_CMD_CRC_ERR	2	rh	**Auto CMD CRC Error** This bit is set when detecting a CRC error in the command response. 0_B No Error 1_B CRC Error Generated
AUTO_CMD_EBIT_ERR	3	rh	**Auto CMD End Bit Error** This bit is set when detecting that the end bit of command response is 0. 0_B No Error 1_B End Bit Error Generated

(table continues...)

(continued)

Field	Bits	Type	Description
AUTO_CMD_IDX_ERR	4	rh	**Auto CMD Index Error** This bit is set if the command index error occurs in response to a command. 0_B No Error 1_B Error
AUTO_CMD_RESP_ERR	5	rh	**Auto CMD Response Error** This bit is set when Response Error Check Enable in the Transfer Mode register is set to 1 and an error is detected in R1 response of either Auto CMD12 or CMD13. This status is ignored if any bit between D00 to D04 is set to 1. 0_B No Error 1_B Error
RSVD_6	6	r	**Reserved** This bit is reserved. It will always return 0.
CMD_NOT_ISSUED_AUTO_CMD12	7	rh	**Command Not Issued By Auto CMD12 Error** If this bit is set to 1, CMD_wo_DAT is not executed due to an Auto CMD12 Error (D04-D01) in this register. This bit is set to 0 when Auto CMD Error is generated by Auto CMD23. 0_B No Error 1_B Not Issued
RSVD_15_8	15:8	r	**Reserved** These bits are reserved. They will always return 0.

29.4.43 Host control 2 register

This register is used to control how the Host Controller operates. This register is applicable for an SD/eMMC mode.

HOST_CTRL2 Offset address: $43E_H$

Host control 2 register Kernel Reset value: 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PRESET_VAL_ENABLE	ASYNC_INT_ENABLE	ADDRESSING	HOST_VER4_ENABLE	CMD23_ENABLE	ADMA2_LEN_MODE	RSVD_9	UHS2_IF_ENABLE	SAMPLE_CLK_SEL	EXEC_TUNING	DRV_STRENGTH_SEL		SIGNALING_EN	UHS_MODE_SEL		
rw	rw	rw	rw	rw	rw	r	rw	rw	rw	rw		rw	rw		

Field	Bits	Type	Description
UHS_MODE_SEL	2:0	rw	**eMMC Speed Mode Select - MODE_SEL** This field is reserved and needs to be written to with '000'. 000_B Standard setting 001_B reserved 010_B Reserved … 111_B Reserved
SIGNALING_EN	3	rw	**1.8 V Signaling Enable** This bit must remain '0'.
DRV_STRENGTH_SEL	5:4	rw	**Driver Strength Select** This bitfield can be ignored.
EXEC_TUNING	6	rw	**Execute Tuning** This bit must remain '0'.
SAMPLE_CLK_SEL	7	rw	**Sampling Clock Select** This bit must remain '0'.
UHS2_IF_ENABLE	8	rw	**UHS-II Interface Enable** This bit must remain '0'.
RSVD_9	9	r	**Reserved** This bit is reserved. It will always return 0.
ADMA2_LEN_MODE	10	rw	**ADMA2 Length Mode** This bit selects ADMA2 Length mode to be either 16-bit or 26-bit. 0_B 16-bit Data Length Mode 1_B 26-bit Data Length Mode
CMD23_ENABLE	11	rw	**CMD23 Enable** If the card supports CMD23, this bit is set to 1. This bit is used to select Auto CMD23 or Auto CMD12 for ADMA3 data transfer. 0_B Auto CMD23 is disabled 1_B Auto CMD23 is enabled

(table continues...)

(continued)

Field	Bits	Type	Description
HOST_VER4_ENABLE	12	rw	**Host Version 4 Enable** This bit selects either Version 3.00 compatible mode or Version 4 mode. Functions of following fields are modified for Host Version 4 mode: • SDMA Address: SDMA uses ADMA System Address ($05F_H$-058_H) instead of SDMA System Address register (003_H-000_H) • ADMA2/ADMA3 selection: ADMA3 is selected by DMA select in Host Control 1 register • 32-bit Block Count: SDMA System Address register (003_H-000_H) is modified to 32-bit Block Count register ***Note**: It is recommended not to program ADMA3 Integrated Descriptor Address registers while operating in Host version less than 4 mode (Host Version 4 Enable = 0).* 0_B Version 3.00 compatible mode 1_B Version 4 mode
ADDRESSING	13	rw	**64-bit Addressing** This bit is effective when Host Version 4 Enable is set to 1. 0_B 32 bits addressing 1_B Reserved
ASYNC_INT_ENABLE	14	rw	**Asynchronous Interrupt Enable** This bit can be set if a card supports asynchronous interrupts and Asynchronous Interrupt Support is set to 1 in the Capabilities register. 0_B Disabled 1_B Enabled
PRESET_VAL_ENABLE	15	rw	**Preset Value Enable** This bit enables automatic selection of SDCLK frequency and Driver strength Preset Value registers. When Preset Value Enable is set, SDCLK frequency generation (Frequency Select and Clock Generator Select) amd driver strength selection is performed by the controller. These values are selected from set of Preset Value registers based on selected speed mode. During initialization, the host driver needs to manually set the clock preset value to SDCLK/Rclk frequency select. Preset value enable can be set after intialization completed, in accordance with PartA2_SD Host_controller_simplified_specification. 0_B SDCLK and Driver Strength are controlled by Host Driver 1_B Automatic Selection of SDCLK frequency and Driver Strength as Preset Values are Enabled

29.4.44 Capabilities 1 register 0 to 31

This register provides the Host Driver with information specific to the Host Controller implementation. The host controller implements these values as fixed. Capabilities register is segregated into two 32-bit registers: CAPABILITIES1_R and CAPABILITIES2_R. The CAPABILITIES1_R register is the lower part of Capabilities register. Writing this register has no effect.

CAPABILITIES1 Offset address: 440_H

Capabilities 1 register 0 to 31 Kernel Reset value: $216C\ 6483_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SLOT_TYPE		ASYNC_INT_SUPPORT	SYS_ADDR_64_V3	SYS_ADDR_64_V4	VOLT_18	VOLT_30	VOLT_33	SUS_RES_SUPPORT	SDMA_SUPPORT	HIGH_SPEED_SUPPORT	RSVD_20	ADMA2_SUPPORT	Embedded_8_BIT	MAX_BLK_LEN	
r		r	r	r	r	r	r	r	r	r	r	r	r	r	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BASE_CLK_FREQ								TOUT_CLK_UNIT	RSVD_6	TOUT_CLK_FREQ					
r								r	r	r					

Field	Bits	Type	Description
TOUT_CLK_FREQ	5:0	r	**Timeout Clock Frequency** This bit shows the base clock frequency used to detect Data Timeout Error. The Timeout Clock unit defines the unit of the field value: • Timeout Clock Unit = 0 [KHz] unit: 1 KHz to 63 KHz • Timeout Clock Unit = 1 [MHz] unit: 1 MHz to 63 MHz • Not 0 unit: 1 KHz to 63 KHz or 1 MHz to 63 MHz • 00 0000b - Get information through another method
RSVD_6	6	r	**Reserved** This bit is reserved. It will always return 0.
TOUT_CLK_UNIT	7	r	**Timeout Clock Unit** This bit shows the unit of base clock frequency used to detect Data TImeout Error. 0_B kHz 1_B MHz

(table continues...)

(continued)

Field	Bits	Type	Description
BASE_CLK_FREQ	15:8	r	**Base Clock Frequency for SD clock** This value indicates the base (maximum) clock frequency for SD Clock. The definition of this field depends on Host Controller Version. • 6-Bit Base Clock Frequency: This mode is supported by the Host Controller version 1.00 and 2.00. The upper 2 bits are not effective and are always 0. The unit values are 1 MHz. The supported clock range is 10 MHz to 63 MHz. 11xx xxxb - Not Supported 0011 1111b - 63 MHz ..... - ..... 0000 0010 - 2 MHz 0000 001b - 1 MHz 0000 0000 - Get information through another method • 8-Bit Base Clock Frequency: This mode is supported by the Host Controller version 3.00. The unit values are 1 MHz. The supported clock range is 10 MHz to 255 MHz. FFh - 255 MHz ..... - ..... 02h - 2 MHz 01h - 1 MHz 00h - Get information through another method If the frequency is 16.5 MHz, the larger value shall be set to 0001001b (17 MHz) because the Host Driver uses this value to calculate the clock divider value and it shall not exceed the upper limit of the SD Clock frequency. If these bits are all 0, the Host system has to get information using a different method.
MAX_BLK_LEN	17:16	r	**Maximum Block Length** This bit indicates the maximum block size that the Host driver can read and write to the buffer in the Host Controller. The buffer transfers this block size without wait cycles. The transfer block length is always 512 bytes for the SD Memory irrespective of this bit. 00_B 512 Byte 01_B 1024 Byte 10_B 2048 Byte 11_B Reserved
Embedded_8_BIT	18	r	**8-bit Support for Embedded Device** This bit indicates whether the Host Controller is capable of using an 8-bit bus width mode. This bit is not effective when the Slot Type is set to 10b. 0_B 8-bit Bus Width not Supported 1_B 8-bit Bus Width Supported

(table continues...)

(continued)

Field	Bits	Type	Description
ADMA2_SUPPORT	19	r	**ADMA2 Support** This bit indicates whether the Host Controller is capable of using ADMA2. 0_B ADMA2 not Supported 1_B ADMA2 Supported
RSVD_20	20	r	**Reserved** This bit is reserved. It will always return 0.
HIGH_SPEED_SUPPORT	21	r	**High Speed Support** This bit indicates whether the Host Controller and the Host System supports High Speed mode and they can supply the SD Clock frequency from 25 MHz to 50 MHz. 0_B High Speed not Supported 1_B High Speed Supported
SDMA_SUPPORT	22	r	**SDMA Support** This bit indicates whether the Host Controller is capable of using SDMA to transfer data between the system memory and the Host Controller directly. 0_B SDMA not Supported 1_B SDMA Supported
SUS_RES_SUPPORT	23	r	**Suspense/Resume Support** This bit indicates whether the Host Controller supports Suspend/ Resume functionality. If this bit is 0, the Host Driver shall not issue either Suspend or Resume commands because the Suspend and Resume mechanism is not supported. 0_B Not Supported 1_B Supported
VOLT_33	24	r	**Voltage Support 3.3 V** 0_B 3.3 V Not Supported 1_B 3.3 V Supported
VOLT_30	25	r	**Voltage Support 3.0 V** 0_B 3.0 V Not Supported 1_B 3.0 V Supported
VOLT_18	26	r	**Voltage Support 1.8 V** 0_B 1.8 V Not Supported 1_B 1.8 V Supported
SYS_ADDR_64_V4	27	r	**64-bit System Address Support for V4** This bit sets the Host Controller to support 64-bit System Addressing of V4 mode. 0_B 64-bit System Address for V4 is Not Supported 1_B 64-bit System Address for V4 is Supported

(table continues...)

(continued)

Field	Bits	Type	Description
SYS_ADDR_64_V3	28	r	**64-bit System Address Support for V3** This bit sets Host Controller to support 64-bit System Addressing of V3 mode. SDMA cannot be used in 64-bit Addressing in Version 3 Mode. 0_B 64-bit System Address for V3 is Not Supported 1_B 64-bit System Address for V3 is Supported
ASYNC_INT_SUPPORT	29	r	**Asynchronous Interrupt Support (SD Mode only)** 0_B Asynchronous Interrupt Not Supported 1_B Asynchronous Interrupt Supported
SLOT_TYPE	31:30	r	**Slot Type** This field indicates usage of a slot by a specific Host System. 00_B Removable Card Slot 01_B Embedded Slot for one Device 10_B Shared Bus Slot (SD mode) 11_B Reserved

29.4.45 Capabilities register 32 to 63

This register provides the Host Driver with information specific to the Host Controller implementation. The host controller implements these values as fixed. Capabilities register is segregated into two 32-bit registers, namely CAPABILITIES1_R and CAPABILITIES2_R. The CAPABILITIES2_R register is upper part of Capabilities register. Writing this register has no effect.

CAPABILITIES2 Offset address: 444_H

Capabilities register 32 to 63 Kernel Reset value: $0800\ 8001_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVD_62_63		RSVD_61	VDD2_18V_SUPPORT	ADMA3_SUPPORT	RSVD_56_58			CLK_MUL							
r		r	r	r	r			r							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RE_TUNING_MODES		USE_TUNING_SDR50	RSVD_44	RETUNE_CNT				RSVD_39	DRV_TYPED	DRV_TYPEC	DRV_TYPEA	UHS2_SUPPORT	DDR50_SUPPORT	SDR104_SUPPORT	SDR50_SUPPORT
r		r	r	r				r	r	r	r	r	r	r	r

Field	Bits	Type	Description
SDR50_SUPPORT	0	r	**SDR50 Support** If SDR104 is supported, this bit shall be set to 1. Bit 45 indicates whether SDR50 requires tuning or not. 0_B SDR50 is not supported 1_B SDR50 is supported
SDR104_SUPPORT	1	r	**SDR104 Support** SDR104 requires tuning. 0_B SDR104 is not supported 1_B SDR104 is supported
DDR50_SUPPORT	2	r	**DDR50 Support** 0_B DDR50 is not supported 1_B DDR50 is supported
UHS2_SUPPORT	3	r	**UHS-II Support** This bit indicates whether Host Controller supports UHS-II. 0_B UHS-II is not supported 1_B UHS-II is supported
DRV_TYPEA	4	r	**Driver Type A Support** This bit indicates support of Driver Type A for 1.8V Signaling. 0_B Driver Type A is not supported 1_B Driver Type A is supported
DRV_TYPEC	5	r	**Driver Type C Support** This bit indicates support of Driver Type C for 1.8 V Signaling. 0_B Driver Type C is not supported 1_B Driver Type C is supported
DRV_TYPED	6	r	**Driver Type D Support** This bit indicates support of Driver Type D for 1.8 V Signaling. 0_B Driver Type D is not supported 1_B Driver Type D is supported
RSVD_39	7	r	**Reserved** This bit is reserved. It will always return 0.

(table continues...)

(continued)

Field	Bits	Type	Description
RETUNE_CNT	11:8	r	**Timer Count for Re-Tuning** 0_H Re-Tuning Timer disabled 1_H 1 seconds 2_H 2 seconds 3_H 2^2 seconds … B_H 2^10 seconds C_H Reserved … E_H Reserved F_H Get information from other source
RSVD_44	12	r	**Reserved** This bit is reserved. It will always return 0.
USE_TUNING_SDR50	13	r	**Use Tuning for SDR50** 0_B SDR50 does not require tuning 1_B SDR50 requires tuning
RE_TUNING_MODES	15:14	r	**Re-Tuning Modes** This field selects re-tuning method and limits the maximum data length. 00_B Mode 1 - Timer 01_B Mode 2 - Timer and Re-Tuning Request 10_B Mode 3 - Auto Re-Tuning (for transfer) Timer and Re-Tuning Request 11_B Reserved
CLK_MUL	23:16	r	**Clock Multiplier** This field indicates clock multiplier of programmable clock generator. Setting to 0 means that Host Controller does not support programmable clock generator. 00_H Clock Multiplier is not Supported 01_H Clock Multiplier M = 2 02_H Clock Multiplier M = 3 … FF_H Clock Multiplier M = 256
RSVD_56_58	26:24	r	**Reserved** These bits are reserved. They will always return 0.
ADMA3_SUPPORT	27	r	**ADMA3 Support** This bit indicates whether the Host Controller is capable of using ADMA3. 0_B ADMA3 not Supported 1_B ADMA3 Supported

(table continues...)

(continued)

Field	Bits	Type	Description
VDD2_18V_SUPPORT	28	r	**1.8 V VDD2 Support** This bit indicates support of VDD2 for Host System. 0_B 1.8 V VDD2 is not Supported 1_B 1.8 V VDD2 is Supported
RSVD_61	29	r	**Reserved** This bit is reserved. It will always return 0.
RSVD_62_63	31:30	r	**Reserved** These bits are reserved. They will always return 0.

29.4.46 Maximum current capabilities register 0 to 31

For VDD1, these registers indicate maximum current capability for each voltage. The value is meaningful if Voltage Support is set in the Capabilities register. If this information is supplied by the Host System through another method, all Maximum Current Capabilities register shall be 0. Writing this register has no effect.

CURR_CAPABILITIES1 Offset address: 448_H

Maximum current capabilities register 0 to 31 Kernel Reset value: $0000\ 0002_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVD_31_24								MAX_CUR_18V							
r								r							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MAX_CUR_30V								MAX_CUR_33V							
r								r							

Field	Bits	Type	Description
MAX_CUR_33V	7:0	r	**Maximum Current for 3.3 V** 00_H Get Information through another method 01_H 4 mA 02_H 8 mA … FF_H 1020 mA
MAX_CUR_30V	15:8	r	**Maximum Current for 3.0 V** 00_H Get Information through another method 01_H 4 mA 02_H 8 mA … FF_H 1020 mA

(table continues...)

(continued)

Field	Bits	Type	Description
MAX_CUR_18V	23:16	r	**Maximum Current for 1.8 V** 00_H Get Information through another method 01_H 4 mA 02_H 8 mA … FF_H 1020 mA
RSVD_31_24	31:24	r	**Reserved** These bits are reserved. They will always return 0.

29.4.47 Maximum current capabilities register 32 to 63

For VDD2, these registers indicate maximum current capability for each voltage. The value is meaningful if Voltage Support is set in the Capabilities register. If this information is supplied by the Host System through another method, all Maximum Current Capabilities register shall be 0. Writing this register has no effect.

CURR_CAPABILITIES2 Offset address: $44C_H$

Maximum current capabilities register 32 to 63 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVD_63_40															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD_63_40								MAX_CUR_VDD2_18V							
r								r							

Field	Bits	Type	Description
MAX_CUR_VDD2_18V	7:0	r	**Maximum Current for 1.8 V VDD2** 00_H Get Information through another method 01_H 4 mA 02_H 8 mA … FF_H 1020 mA
RSVD_63_40	31:8	r	**Reserved** These bits are reserved. They will always return 0.

29.4.48 Force event register for auto CMD error status register

The register is not physically implemented, but is an address at which the Auto CMD Error Status register can be written. This register is applicable for an SD/eMMC mode.

- 1 : Sets each bit of the Auto CMD Error Status register
- 0 : No effect

FORCE_AUTO_CMD_STAT Offset address: 450_H

Force event register for auto CMD error status register Kernel Reset value: 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD_15_8								FORCE_CMD_NOT_ISSUED_AUTO_CMD12	RSVD_6	FORCE_AUTO_CMD_RESP_ERR	FORCE_AUTO_CMD_IDX_ERR	FORCE_AUTO_CMD_EBIT_ERR	FORCE_AUTO_CMD_CRC_ERR	FORCE_AUTO_CMD_TOUT_ERR	FORCE_AUTO_CMD12_NOT_EXEC
w								w	w	w	w	w	w	w	w

Field	Bits	Type	Description
FORCE_AUTO_CMD12_NOT_EXEC	0	w	**Force Event for Auto CMD12 Not Executed** 0_B Not Affected 1_B Auto CMD12 Not Executed Status is set
FORCE_AUTO_CMD_TOUT_ERR	1	w	**Force Event for Auto CMD Timeout Error** 0_B Not Affected 1_B Force Event for Auto CMD Timeout Error Status is set
FORCE_AUTO_CMD_CRC_ERR	2	w	**Force Event for Auto CMD CRC Error** 0_B Not Affected 1_B Force Event for Auto CMD CRC Error Status is set
FORCE_AUTO_CMD_EBIT_ERR	3	w	**Force Event for Auto CMD End Bit Error** 0_B Not Affected 1_B Force Event for Auto CMD End Bit Error Status is set
FORCE_AUTO_CMD_IDX_ERR	4	w	**Force Event for Auto CMD Index Error** 0_B Not Affected 1_B Force Event for Auto CMD Index Error Status is set
FORCE_AUTO_CMD_RESP_ERR	5	w	**Force Event for Auto CMD Response Error** 0_B Not Affected 1_B Force Event for Auto CMD Response Error Status is set
RSVD_6	6	w	**Reserved** This bit is reserved. It will always return 0.
FORCE_CMD_NOT_ISSUED_AUTO_CMD12	7	w	**Force Event for Command Not Issued By Auto CMD12 Error** 0_B Not Affected 1_B Command Not Issued By Auto CMD12 Error Status is set
RSVD_15_8	15:8	w	**Reserved** These bits are reserved. They will always return 0.

29.4.49 Force event register for error interrupt status

This register is not physically implemented, but is an address at which the Error Interrupt Status register can be written. The effect of a write to this address will be reflected in the Error Interrupt Status register if the corresponding bit of the Error Interrupt Status Enable register is set. This register is applicable for an SD/eMMC mode.

- 1 - Sets each bit of the Error Interrupt Status register
- 0 - No effect

FORCE_ERROR_INT_STAT Offset address: 452_H

Force event register for error interrupt status Kernel Reset value: 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FORCE_VENDOR_ERR3	FORCE_VENDOR_ERR2	FORCE_VENDOR_ERR1	FORCE_BOOT_ACK_ERR	FORCE_RESP_ERR	FORCE_TUNING_ERR	FORCE_ADMA_ERR	FORCE_AUTO_CMD_ERR	FORCE_CUR_LMT_ERR	FORCE_DATA_END_BIT_ERR	FORCE_DATA_CRC_ERR	FORCE_DATA_TOUT_ERR	FORCE_CMD_IDX_ERR	FORCE_CMD_END_BIT_ERR	FORCE_CMD_CRC_ERR	FORCE_CMD_TOUT_ERR
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

Field	Bits	Type	Description
FORCE_CMD_TOUT_ERR	0	w	**Force Event for Command Timeout Error (SD/eMMC Mode only)** 0_B Not Affected 1_B Command Timeout Error Status is set
FORCE_CMD_CRC_ERR	1	w	**Force Event for Command CRC Error (SD/eMMC Mode only)** 0_B Not Affected 1_B Command CRC Error Status is set
FORCE_CMD_END_BIT_ERR	2	w	**Force Event for Command End Bit Error (SD/eMMC Mode only)** 0_B Not Affected 1_B Command End Bit Error Status is set
FORCE_CMD_IDX_ERR	3	w	**Force Event for Command Index Error (SD/eMMC Mode only)** 0_B Not Affected 1_B Command Index Error Status is set
FORCE_DATA_TOUT_ERR	4	w	**Force Event for Data Timeout Error (SD/eMMC Mode only)** 0_B Not Affected 1_B Data Timeout Error Status is set
FORCE_DATA_CRC_ERR	5	w	**Force Event for Data CRC Error (SD/eMMC Mode only)** 0_B Not Affected 1_B Data CRC Error Status is set
FORCE_DATA_END_BIT_ERR	6	w	**Force Event for Data End Bit Error (SD/eMMC Mode only)** 0_B Not Affected 1_B Data End Bit Error Status is set

(table continues...)

(continued)

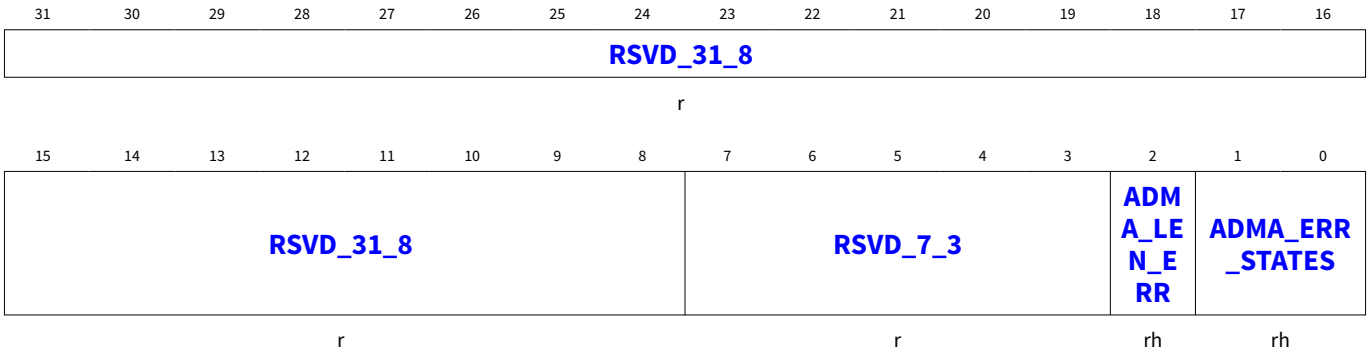
Field	Bits	Type	Description
FORCE_CUR_LMT_ERR	7	w	**Force Event for Current Limit Error** 0_B Not Affected 1_B Current Limit Error Status is set
FORCE_AUTO_CMD_ERR	8	w	**Force Event for Auto CMD Error (SD/eMMC Mode only)** 0_B Not Affected 1_B Auto CMD Error Status is set
FORCE_ADMA_ERR	9	w	**Force Event for ADMA Error** 0_B Not Affected 1_B ADMA Error Status is set
FORCE_TUNING_ERR	10	w	**Force Event for Tuning Error** This bit must remain '0'.
FORCE_RESP_ERR	11	w	**Force Event for Response Error (SD Mode only)** 0_B Not Affected 1_B Response Error Status is set
FORCE_BOOT_ACK_ERR	12	w	**Force Event for Boot Ack error.** 0_B Not Affected 1_B Vendor Specific Error Status is set
FORCE_VENDOR_ERR1	13	w	**Reserved** This bit is reserved. It will always return 0. 0_B Not Affected 1_B Vendor Specific Error Status is set
FORCE_VENDOR_ERR2	14	w	**Reserved** This bit is reserved. It will always return 0. 0_B Not Affected 1_B Vendor Specific Error Status is set
FORCE_VENDOR_ERR3	15	w	**Reserved** This bit is reserved. It will always return 0. 0_B Not Affected 1_B Vendor Specific Error Status is set

29.4.50 ADMA error status register

This register holds the ADMA state during ADMA error. This register is applicable for an SD/eMMC mode. Writing this register has no effect.

ADMA_ERR_STAT Offset address: 454_H

ADMA error status register Kernel Reset value: $0000\ 0000_H$



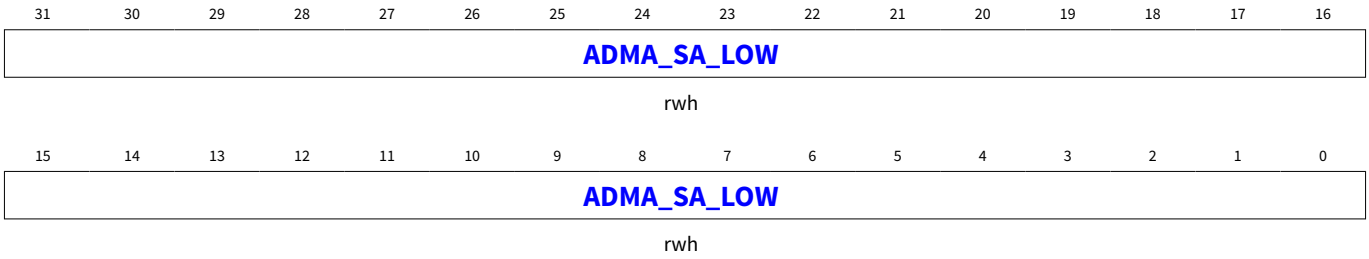
Field	Bits	Type	Description
ADMA_ERR_STATES	1:0	rh	**ADMA Error States** These bits indicate the state of ADMA when an error occurs during ADMA data transfer. 00_B Stop DMA (SYS_ADR register points to a location next to the error descriptor) 01_B Fetch Descriptor (SYS_ADR register points to the error descriptor) 10_B Never set this state (Not used) 11_B Transfer Data (SYS_ADR register points to a location next to the error descriptor)
ADMA_LEN_ERR	2	rh	**ADMA Length Mismatch Error States** This error occurs in the following instances: • While Block Count Enable is being set, the total data length specified by the Descriptor table is different from that specified by the Block Count and Block Length • Total data length cannot be divided by the block length 0_B No Error 1_B Error
RSVD_7_3	7:3	r	**Reserved** These bits are reserved. Read operations may return nonzero values. They shall be ignored.
RSVD_31_8	31:8	r	**Reserved** These bits are reserved. They will always return 0. Volatile: true

29.4.51 ADMA system address register low

This register holds the lower 32-bit system address for DMA transfer. This register is applicable for an SD/eMMC mode.

ADMA_SA_LOW Offset address: 458_H

ADMA system address register low Kernel Reset value: $0000\ 0000_H$



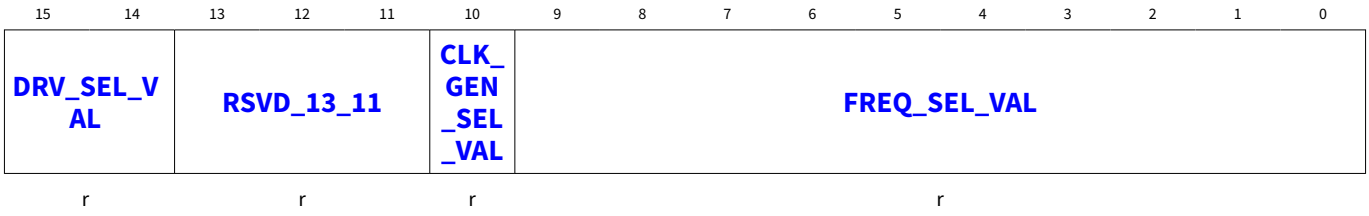
Field	Bits	Type	Description
ADMA_SA_LOW	31:0	rwh	**ADMA System Address** These bits indicate the lower 32 bits of the ADMA system address. • SDMA If Host Version 4 Enable is set to 1, this register holds the system address of the data location • ADMA2 This register holds byte address of executing command of the descriptor table • ADMA3 This register is set by ADMA3. ADMA2 increments the address of this register that points to the next line, every time a Descriptor line is fetched.

29.4.52 Preset value for initialization

This register defines Preset Value for Initialization in SD/eMMC mode. Writing this register has no effect.

PRESET_INIT Offset address: 460_H

Preset value for initialization Kernel Reset value: $007F_H$



Field	Bits	Type	Description
FREQ_SEL_VAL	9:0	r	**SDCLK/RCLK Frequency Select Value** 10-bit preset value to be set in SDCLK/RCLK frequency select field of the clock control kernel register described by a Host System.
CLK_GEN_SEL_VAL	10	r	**Clock Generator Select Value** This bit is effective when the Host Controller supports a programmable clock generator. 0_B Host Controller Ver2.0 Compatible Clock Generator 1_B Programmable Clock Generator

(table continues...)

(continued)

Field	Bits	Type	Description
RSVD_13_11	13:11	r	**Reserved** These bits are reserved. They will always return 0.
DRV_SEL_VAL	15:14	r	**Driver Strength Select Value** Driver strength is supported by 1.8V signaling bus speed modes. This field is meaningless for 3.3V signaling. 00_B Driver Type B is selected 01_B Driver Type A is selected 10_B Driver Type C is selected 11_B Driver Type D is selected

29.4.53 Preset value for default speed

This register defines Preset Value for Default Speed mode in SD mode. Writing this register has no effect.

PRESET_DS Offset address: 462_H

Preset value for default speed Kernel Reset value: 0001_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DRV_SEL_VAL		RSVD_13_11			CLK_GEN_SEL_VAL	FREQ_SEL_VAL									
r		r			r	r									

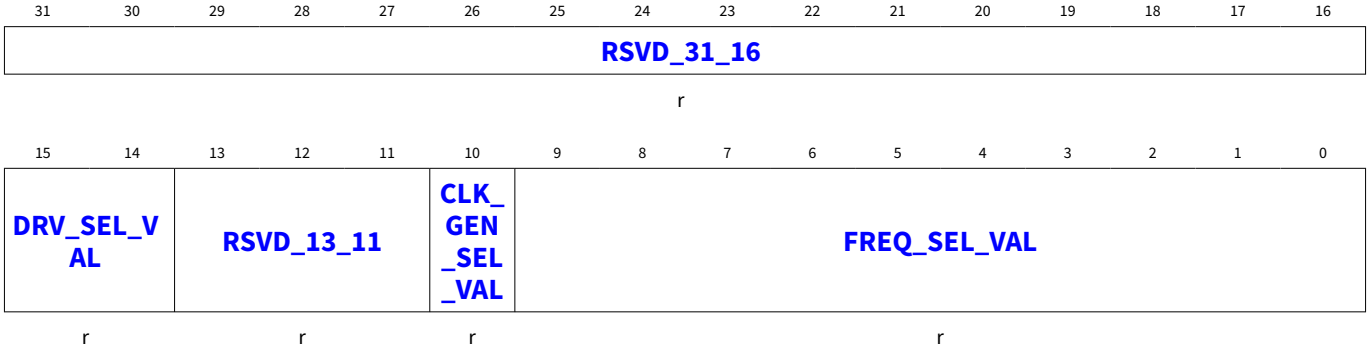
Field	Bits	Type	Description
FREQ_SEL_VAL	9:0	r	**SDCLK/RCLK Frequency Select Value** 10-bit preset value to be set in SDCLK/RCLK frequency select field of the clock control kernel register described by a host System.
CLK_GEN_SEL_VAL	10	r	**Clock Generator Select Value** This bit is effective when Host Controller supports programmable clock generator. 0_B Host Controller Ver2.0 Compatible Clock Generator 1_B Programmable Clock Generator
RSVD_13_11	13:11	r	**Reserved** These bits are reserved. They will always return 0.
DRV_SEL_VAL	15:14	r	**Driver Strength Select Value** Driver strength is supported by 1.8 V signaling bus speed modes. This field is meaningless for 3.3 V signaling. **Value After Reset:** 0_H 00_B Driver Type B is selected 01_B Driver Type A is selected 10_B Driver Type C is selected 11_B Driver Type D is selected

29.4.54 Preset value for high speed

This register defines Preset Value for High Speed mode in SD mode. Writing this register has no effect.

PRESET_HS Offset address: 464_H

Preset value for high speed Kernel Reset value: $0000\ 0000_H$



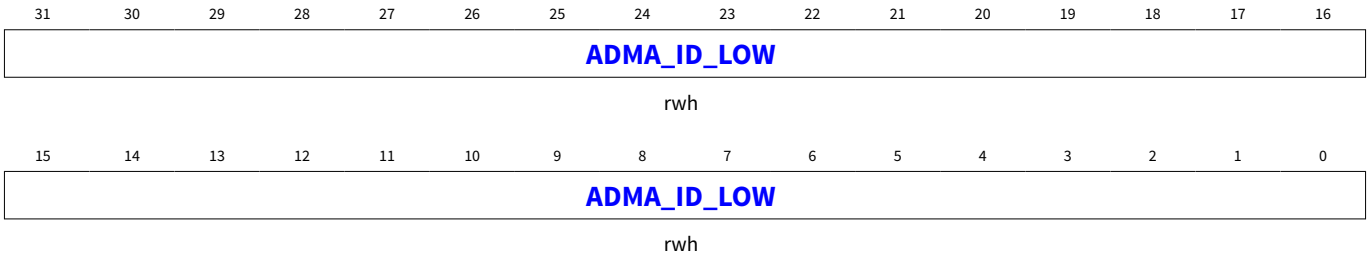
Field	Bits	Type	Description
FREQ_SEL_VAL	9:0	r	**SDCLK/RCLK Frequency Select Value** 10-bit preset value to be set in SDCLK/RCLK frequency select field of the clock control kernel register described by a Host System.
CLK_GEN_SEL_VAL	10	r	**Clock Generator Select Value** This bit is effective when Host Controller supports programmable clock generator. 0_B Host Controller Ver2.0 Compatible Clock Generator 1_B Programmable Clock Generator
RSVD_13_11	13:11	r	**Reserved** These bits are reserved. They will always return 0.
DRV_SEL_VAL	15:14	r	**Driver Strength Select Value** Driver strength is supported by 1.8 V signaling bus speed modes. This field is meaningless for 3.3 V signaling. 00_B Driver Type B is selected 01_B Driver Type A is selected 10_B Driver Type C is selected 11_B Driver Type D is selected
RSVD_31_16	31:16	r	**Reserved** These bits are reserved. They will always return 0.

29.4.55 ADMA3 integrated descriptor address register - low

This register holds the lower 32-bit Integrated Descriptor address.This register is applicable for an SD/eMMC mode.

ADMA_ID_LOW Offset address: 478_H

ADMA3 integrated descriptor address register - low Kernel Reset value: $0000\ 0000_H$



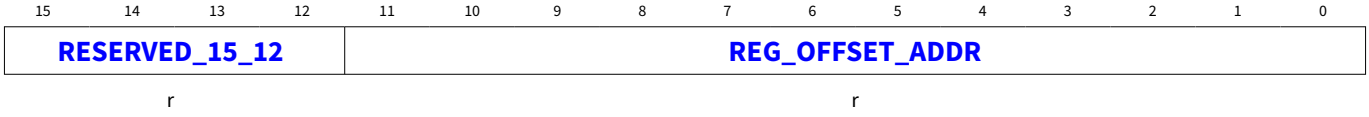
Field	Bits	Type	Description
ADMA_ID_LOW	31:0	rwh	**ADMA Integrated Descriptor Address** This bit indicates the lower 32-bit of the ADMA Integrated Descriptor address. The start address of Integrated Descriptor is set to this register. The ADMA3 fetches one Descriptor Address and increments this field to indicate the next Descriptor address.

29.4.56 Pointer for vendor specific area 1

This register used as a pointer for the Vendor Specific Area 1. Writing this register has no effect.

P_VENDOR_SPECIFIC_AREA Offset address: $4E8_H$

Pointer for vendor specific area 1 Kernel Reset value: 0180_H



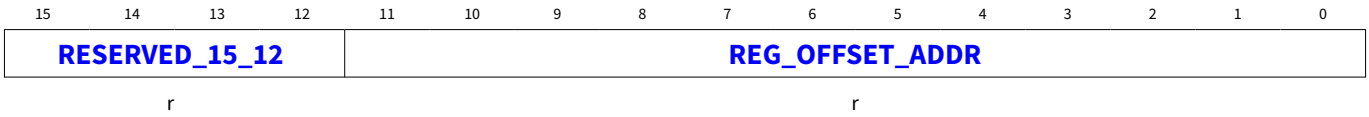
Field	Bits	Type	Description
REG_OFFSET_ADDR	11:0	r	**Base offset Address for Vendor Specific registers.**
RESERVED_15_12	15:12	r	**Reserved** These bits are reserved. They will always return 0.

29.4.57 Pointer for vendor specific area 2

This register is used as a pointer for the Vendor Specific Area 2. Writing this register has no effect.

P_VENDOR2_SPECIFIC_AREA Offset address: $4EA_H$

Pointer for vendor specific area 2 Kernel Reset value: 0300_H



Field	Bits	Type	Description
REG_OFFSET_ADDR	11:0	r	**Base offset Address for Command Queuing registers.**

(table continues...)

(continued)

Field	Bits	Type	Description
RESERVED_15_12	15:12	r	**Reserved** These bits are reserved. They will always return 0.

29.4.58 Slot interrupt status register

This register indicate the Interrupt status of each slot. Writing this register has no effect.

SLOT_INTR_STATUS Offset address: $4FC_H$

Slot interrupt status register Kernel Reset value: 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED_15_8								INTR_SLOT							
r								rh							

Field	Bits	Type	Description
INTR_SLOT	7:0	rh	**Interrupt signal for each Slot** These status bits indicate the logical OR of Interrupt signal and Wakeup signal for each slot. A maximum of 8 slots can be defined. If one interrupt signal is associated with multiple slots, the Host Driver can know which interrupt is generated by reading these status bits. By a power on reset or by setting Software Reset For All, the interrupt signal are de-asserted and this status reads 00h. 01_H Bit 0 - Slot 1 02_H Bit 1 - Slot 2 04_H Bit 2 - Slot 3 08_H Bit 3 - Slot 4 10_H Bit 4 - Slot 5 20_H Bit 5 - Slot 6 40_H Bit 6 - Slot 7 80_H Bit 7 - Slot 8
RESERVED_15_8	15:8	r	**Reserved** These bits are reserved. They will always return 0. **Volatile:** true

29.4.59 Host controller version

This register is used to indicate the Host Controller Version number. Writing this register has no effect.

HOST_CNTRL_VERS Offset address: $4FE_H$

Host controller version Kernel Reset value: 1005_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VENDOR_VERSION_NUM								SPEC_VERSION_NUM							
r								r							

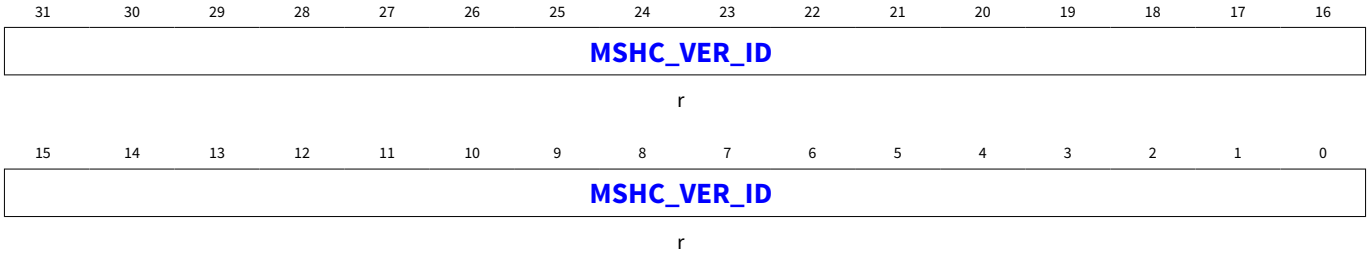
Field	Bits	Type	Description
SPEC_VERSION_NUM	7:0	r	**Specification Version Number** This field indicates Host controller specification version. The upper and lower 4-bits indicate the version. 00_H SD Host Controller Specification Version 1.00 01_H SD Host Controller Specification Version 2.00 02_H SD Host Controller Specification Version 3.00 03_H SD Host Controller Specification Version 4.00 04_H SD Host Controller Specification Version 4.10 05_H SD Host Controller Specification Version 4.20 **others,** Reserved
VENDOR_VERSION_NUM	15:8	r	**Vendor Version Number** This field is reserved for the vendor version number. Host Driver should not use this status.

29.4.60 MSHC version

This register reflects the current release number of SDMMC.

MSHC_VER_ID Offset address: 580_H

MSHC version Kernel Reset value: $3139\ 302A_H$



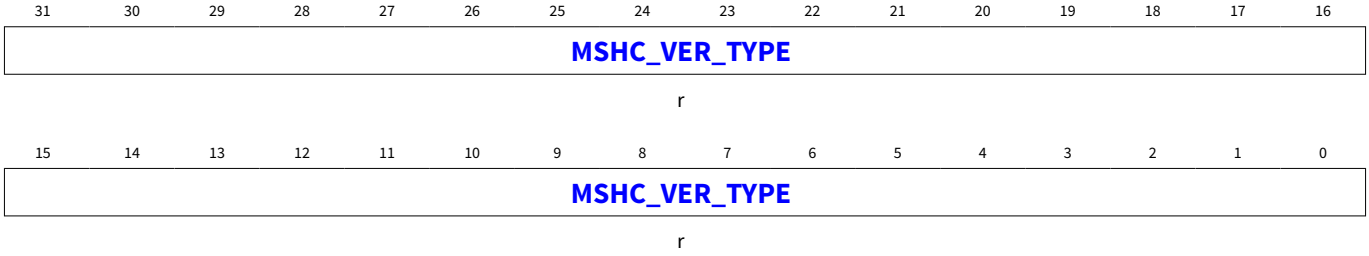
Field	Bits	Type	Description
MSHC_VER_ID	31:0	r	**Current release number** This field indicates the Synopsys DesignWare Cores SDMMC current release number that is read by an application. Current release is 1.80a which is represented in ASCII as 0x313830. Lower 8 bits read from this register can be ignored by the application. Application reading this register in conjunction with MSHC_VER_TYPE_R will gather details of the current release.

29.4.61 MSHC version type

This register reflects the current release type of SDMMC.

MSHC_VER_TYPE Offset address: 584_H

MSHC version type Kernel Reset value: $6561\ 3030_H$



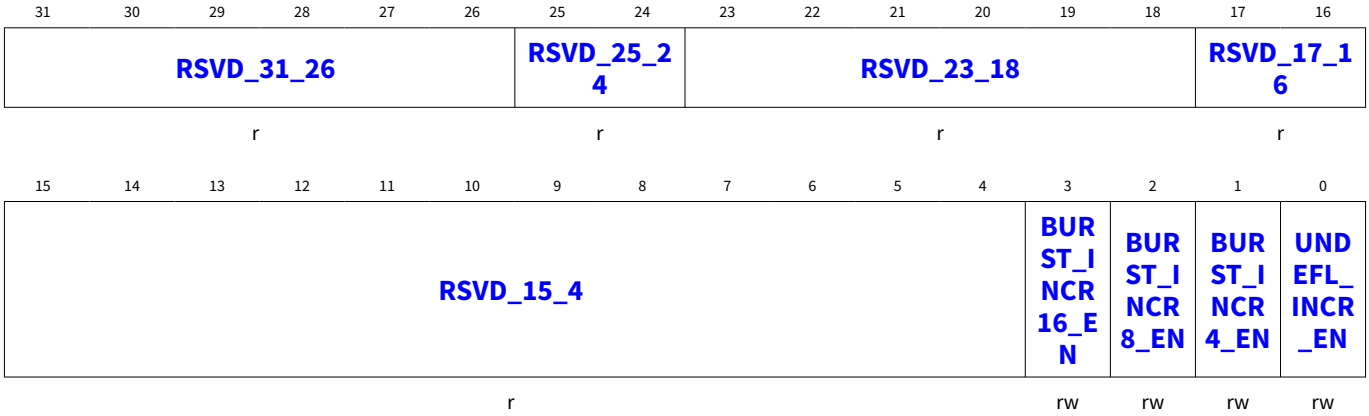
Field	Bits	Type	Description
MSHC_VER_TYPE	31:0	r	**Current release type** This field indicates the Synopsys DesignWare Cores SDMMC current release type that is read by an application. Current release is of type "ga", which is represented in ASCII as 0x6761. Lower 16 bits read from this register can be ignored by the application. Application reading this register in conjunction with MSHC_VER_ID_R will gather details of the current release.

29.4.62 DMA burst control register

This register is used to control the Master bus interface.

MBIU_CTRL Offset address: 590_H

DMA burst control register Kernel Reset value: $0303\ 0007_H$



Field	Bits	Type	Description
UNDEFL_INCR_EN	0	rw	**Controls the generation of undefined length INCR transfer on master interface** Must be disabled for usage of INCR4 and INCR8 bursts. 0_B INCR type bursts are not preferrend on AHB Master interface 1_B INCR type bursts are preferrend on AHB Master interface
BURST_INCR4_EN	1	rw	**Controls generation of INCR4 transfers on master interface** 0_B INCR bursts of length 4 are not generated on the master interface 1_B INCR bursts of length 4 can be generated on the master interface

(table continues...)

(continued)

Field	Bits	Type	Description
BURST_INCR8_EN	2	rw	**Controls generation of INCR8 transfers on master interface** 0_B INCR bursts of length 8 are not generated on the master interface 1_B INCR bursts of length 8 can be generated on the master interface
BURST_INCR16_EN	3	rw	**Controls generation of INCR16 transfers on master interface** Not supported. Must be disabled. 0_B INCR bursts of length 16 are not generated on the master interface 1_B INCR bursts of length 16 can be generated on the master interface
RSVD_15_4	15:4	r	**Reserved - RSVD_7_4** These bits are reserved. They will always return 0.
RSVD_17_16	17:16	r	**Reserved** Write '1' to read '1'.
RSVD_23_18	23:18	r	**Reserved - RSVD_31_8** These bits are reserved. They will always return 0.
RSVD_25_24	25:24	r	**Reserved - RSVD_7_4** Write '1' to read '1'.
RSVD_31_26	31:26	r	**Reserved - RSVD_31_8** These bits are reserved. They will always return 0.

29.4.63 eMMC control register

This register is used to control the eMMC operation.

EMMC_CTRL Offset address: $5AC_H$

eMMC control register Kernel Reset value: $000C_H$

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD												RSVD_3	RSVD_2	DISABLE_DATA_CRC_CHK	CARD_IS_EMMC
r												rw	rw	rw	rw

Field	Bits	Type	Description
CARD_IS_EMMC	0	rw	**eMMC Card present** This bit indicates the type of card connected. An application program this bit based on the card connected to SDMMC. 0_B **NON_EMMC_CARD**: Card connected to is not eMMC card 1_B **EMMC_CARD**: Card connected is eMMC card

(table continues...)

(continued)

Field	Bits	Type	Description
DISABLE_DATA_CRC_CHK	1	rw	**Disable Data CRC Check** This bit controls masking of CRC16 error for Card Write in eMMC mode. This is useful in bus testing (CMD19) for an eMMC device. In bus testing, an eMMC card does not send CRC status for a block, which may generate CRC error. This CRC error can be masked using this bit during bus testing. 0_B **ENABLE**: DATA CRC check is enabled 1_B **DISABLE**: DATA CRC check is disabled
RSVD_2	2	rw	**Reserved** This bit is reserved. Read as 1 and shall be written with 1.
RSVD_3	3	rw	**Reserved** This bit is reserved. Read as 1 and shall be written with 1.
RSVD	15:4	r	**Reserved** These bits (RSVD) of the EMMC_CTRL_R register are reserved. They always return 0.

29.4.64 eMMC boot control register

This register is used to control the eMMC Boot operation.

BOOT_CTRL Offset address: $5AE_H$

eMMC boot control register Kernel Reset value: 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BOOT_TOUT_CNT				RSVD_11_9			BOOT_ACK_ENABLE	VALIDATE_BOOT	RSVD_6_1						MAN_BOOT_EN
rw				r			rw	w	r						rwh

Field	Bits	Type	Description
MAN_BOOT_EN	0	rwh	**Mandatory Boot Enable** This bit is used to initiate the mandatory boot operation. The application shall set this bit along with VALIDATE_BOOT bit. Writing 0 is ignored. The SDMMC clears this bit after the boot transfer is completed or terminated. **Testable:** readOnly 0_B Mandatory boot disable 1_B Mandatory boot enable
RSVD_6_1	6:1	r	**Reserved** These bits are reserved. They will always return 0.

(table continues...)

(continued)

Field	Bits	Type	Description
VALIDATE_BOOT	7	w	**Validate Mandatory Boot Enable bit** This bit is used to validate the MAN_BOOT_EN bit. 0_B Ignore Mandatory boot Enable bit 1_B Validate Mandatory boot enable bit
BOOT_ACK_ENABLE	8	rw	**Boot Acknowledge Enable** When this bit set, SDMMC checks for boot acknowledge start pattern of 0-1-0 during boot operation. This bit is applicable for both mandatory and alternate boot mode. 0_B Boot Ack disable 1_B Boot Ack enable
RSVD_11_9	11:9	r	**Reserved** These bits are reserved. They will always return 0.
BOOT_TOUT_CNT	15:12	rw	**Boot Ack Timeout Counter Value.** This value determines the interval by which boot ack timeout (50 ms) is detected when boot ack is expected during boot operation. 0_H TMCLK * 2^13 1_H TMCLK * 2^14 2_H TMCLK * 2^15 … E_H TMCLK * 2^27 F_H Reserved

29.4.65 Embedded control register

This register is control embedded device. When host controller is connected to removable device this register is not used.

EMBEDDED_CTRL Offset address: 680_H

Embedded control register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVD_31	BACK_END_PWR_CTRL							RSVD_23	INT_PIN_SEL			RSVD_19	CLK_PIN_SEL		
r	rw							r	rw			r	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD_15	BUS_WIDTH_PRESET							RSVD_7_6		NUM_INT_PIN		RSVD_3	NUM_CLK_PIN		
r	r							r		r		r	r		

Field	Bits	Type	Description
NUM_CLK_PIN	2:0	r	**Number of Clock Pins (SD Mode)** This field is not used and can be ignored.

(table continues...)

(continued)

Field	Bits	Type	Description
RSVD_3	3	r	**Reserved** This bit is reserved. It will always return 0.
NUM_INT_PIN	5:4	r	**Number of Interrupt Input Pins** This field is not used and can be ignored.
RSVD_7_6	7:6	r	**Reserved** These bits are reserved. They will always return 0.
BUS_WIDTH_PRESET	14:8	r	**Bus Width Preset (SD Mode)** This field is not used and can be ignored.
RSVD_15	15	r	**Reserved** This bit is reserved. It will always return 0.
CLK_PIN_SEL	18:16	rw	**Clock Pin Select (SD Mode)** This field is not used and should not be changed.
RSVD_19	19	r	**Reserved** This bit is reserved. It will always return 0.
INT_PIN_SEL	22:20	rw	**Interrupt Pin Select** This field is not used and should not be changed.
RSVD_23	23	r	**Reserved** This bit is reserved. It will always return 0.
BACK_END_PWR_CTRL	30:24	rw	**Back-End Power Control (SD Mode)** This field is not used and should not be changed.
RSVD_31	31	r	**Reserved** This bit is reserved. It will always return 0.

29.5 Debug information

The SDMMC module does not include debug feature. Correct functionality of SDMMC is not guaranteed if suspend is enabled in other relevant blocks, with which the SDMMC is transacting.

29.6 References

1. JEDEC eMMC 5.1 Specification - JESD84-B51, February 2015
2. SD Specifications Part A2 SD Host Controller Standard Specification Version 4.20, August 2015
3. SD Specifications Part A2 Host Controller Version 4.20 Supplementary Notes Version 1.00 Draft 0.60 December, 2015
4. SD Specifications Part 1 Physical layer Specification Version 6.00 February, 2017
5. SD Specifications Part E1 SDIO Specification Version 4.10 Sept 2014

29.7 SDMMC revision history

Reference	Description of change(s)
Date range: 2024-08-05 to 2025-01-27	
Pin enabling	Changing the SDMMC pin monitor description
Registers	Correcting bitfield access type to rh or rwh to all hardware writable register bitfields

29.8 TC4Dx SDMMC information

29.8.1 TC4Dx SDMMC configuration

There are no deviations from the generic specification.

29.8.2 TC4Dx SDMMC features

There are no deviations from the generic specification.

29.8.3 TC4Dx SDMMC functional description

In TC4Dx device, the SDMMC module is connected to ComPB bus interface.

29.8.4 TC4Dx SDMMC registers

29.8.4.1 Register address space - SDMMC

Table 1234 Registers address space - SDMMC

Module	Base address	End address	Note
SDMMC0	$F46B0000_H$	$F46B0FFF_H$	FPI slave interface

29.8.4.2 Register overview - access mode glossary

Table 1235 Register overview - access mode glossary

Keyword	Description
E	Access protection using PROT register SDMMC0_PROTE .
SE	Access protection using PROT register SDMMC0_PROTSE .
APU-P	Protection group consisting of registers SDMMC0_ACCEN_WRA , SDMMC0_ACCEN_WRB , SDMMC0_ACCEN_RDA , SDMMC0_ACCEN_RDB , SDMMC0_ACCEN_VM , SDMMC0_ACCEN_PRS .
P	Access protection using *APU-P* registers.
BE	Always returns a Bus Error.
U	No access restrictions.
PROT	Access restrictions as defined in the PROT register access rules.

29.8.4.3 Registers overview - SDMMC0 (ascending offset address)

Table 1236 Registers overview - SDMMC0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
SDMMC0_CLC	Clock control register	000_H	P	P, SV, E	Application Reset	5214
SDMMC0_OCS	OCDS control and status register	004_H	P	SV, P	Debug Reset	5215
SDMMC0_ID	Module identification register	008_H	P	BE	PowerOn Reset	5215
SDMMC0_RST_CTRLA	Reset control register A	$00C_H$	P	P, SV, E	Application Reset	5216
SDMMC0_RST_CTRLB	Reset control register B	010_H	P	P, SV, E	Application Reset	5216
SDMMC0_RST_STAT	Reset status register	014_H	P	BE	Application Reset	5217
SDMMC0_PROTE	PROT register endinit	020_H	U	SV, PROT	Application Reset	5218
SDMMC0_PROTSE	PROT register safe endinit	024_H	U	SV, PROT	Application Reset	5219

(table continues...)

Table 1236 (continued) Registers overview - SDMMC0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
SDMMC0_ACCEN_WRA	Write access enable register A	040_H	U	SE, SV	Application Reset	5221
SDMMC0_ACCEN_WRB	Write access enable register B	044_H	U	SE, SV	Application Reset	5221
SDMMC0_ACCEN_RDA	Read access enable register A	048_H	U	SE, SV	Application Reset	5222
SDMMC0_ACCEN_RDB	Read access enable register B	$04C_H$	U	SE, SV	Application Reset	5222
SDMMC0_ACCEN_VM	VM access enable register	050_H	U	SE, SV	Application Reset	5223
SDMMC0_ACCEN_PRS	PRS access enable register	054_H	U	SE, SV	Application Reset	5223
SDMMC0_SDMASA	SDMA system address register	400_H	P	P	Kernel Reset	5224
SDMMC0_BLOCKSIZE	Block size register	404_H	P	P	Kernel Reset	5225
SDMMC0_BLOCKCOUNT	16-bit block count register	406_H	P	P	Kernel Reset	5226
SDMMC0_ARGUMENT	Argument register	408_H	P	P	Kernel Reset	5227
SDMMC0_XFER_MODE	Transfer mode register	$40C_H$	P	P	Kernel Reset	5227
SDMMC0_CMD	Command register	$40E_H$	P	P	Kernel Reset	5230
SDMMC0_RESP01	Response register 01	410_H	P	BE	Kernel Reset	5232
SDMMC0_RESP23	Response register 23	414_H	P	BE	Kernel Reset	5232
SDMMC0_RESP45	Response register 45	418_H	P	BE	Kernel Reset	5233
SDMMC0_RESP67	Response register 67	$41C_H$	P	BE	Kernel Reset	5233
SDMMC0_BUF_DATA	Buffer data port register	420_H	P	P	Kernel Reset	5234
SDMMC0_PSTATE_REG	Present state register	424_H	P	BE	Kernel Reset	5234
SDMMC0_HOST_CTRL1	Host control 1 register	428_H	P	P	Kernel Reset	5237
SDMMC0_PWR_CTRL	Power control register	429_H	P	P	Kernel Reset	5239
SDMMC0_BGAP_CTRL	Block gap control register	$42A_H$	P	P	Kernel Reset	5240

(table continues...)

Table 1236 (continued) Registers overview - SDMMC0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
SDMMC0_WUP_CTRL	Wakeup control register	$42B_H$	P	P	Kernel Reset	5241
SDMMC0_CLK_CTRL	Clock control kernel register	$42C_H$	P	P	Kernel Reset	5242
SDMMC0_TOUT_CTRL	Timeout control register	$42E_H$	P	P	Kernel Reset	5243
SDMMC0_SW_RST	Software reset register	$42F_H$	P	P	Kernel Reset	5244
SDMMC0_NORMAL_INT_STAT	Normal interrupt status register	430_H	P	P	Kernel Reset	5246
SDMMC0_ERROR_INT_STAT	Error interrupt status register	432_H	P	P	Kernel Reset	5248
SDMMC0_NORMAL_INT_STAT_EN	Normal interrupt status enable register	434_H	P	P	Kernel Reset	5252
SDMMC0_ERROR_INT_STAT_EN	Error interrupt status enable register	436_H	P	P	Kernel Reset	5254
SDMMC0_NORMAL_INT_SIGNAL_EN	Normal interrupt signal enable register	438_H	P	P	Kernel Reset	5256
SDMMC0_ERROR_INT_SIGNAL_EN	Error interrupt signal enable register	$43A_H$	P	P	Kernel Reset	5258
SDMMC0_AUTO_CMD_STAT	Auto CMD status register	$43C_H$	P	P	Kernel Reset	5260
SDMMC0_HOST_CTRL2	Host control 2 register	$43E_H$	P	P	Kernel Reset	5261
SDMMC0_CAPABILITIES1	Capabilities 1 register 0 to 31	440_H	P	BE	Kernel Reset	5264
SDMMC0_CAPABILITIES2	Capabilities register 32 to 63	444_H	P	BE	Kernel Reset	5267
SDMMC0_CURR_CAPABILITIES1	Maximum current capabilities register 0 to 31	448_H	P	BE	Kernel Reset	5270
SDMMC0_CURR_CAPABILITIES2	Maximum current capabilities register 32 to 63	$44C_H$	P	BE	Kernel Reset	5271
SDMMC0_FORCE_AUTO_CMD_STAT	Force event register for auto CMD error status register	450_H	P	P	Kernel Reset	5271
SDMMC0_FORCE_ERROR_INT_STAT	Force event register for error interrupt status	452_H	P	P	Kernel Reset	5273
SDMMC0_ADMA_ERR_STAT	ADMA error status register	454_H	P	BE	Kernel Reset	5274

(table continues...)

Table 1236 (continued) Registers overview - SDMMC0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
SDMMC0_ADMA_SA_LOW	ADMA system address register low	458_H	P	P	Kernel Reset	5275
SDMMC0_PRESET_INIT	Preset value for initialization	460_H	P	BE	Kernel Reset	5276
SDMMC0_PRESET_DS	Preset value for default speed	462_H	P	BE	Kernel Reset	5277
SDMMC0_PRESET_HS	Preset value for high speed	464_H	P	BE	Kernel Reset	5278
SDMMC0_ADMA_ID_LOW	ADMA3 integrated descriptor address register - low	478_H	P	P	Kernel Reset	5278
SDMMC0_P_VENDOR_SPECIFIC_AREA	Pointer for vendor specific area 1	$4E8_H$	P	BE	Kernel Reset	5279
SDMMC0_P_VENDOR2_SPECIFIC_AREA	Pointer for vendor specific area 2	$4EA_H$	P	BE	Kernel Reset	5279
SDMMC0_SLOT_INTR_STATUS	Slot interrupt status register	$4FC_H$	P	BE	Kernel Reset	5280
SDMMC0_HOST_CNTRL_VERS	Host controller version	$4FE_H$	P	BE	Kernel Reset	5280
SDMMC0_MSHC_VER_ID	MSHC version	580_H	P	BE	Kernel Reset	5281
SDMMC0_MSHC_VER_TYPE	MSHC version type	584_H	P	BE	Kernel Reset	5281
SDMMC0_MBIU_CTRL	DMA burst control register	590_H	P	P	Kernel Reset	5282
SDMMC0_EMMC_CTRL	eMMC control register	$5AC_H$	P	P	Kernel Reset	5283
SDMMC0_BOOT_CTRL	eMMC boot control register	$5AE_H$	P	P	Kernel Reset	5284
SDMMC0_EMBEDDED_CTRL	Embedded control register	680_H	P	P	Kernel Reset	5285

29.8.4.4 Device specific registers

There are no device specific register changes.

29.8.5 TC4Dx SDMMC connectivity

The table below lists interfaces to and from this functional block to other blocks in the device.

Table 1237 List of SDMMC interface signals

Interface signals	I/O	Description
FPI_SDMMC_SIF	In	Slave interface
SDMMC_SRI_MIF	Out	Master interface
CLOCK_SDMMC_fSDMMC	In	Peripheral clock
SDMMC_IR_ERR	Out	Normal and Error Interrupts
SDMMC_IR_DMA	Out	Wakeup Interrupts
SDMMC_PORTS_CLK	Out	Card Clock
PORTS_SDMMC_CMD_IN	In	Command in
SDMMC_PORTS_CMD	Out	Command out
PORTS_SDMMC_DAT[0:7]_IN	In	Data Receive
SDMMC_PORTS_DAT[0:7]	Out	Data Transmit

29.8.6 TC4Dx SDMMC revision history

Initial release of the chapter.

30 Queued Synchronous Peripheral Interface (QSPI)

QSPI provides synchronous serial communication with external devices using clock, data-in, data-out, and slave select signals. Operation as both master and slave, with flexible configurations and timing options are also supported.

30.1 Feature list

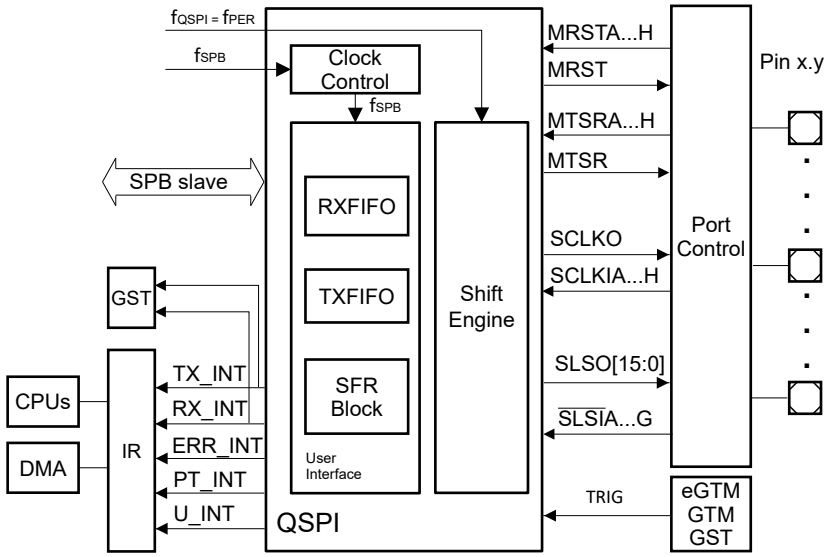
- Master and Slave mode operation
  - Full-duplex operation
  - Half-duplex operation
  - Automatic slave select control
  - Four-wire and three-wire types of connection
- Flexible data format
  - Programmable number of data bits: 2 to 32 data bits (plus parity: 3 to 33 bits)
  - Programmable shift direction: least significant bit (LSB) or most significant bit (MSB) shift first
  - Programmable clock polarity: Idle low or idle high state for the shift clock
  - Programmable clock phase: data shift with leading or trailing edge of the shift clock
- Baud rate generation
  - Baud rates generated from high precision phase locked loop (PLL) clock, asynchronous to the system PLL
- Interrupt generation
  - On a transmitter first-in first-out memory (FIFO) event
  - On a receiver FIFO event
  - On an error condition (receive, baud rate, transmit error, parity error)
  - On a phase transition (At the start of frame for example, or the end of frame)
- Supports control and data handling by the direct memory access (DMA) controller
- Flexible QSPI pin configuration
- Hardware supported parity mode
  - Odd, even, or no parity
- Seven slave select inputs $\overline{\text{SLSIA...G}}$ in Slave mode
- Sixteen programmable slave select outputs SLSO[15:0] in Master mode
  - Automatic SLSO generation with programmable timing
  - Programmable active level and enable control
  - External de-multiplexing of the slave select outputs support
  - Multiple slave select activation (up to 4 slave select outputs)
- Transmission triggering from external timer
- Several module reset options
  - State machine reset per software (only the state machine)
  - Module reset per software (both FIFOs, all registers, and the state machine)
  - Automatic stop option of the state machine in slave mode after baud rate error
- Loop-Back mode
- Interoperability with SSC (Synchronous Serial Channel) and USIC (Universal Serial Interface Channel) modules of Infineon microcontroller families, and with popular (Q)SPI interfaces of multiple suppliers
- Communication stop on RxFIFO full
  - Shift register full and RxFIFO full can pause the communication
  - Interrupt generation

***Note**: Baud rate generation: The maximum baud rate achievable in a particular connection is limited by the timings of both the master and the slave chips, as defined in the corresponding data sheets. Important considerations are the type of Input/Output (I/O) pads used, for example Low Voltage Differential Signalling (LVDS) or Complementary Metal Oxide Semiconductor (CMOS) pads, the I/O power supply voltage, the configured output driver strength, and the allowed electromagnetic radiation and temperature. Due to temperature and voltage change there is variation in loop delay. To compensate that run-time adjustment of sampling point by application software might be required for baud rate higher than 20 Mbit/s*

30.2 Functional overview

The main purpose of the QSPI module is to provide synchronous serial communication with external devices using clock, data-in, data-out and slave select signals. The focus of the module is fast and flexible communication. This can be either point-to-point or master-to-many slaves communication.

Parallel requests from on-chip bus masters to a module will be executed sequentially through the on-chip bus system. A read-modify-write feature provides an atomic read and write sequence where no other master can access the module in between operations. Module hardware semaphores are not supported.



***Note**: In CLOCK chapter this f_{PER} is referenced to as f_{QSPI} for overview purposes*

Figure 645 QSPI block diagram

The figure above shows a high level overview of the QSPI module. The two clock domains of register interface and shift engine are shown, the connection to ports and interrupts. The latter has independent request lines for TX-interrupt (TX_INT), RX-interrupt (RX_INT), Error-interrupt (ERR_INT), Phase transition-interrupt (PT_INT) and User-interrupt (U_INT). These interrupts are connected to the following inputs of the IR: SRC_QSPIwTX, SRC_QSPIwRX, SRC_QSPIwERR, SRC_QSPIwPT, SRC_QSPIwU.

Related information

TC4Dx SMU alarm mapping tables on page 7233

30.2.1 Abstract overview

An abstract overview of the QSPI module is shown in the below picture. This document describes the QSPI module according to this view: the State Machine with its configuration and status capabilities, the User Interface, and the Interrupts.

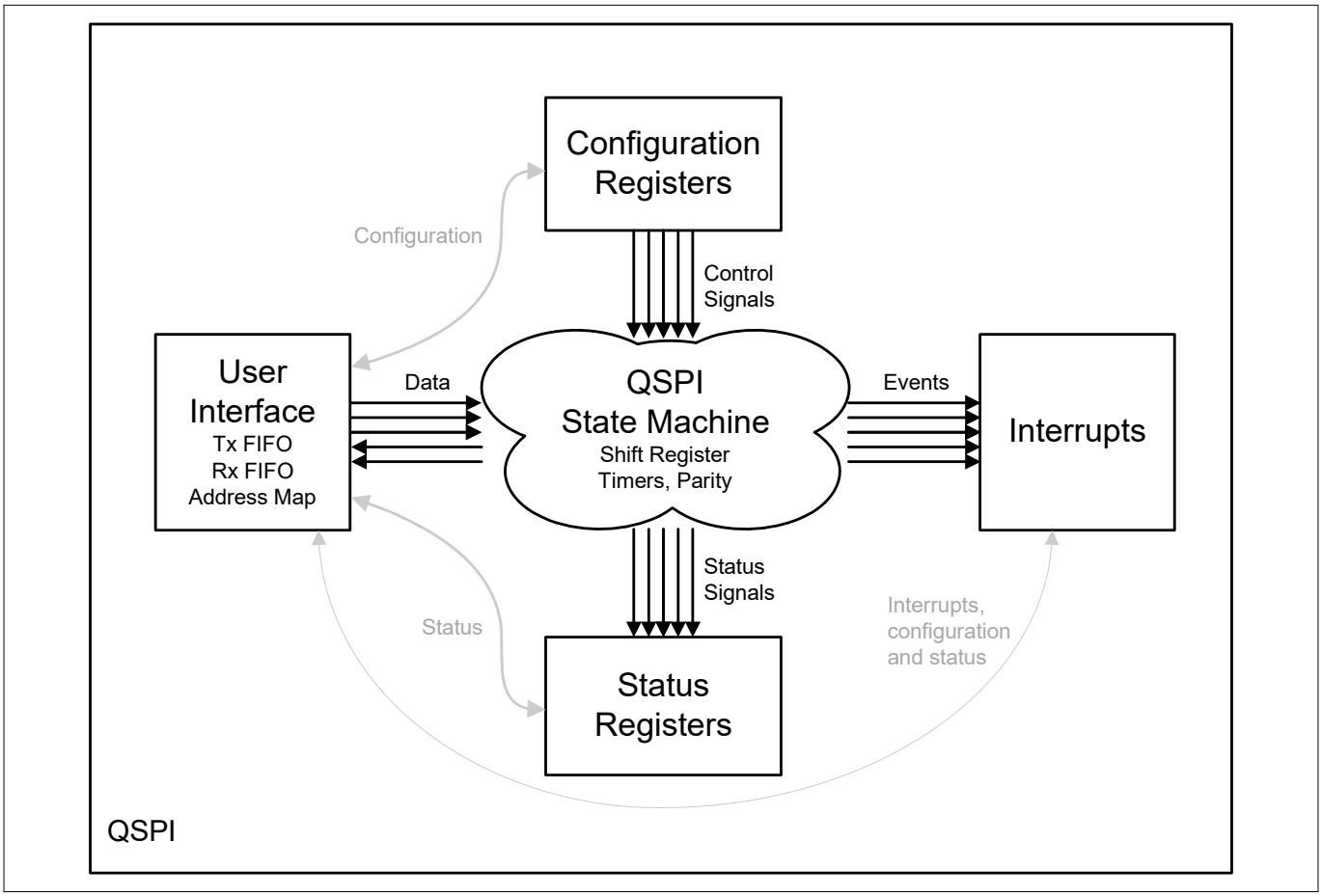


Figure 646 QSPI - abstract overview

30.2.2 Operating modes

The QSPI operates in one of two modes regarding the generation of the serial clock and slave select signals: master or slave mode. The master generates and drives the clock and select signals, the slave receives these signals.

The QSPI operates in one of three modes regarding the direction of the communication and depending whether the transmission and the reception appear simultaneously or not: duplex, half-duplex and simplex mode.

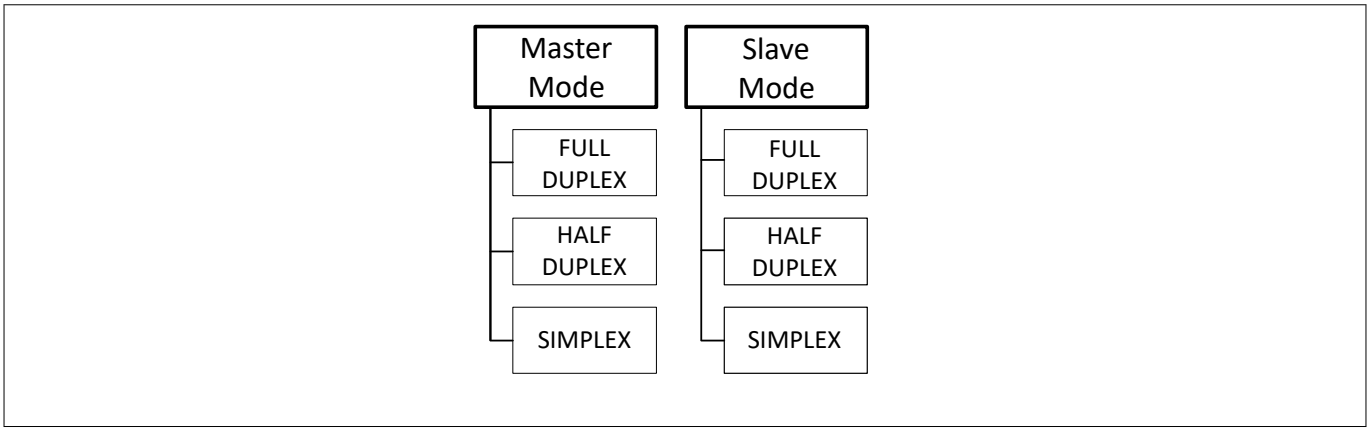


Figure 647 Operating modes overview

This gives six possible combinations for the operating mode of the QSPI: master duplex, half-duplex, and simplex; slave duplex, half-duplex, and simplex.

***Note**: The half-duplex mode can be implemented either by direct connection between two different pins on the pcb, one data output and one data input pin, or by using single pins mapped to both data input and data output signals. See the pinning and port chapters for the pinning definition of a particular product. The module itself does not differentiate between the full-duplex, half-duplex or simplex connection.*

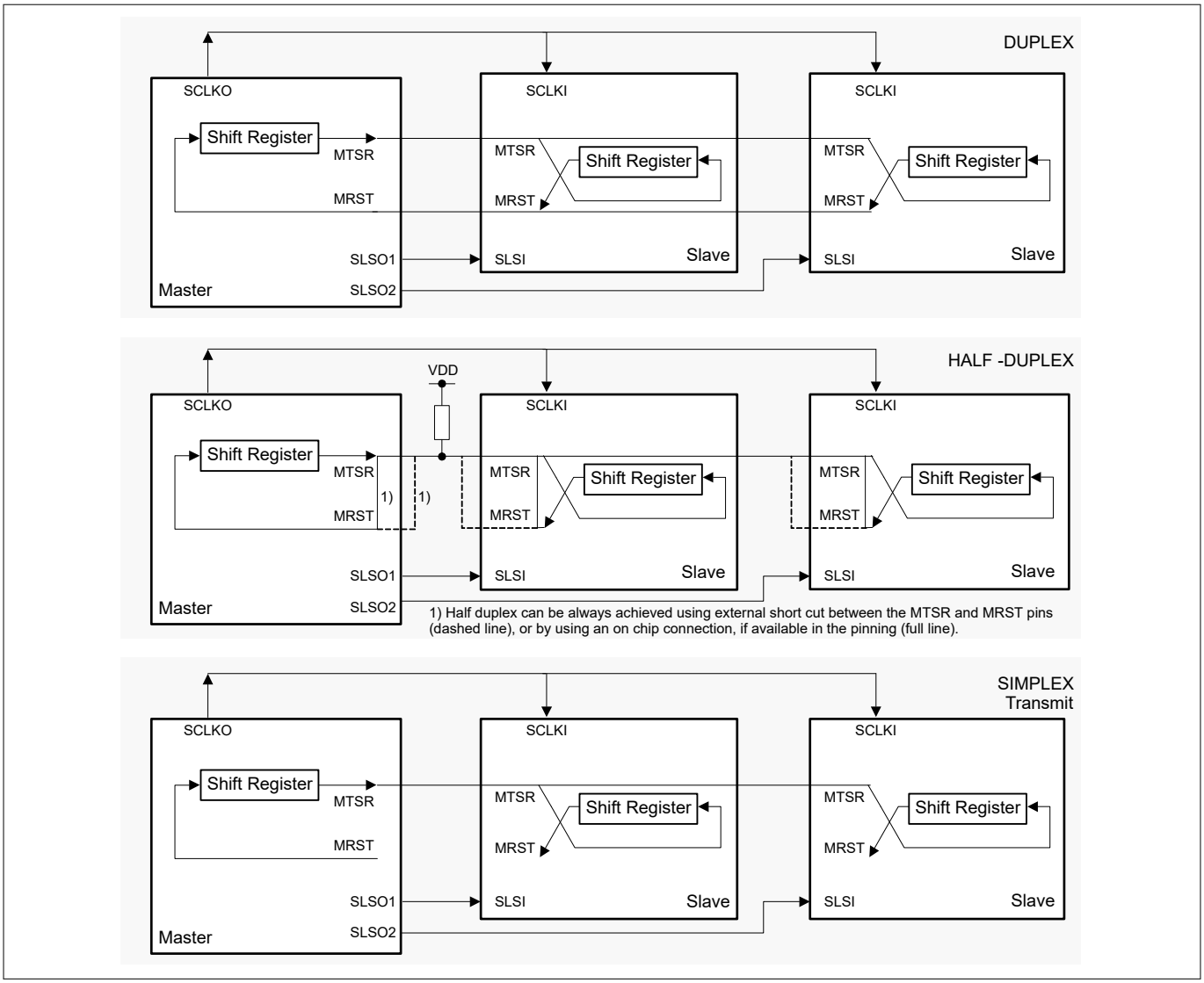


Figure 648 Operating modes and types of connections

30.2.3 Queue support overview

The term “Queue Support” is used in this context to describe the functionality implemented for comfortable switching of the timing configuration of the QSPI frames, depending on the slave select signal which is to be activated. The main feature of the module is the possibility to take both the configuration and data to the TXFIFO, and to track down which TXFIFO entry is configuration, and which data. The QSPI module expects 32 basic configuration bits to be moved with one move (for example DMA move) from some on-chip general purpose RAM to the TXFIFO. These 32 configuration bits from the TXFIFO and the configuration bits from the 8 configuration extension registers ECONz contained in the QSPI module, define together the full configuration of the module. One extension register is used for two slave selects: ECON0 for SLSO0 and SLSO8, ECON1 for SLSO1 and SLSO9...

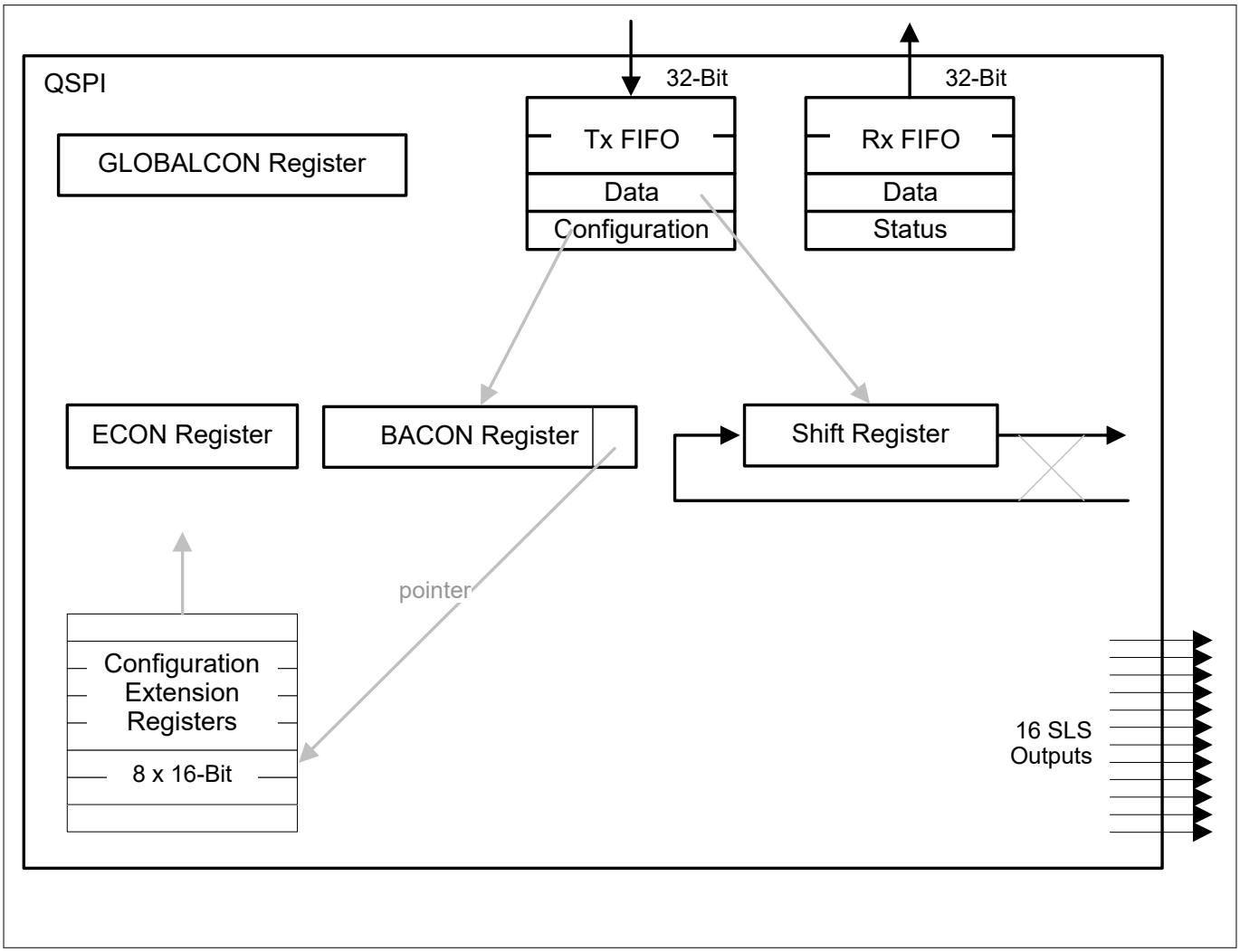


Figure 649 Queue support overview

30.2.4 Architecture overview

The below figure shows the main blocks of the QSPI module. The TxFIFO and RxFIFO provide the user interface. The Shift Register and the “Miscellaneous Logic” block build the state machine of the module. The “Configuration Extensions” block provides comprehensive capabilities for configuring the QSPI frames.

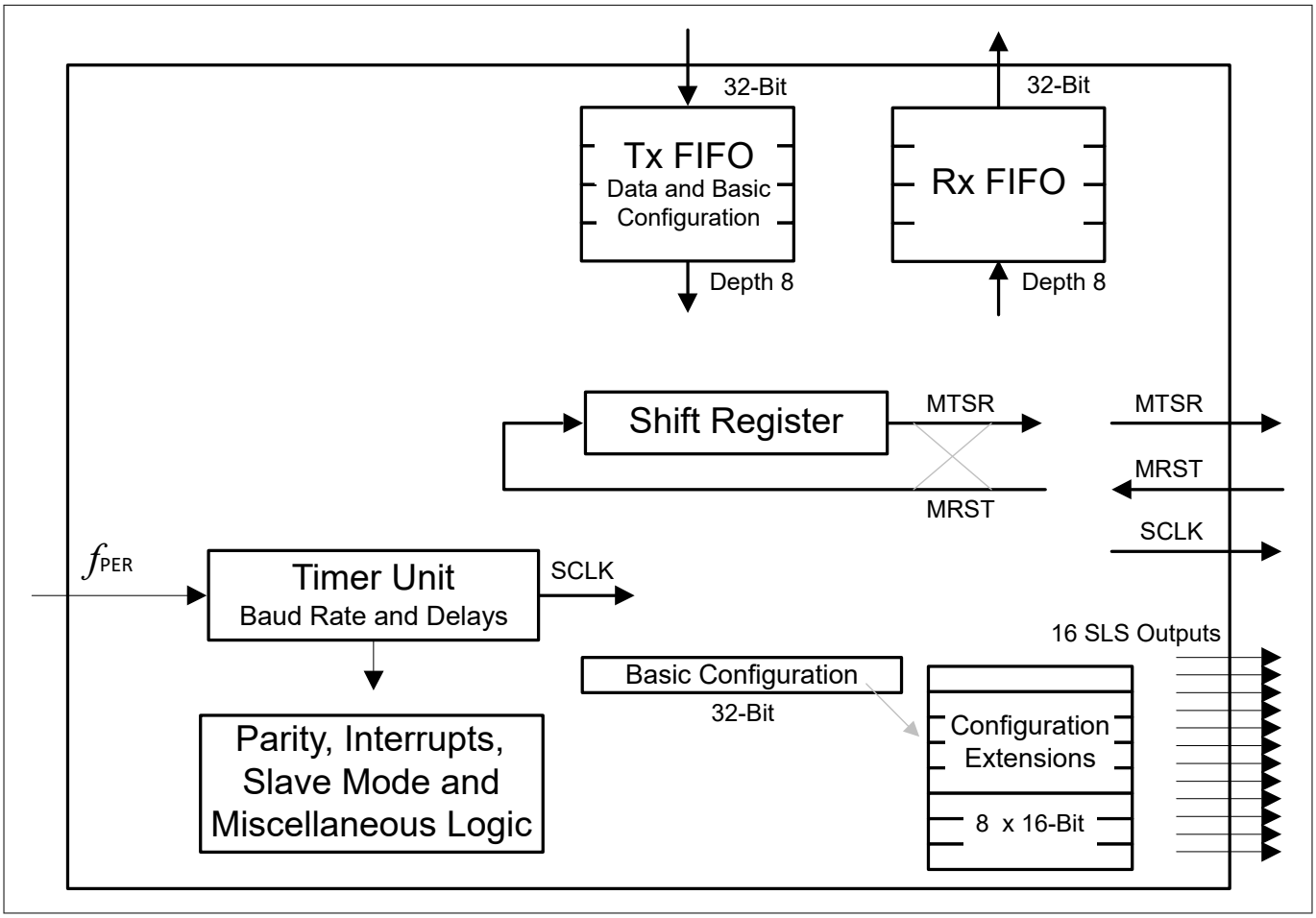


Figure 650 QSPI - architecture overview

30.2.5 Three wire connection

The term “Three Wire Connection” is used in this context to describe a connection without slave select signal. This connection relies solely on counting bits for determining the end of the current frame and resetting the shift register state machine, instead of using the slave select signal for this purpose. This way of communication is less robust than communication with slave select, but it saves a pin on both master and slave device.

The name “three wire connection” is true only in case of full-duplex connection, where exactly three signals (clock, data-in, data-out) are needed. In case of half-duplex and simplex connections only two signals (clock, data-in/data-out common data line) are necessary.

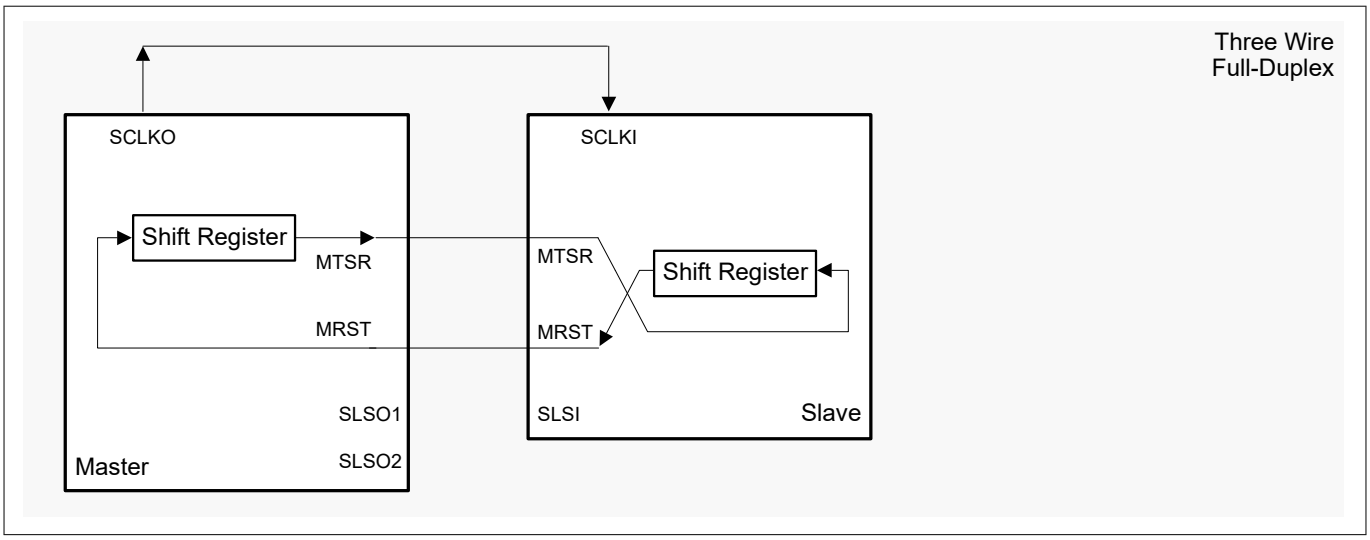


Figure 651 Three wire connection

30.3 Functional description

This chapter provides the description of the module and it's configuration options for functional behavior.

30.3.1 Frequency domains

The state machine of the QSPI is placed in a separate frequency domain operating with a frequency f_{PER} which is generated by the peripheral PLL. This clock is completely asynchronous to f_{SPB}.

In Chapter CLOCK this f_{PER} is referenced to as f_{QSPI} for overview purposes.

The baud rate is determined by the f_{PER} frequency divided by an integer equal to or greater than four.

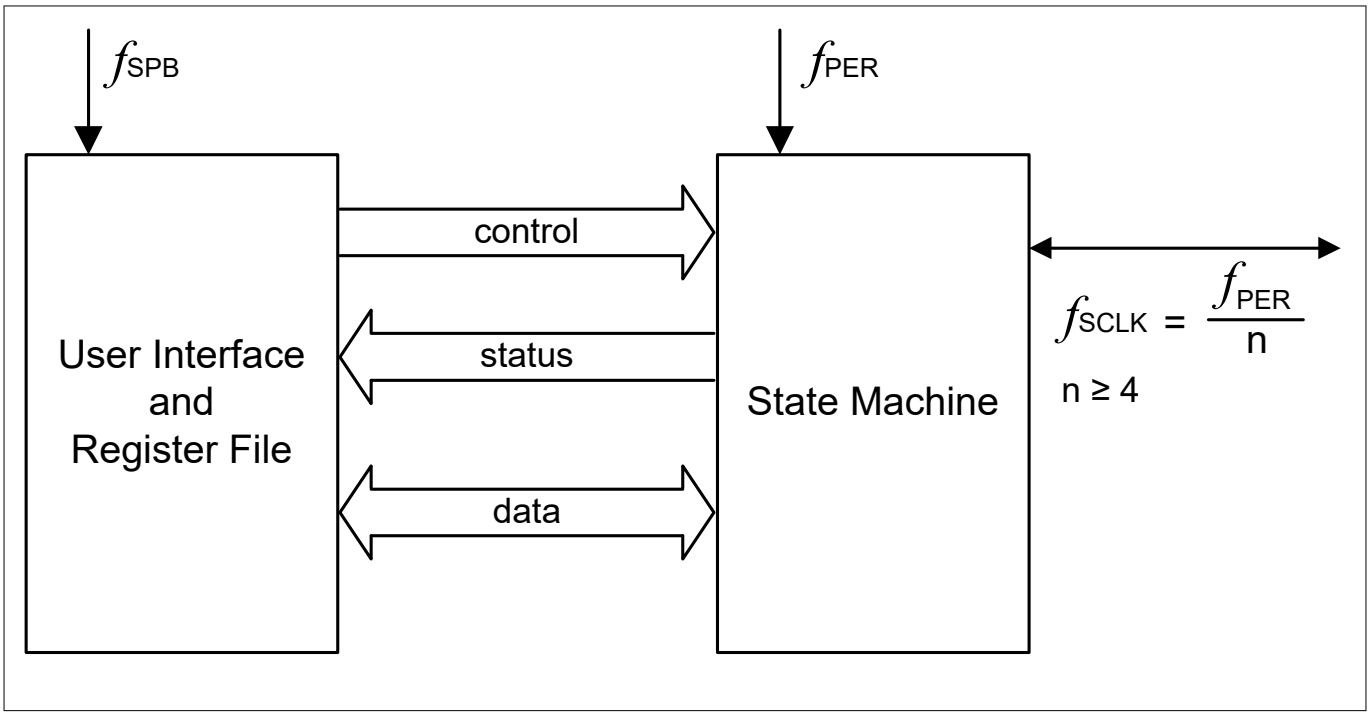


Figure 652 QSPI - frequency domains

30.3.2 Master mode

In master mode, the QSPI module generates the timings, the serial clock, and the slave select signals.

30.3.2.1 Phases of one communication cycle

This section describes the possibilities available for configuring the length of the phases of the QSPI communication: timing delays, data length, duty cycle and data sampling.

A QSPI frame starts with activating a slave select signal SLSO (at transition from idle to leading delay phase), and ends with its deactivation (at transition from trailing delay to wait or idle phase). It is a sequence of five phases: idle delay, leading delay, data, trailing delay and an optional wait. The idle phase is subdivided in two phases of equal length: idle A and idle B.

The following figure shows a full and a compressed view of a QSPI frame and its phases. The full view shows all four signals needed for a QSPI connection. The compressed view represents the phases in a single row, in a way suitable to discuss their properties.

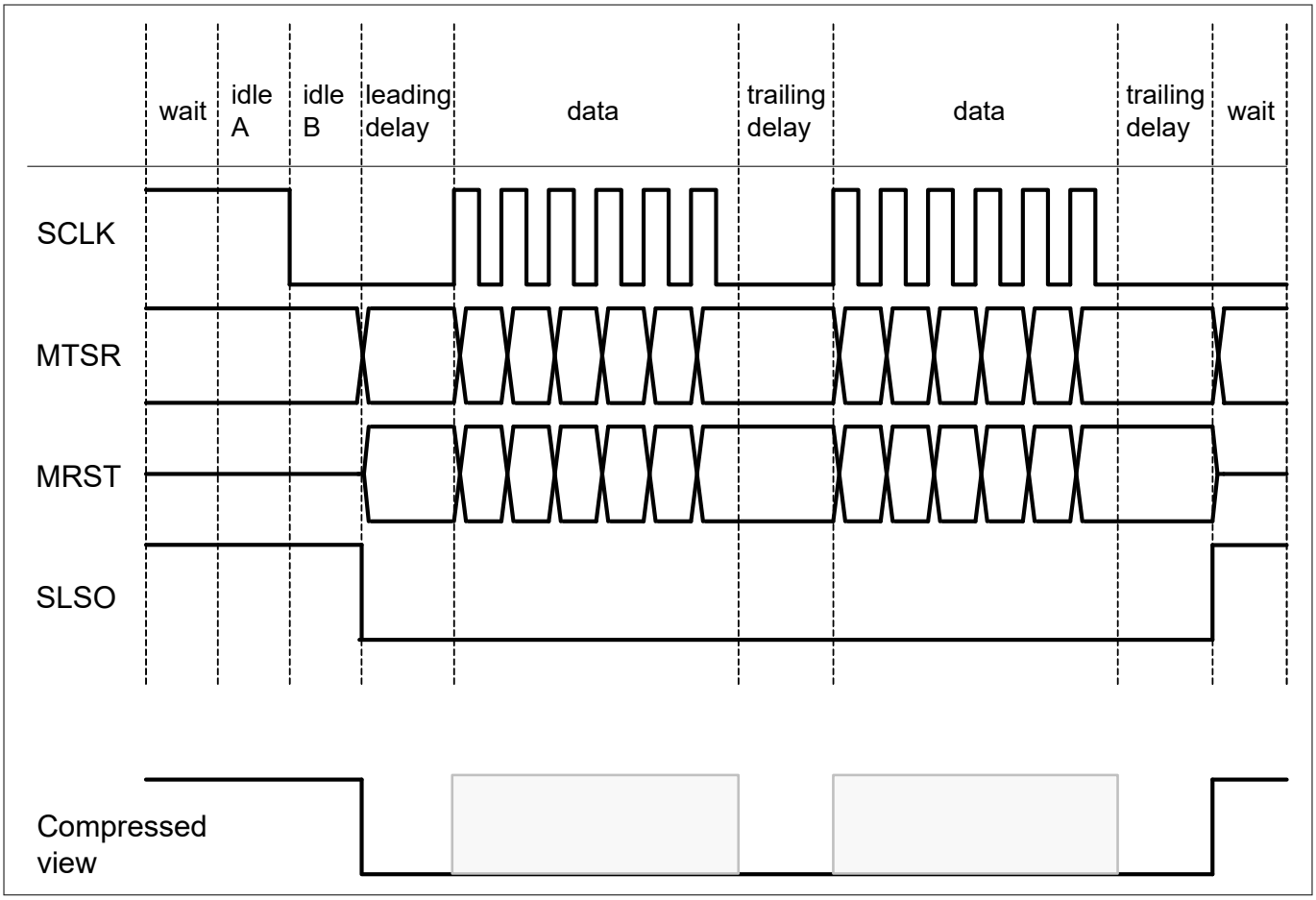


Figure 653 Phases of a QSPI frame (example for data length of 5 bits)

The flexible timing control of the QSPI allows to program each phase in a very wide time range, with sufficient precision. Some examples of QSPI frames, in compressed view, with different lengths of the phases, are shown in the following figure.

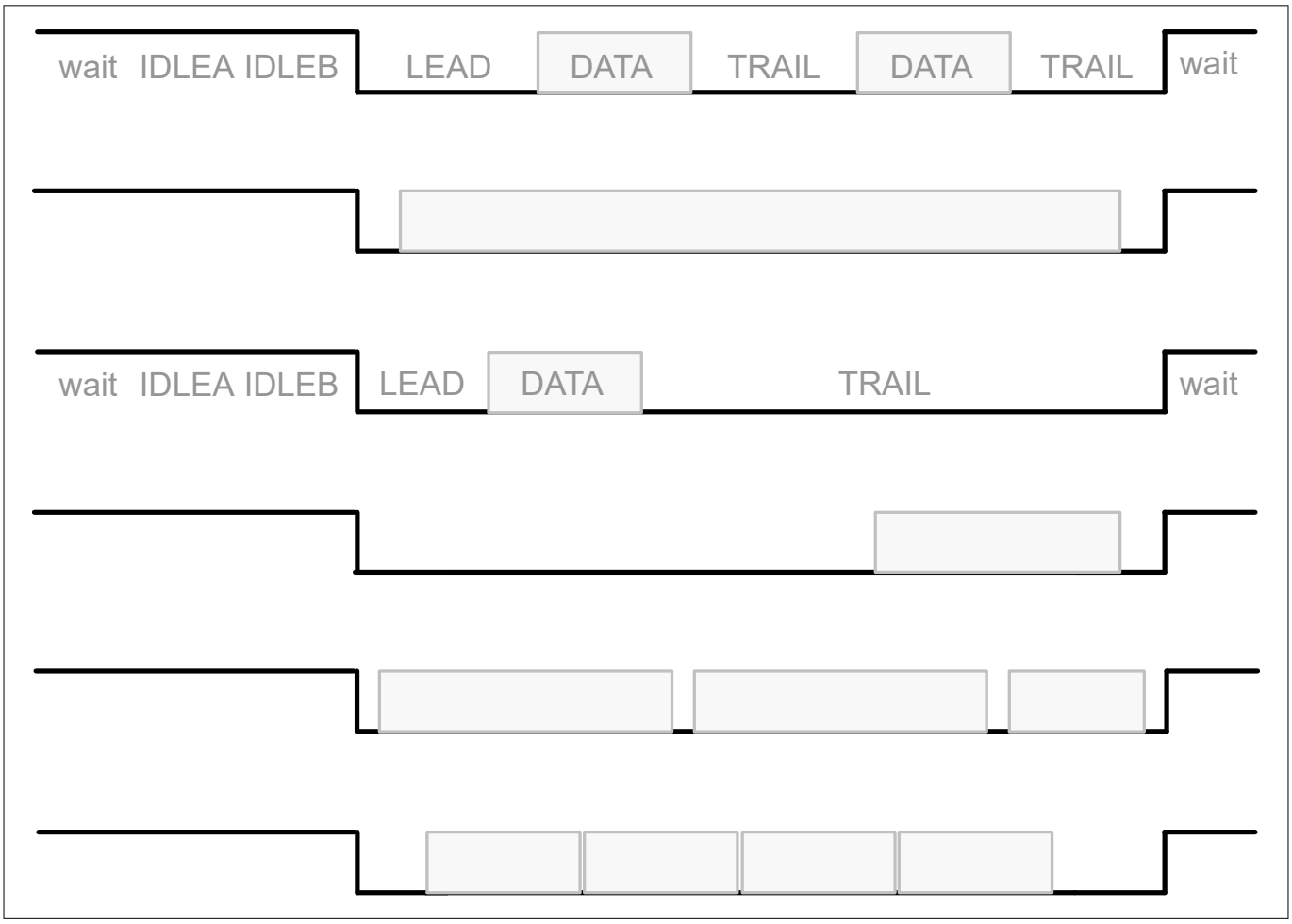


Figure 654 Examples of QSPI frames in compressed view

The length of the phases is programmed using the corresponding bit-fields in the register BACON. There are four main bit-fields defining the length of the phases in t_Q units and several modifier bit-fields defining their range and granularity:

- IDLE
- LEAD
- DATA
- TRAIL

The WAIT phase is simply a loop waiting for a write to the Tx FIFO, without predefined duration, and consequently without a defining bit-field.

The IDLE bit-field defines two sub-phases with the same length IDLEA and IDLEB. At the transition between them the polarity of the serial clock signal for the next slave is set.

In addition, the flexible timing control allows to program the duty cycle and the sampling point properties of the serial clock. This allows to improve to a certain extent the achievable baud rate taking into account the clock asymmetries and the loop-delay.

The ECONz (z=0-7) bit-fields A, B and C define the length of the corresponding bit segments. The sampling point is always at the transition between phases B and C, and the clock duty cycle is defined by the phases A and B+C.

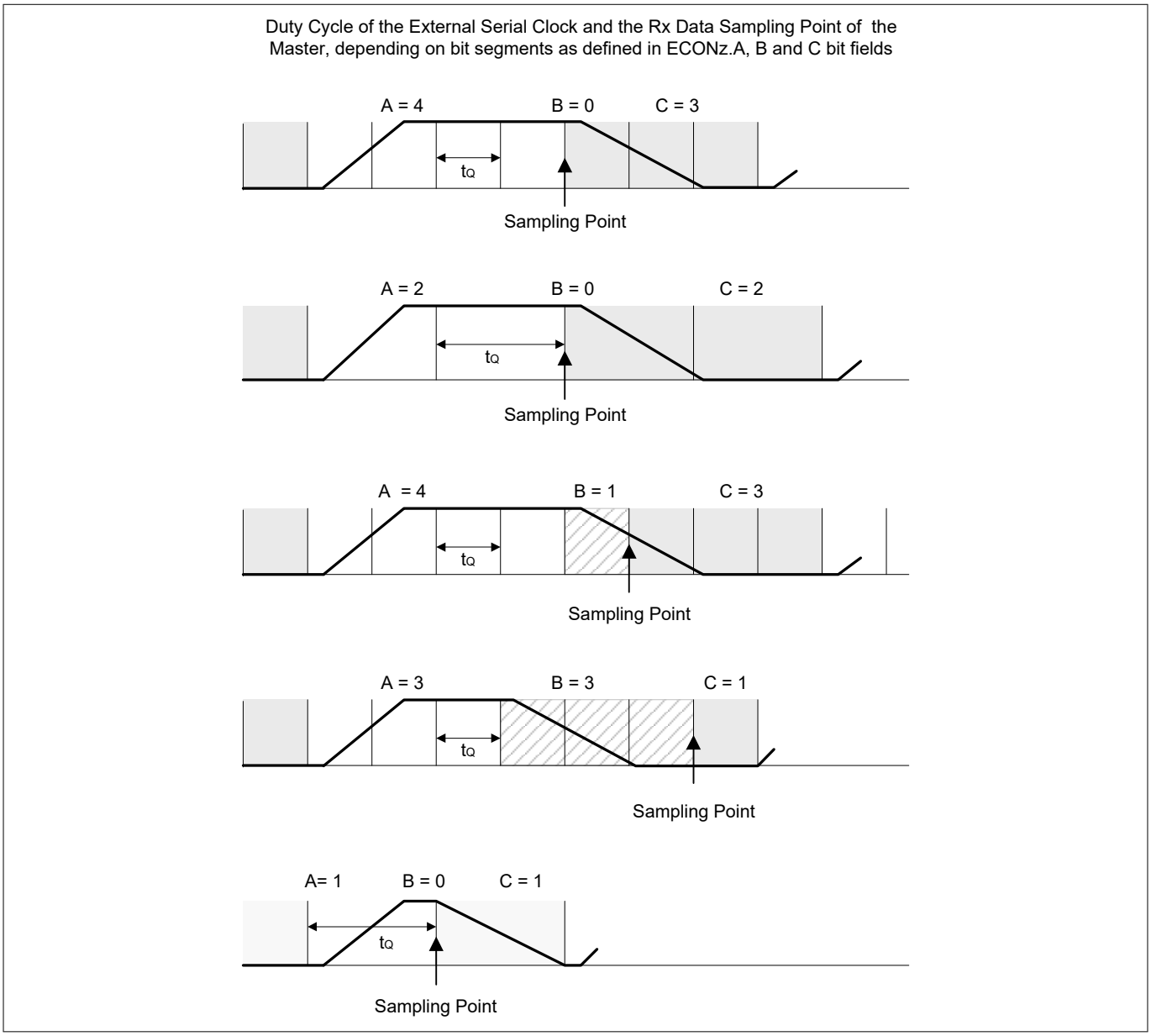


Figure 655 SCLKO duty cycle and the Rx data sampling point relative to SCLKO in master mode

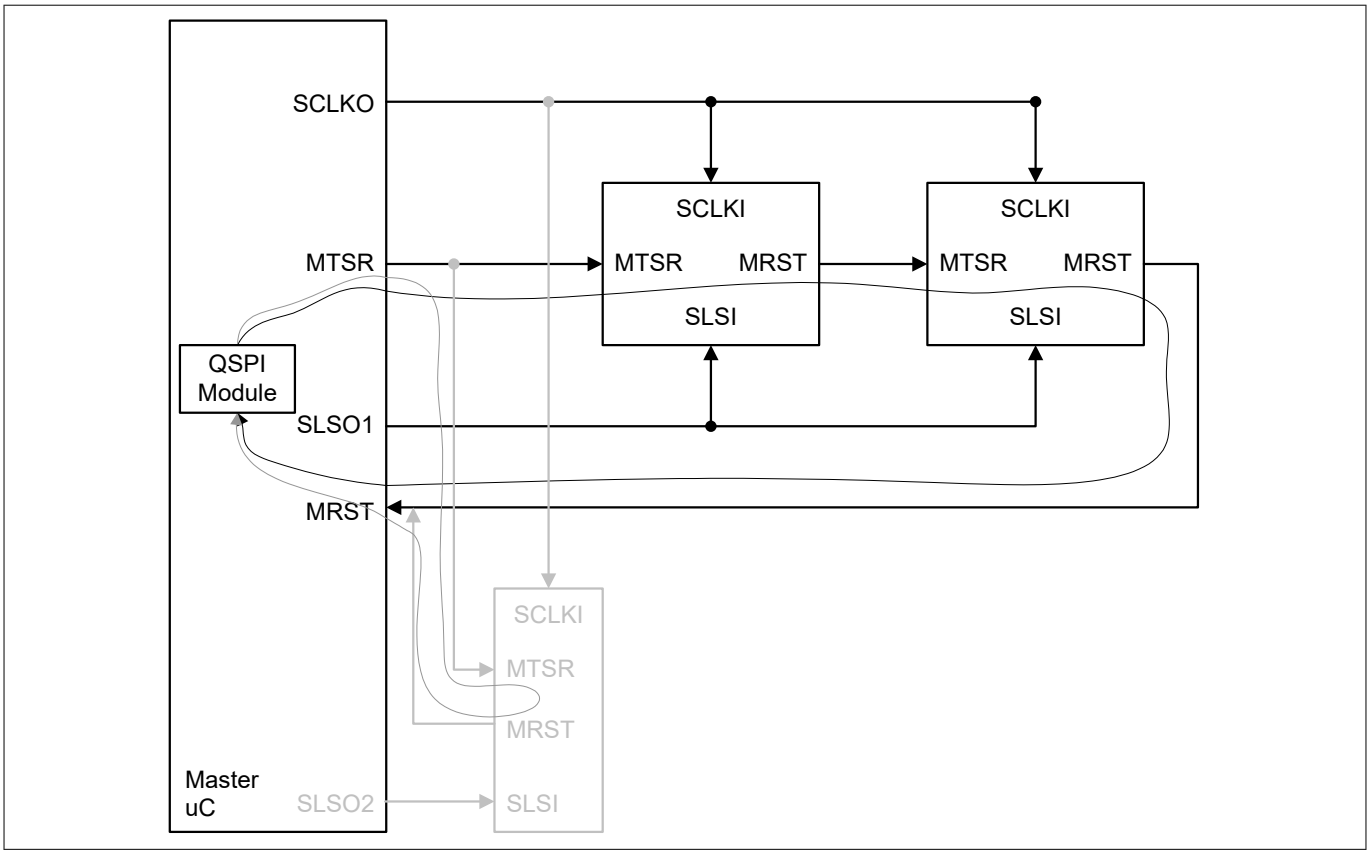


Figure 656 Closed-loop delays in a simple network consisting of one daisy-chain and one point to point connection in parallel

The purpose of the duty cycle control is:

- Compensating the influence of the path and pad asymmetry on the duty cycle
- Compensating the considerable asymmetry of the edges in case of half-duplex connection implemented as open-drain driver with pull-up resistor
- Adjusting the duty cycle optimally to the timing properties of the master and slave and to the properties of the connection (distance and capacitive load)
- Helping to achieve optimal baud rate, especially at higher baud rates which need an odd division factor (for example $5 = 3$ high time $+ 2$ low time or $5 = 2$ high time $+ 3$ low time)

The purpose of the sampling point control is to allow reliable reception of the data transmitted by the slave, at high baud rates. A further improvement on achievable baud rates is possible using High Speed Mode, see related information.

In master receive direction the slave transmitted data reaches the master considerably late relative to the shift edge of the serial clock. This is caused by the loop delay consisting of:

- The propagation delay of the shift clock edge from the master to the slave
- The response delay of the slave
- The propagation delay of the data from the slave to the master

Last but not least, the flexible timing control allows to configure the phase and the polarity of the shift clock.

The clock polarity is configured by bit-field ECONz (z=0-7).CPOL. This bit-field sets the clock idle polarity to low or high.

The clock phase is configured by bit-field ECONz (z=0-7).CPH. This bit-field sets the initial clock delay to 0 or Bit Segment A, as defined by the bit-field ECONz (z=0-7).A.

The CPOL and CPH settings control the master mode. The slave mode timing behavior is fixed to CPOL=0 and CPH=1, and these values must be programmed in the ECONz (z=0-7) register pointed by BACON.CS. The ECONz (z=0-7).A, B and C do not influence the sampling point in the slave mode, the sampling is at the falling edge of the SCLKI.

The following figure shows the QSPI timings for clock phase CPH = 0.

ECONz (z=0-7).CPH = 0 is used when communicating with a slave that delivers a valid value of the first bit immediately after being selected with an SLS signal. In such a case the first clock edge in a frame is used to latch the first bit. The second edge delivers the second bit value, and so on. The last edge in a frame delivers a bit with a to be ignored value.

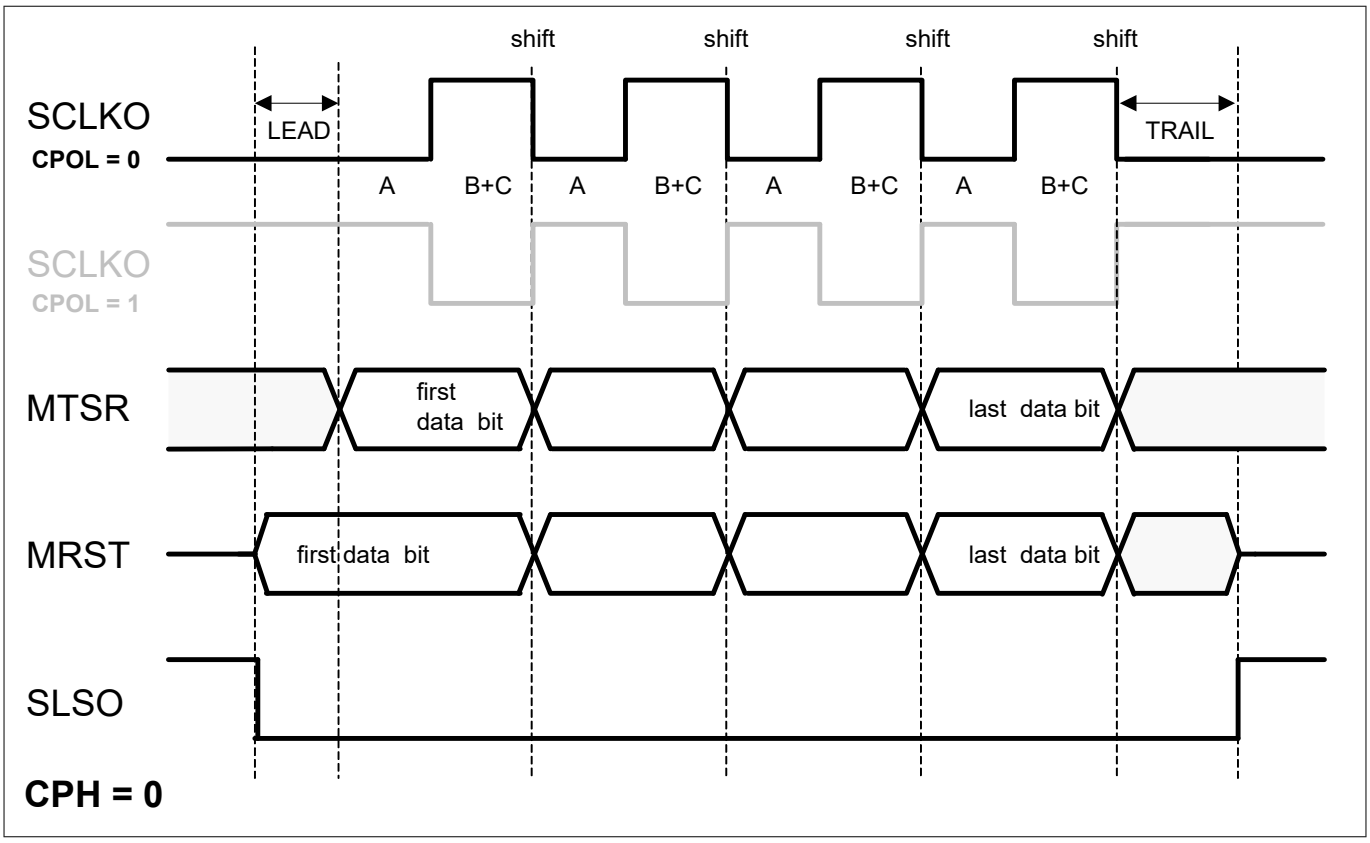


Figure 657 Timing of a transfer with CPH = 0 (example for data length of 4 bits)

The trailing delay starts after the last shift clock period of a data block and is followed either by the deactivating edge of SLSO, or a new data block in continuous mode.

***Attention**: If CPH = 0, then the total delay between the SLSO activating edge and the first edge of the serial clock SCLKO is LEAD + ECONz (z=0-7).A*

The following figure shows the QSPI timings for clock phase CPH = 1.

ECONz (z=0-7).CPH = 1 is used when communicating with a slave that delivers a bit with a random or to be ignored value immediately after being selected with the SLSO signal. In such a case the first clock edge in a frame is used by the slave to shift out the first valid data bit. The second edge latches this bit value, and so on. The last edge in a frame is used to latch the last valid data bit, which normally remains driven until the end of the frame.

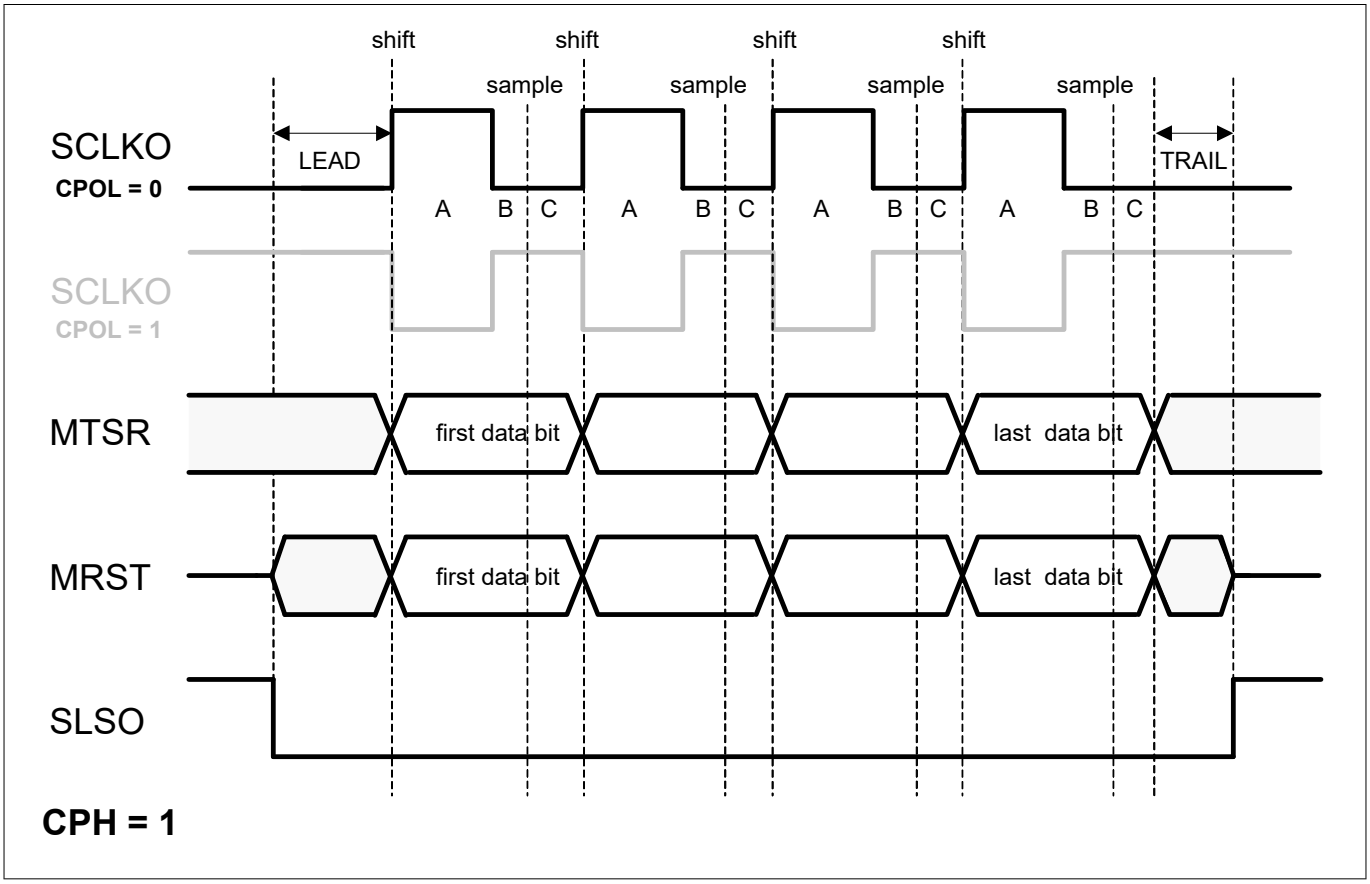


Figure 658 Timing of a transfer with CPH = 1 (example for data length of 4 bits)

The leading delay lasts between the active edge of the SLSO and the first shift clock edge.

***Attention**: If CPH = 1, then the total delay between the last edge of the SCLKO and the SLSO deactivating edge is TRAIL + ECONz (z=0-7).B + ECONz (z=0-7).C*

***Attention**: When 3 wire mode is used, and the polarity of CPH or CPOL is changing, a dummy frame is required. The pad enable to be after the dummy frame. The values of the clock and data signals are not defined before the first data is transmitted.*

Related information

High speed mode on page 5320

30.3.2.2 Configuration extensions

This section describes the possibilities available for configuring the phases of the QSPI communication: timing delays, data lengths, their ranges and granularity.

The QSPI module controls 16 communication channels, which are individually programmable. Each channel is associated to one slave select output and to the corresponding timing and other properties (baud rate, delays, data width, parity...).

The 16 slave select channels of the QSPI module are subdivided in two groups of 8 slave select channels:

- Channels 0 to 7
- Channels 8 to 15, channel 8 having the same ECONz (z=0-7) settings as channel 0, channel 9 as channel 1, and so on

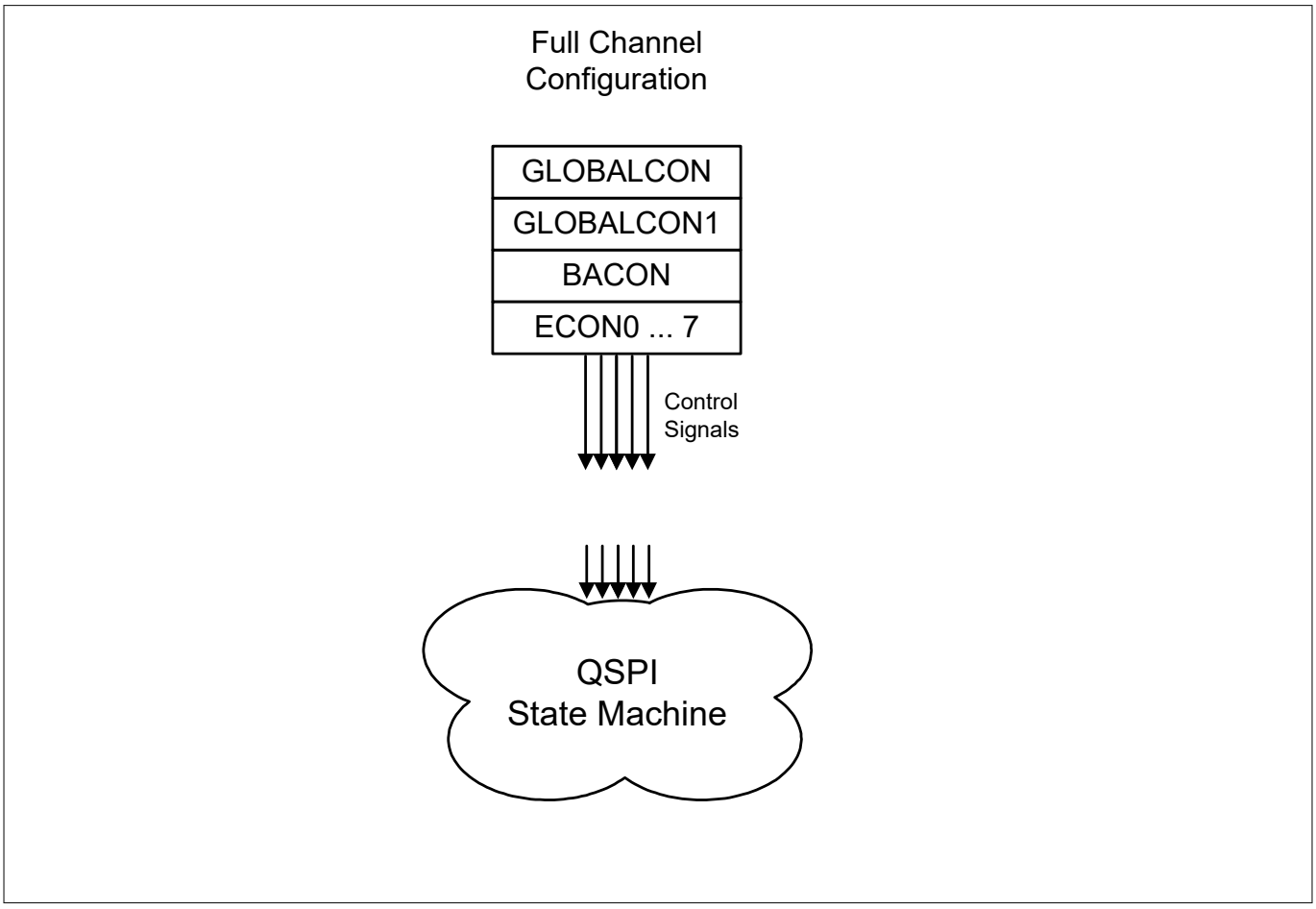


Figure 659 Flexible configuration concept

30.3.2.3 Details of the baud rate and phase duration control

The QSPI phases correspond to states of a simple state machine defining a sequence of simple states and loops. Due to the strict sequential nature of the communication cycle one timer is enough to define the length of each phase. The length of each phase is defined in time quanta, and the length of a time quantum is defined by a time quantum timer. Each QSPI slave select channel has its own time quantum length, based on the module time quantum length: T_{PER} clock period -> one module time quantum -> one channel time quantum

Each QSPI module generates a module time quantum which is further downscale to a channel quantum. This affects both the delays and the baud rates. The assumption is that the pad strength setting, the capacitive load and the corresponding QSPI network layout properties do not allow extreme differences in the baud rates for the different slaves connected to one module. They all have similar speed range: high speed, or medium speed, or low speed. Mixing very high speed and very low speed slaves does not make much sense. Many slaves produce high accumulated capacitive load on the clock and data outputs, which influences the possible baud rates for all slaves. Variations in the baud rates of slaves of one module are supported by the ECONz.Q and the ECONz.A/B/C bitfield settings allowing for a flexible bit time variation between the channels in one module.

Each QSPI slave select channel has its own set of phase lengths, depending on $T_{PER} = \frac{1}{f_{PER}}$.

- $T_{SCLKz} = T_{PER} \times$ GLOBALCON.TQ $\times$ ECONz (z=0-7).Q $\times$ (A + B + C)
- $t_{LEAD} = T_{PER} \times$ BACON.LPRE $\times$ BACON.LEAD
- $t_{TRAIL} = T_{PER} \times$ BACON.TPRE $\times$ BACON.TRAIL
- $t_{IDLEA,B} = T_{PER} \times$ BACON.IPRE $\times$ BACON.IDLE
- $t_{STROBE_DELAY} = T_{PER} \times$ GLOBALCON.TQ $\times$ ECONz (z=0-7).Q $\times$ GLOBALCON.STROBE

***Note**: The equations above contain the representative decimal values of the bit-fields in corresponding units, not their actual binary bit-value. For the formulas using the actual binary values of the bit-fields, see figures in related information.*

The subsequent figure summarizes the above formulas, especially the upper part of the picture shows the fact how T_{PER} is transferred to T_Q and subsequently to T_{Qx}. The lower part shows how different parts of the BACON registers, such as IDLE, LEAD, A, B, C, TRAIL are connected to different source clocks to produce the different timings of a SPI communication.

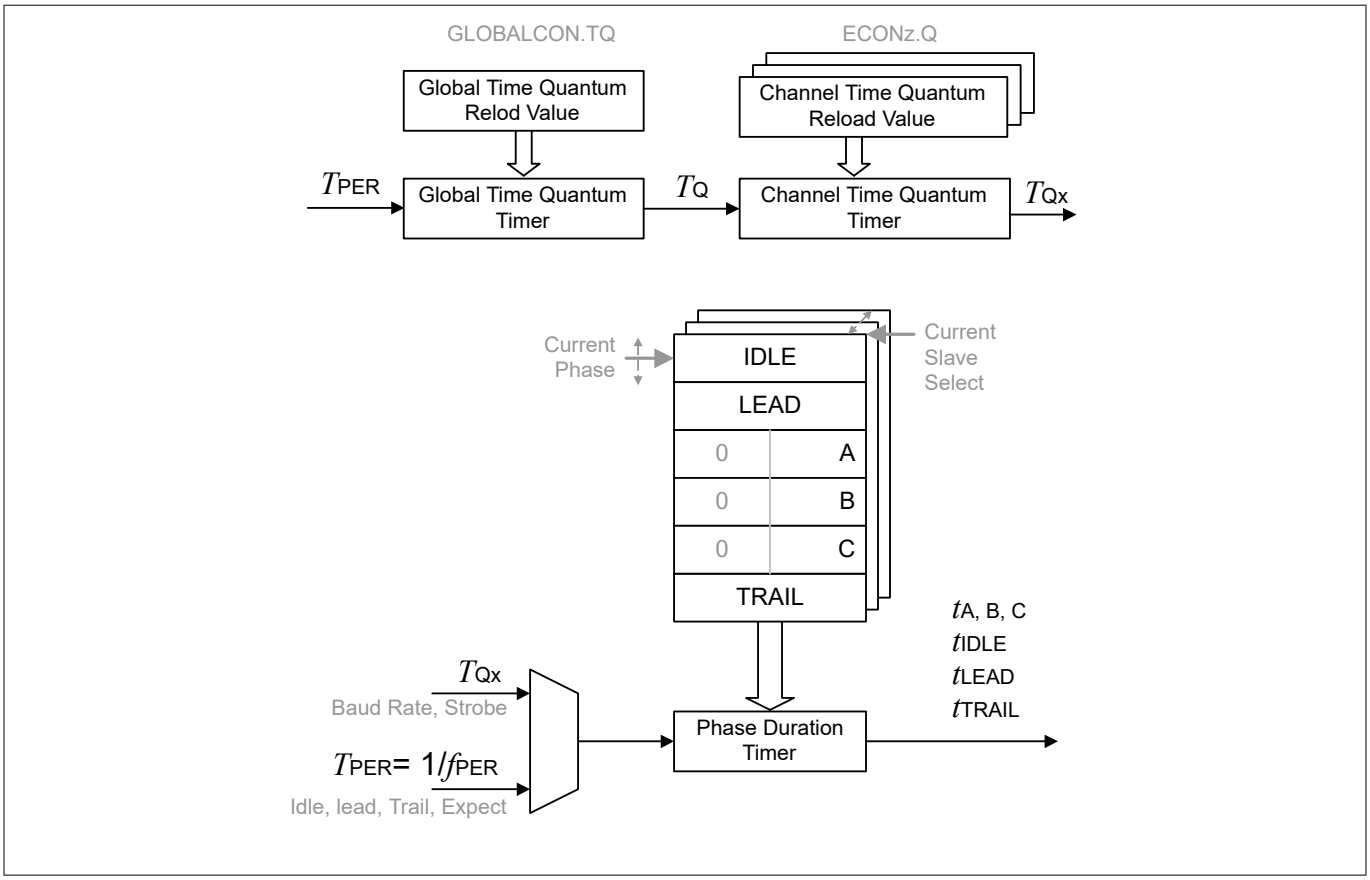


Figure 660 Overview of phase duration control

Related information

Calculation of the baud rates and the delays on page 5313

30.3.2.4 Calculation of the baud rates and the delays

This section gives another alternative view to the calculation of the baud rate and the duration of the delays.

The baud rate generation chain for a channel starts with a common TQ divider. The resulting global time quantum is used by the dedicated Q and A,B,C dividers to generate the baud rates per channel (slave select). See following figure.

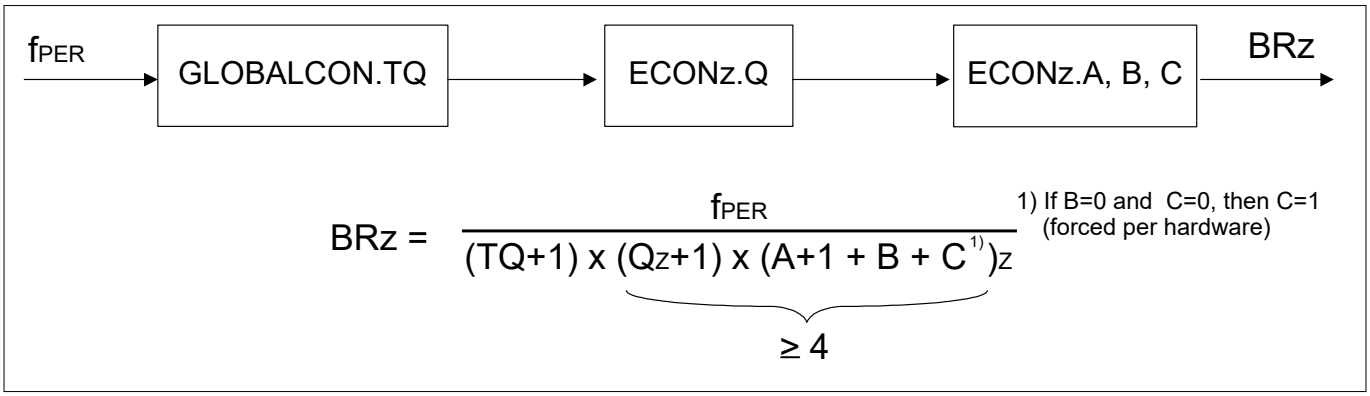


Figure 661 Baud rate calculation

Attention: ***The following condition must be satisfied:*** $(Q+1)\times(A+B+C+1)\geq 4$***. Or alternatively formulated: bit length must be at least*** $4\times T_Q$***.***

Each delay of an active channel (slave select) has a separate and independent divider chain starting with a power-of-four prescaler and an n-divider. See following figure.

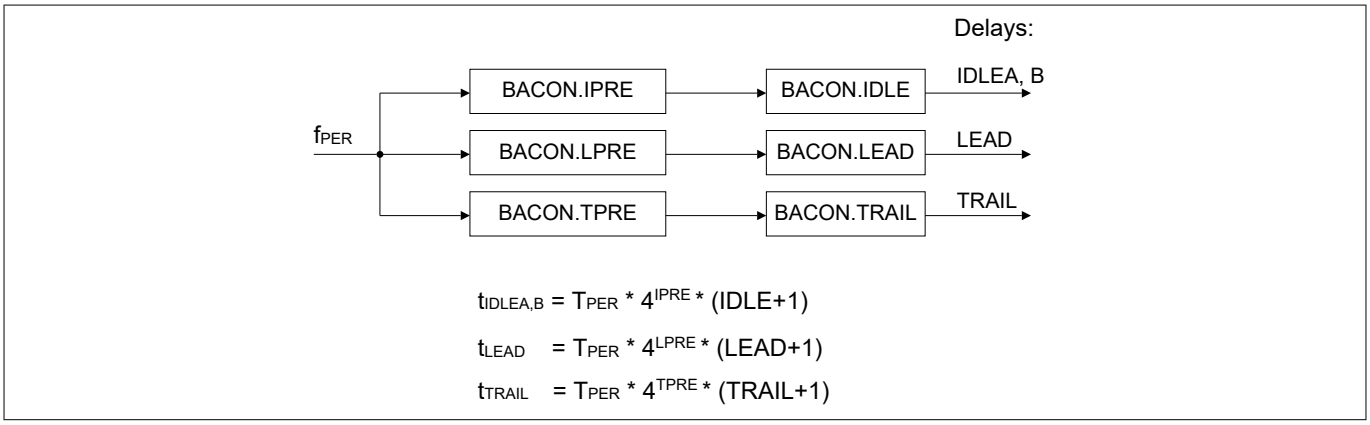


Figure 662 Calculation of the delays

30.3.2.5 State diagram of standard communication cycle

Figure "Standard communication cycle of the state machine (short data)" shows the top view of the standard communication cycle of the QSPI state machine in a hierarchical way. The “Data” state consists of a loop of several “Bit” states, where the number of repetitions is defined with the data width bit-field. Each “Bit” state consists of a simple sequence of “A”, “B”, and “C” states.

A complete (or standard) communication cycle containing all phases, starting with activating the slave select and ending with its deactivation, is executed only if the bit BACON.LAST = 1. Otherwise, the slave select remains active and the module waits for more FIFO data entries.

In Master Mode, the Phase Transition Interrupt events, PT1 and PT2 , each of them can signal one of the phase transitions which are shown in Figure 663 . The events are flagged through STATUS.PT1F and STATUS.PT2F. The values of GLOBALCON1.PT1 and GLOBALCON1.PT2 determine which of the transitions cause the events. The events will also cause interrupts if GLOBALCON1.PT1EN and GLOBALCON1.PT2EN are set to 1. The event flags are cleared through the programming of the FLAGSCLEAR register.

For the Move Counter Mode, see related information.

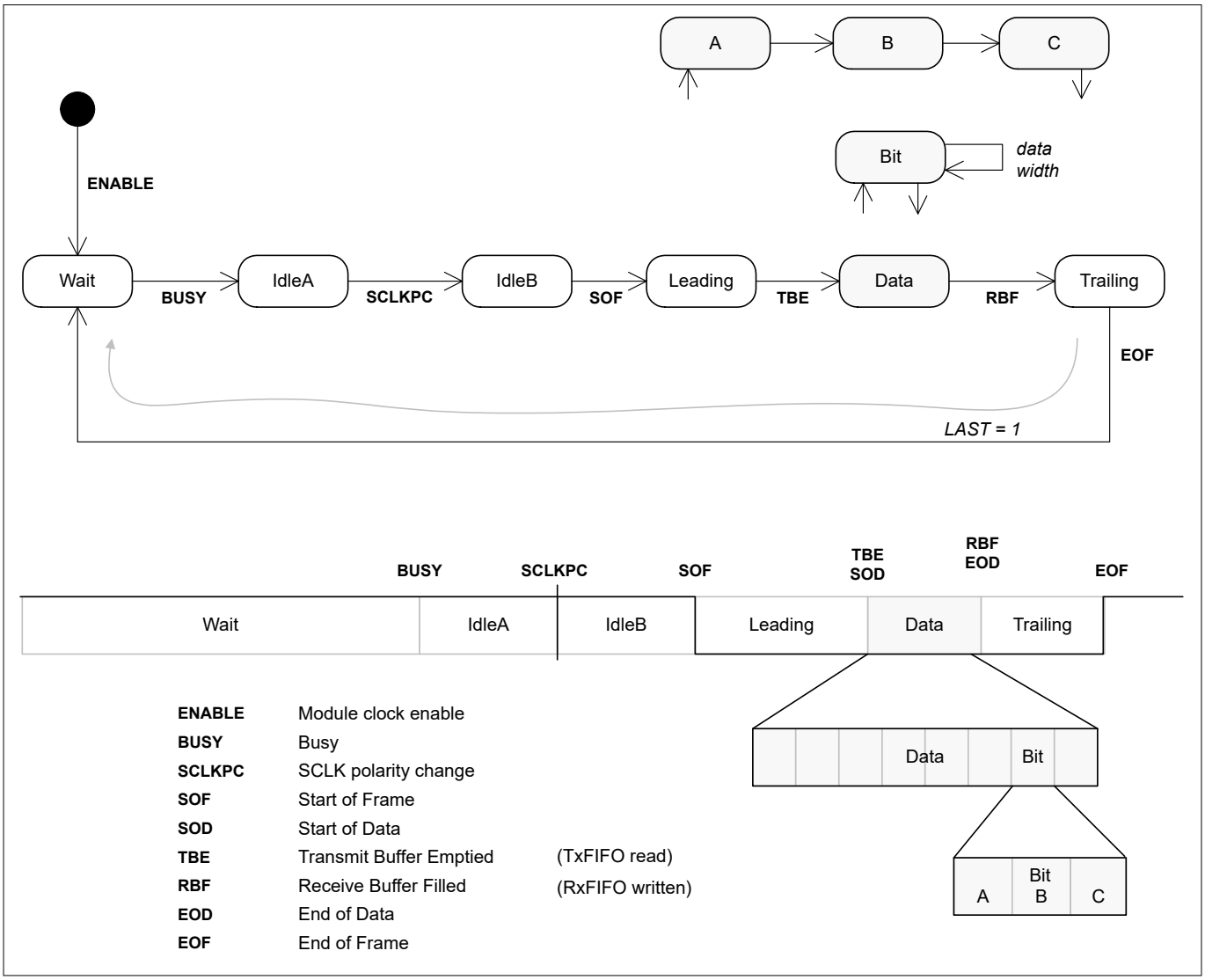


Figure 663 Standard communication cycle of the state machine (short data)

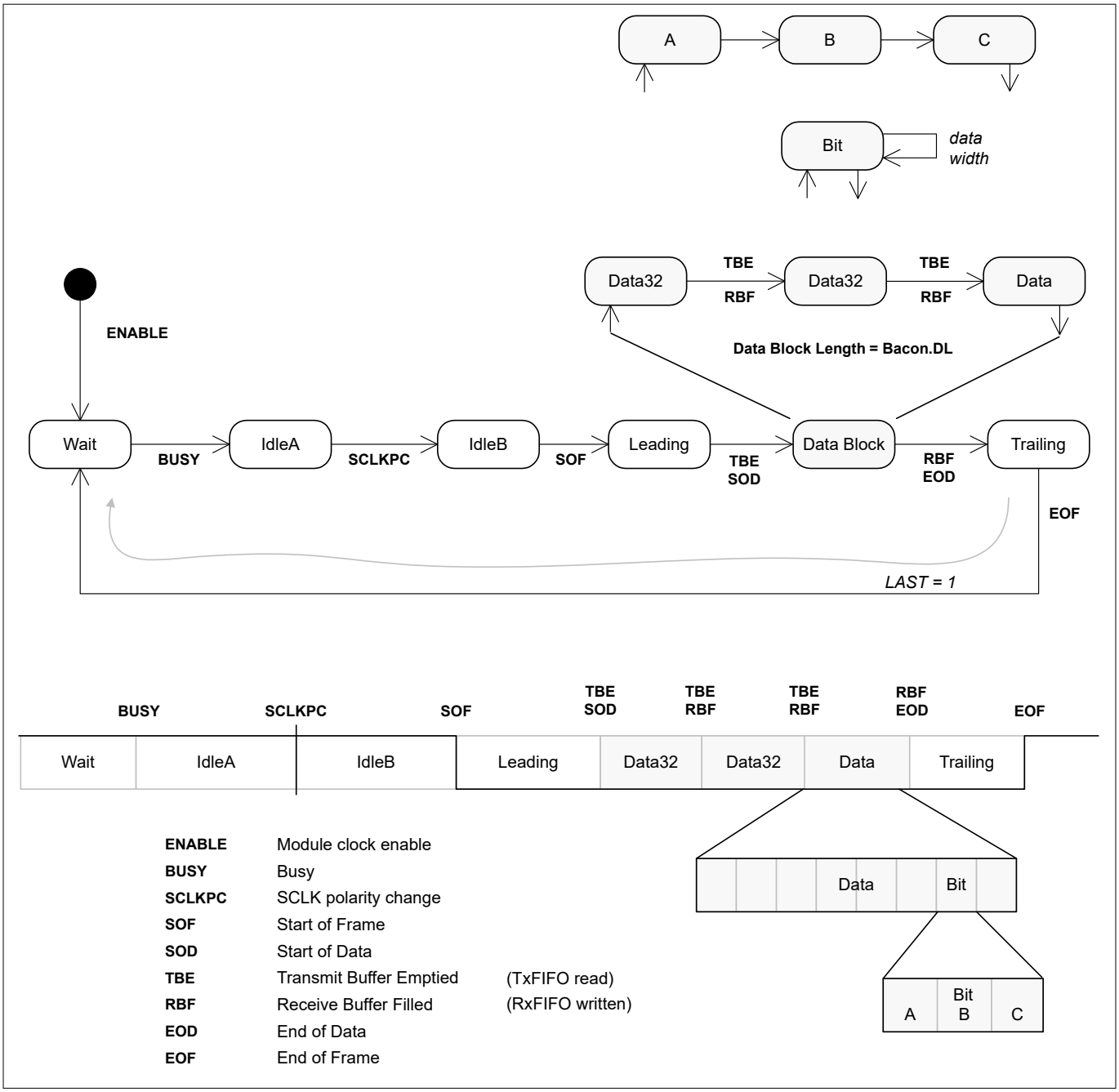


Figure 664 Standard communication cycle of the state machine (long data)

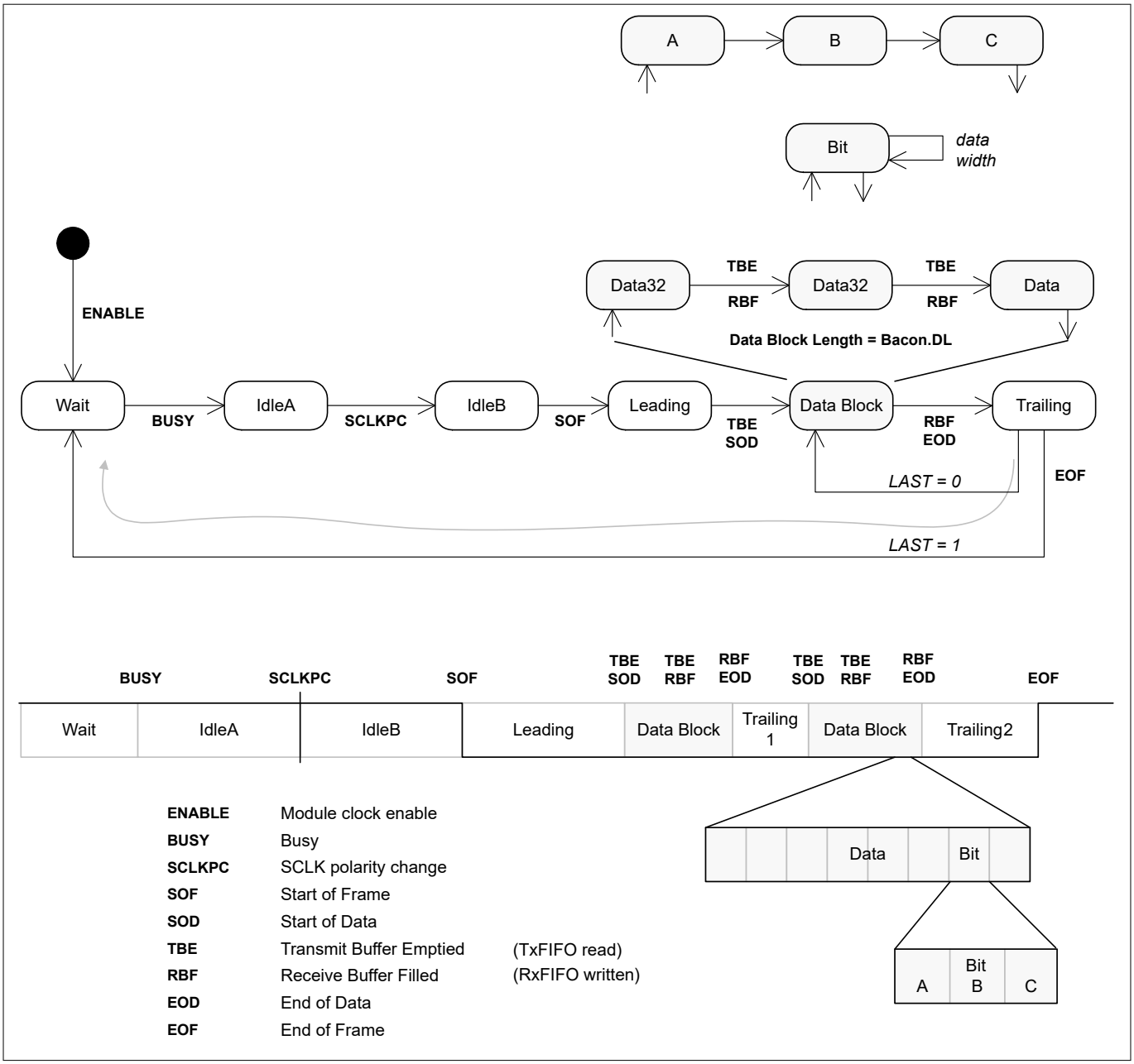


Figure 665 Standard communication cycle of the state machine (continuous)

"Data Block" in the figure above refers to either short data or long data block. When concatenating data blocks in continuous mode, each block can be configured independently as short or long, according to its BACON configuration.

Related information

Move counter mode on page 5342

30.3.2.6 Expect phase

The following figure shows the EXPECT state in the QSPI state machine diagram. This state can be entered in the long data mode and in the continuous mode, where one frame consists of more than one TXFIFO entries. Its purpose is to provide monitoring of the latencies on the FPI bus. In this case, after one FIFO entry has been shifted out and the module expects more bits to shift, according to the BACON.DL, a time-out counting starts. If the expected FIFO entry does not come in time, a time-out occurs.

The duration of the EXPECT state is zero if at the end of the current DATA state the new data is already available in the FIFO.

The maximum allowed duration for EXPECT phase before time-out occurs can be programmed in GLOBALCON.EXPECT. The duration of the EXPECT state can be anywhere between zero and the programmed upper limit, if the next data is written into the TXFIFO any time within the allowed time window. In this case the current frame continues with the SLSO signal remaining active.

The duration of the EXPECT state is exactly maximum if the data did not come in time. Then the current frame is stopped, but the SLSO is not deactivated. A TO (Time - Out) interrupt is raised. The state machine continues the EXPECT state in a loop, waiting for the software intervention, generating an interrupt signal after each expect time interval.

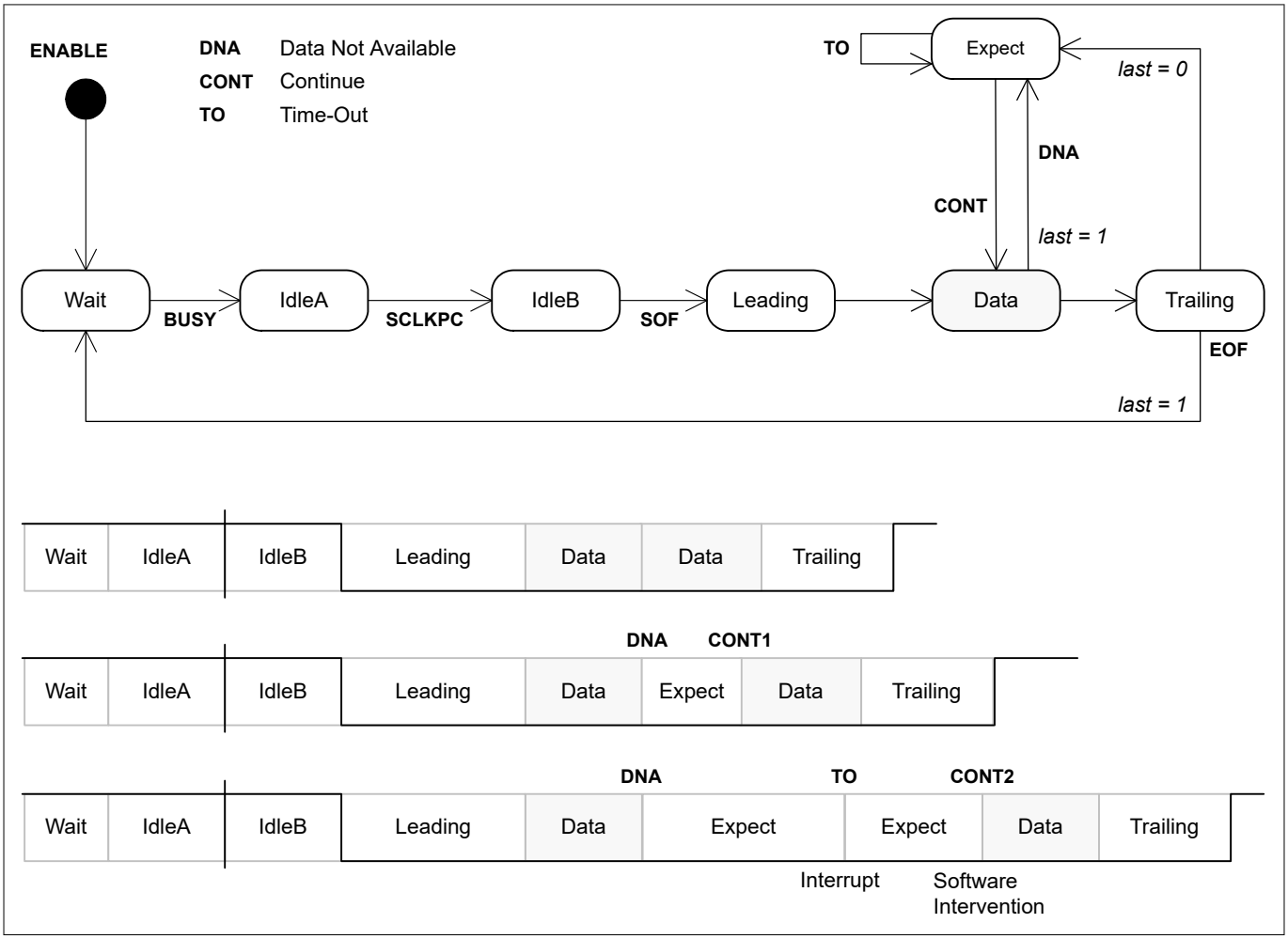


Figure 666 EXPECT cycles of the QSPI state machine

30.3.2.7 External slave select expansion

The external slave select expansion mode (delayed mode) is activated with the bit GLOBALCON.DEL0. This mode enables the connection of an external multiplexer for communication with up to 16 slaves using a single QSPI module while only utilizing 4+1 SLSO outputs pins of the device.

In this mode the binary coding of the bit-field BACON.CS (bits 0 to 3) drives up to four SLSO pins, not demultiplexed by the QSPI, but connected to the selection input of an external demultiplexer. The SSOC.AOL bits can invert the SLSO0...15 signal levels individually; SSOC.OEN bits enable the signals to the output pins.

Using the standard reset values of the register MSSMUX defaults to a similar behavior to previous QSPI generations (shown in the following figure), where BACON.CS drives directly the SLSO1 to SLSO4 signals, and SLSO0 is used for the strobe signal.

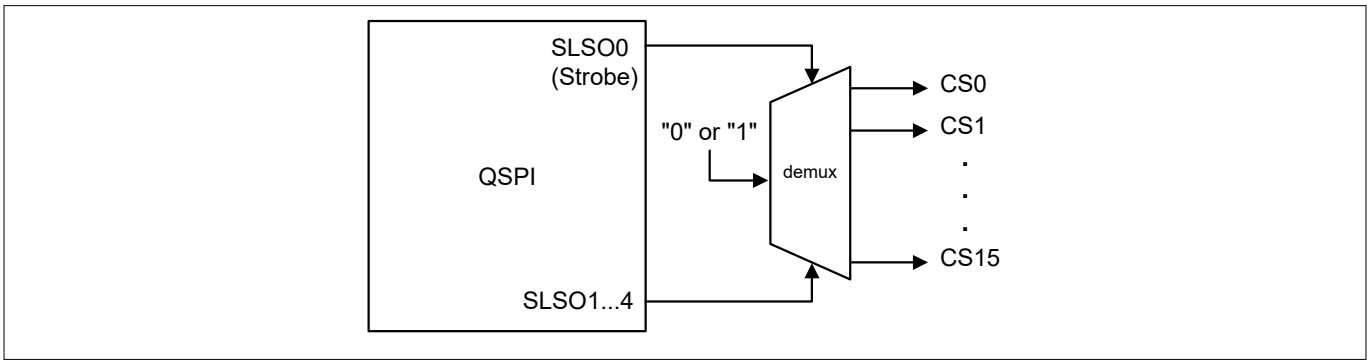


Figure 667 Demultiplexing the slave select signals externally

The External slave select expansion mode is not to the usage of the pins SLSO0-4, but any SLSOs canbe used. I n applications some of the pins corresponding to SLSO0-4 might be used by other features. The figureabove sh ows how to select up to four slave select outputs for the external MUX selection bits using the bits MSSOUT0-3 and how to select the slave select the pin for outputting the strobe signal using the MSSOUTS bitfield.

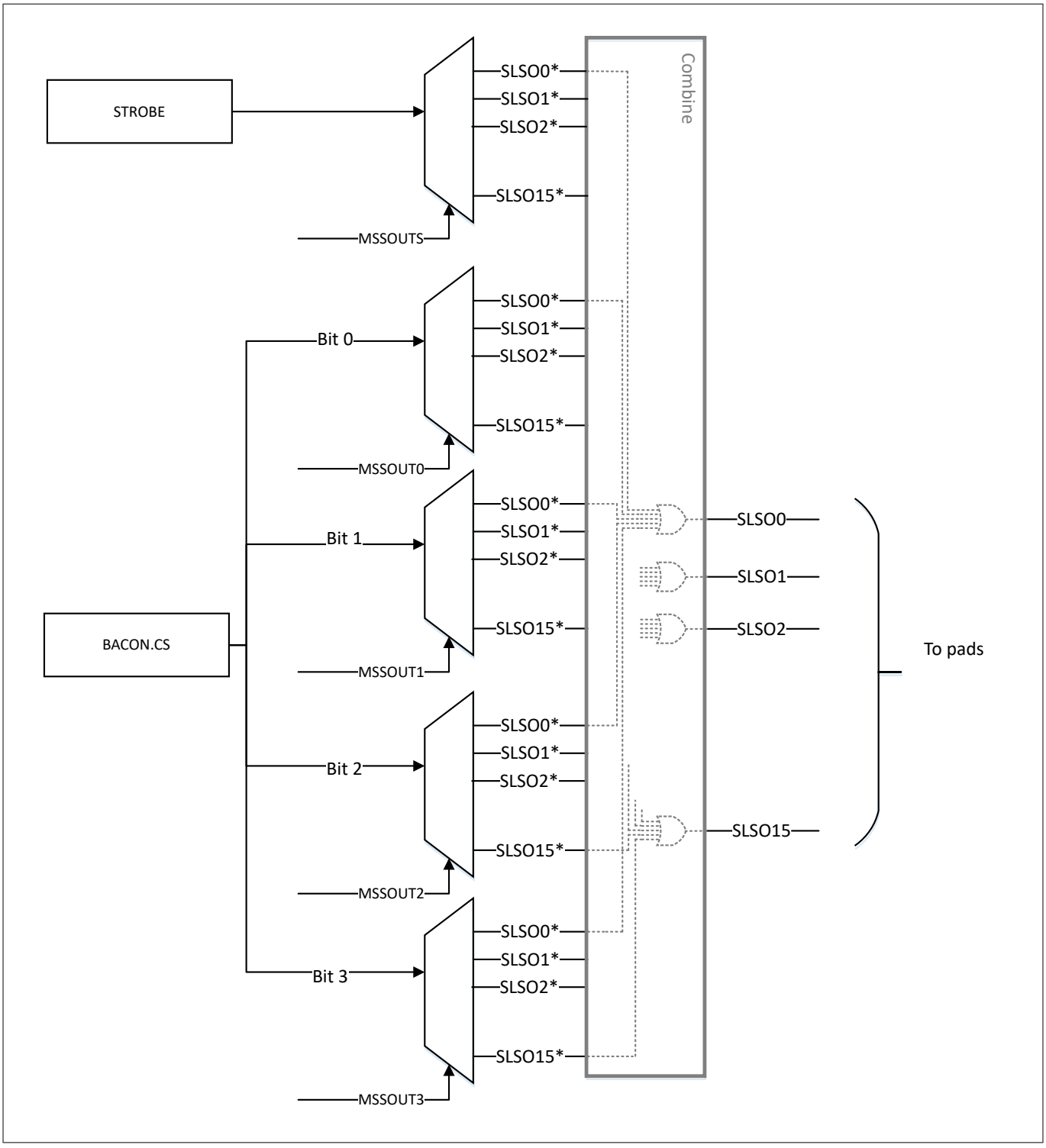


Figure 668 Selecting arbitrary SLSOs for external slave select expansion

In order to ensure glitch free selection, a strobe signal is provided, driven at strobe pin selected by MSSMUXS.MSSOUTS. This signal is delayed relative to the SLSO signals used for external MUX selection bits. This delay is by LS (Lead_Strobe) and TS (Trail_Strobe), equal in duration and can be calculated using the formula for t_{STROBE_DELAY}. The duration of LS and TS can be configured in the GLOBALCON.STROBE in multiples of channel time-quanta units T_Q.

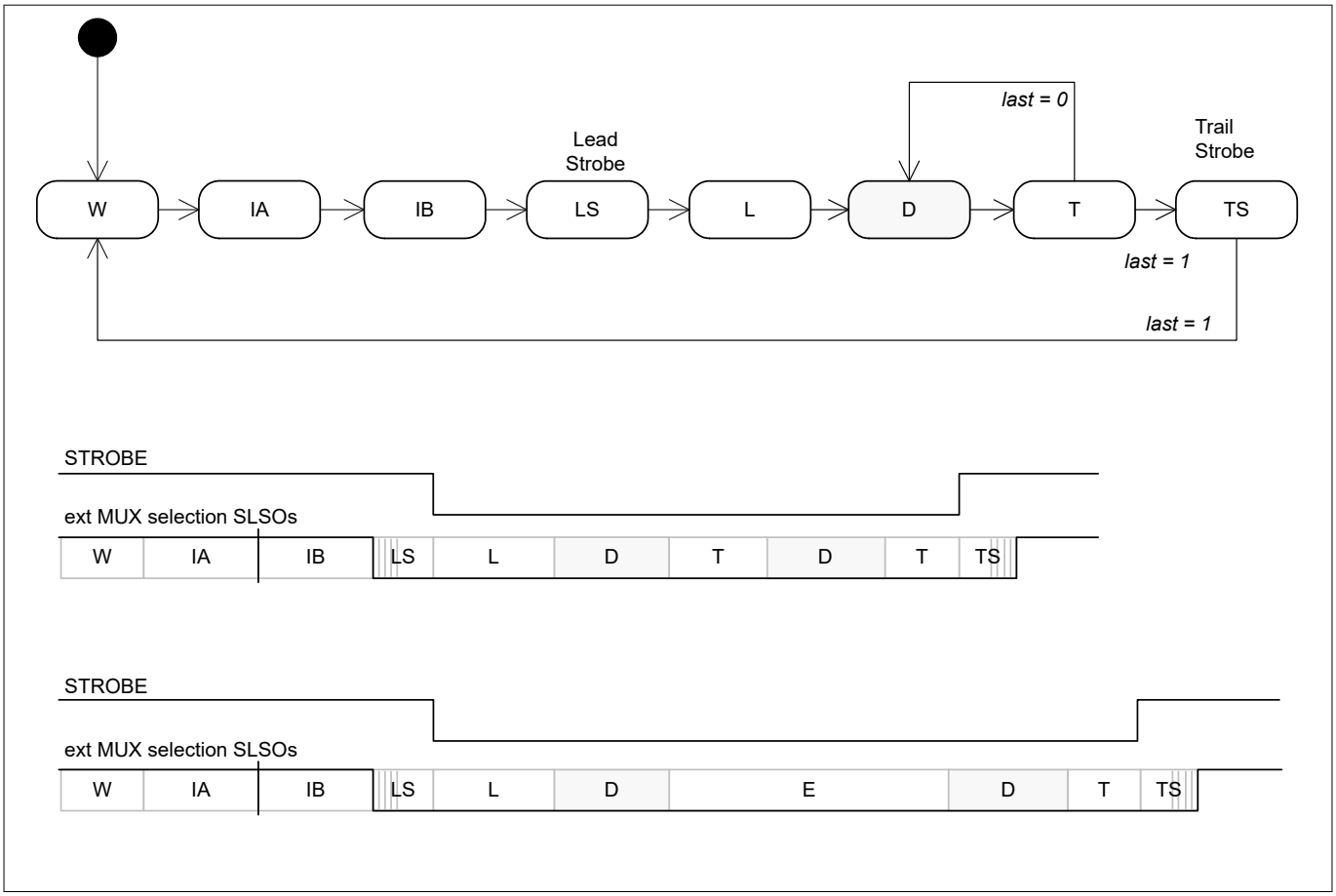


Figure 669 State machine and SLS signals in strobed mode

When using External slave select expansion, all the SLSO signals will be inactive (as per SSOC.AOL), while no communication is ongoing. If a communication is started, the SLSO will change to their corresponding level during "LS" and will go back to inactive during "TS".

Selecting multiple slave simultaneously

It is possible to use the External slave select expansion mode to send messages to multiple slaves simultaneously, as in this mode the SLSO outputs are driven by the binary value of BACON.CS. So if one communication should be received by two slaves, then this can be achieved by using BACON.CS=0b11 for the communication and by mapping the outputs *bit0* and *bit1* to the corresponding SLSO outputs to these slaves. This is done using MSSOUT0 and MSSOUT1. The selected SLSOs will only be active during the ongoing communication (enabled during LS, disabled during TS), as shown in the previous figures.

***Note**: Reception of data from multiple slaves simultaneously is not supported.*

30.3.2.8 High speed mode

The bit-fields A, B and C in ECONz provide means to compensate for the loop delay in SPI communications. But for very high speed communications, these settings of A, B and C might not be sufficient, because they only allow shifting the data sampling point within the current clock cycle. If the loop delay of the system is higher than the period of the envisioned baud-rate, a different approach is needed.

Enabling the High Speed Mode (GLOBALCON.HS=1) allows setting the effective sampling point of the input data on MRST to a point in time after the current clock cycle. For achieving this, an additional parameter D is introduced, which defines an additional phase D. This phase starts after phase B and concurrently with phase

C. The RX data sampling point is then after phases A+B+D in this mode. The phase D has no impact on period or duty cycle of the serial clock.

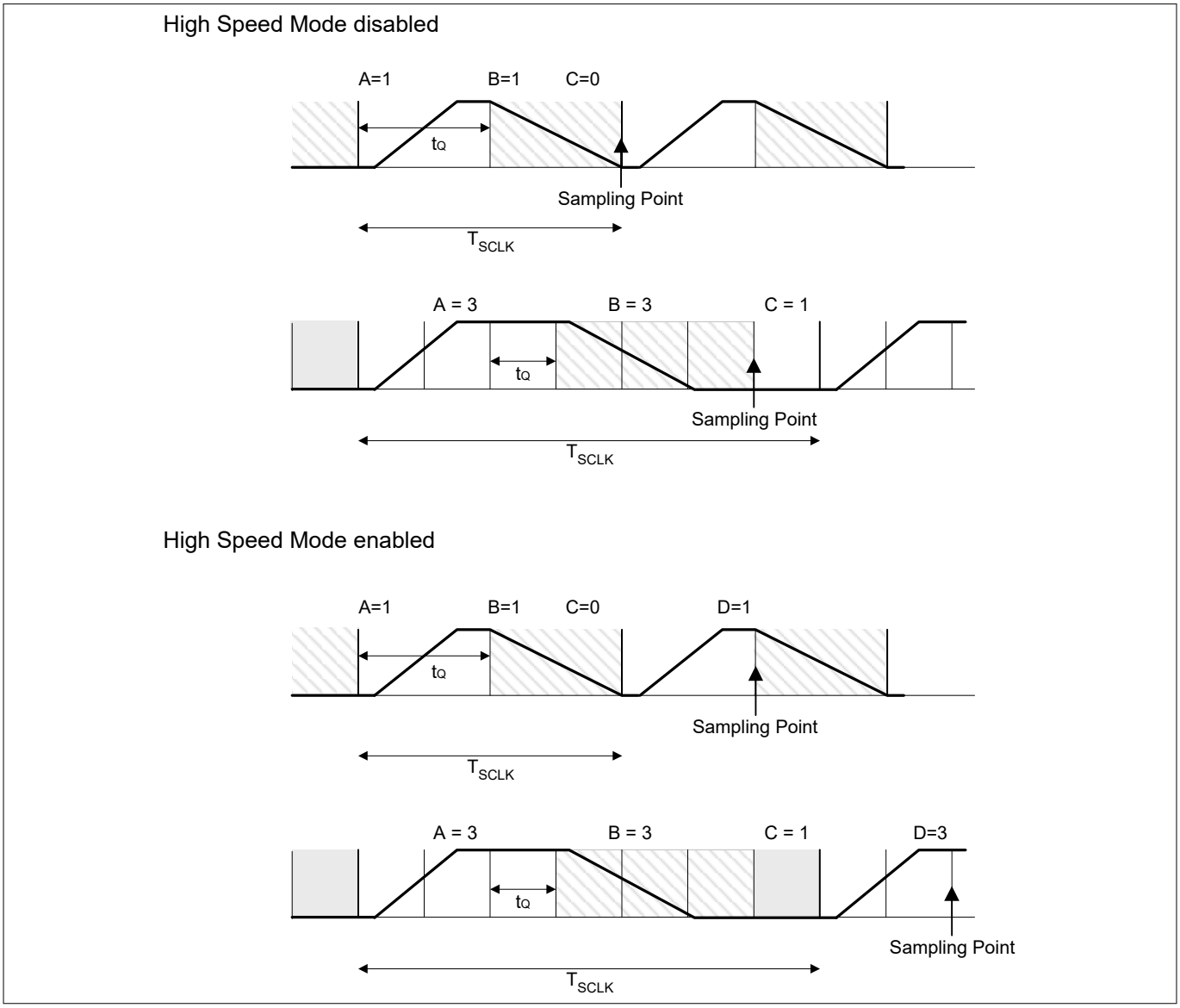


Figure 670 Rx data sampling point in high speed master mode

The figure above visualizes the High Speed Mode. The first two waveforms show two different settings of A, B and C in GLOBALCON.HS = 0. Then the subsequent two waveforms visualize the same settings of A, B and C (same serial clock), but with GLOBALCON.HS=1 and D > 0. In all four waveforms one clock cycle is marked as T_{SLCK} and the corresponding sampling point is given. The waveforms for High Speed Mode show a sampling point outside of the current clock cycle, enabling higher SPI clock frequencies since the compensated loop delay is the sum of phases A+B+D. The last waveform additionally shows the concurrency of phases C and D.

The user must make sure that the TRAIL delay is as least as long as the time corresponding to the length of phase D minus phase C, meaning:

$$(D - C) \times \text{ECONz.}Q \times \text{GLOBALCON.TQ} \leq \text{BACON.TPRE} \times \text{BACON.TRAIL}$$

***Note**: The equations above contain the representative decimal values of the bit-fields in corresponding units, not their actual binary bit-value.*

Related information

Phases of one communication cycle on page 5304

30.3.2.9 Hardware trigger mode

A QSPI hardware triggered **HWT** mode is enabled by setting GLOBALCON.HWTREN=1. In this mode, the CPU writes first the configuration and data in the BACON_ENTRY and DATA_ENTRYx address, respectively and the QSPI module initiates the data transmission on reception of a hardware trigger signal. Different sources for the trigger are selectable in GLOBALCON1.HWTRSEL. The hardware trigger is detected and processed only in the WAIT state. If a trigger signal is received in any other state, then it will be discarded and the corresponding error is flagged in STATUS.ERRORFLAGS and will trigger an interrupt if it is enabled in GLOBALCON1.ERRORENS.

The **HWT** mode is operated in two different ways based on the value of GLOBALCON1.FD bit-field. This bit-field may only be set only when GLOBALCON.HWTREN=1 and Short data mode is used. Setting GLOBALCON1.FD=1 freezes the 32-bit data, which will be sent, so that the same data entry is shifted out on the SPI bus whenever a hardware trigger is received. Setting GLOBALCON1.FD=0, provides the normal QSPI data transmission behavior on every trigger reception. An example illustrating the difference between the two settings is shown in the following figure, here 32-bit data frames are sent with the same configuration on every trigger.

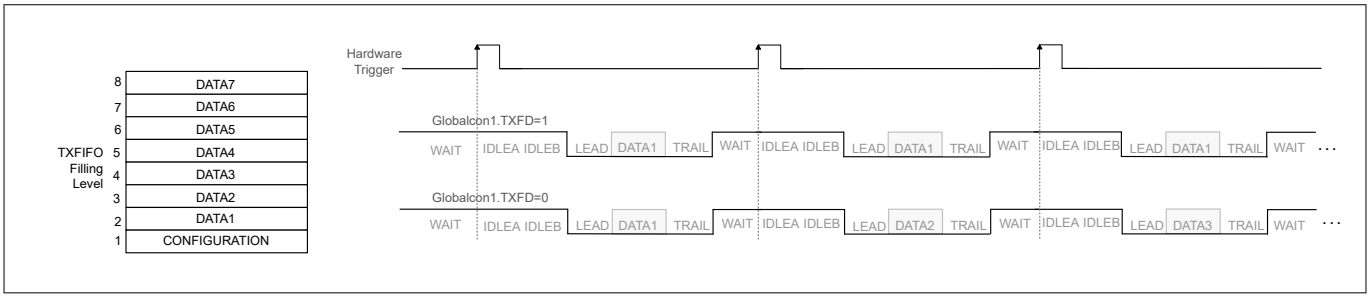


Figure 671 QSPI hardware trigger mode

An empty TXFIFO at the instant of hardware trigger arrival would result in no transaction and the raising of the Trigger Ignored Error. The TXFIFO must be reset by the software whenever switching from a freeze TXFIFO state to an unfreeze TXFIFO state. The selection of the hardware trigger source is configurable based on the GLOBALCON1.HWTRSEL bit-field.

GTM, eGTM and GST trigger inputs

There are two trigger inputs that you can select from by programming GLOBALCON1.HWTRSEL:

- 0 selects GTM

  ***Note**: In products that do not contain GTM, this input is connected to GST.*

- 1 selects eGTM

  ***Note**: In products that do not contain GTM, this input is tied to inactive. It can not be used in these products..*

30.3.3 Slave mode

The slave mode is entered by setting the bit-field GLOBALCON.MS to value 0x1Xb. Before entering the slave mode, the related parameters must be configured, including pin selection for SCLKI, SLSI and MTSR, pin input level, and pull-up/pull-down configuration. The configuration should be done before data transmission starts. Reconfiguration is only allowed when the module is in a wait state. Subsequently a state machine reset should be performed and then the slave mode should be entered.

In slave mode data is being pushed into or out of the module through the serial pins by an external master. The RXFIFO must be regularly emptied by the system in order to avoid data loss, and the TXFIFO must be regularly fed by the system to avoid delivering all “1” data in case of an empty TXFIFO.

If the slave mode relies on a slave select signal to mark the start and, in parallel to the bit counting, the end of a frame, then deactivating the slave signal by the master automatically resets the shift register state machine to wait state.

If the slave mode relies solely on bit counting for determining the frame end, then any SCLKI clock period longer than two bit times is treated as an end of a frame and simultaneously as a baud rate error.

In slave mode, the module simply immediately responds to external clock edges. Therefore the settings for the leading, trailing, idle delays, and duty cycle and sampling point are irrelevant. From all timing settings, only the data length and the baud rate divider setting are relevant. The baud rate setting is important because of its role of watchdog timer of the incoming serial clock.

Generally, the slave mode supports the same user interface as the master mode. This means the BACON-Data sequences remain the same. The SRF bit is ignored.

Limit the data length setting in the slave mode to the range of 2 to 32 bits. The byte setting BACON.BYTE is to be set to zero, that is, it always defines bits.

In case of receive only (simplex receive) the user software must reset the module in order to switch to transmit and receive mode (full duplex). Generally, mode reconfiguration (slave mode to master mode) requires module reset which resets the internal state machines and counters.

In case of slave transmit, the user software must write data to the TXFIFO before the first SCLKI edge of a frame arrives. If the TXFIFO is empty at this point of time, an underflow occurs, all “1” data is delivered, and underflow error interrupt is triggered if enabled. If the TXFIFO is written at the same time when the first SCLKI edge starts the shifting, the underflow occurs, interrupt will be triggered and data from the TXFIFO will be transmitted possibly corrupted. The first bit will be “1”.

In the event of TXFIFO or RXFIFO underflow the FIFOs deliver all “1” data.

In the event of TXFIFO or RXFIFO overflow, the last data is lost. In this case, data or status in the RXFIFO can be lost.

The bit counting in the slave mode operates according to the following rules:

- The incoming bits (or shift clock shift edges) are counted and when the number defined in the BACON.DL is reached, the bits are transferred in the RXFIFO
- If the SLSI input is deactivated, the internal bit counter is reset

In slave mode, sleep requests may be enabled only when there is no pending transmit queue and no pending transmission, and the TXFIFO is empty.

Phase interrupts in slave mode

PT1 interrupt signals the following events:

- BUSY
  - Option 000 from GLOBALCON1.PT1
  - Transmit data is present and is waiting for the shift clock
- SOF
  - Option 010 from GLOBALCON1.PT1
  - Transmission of the first data bit started. It is not connected to SLSI active edge, but to the first clock pulse received
- TBE
  - Option 011 from GLOBALCON1.PT1
  - Data is taken from the TXFIFO by the QSPI shift engine. It will be transmitted with the first shift clock pulse

- RBF
  - Option 100 from GLOBALCON1.PT1
  - Received data is written to the RXFIFO
- EOF
  - Option 101 from GLOBALCON1.PT1
  - The last data bit has been received
- DNA
  - Option 110 from GLOBALCON1.PT1
  - Transmit data not available at the first shift clock edge reception

PT2 interrupt signals the following events:

- EOF
  - Option 101 from GLOBALCON1.PT2
  - SLSI deactivated (rising edge on the SLSI pin)

30.3.3.1 Shift clock phase and polarity in slave mode

In slave mode the shift clock phase and polarity are fixed to CPH=1 and CPOL=0 (have to be programmed accordingly in the ECONz (z=0-7) register selected by BACON.CS) and slave select polarity is fixed to low active. The baud rate that the slave expects is defined in the ECONz (z=0-7) register pointed at by the BACON.CS.

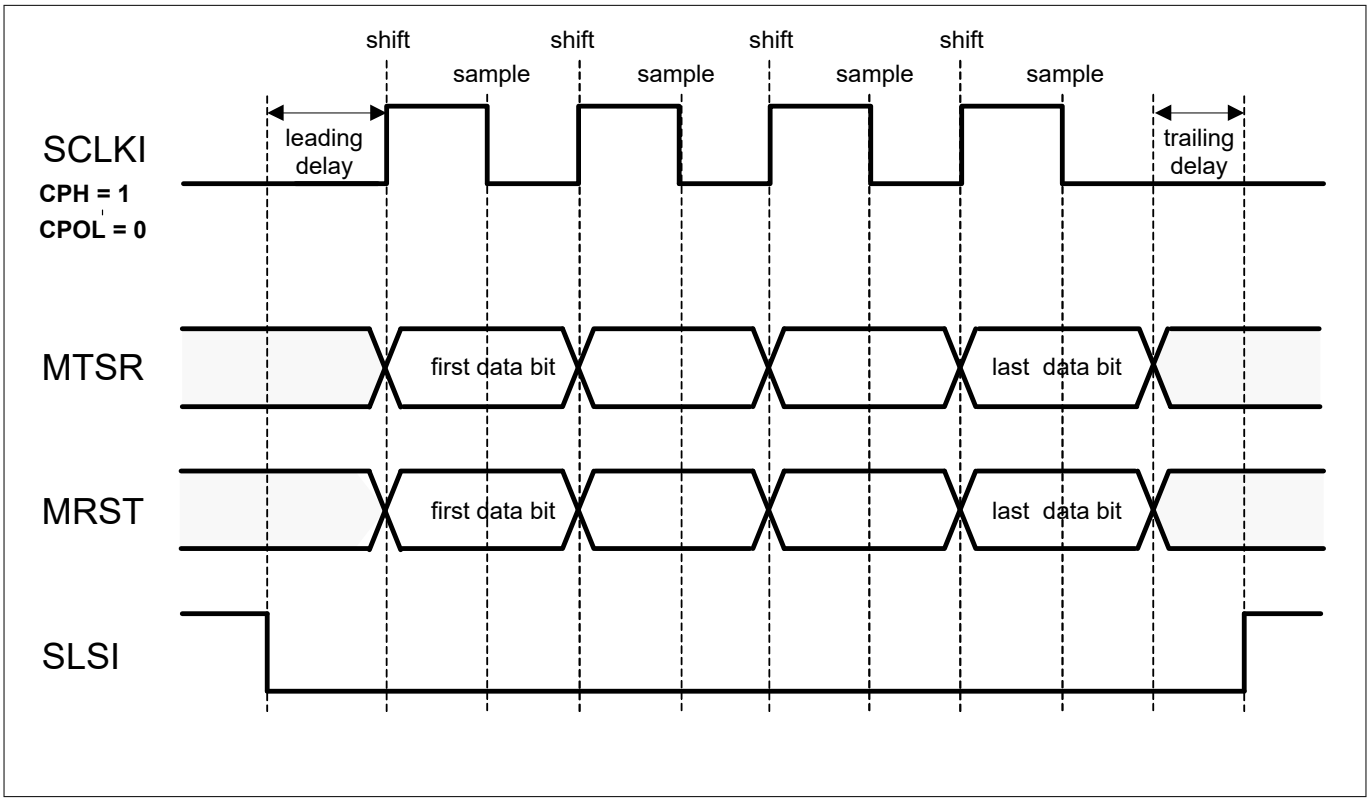


Figure 672 Slave transfer, CPH = 1, CPOL = 0

***Attention**: Slave re-configuration must only be performed in a gap between the frames. However, as a general recommendation, slave mode configuration should be done once only and before the transition starts.*

30.3.3.2 Shift clock monitoring

The QSPI module provides two mechanisms for monitoring the shift clock input signal in slave mode. These mechanisms monitor:

- If the receiving shift clock frequency (baud rate) lies within a certain range and
- If there are spikes on the shift clock input

In case such disturbances are detected, the reaction is always the same:

- The corresponding error flags in STATUS.ERRORFLAGS are set, which can be cleared with FLAGSCLEAR.ERRORCLEARS
- The common error interrupt is raised if enabled in GLOBALCON1.ERRORENS and
- The bit GLOBALCON.EN is automatically cleared, if this feature is enabled by the bit GLOBALCON.AREN. The user software can activate a reset afterward

In case of too high baud rate error, baud rate detection and spike detection both react, where spike detection is more effective. Spike detection reacts immediately, baud rate detection requires several bits with higher baud rate. Baud rate detection uses a window two nominal bit times wide and counts the real bit times: double baud rate error requires at least four real shift clock pulses to be detected, triple baud rate error requires six and so

on. Baud rate detection does not react to all spikes smaller than one kernel clock period. In case of too low baud rate, baud rate error detection reacts immediately, spike detection ignores this case.

30.3.3.2.1 Baud rate error detection

If the shift clock has less than half or more than double the expected baud rate, then:

- The corresponding flag is set in STATUS.ERRORFLAGS
- Error interrupt is raised if enabled in GLOBALCON1.ERRORENS
- Automatic clear of GLOBALCON.EN follows, if enabled by GLOBALCON.AREN

30.3.3.2.2 Spike detection

Short spikes on the clock line due to ground bouncing or clock ringing due to over/undershoots would cause one or more extra bits to be written to the front end receive FIFO of the slave. This extra bit, if no measures are taken by the software, would cause all the subsequent received frames to be corrupted (shifted).

The spike detection mechanism operates up to 50 MBaud and checks if a bouncing spike has occurred on the clock line near to the correct receiving clock edges.

If a spike is detected, the software should use GLOBALCON.RESET bit-field, and should clear the corresponding status flags. Alternatively a complete module reset may be performed through registers RST_CTRLA and RST_CTRLB.

The spike detection is based on two mechanisms:

- Detection of two close writes to the front end receive FIFO by detecting filling level of two
- Using an inverted watchdog set to approximately 75% to 80% SCLKI clock period, which is the optimal range

The inverted watchdog is a timer which starts to count triggered on the write event to the front-end receive FIFO of the slave. During the time the watchdog counts, no edge is allowed to occur. An edge in the forbidden time raises a spike error event.

If a spike occurs in the last 25% of the SCLKI period, when the watchdog has stopped counting, the watchdog will be triggered to count by the spike. The next regular edge will violate the forbidden time and raise a spike event.

The watchdog counts $\mathrm{ECONz}.B + \mathrm{ECONz}.C$ time quantas. The following constraint applies:
$50\,\% \times T_{\mathrm{SCLKI}} < \mathrm{ECONz}.B + \mathrm{ECONz}.C < 100\,\% \times T_{\mathrm{SCLKI}}$.

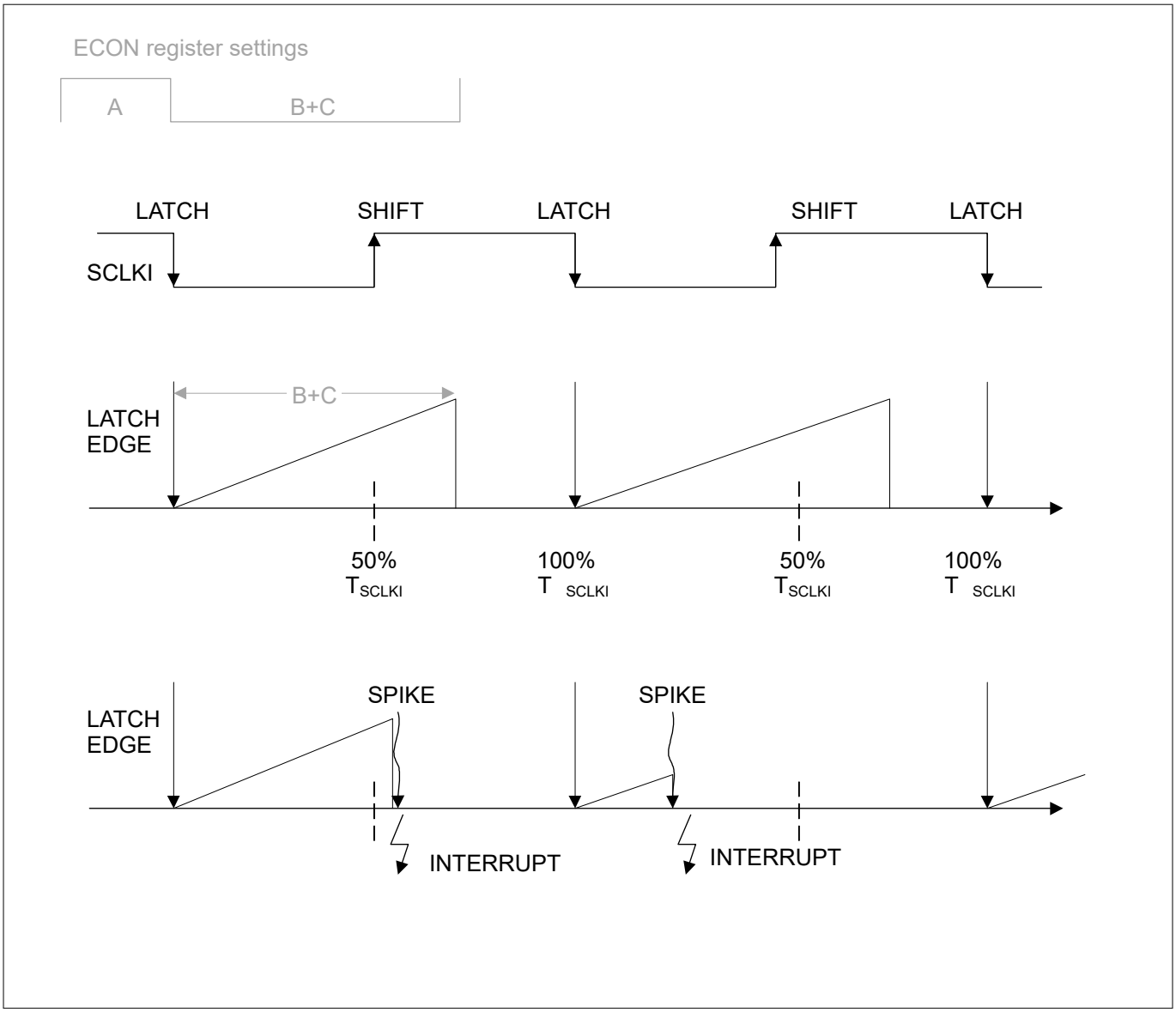


Figure 673 Spike watchdog

30.3.3.2.3 Shift clock monitor flags

The spike detection mechanism sets the flag STATUS1.SPD, if it has detected a spike. The baud rate error detection mechanism sets the flag STATUS1.BRD, if it has detected excessive baud rate deviation.

Writing 1 to SPD and BRD sets the bit and causes an interrupt, if enabled. Writing 0 has no effect.

Both signals are ORed. This combined signal is shown in STATUS.ERRORFLAGS and raises an interrupt, if enabled in GLOBALCON1.ERRORENS. Automatic clear of GLOBALCON.EN follows, if enabled by GLOBALCON.AREN.

If the STATUS.ERRORFLAGS[2] bit is cleared through FLAGSCLEAR.ERRORCLEARS[2], both flags SPD and BRD are automatically cleared.

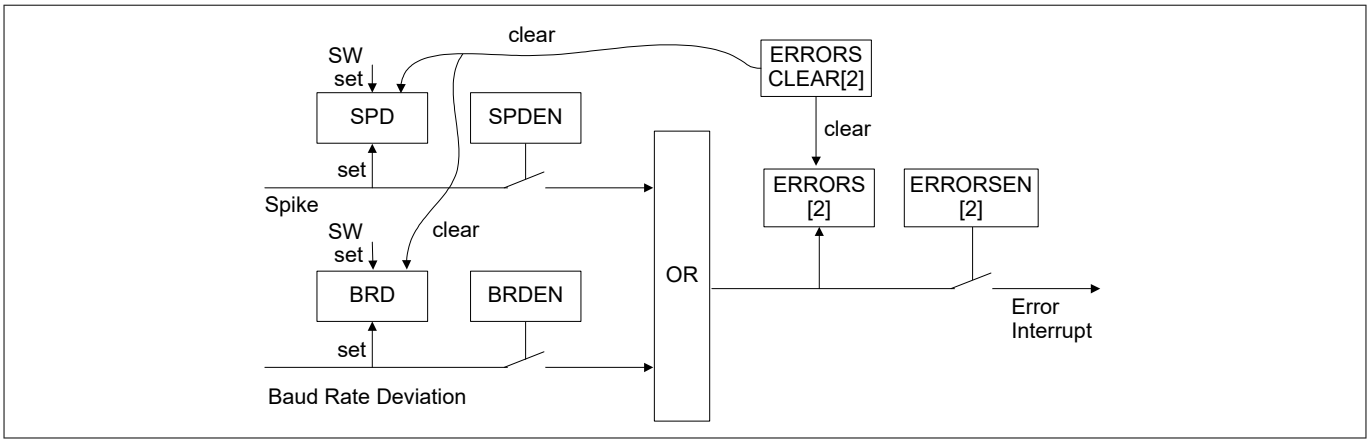


Figure 674 Shift clock monitor flags

30.3.3.3 Parity

Parity settings for transmit and receive are defined by the corresponding bit-fields in the BACON register.

The parity bit is concatenated to the data bits at transmission time. This means that a frame containing 16-bit data is 17 bits long (data + 1 parity). The transmitted parity bit can be read from the STATUS.TPV bit.

The parity bit is removed from the received bits at receive time automatically. The received parity bit can be read from the STATUS.RPV bit.

Attention**: **Parity is available only in short mode, that is, only for data of up to 32 bits length.

Attention**: **The parity bit is concatenated to the data at the LSB location. If the bit-field BACON.MSB = 0 it is shifted out first, else the parity bit is shifted out last.

30.3.4 Operation modes

Generally, the QSPI module transmits a RAM queue of an arbitrary length, fed by a DMA module, and makes pauses between the queues. The QSPI module does not know anything about the queue length or the current position within the queue or the future schedule of the messages that may come.

If the QSPI module gets a request for switching off the clock or pausing the module, it disables all service request lines.

The QSPI module provides two options for switching off the clocks or pausing the module: hard suspend (as soon as possible regarding for example the length of interrupt request pulses), and soft suspend (transition to the PAUSE state after reaching predefined state). The requests can be triggered by hardware signals (disable, suspend, sleep) or by software write to GLOBALCON.EN bit.

Disable, pause and run

After being enabled by clearing the CLC.DISR bit, the QSPI enters the PAUSE state. In this state, the QSPI module can be initialized, the TXFIFO pre-filled, and then, by setting the GLOBALCON.EN bit, put into a RUN state.

The software requests PAUSE state of the running module by clearing the GLOBALCON.EN bit. The software can detect that the QSPI module has reached the PAUSE state by polling for the GLOBALCON.EN=0 and STATUS.PHASE=0 (indicating WAIT state, master mode only).

In PAUSE state, the user interface of the module is active. The registers and the FIFOs can be read and written. However, both the receive and transmit state machines are inactive, in master as well as in the slave mode. This allows for reconfiguration of the module without affecting the serial bus.

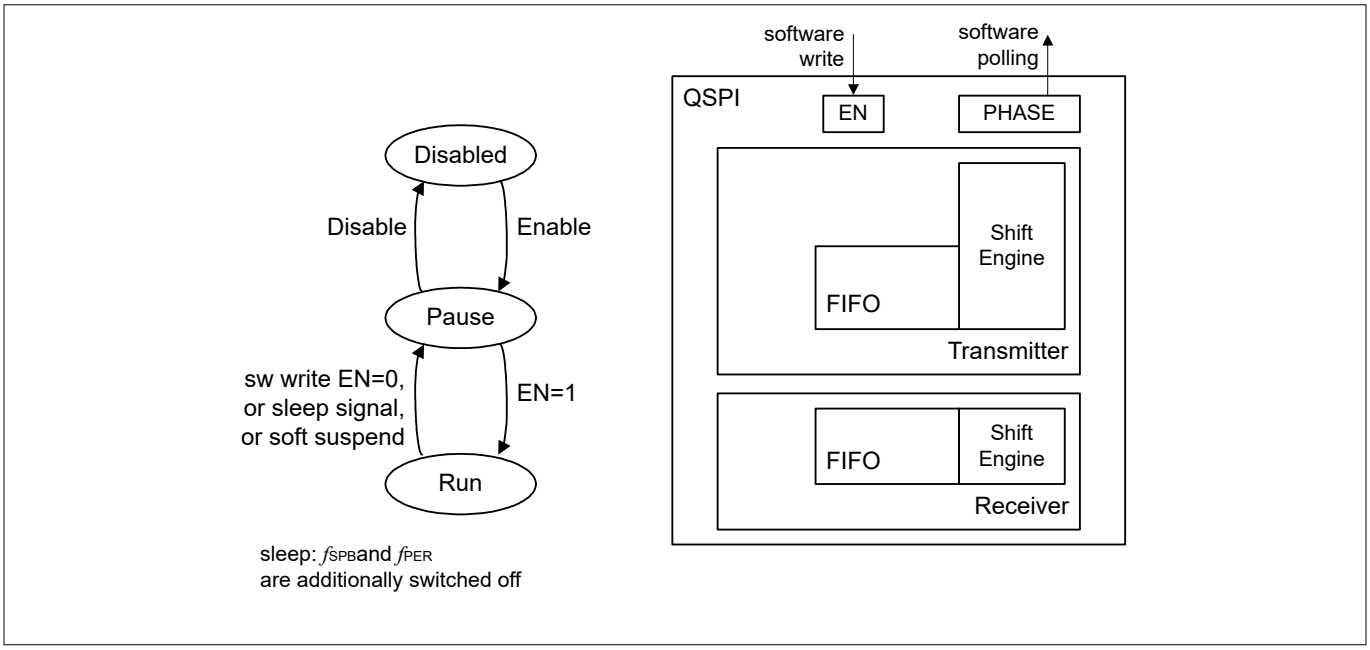


Figure 675 Pause and run overview

The concept of the PAUSE state is reused for handling the soft OCDS suspend and Sleep requests. In contrast to active Pause state and OCDS suspend, the sleep state has the special property that (due to the low power requirement) both FPI and QSPI clocks are switched off.

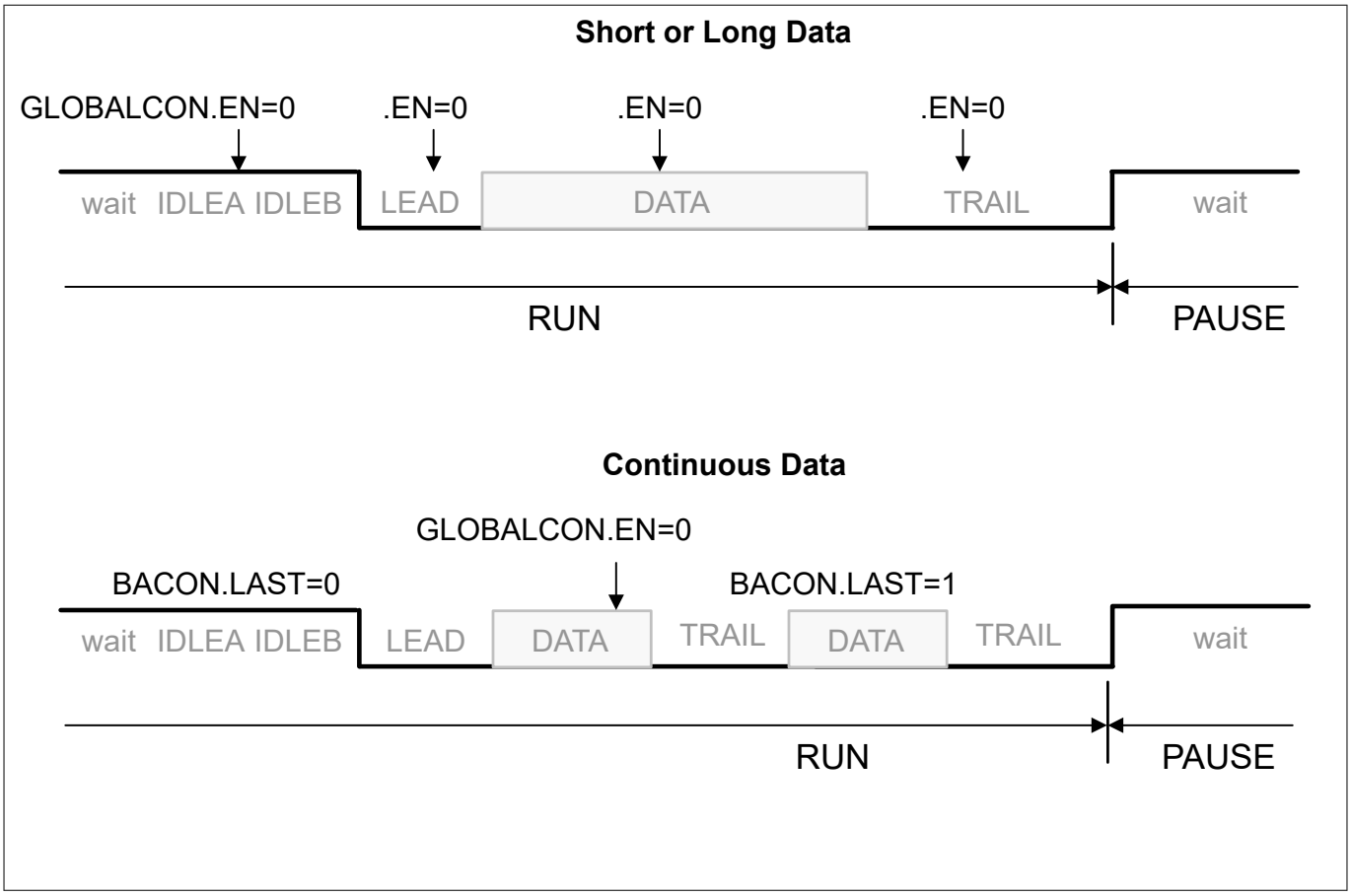


Figure 676 Entering the pause state with GLOBALCON.EN = 0

If a running module receives a request to PAUSE, it waits for the end of the last TRAIL phase of a frame.

For all modes (Short, Long and Continuous), the PAUSE state is indicated by GLOBALCON.EN=0 and STATUS.PHASE=0, master mode only.

30.3.4.1 OCDS suspend

See debug information.

Related information

Debug information on page 5402

30.3.4.2 Sleep mode

Sleep signal is a system wide hardware signal requesting from all modules to go to low power saving state as soon as possible. For the QSPI module, going to sleep mode can be disabled or enabled by setting or clearing the CLC.EDIS bit. The module enters the sleep as soon as possible and remains in this state as long as the hardware sleep signal is active. Upon deactivation of the sleep signal, the QSPI wakes in PAUSE state (GLOBALCON.EN bit is cleared by the QSPI module) and waits for the software to enable the GLOBALCON.EN bit.

If CLC.EDIS bit is cleared, the sleep mode is enabled and the module enters the sleep state:

In master mode:

- After ending the TRAIL delay if a transmission is in progress
- Or immediately if the module is in WAIT state

In slave mode:

- After the bit counter has counted the full predefined number of bits as defined in BACON.DL field
- Immediately if the module is in wait state, no transmission ongoing and shift register empty and TXFIFO empty
- Or immediately if the SLSI is inactive (the one selected by PISEL)

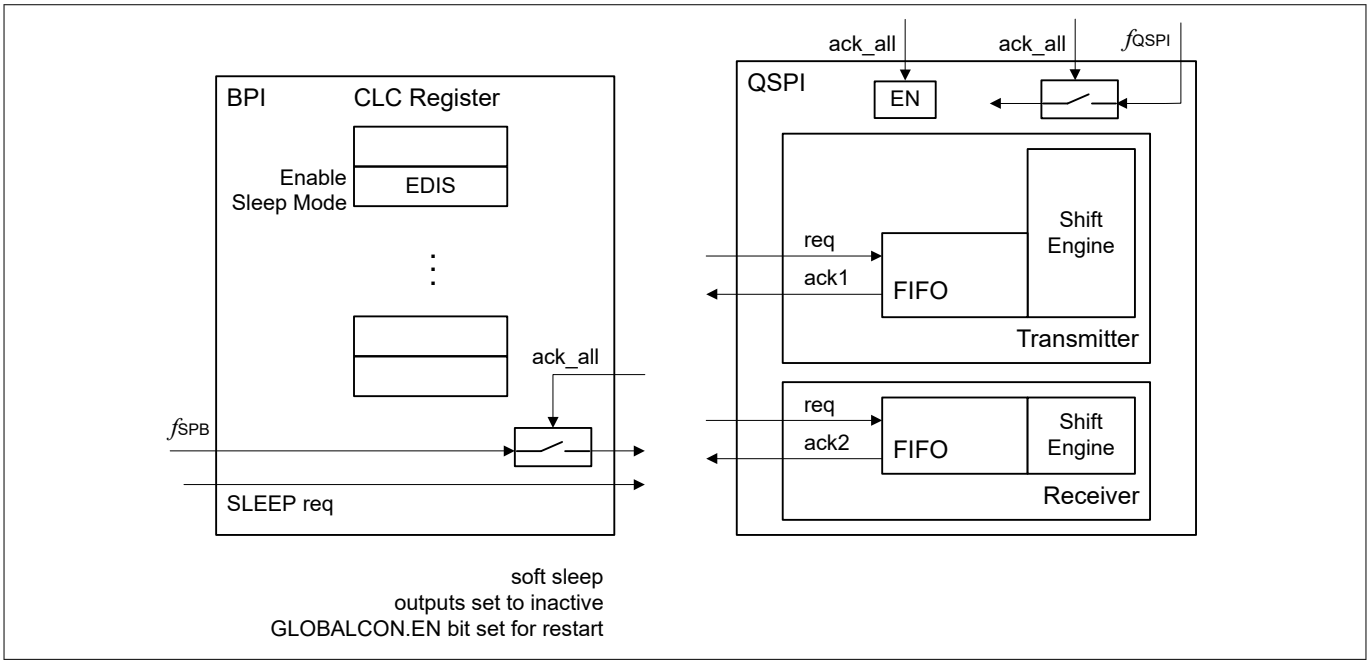


Figure 677 Sleep overview

It is responsibility of the user software to have the sleep enabled in a safe time intervals, where no race conditions or pipeline effects between the QSPI module and the DMA can occur in case of sleep request.

***Note**: CLC disable request and sleep request block immediately kernel register write accesses. This includes also the TXFIFO access. Therefore if the module expects more data (like in long or continuous mode), it will enter the expect phase and remain there and never do an acknowledge and enter the required state. Periodic timeout interrupts will be generated. Therefore it is not recommended to request a sleep or disable during an ongoing long or continuous transfer.*

30.3.4.3 Disabling the QSPI

The software can switch off the QSPI module by setting the CLC.DISR bit. It is responsibility of the user software to issue a disable request in a safe time interval, where no race conditions or pipeline effects between the QSPI

module and the DMA can occur. This is no issue if the switch-off procedure is one-way, that is no continuation of operation without reinitialization or reset of the module is expected. GLOBALCON.EN bit is cleared.

The QSPI module itself behaves in the following way.

In master mode, like entering the sleep mode:

- After ending the TRAIL delay, if a transmission is in progress
- Immediately, if the module is in WAIT state

In slave mode:

- After the bit counter has counted the full predefined number of bits as defined in BACON.DL and BYTE bit-fields, if a transmission is ongoing
- Immediately, if the module is in wait state. No transmission ongoing and shift register empty and TXFIFO empty

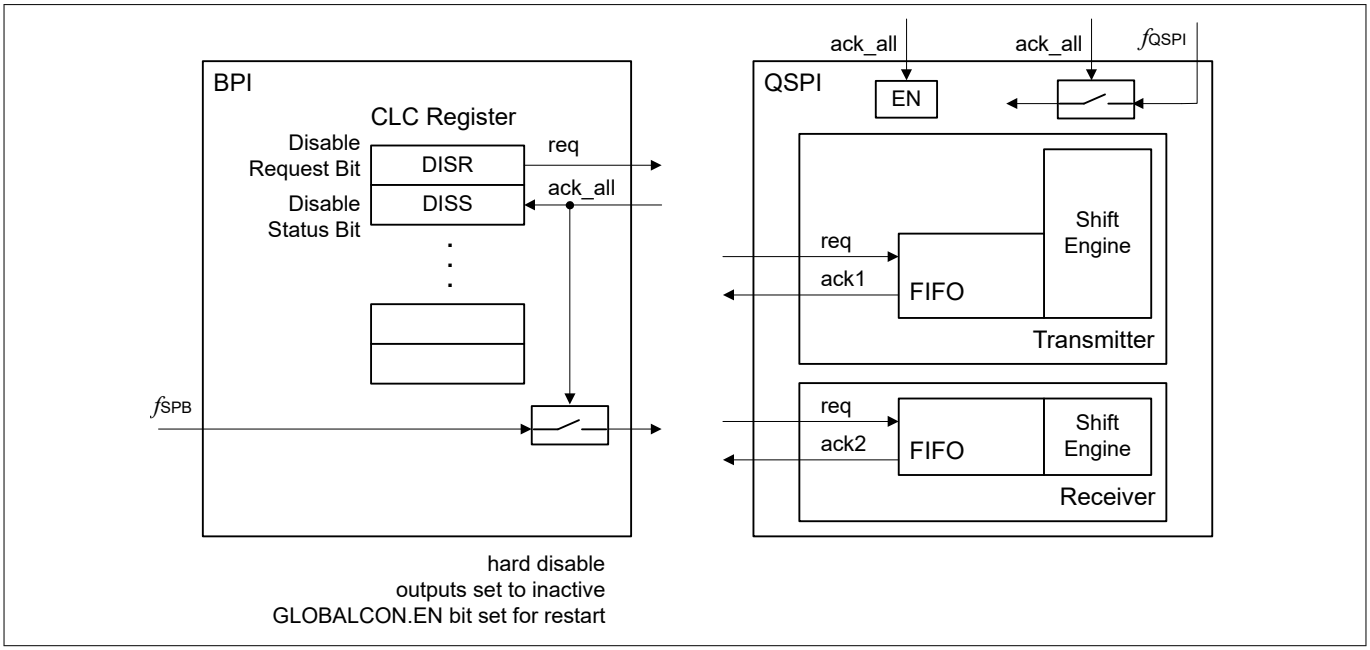


Figure 678 Disable overview

30.3.5 User interface

This section describes the possibilities available for transferring data between the on-chip RAM memories and the QSPI module using the transmit and receive FIFOs. The QSPI features one $8 \times 32 - \text{bit Tx FIFO}$ and one $8 \times 32 - \text{bit Rx FIFO}$. The address ranges for writing data and configuration to the Tx FIFO and for reading data and status from the Rx FIFO consist of address locations with special properties:

- Tx FIFO
  - DATA_ENTRY - writes to any of these eight functionally identical locations are always interpreted as data
  - BACON_ENTRY - writes to this location are always interpreted as configuration
  - MIX_ENTRY - writes to this location are interpreted as data or configuration, based on a set of rules
- Rx FIFO
  - RX_EXIT - reads from this location deliver either data or optionally status, based on a set of rules

Use either MIX_ENTRY or the combination DATA_ENTRY/BACON_ENTRY for sending a frame. Do not combine DATA_ENTRY/BACON_ENTRY and MIX_ENTRY for sending a frame.

The following list gives an overview of the operating modes of QSPI and usage of the TxFIFO entry addresses:

- Short Mode: MIX_ENTRY, DATA_ENTRY/BACON_ENTRY
- Long Mode: MIX_ENTRY, DATA_ENTRY/BACON_ENTRY
- Continuous Mode: MIX_ENTRY, DATA_ENTRY/BACON_ENTRY
- Single Configuration Multiple Frames: MIX_ENTRY
- XXL Mode: DATA_ENTRY/BACON_ENTRY
- Move Counter Mode: DATA_ENTRY/BACON_ENTRY

30.3.5.1 Transmit and receive FIFOs

The Tx_FIFO is filled with both configuration and data entries, which are distributed to the BACON and the shift register at the exit of the Tx_FIFO. The TxFIFO keeps track if an entry is configuration or data. The control and the data elements are automatically distributed to the BACON and shift register. If no data is available, the new BACON is pending.

If a frame is finished and there is data in the TXFIFO:

- In short mode, a new frame starts automatically with the same configuration as the previous one
- In long mode, the extra data is ignored

TXFIFO error conditions:

- FPI bus write to a full TXFIFO generates an overflow interrupt. The write is ignored
- In case of slave mode, $(\mathrm{GLOBALCON.MS} = 1X)$, in case of an underflow, only 1 are delivered

RXFIFO error conditions:

- FPI bus read from an empty RXFIFO generates an underflow interrupt, and delivers only 1 bits
- Hardware attempt to write a full RXFIFO with data or status generates an overflow interrupt. The write attempt is ignored by the RXFIFO
- If the GLOBALCON.SRF bit is set, the communication is stopped in case of full FIFO in master mode. In slave mode, the bit is ignored, no stop can be done
- If the FPI bus frequency is very slow, and the baud rate very high, it can happen that some data is lost on the way between the shift register and a not full RXFIFO. In this case, an overflow interrupt is generated, although the RXFIFO is not full

The Rx_FIFO can be filled with both status and received data entries, depending on GLOBALCON.SI.

***Note**: If a 8-bit or 16-bit write is executed to the TXFIFO entry addresses, the data is mapped 1:1 to its location within the 32-bit wide TXFIFO entry, and the rest is padded with zeros.*

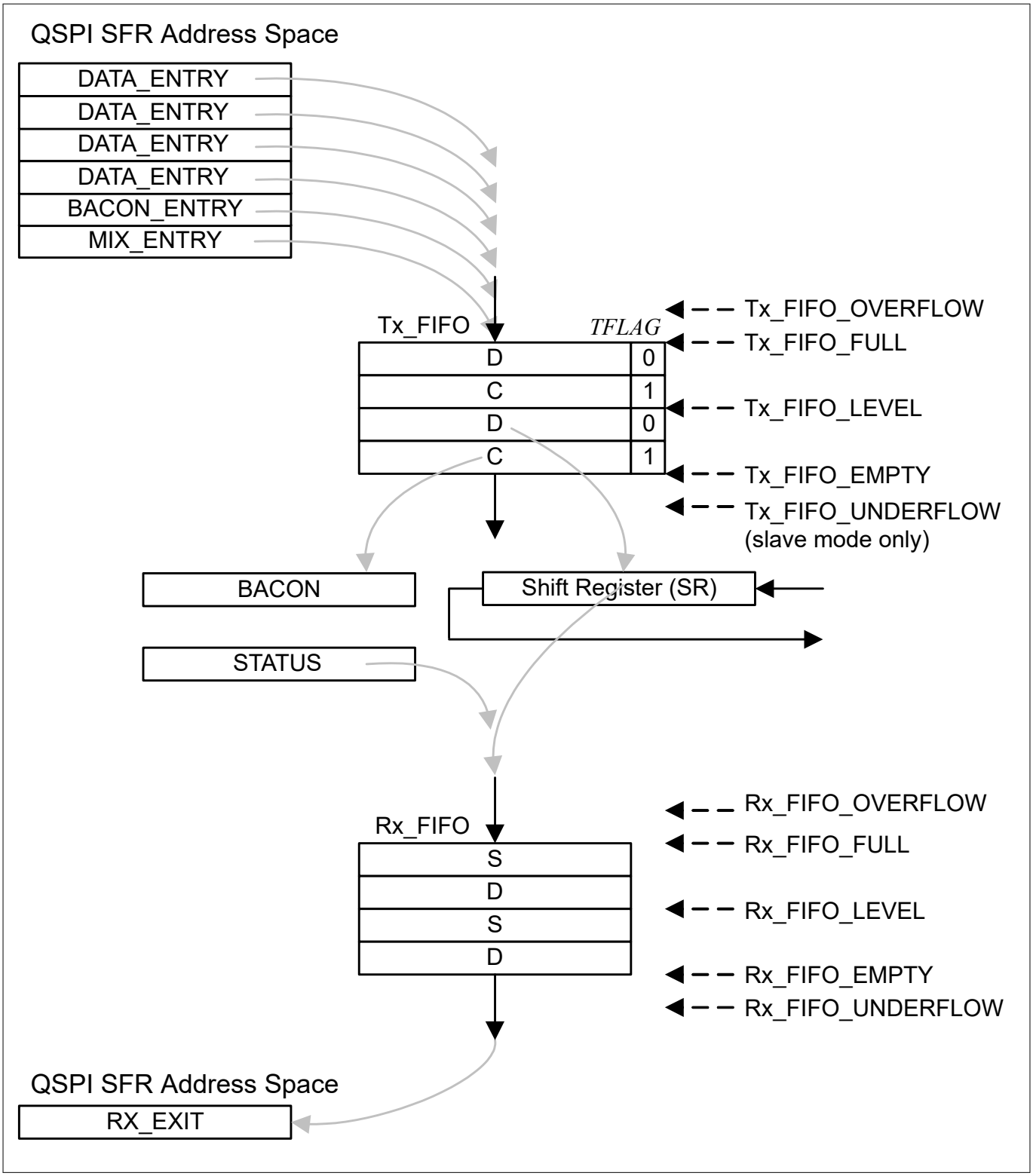


Figure 679 Architecture of the Tx and Rx FIFOs

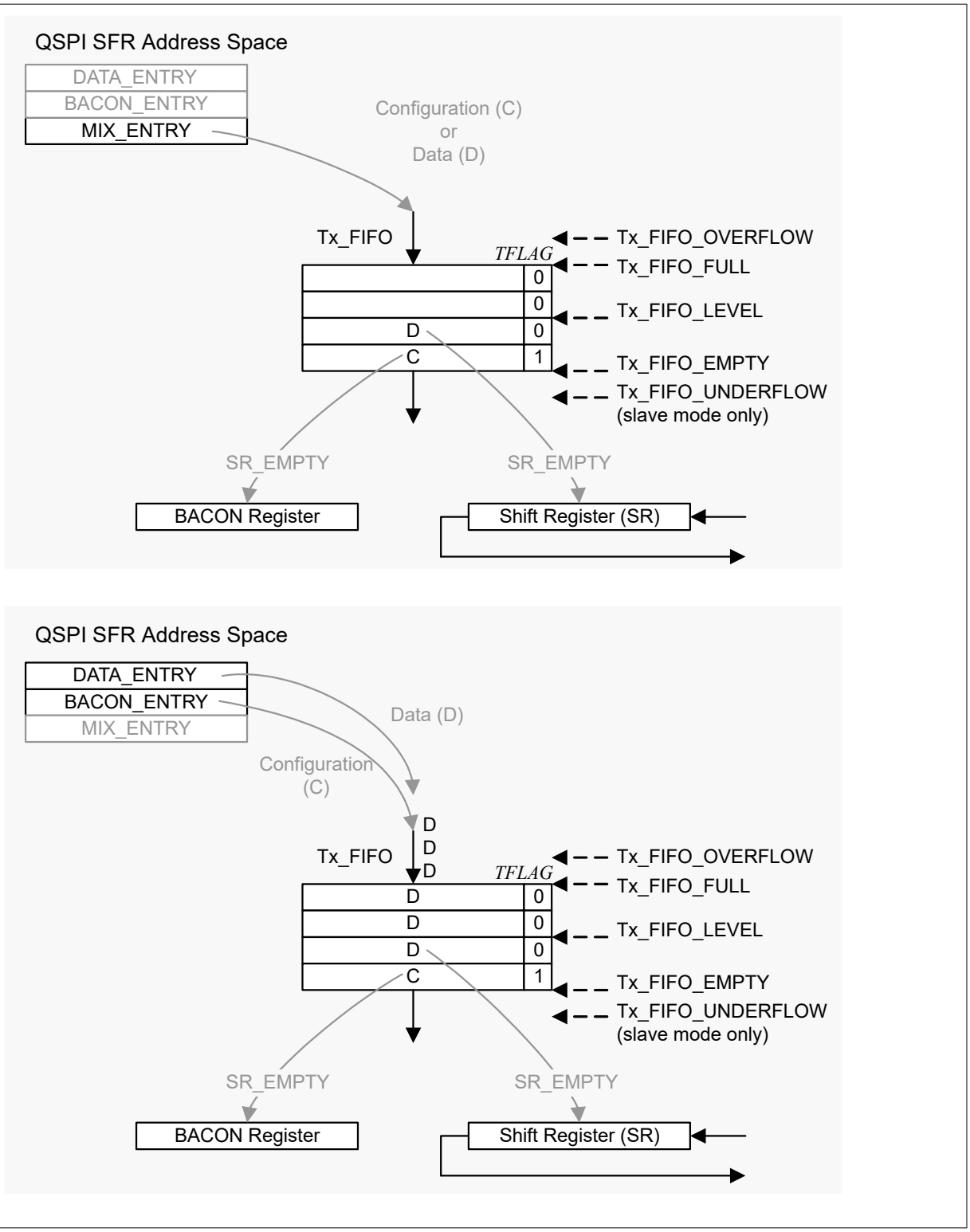


Figure 680 Using Tx FIFO mix and bacon/data entries

30.3.5.1.1 Short data mode

In Short Data Mode, the QSPI module transmits single data with a length of 2 to 32 bits in one frame. This mode is defined by BACON.LAST = 1 and BACON.BYTE = 0.

The transfer cycle is a sequence of the following phases: W?_I_L_D_T. The symbol "?" means that the phase is optional (duration of 0 allowed).

For example, a Tx_FIFO event can be used to trigger one DMA transfer (consisting of two 32-bit moves) to transfer both the BACON and DATA entries from an on-chip RAM memory to the QSPI module.

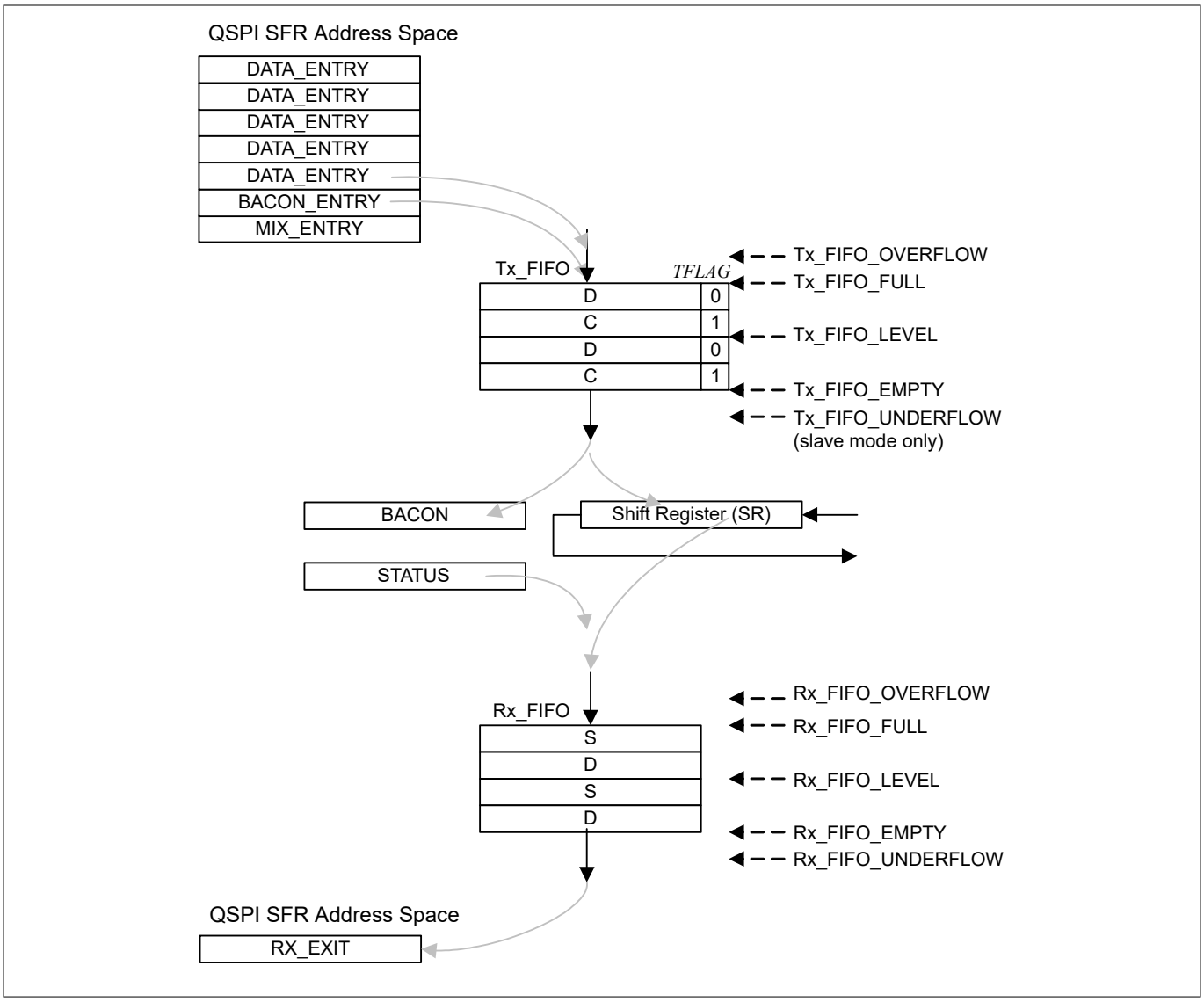


Figure 681 User interface in short data mode

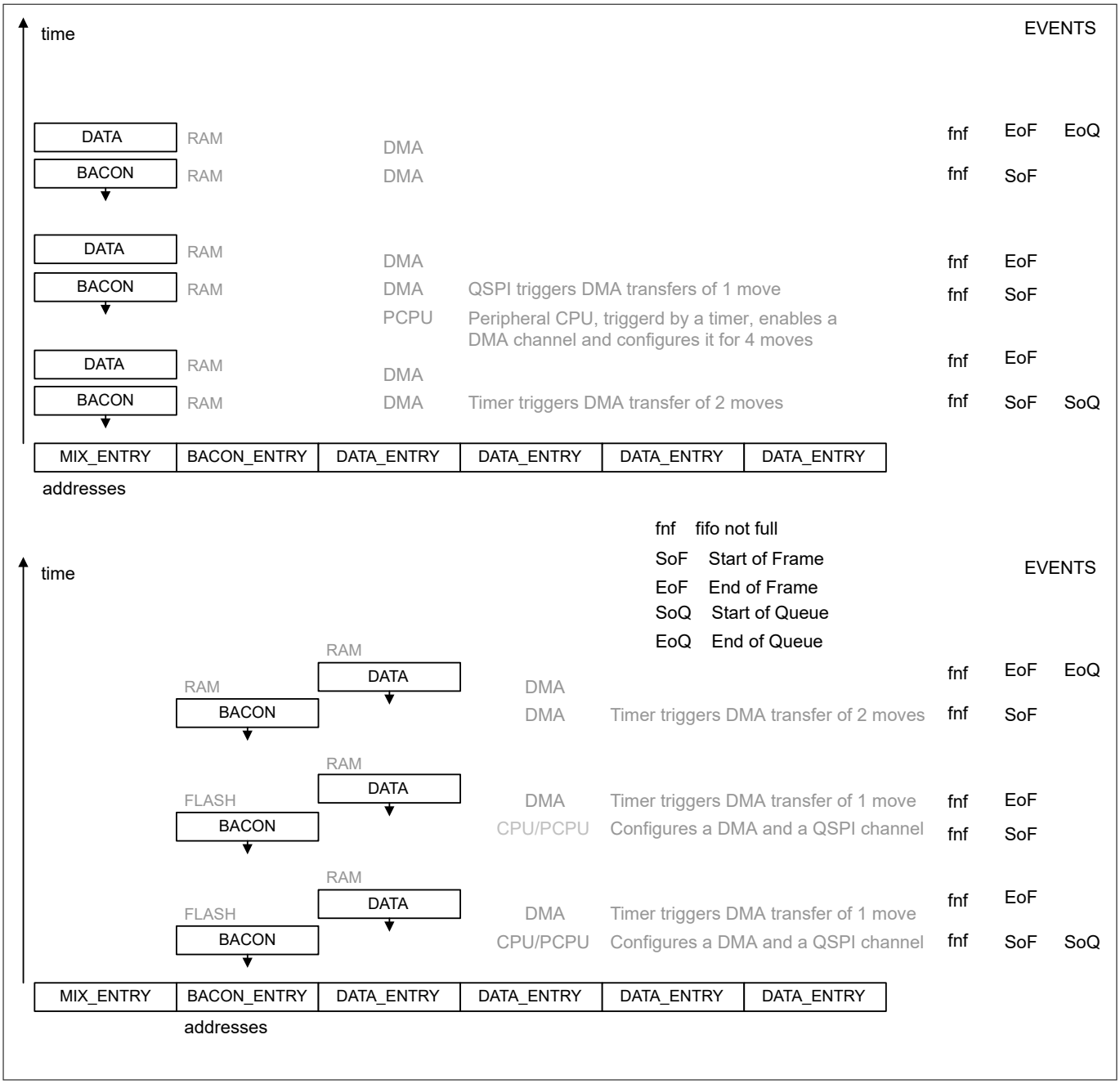


Figure 682 Use diagram of the user interface in short data mode

30.3.5.1.2 Long data mode

In Long Data Mode, the QSPI module transmits bursts of up to 32 bytes (256 bits) in one frame. This mode is defined by programming BACON.LAST = 1 and BACON.BYTE = 1.

One transfer cycle in long data mode is a sequence of the following phases: W?_I_L_D_T. The W_I_L_D_T_I_L_D_T sequence is possible. The symbol "?" indicates that the phase is optional (duration of 0 allowed). Each "D" indicates a data length as defined with BACON.DL and BACON.BYTE, that is, up to 32 bytes. One "D" is made of a number of 32-bit TXFIFO entries "d" and one rest at the end "b", which can be written as D=ddddddb.

The Long Data Mode can be extended by using the XXLCON register.

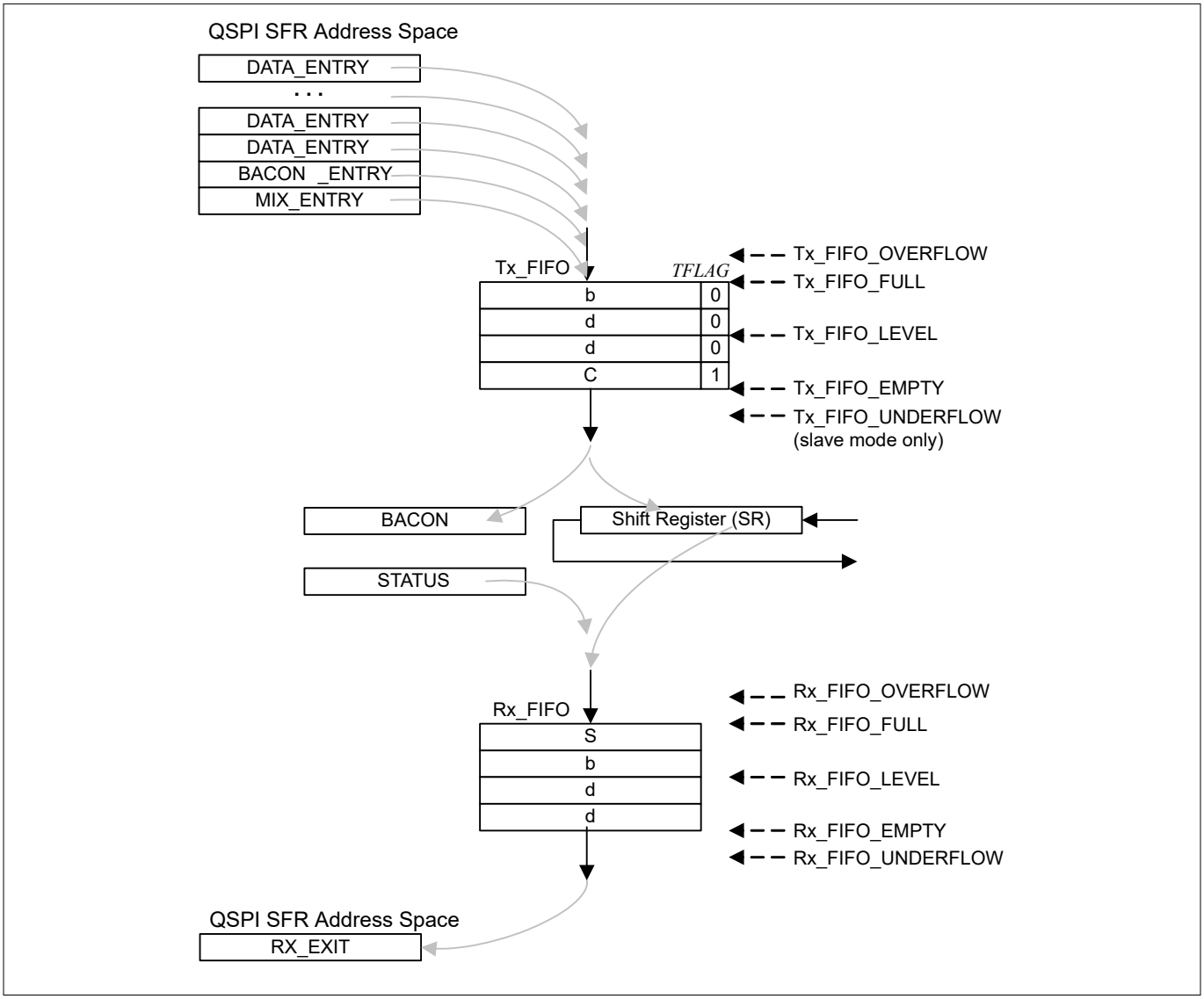


Figure 683 User interface in long data mode

In a DMA transfer example, a Tx_FIFO_EMPTY event can be used to trigger one DMA transfer (consisting of, for example, four 32-bit moves) to transfer the BACON and three DATA entries from RAM memory to the QSPI module. The three data words could contain 80 bits (10 bytes) of data. The three data words could also contain 40 bits (5 bytes), 64 bits (8 bytes) plus one dummy word. In this way short bursts of up to 96 bits can be transmitted.

In such 4 words DMA transfer example, the Tx_FIFO event can be used to transfer two 32-bit words per transfer.

If a configuration word is written when a data word is expected, the configuration word is ignored, and an Unexpected Configuration Error is raised.

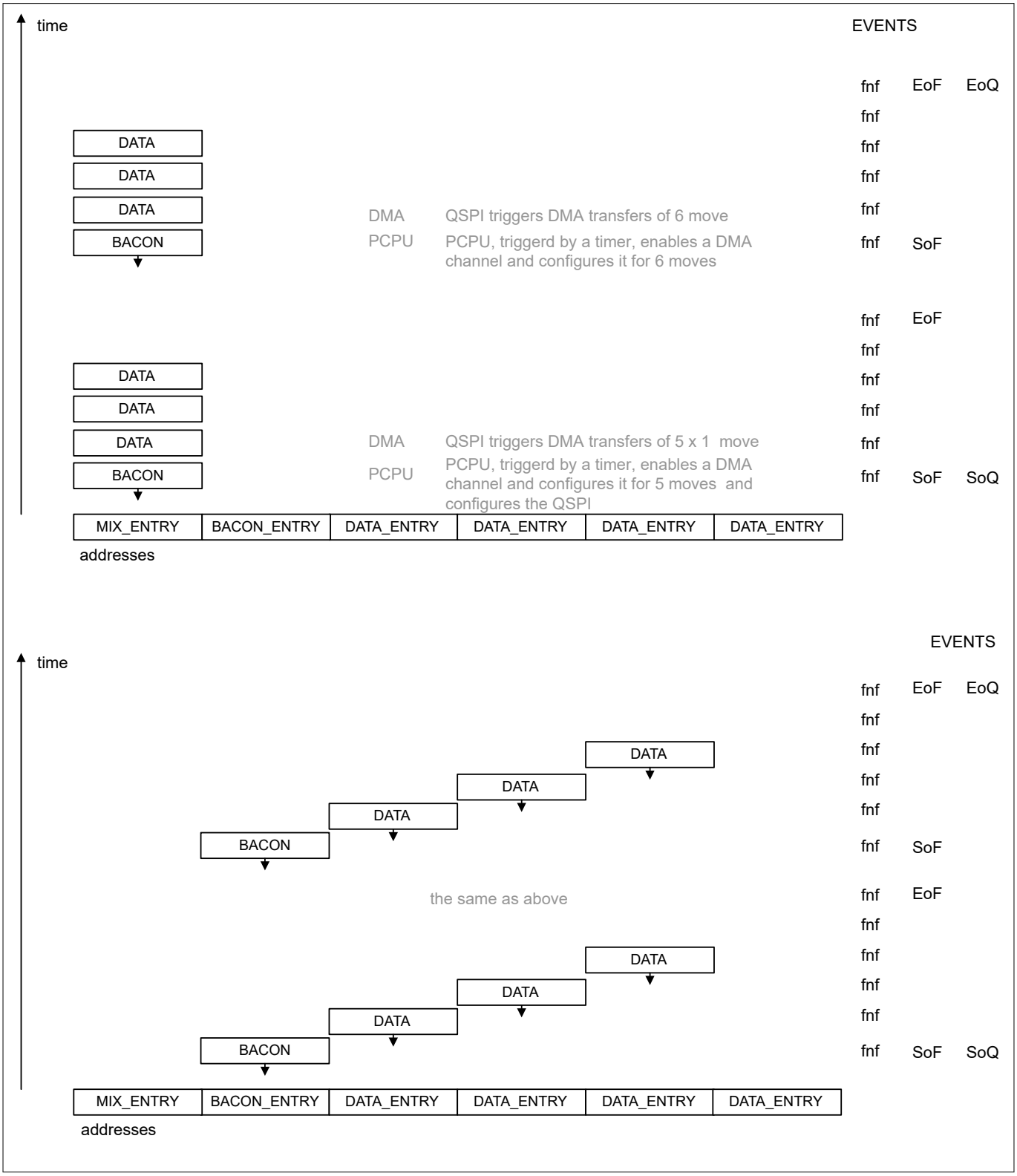


Figure 684 Diagram of the user interface in long data mode

30.3.5.1.3 Move counter mode

The Move Counter (MC) mode enables sending of multiple short frames by extending the short data mode. This mode is enabled when MCCON.MCEN=1 and BACON.BYTE = 0. Two different types are possible:

- BACON.LAST = 0: In this mode, the slave select signal SLSO remains active during the time of MC.MCOUNT write accesses (for example DMA moves) to the TXFIFO are being executed. After the frame has been completed, SLSO is automatically deactivated. After this completion, the software should clear the MCEN if the move counter is not needed anymore
- BACON.LAST = 1: In this mode, the slave select signal SLSO is deactivated after each frame transfer until all the MC.MCOUNT frames have been transmitted

The move counter is a down-counter that triggers an interrupt (if enabled):

- After the move before last has been shifted out
- After the last move has been shifted out

The range of counting is 65535 to 1. The bit field MC.CURRENT shows the currently transmitted frame, the last one being 1. The update of the MC.CURRENT bit field takes place with a couple of cycles delay due to the asynchronous frequency domain of the shift engine. The decrementing is done after shifting the last bit of each data. MC.CURRENT= 0 means no data pending, the complete data of the frame has been shifted out.

The enable, flag, set and clear bits for the Interrupt Before Last (IBL) and Interrupt After Last (IAL) are defined in MCCON register. The interrupt signals are mapped to the User Interrupt line, but have separate enable bits in the MCCON register and do not depend on BACON.UINT. Therefore on user interrupt in MC mode, the software must also check the MCCON.IALF and IBLF flags. All other standard QSPI phase interrupts are triggered as usual. If the data is not fed to the TXFIFO in time, EXPECT phase is activated.

An example sequence of using MC mode is: CPU writes the configuration for the frame once at the start in the BACON_ENTRY address, afterwards DMA writes DATA in DATA_ENTRY. MIX_ENTRY shall not be used. The TXFIFO interrupts trigger the DMA, which itself counts the moves in parallel to the MC register.

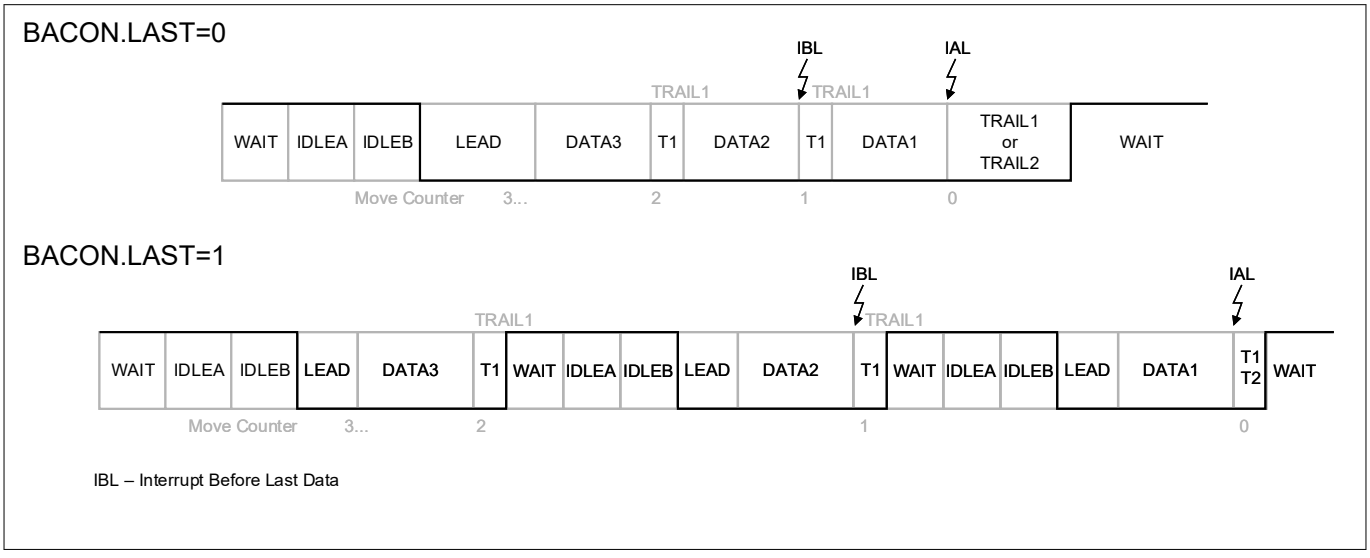


Figure 685 Frame sequence of phases in short move counter continuous mode

Another usage example of using MC mode is daisy-chaining: Here the CPU writes the number of frames for all slaves to the MC register. Afterwards it writes the configuration for the frame of the first slave (BACON3 in the figure) in the BACON_ENTRY address, afterwards DMA writes the data for the first slave (DATA3_2 and DATA3_1) in DATA_ENTRY. MIX_ENTRY shall not be used. The TXFIFO interrupts trigger the DMA, which until all data for the first slave has been written. Then the configuration of the second slave (BACON2) needs to be written to BACON_ENTRY, followed by its data (DATA2_3, DATA2_2 and DATA2_1). This process may be repeated until the whole daisy-chain is filled (in the figure this is an additional BACON1 and DATA1_1). The different BACONs may only vary in the BACON.DL and (or) BACON.MSB setting.

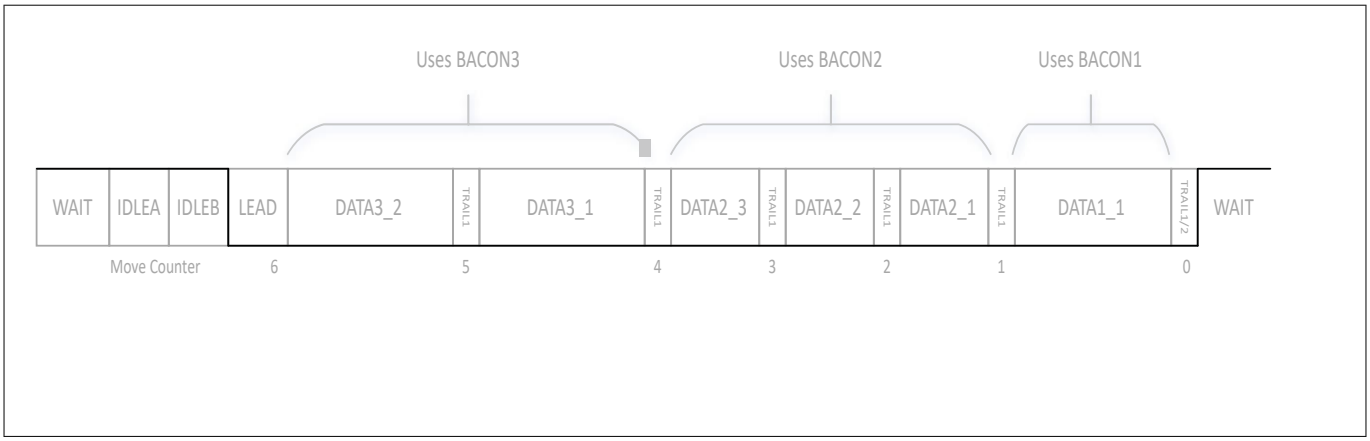


Figure 686 Frame sequence of phases in move counter implementing daisy-chaining

It is possible to combine Continuous Long Mode with Move Counter Mode so that a frame starts with one or more Continuous Long Mode transmissions and ends with one Move Counter Mode transmission.

30.3.5.1.4 Continuous data mode

In Continuous Data Mode, the QSPI transmits a stream of data with an arbitrary length and an active slave select signal. It can be used with both short and long data.

This mode starts by programming a control word containing BACON.LAST = 0 and the first data. The communication continues with the subsequent data entries written. The mode ends with a control word containing BACON.LAST = 1 and the last data. If LAST = 0, then TRAIL is a delay between data blocks. The W_I_L_D_T_D_T_D_T sequence appears. Each “D” represents data block as defined with BACON.DL and BACON.BYTE (up to 256 bits). If LAST = 1 then TRAIL is trailing delay, see related information.

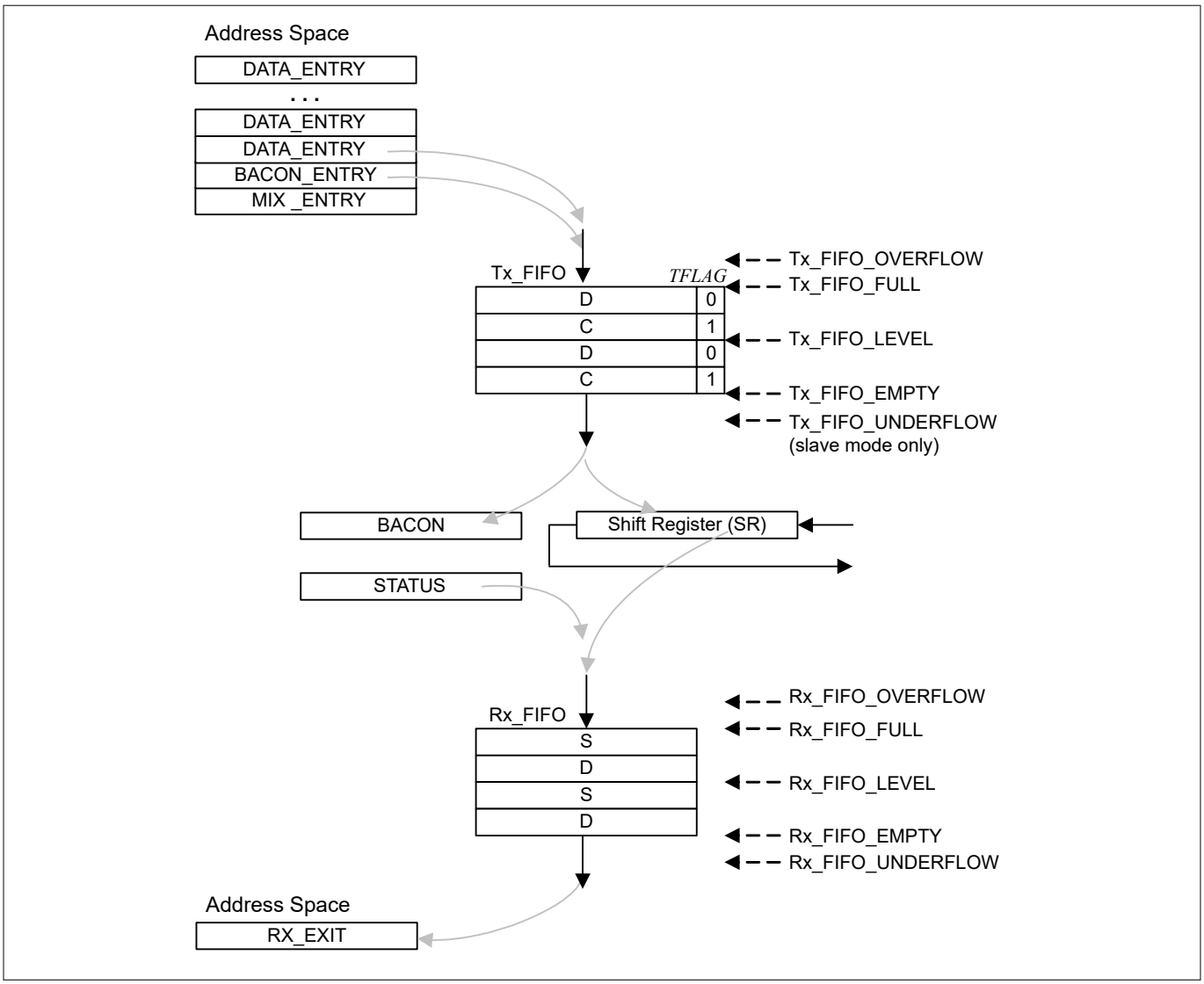


Figure 687 User interface in continuous data mode

The following figure shows an example of continuous data mode using short data. Here, the CPU starts the stream by writing BACON to the BACON_ENTRY address, and afterwards a DMA transfers the whole stream except the last BACON and DATA.

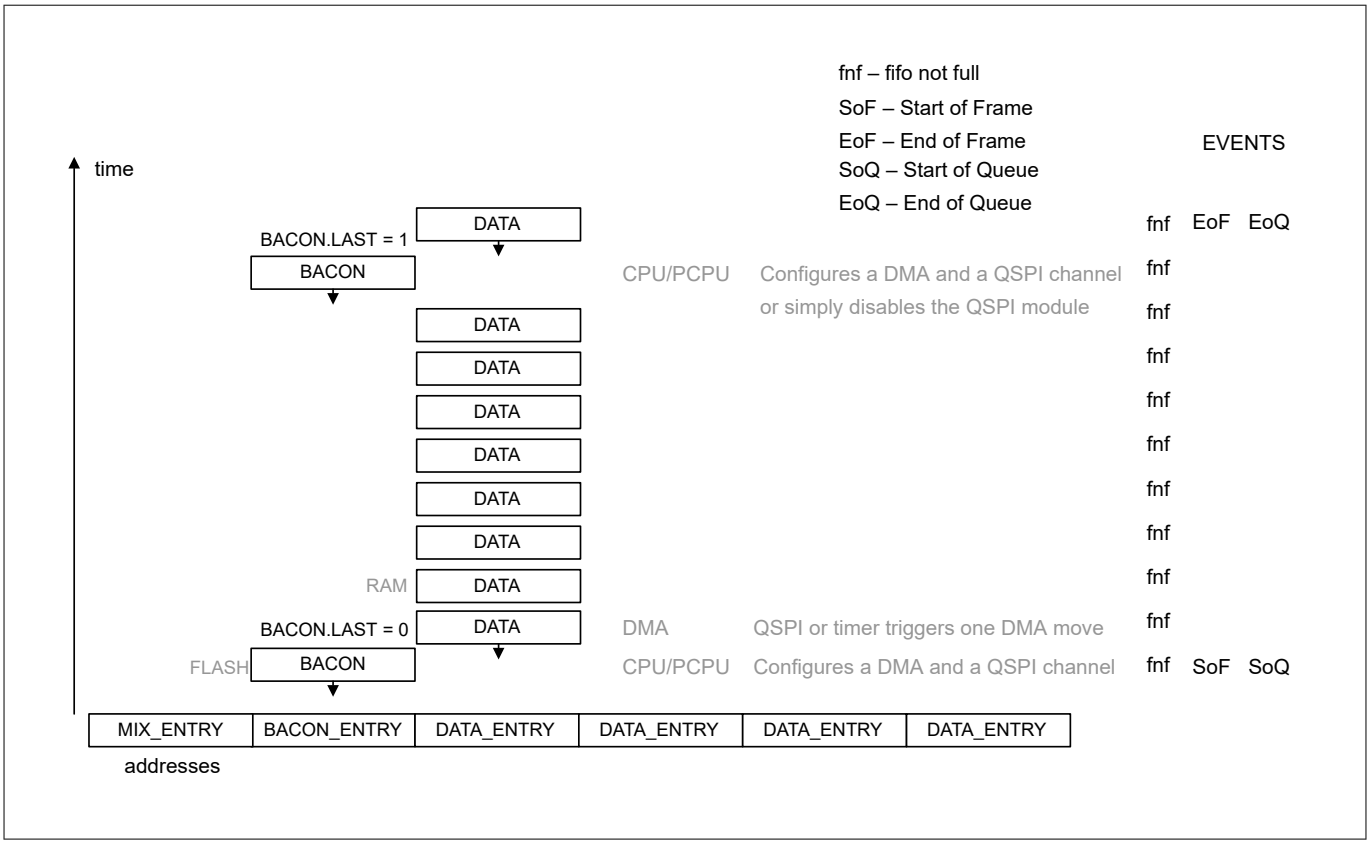


Figure 688 Use diagram of continuous data mode with short data

The following figure shows an example of continuous data mode using long data. Here, the first BACON must be repeated after each data block, but on the other hand, the whole transfer can be done using only the MIX entry and one single DMA channel, without CPU involvement. The loss of efficiency because of the unnecessary extra BACONs can be minimized by using long data blocks.

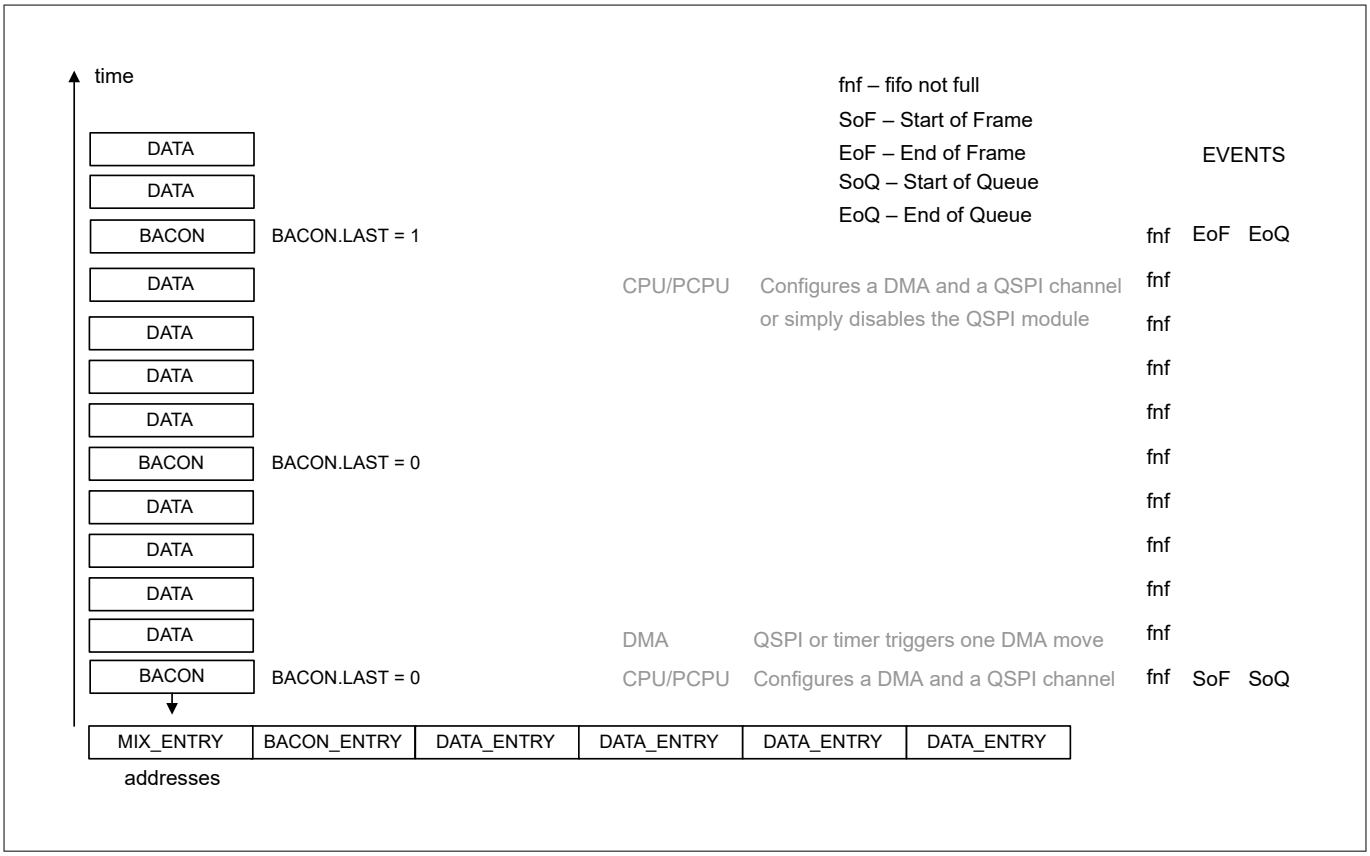


Figure 689 Use diagram of continuous cata mode with long data

Related information

Phases of one communication cycle on page 5304

30.3.5.1.5 Single configuration - multiple frames behavior

If longer sequences of data with length of up to 16-bit have to be sent to single slave in separate frames, then the DMA move for the repeating configuration word can be saved. An example of a TxFIFO contents and the resulting message sequence is shown in the following figure.

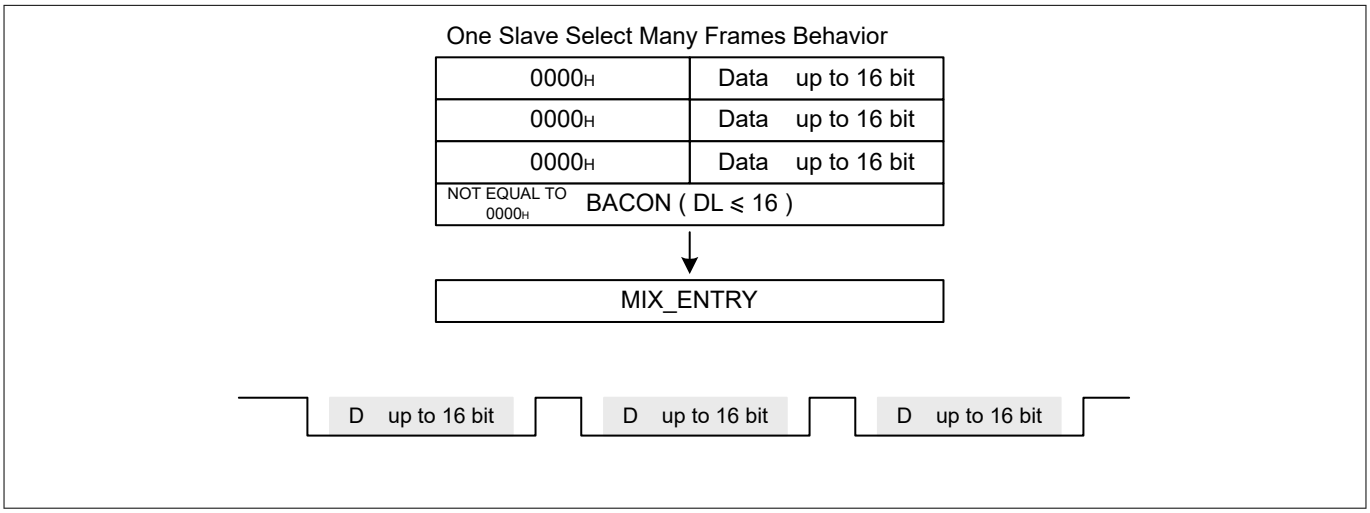


Figure 690 Multiple frames to one channel (example of up to 16-bit data)

In order to support sending short multiple frames to single channel without having to send the same configuration each time, the TXFIFO follows this rule:

- If BACON.DL is less or equal 15 (up to 16-bit data length)
- If MIX_ENTRY address is used
- Then if the upper 16-bit of a 32-bit MIX_ENTRY are 0, the entry is considered data, else configuration

  ***Note**: There is no valid configuration word with upper 16 bits equal to 0.*

- In this case, the TXFIFO marks the entry as data

***Note**: In case the BACON.LAST = 0, the slave select remains active between the frames. The behavior is equal to the continuous mode. The disadvantage of this method is that only up to 16 bits data can be transferred with one FPI bus move. The advantage is that continuous mode is possible using the mix entry.*

30.3.5.1.6 Big endian data format

The bit-field ECONz (z=0-7).BE activates the permutation of the data bytes written to the shift register and read from it to the big endian format and back.

ECONz (z=0-7).BE allows for switching between big and little endian per slave (SLSO signal).

See following figures.

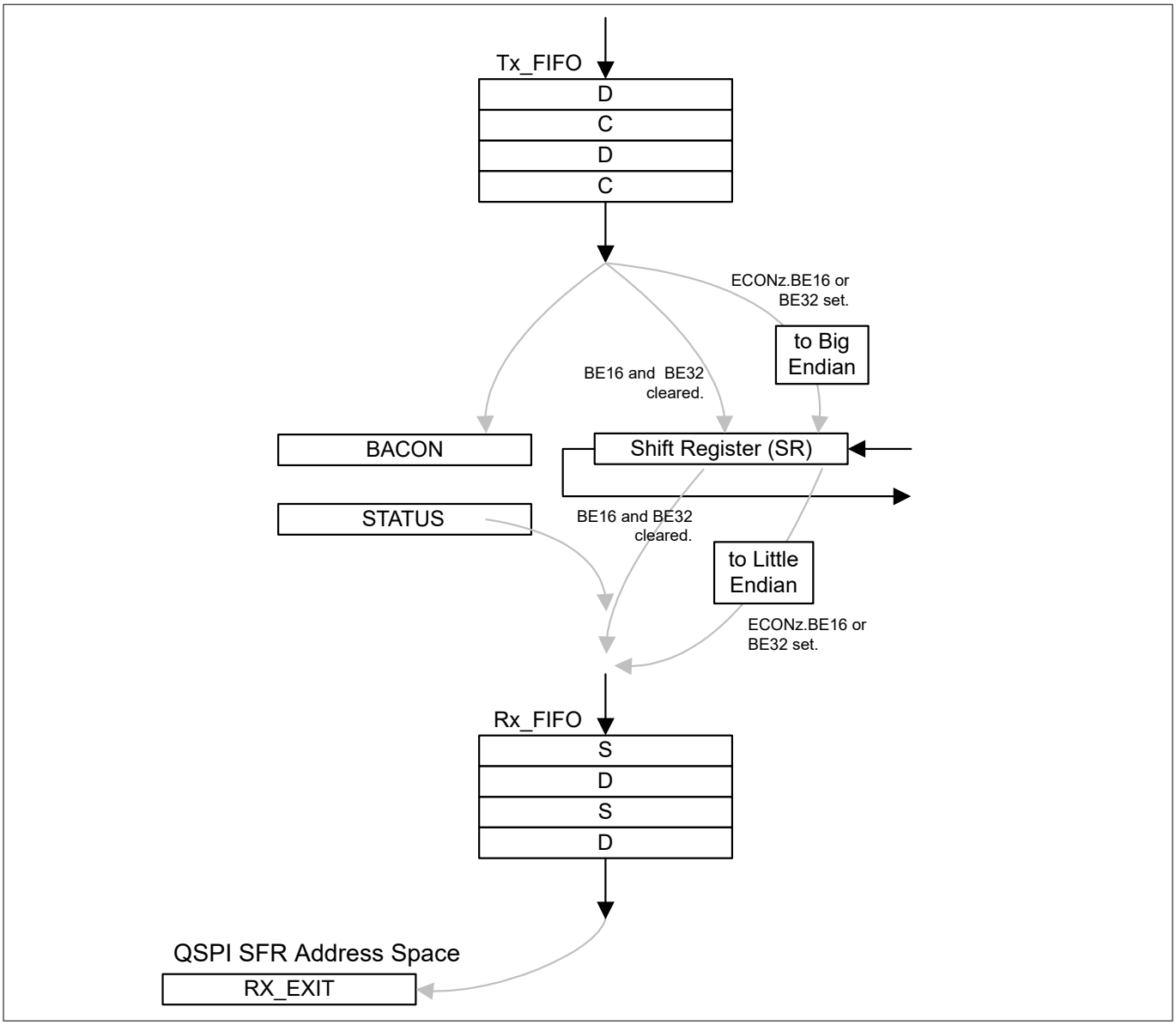


Figure 691 Big endian data path

Converting data from little to big endian and other way around requires one and the same permutation of the data bytes, as shown in the figure above, one permutation block is located before and one after the shift register, so that only data can be permutated, and not BACON or STATUS.

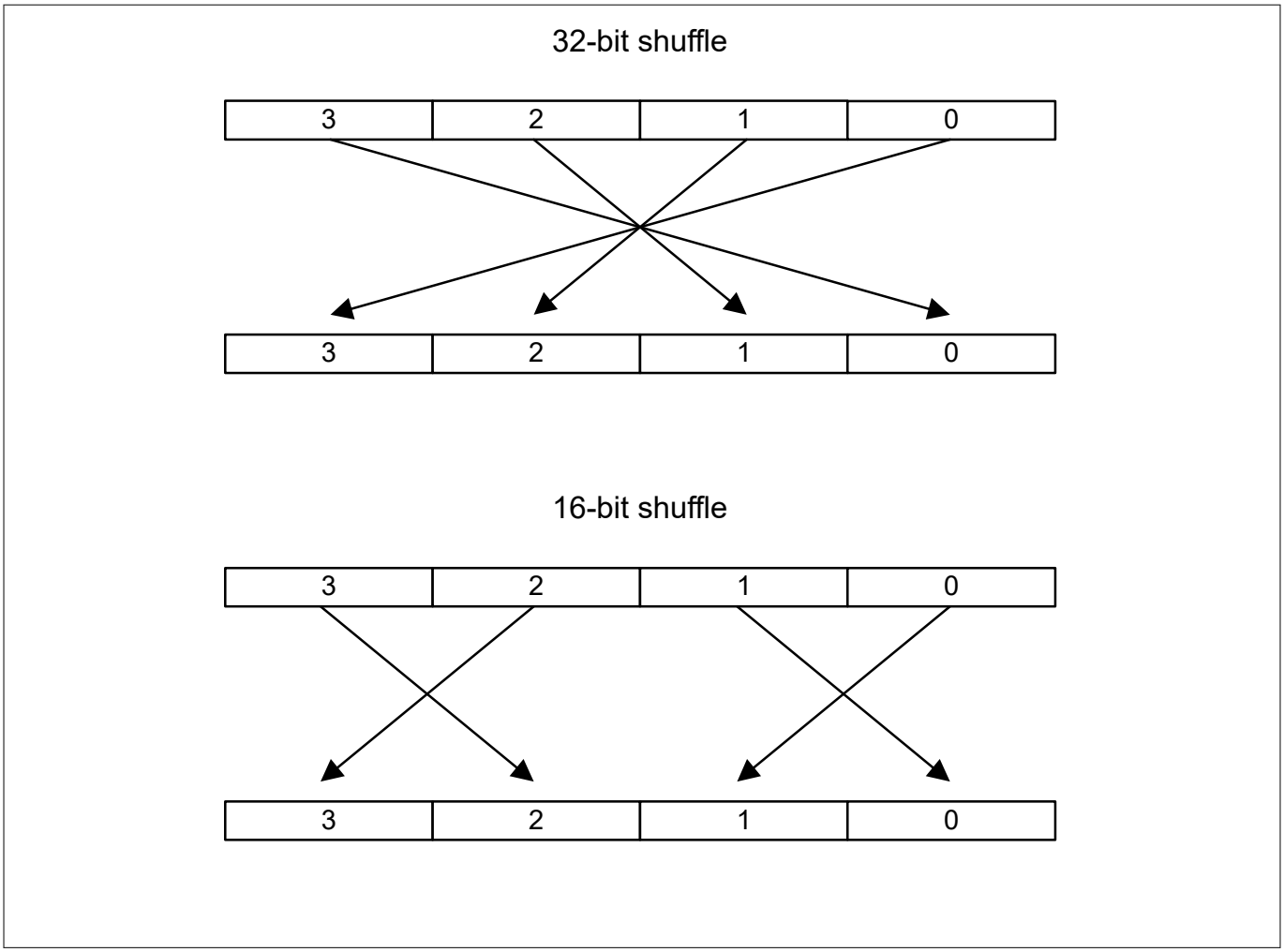


Figure 692 Endianness byte permutation

The endianness byte permutation should be used only for 16-bit and 32-bit data (and multiples of it), and not for other data lengths.

30.3.5.2 Loop-back mode

Loop-Back mode is a master mode test feature used for establishing a module internal connection between the transmit and the receive signal, the same as if the transmit and receive pins would be short connected. It is activated by setting the GLOBALCON.LB bit.

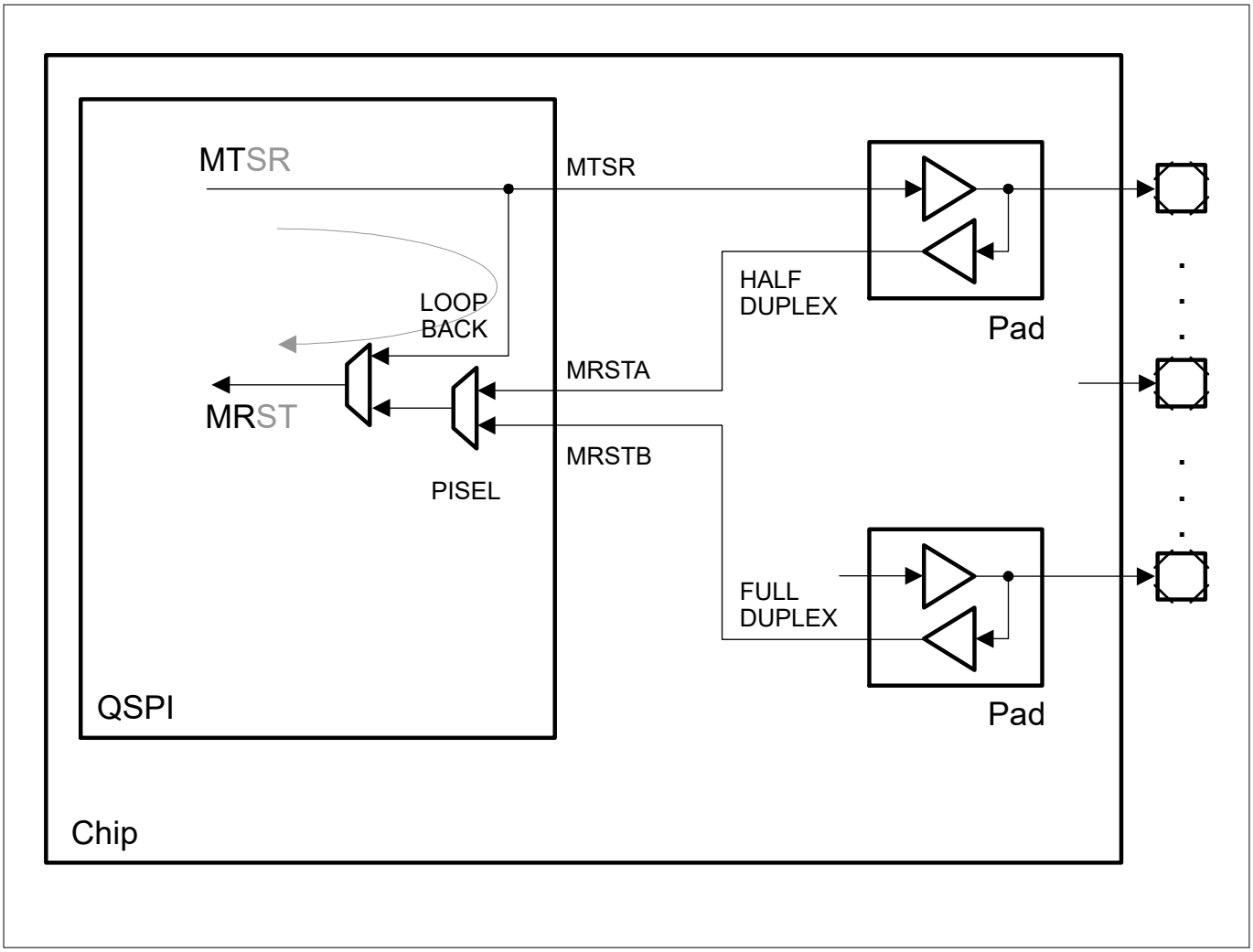


Figure 693 Loop-back mode

30.3.6 Interrupts

There are several types of interrupts:

- Interrupts related to the user interface
- FIFO related interrupts
- Interrupts related to the shift engine (state machine)
  - Error interrupts
  - Phase transition interrupts

According to this classification, here follows the complete list of all the interrupts:

- FIFO related interrupts
  - TX Transmit FIFO Interrupt, requests feeding of the TXFIFO, but does not signal the error events of overflow and underflow (see figure Architecture of the Tx and Rx FIFOs, related information)
  - RX Receive FIFO Interrupt, requests emptying of the RxFIFO, but does not signal the error events of overflow and underflow (see figure Architecture of the Tx and Rx FIFOs, related information)
- ERR Error Interrupt
  - Signals every error condition (see GLOBALCON1.ERRORENS)
  - Signals TX_FIFO_OVERFLOW, TX_FIFO_UNDERFLOW
  - Signals RX_FIFO_OVERFLOW, RX_FIFO_UNDERFLOW
- PT Phase transition interrupt, signaling two out of all phase transitions in the QSPI communication cycle (see figures Standard Communication Cycle of the State Machine (Short Data) and EXPECT Cycles of the QSPI State Machine, related information)
  - PT1 event
  - PT2 event
- U User Interrupt
  - Triggered by the PT1 event during the time BACON.UINT bit keeps this event enabled
- MC Move Counter Mode Interrupts
  - Interrupt before last-triggered when move counter was decremented to 1
  - Interrupt after last-triggered when move counter was decremented to 0

The Transmit event (TX_INT) and Receive event (RX_INT) are also connected to GST, if present in device, for time capture purpose.

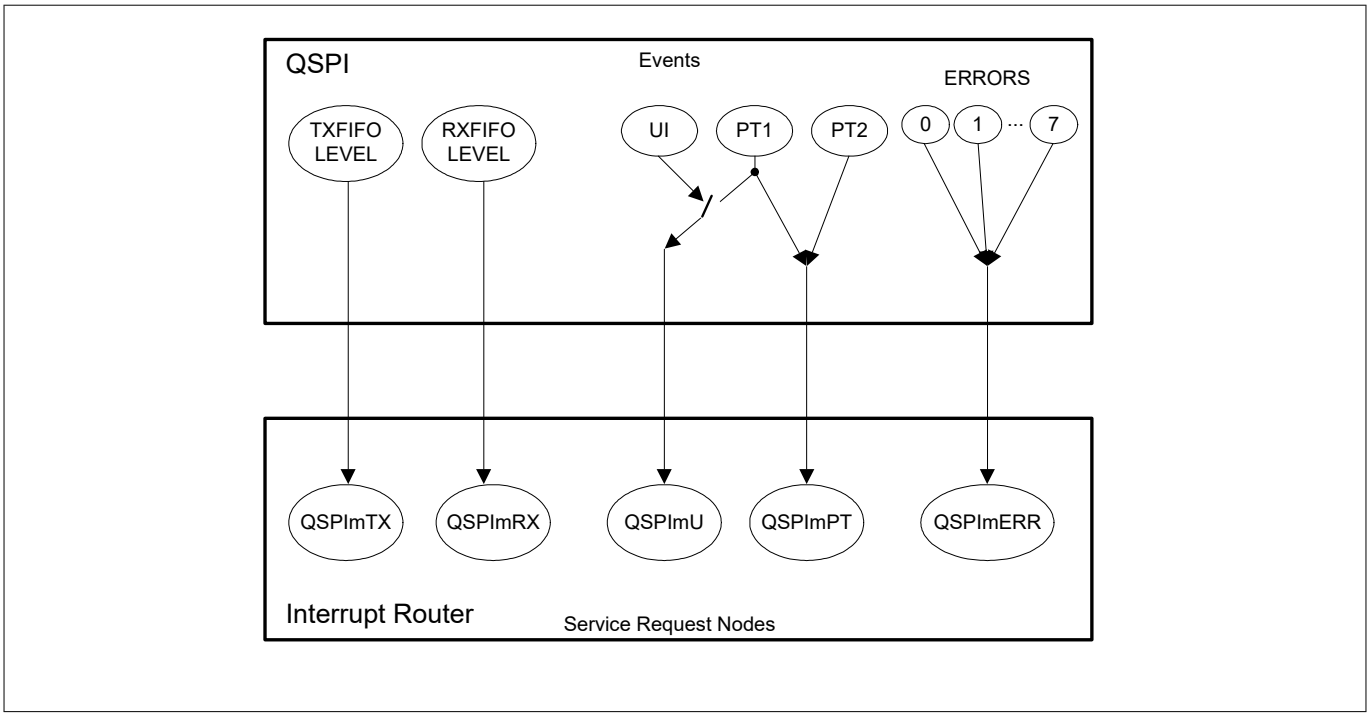


Figure 694 Events and interrupts overview

Related information

Transmit and receive FIFOs on page 5335
State diagram of standard communication cycle on page 5314
Expect phase on page 5317

30.3.6.1 Slave mode SLSI interrupt

In slave mode, deactivating the SLSI signal (SLSI rising edge) triggers the PT interrupt, if PT2 enabled.

30.3.6.2 Interrupt flags behavior

A state of an event flag (low or high) does not influence the flow of its event signal. If an event occurs recurrently, and the event flag remains set (if not cleared by the software interrupt handler), then in that case the interrupt flag (which is cleared by hardware, by an interrupt control unit (ICU), when the interrupt wins an arbitration round) would receive recurrent triggers, and the interrupt flag would be constantly set all the time.

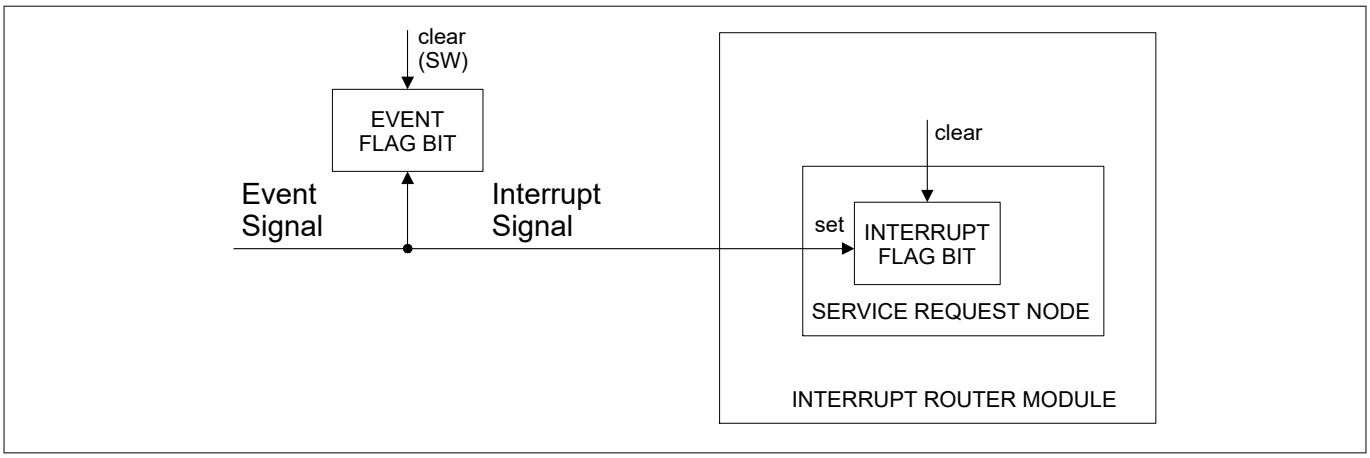


Figure 695 Events and interrupts overview

30.3.6.3 TXFIFO interrupt generation

The TXFIFO provides four interrupt generation modes, selected by the bit-field GLOBALCON1.TXFM:

- Single Move Mode
- Batch Move Mode-1
- Batch Move Mode-2
- Combined Mode

The RXFIFO and the TXFIFO can use the interrupt generation modes independently of each other.

Single move mode

The purpose of the Single Move Mode is to keep the TXFIFO as full as possible, refilling the TXFIFO by writing to it as soon as there is a free element.

An example use case could be using single move mode with a DMA performing single move per TXFIFO interrupt. In this mode the DMA keeps the TXFIFO full. A DMA request is triggered each time a write to the TXFIFO is performed, and there is at least one empty element available. If the write makes the TXFIFO full (no more empty elements), an interrupt is not generated in order to prevent overflow. It is generated later when the shift register (or BACON) reads one element from the TXFIFO, making it not full.

The following figure shows the filling levels of the TXFIFO, and the events associated with each filling level triggering a TXFIFO refill interrupt.

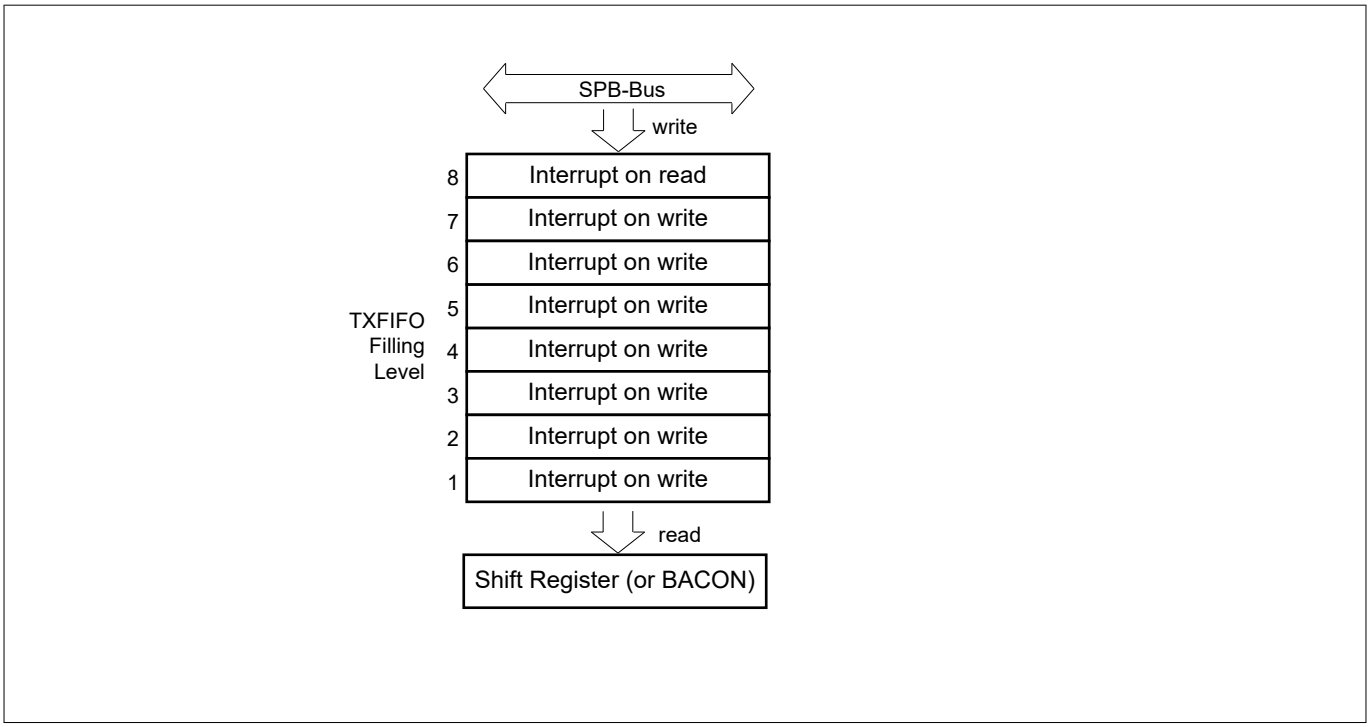


Figure 696 Interrupt generation in the single move mode

In order to initiate the very first chain of the refill interrupts after power-on reset, it is necessary that the software either performs one write to the TXFIFO, or sets the interrupt flag in the TXFIFO interrupt node. Afterwards, the (DMA) interrupt-chain is self sustaining until the whole transaction is over.

Attention**: **In Single Move Mode multiple software writes or block DMA moves would lead to multiple interrupts and (false) transaction lost events. Therefore they should be avoided - only single moves should be used.

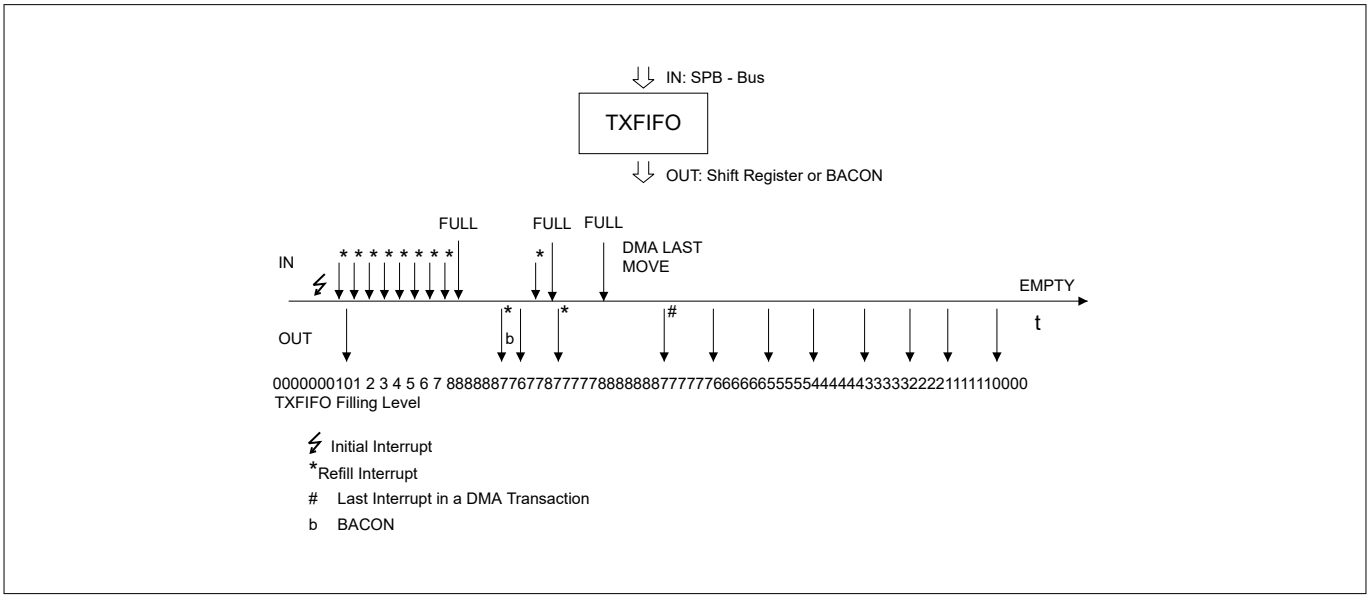


Figure 697 TXFIFO - interrupt operation in single move mode

Batch move mode-1

The purpose of this mode is to reduce the number of interrupts by triggering an interrupt when more than one TXFIFO element is empty. For example, a CPU can be interrupted less frequently and it can perform more than one moves per interrupt.

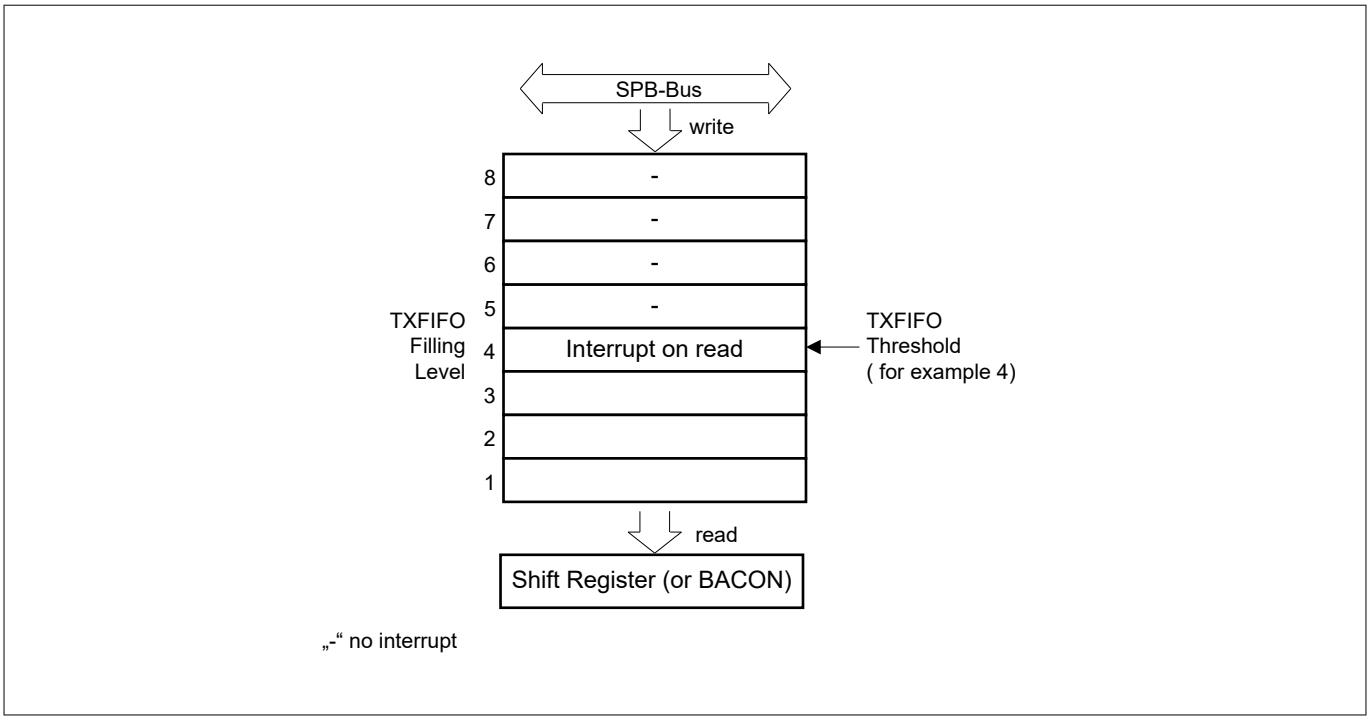


Figure 698 TXFIFO - interrupt generation in the batch move mode-1

In this mode, an interrupt is generated only at one point in the TXFIFO, when the filling level falls below the programmed threshold, implying that there are at least the predefined number of empty TXFIFO elements available.

***Attention**: The TXFIFO must be refilled until the filling level has risen above the interrupt threshold (ensured by polling the filling level), in order to guarantee that the next interrupt will occur.*

At high baud rates and short frames, it could be possible that a couple of frames have been transmitted until the moment the TXFIFO is refilled. For example, if the threshold is set to three full elements, then the interrupt will be set when the filling level falls below three, making two elements free. If the refill moment of the TXFIFO is delayed so long that two frames have been transmitted (or frame and BACON), the TXFIFO will become empty, and refill sequence of two will not reach the level of three. So, the next interrupt would never come unless the filling level is being polled and the CPU keeps refilling in a loop until the threshold has been passed (ideally the TXFIFO would be filled completely with each batch of moves).

Batch move mode-2

Batch Move Mode-2 is a combination of the Batch Move Mode-1 and Single Move Mode in a way such that, an interrupt is generated not only when the filling level falls below the programmed threshold but also on each write to the TXFIFO when the filling level is less than the programmed threshold.

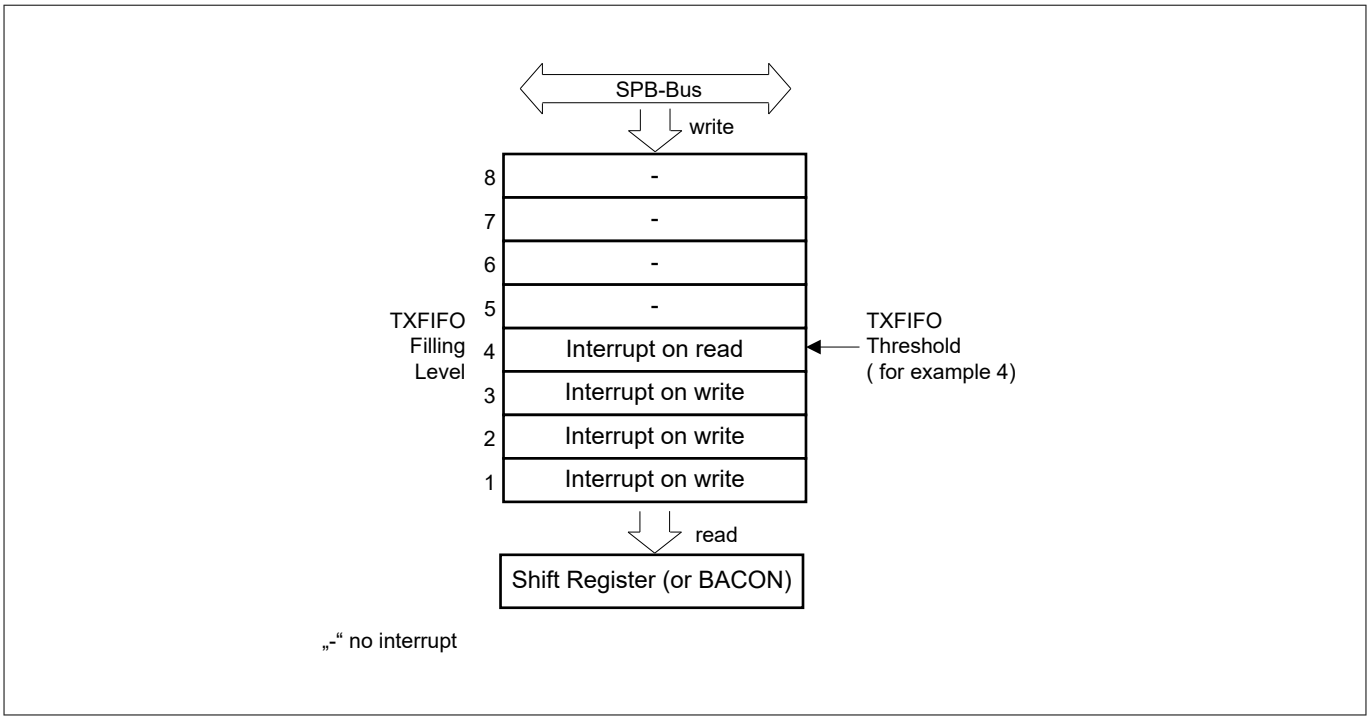


Figure 699 TXFIFO - interrupt generation in the batch move mode-2

Combined mode

TXFIFO generates interrupt each time a data is fetched from it below the pre-programmed threshold, as defined in the GLOBALCON1.TXFIFOINT. Consequently, it generates two close interrupts when delivering BACON and the first following data. If the BACON TX interrupt is not serviced until the DATA1 TX interrupt comes, which will always be the case due to their time proximity, the BACON TX interrupt will be lost. Therefore, it is recommended that a DMA should always perform two moves per interrupt trigger (transfer size of two). Otherwise, the TXFIFO average filling level will go down with the time and at the end remain oscillating between zero and one (except in continuous mode). The DMA transaction loss event that may be raised by the DMA should be ignored.

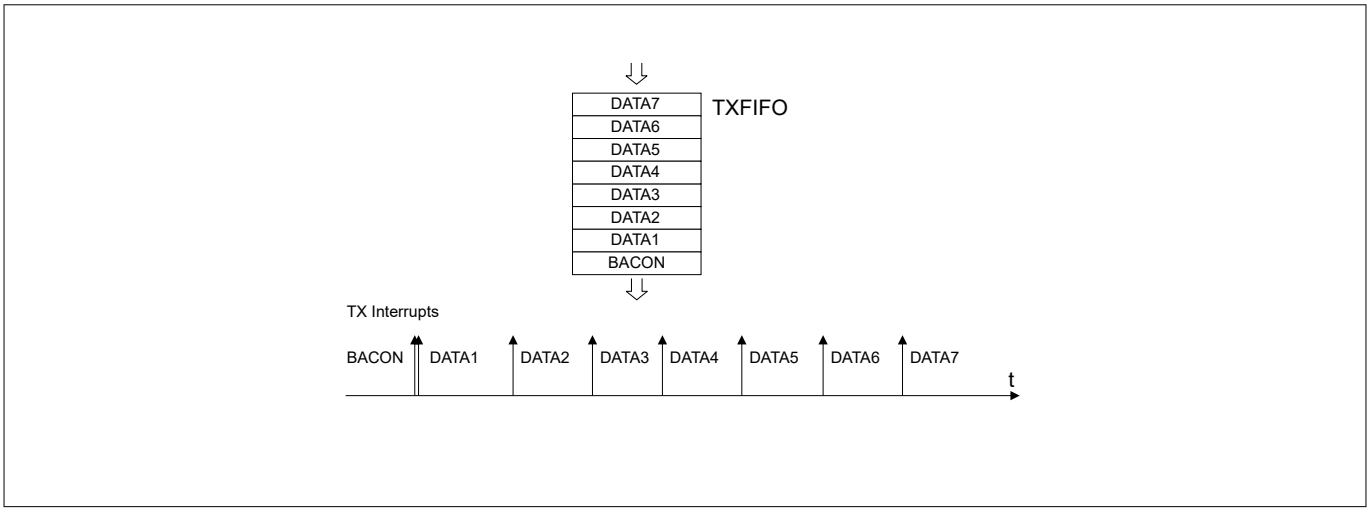


Figure 700 Events and interrupts overview

30.3.6.4 RXFIFO interrupt generation

The RXFIFO provides four interrupt generation modes, selected by the bit-field GLOBALCON1.RXFM:

- Single move mode
- Batch move mode-1
- Batch Move Mode-2
- Combined mode

Single move mode

The purpose of the Single Move Mode is to keep the RXFIFO as empty as possible, by fetching the received elements one by one as soon as possible.

An example use case could be using single move mode with a DMA performing single move per RXFIFO interrupt. In this mode the DMA keeps the RXFIFO as empty as possible. A DMA request is triggered each time a read from the RXFIFO is performed, and the RXFIFO is afterwards still not empty. If the read makes the RXFIFO empty, an interrupt is not generated in order to prevent underflow. It is generated later when the shift register (or STATUS) writes new element to the empty RXFIFO.

The following figure shows the filling levels of the RXFIFO, and the events associated with each filling level triggering a RXFIFO refill interrupt.

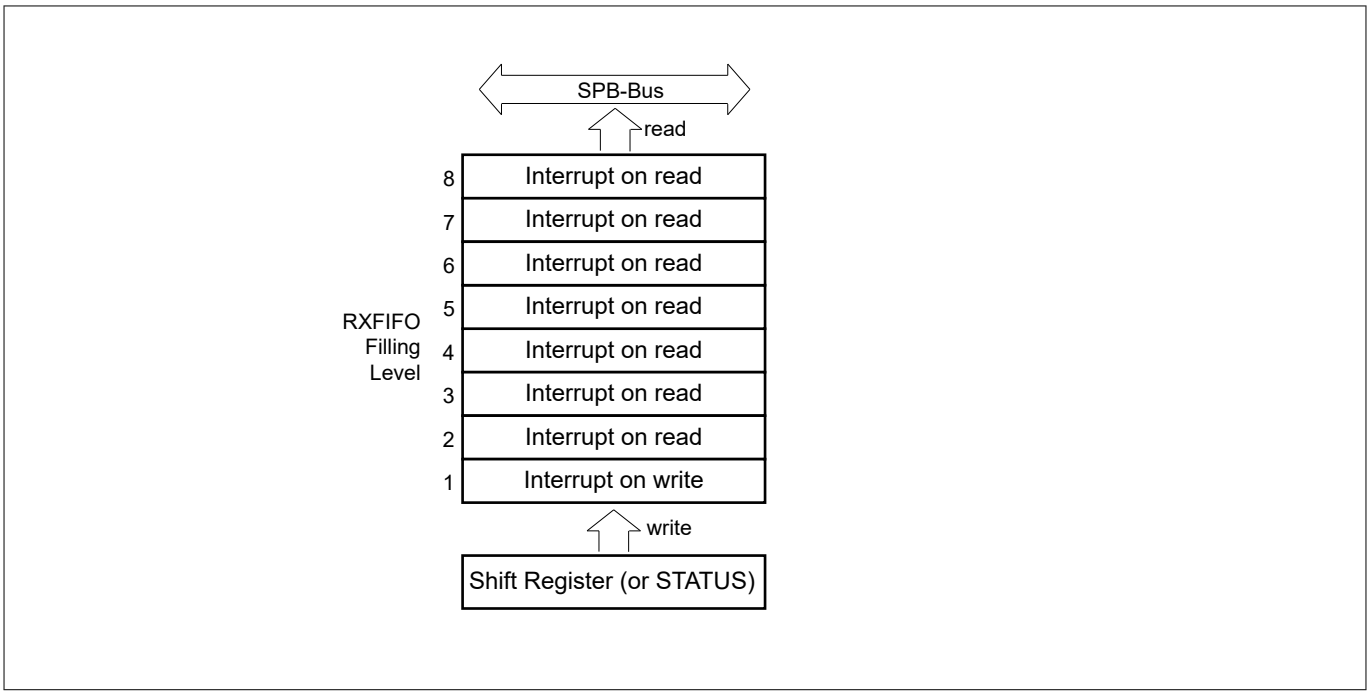


Figure 701 RXFIFO - interrupt triggering in the single move mode

The initial RXFIFO interrupt is triggered by the Shift Register, after it delivers the first received element. Afterwards, the DMA trigger-chain of re-fetch interrupts is self sustaining until the whole transaction is over. At the end of the DMA transaction, there is no service request remaining active in the service request node, due to the fact that the read of the last element in the RXFIFO does not trigger an interrupt.

Attention**: **In Single Move Mode multiple software reads or block DMA moves lead to multiple interrupts and (false) transaction lost events. Therefore they should be avoided - only single moves should be used.

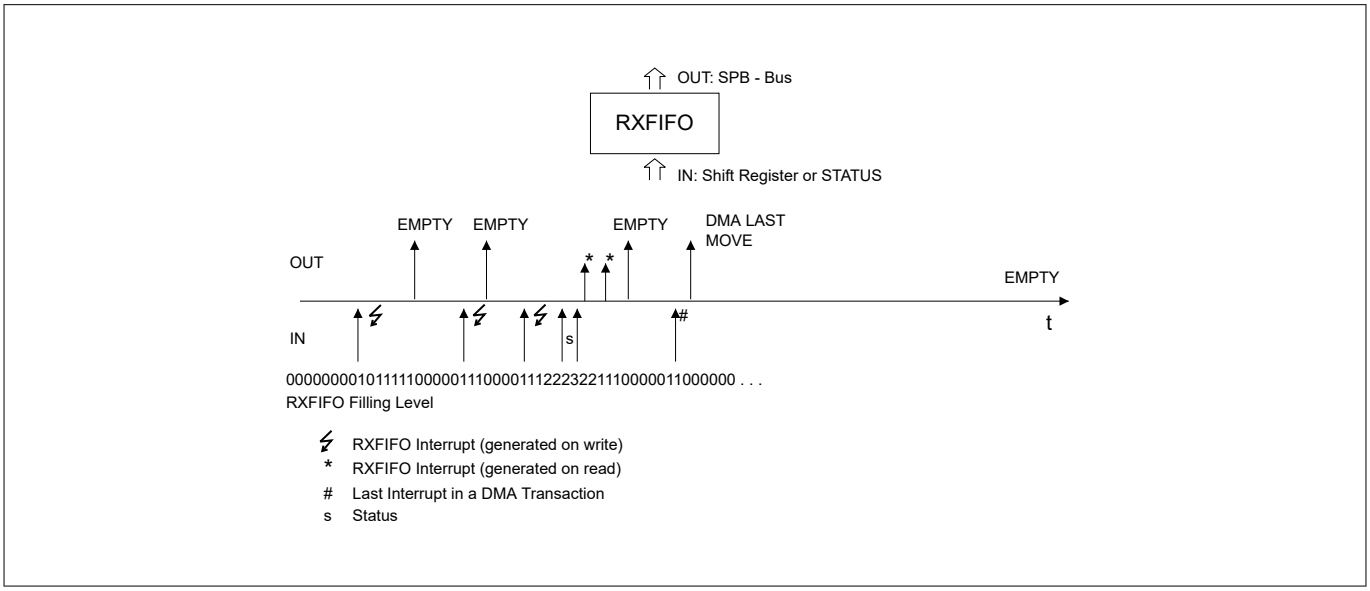


Figure 702 RXFIFO - interrupt operation in single move mode

Batch move mode-1

The purpose of this mode is to reduce the number of interrupts by triggering an interrupt when more than one RXFIFO element is full. For example, a CPU can be interrupted less frequently and it can perform more than one move per interrupt.

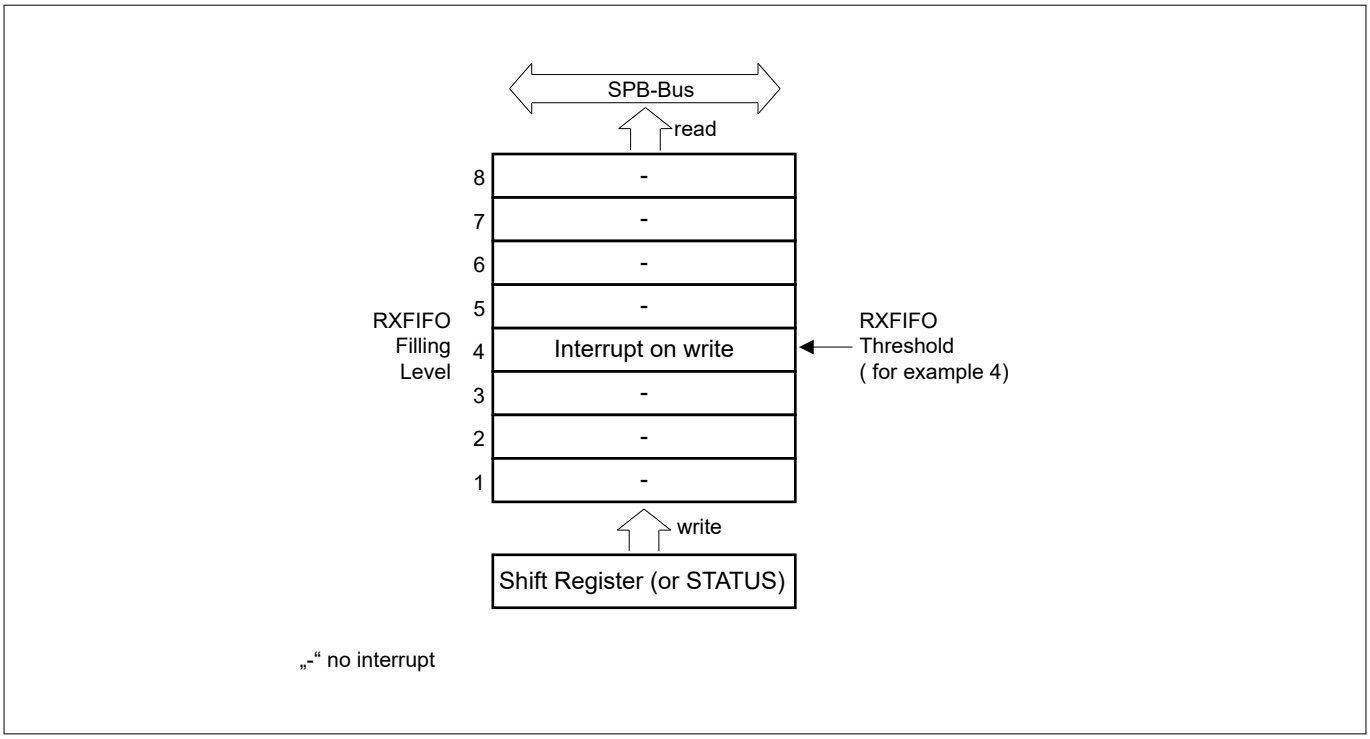


Figure 703 RXFIFO - interrupt triggering in the batch move mode-1

In Batch Move Mode-1, an interrupt is generated only at one point in the RXFIFO, when the filling level rises above the programmed threshold, implying that there are at least the predefined number of full RXFIFO elements available.

Attention*: *It must be guaranteed that the CPU keeps emptying the RXFIFO and polling the RXFIFO filling level until the filling level has fallen below the interrupt threshold (or the RXFIFO is empty), in order to guarantee that next interrupt will occur.

At high baud rates and short frames, it could be possible that more than one frame has been received until the moment the RXFIFO is re-emptied. For example, if the threshold is set to two full elements, then the interrupt will be set when the filling level rises above two. If the emptying moment of the RXFIFO is delayed so long that two additional frames have been received (or a frame and STATUS), the RXFIFO will become full, and re-empty sequence of two will not reach the level of two. So, the next interrupt would never come. This effect cannot occur with threshold levels of 3 and 4. The threshold level of 1 is possible, but does not make much sense (use single move mode instead).

Batch move mode-2

Batch Move Mode-2 is a combination of the Batch Move Mode-1 and Single Move Mode in a way such that, an interrupt is generated not only when the filling level rises above the programmed threshold but also on each read from the RXFIFO when the filling level is higher than the programmed threshold.

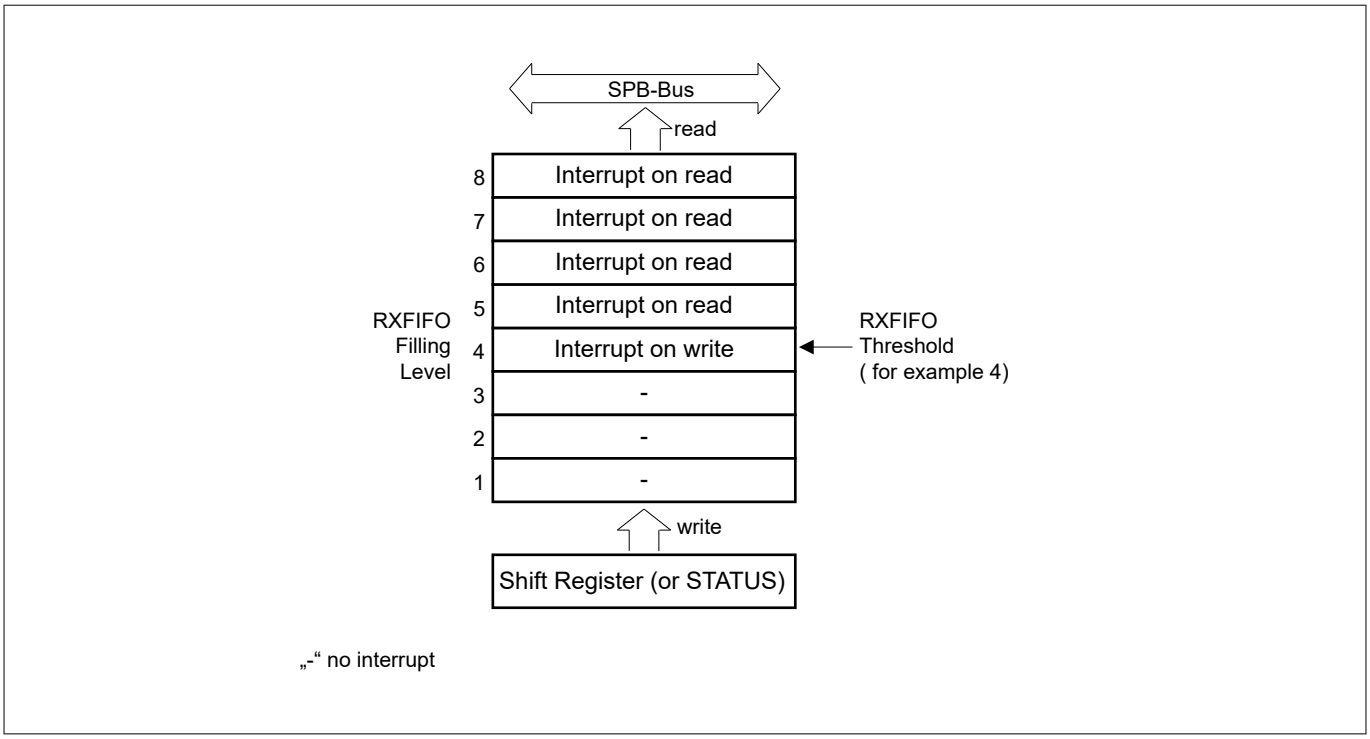


Figure 704 TXFIFO - interrupt generation in the batch move mode-2

Combined mode

RXFIFO generates interrupt each time a data is written to it above the preprogrammed level, as defined in the GLOBALCON1.RXFIFOINT. Consequently, it generates two close interrupts when filled with data and STATUS. If the data RX interrupt is not serviced until the STATUS RX interrupt comes, which will always be the case due to their time proximity, the STATUS RX interrupt will be lost. In such a case, it is recommended that the DMA performs two moves per interrupt trigger (transfer size of two). Otherwise, the RXFIFO average filling level will go up with the time and at the end overflow.

30.3.7 Reset behavior

There are two sources of reset provided by the module for it’s operation:

- KRST bit-field of registers RST_CTRLA/B
- Reset bits in the register GLOBALCON.RESETS

The KRST bit-field puts the whole QSPI module in reset state. All outputs get the reset value.

Software writing the GLOBALCON.RESETS bit will select what will be Reset (TX_FIFO, RX_FIFO, protocol statemachine, registers).

KRSTx resets the whole Module while GLOBALCON.RESETS=0b11 resets only a subset of all the available registers in the module.

30.4 Registers

30.4.1 Register overview - access mode glossary

Table 1238 Register overview - access mode glossary

Keyword	Description
E	Access protection using PROT register PROTE.
SE	Access protection using PROT register PROTSE.
APU-P	Protection group consisting of registers ACCEN_WRA, ACCEN_WRB, ACCEN_RDA, ACCEN_RDB, ACCEN_VM, ACCEN_PRS.
P	Access protection using *APU-P* registers.
BE	Always returns a Bus Error.
U	No access restrictions.
PROT	Access restrictions as defined in the PROT register access rules.

30.4.2 Registers overview - QSPI (ascending offset address)

Table 1239 Registers overview - QSPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CLC	Clock control register	000_H	P	P, SV, E	Application Reset	5364
OCS	OCDS control and status register	004_H	P	SV, P	Debug Reset	5364
ID	Module identification register	008_H	P	BE	PowerOn Reset	5365
RST_CTRLA	Reset control register A	$00C_H$	P	P, SV, E	Application Reset	5366
RST_CTRLB	Reset control register B	010_H	P	P, SV, E	Application Reset	5366
RST_STAT	Reset status register	014_H	P	BE	Application Reset	5367
PROTE	PROT register endinit	020_H	U	SV, PROT	Application Reset	5368
PROTSE	PROT register safe endinit	024_H	U	SV, PROT	Application Reset	5369
ACCEN_WRA	Write access enable register A	040_H	U	SE, SV	Application Reset	5371
ACCEN_WRB	Write access enable register B	044_H	U	SE, SV	Application Reset	5372
ACCEN_RDA	Read access enable register A	048_H	U	SE, SV	Application Reset	5372

(table continues...)

Table 1239 (continued) Registers overview - QSPI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ACCEN_RDB	Read access enable register B	$04C_H$	U	SE, SV	Application Reset	5373
ACCEN_VM	VM access enable register	050_H	U	SE, SV	Application Reset	5373
ACCEN_PRS	PRS access enable register	054_H	U	SE, SV	Application Reset	5374
PISEL	Port input select register	104_H	P	SV, P	Kernel Reset	5374
GLOBALCON	Global configuration register	110_H	P	SV, P	Kernel Reset	5376
GLOBALCON1	Global configuration register 1	114_H	P	SV, P	Kernel Reset	5379
BACON	Basic configuration register	118_H	P	BE	Kernel Reset	5383
ECONz	Configuration extension z	120_H+z* 4	P	SV, P	Kernel Reset	5385
STATUS	Status register	140_H	P	P	Kernel Reset	5387
STATUS1	Status register 1	144_H	P	P	Kernel Reset	5390
SSOC	Slave select output control register	148_H	P	SV, P	Kernel Reset	5391
FLAGSCLEAR	Flags clear register	150_H	P	P	Kernel Reset	5392
XXLCON	Extra large data configuration register	154_H	P	P	Kernel Reset	5393
MIXENTRY	MIX_ENTRY register	158_H	P	P	Kernel Reset	5394
BACONENTRY	BACON_ENTRY register	$15C_H$	P	P	Kernel Reset	5394
DATAENTRYx	DATA_ENTRY register x	160_H+x* 4	P	P	Kernel Reset	5397
RXEXIT	RX_EXIT register	180_H	P	BE	Kernel Reset	5397
RXEXIT# (#=0-7)	RX_EXIT register mirror #	180_H+#* 4	P	BE	Kernel Reset	5397
RXEXITD	RX_EXIT debug register	$1A0_H$	P	BE	Kernel Reset	5398
MC	Move counter register	$1B4_H$	P	P	Kernel Reset	5398
MCCON	Move counter control register	$1B8_H$	P	P	Kernel Reset	5399
MSSMUX	Multiple slave select register	$1BC_H$	P	P	Kernel Reset	5400

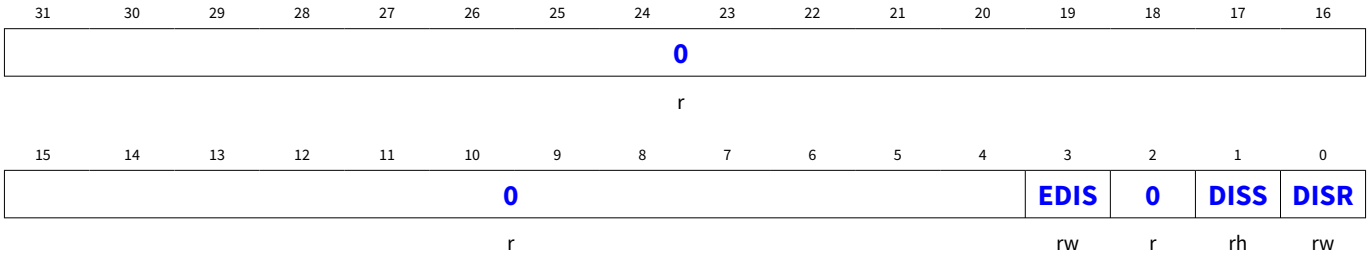
30.4.3 Clock control register

The clock control register CLC allows the programmer to adapt the functionality and power consumption of the module to the requirements of the application.

Register CLC controls the module clock signal and the reactivity to the sleep signal.

CLC Offset address: 000_H

Clock control register Application Reset value: $0000\ 0003_H$



Field	Bits	Type	Description
DISR	0	rw	**Module disable request bit** Used for enable and disable control of the module. 0_B On request: enable the module clock 1_B Off request: stop the module clock
DISS	1	rh	**Module disable status bit** 0_B Module clock is enabled 1_B Module is not clocked
EDIS	3	rw	**Sleep mode enable control** Used to control the module's reaction to sleep mode. 0_B Sleep mode request is enabled and functional 1_B Module disregards the sleep mode control signal
0	2, 31:4	r	**Reserved** Read as 0; should be written with 0.

30.4.4 OCDS control and status register

The OCDS control and status register OCS controls the debug behavior by selecting suspend modes. When OCDS is disabled the suspend control is ineffective.

OCS Offset address: 004_H

OCDS control and status register Debug Reset value: $0000\ 0000_H$

Field	Bits	Type	Description
SUS	27:24	rw	**OCDS suspend control** Controls the sensitivity to the suspend signal coming from the OCDS trigger switch (OTGS) Not listed combinations are reserved. 0_H Will not suspend 1_H Hard suspend Clocks will be disabled immediately 2_H Reserved 3_H Soft suspend mode
SUS_P	28	w	**SUS write protection** SUS is only written when SUS_P is 1, otherwise unchanged. Read as 0.
SUSSTA	29	rh	**Suspend state** 0_B Module is not (yet) suspended 1_B Module is suspended
0	1:0, 2, 3, 23:4, 31:30	r	**Reserved** Read as 0; should be written with 0.

Table 1240 Access mode restrictions of OCS sorted by descending priority

Mode name	Access mode		Description
write 1 to .**SUS_P**	rw	SUS	Set SUS_P during write access
(default)	r	SUS	

30.4.5 Module identification register

ID Offset address: 008_H

Module identification register PowerOn Reset value: $00C0\ C006_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MOD_NUM															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MOD_TYPE								MOD_REV							
r								r							

Field	Bits	Type	Description
MOD_REV	7:0	r	**Module revision** Indicates the revision number of the implementation.

(table continues...)

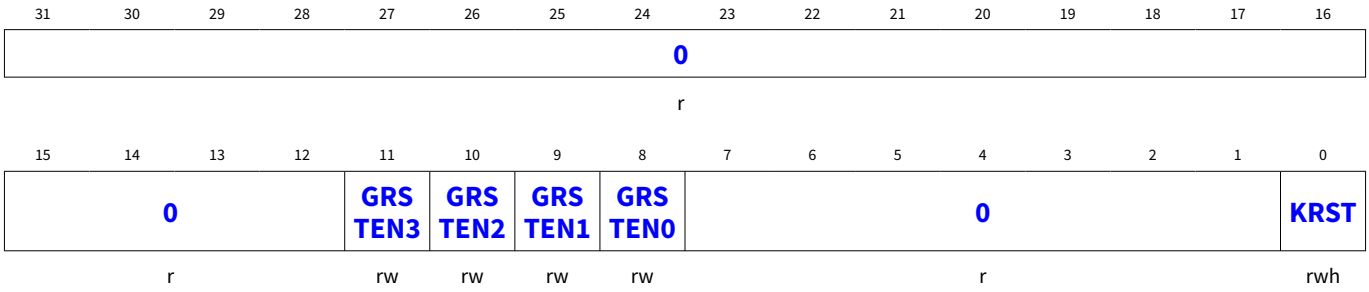
(continued)

Field	Bits	Type	Description
MOD_TYPE	15:8	r	**Module type** This internal marker is fixed to $C0_H$.
MOD_NUM	31:16	r	**Module number** Indicates the module identification number.

30.4.6 Reset control register A

RST_CTRLA Offset address: $00C_H$

Reset control register A Application Reset value: $0000\ 0000_H$

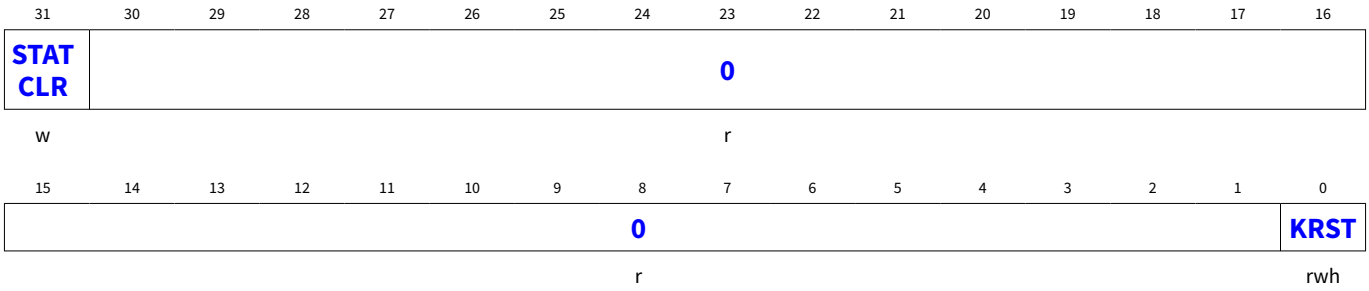


Field	Bits	Type	Description
KRST	0	rwh	**Kernel reset** Request a kernel reset. The requested reset is executed if CTRLA.KRST=1 and CTRLB.KRST=1. KRST is cleared after the kernel reset was executed. 0_B No action 1_B A kernel reset was requested
GRSTENx (x=0-3)	x+8	rw	**Enable for global module reset group x** 0_B Global module reset group x does not have any effect 1_B Global module reset group x results in a kernel reset
0	7:1, 31:12	r	**Reserved** Read as 0; should be written with 0.

30.4.7 Reset control register B

RST_CTRLB Offset address: 010_H

Reset control register B Application Reset value: $0000\ 0000_H$



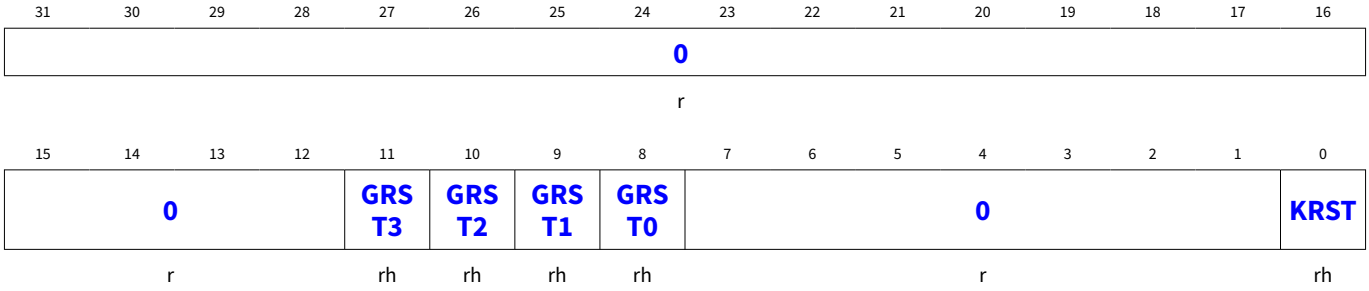
Field	Bits	Type	Description
KRST	0	rwh	**Kernel reset** Request a kernel reset. The requested reset is executed if CTRLA.KRST=1 and CTRLB.KRST=1. KRST is cleared after the kernel reset was executed. 0_B No action 1_B A kernel reset was requested
STATCLR	31	w	**Kernel reset status clear** Clears all status bits in RST_STAT when 1 is written. Read will return 0. 0_B No action 1_B Write with ´1´ clears bits STAT.GRSTx and bit STAT.KRST .
0	30:1	r	**Reserved** Read as 0; should be written with 0.

30.4.8 Reset status register

The reset status register contains the status bits for kernel reset (KRST) and the global module reset groups (GRSTx).

RST_STAT Offset address: 014_H

Reset status register Application Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
KRST	0	rh	**Kernel reset status** Indicates an executed kernel reset. RST_STAT.KRST is set after the execution of a kernel reset in the same clock cycle in which the reset bits are cleared. Clear KRST by setting bit STATCLR in register RST_CTRLB.

(table continues...)

(continued)

Field	Bits	Type	Description
GRSTx (x=0-3)	x+8	rh	**Status for global module reset group x** 0_B Reset was not triggered by global reset group x 1_B Reset was triggered by global reset group x
0	7:1, 31:12	r	**Reserved** Read as 0.

30.4.9 PROT register endinit

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner and in some PROT states, the secure master to update the PROT state as described in the PROT mechanism.

The PROTE register controls lock and unlock of the local endinit (E) protected control registers.

PROTE Offset address: 020_H

PROT register endinit Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OWEN	ODEF	TAGID						PRSEN	PRS			VMEN	VM		
w	rw	rw						rw	rw			rw	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												SWEN	STATE		
r												w	rwh		

Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
VM	18:16	rw	**Virtual machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.

(table continues...)

(continued)

Field	Bits	Type	Description
VMEN	19	rw	**Virtual machine definition enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.
PRSEN	23	rw	**Protection set definition enable for PROT owner** 0_B **Disable**: PRS is not part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; should be written with 0.

Table 1241 Access mode restrictions of PROTE sorted by descending priority

Mode name	Access mode		Description
write 1 to .**SWEN**	rwh	STATE	
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

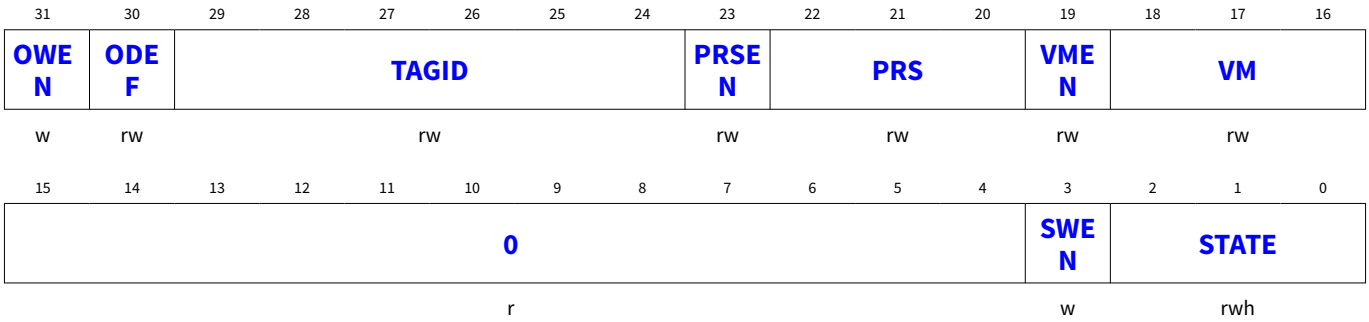
30.4.10 PROT register safe endinit

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner and in some PROT states, the secure master to update the PROT state as described in the PROT mechanism.

The PROTSE register controls lock and unlock of the local safe endinit (SE) protected control registers.

PROTSE Offset address: 024_H

PROT register safe endinit Application Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
VM	18:16	rw	**Virtual machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual machine definition enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.
PRSEN	23	rw	**Protection set definition enable for PROT owner** 0_B **Disable**: PRS is not part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner

(table continues...)

(continued)

Field	Bits	Type	Description
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; should be written with 0.

Table 1242 Access mode restrictions of PROTSE sorted by descending priority

Mode name	Access mode		Description
write 1 to .**SWEN**	rwh	STATE	
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

30.4.11 Write access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

ACCEN_WRA Offset address: 040_H

Write access enable register A Application Reset value: $1000\ 0003_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Write access enable for TAG ID q** This bit enables write access to the access protected region for transactions with the master TAG ID q. 0_B Disabled for write access 1_B Enabled for write access

30.4.12 Write access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

ACCEN_WRB Offset address: 044_H

Write access enable register B Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Write access enable for TAG ID q** This bit enables write access to the access protected region for transactions with the master TAG ID q. 0_B Disabled for write access 1_B Enabled for write access
0	31:8	r	**Reserved** Read as 0; should be written with 0.

30.4.13 Read access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

ACCEN_RDA Offset address: 048_H

Read access enable register A Application Reset value: $FFFF\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Read access enable for TAG ID q** This bit enables read access to the access protected region for transactions with the master TAG ID q. 0_B Disabled for read access 1_B Enabled for read access

30.4.14 Read access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

ACCEN_RDB Offset address: $04C_H$

Read access enable register B Application Reset value: $0000\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Read access enable for TAG ID q** This bit enables read access to the access protected region for transactions with the master TAG ID q. 0_B Disabled for read access 1_B Enabled for read access
0	31:8	r	**Reserved** Read as 0; should be written with 0.

30.4.15 VM access enable register

This register defines which virtual machine encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each VMx.

ACCEN_VM Offset address: 050_H

VM access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for VM ID q** This bit enables read access to the access protected region for transactions with the VM ID q. 0_B Disabled for read access 1_B Enabled for read access

(table continues...)

(continued)

Field	Bits	Type	Description
WRq (q=00-07)	q+16	rw	**Write access enable for VM ID q** This bit enables write access to the access protected region for transactions with the VM ID q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

30.4.16 PRS access enable register

This register defines which protection register set encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each PRSx.

ACCEN_PRS Offset address: 054_H

PRS access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

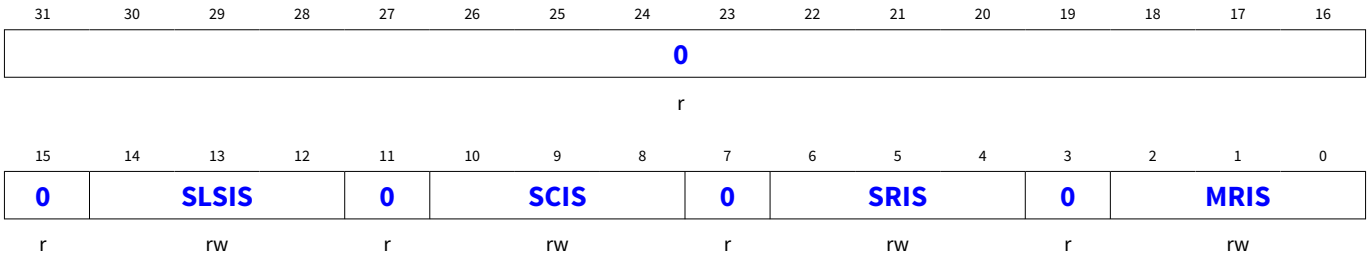
Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for PRS q** This bit enables read access to the access protected region for transactions with the PRS q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for PRS q** This bit enables write access to the access protected region for transactions with the PRS q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

30.4.17 Port input select register

The PISEL register controls the input signal selection of the QSPI module.

PISEL Offset address: 104_H

Port input select register Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
MRIS	2:0	rw	**Master Mode Receive Input Select** MRIS selects one out of eight MRST receive input lines, used in Master Mode. Note that not all inputs are used in every device of the family. Selecting an unused input returns a continuous low value. 000_B MRST input line A is selected for operation 001_B MRST input line B is selected for operation 010_B MRST input line C is selected for operation 011_B MRST input line D is selected for operation 100_B MRST input line E is selected for operation 101_B MRST input line F is selected for operation 110_B MRST input line G is selected for operation 111_B MRST input line H is selected for operation
SRIS	6:4	rw	**Slave Mode Receive Input Select** SRIS selects one out of eight MTSR receive input lines, used in Slave Mode. Note that not all inputs are used in every device of the family. Selecting an unused input returns a continuous low value. 000_B MTSR input line A is selected for operation 001_B MTSR input line B is selected for operation 010_B MTSR input line C is selected for operation 011_B MTSR input line D is selected for operation 100_B MTSR input line E is selected for operation 101_B MTSR input line F is selected for operation 110_B MTSR input line G is selected for operation 111_B MTSR input line H is selected for operation
SCIS	10:8	rw	**Slave Mode Clock Input Select** SCIS selects one out of eight module kernel SCLK input lines that is used as clock input line in slave mode. Note that not all inputs are used in every device of the family. Selecting an unused input returns a continuous low value. 000_B SCLK input line A is selected for operation 001_B SCLK input line B is selected for operation 010_B SCLK input line C is selected for operation 011_B SCLK input line D is selected for operation 100_B SCLK input line E is selected for operation 101_B SCLK input line F is selected for operation 110_B SCLK input line G is selected for operation 111_B SCLK input line H is selected for operation

(table continues...)

(continued)

Field	Bits	Type	Description
SLSIS	14:12	rw	**Slave Mode Slave Select Input Selection** The SLSIS must be programmed properly before the slave mode is set with GLOBALCON.MODE and the module is set to RUN mode. 000_B Slave select input lines are deselected QSPI is operating without slave select input functionality. 001_B SLSI input line A is selected for operation 010_B SLSI input line B is selected for operation 011_B SLSI input line C is selected for operation 100_B SLSI input line D is selected for operation 101_B SLSI input line E is selected for operation 110_B SLSI input line F is selected for operation 111_B SLSI input line G is selected for operation
0	3, 7, 11, 31:15	r	**Reserved** Read as 0; should be written with 0.

30.4.18 Global configuration register

GLOBALCON contains configuration parameters which affect all channels.

***Note**: If the EN bit is cleared first, and then the partial state machine reset activated, then the state machine goes into PAUSE state.*

***Note**: Switching the peripheral clock off and on is done by using two writes: first a write of zero to the CLKSEL bit field, and then a second write switching on the clock source. Between the first and the second write a delay of minimum 4 x (1/f_{PER}) + 2 x (1/f_{CLC}) must be inserted by software, where f_{PER} is the frequency being switched off with the first write. This also applies if between the writes the clock frequency is changed in the CCU.*

GLOBALCON Offset address: 110_H

Global configuration register Kernel Reset value: $000F\ 30FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESETS		CLKSEL	HWTREN	AREN	MS		EN	HS	STIP	SRF	STROBE				
w		rw	rw	rw	rw		rwh	rw	rw	rw	rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DEL0	LB	EXPECT				SI	0	TQ							
rw	rw	rw				rw	r	rw							

Field	Bits	Type	Description
TQ	7:0	rw	**Global Time Quantum Length** Common n-divider scaling the baud rates of all channels in direction of higher or lower baud rates. Must not be changed during a running transaction. 00_H division by 1 01_H division by 2 … FF_H division by 256
SI	9	rw	**Status Injection** Selects if the status register content injection into the RxFIFO is enabled or disabled. The status injections, if enabled, is performed after each data block, depending on the BACON.DL and BACON.BYTE setting. 0_B Disabled 1_B Enabled
EXPECT	13:10	rw	**Time-Out Value for the Expect Phase** expressed in T_{QSPI} units 0_H 64 (2^6) 1_H 128 (2^7) … F_H 2097152 (2^21)
LB	14	rw	**Loop-Back Control** Selects if the transmit output is internally connected to the receive input for test purposes. For detailed description, see the Loop-Back Mode section. 0_B Loop-Back inactive 1_B Loop-Back active
DEL0	15	rw	**Delayed Mode** Switches the delayed mode (external slave select expansion mode) on and off 0_B Delayed mode off 1_B Delayed mode on
STROBE	20:16	rw	**Strobe Delay in Delayed Mode** Defines the length of the delay of the strobe signal for the external multiplexer in T_Q time units as defined for channel z (T_Q units) selected by the current BACON.CS, if GLOBALCON.DEL0 = 1. 00_H 1 01_H 2 … $1F_H$ 32

(table continues...)

(continued)

Field	Bits	Type	Description
SRF	21	rw	**Stop on RxFIFO Full** If this bit is set, the data fetching out of the TxFIFO by the shift register stops when the RxFIFO is full, in order to prevent RxFIFO overflow. Master mode only. 0_B Feature disabled 1_B *Feature enabled*
STIP	22	rw	**Slave Transmit Idle State Polarity** This bit determines the logic level of the Slave Mode transmit signal MRST when the QSPI slave select input signals are inactive (PISEL.SLSIS != 000_B). 0_B MRST = 0 when QSPI is deselected in Slave Mode. 1_B MRST = 1 when QSPI is deselected in Slave Mode
HS	23	rw	**High Speed Mode** Enables high speed QSPI mode. 0_B Feature disabled 1_B Feature enabled
EN	24	rwh	**Enable Bit** Used to request transition between PAUSE and RUN mode per software. Cleared by hardware automatically at leaving the following states: disabled, suspend and sleep. In order to determine if the requested state has actually been reached, the STATUS.PHASE bit field should be polled. See also Operation Modes. 0_B PAUSE requested 1_B RUN requested
MS	26:25	rw	**Master Slave Mode** Selects if the module operates in master or slave mode. This bit field must be configured before the first write to the TXFIFO. 00_B Master Transmit and Receive 01_B Reserved 10_B Slave Transmit and Receive 11_B Slave Transmit and Receive
AREN	27	rw	**Automatic Reset Enable** Enables the reset of the GLOBALCON.EN on baud rate and spike error in slave mode. 0_B Disabled 1_B Enabled
HWTREN	28	rw	**Hardware Trigger Enable** If this bit is set, the data fetching out of the TXFIFO by the shift register starts on the reception of a hardware trigger signal. 0_B Feature disabled 1_B *Feature enabled*

(table continues...)

(continued)

Field	Bits	Type	Description
CLKSEL	29	rw	**Clock Select** Selects the clock source for the asynchronous block. 0_B no clock 1_B f_{PER}
RESETS	31:30	w	**Bits for resetting sub-modules per software** Read as 0. Write to this bit field triggers a reset operation. The duration of reset operation depends on the used clock setups. It is 10 times of the slowest clock between SPB and Kernel. ***Note**: CLOCKSEL shall be enabled to use GLOBALCON.RESETS. After the Reset the CLOCKSEL will be also cleared in case of 10 11.* For resetting the whole module kernel, use alternatively the registers KRST reset mechanism. 00_B No reset triggered 01_B State Machine, TXFIFO and RXFIFO reset, registers not reseted. Only control structures shall be reset. It avoids a re-configuration of the whole IP. After this reset Communication can be restarted. 10_B Registers reset. Clears the configuration of the QSPI registers. New setup have to be programmed. 11_B State Machine, TXFIFO and RXFIFO reset and registers reset
0	8	r	**Reserved** Read as 0; should be written with 0.

30.4.19 Global configuration register 1

GLOBALCON1 contains bit fields for control of the extension of the slave select signals by using an external demultiplexer.

GLOBALCON1 Offset address: 114_H

Global configuration register 1 Kernel Reset value: $001B\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RXFM		TXFM		PT2			PT1			RXFIFOINT			TXFIFOINT		
rw		rw		rw			rw			rw			rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
USREN	HWTRSEL	FD	PT2EN	PT1EN	RXEN	TXEN	ERRORENS								
rw	rw	rw	rw	rw	rw	rw	rw								

Field	Bits	Type	Description
ERRORENS	8:0	rw	**Error Enable Bits** Bits for enabling interrupt on all available error types 000_H All errors disabled 001_H Parity Error (PAREEN) 002_H Unexpected Configuration Error 004_H Trigger Ignored Error (master mode), Baud Rate Error (slave mode) BEN 008_H TXFIFO overflow (software error) 010_H TXFIFO underflow (slave mode) TEN 020_H RXFIFO overflow REN 040_H RXFIFO underflow (software error) 080_H EXPECT timeout 100_H SLSI misplaced inactivation enable
TXEN	9	rw	**Tx Interrupt Event Enable** Enables the Tx interrupt. 0_B Disabled 1_B Enabled
RXEN	10	rw	**RX Interrupt Event Enable** Enables the Rx interrupt. 0_B Disabled 1_B Enabled
PT1EN	11	rw	**Interrupt on PT1 Event Enable** Enables the PT interrupt on an PT1 event, as selected by the PT1 bit field. 0_B Disabled 1_B Enabled
PT2EN	12	rw	**Interrupt on PT2 Event Enable** Enables the PT interrupt on an PT2 event, as selected by the PT2 bit field. 0_B Disabled 1_B Enabled
FD	13	rw	**Freeze Data** Freezes the TXDATA in case the hardware trigger (HWTREN) is selected. 0_B Disabled 1_B Enabled
HWTRSEL	14	rw	**Hardware trigger selection** Selection bit for source of external trigger. 0_B GTM 1_B eGTM

(table continues...)

(continued)

Field	Bits	Type	Description
USREN	15	rw	**Interrupt on USR Event Enable** Enables the USR interrupt on an USR event, as selected by the PT1 bit field. ***Note**: In Move Counter mode of operation, User Interrupt line is reused for the IAL and IBL interrupts, which can be enabled by MCCON.IALEN and MCCON.IBLEN. Bit USREN does not influence these interrupts.* 0_B Disabled 1_B Enabled
TXFIFOINT	18:16	rw	**Transmit FIFO Interrupt Threshold** In Combined Mode, as long as the TXFIFO filling level is equal or less than this threshold, than each move of data or configuration from the TXFIFO triggers a transmit interrupt. In Batch Mode, interrupt is generated only at the moment of filling level falling below the threshold level. In Single Mode, this bit field ignored. 000_B 1 001_B 2 010_B 3 011_B 4 100_B 5 101_B 6 110_B 7 111_B 8
RXFIFOINT	21:19	rw	**Receive FIFO Interrupt Threshold** In Combined Mode, as long as the RXFIFO filling level is equal or greater than this threshold, than each move of data or status (if enabled) into the RXFIFO triggers a receive interrupt. In Batch mode, interrupt is generated only at the moment of filling level exceeding this threshold level. In Single Mode, this bit field is ignored. 000_B 0 001_B 1 010_B 2 011_B 3 100_B 4 101_B 5 110_B 6 111_B 7

(table continues...)

(continued)

Field	Bits	Type	Description
PT1	24:22	rw	**Phase Transition Event 1** Selects the first phase transition to trigger the PT interrupt. 000_B BUSY Master Mode: End of WAIT phase Slave Mode: Transmit data is present waiting for the shift clock 001_B SCLKPC Master Mode: Serial clock polarity change 010_B SOF Master Mode: Start of Frame Slave Mode: Transmission of the first data bit started 011_B TBE Master Mode: Transmit Buffer Emptied Slave Mode: Data is taken from the TXFIFO by the QSPI shift engine 100_B RBF Master Mode: Receive Buffer Filled Slave Mode: Received data is written to the RXFIFO) 101_B EOF Master Mode: End of Frame Slave Mode: The last data bit has been received 110_B DNA Master Mode: Data not Available = Start of Expect 111_B CONT Master Mode: End of EXPECT phase
PT2	27:25	rw	**Phase Transition Event 2** Selects the second phase transition to trigger the PT2 interrupt. In Master Mode, the following events are available. In Slave Mode, only the SLSI signal (rising edge) triggers the interrupt for PT2. For this purpose, always use the setting 101 (EOF). This interrupt is independent of the CS value. 000_B BUSY Master Mode: end of WAIT phase 001_B SCLKPC Master Mode: serial clock polarity change 010_B SOF Master Mode: Start of Frame 011_B TBE Master Mode: Transmit Buffer Emptied 100_B RBF Master Mode: Receive Buffer Filled 101_B EOF Master Mode: End of Frame Slave Mode: SLSI deactivated (rising edge on the SLSI pin) 110_B DNA Master Mode: Data not Available = Start of Expect 111_B CONT Master Mode: End of EXPECT phase
TXFM	29:28	rw	**TXFIFO Mode** Selects the TXFIFO mode. 00_B Combined Move Mode 01_B Single Move Mode 10_B Batch move Mode 1 11_B Batch move Mode 2
RXFM	31:30	rw	**RXFIFO Mode** Selects the RXFIFO mode. 00_B Combined Move Mode 01_B Single Move Mode 10_B Batch move Mode 1 11_B Batch move Mode 2

30.4.20 Basic configuration register

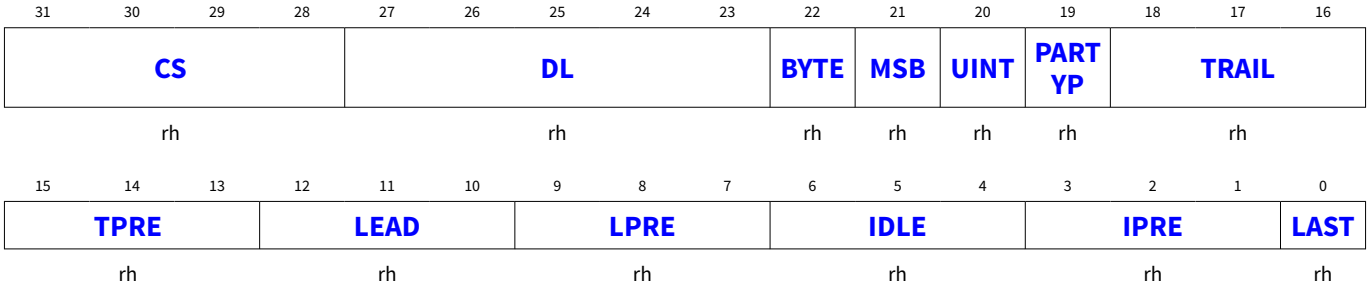
Defines the basic configuration parameters for the current slave select. It can be read by software, or written through a write to the TXFIFO.

If an application does not use the SLSO signals of the QSPI module, but uses software channel selection, then the alternate function multiplexer in the ports must be configured so that the signal SLSO is not selected, but the corresponding bit Pn.OUT.

Alternatively, to emulate SLSO functionality, the software can disable (low) all the SSOC.OEN bits, and then toggle the SSOC.AOL bits.

BACON Offset address: 118_H

Basic configuration register Kernel Reset value: $0F87\ 1C71_H$



Field	Bits	Type	Description
LAST	0	rh	**Last Word in a Frame** Defines if the following data word is last in the current frame or not 0_B Not Last 1_B Last
IPRE	3:1	rh	**Prescaler for the Idle Delay** Length in T_{PER} units 000_B 1 001_B 4 010_B 16 … 111_B 16384
IDLE	6:4	rh	**Idle Delay Length** Defines the length of both idle delays, IDLEA and IDLEB, in T_{PER} units pre scaled with IPRE 000_B 1 unit 001_B 2 units … 111_B 8 units

(table continues...)

(continued)

Field	Bits	Type	Description
LPRE	9:7	rh	**Prescaler for the Leading Delay** Length in T_{PER} units 000_B 1 001_B 4 010_B 16 … 111_B 16384
LEAD	12:10	rh	**Leading Delay Length** Defines the length of the leading delay, in T_{PER} units pre scaled with LPRE 000_B 1 unit 001_B 2 units … 111_B 8 units
TPRE	15:13	rh	**Prescaler for the Trailing Delay** Length in T_{PER} units 000_B 1 001_B 4 010_B 16 … 111_B 16384
TRAIL	18:16	rh	**Trailing Delay Length** Defines the length of the trailing delay, in T_{PER} units pre scaled with TPRE 000_B 1 unit 001_B 2 units … 111_B 8 units
PARTYP	19	rh	**Parity Type** Valid for both receive and transmit direction 0_B Even parity 1_B Odd parity
UINT	20	rh	**User Interrupt at the PT1 Event in the Subsequent Frames** This bit is an enable signal for the PT1 event routed to the User Interrupt Service Request. The interrupt signals are generated until disabled with the next BACON. 0_B Disabled 1_B Enabled

(table continues...)

(continued)

Field	Bits	Type	Description
MSB	21	rh	**Shift MSB or LSB First** This bit sets the shift direction of the shift register. If the MSB option is set, and the data is a block longer than 32 bits, the block must be fed into the TXFIFO in reverse direction, from the end of the block until its beginning. 0_B Shift LSB first 1_B Shift MSB first
BYTE	22	rh	**Byte** Defines if data length is expressed in bits or bytes. In Slave Mode BYTE must be '0'. 0_B DL defines the data length in bits 1_B DL defines the data length in bytes
DL	27:23	rh	**Data Length** Defines the data length in bits or bytes of one data block, depending on the setting of the bit field BYTE. For the maximum baud rate of 50 MBaud, the minimum data length possible is four. 00_H 2 bits if BYTE=0; XXL mode if BYTE=1 01_H 2 bits or bytes … $1F_H$ 32 bits or bytes
CS	31:28	rh	**Channel Select** Selects the channel to which the subsequent data entry belongs (channel = the SLSO signal to be activated and the corresponding ECON configuration extension) This bit field selects one slave in a range of 0 to 15, by driving one SLSO signal out of 16 available. In case of an external demux mode, this bit field appears on the selected SLSO lines as it is, additionally inverted or not, as defined in the SSOC register.

30.4.21 Configuration extension z

Configuration extensions for channels 0 to 15. Register z defines several timing characteristics for two channels: z and z+8.

ECONz (z=0-7) Offset address: 120_H+z*4

Configuration extension z Kernel Reset value: $0000\ 1450_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BE		0												D	
rw		r												rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	PAREN	CPOL	CPH	C		B		A		Q					
r	rw	rw	rw	rw		rw		rw		rw					

Field	Bits	Type	Description
Q	5:0	rw	**Time Quantum** Defines the time quantum length used by A, B, and C to define the baud rate and duty cycle by. This prescaler cascades the prescaler GLOBALCON.TQ. 00_H 1 01_H 2 … $3F_H$ 64
A	7:6	rw	**Bit Segment 1** Length expressed in time quantums of ECONz.Q. 00_B 1 01_B 2 10_B 3 11_B 4
B	9:8	rw	**Bit Segment 2** Length expressed in time quantums of ECONz.Q. 00_B 0 01_B 1 10_B 2 11_B 3
C	11:10	rw	**Bit Segment 3** Length expressed in time quantums of ECONz.Q. 00_B 0 (if B=0, than C is minimum 1 per hardware) 01_B 1 10_B 2 11_B 3
CPH	12	rw	**Clock Phase** Delay of one half SCLK clock cycle. 0_B Disabled 1_B Enabled
CPOL	13	rw	**Clock Polarity** Idle level of the shift clock signal at the SCLK pin 0_B Idle level low 1_B Idle level high
PAREN	14	rw	**Enable Parity Check** This bit field enables both the parity generation in transmit and parity check in receive direction. 0_B Disabled 1_B Enabled

(table continues...)

(continued)

Field	Bits	Type	Description
D	17:16	rw	**Bit Segment 4** Length expressed in time quantums of ECONz.Q. (Only applicable when GLOBALCON.HS is enabled.) 00_B 1 01_B 2 10_B 3 11_B 4
BE	31:30	rw	**Permutate bytes to / from Big Endian** 00_B Disabled 01_B 16-bit big endian 10_B 32-bit big endian 11_B Disabled
0	15, 29:18	r	**Reserved**

30.4.22 Status register

Status register contains bits flagging the current status of the module and its sub-modules.

***Note**: It is not recommended to set the STATUS register flags per software for purposes other than testing. In such cases, use write instructions only, not read-modify-write instructions or bit instructions which compile to read-modify-write instructions.*

When using the RXFIFO status injection feature, only bits 22 to 31 reflect the status at the moment of injection, that is for the latest frame. Due to pipeline effects, the other bits contain not-latest information.

Slave TXFIFO underflow error is activated if the transmission has been started by the master at the time when the slave FIFO was empty, or at the same moment updated. In the second case inconsistent data will be transmitted.

Reading the TXFIFO filling level bit field returns a value which can be used to calculate how many writes can be performed by a CPU without causing an overflow (in case no DMA is programmed to access the TXFIFO in parallel). For example, if the TXFIFO level was one, maximum seven (TXFIFO depth of eigth minus one) write accesses are possible. Due to the volatility of the bit field, the filling level can go down in some nanoseconds after the read.

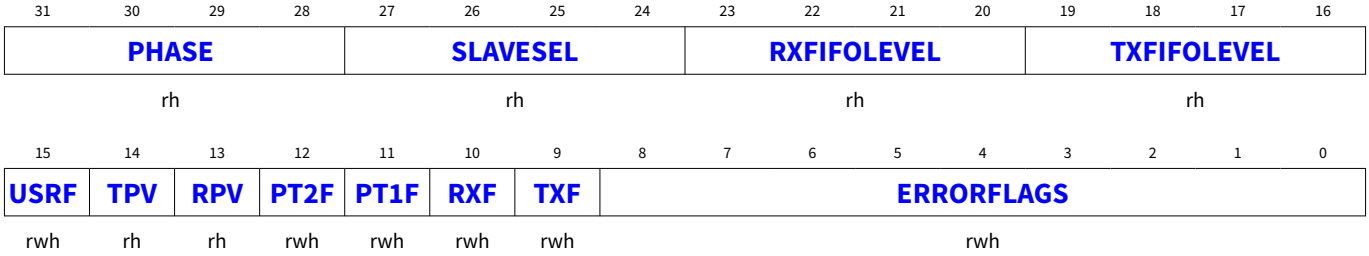
Reading the RXFIFO filling level bit field returns a value which shows directly how many reads can be performed by a CPU without causing an underflow. Due to the volatility of the bit field, the filling level can go up in some nanoseconds after the read.

Reading the PHASE bit field shows a previous phase of the frame, and a PT1F and PT2F flags indicate previous phase transitions. If the communication speed is not too high or the phases duration not very short compared to the software latency delays, which would normally be the case, these would be the latest completed phase and the latest phase transitions. Nevertheless, in case of high baud rates, the duration of the phases must be taken into consideration, but only if these bit fields are used in an application, and not only for debugging purposes.

SLSI misplaced inactivation error interrupt is raised when SLSI is deactivated by the master while the data transfer is still ongoing.

STATUS Offset address: 140_H

Status register Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
ERRORFLAGS	8:0	rwh	**Sticky Flags Signalling Errors** Writing 1 sets the error Flag and triggers an error interrupt, if enabled. Writing 0 has no effect. 000_H No Error 001_H Parity Error 002_H Unexpected Configuration Error 004_H Trigger Ignored Error (master mode), Baud Rate Error (slave mode) 008_H TXFIFO overflow (software error) 010_H TXFIFO underflow (slave mode) 020_H RXFIFO overflow 040_H RXFIFO underflow (software error) 080_H EXPECT time out error 100_H SLSI misplaced inactivation (slave mode)
TXF	9	rwh	**Transmit Interrupt Request Flag** Flags an occurrence of a request to feed the TXFIFO, which is generated when an element is fetched from the FIFO, and the FIFO filling level is equal or less than the set threshold level. Writing 1 sets the flag and triggers an interrupt if GLOBALCON1.TXEN = 1. Writing 0 has no effect.
RXF	10	rwh	**Receive Interrupt Request Flag** Flags an occurrence of a request to empty the RXFIFO, which is generated when an element is written into the FIFO, and the FIFO filling level is equal or greater than the set threshold level. Writing 1 sets the flag and triggers an interrupt if GLOBALCON1.RXEN = 1. Writing 0 has no effect.
PT1F	11	rwh	**Phase Transition 1 Flag** Flags an occurrence of a PT1 event, as selected with the GLOBALCON1.PT1, and triggers an interrupt if GLOBALCON1.PT1EN = 1. Writing 1 sets the flag and triggers an error interrupt. Writing 0 has no effect.

(table continues...)

(continued)

Field	Bits	Type	Description
PT2F	12	rwh	**Phase Transition 2 Flag** In master mode, flags an occurrence of a PT2 event, as selected with the GLOBALCON1.PT2, and triggers an interrupt if GLOBALCON1.PT2EN = 1. In slave mode, set by the SLSI deactivated event. Writing 1 sets the flag and triggers an error interrupt. Writing 0 has no effect.
RPV	13	rh	**Received Parity Value** Displays the last received parity bit, if parity was enabled. Else if the parity is disabled, reads 0.
TPV	14	rh	**Transmitted Parity Value** Displays the last transmitted parity bit, if parity was enabled. Else 0.
USRF	15	rwh	**User Interrupt Request Flag** Flags an occurrence of an USR event. Writing 1 sets the flag and triggers an interrupt if GLOBALCON1.USREN = 1. Writing 0 has no effect.
TXFIFOLEVEL	19:16	rh	**TXFIFO Filling Level** Shows how many entries in the TXFIFO are waiting for transmission 0_H 0 1_H 1 2_H 2 3_H 3 4_H 4 5_H 5 6_H 6 7_H 7 8_H 8 **others,** reserved
RXFIFOLEVEL	23:20	rh	**RXFIFO Filling Level** Shows how many entries in the RXFIFO are waiting for software to move them to RAM 0_H 0 1_H 1 2_H 2 3_H 3 4_H 4 5_H 5 6_H 6 7_H 7 8_H 8 **others,** reserved
SLAVESEL	27:24	rh	**Currently Active Slave Select Flag** Displays the currently active slave select.

(table continues...)

(continued)

Field	Bits	Type	Description
PHASE	31:28	rh	**Flags the ongoing phase** Displays the current phase number. Relevant only in master mode. In slave mode this bit field indicates always 0. Not 0 means busy. 0_H Wait 1_H Idle A 2_H Idle B 3_H Lead 4_H Data 5_H Trail 6_H Expect 7_H Lead Strobe 8_H Trail Strobe

30.4.23 Status register 1

If the STATUS.ERRORFLAGS[2] bit is cleared through FLAGSCLEAR.ERRORCLEARS[2], the both flags SPD and BRD are automatically cleared.

STATUS1 Offset address: 144_H

Status register 1 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SPD	SPD EN	BRD	BRD EN	0											
rwh	rw	rwh	rw	r											

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								BITCOUNT							
r								r							

Field	Bits	Type	Description
BITCOUNT	7:0	r	**Number of bits shifted out** Supports up to 256 bits. A BITCOUNT value of greater than 0 indicates that a transmission is in progress. The value is not accurate; it may be lower than the number of bits shifted. After transmission of the last bit, BITCOUNT is set to zero.
BRDEN	28	rw	**Baud Rate Deviation Enable** Enables the signal path. 0_B Disabled 1_B Enabled

(table continues...)

(continued)

Field	Bits	Type	Description
BRD	29	rwh	**Baud Rate Deviation Flag** Shows if baud rate deviation has been detected. Write of 1 sets the bit and raises the event per software. Write of 0 has no effect. 0_B no event 1_B event detected
SPDEN	30	rw	**Spike Detection Enable** Enables the signal path. 0_B Disabled 1_B Enabled
SPD	31	rwh	**Spike Detection Flag** Shows if spike has been detected. Write of 1 sets the bit and raises the event per software. Write of 0 has no effect. 0_B No event 1_B Event detected
0	27:8	r	**Reserved** Read as 0; should be written with 0.

30.4.24 Slave select output control register

SSOC controls the level of each slave select and enables/disables each one individually.

SSOC Offset address: 148_H

Slave select output control register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OEN															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AOL															
rw															

Field	Bits	Type	Description
AOL	15:0	rw	**Active Output Level for the SLSO Outputs** The idle level is the inverted one. “0” at certain position means active low level for the corresponding SLSO. “1” means active high.
OEN	31:16	rw	**Enable Bits for the SLSO Outputs** In disabled state the SLSO output drives the idle level as defined by the AOL bit field. “0” at certain position means that the corresponding SLSO is disabled. “1” means enabled.

30.4.25 Flags clear register

FLAGSCLEAR Offset address: 150_H

Flags clear register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
USRC	0	0	PT2C	PT1C	RXC	TXC	ERRORCLEARS								
w	r	r	w	w	w	w	w								

Field	Bits	Type	Description
ERRORCLEARS	8:0	w	**Write Only Bits for Clearing the Error Flags** Writing 1 clears the corresponding error flag in the ERORRFLAGS bit field. Reading returns 0. 000_H No clear 001_H Parity Error clear 002_H Unexpected Configuration Error clear 004_H Trigger Ignored Error and Baud Rate Error clear 008_H TXFIFO overflow clear 010_H TXFIFO underflow clear 020_H RXFIFO overflow clear 040_H RXFIFO underflow clear 080_H EXPECT time out clear 100_H SLSI misplaced inactivation clear
TXC	9	w	**Transmit Event Flag Clear** Write of 1 clears the STATUS.TXF bit. Write of 0 has no effect. Read delivers 0. 0_B No action 1_B Clear
RXC	10	w	**Receive Event Flag Clear** Write of 1 clears the STATUS.RXF bit. Write of 0 has no effect. Read delivers 0. 0_B No action 1_B Clear
PT1C	11	w	**PT1 Event Flag Clear** Write of 1 clears the STATUS.PT1F bit. Write of 0 has no effect. Read delivers 0. 0_B No action 1_B Clear

(table continues...)

(continued)

Field	Bits	Type	Description
PT2C	12	w	**PT2 Event Flag Clear** Write of 1 clears the STATUS.PT2F bit. Write of 0 has no effect. Read delivers 0. 0_B No action 1_B Clear
USRC	15	w	**User Event Flag Clear** Write of 1 clears the STATUS.USRF bit. Write of 0 has no effect. Read delivers 0. 0_B No action 1_B Clear
0	13, 14, 31:16	r	**Reserved** Read as 0; should be written with 0.

30.4.26 Extra large data configuration register

The XXLCON register provides counter for sending frames with very long blocks of data by extending the long data mode. It avoids a need for further BACON entries, like those needed in continuous mode.

Data length in this register overrides the BACON.DL setting in XXL mode, when BACON.BYTE=1 and BACON.DL=0. The data is sent to the corresponding slave as defined by BACON.CS bit field.

XXLCON Offset address: 154_H

Extra large data configuration register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BYTECOUNT															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
XDL															
rw															

Field	Bits	Type	Description
XDL	15:0	rw	**Extended Data Length** Defines the length of the data block in bytes in range of 2 to 65536. Overrides BACON.DL when BACON.BYTE=1 and BACON.DL=0. 0000_H 2 bytes 0001_H 2 bytes 0002_H 3 bytes … $FFFF_H$ 65536 bytes

(table continues...)

(continued)

Field	Bits	Type	Description
BYTECOUNT	31:16	rh	**Extended Data Length** In the XXL mode, shows the current state of the internal byte down counter (bytes remaining to be sent). In short and long modes, the value of this bit field is ignored. 0000_H 0 bytes 0001_H 2 bytes 0002_H 3 bytes … $FFFF_H$ 65536 bytes

30.4.27 MIX_ENTRY register

MIXENTRY Offset address: 158_H

MIX_ENTRY register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
E															
w															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E															
w															

Field	Bits	Type	Description
E	31:0	w	**Entry Point to the TxFIFO**

30.4.28 BACON_ENTRY register

BACONENTRY Offset address: $15C_H$

BACON_ENTRY register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CS				DL					BYTE	MSB	UINT	PARTYP	TRAIL		
w				w					w	w	w	w	w		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TPRE			LEAD			LPRE			IDLE			IPRE			LAST
w			w			w			w			w			w

Field	Bits	Type	Description
LAST	0	w	**Last Word in a Frame** Defines if the following data word is last in the current frame or not 0_B Not Last 1_B Last
IPRE	3:1	w	**Prescaler for the Idle Delay** Length in T_{PER} units 000_B 1 001_B 4 010_B 16 … 111_B 16384
IDLE	6:4	w	**Idle Delay Length** Defines the length of both idle delays, IDLEA and IDLEB, in T_{PER} units pre scaled with IPRE 000_B 1 unit 001_B 2 units … 111_B 8 units
LPRE	9:7	w	**Prescaler for the Leading Delay** Length in T_{PER} units 000_B 1 001_B 4 010_B 16 … 111_B 16384
LEAD	12:10	w	**Leading Delay Length** Defines the length of the leading delay, in T_{PER} units pre scaled with LPRE 000_B 1 unit 001_B 2 units … 111_B 8 units
TPRE	15:13	w	**Prescaler for the Trailing Delay** Length in T_{PER} units 000_B 1 001_B 4 010_B 16 … 111_B 16384

(table continues...)

(continued)

Field	Bits	Type	Description
TRAIL	18:16	w	**Trailing Delay Length** Defines the length of the trailing delay, in T_{PER} units pre scaled with TPRE 000_B 1 unit 001_B 2 units … 111_B 8 units
PARTYP	19	w	**Parity Type** Valid for both receive and transmit direction 0_B Even parity 1_B Odd parity
UINT	20	w	**User Interrupt at the PT1 Event in the Subsequent Frames** This bit is an enable signal for the PT1 event routed to the User Interrupt Service Request. The interrupt signals are generated until disabled with the next BACON. 0_B Disabled 1_B Enabled
MSB	21	w	**Shift MSB or LSB First** This bit sets the shift direction of the shift register. If the MSB option is set, and the data is a block longer than 32 bits, the block must be fed into the TXFIFO in reverse direction, from the end of the block until its beginning. 0_B Shift LSB first 1_B Shift MSB first
BYTE	22	w	**Byte** Defines if data length is expressed in bits or bytes. In Slave Mode BYTE must be '0'. 0_B DL defines the data length in bits 1_B DL defines the data length in bytes
DL	27:23	w	**Data Length** Defines the data length in bits or bytes of one data block, depending on the setting of the bit field BYTE. For the maximum baud rate of 50 MBaud, the minimum data length possible is four. 00_H 2 bits if BYTE=0; XXL mode if BYTE=1 01_H 2 bits or bytes … $1F_H$ 32 bits or bytes

(table continues...)

(continued)

Field	Bits	Type	Description
CS	31:28	w	**Channel Select** Selects the channel to which the subsequent data entry belongs (channel = the SLSO signal to be activated and the corresponding ECON configuration extension) This bit field selects one slave in a range of 0 to 15, by driving one SLSO signal out of 16 available. In case of an external demux mode, this bit field appears on the selected SLSO lines as it is, additionally inverted or not, as defined in the SSOC register.

30.4.29 DATA_ENTRY register x

DATAENTRYx (x=0-7) Offset address: 160_H+x*4

DATA_ENTRY register x Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
E															
w															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E															
w															

Field	Bits	Type	Description
E	31:0	w	**Entry Point to the TxFIFO**

30.4.30 RX_EXIT register

RXEXIT Offset address: 180_H

RX_EXIT register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
E															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E															
rh															

Field	Bits	Type	Description
E	31:0	rh	**Read Point from the RxFIFO**

Table 1243 Mirrors of RXEXIT

Short name	Long name	Offset address
RXEXIT# (#=0-7)	RX_EXIT register mirror #	180_H+#*4

30.4.31 RX_EXIT debug register

The register RXEXITD provides a non-destructive address, showing the next available value in the RXFIFO.

***Note**: This register provides a non-volatile access to the RXFIFO. It delivers the same value as RXEXIT, but without affecting read pointer and the filling level of the RXFIFO.*

RXEXITD Offset address: $1A0_H$

RX_EXIT debug register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
E															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E															
rh															

Field	Bits	Type	Description
E	31:0	rh	**Read Point from the RxFIFO**

30.4.32 Move counter register

The Move Counter Register holds the different configuration parameters of the Move Counter Mode

MC Offset address: $1B4_H$

Move counter register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CURRENT															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MCOUNT															
rw															

Field	Bits	Type	Description
MCOUNT	15:0	rw	**Move Count** Defines the number of moves to be performed in short mode, in range of 1 to 65535.
CURRENT	31:16	rh	**Current Status of the Move Counter** Shows the current status of the Move Counter, that is, how many data blocks are to be transmitted until the end of the frame. The value is in the range of 0 to 65535.

30.4.33 Move counter control register

Contains the control bits for the Move Counter Mode. See the description MC register.

MCCON Offset address: $1B8_H$

Move counter control register Kernel Reset value: $0000\ 0000_H$

Field	Bits	Type	Description
TPRE2	2:0	rw	**Prescaler for the Trailing Delay 2** Trailing delay injected in the configuration register for the last data if T2EN is set. Length in units T_{PER} 000_B 1 001_B 4 010_B 16 … 111_B 16384
TRAIL2	5:3	rw	**Last Trailing Delay** Trailing delay injected in the configuration register for the last data if T2EN is set. Defines the length of the leading delay, in T_{PER} units pre scaled with TPRE 000_B 1 unit 001_B 2 units … 111_B 8 units
IBLEN	16	rw	**Interrupt Before Last Enable** Enable bit for this event. 0_B Disabled 1_B Enable
IBLF	17	rh	**Interrupt Before Last Flag** Flag bit for this event. 0_B No event 1_B Event occurred
IBLC	18	w	**Clear Bit for IBLF** Writing 1 clears the IBLF. Writing 0 has no effect. Returns 0 on read.

(table continues...)

(continued)

Field	Bits	Type	Description
IBLS	19	w	**Set Bit for IBLF** Writing 1 sets the IBLF and triggers an interrupt (if enabled). Writing 0 has no effect. Returns 0 on read.
IALEN	20	rw	**Interrupt After Last Enable** Enable bit for this event. 0_B Disabled 1_B Enable
IALF	21	rh	**Interrupt After Last Flag** Flag bit for this event. 0_B No event 1_B Event occurred
IALC	22	w	**Clear Bit for IALF** Writing 1 clears the IALF. Writing 0 has no effect. Returns 0 on read.
IALS	23	w	**Set Bit for IALF** Writing 1 sets the IALF and triggers an interrupt (if enabled). Writing 0 has no effect. Returns 0 on read.
T2EN	30	rw	**TRAIL 2 Injection Enable** This bit has to be configured before the transmission of the frame starts. If set, a new value for the last trailing delay will be injected for the last data block, as defined with the bit field TRAIL2. If not set, the TRAIL value from the latest BACON will be valid also as the last trailing delay.
MCEN	31	rw	**Move Counter Enable** Enables the Move Counter feature. If enabled, the MCOUNT value is taken in consideration, otherwise the standard continuous mode is active.
0	15:6, 29:24	r	**Reserved** Read as 0; should be written with 0.

30.4.34 Multiple slave select register

This registers enables and configures the multiple slave select mode.

MSSMUX Offset address: $1BC_H$

Multiple slave select register Kernel Reset value: $0000\ 4321_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MSSOUTS				0											
rw				r											

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MSSOUT3				MSSOUT2				MSSOUT1				MSSOUT0			
rw				rw				rw				rw			

Field	Bits	Type	Description
MSSOUT0	3:0	rw	**Multiple Slave Select output mux 0** Selects the routing of the corresponding chip select signal. 0_H SLSO0 1_H SLSO1 … F_H SLSO15
MSSOUT1	7:4	rw	**Multiple Slave Select output mux 1** Selects the routing of the corresponding chip select signal. 0_H SLSO0 1_H SLSO1 … F_H SLSO15
MSSOUT2	11:8	rw	**Multiple Slave Select output mux 2** Selects the routing of the corresponding chip select signal. 0_H SLSO0 1_H SLSO1 … F_H SLSO15
MSSOUT3	15:12	rw	**Multiple Slave Select output mux 3** Selects the routing of the corresponding chip select signal. 0_H SLSO0 1_H SLSO1 … F_H SLSO15
MSSOUTS	31:28	rw	**Multiple Slave Select output mux strobe** Selects the routing of the corresponding chip select signal. 0_H SLSO0 1_H SLSO1 … F_H SLSO15
0	27:16	r	**Reserved** Read as 0; should be written with 0.

30.5 Debug information

OCDS suspend

The OCDS hard suspend request, for debugging purposes, simply freezes the QSPI module in the current state. No signal is changed including the service request lines, which remain frozen in the current state.

***Attention**: On chips where hard OCDS suspend simply switches the clock off immediately, without waiting on acknowledge from the module, all output signals are simply frozen.*

The QSPI responds to an OCDS soft suspend request by entering the PAUSE state immediately after the subsequent end of a trail phase, and then sending acknowledge (see below Figure). The behavior is the same as for entering the Sleep mode.

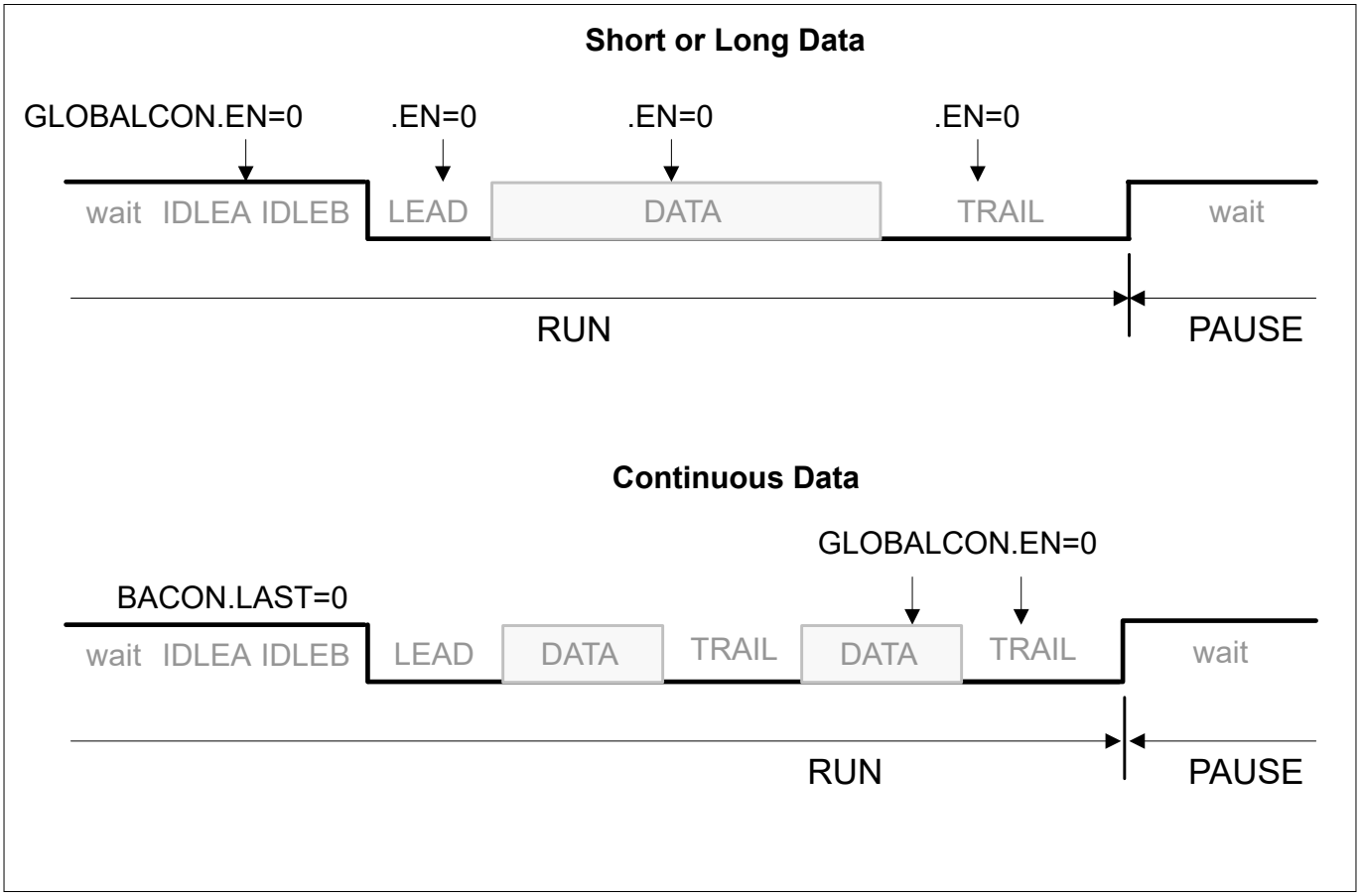


Figure 705 Entering the sleep, soft suspend, and disable state

Reading the FIFO entries by the Cerberus (detected by the FPI master tag) does not modify the FIFO content nor the read and write pointers.

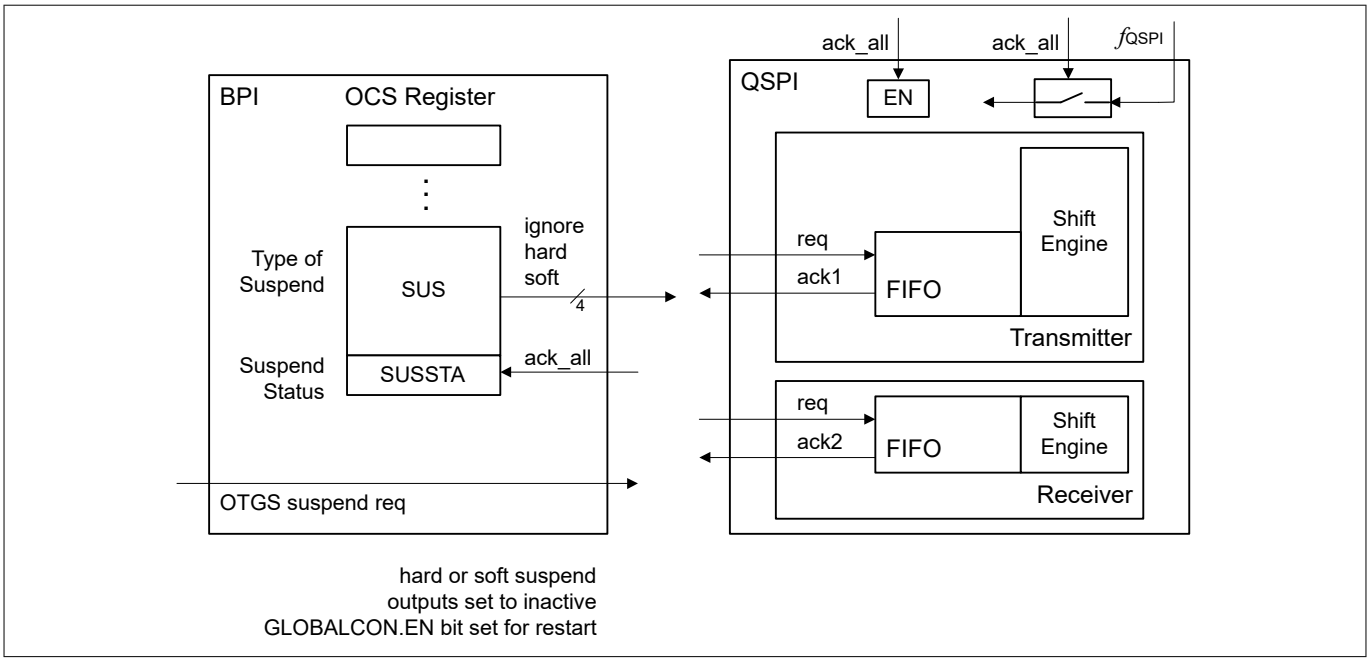


Figure 706 Suspend overview

Please note the following:

- In Hard Suspend Mode:
  - The QSPI kernel clock is switched off immediately
  - Writing to registers is possible but will enable the kernel clock for a few cycles(see note below)
  - Read access to a register does not let any clock cycles through, but is done in combinatorial way. Therefore, RXFIFO pointer does not move

Attention*: *These clock pulses in Hard Suspend Mode can have unintended side effects like signals becoming and staying active. This can affect also other modules, so a QSPI kernel reset might not be sufficient to bring the system into a defined state

- In Soft Suspend Mode:
  - The QSPI kernel clock is switched on, the module is only paused
  - Each read and write can cause state transition in some state machine in the module. Therefore, RXFIFO reacts to the read accesses

30.6 References

This section is not applicable for the module being described.

30.7 QSPI revision history

Initial release of the chapter.

30.8 TC4Dx QSPI information

30.8.1 TC4Dx QSPI configuration

Table 1244 Device specific configuration

Parameter	Value
Number of instances	8

30.8.2 TC4Dx QSPI features

There are no deviations from the generic specification.

30.8.3 TC4Dx QSPI functional description

In TC4Dx device, the QSPI module is connected to ComPB bus interface.

30.8.4 TC4Dx QSPI registers

30.8.4.1 Register address space - QSPI

Table 1245 Registers address space - QSPI

Module	Base address	End address	Note
QSPI0	$F4401000_H$	$F44011FF_H$	Register block QSPI0
QSPI1	$F4401200_H$	$F44013FF_H$	Register block QSPI1
QSPI2	$F4401400_H$	$F44015FF_H$	Register block QSPI2
QSPI3	$F4401600_H$	$F44017FF_H$	Register block QSPI3
QSPI4	$F4401800_H$	$F44019FF_H$	Register block QSPI4
QSPI5	$F4401A00_H$	$F4401BFF_H$	Register block QSPI5
QSPI6	$F4401C00_H$	$F4401DFF_H$	Register block QSPI6
QSPI7	$F4401E00_H$	$F4401FFF_H$	Register block QSPI7

30.8.4.2 Register overview - access mode glossary

Table 1246 Register overview - access mode glossary

Keyword	Description
E	Access protection using PROT register QSPI0_PROTE or QSPI1_PROTE or QSPI2_PROTE or QSPI3_PROTE or QSPI4_PROTE or QSPI5_PROTE or QSPI6_PROTE or QSPI7_PROTE .
SE	Access protection using PROT register QSPI0_PROTSE or QSPI1_PROTSE or QSPI2_PROTSE or QSPI3_PROTSE or QSPI4_PROTSE or QSPI5_PROTSE or QSPI6_PROTSE or QSPI7_PROTSE .
APU-P	Protection group consisting of registers QSPI0_ACCEN_WRA , QSPI0_ACCEN_WRB , QSPI0_ACCEN_RDA , QSPI0_ACCEN_RDB , QSPI0_ACCEN_VM , QSPI0_ACCEN_PRS or QSPI1_ACCEN_WRA , QSPI1_ACCEN_WRB , QSPI1_ACCEN_RDA , QSPI1_ACCEN_RDB , QSPI1_ACCEN_VM , QSPI1_ACCEN_PRS or QSPI2_ACCEN_WRA , QSPI2_ACCEN_WRB , QSPI2_ACCEN_RDA , QSPI2_ACCEN_RDB , QSPI2_ACCEN_VM , QSPI2_ACCEN_PRS or QSPI3_ACCEN_WRA , QSPI3_ACCEN_WRB , QSPI3_ACCEN_RDA , QSPI3_ACCEN_RDB , QSPI3_ACCEN_VM , QSPI3_ACCEN_PRS or QSPI4_ACCEN_WRA , QSPI4_ACCEN_WRB , QSPI4_ACCEN_RDA , QSPI4_ACCEN_RDB , QSPI4_ACCEN_VM , QSPI4_ACCEN_PRS or QSPI5_ACCEN_WRA , QSPI5_ACCEN_WRB , QSPI5_ACCEN_RDA , QSPI5_ACCEN_RDB , QSPI5_ACCEN_VM , QSPI5_ACCEN_PRS or QSPI6_ACCEN_WRA , QSPI6_ACCEN_WRB , QSPI6_ACCEN_RDA , QSPI6_ACCEN_RDB , QSPI6_ACCEN_VM , QSPI6_ACCEN_PRS or QSPI7_ACCEN_WRA , QSPI7_ACCEN_WRB , QSPI7_ACCEN_RDA , QSPI7_ACCEN_RDB , QSPI7_ACCEN_VM , QSPI7_ACCEN_PRS .
P	Access protection using *APU-P* registers.
BE	Always returns a Bus Error.
U	No access restrictions.
PROT	Access restrictions as defined in the PROT register access rules.

30.8.4.3 Register overview - QSPI0 (ascending offset address)

Table 1247 Register overview - QSPI0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
QSPI0_CLC	Clock control register	000_H	P	P, SV, E	Application Reset	5364
QSPI0_OCS	OCDS control and status register	004_H	P	SV, P	Debug Reset	5364
QSPI0_ID	Module identification register	008_H	P	BE	PowerOn Reset	5365
QSPI0_RST_CTRLA	Reset control register A	$00C_H$	P	P, SV, E	Application Reset	5366
QSPI0_RST_CTRLB	Reset control register B	010_H	P	P, SV, E	Application Reset	5366
QSPI0_RST_STAT	Reset status register	014_H	P	BE	Application Reset	5367
QSPI0_PROTE	PROT register endinit	020_H	U	SV, PROT	Application Reset	5368
QSPI0_PROTSE	PROT register safe endinit	024_H	U	SV, PROT	Application Reset	5369
QSPI0_ACCEN_WRA	Write access enable register A	040_H	U	SE, SV	Application Reset	5371
QSPI0_ACCEN_WRB	Write access enable register B	044_H	U	SE, SV	Application Reset	5372
QSPI0_ACCEN_RDA	Read access enable register A	048_H	U	SE, SV	Application Reset	5372
QSPI0_ACCEN_RDB	Read access enable register B	$04C_H$	U	SE, SV	Application Reset	5373
QSPI0_ACCEN_VM	VM access enable register	050_H	U	SE, SV	Application Reset	5373
QSPI0_ACCEN_PRS	PRS access enable register	054_H	U	SE, SV	Application Reset	5374
QSPI0_PISEL	Port input select register	104_H	P	SV, P	Kernel Reset	5374
QSPI0_GLOBALCON	Global configuration register	110_H	P	SV, P	Kernel Reset	5376
QSPI0_GLOBALCON1	Global configuration register 1	114_H	P	SV, P	Kernel Reset	5379
QSPI0_BACON	Basic configuration register	118_H	P	BE	Kernel Reset	5383
QSPI0_ECONz (z=0-7)	Configuration extension z	120_H+z*4	P	SV, P	Kernel Reset	5385
QSPI0_STATUS	Status register	140_H	P	P	Kernel Reset	5387
QSPI0_STATUS1	Status register 1	144_H	P	P	Kernel Reset	5390

(table continues...)

Table 1247 (continued) Register overview - QSPI0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
QSPI0_SSOC	Slave select output control register	148_H	P	SV, P	Kernel Reset	5391
QSPI0_FLAGSCLEAR	Flags clear register	150_H	P	P	Kernel Reset	5392
QSPI0_XXLCON	Extra large data configuration register	154_H	P	P	Kernel Reset	5393
QSPI0_MIXENTRY	MIX_ENTRY register	158_H	P	P	Kernel Reset	5394
QSPI0_BACONENTRY	BACON_ENTRY register	$15C_H$	P	P	Kernel Reset	5394
QSPI0_DATAENTRYx (x=0-7)	DATA_ENTRY register x	160_H+x*4	P	P	Kernel Reset	5397
QSPI0_RXEXIT	RX_EXIT register	180_H	P	BE	Kernel Reset	5397
QSPI0_RXEXIT# (#=0-7)	RX_EXIT register mirror #	180_H+#*4	P	BE	Kernel Reset	5397
QSPI0_RXEXITD	RX_EXIT debug register	$1A0_H$	P	BE	Kernel Reset	5398
QSPI0_MC	Move counter register	$1B4_H$	P	P	Kernel Reset	5398
QSPI0_MCCON	Move counter control register	$1B8_H$	P	P	Kernel Reset	5399
QSPI0_MSSMUX	Multiple slave select register	$1BC_H$	P	P	Kernel Reset	5400

30.8.4.4 Register overview - QSPI1 (ascending offset address)

Table 1248 Register overview - QSPI1 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
QSPI1_CLC	Clock control register	000_H	P	P, SV, E	Application Reset	5364
QSPI1_OCS	OCDS control and status register	004_H	P	SV, P	Debug Reset	5364
QSPI1_ID	Module identification register	008_H	P	BE	PowerOn Reset	5365
QSPI1_RST_CTRLA	Reset control register A	$00C_H$	P	P, SV, E	Application Reset	5366
QSPI1_RST_CTRLB	Reset control register B	010_H	P	P, SV, E	Application Reset	5366
QSPI1_RST_STAT	Reset status register	014_H	P	BE	Application Reset	5367

(table continues...)

Table 1248 (continued) Register overview - QSPI1 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
QSPI1_PROTE	PROT register endinit	020_H	U	SV, PROT	Application Reset	5368
QSPI1_PROTSE	PROT register safe endinit	024_H	U	SV, PROT	Application Reset	5369
QSPI1_ACCEN_WRA	Write access enable register A	040_H	U	SE, SV	Application Reset	5371
QSPI1_ACCEN_WRB	Write access enable register B	044_H	U	SE, SV	Application Reset	5372
QSPI1_ACCEN_RDA	Read access enable register A	048_H	U	SE, SV	Application Reset	5372
QSPI1_ACCEN_RDB	Read access enable register B	$04C_H$	U	SE, SV	Application Reset	5373
QSPI1_ACCEN_VM	VM access enable register	050_H	U	SE, SV	Application Reset	5373
QSPI1_ACCEN_PRS	PRS access enable register	054_H	U	SE, SV	Application Reset	5374
QSPI1_PISEL	Port input select register	104_H	P	SV, P	Kernel Reset	5374
QSPI1_GLOBALCON	Global configuration register	110_H	P	SV, P	Kernel Reset	5376
QSPI1_GLOBALCON1	Global configuration register 1	114_H	P	SV, P	Kernel Reset	5379
QSPI1_BACON	Basic configuration register	118_H	P	BE	Kernel Reset	5383
QSPI1_ECONz (z=0-7)	Configuration extension z	120_H+z*4	P	SV, P	Kernel Reset	5385
QSPI1_STATUS	Status register	140_H	P	P	Kernel Reset	5387
QSPI1_STATUS1	Status register 1	144_H	P	P	Kernel Reset	5390
QSPI1_SSOC	Slave select output control register	148_H	P	SV, P	Kernel Reset	5391
QSPI1_FLAGSCLEAR	Flags clear register	150_H	P	P	Kernel Reset	5392
QSPI1_XXLCON	Extra large data configuration register	154_H	P	P	Kernel Reset	5393
QSPI1_MIXENTRY	MIX_ENTRY register	158_H	P	P	Kernel Reset	5394
QSPI1_BACONENTRY	BACON_ENTRY register	$15C_H$	P	P	Kernel Reset	5394
QSPI1_DATAENTRYx (x=0-7)	DATA_ENTRY register x	160_H+x*4	P	P	Kernel Reset	5397
QSPI1_RXEXIT	RX_EXIT register	180_H	P	BE	Kernel Reset	5397

(table continues...)

Table 1248 (continued) Register overview - QSPI1 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
QSPI1_RXEXIT# (#=0-7)	RX_EXIT register mirror #	180_H+#* 4	P	BE	Kernel Reset	5397
QSPI1_RXEXITD	RX_EXIT debug register	$1A0_H$	P	BE	Kernel Reset	5398
QSPI1_MC	Move counter register	$1B4_H$	P	P	Kernel Reset	5398
QSPI1_MCCON	Move counter control register	$1B8_H$	P	P	Kernel Reset	5399
QSPI1_MSSMUX	Multiple slave select register	$1BC_H$	P	P	Kernel Reset	5400

30.8.4.5 Register overview - QSPI2 (ascending offset address)

Table 1249 Register overview - QSPI2 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
QSPI2_CLC	Clock control register	000_H	P	P, SV, E	Application Reset	5364
QSPI2_OCS	OCDS control and status register	004_H	P	SV, P	Debug Reset	5364
QSPI2_ID	Module identification register	008_H	P	BE	PowerOn Reset	5365
QSPI2_RST_CTRLA	Reset control register A	$00C_H$	P	P, SV, E	Application Reset	5366
QSPI2_RST_CTRLB	Reset control register B	010_H	P	P, SV, E	Application Reset	5366
QSPI2_RST_STAT	Reset status register	014_H	P	BE	Application Reset	5367
QSPI2_PROTE	PROT register endinit	020_H	U	SV, PROT	Application Reset	5368
QSPI2_PROTSE	PROT register safe endinit	024_H	U	SV, PROT	Application Reset	5369
QSPI2_ACCEN_WRA	Write access enable register A	040_H	U	SE, SV	Application Reset	5371
QSPI2_ACCEN_WRB	Write access enable register B	044_H	U	SE, SV	Application Reset	5372
QSPI2_ACCEN_RDA	Read access enable register A	048_H	U	SE, SV	Application Reset	5372
QSPI2_ACCEN_RDB	Read access enable register B	$04C_H$	U	SE, SV	Application Reset	5373
QSPI2_ACCEN_VM	VM access enable register	050_H	U	SE, SV	Application Reset	5373

(table continues...)

Table 1249 (continued) Register overview - QSPI2 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
QSPI2_ACCEN_PRS	PRS access enable register	054_H	U	SE, SV	Application Reset	5374
QSPI2_PISEL	Port input select register	104_H	P	SV, P	Kernel Reset	5374
QSPI2_GLOBALCON	Global configuration register	110_H	P	SV, P	Kernel Reset	5376
QSPI2_GLOBALCON1	Global configuration register 1	114_H	P	SV, P	Kernel Reset	5379
QSPI2_BACON	Basic configuration register	118_H	P	BE	Kernel Reset	5383
QSPI2_ECONz (z=0-7)	Configuration extension z	120_H+z*4	P	SV, P	Kernel Reset	5385
QSPI2_STATUS	Status register	140_H	P	P	Kernel Reset	5387
QSPI2_STATUS1	Status register 1	144_H	P	P	Kernel Reset	5390
QSPI2_SSOC	Slave select output control register	148_H	P	SV, P	Kernel Reset	5391
QSPI2_FLAGSCLEAR	Flags clear register	150_H	P	P	Kernel Reset	5392
QSPI2_XXLCON	Extra large data configuration register	154_H	P	P	Kernel Reset	5393
QSPI2_MIXENTRY	MIX_ENTRY register	158_H	P	P	Kernel Reset	5394
QSPI2_BACONENTRY	BACON_ENTRY register	$15C_H$	P	P	Kernel Reset	5394
QSPI2_DATAENTRYx (x=0-7)	DATA_ENTRY register x	160_H+x*4	P	P	Kernel Reset	5397
QSPI2_RXEXIT	RX_EXIT register	180_H	P	BE	Kernel Reset	5397
QSPI2_RXEXIT# (#=0-7)	RX_EXIT register mirror #	180_H+#*4	P	BE	Kernel Reset	5397
QSPI2_RXEXITD	RX_EXIT debug register	$1A0_H$	P	BE	Kernel Reset	5398
QSPI2_MC	Move counter register	$1B4_H$	P	P	Kernel Reset	5398
QSPI2_MCCON	Move counter control register	$1B8_H$	P	P	Kernel Reset	5399
QSPI2_MSSMUX	Multiple slave select register	$1BC_H$	P	P	Kernel Reset	5400

30.8.4.6 Register overview - QSPI3 (ascending offset address)

Table 1250 Register overview - QSPI3 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
QSPI3_CLC	Clock control register	000_H	P	P, SV, E	Application Reset	5364
QSPI3_OCS	OCDS control and status register	004_H	P	SV, P	Debug Reset	5364
QSPI3_ID	Module identification register	008_H	P	BE	PowerOn Reset	5365
QSPI3_RST_CTRLA	Reset control register A	$00C_H$	P	P, SV, E	Application Reset	5366
QSPI3_RST_CTRLB	Reset control register B	010_H	P	P, SV, E	Application Reset	5366
QSPI3_RST_STAT	Reset status register	014_H	P	BE	Application Reset	5367
QSPI3_PROTE	PROT register endinit	020_H	U	SV, PROT	Application Reset	5368
QSPI3_PROTSE	PROT register safe endinit	024_H	U	SV, PROT	Application Reset	5369
QSPI3_ACCEN_WRA	Write access enable register A	040_H	U	SE, SV	Application Reset	5371
QSPI3_ACCEN_WRB	Write access enable register B	044_H	U	SE, SV	Application Reset	5372
QSPI3_ACCEN_RDA	Read access enable register A	048_H	U	SE, SV	Application Reset	5372
QSPI3_ACCEN_RDB	Read access enable register B	$04C_H$	U	SE, SV	Application Reset	5373
QSPI3_ACCEN_VM	VM access enable register	050_H	U	SE, SV	Application Reset	5373
QSPI3_ACCEN_PRS	PRS access enable register	054_H	U	SE, SV	Application Reset	5374
QSPI3_PISEL	Port input select register	104_H	P	SV, P	Kernel Reset	5374
QSPI3_GLOBALCON	Global configuration register	110_H	P	SV, P	Kernel Reset	5376
QSPI3_GLOBALCON1	Global configuration register 1	114_H	P	SV, P	Kernel Reset	5379
QSPI3_BACON	Basic configuration register	118_H	P	BE	Kernel Reset	5383
QSPI3_ECONz (z=0-7)	Configuration extension z	120_H+z*4	P	SV, P	Kernel Reset	5385
QSPI3_STATUS	Status register	140_H	P	P	Kernel Reset	5387
QSPI3_STATUS1	Status register 1	144_H	P	P	Kernel Reset	5390

(table continues...)

Table 1250 (continued) Register overview - QSPI3 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
QSPI3_SSOC	Slave select output control register	148_H	P	SV, P	Kernel Reset	5391
QSPI3_FLAGSCLEAR	Flags clear register	150_H	P	P	Kernel Reset	5392
QSPI3_XXLCON	Extra large data configuration register	154_H	P	P	Kernel Reset	5393
QSPI3_MIXENTRY	MIX_ENTRY register	158_H	P	P	Kernel Reset	5394
QSPI3_BACONENTRY	BACON_ENTRY register	$15C_H$	P	P	Kernel Reset	5394
QSPI3_DATAENTRYx (x=0-7)	DATA_ENTRY register x	160_H+x*4	P	P	Kernel Reset	5397
QSPI3_RXEXIT	RX_EXIT register	180_H	P	BE	Kernel Reset	5397
QSPI3_RXEXIT# (#=0-7)	RX_EXIT register mirror #	180_H+#*4	P	BE	Kernel Reset	5397
QSPI3_RXEXITD	RX_EXIT debug register	$1A0_H$	P	BE	Kernel Reset	5398
QSPI3_MC	Move counter register	$1B4_H$	P	P	Kernel Reset	5398
QSPI3_MCCON	Move counter control register	$1B8_H$	P	P	Kernel Reset	5399
QSPI3_MSSMUX	Multiple slave select register	$1BC_H$	P	P	Kernel Reset	5400

30.8.4.7 Register overview - QSPI4 (ascending offset address)

Table 1251 Register overview - QSPI4 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
QSPI4_CLC	Clock control register	000_H	P	P, SV, E	Application Reset	5364
QSPI4_OCS	OCDS control and status register	004_H	P	SV, P	Debug Reset	5364
QSPI4_ID	Module identification register	008_H	P	BE	PowerOn Reset	5365
QSPI4_RST_CTRLA	Reset control register A	$00C_H$	P	P, SV, E	Application Reset	5366
QSPI4_RST_CTRLB	Reset control register B	010_H	P	P, SV, E	Application Reset	5366
QSPI4_RST_STAT	Reset status register	014_H	P	BE	Application Reset	5367

(table continues...)

Table 1251 (continued) Register overview - QSPI4 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
QSPI4_PROTE	PROT register endinit	020_H	U	SV, PROT	Application Reset	5368
QSPI4_PROTSE	PROT register safe endinit	024_H	U	SV, PROT	Application Reset	5369
QSPI4_ACCEN_WRA	Write access enable register A	040_H	U	SE, SV	Application Reset	5371
QSPI4_ACCEN_WRB	Write access enable register B	044_H	U	SE, SV	Application Reset	5372
QSPI4_ACCEN_RDA	Read access enable register A	048_H	U	SE, SV	Application Reset	5372
QSPI4_ACCEN_RDB	Read access enable register B	$04C_H$	U	SE, SV	Application Reset	5373
QSPI4_ACCEN_VM	VM access enable register	050_H	U	SE, SV	Application Reset	5373
QSPI4_ACCEN_PRS	PRS access enable register	054_H	U	SE, SV	Application Reset	5374
QSPI4_PISEL	Port input select register	104_H	P	SV, P	Kernel Reset	5374
QSPI4_GLOBALCON	Global configuration register	110_H	P	SV, P	Kernel Reset	5376
QSPI4_GLOBALCON1	Global configuration register 1	114_H	P	SV, P	Kernel Reset	5379
QSPI4_BACON	Basic configuration register	118_H	P	BE	Kernel Reset	5383
QSPI4_ECONz (z=0-7)	Configuration extension z	120_H+z*4	P	SV, P	Kernel Reset	5385
QSPI4_STATUS	Status register	140_H	P	P	Kernel Reset	5387
QSPI4_STATUS1	Status register 1	144_H	P	P	Kernel Reset	5390
QSPI4_SSOC	Slave select output control register	148_H	P	SV, P	Kernel Reset	5391
QSPI4_FLAGSCLEAR	Flags clear register	150_H	P	P	Kernel Reset	5392
QSPI4_XXLCON	Extra large data configuration register	154_H	P	P	Kernel Reset	5393
QSPI4_MIXENTRY	MIX_ENTRY register	158_H	P	P	Kernel Reset	5394
QSPI4_BACONENTRY	BACON_ENTRY register	$15C_H$	P	P	Kernel Reset	5394
QSPI4_DATAENTRYx (x=0-7)	DATA_ENTRY register x	160_H+x*4	P	P	Kernel Reset	5397
QSPI4_RXEXIT	RX_EXIT register	180_H	P	BE	Kernel Reset	5397

(table continues...)

Table 1251 (continued) Register overview - QSPI4 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
QSPI4_RXEXIT# (#=0-7)	RX_EXIT register mirror #	180_H+#* 4	P	BE	Kernel Reset	5397
QSPI4_RXEXITD	RX_EXIT debug register	$1A0_H$	P	BE	Kernel Reset	5398
QSPI4_MC	Move counter register	$1B4_H$	P	P	Kernel Reset	5398
QSPI4_MCCON	Move counter control register	$1B8_H$	P	P	Kernel Reset	5399
QSPI4_MSSMUX	Multiple slave select register	$1BC_H$	P	P	Kernel Reset	5400

30.8.4.8 Register overview - QSPI5 (ascending offset address)

Table 1252 Register overview - QSPI5 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
QSPI5_CLC	Clock control register	000_H	P	P, SV, E	Application Reset	5364
QSPI5_OCS	OCDS control and status register	004_H	P	SV, P	Debug Reset	5364
QSPI5_ID	Module identification register	008_H	P	BE	PowerOn Reset	5365
QSPI5_RST_CTRLA	Reset control register A	$00C_H$	P	P, SV, E	Application Reset	5366
QSPI5_RST_CTRLB	Reset control register B	010_H	P	P, SV, E	Application Reset	5366
QSPI5_RST_STAT	Reset status register	014_H	P	BE	Application Reset	5367
QSPI5_PROTE	PROT register endinit	020_H	U	SV, PROT	Application Reset	5368
QSPI5_PROTSE	PROT register safe endinit	024_H	U	SV, PROT	Application Reset	5369
QSPI5_ACCEN_WRA	Write access enable register A	040_H	U	SE, SV	Application Reset	5371
QSPI5_ACCEN_WRB	Write access enable register B	044_H	U	SE, SV	Application Reset	5372
QSPI5_ACCEN_RDA	Read access enable register A	048_H	U	SE, SV	Application Reset	5372
QSPI5_ACCEN_RDB	Read access enable register B	$04C_H$	U	SE, SV	Application Reset	5373
QSPI5_ACCEN_VM	VM access enable register	050_H	U	SE, SV	Application Reset	5373

(table continues...)

Table 1252 (continued) Register overview - QSPI5 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
QSPI5_ACCEN_PRS	PRS access enable register	054_H	U	SE, SV	Application Reset	5374
QSPI5_PISEL	Port input select register	104_H	P	SV, P	Kernel Reset	5374
QSPI5_GLOBALCON	Global configuration register	110_H	P	SV, P	Kernel Reset	5376
QSPI5_GLOBALCON1	Global configuration register 1	114_H	P	SV, P	Kernel Reset	5379
QSPI5_BACON	Basic configuration register	118_H	P	BE	Kernel Reset	5383
QSPI5_ECONz (z=0-7)	Configuration extension z	120_H+z*4	P	SV, P	Kernel Reset	5385
QSPI5_STATUS	Status register	140_H	P	P	Kernel Reset	5387
QSPI5_STATUS1	Status register 1	144_H	P	P	Kernel Reset	5390
QSPI5_SSOC	Slave select output control register	148_H	P	SV, P	Kernel Reset	5391
QSPI5_FLAGSCLEAR	Flags clear register	150_H	P	P	Kernel Reset	5392
QSPI5_XXLCON	Extra large data configuration register	154_H	P	P	Kernel Reset	5393
QSPI5_MIXENTRY	MIX_ENTRY register	158_H	P	P	Kernel Reset	5394
QSPI5_BACONENTRY	BACON_ENTRY register	$15C_H$	P	P	Kernel Reset	5394
QSPI5_DATAENTRYx (x=0-7)	DATA_ENTRY register x	160_H+x*4	P	P	Kernel Reset	5397
QSPI5_RXEXIT	RX_EXIT register	180_H	P	BE	Kernel Reset	5397
QSPI5_RXEXIT# (#=0-7)	RX_EXIT register mirror #	180_H+#*4	P	BE	Kernel Reset	5397
QSPI5_RXEXITD	RX_EXIT debug register	$1A0_H$	P	BE	Kernel Reset	5398
QSPI5_MC	Move counter register	$1B4_H$	P	P	Kernel Reset	5398
QSPI5_MCCON	Move counter control register	$1B8_H$	P	P	Kernel Reset	5399
QSPI5_MSSMUX	Multiple slave select register	$1BC_H$	P	P	Kernel Reset	5400

30.8.4.9 Register overview - QSPI6 (ascending offset address)

Table 1253 Register overview - QSPI6 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
QSPI6_CLC	Clock control register	000_H	P	P, SV, E	Application Reset	5364
QSPI6_OCS	OCDS control and status register	004_H	P	SV, P	Debug Reset	5364
QSPI6_ID	Module identification register	008_H	P	BE	PowerOn Reset	5365
QSPI6_RST_CTRLA	Reset control register A	$00C_H$	P	P, SV, E	Application Reset	5366
QSPI6_RST_CTRLB	Reset control register B	010_H	P	P, SV, E	Application Reset	5366
QSPI6_RST_STAT	Reset status register	014_H	P	BE	Application Reset	5367
QSPI6_PROTE	PROT register endinit	020_H	U	SV, PROT	Application Reset	5368
QSPI6_PROTSE	PROT register safe endinit	024_H	U	SV, PROT	Application Reset	5369
QSPI6_ACCEN_WRA	Write access enable register A	040_H	U	SE, SV	Application Reset	5371
QSPI6_ACCEN_WRB	Write access enable register B	044_H	U	SE, SV	Application Reset	5372
QSPI6_ACCEN_RDA	Read access enable register A	048_H	U	SE, SV	Application Reset	5372
QSPI6_ACCEN_RDB	Read access enable register B	$04C_H$	U	SE, SV	Application Reset	5373
QSPI6_ACCEN_VM	VM access enable register	050_H	U	SE, SV	Application Reset	5373
QSPI6_ACCEN_PRS	PRS access enable register	054_H	U	SE, SV	Application Reset	5374
QSPI6_PISEL	Port input select register	104_H	P	SV, P	Kernel Reset	5374
QSPI6_GLOBALCON	Global configuration register	110_H	P	SV, P	Kernel Reset	5376
QSPI6_GLOBALCON1	Global configuration register 1	114_H	P	SV, P	Kernel Reset	5379
QSPI6_BACON	Basic configuration register	118_H	P	BE	Kernel Reset	5383
QSPI6_ECONz (z=0-7)	Configuration extension z	120_H+z*4	P	SV, P	Kernel Reset	5385
QSPI6_STATUS	Status register	140_H	P	P	Kernel Reset	5387
QSPI6_STATUS1	Status register 1	144_H	P	P	Kernel Reset	5390

(table continues...)

Table 1253 (continued) Register overview - QSPI6 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
QSPI6_SSOC	Slave select output control register	148_H	P	SV, P	Kernel Reset	5391
QSPI6_FLAGSCLEAR	Flags clear register	150_H	P	P	Kernel Reset	5392
QSPI6_XXLCON	Extra large data configuration register	154_H	P	P	Kernel Reset	5393
QSPI6_MIXENTRY	MIX_ENTRY register	158_H	P	P	Kernel Reset	5394
QSPI6_BACONENTRY	BACON_ENTRY register	$15C_H$	P	P	Kernel Reset	5394
QSPI6_DATAENTRYx (x=0-7)	DATA_ENTRY register x	160_H+x*4	P	P	Kernel Reset	5397
QSPI6_RXEXIT	RX_EXIT register	180_H	P	BE	Kernel Reset	5397
QSPI6_RXEXIT# (#=0-7)	RX_EXIT register mirror #	180_H+#*4	P	BE	Kernel Reset	5397
QSPI6_RXEXITD	RX_EXIT debug register	$1A0_H$	P	BE	Kernel Reset	5398
QSPI6_MC	Move counter register	$1B4_H$	P	P	Kernel Reset	5398
QSPI6_MCCON	Move counter control register	$1B8_H$	P	P	Kernel Reset	5399
QSPI6_MSSMUX	Multiple slave select register	$1BC_H$	P	P	Kernel Reset	5400

30.8.4.10 Register overview - QSPI7 (ascending offset address)

Table 1254 Register overview - QSPI7 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
QSPI7_CLC	Clock control register	000_H	P	P, SV, E	Application Reset	5364
QSPI7_OCS	OCDS control and status register	004_H	P	SV, P	Debug Reset	5364
QSPI7_ID	Module identification register	008_H	P	BE	PowerOn Reset	5365
QSPI7_RST_CTRLA	Reset control register A	$00C_H$	P	P, SV, E	Application Reset	5366
QSPI7_RST_CTRLB	Reset control register B	010_H	P	P, SV, E	Application Reset	5366
QSPI7_RST_STAT	Reset status register	014_H	P	BE	Application Reset	5367

(table continues...)

Table 1254 (continued) Register overview - QSPI7 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
QSPI7_PROTE	PROT register endinit	020_H	U	SV, PROT	Application Reset	5368
QSPI7_PROTSE	PROT register safe endinit	024_H	U	SV, PROT	Application Reset	5369
QSPI7_ACCEN_WRA	Write access enable register A	040_H	U	SE, SV	Application Reset	5371
QSPI7_ACCEN_WRB	Write access enable register B	044_H	U	SE, SV	Application Reset	5372
QSPI7_ACCEN_RDA	Read access enable register A	048_H	U	SE, SV	Application Reset	5372
QSPI7_ACCEN_RDB	Read access enable register B	$04C_H$	U	SE, SV	Application Reset	5373
QSPI7_ACCEN_VM	VM access enable register	050_H	U	SE, SV	Application Reset	5373
QSPI7_ACCEN_PRS	PRS access enable register	054_H	U	SE, SV	Application Reset	5374
QSPI7_PISEL	Port input select register	104_H	P	SV, P	Kernel Reset	5374
QSPI7_GLOBALCON	Global configuration register	110_H	P	SV, P	Kernel Reset	5376
QSPI7_GLOBALCON1	Global configuration register 1	114_H	P	SV, P	Kernel Reset	5379
QSPI7_BACON	Basic configuration register	118_H	P	BE	Kernel Reset	5383
QSPI7_ECONz (z=0-7)	Configuration extension z	120_H+z*4	P	SV, P	Kernel Reset	5385
QSPI7_STATUS	Status register	140_H	P	P	Kernel Reset	5387
QSPI7_STATUS1	Status register 1	144_H	P	P	Kernel Reset	5390
QSPI7_SSOC	Slave select output control register	148_H	P	SV, P	Kernel Reset	5391
QSPI7_FLAGSCLEAR	Flags clear register	150_H	P	P	Kernel Reset	5392
QSPI7_XXLCON	Extra large data configuration register	154_H	P	P	Kernel Reset	5393
QSPI7_MIXENTRY	MIX_ENTRY register	158_H	P	P	Kernel Reset	5394
QSPI7_BACONENTRY	BACON_ENTRY register	$15C_H$	P	P	Kernel Reset	5394
QSPI7_DATAENTRYx (x=0-7)	DATA_ENTRY register x	160_H+x*4	P	P	Kernel Reset	5397
QSPI7_RXEXIT	RX_EXIT register	180_H	P	BE	Kernel Reset	5397

(table continues...)

Table 1254 (continued) Register overview - QSPI7 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
QSPI7_RXEXIT# (#=0-7)	RX_EXIT register mirror #	180_H+#* 4	P	BE	Kernel Reset	5397
QSPI7_RXEXITD	RX_EXIT debug register	$1A0_H$	P	BE	Kernel Reset	5398
QSPI7_MC	Move counter register	$1B4_H$	P	P	Kernel Reset	5398
QSPI7_MCCON	Move counter control register	$1B8_H$	P	P	Kernel Reset	5399
QSPI7_MSSMUX	Multiple slave select register	$1BC_H$	P	P	Kernel Reset	5400

30.8.4.11 Device specific registers

There are no device specific register changes.

30.8.5 TC4Dx QSPI connectivity

The table below lists interfaces to and from this functional block to other blocks in the device.

Table 1255 List of QSPI interface signals

Interface signals	I/O	Description
CLOCK_QSPI_fSPB	In	SPB clock input
CLOCK_QSPI_fQSPI	In	QSPI clock input (fQSPI is referred as fPER in QSPI chapter description)
FPI_QSPI_SIF	In	QSPI slave interface
QSPI0_IR_RX_INT	Out	QSPI0 Receive Service Request
QSPI1_IR_RX_INT	Out	QSPI1 Receive Service Request
QSPI2_IR_RX_INT	Out	QSPI2 Receive Service Request
QSPI3_IR_RX_INT	Out	QSPI3 Receive Service Request
QSPI4_IR_RX_INT	Out	QSPI4 Receive Service Request
QSPI5_IR_RX_INT	Out	QSPI5 Receive Service Request
QSPI6_IR_RX_INT	Out	QSPI6 Receive Service Request
QSPI7_IR_RX_INT	Out	QSPI7 Receive Service Request
QSPI0_IR_TX_INT	Out	QSPI0 Transmit Service Request
QSPI1_IR_TX_INT	Out	QSPI1 Transmit Service Request
QSPI2_IR_TX_INT	Out	QSPI2 Transmit Service Request
QSPI3_IR_TX_INT	Out	QSPI3 Transmit Service Request
QSPI4_IR_TX_INT	Out	QSPI4 Transmit Service Request
QSPI5_IR_TX_INT	Out	QSPI5 Transmit Service Request
QSPI6_IR_TX_INT	Out	QSPI6 Transmit Service Request
QSPI7_IR_TX_INT	Out	QSPI7 Transmit Service Request
QSPI0_IR_ERR_INT	Out	QSPI0 Error Service Request
QSPI1_IR_ERR_INT	Out	QSPI1 Error Service Request
QSPI2_IR_ERR_INT	Out	QSPI2 Error Service Request
QSPI3_IR_ERR_INT	Out	QSPI3 Error Service Request
QSPI4_IR_ERR_INT	Out	QSPI4 Error Service Request
QSPI5_IR_ERR_INT	Out	QSPI5 Error Service Request
QSPI6_IR_ERR_INT	Out	QSPI6 Error Service Request
QSPI7_IR_ERR_INT	Out	QSPI7 Error Service Request
QSPI0_IR_PT_INT	Out	QSPI0 Phase Transition Service Request

(table continues...)

Table 1255 (continued) List of QSPI interface signals

Interface signals	I/O	Description
QSPI1_IR_PT_INT	Out	QSPI1 Phase Transition Service Request
QSPI2_IR_PT_INT	Out	QSPI2 Phase Transition Service Request
QSPI3_IR_PT_INT	Out	QSPI3 Phase Transition Service Request
QSPI4_IR_PT_INT	Out	QSPI4 Phase Transition Service Request
QSPI5_IR_PT_INT	Out	QSPI5 Phase Transition Service Request
QSPI6_IR_PT_INT	Out	QSPI6 Phase Transition Service Request
QSPI7_IR_PT_INT	Out	QSPI7 Phase Transition Service Request
QSPI0_IR_U_INT	Out	QSPI0 User Defined Service Request
QSPI1_IR_U_INT	Out	QSPI1 User Defined Service Request
QSPI2_IR_U_INT	Out	QSPI2 User Defined Service Request
QSPI3_IR_U_INT	Out	QSPI3 User Defined Service Request
QSPI4_IR_U_INT	Out	QSPI4 User Defined Service Request
QSPI5_IR_U_INT	Out	QSPI5 User Defined Service Request
QSPI6_IR_U_INT	Out	QSPI6 User Defined Service Request
QSPI7_IR_U_INT	Out	QSPI7 User Defined Service Request
EGTM_QSPI0_TRIG	In	eGTM Trigger Input
EGTM_QSPI1_TRIG	In	eGTM Trigger Input
EGTM_QSPI2_TRIG	In	eGTM Trigger Input
EGTM_QSPI3_TRIG	In	eGTM Trigger Input
EGTM_QSPI4_TRIG	In	eGTM Trigger Input
EGTM_QSPI5_TRIG	In	eGTM Trigger Input
EGTM_QSPI6_TRIG	In	eGTM Trigger Input
EGTM_QSPI7_TRIG	In	eGTM Trigger Input
PORTS_QSPI0_MRSTA	In	Master SPI data input
PORTS_QSPI1_MRSTA	In	Master SPI data input
PORTS_QSPI2_MRSTA	In	Master SPI data input
PORTS_QSPI3_MRSTA	In	Master SPI data input
PORTS_QSPI4_MRSTA	In	Master SPI data input
PORTS_QSPI5_MRSTA	In	Master SPI data input
PORTS_QSPI6_MRSTA	In	Master SPI data input
PORTS_QSPI7_MRSTA	In	Master SPI data input
PORTS_QSPI0_MRSTB	In	Master SPI data input
PORTS_QSPI1_MRSTB	In	Master SPI data input

(table continues...)

Table 1255 (continued) List of QSPI interface signals

Interface signals	I/O	Description
PORTS_QSPI2_MRSTB	In	Master SPI data input
PORTS_QSPI3_MRSTB	In	Master SPI data input
PORTS_QSPI4_MRSTB	In	Master SPI data input
PORTS_QSPI5_MRSTB	In	Master SPI data input
PORTS_QSPI6_MRSTB	In	Master SPI data input
PORTS_QSPI7_MRSTB	In	Master SPI data input
PORTS_QSPI0_MRSTC	In	Master SPI data input
PORTS_QSPI1_MRSTC	In	Master SPI data input
PORTS_QSPI2_MRSTC	In	Master SPI data input
PORTS_QSPI3_MRSTC	In	Master SPI data input
PORTS_QSPI4_MRSTC	In	Master SPI data input
PORTS_QSPI5_MRSTC	In	Master SPI data input
PORTS_QSPI6_MRSTC	In	Master SPI data input
PORTS_QSPI7_MRSTC	In	Master SPI data input
PORTS_QSPI0_MRSTD	In	Master SPI data input
PORTS_QSPI1_MRSTD	In	Master SPI data input
PORTS_QSPI2_MRSTD	In	Master SPI data input
PORTS_QSPI3_MRSTD	In	Master SPI data input
PORTS_QSPI4_MRSTD	In	Master SPI data input
PORTS_QSPI5_MRSTD	In	Master SPI data input
PORTS_QSPI6_MRSTD	In	Master SPI data input
PORTS_QSPI7_MRSTD	In	Master SPI data input
PORTS_QSPI0_MRSTE	In	Master SPI data input
PORTS_QSPI1_MRSTE	In	Master SPI data input
PORTS_QSPI2_MRSTE	In	Master SPI data input
PORTS_QSPI3_MRSTE	In	Master SPI data input
PORTS_QSPI4_MRSTE	In	Master SPI data input
PORTS_QSPI5_MRSTE	In	Master SPI data input
PORTS_QSPI6_MRSTE	In	Master SPI data input
PORTS_QSPI7_MRSTE	In	Master SPI data input
PORTS_QSPI0_MRSTF	In	Master SPI data input
PORTS_QSPI1_MRSTF	In	Master SPI data input
PORTS_QSPI2_MRSTF	In	Master SPI data input

(table continues...)

Table 1255 (continued) List of QSPI interface signals

Interface signals	I/O	Description
PORTS_QSPI3_MRSTF	In	Master SPI data input
PORTS_QSPI4_MRSTF	In	Master SPI data input
PORTS_QSPI5_MRSTF	In	Master SPI data input
PORTS_QSPI6_MRSTF	In	Master SPI data input
PORTS_QSPI7_MRSTF	In	Master SPI data input
PORTS_QSPI0_MRSTG	In	Master SPI data input
PORTS_QSPI1_MRSTG	In	Master SPI data input
PORTS_QSPI2_MRSTG	In	Master SPI data input
PORTS_QSPI3_MRSTG	In	Master SPI data input
PORTS_QSPI4_MRSTG	In	Master SPI data input
PORTS_QSPI5_MRSTG	In	Master SPI data input
PORTS_QSPI6_MRSTG	In	Master SPI data input
PORTS_QSPI7_MRSTG	In	Master SPI data input
PORTS_QSPI0_MRSTH	In	Master SPI data input
PORTS_QSPI1_MRSTH	In	Master SPI data input
PORTS_QSPI2_MRSTH	In	Master SPI data input
PORTS_QSPI3_MRSTH	In	Master SPI data input
PORTS_QSPI4_MRSTH	In	Master SPI data input
PORTS_QSPI5_MRSTH	In	Master SPI data input
PORTS_QSPI6_MRSTH	In	Master SPI data input
PORTS_QSPI7_MRSTH	In	Master SPI data input
PORTS_QSPI0_MTSRA	In	Slave SPI data input
PORTS_QSPI1_MTSRA	In	Slave SPI data input
PORTS_QSPI2_MTSRA	In	Slave SPI data input
PORTS_QSPI3_MTSRA	In	Slave SPI data input
PORTS_QSPI4_MTSRA	In	Slave SPI data input
PORTS_QSPI5_MTSRA	In	Slave SPI data input
PORTS_QSPI6_MTSRA	In	Slave SPI data input
PORTS_QSPI7_MTSRA	In	Slave SPI data input
PORTS_QSPI0_MTSRB	In	Slave SPI data input
PORTS_QSPI1_MTSRB	In	Slave SPI data input
PORTS_QSPI2_MTSRB	In	Slave SPI data input
PORTS_QSPI3_MTSRB	In	Slave SPI data input

(table continues...)

Table 1255 (continued) List of QSPI interface signals

Interface signals	I/O	Description
PORTS_QSPI4_MTSRB	In	Slave SPI data input
PORTS_QSPI5_MTSRB	In	Slave SPI data input
PORTS_QSPI6_MTSRB	In	Slave SPI data input
PORTS_QSPI7_MTSRB	In	Slave SPI data input
PORTS_QSPI0_MTSRC	In	Slave SPI data input
PORTS_QSPI1_MTSRC	In	Slave SPI data input
PORTS_QSPI2_MTSRC	In	Slave SPI data input
PORTS_QSPI3_MTSRC	In	Slave SPI data input
PORTS_QSPI4_MTSRC	In	Slave SPI data input
PORTS_QSPI5_MTSRC	In	Slave SPI data input
PORTS_QSPI6_MTSRC	In	Slave SPI data input
PORTS_QSPI7_MTSRC	In	Slave SPI data input
PORTS_QSPI0_MTSRD	In	Slave SPI data input
PORTS_QSPI1_MTSRD	In	Slave SPI data input
PORTS_QSPI2_MTSRD	In	Slave SPI data input
PORTS_QSPI3_MTSRD	In	Slave SPI data input
PORTS_QSPI4_MTSRD	In	Slave SPI data input
PORTS_QSPI5_MTSRD	In	Slave SPI data input
PORTS_QSPI6_MTSRD	In	Slave SPI data input
PORTS_QSPI7_MTSRD	In	Slave SPI data input
PORTS_QSPI0_MTSRE	In	Slave SPI data input
PORTS_QSPI1_MTSRE	In	Slave SPI data input
PORTS_QSPI2_MTSRE	In	Slave SPI data input
PORTS_QSPI3_MTSRE	In	Slave SPI data input
PORTS_QSPI4_MTSRE	In	Slave SPI data input
PORTS_QSPI5_MTSRE	In	Slave SPI data input
PORTS_QSPI6_MTSRE	In	Slave SPI data input
PORTS_QSPI7_MTSRE	In	Slave SPI data input
PORTS_QSPI0_MTSRF	In	Slave SPI data input
PORTS_QSPI1_MTSRF	In	Slave SPI data input
PORTS_QSPI2_MTSRF	In	Slave SPI data input
PORTS_QSPI3_MTSRF	In	Slave SPI data input
PORTS_QSPI4_MTSRF	In	Slave SPI data input

(table continues...)

Table 1255 (continued) List of QSPI interface signals

Interface signals	I/O	Description
PORTS_QSPI5_MTSRF	In	Slave SPI data input
PORTS_QSPI6_MTSRF	In	Slave SPI data input
PORTS_QSPI7_MTSRF	In	Slave SPI data input
PORTS_QSPI0_MTSRG	In	Slave SPI data input
PORTS_QSPI1_MTSRG	In	Slave SPI data input
PORTS_QSPI2_MTSRG	In	Slave SPI data input
PORTS_QSPI3_MTSRG	In	Slave SPI data input
PORTS_QSPI4_MTSRG	In	Slave SPI data input
PORTS_QSPI5_MTSRG	In	Slave SPI data input
PORTS_QSPI6_MTSRG	In	Slave SPI data input
PORTS_QSPI7_MTSRG	In	Slave SPI data input
PORTS_QSPI0_MTSRH	In	Slave SPI data input
PORTS_QSPI1_MTSRH	In	Slave SPI data input
PORTS_QSPI2_MTSRH	In	Slave SPI data input
PORTS_QSPI3_MTSRH	In	Slave SPI data input
PORTS_QSPI4_MTSRH	In	Slave SPI data input
PORTS_QSPI5_MTSRH	In	Slave SPI data input
PORTS_QSPI6_MTSRH	In	Slave SPI data input
PORTS_QSPI7_MTSRH	In	Slave SPI data input
PORTS_QSPI0_SCLKA	In	Slave SPI clock inputs
PORTS_QSPI1_SCLKA	In	Slave SPI clock inputs
PORTS_QSPI2_SCLKA	In	Slave SPI clock inputs
PORTS_QSPI3_SCLKA	In	Slave SPI clock inputs
PORTS_QSPI4_SCLKA	In	Slave SPI clock inputs
PORTS_QSPI5_SCLKA	In	Slave SPI clock inputs
PORTS_QSPI6_SCLKA	In	Slave SPI clock inputs
PORTS_QSPI7_SCLKA	In	Slave SPI clock inputs
PORTS_QSPI0_SCLKB	In	Slave SPI clock inputs
PORTS_QSPI1_SCLKB	In	Slave SPI clock inputs
PORTS_QSPI2_SCLKB	In	Slave SPI clock inputs
PORTS_QSPI3_SCLKB	In	Slave SPI clock inputs
PORTS_QSPI4_SCLKB	In	Slave SPI clock inputs
PORTS_QSPI5_SCLKB	In	Slave SPI clock inputs

(table continues...)

Table 1255 (continued) List of QSPI interface signals

Interface signals	I/O	Description
PORTS_QSPI6_SCLKB	In	Slave SPI clock inputs
PORTS_QSPI7_SCLKB	In	Slave SPI clock inputs
PORTS_QSPI0_SCLKC	In	Slave SPI clock inputs
PORTS_QSPI1_SCLKC	In	Slave SPI clock inputs
PORTS_QSPI2_SCLKC	In	Slave SPI clock inputs
PORTS_QSPI3_SCLKC	In	Slave SPI clock inputs
PORTS_QSPI4_SCLKC	In	Slave SPI clock inputs
PORTS_QSPI5_SCLKC	In	Slave SPI clock inputs
PORTS_QSPI6_SCLKC	In	Slave SPI clock inputs
PORTS_QSPI7_SCLKC	In	Slave SPI clock inputs
PORTS_QSPI0_SCLKD	In	Slave SPI clock inputs
PORTS_QSPI1_SCLKD	In	Slave SPI clock inputs
PORTS_QSPI2_SCLKD	In	Slave SPI clock inputs
PORTS_QSPI3_SCLKD	In	Slave SPI clock inputs
PORTS_QSPI4_SCLKD	In	Slave SPI clock inputs
PORTS_QSPI5_SCLKD	In	Slave SPI clock inputs
PORTS_QSPI6_SCLKD	In	Slave SPI clock inputs
PORTS_QSPI7_SCLKD	In	Slave SPI clock inputs
PORTS_QSPI0_SCLKE	In	Slave SPI clock inputs
PORTS_QSPI1_SCLKE	In	Slave SPI clock inputs
PORTS_QSPI2_SCLKE	In	Slave SPI clock inputs
PORTS_QSPI3_SCLKE	In	Slave SPI clock inputs
PORTS_QSPI4_SCLKE	In	Slave SPI clock inputs
PORTS_QSPI5_SCLKE	In	Slave SPI clock inputs
PORTS_QSPI6_SCLKE	In	Slave SPI clock inputs
PORTS_QSPI7_SCLKE	In	Slave SPI clock inputs
PORTS_QSPI0_SCLKF	In	Slave SPI clock inputs
PORTS_QSPI1_SCLKF	In	Slave SPI clock inputs
PORTS_QSPI2_SCLKF	In	Slave SPI clock inputs
PORTS_QSPI3_SCLKF	In	Slave SPI clock inputs
PORTS_QSPI4_SCLKF	In	Slave SPI clock inputs
PORTS_QSPI5_SCLKF	In	Slave SPI clock inputs
PORTS_QSPI6_SCLKF	In	Slave SPI clock inputs

(table continues...)

Table 1255 (continued) List of QSPI interface signals

Interface signals	I/O	Description
PORTS_QSPI7_SCLKF	In	Slave SPI clock inputs
PORTS_QSPI0_SCLKG	In	Slave SPI clock inputs
PORTS_QSPI1_SCLKG	In	Slave SPI clock inputs
PORTS_QSPI2_SCLKG	In	Slave SPI clock inputs
PORTS_QSPI3_SCLKG	In	Slave SPI clock inputs
PORTS_QSPI4_SCLKG	In	Slave SPI clock inputs
PORTS_QSPI5_SCLKG	In	Slave SPI clock inputs
PORTS_QSPI6_SCLKG	In	Slave SPI clock inputs
PORTS_QSPI7_SCLKG	In	Slave SPI clock inputs
PORTS_QSPI0_SCLKH	In	Slave SPI clock inputs
PORTS_QSPI1_SCLKH	In	Slave SPI clock inputs
PORTS_QSPI2_SCLKH	In	Slave SPI clock inputs
PORTS_QSPI3_SCLKH	In	Slave SPI clock inputs
PORTS_QSPI4_SCLKH	In	Slave SPI clock inputs
PORTS_QSPI5_SCLKH	In	Slave SPI clock inputs
PORTS_QSPI6_SCLKH	In	Slave SPI clock inputs
PORTS_QSPI7_SCLKH	In	Slave SPI clock inputs
PORTS_QSPI0_SLSIA	In	Slave select input
PORTS_QSPI1_SLSIA	In	Slave select input
PORTS_QSPI2_SLSIA	In	Slave select input
PORTS_QSPI3_SLSIA	In	Slave select input
PORTS_QSPI4_SLSIA	In	Slave select input
PORTS_QSPI5_SLSIA	In	Slave select input
PORTS_QSPI6_SLSIA	In	Slave select input
PORTS_QSPI7_SLSIA	In	Slave select input
PORTS_QSPI0_SLSIB	In	Slave select input
PORTS_QSPI1_SLSIB	In	Slave select input
PORTS_QSPI2_SLSIB	In	Slave select input
PORTS_QSPI3_SLSIB	In	Slave select input
PORTS_QSPI4_SLSIB	In	Slave select input
PORTS_QSPI5_SLSIB	In	Slave select input
PORTS_QSPI6_SLSIB	In	Slave select input
PORTS_QSPI7_SLSIB	In	Slave select input

(table continues...)

Table 1255 (continued) List of QSPI interface signals

Interface signals	I/O	Description
PORTS_QSPI0_SLSIC	In	Slave select input
PORTS_QSPI1_SLSIC	In	Slave select input
PORTS_QSPI2_SLSIC	In	Slave select input
PORTS_QSPI3_SLSIC	In	Slave select input
PORTS_QSPI4_SLSIC	In	Slave select input
PORTS_QSPI5_SLSIC	In	Slave select input
PORTS_QSPI6_SLSIC	In	Slave select input
PORTS_QSPI7_SLSIC	In	Slave select input
PORTS_QSPI0_SLSID	In	Slave select input
PORTS_QSPI1_SLSID	In	Slave select input
PORTS_QSPI2_SLSID	In	Slave select input
PORTS_QSPI3_SLSID	In	Slave select input
PORTS_QSPI4_SLSID	In	Slave select input
PORTS_QSPI5_SLSID	In	Slave select input
PORTS_QSPI6_SLSID	In	Slave select input
PORTS_QSPI7_SLSID	In	Slave select input
PORTS_QSPI0_SLSIE	In	Slave select input
PORTS_QSPI1_SLSIE	In	Slave select input
PORTS_QSPI2_SLSIE	In	Slave select input
PORTS_QSPI3_SLSIE	In	Slave select input
PORTS_QSPI4_SLSIE	In	Slave select input
PORTS_QSPI5_SLSIE	In	Slave select input
PORTS_QSPI6_SLSIE	In	Slave select input
PORTS_QSPI7_SLSIE	In	Slave select input
PORTS_QSPI0_SLSIF	In	Slave select input
PORTS_QSPI1_SLSIF	In	Slave select input
PORTS_QSPI2_SLSIF	In	Slave select input
PORTS_QSPI3_SLSIF	In	Slave select input
PORTS_QSPI4_SLSIF	In	Slave select input
PORTS_QSPI5_SLSIF	In	Slave select input
PORTS_QSPI6_SLSIF	In	Slave select input
PORTS_QSPI7_SLSIF	In	Slave select input
PORTS_QSPI0_SLSIG	In	Slave select input

(table continues...)

Table 1255 (continued) List of QSPI interface signals

Interface signals	I/O	Description
PORTS_QSPI1_SLSIG	In	Slave select input
PORTS_QSPI2_SLSIG	In	Slave select input
PORTS_QSPI3_SLSIG	In	Slave select input
PORTS_QSPI4_SLSIG	In	Slave select input
PORTS_QSPI5_SLSIG	In	Slave select input
PORTS_QSPI6_SLSIG	In	Slave select input
PORTS_QSPI7_SLSIG	In	Slave select input
QSPI0_PORTS_MRST	Out	Slave SPI data output
QSPI1_PORTS_MRST	Out	Slave SPI data output
QSPI2_PORTS_MRST	Out	Slave SPI data output
QSPI3_PORTS_MRST	Out	Slave SPI data output
QSPI4_PORTS_MRST	Out	Slave SPI data output
QSPI5_PORTS_MRST	Out	Slave SPI data output
QSPI6_PORTS_MRST	Out	Slave SPI data output
QSPI7_PORTS_MRST	Out	Slave SPI data output
QSPI0_PORTS_MTSR	Out	Master SPI data output
QSPI1_PORTS_MTSR	Out	Master SPI data output
QSPI2_PORTS_MTSR	Out	Master SPI data output
QSPI3_PORTS_MTSR	Out	Master SPI data output
QSPI4_PORTS_MTSR	Out	Master SPI data output
QSPI5_PORTS_MTSR	Out	Master SPI data output
QSPI6_PORTS_MTSR	Out	Master SPI data output
QSPI7_PORTS_MTSR	Out	Master SPI data output
QSPI0_PORTS_SCLK	Out	Master SPI clock output
QSPI1_PORTS_SCLK	Out	Master SPI clock output
QSPI2_PORTS_SCLK	Out	Master SPI clock output
QSPI3_PORTS_SCLK	Out	Master SPI clock output
QSPI4_PORTS_SCLK	Out	Master SPI clock output
QSPI5_PORTS_SCLK	Out	Master SPI clock output
QSPI6_PORTS_SCLK	Out	Master SPI clock output
QSPI7_PORTS_SCLK	Out	Master SPI clock output
QSPI0_PORTS_SLSO[15:0]	Out	Master slave select output
QSPI1_PORTS_SLSO[15:0]	Out	Master slave select output

(table continues...)

Table 1255 (continued) List of QSPI interface signals

Interface signals	I/O	Description
QSPI2_PORTS_SLSO[15:0]	Out	Master slave select output
QSPI3_PORTS_SLSO[15:0]	Out	Master slave select output
QSPI4_PORTS_SLSO[15:0]	Out	Master slave select output
QSPI5_PORTS_SLSO[15:0]	Out	Master slave select output
QSPI6_PORTS_SLSO[15:0]	Out	Master slave select output
QSPI7_PORTS_SLSO[15:0]	Out	Master slave select output

30.8.6 TC4Dx QSPI revision history

Initial release of the chapter.

31 Inter-Integrated Circuit (I2C)

The I2C module is a functional block that offers communication functionality according to the I2C-bus specification version 2.1. I2C is either connected to the SPB bus or the COMPB bus with a slave interface including a set of special function registers (SFRs). The bus interface used by the I2C module is defined in the device specific chapter.

31.1 Feature list

- Compatible with I2C-bus specification version 2.1
- Master mode supported
- Multi-master mode supported
- Slave mode supported
- Different speed ranges available for data transfer:
  - Standard mode up to 100 kbit/s (20 kbit/s - 100 kbit/s)
  - Fast mode up to 400 kbit/s (100 kbit/s - 400 kbit/s)
  - High-speed mode up to 3.4 Mbit/s (500 kbit/s - 3.4 Mbit/s)
- 7-bit and 10-bit I2C-bus addressing supported
- Automatic execution of low-level tasks such as:
  - (De)Serialization of the bus data
  - Generation or detection of start and stop signal
  - Generation or detection of acknowledge signal
  - Bus state detection
  - Bus access arbitration in multi-master mode
  - Recognition of device address in slave mode
  - Configurable detection of general call address
  - Configurable repeated start in master mode
- Flexible clock and timing control:
  - Prescaler for I2C kernel clock (from 0 to 255)
  - Bit rate generation through fractional divider
  - I2C-bus signal timing adjustment
- FIFO for buffering data from or to CPU with following features:
  - 8 FIFO stages based on 32-bit width
  - Configurable data alignment (byte, half word, word)
  - Configurable sizes for burst, transmit and receive package
  - FIFO usable as flow controller (seamless DMA flow)
- Advanced interrupt handling:
  - 4 data transfer interrupt requests (burst, last burst, single, last single) resulting in one single data transfer interrupt
  - Protocol interrupt with 7 sources (address match, general call, master code, arbitration lost, not-acknowledge received, transmission end, receive mode)
  - Error interrupt with 4 sources (overflow or underflow of FIFO for transmit or receive)
- Pretended Networking:
  - High-speed mode, fast mode and standard mode work with the system clock frequency f_{I2C}. that can reach a minimum of 5 MHz

31.2 Functional overview

The I2C module communicates with the external world through a pair of I/O lines. A serial data line (SDA) and a serial clock line (SCL) carry the information between the devices.

The SDA and SCL are connected to a positive supply voltage through pull-up resistors. In quiescent state, when the bus is free, both lines are high. During communication the lines are alternatively pulled to low. The output stages of devices connected to the bus must have an open-drain (CMOS) or open-collector (bipolar) to perform a wired-AND function.

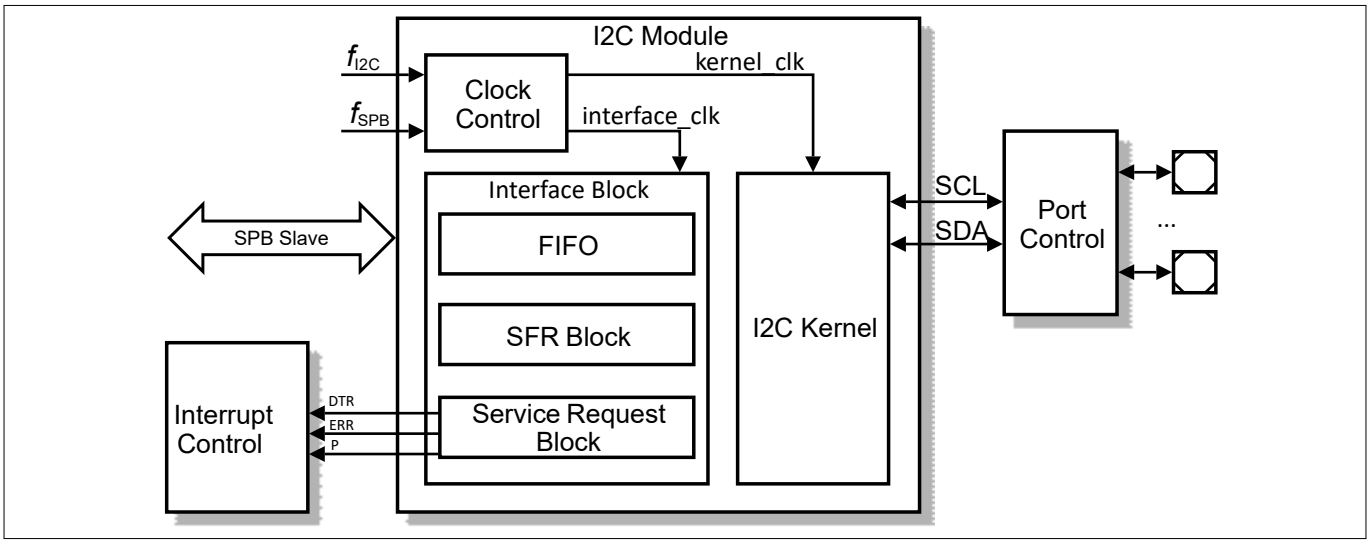


Figure 707 I2C block diagram

The I2C protocol was developed to provide a simple and efficient data transfer between multiple devices over a short distance. It uses a bidirectional serial bus with two wires.

The device can work as master or as slave. The master initiates the transfer, generates the clock pulses, and terminates the transfer; it addresses a slave through a 7-bit or 10-bit address. Data can flow in either direction. In many applications there is only one master, typically a single-chip microcontroller, which communicates with several slaves, for example general purpose peripherals or application specific circuits. However multi-master systems with arbitration and collision detection are also possible.

Data on the I2C-bus can be transferred at rates of up to 100 kbit/s in standard mode, up to 400 kbit/s in fast mode, and up to 3.4 Mbit/s in high-speed mode. A slow slave may stretch the clock period. The number of devices connected to the same I2C-bus is limited by a maximum bus capacitance of 400 pF.

The module relieves the CPU from many time-critical tasks.

The clock for the kernel of the I2C module is derived from the system clock through a pre-scaler. An additional fractional divider generates the desired bit rate. The exact timing of the I2C-bus signals can be adjusted.

A FIFO is used for data transfer between the CPU and the I2C module during transmission and reception. This allows writing and reading of multiple bytes. Data alignment and data sizes can be configured.

Six separate interrupt requests are available: for filling or emptying the FIFO, for reacting on certain protocol events, and for handling errors.

Related information

TC4Dx SMU alarm mapping tables on page 7233

31.3 Functional description

This section gives a description of the kernel, module implementation, and module integration.

31.3.1 I2C kernel description

Functional description of the I2C kernel.

31.3.1.1 I2C protocol

Data is transmitted bit-by-bit on line SDA in conjunction with the clock on line SCL. To start communication, a master device generates a so called start condition. Subsequently data transfer starts. Therefore the data on the SDA line must be stable during the high period of the clock and may only change during SCL low phase. After all data have been transferred, the master closes transmission with a stop condition (see the figure below).

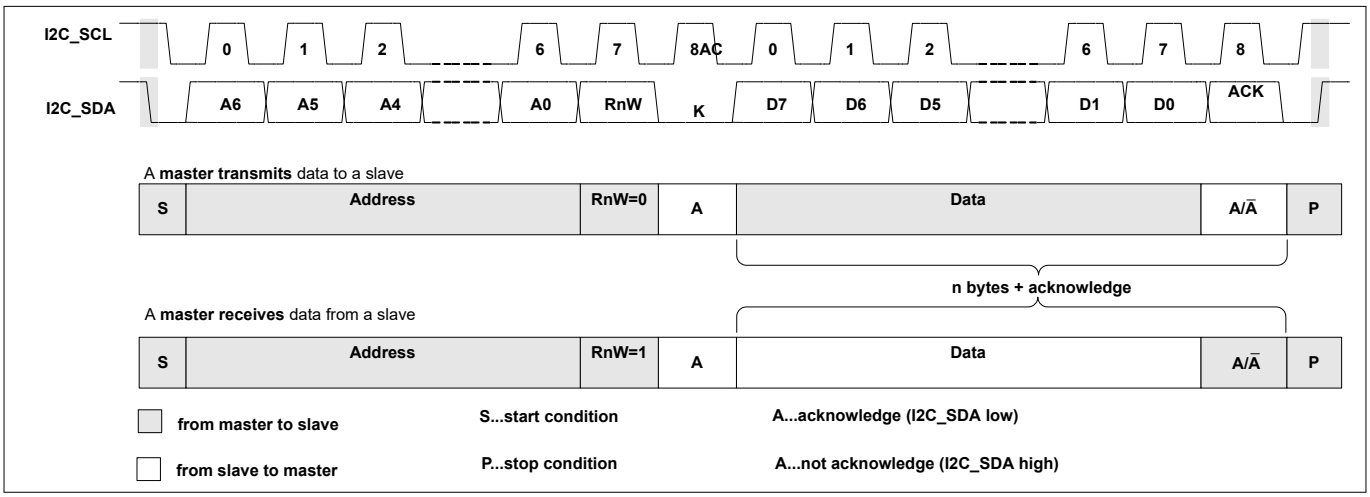


Figure 708 A complete I2C data transfer

Start, restart and stop conditions

A start condition is indicated by a high to low transition on line SDA while SCL remains high, a stop condition is defined by low to high transition under the same condition of high level on line SCL. The bus is considered to be busy after a start condition and to be free again after a stop condition. To allow continuous data transmission or reception without first generating a stop condition, a third condition has been introduced - the restart condition. The bus stays busy if this condition is generated and the same or another slave can be addressed. Start and restart condition are functionally identical.

Address transmission

After the start condition a slave address is sent. Normally, the address is 7 bits long followed by an 8th bit that is a data direction bit (Read/not Write see the figure below). A zero indicates a transmission (write operation); a one indicates a request to data (read operation). Following the address transmission the addressed device responds with an acknowledge (low on line SDA). If no acknowledge is returned, the master can then generate either a stop condition to abort the transfer or a repeated start condition to start another new transfer.

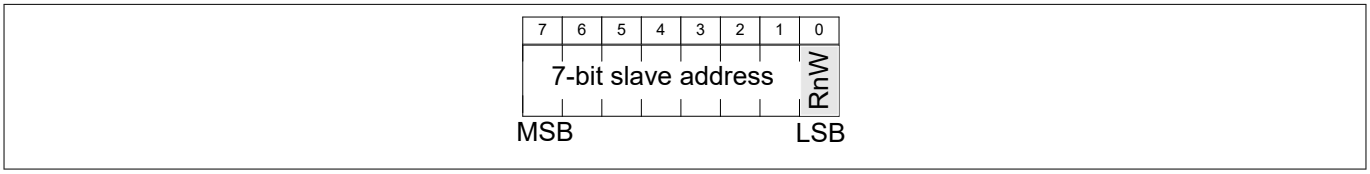


Figure 709 Address byte composition

Two groups of eight addresses (0000 XXX and 1111 XXX) are reserved for purposes, as shown in the table below . For example the bit combination “11110XX” is reserved for 10-bit addressing mode.

Table 1256 Reserved I2C-bus addresses

Address	RnW	Description
0000000	0	General call address
0000000	1	Start byte - no device is allowed to acknowledge
0000001	x	CBUS address - I2C-bus devices may not respond
0000010	x	Reserved for different bus format
0000011	x	Reserved for future use
00001XX	x	High-speed mode master code
11111XX	x	Reserved for future use
11110XX	x	10-bit slave addressing mode

General call

A general call is characterized by transmission of address “0000000” and RnW bit = 0. It is used for addressing every device connected to the I2C-bus. However, if a device does not need any of the data supplied within the general call structure, it can ignore this address by not issuing an acknowledgment. Otherwise it acknowledges this general call and the following data bytes (if required) and behaves like a slave-receiver. The received data (for example programmable part of slave address, address of I2C-bus master) has to be handled by software.

Data transmission

Each byte transferred over the I2C-bus has to be 8 bits long whereby the number of bytes transmitted per transfer is unrestricted. Bytes are transferred MSB (Most Significant Bit) first. If the slave cannot receive or transmit a data byte until it has performed another function, it can hold the clock line SCL low (wired AND connection) to force the master into a wait state. Data transfer continues when the slave releases the clock line.

Acknowledge

Each transferred respectively received byte has to be followed by an acknowledge bit that is set by the recipient (master or slave depending on the transmission direction) onto the data line SDA. The acknowledge-related clock pulse is generated by the master. The device that has to set the acknowledge bit must pull down the SDA line during the acknowledge clock pulse so that it remains stable low during the high period of this clock pulse. Therefore in case of a slave the clock line may be held low until an acknowledge is released.

When the recipient of data does not acknowledge the data, for example because it was unable to receive the data or to transmit any data, the data line must be left high. A master-receiver that does not acknowledge the data from a transmitting slave device thereby tells the slave to stop transmission. The master then generates a stop condition to abort transfer or a repeated start condition to continue.

Clock synchronization

To transfer messages a master generates its own clock on the SCL line. Data is only valid during the high period of the clock. Therefore data on line SDA has to be stable during this period and may only be changed during the low period of clock.

Two special cases must be considered. First, slave devices (or master devices operating as slaves) never generate clocks but are permitted to delay them. According to the following figure, if device 1 as master device for example sets line SCL to low, device 2 as slave device may extend the low phase by setting its SCL output to low. A device such as a microcontroller with limited hardware for the I2C-bus can thereby slow down the bus clock and adapt the master to the internal operating rate of this device.

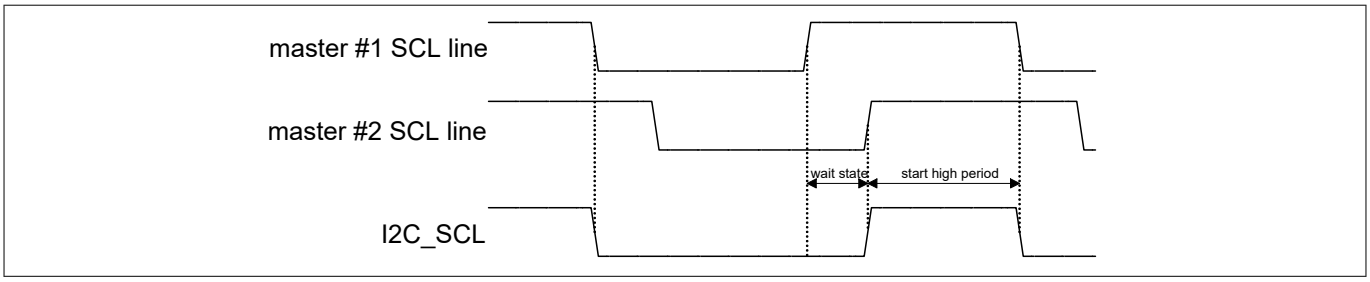


Figure 710 Clock synchronization

Second, if several master devices access the I2C-bus at the same time, the SCL line is influenced by more than one device. As shown in the previous example, the SCL line will be held low by the device with the longest SCL low period. Devices with shorter periods enter a wait state during this time. In this way a synchronized SCL clock is generated.

Arbitration

To allow multi-master mode operation, bus arbitration is required. A master may start a transfer only if the bus is free. Two or more masters may generate a defined start condition simultaneously within a minimum hold time.

Arbitration takes place on the SDA line while SCL line is high, when one master transmits a high level while another master transmits a low level. Because of the wired AND functionality, the low level dominates the high level. As result the device that was trying to transmit a high level recognizes a different level on the SDA line and switches off its SDA and SCL output (see the figure below).

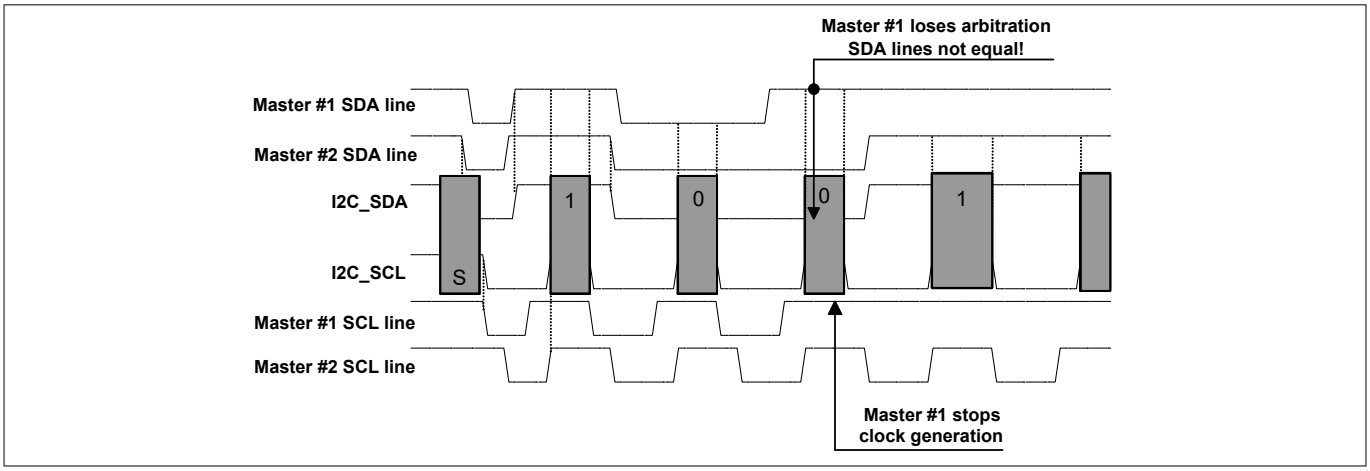


Figure 711 Example of an arbitration procedure

Arbitration may occur in different stages of transmission. A different level may potentially first appear during comparison of the address bits. If each of the masters are trying to address the same device, arbitration continues with comparison of the RnW bit and subsequent data bits (if the masters are transmitting) or acknowledge bits (if the masters are receiving). Because information on the I2C-bus is determined by the winning master, no information is lost during the arbitration process. Since control of the I2C-bus is decided only by comparison of the data transmission stream over line SDA, there is no central master or any order of priority on the bus.

I2C-bus formats

With the described rule types the formats shown in the following figure are possible.

- Master-transmitter transmits to slave-receiver and stops after not-acknowledge

- Master reads slave immediately after transmission of the RnW bit (high level) and the acknowledge from the slave, whereas the master becomes master-receiver and the slave becomes slave-transmitter
- In combined format the regular sequences described above are handled, except for the repeated start condition instead of the stop condition between two sequences

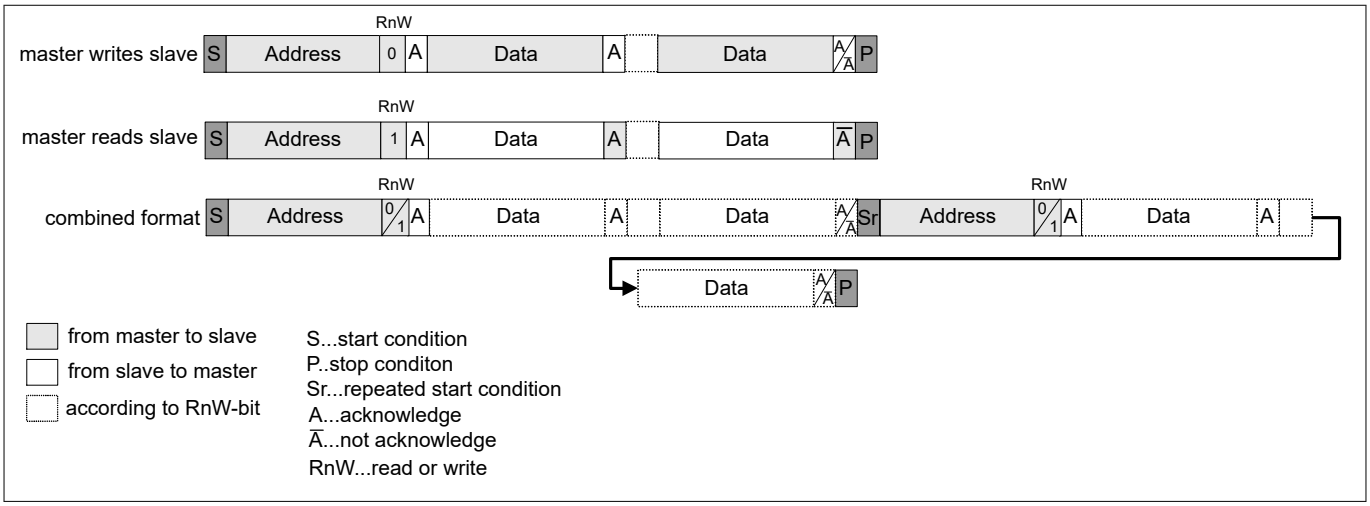


Figure 712 Types of I2C-bus formats

10-bit addressing mode

The I2C address area is to 112 applicable addresses with the 7-bit address mechanism describedabove. It tu rned out, that more combinations were required to prevent problems with the allocation of slaveaddresses for new devices. This problem was resolved with the 10-bit addressing scheme that is compatiblewith 7-bit addressing. Both modes may be used simultaneously.

To use 10 bits for addressing, the preamble address “11110XX” with RnW bit = 0 has to be used, including the two most significant bits “XX” of the 10-bit address, followed by an acknowledge from at least one 10-bit address matching slave and the second address byte containing the eight least significant bits. Only the (combined) 10-bit address matching slave now acknowledges. Data packages may now be transmitted to the addressed slave until stop condition or restart condition. If the master needs to read the slave, a combined format with a restart condition is necessary to

- First transmit the 10-bit address (first data package after preamble address contains second address byte, RnW bit is 0) after the restart condition
- Receive the requested data packages. Note that after the restart condition the preamble address “11110XX” has to be sent one more time, now with RnW bit = 1; the matching slave remembers that it was addressed before (see the following figure). In case of a non matching preamble address, a not-acknowledge is returned

The slave remains active as slave-transceiver until a stop condition or a restart condition occurs. According to the value of the following RnW bit a new slave may be addressed or the same slave is requested to transmit more data. Additionally the following combined formats (and mixtures of these formats) are allowed:

- A master transmits data to a slave and then reads back data from the same slave. The transfer direction is changed after the restart condition
- A master transmits data to one slave and then transmits data to another slave
- 10-bit and 7-bit addressing combined in one serial transfer. After restart condition the new addressing mode is selected by the address itself

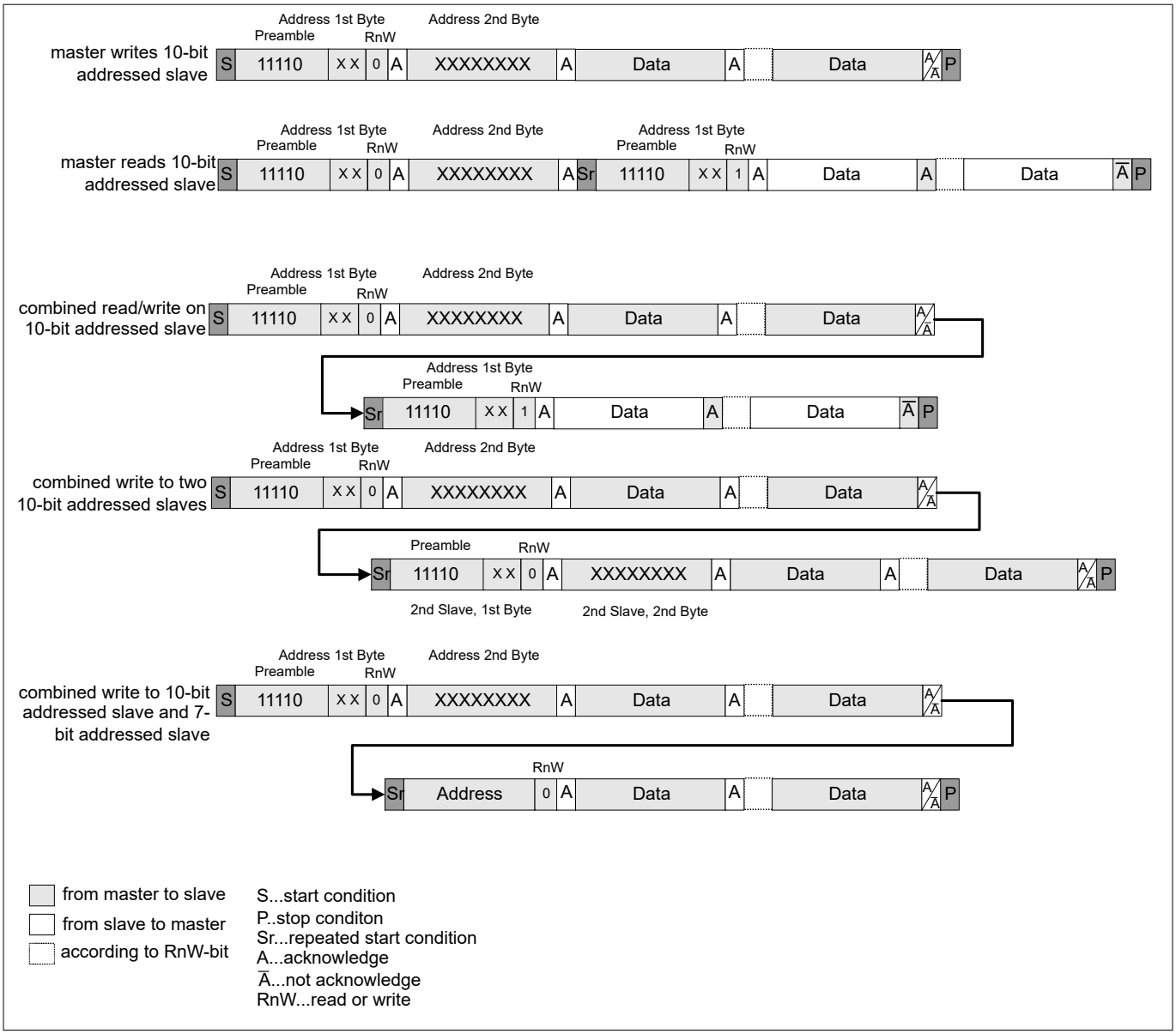


Figure 713 Types of I2C-bus 10-bit address formats

High-speed mode

To support higher baud rates on the I2C-bus (up to 3.4 MBaud) the specification introduces the high-speed mode. To communicate at this higher baud rate the protocol uses a so called master code in place of the address byte to indicate that a master wants to change to the high-speed mode.

Special attention must be paid to meet the requirements of rise and fall times of the signal edges and other timing characteristics when working in high-speed mode. Also it is necessary to disconnect I2C-bus devices which are not able to follow the fast communication.

After the master sends this master code, no device is allowed to acknowledge it. The involved devices change to high-speed mode and the master starts the communication. The high-speed mode transmission behavior is the same as at lower baud rates with the exception that only restart conditions and stop conditions appear (no start condition). Since the master code determines a unique master device to control the bus, the arbitration process is finished after the master code sequence is put on the bus and must not be continued when working at the new communication speed.

Improvements to the regular I2C-bus specification for high-speed mode are given in section 13.1 of the I2C-bus specification version 2.1.

For high-speed mode there exists a separate configuration register FDIVHIGHCFG to program the baud rate. .

Related information

References on page 5512

Baud rate generation for master mode on page 5439

31.3.1.2 Functional restrictions

There are some functional restrictions in this I2C module.

Multi-master mode - master collision

The Multi-master mode is supported with a limitation or restriction in the following situation:

- The I2C module operates in master mode
- Another master wants to access or address the I2C module as slave

The resulting "master collision (problem)" cannot be handled correctly because the incoming address sent by the other master cannot be evaluated. So, if collision occurs second master has to restart access and retransmit the data.

Multi-master mode - hold time for the (repeated) start condition

In the above special situation, this design step of the I2C module violates the timing specification by ~30% for Standard mode and by ~18.6% for Fast mode (minimum 2.8 µs instead 4.0 µs for Standard mode and minimum 0.488 µs instead 0.6 µs for Fast mode). This occurs when data is written into TX FIFO, the I2C module is ready to start transmission and another master starts driving its startbit shortly before the I2C module starts driving. In this case the I2C Finite State Machine tries to win arbitration by reloading approximately half period in baud rate generator, so next SCL clock edge of the I2C module comes earlier than defined by t_7 (refer to datasheet for definition of t_7).

High-speed mode

The standard for the clock duty cycle ratio of the I2C high speed mode is 1:2. See Clocking and timing control in order to achieve the best duty cycle ratio for high speed mode. Note that not for every f_{I2C} frequencies 0% duty cycle ratio deviation can be achieved.

Related information

Clock and timing control on page 5439

31.3.1.3 Clock and timing control

The I2C module offers the following features to control clock and timing:

- Module clock control for integration logic clock and I2C kernel clock
- Baud rate generation
- I2C signal timing adjustment

31.3.1.3.1 Baud rate generation for master mode

A baud rate generation unit in the I2C kernel generates the SCL signal from the kernel_clk, controlled by the settings of parameters FDIVCFG.INC , FDIVCFG.DEC , and TIMCFG.FS_SCL_LOW in registers FDIVCFG for normal and fast mode, FDIVHIGHCFG in high-speed mode, and TIMCFG.

The I2C standard has some special requirements for the clock, such as:

- Asymmetric duty cycle
- Slave or other masters are allowed to delay the clock by extending the low period. For this reason, a master must check if any slave holds the clock line low after the master set the clock line to high. This implies that the state machine of a master needs at least 1 internal clock cycle delay between setting the clock

to high and reading it back to check if it reached high level already. In addition, more delay cycles can be introduced by the time constant of the pull up resistor and the line capacity. In any case, this protocol has the consequence that the actual baud rate will always be lower than the nominal baud rate

Table 1257 I2C baud rate generation configuration for master mode

Mode	EN_SCL_LOW_LEN	FS_SCL_LOW	f_{SCL}	Low time correction	Duty cycle
Standard	0	0	$fSCL = \frac{INC}{2DEC + 3INC} \times fI2C$	-	-
Fast	0	1		-	-
	1	X		1) $SCL_LOW_LEN = \frac{3INC}{2}$	-
High speed	X	X	$fSCL = \frac{INC}{5DEC + 2INC} \times fI2C$	-	1:2

1) This formula can be use to achieve a 50% duty cycle.TIMCFG.SCL_LOW_LEN can be program with other values, but not greater than FDIVCFG.DEC .

TIMCFG.EN_SCL_LOW_LEN, TIMCFG.FS_SCL_LOW and TIMCFG.SCL_LOW_LEN used in the above table are bit-fields in register TIMCFG .

In Table 1258 some example settings to program FDIVCFG.INC , FDIVCFG.DEC and LTC (bit-field TIMCFG.SCL_LOW_LEN) are given and information about the duty cycle is shown.

Table 1258 I2C INC and DEC settings for master mode

I2C mode	f_{I2C} [MHz]	INC	DEC	SCL_LOW_LEN	Duty cycle [%]	f_{SCL} [MHz]
Standard	66.6	1	332	-	56.3	0.099
Fast	66.6	2	170	3	50	0.384
High speed	66.6	46	162	-	30.6	3.399

If a slave device is sensitive for violation of maximum frequency according I2C standard (for example 3.4 MHz in high-speed mode) it is recommended to select a target SCL frequency with some margin to this limit.

31.3.1.3.2 Baud rate generation for slave mode

Baud rate generation is mainly used for master mode, but there is one corner case where a support of the baud rate generation is required in slave mode: when I2C is in slave mode with a pending transmit data request and FIFO is currently empty then SCL is kept low until FIFO is filled after some time. Then transmit data is sent out immediately. SCL is kept low for a minimum time of t_4 as shown in the table below .

In order to keep the set-up time, t_4 , according to the I2C standard, the baud rate generation is used to guarantee the delay on SCL. The formulas in the 3rd and the 4th column of the following table shows how to configure FDIVCFG.DEC and FDIVCFG.INC for a given f_{I2C} and the intended set-up time. The formulas in column four define the real value of t_4 with the selected FDIVCFG.DEC and FDIVCFG.INC values.

It is not recommended to use the same value for FDIVCFG.DEC and FDIVCFG.INC as in master mode, and would lead to additional delay, as defined by the standard, in most cases.

Table 1259 I2C baud rate generation configuration for slave mode

MODE	min t_4[µs]	DEC, INC	Resulting t_4 [µs][1) 2)]
Standard	0.25	$\mathrm{DEC} = \frac{8}{9}((\mathrm{fI2C}) \times (t_4) \times \mathrm{INC} + \mathrm{INC})$	$C = \mathrm{INT}\left(\frac{\mathrm{DEC} + \mathrm{DEC(div)8}}{\mathrm{INC}} - 1\right)$ $\frac{C}{\mathrm{fI2C}} = t_4$
Fast	0.1	3) $\mathrm{DEC} = \frac{4}{5}((\mathrm{fI2C}) \times (t_4) \times \mathrm{INC} + \mathrm{INC})$	$C = \mathrm{INT}\left(\frac{\mathrm{DEC} + \mathrm{DEC(div)4}}{\mathrm{INC}} - 1\right)$ $\frac{C}{\mathrm{fI2C}} = t_4$
High speed	0.01	$\mathrm{DEC} = \mathrm{INC} + 1$	$C = \mathrm{INT}\left(\frac{\mathrm{DEC}}{\mathrm{INC}} + 1\right)$ $\frac{C}{\mathrm{fI2C}} = t_4$

1) The used abbreviation INT is the integer function.
2) The operator div denotes the integer division: a div b = greatest integer not greater than a divided by b.
3) SCL_LOW_LEN can not be greater than DEC.

In the following table some example settings are given to program FDIVCFG.INC and FDIVCFG.DEC. The used abbreviation INT is the integer function.

Table 1260 I2C INC and DEC settings for slave mode

I2C MODE	f_{I2C} [MHz]	INC	DEC	t_4 [ns]
Standard	66.6	20	314	250
Fast	66.6	25	160	105.1
High speed	66.6	250	251	30

The minimum kernel frequency is 8 MHz for standard and fast mode and 55 MHz for high speed mode.

31.3.1.3.3 I2C timing adjustment

The I2C standard includes a number of requirements for the SCL and the SDA timing. These cannot always be fulfilled by the timing of the pads. Therefore, additional timing adjustment options are provided in the controller logic.

Various timings of signals can be shifted by multiples of kernel_clk cycles, as defined in registerTIMCFG .

These settings offer a wide range of timing compensation. Not all of the values can be used and will result in a valid I2C timing, even clock or data hang-up is possible. Detailed analyses should be done - for example during chip evaluation - taking actual kernel_clk and external timing into account.

SDA_DEL_HD_DAT

- Delays SDA output data starting from SCL falling edge for a defined number of kernel_clks
- Must not exceed SCL low period to keep setup time for collision detection at next SCL rising edge

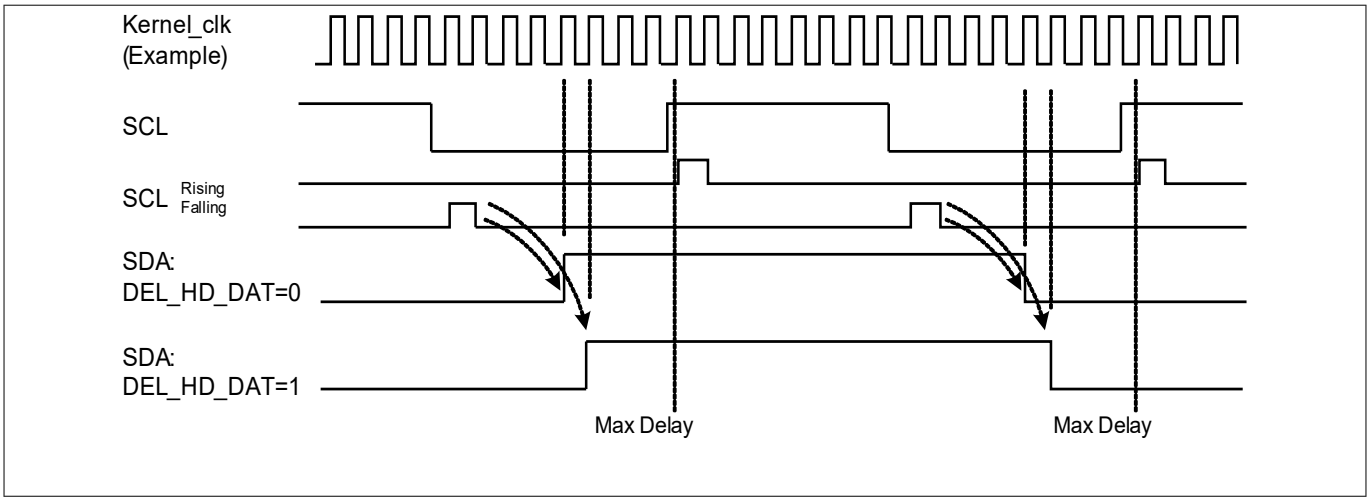


Figure 714 SDA_DEL_HD_DAT

High speed mode corrections

In the following figure, the base period and delay corrections are shown for the entering and stop phases of HS mode.

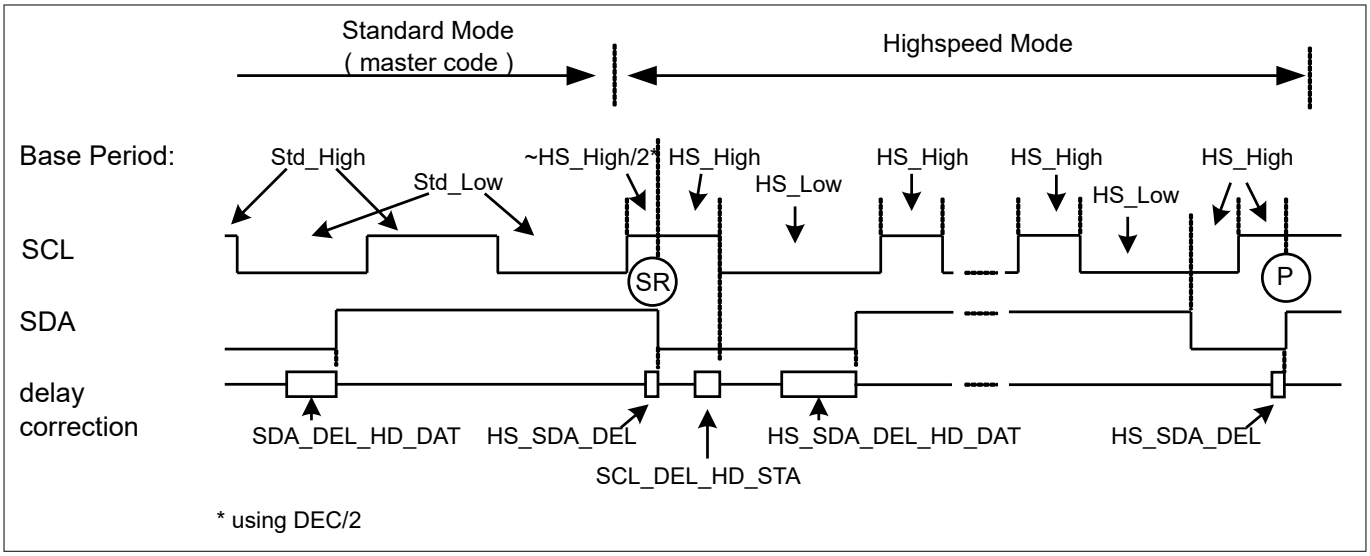


Figure 715 High speed mode corrections

31.3.1.4 I2C kernel control logic

The I2C kernel can work in four different modes:

- Master-transmitter
- Master-receiver
- Slave-transmitter
- Slave-receiver

A state machine description can give a good and fast overview of the functionality. Also nearly all possible conditions can be rebuilt by stepping through the state machine and most of the error handling is covered by it. So this method is used to describe the I2C kernel.

Beginning with the top-level state machine, the starting up procedure is covered and together with the slave process and the master process sub-state machines, the active operations are understood. These three state machines and their state transitions are described in the following sections.

As the I2C-bus working principle is packet orientated, the FIFO is typically configured as flow controller and so it issues the burst- and single-requests. This configuration is established by programming the bits FIFOCFG.RXFC and FIFOCFG.TXFC of the FIFOCFG register. Writing to the TXD register after writing to the TPSCTRL.TPS bit-field of register TPSCTRL starts the transfer.

***Notes**:*

1. *Peripheral is flow controller. The principle is as described before*

2. *Peripheral is not flow controller. In principle it is possible to use the FIFO in non flow controller mode. When the FIFO is not configured to operate as flow controller, the behavior of the FIFO operation has to be considered. The update of the FIFO fill level is only done when a stage is filled completely. In case of a (not protocol compliant) stopped transfer of the transmitter, the device is not able to evaluate the valid bytes since the FIFO issues interrupts and updates fill level only when a stage is filled completely. To prevent this behavior, the FIFO can be, for example, programmed to word alignment, so every received byte issues an interrupt and update. In any case the software has to take care of such upcoming not protocol compliant data transmission stops, and has to handle this for example by programming an appropriate time-out*

The top-level state machine

The following figure shows the top-level state machine. There are four states in the top-level state machine:

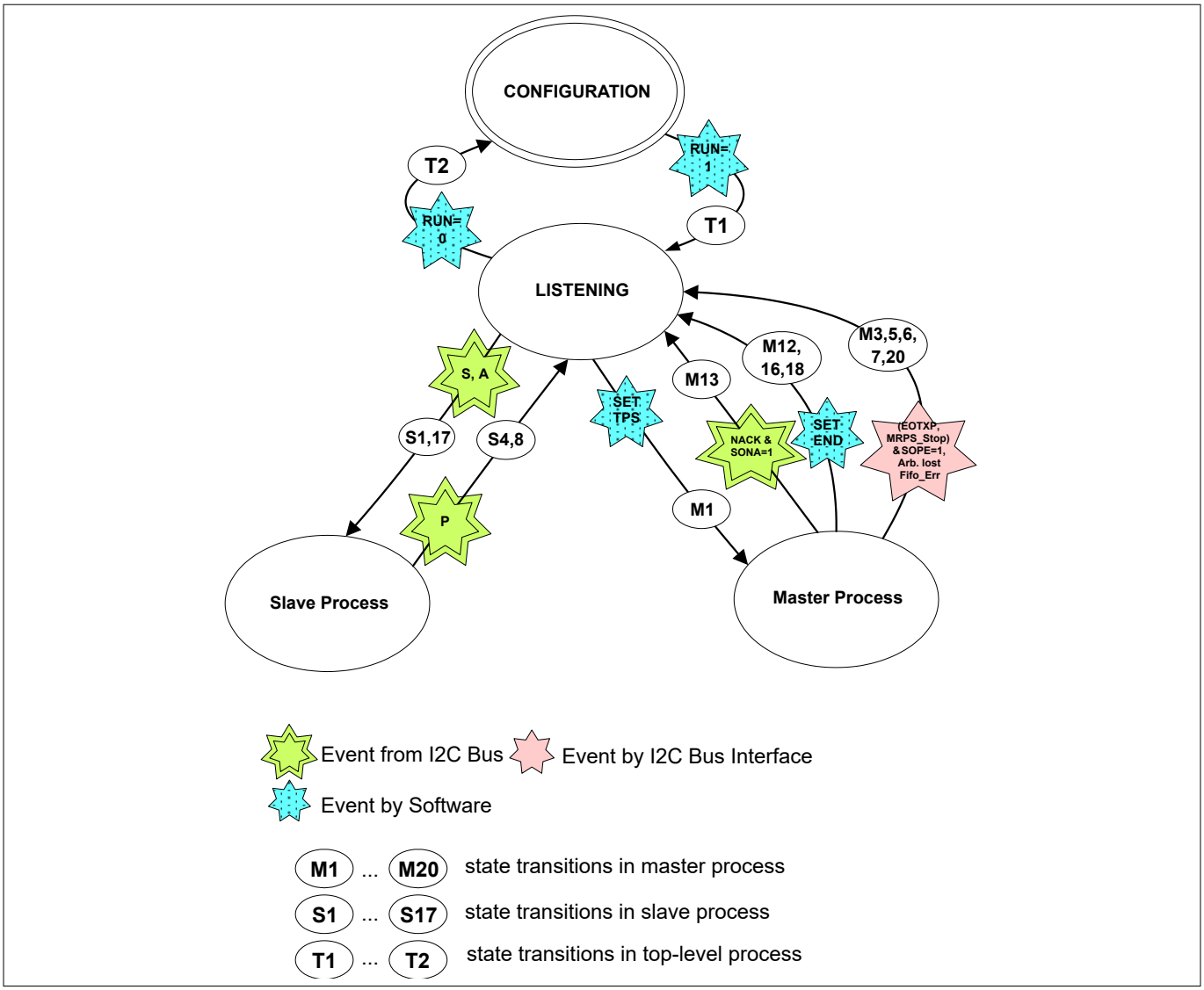


Figure 716 Top-level state machine

CONFIGURATION

The peripheral is inactive. This is indicated by the RUNCTRL.RUN bit in the RUNCTRL register, which is set to 0. In this state the peripheral should be configured by software by writing appropriate data words to the ADDRCFG, FDIVCFG, and FIFOCFG registers. This determines the following features:

- Peripheral is master or slave
- Address when accessed by another device
- Answering behavior to master calls and general calls
- Operating speed on the I2C-bus
- FIFO behavior

LISTENING

The peripheral has been activated by software. This state is entered in master and slave mode. The peripheral has to observe the bus for a certain time (one period of 1/100 kHz should be sufficient), which has to be programmed in software, to ensure that there is no running transmission on the bus after this state has been entered. If no transitions on the I2C-bus are detected during this time period, the bus is supposed to be free. Otherwise the bit-field BUSSTAT.BS (bus status) in register BUSSTAT is set to 01_B (bus busy). Further functionality depends on the configuration: If the peripheral is configured as slave, only the slave process can be entered. If the device works as master, slave and master processes can be entered.

SLAVE PROCESS

A Start condition has been detected. The device is waiting for commands from the I2C-bus. This state represents the operation when acting as slave-transmitter or slave-receiver and is itemized in an own sub state machine.

MASTER PROCESS

The device is able to control the bus and work either as master-receiver or master-transmitter. Before the master can control the bus it has to set its clock frequency by programming its fractional divider with the appropriate value by writing it to the FDIVCFG register (when the I2C-bus is also operating in high-speed mode the appropriate values for INC and DEC have to be written to FDIVHIGHCFG). The master process state is itemized in an own sub state machine.

The transition between the three states on the top-level are:

T1

The RUNCTRL.RUN bit has been set to 1 by software. The peripheral is activated and ready to observe the I2C-bus or react on transmission start commands.

T2

The RUNCTRL.RUN bit has been set to 0 again. This means that because of new configuration effort, the peripheral is switched back to CONFIGURATION mode.

Attention**:* ***When turning off the peripheral it must be assured that no I2C-bus control responsibility is owned by the interface. Therefore the shut off process has to change the peripheral into a "secure" state and free the I2C-bus if necessary (generate a stop condition if in master mode). There are signals provided by the interface block which can be used to guarantee a secure shut off when turning off the I2C kernel.

Slave process sub-state machine

The following figure shows the slave process sub-state machine. There are 5 plus 1 (the listening state belongs to master and slave) different states in this sub-state machine:

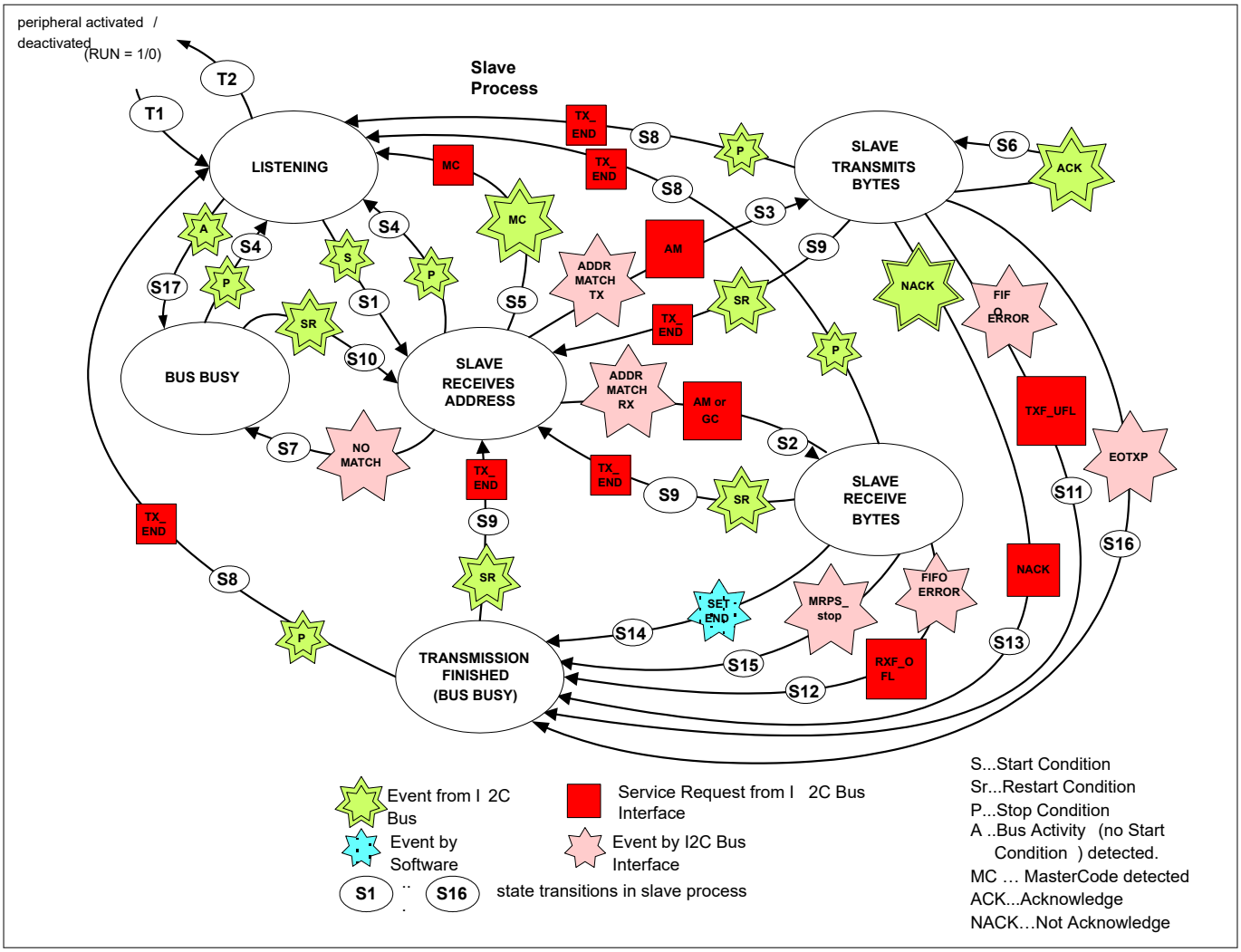


Figure 717 Slave process sub-state machine

LISTENING

This state has been previously described (in figure Top-level state machine). If the device is configured as slave, the module is listening only to the bus activity.

SLAVE RECEIVES ADDRESS

The module is shifting in the serial bits from the I2C-bus and acknowledges if the received byte(s) (first one in 7-bit address mode, first and second one in 10-bit address mode) match the adjusted address. During address matching the slave-receiver has the possibility to slow down the master clock by holding the SCL line low after each bit, since this can be used as a bit-by-bit handshake procedure.

***Note**: The address byte(s) is (are) not shifted into the FIFO.*

SLAVE RECEIVES BYTES

The module is shifting in the serial bits from the I2C-bus and acknowledges when the whole data byte has been transported to the FIFO. The slave-receiver has the possibility to slow down the master clock by holding the SCL line low after each bit since this can be used as a bit-by-bit handshake procedure.

SLAVE TRANSMITS BYTES

The slave is allowed to hold down the SCL line when preparing the data for transmission. When valid data is available the slave releases the SCL line and waits for the clock cycles from the master. The interrupt routine which proceeds when entering this state may write to the TPSCTRL register to initiate a transmission. This is only allowed if the bit-field BUSSTAT.BS (bus status) in register BUSSTAT

in combination with the BUSSTAT.RnW clearly identifies the slave-transmitter mode of the peripheral. Otherwise writing to this register has no effect. First the I2C kernel gets a byte from the FIFO and puts it into a shift register. The module is then putting single data bits onto the I2C-bus and releases the SDA line after the 8th bit is sent.

***Note**: Since the data flow control responsibility is task of the master device the slave should not stop transmitting bytes until the master tells it to do so. Therefore the TPSCTRL.TPS value in register TPSCTRL should be large enough to not run out of data bytes. In case of running out of data, the TRANSMISSION FINISHED state is reentered. The I2C data line is released to high level. The master continues receiving wrong values (0xFF).*

BUS BUSY

The bus is busy and the module is not involved in any data transfer. This is also the entry state when the bus is not free when starting up procedure is taking place (see state S17).

TRANSMISSION FINISHED (BUS BUSY)

The module was involved in any data transfer. In distinction to the BUS BUSY state, the slave is still addressed by a master. Every time this state is left, caused due to a stop/restart condition, a TX_END request is generated as indication that the previous transmission has been closed.

The transitions which are connecting the several states are in detail:

S1

The module is detecting a start condition on the bus. It has to set the bit-field BUSSTAT.BS (bus status)to 01_B. When this situation is occurring the I2C kernel is initialized since this indicates a new data transmission.

S2

The address which has been received matches the one stored in the address field of register ADDRCFG. The BUSSTAT.RnW bit (8th bit in the first received byte) was set to 0. This means that the master wants to write to the slave. The corresponding bit BUSSTAT.RnW is set to 1, that is the device continues working as slave-receiver. Also an acknowledge has to be set on the I2C-bus in the appropriate place (9th clock cycle). The I2C kernel sets the bit-field BUSSTAT.BS (bus status) to 11_B and generates an AM (address match) request.

If the general call (GC) address matching is enabled by setting bit ADDRCFG.GCE , the device has to react also when it receives a general call (0x00). The GC request is issued and also an acknowledge has to be set on the I2C-bus when the GC is detected.

S3

The address which has been received matches the one stored in the address field in the ADDRCFG register and the SDA line was at a high level when the BUSSTAT.RnW bit was sent. This means that the remote master wants to read from the slave. The corresponding bit BUSSTAT.RnW has to be set to 0 (write). Also an acknowledge has to be set on the I2C-bus in the appropriate place (9th clock cycle). The I2C kernel sets the bit-field BUSSTAT.BS (bus status) to 11_B and generates an AM (address match) request. The device is now working as a slave-transmitter.

S4

The device detects a stop condition on the I2C-bus and switches back to SLAVE LISTENING state. No interrupt is generated after the device was not addressed. The bit-field BUSSTAT.BS (bus status) is set to 00_B (bus free).

S5

If the device is configured through bit MCE (master code enable) in the ADDRCFG register to react on a master code, the device can change baud rate to high-speed (Hs) mode when a remote master sends a master code. No device is allowed to acknowledge this master call. The interrupt routine which is called (MC request issued), has to handle the clearing of the interrupt bit in register PIRQSC. The LISTENING state is reentered. From now on the device works as slave in high-speed mode until a stop condition occurs.

S6

The slave-transmitter has received an acknowledge from the master-receiver and continues transmission.

S7

After the reception of the address bytes has been finished and no address match has taken place, the device changes to BUS BUSY state.

S8

The device has detected a stop condition on the I2C-bus after/while it was addressed. A TX_END (transmission end) request is generated, the bus is free again respectively the bit-field BS (bus status) in register BUSSTAT has to be set to 00_B again. The slave is now in LISTENING state.

S9

The slave has received a restart condition (repeated start condition) after/while it was addressed. A TX_END (transmission end) request is generated, the state switches back to SLAVE RECEIVES ADDRESS. The slave is reset again to a new transmission pending state (bit-field BS in register BUSSTAT = 01_B.). When the TBAM is activated (configuration register) the slave has to remember that it was addressed before because this event may be followed by a read access of the master to this device. So the address procedure after reset can be shortened and the transition S3 occurs one byte earlier (master sends 10-bit address mode byte with RnW bit = 1 (read)).

S10

A running transmission is ended by a remote master and this device wants to continue without losing control of the I2C-bus. The slave was not in transmission and resets to receive address bytes again since the new data transmission is concerning all connected slaves.

S11

The peripheral has problems to read data bytes from the FIFO (underflow/not ready). This is indicated by generating a TXF_UFL (TX FIFO underflow) request. It also changes to TRANSMISSION FINISHED state.

***Note**:* *The device is still addressed by a master (bit-field BS in register BUSSTAT = 11_B.) until a stop or restart condition occurs.*

S12

The peripheral has problems to write the received byte into the FIFO (overflow/not ready). This is indicated by generating a RXF_OFL (RX FIFO overflow) request. The slave-transmitter puts a not-acknowledge on the I2C-bus to indicate its situation to the controlling master. This byte is not valid and will be discarded.

***Note**:* *The device is still addressed by a master (bit-field BS in register BUSSTAT = 11_B.) until a stop or restart condition occurs.*

S13

The remote master wishes to close the connection to this slave and sets a not-acknowledge on the I2C-bus. The device generates a NACK (not-acknowledge) request and changes to TRANSMISSION FINISHED state (bit-field BS in register BUSSTAT = 01_B.). The interrupt routine can clear the appropriate bit through the corresponding bit in the PIRQSC register.

S14

The slave wants to stop the transmission and sets the ENDDCTRL.SETEND bit. It puts a not-acknowledge on the I2C-bus after the next received byte. With this the slave gets the possibility to cancel the transmission through software intervention.

S15

The value of received bytes is equal to bit-field MRPS (maximum receive packet size) of the FIFO register MRPSCTRL. The FIFO part of the interface generates the signal MRPS_stop to the I2C kernel, which indicates the end of the received packet capacity. The slave produces a not-acknowledge to the master and changes to TRANSMISSION FINISHED.

S16

The number of transmitted bytes is equal to the TPS bit-field of the FIFO register TPSCTRL. The FIFO controller generates the EOTXP (end of transmit packet) signal to the I2C kernel, which indicates the last byte to transmit. After the last byte the device changes to TRANSMISSION FINISHED.

***Note**: The master is not informed about this situation. Since the data flow control responsibility is task of the master device, it may continue receiving wrong values (0xFF).*

S17

The module detects bus activity; no start condition has been detected before. A transmission may be currently in progress. The device changes to BUS BUSY state. This transition is performed if a start condition has been missed because the device was not activated (CONFIGURATION state).

Master process sub-state machine

The following figure shows the master process sub-state machine. There are four different states in this sub-state machine:

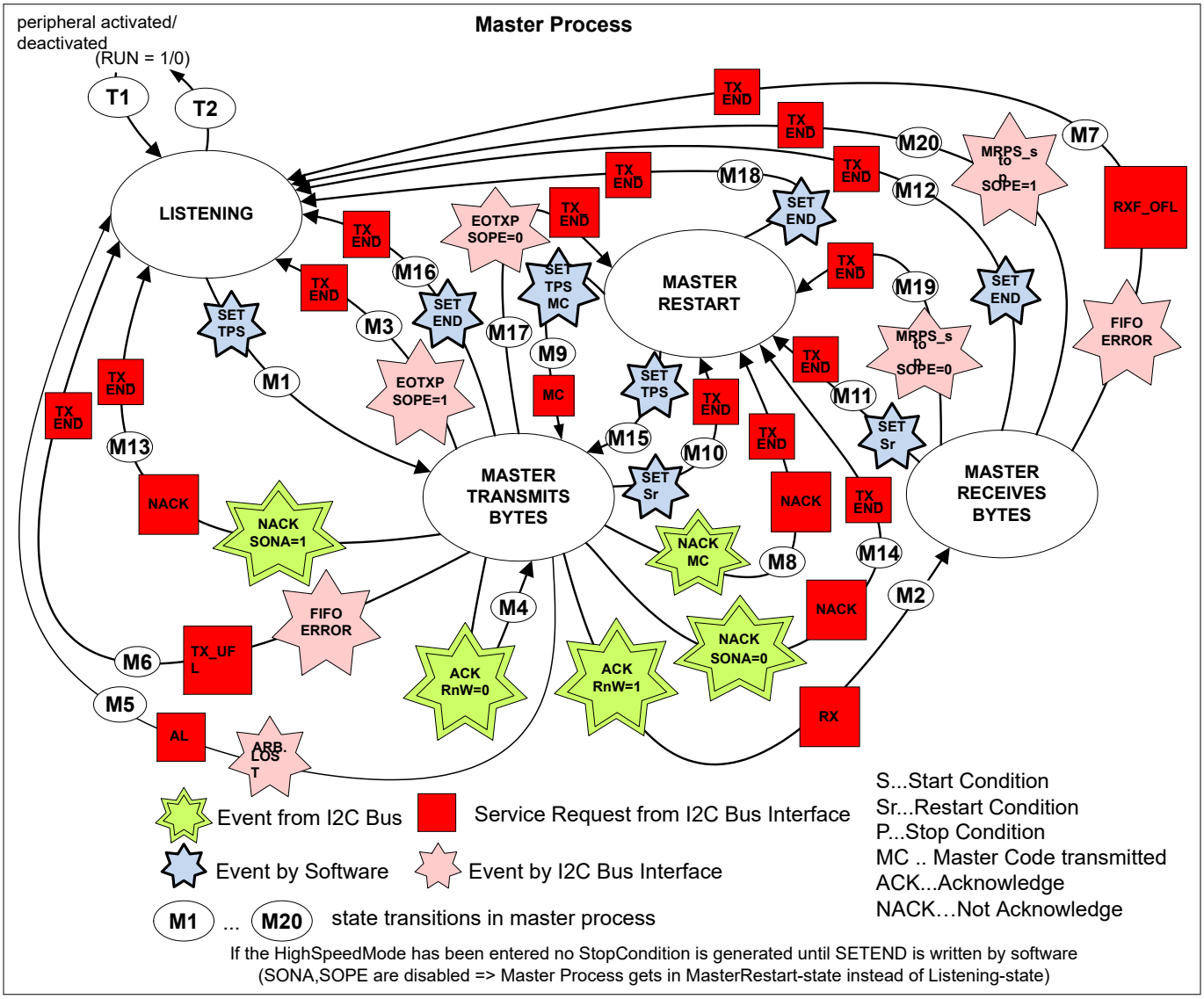


Figure 718 Master process sub-state machine

LISTENING

This state has been previously described (in the figure Top-level state machine). If the device is configured as master, it is listening to events either from software (writing to TPSCTRL register) or from the I2C-bus.

MASTER TRANSMITS BYTES

The device has started a transmission on the I2C-bus by software. The first byte after a start condition contains the address byte including the RnW bit. According to this bit (RnW bit in BUSSTAT register is set by hardware) the direction of the following transfer is determined. Bytes are transmitted on the I2C-bus until the FIFO is empty or a stop is initiated by software.

If bit MCE (master code enable) is set in the ADDRCFG register, the device has to check if the transmitted address byte is a master code.

MASTER RESTART

The device wants to restart a transmission without losing the control over the I2C-bus. The clock SCL stays low as long as no further communication is requested. The only event that should occur in this state is writing to the TPSCTRL register. Nevertheless the device can give up the I2C-bus control by writing to bit SETEND in the ENDDCTRL register. Of course, when this situation occurs, the peripheral produces a stop condition on the I2C-bus.

If bit MCE (master code enable) is set in the ADDRCFG register, the device has to check if the transmitted address byte is a master code.

MASTER RECEIVES BYTES

The master has ordered bytes from the slave and handles the data transfer from the I2C-bus to the FIFO. This means that the bits are shifted into a buffer register in the I2C kernel and afterwards are passed to the FIFO. Every time a byte was transferred to the FIFO successfully the master-receiver produces an acknowledge in the according position.

The state transitions in the master process are:

M1

The software has written the number of bytes which have to be transmitted over the I2C-bus into the control register TPSCTRL. The FIFO generates appropriate data transfer requests which transfers valid data into the FIFO. A start condition is set on the bus and the bit-field BS (bus status) in register BUSSTAT is set to 10_B (busy master). The first (the first and the second) byte is (are) the 7-bit (10-bit) address of the slave which shall be accessed.

If the master wishes to initiate a transition to high-speed mode, the transition is only valid in combination with the MCE (master code enable) bit in the ADDRCFG register. The software has to write the master code into the FIFO, this means also the value of the TPSCTRL is predetermined with 1.

M2

The master receives an acknowledge from slave and the bit RnW in the BUSSTAT register was set to 1 during the addressing phase. A slave has been addressed successfully as transmitter, the state moves to MASTER RECEIVES BYTES. An RX (Receive) request is generated to distinguish between transmit- and receive-FIFO requests in non flow controller mode.

M3

The valid data packet from the FIFO was transferred successfully to the I2C-bus and the FIFO indicates this by the EOTXP (end of transmit packet) signal. If the RnW bit was set to 0 (writing to a slave) and bit SOPE (stop on packet end) in register ADDRCFG was set to 1, the transfer is closed by putting a stop condition on the bus. After this stop condition has been detected the peripheral indicates this successful transfer to the software through the TX_END (transmission end) request. The bus status bit-field is set to 0x0 (bus free).

M4

After the master transmits a byte it releases the SDA line to allow the addressed slave to pull down the SDA line (acknowledge). If this event occurs (and RnW in register BUSSTAT was set to 0), the master can continue to transfer bytes on the I2C-bus.

M5

The SDA line level is not equal to the one that is set by the master. This means that another master also tries to control the I2C-bus. This is indicated by the AL (arbitration lost) request and the device switches to the LISTENING state. The interrupt routine has to clear bit AL in the PIRQSC register. The Arbitration lost indication is not executable when the device works in high-speed mode (no arbitration procedure in high-speed mode since there is only one master remaining). The arbitration process must be operating at

each bit which is sent by the master. Before interrupting the CPU, the bit-field BS (bus status) in register BUSSTAT is set to 01_B.

M6

If the master wants to transmit bytes but the FIFO is empty (underflow), it has to generate a TXF_UFL (TX FIFO underflow) request. The bit-field BS (bus status) in register BUSSTAT is reset to 00_B, and a TX_END (transmission end) request is generated after the master has put a stop condition on the bus.

M7

The I2C kernel wants to transfer bytes to the FIFO but without success. Then it has to produce an RXF_OFL (RX FIFO overflow) request. The current byte is not-acknowledged and a stop condition is put on the bus (bit-field BS in register BUSSTAT is set to 00 and a TX_END (transmission end) request is generated). Requests (SREQ, BREQ, LSREQ or LBREQ) are generated for the outstanding data, the target is to empty the FIFO.

M8

If master code has been transmitted by the device and a NACK (not-acknowledge) has been received, the MASTER RESTART state is entered. The clock SCL stays low as long as no further communication is requested. This is done by writing bit-field TPS in register TPSCTRL (see transition M9).

M9

A new communication in high-speed mode is started by writing bit-field TPS in register TPSCTRL when a master code has been previously transmitted (M8). The switch into high-speed mode (MC request) is done as soon as clock and data are released to high (see the figure below). State MASTER TRANSMITS BYTES is entered. During high-speed mode, all fast and standard mode devices must be disconnected from the bus. This can be done in the MC request.

M10

Bit SETRSC has been set during the transmission of the data (the software wants to change direction/slave and therefore writes to the ENDDCTRL register). The I2C kernel resets and changes to MASTER RESTART state and a TX_END (transmission end) request is generated. Therefore, the master does not lose the control of the I2C-bus after the transmission has ended. The master stops transmission after the current byte was acknowledged or not acknowledged. Valid data in the FIFO are discarded. In this situation the FIFO is flushed to produce an initial situation.

M11

If the master wants to change direction/slave immediately in this state, it has to write to bit SETRSC in the ENDDCTRL register. The master puts a not-acknowledge on the appropriate place during the current transmitted byte. This indicates the slave-transmitter to stop transmission. The master switches to MASTER RESTART state; a TX_END (transmission end) request is generated.

M12

By sending a not-acknowledge on the bus, the master indicates the slave an upcoming stop of the transmission. This is established by writing to bit SETEND in register ENDDCTRl. Stop condition and TX_END (transmission end) request are generated. Requests (SREQ, BREQ, LSREQ or LBREQ) are generated for the outstanding data, the target is to empty the FIFO. The LISTENING state is entered; the bit-field BS (bus status) in register BUSSTAT is set to 00_B.

M13

If the master is receiving a not-acknowledge and bit SONA (stop on not-acknowledge) in the ADDRCFG register is 1, it produces a stop condition, resets the bit-field BS (bus status) in register BUSSTAT and issues a NACK (not-acknowledge) request and a TX_END (transmission end) request. The FIFO is flushed; remaining data in the FIFO stages are discarded.

M14

If bit SONA (stop on not-acknowledge) in the ADDRCFG register is 0, the peripheral wants to remain the controlling master of the I2C-bus after receiving a not-acknowledge. The next state is MASTER RESTART. A NACK (not-acknowledge) request and a TX_END (transmission end) request indicate this event to the software. The FIFO transfers the valid data to the memory.

M15

The master decided to transfer data on the bus again and writes to the TPSCTRL register. The first byte is interpreted as address byte (the 8th bit indicates RnW). The appropriate data transfer requests are activated by the FIFO controller. After the SCL line is high again (slave can stretch the low period) the master is allowed to put a restart condition onto the bus and is again in the MASTER TRANSMITS BYTES state starting to transmit bytes again.

M16

The software has set the SETEND bit in the ENDDCTRL register. The peripheral stops transmission of bytes after the current byte has been written on the bus. The I2C kernel flushes the FIFO since the remaining data in the FIFO is invalid. A TX_END (transmission end) request is generated.

M17

The FIFO indicates the end of the data packet and the peripheral was configured to go into MASTER RESTART state after that event. Since the data transfer has ended as intended a TX_END (transmission end) request is produced.

M18

The master has no data to send anymore but still controls the I2C-bus. Therefore it can use the SETEND bit to give up the I2C-bus control by setting a stop condition. A TX_END (transmission end) request is generated.

M19

A value different to zero is programmed to bit-field MRPS (maximum receive packet size) of the MRPSCTRL register. If the number of received bytes is equal to the MRPS bit-field, the FIFO produces the MRPS_stop signal. When bit SOPE (stop on packet end) in the ADDRCFG register is set to 0, the master generates a not-acknowledge and changes to MASTER RESTART state, so the master does not lose the control over the I2C-bus. A TX_END (transmission end) request is generated.

M20

(See also M19.) The FIFO produces the MRPS_stop signal which indicates that the desired amount of received data bytes is reached. The peripheral generates a not-acknowledge and changes to LISTENING state because bit SOPE (stop on packet end) in the ADDRCFG register is set to 1. Of course, the interface produces a stop condition; so the I2C-bus is free again (change also the bus status bit-field). A TX_END (transmission end) request is also generated. Requests (SREQ, BREQ, LSREQ or LBREQ) are generated for the outstanding data, the target is to empty the FIFO. The LBREQ or the LSREQ indicate the completed reception of data.

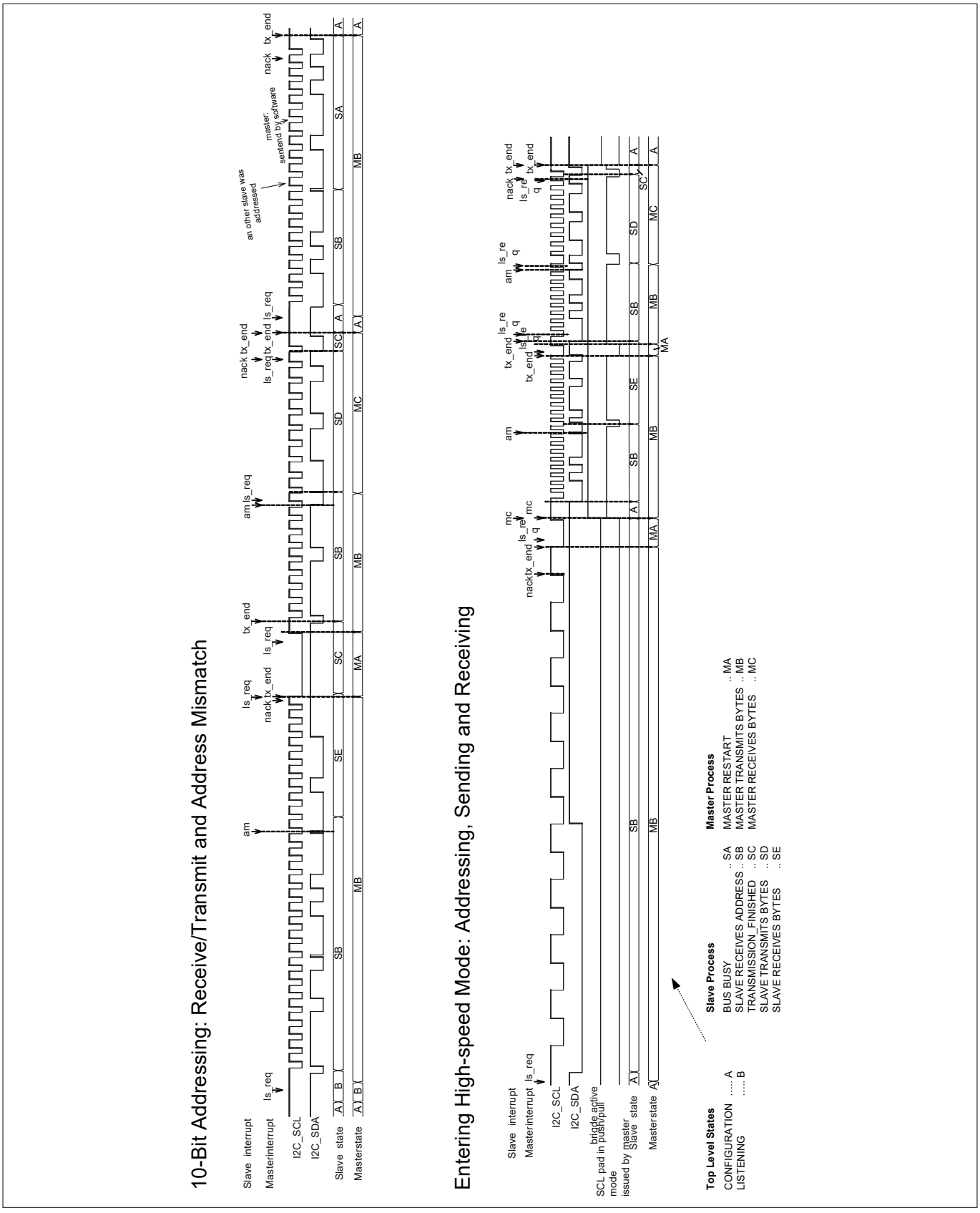


Figure 719 Connection examples including requests in master and slave mode

Related information

Receive request generation on page 5460

Transmit request generation on page 5456

31.3.1.5 FIFO operation

The I2C module uses a FIFO between its kernel and the bus, in order to adapt the character processing speed of the peripheral to the transfer rate of the bus system. The FIFO has a transmission state and a reception state. The following figure shows the bidirectional half duplex FIFO unit with the input and output lines and with the corresponding FIFO registers.

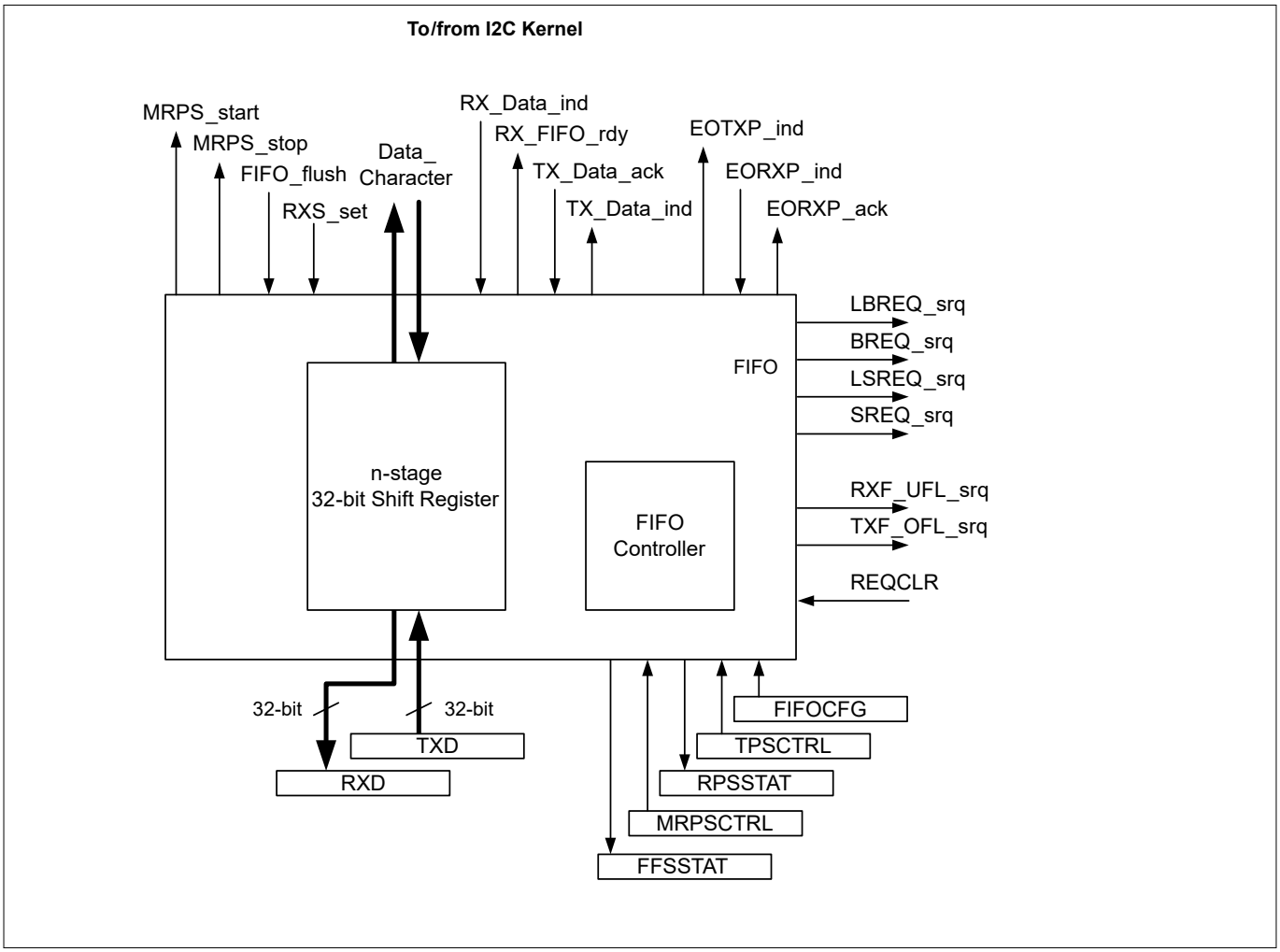


Figure 720 Bidirectional half duplex FIFO

***Notes**:*

1. *When the FIFO is not serviced in time (transmission: filled with transmit data, reception: read the received data), an underflow or overflow event will lead to abortion of the transaction. To avoid this behavior, the software shall be configured to transfer a maximum of 32 bytes per transfer. If more than 32 bytes should be transmitted, the transmissions should be divided in maximum 32 data bytes per transfer*
2. *If bit-field FIFOCFG.CRBC in register FIFOCFG is disabled, a linked list should be used to avoid that the CPU has to get active to clear the request. The DMA channel that serves I2C has to be configured with a linked list that first makes the FIFO-TX/RX transfer and second clears the interrupt by writing 0x0 to register ICR (Interrupt Clear Register)*

31.3.1.5.1 Data transmission

The software can move transmit data to the FIFO by writing to the Transmission Data Register TXD. Moving data to the FIFO can be done according to the data transfer request signals generated by the FIFO controller. But with the register FFSSTAT, which indicates the number of full FIFO stages, the data transfer to the FIFO can be handled completely without data transfer requests. The data alignment within the FIFO is described in Transmit data alignment . The FIFO shifts data to the I2C kernel by means of the handshake lines tx_data_ind and tx_data_ack.

With the signal FIFO_flush the I2C kernel is able to clear the FIFO content. But if a data transfer request is pending, then the FIFO is not cleared until the corresponding signal REQCLR has been cleared.

The bit-field FIFOCFG.CRBC can be configured how to clear the FIFO request.

The I2C-bus interface should know when it is receiving the last character of a data packet from the FIFO. Therefore the software should define a transmit packet size in bit-field TPS of register TPSCTRL and then the FIFO controller indicates the last character of the packet to the I2C kernel by setting the EOTXP (end of transmit packet) signal.

Since TPS is double buffered the software can write the size of the next packet into TPSCTRL.TPS as soon as the TPSCTRL.TPS value of the current packet has been loaded into the TPS counter, that is reading TPS returns 0. It loads the new TPS value as soon as it has room in the FIFO.

Also the characters of the next packet can be moved to the FIFO immediately after the current packet characters as indicated in the following figure . If byte or half word alignment is used as described in Transmit data alignment , then the characters of two different packets must not be aligned in a single stage.

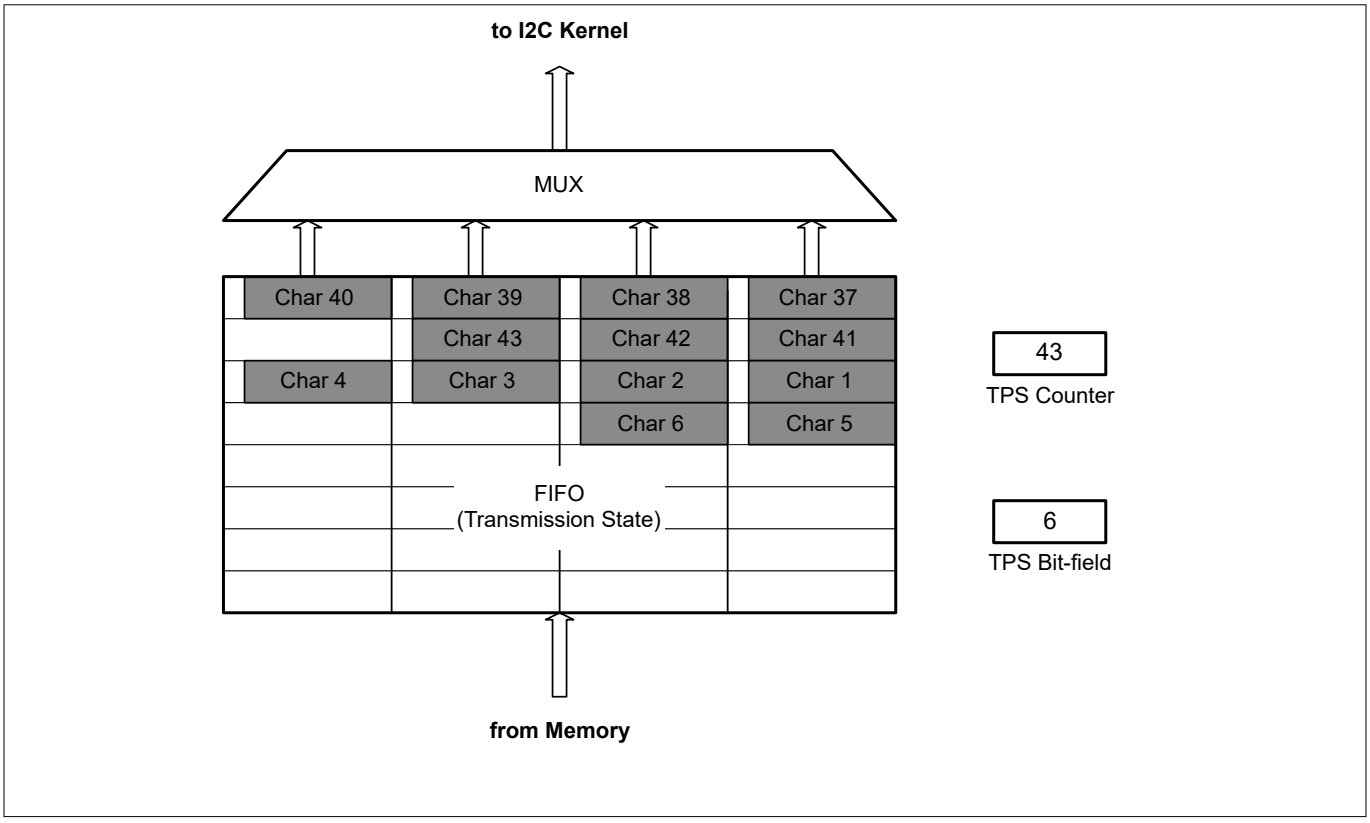


Figure 721 FIFO containing data of two different transmit packets

Error handling

- If a TX FIFO overflow occurs (that is the software writes data too fast to the FIFO), the new incoming character is discarded and a TX FIFO overflow interrupt request is generated
- If a TX FIFO underflow occurs (that is the I2C kernel tries to read from an empty FIFO), the I2C kernel generates a TX FIFO underflow interrupt request

Related information

Transmit request generation on page 5456
Transmit data alignment on page 5456

31.3.1.5.2 Transmit request generation

For the following it is assumed that the FIFO is used as flow controller, selected through bit FIFOCFG.TXFC.

In this case the bit-field TPSCTRL.TPS is not only used for the generation of the EOTXP (end of transmit packet) signal , but also to initiate the whole data transfer. That is the software indicates that it wants to transmit a data packet by writing the packet size to the bit-field TPSCTRL.TPS. Then the FIFO unit asserts burst requests BREQ until the amount of data still to be transferred is less than or equal to the burst size set in bit-field FIFOCFG.TXBS. At this point, if the remaining data is equal to the burst size, then a last burst request LBREQ is issued. Otherwise, single requests SREQ and a last single request LSREQ will be issued.

If a data transfer (BREQ, LSBREQ, SREQ, LSREQ) request is pending, then no more request or interrupt will be done until the corresponding signal REQCLR has been cleared. The CPU (software) or DMA have to transfer the data to the FIFO and clear the generated interrupt.

***Notes**:*

1. *Preloading TPSCTRL.TPS with the packet size of the next frame, before the current packet has been transmitted completely, enables a continuous data throughput (for example of audio data)*
2. *If the FIFO is not used as flow controller, the I2C-bus interface should use bit-field TPSCTRL.TPS to set the EOTXP signal. A burst request BREQ is generated each time the number of empty FIFO stages is equal or greater than the burst size set in bit-field FIFOCFG.TXBS and the previous request was cleared. No single requests SREQ and LSREQ will be issued*

Related information

Data transmission on page 5455

31.3.1.5.3 Transmit data alignment

Depending on bit-field TXFA (TX FIFO Alignment) in register FIFOCFG, the FIFO can deal with byte aligned, half word aligned or word aligned characters (or even more) as it is described in the following.

***Note**:* *Byte alignment is used as synonym for character alignment, half word alignment for character alignment of two data characters, and word alignment for character alignment of four characters.*

Byte aligned TX characters

If the TX characters are byte aligned, then the FIFO extracts the bytes and shifts them to the I2C kernel through a multiplexer as it is shown in the following figure .

Since the number of characters of the packet to be transmitted is not necessarily a multiple of 4, the last word must be filled up with dummy bytes if necessary. The FIFO transmits only the valid bytes of the last word depending on the setting of the bit-field TPSCTRL.TPS.

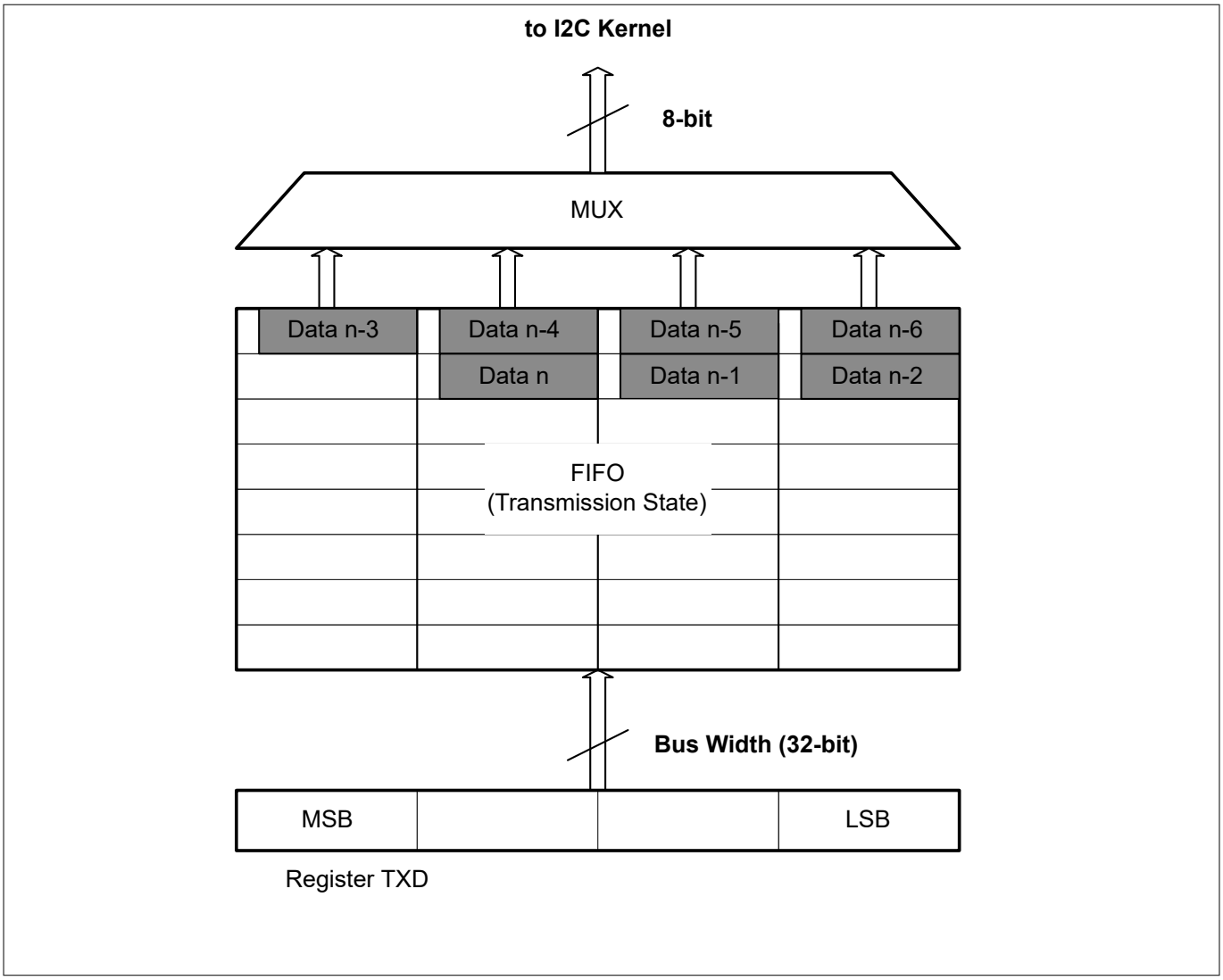


Figure 722 FIFO in transmission state with byte aligned data

Half word aligned

If the TX characters are half word aligned, then the FIFO extracts the half words and shifts them to the I2C kernel through a multiplexer as it is shown in the following figure.

***Note**: The I2C kernel extracts the right data bits from the half word it receives from the FIFO.*

Since the number of characters of the packet to be transmitted is not necessarily a multiple of 2, the last word must be filled up with a dummy half word if necessary. The FIFO transmits only the valid half words of the last word depending on the setting of the bit-field TPSCTRL.TPS .

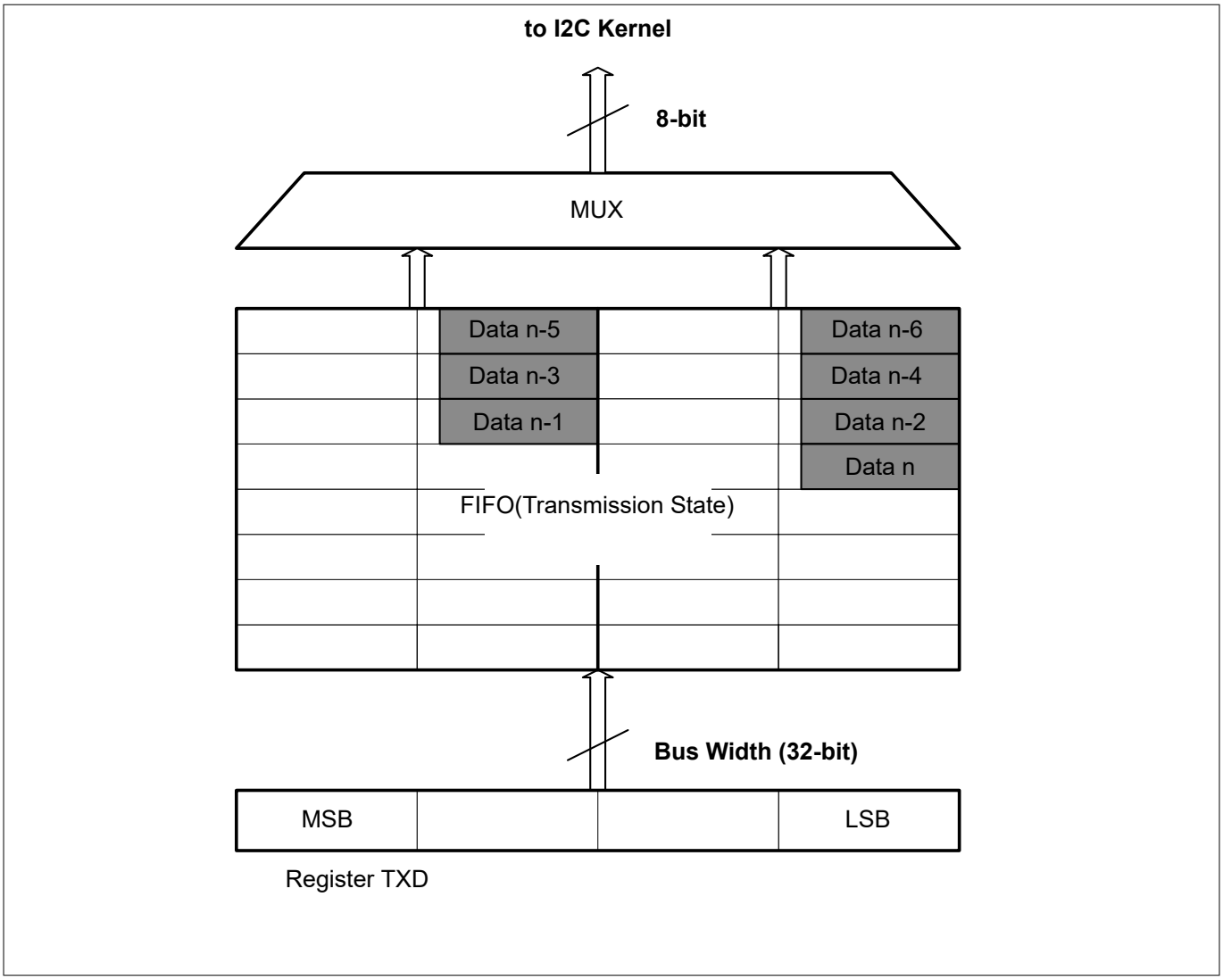


Figure 723 FIFO in transmission state with half word aligned data

Word aligned

If the TX characters are word aligned, then the FIFO shifts them to the I2C kernel as it is shown in the following figure.

***Note**: The I2C kernel extracts the right data bits from the word it receives from the FIFO.*

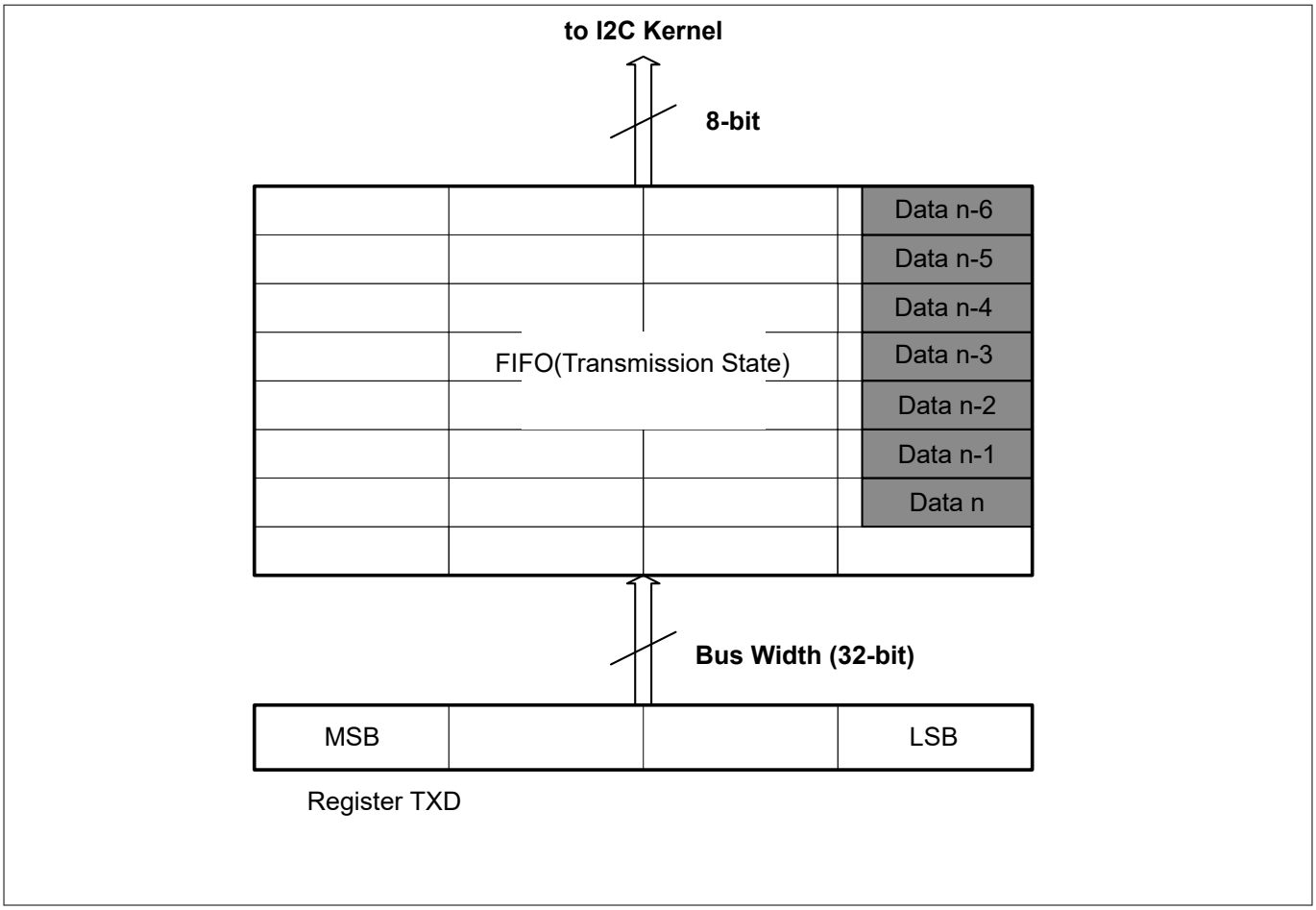


Figure 724 FIFO in transmission state with word aligned data

31.3.1.5.4 Data reception

The software can read received data from the FIFO by reading the Reception Data Register RXD. Reading data from the FIFO should be done according to the data transfer request signals generated by the FIFO controller. The register FFSSTAT indicates the number of filled FIFO stages. The data alignment within the FIFO is described in Receive data alignment . The I2C kernel shifts data into the FIFO by means of the handshake signals RX_FIFO_rdy and RX_Data_ind.

Before the first reception starts, the I2C kernel has to set the RXS_set line. This causes the assertion of RX_FIFO_rdy to indicate the I2C kernel that one data character can be written into the FIFO.

With the signal FIFO_flush the I2C kernel is able to clear the FIFO content. But if a data transfer request is pending and the FIFO is configured as flow controller, then the FIFO is not cleared until the corresponding signal REQCLR has been cleared.

Bit-field FIFOCFG.CRBC can be configured how to clear the FIFO request.

Error handling

- If the RX FIFO is completely full (that is the software reads out the data too slow), the I2C kernel generates an RX FIFO overflow interrupt request
- If an RX FIFO underflow occurs (that is the software reads from empty FIFO), an RX FIFO underflow interrupt request is generated
- If an error (including the FIFO overflow condition described above) is detected in the I2C kernel and the current reception is stopped, it has to generate an EORXP_ind signal, so that the remaining characters in the FIFO can be moved out by means of data transfer requests

Related information

Receive data alignment on page 5462

31.3.1.5.5 Receive request generation

If the whole data of the current received packet has been moved into the buffer, then the I2C kernel will set the EORXP_ind (end of receive packet indication) signal. The received characters, which are shifted in the FIFO, are counted by an RPS (received packet size) counter. If an EORXP_ind occurs, then the content of the RPS counter is moved to the register RPSSTAT and the counter is reset. If data alignment in the FIFO is used, then the RPSSTAT value can be used to check for valid characters in the last read word .

For the following it is assumed that the FIFO is used as flow controller, selected through bit RXFC of the FIFOCFG register.

The FIFO unit asserts burst requests BREQ each time the FIFO filling level is greater than the burst size set in bit-field RXBS of register FIFOCFG. An EORXP_ind causes the FIFO unit to assert burst requests BREQ until the amount of data still to be transferred is less than or equal to the burst size. At this point, if the remaining data is equal to the burst size, a last burst request LBREQ is issued. Otherwise, single requests SREQ and a last single request LSREQ will be issued.

If a data transfer (BREQ, LSBREQ, SREQ, LSREQ) request is pending, then no more request or interrupt will be done until the corresponding signal REQCLR has been cleared. The CPU (software) or DMA have to get the data from the FIFO and clear the generated interrupt.

When the packet has been shifted out of the FIFO by software, the FIFO controller provides the EORXP_ack (end of receive packet acknowledge) signal to the I2C kernel, which allows the kernel to set the next EORXP_ind.

The next received packet may be shifted into FIFO immediately after the current packet, so that the FIFO contains data of two different packets. But the EORXP_ind of the next packet must not be asserted before the EORXP_ind of the current packet has been acknowledged. The RPSSTAT.RPS counter is counting the characters of the next packet, while in the register RPSSTAT the packet size of the current packet remains readable. The first character of the new packet is always shifted in a new FIFO stage as indicated in the following figure.

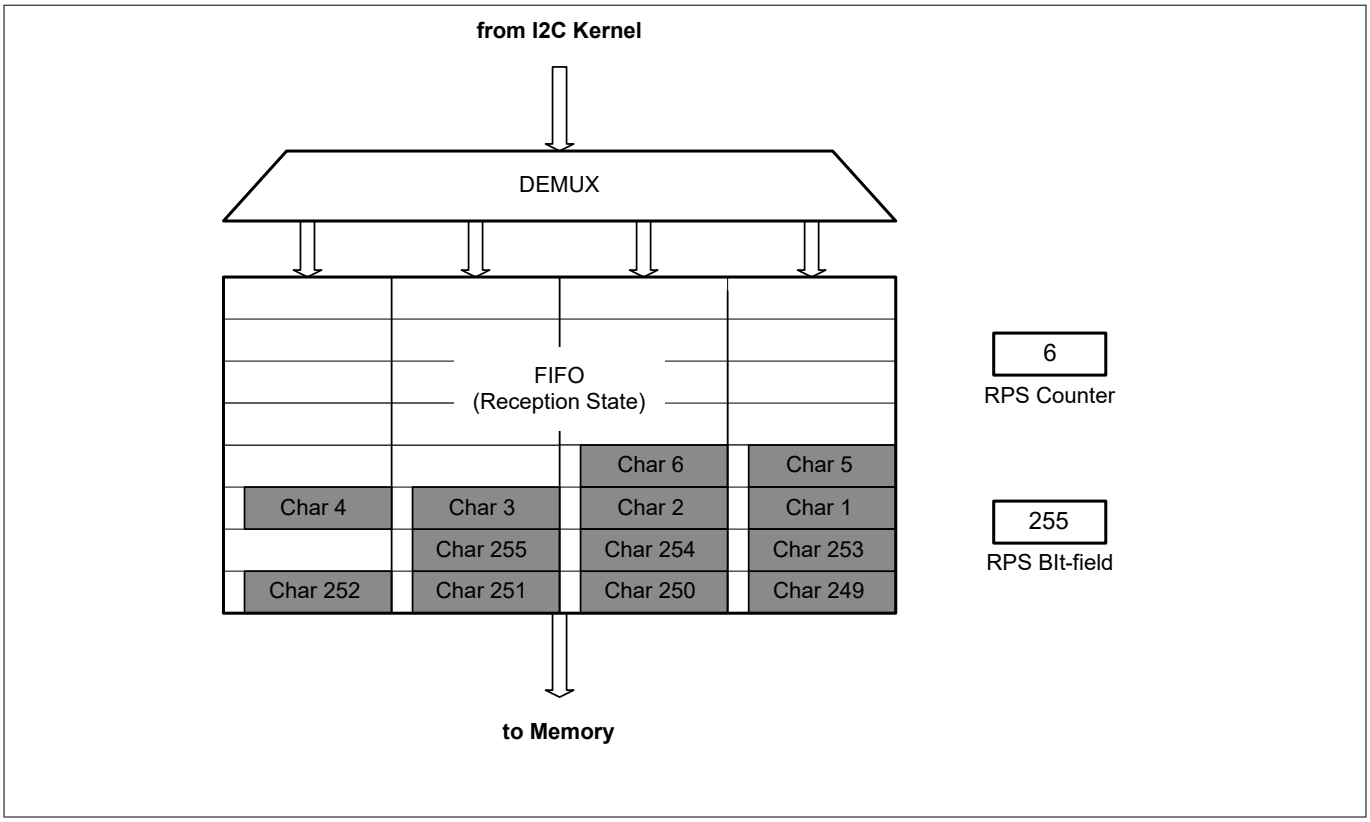


Figure 725 FIFO containing data of two different receive packets

If the number of the received characters of one packet is equal to the maximum received packet size, which is defined by bit-field MRPS in register MRPSCTRL, then the FIFO controller gets into the state, where it generates LBREQ or SREQ and LSREQ. If the I2C kernel generates an EORXP_ind in this state, then this indication is valid for the received packet and the FIFO accepts no more characters from the I2C kernel until the current packet has been moved out of the FIFO or the start of a new packet has been detected by asserting RXS_set.

The bit-field MRPSCTRL.MRPS is double buffered, that is the software can write the MRPSCTRL.MRPS value of the next packet as soon as the current MRPSCTRL.MRPS value has been loaded, that is reading MRPS returns 0. It loads the new MRPS value as soon as it has room in the FIFO. If an MRPS overflow occurs (one character before the last character is received), then the MRPS_stop trigger to the I2C kernel is generated. But if the next MRPS value is written to MRPSCTRL.MRPS before the last data character of current frame has been completely received, then no MRPS_stop is generated. The I2C-bus interface will use the MRPS_stop trigger to stop the reception of the corresponding data packet.

But the MRPSCTRL.MRPS bit-field can also be used to initiate a reception, when the peripheral is master-receiver. Therefore writing to MRPS generates a trigger MRPS_start to the I2C kernel. The next MRPS value written to the MRPS bit-field generates only an MRPS_start when an MRPS_stop was generated before.

The MRPS features can only be used if the FIFO is the flow controller.

***Notes**:*

1. *When the MRPS control feature of the FIFO is used to limit the packet size of a continuous incoming data stream (MRPS is only written once and left unchanged), characters exceeding MRPS are discarded. The FIFO can not handle more than two packets. When the FIFO is filled with data characters of two packets, a further RXS_set will be overseen by the FIFO. Additionally, the FIFO does not set the RX_FIFO_rdy*
2. *If the FIFO is not used as flow controller, a single request SREQ is always generated as soon as the readable FIFO stage is filled up with characters. Additionally, a burst request BREQ is generated as soon as the number of filled FIFO stages is equal to or greater than the burst size set in bit-field RXBS of register FIFOCFG. When the I2C kernel sets EORXP_ind, the FIFO controller provides the EORXP_ack (end of receive packet acknowledge) signal to the I2C kernel, which allows the kernel to set the next EORXP_ind*

Related information

Receive data alignment on page 5462

31.3.1.5.6 Receive data alignment

Depending on bit-field FIFOCFG.RXFA (RX FIFO Alignment), the FIFO can deal with byte aligned, half word aligned or word aligned characters (or even more) as it is described in the following.

Byte aligned

If the RX data should be byte aligned, then the characters from the I2C kernel are aligned and shifted to the FIFO by a de-multiplexer as it is shown in the following figure.

Since the number of characters of the received packet is not necessarily a multiple of 4, the upper bytes of the last word can be invalid. The software has to check for invalid bytes of the last word by means of the bit-field RPSSTAT.RPS (received packet size).

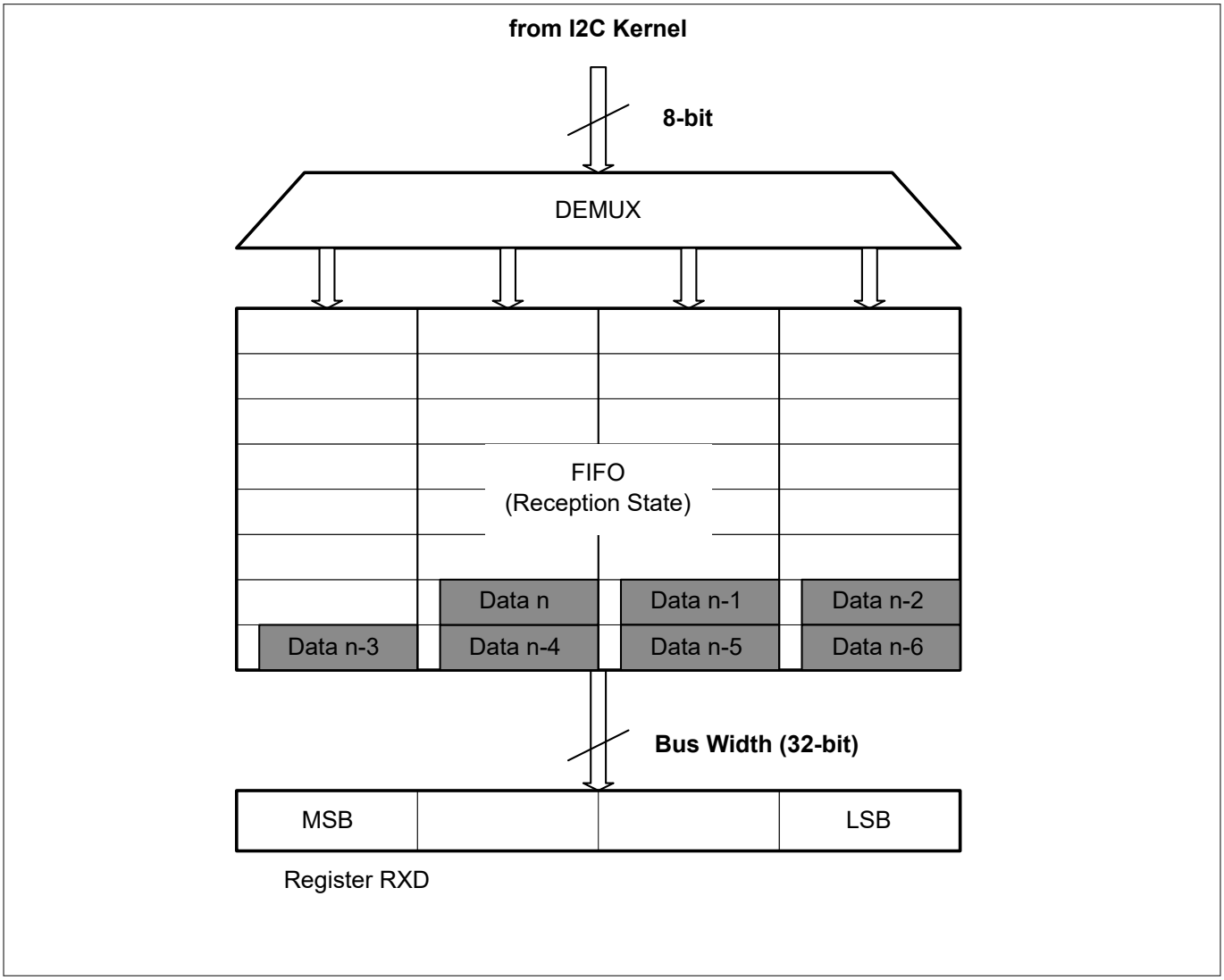


Figure 726 FIFO in reception state with byte aligned data

Half word aligned

If the RX data should be half word aligned, then the characters from the I2C kernel are aligned and shifted to the FIFO by a de-multiplexer as it is shown in the following figure .

***Note**: The I2C kernel fills up the half words with dummy bits.*

Since the number of characters of the received packet is not necessarily a multiple of 2, the upper bytes of the last word can be invalid. The software has to check for invalid half words of the last word by means of the bit-field RPS (received packet size).

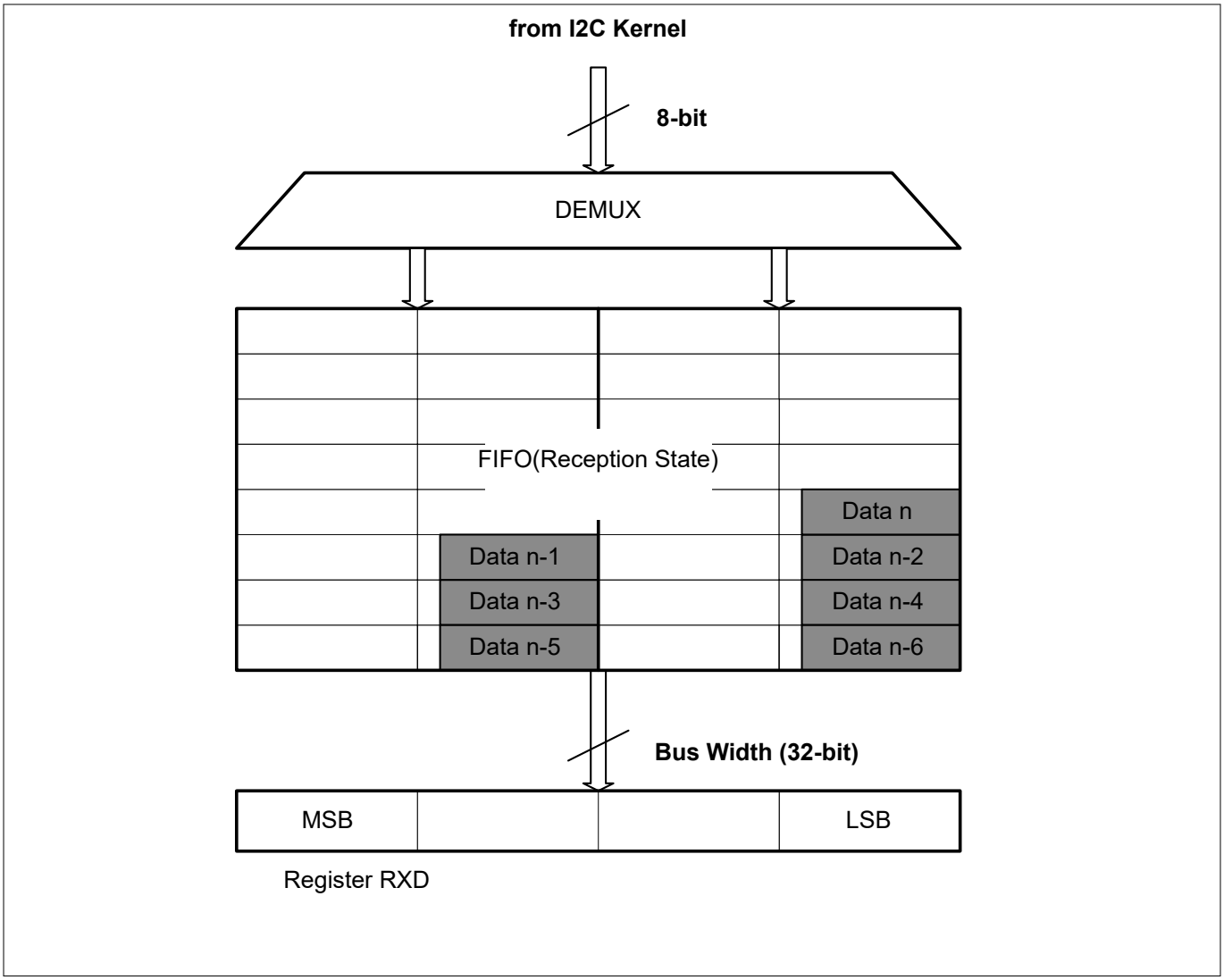


Figure 727 FIFO in reception state with half word aligned data

Word aligned

If the RX data should be word aligned, then the characters from the I2C kernel are shifted to the FIFO as it is shown in the following figure.

***Note**: The I2C kernel fills up the words with dummy bits.*

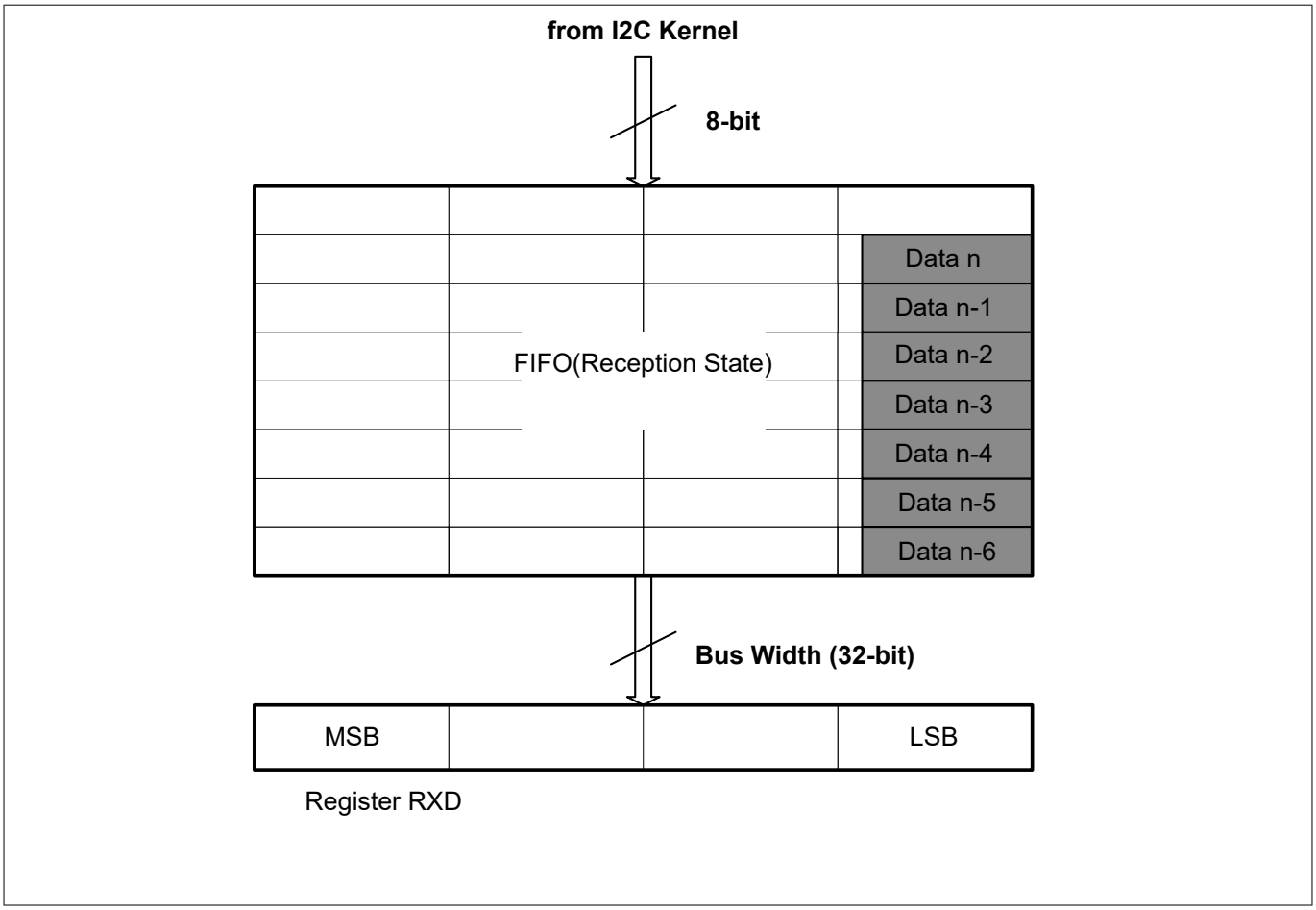


Figure 728 FIFO in reception state with word aligned receive data

31.3.1.5.7 Switching between transmission and reception

Initially, the FIFO is in the TX state, so that the software can initiate a TX transfer simply by writing data to the bit-field TXD.TXD . If the I2C kernel wants the FIFO to switch to the RX state in the meantime, then it can set the RXS_set line and even if transmission is ongoing the TX transfer is aborted. This causes the FIFO to flush its content, switch to the RX state and reset the received packet size counter. If the software wants to write to the register TXD when the FIFO is in RX state, then this causes a bus error. As soon as the FIFO has received an EORXP_ind signal from the kernel and the software has moved all RX characters out of the FIFO through the register RXD, the FIFO automatically switches back into the TX state.

If the software writes to the register TPSCTRL when the FIFO is in the RX state, the TPS value is pending until the FIFO returns to the TX state. Then the transmission is initiated, if the FIFO is the flow controller.

If the FIFO is flow controller and the software writes to the register MRPSCTRL, then an MRPS_start is generated independent of the state of the FIFO.

With the signal FIFO_flush the I2C kernel is able to clear the FIFO content. But if a data transfer request (xREQ) is pending, then the FIFO is not cleared until the corresponding signal REQCLR has been cleared.

If the FIFO has been cleared in the RX state, then it switches back to the TX state afterwards.

31.3.1.5.8 Switching between reception and transmission

When the I2C kernel is changing from RX state to TX state the FIFO is flushed. In order to avoid loosing data when the FIFO is flushed, the software should proceed in the right sequence. In a scenario where the device is addressed as slave and is asked to return data, this new data must be entered in the FIFO only after detection of

the address and TX_END interrupt requests and then can transfer the data to the FIFO or can trigger the DMA that fills the FIFO for the TX transfer.

31.3.1.6 Service request block operation

The SRB (Service Request Block) of the I2C module is used to prepare the interrupt and data transfer requests for the interrupt controller.

31.3.1.6.1 Overview of service requests

The I2C service requests are partially combined in the SRB (see ERRIRQSM, ERRIRQSS and ERRIRQSC as well as PIRQSM, PIRQSS and PIRQSC). The figure below shows an overview of the service requests. The following table provides a list of all service requests of the I2C module.

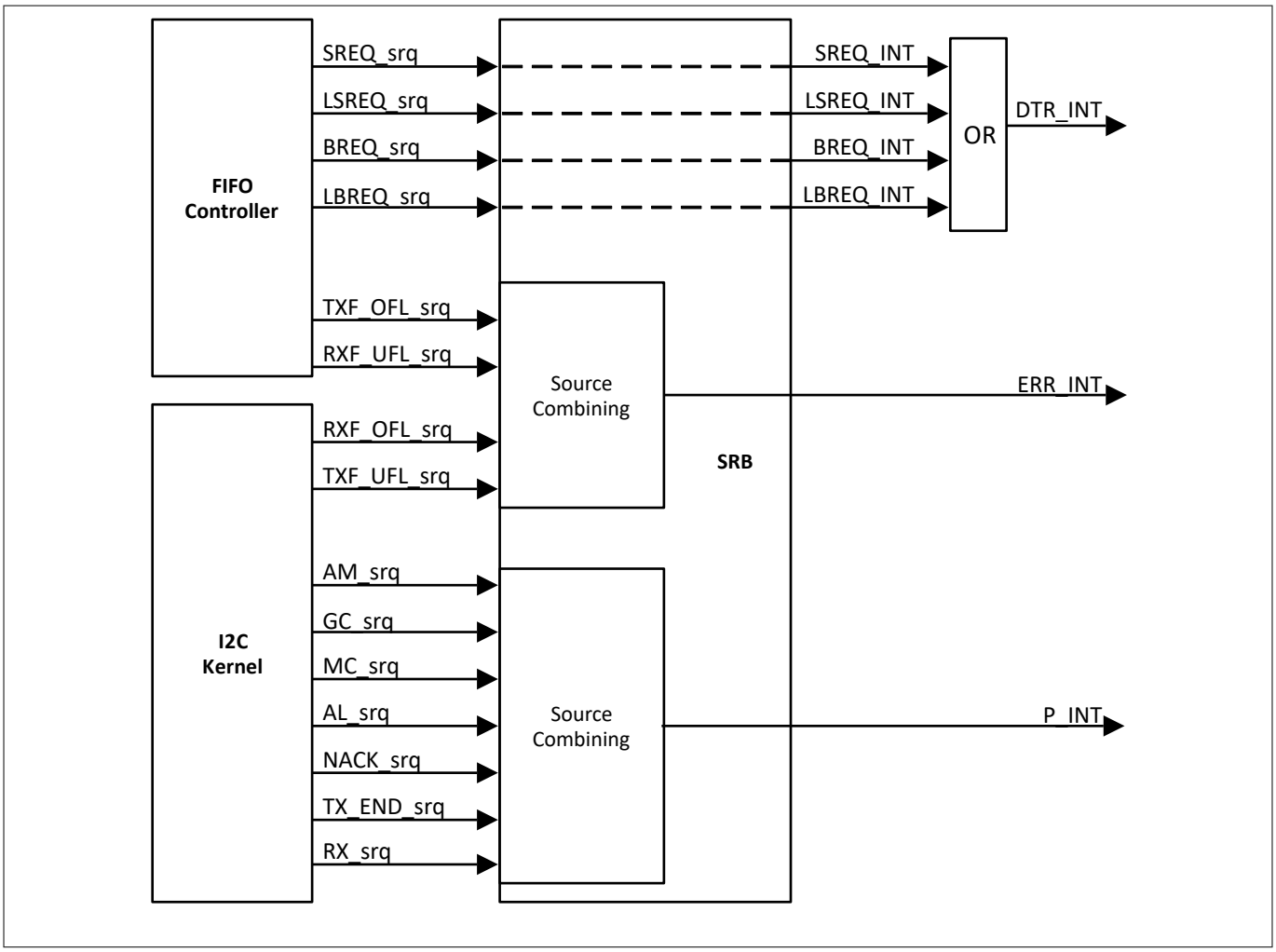


Figure 729 Overview of I2C module service requests

Table 1261 I2C module service requests

Service request	Interrupt request	Description
BREQ_srq	DTR_INT	Burst Data Transfer Request FIFO requests a transfer of a programmed burst number of words from or to the memory.

(table continues...)

Table 1261 (continued) I2C module service requests

Service request	Interrupt request	Description
LBREQ_srq	DTR_INT	Last Burst Data Transfer Request FIFO requests a last burst transfer from or to the memory.
SREQ_srq	DTR_INT	Single Data Transfer Request FIFO requests a single transfer of a word from or to the memory.
LSREQ_srq	DTR_INT	Last Single Data Transfer Request FIFO requests a last single transfer from or to the memory.
TXF_OFL_srq	ERR_INT	TX FIFO Overflow Request FIFO has detected a TX FIFO overflow
TXF_UFL_srq	ERR_INT	TX FIFO Underflow Request I2C kernel has detected a TX FIFO underflow. The transmission is finished after the current byte. A stop condition is generated if the kernel works as master. The kernel changes to listening state
RXF_OFL_srq	ERR_INT	RX FIFO Overflow Request I2C kernel has detected an RX FIFO overflow and the incoming character is discarded. The kernel puts a not-acknowledge on the bus and changes to listening state. A stop condition is generated if the kernel works as master
RXF_UFL_srq	ERR_INT	RX FIFO Underflow Request FIFO has detected an RX FIFO underflow
AM_srq	P_INT	Address Match Request Device (master or slave) has been addressed by a remote master (also indicated in bit-field BS in register BUSSTAT)
GC_srq	P_INT	General Call Request When the general call matching process is activated this interrupt indicates that another master has put a general call on the I2C-bus
MC_srq	P_INT	Master Code Request When the master code matching process is activated this interrupt indicates the appearing of a master code on the I2C-bus issued by a remote master. The request is generated after a not-acknowledge and the clock is released to high again
AL_srq	P_INT	Arbitration Lost Request Arbitration is lost after the device has tried to start a transmission on the I2C_bus
NACK_srq	P_INT	Not-acknowledge Received Request Not-acknowledge received when working as transmitter (that is RnW bit is 0)

(table continues...)

Table 1261 (continued) I2C module service requests

Service request	Interrupt request	Description
TX_END_srq	P_INT	Transmission End Request In master mode this event is produced by the I2C kernel to indicate that the transmission of the current packet has ended properly after the stop condition has been put on the I2C-bus or MASTER RESTART state has been entered. At this point, a restart condition can be generated or the connection can be finished by generating a stop condition. This request is created in master mode at any stop condition to indicate that the bus is free again and it could be obtained. In slave mode this event is produced by the I2C kernel if it was addressed by a master and the current transmission was terminated by a stop or restart condition
RX_srq	P_INT	Receive Mode Request I2C kernel indicates to the FIFO switching from transmit to receive mode. When FIFO is operating in non flow controller mode, this service request can be used to distinguish between transmit and receive FIFO data requests

The generation of these events is visualized in the Top-level state machine. The timing is shown in I2C kernel control logic.

Related information

I2C kernel control logic on page 5443

31.3.1.6.2 Interrupt service request structure

The I2C module provides level based service request lines, which can be processed by an interrupt controller. The service requests must be cleared in the interrupt service routine. For test purposes, all service requests can also be set by SW through a register bit. All service requests can be masked within the peripheral. Furthermore, requests that are not necessarily mutually exclusive are combined in order to reduce the number of request lines.

Flow of data transfer requests

The following figure shows the flow of a data transfer request, which comes from the FIFO controller.

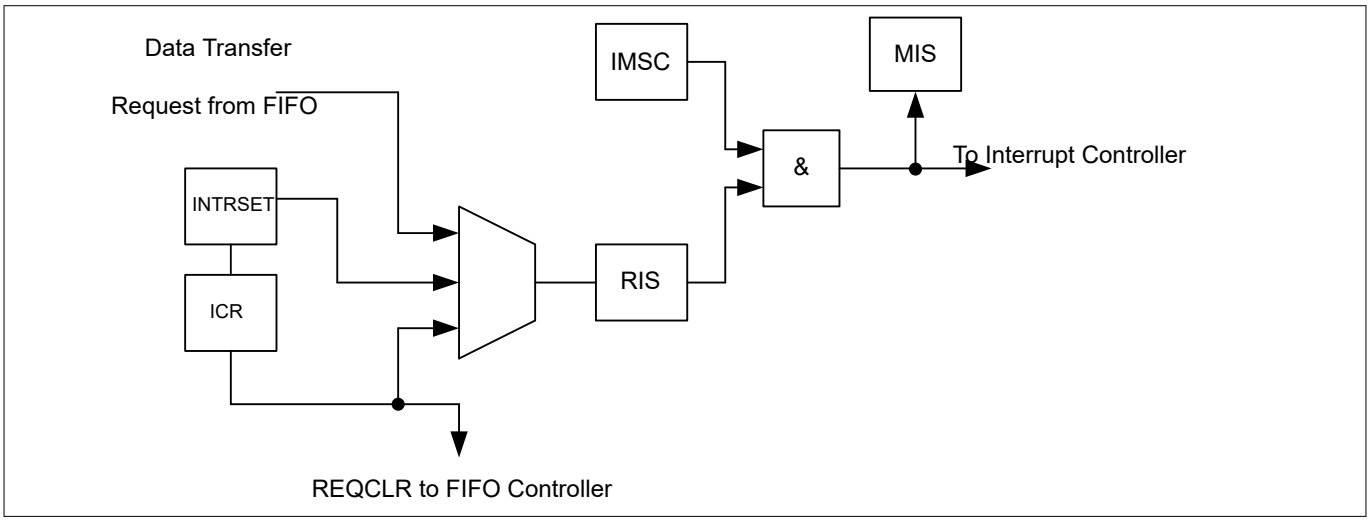


Figure 730 Data transfer request flow

A data transfer request sets the corresponding status bit in the Raw Interrupt Status Register RIS. The status bit can also be set by writing 1 to the corresponding bit in the Interrupt Set Register INTRSET. It will be cleared by writing 1 to the corresponding bit in the Interrupt Clear Register ICR.

The Interrupt Mask Control Register IMSC enables or disables the requests to the interrupt controller. The Masked Interrupt Status Register MIS contains the current masked values of the requests.

If a request is disabled through register IMSC while the request is active, the request will be removed but only until the IMSC bit is set again, unless the request has been cleared in the meantime.

If a request is disabled through register IMSC and the source becomes active, then the request bit will only be set in the RIS register. If the corresponding IMSC bit is later enabled, the request will consequently become active in the MIS register.

Before enabling a request, it is good practice to always first clear the corresponding bit in the RIS register through ICR.

Multiple source interrupt requests

For error and protocol interrupt requests which have multiple sources, the interrupt structure is extended. Additional register sets are implemented within the SRB.

The following figure gives an overview of the combining of several request sources to a single request.

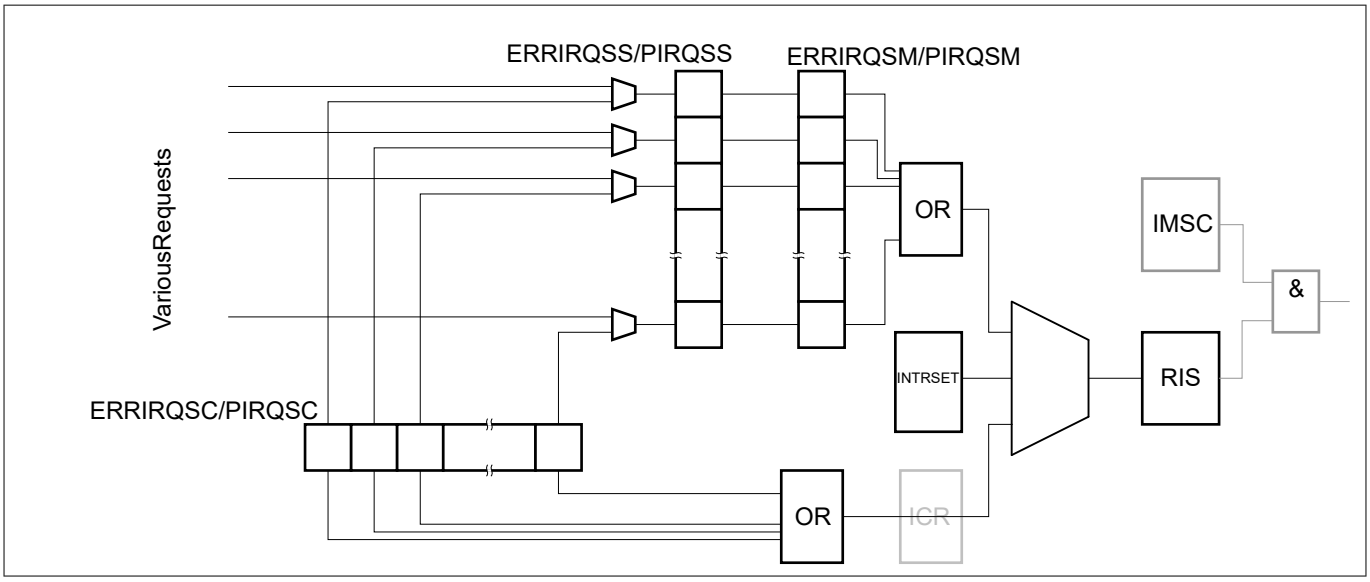


Figure 731 Interrupt request with multiple sources

An interrupt request sets the corresponding raw status bit in the Interrupt Request Source Status Register ERRIRQSS (for error) or PIRQSS (for protocol).

The status bits can be cleared through the Interrupt Request Source Clear Register ERRIRQSC (for error) or PIRQSC (for protocol). If no further error or protocol request sources are active, the whole error or protocol interrupt request is cleared. This register replaces the functionality of the bit I2C_ERR_INT (for error) or bit I2C_P_INT (for protocol) in the Interrupt Clear Register ICR.

The Interrupt Request Source Mask Register ERRIRQSM (for error) or PIRQSM (for protocol) enables or disables the interrupt requests. The request lines from the enabled sources are combined (OR) to a single level sensitive request line, which acts as input for the corresponding bit I2C_ERR_INT (for error) or I2Cm_bit I2C_P_INT (for protocol) in the Raw Interrupt Status Register RIS.

31.3.2 I2C module implementation

This section describes I2C module interfaces with the clock control, port control and interrupt control.

31.3.2.1 Interfaces of the I2C module(s)

The following figure shows the specific implementation details and interconnections of the I2C module(s):

- A clock control block generates the clock signals required for the module
- For the I2C-bus lines SCL and SDA, different I/O lines can be selected
- The interrupt outputs of the module are connected to the interrupt control unit

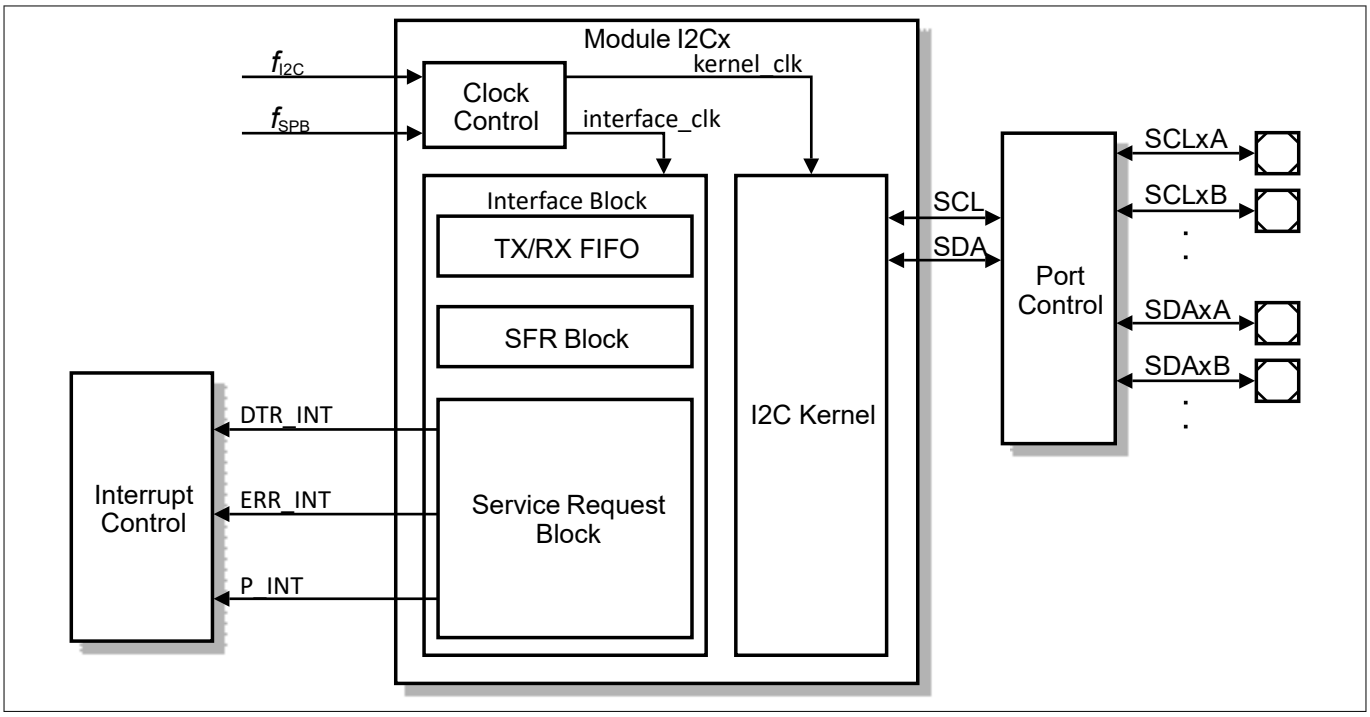


Figure 732 I2C module implementation and interconnections

31.3.2.2 Module clock control

The clock control allows the programmer to adapt the peripheral's functionality and power consumption to the application’s requirement. By programming the kernel’s operating frequency, an optimal ratio between power consumption, EMC behavior and functionality can be achieved. Furthermore, for power saving reasons the peripheral can be disabled as a whole by switching off the peripheral’s clocks through clock gating cells.

A simplified description of the clock control is shown in the following figure . See register CLC1 for a description of the clock control parameters.

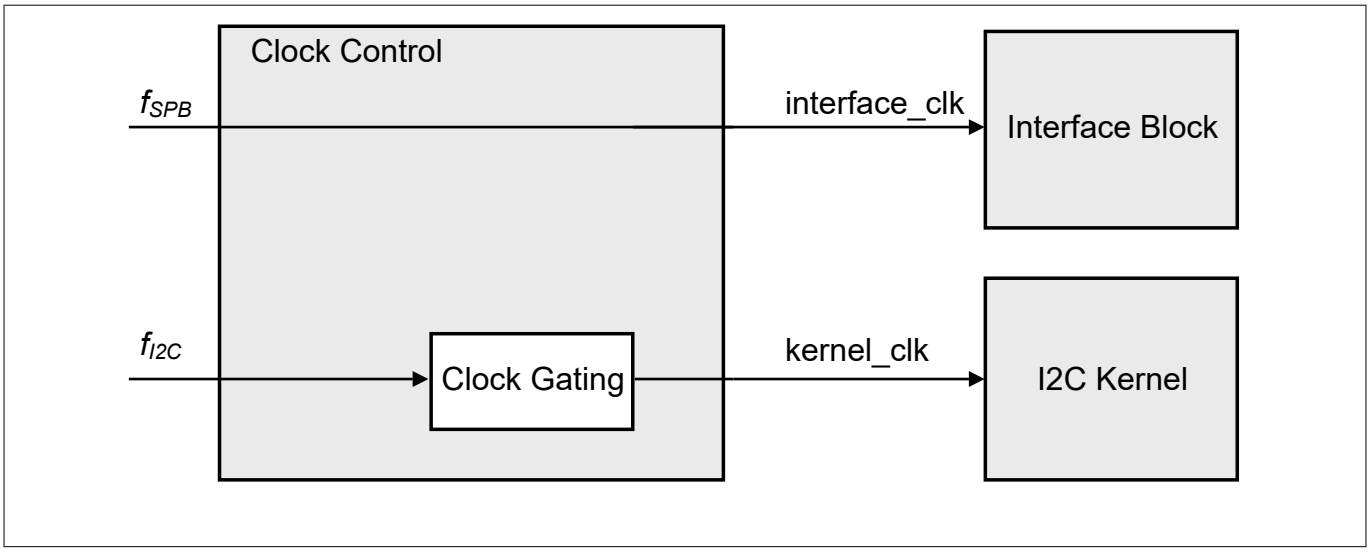


Figure 733 Module clock control

The following clocking modes are supported for the peripheral:

- Disabled mode
- OCDS suspend mode

Disabled mode

In Disabled Mode, the clocks are stopped and register write accesses are not possible (except access to CLC1 register itself).

There are several ways possible to bring the module into disabled mode:

- Software disable
  Set CLC1.DISR = 1 (software disable request)
- Hardware disable
  Must be enabled through CLC1.EDIS = 0
  Drive input signal sleep_n_i = 0(active low: 0= disable; 1 = enable)

In all cases, current bus accesses are finished and a kernel handshake is issued to make sure that the state machine is in a safe state (same as CLC1.FSOE = 0 in OCDS Suspend Mode; see description below).

Module is in Disabled Mode after reset and must be enabled by setting CLC1.DISR = 0 and CLC1.RMC > 0.

OCDS suspend mode

In this mode the clocks are disabled. If bit SPEN in register CLC1 is set, the suspend mode can be entered.

In OCDS suspend mode, all registers of the suspended module can be read. Note that each read during disabled clock can not affect hardware. For example multiple reads to FIFO port registers always return the same value as the FIFO control is not working.

No write access to the registers is supported during suspend mode, except writes to the CLC1 register. For writing other registers the suspend mode for the module has to be removed first by clearing bit SPEN. After the write access is executed, bit SPEN should be set again.

OCDS suspend mode supports two different disable features, selectable by bit FSOE in register CLC1:

***Note**: To write to SPEN or FSOE, bit SBWE in register CLC1 must also be set in the same write access.*

- Secure clock switch off
  Disabling the clock through secure clock switch off additionally activates the handshake mechanism with the peripheral. The peripheral is switched to a secure state before disabling the clock

With secure clock switch off:

- Internal state machine will wait until a safe state is reached
- SCL_out will be tied to 0 to delay I2C bus operation

***Note**: This feature is only available in slave mode. In master mode, there is no difference and I2C will always perform "fast clock switch off"*

- Fast clock switch off

  Selecting the fast clock switch off stops the clock immediately after all pending read and write accesses are finished (including synchronization mechanisms with the I2C kernel)

  With fast clock switch off:

  - I2C will go into the suspend mode immediately after the current register access operation is finished. FSM will remain in the latest state. In master mode, all clocks will be stopped, SCL will hold latest level. In slave mode, SCL_out will be tied 0 to delay I2C bus operation
  - Depending on timing of the input signal of OCDS_suspend_i I2C bus protocol may be violated

  ***Notes**:*

  ***1.** This also applies when I2C is programmed as master but accessed as slave from another master*

  ***2.** SCL = 0 is only driven when bus operation was detected by a start condition*

31.3.2.3 Port and I/O line control

Not only the interconnections between the I2C module and the port I/O lines have to be configured, but also the function and the characteristics of the port pins. The following control operations must be executed for the used port pins:

- Selection of I2C module through output port multiplexer
- Configuration of output port function with open drain
- Configuration of pad driver characteristics
- Selection of I2C input lines

31.3.2.4 Interrupt control

The interrupt functionality is described in more detail under the chapter interrupt router (IR).

The signal names used in the interrupt router description are explained in the following table .

Table 1262 Interrupt router signal names for I2Cm module (m is the instance number)

Signal name	Name in module	Explanation
SRC_I2CmDTR	DTR_INT	Data transfer
SRC_I2CmERR	ERR_INT	Error
SRC_I2CmP	P_INT	Protocol

31.3.3 Module integration overview

This section describes the topics regarding the integration of the module on chip. The I2C module consists of a delivery providing an AHB bus interface integrated into the FPI bus system by using abridge between FPI and AHB.

The bridge contains a slave bus interface providing translation between the FPI and the AHB bus protocols. The module supports additionally the following features:

- Power saving (sleep mode)
- Debug suspend
- Local module reset
- PISEL - port input selection regarding the serial data input

***Note**:* *The FPI slave interface has support for read modify write (RMW) transactions. This type of transaction shall not be used for I2C module and kernel registers which can be written by the kernel, indicated by bit-fields of type rwh . The reason for this limitation is a potential modification of the register content during the transaction.*

The power saving and debug suspend features are supported by the AHB based I2C module itself, and the local module reset and PISEL features are supported by the bridge between FPI and AHB, see the following figure.

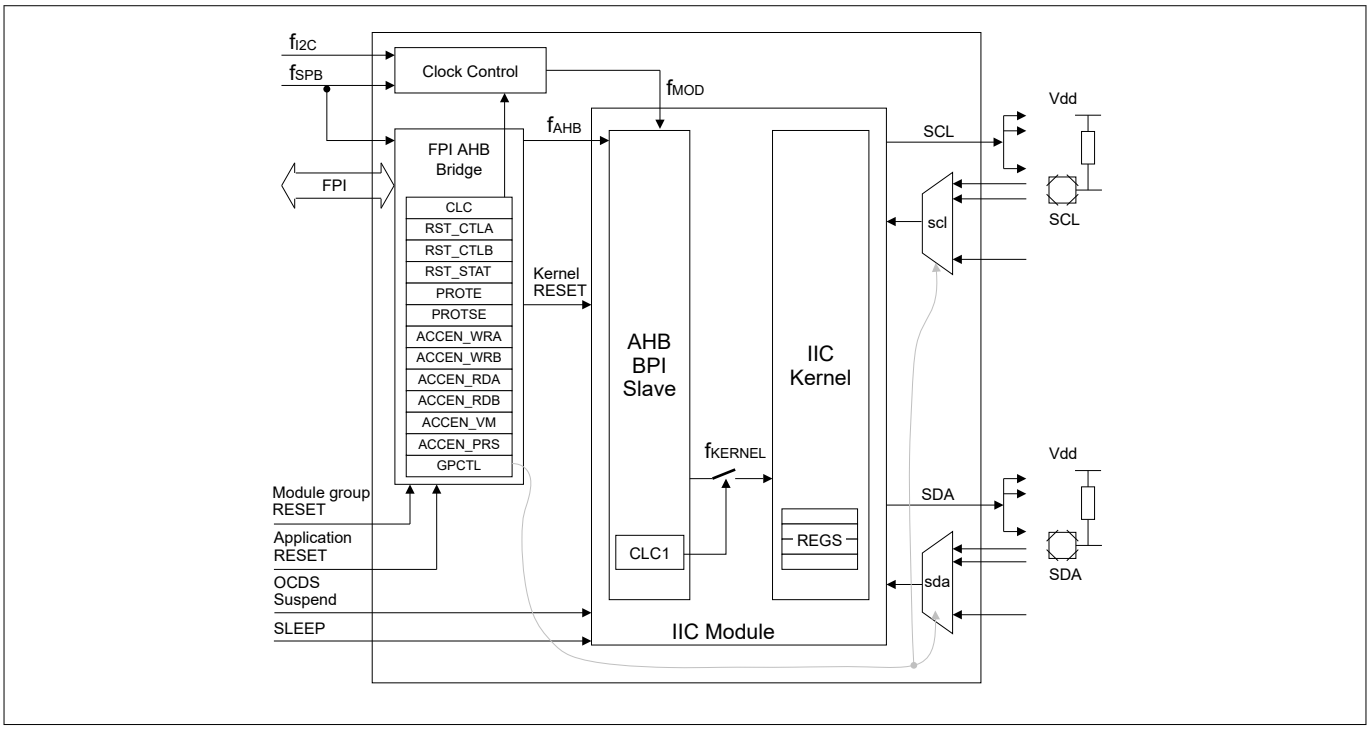


Figure 734 Integration overview

Related information

Registers overview - I2C (ascending offset address) on page 5475

31.4 Registers

31.4.1 Registers overview

This section describes the internal registers of the I2C module. All register names described in this section are also referenced in other parts of the I2C by the module name prefix “I2Cm_”. For an overview of all internal module registers, please refer to the Registers overview table.

Special I2C register access conditions

Besides the general register protection, the I2C module has two main modes that must be considered when programming the peripheral:

- Configuration Mode: In this mode, the peripheral can be prepared for transmission and reception through the configuration registers, which are only writable in this mode. The peripheral is in the configuration mode when bit RUN is set to 0
- Run Mode: In this mode, the peripheral is ready to transmit or receive data. Its configuration registers are locked for write access which will generate a bus error. The peripheral is in the run mode when bit RUN is set to 1

I2C registers overview

The I2C registers are grouped in to the following blocks (see figure below):

- Bus Peripheral Interface Registers
- Global Module Control Registers
- FIFO Registers
- Basic Interrupt Registers
- Error Interrupt Source Registers
- Protocol Interrupt Source Registers

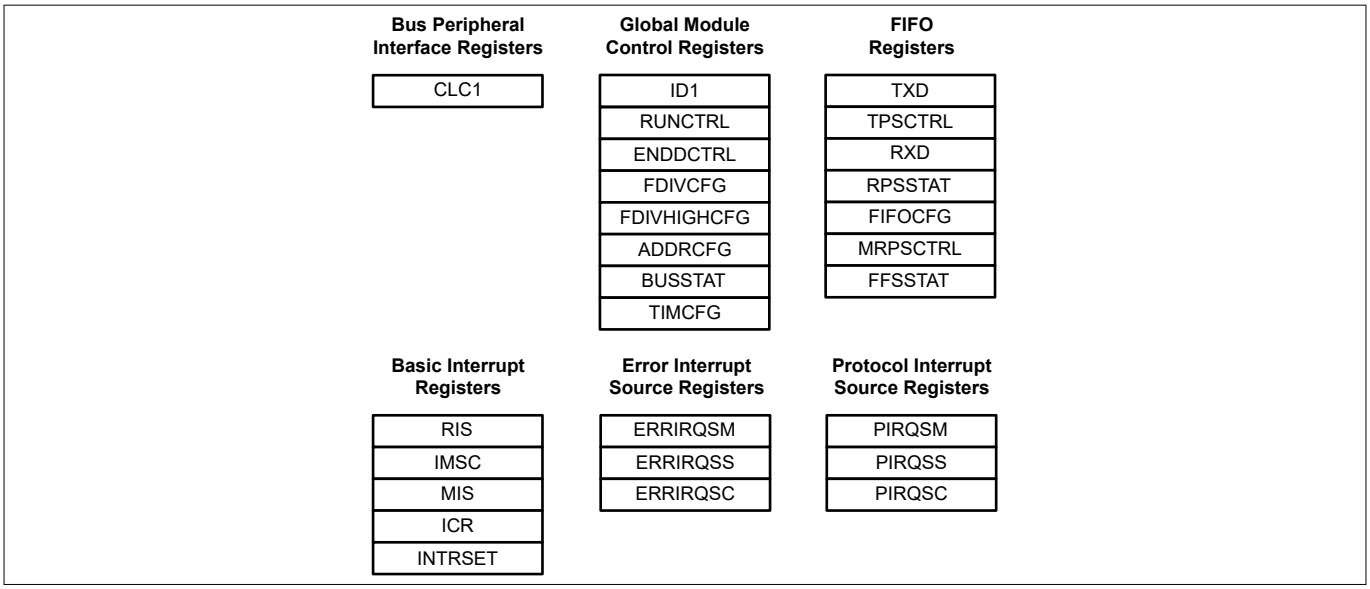


Figure 735 I2C registers

System registers overview

Figure 736 provides an overview of the remaining system registers of the I2C.

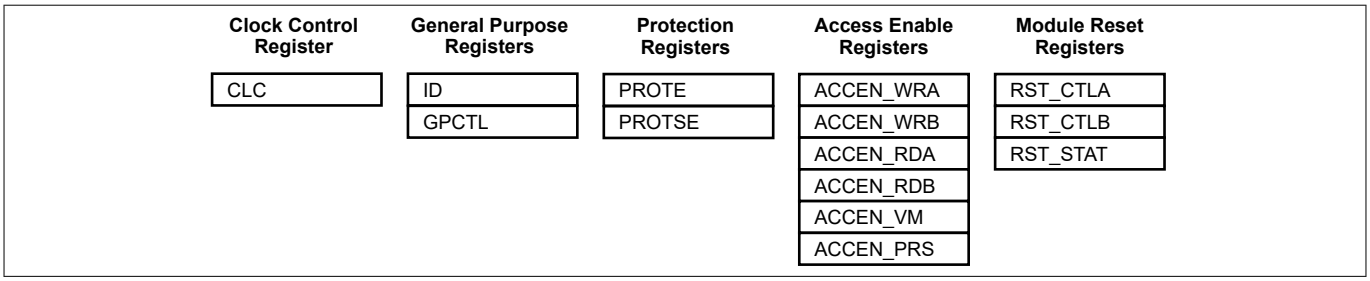


Figure 736 System registers

31.4.1.1 Register overview - access mode glossary

Table 1263 Register overview - access mode glossary

Keyword	Description
E	Access protection using PROT register PROTE.
SE	Access protection using PROT register PROTSE.
APU-P	Protection group consisting of registers ACCEN_WRA, ACCEN_WRB, ACCEN_RDA, ACCEN_RDB, ACCEN_VM, ACCEN_PRS.
P	Access protection using *APU-P* registers.
U	No access restrictions.
BE	Always returns a Bus Error.
nBE	Indicates that no Bus Error is generated when accessing this address range, even though it is either an access to an undefined address or the access does not follow the given rules.
PROT	Access restrictions as defined in the PROT register access rules.

31.4.1.2 Registers overview - I2C (ascending offset address)

Table 1264 Registers overview - I2C (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CLC1	Clock control 1 register	00000_H	U	P	Kernel Reset	5477
ID1	Module identification register	00008_H	U	BE	Kernel Reset	5478
RUNCTRL	RUN control register	00010_H	U	P	Kernel Reset	5479
ENDDCTRL	End data control register	00014_H	U	P	Kernel Reset	5480
FDIVCFG	Fractional divider configuration register	00018_H	U	P	Kernel Reset	5481
FDIVHIGHCFG	Fractional divider high-speed mode configuration register	$0001C_H$	U	P	Kernel Reset	5481
ADDRCFG	Address configuration register	00020_H	U	P	Kernel Reset	5482
BUSSTAT	Bus status register	00024_H	U	BE	Kernel Reset	5483
FIFOCFG	FIFO configuration register	00028_H	U	P	Kernel Reset	5484

(table continues...)

Table 1264 (continued) Registers overview - I2C (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
MRPSCTRL	Maximum received packet size control register	$0002C_H$	U	P	Kernel Reset	5485
RPSSTAT	Received packet size status register	00030_H	U	BE	Kernel Reset	5486
TPSCTRL	Transmit packet size control register	00034_H	U	P	Kernel Reset	5486
FFSSTAT	Filled FIFO stages status register	00038_H	U	BE	Kernel Reset	5487
TIMCFG	Timing configuration register	00040_H	U	P	Kernel Reset	5487
ERRIRQSM	Error interrupt request source mask register	00060_H	U	P	Kernel Reset	5489
ERRIRQSS	Error interrupt request source status register	00064_H	U	BE	Kernel Reset	5490
ERRIRQSC	Error interrupt request source clear register	00068_H	U	P	Kernel Reset	5491
PIRQSM	Protocol interrupt request source mask register	00070_H	U	P	Kernel Reset	5491
PIRQSS	Protocol interrupt request source status register	00074_H	U	BE	Kernel Reset	5492
PIRQSC	Protocol interrupt request source clear register	00078_H	U	P	Kernel Reset	5494
RIS	Raw interrupt status register	00080_H	U	BE	Kernel Reset	5495
IMSC	Interrupt mask control register	00084_H	U	P	Kernel Reset	5496
MIS	Masked interrupt status register	00088_H	U	BE	Kernel Reset	5497
ICR	Interrupt clear register	$0008C_H$	U	P	Kernel Reset	5498
INTRSET	Interrupt set register	00090_H	U	P	Kernel Reset	5499
TXD	Transmission data register	08000_H	nBE	P	Kernel Reset	5500
RXD	Reception data register	$0C000_H$	U	BE	Kernel Reset	5500
CLC	Clock Control Register	10000_H	P	P, SV, E	Application Reset	5501
ID	Module Identification Register	10008_H	P	BE	Application Reset	5501
RST_CTRLA	Reset Control Register A	$1000C_H$	P	P, SV, E	Application Reset	5502
RST_CTRLB	Reset Control Register B	10010_H	P	P, SV, E	Application Reset	5502

(table continues...)

Table 1264 (continued) Registers overview - I2C (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
RST_STAT	Reset Status Register	10014_H	P	BE	Application Reset	5503
PROTE	PROT Register Endinit	10020_H	U	SV, PROT	Application Reset	5504
PROTSE	PROT Register Safe Endinit	10024_H	U	SV, PROT	Application Reset	5505
ACCEN_WRA	Write access enable register A	10040_H	U	SE, SV	Application Reset	5507
ACCEN_WRB	Write access enable register B	10044_H	U	SE, SV	Application Reset	5508
ACCEN_RDA	Read access enable register A	10048_H	U	SE, SV	Application Reset	5508
ACCEN_RDB	Read access enable register B	$1004C_H$	U	SE, SV	Application Reset	5509
ACCEN_VM	VM access enable register	10050_H	U	SE, SV	Application Reset	5509
ACCEN_PRS	PRS access enable register	10054_H	U	SE, SV	Application Reset	5510
GPCTL	General purpose control register	10060_H	SV	SV, P	Application Reset	5511

31.4.1.3 Clock control 1 register

This register controls the clock gating and dividing circuitry of the peripheral.

CLC1 Offset address: 00000_H

Clock control 1 register Kernel Reset value: $0000\ 0003_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								RESERVED							
r								r							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RMC								0		FSOE	SBWE	EDIS	SPEN	DISS	DISR
rwh								r		rw	w	rw	rw	rh	rw

Field	Bits	Type	Description
DISR	0	rw	**Module Disable Request Bit** Used for enable/disable control of the module. 0_B Module disable not requested 1_B Module disable requested

(table continues...)

(continued)

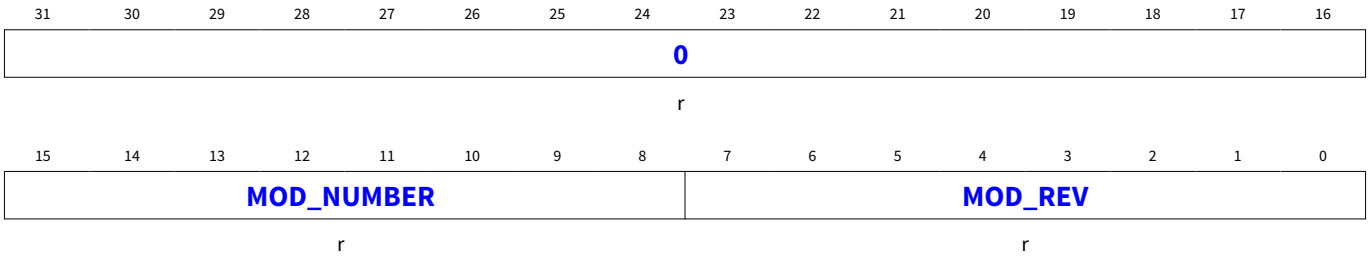
Field	Bits	Type	Description
DISS	1	rh	**Module Disable Status Bit** Bit indicates the current status of the module. 0_B Module enabled 1_B Module disabled
SPEN	2	rw	**Module Suspend Enable Bit for OCDS** 0_B Module suspend disabled 1_B Module suspend enabled
EDIS	3	rw	**External Request Disable** ***Note:*** *This bit is not used in TC4Dx and should always be written with 0.* 0_B External clock disable request is enabled 1_B External clock disable request is disabled
SBWE	4	w	**Module Suspend Bit Write Enable for OCDS** This bit is always read as 0. 0_B Bits SPEN and FSOE are write protected 1_B Bits SPEN and FSOE are overwritten by respective value of SPEN or FSOE
FSOE	5	rw	**Fast Switch Off Enable** 0_B FSOE Clock switch off in OCDS suspend mode via Disable Control Feature (Secure Clock Switch Off) 1_B Fast clock switch off in OCDS suspend mode
RMC	15:8	rwh	**Clock Divider for Standard Run Mode** Max. 8-bit divider value If RMC is set to 0 the module is disabled. ***Note:*** *As long as the new divider value RMC is not valid, reading register returns 0000 00XX$_H$*
RESERVED	23:16,	r	**Reserved** Read as 0; shall be written with 0.
0	7:6, 31:24	r	**Reserved** Read as 0; should be written with 0.

31.4.1.4 Module identification register

This register contains read-only information about the module and its revision.

ID1 Offset address: 00008_H

Module identification register Kernel Reset value: $0000\ 5705_H$



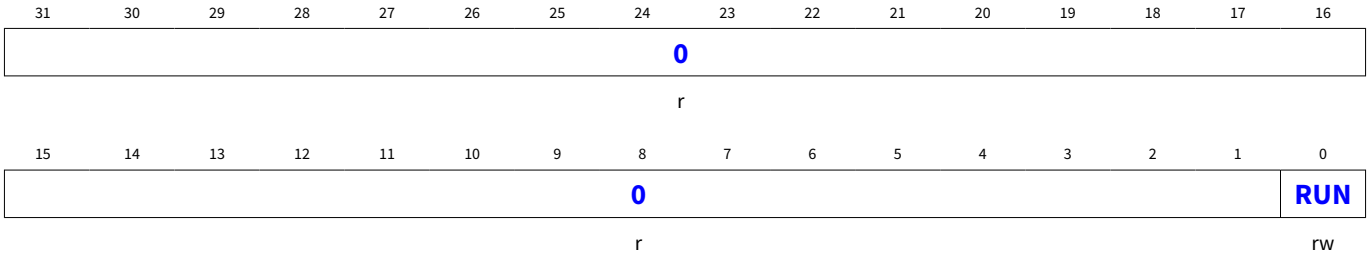
Field	Bits	Type	Description
MOD_REV	7:0	r	**Module Revision Number** This bit-field defines the revision number.
MOD_NUMBER	15:8	r	**Module Number** This bit-field defines the module identification number.
0	31:16	r	**Reserved** Read as 0.

31.4.1.5 RUN control register

This register selects configuration mode or run mode.

RUNCTRL Offset address: 00010_H

RUN control register Kernel Reset value: $0000\ 0000_H$



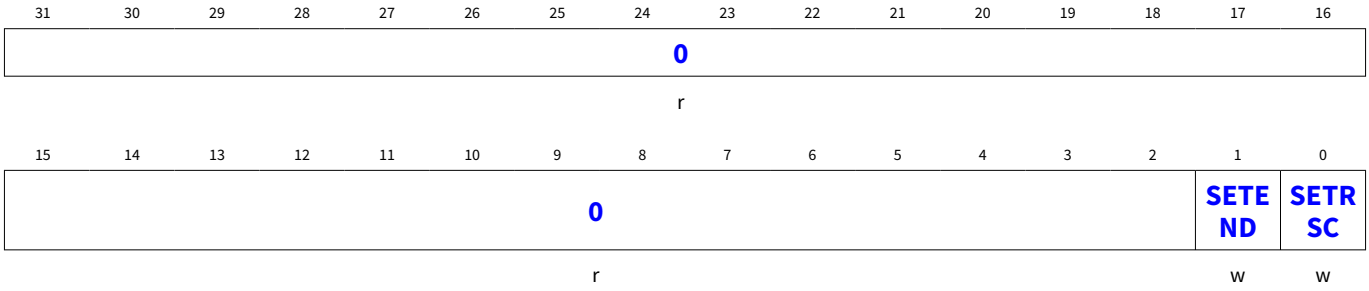
Field	Bits	Type	Description
RUN	0	rw	**Enable I2C-bus Interface** 0_B I2C-bus interface disabled; write access to configuration registers enabled 1_B Participation in I2C-bus communication enabled (if properly configured); write access to configuration registers disabled
0	31:1	r	**Reserved** Read as 0; should be written with 0.

31.4.1.6 End data control register

This register is used to either turn around the data transmission direction or address another slave without sending a stop condition. Also the software can stop the slave-transmitter by sending a not-acknowledge when working as master-receiver or even stop data transmission immediately when operating as master-transmitter. The writing to the bits of this control register is only effective in certain states.

ENDDCTRL Offset address: 00014_H

End data control register Kernel Reset value: $0000\ 0000_H$



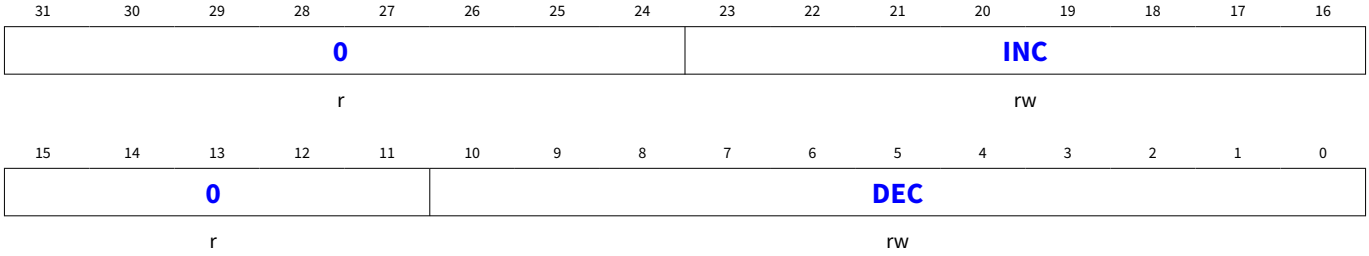
Field	Bits	Type	Description
SETRSC	0	w	**Set Restart Condition** This bit is always read as 0. 0_B Has no effect. 1_B The master wants to restart a data transmission (changing slave/ direction). The effect depends on the current state. MASTER RECEIVES BYTES: The master puts a not-acknowledge on the bus and switches to MASTER RESTART state. MASTER TRANSMITS BYTES: After the current byte has been sent, the master switches to MASTER RESTART state.
SETEND	1	w	**Set End of Transmission** This bit is always read as 0. ***Note**: Do not write 1 to this bit when bus is free. This will cause an abort after the first byte when a new transfer is started.* 0_B Has no effect. 1_B The effect depends on the current state. MASTER RECEIVES BYTES: After receiving the current byte, the master puts a not-acknowledge on the bus to indicate the transmission end to the slave-transmitter. Next it produces a stop condition on the bus and changes its state to LISTENING. MASTER TRANSMITS BYTES: After sending the current byte and receiving an acknowledge or not-acknowledge from the slave-receiver, the master puts a stop condition on the bus to close the data transmission and changes its state to LISTENING. MASTER RESTART: The master puts a stop condition on the bus to close the data transmission and changes its state to LISTENING. SLAVE RECEIVES BYTES: The slave-receiver puts a not-acknowledge on the bus after the received byte and changes its state to TRANSMISSION FINISHED.
0	31:2	r	**Reserved** Read as 0; should be written with 0.

31.4.1.7 Fractional divider configuration register

This configuration register is used to program the fractional divider of the I2C-bus for standard and fast mode. Before the peripheral is switched on by setting the RUN bit, the register should be configured.

FDIVCFG Offset address: 00018_H

Fractional divider configuration register Kernel Reset value: $0000\ 0000_H$



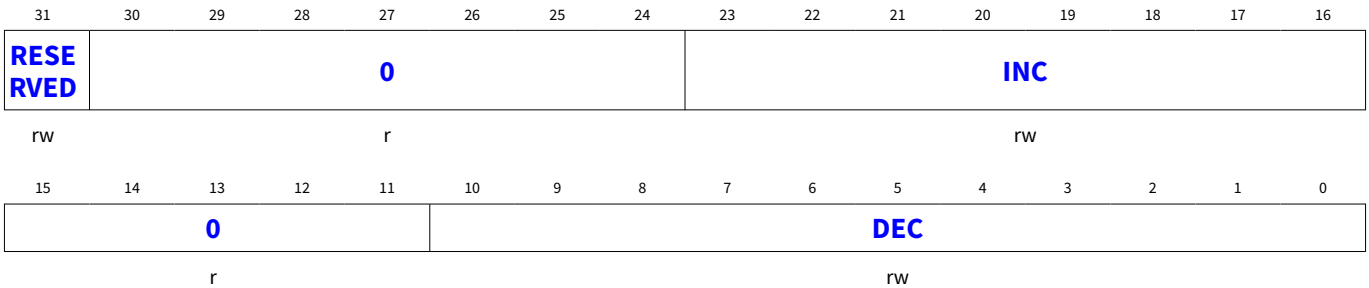
Field	Bits	Type	Description
DEC	10:0	rw	**Decrement Value of Fractional Divider** For standard/fast mode, see Clock and timing control section
INC	23:16	rw	**Increment Value of Fractional Divider** For standard/fast mode, see Clock and timing control section.
0	15:11, 31:24	r	**Reserved** Read as 0; should be written with 0.

31.4.1.8 Fractional divider high-speed mode configuration register

This configuration register is used to program the fractional divider of the I2C-bus for high-speed mode. Before the peripheral is switched on by setting the RUN bit, the register should be configured if high-speed mode is used.

FDIVHIGHCFG Offset address: $0001C_H$

Fractional divider high-speed mode configuration register Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
DEC	10:0	rw	**Decrement Value of Fractional Divider** For high-speed mode, see Clock and timing control section.
INC	23:16	rw	**Increment Value of Fractional Divider** For high-speed mode, see Clock and timing control section

(table continues...)

(continued)

Field	Bits	Type	Description
RESERVED	31	rw	**Reserved** Read as 0; shall be written with 0.
0	15:11, 30:24	r	**Reserved** Read as 0; should be written with 0.

31.4.1.9 Address configuration register

This configuration register contains the I2C-address (when addressed as a slave) and some bits that control the basic operation of the peripheral.

ADDRCFG Offset address: 00020_H

Address configuration register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0										SOPE	SONA	MnS	MCE	GCE	TBAM
r										rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0						ADR									
r						rw									

Field	Bits	Type	Description
ADR	9:0	rw	**I2C-bus Device Address** This bit-field determines the address of the device when addressed as a slave. (Watch out for reserved addresses by referring to I2C-bus spec V2.1.) Depending on setting of TBAM, this is either a 7-bit address (bits [7:1]) or a 10-bit address (bits [9:0]).
TBAM	16	rw	**Ten Bit Address Mode** ***Note**: When this bit is zero, only bits 7 down to 1 of the ADR field are valid.* 0_B 7-bit address mode enabled. 1_B 10-bit address mode enabled.
GCE	17	rw	**General Call Enable** 0_B Ignore general call occurrence. 1_B Enable general call detection; when detected, an acknowledge will be put on the bus
MCE	18	rw	**Master Code Enable** 0_B Device is not able to get along with high-speed mode 1_B Device is able to handle master code
MnS	19	rw	**Master / not Slave** 0_B Peripheral is configured as slave 1_B Peripheral is configured as master

(table continues...)

(continued)

Field	Bits	Type	Description
SONA	20	rw	**Stop on Not-acknowledge** ***Note**: After successful transmission of a master code (during high-speed mode) SONA is not considered till a stop condition is manually generated by SETEND.* 0_B Device changes to MASTER RESTART state. 1_B Device puts a stop condition on the bus and changes to LISTENING state.
SOPE	21	rw	**Stop on Packet End** ***Notes**:* ***1.** This bit-field should be used only in Master Mode. In slave mode should always be 0.* ***2.** If device works as receiver a not-acknowledge is always generated on package end.* ***3.** After successful transmission of a master code (during high-speed mode) SOPE is not considered till a stop condition is manually generated by SETEND.* 0_B Device enters MASTER RESTART state when the data packet end is indicated by the FIFO. 1_B Device puts a stop condition on the bus when the data packet end is indicated by the FIFO and changes to MASTER LISTENING state.
0	15:10, 31:22	r	**Reserved** Read as 0; should be written with 0.

31.4.1.10 Bus status register

This register contains status information of the I2C-bus. This additional information can be used by software to start appropriate actions.

BUSSTAT Offset address: 00024_H

Bus status register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0													RnW	BS	
r													rh	rh	

Field	Bits	Type	Description
BS	1:0	rh	**Bus Status** Shows the current status on the I2C-bus. 00_B I2C-bus is free (no start condition detected). 01_B A start condition has been detected on the bus (bus busy). 10_B The device is working as master and has claimed the control on the I2C-bus (busy master). 11_B A remote master has accessed this device as slave.
RnW	2	rh	**Read/not Write** Set by hardware automatically after address byte has been sent/ received. 0_B Working as transmitter (Write to I2C-bus). 1_B Working as receiver (Read from I2C-bus).
0	31:3	r	**Reserved** Read as 0.

31.4.1.11 FIFO configuration register

This configuration register is used to set up the FIFO before the peripheral is enabled and data is received or transmitted.

FIFOCFG Offset address: 00028_H

FIFO configuration register Kernel Reset value: $0000\ 0022_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0													CRBC	TXFC	RXFC
r													rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		TXFA		0		RXFA		0		TXBS		0		RXBS	
r		rw		r		rw		r		rw		r		rw	

Field	Bits	Type	Description
RXBS	1:0	rw	**RX Burst Size** 00_B 1 word 01_B 2 words 10_B 4 words 11_B Do not use this combination
TXBS	5:4	rw	**TX Burst Size** 00_B 1 word 01_B 2 words 10_B 4 words 11_B Do not use this combination

(table continues...)

(continued)

Field	Bits	Type	Description
RXFA	9:8	rw	**RX FIFO Alignment** 00_B Byte aligned (character alignment) 01_B Half word aligned (character alignment of two characters) 10_B Word aligned (character alignment of four characters) 11_B Do not use this combination
TXFA	13:12	rw	**TX FIFO Alignment** 00_B Byte aligned (character alignment) 01_B Half word aligned (character alignment of two characters) 10_B Word aligned (character alignment of four characters) 11_B Do not use this combination
RXFC	16	rw	**RX FIFO Flow Control** 0_B RX FIFO not as flow controller 1_B RX FIFO as flow controller
TXFC	17	rw	**TX FIFO Flow Control** 0_B TX FIFO not as flow controller 1_B TX FIFO as flow controller
CRBC	18	rw	**Clear Request Behavior Configuration** Used to configure the clear request behavior for the FIFO data request. Can only be used for single request and must be set to “0” when burst accesses are used in the system (eg. when TX/RXBS > 0) 0_B Data request is cleared by Software. 1_B Data request is cleared automatically when Write/Read access to FIFO occurs.
0	3:2, 7:6, 11:10, 15:14, 31:19	r	**Reserved** Read as 0; should be written with 0.

31.4.1.12 Maximum received packet size control register

This register is used to limit the received packet size. The register value may be changed in any state of the FIFO.

MRPSCTRL Offset address: $0002C_H$

Maximum received packet size control register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		MRPS													
r		rwh													

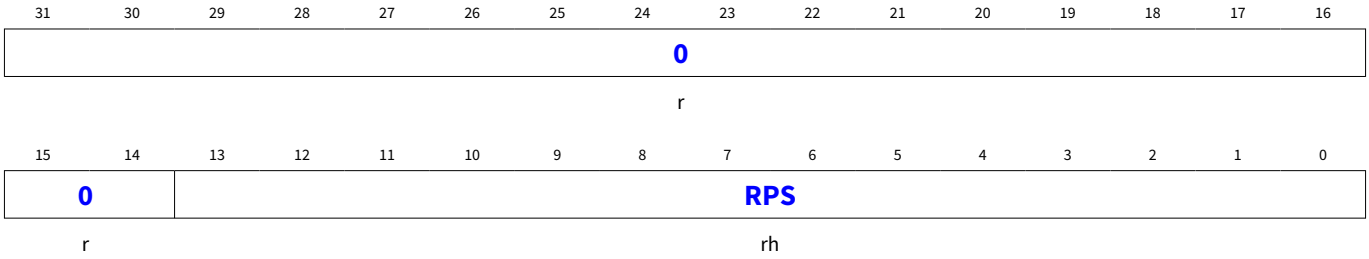
Field	Bits	Type	Description
MRPS	13:0	rwh	**Maximum Received Packet Size** Length in characters of packet to be received; write value range: 0 (unlimited size) to 16383 Reading returns the written value as long as the previous packet has not been read completely from the FIFO. After that, MRPS is loaded to an internal register, reading returns 0 and a new value can be written.
0	31:14	r	**Reserved** Read as 0; should be written with 0.

31.4.1.13 Received packet size status register

This register indicates the size of the received data packet to the software. The software should read this register after the last request of a packet.

RPSSTAT Offset address: 00030_H

Received packet size status register Kernel Reset value: $0000\ 0000_H$



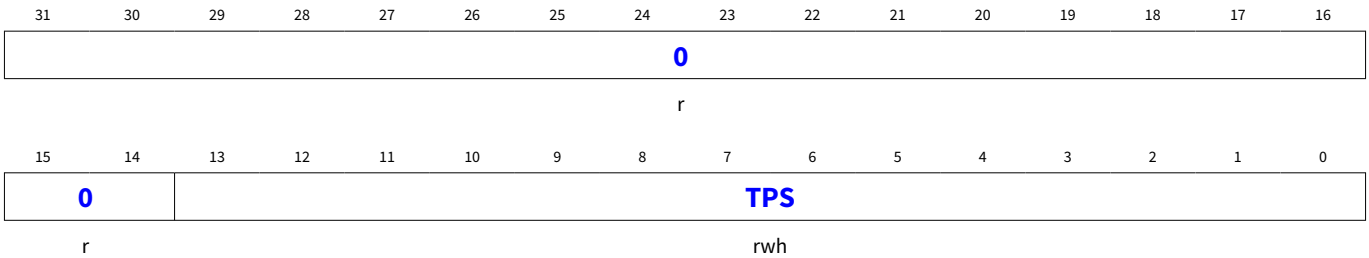
Field	Bits	Type	Description
RPS	13:0	rh	**Received Packet Size** Length in characters of the received packet (0 to 16383)
0	31:14	r	**Reserved** Read as 0.

31.4.1.14 Transmit packet size control register

This register is used to indicate the peripheral the size of the packet to be transmitted. Writing the packet size to this register if the FIFO controller is configured for flow controller mode initiates the data requests (BREQ, SREQ, ...).Writing to this register in configuration state has no impact.

TPSCTRL Offset address: 00034_H

Transmit packet size control register Kernel Reset value: $0000\ 0000_H$



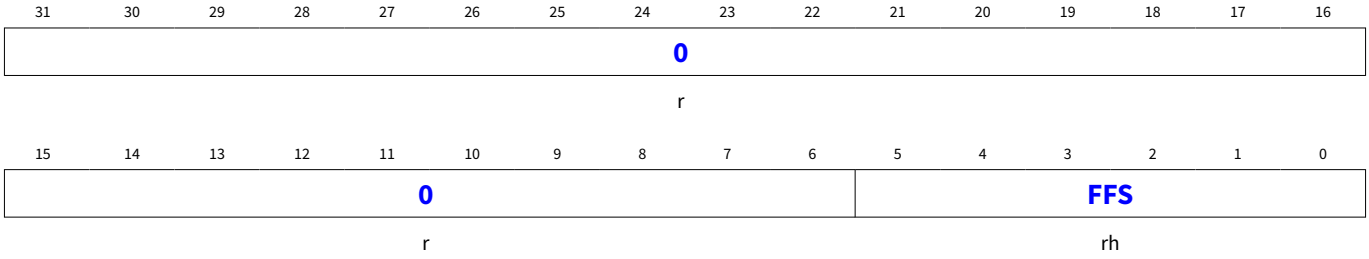
Field	Bits	Type	Description
TPS	13:0	rwh	**Transmit Packet Size** Length in characters of the transmit packet, write value range: 1 to 16383 Reading returns the written value as long as it is not loaded to an internal counter. After that, reading returns 0 and a new value can be written.
0	31:14	r	**Reserved** Read as 0; should be written with 0.

31.4.1.15 Filled FIFO stages status register

This register is used to indicate the number of filled FIFO stages.

FFSSTAT Offset address: 00038_H

Filled FIFO stages status register Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
FFS	5:0	rh	**Filled FIFO Stages** Number of filled FIFO stages (0 to 8)
0	31:6	r	**Reserved** Read as 0.

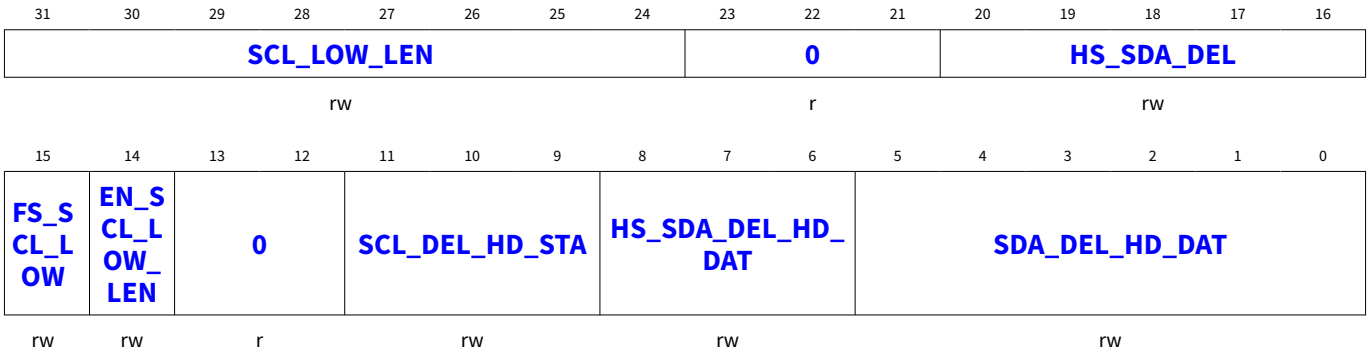
31.4.1.16 Timing configuration register

This configuration register adjusts some timings of the I2C-bus signals SCL and SCA. The delays are given in kernel_clk cycles (denoted as stages below).

The delayed stages may have +/- 1 stage deviation.

TIMCFG Offset address: 00040_H

Timing configuration register Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
SDA_DEL_HD_DAT	5:0	rw	**SDA Delay Stages for Data Hold Time in Standard and Fast modes** SDA delay stages for data hold time in standard and fast modes. ***Note:*** *SDA delay from SCL falling edge but will also affect SDA Setup time relative to next SCL rising edge* 00_H 3 stages delay … $3F_H$ 66 stages delay
HS_SDA_DEL_HD_DAT	8:6	rw	**SDA Delay Stages for Data Hold Time in High-speed Mode** SDA delay stages for data hold time in HS mode. ***Note:*** *SDA delay from SCL falling edge but will also affect SDA Setup time relative to next SCL rising edge* 000_B 3 stages delay … 111_B 10 stages delay
SCL_DEL_HD_STA	11:9	rw	**SCL Delay Stages for Hold Time Start (Restart) Bit** 000_B 2 stages delay … 111_B 9 stages delay
EN_SCL_LOW_LEN	14	rw	**Enable Direct Configuration of SCL Low Period Length in Fast Mode** 0_B SCL low period is a fixed part of the whole period, as defined by FS_SCL_LOW 1_B SCL low period is determined by the setting of SCL_LOW_LEN
FS_SCL_LOW	15	rw	**Set Fast Mode SCL Low Period Timing** The internal duration of the SCL low time with respect to the period length as defined by the baudrate setting, can be enlarged for the Fast Speed Mode, in order to meet the asymmetric duty cycle requirements from the standard. The detailed formulas are given in the functional specification. 0_B Standard mode SCL low period timing. For INC = 1 it is 5/8 of period. 1_B Fast mode SCL low period timing. For INC = 1 it is 6/8 of period.

(table continues...)

(continued)

Field	Bits	Type	Description
HS_SDA_DEL	20:16	rw	**SDA Delay Stages for Start/Stop bit in High-speed Mode** 00_H 3 stages delay … 07_H 10 stages delay
SCL_LOW_LEN	31:24	rw	**SCL Low Length in Fast Mode** If enabled by EN_SCL_LOW_LEN setting, this field determines the extension of the SCL low time. In case of INC = 1, the low time is extended by the number of kernel_clk cycles. In general, there is a more complex formula, as given in the functional specification. The total period time is not changed, i.e., the SCL high period is reduced accordingly. Setting SCL low time to period length or higher is not supported and would lead to unpredictable results.
0	13:12, 23:21	r	**Reserved** Read as 0; should be written with 0.

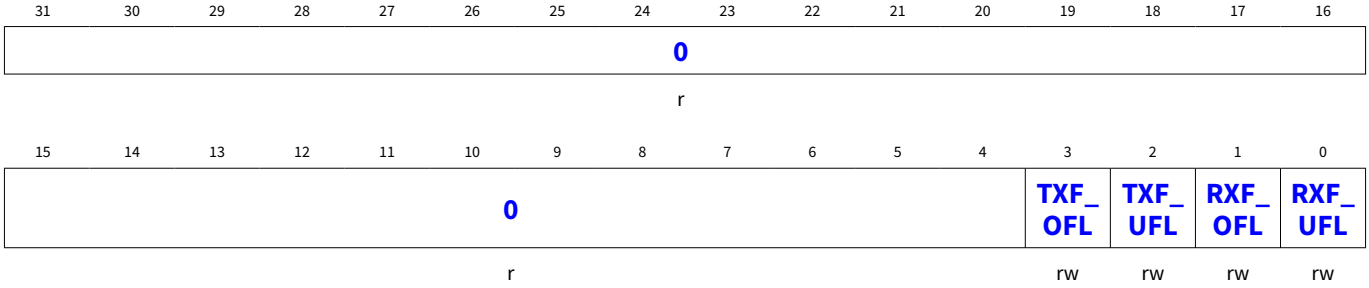
31.4.1.17 Error interrupt request source mask register

A write of 1 to a particular bit of this register enables the corresponding error interrupt request source; a write of 0 disables it. A read to this register returns the current mask bits. After reset all sources are enabled.

The interrupts are explained in detail in description of register ERRIRQSS.

ERRIRQSM Offset address: 00060_H

Error interrupt request source mask register Kernel Reset value: $0000\ 000F_H$



Field	Bits	Type	Description
RXF_UFL	0	rw	**RX FIFO Underflow** 0_B Interrupt request source disabled 1_B Interrupt request source enabled
RXF_OFL	1	rw	**RX FIFO Overflow** 0_B Interrupt request source disabled 1_B Interrupt request source enabled
TXF_UFL	2	rw	**TX FIFO Underflow** 0_B Interrupt request source disabled 1_B Interrupt request source enabled

(table continues...)

(continued)

Field	Bits	Type	Description
TXF_OFL	3	rw	**TX FIFO Overflow** 0_B Interrupt request source disabled 1_B Interrupt request source enabled
0	31:4	r	**Reserved** Read as 0; should be written with 0.

31.4.1.18 Error interrupt request source status register

This read-only register returns the current raw status value (without reflecting the mask) of the error interrupt request sources. A write to this register has no effect. The error status bits are set by hardware and can be cleared by software (via register ERRIRQSC).

ERRIRQSS Offset address: 00064_H

Error interrupt request source status register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												TXF_OFL	TXF_UFL	RXF_OFL	RXF_UFL
r												rh	rh	rh	rh

Field	Bits	Type	Description
RXF_UFL	0	rh	**RX FIFO Underflow** The FIFO has detected an RX FIFO underflow. 0_B No interrupt request 1_B Interrupt request pending
RXF_OFL	1	rh	**RX FIFO Overflow** The I2C kernel has detected a RX FIFO overflow 0_B No interrupt request 1_B Interrupt request pending
TXF_UFL	2	rh	**TX FIFO Underflow** The I2C kernel has detected a TX FIFO underflow. 0_B No interrupt request 1_B Interrupt request pending
TXF_OFL	3	rh	**TX FIFO Overflow** The FIFO has detected a TX FIFO overflow. 0_B No interrupt request 1_B Interrupt request pending
0	31:4	r	**Reserved** Read as 0.

31.4.1.19 Error interrupt request source clear register

On a write of 1 to a particular bit of this write-only register, the corresponding error interrupt request source is cleared and if no further error interrupt request sources are active, the whole error interrupt is cleared. If the corresponding bit is set by SW via the INTRSET register, then it can be cleared by setting any non-reserved bit of the interrupt request source clear register**.** A write of 0 has no effect. Reading the register returns 0.

The interrupts are explained in detail in description of register ERRIRQSS.

ERRIRQSC Offset address: 00068_H

Error interrupt request source clear register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												TXF_OFL	TXF_UFL	RXF_OFL	RXF_UFL
r												w	w	w	w

Field	Bits	Type	Description
RXF_UFL	0	w	**RX FIFO Underflow** 0_B No change 1_B Clear interrupt request source
RXF_OFL	1	w	**RX FIFO Overflow** 0_B No change 1_B Clear interrupt request source
TXF_UFL	2	w	**TX FIFO Underflow** 0_B No change 1_B Clear interrupt request source
TXF_OFL	3	w	**TX FIFO Overflow** 0_B No change 1_B Clear interrupt request source
0	31:4	r	**Reserved** Read as 0; should be written with 0.

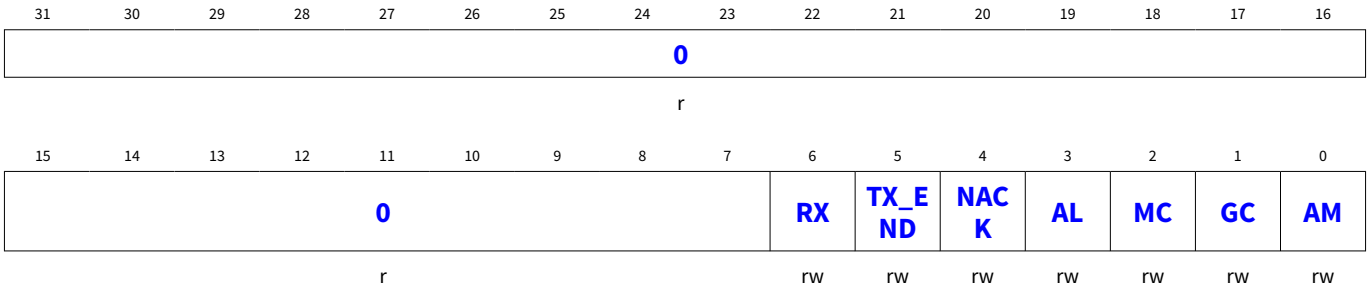
31.4.1.20 Protocol interrupt request source mask register

A write of 1 to a particular bit of this register enables the corresponding protocol interrupt request source; a write of 0 disables it. A read to this register returns the current mask bits. After reset all sources are enabled.

The interrupts are explained in detail in description of register PIRQSS.

PIRQSM Offset address: 00070_H

Protocol interrupt request source mask register Kernel Reset value: $0000\ 007F_H$



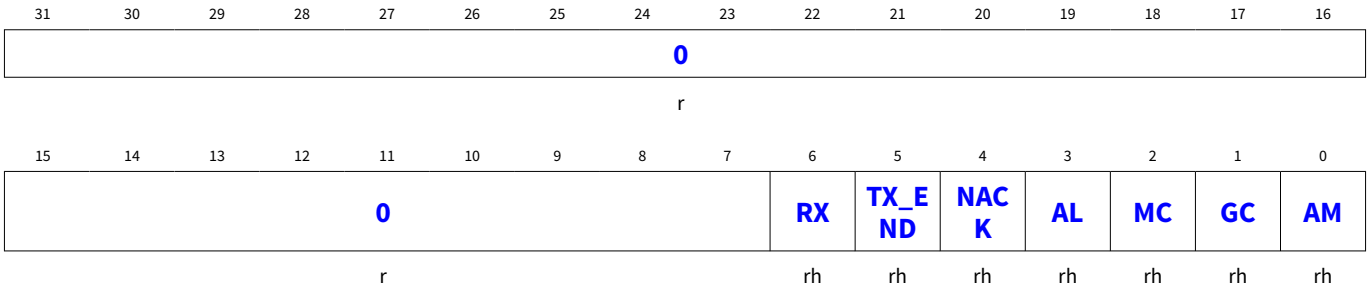
Field	Bits	Type	Description
AM	0	rw	**Address Match** 0_B Interrupt request source disabled 1_B Interrupt request source enabled
GC	1	rw	**General Call** 0_B Interrupt request source disabled 1_B Interrupt request source enabled
MC	2	rw	**Master Code** 0_B Interrupt request source disabled 1_B Interrupt request source enabled
AL	3	rw	**Arbitration Lost** 0_B Interrupt request source disabled 1_B Interrupt request source enabled
NACK	4	rw	**Not-acknowledge Received** 0_B Interrupt request source disabled 1_B Interrupt request source enabled
TX_END	5	rw	**Transmission End** 0_B Interrupt request source disabled 1_B Interrupt request source enabled
RX	6	rw	**Receive Mode** 0_B Interrupt request source disabled 1_B Interrupt request source enabled
0	31:7	r	**Reserved** Read as 0; should be written with 0.

31.4.1.21 Protocol interrupt request source status register

This read-only register returns the current raw status value (without reflecting the mask) of the protocol interrupt request sources. A write to this register has no effect. The protocol interrupt status bits are set by hardware and can be cleared by software (via register PIRQSC).

PIRQSS Offset address: 00074_H

Protocol interrupt request source status register Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
AM	0	rh	**Address Match** Device (in master or slave mode) is addressed by remote master (matching device address). Accordingly, bit-field BS in register BUSSTAT is set to 11. 0_B No interrupt request 1_B Interrupt request pending
GC	1	rh	**General Call** Remote master has transmitted a general call. 0_B No interrupt request 1_B Interrupt request pending
MC	2	rh	**Master Code** Remote master has transmitted a master call. 0_B No interrupt request 1_B Interrupt request pending
AL	3	rh	**Arbitration Lost** Device (master mode) lost the control on the I2C-bus due to losing arbitration procedure. Accordingly, bit-field BS in register BUSSTAT is set to 01. 0_B No interrupt request 1_B Interrupt request pending
NACK	4	rh	**Not-acknowledge Received** When working as transmitter this interrupt indicates a not-acknowledge from the remote receiver. The SW has to decide what further steps have to be taken. 0_B No interrupt request 1_B Interrupt request pending
TX_END	5	rh	**Transmission End** The device has ended the data transfer properly (after stop condition has been put on the bus or the MASTER RESTART state has been entered.) 0_B No interrupt request 1_B Interrupt request pending

(table continues...)

(continued)

Field	Bits	Type	Description
RX	6	rh	**Receive Mode** I2C kernel indicates switching from transmitting data to receiving data. 0_B No interrupt request 1_B Interrupt request pending
0	31:7	r	**Reserved** Read as 0.

31.4.1.22 Protocol interrupt request source clear register

On a write of 1 to a particular bit of this write-only register, the corresponding protocol interrupt request source is cleared and if no further protocol interrupt request sources are active, the whole protocol interrupt is cleared. If the corresponding RIS bit is set by SW via the INTRSET register, then it can be cleared by setting any non-reserved bit of the interrupt request source clear register**.** A write of 0 has no effect. Reading the register returns 0.

The interrupts are explained in detail in description of register PIRQSS.

PIRQSC Offset address: 00078_H

Protocol interrupt request source clear register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0									RX	TX_END	NACK	AL	MC	GC	AM
r									w	w	w	w	w	w	w

Field	Bits	Type	Description
AM	0	w	**Address Match** 0_B No change 1_B Clear Interrupt source
GC	1	w	**General Call** 0_B No change 1_B Clear Interrupt source
MC	2	w	**Master Code** 0_B No change 1_B Clear Interrupt source
AL	3	w	**Arbitration Lost** 0_B No change 1_B Clear Interrupt source

(table continues...)

(continued)

Field	Bits	Type	Description
NACK	4	w	**Not-acknowledge Received** 0_B No change 1_B Clear Interrupt source
TX_END	5	w	**Transmission End** 0_B No change 1_B Clear Interrupt source
RX	6	w	**Receive Mode** 0_B No change 1_B Clear Interrupt source
0	31:7	r	**Reserved** Read as 0; should be written with 0.

31.4.1.23 Raw interrupt status register

This read-only register returns the current raw status value (without reflecting the mask) of the interrupt request sources. One status bit is provided for each request. A write to this register has no effect. The status bits are set by hardware or software (via register INTRSET) and can be cleared by software (via register ICR).

RIS Offset address: 00080_H

Raw interrupt status register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0										I2C_P_INT	I2C_ERR_INT	BREQ_INT	LBREQ_INT	SREQ_INT	LSREQ_INT
r										rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
LSREQ_INT	0	rh	**Last Single Request Interrupt** 0_B No interrupt request 1_B Interrupt request pending
SREQ_INT	1	rh	**Single Request Interrupt** 0_B No interrupt request 1_B Interrupt request pending
LBREQ_INT	2	rh	**Last Burst Request Interrupt** 0_B No interrupt request 1_B Interrupt request pending

(table continues...)

(continued)

Field	Bits	Type	Description
BREQ_INT	3	rh	**Burst Request Interrupt** 0_B No interrupt request 1_B Interrupt request pending
I2C_ERR_INT	4	rh	**I2C Error Interrupt** This is the combined bit for indication of FIFO errors due to overflow and underflow. 0_B No interrupt request 1_B Interrupt request pending
I2C_P_INT	5	rh	**I2C Protocol Interrupt** This is the combined bit for indication of a protocol event in the I2C kernel. 0_B No interrupt request 1_B Interrupt request pending
0	31:6	r	**Reserved** Read as 0.

31.4.1.24 Interrupt mask control register

A write of 1 to a particular bit of this register enables the corresponding interrupt request; a write of 0 disables it. A read to this register returns the current mask bits. After reset all requests are disabled.

IMSC Offset address: 00084_H

Interrupt mask control register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0										I2C_P_INT	I2C_ERR_INT	BREQ_INT	LBREQ_INT	SREQ_INT	LSREQ_INT
r										rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
LSREQ_INT	0	rw	**Last Single Request Interrupt** 0_B Interrupt request disabled 1_B Interrupt request enabled
SREQ_INT	1	rw	**Single Request Interrupt** 0_B Interrupt request disabled 1_B Interrupt request enabled
LBREQ_INT	2	rw	**Last Burst Request Interrupt** 0_B Interrupt request disabled 1_B Interrupt request enabled

(table continues...)

(continued)

Field	Bits	Type	Description
BREQ_INT	3	rw	**Burst Request Interrupt** 0_B Interrupt request disabled 1_B Interrupt request enabled
I2C_ERR_INT	4	rw	**I2C Error Interrupt** 0_B Interrupt request disabled 1_B Interrupt request enabled
I2C_P_INT	5	rw	**I2C Protocol Interrupt** 0_B Interrupt request disabled 1_B Interrupt request enabled
0	31:6	r	**Reserved** Read as 0; should be written with 0.

31.4.1.25 Masked interrupt status register

This read-only register returns the masked status value (derived from registers RIS and IMSC) of the corresponding interrupt requests. A write to this register has no effect.

MIS Offset address: 00088_H

Masked interrupt status register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0										I2C_P_INT	I2C_ERR_INT	BREQ_INT	LBREQ_INT	SREQ_INT	LSREQ_INT
r										rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
LSREQ_INT	0	rh	**Last Single Request Interrupt** 0_B No interrupt request 1_B Interrupt request pending
SREQ_INT	1	rh	**Single Request Interrupt** 0_B No interrupt request 1_B Interrupt request pending
LBREQ_INT	2	rh	**Last Burst Request Interrupt** 0_B No interrupt request 1_B Interrupt request pending
BREQ_INT	3	rh	**Burst Request Interrupt** 0_B No interrupt request 1_B Interrupt request pending

(table continues...)

(continued)

Field	Bits	Type	Description
I2C_ERR_INT	4	rh	**I2C Error Interrupt** This is the combined bit for indication of FIFO errors due to overflow and underflow. 0_B No interrupt request 1_B Interrupt request pending
I2C_P_INT	5	rh	**I2C Protocol Interrupt** This is the combined bit for indication of a protocol event in the I2C kernel. 0_B No interrupt request 1_B Interrupt request pending
0	31:6	r	**Reserved** Read as 0.

31.4.1.26 Interrupt clear register

On a write of 1 to a particular bit of this write-only register, the corresponding interrupt request is cleared; a write of 0 has no effect. Reading the register returns 0.

ICR Offset address: $0008C_H$

Interrupt clear register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												BREQ_INT	LBREQ_INT	SREQ_INT	LSREQ_INT
r												w	w	w	w

Field	Bits	Type	Description
LSREQ_INT	0	w	**Last Single Request Interrupt** 0_B No change 1_B Clear interrupt request
SREQ_INT	1	w	**Single Request Interrupt** 0_B No change 1_B Clear interrupt request
LBREQ_INT	2	w	**Last Burst Request Interrupt** 0_B No change 1_B Clear interrupt request
BREQ_INT	3	w	**Burst Request Interrupt** 0_B No change 1_B Clear interrupt request

(table continues...)

(continued)

Field	Bits	Type	Description
0	31:4	r	**Reserved** Read as 0; should be written with 0.

31.4.1.27 Interrupt set register

On a write of 1 to a particular bit of this write-only register, the corresponding interrupt request is set; a write of 0 has no effect. Reading the register returns 0.

INTRSET Offset address: 00090_H

Interrupt set register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0										I2C_P_INT	I2C_ERR_INT	BREQ_INT	LBREQ_INT	SREQ_INT	LSREQ_INT
r										w	w	w	w	w	w

Field	Bits	Type	Description
LSREQ_INT	0	w	**Last Single Request Interrupt** 0_B No change 1_B Set interrupt request
SREQ_INT	1	w	**Single Request Interrupt** 0_B No change 1_B Set interrupt request
LBREQ_INT	2	w	**Last Burst Request Interrupt** 0_B No change 1_B Set interrupt request
BREQ_INT	3	w	**Burst Request Interrupt** 0_B No change 1_B Set interrupt request
I2C_ERR_INT	4	w	**I2C Error Interrupt** 0_B No change 1_B Set interrupt request
I2C_P_INT	5	w	**I2C Protocol Interrupt** 0_B No change 1_B Set interrupt request
0	31:6	r	**Reserved** Read as 0; should be written with 0.

31.4.1.28 Transmission data register

The software has to write the characters to be transmitted into this register.

An address range of (8000_H to 8003_H) is reserved for the FIFO.

A read access to TXD register is not possible, it will return 0 in all cases. Reading has no effect on the FIFO.

When using byte or half word access from the bus, the TX FIFO pointer will only be increased, if one of the following conditions is fulfilled:

- The most significant byte or half word of the FIFO stage is written
- The packet end is reached

TXD Offset address: 08000_H

Transmission data register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TXD															
w															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXD															
w															

Field	Bits	Type	Description
TXD	31:0	w	**Transmission Data** Characters to be transmitted

31.4.1.29 Reception data register

The software can read the received characters from this register.

An address range of ($C000_H$ to $C003_H$) is reserved for the FIFO .

RXD Offset address: $0C000_H$

Reception data register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RXD															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXD															
rh															

Field	Bits	Type	Description
RXD	31:0	rh	**Reception Data** Received characters

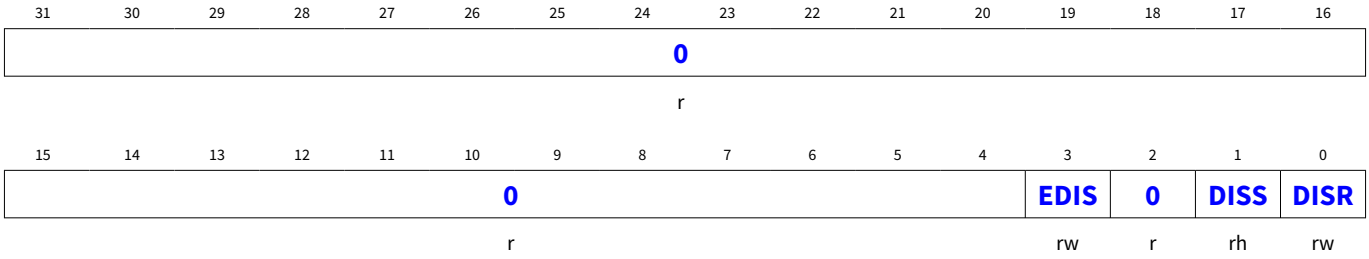
31.4.1.30 Clock Control Register

The Clock Control Register CLC allows the programmer to adapt the functionality and power consumption of the module to the requirements of the application.

Register CLC controls the module clock signal and the reactivity to the sleep signal.

CLC Offset address: 10000_H

Clock Control Register Application Reset value: $0000\ 0003_H$

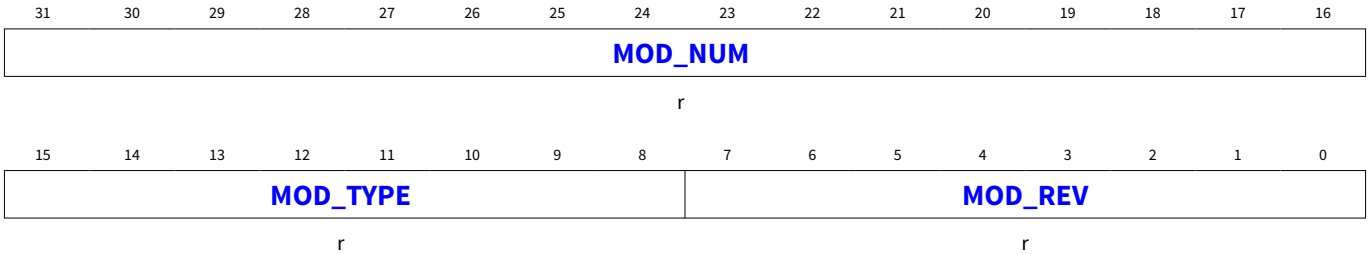


Field	Bits	Type	Description
DISR	0	rw	**Module Disable Request Bit** Used for enable/disable control of the module. 0_B On request: enable the module clock 1_B Off request: stop the module clock
DISS	1	rh	**Module Disable Status Bit** 0_B Module clock is enabled 1_B Off: module is not clocked
EDIS	3	rw	**Sleep Mode Enable Control** Used to control the module's reaction to sleep mode. 0_B Sleep mode request is enabled and functional 1_B Module disregards the sleep mode control signal
0	2, 31:4	r	**Reserved** Read as 0; should be written with 0.

31.4.1.31 Module Identification Register

ID Offset address: 10008_H

Module Identification Register Application Reset value: $00C2\ C000_H$



Field	Bits	Type	Description
MOD_REV	7:0	r	**Module Revision** Indicates the revision number of the implementation.
MOD_TYPE	15:8	r	**Module Type** This internal marker is fixed to $C0_H$.
MOD_NUM	31:16	r	**Module Number** Indicates the module identification number.

31.4.1.32 Reset Control Register A

RST_CTRLA Offset address: $1000C_H$

Reset Control Register A Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

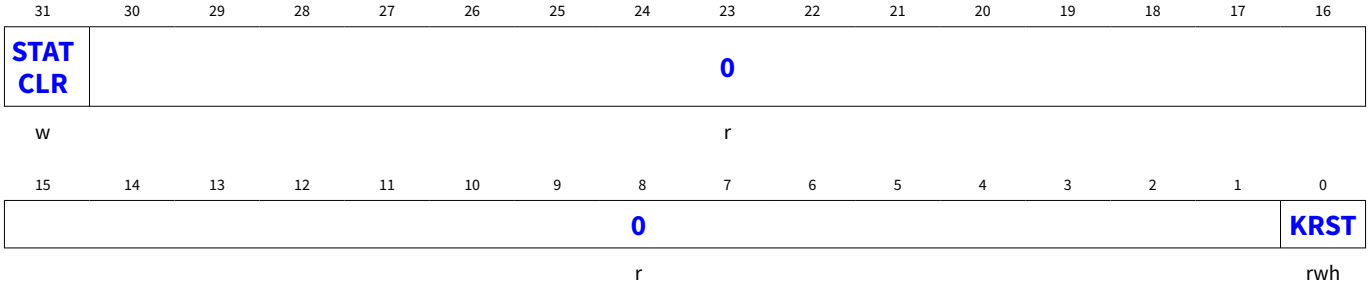
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				GRSTEN3	GRSTEN2	GRSTEN1	GRSTEN0	0							KRST
r				rw	rw	rw	rw	r							rwh

Field	Bits	Type	Description
KRST	0	rwh	**Kernel Reset** Request a kernel reset. The requested reset is executed if CTRLA.KRST=1 and CTRLB.KRST=1. KRST is cleared after the kernel reset was executed. 0_B No action 1_B A kernel reset was requested
GRSTENx (x=0-3)	x+8	rw	**Enable for Global Module Reset Group x** 0_B Global module reset group x does not have any effect 1_B Global module reset group x results in a kernel reset
0	7:1, 31:12	r	**Reserved** Read as 0; should be written with 0.

31.4.1.33 Reset Control Register B

RST_CTRLB Offset address: 10010_H

Reset Control Register B Application Reset value: $0000\ 0000_H$



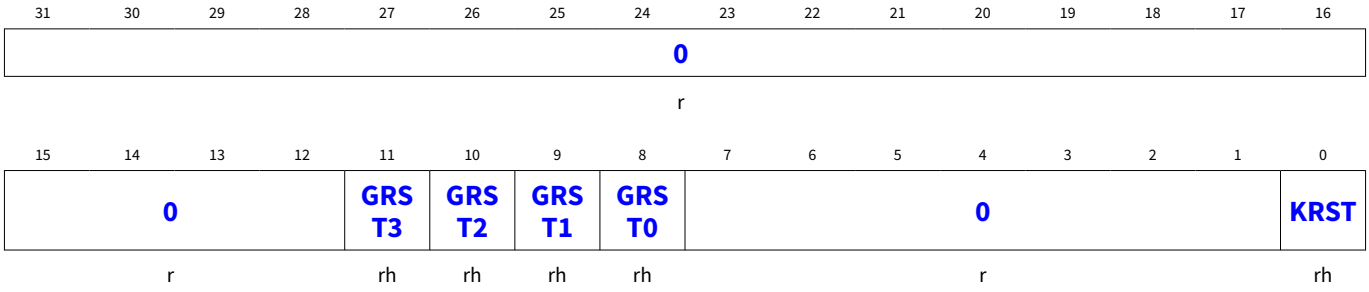
Field	Bits	Type	Description
KRST	0	rwh	**Kernel Reset** Request a kernel reset. The requested reset is executed if CTRLA.KRST=1 and CTRLB.KRST=1. KRST is cleared after the kernel reset was executed. 0_B No action 1_B A kernel reset was requested
STATCLR	31	w	**Kernel Reset Status Clear** Clears all status bits in RST_STAT when 1 is written. Read will return 0. 0_B No action 1_B Write with ´1´ clears bits STAT.GRSTx and bit STAT.KRST .
0	30:1	r	**Reserved** Read as 0; should be written with 0.

31.4.1.34 Reset Status Register

The reset status register contains the status bits for kernel reset (KRST) and the global module reset groups (GRSTx).

RST_STAT Offset address: 10014_H

Reset Status Register Application Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
KRST	0	rh	**Kernel reset status** Indicates an executed kernel reset. RST_STAT.KRST is set after the execution of a kernel reset in the same clock cycle in which the reset bits are cleared. Clear KRST by setting bit STATCLR in register RST_CTRLB.

(table continues...)

(continued)

Field	Bits	Type	Description
GRSTx (x=0-3)	x+8	rh	**Status for global module reset group x** 0_B Reset was not triggered by global reset group x 1_B Reset was triggered by global reset group x
0	7:1, 31:12	r	**Reserved** Read as 0.

31.4.1.35 PROT Register Endinit

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner or CS master to update the PROT state as described in the PROT mechanism.

The PROTE register controls lock / unlock of the local Endinit (E) protected control registers.

PROTE Offset address: 10020_H

PROT Register Endinit Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OWEN	ODEF	TAGID						PRSEN	PRS			VMEN	VM		
w	rw	rw						rw	rw			rw	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												SWEN	STATE		
r												w	rwh		

Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
VM	18:16	rw	**Virtual machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.

(table continues...)

(continued)

Field	Bits	Type	Description
VMEN	19	rw	**Virtual machine definition enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.
PRSEN	23	rw	**Protection set definition enable for PROT owner** 0_B **Disable**: PRS is not part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG-ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; should be written with 0.

Table 1265 Access mode restrictions of PROTE sorted by descending priority

Mode name	Access mode		Description
write 1 to .**SWEN**	rwh	STATE	
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

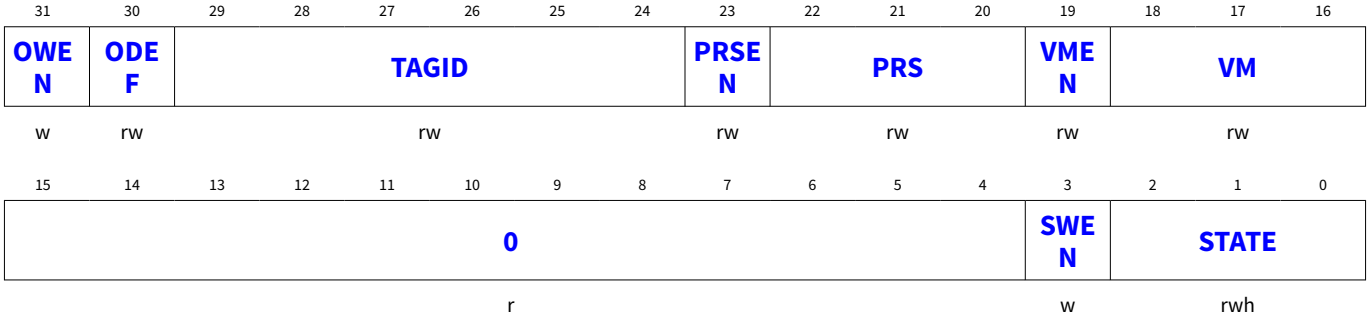
31.4.1.36 PROT Register Safe Endinit

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner or CS master to update the PROT state as described in the PROT mechanism.

The PROTSE register controls lock / unlock of the local Safe Endinit (SE) protected control registers.

PROTSE Offset address: 10024_H

PROT Register Safe Endinit Application Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
VM	18:16	rw	**Virtual machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual machine definition enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.
PRSEN	23	rw	**Protection set definition enable for PROT owner** 0_B **Disable**: PRS is not part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG-ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner

(table continues...)

(continued)

Field	Bits	Type	Description
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; should be written with 0.

Table 1266 Access mode restrictions of PROTSE sorted by descending priority

Mode name	Access mode		Description
write 1 to .**SWEN**	rwh	STATE	
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

31.4.1.37 Write access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

ACCEN_WRA Offset address: 10040_H

Write access enable register A Application Reset value: $1000\ 0003_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access

31.4.1.38 Write access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

ACCEN_WRB Offset address: 10044_H

Write access enable register B Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access
0	31:8	r	**Reserved** Read as 0; should be written with 0.

31.4.1.39 Read access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

ACCEN_RDA Offset address: 10048_H

Read access enable register A Application Reset value: $FFFF\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access

31.4.1.40 Read access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

ACCEN_RDB Offset address: $1004C_H$

Read access enable register B Application Reset value: $0000\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access
0	31:8	r	**Reserved** Read as 0; should be written with 0.

31.4.1.41 VM access enable register

This register defines which virtual machine encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each VMx.

ACCEN_VM Offset address: 10050_H

VM access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for VM ID q** This bit enables read access to the access protected region for transactions with the VM ID q. 0_B Disabled for read access 1_B Enabled for read access

(table continues...)

(continued)

Field	Bits	Type	Description
WRq (q=00-07)	q+16	rw	**Write access enable for VM ID q** This bit enables write access to the access protected region for transactions with the VM ID q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

31.4.1.42 PRS access enable register

This register defines which protection register set encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each PRSx.

ACCEN_PRS Offset address: 10054_H

PRS access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for PRS q** This bit enables read access to the access protected region for transactions with the PRS q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for PRS q** This bit enables write access to the access protected region for transactions with the PRS q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

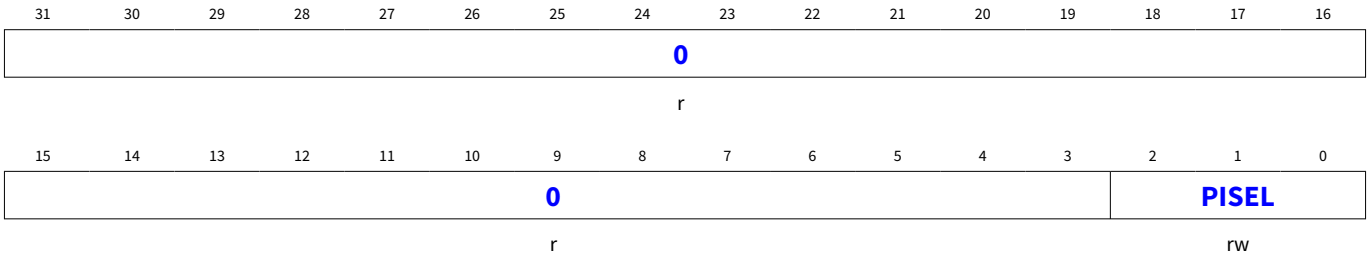
31.4.1.43 General purpose control register

This register implements the PISEL functionality. The SDA and SCL input multiplexors are controlled in parallel with each PISEL setting. Since 3 bits are implemented for PISEL, up to 8 sets of SDA and SCL pins can be defined.

The register is disabled if the module is disabled via CLC.

GPCTL Offset address: 10060_H

General purpose control register Application Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
PISEL	2:0	rw	**Port Input Select** Used to select the input pins providing the serial data and clock input signals, in the range of 0 to 7. ***Note**: In TC4Dx, not all PISEL options are available. See Data Sheet.* 000_B SDAxA and SCLxA are selected. 001_B SDAxB and SCLxB are selected. 010_B SDAxC and SCLxC are selected. 011_B SDAxD and SCLxD are selected. 100_B SDAxE and SCLxE are selected. 101_B SDAxF and SCLxF are selected. 110_B SDAxG and SCLxG are selected. 111_B SDAxH and SCLxH are selected.
0	31:3	r	**Reserved** Reads 0. Should be written with 0.

31.5 Debug information

This section is not applicable for the module being described.

31.6 References

For more information, see the following documentation:

1. I2C-bus specification version 2.1 standard (released January-2000)
2. I2C Inter IC Bus Interface Revision 1.11

31.7 I2C revision history

Initial release of the chapter.

31.8 TC4Dx I2C information

31.8.1 TC4Dx I2C configuration

Table 1267 Device specific parameters

Parameter	Value
Number of instances	3

31.8.2 TC4Dx I2C features

There are no deviations from the generic specification.

31.8.3 TC4Dx I2C functional description

In TC4Dx device, the I2C module is connected to ComPB bus interface.

31.8.4 TC4Dx I2C registers

There are no deviations from the generic specification.

31.8.4.1 Register address space - I2C

Table 1268 Registers address space - I2C

Module	Base address	End address	Note
I2C0	$F44C0000_H$	$F44DFFFF_H$	FPI slave interface
I2C1	$F44E0000_H$	$F44FFFFF_H$	FPI slave interface
I2C2	$F4500000_H$	$F451FFFF_H$	FPI slave interface

31.8.4.2 Register overview - access mode glossary

Table 1269 Register overview - access mode glossary

Keyword	Description
E	Access protection using PROT register I2C0_PROTE or I2C1_PROTE or I2C2_PROTE .
SE	Access protection using PROT register I2C0_PROTSE or I2C1_PROTSE or I2C2_PROTSE .
APU-P	Protection group consisting of registers I2C0_ACCEN_WRA , I2C0_ACCEN_WRB , I2C0_ACCEN_RDA , I2C0_ACCEN_RDB , I2C0_ACCEN_VM , I2C0_ACCEN_PRS or I2C1_ACCEN_WRA , I2C1_ACCEN_WRB , I2C1_ACCEN_RDA , I2C1_ACCEN_RDB , I2C1_ACCEN_VM , I2C1_ACCEN_PRS or I2C2_ACCEN_WRA , I2C2_ACCEN_WRB , I2C2_ACCEN_RDA , I2C2_ACCEN_RDB , I2C2_ACCEN_VM , I2C2_ACCEN_PRS .
P	Access protection using *APU-P* registers.
U	No access restrictions.
BE	Always returns a Bus Error.
nBE	Indicates that no Bus Error is generated when accessing this address range, even though it is either an access to an undefined address or the access does not follow the given rules.
PROT	Access restrictions as defined in the PROT register access rules.

31.8.4.3 Register overview - I2C0 (ascending offset address)

Table 1270 Register overview - I2C0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
I2C0_CLC1	Clock control 1 register	00000_H	U	P	Kernel Reset	5477
I2C0_ID1	Module identification register	00008_H	U	BE	Kernel Reset	5478
I2C0_RUNCTRL	RUN control register	00010_H	U	P	Kernel Reset	5479
I2C0_ENDDCTRL	End data control register	00014_H	U	P	Kernel Reset	5480
I2C0_FDIVCFG	Fractional divider configuration register	00018_H	U	P	Kernel Reset	5481

(table continues...)

Table 1270 (continued) Register overview - I2C0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
I2C0_FDIVHIGHCFG	Fractional divider high-speed mode configuration register	$0001C_H$	U	P	Kernel Reset	5481
I2C0_ADDRCFG	Address configuration register	00020_H	U	P	Kernel Reset	5482
I2C0_BUSTAT	Bus status register	00024_H	U	BE	Kernel Reset	5483
I2C0_FIFOCFG	FIFO configuration register	00028_H	U	P	Kernel Reset	5484
I2C0_MRPSCTRL	Maximum received packet size control register	$0002C_H$	U	P	Kernel Reset	5485
I2C0_RPSSTAT	Received packet size status register	00030_H	U	BE	Kernel Reset	5486
I2C0_TPSCTRL	Transmit packet size control register	00034_H	U	P	Kernel Reset	5486
I2C0_FFSSTAT	Filled FIFO stages status register	00038_H	U	BE	Kernel Reset	5487
I2C0_TIMCFG	Timing configuration register	00040_H	U	P	Kernel Reset	5487
I2C0_ERRIRQSM	Error interrupt request source mask register	00060_H	U	P	Kernel Reset	5489
I2C0_ERRIRQSS	Error interrupt request source status register	00064_H	U	BE	Kernel Reset	5490
I2C0_ERRIRQSC	Error interrupt request source clear register	00068_H	U	P	Kernel Reset	5491
I2C0_PIRQSM	Protocol interrupt request source mask register	00070_H	U	P	Kernel Reset	5491
I2C0_PIRQSS	Protocol interrupt request source status register	00074_H	U	BE	Kernel Reset	5492
I2C0_PIRQSC	Protocol interrupt request source clear register	00078_H	U	P	Kernel Reset	5494
I2C0_RIS	Raw interrupt status register	00080_H	U	BE	Kernel Reset	5495
I2C0_IMSC	Interrupt mask control register	00084_H	U	P	Kernel Reset	5496
I2C0_MIS	Masked interrupt status register	00088_H	U	BE	Kernel Reset	5497
I2C0_ICR	Interrupt clear register	$0008C_H$	U	P	Kernel Reset	5498
I2C0_INTRSET	Interrupt set register	00090_H	U	P	Kernel Reset	5499
I2C0_TXD	Transmission data register	08000_H	nBE	P	Kernel Reset	5500
I2C0_RXD	Reception data register	$0C000_H$	U	BE	Kernel Reset	5500
I2C0_CLC	Clock Control Register	10000_H	P	P, SV, E	Application Reset	5501

(table continues...)

Table 1270 (continued) Register overview - I2C0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
I2C0_ID	Module Identification Register	10008_H	P	BE	Application Reset	5501
I2C0_RST_CTRLA	Reset Control Register A	$1000C_H$	P	P, SV, E	Application Reset	5502
I2C0_RST_CTRLB	Reset Control Register B	10010_H	P	P, SV, E	Application Reset	5502
I2C0_RST_STAT	Reset Status Register	10014_H	P	BE	Application Reset	5503
I2C0_PROTE	PROT Register Endinit	10020_H	U	SV, PROT	Application Reset	5504
I2C0_PROTSE	PROT Register Safe Endinit	10024_H	U	SV, PROT	Application Reset	5505
I2C0_ACCEN_WRA	Write access enable register A	10040_H	U	SE, SV	Application Reset	5507
I2C0_ACCEN_WRB	Write access enable register B	10044_H	U	SE, SV	Application Reset	5508
I2C0_ACCEN_RDA	Read access enable register A	10048_H	U	SE, SV	Application Reset	5508
I2C0_ACCEN_RDB	Read access enable register B	$1004C_H$	U	SE, SV	Application Reset	5509
I2C0_ACCEN_VM	VM access enable register	10050_H	U	SE, SV	Application Reset	5509
I2C0_ACCEN_PRS	PRS access enable register	10054_H	U	SE, SV	Application Reset	5510
I2C0_GPCTL	General purpose control register	10060_H	SV	SV, P	Application Reset	5511

31.8.4.4 Register overview - I2C1 (ascending offset address)

Table 1271 Register overview - I2C1 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
I2C1_CLC1	Clock control 1 register	00000_H	U	P	Kernel Reset	5477
I2C1_ID1	Module identification register	00008_H	U	BE	Kernel Reset	5478
I2C1_RUNCTRL	RUN control register	00010_H	U	P	Kernel Reset	5479
I2C1_ENDDCTRL	End data control register	00014_H	U	P	Kernel Reset	5480
I2C1_FDIVCFG	Fractional divider configuration register	00018_H	U	P	Kernel Reset	5481

(table continues...)

Table 1271 (continued) Register overview - I2C1 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
I2C1_FDIVHIGHCFG	Fractional divider high-speed mode configuration register	$0001C_H$	U	P	Kernel Reset	5481
I2C1_ADDRCFG	Address configuration register	00020_H	U	P	Kernel Reset	5482
I2C1_BUSSTAT	Bus status register	00024_H	U	BE	Kernel Reset	5483
I2C1_FIFOCFG	FIFO configuration register	00028_H	U	P	Kernel Reset	5484
I2C1_MRPSCTRL	Maximum received packet size control register	$0002C_H$	U	P	Kernel Reset	5485
I2C1_RPSSTAT	Received packet size status register	00030_H	U	BE	Kernel Reset	5486
I2C1_TPSCTRL	Transmit packet size control register	00034_H	U	P	Kernel Reset	5486
I2C1_FFSSTAT	Filled FIFO stages status register	00038_H	U	BE	Kernel Reset	5487
I2C1_TIMCFG	Timing configuration register	00040_H	U	P	Kernel Reset	5487
I2C1_ERRIRQSM	Error interrupt request source mask register	00060_H	U	P	Kernel Reset	5489
I2C1_ERRIRQSS	Error interrupt request source status register	00064_H	U	BE	Kernel Reset	5490
I2C1_ERRIRQSC	Error interrupt request source clear register	00068_H	U	P	Kernel Reset	5491
I2C1_PIRQSM	Protocol interrupt request source mask register	00070_H	U	P	Kernel Reset	5491
I2C1_PIRQSS	Protocol interrupt request source status register	00074_H	U	BE	Kernel Reset	5492
I2C1_PIRQSC	Protocol interrupt request source clear register	00078_H	U	P	Kernel Reset	5494
I2C1_RIS	Raw interrupt status register	00080_H	U	BE	Kernel Reset	5495
I2C1_IMSC	Interrupt mask control register	00084_H	U	P	Kernel Reset	5496
I2C1_MIS	Masked interrupt status register	00088_H	U	BE	Kernel Reset	5497
I2C1_ICR	Interrupt clear register	$0008C_H$	U	P	Kernel Reset	5498
I2C1_INTRSET	Interrupt set register	00090_H	U	P	Kernel Reset	5499
I2C1_TXD	Transmission data register	08000_H	nBE	P	Kernel Reset	5500
I2C1_RXD	Reception data register	$0C000_H$	U	BE	Kernel Reset	5500
I2C1_CLC	Clock Control Register	10000_H	P	P, SV, E	Application Reset	5501

(table continues...)

Table 1271 (continued) Register overview - I2C1 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
I2C1_ID	Module Identification Register	10008_H	P	BE	Application Reset	5501
I2C1_RST_CTRLA	Reset Control Register A	$1000C_H$	P	P, SV, E	Application Reset	5502
I2C1_RST_CTRLB	Reset Control Register B	10010_H	P	P, SV, E	Application Reset	5502
I2C1_RST_STAT	Reset Status Register	10014_H	P	BE	Application Reset	5503
I2C1_PROTE	PROT Register Endinit	10020_H	U	SV, PROT	Application Reset	5504
I2C1_PROTSE	PROT Register Safe Endinit	10024_H	U	SV, PROT	Application Reset	5505
I2C1_ACCEN_WRA	Write access enable register A	10040_H	U	SE, SV	Application Reset	5507
I2C1_ACCEN_WRB	Write access enable register B	10044_H	U	SE, SV	Application Reset	5508
I2C1_ACCEN_RDA	Read access enable register A	10048_H	U	SE, SV	Application Reset	5508
I2C1_ACCEN_RDB	Read access enable register B	$1004C_H$	U	SE, SV	Application Reset	5509
I2C1_ACCEN_VM	VM access enable register	10050_H	U	SE, SV	Application Reset	5509
I2C1_ACCEN_PRS	PRS access enable register	10054_H	U	SE, SV	Application Reset	5510
I2C1_GPCTL	General purpose control register	10060_H	SV	SV, P	Application Reset	5511

31.8.4.5 Register overview - I2C2 (ascending offset address)

Table 1272 Register overview - I2C2 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
I2C2_CLC1	Clock control 1 register	00000_H	U	P	Kernel Reset	5477
I2C2_ID1	Module identification register	00008_H	U	BE	Kernel Reset	5478
I2C2_RUNCTRL	RUN control register	00010_H	U	P	Kernel Reset	5479
I2C2_ENDDCTRL	End data control register	00014_H	U	P	Kernel Reset	5480
I2C2_FDIVCFG	Fractional divider configuration register	00018_H	U	P	Kernel Reset	5481

(table continues...)

Table 1272 (continued) Register overview - I2C2 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
I2C2_FDIVHIGHCFG	Fractional divider high-speed mode configuration register	$0001C_H$	U	P	Kernel Reset	5481
I2C2_ADDRCFG	Address configuration register	00020_H	U	P	Kernel Reset	5482
I2C2_BUSSTAT	Bus status register	00024_H	U	BE	Kernel Reset	5483
I2C2_FIFOCFG	FIFO configuration register	00028_H	U	P	Kernel Reset	5484
I2C2_MRPSCTRL	Maximum received packet size control register	$0002C_H$	U	P	Kernel Reset	5485
I2C2_RPSSTAT	Received packet size status register	00030_H	U	BE	Kernel Reset	5486
I2C2_TPSCTRL	Transmit packet size control register	00034_H	U	P	Kernel Reset	5486
I2C2_FFSSTAT	Filled FIFO stages status register	00038_H	U	BE	Kernel Reset	5487
I2C2_TIMCFG	Timing configuration register	00040_H	U	P	Kernel Reset	5487
I2C2_ERRIRQSM	Error interrupt request source mask register	00060_H	U	P	Kernel Reset	5489
I2C2_ERRIRQSS	Error interrupt request source status register	00064_H	U	BE	Kernel Reset	5490
I2C2_ERRIRQSC	Error interrupt request source clear register	00068_H	U	P	Kernel Reset	5491
I2C2_PIRQSM	Protocol interrupt request source mask register	00070_H	U	P	Kernel Reset	5491
I2C2_PIRQSS	Protocol interrupt request source status register	00074_H	U	BE	Kernel Reset	5492
I2C2_PIRQSC	Protocol interrupt request source clear register	00078_H	U	P	Kernel Reset	5494
I2C2_RIS	Raw interrupt status register	00080_H	U	BE	Kernel Reset	5495
I2C2_IMSC	Interrupt mask control register	00084_H	U	P	Kernel Reset	5496
I2C2_MIS	Masked interrupt status register	00088_H	U	BE	Kernel Reset	5497
I2C2_ICR	Interrupt clear register	$0008C_H$	U	P	Kernel Reset	5498
I2C2_INTRSET	Interrupt set register	00090_H	U	P	Kernel Reset	5499
I2C2_TXD	Transmission data register	08000_H	nBE	P	Kernel Reset	5500
I2C2_RXD	Reception data register	$0C000_H$	U	BE	Kernel Reset	5500
I2C2_CLC	Clock Control Register	10000_H	P	P, SV, E	Application Reset	5501

(table continues...)

Table 1272 (continued) Register overview - I2C2 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
I2C2_ID	Module Identification Register	10008_H	P	BE	Application Reset	5501
I2C2_RST_CTRLA	Reset Control Register A	$1000C_H$	P	P, SV, E	Application Reset	5502
I2C2_RST_CTRLB	Reset Control Register B	10010_H	P	P, SV, E	Application Reset	5502
I2C2_RST_STAT	Reset Status Register	10014_H	P	BE	Application Reset	5503
I2C2_PROTE	PROT Register Endinit	10020_H	U	SV, PROT	Application Reset	5504
I2C2_PROTSE	PROT Register Safe Endinit	10024_H	U	SV, PROT	Application Reset	5505
I2C2_ACCEN_WRA	Write access enable register A	10040_H	U	SE, SV	Application Reset	5507
I2C2_ACCEN_WRB	Write access enable register B	10044_H	U	SE, SV	Application Reset	5508
I2C2_ACCEN_RDA	Read access enable register A	10048_H	U	SE, SV	Application Reset	5508
I2C2_ACCEN_RDB	Read access enable register B	$1004C_H$	U	SE, SV	Application Reset	5509
I2C2_ACCEN_VM	VM access enable register	10050_H	U	SE, SV	Application Reset	5509
I2C2_ACCEN_PRS	PRS access enable register	10054_H	U	SE, SV	Application Reset	5510
I2C2_GPCTL	General purpose control register	10060_H	SV	SV, P	Application Reset	5511

31.8.4.6 Device specific registers

There are no device specific register changes.

31.8.5 TC4Dx I2C connectivity

The table below lists interfaces to and from this functional block to other blocks in the device.

Table 1273 List of I2C interface signals

Interface signals	I/O	Description
CLOCK_I2C_fI2C	In	I2C clock input
CLOCK_I2C_fSPB	In	SPB clock input
I2C0_IR_DTR	Out	Data transfer interrupt
I2C1_IR_DTR	Out	Data transfer interrupt
I2C2_IR_DTR	Out	Data transfer interrupt
I2C0_IR_ERR	Out	Error interrupt
I2C1_IR_ERR	Out	Error interrupt
I2C2_IR_ERR	Out	Error interrupt
I2C0_IR_P	Out	Protocol interrupt
I2C1_IR_P	Out	Protocol interrupt
I2C2_IR_P	Out	Protocol interrupt
I2C0_PORTS_SCL	Out	Serial clock output
I2C1_PORTS_SCL	Out	Serial clock output
I2C2_PORTS_SCL	Out	Serial clock output
PORTS_I2C0_SCLA	In	Serial clock input
PORTS_I2C1_SCLA	In	Serial clock input
PORTS_I2C2_SCLA	In	Serial clock input
PORTS_I2C0_SCLB	In	Serial clock input
PORTS_I2C1_SCLB	In	Serial clock input
PORTS_I2C2_SCLB	In	Serial clock input
PORTS_I2C0_SCLC	In	Serial clock input
PORTS_I2C1_SCLC	In	Serial clock input
PORTS_I2C2_SCLC	In	Serial clock input
PORTS_I2C0_SCLD	In	Serial clock input
PORTS_I2C1_SCLD	In	Serial clock input
PORTS_I2C2_SCLD	In	Serial clock input
PORTS_I2C0_SCLE	In	Serial clock input
PORTS_I2C1_SCLE	In	Serial clock input
PORTS_I2C2_SCLE	In	Serial clock input

(table continues...)

Table 1273 (continued) List of I2C interface signals

Interface signals	I/O	Description
PORTS_I2C0_SCLF	In	Serial clock input
PORTS_I2C1_SCLF	In	Serial clock input
PORTS_I2C2_SCLF	In	Serial clock input
PORTS_I2C0_SCLG	In	Serial clock input
PORTS_I2C1_SCLG	In	Serial clock input
PORTS_I2C2_SCLG	In	Serial clock input
PORTS_I2C0_SCLH	In	Serial clock input
PORTS_I2C1_SCLH	In	Serial clock input
PORTS_I2C2_SCLH	In	Serial clock input
I2C0_PORTS_SDA	Out	Serial data output
I2C1_PORTS_SDA	Out	Serial data output
I2C2_PORTS_SDA	Out	Serial data output
PORTS_I2C0_SDAA	In	Serial data input
PORTS_I2C1_SDAA	In	Serial data input
PORTS_I2C2_SDAA	In	Serial data input
PORTS_I2C0_SDAB	In	Serial data input
PORTS_I2C1_SDAB	In	Serial data input
PORTS_I2C2_SDAB	In	Serial data input
PORTS_I2C0_SDAC	In	Serial data input
PORTS_I2C1_SDAC	In	Serial data input
PORTS_I2C2_SDAC	In	Serial data input
PORTS_I2C0_SDAD	In	Serial data input
PORTS_I2C1_SDAD	In	Serial data input
PORTS_I2C2_SDAD	In	Serial data input
PORTS_I2C0_SDAE	In	Serial data input
PORTS_I2C1_SDAE	In	Serial data input
PORTS_I2C2_SDAE	In	Serial data input
PORTS_I2C0_SDAF	In	Serial data input
PORTS_I2C1_SDAF	In	Serial data input
PORTS_I2C2_SDAF	In	Serial data input
PORTS_I2C0_SDAG	In	Serial data input
PORTS_I2C1_SDAG	In	Serial data input
PORTS_I2C2_SDAG	In	Serial data input

(table continues...)

Table 1273 (continued) List of I2C interface signals

Interface signals	I/O	Description
PORTS_I2C0_SDAH	In	Serial data input
PORTS_I2C1_SDAH	In	Serial data input
PORTS_I2C2_SDAH	In	Serial data input

31.8.6 TC4Dx I2C revision history

Initial release of the chapter.

32 CLOCK

The clocking system is responsible for providing all clocks that are necessary for microcontroller operation. This is valid for all functional, test, and debug modes. It also provides clocks for external applications that can be selected from internal sources.

The clocking system is functionally divided into two major clock domains:

- System clock domain for clocks covering central processing units, interconnect and data transfers structures, non-volatile memories, timers, debug, trace, and the core functions of the communication interfaces
- Peripheral clock domain for the protocol related parts of the communication interfaces and analog digital converters

Each domain has a master clock which is used to derive the related sub-clocks by programmable division and clock gating. System clocks are synchronous to each other, and peripheral clocks are asynchronous to each other and to the system clocks. Selected clocks can be routed to external outputs for debug, test, and further processing.

32.1 Feature list

- Physical SoC interface for externally provided clocks (crystal oscillator (*XTAL OSC*) or HSSL system clock (*SYSCLK*))
- Internally generated back-up clock provided by an embedded voltage regulator (*EVR*) oscillator for independent operation in the absence of a crystal clock
- Internally generated linear ramp clock to support power-up and power-down, reset, and emergency operation with low supply voltage drops and low current peaks
- Up-scaling of externally supplied clocks up to a maximum frequency of *f*PLL0 as specified in the datasheet, phase locked loops (System *PLL*)
- Up-scaling of externally supplied clocks up to a maximum frequency of *f*PLL1, *f*PLL2 and *f*PLL3 as specified in the datasheet, phase locked loops (Peripheral *PLL*)
- System clock domain with frequency modulation to reduce electromagnetic interference (EMI)
- Peripheral clock domain with low jitter and low noise to comply with jitter and the signal-to-noise ratio requirements of selected communication protocols and analog to digital converters (*ADC*)
- Peripheral clock domain with frequency modulation for optional use to reduce EMI
- Low skew distribution of all system clocks to enable global synchronous operation within the microcontroller unit (*MCU*)
- Independent generation and distribution of all system and peripheral clocks
- Software programming interface for clock configuration and status observation
- Clock monitoring to support ASIL D safety protection and security
- Provision of selected internal system and peripheral clocks to physical outputs of the MCU
- Reset support functions (clock gating and frequency adaptation)

32.2 Functional overview

The clock system is built up as a chain composed from different building blocks which allow different function parts for the complete chain.

The building blocks as shown in the following figure are:

- Basic clock generation with internal and external oscillators (Clock Source)
- Clock speed up-scaling with phase locked loops (*PLL*)
- Clock distribution with clock control units (*CCU*)
- Individual clock configuration within other functional blocks

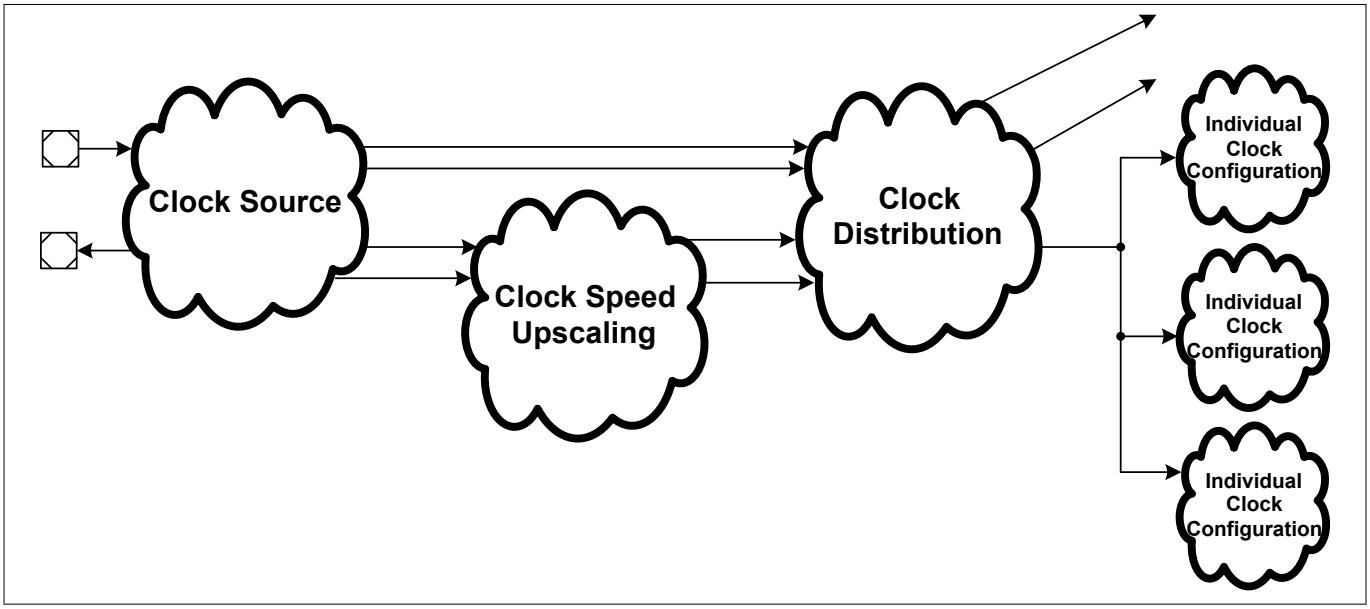


Figure 737 Clock tree

A device specific overview can be found in the device specific section.

For the alarm description please refer to the alarm mapping tables in the SMU functional block and the Safety Manual.

Related information

TC4Dx SMU alarm mapping tables on page 7233

32.3 Functional description

The clock functional block provides very flexible clock generation for the device. During user program execution, the frequencies can be programmed for an optimal ratio between performance and power consumption.

The clocking system provides two major clocking domains:

- System and compute domain containing:
  - Central processing units (*CPU*) and direct memory access units (*DMA*) with its associated memories
  - System infrastructure such as the interconnects, clocking, reset and power control, interrupt system
  - Timers
  - Core function of communication interface controllers (Ethernet, SENT, PSI5 and PSI5-S)
  - Debug and trace system
- Application and Peripheral domain containing
  - Analog to digital converters (*ADC*)
  - Jitter-sensitive protocol parts of the communication interfaces (for example, ERAY, HSCT, QSPI, MSC, ASCLIN, MCAN, I2C, xSPI, and SDMMC)

This separation allows an independent configuration of the desired Baud rates and conversion rates, independent of the power-optimized configuration of the system. In addition, the clocks generated for the system functions are synchronous to each. The clocks generated for the peripheral functions are asynchronous to each other and to the system clocks.

The clocking system architecture can be divided into several sub-functional units for both domains:

- Clock source
  - External input clocks (SYSCLK and XTAL1)
  - Crystal oscillator (XTAL OSC with XTAL1 and XTAL2)
- Backup clock
  - EVR oscillator
  - Ramp oscillator
- Generation
  - System domain PLL (*SYSPLL*)
  - Peripheral domain PLL (*PERPLL*)
- Distribution
  - Configurable and fixed dividers for both domains(*DIV*)
  - Clock multiplexers and clock gates for both domains (*CG*)
  - External clock provision (from both domains to device outputs)
- Hardware-software interface (*HSI*)
  - Flexible Peripheral Interconnection (*FPI*)
  - Configuration and status registers
- Safety and security
  - Safety monitors for both domains(*Safe Mon*)
  - Security monitors (*Sec Mon*)
  - Register Access control and protection (*Acc Ctrl*)

The following diagram shows the top level view of the clocking system including its functional units.

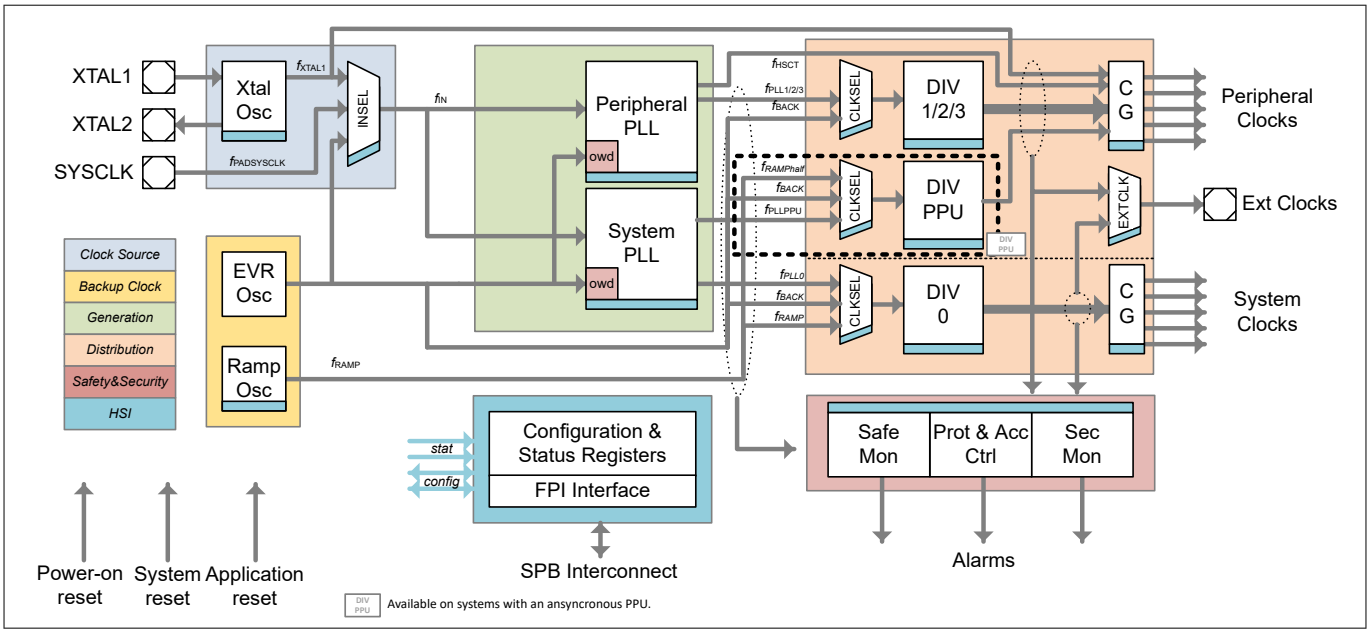


Figure 738 CLOCK block diagram

The figure above shows the clocking system architecture. In devices, where PPU is available, the PPU either runs synchronous or runs slower and asynchronous to the rest of the system. Please refer to the device specific configuration chapter for device specific features and information.

32.3.1 Software interface

In AURIX™ 3rd Generation devices the clocking system has a dedicated flexible peripheral interconnect (*FPI*) slave interface which is connected to the system peripheral bus (*SPB*) interconnect. The FPI slave provides software access to the configuration and status registers of the clocking system.

32.3.2 Clock sources

There are several clock sources available which primary source the system and peripheral clock domains.

There are five clock sources that can be selected to be the input for the downstream connected clock generation (PLL) functions:

1. Internal crystal oscillator driven through the XTAL1 and XTAL2 pads
2. External clock driven through XTAL1 pad
3. HSSL system clock driven through the SYSCLK pad
4. Internal backup clock (for non-safety applications only)
5. fOSC_F is a spike filtered clock only to be used in 20 MHz case to sustain system ESD event, but worse jitter

Software can select between these five sources as described in the following figure and tables:

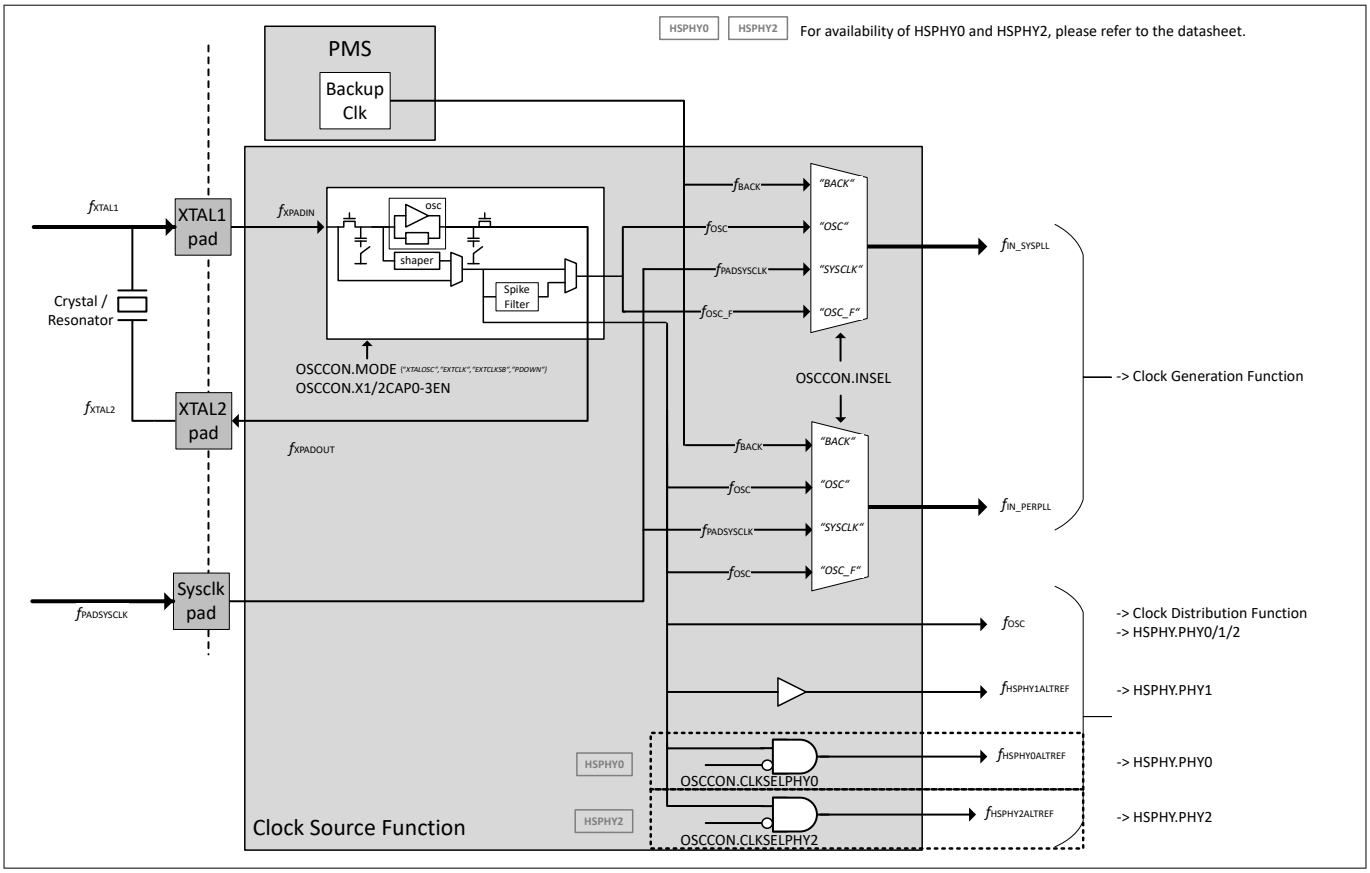


Figure 739 Clock sources

Table 1274 Clock source routing

Clock source	OSCCON.MODE	OSCCON.INSEL	f_{OSC}	f_{IN_SYSPLL}	f_{IN_PERPLL}
Crystal/ resonator	*XTALOSC*	*OSC or OSC_F*	• OSC powered up • Shaper enabled • Shaper not bypassed • Output active	f_{OSC} or f_{OSC_F}	f_{OSC}
External clock	*EXTCLK*	*OSC or OSC_F*	• OSC powered down • Shaper enabled • Shaper not bypassed • Output active	f_{OSC} or f_{OSC_F}	f_{OSC}

(table continues...)

Table 1274 (continued) Clock source routing

Clock source	OSCCON.MODE	OSCCON.INSEL	f_{OSC}	f_{IN_SYSPLL}	f_{IN_PERPLL}
External clock	*EXTCLKSB* [1]	*OSC or OSC_F*	• OSC powered down • Shaper disabled • Shaper bypassed • Output active	f_{OSC} or f_{OSC_F}	f_{OSC}
HSSL system clock	*PDOWN*	*SYSCLK*	• OSC powered down • Shaper disabled • Shaper not bypassed • Output inactive (0)	$f_{PADSYSCLK}$	$f_{PADSYSCLK}$
Backup clock	*PDOWN*	*BACK*	• OSC powered down • Shaper disabled • Shaper not bypassed • Output inactive (0)	f_{BACK}	f_{BACK}

1) This mode should be used only for debug purpose. For normal operation use only modes with shaper enabled.

In addition, software can select the reference clocks for the high-speed PHYs[32] as follows:

Table 1275 HSPHY0 clock reference routing

Clock reference	OSCCON.CLKSELPHY0	$f_{HSPHY0ALTREF}$ (HSPHY.PHY0)
External PCIE refclk	*PCIEREF*[1]	*GND*
Crystal oscillator clock	*OSC*	f_{OSC}

1) In addition, the user has to set the HSPHY PHY0_CTRL1.PRS bits accordingly. For details please refer to the HSPHY UM register section.

Table 1276 HSPHY1 clock reference routing

Clock reference	$f_{HSPHY1ALTREF}$ (HSPHY.PHY1)
Crystal oscillator clock	f_{OSC}

32 For availability of HSPHY0, HSPHY1 and HSPHY2 please refer to the datasheet

Table 1277 HSPHY2 clock reference routing

Clock reference	OSCCON.CLKSELPHY2	$f_{HSPHY0ALTREF}$ (HSPHY.PHY2)
External PCIE refclk	*PCIEREF*[1)]	*GND*
Crystal oscillator clock	*OSC*	f_{OSC}

1) In addition, the user has to set the HSPHY PHY2_CTRL1.PRS bits accordingly. For details please refer to the HSPHY UM register section.

32.3.2.1 Crystal oscillator

The crystal oscillator circuit is used to provide a reference clock for the clock generation function, that means the up-scaling PLLs.

The circuit is designed to work with both an external crystal or ceramic resonator, or an external stable clock source. The main oscillator stage consists of an inverting amplifier with XTAL1 as the input, and XTAL2 as the output with an integrated feedback resistor. A shaper stage is used to convert a sinusoidal oscillation at the pads into a digital clock signal f_{OSC} that can be used as reference for the PLLs and the high speed physical layers (*PHY*). The shaper can be optionally bypassed. A switchable load capacitor array is connected to both pins, XTAL1 and XTAL2, to enable a low bill of material option with integrated on-chip load capacitors. A control register applies the settings for capacitor selection, shaper function and oscillator control like gain selection.

As shown in the figure below, the oscillator circuit consists of the following components:

- Input and output pads, XTAL1 and XTAL2
- Main oscillator stage, a three-stage pierce oscillator
- Adjustable load capacitor arrays, C_1 and C_2
- Shaper circuit and bypass multiplexer
- Control register
- External component, crystal or ceramic resonator

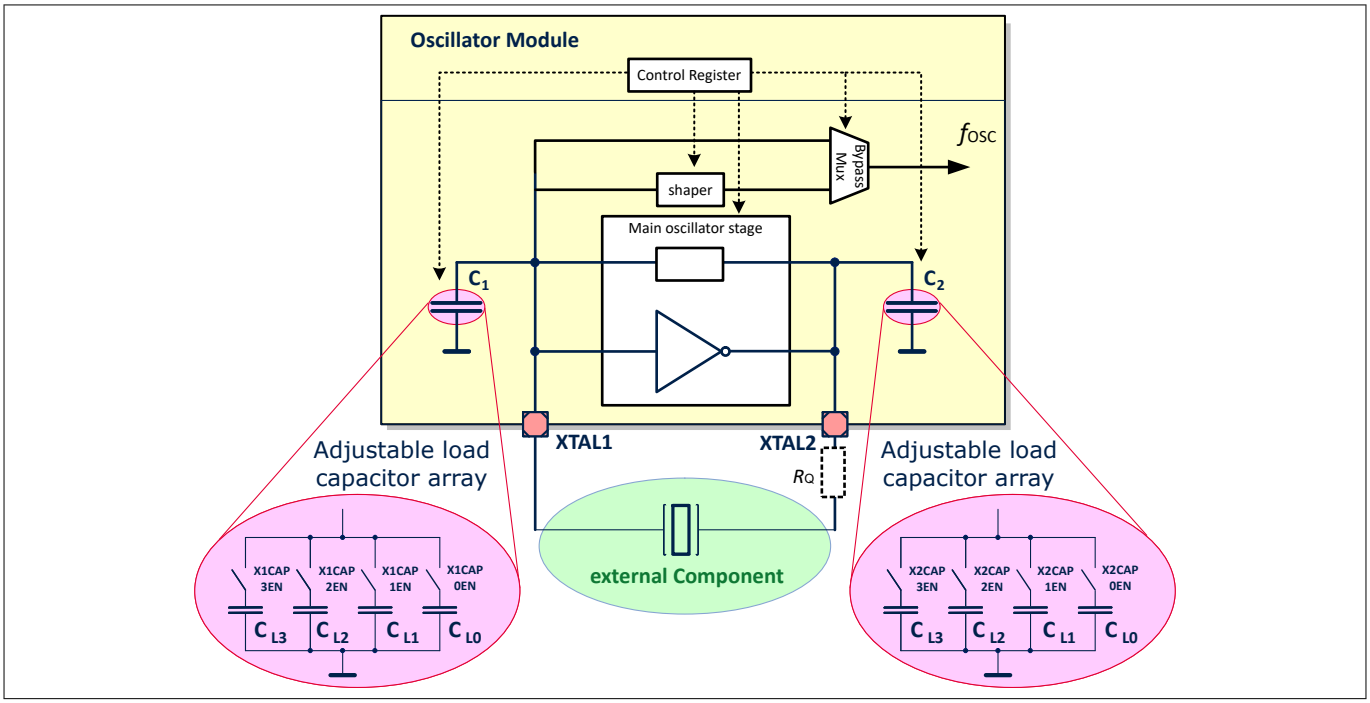


Figure 740 Oscillator block diagram

The crystal oscillator supports four basic operating modes as follows:

1. Crystal mode: Generates an output clock by maintaining oscillation of an external crystal or ceramic resonator
2. External clock mode with shaper: Generates an output clock by forwarding and shaping an external clock
3. External clock mode without shaper: Generates an output clock by forwarding an external clock; this mode is only used for debug purpose, not for field applications, performance is not guaranteed, for field applications use a mode with shaper enabled
4. Power down mode: Oscillator powered down and output is inactive (no signal)

The crystal oscillator can be accessed by software through the following register:

- Crystal oscillator control register (OSCCON): oscillator control parameters, power enable, operating mode, and shaper control

External crystal or ceramic resonator mode

An external oscillator load circuitry can be used, connected to both pins, XTAL1 and XTAL2. Two external load capacitors C_1 and C_2 are needed or the internal loads can be used and configured. In addition, depending on the crystal or ceramic resonator type, a series resistor R_2 is needed to limit the current. A test resistor R_Q may be temporarily inserted to measure the oscillation allowance (negative resistance) of the oscillator circuitry. R_Q values are typically specified by the crystal or ceramic resonator vendor. The C_1 and C_2 values shown in the datasheet can be used as starting points for the negative resistance evaluation and for non-production systems. The crystal oscillator is supplied by the noise reduced $V_{DDEXTOSC}$ and $V_{SSEXTOSC}$ power domains.

The exact values and related operating range are dependent on the crystal or ceramic resonator frequency, and have to be determined and optimized together with the crystal or ceramic resonator vendor using the negative resistance method. Oscillation measurement with the final target system is strongly recommended to verify the input amplitude at XTAL1 and to determine the actual oscillation allowance (margin negative resistance) for the oscillator-crystal or ceramic resonator system. The oscillator can also be used in combination with a ceramic resonator. The final circuitry must also be verified by the resonator vendor. When using an external crystal or ceramic resonator, its frequency must be within the allowed range as specified in the datasheet.

The following figure shows the recommended external circuits for both operating options: External crystal or ceramic resonator mode with and without external components.

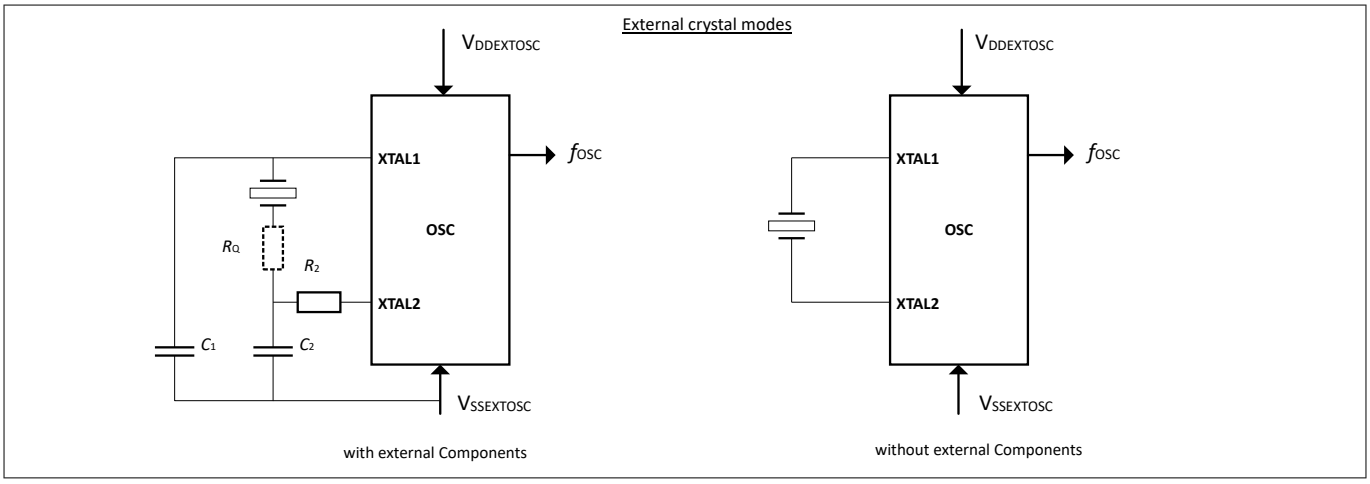


Figure 741 External circuit for crystal or ceramic resonator operation

External input clock mode

When using an external clock signal it must be connected to XTAL1, however, XTAL2 must be left open (unconnected).

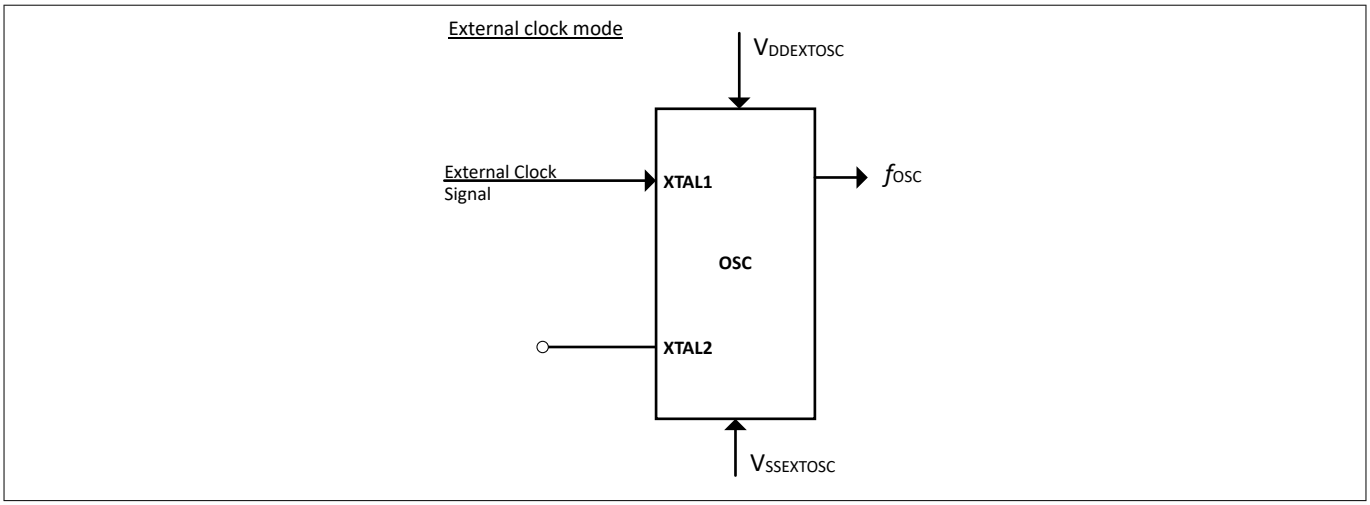


Figure 742 Direct clock input

When supplying the clock signal directly, that means not using an external crystal or ceramic resonator and bypassing the oscillator, the input frequency needs to be in a suitable range for the digitally controlled oscillator (*DCO*) of the PLLs (the value is specified in the datasheet). When using the external input clock mode, the shaper bypass mode is not recommended.

Configuration

A configuration of the oscillator is always required before an external crystal or ceramic resonator can be used as clock source. After a power-on (cold, warm) or system reset the oscillator is disabled and needs to be configured as described in this section. During an application reset, the oscillator is not affected and operates as previously configured, no re-configuration is required for this case.

The oscillator can be configured through the register OSCCON:

It must be enabled by setting bit-field OSCCON.MODE = 000_B. If the integrated capacitors need to be used by the user, they can be enabled through the bit-fields OSCCON.X1CAP0-3EN and OSCCON.X2CAP0-3EN.

***Note**: After powering up the crystal oscillator by setting bit-field OSCCON.MODE = 000_B, it takes up to 3 ms until the oscillator produces a reliable and stable output clock. The user shall implement a waiting period accordingly before any other resources depending on the crystal oscillator need to be used. For further details, refer to the crystal oscillator entries in the datasheet.*

EMC robust clock source fOSC_F

In normal application, the oscillator shaper has a bandwidth of approximately 100 MHz to ensure its jitter performance. In some application, the ECU does not have a metal case and the system ESD pulse can go through the shaper, introducing PLL loss of lock. In this case, the OSC_F can be used instead. An additional low pass filter with a bandwidth of around 50 MHz is sufficient to filter out the system ESD pulse by trading against the jitter performance. When the crystal has a frequency greater than 20 MHz, the OSC_F cannot be used.

Oscillator watchdog

In combination with the PLLs, an input signal monitoring function is implemented. This feature is defined to detect severe malfunctions of an external crystal or ceramic resonator. The system can detect a "loss of clock" at the input or a "too high, too low" input frequency (operation on a higher or lower harmonic).

Each *PLL* (SYSPLL and PERPLL) provides a dedicated oscillator watchdog function that monitors the incoming reference clock frequency f_{IN}.

Both monitors are configured and observed through the following registers:

- OSCMON1: monitor configuration
- OSCSTAT: monitor status

The expected input frequency for both monitors is defined by the bit-field OSCMON1.OSCVAL:

$$f_{IN_EXP} = OSCMON1.OSCVAL - 1 + 16\,Mhz \quad (52)$$

- Bit-fields OSCMON1.SMONEN: enable or disable monitor in SYSPLL
- Bit-fields OSCMON1.PMONEN: enable or disable monitor in PERPLL
- Bit-fields OSCSTAT.SOWDS: reports if input frequency for the SYSPLL is too high, too low or in range
- Bit-fields OSCSTAT.POWDS: reports if input frequency for the PERPLL is too high, too low or in range

Each time before (re)configuring the monitoring function, the watchdog circuit and the related SMU alarms have to be disabled. This avoids unintended alarms in the safety and security alarm management unit (*SMU*). Now the value of the expected input frequency can be set in bit-field OSCMON1.OSCVAL. In the following, the oscillator watchdogs have to be (re)enabled. This initializes the watchdog circuits, presets the status flags OSCSTAT.SOWDS = OSCSTAT.POWDS *InValid*, and re-starts the monitoring with the new configuration. The watchdogs need a full monitoring period of 10 µs + 2 µs evaluation time to flag a valid status. When the status flags are finally changed into *InRange* by the circuit, the input frequency is within the expected range and the SMU alarms can be enabled again.

***Note**: The oscillator watch dog cannot check if the input frequency is in full legal range for PLL operation. It is primarily intended to check if a crystal is oscillating on a harmonic and if an input frequency is available at all. The user has to ensure that the applied input frequency is basically in the allowed range for PLL input. Please check the datasheet accordingly.*

32.3.3 Back-up clock

In AURIX™ 3rd Generation there are two back-up clocks available. Depending on the operating state (power-up, drive, power-down, power-safe, safe-state, and so on) and the user-application, they can be used as alternate clock sources instead of the PLL output clocks.

32.3.3.1 EVR oscillator

An embedded voltage regulator oscillator (*EVR oscillator*) clock with a trimming functionality is available as an alternate primary clock source.

The EVR oscillator provides a stable and reliable 100 MHz clock that can be used for the system if its trimmed accuracy[33] is sufficient for the current operating mode of the application (for example, during power up and boot when the PLLs are not yet activated or in case of a PLL fail operation). It provides less accuracy than an external crystal or ceramic resonator. In addition, it is used as diverse clock for various safety mechanisms.

The back-up clock is always enabled, that means it cannot be disabled by the user, or otherwise controlled in a way that could prevent its general operation.

No other control bits are available except the ones for selecting the backup-clock as source for the clock distribution of the system and peripheral clocks (controlled by bit-field CCUCON.CLKSELS = 01_B) and as optional (non-safety) input reference for the SYSPLL and PERPLL (controlled by bit-field OSCCON.INSEL = 00_B).

32.3.3.2 Ramp oscillator

A second on-chip oscillator is introduced to provide a linear frequency ramp clock to minimize current jumps in the system clock tree during power up and power-down and resets and when a clock emergency occurs.

The ramp oscillator has a register interface as follows:

- Ramp oscillator control register 0 (RAMPCON0): control parameters for the ramp oscillator
- Ramp oscillator status register (RAMPSTAT): actual status of the ramp oscillator

33 The accuracy after trimming is listed in the datasheet

The ramp oscillator's output clock f_{RAMP} represents another alternate clock source during power-up and down, clock emergency switching, or resets, and can be selected as clock source by setting the bit-field CCUCON.CLKSELS accordingly. The ramp process is software controllable by programming an upper target frequency and a ramp command in RAMPCON0. The lower frequency limit equals the frequency of the EVR oscillator which is 100 MHz. When software issues a ramp command, the ramp oscillator executes a linear frequency sweep based on the register defined key parameters. Once the target frequency has been reached, the ramp stops. Ramp status information can be continuously read by software through RAMPSTAT. For further details, refer to the register description.

In order to perform a clock ramp, the oscillator needs to be powered up first by setting the bit-field RAMPCON0.PWR = 1_B. The oscillator is ready to accept a ramp command when the bit-field RAMPSTAT.ACTIVE = 1_B.

RAMPCON0 needs to be programmed with the ramp parameters and a valid command in order to execute a sequence as follows:

- Bit-field RAMPCON0.UFL - defines the upper target frequency in 1 MHz steps for ramp-up, for ramp-down the target is always 100 MHz and cannot be changed. To avoid current over- and undershoots the upper target frequency should be always 100 MHz below the PLL output target frequency
- The step size is fixed to 30 MHz, valid for both directions.
- Bit-field RAMPCON0.CMD - defines the command to be executed which is either *RampUp* or *RampDown*

  Writing the command works as a trigger, that means it is write only, starts the sequence and clears automatically. By writing a valid command, the control parameters are validated towards the ramp oscillator. Changing them without applying a valid command has no immediate effect on the ramp operation

In the RAMPSTAT it is possible to observe the actual status of the ramp oscillator:

- Bit-field RAMPSTAT.ACTIVE - indicates if the ramp oscillator is powered up and ready to accept and execute a command
- Bit-field RAMPSTAT.STAT - reports the actual state of the oscillator's ramp operation which can be *Idle*, *RampUp*, *RampDown*
- Bit-field RAMPSTAT.FSTAT - reports a generic frequency status which reflects if the oscillator resides at the lower or upper target frequency or somewhere in between
- Bit-field RAMPSTAT.FLLLOCK - indicates if the ramp oscillator is locked on the target frequency, which is either 100 MHz on the lower end or at RAMPCON0.UFL on the upper end

The following figures demonstrate a ramp up and down sequence. The values in the figure graphs like frequency, step size and waiting time correspond with the register values listed.

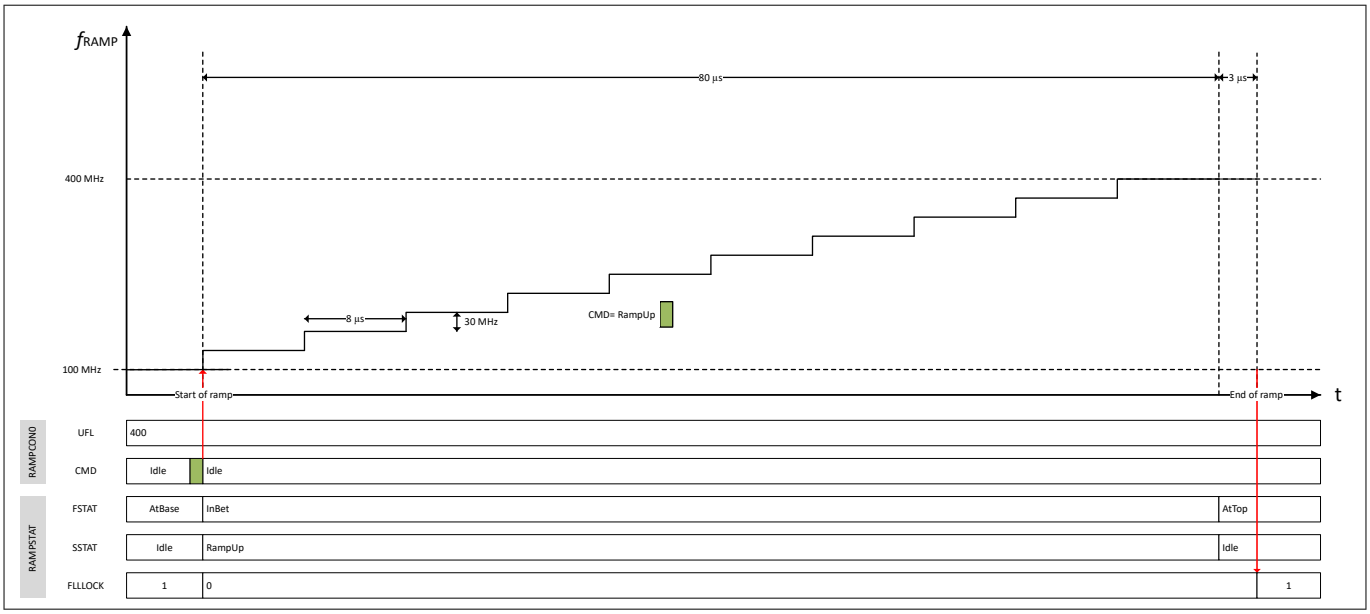


Figure 743 Ramp up sequence

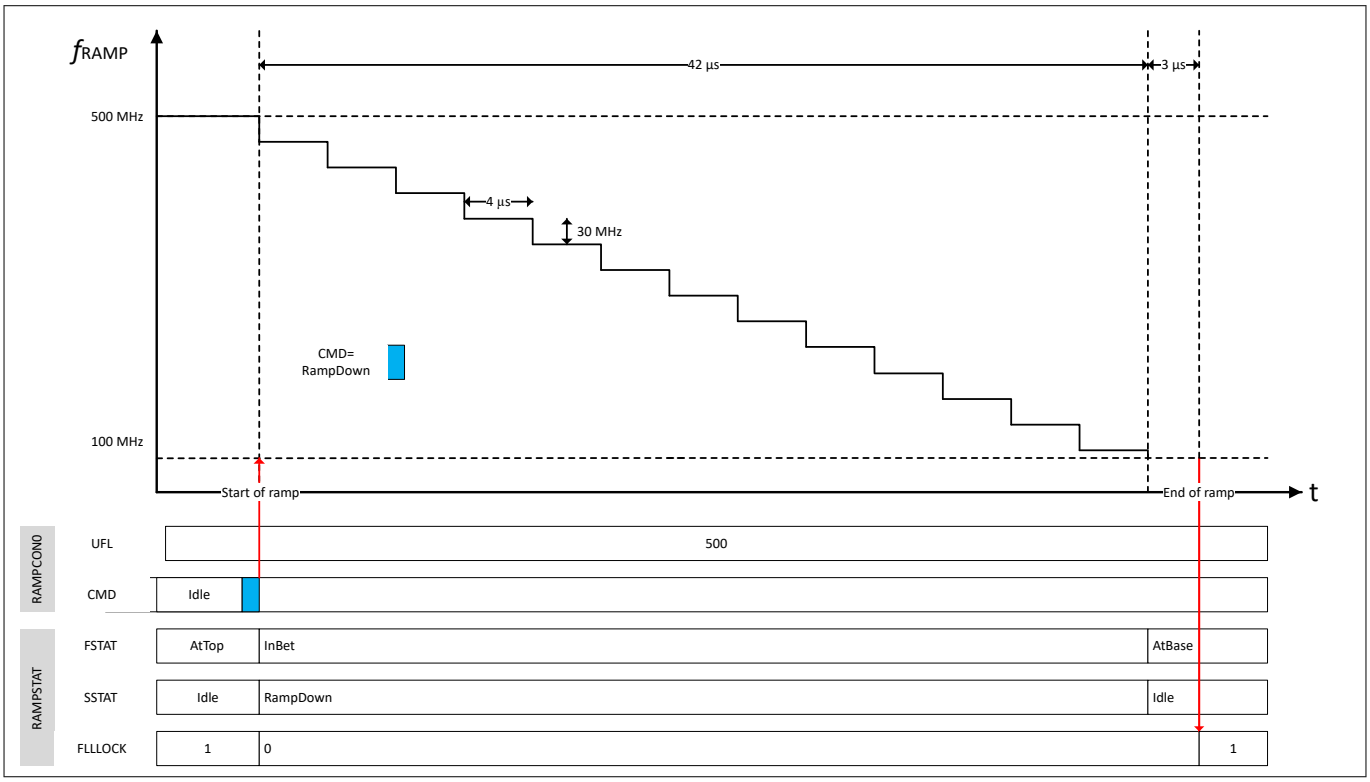


Figure 744 Ramp down sequence

It is not possible to ramp down to a frequency other than the 100 MHz base frequency. To change, for example, from 500 MHz to 300 MHz, the application software has to ramp down to 100 MHz first, reprogram the upper target frequency from 500 MHz to 300 MHz and ramp up again.

In addition to software control, there is hardware control over the ramp oscillator. This is needed to execute frequency ramps for application and system reset and clock emergency switching support. During reset operation or PLL loss of lock, it is desirable to gradually reduce system clock frequency to minimize current over- or undershoots.

A detailed description on how to setup and control the ramp oscillator is given in the Power Pattern chapter.

Attention**:* ***The ramp oscillator is used to support application reset and emergency switching, it shall not be ramped down after the system has been set to its target frequency and is running on the SYSPLL. The ramp oscillator needs to reside at its upper frequency limit in order to establish a smooth clock switch over from PLL to ramp oscillator if either an application reset sequence is started or an emergency switch has to be executed. That means: do not ramp down or switch off the ramp oscillator while running on the SYSPLL. The ramp oscillator has a maximum upper frequency level which is 100 MHz below the PLL frequency. At the same time the upper frequency level of the ramp oscillator should be at least 130 MHz. This implies a minimum PLL frequency of 230 MHz for which the ramp oscillator can be used. Below that frequency level the ramp oscillator must be powered down and must not be used.

***Note**:* *The ramp oscillator shall not be used as a permanent system clock source outside of a ramp-up or a ramp-down scenario..*

Related information

Start-up and shut-down on page 5560

32.3.4 Clock generation

Typical CPU operating speeds are often ten times (or more) faster than the speed of the crystal used as the clock source. Therefore an up-scaling of the externally provided clock source clock is required.

Two phase locked loops (*PLL*) are provided for the up-scaling.

32.3.4.1 SYSPLL

The SYSPLL can convert a low-frequency external clock signal to a high-speed internal clock for maximum performance. It supports a wide range of input and output frequencies by varying the different divider factors.

The SYSPLL also has fail-safe logic that detects degenerate external clock behavior, such as abnormal frequency deviations or a total loss of the external clock. It reports if it loses its lock on the reference clock to support a switch over of the system clocks from PLL clock to internal clock sources (for example, the EVR or ramp clock).

32.3.4.1.1 Features

- Digital controlled oscillator (*DCO*)
- Digital phase detector and loop filter
- DCO lock detection
- Input divider P
- Feedback divider N
- P and N divider establish DCO frequency f_{DCO} as specified in the datasheet
- Two independent outputs for system clocks and separate PPU clock[34)]
- Output integer divider K2 for system clocks
- Output integer divider K3 for PPU clock[34)]
- Glitch-less programming of output dividers K2 and K3
- P, N, K2 dividers combined establish SYSPLL output frequency (f_{PLL0})
- P, N, K3 dividers and K3 pre-divider combined establish PPU-PLL output frequency (f_{PLLPPU})[34)]

34 Only present in devices with an asynchronous PPU. Please refer to the device specific chapters for availability

32 CLOCK

- Oscillator watchdog, crystal harmonic and signal presence detection of input frequencies
- Random frequency modulation

32.3.4.1.2 Functional description

The SYSPLL consists of the following building blocks:

- Control and status register block to provide a programming interface for the control settings and the status observation
- Input divider to scale down the input frequency f_{IN} by a factor P which is configurable through the register section
- PLL-core that consists of:
  - Digital controlled oscillator and a phase locked loop. The loop locks to the input reference frequency f_{REF} while multiplying it with the factor N which is configurable through the register section
  - Lock detection circuit that observes and reports the locking status to the register section
  - Random frequency modulator that can optionally modify the DCO frequency to produce a spread spectrum effect. The relative modulation amplitude (MA %) is configurable through the register section
- Output stage for system clocks composed of an integer divider that scales down the DCO frequency by a factor K2, configurable through the register section
- Output stage for PPU clock composed of a pre-divider for fractional divisions (1.0, 1.1, 1.2, 1.4, 1.6, 1.7, 2.0) and an integer divider that scales down the DCO frequency by a factor K3, configurable through the register section[35)]
- Oscillator watchdog based on the backup clock f_{BACK} to supervise the input frequency f_{IN}, configurable through the register section

The following figure shows the general structure of the SYSPLL. In devices with a synchronous PPU there is no separate PPU clock output because the PPU runs synchronous to the rest of the system and its clock is generated from the PLL0 output. In devices with an asynchronous PPU, the PPU clock is generated by the second SYSPLL output.

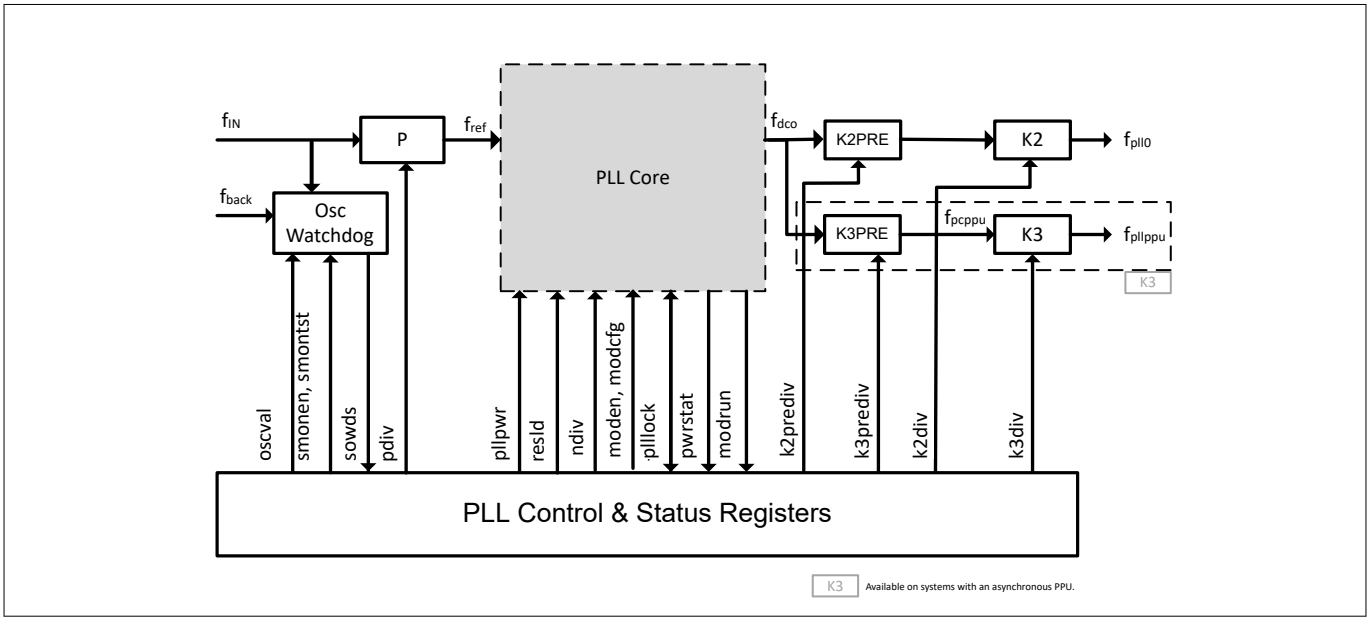


Figure 745 System PLL block diagram

SYSPLL register interface

All control and status values of the SYSPLL are stored in and accessed through:

- SYSPLL configuration register 0 (SYSPLLCON0): P and N divider, power enable, modulation enable, reset lock detection

35 Only present in devices with an asynchronous PPU. Please refer to the device specific chapters for availability

- SYSPLL configuration register 1 (SYSPLLCON1): K2, K3[35] dividers and K3[35] pre-divider
- SYSPLL configuration register 2 (SYSPLLCON2): Modulation control
- SYSPLL status register (SYSPLLSTAT): Power status, lock status, modulation status
- Modulation trim 0 configuration register (SMODCON0): Modulation trimming control

SYSPLL power control

The SYSPLL is disabled after cold and warm power-on reset and needs to be powered up separately by setting bit-field SYSPLLCON0.PLLPWR = 1_B. The PLL power-up status is eventually flagged through bit-field SYSPLLSTAT.PWRSTAT = 1_B.

The PLL can be set into power down mode when it is not needed. The power down mode is entered by setting bit-field SYSPLLCON0.PLLPWR = 0_B and flagged through bit-field SYSPLLSTAT.PWRSTAT = 0_B. While the SYSPLL is in power down mode, no output frequency is generated.

***Note**: Switching to SYSPLL power down mode shall only be performed when the system clock tree is sourced by the backup clock or ramp clock (controlled through bit-field CCUCON.CLKSELS).*

Input frequency watchdog

PLL operation requires a stable input clock frequency of f_{IN}. Therefore it is recommended to check and monitor if an input frequency f_{IN} is available by checking bit-field OSCSTAT.SOWDS == 00_B before enabling and using the PLL.

In case the PLL is driven by the crystal oscillator, an additional wait time after powering up of the crystal oscillator has to be applied. This is due to the fact that the crystal oscillator takes up to 3 ms to generate a reliable output clock after it has been powered up. For detailed timing information of the crystal oscillator, please refer to the datasheet.

Basic SYSPLL operation

The DCO and output frequencies[36] of the SYSPLL are determined by the values of the three dividers and the input reference frequency f_{IN}:

- SYSPLLCON0.PDIV
- SYSPLLCON0.NDIV
- SYSPLLCON1.K2DIV
- SYSPLLCON1.K3DIV and SYSPLLCON1.K3PREDIV[35]

The PLL core input reference frequency f_{REF} is generated by dividing the input frequency f_{IN} by a factor P. The PLL core multiplies the input reference by a factor N to produce the DCO frequency f_{DCO}:

$$f_{\mathrm{REF}} = f_{\mathrm{IN}} \div P,\ f_{\mathrm{DCO}} = f_{\mathrm{REF}} \times N \tag{53}$$

The recommendation is to use as a maximum the f_{DCO} as specified in the datasheet and P to 1, which means SYSPLLCONx.PDIV = 0.

In the system clock output stage, the DCO frequency is divided down by a factor K2.

The output frequency f_{PLL0} is given by:

$$\begin{aligned} f_{\mathrm{PLL0}} &= f_{\mathrm{DCO}} \div (K2 \times 1.0) \rightarrow K2\mathrm{PRE} = 0;\ \text{this value must be used for normal operation} \\ f_{\mathrm{PLL0}} &= f_{\mathrm{DCO}} \div (K2 \times 2.0) \rightarrow K2\mathrm{PRE} = 1 \\ f_{\mathrm{PLL0}} &= f_{\mathrm{DCO}} \div (K2 \times 1.2) \rightarrow K2\mathrm{PRE} = 2 \\ f_{\mathrm{PLL0}} &= f_{\mathrm{DCO}} \div (K2 \times 1.6) \rightarrow K2\mathrm{PRE} = 3 \end{aligned} \tag{54}$$

35 Only present in devices with an asynchronous PPU. Please refer to the device specific chapters for availability

36 Please refer to the datasheet for valid SYSPLL frequencies to determine the appropriate N, P and K2 settings.

SYSPLL lock detection

The SYSPLL has a lock detection feature, that supervises the DCO part of the SYSPLL to differentiate between stable and unstable DCO circuit behavior. The lock detector marks the PLL as locked if the two input frequencies of the lock detector f_{REF} and f_{DIV} (=f_{DCO}÷N) have a phase difference smaller than ½ reference cycle for more than 224 reference cycles.

However, the unlock condition is true immediately if the phase error is bigger than the above given limit. Smaller phase errors can be handled by the PLL without getting unstable and are not marked as "loss of lock".

The SYSPLL may become unlocked, due to a break of the crystal or ceramic resonator or the external clock line. In such a case, an alarm is signaled to the SMU and the PLL switches to an open loop mode and keeps on free running at the last locked frequency.

A modification of the two dividers P and N has a direct influence to the DCO frequency and could lead to a loss of the lock status. However, a modification of the K2 divider has no impact on the lock status, but still changes the SYSPLL output frequency f_{PLL0} directly.

The lock status can be observed in bit-field SYSPLLSTAT.PLLLOCK and the lock detection can be reset by setting bit-field SYSPLLCON0.RESLD = 1_B. Applying the reset will close the loop and eventually enable it to lock again.

Attention**:* ***Once unlocked, the PLL will not make an attempt to lock again until issuing a lock detection reset or a reprogramming of the P or N dividers.

Frequency modulation

The SYSPLL output frequency f_{PLL0} can additionally be modified by a low-frequency modulation to reduce EMI. A random sequence is added to the DCO control word resulting in a randomly modulated f_{DCO}. The rate of the modulation is defined by f_{REF}.

To configure the modulation follow these steps in the given order:

Turn on or off modulation for SYSPLL

```
CLOCK_SYSPLLCON0.MODEN = 0                    // Deactivate Frequency Modulation
CLOCK_SYSPLLCON2.MODCFG = (0x7400 | MG)       // Program Modulation Gain (MG)
if (MG)
     CLOCK_SMODCON0 = 0x80000000              // CAL_EN = 1
     Wait(100ns)                              // Wait to ensure toggling
     CLOCK_SMODCON0 = 0x00000000              // CAL_EN = 0
     CLOCK_SYSPLLCON0.MODEN = 1               // Enable Modulation
     Wait until CLOCK_SYSPLLSTAT.MODRUN == 1  // Wait until modulation is running
end
```

***Note**:* *To turn off modulation, simply set MG=0 in the above sequence*

***Notes**:*
To change the modulation gain:

1. *Turn off modulation*
2. *Turn on modulation with different gain*

The modulation is enabled using bit-field SYSPLLCON0.MODEN. The modulation itself alters the DCO frequency randomly within the range of the configured relative modulation amplitude (MA%). The modulation is controlled through bit-field SYSPLLCON2.MODFCG as follows:

- MODCFG[9:0] = modulation gain

$$\text{modulation gain} = \frac{64}{3.6\text{MHz}} \times f_{\text{DCO}} \times \text{MA}$$
$$\text{Example: MA} = 2\,\% \text{ at} f_{\text{DCO}} = 800\,\text{MHz} \Rightarrow \text{modulation gain } = 0x11C \quad (55)$$

The modulation is performed in a way that the resulting accumulated jitter added by the modulation stays below J_{TOT} (when MA equals to 1.25%).

Configuration sequence for the K-dividers for configuration and re-configuration of PLLs

Special care should be taken during configuration and re-configuration of PLLs with respect to programming the K-dividers to avoid wrong K-divider or dead clock at one of the PLL output paths which can be recovered only through a system reset.

Follow the below configuration sequence for the K-dividers for the configuration or re-configuration of any PLL:

PLL configuration phase

1. PLL off
2. Check for CCU lock: CLOCK_CCUSTAT.B.LCK == 0 with a timeout of at least 100 µs
3. Configure K-dividers in xPLLCON1 register
    a. Set KxPREDIV = div by 2 (xPLLCON1.KxPREDIV = div by 2)
    b. Set KxDIV = 0x01 (xPLLCON1.KxDIV = 0x01)
    c. Set other PLL parameters as described in the user manual
4. PLL on (power PLL)
5. Check for CCU lock: CLOCK_CCUSTAT.B.LCK == 0 with a timeout of at least 100 µs
6. Check if PLL is powered: xPLLSTAT.PWRSTAT == 1 with a timeout of at least 100 µs
7. Wait for 1ms to avoid jitter
8. Check for PLL lock: xPLLSTAT.PLLLOCK == 1 with a timeout of at least 100 µs
9. Configure KxPREDIV (if necessary)
    a. Set KxPREDIV to final value
10. Wait for 10 µs and check for CCU lock: CLOCK_CCUSTAT.B.LCK == 0 with a timeout of at least 100 µs
11. Configure KxDIV
    a. Set KxDIV to final value
    b. Check for CCU lock: CLOCK_CCUSTAT.B.LCK == 0 with a timeout of at least 100 µs

Changing the KxDIV value during run time (KxPREDIV is not allowed to change during run time)

1. Check for PLL lock: xPLLSTAT.PLLLOCK == 1 with a timeout of at least 100 µs
2. Check for CCU lock: xLOCK_CCUSTAT.B.LCK == 0 with a timeout of at least 100 µs
3. Configure K-dividers in xPLLCON1 register
    a. KxDIV = 0x01 (xPLLCON1.KxDIV = 0x01)
4. Wait for 10 µs and check for CCU lock: CLOCK_CCUSTAT.B.LCK == 0 with a timeout of at least 100 µs
5. Configure KxDIV
    a. Set KxDIV to final value
    b. Check for CCU lock: CLOCK_CCUSTAT.B.LCK == 0 with a timeout of at least 100 µs

Related information

Crystal oscillator on page 5531

32.3.4.2 PERPLL

The PERPLL can convert a low-frequency external clock signal to a high-speed internal clock for maximum performance. In AURIX™ 3rd Generation, the PERPLL shall be configured to f_{DCO} = 800 MHz.

The PERPLL also has fail-safe logic that detects degenerate external clock behavior such as abnormal frequency deviations or a total loss of the external clock. It can execute emergency actions if it loses its lock on the external clock.

32.3.4.2.1 Features

- Digital controlled oscillator (*DCO*)
- Digital phase detector and loop filter
- DCO lock detection
- Three separate outputs, f_{PLL1}, f_{PLL2}, f_{PLL3}
- Additional, direct DCO output to deliver 800 MHz HSCT clock to HSCT PHY
- Input divider P
- Feedback divider N
- P and N divider establish DCO frequency, f_{DCO} = 800 MHz as recommended
- Output divider per output, K2, K3, K4
- Glitch-less programming of output dividers K2, K3, K4
- K2, K3, K4 pre-divider for fractional divisions by 1.0, 1.1, 1.2, 1.4, 1.6, 1.7, and 2.0
- P, N, K2, K3, K4 dividers and K2, K3, K4 pre-divider combined establish PLL output frequencies
- Oscillator watchdog, crystal harmonic and signal presence detection of input frequencies
- Optional random frequency modulation to reduce EMI for GPS use-case

32.3.4.2.2 Functional description

The PERPLL basically consist of the following building blocks:

- Control and status register block to provide a programming interface for the control settings and the status observation
- Input divider to scale down the input frequency f_{IN} by a factor P which is configurable through the register section
- PLL-core that consists of:
  - Digital controlled oscillator and a phase locked loop. The loop locks to the input reference frequency f_{REF} while multiplying it with the factor N which is configurable through the register section
  - Lock detection circuit that observes and reports the locking status to the register section
  - Random frequency modulator that can optionally modify the DCO frequency to produce a spread spectrum effect. The relative modulation amplitude (MA %) is configurable through the register section
- Three output stages composed of pre-dividers for fractional divisions (1.0, 1.2, 1.6 and 2.0) and integer dividers that scale down the DCO frequency by a factor K2,K3, K4 configurable through the register section
- Oscillator watchdog based on the backup clock f_{BACK} to supervise the input frequency f_{IN}, configurable through the register section

The following figure shows the PERPLL block structure.

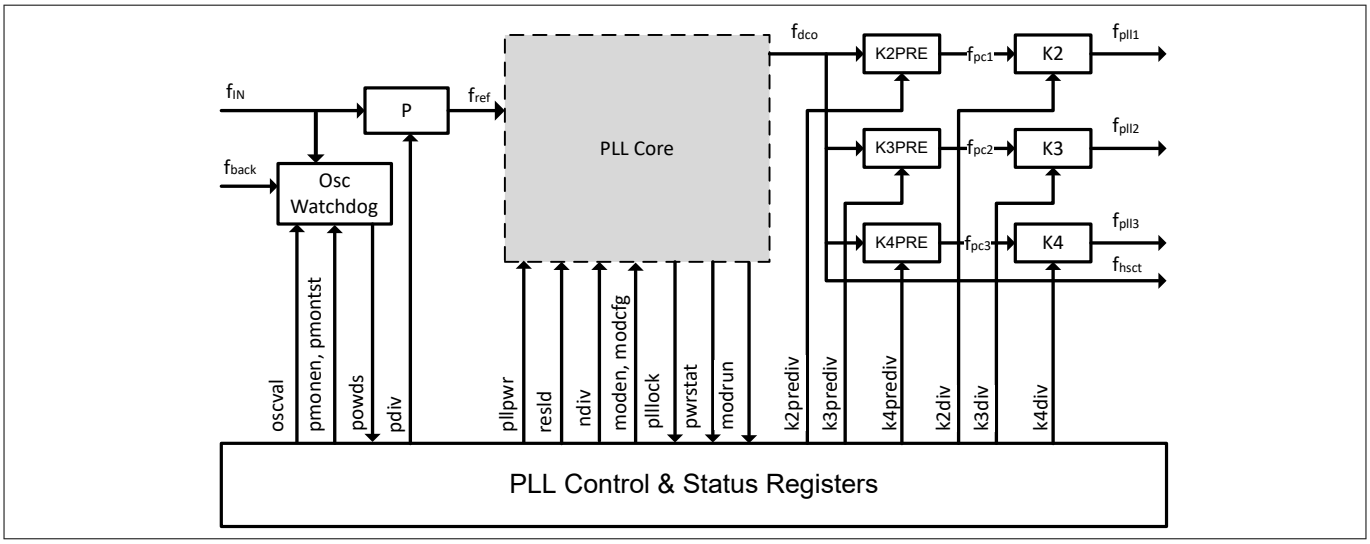


Figure 746 PERPLL block diagram

PERPLL register interface

All control and status values of the SYSPLL are stored in and accessed through:

- PERPLL configuration register 0 (PERPLLCON0): P and N divider, power enable, reset lock detection
- PERPLL configuration register 1 (PERPLLCON1): K2, K3, K4 dividers and K2, K3, K4 pre-dividers
- PERPLL configuration register 2 (PERPLLCON2): Modulation control
- PERPLL status register (PERPLLSTAT): power status, lock status
- Modulation trim 0 configuration register (PMODCON0): Modulation trimming control

PERPLL power control

The PERPLL is disabled after power-on reset and needs to be powered up separately by setting bit-field PERPLLCON0.PLLPWR = 1_B. The PLL power-up status is eventually flagged through bit-field PERPLLSTAT.PWRSTAT = 1_B.

The PLL can be set into power down mode when it is not needed. The power down mode is entered by setting bit-field PERPLLCON0.PLLPWR = 0_B and flagged through bit-field PERPLLSTAT.PWRSTAT = 0_B. While the PERPLL is in power down mode, no output frequency is generated.

***Note**: Switching to PERPLL power down mode shall only be performed when the peripheral clock tree is sourced by the backup clock (controlled through bit-field CCUCON.CLKSELP).*

Input frequency watchdog

PLL operation requires a stable input clock frequency of f_{IN}. Therefore it is recommended to check and monitor if an input frequency f_{IN} is available by checking bit-field OSCSTAT.POWDS == 00_B before enabling and using the PLL.

In case the PLL is driven by the crystal oscillator, an additional wait time after powering up of the crystal oscillator has to be applied. This is due to the fact that the crystal oscillator takes up to 3 ms to generate a reliable output clock after it has been powered up. For detailed timing information of the crystal oscillator, please check the datasheet.

Basic PERPLL operation

The DCO frequency and the three output frequencies[37] of the PERPLL are determined by the values of the dividers and the input reference frequency f_{IN}:

- PERPLLCON0.PDIV
- PERPLLCON0.NDIV
- PERPLLCON1.K2DIV, PERPLLCON1.K3DIV and PERPLLCON1.K4DIV and PERPLLCON1.K2PREDIV, PERPLLCON1.K3PREDIV and PERPLLCON1.K4PREDIV

The PLL core input reference frequency f_{REF} is generated by dividing the input frequency f_{IN} by a factor P. The PLL core multiplies the input reference by a factor N to produce the DCO frequency f_{DCO}:

$$f_{\mathrm{REF}} = f_{\mathrm{IN}} \div P,\ f_{\mathrm{DCO}} = f_{\mathrm{REF}} \times N \tag{56}$$

The recommendation is to use f_{DCO} at 800 MHz and P to 1, that is, means PERPLLCONx.PDIV = 0.

In the output stage, the DCO frequency is divided down by a factor Kx (with x = 2,3,4) combined with KxPREDIV. KxPREDIV is an additional divider circuit that is capable of generating selected non-integer division ratios of factor 1.0, 1.1, 1.2, 1.4, 1.6, 1.7, and 2.0.

The output frequencies $f_{PLL1\text{-}3}$ is given by:

37 Refer to the datasheet for valid PERPLL frequencies to determine appropriate N, P and K2,3,4 and pre-divider settings.

$$
\begin{aligned}
f_{PLL1} &= f_{DCO} \div (K2 \times 1.0) \rightarrow K2PRE = 0 \\
f_{PLL1} &= f_{DCO} \div (K2 \times 2.0) \rightarrow K2PRE = 1 \\
f_{PLL1} &= f_{DCO} \div (K2 \times 1.2) \rightarrow K2PRE = 2 \\
f_{PLL1} &= f_{DCO} \div (K2 \times 1.6) \rightarrow K2PRE = 3
\end{aligned}
$$

$$
\begin{aligned}
f_{PLL2} &= f_{DCO} \div (K3 \times 1.0) \rightarrow K3PRE = 0 \\
f_{PLL2} &= f_{DCO} \div (K3 \times 1.1) \rightarrow K3PRE = 1 \\
f_{PLL2} &= f_{DCO} \div (K3 \times 1.2) \rightarrow K3PRE = 2 \\
f_{PLL2} &= f_{DCO} \div (K3 \times 1.4) \rightarrow K3PRE = 4 \\
f_{PLL2} &= f_{DCO} \div (K3 \times 1.6) \rightarrow K3PRE = 6 \\
f_{PLL2} &= f_{DCO} \div (K3 \times 1.7) \rightarrow K3PRE = 7 \\
f_{PLL2} &= f_{DCO} \div (K3 \times 2.0) \rightarrow K3PRE = 10
\end{aligned}
\tag{57}
$$

$$
\begin{aligned}
f_{PLL3} &= f_{DCO} \div (K4 \times 1.0) \rightarrow K4PRE = 0 \\
f_{PLL3} &= f_{DCO} \div (K4 \times 2.0) \rightarrow K4PRE = 1 \\
f_{PLL3} &= f_{DCO} \div (K4 \times 1.2) \rightarrow K4PRE = 2 \\
f_{PLL3} &= f_{DCO} \div (K4 \times 1.6) \rightarrow K4PRE = 3
\end{aligned}
$$

In addition the PERPLL provides a special clock output f_{HSCT}

$$f_{HSCT} = f_{DCO} \tag{58}$$

PERPLL lock detection

The PERPLL has a lock detection feature, that supervises the DCO part of the PERPLL to differentiate between stable and unstable DCO circuit behavior. The lock detector marks the PLL as locked if its two input frequencies of the lock detector f_{REF} and f_{DIV} (=f_{DCO}÷N) have a phase difference smaller than ½ reference cycle for more than 224 reference cycles.

However, the unlock condition is reported immediately if the phase error is bigger than the above given limit. Smaller phase errors can be handled by the PLL without getting unstable and are not marked as "loss of lock".

The PERPLL may become unlocked, due to a break of the crystal or ceramic resonator or the external clock line. In such a case, an SMU alarm event is generated and the PLL switches to an open loop mode and keeps on free running at the last locked frequency.

The peripheral operation frequencies are controlled by the values of the five dividers: P, N, and K2, K3 and K4. A modification of the two dividers P and N has a direct influence to the DCO frequency and could lead to a loss of the lock status. However, a modification of the K2, K3 and K4 dividers has no impact on the lock status, but still changes the PERPLL output frequencies f_{PLL1}, f_{PLL2} and f_{PLL3} directly.

The lock status can be observed in bit-field PERPLLSTAT.PLLLOCK and the lock detection can be reset by setting bit-field PERPLLCON0.RESLD = 1_B. Applying the reset will close the loop and eventually enable it to lock again.

Attention**: **Once unlocked, the PLL will not make an attempt to lock again until issuing a lock detection reset or a reprogramming of the P or N dividers.

Frequency modulation

The PERPLL output frequencies $f_{PLL1/2/3}$ can additionally be modified by a low-frequency modulation to reduce EMI. A random sequence is added to the DCO control word resulting in a randomly modulated f_{DCO}. The rate of the modulation is defined by f_{REF}.

To configure the modulation follow these steps in the given order:

Turn on or off modulation for PERPLL

```
Wait until CLOCK_CCUSTAT.LCK == 0              // Wait for any ongoing transfer with a timeout
of 100 µs
CLOCK_PERPLLCON0.MODEN = 0                     // Deactivate Frequency Modulation
Wait until CLOCK_CCUSTAT.LCK == 0              // Wait for any ongoing transfer with a timeout
of 100 µs
CLOCK_PERPLLCON2.MODCFG = (0x7400 | MG)        // Program Modulation Gain (MG)
if (MG)
     Wait until CLOCK_CCUSTAT.LCK == 0         // Wait for any ongoing transfer with a timeout
of 100 µs
     CLOCK_PMODCON0 = 0x80000000               // CAL_EN = 1
     Wait(100ns)                               // Wait to ensure toggling
     Wait until CLOCK_CCUSTAT.LCK == 0         // Wait for any ongoing transfer with a timeout
of 100 µs
     CLOCK_PMODCON0 = 0x00000000               // CAL_EN = 0
     Wait until CLOCK_CCUSTAT.LCK == 0         // Wait for any ongoing transfer with a timeout
of 100 µs
     CLOCK_PERPLLCON0.MODEN = 1                // Enable Modulation
     Wait until CLOCK_PERPLLSTAT.MODRUN == 1   // Wait until modulation is running with a
timeout of 100 µs
end
```

***Note**: To turn off modulation, simply set MG=0 in the above sequence*

***Notes**:*
To change the modulation gain:

1. *Turn off modulation*
2. *Turn on modulation with different gain*

The modulation is enabled using bit-field PERPLLCON0.MODEN. The modulation itself alters the DCO frequency randomly within the range of the configured relative modulation amplitude (MA%). The modulation is controlled through bit-field PERPLLCON2.MODFCG as follows:

- MODCFG[9:0] = modulation gain

$$\text{modulation gain} = \frac{64}{3.6\text{MHz}} \times f_{\text{DCO}} \times \text{MA}$$
$$\text{Example: MA} = 2\,\% \text{ at} f_{\text{DCO}} = 800\,\text{MHz} \Rightarrow \text{modulation gain} = 0x11C \quad (59)$$

Configuration sequence for the K-dividers for configuration and re-configuration of PLLs

Special care should be taken during configuration and re-configuration of PLLs with respect to programming the K-dividers to avoid wrong K-divider or dead clock at one of the PLL output paths which can be recovered only through a system reset.

Follow the below configuration sequence for the K-dividers for the configuration or re-configuration of any PLL:

PLL configuration phase

1. PLL off
2. Check for CCU lock: CLOCK_CCUSTAT.B.LCK == 0 with a timeout of at least 100 µs
3. Configure K-dividers in xPLLCON1 register
    a. Set KxPREDIV = div by 2 (xPLLCON1.KxPREDIV = div by 2)
    b. Check for CCU lock: CLOCK_CCUSTAT.B.LCK == 0 with a timeout of at least 100 µs
    c. Set KxDIV = 0x01 (xPLLCON1.KxDIV = 0x01)
    d. Check for CCU lock: CLOCK_CCUSTAT.B.LCK == 0 with a timeout of at least 100 µs
    e. Set other PLL parameters as described in the user manual
    f. Check for CCU lock: CLOCK_CCUSTAT.B.LCK == 0 with a timeout of at least 100 µs
4. PLL on (power PLL)
5. Check for CCU lock: CLOCK_CCUSTAT.B.LCK == 0 with a timeout of at least 100 µs
6. Check if PLL is powered: xPLLSTAT.PWRSTAT == 1 with a timeout of at least 100 µs
7. Wait for 1ms to avoid jitter
8. Check for PLL lock: xPLLSTAT.PLLLOCK == 1 with a timeout of at least 100 µs
9. Configure KxPREDIV (if necessary)
    a. Set KxPREDIV to final value
10. Wait for 10 µs and check for CCU lock: CLOCK_CCUSTAT.B.LCK == 0 with a timeout of at least 100 µs
11. Configure KxDIV
    a. Set KxDIV to final value
    b. Check for CCU lock: CLOCK_CCUSTAT.B.LCK == 0 with a timeout of at least 100 µs

Changing the KxDIV value during run time (KxPREDIV is not allowed to change during run time)

1. Check for PLL lock: xPLLSTAT.PLLLOCK == 1 with a timeout of at least 100 µs
2. Check for CCU lock: xLOCK_CCUSTAT.B.LCK == 0 with a timeout of at least 100 µs
3. Configure K-dividers in xPLLCON1 register
    a. KxDIV = 0x01 (xPLLCON1.KxDIV = 0x01)
4. Wait for 10 µs and check for CCU lock: CLOCK_CCUSTAT.B.LCK == 0 with a timeout of at least 100 µs
5. Configure KxDIV
    a. Set KxDIV to final value
    b. Check for CCU lock: CLOCK_CCUSTAT.B.LCK == 0 with a timeout of at least 100 µs

Related information

Crystal oscillator on page 5531

32.3.5 Distribution

Using the first two parts of the clock system ("clock source" and "clock generation"), all root clocks the system relies on, are defined. These root clocks need to be individually adapted in frequency (divided) and distributed to all functional blocks of the MCU.

For clock distribution, the system is split into several sub-clock domains where the clock speed could be configured individually. There are also limitations for each sub-clock domain derived out of the internal interfaces. Each sub-clock domain defines a logical unit from a clocking perspective point of view.

The clock distribution is done through the clock control unit (*CCU*). The CCU receives the clocks that are created by the two PLLs (f_{PLL0},f_{PLLPPU}[38] and f_{PLL1}, f_{PLL2}, f_{PLL3}), the back-up clock f_{Back}, the ramp clocks f_{Ramp}, $f_{RAMPHALF}$[38], and the crystal oscillator output f_{XTAL}. These clocks are either forwarded directly or divided in order to supply the sub-clock domains.

32.3.5.1 Clock control unit

The clock control unit (*CCU*) finally derives all system and peripheral clocks needed by the functional blocks of the MCU based on the root clocks coming from the PLLs, EVR, ramp and crystal oscillators.

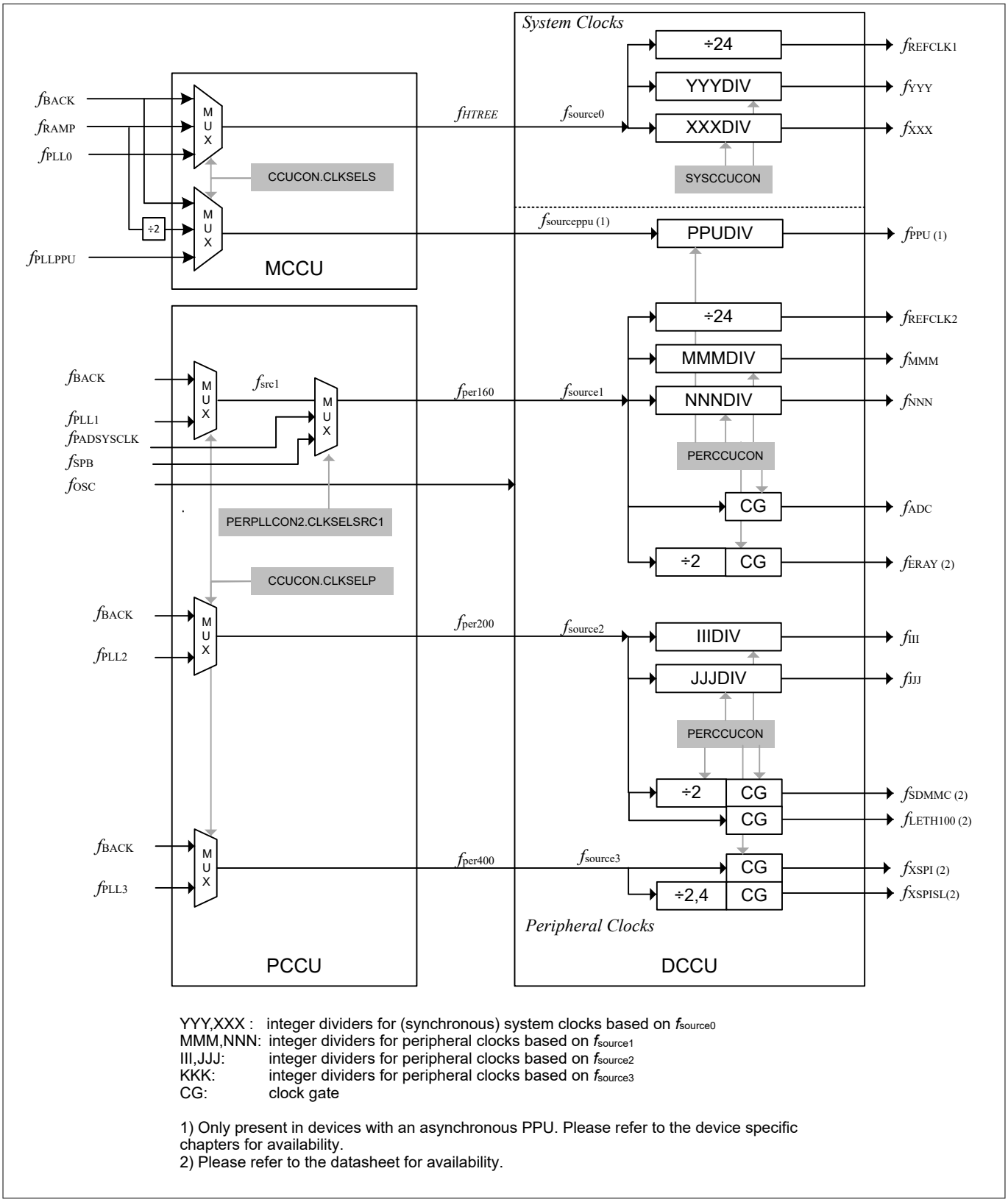
The CCU is logically composed of three units as shown in the following figure:

1. Master CCU (*MCCU*): Generates the root system clock $f_{source0}$ and the root PPU clock $f_{sourceppu}$[39] by selecting either the EVR oscillator f_{BACK}, the ramp oscillator f_{RAMP}, $f_{RAMPHALF}$[39] or the SYSPLL f_{PLL0} as root clock for the system clock tree $f_{HTREE}$and for the PPU clock generation[39]
2. Peripheral CCU (*PCCU*): Generates the root peripheral clocks $f_{source1\text{-}3}$ by selecting either the EVR oscillator f_{BACK} or the PERPLL $f_{PLL1\text{-}3}$ as root clocks
3. Distribution CCU (*DCCU*): Generates all system and peripheral clocks by using (dividing, switching and gating) the output clocks of the MCCU and PCCU. For selected peripheral clocks, the crystal clock f_{XTAL} is used as option

The following figure shows the general structure of the clock control unit.

In devices with a synchronous PPU, there is no separate PPU clock output because the PPU runs synchronous to the rest of the system resources and its clock is generated from the PLL0 output. In devices with an asynchronous PPU, the PPU clock is generated by the second SYSPLL output and is handled like another asynchronous peripheral clock.

38 Only present in devices with an asynchronous PPU. Please refer to the device specific chapters for availability
39 Only present in devices with an asynchronous PPU. Please refer to the device specific chapters for availability



YYY,XXX : integer dividers for (synchronous) system clocks based on $f_{source0}$
MMM,NNN: integer dividers for peripheral clocks based on $f_{source1}$
III,JJJ: integer dividers for peripheral clocks based on $f_{source2}$
KKK: integer dividers for peripheral clocks based on $f_{source3}$
CG: clock gate

1) Only present in devices with an asynchronous PPU. Please refer to the device specific chapters for availability.
2) Please refer to the datasheet for availability.

Figure 747 Clock control unit overview

CCU registers

CCU general:

- Global clock control register (CCUCON): source clock selection control
- Global clock status register (CCUSTAT): clock source selection status

System clocks:

- System CCU clock control register 0 (SYSCCUCON0): Divider and enable setting for system clocks
- System CCU clock control register 1 (SYSCCUCON1): Divider and enable settings for system clocks

Peripheral clocks:

- Peripheral CCU clock control register 0 (PERCCUCON0): Divider and enable settings for peripheral clocks
- Peripheral CCU clock control register 1 (PERCCUCON1): Divider and enable settings for peripheral clocks

Source clock selection and switching

The source clocks for the system, PPU[39] and peripheral clock distribution ($f_{source0}$, $f_{sourceppu}$[39], $f_{source1-3}$) can be selected from the PLL output clocks (f_{PLL0}, f_{PLLPPU}[39], $f_{PLL1/2/3}$), the EVR oscillator clock (f_{BACK}) and the ramp oscillator clock (f_{RAMP}, $f_{RAMPHALF}$[39]). This is done through the CCUCON.

The selection of the input clock is done inside the MCCU with bit-field CCUCON.CLKSELS for the system clocks and inside the PCCU with bit-field CCUCON.CLKSELP for the peripheral clocks. Software selects the source by writing the related value and hardware switches the associated clock multiplexers accordingly.

The actual selection status can be read by software from the CCUSTAT. The bit-fields CCUSTAT.CLKSELS and CCUSTAT.CLKSELP report the currently selected clock sources.

The separation of control and status bit-fields is done to reflect the situation where hardware overrules the actual software setting. This may be the case during clock emergency switching or a frequency ramp triggered by an application reset. The following table lists the settings:

Table 1278 Clock source selection control and status

	CCUCON.CLKSELS	CCUCON.CLKSELP
System clocks sourced by PLL0	00_B	--
System clocks sourced by EVR clock	01_B (default)	--
System clocks sourced by ramp clock	10_B	--
PPU clock [39]sourced by PLLPPU	00_B	--
PPU clock[39] sourced by EVR clock	01_B (default)	--
PPU clock[39] sourced by ramp clock divided	10_B	--
Peripheral clocks sourced by PLL1-3	--	0_B
Peripheral clocks sourced by EVR clock	--	1_B (default)

As shown in the above table, the clock source selection setting for system and peripheral clocks are independent from each other. This is also valid for hardware initiated overruling during emergency switching and application reset execution..

***Note**: The CLKSELS and CLKSELP values in the CCUSTAT register reflect the actual used setting of the clock multiplexers inside the MCCU and PCCU. Under normal operation scenarios this equals the setting in the control register. However, in case of clock emergency switching, status and control values will differ.*

Related information

Start-up and shut-down on page 5560

39 Only present in devices with an asynchronous PPU. Please refer to the device specific chapters for availability

32.3.5.1.1 Module clocks

The clocking system offers many options and a high amount of flexibility. For different functional blocks, options are provided to influence the application execution or performance through the clock control.

Module clock overview

Here we give a detailed overview of the different clocks used for the functional blocks of the MCU:

Table 1279 Central processing units, bus interconnects and timers

Name	Description	Domain
f_{SRI}	Defines the operating performance of the system resource interconnect fabric (*SRI*), and therefore the data exchange rate between all connected master- and slave-agents. The CPUs and DMAs also run on f_{SRI}	System
f_{SPB} [1)]	Defines the operating performance of the system peripheral bus (*SPB*) interconnect, and therefore the data exchange rate between all connected master- and slave-agent	System
f_{CPB} [2)]	Defines the operating performance of the converter peripheral bus (*CPB*) interconnect, and therefore the data exchange rate between all connected master- and slave-agents	System
f_{STM}	Defines the basic frequency for the system timers (*STM*) independent of the rest of the system	System
f_{RCB} [2)]	Defines the operating performance of the radar configuration bus (*RCB*) interconnect, and therefore the data exchange rate between all connected master- and slave-agents	System
f_{GTM} [2)]	Defines the basic frequency for the generic timer (*GTM*) module independent of the rest of the system	System
f_{eGTM} [2)]	Defines the basic frequency for the enhanced generic time (*eGTM*) module independent of the rest of the system	System
$f_{eGTMhalf}$ [2)]	Defines the divided by half frequency for the enhanced generic time (*eGTM*) module. This frequency is always half of f_{eGTM}	System

1) On some devices the COMPB also uses this clock. Please refer to the device specific chapters for further details.
2) Please refer to the datasheet for availability

Table 1280 Parallel processing unit

Name	Description	Domain
f_{PPU} [2)]	Defines the operating frequency of the parallel processing unit	System

Table 1281 Ethernet modules

Name	Description	Domain
f_{GETH} [2)]	Defines the operating frequency of the Gigabit Ethernet subsystem	System
f_{LETH}	Defines the operating frequency of the Light Ethernet mac modules	System

Table 1282 CAN modules

Name	Description	Domain
f_{MCANH} [2)]	Defines the host frequency of the MCAN subsystem	System
f_{CANXLH}	Defines the host frequency of the CANXL subsystem	System

Table 1283 Flash (DMU) controls

Name	Description	Domain
f_{FSI2}	Defines the execution speed of the flash standard interface (*FSI*) for program flash reading operations	System
f_{FSI}	Defines the execution speed for all other FSI operations	System

Table 1284 Cyber security module

Name	Description	Domain
f_{SRICS}	The components of the cyber security real-time module (CSRM) run either on f_{SRICS} or f_{SPB}. f_{SRICS} is configured to be equal to f_{SRI} when the control bit-field SYSCCUCON0.SRICSDIV = 0. If the bit-field is set to 1, f_{SRICS} is half the frequency of f_{SRI}	System

Table 1285 Trace system

Name	Description	Domain
f_{TPB}	Because tracing should be non-intrusive to the application system, a separate clock f_{TPB} for dedicated trace resources is available. Note that f_{TPB} needs to be at least two times higher than f_{SPB}	System
$f_{REFCLK1}$	Defines a fixed timebase for the trace system. This allows the trace system (MCDS) to generate timestamps independent of the selected SRI and SPB clock speeds	System
$f_{REFCLK2}$	Defines a fixed timebase for the trace system. This allows the MCDS to generate timestamps independent of the selected SRI and SPB clock speeds	Peripheral

Table 1286 Miscellaneous clocks

name	Description	Domain
f_{MT0} [2)]	This is the macrotick 0 signal from the flexray (*ERAY*) module which can be routed to the external clock output	System

Table 1287 Peripheral clocks of communication modules

name	Description	Domain
f_{MCAN} [2)]	Defines the basic frequency for the external communication protocol of the MCAN modules independent of the rest of the system	Peripheral
f_{CANXL}	Defines the basic frequency for the external communication protocol of the CANXL modules independent of the rest of the system	Peripheral
f_{MSC} [2)]	Defines the basic frequency for the micro second channel (*MSC*) modules independent of the rest of the system	Peripheral
f_{QSPI}	Defines the basic frequency for the queued synchronous peripheral interface (*QSPI*) modules independent of the rest of the system	Peripheral
$f_{ASCLINF}$	Defines the basic frequencies for the fast asynchronous/synchronous interface (*ASCLINF*) module independent of the rest of the system	Peripheral
$f_{ASCLINS}$	Defines the basic frequencies for the slow asynchronous/synchronous interface (*ASCLINS*) module independent of the rest of the system	Peripheral
f_{I2C}	Defines the basic frequency for the inter-integrated circuit (*I2C*) modules independent of the rest of the system	Peripheral
f_{ERAY} [2)]	Defines the basic frequency for the flexray (*ERAY*) modules independent of the rest of the system	Peripheral

(table continues...)

Table 1287 (continued) Peripheral clocks of communication modules

name	Description	Domain
f_{HSCT} [2]	Defines the basic frequency for the high speed communication tunnel (*HSCT*) module independent of the rest of the system	Peripheral
f_{xSPI} [2]	Defines the basic frequency for the *xSPI* Interface independent of the rest of the system	Peripheral
f_{xSPISL} [1]	Defines the slow basic frequency for the *xSPI* Interface independent of the rest of the system	Peripheral
f_{SDMMC} [2]	Defines the basic frequency for the secure digital multimedia card (*SDMMC*) module independent of the rest of the system	Peripheral
$f_{LETH100}$	Defines the reference clock frequency for the *LETH* modules independent of the rest of the system	Peripheral

1) Please refer to the device specific chapters for availability

Table 1288 Converters

name	Description	Domain
f_{ADC}	Defines the basic frequency for the analog digital converters (*ADC*) independent of the rest of the system	Peripheral

Module clock sources and dividers

The following table lists for all generated clocks what clock source is available and what divider is used to scale the frequency. A divider value of 0 switches off the clocks. In the below listed tables all clock dividers and clock selectors can be programmed through the following registers:

- CCUCON for clock source settings
- SYSCCUCON0 and SYSCCUCON1 for system clock dividers
- PERCCUCON0 and PERCCUCON1 and for peripheral clock dividers and selectors

Table 1289 CCU system clock options

CCU clock output	Clock source				
	f_{BACK} (system reset) [1]	f_{PLL0} [2]	f_{BACK} [3]	f_{RAMP} [4]	Off [5]
f_{SRI}	÷ SRIDIV=1	÷ SRIDIV= 1,2,3,4,5,6,8, 10,12,15 120[6]	÷ SRIDIV= 1,2,3,4,5,6,8, 10,12,15 120[6]	÷ SRIDIV= 1,2,3,4,5,6,8, 10,12,15 120[6]	not possible
f_{SPB} [1]	÷ SPBDIV=2	÷ SPBDIV= 1,2,3,4,5,6,8, 10,12,15 120[6]	÷ SPBDIV= 1,2,3,4,5,6,8, 10,12,15 120[6]	÷ SPBDIV= 1,2,3,4,5,6,8, 10,12,15 120[6]	not possible
f_{CPB} [2]	÷ CPBDIV=2	÷ CPBDIV= 1,2,3,4,5,6,8, 10,12,15 120[6]	÷ CPBDIV= 1,2,3,4,5,6,8, 10,12,15 120[6]	÷ CPBDIV= 1,2,3,4,5,6,8, 10,12,15 120[6]	÷ CPBDIV=0 [7]

(table continues...)

Table 1289 (continued) CCU system clock options

CCU clock output	Clock source				
	f_{BACK} (system reset) 1)	f_{PLL0} 2)	f_{BACK} 3)	f_{RAMP} 4)	Off 5)
f_{TPB}	÷ TPBDIV=1	÷ TPBDIV= 1,2,3,4,5,6,8, 10,12,15 120 6)	÷ TPBDIV= 1,2,3,4,5,6,8, 10,12,15 120 6)	÷ TPBDIV= 1,2,3,4,5,6,8, 10,12,15 120 6)	÷ TPBDIV=0
f_{RCB} 2)	÷ RCBDIV=1	÷ RCBDIV= 1,2,3,4,5,6,8, 10,12,15 120 6)	÷ RCBDIV= 1,2,3,4,5,6,8, 10,12,15 120 6)	÷ RCBDIV= 1,2,3,4,5,6,8, 10,12,15 120 6)	÷ RCBDIV=0
f_{FSI}	÷ FSIDIV=1 8)	÷ FSIDIV= 1,2,3,4,5,6,8, 10,12,15 120 6) 9)	÷ FSIDIV= 1,2,3,4,5,6,8, 10,12,15 120 6) 9)	÷ FSIDIV= 1,2,3,4,5,6,8, 10,12,15 120 6) 9)	
f_{FSI2}	÷ FSI2DIV=1 8)	÷ SRIDIV FSI2DIV=1	÷ SRIDIV FSI2DIV=1	÷ SRIDIV FSI2DIV=1	
f_{SRICS} 10)	÷ SRIDIV=1	÷ SRIDIV= 1,2,3,4,5,6,8, 10,12,15 120 6)	÷ SRIDIV= 1,2,3,4,5,6,8, 10,12,15 120 6)	÷ SRIDIV= 1,2,3,4,5,6,8, 10,12,15 120 6)	not possible
f_{SRICS} 11)	÷ SRIDIV=1	÷ SRIDIV÷ 2 ÷ 120 6)	÷ SRIDIV÷ 2 ÷ 120 6)	÷ SRIDIV÷ 2 ÷ 120 6)	not possible
f_{GTM} 2)	÷ GTMDIV=2	÷ GTMDIV= 1,2,3,4,5,6,8, 10,12,15 120 6)	÷ GTMDIV= 1,2,3,4,5,6,8, 10,12,15 120 6)	÷ GTMDIV= 1,2,3,4,5,6,8, 10,12,15 120 6)	÷ GTMDIV=0
f_{eGTM} 2)	÷ eGTMDIV=0	÷ eGTMDIV= 1,2,3,4,5,6,8, 10,12,15 120 6)	÷ eGTMDIV= 1,2,3,4,5,6,8, 10,12,15 120 6)	÷ eGTMDIV= 1,2,3,4,5,6,8, 10,12,15 120 6)	÷ eGTMDIV=0
f_{STM}	÷ STMDIV=2	÷ STMDIV= 1,2,3,4,5,6,8, 10,12,15 120 6)	÷ STMDIV= 1,2,3,4,5,6,8, 10,12,15 120 6)	÷ STMDIV= 1,2,3,4,5,6,8, 10,12,15 120 6)	÷ STMDIV=0
f_{GETH} 2)	÷ GETHDIV=4	÷ GETHDIV= 1,2,3,4 120 6)	÷ GETHDIV= 1,2,3,4 120 6)	÷ GETHDIV= 1,2,3,4 120 6)	÷ GETHDIV=0

(table continues...)

Table 1289 (continued) CCU system clock options

CCU clock output	Clock source				
	f_{BACK} (system reset) 1)	f_{PLL0} 2)	f_{BACK} 3)	f_{RAMP} 4)	Off 5)
f_{LETH}	÷ LETHDIV=0	÷ LETHDIV= 1,2,3,4 120 6)	÷ LETHDIV= 1,2,3,4 120 6)	÷ LETHDIV= 1,2,3,4 120 6)	÷ LETHDIV=0
f_{MCANH} 2)	÷ MCANHDIV=2	÷ MCANHDIV= 1,2,3,4,5,6,8, 10,12,15	÷ MCANHDIV= 1,2,3,4,5,6,8, 10,12,15	÷ MCANHDIV= 1,2,3,4,5,6,8, 10,12,15	÷ MCANHDIV=0
f_{CANXLH}	÷ CANXLHDIV=2	÷ CANXLHDIV= 1,2,3,4,5,6,8, 10,12,15	÷ CANXLHDIV= 1,2,3,4,5,6,8, 10,12,15	÷ CANXLHDIV= 1,2,3,4,5,6,8, 10,12,15	÷ CANXLHDIV=0
$f_{REFCLK1}$	÷24	÷24	÷24	÷24	not possible

1) Set by system reset, CLKSELS = 01_B
2) Set by SW at anytime, CLKSELS = 00_B
3) Set by HW due to clock emergency switch, set by SW at anytime, CLKSELS = 01_B
4) Set by SW at anytime, set by HW during application reset or clock emergency switch execution, CLKSELS = 10_B
5) Set by SW through respective divider or clock selection setting
6) LPDIV=1, not safety covered
7) Configuring CPBDIV=0 can lead to a bus error.
8) The divider value may be different for certain devices. Please refer to the device specific chapters for further details.
9) Shall be set to match SRIDIV
10) SYSCCUCON0.SRICSDIV = 0
11) SYSCCUCON0.SRICSDIV = 1

Table 1290 PPU clock options

CCU clock output	Clock source				
	f_{BACK} (system reset) 1)	f_{PPUPLL} 2)	f_{BACK} 3)	$f_{RAMP\ DIVIDED}$ 4)	Off 5)
f_{PPU} 2)	÷ PPUDIV=0	÷ PPUDIV= 1,2,3,4,5,6,8, 10,12,15	÷ PPUDIV= 1,2,3,4,5,6,8, 10,12,15	÷ PPUDIV= 1,2,3,4,5,6,8, 10,12,15	÷ PPUDIV=0

1) Set by system reset, CLKSELS = 01_B
2) Set by SW at anytime, CLKSELS = 00_B
3) Set by HW due to clock emergency switch, set by SW at anytime, CLKSELS = 01_B
4) Set by SW at anytime, set by HW during application reset or clock emergency switch execution, CLKSELS = 10_B
5) Set by SW through respective divider or clock selection setting

Table 1291 CCU communication peripheral clock options

CCU clock output	Clock source						
	f_{BACK} (System reset) 1)	f_{PLL1} 2)	f_{PLL2} 2)	f_{PLL3} 2)	f_{BACK} 3)	f_{OSC}	Off 4)
f_{ERAY} 2)	÷2	÷2	–	-	÷2	–	CLKSELSRC1=0 or ERAYPERON = 0
f_{MSC} 2)	÷ MSCDIV=0 $CLKSEL_{MSC}$ =1	÷ MSCDIV= 1,2,3,4,5,6,8, 10,12,15 $CLKSEL_{MSC}$=1	÷ MSCDIV= 1,2,3,4,5,6,8, 10,12,15 $CLKSEL_{MSC}$=2	-	÷ MSCDIV= 1,2,3,4,5,6,8, 10,12,15 $CLKSEL_{MSC}$=1,2	–	CLKSELSRC1=0 or $CLKSEL_{MSC}$=0 or MSCDIV=0
f_{MCAN} 2)	÷ MCANDIV=5 $CLKSEL_{MCAN}$ =1	÷ MCANDIV= 1,2,3,4,5,6,8, 10,12,15 $CLKSEL_{MCAN}$=1	–	-	÷ MCANDIV= 1,2,3,4,5,6,8, 10,12,15 $CLKSEL_{MCAN}$=1	÷1 $CLKSEL_{MCAN}$ =2	CLKSELSRC1=0 or $CLKSEL_{MCAN}$=0 or MCANDIV=0
f_{CANXL}	÷ CANXLDIV=5 $CLKSEL_{CANXL}$ =1	÷ CANXLDIV= 1,2,3,4,5,6,8, 10,12,15 $CLKSEL_{CANXL}$=1	–	-	÷ CANXLDIV= 1,2,3,4,5,6,8, 10,12,15 $CLKSEL_{CANXL}$=1	÷1 $CLKSEL_{CANXL}$ =2	CLKSELSRC1=0 or $CLKSEL_{CANXL}$=0 or CANXLDIV=0
$f_{ASCLINF}$	÷ ASCLINFDIV =1	–	÷ ASCLINFDIV= 1,2,3,4,5,6,8, 10,12,15	-	÷ ASCLINFDIV= 1,2,3,4,5,6,8, 10,12,15	–	ASCLINFDIV =0
$f_{ASCLINS}$	÷ ASCLINSDIV =5 $CLKSEL_{ASCLINS}$ =1	÷ ASCLINSDIV= 1,2,3,4,5,6,8, 10,12,15 $CLKSEL_{ASCLINS}$= 1	–	-	÷ ASCLINSDIV= 1,2,3,4,5,6,8, 10,12,15 $CLKSEL_{ASCLINS}$= 1	÷1 $CLKSEL_{ASCLINS}$ =2	CLKSELSRC1=0 or $CLKSEL_{ASCLINS}$ =0 or ASCLINSDIV=0
f_{QSPI}	÷ QSPIDIV=0 $CLKSEL_{QSPI}$=1	÷ QSPIDIV= 1,2,3,4,5,6,8, 10,12,15 $CLKSEL_{QSPI}$=1	÷ QSPIDIV= 1,2,3,4,5,6,8, 10,12,15 $CLKSEL_{QSPI}$=2	-	÷ QSPIDIV= 1,2,3,4,5,6,8, 10,12,15 $CLKSEL_{QSPI}$=1,2	–	CLKSELSRC1=0 or $CLKSEL_{QSPI}$=0 or QSPIDIV=0
f_{I2C}	÷ I2CDIV=3	–	÷ I2CDIV= 1,2,3,4,5,6,8, 10,12,15	-	÷ I2CDIV= 1,2,3,4,5,6,8, 10,12,15	–	I2CDIV=0
f_{SDMMC} 2)	SDMMCPERON =1	–	÷2	-	÷2 SDMMCPERON =1	–	SDMMCPERON =0

(table continues...)

Table 1291 (continued) CCU communication peripheral clock options

CCU clock output	Clock source						
	f_{BACK} (System reset) 1)	f_{PLL1} 2)	f_{PLL2} 2)	f_{PLL3} 2)	f_{BACK} 3)	f_{OSC}	Off 4)
f_{XSPI} 2)	XSPIPERON=0	–	–	÷1	÷1 XSPIPERON=1	–	XSPIPERON =0 5)
f_{xSPISL} 2)	÷ XSPISLDIV=0	–	-	÷ XSPISLDIV= 1,2,4	÷ XSPISLDIV= 1,2,4	–	XSPISLDIV=0
$f_{REFCLK2}$	÷24	÷24	–	–	÷24	–	CLKSELSRC1= 0
$f_{LETH100}$	LETH100PERON =0	-	÷1	-	÷1	–	LETH100PERON =0

1) Set by HW due to system reset, CLKSELP = 1_B
2) Set by SW at anytime, CLKSELP = 0_B
3) Set by HW due to system reset or clock emergency switch, set by SW at anytime, CLKSELP = 1_B
4) Set by SW through respective divider or clock selection setting
5) Set by HW due to system reset

Table 1292 CCU converter and HSCT speed clock options

CCU clock output	Clock source						
	f_{BACK} (System Reset) 1)	f_{PLL1} 2) 3)	f_{SPB} 4) 3)	$f_{PADSYSCLK}$ 5) 3)	f_{BACK} 6)	f_{DCO} (PERPLL) 7)	Off 8)
f_{ADC}	CLKSELSRC1 =1 ADCPERON= 1	÷1 CLKSELSRC1 =1 ADCPERON= 1	÷1 CLKSELSRC1 =2 ADCPERON= 1	÷1 CLKSELSRC1 =3 ADCPERON= 1	÷1	–	CLKSELSRC1 =0 or ADCPERON= 0
f_{HSCT} 2)	–	–	–	–	–	÷1 HSCTPERON =1	HSCTPERON =0 9)

1) Set by HW due to system reset, CLKSELP = 1_B
2) Set by SW at anytime, CLKSELP = $0_B$and CLKSELSRC1 = 1
3) Refer to the programming sequence as described in PERPLLCON2
4) Set by SW at anytime, CLKSELSRC1 = 2
5) Set by SW at anytime, CLKSELSRC1 = 3
6) Set by HW due to system reset or clock emergency switch, set by SW at anytime, CLKSELP = 1_B
7) The 5-phase 320 MHz HSCT clock is generated within the HSCT Phy. The DCO frequency of the PERPLL must be set to 800 MHz to achieve 320 MBaud data rate.
8) Set by SW through respective divider or clock selection setting

9) Set by HW due to system reset

32.3.5.1.2 Clock divider limitations

For the clock dividers which control the different sub-clock domains or modules, the allowed values are limited and the following ratios have to be observed. The ratios are defined in the following way:

clock A = f_{XXX}; clock B = f_{YYY} the allowed ratio is 1 : n where n has a defined range of values.

Clock A is always faster or equal to clock B in this definition with f_{XXX} [MHz] = n × f_{YYY} [MHz].

For synchronous clocks, n must be an integer value.

In addition, the divider values are limited by the resulting minimum or maximum frequencies. These frequencies are defined in the datasheet.

Table 1293 Divider ratio restrictions

Clock A	Clock B	Allowed values	Recommended value for 400 MHz use-case[1]	Recommended value for 500 MHz use-case[2]
f_{SRI}	f_{SPB}[3]	n ≥ 1[4]	n = 4	n = 5
f_{SRI}	f_{CPB}[6]	n ≥ 1	n = 4	n = 5
f_{SRI}	f_{TPB}	n = 1, 2[5]	n = 2	n = 2
f_{TPB}	f_{GTM}[6]	n ≥ 1	n = 1	NA
f_{TPB}	$f_{eGTMhalf}$[6]	n ≥ 1[7]	n = 1	n = 1
f_{TPB}	f_{SPB}[3]	n ≥ 2	n = 2	n = 2.5[11]
f_{SRI}	f_{STM}	n ≥ 1	n = 1	n = 1
f_{SRIcs}	f_{STM}	n ≥ 1	n = 1	n = 1
f_{SRI}	f_{FSI}	n = 1, 2, 3,4,5[8] [9]	n = 4	n = 5
f_{SRI}	f_{FSI2}	n = 1	n = 1	n = 1
f_{FSI2}	f_{FSI}	n = 1, 2, 3,4, 5 [10]	n = 4	n = 5
f_{FSI}	f_{SPB}[3]	n ≥ 1	n = 2	n = 2
f_{GTM}[6]	f_{SPB}[3]	n ≥ 1	n = 2	NA
f_{MCANH}[6]	f_{MCAN}[6]	n ≥ 1.0[11]	n = 1.25[12] [11]	n = 1.56/1.25 [13] [11]
f_{CANXLH}	f_{CANXL}	n ≥ 1.0[11]	n = 1.25[14] [11]	n = 1.56/1.25 [15] [11]
f_{SRI}	f_{RCB}[6]	n = 2	n = 2	NA
f_{SRICS}	f_{FSI}	n = 1, 2, 3,4,5[16] [17]	n = 4	n = 5

1) The recommended defaults apply for a system source clock frequency of $f_{source0}$ = 400 MHz only.
2) The recommended defaults apply for a system source clock frequency of $f_{source0}$ = 500 MHz only.
3) On some devices the COMPB also uses this clock. Please refer to the device specific chapters for further details.
4) Not all values of n are available for all derivatives. Please refer to the device specific chapters for restrictions on the values of n
5) n=1 shall only be applied if $f_{SRI} \leq f_{TPBmax}$
6) Please refer to the datasheet for availability
7) To avoid timing violations at this interface, the EGTMDIV must not be set to an odd value, but only to 1 or an even value.
8) Note that f_{SRI}/f_{FSI}=1 is not supported by all devices. Please refer to the device specific chapters for further restrictions.
9) Do not apply values that would lower f_{FSI} below its reset value which is 100 MHz. This is needed to guarantee a graceful shutdown operation.
10) Note that f_{FSI2}/f_{FSI}=1 is not supported by all devices. Please refer to the device specific chapters for further restrictions.
11) n is not an integer as the related clocks are asynchronous to each other

12) f_{MCANH} must always be greater than f_{MCAN}, with f_{MCAN} = 160/80 MHz, f_{MCANH} = 200/100 MHz
13) f_{MCANH} must always be greater than f_{MCAN}, with f_{MCAN} = 160/80 MHz, f_{MCANH} = 250/100 MHz
14) f_{CANXLH} must always be greater than f_{CANXL}, with f_{CANXL} = 160/80 MHz, f_{CANXLH} = 200/100 MHz
15) f_{CANXLH} must always be greater than f_{CANXL}, with f_{CANXL} = 160/80 MHz, f_{CANXLH} = 250/100 MHz
16) Note that f_{SRICS}/f_{FSI}=1 is not supported by all devices. Please refer to the device specific chapters for further restrictions.
17) Do not apply values that would lower f_{FSI} below its reset value which is 100 MHz. This is needed to guarantee a graceful shutdown operation.

Attention**: **The recommended values do not necessarily reflect the default configuration after a reset event. Instead they should give a hint about how to configure the system for the optimal performance. In addition, it may happen that applying an allowed ratio on two selected clocks violates the ratio of others. Due to the complexity, this cannot be shown here in all combinations. The User has to take care not to violate clock ratios depending on the enabled clocks and the use-case.

32.3.5.1.3 Start-up and shut-down

For power up, power down, system and application resets or clock emergencies the clocking system needs to be started up or shut down. This is done by firmware, application software, or by hardware support.

Start-up

- After boot phase controlled by application software: Following power-on reset or system reset
- At the end of boot phase controlled by firmware: Following application reset

Shut-down

- For power down execution by application software
- By hardware during system reset, application reset or clock emergency

Software controlled

Start up controlled by application software

For start up, the ramp oscillator is used to increase the system clock frequency in small steps to avoid large current jumps. Following the boot phase, clock system start-up is under control of the application software. The application software knows the device's setup with respect to the availability of an external clock source (type and frequency) and the needed target frequencies for the SYSPLL and PERPLL. For peripheral clock start-up, the ramp oscillator is not used. The fan-out load of the related clock trees and the target frequencies are lower. Therefore, the current jump is lower and no clock ramp is needed. Here the legacy mode of K-divider stepping can be applied.

Basic Steps for start-up:

- T0:
  - Power up and configure crystal oscillator
  - Power up and configure ramp oscillator (target frequency)
  - Power up and configure SYSPLL with P, N to target DCO frequency, K divider to target frequency
  - Power up and configure PERPLL with P, N to target DCO frequency, set K-dividers to the desired frequencies
- T1: Switch system clock selection to ramp oscillator
- T2: Trigger ramp up
- T3: Wait for end of ramp execution and stable target frequency by observing the RAMPSTAT.FLLLOCK bit
- T4: Switch system clock selection to SYSPLL once the SYSPLL is locked, switch peripheral clock selection to PERPLL once PERPLL is locked
- T5: Step down PERPLL K-dividers towards target frequencies

If the maximum system frequency of the device is 400 MHz and the PPU configured frequency is 400 MHz, the PPU is operated with f_{SRI} which is synchronous to the rest of the system, that means the PPU clock is sourced

from $f_{source0}$. Please be aware that the target f_{RAMP} frequency should be 100 MHz below the configured f_{PLL0} frequency.

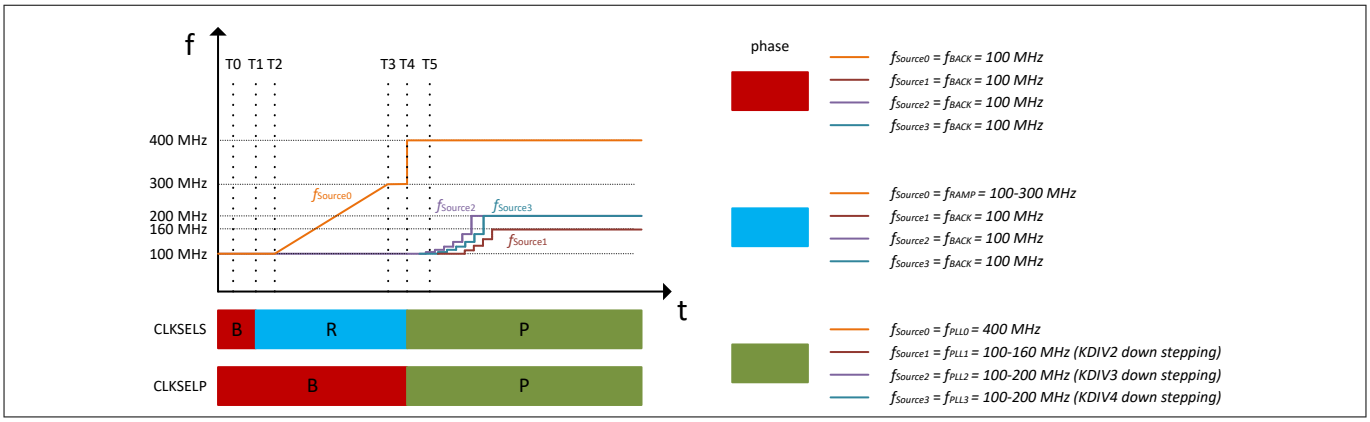


Figure 748 Start up 400 MHz use case

***Note**: Ramp up case : During ramp up it is needed that application software performs a "frequency throttling" in order to avoid generating VDD under voltage that can trigger cold reset.*

Shut down controlled by application software

For shut down, the ramp oscillator is also used to decrease the system clock frequency in small steps to avoid large current jumps. For peripheral clock shut down, the legacy mode of K-divider stepping can be applied. The clock system shut down is under control of the application software.

Basic Steps for shut down:

- T0: Configure the PERPLL K-dividers to achieve 100 MHz frequency
- T1: Switch system clock selection to ramp oscillator once the ramp oscillator reached UFL, switch peripheral clock selection to backup clock
- T2: Trigger ramp down
- T3: Wait for end of ramp execution and stable target frequency
- T4: Switch system clock selection to backup clock
- T5:
  - Power down SYSPLL with P, N to target DCO frequency, K divider to target frequencies
  - Power down PERPLL with P, N to target DCO frequency, K dividers to target frequencies
  - Power down ramp oscillator (ramp slope, target frequency)
  - Power down crystal oscillator

If the maximum system frequency of the device is 400 MHz which is inline with the maximum frequency of the PPU, the PPU is operated with f_{SRI} which is synchronous to the rest of the system, that means the PPU clock is sourced from $f_{source0}$.

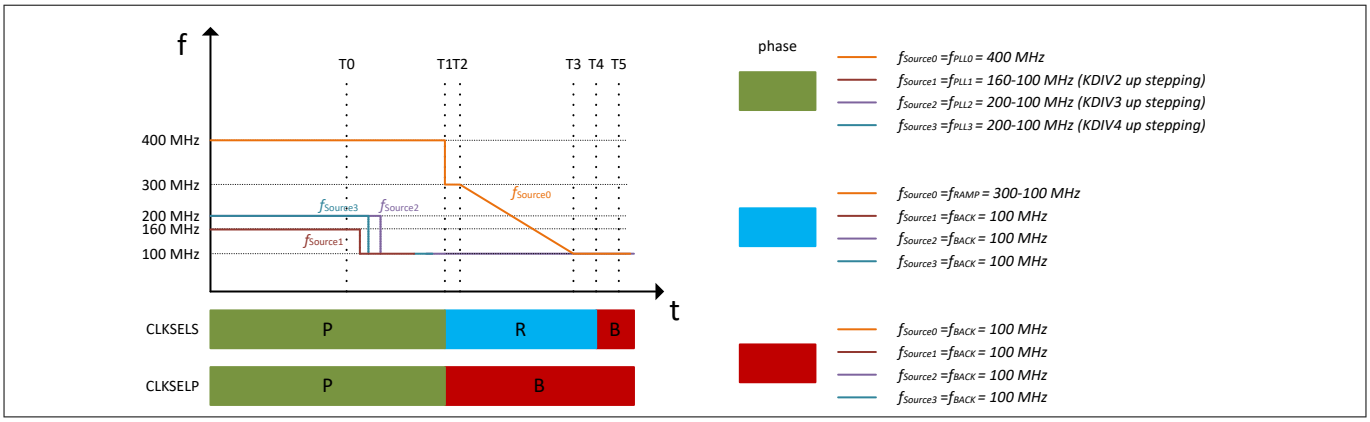


Figure 749 Shut down 400 MHz use case

***Note**: Ramp down case: During ramp down "frequency throttling" of PERPLL is not required. According to load jump assessment, it will not generate a higher VDD voltage that would introduce a long-term reliability issue.*

Hardware-software controlled

During application- or system-reset, system clocks are reduced in frequency before the clock is halted to finally execute the reset. A loss of lock of the *SYSPLL* requires a switch over to a reliable independent clock source. This is also done through continuous frequency reduction using the ramp oscillator.

Application and system reset

For the application and system resets, the device automatically executes a shutdown sequence initiated by the System Modes and Reset (SMR) controller. The processing units, memories and ports are getting triggered to shut down their current operation.

Finally, the clocking system gets triggered to ramp down the system clock frequencies and to halt the clocks while the related hardware reset signals are getting activated to execute the reset. The ramp oscillator is used to gracefully change the frequency of the system root clock f_{HTREE} in small steps. This is done to avoid current load undershoots or overshoot due to an abrupt change in load of active clocked components during reset execution. As resets might occur at any time, hardware takes control of the ramp down process to allow a quick execution of the ramp.

During re-boot following an application reset, firmware re-installs the system clock settings as originally configured by the application. That means, ramp up of root clock frequency as stored by the application in register bit-field CCUCON.CLKSELS. In application and system reset, the PERPLL throttling is not needed.

***Attention**: As $f_{source0}$ is the root source for system clocks, all system clocks are reduced and increased too, while the hardware initiated ramp-down and -up process is executed. Actual settings of related clock dividers are not changed by hardware during that time.*

The following figure shows the sequence which is executed by hardware and firmware in cooperation. The hardware override of the clock selection can be seen in the difference of the CCUCON.CLKSELS and CCUSTAT.CLKSELS. When firmware sets the CCUCON.CLKSELS during re-boot, the override gets cleared.

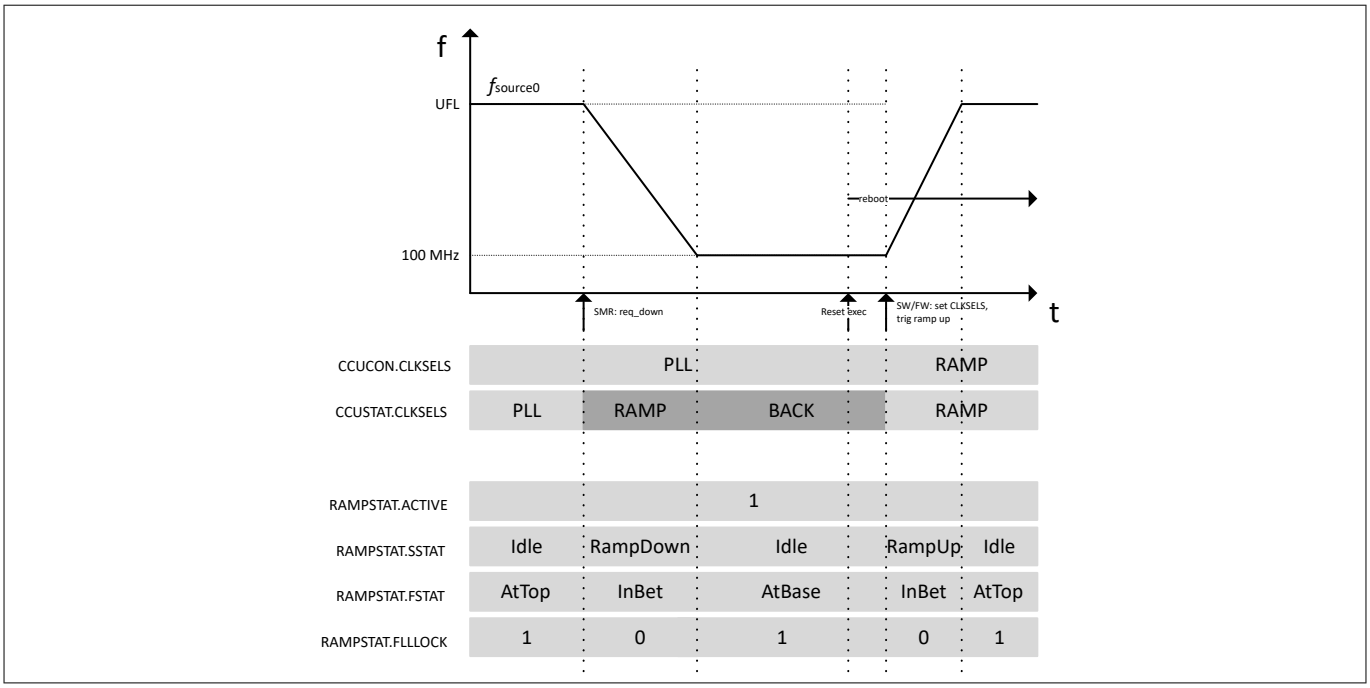


Figure 750 Ramp hardware-software interface during application or system reset

Emergency switch

During normal operation, all module clocks are generated using the PLLs. If there is a failure in PLL operation and they lose their lock, the clocking system automatically switches to an internal, alternate clock source to maintain operation with reduced performance.

In case of a SYSPLL error, the MCCU intermediately switches to the ramp clock f_{RAMP} and finally to f_{BACK} as alternate clock source. After switching over to f_{RAMP}, a ramp down sequence is executed by hardware to reduce current jumps before finally switching over to f_{BACK}.

In case of a PERPLL error, the PCCU directly switches to f_{BACK} as alternate clock source.

Both PLLs operate independent from each other, that means in case of an loss of lock event, only the affected clock domain (System or Peripheral) is switched to its alternate clock source.

A clock emergency switch is triggered by the occurrence of at least one of the following conditions:

- Loss of lock event of the SYSPLL while selected as clock source (CCUCON.CLKSELS = PLL)
- Loss of lock event of the PERPLL while selected as clock source for (CCUCON.CLKSELP = PLL)

A clock emergency switch stays active until CLKSELx is re-written by software.

On detection of clock emergency switch, an alarm is signaled to the SMU.

***Note**: After a clock emergency, the bit-fields CLKSELx have to be re-written. After the root cause for the clock error disappears, the application can reconfigure the clock system, including CLKSELx.*

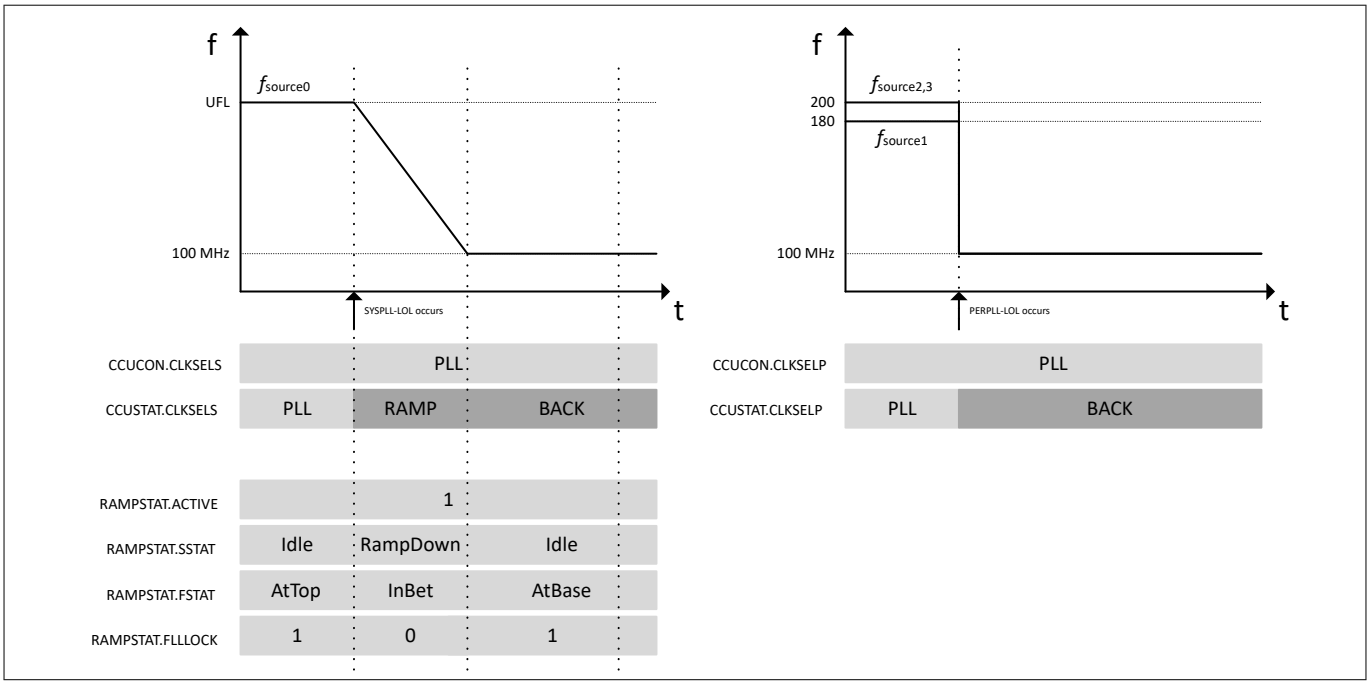


Figure 751 Ramp hardware-software interface during clock emergency switch

32.3.5.2 External clock output

Two external clock outputs are provided through pins EXTCLK0 and EXTCLK1. These external clocks can be enabled or disabled using the external clock control register (EXTCON):

- EXTCON.EN0: enables pin EXTCLK0
- EXTCON.EN1: enables pin EXTCLK1
- EXTCON.SEL0: selects internal clock to be presented at EXTCLK0
- EXTCON.SEL1: selects internal clock to be presented at EXTCLK1
- EXTCON.NSEL1: selects if the clock on EXTCLK1 is inverted or not

***Note**: Changing the content of bit-field EXTCON.SEL0 and EXTCON.SEL1 can lead to glitches in the clock signal at pins EXTCLK0 and EXTCLK1*

There are two kinds of additional clock dividers that can be used to generate auxiliary clocks presented on the external outputs. This is to generate a wide range of frequencies for external use.

1. Fractional divider based on f_{SPB}
2. Integer divider based on f_{SPB}

In addition, EXTCLK1 output can be inverted to support differential output signal generation on GPIOs.

32.3.5.2.1 Fractional divider

The fractional divider function enables application software to generate an external clock derived from the f_{SPB} by using a programmable divider.

Overview

The fractional divider is controlled through the Fractional divider control register (EXTFDCON).

The circuit divides the input clock f_{SPB} either by the factor 1÷n or by a fraction of n÷1024 for any value of n from 0 to 1023. This clock is additionally divided by a factor of two to guarantee a 50% duty cycle and outputs the clock f_{FRAC}.

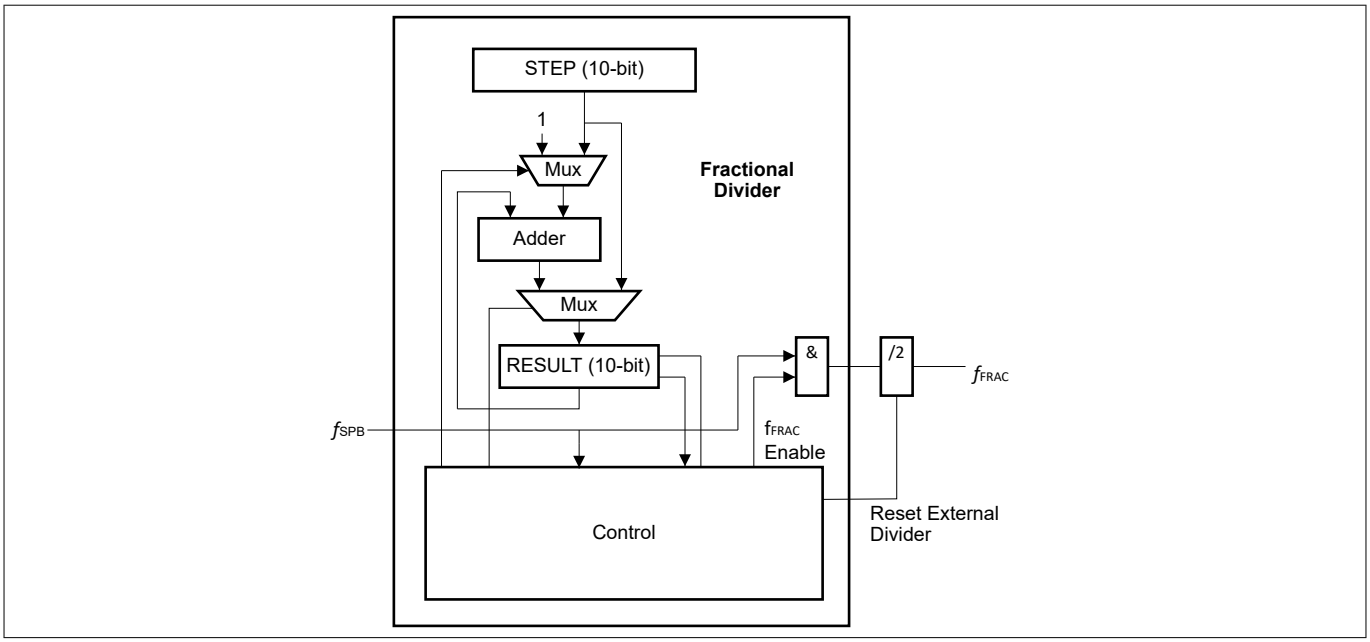


Figure 752 Fractional divider block diagram

The adder logic of the fractional divider can be configured for two operating modes:

- Normal mode: Reload counter (RESULT = RESULT + 1), generating an output clock pulse on counter overflow
- Fractional divider mode: Adder that adds a STEP value to the RESULT value and generates an output clock pulse on counter overflow

Normal divider mode

In normal divider mode (EXTFDCON.DM = 01_B), the fractional divider behaves as a reload counter (addition of +1) that generates an output clock pulse on the transition from $3FF_H$ to 000_H. EXTFDCON.STEP determines the reload value. For details, please refer to the register section).

The output frequencies in Normal Divider Mode are defined according to the following formulas:

$$f_{FRAC} = \frac{f_{SPB} \times \frac{1}{n}}{2}, \text{with } n = 1024 - \text{STEP} \quad (60)$$

In order to get $f_{FRAC} = f_{SPB} \div 2$, STEP must be programmed with $3FF_H$.

Fractional divider mode

When the fractional divider mode is selected (EXTFDCON.DM = 10_B), the output is derived from the input clock f_{SPB} by division of a fraction of n÷1024, for any value of n from 0 to 1023, followed by the division of two. In general, the fractional divider mode makes it possible to program the average output clock frequency with a higher accuracy than in normal divider mode.

In fractional divider mode, a pulse is generated depending on the result of the addition RESULT + STEP. If the addition leads to an overflow over $3FF_H$, a pulse is generated for the divider by two.

***Note**: This clock should only be used to observe the clock frequency/shape but not jitter.*

The output frequencies in fractional divider mode are defined according to the following formula:

$$f_{FRAC} = \frac{f_{SPB} \times \frac{STEP}{1024}}{2} \quad (61)$$

32.3.5.2.2 Output multiplexing

The clock output EXTCLK0 and EXTCLK1 can be configured for either a programmable frequency output f_{FRAC} (fractional divider) or f_{DIV1} (integer divider) or one of the internal system and peripheral frequencies f_{XXX}.

Clocks f_{FRAC} and f_{DIV1}, using $f_{OUT} = \frac{f_{SPB}}{DIV1 + 1}$, are generated through a counter, so the output frequency can be selected in small steps.

The EXTCON provides control over the output generation (frequency and activation).

The following figure summarizes which internal clocks can be multiplexed to the external clock pins. The detailed list of $f_{XXX\text{-}YYY}$ and $f_{III\text{-}JJJ}$ can be taken from register bit-fields EXTCONSEL0 and EXTCON.SEL1.

32.4 Registers

32.4.1 Overview

All clocking system related registers may be accessed by any CPU in the system. However, it is strongly suggested that only one CPU is used to control the clocks to avoid consistency issues. After reset, only CPU0 and the Cerberus are enabled for write access through the APU control registers.

The clocking system contains several registers that cannot be directly written but through a posting mechanism, that means there is a "master" register inside the MCCU and "mirror" registers in DCCU or PCCU. In order to maintain data coherency during a programming sequence, it is necessary to wait with any following register access while a posted write transaction is ongoing and the write finally takes effect on the final destination.

The bit-field CCUSTAT.LCK has been introduced to reflect the status of posted write transactions. It indicates if a writable clock system registers can be updated with a new value or if the register is still locked (due to an ongoing previous write access) and a write action from the bus side has no effect. It shall be monitored to pause the sequence of register accesses until it is cleared by hardware. The worst case delay of CCUSTAT.LCK being set is 70 µs.

***Note**: There will be no bus error reported if there is a write access to a clocking system registers while the lock bit set.*

The lock bit is blocking writes to the following registers:

- OSCCON
- RAMPCON0
- SYSPLLCON0, SYSPLLCON1, SYSPLLCON2
- PERPLLCON0, PERPLLCON1, PERPLLCON2
- PMODCON0
- CCUCON
- SYSCCUCON0, SYSCCUCON1
- PERCCUCON0, PERCCUCON1

The lock bit is set under any of the below listed conditions. It is cleared when the posted write transaction is complete:

- Write to SYSPLLCON1
- Write to PERPLLCON0, PERPLLCON1, PERPLLCON2
- Write to PMODCON0
- Write 1_B to bit-field SYSCCUCON0.UP, SYSCCUCON1.UP
- Write to PERCCUCON0, PERCCUCON1

32.4.2 Register overview - access mode glossary

Table 1294 Register overview - access mode glossary

Keyword	Description
CSE	Access protection using PROT register PROTCSE.
E	Access protection using PROT register PROTE.
SE	Access protection using PROT register PROTSE.
APU-P	Protection group consisting of registers ACCEN_WRA, ACCEN_WRB, ACCEN_RDA, ACCEN_RDB, ACCEN_VM, ACCEN_PRS.
P	Access protection using *APU-P* registers.
APU-PCS	Protection group consisting of registers ACCENCS_WRA, ACCENCS_WRB, ACCENCS_RDA, ACCENCS_RDB, ACCENCS_VM, ACCENCS_PRS.
PCS	Access protection using *APU-PCS* registers.
BE	Always returns a Bus Error.
U	No access restrictions.
PROT	Access restrictions as defined in the PROT register access rules.
STP	Access only during start-up, before start-up protection is set (SMM_STCON.ST_PROT=1).

32.4.3 Registers overview - CLOCK (ascending offset address)

Table 1295 Registers overview - CLOCK (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ID	Module Identification register	0008_H	P	BE	PowerOn Reset	5569
PROTE	PROT Register Endinit	0020_H	U	SV, PROT	Application Reset	5570
PROTSE	PROT register safe endinit	0024_H	U	SV, PROT	Application Reset	5572
PROTCSE	PROT Register Cyber-Secure APU Endinit	0028_H	U	SV, PROT	See 5574	5574
ACCEN_WRA	Write access enable register A	0040_H	U	SE, SV	Application Reset	5575

(table continues...)

Table 1295 (continued) Registers overview - CLOCK (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ACCEN_WRB	Write access enable register B	0044_H	U	SE, SV	Application Reset	5576
ACCEN_RDA	Read access enable register A	0048_H	U	SE, SV	Application Reset	5576
ACCEN_RDB	Read access enable register B	$004C_H$	U	SE, SV	Application Reset	5577
ACCEN_VM	VM access enable register	0050_H	U	SE, SV	Application Reset	5577
ACCEN_PRS	PRS access enable register	0054_H	U	SE, SV	Application Reset	5578
ACCENCS_WRA	ACCENCS write access enable register A	0060_H	U	CSE, SV	See 5579	5579
ACCENCS_WRB	ACCENCS write access enable register B	0064_H	U	CSE, SV	Application Reset	5579
ACCENCS_RDA	ACCENCS read access enable register A	0068_H	U	CSE, SV	Application Reset	5580
ACCENCS_RDB	ACCENCS read access enable register B	$006C_H$	U	CSE, SV	Application Reset	5580
ACCENCS_VM	ACCENCS VM access enable register	0070_H	U	CSE, SV	Application Reset	5581
ACCENCS_PRS	ACCENCS PRS access enable register	0074_H	U	CSE, SV	Application Reset	5582
OSCCON	Crystal oscillator control register	0100_H	P	SV, E, P	Undefined	5582
RAMPCON0	Ramp oscillator control register 0	0200_H	P	SV, E, P	System Reset	5586
RAMPSTAT	Ramp oscillator status register	$020C_H$	P	BE	See 5588	5588
SYSPLLCON0	System-PLL configuration register 0	0300_H	P	SV, E, P	System Reset	5590
SYSPLLCON1	System-PLL configuration register 1	0304_H	P	SV, E, P	Undefined	5591
SYSPLLCON2	System-PLL configuration register 2	0308_H	P	SV, E, P	System Reset	5593
SYSPLLSTAT	System-PLL status register	$030C_H$	P	BE	System Reset	5593
SMODCON0	System-PLL modulation trim configuration register 0	0340_H	P	SV, E, P	PowerOn Reset	5594
PERPLLCON0	Peripheral-PLL configuration register 0	0380_H	P	SV, E, P	System Reset	5595
PERPLLCON1	Peripheral-PLL configuration register 1	0384_H	P	SV, E, P	Undefined	5597

(table continues...)

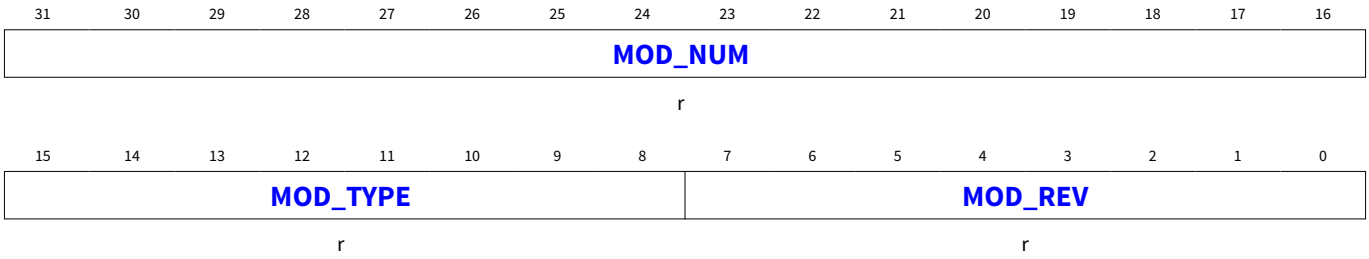
Table 1295 (continued) Registers overview - CLOCK (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PERPLLCON2	Peripheral-PLL configuration register 2	0388_H	P	SV, E, P	System Reset	5599
PERPLLSTAT	Peripheral-PLL status register	$038C_H$	P	BE	System Reset	5600
PMODCON0	Peripheral-PLL modulation trim configuration register 0	$03A0_H$	P	SV, E, P	PowerOn Reset	5601
CCUCON	Global clock control register	0400_H	P	SV, E, P	Undefined	5602
CCUSTAT	Global clock status register	0404_H	P	BE	See 5604	5604
SYSCCUCON0	System CCU clock control register 0	0410_H	P	SV, E, P	Undefined	5605
SYSCCUCON1	System CCU clock control register 1	0414_H	P	SV, E, P	Undefined	5610
PERCCUCON0	Peripheral CCU clock control register 0	0420_H	P	SV, E, P	Undefined	5615
PERCCUCON1	Peripheral CCU clock control register 1	0424_H	P	SV, E, P	Undefined	5620
EXTCON	External clock control register	0430_H	P	SV, E, P	System Reset	5625
EXTFDCON	Fractional divider control register	0434_H	P	SV, E, P	System Reset	5627
CCUMON	Global clock monitor register	0500_H	P	SV, E, P	System Reset	5628
OSCMON0	EVR oscillator monitor register	0504_H	P	SV, E, P	System Reset	5632
OSCMON1	Crystal oscillator monitor register	0508_H	P	SV, E, P	System Reset	5633
OSCSTAT	Crystal oscillator status register	$050C_H$	P	BE	System Reset	5635
RAMPMON	Ramp oscillator monitor register	0510_H	P	SV, E, P	System Reset	5637
MONSTAT	Global clock monitor status register	0518_H	P	BE	PowerOn Reset	5638

32.4.4 Module Identification register

ID Offset address: 0008_H

Module Identification register PowerOn Reset value: $00EE\ C013_H$



Field	Bits	Type	Description
MOD_REV	7:0	r	**Module revision** Indicates the revision number of the implementation.
MOD_TYPE	15:8	r	**Module type** This internal marker is fixed to $C0_H$.
MOD_NUM	31:16	r	**Module number** Indicates the module identification number ($00EE_H$ = CLOCK).

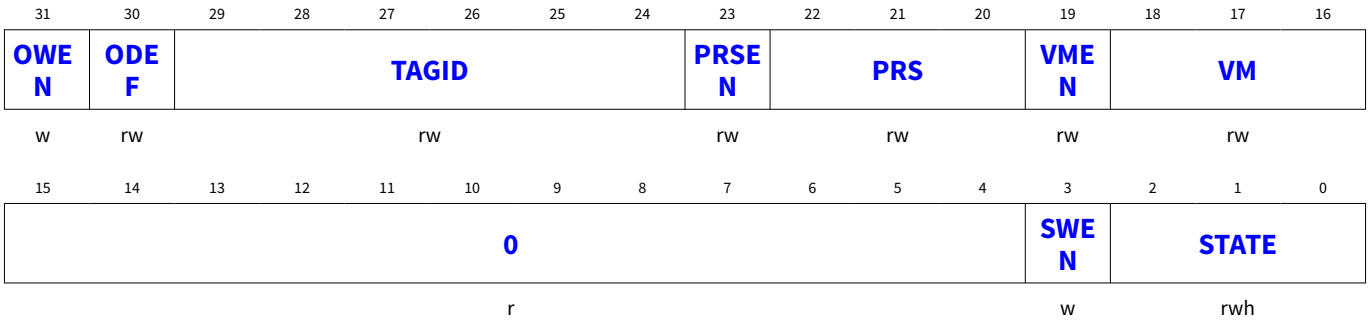
32.4.5 PROT Register Endinit

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner or CS master to update the PROT state as described in the PROT mechanism.

The PROTE register controls lock / unlock of the local Endinit (E) protected control registers.

PROTE Offset address: 0020_H

PROT Register Endinit Application Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
VM	18:16	rw	**Virtual machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual machine definition enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.
PRSEN	23	rw	**Protection set definition enable for PROT owner** 0_B **Disable**: PRS is not part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG-ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; must be written with 0.

Table 1296 Access mode restrictions of PROTE sorted by descending priority

Mode name	Access mode		Description
write 1 to .**SWEN**	rwh	STATE	
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

32.4.6 PROT register safe endinit

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner or CS master to update the PROT state as described in the PROT mechanism.

The PROTSE register controls lock / unlock of the local Safe Endinit (SE) protected control registers.

PROTSE Offset address: 0024_H

PROT register safe endinit Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OWEN	**ODEF**	**TAGID**						**PRSEN**	**PRS**			**VMEN**	**VM**		
w	rw	rw						rw	rw			rw	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												**SWEN**	**STATE**		
r												w	rwh		

Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write

(table continues...)

(continued)

Field	Bits	Type	Description
VM	18:16	rw	**Virtual machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual machine definition enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.
PRSEN	23	rw	**Protection set definition enable for PROT owner** 0_B **Disable**: PRS is not part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG-ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; must be written with 0.

Table 1297 Access mode restrictions of PROTSE sorted by descending priority

Mode name	Access mode		Description
write 1 to .**SWEN**	rwh	STATE	
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

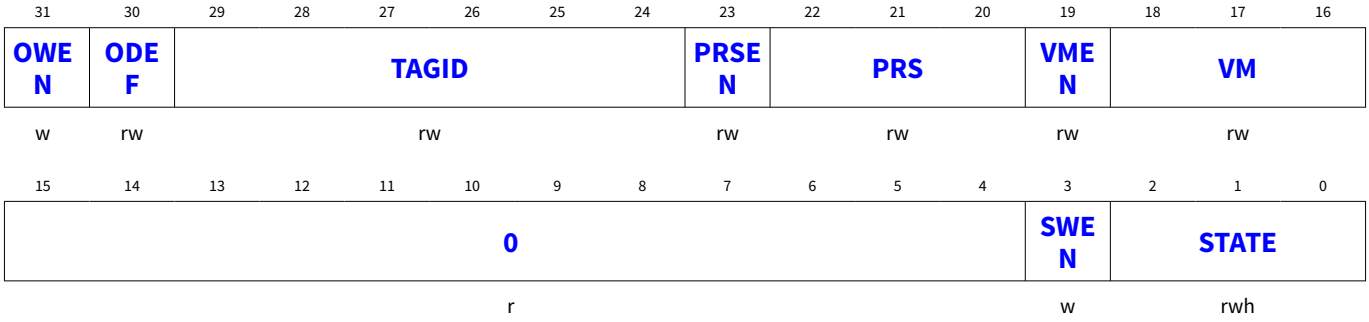
32.4.7 PROT Register Cyber-Secure APU Endinit

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner or CS master to update the PROT state as described in the PROT mechanism.

The PROTCSE register controls lock / unlock of the local Cyber Secure APU Endinit (CSAE) protected control registers.

PROTCSE Offset address: 0028_H

PROT Register Cyber-Secure APU Endinit Reset values see: Table 1299



Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
VM	18:16	rw	**Virtual machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual machine definition enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.

(table continues...)

(continued)

Field	Bits	Type	Description
PRSEN	23	rw	**Protection set definition enable for PROT owner** 0_B **Disable**: PRS is not part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG-ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; must be written with 0.

Table 1298 Access mode restrictions of PROTCSE sorted by descending priority

Mode name	Access mode		Description
write 1 to .**SWEN**	rwh	STATE	
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

Table 1299 Reset values of PROTCSE

Reset type	Reset value	Note
Application Reset	4C00 0000$_H$	ODEF=1 and TAGID default value is D tag ID of CPUCS; Start in Idle state.
After Boot-FW value	7F00 0007$_H$	

32.4.8 Write access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

ACCEN_WRA Offset address: 0040$_H$

Write access enable register A Application Reset value: 1000 0003$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access

32.4.9 Write access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

ACCEN_WRB Offset address: 0044_H

Write access enable register B Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access
0	31:8	r	**Reserved** Read as 0; must be written with 0.

32.4.10 Read access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

ACCEN_RDA Offset address: 0048_H

Read access enable register A Application Reset value: $FFFF\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access

32.4.11 Read access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

ACCEN_RDB Offset address: $004C_H$

Read access enable register B Application Reset value: $0000\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access
0	31:8	r	**Reserved** Read as 0; must be written with 0.

32.4.12 VM access enable register

This register defines which virtual machine encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each VMx.

ACCEN_VM Offset address: 0050_H

VM access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for VM ID q** This bit enables read access to the access protected region for transactions with the VM ID q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for VM ID q** This bit enables write access to the access protected region for transactions with the VM ID q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; must be written with 0.

32.4.13 PRS access enable register

This register defines which protection register set encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each PRSx.

ACCEN_PRS Offset address: 0054_H

PRS access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for PRS q** This bit enables read access to the access protected region for transactions with the PRS q. 0_B Disabled for read access 1_B Enabled for read access

(table continues...)

(continued)

Field	Bits	Type	Description
WRq (q=00-07)	q+16	rw	**Write access enable for PRS q** This bit enables write access to the access protected region for transactions with the PRS q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; must be written with 0.

32.4.14 ACCENCS write access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

ACCENCS_WRA Offset address: 0060_H

ACCENCS write access enable register A Reset values see: Table 1300

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access

Table 1300 Reset values of ACCENCS_WRA

Reset type	Reset value	Note
Application Reset	$0000\ 1000_H$	D and DS tag ID of CPUcs.
After Boot-FW value	$0000\ 0000_H$	

32.4.15 ACCENCS write access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

ACCENCS_WRB Offset address: 0064_H

ACCENCS write access enable register B Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access
0	31:8	r	**Reserved** Read as 0; must be written with 0.

32.4.16 ACCENCS read access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

ACCENCS_RDA Offset address: 0068_H

ACCENCS read access enable register A Application Reset value: $0000\ 1000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access

32.4.17 ACCENCS read access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

ACCENCS_RDB Offset address: $006C_H$

ACCENCS read access enable register B Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access
0	31:8	r	**Reserved** Read as 0; must be written with 0.

32.4.18 ACCENCS VM access enable register

This register defines which virtual machine encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each VMx.

ACCENCS_VM Offset address: 0070_H

ACCENCS VM access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for VM ID q** This bit enables read access to the access protected region for transactions with the VM ID q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for VM ID q** This bit enables write access to the access protected region for transactions with the VM ID q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; must be written with 0.

32.4.19 ACCENCS PRS access enable register

This register defines which protection register set encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each PRSx.

ACCENCS_PRS Offset address: 0074_H

ACCENCS PRS access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for PRS q** This bit enables read access to the access protected region for transactions with the PRS q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for PRS q** This bit enables write access to the access protected region for transactions with the PRS q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; must be written with 0.

32.4.20 Crystal oscillator control register

Note: Please refer to the device specific register chapter for the reset values.

OSCCON Offset address: 0100_H

Crystal oscillator control register Reset value: Undefined

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		CLKSELPHY2	CLKSELPHY0	0	RSVD26	INSEL		0		RSVD21	RSVD20	X2CAP3EN	X2CAP2EN	X2CAP1EN	X2CAP0EN
r		rw	rw	r	rw	rw		r		rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
X1CAP3EN	X1CAP2EN	X1CAP1EN	X1CAP0EN	RSVD11	MODE			HYSCTL		HYSEN	RSVD4	RSVD2		GAINSEL	
rw	rw	rw	rw	rw	rw			rw		rw	rw	rw		rw	

Field	Bits	Type	Description
GAINSEL	1:0	rw	**Oscillator gain selection** Gain of the oscillator output stage. It can be used to adjust the peak to peak amplitude. 00_B Minimum gain configuration. 01_B Intermediate gain configuration. 10_B Intermediate gain configuration. 11_B Maximum gain configuration (default value) Note: It is not recommended to change the default value
RSVD2	3:2	rw	**Reserved** Read as reset value; shall be written with reset value.
RSVD4	4	rw	**Reserved** Read as reset value; shall be written with reset value.
HYSEN	5	rw	**Enable shaper hysteresis** 0_B Hysteresis is disabled 1_B Hysteresis is enabled (recommended)
HYSCTL	7:6	rw	**Shaper hysteresis control. Applicable only if HYSEN=1** 00_B Hysteresis setting 1 (highest hysteresis, default) Note: It is not recommended to change the default value 01_B Hysteresis setting 2; Reserved for future use 10_B Hysteresis setting 3; Reserved for future use 11_B Hysteresis setting 4; Reserved for future use
MODE	10:8	rw	**Oscillator mode** This bit-field defines which mode can be used and if the oscillator entered the power-saving mode or not. 000_B **XTALOSC**: Crystal mode 010_B **EXTCLK**: External clock mode with shaper 011_B **PDOWN**: Power down mode 111_B **EXTCLKSB**: External clock mode without shaper Note: This mode should be used only for debug purpose. For normal operation use only modes with shaper enabled. **others,** Reserved, do not use.
RSVD11	11	rw	**Reserved** Read as reset value; shall be written with reset value.
X1CAP0EN	12	rw	**XTAL1 capacitance 0 enable** ***Note**: Total capacitance for XTAL1 is the sum of the enabled capacitance 0 to 3.* 0_B Capacitance C_{L0} is disabled 1_B Capacitance C_{L0} is enabled

(table continues...)

(continued)

Field	Bits	Type	Description
X1CAP1EN	13	rw	**XTAL1 capacitance 1 enable** ***Note:*** *Total capacitance for XTAL1 is the sum of the enabled capacitance 0 to 3.* 0_B Capacitance C_{L1} is disabled 1_B Capacitance C_{L1} is enabled
X1CAP2EN	14	rw	**XTAL1 capacitance 2 enable** ***Note:*** *Total capacitance for XTAL1 is the sum of the enabled capacitance 0 to 3.* 0_B Capacitance C_{L2} is disabled 1_B Capacitance C_{L2} is enabled
X1CAP3EN	15	rw	**XTAL1 capacitance 3 enable** ***Note:*** *Total capacitance for XTAL1 is the sum of the enabled capacitance 0 to 3.* 0_B Capacitance C_{L3} is disabled 1_B Capacitance C_{L3} is enabled
X2CAP0EN	16	rw	**XTAL2 capacitance 0 enable** ***Note:*** *Total capacitance for XTAL2 is the sum of the enabled capacitance 0 to 3.* 0_B Capacitance C_{L0} is disabled 1_B Capacitance C_{L0} is enabled
X2CAP1EN	17	rw	**XTAL2 capacitance 1 enable** ***Note:*** *Total capacitance for XTAL2 is the sum of the enabled capacitance 0 to 3.* 0_B Capacitance C_{L1} is disabled 1_B Capacitance C_{L1} is enabled
X2CAP2EN	18	rw	**XTAL2 capacitance 2 enable** ***Note:*** *Total capacitance for XTAL2 is the sum of the enabled capacitance 0 to 3.* 0_B Capacitance C_{L2} is disabled 1_B Capacitance C_{L2} is enabled
X2CAP3EN	19	rw	**XTAL2 capacitance 3 enable** ***Note:*** *Total capacitance for XTAL2 is the sum of the enabled capacitance 0 to 3.* 0_B Capacitance C_{L3} is disabled 1_B Capacitance C_{L3} is enabled
RSVD20	20	rw	**Reserved** Read as reset value; shall be written with reset value.

(table continues...)

(continued)

Field	Bits	Type	Description
RSVD21	21	rw	**Reserved** Read as reset value; shall be written with reset value.
INSEL	25:24	rw	**Input selection** This bit-field defines the primary clock source f_{IN} for the System-PLL and Peripheral-PLL as listed below. 00_B **BACK**: f_{BACK} is used as clock source for System-PLL and Peripheral-PLL 01_B **OSC**: f_{OSC} is used as clock source for System-PLL and Peripheral-PLL 10_B **SYSCLK**: $f_{PADSYSCLK}$ is used as clock source for System-PLL and Peripheral-PLL 11_B **OSC_F**: f_{OSC_F} is used as clock source for System-PLL f_{OSC} is used as clock source for Peripheral-PLL
RSVD26	26	rw	**Reserved** Read as reset value; shall be written with reset value.
CLKSELPHY0	28	rw	**Reference clock Input selection** This bit-field defines the alternative reference clock source $f_{HSPHY0ALTREF}$ for the HSPHY.PHY0 PMA module as listed below. ***Note**: In addition, the user has to set the HSPHY PHY0_CTRL1.PRS bits accordingly. For details please refer to the HSPHY UM register section.* ***Note**: Please refer to the datasheet for availability* 0_B **OSC**: f_{OSC} is used as reference clock source 1_B **PCIEREF**: $f_{PCIEREF}$ is used as reference clock source
CLKSELPHY2	29	rw	**Reference clock Input selection** This bit-field defines the alternative reference clock source $f_{HSPHY2ALTREF}$ for the HSPHY.PHY2 PMA module as listed below. ***Note**: In addition, the user has to set the HSPHY PHY2_CTRL1.PRS bits accordingly. For details please refer to the HSPHY UM register section.* ***Note**: Please refer to the datasheet for availability* 0_B **OSC**: f_{OSC} is used as reference clock source 1_B **PCIEREF**: $f_{PCIEREF}$ is used as reference clock source
0	23:22, 27, 31:30	r	**Reserved** Read as reset value; shall be written with reset value.

Table 1301 Access mode restrictions of OSCCON sorted by descending priority

Mode name	Access mode		Description
CCUSTAT.LCK == 0	rw	, CLKSELPHY0, CLKSELPHY2, GAINSEL, HYSCTL, HYSEN, INSEL, MODE , RSVD11 , RSVD2 , RSVD20 , RSVD21 , RSVD26 , RSVD4 , X1CAP0EN, X1CAP1EN, X1CAP2EN, X1CAP3EN, X2CAP0EN, X2CAP1EN, X2CAP2EN, X2CAP3EN	Normal access
(default)	r	, CLKSELPHY0, CLKSELPHY2, GAINSEL, HYSCTL, HYSEN, INSEL, MODE , RSVD11 , RSVD2 , RSVD20 , RSVD21 , RSVD26 , RSVD4 , X1CAP0EN, X1CAP1EN, X1CAP2EN, X1CAP3EN, X2CAP0EN, X2CAP1EN, X2CAP2EN, X2CAP3EN	Limited access

32.4.21 Ramp oscillator control register 0

This register configures the sequencer within the ramp oscillator.

RAMPCON0 Offset address: 0200_H

Ramp oscillator control register 0 System Reset value: $0000\ 0190_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0												PWR	CMD		
r												rw	rwh		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0						UFL									
r						rw									

Field	Bits	Type	Description
UFL	9:0	rw	**Upper frequency limit** Upper frequency limit in 1 MHz units. Acceptable values are integer multiples of 100 MHz up to a maximum of *f*PLL0-100 MHz as per the datasheet. This recommendation is due to the ramp oscillator inaccuracy. ***Note**: A new value only takes effect with the start of the next ramping sequence, that means it does not change a running sequence.* ***Note**: It is mandatory by the application software to enter this value during clocking system setup. This is necessary as the value is used by firmware during the re-boot process after an application reset.* ***Note**: Ensure a waiting time of 20 f_{SPB} clock cycles before the next write access to RAMPCON0.* ***Note**: The minimum upper frequency is UFLmin = 130 [MHz]. Frequencies below this limit should not be used.* *The UFL_min is calculated with the formula : 100 [MHz]+ StepSize_min* *StepSize_min = 30 MHz*
CMD	18:16	rwh	**Ramp sequencer command** These bits will direct a ramp sequence according to its control parameters (*UFL*). Writing to this field directly triggers execution of a ramp sequence. The cmd automatically clears to *IDLE* after 20 f_{SPB} clock cycles. ***Note**: The control parameters are activated together with a command trigger, that means changing them without issuing a command has no direct effect on the actual output frequency of the oscillator or the ramp sequence. The new parameters get active with the next command.* *A command trigger only gets executed when the ramp oscillator is not busy as indicated by RAMPSTAT.SSTAT = Idle. Otherwise it is ignored.* ***Note**: Ensure a waiting time of 20 f_{SPB} clock cycles before the next write access to RAMPCON0.* 000_B **Idle**: Idle, do nothing, actual frequency is kept constant. 001_B **RampUp**: Ramp-up until the upper frequency limit has been reached. 010_B **RampDown**: Ramp-down until the base frequency has been reached. **others,** Reserved, do not use.

(table continues...)

(continued)

Field	Bits	Type	Description
PWR	19	rw	**Power** This bit is used to power-up and power-down the Ramp Oscillator. ***Note:*** *Ensure a waiting time of 20 f_{SPB} clock cycles before the next write access to RAMPCON0.* 0_B power down 1_B power up
0	15:10, 31:20	r	**Reserved** Read as 0; must be written with 0.

Table 1302 Access mode restrictions of RAMPCON0 sorted by descending priority

Mode name	Access mode		Description
CCUSTAT.LCK == 0	rw	PWR, UFL	Normal access
	rwh	CMD	
(default)	r	PWR, UFL	Limited access
	rh	CMD	

32.4.22 Ramp oscillator status register

This register monitors the status of the sequencer within the ramp oscillator.

RAMPSTAT Offset address: $020C_H$

Ramp oscillator status register Reset values see: Table 1303

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0				RSVD27	RSVD26	RSVD25	FLLLOCK	ACTIVE	0	SSTAT		0		FSTAT	
r				rh	rh	rh	rh	rh	r	rh		r		rh	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															
r															

Field	Bits	Type	Description
FSTAT	17:16	rh	**Frequency status** These bits reflect the ramp frequency status of the ramp oscillator. They automatically change during a ramp-up and ramp-down sequence. 00_B **InBet**: The ramp oscillator's output frequency is between the base frequency and the upper frequency limit. 01_B **AtBase**: The ramp oscillator's output frequency equals the base frequency which is 100 MHz. 10_B **AtTop**: The ramp oscillator's output frequency equals the upper frequency limit. **others,** Reserved, undefined state
SSTAT	21:20	rh	**Sequencer status** Status bits to reflect the current state of the ramp sequencer. 00_B **Idle**: The ramp oscillator's control sequencer is idle, that means not executing a ramp. The output frequency f_{RAMP} is constant either at the base frequency or at the top frequency as reported in FSTAT. 01_B **RampUp**: Frequency increasing 10_B **RampDown**: Frequency decreasing **others,** Reserved, undefined state
ACTIVE	23	rh	**Power status** This bit is used to reflect the actual operational status of the ramp oscillator. 0_B Powered down 1_B Powered up
FLLLOCK	24	rh	**FLL lock status** Reflects if the ramp oscillator has locked to its configured target frequency. ***Note**: This field is only valid if SSTAT = Idle; a change from 1 to 0 is considered as "Loss of Lock". A change from 1 to 0 during ramp operation is intended and not considered as "Loss of Lock".* 0_B The ramp oscillator is not locked to its target frequency and shall not be used for system operation. 1_B The ramp oscillator is locked to its target frequency and can be used for system operation.
RSVD25	25	rh	**Reserved** Read as 0.
RSVD26	26	rh	**Reserved** Read as 0.
RSVD27	27	rh	**Reserved** Read as 0.

(table continues...)

(continued)

Field	Bits	Type	Description
0	15:0, 19:18, 22, 31:28	r	**Reserved** Read as 0; must be written with 0.

Table 1303 Reset values of RAMPSTAT

Reset type	Reset value	Note
System Reset	$0000\ 0000_H$	
After Boot-FW value	$0000\ XXXX\ X0XX\ 00XX\ 0000\ 0000\ 0000\ 0000_B$	Value after Application Reset – X depending on the ramp-up oscillator status.

32.4.23 System-PLL configuration register 0

SYSPLLCON0 Offset address: 0300_H

System-PLL configuration register 0 System Reset value: $0000\ 2C00_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0													PDIV		
r													rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	NDIV							0				MODEN	0	RESLD	PLLPWR
r	rw							r				rw	r	rwh	rw

Field	Bits	Type	Description
PLLPWR	0	rw	**System-PLL power mode** ***Note**: If the PLL has been powered down and is re-enabled by PLLPWR = 1, a wait period of 1 ms has to be applied until it is stable without jitter. After asserting this bit,the SYSPLL power status shall be checked (SYSPLLSTAT.PWRSTAT == 1) before the next SYSPLL KDIV configuration* ***Note**: A write to SYSPLLCON0.PLLPWR bit will set CCUSTAT.LCK bit until the handshake is completed.* 0_B The complete System-PLL block is put into a power-down mode and can no longer be used. 1_B Powered-up, normal behavior
RESLD	1	rwh	**Restart DCO Lock Detection** Setting this bit clears bit SYSPLLSTAT.LOCK and restarts the System-PLL's DCO lock detection. Reading this bit returns always a zero.

(table continues...)

(continued)

Field	Bits	Type	Description
MODEN	3	rw	**Modulation enable** This bit controls the activation of the frequency modulation of the System-PLL. 0_B Frequency modulation is not activated 1_B Frequency modulation is activated
NDIV	14:8	rw	**N-divider value** The value the N-divider operates with is NDIV+1.
PDIV	18:16	rw	**P-divider value** The value the P-divider operates with is PDIV+1.
0	2, 7:4, 15, 31:19	r	**Reserved** Read as 0; must be written with 0.

Table 1304 Access mode restrictions of SYSPLLCON0 sorted by descending priority

Mode name	Access mode		Description
CCUSTAT.LCK == 0	rw	MODEN, NDIV, PDIV, PLLPWR	Normal access
	rwh	RESLD	
(default)	r	MODEN, NDIV, PDIV, PLLPWR	Limited access
	rh	RESLD	

32.4.24 System-PLL configuration register 1

***Note**: Writing to this register sets the CCUSTAT.LCK bit and automatically triggers an update transfer of the register value to the System-PLL's k-divider logic.*

Note: Please refer to the device specific register chapter for the reset values.

SYSPLLCON1 Offset address: 0304_H

System-PLL configuration register 1 Reset value: Undefined

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
K3PREDIV				0	K3DIV			0		K2PREDIV		K2DIV			
rw				r	rw			r		rw		rw			

Field	Bits	Type	Description
K2DIV	3:0	rw	**K2-divider value** This bit-field defines the dividing factor of the K2-divider (*K2*). The effective value is K2 = K2DIV+1. $f_{PLL0} = f_{PC0} \div K2$
K2PREDIV	5:4	rw	**PLL0 pre-divider select** This bit-field defines the pre-dividing factor (*K2PRE*) in front of the K2-divider. The effective value of K2PRE is listed below. $f_{PC0} = f_{DCO} \div K2PRE$ ***Note**: The configuration sequence for the K-dividers described in the functional description chapter must be followed in order to change the value of this bitfield, otherwise frequency glitches may occur.* 00_B **DIVBY1_0**: K2PRE = 1.0 ; this value must be used for normal operation 01_B **DIVBY2_0**: K2PRE = 2.0 (reserved)
K3DIV	10:8	rw	**K3-divider value** This bit-field defines the dividing factor of the K3-divider (*K3*). The effective value is K3 = K3DIV+1. $f_{PPU} = f_{PCPPU} \div K3$ ***Note**: Only present in devices with an asynchronous PPU. Please refer to the device specific chapters for availability.*
K3PREDIV	15:12	rw	**PLLPPU pre-divider select** This bit-field defines the pre-dividing factor (*K3PRE*) in front of the K3-divider. The effective value of K3PRE is listed below. $f_{PCPPU} = f_{DCO} \div K3PRE$ ***Note**: Only present in devices with an asynchronous PPU. Please refer to the device specific chapters for availability.* ***Note**: The configuration sequence for the K-dividers described in the functional description chapter must be followed in order to change the value of this bitfield, otherwise frequency glitches may occur.* 0_H **DIVBY1_0**: K3PRE = 1.0 1_H **DIVBY1_1**: K3PRE = 1.1 2_H **DIVBY1_2**: K3PRE = 1.2 4_H **DIVBY1_4**: K3PRE = 1.4 6_H **DIVBY1_6**: K3PRE = 1.6 7_H **DIVBY1_7**: K3PRE = 1.7 A_H **DIVBY2_0**: K3PRE = 2.0 **others,** configurations not present in this list act as the 0_H configuration.

(table continues...)

(continued)

Field	Bits	Type	Description
0	7:6, 11, 31:16	r	**Reserved** Read as reset value; shall be written with reset value.

Table 1305 Access mode restrictions of SYSPLLCON1 sorted by descending priority

Mode name	Access mode		Description
CCUSTAT.LCK == 0	rw	K2DIV, K2PREDIV, K3DIV, K3PREDIV	Normal access
(default)	r	K2DIV, K2PREDIV, K3DIV, K3PREDIV	Limited access

32.4.25 System-PLL configuration register 2

SYSPLLCON2 Offset address: 0308_H

System-PLL configuration register 2 System Reset value: $0000\ 6000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	MODCFG														
r	rw														

Field	Bits	Type	Description
MODCFG	14:0	rw	**Modulation Configuration** This bit-field defines the modulation. MODCFG[9:0] defines the modulation amplitude. Bits MODCFG[14:10] have to be configured with the following setting: 11101_B.
0	31:15	r	**Reserved** Read as 0; must be written with 0.

Table 1306 Access mode restrictions of SYSPLLCON2 sorted by descending priority

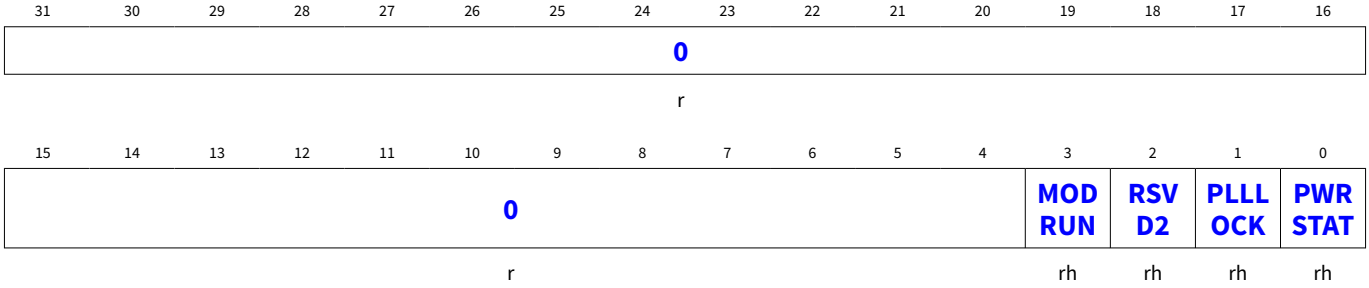
Mode name	Access mode		Description
CCUSTAT.LCK == 0	rw	MODCFG	Normal access
(default)	r	MODCFG	Limited access

32.4.26 System-PLL status register

These registers reflects the settings of the System-PLL.

SYSPLLSTAT Offset address: $030C_H$

System-PLL status register System Reset value: $0000\ 0000_H$



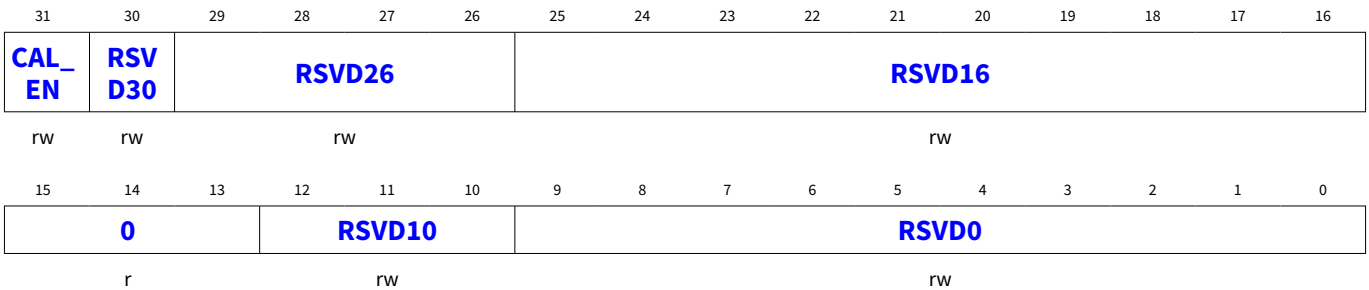
Field	Bits	Type	Description
PWRSTAT	0	rh	**System-PLL power mode status** 0_B System-PLL is powered down 1_B System-PLL is powered up
PLLLOCK	1	rh	**System-PLL Lock status** ***Note**:* *In case of a loss of lock, the f_{DCO} is kept on the previous constant frequency.* 0_B The frequency of the System-PLL is not stable and does not enable system operation 1_B The frequency of the System-PLL is stable and enables system operation
RSVD2	2	rh	**Reserved** Read as 0.
MODRUN	3	rh	**System-PLL Modulation run** This bit indicates if the frequency modulation of the System-PLL is activated or not. 0_B Frequency modulation is not active 1_B Frequency modulation is active
0	31:4	r	**Reserved** Read as 0; must be written with 0.

32.4.27 System-PLL modulation trim configuration register 0

This register defines the software interface for (optional) modulation amplitude trimming. The complete feature is gated by bit SYSPLLCON0.MODEN. If this bit is cleared, this feature is disabled.

SMODCON0 Offset address: 0340_H

System-PLL modulation trim configuration register 0 PowerOn Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
RSVD0	9:0	rw	**Reserved** Read as reset value; shall be written with reset value.
RSVD10	12:10	rw	**Reserved** Read as reset value; shall be written with reset value.
RSVD16	25:16	rw	**Reserved** Read as reset value; shall be written with reset value.
RSVD26	29:26	rw	**Reserved** Read as reset value; shall be written with reset value.
RSVD30	30	rw	**Reserved** Read as reset value; shall be written with reset value.
CAL_EN	31	rw	**Calibration enable** This bit-field is required to enable the frequency modulation for SYSPLL
0	15:13	r	**Reserved** Read as 0; must be written with 0.

Table 1307 Access mode restrictions of SMODCON0 sorted by descending priority

Mode name	Access mode		Description
.**FLLLDEN** == 0	rw	RSVD0 , RSVD10 , RSVD16 , RSVD26 , RSVD30	Normal access
(default)	r	RSVD0 , RSVD10 , RSVD16 , RSVD26 , RSVD30	Limited access

32.4.28 Peripheral-PLL configuration register 0

***Note**: Writing to this register sets the CCUSTAT.LCK bit and automatically triggers an update transfer of the register value to the Peripheral-PLL.*

PERPLLCON0 Offset address: 0380_H

Peripheral-PLL configuration register 0 System Reset value: $0000\ 3E00_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0													PDIV		
r													rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	NDIV							0				MODEN	0	RESLD	PLLPWR
r	rw							r				rw	r	rwh	rw

Field	Bits	Type	Description
PLLPWR	0	rw	**Peripheral-PLL power mode** ***Note**: If the PLL has been powered down and is re-enabled by PLLPWR = 1, a wait period of 1 ms has to be applied until it is stable without jitter. After asserting this bit,the PERPLL power status shall be checked (PERPLLSTAT.PWRSTAT == 1) before the next PERPLL KDIV configuration.* ***Note**: A write to PERPLLCON0.PLLPWR bit will set CCUSTAT.LCK bit until the handshake of the K dividers is completed.* 0_B The complete Peripheral-PLL block is put into a power-down mode and can no longer be used. 1_B Powered-up, normal behavior
RESLD	1	rwh	**Restart DCO lock detection** Setting this bit clears bit PERPLLSTAT.LOCK and restarts the Peripheral-PLL's DCO lock detection. After asserting this bit, other bit-fields in this register shall not be modified before the PLL is locked. Reading this bit returns always a zero.
MODEN	3	rw	**Modulation enable** This bit controls the activation of the frequency modulation of the Peripheral-PLL. 0_B Frequency modulation is not activated 1_B Frequency modulation is activated
NDIV	14:8	rw	**N-divider value** The value the N-divider operates with is NDIV+1.
PDIV	18:16	rw	**P-divider value** The value the P-divider operates with is PDIV+1.
0	2, 7:4, 15, 31:19	r	**Reserved** Read as reset value; shall be written with reset value.

Table 1308 Access mode restrictions of PERPLLCON0 sorted by descending priority

Mode name	Access mode		Description
CCUSTAT.LCK == 0	rw	MODEN, NDIV, PDIV, PLLPWR	Normal access
	rwh	RESLD	
(default)	r	MODEN, NDIV, PDIV, PLLPWR	Limited access
	rh	RESLD	

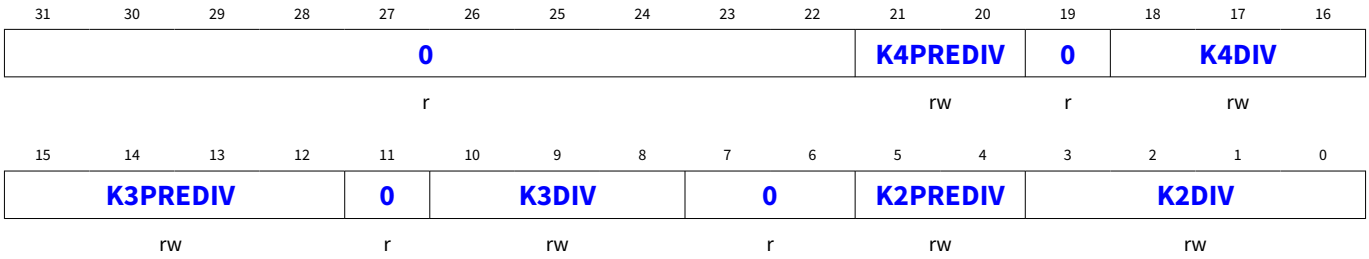
32.4.29 Peripheral-PLL configuration register 1

***Note**: Writing to this register sets the CCUSTAT.LCK bit and automatically triggers an update transfer of the register value to the Peripheral-PLL.*

Note: Please refer to the device specific register chapter for the reset values.

PERPLLCON1 Offset address: 0384_H

Peripheral-PLL configuration register 1 Reset value: Undefined



Field	Bits	Type	Description
K2DIV	3:0	rw	**K2-divider value** This bit-field defines the dividing factor of the K2-divider (*K2*). The effective value is K2 = K2DIV+1. $f_{PLL1} = f_{PC1} \div K2$
K2PREDIV	5:4	rw	**PLL1 pre-divider select** This bit-field defines the pre-dividing factor (*K2PRE*) in front of the K2-divider. The effective value of K2PRE is listed below. $f_{PC1} = f_{DCO} \div K2PRE$ ***Note**: The configuration sequence for the K-dividers described in the functional description chapter must be followed in order to change the value of this bitfield, otherwise frequency glitches may occur.* 00_B **DIVBY1_0**: K2PRE = 1.0 01_B **DIVBY2_0**: K2PRE = 2.0 10_B **DIVBY1_2**: K2PRE = 1.2 11_B **DIVBY1_6**: K2PRE = 1.6
K3DIV	10:8	rw	**K3-divider value** This bit-field defines the dividing factor of the K3-divider (*K3*). The effective value is K3 = K3DIV+1. $f_{PLL2} = f_{PC2} \div K3$

(table continues...)

(continued)

Field	Bits	Type	Description
K3PREDIV	15:12	rw	**PLL2 pre-divider select** This bit-field defines the pre-dividing factor (*K3PRE*) in front of the K3-divider. The effective value of K3PRE is listed below. $f_{PC2} = f_{DCO} \div K3PRE$ ***Note**: The configuration sequence for the K-dividers described in the functional description chapter must be followed in order to change the value of this bitfield, otherwise frequency glitches may occur.* 0_H **DIVBY1_0**: K3PRE = 1.0 1_H **DIVBY1_1**: K3PRE = 1.1 2_H **DIVBY1_2**: K3PRE = 1.2 4_H **DIVBY1_4**: K3PRE = 1.4 6_H **DIVBY1_6**: K3PRE = 1.6 7_H **DIVBY1_7**: K3PRE = 1.7 A_H **DIVBY2_0**: K3PRE = 2.0 **others,** configurations not present in this list act as the 0_H configuration.
K4DIV	18:16	rw	**K4-divider value** This bit-field defines the dividing factor of the K4-divider (*K4*). The effective value is K4 = K4DIV+1. $f_{PLL3} = f_{PC3} \div K4$
K4PREDIV	21:20	rw	**PLL3 pre-divider select** This bit-field defines the pre-dividing factor (*K4PRE*) in front of the K4-divider. The effective value of K4PRE is listed below. $f_{PC3} = f_{DCO} \div K4PRE$ ***Note**: The configuration sequence for the K-dividers described in the functional description chapter must be followed in order to change the value of this bitfield, otherwise frequency glitches may occur.* 00_B **DIVBY1_0**: K4PRE = 1.0 01_B **DIVBY2_0**: K4PRE = 2.0 10_B **DIVBY1_2**: K4PRE = 1.2 11_B **DIVBY1_6**: K4PRE = 1.6
0	7:6, 11, 19, 31:22	r	**Reserved** Read as reset value; shall be written with reset value.

Table 1309 Access mode restrictions of PERPLLCON1 sorted by descending priority

Mode name	Access mode		Description
CCUSTAT.LCK == 0	rw	K2DIV, K2PREDIV, K3DIV, K3PREDIV, K4DIV, K4PREDIV	Normal access

(table continues...)

Table 1309 (continued) Access mode restrictions of PERPLLCON1 sorted by descending priority

Mode name	Access mode		Description
(default)	r	K2DIV, K2PREDIV, K3DIV, K3PREDIV, K4DIV, K4PREDIV	Limited access

32.4.30 Peripheral-PLL configuration register 2

***Note**: Writing to this register sets the CCUSTAT.LCK bit and automatically triggers an update transfer of the register value to the Peripheral-PLL.*

PERPLLCON2 Offset address: 0388_H

Peripheral-PLL configuration register 2 System Reset value: $0001\ 6000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0														CLKSELSRC1	
r														rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	MODCFG														
r	rw														

Field	Bits	Type	Description
MODCFG	14:0	rw	**Modulation Configuration** This bit-field defines the modulation. MODCFG[9:0] defines the modulation amplitude. Bits MODCFG[9:5] are treated as integer part and bits MODCFG[4:0] as fractional part. Bits MODCFG[14:10] have to be configured with the following setting: 11101_B.

(table continues...)

(continued)

Field	Bits	Type	Description
CLKSELSRC1	17:16	rw	**Clock selection for fsource1** This bit-field defines the clock source that is used for the clock generation of $f_{source1}$ for measurement purposes. ***Notes**:* *For switching between two non-zero configurations the following sequence has to be applied:* ***1.** Switch to 00_B* ***2.** Switch to the new target configuration* ***Note**: f_{SPB} or $f_{PADSYSCLK}$ are not intended to be used during normal operation here! Details on how to set f_{SRC1} can be found in CCUCON.CLKSELP and in the **Distribution** chapter in the **Clock control unit overview** figure.* ***Note**: Use the value 00_B only to switch between clocks. Do not make any register accesses while the value is 00_B, as registers synchronized to the per160 clock domain will be affected, for example, PERCCUCON0 and PERCCUCON1.* 00_B $f_{source1}$ clock is stopped 01_B $f_{source1} = f_{SRC1}$ 10_B $f_{source1} = f_{SPB}$ 11_B $f_{source1} = f_{PADSYSCLK}$
0	15, 31:18	r	**Reserved** Read as reset value; shall be written with reset value.

Table 1310 Access mode restrictions of PERPLLCON2 sorted by descending priority

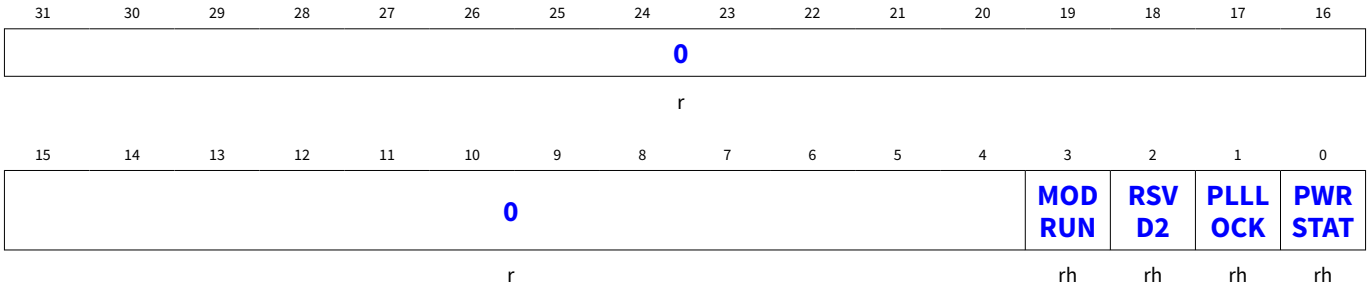
Mode name	Access mode		Description
CCUSTAT.LCK == 0	rw	CLKSELSRC1, MODCFG	Normal access
(default)	r	CLKSELSRC1, MODCFG	Limited access

32.4.31 Peripheral-PLL status register

These registers reflects the settings of the Peripheral-PLL.

PERPLLSTAT Offset address: $038C_H$

Peripheral-PLL status register System Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
PWRSTAT	0	rh	**Peripheral-PLL power mode status** 0_B Peripheral-PLL is powered down 1_B Peripheral-PLL is powered up
PLLLOCK	1	rh	**Peripheral-PLL lock status** ***Note:*** *In case of a loss of lock, the f_{DCO} is kept on the previous constant frequency.* 0_B The frequency of the Peripheral-PLL is not stable and does not enable system operation 1_B The frequency of the Peripheral-PLL is stable and enables system operation
RSVD2	2	rh	**Reserved** Read as 0.
MODRUN	3	rh	**Peripheral-PLL Modulation run** This bit indicates if the frequency modulation of the Peripheral-PLL is activated or not. 0_B Frequency modulation is not active 1_B Frequency modulation is active
0	31:4	r	**Reserved** Read as 0; must be written with 0.

32.4.32 Peripheral-PLL modulation trim configuration register 0

This register defines the software interface for (optional) modulation amplitude trimming. The complete feature is gated by bit PERPLLCON0.MODEN. If this bit is cleared, this feature is disabled.

Writing to this register sets the CCUSTAT.LCK bit and automatically triggers an update transfer of the register value to the Peripheral-PLL.

PMODCON0 Offset address: $03A0_H$

Peripheral-PLL modulation trim configuration register 0 PowerOn Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CAL_EN	RSVD30	RSVD26				RSVD16									
rw	rw	rw				rw									

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			RSVD10			RSVD0									
r			rw			rw									

Field	Bits	Type	Description
RSVD0	9:0	rw	**Reserved** Read as reset value; shall be written with reset value.

(table continues...)

(continued)

Field	Bits	Type	Description
RSVD10	12:10	rw	**Reserved** Read as reset value; shall be written with reset value.
RSVD16	25:16	rw	**Reserved** Read as reset value; shall be written with reset value.
RSVD26	29:26	rw	**Reserved** Read as reset value; shall be written with reset value.
RSVD30	30	rw	**Reserved** Read as reset value; shall be written with reset value.
CAL_EN	31	rw	**Calibration enable** This bit-field is required to enable the frequency modulation for PERPLL
0	15:13	r	**Reserved** Read as 0; must be written with 0.

Table 1311 Access mode restrictions of PMODCON0 sorted by descending priority

Mode name	Access mode		Description
CCUSTAT.LCK == 0 and .**FLLLDEN** == 0	rw	, CAL_EN , RSVD0 , RSVD10 , RSVD16 , RSVD26 , RSVD30	Normal access
(default)	r	, CAL_EN , RSVD0 , RSVD10 , RSVD16 , RSVD26 , RSVD30	Limited access

32.4.33 Global clock control register

This register contains control information for global clocking system functionality.

***Note**: It is not recommended to change the CCUCON configuration during a ramp-up or a ramp-down scenario.*

Note: Please refer to the device specific register chapter for the reset values.

CCUCON Offset address: 0400_H

Global clock control register Reset value: Undefined

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															CLKSELP
r															rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															CLKSELS
r															rw

Field	Bits	Type	Description
CLKSELS	1:0	rw	**Selection for system clock source** This bit-field selects the clock source that is used for the clock generation of $f_{source0}$ (and $f_{sourceppu}$ in devices with an asynchronous PPU). Reading these bits reflects the last written value by software. However, due to hardware controlled emergency switching and upcoming reset triggered by hardware, the actual used value has to be read from CCUSTAT.CLKSELS ***Note**: In general, before switching clock sources, the user has to check the availibility of the related source:* *f_{PLL0} shall only be chosen as clock source if SYSPLLSTAT.PWRSTAT = 1 and SYSPLLSTAT.LOCK = 1.* *f_{RAMP} (and $f_{RAMPHALF}$ in devices with an asynchronous PPU) shall only be chosen as clock source if RAMPSTAT.ACTIVE = 1 and RAMPSTAT.FLLOCK = 1.* 00_B **PLL**: f_{PLL0} is used as clock source $f_{source0}$ 01_B **BACK**: f_{BACK} is used as clock source $f_{source0}$ 10_B **RAMP**: f_{RAMP} is used as clock source $f_{source0}$; $f_{RAMPHALF}$ is used as clock source $f_{sourceppu}$ (only valid for devices with an asynchronous PPU) **others,** Hardware is busy
CLKSELP	16	rw	**Selection for peripheral clock sources** This bit-field selects the clock source that is used for the clock generation of f_{src1}, $f_{source2}$ and $f_{source3}$. Reading this bit reflects the last written value by software. However, due to hardware controlled emergency switching, the actual used value has to be read from CCUSTAT.CLKSELP ***Note**: In general, before switching clock sources, the user has to check the availibility of the related source:* *f_{src1} and $f_{PLL2,3}$ shall only be chosen as clock source if PERPLLSTAT.PWRSTAT = 1 and PERPLLSTAT.LOCK = 1.* *For measurement purposes, $f_{source1}$ can be driven by f_{SPB} or $f_{PADSYSCLK}$ controlled by PERPLLCON2.CLKSELSRC1. Additional details can be found in the **Distribution** chapter in the **Clock control unit overview** figure.* 0_B **PLL**: $f_{src1} = f_{PLL1}$; $f_{source2} = f_{PLL2}$; $f_{source3} = f_{PLL3}$ 1_B **BACK**: f_{BACK} is used as clock source for f_{src1} , $f_{source2}$, and $f_{source3}$
0	15:2, 31:17	r	**Reserved** Read as 0; must be written with 0.

Table 1312 Access mode restrictions of CCUCON sorted by descending priority

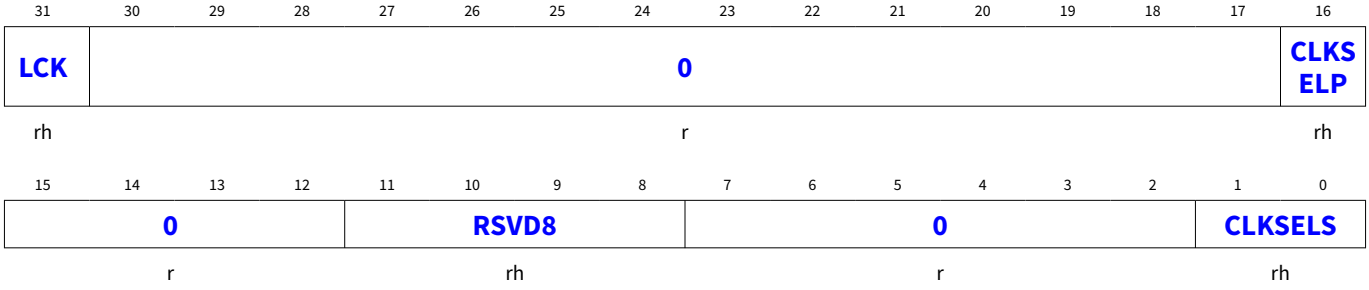
Mode name	Access mode		Description
CCUSTAT.LCK == 0	rw	CLKSELP, CLKSELS	Normal access
(default)	r	CLKSELP, CLKSELS	Limited access

32.4.34 Global clock status register

This register contains status information of central clock system functionality. This is register lock and main clock multiplexer settings.

CCUSTAT Offset address: 0404_H

Global clock status register Reset values see: Table 1313



Field	Bits	Type	Description
CLKSELS	1:0	rh	**Selection for system clock source** This bit-field reflects the actual clock source that is used for the clock generation of $f_{source0}$ (and $f_{sourceppu}$ in devices with an asynchronous PPU). 00_B **PLL**: f_{PLL0} is used as clock source $f_{source0}$ 01_B **BACK**: f_{BACK} is used as clock source $f_{source0}$ 10_B **RAMP**: f_{RAMP} is used as clock source $f_{source0}$; $f_{RAMPHALF}$ is used as clock source $f_{sourceppu}$ (only valid for devices with an asynchronous PPU) **others,**
RSVD8	11:8	rh	**Reserved** Read as 0.
CLKSELP	16	rh	**Selection for peripheral clock sources** This bit-field reflects the clock source that is used for the clock generation of f_{src1}, $f_{source2}$ and $f_{source3}$. 0_B **PLL**: $f_{src1} = f_{PLL1}$; $f_{source2} = f_{PLL2}$; $f_{source3} = f_{PLL3}$ 1_B **BACK**: f_{BACK} is used as clock source for f_{src1} , $f_{source2}$, and $f_{source3}$

(table continues...)

(continued)

Field	Bits	Type	Description
LCK	31	rh	**Register lock status** As the clocking system contains several registers that are updated by a posting mechanism, it is necessary to maintain data coherency using a "lock" while the posted write transaction is ongoing and the register write takes effect at the final destination. This bit indicates if the posting mechanism is active, that means ongoing and not finished yet. While it is set, write access to the respective registers will not be executed without further notification to the application. ***Note**: When set, expect a delay of up to 70µs for the CCUSTAT.LCK bit to clear.* ***Note**: For further details on how and when the lock is set and cleared, refer to the register chapter overview section.* 0_B No posting mechanism is active and clocking system registers can be accessed. 1_B The posting mechanism is active and clocking system registers shall not be accessed.
0	7:2, 15:12, 30:17	r	**Reserved** Read as 0; must be written with 0.

Table 1313 Reset values of CCUSTAT

Reset type	Reset value	Note
System Reset	$0001\ 0001_H$	
Config Sector value	$0000\ 0000_H$	
After Boot-FW value	–000 0000 0000 000– 0000 XXXX 0000 $00XX_B$	Value after application reset: 1) The bits marked “-“ remain the same as before the reset activation. 2) For CLKSELS: XX=01b in case of ramp-up oscillator failure, otherwise - the same as before the reset activation.

32.4.35 System CCU clock control register 0

Note: Please refer to the device specific register chapter for the reset values.

SYSCCUCON0 Offset address: 0410_H

System CCU clock control register 0 Reset value: Undefined

32 CLOCK

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	UP	SRICSDIV	LPDIV	STMDIV				0			FSI2DIV	FSIDIV			
r	w	rh	rw	rw				r			rw	rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SRIDIV				TPBDIV				CPBDIV				SPBDIV			
rw				rw				rw				rw			

Field	Bits	Type	Description
SPBDIV	3:0	rw	**SPB divider reload value** The resulting SPB frequency is configured to $f_{SPB} = f_{source0} \div$ SPBDIV for the allowed configurations as listed below. 0_H Reserved, do not use this combination 1_H $f_{SPB} = f_{source0} \div 1$ 2_H $f_{SPB} = f_{source0} \div 2$ 3_H $f_{SPB} = f_{source0} \div 3$ 4_H $f_{SPB} = f_{source0} \div 4$ 5_H $f_{SPB} = f_{source0} \div 5$ 6_H $f_{SPB} = f_{source0} \div 6$ 7_H Reserved, do not use this combination 8_H $f_{SPB} = f_{source0} \div 8$ 9_H Reserved, do not use this combination A_H $f_{SPB} = f_{source0} \div 10$ B_H Reserved, do not use this combination C_H $f_{SPB} = f_{source0} \div 12$ D_H Reserved, do not use this combination E_H Reserved, do not use this combination F_H $f_{SPB} = f_{source0} \div 15$

(table continues...)

(continued)

Field	Bits	Type	Description
CPBDIV	7:4	rw	**CPB divider reload value** The resulting CPB frequency is configured to $f_{CPB} = f_{source0} \div$ CPBDIV for the allowed configurations as listed below. After reset, the firmware will change the hardware reset value with the one defined in the compile time ConfigData. ***Note**: Please refer to the datasheet for availability* 0_H f_{CPB} is stopped ; Configuring CPBDIV=0 can lead to a bus error. 1_H $f_{CPB} = f_{source0} \div 1$ 2_H $f_{CPB} = f_{source0} \div 2$ 3_H $f_{CPB} = f_{source0} \div 3$ 4_H $f_{CPB} = f_{source0} \div 4$ 5_H $f_{CPB} = f_{source0} \div 5$ 6_H $f_{CPB} = f_{source0} \div 6$ 7_H Reserved, do not use this combination 8_H $f_{CPB} = f_{source0} \div 8$ 9_H Reserved, do not use this combination A_H $f_{CPB} = f_{source0} \div 10$ B_H Reserved, do not use this combination C_H $f_{CPB} = f_{source0} \div 12$ D_H Reserved, do not use this combination E_H Reserved, do not use this combination F_H $f_{CPB} = f_{source0} \div 15$
TPBDIV	11:8	rw	**TPB divider reload value** The resulting TPB frequency is configured to $f_{TPB} = f_{source0} \div$ TPBDIV for the allowed configurations as listed below. 0_H f_{TPB} is stopped 1_H $f_{TPB} = f_{source0} \div 1$ 2_H $f_{TPB} = f_{source0} \div 2$ 3_H $f_{TPB} = f_{source0} \div 3$ 4_H $f_{TPB} = f_{source0} \div 4$ 5_H $f_{TPB} = f_{source0} \div 5$ 6_H $f_{TPB} = f_{source0} \div 6$ 7_H Reserved, do not use this combination 8_H $f_{TPB} = f_{source0} \div 8$ 9_H Reserved, do not use this combination A_H $f_{TPB} = f_{source0} \div 10$ B_H Reserved, do not use this combination C_H $f_{TPB} = f_{source0} \div 12$ D_H Reserved, do not use this combination E_H Reserved, do not use this combination F_H $f_{TPB} = f_{source0} \div 15$

(table continues...)

(continued)

Field	Bits	Type	Description
SRIDIV	15:12	rw	**SRI divider reload value** The resulting SRI frequency is configured to $f_{SRI} = f_{source0} \div$ SRIDIV for the allowed configurations as listed below. 0_H Reserved, do not use this combination 1_H $f_{SRI} = f_{source0} \div 1$ 2_H $f_{SRI} = f_{source0} \div 2$ 3_H $f_{SRI} = f_{source0} \div 3$ 4_H $f_{SRI} = f_{source0} \div 4$ 5_H $f_{SRI} = f_{source0} \div 5$ 6_H $f_{SRI} = f_{source0} \div 6$ 7_H Reserved, do not use this combination 8_H $f_{SRI} = f_{source0} \div 8$ 9_H Reserved, do not use this combination A_H $f_{SRI} = f_{source0} \div 10$ B_H Reserved, do not use this combination C_H $f_{SRI} = f_{source0} \div 12$ D_H Reserved, do not use this combination E_H Reserved, do not use this combination F_H $f_{SRI} = f_{source0} \div 15$
FSIDIV	19:16	rw	**FSI divider reload value** The resulting FSI frequency is configured to $f_{FSI} = f_{source0} \div$ FSIDIV for the allowed configurations as listed below. ***Note**: The needed ratio of $f_{SRI} : f_{FSI}$ must be manually configured and is not automatically set by hardware* ***Note**:* 0_H Reserved, do not use this combination 1_H $f_{FSI} = f_{source0} \div 1$ 2_H $f_{FSI} = f_{source0} \div 2$ 3_H $f_{FSI} = f_{source0} \div 3$ 4_H $f_{FSI} = f_{source0} \div 4$ 5_H $f_{FSI} = f_{source0} \div 5$ 6_H $f_{FSI} = f_{source0} \div 6$ 7_H Reserved, do not use this combination 8_H $f_{FSI} = f_{source0} \div 8$ 9_H Reserved, do not use this combination A_H $f_{FSI} = f_{source0} \div 10$ B_H Reserved, do not use this combination C_H $f_{FSI} = f_{source0} \div 12$ D_H Reserved, do not use this combination E_H Reserved, do not use this combination F_H $f_{FSI} = f_{source0} \div 15$

(table continues...)

(continued)

Field	Bits	Type	Description
FSI2DIV	20	rw	**FSI2 divider reload value** The resulting FSI2 frequency is configured as listed below. 0_B Reserved, do not use this combination 1_B $f_{FSI2} = f_{SRI}$
STMDIV	27:24	rw	**STM divider reload value** The resulting STM frequency is configured to $f_{STM} = f_{source0} \div$ STMDIV for the allowed configurations as listed below. For STMDIV = 0000_B the clock is shut off. 0_H f_{STM} is stopped 1_H $f_{STM} = f_{source0} \div 1$ 2_H $f_{STM} = f_{source0} \div 2$ 3_H $f_{STM} = f_{source0} \div 3$ 4_H $f_{STM} = f_{source0} \div 4$ 5_H $f_{STM} = f_{source0} \div 5$ 6_H $f_{STM} = f_{source0} \div 6$ 7_H Reserved, do not use this combination 8_H $f_{STM} = f_{source0} \div 8$ 9_H Reserved, do not use this combination A_H $f_{STM} = f_{source0} \div 10$ B_H Reserved, do not use this combination C_H $f_{STM} = f_{source0} \div 12$ D_H Reserved, do not use this combination E_H Reserved, do not use this combination F_H $f_{STM} = f_{source0} \div 15$
LPDIV	28	rw	**Low Power divider reload value** This bit overwrites the current divider settings of the listed frequencies to enable a low power mode of operation. The selected LPDIV setting is valid for all clocks derived from f_{XXX} with XXX = SPB, SRI, SRICS, FSI, GETH, LETH, LETH100, CPB, RCB, TPB, GTM, eGTM, and STM. Do not change the PLL setting while in the low power mode. ***Note**:* *LPDIV only affects enabled clocks, that means XXXDIV = 0 will not be changed.* ***Note**:* *With an activated LPDIV the chip is set to a deep power down state (LPDIV=1). When a wake-up signal is received over the communication interfaces, it is possible to quickly return the chip to the "full power" state (LPDIV=0). Only valid in system mode.* 0_B f_{XXX} controlled by the related SYSCCUCON0 and SYSCCUCON1 bit-fields 1_B $f_{XXX} = f_{source0} \div 120$ (if previously enabled)

(table continues...)

(continued)

Field	Bits	Type	Description
SRICSDIV	29	rh	**SRI cyber security clock divider** This bit shows the actual setting of the SRICS frequencies. 0_B $f_{SRICS} = f_{SRI}$; 1_B $f_{SRICS} = f_{SRI} \div 2$;
UP	30	w	**Update request** Writing a '1' to this bit sets the CCUSTAT.LCK bit and requests an update transfer for SYSCCUCON0 and SYCCUCON1 register content to the system clock dividers. Only one UP bit must be set for either SYSCCUCON0 or SYSCCUCON1 to trigger the transfer. This bit always reads as zero. 0_B No action 1_B Start transfer
0	23:21, 31	r	**Reserved** Read as reset value; shall be written with reset value.

Table 1314 Access mode restrictions of SYSCCUCON0 sorted by descending priority

Mode name	Access mode		Description
CCUSTAT.LCK == 0	rw	CPBDIV, FSI2DIV, FSIDIV, LPDIV, SPBDIV, SRIDIV, STMDIV, TPBDIV	Normal access
	w	UP	
(default)	r	CPBDIV, FSI2DIV, FSIDIV, LPDIV, SPBDIV, SRIDIV, STMDIV, TPBDIV, UP	Limited access

32.4.36 System CCU clock control register 1

Note: Please refer to the device specific register chapter for the reset values.

SYSCCUCON1 Offset address: 0414_H

System CCU clock control register 1 Reset value: Undefined

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	**UP**	**0**		**CANXLHDIV**				**LETHDIV**				**MCANHDIV**			
r	w	r		rw				rw				rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
eGTMDIV				**GTMDIV**				**RCBDIV**				**GETHDIV**			
rw				rw				rw				rw			

Field	Bits	Type	Description
GETHDIV	3:0	rw	**GETH divider reload value** The resulting GETH frequency is configured to $f_{GETH} = f_{source0} \div$ GETHDIV for the allowed configurations as listed below. For GETHDIV = 0000_B the clock is shut off. ***Note**: GETHDIV must be enabled (!=0) during an application reset to allow firmware related installation tasks.* ***Note**: Please refer to the datasheet for availability* 0_H f_{GETH} is stopped 1_H $f_{GETH} = f_{source0} \div 1$ 2_H $f_{GETH} = f_{source0} \div 2$ 3_H $f_{GETH} = f_{source0} \div 3$ 4_H $f_{GETH} = f_{source0} \div 4$ 5_H Reserved, do not use this combination 6_H Reserved, do not use this combination 7_H Reserved, do not use this combination 8_H Reserved, do not use this combination 9_H Reserved, do not use this combination A_H Reserved, do not use this combination B_H Reserved, do not use this combination C_H Reserved, do not use this combination … E_H Reserved, do not use this combination F_H Reserved, do not use this combination

(table continues...)

(continued)

Field	Bits	Type	Description
RCBDIV	7:4	rw	**RCB divider reload value** The resulting RCB frequency is configured to $f_{RCB} = f_{source0} \div$ RCBDIV for the allowed configurations as listed below. ***Note**: Please refer to the datasheet for availability* 0_H f_{RCB} is stopped 1_H $f_{RCB} = f_{source0} \div 1$ 2_H $f_{RCB} = f_{source0} \div 2$ 3_H $f_{RCB} = f_{source0} \div 3$ 4_H $f_{RCB} = f_{source0} \div 4$ 5_H $f_{RCB} = f_{source0} \div 5$ 6_H $f_{RCB} = f_{source0} \div 6$ 7_H Reserved, do not use this combination 8_H $f_{RCB} = f_{source0} \div 8$ 9_H Reserved, do not use this combination A_H $f_{RCB} = f_{source0} \div 10$ B_H Reserved, do not use this combination C_H $f_{RCB} = f_{source0} \div 12$ D_H Reserved, do not use this combination E_H Reserved, do not use this combination F_H $f_{RCB} = f_{source0} \div 15$
GTMDIV	11:8	rw	**GTM divider reload value** The resulting GTM frequency is configured to $f_{GTM} = f_{source0} \div$ GTMDIV for the allowed configurations as listed below. ***Note**: Please refer to the datasheet for availability* 0_H f_{GTM} is stopped 1_H $f_{GTM} = f_{source0} \div 1$ 2_H $f_{GTM} = f_{source0} \div 2$ 3_H $f_{GTM} = f_{source0} \div 3$ 4_H $f_{GTM} = f_{source0} \div 4$ 5_H $f_{GTM} = f_{source0} \div 5$ 6_H $f_{GTM} = f_{source0} \div 6$ 7_H Reserved, do not use this combination 8_H $f_{GTM} = f_{source0} \div 8$ 9_H Reserved, do not use this combination A_H $f_{GTM} = f_{source0} \div 10$ B_H Reserved, do not use this combination C_H $f_{GTM} = f_{source0} \div 12$ D_H Reserved, do not use this combination E_H Reserved, do not use this combination F_H $f_{GTM} = f_{source0} \div 15$

(table continues...)

(continued)

Field	Bits	Type	Description
eGTMDIV	15:12	rw	**eGTM divider reload value** The resulting eGTM frequency is configured to $f_{eGTM} = f_{source0} \div$ eGTMDIV for the allowed configurations as listed below. ***Note**: Please refer to the datasheet for availability* 0_H f_{eGTM} is stopped 1_H $f_{eGTM} = f_{source0} \div 1$ 2_H $f_{eGTM} = f_{source0} \div 2$ 3_H $f_{eGTM} = f_{source0} \div 3$ 4_H $f_{eGTM} = f_{source0} \div 4$ 5_H $f_{eGTM} = f_{source0} \div 5$ 6_H $f_{eGTM} = f_{source0} \div 6$ 7_H Reserved, do not use this combination 8_H $f_{eGTM} = f_{source0} \div 8$ 9_H Reserved, do not use this combination A_H $f_{eGTM} = f_{source0} \div 10$ B_H Reserved, do not use this combination C_H $f_{eGTM} = f_{source0} \div 12$ D_H Reserved, do not use this combination E_H Reserved, do not use this combination F_H $f_{eGTM} = f_{source0} \div 15$
MCANHDIV	19:16	rw	**MCANH divider reload value** The resulting MCANH frequency is configured to $f_{MCANH} = f_{source0} \div$ MCANHDIV for the allowed configurations as listed below. For MCANHDIV = 0000_B the clock is shut off. ***Note**: Please refer to the datasheet for availability* 0_H f_{MCANH} is stopped 1_H $f_{MCANH} = f_{source0} \div 1$ 2_H $f_{MCANH} = f_{source0} \div 2$ 3_H $f_{MCANH} = f_{source0} \div 3$ 4_H $f_{MCANH} = f_{source0} \div 4$ 5_H $f_{MCANH} = f_{source0} \div 5$ 6_H $f_{MCANH} = f_{source0} \div 6$ 7_H Reserved, do not use this combination 8_H $f_{MCANH} = f_{source0} \div 8$ 9_H Reserved, do not use this combination A_H $f_{MCANH} = f_{source0} \div 10$ B_H Reserved, do not use this combination C_H $f_{MCANH} = f_{source0} \div 12$ D_H Reserved, do not use this combination E_H Reserved, do not use this combination F_H $f_{MCANH} = f_{source0} \div 15$

(table continues...)

(continued)

Field	Bits	Type	Description
LETHDIV	23:20	rw	**LETH divider reload value** The resulting LETH frequency is configured to $f_{LETH} = f_{source0} \div$ LETHDIV for the allowed configurations as listed below. For LETHDIV = 0000_B, the clock is shut off. ***Note**: Please refer to the datasheet for availability* 0_H f_{LETH} is stopped 1_H $f_{LETH} = f_{source0} \div 1$ 2_H $f_{LETH} = f_{source0} \div 2$ 3_H $f_{LETH} = f_{source0} \div 3$ 4_H $f_{LETH} = f_{source0} \div 4$ 5_H Reserved, do not use this combination 6_H Reserved, do not use this combination 7_H Reserved, do not use this combination 8_H Reserved, do not use this combination 9_H Reserved, do not use this combination A_H Reserved, do not use this combination B_H Reserved, do not use this combination C_H Reserved, do not use this combination … E_H Reserved, do not use this combination F_H Reserved, do not use this combination
CANXLHDIV	27:24	rw	**CANXLH divider reload value** The resulting CANXLH frequency is configured to $f_{CANXLH} = f_{source0} \div$ CANXLHDIV for the allowed configurations as listed below. For CANXLHDIV = 0000_B the clock is shut off. ***Note**: Please refer to the datasheet for availability* 0_H f_{CANXLH} is stopped 1_H $f_{CANXLH} = f_{source0} \div 1$ 2_H $f_{CANXLH} = f_{source0} \div 2$ 3_H $f_{CANXLH} = f_{source0} \div 3$ 4_H $f_{CANXLH} = f_{source0} \div 4$ 5_H $f_{CANXLH} = f_{source0} \div 5$ 6_H $f_{CANXLH} = f_{source0} \div 6$ 7_H Reserved, do not use this combination 8_H $f_{CANXLH} = f_{source0} \div 8$ 9_H Reserved, do not use this combination A_H $f_{CANXLH} = f_{source0} \div 10$ B_H Reserved, do not use this combination C_H $f_{CANXLH} = f_{source0} \div 12$ D_H Reserved, do not use this combination E_H Reserved, do not use this combination F_H $f_{CANXLH} = f_{source0} \div 15$

(table continues...)

(continued)

Field	Bits	Type	Description
UP	30	w	**Update request** Writing a '1' to this bit sets the CCUSTAT.LCK bit and requests an update transfer for SYSCCUCON0 and SYCCUCON1 register content to the system clock dividers. Only one UP bit must be set for either SYSCCUCON0 or SYSCCUCON1 to trigger the transfer. This bit always reads as zero. 0_B No action 1_B Start transfer
0	29:28, 31	r	**Reserved** Read as reset value; shall be written with reset value.

Table 1315 Access mode restrictions of SYSCCUCON1 sorted by descending priority

Mode name	Access mode		Description
CCUSTAT.LCK == 0	rw	CANXLHDIV, GETHDIV, GTMDIV, LETHDIV, MCANHDIV, RCBDIV, eGTMDIV	Normal access
	w	UP	
(default)	r	CANXLHDIV, GETHDIV, GTMDIV, LETHDIV, MCANHDIV, RCBDIV, UP, eGTMDIV	Limited access

32.4.37 Peripheral CCU clock control register 0

***Note**: Please refer to the device specific register chapter for the reset values.*

***Note**: Writing to this register sets the CCUSTAT.LCK bit and automatically triggers an update transfer of the register value to the peripheral clock divider logic.*

PERCCUCON0 Offset address: 0420_H

Peripheral CCU clock control register 0 Reset value: Undefined

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PPUDIV				I2CDIV				0		CLKSELQSPI		QSPIDIV			
rw				rw				r		rw		rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		CLKSELMSC		MSCDIV				0		CLKSELMCAN		MCANDIV			
r		rw		rw				r		rw		rw			

Field	Bits	Type	Description
MCANDIV	3:0	rw	**MCAN divider reload value** The intermediate MCAN frequency is configured to $f_{MCANI} = f_{source1} \div$ MCANDIV for the allowed configurations as listed below. For MCANDIV = 0000_B the clock is shut off. ***Note**: Please refer to the datasheet for availability* 0_H f_{MCANI} is stopped 1_H $f_{MCANI} = f_{source1}$ 2_H $f_{MCANI} = f_{source1} \div 2$ 3_H $f_{MCANI} = f_{source1} \div 3$ 4_H $f_{MCANI} = f_{source1} \div 4$ 5_H $f_{MCANI} = f_{source1} \div 5$ 6_H $f_{MCANI} = f_{source1} \div 6$ 7_H Reserved, do not use this combination 8_H $f_{MCANI} = f_{source1} \div 8$ 9_H Reserved, do not use this combination A_H $f_{MCANI} = f_{source1} \div 10$ B_H Reserved, do not use this combination C_H $f_{MCANI} = f_{source1} \div 12$ D_H Reserved, do not use this combination E_H Reserved, do not use this combination F_H $f_{MCANI} = f_{source1} \div 15$
CLKSELMCAN	5:4	rw	**Clock selection for MCAN** This bit-field defines the clock source that is used for the resulting clock generation of f_{mcan}. ***Note**: For switching between two non-zero configurations the following sequence has to be applied: First step is to switch to 00_B. Second step is to switch to the new target configuration.* ***Note**: Please refer to the datasheet for availability* 00_B f_{MCAN} clock is stopped 01_B f_{MCANI} is used as clock source f_{MCAN} 10_B f_{OSC} is used as clock source f_{MCAN} 11_B Reserved, do not use this combination

(table continues...)

(continued)

Field	Bits	Type	Description
MSCDIV	11:8	rw	**MSC divider reload value** The resulting MSC frequency is configured to $f_{MSC} = f_{sourcemsc} \div$ MSCDIV for the allowed configurations as listed below. For MSCDIV = 0000_B the clock is shut off. ***Note**: Please refer to the datasheet for availability* 0_H f_{MSC} is stopped 1_H $f_{MSC} = f_{sourcemsc}$ 2_H $f_{MSC} = f_{sourcemsc} \div 2$ 3_H $f_{MSC} = f_{sourcemsc} \div 3$ 4_H $f_{MSC} = f_{sourcemsc} \div 4$ 5_H $f_{MSC} = f_{sourcemsc} \div 5$ 6_H $f_{MSC} = f_{sourcemsc} \div 6$ 7_H Reserved, do not use this combination 8_H $f_{MSC} = f_{sourcemsc} \div 8$ 9_H Reserved, do not use this combination A_H $f_{MSC} = f_{sourcemsc} \div 10$ B_H Reserved, do not use this combination C_H $f_{MSC} = f_{sourcemsc} \div 12$ D_H Reserved, do not use this combination E_H Reserved, do not use this combination F_H $f_{MSC} = f_{sourcemsc} \div 15$
CLKSELMSC	13:12	rw	**Clock selection for MSC** This bit-field defines the clock source that is used for the clock generation of $f_{sourcemsc}$. ***Note**: For switching between two non-zero configurations the following sequence has to be applied: First step is to switch to 00_B. Second step is to switch to the new target configuration.* ***Note**: Please refer to the datasheet for availability* 00_B f_{MSC} clock is stopped 01_B $f_{source1}$ is used as clock source $f_{SOURCEMSC}$ 10_B $f_{source2}$ is used as clock source $f_{SOURCEMSC}$ 11_B Reserved, do not use this combination

(table continues...)

(continued)

Field	Bits	Type	Description
QSPIDIV	19:16	rw	**QSPI divider reload value** The resulting QSPI frequency is configured to $f_{QSPI} = f_{sourceqspi} \div$ QSPIDIV for the allowed configurations as listed below. For QSPIDIV = 0000_B the clock is shut off. 0_H f_{QSPI} is stopped 1_H $f_{QSPI} = f_{sourceqspi}$ 2_H $f_{QSPI} = f_{sourceqspi} \div 2$ 3_H $f_{QSPI} = f_{sourceqspi} \div 3$ 4_H $f_{QSPI} = f_{sourceqspi} \div 4$ 5_H $f_{QSPI} = f_{sourceqspi} \div 5$ 6_H $f_{QSPI} = f_{sourceqspi} \div 6$ 7_H Reserved, do not use this combination 8_H $f_{QSPI} = f_{sourceqspi} \div 8$ 9_H Reserved, do not use this combination A_H $f_{QSPI} = f_{sourceqspi} \div 10$ B_H Reserved, do not use this combination C_H $f_{QSPI} = f_{sourceqspi} \div 12$ D_H Reserved, do not use this combination E_H Reserved, do not use this combination F_H $f_{QSPI} = f_{sourceqspi} \div 15$
CLKSELQSPI	21:20	rw	**Clock selection for QSPI** This bit-field defines the clock source that is used for the clock generation of $f_{sourceqspi}$. ***Note**: For switching between two non-zero configurations the following sequence has to be applied: First step is to switch to 00_B. Second step is to switch to the new target configuration.* 00_B f_{QSPI} clock is stopped 01_B $f_{source1}$ is used as clock source $f_{SOURCEQSPI}$ 10_B $f_{source2}$ is used as clock source $f_{SOURCEQSPI}$ 11_B Reserved, do not use this combination

(table continues...)

(continued)

Field	Bits	Type	Description
I2CDIV	27:24	rw	**I2C divider reload value** The resulting I2C frequency is configured to $f_{I2C} = f_{SOURCE2} \div$ I2CDIV for the allowed configurations as listed below. For I2CDIV = 0000_B the clock is shut off. 0_H f_{I2C} is stopped 1_H $f_{I2C} = f_{source2} \div 1$ 2_H $f_{I2C} = f_{source2} \div 2$ 3_H $f_{I2C} = f_{source2} \div 3$ 4_H $f_{I2C} = f_{source2} \div 4$ 5_H $f_{I2C} = f_{source2} \div 5$ 6_H $f_{I2C} = f_{source2} \div 6$ 7_H Reserved, do not use this combination 8_H $f_{I2C} = f_{source2} \div 8$ 9_H Reserved, do not use this combination A_H $f_{I2C} = f_{source2} \div 10$ B_H Reserved, do not use this combination C_H $f_{I2C} = f_{source2} \div 12$ D_H Reserved, do not use this combination E_H Reserved, do not use this combination F_H $f_{I2C} = f_{source2} \div 15$
PPUDIV	31:28	rw	**PPU divider reload value** The resulting PPU frequency is configured to $f_{PPU} = f_{SOURCEPPU} \div$ PPUDIV for the allowed configurations as listed below. For PPUDIV = 0000_B the clock is shut off. ***Note**: Only present in devices with an asynchronous PPU. Please refer to the device specific chapters for avalability* 0_H f_{PPU} is stopped 1_H $f_{PPU} = f_{sourceppu} \div 1$ 2_H $f_{PPU} = f_{sourceppu} \div 2$ 3_H $f_{PPU} = f_{sourceppu} \div 3$ 4_H $f_{PPU} = f_{sourceppu} \div 4$ 5_H $f_{PPU} = f_{sourceppu} \div 5$ 6_H $f_{PPU} = f_{sourceppu} \div 6$ 7_H Reserved, do not use this combination 8_H $f_{PPU} = f_{sourceppu} \div 8$ 9_H Reserved, do not use this combination A_H $f_{PPU} = f_{sourceppu} \div 10$ B_H Reserved, do not use this combination C_H $f_{PPU} = f_{sourceppu} \div 12$ D_H Reserved, do not use this combination E_H Reserved, do not use this combination F_H $f_{PPU} = f_{sourceppu} \div 15$

(table continues...)

(continued)

Field	Bits	Type	Description
0	7:6, 15:14, 23:22	r	**Reserved** Read as 0; must be written with 0.

Table 1316 Access mode restrictions of PERCCUCON0 sorted by descending priority

Mode name	Access mode		Description
CCUSTAT.LCK == 0	rw	CLKSELMCAN, CLKSELMSC, CLKSELQSPI, I2CDIV, MCANDIV, MSCDIV, PPUDIV, QSPIDIV	Normal access
(default)	r	CLKSELMCAN, CLKSELMSC, CLKSELQSPI, I2CDIV, MCANDIV, MSCDIV, PPUDIV, QSPIDIV	Limited access

32.4.38 Peripheral CCU clock control register 1

***Note**: Please refer to the device specific register chapter for the reset values.*

***Note**: Writing to this register sets the CCUSTAT.LCK bit and automatically triggers an update transfer of the register value to the peripheral clock divider logic.*

PERCCUCON1 Offset address: 0424_H

Peripheral CCU clock control register 1 Reset value: Undefined

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0			LETH100PERON	HSCTPERON	SDMCPERON	XSPIPERON	ERAYPERON	CANXLDIV				0			ADCPERON
r			rw	rw	rw	rw	rw	rw				r			rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CLKSELCANXL		CLKSELASCLINS		ASCLINSDIV				XSPISLDIV				ASCLINFDIV			
rw		rw		rw				rw				rw			

Field	Bits	Type	Description
ASCLINFDIV	3:0	rw	**ASCLIN fast divider reload value** The resulting ASCLIN frequency is configured to $f_{ASCLINF} = f_{source2} \div$ ASCLINFDIV for the allowed configurations as listed below. For ASCLINFDIV = 0000_B the clock is shut off. 0_H $f_{ASCLINF}$ is stopped 1_H $f_{ASCLINF} = f_{source2} \div 1$ 2_H $f_{ASCLINF} = f_{source2} \div 2$ 3_H $f_{ASCLINF} = f_{source2} \div 3$ 4_H $f_{ASCLINF} = f_{source2} \div 4$ 5_H $f_{ASCLINF} = f_{source2} \div 5$ 6_H $f_{ASCLINF} = f_{source2} \div 6$ 7_H Reserved, do not use this combination 8_H $f_{ASCLINF} = f_{source2} \div 8$ 9_H Reserved, do not use this combination A_H $f_{ASCLINF} = f_{source2} \div 10$ B_H Reserved, do not use this combination C_H $f_{ASCLINF} = f_{source2} \div 12$ D_H Reserved, do not use this combination E_H Reserved, do not use this combination F_H $f_{ASCLINF} = f_{source2} \div 15$
XSPISLDIV	7:4	rw	**xSPISL divider reload value** The xSPISL (slow) frequency is configured to $f_{xSPISl} = f_{source3} \div$ XSPISLDIV for the allowed configurations as listed below. For XSPISLDIV = 00_B the clock is shut off. Please refer to the device specific register chapter for availability 0_H f_{xSPISL} is stopped 1_H $f_{xSPISL} = f_{source3} \div 1$ 2_H $f_{xSPISL} = f_{source3} \div 2$ 3_H Reserved, do not use this combination 4_H $f_{xSPISL} = f_{source3} \div 4$ 5_H Reserved, do not use this combination 6_H Reserved, do not use this combination 7_H Reserved, do not use this combination 8_H Reserved, do not use this combination 9_H Reserved, do not use this combination A_H Reserved, do not use this combination B_H Reserved, do not use this combination C_H Reserved, do not use this combination … E_H Reserved, do not use this combination F_H Reserved, do not use this combination

(table continues...)

(continued)

Field	Bits	Type	Description
ASCLINSDIV	11:8	rw	**ASCLIN slow divider reload value** The intermediate ASCLIN frequency is configured to $f_{ASCLINSI} = f_{source1} \div$ ASCLINSDIV for the allowed configurations as listed below. For ASCLINSDIV = 0000_B the clock is shut off. 0_H $f_{ASCLINSI}$ is stopped 1_H $f_{ASCLINSI} = f_{source1}$ 2_H $f_{ASCLINSI} = f_{source1} \div 2$ 3_H $f_{ASCLINSI} = f_{source1} \div 3$ 4_H $f_{ASCLINSI} = f_{source1} \div 4$ 5_H $f_{ASCLINSI} = f_{source1} \div 5$ 6_H $f_{ASCLINSI} = f_{source1} \div 6$ 7_H Reserved, do not use this combination 8_H $f_{ASCLINSI} = f_{source1} \div 8$ 9_H Reserved, do not use this combination A_H $f_{ASCLINSI} = f_{source1} \div 10$ B_H Reserved, do not use this combination C_H $f_{ASCLINSI} = f_{source1} \div 12$ D_H Reserved, do not use this combination E_H Reserved, do not use this combination F_H $f_{ASCLINSI} = f_{source1} \div 15$
CLKSELASCLINS	13:12	rw	**Clock selection for ASCLINS** This bit-field defines the clock source that is used for the resulting clock generation of $f_{ASCLINS}$. ***Note**: For switching between two non-zero configurations the following sequence has to be applied: First step is to switch to 00_B. Second step is to switch to the new target configuration.* 00_B $f_{ASCLINS}$ clock is stopped 01_B $f_{ASCLINSI}$ is used as clock $f_{ASCLINS}$ 10_B f_{OSC} is used as clock $f_{ASCLINS}$ 11_B Reserved, do not use this combination
CLKSELCANXL	15:14	rw	**Clock selection for CANXL** This bit-field defines the clock source that is used for the resulting clock generation of f_{canxl}. ***Note**: For switching between two non-zero configurations the following sequence has to be applied: First step is to switch to 00_B. Second step is to switch to the new target configuration.* 00_B f_{CANXL} clock is stopped 01_B f_{CANXLI} is used as clock source f_{CANXL} 10_B f_{OSC} is used as clock source f_{CANXL} 11_B Reserved, do not use this combination

(table continues...)

(continued)

Field	Bits	Type	Description
ADCPERON	16	rw	**Power safe switch-off for ADC clock** This bit is used to control the ADC peripheral clock f_{ADC} for power saving purposes if the logic is not used by the application. 0_B f_{ADC} is stopped 1_B $f_{ADC} = f_{source1}$
CANXLDIV	23:20	rw	**CANXL divider reload value** The intermediate CANXL frequency is configured to $f_{CANXLI} = f_{source1} \div$ CANXLDIV for the allowed configurations as listed below. For CANXLDIV = 0000_B the clock is shut off. 0_H f_{CANXLI} is stopped 1_H $f_{CANXLI} = f_{source1}$ 2_H $f_{CANXLI} = f_{source1} \div 2$ 3_H $f_{CANXLI} = f_{source1} \div 3$ 4_H $f_{CANXLI} = f_{source1} \div 4$ 5_H $f_{CANXLI} = f_{source1} \div 5$ 6_H $f_{CANXLI} = f_{source1} \div 6$ 7_H Reserved, do not use this combination 8_H $f_{CANXLI} = f_{source1} \div 8$ 9_H Reserved, do not use this combination A_H $f_{CANXLI} = f_{source1} \div 10$ B_H Reserved, do not use this combination C_H $f_{CANXLI} = f_{source1} \div 12$ D_H Reserved, do not use this combination E_H Reserved, do not use this combination F_H $f_{CANXLI} = f_{source1} \div 15$
ERAYPERON	24	rw	**Power safe switch-off for ERAY clock** This bit is used to control the ERAY peripheral clock f_{ERAY} for power saving purposes if the logic is not used by the application. ***Note**: Please refer to the datasheet for availability* 0_B f_{ERAY} is stopped 1_B $f_{ERAY} = f_{source1} \div 2$
XSPIPERON	25	rw	**Power safe switch-off for xSPI clock** This bit is used to control the xSPI peripheral clock f_{xSPI} for power saving purposes if the logic is not used by the application. ***Note**: Please refer to the datasheet for availability* 0_B f_{XSPI} is stopped 1_B $f_{XSPI} = f_{source3}$

(table continues...)

(continued)

Field	Bits	Type	Description
SDMMCPERON	26	rw	**Power safe switch-off for SDMMC clock** This bit is used to control the SDMMC peripheral clock f_{SDMMC} for power saving purposes if the logic is not used by the application. ***Note:*** *Please refer to the datasheet for availability* 0_B f_{SDMMC} is stopped 1_B $f_{SDMMC} = f_{source2} \div 2$
HSCTPERON	27	rw	**HSTC IVR power down** Switch off pre-regulator for HSCT ***Note:*** *Please refer to the datasheet for availability* 0_B Pre-regulator powered down 1_B Pre-regulator powered up
LETH100PERON	28	rw	**Power safe switch-off for LETH100 clock** This bit is used to control the LETH reference clock $f_{LETH100}$ for power saving purposes if the logic is not used by the application. ***Note:*** *Please refer to the datasheet for availability* 0_B $f_{LETH100}$ is stopped 1_B $f_{LETH100} = f_{source2}$
0	19:17, 31:29	r	**Reserved** Read as 0; must be written with 0.

Table 1317 Access mode restrictions of PERCCUCON1 sorted by descending priority

Mode name	Access mode		Description
CCUSTAT.LCK == 0	rw	ADCPERON, ASCLINFDIV, ASCLINSDIV, CANXLDIV, CLKSELASCLINS, CLKSELCANXL, ERAYPERON, LETH100PERON, SDMMCPERON, XSPIPERON, XSPISLDIV	Normal access
(default)	r	ADCPERON, ASCLINFDIV, ASCLINSDIV, CANXLDIV, CLKSELASCLINS, CLKSELCANXL, ERAYPERON, LETH100PERON, SDMMCPERON, XSPIPERON, XSPISLDIV	Limited access

32.4.39 External clock control register

***Note**: This register is used to control the external clock routing logic.*

EXTCON Offset address: 0430_H

External clock control register System Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	**DIV1**							**0**	**SEL1**					**NSEL1**	**EN1**
r	rw							r	rw					rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0									**SEL0**					**0**	**EN0**
r									rw					r	rw

Field	Bits	Type	Description
EN0	0	rw	**Output enable for EXTCLK0** ***Note**: If the generation of the external clock on output for pin EXTCLK0 is disabled, the signal is tied to zero.* 0_B No external clock is provided on output. 1_B The configured external clock is provided on output.

(table continues...)

(continued)

Field	Bits	Type	Description
SEL0	6:2	rw	**Clock select for EXTCLK0** This bit-field defines the clock source that is selected as output for pin EXTCLK0. 00_H f_{BACK} 01_H f_{RAMP} 02_H f_{OSC} 03_H f_{PLL0} 04_H f_{PLL1} 05_H f_{PLL2} 06_H f_{PLL3} (Only present in devices with an asynchronous PPU. Please refer to the device specific chapters for availability.) 07_H f_{FRAC} 08_H f_{SPB} 09_H f_{CPB} (Please refer to the datasheet for availability) $0A_H$ f_{TPB} $0B_H$ f_{SRI} $0C_H$ f_{FSI} $0E_H$ f_{STM} $0F_H$ f_{GTM} (Please refer to the datasheet for availability) 10_H f_{eGTM} (Please refer to the datasheet for availability) 11_H f_{GETH} (Please refer to the datasheet for availability) 12_H f_{MCANH} (Please refer to the datasheet for availability) 15_H f_{RCB} (Please refer to the datasheet for availability) 16_H f_{LETH} (Please refer to the datasheet for availability) 17_H f_{PPU} (Only present in devices with an asynchronous PPU. Please refer to the device specific chapters for availability.) 18_H f_{CANXLH} (Please refer to the datasheet for availability) **others,** Reserved, do not use
EN1	16	rw	**Output enable for EXTCLK1** ***Note**:* *If the generation of the external clock on output for pin EXTCLK1 is disabled, the signal is tied to zero.* 0_B No external clock is provided on output. 1_B The configured external clock is provided on output.
NSEL1	17	rw	**Negation selection for EXTCLK1** Selects if the external clock output for pin EXTCLK1 is inverted or not. Note: NSEL1 is not applicable for all clocks signals of SEL1. Please refer to the device specific chapters for further details. 0_B The external clock output is inverted. 1_B The external clock output is not inverted

(table continues...)

(continued)

Field	Bits	Type	Description
SEL1	22:18	rw	**External clock select for EXTCLK1** This bit-field defines the clock source that is selected as output for pin EXTCLK1. 00_H f_{BACK} 01_H f_{RAMP} 02_H f_{OSC} 03_H f_{PLL0} 04_H f_{PLL1} 05_H f_{PLL2} 06_H f_{PLL3} (Only present in devices with an asynchronous PPU. Please refer to the device specific chapters for availability.) 07_H f_{DIV1} 09_H f_{MT0} from the ERAY module (Please refer to the datasheet for availability) $0A_H$ f_{ERAY} (Please refer to the datasheet for availability) $0B_H$ f_{MCAN} (Please refer to the datasheet for availability) $0C_H$ f_{ADC} $0D_H$ f_{QSPI} $0E_H$ f_{MSC} (Please refer to the datasheet for availability) $0F_H$ f_{I2C} 10_H $f_{ASCLINF}$ 11_H $f_{ASCLINS}$ 13_H f_{XSPI} (Please refer to the datasheet for availability) 14_H f_{SDMMC} (Please refer to the datasheet for availability) 15_H f_{PPU} (Only present in devices with an asynchronous PPU. Please refer to the device specific chapters for availability.) 16_H f_{CANXL} (Please refer to the datasheet for availability) 17_H $f_{LETH100}$ (Please refer to the datasheet for availability) 18_H f_{XSPISL} (Please refer to the datasheet for availability) **others,** Reserved, do not use
DIV1	30:24	rw	**Clock divider for EXTCLK1** This value defines the reload value of the divider that generates f_{DIV1} out of f_{SPB} (f_{OUT}=f_{SPB}÷(DIV1+1). The divider itself is cleared each time bit EN1 is cleared.
0	1, 15:7, 23, 31	r	**Reserved** Read as 0; must be written with 0.

32.4.40 Fractional divider control register

EXTFDCON Offset address: 0434_H

Fractional divider control register System Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DISCLK	0	DM		0		STEP									
rw	r	rw		r		rw									

Field	Bits	Type	Description
STEP	9:0	rw	**Step value** In Normal divider mode, STEP contains the reload value for RESULT. In Fractional divider Mode, this bit-field determines the 10-bit value that is added to RESULT with each input clock cycle.
DM	13:12	rw	**Divider mode** These bit-fields determine the functionality of the fractional divider block. 00_B **Disabled**: Fractional divider is switched off; no output clock is generated. 01_B **Normal**: Normal divider mode selected. 10_B **Fractional**: Fractional divider Mode selected. **others,** Same as disabled
DISCLK	15	rw	**Disable clock** 0_B Clock generation of f_{OUT} is enabled according to the setting of bit-field DM. 1_B Fractional divider is stopped.
0	11:10, 14, 31:16	r	**Reserved** Read as 0; must be written with 0.

32.4.41 Global clock monitor register

CCUMON Offset address: 0500_H

Global clock monitor register System Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		KDVPPUMONTST	PLLPPUMONTST	PDIVMONTST	SDIVMONTST	RAMPMONTST	BACKMONTST	KDV3MONTST	KDV2MONTST	KDV1MONTST	KDV0MONTST	PLL3MONTST	PLL2MONTST	PLL1MONTST	PLL0MONTST
r		rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		KDVPPUMONEN	PLLPPUMONEN	PDIVMONEN	SDIVMONEN	RAMPMONEN	BACKMONEN	KDV3MONEN	KDV2MONEN	KDV1MONEN	KDV0MONEN	PLL3MONEN	PLL2MONEN	PLL1MONEN	PLL0MONEN
r		rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
PLL0MONEN	0	rw	**PLL0 clock alive monitor enable** This bit controls the PLL0 output's clock alive monitor. 0_B Monitoring is disabled 1_B Monitoring is enabled.
PLL1MONEN	1	rw	**PLL1 clock alive monitor enable** This bit controls the PLL1 output's clock alive monitor. 0_B Monitoring is disabled 1_B Monitoring is enabled.
PLL2MONEN	2	rw	**PLL2 clock alive monitor enable** This bit controls the PLL2 output's clock alive monitor. 0_B Monitoring is disabled 1_B Monitoring is enabled.
PLL3MONEN	3	rw	**PLL3 clock alive monitor enable** This bit controls the PLL3 output's clock alive monitor. 0_B Monitoring is disabled 1_B Monitoring is enabled.
KDV0MONEN	4	rw	**PLL0 K-divider monitor enable** This bit controls the PLL0 K2-divider monitor. 0_B Monitoring is disabled 1_B Monitoring is enabled.
KDV1MONEN	5	rw	**PLL1 K-divider monitor enable** This bit controls the PLL1 K2-divider monitor. 0_B Monitoring is disabled 1_B Monitoring is enabled.
KDV2MONEN	6	rw	**PLL2 K-divider monitor enable** This bit controls the PLL2 K3-divider monitor. 0_B Monitoring is disabled 1_B Monitoring is enabled.
KDV3MONEN	7	rw	**PLL3 K-divider monitor enable** This bit controls the PLL3 K4-divider monitor. 0_B Monitoring is disabled 1_B Monitoring is enabled.
BACKMONEN	8	rw	**EVR oscillator clock alive monitor enable** This bit controls the EVR's 100 MHz output clock alive monitor. 0_B Monitoring is disabled 1_B Monitoring is enabled.
RAMPMONEN	9	rw	**Ramp clock monitor enable** This bit controls the Ramp Oscillator's output clock alive monitor. 0_B Monitoring is disabled 1_B Monitoring is enabled.

(table continues...)

(continued)

Field	Bits	Type	Description
SDIVMONEN	10	rw	**System clock divider monitor enable** This bit controls the monitor that checks system clock divider outputs. 0_B Monitoring is disabled 1_B Monitoring is enabled.
PDIVMONEN	11	rw	**Peripheral clock divider monitor enable** This bit controls the monitor that checks peripheral clock and PPU clock divider outputs. 0_B Monitoring is disabled 1_B Monitoring is enabled.
PLLPPUMONEN	12	rw	**PLLPPU clock alive monitor enable** This bit controls the PLLPPU output's clock alive monitor. ***Note**: (Only present in devices with an asynchronous PPU. Please refer to the device specific chapters for availability.)* 0_B Monitoring is disabled 1_B Monitoring is enabled.
KDVPPUMONEN	13	rw	**PLLPPU K-divider monitor enable** This bit controls the PLLPPU K3-divider monitor. ***Note**: (Only present in devices with an asynchronous PPU. Please refer to the device specific chapters for availability.)* 0_B Monitoring is disabled 1_B Monitoring is enabled.
PLL0MONTST	16	rw	**PLL0 clock alive monitor test** This bit is used to test the PLL0 clock alive monitoring logic and the related alarm forwarding. 0_B normal operation 1_B Inhibit f_{PLL0} at monitor input. It may take a full monitor reference count period (512 x f_{BACK} cycles) until an alarm is generated. This depends on the selected PLL frequencies.
PLL1MONTST	17	rw	**PLL1 clock alive monitor test** This bit is used to test the PLL1 clock alive monitoring logic and the related alarm forwarding. 0_B normal operation 1_B Inhibit f_{PLL1} at monitor input. It may take a full monitor reference count period (512 x f_{BACK} cycles) until an alarm is generated. This depends on the selected PLL frequencies.

(table continues...)

(continued)

Field	Bits	Type	Description
PLL2MONTST	18	rw	**PLL2 clock alive monitor test** This bit is used to test the PLL2 clock alive monitoring logic and the related alarm forwarding. 0_B normal operation 1_B Inhibit f_{PLL2} at monitor input. It may take a full monitor reference count period (512 x f_{BACK} cycles) until an alarm is generated. This depends on the selected PLL frequencies.
PLL3MONTST	19	rw	**PLL3 clock alive monitor test** This bit is used to test the PLL3 clock alive monitoring logic and the related alarm forwarding. 0_B normal operation 1_B Inhibit f_{PLL3} at monitor input. It may take a full monitor reference count period (512 x f_{BACK} cycles) until an alarm is generated. This depends on the selected PLL frequencies.
KDV0MONTST	20	rw	**PLL0 K-divider monitor test** This bit is used to test the PLL0 K2-divider monitoring logic and the related alarm forwarding. 0_B normal operation 1_B Force comparison error.
KDV1MONTST	21	rw	**PLL1 K-divider monitor test** This bit is used to test the PLL1 K2-divider monitoring logic and the related alarm forwarding. 0_B normal operation 1_B Force comparison error.
KDV2MONTST	22	rw	**PLL2 K-divider monitor test** This bit is used to test the PLL2 K3-divider monitoring logic and the related alarm forwarding. 0_B normal operation 1_B Force comparison error.
KDV3MONTST	23	rw	**PLL3 K-divider monitor test** This bit is used to test the PLL3 K4-divider monitoring logic and the related alarm forwarding. 0_B normal operation 1_B Force comparison error.
BACKMONTST	24	rw	**Backup clock monitor test** This bit is used to test the Backup clock alive monitoring logic and the related alarm forwarding. 0_B normal operation 1_B Inhibit f_{BACK} at monitor input. It may take a full monitor reference count period (512 x f_{PLL0} cycles) until an alarm is generated.

(table continues...)

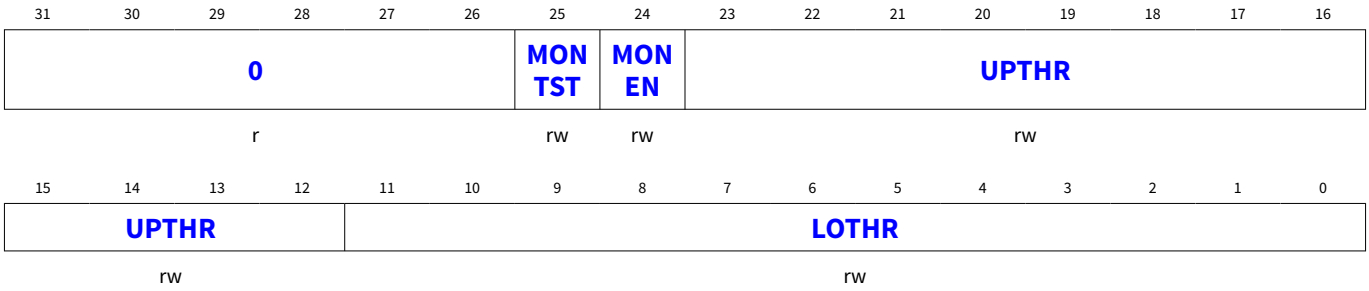
(continued)

Field	Bits	Type	Description
RAMPMONTST	25	rw	**Ramp clock monitor test** This bit is used to test the Ramp clock alive monitoring logic and the related alarm forwarding. 0_B normal operation 1_B Inhibit f_{RAMP} at monitor input. It may take a full monitor reference count period (512 x f_{PLL0} cycles) until an alarm is generated.
SDIVMONTST	26	rw	**System clock divider monitor test** This bit is used to test the system clock divider monitor and the related alarm forwarding. 0_B normal operation 1_B Force comparison error.
PDIVMONTST	27	rw	**Peripheral clock divider monitor test** This bit is used to test the peripheral clock divider monitor and the related alarm forwarding. 0_B normal operation 1_B Force comparison error.
PLLPPUMONTST	28	rw	**PLLPPU clock alive monitor test** This bit is used to test the PLLPPU clock alive monitoring logic and the related alarm forwarding. ***Note**: (Only present in devices with an asynchronous PPU. Please refer to the device specific chapters for availability.)* 0_B normal operation 1_B Inhibit f_{PLLPPU} at monitor input. It may take a full monitor reference count period (512 x f_{BACK} cycles) until an alarm is generated. This depends on the selected PLL frequencies.
KDVPPUMONTST	29	rw	**PLL3 K-divider monitor test** This bit is used to test the PLLPPU K3-divider monitoring logic and the related alarm forwarding. ***Note**: (Only present in devices with an asynchronous PPU. Please refer to the device specific chapters for availability.)* 0_B normal operation 1_B Force comparison error.
0	15:14, 31:30	r	**Reserved** Read as 0; must be written with 0.

32.4.42 EVR oscillator monitor register

OSCMON0 Offset address: 0504_H

EVR oscillator monitor register System Reset value: $0000\ 0000_H$

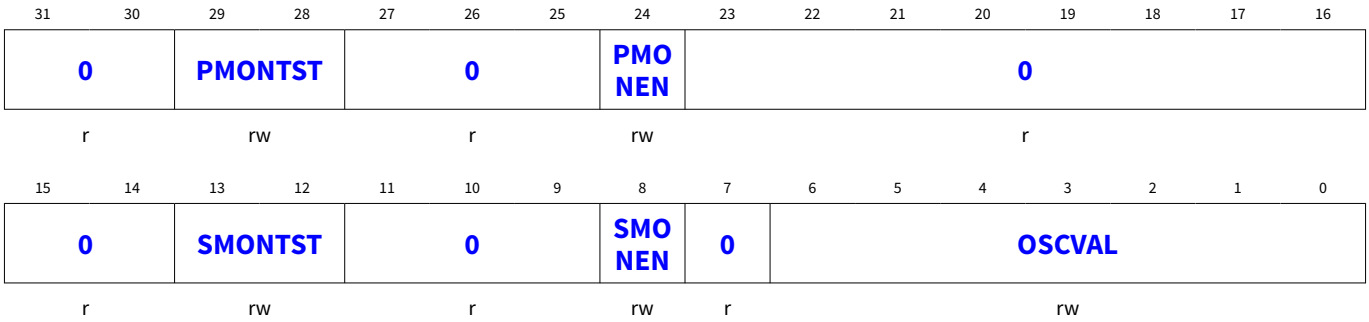


Field	Bits	Type	Description
LOTHR	11:0	rw	**Monitor lower threshold** lower threshold = 512÷f_{PLL0} x 0.9 x 100 MHz ***Note:*** *For proper operation and to avoid false alarms, the monitor needs to be disabled through MONEN=0 before programming the threshold values.*
UPTHR	23:12	rw	**Monitor upper threshold** upper threshold = 512÷f_{PLL0} x 1.1 x 100 MHz ***Note:*** *For proper operation and to avoid false alarms, the monitor needs to be disabled through MONEN=0 before programming the threshold values.*
MONEN	24	rw	**Monitor enable** 0_B Monitoring is disabled 1_B Monitoring is enabled
MONTST	25	rw	**Monitor test** Set this bit to 1 to test alarm generation. The test enable bit is a direct trigger for the alarm. 0_B Normal operation 1_B Test alarm will be generated
0	31:26	r	**Reserved** Read as 0; must be written with 0.

32.4.43 Crystal oscillator monitor register

OSCMON1 Offset address: 0508_H

Crystal oscillator monitor register System Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
OSCVAL	6:0	rw	**Reference input frequency value for crystal oscillator monitors** This bit-field defines the oscillator frequency reference value for the System- and Peripheral-PLL crystal oscillator monitors. Valid range is from 16 to 50 MHz, that means 1 to 35. f_{IN_REF} = (OSCVAL + 15) MHz ***Note**: For any other value set outside the valid range of f_{IN_REF}, the status of flags SOWDS and POWDS is undefined.*
SMONEN	8	rw	**System-PLL crystal oscillator monitor enable** This bit enables the System-PLL's crystal oscillator monitor. 0_B Monitoring is disabled and kept in reset. 1_B Monitoring is enabled.
SMONTST	13:12	rw	**System-PLL crystal oscillator monitor test** This bit is used to test the System-PLL's crystal oscillator monitoring logic and the related alarm forwarding. ***Note**: SMONEN must be set prior to triggering SMONTST.* 00_B **TestOff**: Normal operation 01_B **TestLV**: Overrides, that means minimizes the internal monitoring value to trigger the LV alarm. 10_B **TestHV**: Overrides, that means maximizes the internal monitoring value to trigger the HV alarm. **others,** Reserved, do not use.
PMONEN	24	rw	**Peripheral-PLL crystal oscillator monitor enable** This bit enables the Peripheral-PLL's crystal oscillator monitor. 0_B Monitoring is disabled and kept in reset. 1_B Monitoring is enabled.

(table continues...)

(continued)

Field	Bits	Type	Description
PMONTST	29:28	rw	**Peripheral-PLL crystal oscillator monitor test** This bit is used to test the Peripheral-PLL's crystal oscillator monitoring logic and the related alarm forwarding. ***Note:*** *PMONEN must be set prior to triggering PMONTST.* 00_B **TestOff**: Normal operation 01_B **TestLV**: Overrides, that means minimizes the internal monitoring value to trigger the LV alarm. 10_B **TestHV**: Overrides, that means maximizes the internal monitoring value to trigger the HV alarm. **others,** Reserved, do not use.
0	7, 11:9, 23:14, 27:25, 31:30	r	**Reserved** Read as 0; must be written with 0.

32.4.44 Crystal oscillator status register

OSCSTAT Offset address: $050C_H$

Crystal oscillator status register System Reset value: $0003\ 0003_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0											OAR	0		POWDS	
r											rh	r		rh	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0														SOWDS	
r														rh	

Field	Bits	Type	Description
SOWDS	1:0	rh	**Oscillator watchdog status for System-PLL** Thes bits indicate if the frequency input f_{IN} is in valid range to be usable for the DCO part of the System-PLL. This is checked by the crystal oscillator monitor (watchdog) of the System-PLL using the backup clock f_{BACK}. By using the reference frequency (f_{IN_REF}), the crystal oscillator's watchdog lower and upper limits calculate as follows: $f_{LV} = (f_{IN_REF} \times 0.75 - 0.31)$ MHz $f_{HV} = (f_{IN_REF} \times 1.46 + 0.29)$ MHz ***Note**: The OSCMON1.OSCVAL value must be set within its legal range (1 to 64) and the monitor must be enabled through OSCMON1.SMONEN = 1. After enabling, it takes a full measurement period plus evaluation (10 µs + 2 µs) until the output reflects a vaild result.* 00_B **InRange**: The OSC frequency is usable. Frequency f_{OSC} is in range: $f_{LV} < f_{OSC} < f_{HV}$ 01_B **TooLow**: The OSC frequency is not usable. Frequency f_{OSC} is too low: $f_{OSC} < f_{LV}$ 10_B **TooHigh**: The OSC frequency is not usable. Frequency f_{OSC} is too high: $f_{OSC} > f_{HV}$ 11_B **InValid**: System-PLL OWD Monitor Result not valid (yet). This is the default value after reset or when the monitor is disabled.
POWDS	17:16	rh	**Oscillator watchdog status for Peripheral-PLL** These bits indicate if the frequency input f_{IN} is in valid range to be usable for the DCO part of the Peripheral-PLL. This is checked by the crystal oscillator monitor (watchdog) of the Peripheral-PLL using the backup clock f_{BACK}. By using the reference frequency (f_{IN_REF}), the crystal oscillator's watchdog lower and upper limits calculate as follows: $f_{LV} = (f_{IN_REF} \times 0.75 - 0.31)$ MHz $f_{HV} = (f_{IN_REF} \times 1.46 + 0.29)$ MHz ***Note**: The OSCMON1.OSCVAL value must be set within its legal range (1 to 64) and the monitor must be enabled through OSCMON1.PMONEN = 1. After enabling, it takes a full measurement period plus evaluation (10 µs + 2 µs) until the output reflects a vaild result.* 00_B **InRange**: The OSC frequency is usable. Frequency f_{OSC} is in range: $f_{LV} < f_{OSC} < f_{HV}$ 01_B **TooLow**: The OSC frequency is not usable. Frequency f_{OSC} is too low: $f_{OSC} < f_{LV}$ 10_B **TooHigh**: The OSC frequency is not usable. Frequency f_{OSC} is too high: $f_{OSC} > f_{HV}$ 11_B **InValid**: Peripheral-PLL OWD Monitor Result not valid (yet). This is the default value after reset or when the monitor is disabled.

(table continues...)

(continued)

Field	Bits	Type	Description
OAR	20	rh	**Oscillator amplitude reached** Oscillator amplitude reached during negative resistance test. 0_B Oscillator amplitude not reached at XTAL1 pad (f_{XPADIN}) 1_B Oscillator amplitude reached at XTAL1 pad (f_{XPADIN})
0	15:2, 19:18, 31:21	r	**Reserved** Read as 0; must be written with 0.

32.4.45 Ramp oscillator monitor register

This register is used to control the frequency monitor of the ramp oscillator.

RAMPMON Offset address: 0510_H

Ramp oscillator monitor register System Reset value: $1020\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0			FLLLDEN	0			RESFLLLD	RSVD22		RSVD20		RSVD16			
r			rw	r			rwh	rw		rw		rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			RSVD12	RSVD10		RSVD8		RSVD0							
r			rw	rw		rw		rw							

Field	Bits	Type	Description
RSVD0	7:0	rw	**Reserved** Read as reset value; shall be written with reset value.
RSVD8	9:8	rw	**Reserved** Read as reset value; shall be written with reset value.
RSVD10	11:10	rw	**Reserved** Read as reset value; shall be written with reset value.
RSVD12	12	rw	**Reserved** Read as reset value; shall be written with reset value.
RSVD16	19:16	rw	**Reserved** Read as reset value; shall be written with reset value.
RSVD20	21:20	rw	**Reserved** Read as reset value; shall be written with reset value.
RSVD22	23:22	rw	**Reserved** Read as reset value; shall be written with reset value.

(table continues...)

(continued)

Field	Bits	Type	Description
RESFLLLD	24	rwh	**Restart DCO Lock Detection** Setting this bit clears bit RAMPSTAT.FLLLOCK and restarts the ramp oscillator's lock detection. Reading this bit returns always a zero.
FLLLDEN	28	rw	**Frequency lock detection enable** Frequency lock detection enable. 0_B Frequency monitor inactive and in reset 1_B Frequency monitor active
0	15:13, 27:25, 31:29	r	**Reserved** Read as 0; must be written with 0.

Table 1318 Access mode restrictions of RAMPMON sorted by descending priority

Mode name	Access mode		Description
RAMPMON.FMEN == 0 and **.FMEN** == 0 and **RAMPMON.FLLLDEN** == 0 and **.FLLLDEN** == 0	rw	, RSVD0 , RSVD10 , RSVD12 , RSVD16 , RSVD20 , RSVD22 , RSVD8	Normal access
(default)	r	, RSVD0 , RSVD10 , RSVD12 , RSVD16 , RSVD20 , RSVD22 , RSVD8	Limited access

32.4.46 Global clock monitor status register

MONSTAT Offset address: 0518_H

Global clock monitor status register PowerOn Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PDIV								**SDIV**							
rh								rh							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0					**KDV3**	**KDV2**	**KDV1**	**0**					**RES2**	**RES1**	**RES0**
r					rh	rh	rh	r					rh	rh	rh

Field	Bits	Type	Description
RES0	0	rh	**RES0** Reserved. Internal.
RES1	1	rh	**RES1** Reserved. Internal.

(table continues...)

(continued)

Field	Bits	Type	Description
RES2	2	rh	**RES2** Reserved. Internal.
KDV1	8	rh	**PLL1 K-divider monitor status** This bit stores the PLL1 K-divider monitor alarm.
KDV2	9	rh	**PLL2 K-divider monitor status** This bit stores the PLL2 K-divider monitor alarm.
KDV3	10	rh	**PLL3 K-divider monitor status** This bit stores the PLL3 K-divider monitor alarm.
SDIV	23:16	rh	**System clock divider monitor status** These bits store the alarms of the system clock divider monitor. Every bit represents one status bit from one DCCU. The test is successful only if all clusters' status bits are set upon the test. ***Note***: MONSTAT.*SDIV mirrors the alarm status of DCCUs, with each bit representing one DCCU instance. The number of expected bits corresponds to the number of DCCU instances. For each DCCU cluster bit-field,* *SDIV[0] corresponds to DCCU instance 0;* *SDIV[1] corresponds to DCCU instance 1;* *SDIV[2] corresponds to DCCU instance 2;* *SDIV[3] corresponds to DCCU instance 3;* *SDIV[4] corresponds to DCCU instance 4;* *SDIV[5] corresponds to DCCU instance 5;* *SDIV[6] corresponds to DCCU instance 6* ***Note***: *Ensure that the status of all DCCU clusters is verified by examining the corresponding bit-fields. Check only active cluster indexes, excluding bit (7) which is reserved.*

(table continues...)

(continued)

Field	Bits	Type	Description
PDIV	31:24	rh	**Peripheral clock divider monitor status** These bits store the alarms of the peripheral clock divider monitor. Every bit represents one status bit from one DCCU. The test is successful only if all clusters' status bits are set upon the test. ***Note**:* MONSTAT.*PDIV mirrors the alarm status of DCCUs, with each bit representing one DCCU instance. The number of expected bits corresponds to the number of DCCU instances. For each DCCU cluster bit-field,* *PDIV[0] corresponds to DCCU instance 0;* *PDIV[1] corresponds to DCCU instance 1;* *PDIV[2] corresponds to DCCU instance 2;* *PDIV[3] corresponds to DCCU instance 3;* *PDIV[4] corresponds to DCCU instance 4;* *PDIV[5] corresponds to DCCU instance 5;* *PDIV[6] corresponds to DCCU instance 6* ***Note**:* *Ensure that the status of all DCCU clusters is verified by examining the corresponding bit-fields. Check only active cluster indexes, excluding bit (7) which is reserved.*
0	7:3, 15:11	r	**Reserved** Read as 0; must be written with 0.

32.5 Debug information

This section is not applicable for the module being described.

32.6 References

This section is not applicable for the module being described.

32.7 CLOCK revision history

Reference	Description of change(s)
Date range: 2024-10-16 to 2025-02-28	
Crystal oscillator control register	• OSCCON.SUPM is reserved and must not be used. This reference is now removed from the register description.
Crystal oscillator	• Supply mode is reserved and must not be used. This reference is now removed from the chapter.
Features	• K2 pre-divider for SYSPLL is reserved and must not be used. This reference is now removed from the feature list.
Clock divider limitations	• Removed /2 and the footnote.
Functional description	• Added K2PRE to figure System PLL block diagram • Changed the equation and added remark for K2PRE=0. • Changed remark for K2PRE=0.
Module clocks	• FIS2DIV=0 mode option is reserved and must not be used. The reference is now removed from the CCU system clock options table.
Clock divider limitations	• Added line for relation between fSRICS and fFSI
Module clocks	• Added footnote that configuring CPBDIV=0 can lead to a bus error.
Feature list	• Refer to Datasheet in Feature list with respect to Peripheral PLL output frequencies
Clock divider limitations	• Added footnote to the Divider ratio restrictions table with respect to derivative specific deviations for f_{SRI}/f_{FSI}.
Clock divider limitations	• Added footnote to the Divider ratio restrictions table with respect to derivative specific deviations for f_{FSI2}/f_{FSI}.
Module clock sources and dividers	• Corrected CCU system clock options table with respect to fBACK entry for FSI2DIV and added footnote to check for device specific deviations.
Date range: 2024-05-15 to 2024-10-15	
Software controlled	• Typo fixed in RAMPSTAT link
Feature list	• Added "up to" to the sentence about Peripheral PLL
Clock sources	• Added a footnote to EXTCLKSB
MONSTAT	• Describe MONSTAT register KDIV1, KDIV2, KDIV3 bitfields • Changed description of SDIV and PDIV
(Whole functional block)	• Change phrase CLOCKING to CLOCK
Clock control unit	• Added feGTMhalf in module clock overview and divider ratio limitations
Module clocks	• Corrected the note for fTPB

32 CLOCK

Reference	Description of change(s)
OSCCON	• OSCCON Reset value was moved to the derivative specific section and added a note to that effect • OSCCON MODE descriptions aligned to descriptions in Crystal Oscillator chapter
PERPLLCON1, SYSPLLCON1	• Note corrected from SYSPLLCON1 and PERPLLCON1 KxPREDIV
CCUCON	• CCUCON Reset value was moved to the derivative specific section and added a note to that effect

32.8 TC4Dx CLOCK information

32.8.1 TC4Dx CLOCK configuration

There are no deviations from the generic specification.

32.8.2 TC4Dx CLOCK features

- Maximum clock frequency of system clocks is 500 MHz
- PPU integration with asynchronous clocking with respect to system clocks
  - PPUDIV bit-field located in PERCCUCON0
- Second System-PLL output used to provide dedicated PPU source clock f_{PLLPPU}
- Maximum PPU clock frequency is 454 MHz
  - f_{PLLPPU} configured through bit-fields SYSPLLCON1.K3DIV and K3PREDIV
- Clock provided to PPU during ramp process (start-up/shut-down and clock emergency) is $f_{RAMP} \div 2$
- COMPB interconnect is present and uses f_{SPB}
- Seven Distribution CCU (DCCU) instances: 0, 1, 2, 3, 4, 5, and 6

32.8.3 TC4Dx CLOCK functional description

SYSPLL

Features

In addition, TC4Dx also includes:

- K3 pre-divider (phase combiner) for fractional divisions by 1.0, 1.1, 1.2, 1.4, 1.6, 1.7, 2.0 for PPU-PLL pre-divider frequencies (f_{PCPPU})

Functional description

In addition, the SYSPLL also consists of the following building blocks:

- Output stage for PPU clock composed of a pre-divider for fractional divisions (1.0, 1.1, 1.2, 1.4, 1.6, 1.7, 2.0) and an integer divider that scales down the DCO frequency by a factor K3, configurable through the register section

SYSPLL register interface

In addition, all control and status values of the SYSPLL are stored in and accessed through:

- SYSPLL configuration register 1 (SYSPLLCON1): K2, K3 dividers and K3 pre-divider

Basic SYSPLL operation

In addition, the following dividers are available in TC4Dx:

- SYSPLLCON1.K3DIV and SYSPLLCON1.K3PREDIV

In the PPU clock output stage, the DCO frequency is divided down by a factor K3 combined with K3PRE. K3PRE is an additional divider circuit that is capable of generating selected non-integer division ratios of factor 1.0, 1.1, 1.2, 1.4, 1.6, 1.7 and 2.0.

The output frequency f_{PLLPPU} is given by:

$$
\begin{aligned}
f_{\mathrm{PLLPPU}} &= f_{\mathrm{DCO}} \div (K3 \times 1.0) \rightarrow K3\mathrm{PRE} = 0 \\
f_{\mathrm{PLLPPU}} &= f_{\mathrm{DCO}} \div (K3 \times 1.1) \rightarrow K3\mathrm{PRE} = 1 \\
f_{\mathrm{PLLPPU}} &= f_{\mathrm{DCO}} \div (K3 \times 1.2) \rightarrow K3\mathrm{PRE} = 2 \\
f_{\mathrm{PLLPPU}} &= f_{\mathrm{DCO}} \div (K3 \times 1.4) \rightarrow K3\mathrm{PRE} = 4 \\
f_{\mathrm{PLLPPU}} &= f_{\mathrm{DCO}} \div (K3 \times 1.6) \rightarrow K3\mathrm{PRE} = 6 \\
f_{\mathrm{PLLPPU}} &= f_{\mathrm{DCO}} \div (K3 \times 1.7) \rightarrow K3\mathrm{PRE} = 7 \\
f_{\mathrm{PLLPPU}} &= f_{\mathrm{DCO}} \div (K3 \times 2.0) \rightarrow K3\mathrm{PRE} = 10
\end{aligned}
\tag{62}
$$

***Note**: If the KDIVs are changed during runtime, CCUMON.KDV0MONEN and CCUMON.KDVPPUMONEN must be disabled before changing the KDIV values and then re-enabled after the change.*

PERPLL

Basic PERPLL operation

***Note**: If the KDIVs are changed during runtime, CCUMON.KDVxMONEN (x=1..3) must be disabled before changing the KDIV values and then re-enabled after the change.*

Distribution

Clock control unit

In TC4Dx, the PPU clock is generated by the second SYSPLL output and is handled like another asynchronous peripheral clock.

Clock divider limitations

Table 1319 Divider ratio restrictions

Clock A	Clock B	Allowed values
f_{SRI}	f_{SPB}	n ≥ 1

Start-up and shut-down

Software controlled

Startup controlled by application software

If the target system frequency of the device is 500 MHz which is higher than the maximum frequency of the PPU, the PPU must run slower and asynchronous to the rest of the system. That means the PPU has a dedicated clock f_{PPU} which is sourced from $f_{sourceppu}$. This is a fixed configuration which cannot be changed for the device. During the ramp phase, the $f_{sourceppu}$ is driven by a divided version of the ramp clock. This is needed because the ramp oscillator has only one target frequency which is set to 500 MHz while the target frequency of the PPU is lower. This avoids overclocking of the PPU. Please be aware that the target f_{RAMP} frequency should be 100 MHz below the configured f_{PLL0} frequency.

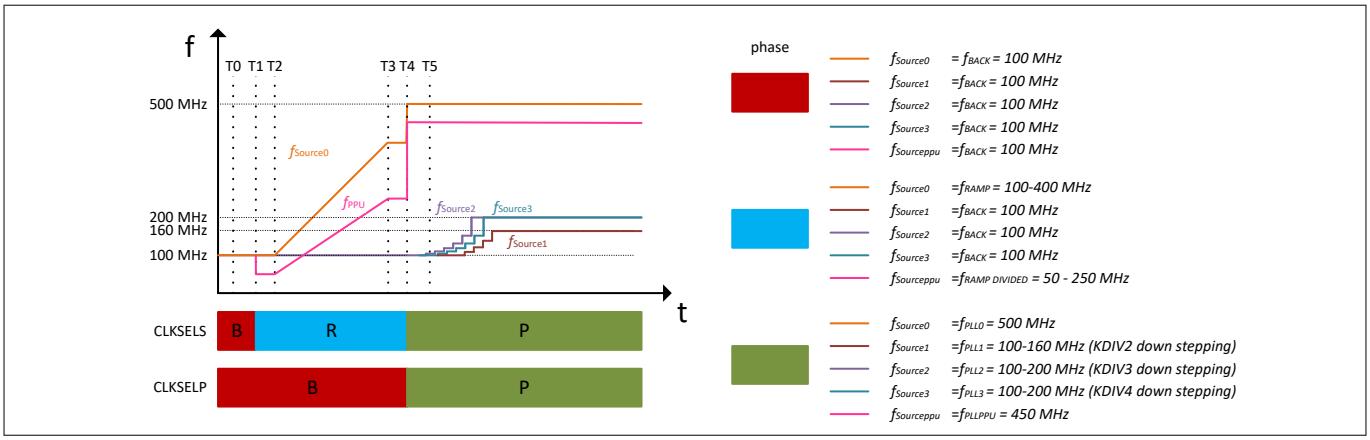


Figure 753 Start up 500 MHz use case

Shutdown controlled by application software

If the maximum system frequency of the device is 500 MHz which is higher than the maximum frequency of the PPU, the PPU must run slower and asynchronous to the rest of the system. That means the PPU has a dedicated clock f_{PPU} which is sourced from $f_{sourceppu}$. This is a fixed configuration which cannot be changed for the device. During the ramp phase, the $f_{sourceppu}$ is driven by a divided version of the ramp clock.

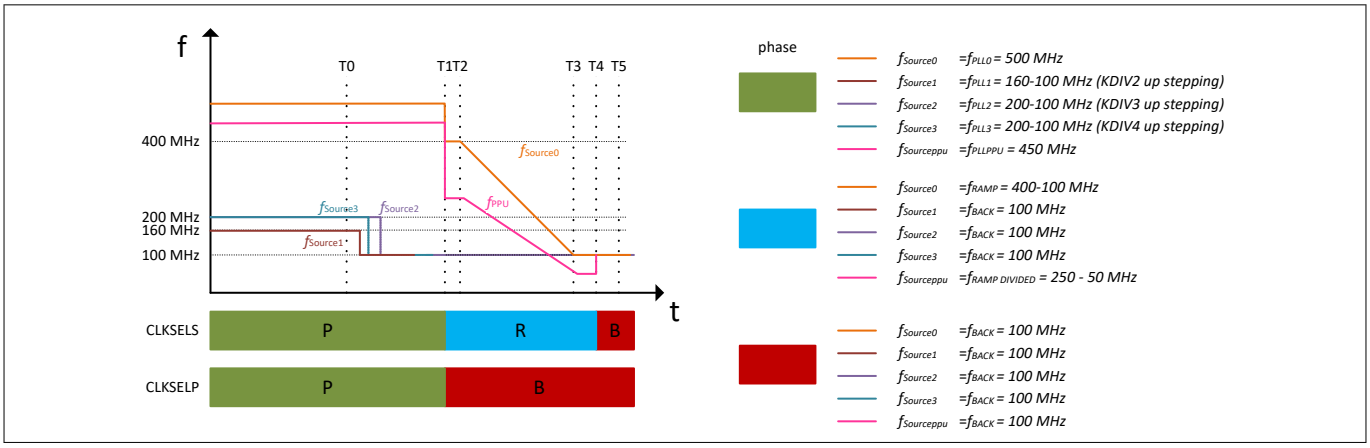


Figure 754 Shut down 500 MHz use case

32.8.4 TC4Dx CLOCK registers

32.8.4.1 Register address space - CLOCK

Table 1320 Registers address space - CLOCK

Module	Base address	End address	Note
CLOCK	$F0064000_H$	$F0067FFF_H$	FPI slave interface

32.8.4.2 Register overview - access mode glossary

Table 1321 Register overview - access mode glossary

Keyword	Description
CSE	Access protection using PROT register CLOCK_PROTCSE .
E	Access protection using PROT register CLOCK_PROTE .
SE	Access protection using PROT register CLOCK_PROTSE .
APU-P	Protection group consisting of registers CLOCK_ACCEN_WRA , CLOCK_ACCEN_WRB , CLOCK_ACCEN_RDA , CLOCK_ACCEN_RDB , CLOCK_ACCEN_VM , CLOCK_ACCEN_PRS .
P	Access protection using *APU-P* registers.
APU-PCS	Protection group consisting of registers CLOCK_ACCENCS_WRA , CLOCK_ACCENCS_WRB , CLOCK_ACCENCS_RDA , CLOCK_ACCENCS_RDB , CLOCK_ACCENCS_VM , CLOCK_ACCENCS_PRS .
PCS	Access protection using *APU-PCS* registers.
BE	Always returns a Bus Error.
U	No access restrictions.
PROT	Access restrictions as defined in the PROT register access rules.
STP	Access only during start-up, before start-up protection is set (SMM_STCON.ST_PROT=1).

32.8.4.3 Registers overview - CLOCK (ascending offset address)

Table 1322 Registers overview - CLOCK (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CLOCK_ID	Module Identification register	0008_H	P	BE	PowerOn Reset	5569
CLOCK_PROTE	PROT Register Endinit	0020_H	U	SV, PROT	Application Reset	5570
CLOCK_PROTSE	PROT register safe endinit	0024_H	U	SV, PROT	Application Reset	5572
CLOCK_PROTCSE	PROT Register Cyber-Secure APU Endinit	0028_H	U	SV, PROT	5574	5574
CLOCK_ACCEN_WRA	Write access enable register A	0040_H	U	SE, SV	Application Reset	5575

(table continues...)

Table 1322 (continued) Registers overview - CLOCK (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CLOCK_ACCEN_WRB	Write access enable register B	0044_H	U	SE, SV	Application Reset	5576
CLOCK_ACCEN_RDA	Read access enable register A	0048_H	U	SE, SV	Application Reset	5576
CLOCK_ACCEN_RDB	Read access enable register B	$004C_H$	U	SE, SV	Application Reset	5577
CLOCK_ACCEN_VM	VM access enable register	0050_H	U	SE, SV	Application Reset	5577
CLOCK_ACCEN_PRS	PRS access enable register	0054_H	U	SE, SV	Application Reset	5578
CLOCK_ACCENCS_WRA	ACCENCS write access enable register A	0060_H	U	CSE, SV	5579	5579
CLOCK_ACCENCS_WRB	ACCENCS write access enable register B	0064_H	U	CSE, SV	Application Reset	5579
CLOCK_ACCENCS_RDA	ACCENCS read access enable register A	0068_H	U	CSE, SV	Application Reset	5580
CLOCK_ACCENCS_RDB	ACCENCS read access enable register B	$006C_H$	U	CSE, SV	Application Reset	5580
CLOCK_ACCENCS_VM	ACCENCS VM access enable register	0070_H	U	CSE, SV	Application Reset	5581
CLOCK_ACCENCS_PRS	ACCENCS PRS access enable register	0074_H	U	CSE, SV	Application Reset	5582
CLOCK_OSCCON	Crystal oscillator control register	0100_H	P	SV, E, P	System Reset	5648
CLOCK_RAMPCON0	Ramp oscillator control register 0	0200_H	P	SV, E, P	System Reset	5586
CLOCK_RAMPSTAT	Ramp oscillator status register	$020C_H$	P	BE	5588	5588
CLOCK_SYSPLLCON0	System-PLL configuration register 0	0300_H	P	SV, E, P	System Reset	5590
CLOCK_SYSPLLCON1	System-PLL configuration register 1	0304_H	P	SV, E, P	System Reset	5652
CLOCK_SYSPLLCON2	System-PLL configuration register 2	0308_H	P	SV, E, P	System Reset	5593
CLOCK_SYSPLLSTAT	System-PLL status register	$030C_H$	P	BE	System Reset	5593
CLOCK_SMODCON0	System-PLL modulation trim configuration register 0	0340_H	P	SV, E, P	PowerOn Reset	5594
CLOCK_PERPLLCON0	Peripheral-PLL configuration register 0	0380_H	P	SV, E, P	System Reset	5595
CLOCK_PERPLLCON1	Peripheral-PLL configuration register 1	0384_H	P	SV, E, P	System Reset	5654

(table continues...)

Table 1322 (continued) Registers overview - CLOCK (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CLOCK_PERPLLCON2	Peripheral-PLL configuration register 2	0388_H	P	SV, E, P	System Reset	5599
CLOCK_PERPLLSTAT	Peripheral-PLL status register	$038C_H$	P	BE	System Reset	5600
CLOCK_PMODCON0	Peripheral-PLL modulation trim configuration register 0	$03A0_H$	P	SV, E, P	PowerOn Reset	5601
CLOCK_CCUCON	Global clock control register	0400_H	P	SV, E, P	System Reset	5656
CLOCK_CCUSTAT	Global clock status register	0404_H	P	BE	5604	5604
CLOCK_SYSCCUCON0	System CCU clock control register 0	0410_H	P	SV, E, P	See 5657	5657
CLOCK_SYSCCUCON1	System CCU clock control register 1	0414_H	P	SV, E, P	See 5661	5661
CLOCK_PERCCUCON0	Peripheral CCU clock control register 0	0420_H	P	SV, E, P	See 5665	5665
CLOCK_PERCCUCON1	Peripheral CCU clock control register 1	0424_H	P	SV, E, P	See 5670	5670
CLOCK_EXTCON	External clock control register	0430_H	P	SV, E, P	System Reset	5674
CLOCK_EXTFDCON	Fractional divider control register	0434_H	P	SV, E, P	System Reset	5627
CLOCK_CCUMON	Global clock monitor register	0500_H	P	SV, E, P	System Reset	5628
CLOCK_OSCMON0	EVR oscillator monitor register	0504_H	P	SV, E, P	System Reset	5632
CLOCK_OSCMON1	Crystal oscillator monitor register	0508_H	P	SV, E, P	System Reset	5633
CLOCK_OSCSTAT	Crystal oscillator status register	$050C_H$	P	BE	System Reset	5635
CLOCK_RAMPMON	Ramp oscillator monitor register	0510_H	P	SV, E, P	System Reset	5637
CLOCK_MONSTAT	Global clock monitor status register	0518_H	P	BE	PowerOn Reset	5638

32.8.4.4 Crystal oscillator control register

CLOCK_OSCCON Offset address: 0100_H

Crystal oscillator control register System Reset value: $0100\ 0723_H$

32 CLOCK

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		CLKSELPHY2	CLKSELPHY0	0	RSVD26	INSEL		0		RSVD21	RSVD20	X2CAP3EN	X2CAP2EN	X2CAP1EN	X2CAP0EN
r		rw	rw	r	rw	rw		r		rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
X1CAP3EN	X1CAP2EN	X1CAP1EN	X1CAP0EN	RSVD11	MODE			HYSCTL		HYSEN	RSVD4	RSVD2		GAINSEL	
rw	rw	rw	rw	rw	rw			rw		rw	rw	rw		rw	

Field	Bits	Type	Description
GAINSEL	1:0	rw	**Oscillator gain selection** Gain of the oscillator output stage. It can be used to adjust the peak to peak amplitude. 00_B Minimum gain configuration. 01_B Intermediate gain configuration. 10_B Intermediate gain configuration. 11_B Maximum gain configuration (default value) Note: It is not recommended to change the default value
RSVD2	3:2	rw	**Reserved** Read as reset value; shall be written with reset value.
RSVD4	4	rw	**Reserved** Read as reset value; shall be written with reset value.
HYSEN	5	rw	**Enable shaper hysteresis** 0_B Hysteresis is disabled 1_B Hysteresis is enabled (recommended)
HYSCTL	7:6	rw	**Shaper hysteresis control. Applicable only if HYSEN=1** 00_B Hysteresis setting 1 (highest hysteresis, default) Note: It is not recommended to change the default value 01_B Hysteresis setting 2; Reserved for future use 10_B Hysteresis setting 3; Reserved for future use 11_B Hysteresis setting 4; Reserved for future use
MODE	10:8	rw	**Oscillator mode** This bit-field defines which mode can be used and if the oscillator entered the power-saving mode or not. 000_B **XTALOSC**: Crystal mode 010_B **EXTCLK**: External clock mode with shaper 011_B **PDOWN**: Power down mode 111_B **EXTCLKSB**: External clock mode without shaper Note: This mode should be used only for debug purpose. For normal operation use only modes with shaper enabled. **others,** Reserved, do not use.

(table continues...)

(continued)

Field	Bits	Type	Description
RSVD11	11	rw	**Reserved** Read as reset value; shall be written with reset value.
X1CAP0EN	12	rw	**XTAL1 capacitance 0 enable** ***Note:*** *Total capacitance for XTAL1 is the sum of the enabled capacitance 0 to 3.* 0_B Capacitance C_{L0} is disabled 1_B Capacitance C_{L0} is enabled
X1CAP1EN	13	rw	**XTAL1 capacitance 1 enable** ***Note:*** *Total capacitance for XTAL1 is the sum of the enabled capacitance 0 to 3.* 0_B Capacitance C_{L1} is disabled 1_B Capacitance C_{L1} is enabled
X1CAP2EN	14	rw	**XTAL1 capacitance 2 enable** ***Note:*** *Total capacitance for XTAL1 is the sum of the enabled capacitance 0 to 3.* 0_B Capacitance C_{L2} is disabled 1_B Capacitance C_{L2} is enabled
X1CAP3EN	15	rw	**XTAL1 capacitance 3 enable** ***Note:*** *Total capacitance for XTAL1 is the sum of the enabled capacitance 0 to 3.* 0_B Capacitance C_{L3} is disabled 1_B Capacitance C_{L3} is enabled
X2CAP0EN	16	rw	**XTAL2 capacitance 0 enable** ***Note:*** *Total capacitance for XTAL2 is the sum of the enabled capacitance 0 to 3.* 0_B Capacitance C_{L0} is disabled 1_B Capacitance C_{L0} is enabled
X2CAP1EN	17	rw	**XTAL2 capacitance 1 enable** ***Note:*** *Total capacitance for XTAL2 is the sum of the enabled capacitance 0 to 3.* 0_B Capacitance C_{L1} is disabled 1_B Capacitance C_{L1} is enabled
X2CAP2EN	18	rw	**XTAL2 capacitance 2 enable** ***Note:*** *Total capacitance for XTAL2 is the sum of the enabled capacitance 0 to 3.* 0_B Capacitance C_{L2} is disabled 1_B Capacitance C_{L2} is enabled

(table continues...)

(continued)

Field	Bits	Type	Description
X2CAP3EN	19	rw	**XTAL2 capacitance 3 enable** ***Note**: Total capacitance for XTAL2 is the sum of the enabled capacitance 0 to 3.* 0_B Capacitance C_{L3} is disabled 1_B Capacitance C_{L3} is enabled
RSVD20	20	rw	**Reserved** Read as reset value; shall be written with reset value.
RSVD21	21	rw	**Reserved** Read as reset value; shall be written with reset value.
INSEL	25:24	rw	**Input selection** This bit-field defines the primary clock source f_{IN} for the System-PLL and Peripheral-PLL as listed below. 00_B **BACK**: f_{BACK} is used as clock source for System-PLL and Peripheral-PLL 01_B **OSC**: f_{OSC} is used as clock source for System-PLL and Peripheral-PLL 10_B **SYSCLK**: $f_{PADSYSCLK}$ is used as clock source for System-PLL and Peripheral-PLL 11_B **OSC_F**: f_{OSC_F} is used as clock source for System-PLL f_{OSC} is used as clock source for Peripheral-PLL
RSVD26	26	rw	**Reserved** Read as reset value; shall be written with reset value.
CLKSELPHY0	28	rw	**Reference clock Input selection** This bit-field defines the alternative reference clock source $f_{HSPHY0ALTREF}$ for the HSPHY.PHY0 PMA module as listed below. ***Note**: In addition, the user has to set the HSPHY PHY0_CTRL1.PRS bits accordingly. For details please refer to the HSPHY UM register section.* 0_B **OSC**: f_{OSC} is used as reference clock source 1_B **PCIEREF**: $f_{PCIEREF}$ is used as reference clock source
CLKSELPHY2	29	rw	**Reference clock Input selection** This bit-field defines the alternative reference clock source $f_{HSPHY2ALTREF}$ for the HSPHY.PHY2 PMA module as listed below. ***Note**: In addition, the user has to set the HSPHY PHY2_CTRL1.PRS bits accordingly. For details please refer to the HSPHY UM register section.* 0_B **OSC**: f_{OSC} is used as reference clock source 1_B **PCIEREF**: $f_{PCIEREF}$ is used as reference clock source
0	23:22, 27, 31:30	r	**Reserved** Read as reset value; shall be written with reset value.

Table 1323 Access mode restrictions of CLOCK_OSCCON sorted by descending priority

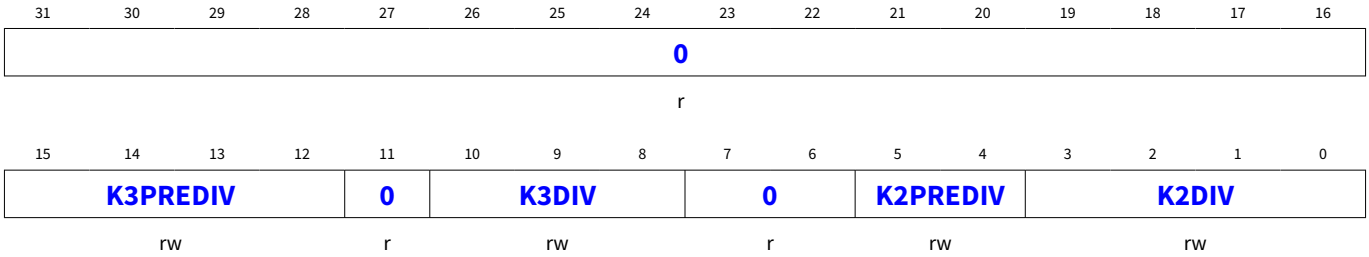
Mode name	Access mode		Description
CCUSTAT.LCK == 0	rw	, CLKSELPHY0, CLKSELPHY2, GAINSEL, HYSCTL, HYSEN, INSEL, MODE , RSVD11 , RSVD2 , RSVD20 , RSVD21 , RSVD26 , RSVD4 , X1CAP0EN, X1CAP1EN, X1CAP2EN, X1CAP3EN, X2CAP0EN, X2CAP1EN, X2CAP2EN, X2CAP3EN	Normal access
(default)	r	, CLKSELPHY0, CLKSELPHY2, GAINSEL, HYSCTL, HYSEN, INSEL, MODE , RSVD11 , RSVD2 , RSVD20 , RSVD21 , RSVD26 , RSVD4 , X1CAP0EN, X1CAP1EN, X1CAP2EN, X1CAP3EN, X2CAP0EN, X2CAP1EN, X2CAP2EN, X2CAP3EN	Limited access

32.8.4.5 System-PLL configuration register 1

***Note**: Writing to this register sets the .LCK bit and automatically triggers an update transfer of the register value to the System-PLL's k-divider logic.*

CLOCK_SYSPLLCON1 Offset address: 0304_H

System-PLL configuration register 1 System Reset value: $0000\ 070F_H$



Field	Bits	Type	Description
K2DIV	3:0	rw	**K2-divider value** This bit-field defines the dividing factor of the K2-divider (*K2*). The effective value is K2 = K2DIV+1. $f_{PLL0} = f_{PC0} \div K2$

(table continues...)

(continued)

Field	Bits	Type	Description
K2PREDIV	5:4	rw	**PLL0 pre-divider select** This bit-field defines the pre-dividing factor (*K2PRE*) in front of the K2-divider. The effective value of K2PRE is listed below. $f_{PC0} = f_{DCO} \div K2PRE$ ***Note**: The configuration sequence for the K-dividers described in the functional description chapter must be followed in order to change the value of this bitfield, otherwise frequency glitches may occur.* 00_B **DIVBY1_0**: K2PRE = 1.0 ; this value must be used for normal operation 01_B **DIVBY2_0**: K2PRE = 2.0 (reserved)
K3DIV	10:8	rw	**K3-divider value** This bit-field defines the dividing factor of the K3-divider (*K3*). The effective value is K3 = K3DIV+1. $f_{PPU} = f_{PCPPU} \div K3$
K3PREDIV	15:12	rw	**PLLPPU pre-divider select** This bit-field defines the pre-dividing factor (*K3PRE*) in front of the K3-divider. The effective value of K3PRE is listed below. $f_{PCPPU} = f_{DCO} \div K3PRE$ ***Note**: The configuration sequence for the K-dividers described in the functional description chapter must be followed in order to change the value of this bitfield, otherwise frequency glitches may occur.* 0_H **DIVBY1_0**: K3PRE = 1.0 1_H **DIVBY1_1**: K3PRE = 1.1 2_H **DIVBY1_2**: K3PRE = 1.2 4_H **DIVBY1_4**: K3PRE = 1.4 6_H **DIVBY1_6**: K3PRE = 1.6 7_H **DIVBY1_7**: K3PRE = 1.7 A_H **DIVBY2_0**: K3PRE = 2.0 **others,** configurations not present in this list act as the 0_H configuration.
0	7:6, 11, 31:16	r	**Reserved** Read as reset value; shall be written with reset value.

Table 1324 Access mode restrictions of CLOCK_SYSPLLCON1 sorted by descending priority

Mode name	Access mode		Description
CCUSTAT.LCK == 0	rw	K2DIV, K2PREDIV, K3DIV, K3PREDIV	Normal access
(default)	r	K2DIV, K2PREDIV, K3DIV, K3PREDIV	Limited access

32.8.4.6 Peripheral-PLL configuration register 1

***Note**: Writing to this register sets the .LCK bit and automatically triggers an update transfer of the register value to the Peripheral-PLL.*

CLOCK_PERPLLCON1 Offset address: 0384_H

Peripheral-PLL configuration register 1 System Reset value: $0007\ 070F_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0										K4PREDIV		0	K4DIV		
r										rw		r	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
K3PREDIV				0	K3DIV			0		K2PREDIV		K2DIV			
rw				r	rw			r		rw		rw			

Field	Bits	Type	Description
K2DIV	3:0	rw	**K2-divider value** This bit-field defines the dividing factor of the K2-divider (*K2*). The effective value is K2 = K2DIV+1. $f_{PLL1} = f_{PC1} \div K2$
K2PREDIV	5:4	rw	**PLL1 pre-divider select** This bit-field defines the pre-dividing factor (*K2PRE*) in front of the K2-divider. The effective value of K2PRE is listed below. $f_{PC1} = f_{DCO} \div K2PRE$ ***Note**: The configuration sequence for the K-dividers described in the functional description chapter must be followed in order to change the value of this bitfield, otherwise frequency glitches may occur.* 00_B **DIVBY1_0**: K2PRE = 1.0 01_B **DIVBY2_0**: K2PRE = 2.0 10_B **DIVBY1_2**: K2PRE = 1.2 11_B **DIVBY1_6**: K2PRE = 1.6
K3DIV	10:8	rw	**K3-divider value** This bit-field defines the dividing factor of the K3-divider (*K3*). The effective value is K3 = K3DIV+1. $f_{PLL2} = f_{PC2} \div K3$

(table continues...)

(continued)

Field	Bits	Type	Description
K3PREDIV	15:12	rw	**PLL2 pre-divider select** This bit-field defines the pre-dividing factor (*K3PRE*) in front of the K3-divider. The effective value of K3PRE is listed below. $f_{PC2} = f_{DCO} \div K3PRE$ ***Note**: The configuration sequence for the K-dividers described in the functional description chapter must be followed in order to change the value of this bitfield, otherwise frequency glitches may occur.* 0_H **DIVBY1_0**: K3PRE = 1.0 1_H **DIVBY1_1**: K3PRE = 1.1 2_H **DIVBY1_2**: K3PRE = 1.2 4_H **DIVBY1_4**: K3PRE = 1.4 6_H **DIVBY1_6**: K3PRE = 1.6 7_H **DIVBY1_7**: K3PRE = 1.7 A_H **DIVBY2_0**: K3PRE = 2.0 **others,** configurations not present in this list act as the 0_H configuration.
K4DIV	18:16	rw	**K4-divider value** This bit-field defines the dividing factor of the K4-divider (*K4*). The effective value is K4 = K4DIV+1. $f_{PLL3} = f_{PC3} \div K4$
K4PREDIV	21:20	rw	**PLL3 pre-divider select** This bit-field defines the pre-dividing factor (*K4PRE*) in front of the K4-divider. The effective value of K4PRE is listed below. $f_{PC3} = f_{DCO} \div K4PRE$ ***Note**: The configuration sequence for the K-dividers described in the functional description chapter must be followed in order to change the value of this bitfield, otherwise frequency glitches may occur.* 00_B **DIVBY1_0**: K4PRE = 1.0 01_B **DIVBY2_0**: K4PRE = 2.0 10_B **DIVBY1_2**: K4PRE = 1.2 11_B **DIVBY1_6**: K4PRE = 1.6
0	7:6, 11, 19, 31:22	r	**Reserved** Read as reset value; shall be written with reset value.

Table 1325 Access mode restrictions of CLOCK_PERPLLCON1 sorted by descending priority

Mode name	Access mode		Description
CCUSTAT.LCK == 0	rw	K2DIV, K2PREDIV, K3DIV, K3PREDIV, K4DIV, K4PREDIV	Normal access

(table continues...)

Table 1325 (continued) Access mode restrictions of CLOCK_PERPLLCON1 sorted by descending priority

Mode name	Access mode		Description
(default)	r	K2DIV, K2PREDIV, K3DIV, K3PREDIV, K4DIV, K4PREDIV	Limited access

32.8.4.7 Global clock control register

This register contains control information for global clocking system functionality.

***Note**: It is not recommended to change the CCUCON configuration during a ramp-up or a ramp-down scenario.*

CLOCK_CCUCON Offset address: 0400_H

Global clock control register System Reset value: $0001\ 0001_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															CLKSELP
r															rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															CLKSELS
r															rw

Field	Bits	Type	Description
CLKSELS	1:0	rw	**Selection for system clock source** This bit-field selects the clock source that is used for the clock generation of $f_{source0}$ (and $f_{sourceppu}$ in devices with an asynchronous PPU). Reading these bits reflects the last written value by software. However, due to hardware controlled emergency switching and upcoming reset triggered by hardware, the actual used value has to be read from .CLKSELS ***Note**: In general, before switching clock sources, the user has to check the availibility of the related source:* *f_{PLL0} shall only be chosen as clock source if .PWRSTAT = 1 and .LOCK = 1.* *f_{RAMP} (and $f_{RAMPHALF}$ in devices with an asynchronous PPU) shall only be chosen as clock source if .ACTIVE = 1 and .FLLOCK = 1.* 00_B **PLL**: f_{PLL0} is used as clock source $f_{source0}$ 01_B **BACK**: f_{BACK} is used as clock source $f_{source0}$ 10_B **RAMP**: f_{RAMP} is used as clock source $f_{source0}$; $f_{RAMPHALF}$ is used as clock source $f_{sourceppu}$ (only valid for devices with an asynchronous PPU) **others,** Hardware is busy

(table continues...)

(continued)

Field	Bits	Type	Description
CLKSELP	16	rw	**Selection for peripheral clock sources** This bit-field selects the clock source that is used for the clock generation of f_{src1}, $f_{source2}$ and $f_{source3}$. Reading this bit reflects the last written value by software. However, due to hardware controlled emergency switching, the actual used value has to be read from .CLKSELP ***Note**: In general, before switching clock sources, the user has to check the availibility of the related source:* *f_{src1} and $f_{PLL2,3}$ shall only be chosen as clock source if .PWRSTAT = 1 and .LOCK = 1.* *For measurement purposes, $f_{source1}$ can be driven by f_{SPB} or $f_{PADSYSCLK}$ controlled by .CLKSELSRC1. Additional details can be found in the **Distribution** chapter in the **Clock control unit overview** figure.* 0_B **PLL**: $f_{src1} = f_{PLL1}$; $f_{source2} = f_{PLL2}$; $f_{source3} = f_{PLL3}$ 1_B **BACK**: f_{BACK} is used as clock source for f_{src1} , $f_{source2}$, and $f_{source3}$
0	15:2, 31:17	r	**Reserved** Read as 0; must be written with 0.

Table 1326 Access mode restrictions of CLOCK_CCUCON sorted by descending priority

Mode name	Access mode		Description
CCUSTAT.LCK == 0	rw	CLKSELP, CLKSELS	Normal access
(default)	r	CLKSELP, CLKSELS	Limited access

32.8.4.8 System CCU clock control register 0

CLOCK_SYSCCUCON0 Offset address: 0410_H

System CCU clock control register 0 Reset values see: Table 1328

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	**UP**	**SRICSDIV**	**LPDIV**	**STMDIV**				**0**			**FSI2DIV**	**FSIDIV**			
r	w	rh	rw	rw				r			rw	rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SRIDIV				**TPBDIV**				**RSVD4**				**SPBDIV**			
rw				rw				rw				rw			

Field	Bits	Type	Description
SPBDIV	3:0	rw	**SPB divider reload value** The resulting SPB frequency is configured to $f_{SPB} = f_{source0} \div$ SPBDIV for the allowed configurations as listed below. 0_H Reserved, do not use this combination 1_H $f_{SPB} = f_{source0} \div 1$ 2_H $f_{SPB} = f_{source0} \div 2$ 3_H $f_{SPB} = f_{source0} \div 3$ 4_H $f_{SPB} = f_{source0} \div 4$ 5_H $f_{SPB} = f_{source0} \div 5$ 6_H $f_{SPB} = f_{source0} \div 6$ 7_H Reserved, do not use this combination 8_H $f_{SPB} = f_{source0} \div 8$ 9_H Reserved, do not use this combination A_H $f_{SPB} = f_{source0} \div 10$ B_H Reserved, do not use this combination C_H $f_{SPB} = f_{source0} \div 12$ D_H Reserved, do not use this combination E_H Reserved, do not use this combination F_H $f_{SPB} = f_{source0} \div 15$
RSVD4	7:4	rw	**Reserved** Read as reset value; shall be written with reset value.
TPBDIV	11:8	rw	**TPB divider reload value** The resulting TPB frequency is configured to $f_{TPB} = f_{source0} \div$ TPBDIV for the allowed configurations as listed below. 0_H f_{TPB} is stopped 1_H $f_{TPB} = f_{source0} \div 1$ 2_H $f_{TPB} = f_{source0} \div 2$ 3_H $f_{TPB} = f_{source0} \div 3$ 4_H $f_{TPB} = f_{source0} \div 4$ 5_H $f_{TPB} = f_{source0} \div 5$ 6_H $f_{TPB} = f_{source0} \div 6$ 7_H Reserved, do not use this combination 8_H $f_{TPB} = f_{source0} \div 8$ 9_H Reserved, do not use this combination A_H $f_{TPB} = f_{source0} \div 10$ B_H Reserved, do not use this combination C_H $f_{TPB} = f_{source0} \div 12$ D_H Reserved, do not use this combination E_H Reserved, do not use this combination F_H $f_{TPB} = f_{source0} \div 15$

(table continues...)

(continued)

Field	Bits	Type	Description
SRIDIV	15:12	rw	**SRI divider reload value** The resulting SRI frequency is configured to $f_{SRI} = f_{source0} \div$ SRIDIV for the allowed configurations as listed below. 0_H Reserved, do not use this combination 1_H $f_{SRI} = f_{source0} \div 1$ 2_H $f_{SRI} = f_{source0} \div 2$ 3_H $f_{SRI} = f_{source0} \div 3$ 4_H $f_{SRI} = f_{source0} \div 4$ 5_H $f_{SRI} = f_{source0} \div 5$ 6_H $f_{SRI} = f_{source0} \div 6$ 7_H Reserved, do not use this combination 8_H $f_{SRI} = f_{source0} \div 8$ 9_H Reserved, do not use this combination A_H $f_{SRI} = f_{source0} \div 10$ B_H Reserved, do not use this combination C_H $f_{SRI} = f_{source0} \div 12$ D_H Reserved, do not use this combination E_H Reserved, do not use this combination F_H $f_{SRI} = f_{source0} \div 15$
FSIDIV	19:16	rw	**FSI divider reload value** The resulting FSI frequency is configured to $f_{FSI} = f_{source0} \div$ FSIDIV for the allowed configurations as listed below. ***Note**: The needed ratio of $f_{SRI} : f_{FSI}$ must be manually configured and is not automatically set by hardware* ***Note**:* 0_H Reserved, do not use this combination 1_H $f_{FSI} = f_{source0} \div 1$ 2_H $f_{FSI} = f_{source0} \div 2$ 3_H $f_{FSI} = f_{source0} \div 3$ 4_H $f_{FSI} = f_{source0} \div 4$ 5_H $f_{FSI} = f_{source0} \div 5$ 6_H $f_{FSI} = f_{source0} \div 6$ 7_H Reserved, do not use this combination 8_H $f_{FSI} = f_{source0} \div 8$ 9_H Reserved, do not use this combination A_H $f_{FSI} = f_{source0} \div 10$ B_H Reserved, do not use this combination C_H $f_{FSI} = f_{source0} \div 12$ D_H Reserved, do not use this combination E_H Reserved, do not use this combination F_H $f_{FSI} = f_{source0} \div 15$

(table continues...)

(continued)

Field	Bits	Type	Description
FSI2DIV	20	rw	**FSI2 divider reload value** The resulting FSI2 frequency is configured as listed below. 0_B Reserved, do not use this combination 1_B $f_{FSI2} = f_{SRI}$
STMDIV	27:24	rw	**STM divider reload value** The resulting STM frequency is configured to $f_{STM} = f_{source0} \div$ STMDIV for the allowed configurations as listed below. For STMDIV = 0000_B the clock is shut off. 0_H f_{STM} is stopped 1_H $f_{STM} = f_{source0} \div 1$ 2_H $f_{STM} = f_{source0} \div 2$ 3_H $f_{STM} = f_{source0} \div 3$ 4_H $f_{STM} = f_{source0} \div 4$ 5_H $f_{STM} = f_{source0} \div 5$ 6_H $f_{STM} = f_{source0} \div 6$ 7_H Reserved, do not use this combination 8_H $f_{STM} = f_{source0} \div 8$ 9_H Reserved, do not use this combination A_H $f_{STM} = f_{source0} \div 10$ B_H Reserved, do not use this combination C_H $f_{STM} = f_{source0} \div 12$ D_H Reserved, do not use this combination E_H Reserved, do not use this combination F_H $f_{STM} = f_{source0} \div 15$
LPDIV	28	rw	**Low Power divider reload value** This bit overwrites the current divider settings of the listed frequencies to enable a low power mode of operation. The selected LPDIV setting is valid for all clocks derived from f_{XXX} with XXX = SPB, SRI, SRICS, FSI, GETH, LETH, LETH100, CPB, RCB, TPB, GTM, eGTM, and STM. Do not change the PLL setting while in the low power mode. ***Note**:* *LPDIV only affects enabled clocks, that means XXXDIV = 0 will not be changed.* ***Note**:* *With an activated LPDIV the chip is set to a deep power down state (LPDIV=1). When a wake-up signal is received over the communication interfaces, it is possible to quickly return the chip to the "full power" state (LPDIV=0). Only valid in system mode.* 0_B f_{XXX} controlled by the related SYSCCUCON0 and SYSCCUCON1 bit-fields 1_B $f_{XXX} = f_{source0} \div 120$ (if previously enabled)

(table continues...)

(continued)

Field	Bits	Type	Description
SRICSDIV	29	rh	**SRI cyber security clock divider** This bit shows the actual setting of the SRICS frequencies. 0_B $f_{SRICS} = f_{SRI}$; 1_B $f_{SRICS} = f_{SRI} \div 2$;
UP	30	w	**Update request** Writing a '1' to this bit sets the .LCK bit and requests an update transfer for SYSCCUCON0 and SYCCUCON1 register content to the system clock dividers. Only one UP bit must be set for either SYSCCUCON0 or SYSCCUCON1 to trigger the transfer. This bit always reads as zero. 0_B No action 1_B Start transfer
0	23:21, 31	r	**Reserved** Read as reset value; shall be written with reset value.

Table 1327 Access mode restrictions of CLOCK_SYSCCUCON0 sorted by descending priority

Mode name	Access mode		Description
CCUSTAT.LCK == 0	rw	FSI2DIV, FSIDIV, LPDIV, RSVD4, SPBDIV, SRIDIV, STMDIV, TPBDIV	Normal access
	w	UP	
(default)	r	FSI2DIV, FSIDIV, LPDIV, RSVD4, SPBDIV, SRIDIV, STMDIV, TPBDIV, UP	Limited access

Table 1328 Reset values of CLOCK_SYSCCUCON0

Reset type	Reset value	Note
After Boot-FW value	0211 1102$_H$	

32.8.4.9 System CCU clock control register 1

CLOCK_SYSCCUCON1 Offset address: 0414_H

System CCU clock control register 1 Reset values see: Table 1330

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	**UP**	**0**		**CANXLHDIV**				**LETHDIV**				**MCANHDIV**			
r	w	r		rw				rw				rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
eGTMDIV				**RSVD8**				**RSVD4**				**GETHDIV**			
rw				rw				rw				rw			

Field	Bits	Type	Description
GETHDIV	3:0	rw	**GETH divider reload value** The resulting GETH frequency is configured to $f_{GETH} = f_{source0} \div$ GETHDIV for the allowed configurations as listed below. For GETHDIV = 0000_B the clock is shut off. ***Note**: GETHDIV must be enabled (!=0) during an application reset to allow firmware related installation tasks.* 0_H f_{GETH} is stopped 1_H $f_{GETH} = f_{source0} \div 1$ 2_H $f_{GETH} = f_{source0} \div 2$ 3_H $f_{GETH} = f_{source0} \div 3$ 4_H $f_{GETH} = f_{source0} \div 4$ 5_H Reserved, do not use this combination 6_H Reserved, do not use this combination 7_H Reserved, do not use this combination 8_H Reserved, do not use this combination 9_H Reserved, do not use this combination A_H Reserved, do not use this combination B_H Reserved, do not use this combination C_H Reserved, do not use this combination … E_H Reserved, do not use this combination F_H Reserved, do not use this combination
RSVD4	7:4	rw	**Reserved** Read as reset value; shall be written with reset value.
RSVD8	11:8	rw	**Reserved** Read as reset value; shall be written with reset value.

(table continues...)

(continued)

Field	Bits	Type	Description
eGTMDIV	15:12	rw	**eGTM divider reload value** The resulting eGTM frequency is configured to $f_{eGTM} = f_{source0} \div$ eGTMDIV for the allowed configurations as listed below. 0_H f_{eGTM} is stopped 1_H $f_{eGTM} = f_{source0} \div 1$ 2_H $f_{eGTM} = f_{source0} \div 2$ 3_H $f_{eGTM} = f_{source0} \div 3$ 4_H $f_{eGTM} = f_{source0} \div 4$ 5_H $f_{eGTM} = f_{source0} \div 5$ 6_H $f_{eGTM} = f_{source0} \div 6$ 7_H Reserved, do not use this combination 8_H $f_{eGTM} = f_{source0} \div 8$ 9_H Reserved, do not use this combination A_H $f_{eGTM} = f_{source0} \div 10$ B_H Reserved, do not use this combination C_H $f_{eGTM} = f_{source0} \div 12$ D_H Reserved, do not use this combination E_H Reserved, do not use this combination F_H $f_{eGTM} = f_{source0} \div 15$
MCANHDIV	19:16	rw	**MCANH divider reload value** The resulting MCANH frequency is configured to $f_{MCANH} = f_{source0} \div$ MCANHDIV for the allowed configurations as listed below. For MCANHDIV = 0000_B the clock is shut off. 0_H f_{MCANH} is stopped 1_H $f_{MCANH} = f_{source0} \div 1$ 2_H $f_{MCANH} = f_{source0} \div 2$ 3_H $f_{MCANH} = f_{source0} \div 3$ 4_H $f_{MCANH} = f_{source0} \div 4$ 5_H $f_{MCANH} = f_{source0} \div 5$ 6_H $f_{MCANH} = f_{source0} \div 6$ 7_H Reserved, do not use this combination 8_H $f_{MCANH} = f_{source0} \div 8$ 9_H Reserved, do not use this combination A_H $f_{MCANH} = f_{source0} \div 10$ B_H Reserved, do not use this combination C_H $f_{MCANH} = f_{source0} \div 12$ D_H Reserved, do not use this combination E_H Reserved, do not use this combination F_H $f_{MCANH} = f_{source0} \div 15$

(table continues...)

(continued)

Field	Bits	Type	Description
LETHDIV	23:20	rw	**LETH divider reload value** The resulting LETH frequency is configured to $f_{LETH} = f_{source0} \div$ LETHDIV for the allowed configurations as listed below. For LETHDIV = 0000_B, the clock is shut off. 0_H f_{LETH} is stopped 1_H $f_{LETH} = f_{source0} \div 1$ 2_H $f_{LETH} = f_{source0} \div 2$ 3_H $f_{LETH} = f_{source0} \div 3$ 4_H $f_{LETH} = f_{source0} \div 4$ 5_H Reserved, do not use this combination 6_H Reserved, do not use this combination 7_H Reserved, do not use this combination 8_H Reserved, do not use this combination 9_H Reserved, do not use this combination A_H Reserved, do not use this combination B_H Reserved, do not use this combination C_H Reserved, do not use this combination … E_H Reserved, do not use this combination F_H Reserved, do not use this combination
CANXLHDIV	27:24	rw	**CANXLH divider reload value** The resulting CANXLH frequency is configured to $f_{CANXLH} = f_{source0} \div$ CANXLHDIV for the allowed configurations as listed below. For CANXLHDIV = 0000_B the clock is shut off. 0_H f_{CANXLH} is stopped 1_H $f_{CANXLH} = f_{source0} \div 1$ 2_H $f_{CANXLH} = f_{source0} \div 2$ 3_H $f_{CANXLH} = f_{source0} \div 3$ 4_H $f_{CANXLH} = f_{source0} \div 4$ 5_H $f_{CANXLH} = f_{source0} \div 5$ 6_H $f_{CANXLH} = f_{source0} \div 6$ 7_H Reserved, do not use this combination 8_H $f_{CANXLH} = f_{source0} \div 8$ 9_H Reserved, do not use this combination A_H $f_{CANXLH} = f_{source0} \div 10$ B_H Reserved, do not use this combination C_H $f_{CANXLH} = f_{source0} \div 12$ D_H Reserved, do not use this combination E_H Reserved, do not use this combination F_H $f_{CANXLH} = f_{source0} \div 15$

(table continues...)

(continued)

Field	Bits	Type	Description
UP	30	w	**Update request** Writing a '1' to this bit sets the .LCK bit and requests an update transfer for SYSCCUCON0 and SYCCUCON1 register content to the system clock dividers. Only one UP bit must be set for either SYSCCUCON0 or SYSCCUCON1 to trigger the transfer. This bit always reads as zero. 0_B No action 1_B Start transfer
0	29:28, 31	r	**Reserved** Read as reset value; shall be written with reset value.

Table 1329 Access mode restrictions of CLOCK_SYSCCUCON1 sorted by descending priority

Mode name	Access mode		Description
CCUSTAT.LCK == 0	rw	CANXLHDIV, GETHDIV, LETHDIV, MCANHDIV, RSVD4, RSVD8, eGTMDIV	Normal access
	w	UP	
(default)	r	CANXLHDIV, GETHDIV, LETHDIV, MCANHDIV, RSVD4, RSVD8, UP, eGTMDIV	Limited access

Table 1330 Reset values of CLOCK_SYSCCUCON1

Reset type	Reset value	Note
After Boot-FW value	0242 0004$_H$	

32.8.4.10 Peripheral CCU clock control register 0

***Note**:*

***Note**: Writing to this register sets the .LCK bit and automatically triggers an update transfer of the register value to the peripheral clock divider logic.*

CLOCK_PERCCUCON0 Offset address: 0420$_H$

Peripheral CCU clock control register 0 Reset values see: Table 1332

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PPUDIV				I2CDIV				0		CLKSELQSPI		QSPIDIV			
rw				rw				r		rw		rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		CLKSELMSC		MSCDIV				0		CLKSELMCAN		MCANDIV			
r		rw		rw				r		rw		rw			

Field	Bits	Type	Description
MCANDIV	3:0	rw	**MCAN divider reload value** The intermediate MCAN frequency is configured to $f_{MCANI} = f_{source1} \div$ MCANDIV for the allowed configurations as listed below. For MCANDIV = 0000_B the clock is shut off. 0_H f_{MCANI} is stopped 1_H $f_{MCANI} = f_{source1}$ 2_H $f_{MCANI} = f_{source1} \div 2$ 3_H $f_{MCANI} = f_{source1} \div 3$ 4_H $f_{MCANI} = f_{source1} \div 4$ 5_H $f_{MCANI} = f_{source1} \div 5$ 6_H $f_{MCANI} = f_{source1} \div 6$ 7_H Reserved, do not use this combination 8_H $f_{MCANI} = f_{source1} \div 8$ 9_H Reserved, do not use this combination A_H $f_{MCANI} = f_{source1} \div 10$ B_H Reserved, do not use this combination C_H $f_{MCANI} = f_{source1} \div 12$ D_H Reserved, do not use this combination E_H Reserved, do not use this combination F_H $f_{MCANI} = f_{source1} \div 15$
CLKSELMCAN	5:4	rw	**Clock selection for MCAN** This bit-field defines the clock source that is used for the resulting clock generation of f_{mcan}. ***Note**: For switching between two non-zero configurations the following sequence has to be applied: First step is to switch to 00_B. Second step is to switch to the new target configuration.* 00_B f_{MCAN} clock is stopped 01_B f_{MCANI} is used as clock source f_{MCAN} 10_B f_{OSC} is used as clock source f_{MCAN} 11_B Reserved, do not use this combination

(table continues...)

(continued)

Field	Bits	Type	Description
MSCDIV	11:8	rw	**MSC divider reload value** The resulting MSC frequency is configured to $f_{MSC} = f_{sourcemsc} \div$ MSCDIV for the allowed configurations as listed below. For MSCDIV = 0000_B the clock is shut off. 0_H f_{MSC} is stopped 1_H $f_{MSC} = f_{sourcemsc}$ 2_H $f_{MSC} = f_{sourcemsc} \div 2$ 3_H $f_{MSC} = f_{sourcemsc} \div 3$ 4_H $f_{MSC} = f_{sourcemsc} \div 4$ 5_H $f_{MSC} = f_{sourcemsc} \div 5$ 6_H $f_{MSC} = f_{sourcemsc} \div 6$ 7_H Reserved, do not use this combination 8_H $f_{MSC} = f_{sourcemsc} \div 8$ 9_H Reserved, do not use this combination A_H $f_{MSC} = f_{sourcemsc} \div 10$ B_H Reserved, do not use this combination C_H $f_{MSC} = f_{sourcemsc} \div 12$ D_H Reserved, do not use this combination E_H Reserved, do not use this combination F_H $f_{MSC} = f_{sourcemsc} \div 15$
CLKSELMSC	13:12	rw	**Clock selection for MSC** This bit-field defines the clock source that is used for the clock generation of $f_{sourcemsc}$. ***Note**: For switching between two non-zero configurations the following sequence has to be applied: First step is to switch to 00_B. Second step is to switch to the new target configuration.* 00_B f_{MSC} clock is stopped 01_B $f_{source1}$ is used as clock source $f_{SOURCEMSC}$ 10_B $f_{source2}$ is used as clock source $f_{SOURCEMSC}$ 11_B Reserved, do not use this combination

(table continues...)

(continued)

Field	Bits	Type	Description
QSPIDIV	19:16	rw	**QSPI divider reload value** The resulting QSPI frequency is configured to $f_{QSPI} = f_{sourceqspi} \div$ QSPIDIV for the allowed configurations as listed below. For QSPIDIV = 0000_B the clock is shut off. 0_H f_{QSPI} is stopped 1_H $f_{QSPI} = f_{sourceqspi}$ 2_H $f_{QSPI} = f_{sourceqspi} \div 2$ 3_H $f_{QSPI} = f_{sourceqspi} \div 3$ 4_H $f_{QSPI} = f_{sourceqspi} \div 4$ 5_H $f_{QSPI} = f_{sourceqspi} \div 5$ 6_H $f_{QSPI} = f_{sourceqspi} \div 6$ 7_H Reserved, do not use this combination 8_H $f_{QSPI} = f_{sourceqspi} \div 8$ 9_H Reserved, do not use this combination A_H $f_{QSPI} = f_{sourceqspi} \div 10$ B_H Reserved, do not use this combination C_H $f_{QSPI} = f_{sourceqspi} \div 12$ D_H Reserved, do not use this combination E_H Reserved, do not use this combination F_H $f_{QSPI} = f_{sourceqspi} \div 15$
CLKSELQSPI	21:20	rw	**Clock selection for QSPI** This bit-field defines the clock source that is used for the clock generation of $f_{sourceqspi}$. ***Note**: For switching between two non-zero configurations the following sequence has to be applied: First step is to switch to 00_B. Second step is to switch to the new target configuration.* 00_B f_{QSPI} clock is stopped 01_B $f_{source1}$ is used as clock source $f_{SOURCEQSPI}$ 10_B $f_{source2}$ is used as clock source $f_{SOURCEQSPI}$ 11_B Reserved, do not use this combination

(table continues...)

(continued)

Field	Bits	Type	Description
I2CDIV	27:24	rw	**I2C divider reload value** The resulting I2C frequency is configured to $f_{I2C} = f_{SOURCE2} \div$ I2CDIV for the allowed configurations as listed below. For I2CDIV = 0000_B the clock is shut off. 0_H f_{I2C} is stopped 1_H $f_{I2C} = f_{source2} \div 1$ 2_H $f_{I2C} = f_{source2} \div 2$ 3_H $f_{I2C} = f_{source2} \div 3$ 4_H $f_{I2C} = f_{source2} \div 4$ 5_H $f_{I2C} = f_{source2} \div 5$ 6_H $f_{I2C} = f_{source2} \div 6$ 7_H Reserved, do not use this combination 8_H $f_{I2C} = f_{source2} \div 8$ 9_H Reserved, do not use this combination A_H $f_{I2C} = f_{source2} \div 10$ B_H Reserved, do not use this combination C_H $f_{I2C} = f_{source2} \div 12$ D_H Reserved, do not use this combination E_H Reserved, do not use this combination F_H $f_{I2C} = f_{source2} \div 15$
PPUDIV	31:28	rw	**PPU divider reload value** The resulting PPU frequency is configured to $f_{PPU} = f_{SOURCEPPU} \div$ PPUDIV for the allowed configurations as listed below. For PPUDIV = 0000_B the clock is shut off. 0_H f_{PPU} is stopped 1_H $f_{PPU} = f_{sourceppu} \div 1$ 2_H $f_{PPU} = f_{sourceppu} \div 2$ 3_H $f_{PPU} = f_{sourceppu} \div 3$ 4_H $f_{PPU} = f_{sourceppu} \div 4$ 5_H $f_{PPU} = f_{sourceppu} \div 5$ 6_H $f_{PPU} = f_{sourceppu} \div 6$ 7_H Reserved, do not use this combination 8_H $f_{PPU} = f_{sourceppu} \div 8$ 9_H Reserved, do not use this combination A_H $f_{PPU} = f_{sourceppu} \div 10$ B_H Reserved, do not use this combination C_H $f_{PPU} = f_{sourceppu} \div 12$ D_H Reserved, do not use this combination E_H Reserved, do not use this combination F_H $f_{PPU} = f_{sourceppu} \div 15$
0	7:6, 15:14, 23:22	r	**Reserved** Read as 0; must be written with 0.

Table 1331 Access mode restrictions of CLOCK_PERCCUCON0 sorted by descending priority

Mode name	Access mode		Description
CCUSTAT.LCK == 0	rw	CLKSELMCAN, CLKSELMSC, CLKSELQSPI, I2CDIV, MCANDIV, MSCDIV, PPUDIV, QSPIDIV	Normal access
(default)	r	CLKSELMCAN, CLKSELMSC, CLKSELQSPI, I2CDIV, MCANDIV, MSCDIV, PPUDIV, QSPIDIV	Limited access

Table 1332 Reset values of CLOCK_PERCCUCON0

Reset type	Reset value	Note
After Boot-FW value	5310 1015$_H$	

32.8.4.11 Peripheral CCU clock control register 1

***Note**:*

***Note**: Writing to this register sets the .LCK bit and automatically triggers an update transfer of the register value to the peripheral clock divider logic.*

CLOCK_PERCCUCON1 Offset address: 0424$_H$

Peripheral CCU clock control register 1 Reset values see: Table 1334

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0			LETH100PERON	HSCTPERON	SDMMCPERON	XSPIPERON	ERAYPERON	CANXLDIV				0			ADCPERON
r			rw	rw	rw	rw	rw	rw				r			rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CLKSELCANXL		CLKSELASCLINS		ASCLINSDIV				XSPISLDIV				ASCLINFDIV			
rw		rw		rw				rw				rw			

Field	Bits	Type	Description
ASCLINFDIV	3:0	rw	**ASCLIN fast divider reload value** The resulting ASCLIN frequency is configured to $f_{ASCLINF} = f_{source2} \div$ ASCLINFDIV for the allowed configurations as listed below. For ASCLINFDIV = 0000_B the clock is shut off. 0_H $f_{ASCLINF}$ is stopped 1_H $f_{ASCLINF} = f_{source2} \div 1$ 2_H $f_{ASCLINF} = f_{source2} \div 2$ 3_H $f_{ASCLINF} = f_{source2} \div 3$ 4_H $f_{ASCLINF} = f_{source2} \div 4$ 5_H $f_{ASCLINF} = f_{source2} \div 5$ 6_H $f_{ASCLINF} = f_{source2} \div 6$ 7_H Reserved, do not use this combination 8_H $f_{ASCLINF} = f_{source2} \div 8$ 9_H Reserved, do not use this combination A_H $f_{ASCLINF} = f_{source2} \div 10$ B_H Reserved, do not use this combination C_H $f_{ASCLINF} = f_{source2} \div 12$ D_H Reserved, do not use this combination E_H Reserved, do not use this combination F_H $f_{ASCLINF} = f_{source2} \div 15$
XSPISLDIV	7:4	rw	**xSPISL divider reload value** The xSPISL (slow) frequency is configured to $f_{xSPISl} = f_{source3} \div$ XSPISLDIV for the allowed configurations as listed below. For XSPISLDIV = 00_B the clock is shut off. 0_H f_{xSPISL} is stopped 1_H $f_{xSPISL} = f_{source3} \div 1$ 2_H $f_{xSPISL} = f_{source3} \div 2$ 3_H Reserved, do not use this combination 4_H $f_{xSPISL} = f_{source3} \div 4$ 5_H Reserved, do not use this combination 6_H Reserved, do not use this combination 7_H Reserved, do not use this combination 8_H Reserved, do not use this combination 9_H Reserved, do not use this combination A_H Reserved, do not use this combination B_H Reserved, do not use this combination C_H Reserved, do not use this combination … E_H Reserved, do not use this combination F_H Reserved, do not use this combination

(table continues...)

(continued)

Field	Bits	Type	Description
ASCLINSDIV	11:8	rw	**ASCLIN slow divider reload value** The intermediate ASCLIN frequency is configured to $f_{ASCLINSI} = f_{source1} \div$ ASCLINSDIV for the allowed configurations as listed below. For ASCLINSDIV = 0000_B the clock is shut off. 0_H $f_{ASCLINSI}$ is stopped 1_H $f_{ASCLINSI} = f_{source1}$ 2_H $f_{ASCLINSI} = f_{source1} \div 2$ 3_H $f_{ASCLINSI} = f_{source1} \div 3$ 4_H $f_{ASCLINSI} = f_{source1} \div 4$ 5_H $f_{ASCLINSI} = f_{source1} \div 5$ 6_H $f_{ASCLINSI} = f_{source1} \div 6$ 7_H Reserved, do not use this combination 8_H $f_{ASCLINSI} = f_{source1} \div 8$ 9_H Reserved, do not use this combination A_H $f_{ASCLINSI} = f_{source1} \div 10$ B_H Reserved, do not use this combination C_H $f_{ASCLINSI} = f_{source1} \div 12$ D_H Reserved, do not use this combination E_H Reserved, do not use this combination F_H $f_{ASCLINSI} = f_{source1} \div 15$
CLKSELASCLINS	13:12	rw	**Clock selection for ASCLINS** This bit-field defines the clock source that is used for the resulting clock generation of $f_{ASCLINS}$. ***Note**: For switching between two non-zero configurations the following sequence has to be applied: First step is to switch to 00_B. Second step is to switch to the new target configuration.* 00_B $f_{ASCLINS}$ clock is stopped 01_B $f_{ASCLINSI}$ is used as clock $f_{ASCLINS}$ 10_B f_{OSC} is used as clock $f_{ASCLINS}$ 11_B Reserved, do not use this combination
CLKSELCANXL	15:14	rw	**Clock selection for CANXL** This bit-field defines the clock source that is used for the resulting clock generation of f_{canxl}. ***Note**: For switching between two non-zero configurations the following sequence has to be applied: First step is to switch to 00_B. Second step is to switch to the new target configuration.* 00_B f_{CANXL} clock is stopped 01_B f_{CANXLI} is used as clock source f_{CANXL} 10_B f_{OSC} is used as clock source f_{CANXL} 11_B Reserved, do not use this combination

(table continues...)

(continued)

Field	Bits	Type	Description
ADCPEREON	16	rw	**Power safe switch-off for ADC clock** This bit is used to control the ADC peripheral clock f_{ADC} for power saving purposes if the logic is not used by the application. 0_B f_{ADC} is stopped 1_B $f_{ADC} = f_{source1}$
CANXLDIV	23:20	rw	**CANXL divider reload value** The intermediate CANXL frequency is configured to $f_{CANXLI} = f_{source1} \div$ CANXLDIV for the allowed configurations as listed below. For CANXLDIV = 0000_B the clock is shut off. 0_H f_{CANXLI} is stopped 1_H $f_{CANXLI} = f_{source1}$ 2_H $f_{CANXLI} = f_{source1} \div 2$ 3_H $f_{CANXLI} = f_{source1} \div 3$ 4_H $f_{CANXLI} = f_{source1} \div 4$ 5_H $f_{CANXLI} = f_{source1} \div 5$ 6_H $f_{CANXLI} = f_{source1} \div 6$ 7_H Reserved, do not use this combination 8_H $f_{CANXLI} = f_{source1} \div 8$ 9_H Reserved, do not use this combination A_H $f_{CANXLI} = f_{source1} \div 10$ B_H Reserved, do not use this combination C_H $f_{CANXLI} = f_{source1} \div 12$ D_H Reserved, do not use this combination E_H Reserved, do not use this combination F_H $f_{CANXLI} = f_{source1} \div 15$
ERAYPEREON	24	rw	**Power safe switch-off for ERAY clock** This bit is used to control the ERAY peripheral clock f_{ERAY} for power saving purposes if the logic is not used by the application. 0_B f_{ERAY} is stopped 1_B $f_{ERAY} = f_{source1} \div 2$
XSPIPEREON	25	rw	**Power safe switch-off for xSPI clock** This bit is used to control the xSPI peripheral clock f_{xSPI} for power saving purposes if the logic is not used by the application. 0_B f_{XSPI} is stopped 1_B $f_{XSPI} = f_{source3}$
SDMMCPEREON	26	rw	**Power safe switch-off for SDMMC clock** This bit is used to control the SDMMC peripheral clock f_{SDMMC} for power saving purposes if the logic is not used by the application. 0_B f_{SDMMC} is stopped 1_B $f_{SDMMC} = f_{source2} \div 2$
HSCTPEREON	27	rw	**HSTC IVR power down** Switch off pre-regulator for HSCT 0_B Pre-regulator powered down 1_B Pre-regulator powered up

(table continues...)

(continued)

Field	Bits	Type	Description
LETH100PERON	28	rw	**Power safe switch-off for LETH100 clock** This bit is used to control the LETH reference clock $f_{LETH100}$ for power saving purposes if the logic is not used by the application. 0_B $f_{LETH100}$ is stopped 1_B $f_{LETH100} = f_{source2}$
0	19:17, 31:29	r	**Reserved** Read as 0; must be written with 0.

Table 1333 Access mode restrictions of CLOCK_PERCCUCON1 sorted by descending priority

Mode name	Access mode		Description
CCUSTAT.LCK == 0	rw	ADCPERON, ASCLINFDIV, ASCLINSDIV, CANXLDIV, CLKSELASCLINS, CLKSELCANXL, ERAYPERON, LETH100PERON, SDMMCPERON, XSPIPERON, XSPISLDIV	Normal access
(default)	r	ADCPERON, ASCLINFDIV, ASCLINSDIV, CANXLDIV, CLKSELASCLINS, CLKSELCANXL, ERAYPERON, LETH100PERON, SDMMCPERON, XSPIPERON, XSPISLDIV	Limited access

Table 1334 Reset values of CLOCK_PERCCUCON1

Reset type	Reset value	Note
After Boot-FW value	1551 1201_H	

32.8.4.12 External clock control register

***Note**: This register is used to control the external clock routing logic.*

CLOCK_EXTCON Offset address: 0430_H

External clock control register System Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	DIV1							0	SEL1					NSEL1	EN1
r	rw							r	rw					rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0									SEL0					0	EN0
r									rw					r	rw

Field	Bits	Type	Description
EN0	0	rw	**Output enable for EXTCLK0** ***Note**: If the generation of the external clock on output for pin EXTCLK0 is disabled, the signal is tied to zero.* 0_B No external clock is provided on output. 1_B The configured external clock is provided on output.
SEL0	6:2	rw	**Clock select for EXTCLK0** This bit-field defines the clock source that is selected as output for pin EXTCLK0. 00_H f_{BACK} 01_H f_{RAMP} 02_H f_{OSC} 03_H f_{PLL0} 04_H f_{PLL1} 05_H f_{PLL2} 06_H f_{PLL3} 07_H f_{FRAC} 08_H f_{SPB} $0A_H$ f_{TPB} $0B_H$ f_{SRI} $0C_H$ f_{FSI} $0E_H$ f_{STM} 10_H f_{eGTM} 11_H f_{GETH} 12_H f_{MCANH} 16_H f_{LETH} 17_H f_{PPU} 18_H f_{CANXLH} **others,** Reserved, do not use
EN1	16	rw	**Output enable for EXTCLK1** ***Note**: If the generation of the external clock on output for pin EXTCLK1 is disabled, the signal is tied to zero.* 0_B No external clock is provided on output. 1_B The configured external clock is provided on output.
NSEL1	17	rw	**Negation selection for EXTCLK1** Selects if the external clock output for pin EXTCLK1 is inverted or not. 0_B The external clock output is inverted. 1_B The external clock output is not inverted

(table continues...)

(continued)

Field	Bits	Type	Description
SEL1	22:18	rw	**External clock select for EXTCLK1** This bit-field defines the clock source that is selected as output for pin EXTCLK1. 00_H f_{BACK} 01_H f_{RAMP} 02_H f_{OSC} 03_H f_{PLL0} 04_H f_{PLL1} 05_H f_{PLL2} 06_H f_{PLL3} 07_H f_{DIV1} 09_H f_{MT0} from the ERAY module $0A_H$ f_{ERAY} $0B_H$ f_{MCAN} $0C_H$ f_{ADC} $0D_H$ f_{QSPI} $0E_H$ f_{MSC} $0F_H$ f_{I2C} 10_H $f_{ASCLINF}$ 11_H $f_{ASCLINS}$ 13_H f_{XSPI} 14_H f_{SDMMC} 15_H f_{PPU} 16_H f_{CANXL} 17_H $f_{LETH100}$ 18_H f_{XSPISL} **others,** Reserved, do not use
DIV1	30:24	rw	**Clock divider for EXTCLK1** This value defines the reload value of the divider that generates f_{DIV1} out of f_{SPB} (f_{OUT}=f_{SPB}÷(DIV1+1). The divider itself is cleared each time bit EN1 is cleared.
0	1, 15:7, 23, 31	r	**Reserved** Read as 0; must be written with 0.

32.8.5 TC4Dx CLOCK connectivity

The table below lists interfaces to and from this functional block to other blocks in the device.

Table 1335 List of CLOCK interface signals

Interface signals	I/O	Description
PORTS_CLOCK_fPADSYSCLK	In	Clock input from external
PORTS_CLOCK_fXTAL1	In	Oscillator input from external

(table continues...)

Table 1335 (continued) List of CLOCK interface signals

Interface signals	I/O	Description
CLOCK_PORTS_fXTAL2	Out	TX interface for oscillator feedback path
CLOCK_FB_fSRI	Out	System Clock provided to SRI crossbars
CLOCK_FB_fSPB	Out	System Clocks provided to SPB interconnect
CLOCK_FB_fTPB	Out	System Clocks provided to TPB interconnect
CLOCK_FB_fRCB	Out	System Clocks provided to RCB interconnect
CLOCK_FB_fCPB	Out	System Clocks provided to CPB interconnect
CLOCK_EGTM_feGTM	Out	System Clocks provided to eGTM
CLOCK_CPU_fSTM	Out	System Clocks provided to system timer
CLOCK_GETH_fGETH	Out	System Clocks provided to gigabit ethernet module
CLOCK_LETH_fLETH	Out	System Clocks provided to light ethernet module
CLOCK_LETH_fLETH100	Out	Peripheral Clocks provided to light ethernet module
CLOCK_CAN_fMCANH	Out	System Clocks provided to MCAN module
CLOCK_CANXL_fCANXLH	Out	System Clocks provided to CAN XL module
CLOCK_ERAY_fERAY	Out	Peripheral Clocks provided to ERAY module
CLOCK_MSC_fMSC	Out	Peripheral Clocks provided to MSC module
CLOCK_CAN_fMCAN	Out	Peripheral Clocks provided to MCAN module
CLOCK_CANXL_fCANXL	Out	Peripheral Clocks provided to CAN XLmodule
CLOCK_ASCLIN_fASCLINF	Out	Peripheral Clocks provided to ASCLIN module
CLOCK_ASCLIN_fASCLINS	Out	Peripheral Clocks provided to ASCLIN module
CLOCK_QSPI_fQSPI	Out	Peripheral Clocks provided to QSPI module
CLOCK_ADC_fADC	Out	Peripheral Clocks provided to converter system
CLOCK_I2C_fI2C	Out	Peripheral Clocks provided to I2C module
CLOCK_HSPHY_fxSPI	Out	Peripheral Clocks provided to HSPHY for xSPI module. Present only when xSPI high speed mode is supported.
CLOCK_XSPI_fxSPISL	Out	Peripheral Clocks provided to xSPI module
CLOCK_SDMMC_fSDMMC	Out	Peripheral Clocks provided to SDMMC module
CLOCK_HSPHY_fOSC	Out	Peripheral Clocks provided to HSPHY module
CLOCK_HSPHY_fHSPHY0ALTREF	Out	Peripheral Clocks provided to HSPHY module
CLOCK_HSPHY_fHSPHY1ALTREF	Out	Peripheral Clocks provided to HSPHY module
CLOCK_PORTS_fEXTCLK	Out	TX interface for transmission of status, debug and test data
ERAY0_CLOCK_fMT	In	Time base to be used in for external clock distribution
ERAY1_CLOCK_fMT	In	Time base to be used in for external clock distribution
CLOCK_EGTM_feGTMhalf	Out	System Clocks provided to eGTM

(table continues...)

Table 1335 (continued) List of CLOCK interface signals

Interface signals	I/O	Description
CLOCK_PPU_fPPU	Out	System clock provided to PPU
CLOCK_LLI_fPPU	Out	System clock provided to PPU
CLOCK_HSSL_fHSCT	Out	HSCT Clock
CLOCK_HSSL_REFCLK	Out	Reference Clock for HSSL
CLOCK_PCIE_fOSC	Out	XTAL Clock to PCIE
CLOCK_DMA_fSTM	Out	System Clock provided to SDMA
CLOCK_SCU_EXTCLK0	Out	EXTCLK0 provided to SCU
CLOCK_SCU_EXTCLK1	Out	EXTCLK1 provided to SCU
PMS_CLOCK_fBACK	In	Backup clock

32.8.6 TC4Dx CLOCK revision history

Reference	Description of change(s)
Date range: 2024-10-16 to 2025-02-28	
Crystal oscillator control register	• OSCCON.SUPM is reserved and must not be used. This reference is now removed from the register description.
Date range: 2024-05-15 to 2024-10-15	
Clock control unit	• Added feGTMhalf in module clock overview and divider ratio limitations • Update to connectivity
OSCCON	• OSCCON MODE descriptions aligned to descriptions in Crystal Oscillator chapter
PERPLLCON1, SYSPLLCON1	• Note corrected from SYSPLLCON1 and PERPLLCON1 KxPREDIV
CCUCON	• CCUCON Reset value was moved to the derivative specific section and added a note to that effect
TC4Dx CLOCK features	• Removed sentence about SYSCCUCONx and PERCCUCONx

33 Logic-Test (TRI)

LBIST (Logic Built-In Self Test) is an on-chip hardware mechanism that can be used to detect latent faults in the microcontroller. The LBIST implementation in the AURIX™ TC4xx platform allows the execution of periodic self-tests of the microcontroller logic. It is accessed through registers in the test register interface (TRI) unit.

33.1 Feature list

- Logic-BIST (built-in self-test) provides a full, automatic test of the digital components within the device
- To support Logic-BIST, the chip design is clustered into hierarchical scan domains
- Two LBIST test scenarios are supported:
  1. Key-on LBIST: Targeting only stuck-at faults of on chip functional safety logic within each hierarchical domain
  2. Key-off LBIST: Targeting stuck-at (permanent) faults of all logic within each hierarchical domain
- Key-on LBIST features:
  - Can be executed automatically during the boot sequence of the device (using default firmware configuration) or started manually based on user software trigger
  - Key-On LBIST only involves safety-relevant flip-flops (~8-10% of total flop count) within any hierarchical domain. Other flip-flops are not toggled
  - Target stuck-at coverage: 90%
  - Target execution time: t_{LBIST}
  - Because of the reduced toggle rate, multiple hierarchical scan domains of the chip design can be tested in parallel
- Key-off LBIST features:
  - Can be configured and started for a single or multiple hierarchical domains by user application software.
  - Key-off LBIST involves all flip-flops within a hierarchical domain, therefore in case of a multiple domain configuration the key-off LBIST will test the selected domains sequentially
  - Target stuck-at coverage for full digital hierarchical domain: 90%
  - Target execution time for stuck-at test: 50 ms per hierarchical scan domain
- Number of patterns (that is the number of shift and capture sequences) can be configured. For stored Multi Input Shift Register (MISR) signatures a specific key-on and key-off configuration is provided
- After any LBIST execution, the system will execute a LBIST reset and reboot automatically
- Execution status and MISR results are provided for each hierarchical domain after reboot of the device, to be compared against golden signatures by application software
- LBIST power reduction features:
  - Configurable clock frequency for LBIST controller (affecting scan-chain shift and capture speed)
  - Split-shift:
    - Allows to distribute each scan shift cycle to 2 or 4 clock pulses to reduce the current jump for each shift clock pulse
  - Key-On LBIST to cover the functional safety logic (for example the lockstep compare logic in CPU) exclusively involves safety-relevant flip-flops (8-10%) within any hierarchical domain

33.2 Functional overview

Logic-BIST is an automatic structural self-testing mechanism, which is based on the structural scan test approach. It serves two main purposes:

- Safety measures (“key-on”): 90% stuck-at test coverage on all safety monitoring logic (lock-step comparator, ECC, SMU, …) within 5-6 ms execution time at system start-up
- In-system test (“key-off”): 90% stuck-at test coverage on all digital logic in 50 ms execution time per hierarchical scan domain at system shut-down

Due to complexity the device is divided into several hierarchical scan domains. Domain structure (number, complexity and LBIST runtime per domain) varies from device to device. Start of LBIST in any hierarchical domain brings the entire device automatically to scan mode during which no functional operation is possible. During LBIST execution the IOs reflect a static reset behavior.

The execution of Logic built-in self test in the microcontroller application mode is based on the design for test structures implemented for production testing and therefore reusing scan chains, control, and status mechanisms already available in microcontroller. The scan test operation is enabled through dedicated pseudo-random pattern generator (PRPG) modules, which generate patterns to be applied to the corresponding hierarchical domains. The PRPG generates pseudo-random patterns using a linear feedback shift register (LFSR). These patterns are loaded (this mean serially shifted) into the design scan-chains.

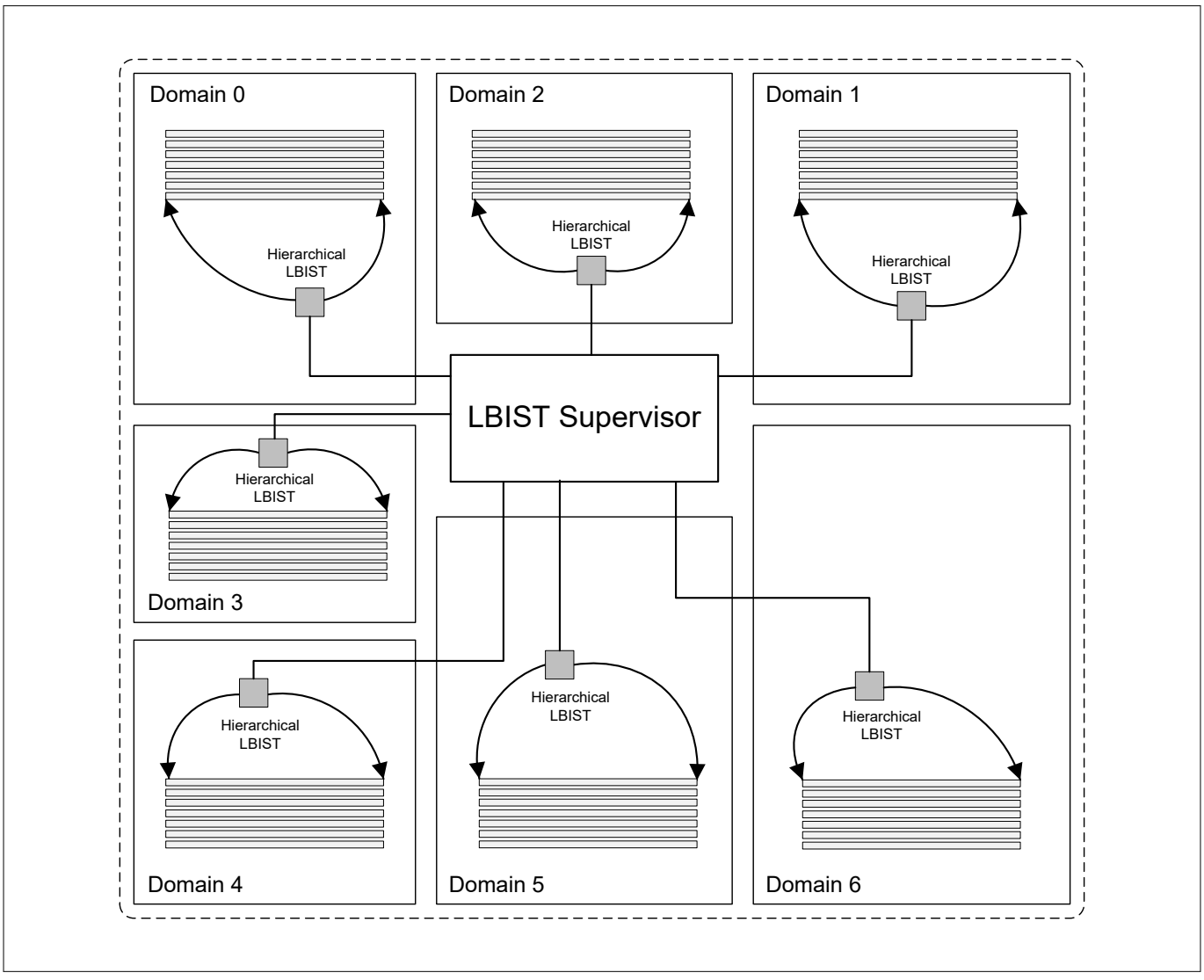


Figure 755 Hierarchical LBIST architecture

It shall be noted, that LBIST is a scan based structural test method which also covers the redundancy logic of the SRAMs within each hierarchical scan domain. Therefore the SRAM contents is not reliable after LBIST execution except for the Stand-by controller (SCR) SRAMs.

In the following sections the general LBIST operation scheme, possible configuration options and application scenarios are explained. In this conjunction also the LBIST integration into a hierarchical scan architecture is shown.

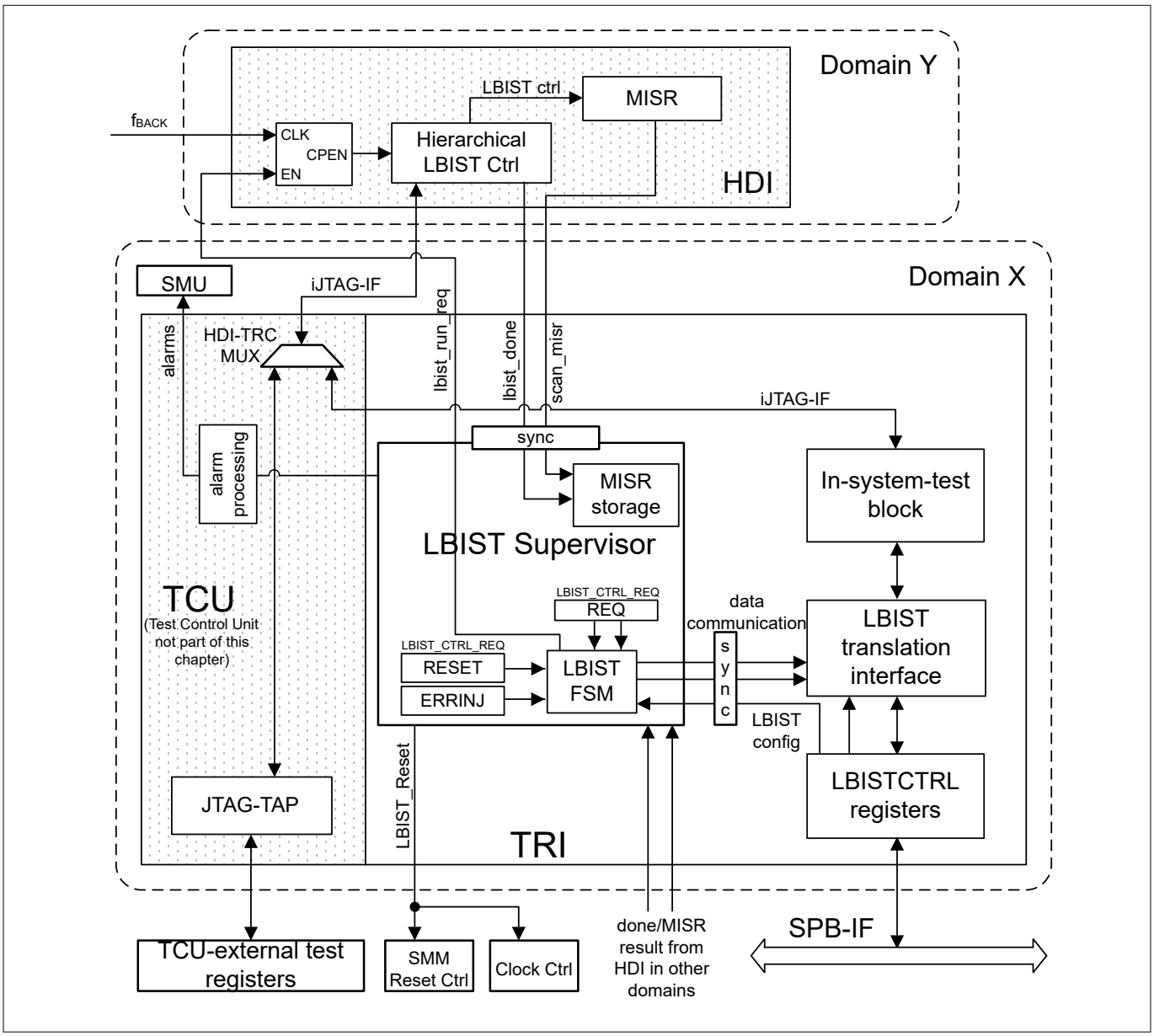


Figure 756 TRI block diagram

LBIST operation is controlled through a central LBIST supervisor logic, which resides in the test register interface (TRI) block. From this central supervisor all LBIST operations are controlled through a single state machine (LBIST FSM). From application side the LBIST operation is accessed by writing and reading specific LBISTCTRL registers through the SPB interface (SPB-IF). In this way a specific LBIST configuration (LBIST config) is handed over to the LBIST supervisor.

Once LBIST operation is started, the LBIST supervisor block configures the hierarchical LBIST controllers (Hierarchical LBIST Ctrl) through an iJTAG interface (iJTAG-IF). These hierarchical LBIST controllers are part of the hierarchical TRI interface (HDI), which is implemented within each domain. The LBIST translation interface and the in-system-test block within TRI will be used to establish a data communication between LBIST supervisor and the hierarchical LBIST controllers.

33 Logic-Test (TRI)

After initialization of the hierarchical LBIST controllers the LBIST supervisor will start each hierarchical LBIST operation individually through a lbist_run_req signal. It will then await the successful termination of the LBIST operation in each domain, which is indicated through individual lbist_done signals. As soon LBIST operation in all domains is finished, the LBIST supervisor block captures the domain individual MISR signature results (scan_misr) and triggers a LBIST reset to the whole system.

After LBIST operation is finished the LBIST FSM has to be set into a safe reset state through a specific RESET bit within the LBISTCTRL registers.

Specific alarm signals are defined to indicate any unwanted LBIST activity within the TRI to SMU. These alarms are processed through the test control unit (TCU) and can be triggered manually for testing purpose through the ERRINJ bit within the LBISTCTRL registers.

For the alarm description please refer to the alarm mapping tables in the SMU functional block and the Safety Manual.

Related information

TC4Dx SMU alarm mapping tables on page 7233

33.3 Functional description

This section describes the general operation scheme as well as the configuration options and the application scenarios of the LBIST functional block.

Related information

Registers overview - TRI (ascending offset address) on page 5694

33.3.1 Software interface

In AURIX™ 3rd Generation devices the TRI has a dedicated register interface which is connected to the SPB interconnect. It connects to the configuration and status registers of the clocking system and implements the programming and software interface.

General information about the access protection scheme can be found in the related information.

Related information

Access protection on page 63
PROT mechanism on page 64
Access protection unit (APU) on page 76

33.3.2 LBIST operation scheme

The Logic-BIST function is a structural test method for automatic in-system health check of the digital design part. LBIST schemes use on-chip circuitry to generate test stimuli and analyze test responses, without any help from a chip-external test system.

LBIST covers all digital logic of the device with the following exceptions:

- Logic in power management system, linked to the EVR and standby controller
- A part on the test register interface unit (TRI), including the LBIST controller

The LBIST structure applies pseudo-random patterns generated by a PRPG (Pseudo-Random Pattern Generator) to a full-scan circuit in parallel and compacts the test responses into a signature with a MISR (Multiple-Input Signature Register).

In this conjunction it shall be noted, that the microcontroller is partitioned into hierarchical domains, from which each one is scan tested independently through a local LBIST controller. This LBIST controller is implemented as part of a local HDI (Hierarchical TRI Interface) block. A central LBIST supervisor block is implemented in the TRI which is responsible for the organized execution of the whole LBIST procedure. From this supervisor all LBIST operations are controlled through a single state machine ('LBIST FSM').

After LBIST execution has been triggered from system-side (through LBIST_CTRL_CFGA ... LBIST_CTRL_CFGF, LBIST_CTRL_DOM and LBIST_CTRL_REQ registers), the LBIST FSM in TRI automatically initializes the domain-individual LBIST controller blocks (Hierarchical LBIST Ctrl) through a chip internal iJTAG network and also switches the hierarchical domain locally into scan mode. This initialization is done by using the LBIST translation interface and the in-system-test block within TRI to establish a data communication between LBIST supervisor and the hierarchical LBIST controllers. If this domain specific scan mode is active, the design's flip-flops are not operated in their functional mode but are organized in scan-chains to allow a serial shift of these pseudo-random patterns into the digital design. For each domain where this is done, the local (hierarchical) LBIST controller in HDI starts LBIST execution as soon as the LBIST supervisor block activated its local operating clock (by changing the corresponding 'lbist_run_req' signal to 1). Each hierarchical LBIST controller will execute the automatic scan test according to its configuration data, concerning number of scan-loads and execution-speed. A specific shift counter keeps track of the number of shift-cycles to issue for each pattern.

The LBIST controller in each hierarchical domain also uses the shift counter to separate the shift cycles from the capture cycles. Once a random pattern has been completely loaded into the digital logic scan-chains, the LBIST controller generates a single capture cycle. In this case the flip-flops are switched back from serial scan-chain

mode to their normal functional behavior, therefore allowing capture of an actual state of the combinatorial logic (see the following figure).

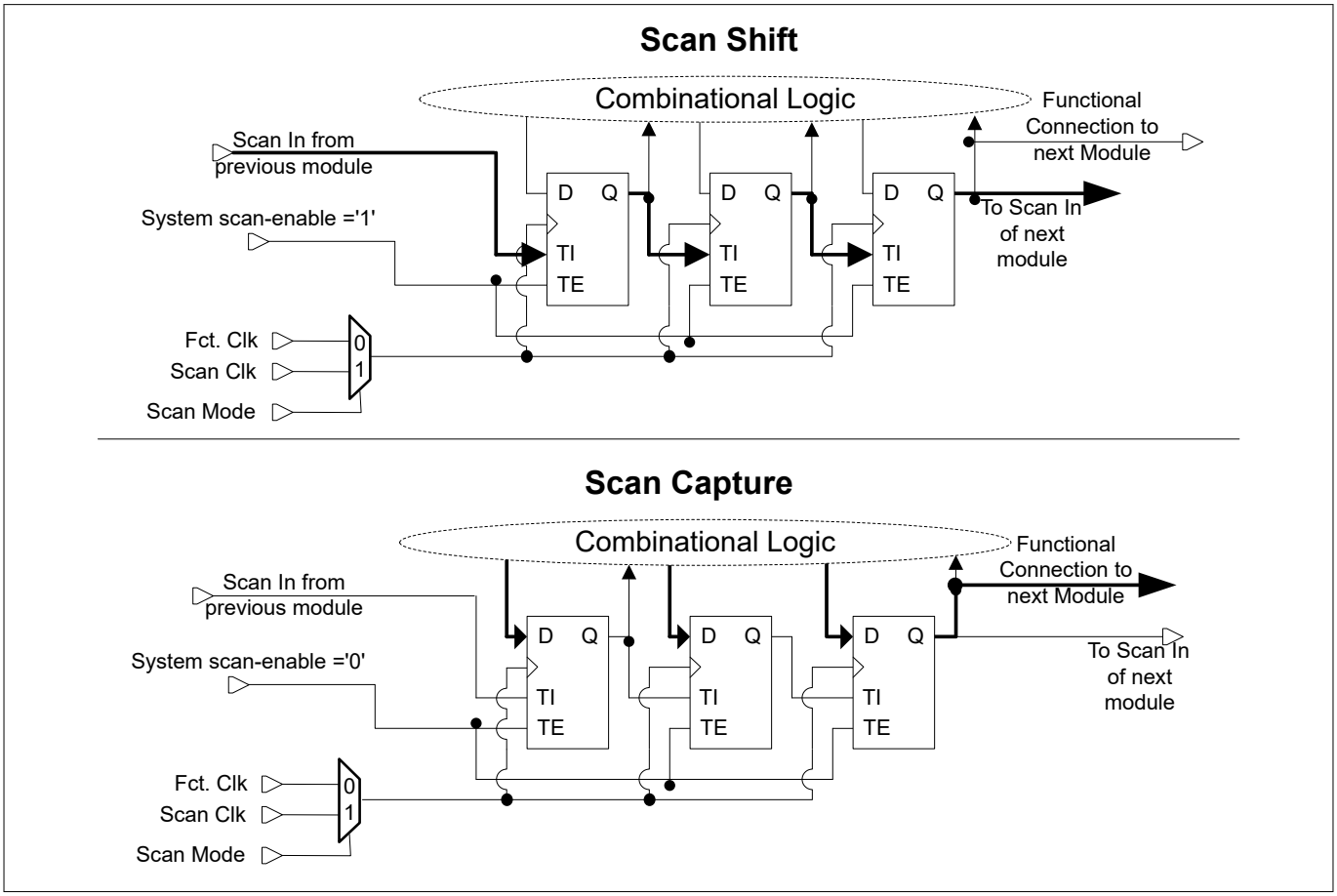


Figure 757 Scan shift and capture process

The pattern for each scan-chain load is generated by the domain-local PRPG as configured through the local LBIST controller. The number of executed scan loads within a domain is selectable by software through the LBIST_CTRL_CFGA.PATTERNS field. This field must be set (together with the other configuration options) before starting LBIST execution. Each domain-local LBIST controller keeps track of how many patterns have been driven through the core and automatically stops the LBIST run when the programmed LBIST_CTRL_CFGA.PATTERNS value has been reached. It will also indicate the successful LBIST termination to the supervisor block its individual 'lbist_done' signal.

During serial scan-chain shift cycles the outputs of the core flip-flop-chains are compressed through a XOR-gate network implemented in specific scan compaction logic and permanently monitored through the MISR block, which generates a unique signature for each hierarchical domain. This signature allows application software to check if the MCU was tested by LBIST with or without errors (see the following figure). The LBIST controller will automatically put the MISR into a static hold condition during the warmup-phase. This means at the start of each LBIST operation 3-4 warmup patterns will be executed during which pseudo-random pattern are shifted through the scan-chains without checking the results. This ensures a proper initialization of the scan-chain values and also increases the run time stability of LBIST. After successful LBIST termination the MISR signature result is also handed over to the central supervisor block in TRI through the domain individual 'scan_misr' vector.

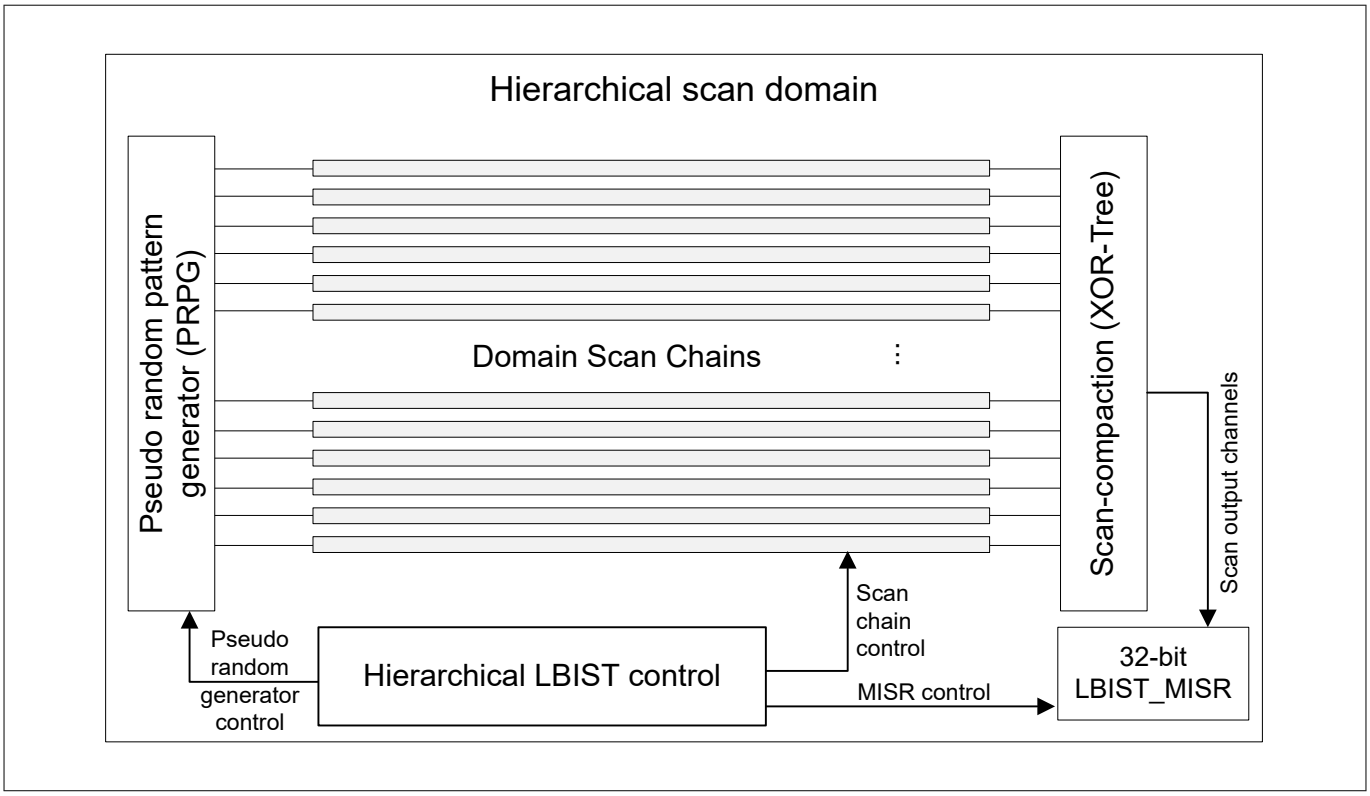


Figure 758 Scan components in LBIST

After LBIST execution is finished for a specific domain the LBIST operation of the next requested domain will be started. After LBIST is finished for all hierarchical domains for which it was requested, the MISR signature results from all these domains can be read out through the LBIST_STAT_RESx_SIG.SIGNATURE fields. Their value is only valid if LBIST_STAT_TEST.DONE='1' and the corresponding LBIST_STAT_TEST.DOMDONE<0-7> bit also indicates a high value, meaning LBIST run terminated successfully. In general the LBIST_STAT_TEST.DOMDONE<0-7> bits indicate for which domains a LBIST procedure has been executed since the last reset. It shall be noted, that for hierarchical scan domains which are not existing on the device the corresponding LBIST_STAT_TEST.DOMDONEx bits will always indicate a stable high value and the corresponding LBIST_STAT_TEST.SIGNATURE result bits show a stable low value.

LBIST execution always terminates with an LBIST reset ('cold_porst_req'), which affects all logical parts except the PMS/EVR system (also in case of a LBIST watchdog timeout counter overflow). From software side this reset will be clearly distinguishable from an external cold PORST (PMS/EVR settings of the startup software are not affected by this LBIST reset). In this way application software can easily verify if the LBIST execution was properly terminated and not interrupted by an external warm or cold PORST. For that purpose the PMS_CLDRSTRIG register in PMS and also the WRMRSTTRIG and WRMRSTTRIGH registers in SMM contain two status bits which reflect the LBIST termination status:

- PMS_CLDRSTRIG.LBPORST: This status bit indicates if LBIST execution was terminated earlier due to an external warm or cold PORST assertion. If the status of this bit-field is 0, the application must still check the PMS_CLDRSTRIG.LBTERM to determine if the LBIST was terminated properly
- PMS_CLDRSTRIG.LBTERM: This bit stores the status if the LBIST execution was terminated by the LBIST supervisor block through a LBIST reset

As mentioned above, a successfully finished LBIST procedure is indicated by the LBIST_STAT_TEST.DONE and the LBIST_STAT_TEST.DOMDONE<1-7> bits. All contents of the LBIST_STAT_TEST and also of the LBIST_STAT_RESx_SIG register fields is automatically restored after any system reset with the captured values from LBIST supervisor block. In case of an external warm or cold power-on resets, these stored values in LBIST supervisor are cleared, which also resets the DOMDONE<1-7> bits and LBIST_STAT_RESx_SIG.SIGNATURE

registers to 0. The same behavior will also be visible if the central LBIST controller is manually reset by setting LBIST_CTRL_REQ.RESET bit to high value.

Attention: Some of the hierarchical domains contain logic, whose power supply can be disabled through the PMS_PD_CON0 and PMS_PD_CON1 registers in PMS. For a proper LBIST execution it must always be ensured that all logic in the targeted domain(s) is actively powered.

Attention: During LBIST operation it is strongly recommended to keep the TESTMODE_N / P20.2 input pin at a static high level. The general purpose input functionality of this pin is not available while a LBIST procedure is underway. In case of an externally triggered warm power-on reset or in case of a power fail the TESTMODE_N / P20.2 input value will determine, if the device is operating in mission mode or in general test mode. So driving it with a different value than static H level during LBIST operation induces a certain risk that the device will erroneously branch into test mode.

33.3.3 LBIST configuration options

The following sections describe the possible LBIST control and configuration options, which can be accessible through the LBIST_CTRL_CFGA ... LBIST_CTRL_CFGF, LBIST_CTRL_DOM and LBIST_CTRL_REQ registers:

- LBIST_CTRL_REQ.REQ, REQRED: If both of these bits are set to high value, the execution of a new LBIST procedure is triggered. All domains which are selected in the LBIST_CTRL_DOM.SEL<0-7> register will be included. Precondition to start a new LBIST run is that the central LBIST controller currently remains in a safe idle state (indicated through LBIST_STAT_TEST.DONE='0'). This state can be reached through an external warm or cold PORST or by setting LBIST_CTRL_REQ.RESET='1'
- LBIST_CTRL_REQ.RESET: This bit resets the central LBIST controller to its safe idle state. After a successful LBIST run, the LBIST alarm signal remains active until this bit has been set to high value. This will also clear the stored MISR results and allow to trigger a new LBIST procedure
- LBIST_CTRL_REQ.FAILCIN: This bit-field has no direct hardware effect on LBIST execution itself, but serves as memory function for the application software. The value written to this field will be stored in the central LBIST controller and mirrored to the LBIST_STAT_TEST.FAILCOUT field after LBIST execution is finished
- LBIST_CTRL_REQ.ERRINJ: If this bit is set to high value, the LBIST and test mode alarm signals will be asserted for testing purpose. This function should only be used to ensure that the alarm signals are still working properly. The bit has no direct influence on the LBIST operation
- LBIST_CTRL_CFGA.PATTERNS: This field defines the LBIST pattern count which will be executed during LBIST procedure for each selected hierarchical domain. The value programmed to the LBIST_CTRL_CFGA.PATTERNS field determines the number of scan-capture phases and not the number of scan-chain load-unload phases: A value of 00001_h will result in two scan-chain loads with one capture in between; a value of 00002_h will result in 3 scan-chain loads with 2 captures and so on). Consequently a value of 00000_h is not valid, because no capture would be executed in this case
- LBIST_CTRL_CFGA.TFREQU: This field allows to determine the iJTAG clock speed. The iJTAG network is used to initialize the hierarchical LBIST controllers after a LBIST procedure has been triggered through LBIST_CTRL_REQ register. Main intention is to operate the iJTAG network with TFREQU=1 → 50 MHz, but slower frequencies like TFREQU=2 → 33 MHz or TFREQU=3 → 25 MHz may also be applied in case of stability problems. Please note that this TFREQU field has only a low influence in the overall LBIST execution time, because it only affects the initialization phase. The operation frequency of the local LBIST controllers inside the hierarchical domains is influenced independently through the LBIST_CTRL_CFGA.FREQU field
- LBIST_CTRL_CFGA.SAFESEL[40]: This bit is directly used to decide if all scannable logic in the selected hierarchical domains will be targeted by LBIST operation (SAFESEL='0') or only that part, which is assigned to primary safety mechanisms (SAFESEL='1'). The SAFESEL bit value will be stored In the central LBIST supervisor block and mirrored to the LBIST_STAT_TEST.SAFEIND field after LBIST execution finished. In this

[40] It shall be noted, that the explicit LBIST test of safety structures is not implemented in all derivatives of the AURIX™ TC4xx platform. The information if a specific "key-on" LBIST scenario is implemented or not, can be found in the device specific Logic-Test (TRI) description.

way the application firmware can recognize which kind of LBIST operation has been triggered. Please note that the safety LBIST option shall only be used for stuck-at tests with a stable 100 MHz clock source

- LBIST_CTRL_CFGA.ENPARA: If LBIST test of safety-mechanism structures is selected (LBIST_CTRL_CFGA.SAFESEL='1'), the value of this bit chooses if LBIST execution of all selected scan domains is triggered in parallel (if ENPARA='1') or one after the other (if ENPARA='0'). The parallel execution is significantly faster but only possible if acceptable from power side. In case of full LBIST test LBIST_CTRL_CFGA.SAFESEL='0' only a sequential execution is possible
- LBIST_CTRL_CFGA.KEYOFF: This bit has no direct influence on the LBIST execution sequence but just serves as indicator for the application software, that the current LBIST procedure was launched under key-off conditions. Consequently it will be stored in the central LBIST supervisor block and is mirrored back to the LBIST_STAT_TEST.KEYIND field after the LBIST execution finished
- LBIST_CTRL_CFGA.SPLITSH: This field influences the on-chip power consumption during LBIST execution. If set to 0x0, all scan-chains are shifted concurrently during LBIST operation, which may cause the power consumption to exceed the one of the maximum power pattern. This might cause stability problems in certain application environments depending on the general supply strategy. If power related stability problems should occur during LBIST execution it is possible to divide the core scan-chains either into two (SPLITSH=0x3) or four (SPLITSH=0x2) partitions, which are then shifted sequentially. Depending on the selected split-shift operation mode, the LBIST execution time will increase by a factor of 2 or 4. This however can be compensated by also increasing the shift operation frequency through FREQU. In fact the average power consumption remains equal if split-shift 2 is selected instead of concurrent shift and the operation frequency is doubled. However the split-shift operation is still of advantage, because it helps to distribute on-chip activity more equally over time
- LBIST_CTRL_CFGA.FREQU: This field will influence the LBIST execution speed (shift-frequency). The local LBIST Controllers are operating on base of a 100 MHz oscillator, whose output frequency can be scaled down by an integer value between 1 to 16 (determined through FREQU hex value + 1). A FREQU value of 1 or higher is recommended for a stable LBIST execution: FREQU=1 → 50 MHz, FREQU=2 → 33 MHz, FREQ=3 → 25 MHz, FREQU=4 → 20 MHz, …, FREQU=15 → 6.25 MHz. Value FREQU=0 [100 MHz] is only to be used in conjunction with LBIST_CTRL_CFGA.SPLITSH=3 or 2 for a stable LBIST execution.
- LBIST_CTRL_CFGB ... LBIST_CTRL_CFGE.LENGTH<0-7>, SLENGTH<0-7>: These register fields determine how many shift-cycles are necessary to load the scan chains of the hierarchical domains. Individual values have to be defined for each of these domains, because their scan implementation differs among the family products. The contents of these registers is automatically initialized when booting the microcontroller and shall not be changed by application software
- LBIST_CTRL_CFGF.LBCGTRAN: The value of this field influences the overall capture clock activity. It allows to weigh between coverage gain and power dissipation during LBIST capture phases. The contents of these registers is automatically initialized when booting the microcontroller and shall not be changed by application software
- LBIST_CTRL_CFGF.TPOBSDIS, TPCTRDIS, OBSSCDIS: These register bits allow to disable the scan control and scan observe points as well as specific scan shift observation FFs, which help to increase the LBIST efficiency. The contents of these registers is automatically initialized when booting the microcontroller and shall not be changed by application software
- LBIST_CTRL_DOM.SEL<0-7>: Through these bits the hierarchical domains are selected, which shall be included into the next LBIST procedure (started by setting LBIST_CTRL_REQ.REQ & REQRED='1')

The following sections describe the LBIST status and result fields which are accessible through the LBIST_STAT_TEST and LBIST_STAT_RESx_SIG registers:

- LBIST_STAT_TEST.DONE: The bit indicates if a LBIST procedure has been executed since the last power-on-reset or after LBIST supervisor block has been restarted through LBIST_CTRL_REQ.RESET
- LBIST_STAT_TEST.SAFEIND, KEYIND, FAILCOUT: The values of these fields reflect the latest settings from LBIST_CTRL_CFGA.SAFESEL, KEYOFF and LBIST_CTRL_REQ.FAILCIN register bits. These are captured in the

central LBIST supervisor block when starting the LBIST procedure to allow an easy identification of the selected LBIST scenario from application side

- LBIST_STAT_TEST.DOMDONEx: These bits indicate if the corresponding hierarchical domain has been included in the previous LBIST run or not For hierarchical scan domains, which are not existing on the device, the corresponding DOMDONEx bits will always indicate a stable high value.
- LBIST_STAT_RESx_SIG.SIGNATURE: After a LBIST procedure has been finished, the MISR signature result from each hierarchical domain is stored in the SIGNATURE field of the corresponding LBIST_STAT_RESx_SIG register. The results are valid as long the LBIST_STAT_TEST.DONE='1' and LBIST_STAT_TEST.DOMDONEx bits are reflecting a high value. For hierarchical scan domains, which are not existing on the device, the corresponding SIGNATURE field will always indicate a stable low value.

Summary for LBIST execution: Before starting a LBIST operation, the requested configuration must be selected through LBIST_CTRL_CFGA ... LBIST_CTRL_CFGF and LBIST_CTRL_DOM registers. As a default these registers will be initialized for a typical "key-on" LBIST operation scenario. In order to trigger a new LBIST procedure the LBIST_CTRL_REQ.LBISTREQ & REQRED bits both must be written with a high value. A new LBIST sequence will only start if the LBIST_STAT_TEST.DONE bit reflects a low value. The high value of LBIST_STAT_TEST.DONE bit is also an indication that the LBIST operation terminated normally and the SIGNATURE values are ready for readout. Once this DONE bit is set to high, its value can be changed back to low by setting the LBIST_CTRL_REQ.RESET bit to high value. This will also reset all SIGNATURE results within the LBIST_STAT_RESx_SIG registers from the previous run.

Example configuration for key on LBIST execution: For an easier understanding of the LBIST configuration possibilities a typical "key-on" setup for LBIST is explained in the following.

- LBIST_CTRL_CFGA.PATTERNS: $0FC0_H$ → It is recommended to select a value in the range of ~4032 scan patterns in order to achieve a test coverage of 90%
- LBIST_CTRL_CFGA.TFREQU: 01_B → Selects a iJTAG frequency of 50 MHz. In case of stability problems during LBIST setup use higher settings like 10_B or 11_B. Do not set this field to 0
- LBIST_CTRL_CFGA.SAFESEL: 1_B → Exclusively test the functional safety logic during "key-on" LBIST for a shorter execution time. This will also indicate to application software that previous LBIST was targeting safety logic after LBIST execution (LBIST_STAT_TEST.SAFEIND = 1_B)
- LBIST_CTRL_CFGA.ENPARA: 1_B → Select parallel LBIST execution for multiple hierarchical domains during "key-on" LBIST for a shorter execution time
- LBIST_CTRL_CFGA.KEYOFF: 0_B → Set this bit to 0_B value to indicate to application software that previous LBIST was executed under "key-on"conditions after LBIST execution (LBIST_STAT_TEST.KEYIND = 0_B)
- LBIST_CTRL_CFGA.SPLITSH: 11_B → Select split-shift with 2 scan partitions to reduce the number of flops, which are toggling concurrently. In case of stability problems during LBIST execution use 10_B (split-shift with 4 scan partitions). This will further reduce the maximum shift power but also increase LBIST execution time
- Behavior of the GPIOs (except shared standby pads) is automatically determined through the PMS_PAD_STAT.ALLTRIST register bit value in PMS. If this bit is set to 0_B value all GPIOs remain in a static weak pull-up behavior during LBIST execution. If it is set it to 1_B value all GPIOs show a static tri-state behavior. The behavior of the standby pads (P32.0-4, P33 and P34.1-5) during LBIST operation depends on the SCR_CON0.PADLBIST register bit value in PMS. The default reset value of this bit determines that those pads are handled identically to all other GPIOs during LBIST. In order to support real time clock functionality it is possible to change the SCR_CON0.PADLBIST bit value before triggering LBIST. This allows to keep the standby pads active while LBIST operation is underway. Specific isolation will ensure that no undefined input data can propagate from the standby pads into any scan testable design areas.
- LBIST_CTRL_CFGA.FREQU: 0_H → Selects a LBIST controller operation frequency of 100 MHz. In case of stability problems during LBIST setup use higher settings like 1_H, 2_H or 3_H. This will reduce the maximum shift power but also increase LBIST execution time
- The values in LBIST_CTRL_CFGB ... LBIST_CTRL_CFGE.LENGTH<0-7>, SLENGTH<0-7> fields are automatically initialized during system boot. Their values shall not be changed from application side

- The values in LBIST_CTRL_CFGF.TPOBSDIS, TPCTRDIS, OBSSCDIS and LBCGTRAN register fields are automatically initialized during system boot. Their values shall not be changed from application side
- LBIST_CTRL_DOM.SEL<0-7> = 1_B → As a default all hierarchical scan domains are selected for LBIST execution. If certain domains shall be excluded from LBIST run the corresponding SEL bit can be set to 0_B value
- LBIST_CTRL_REQ.FAILCIN = 0_H → As a default the fail counter shall be set to 0_H value. In case of MISR result mismatches the LBIST, execution can be repeated with increased FAILCIN values. This value will be stored during LBIST execution and indicated to application software through LBIST_STAT_TEST.FAILCOUT field
- LBIST_CTRL_REQ.REQ, REQ_RED → Write both bits with a 1_B value to start a new LBIST run. This will only work if the LBIST controller is in idle state and the LBIST_STAT_TEST.DONE bit indicates a 0_B value. If this is not the case, reset the LBIST controller first by programming LBIST_CTRL_REQ.RESET = 1_B

33.3.4 LBIST application scenarios

In general the LBIST operation clearly distinguishes between 2 different scenarios:

1. **Safety measure, used during “key-on” phase of the device:** In this case it is necessary to check the on-chip primary safety mechanism structures for a fault free behavior. For that purpose a specific LBIST operation mode is implemented, which only involves the logical structure from safety mechanism logic into the test process. In this way the power consumption of that kind of LBIST test is significantly reduced and it is possible to execute it fast enough to achieve 90% stuck-at test coverage within t_{LBIST}. [41)]
2. **Complete in-system test, used during “key-off” phase of the device:** In contrast to the above mode significantly more time is available for execution of this kind of LBIST test). Consequently it targets the complete logical structure within a selectable hierarchical domain. Therefore the power consumption for the "key-off" LBIST lies significantly above that of the "key-on" scenario. The achievable test coverage depends on the available time budget and so, on the number of scan patterns

From above 2 scenarios the following LBIST operation modes are derived:

1. **Parallel "key-on" safety stuck-at LBIST:** The safety structures of all hierarchical domains are tested for stuck-at faults. Due to the reduced power consumption this is done in a parallel way to minimize execution time. The goal is to achieve a stuck-at coverage of 90% for the safety structures within t_{LBIST}.
2. **Sequential "key-off" stuck-at LBIST with stable frequency:** This kind of LBIST operation always targets the full logic of a selectable hierarchical domain. Due to power limitations it is not possible to test more than one domain at a time. If multiple domains are selected they will be tested sequentially. The goal of this test mode is to achieve a stuck-at coverage of 90% for the complete logic of a hierarchical domain within 50 ms (per domain)

The general interaction of firmware and application software with the LBIST configuration and result registers is illustrated in the following figure. During the MCU boot phase the first ("key-on") LBIST operation mode from the above list is required to be executed. Consequently the LBIST_CTRL_CFGA registers are automatically initialized with this configuration through the firmware. In addition to this the "key-on" LBIST execution can also be triggered automatically through the firmware after an externally triggered cold power-on reset if this is configured in the LBISTEXE.LBEX UCB. In case of a LBIST reset or a warm PORST firmware directly continues with the boot process so all further LBIST activities have to be triggered from application software side.

After an LBIST execution and LBIST reset the application software can read through the LBIST_STAT_TEST.KEYIND flag, if the current boot phase was initiated by a completed “key-off” LBIST operation or not. In case of an active “key-off” flag certain applications might not be able to grant sufficient time to compare the MISR results with any expected values and to react accordingly. In those scenarios the MISR results shall just be saved to the Flash before continuing with the “key-off” sequence. In this way they can

41 It shall be noted, that the explicit LBIST test of safety structures is not implemented in all derivatives of the AURIX™ TC4xx platform. Especially those devices, which are defined as ASIL-B, can achieve the required stuck-at test coverage of 60% within t_{LBIST} by using conventional ("key-off") LBIST execution, which involves all logical structures in the device. The information if a specific "key-on" LBIST scenario is implemented or not, can be found in the device specific Logic-Test (TRI) description.

be checked during the next “key-on” boot sequence together with the results from the safety LBIST (which might have been automatically started from firmware-side). In this conjunction it is recommended to always clear the “key-off” MISR results from Flash after a successful check to make sure, that they are not erroneously reused for the next LBIST execution.

The flowchart of the following figure shows how the different LBIST modes should be handled from firmware and application software side. The red numbers in the figure indicate which flow branches are used for the following LBIST use-cases:

1. Boot without LBIST execution or MISR-check
2. Boot without LBIST execution, followed by a check of the MISR-signature from a previously executed “key-off” LBIST-run (stored in Flash)
3. Automatic execution of safety-LBIST by firmware during “key-on”, followed by a check of the MISR-results by App-software (no MISR-signature stored in Flash from a previous “key-off” LBIST-run)
4. Automatic execution of safety-LBIST by firmware during “key-on”, followed by a check of the MISR-results by App-software (including the MISR-signature results stored in Flash from a previous “key-off” LBIST-run)
5. Manual execution of safety-LBIST by App-software during “key-on” or within application, followed by a check of the safety MISR-results
6. Manual execution of full LBIST by App-software during “key-off” or other LBIST-execution case, followed by a check of the full MISR-results
7. Manual execution of full LBIST by App-software during “key-off” LBIST-execution case, followed by storing the MISR-results in Flash. The check has to be done during next “key-on” sequence

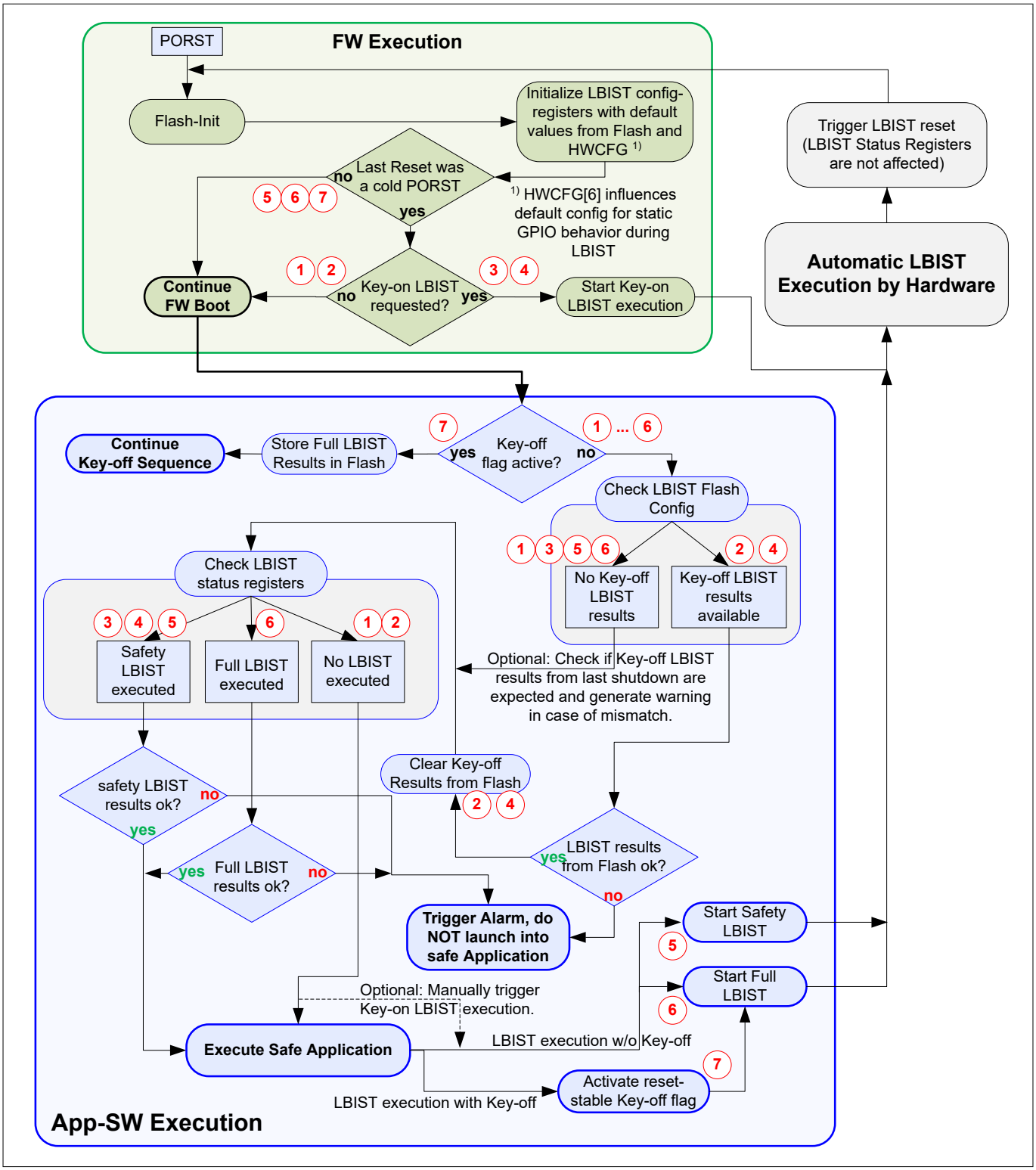


Figure 759 Self-test logic BIST firmware and software flow

Related information

LBIST configuration options on page 5687
Firmware on page 8773

33.3.5 Other Logic Test (TRI) configuration registers

The Logic Test (TRI) IP includes further registers which are not related to the LBIST functionality but are used for diverse debug features.

The following registers belong to this category:

- DXCCTR: This register determines, if the debug over CAN physical layer (DXCPL) and the application test mode (ATM) functions are available or disabled. The register's fields are automatically initialized through the boot firmware according to the installed configuration sector values in the non volatile memory.

33.4 Registers

33.4.1 Register overview - access mode glossary

Table 1336 Register overview - access mode glossary

Keyword	Description
CSE	Access protection using PROT register PROTCSE.
E	Access protection using PROT register PROTE.
SE	Access protection using PROT register PROTSE.
APU-P	Protection group consisting of registers ACCEN_WRA, ACCEN_WRB, ACCEN_RDA, ACCEN_RDB, ACCEN_VM, ACCEN_PRS.
P	Access protection using *APU-P* registers.
SV	Access only when supervisor mode is active on the interconnect.
BE	Always returns a Bus Error.
U	No access restrictions.
PROT	Access restrictions as defined in the PROT register access rules.
STP	Access only during start-up, before start-up protection is set (SMM_STCON.ST_PROT=1).

33.4.2 Registers overview - TRI (ascending offset address)

Table 1337 Registers overview - TRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CLC	Clock control register	000_H	P	P, SV, E	Application Reset	5695
ID	Module identification register	008_H	P	BE	PowerOn Reset	5696
PROTE	PROT register endinit	020_H	U	SV, PROT	Application Reset	5697
PROTSE	PROT register safe endinit	024_H	U	SV, PROT	Application Reset	5698
PROTCSE	PROT register cyber-secure endinit	028_H	U	SV, PROT	Application Reset	5700
ACCEN_WRA	Write access enable register A	040_H	U	SE, SV	Application Reset	5701
ACCEN_WRB	Write access enable register B	044_H	U	SE, SV	Application Reset	5702
ACCEN_RDA	Read access enable register A	048_H	U	SE, SV	Application Reset	5702
ACCEN_RDB	Read access enable register B	$04C_H$	U	SE, SV	Application Reset	5703
ACCEN_VM	VM access enable register	050_H	U	SE, SV	Application Reset	5703

(table continues...)

Table 1337 (continued) Registers overview - TRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ACCEN_PRS	PRS access enable register	054_H	U	SE, SV	Application Reset	5704
LBIST_CTRL_REQ	Logic BIST request register	060_H	U	SV, E	PowerOn Reset	5705
LBIST_CTRL_CFGA	Logic BIST configuration register A	064_H	U	SV, E	See 5706	5706
LBIST_CTRL_CFGB	Logic BIST configuration register B	068_H	U	SV, E	See 5709	5709
LBIST_CTRL_CFGC	Logic BIST configuration register C	$06C_H$	U	SV, E	See 5710	5710
LBIST_CTRL_CFGD	Logic BIST configuration register D	070_H	U	SV, E	See 5710	5710
LBIST_CTRL_CFGE	Logic BIST configuration register E	074_H	U	SV, E	See 5711	5711
LBIST_CTRL_CFGF	Logic BIST configuration register F	078_H	U	SV, E	See 5712	5712
LBIST_CTRL_DOM	Logic BIST domain selection	$07C_H$	U	SV, E	See 5714	5714
LBIST_STAT_TEST	Logic BIST status register	080_H	U	BE	See 5716	5716
LBIST_STAT_RESx_SIG	Logic BIST result x register	084_H+x*4	U	BE	See 5720	5720
DXCCTR	Debug over CAN enabling control	$0F8_H$	U	STP, CSE	See 5720	5720

33.4.3 Clock control register

The Clock Control Register CLC allows the programmer to adapt the functionality and power consumption of the module to the requirements of the application.

Register CLC controls the module clock signal and the reactivity to the sleep signal.

CLC Offset address: 000_H

Clock control register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												EDIS	0	DISS	DISR
r												rw	r	rh	rw

Field	Bits	Type	Description
DISR	0	rw	**Module disable eequest bit** Used for enable/disable control of the module. 0_B On request: enable the module clock 1_B Off request: stop the module clock
DISS	1	rh	**Module disable status bit** When CLC.DISS=1_B, registers CLC, ID, PROTE, PROTSE and ACCEN_* are still accessible. Accesses to other Logic-Test (TRI) registers are not executed and will result in a bus error. 0_B Module clock is enabled 1_B Off: module is not clocked
EDIS	3	rw	**Sleep mode enable control** Used to control the module's reaction to sleep mode. 0_B Sleep mode request is enabled and functional 1_B Module disregards the sleep mode control signal
0	2, 31:4	r	**Reserved** Read as 0; should be written with 0.

33.4.4 Module identification register

ID Offset address: 008_H

Module identification register PowerOn Reset value: $0109\ C001_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MOD_NUM															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MOD_TYPE								MOD_REV							
r								r							

Field	Bits	Type	Description
MOD_REV	7:0	r	**Module revision** Indicates the revision number of the implementation.
MOD_TYPE	15:8	r	**Module type** This internal marker is fixed to $C0_H$.
MOD_NUM	31:16	r	**Module number** Indicates the module identification number (example: $00C5_H$ = SARADC).

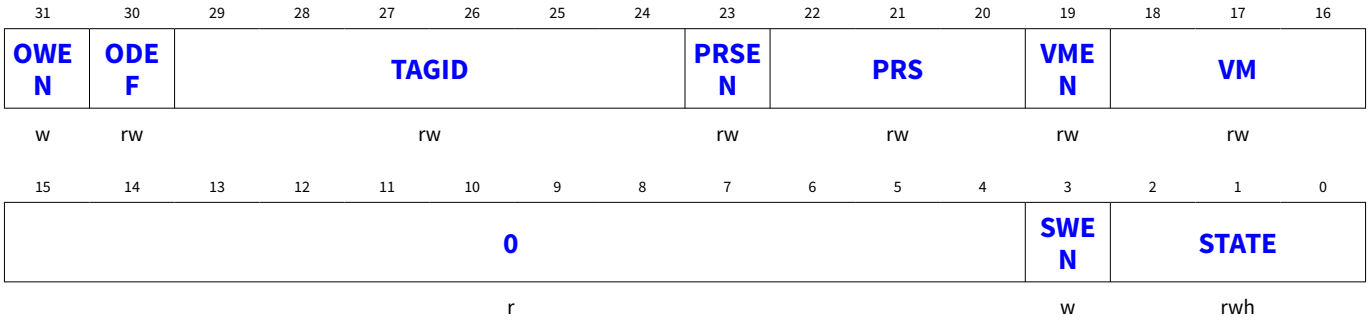
33.4.5 PROT register endinit

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner and in some PROT states, the secure master to update the PROT state as described in the PROT mechanism.

The PROTE register controls lock / unlock of the local Endinit (E) protected control registers.

PROTE Offset address: 020_H

PROT register endinit Application Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
VM	18:16	rw	**Virtual machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual machine definition Enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.

(table continues...)

(continued)

Field	Bits	Type	Description
PRSEN	23	rw	**Protection set definition enable for PROT owner** 0_B **Disable**: PRS is not part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; should be written with 0.

Table 1338 Access mode restrictions of PROTE sorted by descending priority

Mode name	Access mode		Description
write 1 to .**SWEN**	rwh	STATE	
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

33.4.6 PROT register safe endinit

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner and in some PROT states, the secure master to update the PROT state as described in the PROT mechanism.

The PROTSE register controls lock / unlock of the local Safe Endinit (SE) protected control registers.

PROTSE Offset address: 024_H

PROT register safe endinit Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OWEN	ODEF	TAGID						PRSEN	PRS			VMEN	VM		
w	rw	rw						rw	rw			rw	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												SWEN	STATE		
r												w	rwh		

Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
VM	18:16	rw	**Virtual machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual machine definition Enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.
PRSEN	23	rw	**Protection set definition enable for PROT owner** 0_B **Disable**: PRS is not part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; should be written with 0.

Table 1339 Access mode restrictions of PROTSE sorted by descending priority

Mode name	Access mode		Description
write 1 to .**SWEN**	rwh	STATE	
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

33.4.7 PROT register cyber-secure endinit

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner and in some PROT states, the secure master to update the PROT state as described in the PROT mechanism.

The PROTCSE register controls lock / unlock of the local Cyber Secure Endinit (CSE) protected control registers.

PROTCSE Offset address: 028_H

PROT register cyber-secure endinit Application Reset value: $4C00\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OWEN	**ODEF**	**TAGID**						**PRSEN**	**PRS**			**VMEN**	**VM**		
w	rw	rw						rw	rw			rw	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												**SWEN**	**STATE**		
r												w	rwh		

Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write

(table continues...)

(continued)

Field	Bits	Type	Description
VM	18:16	rw	**Virtual machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual machine definition Enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.
PRSEN	23	rw	**Protection set definition enable for PROT owner** 0_B **Disable**: PRS is not part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; should be written with 0.

Table 1340 Access mode restrictions of PROTCSE sorted by descending priority

Mode name	Access mode		Description
write 1 to .**SWEN**	rwh	STATE	
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

33.4.8 Write access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

ACCEN_WRA Offset address: 040_H

Write access enable register A Application Reset value: $1000\ 0003_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access

33.4.9 Write access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

ACCEN_WRB Offset address: 044_H

Write access enable register B Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access
0	31:8	r	**Reserved** Read as 0; should be written with 0.

33.4.10 Read access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

ACCEN_RDA Offset address: 048_H

Read access enable register A Application Reset value: $FFFF\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access

33.4.11 Read access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

ACCEN_RDB Offset address: $04C_H$

Read access enable register B Application Reset value: $0000\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access
0	31:8	r	**Reserved** Read as 0; should be written with 0.

33.4.12 VM access enable register

This register defines which virtual machine encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each VMx.

ACCEN_VM Offset address: 050_H

VM access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for VM ID q** This bit enables read access to the access protected region for transactions with the VM ID q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for VM ID q** This bit enables write access to the access protected region for transactions with the VM ID q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

33.4.13 PRS access enable register

This register defines which protection register set encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each PRSx.

ACCEN_PRS Offset address: 054_H

PRS access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for PRS q** This bit enables read access to the access protected region for transactions with the PRS q. 0_B Disabled for read access 1_B Enabled for read access

(table continues...)

(continued)

Field	Bits	Type	Description
WRq (q=00-07)	q+16	rw	**Write access enable for PRS q** This bit enables write access to the access protected region for transactions with the PRS q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

33.4.14 Logic BIST request register

LBIST_CTRL_REQ Offset address: 060_H

Logic BIST request register PowerOn Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
REQ_RED	ERRINJ	0													
rwh	rwh	r													

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FAILCIN			0											RESET	REQ
rw			r											w	rwh

Field	Bits	Type	Description
REQ	0	rwh	**LBIST request** If written high this bit requests the execution of an automatic scan-test procedure for each hierarchical scan domain which is selected via LBIST_CTRL_DOM.SEL0..7. Due to safety reasons the LBIST_CTRL_REQ.REQ_RED bit as well must be set to high value in order to generate the LBIST request. The request will only be approved if LBIST_STAT_TEST.DONE bit reflects a low value, meaning no LBIST-procedure was triggered since the last power-on-reset or LBIST-controller has been restarted through the LBIST_CTRL_REQ.RESET-bit. This bit will reset automatically after internal synchronisation of the LBIST request. ***Note**: LBIST execution time depends on the number of scan-loads as defined in the LBIST_CTRL_CFGA.PATTERNS field.*
RESET	1	w	**LBIST-reset** If written high this bit synchronously brings back the LBIST-controller to its initial Reset/Idle-state and also clears the stored MISR-signature to allow another execution from CPU-side. As a consequence the LBIST_STAT_TEST.DONE- and LBIST_STAT_RESx_SIG.SIGNATURE-bits will be set to low value. If read this bit always returns a low value.

(table continues...)

(continued)

Field	Bits	Type	Description
FAILCIN	15:13	rw	**Fail counter input** This bit-field has no direct hardware effect on LBIST-execution itself, but shall serve as memory function for the application software. In case of a LBIST failure software can use it to memorize the number of times LBIST was not executed successfully or MISR-results did not match to the expected value. Because any LBIST operation will terminate with an internal reset and reboot-phase during which all application registers settings are lost, application needs to be able to remember the number of LBIST fails afterwards. The value written to of this bit-filed is memorized during a LBIST-procedure and indicated through the LBIST_STAT_TEST.FAILCOUT-field after the internal reset and reboot-phase.
ERRINJ	30	rwh	**LBIST / test-mode alarm error injection** If written high, this bit triggers both the LBIST- and the test-mode-alarm. This is required to allow self-testing of all LBIST-(and test-mode-)related safety mechanisms in the TCU. The bit will be reset automatically once the LBIST and test-mode alarm indicator signals from TCU are asserted. From these indicator signals TRI will also generate corresponding alarm trigger signals for SMU.
REQ_RED	31	rwh	**LBIST request redundancy** This bit represents the safety double of LBIST_CTRL_REQ.REQ. In order to generate a new LBIST request both, REQ_RED and REQ bits must be set to high due to safety reasons. The request will only be approved if LBIST_STAT_TEST.DONE bit reflects a low value. This bit will reset automatically after internal synchronisation of the LBIST request.
0	12:2, 29:16	r	**Reserved** Read as 0; should be written with 0.

Table 1341 Access mode restrictions of LBIST_CTRL_REQ sorted by descending priority

Mode name	Access mode		Description
write 1 to .**REQ_RED**	rwh	REQ	Set REQ_RED during write access
(default)	rh	REQ	

33.4.15 Logic BIST configuration register A

LBIST_CTRL_CFGA Offset address: 064_H

Logic BIST configuration register A Reset values see: Table 1342

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FREQU				0	SPLITSH		KEY OFF	RSV D23	ENP ARA	SAFE SEL	TFREQU		RSV D18	PATTERNS	
rw				r	rw		rw	rw	rw	rw	rw		rw	rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PATTERNS															
rw															

Field	Bits	Type	Description
PATTERNS	17:0	rw	**LBIST pattern number** This field defines the number of scan-patterns (scan-loads), which will be executed during the LBIST-procedure for each hierarchical block selected via LBIST_CTRL_DOM.SEL0..7. Please note that the value programmed to this field determines the number scan-capture phases not the number of scan-chain load-unload phases. This means a value of 0x00001 will result in two scan-chain loads with on capture in-between, a value of 0x00002 will result in 3 scan-chain loads with 2 captures and so on. Consequently a value of 0x00000 is not valid, because no capture would be executed in this case.
RSVD18	18	rw	**Reserved** Must not be changed from its reset value.
TFREQU	20:19	rw	**LBIST transaction frequency selection** Through this bit-field a pre-scaler factor between 2..4 is selectable for LBIST transaction clock (derived from EVR-oscillator). It allows to determine the operation speed of the on-chip iJTAG interface through which the LBIST configuration is transferred from the SPB-registers to the hierarchical scan domain. Value of this bit-field is mirrored inside of LBIST-controller and becomes effective with the start of a new LBIST-procedure (via LBIST_CTRL_REQ.REQ/REQ_RED). Please note that a divider factor of 1 is not supported, so this field must be set to a value between 1-3 to enable LBIST initialization. 00_B iJTAG clock disabled (no LBIST initialization possible). 01_B iJTAG interface runs at 50MHz. 10_B iJTAG interface runs at 33MHz. 11_B iJTAG interface runs at 25MHz.
SAFESEL	21	rw	**Safety LBIST test indicator** The value of this bit determines if LBIST-operation is executed only on the safety-relevant (safety-LBIST) or on the complete digital logic (full-scan) of each hierarchical scan domain which is selected via LBIST_CTRL_DOM.SEL0..7. For boot operations with limited execution time this option allows to exclude all non-safety-relevant parts of the digital logic from the scan-path. This allows a lower power consumption and hence enables faster execution of the LBIST procedure. For application scenarios with relaxed time conditions a complete LBIST test is executed on all parts of the hierarchical domain. 0_B For each hierarchical domain selected via LBIST_CTRL_DOM.SEL0..7 LBIST is performed with full-scan operation. This condition is applicable for key-off or all in-system-test application scenarios where sufficient LBIST execution time is available. 1_B For each hierarchical domain selected via LBIST_CTRL_DOM.SEL0..7 only those scan-chains are operated during LBIST execution which contain safety-relevant logic. This condition typically refers to key-on application scenarios where LBIST execution time is limited.

(table continues...)

(continued)

Field	Bits	Type	Description
ENPARA	22	rw	**Parallel LBIST execution indicator** This bit allows to select if LBIST test is executed for all hierarchical hierarchical scan domains sequentially or if LBIST is operated for all hierarchies in parallel. The second option can safe significant operation time for the total key-on LBIST scenario but may only be used if permissible from power consumption perspective. 0_B LBIST operation is executed sequentially for each hierarchical domain selected via LBIST_CTRL_DOM.SEL0..7. 1_B LBIST operation is executed concurrently for all hierarchical domain at a time which are selected via LBIST_CTRL_DOM.SEL0..7. This setting can only be used if permissable from power consumption perspective.
RSVD23	23	rw	**Reserved** Must not be changed from its reset value.
KEYOFF	24	rw	**Key-off indicator** This bit has no direct hardware effect on LBIST-execution itself, but shall serve as memory function for the application software. In case of a key-off request the application may trigger a final LBIST procedure before shutting down the system. Because this LBIST procedure will terminate with an additional reset and reboot-phase during which all application registers settings are lost, application needs to be able to distinguish the key-on case from the key-off case. The value of this bit is memorized during a LBIST-procedure and indicated through the LBIST_STAT_TEST.KEYIND-bit after the internal reset and reboot-phase. 0_B After the next LBIST-execution the LBIST_STAT_TEST.KEYIND-bit will indicate a low value, meaning the previous LBIST operation was triggered during a key-on sequence or a normal run-time event. 1_B After the next LBIST-execution the LBIST_STAT_TEST.KEYIND-bit will indicate a high value, meaning the previous LBIST operation was triggered on base of a pending key-off request.
SPLITSH	26:25	rw	**LBIST split-shift selection** The value of this bit will allow to run LBIST with partitioned scan-shift operation in order to reduce the power consumption. 00_B Concurrent scan-shift is selected. 01_B Concurrent scan-shift is selected. 10_B Partitioned scan-shift is selected (four scan partitions). 11_B Partitioned scan-shift is selected (two scan partitions).
FREQU	31:28	rw	**LBIST frequency selection** Through this bit-field a pre-scaler factor between 1..16 is selectable for LBIST operation clock (derived from EVR-oscillator). It allows to determine the LBIST scan-shift frequency. Value of these bit-field will be transported to the local LBIST-controllers within each domain and becomes effective with the start of a new LBIST-procedure (via LBIST_CTRL_REQ.REQ/REQ_RED).
0	27	r	**Reserved** Read as 0; should be written with 0.

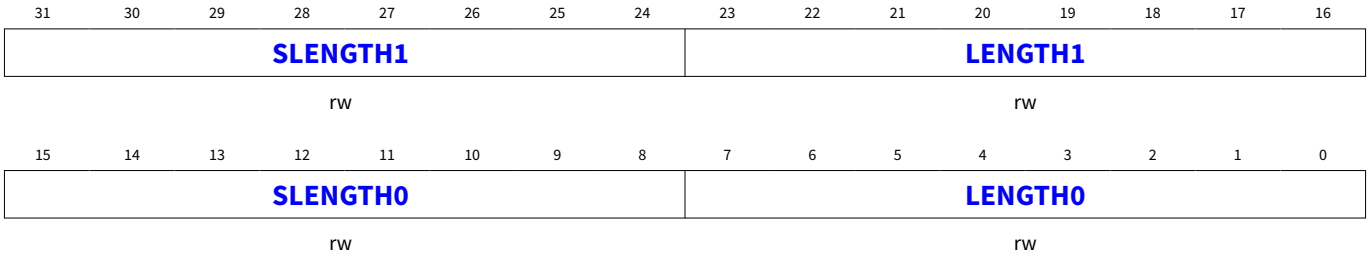
Table 1342 Reset values of LBIST_CTRL_CFGA

Reset type	Reset value	Note
PowerOn Reset	0000 0000_H	
After Boot-FW Value	---- X--- X--- -X-- ---- ---- ---- ----$_B$	The correct CFS value of the LBIST_CTRL_CFGA bit-fields has to be looked up in the device-specific section.

33.4.16 Logic BIST configuration register B

LBIST_CTRL_CFGB Offset address: 068_H

Logic BIST configuration register B Reset values see: Table 1343



Field	Bits	Type	Description
LENGTH0	7:0	rw	**LBIST maximum scan-chain length for scan domain #0** If LBIST_CTRL_CFGA.SAFESEL='0' this field defines the number of shift-cycles for each LBIST scan-load (applied to hierarchical scan domain #0). It will be automatically loaded with the product-specific value, stored in Flash config-sector during startup-software execution.
SLENGTH0	15:8	rw	**LBIST maximum safety scan-chain length for scan domain #0** If LBIST_CTRL_CFGA.SAFESEL='1' this field defines the number of shift-cycles for each LBIST scan-load (applied to hierarchical scan domain #0). It will be automatically loaded with the product-specific value, stored in Flash config-sector during startup-software execution.
LENGTH1	23:16	rw	**LBIST maximum scan-chain length for scan domain #1** If LBIST_CTRL_CFGA.SAFESEL='0' this field defines the number of shift-cycles for each LBIST scan-load (applied to hierarchical scan domain #1). It will be automatically loaded with the product-specific value, stored in Flash config-sector during startup-software execution.
SLENGTH1	31:24	rw	**LBIST maximum safety scan-chain length for scan domain #1** If LBIST_CTRL_CFGA.SAFESEL='1' this field defines the number of shift-cycles for each LBIST scan-load (applied to hierarchical scan domain #1). It will be automatically loaded with the product-specific value, stored in Flash config-sector during startup-software execution.

Table 1343 Reset values of LBIST_CTRL_CFGB

Reset type	Reset value	Note
PowerOn Reset	0000 0000_H	
After Boot-FW Value	---- ----$_H$	The correct CFS value of the LBIST_CTRL_CFGB bit-fields has to be looked up in the device-specific section.

33.4.17 Logic BIST configuration register C

LBIST_CTRL_CFGC Offset address: $06C_H$

Logic BIST configuration register C Reset values see: Table 1344

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SLENGTH3								LENGTH3							
rw								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SLENGTH2								LENGTH2							
rw								rw							

Field	Bits	Type	Description
LENGTH2	7:0	rw	**LBIST maximum scan-chain length for scan domain #2** If LBIST_CTRL_CFGA.SAFESEL='0' this field defines the number of shift-cycles for each LBIST scan-load (applied to hierarchical scan domain #2). It will be automatically loaded with the product-specific value, stored in Flash config-sector during startup-software execution.
SLENGTH2	15:8	rw	**LBIST maximum safety scan-chain length for scan domain #2** If LBIST_CTRL_CFGA.SAFESEL='1' this field defines the number of shift-cycles for each LBIST scan-load (applied to hierarchical scan domain #2). It will be automatically loaded with the product-specific value, stored in Flash config-sector during startup-software execution.
LENGTH3	23:16	rw	**LBIST maximum scan-chain length for scan domain #3** If LBIST_CTRL_CFGA.SAFESEL='0' this field defines the number of shift-cycles for each LBIST scan-load (applied to hierarchical scan domain #3). It will be automatically loaded with the product-specific value, stored in Flash config-sector during startup-software execution.
SLENGTH3	31:24	rw	**LBIST maximum safety scan-chain length for scan domain #3** If LBIST_CTRL_CFGA.SAFESEL='1' this field defines the number of shift-cycles for each LBIST scan-load (applied to hierarchical scan domain #3). It will be automatically loaded with the product-specific value, stored in Flash config-sector during startup-software execution.

Table 1344 Reset values of LBIST_CTRL_CFGC

Reset type	Reset value	Note
PowerOn Reset	$0000\ 0000_H$	
After Boot-FW Value	$----\ ----_H$	The correct CFS value of the LBIST_CTRL_CFGC bit-fields has to be looked up in the device-specific section.

33.4.18 Logic BIST configuration register D

LBIST_CTRL_CFGD Offset address: 070_H

Logic BIST configuration register D Reset values see: Table 1345

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SLENGTH5								LENGTH5							
rw								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SLENGTH4								LENGTH4							
rw								rw							

Field	Bits	Type	Description
LENGTH4	7:0	rw	**LBIST maximum scan-chain length for scan domain #4** If LBIST_CTRL_CFGA.SAFESEL='0' this field defines the number of shift-cycles for each LBIST scan-load (applied to hierarchical scan domain #4). It will be automatically loaded with the product-specific value, stored in Flash config-sector during startup-software execution.
SLENGTH4	15:8	rw	**LBIST maximum safety scan-chain length for scan domain #4** If LBIST_CTRL_CFGA.SAFESEL='1' this field defines the number of shift-cycles for each LBIST scan-load (applied to hierarchical scan domain #4). It will be automatically loaded with the product-specific value, stored in Flash config-sector during startup-software execution.
LENGTH5	23:16	rw	**LBIST maximum scan-chain length for scan domain #5** If LBIST_CTRL_CFGA.SAFESEL='0' this field defines the number of shift-cycles for each LBIST scan-load (applied to hierarchical scan domain #5). It will be automatically loaded with the product-specific value, stored in Flash config-sector during startup-software execution.
SLENGTH5	31:24	rw	**LBIST maximum safety scan-chain length for scan domain #5** If LBIST_CTRL_CFGA.SAFESEL='1' this field defines the number of shift-cycles for each LBIST scan-load (applied to hierarchical scan domain #5). It will be automatically loaded with the product-specific value, stored in Flash config-sector during startup-software execution.

Table 1345 Reset values of LBIST_CTRL_CFGD

Reset type	Reset value	Note
PowerOn Reset	0000 0000$_H$	
After Boot-FW Value	---- ----$_H$	The correct CFS value of the LBIST_CTRL_CFGD bit-fields has to be looked up in the device-specific section.

33.4.19 Logic BIST configuration register E

LBIST_CTRL_CFGE Offset address: 074$_H$

Logic BIST configuration register E Reset values see: Table 1346

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SLENGTH7								LENGTH7							
rw								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SLENGTH6								LENGTH6							
rw								rw							

Field	Bits	Type	Description
LENGTH6	7:0	rw	**LBIST maximum scan-chain length for scan domain #6** If LBIST_CTRL_CFGA.SAFESEL='0' this field defines the number of shift-cycles for each LBIST scan-load (applied to hierarchical scan domain #6). It will be automatically loaded with the product-specific value, stored in Flash config-sector during startup-software execution.
SLENGTH6	15:8	rw	**LBIST maximum safety scan-chain length for scan domain #6** If LBIST_CTRL_CFGA.SAFESEL='1' this field defines the number of shift-cycles for each LBIST scan-load (applied to hierarchical scan domain #6). It will be automatically loaded with the product-specific value, stored in Flash config-sector during startup-software execution.
LENGTH7	23:16	rw	**LBIST maximum scan-chain length for scan domain #7** If LBIST_CTRL_CFGA.SAFESEL='0' this field defines the number of shift-cycles for each LBIST scan-load (applied to hierarchical scan domain #7). It will be automatically loaded with the product-specific value, stored in Flash config-sector during startup-software execution.
SLENGTH7	31:24	rw	**LBIST maximum safety scan-chain length for scan domain #7** If LBIST_CTRL_CFGA.SAFESEL='1' this field defines the number of shift-cycles for each LBIST scan-load (applied to hierarchical scan domain #7). It will be automatically loaded with the product-specific value, stored in Flash config-sector during startup-software execution.

Table 1346 Reset values of LBIST_CTRL_CFGE

Reset type	Reset value	Note
PowerOn Reset	0000 0000_H	
After Boot-FW Value	---- $----_H$	The correct CFS value of the LBIST_CTRL_CFGE bit-fields has to be looked up in the device-specific section.

33.4.20 Logic BIST configuration register F

LBIST_CTRL_CFGF Offset address: 078_H

Logic BIST configuration register F Reset values see: Table 1347

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD12				0							OBSSCDIS	TPCTRDIS	TPOBSDIS	LBCGTRAN	
rw				r							rw	rw	rw	rw	

Field	Bits	Type	Description
LBCGTRAN	1:0	rw	**(LBIST) clock gate transparency control** For LBIST-operation it is possible to force certain groups of power-related clock-gate stages transparent through a specific 'pwrcmp_clkgate_en_o[6:0]' vector during the scan capture phase. This enables a quick coverage gain but also increases the overall power dissipation during LBIST-operation. Therefore this function shall be handled with care. 00_B All off: All 'pwrcmp_clkgate_en_o[6:0]' signals are constrained to low value during capture-phase. 01_B One-hot encoding: Only 1 'pwrcmp_clkgate_en_o[6:0]' signal may be set to high during capture-phase. 10_B Two-hot encoding: Only 2 'pwrcmp_clkgate_en_o[6:0]' signals may be set to high during capture-phase. 11_B Three-hot encoding: Only 3 'pwrcmp_clkgate_en_o[6:0]' signals may be set to high during capture-phase.
TPOBSDIS	2	rw	**Test observe point disabling** Setting this bit to low enables all scan observe test points for test coverage optimization. Setting this bit to high value will prevent the test observe FFs from capturing any scan-data. In this case they will only be clocked during the scan-shift phase.
TPCTRDIS	3	rw	**Test control point disabling** Setting this bit to low enables all scan control test points for test coverage optimization. Setting this bit to high value will prevent the test control FFs from influencing any structural paths within the scan design. In this case they will only be clocked during the scan-shift phase.
OBSSCDIS	4	rw	**Scan shift observation disabling** Setting this bit to low enables the observation scan feature in LBIST-operation under the precondition that LBIST_CTRL_CFGF.TPOBSDIS='0'. Setting this bit to high value will prevent the scan observation FFs from capturing any data during scan shift and capture process. Active scan observe FFs reduce the time to achieve the target stuck-at test-coverage compared to standard LBIST operation, where test observation points are active during capture phase only.
RSVD12	15:12	rw	**Reserved** Must not be changed from its reset value.
0	11:5, 31:16	r	**Reserved** Read as 0; should be written with 0.

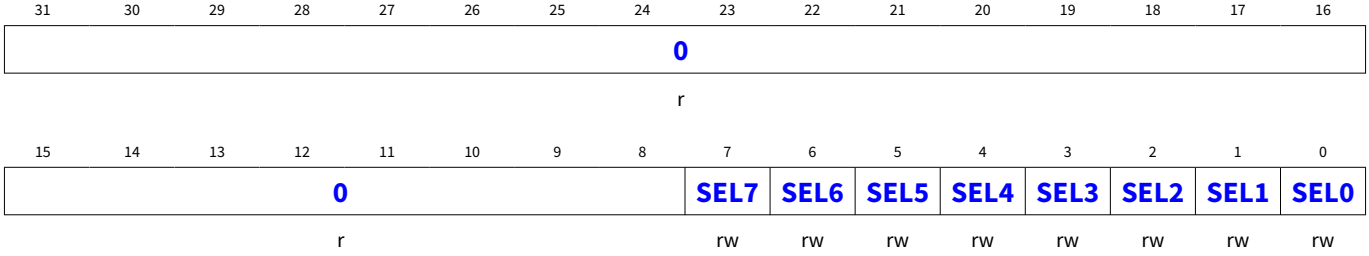
Table 1347 Reset values of LBIST_CTRL_CFGF

Reset type	Reset value	Note
PowerOn Reset	$0000\ 0000_H$	
After Boot-FW Value	$0000\ 0000\ 0000\ 0000$ $UUXX\ XXXX\ XXX–\ ----_B$	The correct CFS value of the LBIST_CTRL_CFGF bit-fields has to be looked up in the device-specific section.

33.4.21 Logic BIST domain selection

LBIST_CTRL_DOM Offset address: $07C_H$

Logic BIST domain selection Reset values see: Table 1348



Field	Bits	Type	Description
SEL0	0	rw	**LBIST hierarchical scan domain 0 selection** This bit-field defines for which hierarchical scan domains LBIST operation will be performed. LBIST operations will usually be executed sequential for each selected domain. If permissible from power side it is possible to select the concurrent LBIST execution for all selected hierarchical domains via LBIST_CTRL_CFGA.ENPARA='1'. 0_B No LBIST will be executed for hierarchical scan domain #0 if a new LBIST procedure is triggered via LBIST_CTRL_REQ.LBISTREQ-bit. 1_B LBIST operation will be executed for hierarchical scan domain #0 if LBIST procedure is triggered via LBIST_CTRL_REQ.LBISTREQ-bit.
SEL1	1	rw	**LBIST hierarchical scan domain 1 selection** This bit-field defines for which hierarchical scan domains LBIST operation will be performed. LBIST operations will usually be executed sequential for each selected domain. If permissible from power side it is possible to select the concurrent LBIST execution for all selected hierarchical domains via LBIST_CTRL_CFGA.ENPARA='1'. 0_B No LBIST will be executed for hierarchical scan domain #1 if a new LBIST procedure is triggered via LBIST_CTRL_REQ.LBISTREQ-bit. 1_B LBIST operation will be executed for hierarchical scan domain #1 if LBIST procedure is triggered via LBIST_CTRL_REQ.LBISTREQ-bit.
SEL2	2	rw	**LBIST hierarchical scan domain 2 selection** This bit-field defines for which hierarchical scan domains LBIST operation will be performed. LBIST operations will usually be executed sequential for each selected domain. If permissible from power side it is possible to select the concurrent LBIST execution for all selected hierarchical domains via LBIST_CTRL_CFGA.ENPARA='1'. 0_B No LBIST will be executed for hierarchical scan domain #2 if a new LBIST procedure is triggered via LBIST_CTRL_REQ.LBISTREQ-bit. 1_B LBIST operation will be executed for hierarchical scan domain #2 if LBIST procedure is triggered via LBIST_CTRL_REQ.LBISTREQ-bit.

(table continues...)

(continued)

Field	Bits	Type	Description
SEL3	3	rw	**LBIST hierarchical scan domain 3 selection** This bit-field defines for which hierarchical scan domains LBIST operation will be performed. LBIST operations will usually be executed sequential for each selected domain. If permissible from power side it is possible to select the concurrent LBIST execution for all selected hierarchical domains via LBIST_CTRL_CFGA.ENPARA='1'. 0_B No LBIST will be executed for hierarchical scan domain #3 if a new LBIST procedure is triggered via LBIST_CTRL_REQ.LBISTREQ-bit. 1_B LBIST operation will be executed for hierarchical scan domain #3 if LBIST procedure is triggered via LBIST_CTRL_REQ.LBISTREQ-bit.
SEL4	4	rw	**LBIST hierarchical scan domain 4 selection** This bit-field defines for which hierarchical scan domains LBIST operation will be performed. LBIST operations will usually be executed sequential for each selected domain. If permissible from power side it is possible to select the concurrent LBIST execution for all selected hierarchical domains via LBIST_CTRL_CFGA.ENPARA='1'. 0_B No LBIST will be executed for hierarchical scan domain #4 if a new LBIST procedure is triggered via LBIST_CTRL_REQ.LBISTREQ-bit. 1_B LBIST operation will be executed for hierarchical scan domain #4 if LBIST procedure is triggered via LBIST_CTRL_REQ.LBISTREQ-bit.
SEL5	5	rw	**LBIST hierarchical scan domain 5 selection** This bit-field defines for which hierarchical scan domains LBIST operation will be performed. LBIST operations will usually be executed sequential for each selected domain. If permissible from power side it is possible to select the concurrent LBIST execution for all selected hierarchical domains via LBIST_CTRL_CFGA.ENPARA='1'. 0_B No LBIST will be executed for hierarchical scan domain #5 if a new LBIST procedure is triggered via LBIST_CTRL_REQ.LBISTREQ-bit. 1_B LBIST operation will be executed for hierarchical scan domain #5 if LBIST procedure is triggered via LBIST_CTRL_REQ.LBISTREQ-bit.
SEL6	6	rw	**LBIST hierarchical scan domain 6 selection** This bit-field defines for which hierarchical scan domains LBIST operation will be performed. LBIST operations will usually be executed sequential for each selected domain. If permissible from power side it is possible to select the concurrent LBIST execution for all selected hierarchical domains via LBIST_CTRL_CFGA.ENPARA='1'. 0_B No LBIST will be executed for hierarchical scan domain #6 if a new LBIST procedure is triggered via LBIST_CTRL_REQ.LBISTREQ-bit. 1_B LBIST operation will be executed for hierarchical scan domain #6 if LBIST procedure is triggered via LBIST_CTRL_REQ.LBISTREQ-bit.

(table continues...)

(continued)

Field	Bits	Type	Description
SEL7	7	rw	**LBIST hierarchical scan domain 7 selection** This bit-field defines for which hierarchical scan domains LBIST operation will be performed. LBIST operations will usually be executed sequential for each selected domain. If permissible from power side it is possible to select the concurrent LBIST execution for all selected hierarchical domains via LBIST_CTRL_CFGA.ENPARA='1'. 0_B No LBIST will be executed for hierarchical scan domain #7 if a new LBIST procedure is triggered via LBIST_CTRL_REQ.LBISTREQ-bit. 1_B LBIST operation will be executed for hierarchical scan domain #7 if LBIST procedure is triggered via LBIST_CTRL_REQ.LBISTREQ-bit.
0	31:8	r	**Reserved** Read as 0; should be written with 0.

Table 1348 Reset values of LBIST_CTRL_DOM

Reset type	Reset value	Note
PowerOn Reset	0000 0000$_H$	
After Boot-FW Value	0000 00--$_H$	The correct CFS value of the LBIST_CTRL_DOM bit-fields has to be looked up in the device-specific section.

33.4.22 Logic BIST status register

LBIST_STAT_TEST Offset address: 080$_H$

Logic BIST status register Reset values see: Table 1349

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								DOMDONE7	DOMDONE6	DOMDONE5	DOMDONE4	DOMDONE3	DOMDONE2	DOMDONE1	DOMDONE0
r								rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FAILCOUT			0									KEYIND	RSVD2	SAFEIND	DONE
rh			r									rh	rh	rh	rh

Field	Bits	Type	Description
DONE	0	rh	**LBIST execution indicator** This bit indicates the actual LBIST-controller execution status: 0_B No LBIST executed since last power-on-reset or LBIST-controller has been restarted (via LBIST_CTRL_REQ.RESET function) or LBIST was interrupted by external PORST during runtime (LBIST completion status is indicated via LBIST_STAT_TEST.DOMDONE0..7). Values in LBIST_STAT_RESx_SIG.SIGNATURE-fields are all set to low value. 1_B A LBIST-procedure successfully finished since last power-on-reset for all hierarchical scan domains which are selected via LBIST_CTRL_DOM.SEL0..7. Values in LBIST_STAT_RESx_SIG.SIGNATURE-fields reflect the resulting MISR-signature for each hierarchical scan domain.
SAFEIND	1	rh	**Safety LBIST memory bit** The value of this bit indicates if the previous Logic-BIST operation was executed under safety-LBIST test or a full-scan test conditions (please refer to LBIST_CTRL_CFGA.SAFESEL definition). This bit is only valid in LBIST_STAT_TEST.DONE='1' case. In DONE='0' case it always reflects a low value. 0_B The previous Logic-BIST operation was executed under full-scan test conditions. This means all design parts of the selected hierarchical domains are involved. 1_B The previous Logic-BIST operation was executed under safety-LBIST test conditions. This mean only safety-relevant design parts of the selected hierarchical domains are involved.
RSVD2	2	rh	**Reserved** Internally used function. Read value can be ignored.
KEYIND	3	rh	**Key-off LBIST memory bit** The value of this bit indicates if the previous LBIST-operation was triggered on base of a pending key-off request (please refer to LBIST_CTRL_CFGA.KEYOFF definition). This bit is only valid in LBIST_STAT_TEST.DONE='1' case. In DONE='0' case it always reflects a low value. 0_B The previous LBIST test was triggered during a key-on sequence or a normal run-time event. 1_B The previous LBIST test was triggered on base of a pending key-off request.
FAILCOUT	15:13	rh	**Fail counter output** This bit-field indicates how many times a previously triggered LBIST-operation failed (please refer to the LBIST_CTRL_REQ.FAILCIN-field definition). This field is only valid in LBIST_STAT_TEST.DONE='1' case. In DONE='0' case it always reflects a "000" value.

(table continues...)

(continued)

Field	Bits	Type	Description
DOMDONE0	16	rh	**LBIST hierarchical scan domain 0 execution status indicator** This bit indicates the actual LBIST-controller execution status for each hierarchical scan domain. If the scan domain is not implemented this bit always indicates a high value. 0_B No LBIST executed for hierarchical scan domain #0 since last power-on-reset or LBIST-controller has been restarted. Values in LBIST_STAT_RESx_SIG.SIGNATURE-field are all set to low value. 1_B An LBIST-procedure successfully finished for hierarchical scan domain #0 since last power-on-reset. Values in LBIST_STAT_RESx_SIG.SIGNATURE-field reflect the resulting MISR-signature.
DOMDONE1	17	rh	**LBIST hierarchical scan domain 1 execution status indicator** This bit indicates the actual LBIST-controller execution status for each hierarchical scan domain. If the scan domain is not implemented this bit always indicates a high value. 0_B No LBIST executed for hierarchical scan domain #1 since last power-on-reset or LBIST-controller has been restarted. Values in LBIST_STAT_RESx_SIG.SIGNATURE-field are all set to low value. 1_B An LBIST-procedure successfully finished for hierarchical scan domain #1 since last power-on-reset. Values in LBIST_STAT_RESx_SIG.SIGNATURE-field reflect the resulting MISR-signature.
DOMDONE2	18	rh	**LBIST hierarchical scan domain 2 execution status indicator** This bit indicates the actual LBIST-controller execution status for each hierarchical scan domain. If the scan domain is not implemented this bit always indicates a high value. 0_B No LBIST executed for hierarchical scan domain #2 since last power-on-reset or LBIST-controller has been restarted. Values in LBIST_STAT_RESx_SIG.SIGNATURE-field are all set to low value. 1_B An LBIST-procedure successfully finished for hierarchical scan domain #2 since last power-on-reset. Values in LBIST_STAT_RESx_SIG.SIGNATURE-field reflect the resulting MISR-signature.
DOMDONE3	19	rh	**LBIST hierarchical scan domain 3 execution status indicator** This bit indicates the actual LBIST-controller execution status for each hierarchical scan domain. If the scan domain is not implemented this bit always indicates a high value. 0_B No LBIST executed for hierarchical scan domain #3 since last power-on-reset or LBIST-controller has been restarted. Values in LBIST_STAT_RESx_SIG.SIGNATURE-field are all set to low value. 1_B An LBIST-procedure successfully finished for hierarchical scan domain #3 since last power-on-reset. Values in LBIST_STAT_RESx_SIG.SIGNATURE-field reflect the resulting MISR-signature.

(table continues...)

(continued)

Field	Bits	Type	Description
DOMDONE4	20	rh	**LBIST hierarchical scan domain 4 execution status indicator** This bit indicates the actual LBIST-controller execution status for each hierarchical scan domain. If the scan domain is not implemented this bit always indicates a high value. 0_B No LBIST executed for hierarchical scan domain #4 since last power-on-reset or LBIST-controller has been restarted. Values in LBIST_STAT_RESx_SIG.SIGNATURE-field are all set to low value. 1_B An LBIST-procedure successfully finished for hierarchical scan domain #4 since last power-on-reset. Values in LBIST_STAT_RESx_SIG.SIGNATURE-field reflect the resulting MISR-signature.
DOMDONE5	21	rh	**LBIST hierarchical scan domain 5 execution status indicator** This bit indicates the actual LBIST-controller execution status for each hierarchical scan domain. If the scan domain is not implemented this bit always indicates a high value. 0_B No LBIST executed for hierarchical scan domain #5 since last power-on-reset or LBIST-controller has been restarted. Values in LBIST_STAT_RESx_SIG.SIGNATURE-field are all set to low value. 1_B An LBIST-procedure successfully finished for hierarchical scan domain #5 since last power-on-reset. Values in LBIST_STAT_RESx_SIG.SIGNATURE-field reflect the resulting MISR-signature.
DOMDONE6	22	rh	**LBIST hierarchical scan domain 6 execution status indicator** This bit indicates the actual LBIST-controller execution status for each hierarchical scan domain. If the scan domain is not implemented this bit always indicates a high value. 0_B No LBIST executed for hierarchical scan domain #6 since last power-on-reset or LBIST-controller has been restarted. Values in LBIST_STAT_RESx_SIG.SIGNATURE-field are all set to low value. 1_B An LBIST-procedure successfully finished for hierarchical scan domain #6 since last power-on-reset. Values in LBIST_STAT_RESx_SIG.SIGNATURE-field reflect the resulting MISR-signature.
DOMDONE7	23	rh	**LBIST hierarchical scan domain 7 execution status indicator** This bit indicates the actual LBIST-controller execution status for each hierarchical scan domain. If the scan domain is not implemented this bit always indicates a high value. 0_B No LBIST executed for hierarchical scan domain #7 since last power-on-reset or LBIST-controller has been restarted. Values in LBIST_STAT_RESx_SIG.SIGNATURE-field are all set to low value. 1_B An LBIST-procedure successfully finished for hierarchical scan domain #7 since last power-on-reset. Values in LBIST_STAT_RESx_SIG.SIGNATURE-field reflect the resulting MISR-signature.
0	12:4, 31:24	r	**Reserved** Read as 0; should be written with 0.

Table 1349 Reset values of LBIST_STAT_TEST

Reset type	Reset value	Note
PowerOn Reset	0000 0000_H	
After Boot-FW Value	0000 0000 ---- ---- ---0 0000 0000 -X--$_B$	After reset the bits in this register will be updated automatically with the stored values in the central LBIST supervisor block.

33.4.23 Logic BIST result x register

LBIST_STAT_RESx_SIG (x=0-7) Offset address: 084_H+x*4

Logic BIST result x register Reset values see: Table 1350

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SIGNATURE															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SIGNATURE															
rh															

Field	Bits	Type	Description
SIGNATURE	31:0	rh	**LBIST domain x signature** This field reflects the MISR signature from the last LBIST execution of hierarchical scan domain #x. It is mirrored from LBIST-controller inside TRI and only valid if LBIST_STAT_TEST.DONE is read with a high value. In case of a restart of the LBIST-controller (via LBIST_CTRL_REQ.RESET function), the signature value will be synchronously reset to all-0. Please check the device specific section for a description on the SIGNATURE value, depending on the LBIST run configuration.

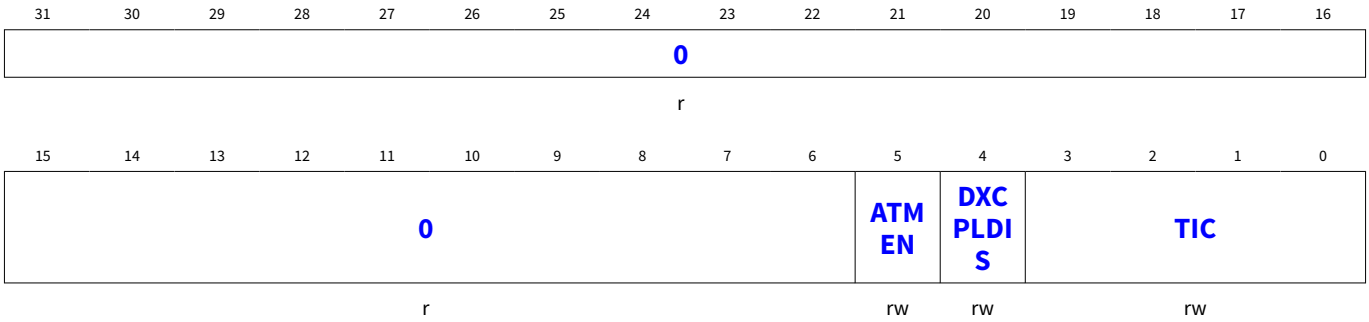
Table 1350 Reset values of LBIST_STAT_RESx_SIG (x=0-7)

Reset type	Reset value	Note
PowerOn Reset	0000 0000_H	
After Boot-FW Value	---- ----$_H$	After reset the SIGNATURE bits in this register will be updated automatically with the stored values in the central LBIST supervisor block.

33.4.24 Debug over CAN enabling control

DXCCTR Offset address: $0F8_H$

Debug over CAN enabling control Reset values see: Table 1351



Field	Bits	Type	Description
TIC	3:0	rw	**Tool interface control** DAP over CAN Physical Layer (DXCPL). 0_H DXCPL will be disabled if not already activated by the tool after power-on-reset. 1_H DXCPL will stay enabled/activated for standard CAN pins. 2_H DXCPL is enabled for alternative CAN pins. **others,** Reserved, do not use.
DXCPLDIS	4	rw	**Debug over CAN disabling control** This bit determines if Debug over CAN (DXCPL) operation mode is enabled or not. 0_B DXCPL operation mode can be enabled. 1_B DXCPL operation mode cannot be enabled (if it has already been entered it will be disabled again).
ATMEN	5	rw	**Application test mode enabling control** This bit determines if after enabling of Debug over CAN (DXCPL) operation mode the Application Test Mode (ATM) can be entered or not. 0_B ATM cannot be enabled after DXCPL entry. 1_B ATM can be enabled after DXCPL entry.
0	31:6	r	**Reserved** Read as 0; should be written with 0.

Table 1351 Reset values of DXCCTR

Reset type	Reset value	Note
PowerOn Reset	0000 0001_H	
After Boot-FW Value	0000 0000 0000 0000 0000 0000 00–– $----_B$	From CFS or UCB sector value

33.5 Debug information

This section is not applicable for the module being described.

33.6 References

This section is not applicable for the module being described.

33.7 TRI revision history

Initial release of the chapter.

33.8 TC4Dx TRI information

33.8.1 TC4Dx TRI configuration

It shall be noted, that the term '_DFT_', which is used in the TC4Dx TRI connectivity, means a synonym for '_TRI_' and always refers to connections between Logic-Test (TRI) and other functional blocks.

33.8.1.1 LBIST domain assignment for TC4Dx

The following hierarchical scan domains are available in TC4Dx product:

Table 1352 Hierarchical scan domain assignment to LBIST_CTRL_DOM

Domain #	LBIST_CTRL_DOM	Domain Name	Description
0	SEL0	SRI0	Compute domain 1 (CPU0, CPU1, CPU2)
1	SEL1	SRI5	Communication domain (LETH, MCAN, I2C, RGMII)
2	SEL2	SRI4_Conv	Cyber security CPU and Converter
3	SEL3	SRI2	High speed interfaces (XGETH, PCIE)
4	SEL4	SRI3_PPU	PPU
5	SEL5	PMS	Power management
6	SEL6	SRI1	Compute domain 2 (CPU3, CPU4, CPU5, LMU)
7	SEL7	unused	

The default boot configuration of the LBIST control registers will select domains #0-#6. Those domains will be tested if automatic key-on LBIST execution after externally triggered cold power-on reset is selected in the UCB. In order to allow a successful LBIST execution of domain #3 the V_{DDPHYx} supply rails have to be connected to the V_{DD} supply voltage even if the respective HSPHYs are not used.

33.8.1.2 Key-on LBIST scenario for TC4Dx

The following default LBIST key-on configuration is provided automatically after cold power-on reset in TC4Dx A11 product. This will target only the safety logic in the selected hierarchical scan domains:

Table 1353 LBIST_CTRL_CFGA key-on default configuration for TC4Dx A11

Configuration type	Configuration value
Pattern count [PATTERNS]	5000 [1388_H]
iJTAG configuration speed [TFREQ]	25 MHz [11_B]
Safety LBIST test indicator [SAFESEL]	Test safety logic only [1_B]
Parallel Test Execution [ENPARA]	Run LBIST of hierarchical scan domains in parallel [1_B]
Key-off indicator [KEYOFF]	No key-off indication [0_B]
LBIST split-shift selection [SPLITSH]	Split-shift 2 [11_B]
LBIST frequency selection [FREQU]	100 MHz [0_H]

In TC4Dx A21 product the default LBIST key-on configuration has been changed in a way, that significant less patterns are executed with a reduced frequency. In this way the power consumption of key-on LBIST will be

reduced by roughly 50% but the test coverage goal of 90% is still achieved for the safety mechanisms in all hierarchical scan domains.

Table 1354 LBIST_CTRL_CFGA key-on default configuration for TC4Dx A21

Configuration type	Configuration value
Pattern count [PATTERNS]	2100 [834_H]
iJTAG configuration speed [TFREQ]	25 MHz [11_B]
Safety LBIST test indicator [SAFESEL]	Test safety logic only [1_B]
Parallel Test Execution [ENPARA]	Run LBIST of hierarchical scan domains in parallel [1_B]
Key-off indicator [KEYOFF]	No key-off indication [0_B]
LBIST split-shift selection [SPLITSH]	Split-shift 2 [11_B]
LBIST frequency selection [FREQU]	50 MHz [1_H]

The following table gives an overview about the maximal scan chain length for each hierarchical scan domain. LENGTHx register fields represent the scan chain lengths for key-off LBIST, SLENGTHx register fields determine the scan chain lengths for the key-on LBIST scenario. The maximal number of FFs in a domain's scan chain determines the number of clock cycles, which are required to execute one LBIST pattern and therefore is direct proportional to the LBIST execution time.

Table 1355 LBIST_CTRL_CFGB - LBIST_CTRL_CFGE maximal scan chain length

Domain #	Register	Field Name	Max chain length [Default value]
0 [SRI0]	LBIST_CTRL_CFGB	LENGTH0	166 [$A6_H$] (preliminary value)
0 [SRI0]	LBIST_CTRL_CFGB	SLENGTH0	52 [34_H] (preliminary value)
1 [SRI5]	LBIST_CTRL_CFGB	LENGTH1	77 [$4D_H$] (preliminary value)
1 [SRI5]	LBIST_CTRL_CFGB	SLENGTH1	51 [33_H] (preliminary value)
2 [SRI4]	LBIST_CTRL_CFGC	LENGTH2	53 [35_H] (preliminary value)
2 [SRI4]	LBIST_CTRL_CFGC	SLENGTH2	40 [28_H] (preliminary value)
3 [SRI2]	LBIST_CTRL_CFGC	LENGTH3	132 [84_H] (preliminary value)
3 [SRI2]	LBIST_CTRL_CFGC	SLENGTH3	46 [$2E_H$] (preliminary value)
4 [PPU]	LBIST_CTRL_CFGD	LENGTH4	65 [41_H] (preliminary value)
4 [PPU]	LBIST_CTRL_CFGD	SLENGTH4	40 [28_H] (preliminary value)
5 [PMS]	LBIST_CTRL_CFGD	LENGTH5	53 [35_H] (preliminary value)
5 [PMS]	LBIST_CTRL_CFGD	SLENGTH5	40 [28_H] (preliminary value)
6 [SRI1]	LBIST_CTRL_CFGE	LENGTH6	168 [$A8_H$] (preliminary value)
6 [SRI1]	LBIST_CTRL_CFGE	SLENGTH6	50 [32_H] (preliminary value)
7 [unused]	LBIST_CTRL_CFGE	LENGTH7	0 [0_H]
7 [unused]	LBIST_CTRL_CFGE	SLENGTH7	0 [0_H]

Expected MISR Signatures for key-on LBIST scenario in TC4Dx A11 product:

Table 1356 Expected MISR Signatures for key-on LBIST for TC4Dx A11

Domain #	Register	Field Name	MISR expected value
0 [SRI0]	LBIST_STAT_RESx_SIG (x=0)	SIGNATURE	$00AF297B_H$
1 [SRI5]	LBIST_STAT_RESx_SIG (x=1)	SIGNATURE	$00E616E6_H$
2 [SRI4]	LBIST_STAT_RESx_SIG (x=2)	SIGNATURE	$008D80F1_H$
3 [SRI2] 1)	LBIST_STAT_RESx_SIG (x=3)	SIGNATURE	$00C7C5D7_H$ 1)
4 [PPU]	LBIST_STAT_RESx_SIG (x=4)	SIGNATURE	$00771D59_H$
5 [PMS]	LBIST_STAT_RESx_SIG (x=5)	SIGNATURE	$00BBFC68_H$
6 [SRI1]	LBIST_STAT_RESx_SIG (x=6)	SIGNATURE	$0015E88B_H$
7 [unused]	LBIST_STAT_RESx_SIG (x=7)	SIGNATURE	$0000\ 0000_H$

1) Domain #3 can only be targeted for key-on or key-off LBIST if V_{DDPHYx} power supplies are connected either to a dedicated external source or to the V_{DD} supply voltage.

Expected MISR Signatures for key-on LBIST scenario in TC4Dx A21 product:

Table 1357 Expected MISR Signatures for key-on LBIST for TC4Dx A21

Domain #	Register	Field Name	MISR expected value
0 [SRI0]	LBIST_STAT_RESx_SIG (x=0)	SIGNATURE	$0000A886_H$
1 [SRI5]	LBIST_STAT_RESx_SIG (x=1)	SIGNATURE	$00193A6E_H$
2 [SRI4]	LBIST_STAT_RESx_SIG (x=2)	SIGNATURE	$008B531D_H$
3 [SRI2] 1)	LBIST_STAT_RESx_SIG (x=3)	SIGNATURE	$00BFAFE0_H$ 1)
4 [PPU]	LBIST_STAT_RESx_SIG (x=4)	SIGNATURE	$00B1FC21_H$
5 [PMS]	LBIST_STAT_RESx_SIG (x=5)	SIGNATURE	$004E80E9_H$
6 [SRI1]	LBIST_STAT_RESx_SIG (x=6)	SIGNATURE	$0030F32F_H$
7 [unused]	LBIST_STAT_RESx_SIG (x=7)	SIGNATURE	$0000\ 0000_H$

The following execution times are expected for key-on LBIST scenario for TC4Dx A11 product:

Table 1358 Expected MISR run times for key-on LBIST for TC4Dx A11

Domain #	Domain Name	Expected key-on LBIST run time [ms]
0	SRI0	~5.9
1	SRI5	~5.8
2	SRI4_Conv	~4.7
3	SRI2	~5.3
4	SRI3_PPU	~4.7
5	PMS	~4.7
6	SRI1	~5.7
7	unused	0
Parallel key on LBIST execution for all domains		~6

The following execution times are expected for key-on LBIST scenario for TC4Dx A21 product:

Table 1359 Expected MISR run times for key-on LBIST for TC4Dx A21

Domain #	Domain Name	Expected key-on LBIST run time [ms]
0	SRI0	~5.0
1	SRI5	~4.9
2	SRI4_Conv	~3.9
3	SRI2	~4.5
4	SRI3_PPU	~3.9
5	PMS	~3.9
6	SRI1	~4.8
7	unused	0
Parallel key on LBIST execution for all domains		~5.1

33.8.1.3 Key-off LBIST scenario for TC4Dx

The following LBIST key-off configuration is recommended for a complete LBIST test, targeting the full logic in the selected hierarchical scan domains in TC4Dx product:

Table 1360 LBIST_CTRL_CFGA key-off default configuration

Configuration type	Configuration value
Pattern count [PATTERNS]	5000 [1388_H]
iJTAG configuration speed [TFREQ]	25 MHz [11_B]
Safety LBIST test indicator [SAFESEL]	Test full logic [0_B]
Parallel Test Execution [ENPARA]	Run LBIST of hierarchical scan domains in sequence [0_B]
Key-off indicator [KEYOFF]	Activate key-off indication [1_B]
LBIST split-shift selection [SPLITSH]	Split-shift 2 [11_B]
LBIST frequency selection [FREQU]	33 MHz [2_H]

Expected MISR Signatures for key-off LBIST scenario in TC4Dx A11 product:

Table 1361 Expected MISR Signatures for key-off LBIST for TC4Dx A11

Domain #	Register	Field Name	MISR expected value
0 [SRI0]	LBIST_STAT_RESx_SIG (x=0)	SIGNATURE	$002709F2_H$
1 [SRI5]	LBIST_STAT_RESx_SIG (x=1)	SIGNATURE	$002DD51C_H$
2 [SRI4]	LBIST_STAT_RESx_SIG (x=2)	SIGNATURE	$005B1D61_H$
3 [SRI2] 1)	LBIST_STAT_RESx_SIG (x=3)	SIGNATURE	$00C1DEB4_H$ 1)
4 [PPU]	LBIST_STAT_RESx_SIG (x=4)	SIGNATURE	$00E6DE37_H$
5 [PMS]	LBIST_STAT_RESx_SIG (x=5)	SIGNATURE	$00B5BF18_H$
6 [SRI1]	LBIST_STAT_RESx_SIG (x=6)	SIGNATURE	$008FA6CD_H$

(table continues...)

Table 1361 (continued) Expected MISR Signatures for key-off LBIST for TC4Dx A11

Domain #	Register	Field Name	MISR expected value
7 [unused]	LBIST_STAT_RESx_SIG (x=7)	SIGNATURE	0000 0000$_H$

1) Domain #3 can only be targeted for key-on or key-off LBIST if V_{DDPHYx} power supplies are connected either to a dedicated external source or to the V_{DD} supply voltage.

Expected MISR Signatures for key-off LBIST scenario in TC4Dx A21 product:

Table 1362 Expected MISR Signatures for key-off LBIST for TC4Dx A21

Domain #	Register	Field Name	MISR expected value
0 [SRI0]	LBIST_STAT_RESx_SIG (x=0)	SIGNATURE	002709F2$_H$
1 [SRI5]	LBIST_STAT_RESx_SIG (x=1)	SIGNATURE	004FE4A0$_H$
2 [SRI4]	LBIST_STAT_RESx_SIG (x=2)	SIGNATURE	008DC9A7$_H$
3 [SRI2] *1)*	LBIST_STAT_RESx_SIG (x=3)	SIGNATURE	00F98D94$_H$ *1)*
4 [PPU]	LBIST_STAT_RESx_SIG (x=4)	SIGNATURE	00E6DE37$_H$
5 [PMS]	LBIST_STAT_RESx_SIG (x=5)	SIGNATURE	00B5BF18$_H$
6 [SRI1]	LBIST_STAT_RESx_SIG (x=6)	SIGNATURE	008FA6CD$_H$
7 [unused]	LBIST_STAT_RESx_SIG (x=7)	SIGNATURE	0000 0000$_H$

The following execution times are expected for key-off LBIST scenario:

Table 1363 Expected MISR run times for key-off LBIST

Domain #	Domain Name	Expected key-off LBIST run time [ms]
0	SRI0	~52.4
1	SRI5	~25.5
2	SRI4_Conv	~18.2
3	SRI2	~42.1
4	SRI3_PPU	~21.8
5	PMS	~18.2
6	SRI1	~53.0
7	unused	0
Serial key-off LBIST execution for all domains		~235

33.8.1.4 Miscellaneous LBIST settings for TC4Dx

The following miscellaneous LBIST default settings are used for LBIST_CTRL_CFGF and LBIST_CTRL_DOM registers:

Table 1364 LBIST_CTRL_CFGF default configuration

Register bit field	Configuration value
Clock gate transparency control [LBCGTRAN]	Three hot encoding [11_B]
Test observe point disabling [TPOBSDIS]	Test observe points enabled [0_B]
Test control point disabling [TPCTRDIS]	Test control points enabled [0_B]
Scan shift observation disabling [OBSSCDIS]	Scan shift observation points enabled [0_B]

Table 1365 LBIST_CTRL_DOM default configuration

Register bit field	Configuration value
LBIST hierarchical scan domain 0-7 selection [SEL7:0]	Domains #0-#6 are selected for LBIST [$0111\ 1111_B$] Domain #7 does not exist in TC4Dx product

33.8.2 TC4Dx TRI features

There are no deviations from the generic specification.

33.8.3 TC4Dx TRI functional description

There are no deviations from the generic specification.

33.8.4 TC4Dx TRI registers

33.8.4.1 Register address space - TRI

Table 1366 Registers address space - TRI

Module	Base address	End address	Note
TRI	$F0068000_H$	$F00680FF_H$	SCU: Connections to FPI/BPI bus

33.8.4.2 Register overview - access mode glossary

Table 1367 Register overview - access mode glossary

Keyword	Description
CSE	Access protection using PROT register TRI_PROTCSE .
E	Access protection using PROT register TRI_PROTE .
SE	Access protection using PROT register TRI_PROTSE .
APU-P	Protection group consisting of registers TRI_ACCEN_WRA , TRI_ACCEN_WRB , TRI_ACCEN_RDA , TRI_ACCEN_RDB , TRI_ACCEN_VM , TRI_ACCEN_PRS .
P	Access protection using *APU-P* registers.
SV	Access only when supervisor mode is active on the interconnect.
BE	Always returns a Bus Error.
U	No access restrictions.
PROT	Access restrictions as defined in the PROT register access rules.
STP	Access only during start-up, before start-up protection is set (SMM_STCON.ST_PROT=1).

33.8.4.3 Registers overview - TRI (ascending offset address)

Table 1368 Registers overview - TRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
TRI_CLC	Clock control register	000_H	P	P, SV, E	Application Reset	5695
TRI_ID	Module identification register	008_H	P	BE	PowerOn Reset	5696
TRI_PROTE	PROT register endinit	020_H	U	SV, PROT	Application Reset	5697
TRI_PROTSE	PROT register safe endinit	024_H	U	SV, PROT	Application Reset	5698
TRI_PROTCSE	PROT register cyber-secure endinit	028_H	U	SV, PROT	Application Reset	5700
TRI_ACCEN_WRA	Write access enable register A	040_H	U	SE, SV	Application Reset	5701

(table continues...)

Table 1368 (continued) Registers overview - TRI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
TRI_ACCEN_WRB	Write access enable register B	044_H	U	SE, SV	Application Reset	5702
TRI_ACCEN_RDA	Read access enable register A	048_H	U	SE, SV	Application Reset	5702
TRI_ACCEN_RDB	Read access enable register B	$04C_H$	U	SE, SV	Application Reset	5703
TRI_ACCEN_VM	VM access enable register	050_H	U	SE, SV	Application Reset	5703
TRI_ACCEN_PRS	PRS access enable register	054_H	U	SE, SV	Application Reset	5704
TRI_LBIST_CTRL_REQ	Logic BIST request register	060_H	U	SV, E	PowerOn Reset	5705
TRI_LBIST_CTRL_CFGA	Logic BIST configuration register A	064_H	U	SV, E	5706	5706
TRI_LBIST_CTRL_CFGB	Logic BIST configuration register B	068_H	U	SV, E	5709	5709
TRI_LBIST_CTRL_CFGC	Logic BIST configuration register C	$06C_H$	U	SV, E	5710	5710
TRI_LBIST_CTRL_CFGD	Logic BIST configuration register D	070_H	U	SV, E	5710	5710
TRI_LBIST_CTRL_CFGE	Logic BIST configuration register E	074_H	U	SV, E	5711	5711
TRI_LBIST_CTRL_CFGF	Logic BIST configuration register F	078_H	U	SV, E	5712	5712
TRI_LBIST_CTRL_DOM	Logic BIST domain selection	$07C_H$	U	SV, E	5714	5714
TRI_LBIST_STAT_TEST	Logic BIST status register	080_H	U	BE	5716	5716
TRI_LBIST_STAT_RESx_SIG (x=0-7)	Logic BIST result x register	084_H+x* 4	U	BE	5720	5720
TRI_DXCCTR	Debug over CAN enabling control	$0F8_H$	U	STP, CSE	5720	5720

33.8.4.4 Device specific registers

There are no device specific register changes.

33.8.5 TC4Dx TRI connectivity

The table below lists interfaces to and from this functional block to other blocks in the device.

Table 1369 List of TRI interface signals

Interface signals	I/O	Description
PMS_DFT_fBACK	In	Backup clock input
FPI_DFT_SIF	In	TRI SW slave interface
DFT_SMM_LBIST_reset	Out	Override system wide reset control for LBIST termination
DFT_SMU_alarms	Out	LBIST Alarm / Testmode Alarm Signals

33.8.6 TC4Dx TRI revision history

Reference	Description of change(s)
Date range: 2024-08-17 to 2025-01-10	
Key-on LBIST scenario for TC4Dx, Key-off LBIST scenario for TC4Dx	• Added MISR signature reference values for TC4Dx A21 key-on and key-off LBIST

34 Volatile Memory Test (VMT)

The Volatile Memory Test (VMT) controls and monitors the test, initialization, and data integrity checking functions of the various internal memories in the device.

Each SRAM has digital logic surrounding it, known as SRAM Support Hardware (SSH). The SSH is a hardware block that controls the error detection and correction, Memory Built-In-Self-Test (MBIST) of internal memories. Each SSH block provides a unified interface for controlling its various functions. There are multiple such SSH instances, each of which controls one or more of the different internal memories. The VMT provides register interfaces to configure and control these various memory controllers.

34.1 Feature list

- A unified register interface to control the operation and functionality of each internal SSH instance:
  - Each VMTx provides a unified register interface to control the operation and functionality of each internal SSH instance
  - Each VMTx can be used to control various configurable test types for each of the SRAM blocks in the system
- Data initialization:
  - Each SRAM block in the system can be hardware initialized through the VMT
  - Security sensitive memories can be auto-initialized to avoid unauthorized access
- Memory error correction and detection:
  - Memory error correction and detection for SRAM blocks configurable through the VMT
  - Correctable and uncorrectable error detection
  - Address error detection
- Alarm notification to the SMU for each SRAM or SSH:
  - From each SRAM or SSH, three alarms are sent to the VMT, which are then OR combined for each alarm type and forwarded to the SMU, although separately for security and non-security related SSHs. The alarms are:
    - Correctable error (CE) alarm
    - Uncorrectable error (UCE) alarm
    - Miscellaneous error (ME) alarm
- MBIST done interrupt to signal the completion of all running tests

34.2 Functional overview

Multiple VMT instances (VMTx) are distributed over the chip. Each VMTx handles up to 32 SSHs. The figure shows the system level view of the VMTx, SSH instances, and SRAMs.

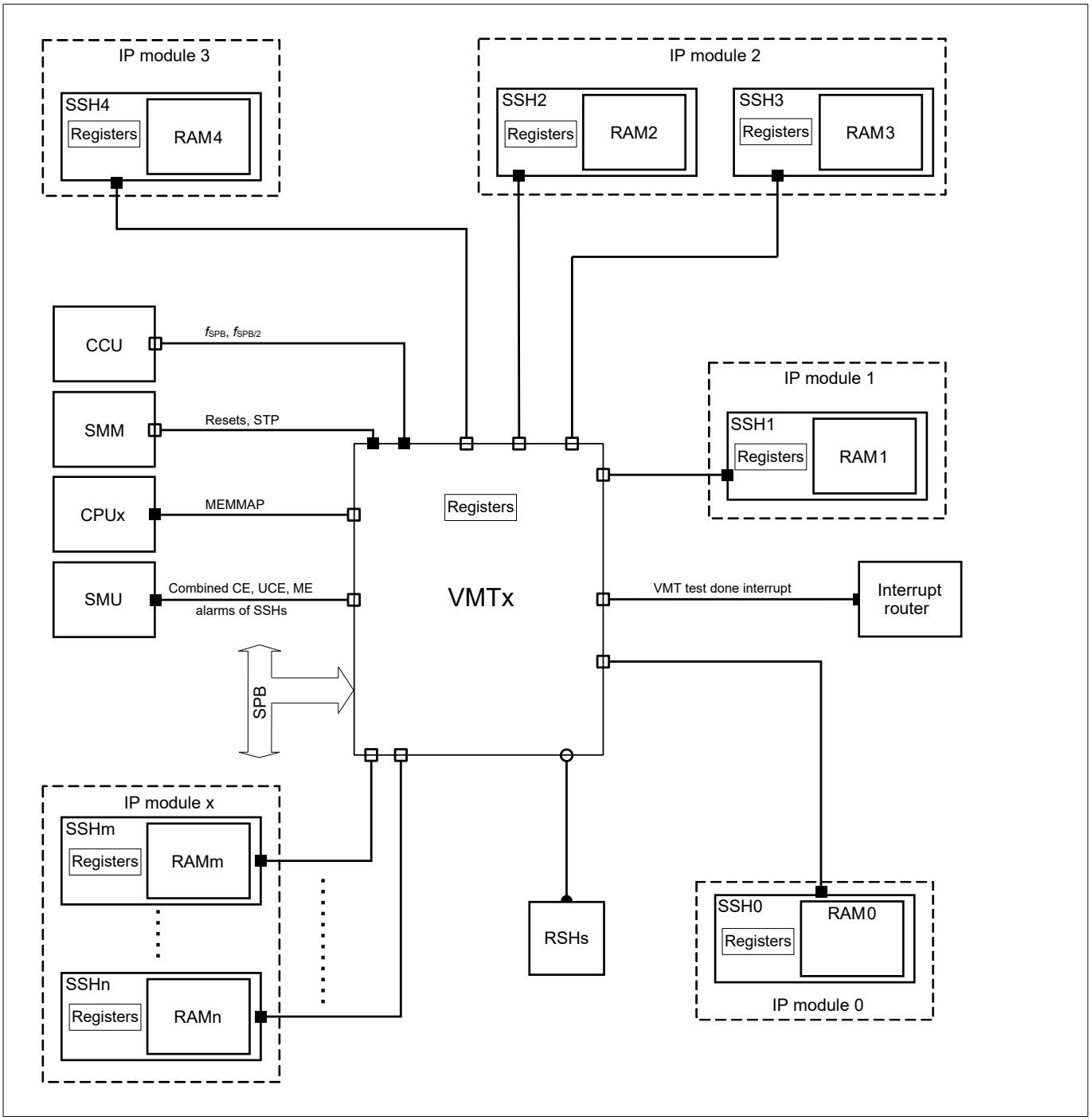


Figure 760 VMT block diagram

For the alarm description please refer to the alarm mapping tables in the SMU functional block and the Safety Manual.

Note that STP connections from SMM as well as ROM support hardware (RSH) connections shown in the figure above are not relevant for the user and will not be explained in greater detail herein.

Different functional blocks such as the CPU, LMU, or CAN for example, may have one or more SRAMs inside them. Every SRAM is embedded in its own SSH module, with its own SSH registers and MBIST logic. Each SSH is separately connected to a VMTx through internal interfaces.

Each VMTx is connected to the SPB and provides access to the registers of the SSHs connected to it.

Each SPB access is internally forwarded and returned, to and from the corresponding SSH by the VMTx.

34 Volatile Memory Test (VMT)

Related information

TC4Dx SMU alarm mapping tables on page 7233

34.3 Functional description

The following chapters provide a functional overview of VMTx, SSH, and SRAM.

34.3.1 SRAM support hardware (SSH)

Any on-chip SRAM in the AURIX™ 3rd Generation is assigned to an SSH (SRAM support hardware) for test purposes. A single SSH instance can cover multiple on-chip SRAMs. Any SSH is assigned to one of the implemented VMT modules. All SSH instances are connected to an internal bus through a unified VMTx-SSH Interface. The SSH provides direct access to the memories, without involving the CPU.

34.3.2 Control and status interfaces

The following section explains the hardware interface to access each individual SSH instance by the CPU through the system peripheral bus (SPB).

34.3.2.1 Interface to the CPU

The registers of an individual SSH can be accessed by the CPU via the VMTx through the peripheral bus.

The registers of individual controllers are located in the SSH block. Logically they all have a VMTx system address and are accessed through SPB accesses from the VMTx. As this goes through the VMTx->SSH interface, the accesses to individual SSH registers will be slower than normal SPB accesses to peripheral registers.

***Note**: Any SSH register writes are ignored during an ongoing test and the values are not written to the register, however a read access will return the correct value. In both cases the access finishes normally on the SPB without bus errors. The MCi_MCONTROL can be written to during a test to clear the start and resume bits.*

34.3.3 Enabling the SRAM support hardware (SSH)

Each SSH has an enable bit associated with it in the MEMTEST(CS). Refer to the register descriptions for the implemented enable bits.

The control registers of each SSH are accessible only when the SSH is enabled (MEMTEST(CS).ENn = 1).

In each SSH instance, the only exceptions to this are:

- MC i Nominal ECC safety register
- MC i Nominal Memory ECC detection register
- MC i Nominal error information register
- MC i Nominal error tracking register
- MC i Nominal alarm sources configuration register
- MC i Nominal SSH safety faults status register

These registers are available at all times to permit easy runtime access to ECC features. That is, above noted registers can be read and written with the only precondition that the VMT clock is enabled in the CLC register and no register access protections are violated.

All other registers in the SSH can only be accessed when the SSH is enabled in the MEMTEST(CS) register, otherwise the VMT returns a bus error on the SPB.

For certain SSHs, enabling or disabling the SSH in the MEMTEST(CS) register may result in whole or part of the memory being automatically initialized, depending on the configuration. This auto-initialization can take hundreds of clock cycles to complete. During this time, no SSH register is accessible, including registers that are

normally accessible without the SSH being enabled. Any register access to the SSH running the auto-initialization will result in a bus error on the SPB.

***Note**: The clock input to the module containing the SRAM has to be enabled upstream in the CCU of the CLOCK functional block, before any SSH registers can be accessed through the VMT. SSH register access is not gated by the local CLC register of that module.*

1. When an SSH is enabled, functional access to the memory is unavailable. When a memory is being tested, the software shall prevent any functional access to that memory. For example, while testing a CPU's local memory, the CPU should be in an idle state, or executing from other memory.
2. Correct execution of an SSH operation, for example an MBIST test, requires that both the VMTx and the module containing the memory-under-test are operational for the duration of the SSH operation. A reset of the module containing the memory-under-test, may result in unexpected behavior when an SSH is enabled. Software has to take care that events such as application reset, module resets, standby entry, clock frequency change, and so forth are not triggered when an SSH is enabled. For example, a module reset may result in the module trying to access the SRAM functionally after the reset, when the SSH is still enabled, therefore functional access is disabled. Such scenarios have to be avoided by software.
3. If communication is ongoing (for example, SSH register read or write) between the VMTx and SSH when a module reset occurs, the communication is aborted and a bus error is triggered on the SPB within a few cycles, without the need to wait for a VMTx-SSH communication timeout. That is, an SPB error during reads and silent ignore (no error) during writes. The SSH also generates a miscellaneous alarm in this case and the corresponding error status is stored in a FAULTSTS register. Only the ongoing read/write may be corrupted, future accesses are not affected
4. In case a non-existing SSH register is accessed in an existing SSH instance, this will result in a miscellaneous alarm and the corresponding error status is stored in FAULTSTS. If a non-existing SSH instance is accessed, it will result in a bus error on the SPB.
5. If a non-existing VMT register is accessed, it will result in a bus error on the SPB.

Related information

Security-sensitive memories and auto-initialization on page 5737

34.3.3.1 Security-sensitive memories and auto-initialization

Certain internal memories are considered security sensitive. Each VMTx supports up to 32 SSHs. A SSH on index "i" is either considered nominal or security sensitive. The configuration registers in the VMTx are separated for security sensitive and nominal SSHs to allow for different protections. However, the general functionality of the registers is the same. To configure or enable security sensitive SSHs, the cyber security (CS) registers in the VMTx have to be used:

Table 1370 VMT registers for nominal and security sensitive SSHs

Nominal SSH MCi	Security sensitive SSH MCCSi
MEMTEST	MEMTESTCS
MEMMAP	MEMMAPCS
MEMSTAT	MEMSTATCS
MEMDONE	MEMDONECS
MEMFDA	MEMFDACS
CEALARM	CEALARMCS

(table continues...)

Table 1370 (continued) VMT registers for nominal and security sensitive SSHs

Nominal SSH MCi	Security sensitive SSH MCCSi
UCEALARM	UCEALARMCS
MEALARM	MEALARMCS

Security sensitive memories support auto-initialization. For certain applications which are security sensitive, reading or modifying the contents of these memories through the SSH modules has to be prevented by the VMTx. For these memories, enabling the corresponding SSH results in the memory being cleared, before the SSH is actually enabled. This operation can take hundreds of clock cycles. The software must wait for the MEMSTAT(CS).AIUn bit to be cleared before the SSH will be enabled.

Refer to subchapter "VMT security information" in the device specific configuration chapter for more information on which memories are considered security related and which memories feature auto-initialization.

In the case of non-security applications, for example safety, it may be important that the modification of the memory contents is allowed through the SSH, for example ECC error injection.

It is possible to enable or disable the auto-initialization and partial-erase features. Refer to the RAMINIT description in Register Overview - UCB0_17 (ascending Offset Address) and the CSSRAM_init description in Register Overview - UCB1_15 (ascending Offset Address), which can be found in the FIRMWARE chapter of the User Manual.

Related information

VMT security information on page 5841

34.3.3.1.1 Security applications

For security applications, the contents of the security sensitive memories can be auto-initialized to erase the existing contents. Automatic initialization of security-sensitive memories is triggered whenever the corresponding MEMTEST(CS).ENn or MEMMAP(CS).ENn is changed, that is, the corresponding SSH is enabled or disabled or if memory map mode is changed.

The MEMSTAT(CS).AIUn bits indicate whether an automatic data initialization of memory n has been triggered and whether the initialization has completed. If an AIUn bit in MEMSTAT(CS) register is set, this means that auto-initialization for that memory is still underway. The SSH is enabled or disabled only after the auto-initialization has completed. The software can wait for the auto-initialization to be completed by polling this bit.

34.3.3.1.2 Nominal applications

If RAMIN=11_B in the RAMINIT or CSSRAM_init registers, then no automatic initialization of RAM content is performed when SSH modules are enabled or disabled with MEMTEST(CS).ENn or MEMMAP(CS).ENn register bits.

This permits the use of the SSH by an application without unwanted corruption of memory content. Use cases for this include error injection, data modification or runtime memory testing.

34.3.3.2 Memory map selection

The memory mapping enable register (MEMMAP(CS)) has configurable control bits to select memory-mapped test mode for CPU cache and tag memory. The cache and tag memories are mapped to the system address space when the corresponding control bit is set. The MBIST tests and other SSH operations work on the SRAM independent of the MEMMAP(CS) register settings.

Refer to subchapter "MEMMAP configuration" in the device specific configuration chapter of this manual for more information.

Related information

MEMMAP configuration on page 5837

34.3.4 SRAM support hardware (SSH) operation

The operation and functionality of each internal SSH instance can be controlled from a set of registers. These registers are accessible through the VMTx. An SSH instance has to be enabled first before the memory can be tested.

The main functionalities provided by each SSH instance are explained in the following sections.

Related information

Enabling the SRAM support hardware (SSH) on page 5736

34.3.4.1 Memory testing and initialization

Here we explain how to configure the SSH instances using the VMTx, to perform various tests on the memory and obtain the results, or initialize the memories from the SSH.

***Note**: The following operations can be performed only when the SSH instance under test has been enabled using the corresponding ENn bit in the MEMTEST(CS) register.*

34.3.4.1.1 Starting a memory test sequence

As a first step, before the MCONTROL.START is set, the software should properly configure the configuration registers, CONFIG0 (MC i Nominal configuration register 0) and CONFIG1 (MC i Nominal configuration register 1) registers.

CONFIG0.ACCSTYPE specifies the access type (read or write) to be performed on each single address in the current test sequence, while CONFIG1.ACCSPAT specifies the access pattern. CONFIG0.NUMACCS specifies the total number of accesses to a single address in the current marching element. The SSH supports some complex addressing schemes. The software can enable this using the CONFIG1.AG_MOD bits.

Each memory test sequence is started by writing to the MCONTROL.START bit of the corresponding SSH (MC i Nominal MBIST control register). Once started, the SSH test hardware proceeds to test the SRAM by accessing each memory address one after the other until the last address of the memory test range is reached. When the test is complete, the MSTATUS.DONE (Nominal memory test done status register) is set by the hardware, if the software has cleared the MCONTROL.START bit. Software can poll the MEMDONE(CS) (Nominal or Cyber-secure memory test done status register) in the VMTx to get the status of test completion. This register just reflects the DONE bit in the MSTATUS register of each SSH.

It is possible to select a fixed range of the memory to be tested, instead of the whole memory, using the RANGE.RAEN bit (MC i Nominal range register, single address mode). When RANGE.RAEN is set to ‘1’, the RANGE.ADDR[14:0] bits are interpreted as two separate addresses, providing the upper and lower boundaries of the range to be tested. The lower address boundary is given by (RANGE.ADDR[6:0] & “00000000”b) while the upper boundary is given by (RANGE.ADDR[13:7] & “11111111”b).

Once the test is complete (MSTATUS.DONE=1), the MSTATUS.FAIL bit will be set in case of any failures.

34.3.4.1.2 MBIST done interrupt

On completion of all running tests, a VMTx signals an interrupt to the interrupt router (IR). The IR provides a dedicated interrupt control register SRC_VMTwSR for each VMTx. The reset value of this signal is high. When a test on any SRAM is started, this signal goes low. On completion of all ongoing tests, this signal again goes high, and this rising edge triggers the interrupt.

34.3.4.1.3 Getting detailed memory test results

The MSTATUS.FAIL and MSTATUS.DONE bits, which can be polled from the VMTx itself through the MEMDONE(CS) (Nominal or Cyber-secure memory test done status register) and MEMFDA(CS) (Nominal or Cyber-secure memory test FDA status register), provide a general pass and fail information as well as test completion status.

If MCONTROL.FAILDMP = 1, the test stops after a failure and the fail information is immediately available for dump. MSTATUS.FDA (fail dump available) is set in this case. Any dump information has to be polled from registers RDBFLy and ETRR. RDBFLy contains the fail bit map and ETRR the failed address. Reading MSTATUS and the RDBFLy(n-1) registers with MSTATUS.FDA = 1 will reset MSTATUS.FDA back to 0. A consequent setting of MCONTROL.RESUME will resume the interrupted test sequence.

The fail bitmap is simply (expected data pattern) XOR (actual data pattern).

The RANGE register can be used to run consecutive tests on constantly shifted memory ranges so that in the end the complete memory has been analyzed.

Please note that if a test is stopped due to an intermediate failure, the MSTATUS.DONE is not set. MSTATUS.DONE is set only once the whole test sequence is completed and MCONTROL.START has been cleared by software.

***Note**: In order to finish the test without the fail information, MCONTROL.FAILDMP should be reset by SW in the same write access to the MCONTROL, where MCONTROL.RESUME is set.*

34.3.4.1.4 Error injection during memory tests

It is possible to manually inject errors during memory tests, for example, to test the software.

For a non-destructive MBIST test, an error in the data can be introduced by programming a word with the wrong ECC before the test, using the ECCMAP bits.

During a destructive MBIST test, it is possible to inject data errors with the MCONTROL.INJERR bit. When this bit is set (and RANGE.RAEN = 1), then the RANGE.ADDR field is taken as a pointer to a physical SRAM location to which write accesses during the test are not executed. The following steps show how to use this method:

- Write ones at the memory location where error injection is desired with the procedure explained in the section „Writing to a single memory location“
  - For a single bit error, write a single one while keeping all other bits to zero
  - For two or more bit errors, write two or more ones accordingly
- Configure the destructive MBIST as explained in the corresponding chapter, but for the steps Kx:Test or K3:Extract set the MCONTROL.INJERR to one, set RANGE.RAEN to one and then write in RANGE.ADDR the address of the memory location where the error was previously injected
- Start the MBIST by setting MCONTROL.START to one

The destructive MBIST then runs over the full memory, and writes are not executed on the address corresponding to RANGE.ADDR. This results in a mismatch of the expected data during the test, resulting in a FAIL.

The method explained above is limited to a single address in the memory. For injection of errors at multiple memory locations, like for example multiple single bit errors or a bitline error, the following method can be applied:

- Initialize the memory to all zero by following the procedure explained in the section „Filling a memory with defined contents“
- Write ones at the memory locations where error injection is desired with the procedure explained in the section „Writing to a single memory location“
  - For multiple single bit errors, write a single one while keeping all other bits to zero at each memory location
  - For two or more bit errors, write two or more ones accordingly
  - For a bitline error, write a single one at the same bit position for at least three successive memory locations
- Start the destructive MBIST as explained in the corresponding chapter, but for the steps Kx:Test or K3:Extract use the following sequence instead: {r0,w1,r1,w1}
  - While 3 accesses {r0,w1,r1} would be sufficient, using 4 keeps the runtime of the MBIST neutral compared to the original destructive MBIST procedure

The destructive MBIST then runs over the full memory and during the first read access, any bits previously set to one are identified as failed bits.

With errors injected, all normal diagnostics and notifications can be tested. That is, alarms are triggered, errors are tracked in the ETRR/ERRINFO registers during a non-destructive test. During a destructive MBIST test, the MSTATUS FAIL bit is set, and when MCONTROL FAILDMP = 1, the fail bitmap is captured in the RDBFLy registers, MSTATUS.FDA is set and the failing local address is captured in the ETRR register.

Address errors are triggered during an MBIST by simply setting SFLE bit to 1.

***Note**: Setting SFLE bit to 1 will trigger an address error from each address that is accessed until the SFLE is set back to 0.*

Related information

Writing to a single memory location on page 5743
Destructive MBIST on page 5744
Filling a memory with defined contents on page 5741

34.3.4.1.5 Filling a memory with defined contents

The SSH can be used to fill a memory range or a complete memory with a defined pattern very fast, that is, one write access per cycle with the full memory data width, using the MCONTROL.DINIT bit. For this, ensure that MCONTROL.SRAM_CLR = 0.

Before setting the MCONTROL.DINIT, the software should first fill the RDBFL register with the desired bit pattern (MC i Nominal read data and bit flip register y). It is not mandatory to have a valid ECC in this pattern.

Next, the RANGE register needs to be set with the memory range into which the pattern needs to be filled.

Next, set the MCONTROL.DINIT, and then the MCONTROL.START bit should be set to start initializing the RAM. The software should then clear the MCONTROL.START.

When MSTATUS.DONE bit is set by the hardware, the memory filling operation is complete.

***Note**: It is generally recommended to initialize the whole address range of an SRAM.*

34.3.4.1.6 Initializing SRAMs

Using the MCONTROL.SRAM_CLR bit, it is possible to initialize the complete SRAM.

***Note**: The SRAM_CLR functionality shall not be used with the SSH instances specified with "SRAM clear = no" in the chapter "SSH functionality exceptions". If an SRAM_CLR operation is started on any of these memories, its content is filled with ECC-incorrect values which can then lead to ECC alarms when accessing them. These memories shall be initialized by the application software instead, for example with an initialization routine or a DMA request. For the software initialization, it is recommended to use the TriCore™ CPU instruction ST.D (Store Doubleword) to avoid potential traps and alarms due to ECC errors.*

For this operation, enable the SSH (MEMTEST(CS).ENn = 1), and:

1. Set the MCONTROL.SRAM_CLR
2. Start the initialization using MCONTROL.START
3. Clear the MCONTROL.START
4. Wait for the end of the initialization by polling the MSTATUS.DONE (can also be polled using the MEMDONE(CS) register) bit
5. Clear the MCONTROL.SRAM_CLR and leave the test mode (MEMTEST(CS).ENn = 0)

***Note**: This will trigger an UCE alarm. Therefore, the alarm reaction may need to be disabled before. Software can set ALMSRCS.OPENE before.*

With this initialization, the complete SRAM will be filled with ECC-correct zero value.

Clear the UCERR and OPERR flags set due to this operation. Re-enable ALMSRCS.OPENE if it was disabled before the test.

***Note**: For SRAMs described in chapter "Secure-sensitive memories and auto-initialization" enabling or disabling the SSH using the MEMTEST(CS) register takes time for the initialization to complete. Therefore software must wait for the corresponding MEMSTAT(CS).AIUn bits to be cleared, to ensure that this operation is complete.*

Related information

SSH functionality exceptions on page 5835
Security-sensitive memories and auto-initialization on page 5737

34.3.4.1.7 Reading a single memory location

The SSH can also be used to read the content of a single word. The RDBFLy register holds the content of a complete memory word and so it is possible to read all memory bits, including ECC bits. The necessary steps are:

***Note**: This will trigger an UCE alarm. Therefore, the alarm reaction may need to be disabled before.*

1. Enter memory test mode by setting MEMTEST(CS).ENn = 1
2. Initialize registers
   a. RANGE.RAEN:= 0 (range disabled = single address) and address to be read in RANGE.ADDR
   b. CONFIG0:= 1001_H (NUMACCS = 1_H, ACCSTYPE = 01_H (read))
   c. CONFIG1:= 0000_H (linear mode, non inverted pattern)
3. MCONTROL:= 0009_H (FAILDMP = 0, direction up, start): Start read operation. MSTATUS.DONE will be cleared now

4. MCONTROL:= 0008_H (clear START)
5. Wait for MSTATUS.DONE to be set again (poll the corresponding bit in the MEMDONE(CS))
6. Read RDBFLy register
7. Leave memory test mode by setting MEMTEST(CS).ENn = 0
8. Clear the UCERR and OPERR flags set due to this operation

34.3.4.1.8 Writing to a single memory location

The SSH can also be used to write the content of RDBFLy register to a single memory location. RDBFLy holds the contents of a complete memory word and so it is possible to write to all memory bits, including ECC bits. The necessary steps are:

***Note**: This will trigger an UCE alarm. Therefore, the alarm reaction may need to be disabled before.*

1. Enter memory test mode by setting MEMTEST(CS).ENn = 1
2. Initialize registers RDBFLy: = write data
   a. RANGE: = RAEN = 0 (range disabled = single address) and address to be written to in RANGE.ADDR
   b. CONFIG0: = 1000_H (NUMACCS = 1_H, ACCSTYPE = 00_H (write))
   c. CONFIG1: = 0000_H (linear mode, non inverted pattern)
3. MCONTROL: = 0009_H (FAILDMP = 0, direction up, start): Start write operation. MSTATUS.DONE will be cleared now
4. MCONTROL : = 0008_H (clear START)
5. Wait for MSTATUS.DONE to be set again (poll the corresponding bit in the MEMDONE(CS))
6. Leave memory test mode by setting MEMTEST(CS).ENn = 0
7. Clear the UCERR and OPERR flags set due to this operation

Related information

SRAM addressing on page 5754

34.3.5 Resets and clocks in the VMTx, SSH and SRAM

The sections below provide details about the clocks and reset types affecting the VMTx, the SRAMs embedded in each module, and their surrounding SSHs.

34.3.5.1 Clock domains

The VMTx runs on the SPB clock. The interface between the VMTx and SSH runs on a clock which is always at a fixed divider of 1/2 times the SPB clock. The SRAM and the SSH logic registers, such as Finite State Machine (FSM), run at the module clock frequency, that is, the module where the SRAM is embedded in. The SSH register accesses and alarm forwarding still work correctly even if the system is running with a lower clock divider in the LPDIV mode (see SYSCCUCON0 CLOCK register description).

Table 1371 VMT, SSH and SRAMs clock domains

	Clock
VMTx	fSPB, Communication to SSH on fSPB/2 clock
SSH	Communication to VMTx on fSPB/2 clock, other SSH logic on module clock
SRAM	Module clock
Alarms	Propagated to SMU on fSPB/2 clock

34.3.5.2 Reset domains

There are different reset domains to be considered within the VMTx, SSH, and SRAMs.

Table 1372 VMTx, SSH and SRAM reset domains

Module/Register/Function	Reset domain
VMTx and alarm path VMTx<->SMU	Application reset
VMTx: Interface to SSH (including FFs in the interface path)	Application reset
SSH: ECCS, ALMSRCS registers and notification enable bits	Application reset
SSH: Alarm status flags (ECCD.CERR, UCERR, and MERR)	Application reset
SSH: ETRR/ERRINFO, FAULTSTS registers and ECCD.VAL	Warm PORST
SSH: Test related registers (MCONTROL, CONFIG0, CONFIG1, MSTATUS, RDBFLy, and RANGE)	Application reset
SRAM	Cold PORST

34.3.5.2.1 Alarm status after reset

If an alarm occurred, the system may perform a reset.

The alarm status bits in the ECCD register (CERR, UCERR and MERR) are cleared after an application reset.

However, the error status bits in ETRR, ERRINFO and FAULTSTS registers are still available for diagnostic purposes until the next power-on reset. They can only be cleared by software and are reset only with a warm PORST.

The alarm itself is also cleared with an application reset. This prevents a single alarm creating a reset loop.

34.3.6 MBIST algorithms

In order to check the integrity of the SRAM, an MBIST may be run by configuring the SSH.

The test type and parameters are programmed using the SSH registers CONFIG0 and CONFIG1, and certain parameters in the MCONTROL register.

Additional memory test algorithms are supported by the SSH, but are used in production test modes only.

34.3.6.1 Destructive MBIST

A destructive MBIST algorithm is implemented that offers a higher permanent hardware random fault diagnostic coverage than the former non-destructive inversion test (NDIT). The basic principle of operation is that the SSH hardware writes a known pattern into a specific memory location and reads it back. By comparing the read data with the expected value, faults in individual cells and memory logic can be found.

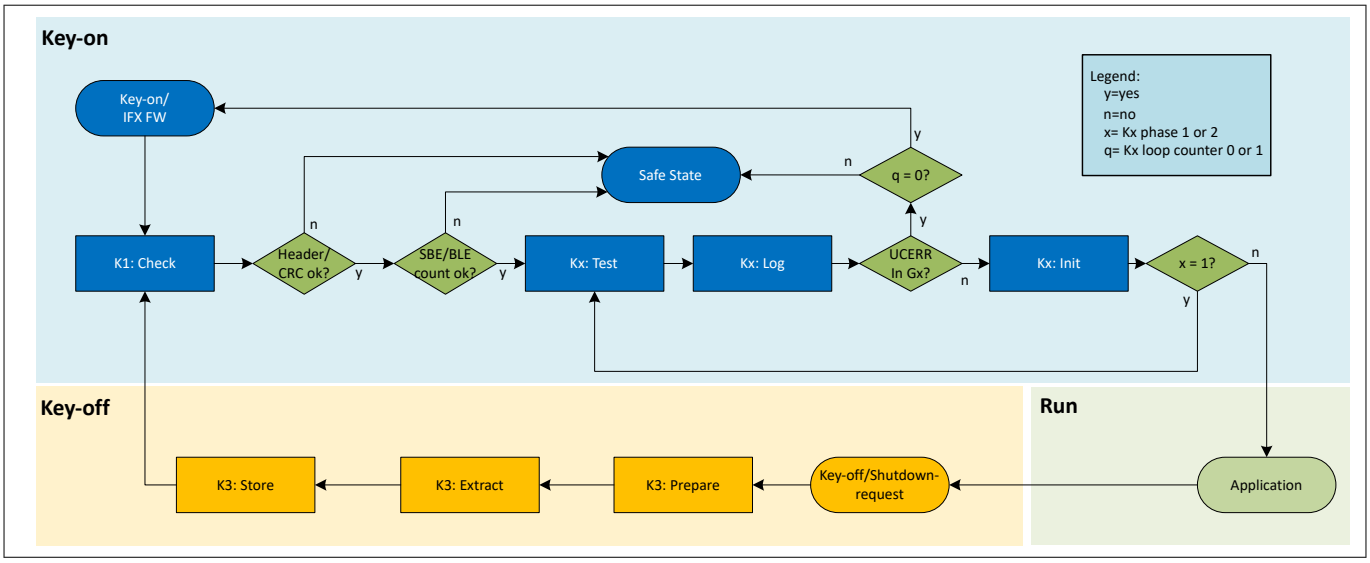


Figure 761 MBIST Cycle

The destructive MBIST is divided in two macro-phases:

- Key-on MBIST (K1 and K2)
- Key-off MBIST(K3)

The separation into macro-phases essentially addresses start-up time and load jump constraints, and is the first choice in case system start-up time is critical. Both key-on and key-off macro phases could be merged into a single consolidated flow if the system start-up time requirements allow it. Such procedure saves complexity induced by the data-transfer from key-on to key-off.

The key-on phase essentially focuses on identification of uncorrectable errors (UCERR); in case of occurrence, a retest cycle shall be initiated. Typically done by reset trigger and re-execution of MBIST, in order to distinguish soft from persistent fault. The existence of UCERR is regarded as serious degradation resulting in unavailability of the functions related to the affected memory.

Both key-on and key-off MBIST phases are executed from the Main-Ctrl-Core (typically CPU0) and need a Working RAM (WR) to temporarily store test relevant data. Since RAM access is blocked while the test is running, the WR needs to be tested separately. For this purpose, the key-on phase MBIST is divided into two sub-phases K1 and K2:

- K1 shall test at least the *Working-RAM of Main-Ctrl-Core* (WR)
- K2 all others (group G2 which consists of subgroups G2_1 and G2_2)

G1, G2_1 and G2_2

To provide better guidance into which memories are targeted by each test phase, the MCU memory instances are divided into three groups: G1, G2_1, and G2_2. The mapping of the memory instances to each memory group can be found under the chapter "SSH instances" in the VMT device specific section of this document.

In order to integrate considerations for cache enabling and secure-boot, K1 naturally combines test of TAG and DMA-RAMs, which are allocated to group G1. G2_2 contains SRAMs integrated to modules providing own ECC-encoding, G2_1 all others which are using Infineon ECC algorithm.

The key-on macro-phases K1 and K2 are both source and recipient of information with respect to key-off macro-phase K3. As mentioned before, the key-on phase focuses on detecting critical errors like UCERR. In case no critical errors are found, K1 and K2 are used to scan for non-critical permanent errors, that is, errors that do not have an impact on safety nor availability of the computing platform.

The result of this scan is a list of the memory instances containing non-critical errors that is provided as input to for K3. This list is provided in a data structure called FAILID which is explained with more details later in this chapter. K3 then takes the FAILID structure and performs a more detailed analysis on each of the memory instances to count and classify the non-critical errors.

This refined scan information is then passed from K3 to K1. Each K1 execution receives and checks the information provided by the previous K3 cycle in order to judge if the accumulated non-critical errors are above the tolerable threshold.

Micro-phases

The targets described above, explain the further split of macro-phases into micro-phases, namely:

- K1 holds a Check-Phase (K1: Check); interlinked with the result stored by K3 (K3: Store)
- K1 and K2 contains Test and Log phases (Kx: Test and Kx: Log)

The K3: Prepare micro-phase shown in the previous MBIST cycle figure reads in the FAILID data structure provided by the key-on phase to select the memory instances that need further analysis. The extraction phase K3: Extract then performs the detailed analysis of non-critical permanent errors that serves essentially two purposes:

- *Logging* of occurring fail patterns, that is, determination of error number and type for G1 and G2
- Identification of early SRAM degradation revealing as fast accumulation, relative to number of reset cycles and operating hours, of correctable hard errors. The thresholds for accumulated errors are specified in the product specific datasheet under the parameters Maximum number of acceptable single-bit errors (M_{SBE}) and Maximum number of bitline-oriented errors (M_{BLE})

In the case that during the next K1 execution the K1: Check phase detects a violation of the M_{SBE} or M_{BLE} thresholds, K1 has to transfer the MCU to a safe state. The maximum acceptable error thresholds are related to decreased robustness due to potential combination of hard errors with soft errors. It is derived from technology-specific base-failure rate and MCU-specific SRAM architecture.

Data storage

The data exchanged between K3: Store and K1: Check shall be stored in a way that its availability is ensured for the next key-on event, for example, by saving the FAILID structure in the EEPROM area at the end of K3: Store. Data to be provided from Kx: Log to K3: Prepare can alternatively be intermediately stored in RAM (MCU stays powered). In the latter case, to avoid interference from other software tasks, it is recommended to restrict the access to the selected RAM region, for example, by using the range-adjustable APU of DLMU.

In order to enable validation of respective data integrity, FAILID contains a header structure containing its Storage Address Location (SAL), a flag indicating its validity (V-flag), as well as CRC. Correct start-up, even in case of fails during programming cycle (for example, non-stable supply), can be assured by a redundancy approach for FAILID: Storage locations for FAILID might be foreseen twice (FAILID_ORIG/_COPY) with the V-flag (one in ORIG and one in COPY) being used to indicate the validity state.

Consequently, during all steps of an update, one of the copies is kept valid. This redundant storage strategy is however not mandatory: it is up to the system integrator to decide if it is worth the effort.

Implementation

Implementations of K1, K2, and K3 employ a destructive test-approach, that is, SRAM data content is overwritten by test pattern and read back/compared with expected value. In this sense, it does not distinguish between logical and ECC-bits, but provides full coverage for hard errors irrespective of bit-usage and SRAM architecture. Therefore, after the destructive testing is completed, the Kx: Init initialization phase is needed to bring the tested RAMs to an ECC-coherent state.

For the memory groups G1 and G2_1 the SSH can be used for the initialization as shown in the subchapter Initializing SRAMs. For memories in the group G2_2, the initialization can only be performed by software-based write operations. For certain functional blocks (for example PPU) this could be done later as part of their module initialization sequence and therefore not necessarily part of Kx: Init.

FAILID reference structure

HEADER						
VMTj	MCi	#SBE	#BLE	Error-Address0	Failposition0	FailMarker0
				Error-Address1	Failposition1	FailMarker1
				Error-Address2	Failposition2	FailMarker2
				…	…	…
				Error-Address9	Failposition9	FailMarker9
…	…	…	…	…	…	…
CRC						

Figure 762 FAILID reference structure

The figure is a visual representation in table form of the contents and structure of the FAILID data structure. The fields colored with yellow are the ones provided by the K3: Store phase and saved to non-volatile memory. They consist of:

- Header: containing the SAL and the V-flag, where a V-flag value equal to $57B5327_H$ means valid while any other means invalid
- VMTj and MCi correspond to the indexes of the RAM instance as provided in the SSH instances table in the device specific section of this chapter
- #SBE and #BLE contain respectively the amount of single bit errors and bit-line errors found
- CRC contains the CRC checksum for the entire FAILID structure stored in non-volatile memory (yellow fields), of course excluding the CRC itself

The fields colored with green on the other hand, are used only temporarily by the K3: Extract phase and not saved after key-off. They consist of:

- Error-AddressX with X=0...9: local physical address to the specific RAM instance, not the system address. It is provided by the MBIST engine in the ETRR register whenever an error is found by the destructive test procedure
- FailPositionX[63:0] with X=0...9 contains the position of a failed bit within an address. For example, if the 6th bit of an address is found to be faulty, then this field's value would be 6_H
- FailMarkerX[15:0] with X=0...9 contains the type of error found: 0_H = no error, 1_H = SBE and 2_H = BLE

***Note**: Array of Error-Address, vectors for Failposition and FailMarker shall be initialized to 0_B.*

***Note**: The number of rows per RAM instance in the green portion of the FAILID data structure depends on the M_{SBE} datasheet parameter. See also step 5 of K3: Extract below.*

Related information

SSH instances on page 5825

34.3.6.2 MBIST cycle test phases

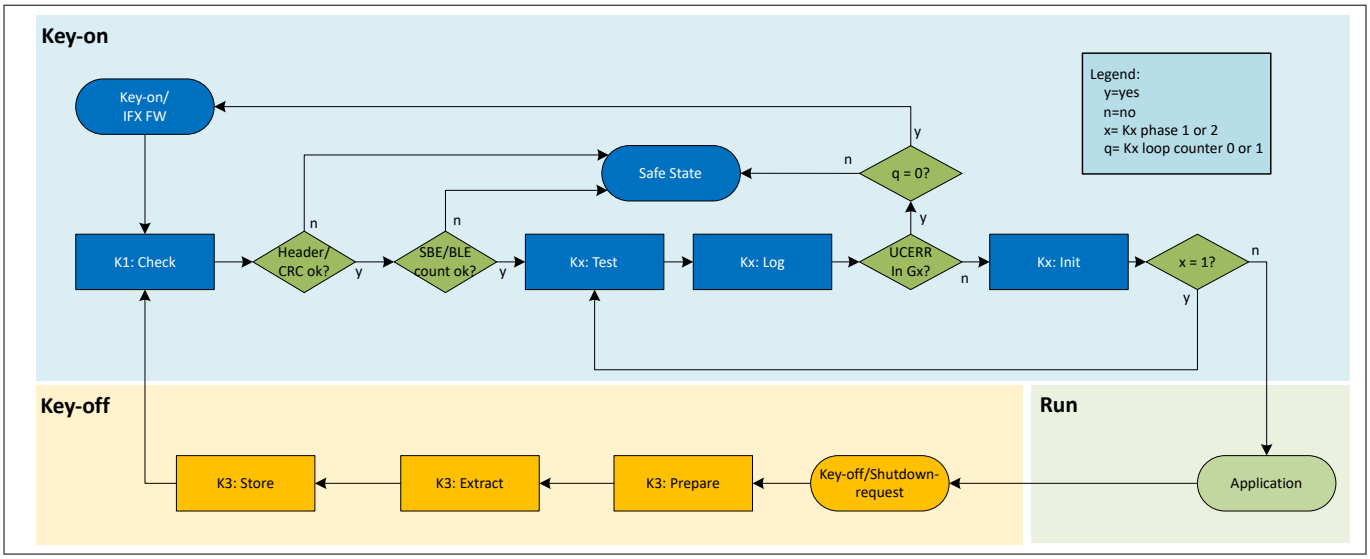


Figure 763 MBIST Cycle

The following sections explain with more detail the different test phases shown in the MBIST Cycle.

K1: Check

The main purpose of the first phase of key-on MBIST is evaluation of result of previous Key-off phase, in order to determine health status of the memory stack. This can be done as follows:

1. Read in the FAILID structure from its storage location in non-volatile memory and check CRC and Header contents:
   a. In case of a SAL or CRC mismatch, or V-flag is not equal to $57B5327F_H$ (that is, no valid information on HE is available), then a transition to safe-state shall be triggered

      ***Note**: Only in case both redundant structures (ORIG and COPY) fail the previous step, it shall be regarded as fail*

      ***Note**: An alternative to direct safe-state transition is immediate re-generation of FAILID within the Key-off cycle. The necessity of such a measure has to be decided in context of ECU architecture and application*

2. Add together the #SBE of each MCi to get the overall number of SBE in the MCU. Do the same for the #BLE entries
3. In case the added overall MCU SBE > M_{SBE} or BLE>M_{BLE} datasheet parameters, trigger transition to safe-state. Otherwise proceed to Kx: Test

***Note**: Steps for preparation of initial FAILID are described as part of K3: Store*

Kx: Test

For K1 the memory group G1 covers the priority group as required for setup of stack/CSA, Cache-enabling, and Secure Boot.

For K2 the memory group G2 covers the remainder of the MCU memories. The z and i indexes of VMTz and MCi respectively shall be taken from the SSH instances table in the device specific section of this chapter.

For all i in Gx:

***Note**: The following steps are performed at the clock frequency f_{BACK} in order to achieve maximum test-parallelism within DS-specified load-jump constraints. Therefore, the single steps below can be applied for all the memories within a memory group Gx at the same time.*

1. Set the corresponding bit-fields of the MEMTEST(CS) [i] = 1_B and check for MEMSTAT(CS) [i] == 0_B

   ***Note**: It is recommended to set testmode for all memories targeted by test in a single write operation to the respective MEMTEST(CS). Rationale: In case Auto-Data-Init/Partial-Erase has been activated (UCB0_17/18, namely, UCB_RTC_USERCFG) such proceeding will limit wait.*

   ***Note**: In order to avoid generation of spurious alarms (including forwarding to and capturing in SMU), it is recommended to disable their generation through ALMSRCS == 7_H. Alarm disabling will result in generation of miscellaneous alarm, which shall be cleared after completion of MBIST sequence*

2. Program March {w1,r1,w0,r0}: CONFIG0 = $400A_H$, CONFIG1 = 0003_H
3. Start Test by MCONTROL = 0001_H
4. Check for MSTATUS.DONE == 0_B, then set MCONTROL = 0000_H

   ***Note**: Former check can be replaced by short wait time (few cycles) before clearing of MCONTROL is performed*

5. Check for test being completed using MSTATUS.DONE == 1_B

   ***Note**: The test completion time is documented under the key-on tMBIST parameter in the datasheet.*

Kx: Log

1. Evaluate result using ECCD:
   - **a.** If UCERR==1_B, check counter q:
     1. q==0_B --> Increment q and trigger additional cycle (typically with reset) to exclude potential effect of Soft-Errors
     2. q==1_B --> Trigger transition to Safe State
   - **b.** If CERR = 1_B, store the VMTz and MCi indexes in the FAILID data structure
   - **c.** Store the FAILID data structure in a memory location for later access by the K3 test phase and assure its non-modification during application run-time (for example, protected RAM-Range)

     ***Note**: The Header, V-flag and CRC of the FAILID data structure are not mandatory at this stage.*

Kx: Init

1. For memory groups G1 and G2_1:
   - **a.** Set MCONTROL = 8001_H (bit-fields SRAM_CLR and START) and check for MSTATUS.DONE == 0_B
   - **b.** Then clear the START bit by setting the setting MCONTROL = 8000_H and check for test being completed with MSTATUS.DONE == 1_B
2. For ALL memories: MEMTEST(CS) [i] = 0_B and check for MEMSTAT(CS) [i] == 0_B

   ***Note**: Only after this step, functional access to memories-under-test is available. Alarm generation shall be re-enabled through ALMSRCS == 27_H*

3. For G2_2: Initialization shall be performed through the TriCore™ CPU instruction ST.D (Store Double-word) to avoid potential traps due to ECC errors. Alternatively, the RAM can be initialized with code or data copied from another location (for example with DMA)

   ***Note**: This step can also be performed at later time-point, for example, before PPU starts execution or before COM-Stack is ramped for GETH*

K3: Prepare

1. Read in the FAILID data structure in order to identify the memories which require further testing. In case the working RAM (WR), that is the RAM that holds the stack, variables, temporary data, and so on, is in the list, then principally two possibilities exist:
   a. Execute K3 on an alternate CPU different from the core which is associated with WR (here the corresponding alternate WR is assumed to be fault-free). This implies that the respective code needs to be executable on different CPUs (position-independent)
   b. Execute K3 on all memories but WR. Then migrate the respective data to different memory within the same CPU (for example, from DSPR to PSPR or DLMU) and test WR

K3: Extract

The purpose of this test phase is to perform a detailed analysis of non-critical single bit errors present in a RAM instance. The previous Kx: Test phase marks RAM instances containing at least one single bit error without any further details.

This detailed analysis allows to:

- Count the number of individual single bit errors
- Detect bitline-oriented errors

This is done by scanning through the whole memory range and collecting the location of each error found. If more than a certain number of error bits are located along the same bitline, then this is considered a bitline-oriented error. Otherwise they can be considered as single isolated errors. In order to detect bitline-oriented errors, details of the RAM architecture are required.

The figure shows an example of how the RAM architecture looks like. Each tower contains multiple bit columns and each bit column contains multiple bitlines. The multiplexer factor is the number of bitlines per bit column, however this does not play a role in this step. This is because there is a 1:1 mapping between each bitline inside the RAM and each individual bit in the MCi_RDBFLy registers, as shown in the figure, regardless of multiplexer factor. So for example, if the number of bitlines per tower (tower width) is equal to 32 and there are a total of 2 towers, 2x32=64 total bits of the MCi_RDBFLy registers are used to capture errors found during the test. Since each MCi_RDBFLy has a width of 16 bits, four of these registers are used in this example.

***Note**: The number of RDBFLys, number of towers per SSH, tower width can be found in the device specific part of the user manual.*

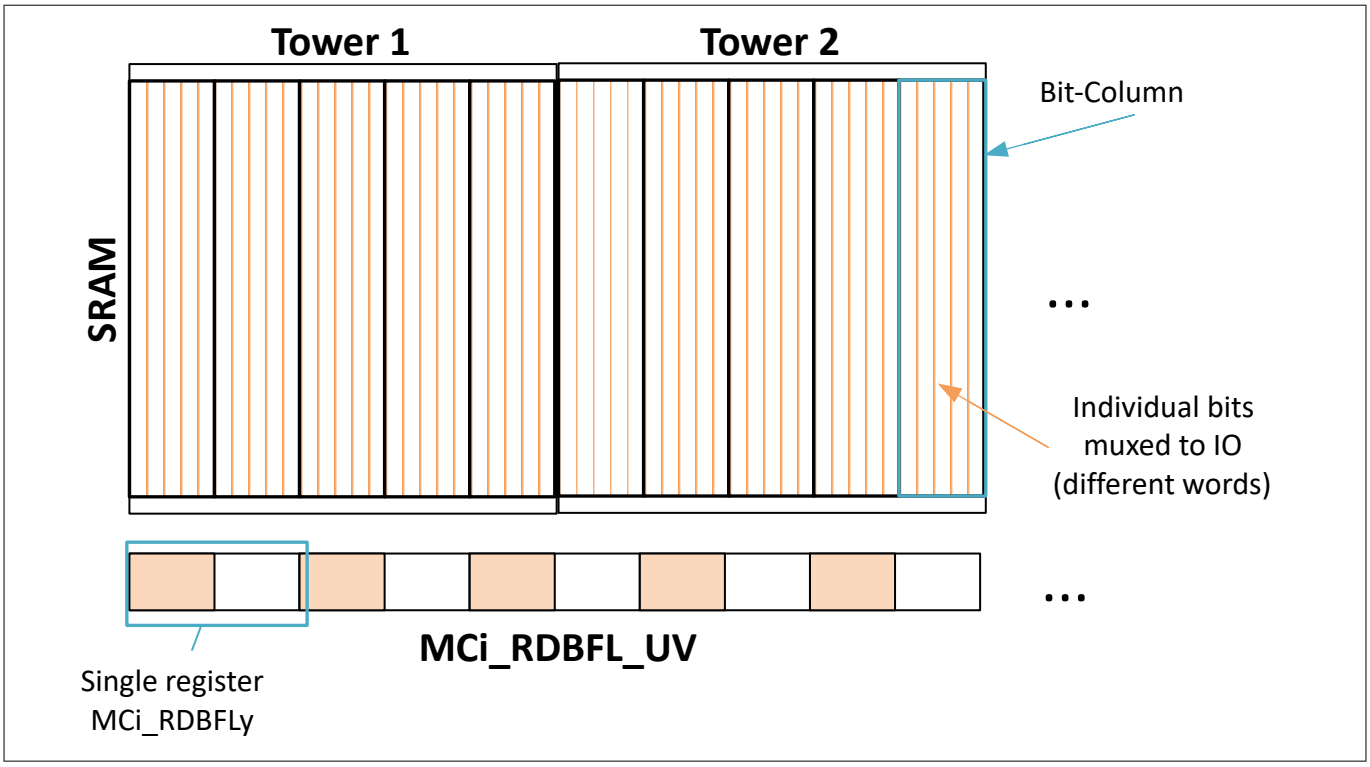


Figure 764 SRAM microarchitecture

Exemplary SRAM microarchitecture: Two macros 'Towers' including associated RDBFLy registers are shown, where RDBFLy_UV is the concatenation done by software of all RDBFLy. Upon execution of reads during MBIST execution, the XOR pattern between individual bits (stored in a tower) and expectation according to used March element are written to the RDBFLy registers. The number of register bits used by the MBIST test is determined by the number of towers and the tower width. The total number of available RDBFLy registers (and bits) could however be greater than that.

***Note**: The following steps shall be executed sequentially for each memory within FAILID. No parallel execution of memories in this step.*

For each memory instance in the FAILID data structure, a fail bitmap and an evaluation of the respective failure type needs to be established. For this purpose, the FAILID table loaded in the previous step is extended with the three green columns shown in the FAILID reference structure, containing the error address, its bitline position and the identified error type (Error-Address, FailPosition, and FailMarker respectively). The extraction is performed considering each RAM instance's microarchitecture and the following associated parameters: Tower Width (tw) and Tower Count (tc).

Steps to be performed are:

1. Set MEMTEST[i] = 1_B and wait for MEMSTAT[i] == 0_B

***Note**: In order to avoid generation of spurious alarms (including forwarding to and capturing in SMU), it is recommended to disable their generation through MCi_ALMSRCS == 7_H. Alarm disabling will result in generation of miscellaneous alarm, which shall be cleared after completion of MBIST sequence*

2. Program March {w1,r1,w0,r0}: CONFIG0 = $400A_H$, CONFIG1 = 0003_H

3. Enable MCONTROL.FAILDMP and start test by MCONTROL = 0201_H

***Note**: The test completion time is documented under the key-off tMBIST parameter in the datasheet.*

4. Check for MSTATUS.DONE == 0_B, then set MCONTROL = 0200_H and initialize the failbit counter FBC to zero
5. While MSTATUS.DONE is not equal to MSTATUS1_B, poll MSTATUS.FDA:
   a. If MSTATUS.FDA==1_B --> Bitmap data available:
      1. Increment failbit count FBC and temporarily store ETRR and *all* RDBFLy (associated with the SSH-under-test) to WR
      2. Concatenate all the RDBFLy (provides an 'unrolled view' RDBFLy_UV)
      3. Then the individual bits of RDBFLy_UV are allocated to the individual towers, through consideration of tower width as index-increment, that is, all bits with index [(t+1)*tw-1:t*tw] are to be evaluated to collect the fail position in tower t, with t=0...tc, found in the SSH instances subchapter of the device specific section of this functional block
      4. Consequently, the position of 1_B in this range is stored 8 bit-encoded to fail position[8(t+1)-1,8t] in FAILID
      5. This provides the SSH-internal error-address, as well as the tower-granular (muxed) failbit-positions. The latter are then marked by '1' (XOR with expected pattern) in the entries of the assembled SSHs fail-table
      6. If FBC ≤ M_{SBE}, then resume the test by setting MCONTROL = 0202_H. Otherwise run the MBIST to the end with FAILDMP disabled by setting MCONTROL = 0002_H
   b. Once MSTATUS.DONE == 1_B --> Evaluate failure-type from information stored to WR in previous step:
      1. Soft CE: No fail information available, that is, the memory has been mapped to FAILID due to a transient event during Kx: Test. Probability is very low to observe such case
      2. Bitline-oriented errors: If three or more entries showing the same bit-position in the same tower, the respective memory instance contains a bitline-oriented error, which shall be counted as such in FAILID (respective entry in FailMarker[2(t+1)-1,2t] is set to *2*)
      3. Hard individual CEs: If less than three entries in the SSHs fail-table (that is, entries with different ETRRs) have the same failbit-position in RDBFLy (allocated to same tower), all entries in the table shall be counted as individual errors (respective entry in FailMarker[2(t+1)-1,2t] is set to 1_B)
   c. Clear MEMTEST(CS) [i] = 0_B and wait for MEMSTAT(CS) [i] == 0_B

***Note**: Alarm generation shall be reenabled through ALMSRCS == 27_H*

K3: Store

This phase essentially focuses on removing temporary data, comparing with results from previous K3 executions stored in non-volatile memory, and finally updating FAILID in non-volatile memory if required; it is to be seen as prerequisite for next execution of K1: Check. Following steps are to be performed:

1. Number of bitline-oriented errors are determined by evaluating number of different Error-Address entries containing 10_B in FailMarker, that is, all entries marked with 10_B are associated with a singular BL-oriented fault
2. Number of SBE are determined by number of Error-Address entries with FailMarker equal to 01_B
3. The counted numbers are finally stored in the first row associated with the SSH under consideration as shown by the yellow color coding in the FAILID reference structure. All other rows are removed from the table
4. Consequently, the SBE and BLE counts are compared against the FAILID version currently stored in non-volatile memory, in order to check whether update of the latter is required (that is, table update shall be only performed upon increase of SBE/BLE). If this is case, the following steps shall be performed:
   a. Invalidate or erase (both resulting in invalid V-flag) FAILID_ORIG, recalculate CRC and store the updated fail information array together with V-flag to targeted SAL (during this step FAILID_COPY is assumed to be valid containing data from previous K3 cycle)
   b. In case update of FAILID_ORIG has been successfully completed, FAILID_COPY can be kept in its current state, considering that K1: Check is evaluating the first valid data structure, starting with

ORIG. COPY shall then be used for next required update (while invalidating V-flag of ORIG after completion of operation on COPY)

***Note**: The device shall be initially programmed with a valid FAILID information containing valid Header and CRC (in case duplication ORIG/COPY is used, pre-programming of one is sufficient).*

34.3.6.3 NDT for DLMU

There are applications that during the startup phase generate non-ASIL data before the execution of MBIST. This data is needed for later stages of the startup procedure and for higher layers of the software stack. For this purpose a mailbox (MB) RAM can be designated as storage location for this data. However, the destructive MBIST cannot be applied on the MB, as it would wipe out the previously stored data.

In case DLMU is used for mailbox purpose and is therefore excluded from destructive MBIST test sequence described above, a non-destructive test (NDT) can be executed on this memory. The latter (*MB*) shall be excluded from memory groups G1 and G2, and tested using the configuration described below. This can be done either as integral part of Kx: Test or also at a later point in time (for example, after Kx: Init, in order to save the mandatory initialization step before running the actual test).

***Note**: In this context, the MB RAM is used for non-ASIL data only. In case that storage of ASIL data in MB RAM is required, then additional measures against data corruption shall be applied, for example, appended CRC checksum.*

The NDT algorithm reads a word from the SRAM (data + ECC bits), inverts it and writes it back. If the ECC does not match the expected value when reading, an error is assumed and notified. The target pattern to be executed here is {r, w*, r*, w, r}, where r stands for read and w for write. An (*) symbol in this notation always indicates that the data during that access is inverted with respect to the original content in the RAM.

Before using the MB RAM, it must be initialized completely either by software or by using the SSH functionalities. This is necessary to ensure that all contents of the RAM are ECC coherent. Refer to the chapter "Initializing SRAMs" for instructions on how to initialize a RAM using the SSH functionality.

After this initialization step, the MB RAM is ready for use. At a later point in time, for example as part of Kx: Test, the actual NDT can be executed.

Testing steps:

1. Set MEMTEST(CS)[i] = 1_B and wait for MEMSTAT(CS)[i] == 0_B

   ***Note**: In order to avoid generation of spurious alarms (including forwarding to and capturing in SMU), it is recommended to disable their generation through ALMSRCS == 7_H. Alarm disabling will result in generation of misc. alarm, which shall be cleared after completion of MBIST sequence.*

2. Set CONFIG0 = 5015_H and CONFIG1= 5018_H
3. Set MCONTROL = 1_H, and check for MSTATUS.DONE == 0_B
4. Then set MCONTROL = 0_H and check for initialization being completed through MSTATUS.DONE == 1_B

Closure step:

1. Evaluate result using ECCD:
   a. If UCERR==1_B: Trigger transition to Safe State
   b. Otherwise proceed with disabling of MB's SSH: MEMTEST(CS)[i] = 0_B and check for MEMSTAT(CS)[i] == 0_B

      ***Note**: Alarm generation shall be reenabled through ALMSRCS == 27_H*

Related information

Initializing SRAMs on page 5742

34.3.7 SRAM addressing

When considering SRAM addressing, different levels of addressing need to be considered. At the highest level, all SRAMs which appear on the overall system memory map could be potentially accessed via a portion of the system address space. This is the logical address space to access the SRAM as far as the system is concerned.

However, from the SSH point of view, this system level logical address cannot be seen, since the SSH is tightly coupled to the SRAM, and the system address translation or mapping occurs at a higher level before arriving at the SSH. Therefore, unless otherwise specifically mentioned, any “address” in this chapter does not correspond to any of the system address that is mentioned in the memory map chapter.

Within the SSH, the logical addresses increment linearly from 00_H until a maximum address depending on the size of the SRAM.

During normal system operation, the incoming address is directly input to the SRAM. This address at the input of the SRAM is also stored in ETRR register in case of any error. In order to access the error address location stored in the ETRR during normal functional mode, the same address can be provided to the RANGE.ADDR, with RANGE.RAEN = 0.

***Note**: For actual individual SSH reset values of RANGE, refer to the SSH instances table.*

Related information

SSH instances on page 5825

34.3.8 Safety measures

Always refer to the device specific safety manual to find out which safety mechanisms of this functional block need to be considered for safety relevant applications.

The following chapter aims to provide a generic description of the safety measures.

There are a number of safety features that help to detect random hardware faults in the SRAM cells as well as periphery.

Random hardware faults in the data (data path as well as SRAM cell), address logic within the SRAM, and control logic within the SRAM are detected by safety mechanisms.

Combined safety notifications (alarms) are provided to the SMU separately for security and non-security related SSHs.

This section provides a short summary of these safety mechanisms and safety notifications. Additionally, some of these features may be explained in more detail in other sections.

34.3.8.1 Safety features

The safety features implemented in the SRAMs, SSH instances, and the VMTx are the following:

- Error detection and correction codes, logic and ECC monitoring
- Address error monitor
- SRAM multiplexer factor
- Error tracking registers
- Safety flip-flop implementation for critical register bits

34.3.8.1.1 SRAM error detection and correction (EDC/ECC)

EDC (Error detecting code) or ECC (Error correcting code) is available for the stored data of selected SRAMs. Refer to the subchapter "SSH instances" in the device specific configuration chapter of this manual to find out which SRAMs feature EDC or ECC.

During a WRITE operation, checksum bits (ECC bits) for the input data word are generated using an ECC encoder. These bits are stored in the SRAM along with and additionally to the data word itself.

During a READ operation, an ECC decoder compares the read data word and ECC bits to the expected ECC value. If due to any random hardware failure or soft error for example, the data word stored in the SRAM was corrupted, then there will be a mismatch compared to the expected ECC value. This is notified as an alarm and the corresponding bits in the error flag register are set.

Two different kinds of ECC codes are used in the system for different SRAMs: SECDED codes and DED codes.

- SECDED codes can detect: a double bit error (DBE), as well as correct a single bit error (SBE). That is, the Hamming distance of these codes is four
- DED codes can detect up to a double bit error, but cannot correct any error

The column “ECC Type” in the tables under the subchapter "SSH instances" in the device specific configuration chapter of this manual show which type of ECC code is used for each SRAM.

There are two error notifications sent to the SMU from the ECC decoder through the CE and UCE alarms. For SRAMs with SECDED ECC these are correctable error event (that is, SBE detected and corrected) alarm and uncorrectable error event (that is, DBE detected) alarm. For SRAMs with DED ECC, there is only one error notification and this is notified by the DBE error and UCE alarm.

SBE notification (depending on ECCS.ECE, forwarded as CE alarm or UCE alarm) is enabled by setting ALMSRCS.SBE = 1.

DBE notification (forwarded as UCE alarm) as well as tracking in the ETRR/ERRINFO is enabled by setting ALMSRCS.DBE = 1.

***Note**: The ECCS.CENE, UCENE, and TRE bits act as a further global enable and have to be set for the ECC alarm notifications to be sent to SMU as well as to enable error tracking in the ETRR/ERRINFO. Please refer to the chapter "Safety notifications" for more details.*

The number of ECC words in a SRAM tower is defined by the ECC granularity, described in the SSH instances table in the device specific configuration chapter.

For example: An ECC granularity of two implies that there are two ECC words per physical SRAM block.

In the product family most SSHs with SECDED capability are equipped with a monitor to allow for fault detection within the ECC encoders and decoders. A detected fault is forwarded as UCE alarm (FAULTSTS.OPERR[4] and FAULTSTS.OPERR[5]).

Some SSHs are not equipped with a monitor for ECC encoders and decoders. Refer to the section "SSH functionality exceptions" in the device specific configuration chapter for more information.

Related information

Error mapping on page 5760
Safety notifications on page 5757
SSH functionality exceptions on page 5835
SSH instances on page 5825

34.3.8.1.2 Address error monitor

Similar to how the ECC helps to monitor errors in the data, the address error monitoring mechanism helps to detect failures in the address generation logic within the SRAMs.

An error in the address generation within the SRAM can lead to:

- A valid input address selecting no wordline/bitline
- A valid input address selecting multiple wordlines/bitlines
- A valid input address selecting a wrong wordline/bitline
- A valid input address selecting a non-existing (for example out of range) wordline/bitline

All the above SRAM failure modes are detected by the address error monitor.

Whenever an error is detected by the address error monitor, an alarm is sent to the SMU.

The address error detection, tracking and notification is enabled by setting ALMSRCS.ADDRE, and the alarm is forwarded as UCE alarm.

If the address error detection is enabled (ALMSRCS.ADDRE = 1) and an error is detected, then it is tracked in the ETRR and ERRINFO registers (ERRINFO.ADDRERR is set). Please note that the ECCS.UCENE and TRE bits act as a further global enable and have to be set for the address error alarm notifications to be sent to SMU as well as to enable error tracking in the ETRR/ERRINFO.

34.3.8.1.3 SRAM multiplexer factor

In order to reduce the probability of a single random hardware fault (for example, an alpha particle strike or contact fail) causing more than a single-bit error, the bits in each logical data word are interleaved by a certain factor (4, 8, or 16). This means, two consecutive bits in a logical data word are actually physically 4, 8, or 16 bits apart. The multiplexer factor used in each SRAM is mentioned in the SSH instances table.

34.3.8.1.4 Error tracking registers

The error tracking registers store the address (ETRR) and the corresponding type (ERRINFO) of errors in the SRAM.

Error tracking is globally enabled by setting ECCS.TRE. Tracking of individual error types (DBE, ADDRE) can be enabled or disabled with the corresponding bits in the ALMSRCS register.

***Note**: The tracking of single bit errors is not implemented.*

Error tracking

It is possible for the software to track the error type and the address in which it occurred. The ECCS.TRE enables the tracking of the address and type of a detected error.

***Note**: The address corresponds to the local SRAM address. Refer to the chapter "SRAM addressing" for more details.*

***Note**: The tracking capabilities are reduced for some SSHs. Refer to the device specific configuration chapter for more information.*

Whenever address tracking is enabled, the ETRR contains the physical address of the first detected error. The ECCD.VAL is a bit field associated with the ETRR register. When the ETRR register contains a valid error address, the ECCD.VAL bit is also set. The ERRINFO register contains the error type (uncorrectable or address error) at the address in the ETRR register entry.

Software can clear the ECCD.VAL and the ETRR and ERRINFO registers by setting the tracking clear bit, ECCD.TRC. Only the first error after register clear is tracked. Errors happening afterwards do not overwrite the content of the ETRR and ERRINFO registers.

Related information

SRAM addressing on page 5754

34.3.8.1.5 Safety flip-flops

Certain critical registers have to be protected against soft errors changing their values. Some flip-flops are implemented in the VMTx itself while others are part of each SSH instance.

The safety flip-flops (SFF) in the VMTx report errors directly to the SMU. The SFFs in the SSH report errors through the UCE alarm and the status through FAULTSTS.OPERR[2].

Self testing of the SSH SFFs is triggered through the SFFD bit. This bit is cleared when the test is complete. The FAIL status from the test is notified through FAULTSTS.MISCERR[0] - but this does not trigger an ME alarm.

34.3.8.2 Safety notifications

The error notifications from all security and non-security related SRAMs/SSHs in the system can be reported to the Safety Management Unit (SMU). The SMU may be programmed to initiate appropriate action.

As shown in the diagram below, each SSH provides the following alarms to the VMTx:

- Correctable Error (CE) Alarm - Status through ECCD.CERR bit
- Uncorrectable Error (UCE) Alarm - Status through ECCD.UCERR bit
- Miscellaneous Error (ME) Alarm - Status through ECCD.MERR bit

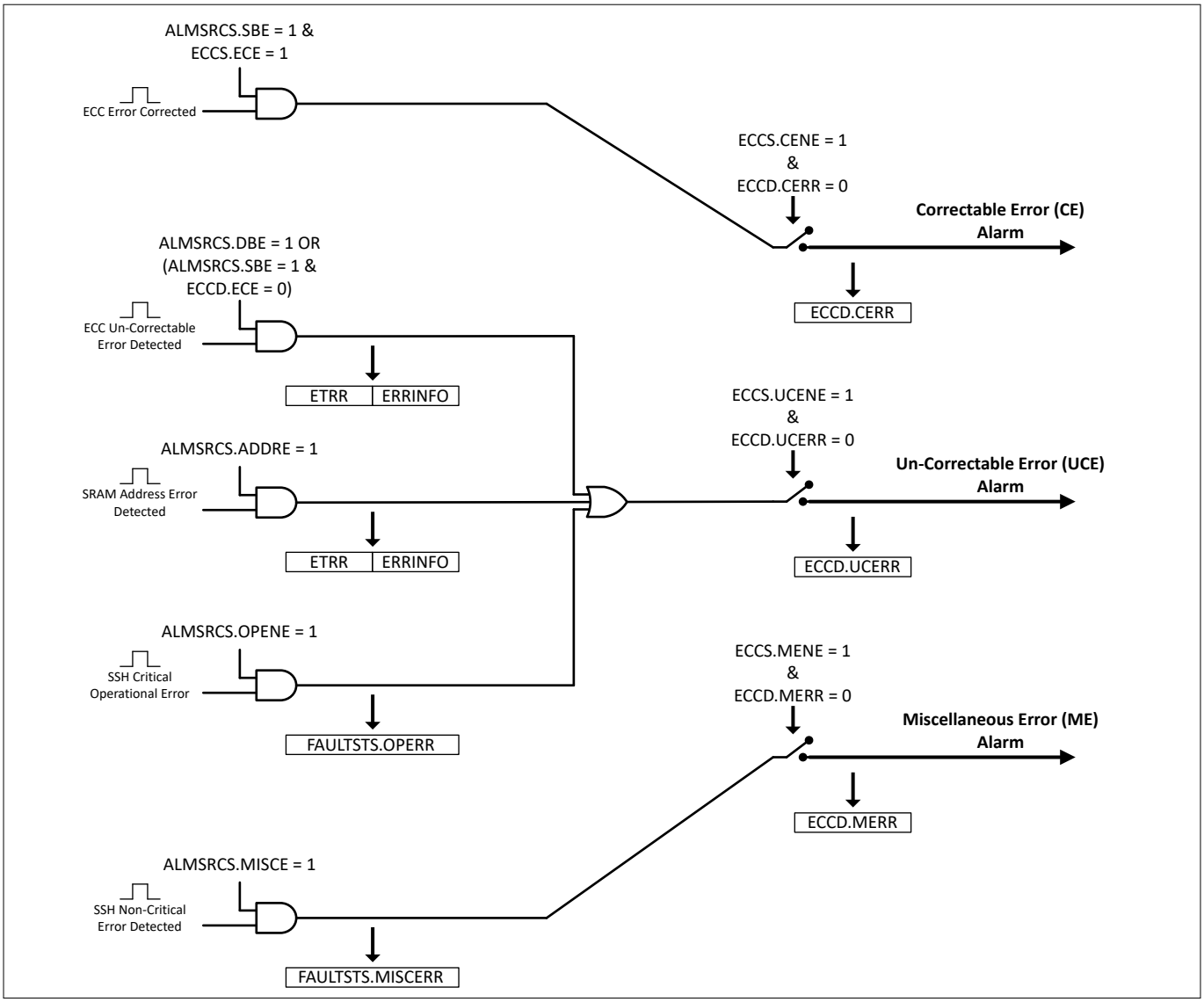


Figure 765 Alarms and error detection enabling (ECCS.TRE assumed to be set)

Alarms of the same type (CE, UCE and ME) originating from all SSH instances are merged together with an OR-gate in the VMTx (separately for security and non-security related SSHs) and then forwarded to the SMU as a single alarm. Software can check CEALARM(CS), UCEALARM(CS) and MEALARM(CS) registers to check which SSH caused the alarm.

When an alarm occurs, the corresponding *ERR bit in the ECCD register is set. No new alarms of the same type are forwarded as long as this bit remains set. Therefore, once an alarm occurs, the software shall clear the corresponding *ERR bit in order to enable forwarding of further alarms of the same type.

Normally, there is a time window of multiple clock cycles from when an alarm event occurs (that is, ECCD.*ERR is set) until the software reacts and clears the flag to enable forwarding of additional alarms. During this time window, additional new errors corresponding to the same alarm type may occur. For any such new event, no new alarm is generated.

Even if the new hardware event occurs exactly at the same cycle as the software clear event takes effect - the ECCD bits are cleared and a new alarm is not generated.

Table 1373 SSH safety notifications summary

Alarm	Errors or events mapped	Enable	Software handling/clearing
CE	- ECC Correction Event Occurred	ECCS.CENE = 1 and ECCS.ECE = 1 and ALMSRCS.SBE = 1	ECCD.CERR has to be cleared to forward further CE alarms
UCE	-ECC Uncorrectable Event Detected -Address Error Detected -Error Critical for SSH Operation Detected (including Critical SFF Errors)	ECCS.UCENE = 1 and (ALMSRCS.DBE = 1 or (ECCS.ECE = 0 and ALMSRCS.SBE = 1) or ALMSRCS.ADDRE = 1 or ALMSRCS.OPENE = 1)	ECCD.UCERR has to be cleared to forward further UCE alarms. The errors and status are always tracked in the ETRR, ERRINFO, and FAULTSTS registers independent of the value of UCERR
ME	- Less Critical SFF Errors (Enable Bits in ECCS, ALMSRCS) - Disabling of alarm notifications - Disabling of safety mechanisms	ECCS.MENE = 1 and ALMSRCS.MISCE = 1	ECCD.MERR has to be cleared to forward further ME alarms

34.3.8.2.1 Alarm handling

There are three alarms from each SSH and different events/errors are mapped to each alarm. Each alarm is enabled by the corresponding enable bit in the ECCS register (that is, ECCS.CENE, UCENE, and MENE).

When an alarm occurs, the corresponding status flag in the ECCD register is set (that is, ECCD.CERR; UCERR, and MERR).

For the events mapped to each alarm type, there are separate enable bits in the ALMSRCS register (all enabled by default).

For DBE error event and address faults (ADDRE) - the status of the error can be obtained from the ETRR and ERRINFO registers (including ECCD.VALID bits).

For safety reasons, some operational faults are detected which affect the normal operation or access to the SRAM (for example, SSH gets accidentally enabled, auto-data-init gets triggered). These are also notified through the UCE alarm, and can be enabled from the ALMSRCS register OPENE bit (Operational Error Notification Enable). Refer to the tables “Mapping of errors to FAULTSTS.OPERR and UCE alarm” and “Mapping of errors to FAULTSTS.MISCERR and ME alarm” in the "Error mapping" chapter of this manual for detailed information on error mapping.

Attention: The VMTx clock has to be enabled via the CLC register of the VMTx to forward any SSH alarm.

Certain faults do not directly impact the usage of the SRAM, but if left latent, could potentially have a serious effect. For example, the ECC or address error notification enable bits or the alarm enable bits get flipped due to a soft error. Such error notifications are enabled by the ALMSRCS.MISCE bit.

Diagnostics

When an alarm is raised by a SSH, the software has to check multiple registers to diagnose the cause of the fault.

ECCD.CERR, UCERR, and MERR bits give the overall status of which alarm occurred.

ETRR and ERRINFO registers store the address and the fault type of a fault in the SRAM (that is, DBE or ADDR error). FAULTSTS register bits store the status of other faults in the SSH itself which are mapped to UCE or ME alarm.

All the error status bits (ETRR, ERRINFO, FAULTSTS, and ECCD.VAL) are preserved until a power-on reset. The alarm status (ECCD.CERR, UCERR, MERR, and SERR) are cleared on an application reset.

Software can clear the bits by writing to them (when specified) or by setting the ECCD.TRC bit as appropriate.

***Note**: The VMTx needs three fSPB/2 clock cycles (6 fSPB cycles) to process SSH alarms.*

Error mapping

Both the UCE and the ME alarms have multiple events mapped to them. These events are separated in the design in such a way that the underlying hardware logic that can potentially cause the fault is covered as much as possible.

The OPERR bit field in the FAULTSTS register stores the status of various errors critical to SSH or SRAM operation.

Table 1374 Mapping of errors to FAULTSTS.OPERR and UCE alarm

Error status bit	Error description during normal application run	Hints for software reaction for recovery
FAULTSTS.OPERR[0]	SSH enabled. Functional access to SRAM is disabled	Perform an application reset
FAULTSTS.OPERR[1]	Auto-data-init or partial erase (caches) was triggered, resulting in overwriting of SRAM data	Perform an application reset
FAULTSTS.OPERR[2]	Safety flip-flop error detected in one of the registers. After a self test triggered by ECCS.SFFD, this error has to be taken into account together with FAULTSTS.MISCERR[0]	Perform a system reset. - Initialize the SRAM and SSH registers - Trigger a safety flip-flop self test via ECCS.SFFD. - Check that the test does not report a fail (FAULTSTS.MISCERR[0]). If the error still persists and cannot be cleared, a warm PORST has to be issued before performing the initialization and safety flip-flop self test
FAULTSTS.OPERR[3]	MBIST FSM got triggered, test multiplexers got enabled in the data path, or some other random hardware failure caused data in the SRAM to be overwritten or corrupted by a part of SSH logic	Perform an application reset
FAULTSTS.OPERR[4]	A mismatch occurred between one of the outputs of the ECC encoder and the ECC monitoring encoder	Perform an application reset
FAULTSTS.OPERR[5]	A mismatch occurred between one of the outputs of the ECC decoder and the ECC monitoring decoder	Perform an application reset

Please note:

- For FSI, a system reset is required
- For SCR RAMs, a cold PORST is required

The MISCERR bit filed in the FAULTSTS register stores the status of various errors which may not be directly critical for SSH or SRAM operation, but if left latent, may result in a critical failure in the future. One example is unintended deactivation of the ECC due to a bit-flip.

Table 1375 Mapping of errors to FAULTSTS.MISCERR and ME alarm

Error status bit	Error description during normal application run	Proposed software reaction for recovery
FAULTSTS.MISCERR[0]	This bit can be set only after the software has triggered a safety flip-flop self test via the ECCS.SFFD bit. If the test detected an error and the test failed, this bit is set. The ME alarm is not triggered in this case	Perform a system reset. - Initialize the SRAM and SSH registers - Trigger a safety flip-flop self test via ECCS.SFFD. - Check that the test does not report a fail (FAULTSTS.MISCERR[0]). If the error still persists and cannot be cleared, a warm PORST has to be issued before performing the initialization and safety flip-flop self test
FAULTSTS.MISCERR[1]	Any of the alarm notifications of UCENE or CENE got disabled (that is, one or more of the bits CENE or UCENE in the ECCS register went from 1 to 0) ***Note:*** *ECCS.MENE is protected by a safety flip-flop.*	Re-initialize SSH registers (ECCS)
FAULTSTS.MISCERR[2]	Any of the safety mechanisms got disabled (that is, ECE, TRE, SBE, DBE, ADDRE, OPENE) was set from 1 to 0 ***Note:*** *ALMSRCS.MISCE is protected by a safety flip-flop.*	Re-initialize SSH registers (ECCD, ALMSRCS)
FAULTSTS.MISCERR[3]	Access to a non-existing register in an existing SSH instance	Provide a valid address (that is, the address of a register that does exist in an SSH instance)
FAULTSTS.MISCERR[4]	An ongoing VMTx-SSH communication was aborted due to a module reset of the IP where the relevant SSH is embedded	

Error injection and alarm triggering

To ensure proper behavior of safety related software, it may be required to inject errors and trigger different safety notifications.

In order to trigger ECC errors (status denoted by ERRINFO.DBERR), the ECCMAP bits may be used. When ECCMAP is not equal to 0, then either data or ECC bits alone can be accessed. Using this feature, single or double bit soft errors may be introduced in the SRAM.

***Note**: When using ECCMAP to inject errors for Non-Destructive MBIST Test (NDT), the errors shall be injected before the NDT. ECCMAP shall set back to zero before starting and during the NDT.*

The address errors (status denoted by ERRINFO.ADDRERR) may be triggered by setting ECCS.SFLE, which can force an error into the address by flipping one address bit.

The CE and UCE alarms may be triggered by injecting an ECC or address error.

In order to trigger an ME alarm for test purposes, any of the safety notifications can be simply disabled. This will trigger an ME alarm.

34.4 Registers

34.4.1 Register overview - access mode glossary

Table 1376 Register overview - access mode glossary

Keyword	Description
CSP	Access protection using PROT register PROTCSP.
E	Access protection using PROT register PROTE.
SE	Access protection using PROT register PROTSE.
APU-PCS	Protection group consisting of registers ACCENCS_WRA, ACCENCS_WRB, ACCENCS_RDA, ACCENCS_RDB, ACCENCS_VM, ACCENCS_PRS.
PCS	Access protection using *APU-PCS* registers.
APU-PNOM	Protection group consisting of registers ACCENNOM_WRA, ACCENNOM_WRB, ACCENNOM_RDA, ACCENNOM_RDB, ACCENNOM_VM, ACCENNOM_PRS.
PNOM	Access protection using *APU-PNOM* registers.
BE	Always returns a Bus Error.
U	No access restrictions.
PROT	Access restrictions as defined in the PROT register access rules.
STP	Access only during start-up, before start-up protection is set (SMM_STCON.ST_PROT=1).
32	Access only when using 32-bit width.

34.4.2 Registers overview - VMT (ascending offset address)

Table 1377 Registers overview - VMT (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CLC	Clock control register	00000_H	PNOM	PNOM, SV, E	See 5766	5766
ID	Module identification register	00008_H	PNOM	BE	PowerOn Reset	5767
PROTE	PROT register endinit	00020_H	U	SV, PROT	Application Reset	5768
PROTSE	PROT register safe endinit	00024_H	U	SV, PROT	Application Reset	5769
PROTCSP	PROT register cyber-secure APU endinit	$0002C_H$	U	SV, PROT	Application Reset	5771
ACCENNOM_WRA	ACCENNOM write access enable register A	00040_H	U	SE, SV	Application Reset	5773
ACCENNOM_WRB	ACCENNOM write access enable register B	00044_H	U	SE, SV	Application Reset	5773
ACCENNOM_RDA	ACCENNOM read access enable register A	00048_H	U	SE, SV	Application Reset	5774

(table continues...)

Table 1377 (continued) Registers overview - VMT (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ACCENNOM_RDB	ACCENNOM read access enable register B	$0004C_H$	U	SE, SV	Application Reset	5774
ACCENNOM_VM	ACCENNOM VM access enable register	00050_H	U	SE, SV	Application Reset	5775
ACCENNOM_PRS	ACCENNOM PRS access enable register	00054_H	U	SE, SV	Application Reset	5776
ACCENCS_WRA	ACCENCS write access enable register A	00060_H	U	CSP, SV	Application Reset	5776
ACCENCS_WRB	ACCENCS write access enable register B	00064_H	U	CSP, SV	Application Reset	5777
ACCENCS_RDA	ACCENCS read access enable register A	00068_H	U	CSP, SV	Application Reset	5777
ACCENCS_RDB	ACCENCS read access enable register B	$0006C_H$	U	CSP, SV	Application Reset	5778
ACCENCS_VM	ACCENCS VM access enable register	00070_H	U	CSP, SV	Application Reset	5778
ACCENCS_PRS	ACCENCS PRS access enable register	00074_H	U	CSP, SV	Application Reset	5779
MEMTEST	Nominal memory MBIST enable register	00080_H	PNOM	SV, PNOM	Application Reset	5780
MEMMAP	Nominal memory mapping enable register	00084_H	PNOM	SV, PNOM	Application Reset	5780
MEMSTAT	Nominal memory status register	00088_H	PNOM	BE	Application Reset	5780
MEMDONE	Nominal memory test done status register	$0008C_H$	PNOM	BE	Application Reset	5781
MEMFDA	Nominal memory test FDA status register	00090_H	PNOM	BE	Application Reset	5782
CEALARM	Nominal SSH CE alarm register	$000A4_H$	PNOM	BE	Application Reset	5782
UCEALARM	Nominal SSH UCE alarm register	$000A8_H$	PNOM	BE	Application Reset	5783
MEALARM	Nominal SSH ME alarm register	$000AC_H$	PNOM	BE	Application Reset	5783
MEMTESTCS	Cyber-secure memory MBIST enable register	$000B4_H$	PCS	SV, PCS	Application Reset	5784
MEMMAPCS	Cyber-secure memory mapping enable register	$000B8_H$	PCS	SV, PCS	Application Reset	5784
MEMSTATCS	Cyber-secure memory status register	$000BC_H$	PCS	BE	Application Reset	5785
MEMDONECS	Cyber-secure memory test done status register	$000C0_H$	PCS	BE	Application Reset	5785

(table continues...)

Table 1377 (continued) Registers overview - VMT (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
MEMFDACS	Cyber-secure memory test FDA status register	$000C4_H$	PCS	BE	Application Reset	5786
CEALARMCS	Cyber-secure SSH CE alarm register	$000D8_H$	PCS	BE	Application Reset	5786
UCEALARMCS	Cyber-secure SSH UCE alarm register	$000DC_H$	PCS	BE	Application Reset	5787
MEALARMCS	Cyber-Secure SSH ME alarm register	$000E0_H$	PCS	BE	Application Reset	5788
MCi_CONFIG0	MC i Nominal configuration register 0	01000_H+i *200_H	PNOM, 32	PNOM, 32	Application Reset	5788
MCi_CONFIG1	MC i Nominal configuration register 1	01004_H+i *200_H	PNOM, 32	PNOM, 32	Application Reset	5789
MCi_MCONTROL	MC i Nominal MBIST control register	$0100C_H$+ i*200_H	32, PNOM	SV, PNOM, 32	Application Reset	5791
MCi_MSTATUS	MC i Nominal status register	01010_H+i *200_H	32, PNOM	BE	Application Reset	5793
MCi_RANGE	MC i Nominal range register, single address mode	01014_H+i *200_H	PNOM, 32	PNOM, 32	Application Reset	5795
MCi_REVID	MC i Nominal revision ID register	$0101C_H$+ i*200_H	32, PNOM	BE	Application Reset	5796
MCi_ECCS	MC i Nominal ECC safety register	01020_H+i *200_H	32, PNOM	SV, PNOM, 32	Application Reset	5797
MCi_ECCD	MC i Nominal Memory ECC detection register	01024_H+i *200_H	32, PNOM	SV, PNOM, 32	See 5799	5799
MCi_ETRR	MC i Nominal error tracking register	01028_H+i *200_H	32, PNOM	BE	PowerOn Reset	5801
MCi_RDBFLy	MC i Nominal read data and bit flip register y	01080_H+i *200_H+y *4	PNOM, 32	PNOM, 32	Application Reset	5801
MCi_ALMSRCS	MC i Nominal alarm sources configuration register	$011D0_H$+ i*200_H	32, PNOM	SV, PNOM, 32	See 5802	5802
MCi_FAULTSTS	MC i Nominal SSH safety faults status register	$011D4_H$+ i*200_H	32, PNOM	SV, PNOM, 32	PowerOn Reset	5803
MCi_ERRINFO	MC i Nominal error information register	$011D8_H$+ i*200_H	32, PNOM	BE	PowerOn Reset	5805
MCCSi_CONFIG0	MCCS i Cyber-secure configuration register 0	11000_H+i *200_H	PCS, 32	PCS, 32	Application Reset	5806
MCCSi_CONFIG1	MCCS i Cyber-secure configuration register 1	11004_H+i *200_H	PCS, 32	PCS, 32	Application Reset	5806
MCCSi_MCONTROL	MCCS i Cyber-secure MBIST control register	$1100C_H$+ i*200_H	PCS, 32	SV, PCS, 32	Application Reset	5808

(table continues...)

Table 1377 (continued) Registers overview - VMT (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
MCCSi_MSTATUS	MCCS i Cyber-secure status register	11010_H+i *200_H	32, PCS	BE	Application Reset	5810
MCCSi_RANGE	MCCS i Cyber-secure range register, single address mode	11014_H+i *200_H	PCS, 32	PCS, 32	Application Reset	5812
MCCSi_REVID	MCCS i Cyber-secure revision ID register	$1101C_H$+ i*200_H	32, PCS	BE	Application Reset	5813
MCCSi_ECCS	MCCS i Cyber-secure ECC safety register	11020_H+i *200_H	PCS, 32	SV, PCS, 32	Application Reset	5814
MCCSi_ECCD	MCCS i Cyber-secure memory ECC detection register	11024_H+i *200_H	PCS, 32	SV, PCS, 32	See 5816	5816
MCCSi_ETRR	MCCS i Cyber-secure error tracking register	11028_H+i *200_H	32, PCS	BE	PowerOn Reset	5818
MCCSi_RDBFLy	MCCS i Cyber-secure read data and bit flip register y	11080_H+i *200_H+y *4	PCS, 32	PCS, 32	Application Reset	5818
MCCSi_ALMSRCS	MCCS i Cyber-secure alarm sources configuration register	$111D0_H$+ i*200_H	PCS, 32	SV, PCS, 32	See 5819	5819
MCCSi_FAULTSTS	MCCS i Cyber-secure SSH safety faults status register	$111D4_H$+ i*200_H	PCS, 32	SV, PCS, 32	PowerOn Reset	5821
MCCSi_ERRINFO	MCCS i Cyber-secure error information register	$111D8_H$+ i*200_H	32, PCS	BE	PowerOn Reset	5822

34.4.3 Clock control register

The Clock Control Register CLC allows the programmer to adapt the functionality and power consumption of the module to the requirements of the application.

Register CLC controls the module clock signal and the reactivity to the sleep signal.

CLC Offset address: 00000_H

Clock control register Reset values see: Table 1378

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												EDIS	0	DISS	DISR
r												rw	r	rh	rw

Field	Bits	Type	Description
DISR	0	rw	**Module Disable Request Bit** Used for enable/disable control of the module. 0_B On request: enable the module clock 1_B Off request: stop the module clock
DISS	1	rh	**Module Disable Status Bit** When CLC.DISS=1_B, registers CLC, ID, PROTE, PROTSE, PROTCSP, ACCENNOM and ACCENCS are still accessible. Accesses to other VMT registers are not executed and will result in a bus error. 0_B Module clock is enabled 1_B Module clock is disabled
EDIS	3	rw	**Sleep Mode Enable Control** Used to control the module's reaction to sleep mode. 0_B Sleep mode request is enabled and functional 1_B Module disregards the sleep mode control signal
0	2, 31:4	r	**Reserved, write 0**

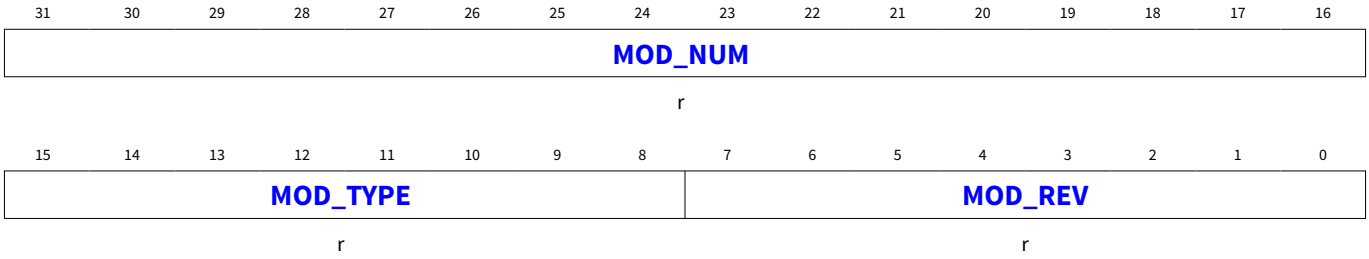
Table 1378 Reset values of CLC

Reset type	Reset value	Note
Application Reset	0000 0003$_H$	
After Boot-FW Value	0000 0000$_H$	Value installed by Boot Firmware.

34.4.4 Module identification register

ID Offset address: 00008$_H$

Module identification register PowerOn Reset value: 00C5 C013$_H$



Field	Bits	Type	Description
MOD_REV	7:0	r	**Module Revision** Indicates the revision number of the implementation.
MOD_TYPE	15:8	r	**Module Type** This internal marker is fixed to C0$_H$.
MOD_NUM	31:16	r	**Module Number** Indicates the module identification number.

34.4.5 PROT register endinit

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner or CS master to update the PROT state as described in the PROT mechanism.

The PROTE register controls lock / unlock of the local Endinit (E) protected control registers.

PROTE Offset address: 00020_H

PROT register endinit Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OWEN	ODEF	TAGID						PRSEN	PRS			VMEN	VM		
w	rw	rw						rw	rw			rw	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												SWEN	STATE		
r												w	rwh		

Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
VM	18:16	rw	**Virtual Machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual Machine definition Enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection Set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.

(table continues...)

(continued)

Field	Bits	Type	Description
PRSEN	23	rw	**Protection Set definition Enable for PROT owner** 0_B **Disable**: PRS is not part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; must be written as 0.

Table 1379 Access mode restrictions of PROTE sorted by descending priority

Mode name	Access mode		Description
PROT_OWNER and write 1 to .**SWEN**	rwh	STATE	PROT owner can modify the PROT state (for allowed and valid transitions)
PROT_OWNER and write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	PROT owner can modify the existing PROT owner
SECURE_MASTER and write 1 to .**SWEN**	rwh	STATE	Secure master can modify the PROT state (for allowed and valid transitions)
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

34.4.6 PROT register safe endinit

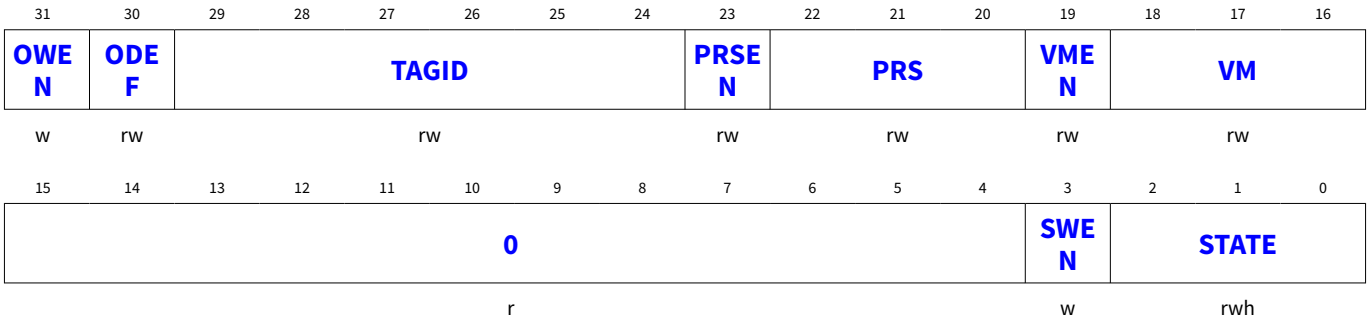
The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner or CS master to update the PROT state as described in the PROT mechanism.

The PROTSE register controls lock / unlock of the local Safe Endinit (SE) protected control registers.

PROTSE Offset address: 00024_H

PROT register safe endinit Application Reset value: $0000\ 0000_H$

34 Volatile Memory Test (VMT)



Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
VM	18:16	rw	**Virtual Machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual Machine definition Enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection Set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.
PRSEN	23	rw	**Protection Set definition Enable for PROT owner** 0_B **Disable**: PRS is not part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner

(table continues...)

(continued)

Field	Bits	Type	Description
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; must be written as 0.

Table 1380 Access mode restrictions of PROTSE sorted by descending priority

Mode name	Access mode		Description
PROT_OWNER and write 1 to .**SWEN**	rwh	STATE	PROT owner can modify the PROT state (for allowed and valid transitions)
PROT_OWNER and write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	PROT owner can modify the existing PROT owner
SECURE_MASTER and write 1 to .**SWEN**	rwh	STATE	Secure master can modify the PROT state (for allowed and valid transitions)
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

34.4.7 PROT register cyber-secure APU endinit

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner or CS master to update the PROT state as described in the PROT mechanism.

The PROTCSP register controls lock / unlock of the local protected control registers by the CS Master.

PROTCSP Offset address: $0002C_H$

PROT register cyber-secure APU endinit Application Reset value: $4C00\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OWEN	**ODEF**	**TAGID**						**PRSEN**	**PRS**			**VMEN**	**VM**		
w	rw	rw						rw	rw			rw	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												**SWEN**	**STATE**		
r												w	rwh		

Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
VM	18:16	rw	**Virtual Machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual Machine definition Enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection Set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.
PRSEN	23	rw	**Protection Set definition Enable for PROT owner** 0_B **Disable**: PRS is not part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; must be written as 0.

Table 1381 Access mode restrictions of PROTCSP sorted by descending priority

Mode name	Access mode		Description
PROT_OWNER and write 1 to .**SWEN**	rwh	STATE	PROT owner can modify the PROT state (for allowed and valid transitions)
PROT_OWNER and write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	PROT owner can modify the existing PROT owner
SECURE_MASTER and write 1 to .**SWEN**	rwh	STATE	Secure master can modify the PROT state (for allowed and valid transitions)
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

34.4.8 ACCENNOM write access enable register A

This register defines which master TAG ID encodings are enabled or disabled for write accesses to the access protected region.

ACCENNOM_WRA Offset address: 00040_H

ACCENNOM write access enable register A Application Reset value: $1000\ 0003_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	**EN30**	**EN29**	**EN28**	**EN27**	**EN26**	**EN25**	**EN24**	**EN23**	**EN22**	**EN21**	**EN20**	**EN19**	**EN18**	**EN17**	**EN16**
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	**EN14**	**EN13**	**EN12**	**EN11**	**EN10**	**EN09**	**EN08**	**EN07**	**EN06**	**EN05**	**EN04**	**EN03**	**EN02**	**EN01**	**EN00**
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Write access enable for Master TAG ID q** This bit enables write access to the access protected region for transactions with the Master TAG ID q 0_B Disabled for write access 1_B Enabled for write access

34.4.9 ACCENNOM write access enable register B

This register defines which master TAG ID encodings are enabled or disabled for write accesses to the access protected region.

ACCENNOM_WRB Offset address: 00044_H

ACCENNOM write access enable register B Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Write access enable for Master TAG ID q** This bit enables write access to the access protected region for transactions with the Master TAG ID q 0_B Disabled for write access 1_B Enabled for write access
0	31:8	r	**Reserved** Read as 0; must be written as 0.

34.4.10 ACCENNOM read access enable register A

This register defines which master TAG ID encodings are enabled or disabled for read accesses to the access protected region.

ACCENNOM_RDA Offset address: 00048_H

ACCENNOM read access enable register A Application Reset value: $FFFF\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Read access enable for Master TAG ID q** This bit enables read access to the access protected region for transactions with the Master TAG ID q 0_B Disabled for read access 1_B Enabled for read access

34.4.11 ACCENNOM read access enable register B

This register defines which master TAG ID encodings are enabled or disabled for read accesses to the access protected region.

ACCENNOM_RDB Offset address: $0004C_H$

ACCENNOM read access enable register B Application Reset value: $0000\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Read access enable for Master TAG ID q** This bit enables read access to the access protected region for transactions with the Master TAG ID q 0_B Disabled for read access 1_B Enabled for read access
0	31:8	r	**Reserved** Read as 0; must be written as 0.

34.4.12 ACCENNOM VM access enable register

This register defines which virtual machine encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each VMx.

ACCENNOM_VM Offset address: 00050_H

ACCENNOM VM access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for VM ID q** This bit enables read access to the access protected region for transactions with the VM ID q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for VM ID q** This bit enables write access to the access protected region for transactions with the VM ID q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; must be written as 0.

34.4.13 ACCENNOM PRS access enable register

This register defines which protection register set encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each PRSx.

ACCENNOM_PRS Offset address: 00054_H

ACCENNOM PRS access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for PRS q** This bit enables read access to the access protected region for transactions with the PRS q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for PRS q** This bit enables write access to the access protected region for transactions with the PRS q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; must be written as 0.

34.4.14 ACCENCS write access enable register A

This register defines which master TAG ID encodings are enabled or disabled for write accesses to the access protected region.

ACCENCS_WRA Offset address: 00060_H

ACCENCS write access enable register A Application Reset value: $0000\ 3000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Write access enable for Master TAG ID q** This bit enables write access to the access protected region for transactions with the Master TAG ID q 0_B Disabled for write access 1_B Enabled for write access

34.4.15 ACCENCS write access enable register B

This register defines which master TAG ID encodings are enabled or disabled for write accesses to the access protected region.

ACCENCS_WRB Offset address: 00064_H

ACCENCS write access enable register B Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Write access enable for Master TAG ID q** This bit enables write access to the access protected region for transactions with the Master TAG ID q 0_B Disabled for write access 1_B Enabled for write access
0	31:8	r	**Reserved** Read as 0; must be written as 0.

34.4.16 ACCENCS read access enable register A

This register defines which master TAG ID encodings are enabled or disabled for read accesses to the access protected region.

ACCENCS_RDA Offset address: 00068_H

ACCENCS read access enable register A Application Reset value: $0000\ 3000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Read access enable for Master TAG ID q** This bit enables read access to the access protected region for transactions with the Master TAG ID q 0_B Disabled for read access 1_B Enabled for read access

34.4.17 ACCENCS read access enable register B

This register defines which master TAG ID encodings are enabled or disabled for read accesses to the access protected region.

ACCENCS_RDB Offset address: $0006C_H$

ACCENCS read access enable register B Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Read access enable for Master TAG ID q** This bit enables read access to the access protected region for transactions with the Master TAG ID q 0_B Disabled for read access 1_B Enabled for read access
0	31:8	r	**Reserved** Read as 0; must be written as 0.

34.4.18 ACCENCS VM access enable register

This register defines which virtual machine encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each VMx.

ACCENCS_VM Offset address: 00070_H

ACCENCS VM access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for VM ID q** This bit enables read access to the access protected region for transactions with the VM ID q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for VM ID q** This bit enables write access to the access protected region for transactions with the VM ID q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; must be written as 0.

34.4.19 ACCENCS PRS access enable register

This register defines which protection register set encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each PRSx.

ACCENCS_PRS Offset address: 00074_H

ACCENCS PRS access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for PRS q** This bit enables read access to the access protected region for transactions with the PRS q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for PRS q** This bit enables write access to the access protected region for transactions with the PRS q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; must be written as 0.

34.4.20 Nominal memory MBIST enable register

The memory test register MEMTEST holds CPU configurable select bits for the various SSH instances. See the device specific chapter for mapping of memory controller numbers.

MEMTEST Offset address: 00080_H

Nominal memory MBIST enable register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN9	EN8	EN7	EN6	EN5	EN4	EN3	EN2	EN1	EN0
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
ENn (n=0-31)	n	rwh	**Enable/Disable SSHn** Writes change status of SSH / Reads provide current status of SSH 0_B Disable SSH/SSH is disabled. 1_B Enable SSH/SSH is enabled.

34.4.21 Nominal memory mapping enable register

The memory mapping enable register MEMMAP has configurable control bits to select memory-mapped test mode. Refer to the device specific configuration chapter for more information.

MEMMAP Offset address: 00084_H

Nominal memory mapping enable register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN9	EN8	EN7	EN6	EN5	EN4	EN3	EN2	EN1	EN0
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
ENn (n=0-31)	n	rwh	**Enable/Disable memory mapping n** 0_B Normal cache function 1_B Memory-mapped

34.4.22 Nominal memory status register

The memory status register MEMSTAT shows whether each SSH instance is currently executing an automatic initialization sequence.

MEMSTAT Offset address: 00088_H

Nominal memory status register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AIU31	AIU30	AIU29	AIU28	AIU27	AIU26	AIU25	AIU24	AIU23	AIU22	AIU21	AIU20	AIU19	AIU18	AIU17	AIU16
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AIU15	AIU14	AIU13	AIU12	AIU11	AIU10	AIU9	AIU8	AIU7	AIU6	AIU5	AIU4	AIU3	AIU2	AIU1	AIU0
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
AIUn (n=0-31)	n	rh	**Autoinitialization of Cache Partition Underway n** This bit indicates whether an automatic data initialization has been triggered by a change of state of MEMTEST.ENn or MEMMAP.ENn and whether the initialization has completed. 0_B SSHn is not running an autoinitialization. 1_B SSHn is running an autoinitialization.

34.4.23 Nominal memory test done status register

Each bit in the memory test done status register MEMDONE reflects the status of the MSTATUS.DONE bit in the corresponding SSH.

It notifies when a test is finished. It is 0 while a test is running and set to 1 again by hardware once the test is finished. This only happens if MCi_MCONTROL.START is reset by software first.

MEMDONE Offset address: $0008C_H$

Nominal memory test done status register Application Reset value: $FFFF\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DONE31	DONE30	DONE29	DONE28	DONE27	DONE26	DONE25	DONE24	DONE23	DONE22	DONE21	DONE20	DONE19	DONE18	DONE17	DONE16
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DONE15	DONE14	DONE13	DONE12	DONE11	DONE10	DONE9	DONE8	DONE7	DONE6	DONE5	DONE4	DONE3	DONE2	DONE1	DONE0
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
DONEn (n=0-31)	n	rh	**Memory Test Done Status n** This bit indicates whether an SRAM test is underway. 0_B Test is running. 1_B Test is finished.

34.4.24 Nominal memory test FDA status register

Each bit in the memory fail dump available status register MEMFDA reflects the status of the MSTATUS.FDA bit in the corresponding SSH.

MEMFDA Offset address: 00090_H

Nominal memory test FDA status register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FDA31	FDA30	FDA29	FDA28	FDA27	FDA26	FDA25	FDA24	FDA23	FDA22	FDA21	FDA20	FDA19	FDA18	FDA17	FDA16
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FDA15	FDA14	FDA13	FDA12	FDA11	FDA10	FDA9	FDA8	FDA7	FDA6	FDA5	FDA4	FDA3	FDA2	FDA1	FDA0
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
FDAn (n=0-31)	n	rh	**Memory Test FDA Status** This bit indicates whether Fail Bit Map data are available for read out. The value of this bit is only reliable if the corresponding MEMDONE bit is set to '1'. 0_B SSHn fail dump data not available. 1_B SSHn fail dump data available.

34.4.25 Nominal SSH CE alarm register

Each bit in the SSH CE alarm status register CEALARM reflects the status of the CE alarm signal of the corresponding SSH.

CEALARM Offset address: $000A4_H$

Nominal SSH CE alarm register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CEALM31	CEALM30	CEALM29	CEALM28	CEALM27	CEALM26	CEALM25	CEALM24	CEALM23	CEALM22	CEALM21	CEALM20	CEALM19	CEALM18	CEALM17	CEALM16
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CEALM15	CEALM14	CEALM13	CEALM12	CEALM11	CEALM10	CEALM9	CEALM8	CEALM7	CEALM6	CEALM5	CEALM4	CEALM3	CEALM2	CEALM1	CEALM0
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
CEALMn (n=0-31)	n	rh	**SSH CE Alarm Occurred** This bit indicates whether a CE alarm has been triggered by the corresponding SSH. 0_B SSHn ECCD.CERR = 0. 1_B SSHn ECCD.CERR = 1.

34.4.26 Nominal SSH UCE alarm register

Each bit in the SSH UCE alarm status register UCEALARM reflects the status of the UCE alarm signal of the corresponding SSH.

UCEALARM Offset address: 000A8$_H$

Nominal SSH UCE alarm register Application Reset value: 0000 0000$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UCEALM31	UCEALM30	UCEALM29	UCEALM28	UCEALM27	UCEALM26	UCEALM25	UCEALM24	UCEALM23	UCEALM22	UCEALM21	UCEALM20	UCEALM19	UCEALM18	UCEALM17	UCEALM16
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UCEALM15	UCEALM14	UCEALM13	UCEALM12	UCEALM11	UCEALM10	UCEALM9	UCEALM8	UCEALM7	UCEALM6	UCEALM5	UCEALM4	UCEALM3	UCEALM2	UCEALM1	UCEALM0
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
UCEALMn (n=0-31)	n	rh	**SSH UCE Alarm Occurred** This bit indicates whether an UCE alarm has been triggered by the corresponding SSH. 0_B SSHn ECCD.UCERR = 0. 1_B SSHn ECCD.UCERR = 1.

34.4.27 Nominal SSH ME alarm register

Each bit in the SSH ME alarm status register MEALARM reflects the status of the ME alarm signal of the corresponding SSH.

MEALARM Offset address: 000AC$_H$

Nominal SSH ME alarm register Application Reset value: 0000 0000$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MEALM31	MEALM30	MEALM29	MEALM28	MEALM27	MEALM26	MEALM25	MEALM24	MEALM23	MEALM22	MEALM21	MEALM20	MEALM19	MEALM18	MEALM17	MEALM16
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MEALM15	MEALM14	MEALM13	MEALM12	MEALM11	MEALM10	MEALM9	MEALM8	MEALM7	MEALM6	MEALM5	MEALM4	MEALM3	MEALM2	MEALM1	MEALM0
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
MEALMn (n=0-31)	n	rh	**SSH ME Alarm Occurred** This bit indicates whether an ME alarm has been triggered by the corresponding SSH. 0_B SSHn ECCD.MERR = 0. 1_B SSHn ECCD.MERR = 1.

34.4.28 Cyber-secure memory MBIST enable register

The memory test register MEMTESTCS holds CPU configurable select bits for security related SSH instances. See the device specific chapter for mapping of memory controller numbers.

MEMTESTCS Offset address: $000B4_H$

Cyber-secure memory MBIST enable register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN9	EN8	EN7	EN6	EN5	EN4	EN3	EN2	EN1	EN0
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
ENn (n=0-31)	n	rwh	**Enable/Disable SSHn** Writes change status of SSH / Reads provide current status of SSH 0_B Disable SSH/SSH is disabled. 1_B Enable SSH/SSH is enabled.

34.4.29 Cyber-secure memory mapping enable register

The memory mapping enable register MEMMAPCS has configurable control bits to select memory-mapped test mode. Refer to the device specific configuration chapter for more information.

MEMMAPCS Offset address: $000B8_H$

Cyber-secure memory mapping enable register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN9	EN8	EN7	EN6	EN5	EN4	EN3	EN2	EN1	EN0
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
ENn (n=0-31)	n	rwh	**Enable/Disable memory mapping n** 0_B Normal cache function 1_B Memory-mapped

34.4.30 Cyber-secure memory status register

The memory status register MEMSTATCS shows whether each SSH instance is currently executing an automatic initialization sequence.

MEMSTATCS Offset address: $000BC_H$

Cyber-secure memory status register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AIU31	AIU30	AIU29	AIU28	AIU27	AIU26	AIU25	AIU24	AIU23	AIU22	AIU21	AIU20	AIU19	AIU18	AIU17	AIU16
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AIU15	AIU14	AIU13	AIU12	AIU11	AIU10	AIU9	AIU8	AIU7	AIU6	AIU5	AIU4	AIU3	AIU2	AIU1	AIU0
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
AIUn (n=0-31)	n	rh	**Autoinitialization of Cache Partition Underway n** This bit indicates whether an automatic data initialization has been triggered by a change of state of MEMTESTCS.ENn or MEMMAPCS.ENn and whether the initialization has completed. 0_B SSHn is not running an autoinitialization. 1_B SSHn is running an autoinitialization.

34.4.31 Cyber-secure memory test done status register

Each bit in the memory test done status register MEMDONECS reflects the status of the MSTATUS.DONE bit in the corresponding SSH.

It notifies when a test is finished. It is 0 while a test is running and set to 1 again by hardware once the test is finished. This only happens if MCCSi_MCONTROL.START is reset by software first.

MEMDONECS Offset address: $000C0_H$

Cyber-secure memory test done status register Application Reset value: $FFFF\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DONE31	DONE30	DONE29	DONE28	DONE27	DONE26	DONE25	DONE24	DONE23	DONE22	DONE21	DONE20	DONE19	DONE18	DONE17	DONE16
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DONE15	DONE14	DONE13	DONE12	DONE11	DONE10	DONE9	DONE8	DONE7	DONE6	DONE5	DONE4	DONE3	DONE2	DONE1	DONE0
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
DONEn (n=0-31)	n	rh	**Memory Test Done Status n** This bit indicates whether an SRAM test is underway. 0_B Test is running. 1_B Test is finished.

34.4.32 Cyber-secure memory test FDA status register

Each bit in the memory fail dump available status register MEMFDACS reflects the status of the MSTATUS.FDA bit in the corresponding SSH.

MEMFDACS Offset address: $000C4_H$

Cyber-secure memory test FDA status register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FDA31	FDA30	FDA29	FDA28	FDA27	FDA26	FDA25	FDA24	FDA23	FDA22	FDA21	FDA20	FDA19	FDA18	FDA17	FDA16
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FDA15	FDA14	FDA13	FDA12	FDA11	FDA10	FDA9	FDA8	FDA7	FDA6	FDA5	FDA4	FDA3	FDA2	FDA1	FDA0
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
FDAn (n=0-31)	n	rh	**Memory Test FDA Status** This bit indicates whether Fail Bit Map data are available for read out. The value of this bit is only reliable if the corresponding MEMDONECS bit is set to '1'. 0_B SSHn fail dump data not available. 1_B SSHn fail dump data available.

34.4.33 Cyber-secure SSH CE alarm register

Each bit in the cyber-secure SSH CE alarm status register CEALARM reflects the status of the CE alarm signal of the corresponding SSH.

CEALARMCS Offset address: $000D8_H$

Cyber-secure SSH CE alarm register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CEALM31	CEALM30	CEALM29	CEALM28	CEALM27	CEALM26	CEALM25	CEALM24	CEALM23	CEALM22	CEALM21	CEALM20	CEALM19	CEALM18	CEALM17	CEALM16
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CEALM15	CEALM14	CEALM13	CEALM12	CEALM11	CEALM10	CEALM9	CEALM8	CEALM7	CEALM6	CEALM5	CEALM4	CEALM3	CEALM2	CEALM1	CEALM0
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
CEALMn (n=0-31)	n	rh	**SSH CE Alarm Occurred** This bit indicates whether a CE alarm has been triggered by the corresponding SSH. 0_B SSHn ECCD.CERR = 0. 1_B SSHn ECCD.CERR = 1.

34.4.34 Cyber-secure SSH UCE alarm register

Each bit in the cyber-secure SSH UCE alarm status register UCEALARMCS reflects the status of the UCE alarm signal of the corresponding SSH.

UCEALARMCS Offset address: $000DC_H$

Cyber-secure SSH UCE alarm register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UCEALM31	UCEALM30	UCEALM29	UCEALM28	UCEALM27	UCEALM26	UCEALM25	UCEALM24	UCEALM23	UCEALM22	UCEALM21	UCEALM20	UCEALM19	UCEALM18	UCEALM17	UCEALM16
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UCEALM15	UCEALM14	UCEALM13	UCEALM12	UCEALM11	UCEALM10	UCEALM9	UCEALM8	UCEALM7	UCEALM6	UCEALM5	UCEALM4	UCEALM3	UCEALM2	UCEALM1	UCEALM0
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
UCEALMn (n=0-31)	n	rh	**SSH UCE Alarm Occurred** This bit indicates whether an UCE alarm has been triggered by the corresponding SSH. 0_B SSHn ECCD.UCERR = 0. 1_B SSHn ECCD.UCERR = 1.

34.4.35 Cyber-Secure SSH ME alarm register

Each bit in the cyber-secure SSH ME alarm status register MEALARMCS reflects the status of the ME alarm signal of the corresponding SSH.

MEALARMCS Offset address: 000E0$_H$

Cyber-Secure SSH ME alarm register Application Reset value: 0000 0000$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MEALM31	MEALM30	MEALM29	MEALM28	MEALM27	MEALM26	MEALM25	MEALM24	MEALM23	MEALM22	MEALM21	MEALM20	MEALM19	MEALM18	MEALM17	MEALM16
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MEALM15	MEALM14	MEALM13	MEALM12	MEALM11	MEALM10	MEALM9	MEALM8	MEALM7	MEALM6	MEALM5	MEALM4	MEALM3	MEALM2	MEALM1	MEALM0
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
MEALMn (n=0-31)	n	rh	**SSH ME Alarm Occurred** This bit indicates whether an ME alarm has been triggered by the corresponding SSH. 0_B SSHn ECCD.MERR = 0. 1_B SSHn ECCD.MERR = 1.

34.4.36 MC i Nominal configuration register 0

The bits in the MCONTROL and CONFIG0/1 registers can be used to control and program any march and hammer sequences. All bits concerning these test are concentrated here. All bits do not change during a test run. Setting MCONTROL.START will start the tests defined here. MSTATUS.DONE is reset at the beginning of a test and set after completion once MCONTROL.START is cleared by software. If no legal operation was defined in CONFIG1.AG_MOD nothing will be done but the handshake of MCONTROL.START and MSTATUS.DONE is carried out.

The reset values of the CONFIG0/1 and MCONTROL registers will perform a {↑(w0,r0)} operation (direction up, write 0 to all cells and check for 0) which initializes the whole memory with 0 and checks the contents if MCONTROL.START is set. This is the start sequence of many tests.

MCi_CONFIG0 (i=0-31) Offset address: 01000$_H$+i*200$_H$

MC i Nominal configuration register 0 Application Reset value: 0000 2002$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NUMACCS				0				ACCSTYPE							
rw				r				rw							

Field	Bits	Type	Description
ACCSTYPE	7:0	rw	**Access type** This field specifies the type of access which is being performed to each single address in the current marching element. ACCSTYPE[n] specifies the n-th access of the marching element. 0_B write access 1_B read access
NUMACCS	15:12	rw	**Number of accesses per address** This field specifies the total number of accesses which are being performed to each single address in the current marching element. Allowed values: 0-8 (Due to size limitation of CONFIG0.ACCSTYPE and CONFIG1.ACCSPAT fields). If NUMACCS=0 no access will be performed. If NUMACCS > 8, 8 accesses will be performed.
0	11:8, 31:16	r	**Reserved** Read as 0; must be written as 0.

34.4.37 MC i Nominal configuration register 1

MCi_CONFIG1 (i=0-31) Offset address: $01004_H+i*200_H$

MC i Nominal configuration register 1 Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AG_MOD				SELFASTB				ACCSPAT							
rw				rw				rw							

Field	Bits	Type	Description
ACCSPAT	7:0	rw	**Access Pattern** When AG_MOD selects any test other than the non-destructive test, this field specifies directly the bit pattern (i.e. '0' or '1') which is being used for an access to each single address in the current marching element. ACCSPAT[n] specifies the n-th access of the marching element. These patterns are toggled according to MCONTROL.BITTOG and MCONTROL.ROWTOG. When AG_MOD selects the non-destructive test: Program 0 when the previous read access was with normal data; and 1 when the previous read was with inverted data. Note: When considering the previous read access, consider that the last access is a previous access to the first, as a 'wrap around'. Refer to section on non-destructive test for more details on how to program these bits.
SELFASTB	11:8	rw	**Select Fast Bit** Reserved Read as 0; must be written as 0.
AG_MOD	15:12	rw	**Address Generator Mode** These bits enable the special hardware for performing the more complex addressing schemes. In case RANGE.RAEN (range enable) is set to 0 (single access) linear address mode has to be selected and NUMACCS set to 1. 1_H-4_H, 8_H, and A_H are reserved. 0_H run the test with linear address generation 5_H run the non-destructive test. The march elements, direction and backgrounds are defined by other settings in CONFIG0/1 and MCONTROL. For this test, the SRAM has to be pre-initialized with valid content (i.e. with ECC correct data). Unlike a normal MBIST march test, this test uses the ECC itself to find errors in the data. The result of the test is not reflected via MSTATUS.FAIL - instead, the detected ECC errors are tracked in the ETRR and ERRINFO registers, and additionally ECCD:*ERR bits if the alarm notifications are enabled. **others,** only the handshake of START and DONE is carried out.
0	31:16	r	**Reserved** Read as 0; must be written as 0.

34.4.38 MC i Nominal MBIST control register

The bits in the MCONTROL and CONFIG0/1 registers can be used to control and program any march and hammer sequences. All bits concerning these test are concentrated here. All bits do not change during a test run. Setting MCONTROL.START will start the tests defined here. MSTATUS.DONE is reset at the beginning of a test and set after completion once MCONTROL.START is cleared by software. If no legal operation was defined in CONFIG1.AG_MOD nothing will be done but the handshake of MCONTROL.START and MSTATUS.DONE is carried out.

The reset values of the CONFIG0/1 and MCONTROL registers will perform a {↑(w0,r0)} operation (direction up, write 0 to all cells and check for 0) which initializes the whole memory with 0 and checks the contents if MCONTROL.START is set. This is the start sequence of many tests.

MCi_MCONTROL (i=0-31) Offset address: $0100C_H+i*200_H$

MC i Nominal MBIST control register Application Reset value: $0000\ 0008_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SRAM_CLR	0	RSVD12		INJERR	0	FAILDMP	RSVD8	BITTOG	ROWTOG	RCADR	DINIT	DIR	ESTF	RESUME	START
rw	r	rw		rw	r	rw	rw	rw	rw	rw	rw	rw	rw	rwh	rw

Field	Bits	Type	Description
START	0	rw	**START** If this bit is written to 1 by software the memory test will start. If it is reset by software, and the test has finished, MSTATUS.DONE will be set to 1. If MCONTROL.FAILDMP is set, a fail will stop the current execution. MCONTROL.RESUME will continue a suspended test. MCONTROL.START and MCONTROL.FAILDMP must be set with the same write access by the software. 0_B No test started, finished or waiting for test end 1_B Start memory test
RESUME	1	rwh	**Resume failed test** This bit allows a failing test that got suspended to be resumed after the dump of the fail bit map. A restart is possible only if MSTATUS.FDA was reset by hardware. It will be reset by hardware once the test is resumed. 0_B Do not resume 1_B Resume suspended MBIST run
ESTF	2	rw	**Enable Sticky Fail Bit** This bit enables the sticky fail bit MSTATUS.SFAIL. If set any fails will be collected in MSTATUS.SFAIL. Resetting this bit to 0 will also reset MSTATUS.SFAIL. 0_B Do not collect fail events 1_B Collect fail events

(table continues...)

(continued)

Field	Bits	Type	Description
DIR	3	rw	**Direction Select** This field specifies the direction of a memory test operation. 0_B DOWN: Address direction is highest to lowest. 1_B UP: Address direction is lowest to highest.
DINIT	4	rw	**Data Initialization Enable** This bit enables a write of the RDBFL data to all locations defined by the range register. RDBFL can contain data that will produce an ECC error. Execution is started with MCONTROL.START. For this predefined action any information contained in CONFIG0/1 registers and the bits MCONTROL.BITTOG, MCONTROL.ROWTAG and MCONTROL.DIR are ignored. 0_B Disabled 1_B Enabled
RCADR	5	rw	**Fast Row / Fast Column Addressing Scheme Select** This bit selects between fast row and fast column addressing. 'Fast Row' moves along the word-lines first and then in bit-line direction, 'Fast Column' along the bit-lines first. 0_B Fast row 1_B Fast column
ROWTOG	6	rw	**Row toggling** This field specifies whether to toggle the used bit pattern (non inverted/ inverted) with each physical memory row. This is required when writing a checkerboard pattern or a row stripe pattern. 0_B Do not toggle 1_B Do toggle with each row
BITTOG	7	rw	**Bit toggling** This field specifies whether to toggle the used bit pattern (non inverted/ inverted) with each physical memory column. This is required when writing a checkerboard pattern or a column stripe pattern. 0_B Do not toggle 1_B Do toggle with each column
RSVD8	8	rw	**Reserved** Read as 0; must be written as 0.

(table continues...)

(continued)

Field	Bits	Type	Description
FAILDMP	9	rw	**Fail bit map dump** This field enables a dump of the failing address and a fail bit map after a fault has been detected. The memory test is suspended afterwards and resumed by MCONTROL.RESUME. MSTATUS.FDA shows that a fail dump is available. In order to resume the test without the fail information, MCONTROL.FAILDMP should be reset by SW in the same write access to the MCONTROL, where MCONTROL.RESUME is set. In case a fail dump is available, RDBFL will contain the fail bit map and ETRR the failing address. MCONTROL.START and MCONTROL.FAILDMP must be set with the same write access by the software. 0_B Do not dump 1_B Dump each fault
INJERR	11	rw	**Inject Error** Enables error-injection during march tests. This is supported only for linear march tests if RANGE.RAEN = 1. This bit should be set before a march test is started and before the address where an error should be injected is written into the RANGE register with RAEN = 1. 0_B Do not mask any writes during march tests. RANGE.ADDR and RANGE.RAEN used normally. 1_B Use RANGE.ADDR as a pointer to a physical SRAM address to which write accesses during a march test will not be executed. This bit helps in error injection during a march test over the whole SRAM. This bit has an effect only when RANGE.RAEN is also set. With MCONTROL.INJERR and RANGE.RAEN = '1', the test is by default run over the entire SRAM.
RSVD12	13:12	rw	**Reserved** Read as 0; must be written as 0.
SRAM_CLR	15	rw	**Clear the SRAM** This bit initializes the complete SRAM with ECC correct 'All-0' data. Execution is started with MCONTROL.START. For this predefined action any information contained in CONFIG0/1, RANGE and the bits MCONTROL.BITTOG, MCONTROL.ROWTOG and MCONTROL.DIR are ignored. This bit shall not be set together with other initialization or test configurations. After the SRAM clearing is complete, the software has to reset this bit back to '0' before disabling the SSH. 0_B Do not clear the entire SRAM. 1_B Clear the entire SRAM. The SRAM is fully filled with zeroes and is also ECC correct.
0	10, 14, 31:16	r	**Reserved** Read as 0; must be written as 0.

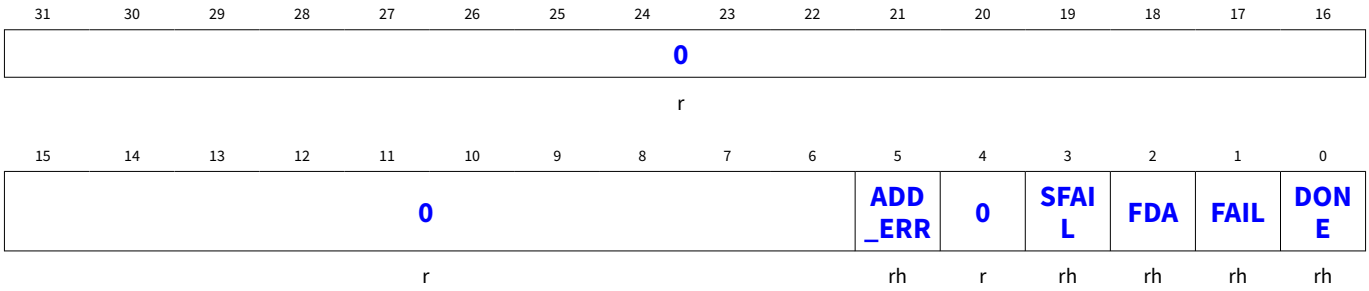
34.4.39 MC i Nominal status register

The bits in the status register show the status of the currently running and last test respectively.

MCi_MSTATUS (i=0-31) Offset address: $01010_H+i*200_H$

MC i Nominal status register Application Reset value: $0000\ 0001_H$

34 Volatile Memory Test (VMT)



Field	Bits	Type	Description
DONE	0	rh	**DONE** This bit is reset at the start of a test and set when a test is completed and MCONTROL.START was reset by software. It is not set when a test is interrupted for fail dump. 0_B Test is underway or a test has finished but MCONTROL.START was not reset by software 1_B Test has finished and MCONTROL.START was reset by software
FAIL	1	rh	**FAIL** This bit will be reset when a test is being started. It will be set to '1' by hardware under the following conditions: a) when performing a test: FAIL is set if the memory has at least one fault. Here fault includes any error in the data, as well as an address error, when enabled via ALMSRCS.ADDRE. In this case, an application reset has to be issued to clear the FAIL bit. b) when performing a test with MCONTROL.FAILDMP = '1': FAIL is set if the memory has at least one fault. The test is stopped and the fail bit map should be read from the RDBFL register 0_B No error occurred 1_B Detailed description see above
FDA	2	rh	**Fail Dump Available** This bit shows that a fail has occurred if MCONTROL.FAILDMP is set. The test is suspended and fail dump information is available. The fail bit map is in RDBFL and the associated address is in ETRR. As long as no fail has occurred RDBFL contains the last read information and ETRR has no valid data . This bit will be set by hardware. It will be reset when MSTATUS was read with MSTATUS.FDA = 1 and the dump information was read from ETRR and RDBFL. Only the last read from the last word of RDBFL is checked by the hardware and taken as an indication for a complete read. A suspended test will be resumed by MCONTROL.RESUME if FDA was reset. This forms some sort of handshake to insure that a suspended test can only be resumed (by a broadcasted) MCONTROL.RESUME if the last fail information was actually collected. 0_B No fail dump data available. A suspended MBIST run can be resumed. 1_B Fail dump data is available and waiting for read.

(table continues...)

(continued)

Field	Bits	Type	Description
SFAIL	3	rh	**Sticky Fail Bit** This bit is set to 1 together with MSTATUS.FAIL provided MCONTROL.ESTF is set. In contrast to MSTATUS.FAIL it will not be reset when a new test is started. Therefore it will collect fail information over more than one MBIST run. It will be reset when MCONTROL.ESTF is reset, or MBIST mode is switched off. 0_B No fail collected. 1_B A fail occured during one of the test runs since MCONTROL.ESTF was set to 1.
ADD_ERR	5	rh	**Address Error Detected During March Test** This bit will be reset when a test is being started. It will be set to '1' by hardware if the memory has an address error and the address error detection is enabled via ALMSRCS.ADDRE. 0_B No address error occurred 1_B Detailed description see above
0	4, 31:6	r	**Reserved** Read as 0; must be written as 0.

34.4.40 MC i Nominal range register, single address mode

The RANGE register can only be written if no test is underway and can be used to run a test only on a dedicated part of the RAM.

The range can be set in 256 word increments.

The RANGE register can be used to write to one specific address or read from it. In this case the range is disabled with RANGE.RAEN = '0' and the remaining part of the register is used as the address field.

The RANGE register can also be used to inject errors during a March test. This functionality can be enabled with MCONTROL.INJERR = '1'. MCONTROL.INJERR has to be set to '1' before the address (with RANGE.RAEN = '0') where an error should be injected is written into the RANGE register.

MCi_RANGE (i=0-31) Offset address: $01014_H+i*200_H$

MC i Nominal range register, single address mode Application Reset value: $0000\ XXXX_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RAEN	ADDR														
rw	rw														

Field	Bits	Type	Description
ADDR	14:0	rw	**Address** When RAEN = 0, this field specifies the address of a single memory location. Reads and writes to this location are possible. When RAEN=1, this field is interpreted as 2 different fields. ADDR[13:7] is interpreted as upper range limit. ADDR[6:0] is interpreted as lower range limit. For smaller SRAMs which require fewer address bits, the MSB bits are reserved. Writes to these bits are ignored, and reads return ‘0’.
RAEN	15	rw	**Range Enable** When RAEN = 0, a single word can be addressed for read or write. Config registers have to be set as follows: CONFIG0.NUMACCS:= '0001' (single access) CONFIG1.AG_MOD := '0000' (linear) MCONTROL.DIR :=1 (up) For read just the value in this location will be delivered. No check against expected values is done; i.e. MSTATUS.FAIL will not be set. When RAEN = 1, ADDR[13:7] is interpreted as upper range limit. ADDR[6:0] is interpreted as lower range limit. 0_B Disabled; single address mode is used 1_B Enabled; upper and lower range limit is used
0	31:16	r	**Reserved** Read as 0; must be written as 0.

Setting Address Ranges

If the RAEN field is set to ‘1’ then range mode is enabled. In this case, the ADDR field of the RANGE register is interpreted as two separate fields:

ADDR[13:7] is interpreted as the upper range limit.

This field specifies the upper logical block address limit in 256 word increments. Upper end of the address range is the concatenation of UPLIMIT and 0b11111111.

ADDR[6:0] is interpreted as the lower range limit

This field specifies the lower logical block address limit in 256 word increments. Lower end of the address range is the concatenation of LOLIMIT and 0b00000000.

Note that the default reset value of the ADDR field will be set to the maximum range of the physical memory. The default behaviour is therefore that an initialization or test will operate over the whole memory.

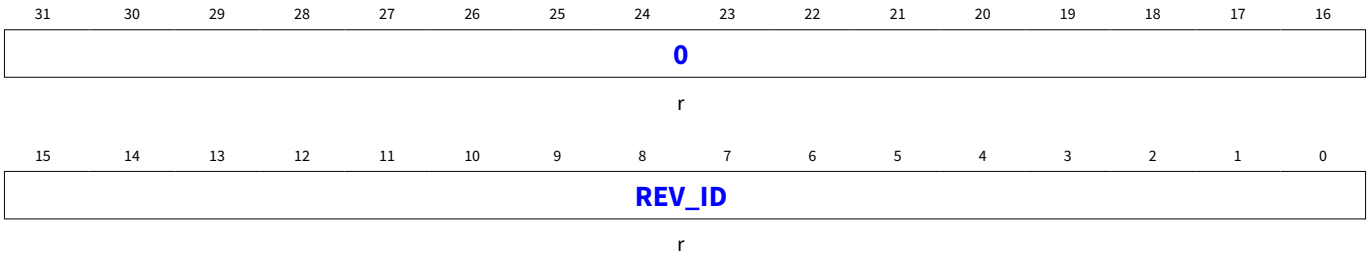
Also note that for smaller memories which require less than 15 bits of addressing, the relevant MSB bits of the address field are reserved.

34.4.41 MC i Nominal revision ID register

The revision ID register contains a hard-coded read only constant which describes the current version of the SSH IP.

MCi_REVID (i=0-31) Offset address: $0101C_H+i*200_H$

MC i Nominal revision ID register Application Reset value: $0000\ 0710_H$



Field	Bits	Type	Description
REV_ID	15:0	r	**Revision Identifier** This field defines the currently implemented release, version and functionality of the used SSH controller to track the SSH version for easier handling at the tester.
0	31:16	r	**Reserved** Read as 0; must be written as 0.

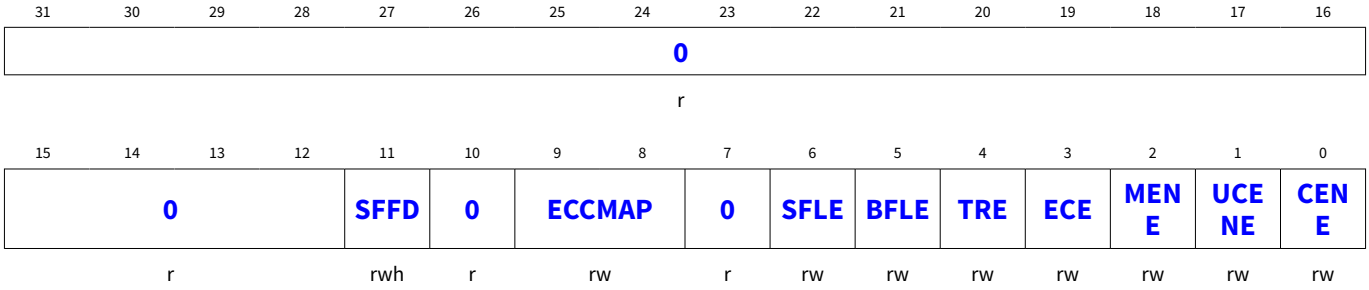
34.4.42 MC i Nominal ECC safety register

This register controls the various error detection and notification modes. This register can be accessed even if the corresponding SSH is not enabled using the MEMTEST.ENn bit.

The hardware features that implement fault tolerance mechanisms for the SRAMs (e.g. single bit correction) shall be enabled per default after any reset. This is ensured by the reset value of the ECC safety register where ECCS.ECE = 1 after a reset.

MCi_ECCS (i=0-31) Offset address: $01020_H+i*200_H$

MC i Nominal ECC safety register Application Reset value: $0000\ 001F_H$



Field	Bits	Type	Description
CENE	0	rw	**ECC Correction Event Alarm Notification Enable** This bit enables the forwarding of the CE alarm from the SSH to the SMU. 0_B Do not forward CE alarm to SMU. 1_B Forward the CE alarm to SMU.
UCENE	1	rw	**Uncorrectable Alarm Notification Enable** This bit enables the forwarding of the UCE alarm from the SSH to the SMU. Refer to the section on safety for more details. 0_B Do not forward the UCE alarm to the SMU. 1_B Forward the UCE alarm to the SMU

(table continues...)

(continued)

Field	Bits	Type	Description
MENE	2	rw	**Miscellaneous Alarm Notification Enable: MENE** This bit enables the forwarding of the ME alarm from the SSH to the SMU. Refer to the section on safety for more details. 0_B Do not forward the ME alarm to the SMU. 1_B Forward the ME alarm to the SMU.
ECE	3	rw	**Error Correction Enable** This enables the single bit error correction by the ECC. If this bit is 1, single bit errors are flagged via the CE alarm. If this bit is 0, single bit errors are flagged via the UCE alarm. 0_B Do not correct correctable errors. 1_B Correct correctable errors
TRE	4	rw	**Tracking Enable** Uncorrectable and address errors will be tracked, if the associated notification enable bit is set. This bit is enabled by default. 0_B Do not track address of detected error. 1_B Track address of detected error.
BFLE	5	rw	**Bit Flip Enable** 0_B Normal operation. 1_B Test mode only. Flips data and check bits according to RDBFL.
SFLE	6	rw	**Signature Bit Flip Enables** If this bit is set and address error detection is enabled (ALMSRCS.ADDRE = 1) then an address error is notified and tracked in the ETRR and ERRINFO registers. In addition an alarm is generated if enabled. 0_B Do not force address error injection. 1_B Forces address error injection by flipping bit 0 of the address to the address error detection logic, but not to the SRAM. This results in an address error.
ECCMAP	9:8	rw	**ECC Bit Mapping Mode** ECCMAP sets three different test modes to allow access to data or ECC bits separately and independently. 00_B Normal operation 01_B Test mode. Only data bits mapped. All ECC functionality disabled. 10_B Test mode. ECC check bits mapped to lower data bit positions. Other bits read as zero. All ECC functionality disabled. Data bits are not affected by write operations. 11_B Test mode. Do not use this setting.
SFFD	11	rwh	**Safety Flip-Flop Diagnostics** Safety flip-flop diagnostics bit. Setting this bit triggers a safety flip-flop self test. The result of the test (i.e. any error status in the safety FFs) - can be obtained from FAULTSTS.OPERR or FAULTSTS.MISCERR fields. 0_B Do not trigger a SFF self test. 1_B Trigger a SFF self test. Bit is cleared automatically by the hardware when the test is complete.

(table continues...)

(continued)

Field	Bits	Type	Description
0	7, 10, 31:12	r	**Reserved** Read as 0; must be written as 0.

34.4.43 MC i Nominal Memory ECC detection register

The ECC detection register contains information on the detected errors. The error bitfields of this register (SERR, CERR, UCERR, MERR) will be reset for easier software handling under the following conditions.

- The SSH is in MBIST mode (MEMTEST.ENn = 1) and no test is running (MEMDONE.DONEn = 0) and MCONTROL is written with both MCONTROL.START = 1 and MCONTROL.FAILDMP =1
- When the test is running with MCONTROL.FAILDMP = 1

The error bits will not be set as long as a test is running with MCONTROL.FAILDMP = 1.

MCi_ECCD (i=0-31) Offset address: $01024_H+i*200_H$

MC i Nominal Memory ECC detection register Reset values see: Table 1382

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0										VAL	TRC	MERR	UCERR	CERR	SERR
r										rh	w	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
SERR	0	rwh	**Error Detected** Write of ’0’ clears the sticky status. Write of ’1’ has no effect. In the case of a write of ‘0’ simultaneously with an error detection, the setting of the bit by hardware will take priority. This bit is reset with an application reset. Read as: 0_B No error detected. 1_B An error was detected and alarm forwarded: CERR, UCERR or MERR.
CERR	1	rwh	**Correctable Error Alarm Occurred** Write of ’0’ clears the bit, and enables further alarms to be forwarded to SMU. Write of ’1’ has no effect. Read as: 0_B No CE alarm event occurred. 1_B CE alarm event occurred.

(table continues...)

(continued)

Field	Bits	Type	Description
UCERR	2	rwh	**Uncorrectable Error Alarm Occurred** Write of ’0’ clears the bit, and enables further alarms to be forwarded to SMU. When the bit is set, software can perform additional diagnostics from the information in the ETRR/ERRINFO registers. Refer to the safety section for more details. Write of '1' has no effect. This bit is cleared on an application reset. Read as: 0_B No UCE alarm event occurred. 1_B UCE alarm event occurred.
MERR	3	rwh	**Miscellaneous Error Alarm Occurred** Write of ’0’ clears the bit, and enables further alarms to be forwarded to SMU. When the bit is set, software can perform additional diagnostics from the information in the FAULTSTS register. Refer to the safety section for more details. Write of '1' has no effect. This bit is reset with an application reset. Read as: 0_B No ME alarm event occurred. 1_B ME alarm event occurred.
TRC	4	w	**Tracking Clear** Writing this bit with '1' clears the VAL bit and the contents of the ETRR and ERRINFO registers. This bit will always read 0. 0_B No effect. 1_B Clears the ETRR and ERRINFO registers as well as the ECCD.VAL bit.
VAL	5	rh	**Valid Bits** The tracking register (ETRR) has a valid bit associated. Reset by ECCD.TRC. One error tracking register is available and one valid bit. This bit is preserved until a PORST.
0	31:6	r	**Reserved** Read as 0; must be written as 0.

Table 1382 Reset values of MCi_ECCD (i=0-31)

Reset type	Reset value	Note
PowerOn Reset	0000 0000 0000 0000 0000 0000 000X XXXX$_B$	Warm PORST
Application Reset	XXXX XXXX XXXX XXXX XXXX XXXX XXX0 0000$_B$	Application Reset

34.4.44 MC i Nominal error tracking register

This register contains the address of a detected error. One register is implemented per SSH instance. ETRR contains the first detected error. Uncorrectable and address errors are stored here in the same manner.

The error type (double-bit error (DBERR) or address error (ADDRERR)) for the tracked address is stored in the ERRINFO register. Only uncorrectable errors (double-bit errors or address errors) are tracked.

Only one error can be tracked at one time. If more than one error occurs at the same time in different memory blocks (SRAM towers) then the error that occurred in the lowest memory index will be tracked. The lowest memory index in which an error occurs will be stored in ERRINFO.MBI. In case multiple errors occur in two different halves of the same memory block (ECC granularity = 2), the errors in two different halves will be combined to a single error and the address of the corresponding memory block (SRAM tower) will be stored in the ETRR register.

The UCERR and the SERR bits in the ECCD register are also set accordingly if enabled.

The address tracked is physical, i.e. the address is directly equivalent to the SRAM address input signals.

The associated valid bit is contained in the register ECCD.VAL. ECCD.TRC clears the valid bit and the contents of ETRR and ERRINFO will also be cleared.

MCi_ETRR (i=0-31) Offset address: $01028_H+i*200_H$

MC i Nominal error tracking register PowerOn Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	ADDR														
r	rh														

Field	Bits	Type	Description
ADDR	14:0	rh	**Address of Error** Address of the first error detected since last clear operation. If some MSBs are not required for addressing smaller memories, they are reserved and read as ‘0’.
0	31:15	r	**Reserved** Read as 0; must be written as 0.

34.4.45 MC i Nominal read data and bit flip register y

This register is used for several purposes whenever a register with the size of the memory width is needed.

Normally when test mode is enabled this register contains the data which is directly read back from the RAM during the last read access.

During test mode, it contains the bit flip information. If ECCS.BFLE is set, this information is used to flip bits written to the SRAM.

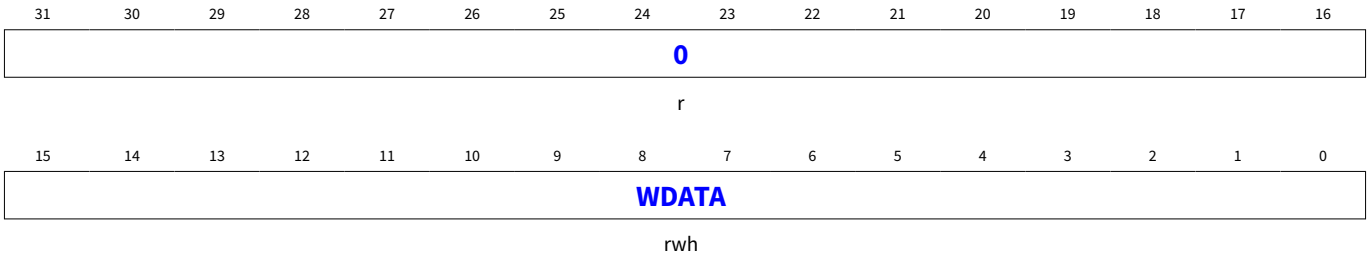
After a failed test it contains the fail bit map if MCONTROL.FAILDMP is set. The corresponding failure address is contained in the error tracking register.

When MCONTROL.DINIT is set, the content of RDBFL is written to all locations within range.

Writes to this register are not permitted while a test is underway.

MCi_RDBFLy (i=0-31;y=0-79) Offset address: $01080_H+i*200_H+y*4$

MC i Nominal read data and bit flip register y Application Reset value: $0000\ 0000_H$



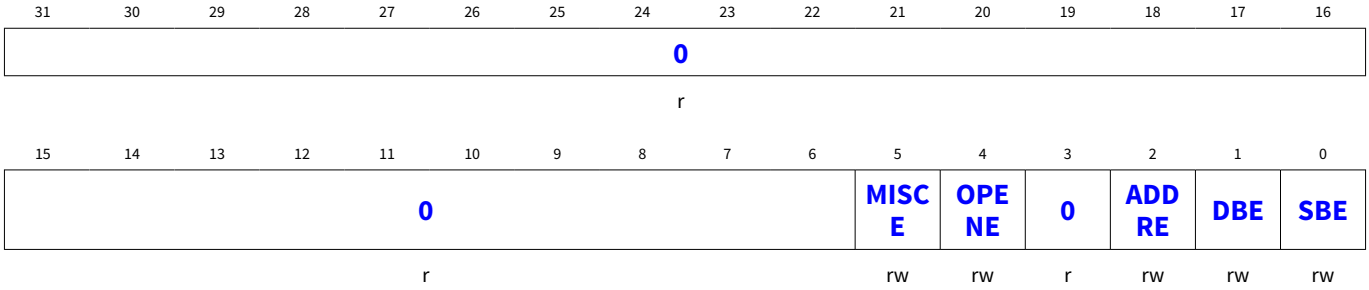
Field	Bits	Type	Description
WDATA	15:0	rwh	**Word Data** This field contains the data of the last memory read operation.
0	31:16	r	**Reserved** Read as 0; must be written as 0.

34.4.46 MC i Nominal alarm sources configuration register

This register controls the internal sources of the 3 SSH alarms (CE, UCE and ME). Individual sources can be enabled and disabled, and additionally provides individual status bits for the error/event sources.

MCi_ALMSRCS (i=0-31) Offset address: $011D0_H+i*200_H$

MC i Nominal alarm sources configuration register Reset values see: Table 1383



Field	Bits	Type	Description
SBE	0	rw	**Single Bit Error Notification** This bit enables ECC Single Bit Detection/Correction event to generate a CE or UCE alarm. If ECCS.ECE bit is '1', then SBE errors generate a CE alarm. Otherwise an UCE alarm is generated. Please note that SBEs are not tracked. 0_B SBEs are not notified via an alarm. 1_B SBEs are notified via an alarm (CE if ECCS.ECE = 1, UCE if ECCS.ECE = 0).
DBE	1	rw	**Double Bit Error Notification Enable** This bit enables ECC Double Bit Errors in the SRAM to be tracked and to generate an UCE alarm. The error status can be read from the ERRINFO register (ERRINFO.DBERR). 0_B DBEs are neither tracked in the ETRR, nor notified via an alarm. 1_B DBEs are tracked in the ETRR and ERRINFO, and notified via a UCE alarm.

(table continues...)

(continued)

Field	Bits	Type	Description
ADDRE	2	rw	**Address Error Notification Enable** This bit enables the detection and tracking of address faults in the SRAM and to generate an UCE alarm. The error status can be read from the ERRINFO register (ERRINFO.ADDRERR). 0_B Address faults in the SRAM are neither tracked in the ETRR, nor notified via an alarm. 1_B Address faults in the SRAM are tracked in the ETRR and ERRINFO, and notified via a UCE alarm.
OPENE	4	rw	**SSH Operational Error Notification Enable** This bit enables the forwarding of many errors which are critical to the operation of the SRAM or SSH. These errors generate an UCE alarm. The error status can be read from FAULTSTS.OPERR bits. This bit is enabled by default. 0_B Do not enable the detection and UCE alarm generation for SSH/SRAM operation critical errors. 1_B Enable the detection and UCE alarm generation for SSH/SRAM operation critical errors.
MISCE	5	rw	**SSH Miscellaneous Errors Notification Enable** This bit enables the forwarding of errors which may be critical to the operation of the SRAM or SSH. These errors generate an ME alarm. The error status can be read from FAULSTS.MISCERR. This bit is enabled by default. 0_B Do not enable the detection and ME alarm generation for miscellaneous SSH/SRAM errors. 1_B Enable the detection and ME alarm generation for miscellaneous SSH/SRAM errors.
0	3, 31:6	r	**Reserved** Read as 0; must be written as 0.

Table 1383 Reset values of MCi_ALMSRCS (i=0-31)

Reset type	Reset value	Note
After Boot-FW Value	$0000\ 0037_H$	Value installed by Boot Firmware.

34.4.47 MC i Nominal SSH safety faults status register

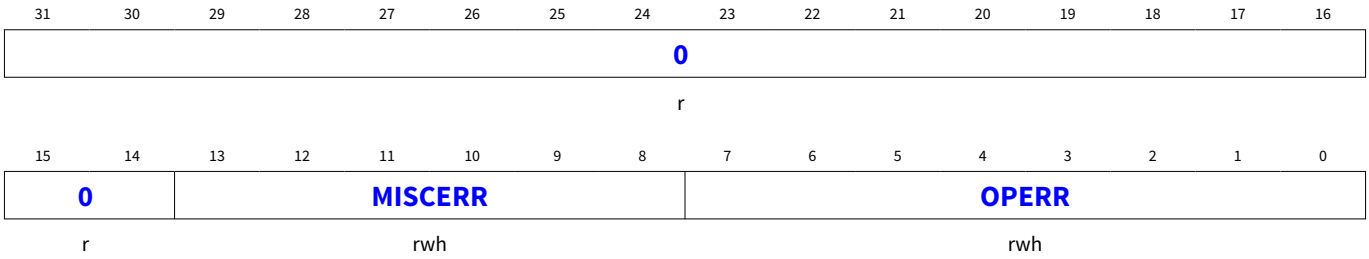
This register shows the status of the errors detected in the SRAM, which are notified by the UCE and ME alarms.

After power-up and before initialization by the SSW the safety flip-flops in the SSH can indicate a fault due to uninitialized internal registers. As a consequence MCi_FAULTSTS.OPERR[2] could be set and result in an alarm. LBIST does initialize the internal registers and clears the error. Alarms resulting from MCi_FAULTSTS.OPERR[2] should be ignored during start-up and cleared right after execution of the SSW in case LBIST was not run.

MCi_FAULTSTS (i=0-31) Offset address: $011D4_H+i*200_H$

MC i Nominal SSH safety faults status register PowerOn Reset value: $0000\ 0000_H$

34 Volatile Memory Test (VMT)



Field	Bits	Type	Description
OPERR	7:0	rwh	**SSH Critical Operation Error Status** Each bit reflects the status of a separate error source as described below. Bit field is enabled by ALMSRCS.OPENE. If multiple errors happened, multiple bits are set at the same time. To clear, write '0'. Write of '1' has no effect. Even if any bit is set, further errors are still forwarded. Unspecified bits are reserved for future use and shall always return 0. 01_H SSH has been enabled. Functional access to SRAM is disabled. 02_H Auto-data-init or partial-erase has been triggered. Part or whole of the SRAM may be overwritten. 04_H An error has been detected by safety flip-flops in one of the registers in the SSH. This bit is also always set at the end of a safety flip-flop self test (triggered by setting ECCS.SFFD) since the error paths are tested - but a real fail in the test is only indicated if FAULTSTS.MISCERR[0] is also set. 08_H Unexpected triggering of MBIST FSM, or test access to SRAM, or test features (e.g. multiplexer) in the data path, leading to potential data corruption. 10_H A mismatch occurred between one of the outputs of the ECC encoder and the ECC monitoring encoder. 20_H A mismatch occurred between one of the outputs of the ECC decoder and the ECC monitoring decoder.
MISCERR	13:8	rwh	**SSH Miscellaneous Error Status** Each bit reflects the status of a separate error source as described below. Bit field is enabled by ALMSRCS.MISCE. If multiple errors happened, multiple bits are set at the same time. To clear, write '0'. Write of '1' has no effect. Even if any bit is set, further errors are still forwarded. Unspecified bits are reserved for future use and shall always return 0. 01_H Failure detected during safety flip-flop self test (triggered by setting ECCS.SFFD). The ME alarm is not triggered by this fail. Hence the software shall read the status of this bit to check if the safety flip-flop self test failed. 02_H Alarm notification disabling detected. Any of the alarms may not be forwarded in the future. 04_H Safety mechanism disabling detected. Some of the SRAM or SSH related errors may not be detected in the future. 08_H Attempted access to a non-existing register in an existing SSH instance. 10_H An ongoing VMTx-SSH communication was aborted due to a module reset of the IP where the relevant SSH is embedded.

(table continues...)

(continued)

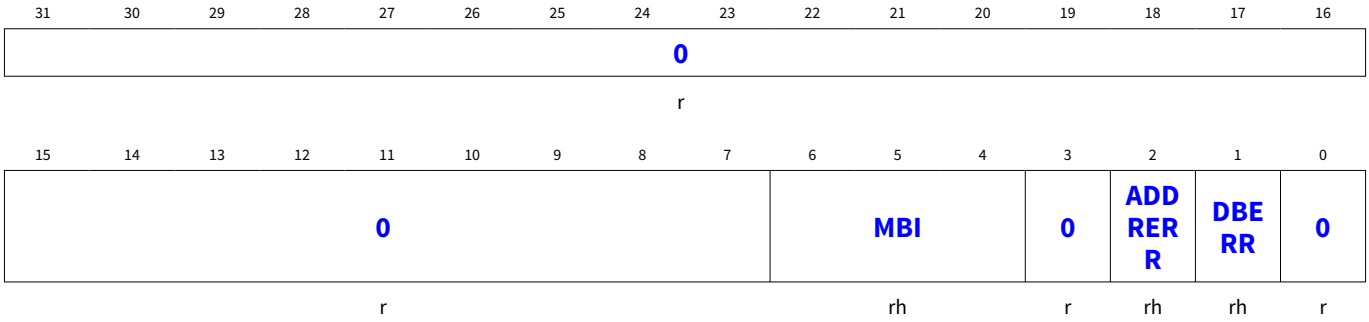
Field	Bits	Type	Description
0	31:14	r	**Reserved** Read as 0; must be written as 0.

34.4.48 MC i Nominal error information register

The register contains the type of error at the memory location corresponding to the ETRR address entry. The ERRINFO register is valid only when the ETRR entry is valid (ECCD.VAL = 1) and is cleared by ECCD.TRC.

MCi_ERRINFO (i=0-31) Offset address: $011D8_H+i*200_H$

MC i Nominal error information register PowerOn Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
DBERR	1	rh	**Double Bit Error Detected** This bit will be set to '1' by hardware if the memory has a double bit (or uncorrectable) error and the double bit error notification is enabled via ALMSRCS.DBE. The bit is cleared by ECCD.TRC. 0_B No double bit error detected at the memory address in the corresponding ETRR register. 1_B Double bit error detected at the memory address in the corresponding ETRR register.
ADDRERR	2	rh	**Address Fault Detected** This bit will be set to '1' by hardware if the memory has an address error and the address error notification is enabled via ALMSRCS.ADDRE. The bit is cleared by ECCD.TRC 0_B No address error detected at the memory address in the corresponding ETRR register. 1_B Address error detected at the memory address in the corresponding ETRR register.
MBI	6:4	rh	**Memory Block Index of Error** If more than one memory is implemented in parallel, these three bits contain the index of the erroneous memory block. The tracked address belongs to this memory. Otherwise these bits always are set to 0.
0	0, 3, 31:7	r	**Reserved** Read as 0; must be written as 0.

34.4.49 MCCS i Cyber-secure configuration register 0

The bits in the MCONTROL and CONFIG0/1 registers can be used to control and program any march and hammer sequences. All bits concerning these test are concentrated here. All bits do not change during a test run. Setting MCONTROL.START will start the tests defined here. MSTATUS.DONE is reset at the beginning of a test and set after completion once MCONTROL.START is cleared by software. If no legal operation was defined in CONFIG1.AG_MOD nothing will be done but the handshake of MCONTROL.START and MSTATUS.DONE is carried out.

The reset values of the CONFIG0/1 and MCONTROL registers will perform a {↑(w0,r0)} operation (direction up, write 0 to all cells and check for 0) which initializes the whole memory with 0 and checks the contents if MCONTROL.START is set. This is the start sequence of many tests.

MCCSi_CONFIG0 (i=0-31) Offset address: $11000_H+i*200_H$

MCCS i Cyber-secure configuration register 0 Application Reset value: $0000\ 2002_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NUMACCS				0				ACCSTYPE							
rw				r				rw							

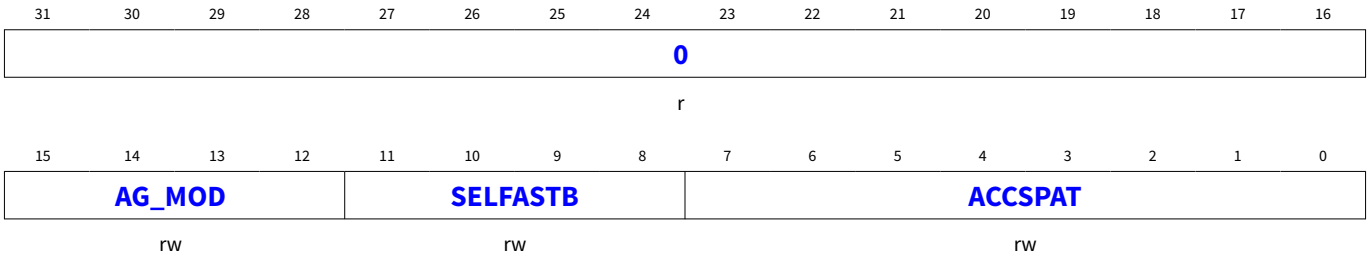
Field	Bits	Type	Description
ACCSTYPE	7:0	rw	**Access type** This field specifies the type of access which is being performed to each single address in the current marching element. ACCSTYPE[n] specifies the n-th access of the marching element. 0_B write access 1_B read access
NUMACCS	15:12	rw	**Number of accesses per address** This field specifies the total number of accesses which are being performed to each single address in the current marching element. Allowed values: 0-8 (Due to size limitation of CONFIG0.ACCSTYPE and CONFIG1.ACCSPAT fields). If NUMACCS=0 no access will be performed. If NUMACCS > 8, 8 accesses will be performed.
0	11:8, 31:16	r	**Reserved** Read as 0; must be written as 0.

34.4.50 MCCS i Cyber-secure configuration register 1

MCCSi_CONFIG1 (i=0-31) Offset address: $11004_H+i*200_H$

MCCS i Cyber-secure configuration register 1 Application Reset value: $0000\ 0000_H$

34 Volatile Memory Test (VMT)



Field	Bits	Type	Description
ACCSPAT	7:0	rw	**Access Pattern** When AG_MOD selects any test other than the non-destructive test, this field specifies directly the bit pattern (i.e. '0' or '1') which is being used for an access to each single address in the current marching element. ACCSPAT[n] specifies the n-th access of the marching element. These patterns are toggled according to MCONTROL.BITTOG and MCONTROL.ROWTOG. When AG_MOD selects the non-destructive test: Program 0 when the previous read access was with normal data; and 1 when the previous read was with inverted data. Note: When considering the previous read access, consider that the last access is a previous access to the first, as a 'wrap around'. Refer to section on non-destructive test for more details on how to program these bits.
SELFASTB	11:8	rw	**Select Fast Bit** Reserved Read as 0; must be written as 0.
AG_MOD	15:12	rw	**Address Generator Mode** These bits enable the special hardware for performing more complex addressing schemes. In case RANGE.RAEN (range enable) is set to 0 (single access) linear address mode has to be selected and NUMACCS set to 1. 1_H-4_H, 8_H, and A_H are reserved. 0_H run the test with linear address generation 5_H run the non-destructive test. The march elements, direction and backgrounds are defined by other settings in CONFIG0/1 and MCONTROL. For this test, the SRAM has to be pre-initialized with valid content (i.e. with ECC correct data). Unlike a normal MBIST march test, this test uses the ECC itself to find errors in the data. The result of the test is not reflected via MSTATUS.FAIL - instead, the detected ECC errors are tracked in the ETRR and ERRINFO registers, and additionally ECCD:*ERR bits if the alarm notifications are enabled. **others,** only the handshake of START and DONE is carried out.
0	31:16	r	**Reserved** Read as 0; must be written as 0.

34.4.51 MCCS i Cyber-secure MBIST control register

The bits in the MCONTROL and CONFIG0/1 registers can be used to control and program any march and hammer sequences. All bits concerning these test are concentrated here. All bits do not change during a test run. Setting MCONTROL.START will start the tests defined here. MSTATUS.DONE is reset at the beginning of a test and set after completion once MCONTROL.START is cleared by software. If no legal operation was defined in CONFIG1.AG_MOD nothing will be done but the handshake of MCONTROL.START and MSTATUS.DONE is carried out.

The reset values of the CONFIG0/1 and MCONTROL registers will perform a {↑(w0,r0)} operation (direction up, write 0 to all cells and check for 0) which initializes the whole memory with 0 and checks the contents if MCONTROL.START is set. This is the start sequence of many tests.

MCCSi_MCONTROL (i=0-31) Offset address: $1100C_H+i*200_H$

MCCS i Cyber-secure MBIST control register Application Reset value: $0000\ 0008_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SRAM_CLR	0	RSVD12		INJERR	0	FAILDMP	RSVD8	BITTOG	ROWTOG	RCADR	DINIT	DIR	ESTF	RESUME	START
rw	r	rw		rw	r	rw	rw	rw	rw	rw	rw	rw	rw	rwh	rw

Field	Bits	Type	Description
START	0	rw	**START** If this bit is written to 1 by software the memory test will start. If it is reset by software, and the test has finished, MSTATUS.DONE will be set to 1. If MCONTROL.FAILDMP is set, a fail will stop the current execution. MCONTROL.RESUME will continue a suspended test. MCONTROL.START and MCONTROL.FAILDMP must be set with the same write access by the software. 0_B No test started, finished or waiting for test end 1_B Start memory test
RESUME	1	rwh	**Resume failed test** This bit allows a failed test that got suspended to be resumed after the dump of the fail bit map. A restart is possible only if MSTATUS.FDA was reset by hardware. It will be reset by hardware once the test is resumed. 0_B Do not resume 1_B Resume suspended MBIST run
ESTF	2	rw	**Enable Sticky Fail Bit** This bit enables the sticky fail bit MSTATUS.SFAIL. If set any fails will be collected in MSTATUS.SFAIL. Resetting this bit to 0 will also reset MSTATUS.SFAIL. 0_B Do not collect fail events 1_B Collect fail events

(table continues...)

(continued)

Field	Bits	Type	Description
DIR	3	rw	**Direction Select** This field specifies the direction of a memory test operation. 0_B DOWN: Address direction is highest to lowest. 1_B UP: Address direction is lowest to highest.
DINIT	4	rw	**Data Initialization Enable** This bit enables a write of the RDBFL data to all locations defined by the range register. RDBFL can contain data that will produce an ECC error. Execution is started with MCONTROL.START. For this predefined action any information contained in CONFIG0/1 registers and the bits MCONTROL.BITTOG, MCONTROL.ROWTOG and MCONTROL.DINIT are ignored. 0_B Disabled 1_B Enabled
RCADR	5	rw	**Fast Row / Fast Column Addressing Scheme Select** This bit selects between fast row and fast column addressing. 'Fast Row' moves along the word-lines first and then in bit-line direction, 'Fast Column' along the bit-lines first. 0_B Fast row 1_B Fast column
ROWTOG	6	rw	**Row toggling** This field specifies whether to toggle the used bit pattern (non inverted/ inverted) with each physical memory row. This is required when writing a checkerboard pattern or a row stripe pattern. 0_B Do not toggle 1_B Do toggle with each row
BITTOG	7	rw	**Bit toggling** This field specifies whether to toggle the used bit pattern (non inverted/ inverted) with each physical memory column. This is required when writing a checkerboard pattern or a column stripe pattern. 0_B Do not toggle 1_B Do toggle with each column
RSVD8	8	rw	**Reserved** Read as 0; must be written as 0.

(table continues...)

(continued)

Field	Bits	Type	Description
FAILDMP	9	rw	**Fail bit map dump** This field enables a dump of the failing address and a fail bit map after a fault has been detected. The memory test is suspended afterwards and resumed by MCONTROL.RESUME. MSTATUS.FDA shows that a fail dump is available. In order to resume the test without the fail information, MCONTROL.FAILDMP should be reset by SW in the same write access to the MCONTROL, where MCONTROL.RESUME is set. In case a fail dump is available, RDBFL will contain the fail bit map and ETRR the failing address. MCONTROL.START and MCONTROL.FAILDMP must be set with the same write access by the software. 0_B Do not dump 1_B Dump each fault
INJERR	11	rw	**Inject Error** Enables error-injection during march tests. This is supported only for linear march tests if RANGE.RAEN = 1. This bit should be set before a march test is started and before the address where an error should be injected is written into the RANGE register with RAEN = 1. 0_B Do not mask any writes during march tests. RANGE.ADDR and RANGE.RAEN used normally. 1_B Use RANGE.ADDR as a pointer to a physical SRAM address to which write accesses during a march test will not be executed. This bit helps in error injection during a march test over the whole SRAM. This bit has an effect only when RANGE.RAEN is also set. With MCONTROL.INJERR and RANGE.RAEN = '1', the test is by default run over the entire SRAM.
RSVD12	13:12	rw	**Reserved** Read as 0; must be written as 0.
SRAM_CLR	15	rw	**Clear the SRAM** This bit initializes the complete SRAM with ECC correct 'All-0' data. Execution is started with MCONTROL.START. For this predefined action any information contained in CONFIG0/1, RANGE registers and the bits MCONTROL.BITTOG, MCONTROL.ROWTOG and MCONTROL.DIR are ignored. This bit shall not be set together with other initialization or test configurations. After the SRAM clearing is complete, the software has to reset this bit back to '0' before disabling the SSH. 0_B Do not clear the entire SRAM. 1_B Clear the entire SRAM. The SRAM is fully filled with zeroes and is also ECC correct.
0	10, 14, 31:16	r	**Reserved** Read as 0; must be written as 0.

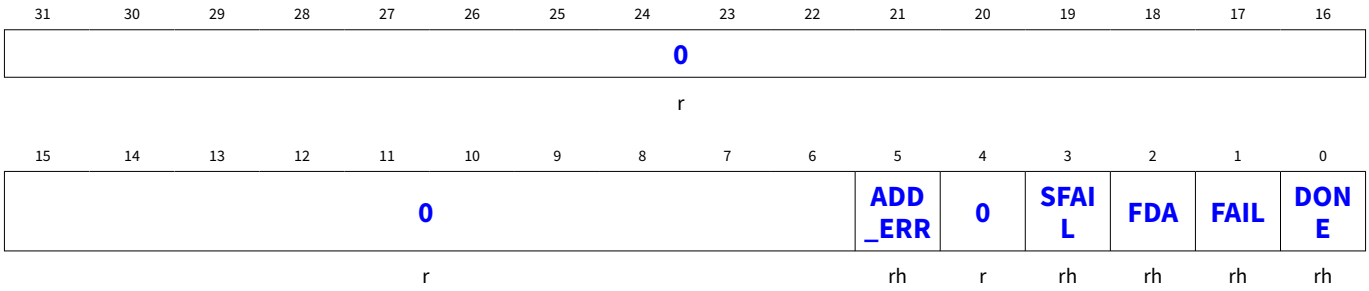
34.4.52 MCCS i Cyber-secure status register

The bits in the status register show the status of the currently running and last test respectively.

MCCSi_MSTATUS (i=0-31) Offset address: $11010_H+i*200_H$

MCCS i Cyber-secure status register Application Reset value: $0000\ 0001_H$

34 Volatile Memory Test (VMT)



Field	Bits	Type	Description
DONE	0	rh	**DONE** This bit is reset at the start of a test and set when a test is completed and MCONTROL.START was reset by software. It is not set when a test is interrupted for fail dump. 0_B Test is underway or a test has finished but MCONTROL.START was not reset by software 1_B Test has finished and MCONTROL.START was reset by software
FAIL	1	rh	**FAIL** This bit will be reset when a test is being started. It will be set to '1' by hardware under the following conditions: a) when performing a test: FAIL is set, if the memory has at least one fault. Here fault includes any error in the data, as well as an address error, when enabled via ALMSRCS.ADDRE. In this case, an application reset has to be issued to clear the FAIL bit. b) when performing a test with MCONTROL.FAILDMP = '1': FAIL is set, if the memory has at least one fault. The test is stopped and the fail bit map should be read from the RDBFL register 0_B No error occurred 1_B Detailed description see above
FDA	2	rh	**Fail Dump Available** This bit shows that a fail has occurred if MCONTROL.FAILDMP is set. The test is suspended and fail dump information is available. The fail bit map is in RDBFL and the associated address is in ETRR. As long as no fail has occurred RDBFL contains the last read information and ETRR has no valid data . This bit will be set by hardware. It will be reset when MSTATUS was read with MSTATUS.FDA = 1 and the dump information was read from ETRR and RDBFL. Only the last read from the last word of RDBFL is checked by the hardware and taken as an indication for a complete read. A suspended test will be resumed by MCONTROL.RESUME if FDA was reset. This forms some sort of handshake to insure that a suspended test can only be resumed (by a broadcasted) MCONTROL.RESUME if the last fail information was actually collected. 0_B No fail dump data available. A suspended MBIST run can be resumed. 1_B Fail dump data is available and waiting for read.

(table continues...)

(continued)

Field	Bits	Type	Description
SFAIL	3	rh	**Sticky Fail Bit** This bit is set to 1 together with MSTATUS.FAIL provided MCONTROL.ESTF is set. In contrast to MSTATUS.FAIL it will not be reset when a new test is started. Therefore it will collect fail information over more than one MBIST run. It will be reset when MCONTROL.ESTF is reset, or MBIST mode is switched off. 0_B No fail collected. 1_B A fail occurred during one of the test runs since MCONTROL.ESTF was set to 1.
ADD_ERR	5	rh	**Address Error Detected During March Test** This bit will be reset when a test is being started. It will be set to '1' by hardware if the memory has an address error and the address error detection is enabled via ALMSRCS.ADDRE. 0_B No address error occurred 1_B Detailed description see above
0	4, 31:6	r	**Reserved** Read as 0; must be written as 0.

34.4.53 MCCS i Cyber-secure range register, single address mode

The RANGE register can only be written if no test is underway and can be used to run a test only on a dedicated part of the RAM.

The range can be set in 256 word increments.

The RANGE register can be used to write to one specific address or read from it. In this case the range is disabled with RANGE.RAEN = '0' and the remaining part of the register is used as the address field.

The RANGE register can also be used to inject errors during a March test. This functionality can be enabled with MCONTROL.INJERR = '1'. MCONTROL.INJERR has to be set to '1' before the address (with RANGE.RAEN = '0') where an error should be injected is written into the RANGE register.

MCCSi_RANGE (i=0-31) Offset address: $11014_H+i*200_H$

MCCS i Cyber-secure range register, single address mode Application Reset value: $0000\ XXXX_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RAEN	ADDR														
rw	rw														

Field	Bits	Type	Description
ADDR	14:0	rw	**Address** When RAEN = 0, this field specifies the address of a single memory location. Reads and writes to this location are possible. When RAEN=1, this field is interpreted as 2 different fields. ADDR[13:7] is interpreted as upper range limit. ADDR[6:0] is interpreted as lower range limit. For smaller SRAMs which require fewer address bits, the MSB bits are reserved. Writes to these bits are ignored, and reads return ‘0’.
RAEN	15	rw	**Range Enable** When RAEN = 0, a single word can be addressed for read or write. Config registers have to be set as follows: CONFIG0.NUMACCS:= '0001' (single access) CONFIG1.AG_MOD := '0000' (linear) MCONTROL.DIR :=1 (up) For read just the value in this location will be delivered. No check against expected values is done; i.e. MSTATUS.FAIL will not be set. When RAEN = 1, ADDR[13:7] is interpreted as upper range limit. ADDR[6:0] is interpreted as lower range limit. 0_B Disabled; single address mode is used 1_B Enabled; upper and lower range limit is used
0	31:16	r	**Reserved** Read as 0; must be written as 0.

Setting Address Ranges

If the RAEN field is set to ‘1’ then range mode is enabled. In this case, the ADDR field of the RANGE register is interpreted as two separate fields:

ADDR[13:7] is interpreted as the upper range limit.

This field specifies the upper logical block address limit in 256 word increments. Upper end of the address range is the concatenation of UPLIMIT and 0b11111111.

ADDR[6:0] is interpreted as the lower range limit

This field specifies the lower logical block address limit in 256 word increments. Lower end of the address range is the concatenation of LOLIMIT and 0b00000000.

Note that the default reset value of the ADDR field will be set to the maximum range of the physical memory. The default behaviour is therefore that an initialization or test will operate over the whole memory.

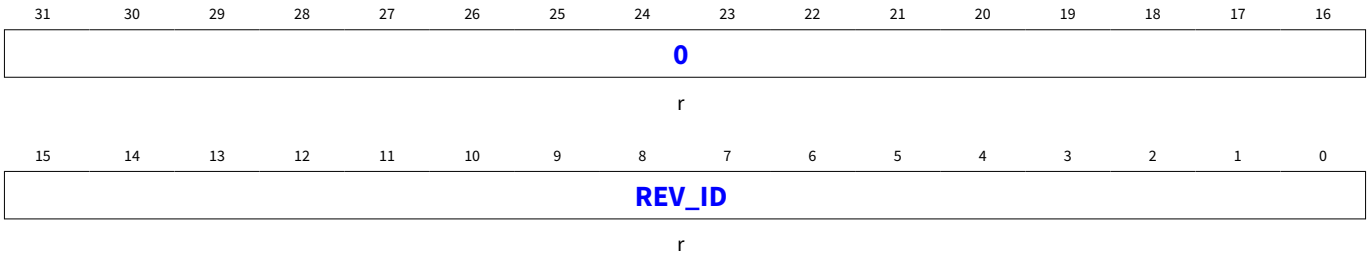
Also note that for smaller memories which require less than 15 bits of addressing, the relevant MSB bits of the address field are reserved.

34.4.54 MCCS i Cyber-secure revision ID register

The revision ID register contains a hard-coded read only constant which describes the current version of the SSH IP.

MCCSi_REVID (i=0-31) Offset address: $1101C_H+i*200_H$

MCCS i Cyber-secure revision ID register Application Reset value: $0000\ 0710_H$



Field	Bits	Type	Description
REV_ID	15:0	r	**Revision Identifier** This field defines the currently implemented release, version and functionality of the used SSH controller to track the SSH version for easier handling at the tester.
0	31:16	r	**Reserved** Read as 0; must be written as 0.

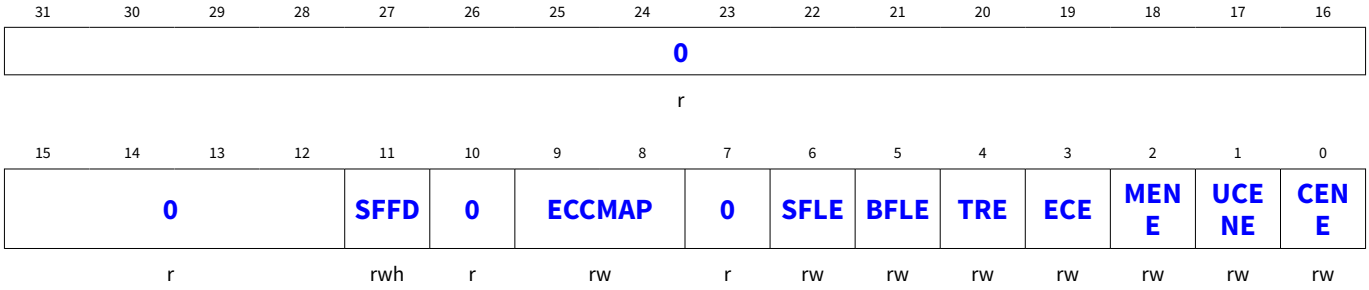
34.4.55 MCCS i Cyber-secure ECC safety register

This register controls the various error detection and notification modes. This register can be accessed even if the corresponding SSH is not enabled using the MEMTESTCS.ENn bit.

The hardware features that implement fault tolerance mechanisms for the SRAMs (e.g. single bit correction) shall be enabled per default after any reset. This is ensured by the reset value of the ECC safety register where ECCS.ECE = 1 after a reset.

MCCSi_ECCS (i=0-31) Offset address: $11020_H+i*200_H$

MCCS i Cyber-secure ECC safety register Application Reset value: $0000\ 001F_H$



Field	Bits	Type	Description
CENE	0	rw	**ECC Correction Event Alarm Notification Enable** This bit enables the forwarding of the CE alarm from the SSH to the SMU. 0_B Do not forward CE alarm to SMU. 1_B Forward the CE alarm to SMU.
UCENE	1	rw	**Uncorrectable Alarm Notification Enable** This bit enables the forwarding of the UCE alarm from the SSH to the SMU. Refer to the section on safety for more details. 0_B Do not forward the UCE alarm to the SMU. 1_B Forward the UCE alarm to the SMU

(table continues...)

(continued)

Field	Bits	Type	Description
MENE	2	rw	**Miscellaneous Alarm Notification Enable** This bit enables the forwarding of the ME alarm from the SSH to the SMU. Refer to the section on safety for more details. 0_B Do not forward the ME alarm to the SMU. 1_B Forward the ME alarm to the SMU.
ECE	3	rw	**Error Correction Enable** This enables the single bit error correction by the ECC. If this bit is 1, single bit errors are flagged via the CE alarm. If this bit is 0, single bit errors are flagged via the UCE alarm. 0_B Do not correct correctable errors. 1_B Correct correctable errors
TRE	4	rw	**Tracking Enable** Uncorrectable and address errors will be tracked, if the associated notification enable bit is set. This bit is enabled by default. 0_B Do not track address of detected error. 1_B Track address of detected error.
BFLE	5	rw	**Bit Flip Enable** 0_B Normal operation. 1_B Test mode only. Flips data and check bits according to RDBFL.
SFLE	6	rw	**Signature Bit Flip Enable** If this bit is set and address error detection is enabled (ALMSRCS.ADDRE = 1) then an address error is notified and tracked in the ETRR and ERRINFO registers. In addition an alarm is generated if enabled. 0_B Do not force address error injection. 1_B Forces address error injection by flipping bit 0 of the address to the address error detection logic, but not to the SRAM. This results in an address error.
ECCMAP	9:8	rw	**ECC Bit Mapping Mode** ECCMAP sets three different test modes to allow access to data or ECC bits separately and independently. 00_B Normal operation 01_B Test mode. Only data bits mapped. All ECC functionality disabled. 10_B Test mode. ECC check bits mapped to lower data bit positions. Other bits read as zero. All ECC functionality disabled. Data bits are not affected by write operations. 11_B Test mode. Do not use this setting.
SFFD	11	rwh	**Safety Flip-Flop Diagnostics** Safety flip-flop diagnostics bit. Setting this bit triggers a safety flip-flop self test. The result of the test (i.e. any error status in the SFFs) - can be obtained from the FAULTSTS.OPERR or FAULTSTS.MISCERR fields. 0_B Do not trigger a SFF self test. 1_B Trigger a SFF self test. Bit is cleared automatically by the hardware when the test is complete.

(table continues...)

(continued)

Field	Bits	Type	Description
0	7, 10, 31:12	r	**Reserved** Read as 0; must be written as 0.

34.4.56 MCCS i Cyber-secure memory ECC detection register

The ECC detection register contains information on the detected errors. The error bitfields of this register (SERR, CERR, UCERR, MERR) shall be reset for easier software handling under the following conditions.

- The SSH is in MBIST mode (MEMTESTCS.ENn = 1) and no test is running (MEMDONECS.DONEn = 0) and MCONTROL is written with both MCONTROL.START = 1 & MCONTROL.FAILDMP = 1.
- When the test is running with MCONTROL.FAILDMP = 1

The error bits will not be set as long as a test is running with MCONTROL.FAILDMP = 1.

MCCSi_ECCD (i=0-31) Offset address: $11024_H+i*200_H$

MCCS i Cyber-secure memory ECC detection register Reset values see: Table 1384

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0										VAL	TRC	MERR	UCERR	CERR	SERR
r										rh	w	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
SERR	0	rwh	**Error Detected** Write of ’0’ clears the sticky status. Write of ’1’ has no effect. In the case of a write of ‘0’ simultaneously with an error detection, the setting of the bit by hardware will take priority. This bit is reset with an application reset. Read as: 0_B No error detected. 1_B An error was detected and alarm forwarded: CERR, UCERR or MERR.
CERR	1	rwh	**Correctable Error Alarm Occurred** Write of ’0’ clears the bit, and enables further alarms to be forwarded to SMU. Write of ’1’ has no effect. Read as: 0_B No CE alarm event occurred. 1_B CE alarm event occurred.

(table continues...)

(continued)

Field	Bits	Type	Description
UCERR	2	rwh	**Uncorrectable Error Alarm Occurred** Write of ’0’ clears the bit, and enables further alarms to be forwarded to SMU. When the bit is set, software can perform additional diagnostics from the information in the ETRR/ERRINFO registers. Refer to the safety section for more details. Write of '1' has no effect. This bit is cleared on an application reset. Read as: 0_B No UCE alarm event occurred. 1_B UCE alarm event occurred
MERR	3	rwh	**Miscellaneous Error Alarm Occurred** Write of ’0’ clears the bit, and enables further alarms to be forwarded to SMU. When the bit is set, software can perform additional diagnostics from the information in the FAULTSTS register. Refer to the safety section for more details. Write of '1' has no effect. This bit is reset with an application reset. Read as: 0_B No ME alarm event occurred. 1_B ME alarm event occurred.
TRC	4	w	**Tracking Clear** Writing this bit with '1' clears the VAL bit and the ETRR and ERRINFO registers. This bit will always read 0. 0_B No effect. 1_B Clear the ETRR and ERRINFO registers as well as the ECCD.VAL bit.
VAL	5	rh	**Valid Bits** The tracking register (ETRR) has a valid bit associated. Reset by ECCD.TRC. One error tracking register is available and one valid bit. This bit is preserved until a PORST.
0	31:6	r	**Reserved** Read as 0; must be written as 0.

Table 1384 Reset values of MCCSi_ECCD (i=0-31)

Reset type	Reset value	Note
PowerOn Reset	0000 0000 0000 0000 0000 0000 000X XXXX$_B$	Warm PORST
Application Reset	XXXX XXXX XXXX XXXX XXXX XXXX XXX0 0000$_B$	Application Reset

34.4.57 MCCS i Cyber-secure error tracking register

This register contains the address of a detected error. One register is implemented per SSH instance. ETRR contains address of the first detected error. Errors happening afterwards do not overwrite the content of the ETRR register. Uncorrectable and address errors are stored here in the same manner.

The error type (double-bit error (DBERR) or address error (ADDRERR)) for the tracked address is stored in the ERRINFO register. Only uncorrectable errors (double-bit errors or address errors) are tracked.

Only one error can be tracked at one time. If more than one error occurs at the same time in different memory blocks (SRAM towers) then the error that occurred in the lowest memory index will be tracked. The lowest memory index in which an error occurs will be stored in ERRINFO.MBI. In case multiple errors occur in two different halves of the same memory block (ECC granularity = 2), the errors in two different halves will be combined to a single error and the address of the corresponding memory block (SRAM tower) will be stored in the ETRR register.

The UCERR and the SERR bits in the ECCD register are also set accordingly if enabled.

The address tracked is physical, i.e. the address is directly equivalent to the SRAM address input signals.

The associated valid bit is contained in the register ECCD.VAL. ECCD.TRC clears the valid bit and the contents of ETRR and ERRINFO will also be cleared.

MCCSi_ETRR (i=0-31) Offset address: $11028_H+i*200_H$

MCCS i Cyber-secure error tracking register PowerOn Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	ADDR														
r	rh														

Field	Bits	Type	Description
ADDR	14:0	rh	**Address of Error** Address of the first error detected since last clear operation. If some MSBs are not required for addressing smaller memories, they are reserved and read as ‘0’.
0	31:15	r	**Reserved** Read as 0; must be written as 0.

34.4.58 MCCS i Cyber-secure read data and bit flip register y

This register is used for several purposes whenever a register with the size of the memory width is needed.

Normally when test mode is enabled this register contains the data which is directly read back from the RAM during the last read access.

During test mode, it contains the bit flip information. If ECCS.BFLE is set, this information is used to flip bits written to the SRAM.

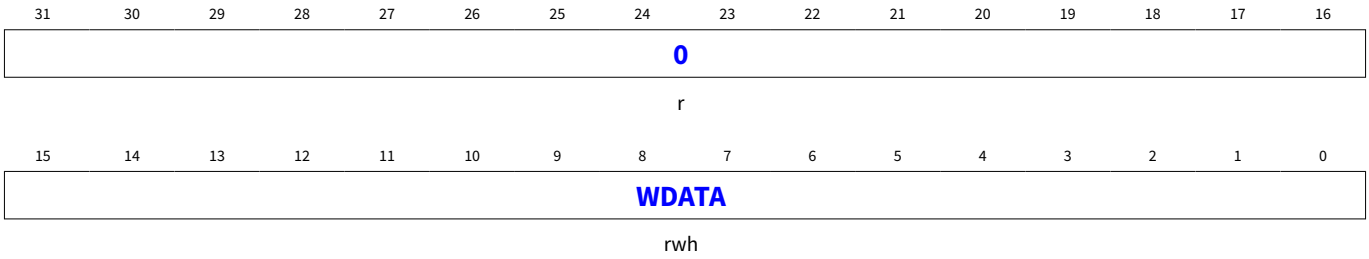
After a failed test it contains the fail bit map if MCONTROL.FAILDMP is set. The corresponding failure address is contained in the error tracking register.

When MCONTROL.DINIT is set, the content of RDBFL is written to all locations within range.

Writes to this register are not permitted while a test is underway.

MCCSi_RDBFLy (i=0-31;y=0-79) Offset address: $11080_H+i*200_H+y*4$

MCCS i Cyber-secure read data and bit flip register y Application Reset value: $0000\ 0000_H$



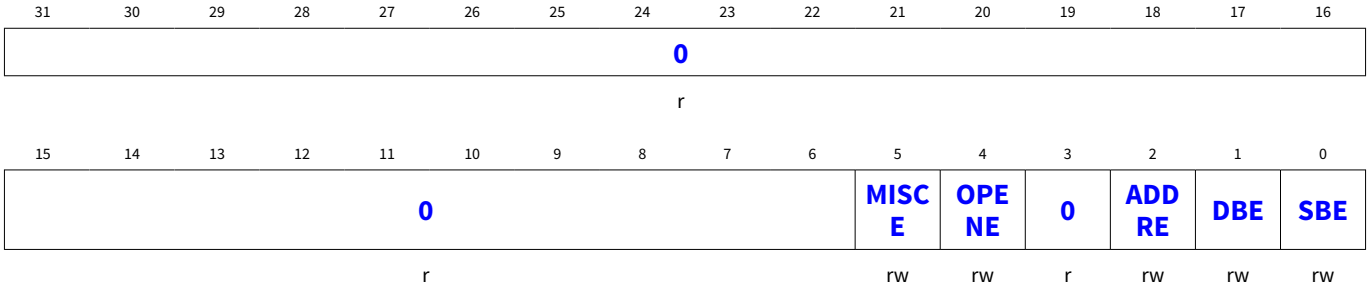
Field	Bits	Type	Description
WDATA	15:0	rwh	**Word Data** This field contains the data of the last memory read operation.
0	31:16	r	**Reserved** Read as 0; must be written as 0.

34.4.59 MCCS i Cyber-secure alarm sources configuration register

This register controls the internal sources of the 3 SSH alarms (CE, UCE and ME). Individual sources can be enabled and disabled.

MCCSi_ALMSRCS (i=0-31) Offset address: $111D0_H+i*200_H$

MCCS i Cyber-secure alarm sources configuration register Reset values see: Table 1385



Field	Bits	Type	Description
SBE	0	rw	**Single Bit Error Notification** This bit enables ECC Single Bit Detection/Correction event to generate an CE or UCE alarm. If ECCS.ECE bit is '1', then SBEs generate an CE alarm. Otherwise an UCE alarm is generated. Please note that SBEs are not tracked. 0_B SBE errors are not notified via an alarm. 1_B SBE errors are notified via an alarm (CE if ECCS.ECE = 1, UCE if ECCS.ECE = 0).

(table continues...)

(continued)

Field	Bits	Type	Description
DBE	1	rw	**Double Bit Error Notification Enable** This bit enables ECC Double Bit Errors in the SRAM to be tracked and to generate an UCE alarm. The error status can be read from the ERRINFO register (ERRINFO.DBERR). 0_B DBE errors are neither tracked in the ETRR, nor notified via an alarm. 1_B DBE errors are tracked in the ETRR & ERRINFO, and notified via a UCE alarm.
ADDRE	2	rw	**Address Error Notification Enable** This bit enables the detection and tracking of address faults in the SRAM and to generate an UCE alarm. The error status can be read from the ERRINFO register (ERRINFO.ADDRERR). 0_B Address faults in the SRAM are neither tracked in the ETRR, nor notified via an alarm. 1_B Address faults in the SRAM are tracked in the ETRR & ERRINFO, and notified via a UCE alarm.
OPENE	4	rw	**SSH Operational Error Notification Enable** This bit enables the forwarding of many errors which are critical to the operation of the SRAM or SSH. These errors generate an UCE alarm. The error status can be read from FAULTSTS.OPERR bits. This bit is enabled by default. 0_B Do not enable the detection and UCE alarm generation for SSH/SRAM operation critical errors. 1_B Enable the detection and UCE alarm generation for SSH/SRAM operation critical errors.
MISCE	5	rw	**SSH Miscellaneous Errors Notification Enable** This bit enables the forwarding of errors which may be critical to the operation of the SRAM or SSH. These errors generate an ME alarm. The error status can be read from FAULSTS.MISCERR. This bit is enabled by default. 0_B Do not enable the detection and ME alarm generation for miscellaneous SSH/SRAM errors. 1_B Enable the detection and miscellaneous alarm generation for misc. SSH/SRAM errors.
0	3, 31:6	r	**Reserved** Read as 0; must be written as 0.

Table 1385 Reset values of MCCSi_ALMSRCS (i=0-31)

Reset type	Reset value	Note
After Boot-FW Value	$0000\ 0037_H$	Value installed by Boot Firmware.

34.4.60 MCCS i Cyber-secure SSH safety faults status register

This register shows the status of the errors detected in the SRAM, which are notified by the UCE and ME alarms.

After power-up and before initialization by the SSW the safety flip-flops in the SSH can indicate a fault due to uninitialized internal registers. As a consequence MCCSi_FAULTSTS.OPERR[2] could be set and result in an alarm. LBIST does initialize the internal registers and clears the error. Alarms resulting from MCCSi_FAULTSTS.OPERR[2] should be ignored during start-up and cleared right after execution of the SSW in case LBIST was not run.

MCCSi_FAULTSTS (i=0-31) Offset address: $111D4_H+i*200_H$

MCCS i Cyber-secure SSH safety faults status register PowerOn Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		MISCERR						OPERR							
r		rwh						rwh							

Field	Bits	Type	Description
OPERR	7:0	rwh	**SSH Critical Operation Error Occurred** Each bit reflects the status of a separate error source as described below. Bit field is enabled by ALMSRCS.OPENE. If multiple errors happened, multiple bits are set at the same time. To clear, write '0'. Write of '1' has no effect. Even if any bit is set, further errors are still forwarded. Unspecified bits are reserved for future use and shall always return 0. 01_H SSH has been enabled. Functional access to SRAM is disabled. 02_H Auto-data-init or partial-erase has been triggered. Part or whole of the SRAM may be overwritten. 04_H An error has been detected by safety flip-flops in one of the registers in the SSH. This bit is also always set at the end of a safety flip-flop self test (triggered by setting ECCS.SFFD) since the error paths are tested - but a real fail in the test is only indicated if FAULTSTS.MISCERR[0] is also set. 08_H Unexpected triggering of MBIST FSM, or test access to SRAM, or test features (e.g. multiplexer) in the data path, leading to potential data corruption. 10_H A mismatch occurred between one of the outputs of the ECC encoder and the ECC monitoring encoder. 20_H A mismatch occurred between one of the outputs of the ECC decoder and the ECC monitoring decoder.

(table continues...)

(continued)

Field	Bits	Type	Description
MISCERR	13:8	rwh	**SSH Miscellaneous Error Status** Each bit reflects the status of a separate error source as described below. Bit field is enabled by ALMSRCS.MISCE. If multiple errors happened, multiple bits are set at the same time. To clear, write '0'. Write of '1' has no effect. Even if any bit is set, further errors are still forwarded. Unspecified bits are reserved for future use and shall always return 0. 01_H Failure detected during safety flip-flop self test (triggered by setting ECCS.SFFD). The ME alarm is not triggered by this fail. Hence the software shall read the status of this bit to check if the safety flip-flop self test failed. 02_H Alarm notification disabling detected. Any of the alarms may not be forwarded in the future. 04_H Safety mechanism disabling detected. Some of the SRAM or SSH related errors may not be detected in the future. 08_H Attempted access to a non-existing register in an existing SSH instance. 10_H An ongoing VMTx-SSH communication was aborted due to a module reset of the IP where the relevant SSH is embedded.
0	31:14	r	**Reserved** Read as 0; must be written as 0.

34.4.61 MCCS i Cyber-secure error information register

The register contains the type of error at the memory location corresponding to the ETRR address entry. The ERRINFO register is valid only when the ETRR entry is valid (ECCD.VAL = 1) and is cleared by ECCD.TRC.

MCCSi_ERRINFO (i=0-31) Offset address: $111D8_H+i*200_H$

MCCS i Cyber-secure error information register PowerOn Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0									MBI			0	ADDRERR	DBERR	0
r									rh			r	rh	rh	r

Field	Bits	Type	Description
DBERR	1	rh	**Double Bit Error Detected** This bit will be set to '1' by hardware if the memory has a double bit (or uncorrectable) error and the double bit error notification is enabled via ALMSRCS.DBE. The bit is cleared by ECCD.TRC. 0_B No double bit error detected at the memory address in the corresponding ETRR register. 1_B Double bit error detected at the memory address in the corresponding ETRR register.
ADDRERR	2	rh	**Address Fault Detected** This bit will be set to '1' by hardware if the memory has an address error and the address error notification is enabled via ALMSRCS.ADDRE. The bit is cleared by ECCD.TRC 0_B No address error detected at the memory address in the corresponding ETRR register. 1_B Address error detected at the memory address in the corresponding ETRR register.
MBI	6:4	rh	**Memory Block Index of Error** If more than one memory is implemented in parallel, these three bits contain the index of the erroneous memory block. The tracked address belongs to this memory. Otherwise these bits always are set to 0.
0	0, 3, 31:7	r	**Reserved** Read as 0; must be written as 0.

34.5 Debug information

This section is not applicable for the module being described.

34.6 References

This section is not applicable for the module being described.

34.7 VMT revision history

Reference	Description of change(s)
Date range: 2024-08-17 to 2025-01-31	
MCCSi_MCONTROL	• Removed reference to unsupported bit MCONTROL.GATH in MCONTROL.FAILDMP description.
MCi_MCONTROL	• Removed reference to unsupported bit MCONTROL.GATH in MCONTROL.FAILDMP description.
MCCSi_MCONTROL and MCi_MCONTROL	• Corrected field definition for bit RSVD8 and bit-field RSVD12 [13:12]

34.8 TC4Dx VMT information

34.8.1 TC4Dx VMT configuration

34.8.1.1 VMT module mapping

Multiple VMTs (VMTx) are distributed over the chip. Each VMTx handles up to 32 SSHs (SRAM Support Hardware). The following table gives an overview of which modules are covered by which VMT instance.

Table 1386 VMT module mapping

VMTx (x=)	Modules
0	CPU0-2, CSS, DMA0, MCDS4P, LMU0-1, LMU4-6
1	LMU2-3, LMU7-9, DMA1, MCDS2P, ERAY0-1, PSI5, CPU3-5
2	FSI0-1, HSPHY, PCIE0, GETH, PCIE1
3	PPU
4	CSRM, PKC, CDSP
5	CANXL, DRE, CAN0-4, SDMMC, LETH
6	SCR

34.8.1.2 SSH instances

The SRAMs have different configurations. The following table shows the instance-specific configurations of the SSH in each VMTx instance. Each SSH bit-index position "i" can be directly derived from "MC(CS)i" column in the SSH instances table for the following registers:

- MEMTEST(CS)
- MEMSTAT(CS)
- MEMDONE(CS)
- MEMFDA(CS)
- CEALARM(CS)
- UCEALARM(CS)
- MEALARM(CS)

Table 1387 SSH instances

VMTx (x=)	MC(CS)i (i=)	SRAM Name	Group	ECC Type	ECC Granularity	MUX Factor	# of Towers	Tower width	# of RDBFLs	Reset-Val MC(CS)i_RANGE	Note
0	0	CPU0_DMEM	G1	SECDED	2	16	4	44	11	$9F80_H$	SRAM_CLR
0	1	CPU0_DTAG	G1	SECDED	2	4	1	64	4	8000_H	SRAM_CLR

(table continues...)

Table 1387 (continued) SSH instances

VMTx (x=)	MC(CS)i (i=)	SRAM Name	Group	ECC Type	ECC Granularity	MUX Factor	# of Towers	Tower width	# of RDBFLs	Reset-Val MC(CS)i_RANGE	Note
0	2	CPU0_PMEM	G2_1	SECDED	1	8	2	72	9	$8B80_H$	SRAM_CLR
0	3	CPU0_PTAG	G1	DED	2	4	1	58	4	8080_H	SRAM_CLR
0	4	CPU0_DLMU_STBY	G2_1	SECDED	2	8	1	144	9	$BF80_H$	SRAM_CLR NDT
0	5	CPU1_DMEM	G1	SECDED	2	16	4	44	11	$9F80_H$	SRAM_CLR
0	6	CPU1_DTAG	G2_1	SECDED	2	4	1	64	4	8000_H	SRAM_CLR
0	7	CPU1_PMEM	G2_1	SECDED	1	8	2	72	9	$8B80_H$	SRAM_CLR
0	8	CPU1_PTAG	G2_1	DED	2	4	1	58	4	8080_H	SRAM_CLR
0	9	CPU1_DLMU_STBY	G2_1	SECDED	2	8	1	144	9	$BF80_H$	SRAM_CLR NDT
0	10	CPU2_DMEM	G1	SECDED	2	16	4	44	11	$9F80_H$	SRAM_CLR
0	11	CPU2_DTAG	G2_1	SECDED	2	4	1	64	4	8000_H	SRAM_CLR
0	12	CPU2_PMEM	G2_1	SECDED	1	8	2	72	9	$8B80_H$	SRAM_CLR
0	13	CPU2_PTAG	G2_1	DED	2	4	1	58	4	8080_H	SRAM_CLR
0	14	CPU2_DLMU_STBY	G2_1	SECDED	2	8	1	144	9	$BF80_H$	SRAM_CLR NDT
0	15_CS	CSS_KEYATTR	N/A	SECDED	1	4	1	22	2	8080_H	**EXCLUDE FROM TEST**
0	16_CS	CSS_KEYSTORE	N/A	SECDED	1	4	4	39	10	8080_H	**EXCLUDE FROM TEST**

(table continues...)

Table 1387 (continued) SSH instances

VMTx (x=)	MC(CS)i (i=)	SRAM Name	Group	ECC Type	ECC Granularity	MUX Factor	# of Towers	Tower width	# of RDBFLs	Reset-Val MC(CS)i_RANGE	Note
0	17_CS	CSS_FIFO0	G2_1	SECDED	1	4	4	39	10	8080_H	**EXCLUDE FROM TEST**
0	18_CS	CSS_FIFO1	G2_1	SECDED	1	4	4	39	10	8080_H	**EXCLUDE FROM TEST**
0	19_CS	CSS_CONTEXT	G2_1	SECDED	1	4	4	39	10	8080_H	**EXCLUDE FROM TEST**
0	20	SADMA0	G1	SECDED	1	4	4	72	18	8000_H	SRAM_CLR
0	21	MCDS4P_TBUF	N/A	NOECC	1	4	2	128	16	8180_H	**EXCLUDE FROM TEST**
0	22	MCDS4P_LUT	N/A	NOECC	1	8	4	32	8	8380_H	**EXCLUDE FROM TEST**
0	23	LMU0	G2_1	SECDED	2	8	1	144	9	$BF80_H$	SRAM_CLR
0	24	LMU1	G2_1	SECDED	2	8	1	144	9	$BF80_H$	SRAM_CLR
0	25	LMU4	G2_1	SECDED	2	8	1	144	9	$BF80_H$	SRAM_CLR
0	26	LMU5	G2_1	SECDED	2	8	1	144	9	$BF80_H$	SRAM_CLR
0	27	LMU6	G2_1	SECDED	2	8	1	144	9	$BF80_H$	SRAM_CLR
1	0	LMU2	G2_1	SECDED	2	8	1	144	9	$BF80_H$	SRAM_CLR
1	1	LMU3	G2_1	SECDED	2	8	1	144	9	$BF80_H$	SRAM_CLR
1	2	LMU7	G2_1	SECDED	2	8	1	144	9	$BF80_H$	SRAM_CLR

(table continues...)

Table 1387 (continued) SSH instances

VMTx (x=)	MC(CS)i (i=)	SRAM Name	Group	ECC Type	ECC Granularity	MUX Factor	# of Towers	Tower width	# of RDBFLs	Reset-Val MC(CS)i_RANGE	Note
1	3	LMU8	G2_1	SECDED	2	8	1	144	9	$BF80_H$	SRAM_CLR
1	4	LMU9	G2_1	SECDED	2	8	1	144	9	$BF80_H$	SRAM_CLR
1	5	SADMA1	G2_1	SECDED	1	4	4	72	18	8000_H	SRAM_CLR
1	6	MCDS2P_TBUF	N/A	NOECC	1	4	2	128	16	8080_H	**EXCLUDE FROM TEST**
1	7	MCDS2P_LUT	N/A	NOECC	1	8	2	32	4	8380_H	**EXCLUDE FROM TEST**
1	8	ERAY_OBF0	G2_1	SECDED	1	4	2	39	5	8000_H	SRAM_CLR
1	9	ERAY_OBF1	G2_1	SECDED	1	4	2	39	5	8000_H	SRAM_CLR
1	10	ERAY_TBF_IBF0	G2_1	SECDED	1	4	4	39	10	8000_H	SRAM_CLR
1	11	ERAY_TBF_IBF1	G2_1	SECDED	1	4	4	39	10	8000_H	SRAM_CLR
1	12	ERAY_MBF0	G2_1	SECDED	1	4	1	39	3	8380_H	SRAM_CLR
1	13	ERAY_MBF1	G2_1	SECDED	1	4	1	39	3	8380_H	SRAM_CLR
1	14	PSI5	G2_1	SECDED	1	4	1	39	3	8000_H	SRAM_CLR
1	15	CPU3_DMEM	G1	SECDED	2	16	4	44	11	$9F80_H$	SRAM_CLR
1	16	CPU3_DTAG	G2_1	SECDED	2	4	1	64	4	8000_H	SRAM_CLR
1	17	CPU3_PMEM	G2_1	SECDED	1	8	2	72	9	$8B80_H$	SRAM_CLR

(table continues...)

Table 1387 (continued) SSH instances

VMTx (x=)	MC(CS)i (i=)	SRAM Name	Group	ECC Type	ECC Granularity	MUX Factor	# of Towers	Tower width	# of RDBFLs	Reset-Val MC(CS)i_RANGE	Note
1	18	CPU3_PTAG	G2_1	DED	2	4	1	58	4	8080_H	SRAM_CLR
1	19	CPU3_DLMU_STBY	G2_1	SECDED	2	8	1	144	9	$BF80_H$	SRAM_CLR NDT
1	20	CPU4_DMEM	G1	SECDED	2	16	4	44	11	$9F80_H$	SRAM_CLR
1	21	CPU4_DTAG	G2_1	SECDED	2	4	1	64	4	8000_H	SRAM_CLR
1	22	CPU4_PMEM	G2_1	SECDED	1	8	2	72	9	$8B80_H$	SRAM_CLR
1	23	CPU4_PTAG	G2_1	DED	2	4	1	58	4	8080_H	SRAM_CLR
1	24	CPU4_DLMU_STBY	G2_1	SECDED	2	8	1	144	9	$BF80_H$	SRAM_CLR NDT
1	25	CPU5_DMEM	G1	SECDED	2	16	4	44	11	$9F80_H$	SRAM_CLR
1	26	CPU5_DTAG	G2_1	SECDED	2	4	1	64	4	8000_H	SRAM_CLR
1	27	CPU5_PMEM	G2_1	SECDED	1	8	2	72	9	$8B80_H$	SRAM_CLR
1	28	CPU5_PTAG	G2_1	DED	2	4	1	58	4	8080_H	SRAM_CLR
1	29	CPU5_DLMU_STBY	G2_1	SECDED	2	8	1	144	9	$BF80_H$	SRAM_CLR NDT
2	0	FSI0	N/A	SECDED	2	8	1	78	5	8780_H	**EXCLUDE FROM TEST**
2	1_CS	FSI1	N/A	SECDED	2	8	1	78	5	8780_H	**EXCLUDE FROM TEST**

(table continues...)

Table 1387 (continued) SSH instances

VMTx (x=)	MC(CS)i (i=)	SRAM Name	Group	ECC Type	ECC Granularity	MUX Factor	# of Towers	Tower width	# of RDBFLs	Reset-Val MC(CS)i_RANGE	Note
2	2	HSPHY_PCS0	N/A	NOECC	1	8	2	16	2	8780_H	**EXCLUDE FROM TEST**
2	3	HSPHY_PCS1	N/A	NOECC	1	8	2	16	2	8780_H	**EXCLUDE FROM TEST**
2	4	HSPHY_PCS2	N/A	NOECC	1	8	2	16	2	8780_H	**EXCLUDE FROM TEST**
2	5	PCIE0_DMA_RBUF	G2_2	SECDED	1	8	1	39	3	8180_H	Skip MBIST init
2	6	PCIE0_RX_DQ	G2_2	SECDED	1	4	1	46	3	8100_H	Skip MBIST init
2	7	PCIE0_RX_HQ	G2_2	SECDED	1	4	2	85	11	8000_H	Skip MBIST init
2	8	PCIE0_RETRY_BUF	G2_2	SECDED	1	4	1	47	3	8100_H	Skip MBIST init
2	9	PCIE0_CMPL_BUF	G2_2	SECDED	1	4	1	87	6	8000_H	Skip MBIST init
2	10	XGMAC_AXI	G2_1	SECDED	1	4	4	72	18	8380_H	SRAM_CLR
2	11	XGMAC_DMI	G2_2	SECDED	1	4	1	72	5	8080_H	Skip MBIST init
2	12	XGMAC0_GCL	G2_2	SECDED	1	8	1	39	3	8180_H	Skip MBIST init
2	13	XGMAC1_GCL	G2_2	SECDED	1	8	1	39	3	8180_H	Skip MBIST init

(table continues...)

Table 1387 (continued) SSH instances

VMTx (x=)	MC(CS)i (i=)	SRAM Name	Group	ECC Type	ECC Granularity	MUX Factor	# of Towers	Tower width	# of RDBFLs	Reset-Val MC(CS)i_RANGE	Note
2	14	XGMAC0_RX_EVEN	G2_2	SECDED	1	4	1	76	5	8380_H	Skip MBIST init
2	15	XGMAC0_RX_ODD	G2_2	SECDED	1	4	1	76	5	8380_H	Skip MBIST init
2	16	XGMAC1_RX_EVEN	G2_2	SECDED	1	4	1	76	5	8380_H	Skip MBIST init
2	17	XGMAC1_RX_ODD	G2_2	SECDED	1	4	1	76	5	8380_H	Skip MBIST init
2	18	XGMAC0_TX_EVEN	G2_2	SECDED	1	4	1	76	5	8380_H	Skip MBIST init
2	19	XGMAC0_TX_ODD	G2_2	SECDED	1	4	1	76	5	8380_H	Skip MBIST init
2	20	XGMAC1_TX_EVEN	G2_2	SECDED	1	4	1	76	5	8380_H	Skip MBIST init
2	21	XGMAC1_TX_ODD	G2_2	SECDED	1	4	1	76	5	8380_H	Skip MBIST init
2	22	PCIE1_DMA_RBUF	G2_2	SECDED	1	8	1	39	3	8180_H	Skip MBIST init
2	23	PCIE1_RX_DQ	G2_2	SECDED	1	4	1	46	3	8100_H	Skip MBIST init
2	24	PCIE1_RX_HQ	G2_2	SECDED	1	4	2	85	11	8000_H	Skip MBIST init
2	25	PCIE1_RETRY_BUF	G2_2	SECDED	1	4	1	47	3	8100_H	Skip MBIST init
2	26	PCIE1_CMPL_BUF	G2_2	SECDED	1	4	1	87	6	8000_H	Skip MBIST init

(table continues...)

Table 1387 (continued) SSH instances

VMTx (x=)	MC(CS)i (i=)	SRAM Name	Group	ECC Type	ECC Granularity	MUX Factor	# of Towers	Tower width	# of RDBFLs	Reset-Val MC(CS)i_RANGE	Note
3	0	PPU_IC_TAG	G2_2	SECDED	1	4	4	30	8	8000_H	Skip MBIST init
3	1	PPU_IC_DATA	G2_2	SECDED	1	4	4	80	20	8080_H	Skip MBIST init
3	2	PPU_DC_TAG	G2_2	SECDED	1	4	4	28	7	8000_H	Skip MBIST init
3	3	PPU_DC_DATA	G2_2	SECDED	1	4	4	80	20	8080_H	Skip MBIST init
3	4	PPU_BC	G2_2	NOECC	1	4	2	68	9	8080_H	Skip MBIST init
3	5	PPU_PT	G2_2	NOECC	1	4	2	16	2	8080_H	Skip MBIST init
3	6	PPU_RTT_INT	G2_2	NOECC	1	4	1	64	4	8180_H	Skip MBIST init
3	7	PPU_VEC_MEM0	G2_2	SECDED	1	8	8	24	12	8780_H	
3	8	PPU_VEC_MEM1	G2_2	SECDED	1	8	8	24	12	8780_H	
3	9	PPU_LLM_RAM_XL	G2_2	SECDED	1	8	8	73	37	$8F80_H$	
4	0_CS	CSRM_DMEM	G1	SECDED	2	16	4	44	11	$9F80_H$	SRAM_CLR
4	1_CS	CSRM_DTAG	G2_1	SECDED	2	4	1	64	4	8000_H	SRAM_CLR
4	2_CS	CSRM_PMEM	G2_1	SECDED	1	8	2	72	9	$8B80_H$	SRAM_CLR
4	3_CS	CSRM_PTAG	G2_1	DED	2	4	1	58	4	8080_H	SRAM_CLR

(table continues...)

Table 1387 (continued) SSH instances

VMTx (x=)	MC(CS)i (i=)	SRAM Name	Group	ECC Type	ECC Granularity	MUX Factor	# of Towers	Tower width	# of RDBFLs	Reset-Val MC(CS)i_RANGE	Note
4	5_CS	PKC	G2_1	SECDED	1	4	2	76	10	8180_H	SRAM_CLR
4	6	CDSP0_DATA	G2_2	SECDED	1	4	2	39	5	8100_H	
4	7	CDSP0_PROG	G2_1	SECDED	2	4	2	78	10	8080_H	SRAM_CLR
5	0	CANXL	G2_1	SECDED	1	8	1	39	3	8780_H	SRAM_CLR
5	1	DRE_RAM	G2_1	SECDED	1	8	1	72	5	8780_H	SRAM_CLR
5	2	MCAN10	G2_1	SECDED	1	16	1	39	3	9180_H	SRAM_CLR
5	3	MCAN20	G2_1	SECDED	1	16	1	39	3	8980_H	SRAM_CLR
5	4	MCAN21	G2_1	SECDED	1	16	1	39	3	8980_H	SRAM_CLR
5	5	MCAN22	G2_1	SECDED	1	16	1	39	3	8980_H	SRAM_CLR
5	6	MCAN23	G2_1	SECDED	1	16	1	39	3	8980_H	SRAM_CLR
5	7	SDMMC	G2_1	SECDED	1	4	1	39	3	8000_H	SRAM_CLR
5	8	LETH0_TX_FIFO	G2_1	SECDED	1	4	2	42	6	8380_H	SRAM_CLR
5	9	LETH1_TX_FIFO	G2_1	SECDED	1	4	2	42	6	8380_H	SRAM_CLR
5	10	LETH2_TX_FIFO	G2_1	SECDED	1	4	2	42	6	8380_H	SRAM_CLR
5	11	LETH3_TX_FIFO	G2_1	SECDED	1	4	2	42	6	8380_H	SRAM_CLR
5	12	LETH0_RX_FIFO	G2_1	SECDED	1	4	2	42	6	8180_H	SRAM_CLR

(table continues...)

Table 1387 (continued) SSH instances

VMTx (x=)	MC(CS)i (i=)	SRAM Name	Group	ECC Type	ECC Granularity	MUX Factor	# of Towers	Tower width	# of RDBFLs	Reset-Val MC(CS)i_RANGE	Note
5	13	LETH1_RX_FIFO	G2_1	SECDED	1	4	2	42	6	8180_H	SRAM_CLR
5	14	LETH2_RX_FIFO	G2_1	SECDED	1	4	2	42	6	8180_H	SRAM_CLR
5	15	LETH3_RX_FIFO	G2_1	SECDED	1	4	2	42	6	8180_H	SRAM_CLR
5	16	LETH0_RX_PARSER	G2_1	SECDED	1	4	2	56	7	8000_H	SRAM_CLR
5	17	LETH1_RX_PARSER	G2_1	SECDED	1	4	2	56	7	8000_H	SRAM_CLR
5	18	LETH2_RX_PARSER	G2_1	SECDED	1	4	2	56	7	8000_H	SRAM_CLR
5	19	LETH3_RX_PARSER	G2_1	SECDED	1	4	2	56	7	8000_H	SRAM_CLR
5	20	LETH0_GCL	G2_1	SECDED	1	4	1	36	3	8000_H	SRAM_CLR
5	21	LETH1_GCL	G2_1	SECDED	1	4	1	36	3	8000_H	SRAM_CLR
5	22	LETH2_GCL	G2_1	SECDED	1	4	1	36	3	8000_H	SRAM_CLR
5	23	LETH3_GCL	G2_1	SECDED	1	4	1	36	3	8000_H	SRAM_CLR
6	0	SCR_XRAM_L	N/A	SECDED	1	8	1	56	4	$8F80_H$	**EXCLUDE FROM TEST**

***Note**: Always refer to the device specific safety manual to find out which functional block SRAMs need to be considered for destructive MBIST*

34.8.1.3 SSH functionality exceptions

Depending on the SSH configuration, some functionalities may be or not available. The followingfunction s might be affected:

- SRAM Clear (MCONTROL.SRAM_CLR)
- Non-Destructive Test (NDT)
- Restricted error tracking:
  - No alarm and error indication in case of single (SBERR) errors from SSH
  - Tracking of double (DBERR) errors is not supported

***Note**: In AURIX™ 3rd Generation the default memory test algorithm is the destructive MBIST. The Non-Destructive MBIST is still available but with limited diagnostic capabilities and only for some memories of the MCU.*

The table "SSH instance exceptions" gives an overview about the functionality exceptions.

***Note**: For all SSHs listed in the table "SSH instance exceptions" there is no ECC monitoring available.*

***Note**: The SRAM_CLR functionality shall not be used with the SSH instances specified with “SRAM clear = no”. If an SRAM_CLR operation is started on any of these memories, its content is filled with ECC-incorrect values which can then lead to ECC alarms when accessing them. These memories shall be initialized by the application software instead, for example with an initialization routine or a DMA request. For the software initialization, it is recommended to use the TriCore™ CPU instruction ST.D (Store Doubleword) to avoid potential traps and alarms due to ECC errors.*

Table 1388 SSH instance exceptions

VMTx(x=)	MCi(i=)	Module MCi	SRAM clear	NDT	DBERR tracking
0	21	MCDS4P_TBUF	no	no	no
0	22	MCDS4P_LUT	no	no	no
1	6	MCDS2P_TBUF	no	no	no
1	7	MCDS2P_LUT	no	no	no
2	2	HSPHY_PCS0	no	no	no
2	3	HSPHY_PCS1	no	no	no
2	4	HSPHY_PCS2	no	no	no
2	5	PCIE0_DMA_RBUF	no	no	no
2	6	PCIE0_RX_DQ	no	no	no
2	7	PCIE0_RX_HQ	no	no	no
2	8	PCIE0_RETRY_BUF	no	no	no
2	9	PCIE0_CMPL_BUF	no	no	no
2	11	XGMAC_DMI	no	no	no

(table continues...)

Table 1388 (continued) SSH instance exceptions

VMTx(x=)	MCi(i=)	Module MCi	SRAM clear	NDT	DBERR tracking
2	12	XGMAC0_GCL	no	no	no
2	13	XGMAC1_GCL	no	no	no
2	14	XGMAC0_RX_EVEN	no	no	no
2	15	XGMAC0_RX_ODD	no	no	no
2	16	XGMAC1_RX_EVEN	no	no	no
2	17	XGMAC1_RX_ODD	no	no	no
2	18	XGMAC0_TX_EVEN	no	no	no
2	19	XGMAC0_TX_ODD	no	no	no
2	20	XGMAC1_TX_EVEN	no	no	no
2	21	XGMAC1_TX_ODD	no	no	no
2	22	PCIE1_DMA_RBUF	no	no	no
2	23	PCIE1_RX_DQ	no	no	no
2	24	PCIE1_RX_HQ	no	no	no
2	25	PCIE1_RETRY_BUF	no	no	no
2	26	PCIE1_MCPL_BUF	no	no	no
3	0	PPU_IC_TAG	no	no	no
3	1	PPU_IC_DATA	no	no	no
3	2	PPU_DC_TAG	no	no	no
3	3	PPU_DC_DATA	no	no	no
3	4	PPU_BC	no	no	no
3	5	PPU_PT	no	no	no
3	6	PPU_RTT_INT	no	no	no
3	7	PPU_VEC_MEM0	no	no	no
3	8	PPU_VEC_MEM1	no	no	no
3	9	PPU_LLM_RAM_XL	no	no	no

(table continues...)

Table 1388 (continued) SSH instance exceptions

VMTx(x=)	MCi(i=)	Module MCi	SRAM clear	NDT	DBERR tracking
4	6	CDSP0_DATA	no	no	no
6	0	SCR_XRAM_L	no	no	no

34.8.1.4 MEMMAP configuration

Table 1389 MEMMAP configuration

VMTx (x=)	Bits	Field	Type	Description
0	0	CPU0_DCMAP	rwh	**CPU0 DCache Mapping** 0_B Normal cache function 1_B Memory-mapped
0	1	CPU0_DTMAP	rh	**CPU0 DTAG Mapping** Read only. Mirrors the state of CPU0_DCMAP. CPU D-cache memories can only be mapped simultaneously. 0_B Normal cache function 1_B Memory-mapped
0	2	CPU0_PCMAP	rwh	**CPU0 PCACHE Mapping** 0_B Normal cache function 1_B Memory-mapped
0	3	CPU0_PTMAP	rh	**CPU0 PTAG Mapping** Read only. Mirrors the state of CPU0_PCMAP. CPU P-cache memories can only be mapped simultaneously. 0_B Normal cache function 1_B Memory-mapped
0	4	Reserved	r	**Reserved** Read as 0; must be written as 0.
0	5	CPU1_DCMAP	rwh	**CPU1 DCache Mapping** 0_B Normal cache function 1_B Memory-mapped
0	6	CPU1_DTMAP	rh	**CPU1 DTAG Mapping** Read only. Mirrors the state of CPU1_DCMAP. CPU D-cache memories can only be mapped simultaneously. 0_B Normal cache function 1_B Memory-mapped
0	7	CPU1_PCMAP	rwh	**CPU1 PCACHE Mapping** 0_B Normal cache function 1_B Memory-mapped

(table continues...)

Table 1389 (continued) MEMMAP configuration

VMTx (x=)	Bits	Field	Type	Description
0	8	CPU1_PTMAP	rh	**CPU1 PTAG Mapping** Read only. Mirrors the state of CPU1_PCMAP. CPU P-cache memories can only be mapped simultaneously. 0_B Normal cache function 1_B Memory-mapped
0	9	Reserved	r	**Reserved** Read as 0; must be written as 0.
0	10	CPU2_DCMAP	rwh	**CPU2 DCache Mapping** 0_B Normal cache function 1_B Memory-mapped
0	11	CPU2_DTMAP	rh	**CPU2 DTAG Mapping** Read only. Mirrors the state of CPU2_DCMAP. CPU D-cache memories can only be mapped simultaneously. 0_B Normal cache function 1_B Memory-mapped
0	12	CPU2_PCMAP	rwh	**CPU2 PCACHE Mapping** 0_B Normal cache function 1_B Memory-mapped
0	13	CPU2_PTMAP	rh	**CPU2 PTAG Mapping** Read only. Mirrors the state of CPU2_PCMAP. CPU P-cache memories can only be mapped simultaneously. 0_B Normal cache function 1_B Memory-mapped
0	31:14	Reserved	r	**Reserved** Read as 0; must be written as 0.
1	15	CPU3_DCMAP	rwh	**CPU3 DCache Mapping** 0_B Normal cache function 1_B Memory-mapped
1	16	CPU3_DTMAP	rh	**CPU3 DTAG Mapping** Read only. Mirrors the state of CPU3_DCMAP. CPU D-cache memories can only be mapped simultaneously. 0_B Normal cache function 1_B Memory-mapped
1	17	CPU3_PCMAP	rwh	**CPU3 PCACHE Mapping** 0_B Normal cache function 1_B Memory-mapped

(table continues...)

Table 1389 (continued) MEMMAP configuration

VMTx (x=)	Bits	Field	Type	Description
1	18	CPU3_PTMAP	rh	**CPU3 PTAG Mapping** Read only. Mirrors the state of CPU3_PCMAP. CPU P-cache memories can only be mapped simultaneously. 0_B Normal cache function 1_B Memory-mapped
1	19	Reserved	r	**Reserved** Read as 0; must be written as 0.
1	20	CPU4_DCMAP	rwh	**CPU4 DCache Mapping** 0_B Normal cache function 1_B Memory-mapped
1	21	CPU4_DTMAP	rh	**CPU4 DTAG Mapping** Read only. Mirrors the state of CPU4_DCMAP. CPU D-cache memories can only be mapped simultaneously. 0_B Normal cache function 1_B Memory-mapped
1	22	CPU4_PCMAP	rwh	**CPU4 PCACHE Mapping** 0_B Normal cache function 1_B Memory-mapped
1	23	CPU4_PTMAP	rh	**CPU4 PTAG Mapping** Read only. Mirrors the state of CPU4_PCMAP. CPU P-cache memories can only be mapped simultaneously. 0_B Normal cache function 1_B Memory-mapped
1	24	Reserved	r	**Reserved** Read as 0; must be written as 0.
1	25	CPU5_DCMAP	rwh	**CPU5 DCache Mapping** 0_B Normal cache function 1_B Memory-mapped
1	26	CPU5_DTMAP	rh	**CPU5 DTAG Mapping** Read only. Mirrors the state of CPU5_DCMAP. CPU D-cache memories can only be mapped simultaneously. 0_B Normal cache function 1_B Memory-mapped
1	27	CPU5_PCMAP	rwh	**CPU5 PCACHE Mapping** 0_B Normal cache function 1_B Memory-mapped

(table continues...)

Table 1389 (continued) MEMMAP configuration

VMTx (x=)	Bits	Field	Type	Description
1	28	CPU5_PTMAP	rh	**CPU5 PTAG Mapping** Read only. Mirrors the state of CPU5_PCMAP. CPU P-cache memories can only be mapped simultaneously. 0_B Normal cache function 1_B Memory-mapped
1	14:0, 31:29	Reserved	r	**Reserved** Read as 0; must be written as 0.
2	31:0	Reserved	r	**Reserved** Read as 0; must be written as 0.
3	31:0	Reserved	r	**Reserved** Read as 0; must be written as 0.
4	31:0	Reserved	r	**Reserved** Read as 0; must be written as 0.
5	31:0	Reserved	r	**Reserved** Read as 0; must be written as 0.
6	31:0	Reserved	r	**Reserved** Read as 0; must be written as 0.

Table 1390 MEMMAPCS configuration

VMTx (x=)	Bits	Field	Type	Description
0	31:0	Reserved	r	**Reserved** Read as 0; must be written as 0.
1	31:0	Reserved	r	**Reserved** Read as 0; must be written as 0.
2	31:0	Reserved	r	**Reserved** Read as 0; must be written as 0.
3	31:0	Reserved	r	**Reserved** Read as 0; must be written as 0.
4	0	CSRM_DCMAP	rwh	**CSRM DCache Mapping** 0_B Normal cache function 1_B Memory-mapped

(table continues...)

Table 1390 (continued) MEMMAPCS configuration

4	1	CSRM_DTMAP	rh	**CSRM DTAG Mapping** Read only. Mirrors the state of CSRM_DCMAP. CPU D-cache memories can only be mapped simultaneously. 0_B Normal cache function 1_B Memory-mapped
4	2	CSRM_PCMAP	rwh	**CSRM PCACHE Mapping** 0_B Normal cache function 1_B Memory-mapped
4	3	CSRM_PTMAP	rh	**CSRM PTAG Mapping** Read only. Mirrors the state of CSRM_PCMAP. CPU P-cache memories can only be mapped simultaneously. 0_B Normal cache function 1_B Memory-mapped
4	31:4	Reserved	r	**Reserved** Read as 0; must be written as 0.
5	31:0	Reserved	r	**Reserved** Read as 0; must be written as 0.
6	31:0	Reserved	r	**Reserved** Read as 0; must be written as 0.

34.8.1.5 VMT security information

The following memories are considered security related and feature auto-initialization:

- CSS_KEYATTR
- CSS_KEYSTORE
- CSS_FIFOx (x = 0 or 1)
- CSS_CONTEXT
- PKC
- FSIx (x = 1)
- CSRM_DMEM
- CSRM_PMEM
- CSRM_DTAG
- CSRM_PTAG

The following memories are considered non-security related but are also featuring auto-initialization:

- CPUx_DMEM(x = 0 - 5) - Only the cache part of the memory is considered security related.
- CPUx_DTAG(x = 0 - 5)
- CPUx_PMEM(x = 0 - 5) - Only the cache part of the memory is considered security related.
- CPUx_PTAG(x = 0 - 5)
- FSIx (x = 0)

34.8.2 TC4Dx VMT features

There are no deviations from the generic specification.

34.8.3 TC4Dx VMT functional description

There are no deviations from the generic specification.

34.8.4 TC4Dx VMT registers

34.8.4.1 Register address space - VMT

Table 1391 Registers address space - VMT

Module	Base address	End address	Note
VMT0	$F0400000_H$	$F041FFFF_H$	FPI slave interface
VMT1	$F0420000_H$	$F043FFFF_H$	FPI slave interface
VMT2	$F0440000_H$	$F045FFFF_H$	FPI slave interface
VMT3	$F0460000_H$	$F047FFFF_H$	FPI slave interface
VMT4	$F0480000_H$	$F049FFFF_H$	FPI slave interface
VMT5	$F04A0000_H$	$F04BFFFF_H$	FPI slave interface
VMT6	$F04C0000_H$	$F04DFFFF_H$	FPI slave interface

34.8.4.2 Register overview - access mode glossary

Table 1392 Register overview - access mode glossary

Keyword	Description
CSP	Access protection using PROT register VMT0_PROTCSP or VMT1_PROTCSP or VMT2_PROTCSP or VMT3_PROTCSP or VMT4_PROTCSP or VMT5_PROTCSP or VMT6_PROTCSP .
E	Access protection using PROT register VMT0_PROTE or VMT1_PROTE or VMT2_PROTE or VMT3_PROTE or VMT4_PROTE or VMT5_PROTE or VMT6_PROTE .
SE	Access protection using PROT register VMT0_PROTSE or VMT1_PROTSE or VMT2_PROTSE or VMT3_PROTSE or VMT4_PROTSE or VMT5_PROTSE or VMT6_PROTSE .
APU-PCS	Protection group consisting of registers VMT0_ACCENCS_WRA , VMT0_ACCENCS_WRB , VMT0_ACCENCS_RDA , VMT0_ACCENCS_RDB , VMT0_ACCENCS_VM , VMT0_ACCENCS_PRS or VMT1_ACCENCS_WRA , VMT1_ACCENCS_WRB , VMT1_ACCENCS_RDA , VMT1_ACCENCS_RDB , VMT1_ACCENCS_VM , VMT1_ACCENCS_PRS or VMT2_ACCENCS_WRA , VMT2_ACCENCS_WRB , VMT2_ACCENCS_RDA , VMT2_ACCENCS_RDB , VMT2_ACCENCS_VM , VMT2_ACCENCS_PRS or VMT3_ACCENCS_WRA , VMT3_ACCENCS_WRB , VMT3_ACCENCS_RDA , VMT3_ACCENCS_RDB , VMT3_ACCENCS_VM , VMT3_ACCENCS_PRS or VMT4_ACCENCS_WRA , VMT4_ACCENCS_WRB , VMT4_ACCENCS_RDA , VMT4_ACCENCS_RDB , VMT4_ACCENCS_VM , VMT4_ACCENCS_PRS or VMT5_ACCENCS_WRA , VMT5_ACCENCS_WRB , VMT5_ACCENCS_RDA , VMT5_ACCENCS_RDB , VMT5_ACCENCS_VM , VMT5_ACCENCS_PRS or VMT6_ACCENCS_WRA , VMT6_ACCENCS_WRB , VMT6_ACCENCS_RDA , VMT6_ACCENCS_RDB , VMT6_ACCENCS_VM , VMT6_ACCENCS_PRS .
PCS	Access protection using *APU-PCS* registers.

(table continues...)

Table 1392 (continued) Register overview - access mode glossary

Keyword	Description
APU-PNOM	Protection group consisting of registers VMT0_ACCENNOM_WRA , VMT0_ACCENNOM_WRB , VMT0_ACCENNOM_RDA , VMT0_ACCENNOM_RDB , VMT0_ACCENNOM_VM , VMT0_ACCENNOM_PRS or VMT1_ACCENNOM_WRA , VMT1_ACCENNOM_WRB , VMT1_ACCENNOM_RDA , VMT1_ACCENNOM_RDB , VMT1_ACCENNOM_VM , VMT1_ACCENNOM_PRS or VMT2_ACCENNOM_WRA , VMT2_ACCENNOM_WRB , VMT2_ACCENNOM_RDA , VMT2_ACCENNOM_RDB , VMT2_ACCENNOM_VM , VMT2_ACCENNOM_PRS or VMT3_ACCENNOM_WRA , VMT3_ACCENNOM_WRB , VMT3_ACCENNOM_RDA , VMT3_ACCENNOM_RDB , VMT3_ACCENNOM_VM , VMT3_ACCENNOM_PRS or VMT4_ACCENNOM_WRA , VMT4_ACCENNOM_WRB , VMT4_ACCENNOM_RDA , VMT4_ACCENNOM_RDB , VMT4_ACCENNOM_VM , VMT4_ACCENNOM_PRS or VMT5_ACCENNOM_WRA , VMT5_ACCENNOM_WRB , VMT5_ACCENNOM_RDA , VMT5_ACCENNOM_RDB , VMT5_ACCENNOM_VM , VMT5_ACCENNOM_PRS or VMT6_ACCENNOM_WRA , VMT6_ACCENNOM_WRB , VMT6_ACCENNOM_RDA , VMT6_ACCENNOM_RDB , VMT6_ACCENNOM_VM , VMT6_ACCENNOM_PRS .
PNOM	Access protection using *APU-PNOM* registers.
BE	Always returns a Bus Error.
U	No access restrictions.
PROT	Access restrictions as defined in the PROT register access rules.
STP	Access only during start-up, before start-up protection is set (SMM_STCON.ST_PROT=1).
32	Access only when using 32-bit width.

34.8.4.3 Register overview - VMT0 (ascending offset address)

Table 1393 Register overview - VMT0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
VMT0_CLC	Clock control register	00000_H	PNOM	PNOM, SV, E	5766	5766
VMT0_ID	Module identification register	00008_H	PNOM	BE	PowerOn Reset	5767
VMT0_PROTE	PROT register endinit	00020_H	U	SV, PROT	Application Reset	5768
VMT0_PROTSE	PROT register safe endinit	00024_H	U	SV, PROT	Application Reset	5769
VMT0_PROTCSP	PROT register cyber-secure APU endinit	$0002C_H$	U	SV, PROT	Application Reset	5771
VMT0_ACCENNOM_WRA	ACCENNOM write access enable register A	00040_H	U	SE, SV	Application Reset	5773
VMT0_ACCENNOM_WRB	ACCENNOM write access enable register B	00044_H	U	SE, SV	Application Reset	5773

(table continues...)

Table 1393 (continued) Register overview - VMT0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
VMT0_ACCENNOM_RDA	ACCENNOM read access enable register A	00048_H	U	SE, SV	Application Reset	5774
VMT0_ACCENNOM_RDB	ACCENNOM read access enable register B	$0004C_H$	U	SE, SV	Application Reset	5774
VMT0_ACCENNOM_VM	ACCENNOM VM access enable register	00050_H	U	SE, SV	Application Reset	5775
VMT0_ACCENNOM_PRS	ACCENNOM PRS access enable register	00054_H	U	SE, SV	Application Reset	5776
VMT0_ACCENCS_WRA	ACCENCS write access enable register A	00060_H	U	CSP, SV	Application Reset	5776
VMT0_ACCENCS_WRB	ACCENCS write access enable register B	00064_H	U	CSP, SV	Application Reset	5777
VMT0_ACCENCS_RDA	ACCENCS read access enable register A	00068_H	U	CSP, SV	Application Reset	5777
VMT0_ACCENCS_RDB	ACCENCS read access enable register B	$0006C_H$	U	CSP, SV	Application Reset	5778
VMT0_ACCENCS_VM	ACCENCS VM access enable register	00070_H	U	CSP, SV	Application Reset	5778
VMT0_ACCENCS_PRS	ACCENCS PRS access enable register	00074_H	U	CSP, SV	Application Reset	5779
VMT0_MEMTEST	Nominal memory MBIST enable register	00080_H	PNOM	SV, PNOM	Application Reset	5780
VMT0_MEMMAP	Nominal memory mapping enable register	00084_H	PNOM	SV, PNOM	Application Reset	5780
VMT0_MEMSTAT	Nominal memory status register	00088_H	PNOM	BE	Application Reset	5780
VMT0_MEMDONE	Nominal memory test done status register	$0008C_H$	PNOM	BE	Application Reset	5781
VMT0_MEMFDA	Nominal memory test FDA status register	00090_H	PNOM	BE	Application Reset	5782
VMT0_CEALARM	Nominal SSH CE alarm register	$000A4_H$	PNOM	BE	Application Reset	5782
VMT0_UCEALARM	Nominal SSH UCE alarm register	$000A8_H$	PNOM	BE	Application Reset	5783
VMT0_MEALARM	Nominal SSH ME alarm register	$000AC_H$	PNOM	BE	Application Reset	5783
VMT0_MEMTESTCS	Cyber-secure memory MBIST enable register	$000B4_H$	PCS	SV, PCS	Application Reset	5784
VMT0_MEMMAPCS	Cyber-secure memory mapping enable register	$000B8_H$	PCS	SV, PCS	Application Reset	5784
VMT0_MEMSTATCS	Cyber-secure memory status register	$000BC_H$	PCS	BE	Application Reset	5785

(table continues...)

Table 1393 (continued) Register overview - VMT0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
VMT0_MEMDONECS	Cyber-secure memory test done status register	$000C0_H$	PCS	BE	Application Reset	5785
VMT0_MEMFDACS	Cyber-secure memory test FDA status register	$000C4_H$	PCS	BE	Application Reset	5786
VMT0_CEALARMCS	Cyber-secure SSH CE alarm register	$000D8_H$	PCS	BE	Application Reset	5786
VMT0_UCEALARMCS	Cyber-secure SSH UCE alarm register	$000DC_H$	PCS	BE	Application Reset	5787
VMT0_MEALARMCS	Cyber-Secure SSH ME alarm register	$000E0_H$	PCS	BE	Application Reset	5788
VMT0_MCi_CONFIG0 (i=0-31)	MC i Nominal configuration register 0	01000_H+i*200_H	PNOM, 32	PNOM, 32	Application Reset	5788
VMT0_MCi_CONFIG1 (i=0-31)	MC i Nominal configuration register 1	01004_H+i*200_H	PNOM, 32	PNOM, 32	Application Reset	5789
VMT0_MCi_MCONTROL (i=0-31)	MC i Nominal MBIST control register	$0100C_H$+i*200_H	32, PNOM	SV, PNOM, 32	Application Reset	5791
VMT0_MCi_MSTATUS (i=0-31)	MC i Nominal status register	01010_H+i*200_H	32, PNOM	BE	Application Reset	5793
VMT0_MCi_RANGE (i=0-31)	MC i Nominal range register, single address mode	01014_H+i*200_H	PNOM, 32	PNOM, 32	Application Reset	5795
VMT0_MCi_REVID (i=0-31)	MC i Nominal revision ID register	$0101C_H$+i*200_H	32, PNOM	BE	Application Reset	5796
VMT0_MCi_ECCS (i=0-31)	MC i Nominal ECC safety register	01020_H+i*200_H	32, PNOM	SV, PNOM, 32	Application Reset	5797
VMT0_MCi_ECCD (i=0-31)	MC i Nominal Memory ECC detection register	01024_H+i*200_H	32, PNOM	SV, PNOM, 32	5799	5799
VMT0_MCi_ETRR (i=0-31)	MC i Nominal error tracking register	01028_H+i*200_H	32, PNOM	BE	PowerOn Reset	5801
VMT0_MCi_RDBFLy (i=0-31;y=0-79)	MC i Nominal read data and bit flip register y	01080_H+i*200_H+y*4	PNOM, 32	PNOM, 32	Application Reset	5801
VMT0_MCi_ALMSRCS (i=0-31)	MC i Nominal alarm sources configuration register	$011D0_H$+i*200_H	32, PNOM	SV, PNOM, 32	5802	5802

(table continues...)

Table 1393 (continued) Register overview - VMT0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
VMT0_MCi_FAULTSTS (i=0-31)	MC i Nominal SSH safety faults status register	$011D4_H$+ i*200_H	32, PNOM	SV, PNOM, 32	PowerOn Reset	5803
VMT0_MCi_ERRINFO (i=0-31)	MC i Nominal error information register	$011D8_H$+ i*200_H	32, PNOM	BE	PowerOn Reset	5805
VMT0_MCCSi_CONFIG0 (i=0-31)	MCCS i Cyber-secure configuration register 0	11000_H+i *200_H	PCS, 32	PCS, 32	Application Reset	5806
VMT0_MCCSi_CONFIG1 (i=0-31)	MCCS i Cyber-secure configuration register 1	11004_H+i *200_H	PCS, 32	PCS, 32	Application Reset	5806
VMT0_MCCSi_MCONTROL (i=0-31)	MCCS i Cyber-secure MBIST control register	$1100C_H$+ i*200_H	PCS, 32	SV, PCS, 32	Application Reset	5808
VMT0_MCCSi_MSTATUS (i=0-31)	MCCS i Cyber-secure status register	11010_H+i *200_H	32, PCS	BE	Application Reset	5810
VMT0_MCCSi_RANGE (i=0-31)	MCCS i Cyber-secure range register, single address mode	11014_H+i *200_H	PCS, 32	PCS, 32	Application Reset	5812
VMT0_MCCSi_REVID (i=0-31)	MCCS i Cyber-secure revision ID register	$1101C_H$+ i*200_H	32, PCS	BE	Application Reset	5813
VMT0_MCCSi_ECCS (i=0-31)	MCCS i Cyber-secure ECC safety register	11020_H+i *200_H	PCS, 32	SV, PCS, 32	Application Reset	5814
VMT0_MCCSi_ECCD (i=0-31)	MCCS i Cyber-secure memory ECC detection register	11024_H+i *200_H	PCS, 32	SV, PCS, 32	5816	5816
VMT0_MCCSi_ETRR (i=0-31)	MCCS i Cyber-secure error tracking register	11028_H+i *200_H	32, PCS	BE	PowerOn Reset	5818
VMT0_MCCSi_RDBFLy (i=0-31;y=0-79)	MCCS i Cyber-secure read data and bit flip register y	11080_H+i *200_H+y *4	PCS, 32	PCS, 32	Application Reset	5818
VMT0_MCCSi_ALMSRCS (i=0-31)	MCCS i Cyber-secure alarm sources configuration register	$111D0_H$+ i*200_H	PCS, 32	SV, PCS, 32	5819	5819
VMT0_MCCSi_FAULTSTS (i=0-31)	MCCS i Cyber-secure SSH safety faults status register	$111D4_H$+ i*200_H	PCS, 32	SV, PCS, 32	PowerOn Reset	5821

(table continues...)

Table 1393 (continued) Register overview - VMT0 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
VMT0_MCCSi_ERRINFO (i=0-31)	MCCS i Cyber-secure error information register	$111D8_H$+ i*200_H	32, PCS	BE	PowerOn Reset	5822

34.8.4.4 Register overview - VMT1 (ascending offset address)

Table 1394 Register overview - VMT1 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
VMT1_CLC	Clock control register	00000_H	PNOM	PNOM, SV, E	5766	5766
VMT1_ID	Module identification register	00008_H	PNOM	BE	PowerOn Reset	5767
VMT1_PROTE	PROT register endinit	00020_H	U	SV, PROT	Application Reset	5768
VMT1_PROTSE	PROT register safe endinit	00024_H	U	SV, PROT	Application Reset	5769
VMT1_PROTCSP	PROT register cyber-secure APU endinit	$0002C_H$	U	SV, PROT	Application Reset	5771
VMT1_ACCENNOM_WRA	ACCENNOM write access enable register A	00040_H	U	SE, SV	Application Reset	5773
VMT1_ACCENNOM_WRB	ACCENNOM write access enable register B	00044_H	U	SE, SV	Application Reset	5773
VMT1_ACCENNOM_RDA	ACCENNOM read access enable register A	00048_H	U	SE, SV	Application Reset	5774
VMT1_ACCENNOM_RDB	ACCENNOM read access enable register B	$0004C_H$	U	SE, SV	Application Reset	5774
VMT1_ACCENNOM_VM	ACCENNOM VM access enable register	00050_H	U	SE, SV	Application Reset	5775
VMT1_ACCENNOM_PRS	ACCENNOM PRS access enable register	00054_H	U	SE, SV	Application Reset	5776
VMT1_ACCENCS_WRA	ACCENCS write access enable register A	00060_H	U	CSP, SV	Application Reset	5776
VMT1_ACCENCS_WRB	ACCENCS write access enable register B	00064_H	U	CSP, SV	Application Reset	5777
VMT1_ACCENCS_RDA	ACCENCS read access enable register A	00068_H	U	CSP, SV	Application Reset	5777
VMT1_ACCENCS_RDB	ACCENCS read access enable register B	$0006C_H$	U	CSP, SV	Application Reset	5778
VMT1_ACCENCS_VM	ACCENCS VM access enable register	00070_H	U	CSP, SV	Application Reset	5778

(table continues...)

Table 1394 (continued) Register overview - VMT1 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
VMT1_ACCENCS_PRS	ACCENCS PRS access enable register	00074_H	U	CSP, SV	Application Reset	5779
VMT1_MEMTEST	Nominal memory MBIST enable register	00080_H	PNOM	SV, PNOM	Application Reset	5780
VMT1_MEMMAP	Nominal memory mapping enable register	00084_H	PNOM	SV, PNOM	Application Reset	5780
VMT1_MEMSTAT	Nominal memory status register	00088_H	PNOM	BE	Application Reset	5780
VMT1_MEMDONE	Nominal memory test done status register	$0008C_H$	PNOM	BE	Application Reset	5781
VMT1_MEMFDA	Nominal memory test FDA status register	00090_H	PNOM	BE	Application Reset	5782
VMT1_CEALARM	Nominal SSH CE alarm register	$000A4_H$	PNOM	BE	Application Reset	5782
VMT1_UCEALARM	Nominal SSH UCE alarm register	$000A8_H$	PNOM	BE	Application Reset	5783
VMT1_MEALARM	Nominal SSH ME alarm register	$000AC_H$	PNOM	BE	Application Reset	5783
VMT1_MEMTESTCS	Cyber-secure memory MBIST enable register	$000B4_H$	PCS	SV, PCS	Application Reset	5784
VMT1_MEMMAPCS	Cyber-secure memory mapping enable register	$000B8_H$	PCS	SV, PCS	Application Reset	5784
VMT1_MEMSTATCS	Cyber-secure memory status register	$000BC_H$	PCS	BE	Application Reset	5785
VMT1_MEMDONECS	Cyber-secure memory test done status register	$000C0_H$	PCS	BE	Application Reset	5785
VMT1_MEMFDACS	Cyber-secure memory test FDA status register	$000C4_H$	PCS	BE	Application Reset	5786
VMT1_CEALARMCS	Cyber-secure SSH CE alarm register	$000D8_H$	PCS	BE	Application Reset	5786
VMT1_UCEALARMCS	Cyber-secure SSH UCE alarm register	$000DC_H$	PCS	BE	Application Reset	5787
VMT1_MEALARMCS	Cyber-Secure SSH ME alarm register	$000E0_H$	PCS	BE	Application Reset	5788
VMT1_MCi_CONFIG0 (i=0-31)	MC i Nominal configuration register 0	01000_H+i*200_H	PNOM, 32	PNOM, 32	Application Reset	5788
VMT1_MCi_CONFIG1 (i=0-31)	MC i Nominal configuration register 1	01004_H+i*200_H	PNOM, 32	PNOM, 32	Application Reset	5789

(table continues...)

Table 1394 (continued) Register overview - VMT1 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
VMT1_MCi_MCONTROL (i=0-31)	MC i Nominal MBIST control register	$0100C_H$+i*200_H	32, PNOM	SV, PNOM, 32	Application Reset	5791
VMT1_MCi_MSTATUS (i=0-31)	MC i Nominal status register	01010_H+i*200_H	32, PNOM	BE	Application Reset	5793
VMT1_MCi_RANGE (i=0-31)	MC i Nominal range register, single address mode	01014_H+i*200_H	PNOM, 32	PNOM, 32	Application Reset	5795
VMT1_MCi_REVID (i=0-31)	MC i Nominal revision ID register	$0101C_H$+i*200_H	32, PNOM	BE	Application Reset	5796
VMT1_MCi_ECCS (i=0-31)	MC i Nominal ECC safety register	01020_H+i*200_H	32, PNOM	SV, PNOM, 32	Application Reset	5797
VMT1_MCi_ECCD (i=0-31)	MC i Nominal Memory ECC detection register	01024_H+i*200_H	32, PNOM	SV, PNOM, 32	5799	5799
VMT1_MCi_ETRR (i=0-31)	MC i Nominal error tracking register	01028_H+i*200_H	32, PNOM	BE	PowerOn Reset	5801
VMT1_MCi_RDBFLy (i=0-31;y=0-79)	MC i Nominal read data and bit flip register y	01080_H+i*200_H+y*4	PNOM, 32	PNOM, 32	Application Reset	5801
VMT1_MCi_ALMSRCS (i=0-31)	MC i Nominal alarm sources configuration register	$011D0_H$+i*200_H	32, PNOM	SV, PNOM, 32	5802	5802
VMT1_MCi_FAULTSTS (i=0-31)	MC i Nominal SSH safety faults status register	$011D4_H$+i*200_H	32, PNOM	SV, PNOM, 32	PowerOn Reset	5803
VMT1_MCi_ERRINFO (i=0-31)	MC i Nominal error information register	$011D8_H$+i*200_H	32, PNOM	BE	PowerOn Reset	5805
VMT1_MCCSi_CONFIG0 (i=0-31)	MCCS i Cyber-secure configuration register 0	11000_H+i*200_H	PCS, 32	PCS, 32	Application Reset	5806
VMT1_MCCSi_CONFIG1 (i=0-31)	MCCS i Cyber-secure configuration register 1	11004_H+i*200_H	PCS, 32	PCS, 32	Application Reset	5806
VMT1_MCCSi_MCONTROL (i=0-31)	MCCS i Cyber-secure MBIST control register	$1100C_H$+i*200_H	PCS, 32	SV, PCS, 32	Application Reset	5808

(table continues...)

Table 1394 (continued) Register overview - VMT1 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
VMT1_MCCSi_MSTATUS (i=0-31)	MCCS i Cyber-secure status register	11010_H+i*200_H	32, PCS	BE	Application Reset	5810
VMT1_MCCSi_RANGE (i=0-31)	MCCS i Cyber-secure range register, single address mode	11014_H+i*200_H	PCS, 32	PCS, 32	Application Reset	5812
VMT1_MCCSi_REVID (i=0-31)	MCCS i Cyber-secure revision ID register	$1101C_H$+i*200_H	32, PCS	BE	Application Reset	5813
VMT1_MCCSi_ECCS (i=0-31)	MCCS i Cyber-secure ECC safety register	11020_H+i*200_H	PCS, 32	SV, PCS, 32	Application Reset	5814
VMT1_MCCSi_ECCD (i=0-31)	MCCS i Cyber-secure memory ECC detection register	11024_H+i*200_H	PCS, 32	SV, PCS, 32	5816	5816
VMT1_MCCSi_ETRR (i=0-31)	MCCS i Cyber-secure error tracking register	11028_H+i*200_H	32, PCS	BE	PowerOn Reset	5818
VMT1_MCCSi_RDBFLy (i=0-31;y=0-79)	MCCS i Cyber-secure read data and bit flip register y	11080_H+i*200_H+y*4	PCS, 32	PCS, 32	Application Reset	5818
VMT1_MCCSi_ALMSRCS (i=0-31)	MCCS i Cyber-secure alarm sources configuration register	$111D0_H$+i*200_H	PCS, 32	SV, PCS, 32	5819	5819
VMT1_MCCSi_FAULTSTS (i=0-31)	MCCS i Cyber-secure SSH safety faults status register	$111D4_H$+i*200_H	PCS, 32	SV, PCS, 32	PowerOn Reset	5821
VMT1_MCCSi_ERRINFO (i=0-31)	MCCS i Cyber-secure error information register	$111D8_H$+i*200_H	32, PCS	BE	PowerOn Reset	5822

34.8.4.5 Register overview - VMT2 (ascending offset address)

Table 1395 Register overview - VMT2 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
VMT2_CLC	Clock control register	00000_H	PNOM	PNOM, SV, E	5766	5766
VMT2_ID	Module identification register	00008_H	PNOM	BE	PowerOn Reset	5767
VMT2_PROTE	PROT register endinit	00020_H	U	SV, PROT	Application Reset	5768

(table continues...)

Table 1395 (continued) Register overview - VMT2 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
VMT2_PROTSE	PROT register safe endinit	00024_H	U	SV, PROT	Application Reset	5769
VMT2_PROTCSP	PROT register cyber-secure APU endinit	$0002C_H$	U	SV, PROT	Application Reset	5771
VMT2_ACCENNOM_WRA	ACCENNOM write access enable register A	00040_H	U	SE, SV	Application Reset	5773
VMT2_ACCENNOM_WRB	ACCENNOM write access enable register B	00044_H	U	SE, SV	Application Reset	5773
VMT2_ACCENNOM_RDA	ACCENNOM read access enable register A	00048_H	U	SE, SV	Application Reset	5774
VMT2_ACCENNOM_RDB	ACCENNOM read access enable register B	$0004C_H$	U	SE, SV	Application Reset	5774
VMT2_ACCENNOM_VM	ACCENNOM VM access enable register	00050_H	U	SE, SV	Application Reset	5775
VMT2_ACCENNOM_PRS	ACCENNOM PRS access enable register	00054_H	U	SE, SV	Application Reset	5776
VMT2_ACCENCS_WRA	ACCENCS write access enable register A	00060_H	U	CSP, SV	Application Reset	5776
VMT2_ACCENCS_WRB	ACCENCS write access enable register B	00064_H	U	CSP, SV	Application Reset	5777
VMT2_ACCENCS_RDA	ACCENCS read access enable register A	00068_H	U	CSP, SV	Application Reset	5777
VMT2_ACCENCS_RDB	ACCENCS read access enable register B	$0006C_H$	U	CSP, SV	Application Reset	5778
VMT2_ACCENCS_VM	ACCENCS VM access enable register	00070_H	U	CSP, SV	Application Reset	5778
VMT2_ACCENCS_PRS	ACCENCS PRS access enable register	00074_H	U	CSP, SV	Application Reset	5779
VMT2_MEMTEST	Nominal memory MBIST enable register	00080_H	PNOM	SV, PNOM	Application Reset	5780
VMT2_MEMMAP	Nominal memory mapping enable register	00084_H	PNOM	SV, PNOM	Application Reset	5780
VMT2_MEMSTAT	Nominal memory status register	00088_H	PNOM	BE	Application Reset	5780
VMT2_MEMDONE	Nominal memory test done status register	$0008C_H$	PNOM	BE	Application Reset	5781
VMT2_MEMFDA	Nominal memory test FDA status register	00090_H	PNOM	BE	Application Reset	5782
VMT2_CEALARM	Nominal SSH CE alarm register	$000A4_H$	PNOM	BE	Application Reset	5782
VMT2_UCEALARM	Nominal SSH UCE alarm register	$000A8_H$	PNOM	BE	Application Reset	5783

(table continues...)

Table 1395 (continued) Register overview - VMT2 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
VMT2_MEALARM	Nominal SSH ME alarm register	$000AC_H$	PNOM	BE	Application Reset	5783
VMT2_MEMTESTCS	Cyber-secure memory MBIST enable register	$000B4_H$	PCS	SV, PCS	Application Reset	5784
VMT2_MEMMAPCS	Cyber-secure memory mapping enable register	$000B8_H$	PCS	SV, PCS	Application Reset	5784
VMT2_MEMSTATCS	Cyber-secure memory status register	$000BC_H$	PCS	BE	Application Reset	5785
VMT2_MEMDONECS	Cyber-secure memory test done status register	$000C0_H$	PCS	BE	Application Reset	5785
VMT2_MEMFDACS	Cyber-secure memory test FDA status register	$000C4_H$	PCS	BE	Application Reset	5786
VMT2_CEALARMCS	Cyber-secure SSH CE alarm register	$000D8_H$	PCS	BE	Application Reset	5786
VMT2_UCEALARMCS	Cyber-secure SSH UCE alarm register	$000DC_H$	PCS	BE	Application Reset	5787
VMT2_MEALARMCS	Cyber-Secure SSH ME alarm register	$000E0_H$	PCS	BE	Application Reset	5788
VMT2_MCi_CONFIG0 (i=0-31)	MC i Nominal configuration register 0	01000_H+i*200_H	PNOM, 32	PNOM, 32	Application Reset	5788
VMT2_MCi_CONFIG1 (i=0-31)	MC i Nominal configuration register 1	01004_H+i*200_H	PNOM, 32	PNOM, 32	Application Reset	5789
VMT2_MCi_MCONTROL (i=0-31)	MC i Nominal MBIST control register	$0100C_H$+i*200_H	32, PNOM	SV, PNOM, 32	Application Reset	5791
VMT2_MCi_MSTATUS (i=0-31)	MC i Nominal status register	01010_H+i*200_H	32, PNOM	BE	Application Reset	5793
VMT2_MCi_RANGE (i=0-31)	MC i Nominal range register, single address mode	01014_H+i*200_H	PNOM, 32	PNOM, 32	Application Reset	5795
VMT2_MCi_REVID (i=0-31)	MC i Nominal revision ID register	$0101C_H$+i*200_H	32, PNOM	BE	Application Reset	5796
VMT2_MCi_ECCS (i=0-31)	MC i Nominal ECC safety register	01020_H+i*200_H	32, PNOM	SV, PNOM, 32	Application Reset	5797
VMT2_MCi_ECCD (i=0-31)	MC i Nominal Memory ECC detection register	01024_H+i*200_H	32, PNOM	SV, PNOM, 32	5799	5799

(table continues...)

Table 1395 (continued) Register overview - VMT2 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
VMT2_MCi_ETRR (i=0-31)	MC i Nominal error tracking register	01028_H+i *200_H	32, PNOM	BE	PowerOn Reset	5801
VMT2_MCi_RDBFLy (i=0-31;y=0-79)	MC i Nominal read data and bit flip register y	01080_H+i *200_H+y *4	PNOM, 32	PNOM, 32	Application Reset	5801
VMT2_MCi_ALMSRCS (i=0-31)	MC i Nominal alarm sources configuration register	$011D0_H$+ i*200_H	32, PNOM	SV, PNOM, 32	5802	5802
VMT2_MCi_FAULTSTS (i=0-31)	MC i Nominal SSH safety faults status register	$011D4_H$+ i*200_H	32, PNOM	SV, PNOM, 32	PowerOn Reset	5803
VMT2_MCi_ERRINFO (i=0-31)	MC i Nominal error information register	$011D8_H$+ i*200_H	32, PNOM	BE	PowerOn Reset	5805
VMT2_MCCSi_CONFIG0 (i=0-31)	MCCS i Cyber-secure configuration register 0	11000_H+i *200_H	PCS, 32	PCS, 32	Application Reset	5806
VMT2_MCCSi_CONFIG1 (i=0-31)	MCCS i Cyber-secure configuration register 1	11004_H+i *200_H	PCS, 32	PCS, 32	Application Reset	5806
VMT2_MCCSi_MCONTROL (i=0-31)	MCCS i Cyber-secure MBIST control register	$1100C_H$+ i*200_H	PCS, 32	SV, PCS, 32	Application Reset	5808
VMT2_MCCSi_MSTATUS (i=0-31)	MCCS i Cyber-secure status register	11010_H+i *200_H	32, PCS	BE	Application Reset	5810
VMT2_MCCSi_RANGE (i=0-31)	MCCS i Cyber-secure range register, single address mode	11014_H+i *200_H	PCS, 32	PCS, 32	Application Reset	5812
VMT2_MCCSi_REVID (i=0-31)	MCCS i Cyber-secure revision ID register	$1101C_H$+ i*200_H	32, PCS	BE	Application Reset	5813
VMT2_MCCSi_ECCS (i=0-31)	MCCS i Cyber-secure ECC safety register	11020_H+i *200_H	PCS, 32	SV, PCS, 32	Application Reset	5814
VMT2_MCCSi_ECCD (i=0-31)	MCCS i Cyber-secure memory ECC detection register	11024_H+i *200_H	PCS, 32	SV, PCS, 32	5816	5816
VMT2_MCCSi_ETRR (i=0-31)	MCCS i Cyber-secure error tracking register	11028_H+i *200_H	32, PCS	BE	PowerOn Reset	5818

(table continues...)

Table 1395 (continued) Register overview - VMT2 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
VMT2_MCCSi_RDBFLy (i=0-31;y=0-79)	MCCS i Cyber-secure read data and bit flip register y	11080_H+i*200_H+y*4	PCS, 32	PCS, 32	Application Reset	5818
VMT2_MCCSi_ALMSRCS (i=0-31)	MCCS i Cyber-secure alarm sources configuration register	$111D0_H$+i*200_H	PCS, 32	SV, PCS, 32	5819	5819
VMT2_MCCSi_FAULTSTS (i=0-31)	MCCS i Cyber-secure SSH safety faults status register	$111D4_H$+i*200_H	PCS, 32	SV, PCS, 32	PowerOn Reset	5821
VMT2_MCCSi_ERRINFO (i=0-31)	MCCS i Cyber-secure error information register	$111D8_H$+i*200_H	32, PCS	BE	PowerOn Reset	5822

34.8.4.6 Register overview - VMT3 (ascending offset address)

Table 1396 Register overview - VMT3 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
VMT3_CLC	Clock control register	00000_H	PNOM	PNOM, SV, E	5766	5766
VMT3_ID	Module identification register	00008_H	PNOM	BE	PowerOn Reset	5767
VMT3_PROTE	PROT register endinit	00020_H	U	SV, PROT	Application Reset	5768
VMT3_PROTSE	PROT register safe endinit	00024_H	U	SV, PROT	Application Reset	5769
VMT3_PROTCSP	PROT register cyber-secure APU endinit	$0002C_H$	U	SV, PROT	Application Reset	5771
VMT3_ACCENNOM_WRA	ACCENNOM write access enable register A	00040_H	U	SE, SV	Application Reset	5773
VMT3_ACCENNOM_WRB	ACCENNOM write access enable register B	00044_H	U	SE, SV	Application Reset	5773
VMT3_ACCENNOM_RDA	ACCENNOM read access enable register A	00048_H	U	SE, SV	Application Reset	5774
VMT3_ACCENNOM_RDB	ACCENNOM read access enable register B	$0004C_H$	U	SE, SV	Application Reset	5774
VMT3_ACCENNOM_VM	ACCENNOM VM access enable register	00050_H	U	SE, SV	Application Reset	5775
VMT3_ACCENNOM_PRS	ACCENNOM PRS access enable register	00054_H	U	SE, SV	Application Reset	5776
VMT3_ACCENCS_WRA	ACCENCS write access enable register A	00060_H	U	CSP, SV	Application Reset	5776

(table continues...)

Table 1396 (continued) Register overview - VMT3 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
VMT3_ACCENCS_WRB	ACCENCS write access enable register B	00064_H	U	CSP, SV	Application Reset	5777
VMT3_ACCENCS_RDA	ACCENCS read access enable register A	00068_H	U	CSP, SV	Application Reset	5777
VMT3_ACCENCS_RDB	ACCENCS read access enable register B	$0006C_H$	U	CSP, SV	Application Reset	5778
VMT3_ACCENCS_VM	ACCENCS VM access enable register	00070_H	U	CSP, SV	Application Reset	5778
VMT3_ACCENCS_PRS	ACCENCS PRS access enable register	00074_H	U	CSP, SV	Application Reset	5779
VMT3_MEMTEST	Nominal memory MBIST enable register	00080_H	PNOM	SV, PNOM	Application Reset	5780
VMT3_MEMMAP	Nominal memory mapping enable register	00084_H	PNOM	SV, PNOM	Application Reset	5780
VMT3_MEMSTAT	Nominal memory status register	00088_H	PNOM	BE	Application Reset	5780
VMT3_MEMDONE	Nominal memory test done status register	$0008C_H$	PNOM	BE	Application Reset	5781
VMT3_MEMFDA	Nominal memory test FDA status register	00090_H	PNOM	BE	Application Reset	5782
VMT3_CEALARM	Nominal SSH CE alarm register	$000A4_H$	PNOM	BE	Application Reset	5782
VMT3_UCEALARM	Nominal SSH UCE alarm register	$000A8_H$	PNOM	BE	Application Reset	5783
VMT3_MEALARM	Nominal SSH ME alarm register	$000AC_H$	PNOM	BE	Application Reset	5783
VMT3_MEMTESTCS	Cyber-secure memory MBIST enable register	$000B4_H$	PCS	SV, PCS	Application Reset	5784
VMT3_MEMMAPCS	Cyber-secure memory mapping enable register	$000B8_H$	PCS	SV, PCS	Application Reset	5784
VMT3_MEMSTATCS	Cyber-secure memory status register	$000BC_H$	PCS	BE	Application Reset	5785
VMT3_MEMDONECS	Cyber-secure memory test done status register	$000C0_H$	PCS	BE	Application Reset	5785
VMT3_MEMFDACS	Cyber-secure memory test FDA status register	$000C4_H$	PCS	BE	Application Reset	5786
VMT3_CEALARMCS	Cyber-secure SSH CE alarm register	$000D8_H$	PCS	BE	Application Reset	5786
VMT3_UCEALARMCS	Cyber-secure SSH UCE alarm register	$000DC_H$	PCS	BE	Application Reset	5787
VMT3_MEALARMCS	Cyber-Secure SSH ME alarm register	$000E0_H$	PCS	BE	Application Reset	5788

(table continues...)

Table 1396 (continued) Register overview - VMT3 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
VMT3_MCi_CONFIG0 (i=0-31)	MC i Nominal configuration register 0	01000_H+i *200_H	PNOM, 32	PNOM, 32	Application Reset	5788
VMT3_MCi_CONFIG1 (i=0-31)	MC i Nominal configuration register 1	01004_H+i *200_H	PNOM, 32	PNOM, 32	Application Reset	5789
VMT3_MCi_MCONTROL (i=0-31)	MC i Nominal MBIST control register	$0100C_H$+ i*200_H	32, PNOM	SV, PNOM, 32	Application Reset	5791
VMT3_MCi_MSTATUS (i=0-31)	MC i Nominal status register	01010_H+i *200_H	32, PNOM	BE	Application Reset	5793
VMT3_MCi_RANGE (i=0-31)	MC i Nominal range register, single address mode	01014_H+i *200_H	PNOM, 32	PNOM, 32	Application Reset	5795
VMT3_MCi_REVID (i=0-31)	MC i Nominal revision ID register	$0101C_H$+ i*200_H	32, PNOM	BE	Application Reset	5796
VMT3_MCi_ECCS (i=0-31)	MC i Nominal ECC safety register	01020_H+i *200_H	32, PNOM	SV, PNOM, 32	Application Reset	5797
VMT3_MCi_ECCD (i=0-31)	MC i Nominal Memory ECC detection register	01024_H+i *200_H	32, PNOM	SV, PNOM, 32	5799	5799
VMT3_MCi_ETRR (i=0-31)	MC i Nominal error tracking register	01028_H+i *200_H	32, PNOM	BE	PowerOn Reset	5801
VMT3_MCi_RDBFLy (i=0-31;y=0-79)	MC i Nominal read data and bit flip register y	01080_H+i *200_H+y *4	PNOM, 32	PNOM, 32	Application Reset	5801
VMT3_MCi_ALMSRCS (i=0-31)	MC i Nominal alarm sources configuration register	$011D0_H$+ i*200_H	32, PNOM	SV, PNOM, 32	5802	5802
VMT3_MCi_FAULTSTS (i=0-31)	MC i Nominal SSH safety faults status register	$011D4_H$+ i*200_H	32, PNOM	SV, PNOM, 32	PowerOn Reset	5803
VMT3_MCi_ERRINFO (i=0-31)	MC i Nominal error information register	$011D8_H$+ i*200_H	32, PNOM	BE	PowerOn Reset	5805
VMT3_MCCSi_CONFIG0 (i=0-31)	MCCS i Cyber-secure configuration register 0	11000_H+i *200_H	PCS, 32	PCS, 32	Application Reset	5806

(table continues...)

Table 1396 (continued) Register overview - VMT3 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
VMT3_MCCSi_CONFIG1 (i=0-31)	MCCS i Cyber-secure configuration register 1	11004_H+i*200_H	PCS, 32	PCS, 32	Application Reset	5806
VMT3_MCCSi_MCONTROL (i=0-31)	MCCS i Cyber-secure MBIST control register	$1100C_H$+i*200_H	PCS, 32	SV, PCS, 32	Application Reset	5808
VMT3_MCCSi_MSTATUS (i=0-31)	MCCS i Cyber-secure status register	11010_H+i*200_H	32, PCS	BE	Application Reset	5810
VMT3_MCCSi_RANGE (i=0-31)	MCCS i Cyber-secure range register, single address mode	11014_H+i*200_H	PCS, 32	PCS, 32	Application Reset	5812
VMT3_MCCSi_REVID (i=0-31)	MCCS i Cyber-secure revision ID register	$1101C_H$+i*200_H	32, PCS	BE	Application Reset	5813
VMT3_MCCSi_ECCS (i=0-31)	MCCS i Cyber-secure ECC safety register	11020_H+i*200_H	PCS, 32	SV, PCS, 32	Application Reset	5814
VMT3_MCCSi_ECCD (i=0-31)	MCCS i Cyber-secure memory ECC detection register	11024_H+i*200_H	PCS, 32	SV, PCS, 32	5816	5816
VMT3_MCCSi_ETRR (i=0-31)	MCCS i Cyber-secure error tracking register	11028_H+i*200_H	32, PCS	BE	PowerOn Reset	5818
VMT3_MCCSi_RDBFLy (i=0-31;y=0-79)	MCCS i Cyber-secure read data and bit flip register y	11080_H+i*200_H+y*4	PCS, 32	PCS, 32	Application Reset	5818
VMT3_MCCSi_ALMSRCS (i=0-31)	MCCS i Cyber-secure alarm sources configuration register	$111D0_H$+i*200_H	PCS, 32	SV, PCS, 32	5819	5819
VMT3_MCCSi_FAULTSTS (i=0-31)	MCCS i Cyber-secure SSH safety faults status register	$111D4_H$+i*200_H	PCS, 32	SV, PCS, 32	PowerOn Reset	5821
VMT3_MCCSi_ERRINFO (i=0-31)	MCCS i Cyber-secure error information register	$111D8_H$+i*200_H	32, PCS	BE	PowerOn Reset	5822

34.8.4.7 Register overview - VMT4 (ascending offset address)

Table 1397 Register overview - VMT4 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
VMT4_CLC	Clock control register	00000_H	PNOM	PNOM, SV, E	5766	5766
VMT4_ID	Module identification register	00008_H	PNOM	BE	PowerOn Reset	5767
VMT4_PROTE	PROT register endinit	00020_H	U	SV, PROT	Application Reset	5768
VMT4_PROTSE	PROT register safe endinit	00024_H	U	SV, PROT	Application Reset	5769
VMT4_PROTCSP	PROT register cyber-secure APU endinit	$0002C_H$	U	SV, PROT	Application Reset	5771
VMT4_ACCENNOM_WRA	ACCENNOM write access enable register A	00040_H	U	SE, SV	Application Reset	5773
VMT4_ACCENNOM_WRB	ACCENNOM write access enable register B	00044_H	U	SE, SV	Application Reset	5773
VMT4_ACCENNOM_RDA	ACCENNOM read access enable register A	00048_H	U	SE, SV	Application Reset	5774
VMT4_ACCENNOM_RDB	ACCENNOM read access enable register B	$0004C_H$	U	SE, SV	Application Reset	5774
VMT4_ACCENNOM_VM	ACCENNOM VM access enable register	00050_H	U	SE, SV	Application Reset	5775
VMT4_ACCENNOM_PRS	ACCENNOM PRS access enable register	00054_H	U	SE, SV	Application Reset	5776
VMT4_ACCENCS_WRA	ACCENCS write access enable register A	00060_H	U	CSP, SV	Application Reset	5776
VMT4_ACCENCS_WRB	ACCENCS write access enable register B	00064_H	U	CSP, SV	Application Reset	5777
VMT4_ACCENCS_RDA	ACCENCS read access enable register A	00068_H	U	CSP, SV	Application Reset	5777
VMT4_ACCENCS_RDB	ACCENCS read access enable register B	$0006C_H$	U	CSP, SV	Application Reset	5778
VMT4_ACCENCS_VM	ACCENCS VM access enable register	00070_H	U	CSP, SV	Application Reset	5778
VMT4_ACCENCS_PRS	ACCENCS PRS access enable register	00074_H	U	CSP, SV	Application Reset	5779
VMT4_MEMTEST	Nominal memory MBIST enable register	00080_H	PNOM	SV, PNOM	Application Reset	5780
VMT4_MEMMAP	Nominal memory mapping enable register	00084_H	PNOM	SV, PNOM	Application Reset	5780
VMT4_MEMSTAT	Nominal memory status register	00088_H	PNOM	BE	Application Reset	5780

(table continues...)

Table 1397 (continued) Register overview - VMT4 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
VMT4_MEMDONE	Nominal memory test done status register	$0008C_H$	PNOM	BE	Application Reset	5781
VMT4_MEMFDA	Nominal memory test FDA status register	00090_H	PNOM	BE	Application Reset	5782
VMT4_CEALARM	Nominal SSH CE alarm register	$000A4_H$	PNOM	BE	Application Reset	5782
VMT4_UCEALARM	Nominal SSH UCE alarm register	$000A8_H$	PNOM	BE	Application Reset	5783
VMT4_MEALARM	Nominal SSH ME alarm register	$000AC_H$	PNOM	BE	Application Reset	5783
VMT4_MEMTESTCS	Cyber-secure memory MBIST enable register	$000B4_H$	PCS	SV, PCS	Application Reset	5784
VMT4_MEMMAPCS	Cyber-secure memory mapping enable register	$000B8_H$	PCS	SV, PCS	Application Reset	5784
VMT4_MEMSTATCS	Cyber-secure memory status register	$000BC_H$	PCS	BE	Application Reset	5785
VMT4_MEMDONECS	Cyber-secure memory test done status register	$000C0_H$	PCS	BE	Application Reset	5785
VMT4_MEMFDACS	Cyber-secure memory test FDA status register	$000C4_H$	PCS	BE	Application Reset	5786
VMT4_CEALARMCS	Cyber-secure SSH CE alarm register	$000D8_H$	PCS	BE	Application Reset	5786
VMT4_UCEALARMCS	Cyber-secure SSH UCE alarm register	$000DC_H$	PCS	BE	Application Reset	5787
VMT4_MEALARMCS	Cyber-Secure SSH ME alarm register	$000E0_H$	PCS	BE	Application Reset	5788
VMT4_MCi_CONFIG0 (i=0-31)	MC i Nominal configuration register 0	01000_H+i *200_H	PNOM, 32	PNOM, 32	Application Reset	5788
VMT4_MCi_CONFIG1 (i=0-31)	MC i Nominal configuration register 1	01004_H+i *200_H	PNOM, 32	PNOM, 32	Application Reset	5789
VMT4_MCi_MCONTROL (i=0-31)	MC i Nominal MBIST control register	$0100C_H$+ i*200_H	32, PNOM	SV, PNOM, 32	Application Reset	5791
VMT4_MCi_MSTATUS (i=0-31)	MC i Nominal status register	01010_H+i *200_H	32, PNOM	BE	Application Reset	5793
VMT4_MCi_RANGE (i=0-31)	MC i Nominal range register, single address mode	01014_H+i *200_H	PNOM, 32	PNOM, 32	Application Reset	5795

(table continues...)

Table 1397 (continued) Register overview - VMT4 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
VMT4_MCi_REVID (i=0-31)	MC i Nominal revision ID register	$0101C_H$+i*200_H	32, PNOM	BE	Application Reset	5796
VMT4_MCi_ECCS (i=0-31)	MC i Nominal ECC safety register	01020_H+i*200_H	32, PNOM	SV, PNOM, 32	Application Reset	5797
VMT4_MCi_ECCD (i=0-31)	MC i Nominal Memory ECC detection register	01024_H+i*200_H	32, PNOM	SV, PNOM, 32	5799	5799
VMT4_MCi_ETRR (i=0-31)	MC i Nominal error tracking register	01028_H+i*200_H	32, PNOM	BE	PowerOn Reset	5801
VMT4_MCi_RDBFLy (i=0-31;y=0-79)	MC i Nominal read data and bit flip register y	01080_H+i*200_H+y*4	PNOM, 32	PNOM, 32	Application Reset	5801
VMT4_MCi_ALMSRCS (i=0-31)	MC i Nominal alarm sources configuration register	$011D0_H$+i*200_H	32, PNOM	SV, PNOM, 32	5802	5802
VMT4_MCi_FAULTSTS (i=0-31)	MC i Nominal SSH safety faults status register	$011D4_H$+i*200_H	32, PNOM	SV, PNOM, 32	PowerOn Reset	5803
VMT4_MCi_ERRINFO (i=0-31)	MC i Nominal error information register	$011D8_H$+i*200_H	32, PNOM	BE	PowerOn Reset	5805
VMT4_MCCSi_CONFIG0 (i=0-31)	MCCS i Cyber-secure configuration register 0	11000_H+i*200_H	PCS, 32	PCS, 32	Application Reset	5806
VMT4_MCCSi_CONFIG1 (i=0-31)	MCCS i Cyber-secure configuration register 1	11004_H+i*200_H	PCS, 32	PCS, 32	Application Reset	5806
VMT4_MCCSi_MCONTROL (i=0-31)	MCCS i Cyber-secure MBIST control register	$1100C_H$+i*200_H	PCS, 32	SV, PCS, 32	Application Reset	5808
VMT4_MCCSi_MSTATUS (i=0-31)	MCCS i Cyber-secure status register	11010_H+i*200_H	32, PCS	BE	Application Reset	5810
VMT4_MCCSi_RANGE (i=0-31)	MCCS i Cyber-secure range register, single address mode	11014_H+i*200_H	PCS, 32	PCS, 32	Application Reset	5812
VMT4_MCCSi_REVID (i=0-31)	MCCS i Cyber-secure revision ID register	$1101C_H$+i*200_H	32, PCS	BE	Application Reset	5813

(table continues...)

Table 1397 (continued) Register overview - VMT4 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
VMT4_MCCSi_ECCS (i=0-31)	MCCS i Cyber-secure ECC safety register	11020_H+i*200_H	PCS, 32	SV, PCS, 32	Application Reset	5814
VMT4_MCCSi_ECCD (i=0-31)	MCCS i Cyber-secure memory ECC detection register	11024_H+i*200_H	PCS, 32	SV, PCS, 32	5816	5816
VMT4_MCCSi_ETRR (i=0-31)	MCCS i Cyber-secure error tracking register	11028_H+i*200_H	32, PCS	BE	PowerOn Reset	5818
VMT4_MCCSi_RDBFLy (i=0-31;y=0-79)	MCCS i Cyber-secure read data and bit flip register y	11080_H+i*200_H+y*4	PCS, 32	PCS, 32	Application Reset	5818
VMT4_MCCSi_ALMSRCS (i=0-31)	MCCS i Cyber-secure alarm sources configuration register	$111D0_H$+i*200_H	PCS, 32	SV, PCS, 32	5819	5819
VMT4_MCCSi_FAULTSTS (i=0-31)	MCCS i Cyber-secure SSH safety faults status register	$111D4_H$+i*200_H	PCS, 32	SV, PCS, 32	PowerOn Reset	5821
VMT4_MCCSi_ERRINFO (i=0-31)	MCCS i Cyber-secure error information register	$111D8_H$+i*200_H	32, PCS	BE	PowerOn Reset	5822

34.8.4.8 Register overview - VMT5 (ascending offset address)

Table 1398 Register overview - VMT5 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
VMT5_CLC	Clock control register	00000_H	PNOM	PNOM, SV, E	5766	5766
VMT5_ID	Module identification register	00008_H	PNOM	BE	PowerOn Reset	5767
VMT5_PROTE	PROT register endinit	00020_H	U	SV, PROT	Application Reset	5768
VMT5_PROTSE	PROT register safe endinit	00024_H	U	SV, PROT	Application Reset	5769
VMT5_PROTCSP	PROT register cyber-secure APU endinit	$0002C_H$	U	SV, PROT	Application Reset	5771
VMT5_ACCENNOM_WRA	ACCENNOM write access enable register A	00040_H	U	SE, SV	Application Reset	5773
VMT5_ACCENNOM_WRB	ACCENNOM write access enable register B	00044_H	U	SE, SV	Application Reset	5773

(table continues...)

Table 1398 (continued) Register overview - VMT5 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
VMT5_ACCENNOM_RDA	ACCENNOM read access enable register A	00048_H	U	SE, SV	Application Reset	5774
VMT5_ACCENNOM_RDB	ACCENNOM read access enable register B	$0004C_H$	U	SE, SV	Application Reset	5774
VMT5_ACCENNOM_VM	ACCENNOM VM access enable register	00050_H	U	SE, SV	Application Reset	5775
VMT5_ACCENNOM_PRS	ACCENNOM PRS access enable register	00054_H	U	SE, SV	Application Reset	5776
VMT5_ACCENCS_WRA	ACCENCS write access enable register A	00060_H	U	CSP, SV	Application Reset	5776
VMT5_ACCENCS_WRB	ACCENCS write access enable register B	00064_H	U	CSP, SV	Application Reset	5777
VMT5_ACCENCS_RDA	ACCENCS read access enable register A	00068_H	U	CSP, SV	Application Reset	5777
VMT5_ACCENCS_RDB	ACCENCS read access enable register B	$0006C_H$	U	CSP, SV	Application Reset	5778
VMT5_ACCENCS_VM	ACCENCS VM access enable register	00070_H	U	CSP, SV	Application Reset	5778
VMT5_ACCENCS_PRS	ACCENCS PRS access enable register	00074_H	U	CSP, SV	Application Reset	5779
VMT5_MEMTEST	Nominal memory MBIST enable register	00080_H	PNOM	SV, PNOM	Application Reset	5780
VMT5_MEMMAP	Nominal memory mapping enable register	00084_H	PNOM	SV, PNOM	Application Reset	5780
VMT5_MEMSTAT	Nominal memory status register	00088_H	PNOM	BE	Application Reset	5780
VMT5_MEMDONE	Nominal memory test done status register	$0008C_H$	PNOM	BE	Application Reset	5781
VMT5_MEMFDA	Nominal memory test FDA status register	00090_H	PNOM	BE	Application Reset	5782
VMT5_CEALARM	Nominal SSH CE alarm register	$000A4_H$	PNOM	BE	Application Reset	5782
VMT5_UCEALARM	Nominal SSH UCE alarm register	$000A8_H$	PNOM	BE	Application Reset	5783
VMT5_MEALARM	Nominal SSH ME alarm register	$000AC_H$	PNOM	BE	Application Reset	5783
VMT5_MEMTESTCS	Cyber-secure memory MBIST enable register	$000B4_H$	PCS	SV, PCS	Application Reset	5784
VMT5_MEMMAPCS	Cyber-secure memory mapping enable register	$000B8_H$	PCS	SV, PCS	Application Reset	5784
VMT5_MEMSTATCS	Cyber-secure memory status register	$000BC_H$	PCS	BE	Application Reset	5785

(table continues...)

Table 1398 (continued) Register overview - VMT5 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
VMT5_MEMDONECS	Cyber-secure memory test done status register	$000C0_H$	PCS	BE	Application Reset	5785
VMT5_MEMFDACS	Cyber-secure memory test FDA status register	$000C4_H$	PCS	BE	Application Reset	5786
VMT5_CEALARMCS	Cyber-secure SSH CE alarm register	$000D8_H$	PCS	BE	Application Reset	5786
VMT5_UCEALARMCS	Cyber-secure SSH UCE alarm register	$000DC_H$	PCS	BE	Application Reset	5787
VMT5_MEALARMCS	Cyber-Secure SSH ME alarm register	$000E0_H$	PCS	BE	Application Reset	5788
VMT5_MCi_CONFIG0 (i=0-31)	MC i Nominal configuration register 0	01000_H+i*200_H	PNOM, 32	PNOM, 32	Application Reset	5788
VMT5_MCi_CONFIG1 (i=0-31)	MC i Nominal configuration register 1	01004_H+i*200_H	PNOM, 32	PNOM, 32	Application Reset	5789
VMT5_MCi_MCONTROL (i=0-31)	MC i Nominal MBIST control register	$0100C_H$+i*200_H	32, PNOM	SV, PNOM, 32	Application Reset	5791
VMT5_MCi_MSTATUS (i=0-31)	MC i Nominal status register	01010_H+i*200_H	32, PNOM	BE	Application Reset	5793
VMT5_MCi_RANGE (i=0-31)	MC i Nominal range register, single address mode	01014_H+i*200_H	PNOM, 32	PNOM, 32	Application Reset	5795
VMT5_MCi_REVID (i=0-31)	MC i Nominal revision ID register	$0101C_H$+i*200_H	32, PNOM	BE	Application Reset	5796
VMT5_MCi_ECCS (i=0-31)	MC i Nominal ECC safety register	01020_H+i*200_H	32, PNOM	SV, PNOM, 32	Application Reset	5797
VMT5_MCi_ECCD (i=0-31)	MC i Nominal Memory ECC detection register	01024_H+i*200_H	32, PNOM	SV, PNOM, 32	5799	5799
VMT5_MCi_ETRR (i=0-31)	MC i Nominal error tracking register	01028_H+i*200_H	32, PNOM	BE	PowerOn Reset	5801
VMT5_MCi_RDBFLy (i=0-31;y=0-79)	MC i Nominal read data and bit flip register y	01080_H+i*200_H+y*4	PNOM, 32	PNOM, 32	Application Reset	5801
VMT5_MCi_ALMSRCS (i=0-31)	MC i Nominal alarm sources configuration register	$011D0_H$+i*200_H	32, PNOM	SV, PNOM, 32	5802	5802

(table continues...)

Table 1398 (continued) Register overview - VMT5 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
VMT5_MCi_FAULTSTS (i=0-31)	MC i Nominal SSH safety faults status register	$011D4_H$+ i*200_H	32, PNOM	SV, PNOM, 32	PowerOn Reset	5803
VMT5_MCi_ERRINFO (i=0-31)	MC i Nominal error information register	$011D8_H$+ i*200_H	32, PNOM	BE	PowerOn Reset	5805
VMT5_MCCSi_CONFIG0 (i=0-31)	MCCS i Cyber-secure configuration register 0	11000_H+i *200_H	PCS, 32	PCS, 32	Application Reset	5806
VMT5_MCCSi_CONFIG1 (i=0-31)	MCCS i Cyber-secure configuration register 1	11004_H+i *200_H	PCS, 32	PCS, 32	Application Reset	5806
VMT5_MCCSi_MCONTROL (i=0-31)	MCCS i Cyber-secure MBIST control register	$1100C_H$+ i*200_H	PCS, 32	SV, PCS, 32	Application Reset	5808
VMT5_MCCSi_MSTATUS (i=0-31)	MCCS i Cyber-secure status register	11010_H+i *200_H	32, PCS	BE	Application Reset	5810
VMT5_MCCSi_RANGE (i=0-31)	MCCS i Cyber-secure range register, single address mode	11014_H+i *200_H	PCS, 32	PCS, 32	Application Reset	5812
VMT5_MCCSi_REVID (i=0-31)	MCCS i Cyber-secure revision ID register	$1101C_H$+ i*200_H	32, PCS	BE	Application Reset	5813
VMT5_MCCSi_ECCS (i=0-31)	MCCS i Cyber-secure ECC safety register	11020_H+i *200_H	PCS, 32	SV, PCS, 32	Application Reset	5814
VMT5_MCCSi_ECCD (i=0-31)	MCCS i Cyber-secure memory ECC detection register	11024_H+i *200_H	PCS, 32	SV, PCS, 32	5816	5816
VMT5_MCCSi_ETRR (i=0-31)	MCCS i Cyber-secure error tracking register	11028_H+i *200_H	32, PCS	BE	PowerOn Reset	5818
VMT5_MCCSi_RDBFLy (i=0-31;y=0-79)	MCCS i Cyber-secure read data and bit flip register y	11080_H+i *200_H+y *4	PCS, 32	PCS, 32	Application Reset	5818
VMT5_MCCSi_ALMSRCS (i=0-31)	MCCS i Cyber-secure alarm sources configuration register	$111D0_H$+ i*200_H	PCS, 32	SV, PCS, 32	5819	5819
VMT5_MCCSi_FAULTSTS (i=0-31)	MCCS i Cyber-secure SSH safety faults status register	$111D4_H$+ i*200_H	PCS, 32	SV, PCS, 32	PowerOn Reset	5821

(table continues...)

Table 1398 (continued) Register overview - VMT5 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
VMT5_MCCSi_ERRINFO (i=0-31)	MCCS i Cyber-secure error information register	$111D8_H$+ i*200_H	32, PCS	BE	PowerOn Reset	5822

34.8.4.9 Register overview - VMT6 (ascending offset address)

Table 1399 Register overview - VMT6 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
VMT6_CLC	Clock control register	00000_H	PNOM	PNOM, SV, E	5766	5766
VMT6_ID	Module identification register	00008_H	PNOM	BE	PowerOn Reset	5767
VMT6_PROTE	PROT register endinit	00020_H	U	SV, PROT	Application Reset	5768
VMT6_PROTSE	PROT register safe endinit	00024_H	U	SV, PROT	Application Reset	5769
VMT6_PROTCSP	PROT register cyber-secure APU endinit	$0002C_H$	U	SV, PROT	Application Reset	5771
VMT6_ACCENNOM_WRA	ACCENNOM write access enable register A	00040_H	U	SE, SV	Application Reset	5773
VMT6_ACCENNOM_WRB	ACCENNOM write access enable register B	00044_H	U	SE, SV	Application Reset	5773
VMT6_ACCENNOM_RDA	ACCENNOM read access enable register A	00048_H	U	SE, SV	Application Reset	5774
VMT6_ACCENNOM_RDB	ACCENNOM read access enable register B	$0004C_H$	U	SE, SV	Application Reset	5774
VMT6_ACCENNOM_VM	ACCENNOM VM access enable register	00050_H	U	SE, SV	Application Reset	5775
VMT6_ACCENNOM_PRS	ACCENNOM PRS access enable register	00054_H	U	SE, SV	Application Reset	5776
VMT6_ACCENCS_WRA	ACCENCS write access enable register A	00060_H	U	CSP, SV	Application Reset	5776
VMT6_ACCENCS_WRB	ACCENCS write access enable register B	00064_H	U	CSP, SV	Application Reset	5777
VMT6_ACCENCS_RDA	ACCENCS read access enable register A	00068_H	U	CSP, SV	Application Reset	5777
VMT6_ACCENCS_RDB	ACCENCS read access enable register B	$0006C_H$	U	CSP, SV	Application Reset	5778
VMT6_ACCENCS_VM	ACCENCS VM access enable register	00070_H	U	CSP, SV	Application Reset	5778

(table continues...)

Table 1399 (continued) Register overview - VMT6 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
VMT6_ACCENCS_PRS	ACCENCS PRS access enable register	00074_H	U	CSP, SV	Application Reset	5779
VMT6_MEMTEST	Nominal memory MBIST enable register	00080_H	PNOM	SV, PNOM	Application Reset	5780
VMT6_MEMMAP	Nominal memory mapping enable register	00084_H	PNOM	SV, PNOM	Application Reset	5780
VMT6_MEMSTAT	Nominal memory status register	00088_H	PNOM	BE	Application Reset	5780
VMT6_MEMDONE	Nominal memory test done status register	$0008C_H$	PNOM	BE	Application Reset	5781
VMT6_MEMFDA	Nominal memory test FDA status register	00090_H	PNOM	BE	Application Reset	5782
VMT6_CEALARM	Nominal SSH CE alarm register	$000A4_H$	PNOM	BE	Application Reset	5782
VMT6_UCEALARM	Nominal SSH UCE alarm register	$000A8_H$	PNOM	BE	Application Reset	5783
VMT6_MEALARM	Nominal SSH ME alarm register	$000AC_H$	PNOM	BE	Application Reset	5783
VMT6_MEMTESTCS	Cyber-secure memory MBIST enable register	$000B4_H$	PCS	SV, PCS	Application Reset	5784
VMT6_MEMMAPCS	Cyber-secure memory mapping enable register	$000B8_H$	PCS	SV, PCS	Application Reset	5784
VMT6_MEMSTATCS	Cyber-secure memory status register	$000BC_H$	PCS	BE	Application Reset	5785
VMT6_MEMDONECS	Cyber-secure memory test done status register	$000C0_H$	PCS	BE	Application Reset	5785
VMT6_MEMFDACS	Cyber-secure memory test FDA status register	$000C4_H$	PCS	BE	Application Reset	5786
VMT6_CEALARMCS	Cyber-secure SSH CE alarm register	$000D8_H$	PCS	BE	Application Reset	5786
VMT6_UCEALARMCS	Cyber-secure SSH UCE alarm register	$000DC_H$	PCS	BE	Application Reset	5787
VMT6_MEALARMCS	Cyber-Secure SSH ME alarm register	$000E0_H$	PCS	BE	Application Reset	5788
VMT6_MCi_CONFIG0 (i=0-31)	MC i Nominal configuration register 0	01000_H+i*200_H	PNOM, 32	PNOM, 32	Application Reset	5788
VMT6_MCi_CONFIG1 (i=0-31)	MC i Nominal configuration register 1	01004_H+i*200_H	PNOM, 32	PNOM, 32	Application Reset	5789

(table continues...)

Table 1399 (continued) Register overview - VMT6 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
VMT6_MCi_MCONTROL (i=0-31)	MC i Nominal MBIST control register	$0100C_H$+i*200_H	32, PNOM	SV, PNOM, 32	Application Reset	5791
VMT6_MCi_MSTATUS (i=0-31)	MC i Nominal status register	01010_H+i*200_H	32, PNOM	BE	Application Reset	5793
VMT6_MCi_RANGE (i=0-31)	MC i Nominal range register, single address mode	01014_H+i*200_H	PNOM, 32	PNOM, 32	Application Reset	5795
VMT6_MCi_REVID (i=0-31)	MC i Nominal revision ID register	$0101C_H$+i*200_H	32, PNOM	BE	Application Reset	5796
VMT6_MCi_ECCS (i=0-31)	MC i Nominal ECC safety register	01020_H+i*200_H	32, PNOM	SV, PNOM, 32	Application Reset	5797
VMT6_MCi_ECCD (i=0-31)	MC i Nominal Memory ECC detection register	01024_H+i*200_H	32, PNOM	SV, PNOM, 32	5799	5799
VMT6_MCi_ETRR (i=0-31)	MC i Nominal error tracking register	01028_H+i*200_H	32, PNOM	BE	PowerOn Reset	5801
VMT6_MCi_RDBFLy (i=0-31;y=0-79)	MC i Nominal read data and bit flip register y	01080_H+i*200_H+y*4	PNOM, 32	PNOM, 32	Application Reset	5801
VMT6_MCi_ALMSRCS (i=0-31)	MC i Nominal alarm sources configuration register	$011D0_H$+i*200_H	32, PNOM	SV, PNOM, 32	5802	5802
VMT6_MCi_FAULTSTS (i=0-31)	MC i Nominal SSH safety faults status register	$011D4_H$+i*200_H	32, PNOM	SV, PNOM, 32	PowerOn Reset	5803
VMT6_MCi_ERRINFO (i=0-31)	MC i Nominal error information register	$011D8_H$+i*200_H	32, PNOM	BE	PowerOn Reset	5805
VMT6_MCCSi_CONFIG0 (i=0-31)	MCCS i Cyber-secure configuration register 0	11000_H+i*200_H	PCS, 32	PCS, 32	Application Reset	5806
VMT6_MCCSi_CONFIG1 (i=0-31)	MCCS i Cyber-secure configuration register 1	11004_H+i*200_H	PCS, 32	PCS, 32	Application Reset	5806
VMT6_MCCSi_MCONTROL (i=0-31)	MCCS i Cyber-secure MBIST control register	$1100C_H$+i*200_H	PCS, 32	SV, PCS, 32	Application Reset	5808

(table continues...)

Table 1399 (continued) Register overview - VMT6 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
VMT6_MCCSi_MSTATUS (i=0-31)	MCCS i Cyber-secure status register	11010_H+i*200_H	32, PCS	BE	Application Reset	5810
VMT6_MCCSi_RANGE (i=0-31)	MCCS i Cyber-secure range register, single address mode	11014_H+i*200_H	PCS, 32	PCS, 32	Application Reset	5812
VMT6_MCCSi_REVID (i=0-31)	MCCS i Cyber-secure revision ID register	$1101C_H$+i*200_H	32, PCS	BE	Application Reset	5813
VMT6_MCCSi_ECCS (i=0-31)	MCCS i Cyber-secure ECC safety register	11020_H+i*200_H	PCS, 32	SV, PCS, 32	Application Reset	5814
VMT6_MCCSi_ECCD (i=0-31)	MCCS i Cyber-secure memory ECC detection register	11024_H+i*200_H	PCS, 32	SV, PCS, 32	5816	5816
VMT6_MCCSi_ETRR (i=0-31)	MCCS i Cyber-secure error tracking register	11028_H+i*200_H	32, PCS	BE	PowerOn Reset	5818
VMT6_MCCSi_RDBFLy (i=0-31;y=0-79)	MCCS i Cyber-secure read data and bit flip register y	11080_H+i*200_H+y*4	PCS, 32	PCS, 32	Application Reset	5818
VMT6_MCCSi_ALMSRCS (i=0-31)	MCCS i Cyber-secure alarm sources configuration register	$111D0_H$+i*200_H	PCS, 32	SV, PCS, 32	5819	5819
VMT6_MCCSi_FAULTSTS (i=0-31)	MCCS i Cyber-secure SSH safety faults status register	$111D4_H$+i*200_H	PCS, 32	SV, PCS, 32	PowerOn Reset	5821
VMT6_MCCSi_ERRINFO (i=0-31)	MCCS i Cyber-secure error information register	$111D8_H$+i*200_H	32, PCS	BE	PowerOn Reset	5822

34.8.4.10 Device specific registers

There are no device specific register changes.

34.8.5 TC4Dx VMT connectivity

The table below lists interfaces to and from this functional block to other blocks in the device.

Table 1400 List of VMT interface signals

Interface signals	I/O	Description
VMT_SMU_ALARM[5:0]	Out	SSH Alarms
SMM_VMT_ApplicationReset	In	Application Reset
SMM_VMT_DebugReset	In	Debug Reset
CLOCK_VMT_fSPB	In	SPB clock input
VMT_IR_SR[6:0]	Out	Memory Test Done Interrupt

34.8.6 TC4Dx VMT revision history

Reference	Description of change(s)
Date range: 2024-08-17 to 2025-01-10	
–	• No visible changes

35 System Control Unit (SCU)

The System Control Unit (SCU) provides a collection of functions that are common for all CPUs like trap/NMI handling, global overlay control and software-over-the-air (SOTA) swap control, and various system functions that includes ERU cross-triggering, basic chip-information.

35.1 Feature list

- Trap or NMI (non-maskable interrupt) generation (TR)
- External request handling (ERU)
- CPU Global overlay control (OVC)
- SOTA control register (SOTACTRL)
- Chip information register (CHIPINFO)

35.2 Functional overview

SCU provides common CPU and system functions that include trap handling, global overlay control, SOTA swap control, ERU triggering and chip information. The main components of the SCU are illustrated in the following figure.

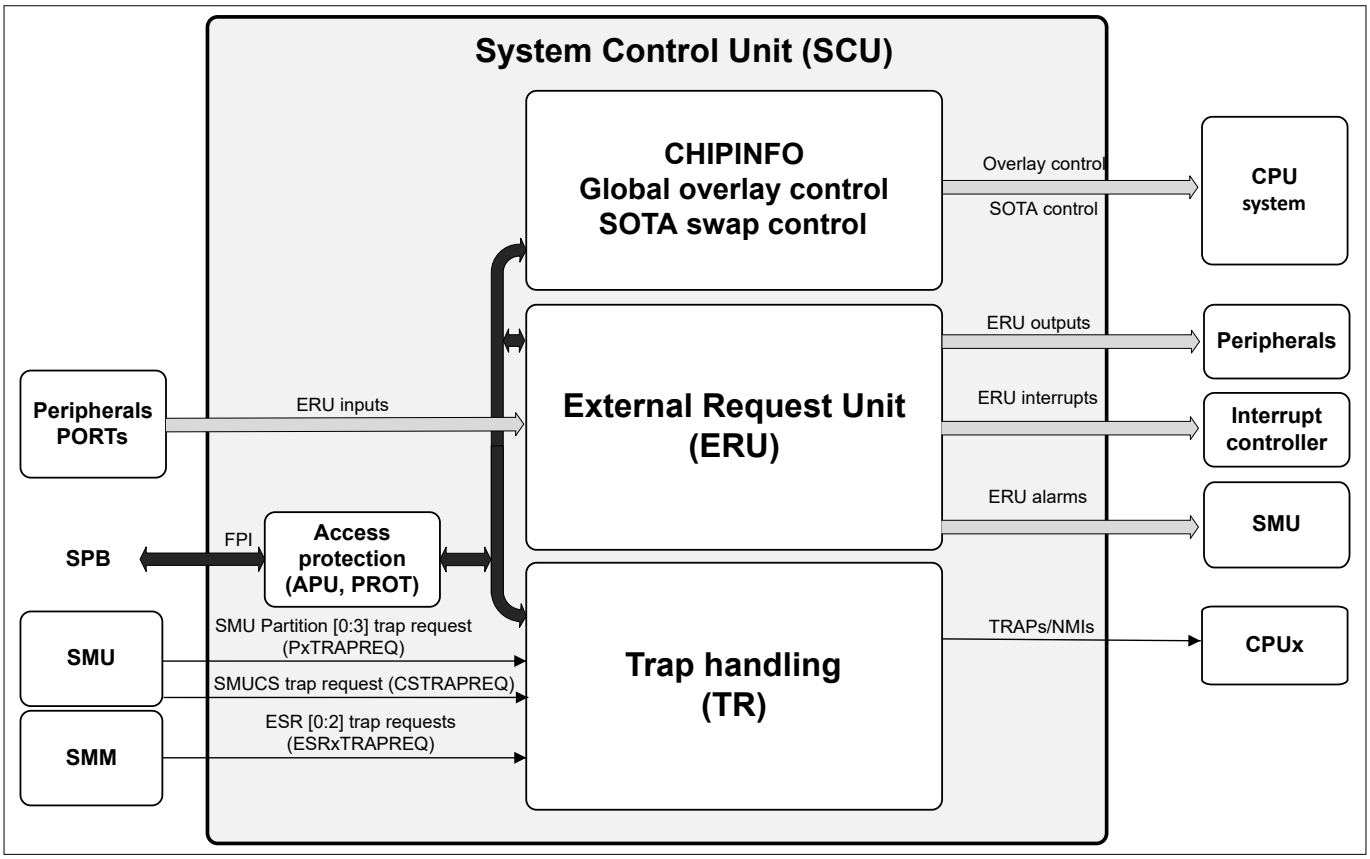


Figure 766 SCU block diagram

For the alarm description please refer to the alarm mapping tables in the SMU functional block and the Safety Manual.

Trap/NMI handling for both non-secure and secure CPUs is done inside TR block. ERU block provides cross triggering functionality, connecting several ports and peripherals as sources to interrupt controller, SMU, ADC, timers and other peripherals as destinations. Beside TR and ERU functions, SCU also includes several control registers for common CPU and system functionality: memory overlay, SOTA swap control and basic chip information.

35 System Control Unit (SCU)

Related information

TC4Dx SMU alarm mapping tables on page 7233

35.3 Functional description

Details on the SCU functionality are covered in following sub-sections.

35.3.1 Trap generation (TR)

The Trap generation function enables the application software to map different trap events from SMU, ESR-pins and SW to the NMI inputs of implemented CPUs. In addition, it provides special functions to separate the security and four non-security applications.

To set a trap-request flag in a STAT (status) register (TRAPS_CPU_STAT or TRAPS_CS_STAT), there shall be either a trap source event or intentional trap-set by software by writing to the corresponding bit-field in a SET register (TRAPS_CPU_SETCLRGRPg_SET or TRAPS_CS_SETCLR_SET). Each trap-request flag in a STAT register can be cleared by software by writing to the corresponding bit-field in a CLR register (TRAPS_CPU_SETCLRGRPg_CLR or TRAPS_CS_SETCLR_CLR).

Register bits in a DIS register (TRAPS_CPU_DISy, y=0-5, and TRAPS_CS_DIS) determine, for each CPU separately, which traps are forwarded as NMI to the respective CPU. If a trap source is disabled for a certain CPU, setting of a STAT flag will not trigger a trap-routine in that CPU.

***Note**: In case a certain CPU is not available in the device (smaller devices can have less than 6 CPU cores), its respective DIS register will still be there, but its function will be ignored.*

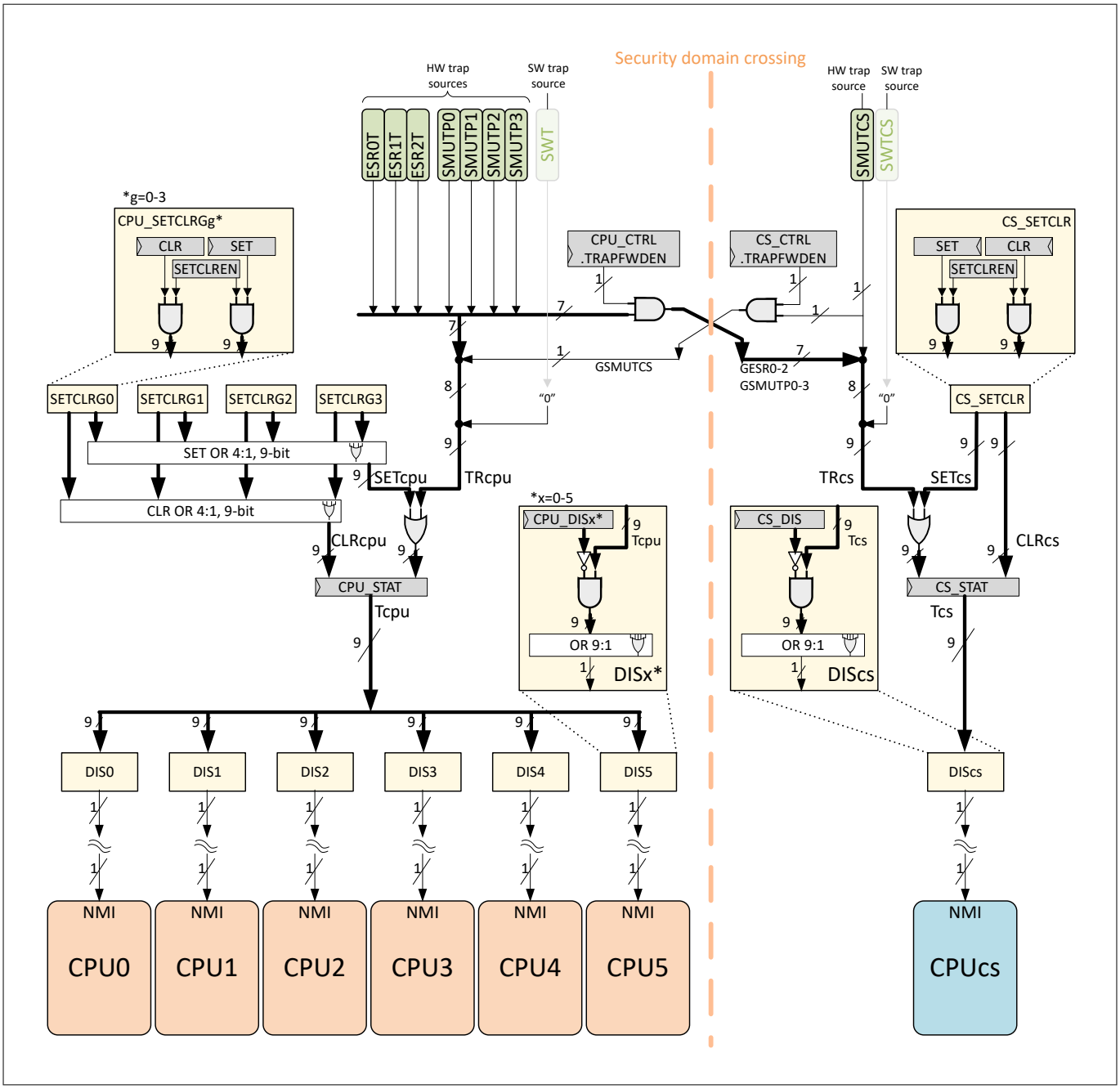


Figure 767 Trap generation

Trap sources

There are 8 hardware trap sources (also mentioned in figure as HW trap sources) and 1 dedicated SW trap source:

- SMU - 5 sources resulting in 5 STAT flags:
  - 4 from each SMU_SAFE partition - SMUTP0-3
  - 1 from SMU_CS partition - SMUTCS
- ESR - 3 sources resulting in 3 STAT flags:
  - ESR0T, ESR1T and ESR2T
- Software - 1 dedicated STAT flag:
  - SWT for non-security and SWTCS for security application

Trap-request from the SMU_CS partition is assigned to the security application.

Four SMU_SAFE sources and three ESR sources are assigned to the non-security application.

Both security and non-security applications can trigger a software trap by setting a dedicated SWTCS or SWT flag in the corresponding STAT register.

Trap control overview

- CPU traps may be triggered by hardware and software trap sources
- CPU trap-request events are captured in a status register (STAT)
  - There are separate status registers for:
    - The secured domain (TRAPS_CS_STAT)
    - The non-secured domain (TRAPS_CPU_STAT)
- It is possible to set or clear captured CPU trap-request flags by software, using SET and CLR registers
  - To achieve freedom from interference in between the secure application and the four non-secure application, control over SET and CLR functionality of each trap request source is configurable to up-to 4 non-secured partitions and 1 secured partition:
    - TRAPS_CPU: SETCLRGRPg_SET, SETCLRGRPg_CLR, SETCLRGRPg_SETCLREN registers
    - TRAPS_CS: SETCLR_SET, SETCLR_CLR, SETCLR_SETCLREN registers
- It is possible to disable or enable traps for each CPU individually using TRAPS_CPU_DISy and TRAPS_CS_DIS registers
  - The logical "OR"-combined resulting trap is routed to each CPU (CPU0-5, CPUcs) as an NMI input

Before enabling a trap source through the DIS register it is recommended to clear the trap request flag in the STAT register by writing to the respective CLR register. It is recommended to clear the trap-request flag in the STAT register by writing to the respective CLR register before a trap source is enabled in the DIS register. The trap-request flag could be set before the trap source is enabled, and simply enabling the trap source could result in an unintended CPU trap.

It is also recommended to clear the trap-request flag at the end of a trap handling routine.

35.3.1.1 Trap handling

Storing, setting and clearing traps

- For each trap source, there is a dedicated status bit in the STAT register, indicating if a trap request was received. A total of 9 bits are available in TRAPS_*_STAT registers (respectively TRAPS_CPU_STAT, TRAPS_CS_STAT)
- The SCU provides for each of the 5 partitions/groups, one security and four non-security groups, a dedicated access protection and dedicated SET, CLR, and SETCLREN registers.
  - SET and CLR registers are read and write protected by the corresponding group APU (ACCENGRPg (PGg) for non-security, ACCENCS (PCS) for security application)
  - SETCLREN registers are protected for write by PROTE and PROTCSE, and by the APU registers ACCENNOM (PNOM) and ACCENCS (PCS) for read

Enabling traps to be seen on NMI input of CPUs

- For each CPU (CPU0-5, CPUcs), there is a disable register (TRAPS_CPU_DISy, TRAPS_CS_DIS) to selectively mask trap requests.
- TRAPS_CPU_DIS0-5 registers for CPU0-5 are protected by PROTE for write and by general APU ACCENNOM (PNOM) for read
- TRAPS_CS_DIS register for CPUcs is protected by PROTCSE for write and by CS-APU ACCENCS (PCS) for read

Security considerations

Two trap domains are defined:

- A secured trap domain, containing the CPUcs and the related SMU_CS trap request (SMUTCS). This domain is controlled through TRAPS_CS_* registers
- A non-secured trap domain, containing the four non-secure partitions to control the seven non-secure trap requests to CPU0-5 (ESRyT, SMUTPx, y = 0-2, x = 0-3). This domain is controlled through TRAPS_CPU_* registers

For isolation between the two domains, a trap-request-forwarding gate is defined through the TRAPS_*_CTRL.TRAPFWDEN configuration bit (TRAPS_CPU_CTRL, TRAPS_CS_CTRL), allowing each domain to decide whether traps from that domain will be visible on the other domain. The default behavior is:

- After application reset, traps from the secured domain are not visible in the non-secured domain. Security software can decide to enable visibility
- After application reset, traps from the non-secure domain are not visible in the secured domain. Application software can decide to enable visibility

In addition to the gate, there is a dedicated DIS register (TRAPS_CPU_DIS0-5, TRAPS_CS_DIS) per CPU to enable or disable individual trap sources on a per CPU granularity.

Reset behavior

After application reset only CPU0 and CPUcs are in RUN mode. The default TRAPS behavior is:

- Both CPU_CTRL.TRAPFWDEN and CS_CTRL.TRAPFWDEN bits are disabled
- Access protections:
  - ACCENGRPg_{WR/RD}{A/B}: D and DS TAG IDs of CPU0 along with Cerberus are enabled for write access. All the master agent functions are enabled for read access.
  - ACCENNOM_{WR/RD}{A/B}: D and DS TAG IDs of CPU0 along with Cerberus are enabled for write access. All the master agent functions are enabled for read access.
  - ACCENCS_{WR/RD}{A/B}: Only D TAG ID of CPUcs is enabled for read and write, all other master agent functions are disabled for read and write
- SETCLR registers:
  - CPU_SETCLRGRPg: These registers enable SET and CLR of all 9 trap sources for the corresponding group
  - CS_SETCLR: The register enables SET and CLR of only SMUTCS and SWTCS trap sources
- Default values for DIS registers are:
  - CPU_DIS0: SMUTP0, SWT are enabled; other trap sources are disabled
  - CPU_DIS1-5: All trap sources are disabled
  - CS_DIS: SMUTCS and SWTCS are enabled; other trap sources are disabled

35.3.2 External request unit (ERU)

The external request unit (ERU) is a versatile event and pattern detection unit. Its major task is the generation of interrupt trigger events, alarms and other trigger signals based on selectable trigger events (for example to generate external interrupt requests if an edge occurs at an input pin). The detected events can also be used by other target modules to trigger or to gate module-specific actions.

The available trigger sources and defined target modules are defined in sections “ERU Input Pin Connections” and “ERU Output Pin Connections” respectively.

35.3.2.1 Feature list

- Supports generation of interrupt trigger events based on selectable trigger events at different inputs
- 8 independent input channels for input selection and conditioning of trigger or gating functions
- Event distribution with a Connecting Matrix which defines the events of the Input Channel x that lead to a reaction of an Output Channel y
- 8 independent Output Channels for combination of events, definition of their effects and distribution to the system (for example interrupt generation, timer triggering ...)

35.3.2.2 Introduction

The ERU can be split in three main functional parts:

- Input Channels x
- Connecting Matrix
- Output Channels y

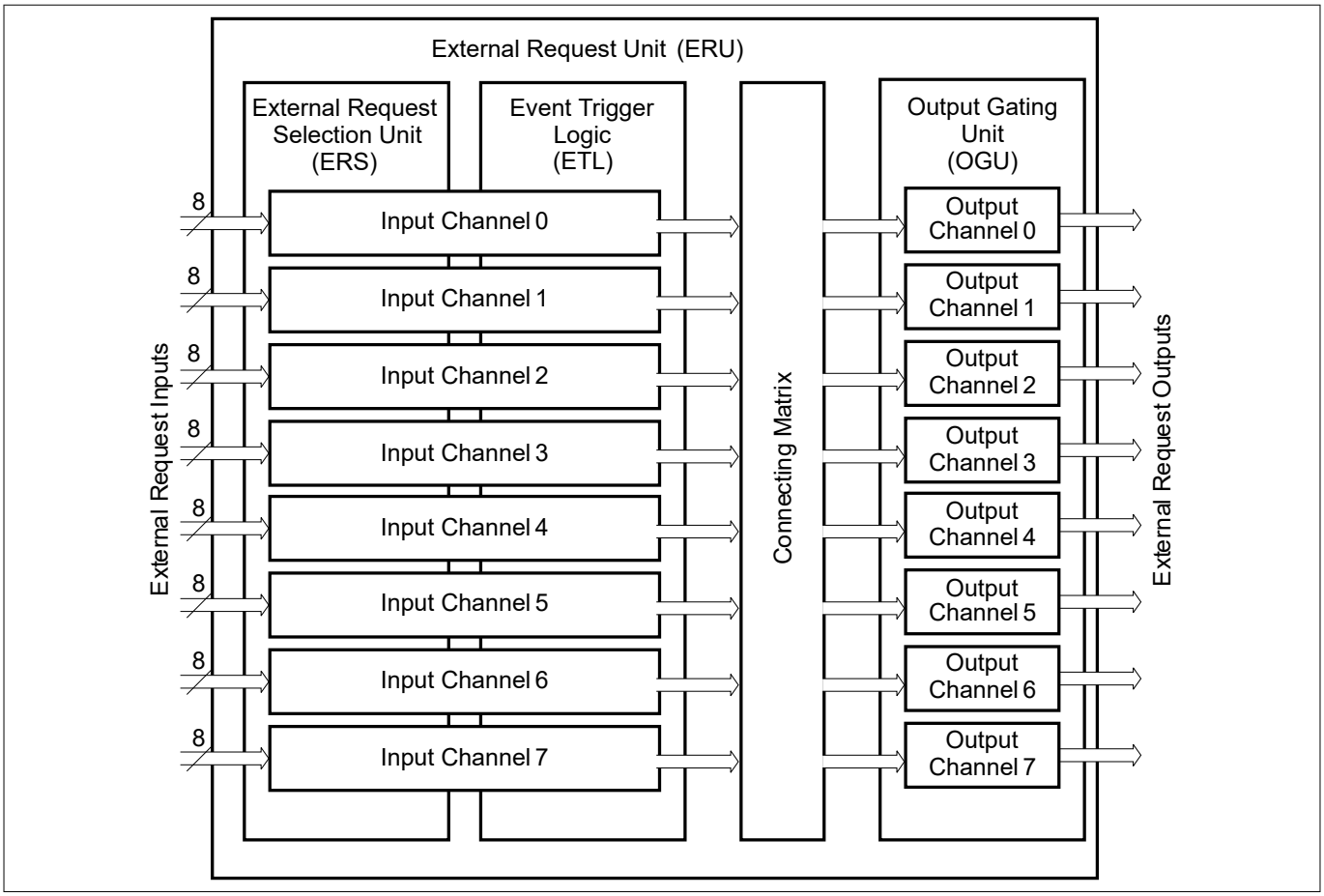


Figure 768 External request unit overview

These tasks are handled by the following building blocks:

- An External Request Select unit (ERSx) per input channel allows the selection of one input out of the 8 possible available inputs
- An Event Trigger Logic (ETLx) per input channel allows the definition of the transition (edge selection or by software) that leads to a trigger event and can also store this status. Here, the input levels of the selected signals are translated into events (event detected = event flag becomes set, independent of the polarity of the original input signals)
- The Connecting Matrix distributes the events and status flags generated by the input channels to the output channels
- An Output Gating Unit (OGUy) per output channel that combines the available trigger events and status information from the input channels. An event at one input channel can be forwarded to several output channels, or also events of several input channels can be combined to a reaction of one output channel (pattern detection). Different types of reactions are possible, for example interrupt generation (based on signals ERU_IOUTy)

35.3.2.3 REQx digital port input glitch filter (FILT)

Signal noise can lead to glitches on input pins from PORTS. These glitches due to signal noise may be suppressed by digital glitch filters similar to Filter and Prescaler Cells (FPC) in Delayed Debounce Filter Mode with up and down (no reset).

The filter calculates the integral of the signal. If the integral reaches a programmable saturation point, the input change is passed on to the ERS.

The REQx glitch filters are only available on the REQx inputs coming directly from PORTS. They are enabled and configured using register ERU_EIFILT. The input filter sample frequency is f_{FILT},where $f_{FILT} = f_{SPB}$/ ERU_EIFILT.FILTDIV.

On each new input filter sample, if the state of the input sample differs from the current state of the filter output, then an internal counter is incremented by one. When the state of the input sample matches the current state of the filter output and the counter is not zero, the counter is decremented by one. When the counter matches the compare threshold value stored in ERU_EIFILT.DEPTH, the state of the filter output is toggled and the counter is reset to zero, as shown in the next figure.

The filter clock divider and threshold can both be independently programmed to a value between 1 and 15, giving a range of possible glitch characteristics from 10 ns to more than 2 µs, assuming f_{FILT} of 100 MHz. Typically a threshold (depth) of 3 to 5 T_{filt} is sufficient. If DEPTH or FILTDIV is 0 (default value) all filters are inactive.

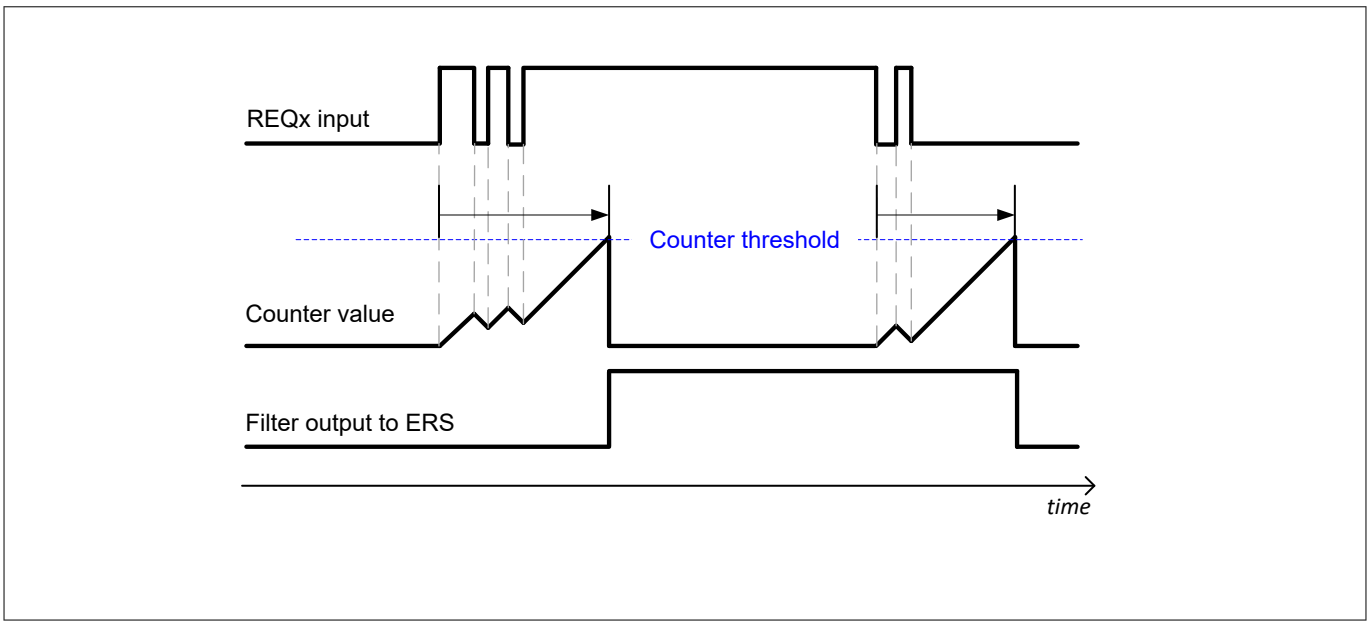


Figure 769 REQx digital filter

35.3.2.4 External request selector unit (ERS)

Each ERS selects one of eight inputs as the one input signal of the respective input channel. Figure below shows the structure of this block.

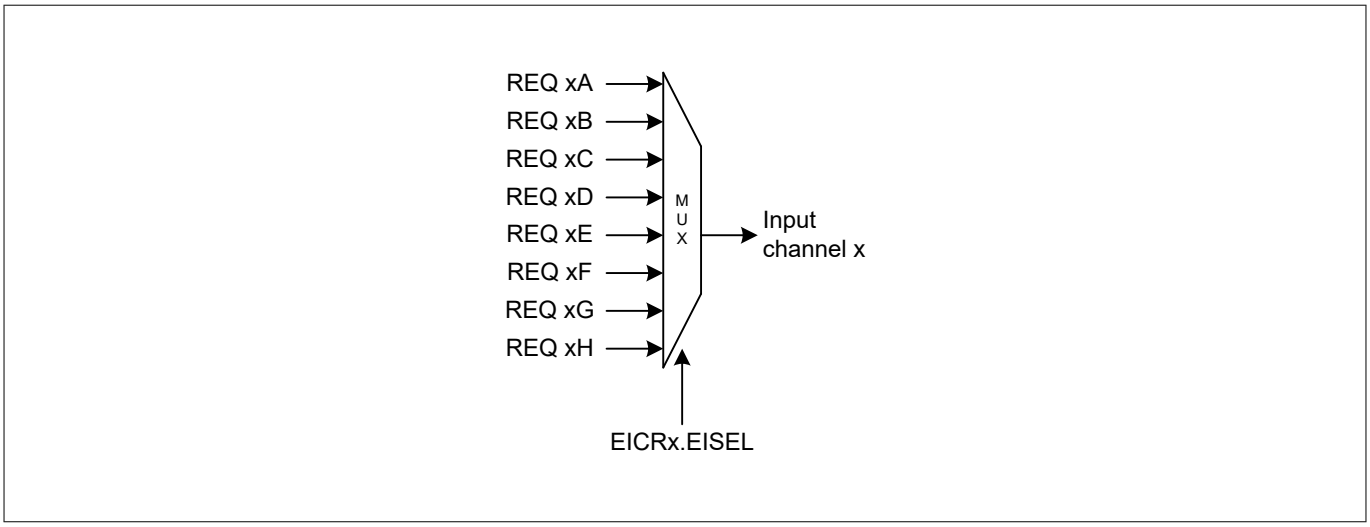


Figure 770 External request select unit overview

The ERS unit for channel x is controlled through bit-field ERU_EICRx.EISEL. Since ERU inputs can in general come from signals in different clock domains it is necessary to synchronize the signals to the local clock. In order to save power additional control is available through ERU_EICRx.EIEN to explicitly enable or disable clock of input synchronization logic.

***Note**: The External Request Unit (ERU) may react on changes of control registers even if there is no level change or edge at its inputs. For example in case the level at the input of the input synchronization logic is different from the registered value, switching on the synchronization function with the ERU_EICRx.EIEN bits will then be perceived by the ERU as an edge and will therefore result in the configured ERU reaction.*

35.3.2.5 Event trigger logic (ETL)

For each Input Channel x, an Event trigger logic ETLx derives a trigger event and a status from the input channel x delivered by the associated ERSx unit. Each ETLx is based on an edge detection block, where the detection of a rising or a falling edge can be individually enabled. Both edges lead to a trigger event if both enable bits are set (for example to handle a toggling input). Each ETLx unit has an associated ERU_EICRx register, that controls it (the register also holds control bits for the associated ERSx unit pair).

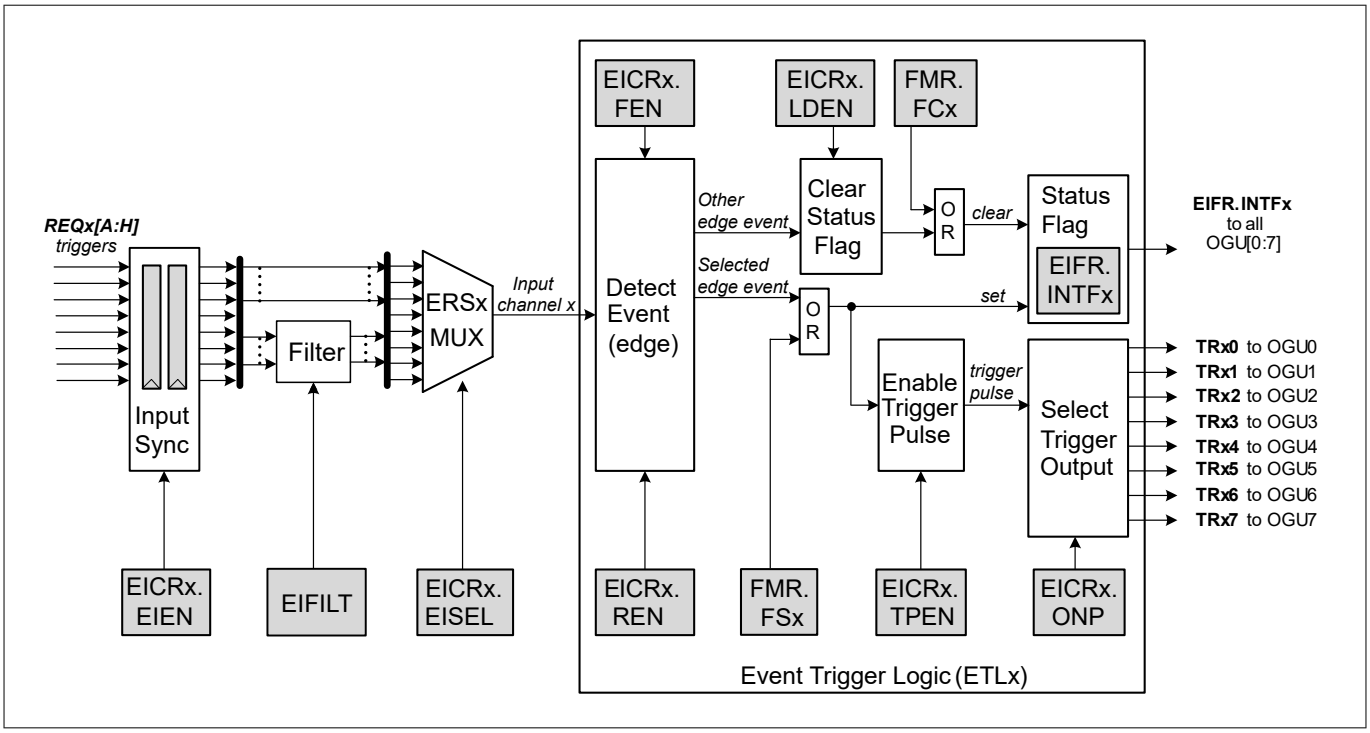


Figure 771 Event trigger logic overview

When the selected edge event is detected, the status flag ERU_EIFR.INTFx becomes set. The status flag is cleared automatically if the “opposite” event is detected, if so is enabled through bit ERU_EICRx.LDEN=1. For example, if only the falling edge detection is enabled to set the status flag, it is cleared when the rising edge is detected. In this mode, it can be used for pattern detection where the actual status of the input is important (enabling both edge detections is not useful in this mode). The output of the status flag is connected to all following Output Gating Units (OGUy) in parallel (see figure in related information) to provide pattern detection capability of all OGUy units based on different or the same status flags.

In addition to the modification of the status flag, a trigger pulse output TRxy of ETLx can be enabled (by bit ERU_EICRx.TPEN) and selected to trigger actions in one of the OGUy units. The target OGUy for the trigger is selected by bit-field ERU_EICRx.ONP. The trigger becomes active when the selected edge event is detected, independently from the status flag ERU_EIFR.INTFx.

Related information

Connecting matrix on page 5883

35.3.2.6 Connecting matrix

The Connecting matrix distributes the trigger signals (TRxy) and status signals (ERU_EIFR.INTFx) from the different ETLx units between the OGUy units. Figure below provides a complete overview of the connections between the ETLx and the OGUy units.

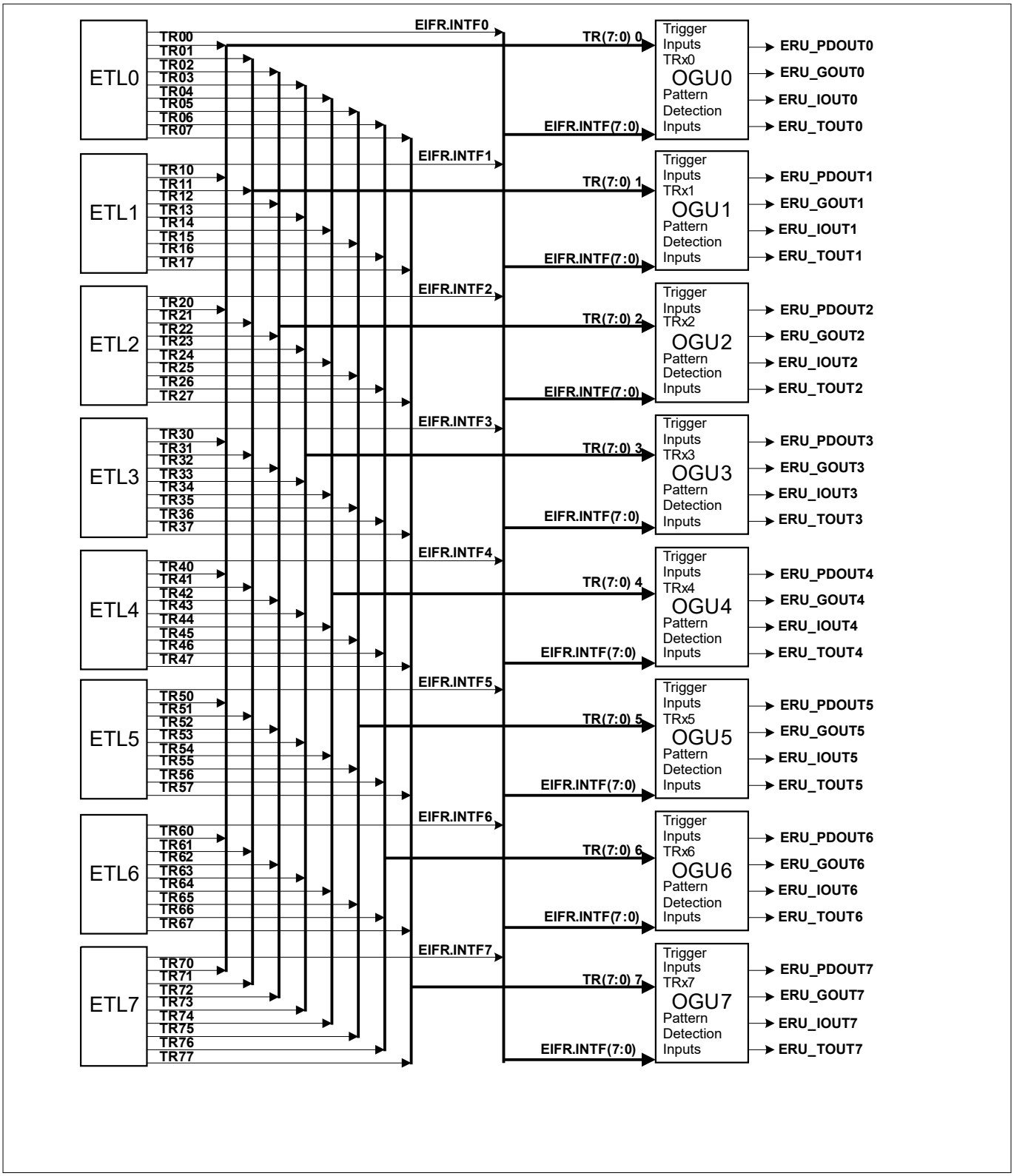


Figure 772 Connecting matrix between ETLx and OGUy

35.3.2.7 Output gating unit (OGU)

Each OGUy unit combines the available trigger events and status flags from the Input Channels and distributes the results to the system. Next figure illustrates the logic blocks within an OGUy unit. All functions of an OGUy unit are controlled by the associated ERU_IGCRy registers, one for each output channel. The function of an OGUy unit can be split into two parts:

- Trigger combination: All trigger signals TRxy from the Input Channels that are enabled and directed to OGUy and a pattern change event (if enabled) are logically OR-combined
- Pattern detection: The status flags ERU_EIFR.INTFx of the Input Channels can be enabled to take part in the pattern detection. A pattern match is detected while all enabled status flags are set, and stored in ERU_PDRR register

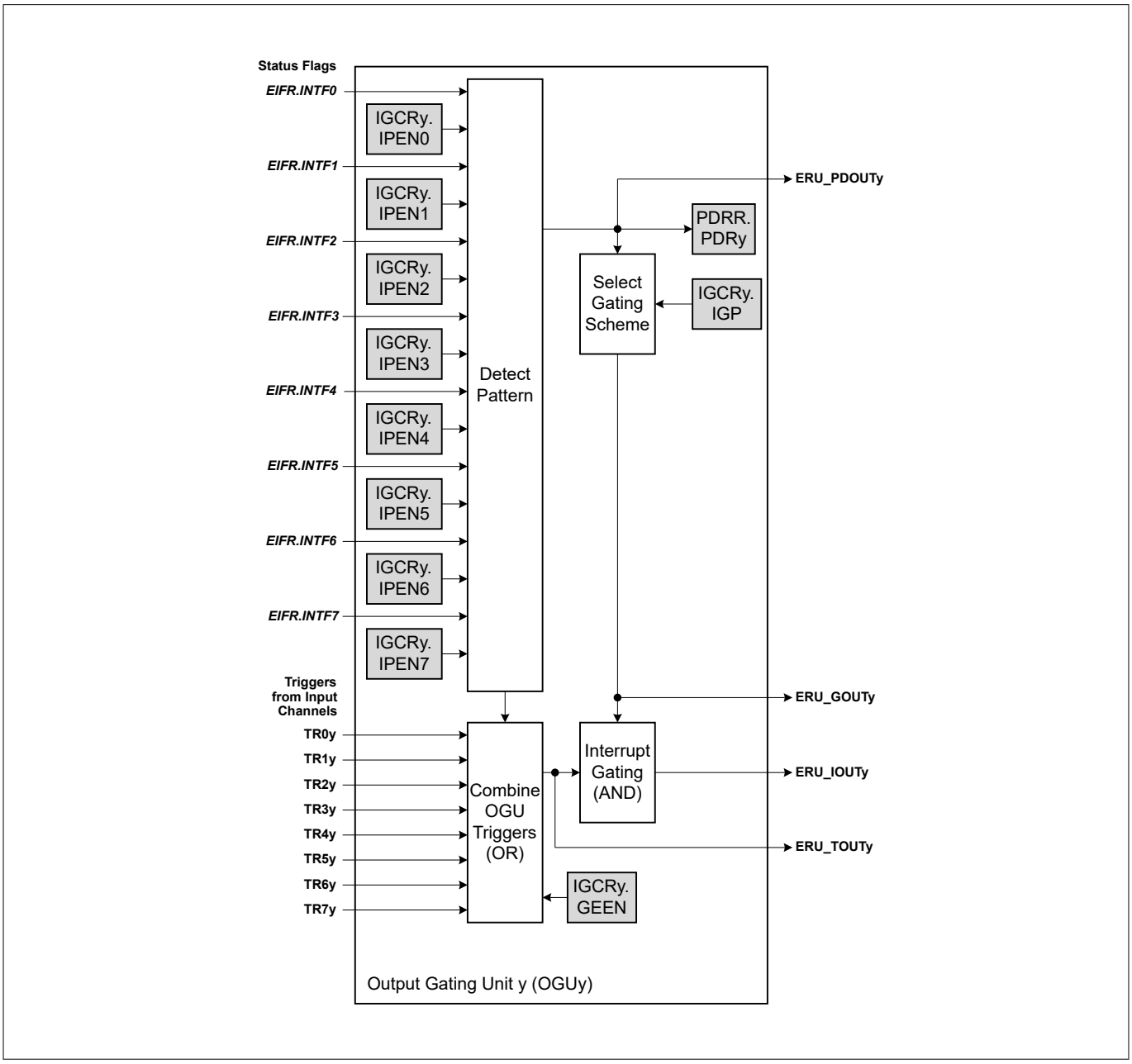


Figure 773 Output gating unit for output channel y

Each OGUy units generates 4 output signals that are distributed to the system.

- ERU_PDOUTy to directly output the pattern match information for gating purposes in defined target modules (pattern match ERU_PDRR.PDRy = '1')
- ERU_GOUTy to output the pattern match or pattern miss information (inverted pattern match), or a permanent 0 or 1 under software control for gating purposes in other modules
- ERU_TOUTy as combination of a peripheral trigger, a pattern detection result change event, or the ETLx trigger outputs TRxy to trigger actions in other modules
- ERU_IOUTy as gated trigger output (ERU_GOUTy logical AND-combined with ERU_TOUTy) to trigger interrupts (for example the interrupt generation can be gated to allow interrupt activation only during a certain time window)

35.3.2.7.1 Trigger combination

The “Combine OGU Triggers” block logically OR-combines different trigger inputs to form a common trigger ERU_TOUTy. Possible trigger inputs are:

- In each ETLx unit of the Input Channels, the trigger output TRxy can be enabled and the trigger event can be directed to one of the OGUy units
- In case that at least one pattern detection input is enabled (ERU_IGCRy.IPENx) and a change of the pattern detection result from pattern match to pattern miss (or vice-versa) is detected, a trigger event is generated to indicate a pattern detection result event (if enabled by ERU_IGCRy.GEEN)

The Trigger combination offers the possibility to program different trigger criteria for several input signals (independently for each Input Channel) or peripheral signals, and to combine their effects to a single output, for example to generate an interrupt or to start an ADC conversion. This combination capability allows the generation of an interrupt per OGUy that can be triggered by several inputs (multitude of request sources leads to one reaction).

35.3.2.7.2 Pattern detection

The “Detect Pattern” block allows the combination of the status flags of all ETLx units. Each status flag can be individually included or excluded from the pattern detection for each OGUy, through control bits ERU_IGCRy.IPENx. The pattern detection block outputs the following pattern detection results:

- Pattern match (ERU_PDRR.PDRy = '1' and ERU_PDOUTy = '1'): A pattern match is indicated while all status flags that are included in the pattern detection are '1'
- Pattern miss (ERU_PDRR.PDRy = '0' and ERU_PDOUTy = '0'): A pattern miss is indicated while at least one of the status flags that are included in the pattern detection is '0'

In addition, the pattern detection can deliver a trigger event if the pattern detection result changes from match to miss or vice-versa (if enabled by ERU_IGCRy.GEEN='1'). The pattern result change event is logically OR-combined with the other enabled trigger events to support interrupt generation or to trigger other module functions (for example in an ADC). The event is indicated when the pattern detection result changes and ERU_PDRR.PDRy becomes updated.

The interrupt generation in the OGUy is based on the trigger ERU_TOUTy that can be gated (masked) with the pattern detection result ERU_PDOUTy. This allows an automatic and reproducible generation of interrupts during a certain time window, where the request event is elaborated by the trigger combination block and the time window information (gating) is given by the pattern detection. For example, interrupts can be issued on a regular time base while a combination of input signals occurs (pattern detection based on ETLx status bits).

In total eight interrupts are generated by the ERU. The mapping of interrupt requests to OGUy blocks is shown in the ERU connection diagram.

A programmable gating scheme introduces flexibility to adapt to application requirements and allows the generation of interrupt requests ERU_IOUTy under different conditions:

- Pattern match (ERU_IGCRy.IGP = 10_B): An interrupt request is issued when a trigger event occurs while the pattern detection shows a pattern match

- Pattern miss (ERU_IGCRy.IGP = 11_B): An interrupt request is issued when the trigger event occurs while the pattern detection shows a pattern miss
- Independent of pattern detection (ERU_IGCRy.IGP = 01_B): In this mode, each occurring trigger event leads to an interrupt request. The pattern detection output can be used independently from the trigger combination for gating purposes of other peripherals (independent use of ERU_TOUTy and ERU_PDOUTy with interrupt requests on trigger events)
- No interrupts (ERU_IGCRy.IGP = 00_B, default setting). In this mode, an occurring trigger event does not lead to an interrupt request. The pattern detection output can be used independently from the trigger combination for gating purposes of other peripherals (independent use of ERU_TOUTy and ERU_PDOUTy without interrupt requests on trigger events)

35.3.2.7.3 Triggering SMU alarms

It is possible to trigger alarms to the SMU through the functional outputs of the ERU.

This is normally used to trigger an alarm from an external component, for example the component fail signal is connected to the ERU inputs and when it fails the ERU detects a transition on this input, it will trigger an alarm to the SMU.

The ERU outputs that are connected to the SMU are the ERU_IOUTy signals - ERU_IOUT0 throughout ERU_IOUT7.

Please address the SMU chapter to know which alarm lines are being used for each signal.

To enable the triggers on the ERU, one shall do the following steps:

1. Identify which ERU input signal shall be used as an alarm signal

2. Configure the specific ETLx unit by:
- Configure the ERU_EICRx.EISEL bit-field to select the input
- Configure the ERU_EICRx.REN and ERU_EICRx.FEN bit-fields to select if the input signal is active on a rising edge or falling edge
- If the trigger signal from the ETLx shall be used instead of the status flag, as source for the alarm, then configure the ERU_EICRx.ONP bit-field

3. Configure the specific OGUy that shall be used to trigger the SMU alarm - this is linked to which output shall be used, for example. ERU_IOUT4, then the OGU4 shall be configured:
- Configure the ERU_IGCRy.IPEN0 to IPEN7 if one or more status flags are used to generate or gate the trigger signal
- Configure ERU_IGCRy.GEEN bit-field to use the pattern detection path to trigger/generate the alarm/IOUT
- Configure the ERU_IGCRy.IGP bit-field to select the gating scheme for the output alarm/IOUT

One shall address the SMU chapter, to understand if additional configuration is needed to enable a specific alarm line.

35.3.2.8 ERU input pin connections

***Note**: All signal connections listed in the table below are subject to availability: if respective Module or Port Pin is not available in a given product and package then the connection is not available either.*

Table 1401 External request unit - input connections

ERS mux	Input channel	Module	Module signal/pin
ERS0	Input A	PORTS	P15.4

(table continues...)

Table 1401 (continued) External request unit - input connections

ERS mux	Input channel	Module	Module signal/pin
	Input B	PORTS	P10.7
	Input C	PMS (RTC)	PMS_RTC_MUXOUT
	Input D	MSC	FCLP
	Input E	CPU5 (STM)	CPU5_SCU_STM_RST_REQ
	Input F	PORTS	P02.7
	Input G	eGTM	eGTM_SCU_TRIG(0)
	Input H	-	-
ERS1	Input A	PORTS	P14.3
	Input B	PORTS	P10.8
	Input C	eGTM	eGTM_SCU_TRIG(1)
	Input D	CPU0 (STM)	CPU0_SCU_STM_RST_REQ
	Input E	-	-
	Input F	CLOCK	EXTCLK0
	Input G	PORTS	P16.1
	Input H	-	-
ERS2	Input A	PORTS	P10.2
	Input B	PORTS	P02.1
	Input C	PORTS	P00.4
	Input D	ERAY0	MACROTICK
	Input E	PMS (WUT)	WUT-Underflow
	Input F	eGTM	eGTM_SCU_TRIG(2)
	Input G	-	-
	Input H	-	-
ERS3	Input A	PORTS	P10.3
	Input B	PORTS	P14.1
	Input C	PORTS	P02.0
	Input D	CPU1 (STM)	CPU1_SCU_STM_RST_REQ
	Input E	eGTM	eGTM_SCU_TRIG(3)
	Input F	-	-
	Input G	-	-
	Input H	-	-

(table continues...)

Table 1401 (continued) External request unit - input connections

ERS mux	Input channel	Module	Module signal/pin
ERS4	Input A	PORTS	P33.7
	Input B	GTM	GTM_SCU_TRIG(0)
	Input C	GPT120	T3OUT
	Input D	PORTS	P15.5
	Input E	SMM	ESR2
	Input F	-	
	Input G	PORTS	P25.1
	Input H	-	-
ERS5	Input A	PORTS	P15.8
	Input B	GTM	GTM_SCU_TRIG(1)
	Input C	GPT120	T6OUT
	Input D	CPU2 (STM)	CPU2_SCU_STM_RST_REQ
	Input E	PORTS	P22.4
	Input F	CLOCK	EXTCLK1
	Input G	PORTS	P02.8
	Input H	-	-
ERS6	Input A	PORTS	P20.0
	Input B	PORTS	P11.10
	Input C	SMM	ESR0
	Input D	CPU3 (STM)	CPU3_SCU_STM_RST_REQ
	Input E	GTM	GTM_SCU_TRIG(2)
	Input F	PORTS	P23.1
	Input G	GPT120	T6OFL
	Input H	-	-
ERS7	Input A	PORTS	P20.9
	Input B	PORTS	P15.1
	Input C	SMM	ESR1
	Input D	ERAY1	MACROTICK
	Input E	CPU4 (STM)	CPU4_SCU_STM_RST_REQ
	Input F	GTM	GTM_SCU_TRIG(3)
	Input G	PORTS	P02.11
	Input H	-	-

35.3.2.9 ERU output pin connections

***Note**: All signal connections listed in the table below are subject to availability: if respective module, module function or Port pin is not available in a given product and package then the connection is not available either.*

Table 1402 External request unit - output connections

Output channel	Output	Module	Module signal/pin
OGU0	IOUT0	SMU	ALM_SCUERU0
		ADC	ADC_SCUERU0
		IR	SRC_SCUERU0
		GST	GST_SCUERU0
		LETH0	P0_SCU_AUX1
		GETH0	P0_SCU_AUX1
	PDOUT0	GTM	TIMnINSEL
		eGTM	TIMnINSEL
		ERAY0	STPWT(0)
		ERAY1	STPWT(0)
OGU1	IOUT1	SMU	ALM_SCUERU1
		ADC	ADC_SCUERU1
		IR	SRC_SCUERU1
		LETH0	P0_SCU_AUX2
	PDOUT1	GTM	TIMnINSEL
		eGTM	TIMnINSEL
		ERAY0	STPWT(1)
		ERAY1	STPWT(1)
OGU2	IOUT2	SMU	ALM_SCUERU2
		ADC	ADC_SCUERU2
		IR	SRC_SCUERU2
		LETH0	P1_SCU_AUX1
		GETH0	P1_SCU_AUX1
	PDOUT2	GTM	TIMnINSEL
		eGTM	TIMnINSEL
		ERAY0	STPWT(2)
		ERAY1	STPWT(2)
OGU3	IOUT3	SMU	ALM_SCUERU3
		ADC	ADC_SCUERU3

(table continues...)

Table 1402 (continued) External request unit - output connections

Output channel	Output	Module	Module signal/pin
		IR	SRC_SCUERU3
		LETH0	P1_SCU_AUX2
	PDOUT3	GTM	TIMnINSEL
		eGTM	TIMnINSEL
		ERAY0	STPWT(3)
		ERAY1	STPWT(3)
OGU4	IOUT4	SMU	ALM_SCUERU4
		ADC	ADC_SCUERU4
		IR	SRC_SCUERU4
		GST	GST_SCUERU4
		GPT12	GPT12_SCUERU4
		LETH0	P2_SCU_AUX1
	PDOUT4	GPT120	T3INC
		GTM	TIMnINSEL
		eGTM	TIMnINSEL
OGU5	IOUT5	SMU	ALM_SCUERU5
		ADC	ADC_SCUERU5
		IR	SRC_SCUERU5
		GPT12	GPT12_SCUERU5
		LETH0	P2_SCU_AUX2
	PDOUT5	GTM	TIMnINSEL
		eGTM	TIMnINSEL
OGU6	IOUT6	SMU	ALM_SCUERU6
		ADC	ADC_SCUERU6
		IR	SRC_SCUERU6
		GST	GST_SCUERU6
		GPT12	GPT12_SCUERU6
		LETH0	P3_SCU_AUX1
		LETH1	P0_SCU_AUX1
	PDOUT6	GPT120	CAPINB0
		GTM	TIMnINSEL
		eGTM	TIMnINSEL
OGU7	IOUT7	SMU	ALM_SCUERU7

(table continues...)

Table 1402 (continued) External request unit - output connections

Output channel	Output	Module	Module signal/pin
		ADC	ADC_SCUERU7
		IR	SRC_SCUERU7
		LETH1	P0_SCU_AUX2
	PDOUT7	GTM	TIMnINSEL
		eGTM	TIMnINSEL

35.3.3 Global overlay controls

Overlay functionality is described in detail in the CPU Overlay sub-chapter named "Data Access Overlay (OVC)".

Two registers globally control the overlay operation for all CPUs:

- Overlay Enable Register OVCENABLE disables or enables data access overlay individually for each CPU
  - Global Overlay Disable (GOD) signal is one-time programmable bit used to globally disable overlay function until the next application reset
- Overlay Control Register OVCCON can be used to perform the following actions on a selected set of CPUs:
  - Concurrently enable/disable selected overlay blocks
  - Invalidate data cache

All overlay control registers are reset to their default values with the Application Reset.

***Note**: In case a certain CPU is not available in the device (smaller devices can have less than 6 CPU cores), its respective overlay control bits will still be there, but their function will be ignored.*

Related information

Data access overlay (OVC) on page 159

35.3.4 Miscellaneous system control

This section collects functions that serve various system control purposes.

35.3.4.1 Chip information register (CHIPINFO)

The CHIPINFO register can be used to determine basic information about the device, such as:

- Department within Infineon Technologies AG (for example Automotive), through the DEPT bit-field
- The JEDEC normalized manufacturer code, through the MANUF bit-field
- Revision number of the TC4x device, through the CHREV bit-field
- Product family, through CHTEC bit-field

35.3.4.2 SOTA address region control

The software-over-the-air (SOTA) control register SOTACTRL provides the firmware with the capability to install the currently used address region for supporting the SOTA feature.

The currently used address region is selected through the SOTACTRL.SOTASEL register bit-field. More details on the SOTACTRL register are provided in the SCU registers and CPU sections.

35.4 Registers

35.4.1 Register overview - access mode glossary

Table 1403 Register overview - access mode glossary

Keyword	Description
CSAE	Access protection using PROT register PROTCSAE.
CSE	Access protection using PROT register PROTCSE.
E	Access protection using PROT register PROTE.
SE	Access protection using PROT register PROTSE.
APU-PCS	Protection group consisting of registers ACCENCS_WRA, ACCENCS_WRB, ACCENCS_RDA, ACCENCS_RDB, ACCENCS_VM, ACCENCS_PRS.
PCS	Access protection using *APU-PCS* registers.
APU-PGg (g=0-3)	Protection group consisting of registers ACCENGRPg_WRA, ACCENGRPg_WRB, ACCENGRPg_RDA, ACCENGRPg_RDB, ACCENGRPg_VM, ACCENGRPg_PRS.
PGg	Access protection using *APU-PGg* registers.
APU-PNOM	Protection group consisting of registers ACCENNOM_WRA, ACCENNOM_WRB, ACCENNOM_RDA, ACCENNOM_RDB, ACCENNOM_VM, ACCENNOM_PRS.
PNOM	Access protection using *APU-PNOM* registers.
BE	Always returns a Bus Error.
U	No access restrictions.
PROT	Access restrictions as defined in the PROT register access rules.
STP	Access only during start-up, before start-up protection is set (SMM_STCON.ST_PROT=1).

35.4.2 Registers overview - SCU (ascending offset address)

Table 1404 Registers overview - SCU (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ID	Module identification register	008_H	PNOM	BE	PowerOn Reset	5896
PROTE	PROT register endinit	018_H	U	SV, PROT	Application Reset	5896
PROTSE	PROT register safe endinit	$01C_H$	U	SV, PROT	Application Reset	5898
PROTCSE	PROT register cyber-secure endinit	020_H	U	SV, PROT	Application Reset	5899
PROTCSAE	PROT register cyber-secure APU endinit	024_H	U	SV, PROT	Application Reset	5901
ACCENNOM_WRA	ACCENNOM write access enable register A	028_H	U	SE, SV	Application Reset	5903

(table continues...)

Table 1404 (continued) Registers overview - SCU (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ACCENNOM_WRB	ACCENNOM write access enable register B	$02C_H$	U	SE, SV	Application Reset	5903
ACCENNOM_RDA	ACCENNOM read access enable register A	030_H	U	SE, SV	Application Reset	5904
ACCENNOM_RDB	ACCENNOM read access enable register B	034_H	U	SE, SV	Application Reset	5904
ACCENNOM_VM	ACCENNOM VM access enable register	038_H	U	SE, SV	Application Reset	5905
ACCENNOM_PRS	ACCENNOM PRS access enable register	$03C_H$	U	SE, SV	Application Reset	5905
ACCENGRPg_WRA	ACCENGRPg write access enable register A	048_H+g* 20_H	U	SE, SV	Application Reset	5906
ACCENGRPg_WRB	ACCENGRPg write access enable register B	$04C_H$+g* 20_H	U	SE, SV	Application Reset	5907
ACCENGRPg_RDA	ACCENGRPg read access enable register A	050_H+g* 20_H	U	SE, SV	Application Reset	5907
ACCENGRPg_RDB	ACCENGRPg read access enable register B	054_H+g* 20_H	U	SE, SV	Application Reset	5908
ACCENGRPg_VM	ACCENGRPg VM access enable register	058_H+g* 20_H	U	SE, SV	Application Reset	5908
ACCENGRPg_PRS	ACCENGRPg PRS access enable register	$05C_H$+g* 20_H	U	SE, SV	Application Reset	5909
ACCENCS_WRA	ACCENCS write access enable register A	$0C8_H$	U	CSAE, SV	Application Reset	5909
ACCENCS_WRB	ACCENCS write access enable register B	$0CC_H$	U	CSAE, SV	Application Reset	5910
ACCENCS_RDA	ACCENCS read access enable register A	$0D0_H$	U	CSAE, SV	Application Reset	5910
ACCENCS_RDB	ACCENCS read access enable register B	$0D4_H$	U	CSAE, SV	Application Reset	5911
ACCENCS_VM	ACCENCS VM access enable register	$0D8_H$	U	CSAE, SV	Application Reset	5911
ACCENCS_PRS	ACCENCS PRS access enable register	$0DC_H$	U	CSAE, SV	Application Reset	5912
TRAPS_CPU_CTRL	Control register for traps related to CPU0-5	$0E8_H$	PNOM	SV, PNOM	Application Reset	5913
TRAPS_CPU_SETCLRGRPg_SETCLREN	SET/CLR enable register for group g for traps related to CPU0-5	$0EC_H$+g* 12	PNOM	E, SV, PNOM	Application Reset	5913
TRAPS_CPU_SETCLRGRPg_SET	Trap set register for group g for traps related to CPU0-5	$0F0_H$+g* 12	PGg	PGg, SV	Application Reset	5914

(table continues...)

Table 1404 (continued) Registers overview - SCU (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
TRAPS_CPU_SETCLRGRPg_CLR	Trap clear register for group g for traps related to CPU0-5	$0F4_H$+g*12	PGg	PGg, SV	Application Reset	5915
TRAPS_CPU_STAT	Trap status register for traps related to CPU0-5	$11C_H$	PNOM	BE	See 5916	5916
TRAPS_CPU_DISy	Trap disable register for single CPUy	120_H+y*4	PNOM	E, SV, PNOM	Application Reset	5917
TRAPS_CS_CTRL	Control register for traps related to CPUcs	138_H	PCS	PCS, SV	Application Reset	5918
TRAPS_CS_SETCLR_SETCLREN	SET/CLR enable register for traps related to CPUcs	$13C_H$	PCS	CSE, SV, PCS	Application Reset	5918
TRAPS_CS_SETCLR_SET	Trap set register for traps related to CPUcs	140_H	PCS	PCS, SV	Application Reset	5919
TRAPS_CS_SETCLR_CLR	Trap clear register for traps related to CPUcs	144_H	PCS	PCS, SV	Application Reset	5920
TRAPS_CS_STAT	Trap status register for traps related to CPUcs	148_H	PCS	BE	Application Reset	5921
TRAPS_CS_DIS	Trap disable register for CPUcs	$14C_H$	PCS	CSE, SV, PCS	Application Reset	5921
ERU_EIFILT	External input filter register	150_H	PNOM	E, SV, PNOM	Application Reset	5922
ERU_EICRx	External input channel register x	154_H+x*4	PNOM	E, SV, PNOM	Application Reset	5925
ERU_EIFR	External input flag register	174_H	PNOM	BE	Application Reset	5927
ERU_FMR	Flag modification register	178_H	PNOM	PNOM, SV	Application Reset	5927
ERU_IGCRy	Interrupt gating control register y	$17C_H$+y*4	PNOM	E, SV, PNOM	Application Reset	5928
ERU_PDRR	Pattern detection result register	$19C_H$	PNOM	BE	Application Reset	5929
OVCENABLE	Overlay enable register	$1A0_H$	PNOM	E, SV, PNOM	Application Reset	5930
OVCCON	Overlay control register	$1A4_H$	PNOM	PNOM, SV	Application Reset	5931
CHIPINFO	Chip information register	$1A8_H$	PNOM	BE	See 5933	5933
SOTACTRL	Software-over-the-air control register	$1AC_H$	PNOM	PNOM, STP	See 5934	5934

35.4.3 Module identification register

ID Offset address: 008_H

Module identification register PowerOn Reset value: $00C4\ C001_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MOD_NUM															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MOD_TYPE								MOD_REV							
r								r							

Field	Bits	Type	Description
MOD_REV	7:0	r	**Module revision number** This bit field indicates the revision number of the SCU module (01_H). Bits [5:0] define the SCU module IP revision number. Bits [7:6] define the family. This allows a continuous numbering of the revision AND a distinction of the different SCU generations. The revision number starts with 000001 and is incremented when a SCU change is detectable by customer software. MOD_REV for other variants will be defined by Design Team.
MOD_TYPE	15:8	r	**Module type** The bit field is set to $C0_H$, which defines the module as a 32-bit module.
MOD_NUM	31:16	r	**Module number value** This bit field defines the module identification number.The identification number for the SCU is $00C4_H$.

35.4.4 PROT register endinit

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner and in some PROT states, the secure master to update the PROT state as described in the PROT mechanism.

The PROTE register controls lock / unlock of the local Endinit (E) protected control registers.

PROTE Offset address: 018_H

PROT register endinit Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OWEN	ODEF	TAGID						PRSEN	PRS			VMEN	VM		
w	rw	rw						rw	rw			rw	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												SWEN	STATE		
r												w	rwh		

Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
VM	18:16	rw	**Virtual Machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual Machine definition Enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection Set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.
PRSEN	23	rw	**Protection Set definition Enable for PROT owner** 0_B **Disable**: PRS is not part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; should be written with 0.

Table 1405 Access mode restrictions of PROTE sorted by descending priority

Mode name	Access mode		Description
write 1 to .**SWEN**	rwh	STATE	
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

35.4.5 PROT register safe endinit

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner and in some PROT states, the secure master to update the PROT state as described in the PROT mechanism.

The PROTSE register controls lock / unlock of the local Safe Endinit (SE) protected control registers.

PROTSE Offset address: $01C_H$

PROT register safe endinit Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OWEN	**ODEF**	**TAGID**						**PRSEN**	**PRS**			**VMEN**	**VM**		
w	rw	rw						rw	rw			rw	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												**SWEN**	**STATE**		
r												w	rwh		

Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write

(table continues...)

(continued)

Field	Bits	Type	Description
VM	18:16	rw	**Virtual Machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual Machine definition Enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection Set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.
PRSEN	23	rw	**Protection Set definition Enable for PROT owner** 0_B **Disable**: PRS is not part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; should be written with 0.

Table 1406 Access mode restrictions of PROTSE sorted by descending priority

Mode name	Access mode		Description
write 1 to .**SWEN**	rwh	STATE	
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

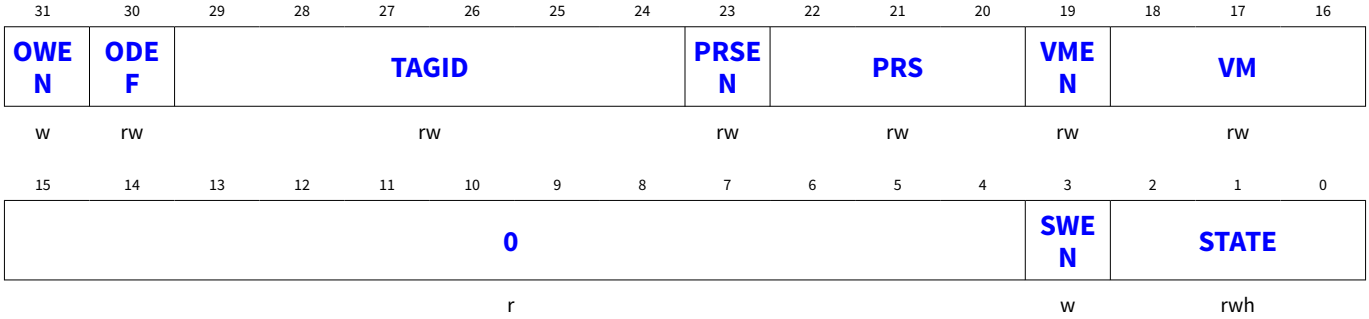
35.4.6 PROT register cyber-secure endinit

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner and in some PROT states, the secure master to update the PROT state as described in the PROT mechanism.

The PROTCSE register controls lock / unlock of the local Cyber Secure Endinit (CSE) protected control registers.

PROTCSE Offset address: 020_H

PROT register cyber-secure endinit Application Reset value: $4C00\ 0000_H$



Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
VM	18:16	rw	**Virtual Machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual Machine definition Enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection Set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.
PRSEN	23	rw	**Protection Set definition Enable for PROT owner** 0_B **Disable**: PRS is not part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner

(table continues...)

(continued)

Field	Bits	Type	Description
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; should be written with 0.

Table 1407 Access mode restrictions of PROTCSE sorted by descending priority

Mode name	Access mode		Description
write 1 to .**SWEN**	rwh	STATE	
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

35.4.7 PROT register cyber-secure APU endinit

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner and in some PROT states, the secure master to update the PROT state as described in the PROT mechanism.

The PROTCSAE register controls lock / unlock of the local Cyber Secure APU Endinit (CSAE) protected control registers.

PROTCSAE Offset address: 024_H

PROT register cyber-secure APU endinit Application Reset value: $4C00\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OWEN	**ODEF**	**TAGID**						**PRSEN**	**PRS**			**VMEN**	**VM**		
w	rw	rw						rw	rw			rw	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												**SWEN**	**STATE**		
r												w	rwh		

Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
VM	18:16	rw	**Virtual Machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual Machine definition Enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection Set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.
PRSEN	23	rw	**Protection Set definition Enable for PROT owner** 0_B **Disable**: PRS is not part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; should be written with 0.

Table 1408 Access mode restrictions of PROTCSAE sorted by descending priority

Mode name	Access mode		Description
write 1 to .**SWEN**	rwh	STATE	
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

35.4.8 ACCENNOM write access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

ACCENNOM_WRA Offset address: 028_H

ACCENNOM write access enable register A Application Reset value: $1000\ 0003_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	**EN30**	**EN29**	**EN28**	**EN27**	**EN26**	**EN25**	**EN24**	**EN23**	**EN22**	**EN21**	**EN20**	**EN19**	**EN18**	**EN17**	**EN16**
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	**EN14**	**EN13**	**EN12**	**EN11**	**EN10**	**EN09**	**EN08**	**EN07**	**EN06**	**EN05**	**EN04**	**EN03**	**EN02**	**EN01**	**EN00**
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access

35.4.9 ACCENNOM write access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

ACCENNOM_WRB Offset address: $02C_H$

ACCENNOM write access enable register B Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								**EN39**	**EN38**	**EN37**	**EN36**	**EN35**	**EN34**	**EN33**	**EN32**
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access
0	31:8	r	**Reserved** Read as 0; should be written with 0.

35.4.10 ACCENNOM read access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

ACCENNOM_RDA Offset address: 030_H

ACCENNOM read access enable register A Application Reset value: FFFF $FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access

35.4.11 ACCENNOM read access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

ACCENNOM_RDB Offset address: 034_H

ACCENNOM read access enable register B Application Reset value: 0000 $00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access
0	31:8	r	**Reserved** Read as 0; should be written with 0.

35.4.12 ACCENNOM VM access enable register

This register defines which virtual machine encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each VMx.

ACCENNOM_VM Offset address: 038_H

ACCENNOM VM access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for VM ID q** This bit enables read access to the access protected region for transactions with the VM ID q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for VM ID q** This bit enables write access to the access protected region for transactions with the VM ID q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

35.4.13 ACCENNOM PRS access enable register

This register defines which protection register set encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each PRSx.

ACCENNOM_PRS Offset address: $03C_H$

ACCENNOM PRS access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for PRS q** This bit enables read access to the access protected region for transactions with the PRS q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for PRS q** This bit enables write access to the access protected region for transactions with the PRS q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

35.4.14 ACCENGRPg write access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

ACCENGRPg_WRA (g=0-3) Offset address: 048_H+g*20_H

ACCENGRPg write access enable register A Application Reset value: $1000\ 0003_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access

35.4.15 ACCENGRPg write access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

ACCENGRPg_WRB (g=0-3) Offset address: $04C_H+g*20_H$

ACCENGRPg write access enable register B Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access
0	31:8	r	**Reserved** Read as 0; should be written with 0.

35.4.16 ACCENGRPg read access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

ACCENGRPg_RDA (g=0-3) Offset address: 050_H+g*20_H

ACCENGRPg read access enable register A Application Reset value: $FFFF\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access

35.4.17 ACCENGRPg read access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

ACCENGRPg_RDB (g=0-3) Offset address: 054_H+g*20_H

ACCENGRPg read access enable register B Application Reset value: $0000\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access
0	31:8	r	**Reserved** Read as 0; should be written with 0.

35.4.18 ACCENGRPg VM access enable register

This register defines which virtual machine encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each VMx.

ACCENGRPg_VM (g=0-3) Offset address: 058_H+g*20_H

ACCENGRPg VM access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for VM ID q** This bit enables read access to the access protected region for transactions with the VM ID q. 0_B Disabled for read access 1_B Enabled for read access

(table continues...)

(continued)

Field	Bits	Type	Description
WRq (q=00-07)	q+16	rw	**Write access enable for VM ID q** This bit enables write access to the access protected region for transactions with the VM ID q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

35.4.19 ACCENGRPg PRS access enable register

This register defines which protection register set encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each PRSx.

ACCENGRPg_PRS (g=0-3) Offset address: $05C_H+g*20_H$

ACCENGRPg PRS access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for PRS q** This bit enables read access to the access protected region for transactions with the PRS q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for PRS q** This bit enables write access to the access protected region for transactions with the PRS q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

35.4.20 ACCENCS write access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

ACCENCS_WRA Offset address: $0C8_H$

ACCENCS write access enable register A Application Reset value: $0000\ 1000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access

35.4.21 ACCENCS write access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

ACCENCS_WRB Offset address: $0CC_H$

ACCENCS write access enable register B Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access
0	31:8	r	**Reserved** Read as 0; should be written with 0.

35.4.22 ACCENCS read access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

ACCENCS_RDA Offset address: $0D0_H$

ACCENCS read access enable register A Application Reset value: $0000\ 1000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access

35.4.23 ACCENCS read access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

ACCENCS_RDB Offset address: $0D4_H$

ACCENCS read access enable register B Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access
0	31:8	r	**Reserved** Read as 0; should be written with 0.

35.4.24 ACCENCS VM access enable register

This register defines which virtual machine encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each VMx.

ACCENCS_VM Offset address: $0D8_H$

ACCENCS VM access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for VM ID q** This bit enables read access to the access protected region for transactions with the VM ID q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for VM ID q** This bit enables write access to the access protected region for transactions with the VM ID q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

35.4.25 ACCENCS PRS access enable register

This register defines which protection register set encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each PRSx.

ACCENCS_PRS Offset address: $0DC_H$

ACCENCS PRS access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for PRS q** This bit enables read access to the access protected region for transactions with the PRS q. 0_B Disabled for read access 1_B Enabled for read access

(table continues...)

(continued)

Field	Bits	Type	Description
WRq (q=00-07)	q+16	rw	**Write access enable for PRS q** This bit enables write access to the access protected region for transactions with the PRS q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

35.4.26 Control register for traps related to CPU0-5

Designated control register for additional features regarding traps.

TRAPS_CPU_CTRL Offset address: $0E8_H$

Control register for traps related to CPU0-5 Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															TRAPFWDEN
r															rw

Field	Bits	Type	Description
TRAPFWDEN	0	rw	**Traps forwarding enable** If enabled, all trap-requests from this security domain will not be gated and could be seen on the other security domain. Default reset value is 0, gating of trap-requests so that they are not seen on the security domain. 0_B No trap request is allowed to pass to another security domain 1_B All trap requests from this security domain are allowed to pass onto another security domain.
0	31:1	r	**Reserved** Read as 0; should be written with 0.

35.4.27 SET/CLR enable register for group g for traps related to CPU0-5

TRAPS_CPU_SETCLRGRPg_SETCLREN (g=0) Offset address: $0EC_H+g*12$

SET/CLR enable register for group g for traps related to CPU0-5 Application Reset value: $0000\ 03F7_H$

TRAPS_CPU_SETCLRGRPg_SETCLREN (g=1-3) Offset address: $0EC_H+g*12$

SET/CLR enable register for group g for traps related to CPU0-5 Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0						SWT	GSM UTCS	SMU TP3	SMU TP2	SMU TP1	SMU TP0	0	ESRT 2	ESRT 1	ESRT 0
r						rw	rw	rw	rw	rw	rw	r	rw	rw	rw

Field	Bits	Type	Description
ESRTx (x=0-2)	x	rw	**Enable setting and clearing of ESRTx for the group g** 0_B Set and clear are disabled from this group for this trap source. 1_B Set and clear are enabled from this group for this trap source.
SMUTPy (y=0-3)	y+4	rw	**Enable setting and clearing of SMUTPy for the group g** 0_B Set and clear are disabled from this group for this trap source. 1_B Set and clear are enabled from this group for this trap source.
GSMUTCS	8	rw	**Enable setting and clearing of GSMUTCS for the group g** 0_B Set and clear are disabled from this group for this trap source. 1_B Set and clear are enabled from this group for this trap source.
SWT	9	rw	**Enable setting and clearing of SWT for the group g** 0_B Set and clear are disabled from this group for this trap source. 1_B Set and clear are enabled from this group for this trap source.
0	3, 31:10	r	**Reserved** Read as 0; should be written with 0.

35.4.28 Trap set register for group g for traps related to CPU0-5

Setting these bits sets corresponding status bits in TRAPS_CPU_STAT if enabled by SETCLREN mask from respective group g.

Reading this bit returns always zero.

TRAPS_CPU_SETCLRGRPg_SET (g=0-3) Offset address: $0F0_H+g*12$

Trap set register for group g for traps related to CPU0-5 Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0						SWT	GSM UTCS	SMU TP3	SMU TP2	SMU TP1	SMU TP0	0	ESRT 2	ESRT 1	ESRT 0
r						w	w	w	w	w	w	r	w	w	w

Field	Bits	Type	Description
ESRTx (x=0-2)	x	w	**Set trap request flag ESRTx** 0_B No effect. 1_B Sets the corresponding trap request flag.
SMUTPx (x=0-3)	x+4	w	**Set trap request flag SMUTPx** 0_B No effect. 1_B Sets the corresponding trap request flag.
GSMUTCS	8	w	**Set trap request flag GSMUTCS** 0_B No effect. 1_B Sets the corresponding trap request flag.
SWT	9	w	**Set trap request flag SWT** 0_B No effect. 1_B Sets the corresponding trap request flag.
0	3, 31:10	r	**Reserved** Read as 0; should be written with 0.

35.4.29 Trap clear register for group g for traps related to CPU0-5

Setting these bits clears corresponding status bits in TRAPS_CPU_STAT if enabled by SETCLREN mask from respective group g.

Reading this bit returns always zero.

TRAPS_CPU_SETCLRGRPg_CLR (g=0-3) Offset address: $0F4_H+g*12$

Trap clear register for group g for traps related to CPU0-5 Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0						SWT	GSMUTCS	SMUTP3	SMUTP2	SMUTP1	SMUTP0	0	ESRT2	ESRT1	ESRT0
r						w	w	w	w	w	w	r	w	w	w

Field	Bits	Type	Description
ESRTx (x=0-2)	x	w	**Clear trap request flag ESRTx** 0_B No effect. 1_B Clears the corresponding trap request flag.
SMUTPx (x=0-3)	x+4	w	**Clear trap request flag SMUTPx** 0_B No effect. 1_B Clears the corresponding trap request flag.

(table continues...)

(continued)

Field	Bits	Type	Description
GSMUTCS	8	w	**Clear trap request flag GSMUTCS** 0_B No effect. 1_B Clears the corresponding trap request flag.
SWT	9	w	**Clear trap request flag SWT** 0_B No effect. 1_B Clears the corresponding trap request flag.
0	3, 31:10	r	**Reserved** Read as 0; should be written with 0.

35.4.30 Trap status register for traps related to CPU0-5

These bits are individually set if a corresponding trap request event is triggered.

Each bit can also be set by setting corresponding bit in TRAPS_CPU_SETCLRGRPg_SET register. Action of setting is protected by respective group APU and masking register SETCLREN.

Each bit can be cleared by setting corresponding bit in TRAPS_CPU_SETCLRGRPg_CLR register. Action of clearing is protected by respective group APU and masking register SETCLREN.

Observed reset value after boot will be 0 for all trap sources, except for ESR0 where it depends upon ARI mode because of ESR pin transition.

TRAPS_CPU_STAT Offset address: $11C_H$

Trap status register for traps related to CPU0-5 Reset values see: Table 1409

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0						SWT	GSMUTCS	SMUTP3	SMUTP2	SMUTP1	SMUTP0	0	ESRT2	ESRT1	ESRT0
r						rh	rh	rh	rh	rh	rh	r	rh	rh	rh

Field	Bits	Type	Description
ESRTx (x=0-2)	x	rh	**ESRx trap request flag** 0_B No trap was requested since this bit was cleared the last time 1_B A trap was requested since this bit was cleared the last time
SMUTPx (x=0-3)	x+4	rh	**SMU trap request flag for SMU partition x** 0_B No trap was requested since this bit was cleared the last time 1_B A trap was requested since this bit was cleared the last time
GSMUTCS	8	rh	**Gated SMU trap request flag related to cyber security** 0_B No trap was requested since this bit was cleared the last time 1_B A trap was requested since this bit was cleared the last time

(table continues...)

(continued)

Field	Bits	Type	Description
SWT	9	rh	**Software trap request flag** 0_B No trap was requested since this bit was cleared the last time 1_B A trap was requested since this bit was cleared the last time
0	3, 31:10	r	**Reserved** Read as 0; should be written with 0.

Table 1409 Reset values of TRAPS_CPU_STAT

Reset type	Reset value	Note
Application Reset	$0000\ 0000_H$	
After Boot-FW Value	0000 0000 0000 0000 0000 0000 0000 $000X_B$	Observed reset value for ESR0T will depend upon ARI mode because of ESR pin transition.

35.4.31 Trap disable register for single CPUy

TRAPS_CPU_DISy (y=0) Offset address: 120_H+y*4

Trap disable register for single CPUy Application Reset value: $0000\ 01E7_H$

TRAPS_CPU_DISy (y=1-5) Offset address: 120_H+y*4

Trap disable register for single CPUy Application Reset value: $0000\ 03F7_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0						SWT	GSM UTCS	SMU TP3	SMU TP2	SMU TP1	SMU TP0	0	ESRT 2	ESRT 1	ESRT 0
r						rw	rw	rw	rw	rw	rw	r	rw	rw	rw

Field	Bits	Type	Description
ESRTx (x=0-2)	x	rw	**Disable trap request ESRTx on CPUy** 0_B A trap can be generated for this source 1_B No trap can be generated for this source
SMUTPx (x=0-3)	x+4	rw	**Disable trap request SMUTPx on CPUy** 0_B A trap can be generated for this source 1_B No trap can be generated for this source
GSMUTCS	8	rw	**Disable trap request GSMUTCS on CPUy** 0_B A trap can be generated for this source 1_B No trap can be generated for this source
SWT	9	rw	**Disable trap request SWT on CPUy** 0_B A trap can be generated for this source 1_B No trap can be generated for this source

(table continues...)

(continued)

Field	Bits	Type	Description
0	3, 31:10	r	**Reserved** Read as 0; should be written with 0.

35.4.32 Control register for traps related to CPUcs

Designated control register for additional features regarding traps.

TRAPS_CS_CTRL Offset address: 138_H

Control register for traps related to CPUcs Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															TRAPFWDEN
r															rw

Field	Bits	Type	Description
TRAPFWDEN	0	rw	**Traps forwarding enable** If enabled, all trap-requests from the CPUcs related trap control logic are forwarded to the CPUy related trap control logic; y=5-0, exact number of implemented CPUs is described in the product specific User´s manual. Default reset value is 0, gating of trap-requests so that they are not seen on other, non security, domain. 0_B No trap request is forwarded to CPUy related trap control logic. 1_B All trap requests are forwarded to CPUy related trap control logic.
0	31:1	r	**Reserved** Read as 0; should be written with 0.

35.4.33 SET/CLR enable register for traps related to CPUcs

TRAPS_CS_SETCLR_SETCLREN Offset address: $13C_H$

SET/CLR enable register for traps related to CPUcs Application Reset value: $0000\ 0300_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0						SWTCS	SMUTCS	GSMUTP3	GSMUTP2	GSMUTP1	GSMUTP0	0	GESRT2	GESRT1	GESRT0
r						rw	rw	rw	rw	rw	rw	r	rw	rw	rw

Field	Bits	Type	Description
GESRTx (x=0-2)	x	rw	**Enable setting and clearing of GESRTx on CPUcs** 0_B Set and clear are not enabled from this group for this trap source. 1_B Set and clear are enabled from this group for this trap source.
GSMUTPy (y=0-3)	y+4	rw	**Enable setting and clearing of GSMUTPy on CPUcs** 0_B Set and clear are not enabled from this group for this trap source. 1_B Set and clear are enabled from this group for this trap source.
SMUTCS	8	rw	**Enable setting and clearing of SMUTCS on CPUcs** 0_B Set and clear are not enabled from this group for this trap source. 1_B Set and clear are enabled from this group for this trap source.
SWTCS	9	rw	**Enable setting and clearing of SWTCS on CPUcs** 0_B Set and clear are not enabled from this group for this trap source. 1_B Set and clear are enabled from this group for this trap source.
0	3, 31:10	r	**Reserved** Read as 0; should be written with 0.

35.4.34 Trap set register for traps related to CPUcs

Setting these bits sets corresponding status bits in TRAPS_CS_STAT if enabled by TRAPCS/SETCLR/SETCLREN mask.

Reading this bit returns always zero. Action of setting is protected by APU-PCS and masking register SETCLREN.

TRAPS_CS_SETCLR_SET Offset address: 140_H

Trap set register for traps related to CPUcs Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0						SWTCS	SMUTCS	GSMUTP3	GSMUTP2	GSMUTP1	GSMUTP0	0	GESRT2	GESRT1	GESRT0
r						w	w	w	w	w	w	r	w	w	w

Field	Bits	Type	Description
GESRTx (x=0-2)	x	w	**Set trap request flag GESRTx** 0_B No effect. 1_B Sets the corresponding trap request flag.
GSMUTPx (x=0-3)	x+4	w	**Set trap request flag GSMUTPx** 0_B No effect. 1_B Sets the corresponding trap request flag.

(table continues...)

(continued)

Field	Bits	Type	Description
SMUTCS	8	w	**Set trap request flag SMUTCS** 0_B No effect. 1_B Sets the corresponding trap request flag.
SWTCS	9	w	**Set trap request flag SWTCS** 0_B No effect. 1_B Sets the corresponding trap request flag.
0	3, 31:10	r	**Reserved** Read as 0; should be written with 0.

35.4.35 Trap clear register for traps related to CPUcs

Setting these bits clears corresponding status bits in TRAPS_CS_STAT if enabled by TRAPCS/SETCLR/SETCLREN mask.

Reading this bit returns always zero. Action of clearing is protected by APUcs and masking register SETCLREN.

TRAPS_CS_SETCLR_CLR Offset address: 144_H

Trap clear register for traps related to CPUcs Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0						SWTCS	SMUTCS	GSMUTP3	GSMUTP2	GSMUTP1	GSMUTP0	0	GESRT2	GESRT1	GESRT0
r						w	w	w	w	w	w	r	w	w	w

Field	Bits	Type	Description
GESRTx (x=0-2)	x	w	**Clear trap request flag GESRTx** 0_B No effect. 1_B Clears the corresponding trap request flag.
GSMUTPx (x=0-3)	x+4	w	**Clear trap request flag GSMUTPx** 0_B No effect. 1_B Clears the corresponding trap request flag.
SMUTCS	8	w	**Clear trap request flag SMUTCS** 0_B No effect. 1_B Clears the corresponding trap request flag.
SWTCS	9	w	**Clear trap request flag SWTCS** 0_B No effect. 1_B Clears the corresponding trap request flag.
0	3, 31:10	r	**Reserved** Read as 0; should be written with 0.

35.4.36 Trap status register for traps related to CPUcs

These bits are individually set if a corresponding trap request event is triggered.
Each bit can also be set by setting corresponding bit in TRAPS_CS_SETCLR_SET register.
Each bit can be cleared by setting corresponding bit in TRAPS_CS_SETCLR_CLR register.
Observed reset value is 0 for all trap sources.

TRAPS_CS_STAT Offset address: 148_H

Trap status register for traps related to CPUcs Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0						SWTCS	SMUTCS	GSMUTP3	GSMUTP2	GSMUTP1	GSMUTP0	0	GESRT2	GESRT1	GESRT0
r						rh	rh	rh	rh	rh	rh	r	rh	rh	rh

Field	Bits	Type	Description
GESRTx (x=0-2)	x	rh	**Gated ESRx trap request flag** 0_B No trap was requested since this bit was cleared the last time 1_B A trap was requested since this bit was cleared the last time
GSMUTPx (x=0-3)	x+4	rh	**Gated SMU Alarm Trap Request Flag for SMU Partition x** 0_B No trap was requested since this bit was cleared the last time 1_B A trap was requested since this bit was cleared the last time
SMUTCS	8	rh	**SMU cyber security alarm trap request flag** 0_B No trap was requested since this bit was cleared the last time 1_B A trap was requested since this bit was cleared the last time
SWTCS	9	rh	**Software trap request flag** 0_B No trap was requested since this bit was cleared the last time 1_B A trap was requested since this bit was cleared the last time
0	3, 31:10	r	**Reserved** Read as 0; should be written with 0.

35.4.37 Trap disable register for CPUcs

TRAPS_CS_DIS Offset address: $14C_H$

Trap disable register for CPUcs Application Reset value: $0000\ 00F7_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0						SWTCS	SMUTCS	GSMUTP3	GSMUTP2	GSMUTP1	GSMUTP0	0	GESRT2	GESRT1	GESRT0
r						rw	rw	rw	rw	rw	rw	r	rw	rw	rw

Field	Bits	Type	Description
GESRTx (x=0-2)	x	rw	**Disable trap request GESRTx on CPUcs** 0_B A trap request can be generated for this source 1_B No trap request can be generated for this source
GSMUTPy (y=0-3)	y+4	rw	**Disable trap request GSMUTPy on CPUcs** 0_B A trap request can be generated for this source 1_B No trap request can be generated for this source
SMUTCS	8	rw	**Disable trap request SMUTCS on CPUcs** 0_B A trap request can be generated for this source 1_B No trap request can be generated for this source
SWTCS	9	rw	**Disable trap request SWTCS on CPUcs** 0_B A trap request can be generated for this source 1_B No trap request can be generated for this source
0	3, 31:10	r	**Reserved** Read as 0; should be written with 0.

35.4.38 External input filter register

The external input filter register enables and configures optional digital glitch filters on the ERS inputs from PORTS.

ERU_EIFILT Offset address: 150_H

External input filter register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DEPTH				FILTDIV				FILRQ7G	FILRQ7B	FILRQ7A	FILRQ6F	FILRQ6B	FILRQ6A	FILRQ5G	FILRQ5E
rw				rw				rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FILRQ5A	FILRQ4G	FILRQ4D	FILRQ4A	FILRQ3C	FILRQ3B	FILRQ3A	FILRQ2C	FILRQ2B	FILRQ2A	FILRQ1G	FILRQ1B	FILRQ1A	FILRQ0F	FILRQ0B	FILRQ0A
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
FILRQ0A	0	rw	**Filter enable for REQ0A** 0_B REQ is unfiltered 1_B REQ glitch filter is enabled
FILRQ0B	1	rw	**Filter enable for REQ0B** 0_B REQ is unfiltered 1_B REQ glitch filter is enabled
FILRQ0F	2	rw	**Filter enable for REQ0F** 0_B REQ is unfiltered 1_B REQ glitch filter is enabled
FILRQ1A	3	rw	**Filter enable for REQ1A** 0_B REQ is unfiltered 1_B REQ glitch filter is enabled
FILRQ1B	4	rw	**Filter enable for REQ1B** 0_B REQ is unfiltered 1_B REQ glitch filter is enabled
FILRQ1G	5	rw	**Filter enable for REQ1G** 0_B REQ is unfiltered 1_B REQ glitch filter is enabled
FILRQ2A	6	rw	**Filter enable for REQ2A** 0_B REQ is unfiltered 1_B REQ glitch filter is enabled
FILRQ2B	7	rw	**Filter enable for REQ2B** 0_B REQ is unfiltered 1_B REQ glitch filter is enabled
FILRQ2C	8	rw	**Filter enable for REQ2C** 0_B REQ is unfiltered 1_B REQ glitch filter is enabled
FILRQ3A	9	rw	**Filter enable for REQ3A** 0_B REQ is unfiltered 1_B REQ glitch filter is enabled
FILRQ3B	10	rw	**Filter enable for REQ3B** 0_B REQ is unfiltered 1_B REQ glitch filter is enabled
FILRQ3C	11	rw	**Filter enable for REQ3C** 0_B REQ is unfiltered 1_B REQ glitch filter is enabled
FILRQ4A	12	rw	**Filter enable for REQ4A** 0_B REQ is unfiltered 1_B REQ glitch filter is enabled

(table continues...)

(continued)

Field	Bits	Type	Description
FILRQ4D	13	rw	**Filter enable for REQ4D** 0_B REQ is unfiltered 1_B REQ glitch filter is enabled
FILRQ4G	14	rw	**Filter enable for REQ4G** 0_B REQ is unfiltered 1_B REQ glitch filter is enabled
FILRQ5A	15	rw	**Filter enable for REQ5A** 0_B REQ is unfiltered 1_B REQ glitch filter is enabled
FILRQ5E	16	rw	**Filter enable for REQ5E** 0_B REQ is unfiltered 1_B REQ glitch filter is enabled
FILRQ5G	17	rw	**Filter enable for REQ5G** 0_B REQ is unfiltered 1_B REQ glitch filter is enabled
FILRQ6A	18	rw	**Filter enable for REQ6A** 0_B REQ is unfiltered 1_B REQ glitch filter is enabled
FILRQ6B	19	rw	**Filter enable for REQ6B** 0_B REQ is unfiltered 1_B REQ glitch filter is enabled
FILRQ6F	20	rw	**Filter enable for REQ6F** 0_B REQ is unfiltered 1_B REQ glitch filter is enabled
FILRQ7A	21	rw	**Filter enable for REQ7A** 0_B REQ is unfiltered 1_B REQ glitch filter is enabled
FILRQ7B	22	rw	**Filter enable for REQ7B** 0_B REQ is unfiltered 1_B REQ glitch filter is enabled
FILRQ7G	23	rw	**Filter enable for REQ7G** 0_B REQ is unfiltered 1_B REQ glitch filter is enabled
FILTDIV	27:24	rw	**Digital glitch filter clock predivider** This field controls a predivider to generate the digital filter sample clock $T_{filt} = T_{spb}$ * FILTDIV A value of zero in this register disables all glitch filtering.

(table continues...)

(continued)

Field	Bits	Type	Description
DEPTH	31:28	rw	**Digital glitch filter depth** DEPTH determines the number of port input samples considered in the calculation of the floating average digital filter output for all enabled FILRQ filters. A value of zero in this register disables all glitch filtering.

35.4.39 External input channel register x

Each external input channel register ERU_EICRx contains bits to configure the external request selection (ERS) and the event trigger logic (ETL).

ERU_EICRx (x=0-7) Offset address: 154_H+x*4

External input channel register x Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	ONP			TPEN	LDEN	REN	FEN	0	EISEL			0		EIEN_P	EIEN
r	rw			rw	rw	rw	rw	r	rw			r		w	rw

Field	Bits	Type	Description
EIEN	0	rw	**External input enable for channel x** Used to enable clock for synchonization logic of external inputs related to this channel. 0_B Synchronization logic clock is disabled 1_B Synchronization logic clock is enabled
EIEN_P	1	w	**Write protection for EIEN** This bit enables ERU_EICRx.EIEN write during ERU_EICRx.EICRx write. Return 0 if read. 0_B EIEN remains unchanged. 1_B EIEN can be changed with write access to register ERU_EICR0
EISEL	6:4	rw	**External input selection** This bit field determines which input line is selected for input channel. 000_B Input A is selected. 001_B Input B is selected. 010_B Input C is selected. 011_B Input D is selected. 100_B Input E is selected. 101_B Input F is selected. 110_B Input G is selected. 111_B Input H is selected.

(table continues...)

(continued)

Field	Bits	Type	Description
FEN	8	rw	**Falling Edge Enable** This bit determines if the falling edge of input channel is used to set bit INTF. 0_B The falling edge is not used 1_B The detection of a falling edge of input channel generates a trigger event. INTF becomes set.
REN	9	rw	**Rising edge enable** This bit determines if the rising edge of input channel is used to set bit INTF. 0_B The rising edge is not used 1_B The detection of a rising edge of input channel generates a trigger event. INTF becomes set.
LDEN	10	rw	**Level detection enable** This bit determines if bit ERU_EIFR.INTF is cleared automatically if an edge of the input channel is detected, which has not been selected (rising edge with REN=0 or falling edge with FEN=0). 0_B Bit INTF will not be cleared 1_B Bit INTF will be cleared
TPEN	11	rw	**Trigger pulse enable** This bit enables the generation of a trigger pulse event when a selected edge is detected. 0_B The trigger event is disabled 1_B The trigger event is enabled
ONP	14:12	rw	**Output node pointer** This bit field determines the destination (output channel) for related trigger pulse event (if enabled by TPEN). 000_B An event from input ETLx triggers output OGU0 (signal TRx0). 001_B An event from input ETLx triggers output OGU1 (signal TRx1) 010_B An event from input ETLx triggers output OGU2 (signal TRx2) 011_B An event from input ETLx triggers output OGU3 (signal TRx3) 100_B An event from input ETLx triggers output OGU4 (signal TRx4) 101_B An event from input ETLx triggers output OGU5 (signal TRx5) 110_B An event from input ETLx triggers output OGU6 (signal TRx6) 111_B An event from input ETLx triggers output OGU7 (signal TRx7)
0	3:2, 7, 31:15	r	**Reserved** Read as 0; should be written with 0.

Table 1410 Access mode restrictions of ERU_EICRx (x=0-7) sorted by descending priority

Mode name	Access mode		Description
write 1 to .**EIEN_P**	rw	EIEN	
(default)	r	EIEN	

35.4.40 External input flag register

The external input flag register ERU_EIFR contains all status flags for the external input channels. The bits in this register can be cleared by software by setting ERU_FMR.FCx, and set by setting ERU_FMR.FSx.

ERU_EIFR Offset address: 174_H

External input flag register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								INTF7	INTF6	INTF5	INTF4	INTF3	INTF2	INTF1	INTF0
r								rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
INTFx (x=0-7)	x	rh	**External event flag of channel x** This bit monitors the status flag of the event trigger condition for the input channel x. This bit is automatically cleared when the selected condition (see ERU_EICRx.REN, ERU_EICRx.FEN) is no longer met (if ERU_EICRx.LDEN='1') or remains set until it is cleared by software (if ERU_EICRx.LDEN='0').
0	31:8	r	**Reserved** Read as 0; should be written with 0.

35.4.41 Flag modification register

If a set event and a clear event for bit ERU_EIFR.INTFx occur at the same time, the set event is taken into account.

ERU_FMR Offset address: 178_H

Flag modification register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								FC7	FC6	FC5	FC4	FC3	FC2	FC1	FC0
r								w	w	w	w	w	w	w	w

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								FS7	FS6	FS5	FS4	FS3	FS2	FS1	FS0
r								w	w	w	w	w	w	w	w

Field	Bits	Type	Description
FSx (x=0-7)	x	w	**Set flag ERU_EIFR.INTFx for channel x** Setting this bit will set the corresponding bit ERU_INTFx in register ERU_EIFR. Reading this bit always delivers a 0. If both FSx and FCx are written with '1' in the same access then set takes precedence. 0_B The bit x in register ERU_EIFR is not modified 1_B The bit x in register ERU_EIFR is set
FCx (x=0-7)	x+16	w	**Clear flag ERU_INTFx for channel x** Setting this bit will clear the corresponding bit ERU_EIFR.INTFx in register ERU_EIFR. Reading this bit always delivers a 0. If both FSx and FCx are written with '1' in the same access then set takes precedence. 0_B The bit x in register ERU_EIFR is not modified 1_B The bit x in register ERU_EIFR is cleared
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

35.4.42 Interrupt gating control register y

Each interrupt gating control registers ERU_IGCRy contains bits to enable the pattern detection and to control the gating for one output channel.

ERU_IGCRy (y=0-7) Offset address: $17C_H+y*4$

Interrupt gating control register y Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IGP		GEEN	0					IPEN7	IPEN6	IPEN5	IPEN4	IPEN3	IPEN2	IPEN1	IPEN0
rw		rw	r					rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
IPENx (x=0-7)	x	rw	**Flag pattern enable for input channel x** Bit IPENx determines if the flag INTFx of channel x takes part in the pattern detection for the gating of the requests for the output signals GOUTy and IOUTy. 0_B The bit INTFx does not take part in the pattern detection 1_B The bit INTFx is taken into consideration for the pattern detection

(table continues...)

(continued)

Field	Bits	Type	Description
GEEN	13	rw	**Generate event enable** Bit GEEN enables the generation of a trigger event for output channel y when the result of the pattern detection changes. If enabled, a trigger (e.g. for an interrupt) is generated during the first clock cycle when a pattern is detected or when pattern is no longer detected. 0_B The trigger generation at a change of the pattern detection result is disabled. 1_B The trigger generation at a change of the pattern detection result is enabled.
IGP	15:14	rw	**Interrupt gating pattern** In each register ERU_IGCRy, bit-field IGP determines how the pattern detection influences the output lines GOUTy and IOUTy. 00_B IOUTy is inactive.The pattern is not considered. 01_B IOUTy is activated in response to a trigger event. The pattern is not considered. 10_B The detected pattern is considered. IOUTy is activated if a trigger event occurs while the pattern is present. 11_B The detected pattern is considered. IOUTy is activated if a trigger event occurs while the pattern is not present.
0	12:8, 31:16	r	**Reserved** Read as 0; should be written with 0.

35.4.43 Pattern detection result register

The Pattern detection result register monitors the combinatorial output status of the pattern detection units.

ERU_PDRR Offset address: $19C_H$

Pattern detection result register Application Reset value: $0000\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								PDR 7	PDR 6	PDR 5	PDR 4	PDR 3	PDR 2	PDR 1	PDR 0
r								rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
PDRy (y=0-7)	y	rh	**Pattern detection result of channel y** This bit indicates the current state of the PDOUTy signal.
0	31:8	r	**Reserved** Read as 0; should be written with 0.

35.4.44 Overlay enable register

OVCENABLE

Overlay enable register

Offset address: $1A0_H$

Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0							GOD	0		OVEN5	OVEN4	OVEN3	OVEN2	OVEN1	OVEN0
r							rw1s	r		rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
OVEN0	0	rw	**Overlay enable 0** 0_B OVC is disabled on CPU0. All Overlay redirections are disabled regardless of the state of OVC0_RABRy.OVEN. 1_B OVC is enabled on CPU0.
OVEN1	1	rw	**Overlay enable 1 (If product has CPU1)** 0_B OVC is disabled on CPU1. All Overlay redirections are disabled regardless of the state of OVC1_RABRy.OVEN. 1_B OVC is enabled on CPU1.
OVEN2	2	rw	**Overlay enable 2 (If product has CPU2)** 0_B OVC is disabled on CPU2. All Overlay redirections are disabled regardless of the state of OVC2_RABRy.OVEN. 1_B OVC is enabled on CPU2.
OVEN3	3	rw	**Overlay enable 3 (If product has CPU3)** 0_B OVC is disabled on CPU3. All Overlay redirections are disabled regardless of the state of OVC3_RABRy.OVEN. 1_B OVC is enabled on CPU3.
OVEN4	4	rw	**Overlay enable 4 (If product has CPU4)** 0_B OVC is disabled on CPU4. All Overlay redirections are disabled regardless of the state of OVC4_RABRy.OVEN. 1_B OVC is enabled on CPU4.
OVEN5	5	rw	**Overlay enable 5 (If product has CPU5)** 0_B OVC is disabled on CPU5. All Overlay redirections are disabled regardless of the state of OVC5_RABRy.OVEN. 1_B OVC is enabled on CPU5.
GOD	8	rw1s	**Global overlay disable** Writing '1' would set this bit. Clearing to '0' is only possible with application reset. 0_B No action 1_B Globally disable overlay function
0	7:6, 31:9	r	**Reserved** Read as 0; should be written with 0.

35.4.45 Overlay control register

CSEL bits related to not implemented CPUs have no effect.

OVCCON Offset address: $1A4_H$

Overlay control register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0						OVCONF_P	OVCONF	0					DCINVAL	OVSTP	OVSTRT
r						w	rw	r					w	w	w

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0										CSEL5	CSEL4	CSEL3	CSEL2	CSEL1	CSEL0
r										w	w	w	w	w	w

Field	Bits	Type	Description
CSEL0	0	w	**CPU select 0** Return 0 if read. 0_B CPU0 not affected. 1_B Action selected by OVSTRT, OVSTP, DCINVAL bits, set by the same register write access, is applied to CPU0.
CSEL1	1	w	**CPU select 1** Return 0 if read. 0_B CPU1 not affected. 1_B Action selected by OVSTRT, OVSTP, DCINVAL bits, set by the same register write access, is applied to CPU1.
CSEL2	2	w	**CPU select 2** Return 0 if read. 0_B CPU2 not affected. 1_B Action selected by OVSTRT, OVSTP, DCINVAL bits, set by the same register write access, is applied to CPU2.
CSEL3	3	w	**CPU select 3** Return 0 if read. 0_B CPU3 not affected. 1_B Action selected by OVSTRT, OVSTP, DCINVAL bits, set by the same register write access, is applied to CPU3.
CSEL4	4	w	**CPU select 4** Return 0 if read. 0_B CPU4 not affected. 1_B Action selected by OVSTRT, OVSTP, DCINVAL bits, set by the same register write access, is applied to CPU4.

(table continues...)

(continued)

Field	Bits	Type	Description
CSEL5	5	w	**CPU select 5** Return 0 if read. 0_B CPU5 not affected. 1_B Action selected by OVSTRT, OVSTP, DCINVAL bits, set by the same register write access, is applied to CPU5.
OVSTRT	16	w	**Overlay start** No action is taken if OVSTP is also set. Return 0 if read. 0_B No action 1_B For each CPU selected with CSEL, all the blocks selected with OVCx_OSEL will be activated. In the selected CPUs all the blocks deselected with OVCx_OSEL will be deactivated.
OVSTP	17	w	**Overlay stop** No action is taken if OVSTRT is also set. Return 0 if read. 0_B No action 1_B For CPUs selected with CSEL, all the overlay blocks are deactivated. OVCx_RABRy.OVEN bits are cleared.
DCINVAL	18	w	**Data cache invalidate** Data cache is affected only in the CPUs selected with CSEL. Return 0 if read. 0_B No action 1_B Data Cache Lines in DMI are invalidated [1)]
OVCONF	24	rw	**Overlay configured** Overlay configured status bit This bit may be used as handshake bit between a debug device (via JTAG interface and Cerberus) and CPU(s). 0_B Overlay is not configured or it has been already started 1_B Overlay block control registers are configured and ready for overlay start
OVCONF_P	25	w	**Write protection for OVCONF** This bit enables OVCONF write during OVCCON write. Return 0 if read. 0_B OVCONF remains unchanged. 1_B OVCONF can be changed with write access to register OVCCON.
0	15:6, 23:19, 31:26	r	**Reserved** Read as 0; should be written with 0.

1) Dirty (modified) cache lines are not affected by this operation. If data cache contains modified data, it is not invalidated, and has to be written-back and invalidated by the user. Therefore, it is highly recommended to either: access overlaid data in read-only mode, or use only non-cached access.

Table 1411 Access mode restrictions of OVCCON sorted by descending priority

Mode name	Access mode		Description
write 1 to .**OVCONF_P**	rw	OVCONF	
(default)	r	OVCONF	

35.4.46 Chip information register

CHIPINFO Offset address: $1A8_H$

Chip information register Reset values see: Table 1412

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								CHTEC		CHREV					
r								rh		rh					

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MANUF											DEPT				
r											r				

Field	Bits	Type	Description
DEPT	4:0	r	**Department identification number** = 00_H: indicates the Automotive & Industrial microcontroller department within Infineon Technologies.
MANUF	15:5	r	**Manufacturer identification number** This is a JEDEC normalized manufacturer code. MANUF = $C1_H$ stands for Infineon Technologies.
CHREV	21:16	rh	**Chip revision number** This bit field indicates the revision number of the TC4Dx device. The value of this bit field is defined in the product Data Sheet. Bits [3:0] are used to indicate the steps. These updates can be done with any metal-fix or FW ROM change. Bits [5:4] define the major silicon design steps (A, B, C, D, ...). These bits can be changed only with a major redesign. 00_H A-step silicon … $0F_H$ A-step silicon 10_H B-step silicon … $1F_H$ B-step silicon 20_H Reserved … $3F_H$ Reserved

(table continues...)

(continued)

Field	Bits	Type	Description
CHTEC	23:22	rh	**Chip family** These bits indicate the product family and are changed only with a redesign. 00_B Reserved 01_B SAx-TC2xx 10_B SAx-TC3xx 11_B SAx-TC4xx
0	31:24	r	**Reserved** Read as 0.

Table 1412 Reset values of CHIPINFO

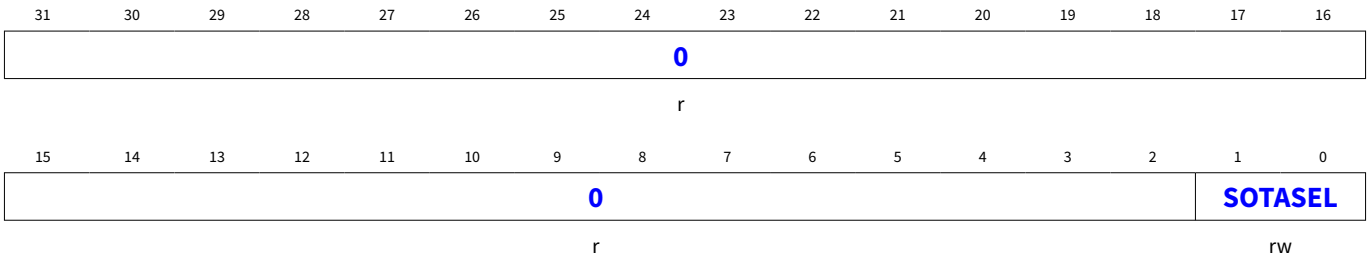
Reset type	Reset value	Note
After Boot-FW Value	0000 0000 11XX XXXX 0001 1000 0010 0000_B	The value of CHREV bit-field is defined in the product Data Sheet.

35.4.47 Software-over-the-air control register

Provides the capability for software control the currently used address region for SOTA.

SOTACTRL Offset address: $1AC_H$

Software-over-the-air control register Reset values see: Table 1413



Field	Bits	Type	Description
SOTASEL	1:0	rw	**Address configuration** Currently used address region (A/B selection) for CPUs other than the CPUcs. 00_B Reserved 01_B 'SOTA-B' mode for CPUs other than the CPUcs 10_B Linear or 'SOTA-A' mode for CPUs other than the CPUcs. 11_B Reserved
0	31:2	r	**Reserved** Read as 0.

Table 1413 Reset values of SOTACTRL

Reset type	Reset value	Note
After Boot-FW Value	0000 0000 0000 0000 0000 0000 0000 00XX_B	SOTASEL value is inserted by Boot Firmware

35.5 Debug information

This section is not applicable for the module being described.

35.6 References

This section is not applicable for the module being described.

35.7 SCU revision history

Reference	Description of change(s)
Date range: 2024-08-17 to 2024-09-04	
ERU output pin connections	• Added ERU output connections to GETH

35.8 TC4Dx SCU information

35.8.1 TC4Dx SCU configuration

There are no deviations from the generic specification.

35.8.2 TC4Dx SCU features

There are no deviations from the generic specification.

35.8.3 TC4Dx SCU functional description

There are no deviations from the generic specification.

35.8.4 TC4Dx SCU registers

35.8.4.1 Register address space - SCU

Table 1414 Registers address space - SCU

Module	Base address	End address	Note
SCU	$F0024000_H$	$F00243FF_H$	SCU: Connections to FPI bus

35.8.4.2 Register overview - access mode glossary

Table 1415 Register overview - access mode glossary

Keyword	Description
CSAE	Access protection using PROT register SCU_PROTCSAE .
CSE	Access protection using PROT register SCU_PROTCSE .
E	Access protection using PROT register SCU_PROTE .
SE	Access protection using PROT register SCU_PROTSE .
APU-PCS	Protection group consisting of registers SCU_ACCENCS_WRA , SCU_ACCENCS_WRB , SCU_ACCENCS_RDA , SCU_ACCENCS_RDB , SCU_ACCENCS_VM , SCU_ACCENCS_PRS .
PCS	Access protection using *APU-PCS* registers.
APU-PGg (g=0-3)	Protection group consisting of registers SCU_ACCENGRPg_WRA , SCU_ACCENGRPg_WRB , SCU_ACCENGRPg_RDA , SCU_ACCENGRPg_RDB , SCU_ACCENGRPg_VM , SCU_ACCENGRPg_PRS .
PGg	Access protection using *APU-PGg* registers.
APU-PNOM	Protection group consisting of registers SCU_ACCENNOM_WRA , SCU_ACCENNOM_WRB , SCU_ACCENNOM_RDA , SCU_ACCENNOM_RDB , SCU_ACCENNOM_VM , SCU_ACCENNOM_PRS .
PNOM	Access protection using *APU-PNOM* registers.
BE	Always returns a Bus Error.
U	No access restrictions.
PROT	Access restrictions as defined in the PROT register access rules.
STP	Access only during start-up, before start-up protection is set (SMM_STCON.ST_PROT=1).

35.8.4.3 Registers overview - SCU (ascending offset address)

Table 1416 Registers overview - SCU (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
SCU_ID	Module identification register	008_H	PNOM	BE	PowerOn Reset	5896
SCU_PROTE	PROT register endinit	018_H	U	SV, PROT	Application Reset	5896

(table continues...)

Table 1416 (continued) Registers overview - SCU (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
SCU_PROTSE	PROT register safe endinit	$01C_H$	U	SV, PROT	Application Reset	5898
SCU_PROTCSE	PROT register cyber-secure endinit	020_H	U	SV, PROT	Application Reset	5899
SCU_PROTCSAE	PROT register cyber-secure APU endinit	024_H	U	SV, PROT	Application Reset	5901
SCU_ACCENNOM_WRA	ACCENNOM write access enable register A	028_H	U	SE, SV	Application Reset	5903
SCU_ACCENNOM_WRB	ACCENNOM write access enable register B	$02C_H$	U	SE, SV	Application Reset	5903
SCU_ACCENNOM_RDA	ACCENNOM read access enable register A	030_H	U	SE, SV	Application Reset	5904
SCU_ACCENNOM_RDB	ACCENNOM read access enable register B	034_H	U	SE, SV	Application Reset	5904
SCU_ACCENNOM_VM	ACCENNOM VM access enable register	038_H	U	SE, SV	Application Reset	5905
SCU_ACCENNOM_PRS	ACCENNOM PRS access enable register	$03C_H$	U	SE, SV	Application Reset	5905
SCU_ACCENGRPg_WRA (g=0-3)	ACCENGRPg write access enable register A	048_H+g*20_H	U	SE, SV	Application Reset	5906
SCU_ACCENGRPg_WRB (g=0-3)	ACCENGRPg write access enable register B	$04C_H$+g*20_H	U	SE, SV	Application Reset	5907
SCU_ACCENGRPg_RDA (g=0-3)	ACCENGRPg read access enable register A	050_H+g*20_H	U	SE, SV	Application Reset	5907
SCU_ACCENGRPg_RDB (g=0-3)	ACCENGRPg read access enable register B	054_H+g*20_H	U	SE, SV	Application Reset	5908
SCU_ACCENGRPg_VM (g=0-3)	ACCENGRPg VM access enable register	058_H+g*20_H	U	SE, SV	Application Reset	5908
SCU_ACCENGRPg_PRS (g=0-3)	ACCENGRPg PRS access enable register	$05C_H$+g*20_H	U	SE, SV	Application Reset	5909
SCU_ACCENCS_WRA	ACCENCS write access enable register A	$0C8_H$	U	CSAE, SV	Application Reset	5909
SCU_ACCENCS_WRB	ACCENCS write access enable register B	$0CC_H$	U	CSAE, SV	Application Reset	5910
SCU_ACCENCS_RDA	ACCENCS read access enable register A	$0D0_H$	U	CSAE, SV	Application Reset	5910

(table continues...)

Table 1416 (continued) Registers overview - SCU (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
SCU_ACCENCS_RDB	ACCENCS read access enable register B	$0D4_H$	U	CSAE, SV	Application Reset	5911
SCU_ACCENCS_VM	ACCENCS VM access enable register	$0D8_H$	U	CSAE, SV	Application Reset	5911
SCU_ACCENCS_PRS	ACCENCS PRS access enable register	$0DC_H$	U	CSAE, SV	Application Reset	5912
SCU_TRAPS_CPU_CTRL	Control register for traps related to CPU0-5	$0E8_H$	PNOM	SV, PNOM	Application Reset	5913
SCU_TRAPS_CPU_SETCLRGRPg_SETCLREN (g=0-3)	SET/CLR enable register for group g for traps related to CPU0-5	$0EC_H$+g*12	PNOM	E, SV, PNOM	Application Reset	5913
SCU_TRAPS_CPU_SETCLRGRPg_SET (g=0-3)	Trap set register for group g for traps related to CPU0-5	$0F0_H$+g*12	PGg	PGg, SV	Application Reset	5914
SCU_TRAPS_CPU_SETCLRGRPg_CLR (g=0-3)	Trap clear register for group g for traps related to CPU0-5	$0F4_H$+g*12	PGg	PGg, SV	Application Reset	5915
SCU_TRAPS_CPU_STAT	Trap status register for traps related to CPU0-5	$11C_H$	PNOM	BE	5916	5916
SCU_TRAPS_CPU_DISy (y=0-5)	Trap disable register for single CPUy	120_H+y*4	PNOM	E, SV, PNOM	Application Reset	5917
SCU_TRAPS_CS_CTRL	Control register for traps related to CPUcs	138_H	PCS	PCS, SV	Application Reset	5918
SCU_TRAPS_CS_SETCLR_SETCLREN	SET/CLR enable register for traps related to CPUcs	$13C_H$	PCS	CSE, SV, PCS	Application Reset	5918
SCU_TRAPS_CS_SETCLR_SET	Trap set register for traps related to CPUcs	140_H	PCS	PCS, SV	Application Reset	5919
SCU_TRAPS_CS_SETCLR_CLR	Trap clear register for traps related to CPUcs	144_H	PCS	PCS, SV	Application Reset	5920
SCU_TRAPS_CS_STAT	Trap status register for traps related to CPUcs	148_H	PCS	BE	Application Reset	5921
SCU_TRAPS_CS_DIS	Trap disable register for CPUcs	$14C_H$	PCS	CSE, SV, PCS	Application Reset	5921
SCU_ERU_EIFILT	External input filter register	150_H	PNOM	E, SV, PNOM	Application Reset	5922
SCU_ERU_EICRx (x=0-7)	External input channel register x	154_H+x*4	PNOM	E, SV, PNOM	Application Reset	5925

(table continues...)

Table 1416 (continued) Registers overview - SCU (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
SCU_ERU_EIFR	External input flag register	174_H	PNOM	BE	Application Reset	5927
SCU_ERU_FMR	Flag modification register	178_H	PNOM	PNOM, SV	Application Reset	5927
SCU_ERU_IGCRy (y=0-7)	Interrupt gating control register y	$17C_H$+y*4	PNOM	E, SV, PNOM	Application Reset	5928
SCU_ERU_PDRR	Pattern detection result register	$19C_H$	PNOM	BE	Application Reset	5929
SCU_OVCENABLE	Overlay enable register	$1A0_H$	PNOM	E, SV, PNOM	Application Reset	5930
SCU_OVCCON	Overlay control register	$1A4_H$	PNOM	PNOM, SV	Application Reset	5931
SCU_CHIPINFO	Chip information register	$1A8_H$	PNOM	BE	5933	5933
SCU_SOTACTRL	Software-over-the-air control register	$1AC_H$	PNOM	PNOM, STP	5934	5934

35.8.4.4 Device specific registers

There are no device specific register changes.

35.8.5 TC4Dx SCU connectivity

The table below lists interfaces to and from this functional block to other blocks in the device.

Table 1417 List of SCU interface signals

Interface signals	I/O	Description
CLOCK_SCU_fSPB	In	fSPB
SMU_SCU_PxTRAPREQ	In	TRAP: Non-secure SMU partitions request x=[3:0]
SMU_SCU_CSTRAPREQ	In	TRAP: Secure SMU partition trap req
SMM_SCU_ESRxTRAPREQ	In	TRAP: ESR trap request x=[2:0]
SCU_CPU0_TRAP	Out	TRAP: NMI to CPU
SCU_CPU1_TRAP	Out	TRAP: NMI to CPU
SCU_CPU2_TRAP	Out	TRAP: NMI to CPU
SCU_CPU3_TRAP	Out	TRAP: NMI to CPU
SCU_CPU4_TRAP	Out	TRAP: NMI to CPU
SCU_CPU5_TRAP	Out	TRAP: NMI to CPU
SCU_CPUCS_CSTRAP	Out	TRAP: NMI to CPUcs
PORTS_SCU_E_REQ0A	In	ERU: PORTS inputs
PORTS_SCU_E_REQ0B	In	ERU: PORTS inputs
PORTS_SCU_E_REQ0F	In	ERU: PORTS inputs
PORTS_SCU_E_REQ1A	In	ERU: PORTS inputs
PORTS_SCU_E_REQ1B	In	ERU: PORTS inputs
PORTS_SCU_E_REQ1G	In	ERU: PORTS inputs
PORTS_SCU_E_REQ2A	In	ERU: PORTS inputs
PORTS_SCU_E_REQ2B	In	ERU: PORTS inputs
PORTS_SCU_E_REQ2C	In	ERU: PORTS inputs
PORTS_SCU_E_REQ3A	In	ERU: PORTS inputs
PORTS_SCU_E_REQ3B	In	ERU: PORTS inputs
PORTS_SCU_E_REQ3C	In	ERU: PORTS inputs
PORTS_SCU_E_REQ4A	In	ERU: PORTS inputs
PORTS_SCU_E_REQ4D	In	ERU: PORTS inputs
PORTS_SCU_E_REQ4G	In	ERU: PORTS inputs
PORTS_SCU_E_REQ5A	In	ERU: PORTS inputs
PORTS_SCU_E_REQ5E	In	ERU: PORTS inputs
PORTS_SCU_E_REQ5G	In	ERU: PORTS inputs

(table continues...)

Table 1417 (continued) List of SCU interface signals

Interface signals	I/O	Description
PORTS_SCU_E_REQ6A	In	ERU: PORTS inputs
PORTS_SCU_E_REQ6B	In	ERU: PORTS inputs
PORTS_SCU_E_REQ6F	In	ERU: PORTS inputs
PORTS_SCU_E_REQ7A	In	ERU: PORTS inputs
PORTS_SCU_E_REQ7B	In	ERU: PORTS inputs
PORTS_SCU_E_REQ7G	In	ERU: PORTS inputs
MSC0_SCU_FCLP	In	ERU: MSC FCLP input
CLOCK_SCU_EXTCLK0	In	ERU: EXTCLK0 input
CLOCK_SCU_EXTCLK1	In	ERU: EXTCLK1 input
CPU0_SCU_STM_RST_REQ	In	ERU: STM input
CPU1_SCU_STM_RST_REQ	In	ERU: STM input
CPU2_SCU_STM_RST_REQ	In	ERU: STM input
CPU3_SCU_STM_RST_REQ	In	ERU: STM input
CPU4_SCU_STM_RST_REQ	In	ERU: STM input
CPU5_SCU_STM_RST_REQ	In	ERU: STM input
SMM_SCU_ESR0	In	ERU: SMM ESR input
SMM_SCU_ESR1	In	ERU: SMM ESR input
SMM_SCU_ESR2	In	ERU: SMM ESR input
ERAY0_SCU_MACROTICK	In	ERU: ERAY MACROTICK input
ERAY1_SCU_MACROTICK	In	ERU: ERAY MACROTICK input
EGTM_SCU_TRIG[3:0]	In	ERU: EGTM inputs
SCU_IR_ERU[7:0]	Out	ERU: Interrupts
SCU_SMU_SFF_ALARM	Out	SCU SFF uncorrectable error alarm
SCU_SMU_ALM_SCUERU[7:0]	Out	ERU: Alarms
SCU_ADC_ERU[7:0]	Out	ERU: ADC outputs
SCU_ERAY0_STPWT[3:0]	Out	ERU: ERAY STPWT outputs
SCU_ERAY1_STPWT[3:0]	Out	ERU: ERAY STPWT outputs
SCU_EGTM_ERU_TIMnINSEL	Out	ERU: EGTM TIM output n=[2:0]
SCU_CPU0_OVEN	Out	Overlay enable output
SCU_CPU1_OVEN	Out	Overlay enable output
SCU_CPU2_OVEN	Out	Overlay enable output
SCU_CPU3_OVEN	Out	Overlay enable output
SCU_CPU4_OVEN	Out	Overlay enable output

(table continues...)

Table 1417 (continued) List of SCU interface signals

Interface signals	I/O	Description
SCU_CPU5_OVEN	Out	Overlay enable output
SCU_CPU0_OVSTART	Out	Overlay start output
SCU_CPU1_OVSTART	Out	Overlay start output
SCU_CPU2_OVSTART	Out	Overlay start output
SCU_CPU3_OVSTART	Out	Overlay start output
SCU_CPU4_OVSTART	Out	Overlay start output
SCU_CPU5_OVSTART	Out	Overlay start output
SCU_CPU0_OVSTOP	Out	Overlay stop output
SCU_CPU1_OVSTOP	Out	Overlay stop output
SCU_CPU2_OVSTOP	Out	Overlay stop output
SCU_CPU3_OVSTOP	Out	Overlay stop output
SCU_CPU4_OVSTOP	Out	Overlay stop output
SCU_CPU5_OVSTOP	Out	Overlay stop output
SCU_CPU0_OVDCINVAL	Out	Overlay data-cache invalidate output
SCU_CPU1_OVDCINVAL	Out	Overlay data-cache invalidate output
SCU_CPU2_OVDCINVAL	Out	Overlay data-cache invalidate output
SCU_CPU3_OVDCINVAL	Out	Overlay data-cache invalidate output
SCU_CPU4_OVDCINVAL	Out	Overlay data-cache invalidate output
SCU_CPU5_OVDCINVAL	Out	Overlay data-cache invalidate output
SCU_CPU0_SOTASEL[1:0]	Out	Double-bit for SOTA swap control
SCU_CPU1_SOTASEL[1:0]	Out	Double-bit for SOTA swap control
SCU_CPU2_SOTASEL[1:0]	Out	Double-bit for SOTA swap control
SCU_CPU3_SOTASEL[1:0]	Out	Double-bit for SOTA swap control
SCU_CPU4_SOTASEL[1:0]	Out	Double-bit for SOTA swap control
SCU_CPU5_SOTASEL[1:0]	Out	Double-bit for SOTA swap control

35.8.6 TC4Dx SCU revision history

Reference	Description of change(s)
Date range: 2024-08-17 to 2024-09-04	
TC4Dx SCU connectivity	• Removed ERU output connections to LETH (LETH AUX feature not available in TC4Dx)

36 Power Management System (PMS)

The Power Management System (PMS) provides supply voltage selection, internal supply voltage generation, supply voltage monitoring, power domain and isolation control, and power management functions for the microcontroller. It also supports other microcontroller infrastructure functions through reset triggers, standby mode transitions, and real-time clock services.

36.1 Feature list

Power Management System

- Integrated on-chip voltage regulators enabling a 5 V ± 10% or 3.3 V ± 10% single source power supply concept. External supply on all rails also supported
- Core buck regulator (EVRC) to generate V_{DD} core supply from $V_{DDEXTDC}$ input supply up to 7 V (depending upon product variant) to enable MOSFET supply from external pre-regulator. Synchronization support with external DCDC regulators
- Static voltage scaling support between 0.95 V – 1.025 V ± 10%, depending on product variant, in steps of 2.83 mV (using the embedded voltage regulator EVRC), depending on product variant and device type
- Primary under-voltage monitoring of supply rails including V_{DD}, V_{DDEXT},$V_{DDEXTHS}$, ($V_{DDEXTDC}$-V_{SSDCHS}), $V_{DDEXTDC}$, V_{DDDCLS}, $V_{DDEVRSB}$, V_{DD3PMS}, $V_{DDP3NVM}$, V_{DDM}, $V_{DDPHPHYx}$, $V_{DDPHRIFx}$, V_{DDHSIF}, V_{DDPAD}, V_{DDCPUx}, V_{DDPPU}, V_{DDLMUx}, V_{DDPHYx}, and V_{DDRIFx}, depending on product variant (not all supply rails are available on each product variant), irrespective of whether externally supplied or internally generated, to put the microcontroller into cold reset state
- Secondary independent over and under-voltage monitoring on $V_{DDEVRSB}$, V_{DD3PMS}, V_{DDEXT},$V_{DDEXTHS}$, $V_{DDP3NVM}$, V_{DDM}, V_{DDFLEX}, V_{DD}, V_{DDPMS0}, V_{DDPMS1}, V_{DDPMS2}, $V_{DDSBRAM}$, V_{DDPAD}, $V_{DDPHPHYx}$, $V_{DDPHRIFx}$, V_{DDHSIF}, V_{DDPHYx}, V_{DDRIFx}, V_{DDCPUx}, V_{DDPPU}, V_{DDLMUx}, and $V_{DDEXTDC}$ supplies, depending on product variant (not all supply rails are available on each product variant). In case of violation of the thresholds, alarms are reported to the SMU (safety and security alarm management unit) and CSRM (cyber security real-time module)
- Support of multiple standby power domains PMS0, PMS1, and PMS2 supplied by separate standby supply pin ($V_{DDEVRSB}$) and internal standby core regulators to meet low quiescent standby current targets. Wake-up possible from RTC, WUT, pin events, and V_{DDEXT} supply ramp events
- 256 kB standby RAM support split across CPU0 dLMU and CPU1 dLMU and 32 kB SCR XRAM supplied in standby mode, depending on product variant
- Real Time Clock and timer support with external 32 kHz crystal with temperature trimming, depending on product variant
- Enhanced standby controller subsystem running up to maximum frequency of 100 MHz and with 32 kB XRAM, depending on product variant

Power distribution

- Support of multiple core power domains for CPU, PPU, and LMU functional blocks (depending on product variant) to reduce overall power consumption for use cases where the modules are not required
- Separate ADC power domain supplied by V_{DDM}/V_{SSM} supply rails to ensure minimum noise interference from the digital core and pad domains. Separate V_{DDM}/V_{AREF} reference supply for ADC modules
- Mixed IO support
  - Flexport (V_{DDFLEX}) has separate dedicated supply pins and may be supplied with either 5 V or 3.3 V
  - ADC channels supported in dedicated analog and shared digital port (P00/P33) domains
- XTAL/Oscillator/PLL supply with separate supply and ground pins for minimal noise interference
- Separate port domain for standby controller and standby functions supplied by $V_{DDEVRSB}$ supply rail
- Separate port power domains for high speed communication interfaces
  - SGMIIx and PCIe modules supplied by dedicated $V_{DDPHPHYx}$ and V_{DDPHYx} supply rails
  - NOR Flash interface or ethernet RGMII supplied by dedicated V_{DDHSIF} supply rail
  - Ethernet RMII/MII supplied by dedicated V_{DDFLEX} supply rail

36.2 Functional overview

The PMS functional block constitutes the power management infrastructure, control of power related sub-functions of all modules, control of multiple power domains and voltage rails, supply generation using internal regulators, supply voltage monitors, and system power transitions.

On-chip voltage regulators are implemented in TC4xx thereby enabling a single source power supply concept. The external nominal system supply from external regulator may be either 5 V or 3.3 V. The embedded core voltage regulator (EVRC) in turn may be used to generate the V_{DD} core supply for the core domain depending on hardware configuration pin settings. It is also possible to supply all supply pin voltages externally. All supply and generated voltages need to be available and within the operational limits to release the reset. The primary voltage monitors check for brownout conditions, setting the device into cold power-on reset state in case any voltage is below the operational limits.

This power management block ensures adequate power delivery to all modules, load transient management, and interfaces to external regulators.

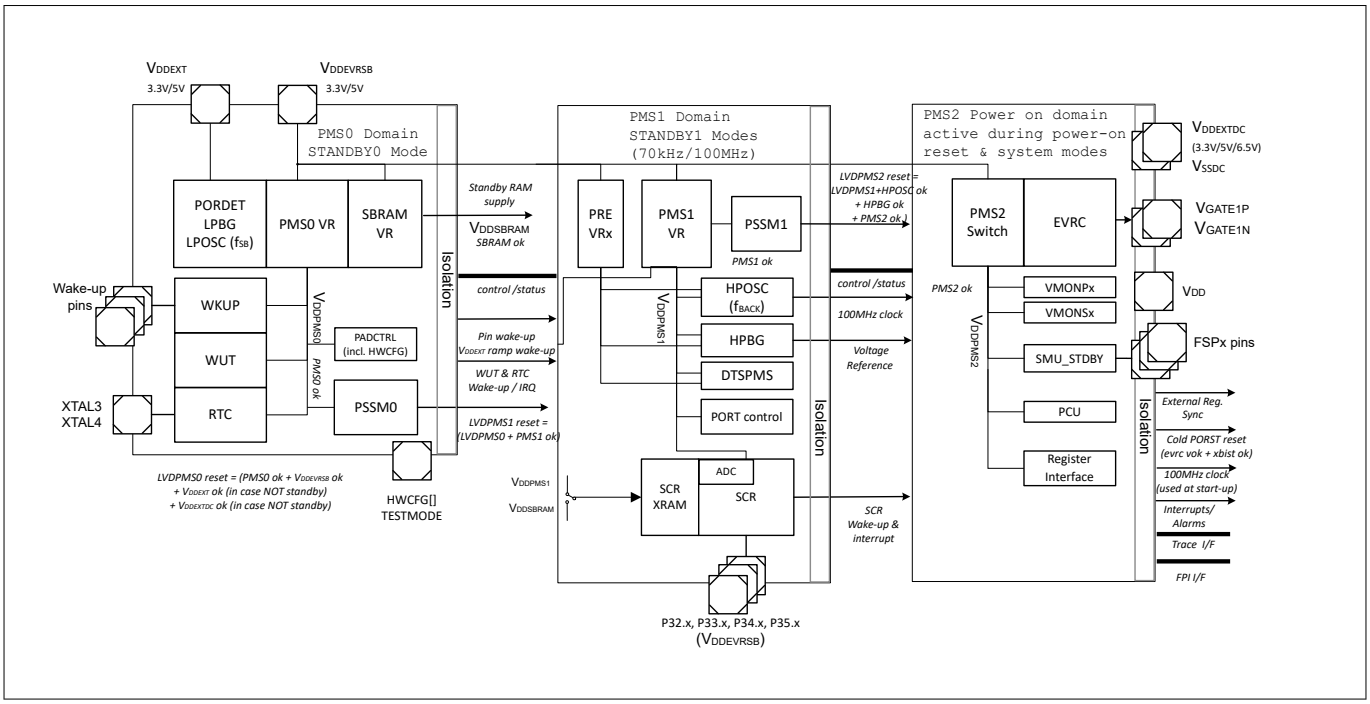


Figure 774 PMS block diagram (not all features are available for all variants)

For the alarm description please refer to the alarm mapping tables in the SMU functional block and the Safety Manual.

Related information

TC4Dx SMU alarm mapping tables on page 7233

36.3 Functional description

This part describes the functionality of the PMS. Details on the various functions are provided in the enclosed sections.

36.3.1 Power supply rails and distribution

The following supply rails are implemented on TC4xx:

The following table provides an overview of the external supply rails. Refer to the functional description for the specific product variant for detailed information.

Table 1418 External supply rails

Voltage rail	Description
$V_{DDEVRSB}$	Standby domain supply. Separate dedicated standby supply for: • Standby pad domain (P32/33/34/35) • Standby power domains (PMS0, PMS1, PMS2) • Standby controller (SCR) 3.3 V or 5 V depending on application requirements
V_{DD3PMS}	Dedicated PMS supply for Power Management components. Generated by internal EVR3PMS voltage regulator (not available in all derivatives). 2.7V
$V_{DDEXTHS}$	I/O port supply. Separate dedicated supply for high speed communication modules. 3.3 V or 5 V depending on application requirements $V_{DDEXTHS_SENSE}$ sense supply is an unloaded $V_{DDEXTHS}$ pin to sense internal $V_{DDEXTHS}$ voltage to account for voltage drops.
V_{DDEXT}	I/O port supply. Main IO pad supply for ports and pins. 3.3 V or 5 V depending on application requirements V_{DDEXT_SENSE} sense supply is an unloaded V_{DDEXT} pin to sense internal V_{DDEXT} voltage to account for voltage drops.
$V_{DDEXTOSC}$	I/O supply for XTAL pads. Supply for PLLs, Clock generation and VTMON logic. 3.3 V or 5 V depending on application requirements
$V_{DDP3NVMx}$	Supply voltage for non-volatile memory 3.3 V
V_{DDM}	Separate dedicated ADC VDDM supply for ADC and DSADC modules. 3.3 V or 5 V depending on application requirements
V_{DDFLEX}	Flexport supply. Separate dedicated supply for high speed communication modules. 3.3 V or 5 V depending on application requirements

(table continues...)

Table 1418 (continued) External supply rails

Voltage rail	Description
$V_{DDEXTDC}$	This is the supply of the EVRC and of the external MOSFETs. EVRC and external MOSFETs must be supplied with the same voltage. Dedicated supply for the EVRC, MOSFET gate drivers, and for the external MOSFETs. 3.3 V to 6.5 V depending on application requirements
$V_{DDPHPHY0}$	High-speed PHY 0 supply. Separate dedicated IO and module supply for high-speed PHY 0 module 1.8 V
$V_{DDPHPHY1}$	High-speed PHY 1 supply. Separate dedicated IO and module supply for high-speed PHY 1 module 1.8 V
$V_{DDPHPHY2}$	High-speed PHY 2 supply. Separate dedicated IO and module supply for high-speed PHY 2 module 1.8 V
$V_{DDP3RIFx}$	Radar interface 0 supply (only on certain product variants). To be connected to V_{DDEXT}. Input supply for the embedded D-PHY LDO. 3.3 V
$V_{DDPHRIFx}$	Radar interface 0 supply (only on certain product variants). Separate dedicated IO and module supply for CSI2 D-PHY module. This is connected to the output of the embedded D-PHY LDO. If using the embedded D-PHY LDO, then this supply pin should be connected to an external capacitor. Only if not using the embedded D-PHY LDO, a separate 1.8 V supply can be connected to this pin. During start-up, including LBIST, the embedded D-PHY LDO is disabled. It can be enabled after the start-up, by the application software. Therefore, the application must ensure that the D-PHY pins are not externally driven with a voltage level above or equal to 0.7 V, as long as the $V_{DDPHRIF0}$ is not powered. 1.8 V
V_{DDHSIF}	RGMII or NOR Flash I/F supply. Separate dedicated IO and module supply for the xSPI interface and RGMII module. 3.3 V for RGMII usage 1.8 V or 3.3 V for xSPI usage, depending on application requirements

(table continues...)

Table 1418 (continued) External supply rails

Voltage rail	Description
V_{DD}	Core supply of the device. The supply is split internally in domain supplies V_{DDCPUx}, V_{DDPPU}, and V_{DDLMU}. Depending on product variant, either: • or 1.0 V on slow and typical (POR) samples, respectively 0.95 V on fast samples • or 1.025 V on slow and typical (POR) samples, respectively 0.975 V on fast samples The same voltage split applies to all supply rails with typical value of 1.0 V/ 0.95 V, or 1.025 V/ 0.975 V V_{DD_SENSE} sense supply is an unloaded VDD pin to sense internal VDD voltage to account for voltage drops.
V_{DDPHY0}	Ethernet SGMII0 core and IO supply, or debugging interface core and IO supply (depending on product variant).
V_{DDPHY1}	Ethernet SGMII1 or PCIe core and IO supply. Separate dedicated core supply for SGMII1 or PCIe module
V_{DDPHY2}	Debugging interface core and IO supply (only on certain product variants). Separate dedicated core supply for SGBT debugging module
V_{DDRIFx}	Radar interface core and IO supply (only on certain product variants). Separate dedicated core supply for CSI2 D-PHY module
V_{DDSB}	Dedicated LMU supply

The following table provides an overview of the internal supply rails. Depending upon the product variants all supply rails may not be available. Refer to the functional description for the specific product variant for detailed information.

Table 1419 Internal power domains

Voltage rail	Description
V_{SSDCHS}	EVRC MOSFET driver supply. Internal elevated ground supply for VGATE1P P channel driver. • 0 V if $V_{DDEXTDC}$ = 3.3 V/5 V • 1.5 V if $V_{DDEXTDC}$ = 6.5 V 0 V - 1.5 V
V_{DDDCLS}	EVRC MOSFET driver supply. Internal supply for VGATE1N N channel driver. • 3.3 V if $V_{DDEXTDC}$ = 3.3 V • 5 V if $V_{DDEXTDC}$ = 5 V/6.5 V 3.3 V - 5 V
V_{DDLMU0}	Switchable LMU power domain. Internal supply rail for the LMU.

(table continues...)

Table 1419 (continued) Internal power domains

Voltage rail	Description
V_{DDLMU1}	Switchable LMU power domain. Internal supply rail for the LMU.
V_{DDPAD}	V_{DD} pads supply voltage.
V_{DDPPU}	Switchable PPU power supply. Internal supply rail after power domain switches.
V_{DDPMS0}	Internal core supply for PMS0 domain. 0.95 V
V_{DDPMS1}	Internal core supply for PMS1 domain. 0.95 V/1.0 V
V_{DDPMS2}	Internal core supply for PMS2 domain. 0.95 V/1.0 V
$V_{DDSBRAM}$	Internal core supply for Standby RAMs. 0.95 V

The following table shows the state of the supply rails during the different system modes. The mentioned system modes are defined in the standby mode management (SMM) chapter. The names STANDBY0 ($V_{DDEVRSB}$) and STANDBY1 ($V_{DDEVRSB}$) denote the STANDBY0 mode respectively STANDBY1 mode where only $V_{DDEVRSB}$ is supplied. The names STANDBY0 ($V_{DDEVRSB}$ + V_{DDEXT}) and STANDBY1 ($V_{DDEVRSB}$ + V_{DDEXT}) denote the STANDBY0 mode respectively STANDBY1 mode where both $V_{DDEVRSB}$ and V_{DDEXT} are supplied.

Table 1420 Supply rail states during different system modes

Voltage rail	Start-up till cold PORST release	RUN0 mode	RUN1 mode	Cranking[1)]	STANDBY0 ($V_{DDEVRSB}$)	STANDBY0 ($V_{DDEVRSB}$+ V_{DDEXT})	STANDBY1 ($V_{DDEVRSB}$)	STANDBY1 ($V_{DDEVRSB}$+ V_{DDEXT})
$V_{DDEVRSB}$ V_{DD3PMS}	ON	ON	ON	ON	ON	ON	ON	ON
V_{DDEXT} $V_{DDEXTHS}$	ON	ON	ON	ON	OFF	ON	OFF	ON
V_{DDM}	Supply ramp-up phase. Supplied modules in reset	ON	ON	ON	OFF	ON/OFF[2)]	OFF	ON/OFF[2)]

(table continues...)

Table 1420 (continued) Supply rail states during different system modes

Voltage rail	Start-up till cold PORST release	RUN0 mode	RUN1 mode	Cranking[1]	STANDBY0 ($V_{DDEVRSB}$)	STANDBY0 ($V_{DDEVRSB}$+ V_{DDEXT})	STANDBY1 ($V_{DDEVRSB}$)	STANDBY1 ($V_{DDEVRSB}$+ V_{DDEXT})
V_{DDFLEX}	Supply ramp-up phase. Supplied modules in reset	ON	ON	ON	OFF	ON/OFF[2]	OFF	ON/OFF[2]
$V_{DDP3NVM}$	Supply ramp-up phase. Supplied modules in reset	ON	ON	ON	OFF	OFF/ON[3]	OFF	OFF/ON[3]
$V_{DDP3RIFx}$	Supply ramp-up phase. Supplied modules in reset	ON	ON	ON	OFF	OFF/ON[3]	OFF	OFF/ON[3]
$V_{DDEXTDC}$	ON	ON	ON	ON	OFF	OFF/ON[3] [4]	OFF	OFF/ON[3] [4]
V_{SSDCHS}	ON/OFF[5]	ON/OFF[5]	ON/ OFF[5]	ON/OFF[5]	OFF	ON/OFF[5]	OFF	ON/OFF[5]
V_{DDDCLS}	ON	ON	ON	ON	OFF	OFF	OFF	OFF
$V_{DDPHPHY0}$	Supply ramp-up phase. Supplied modules in reset.	ON/OFF[6]	ON/ OFF[6]	ON/OFF[6]	OFF[7]	OFF[7]	OFF[7]	OFF[7]
$V_{DDPHPHY1}$	Supply ramp-up phase. Supplied modules in reset.	ON/OFF[6]	ON/ OFF[6]	ON/OFF[6]	OFF[7]	OFF[7]	OFF[7]	OFF[7]
$V_{DDPHPHY2}$	Supply ramp-up phase. Supplied modules in reset.	ON/OFF[6]	ON/ OFF[6]	ON/OFF[6]	OFF[7]	OFF[7]	OFF[7]	OFF[7]

(table continues...)

Table 1420 (continued) Supply rail states during different system modes

Voltage rail	Start-up till cold PORST release	RUN0 mode	RUN1 mode	Cranking[1)]	STANDBY0 ($V_{DDEVRSB}$)	STANDBY0 ($V_{DDEVRSB}$+ V_{DDEXT})	STANDBY1 ($V_{DDEVRSB}$)	STANDBY1 ($V_{DDEVRSB}$+ V_{DDEXT})
$V_{DDPHRIFx}$	Supply ramp-up phase. Supplied modules in reset.	ON[8)]	ON[8)]	ON[8)]	OFF	OFF	OFF	OFF
V_{DDHSIF}	Supply ramp-up phase. Supplied modules in reset.	ON	ON	ON	OFF	OFF	OFF	OFF
V_{DD}	Supply ramp-up phase. Supplied modules in reset.	ON	ON	ON	OFF	OFF	OFF	OFF
V_{DDPPU}	Supply ramp-up phase. Supplied modules in reset.	ON	ON/ OFF#unique_362 / unique_362_Connect_42_fn_fdf_ygl_pkb	ON	OFF	OFF	OFF	OFF
V_{DDLMU}	Supply ramp-up phase. Supplied modules in reset.	ON	ON/ OFF#unique_362 / unique_362_Connect_42_fn_fdf_ygl_pkb	ON	OFF	OFF	OFF	OFF
V_{DDPHY0}	Supply ramp-up phase. Supplied modules in reset.	ON[9)]	ON[9)]	ON[9)]	OFF	OFF	OFF	OFF

(table continues...)

Table 1420 (continued) Supply rail states during different system modes

Voltage rail	Start-up till cold PORST release	RUN0 mode	RUN1 mode	Cranking[1]	STANDBY0 ($V_{DDEVRSB}$)	STANDBY0 ($V_{DDEVRSB}$+ V_{DDEXT})	STANDBY1 ($V_{DDEVRSB}$)	STANDBY1 ($V_{DDEVRSB}$+ V_{DDEXT})
V_{DDPHY1}	Supply ramp-up phase. Supplied modules in reset.	ON[9]	ON[9]	ON[9]	OFF	OFF	OFF	OFF
V_{DDPHY2}	Supply ramp-up phase. Supplied modules in reset.	ON[9]	ON[9]	ON[9]	OFF	OFF	OFF	OFF
V_{DDRIFx}	Supply ramp-up phase. Supplied modules in reset.	ON	ON	ON	OFF	OFF	OFF	OFF
V_{DDSB}	Supply ramp-up phase. Supplied modules in reset.	ON	ON	ON	OFF	OFF	OFF	OFF
V_{DDPMS0}	ON	ON	ON	ON	ON	ON	ON	ON
V_{DDPMS1}	ON	ON	ON	ON	OFF	OFF	ON	ON
V_{DDPMS2}	ON	ON	ON	ON	OFF	OFF	OFF	OFF
$V_{DDSBRAM}$	ON	ON	ON	ON	ON	ON	ON	ON

1) See cranking voltage ranges in the datasheet.
2) Depending on external regulator capabilities: ON if supply during standby is supported, OFF otherwise.
3) Depending on external regulator capabilities: OFF if switch-off independently from V_{DDEXT} during standby is supported, ON otherwise.
4) If $V_{DDEXTDC}$ remains supplied in standby mode, the additional current I_{DCSTBY} is drawn, as specified in the datasheet.
5) ON if $V_{DDEXTDC}$>5 V (depending on HWCFG[2:1] setting), OFF otherwise
6) ON if at least one high-speed PHY is used, OFF otherwise. If no PHY is used, it is recommended to tie to GND via resistor
7) Wake-up from standby possible through wake-up capable pins (see section on standby exit and wake-up sources).
8) The embedded LDO must be enabled after start-up and after LBIST by application SW, or, alternatively, a 1.8 V external supply can be used. While the $V_{DDPHRIFx}$ supply is not powered, all D-PHY pins must not be driven with a voltage level above or equal to 0.7 V.
9) Required by key-on LBIST to supply V_{DDPHYx} even if the respective high-speed PHY is not used.

If not all high-speed PHYs (like Ethernet SGMII, PCIe, or debug interface) which are available on a certain MCU product variant are used, then following constraints must be considered:

- If at least one high-speed PHY is used, then all $V_{DDPHPHYx}$ supply rails must be connected to 1.8 V
- All V_{DDPHYx} supply rails must be connected to the V_{DD} supply voltage, independently if the respective high-speed PHY is used or not, in order to enable testing of all safety logic in the SRI clusters

In the following table, all supply rails shall have been ramped up to their minimum voltage operational limits as documented in the datasheet before the external regulator releases the warm PORST of the MCU via the external PORST pin. It is not allowed to leave any supply rail unsupplied after the warm PORST reset release by the external regulator. If a supply rail is not supplied after the warm PORST release, the primary voltage monitors will keep the MCU in cold PORST state until all monitored supply rails are above their minimum operational limits ($V_{RSTxx}/V_{xxPRIUV}$). Also if a supply rail has a failure during operation, the primary voltage monitors will trigger a cold PORST and will keep the MCU in cold PORST as long as the supply rail is below the respective cold PORST threshold ($V_{RSTxx}/V_{xxPRIUV}$). In addition, at start-up, the PBIST1 and PBIST2 voltage tests are done, for over-voltage and under-voltage conditions, as described in the respective PBIST1/2 section, and if PBIST1 or PBIST2 fails, the MCU does not release the cold PORST. In the following table, multiple sub-columns mean several voltage levels are allowed, depending on the application requirements.

Table 1421 Allowed combinations of nominal supply voltages for the external voltage rails

Supply rails	V_{DDEXT} = 5 V nominal voltage level[2]		V_{DDEXT} = $V_{DDEVRSB}$ = 3.3 V nominal voltage level	
$V_{DDEVRSB}$	5 V	3.3 V	3.3 V	
$V_{DDEXTHS}$	5 V	3.3 V	3.3 V	
V_{DDEXT}	5 V		3.3 V	
$V_{DDP3NVM}$	N/A[1]		3.3 V	
$V_{DDP3RIF0}$	N/A[1]		3.3 V	
V_{DDM}	5 V only, no 3.3 V allowed		5 V[2]	3.3 V
V_{DDFLEX}	5 V	3.3 V	3.3 V	
$V_{DDEXTDC}$	3.3-5 V/5.5-6.5 V[3]		3.3-5 V/5.5-6.5 V[3]	
$V_{DDPHPHY0}$	1.8 V[6]		1.8 V[6]	
$V_{DDPHPHY1}$	1.8 V[6]		1.8 V[6]	
$V_{DDPHPHY2}$	1.8 V[6]		1.8 V[6]	
$V_{DDPHRIF0}$	N/A[1]		1.8 V[4]	
V_{DDHSIF}	1.8 V/3.3 V[5]		1.8 V/3.3 V[5]	
V_{DD}	1.025 V/0.975 V[6] (external supply or generated by EVRC)		1.025 V/0.975 V[6] (external supply or generated by EVRC)	
V_{DDPHY0}	1.025 V/0.975 V[6] (external supply or generated by EVRC)		1.025 V/0.975 V[6] (external supply or generated by EVRC)	
V_{DDPHY1}	1.025 V/0.975 V[6] (external supply or generated by EVRC)		1.025 V/0.975 V[6] (external supply or generated by EVRC)	
V_{DDPHY2}	1.025 V/0.975 V[6] (external supply or generated by EVRC)		1.025 V/0.975 V[6] (external supply or generated by EVRC)	
V_{DDRIF0}	N/A[1]		1.025 V/0.975 V (external supply or generated by EVRC)	
V_{DDSB}	1.025 V/0.975 V[6] (external supply or generated by EVRC)		1.025 V/0.975 V[6] (external supply or generated by EVRC)	

1) Supply rail not available on product variants with 5 V supply.
2) Except in certain product variants, where the voltage is limited to 3.3 V
3) Depending on the available external pre-regulator voltage. $V_{DDEXTDC}$ is limited to the datasheet parameter $V_{DDEXTDC}$ for maximum allowed voltage
4) Can be supplied from on-chip regulator.
5) V_{DDHSIF} voltage options:
   - 3.3 V for RGMII usage
   - 1.8 V or 3.3 V for xSPI usage, depending on application requirements
6) Voltage level dependent on product variant and sample type

Related information

TC4Dx power supply rails and distribution on page 6147

36.3.2 PMS infrastructure and domains

The PMS is comprised of 3 power domain partitions PMS0, PMS1, and PMS2, to support low power STANDBY0 and STANDBY1 modes.

The PMS0 domain constitutes the smallest standby domain with minimum functionality required to recognize wake-up from pins, supply and wake-up timer modules. The PMS0 domain contains the V_{DDEXT} and $V_{DDEVRSB}$ start-up detectors which release the LVD reset, the 70 kHz standby clock, the RTC with 32 kHz external crystal clock[42)] and regulators to keep standby RAM supplied[42)].

The PMS1 domain constitutes the next scalable standby domain including the 8-bit standby controller subsystem, the high precision band-gap for SCR ADC conversions and high precision 100 MHz clock which serves as SCR clock, back-up clock and start-up clock for the system. The external hardware configuration pins (*HWCFG[1-6]*) are latched in PMS to distinguish various boot modes .

The PMS2 domain includes the regulators and monitors. The EVRC synchronous step-down regulator controls the external complementary MOSFETs through VGATEx pins to buffer the energy in a LC filter in order to generate a regulated core supply. The primary under-voltage monitoring of the external and internal supplies are inherently carried out by multiple VMONPx monitors to ensure proper functioning of the system . Additional VMONSx secondary safety over-voltage and under-voltage monitoring against programmable thresholds is provided for all supplied and generated voltages . The standby SMU ensures the activation of the safe error pin path in case of critical safety alarms to inform external regulator or external safety monitors.

Related information

LVD reset on page 5967
Supply mode selection on page 5959
Step-down regulator (EVRC) on page 5979
Primary voltage monitors VMONPx and cold PORST on page 5971
Secondary over- and under-voltage monitors (VMONSx) on page 6003

36.3.2.1 Independent standby supply domains (PMS0, PMS1, PMS2)

The PMS standby domains include the following components. Refer to the functional description for the specific product variant for an overview figure of the components.

42 Except in certain product variants

Table 1422 PMS domains and included functions

Functional Block	Function/Component	Description
PMS	PMS0, PMS1, and PMS2	Standby power domain partitions. PMS0.VR, PMS1.VR and PMS0.SBRAMVR are the internal regulators to supply these partitions. PMS2 Switch isolates the PMS2 domain from the PMS1 domain
	PMS0.PORDET	Low-voltage supply detectors (for power-on reset release)
	PMS0.PSSM0 PMS1.PSSM1	Power start-up state machines
	PMS0.LPBG	Low-power band-gap voltage reference
	PMS0.LPOSC	70 kHz standby clock source (fSB)
	PMS0.WKUP	Wake-up unit to wake-up on pin events or V_{DDEXT} supply-ramp-up
	PMS0.WUT	Wake-up timer
	PMS0.RTC	Real time clock and 32 kHz XTAL support
	PMS1.HPOSC	100 MHz infrastructure clock (fBACK)
	PMS1.HPBG	Secondary high-precision band-gap voltage reference
	PMS0 VR SBRAM VR PMS1 VR PRE VRx PMS2 Switch	Internal voltage regulators for the PMS internal supply rails, respectively power switch for the V_{DDPMS2} rail
	PMS2.EVRC	Embedded core voltage regulator
	PMS2.VMONPx	Primary voltage monitors and cold PORST reset
	PMS2.VMONSx	Secondary voltage monitors and safety alarms
SCR	SCR modules	8-bit standby controller sub-system integrated in the PMS1 domain
DTS	DTSPMS	Die temperature module integrated in the PMS1 domain
SMU	SMU_STDBY	Standby SMU integrated in the PMS2 domain

The objective of the integrated voltage regulators (PMS0 VR, SBRAM VR, PRE VRx, PMS1 VR) is to supply the basic infrastructure components, the standby domain, and certain safety components with a dedicated low-noise supply independent from the rest of the chip. The V_{DDPMSx} regulators are supplied by the external 5 V or 3.3 V $V_{DDEVRSB}$ supply. They are implemented as low drop-out regulators, generating the V_{DDPMSx} internal voltage, which is buffered internally and is not routed to any external supply pin. Since $V_{DDPMS0-1}$ domains are powered in STANDBY1 mode, they are implemented to have low power consumption to meet the $I_{STANDBY}$ current parameter limits in the data sheet. The V_{DDPMSx} supplies the high precision band-gap, the 100 MHz infrastructure clock source, and the embedded EVRC regulator, as voltage regulators have to be supplied independently from their generated voltages. The V_{DDPMSx} regulators also supply the wake-up timer, the standby controller, and a part of the port domain (Port 32-35). During standby mode, the standby RAMs are supplied by the $V_{DDSBRAM}$ (SBRAM VR) regulator.

Related information

TC4Dx independent standby supply domains (PMS0, PMS1, PMS2) on page 6148

36.3.2.2 Reference voltage generation (HPBG)

The objective of the secondary[43] high precision band-gap (HPBG) and reference current circuitry is to provide an accurate voltage reference and reference currents to various modules. The reference is used by EVRC and supply monitors.

The secondary high precision band-gap reference is checked against the primary low-power band-gap reference or V_{DDPMSx} voltage to detect band-gap drifts during the start-up phase. This is part of the power BIST which is carried out only during a supply ramp-up. Furthermore, the HPBG voltage can be measured on an ADC channel configured through GLOBCFG.SVCH and GLOBCFG.SVSIG bit-fields. This provides a diagnostic between the internal band-gap reference and the external ADC V_{AREFx} reference voltage.

A dynamic temperature-dependent trimming of the high-precision band-gap is available and can be enabled in the HPBGTRIM register. This feature is enabled using the HPBGTRIM.TRIMEN bit and it applies a selected trim value, as documented in the HPBGTRIM.TRIMx bit-fields, depending on the temperature measured by the PMS die temperature sensor (DTSPMS). Using this trimming feature requires the DTSPMS to be enabled and started (through the DTS_CON register). During LIN communication, the application software must ensure that the trim value is not changed until the end of the communication schedule, therefore it must set the SCR bit LIN_BCON.TTDIS to 1_B at the beginning of the LIN communication schedule and clear it to 0_B at the end of the communication schedule. The PMS hardware ensures that a temperature-dependent change of the trim value is only allowed when the SCR bit LIN_BCON.TTDIS is equal to 0_B.

The application software must be aware that the HPBGTRIM register is reset whenever entering STANDBY0 mode (to the value indicated by the LVDPMS1 reset type). Therefore, if an application switches from STANDBY0 mode directly to STANDBY1 mode (for instance, if the application enters STANDBY0, then wakes up to STANDBY1) and if the application requires the dynamic trimming feature for increased HPBG accuracy in STANDBY1 mode, it needs to store the content of this register into the SCR standby XRAM before going into STBANDBY0. After waking up from STANDBY0 directly into STANDBY1, the SCR software can read the stored values from SCR XRAM and perform the following steps in order to emulate the hardware behavior of the temperature-dependent trimming:

- Select the temperature-dependent TRIMx value depending on the DTS_STAT.RESULT temperature reading
- Add the TRIMx value (considering sign interpretation) to the content of the SCR SCU_PMSHPBG_BGTRIM register

Related information

Power built-in self-test (PBIST1/2) on page 6008

36.3.2.3 70 kHz standby clock (f_{SB}) and 100 MHz infrastructure clock (f_{BACKT})

The 70 kHz clock source is used during standby mode. Its accuracy is specified by the f_{SB} parameter in the datasheet. This clock source is trimmed after start-up.

The 100 MHz clock source is a precise on-chip clock used by the PMS infrastructure including on-chip voltage regulators, state machines, and the Standby Controller (SCR). After start-up, the 100 MHz clock source has a lower accuracy (see f_{BACKUT} parameter in the datasheet) and the clock source is later trimmed by the start-up software (see f_{BACKT} parameter in the data sheet).

A fine trimming is option is available in the register field HPOSCCON1.OSCFTRIM. The fine trim value is reset on a cold PORST reset, to be able to revert to the default setting in case of erroneous trimming. The frequency trim steps are non-linear and dependent on process, temperature, and voltage.

43 There are two band-gaps implemented in the PMS, a primary low-power band-gap, and a secondary high-precision band-gap.

A dynamic temperature-dependent trimming of the 100 MHz clock is available and can be enabled in the HPOSCTRIM register. This feature is enabled using the HPOSCTRIM.TRIMEN bit and it applies a selected trim value, as documented in the HPOSCTRIM.TRIMx bit-fields, depending on the temperature measured by the PMS die temperature sensor (DTSPMS). Using this trimming feature requires the DTSPMS to be enabled and started (through the DTS_CON register). During LIN communication, the application software must ensure that the trim value is not changed until the end of the communication schedule, therefore it must set the SCR bit LIN_BCON.TTDIS to 1_B at the beginning of the LIN communication schedule and clear it to 0_B at the end of the communication schedule. The PMS hardware ensures that a temperature-dependent change of the trim value is only allowed when the SCR bit LIN_BCON.TTDIS is equal to 0_B.

The application software must be aware that HPOSCTRIM and HPOSCCON1 registers are reset whenever entering STANDBY0 mode (to the value indicated by the LVDPMS1 reset type). Therefore, if an application switches from STANDBY0 mode directly to STANDBY1 mode (for instance, if the application enters STANDBY0, then wakes up to STANDBY1) and if the application requires the dynamic trimming feature for increased accuracy in STANDBY1 mode (for instance for LIN communication), it needs to store the content of these registers into the SCR standby XRAM before going into STBANDBY0. After waking up from STANDBY0 directly into STANDBY1, the SCR software can read the stored values from SCR XRAM and perform the following steps in order to emulate the hardware behavior of the temperature-dependent trimming:

- Select the temperature-dependent TRIMx value depending on the DTS_STAT.RESULT temperature reading
- Read the previously stored HPOSCCON1.OSCFTRIM value
- Add the OSCFTRIM and the TRIMx values (considering sign interpretation) and write the result into the SCR registers SCU_PMSHPOSC_FTRIM7_0 and SCU_PMSHPOSC_FTRIM8_8

36.3.3 Supply start-up and POR release

This section provides details on the PMS functionality related to the supply selection, ramp-up and ramp-down behavior, and power-on reset (POR).

36.3.3.1 Supply mode selection

Refer to the functional description for the specific product variant for an image showing the hardware configuration (HWCFG) pins.

The choice of the supply scheme at start-up is based on the latched status of *HWCFG[2:1]* pins before cold PORST release and is indicated by the PAD_STAT.HWCFGx status flags. Following supply modes are available depending on product variant and are further enumerated in the following table:

- Single source 5 V supply level ($V_{DDEXTDC} = V_{DDEXT}$ = 5 V, or V_{DDEXT} = 5 V and 3.3 V ≤ $V_{DDEXTDC}$ ≤ 5 V) with EVRC in SMPS mode using external switches. This 5 V supply mode is not available on certain product variants
- Single source 3.3 V supply level ($V_{DDEXTDC} = V_{DDEXT}$ = 3.3 V, or V_{DDEXT} = 3.3 V and 3.3 V ≤ $V_{DDEXTDC}$ ≤ 5 V) with EVRC in SMPS mode using external switches. On certain product variantsonly $V_{DDEXTDC}$ = 3.3 V is supported
- High-voltage supply level (5.5 V[44] ≤ $V_{DDEXTDC}$ *(Typ)* ≤ 6.5 V, not available on certain product variants) with EVRC in SMPS mode using external switches:
  - V_{DDEXT} = 5 V
  - V_{DDEXT} = 3.3 V
- V_{DD} supply provided externally and EVRC is in disabled state:
  - $V_{DDEXTDC} = V_{DDEXT}$ = 5 V and V_{DD} supplied externally. Not available on certain product variants
  - $V_{DDEXTDC} = V_{DDEXT}$ = 3.3 V and V_{DD} supplied externally

[44] Minimum static voltage value for the high-voltage mode. Dynamic undershoots down to 5 V are supported in this mode.

The above-mentioned supply voltage conditions refer to the nominal values of the supply voltages (static conditions). In addition, a dynamic operational range for each supply voltage, as defined in the supply operating conditions in the data sheet, is supported.

EVRC is enabled or disabled at start-up through the *HWCFG[2:1]* configuration pins. In case EVRC is enabled, *VGATE1P* and *VGATE1N* pins must be connected to the gate of an external P-channel MOSFET and N-channel MOSFET respectively. In case EVRC is disabled, it is recommended to tie $V_{DDEXTDC}$ to V_{DDEXT}.

On certain product variants , *HWCFG[2]* is used to configure the high-end (HE) or low-end (LE) configuration of the EVRC, if enabled, with respect to the available switching frequencies:

- *HWCFG[2]*=0 selects the EVRC HE configuration, with supported f_{DCDC} frequencies of either 0.8 MHz or 0.45 MHz
- *HWCFG[2]*=1 selects the EVRC LE configuration, with supported f_{DCDC} frequencies of either 1.82 MHz or 0.8 MHz

The allowed combinations of nominal external supply voltages among different supply rails are documented in chapter Power supply rails and distribution . All externally provided supplies must be available and be stable before warm PORST reset release by the external regulator(s) (the external regulator releases a warm PORST of the MCU via the external PORST pin). If an externally provided supply rail is not available after the warm PORST release, the primary voltage monitors will keep the MCU in cold PORST state until all monitored supply rails are above their minimum operational limits (V_{RSTxx}/$V_{xxPRIUV}$). In addition, at start-up, the PBIST1 and PBIST2 voltage tests are done, for over-voltage and under-voltage conditions, as described in the respective PBIST1/2 section, and if PBIST1 or PBIST2 fails, the MCU does not release the cold PORST.

HWCFG[2:1] are latched during supply ramp-up and the EVRC regulator, if enabled, is consequently started. The latched values are stored in the PAD_STAT.HWCFGx register bits. The latched values are retained through a cold PORST and are only reset if LVD (Low Voltage Detector) reset (LVDANA reset) is asserted. HWCFG signals are filtered through a spike/glitch filter and are monitored for a constant level over a 28 µs - 115 µs nominal debouncing period before the value is considered as valid so as to ensure reliable operation in noisy environment. The current state of EVRC is reflected in the EVRC_STAT0.EVRC flag. For small package variants, some of the *HWCFG* configuration pins may be absent and the EVRC is activated by default at start-up.

HWCFG[6] pin state is propagated to pads in the V_{DDEXT} and $V_{DDEVRSB}$ domain during early V_{DDEXT} supply ramp-up ($V_{DDEXT} \geq V_{DDPPA}$, data sheet parameter) to decide and set the default reset state of port pins as early as possible. During the initial ramp-up phase of V_{DDEXT} and $V_{DDEVRSB}$ supply voltage from 0 V up to V_{DDPPA} limit, the voltage levels of pins are undefined till the transistor threshold voltages are reached. After V_{DDPPA} limit, the pins behave as inputs with pull-up if *HWCFG[6]* = 1 or are in tristate if *HWCFG[6]* = 0. During the later stage of ramp-up, after $V_{DDEXT} \geq V_{LVDRSTEXT}$ (data sheet parameter) and after LVDANA reset release, the *HWCFG[6]* inverted value is latched and stored in the PAD_STAT.ALLTRIST and PAD_STAT.SBTRIST register bits.

HWCFG[1,2,3,6] pins have weak internal pull-up active at start-up, irrespective of *HWCFG[6]* pin level, to ensure that the device boots with a defined configuration if *HWCFG[1,2,3,6]* pins are left unconnected. *HWCFG[1,2,3,6]* pins are only latched by the PMS on every initial supply ramp-up and are not re-latched during warm reset events (warm PORST, system or application resets) or upon exit from standby mode (all standby modes). All *HWCFG* pins are additionally latched by the SMM functional block in the STSTAT register (start-up status register) on every warm reset release, see the documentation of the SMM functional block.

- *HWCFG[6]* and *HWCFG[1,2]* are recognized as high when the respective pin is open or pulled up to V_{DDEXT} supply with pull device on the external system, according to the R_{HWCFG} value indicated in the data sheet
- *HWCFG[6]* and *HWCFG[1,2]* are recognized as low when the respective pin is pulled down to *GND* with pull device on the external system, according to the R_{HWCFG} value indicated in the data sheet

Table 1423 Pad state depending on $V_{DDEVRSB}$ and V_{DDEXT} supply

$V_{DDEVRSB}$ level	V_{DDEXT} level	$V_{DDEVRSB}$ pads state	V_{DDEXT} pads state
$V_{DDEVRSB} > V_{LVDRSTSB}$	$V_{DDEXT} < V_{DDPPA}$	Undefined (pull-up not guaranteed)	Undefined
$V_{DDEVRSB} > V_{LVDRSTSB}$	$V_{DDPPA} \le V_{DDEXT} < V_{LVDRSTEXT}$	As defined by *HWCFG[6]*	As defined by *HWCFG[6]*
$V_{DDEVRSB} > V_{LVDRSTSB}$	$V_{DDEXT} > V_{LVDRSTEXT}$	As defined by PAD_CON.SBTRIST	As defined by PAD_CON.ALLTRIST
$V_{DDEVRSB} < V_{LVDRSTSB}$	$V_{DDEXT} < V_{DDPPA}$	Undefined	Undefined
$V_{DDPPSB} \le V_{DDEVRSB} < V_{LVDRSTSB}$	$V_{DDPPA} \le V_{DDEXT} < V_{LVDRSTEXT}$	As defined by *HWCFG[6]*	As defined by *HWCFG[6]*

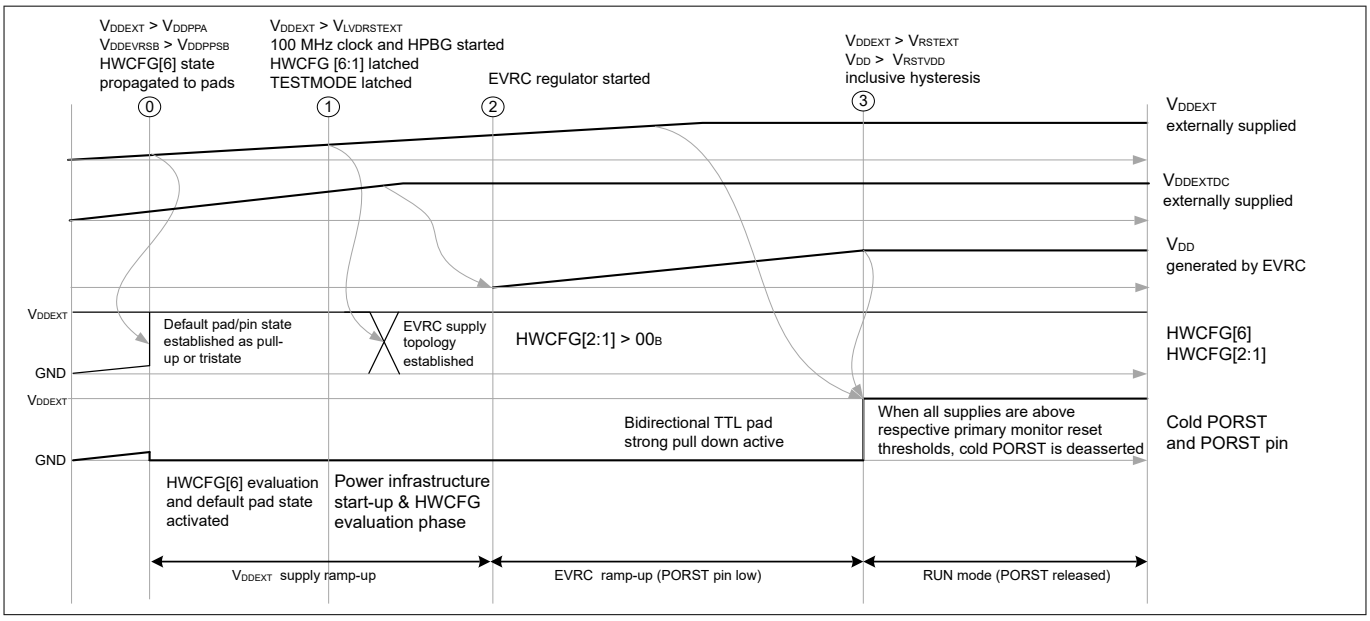


Figure 775 *HWCFG* latching

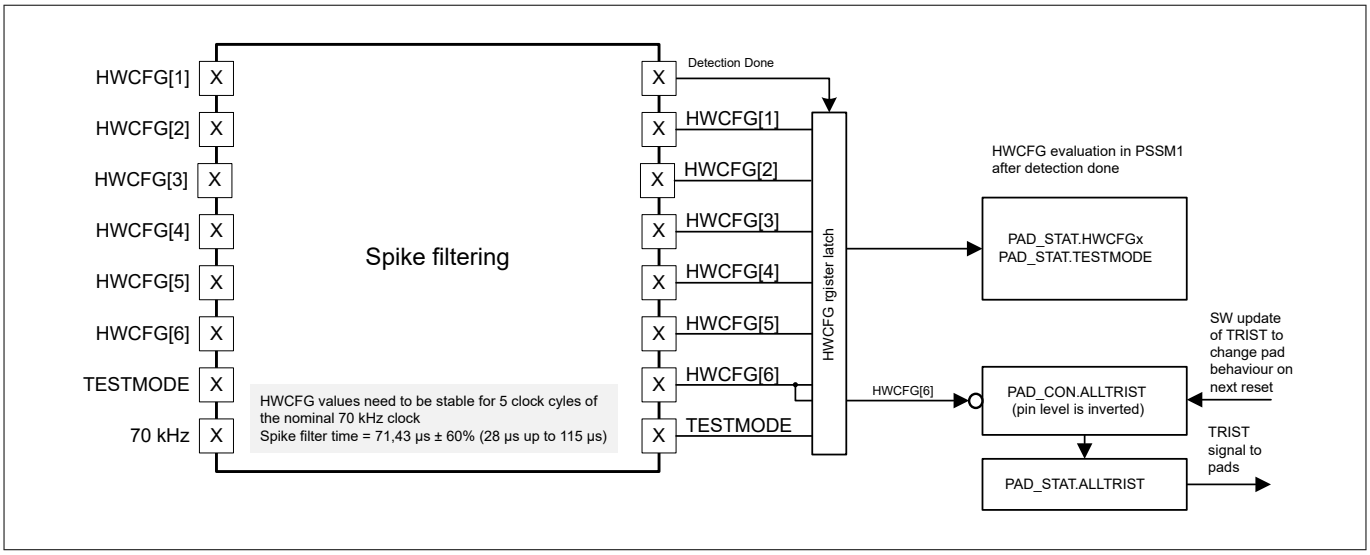


Figure 776 *HWCFG* filtering and evaluation

Table 1424 Supply mode and topology selection

Power mode	*HWCFG[2:1]*	*VGATE1P*/*VGATE1N*	Supply pin voltage level/source	Selected supply scheme
h1.)	00_B on variants where HWCFG[2:1] is used for V_{DDEXT} & $V_{DDEXTDC}$ range selection.	*VGATE1P* internal pull-up to $V_{DDEXTDC}$, externally left open. *VGATE1N* internal pull-down to $V_{SSEXTDC}$, externally left open.	V_{DD} supplied externally. $V_{DDEXTDC}$ & V_{DDEXT} & $V_{DDEVRSB}$ = 5 V	EVRC inactive, V_{DD} external supply. Standby mode is supported and V_{DD} supply shall be switched off by external regulator after standby state is entered.
	$X0_B$ on variants where only HWCFG[2:1] is used for the EVRC enable/ disable & switching frequency selection.		not supported	
h2.)	00_B on variants where HWCFG[2:1] is used for V_{DDEXT} & $V_{DDEXTDC}$ range selection.	*VGATE1P* internal pull-up to $V_{DDEXTDC}$, externally left open. *VGATE1N* internal pull-down to $V_{SSEXTDC}$, externally left open.	V_{DD} supplied externally. $V_{DDEXTDC}$ & V_{DDEXT} & $V_{DDEVRSB}$ = 3.3 V	EVRC inactive, V_{DD} external supply. 3.3 V external supply. Standby mode is supported and V_{DD} supply shall be switched off by external regulator after standby state is entered.
	$X0_B$ on variants where only HWCFG[2:1] is used for the EVRC enable/ disable & switching frequency selection.		V_{DD} supplied externally. $V_{DDEXTDC}$ & V_{DDEXT} & $V_{DDEVRSB}$ = 3.3 V	
e.)	01_B on variants where HWCFG[2:1] is used for V_{DDEXT} & $V_{DDEXTDC}$ range selection.	*VGATE1P*/*VGATE1N* connected to gate of P-/N-ch. MOSFET	V_{DDEXT} & $V_{DDEVRSB}$ = 3.3 V 3.3 V ≤ $V_{DDEXTDC}$ ≤ 5 V V_{DD} supplied by EVRC.	3.3 V single source supply. EVRC in SMPS mode. Standby mode supported.

(table continues...)

Table 1424 (continued) Supply mode and topology selection

Power mode	*HWCFG[2:1]*	*VGATE1P/VGATE1N*	Supply pin voltage level/source	Selected supply scheme
	$X1_B$ on variants where only HWCFG[2:1] is used for the EVRC enable/ disable & switching frequency selection.		$V_{DDEXTDC}$ = 3.3 V V_{DD} supplied by EVRC.	
a.)	10_B on variants where HWCFG[2:1] is used for V_{DDEXT} & $V_{DDEXTDC}$ range selection.	*VGATE1P/VGATE1N* connected to gate of P-/N-ch. MOSFET	V_{DDEXT} & $V_{DDEVRSB}$ = 5 V 3.3 V ≤ $V_{DDEXTDC}$ ≤ 5 V V_{DD} supplied by EVRC.	5 V single source supply. EVRC in SMPS mode. Standby mode supported.
	Not supported on variants where only HWCFG[2:1] is used for the EVRC enable/ disable & switching frequency selection.		Not supported.	
b1.)	11_B on variants where HWCFG[2:1] is used for V_{DDEXT} & $V_{DDEXTDC}$ range selection.	*VGATE1P/VGATE1N* connected to gate of P-/N-ch. MOSFET	5.5 V[44] ≤ $V_{DDEXTDC}$ ≤ 6.5 V V_{DDEXT} & $V_{DDEVRSB}$ = 5 V V_{DD} supplied by EVRC.	High-voltage $V_{DDEXTDC}$ mode. 5 V external supply. EVRC in SMPS mode. Standby mode supported (with higher standby current, see I_{DCSTBY} datasheet parameter).

(table continues...)

[44] Minimum static voltage value for the high-voltage mode. Dynamic undershoots down to 5 V are supported in this mode.

Table 1424 (continued) Supply mode and topology selection

Power mode	*HWCFG[2:1]*	*VGATE1P*/*VGATE1N*	Supply pin voltage level/source	Selected supply scheme
	Not supported on variants where only HWCFG[2:1] is used for the EVRC enable/ disable & switching frequency selection.		Not supported.	
b2.)	11_B on variants where HWCFG[2:1] is used for V_{DDEXT} & $V_{DDEXTDC}$ range selection.	*VGATE1P*/*VGATE1N* connected to gate of P-/N-ch. MOSFET	5.5 V[44] ≤ $V_{DDEXTDC}$ ≤ 6.5 V V_{DDEXT} & $V_{DDEVRSB}$ = 3.3 V V_{DD} supplied by EVRC.	High-voltage $V_{DDEXTDC}$ mode. 3.3 V external supply. EVRC in SMPS mode. Standby mode supported (with higher standby current, see I_{DCSTBY} datasheet parameter).
	Not supported on variants where only HWCFG[2:1] is used for the EVRC enable/ disable & switching frequency selection.		Not supported.	

If *HWCFG[2,1]* pins are left unconnected, it is ensured that EVRC is active, owing to the internal weak pull-up active by default after start-up/cold PORST. Only the nominal voltage values of the respective rails are indicated in the table. The tolerances of the supply voltages are documented in the datasheet.

For *HWCFG[2:1]*=00_B, all three supply rails $V_{DDEXTDC}$, V_{DDEXT}, and $V_{DDEVRSB}$ must be on the same level, that is $V_{DDEXTDC}$ = V_{DDEXT} = $V_{DDEVRSB}$ = either 5 V or 3.3 V (on certain product variants only 3.3 V is supported). Also when *HWCFG[2:1]*= 01_B, or 10_B, or 11_B, V_{DDEXT} and $V_{DDEVRSB}$ must be on the same level, that is V_{DDEXT} = $V_{DDEVRSB}$ = either 5 V or 3.3 V (on certain product variants only 3.3 V is supported).

Related information

Step-down regulator (EVRC) on page 5979
Power supply rails and distribution on page 5948
TC4Dx supply mode selection on page 6149

36.3.3.1.1 Using HWCFG pins as GPIOs

If *HWCFG[5:1]* are not exclusively used as configuration pins and are left unconnected or used as GPIOs after startup then certain conditions are to be carefully considered

44 Minimum static voltage value for the high-voltage mode. Dynamic undershoots down to 5 V are supported in this mode.

If a deterministic behavior for the pins during reset execution is wanted the *HWCFG[6]* pin needs to be tied to 1_B to ensure pull-up behavior for all GPIO pins, including HWCFG pins, during and after reset execution.

If *HWCFG[5:1]* pins are used as output pins then the following conditions are to be considered

- If *HWCFG[6]* is tied to 1_B or left open then *HWCFG[6:1]* will be configured as weak pull-up. This weak pull-up condition of any particular pin can be overridden by connecting appropriate voltage levels from outside
- If *HWCFG[6]* is tied to 0_B then *HWCFG[5:1]* are in tristate. During the reset execution *HWCFG[5:1]* pins are needed to be driven to appropriate voltage

If *HWCFG[5:1]* are used as input pins then the following condition is to be considered

- If *HWCFG[5:1]* are used as inputs then during the during the boot up phase it to ensured those pins are connected to expected stable voltage levels

If *HWCFG[5:1]* are unconnected then depending on which *HWCFG* pin is unconnected different behaviors are expected

- If *HWCFG[6]* is unconnected then before the cold PORST release it would be sampled as 1_B. As a result of it the internal pull-up of all the GPIOs including *HWCFG* pins will be enabled unless the Pad configuration register.ALLTRIST bit field is set to 1_B. If Pad configuration register.ALLTRIST bit field is set to 1_B during runtime then after application reset no deterministic behavior can be ensured for the unconnected *HWCFG* pins. It is recommended to avoid this situation
- if HWCFG[2:1] are unconnected then for proper operation of EVRC, it has to be ensured that $V_{DDEXTDC}$ is within the range specified by the datasheet parameter $V_{INEXTDC}$ for HWCFG[2:1]=11_B
- If *HWCFG[5:3]* pins are unconnected then
  - If *HWCFG[6]* is sampled as 1_B and Pad configuration register.ALLTRIST bit-field is set to 0_B during runtime then the *HWCFG[3]* pin would be pulled up which will ensure start from BMHD
  - If If *HWCFG[6]* is tied to 0_B and Pad configuration register.ALLTRIST bit-field is set to 1_B during runtime then *HWCFG[5:3]* pins are in tristate. After the application reset no deterministic behavior for *HWCFG[5:3]* pins can be expected which can lead to unwanted boot behavior. In this scenario at least *HWCFG[3]* pin must be connected to 1_B to ensure the start from BMHD BMI
  - If If *HWCFG[6]* is tied to 0_B and Pad configuration register.ALLTRIST bit-field is set to 1_B during runtime then *HWCFG[5:3]* pins are in tristate. After the application reset no deterministic behavior for *HWCFG[5:3]* pins can be expected which can lead to unwanted boot behavior. In addition if *HWCFG[3]* is connected to 0_B then *HWCFG[5:4]* must be defined to valid input (0_B or 1_B)
  - If UCB1_15_BML.BML bit-field is activated then the HWCFG[5:3] states are ignored

36.3.3.2 Supply ramp-up and ramp-down behavior

This section presents the behavior of the supply rails during ramp-up, ramp-down, and dynamic operational phases.

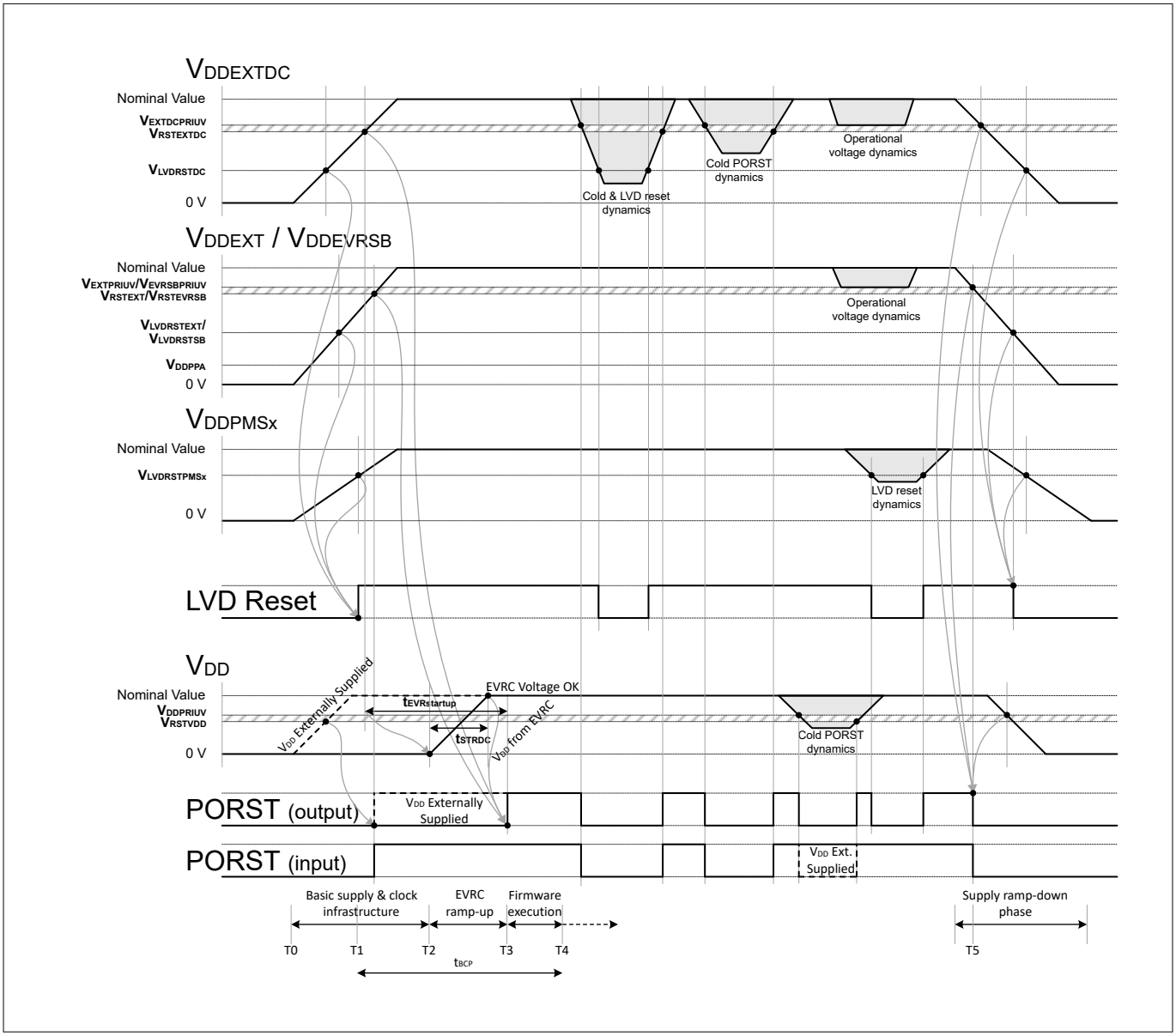


Figure 777 Supply ramp-up and ramp-down behavior

- The start-up slew rates for the supply rails shall comply to the respective data sheet parameters dV_{XX}/dt. The slope is defined as the maximum tangential slope between 0% to 100% voltage level. The actual waveform may not represent the specification
- In case of an external regulator for the V_{DD} supply rail, it is ensured that the load jumps to the external regulator (from the V_{DD} rail) are limited during the start-up phase to the dI_{DDDYN_STRT} data sheet parameter
- PORST is active or asserted when either PORST (input) or PORST (output) is active or asserted
- PORST (input) active means that the reset is held active by external agents by pulling the PORST pin low. It is recommended to keep the PORST (input) asserted until the external supply is above the respective primary reset threshold (V_{RSTxx} parameter in the datasheet)
- PORST (output) active means that the microcontroller asserts the reset internally and drives the PORST pin low, propagating the reset to external devices. The PORST (output) is asserted by the MCU when at least one supply domain violates its primary under-voltage reset threshold. The PORST (output) is de-asserted

by the microcontroller when all relevant supplies (with VMONP_VxxRST.RESETOFF = 0) are above their primary reset thresholds and the basic supply and clock infrastructure is available

The power sequence as shown in the figure above is enumerated as follows:

- T0 is the begin of the supply ramp-up from the external regulator
- T1 is the point in time when LVD reset is released. The LVD resets are released when $V_{DDEXTDC}$, V_{DDEXT}, $V_{DDEVRSB}$, and V_{DDPMSx} are above their LVD reset levels (denoted as $V_{LVDRSTxx}$)
- T0 up to T2 refers to the period in time when basic supply and clock infrastructure components are made available as the external supply ramps up. The band-gap and internal clock sources are started. The supply mode is evaluated based on the *HWCFG* pins. These events are initiated after LVD reset release at T1
- T2 refers to the point in time where a soft start of the EVRC is initiated. PORST (input) does not have any effect on the EVRC output and the regulator continues to generate the respective voltage though PORST is asserted and the device is in reset state. The generated voltage follows a soft ramp-up over the t_{STRDC} time (data sheet parameter) to avoid overshoots
- T3 refers to the point in time when all supplies are above their primary reset thresholds, denoted as V_{RSTxx}. There are two parameters denoting the primary reset threshold, respectively V_{RSTxx} (which is the untrimmed threshold) and $V_{xxPRIUV}$ (which is the trimmed threshold). Their values are located closely to each other and are indicated therefore as a thin hashed region in the figure above. During the supply ramp-up phase, the untrimmed threshold V_{RSTxx} is used for the first PORST release. After the firmware execution phase, the trimmed threshold $V_{xxPRIUV}$ is used for the PORST assertions. Subsequent reset releases occur at the untrimmed threshold V_{RSTxx}, since the primary voltage monitors reset thresholds VMONP_VxxRST.RESETTRIM are being reset to the V_{RSTxx} values also after a cold PORST (see the "Cold PORST" reset values of the VMONP_VxxRST registers)
- The supply start-up phase is completed when the regulator outputs are stable and operational and all voltages have ramped up. Consequently cold PORST reset is released when all voltages on the primary monitors are above their minimum operational limits (V_{RSTxx}/$V_{xxPRIUV}$)
- T4 refers to the point in time when the firmware execution is completed and the user code execution starts, with CPU0 running at the f_{BACKT} clock frequency. The time between T1 and T4 is documented as t_{BCP} (cold power on reset boot time, datasheet parameter)
- T5 refers to the point in time during the supply ramp-down phase when at least one of the supplies (either externally provided or internally generated) drops below its respective primary under-voltage reset threshold $V_{xxPRIUV}$

36.3.3.3 LVD reset

This section introduces the power-on reset detectors for minimum voltage detection (PMS0.PORDET, also called LVD detectors) and the low voltage detection (LVD) reset concepts.

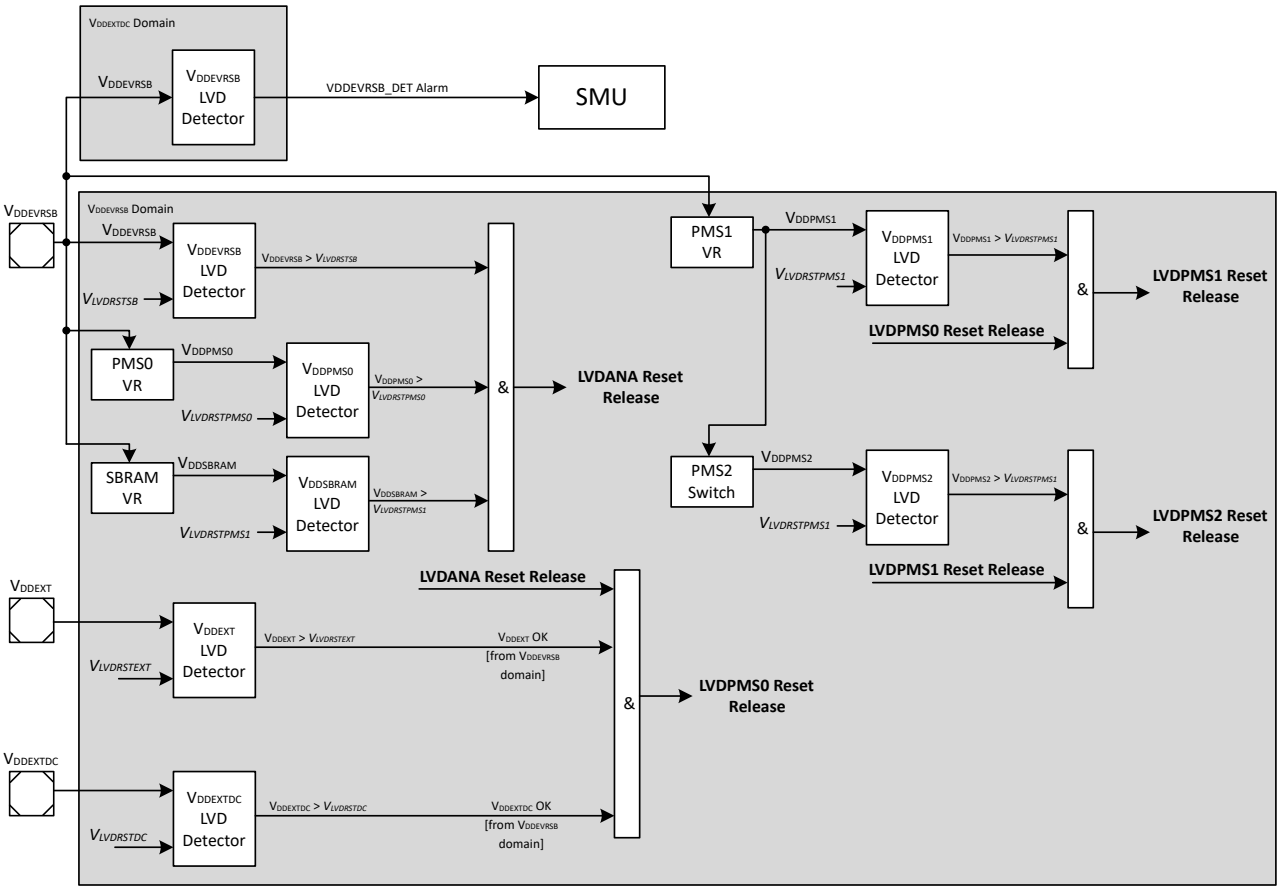
The power-on reset detectors constitute a minimum voltage detection logic, which ensures that a minimum voltage level is available on the V_{DDEXT}, $V_{DDEVRSB}$, $V_{DDEXTDC}$ external supplies, and on the internally generated V_{DDPMSx} supplies. The V_{DDEXT} supply is monitored and compared to the minimum level defined by the $V_{LVDRSTEXT}$ parameter, to ensure that adequate voltage is available to latch the *HWCFG* pins. $V_{DDEVRSB}$ is monitored for a minimum voltage level of $V_{LVDRSTSB}$. Likewise, the internal V_{DDPMSx} supplies are monitored for minimum $V_{LVDRSTPMSx}$ voltage level by detectors with built-in reference (low-power band-gap).

Power-on detectors are available for the $V_{DDEVRSB}$, V_{DDEXT}, $V_{DDEXTDC}$, V_{DDPMS0}, V_{DDPMS1}, V_{DDPMS2}, and $V_{DDSBRAM}$ (if available on the respective product variant), to ensure a proper minimum power detection, robust start-up, and standby operation. Under-voltage of the respective supplies will lead to the assertion of the LVD (Low Voltage Detector) reset. Assertion of LVD reset is reflected in the CLDRSTSTAT register (4 LVDx status bits for the 4 different LVD reset types) and can be evaluated by the software at the next start-up.

There are four types of LVD reset, as shown in the following figure: LVDANA, LVDPMS0, LVDPMS1, and LVDPMS2. The reset levels are hierarchical in nature (LVDANA < LVDPMS0 < LVDPMS1 < LVDPMS2). If a lower-level reset is asserted, also the higher-level resets are asserted as well. The following figure shows the LVD reset release.

Figure 778 LVD reset types

36 Power Management System (PMS)



LVDANA reset is released when $V_{DDEVRSB}$, V_{DDPMS0}, and $V_{DDSBRAM}$ are above their respective minimum voltage levels ($V_{LVDRSTSB}$, $V_{LVDRSTPMS0}$, and $V_{LVDRSTPMS1}$ respectively). This reset affects several analog modules within the PMS. The LVDPMS0 reset affects the PMS0 domain and is released after the LVDANA reset, when, in addition, V_{DDEXT}, and $V_{DDEXTDC}$ are above their minimum voltage levels ($V_{LVDRSTEXT}$ and $V_{LVDRSTDC}$ respectively). The LVDPMS1 reset affects the PMS1 domain and is released after the LVDPMS0 reset, when, in addition, V_{DDPMS1} is above its minimum voltage level defined by the $V_{LVDPMS1}$ parameter. The LVDPMS2 reset affects the PMS2 domain and is released after the LVDPMS1 reset, when, in addition, V_{DDPMS2} is above its minimum voltage level defined by the same parameter $V_{LVDPMS1}$.

After the release of the LVDPMS1 reset, the 100 MHz clock and high precision band-gap are consequently started. The *HWCFG* pins are evaluated (after the release of LVDPMS0 reset) to establish the supply mode which needs to be activated. Further, after LVDPMS2 reset release, if the EVRC is enabled by the *HWCFG* pins, it is started in a soft ramp-up to ensure a V_{DD} voltage ramp-up with minimal overshoots.

For all PMS0 registers (which are reset by the LVDANA or by the LVDPMS0 reset), indicated by an "LVDANA Reset" or an "LVDPMS0 Reset" type in the reset table of the respective register, it is recommended to:

- Wait before reading from the register until REGACCSTAT.FIFOEMPTY = 1
- Wait before writing to the register until REGACCSTAT.FIFOFULL = 0

These checks are recommended since the full status of the FIFO in the register access interface could be reached after writing 32 times to LVDANA or LVDPMS0 registers within 14 µs. In order to check for the status of the FIFOEMPTY and FIFOFULL bits, there are corresponding interrupts which can be enabled in the INT_CON2 register.

36.3.3.4 PORST pin

The PORST pin is the main reset input pin and triggers a warm power on reset of the device. It shall be connected to the external regulator or supply monitor. If this pin is left open then the default behavior is that the device remains in a reset state.

The PORST pin supports following functions:

- Warm PORST reset input function: It is expected that external monitor or regulator asserts PORST pin if operating conditions are violated, namely supply voltages. It is also expected that the external monitor asserts PORST pin if microcontroller is stuck and not responding or the external watchdog was not serviced by the microcontroller.
- If PORST pin is externally asserted, a warm PORST reset is triggered. It is ensured that all warm resets are sequenced internally and RAM contents are not affected. Reset is executed only after RAM write operations are completed and processing units have reached IDLE state.
- If the PORST pin is kept asserted externally after a system start-up and supply voltages are within operating range, then the warm PORST reset state will be kept extended until PORST pin is de-asserted.
- The PORST pin is driven low by PMS on a cold PORST reset trigger by activating a strong pull-down capable of sinking I_{PORST} current as documented in the datasheet. The strong pull-down is deactivated after system is back to RUN0 mode. During initial Boot strong pull-down is only released on reaching RUN0 (Operation) mode. During Standby state and during transition from Standby state to RUN0 (Operation) mode, strong pull-down is not activated.
- The PORST pin has an integrated weak pull-down function to ensure that device stays in reset state if pin connection or PCB routing to the PORST pin is broken. The weak pull-down is active in all system states except Standby mode. In Standby mode, the weak pull-down is deactivated to reduce standby current. The weak pull-down is activated by default during Supply Start-up phase and during Standby-exit.
- Multiple masters can trigger PORST if connected together in an open-drain circuit with an external pull-up. Depending on push-pull or open drain implementation of reset output by external regulator, care should be taken in the dimensioning of the pull-up and series resistances on PORST connection.
- The PORST pin has a basic analog spike filter to filter out noise as documented in the datasheet. It supports TTL input level for 5 V operation and degraded TTL levels for 3.3 V operations as documented in the datasheet.
- In addition to the inherent analog spike filter, PORST pin supports a digital filter that can be additionally activated via PMS.PAD_CON.PORSTDFEN register bitfield to filter higher noise at input.

36.3.3.5 External service request interface (ESRx)

The external service request ESRx pin functions are described in this section.

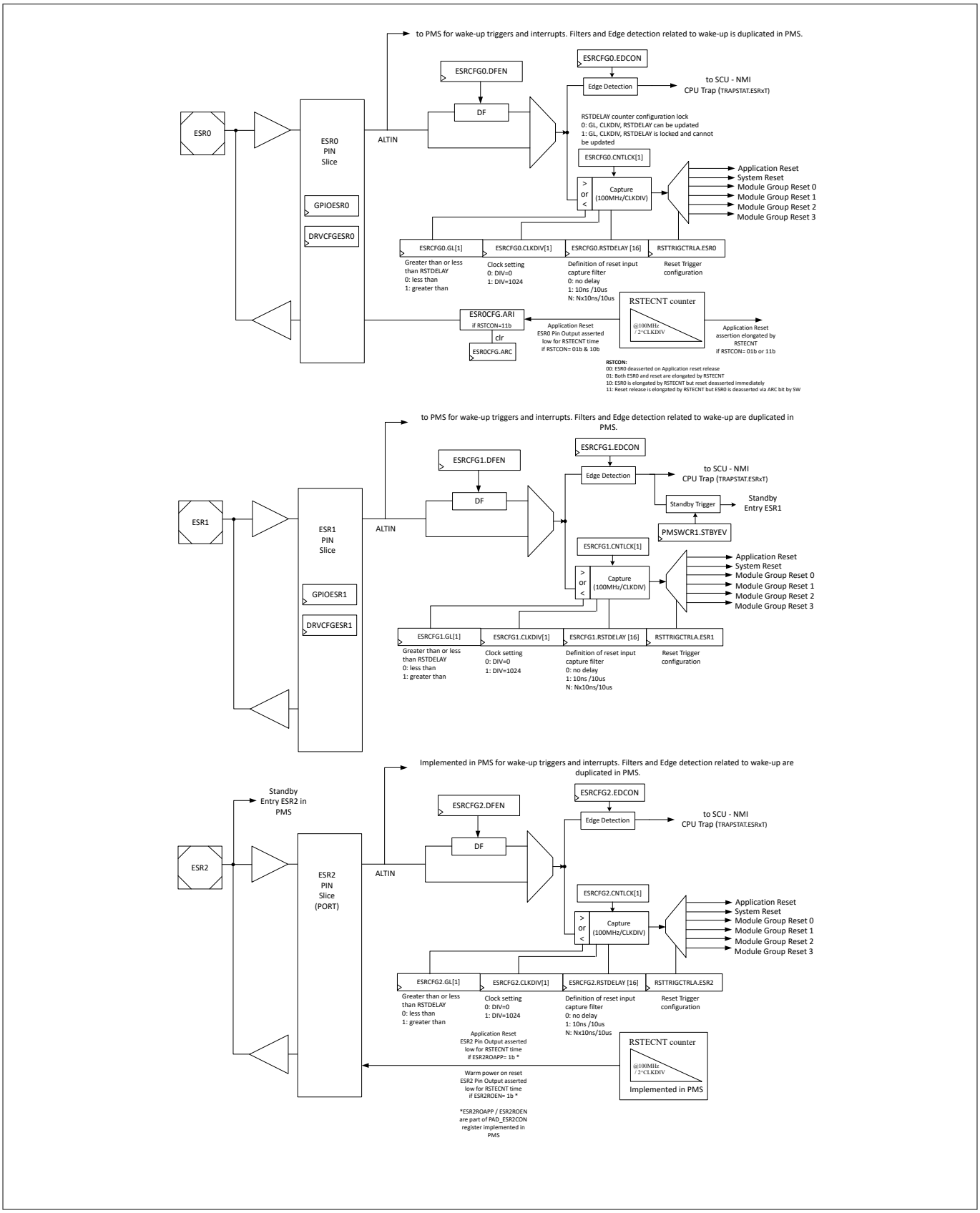


Figure 779 ESR pins connectivity and behavior

36.3.3.6 Primary voltage monitors VMONPx and cold PORST

Primary under-voltage monitoring is carried out for the following supply rails, to ensure proper functioning of the system: V_{DD}, V_{DDEXT}, ($V_{DDEXTDC}$-V_{SSDCHS}), $V_{DDEXTDC}$, V_{DDDCLS}, $V_{DDEVRSB}$, $V_{DDP3NVM}$, V_{DDM}, $V_{DDPHPHYx}$, $V_{DDPHRIFx}$, V_{DDHSIF}, V_{DDPAD}, V_{DDCPUx}, V_{DDPPU}, V_{DDLMUx}, V_{DDSB}, V_{DDPHYx}, and V_{DDRIFx}, depending on product variant (not all supply rails are available on each product variant).

The primary supply monitors are monitoring the ramp-up of external supply voltages and keep the MCU in cold PORST state as long as the supply has not reached the operational region. Likewise, they allow detecting ramp-down or brown-out conditions of the external supplies, such that the device can be brought into a cold PORST state when the voltage has dropped below the lowest operational threshold. Nevertheless, it is recommended to monitor externally all supplies generated external to the MCU and assert the PORST reset pin in case of violation of the lowest operational limits.

The thresholds for the primary monitors are determined by the lowest possible voltages for the correct functioning of the system. The thresholds and their respective tolerance are documented in the datasheet as $V_{xxPRIUV}$ parameters. In case of violation of these thresholds, cold PORST is activated and the PORST pin is pulled low (strong current sink), thus setting the device into reset state. Following the reset release and firmware boot, it can be inferred from the CLDRSTSTAT, CLDRSTRIG, and CLDRSTRIGL registers as to whether the violation of these thresholds led to the previous reset. CLDRSTSTAT shows the asserted reset types which occurred after the last CLDRSTSTATCLR.CLEAR. CLDRSTRIG shows the asserted reset triggers which occurred after the last CLDRSTSTATCLR.CLEAR. The CLDRSTRIGL register shows only the last reset trigger event. After evaluating the reset during start-up, the flags can be cleared through the CLDRSTSTATCLR.CLEAR bit.

The primary under-voltage cold PORST thresholds are trimmed through the VMONP_VxxRST registers. The under-voltage cold PORST triggering is activated or deactivated through the VMONP_VxxRST.RESETOFF bits, individually for each supply rail. There are following exceptions:

- For the ($V_{DDEXTDC}$-V_{SSDCHS}) and V_{DDDCLS} internal supply rails (if available on the respective product variant), the cold PORST triggering is available only in the high-voltage $V_{DDEXTDC}$ mode, when *HWCFG[2:1]* = 11_B. In all other modes, the VMONP_VSSDCHSRST.RESETOFF and VMONP_VDDDCLSRST.RESETOFF bits are ignored and under-voltage cold PORST is not triggered for these rails
- For the $V_{DDEXTDC}$ rail:
  - If EVRC is enabled (when *HWCFG[2:1]* = 01_B, or 10_B, or 11_B or on certain product variants only *HWCFG[1]*=1_B),
    - upon the $V_{DDEXTDC}$ rail under-voltage occurrence, a power state corresponding to an EVRC failure, which then switches off the EVRC. Therefore, the $V_{DDEXTDC}$ primary monitor does not just trigger a cold PORST upon under-voltage, but a deeper power state where the EVRC regulator is switched off and then restarted after the $V_{DDEXTDC}$ voltage recovers
    - Since, the VMONP_VDDEXTDCRST.RESETOFF setting is reset to 0 at each cold PORST, it can occur that this deeper power state is triggered during cold PORST whenever $V_{DDEXTDC}$ falls below the primary monitor under-voltage threshold
  - If EVRC is disabled, the VMONP_VDDEXTDCRST.RESETOFF bit is ignored and no under-voltage reaction is triggered for the $V_{DDEXTDC}$ rail

- For the $V_{DDEVRSB}$ rail:
  - If EVRC is enabled,
    - the primary voltage monitor triggers, upon an under-voltage occurrence, a power state corresponding to an EVRC failure, which then switches off the EVRC (if VMONP_VDDEVRSBRST.RESETOFF = 0)
    - Since the VMONP_VDDEVRSBRST.RESETOFF setting is reset to 0 at each cold PORST, it can occur that this deeper power state is triggered during cold PORST whenever $V_{DDEVRSB}$ falls below the primary monitor under-voltage threshold
  - If EVRC is disabled, the primary voltage monitor triggers, upon an under-voltage occurrence, a cold PORST (if VMONP_VDDEVRSBRST.RESETOFF = 0)
- For all switchable power domain voltage rails (if switchable power domains are available on the respective product variant), the boot firmware always disables the under-voltage cold PORST triggering, at every boot-up, by setting the VMONP_VxxRST.RESETOFF bits to 1. If the application does not switch off these switchable power domains, then in order to enable the under-voltage cold PORST triggering for these power domains, the application software must configure the respective VMONP_VxxRST.RESETOFF bits back to 0

The cold PORST is asserted when the supply voltage drops below the VMONP_VxxRST.RESETTRIM+VMONP_VxxRST.RESETPTRIM value. During cold PORST reset release, to avoid consecutive PORST toggling during slow supply ramp-ups, a voltage hysteresis is supported as configured through the VMONP_VxxRST.RESETHYS bit-field. The cold PORST is de-asserted or released when the supply rises above the RESETTRIM+RESETPTRIM+RESETHYS value. The PORST pin is driven low for a minimum time of t_{EVRPOR} (data sheet parameter) upon recognition of cold PORST, irrespective whether the voltages have been immediately restored, such that there is an adequate time to recognize the PORST event externally.

The VMONP_VxxRST.RESETPTRIM and VMONP_VxxRST.RESETHYS bit-fields are initialized after LVD reset (LVDPMS2), after cold PORST, and by the boot firmware and shall not be re-configured by the application software. Note that after each supply ramp-up and cold PORST primary voltage monitor thresholds for the voltage rails V_{DD}, V_{DDPAD}, V_{DDCPUx}, V_{DDPPU}, V_{DDLMUx}, V_{DDSB}, V_{DDPHYx}, V_{DDRIFx} (not all supply rails are available on each product variant) are configured for the lower voltage level. Therefore after start-up the application software must readjust these thresholds to correct voltage levels for each device type. The correct V_{DD} voltage level for the respective device can be read from the VID_CON0.VDDNOM bits and the targeted thresholds for the different device types are defined in the datasheet table for primary voltage monitoring characteristics.

Refer to the functional description for the specific product variant for an image showing an overview of the primary voltage monitors.

The supplies monitored by primary voltage monitors are measured by primary monitor ADCs and the measured values are updated in the VMONP_VxxSTAT registers, after conversion completion. As indicated in the overview figures for the primary voltage monitors of the respective product variants, some supply rails are measured by tracking ADCs, with filter parameters configured in the VMONP_TRKTRIM0 register. The other supply rails are measured by SAR ADCs, with similar characteristics as the ADCs used for the secondary voltage monitors.

In case of primary monitor cold PORST threshold violation, the status bit CLDRSTSTAT.CLDPORST is set to indicate the event. In addition, the bits corresponding to the respective supply voltage violations are set in the CLDRSTRIG register (CLDRSTRIG.Vxx bits). These bits may be evaluated during consequent start-up after a cold PORST, to recognize which of the monitored supply rails experienced the power failure. If several bits are set in the CLDRSTRIG register, the last reset trigger event can be observed in the CLDRSTRIGL register. After reading the status from these registers, the application software should clear the flags through the CLDRSTSTATCLR.CLEAR bit.

Besides the primary under-voltage cold PORST thresholds, configured in the VMONP_VxxRST registers, a separate set of under-voltage and over-voltage thresholds are configured in the VMONP_VxxCON registers, by programming the UVVAL and OVVAL bit-fields. These under-voltage and over-voltage thresholds are used to configure the operational range of each monitored supply rail.

The violation of the primary under-voltage and over-voltage thresholds (that is of the operational limits), is communicated to the CSRM security module and to the SMU, if the respective under-voltage and over-voltage alarms are enabled, through the UVENABLE, respectively OVENABLE bits in the VMONP_VxxCON registers. The violations of the under-voltage and over-voltage thresholds are indicated in the VMONP_VxxSTAT status flags.

For security reasons, the primary monitor ADC measurements are used to detect voltage fluctuations on the supply rails and to provide alarms also to the CSRM, besides the SMU. The CSRM module may lock access to PMS registers, so that supply generation cannot be influenced by other masters. This measure ensures that malicious software does not manipulate the supplies to gain access to the system.

Note that the VMONP_VSSDCHSCON/VMONP_VSSDCHSRST/VMONP_VSSDCHSSTAT registers refer to the monitor of the ($V_{DDEXTDC}$-V_{SSDCHS}) voltage difference (if V_{SSDCHS} is available on the respective product variant).

The following figure shows the signal flows through a primary voltage monitor and indicates the related register bit-fields for the configuration and status of the primary monitors.

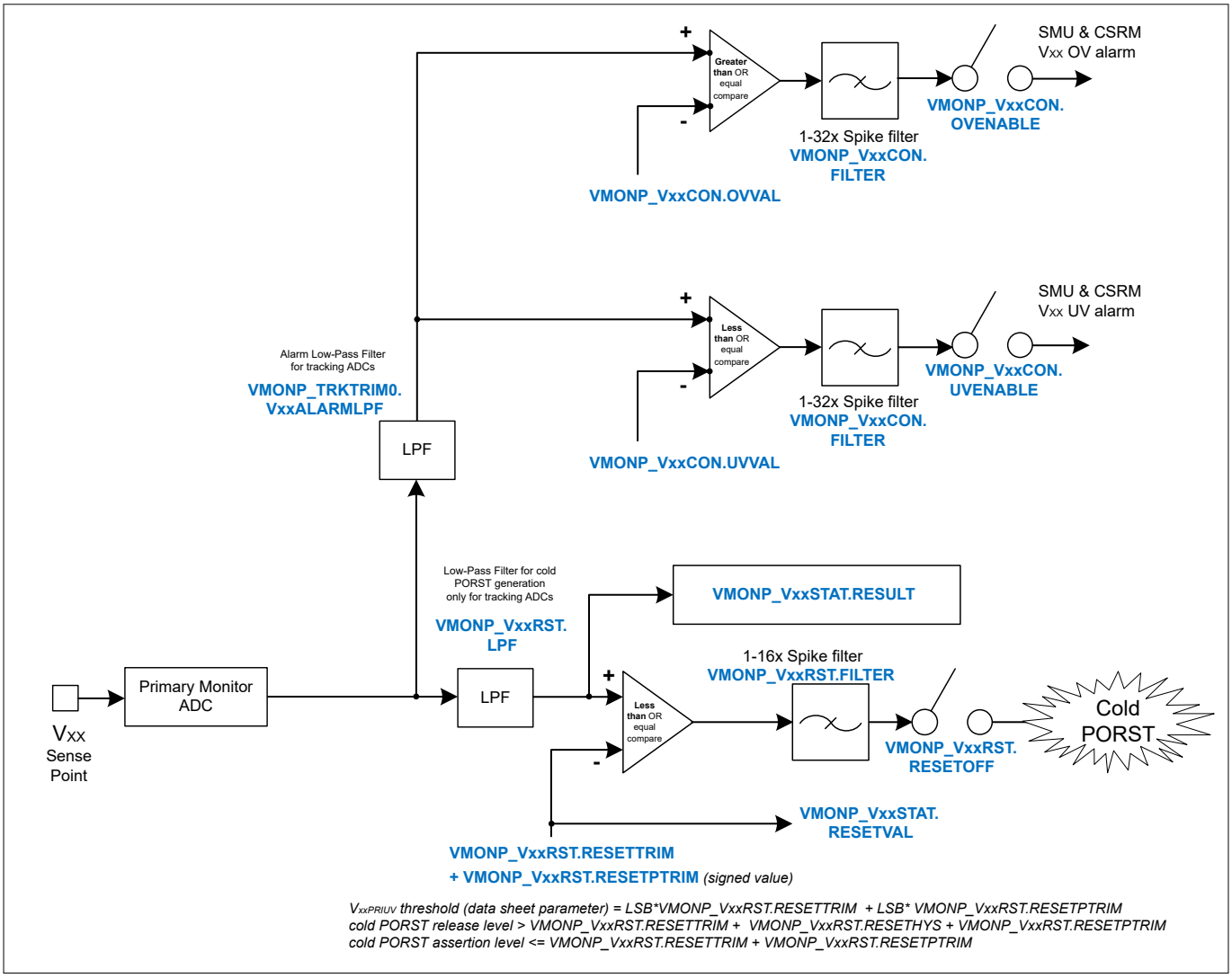


Figure 780 Primary voltage monitors configuration and outputs

Only the supply rails monitored on tracking ADCs (that is on VMONP0-3, see the overview figure) have an implemented low-pass filter (LPF) on the cold PORST trigger path, configured in the VMONP_VxxRST.LPF bit-field. The other supply rails on the SAR ADC (that is on VMONP4, see the overview figure), do not have this filter implemented and, for these supply rails, writing into the VMONP_VxxRST.LPF bit-field has no effect.

The last value measured by the primary monitoring ADCs is updated, for each supply rail, in the VMONP_VxxSTAT.RESULT bit-field, when the respective channel is converted. Conversions occur on a primary ADC channel if any of the following conditions is true:

- The supply rail is monitored by a tracking ADC - the tracking ADC channels are always converted
- If VMONP_VxxCON.UVENABLE or VMONP_VxxCON.OVENABLE is not 0 for the respective rail
- If VMONP_VxxRST.RESETOFF is not 1 for the respective rail
- For the V_{DD}, or V_{DDEXT} or $V_{DDEXTDC}$ rail if peak voltage detection is enabled (PEAK_Vxx.ENABLE = 1)
- For the V_{DD}, or V_{DDEXT} or $V_{DDEXTDC}$ rail if drift monitoring is enabled (DRMON_Vxx.ENABLE = 1)
- If VMONP_VxxCON.UVENABLE and VMONP_VxxCON.OVENABLE have just been set to zero but the conversion already started
- If VMONP_VxxCON.RESETOFF has just been set to 1 but the conversion already started

The VMONP_VxxSTAT.RESULT value is stored without spike filtering.

The under-voltage threshold for triggering the cold PORST is configured in the VMONP_VxxRST.RESETTRIM and VMONP_VxxRST.RESETPTRIM bit-fields, as shown in the previous figure . A spike filter is applied on the cold PORST trigger output, configured in VMONP_VxxRST.FILTER. The cold PORST triggering by the primary monitors can be enabled or disabled through the VMONP_VxxRST .RESETOFF bit. If the software enables the under-voltage reset for a voltage rail, by setting the corresponding VMONP_VxxRST.RESETOFF bit to zero (including bit protection), then the rail voltage will be evaluated and the under-voltage reset is then only asserted if the measured voltage is below the configured threshold and if the under-voltage evaluation passes all configured filters (VMONP_VxxRST.LPF and VMONP_VxxRST.FILTER).

Only the supply rails monitored on tracking ADCs (that is on VMONP0-3, see the overview figure) have an implemented low-pass filter (LPF) on alarm triggering path, configured in the VMONP_TRKTRIM0.VxxALARMLPF bit-fields, for the respective voltage rail (as shown in the 'Primary voltage monitors configuration and outputs' figure, above).

The over-voltage and under-voltage alarm thresholds are configured in the VMONP_VxxCON.OVVAL and VMONP_VxxCON.UVVAL, respectively. Another spike filter is applied on the alarm trigger outputs, configured in VMONP_VxxCON.FILTER. The over-voltage and under-voltage alarms triggering by the primary monitors can be enabled or disabled through the VMONP_VxxCON.OVENABLE, respectively VMONP_VxxCON.UVENABLE bits.

The measurement latency of the monitoring ADCs is determined as follows:

- For core tracking ADCs, the measurement latency is defined by the $t_{TRKCORE}$ data sheet parameter. The time duration for triggering cold PORST or alarms of a supply rail on a core tracking ADC is equal to $t_{TRKCORE}$ multiplied by FILTER + 1 (which is the spike filter setting, either VMONP_VxxRST.FILTER, for cold PORST triggering, or VMONP_VxxCON.FILTER, for alarm triggering) and multiplied by the number of input channels used on the respective tracking ADC. VMONP0 is, for instance, a core tracking ADC, where only one input channel is used
- For high-voltage tracking ADCs, the measurement latency is defined by the t_{TRKHV} data sheet parameter. The time duration for triggering cold PORST or alarms of a supply rail on a high-voltage tracking ADC is equal to t_{TRKHV} multiplied by FILTER + 1 (which is the spike filter setting, either VMONP_VxxRST.FILTER, for cold PORST triggering, or VMONP_VxxCON.FILTER, for alarm triggering) and multiplied by the number of input channels used on the respective tracking ADC. VMONP1, VMONP2, and VMONP3 are examples of high-voltage tracking ADCs. VMONP1 uses only one input channel, whereas VMONP2 and VMONP3 use two input channels
- For core-level channels of a SAR ADC, the measurement latency is defined by the $t_{SARCORE}$ data sheet parameter. The time duration for triggering cold PORST or alarms of a supply rail on a core SAR ADC input is equal to $t_{SARCORE}$ multiplied by FILTER + 1 (which is the spike filter setting, either VMONP_VxxRST.FILTER, for cold PORST triggering, or VMONP_VxxCON.FILTER, for alarm triggering) and multiplied by the number of input channels used on the respective SAR ADC. VMONP4 is a SAR ADC, which uses several core-level and high-voltage input channels
- For high-voltage channels of a SAR ADC, the measurement latency is defined by the t_{SARHV} data sheet parameter. The time duration for triggering cold PORST or alarms of a supply rail on a high-voltage SAR ADC input is equal to t_{SARHV} multiplied by FILTER + 1 (which is the spike filter setting, either

VMONP_VxxRST.FILTER, for cold PORST triggering, or VMONP_VxxCON.FILTER, for alarm triggering) and multiplied by the number of input channels used on the respective SAR ADC

The LSB values for VMONP_VxxRST.RESETTRIM, VMONP_VxxRST.RESETHYS, VMONP_VxxCON.UVVAL, and VMONP_VxxCON.OVVAL values, as well as the measurement latencies are provided in the following tables:

Table 1425 Primary voltage monitors parameters

Voltage rail	VMONP LSB datasheet parameter	Primary under-voltage reset threshold after trimming - datasheet parameter	Effect of primary under-voltage reset threshold violation[1)]
V_{DD}	$V_{TRKCORELSB}$	$V_{DDPRIUV}$	Cold PORST
V_{DDEXT}	$V_{TRKHVLSB}$	$V_{EXTPRIUV}$	Cold PORST
($V_{DDEXTDC}$-V_{SSDCHS})	$V_{TRKHVLSB}$	$V_{DCHSPRIUV}$	Cold PORST only if *HWCFG[2:1]* = 11_B
$V_{DDEXTDC}$	$V_{TRKHVLSB}$	$V_{EXTDCPRIUV}$	Only when EVRC is enabled, a power state transition corresponding to an EVRC failure is triggered. The primary monitor triggers not only a cold PORST, but a deeper power state where the EVRC regulator is disabled, until the supply raises above the reset release threshold.
V_{DDDCLS}	$V_{TRKHVLSB}$	$V_{DCLSPRIUV}$	Cold PORST only if *HWCFG[2:1]* = 11_B
$V_{DDEVRSB}$	$V_{SARHVLSB}$	$V_{EVRSBPRIUV}$	Cold PORST
$V_{DDP3NVM}$	$V_{SARHVLSB}$	$V_{DDP3NVMPRIUV}$	Cold PORST
V_{DDM}	$V_{SARHVLSB}$	$V_{DDMPRIUV}$	Cold PORST
$V_{DDPHPHYx}$	$V_{SARHVLSB}$	$V_{PHPHYPRIUV}$	Cold PORST
$V_{DDPHRIFx}$	$V_{SARHVLSB}$	$V_{PHRIFPRIUV}$	Cold PORST
V_{DDHSIF}	$V_{SARHVLSB}$	$V_{HSIFPRIUV}$	Cold PORST
V_{DDPAD}	$V_{SARCORELSB}$	$V_{DDPADPRIUV}$	Cold PORST
V_{DDCPUx}	$V_{SARCORELSB}$	$V_{DDCPUPRIUV}$	Cold PORST
V_{DDPPU}	$V_{SARCORELSB}$	$V_{DDPPUPRIUV}$	Cold PORST
V_{DDLMUx}	$V_{SARCORELSB}$	$V_{DDLMUPRIUV}$	Cold PORST
V_{DDPHYx}	$V_{SARCORELSB}$	$V_{DDPHYPRIUV}$	Cold PORST
V_{DDRIFx}	$V_{SARCORELSB}$	$V_{DDRIFPRIUV}$	Cold PORST

1) Only if the respective VMONP_VxxRST.RESETOFF bit is set to zero.

Table 1426 Primary voltage monitors measurement latencies

Voltage rail	Measurement latency[1)]
V_{DD}	$t_{TRKCORE}$ * (FILTER+1)
V_{DDEXT}	t_{TRKHV} * (FILTER+1)

(table continues...)

Table 1426 (continued) Primary voltage monitors measurement latencies

Voltage rail	Measurement latency[1]
($V_{DDEXTDC}$-V_{SSDCHS})	t_{TRKHV} * (FILTER+1) * number of channels on VMONP2 ADC
$V_{DDEXTDC}$	t_{TRKHV} * (FILTER+1) * number of channels on VMONP2 ADC
V_{DDDCLS}	t_{TRKHV} * (FILTER+1) * number of channels on VMONP3 ADC
$V_{DDEVRSB}$	t_{SARHV} * (FILTER+1) * number of converted channels on VMONP4 ADC[1]
$V_{DDP3NVM}$	t_{SARHV} * (FILTER+1) * number of converted channels on VMONP4 ADC[1]
V_{DDM}	t_{SARHV} * (FILTER+1) * number of converted channels on VMONP4 ADC[1]
$V_{DDPHPHYx}$	t_{SARHV} * (FILTER+1) * number of converted channels on VMONP4 ADC[1]
$V_{DDPHRIFx}$	t_{SARHV} * (FILTER+1) * number of converted channels on VMONP4 ADC[1]
V_{DDHSIF}	t_{SARHV} * (FILTER+1) * number of converted channels on VMONP4 ADC[1]
V_{DDPAD}	$t_{SARCORE}$ * (FILTER+1) * number of converted channels on VMONP4 ADC[1]
V_{DDCPUx}	$t_{SARCORE}$ * (FILTER+1) * number of converted channels on VMONP4 ADC[1]
V_{DDPPU}	$t_{SARCORE}$ * (FILTER+1) * number of converted channels on VMONP4 ADC[1]
V_{DDLMUx}	$t_{SARCORE}$ * (FILTER+1) * number of converted channels on VMONP4 ADC[1]
V_{DDPHYx}	$t_{SARCORE}$ * (FILTER+1) * number of converted channels on VMONP4 ADC[1]
V_{DDRIFx}	$t_{SARCORE}$ * (FILTER+1) * number of converted channels on VMONP4 ADC[1]

1) The latencies are dependent on how many input channels are converted on the respective ADC. Not all monitoring channels are used on all products, subject to the availability of the respective voltage rails.

Related information

TC4Dx primary voltage monitors on page 6150
Peak voltage detection on page 6006
Drift monitoring on page 6007

36.3.3.6.1 Enabling non-mandatory supply rails

Depending on product variant, there are several supply rails which are not mandatory to be powered at start-up. These supply rails are indicated by the default value of VMONP_VxxRST.RESETOFF = 1_B. This indicates that cold PORST can be released at start-up without powering the respective supply rail. It also means that primary voltage monitoring is not enabled for this rail, but must be enabled by the application SW if this supply rail is required by the application.

If the application requires to use a supply rail with VMONP_VxxRST.RESETOFF = 1_B, then the rail must be powered and the application SW must perform the following steps to enable monitoring and usage of this supply domain:

1. Set VMONP_VxxCON.OVENABLE, to enable primary monitoring ADC conversion on the respective supply rail
2. Poll VMONP_VxxSTAT.RESULT bit-field and compare against VMONP_VxxSTAT.RESETVAL
3. After VMONP_VxxSTAT.RESULT > VMONP_VxxSTAT.RESETVAL, SW should set VMONP_VxxRST.RESETOFF = 0_B (consider setting also the protection bit VMONP_VxxRST.RESETOFF_P in the same write operation)
4. Depending on the monitoring requirements of the application, configure VMONP_VxxCON.OVVAL, VMONP_VxxCON.OVENABLE, VMONP_VxxCON.UVVAL, and VMONP_VxxCON.UVENABLE as needed. As

mentioned in the primary voltage monitors section, the thresholds and alarm functions configured in the VMONP_VxxCON registers

36.3.3.7 Switching off power domains

This section describes the function and usage of switchable power domains, on the product variants which support this feature (see internal supply rails and their availability on each product variant). This section does not apply to product variants which do not support switchable power domains.

The following figure shows the relevant voltage, reset, and load current flows for switching off and on the power domain switches for the CPUx, PPU, LMU, and other switchable power domains. All power domains are active by default and are intended to be switched off after a cold start-up to reduce power consumption. Power domain activation and deactivation is intended to be static in nature and shall be done explicitly at start-up. It is not allowed to dynamically activate / deactivate the power domains during full operational modes.

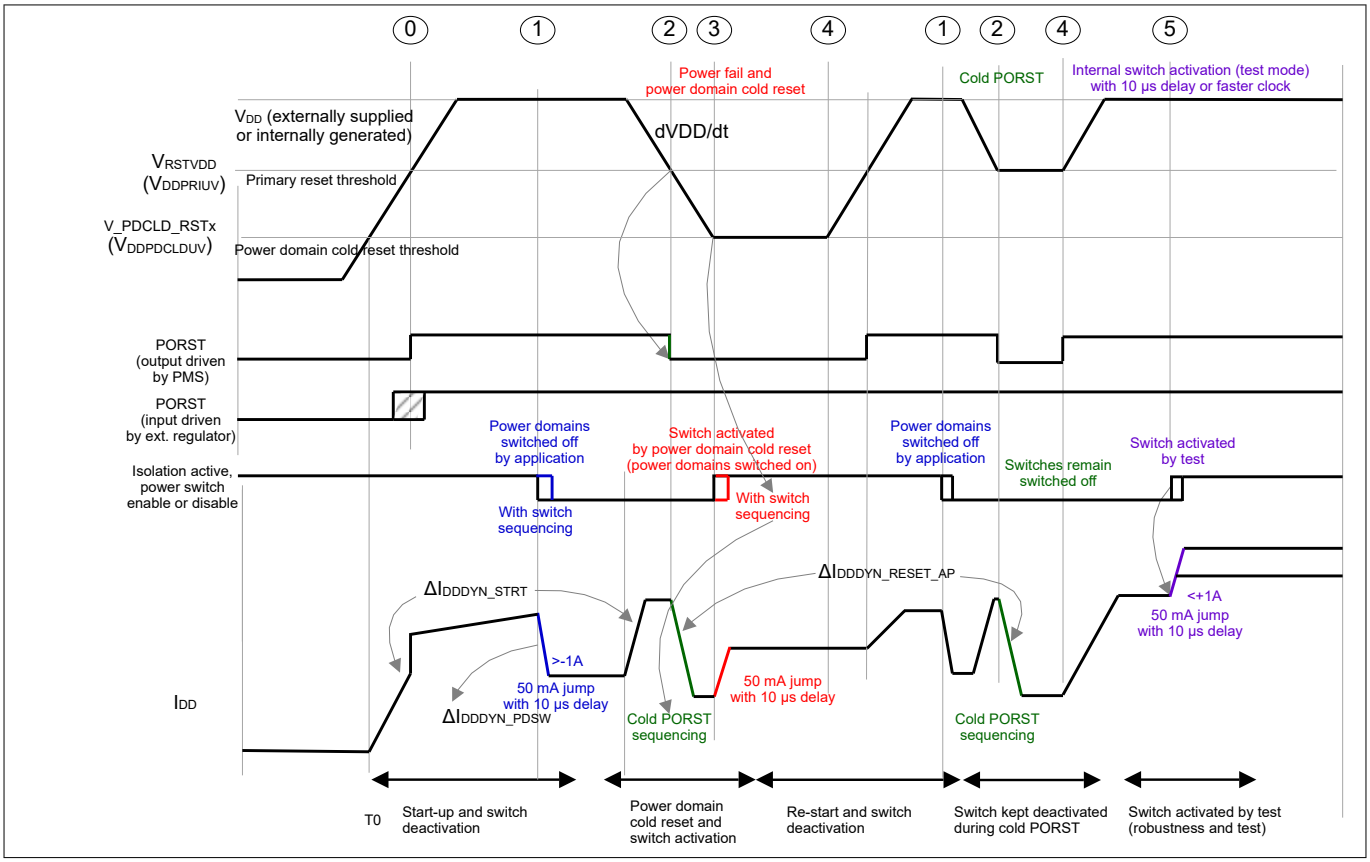


Figure 781 Power domains switch-off

At time point 0, V_{DD} has ramped up and has exceeded the primary reset threshold V_{RSTVDD}. At time point 1, the application switches off the unneeded power domains and the load current drops. At time point 2, a power fail occurs and the V_{DD} voltage crosses the primary under-voltage threshold $V_{DDPRIUV}$. If the voltage continues to drop, it can further cross the power domain cold reset threshold $V_{DDPDCLDUV}$, as shown at time point 3. Later, when the V_{DD} voltage recovers and ramps back up again, it crosses the $V_{DDPDCLDUV}$ threshold, respectively $V_{DDPRIUV}$ (or V_{RSTVDD}) at time points 4 in the previous figure. A test mode activation of the power switches is shown at time point 5.

The PORST is released when all supply voltages, including V_{DD}, have reached their respective primary monitor under-voltage thresholds. After PORST release, the application may switch off the power domains for the CPUx, PPU, or LMUx, by following the next sequence:

- Disable the primary monitor under-voltage reset, primary monitor alarms, and secondary monitor alarms for the domains to be switched off. This step is required before enabling the power domain switch-off by

the application. Otherwise, under-voltage reset or under-voltage alarms could be triggered when the domain is switched off

- Configure the individual clock dividers for the power switches of each power domain as needed in the PD_CON0.CLKCPU, PD_CON0.CLKPPU, or PD_CON0.CLKLMU bit-fields, respectively (depending on power domain availability on the respective product variant)
- Enable the power-down of the respective domains through the *PDEN bits in the PD_CON0 register (setting the respective protection bit(s) is required in the same write operation)
- After the powering down of the respective domains is enabled, the application software can trigger the power-down request for the respective domains through the *OFFREQ bits in the PD_CON1 register (also requiring setting the respective protection bit(s)). This request will switch off the power switches for the selected power domains

The status of the power domain switches can be read by the application from the PD_STAT0 .xxSWSTAT bit-fields (a value of 1 means the power domain is switched off, whereas a value of 0 means the power domain is switched on). Furthermore, from the PD_STAT0.xxREQ bit-fields it can be inferred which power domain switch off was requested after the last cold PORST.

If a power failure occurs and the V_{DD} voltage drops below the primary under-voltage monitor threshold, indicated by the $V_{DDPRIUV}$ parameter, a cold power on reset is triggered by the PMS and the MCU is in reset. If, in addition, the V_{DD} voltage continues to drop further and crosses the power domain cold reset threshold, indicated by the $V_{DDPDCLDUV}$ parameter, the power switches are also reset (re-activated) and the power domains are switched back on. After such a power domain cold reset occurs, the application software has to re-enable the power switches again after the power-up sequence, after the reset release, using the same procedure and registers as described above. When the V_{DD} voltage reaches the $V_{DDPDCLDUV}$ threshold, an immediate (that is asynchronous) cold PORST is triggered by the PMS to the SMM.

If only the cold PORST occurs, without V_{DD} crossing the $V_{DDPDCLDUV}$ threshold, the power switches remain switched off as configured by the application. If the V_{DD} voltage only crosses the $V_{DDPRIUV}$ threshold, but does not cross the lower $V_{DDPDCLDUV}$ threshold, the PMS triggers a sequenced cold PORST trigger to the SMM.

The $V_{DDPDCLDUV}$ threshold parameter is described in the datasheet and is trimmed by firmware using the VMONP_VDDCLDRST.VDDPDCLDUV register bit-field. This reset threshold can be disabled using the VMONP_VDDCLDRST.VDDPDCLDOFF (and respective bit protection) bit. The VMONP_VDDCLDRST.VDDPDCLDUV and VMONP_VDDCLDRST.VDDPDCLDOFF bit-fields shall not be re-configured by the application software. The following figure shows the two reset paths available in the V_{DD} primary monitor, for cold PORST and for power domain (PD) cold reset.

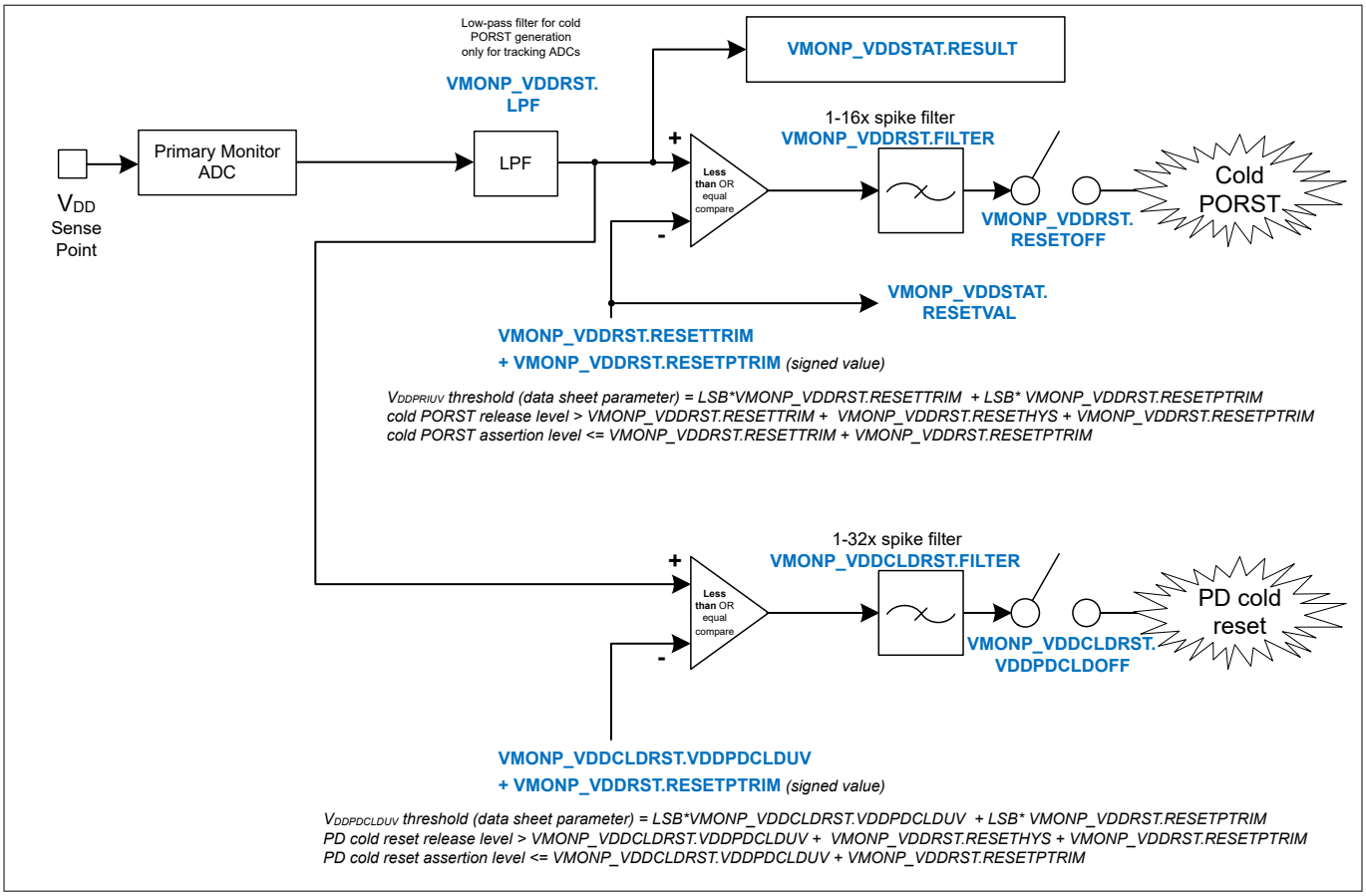


Figure 782 Overview of the reset paths in the V_{DD} primary voltage monitor

36.3.4 Supply generation

This section describes the supply generation function of the PMS, which includes the embedded DCDC regulator (EVRC).

36.3.4.1 Step-down regulator (EVRC)

This section describes the EVRC step-down regulator, generating the V_{DD} supply.

The step-down regulator provides a higher efficiency of power conversion compared to a linear voltage regulator. However it requires additional external components and injects more switching noise into the system. The integrated EVRC regulator modulates an external charge device to buffer the energy in a LC filter in order to generate a regulated digital (core) supply.

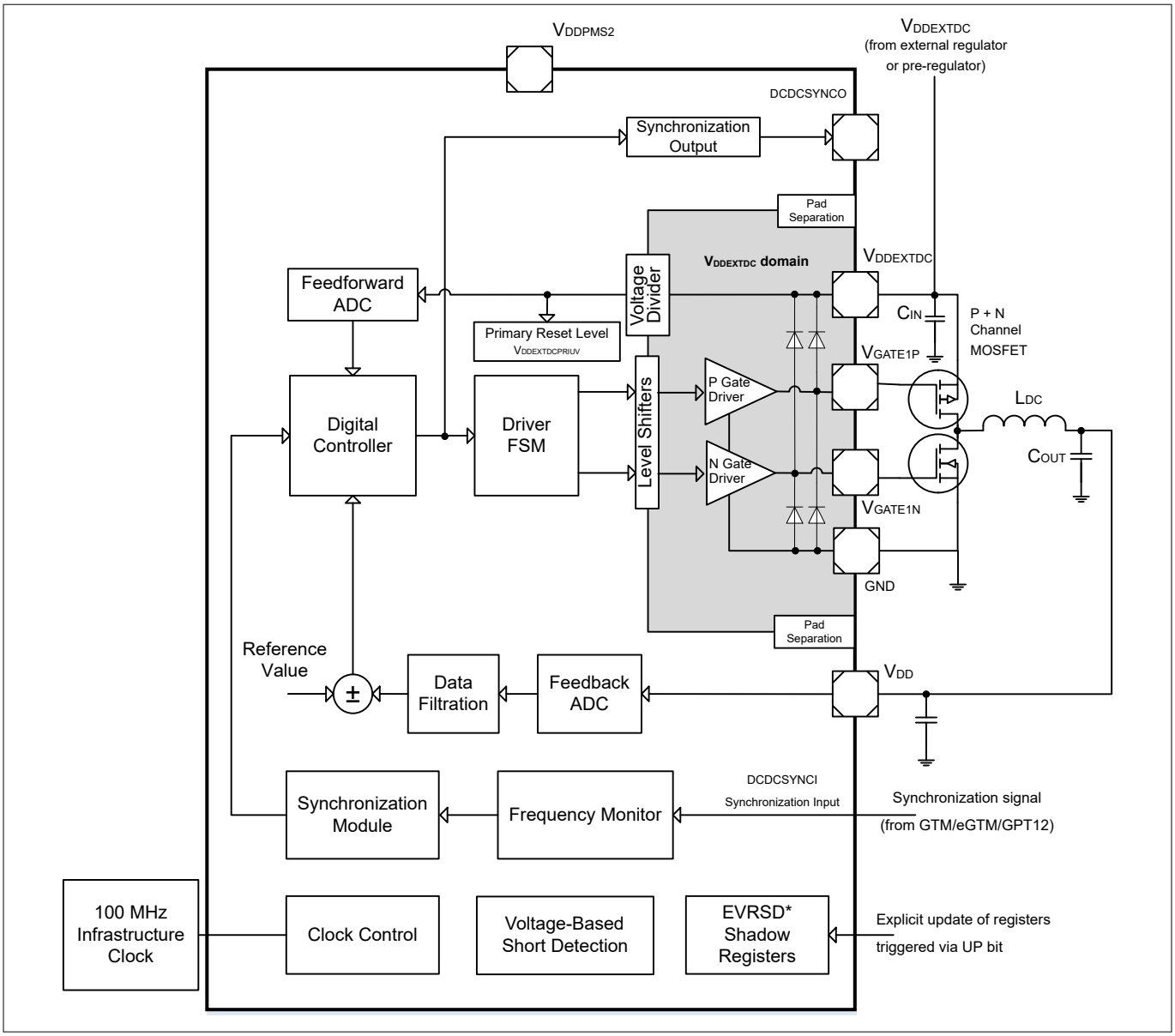


Figure 783 EVRC step-down regulator

The following table gives a brief description of the main components of the EVRC.

Table 1427 EVRC components

Component	Description
Digital Controller	Block implementing the digital regulation loop of the DCDC converter
Driver FSM	FSM adjusting the duty cycle for the gate drivers
P and N Gate Drivers	Internal gate drivers for driving the external power MOSFETs
VGATE1P, *VGATE1N*	Pins connected to the gate of external P-channel and N-channel MOSFETs
Level Shifters	Level shifters converting the drive signals from the V_{DDPMS2} domain to the $V_{DDEXTDC}$ domain

(table continues...)

Table 1427 (continued) EVRC components

Component	Description
Feed-forward ADC (FFADC)	ADC providing the feed-forward information to the digital controller, measuring the $V_{DDEXTDC}$ supply in order to facilitate a parameter switch in case the voltage crosses a threshold
Feedback ADC (FBADC)	Tracking ADC providing the feed-back information to the digital controller, measuring the output V_{DD} voltage. The measured voltage is reported in the EVRC_STAT1.FBADCV bit-field.
Data Filtration	Digital filter applied to the measured output voltage before providing it to the digital controller. Configured through the EVRC_COEFF0.FBADCLPF bit-field
Voltage Divider	Voltage divider for the $V_{DDEXTDC}$ feed-forward voltage for adapting to the valid voltage range of the FFADC. It is part of the FFADC input channels
Primary reset level $V_{DDEXTDCPRIUV}$	This is just an indication of the location of the sense point for the $V_{DDEXTDC}$ primary voltage monitor, which compares the $V_{DDEXTDC}$ with the primary under-voltage reset level $V_{DDEXTDCPRIUV}$
Synchronization Output	Scaled switching output derived from the EVRC switching signal, to synchronize an external regulator to the internal EVRC regulator
Pad Separation	Isolation between the $V_{DDEXTDC}$ pad domain and lower voltage pads
Reference Value	Reference voltage from the secondary high-precision band-gap (HPBG)
DCDCSYNCI Synchronization Input	Synchronization input from the eGTM, or GTM, or GPT12 module to synchronize frequency and phase of the internal EVRC regulator to an external DCDC regulator. The input from GPT12 is only available on certain product variants
Frequency Monitor	Monitoring function to detect when the input synchronization frequency is within the valid range
Synchronization Module	Module performing the frequency and phase synchronization to the input from an external regulator
100 MHz Infrastructure Clock	The PMS infrastructure on-chip clock, also used by the EVRC regulator
Clock Control	Clock control functions
Voltage-Based Short Detection	Detection of EVRC short to supply/ground, based on programmable voltage thresholds (configurable through the EVRC_CON2.SHHTHR, EVRC_CON2.SHLTHR, EVRC_CON2.SHHTIM, and EVRC_CON2.SHLTIM bit-fields)
EVRSD* Shadow Registers	Internal regulator registers that retain their values during standby mode. They are updated from the PMS register settings whenever software triggers a shadow register update through the EVRC_CON0.UP bit. After setting the UP bit, the application SW must wait for the UP bit to be de-asserted, before further writing into EVRC registers which have shadow registers. Otherwise the write accesses would be lost

The EVRC and external MOSFETs must be supplied with the same $V_{DDEXTDC}$ voltage.

Related information

Output voltage setting and voltage scaling on page 5985

Primary voltage monitors VMONPx and cold PORST on page 5971

36.3.4.1.1 Supply voltages

Following supply voltages are applied to the EVRC module:

- V_{DDPMS2}, which is one of the PMS internal supplies, derived from the V_{DDPMS1}supply (using an internal switch), is applied to most of the EVRC regulator components
- $V_{DDEXTDC}$ is the supply from the external pre-regulator and is applied to the complementary MOSFET switches and to the *VDDEXTDC* pin supplying the internal $V_{DDEXTDC}$ domain of the EVRC regulator (mainly including the gate drivers)
- Only on applicable product variants: V_{SSDCHS} is an internal supply within the $V_{DDEXTDC}$ domain, internally generated for the high-side driver
- Only on applicable product variants: V_{DDDCLS} is an internal supply within the $V_{DDEXTDC}$ domain, internally generated for the low-side driver

36.3.4.1.2 Switching frequency

The EVRC step-down converter switches the complementary P-channel and N-channel MOSFETs at a defined switching frequency using pulse width modulation (PWM) and it operates in continuous conduction mode (the inductor current does not fall to zero during the commutation cycle).

The nominal switching frequency is configured through EVRC_CON1.SDFREQ register bits, with a resolution of 100 MHz clock cycles (base clock frequency, that is a switching period resolution of 10 ns). The frequency is derived from the EVRC_CON1.SDFREQ setting as follows:

f_{DCDC} = 100 MHz/EVRC_CON1.SDFREQ

Depending on product variant, up to three nominal frequency settings are supported: 0.45 MHz, 0.8 MHz, and 1.82 MHz. The reset configuration of EVRC_CON1.SDFREQ generates f_{DCDC} = 0.8 MHz. To configure the different switching frequencies, use the following settings:

Table 1428 Parameter configuration for different switching frequencies

Condition	Setting	Comments
f_{DCDC} = 0.45 MHz	Set EVRC_CON1.SDFREQ to DE_H ***Note**: 100 MHz/SDFREQ=0.45 MHz*	Not supported on all product variants. For details on the supported switching frequencies see the data sheet of each product
f_{DCDC} = 0.8 MHz	Set EVRC_CON1.SDFREQ to $7D_H$ ***Note**: 100 MHz/SDFREQ=0.8 MHz*	
f_{DCDC} = 1.82 MHz	Set EVRC_CON1.SDFREQ to 37_H ***Note**: 100 MHz/SDFREQ=1.82 MHz*	Not supported on all product variants. For details on the supported switching frequencies see the data sheet of each product

36.3.4.1.3 Duty cycle

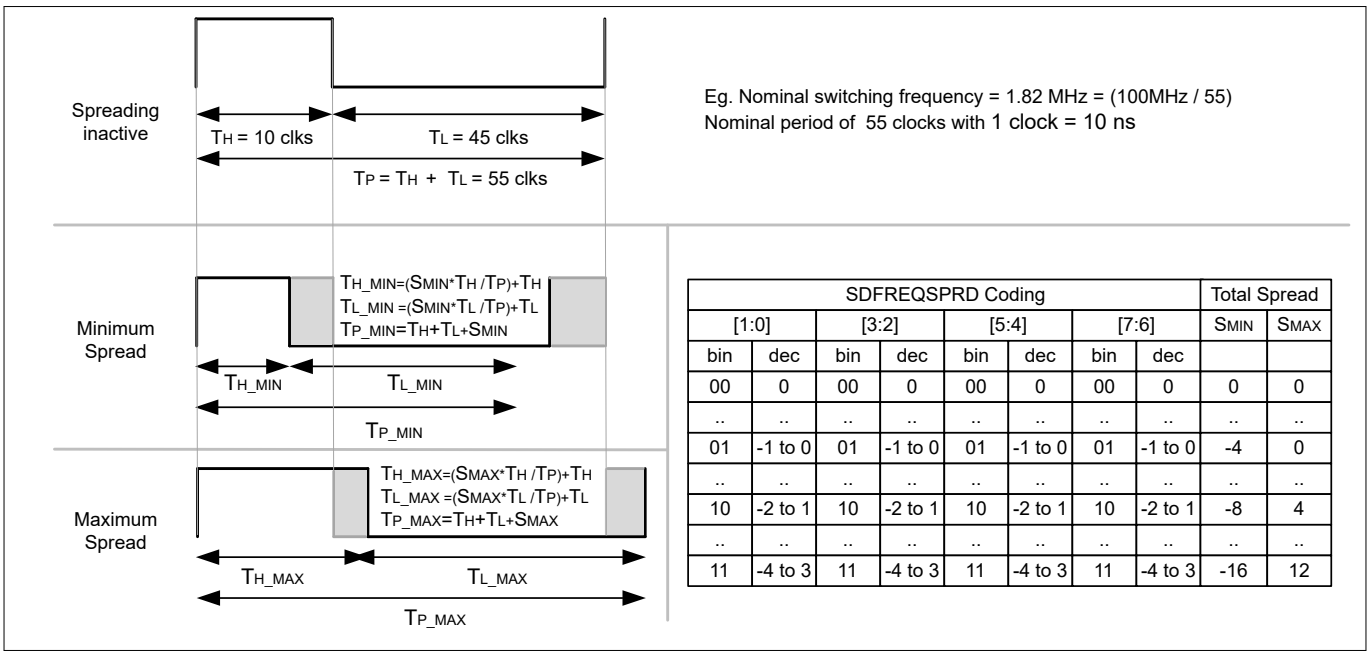
The duty cycle of the switching signal controlling the external power MOSFETs is determined by the input and output voltage values and controlled by the digital controller, which reacts to the deviation of the input and output voltage against a reference voltage. The output voltage value (configured in the EVRC_CON0.SDVOUTSEL register bit-field) is approximately equal to the input voltage multiplied by the duty cycle factor, neglecting the parasitic components and losses. This relationship determines the duty cycle value. The target of the digital controller is to compute the duty cycle and the skip pulse information considering the output voltage measured by the feedback ADC. The duty cycle is limited between a maximum and minimum value (configured through the EVRC_COEFF4.MINTON and MINTOFF bit-fields) in order to ensure proper

commutation in the power switches. The duty cycle is realized by a digital pulse width modulator which controls the respective gate drivers.

The duty cycle of the previous period is indicated in the EVRC_STAT1.DUTYCYCLE status bits. It has a resolution of the base clock frequency of 100 MHz internal clock. In case the controller output is above a certain threshold (configured through the EVRC_CON4.SKPLSTHR) at the start of a switching period and based on internal controller states, a pulse skipping mechanism is employed and the p-channel MOSFET will remain for a whole switching period in the OFF state.

36.3.4.1.4 Frequency spreading

In order to comply with EMI/EMC requirements, frequency spreading is supported. When activated, the switching period is varied randomly between a minimum and a maximum spread around the nominal value as depicted in the following figure.



SDFREQSPRD Coding								Total Spread	
[1:0]		[3:2]		[5:4]		[7:6]		SMIN	SMAX
bin	dec	bin	dec	bin	dec	bin	dec		
00	0	00	0	00	0	00	0	0	0
..	..	..	..	..	..	..	..	..	..
01	-1 to 0	01	-1 to 0	01	-1 to 0	01	-1 to 0	-4	0
..	..	..	..	..	..	..	..	..	..
10	-2 to 1	10	-2 to 1	10	-2 to 1	10	-2 to 1	-8	4
..	..	..	..	..	..	..	..	..	..
11	-4 to 3	11	-4 to 3	11	-4 to 3	11	-4 to 3	-16	12

Figure 784 EVRC regulator switching frequency spreading (without input synchronization)

Only the switching frequency is changed, while it is ensured that duty cycle changes are compensated over consecutive periods, in order to keep the voltage ripple induced by frequency spreading to a minimum.

Frequency spreading is activated and configured by setting the value of the bit-field EVRC_CON1.SDFREQSPRD. The reference setting for enabling frequency spreading is EVRC_CON1.SDFREQSPRD=02_H. An example is presented below.

Assume the switching frequency has been configured to 1.82 MHz (that is EVRC_CON1.SDFREQ=36_H, the switching period is 55 clocks). The nominal duty cycle for a $V_{DDEXTDC}$ supply of 5 V would be:

$D = 0.9\ V/5\ V = 0.18$

This duty cycle value leads to $T_H = 0.18*55 = 10$ clocks (on-time), respectively $T_L = 45$ clocks (off-time). The frequency spreading is configured in EVRC_CON1.SDFREQSPRD, as shown in the figure above. For each of the bit-pairs EVRC_CON1.SDFREQSPRD[1:0], [3:2], [5:4], and [7:6], the spreading factors S_{MIN} and S_{MAX} (both initially zero) are adjusted as follows:

- 00_B: Contributes with 0 to both S_{MIN} and S_{MAX}
- 01_B: Contributes with -1 to S_{MIN} and with 0 to S_{MAX}
- 10_B: Contributes with -2 to S_{MIN} and with +1 to S_{MAX}
- 11_B: Contributes with -4 to S_{MIN} and with +3 to S_{MAX}

Here are a few examples:

- EVRC_CON1.SDFREQSPRD = 0x02$_H$, then, S_{MIN} = -2 and S_{MAX} = 1
- EVRC_CON1.SDFREQSPRD = 0x15$_H$, then, S_{MIN} = -3 (that is $3 \times (-1)$) and S_{MAX} = 0
- EVRC_CON1.SDFREQSPRD = 0x1A$_H$, then, S_{MIN} = -5 (that is $2 \times (-2) + (-1)$) and S_{MAX} = 2 (that is $2 \times (+1) + 0$)
- EVRC_CON1.SDFREQSPRD = 0x7F$_H$, then, S_{MIN} = -13 (that is$3 \times (-4) + (-1)$) and S_{MAX} = 9 (that is $3 \times (+3) + 0$)
- and so on.

For the first example with S_{MIN}=-2 and S_{MAX} = 1, the minimum and maximum high/low state durations, respectively gate signal periods, are calculated as follows:

$T_{H_MIN} = (S_{MIN} * T_H / T_P) + T_H$ = 9.636 clks

$T_{L_MIN} = (S_{MIN} * T_L / T_P) + T_L$ = 43.364 clks

$T_{P_MIN} = T_{H_MIN} + T_{L_MIN}$ = 53 clks

$T_{H_MAX} = (S_{MAX} * T_H / T_P) + T_H$ = 10.182 clks

$T_{L_MAX} = (S_{MAX} * T_L / T_P) + T_L$ = 45.818 clks

$T_{P_MAX} = T_{H_MAX} + T_{L_MAX}$ = 56 clks

In the above example, the frequency would spread between:

f_{MIN} = 100 MHz/T_{P_MAX} = 1.79 MHz

f_{MAX} = 100 MHz/T_{P_MIN} = 1.89 MHz

The frequency spreading unit randomizes the switching frequency of the EVRC within the [f_{MIN}, f_{MAX}] range, programmable as described above. The principle is to randomly discard one clock cycle at 100 MHz from time to time. If this happens, the switching period becomes 1 clock cycle longer compared to the nominal value. The clocks to be removed are selected using a pseudo-random number generator. The clock cycles are skipped from the regular clock train in a random fashion to get uncorrelated noise to the DC/DC control algorithm (and output voltage) and to get an uncorrelated frequency jitter.

If there is a need for concurrently activating both input synchronization and frequency spreading, it is then recommended to program only bit-fields EVRC_CON1.SDFREQSPRD[1:0] and EVRC_CON1.SDFREQSPRD[3:2], leaving SDFREQSPRD[5:4] = 00$_B$ and SDFREQSPRD[7:6] = 00$_B$. In this case, when both features are enabled, the resulting spreading limits S_{MIN} and S_{MAX} are calculated differently, as shown in the following figure.

SDFREQSPRD Coding								Total Spread	
[1:0]		[3:2]		[5:4]		[7:6]		SMIN	SMAX
bin	dec	bin	dec	bin	dec	bin	dec		
00	0	00	0	00	0	00	0	0	0
01	1	00	0	00	0	00	0	-1	1
10	2	00	0	00	0	00	0	-3	3
11	3	00	0	00	0	00	0	-7	7
00	0	01	1	00	0	00	0	-1	1
01	1	01	1	00	0	00	0	-1	1
10	2	01	1	00	0	00	0	-3	3
11	3	01	1	00	0	00	0	-7	7
00	0	10	2	00	0	00	0	-3	3
01	1	10	2	00	0	00	0	-3	3
10	2	10	2	00	0	00	0	-3	3
11	3	10	2	00	0	00	0	-7	7
00	0	11	3	00	0	00	0	-7	7
01	1	11	3	00	0	00	0	-7	7
10	2	11	3	00	0	00	0	-7	7
11	3	11	3	00	0	00	0	-7	7

Figure 785 Frequency spreading limits when input synchronization is also enabled

36.3.4.1.5 Output voltage setting and voltage scaling

The EVRC output voltage level is configured in the EVRC_CON0.SDVOUTSEL bit-field and is preconfigured by the reset value to the nominal V_{DD} value (see the typical value of the V_{DDDCUT} parameter in the datasheet) of the respective product variant. The voltage is calculated as:

V_{DD} = *LSB* * EVRC_CON0.SDVOUTSEL

where LSB = dV_{FBADC} datasheet parameter.

This register bit-field requires a parameter update via EVRC_CON0.UP and reaching of the new target value is indicated via EVRC_STAT0.SDVOK bit.

To change the EVRC output voltage level, primary reset thresholds, and secondary over- and under-voltage thresholds, all settings shall be carried out in a sequential manner as shown below:

- Disable the core supply secondary under-voltage alarm through the VMONS_VDDCON register: VMONS_VDDCON.UVMOD=00_B. Over-voltage monitoring is left active at the limit as configured in VMONS_VDDCON.OVVAL register bits
- VMONS_VDDCON.OVVAL = $V_{DDMON}/V_{SARCORELSB}$ (over-voltage threshold value derived from the datasheet V_{DDMON} example values)
- The EVRC set-point value may be changed in maximum steps of 2.5% of the nominal V_{DD} value, to limit the current dynamics. The default EVRC set-point value is changed in EVRC_CON0.SDVOUTSEL register if the internal EVRC regulator is used: EVRC_CON0.SDVOUTSEL = [(V_{DD} set-point)/LSB], according to the nominal value of the V_{DD} parameter in the datasheet (supply operating conditions section, for the respective voltage ID, VID). If the V_{DD} core supply is provided externally, then the external regulator needs to be configured to the new voltage value. The actual trimmed set-point value of the internal EVRC can be read from the status register EVRC_STAT0.SETPOINT respectively
- The V_{DD} voltage value is measured and is indicated in the EVRC_STAT1.FBADCV register bits. This may allow a plausibility check whether the voltage change happened by comparing the value before and after the change. The voltage change would take less than 20 µs in case of internal regulators to be visible at the V_{DD} rail. In case of the internal EVRC regulator, the reaching of the new set-point is indicated through EVRC_STAT0.SDVOK bit

- The primary reset threshold limit is trimmed through VMONP_VDDRST.RESETTRIM register bits. The reset threshold must be adapted for the V_{DD} voltage change, and is derived from the $V_{DDPRIUV}$ datasheet parameter, corresponding to the selected V_{DD} set-point in the $V_{DDPRIUV}$ parameter conditions. The actual reset trim value used can be read out from the VMONP_VDDSTAT.RESETVAL status register bits
- VMONP_VDDRST.RESETTRIM = $V_{DDPRIUV}/V_{TRKCORELSB}$ (derived from the datasheet $V_{DDPRIUV}$ value)
- If not enough margin is available for the application, secondary under-voltage monitoring of core voltage may be completely disabled. Otherwise the default limit of VMONS_VDDCON.UVVAL bits may be adapted corresponding to the selected V_{DD} set-point, as documented in the V_{DDMON} parameter conditions
- VMONS_VDDCON.UVVAL = $V_{DDMON}/V_{SARCORELSB}$ (under-voltage threshold value derived from the datasheet V_{DDMON} example values)

The following figure shows the available configuration settings and status bit-field for the V_{DD} feedback ADC of the EVRC:

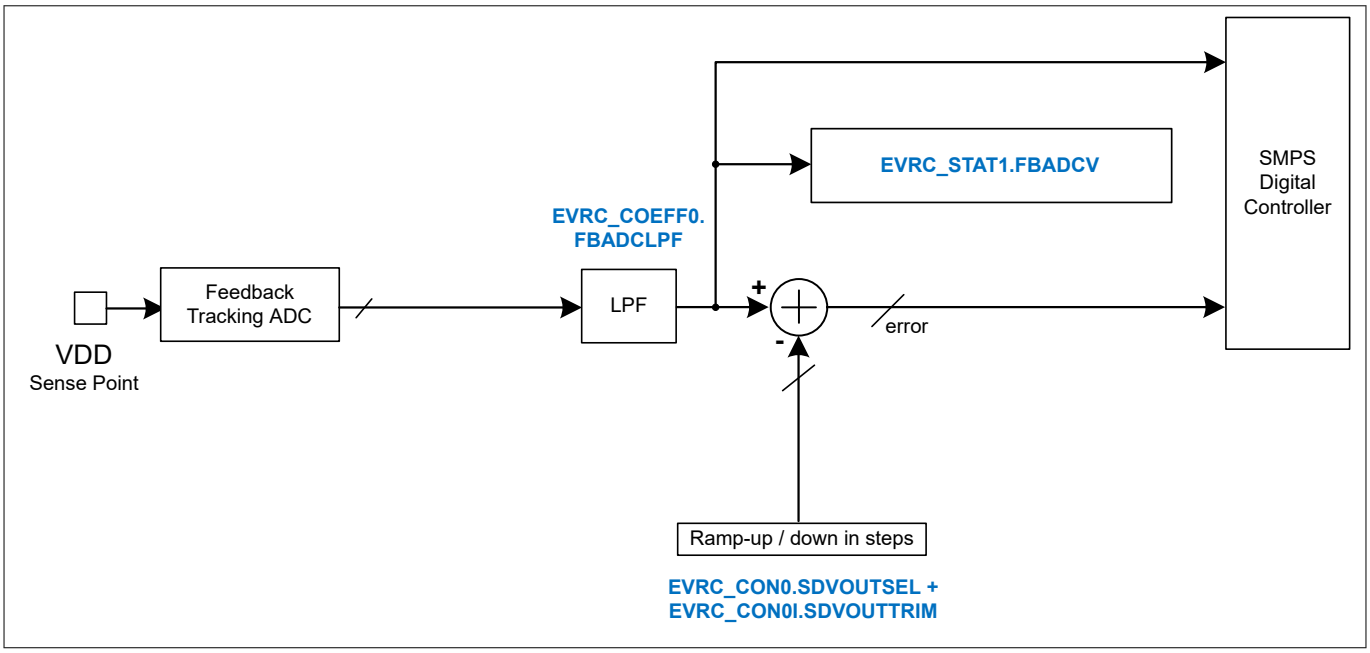


Figure 786 V_{DD} feedback ADC settings

The LPF configuration is done in the EVRC_COEFF0.FBADCLPF bit-field and the last conversion result of the feedback ADC, after the LPF, can be read in the EVRC_STAT1.FBADCV bit-field.

36.3.4.1.6 Driver strength adaptation

Programmable slope control of the external MOSFETs is implemented by controlling the driver strength and slew rate and shall be adapted to the external components. This has an impact on the efficiency, as dynamic switching losses increase with decreasing the switching slope.

The driver strength for the N and P driver current is adjusted in the EVRC_COEFF2 and EVRC_COEFF3 register bit-fields. 16 settings are available, increasing from minimum to full driver strength.

Further, trimming options for the commutation parameters are available:

- For 5 V and high-voltage mode supply, if supported by the respective product variant, in the EVRC_COEFF3 register
- For 3.3 V supply, in the EVRC_COEFF2 register

These settings shall be adapted to the switching transistors.

Related information

Recommended settings

36.3.4.1.7 Load jump and dynamic response

Maximum load jumps on the V_{DD} domain are limited by the range of the nominal voltage for the maximum overshoots/undershoots (see the V_{DDDC} and V_{DD} parameters in the datasheet). The load transients on the V_{DD} rail are described by the dI_{DDDYN_x} parameters in the datasheet. Restricting the maximum load jumps to the limits of the dI_{DDDYN_RIPPLE} parameter leads to better EMI/EMC performance on the system level, which can be achieved by a user application through the gradual MCU frequency/load control.

The reaction of the EVRC on a load jump can be decomposed into three phases:

1. In the first phase, the capacitor is linearly discharged and the regulator has not yet detected the load jump. The ESR voltage peak occurs in this phase, approximated by I_{load_jump} * *ESR*. By increasing the switching frequency, the ESR overshoot can be reduced, as well as the inductor current ripple
2. In the second phase, the regulator has recognized an increase in output voltage and tries to reduce the duty cycle rapidly. In this phase, the inductor current is decreasing linearly and the capacitor voltage is increasing in a parabolic fashion. The ESR voltage peak gradually disappears in the second phase
3. In the third phase, the regulator regulates back to the new load current. The output voltage which reached a maximum at the end of the second phase starts to gradually ramp down to the nominal output voltage. The overshoot is generally limited as documented by the EVRC parameters in the datasheet

36.3.4.1.8 Regulator efficiency

The regulator efficiency varies for different components and frequency cases at different temperatures, supply voltages, and load current. Typical values for different switching frequencies are specified in the datasheet (n_{DC} parameter). The efficiency is optimized for nominal load currents. As the load current reduces or increases beyond this range, the efficiency drops.

The reduction of switching frequency improves the efficiency with the cost of higher ripple, inductor size, and regulator transient response.

36.3.4.1.9 Start-up mode

The EVRC regulator is started during the ramp-up of the $V_{DDEXTDC}$ supply, after the $V_{DDEXTDC}$ voltage has reached the $V_{RSTEXTDC}$ level (datasheet parameter). During the start-up phase, a different control strategy is used, in order to avoid a current overload in the coil, an overshoot of the output V_{DD} voltage, and to charge the output capacitor in shortest time.

During this phase, the EVRC uses the parameter set which has been loaded in the shadow registers after reset (or, if no parameters were loaded, it starts with the reset configuration). The EVRC start-up flow is depicted in the following figure.

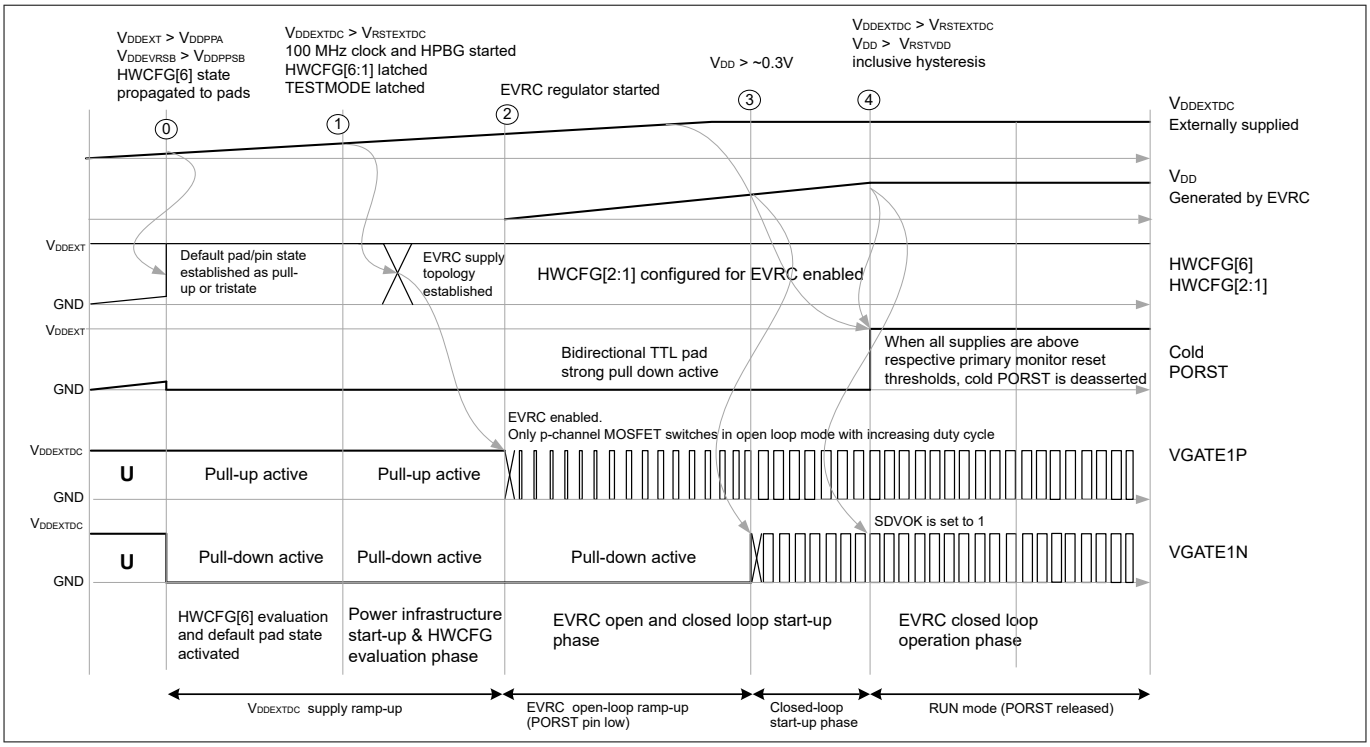


Figure 787 EVRC start-up flow

The start-up procedure occurs as follows:

1. To enable the EVRC regulator, the *HWCFG[1:2]* pins shall be asserted (for example through external or internal pull devices). Its state is latched during $V_{DDEXTDC}$ ramp-up, indicated at time point 1 in the figure above. The latched *HWCFG* pin state can be checked through PADSTAT.HWCFGx status bits and the enable state of EVRC can be checked through the EVRC_STAT0.EVRC flag
2. During the $V_{DDEXTDC}$ ramp-up, the *VGATE1P* signal is in pull-up (PU) state, such that the P-channel MOSFET is completely switched off and the *VGATE1N* pin is in pull-down (PD) state
3. Once the $V_{DDEXTDC}$ voltage reaches the $V_{RSTEXTDC}$ level (data sheet parameter), EVRC is ramping up and starts operating in open-loop mode, as indicated at time point 2 in the figure above. In this mode, only the P-channel MOSFET (*VGATE1P*) is switched on/off, with a gradually increasing duty cycle, starting at a small preset value (programmable through EVRC_COEFF4.MINTON). Soft ramp-up avoids undesired dynamics and too large coil currents, while switching only the P-channel MOSFET facilitates a fast ramp-up to the target voltage without discharges of the capacitor during switching periods. The default switching frequency during open-loop operation is of approximately 0.45 MHz
4. During open-loop operation, the *VGATE1N* signal remains low and keeps the N-channel MOSFET switched off. Only the bulk diode of the NMOS conducts. This prevents negative inductor current in case of a cold MCU reset with pre-charged output buffer capacitor
5. The open-loop operation ends as soon as the output voltage V_{DD}, measured by the feed-back ADC, has crossed a threshold value (characteristic to the EVRC regulator design), as indicated at time point 3 in the figure above . At this point, the controller settings are adapted and the normal closed loop operation begins. The switching frequency is changed to the configured value (in EVRC_CON1.SDFREQ) and the *VGATE1N* pin starts driving the N-channel MOSFET. Both N-channel and P-channel MOSFETs switch with appropriate duty cycle
6. After a voltage transient (typically an overshoot), EVRC is ready and the OK status of the regulator output voltage is asserted in the EVRC_STAT0.SDVOK bit, as indicated at time point 4 in the figure above . Once

the output voltage is stable and all supplies are above the reset threshold, the cold PORST is released. After reset release, EVRC runs in the normal operation Run Mode

7. EVRC_STAT0.SDVOK remains set during Run Mode and is reset only if the V_{DD} set-point was changed or upon a V_{DD} droop request during Run Mode

36.3.4.1.10 Accuracy trimming

After start-up, the firmware (FW) updates the regulator parameters (trimming of the regulator components) to achieve a more accurate EVRC output voltage and improved performance (dynamic regulation). The figure below illustrates the accuracy improvement after trimming.

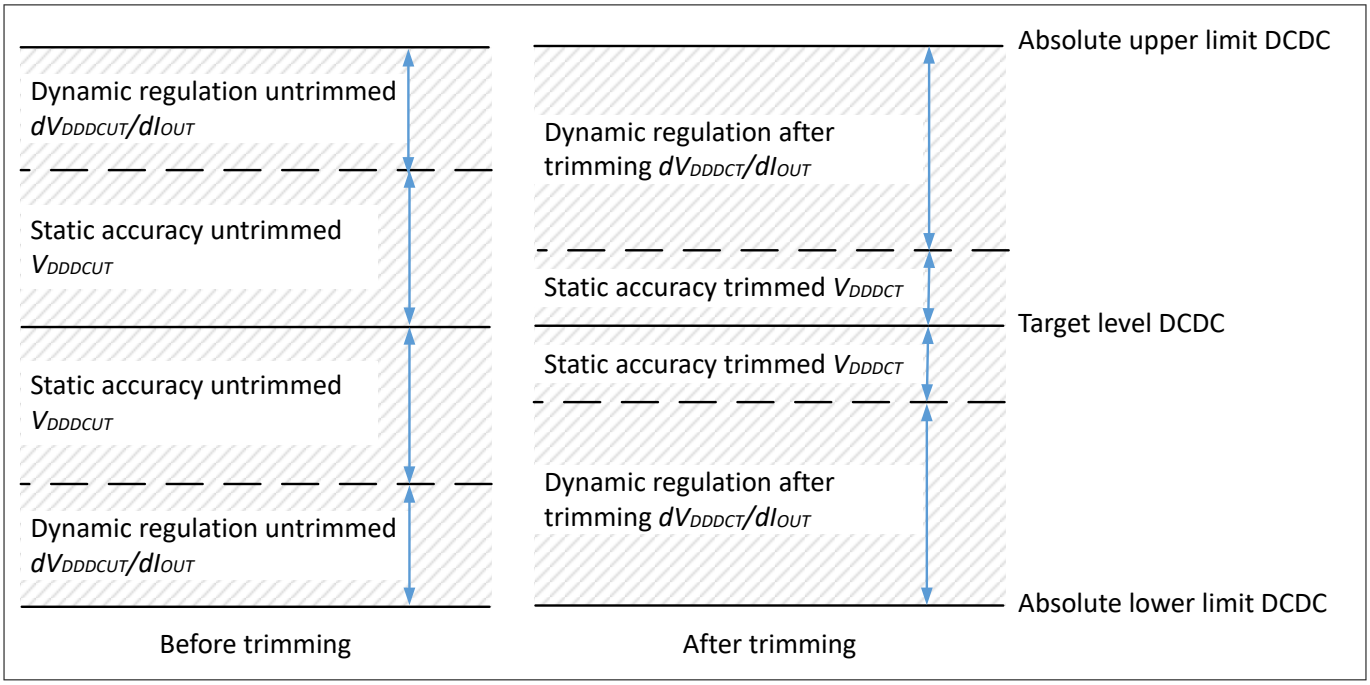


Figure 788 EVRC output voltage accuracy before and after firmware trimming

The overall accuracy can be split into following components:

- Static accuracy: V_{DDDCUT} before trimming, V_{DDDCT} after trimming
- Dynamic response bandwidth of the converter to load jumps: dV_{DDDCUT}/dI_{OUT} before trimming, respectively dV_{DDDCT}/dI_{OUT} after trimming

During the Start-Up phase and firmware execution, the regulator components are in untrimmed state. The static accuracy is V_{DDDCUT}, therefore the allowed dynamic regulation margin is limited to dV_{DDDCUT}/dI_{OUT}. During this phase, the system runs at limited frequency, which is 100 MHz for CPUs/SRI bus and 50 MHz for peripherals, and the total start-up current is approximately 1/3rd of the normal operation current.

Once the firmware trims the analog components, the static accuracy of the regulator improves down to V_{DDDCT}, which allows for larger dynamic margin consequently, as indicated by the dV_{DDDCT}/dI_{OUT} parameter. The regulator can then handle larger load jumps and the application can increase the system frequency.

The register settings of the regulator shall be adapted after start-up (with respect to component trimming) as described in the recommended settings section . Frequency spreading is activated after trimming has been carried out. The register settings for the regulator shall be updated before ramping up the main system clock or activating additional CPU subsystems.

Related information

Recommended settings

36.3.4.1.11 Supply ramp-down

During supply ramp-down phase or standby mode entry, the EVRC ensures a graceful shutdown devoid of output voltage overshoots and coil overload. The gate drivers are always actively keeping off the external MOSFETs during the EVRC shutdown mode.

36.3.4.1.12 Input synchronization

For applications which require DCDC synchronization across the system (potentially for improved EMC), the EVRC regulator features a synchronization input (DCDCSYNCI). This input can be connected to the eGTM, or GTM, or GPT12 module (depending on availability of these modules on the respective product variant) to synchronize the frequency and phase of the PWM signal to an external DCDC regulator (which may for example run at lower switching frequencies than the internal regulator). For this feature, the eGTM, GTM, and GPT12 modules provide two phase synchronized PWM signals:

- A PWM output, routed to the internal EVRC regulator
- A bi-directional (input/output) signal, to be connected to an external DCDC regulator through a port pin, as shown in the figure below

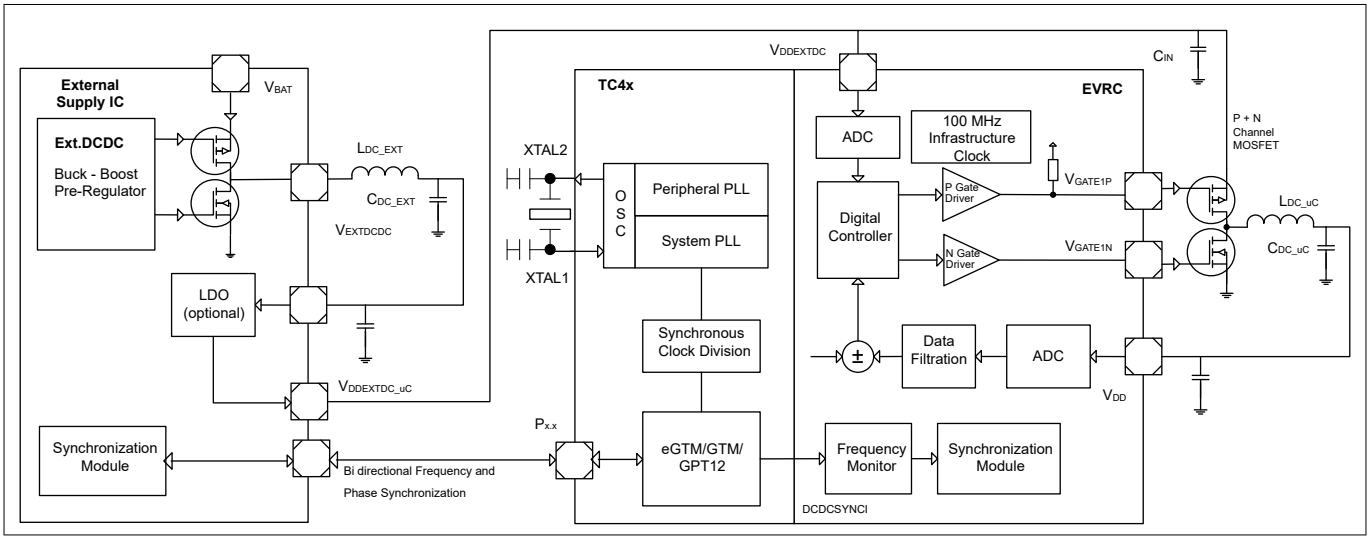


Figure 789 EVRC synchronization input

Important aspects of the synchronization feature are enumerated below:

- The nominal frequencies of the synchronization input are, depending on product variant, 1.82 MHz, 0.8 MHz, or 0.45 MHz, as specified in the datasheet ($f_{DCDCSYNC}$). A bi-directional signal to/from the external regulator is provided by the eGTM, or GTM, or GPT12 module
- Frequency monitoring is carried out in the region between the minimum and maximum values of the $f_{DCDCSYNC}$ parameter, as specified in the datasheet
- Loss of synchronization lock event is triggered if the switching frequency range is violated. The loss of synchronization lock event is indicated through status bit (EVRC_STAT0.SYNCLCK) and interrupt (see INT_CON2.DCDCSYNCLCK bit)
- Lock hysteresis is provided for the synchronization lock signal to avoid limit cycling behavior, as defined in the EVRC_CON1.SYNCHYST bit-field
- The ARU-connected Timer Output Module (ATOM) channels ATOM [0-8]_1 and the Timer Output Module (TOM) channels TOM [0-5]_1 of either the GTM or the eGTM functional block, or the T6OUT output from the GPT12 module (depending on availability of these modules on the respective product variant) can be connected internally to the EVRC synchronization input

Requirements for the synchronization function

In order to use the input synchronization function, the following requirements must be met:

- Synchronization to an external signal is only supported in normal PWM operation mode (run mode). The feature is not supported during start-up mode
- The frequency of the input synchronization signal shall be within the supported monitoring range, as defined in the datasheet (see $f_{DCDCSYNC}$ parameter)
- Component tolerances shall support the frequency monitoring range and the dynamics introduced in case of synchronization loss
- The maximum tolerable deviation from the nominal EVRC switching frequency (configured in EVRC_CON1.SDFREQ) shall be within a configurable maximum deviation, which is configured in EVRC_CON1.SYMCMAXDEV
- The duty cycle of the input signal is not monitored, but must be wide enough to allow proper sampling of the rising edges (minimum 2 cycles of the 100 MHz clock, that is 20 ns)

The synchronization signal may be subjected to delays and jitter when it crosses over from the system PLL domain to the back-up clock domain, which would have an impact on the static accuracy of the EVRC regulator.

Frequency step changes are limited to ±100 kHz from the eGTM, or GTM, or GPT12 module and the dynamic frequency change in case of loss of synchronization is limited to ±200 kHz.

Enabling or disabling synchronization

The synchronization function can be enabled or disabled by configuring the EVRC_CON3.SYNCIN register bit.

Synchronization lock status

The locked or unlocked status of the synchronization function is indicated by the EVRC_STAT0.SYNCLCK status bit.

Hysteresis for synchronization locking and unlocking

A hysteresis is applied for the locking and unlocking, such that toggling lock behavior is avoided around the frequency monitoring limits. The hysteresis width is configured in the EVRC_CON1.SYNCHYST register bit-field. The recommended setting for the EVRC_CON1.SYNCHYST register bit-field for the different f_{DCDC} is presented in the following table,

Table 1429 Recommended SYNCHYST settings for different switching frequencies

Condition	Recommended setting
f_{DCDC} = 0.45 MHz	Set EVRC_CON1.SYNCHYST to 001001_B
f_{DCDC} = 0.8 MHz	Set EVRC_CON1.SYNCHYST to 000101_B
f_{DCDC} = 1.82 MHz	Set EVRC_CON1.SYNCHYST to 000010_B

The EVRC locks to the synchronization signal if its period is within the following range:

- Sync. signal period < EVRC_CON1.SDFREQ + EVRC_CON1.SYNCMAXDEV - EVRC_CON1.SYNCHYST
- Sync. signal period > EVRC_CON1.SDFREQ - EVRC_CON1.SYNCMAXDEV + EVRC_CON1.SYNCHYST

Synchronization is unlocked when the synchronization signal leaves the hysteresis range, that is:

- Sync. signal period > EVRC_CON1.SDFREQ + EVRC_CON1.SYNCMAXDEV
- Sync. signal period < EVRC_CON1.SDFREQ - EVRC_CON1.SYNCMAXDEV

where EVRC_CON1.SDFREQ is the period of the EVRC switching frequency f_{DCDC} , EVRC_CON1.SYNCMAXDEV is the maximum deviation, expressed in 100 MHz cycles, and EVRC_CON1.SYNCHYST is the locking hysteresis, also expressed in 100 MHz cycles.

The maximum deviation EVRC_CON1.SYNCMAXDEV must be chosen to be greater than the value of EVRC_CON1.SYNCHYST and unequal to zero (otherwise the EVRC will lock to the synchronization input only if

its period matches the period programmed in EVRC_CON1.SDFREQ). Also the maximum frequency monitoring range (as defined in the datasheet) has to be considered.

Loss of synchronization

In case the synchronization input signal violates the programmed limits, the EVRC unlocks and continues switching with the programmed nominal switching period (as defined in EVRC_CON1.SDFREQ). The dynamics introduced by the synchronization loss are limited to ±200 kHz and must be supported by component tolerances, as previously mentioned (see requirements for the synchronization function). The loss of synchronization lock event is indicated by an interrupt which can be enabled through the INT_CON2.DCDCSYNCLCK register bit.

Frequency spreading

Frequency spreading is available and can be used when the synchronization input is enabled. When enabled, the synchronized rising edge of the PWM signal is delayed by a random number of cycles, as specified in the section related to frequency spreading.

However, frequency spreading may be deactivated (in the EVRC_CON1.SDFREQSPRD register bit-field), since synchronization inherently provides additional jitter on the switching signals, which may be adequate for the EMI/EMC performance. It is therefore recommended to deactivate frequency spreading when using input synchronization, since the combination of frequency spreading and synchronization jitter may increase the V_{DD} voltage ripple.

Synchronization lock procedure

During the synchronization lock phase, an application is expected to perform the following steps:

- Check the EVRC nominal switching frequency in the EVRC_CON1.SDFREQ register bits. The configured nominal frequency must be either 1.82 MHz, or 0.8 MHz, or 0.45 MHz (depending on the frequencies supported by the respective product variant, and according to the application requirements)
- Configure EVRC_CON1.SYNCMAXDEV and SYNCHYST fields. These values are responsible for the frequency deviation from the nominal frequency and the hysteresis window for locking and unlocking respectively. Violation of the SYNCMAXDEV limits leads to a loss of synchronization lock
- Select a synchronization input in EVRC_CON3.SYNCMUXSEL. The eGTM, or GTM, or GPT12 module shall be configured accordingly for the input signal. The output signal to the EVRC and to the external DCDC regulator is either:
  - GTM LCDCDCOUT signal, selected through the LCDCDCOUTSEL register in the GTM module
  - eGTM LCDCDCOUT signal, selected through the LCDCDCOUTSEL register in the eGTM module
  - GPT12 T6OUT signal

  The signal to the external DCDC regulator can be provided with a phase shift, if necessary
- When no load/line transients are ongoing, EVRC_CON3.SYNCIN shall be set to initiate the synchronization lock procedure
- A successful synchronization lock is reflected in the EVRC_STAT0.SYNCLCK status bit. The procedure takes less than 4 switching periods

Synchronization unlock procedure

Before entering standby mode, the synchronization unlock procedure shall be performed:

- The EVRC_CON3.SYNCIN bit shall be reset to initiate the synchronization unlock procedure
- As a result, the EVRC_STAT0.SYNCLCK bit is set into unlocked state. The Synchronization unlock interrupt is generated if enabled (in INT_CON2.DCDCSYNCLCK). The synchronization unlock procedure takes less than 4 switching periods. The synchronization logic is consequently disabled
- The synchronization input signal from the GTM, or eGTM, or GPT12 module is consequently deselected through configuration of the EVRC_CON3.SYNCMUXSEL bit-field by software. The phase shifted signal to the external DCDC regulator from GTM, or eGTM, or GPT12 is also deactivated

Synchronization unlock event

If the period of the incoming synchronization signal is longer than the nominal period + maximum deviation or shorter than nominal period - maximum deviation (see the above section on locking/unlocking hysteresis), a synchronization unlock event is triggered:

- The EVRC_STAT0.SYNCLCK status bit switches to unlocked state. The synchronization unlock interrupt is generated if enabled (in INT_CON2.DCDCSYNCLCK)
- EVRC switches back to the internal 100 MHz clock source at the default switching frequency with an arbitrary phase. The dynamic voltage transient may be higher during loss of synchronization with consequent load jump and might violate voltage accuracy limits during this frequency switch
- A re-lock is initiated automatically according to the hysteresis, that is only if the incoming signal period is less than:
  - Nominal period + maximum deviation - sync hysteresis

  and higher than:
  - Nominal period - maximum deviation + sync hysteresis

  (see the above section on locking/unlocking hysteresis for the respective settings)

Related information

Start-up mode on page 5987
Switching frequency on page 5982
Frequency spreading on page 5983
Accuracy trimming on page 5989

36.3.4.1.13 Short detection

The EVRC features a voltage-based detection mechanism for short to supply (short to high voltage) and short to ground (short to low voltage) failures.

The short-to-high voltage detection can be enabled by application software through the EVRC_CON2.SHHEN bit. The detection threshold is determined using the EVRC_CON2.SHHTHR bit-field, which can be adjusted by application software:

Short-to-high detection voltage threshold = (EVRC_CON0.SDVOUTSEL + EVRC_CON2.SHHTHR) * $V_{TRKCORELSB}$

The short-to-high detection is triggered when the EVRC output voltage exceeds the configured threshold for a time longer or equal to the detection time configured in the EVRC_CON2.SHHTIM bit-field. The EVRC_CON2.SHHTIM bit-field can be adjusted by the application software and is configured in milliseconds.

The short-to-low voltage detection can be enabled by application software through the EVRC_CON2.SHLEN bit. The detection threshold is determined using the EVRC_CON2.SHLTHR bit-field, which can be adjusted by application software:

Short-to-low detection voltage threshold = (EVRC_CON0.SDVOUTSEL - EVRC_CON2.SHLTHR) * $V_{TRKCORELSB}$

The short-to-low detection is triggered when the EVRC output voltage drops below the configured threshold for a time longer or equal to the detection time configured in the EVRC_CON2.SHLTIM bit-field. The EVRC_CON2.SHLTIM bit-field can be adjusted by the application software and is configured in milliseconds.

If the short detection is enabled (using EVRC_CON2.SHHEN or EVRC_CON2.SHLEN), an alarm is triggered to the SMU in case of a short detection.

If desired by the application, in addition to the short detection alarm, a switch-off of the EVRC can be triggered after a short detection. The switch-off of the EVRC can be enabled by application software through the EVRC_CON2.SHOFFEN bit. If this bit is enabled, the EVRC will be switched off with a delay of approximately 3 ms after a short detection.

36.3.4.1.14 Recommended settings

This section presents a few examples of settings for different load capabilities, switching frequencies, and external voltages.

Application-dependent settings

For different application profiles, the following parameters are configured:

Table 1430 Parameter configuration for different application profiles

Condition	Parameters
MOSFETs BUK4D38-20P and BUK4D16-20, $I_{DD} \le I_{DDDCMAX}$ HE case, f_{DCDC} = 0.8 MHz	• EVRC_CON2 = 0x33C29F0 • EVRC_CON3 = 0x1200 • EVRC_CON4 = 0x60 • EVRC_COEFF2 = 0x02F7EFFD • EVRC_COEFF3 = 0x24F8CFAF • EVRC_COEFF4 =0x00664664 • EVRC_COEFF5 = 0x0000FFFF • Set EVRC_CON0.UP = 1 • Wait for 30 µs • Check that EVRC_CON0.UP is de-asserted, then continue • EVRC_CON1 = 0x098A007D • EVRC_COEFF0 = 0x8483C014 • EVRC_COEFF1 = 0x09C01EB8 • Set EVRC_CON0.UP = 1 • Wait for 20 µs • Check that EVRC_CON0.UP is de-asserted, then continue • EVRC_CON1 = 0x098A007D • EVRC_CON4=0x000000B1 • EVRC_COEFF0 = 0x84942003 • EVRC_COEFF1 = 0x0A3000C6 • Set EVRC_CON0.UP = 1 • Wait for 30 µs • Check that EVRC_CON0.UP is de-asserted, then continue

(table continues...)

Table 1430 (continued) Parameter configuration for different application profiles

Condition	Parameters
MOSFETs BUK4D38-20P and BUK4D16-20, $I_{DD} \leq I_{DDDCMAX}$ HE case, f_{DCDC} = 0.45 MHz	• EVRC_CON2 = 0x33C29F0 • EVRC_CON3 = 0x1200 • EVRC_CON4 = 0x60 • EVRC_COEFF2 = 0x02F7EFFD • EVRC_COEFF3 = 0x24F8CFAF • EVRC_COEFF4 =0x00664664 • EVRC_COEFF5 = 0x0000FFFF • Set EVRC_CON0.UP = 1 • Wait for 30 µs • Check that EVRC_CON0.UP is de-asserted, then continue • EVRC_CON1 = 0x0D9200DE • EVRC_COEFF0 = 0x64BCA324 • EVRC_COEFF1 = 0x0A232299 • Set EVRC_CON0.UP = 1 • Wait for 20 µs • Check that EVRC_CON0.UP is de-asserted, then continue • EVRC_CON1 = 0x0D9200DE • EVRC_CON4 = 0x000000C1 • EVRC_COEFF0 = 0x64BCA324 • EVRC_COEFF1 = 0x0A232299 • Set EVRC_CON0.UP = 1 • Wait for 30 µs • Check that EVRC_CON0.UP is de-asserted, then continue

(table continues...)

Table 1430 (continued) Parameter configuration for different application profiles

Condition	Parameters
MOSFETs SQA407CEJW and SQA446CEJW, $I_{DD} \leq I_{DDDCMAX}$ HE case, f_{DCDC} = 0.8 MHz	• EVRC_CON2 = 0x33C29F0 • EVRC_CON3 = 0x1200 • EVRC_CON4 = 0x60 • EVRC_COEFF2 = 0x02F75FA9 • EVRC_COEFF3 = 0x04F88F6A • EVRC_COEFF4 =0x00664664 • EVRC_COEFF5 = 0x00148F8C • Set EVRC_CON0.UP = 1 • Wait for 30 µs • Check that EVRC_CON0.UP is de-asserted, then continue • EVRC_CON1 = 0x098A007D • EVRC_COEFF0 = 0x8483C014 • EVRC_COEFF1 = 0x09C01EB8 • Set EVRC_CON0.UP = 1 • Wait for 20 µs • Check that EVRC_CON0.UP is de-asserted, then continue • EVRC_CON1 = 0x098A007D • EVRC_CON4 = 0x000000B1 • EVRC_COEFF0 = 0x84942003 • EVRC_COEFF1 = 0x0A3000C6 • Set EVRC_CON0.UP = 1 • Wait for 30 µs • Check that EVRC_CON0.UP is de-asserted, then continue

(table continues...)

Table 1430 (continued) Parameter configuration for different application profiles

Condition	Parameters
MOSFETs SQA407CEJW and SQA446CEJW, $I_{DD} \leq I_{DDDCMAX}$ HE case, f_{DCDC} = 0.45 MHz	• EVRC_CON2 = 0x33C29F0 • EVRC_CON3 = 0x1200 • EVRC_CON4 = 0x60 • EVRC_COEFF2 = 0x02F75FA9 • EVRC_COEFF3 = 0x04F88F6A • EVRC_COEFF4 =0x00664664 • EVRC_COEFF5 = 0x00148F8C • Set EVRC_CON0.UP = 1 • Wait for 30 µs • Check that EVRC_CON0.UP is de-asserted, then continue • EVRC_CON1 = 0x0D9200DE • EVRC_COEFF0 = 0x64BCA324 • EVRC_COEFF1 = 0x0A232299 • Set EVRC_CON0.UP = 1 • Wait for 20 µs • Check that EVRC_CON0.UP is de-asserted, then continue • EVRC_CON1 = 0x0D9200DE • EVRC_CON4 = 0x000000C1 • EVRC_COEFF0 = 0x64BCA324 • EVRC_COEFF1 = 0x0A232299 • Set EVRC_CON0.UP = 1 • Wait for 30 µs • Check that EVRC_CON0.UP is de-asserted, then continue

External voltage settings

There is no specific EVRC parameter setting for $V_{DDEXTDC}$ = 3.3 V vs. $V_{DDEXTDC}$ = 5 V, this configuration is set at start-up through the *HWCFG* pins.

36.3.4.2 External regulator interface and voltage scaling

This section describes the interface between PMS and an external regulator or power management IC and provides details on the voltage scaling procedure.

External regulator interface

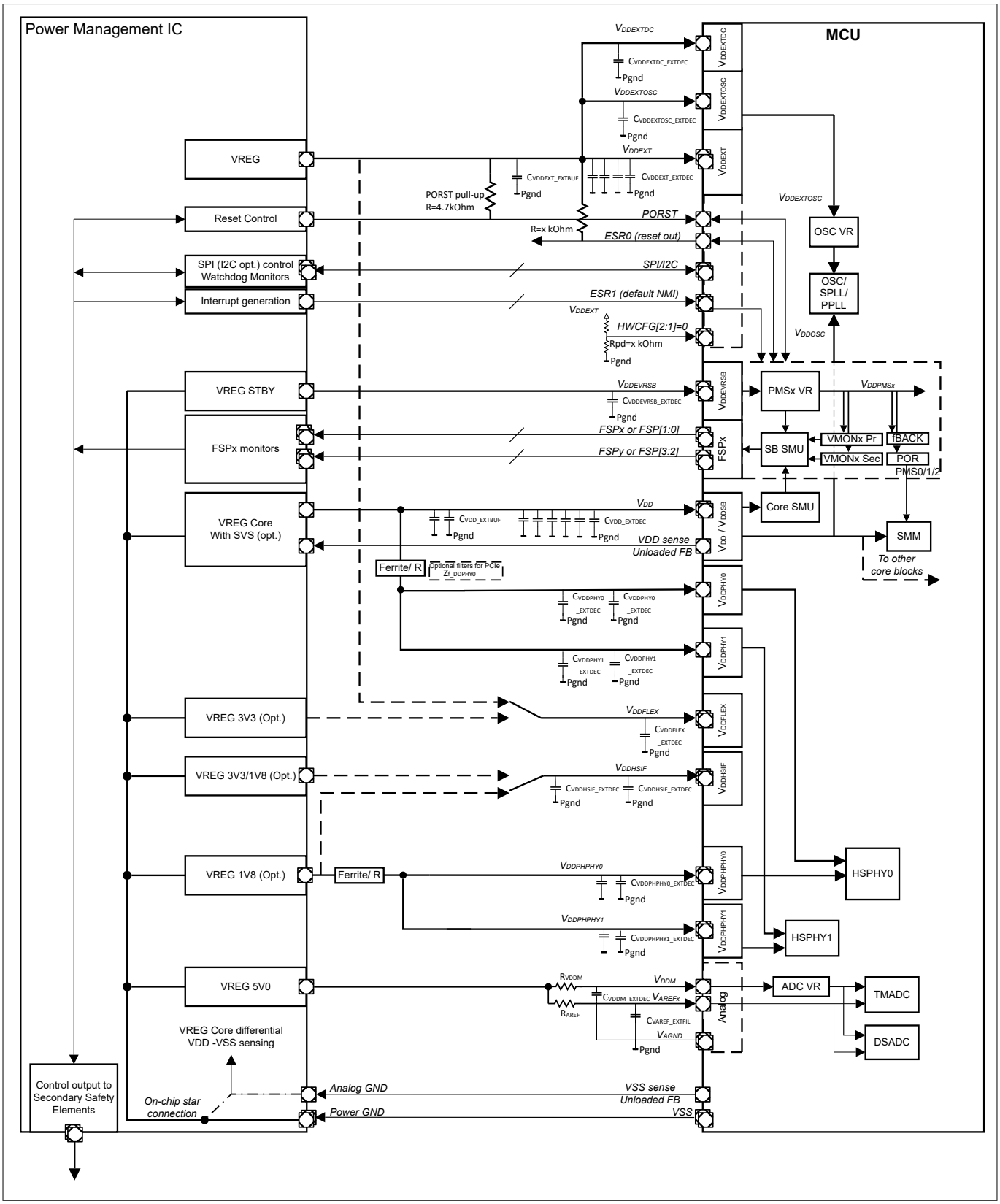


Figure 790 External regulator interface and capacitors. All features may not be available in all variants.

The previous figure illustrates an abstract of the connectivity scheme between the MCU and the external regulator (or power management IC), including the supply rails and control interfaces. While some of them are optional, the majority of the interfaces are mandatory for MCU operation.

Mandatory supply rails:

- V_{DDEXT}
- $V_{DDEXTDC}$ supply for the EVRC regulator. $V_{DDEXTDC}$ shall be supplied with the same voltage as V_{DDEXT} if the V_{DD} voltage is supplied externally
- $V_{DDEVRSB}$ supply for the PMS subsystem and $V_{DDEVRSB}$-buffered pads. Powered from an individual regulator, to provide availability of the PMS0/PMS1/PMS2 domains during the cold reset of the other power domains. It usually provides support for the low-power standby function. Optionally it may be powered from the same regulator as the V_{DDEXT} supply rail
- V_{DDFLEX} (if available on the respective product variant), may be supplied from an individual regulator, or connected to the same regulator as the V_{DDEXT} supply rail
- V_{DD}/V_{DDSB} supply for the core domain (if available on the respective product variant). Optionally, the responsible regulator supports static voltage scaling to leverage the production characteristics of an individual chip for power consumption improvement. The unloaded V_{DD} and unloaded *GND* pins may be used by the V_{DD} regulator for direct feedback connection, to compensate for the static voltage drop of the PCB and device package parasitics
- V_{DDHSIF} (if available on the respective product variant), may be supplied from an individual regulator, or connected to the same regulator as the V_{DDEXT}, V_{DDFLEX}, or $V_{DDPHPHYx}$ supply rail
- V_{DDM}/V_{AREFx} for the analog domain. V_{AREFx} is used as the reference supply inputs for the TMADC and DSADC

Optional supply rails:

- $V_{DDPHPHYx}$/V_{DDPHYx} are related to the SGMII and PCIe PHY interfaces (if available on the respective product variant). Optional external filter elements may be used for each interface instance (x = 0, 1, 2)

Mandatory control links:

- *PORST* bi-directional open-drain control interface. Input: Warm reset request from external reset element. Output: Cold reset signal generated internally by the primary voltage monitors (VMONP). It must be pulled up to the same supply as V_{DDEXT}
- *HWCFG[2:1]* input, deciding the EVRC usage and its configuration

Optional control links:

- *ESR0*: Reset elongation output. Used by default as the reset control from the MCU to other external ICs. Optionally used as NMI/trap generation input
- *ESR1/2*: Trap inputs. Optionally used as reset elongation outputs or system/application reset request inputs
- *ESR0/1/2*: May be alternatively configured for the module group reset (MGRx) request input
- *ESR0/1/2*: May be alternatively configured as GPIOs
- *ESR2*: May be alternatively configured as standby wake-up trigger
- FSPx: Fault signal protocol outputs. Used as an interface controlled from the standby SMU to signal an internal alarm occurrence to the external supervisory elements. Depending on the chosen FSP protocol, the *FSP0/1/2/3* outputs may be used individually or in groups with various alarm mapping. It is decided at the system level what kinds of reactions (for example cold or warm reset) each of the FSPx mapped alarms are expected to trigger
- SPI/I2C: Standard interface to the external PMIC, used for control functions as well as for safety service functions such as external watchdogs

Static voltage scaling

Static voltage scaling is the procedure of adapting the core voltage to the required nominal level based on the characteristics of the particular MCU device sample. TC4xx devices are designed to support static voltage scaling, leveraging production information of each device sample in order to achieve a better power to performance ratio.

On the TC4xx product variants supporting this feature, the voltage scaling procedure differs depending on whether the V_{DD} voltage is supplied by an external core voltage regulator or by the embedded EVRC and is illustrated in the following.

Static V_{DD} voltage scaling with external core voltage generation

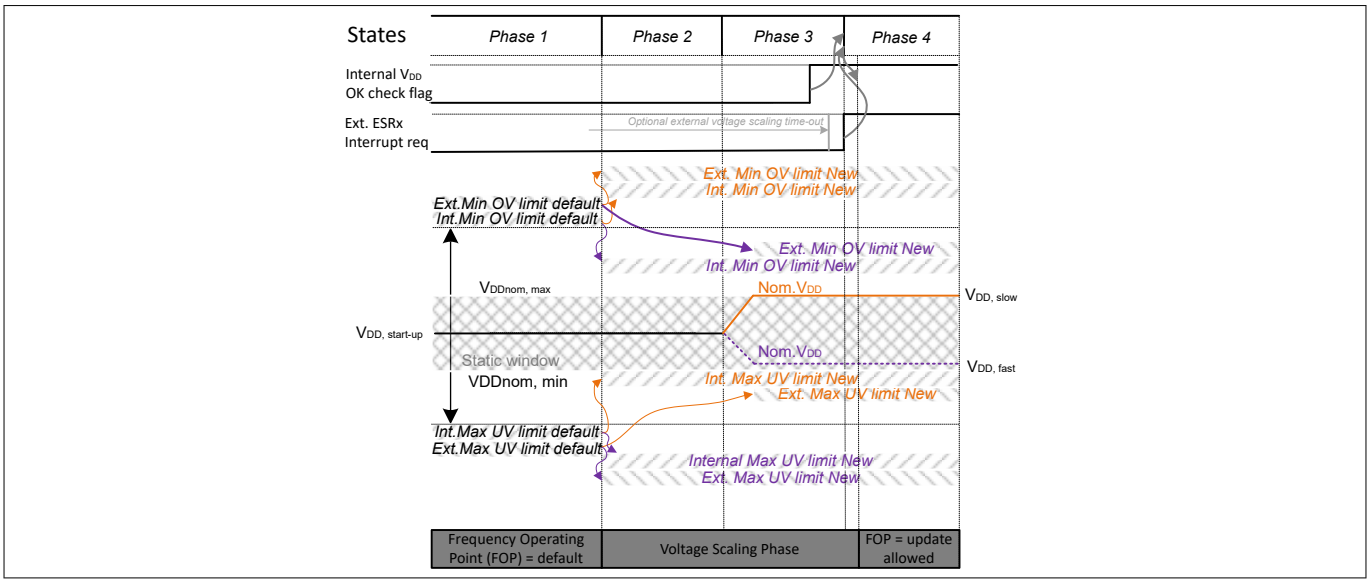


Figure 791 Static V_{DD} voltage scaling with external core voltage generation

Phase 1 (internal monitoring threshold adaptation and voltage scaling trigger):

- Read-out of the stored device type and corresponding nominal voltage level (voltage ID) in the VID_CON0 register
- Disable the V_{DD} primary under-voltage reset in the VMONP_VDDRST register, primary OV/UV alarms in the VMONP_VDDCON register, and the secondary OV/UV alarms in the VMONS_VDDCON register. Optionally only the OV or UV alarms can be disabled individually, depending on the direction of the required V_{DD} voltage change
- Adapt the under-voltage reset (in the VMONP_VDDRST.RESETTRIM bit-field), as well as the OV/UV alarm thresholds to the new voltage set-point
- Send the command to the external core voltage regulator for the new V_{DD} voltage set-point. It is expected that the interface to the external regulator is already initialized

Phase 2 (external monitoring threshold adaptation):

- Depending on the system safety integrity requirements, the external voltage monitoring limits (of the external V_{DD} regulator) may need to be adapted to the new voltage set-point. The exact procedure depends on the respective implementation of this feature in the external power management IC. The example flow illustrated in the previous figure shows the change of the OV and UV limits done in an order that depends on the direction of the required voltage change. For instance, a new set-point greater than the default V_{DD} voltage results in an OV threshold change before the V_{DD} voltage change, while the UV threshold can be changed after raising the V_{DD} voltage

Phase 3 (voltage scaling active phase):

- The external V_{DD} regulator reaches the new desired voltage set-point
- If under-voltage reset and voltage monitor alarms are not disabled, it is recommended to minimize the load dynamics during the phases 2 and 3 by reducing the MCU utilization to a minimum acceptable by the application in order to prevent under-voltage resets and OV/UV alarms until the voltage reaches the target nominal value

Phase 4 (voltage scaling completed and re-enabling of alarms):

- The software can read the new V_{DD} voltage value from the status bits of the primary or secondary voltage monitors. The software can also optionally wait for an interrupt from the external core voltage regulator as soon as the voltage scaling active phase is completed

- If no confirmation of the voltage scaling status is received within a user defined timeout, the software may repeat the flow or trigger an error handler. If a successful voltage scaling status is achieved, the software may now re-enable the under-voltage reset and the voltage monitor alarms
- After successful completion of the above steps, the operation can continue with the new frequency operation point

Static V_{DD} voltage scaling with EVRC

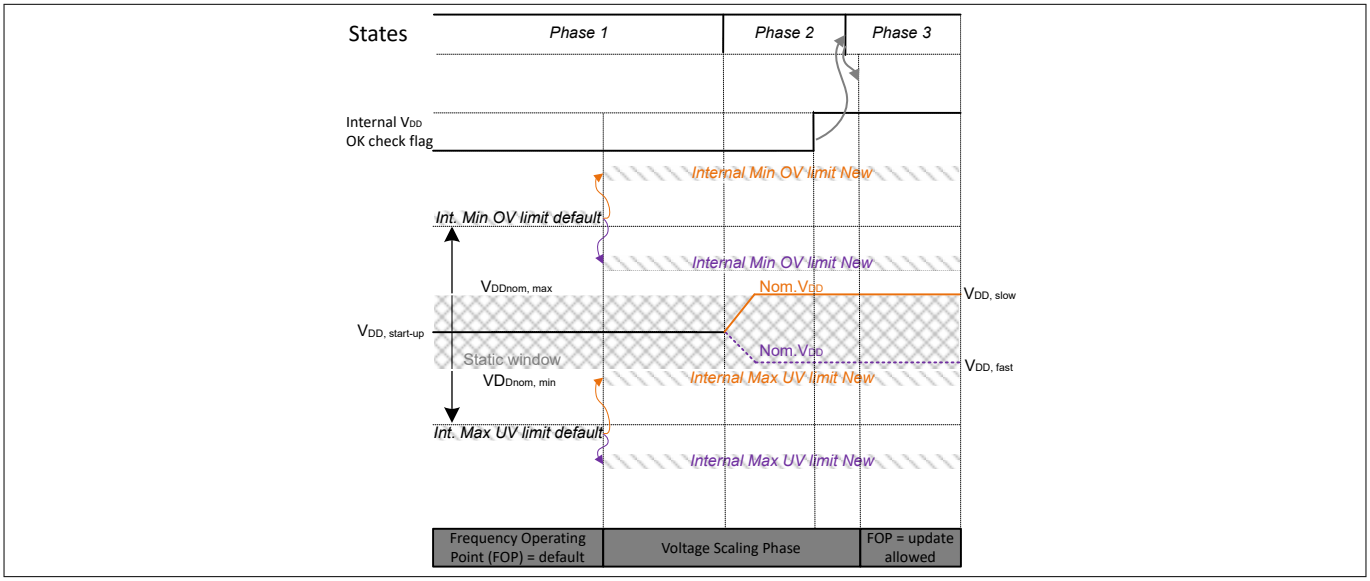


Figure 792 Static V_{DD} voltage scaling with EVRC

Phase 1 (internal monitoring threshold adaptation and voltage scaling trigger):

- Read-out of the stored device type and corresponding nominal voltage level (voltage ID) in the VID_CON0 register
- Disable the V_{DD} primary under-voltage reset in the VMONP_VDDRST register, primary OV/UV alarms in the VMONP_VDDCON register, and the secondary OV/UV alarms through the VMONS_VDDCON register. Optionally only the OV or UV alarms can be disabled individually, depending on the direction of the required V_{DD} voltage change
- Adapt the under-voltage reset (in the VMONP_VDDRST.RESETTRIM bit-field), as well as the OV/UV alarm thresholds to the new voltage set-point

Phase 2 (voltage scaling active phase):

- The voltage scaling active phase is started as soon as the software changes the default EVRC set-point through the EVRC_CON0.SDVOUTSEL bit-field
- If under-voltage reset and voltage monitor alarms are not disabled, it is recommended to minimize the load dynamics during the phases 2 and 3 by reducing the MCU utilization to a minimum acceptable by the application, in order to prevent OV/UV alarms until the voltage reaches the target nominal value

Phase 3 (voltage scaling completed and re-enabling of alarms)

- The software can poll the EVRC_STAT0.SDVOK bit which indicates when the new regulator set-point has been reached
- If no confirmation of the voltage scaling status is received within a user defined timeout, the software may repeat the flow or trigger an error handler. If a successful voltage scaling status is achieved, the software may now re-enable the under-voltage reset and the voltage monitor alarms
- After successful completion of the above steps, the operation can continue with the new frequency operation point

Related information

Output voltage setting and voltage scaling on page 5985

36.3.5 Voltage monitoring and safety

The following section describes the secondary voltage monitors, other voltage monitoring functions, and related safety aspects.

Related information

Primary voltage monitors VMONPx and cold PORST on page 5971

36.3.5.1 Secondary over- and under-voltage monitors (VMONSx)

Secondary over-voltage and under-voltage monitoring against programmable thresholds is provided for the following supply rails: $V_{DDEVRSB}$, V_{DDEXT}, $V_{DDP3NVM}$, V_{DDM}, V_{DDFLEX}, V_{DD}, V_{DDPMS0}, V_{DDPMS1}, V_{DDPMS2}, $V_{DDSBRAM}$, V_{DDPAD}, $V_{DDPHPHYx}$, $V_{DDPHRIFx}$, V_{DDHSIF}, V_{DDPHYx}, V_{DDRIFx}, V_{DDCPUx}, V_{DDPPU}, V_{DDLMUx}, and $V_{DDEXTDC}$, depending on product variant (not all supply rails are available on each product variant). The secondary monitors are based on a secondary band-gap reference independent from the primary band-gap reference used by the embedded PMS voltage regulators. In case of a threshold violation, an SMU alarm event is generated (as configured by the SW) and an interrupt is triggered (if enabled by the SW). The tolerance of the threshold for each monitored supply rail is documented in the data sheet as V_{XXMON} parameter.

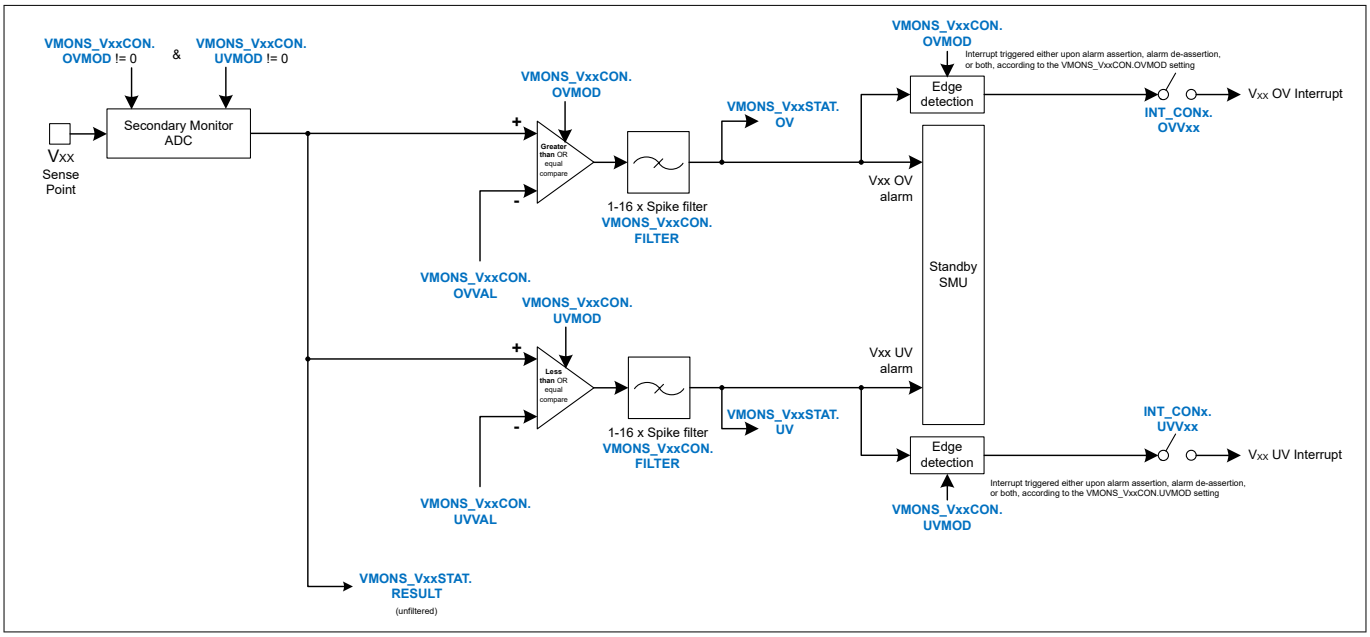


Figure 793 Secondary voltage monitors configuration and outputs

Refer to the functional description for the specific product variant for an image showing an overview of the secondary voltage monitors.

The appropriate thresholds for voltage monitoring can be programmed in the VMONS_VxxCON registers. The over-voltage and under-voltage threshold settings in the OVVAL and UVVAL bit-fields can be calculated by linear interpolation from the multiple voltage levels and corresponding thresholds provided in the V_{XXMON} parameters in the data sheet. The secondary monitor violation is notified through an alarm, for over-voltage if OVMOD ≠ 00_B, respectively for under-voltage if UVMOD ≠ 00_B, independent of the threshold crossing direction. In addition to the SMU alarm, an interrupt is triggered if the corresponding INT_CONx.OVVxx respectively INT_CONx.UVVxx bit is enabled, either upon alarm assertion, alarm de-assertion, or both, as programmed in the UVMOD and OVMOD bit-fields (00_B = interrupt disabled, 01_B = interrupt triggered upon alarm assertion, 10_B = interrupt triggered when the alarm is de-asserted, and 11_B = interrupt triggered either when the alarm is asserted or de-asserted). In case of a violation, the respective status flags (OV or UV) are set in the corresponding VMONS_VxxSTAT registers. These status bits may be evaluated to differentiate which supply rail has triggered the alarm event to the SMU. The VMONS_VxxSTAT UV and OV flags reflect the instantaneous status

of the under-voltage or over-voltage event. They are not latched within the PMS registers, instead they are latched in the standby SMU registers.

In case both OV and UV thresholds are configured to the same value (or within a distance lower than the accuracy of the voltage monitors), then both OV and UV alarms could be triggered at the same time.

A configurable spike filtering function is applied on the consecutive ADC results, and can be individually configured in the VMONS_VxxCON.FILTER bit-fields. A sequence of at least VMONS_VxxCON.FILTER+1 consecutive threshold violations is then required to trigger the OV/UV event. This filter is applied for generating the SMU alarms and for setting the OV/UV status bits in the VMONS-VxxSTAT registers.

The supply rails connected to the same secondary monitor (VMONS0 or VMONS1) are converted one after another in a continuous scan mode. Please refer to the figure, 'Overview of the secondary voltage monitors' in the device specific section of the PMS chapter to see which supply rails are connected to which secondary monitor. The monitoring latency of each supply rail depends on the spike filter setting, the supply rail conversion latency and the number of supply channels being converted by each secondary monitor as given below,

- The latency for $V_{DDEXTDC}$ rail is t_{TRKHV}*(FILTER+1), where t_{TRKHV} is datasheet parameter and FILTER refers to the spike filter setting
- For the voltage rails connected to VMONS0 or VMONS1 the latency is given by the following formula:

  T_{VMONSx}=(FILTER+1)*((A*t_{SARHV})+(B*$t_{SARCORE}$))

  In the formula above t_{SARHV} and $t_{SARCORE}$ are datasheet parameters and FILTER refers to the spike filter setting and x is the index of the VMONSx. A is the number of voltage rails monitored by VMONSx that have an individual conversion latency t_{SARHV} and B is the number of voltage rails monitored by VMONSx that have an individual conversion latency of $t_{SARCORE}$. Please refer to the table “Secondary voltage monitor parameters” below to find out which conversion latency applies for each voltage rail. Please note that A and B values depend on the number of channels being actually monitored by a specific VMONSx as monitoring of a supply rail can be selectively activated or deactivated by the user

***Note**: VMONS1 is not available for all derivatives.*

As shown in the figure below, the $V_{DDEXTDC}$ voltage is monitored using a tracking ADC (called VMONP3), while all other voltages are monitored by SAR ADCs. Therefore, there is a difference in the configuration of the secondary monitor for $V_{DDEXTDC}$ vs. all other monitored voltages, as the tracking ADCs have a low-pass filter in the alarm triggering path. The alarm low-pass filter for the $V_{DDEXTDC}$ secondary monitor tracking ADC is configured in the VMONP_TRKTRIM0.VDDEXTDCVMONSLPF bit-field.

The VMONS_LVLSEL register bit-fields shall not be modified by the application software (as they are not related to the secondary monitoring thresholds, but are instead used by other modules of the MCU, to differentiate between the different possible voltage levels used by applications on the respective supply rails). The VMONS_LVLSEL.NVMMINSEL bit-field indicates the threshold for stable NVM operation, while the VMONS_LVLSTAT.NVMMIN status bit indicates the VMONS evaluation of V_{DDEXT} voltage against this threshold, using an internal spike filter of 2 samples.

The last value measured by the secondary monitoring ADCs is updated, for each supply rail, in the VMONS_VxxSTAT.RESULT bit-field, when the respective channel is converted. Conversions occur on a secondary ADC channel if any of the following conditions is true:

- The supply rail is on a tracking ADC (that is $V_{DDEXTDC}$) - the tracking ADC channels are always converted
- For the V_{DDEXT} and V_{DDM} rails the channels are always converted
- If UVMOD or OVMOD is not 0
- For the V_{DDPMS0} or V_{DDPMS2} rail if drift monitoring is enabled (DRMON_Vxx.ENABLE = 1)
- If UVMOD and OVMOD have just been set to zero but the conversion already started

The VMONS_VxxSTAT.RESULT value is stored without spike filtering and shows the current value measured by the ADC.

The VMONS_VxxSTAT.ACTIVITY bit-field stores an activity counter, which serves as a safety check that the respective secondary monitor is currently running. This counter cumulatively counts the end of conversion signals per secondary monitor VMONSx, in a background scan over all channels of a VMONSx monitor and respective filter configurations of each channel. The total number of conversions is ConvTot = Σ[ChX * (VxxCON.FILTER+1)]. The counter is reset to zero on a ConvTot overflow.

To configure the VMONS_VxxCON.UVVAL and VMONS_VxxCON.OVVAL values, the LSB value for each voltage rail is provided in the following table:

Table 1431 Secondary voltage monitors parameters

Voltage rail	VMONS LSB datasheet parameter	Secondary voltage monitor accuracy after trimming - datasheet parameter	Effects of secondary voltage monitoring thresholds violation[1)]
$V_{DDEVRSB}$	$V_{SARHVLSB}$	$V_{DDEVRSBMON}$	UV, OV alarm
V_{DDEXT}	$V_{SARHVLSB}$	$V_{DDEXTMON}$	UV, OV alarm
$V_{DDP3NVM}$	$V_{SARHVLSB}$	$V_{DDP3NVMMON}$	UV, OV alarm
V_{DDM}	$V_{SARHVLSB}$	V_{DDMMON}	UV, OV alarm
V_{DDFLEX}	$V_{SARHVLSB}$	$V_{DDFLEXMON}$	UV, OV alarm
V_{DD}	$V_{SARCORELSB}$	V_{DDMON}	UV, OV alarm
V_{DDPMS0}	$V_{SARCORELSB}$	$V_{DDPMS0MON}$	UV, OV alarm
$V_{DDPMS1,2}$	$V_{SARCORELSB}$	$V_{DDPMS2MON}$	UV, OV alarm
$V_{DDSBRAM}$	$V_{SARCORELSB}$	$V_{DDPMS0MON}$	UV, OV alarm
V_{DDPAD}	$V_{SARCORELSB}$	$V_{DDPADMON}$	UV, OV alarm
$V_{DDPHPHYx}$	$V_{SARHVLSB}$	$V_{DDPHPHYxMON}$	UV, OV alarm
$V_{DDPHRIFx}$	$V_{SARHVLSB}$	$V_{DDPHRIFxMON}$	UV, OV alarm
V_{DDHSIF}	$V_{SARHVLSB}$	$V_{DDHSIFMON}$	UV, OV alarm
V_{DDPHYx}	$V_{SARCORELSB}$	$V_{DDPHYxMON}$	UV, OV alarm
V_{DDRIFx}	$V_{SARCORELSB}$	$V_{DDRIFxMON}$	UV, OV alarm
V_{DDCPUx}	$V_{SARCORELSB}$	$V_{DDCPUxMON}$	UV, OV alarm
V_{DDPPU}	$V_{SARCORELSB}$	$V_{DDPPUMON}$	UV, OV alarm
V_{DDLMUx}	$V_{SARCORELSB}$	$V_{DDLMUMON}$	UV, OV alarm
$V_{DDEXTDC}$	$V_{TRKHVLSB}$	$V_{DDEXTDCMON}$	UV, OV alarm

1) Only if the corresponding VMONS_VxxCON.UVMOD, respectively VMONS_VxxCON.OVMOD, are not zero.

In case of over-voltage supply alarms, it may be ensured that the supply to the device is switched off to avoid damage. The Error Pin Fail Safe Protocol (*FSP*x), triggered by the standby SMU (if enabled in the standby SMU CMDSTDBY register bit-fields), ensures that the over-voltage condition is communicated to the external safety supply even when the MCU is in warm reset state.

After start-up, it may happen that supply over- or under-voltage alarms may already have been triggered depending on residual start-up voltages or supply dynamics. Likewise during VMONS_VxxCON reconfigurations, spurious alarms may be raised depending on filter state and changed configuration. Therefore, before activating SMU alarm generation or triggering latent fault supply alarm tests, the secondary monitors and filters need to be completely reset. It needs to be ensured that SMU alarms and associated interrupts are foremost deactivated in VMONS_VxxCON and INT_CONx registers. Alarms and interrupts are then consequently

re-configured to the intended voltage level and filter settings in VMONS_VxxCON. The spike filters for over-voltage and under-voltage alarms are automatically reset whenever writing into the VMONS_VxxCON.OVVAL, respectively into VMONS_VxxCON.UVVAL bit-fields, therefore there is no need to explicitly de-activate and re-activate the filters. A delay time of 4 µs has to be awaited by the application SW before alarm activation after configuration is changed in VMONS_VxxCON registers.

In case of application and system resets, PMS alarms occurring during the respective reset shutdown and release will be reflected in SMU_stdby AGX alarm status registers and consequently SMU_stdby FSP reaction may be triggered, if configured in the SMU_stdby AGFSP.FEx registers. On the contrary, PMS alarms occurring during warm reset phase will not be latched in SMU_core AGX alarm status registers, as these registers are in reset state. Furthermore, alarms which have occurred during the reset phase would not be consequently forwarded to the SMU_core on reset release.

Related information

Drift monitoring on page 6007

TC4Dx secondary over- and under-voltage monitors on page 6151

36.3.5.2 Peak voltage detection

The peak voltage detection feature performs a continuous recording of the absolute maximum voltage on the V_{DD}, V_{DDEXT}, and $V_{DDEXTDC}$ supply rails across a driving cycle. SW can read the recorded value and reset the corresponding register to restart the measurement. This can be used for reliability and lifetime profile checks.

The measurement is performed by the primary voltage monitoring (tracking) ADC of the respective rail. Therefore, only voltage peaks with transients within the maximum slope specified in the data sheet for the primary voltage monitors are captured.

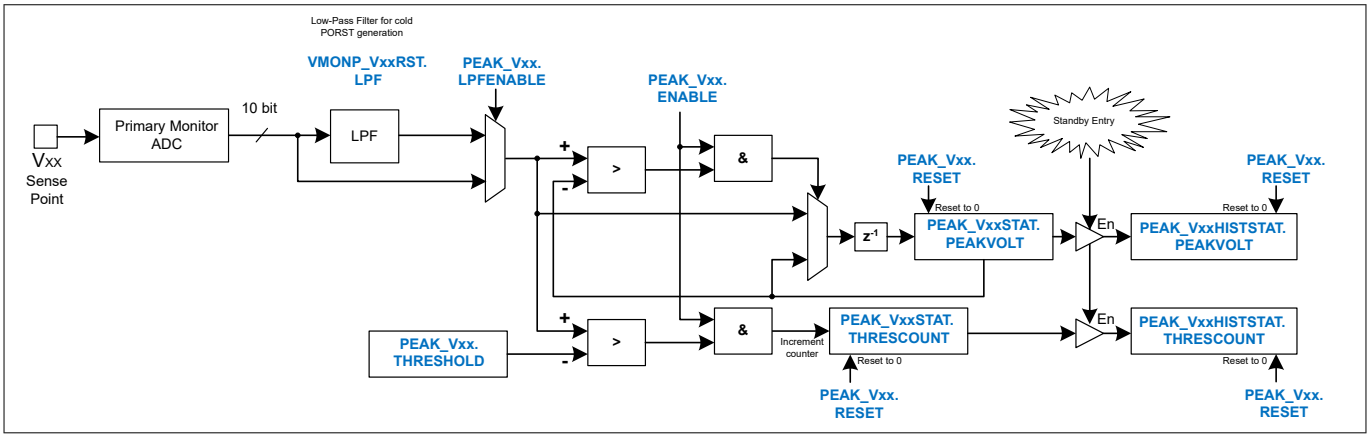


Figure 794 Overview of the peak voltage detection feature

The feature can be enabled by setting the PEAK_Vxx.ENABLE bit, for the respective supply rail (as mentioned before, the feature is supported for the V_{DD}, V_{DDEXT}, and $V_{DDEXTDC}$ supply rails). A low-pass filter can be enabled by setting the PEAK_Vxx.LPFENABLE bit. When this bit is set, the same LPF filter as used by the primary voltage monitors for cold PORST generation (which is configured in the VMONP_VxxRST.LPF bit-field) is applied also to the peak voltage detection.

The maximum absolute voltage value detected by the ADC is updated in the PEAK_VxxSTAT.PEAKVOLT bit-field, whenever a new maximum value has been measured. In addition, a configurable threshold can be defined by the application SW in the PEAK_Vxx.THRESHOLD bit-field. All measured voltage samples larger than this threshold are then counted in the PEAK_VxxSTAT.THRESCOUNT bit-field.

In order to retain the measurement through a standby transition, a set of history status registers are defined in the PMS1 domain, named PEAK_VxxHISTSTAT (their content is kept in STANDBY1 mode, but not in STANDBY0 mode). These registers copy the content of the PEAK_VxxSTAT registers upon a standby entry and are available for reading by SW after the wake-up.

SW can restart the measurement at any time by setting the PEAK_Vxx.RESET bit, which resets the PEAK_VxxSTAT and PEAK_VxxHISTSTAT registers to their respective reset values.

If the peak voltage detection is disabled after running a measurement, by setting PEAK_Vxx.ENABLE=0, the corresponding PEAK_VxxSTAT registers are not updated anymore with new measurements, that is the PEAKVOLT and THRESCOUNT bit-fields keep their current values until the next LVD reset or standby transition (as PEAK_VxxSTAT registers are being reset in standby mode or at an LVDPMS2 reset).

The PEAK_VxxHISTSTAT registers are updated with the content of the respective PEAK_VxxSTAT registers at every standby entry, independent of the PEAK_VxxENABLE setting. The PEAK_VxxHISTSTAT registers are being reset in STANDBY0 mode or at an LVDPMS1 reset.

36.3.5.3 Drift monitoring

The drift monitoring feature targets the detection of a slow drift of the voltage levels caused by for example reliability, aging, or other effects with slow impact over the lifetime of an MCU device. It includes monitoring of the V_{DD}, V_{DDEXT}, $V_{DDEXTDC}$, V_{DDM}, V_{DDPMS0}, and V_{DDPMS2} supply rails.

The measurement is carried out by the primary monitoring (tracking) ADCs for V_{DD}, V_{DDEXT}, and $V_{DDEXTDC}$, respectively by the secondary monitoring (SAR) ADC for V_{DDM}, V_{DDPMS0}, and V_{DDPMS2}.

The evaluation of a potential drift of the voltage level must be done by the application SW. The MCU HW provides only the filtered value of the voltage level for the measured supply rails, accumulated across a driving cycle (that is since the last reset or standby event) using a filter function as described in the following.

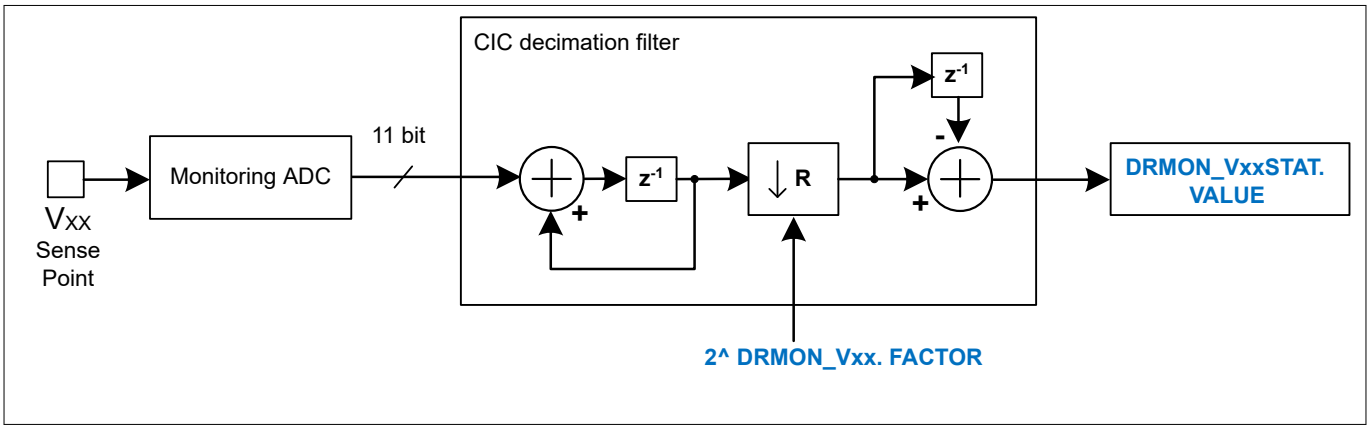


Figure 795 Overview of the hardware feature for drift monitoring

The feature can be enabled by setting the DRMON_Vxx.ENABLE bit, for the respective supply rail.

A single-stage cascaded-integrator-comb (CIC) filter is implemented, as shown in the previous figure. The CIC filter factor R can be configured in the DRMON_Vxx.FACTOR bit-field (R = 2^FACTOR). Following the CIC filter, a median filter is appended, which outputs the filtered value stored in the DRMON_VxxSTAT.VALUE bit-field.

The application SW can read at any time the filtered value of a monitored rail in the respective DRMON_VxxSTAT register, it can decide if it deviates from the nominal value by an unacceptable amount, and can then take appropriate actions.

A re-configuration of the DRMON_Vxx.FACTOR bit-field during a running measurement invalidates and resets the VALUE result in the corresponding DRMON_VxxSTAT register. If the ENABLE bit is kept asserted, the measurement restarts with the new FACTOR setting. To just restart the measurement without changing the FACTOR setting, the SW can re-write the same FACTOR setting in the DRMON_Vxx register. A write access into the DRMON_Vxx.FACTOR bit-field triggers a restart of the measurement cycle.

When the measurement is disabled, by clearing the DRMON_Vxx.ENABLE bit, the corresponding DRMON_VxxSTAT.VALUE bit-field keeps its result until a new measurement is started or until the next reset or standby.

During MONBIST, the drift monitoring results in the DRMON_VxxSTAT.VALUE bit-fields are not being updated.

36.3.5.4 Power built-in self-test (PBIST1/2)

A power built-in self-test (PBIST) at start-up allows the testing of supply levels (for over-voltage and under-voltage conditions), power functions, and voltage monitors before cold PORST reset release. This test is distributed into two phases, one executed before EVRC start-up (PBIST1) and the second one executed afterwards.

The first phase of the test, PBIST1, targets the checks for EVRC operation. In this phase, the V_{DDEXT} and $V_{DDEVRSB}$ voltage rails are tested for over-voltage conditions, and the $V_{DDEXTDC}$, V_{DDPMS0}, V_{DDPMS1}, and V_{DDPMS2} voltage rails are tested for over-voltage and under-voltage conditions. Based on the configuration of the *HWCFG[2:1]* pins, EVRC is started only after the PBIST1 phase is successfully completed.

The PBIST2 phase occurs after EVRC is started (if activated) and after it reaches the Voltage OK state (indicated by the EVRC_STAT0.SDVOK bit). Also, this test is only executed after all voltage rails reach the reset release threshold (including reset hysteresis). The voltage rails tested during PBIST2 and the voltage monitoring ADCs used in the tests are indicated in the following table.

The following table provides an overview of the voltage rails and over-voltage/under-voltage tests executed during PBIST1 and PBIST2, if the respective voltage rails are available on the considered product variant:

Table 1432 PBIST overview

Voltage rail	ADC used	PBIST1		PBIST2	
		Over-voltage test	Under-voltage test	Over-voltage test	Under-voltage test
V_{DDEXT}	VMONP	Yes	Yes	Yes	Yes
$V_{DDEVRSB}$	VMONP	Yes	Yes	Yes	Yes
$V_{DDP3NVM}$	VMONP	Yes	Yes	Yes	Yes
$V_{DDEXTDC}$	VMONP	Yes	Yes	Yes	Yes
V_{DDDCLS}	VMONP	No	No	Yes[1)]	Yes[1)]
$V_{DDEXTDC}$-V_{SSDCHS}	VMONP	No	No	Yes[1)]	Yes[1)]
V_{DD}	VMONP	No	No	Yes	Yes
V_{DDM}	VMONP	No	No	Yes	No
V_{DDPMS0}	VMONS	Yes	Yes	Yes	Yes
V_{DDPMS1}	VMONS	Yes	Yes	Yes	Yes
V_{DDPMS2}	VMONS	Yes	Yes	Yes	Yes

1) Voltage rail tested in PBIST only if *HWCFG[2:1]*=11_B

For all voltage rails documented in the table above, used for PBIST1/2 testing, the reset values of the VMONP_VxxSTAT.OV and VMONP_VxxSTAT.UV status flags (if VMONP ADC is used for PBIST testing), respectively of the the VMONS_VxxSTAT.OV, and VMONS_VxxSTAT.UV status flags (if VMONS ADC is used for PBIST testing) are set to "false positive" before running the PBIST. After the PBIST runs, the flags are cleared if the voltages are within the limits configured in VMONP_VxxCON.OVVAL/UVVAL, respectively VMONS_VxxCON.OVVAL/UVVAL. If PBIST1 or PBIST2 fails, the MCU does not release the cold PORST.

36.3.5.5 Standby SMU integration and monitor BIST (MONBIST)

MONBIST is a built-in self test for the functionality of secondary voltage monitors and for the alarm generation path and can be triggered by user software after reset release. This secondary monitor BIST ensures a high

latent fault coverage for the secondary monitors and for the associated alarm and error pin fault logic, which is routed to the standby SMU.

The MONBIST can be triggered during start-up by setting the MONBISTCTRL.TSTEN register bit in the Standby SMU module. During the ongoing MONBIST, the test of the PMS safety flip-flops (SFF) shall not be triggered. MONBIST tests only the monitors and alarms for which the OVMOD/UVMOD bit-fields have been enabled by the application software prior to triggering the test. MONBIST test can take up to 30 µs execution time. The procedure is as follows:

- User SW configuration steps before MONBIST execution:
  - The user software shall enable the standby SMU through the CMDSTDBY.SMUEN register bit (of the Standby SMU module), to enable MONBIST functionality
  - The (standby SMU) MONBISTCTRL.TSTCLR bit shall be set foremost to clear all the flags and reset the test logic. This clears the MONBISTCTRL.TSTEN bit, as well as the TSTRUN, TSTDONE, TSTOK, SMUERRx, and PMSERR bits in the (standby SMU) MONBISTSTAT register
  - The STANDBY_CON0.VDDEXTSBEN and STANDBY_CON1.VDDSBEN bits shall be set to "disabled", to avoid triggering a standby entry when the MONBIST test simulates the occurrence of an under-voltage condition on the V_{DDEXT} and V_{DD} supply rails
  - The user software must clear all spike filters of the secondary voltage monitors before running MONBIST, by setting all VMONS_VxxCON.FILTER = 0
  - VMONS_VxxCON.OVMOD is set to 01_B and VMONS_VxxCON.UVMOD is set to 10_B to activate under-voltage and over-voltage alarms
  - The corresponding over-voltage and under-voltage interrupts are disabled by clearing the INT_CONx.OVVxx and INT_CONx.UVVxx register bit-fields
  - FSP reactions on alarms are disabled by setting AGSTDBYi_FSPj.FEx to 0 (in the standby SMU)
  - (Standby SMU) CMDSTDBY.FSP0EN, FSP1EN, FSP2EN, and FSP3EN configuration bits are cleared to avoid spurious error pin activation during MONBIST
  - (Standby SMU) CMDSTDBY.ASCE is set to 1 to ensure that all pending alarms are cleared in AGx registers
  - Consequently the MONBIST is enabled through the (standby SMU) MONBISTCTRL.TSTEN register bit
- MONBIST execution in HW:
  - The (standby SMU) MONBISTSTAT.TSTRUN register bit is then set by hardware, to indicate an ongoing test by MONBIST logic
  - Once the test is completed, the hardware sets the MONBISTSTAT.TSTDONE bit and clears the MONBISTSTAT.TSTRUN bit
  - The MONBISTSTAT.TSTOK bit indicates if the test was successfully completed
  - The MONBISTSTAT.SMUERRx (where x = 0,1,2,3) and MONBISTSTAT.PMSERR bits indicate that errors were detected during the MONBIST
- User SW configuration steps after MONBIST execution:
  - The (standby SMU)AGTSTDBY0_STS .FSPxERR (where x = 0,1,2,3) bits shall be cleared by software after MONBIST before enabling FSP reaction. If alarms happened during MONBIST, status registers may be updated and shall be cleared before Standby SMU initialization. MONBISTCTRL.TSTEN bit is cleared at the end of MONBIST
  - After MONBIST execution, the user software must restore the configuration of the spike filters of the secondary voltage monitors (VMONS_VxxCON.FILTER) either to the reset configuration or as required by the application

During MONBIST execution, the drift monitoring results (if enabled) are not being updated.

36.3.6 Interrupts

Interrupts from PMS are triggered only during RUN mode. During STANDBY mode, there are no interrupt triggers but some interrupt events are recognized as wake-up events depending on configuration. Following events may be configured to lead to interrupts routed to the interrupt router (IR) in normal run and sleep system modes. If enabled by the related interrupt enable bit in the registers INT_CONx, an interrupt pulse is generated on one of the service request outputs (SRC_PMS0, SRC_PMS1, SRC_PMS2, SRC_PMS3, SRC_PMS4, SRC_PMS5, SRC_PMS6, or SRC_PMS7). Interrupts are forwarded from PMS to the IR module within 4 f_{SPB} clock cycles (see datasheet parameter) after the occurrence of the event.

Refer to the functional description for the specific product variant for an image showing the interrupt sources and events.

Related information

TC4Dx interrupts on page 6152

36.3.7 Standby mode management

The PMS is partitioned into 3 power domains PMS0, PMS1 and PMS2 to ensure scalable standby modes with lowest standby currents.

Standby domains

The PMS0 power domain is the smallest standby domain with bare minimum functionality required to keep standby RAMs supplied and recognize wake-up from pins, supply ramp-up, and timer modules. In STANDBY0 mode only PMS0 power domain is supplied and the remaining part of the microcontroller is completely switched off. The PMS0 domain contains the V_{DDEXT} wake-up detector, Pin Wake-up unit, the Real Time Clock (RTC), the Wake-Up Timer (WUT) and standby RAM regulator to keep dLMU standby RAMs supplied.

The PMS1 domain offers the next scalable standby domain including the 8-bit standby controller (SCR) with the capability of running on the 100 MHz back-up clock or the 70 kHz internal standby clock. In STANDBY1 mode, both PMS0 and PMS1 power domains are supplied and the remaining part of the microcontroller is completely switched off. Both PMS0 and PMS1 standby power domains are supplied from the $V_{DDEVRSB}$ supply rail which also supplies a part of Port domain (P3x.y). V_{DDEXT} supply may continue to be kept supplied in case wake up from pins supplied by V_{DDEXT} rail is required.

The PMS1 domain supports configurable activation or deactivation of 100 MHz back-up clock and switching to 70 kHz standby clock for the 8-bit standby controller to reduce standby currents. Furthermore, if SCR ADCOMP analog channel conversions are not required, the high precision band-gap and ADC may also be completely powered off to reduce standby currents. The activation and deactivation of the clock, band-gap and ADC modules are managed by SCR software. During transition from STANDBY1 to STANDBY0 mode, the SCR XRAM is switched from V_{DDPMS1} supply to $V_{DDSBRAM}$ supply, in case data retention is enabled (through the STANDBY_CON0.SCRRAMSEL bit-field).

36 Power Management System (PMS)

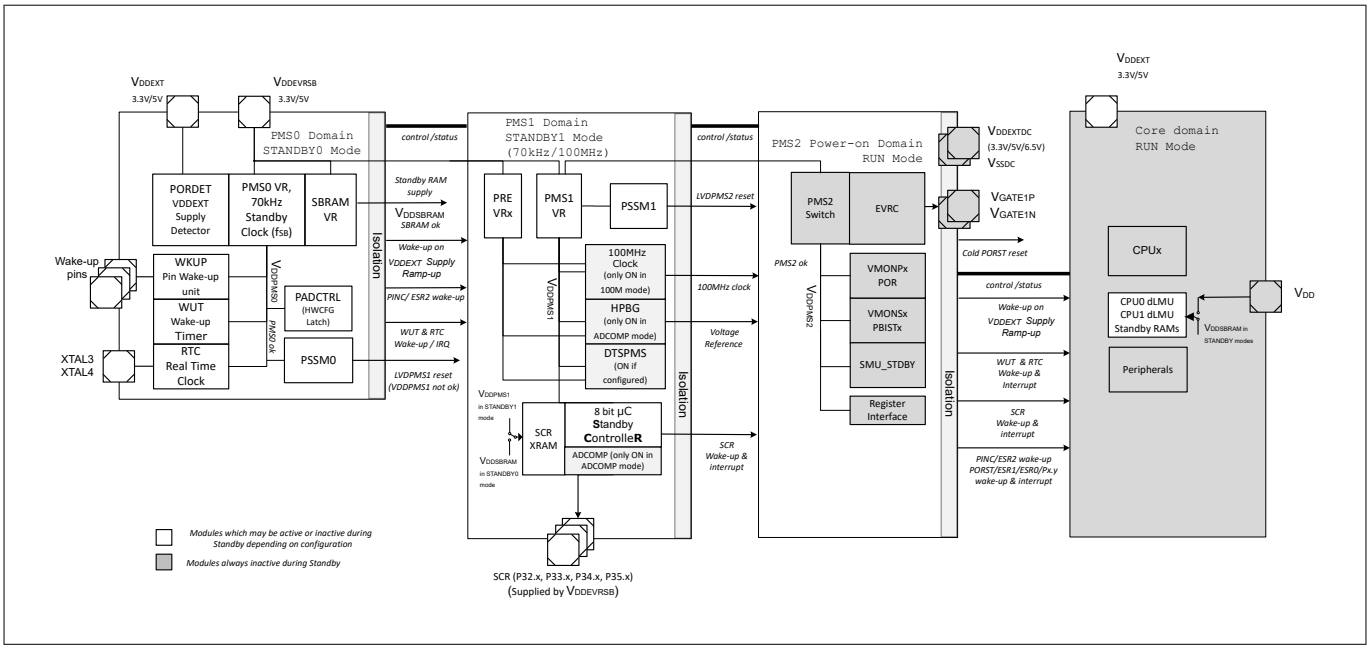


Figure 796 Standby domains

The $V_{DDEVRSB}$ and V_{DDEXT} supply rails may be supplied by a single external supply source or have separate supply sources depending on the application. The $I_{DDEVRSB}$ datasheet parameter is however considering only the current consumption from the $V_{DDEVRSB}$ pins. If single supply source is used (same external regulator for V_{DDEXT} and $V_{DDEVRSB}$), the external regulator would support low quiescent power modes to allow low standby currents at system level. It is also allowed that V_{DDM} and V_{DDFLEX} supply rails are also supplied from the same V_{DDEXT} supply source but at the cost of higher standby currents. As shown in the following figure, only PMS0 and PMS1 domains are kept supplied during standby modes.

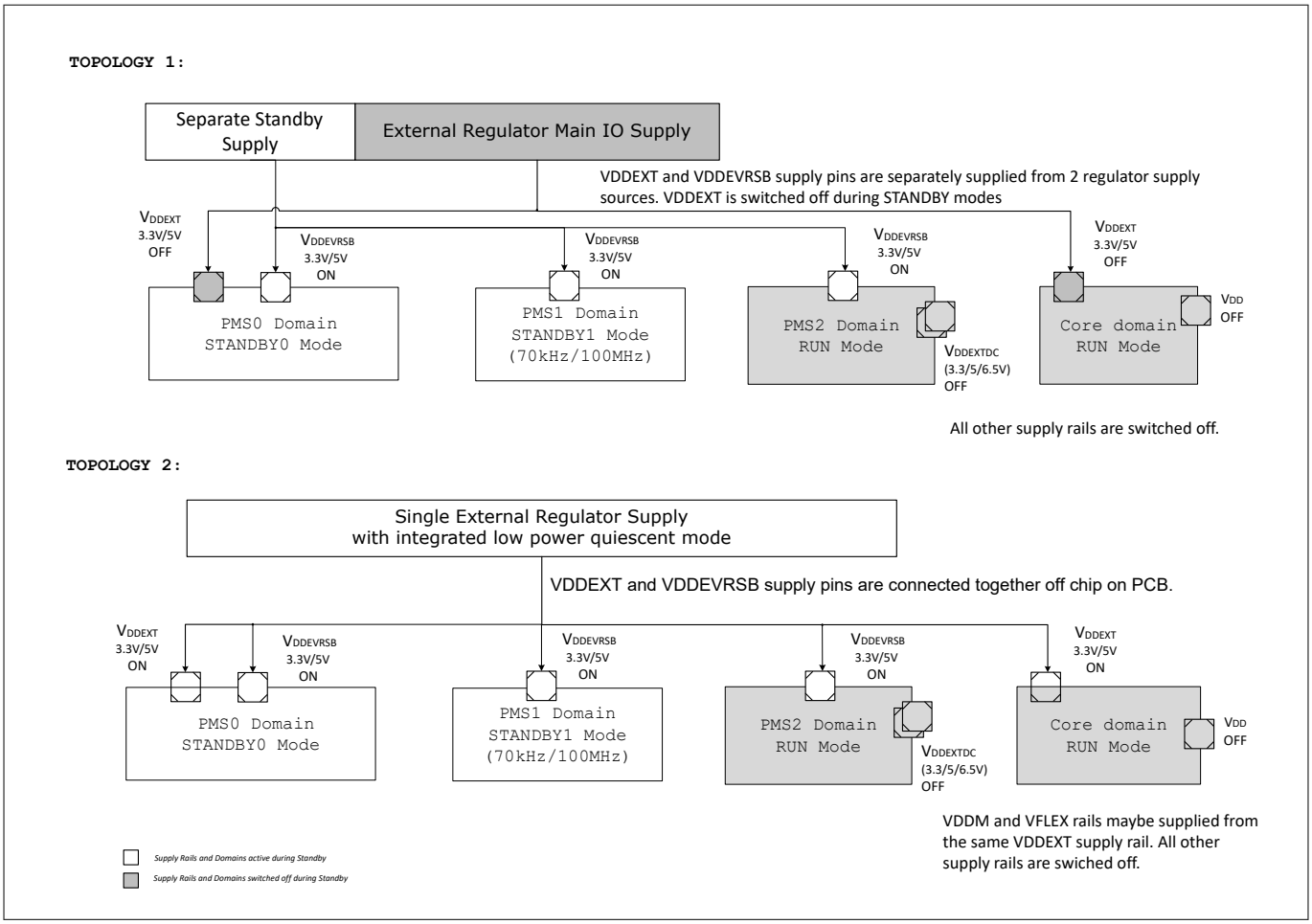


Figure 797 Standby domains and supply topologies

36.3.7.1 Standby modes

Standby modes

Following STANDBY Modes are supported:

- STANDBY0 Mode:

  Only PMS0 domain is supplied. This refers to the Standby state with lowest power consumption running at 70 kHz clock. RTC, WUT, Standby RAM retention, SCR RAM retention and Pin wake-up modules may be active during this state
- STANDBY1 Modes:

  ***Note**: Both PMS0 and PMS1 domains are supplied.*

  1. STANDBY1_70k State: In this state PMS0 and PMS1 domains are both active and running at 70 kHz clock. Standby Controller (SCR) is active during this mode running at 70 kHz clock. Standby Port domain (P32/P33/P34/P35) is additionally active in this state
  2. STANDBY1_100M State: In this state PMS0 and PMS1 domains are both active and PMS1 is running at 100 MHz clock. Standby Controller (SCR) is active during this mode running at up to 100 MHz clock. Standby Port domain (P32/P33/P34/P35) is active in this state. The high precision band-gap and SCR ADCOMP modules are switched off during this state
  3. STANDBY1_ADCOMP State: In this state PMS0 and PMS1 domains are both active and PMS1 is running up to 100 MHz clock. Standby Controller (SCR) is active during this mode running at up to 100 MHz clock. Standby Port domain (P32/P33/P34/P35) is active in this state. In addition the high

precision band-gap and SCR ADCOMP module are active during this state and ADC conversions can be additionally carried out

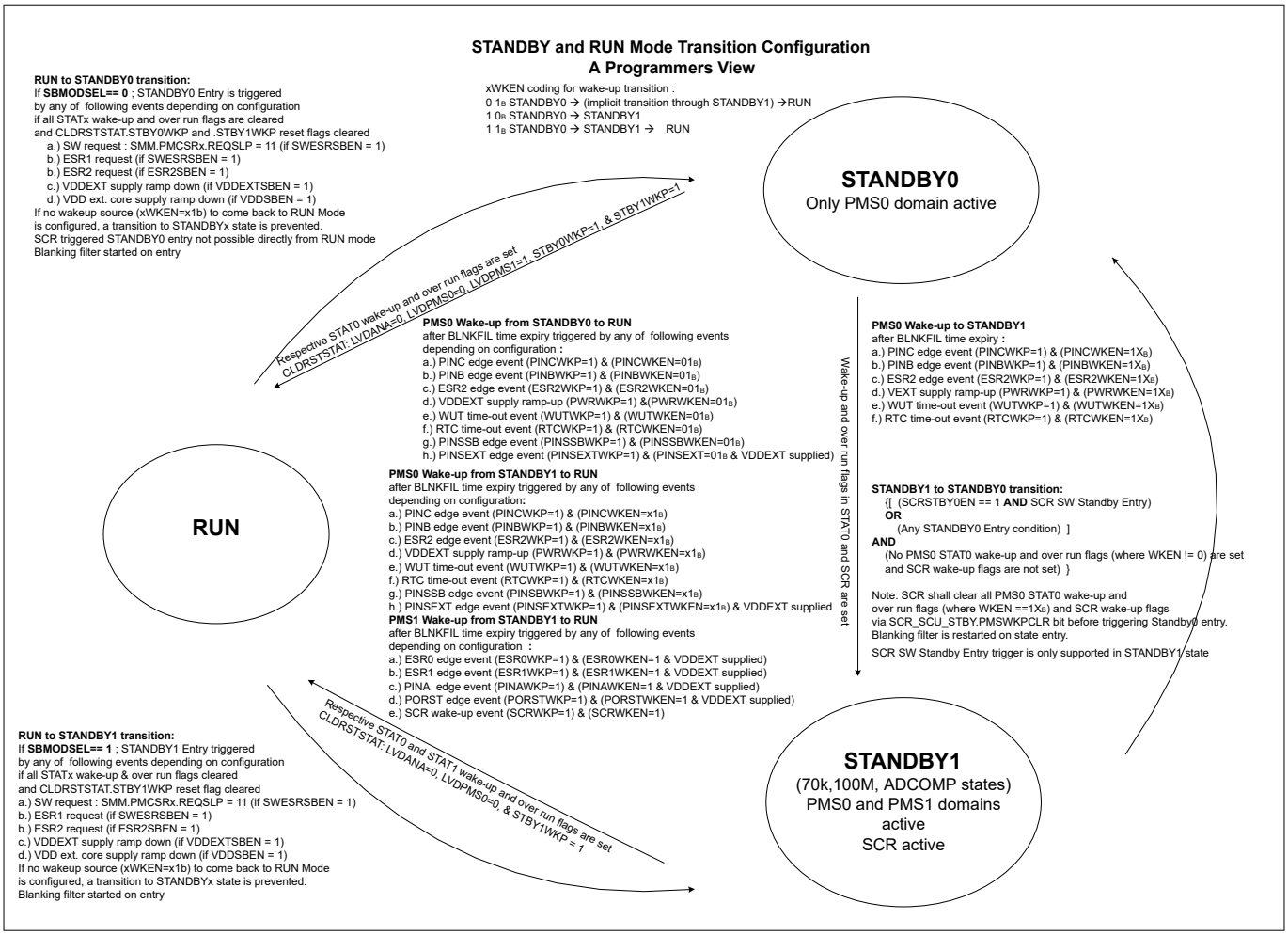


Figure 798 Standby mode configuration and transitions

STANDBY Mode transitions and conditions for transitions are depicted in the figure above.

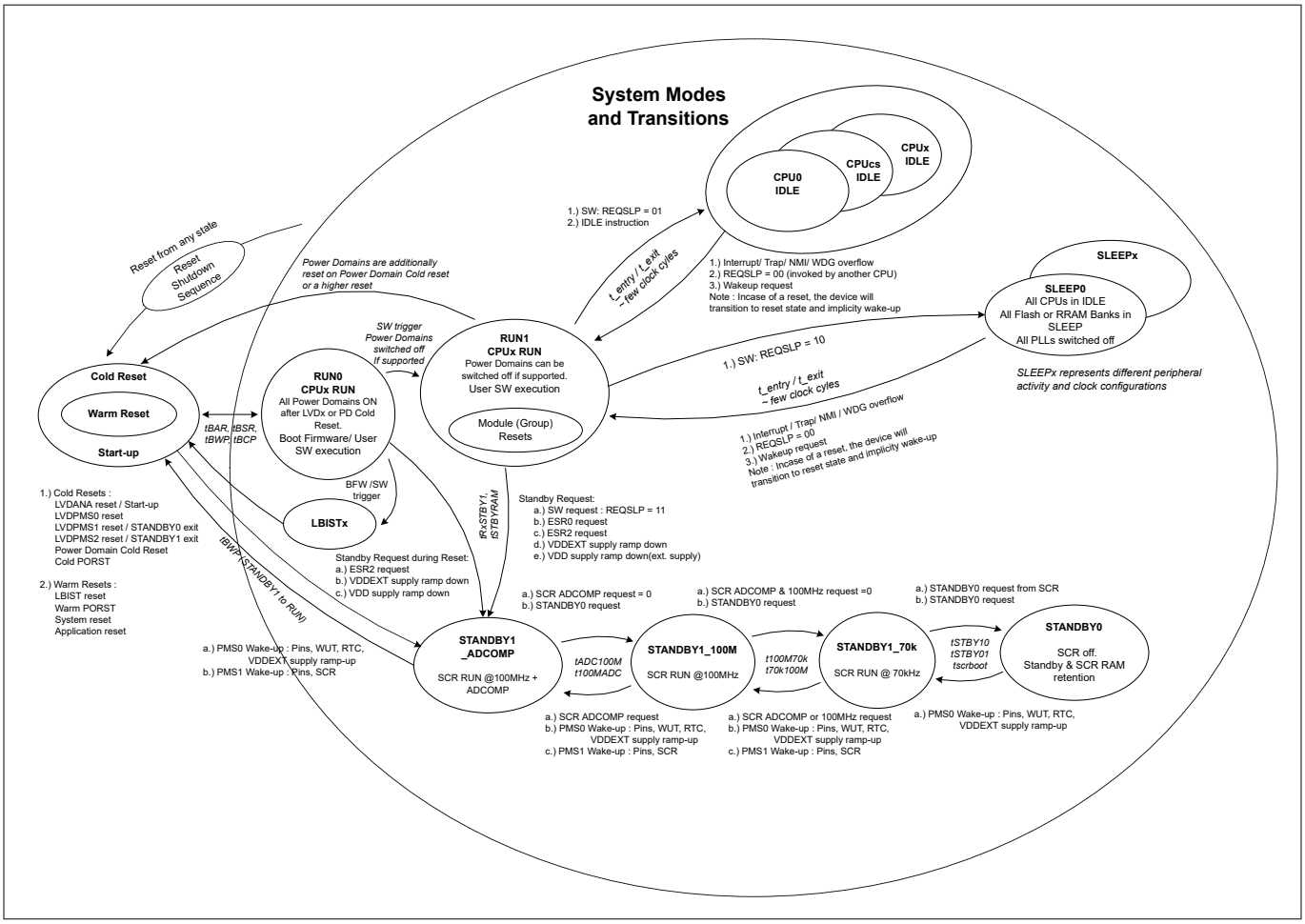


Figure 799 System states and transitions

The above figure depicts the STANDBY mode transitions in context of the System modes. STANDBY modes are entered from RUN1 state on a STANDBY request. Transition within STANDBY states take place sequentially from STANDBY1_ADCOMP to STANDBY1_100M to STANDBY1_70k to finally STANDBY0 state depending on transition conditions.

Table 1433 Standby entry and WUT and RTC wake-up use cases

Mode transition	Wake-up event WUT	Wake-up event RTC
STANDBY0 ← → RUN	WUT in Auto Stop or Auto reload mode. WAKEUP_CON0.WUTWKEN = 1 STANDBY0 entry triggered by V_{DDEXT} or V_{DD} supply ramp-down or software request, or *ESR1* pin rising / falling edge event or *ESR2* falling edge event. On wake-up, *ESR2* pin can be asserted to ramp V_{DDEXT} or V_{DD} external supply if not available. Wake-up flag cleared by CPU via WKPCLR0 register before renewed standby entry.	RTC running on 70 kHz Standby clock or 32 kHz external crystal. WAKEUP_CON0.RTCWKEN = 1 STANDBY0 entry triggered by V_{DDEXT} or V_{DD} supply ramp-down or software request, or *ESR1* pin rising / falling edge event or *ESR2* falling edge event. On wake-up, *ESR2* pin can be asserted to ramp V_{DDEXT} or V_{DD} external supply if not available. Wake-up flag cleared by CPU via WKPCLR0 register before renewed standby entry.

(table continues...)

Table 1433 (continued) Standby entry and WUT and RTC wake-up use cases

Mode transition	Wake-up event WUT	Wake-up event RTC
STANDBY0 ← → STANDBY1	WUT in Auto Stop or Auto reload mode. WAKEUP_CON0.WUTWKEN = 2 STANDBY0 entry triggered by SCR. Wake-up flag cleared by SCR via WKPCLR register before renewed standby entry.	RTC Calibration maybe done depending on temperature measurement in SCR. WAKEUP_CON0.RTCWKEN = 2 STANDBY0 entry triggered by SCR. Wake-up flag cleared by SCR via WKPCLR register before renewed standby entry.
STANDBY1 ← → RUN	WUT in Auto Stop or Auto reload mode. Only transition between RUN and STANDBY1 mode allowed. WAKEUP_CON0.WUTWKEN = 1 STANDBY1 entry triggered by V_{DDEXT} or V_{DD} supply ramp-down or software request, or *ESR1* pin rising / falling edge event or *ESR2* falling edge event. On wake-up, *ESR2* pin can be asserted to ramp V_{DDEXT} or V_{DD} external supply if not available. Wake-up flag cleared by CPU via WKPCLR0 register before renewed standby entry.	Only transition between RUN and STANDBY1 mode allowed. WAKEUP_CON0.RTCWKEN = 1 STANDBY1 entry triggered by V_{DDEXT} or V_{DD} supply ramp-down or software request, or *ESR1* pin rising / falling edge event or *ESR2* falling edge event. On wake-up, *ESR2* pin can be asserted to ramp V_{DDEXT} or V_{DD} external supply if not available. Wake-up flag cleared by CPU via WKPCLR0 register before renewed standby entry.
Sequential wake-up from STANDBY0 ← → STANDBY1 ← → RUN	WUT in Auto reload mode. WAKEUP_CON0.WUTWKEN = 3 STANDBY0 entry triggered by V_{DDEXT} or V_{DD} supply ramp-down or software request, or *ESR1* pin rising / falling edge event or *ESR2* falling edge event. On wake-up, *ESR2* pin can be asserted to ramp V_{DDEXT} or V_{DD} external supply if not available on a STANDBY1 to RUN transition. Wake-up flag cleared by CPU via WKPCLR0 register before renewed standby entry. No need for interim wake-up clear in STANDBY1 mode. Wake-up flag triggers the first wake-up to STANDBY1 and over-run flag triggers the second wake-up to RUN.	WAKEUP_CON0.RTCWKEN = 3 STANDBY0 entry triggered by V_{DDEXT} or V_{DD} supply ramp-down or software request, or *ESR1* pin rising / falling edge event or *ESR2* falling edge event. On wake-up, *ESR2* pin can be asserted to ramp V_{DDEXT} or V_{DD} external supply if not available on a STANDBY1 to RUN transition. Wake-up flag cleared by CPU via WKPCLR0 register before renewed standby entry. No need for interim wake-up clear in STANDBY1 mode. Wake-up flag triggers the first wake-up to STANDBY1 and over-run flag triggers the second wake-up to RUN.

Table 1434 Standby entry and SCR and supply wake-up use cases

Mode transition	Wake-up event V_{DDEXT} supply ramp-up	Wake-up event SCR
STANDBY0 ← → RUN	STANDBY0 entry event - V_{DDEXT} or V_{DD} supply ramp-down. WAKEUP_CON0.PWRWKEN = 1 Wake-up flag cleared by CPU via WKPCLR0 register before renewed standby entry.	Not supported
STANDBY0 ← → STANDBY1	STANDBY0 entry event - V_{DDEXT} or V_{DD} supply ramp-down. WAKEUP_CON0.PWRWKEN = 2 Wake-up flag cleared by SCR via WKPCLR register before renewed standby entry.	Not supported
STANDBY1 ← → RUN	Only transition between RUN and STANDBY1 mode allowed. STANDBY1 entry event - V_{DDEXT} or V_{DD} supply ramp-down. WAKEUP_CON0.PWRWKEN = 1 Wake-up flag cleared by CPU via WKPCLR0 register before renewed standby entry.	SCR is active. Only transition between RUN and STANDBY1 mode allowed. WAKEUP_CON0.SCRWKEN = 1 STANDBY1 entry triggered by V_{DDEXT} or V_{DD} supply ramp-down or software request, or *ESR1* pin rising / falling edge event or *ESR2* falling edge event. On wake-up, *ESR2* pin can be asserted to ramp V_{DDEXT} or V_{DD} external Supply if not available. Wake-up flag cleared by CPU via WKPCLR0 register before renewed standby entry.
Sequential wake-up from STANDBY0 ← → STANDBY1 ← → RUN	Not supported	Not supported

Table 1435 Standby entry and pin wake-up use cases

Mode transition	Wake-up event *PINC*/*ESR2*/*PINB*	Wake-up event *PINSSB*/*PINSEXT*	Wake-up event *PINA*/*ESR0*/*ESR1*/*PORST*
STANDBY0 ← → RUN	WAKEUP_CON0. PINC/PINB/ ESR2WKEN = 1. EDCON and DFEN configures the pin edge and filter. STANDBY0 entry triggered by V_{DDEXT} or V_{DD} supply ramp-down or software request, or *ESR1* pin rising / falling edge event or *ESR2* falling edge event. On a pin edge event , wake-up is triggered. On wake-up, *ESR2* pin can be asserted to ramp V_{DDEXT} or V_{DD} external supply if not available. Wake-up flag cleared by CPU via WKPCLR0 register before renewed standby entry.	WAKEUP_CON0. PINSSBWKEN/ PINSEXTWKEN = 1. Wake-up pin configuration in PORT registers. STANDBY0 entry triggered by V_{DDEXT} or V_{DD} supply ramp-down or software request, or *ESR1* pin rising / falling edge event or *ESR2* falling edge event. On a pin edge event , wake-up is triggered. On wake-up, *ESR2* pin can be asserted to ramp V_{DDEXT} or V_{DD} external supply if not available. Wake-up source can be read out from PORT registers as to which pin caused the wake-up. Wake-up flag cleared by CPU via WKPCLR0 register before renewed standby entry.	Not supported
STANDBY0 ← → STANDBY1	WAKEUP_CON0. PINC/ PINB / ESR2WKEN = 2. EDCON and DFEN configures the pin edge and filter. STANDBY0 entry triggered by SCR. Wake-up flag cleared by SCR via WKPCLR register before renewed Standby entry.	Not supported	Not supported

(table continues...)

Table 1435 (continued) Standby entry and pin wake-up use cases

Mode transition	Wake-up event *PINC*/*ESR2*/*PINB*	Wake-up event *PINSSB*/*PINSEXT*	Wake-up event *PINA*/*ESR0*/*ESR1*/*PORST*
STANDBY1 ← → RUN	Only transition between RUN and STANDBY1 mode allowed. WAKEUP_CON0. PINC/ PINB/ ESR2WKEN = 1. EDCON and DFEN configures the pin edge and filter. STANDBY1 entry triggered by V_{DDEXT} or V_{DD} supply ramp-down or software request, or *ESR1* pin rising / falling edge event or *ESR2* falling edge event. On wake-up, *ESR2* pin can be asserted to ramp V_{DDEXT} or V_{DD} external supply if not available. Wake-up flag cleared by CPU via WKPCLR0 register before renewed standby entry.	Only transition between RUN and STANDBY1 mode allowed. WAKEUP_CON0. PINSSBWKEN/ PINSEXTWKEN= 1. Wake-up pin configuration in PORT registers. STANDBY1 entry triggered by V_{DDEXT} or V_{DD} supply ramp-down or software request, or *ESR1* pin rising / falling edge event or *ESR2* falling edge event. On wake-up, *ESR2* pin can be asserted to ramp V_{DDEXT} or V_{DD} external Supply if not available. Wake-up source can be read out from PORT registers as to which pin caused the wake-up. Wake-up flag cleared by CPU via WKPCLR0 register before renewed Standby entry.	Only transition between RUN and STANDBY1 mode allowed. WAKEUP_CON0. PINA/ESR0/ ESR1/ PORSTWKEN = 1. EDCON and DFEN configures the pin edge and filter except for PORST pin. STANDBY1 entry triggered by V_{DDEXT} or V_{DD} supply ramp-down or software request, or *ESR1* pin rising / falling edge event or *ESR2* falling edge event. On wake-up, *ESR2* pin can be asserted to ramp V_{DDEXT} or V_{DD} external supply if not available. Wake-up flag cleared by CPU via WKPCLR0 register before renewed standby entry.
Sequential wake-up from STANDBY0 ← → STANDBY1 ← → RUN	WAKEUP_CON0. PINC/ PINB/ ESR2WKEN = 3 STANDBY0 entry triggered by V_{DDEXT} or V_{DD} supply ramp-down or software request, or *ESR1* pin rising / falling edge event or *ESR2* falling edge event. On wake-up, *ESR2* pin can be asserted to ramp V_{DDEXT} or V_{DD} external supply if not available on a STANDBY1 to RUN transition. Wake-up flag cleared by CPU via WKPCLR0 register before renewed standby entry. No need for interim wake-up clear in STANDBY1 mode. Wake-up flag triggers the first wake-up to STANDBY1 and over-run flag triggers the second wake-up to RUN.	Not supported	Not supported

There is a power saving option available for STANDBY1_100M mode, for intermediate performance use cases, in which the standby controller (SCR) runs with maximum 20 MHz frequency. This option can be configured by the

SCR, in the SCR register SCU_PMS1IVRCON. The feature allows the configuration of the embedded V_{DDPMS1} voltage regulator into a medium power mode, with 2 current sinks disabled (that is, SCU_PMS1IVRCON.SINKCTRLEN = 1_B and SCU_PMS1IVRCON.SINKCTRL = 01_B), resulting into a reduction of the $V_{DDEVRSB}$ current consumption in STANDBY1 mode, for use cases in which the load jumps in the PMS1 domain are reduced. Refer to the SCR chapter for more details. The default configuration is with all current sinks enabled (SINKCTRLEN = 0_B, full SCR performance allowed). In case SINKCTRLEN = 1_B, the state transition times $t_{100MADC}$, $t_{ADC100M}$, and $t_{100M70k}$ (datasheet parameters) are increased.

36.3.7.2 Standby entry

Standby mode entry section documents the trigger events to enter standby state and the sequence to be followed from the system side for a mode transition.

36.3.7.2.1 Standby entry trigger events

STANDBY Entry is triggered by any of the following events to enter STANDBY0 or STANDBY1 mode from RUN mode depending on SBMODSEL bit.

1. Standby entry on V_{DDEXT} supply ramp-down is triggered by the V_{DDEXT} secondary under-voltage event. As an additional robustness measure to retain Standby RAM content during fast supply ramp-down, Standby entry is triggered latest when the primary under-voltage event happens. In this case, cold PORST is not triggered, instead the device enters Standby mode when VDDEXTSBEN enable bit is set
2. Standby entry on V_{DD} supply ramp-down is triggered by the V_{DD} secondary under-voltage event. As an additional robustness measure to retain Standby RAM content during fast supply ramp-down, Standby entry is triggered latest when the primary under-voltage event happens. In this case, cold PORST is not triggered, instead the device enters Standby mode when VDDSBEN enable bit is set. Standby entry on VDD ramp-down is always possible, independent of whether VDDEXT and VDDEVRSB is supplied by separate or common supply rail
3. Standby entry on Software request (write to register bit-field PMCSRx.REQSLP = 11B). The software request is handled in SMM module and need to be additionally configured in SMM.PMSWCR1.STBYEV = 0 and STANDBY_CON1.SWESRSBEN = 1 register bit-fields. Standby mode through software may be entered based on a singular decision from a master CPU based on the configuration in the CPUSEL register. It may also be entered as an unanimous decision of all the CPUs when ALL of the PMCSRx registers in the system request it and CPUSEL = 111B
4. Standby entry on *ESR1* pin rising or falling edge event. The selection of rising and falling edge and the filter is configured in SMM ESRCFG1.EDCON and DFEN register bit-fields. The ESR 1 request is handled in SMM module and need to be additionally configured through SMM.PMSWCR1.STBYEV = 1 STANDBY_CON1.SWESRSBEN = 1 register bit-fields
5. Standby entry on *ESR2* pin falling edge event, if configured in STANDBY_CON0.ESR2SBEN = 1 register bit-field. A digital filter can be selected in the STANDBY_CON0.ESR2DFEN register bit-field

SCR can trigger STANDBY1 to STANDBY0 entry on a SCR software request if SCRSTBY0EN bit is set.

To ensure the standby entry triggered by the secondary under-voltage event, for a supply ramp-down slope of maximum 10 V/ms, it is recommended to configure the VMONS under-voltage threshold higher than the $V_{xxPRIUV}$ level by at least 2% of the nominal voltage level on the respective rail.

36.3.7.2.2 Configuration for standby entry sequence on V_{DDEXT} ramp-down and wake-up on V_{DDEXT} ramp-up

It is recommended to ramp-down modules before entering Standby mode to avoid large load jumps during Standby entry. The required steps are detailed here.

1. Disabling Processing Units and Peripherals sequentially:
   - All peripherals and interrupts associated with CPUs except master CPU are switched off and all pending interrupt flags shall be cleared. This is to avoid wake-up of the CPUs once they are put into IDLE state
   - All CPUs except the master CPU are sequentially put into IDLE state. PPU is also set into IDLE state if available
   - All watchdogs may be disabled or reconfigured for slower modes
   - Peripheral module clocks are switched off in the respective CLC.DISR registers
   - Standby SMU module shall be disabled before going into Standby mode. The SMM PMSWCR1.IRADIS bit set to disable Idle Request Acknowledge sequence activation for fast Standby Mode entry
2. Ramping Down System Clock:
   - System Clock is ramped down using the Ramp Oscillator/FLL and switched to internal 100 MHz clock source
   - System PLL and Peripheral PLL are switched off
   - Clock dividers are programmed to lower values
   - Master CPU frequency reduction in steps or using the ramp oscillator compliant to load jump constraints
3. Switching from Flash to RAM Execution:
   - Master CPU code execution switched from Flash to Program RAM
   - Flash modules may be deactivated
4. Standby RAM Configuration:
   - Standby RAM blocks which need to be kept supplied during STANDBY modes are selected through STANDBY_CON0.SBRAMSEL and STANDBY_CON0.SCRRAMSEL bits. D-Cache write back to be executed before Standby entry
5. Pad State during Standby:
   - Configure pad state through PAD_CON.ALLTRIST bit for V_{DDEXT} supplied pins and PAD_CON.SBTRIST bit for $V_{DDEVRSB}$ supplied pins. All pads may be set into tristate or have pull-up device active
   - Configure PAD_CON.ESR0PD bit to set *ESR0* behavior as reset output active or tristate during Standby
   - In case of *HWCFG [2:1,6]* pins it is recommended to tie them to external pull devices
6. SCR Configuration during Standby:
   - SCR may be kept running before entering the Standby state. In standby mode the shared ports supplied by $V_{DDEVRSB}$ are configured to be in tristate or pulled up or left to the control of SCR . Note that if the port control is granted to SCR by setting PCRSEL.PCSRx bit to 1_B but the port is disabled by SCR by setting SCR_Pn_PDISC.PDISm bit to 1_B then the behavior of the port depends upon the PAD_CON.SBTRIST bit
   - The enabling of wake-up from SCR through WAKEUP_CON1.SCRWKEN shall be programmed if wake-up source is SCR
   - Take care that SCR is not reset on a standby entry by clearing SCR_CON0.PORSTREQ bit-field. Select the 70 kHz Standby clock source (fSB) through SCR_CON0.SCRCLKSEL bit if required
7. Clearing Wake-up flags:
   - All xWKP/xOVRUN flags activated by the respective xWKEN bits shall be cleared before a renewed Standby entry request, otherwise the System will remain in RUN state and not enter STANDBY state

8. Selection of either entry to STANDBY0 or STANDBY1 state when entering Standby from RUN mode through STANDBY_CON1.SBMODSEL bit
   - Transitions from STANDBY1 to STANDBY0 state can be enabled by configuring the STANDBY_CON0.SCRSTBY0EN register bit. It shall be ensured that the primary under-voltage reset monitors are active before Standby entry is triggered and shall not be disabled
9. Configure Blanking filter:
   - Blanking filter blocks immediate wake-up of the device for a configurable duration during and immediately after standby entry trigger
   - A minimum blanking filter shall be configured to suppress wake-up during transition to standby through WAKEUP_CON0.BLNKFIL bits. The wake-up events happening during the blanking filter active phase would be lost
   - The wake-up event is recognized and triggered only in the STANDBY state after blanking filter expiry and not during transition states. If the same event trigger (for example ESR2) is used for standby entry and for wake-up, then it has to be ensured that the blanking filter is activated
10. Wake-up Trigger Configurations:
   - Enable wake-up on V_{DDEXT} supply ramp-up using the WAKEUP_CON0.PWRWKEN bit
   - Ensure that both PWRWKEN and STANDBY_CON0.VDDEXTSBEN register bits are both set before entering Standby mode
11. Standby Entry configuration on V_{DDEXT} supply ramp-down:
   - Configure Standby entry event in VDDEXTSBEN and VDDSBEN register bits respectively. In case standby entry is triggered by V_{DDEXT} supply ramp-down, the threshold is configured in VMONS_VDDEXTCON.UVVAL and transition condition in VMONS_VDDEXTCON.UVMOD register bit-fields respectively
   - In case of nominal V_{DDEXT} supply voltage of 5 V, it is recommended to configure the UVVAL register bit-field at 4 V for standby entry to have adequate distance to primary reset levels as well as operational region limits
   - In case of nominal V_{DDEXT} supply voltage of 3.3 V, it is recommended to configure the UVVAL register bit-field at 3.1 V for standby entry above primary reset levels. Nevertheless since there is only a minimal margin to reset levels in this case, Standby entry is additionally triggered by the crossing of primary under-voltage limits if the VDDEXTSBEN register bit is set to ensure Standby entry in case of fast V_{DDEXT} slopes
   - In states where VMONP or VMONS monitors are not available, the standby entry is triggered by LVD detector on V_{DDEXT} supply ramp-down
   - During V_{DDEXT} supply ramp-down, depending on external regulator and V_{DD} buffer capacitors, it may happen that consequently V_{DD} ramps down faster before the V_{DDEXT} standby entry threshold is reached. In this case once V_{DD} supply drops below primary under-voltage threshold, cold PORST state is entered and maintained until the V_{DDEXT} supply ramps down to the standby entry threshold. Additionally in case of external V_{DD} supply, to avoid a power fail event before going into standby mode, standby entry may also be triggered on a threshold additionally based on V_{DD} supply level by setting STANDBY_CON1.VDDSBEN bit to 1_B . If V_{DD} is generated using the EVRC DCDC regulator, standby entry using V_{DD} ramp-down is not supported
   - If Standby entry is triggered by V_{DD} supply ramp-down, the threshold is configured in VMONS_VDDCON.UVVAL and the transition condition in the VMONS_VDDCON.UVMOD register respectively
   - Standby entry filter based on multiple secondary monitor events can be configured in the VMONS_VxxCON.FILTER register bit-field. The VDDEXTSBEN register bit shall be set to ensure that

when V_{DDEXT} supply is removed during Standby state, no LVD reset is generated consequently exiting from Standby mode

- When entering standby with external MOSFET complementary switch in case of EVRC regulator, it should be taken care that when V_{DDEXT} supply is ramped down that also the supply to the MOSFETs is ramped down along with the V_{DDEXT} supply. Otherwise it may happen that the $V_{DDEXTDC}$ supply is still held high through the diode path through the *VGATE1P/VGATE1N* pins to the $V_{DDEXTDC}$ supply rail. If $V_{DDEXTDC}$ and V_{DDEXT} are connected together, this would lead to immediate wake-up after blanking time has expired as $V_{DDEXTDC}$ supply is above the wake-up voltage threshold between 2,6 - 2,97 V. Adequate blanking filter time shall be configured before entering standby to ensure that $V_{DDEXTDC}$ has ramped down completely within this time to avoid immediate wake-up

12. External Standby trigger and Standby transition:

- The external regulator is communicated to switch off V_{DDEXT} supply. A controlled ramp-down of V_{DDEXT} slope during Standby entry is recommended from the external regulator. For example - 0.5 V/ms to 1.5 V/ms
- Ensure that the V_{DDEXT} supply is ramped below the V_{DDEXT} LVD reset level after standby entry before the blanking filter time has expired. This is to avoid an immediate wake-up triggered by the residual V_{DDEXT} voltage if it is above V_{DDEXT} LVD reset level after blanking time has expired
- Ensure that the V_{DD} supply rail is consequently switched off after V_{DDEXT} ramp-down to reduce standby current within the blanking filter time
- Once a Standby entry event is recognized, the primary under-voltage reset generation is disabled and Standby RAM supply is switched from V_{DD} to $V_{DDSBRAM}$ within a few clock cycles. On entry into Standby mode, the blanking filter is activated. The external standby regulator continues to supply the Standby domain through $V_{DDEVRSB}$ supply pin. The blanking filter is always activated on Standby entry

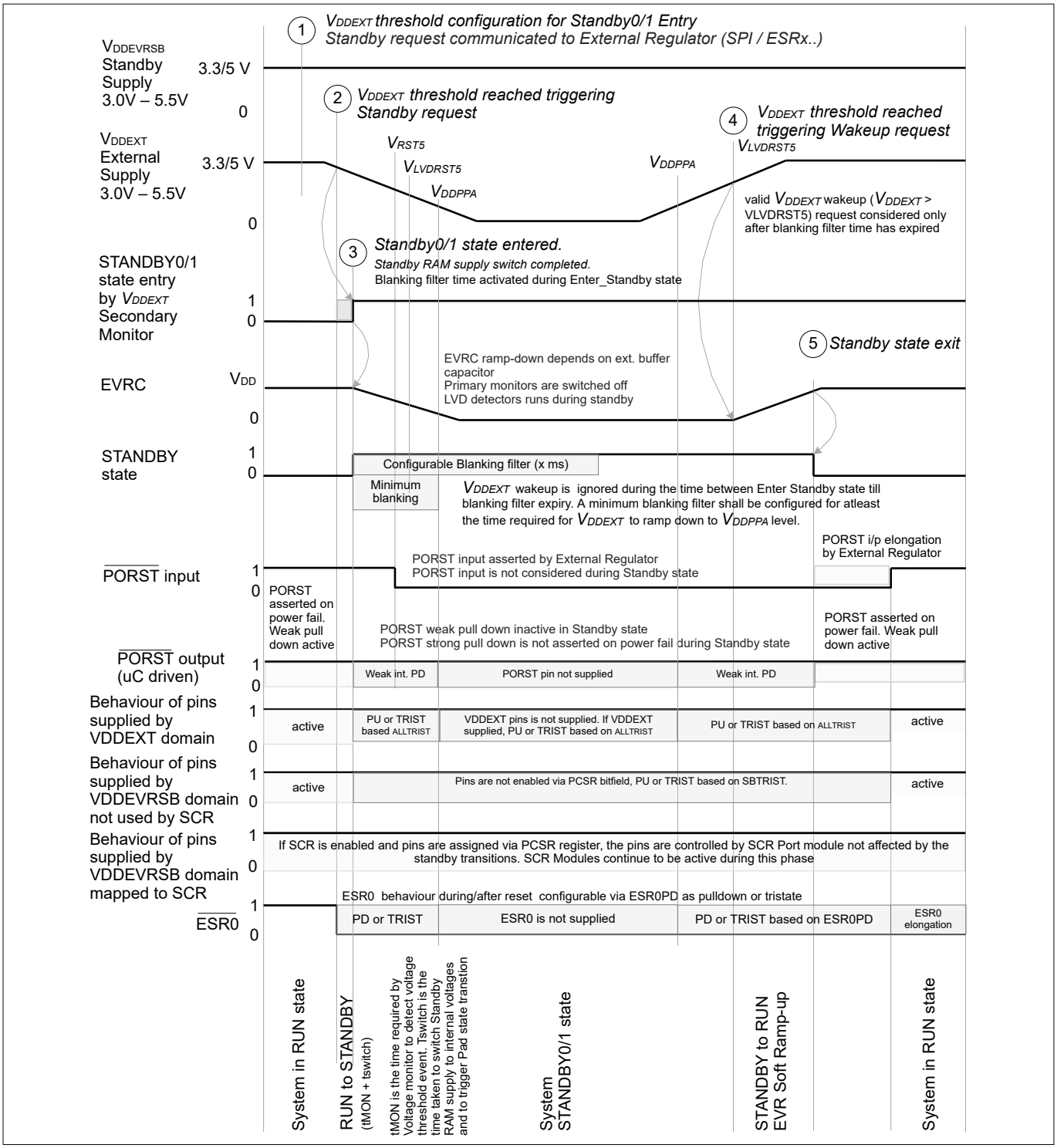


Figure 800 Standby entry on V_{DDEXT} ramp-down and wake-up on V_{DDEXT} ramp-up

36.3.7.2.3 Configuration for standby entry sequence upon SW or ESRx trigger

1. Disabling Processing Units and Peripherals sequentially:

- All peripherals and interrupts associated with CPUs except master CPU are switched off and all pending interrupt flags shall be cleared. This is to avoid wake-up of the CPUs once they are put into IDLE state

- All CPUs except the master CPU are sequentially put into IDLE state. PPU is also set into IDLE state if available
- All watchdogs may be disabled or reconfigured for slower modes
- Peripheral module clocks are switched off in the respective CLC.DISR registers
- Standby SMU module shall be disabled before going into Standby mode. The SMM PMSWCR1.IRADIS bit set to disable Idle Request Acknowledge sequence activation for fast Standby Mode entry

2. Ramping Down System Clock:

- System Clock is ramped down using the Ramp Oscillator/FLL and switched to internal 100 MHz clock source
- System PLL and Peripheral PLL are switched off
- Clock dividers are programmed to lower values
- Master CPU frequency reduction in steps or using the ramp oscillator compliant to load jump constraints

3. Switching from Flash to RAM Execution:

- Master CPU code execution switched from Flash to Program RAM
- Flash modules may be deactivated

4. Standby RAM Configuration:

- Standby RAM blocks which need to be kept supplied during STANDBY modes are selected through STANDBY_CON0.SBRAMSEL and STANDBY_CON0.SCRRAMSEL bits. D-Cache write back to be executed before Standby entry

5. Pad State during Standby:

- Configure pad state through PAD_CON.ALLTRIST bit for V_{DDEXT} supplied pins and PAD_CON.SBTRIST bit for $V_{DDEVRSB}$ supplied pins. All pads may be set into tristate or have pull-up device active
- Configure PAD_CON.ESR0PD bit to set *ESR0* behavior as reset output active or tristate during Standby
- In case of *HWCFG [2:1,6]* pins it is recommended to tie them to external pull devices

6. SCR Configuration during Standby:

- SCR may be kept running before entering the Standby state. In standby mode the shared ports supplied by $V_{DDEVRSB}$ are configured to be in tristate or pulled up or left to the control of SCR . Note that if the port control is granted to SCR by setting PCRSEL.PCSRx bit to 1_B but the port is disabled by SCR by setting SCR_Pn_PDISC.PDISm bit to 1_B then the behavior of the port depends upon the PAD_CON.SBTRIST bit
- The enabling of wake-up from SCR through WAKEUP_CON1.SCRWKEN shall be programmed if wake-up source is SCR
- Take care that SCR is not reset on a standby entry by clearing SCR_CON0.PORSTREQ bit-field. Select the 70 kHz Standby clock source (fSB) through SCR_CON0.SCRCLKSEL bit if required

7. Clearing Wake-up flags:

- All xWKP/xOVRUN flags activated by the respective xWKEN bits shall be cleared before a renewed Standby entry request, otherwise the System will remain in RUN state and not enter STANDBY state

8. Selection of either entry to STANDBY0 or STANDBY1 state when entering Standby from RUN mode through STANDBY_CON1.SBMODSEL bit

- Transitions from STANDBY1 to STANDBY0 state can be enabled by configuring the STANDBY_CON0.SCRSTBY0EN register bit. It shall be ensured that the primary under-voltage reset monitors are active before Standby entry is triggered and shall not be disabled

9. Configure Blanking filter:

- Blanking filter blocks immediate wake-up of the device for a configurable duration during and immediately after standby entry trigger

- A minimum blanking filter shall be configured to suppress wake-up during transition to standby through WAKEUP_CON0.BLNKFIL bits. The wake-up events happening during the blanking filter active phase would be lost
- The wake-up event is recognized and triggered only in the STANDBY state after blanking filter expiry and not during transition states. If the same event trigger (for example ESR2) is used for standby entry and for wake-up, then it has to be ensured that the blanking filter is activated

10. One of the following Wake-up Trigger Configurations may be chosen:

- Wake-up Timer may also be kept running before entering the Standby state. The wake-up from WUT may be activated through WAKEUP_CON0.WUTWKEN bit
- Real Time Clock may also be kept running before entering the Standby state. The wake-up from RTC may be activated through WAKEUP_CON0.RTCWKEN bit
- Wake-up configuration and filter activation for *ESR2*, *PINC*, *PORST*, *PINA*, *PINB*, *ESR0*, and *ESR1* is configured through xWKEN, EDCON, and DFEN bits

11. Standby Entry configuration:

- Configure Standby Entry event in SMM.PMSWCR1.STBYEV register bits. Standby request is issued through software or ESRx event

12. Pending Interrupt handling before Standby request:

- If a write to PMCSRx.REQSLP register bit-field occurs while the interrupt router has a pending interrupt for CPU, the device may not enter Standby mode. To ensure the transition to Standby mode, all interrupts that are not intended to cause Run Mode to be re-entered or retained, should either have the SRE bit cleared in the respective SRN or be guaranteed to have the SRR bit clear
- So long as the SRE bit and SRR bit are not both set, there will not be a pending interrupt to inhibit standby mode transition. If the SRR bits are cleared, after the last SRN is modified, there also needs to be a synchronization step for the interrupt router outputs to reflect the update before the PMCSRx.REQSLP bit-field is written
- To ensure a deterministic end of CPU execution after the Standby mode request, the write to PMCSRx.REQSLP should be followed by a DSYNC and a WAIT instruction

13. Standby trigger and Standby transition:

- Once a Standby entry event is recognized, the primary under-voltage reset generation is disabled and the Standby RAM supply is switched from V_{DD} to $V_{DDSBRAM}$ within a few clock cycles. On entry into Standby mode, the blanking filter is activated. The blanking filter is always activated on entry to the Standby state

36.3.7.3 Interaction of warm reset and standby mode transitions

This section addresses following use cases:

- A standby request by V_{DDEXT} ramp-down is issued during warm reset
- A warm reset is triggered when a standby mode transition is ongoing

The PMS has separate state machines, which operate independently from the rest of the MCU. The PMS states are not affected by warm resets (including system reset and application reset). The table "Effect of Reset on Device Functions" from the SMM chapter shows how the MCU modules are affected by different reset types.

Therefore, standby mode entry is achieved even in the warm reset state of the MCU system modes.

36.3.7.4 Wake-up from standby

Wake-up from standby modes is triggered by supply ramp-up events, rising or falling edge events on multiple pins, and compare match events from timers and real time clock units.

Exiting standby mode - wake-up events

Wake-up events are categorized as shown in the following figures as to whether the device is in STANDBY0 or STANDBY1 state and whether V_{DDEXT} is kept supplied during Standby state in addition to $V_{DDEVRSB}$ supply. If V_{DDEXT} is not supplied, the wake-up is delayed until V_{DDEXT} has ramped up. The wake-up event in case of standby mode occurs after the blanking filter time has expired upon any of the following triggers:

1. On a V_{DDEXT} Supply ramp-up after the blanking filter time has expired. The Standby controller unit or *ESR2* pin maybe used to re-trigger V_{DDEXT} IO supply ramp-up to the external regulator
   - SCR may be active and wake-up can indirectly be triggered through SCR by communicating to external regulator to request the ramp-up of V_{DDEXT} voltage
   - Likewise wake-up can indirectly be triggered through *ESR2* pin by communicating to external regulator to request the ramp-up of V_{DDEXT} voltage configured through PAD_CON.ESR2WKPRUNPD register bit-field

   After V_{DDEXT} wake-up is recognized, it is expected that the V_{DDEXT} supply is stable afterwards. The detection time of the V_{DDEXT} ramp-up supply detector itself on reaching $V_{LVDRSTEXT}$ level is within 10 µs. Nevertheless the complete time for start-up from Standby mode is quite the same as that for normal start-up and is documented t_{BP} parameter in datasheet. In case of immediate V_{DDEXT} power fail consequent to V_{DDEXT} wake-up, LVD reset or cold PORST may be triggered. It is expected that the V_{DDEXT} supply has ramped down within the configured blanking filter time. Blanking filter shall be configured for at least the time required for V_{DDEXT} to ramp down to $V_{LVDRSTEXT}$ level + 1 ms. V_{DDEXT} wake-up is ignored during the time between Enter Standby state till blanking filter expiry. Wake-up triggered on a V_{DDEXT} Supply ramp-up is indicated in WAKEUP_STAT0.PWRWKP register bit and shall be cleared by WAKEUP_CLR0.PWRWKPCLR clear register bit
2. Wake-up from SCR as indicated in WAKEUP_STAT1.SCRWKP status register is in turn caused by the following events:
   - Edge transitions at the shared ports
   - RTC interrupt
   - SCR watchdog overflow
   - Selected interrupts from communication modules
   - ADCOMP analog channel compare event

   The WAKEUP_STAT1.SCRWKP flag is set on wake-up. The WAKEUP_STAT1.SCROVRUN flag is set to indicate overrun behavior in case of multiple unserviced wake-ups. The SCR wake-up is ignored during the time between the enter standby transition until the blanking filter time expiry
3. Wake-up from Wake-up Timer (WUT)
4. Wake-up from Real Time Clock (RTC)
5. Wake-up from Pins: External events may be mapped to ESRx/PINx pins in turn acting as wake-up signals for the system. In Run Mode, *ESR1* pin may be used as NMI interface from external devices and *ESR0* for reset output propagation. *ESR2* pin may be used to indicate wake-up events to external devices. In Standby Mode, an edge event on any of the ESRx pins may be configured to trigger wake-up of the main core domain through WAKEUP_CONx.ESRxWKEN bit and is reflected in WAKEUP_STATx.ESRxWKP status flag. It can be configured to trigger a wake-up on rising, falling, or both edges via WAKEUP_CONx.ESRxEDCON bit. The minimum pulse width of the external wake-up input signal without the digital filter activated shall be at least 3 clock cycles (of the 70 kHz standby clock). Glitches on ESRx input are filtered out by activating the filter through WAKEUP_CONx.ESRxDFEN bit. The reset behavior of the *ESR*x pins is documented in the External Service Requests section in the SMM chapter. Additional pins (*PINA*, *PINB*, and *PINC*) may likewise be configured to trigger wake-up through xEDCON, xDFEN, and

xWKEN bits. On wake-up, WAKEUP_STATx.PINxWKP event flags provide information as to the wake-up source. Application SW must take care after wake-up to clear the event flags through WAKEUP_CLRx.xWKPCLR flags, otherwise standby re-entry is not possible. In case new wake-up events are captured while WAKEUP_STATx.xWKP flags are still set, then WAKEUP_STATx.xOVRRUN flags are set to indicate an overrun state owing to consecutive unserviced wake-up events. xWKP / xOVRUN flags are only set during Standby mode after Blanking Filter expiry when the respective wake-up event happens. The wake-up feature from PORST, ESR0, ESR1, PINA and PINSEXT pins is supported only if VDDEXT is kept supplied in standby modes, otherwise wake-up trigger shall be disabled in xWKEN bit-fields

- Wake-up through *PINC* (P33.12) edge rising or falling edge event
- Wake-up through *ESR2* (P33.11) edge rising or falling edge event
- Wake-up through *PINSSB* low level detection. Refer PORT chapter for more information. The blanking filter shall be enabled for standby entry time to ensure wake-up is triggered
- Wake-up through *PINSEXT* low level detection if V_{DDEXT} is supplied. Refer PORT chapter for more information. If V_{DDEXT} is not supplied during standby mode, the PINSEXTWKEN shall be set to 0 to avoid spurious wake-up. The blanking filter shall be enabled for standby entry time to ensure wake-up is triggered
- Wake-up through *PINB* (P33.13) edge rising or falling edge event
- Wake-up through *PINA* (P14.1) if V_{DDEXT} is supplied. If V_{DDEXT} is not supplied during standby mode, the PINAWKEN shall be set to 0 to avoid spurious wake-up
- Wake-up through *ESR1* if V_{DDEXT} is supplied. If V_{DDEXT} is not supplied during standby mode, the ESR1WKEN shall be set to 0 to avoid spurious wake-up
- Wake-up through *ESR0* if V_{DDEXT} is supplied. If V_{DDEXT} is not supplied during standby mode, the ESR0WKEN shall be set to 0 to avoid spurious wake-up
- Wake-up through *PORST* if V_{DDEXT} is supplied. If V_{DDEXT} is not supplied during standby mode, the PORSTWKEN shall be set to 0 to avoid spurious wake-up

The wake-up from STANDBY0 state is indicated in the CLDRSTSTAT.STBY0WKP and CLDRSTSTAT.STBY1WKP status bits and wake-up from STANDBY1 state is indicated in the CLDRSTSTAT.STBY1WKP status bit respectively. The EVRC regulator is ramped up on wake-up based on the earlier latched configuration in HWCFGx register bits. On wake-up, all pads are either in tristate or are connected to pull-ups as indicated in PAD_STAT.ALLTRIST and PAD_STAT.SBTRIST register bits. *ESR0* behavior is indicated in PAD_CON.ESR0PD register bit. If standby RAM was supplied during the standby state, it is indicated in the STANDBY_CON0.SBRAMSEL register bits. CLDRSTSTATLVDANA bit indicates if the RAM supply was reliable during standby. If CLDRSTSTAT.LVDANA or CLDRSTSTAT.LVDPMS0 is set to 1, this indicates that any of the required supplies for the respective standby mode was not reliable. CLDRSTSTAT.LVDPMS1 bit indicates if the PMS1 domain was in reset.

In case of a transition from STANDBY0 to RUN mode, it is expected that both CLDRSTSTAT.STBY0WKP and CLDRSTSTAT.STBY1WKP bits are set. If any of the CLDRSTSTAT.LVDANA or CLDRSTSTAT.LVDPMS0 is set, then a voltage issue was detected during standby.

In case of a transition from STANDBY1 to RUN mode, it is expected that the CLDRSTSTAT.STBY1WKP bit is set. If any of the CLDRSTSTAT.LVDANA, CLDRSTSTAT.LVDPMS0, or CLDRSTSTAT.LVDPMS1 bits are set, then a voltage issue was detected during standby.

The reset relevant flags shall be cleared through the CLDRSTSTATCLR.CLEAR register bit. The wake-up and over-run status flags are set in WAKEUP_STATx registers and shall be cleared in the WAKEUP_CLRx registers. The wake-up time is nearly the same as the normal boot time as EVRC needs to be started and firmware needs to be consequently executed. The PORST pin output driving function by the PMS is resumed only after all supplies have ramped up.

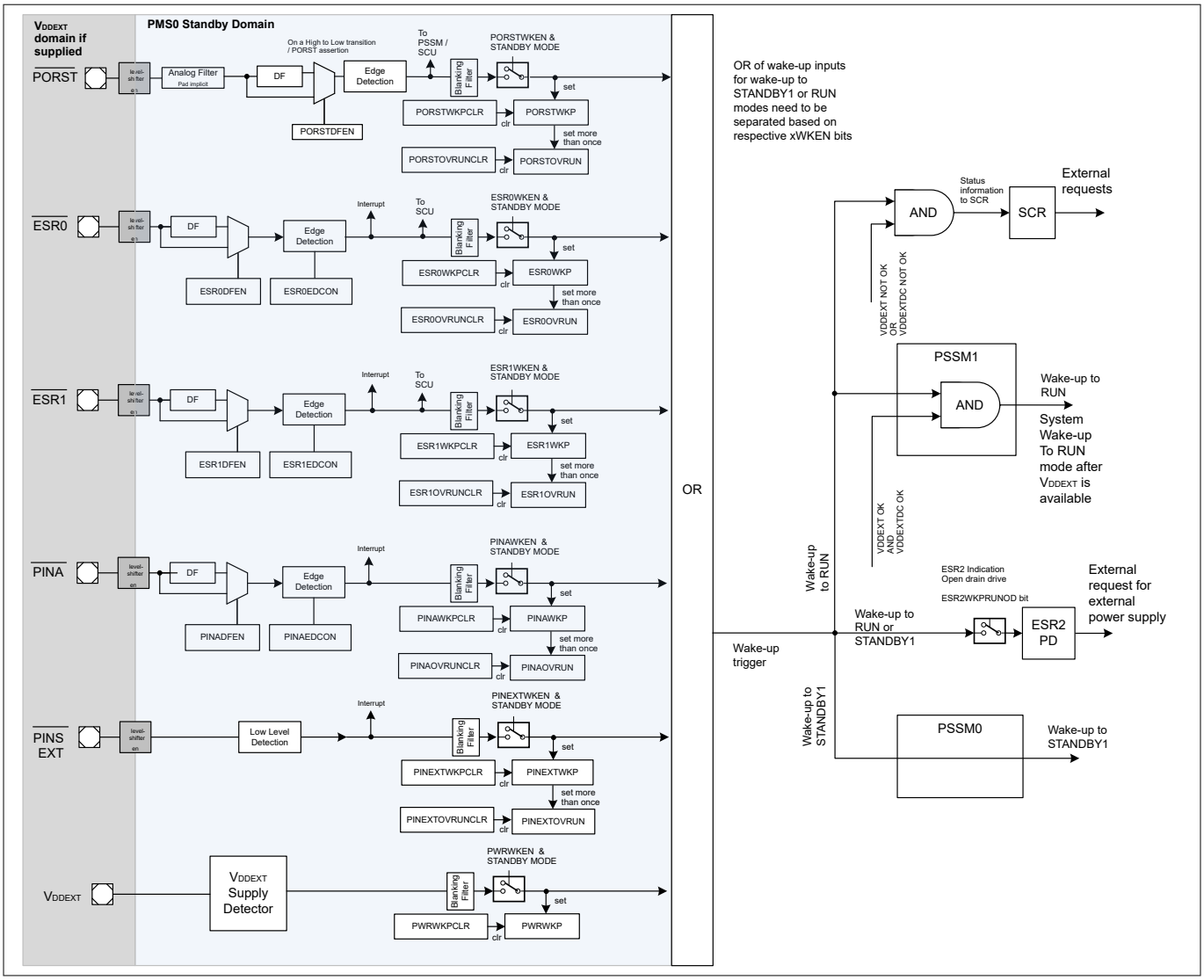


Figure 801 Wake-up sources when V_{DDEXT} is available

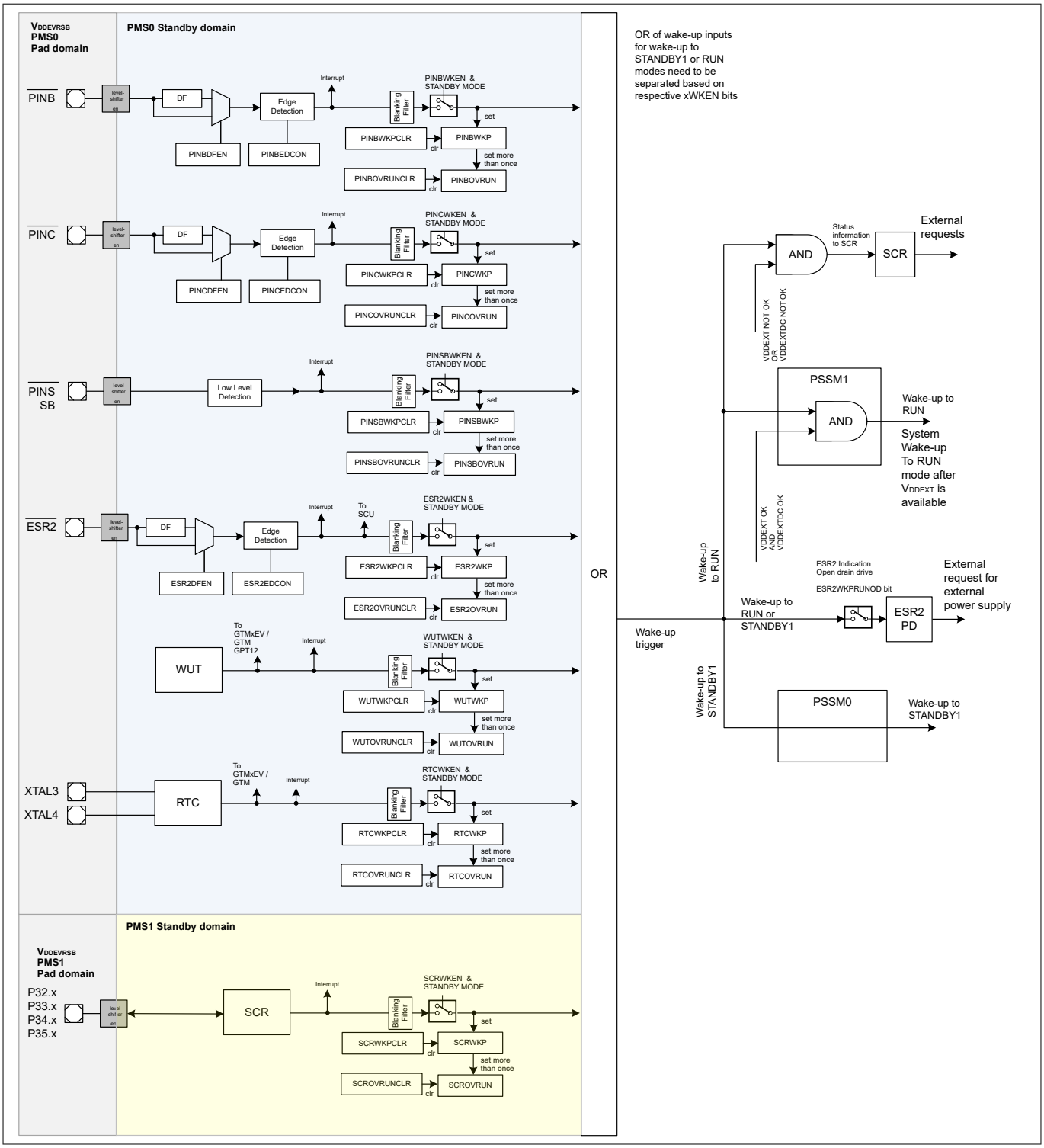


Figure 802 Wake-up sources when $V_{DDEVRSB}$ is available

Exiting standby mode - power fail or reset event

A power fail event of the Standby supply ($V_{DDEVRSB}$ pin) during Standby mode may inevitably result in the loss of Standby RAM contents. Consequently, LVDANA reset event is issued and the Standby domain is set into reset. An under-voltage violation of the $V_{DDSBRAM}$ pre-regulator is indicated in the CLDRSTSTAT.LVDANA flag, which can be used as an indication of a standby supply fail, respectively if the standby RAM content is, as a consequence, not reliable anymore. Cold reset flags CLDRSTSTAT.LVDPMS2, .CLDPORST, and .PDCLDRST flags would always be set after the wake-up from all standby modes, as these domains are devoid of power during STANDBY to

eliminate leakage current. It is recommended to keep a copy of the critical data also in Dflash in order to mitigate the effects if unwanted power fail events cannot be avoided.

In case of V_{DDEXT} supply wake-up, *PORST* pin, *ESR0/1* pins and *PINA* input are not evaluated during Standby mode as it is supplied by V_{DDEXT} domain which is switched off during the STANDBY mode. In case V_{DDEXT} domain is supplied during STANDBY mode, the Standby domain is woken up on PORST assertion depending on PORSTWKEN bit. The SCR may also set into reset simultaneously depending on PORSTREQ bit. The device boots up ramping up the regulators followed by firmware execution similar to a normal device start-up. On PORST wake-up, PORSTWKP event flag is set providing information as to the wake-up source. It should be taken care after wake-up to clear the event flags via PORSTWKPCLR register. In case new PORST wake-up events are captured while PORSTWKP flags are still set, then PORSTOVRRUN flags are set to indicate an overrun state owing to consecutive un-serviced wake-up events. The overrun flag is cleared via PORSTOVRUNCLR bit.

The Standby RAM contents are kept intact after a wake-up caused by PORST assertion. In case of V_{DD} supply under-voltage condition during wake-up phase, It is ensured that the Standby RAM is kept supplied by $V_{DDSBRAM}$ until V_{DD} is back in operational range to avoid Standby RAM data loss or corruption during transition. Reset is propagated to external devices via the *ESR0* pin on exit from Standby mode depending on ESR0PD configuration. Firmware may elongate *ESR0* reset output depending on *ESR0* configuration (which is done in the SMM, see the SMM ESR0CNTCTRL register).

Additional PORST digital filter activated via PORSTDF bit provides additional spike filtering of at least $t_{PORSTDF}$ duration to provide enhanced immunity against spurious spikes. This is in addition to the inherent analog PORST filter delay of the *PORST* pad / pin as documented in the data sheet. After cold PORST the additional PORST digital filter delay is by default inactive. If V_{DDEXT} is supplied, PORST (high to low) during Standby state after blanking filter expiry triggers wake-up.

Table 1436 *PORST* pin assertion behavior on PMS and SCR subsystem during system modes

Reaction to *PORST* pin assertion	RUN mode SLEEP mode	STANDBY1 mode
No reaction	No effect on PMS domain. No reaction on SCR if PORSTREQ = 0 but an SCR_NMI is triggered via the RST bit.	No effect on PMS domain if PORSTWKEN = 0. No reaction on SCR if PORSTREQ = 0
Wake-up	No effect on PMS or SCR domain as the system is already awake	PMS Standby to RUN transition takes place on PORST assertion when V_{DDEXT} is supplied if PORSTWKEN=1
Reset	No reset of PMS domain SCR is reset if PORSTREQ = 1	No reset of PMS domain SCR is reset if PORSTREQ = 1

36.3.7.5 Standby RAM retention

The standby RAM feature is device specific and therefore not applicable to all product variants, please refer to the device feature set in the data sheet (low power features). Therefore, also the STANDBY_CON0 register, which is used to configure this feature, is described in the device-specific section.

The RAMs remain supplied during Standby mode if configured in STANDBY_CON0.SBRAMSEL bits. On wake-up, the status which Standby RAMs remain supplied is reflected in STANDBY_CON0.SBRAMSEL bits. Additional RAM integrity checks may be carried out after wake-up.

The Standby RAM cell array is supplied by a separate supply pin ($V_{DDEVRSB}$) during Standby state via the internal $V_{DDSBRAM}$ regulator. It shall be ensured that the external standby supply source continues to supply $V_{DDEVRSB}$ supply pin during Standby state in operational region. Standby supply status is also monitored and indicated via CLDRSTSTAT.LVDANA status bit which indicates $V_{DDSBRAM}$ supply under-voltage LVD reset. It is to be taken care by the Start-up software after wake-up that Standby RAMs are not initialized if SBRAMSEL bits are set and LVDANA reset had not taken place. Furthermore, if STANDBY_CON0.SBRAMSEL bits are set to enable Standby RAM function and there was a V_{DD} primary under-voltage event, it is ensured that the Standby RAM supply is switched immediately to $V_{DDSBRAM}$ supply rail. This ensures that RAM contents are not corrupted also during main V_{DD} core supply loss.

The SCR RAM constitutes ECC protected 32 kB of XRAM. The SCR XRAM remains supplied during STANDBY0 mode if configured in STANDBY_CON0.SCRRAMSEL bits by $V_{DDSBRAM}$ regulator. On wake-up, the status which Standby RAMs remain supplied is reflected in STANDBY_CON0.SCRRAMSEL bits. During STANDBY1 mode and RUN modes, the SCR XRAM is supplied by V_{DDPMS1} regulator.

36.3.7.6 Wake-up timer (WUT)

The Wake-up Timer is a basic low power counter which may be used to wake-up the system periodically from STANDBY mode. The timer may also be used during RUN, IDLE or SLEEP modes. The following list enumerates the salient features:

- 24-bit counter running on 70 kHz clock source with programmable reload value
- 24-bit counter status register providing the current count value
- Timer resolution of 70 kHz or (70 kHz/2^10) configured through a clock divider:
  - 14.3 µs resolution: 14.3 µs - 240 s range ± tolerance after trimming as stated in datasheet
  - 14.3 ms resolution: 14.3 ms - 2.7 days range ± tolerance after trimming as stated in datasheet
- 3 operating modes:
  - Auto Reload mode - WUT is started and stopped through Software. Automatic reload on counter underflow triggers a system wake-up. WUT runs continuously through STANDBY and run mode transitions
  - STANDBY0 Auto Stop mode - Counter starts counting down from reload value on STANDBY0 entry. Counter stops on underflow and triggers a system wake-up
  - STANDBY1 Auto Stop mode - Counter starts counting down from reload value on STANDBY1 entry. Counter stops on underflow and triggers a system wake-up
- Events on WUT counter underflow:
  - Interrupt request on SRC_PMS2 (WUT) interrupt node on counter underflow during RUN modes
  - Wake-up trigger on counter underflow during STANDBY modes
  - Capture trigger on counter underflow to eGTM (TIM0_7.11) and GTM (TIM0_7.11) for trimming purpose
- Over-run indication of consecutive unserviced wake-up triggers

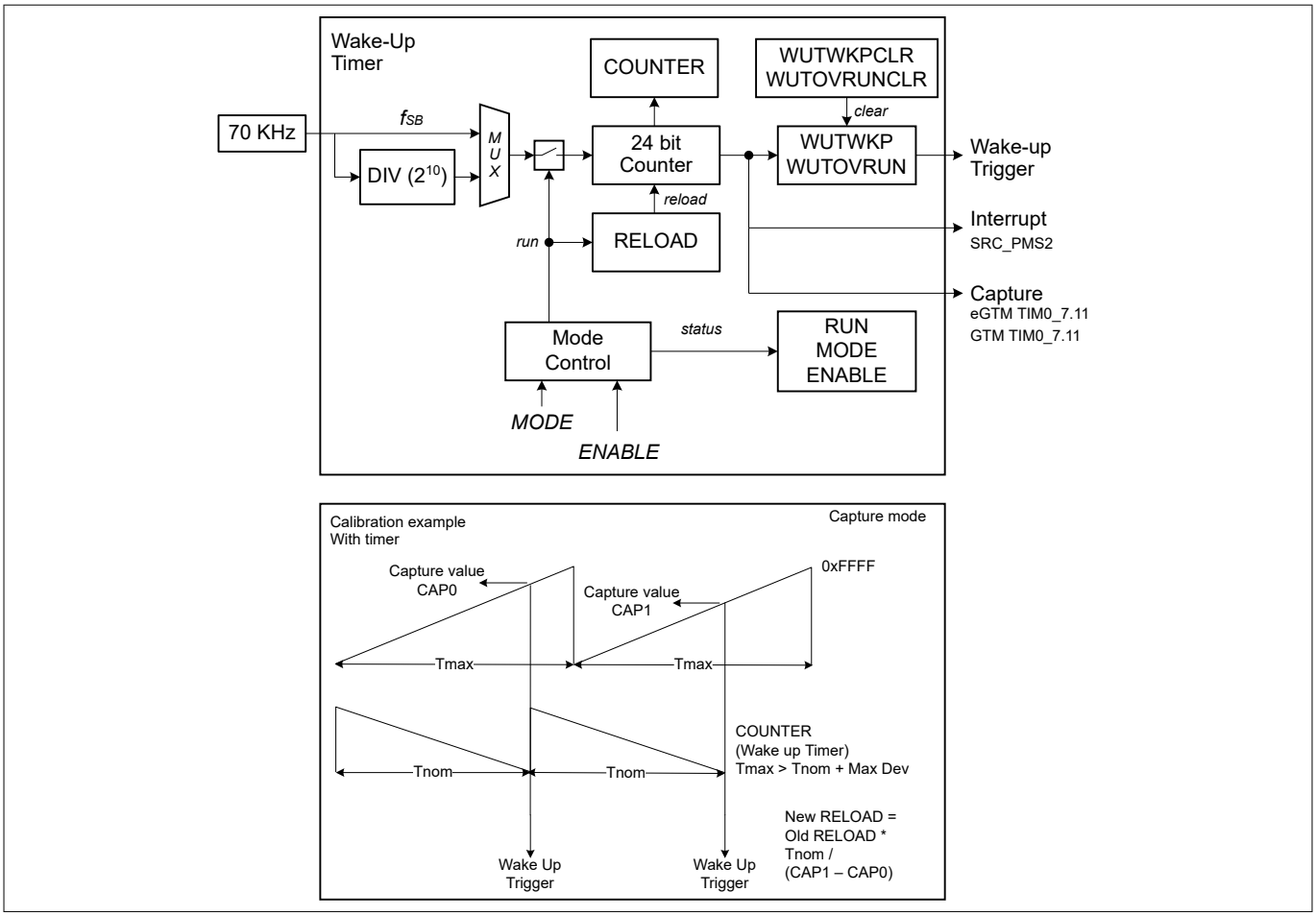


Figure 803 Wake-up timer (WUT)

Table 1437 Wake-up timer operation and modes

WUT_CON.ENABLE	WUT_CON.MODE	Mode description
0_B	X_B	WUT is disabled and counter is stopped. WUT_CON.RELOAD, MODE and DIV value shall be updated only when WUT is not enabled. WUT_STAT.COUNTER and .RUN read 0. In addition,WAKEUP_STAT0.WUTWKP and .WUTOVRUN flags are not updated and retain earlier value irrespective of WUTWKEN setting and should be cleared explicitly by the user.

(table continues...)

Table 1437 (continued) Wake-up timer operation and modes

WUT_CON.ENABLE	WUT_CON.MODE	Mode description
1_B	00_B	Software Auto Reload mode : WUT starts running when WUT_CON.ENABLE = 1 and .MODE = 0 is set. WAKEUP_CON0.WUTWKEN = 1 must be set to activate system wake-up from STANDBY state. WUT_STAT.RUN bit is set indicating that WUT timer is currently running. WUT_STAT.COUNTER bit-field indicates the actual counter value. On counter underflow, WUT is automatically reloaded with the WUT_CON.RELOAD value. During STANDBY, WUT underflow triggers system wake-up if WAKEUP_CON0.WUTWKEN is set and WAKEUP_STAT0.WUTWKP flag is set. During Run, Idle or Sleep modes, WUT underflow triggers an interrupt request and WAKEUP_STAT0.WUTWKP flag is set. WUT_CON.RELOAD and DIV value shall not be updated in this state. On wake-up, the WAKEUP_STAT0.WUTWKP flag shall be cleared by WAKEUP_CLR0.WUTWKPCLR bit. In case of un-serviced consecutive counter underflow events, WAKEUP_STAT0.WUTOVRUN flag is set to indicate an over-run wake-up event. Interrupt over-run event can be detected via SRC.IOV bit during RUN mode.
1_B	01_B	STANDBY0 PMS0 Auto Stop mode: The mode is selected by setting WUT_CON.ENABLE = 1, .MODE = 1. WAKEUP_CON0.WUTWKEN = 1 is set to activate system wake-up from STANDBY state. WUT starts running only when STANDBY0 mode is entered. On counter underflow, WUT stops running and the wake-up of system is triggered and WAKEUP_STAT0.WUTWKP flag is set. WUT starts running again on the next STANDBY0 mode entry. The intention is to have the timer running only during the STANDBY0 state. WUT_STAT.COUNTER reloads WUT_CON.RELOAD value on a wake-up. WUT_STAT.RUN reads 0 after a wake-up. WUT_CON.RELOAD and DIV value shall not be updated in this state. On wake-up, the WAKEUP_STAT0.WUTWKP flag shall be cleared by WAKEUP_CLR0.WUTWKPCLR bit in RUN mode. In case of wake-up to STANDBY1 state, the WUTWKP flag is cleared by SCR.WKPCLR clear flag event. In case system was woken up by other wake-up triggers while WUT was still running, an interrupt request is generated on WUT underflow.

(table continues...)

Table 1437 (continued) Wake-up timer operation and modes

WUT_CON.ENABLE	WUT_CON.MODE	Mode description
1_B	10_B	STANDBY1 PMS1 Auto Stop mode: The mode is selected by setting WUT_CON.ENABLE = 1, .MODE = 2. WAKEUP_CON0.WUTWKEN = 1 is set to activate system wake-up from STANDBY state. WUT starts running only when STANDBY1 mode is entered. On counter underflow, WUT stops running and the wake-up of system to RUN mode is triggered and WAKEUP_STAT0.WUTWKP flag is set. WUT starts running again on the next STANDBY1 mode entry. The intention is to have the timer running only during the STANDBY1 state. WUT does not run in either STANDBY0 state. In case of STANDBY0 entry, WUT freezes at the last count value and starts counting again from that value on STANDBY1 mode re-entry. WUT_STAT.COUNTER reloads WUT_CON.RELOAD value on a wake-up. WUT_STAT.RUN reads 0 after a wake-up. WUT_CON.RELOAD and DIV value shall not be updated in this state. On wake-up, the WAKEUP_STAT0.WUTWKP flag shall be cleared by WAKEUP_CLR0.WUTWKPCLR bit. In case system was woken up by other wake-up triggers while WUT was still running, an interrupt request is generated on WUT underflow.
1_B	11_B	Reserved

Table 1438 WUTWKEN configuration and mode transitions

WUTWKEN	Intended system mode transition	Intended WUT modes to be used
0 1 B	STANDBY0 from/to RUN	Software Auto reload mode STANDBY0 Auto Stop mode
1 0 B	STANDBY0 from/to STANDBY1	Software Auto reload mode STANDBY0 Auto Stop mode - Clear WUTWKP flag via SCR.WKPCLR bit in SCR in STANDBY1 mode to be able to go back to STANDBY0 mode.
1 1 B	STANDBY1 from/to RUN	Software Auto reload mode STANDBY1 Auto Stop mode - Only transition between RUN and STANDBY1 mode is allowed in this mode.
1 1 B	STANDBY0 from/to STANDBY1 from/to RUN	Software Auto reload mode - In STANDBY0 mode, on first wake-up, transition happens to STANDBY1. Consecutive Wake-up event may trigger a transition to RUN mode. Both WUTWKP and WUTOVRUN flags would be set in this case.

In case of timer underflow, an interrupt is issued on the interrupt node SRC_PMSx (WUT). Wake-up Timer reload value maybe trimmed during Run mode by comparing the time stamp on a WUT underflow captured by a GTM-TIM or eGTM based on a more precise clock as shown in the figure above. This allows to compensate on short term the 70 kHz (fSB) clock source variations owing to technology, voltage and temperature.

36.3.7.7 RTC - real time clock

The PMS integrates a real time clock (RTC) running at 32.768 kHz, with temperature trimming and parametric characteristics as specified in the datasheet.

The RTC in the PMS provides a free running 64-bit global system timer, which keeps running through all resets except the LVDPMS0 reset. All 64 bits of the counter can be read synchronously.

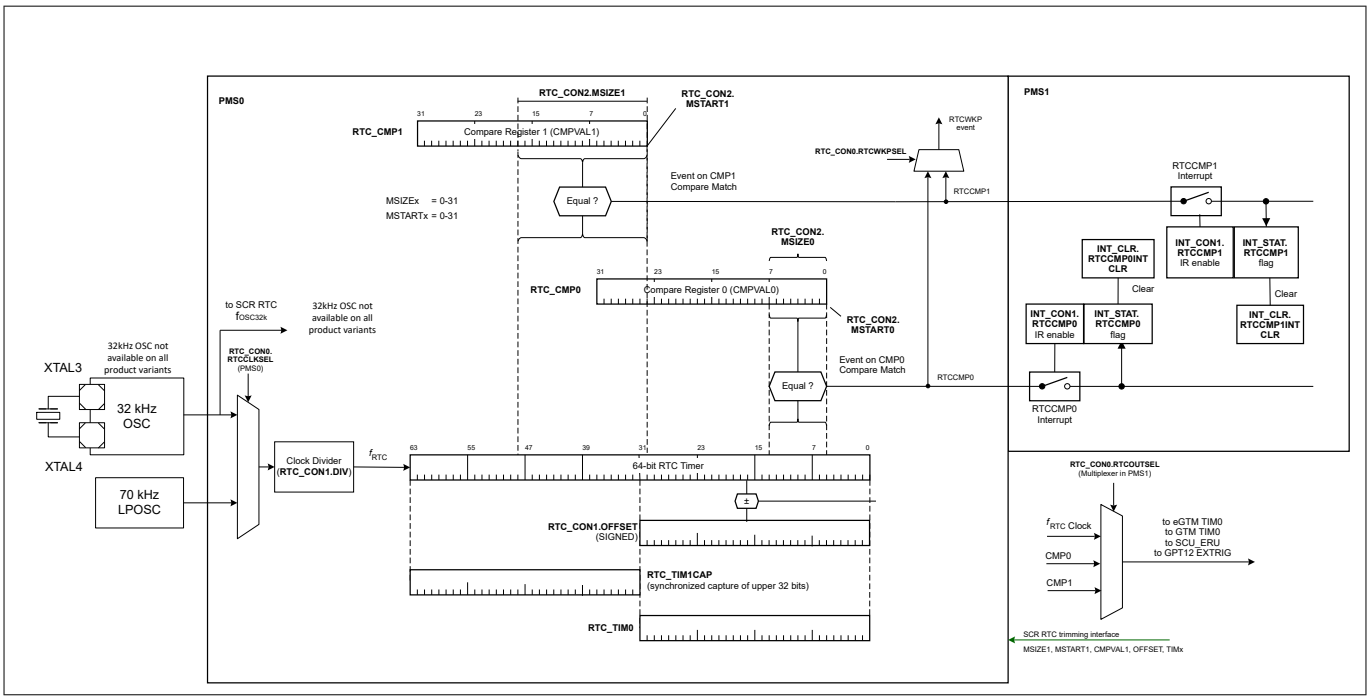


Figure 804 General block diagram of the RTC module

The clock source of the RTC can be selected between a 32 kHz oscillator (also used by the RTC of the SCR subsystem) and the 70 kHz low-power oscillator of the PMS, using the RTC_CON0.RTCCLKSEL configuration bit. On certain product variants, the 32kHz oscillator is not available. The clock of the RTC can be enabled or disabled using the RTC_CON0.RTCEN bit. The counter period can be configured by adjusting the clock divider with the RTC_CON1.DIV bit-field (16-bit). The division factor is DIV+1, that is if the RTC input frequency is 32.768 kHz, use for example RTC_CON1.DIV=$7FFF_H$ (division by 32768) to generate a 1 second time base.

Due to the 64-bit width of the RTC timer, it is not possible to read its entire content with one instruction. It needs to be load with two load instructions. Since the timer would continue to count between the two load operations, there is a chance that the two values read are not consistent (due to a possible overflow from the low part of the timer to the high part between the two read operations). To enable a synchronous and consistent reading of the RTC content, a capture register RTC_TIM1CAP is implemented. It latches the content of the high part of the RTC each time when the register RTC_TIM0 is read. Thus, RTC_TIM1CAP holds the upper value of the timer at exactly the same time when the lower part is read. The second read operation would then read the content of the RTC_TIM1CAP to get the complete timer value.

The content of the 64-bit RTC timer can be compared against the content of two compare values stored in the RTC_CMP0 and RTC_CMP1 registers. Service interrupts can be generated on a compare match of the RTC timer with the RTC_CMP0 or RTC_CMP1 registers and can be enabled via the INT_CON2.RTCCMP0 respectively INT_CON2.RTCCMP1 bits. A configurable number of bits can be selected from the RTC_CMP0 and RTC_CMP1 registers for the compare operation, by programming the RTC_CON2.MSIZE0 respectively MSIZE1 bit-fields. As shown in the previous figure, only a sub-set of the RTC_CMP0 and RTC_CMP1 registers, as defined in the MSIZE0 and MSIZE1, will then be selected for the compare match.

Two parameters are programmable for the compare operation:

1. RTC_CON2.MSIZE0/MSIZE1: The number of bits which are selected from the registers RTC_CMP0/ RTC_CMP1 for the compare operation, can be programmed from 0 to 31
2. RTC_CON2.MSTART0/MSTART1: The first bit location in the 64-bit timer that is taken for the compare operation can be programmed from 0 to 31. Note that by setting MSTART0/1 to 31 and MSIZE0/1 to 31, the bits 31 to 62 from RTC_CMP0/1 will be used for the compare operation (bit 63 of the RTC timer is not included in the compare operation).

These programming capabilities enable a very flexible compare functionality. It is possible to detect bit transitions of a single bit n (n = 0 to 31) within the 64-bit RTC by setting MSIZE = 0 and MSTART = n

The example illustrated in the previous figure shows a compare operation in which following parameters are programmed:

- MSIZE1 = $10001_B = 17_D$; MSTART1 = $11111_B = 31_D$
- MSIZE0 = $00111_B = 7_D$; MSTART0 = $00110_B = 6_D$

Upon a compare match event on the RTC_CMP0 or RTC_CMP1, an interrupt is triggered if enabled in INT_CON2.RTCCMP0 or INT_CON2.RTCCMP1.

Upon a compare match event on the RTC_CMP0 or RTC_CMP1, a wake-up request is triggered (RTCWKP) in the PMS0 standby domain, issuing a wake-up to STANDBY1 or to RUN mode depending on the configuration of WAKEUP_CON0.RTCWKEN bit. The wake-up request is selectable between RTC_CMP0 and RTC_CMP1 via RTC_CON0.RTCWKPSEL configuration bit.

A trimming of the RTC clock can be done by the application software, using the RTC_CON1.OFFSET bit-field, in order to correct the crystal frequency accuracy as needed by the application. The application configures the RTC_CMP1 compare register and MSIZE1, MSTART1 parameters, to trigger a compare match event whenever the application needs to check and apply the trimming. Then the application programs the required offset in RTC_CON1.OFFSET (signed value). The offset correction is then only applied at the next RTC_CMP0 compare match (as configured using RTC_CMP0, MSTART0, and MSIZE0).

The CMP0 match shall be used to generate RTC timer interrupt and CMP1 match interrupt to trim the crystal based on temperature. It is recommended that CMP1 match is less frequent than CMP0 match and therefore CMP1 compare match period shall have a higher value than CMP0 compare match period. Likewise it is expected that CMP0 compare match period shall have a higher value than OFFSET. As OFFSET representing the periodic crystal trim value with temperature deviations is expected to have a much smaller value than RTC timer tick period triggered on CMP0 interrupt. If OFFSET has larger values, the effects of negative OFFSET application need to be considered as shown in the figure below. The RTC configuration of MSIZE1, MSTART1 and CMPVAL1 can also be configured by the SCR for RTC trimming in STANDBY1 mode. f_{RTC} and RTC timer values are also provided as inputs to SCR.

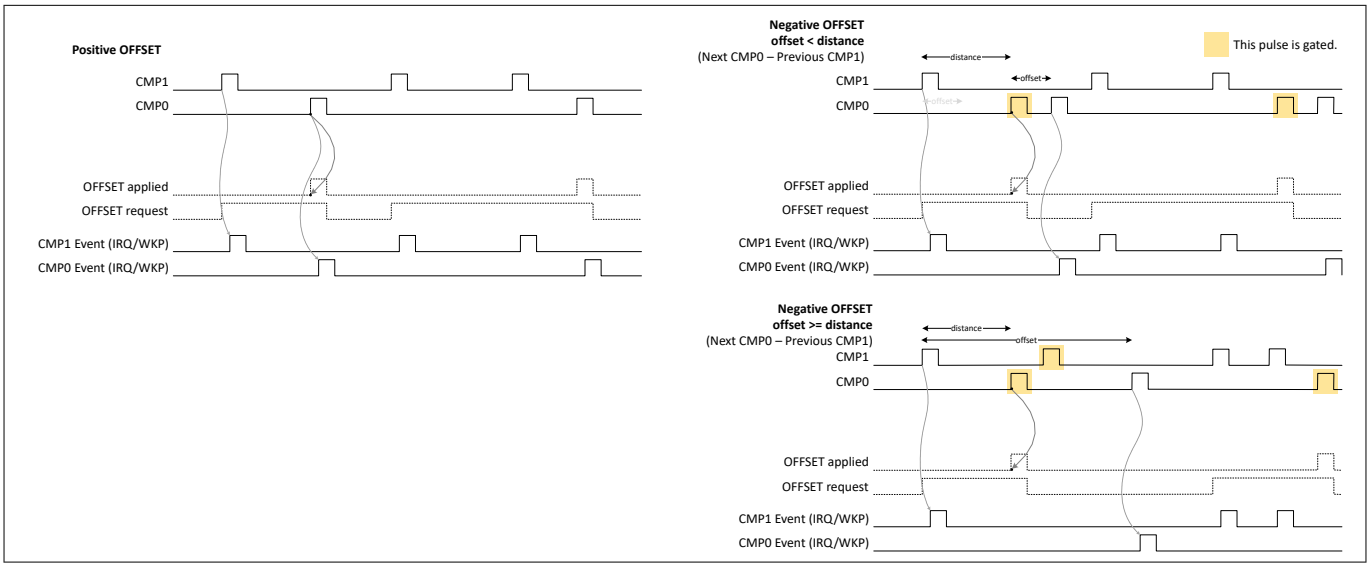


Figure 805 OFFSET application on running RTC timer

Behavior on a positive OFFSET correction:

- OFFSET is triggered on a CMP1 match but the offset addition to the actual timer value is done at the consequent CMP0 match

Behavior on a negative OFFSET correction:

- OFFSET is triggered on a CMP1[x] match but the offset subtraction to the actual timer value is done at the consequent CMP0[y] match
- CMP0[y] event is suppressed and interrupt/wake-up is triggered on next CMP0[y+1] match. This is because once negative OFFSET is applied, CMP0 match would be again triggered spuriously after the OFFSET time interval is expired
- If a CMP1[x+1] match comes before the latter CMP0[y+1] match, it is also suppressed and no OFFSET correction takes place at that CMP1[x+1] match

The output of the RTC timer can be selected using the RTC_CON0.RTCOUTSEL bit-field between the following three signals:

- The clock of the RTC, f_{RTC}
- The CMP0 compare output
- The CMP1 compare output

This output is then routed to eGTM, GTM, SCU_ERU, and GPT12 modules.

When using the 32 kHz oscillator as input for the RTC (not available on certain product variants), this oscillator can be further configured using the bit-fields from register RTC_CON0. The oscillator is enabled through RTC_CON0.RTCOSCEN. After enabling the oscillator, a wait time of maximum 5 seconds must be considered, for the oscillator to start up, before switching the RTC clock source to 32 kHz using RTC_CON0.RTCCLKSEL. An oscillator amplitude regulation can be enabled by setting the bit RTC_CON0.REGEN. As a result, the oscillation amplitude at the *XTPADIN* pin is approximately 0.5 V, independent of the gain settings and on board elements. The gain of the oscillator circuit can be configured via the RTC_CON0.GAINSEL setting, with a default setting of 01_B, that is a gain setting of 4x.

36.3.8 Die temperature sensor (DTSPMS)

The die temperature sensor in the PMS (DTSPMS) indicates the local junction temperature in the PMS subsystem. It is one of the multiple die temperature sensor (DTS) instances of TC4xx, which are distributed across the die and described in more detail in the VTMON functional block chapter.

The DTS periodically generates a measurement result that can be converted into the current die temperature at the location of the sensor. In the event of thermal limit violations, alarms can be generated to support thermal performance throttling operations. Each DTS operates within the T_{SR} temperature range specified in the data sheet, with an accuracy of T_{NL}+T_{CALACC} (see data sheet parameter values). The temperature sensing resolution is 0.22 K/code, with a maximum conversion time of not more than the value of the t_M parameter documented in the data sheet. Gain and offsets errors are compensated by trimming executed by start-up software, and systematic non-linearity is linearized through digital post-processing.

The analog circuitry for the DTS is enabled by setting the DTS_CON.EN register bit. When the DTS_CON.START register bit is set, temperature measurements will be carried out continuously during RUN/SLEEP modes, with the latest measurement result being updated in the DTS_STAT.RESULT bit-field. Both the DTS_CON.EN and DTS_CON.START bits can be set concurrently, though the first conversion will take slightly longer for the analog circuitry to be properly initialized. The DTS_STAT.DTSREADY register bit is set when the analog circuitry has finished the first conversion and only then the alarms are forwarded to the SMU.

The DTSPMS and corresponding registers are not affected by a warm PORST, system or application reset; consequently the temperature result in the DTS_STAT register, from an earlier conversion, is available for immediate use after any warm reset. All other DTS instances in VTMON (except DTSPMS) are reset upon an application reset.

The DTS result DTS_STAT.RESULT can be converted to temperature by the following transfer function:

T[°C] = DTS_STAT.RESULT / g_{nom}-273.15, where g_{nom} = 4.781 Codes/K

A peak temperature detector records the highest temperature measurement result in DTS_STAT.PEAK since the last reset operation. This can be reset by writing a '1' into the DTS_CON.PEAKDETRST register field.

Upper and lower thermal warnings with adjustable threshold, hysteresis, and filters are supported and are intended for thermal throttling and control functions. In addition, a thermal-shutdown warning is included, to trigger a reset (if configured in the SMU as an alarm reaction) during thermal runaway scenarios. All warnings can be triggered only when the DTS_CON.WRNEN bit is set. The DTS_CON.WRNEN bit is disabled per default, such that alarms are disabled after start-up.

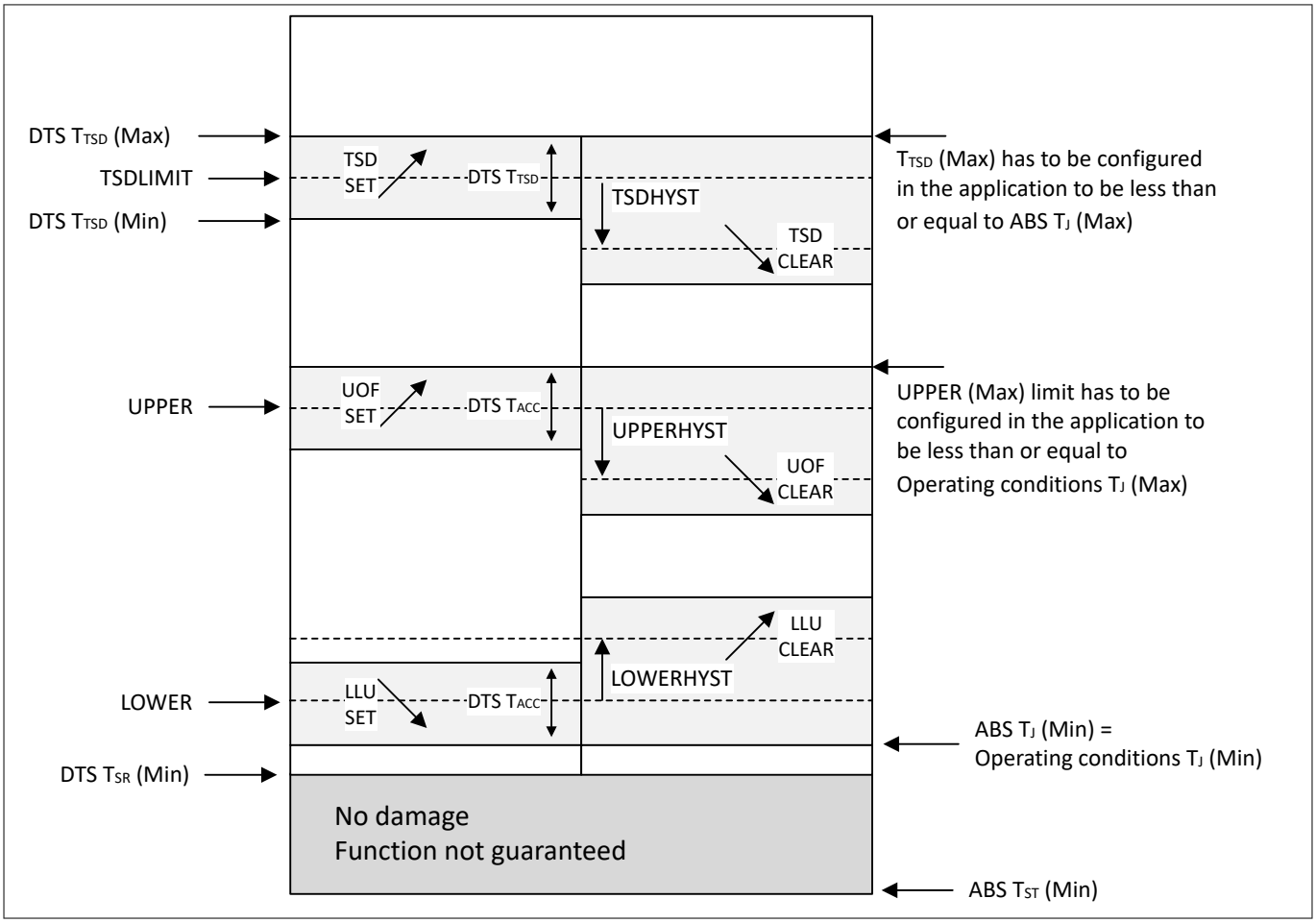


Figure 806 DTS alarm generation

The functionality of all warnings is summarized in the following. Die temperature thermal shutdown, upper, and lower limits are configured in the DTS_LIM2.TSDLIMIT, DTS_LIM1.UPPER, and DTS_LIM1.LOWER register bits respectively. Upon violation of these limits, DTS_STAT.TSD, DTS_STAT.UOF, and DTS_STAT.LLU status bits are set and alarms are forwarded to the SMU and CSRM. After start-up, the DTS limits have to be re-configured appropriately, depending on the application, before alarm reactions from SMU or CSRM are activated. Only when a new DTS conversion result is available, the DTS comparators are consequently triggered to check the actual DTS_STAT.RESULT against the thermal shutdown, upper, and lower limits.

The set criteria for all warnings also allows the implementation of a filter, whereby a warning will be triggered only when the configured limit is crossed for a number of consecutive temperature measurements (that is limit violations). The DTS_STAT.TSD warning is triggered when DTS_STAT.RESULT exceeds DTS_LIM2.TSDLIMIT for DTS_LIM2.TSDFILT + 1 consecutive conversions. The DTS_STAT.UOF warning is triggered when DTS_STAT.RESULT exceeds DTS_LIM1.UPPER for DTS_CON.UPPERFILT + 1 consecutive conversions. The DTS_STAT.LLU warning is triggered when DTS_STAT.RESULT is below DTS_LIM1.LOWER for DTS_CON.LOWERFILT + 1 consecutive conversions.

The clear criteria for DTS_STAT.TSD warning is satisfied when DTS_STAT.RESULT is below DTS_LIM2.TSDLIMIT - DTS_LIM2.TSDHYST. The clear criteria for DTS_STAT.UOF warning is satisfied when DTS_STAT.RESULT is below DTS_LIM1.UPPER - DTS_LIM1.UPPERHYST. Similarly, the clear criteria for DTS_STAT.LLU warning is satisfied when DTS_STAT.RESULT exceeds DTS_LIM1.LOWER + DTS_LIM1.LOWERHYST.

36.3.9 Standby controller extension RAM (XRAM)

The SCR XRAM is also protected by PMS access protection mechanisms.

The following tables show the memory overview and the according access protection mechanisms employed.

Table 1439 SCR XRAM memory overview

Short name	Long name	Address	Access mode	
			Read	Write
PMS_XRAM	Standby controller extension RAM (8000_H byte)	$F0240000_H$	P	SV, E, P

Table 1440 Access mode glossary

Keyword	Description
P	Access protection using APU-P registers.
E	Access protection using PROT register PMS_PROTE.
SV	Access only when supervisor mode is active on the interconnect.

36.4 Registers

36.4.1 Register overview - access mode glossary

Table 1441 Register overview - access mode glossary

Keyword	Description
CSE	Access protection using PROT register PROTCSE.
E	Access protection using PROT register PROTE.
SE	Access protection using PROT register PROTSE.
APU-P	Protection group consisting of registers ACCEN_WRA, ACCEN_WRB, ACCEN_RDA, ACCEN_RDB, ACCEN_VM, ACCEN_PRS.
P	Access protection using *APU-P* registers.
APU-Pcs	Protection group consisting of registers ACCENCS_WRA, ACCENCS_WRB, ACCENCS_RDA, ACCENCS_RDB, ACCENCS_VM, ACCENCS_PRS.
Pcs	Access protection using *APU-Pcs* registers.
U	No access restrictions.
PROT	Access restrictions as defined in the PROT register access rules.
BE	Always returns a Bus Error.
STP	Access only during start-up, before start-up protection is set (SMM_STCON.ST_PROT=1).
M	Indicates a module specific access condition. Refer to the register description for details of the specific access condition.

36.4.2 Registers overview - PMS (ascending offset address)

Table 1442 Registers overview - PMS (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PROTE	PROT register endinit	9000_H	U	SV, PROT	Application Reset	6044
PROTSE	PROT register safe endinit	9004_H	U	SV, PROT	Application Reset	6046
ID	Module identification register	9008_H	P	BE	Application Reset	6048
ACCEN_WRA	Write access enable register A	$900C_H$	U	SE, SV	Application Reset	6048
ACCEN_WRB	Write access enable register B	9010_H	U	SE, SV	Application Reset	6049
ACCEN_RDA	Read access enable register A	9014_H	U	SE, SV	Application Reset	6049
ACCEN_RDB	Read access enable register B	9018_H	U	SE, SV	Application Reset	6050

(table continues...)

Table 1442 (continued) Registers overview - PMS (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ACCEN_VM	VM access enable register	$901C_H$	U	SE, SV	Application Reset	6050
ACCEN_PRS	PRS access enable register	9020_H	U	SE, SV	Application Reset	6051
REGACCSTAT	Register access status register	9024_H	P	BE	LVDPMS2 Reset	6052
PROTCSE	PROT register cyber security endinit	9028_H	U	SV, PROT	Application Reset	6053
ACCENCS_WRA	Write access enable register A	$902C_H$	U	CSE, SV	Application Reset	6054
ACCENCS_WRB	Write access enable register B	9030_H	U	CSE, SV	Application Reset	6055
ACCENCS_RDA	Read access enable register A	9034_H	U	CSE, SV	Application Reset	6055
ACCENCS_RDB	Read access enable register B	9038_H	U	CSE, SV	Application Reset	6056
ACCENCS_VM	VM access enable register	$903C_H$	U	CSE, SV	Application Reset	6056
ACCENCS_PRS	PRS access enable register	9040_H	U	CSE, SV	Application Reset	6057
VMONP_TRKTRIM0	Tracking ADC trim register 0	9138_H	P	SV, E, P	See 6058	6058
PEAK_VDD	VDD peak voltage detection configuration register	9204_H	P	P	LVDPMS2 Reset	6060
PEAK_VDDEXT	VDDEXT peak voltage detection configuration register	9208_H	P	P	LVDPMS2 Reset	6060
PEAK_VDDEXTDC	VDDEXTDC peak voltage detection configuration register	$920C_H$	P	P	LVDPMS2 Reset	6061
PEAK_VDDSTAT	VDD peak voltage detection status register	9210_H	P	BE	LVDPMS2 Reset	6062
PEAK_VDDEXTSTAT	VDDEXT peak voltage detection status register	9214_H	P	BE	LVDPMS2 Reset	6063
PEAK_VDDEXTDCSTAT	VDDEXTDC peak voltage detection status register	9218_H	P	BE	LVDPMS2 Reset	6063
PEAK_VDDHISTSTAT	VDD peak voltage detection history status register	$921C_H$	P	BE	LVDPMS1 Reset	6064
PEAK_VDDEXTHISTSTAT	VDDEXT peak voltage detection history status register	9220_H	P	BE	LVDPMS1 Reset	6064

(table continues...)

Table 1442 (continued) Registers overview - PMS (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PEAK_VDDEXTDCHISTSTAT	VDDEXTDC peak voltage detection history status register	9224_H	P	BE	LVDPMS1 Reset	6065
DRMON_VDD	VDD drift monitoring configuration register	9228_H	P	P	LVDPMS2 Reset	6066
DRMON_VDDSTAT	VDD drift monitor status register	$922C_H$	P	BE	LVDPMS2 Reset	6066
DRMON_VDDEXT	VDDEXT drift monitoring configuration register	9230_H	P	P	LVDPMS2 Reset	6067
DRMON_VDDEXTSTAT	VDDEXT drift monitor status register	9234_H	P	BE	LVDPMS2 Reset	6067
DRMON_VDDEXTDC	VDDEXTDC drift monitoring configuration register	9238_H	P	P	LVDPMS2 Reset	6068
DRMON_VDDEXTDCSTAT	VDDEXTDC drift monitor status register	$923C_H$	P	BE	LVDPMS2 Reset	6068
DRMON_VDDM	VDDM drift monitoring configuration register	9240_H	P	P	LVDPMS2 Reset	6069
DRMON_VDDMSTAT	VDDM drift monitor status register	9244_H	P	BE	LVDPMS2 Reset	6069
DRMON_VDDPMS0	VDDPMS0 drift monitoring configuration register	9248_H	P	P	LVDPMS2 Reset	6070
DRMON_VDDPMS0STAT	VDDPMS0 drift monitor status register	$924C_H$	P	BE	LVDPMS2 Reset	6070
DRMON_VDDPMS2	VDDPMS2 drift monitoring configuration register	9250_H	P	P	LVDPMS2 Reset	6071
DRMON_VDDPMS2STAT	VDDPMS2 drift monitor status register	9254_H	P	BE	LVDPMS2 Reset	6071
VID_CON0	Voltage ID (VID) configuration register 0	9258_H	U	STP|TM	See 6072	6072
INT_CON2	Interrupt configuration register 2	9274_H	P	SV, E, P	See 6072	6072
INT_STAT	Interrupt status register	9278_H	P	SV, E, P	See 6075	6075
INT_CLR	Interrupt status clear register	$927C_H$	P	SV, E, P	See 6078	6078
PAD_CON	Pad configuration register	9280_H	P	SV, E, P	LVDANA Reset	6079
PAD_ESR2CON	ESR2 pad configuration register	9288_H	P	SV, E, P	See 6081	6081
HPOSCCON1	HPOSC configuration register 1	9294_H	P	SV, E, P	See 6082	6082
HPOSCTRIM	HPOSC trim register for dynamic trimming over temperature	$929C_H$	P	SV, E, P	See 6083	6083

(table continues...)

Table 1442 (continued) Registers overview - PMS (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
HPBGTRIM	HPBG trim register for dynamic trimming over temperature	$92A0_H$	P	SV, E, P	See 6084	6084
WAKEUP_CON0	Wake-up configuration register 0	$92AC_H$	P	SV, E, P	See 6085	6085
WAKEUP_STAT0	Wake-up status register 0	$92B0_H$	P	BE	See 6090	6090
WAKEUP_CLR0	Standby and wake-up status clear register 0	$92B4_H$	P	SV, E, P	See 6092	6092
WAKEUP_CON1	Wake-up configuration register 1	$92B8_H$	P	SV, E, P	See 6094	6094
WAKEUP_STAT1	Wake-up status register 1	$92BC_H$	P	BE	See 6096	6096
WAKEUP_CLR1	Standby and wake-up status clear register 1	$92C0_H$	P	SV, E, P	See 6098	6098
STANDBY_CON1	Standby configuration register 1	$92C8_H$	P	SV, E, P	LVDPMS1 Reset	6100
WUT_CON	Wake-up timer configuration register	$92CC_H$	P	SV, E, P	See 6101	6101
WUT_STAT	Wake-up timer status register	$92D0_H$	P	BE	See 6102	6102
SCR_CON0	SCR configuration register 0	$92D4_H$	P	SV, E, P	See 6103	6103
SCR_CON1	SCR configuration register 1	$92D8_H$	P	SV, E, P, M	LVDPMS1 Reset	6105
SCR_STAT	SCR status register	$92DC_H$	P	SV, E, P, M	LVDPMS1 Reset	6106
EVRC_STAT1	EVRC status register 1	$92E4_H$	P	BE	LVDPMS2 Reset	6108
EVRC_CON1	EVRC configuration register 1	$92F0_H$	P	SV, E, P	See 6108	6108
EVRC_CON2	EVRC configuration register 2	$92F4_H$	P	SV, E, P	See 6111	6111
EVRC_CON3	EVRC configuration register 3	$92F8_H$	P	SV, E, P	See 6112	6112
EVRC_CON4	EVRC configuration register 4	$92FC_H$	P	SV, E, P	See 6114	6114
EVRC_COEFF0	EVRC SD coefficient register 0	9300_H	P	SV, E, P	See 6115	6115
EVRC_COEFF1	EVRC SD coefficient register 1	9304_H	P	SV, E, P	See 6117	6117
EVRC_COEFF2	EVRC SD coefficient register 2	9308_H	P	SV, E, P	See 6120	6120
EVRC_COEFF3	EVRC SD coefficient register 3	$930C_H$	P	SV, E, P	See 6122	6122
EVRC_COEFF4	EVRC SD coefficient register 4	9310_H	P	SV, E, P	See 6125	6125

(table continues...)

Table 1442 (continued) Registers overview - PMS (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
EVRC_COEFF5	EVRC SD coefficient register 5	9314_H	P	SV, E, P	See 6127	6127
RTC_CON1	RTC configuration register 1	$931C_H$	P	SV, E, P	See 6128	6128
RTC_CON2	RTC configuration register 2	9320_H	P	SV, E, P	See 6129	6129
RTC_CMP0	RTC compare register 0	9324_H	P	SV, E, P	See 6130	6130
RTC_CMP1	RTC compare register 1	9328_H	P	SV, E, P	See 6131	6131
RTC_TIM0	RTC timer counter register 0	$932C_H$	P	BE	See 6131	6131
RTC_TIM1CAP	RTC timer capture register	9330_H	P	BE	LVDPMS2 Reset	6132
DTS_CON	Die temperature sensor control register	9334_H	U	E, P, SV	See 6132	6132
DTS_LIM1	Die temperature sensor limit register 1	9338_H	U	E, P, SV	LVDPMS1 Reset	6134
DTS_LIM2	Die temperature sensor limit register 2	$933C_H$	U	E, P, SV	LVDPMS1 Reset	6134
DTS_STAT	Die temperature sensor status register	9340_H	P	BE	LVDPMS1 Reset	6135
CLDRSTCON	Cold reset configuration register	9350_H	P	SV, E, P	LVDPMS1 Reset	6136
CLDRSTSTATCLR	Cold reset status clear register	9360_H	P	SV, E, P	LVDPMS2 Reset	6137
OTSS	OCDS trigger set select register	9364_H	P	P	See 6137	6137
OTSC1	OCDS trigger set control 1 register	$936C_H$	P	P	See 6138	6138
LVDPMS1USRINFO	Register to store user info in PMS1 domain and reset by LVDPMS1	9370_H	P	P	See 6139	6139
CLDPORSTUSRINFO	Register to store user info in PMS2 domain and reset by cold PORST	9374_H	P	P	See 6140	6140
WRMPORSTUSRINFO	Register to store user info in PMS2 domain and reset by warm PORST	9378_H	P	P	See 6140	6140

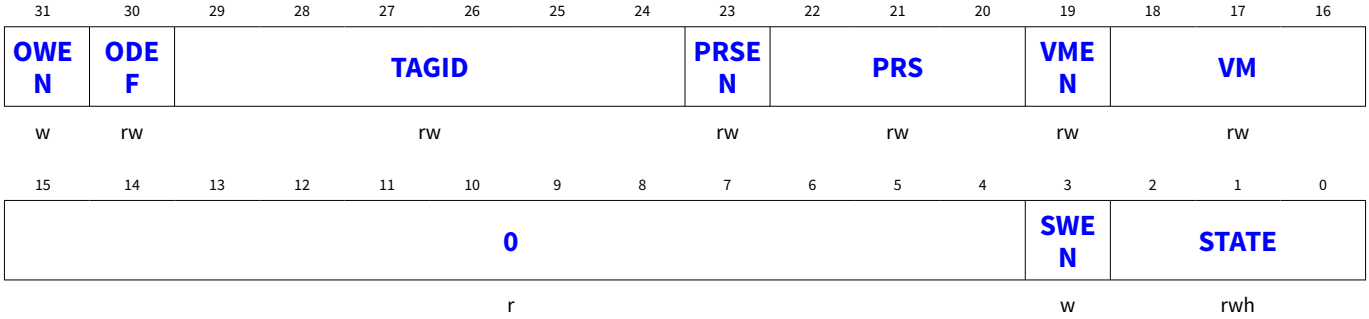
36.4.3 PROT register endinit

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner or CS master to update the PROT state as described in the PROT mechanism.

The PROTE register controls lock and unlock of the local endinit (E) protected control registers.

PROTE Offset address: 9000_H

PROT register endinit Application Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
VM	18:16	rw	**Virtual Machine definition for PROT owner** Defines the VM portion of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual Machine definition Enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection set definition for PROT owner** Defines the PRS portion of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.
PRSEN	23	rw	**Protection set definition enable for PROT owner** 0_B **Disable**: PRS is part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG ID definition for PROT owner** Defines the TAGID portion of the PROT owner. This field is ignored if ODEF=0_B.

(table continues...)

(continued)

Field	Bits	Type	Description
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner
OWEN	31	w	**Owner write enable** 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; should be written with 0.

Table 1443 Access mode restrictions of PROTE sorted by descending priority

Mode name	Access mode		Description
write 1 to .**SWEN**	rwh	STATE	
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

36.4.4 PROT register safe endinit

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner or CS master to update the PROT state as described in the PROT mechanism.

The PROTSE register controls lock and unlock of the local safe endinit (SE) protected control registers.

PROTSE Offset address: 9004_H

PROT register safe endinit Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OWEN	**ODEF**	**TAGID**						**PRSEN**	**PRS**			**VMEN**	**VM**		
w	rw	rw						rw	rw			rw	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												**SWEN**	**STATE**		
r												w	rwh		

Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
VM	18:16	rw	**Virtual Machine definition for PROT owner** Defines the VM portion of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual Machine definition Enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection set definition for PROT owner** Defines the PRS portion of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.
PRSEN	23	rw	**Protection set definition enable for PROT owner** 0_B **Disable**: PRS is part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG ID definition for PROT owner** Defines the TAGID portion of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner
OWEN	31	w	**Owner write enable** 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; should be written with 0.

Table 1444 Access mode restrictions of PROTSE sorted by descending priority

Mode name	Access mode		Description
write 1 to .**SWEN**	rwh	STATE	
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

36.4.5 Module identification register

ID Offset address: 9008_H

Module identification register Application Reset value: $00E8\ C001_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MOD_NUM															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MOD_TYPE								MOD_REV							
r								r							

Field	Bits	Type	Description
MOD_REV	7:0	r	**Module revision** Indicates the revision number of the implementation. …
MOD_TYPE	15:8	r	**Module type** This bit field is fixed coded as $C0_H$. It defines a 32-bit module.
MOD_NUM	31:16	r	**Module number** This bit field defines the module identification number. The identification number for the PMS is $00E8_H$.

36.4.6 Write access enable register A

This register provides controls to enable or disable write accesses for transactions with TAG IDs corresponding to the numbered bits.

ACCEN_WRA Offset address: $900C_H$

Write access enable register A Application Reset value: $1000\ 0003_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Write access enable for Master TAG ID q** This bit enables write access to the access protected region for transactions with the Master TAG ID q. 0_B Disabled for write access 1_B Enabled for write access

36.4.7 Write access enable register B

This register provides controls to enable or disable write accesses for transactions with TAG IDs corresponding to the numbered bits.

ACCEN_WRB Offset address: 9010_H

Write access enable register B Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Write access enable for Master TAG ID q** This bit enables write access to the access protected region for transactions with the Master TAG ID q. 0_B Disabled for write access 1_B Enabled for write access
0	31:8	r	**Reserved** Read as 0; should be written with 0.

36.4.8 Read access enable register A

This register provides controls to enable or disable read accesses for transactions with TAG IDs corresponding to the numbered bits.

ACCEN_RDA Offset address: 9014_H

Read access enable register A Application Reset value: $FFFF\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Read access enable for Master TAG ID q** This bit enables read access to the access protected region for transactions with the Master TAG ID q. 0_B Disabled for read access 1_B Enabled for read access

36.4.9 Read access enable register B

This register provides controls to enable or disable read accesses for transactions with TAG IDs corresponding to the numbered bits.

ACCEN_RDB Offset address: 9018_H

Read access enable register B Application Reset value: $0000\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Read access enable for Master TAG ID q** This bit enables read access to the access protected region for transactions with the Master TAG ID q. 0_B Disabled for read access 1_B Enabled for read access
0	31:8	r	**Reserved** Read as 0; should be written with 0.

36.4.10 VM access enable register

This register provides separate controls to enable or disable read and write accesses to registers for transactions with VM corresponding to the numbered bits. Bits WR0 to WR7 control write access and bits RD0 to RD7 control read access.

ACCEN_VM Offset address: $901C_H$

VM access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for VM ID q** This bit enables read access to the access protected region for transactions with the VM ID q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for VM ID q** This bit enables write access to the access protected region for transactions with the VM ID q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

36.4.11 PRS access enable register

This register provides separate controls to enable or disable read and write accesses to registers for transactions with PRS corresponding to the numbered bits. Bits WR0 to WR7 control write access and bits RD0 to RD7 control read access.

ACCEN_PRS Offset address: 9020_H

PRS access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for PRS q** This bit enables read access to the access protected region for transactions in the PRS q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for PRS q** This bit enables write access to the access protected region for transactions in the PRS q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

36.4.12 Register access status register

REGACCSTAT Offset address: 9024_H

Register access status register LVDPMS2 Reset value: $0000\ 0002_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0														FIFOEMPTY	FIFOFULL
r														rh	rh

Field	Bits	Type	Description
FIFOFULL	0	rh	**RAI FIFO full status** Full status of the FIFO in the register access interface (RAI). This status could be reached after writing 32 times to PMS0 registers within 14 µs. PMS0 registers are indicated by an "LVDPMS0 Reset" or an "LVDANA Reset" type in the reset table of the respective register. 0_B RAI FIFO not full. 1_B RAI FIFO is full.
FIFOEMPTY	1	rh	**RAI FIFO empty status** Empty status of the FIFO in the register access interface (RAI). This status gets set when all previous write accesses to PMS0 registers have been executed. PMS0 registers are indicated by an "LVDPMS0 Reset" type in the reset table of the respective register. It is recommended to check the FIFOEMPTY status before writing into PMS0 registers. 0_B RAI FIFO not empty. 1_B RAI FIFO is empty.

(table continues...)

(continued)

Field	Bits	Type	Description
0	31:2	r	**Reserved** Read as 0.

36.4.13 PROT register cyber security endinit

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner or CS master to update the PROT state as described in the PROT mechanism.

The PROTCSE register controls lock and unlock of the local cyber security endinit (CSE) protected control registers.

PROTCSE Offset address: 9028_H

PROT register cyber security endinit Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OWEN	ODEF	TAGID						PRSEN	PRS			VMEN	VM		
w	rw	rw						rw	rw			rw	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												SWEN	STATE		
r												w	rwh		

Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
VM	18:16	rw	**Virtual Machine definition for PROT owner** Defines the VM portion of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.

(table continues...)

(continued)

Field	Bits	Type	Description
VMEN	19	rw	**Virtual Machine definition Enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection set definition for PROT owner** Defines the PRS portion of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.
PRSEN	23	rw	**Protection set definition enable for PROT owner** 0_B **Disable**: PRS is part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG ID definition for PROT owner** Defines the TAGID portion of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner
OWEN	31	w	**Owner write enable** 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; should be written with 0.

Table 1445 Access mode restrictions of PROTCSE sorted by descending priority

Mode name	Access mode		Description
write 1 to .**SWEN**	rwh	STATE	
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

36.4.14 Write access enable register A

This register provides controls to enable or disable write accesses for transactions with TAG IDs corresponding to the numbered bits.

ACCENCS_WRA Offset address: $902C_H$

Write access enable register A Application Reset value: $1000\ 0003_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Write access enable for Master TAG ID q** This bit enables write access to the access protected region for transactions with the Master TAG ID q. 0_B Disabled for write access 1_B Enabled for write access

36.4.15 Write access enable register B

This register provides controls to enable or disable write accesses for transactions with TAG IDs corresponding to the numbered bits.

ACCENCS_WRB Offset address: 9030_H

Write access enable register B Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Write access enable for Master TAG ID q** This bit enables write access to the access protected region for transactions with the Master TAG ID q. 0_B Disabled for write access 1_B Enabled for write access
0	31:8	r	**Reserved** Read as 0; should be written with 0.

36.4.16 Read access enable register A

This register provides controls to enable or disable read accesses for transactions with TAG IDs corresponding to the numbered bits.

ACCENCS_RDA Offset address: 9034_H

Read access enable register A Application Reset value: $FFFF\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Read access enable for Master TAG ID q** This bit enables read access to the access protected region for transactions with the Master TAG ID q. 0_B Disabled for read access 1_B Enabled for read access

36.4.17 Read access enable register B

This register provides controls to enable or disable read accesses for transactions with TAG IDs corresponding to the numbered bits.

ACCENCS_RDB Offset address: 9038_H

Read access enable register B Application Reset value: $0000\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Read access enable for Master TAG ID q** This bit enables read access to the access protected region for transactions with the Master TAG ID q. 0_B Disabled for read access 1_B Enabled for read access
0	31:8	r	**Reserved** Read as 0; should be written with 0.

36.4.18 VM access enable register

This register provides separate controls to enable or disable read and write accesses to registers for transactions with VM corresponding to the numbered bits. Bits WR0 to WR7 control write access and bits RD0 to RD7 control read access.

ACCENCS_VM Offset address: $903C_H$

VM access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for VM ID q** This bit enables read access to the access protected region for transactions with the VM ID q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for VM ID q** This bit enables write access to the access protected region for transactions with the VM ID q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

36.4.19 PRS access enable register

This register provides separate controls to enable or disable read and write accesses to registers for transactions with PRS corresponding to the numbered bits. Bits WR0 to WR7 control write access and bits RD0 to RD7 control read access.

ACCENCS_PRS Offset address: 9040_H

PRS access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for PRS q** This bit enables read access to the access protected region for transactions in the PRS q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for PRS q** This bit enables write access to the access protected region for transactions in the PRS q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

36.4.20 Tracking ADC trim register 0

VMONP_TRKTRIM0 Offset address: 9138_H

Tracking ADC trim register 0 Reset values see: Table 1446

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0						0		VSSDCHSALARMLPF0		VDDEXTDCALARMLPF		VDDEXTALARMLPF		VDDALARMLPF	
r						r		rwr		rw		rw		rw	

Field	Bits	Type	Description
VDDALARMLPF	1:0	rw	**VDD supply primary monitor alarm LPF coefficient** This bitfield configures the coefficient of the Low Pass Filter of the Primary Monitor tracking ADC counter value for SMU & CSRM alarms. y [k] = { y [k-1] * (1-a) } + { x [k] * a} ; y [k] is filter output; x [k] is ADC output a = {1 / (2 ^ LPF)}.If LPF = 0, the filter output is the same as ADC output. 00 : 2 clock delay 01 : 3 clock delay 10 : 5 clock delay 11 : 9 clock delay

(table continues...)

(continued)

Field	Bits	Type	Description
VDDEXTALARMLPF	3:2	rw	**VDDEXT supply primary monitor alarm LPF coefficient** This bitfield configures the coefficient of the Low Pass Filter of the Primary Monitor tracking ADC counter value for SMU & CSRM alarms. y [k] = { y [k-1] * (1-a) } + { x [k] * a} ; y [k] is filter output; x [k] is ADC output a = {1 / (2 ^ LPF)}.If LPF = 0, the filter output is the same as ADC output. 00 : 2 clock delay 01 : 3 clock delay 10 : 5 clock delay 11 : 9 clock delay
VDDEXTDCALARMLPF	5:4	rw	**VDDEXTDC supply primary monitor alarm LPF coefficient** This bitfield configures the coefficient of the Low Pass Filter of the Primary Monitor tracking ADC counter value for SMU & CSRM alarms. y [k] = { y [k-1] * (1-a) } + { x [k] * a} ; y [k] is filter output; x [k] is ADC output a = {1 / (2 ^ LPF)}.If LPF = 0, the filter output is the same as ADC output. 00 : 2 clock delay 01 : 3 clock delay 10 : 5 clock delay 11 : 9 clock delay
VSSDCHSALARMLPF	7:6	rw	**VSSDCHS supply primary monitor alarm LPF coefficient** This bitfield configures the coefficient of the Low Pass Filter of the Primary Monitor tracking ADC counter value for SMU & CSRM alarms. y [k] = { y [k-1] * (1-a) } + { x [k] * a} ; y [k] is filter output; x [k] is ADC output a = {1 / (2 ^ LPF)}.If LPF = 0, the filter output is the same as ADC output. 00 : 2 clock delay 01 : 3 clock delay 10 : 5 clock delay 11 : 9 clock delay
0	7:6, 9:8, 31:12	r	**Reserved** Read as 0; should be written with 0.

Table 1446 Reset values of VMONP_TRKTRIM0

Reset type	Reset value	Note
LVDPMS2 Reset	0000 0FFF$_H$	
Cold PORST	0000 0FFF$_H$	
After Boot-FW Value	0000 0FFF$_H$	

36.4.21 VDD peak voltage detection configuration register

PEAK_VDD Offset address: 9204_H

VDD peak voltage detection configuration register LVDPMS2 Reset value: $0000\ 1001_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESET								0							
w								r							

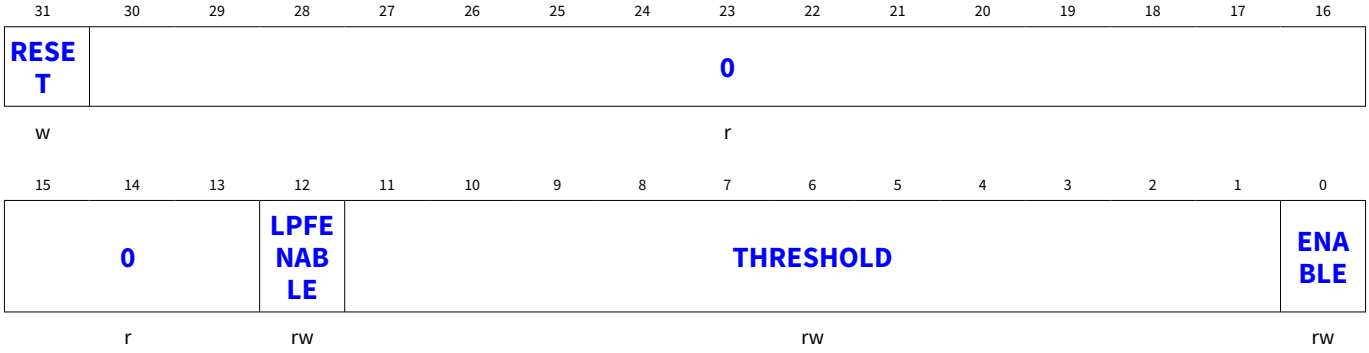
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			LPFENABLE	THRESHOLD											ENABLE
r			rw	rw											rw

Field	Bits	Type	Description
ENABLE	0	rw	**Enable/disable peak voltage detection for respective supply**
THRESHOLD	11:1	rw	**Threshold value configured by sw for peak voltage detection** Application software can configure a threshold value for detecting peak voltages higher than this threshold. Expressed as number of LSBs of the voltage monitoring ADC for the respective supply voltage.
LPFENABLE	12	rw	**Enable/disable the low-pass filtering of the recorded peak values** This bit allows to either apply the same low-pass filter as applied for the primary voltage monitor alarms to the recorded peak voltage value, or use the unfiltered ADC results instead.
RESET	31	w	**SW reset bit for peak voltage detection** Application software can restart the peak voltage detection cycle by setting this bit to 1, which resets the PEAK_VxxSTAT und PEAK_VxxHISTSTAT registers to their reset value for the respective Vxx voltage rail.
0	30:13	r	**Reserved** Read as 0; should be written with 0.

36.4.22 VDDEXT peak voltage detection configuration register

PEAK_VDDEXT Offset address: 9208_H

VDDEXT peak voltage detection configuration register LVDPMS2 Reset value: $0000\ 1001_H$



Field	Bits	Type	Description
ENABLE	0	rw	**Enable/disable peak voltage detection for respective supply**
THRESHOLD	11:1	rw	**Threshold value configured by sw for peak voltage detection** Application software can configure a threshold value for detecting peak voltages higher than this threshold. Expressed as number of LSBs of the voltage monitoring ADC for the respective supply voltage.
LPFENABLE	12	rw	**Enable/disable the low-pass filtering of the recorded peak values** This bit allows to either apply the same low-pass filter as applied for the primary voltage monitor alarms to the recorded peak voltage value, or use the unfiltered ADC results instead.
RESET	31	w	**SW reset bit for peak voltage detection** Application software can restart the peak voltage detection cycle by setting this bit to 1, which resets the PEAK_VxxSTAT und PEAK_VxxHISTSTAT registers to their reset value for the respective Vxx voltage rail.
0	30:13	r	**Reserved** Read as 0; should be written with 0.

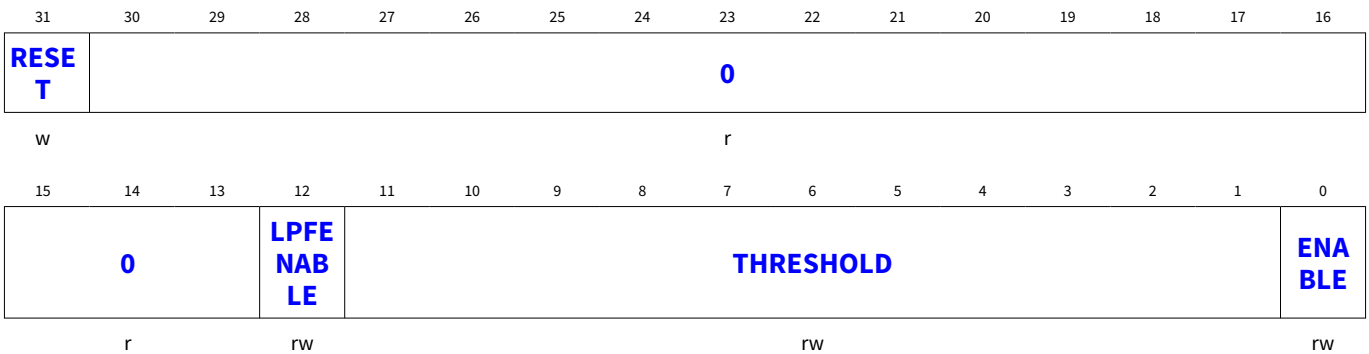
36.4.23 VDDEXTDC peak voltage detection configuration register

PEAK_VDDEXTDC

VDDEXTDC peak voltage detection configuration register

Offset address: $920C_H$

LVDPMS2 Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
ENABLE	0	rw	**Enable/disable peak voltage detection for respective supply**
THRESHOLD	11:1	rw	**Threshold value configured by sw for peak voltage detection** Application software can configure a threshold value for detecting peak voltages higher than this threshold. Expressed as number of LSBs of the voltage monitoring ADC for the respective supply voltage.
LPFENABLE	12	rw	**Enable/disable the low-pass filtering of the recorded peak values** This bit allows to either apply the same low-pass filter as applied for the primary voltage monitor alarms to the recorded peak voltage value, or use the unfiltered ADC results instead.
RESET	31	w	**SW reset bit for peak voltage detection** Application software can restart the peak voltage detection cycle by setting this bit to 1, which resets the PEAK_VxxSTAT und PEAK_VxxHISTSTAT registers to their reset value for the respective Vxx voltage rail.
0	30:13	r	**Reserved** Read as 0; should be written with 0.

36.4.24 VDD peak voltage detection status register

PEAK_VDDSTAT Offset address: 9210_H

VDD peak voltage detection status register LVDPMS2 Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
THRESCOUNT															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
THRESCOUNT					PEAKVOLT										
rh					rh										

Field	Bits	Type	Description
PEAKVOLT	10:0	rh	**Peak voltage detected value** This bit-field indicates the peak value detected for the respective supply voltage. Expressed as number of LSBs of the voltage monitoring ADC for the respective supply voltage.
THRESCOUNT	31:11	rh	**Detected sample count above configured threshold value** This bit-field indicates the number of voltage samples which were detected, after filtering, above the configured threshold value (PEAK_Vxx.THRESHOLD). During the reset phase if the peak detection is enabled for a supply rail and the the peak detection threslod is not configured (default 0x0) then THRESCOUNT must be ignored. The application software must configure the proper peak detection threshold and then reset the PEAK_VxxSTAT register.

36.4.25 VDDEXT peak voltage detection status register

PEAK_VDDEXTSTAT Offset address: 9214_H

VDDEXT peak voltage detection status register LVDPMS2 Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
THRESCOUNT															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
THRESCOUNT					PEAKVOLT										
rh					rh										

Field	Bits	Type	Description
PEAKVOLT	10:0	rh	**Peak voltage detected value** This bit-field indicates the peak value detected for the respective supply voltage. Expressed as number of LSBs of the voltage monitoring ADC for the respective supply voltage.
THRESCOUNT	31:11	rh	**Detected sample count above configured threshold value** This bit-field indicates the number of voltage samples which were detected, after filtering, above the configured threshold value (PEAK_Vxx.THRESHOLD). During the reset phase if the peak detection is enabled for a supply rail and the the peak detection threslod is not configured (default 0x0) then THRESCOUNT must be ignored. The application software must configure the proper peak detection threshold and then reset the PEAK_VxxSTAT register.

36.4.26 VDDEXTDC peak voltage detection status register

PEAK_VDDEXTDCSTAT Offset address: 9218_H

VDDEXTDC peak voltage detection status register LVDPMS2 Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
THRESCOUNT															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
THRESCOUNT					PEAKVOLT										
rh					rh										

Field	Bits	Type	Description
PEAKVOLT	10:0	rh	**Peak voltage detected value** This bit-field indicates the peak value detected for the respective supply voltage. Expressed as number of LSBs of the voltage monitoring ADC for the respective supply voltage.
THRESCOUNT	31:11	rh	**Detected sample count above configured threshold value** This bit-field indicates the number of voltage samples which were detected, after filtering, above the configured threshold value (PEAK_Vxx.THRESHOLD). During the reset phase if the peak detection is enabled for a supply rail and the the peak detection threslod is not configured (default 0x0) then THRESCOUNT must be ignored. The application software must configure the proper peak detection threshold and then reset the PEAK_VxxSTAT register.

36.4.27 VDD peak voltage detection history status register

PEAK_VDDHISTSTAT Offset address: $921C_H$

VDD peak voltage detection history status register LVDPMS1 Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
THRESCOUNT															
rh															

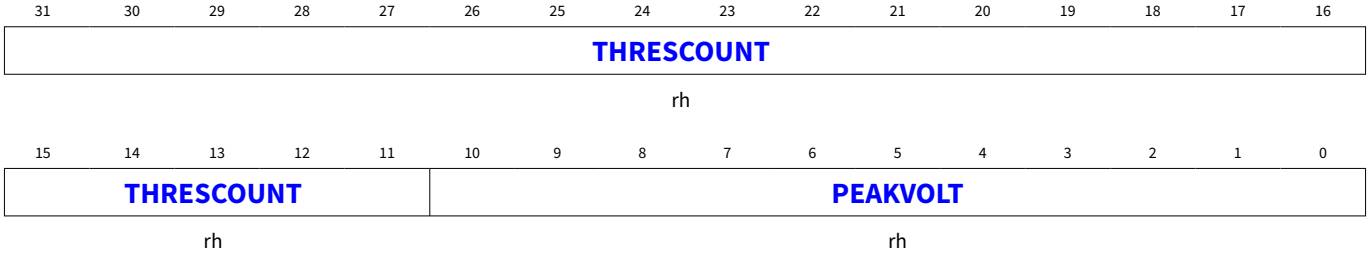
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
THRESCOUNT					PEAKVOLT										
rh					rh										

Field	Bits	Type	Description
PEAKVOLT	10:0	rh	**Peak voltage detected value** This bit-field indicates the peak value detected for the respective supply voltage. Expressed as number of LSBs of the secondary monitor ADC for the respective supply voltage. Peak Voltage = [LSB * (VxxPVDVAL-1)] ; LSB = tbd mV
THRESCOUNT	31:11	rh	**Detected sample count above configured threshold value** This bit-field indicates the number of voltage samples which were detected, after filtering, above the configured threshold value (PVDVxxTHR).

36.4.28 VDDEXT peak voltage detection history status register

PEAK_VDDEXTHISTSTAT Offset address: 9220_H

VDDEXT peak voltage detection history status register LVDPMS1 Reset value: $0000\ 0000_H$

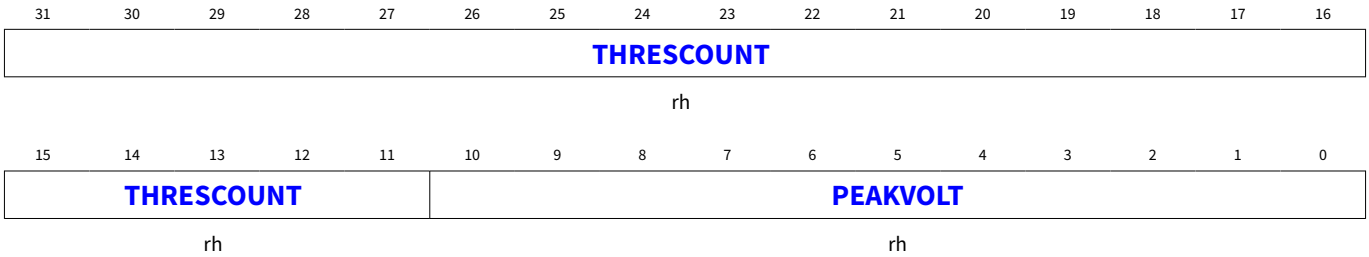


Field	Bits	Type	Description
PEAKVOLT	10:0	rh	**Peak voltage detected value** This bit-field indicates the peak value detected for the respective supply voltage. Expressed as number of LSBs of the secondary monitor ADC for the respective supply voltage. Peak Voltage = [LSB * (VxxPVDVAL-1)] ; LSB = tbd mV
THRESCOUNT	31:11	rh	**Detected sample count above configured threshold value** This bit-field indicates the number of voltage samples which were detected, after filtering, above the configured threshold value (PVDVxxTHR).

36.4.29 VDDEXTDC peak voltage detection history status register

PEAK_VDDEXTDCHISTSTAT Offset address: 9224_H

VDDEXTDC peak voltage detection history status register LVDPMS1 Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
PEAKVOLT	10:0	rh	**Peak voltage detected value** This bit-field indicates the peak value detected for the respective supply voltage. Expressed as number of LSBs of the secondary monitor ADC for the respective supply voltage. Peak Voltage = [LSB * (VxxPVDVAL-1)] ; LSB = tbd mV
THRESCOUNT	31:11	rh	**Detected sample count above configured threshold value** This bit-field indicates the number of voltage samples which were detected, after filtering, above the configured threshold value (PVDVxxTHR).

36.4.30 VDD drift monitoring configuration register

DRMON_VDD Offset address: 9228_H

VDD drift monitoring configuration register LVDPMS2 Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								FACTOR				0			ENABLE
r								rwh				r			rw

Field	Bits	Type	Description
ENABLE	0	rw	**Enable/disable drift monitoring**
FACTOR	7:4	rwh	**Exponent of the downsampling factor R** Downsampling factor R = 2^FACTOR.
0	3:1, 31:8	r	**Reserved** Read as 0; should be written with 0.

36.4.31 VDD drift monitor status register

DRMON_VDDSTAT Offset address: $922C_H$

VDD drift monitor status register LVDPMS2 Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0					VALUE										
r					rh										

Field	Bits	Type	Description
VALUE	10:0	rh	**Accumulated filtered value** This bit-field indicates the supply voltage filtered value accumulated for drift monitoring. Expressed as number of LSBs of the voltage monitoring ADC for the respective rail.
0	31:11	r	**Reserved** Read as 0.

36.4.32 VDDEXT drift monitoring configuration register

DRMON_VDDEXT Offset address: 9230_H

VDDEXT drift monitoring configuration register LVDPMS2 Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								FACTOR				0			ENABLE
r								rwh				r			rw

Field	Bits	Type	Description
ENABLE	0	rw	**Enable/disable drift monitoring**
FACTOR	7:4	rwh	**Exponent of the downsampling factor R** Downsampling factor R = 2^FACTOR.
0	3:1, 31:8	r	**Reserved** Read as 0; should be written with 0.

36.4.33 VDDEXT drift monitor status register

DRMON_VDDEXTSTAT Offset address: 9234_H

VDDEXT drift monitor status register LVDPMS2 Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0					VALUE										
r					rh										

Field	Bits	Type	Description
VALUE	10:0	rh	**Accumulated filtered value** This bit-field indicates the supply voltage filtered value accumulated for drift monitoring. Expressed as number of LSBs of the voltage monitoring ADC for the respective rail.
0	31:11	r	**Reserved** Read as 0.

36.4.34 VDDEXTDC drift monitoring configuration register

DRMON_VDDEXTDC Offset address: 9238_H

VDDEXTDC drift monitoring configuration register LVDPMS2 Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								FACTOR				0			ENABLE
r								rwh				r			rw

Field	Bits	Type	Description
ENABLE	0	rw	**Enable/disable drift monitoring**
FACTOR	7:4	rwh	**Exponent of the downsampling factor R** Downsampling factor R = 2^FACTOR.
0	3:1, 31:8	r	**Reserved** Read as 0; should be written with 0.

36.4.35 VDDEXTDC drift monitor status register

DRMON_VDDEXTDCSTAT Offset address: $923C_H$

VDDEXTDC drift monitor status register LVDPMS2 Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0					VALUE										
r					rh										

Field	Bits	Type	Description
VALUE	10:0	rh	**Accumulated filtered value** This bit-field indicates the supply voltage filtered value accumulated for drift monitoring. Expressed as number of LSBs of the voltage monitoring ADC for the respective rail.
0	31:11	r	**Reserved** Read as 0.

36.4.36 VDDM drift monitoring configuration register

DRMON_VDDM Offset address: 9240_H

VDDM drift monitoring configuration register LVDPMS2 Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								FACTOR				0			ENABLE
r								rwh				r			rw

Field	Bits	Type	Description
ENABLE	0	rw	**Enable/disable drift monitoring**
FACTOR	7:4	rwh	**Exponent of the downsampling factor R** Downsampling factor R = 2^FACTOR.
0	3:1, 31:8	r	**Reserved** Read as 0; should be written with 0.

36.4.37 VDDM drift monitor status register

DRMON_VDDMSTAT Offset address: 9244_H

VDDM drift monitor status register LVDPMS2 Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0					VALUE										
r					rh										

Field	Bits	Type	Description
VALUE	10:0	rh	**Accumulated filtered value** This bit-field indicates the supply voltage filtered value accumulated for drift monitoring. Expressed as number of LSBs of the voltage monitoring ADC for the respective rail.
0	31:11	r	**Reserved** Read as 0.

36.4.38 VDDPMS0 drift monitoring configuration register

DRMON_VDDPMS0 Offset address: 9248_H

VDDPMS0 drift monitoring configuration register LVDPMS2 Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								FACTOR				0			ENABLE
r								rwh				r			rw

Field	Bits	Type	Description
ENABLE	0	rw	**Enable/disable drift monitoring**
FACTOR	7:4	rwh	**Exponent of the downsampling factor R** Downsampling factor R = 2^FACTOR.
0	3:1, 31:8	r	**Reserved** Read as 0; should be written with 0.

36.4.39 VDDPMS0 drift monitor status register

DRMON_VDDPMS0STAT Offset address: $924C_H$

VDDPMS0 drift monitor status register LVDPMS2 Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

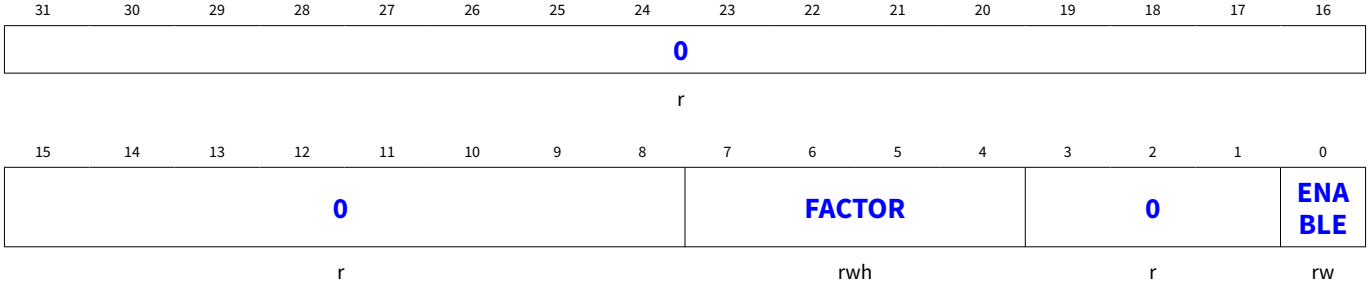
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0					VALUE										
r					rh										

Field	Bits	Type	Description
VALUE	10:0	rh	**Accumulated filtered value** This bit-field indicates the supply voltage filtered value accumulated for drift monitoring. Expressed as number of LSBs of the voltage monitoring ADC for the respective rail.
0	31:11	r	**Reserved** Read as 0.

36.4.40 VDDPMS2 drift monitoring configuration register

DRMON_VDDPMS2 Offset address: 9250_H

VDDPMS2 drift monitoring configuration register LVDPMS2 Reset value: $0000\ 0000_H$

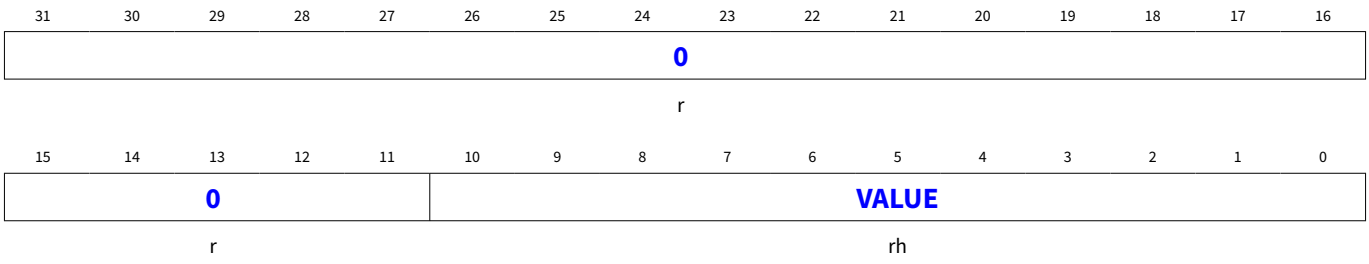


Field	Bits	Type	Description
ENABLE	0	rw	**Enable/disable drift monitoring**
FACTOR	7:4	rwh	**Exponent of the downsampling factor R** Downsampling factor R = 2^FACTOR.
0	3:1, 31:8	r	**Reserved** Read as 0; should be written with 0.

36.4.41 VDDPMS2 drift monitor status register

DRMON_VDDPMS2STAT Offset address: 9254_H

VDDPMS2 drift monitor status register LVDPMS2 Reset value: $0000\ 0000_H$

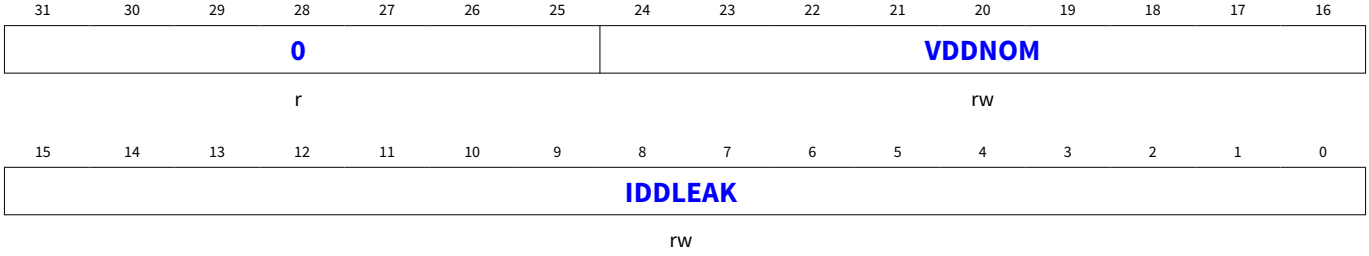


Field	Bits	Type	Description
VALUE	10:0	rh	**Accumulated filtered value** This bit-field indicates the supply voltage filtered value accumulated for drift monitoring. Expressed as number of LSBs of the voltage monitoring ADC for the respective rail.
0	31:11	r	**Reserved** Read as 0.

36.4.42 Voltage ID (VID) configuration register 0

VID_CON0 Offset address: 9258_H

Voltage ID [VID] configuration register 0 Reset values see: Table 1447



Field	Bits	Type	Description
IDDLEAK	15:0	rw	**Leakage current at 127°C** Total leakage current $I_{DD} + I_{DDPHYx}$ for this device, measured at T_j=127°C, at the lower nominal V_{DD} voltage level as documented in the data sheet operating conditions. LSB= 1mA
VDDNOM	24:16	rw	**Nominal VDD voltage level** V_{DD} nominal voltage level = VDDNOM * LSB LSB = $V_{TRKCORELSB}$ data sheet parameter Only two values are allowed, either the V_{DD} value for Slow and POR samples,or the V_{DD} value for fast samples, according to the typical value documented in the data sheet. For fast devices, the lower typical V_{DD} voltage level as documented in the data sheet operating conditions is used (0x159). For slow and typical devices, the higher typical V_{DD} voltage level as documented in the data sheet operating conditions is used (0x16A).
0	31:25	r	**Reserved** Read as 0; should be written with 0.

Table 1447 Reset values of VID_CON0

Reset type	Reset value	Note
Cold PowerOn Reset	$0000\ 0000_H$	
After Boot-FW Value	$0000\ 000–\ ----\ ----\ ----\ ----\ ----\ ----\ ----\ ----_B$	

36.4.43 Interrupt configuration register 2

INT_CON2 Offset address: 9274_H

Interrupt configuration register 2 Reset values see: Table 1448

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0										FIFOEMPTY	FIFOFULL	VDDEXTDCLVL	DCDCSYNCLCK	DCDCVOK	SCRPINRST
r										rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SCRWDT	SRECC	SCRRST	SCRINT1	SCRINT0	PINSEXTWKP	PINSSBWKP	PINCWKP	PINBWKP	PINAWKP	ESR2WKP	ESR1WKP	ESR0WKP	WUTWKP	RTCCMP1	RTCCMP0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RTCCMP0	0	rw	**RTC CMP0 interrupt enable** Interrupt triggered on an RTC CMP0 compare match event. 0_B Interrupt is disabled. 1_B Interrupt is enabled.
RTCCMP1	1	rw	**RTC CMP1 interrupt enable** Interrupt triggered on an RTC CMP1 compare match event. 0_B Interrupt is disabled. 1_B Interrupt is enabled.
WUTWKP	2	rw	**WUTWKP interrupt enable** Interrupt triggered on a WUTCNT underflow event. 0_B Interrupt is disabled. 1_B Interrupt is enabled.
ESR0WKP	3	rw	**ESR0WKP interrupt enable** Interrupt triggered on a ESR0WKP event. 0_B Interrupt is disabled. 1_B Interrupt is enabled.
ESR1WKP	4	rw	**ESR1WKP interrupt enable** Interrupt triggered on a ESR1WKP event. 0_B Interrupt is disabled. 1_B Interrupt is enabled.
ESR2WKP	5	rw	**ESR2WKP interrupt enable** Interrupt triggered on a ESR2WKP event. 0_B Interrupt is disabled. 1_B Interrupt is enabled.
PINAWKP	6	rw	**PINAWKP interrupt enable** Interrupt triggered on a PINAWKP event. 0_B Interrupt is disabled. 1_B Interrupt is enabled.

(table continues...)

(continued)

Field	Bits	Type	Description
PINBWKP	7	rw	**PINBWKP interrupt enable** Interrupt triggered on a PINBWKP event. 0_B Interrupt is disabled. 1_B Interrupt is enabled.
PINCWKP	8	rw	**PINCWKP interrupt enable** Interrupt triggered on a PINCWKP event. 0_B Interrupt is disabled. 1_B Interrupt is enabled.
PINSSBWKP	9	rw	**PINSSBWKP interrupt enable** Interrupt triggered on a PINSSBWKP event. 0_B Interrupt is disabled. 1_B Interrupt is enabled.
PINSEXTWKP	10	rw	**PINSEXTWKP interrupt enable** Interrupt triggered on a PINSEXTWKP event. 0_B Interrupt is disabled. 1_B Interrupt is enabled.
SCRINT0	11	rw	**SCRINT0 interrupt enable** Interrupt triggered on a SCRINT0 event triggered by SCR to PMS to decode information in SCR_STAT.SCRINT register. 0_B Interrupt is disabled. 1_B Interrupt is enabled.
SCRINT1	12	rw	**SCRINT1 interrupt enable** Interrupt triggered on a SCRINT1 event triggered by SCR to PMS to decode information in SCR_STAT.SCRINT register. 0_B Interrupt is disabled. 1_B Interrupt is enabled.
SCRRST	13	rw	**SCRRST interrupt enable** Interrupt triggered by SCR to PMS on an internal SCR software reset. 0_B Interrupt is disabled. 1_B Interrupt is enabled.
SCRECC	14	rw	**SCRECC interrupt enable** Interrupt triggered by SCR to PMS on an internal RAM double bit ECC error. 0_B Interrupt is disabled. 1_B Interrupt is enabled.
SCRWDT	15	rw	**SCRWDT interrupt enable** Interrupt triggered by SCR to PMS on an internal SCR watchdog timeout error. 0_B Interrupt is disabled. 1_B Interrupt is enabled.

(table continues...)

(continued)

Field	Bits	Type	Description
SCRPINRST	16	rw	**SCRPINRST interrupt enable** Interrupt triggered by SCR to PMS on an internal SCR pin reset. 0_B Interrupt is disabled. 1_B Interrupt is enabled.
DCDCVOK	17	rw	**SDVOK interrupt enable** Interrupt triggered on EVRC_STAT0.SDVOK rising edge event. 0_B Interrupt is disabled. 1_B Interrupt is enabled.
DCDCSYNCLCK	18	rw	**EVRC SYNCLCK interrupt enable** Interrupt triggered on a state change of EVRC_STAT0.SYNCLCK bitfield. 0_B Interrupt is disabled. 1_B Interrupt is enabled.
VDDEXTDCLVL	19	rw	**VDDEXTDCLVL interrupt enable** Interrupt triggered on a state change of EVRC_STAT0.VDDEXTDCLVL bitfield. 0_B Interrupt is disabled. 1_B Interrupt is enabled.
FIFOFULL	20	rw	**FIFOFULL interrupt enable** Interrupt triggered when REGACCSTAT.FIFOFULL is set. 0_B Interrupt is disabled. 1_B Interrupt is enabled.
FIFOEMPTY	21	rw	**FIFOEMPTY interrupt enable** Interrupt triggered when REGACCSTAT.FIFOEMPTY is set. 0_B Interrupt is disabled. 1_B Interrupt is enabled.
0	31:22	r	**Reserved** Read as 0; should be written with 0.

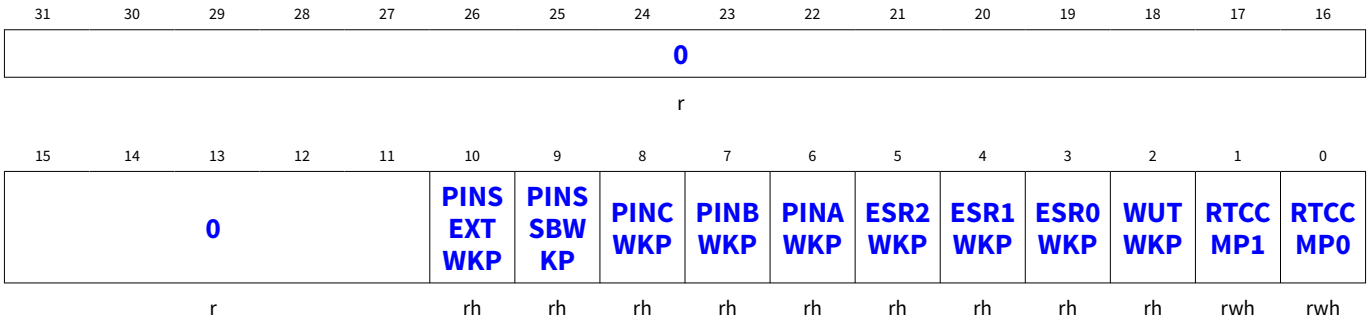
Table 1448 Reset values of INT_CON2

Reset type	Reset value	Note
LVDPMS2 Reset	0000 0000_H	
Cold PORST	0000 0000_H	

36.4.44 Interrupt status register

INT_STAT Offset address: 9278_H

Interrupt status register Reset values see: Table 1449



Field	Bits	Type	Description
RTCCMP0	0	rwh	**RTC CMP0 interrupt flag** In case interrupt was triggered by RTC event during RUN mode, this flag is set. The bit shall be cleared explicitly via INT_CLR.RTCCMP0INTCLR bit after interrupt is serviced. Application SW can also set this bit by writing 1 to it. 0_B No interrupt event detected from RTC. 1_B An interrupt event from RTC detected.
RTCCMP1	1	rwh	**RTC CMP1 interrupt flag** In case interrupt was triggered by RTC event during RUN mode, this flag is set. The bit shall be cleared explicitly via INT_CLR.RTCCMP1INTCLR bit after interrupt is serviced. Application SW can also set this bit by writing 1 to it. 0_B No interrupt event detected from RTC. 1_B An interrupt event from RTC detected.
WUTWKP	2	rh	**WUTWKP interrupt flag** In case interrupt was triggered by Wake-Up Timer (WUT) event during RUN mode, this flag is set. The bit shall be cleared explicitly via INT_CLR.WUTINTCLR bit after interrupt is serviced. 0_B No interrupt event detected from WUT. 1_B An interrupt event from WUT detected.
ESR0WKP	3	rh	**ESR0WKP interrupt flag** In case interrupt was triggered by ESR0 pin event during RUN mode, this flag is set. The bit shall be cleared explicitly via INT_CLR.ESR0INTCLR bit after interrupt is serviced. 0_B No interrupt event detected on ESR0 input. 1_B An interrupt event as defined by WAKEUP_CON1. ESR0EDCON detected on ESR0 input.
ESR1WKP	4	rh	**ESR1WKP interrupt flag** In case interrupt was triggered by ESR1 pin event during RUN mode, this flag is set. The bit shall be cleared explicitly via INT_CLR.ESR1INTCLR bit after interrupt is serviced. 0_B No interrupt event detected on ESR1 input. 1_B An interrupt event as defined by WAKEUP_CON1. ESR1EDCON detected on ESR1 input.

(table continues...)

(continued)

Field	Bits	Type	Description
ESR2WKP	5	rh	**ESR2WKP interrupt flag** In case interrupt was triggered by ESR2 pin event during RUN mode, this flag is set. The bit shall be cleared explicitly via INT_CLR.ESR2INTCLR bit after interrupt is serviced. 0_B No interrupt event detected on ESR2 input. 1_B An interrupt event as defined by WAKEUP_CON0.ESR2EDCON detected on ESR2 input.
PINAWKP	6	rh	**PINAWKP interrupt flag** In case interrupt was triggered by PINA pin event during RUN mode, this flag is set. The bit shall be cleared explicitly via INT_CLR.PINAINTCLR bit after interrupt is serviced. 0_B No interrupt event detected on Pin A input. 1_B An interrupt event as defined by WAKEUP_CON1. PINAEDCON detected on Pin A input.
PINBWKP	7	rh	**PINBWKP interrupt flag** In case interrupt was triggered by PINB pin event during RUN mode, this flag is set. The bit shall be cleared explicitly via INT_CLR.PINBINTCLR bit after interrupt is serviced. 0_B No interrupt event detected on the Pin B input. 1_B An interrupt event as defined by WAKEUP_CON0. PINBEDCON detected on Pin B input.
PINCWKP	8	rh	**PINCWKP interrupt flag** In case interrupt was triggered by PINC pin event during RUN mode, this flag is set. The bit shall be cleared explicitly via INT_CLR.PINCINTCLR bit after interrupt is serviced. 0_B No interrupt event detected on the Pin C input. 1_B An interrupt event as defined by WAKEUP_CON0. PINCEDCON detected on Pin C input.
PINSSBWKP	9	rh	**PINSSBWKP interrupt flag** 0_B No interrupt event detected on the $V_{DDEVRSB}$ pins input. 1_B An interrupt event as detected on $V_{DDEVRSB}$ pins input.
PINSEXTWKP	10	rh	**PINSEXTWKP interrupt flag** 0_B No interrupt event detected on the V_{DDEXT} pins input. 1_B An interrupt event as detected on V_{DDEXT} pins input.
0	31:11	r	**Reserved** Read as 0.

Table 1449 Reset values of INT_STAT

Reset type	Reset value	Note
LVDPMS2 Reset	$0000\ 0000_H$	
Cold PORST	$0000\ 0000_H$	

36.4.45 Interrupt status clear register

INT_CLR Offset address: $927C_H$

Interrupt status clear register Reset values see: Table 1450

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0					PINSEXTINTCLR	PINSSBINTCLR	PINCINTCLR	PINBINTCLR	PINAINTCLR	ESR2INTCLR	ESR1INTCLR	ESR0INTCLR	WUTINTCLR	RTCCMP1INTCLR	RTCCMP0INTCLR
r					w	w	w	w	w	w	w	w	w	w	w

Field	Bits	Type	Description
RTCCMP0INTCLR	0	w	**RTC CMP0 interrupt indication flag clear** 0_B No action 1_B Clear INT_STAT.RTCCMP0 bit.
RTCCMP1INTCLR	1	w	**RTC CMP1 interrupt indication flag clear** 0_B No action 1_B Clear INT_STAT.RTCCMP1 bit.
WUTINTCLR	2	w	**WUT interrupt indication flag clear** 0_B No action 1_B Clear INT_STAT.WUTWKP bit.
ESR0INTCLR	3	w	**ESR0 interrupt indication flag clear** 0_B No action 1_B Clear INT_STAT.ESR0WKP bit.
ESR1INTCLR	4	w	**ESR1 interrupt indication flag clear** 0_B No action 1_B Clear INT_STAT.ESR1WKP bit.
ESR2INTCLR	5	w	**ESR2 interrupt indication flag clear** 0_B No action 1_B Clear INT_STAT.ESR2WKP bit.
PINAINTCLR	6	w	**PINA interrupt indication flag clear** 0_B No action 1_B Clear INT_STAT.PINAWKP bit.
PINBINTCLR	7	w	**PINB interrupt indication flag clear** 0_B No action 1_B Clear INT_STAT.PINBWKP bit.
PINCINTCLR	8	w	**PINC interrupt indication flag clear** 0_B No action 1_B Clear INT_STAT.PINCWKP bit.

(table continues...)

(continued)

Field	Bits	Type	Description
PINSSBINTCLR	9	w	**VDDEVRSB pins interrupt indication flag clear** 0_B No action 1_B Clear INT_STAT.PINSSBWKP bit.
PINSEXTINTCLR	10	w	**VDDEXT pins interrupt indication flag clear** 0_B No action 1_B Clear INT_STAT.PINSEXTWKP bit.
0	31:11	r	**Reserved** Read as 0; should be written with 0.

Table 1450 Reset values of INT_CLR

Reset type	Reset value	Note
LVDPMS2 Reset	0000 0000$_H$	
Cold PORST	0000 0000$_H$	

36.4.46 Pad configuration register

Before accessing, check the REGACCSTAT.FIFOEMPTY and REGACCSTAT.FIFOFULL status as described in the user manual (LVD reset section).

PAD_CON Offset address: 9280$_H$

Pad configuration register LVDANA Reset value: 0000 0040$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								PORSTDFEN		ESR2WKPRUNPD	ESR0PD	SBTRIST_P	SBTRIST	ALLTRIST_P	ALLTRIST
r								rw		rw	rw	w	rwh	w	rwh

Field	Bits	Type	Description
ALLTRIST	0	rwh	**Tristate enable for all pads** This bit decides whether pads behave as inputs with weak pull-up or tristate on reset assertion/de-assertion or Standby- Wake-up transition. After supply ramp-up or LVD reset, ALLTRIST = ! HWCFG[6]. 0_B No request to switch the input pad state of all the pads to tristate from pull-up (default reset state) 1_B Pad domain in tristate.

(table continues...)

(continued)

Field	Bits	Type	Description
ALLTRIST_P	1	w	**Bit protection for ALLTRIST** Writing 1 into this bit enables that ALLTRIST bit can be changed in the same write operation. 0_B TRISTREQ keeps the previous state and cannot be changed. 1_B TRISTREQ bit can be changed with a write operation.
SBTRIST	2	rwh	**Tristate enable for standby pads** This bit decides whether Standby pads behave as inputs with weak pull-up or tristate on reset assertion/de-assertion or Standby- Wake-up transition. After supply ramp-up or LVD reset, SBTRIST = ! HWCFG[6]. 0_B No request to switch the input pad state of all the pads to tristate from pull-up (default reset state) 1_B Pad domain in tristate.
SBTRIST_P	3	w	**Bit protection for SBTRIST** Writing 1 into this bit enables that SBTRIST bit can be changed in the same write operation. 0_B TRISTREQ keeps the previous state and cannot be changed. 1_B TRISTREQ bit can be changed with a write operation.
ESR0PD	4	rw	**ESR0 pull-down enable** This bit configures ESR0 pin behavior either as reset output (pull-down) or tristate/pull-up during Standby mode if VDDEXT is supplied. 0_B ESR0 configured as tristate or pull-up (depending on ALLTRIST setting) during Standby state 1_B ESR0 in pull-down state during Standby state.
ESR2WKPRUNPD	5	rw	**ESR2 is pulled down in case of STBY to RUN transition** 0_B No pull low. 1_B Pulled low during STBY to RUN transition. The deassertion of active pull down is delayed by maximum of 1 clock cycle of 70KHz clock once the STBY to RUN transition is completed

(table continues...)

(continued)

Field	Bits	Type	Description
PORSTDFEN	7:6	rw	**PORST digital filter enable** This bit field enables additional PORST digital filter ($t_{PORSTDF}$ parameter) to provide enhanced immunity against spurious spikes. 00_B PORST recognition delay = Analog PORST pad filter delay (see t_{SF2} data sheet parameter). 01_B PORST recognition delay = Analog PORST pad filter delay (see t_{SF2} data sheet parameter) + digital filter delay. If the system is in RUN or STANDBY1 mode at 100MHz, then the digital filter delay is ~800ns (20 clock cycles of 25MHz). Otherwise (system runs at 70kHz), the digital filter delay is ~28.6us (2 clock cycles of 70kHz). Exception: When PORST is used for waking up from standby, the digital filter delay is always ~28.6us (2 clock cycles of 70kHz). 10_B PORST recognition delay = Analog PORST pad filter delay (see t_{SF2} data sheet parameter) + digital filter delay. If the system is in RUN or STANDBY1 mode at 100MHz, then the digital filter delay is ~8.1us (203 clock cycles of 25MHz). Otherwise (system runs at 70kHz), the digital filter delay is ~28.6us (2 clock cycles of 70kHz). Exception: When PORST is used for waking up from standby, the digital filter delay is always ~28.6us (2 clock cycles of 70kHz). 11_B Reserved
0	31:8	r	**Reserved** Read as 0; should be written with 0.

Table 1451 Access mode restrictions of PAD_CON sorted by descending priority

Mode name	Access mode		Description
write 1 to .**ALLTRIST_P**	rwh	ALLTRIST	
write 1 to .**SBTRIST_P**	rwh	SBTRIST	
(default)	rh	ALLTRIST, SBTRIST	

36.4.47 ESR2 pad configuration register

PAD_ESR2CON Offset address: 9288_H

ESR2 pad configuration register Reset values see: Table 1452

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CLKDIV				RSTECNT											
rw				rw											

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0														ESR2ROAPP	ESR2ROEN
r														rw	rw

Field	Bits	Type	Description
ESR2ROEN	0	rw	**ESR reset output elongation in case of warm PORST enabled** ESR Reset output elongation in case of warm PORST enabled.
ESR2ROAPP	1	rw	**ESR reset output elongation in case of application reset enabled** ESR Reset output elongation in case of application reset enabled.
RSTECNT	27:16	rw	**Reload value for reset and ESR2 output elongation counter** On reset release, ESR2 output pin is elongated for RSTECNT+1 clock cycles of RSTECNT base clock. 000_H elongation by 1 clock cycle of RSTECNT base clock FFE_H elongation by 4095 clock cycles of RSTECNT base clock. Maximum allowed RSTECNT value is FFE_H
CLKDIV	31:28	rw	**Clock divider for 100MHz base clock** RSTECNT base clock = 100MHz / 2^CLKDIV
0	15:2	r	**Reserved** Read as 0; should be written with 0.

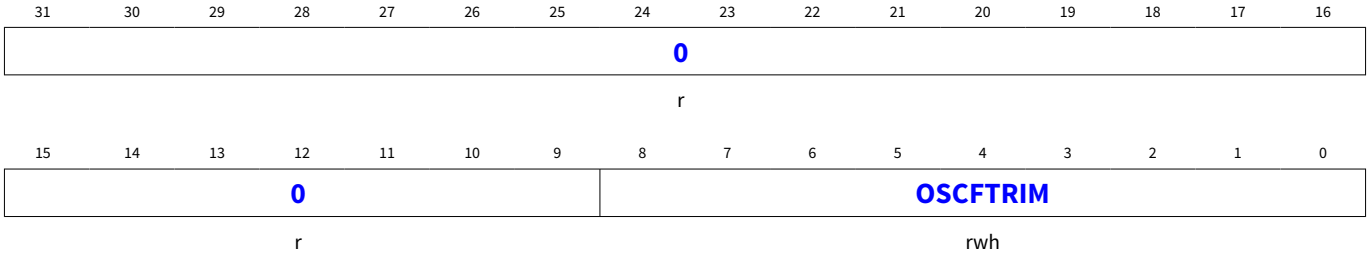
Table 1452 Reset values of PAD_ESR2CON

Reset type	Reset value	Note
LVDPMS2 Reset	$0000\ 0000_H$	
After Boot-FW Value	$0000\ 0000_H$	

36.4.48 HPOSC configuration register 1

HPOSCCON1 Offset address: 9294_H

HPOSC configuration register 1 Reset values see: Table 1453



Field	Bits	Type	Description
OSCFTRIM	8:0	rwh	**Back-up clock fine trim value** Fine trim value in two-complement representation for individual device trimming during operation. $f_{HPOSCLSBFT}$ specified as parameter. 000_H 0 LSB trimming 001_H +1 LSB trimming 002_H +2 LSB trimming 003_H +3 LSB trimming $0FF_H$ +255 LSB trimming 100_H -256 LSB trimming $1FD_H$ -3 LSB trimming $1FE_H$ -2 LSB trimming $1FF_H$ -1 LSB trimming
0	31:9	r	**Reserved** Read as 0; should be written with 0.

Table 1453 Reset values of HPOSCCON1

Reset type	Reset value	Note
LVDPMS1 Reset	$0000\ 0000_H$	
Cold PORST	$0000\ 0000_H$	

36.4.49 HPOSC trim register for dynamic trimming over temperature

HPOSCTRIM

HPOSC trim register for dynamic trimming over temperature

Offset address: $929C_H$

Reset values see: Table 1454

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0							TRIM7			TRIM6			TRIM5		
r							rw			rw			rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TRIM4			TRIM3			TRIM2			TRIM1			TRIM0			TRIM EN
rw			rw			rw			rw			rw			rw

Field	Bits	Type	Description
TRIMEN	0	rw	**Enable dynamic trimming over temperature** 0_B Disable dynamic trimming 1_B Enable dynamic trimming
TRIM0	3:1	rw	**Dynamic trim value for temperature range 0** Dynamic trim value for the temperature range 0 ≤ DTS_STAT.RESULT < 1280

(table continues...)

(continued)

Field	Bits	Type	Description
TRIM1	6:4	rw	**Dynamic trim value for temperature range 1** Dynamic trim value for the temperature range 1280 ≤ DTS_STAT.RESULT < 1408
TRIM2	9:7	rw	**Dynamic trim value for temperature range 2** Dynamic trim value for the temperature range 1408 ≤ DTS_STAT.RESULT < 1536
TRIM3	12:10	rw	**Dynamic trim value for temperature range 3** Dynamic trim value for the temperature range 1536 ≤ DTS_STAT.RESULT < 1664
TRIM4	15:13	rw	**Dynamic trim value for temperature range 4** Dynamic trim value for the temperature range 1664 ≤ DTS_STAT.RESULT < 1792
TRIM5	18:16	rw	**Dynamic trim value for temperature range 5** Dynamic trim value for the temperature range 1792 ≤ DTS_STAT.RESULT < 1920
TRIM6	21:19	rw	**Dynamic trim value for temperature range 6** Dynamic trim value for the temperature range 1920 ≤ DTS_STAT.RESULT < 2048
TRIM7	24:22	rw	**Dynamic trim value for temperature range 7** Dynamic trim value for the temperature range 2048 ≤ DTS_STAT.RESULT < 4096
0	31:25	r	**Reserved** Read as 0; should be written with 0.

Table 1454 Reset values of HPOSCTRIM

Reset type	Reset value	Note
LVDPMS1 Reset	0000 0000_H	
After Boot-FW Value	0000 000– ---- ---- ---- ---- ---- ---1_B	

36.4.50 HPBG trim register for dynamic trimming over temperature

HPBGTRIM

HPBG trim register for dynamic trimming over temperature

Offset address: $92A0_H$

Reset values see: Table 1455

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0							TRIM7			TRIM6			TRIM5		
r							rw			rw			rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TRIM4			TRIM3			TRIM2			TRIM1			TRIM0			TRIM EN
rw			rw			rw			rw			rw			rw

Field	Bits	Type	Description
TRIMEN	0	rw	**Enable dynamic trimming over temperature** 0_B Disable dynamic trimming 1_B Enable dynamic trimming
TRIM0	3:1	rw	**Dynamic trim value for temperature range 0** Dynamic trim value for the temperature range 0 ≤ DTS_STAT.RESULT < 1280
TRIM1	6:4	rw	**Dynamic trim value for temperature range 1** Dynamic trim value for the temperature range 1280 ≤ DTS_STAT.RESULT < 1408
TRIM2	9:7	rw	**Dynamic trim value for temperature range 2** Dynamic trim value for the temperature range 1408 ≤ DTS_STAT.RESULT < 1536
TRIM3	12:10	rw	**Dynamic trim value for temperature range 3** Dynamic trim value for the temperature range 1536 ≤ DTS_STAT.RESULT < 1664
TRIM4	15:13	rw	**Dynamic trim value for temperature range 4** Dynamic trim value for the temperature range 1664 ≤ DTS_STAT.RESULT < 1792
TRIM5	18:16	rw	**Dynamic trim value for temperature range 5** Dynamic trim value for the temperature range 1792 ≤ DTS_STAT.RESULT < 1920
TRIM6	21:19	rw	**Dynamic trim value for temperature range 6** Dynamic trim value for the temperature range 1920 ≤ DTS_STAT.RESULT < 2048
TRIM7	24:22	rw	**Dynamic trim value for temperature range 7** Dynamic trim value for the temperature range 2048 ≤ DTS_STAT.RESULT < 4096
0	31:25	r	**Reserved** Read as 0; should be written with 0.

Table 1455 Reset values of HPBGTRIM

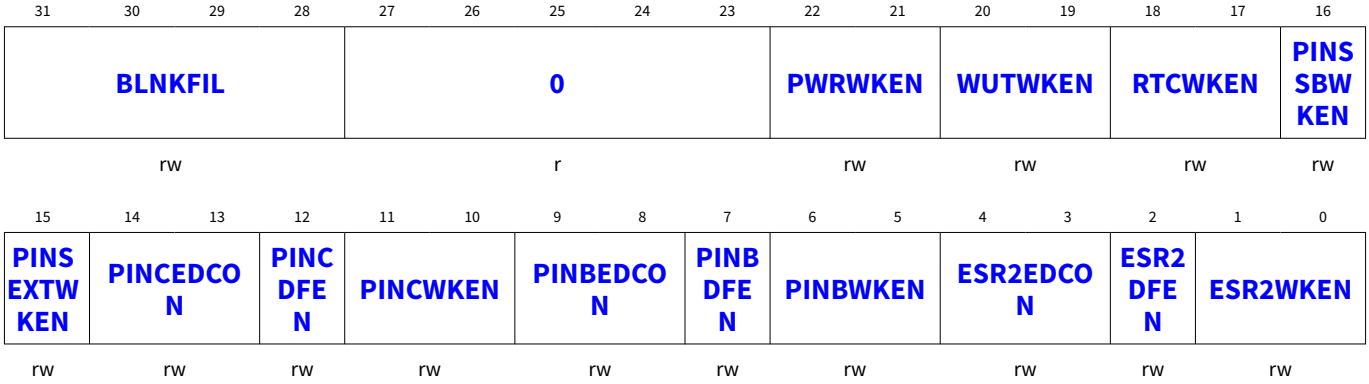
Reset type	Reset value	Note
LVDPMS1 Reset	0000 0000_H	
After Boot-FW Value	0000 000– ---- ---- ---- ---- ---- ---1_B	

36.4.51 Wake-up configuration register 0

Before accessing, check the REGACCSTAT.FIFOEMPTY and REGACCSTAT.FIFOFULL status as described in the user manual (LVD reset section).

WAKEUP_CON0 Offset address: $92AC_H$

Wake-up configuration register 0 Reset values see: Table 1456



Field	Bits	Type	Description
ESR2WKEN	1:0	rw	**ESR2 wake-up enable from standby** These bits configure wake-up via ESR2 pin from STANDBY mode. 00_B Wake-up via ESR2 pin is disabled. 01_B Wake-up via ESR2 pin from any standby mode directly to RUN mode is enabled. 10_B Wake-up via ESR2 pin from STANDBY0 to STANDBY1 mode is enabled. 11_B Sequential wake-up via ESR2 pin from STANDBY0: Wake-up from STANDBY0 to STANDBY1, setting the WAKEUP_STAT0.ESR2WKP flag. Subsequent wake-up from STANDBY1 to RUN mode, setting the WAKEUP_STAT0.ESR2OVRUN flag.
ESR2DFEN	2	rw	**ESR2 digital filter enable** This bit activates digital spike filter. If the digital filter (majority filter of 3 consecutive values, running on the 70 kHz standby clock) is enabled, then pulses less than 5 µs are suppressed and pulses longer than 50 µs will always result in a trigger. 0_B The filter is bypassed 1_B The filter is used
ESR2EDCON	4:3	rw	**ESR2 edge detection control** This bit field defines the edge of a ESR2 wake-up trigger 00_B No trigger is generated 01_B A trigger is generated upon a rising edge 10_B A trigger is generated upon a falling edge 11_B A trigger is generated upon a rising OR falling edge

(table continues...)

(continued)

Field	Bits	Type	Description
PINBWKEN	6:5	rw	**Pin B wake-up enable from standby** These bits configure wake-up via PINB pin from STANDBY mode. 00_B System wake-up via Pin B is disabled. 01_B Wake-up via Pin B from any standby mode directly to RUN mode is enabled. 10_B Wake-up via Pin B from STANDBY0 to STANDBY1 mode is enabled. 11_B Sequential wake-up via Pin B from STANDBY0: Wake-up from STANDBY0 to STANDBY1, setting the WAKEUP_STAT0.PINBWKP flag. Subsequent wake-up from STANDBY1 to RUN mode, setting the WAKEUP_STAT0.PINBOVRUN flag.
PINBDFEN	7	rw	**PINB digital filter enable** This bit activates digital spike filter. If the digital filter (majority filter of 3 consecutive values, running on the 70 kHz standby clock) is enabled, then pulses less than 5 µs are suppressed and pulses longer than 50 µs will always result in a trigger. 0_B The filter is bypassed 1_B The filter is used
PINBEDCON	9:8	rw	**PINB edge detection control** This bit field defines the edge of a Pin B wake-up trigger 00_B No trigger is generated 01_B A trigger is generated upon a rising edge 10_B A trigger is generated upon a falling edge 11_B A trigger is generated upon a rising OR falling edge
PINCWKEN	11:10	rw	**Pin C wake-up enable from standby** These bits configure wake-up via PINC pin from STANDBY mode. 00_B Wake-up via Pin C is disabled. 01_B Wake-up via Pin C from any standby mode directly to RUN mode is enabled. 10_B Wake-up via Pin C from STANDBY0 to STANDBY1 mode is enabled. 11_B Sequential wake-up via Pin C from STANDBY0: Wake-up from STANDBY0 to STANDBY1, setting the WAKEUP_STAT0.PINCWKP flag. Subsequent wake-up from STANDBY1 to RUN mode, setting the WAKEUP_STAT0.PINCOVRUN flag.
PINCDFEN	12	rw	**PINC digital filter enable** This bit activates digital spike filter. If the digital filter (majority filter of 3 consecutive values, running on the 70 kHz standby clock) is enabled, then pulses less than 5 µs are suppressed and pulses longer than 50 µs will always result in a trigger. 0_B The filter is bypassed 1_B The filter is used

(table continues...)

(continued)

Field	Bits	Type	Description
PINCEDCON	14:13	rw	**PINC edge detection control** This bit field defines the edge of a Pin C wake-up trigger 00_B No trigger is generated 01_B A trigger is generated upon a rising edge 10_B A trigger is generated upon a falling edge 11_B A trigger is generated upon a rising OR falling edge
PINSEXTWKEN	15	rw	**VDDEXT pins wake-up enable from standby** These bits configure wake-up via VDDEXT pins from STANDBY mode. 0_B Wake-up via VDDEXT pins is disabled. 1_B Wake-up via VDDEXT pins from any standby mode directly to RUN mode is enabled.
PINSSBWKEN	16	rw	**VDDEVRSB pins wake-up enable from standby** These bits configure wake-up via VDDEVRSB pins from STANDBY mode. 0_B Wake-up via VDDEVRSB pins is disabled. 1_B Wake-up via VDDEVRSB pins from any standby mode directly to RUN mode is enabled.
RTCWKEN	18:17	rw	**RTC wake-up enable from standby** These bits configure wake-up via RTC from STANDBY mode. 00_B Wake-up via RTC is disabled. 01_B Wake-up via RTC from any standby mode directly to RUN mode is enabled. 10_B Wake-up via RTC from STANDBY0 to STANDBY1 mode is enabled. 11_B Sequential wake-up via RTC from STANDBY0: Wake-up from STANDBY0 to STANDBY1, setting the WAKEUP_STAT0.RTCWKP flag. Subsequent wake-up from STANDBY1 to RUN mode, setting the WAKEUP_STAT0.RTCOVRUN flag.
WUTWKEN	20:19	rw	**WUT wake-up enable from standby** These bits configure wake-up via WUT from STANDBY mode. 00_B Wake-up via WUT is disabled. 01_B Wake-up via WUT from any standby mode directly to RUN mode is enabled. 10_B Wake-up via WUT from STANDBY0 to STANDBY1 mode is enabled. 11_B Sequential wake-up via WUT from STANDBY0: Wake-up from STANDBY0 to STANDBY1, setting the WAKEUP_STAT0.WUTWKP flag. Subsequent wake-up from STANDBY1 to RUN mode, setting the WAKEUP_STAT0.WUTOVRUN flag.

(table continues...)

(continued)

Field	Bits	Type	Description
PWRWKEN	22:21	rw	**Standby wake-up enable on VDDEXT supply ramp-up** This bit field enables wake-up on VDDEXT supply ramp-up after blanking filter time has expired. This is supported only in case Standby domain is supplied separately via VDDEVRSB supply pin and VDDEXT rail is switched off during Standby. 00_B Wake-up on VDDEXT supply ramp-up is disabled. Blanking filter configuration has no effect. 01_B Wake-up on VDDEXT supply ramp-up from any standby mode directly to RUN mode, after blanking filter time expiry, is enabled. 10_B Wake-up on VDDEXT supply ramp-up from STANDBY0 to STANDBY1 mode, after blanking time expiry, is enabled. 11_B Reserved
BLNKFIL	31:28	rw	**Blanking filter delay for wake-up** This bitfield enables a nominal blanking filter delay time immediately after Standby entry only after which a valid wake-up event is recognized and reacted upon. The actual delay may vary +- 30% to this nominal value. ***Note**: All other bit combinations are reserved. Incase WUT is used as a wake-up source, the blanking filter should be configured for a period greater than 0 µs.* 0_H 0 µs 1_H ~156 µs 2_H 1,25 ms 3_H 2,5 ms 4_H 5 ms 5_H 10 ms 6_H 20 ms 7_H 40 ms 8_H 80 ms 9_H 160 ms A_H 320 ms B_H 640 ms C_H 1280 ms D_H 2560 ms
0	27:23	r	**Reserved** Read as 0; should be written with 0.

Table 1456 Reset values of WAKEUP_CON0

Reset type	Reset value	Note
LVDANA Reset	$1000\ 0000_H$	
LVDPMS0 Reset	$1000\ 0000_H$	

36.4.52 Wake-up status register 0

Before accessing, check the REGACCSTAT.FIFOEMPTY and REGACCSTAT.FIFOFULL status as described in the user manual (LVD reset section).

WAKEUP_STAT0 Offset address: $92B0_H$

Wake-up status register 0 Reset values see: Table 1457

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0											WUTOVRUN	RTCOVRUN	PINCOVRUN	PINBOVRUN	ESR2OVRUN
r											rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								PWRWKP	WUTWKP	RTCWKP	PINSSBWKP	PINSEXTWKP	PINCWKP	PINBWKP	ESR2WKP
r								rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
ESR2WKP	0	rh	**ESR2 wake-up flag** In case wake-up was triggered by ESR2 pin during STANDBY, this flag is set. The bit shall be cleared explicitly after wakeup via WAKEUP_CLR0.ESR2WKPCLR bit before next STANDBY entry. 0_B No wake-up event detected on ESR2 input during STANDBY. 1_B An event as defined by WAKEUP_CON0.ESR2EDCON detected on ESR2 input.
PINBWKP	1	rh	**Pin B wake-up flag** In case wake-up was triggered by PINB pin during STANDBY, this flag is set. The bit shall be cleared explicitly after wakeup via WAKEUP_CLR0.PINBWKPCLR bit before next STANDBY entry. 0_B No wake-up event occurred on the Pin B input during STANDBY. 1_B An event as defined by WAKEUP_CON0. PINBEDCON detected on Pin B input.
PINCWKP	2	rh	**Pin C wake-up flag** In case wake-up was triggered by PINC pin during STANDBY, this flag is set. The bit shall be cleared explicitly after wakeup via WAKEUP_CLR0.PINCWKPCLR bit before next STANDBY entry. 0_B No wake-up event occurred on the Pin C input during STANDBY. 1_B An event as defined by WAKEUP_CON0. PINCEDCON detected on Pin C input.
PINSEXTWKP	3	rh	**VDDEXT pins wake-up flag** In case wake-up was triggered by VDDEXT pins during STANDBY, this flag is set. The bit shall be cleared explicitly after wakeup via WAKEUP_CLR0.PINSEXTWKPCLR bit before next STANDBY entry. 0_B No wake-up event occurred on the VDDEXT pins input during STANDBY. 1_B An event detected on VDDEXT pins input.

(table continues...)

(continued)

Field	Bits	Type	Description
PINSSBWKP	4	rh	**VDDEVRSB pins wake-up flag** In case wake-up was triggered by VDDEVRSB pins during STANDBY, this flag is set. The bit shall be cleared explicitly after wakeup via WAKEUP_CLR0.PINSSBWKPCLR bit before next STANDBY entry. 0_B No wake-up event occurred on the VDDEXT pins input during STANDBY. 1_B An event detected on VDDEVRSB pins input.
RTCWKP	5	rh	**RTC wake-up flag** In case wake-up was triggered by RTC during STANDBY, this flag is set. The bit shall be cleared explicitly after wakeup via WAKEUP_CLR0.RTCWKPCLR bit before next STANDBY entry. 0_B No wake-up event detected due to RTC underflow. 1_B A wake-up event from STANDBY was detected due to RTC underflow.
WUTWKP	6	rh	**WUT wake-up flag** In case wake-up was triggered by Wake-up timer during STANDBY, this flag is set. The bit shall be cleared explicitly after wakeup via WAKEUP_CLR0.WUTWKPCLR bit before next STANDBY entry. 0_B No wake-up event detected due to WUT underflow. 1_B A wake-up event from STANDBY was detected due to WUT underflow.
PWRWKP	7	rh	**Wake-up event on VDDEXT supply ramp-up** In case wake-up was triggered by VDDEXT ramp-up pin during STANDBY, this flag is set. The bit shall be cleared explicitly after wakeup via WAKEUP_CLR0.PWRWKPCLR bit before next STANDBY entry. 0_B No VDDEXT supply wake-up event detected. 1_B VDDEXT monitor threshold exceeded on VDDEXT supply ramp-up leading to system wake-up from STANDBY.
ESR2OVRUN	16	rh	**ESR2 overrun status flag** This flag indicates that a consecutive ESR2 wake-up event occurred while ESR2WKP flag was already set during STANDBY. The bit shall be cleared explicitly after wakeup via WAKEUP_CLR0.ESR2OVRUNCLR bit before next STANDBY entry. 0_B No overrun condition detected on ESR1 input. 1_B An overrun condition detected on ESR1 input.
PINBOVRUN	17	rh	**Pin B overrun status flag** This flag indicates that a consecutive PINB wake-up event occurred while PINBWKP flag was already set during STANDBY. The bit shall be cleared explicitly after wakeup via WAKEUP_CLR0.PINBOVRUNCLR bit before next STANDBY entry. 0_B No overrun condition detected on Pin B input. 1_B An overrun condition detected on Pin B input.

(table continues...)

(continued)

Field	Bits	Type	Description
PINCOVRUN	18	rh	**Pin C overrun status flag** This flag indicates that a consecutive PINC wake-up event occurred while PINCWKP flag was already set during STANDBY. The bit shall be cleared explicitly after wakeup via WAKEUP_CLR0.PINCOVRUNCLR bit before next STANDBY entry. 0_B No overrun condition detected on Pin C input. 1_B An overrun condition detected on Pin C input.
RTCOVRUN	19	rh	**RTC overrun status flag** This flag indicates that a consecutive RTC wake-up event occurred while RTCWKP flag was already set during STANDBY. The bit shall be cleared explicitly after wakeup via WAKEUP_CLR0.RTCOVRUNCLR bit before next STANDBY entry. 0_B No overrun condition detected of RTC events. 1_B An overrun condition detected of RTC events.
WUTOVRUN	20	rh	**WUT overrun status flag** This flag indicates that a consecutive WUT wake-up event occurred while WUTWKP flag was already set during STANDBY. The bit shall be cleared explicitly after wakeup via WAKEUP_CLR0.WUTOVRUNCLR bit before next STANDBY entry. WUT_CON.RELOAD need to be greater than 10 during Standby mode to be able to latch consecutive WUT underflow events and update the WUTOVRUN register bitfield. 0_B No overrun condition detected of WUT events. 1_B An overrun condition detected of WUT events.
0	15:8, 31:21	r	**Reserved** Read as 0.

Table 1457 Reset values of WAKEUP_STAT0

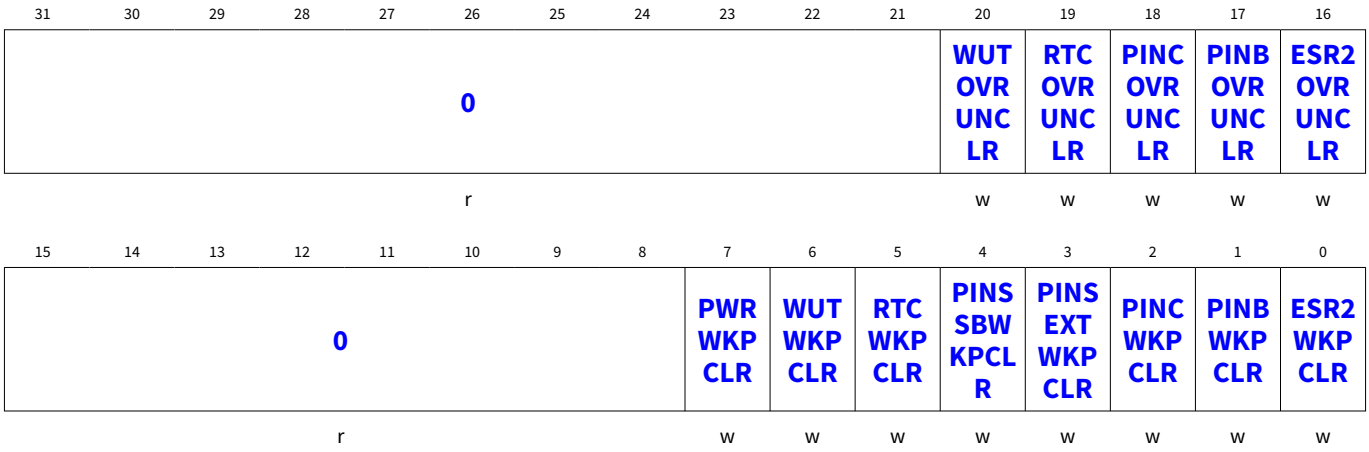
Reset type	Reset value	Note
LVDANA Reset	$0000\ 0000_H$	
LVDPMS0 Reset	$0000\ 0000_H$	

36.4.53 Standby and wake-up status clear register 0

Before accessing, check the REGACCSTAT.FIFOEMPTY and REGACCSTAT.FIFOFULL status as described in the user manual (LVD reset section).

WAKEUP_CLR0 Offset address: $92B4_H$

Standby and wake-up status clear register 0 Reset values see: Table 1458



Field	Bits	Type	Description
ESR2WKPCLR	0	w	**ESR2 wake-up indication flag clear** 0_B No action 1_B WAKEUP_STAT0.ESR2WKP bit cleared.
PINBWKPCLR	1	w	**PINB wake-up indication flag clear** 0_B No action 1_B WAKEUP_STAT0.PINBWKP bit cleared.
PINCWKPCLR	2	w	**PINC wake-up indication flag clear** 0_B No action 1_B WAKEUP_STAT0.PINCWKP bit cleared.
PINSEXTWKPCLR	3	w	**VDDEXT pins wake-up indication flag clear** 0_B No action 1_B WAKEUP_STAT0.PINSEXTWKP bit cleared.
PINSSBWKPCLR	4	w	**VDDEVRSB pins wake-up indication flag clear** 0_B No action 1_B WAKEUP_STAT0.PINSSBWKP bit cleared.
RTCWKPCLR	5	w	**RTC wake-up indication flag clear** 0_B No action 1_B WAKEUP_STAT0.RTCWKP bit cleared.
WUTWKPCLR	6	w	**WUT wake-up indication flag clear** 0_B No action 1_B WAKEUP_STAT0.WUTWKP bit cleared.
PWRWKPCLR	7	w	**PWRWKP wake-up indication flag clear** 0_B No action 1_B WAKEUP_STAT0.PWRWKP bit cleared.
ESR2OVRUNCLR	16	w	**ESR2 overrun status indication flag clear** 0_B No action 1_B WAKEUP_STAT0.ESR2OVRUN bit cleared.

(table continues...)

(continued)

Field	Bits	Type	Description
PINBOVRUNCLR	17	w	**PINB overrun status indication flag clear** 0_B No action 1_B WAKEUP_STAT0.PINBOVRUN bit cleared.
PINCOVRUNCLR	18	w	**PINC overrun status indication flag clear** 0_B No action 1_B WAKEUP_STAT0.PINCOVRUN bit cleared.
RTCOVRUNCLR	19	w	**RTC overrun status indication flag clear** 0_B No action 1_B WAKEUP_STAT0.RTCOVRUN bit cleared.
WUTOVRUNCLR	20	w	**WUT overrun status indication flag clear** 0_B No action 1_B WAKEUP_STAT0.WUTOVRUN bit cleared.
0	15:8, 31:21	r	**Reserved** Read as 0; should be written with 0.

Table 1458 Reset values of WAKEUP_CLR0

Reset type	Reset value	Note
LVDANA Reset	0000 0000$_H$	
LVDPMS0 Reset	0000 0000$_H$	

36.4.54 Wake-up configuration register 1

Before accessing, check the REGACCSTAT.FIFOEMPTY and REGACCSTAT.FIFOFULL status as described in the user manual (LVD reset section).

WAKEUP_CON1 Offset address: 92B8$_H$

Wake-up configuration register 1 Reset values see: Table 1459

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		PORSTWKEN	SCRWKEN	PINAEDCON		PINADFEN	PINAWKEN	ESR1EDCON		ESR1DFEN	ESR1WKEN	ESR0EDCON		ESR0DFEN	ESR0WKEN
r		rw	rw	rw		rw	rw	rw		rw	rw	rw		rw	rw

Field	Bits	Type	Description
ESR0WKEN	0	rw	**ESR0 wake-up enable from standby** These bits configure wake-up via ESR0 pin from STANDBY mode. 0_B System wake-up via ESR0 pin is disabled. 1_B System wake-up is enabled via ESR0 pin.

(table continues...)

(continued)

Field	Bits	Type	Description
ESR0DFEN	1	rw	**ESR0 digital filter enable** This bit activates digital spike filter. If the digital filter (majority filter of 3 consecutive values, running on the 70 kHz standby clock) is enabled, then pulses less than 5 µs are suppressed and pulses longer than 50 µs will always result in a trigger. 0_B The filter is bypassed 1_B The filter is used
ESR0EDCON	3:2	rw	**ESR0 edge detection control** This bit field defines the edge of a ESR0 wake-up trigger 00_B No trigger is generated 01_B A trigger is generated upon a rising edge 10_B A trigger is generated upon a falling edge 11_B A trigger is generated upon a rising OR falling edge
ESR1WKEN	4	rw	**ESR1 wake-up enable from standby** These bits configure wake-up via ESR1 pin from STANDBY mode. 0_B System wake-up via ESR1 pin is disabled. 1_B System wake-up is enabled via ESR1 pin.
ESR1DFEN	5	rw	**ESR1 digital filter enable** This bit activates digital spike filter. If the digital filter (majority filter of 3 consecutive values, running on the 70 kHz standby clock) is enabled, then pulses less than 5 µs are suppressed and pulses longer than 50 µs will always result in a trigger. 0_B The filter is bypassed 1_B The filter is used
ESR1EDCON	7:6	rw	**ESR1 edge detection control** This bit field defines the edge of a ESR1 wake-up trigger 00_B No trigger is generated 01_B A trigger is generated upon a rising edge 10_B A trigger is generated upon a falling edge 11_B A trigger is generated upon a rising OR falling edge
PINAWKEN	8	rw	**Pin A wake-up enable from standby** These bits configure wake-up via PINA pin from STANDBY mode. 0_B System wake-up via Pin A is disabled. 1_B System wake-up is enabled via Pin A.
PINADFEN	9	rw	**PINA digital filter enable** This bit activates digital spike filter. If the digital filter (majority filter of 3 consecutive values, running on the 70 kHz standby clock) is enabled, then pulses less than 5 µs are suppressed and pulses longer than 50 µs will always result in a trigger. 0_B The filter is bypassed 1_B The filter is used

(table continues...)

(continued)

Field	Bits	Type	Description
PINAEDCON	11:10	rw	**PINA edge detection control** This bit field defines the edge of a Pin A wake-up trigger 00_B No trigger is generated 01_B A trigger is generated upon a rising edge 10_B A trigger is generated upon a falling edge 11_B A trigger is generated upon a rising OR falling edge
SCRWKEN	12	rw	**Standby controller wake-up enable from standby** These bits configure wake-up via SCR from STANDBY mode. 0_B System wake-up via 8 bit Standby Controller is disabled. 1_B System wake-up is enabled via 8 bit Standby Controller.
PORSTWKEN	13	rw	**PORST pin wake-up enable from standby** These bits configure wake-up via PORST pin from STANDBY mode. 0_B System wake-up via PORST pin is disabled. 1_B System wake-up via PORST pin is enabled.
0	31:14	r	**Reserved** Read as 0; should be written with 0.

Table 1459 Reset values of WAKEUP_CON1

Reset type	Reset value	Note
LVDANA Reset	$0000\ 0000_H$	
LVDPMS0 Reset	$0000\ 0000_H$	

36.4.55 Wake-up status register 1

Before accessing, check the REGACCSTAT.FIFOEMPTY and REGACCSTAT.FIFOFULL status as described in the user manual (LVD reset section).

WAKEUP_STAT1 Offset address: $92BC_H$

Wake-up status register 1 Reset values see: Table 1460

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0											POR STOV RUN	SCR OVR UN	PINA OVR UN	ESR1 OVR UN	ESR0 OVR UN
r											rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0											POR STW KP	SCR WKP	PINA WKP	ESR1 WKP	ESR0 WKP
r											rh	rh	rh	rh	rh

Field	Bits	Type	Description
ESR0WKP	0	rh	**ESR0 wake-up flag** In case wake-up was triggered by ESR0 pin during STANDBY, this flag is set. The bit shall be cleared explicitly after wakeup via WAKEUP_CLR1.ESR0WKPCLR bit before next STANDBY entry. 0_B No wake-up event detected on ESR0 input during STANDBY. 1_B An event as defined by WAKEUP_CON1. ESR0EDCON detected on ESR0 input.
ESR1WKP	1	rh	**ESR1 wake-up flag** In case wake-up was triggered by ESR1 pin during STANDBY, this flag is set. The bit shall be cleared explicitly after wakeup via WAKEUP_CLR1.ESR1WKPCLR bit before next STANDBY entry. 0_B No wake-up event detected on ESR1 input during STANDBY. 1_B An event as defined by WAKEUP_CON1.ESR1EDCON detected on ESR1 input.
PINAWKP	2	rh	**Pin wake-up flag** In case wake-up was triggered by PINA pin during STANDBY, this flag is set. The bit shall be cleared explicitly after wakeup via WAKEUP_CLR1.PINAWKPCLR bit before next STANDBY entry. 0_B No wake-up event detected on Pin A input during STANDBY. 1_B An event as defined by WAKEUP_CON1. PINAEDCON detected on Pin A input.
SCRWKP	3	rh	**SCR wake-up flag** In case wake-up is triggered by SCR to the main controller during STANDBY, this flag is set. The bit shall be cleared explicitly after wakeup via WAKEUP_CLR1.SCRWKPCLR bit before next STANDBY entry. 0_B No SCR wake-up event detected. 1_B A SCR wake-up event occurred.
PORSTWKP	4	rh	**PORST wake-up flag** In case wake-up was triggered by PORST pin during STANDBY, this flag is set. The bit shall be cleared explicitly after wakeup via WAKEUP_CLR1.PORSTWKPCLR bit before next STANDBY entry. 0_B No wake-up event detected on PORST input during STANDBY if enabled via WAKEUP_CON1.PORSTWKEN bit. 1_B A wake-up event detected on PORST input if enabled via WAKEUP_CON1.PORSTWKEN bit.
ESR0OVRUN	16	rh	**ESR0 overrun status flag** This flag indicates that a consecutive ESR0 wake-up event occurred while ESR0WKP flag was already set during STANDBY. The bit shall be cleared explicitly after wakeup via WAKEUP_CLR1.ESR0OVRUNCLR bit before next STANDBY entry. 0_B No overrun condition detected on ESR0 input. 1_B An overrun condition detected on ESR0 input.

(table continues...)

(continued)

Field	Bits	Type	Description
ESR1OVRUN	17	rh	**ESR1 overrun status flag** This flag indicates that a consecutive ESR1 wake-up event occurred while ESR1WKP flag was already set during STANDBY. The bit shall be cleared explicitly after wakeup via WAKEUP_CLR1.ESR1OVRUNCLR bit before next STANDBY entry. 0_B No overrun condition detected on ESR1 input. 1_B An overrun condition detected on ESR1 input.
PINAOVRUN	18	rh	**Pin A overrun status flag** This flag indicates that a consecutive PINA wake-up event occurred while PINAWKP flag was already set during STANDBY. The bit shall be cleared explicitly after wakeup via WAKEUP_CLR1.PINAOVRUNCLR bit before next STANDBY entry. 0_B No overrun condition detected on Pin A input. 1_B An overrun condition detected on Pin A input.
SCROVRUN	19	rh	**SCR overrun status flag** This flag indicates that a consecutive SCR wake-up event occurred while SCRWKP flag was already set during STANDBY. The bit shall be cleared explicitly after wakeup via WAKEUP_CLR1.SCROVRUNCLR bit before next STANDBY entry. 0_B No overrun condition detected of SCR wake-up event. 1_B An overrun condition detected of SCR wake-up event.
PORSTOVRUN	20	rh	**PORST overrun status flag** This flag indicates that a consecutive PORST wake-up event occurred while PORSTWKP flag was already set during STANDBY. The bit shall be cleared explicitly after wakeup via WAKEUP_CLR1.PORSTOVRUNCLR bit before next STANDBY entry. 0_B No overrun condition detected on PORST input if enabled via WAKEUP_CON1.PORSTWKEN bit. 1_B An overrun condition detected on PORST input if enabled via WAKEUP_CON1.PORSTWKEN bit.
0	15:5, 31:21	r	**Reserved** Read as 0.

Table 1460 Reset values of WAKEUP_STAT1

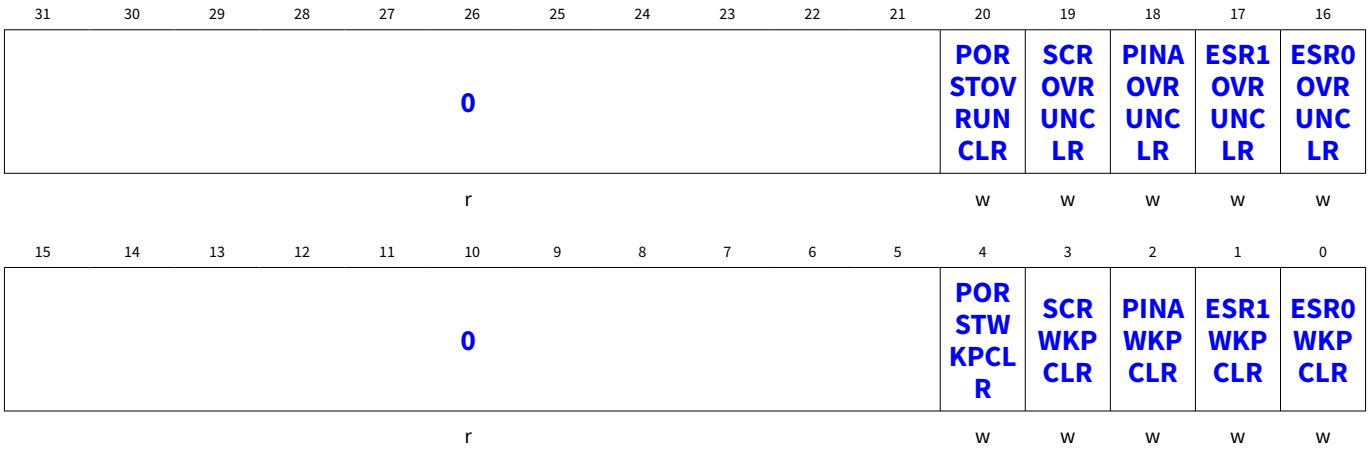
Reset type	Reset value	Note
LVDANA Reset	$0000\ 0000_H$	
LVDPMS0 Reset	$0000\ 0000_H$	

36.4.56 Standby and wake-up status clear register 1

Before accessing, check the REGACCSTAT.FIFOEMPTY and REGACCSTAT.FIFOFULL status as described in the user manual (LVD reset section).

WAKEUP_CLR1 Offset address: $92C0_H$

Standby and wake-up status clear register 1 Reset values see: Table 1461



Field	Bits	Type	Description
ESR0WKPCLR	0	w	**ESR0 wake-up indication flag clear** 0_B No action 1_B WAKEUP_STAT1.ESR0WKP bit cleared.
ESR1WKPCLR	1	w	**ESR1 wake-up indication flag clear** 0_B No action 1_B WAKEUP_STAT1.ESR1WKP bit cleared.
PINAWKPCLR	2	w	**PINA wake-up indication flag clear** 0_B No action 1_B WAKEUP_STAT1.PINAWKP bit cleared.
SCRWKPCLR	3	w	**SCR wake-up indication flag clear** 0_B No action 1_B WAKEUP_STAT1.SCRWKP bit cleared.
PORSTWKPCLR	4	w	**PORST wake-up indication flag clear** 0_B No action 1_B WAKEUP_STAT1.PORSTWKP bit cleared.
ESR0OVRUNCLR	16	w	**ESR0 overrun status indication flag clear** 0_B No action 1_B WAKEUP_STAT1.ESR0OVRUN bit cleared.
ESR1OVRUNCLR	17	w	**ESR1 overrun status indication flag clear** 0_B No action 1_B WAKEUP_STAT1.ESR1OVRUN bit cleared.
PINAOVRUNCLR	18	w	**PINA overrun status indication flag clear** 0_B No action 1_B WAKEUP_STAT1.PINAOVRUN bit cleared.
SCROVRUNCLR	19	w	**SCR overrun status indication flag clear** 0_B No action 1_B WAKEUP_STAT1.SCROVRUN bit cleared.

(table continues...)

(continued)

Field	Bits	Type	Description
PORSTOVRUNCLR	20	w	**PORST overrun status indication flag clear** 0_B No action 1_B WAKEUP_STAT1.PORSTOVRUN bit cleared.
0	15:5, 31:21	r	**Reserved** Read as 0; should be written with 0.

Table 1461 Reset values of WAKEUP_CLR1

Reset type	Reset value	Note
LVDANA Reset	0000 0000$_H$	
LVDPMS0 Reset	0000 0000$_H$	

36.4.57 Standby configuration register 1

STANDBY_CON1 Offset address: 92C8$_H$

Standby configuration register 1 LVDPMS1 Reset value: 0000 0000$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0													SWESRSBEN	VDDSBEN	SBMODSEL
r													rw	rw	rw

Field	Bits	Type	Description
SBMODSEL	0	rw	**Standby mode selection: STDBY0 or STBY1 state** Selection of either STANDBY0 or STANDBY1 state when entering Standby from RUN mode. 0_B STBY0 state is selected in standby mode. 1_B STBY1 state is selected in standby mode.
VDDSBEN	1	rw	**Standby entry on VDD supply ramp-down** This bit field enables Standby Entry on VDD supply ramp-down. This is supported only in case Standby domain is supplied separately via VDDEVRSB supply pin and VDD rail is switched off during Standby. The voltage threshold for entry is configured in the VMONS_VDDCON register. 0_B Standby Entry on VDD supply ramp-down is disabled. 1_B Standby Entry triggered on a VDD Supply undervoltage event (VDDUV). Blanking filter active on Standby mode entry.

(table continues...)

(continued)

Field	Bits	Type	Description
SWESRSBEN	2	rw	**Standby entry allowed by STBEV SW trigger on ESR events** 0_B Standby Entry on STBEV SW trigger on ESR events is disabled. 1_B Standby Entry triggered on STBEV SW trigger on ESR events. Blanking filter active on Standby mode entry.
0	31:3	r	**Reserved** Read as 0; should be written with 0.

36.4.58 Wake-up timer configuration register

Before accessing, check the REGACCSTAT.FIFOEMPTY and REGACCSTAT.FIFOFULL status as described in the user manual (LVD reset section).

WUT_CON Offset address: $92CC_H$

Wake-up timer configuration register Reset values see: Table 1462

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0				MODE		DIV	ENABLE	RELOAD							
r				rw		rw	rw	rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RELOAD															
rw															

Field	Bits	Type	Description
RELOAD	23:0	rw	**WUT reload value** The counter starts counting down from the RELOAD value. The current value of counter is indicated in WUT_STAT.COUNTER. On COUNTER underflow, a reload COUNTER = RELOAD takes place in auto reload mode.
ENABLE	24	rw	**WUT enable** This bit enables the Wake-up Timer. The status bit WUT_STAT.ENABLE is set once Wake-up Timer is enabled. 0_B Wake-up timer (WUT) disable request 1_B Wake-up timer (WUT) enable request.
DIV	25	rw	**WUT clock divider** A write to this register bitfield may trigger immediate update of the clock divider. 0_B Wake-up timer (WUT) clock = fSB = 70 KHz clock. 1_B Wake-up timer (WUT) clock = fSB (70 KHz) / 210.

(table continues...)

(continued)

Field	Bits	Type	Description
MODE	27:26	rw	**WUT mode selection** This bit configures the Wake-up Timer mode. The status bit WUT_STAT.MODE is respectively updated. A write to this register bitfield may trigger immediate update. 00_B Wake-up timer (WUT) auto reload mode selected. 01_B Wake-up timer (WUT) STANDBY0 auto stop mode selected. 10_B Wake-up timer (WUT) STANDBY1 auto stop mode selected. 11_B Reserved
0	31:28	r	**Reserved** Read as 0; should be written with 0.

Table 1462 Reset values of WUT_CON

Reset type	Reset value	Note
LVDANA Reset	$0000\ 0000_H$	
LVDPMS0 Reset	$0000\ 0000_H$	

36.4.59 Wake-up timer status register

Before accessing, check the REGACCSTAT.FIFOEMPTY and REGACCSTAT.FIFOFULL status as described in the user manual (LVD reset section).

WUT_STAT Offset address: $92D0_H$

Wake-up timer status register Reset values see: Table 1463

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0				MODE		RUN	ENABLE	COUNTER							
r				rh		rh	rh	rh							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
COUNTER															
rh															

Field	Bits	Type	Description
COUNTER	23:0	rh	**WUT counter value** The current WUT counter value is indicated in this register bitfield. The COUNTER value may have a deviation of 3 additional clock cycles to the expected counter value owing to synchronization overheads. The WUT clock is based on standby 70 kHz clock with ~ +- 30% variation. The counter depending on the mode can run through a RUN to STANDBY to RUN mode transition without interruption.

(table continues...)

(continued)

Field	Bits	Type	Description
ENABLE	24	rh	**WUT enable status** This bit indicates whether WUT is enabled. This bit is updated when WUT_CON.ENABLE bit is updated. 0_B Wake-up timer (WUT) is disabled. 1_B Wake-up timer (WUT) is enabled.
RUN	25	rh	**WUT run status** This bit indicates whether WUT is currently running. Due to synchronization to 70 KHz (fSB) WUT clock, setting of flag after enable may take up to 55 µs. 0_B Wake-up timer (WUT) is inactive. 1_B Wake-up timer (WUT) is active.
MODE	27:26	rh	**WUT mode status** This bit indicates the current WUT mode. This bit is updated when WUT_CON.MODE bit is updated. 00_B Wake-up timer (WUT) auto reload mode selected. 01_B Wake-up timer (WUT) STANDBY0 auto stop mode selected. 10_B Wake-up timer (WUT) STANDBY1 auto stop mode selected. 11_B Reserved
0	31:28	r	**Reserved** Read as 0.

Table 1463 Reset values of WUT_STAT

Reset type	Reset value	Note
LVDANA Reset	0000 0000$_H$	
LVDPMS0 Reset	0000 0000$_H$	

36.4.60 SCR configuration register 0

Before accessing, check the REGACCSTAT.FIFOEMPTY and REGACCSTAT.FIFOFULL status as described in the user manual (LVD reset section).

SCR_CON0 Offset address: 92D4$_H$

SCR configuration register 0 Reset values see: Table 1465

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0													SCRCFG		
r													rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0										PADLBIST	SCRCLKSEL	PORSTREQ_P	PORSTREQ	SCREN_P	SCREN
r										rw	rw	w	rw	w	rw

Field	Bits	Type	Description
SCREN	0	rw	**Standby controller enable request** SCR MBIST maybe activated independent of this bit. 0_B 8 bit Standby Controller is disabled. After setting SCREN=0, software must not access anymore SCR_CON1 or SCR_STAT registers. 1_B 8 bit Standby Controller is enabled
SCREN_P	1	w	**Bit protection for SCREN** 0_B Bit SCREN is not updated 1_B Bit SCREN can be updated
PORSTREQ	2	rw	**SCR reset behavior on warm PORST in normal RUN / SLEEP mode** 0_B 8 bit Standby Controller is not reset when warm PORST pin is asserted. 1_B 8 bit Standby Controller is reset when warm PORST pin is asserted. warm PORST usage in normal and standby mode.
PORSTREQ_P	3	w	**Bit protection for PORSTREQ** 0_B Bit PORSTREQ is not updated 1_B Bit PORSTREQ can be updated
SCRCLKSEL	4	rw	**Default SCR clock selection when SCR starts** 0_B SCR starts with 70 kHz oscillator clock. SCR can override and request 100 MHz clock activation through SCU_CLK.DIV_REQ settings. If SCU_CLK.DIV_REQ = 0xF, then SCR boots with 70kHz clock. 1_B SCR starts with 20 MHz clock. SCR can override the SCU_CLK.DIV_REQ setting. If SCU_CLK.DIV_REQ = 0xF, then based on the SCR request, the 100 MHz clock is switched off.
PADLBIST	5	rw	**Control of standby pads during LBIST** Configures if LBIST takes control of the standby PADs (or if instead the control during LBIST is maintained by the SCR). 0_B SCR maintains control of the standby pads during LBIST 1_B LBIST takes control of the standby pads

(table continues...)

(continued)

Field	Bits	Type	Description
SCRCFG	18:16	rw	**Hardware configuration of the 8 bit SCR controller.** ***Note**: Any change in SCRCFG is followed by a SCRSTREQ reset request of the 8 bit controller to start off in the chosen mode. All other bit combinations are reserved. Writing to SCR_CON0.SCRCFG with values != USERMODE will have an immediate effect on the enabling of debug pins.* 000_B User Mode is selected (Execution from 0000 001_B OCDS Mode with debug from SOC (DAP or SPD) interface selected 010_B OCDS Mode with SCR pins DAP0_0 and DAP1_0 selected 011_B OCDS Mode with SCR pins DAP0_1 and DAP1_1 selected 100_B OCDS Mode with SCR pin SPD_0 selected 101_B OCDS Mode with SCR pin SPD_1 selected 110_B Reserved 111_B Reserved
0	15:6, 31:19	r	**Reserved** Read as 0; should be written with 0.

Table 1464 Access mode restrictions of SCR_CON0 sorted by descending priority

Mode name	Access mode		Description
write 1 to .**PORSTREQ_P**	rw	PORSTREQ	
write 1 to .**SCREN_P**	rw	SCREN	
(default)	r	PORSTREQ, SCREN	

Table 1465 Reset values of SCR_CON0

Reset type	Reset value	Note
LVDANA Reset	$0000\ 0030_H$	
LVDPMS0 Reset	$0000\ 0030_H$	

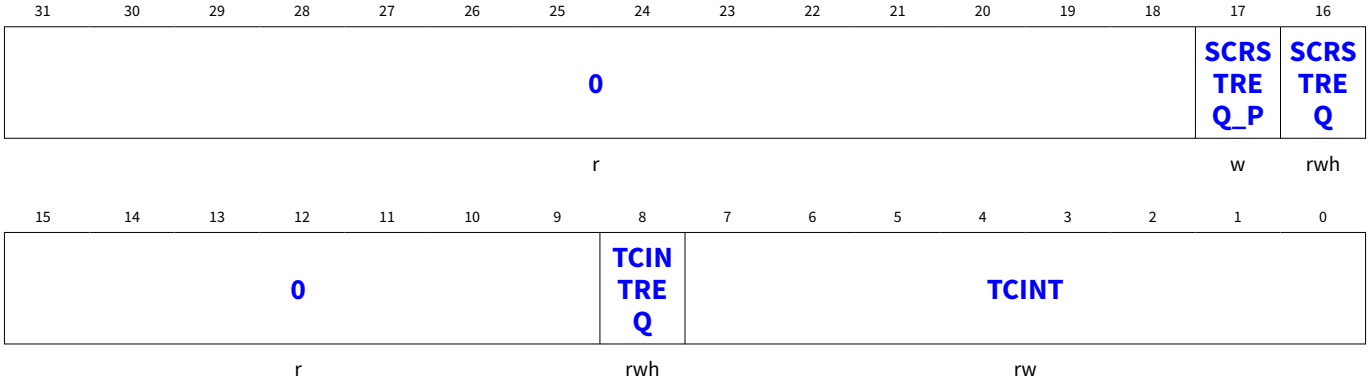
36.4.61 SCR configuration register 1

Write access protected by module-specific ("M") protection, as follows:

- When SCR_CON0.SCREN=0, write access to SCR_CON1 register is blocked.
- In order to write into the SCR_CON1 register, proceed as follows:
- If SCR_CON0.SCREN=0, then enable the SCR by writing SCR_CON0.SCREN=1
- Poll REGACCSTAT.FIFOEMPTY until FIFOEMPTY=1
- Then write into SCR_CON1 as needed.

SCR_CON1 Offset address: $92D8_H$

SCR configuration register 1 LVDPMS1 Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
TCINT	7:0	rw	**Data exchange from PMS main domain to standby controller** This bit field allows fast data exchange from PMS to SCR. The data may be read by SCR consequent to an interrupt request (TCINTREQ) from PMS/CPUx to SCR to decode the interrupt.
TCINTREQ	8	rwh	**SW interrupt request from PMS to standby controller** Setting this bit triggers an interrupt to the 8 bit Standby controller.
SCRSTREQ	16	rwh	**Standby controller reset request** 0_B No request for main reset of the 8 bit Standby Controller. 1_B 8 bit Standby Controller reset request.
SCRSTREQ_P	17	w	**Standby controller reset request enable** 0_B Bit SCRSTREQ is not updated 1_B Bit SCRSTREQ can be updated
0	15:9, 31:18	r	**Reserved** Read as 0; should be written with 0.

Table 1466 Access mode restrictions of SCR_CON1 sorted by descending priority

Mode name	Access mode		Description
write 1 to .**SCRSTREQ_P**	rwh	SCRSTREQ	
(default)	rh	SCRSTREQ	

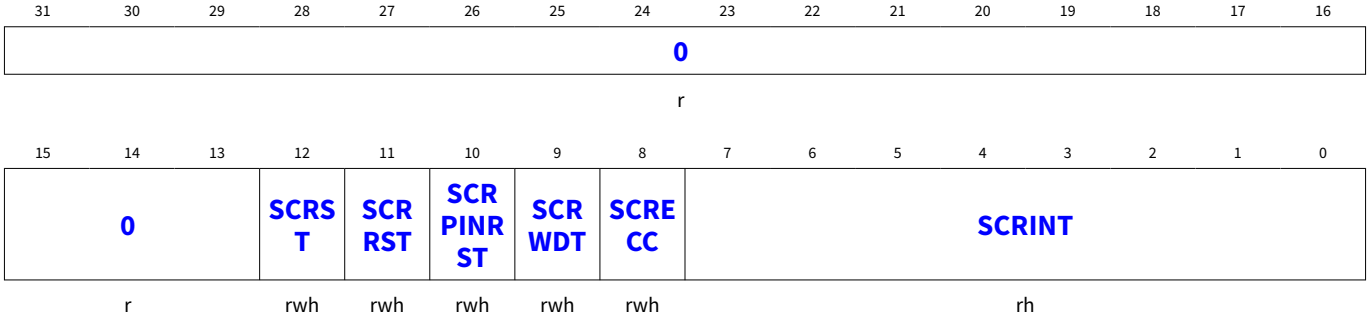
36.4.62 SCR status register

Write access protected by module-specific ("M") protection, as follows:

- When SCR_CON0.SCREN=0, write access to SCR_STAT register is blocked.
- In order to write into the SCR_STAT register, proceed as follows:
- If SCR_CON0.SCREN=0, then enable the SCR by writing SCR_CON0.SCREN=1
- Poll REGACCSTAT.FIFOEMPTY until FIFOEMPTY=1
- Then write into SCR_STAT as needed.

SCR_STAT Offset address: $92DC_H$

SCR status register LVDPMS1 Reset value: $0000\ 1000_H$



Field	Bits	Type	Description
SCRINT	7:0	rh	**Data exchange from standby controller to PMS main domain** This bit field allows fast data exchange from SCR to PMS/CPUx. The data maybe read by CPUx consequent to an interrupt from the SCR to decode the interrupt. Incase SCR is enabled, at the end of the SCR Firmware routine, a value of 80H is set in SCRINT register to indicate that SCR has finished executing the startup code.
SCRECC	8	rwh	**SCR RAM ECC error / reset flag** ***Note**: The flag is set by SCR and cleared by explicit write to the register bit. The flag is not cleared by SCR. If reset trigger by SCR RAM ECC error is enabled in SCR, this flag is set.* 0_B No ECC error / reset reported by SCR. 1_B ECC error / reset was detected in SCR RAM.
SCRWDT	9	rwh	**SCR watchdog timer error / reset flag** ***Note**: The flag is set by SCR and cleared by explicit write to the register bit. The flag is not cleared by SCR. If reset trigger by SCR WDT error is enabled in SCR, this flag is set.* 0_B No WDT error / reset reported by SCR. 1_B WDT timer error / reset reported by SCR.
SCRPINRST	10	rwh	**SCR pin reset flag** ***Note**: The flag is set by SCR and cleared by explicit write to the register bit. The flag is not cleared by SCR. When the SCR is being reset triggered by SCR pin, this flag is set.* 0_B No reset occurred in SCR. 1_B A reset has occurred in SCR.
SCRRST	11	rwh	**SCR software reset flag** ***Note**: The flag is set by SCR and cleared by explicit write to the register bit. The flag is not cleared by SCR. When the SCR is being reset triggered by SCR software, this flag is set.* 0_B No reset occurred in SCR. 1_B A reset has occurred in SCR.

(table continues...)

(continued)

Field	Bits	Type	Description
SCRST	12	rwh	**Standby controller reset indication flag** This bit is set after a power-on reset as SCR is in reset state. This bit is consequently set when a reset is issued via SCR_CON1.SCRSTREQ bit. This status flag is set on every SCR reset caused by any SCR reset source. This flag cannot be cleared while the SCR reset is active. The SCR reset is active for four PCLK cycles (SCR clock PCLK). 0_B No reset of Standby controller took place. 1_B Reset of Standby controller took place.
0	31:13	r	**Reserved** Read as 0.

36.4.63 EVRC status register 1

The EVRC status registers EVRC_STATx are updated during start-up and operation with the actual EVRC status and therefore the read value may differ from the reset value.

EVRC_STAT1 Offset address: $92E4_H$

EVRC status register 1 LVDPMS2 Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		DUTYCYCLE									FBADCV				
r		rh									rh				

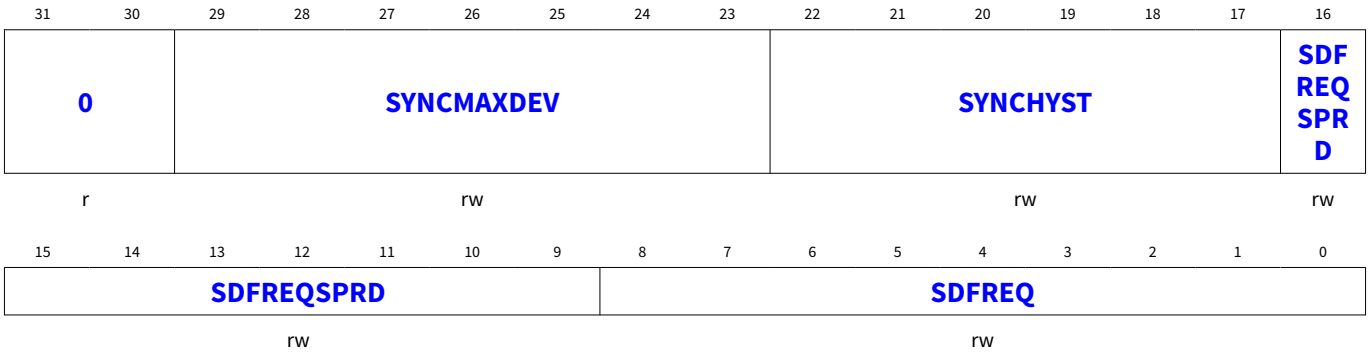
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FBADCV				FFADCV											
rh				rh											

Field	Bits	Type	Description
FFADCV	11:0	rh	**Feed-forward ADC value, after the LPF**
FBADCV	20:12	rh	**Feedback ADC value, after the LPF**
DUTYCYCLE	29:21	rh	**DPWM control output status** This bit field reflects the actual PWM output of the controller provided to the external MOSFET switches.
0	31:30	r	**Reserved** Read as 0.

36.4.64 EVRC configuration register 1

EVRC_CON1 Offset address: $92F0_H$

EVRC configuration register 1 Reset values see: Table 1468



Field	Bits	Type	Description
SDFREQ	8:0	rw	**Regulator switching frequency (or oversampling factor in PWM mode)** Sets the switching frequency during closed-loop PWM phase. The frequency can be calculated as f_{DCDC} = 100MHz / SDFREQ Note: synchronization input and frequency spreading, when enabled, may alter the value of the switching frequency during closed-loop operation, on a period-to-period basis. Only up to 3 values are supported, depending on the product variant (see f_{DCDC} parameter in the datasheet): SDFREQ = 125 for f_{DCDC} = 0.8MHz SDFREQ = 222 for f_{DCDC} = 0.45MHz SDFREQ = 55 for f_{DCDC} = 1.82MHz
SDFREQSPRD	16:9	rw	**Frequency spreading configuration** • 00_B: no spreading • 01_B: [-1, 0] clock periods randomly added to T_{DCDC} (i.e. $1/f_{DCDC}$) • 10_B: [-2, +1] clock periods randomly added to T_{DCDC} (i.e. $1/f_{DCDC}$) • 11_B: [-4, +3] clock periods randomly added to T_{DCDC} (i.e. $1/f_{DCDC}$) • 00_B: no spreading • 01_B: [-1, +1] clock periods randomly added to $T_{DCDCSYNC}$ (i.e. $1/f_{DCDCSYNC}$) • 10_B: [-3, +3] clock periods randomly added to $T_{DCDCSYNC}$ (i.e. $1/f_{DCDCSYNC}$) • 11_B: [-7, +7] clock periods randomly added to $T_{DCDCSYNC}$ (i.e. $1/f_{DCDCSYNC}$) B

(table continues...)

(continued)

Field	Bits	Type	Description
SYNCHYST	22:17	rw	**Lock and unlock hysteresis window** This bitfield defines the hysteresis window for synchronization locking and unlocking. The limit is applied to the period counter running at 100 MHz. Upper unlock condition= SDFREQ + SYNCMAXDEV Upper lock condition= SDFREQ + SYNCMAXDEV - SYNCHYST Lower unlock condition = SDFREQ - SYNCMAXDEV Lower lock condition = SDFREQ - SYNCMAXDEV + SYNCHYST SYNCHYST = round [df_{HYST} * (SDFREQ ± SYNCMAXDEV)^2 / [100MHz ± df_{HYST} * (SDFREQ ± SYNCMAXDEV)]] First hysteresis band: df_{HYST} = 100 MHz / (SDFREQ + SYNCMAXDEV - SYNCHYST) - 100MHz / (SDFREQ + SYNCMAXDEV) Second hysteresis band: df_{HYST} = 100 MHz / (SDFREQ - SYNCMAXDEV) - 100MHz / (SDFREQ - SYNCMAXDEV + SYNCHYST)
SYNCMAXDEV	29:23	rw	**Maximum deviation of the synchronization input frequency** This bitfield defines the maximum allowed frequency deviation of the synchronization input signal frequency from the programmed nominal DCDC switching frequency (EVRC_CON1.SDFREQ). Violation of limit leads to loss of synchronization.The frequency window is defined as follows df_{MAXDEV} = 100 MHz *(2*SYNCMAXDEV) / (SDFREQ^2-SYNCMAXDEV^2) SYNCMAXDEV = round [sqrt((100 MHz / df_{MAXDEV})^2 + SDFREQ^2) - (100 MHz / df_{MAXDEV})]
0	31:30	r	**Reserved** Read as 0; should be written with 0.

Table 1467 Access mode restrictions of EVRC_CON1 sorted by descending priority

Mode name	Access mode		Description
EVRC_CON0.UP == 0	rw	SDFREQ, SDFREQSPRD, SYNCHYST, SYNCMAXDEV	
(default)	r	SDFREQ, SDFREQSPRD, SYNCHYST, SYNCMAXDEV	

Table 1468 Reset values of EVRC_CON1

Reset type	Reset value	Note
LVDPMS2 reset for high-frequency configuration	098A 007D$_H$	
LVDPMS2 reset for low-frequency configuration	098A 007D$_H$	

(table continues...)

Table 1468 (continued) Reset values of EVRC_CON1

Reset type	Reset value	Note
Cold PORST for high-frequency configuration	098A 007D$_H$	
Cold PORST for low-frequency configuration	098A 007D$_H$	
After boot-FW value for low-frequency configuration	098A 007D$_H$	
After boot-FW value for high-frequency configuration	098A 007D$_H$	

36.4.65 EVRC configuration register 2

EVRC_CON2 Offset address: 92F4$_H$

EVRC configuration register 2 Reset values see: Table 1469

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0				SHLTIM				SHHTIM				SHLTHR			
r				rw				rw				rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SHLTHR				SHHTHR								0	SHOFFEN	SHLEN	SHHEN
rw				rw								r	rw	rw	rw

Field	Bits	Type	Description
SHHEN	0	rw	**Short-to-high detection enable** 0_B Short-to-high detection disabled 1_B Short-to-high detection enabled
SHLEN	1	rw	**Short-to-low detection enable** 0_B Short-to-low detection disabled 1_B Short-to-low detection enabled
SHOFFEN	2	rw	**Enable bit for EVRC switch-off upon short detection** This bit enables the switch-off of the EVRC regulator after a delay of ~3ms after an EVRC short-circuit condition has been detected (either to high-voltage or to low-voltage). 0_B EVRC is not switched off after a short detection. 1_B EVRC is switched off after a short detection.

(table continues...)

(continued)

Field	Bits	Type	Description
SHHTHR	11:4	rw	**Short-to-high detection threshold** This bit-field sets the threshold for the short to high voltage detection. The threshold value is (EVRC_CON0.SDVOUTSEL + EVRC_CON2.SHHTHR) * $V_{TRKCORELSB}$ (see data sheet parameter).
SHLTHR	19:12	rw	**Short-to-low detection threshold** This bit-field sets the threshold for the short to low voltage detection. The threshold value is: (EVRC_CON0.SDVOUTSEL - EVRC_CON2.SHLTHR) * $V_{TRKCORELSB}$ (see data sheet parameter).
SHHTIM	23:20	rw	**Timeout for short-to-high detection** This bit-field sets the time duration, in milliseconds, until a short-to-high detection is triggered. The short-to-high condition must persist for at least the configured time, in order to trigger a short detection. LSB = 1 ms
SHLTIM	27:24	rw	**Timeout for short-to-low detection** This bit-field sets the time duration, in milliseconds, until a short-to-low detection is triggered. The short-to-low condition must persist for at least the configured time, in order to trigger a short detection. LSB = 1 ms
0	3, 31:28	r	**Reserved** Read as 0; should be written with 0.

Table 1469 Reset values of EVRC_CON2

Reset type	Reset value	Note
LVDPMS2 Reset	033C 29F0$_H$	
Cold PORST	033C 29F0$_H$	
After Boot-FW Value	033C 29F0$_H$	

36.4.66 EVRC configuration register 3

EVRC_CON3 Offset address: 92F8$_H$

EVRC configuration register 3 Reset values see: Table 1470

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		VOLTOKLPF		VOLTOKTHR				0	SYNCDIVFAC			SYNCMUXSEL		SYNCOUT	SYNCIN
r		rw		rw				r	rw			rw		rw	rw

Field	Bits	Type	Description
SYNCIN	0	rw	**EVRC input synchronization enable** This bit enables the EVRC input synchronization to the external DCDCSYNCI input signal. When set to 1, the DCDC will start to lock to the external synchronization input signal. This EVRC Synchronization status is indicated in EVRC_STAT0.SYNCLCK status bits. 0_B Synchronization of EVRC switching gate outputs to external input signal is disabled. 1_B Synchronization of EVRC switching gate outputs to external input signal is enabled.
SYNCOUT	1	rw	**EVRC DCDCSYNCO output synchronization enable** This bitfield enables the synchronization output DCDCSYNCO of the EVRC, to synchronize an external SMPS regulator to the internal EVRC regulator. 0_B DC-DC Synchronisation signal not available. 1_B DC-DC Synchronisation signal available.
SYNCMUXSEL	3:2	rw	**Synchronisation input multiplexer** This bitfield selects synchronisation input either from GTM or eGTM inputs to be forwarded to EVRC SMPS regulator. 00_B Synchronization input open or unconnected. 01_B eGTM 10_B GTM 11_B Reserved
SYNCDIVFAC	6:4	rw	**Switching frequency division factor for external synchronization** This bit field defines the divider factor for the SMPS switching output to generate the DCDCSYNCO output to synchronize an external regulator to the internal EVRC regulator. The signal is routed to the pin if enabled through EVRC_CON3.SYNCOUT bit. If the divider factor is changed while the output synchronization is enabled (EVRC_CON3.SYNCOUT=1), then the first period of the synchronization output signal (after the change) can be shorter than what both, the old and the new divider factor value, indicate. In cases where this behavior is not acceptable, the divider factor must not be changed when the output synchronization is enabled. All other combinations are reserved. 000_B $f_{DCDCSYNCO} = f_{DCDC}$. The actual duty cycle is routed. 001_B $f_{DCDCSYNCO} = f_{DCDC}/2$. Duty cycle is constant at 50%. 010_B $f_{DCDCSYNCO} = f_{DCDC}/4$. Duty cycle is constant at 50%. 011_B $f_{DCDCSYNCO} = f_{DCDC}/8$. Duty cycle is constant at 50%. 100_B $f_{DCDCSYNCO} = f_{DCDC}/16$. Duty cycle is constant at 50%. 101_B $f_{DCDCSYNCO} = f_{DCDC}/32$. Duty cycle is constant at 50%. **others,** Reserved
VOLTOKTHR	11:8	rw	**Threshold level for the voltage OK logic** This bitfield controls the threshold for the dection in the Voltage OK logic. It is expressed in ADC LSBs, nominally 2.83mV.

(table continues...)

(continued)

Field	Bits	Type	Description
VOLTOKLPF	13:12	rw	**Filter coefficient for the voltage OK logic** This bitfield controls the time constant of the Voltage OK filter. Larger values correspond to lower time constant of the filter.
0	7, 31:14	r	**Reserved** Read as 0; should be written with 0.

Table 1470 Reset values of EVRC_CON3

Reset type	Reset value	Note
LVDPMS2 Reset	0000 1200$_H$	
Cold PORST	0000 1200$_H$	
After Boot-FW Value	0000 1200$_H$	

36.4.67 EVRC configuration register 4

EVRC_CON4 Offset address: 92FC$_H$

EVRC configuration register 4 Reset values see: Table 1471

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
INTGOFFS				ZEROBIN				SKPLSTHR				0		RMPFFEN	SKPLSEN
rw				rw				rw				r		rw	rw

Field	Bits	Type	Description
SKPLSEN	0	rw	**Enable skip pulse feature** Enables the possibility to skip pulses in the PWM signal, in case of detection of large overshoots on the feedback voltage. The detection threshold can be set in the SKPLSTHR bitfield. When enabled, the switching frequency is allowed to momentarily change (i.e. decrease) during transients. 0_B 0: Skip pulse disabled 0_B 1: Skip pulse enabled
RMPFFEN	1	rw	**Enable feed-forward correction of artificial ramp slope** When enabled, the slope of the artificial ramp depends on the feed-forward voltage ($V_{DDEXTDC}$). 0_B 0: disabled. The slope of the artificial ramp is constant. 0_B 1: enabled. The slope of the artificial ramp is adapted based on the VDDEXTDC monitored value.

(table continues...)

(continued)

Field	Bits	Type	Description
SKPLSTHR	7:4	rw	**Skip pulse threshold** Digital representation of the threshold level needed to trigger a skip pulse. A larger value leads to larger threshold, decreasing the likelihood of a skip pulse.
ZEROBIN	11:8	rw	**Zero-bin setting** This bit-field adds an offset to the zero-bin, used in the integrator.
INTGOFFS	15:12	rw	**Initial offset of the integrator** Preloaded value that is used to initialize the integrator before the transition to PWM. It can be used to minimize the transients occurring at the transition from open-loop to closed-loop operation.
0	3:2, 31:16	r	**Reserved** Read as 0; should be written with 0.

Table 1471 Reset values of EVRC_CON4

Reset type	Reset value	Note
LVDPMS2 Reset	0000 0060$_H$	
Cold PORST	0000 0060$_H$	
After Boot-FW Value	0000 0060$_H$	

36.4.68 EVRC SD coefficient register 0

EVRC_COEFF0 Offset address: 9300$_H$

EVRC SD coefficient register 0 Reset values see: Table 1473

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SMCDRIFTKP				FFBLPF			FBADCLPF			SMCK1FRAC3V3		SMCK1FRAC5V		SMCK3FRAC5V	
rw				rw			rw			rw		rw		rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SMCK3FRAC5V	SMCK2FRAC5V			SMCK35V				SMCK25V				SMCK1			
rw	rw			rw				rw				rw			

Field	Bits	Type	Description
SMCK1	3:0	rw	**Proportional coefficient for the voltage error in SMC** The proportional coefficient of sliding mode control. Note that the value of this register corresponds to bitshifts, i.e. multiplication/division by powers of 2.

(table continues...)

(continued)

Field	Bits	Type	Description
SMCK25V	7:4	rw	**Coefficient for the estimated current in SMC (5V and higher, nominal)** The current estimation coefficient of sliding mode control. Note that the value of this register corresponds to bitshifts, i.e. multiplication/ division by powers of 2. If a finer resolution is desired, SMCK2FRAC5V should be used.
SMCK35V	11:8	rw	**Coefficient for the integrator in SMC (5V and higher, nominal)** The integrator coefficient of sliding mode control. Note that the value of this register corresponds to bitshifts, i.e. multiplication/division by powers of 2. If a finer resolution is desired, SMCK3FRAC5V should be used.
SMCK2FRAC5V	14:12	rw	**Fractional coefficient for the estimated current in SMC (5V and higher, nominal)** 000_B Disabled 001_B Add 1/8 of K2 010_B Add 2/8 of K2 011_B Add 3/8 of K2 100_B Add 4/8 of K2 101_B Add 5/8 of K2 110_B Add 6/8 of K2 111_B Add 7/8 of K2
SMCK3FRAC5V	17:15	rw	**Fractional coefficient for the integrator in SMC (5V and higher, nominal)** 000_B Disabled 001_B Add 1/8 of K3 010_B Add 2/8 of K3 011_B Add 3/8 of K3 100_B Add 4/8 of K3 101_B Add 5/8 of K3 110_B Add 6/8 of K3 111_B Add 7/8 of K3
SMCK1FRAC5V	19:18	rw	**Fractional coefficient for the proportional coefficient for the voltage error in SMC (5V and higher, nominal)**
SMCK1FRAC3V3	21:20	rw	**Fractional coefficient for the proportional coefficient for the voltage error in SMC (3V3, nominal)**
FBADCLPF	24:22	rw	**FBADC output filter coefficient** Coefficient of the filter applied to the output of the FBADC.
FFBLPF	27:25	rw	**Filter coefficient for voltage error** Filter coefficient for the voltage error term in sliding mode control.
SMCDRIFTKP	31:28	rw	**Proportional term for the drift compensation loop** Proportional coefficient of the current drift compensation loop. Note that the value of this register corresponds to bitshifts, i.e. multiplication/division by powers of 2. If a finer resolution is desired, EVRC_COEFF1.SMCDRIFTKPFRAC should be used.

Table 1472 Access mode restrictions of EVRC_COEFF0 sorted by descending priority

Mode name	Access mode		Description
EVRC_CON0.UP == 0	rw	FBADCLPF, FFBLPF, SMCDRIFTKP, SMCK1, SMCK1FRAC3V3, SMCK1FRAC5V, SMCK25V, SMCK2FRAC5V, SMCK35V, SMCK3FRAC5V	
(default)	r	FBADCLPF, FFBLPF, SMCDRIFTKP, SMCK1, SMCK1FRAC3V3, SMCK1FRAC5V, SMCK25V, SMCK2FRAC5V, SMCK35V, SMCK3FRAC5V	

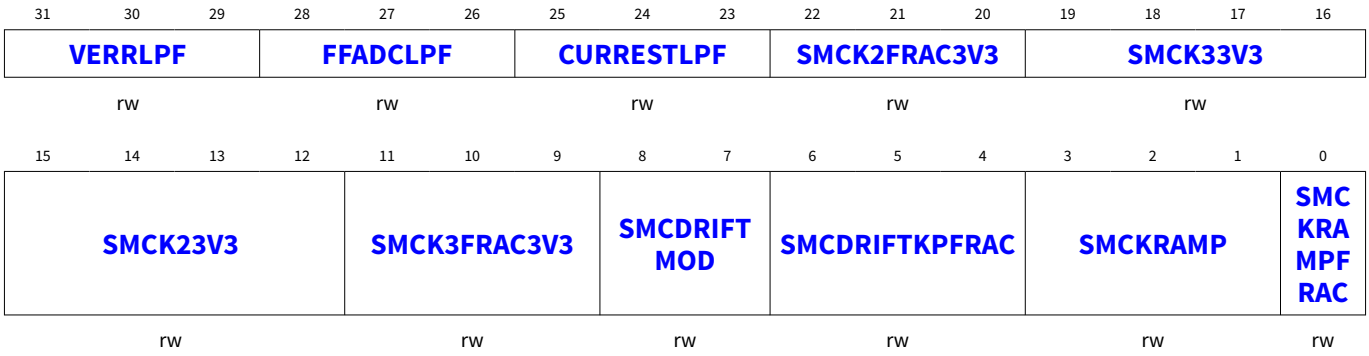
Table 1473 Reset values of EVRC_COEFF0

Reset type	Reset value	Note
LVDPMS2 reset for high-frequency configuration	8296 2014$_H$	
LVDPMS2 reset for low-frequency configuration	86A8 2124$_H$	
Cold PORST for high-frequency configuration	8296 2014$_H$	
Cold PORST for low-frequency configuration	86A8 2124$_H$	
After boot-FW value for low-frequency configuration	86A8 2124$_H$	
After boot-FW value for high-frequency configuration	8296 2014$_H$	

36.4.69 EVRC SD coefficient register 1

EVRC_COEFF1 Offset address: 9304$_H$

EVRC SD coefficient register 1 Reset values see: Table 1475



Field	Bits	Type	Description
SMCKRAMPFRAC	0	rw	**Fractional bit of artificial ramp coefficient** Fractional coefficient for the scaling factor of the artificial ramp of sliding mode control.
SMCKRAMP	3:1	rw	**Artificial ramp coefficient** Scaling factor of the artificial ramp of sliding mode control. Note that the value of this register corresponds to bitshifts, i.e. multiplication/ division by powers of 2. SMCKRAMPFRAC represents a fractional coefficient for the scaling factor, the total scaling factor is given by bits[3:1]+SMCKRAMPFRAC*0.5
SMCDRIFTKPFRAC	6:4	rw	**Fractional coefficient for the proportional term of the drift compensation loop** 000_B Disabled 001_B Add 1/8 of drift_kp 010_B Add 2/8 of drift_kp 011_B Add 3/8 of drift_kp 100_B Add 4/8 of drift_kp 101_B Add 5/8 of drift_kp 110_B Add 6/8 of drift_kp 111_B Add 7/8 of drift_kp
SMCDRIFTMOD	8:7	rw	**Drift compensation mode** 00_B Off 01_B Positive slope compensation 10_B Negative slope compensation 11_B Both slopes
SMCK3FRAC3V3	11:9	rw	**Fractional coefficient for the integrator in SMC (3V3, nominal)** 000_B Disabled 001_B Add 1/8 of K3 010_B Add 2/8 of K3 011_B Add 3/8 of K3 100_B Add 4/8 of K3 101_B Add 5/8 of K3 110_B Add 6/8 of K3 111_B Add 7/8 of K3

(table continues...)

(continued)

Field	Bits	Type	Description
SMCK23V3	15:12	rw	**Coefficient for the estimated current in SMC (3V3, nominal)** The current estimation coefficient of sliding mode control. Note that the value of this register corresponds to bitshifts, i.e. multiplication/ division by powers of 2. If a finer resolution is desired, SMCK2FRAC3V3 should be used.
SMCK33V3	19:16	rw	**Coefficient for the integrator in SMC (3V3 and higher, nominal)** The integrator coefficient of sliding mode control. Note that the value of this register corresponds to bitshifts, i.e. multiplication/division by powers of 2. If a finer resolution is desired, SMCK3FRAC3V3 should be used.
SMCK2FRAC3V3	22:20	rw	**Fractional coefficient for the estimated current in SMC (3V3, nominal)** 000_B Disabled 001_B Add 1/8 of K2 010_B Add 2/8 of K2 011_B Add 3/8 of K2 100_B Add 4/8 of K2 101_B Add 5/8 of K2 110_B Add 6/8 of K2 111_B Add 7/8 of K2
CURRESTLPF	25:23	rw	**FBADC output filter coefficient only used in the current estimator** Coefficient of the filter on the output voltage used inside the current estimator.
FFADCLPF	28:26	rw	**FFADC output filter coefficient** Coefficient of the filter applied to the output of the FFADC.
VERRLPF	31:29	rw	**FBADC output filter coefficient only used in the integrator** Coefficient of the filter on the output voltage used inside the integrator. Setting it to 0 bypasses the filter completely.

Table 1474 Access mode restrictions of EVRC_COEFF1 sorted by descending priority

Mode name	Access mode		Description
EVRC_CON0.UP == 0	rw	CURRESTLPF, FFADCLPF, SMCDRIFTKPFRAC, SMCDRIFTMOD, SMCK23V3, SMCK2FRAC3V3, SMCK33V3, SMCK3FRAC3V3, SMCKRAMP, SMCKRAMPFRAC, VERRLPF	
(default)	r	CURRESTLPF, FFADCLPF, SMCDRIFTKPFRAC, SMCDRIFTMOD, SMCK23V3, SMCK2FRAC3V3, SMCK33V3, SMCK3FRAC3V3, SMCKRAMP, SMCKRAMPFRAC, VERRLPF	

Table 1475 Reset values of EVRC_COEFF1

Reset type	Reset value	Note
LVDPMS2 reset for high-frequency configuration	0920 188E$_H$	
LVDPMS2 reset for low-frequency configuration	0921 20AA$_H$	
Cold PORST for high-frequency configuration	0920 188E$_H$	
Cold PORST for low-frequency configuration	0921 20AA$_H$	
After boot-FW value for low-frequency configuration	0921 20AA$_H$	
After boot-FW value for high-frequency configuration	0920 188E$_H$	

36.4.70 EVRC SD coefficient register 2

EVRC_COEFF2 Offset address: 9308$_H$

EVRC SD coefficient register 2 Reset values see: Table 1477

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DRV3V3NOVLRISE				DRV3V3NOVLFALL				DRV3V3NSTRCLAMP				DRV3V3NSTRON			
rw				rw				rw				rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DRV3V3NSTROFF				DRV3V3PSTRCLAMP				DRV3V3PSTRON				DRV3V3PSTROFF			
rw				rw				rw				rw			

Field	Bits	Type	Description
DRV3V3PSTROFF	3:0	rw	**Strength setting driver external PMOS (turn-off)** Driver strength used during the turn-off phase of the external PMOS. Higher values correspond to higher driver strength, i.e. lower equivalent resistance.
DRV3V3PSTRON	7:4	rw	**Strength setting driver external PMOS (turn-on)** Driver strength used during the turn-on phase of the external PMOS. Higher values correspond to higher driver strength, i.e. lower equivalent resistance.

(table continues...)

(continued)

Field	Bits	Type	Description
DRV3V3PSTRCLAMP	11:8	rw	**Strength setting driver external PMOS (clamping phase)** Driver strength used during the clamping phase of the external PMOS. The clamping phase occurs with a delay of EVRC_COEFF4.DRVPCLAMPDEL clock cycles after the turn-on phase is started. Higher values correspond to higher driver strength, i.e. lower equivalent resistance.
DRV3V3NSTROFF	15:12	rw	**Strength setting driver external NMOS (turn-off)** Driver strength used during the turn-off phase of the external NMOS. Higher values correspond to higher driver strength, i.e. lower equivalent resistance.
DRV3V3NSTRON	19:16	rw	**Strength setting driver external NMOS (turn-on)** Driver strength used during the turn-on phase of the external NMOS. Higher values correspond to higher driver strength, i.e. lower equivalent resistance.
DRV3V3NSTRCLAMP	23:20	rw	**Strength setting driver external NMOS (clamping phase)** Driver strength used during the clamping phase of the external NMOS. The clamping phase occurs with a delay of EVRC_COEFF4.DRVNCLAMPDEL clock cycles after the turn-on phase is started. Higher values correspond to higher driver strength, i.e. lower equivalent resistance.
DRV3V3NOVLFALL	27:24	rw	**Non-overlapping time (falling edge) between n-cmd and p-cmd** Non-overlapping time (expressed in clock cycles) added at the falling edge of the switching node between the turn-off of the external PMOS and the turn-on of the external NMOS in order to avoid shoot-through currents. Non-overlapping time (expressed in clock cycles) = (bits[3:1] + 1) + bit[0] * 0.5 Minimum value: 1 clock cycle
DRV3V3NOVLRISE	31:28	rw	**Non-overlapping time (rising edge) between n-cmd and p-cmd** Non-overlapping time (expressed in clock cycles) added at the rising edge of the switching node between the turn-off of the external NMOS and the turn-on of the external PMOS in order to avoid shoot-through currents. Non-overlapping time (expressed in clock cycles) = (bits[3:1] + 1) + bit[0] * 0.5 Minimum value: 1 clock cycle

Table 1476 Access mode restrictions of EVRC_COEFF2 sorted by descending priority

Mode name	Access mode		Description
EVRC_CON0.UP == 0	rw	DRV3V3NOVLFALL, DRV3V3NOVLRISE, DRV3V3NSTRCLAMP, DRV3V3NSTROFF, DRV3V3NSTRON, DRV3V3PSTRCLAMP, DRV3V3PSTROFF, DRV3V3PSTRON	
(default)	r	DRV3V3NOVLFALL, DRV3V3NOVLRISE, DRV3V3NSTRCLAMP, DRV3V3NSTROFF, DRV3V3NSTRON, DRV3V3PSTRCLAMP, DRV3V3PSTROFF, DRV3V3PSTRON	

Table 1477 Reset values of EVRC_COEFF2

Reset type	Reset value	Note
LVDPMS2 reset for high-frequency configuration	13F7 EFBF$_H$	
LVDPMS2 reset for low-frequency configuration	13F7 EFBF$_H$	
Cold PORST for high-frequency configuration	13F7 EFBF$_H$	
Cold PORST for low-frequency configuration	13F7 EFBF$_H$	
After boot-FW value for low-frequency configuration	13F7 EFBF$_H$	
After boot-FW value for high-frequency configuration	13F7 EFBF$_H$	

36.4.71 EVRC SD coefficient register 3

EVRC_COEFF3 Offset address: 930C$_H$

EVRC SD coefficient register 3 Reset values see: Table 1479

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DRV5VNOVLRISE				DRV5VNOVLFALL				DRV5VNSTRCLAMP				DRV5VNSTRON			
rw				rw				rw				rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DRV5VNSTROFF				DRV5VPSTRCLAMP				DRV5VPSTRON				DRV5VPSTROFF			
rw				rw				rw				rw			

Field	Bits	Type	Description
DRV5VPSTROFF	3:0	rw	**Strength setting driver external PMOS (turn-off)** Driver strength used during the turn-off phase of the external PMOS. Higher values correspond to higher driver strength, i.e. lower equivalent resistance.
DRV5VPSTRON	7:4	rw	**Strength setting driver external PMOS (turn-on)** Driver strength used during the turn-on phase of the external PMOS. Higher values correspond to higher driver strength, i.e. lower equivalent resistance.
DRV5VPSTRCLAMP	11:8	rw	**Strength setting driver external PMOS (clamping phase)** Driver strength used during the clamping phase of the external PMOS. The clamping phase occurs with a delay of EVRC_COEFF4.DRVPCLAMPDEL clock cycles after the turn-on phase is started. Higher values correspond to higher driver strength, i.e. lower equivalent resistance.
DRV5VNSTROFF	15:12	rw	**Strength setting driver external NMOS (turn-off)** Driver strength used during the turn-off phase of the external NMOS. Higher values correspond to higher driver strength, i.e. lower equivalent resistance.
DRV5VNSTRON	19:16	rw	**Strength setting driver external NMOS (turn-on)** Driver strength used during the turn-on phase of the external NMOS. Higher values correspond to higher driver strength, i.e. lower equivalent resistance.
DRV5VNSTRCLAMP	23:20	rw	**Strength setting driver external NMOS (clamping phase)** Driver strength used during the clamping phase of the external NMOS. The clamping phase occurs with a delay of EVRC_COEFF4.DRVNCLAMPDEL clock cycles after the turn-on phase is started. Higher values correspond to higher driver strength, i.e. lower equivalent resistance.
DRV5VNOVLFALL	27:24	rw	**Non-overlapping time (falling edge) between n-cmd and p-cmd** Non-overlapping time (expressed in clock cycles) added at the falling edge of the switching node between the turn-off of the external PMOS and the turn-on of the external NMOS in order to avoid shoot-through currents. Non-overlapping time (expressed in clock cycles) = (bits[3:1] + 1) + bit[0] * 0.5 Minimum value: 1 clock cycle

(table continues...)

(continued)

Field	Bits	Type	Description
DRV5VNOVLRISE	31:28	rw	**Non-overlapping time (rising edge) between n-cmd and p-cmd** Non-overlapping time (expressed in clock cycles) added at the rising edge of the switching node between the turn-off of the external NMOS and the turn-on of the external PMOS in order to avoid shoot-through currents. Non-overlapping time (expressed in clock cycles) = (bits[3:1] + 1) + bit[0] * 0.5 Minimum value: 1 clock cycle

Table 1478 Access mode restrictions of EVRC_COEFF3 sorted by descending priority

Mode name	Access mode		Description
EVRC_CON0.UP == 0	rw	DRV5VNOVLFALL, DRV5VNOVLRISE, DRV5VNSTRCLAMP, DRV5VNSTROFF, DRV5VNSTRON, DRV5VPSTRCLAMP, DRV5VPSTROFF, DRV5VPSTRON	
(default)	r	DRV5VNOVLFALL, DRV5VNOVLRISE, DRV5VNSTRCLAMP, DRV5VNSTROFF, DRV5VNSTRON, DRV5VPSTRCLAMP, DRV5VPSTROFF, DRV5VPSTRON	

Table 1479 Reset values of EVRC_COEFF3

Reset type	Reset value	Note
LVDPMS2 reset for high-frequency configuration	35F8 CF9F$_H$	
LVDPMS2 reset for low-frequency configuration	35F8 CF9F$_H$	
Cold PORST for high-frequency configuration	35F8 CF9F$_H$	
Cold PORST for low-frequency configuration	35F8 CF9F$_H$	
After boot-FW value for low-frequency configuration	35F8 CF9F$_H$	

(table continues...)

Table 1479 (continued) Reset values of EVRC_COEFF3

Reset type	Reset value	Note
After boot-FW value for high-frequency configuration	35F8 CF9F$_H$	

36.4.72 EVRC SD coefficient register 4

EVRC_COEFF4 Offset address: 9310$_H$

EVRC SD coefficient register 4 Reset values see: Table 1481

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								MINTOFF				MINTON			
r								rw				rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DRVFFHYS				DRVFFTHRES				DRVNCLAMPDEL				DRVPCLAMPDEL			
rw				rw				rw				rw			

Field	Bits	Type	Description
DRVPCLAMPDEL	3:0	rw	**Delay for clamping external PMOS** Delay, expressed in clock cycles, from the start of the switching phase after which clamping is applied, i.e. the strength DRVxxxPSTR is used.
DRVNCLAMPDEL	7:4	rw	**Delay for clamping external NMOS** Delay, expressed in clock cycles, from the start of the switching phase after which clamping is applied, i.e. the strength DRVxxxNSTR is used.
DRVFFTHRES	11:8	rw	**Driver feed-forward threshold** Threshold used to switch between 3V3 and 5V driver configuration based on feedforward voltage ($V_{DDEXTDC}$). The corresponding voltage can be calculated as: 2.83mV * (1166 + 32 * DRVFFTHRES) which results in a range from 3.3V to 4.65V.
DRVFFHYS	15:12	rw	**Driver feed-forward hysteresis** Hysteresis band applied when switching from 5V to 3V3 configuration. The corresponding voltage can be calculated as 2.83mV * 16 * DRVFFHYS.
MINTON	19:16	rw	**Minimum on time during PWM operation** This bit-field configures the minimum on-time within one switching period during closed loop operation. Expressed as number of 100MHz clock periods. The minimum on-time will be ignored if skip pulse is enabled and the skip pulse conditions are verified. MINTON = max(floor(EVRC_COEFF2.DRV3V3NOVLRISE[3:1] or EVRC_COEFF3.DRV5VNOVLRISE[3:1], depending on application voltage level), EVRC_COEFF5.DRVNBOOSTOFFDEL) + 1

(table continues...)

(continued)

Field	Bits	Type	Description
MINTOFF	23:20	rw	**Minimum off time during PWM operation** This bit-field configures the minimum off-time within one switching period during closed loop operation. Expressed as number of 100MHz clock periods. MINTOFF = max(floor(EVRC_COEFF2.DRV3V3NOVLFALL[3:1] or EVRC_COEFF3.DRV5VNOVLFALL[3:1], depending on application voltage level), EVRC_COEFF5.DRVPBOOSTOFFDEL) + 1
0	31:24	r	**Reserved** Read as 0; should be written with 0.

Table 1480 Access mode restrictions of EVRC_COEFF4 sorted by descending priority

Mode name	Access mode		Description
EVRC_CON0.UP == 0	rw	DRVFFHYS, DRVFFTHRES, DRVNCLAMPDEL, DRVPCLAMPDEL, MINTOFF, MINTON	
(default)	r	DRVFFHYS, DRVFFTHRES, DRVNCLAMPDEL, DRVPCLAMPDEL, MINTOFF, MINTON	

Table 1481 Reset values of EVRC_COEFF4

Reset type	Reset value	Note
LVDPMS2 reset for high-frequency configuration	$0077\ 4665_H$	
LVDPMS2 reset for low-frequency configuration	$0077\ 4665_H$	
Cold PORST for high-frequency configuration	$0077\ 4665_H$	
Cold PORST for low-frequency configuration	$0077\ 4665_H$	
After boot-FW value for low-frequency configuration	$0077\ 4665_H$	
After boot-FW value for high-frequency configuration	$0077\ 4665_H$	

36.4.73 EVRC SD coefficient register 5

EVRC_COEFF5 Offset address: 9314_H

EVRC SD coefficient register 5 Reset values see: Table 1483

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								DRVNBOOSTOFFDEL		DRVNBOOSTONDEL		DRVPBOOSTOFFDEL		DRVPBOOSTONDEL	
r								rw		rw		rw		rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DRV5VNSTRBOOST				DRV5VPSTRBOOST				DRV3V3NSTRBOOST				DRV3V3PSTRBOOST			
rw				rw				rw				rw			

Field	Bits	Type	Description
DRV3V3PSTRBOOST	3:0	rw	**Strength setting driver external PMOS (boost phase)** Driver strength used during the boost phase of the external PMOS. Higher values correspond to higher driver strength, i.e. lower equivalent resistance.
DRV3V3NSTRBOOST	7:4	rw	**Strength setting driver external NMOS (boost phase)** Driver strength used during the boost phase of the external NMOS. Higher values correspond to higher driver strength, i.e. lower equivalent resistance.
DRV5VPSTRBOOST	11:8	rw	**Strength setting driver external PMOS (boost phase)** Driver strength used during the boost phase of the external PMOS. Higher values correspond to higher driver strength, i.e. lower equivalent resistance.
DRV5VNSTRBOOST	15:12	rw	**Strength setting driver external NMOS (boost phase)** Driver strength used during the boost phase of the external NMOS. Higher values correspond to higher driver strength, i.e. lower equivalent resistance.
DRVPBOOSTONDEL	17:16	rw	**Boosting-time of external PMOS (turn-on)** Time duration, expressed in clock periods, where DRVxxxPSTRBOOST strength is applied at the beginning of the turn-on phase.
DRVPBOOSTOFFDEL	19:18	rw	**Boosting-time of external PMOS (turn-off)** Time duration, expressed in clock periods, where DRVxxxPSTRBOOST strength is applied at the beginning of the turn-off phase. EVRC_COEFF5.DRVPBOOSTOFFDEL <= min(EVRC_COEFF2.DRV3V3NOVLFALL(3:1)+1, EVRC_COEFF3.DRV5VNOVLFALL(3:1)+1)
DRVNBOOSTONDEL	21:20	rw	**Boosting-time of external NMOS (turn-on)** Time duration, expressed in clock periods, where DRVxxxNSTRBOOST strength is applied at the beginning of the turn-on phase.

(table continues...)

(continued)

Field	Bits	Type	Description
DRVNBOOSTOFFDEL	23:22	rw	**Boosting-time of external NMOS (turn-off)** Time duration, expressed in clock periods, where DRVxxxNSTRBOOST strength is applied at the beginning of the turn-off phase. EVRC_COEFF5.DRVNBOOSTOFFDEL <= min(EVRC_COEFF2.DRV3V3NOVLRISE(3:1)+1, EVRC_COEFF3.DRV5VNOVLRISE(3:1)+1)
0	31:24	r	**Reserved** Read as 0; should be written with 0.

Table 1482 Access mode restrictions of EVRC_COEFF5 sorted by descending priority

Mode name	Access mode		Description
EVRC_CON0.UP == 0	rw	DRV3V3NSTRBOOST, DRV3V3PSTRBOOST, DRV5VNSTRBOOST, DRV5VPSTRBOOST, DRVNBOOSTOFFDEL, DRVNBOOSTONDEL, DRVPBOOSTOFFDEL, DRVPBOOSTONDEL	
(default)	r	DRV3V3NSTRBOOST, DRV3V3PSTRBOOST, DRV5VNSTRBOOST, DRV5VPSTRBOOST, DRVNBOOSTOFFDEL, DRVNBOOSTONDEL, DRVPBOOSTOFFDEL, DRVPBOOSTONDEL	

Table 1483 Reset values of EVRC_COEFF5

Reset type	Reset value	Note
LVDPMS2 Reset	0000 FFFF$_H$	
Cold PORST	0000 FFFF$_H$	
After Boot-FW Value	0000 FFFF$_H$	

36.4.74 RTC configuration register 1

Before accessing, check the REGACCSTAT.FIFOEMPTY and REGACCSTAT.FIFOFULL status as described in the user manual (LVD reset section).

RTC_CON1 Offset address: 931C$_H$

RTC configuration register 1 Reset values see: Table 1484

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OFFSET															
rwh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DIV															
rw															

Field	Bits	Type	Description
DIV	15:0	rw	**Clock divider** Up to 2^16 clock divider. The clock frequency is divided through DIV+1. If the RTC input frequency is 32kHz, to generate a timer period of e.g. 1s, use DIV = $7FFF_H$ (for division by 32768).
OFFSET	31:16	rwh	**Signed offset value**

Table 1484 Reset values of RTC_CON1

Reset type	Reset value	Note
LVDANA Reset	$0000\ 0000_H$	
LVDPMS0 Reset	$0000\ 0000_H$	

36.4.75 RTC configuration register 2

Before accessing, check the REGACCSTAT.FIFOEMPTY and REGACCSTAT.FIFOFULL status as described in the user manual (LVD reset section).

RTC_CON2 Offset address: 9320_H

RTC configuration register 2 Reset values see: Table 1485

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0			MSTART1					0			MSIZE1				
r			rwh					r			rwh				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			MSTART0					0			MSIZE0				
r			rw					r			rw				

Field	Bits	Type	Description
MSIZE0	4:0	rw	**Compare register size for CMP0** This bit field determines the number of bits in register CMP0 (starting from bit 0) that are used for the compare operation. 00_H CMP0[0] used for compare operation 01_H CMP0[1:0] used for compare operation $1E_H$ CMP0[30:0] used for compare operation $1F_H$ CMP0[31:0] used for compare operation

(table continues...)

(continued)

Field	Bits	Type	Description
MSTART0	12:8	rw	**Start bit location for CMP0** This bit field determines the lowest bit number of the 64-bit RTC that is compared with the content of register CMP0 bit 0. The number of bits to be compared is defined by bit field MSIZE0. 00_H RTC[0] is the lowest bit number 01_H RTC[1] is the lowest bit number $1E_H$ RTC[30] is the lowest bit number $1F_H$ RTC[31] is the lowest bit number
MSIZE1	20:16	rwh	**Compare register size for CMP1** This bit field determines the number of bits in register CMP1 (starting from bit 0) that are used for the compare operation. 00_H CMP1[0] used for compare operation 01_H CMP1[1:0] used for compare operation $1E_H$ CMP1[30:0] used for compare operation $1F_H$ CMP1[31:0] used for compare operation
MSTART1	28:24	rwh	**Start bit location for CMP1** This bit field determines the lowest bit number of the 64-bit RTC that is compared with the content of register CMP1 bit 0. The number of bits to be compared is defined by bit field MSIZE1. 00_H RTC[0] is the lowest bit number 01_H RTC[1] is the lowest bit number $1E_H$ RTC[30] is the lowest bit number $1F_H$ RTC[31] is the lowest bit number
0	7:5, 15:13, 23:21, 31:29	r	**Reserved** Read as 0; should be written with 0.

Table 1485 Reset values of RTC_CON2

Reset type	Reset value	Note
LVDANA Reset	$0000\ 0000_H$	
LVDPMS0 Reset	$0000\ 0000_H$	

36.4.76 RTC compare register 0

Before accessing, check the REGACCSTAT.FIFOEMPTY and REGACCSTAT.FIFOFULL status as described in the user manual (LVD reset section).

RTC_CMP0 Offset address: 9324_H

RTC compare register 0 Reset values see: Table 1486

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CMPVAL0															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CMPVAL0															
rw															

Field	Bits	Type	Description
CMPVAL0	31:0	rw	**Compare value [31:0]** This bit-field holds up to 32 bits of the compare value.

Table 1486 Reset values of RTC_CMP0

Reset type	Reset value	Note
LVDANA Reset	0000 0000$_H$	
LVDPMS0 Reset	0000 0000$_H$	

36.4.77 RTC compare register 1

Before accessing, check the REGACCSTAT.FIFOEMPTY and REGACCSTAT.FIFOFULL status as described in the user manual (LVD reset section).

RTC_CMP1 Offset address: 9328$_H$

RTC compare register 1 Reset values see: Table 1487

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CMPVAL1															
rwh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CMPVAL1															
rwh															

Field	Bits	Type	Description
CMPVAL1	31:0	rwh	**Compare value [63:32]** This bit-field holds up to 32 bits of the compare value.

Table 1487 Reset values of RTC_CMP1

Reset type	Reset value	Note
LVDANA Reset	0000 0000$_H$	
LVDPMS0 Reset	0000 0000$_H$	

36.4.78 RTC timer counter register 0

RTC_TIM0 Offset address: 932C$_H$

RTC timer counter register 0 Reset values see: Table 1488

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TIMER															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TIMER															
rh															

Field	Bits	Type	Description
TIMER	31:0	rh	**Timer[31:0]** Lower 32-bit timer capture value.

Table 1488 Reset values of RTC_TIM0

Reset type	Reset value	Note
LVDANA Reset	0000 0000_H	
LVDPMS0 Reset	0000 0000_H	

36.4.79 RTC timer capture register

RTC_TIM1CAP Offset address: 9330_H

RTC timer capture register LVDPMS2 Reset value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TIMER															
rh															

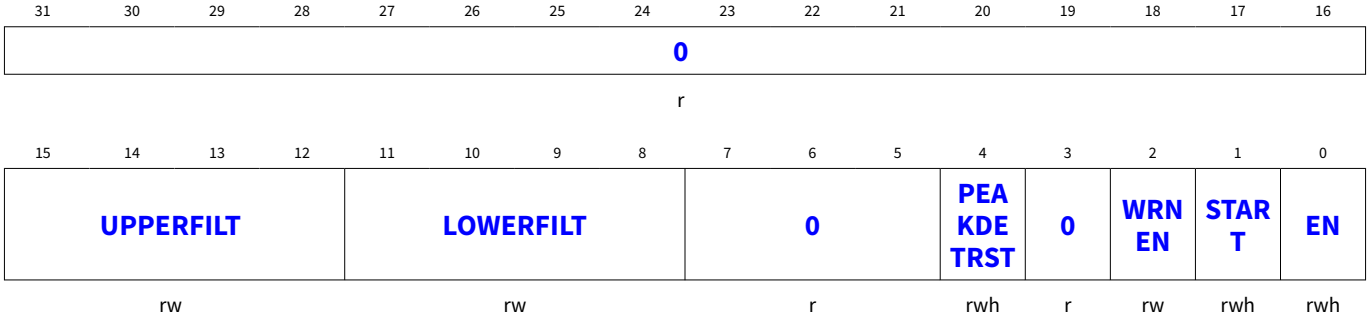
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TIMER															
rh															

Field	Bits	Type	Description
TIMER	31:0	rh	**Timer[63:32]** Synchronous upper 32-bit capture when RTC_TIM0 register is read.

36.4.80 Die temperature sensor control register

DTS_CON Offset address: 9334_H

Die temperature sensor control register Reset values see: Table 1489



Field	Bits	Type	Description
EN	0	rwh	**DTS enable** This field is used to enable the DTS analog circuitry. 0_B DTS is disabled 1_B DTS is enabled
START	1	rwh	**ADC conversion control** This field is used to start DTS conversions. 0_B DTS conversion is stopped or stopping 1_B DTS conversion is started
WRNEN	2	rw	**Warning enable** This field is used as a global enable for all warnings/alarms. 0_B All warnings are disabled 1_B All warnings are enabled
PEAKDETRST	4	rwh	**DTS peak temperature detector reset** This field, when written with '1', resets the peak temperature detector. The bit is cleared to 0 after the peak temperature detector has been reset. 0_B Peak temperature detector is not reset 1_B Peak temperature detector is reset
LOWERFILT	11:8	rw	**Temperature-sensing mode warning filter** This field configures the filter length for generation of LLU warning/ alarm. Warning/alarm is triggered after RESULT goes under LOWER by (LOWERFILT + 1) consecutive conversions.
UPPERFILT	15:12	rw	**Temperature-sensing mode warning filter** This field configures the filter length for generation of UOF warning/ alarm. Warning/alarm is triggered after RESULT exceeds UPPER by (UPPERFILT + 1) consecutive conversions.
0	3, 7:5, 31:16	r	**Reserved** Read as 0; should be written with 0.

Table 1489 Reset values of DTS_CON

Reset type	Reset value	Note
LVDPMS1 Reset	0000 0000$_H$	
After Boot-FW Value	0000 0003$_H$	

36.4.81 Die temperature sensor limit register 1

DTS_LIM1 Offset address: 9338$_H$

Die temperature sensor limit register 1 LVDPMS1 Reset value: 07FF 0469$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	UPPERHYST			UPPER											
r	rw			rw											

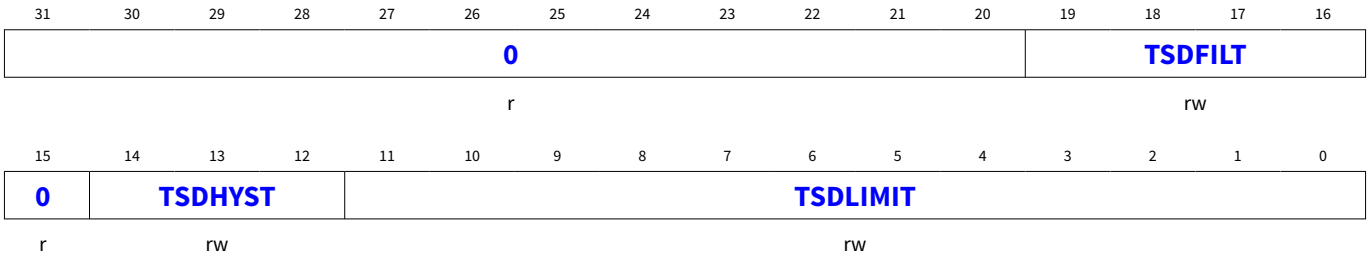
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	LOWERHYST			LOWER											
r	rw			rw											

Field	Bits	Type	Description
LOWER	11:0	rw	**DTS lower warning threshold** This field configures the falling threshold for LLU warning/alarm. (Default at -37°C)
LOWERHYST	14:12	rw	**Temperature-sensing mode warning hysteresis** This field configures the hysteresis for rising threshold of LLU warning/alarm, defined by (LOWER + LOWERHYST).
UPPER	27:16	rw	**DTS upper warning threshold** This field configures the rising threshold for UOF warning/alarm. (Default at 155°C)
UPPERHYST	30:28	rw	**Temperature-sensing mode warning hysteresis** This field configures the hysteresis for falling threshold of UOF warning/alarm, defined by (UPPER - UPPERHYST).
0	15, 31	r	**Reserved** Read as 0; should be written with 0.

36.4.82 Die temperature sensor limit register 2

DTS_LIM2 Offset address: 933C$_H$

Die temperature sensor limit register 2 LVDPMS1 Reset value: 0000 0850$_H$



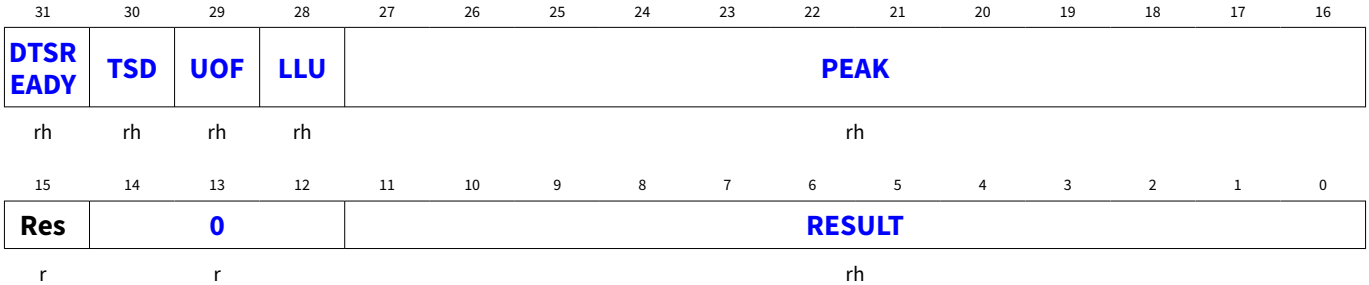
Field	Bits	Type	Description
TSDLIMIT	11:0	rw	**Temperature shut down warning threshold** This field configures the rising threshold for Temperature Shut down warning/alarm. (Default at 172°C)
TSDHYST	14:12	rw	**Temperature-sensing mode warning hysteresis** This field configures the hysteresis for falling threshold of TSD, defined by (TSDLIMIT - TSDHYST).
TSDFILT	19:16	rw	**Temperature-sensing mode warning filter** This field configures the filter length for generation of TSD warning/ alarm. Warning is triggered after RESULT exceeds TSDLIMIT by (TSDFILT + 1) consecutive conversions.
0	15, 31:20	r	**Reserved** Read as 0; should be written with 0.

36.4.83 Die temperature sensor status register

The DTS status register value is updated regularly by a polling state machine which periodically polls the local DTS status registers and transfers the data every 200us interval. The status register is updated ~5us after the conversion is completed at the local DTS instance.

DTS_STAT Offset address: 9340_H

Die temperature sensor status register LVDPMS1 Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
RESULT	11:0	rh	**DTS result** This field contains the last temperature measurement result. This can be converted to temperature (in Kelvins) by dividing the readout by 4.781.

(table continues...)

(continued)

Field	Bits	Type	Description
PEAK	27:16	rh	**DTS peak temperature result** This field contains the peak temperature readout. This can be converted to temperature (Kelvins) by dividing by 4.781.
LLU	28	rh	**Low temperature warning** This field indicates if an under-temperature warning is active. 0_B Warning not active Last DTS result > DTSxLIM1.LOWER. 1_B Warning triggered Last DTS result ≤ DTSxLIM1.LOWER.
UOF	29	rh	**High temperature warning** This field indicates if an over-temperature warning is active. 0_B Warning not active Last DTS result < DTSxLIM1.UPPER. 1_B Warning triggered Last DTS result ≥ DTSxLIM1.UPPER.
TSD	30	rh	**Thermal shutdown warning** This field indicates if a thermal shutdown warning is active. 0_B Warning not active Last DTS result < DTSxLIM2.TSDLIMIT. 1_B Warning triggered Last DTS result ≥ DTSxLIM2.TSDLIMIT.
DTSREADY	31	rh	**Ready indicator** This bit is cleared when DTS is disabled, and will be set only after we have the first DTS conversion result. Any alarm generation is gated with this bit, to avoid interrupts/alarms from triggering before the first conversion is complete. 0_B DTS is disabled, or still starting up 1_B DTS has completed startup
0	14:12	r	**Reserved** Read as 0.

36.4.84 Cold reset configuration register

CLDRSTCON Offset address: 9350_H

Cold reset configuration register LVDPMS1 Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

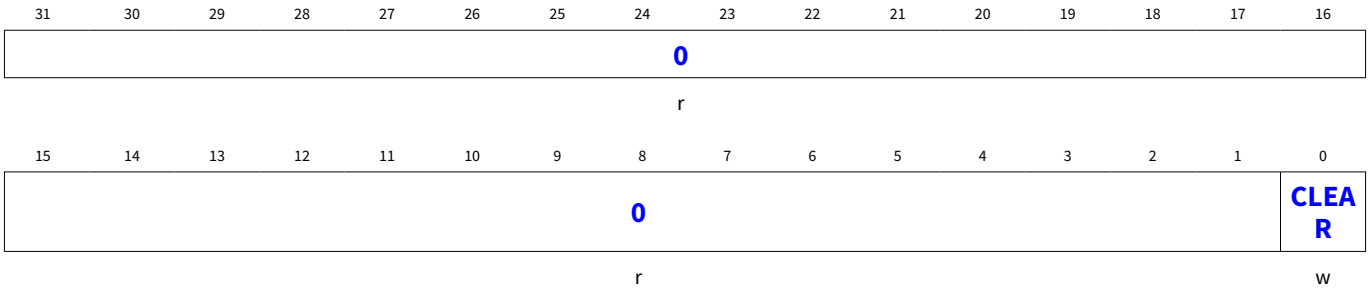
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								CPTRIG							
r								w							

Field	Bits	Type	Description
CPTRIG	7:0	w	**Cold PORST trigger** Writing the 10100101 (0xA5) sequence into this bit-field triggers a cold PORST.
0	31:8	r	**Reserved** Read as 0; should be written with 0.

36.4.85 Cold reset status clear register

CLDRSTSTATCLR Offset address: 9360_H

Cold reset status clear register LVDPMS2 Reset value: $0000\ 0000_H$

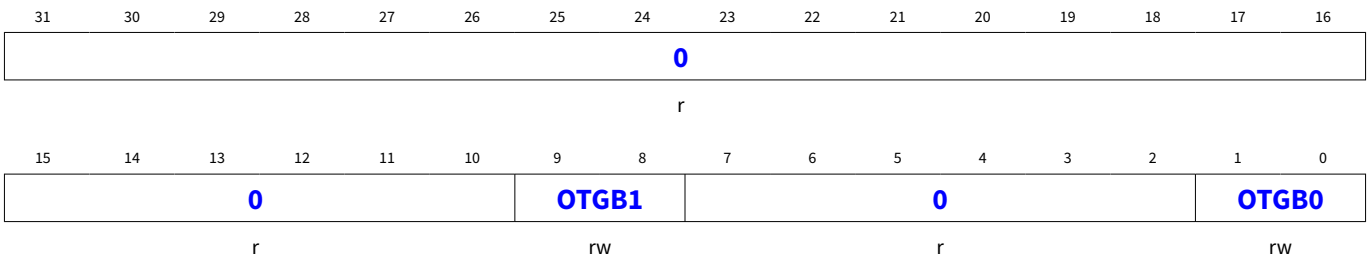


Field	Bits	Type	Description
CLEAR	0	w	**Clear cold reset status bits** This bit simultaneously clears the sticky status bits which may indicate any previous cold reset. 0_B No action 1_B Status bits cleared.
0	31:1	r	**Reserved** Read as 0; should be written with 0.

36.4.86 OCDS trigger set select register

OTSS Offset address: 9364_H

OCDS trigger set select register Reset values see: Table 1490



Field	Bits	Type	Description
OTGB0	1:0	rw	**Trigger set for OTGB0** 00_B No trigger set selected 01_B PMS trigger set 10_B EVRC trigger set **others,** reserved
OTGB1	9:8	rw	**Trigger set for OTGB1** 00_B No trigger set selected 01_B PMS trigger set 10_B EVRC trigger set **others,** reserved
0	7:2, 31:10	r	**Reserved** Read as 0; must be written with 0.

Table 1490 Reset values of OTSS

Reset type	Reset value	Note
LVDPMS2 Reset	$0000\ 0000_H$	
Debug Reset	$0000\ 0000_H$	

36.4.87 OCDS trigger set control 1 register

OTSC1 Offset address: $936C_H$

OCDS trigger set control 1 register Reset values see: Table 1491

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0											Res				
r											r				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				B1EVRC				0				B0EVRC			
r				rw				r				rw			

Field	Bits	Type	Description
B0EVRC	3:0	rw	**OTGB0 EVRC trigger set selection** 0_H No trigger selected 1_H EVRC_STAT1.FFADCV 2_H EVRC_STAT1.FBADCV 3_H EVRC_STAT0 status bits 4_H EVRC_STAT1.DUTYCYCLE 5_H DROOP **others,** Reserved (no trigger selected)

(table continues...)

(continued)

Field	Bits	Type	Description
B1EVRC	11:8	rw	**OTGB1 EVRC trigger set selection** 0_H No trigger selected 1_H EVRC_STAT1.FFADCV 2_H EVRC_STAT1.FBADCV 3_H EVRC_STAT0 status bits 4_H EVRC_STAT1.DUTYCYCLE 5_H DROOP **others,** Reserved (no trigger selected)
0	7:4, 15:12, 31:21	r	**Reserved** Read as 0; must be written with 0.

Table 1491 Reset values of OTSC1

Reset type	Reset value	Note
LVDPMS2 Reset	0000 0000$_H$	
Debug Reset	0000 0000$_H$	

36.4.88 Register to store user info in PMS1 domain and reset by LVDPMS1

User data register (cleared only on LVDPMS1 reset). This may be used by an application to store information.

LVDPMS1USRINFO Offset address: 9370$_H$

Register to store user info in PMS1 domain and reset by LVDPMS1 Reset values see: Table 1492

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DATA															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA															
rw															

Field	Bits	Type	Description
DATA	31:0	rw	**User data** User defined data.

Table 1492 Reset values of LVDPMS1USRINFO

Reset type	Reset value	Note
LVDPMS1 Reset	0000 0000$_H$	
After Boot-FW Value	0000 0000 0000 0000 0000 0000 0000 0–––$_B$	The lowest 3 bits may contain a non-zero value if LBIST was executed at start-up.

36.4.89 Register to store user info in PMS2 domain and reset by cold PORST

User data register (cleared only on cold PORST). This may be used by an application to store information.

CLDPORSTUSRINFO Offset address: 9374_H

Register to store user info in PMS2 domain and reset by cold PORST Reset values see: Table 1493

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DATA															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA															
rw															

Field	Bits	Type	Description
DATA	31:0	rw	**User data** User defined data.

Table 1493 Reset values of CLDPORSTUSRINFO

Reset type	Reset value	Note
LVDPMS2 Reset	$0000\ 0000_H$	
Cold PORST	$0000\ 0000_H$	

36.4.90 Register to store user info in PMS2 domain and reset by warm PORST

User data register (cleared on warm PORST). This may be used by an application to store information.

WRMPORSTUSRINFO Offset address: 9378_H

Register to store user info in PMS2 domain and reset by warm PORST Reset values see: Table 1494

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DATA															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA															
rw															

Field	Bits	Type	Description
DATA	31:0	rw	**User data** User defined data.

Table 1494 Reset values of WRMPORSTUSRINFO

Reset type	Reset value	Note
LVDPMS2 Reset	$0000\ 0000_H$	
Warm PORST	$0000\ 0000_H$	

36.5 Debug information

This section describes the debug features available for the PMS functional block.

36.5.1 OCDS Trigger Bus (OTGB) Interface

This section describes the PMS signals and internal status information available on the debugging interface.

The PMS OTGB interface provides debugging information including:

- Primary voltage monitors ADC measurement values for all VMONP channels
- Secondary voltage monitors ADC measurement values for V_{DDPMS0} and V_{DDPMS2}
- Die temperature sensor (DTSPMS) measurement values
- Peak voltage detection measurement values
- Wake-up timer counter values
- RTC counter values
- EVRC voltages and status information

The next figure shows an overview of the interface, register bit-fields for selecting the debug output, and the available debug triggers:

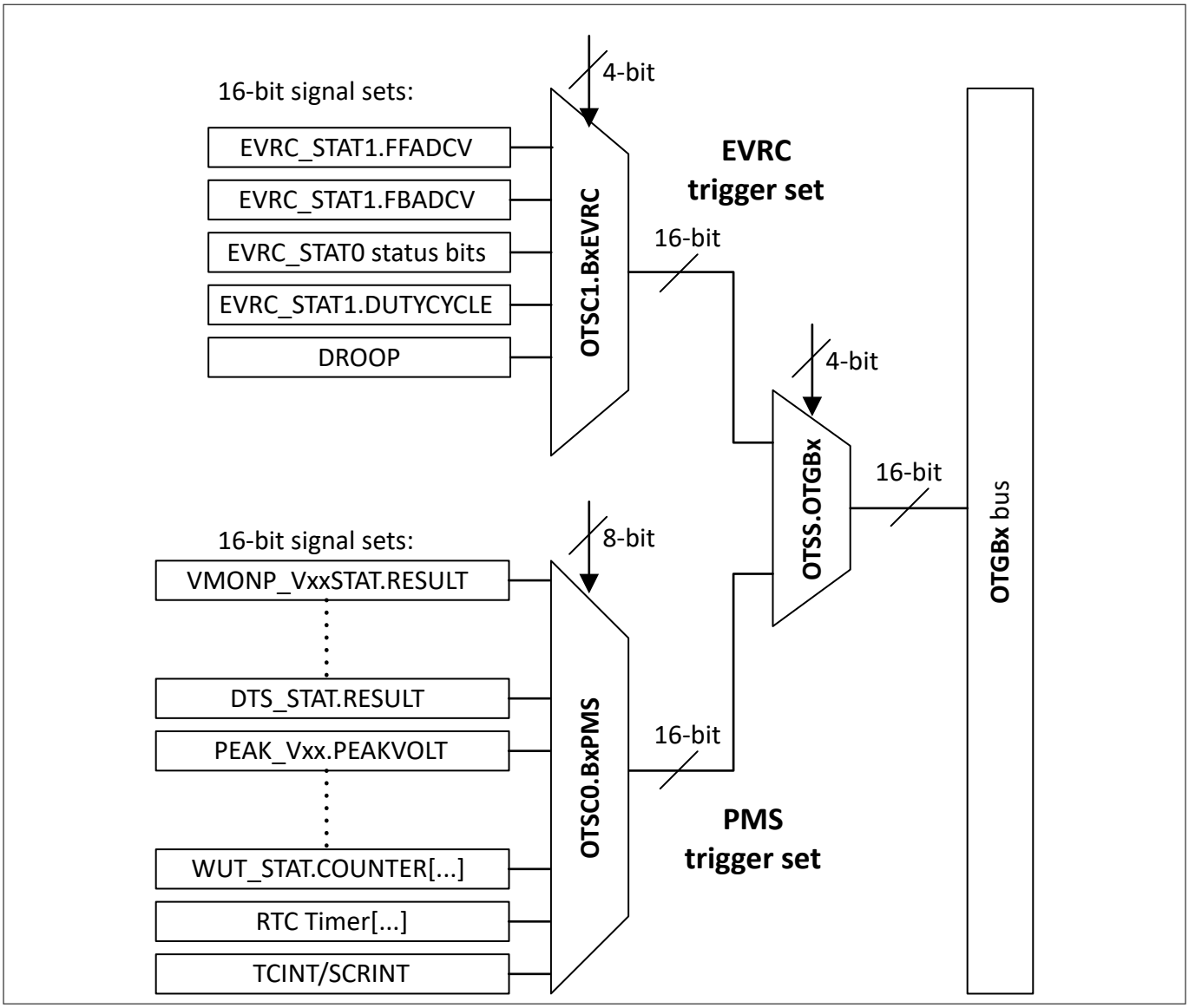


Figure 807 Overview of the PMS OTGB interface

The PMS OTGB interface module has two 16-bit trigger sets: the PMS and the EVRC trigger set. Each trigger set can be selected with the OTSS.OTGBx register bit-fields, for mapping to either OTGB0 or OTGB1 buses.

Table 1495 PMS trigger set

OTSC0.BxPMS selection	Bits	Description
VMONP_VDDSTAT.RESULT	[10:0]	Primary voltage monitor measurement result
VMONP_VDDEXTSTAT.RESULT	[10:0]	Primary voltage monitor measurement result
VMONP_VSSDCHSSTAT.RESULT	[10:0]	Primary voltage monitor measurement result
VMONP_VDDEXTDCSTAT.RESULT	[10:0]	Primary voltage monitor measurement result
VMONP_VDDDCLSSTAT.RESULT	[10:0]	Primary voltage monitor measurement result
VMONP_VDDEVRSBSTAT.RESULT	[10:0]	Primary voltage monitor measurement result
VMONP_VDDMSTAT.RESULT	[10:0]	Primary voltage monitor measurement result

(table continues...)

Table 1495 (continued) PMS trigger set

OTSC0.BxPMS selection	Bits	Description
VMONP_VDDPHPHY0STAT.RESULT	[10:0]	Primary voltage monitor measurement result
VMONP_VDDPHPHY1STAT.RESULT	[10:0]	Primary voltage monitor measurement result
VMONP_VDDPHRIF0STAT.RESULT	[10:0]	Primary voltage monitor measurement result
VMONP_VDDPHPHY2STAT.RESULT	[10:0]	Primary voltage monitor measurement result
VMONP_VDDP3NVMSTAT.RESULT	[10:0]	Primary voltage monitor measurement result
VMONP_VDDHSIFSTAT.RESULT	[10:0]	Primary voltage monitor measurement result
VMONP_VDDPADSTAT.RESULT	[10:0]	Primary voltage monitor measurement result
VMONP_VDDCPUASTAT.RESULT	[10:0]	Primary voltage monitor measurement result
VMONP_VDDLMU0STAT.RESULT	[10:0]	Primary voltage monitor measurement result
VMONP_VDDCPUBSTAT.RESULT	[10:0]	Primary voltage monitor measurement result
VMONP_VDDLMU1STAT.RESULT	[10:0]	Primary voltage monitor measurement result
VMONP_VDDPPUSTAT.RESULT	[10:0]	Primary voltage monitor measurement result
VMONP_VDDPHY0STAT.RESULT	[10:0]	Primary voltage monitor measurement result
VMONP_VDDPHY1STAT.RESULT	[10:0]	Primary voltage monitor measurement result
VMONP_VDDRIF0STAT.RESULT	[10:0]	Primary voltage monitor measurement result
VMONP_VDDPHY2STAT.RESULT	[10:0]	Primary voltage monitor measurement result
VMPONS_VDDPMS0STAT.RESULT	[10:0]	Secondary voltage monitor measurement result
VMPONS_VDDPMS2STAT.RESULT	[10:0]	Secondary voltage monitor measurement result
DTS_STAT.RESULT	[11:0]	PMS temperature sensor measurement result
PEAK_VDD.PEAKVOLT	[10:0]	Peak voltage detection measurement result
PEAK_VDDEXT.PEAKVOLT	[10:0]	Peak voltage detection measurement result
PEAK_VDDEXTDC.PEAKVOLT	[10:0]	Peak voltage detection measurement result
WUT_STAT.COUNTER[11:0]	[11:0]	Lower half of WUT counter value
WUT_STAT.COUNTER[23:12]	[11:0]	Upper half of WUT counter value
RTC Timer[15:0]	[15:0]	Part of the RTC timer value
RTC Timer[31:16]	[15:0]	Part of the RTC timer value
RTC Timer[47:32]	[15:0]	Part of the RTC timer value
RTC Timer[63:48]	[15:0]	Part of the RTC timer value
TCINT/SCRINT	[7:0] SCRINT [15:8] TCINT	PMS and SCR output and input bus interface

Table 1496 EVRC trigger set

OTSC1.BxEVRC selection	Bits	Description
EVRC_STAT1.FFADCV	[11:0]	EVRC input voltage value (feed-forward ADC measurement result)
EVRC_STAT1.FBADCV	[10:0]	EVRC output voltage value (feedback ADC measurement result, including offset compensation)
EVRC_STAT0 Status Bits	[0] EVRCSYNCIN	EVRC synchronization input signal
	[1] EVRC_STAT0.SYNCLCK	EVRC input synchronization locked status
	[2] EVRC_STAT0.EVRCSHLV	Short to ground flag
	[3] EVRC_STAT0.EVRCSHHV	Short to supply flag
	[12:4] EVRC_STAT0.SETPOINT	EVRC target output voltage
	[13] EVRC_STAT0.SDVOK	EVRC voltage OK status
EVRC_STAT1.DUTYCYCLE	[8:0]	PWM duty cycle provided to the external MOSFET switches
DROOP	[3:0] DROOPSTEP	Droop compensation amplitude
	[4] DROOPDIR	Droop compensation direction (0=down, 1=up)

36.6 References

This section is not applicable for the module being described.

36.7 PMS revision history

Reference	Description of change(s)
Date range: 2025-03-28 to 2024-02-03	
Recommended settings	• Changed value of EVRC_COEFF3.DRV5VNSTRON bit-field
Standby controller extension RAM (XRAM)	• Added section with information on SCR XRAM access protection

36.8 TC4Dx PMS information

36.8.1 TC4Dx PMS configuration

There are no deviations from the generic specification.

36.8.2 TC4Dx PMS features

There are no deviations from the generic specification.

36.8.3 TC4Dx PMS functional description

This part provides additional information about the functionality of the PMS for the TC4Dx product variant.

36.8.3.1 TC4Dx power supply rails and distribution

This section describes the power supply rails implemented on the TC4Dx. Refer to the generic functional description for typical values.

The following external supply rails are available:

- $V_{DDEVRSB}$
- V_{DDEXT}
- V_{DDM}
- $V_{DDEXTDC}$
- V_{DD}
- $V_{DDPHPHY0}$
- $V_{DDPHPHY1}$
- $V_{DDPHPHY2}$
- V_{DDPHY0}
- V_{DDPHY1}
- V_{DDPHY2}
- V_{DDHSIF}

The following internal supply rails are available:

- V_{DDPMS0}
- V_{DDPMS1}
- V_{DDPMS2}
- V_{DDDCLS}
- V_{SSDCHS}
- V_{DDLMU0}
- V_{DDLMU1}
- V_{DDPPU}
- V_{DDPAD}

Supply rail mapping for V_{DDCPUA} is not applicable for TC4Dx.

The following figure provides an overview of the supply rails listed above and in the generic PMS functional description.

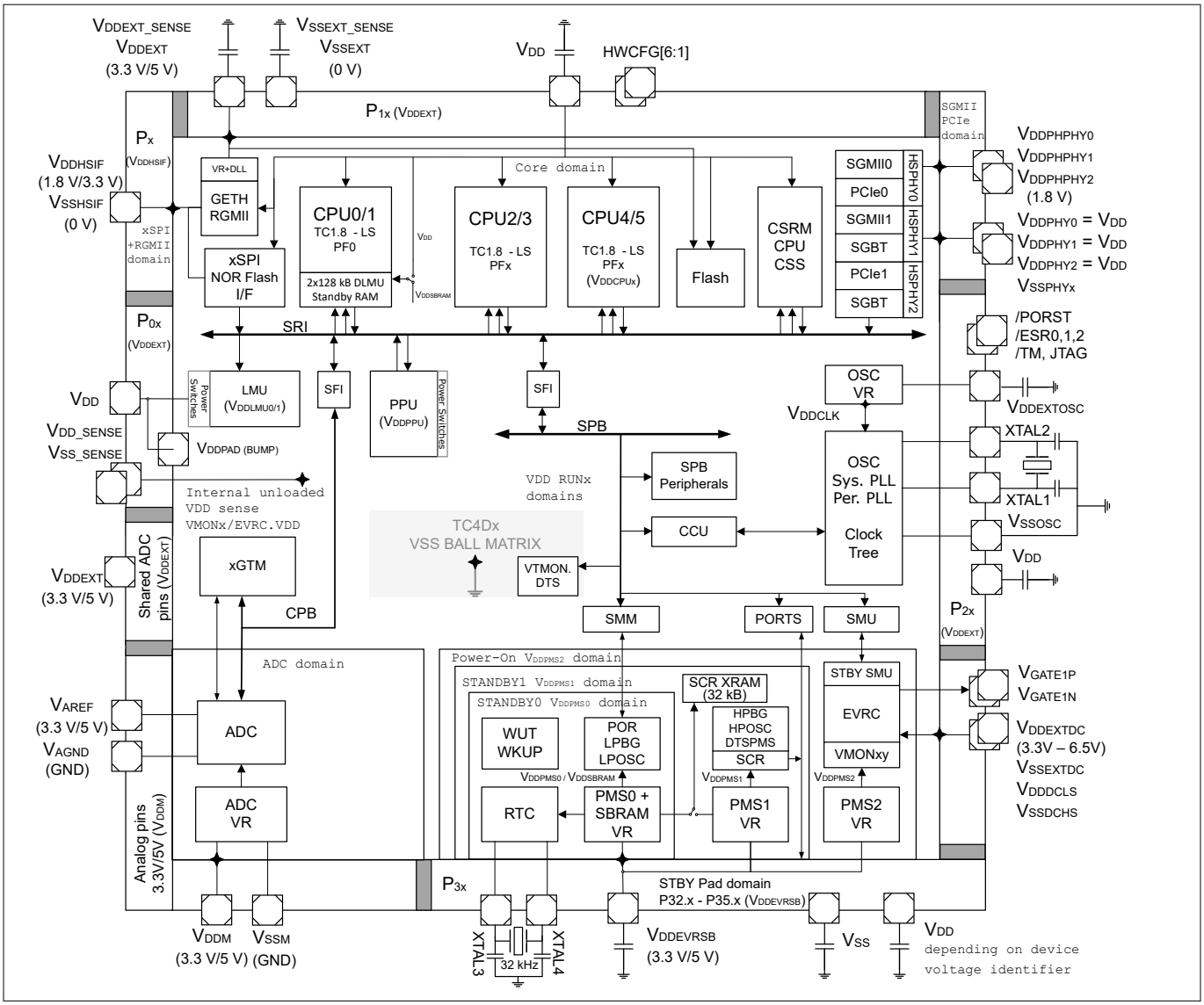


Figure 808 TC4Dx supply rails overview

Related information

Power supply rails and distribution on page 5948

36.8.3.2 TC4Dx independent standby supply domains (PMS0, PMS1, PMS2)

This section describes the standby supply domains for the power management system.

The PMS standby domains include the components shown in the figure below. Refer to the generic functional description for a list of the domains and included functions.

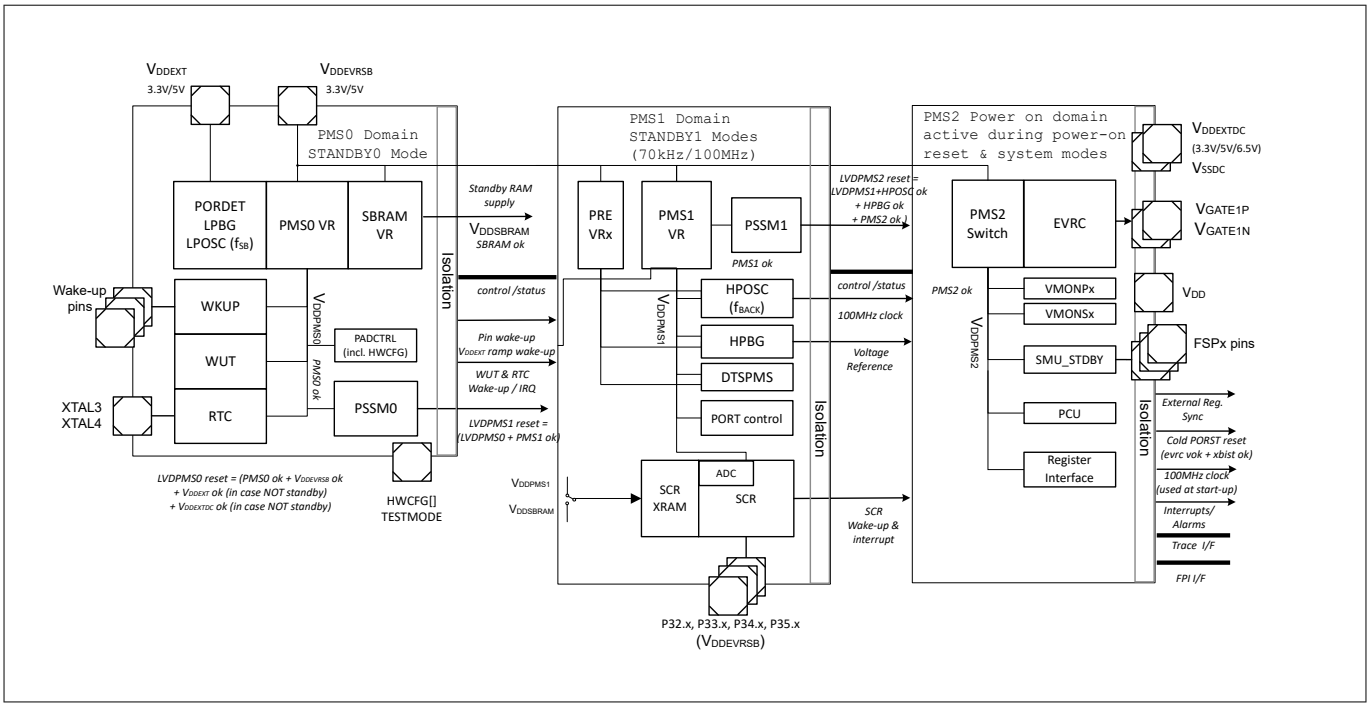


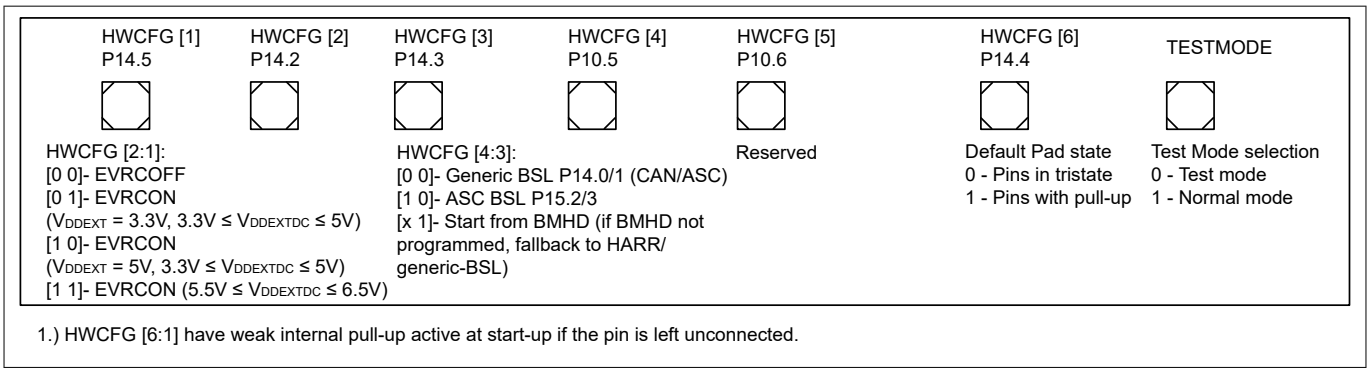
Figure 809 Standby supply domains of the power management system

Related information

Independent standby supply domains (PMS0, PMS1, PMS2) on page 5956

36.8.3.3 TC4Dx supply mode selection

This section provides additional details related to the supply selection for the TC4Dx product variant.



1.) HWCFG [6:1] have weak internal pull-up active at start-up if the pin is left unconnected.

Figure 810 Hardware configuration (HWCFG) pins for TC4Dx product variant

Related information

Supply mode selection on page 5959

36.8.3.4 TC4Dx primary voltage monitors

This section provides additional details related to the primary voltage monitors VMONPx and cold PORST for the TC4Dx product variant.

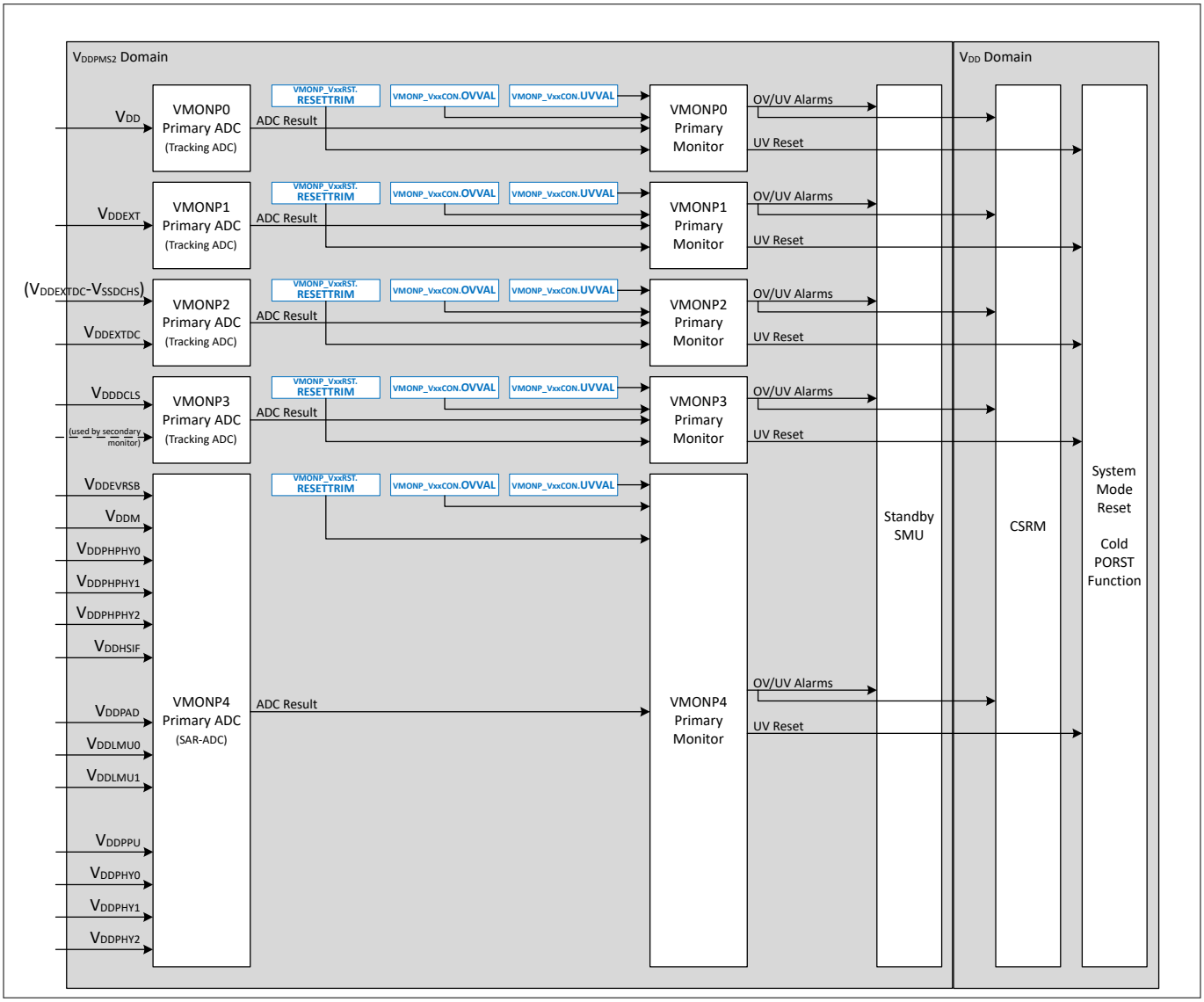


Figure 811 Overview of the primary voltage monitors for the TC4Dx product variant

Related information

Primary voltage monitors VMONPx and cold PORST on page 5971
Peak voltage detection on page 6006
Drift monitoring on page 6007

36.8.3.5 TC4Dx secondary over- and under-voltage monitors

This section provides additional details related to the secondary over- and under-voltage monitors (VMONSx) for the TC4Dx product variant.

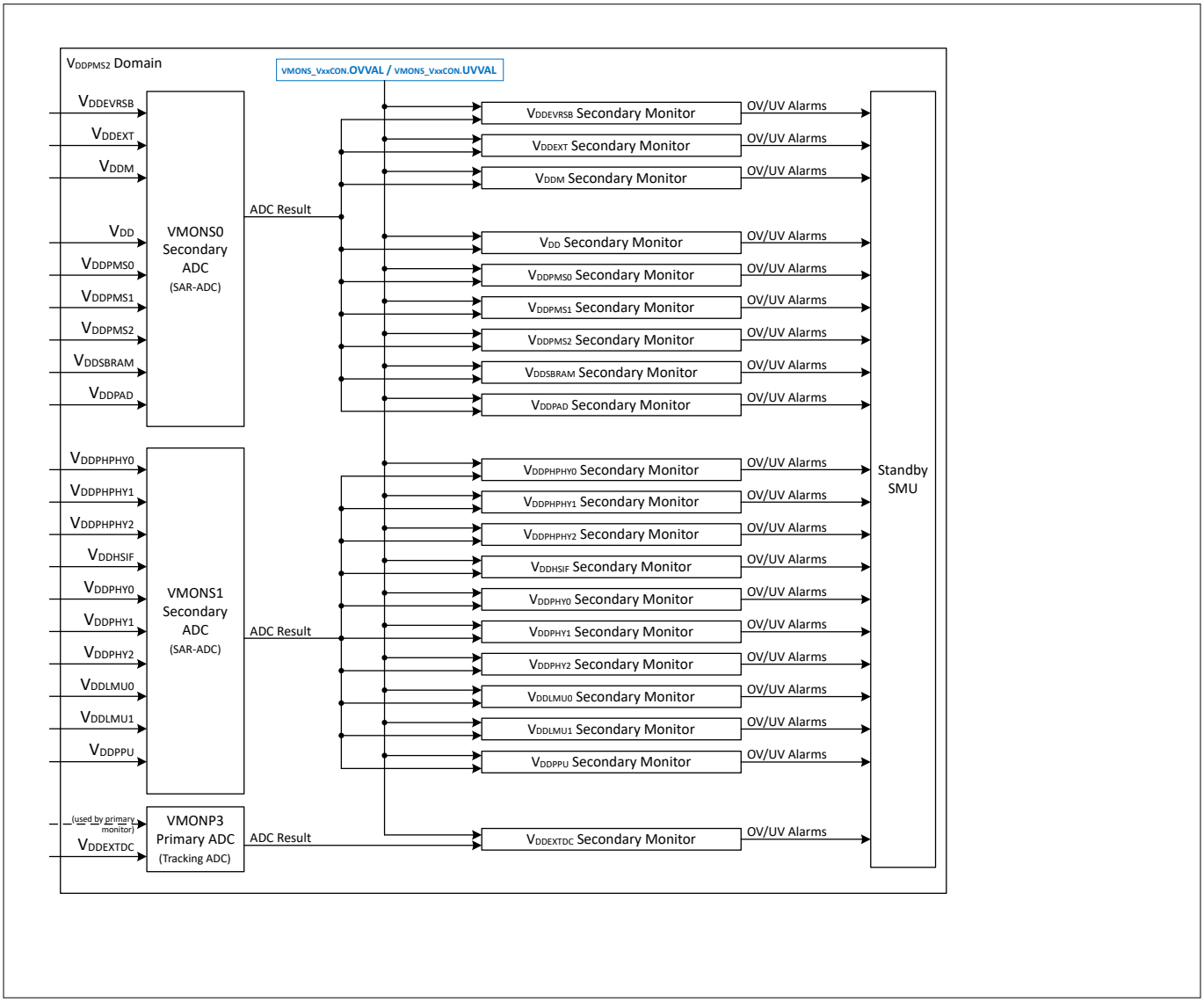


Figure 812 Overview of the secondary voltage monitors for the TC4Dx product variant

Related information

Secondary over- and under-voltage monitors (VMONSx) on page 6003

36.8.3.6 TC4Dx interrupts

This section provides additional details related to the interrupt sources and events for the TC4Dx product variant.

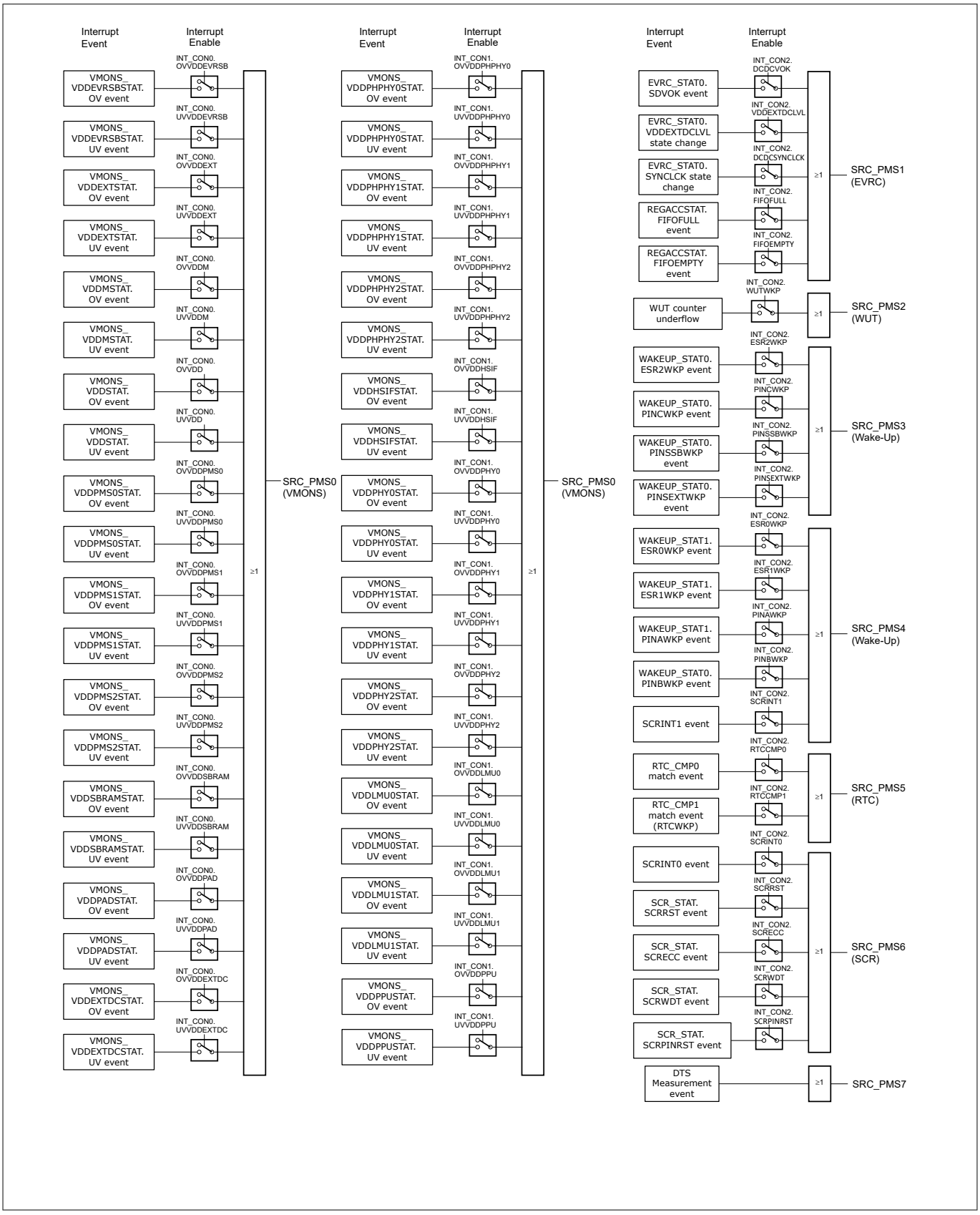


Figure 813 Interrupt sources and events for the TC4Dx product variant

Related information

Interrupts on page 6010

36.8.4 TC4Dx PMS registers

There are no deviations from the generic specification.

36.8.4.1 Memory overview tables of PMS

Table 1497 Memory overview - PMS (ascending address)

Short name	Long name	Address
PMS_XRAM	Standby controller extension RAM (8000_H Byte)	$F0240000_H$

36.8.4.2 Register address space - PMS

Table 1498 Registers address space - PMS

Module	Base address	End address	Note
PMS	$F0240000_H$	$F0249FFF_H$	FPI slave interface

36.8.4.3 Register overview - access mode glossary

Table 1499 Register overview - access mode glossary

Keyword	Description
CSE	Access protection using PROT register PMS_PROTCSE .
E	Access protection using PROT register PMS_PROTE .
SE	Access protection using PROT register PMS_PROTSE .
APU-P	Protection group consisting of registers PMS_ACCEN_WRA , PMS_ACCEN_WRB , PMS_ACCEN_RDA , PMS_ACCEN_RDB , PMS_ACCEN_VM , PMS_ACCEN_PRS .
P	Access protection using *APU-P* registers.
APU-Pcs	Protection group consisting of registers PMS_ACCENCS_WRA , PMS_ACCENCS_WRB , PMS_ACCENCS_RDA , PMS_ACCENCS_RDB , PMS_ACCENCS_VM , PMS_ACCENCS_PRS .
Pcs	Access protection using *APU-Pcs* registers.
U	No access restrictions.
PROT	Access restrictions as defined in the PROT register access rules.
BE	Always returns a Bus Error.
STP	Access only during start-up, before start-up protection is set (SMM_STCON.ST_PROT=1).
M	Indicates a module specific access condition. Refer to the register description for details of the specific access condition.

36.8.4.4 Registers overview - PMS (ascending offset address)

Table 1500 Registers overview - PMS (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PMS_PROTE	PROT register endinit	9000_H	U	SV, PROT	Application Reset	6044
PMS_PROTSE	PROT register safe endinit	9004_H	U	SV, PROT	Application Reset	6046
PMS_ID	Module identification register	9008_H	P	BE	Application Reset	6048
PMS_ACCEN_WRA	Write access enable register A	$900C_H$	U	SE, SV	Application Reset	6048
PMS_ACCEN_WRB	Write access enable register B	9010_H	U	SE, SV	Application Reset	6049
PMS_ACCEN_RDA	Read access enable register A	9014_H	U	SE, SV	Application Reset	6049
PMS_ACCEN_RDB	Read access enable register B	9018_H	U	SE, SV	Application Reset	6050
PMS_ACCEN_VM	VM access enable register	$901C_H$	U	SE, SV	Application Reset	6050
PMS_ACCEN_PRS	PRS access enable register	9020_H	U	SE, SV	Application Reset	6051
PMS_REGACCSTAT	Register access status register	9024_H	P	BE	LVDPMS2 Reset	6052
PMS_PROTCSE	PROT register cyber security endinit	9028_H	U	SV, PROT	Application Reset	6053
PMS_ACCENCS_WRA	Write access enable register A	$902C_H$	U	CSE, SV	Application Reset	6054
PMS_ACCENCS_WRB	Write access enable register B	9030_H	U	CSE, SV	Application Reset	6055
PMS_ACCENCS_RDA	Read access enable register A	9034_H	U	CSE, SV	Application Reset	6055
PMS_ACCENCS_RDB	Read access enable register B	9038_H	U	CSE, SV	Application Reset	6056
PMS_ACCENCS_VM	VM access enable register	$903C_H$	U	CSE, SV	Application Reset	6056
PMS_ACCENCS_PRS	PRS access enable register	9040_H	U	CSE, SV	Application Reset	6057
PMS_VMONP_VDDCON	VDD primary voltage monitor configuration register	9044_H	P	SV, E, P	See 6166	6166
PMS_VMONP_VDDRST	VDD primary voltage monitor reset configuration register	9048_H	Pcs	SV, SE, Pcs	See 6167	6167

(table continues...)

Table 1500 (continued) Registers overview - PMS (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PMS_VMONP_VDDCLDRST	Power domain cold reset configuration register	$904C_H$	Pcs	SV, SE, Pcs	See 6168	6168
PMS_VMONP_VDDSTAT	VDD primary voltage monitor status register	9050_H	P	BE	See 6169	6169
PMS_VMONP_VDDEXTCON	VDDEXT primary voltage monitor configuration register	9054_H	P	SV, E, P	See 6170	6170
PMS_VMONP_VDDEXTRST	VDDEXT primary voltage monitor reset configuration register	9058_H	Pcs	SV, SE, Pcs	See 6172	6172
PMS_VMONP_VDDEXTSTAT	VDDEXT primary voltage monitor status register	$905C_H$	P	BE	See 6173	6173
PMS_VMONP_VSSDCHSCON	Primary voltage monitor configuration register for the (VDDEXTDC - VSSDCHS) voltage difference	9060_H	P	SV, E, P	See 6174	6174
PMS_VMONP_VSSDCHSRST	Primary voltage monitor reset configuration register for the (VDDEXTDC - VSSDCHS) voltage difference	9064_H	Pcs	SV, SE, Pcs	See 6176	6176
PMS_VMONP_VSSDCHSSTAT	Primary voltage monitor status register for the (VDDEXTDC - VSSDCHS) voltage difference	9068_H	P	BE	See 6177	6177
PMS_VMONP_VDDEXTDCCON	VDDEXTDC primary voltage monitor configuration register	$906C_H$	P	SV, E, P	See 6178	6178
PMS_VMONP_VDDEXTDCRST	VDDEXTDC primary voltage monitor reset configuration register	9070_H	Pcs	SV, SE, Pcs	See 6180	6180
PMS_VMONP_VDDEXTDCSTAT	VDDEXTDC primary voltage monitor status register	9074_H	P	BE	See 6182	6182
PMS_VMONP_VDDDCLSCON	VDDDCLS primary voltage monitor configuration register	9078_H	P	SV, E, P	See 6183	6183
PMS_VMONP_VDDDCLSRST	VDDDCLS primary voltage monitor reset configuration register	$907C_H$	Pcs	SV, SE, Pcs	See 6185	6185
PMS_VMONP_VDDDCLSSTAT	VDDDCLS primary voltage monitor status register	9080_H	P	BE	See 6186	6186
PMS_VMONP_VDDEVRSBCON	VDDEVRSB primary voltage monitor configuration register	9084_H	P	SV, E, P	See 6187	6187

(table continues...)

Table 1500 (continued) Registers overview - PMS (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PMS_VMONP_VDDEVRSBRST	VDDEVRSB primary voltage monitor reset configuration register	9088_H	Pcs	SV, SE, Pcs	See 6189	6189
PMS_VMONP_VDDEVRSBSTAT	VDDEVRSB primary voltage monitor status register	$908C_H$	P	BE	See 6190	6190
PMS_VMONP_VDDMCON	VDDM primary voltage monitor configuration register	9090_H	P	SV, E, P	See 6191	6191
PMS_VMONP_VDDMRST	VDDM primary voltage monitor reset configuration register	9094_H	Pcs	SV, SE, Pcs	See 6193	6193
PMS_VMONP_VDDMSTAT	VDDM primary voltage monitor status register	9098_H	P	BE	See 6194	6194
PMS_VMONP_VDDPHPHY0CON	VDDPHPHY0 primary voltage monitor configuration register	$909C_H$	P	SV, E, P	See 6195	6195
PMS_VMONP_VDDPHPHY0RST	VDDPHPHY0 primary voltage monitor reset configuration register	$90A0_H$	Pcs	SV, SE, Pcs	See 6196	6196
PMS_VMONP_VDDPHPHY0STAT	VDDPHPHY0 primary voltage monitor status register	$90A4_H$	P	BE	See 6198	6198
PMS_VMONP_VDDPHPHY1CON	VDDPHPHY1 primary voltage monitor configuration register	$90A8_H$	P	SV, E, P	See 6199	6199
PMS_VMONP_VDDPHPHY1RST	VDDPHPHY1 primary voltage monitor reset configuration register	$90AC_H$	Pcs	SV, SE, Pcs	See 6200	6200
PMS_VMONP_VDDPHPHY1STAT	VDDPHPHY1 primary voltage monitor status register	$90B0_H$	P	BE	See 6202	6202
PMS_VMONP_VDDPHPHY2CON	VDDPHPHY2 primary voltage monitor configuration register	$90B4_H$	P	SV, E, P	See 6203	6203
PMS_VMONP_VDDPHPHY2RST	VDDPHPHY2 primary voltage monitor reset configuration register	$90B8_H$	Pcs	SV, SE, Pcs	See 6204	6204
PMS_VMONP_VDDPHPHY2STAT	VDDPHPHY2 primary voltage monitor status register	$90BC_H$	P	BE	See 6206	6206
PMS_VMONP_VDDHSIFCON	VDDHSIF primary voltage monitor configuration register	$90C0_H$	P	SV, E, P	See 6207	6207

(table continues...)

Table 1500 (continued) Registers overview - PMS (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PMS_VMONP_VDDHSIFRST	VDDHSIF primary voltage monitor reset configuration register	$90C4_H$	Pcs	SV, SE, Pcs	See 6208	6208
PMS_VMONP_VDDHSIFSTAT	VDDHSIF primary voltage monitor status register	$90C8_H$	P	BE	See 6210	6210
PMS_VMONP_VDDPADCON	VDDPAD primary voltage monitor configuration register	$90D8_H$	P	SV, E, P	See 6211	6211
PMS_VMONP_VDDPADRST	VDDPAD primary voltage monitor reset configuration register	$90DC_H$	Pcs	SV, SE, Pcs	See 6212	6212
PMS_VMONP_VDDPADSTAT	VDDPAD primary voltage monitor status register	$90E0_H$	P	BE	See 6214	6214
PMS_VMONP_VDDLMU0CON	VDDLMU0 primary voltage monitor configuration register	$90E4_H$	P	SV, E, P	See 6215	6215
PMS_VMONP_VDDLMU0RST	VDDLMU0 primary voltage monitor reset configuration register	$90E8_H$	Pcs	SV, SE, Pcs	See 6216	6216
PMS_VMONP_VDDLMU0STAT	VDDLMU0 primary voltage monitor status register	$90EC_H$	P	BE	See 6217	6217
PMS_VMONP_VDDLMU1CON	VDDLMU1 primary voltage monitor configuration register	$90F0_H$	P	SV, E, P	See 6218	6218
PMS_VMONP_VDDLMU1RST	VDDLMU1 primary voltage monitor reset configuration register	$90F4_H$	Pcs	SV, SE, Pcs	See 6219	6219
PMS_VMONP_VDDLMU1STAT	VDDLMU1 primary voltage monitor status register	$90F8_H$	P	BE	See 6221	6221
PMS_VMONP_VDDPPUCON	VDDPPU primary voltage monitor configuration register	$90FC_H$	P	SV, E, P	See 6222	6222
PMS_VMONP_VDDPPURST	VDDPPU primary voltage monitor reset configuration register	9100_H	Pcs	SV, SE, Pcs	See 6223	6223
PMS_VMONP_VDDPPUSTAT	VDDPPU primary voltage monitor status register	9104_H	P	BE	See 6225	6225
PMS_VMONP_VDDPHY0CON	VDDPHY0 primary voltage monitor configuration register	9114_H	P	SV, E, P	See 6226	6226
PMS_VMONP_VDDPHY0RST	VDDPHY0 primary voltage monitor reset configuration register	9118_H	Pcs	SV, SE, Pcs	See 6227	6227

(table continues...)

Table 1500 (continued) Registers overview - PMS (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PMS_VMONP_VDDPHY0STAT	VDDPHY0 primary voltage monitor status register	$911C_H$	P	BE	See 6229	6229
PMS_VMONP_VDDPHY1CON	VDDPHY1 primary voltage monitor configuration register	9120_H	P	SV, E, P	See 6230	6230
PMS_VMONP_VDDPHY1RST	VDDPHY1 primary voltage monitor reset configuration register	9124_H	Pcs	SV, SE, Pcs	See 6231	6231
PMS_VMONP_VDDPHY1STAT	VDDPHY1 primary voltage monitor status register	9128_H	P	BE	See 6233	6233
PMS_VMONP_VDDPHY2CON	VDDPHY2 primary voltage monitor configuration register	$912C_H$	P	SV, E, P	See 6234	6234
PMS_VMONP_VDDPHY2RST	VDDPHY2 primary voltage monitor reset configuration register	9130_H	Pcs	SV, SE, Pcs	See 6235	6235
PMS_VMONP_VDDPHY2STAT	VDDPHY2 primary voltage monitor status register	9134_H	P	BE	See 6236	6236
PMS_VMONP_TRKTRIM0	Tracking ADC trim register 0	9138_H	P	SV, E, P	See 6237	6237
PMS_VMONS_VDDEVRSBCON	VDDEVRSB secondary monitor configuration register	9144_H	P	SV, E, P	See 6239	6239
PMS_VMONS_VDDEVRSBSTAT	VDDEVRSB secondary monitor status register	9148_H	P	BE	See 6241	6241
PMS_VMONS_VDDEXTCON	VDDEXT secondary monitor configuration register	$914C_H$	P	SV, E, P	See 6241	6241
PMS_VMONS_VDDEXTSTAT	VDDEXT secondary monitor status register	9150_H	P	BE	See 6243	6243
PMS_VMONS_VDDMCON	VDDM secondary monitor configuration register	9154_H	P	SV, E, P	See 6243	6243
PMS_VMONS_VDDMSTAT	VDDM secondary monitor status register	9158_H	P	BE	See 6245	6245
PMS_VMONS_VDDCON	VDD secondary monitor configuration register	9164_H	P	SV, E, P	See 6245	6245
PMS_VMONS_VDDSTAT	VDD secondary monitor status register	9168_H	P	BE	See 6247	6247
PMS_VMONS_VDDPMS0CON	VDDPMS0 secondary monitor configuration register	$916C_H$	P	SV, E, P	See 6247	6247
PMS_VMONS_VDDPMS0STAT	VDDPMS0 secondary monitor status register	9170_H	P	BE	See 6249	6249

(table continues...)

Table 1500 (continued) Registers overview - PMS (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PMS_VMONS_VDDPMS1CON	VDDPMS1 secondary monitor configuration register	9174_H	P	SV, E, P	See 6249	6249
PMS_VMONS_VDDPMS1STAT	VDDPMS1 secondary monitor status register	9178_H	P	BE	See 6251	6251
PMS_VMONS_VDDPMS2CON	VDDPMS2 secondary monitor configuration register	$917C_H$	P	SV, E, P	See 6251	6251
PMS_VMONS_VDDPMS2STAT	VDDPMS2 secondary monitor status register	9180_H	P	BE	See 6253	6253
PMS_VMONS_VDDSBRAMCON	VDDSBRAM secondary monitor configuration register	9184_H	P	SV, E, P	See 6253	6253
PMS_VMONS_VDDSBRAMSTAT	VDDSBRAM secondary monitor status register	9188_H	P	BE	See 6255	6255
PMS_VMONS_VDDPADCON	VDDPAD secondary monitor configuration register	$918C_H$	P	SV, E, P	See 6255	6255
PMS_VMONS_VDDPADSTAT	VDDPAD secondary monitor status register	9190_H	P	BE	See 6257	6257
PMS_VMONS_VDDPHPHY0CON	VDDPHPHY0 secondary monitor configuration register	$919C_H$	P	SV, E, P	See 6257	6257
PMS_VMONS_VDDPHPHY0STAT	VDDPHPHY0 secondary monitor status register	$91A0_H$	P	BE	See 6259	6259
PMS_VMONS_VDDPHPHY1CON	VDDPHPHY1 secondary monitor configuration register	$91A4_H$	P	SV, E, P	See 6259	6259
PMS_VMONS_VDDPHPHY1STAT	VDDPHPHY1 secondary monitor status register	$91A8_H$	P	BE	See 6261	6261
PMS_VMONS_VDDPHPHY2CON	VDDPHPHY2 secondary monitor configuration register	$91AC_H$	P	SV, E, P	See 6261	6261
PMS_VMONS_VDDPHPHY2STAT	VDDPHPHY2 secondary monitor status register	$91B0_H$	P	BE	See 6263	6263
PMS_VMONS_VDDHSIFCON	VDDHSIF secondary monitor configuration register	$91B4_H$	P	SV, E, P	See 6263	6263
PMS_VMONS_VDDHSIFSTAT	VDDHSIF secondary monitor status register	$91B8_H$	P	BE	See 6265	6265
PMS_VMONS_VDDPHY0CON	VDDPHY0 secondary monitor configuration register	$91C4_H$	P	SV, E, P	See 6265	6265

(table continues...)

Table 1500 (continued) Registers overview - PMS (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PMS_VMONS_VDDPHY0STAT	VDDPHY0 secondary monitor status register	$91C8_H$	P	BE	See 6267	6267
PMS_VMONS_VDDPHY1CON	VDDPHY1 secondary monitor configuration register	$91CC_H$	P	SV, E, P	See 6267	6267
PMS_VMONS_VDDPHY1STAT	VDDPHY1 secondary monitor status register	$91D0_H$	P	BE	See 6269	6269
PMS_VMONS_VDDPHY2CON	VDDPHY2 secondary monitor configuration register	$91D4_H$	P	SV, E, P	See 6269	6269
PMS_VMONS_VDDPHY2STAT	VDDPHY2 secondary monitor status register	$91D8_H$	P	BE	See 6271	6271
PMS_VMONS_VDDLMU0CON	VDDLMU0 secondary monitor configuration register	$91DC_H$	P	SV, E, P	See 6271	6271
PMS_VMONS_VDDLMU0STAT	VDDLMU0 secondary monitor status register	$91E0_H$	P	BE	See 6273	6273
PMS_VMONS_VDDLMU1CON	VDDLMU1 secondary monitor configuration register	$91E4_H$	P	SV, E, P	See 6273	6273
PMS_VMONS_VDDLMU1STAT	VDDLMU1 secondary monitor status register	$91E8_H$	P	BE	See 6275	6275
PMS_VMONS_VDDPPUCON	VDDPPU secondary monitor configuration register	$91EC_H$	P	SV, E, P	See 6275	6275
PMS_VMONS_VDDPPUSTAT	VDDPPU secondary monitor status register	$91F0_H$	P	BE	See 6277	6277
PMS_VMONS_VDDEXTDCCON	VDDEXTDC secondary monitor configuration register	$91F4_H$	P	SV, E, P	See 6277	6277
PMS_VMONS_VDDEXTDCSTAT	VDDEXTDC secondary monitor status register	$91F8_H$	P	BE	See 6279	6279
PMS_VMONS_LVLSEL	Secondary monitor level selection register	$91FC_H$	Pcs	SV, SE, Pcs, STP	See 6280	6280
PMS_VMONS_LVLSTAT	Secondary monitor level selection status register	9200_H	P	BE	See 6281	6281
PMS_PEAK_VDD	VDD peak voltage detection configuration register	9204_H	P	P	LVDPMS2 Reset	6060
PMS_PEAK_VDDEXT	VDDEXT peak voltage detection configuration register	9208_H	P	P	LVDPMS2 Reset	6060

(table continues...)

Table 1500 (continued) Registers overview - PMS (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PMS_PEAK_VDDEXTDC	VDDEXTDC peak voltage detection configuration register	$920C_H$	P	P	LVDPMS2 Reset	6061
PMS_PEAK_VDDSTAT	VDD peak voltage detection status register	9210_H	P	BE	LVDPMS2 Reset	6062
PMS_PEAK_VDDEXTSTAT	VDDEXT peak voltage detection status register	9214_H	P	BE	LVDPMS2 Reset	6063
PMS_PEAK_VDDEXTDCSTAT	VDDEXTDC peak voltage detection status register	9218_H	P	BE	LVDPMS2 Reset	6063
PMS_PEAK_VDDHISTSTAT	VDD peak voltage detection history status register	$921C_H$	P	BE	LVDPMS1 Reset	6064
PMS_PEAK_VDDEXTHISTSTAT	VDDEXT peak voltage detection history status register	9220_H	P	BE	LVDPMS1 Reset	6064
PMS_PEAK_VDDEXTDCHISTSTAT	VDDEXTDC peak voltage detection history status register	9224_H	P	BE	LVDPMS1 Reset	6065
PMS_DRMON_VDD	VDD drift monitoring configuration register	9228_H	P	P	LVDPMS2 Reset	6066
PMS_DRMON_VDDSTAT	VDD drift monitor status register	$922C_H$	P	BE	LVDPMS2 Reset	6066
PMS_DRMON_VDDEXT	VDDEXT drift monitoring configuration register	9230_H	P	P	LVDPMS2 Reset	6067
PMS_DRMON_VDDEXTSTAT	VDDEXT drift monitor status register	9234_H	P	BE	LVDPMS2 Reset	6067
PMS_DRMON_VDDEXTDC	VDDEXTDC drift monitoring configuration register	9238_H	P	P	LVDPMS2 Reset	6068
PMS_DRMON_VDDEXTDCSTAT	VDDEXTDC drift monitor status register	$923C_H$	P	BE	LVDPMS2 Reset	6068
PMS_DRMON_VDDM	VDDM drift monitoring configuration register	9240_H	P	P	LVDPMS2 Reset	6069
PMS_DRMON_VDDMSTAT	VDDM drift monitor status register	9244_H	P	BE	LVDPMS2 Reset	6069
PMS_DRMON_VDDPMS0	VDDPMS0 drift monitoring configuration register	9248_H	P	P	LVDPMS2 Reset	6070
PMS_DRMON_VDDPMS0STAT	VDDPMS0 drift monitor status register	$924C_H$	P	BE	LVDPMS2 Reset	6070
PMS_DRMON_VDDPMS2	VDDPMS2 drift monitoring configuration register	9250_H	P	P	LVDPMS2 Reset	6071
PMS_DRMON_VDDPMS2STAT	VDDPMS2 drift monitor status register	9254_H	P	BE	LVDPMS2 Reset	6071

(table continues...)

Table 1500 (continued) Registers overview - PMS (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PMS_VID_CON0	Voltage ID (VID) configuration register 0	9258_H	U	STP|TM	6072	6072
PMS_PD_CON0	Power domain configuration register 0	9260_H	P	SV, E, P	See 6281	6281
PMS_PD_CON1	Power domain configuration register 1	9264_H	P	SV, E, P	See 6283	6283
PMS_PD_STAT0	Power domain status register 0	9268_H	P	BE	See 6285	6285
PMS_INT_CON0	Interrupt configuration register 0	$926C_H$	P	SV, E, P	See 6286	6286
PMS_INT_CON1	Interrupt Configuration Register 1	9270_H	P	SV, E, P	See 6289	6289
PMS_INT_CON2	Interrupt configuration register 2	9274_H	P	SV, E, P	6072	6072
PMS_INT_STAT	Interrupt status register	9278_H	P	SV, E, P	6075	6075
PMS_INT_CLR	Interrupt status clear register	$927C_H$	P	SV, E, P	6078	6078
PMS_PAD_CON	Pad configuration register	9280_H	P	SV, E, P	LVDANA Reset	6079
PMS_PAD_STAT	Pad status register	9284_H	P	BE	See 6292	6292
PMS_PAD_ESR2CON	ESR2 pad configuration register	9288_H	P	SV, E, P	6081	6081
PMS_HPOSCCON1	HPOSC configuration register 1	9294_H	P	SV, E, P	6082	6082
PMS_HPOSCTRIM	HPOSC trim register for dynamic trimming over temperature	$929C_H$	P	SV, E, P	6083	6083
PMS_HPBGTRIM	HPBG trim register for dynamic trimming over temperature	$92A0_H$	P	SV, E, P	6084	6084
PMS_PMS0TRIM	Trim register for PMS0 components	$92A8_H$	Pcs	SV, SE, Pcs	See 6293	6293
PMS_WAKEUP_CON0	Wake-up configuration register 0	$92AC_H$	P	SV, E, P	6085	6085
PMS_WAKEUP_STAT0	Wake-up status register 0	$92B0_H$	P	BE	6090	6090
PMS_WAKEUP_CLR0	Standby and wake-up status clear register 0	$92B4_H$	P	SV, E, P	6092	6092
PMS_WAKEUP_CON1	Wake-up configuration register 1	$92B8_H$	P	SV, E, P	6094	6094
PMS_WAKEUP_STAT1	Wake-up status register 1	$92BC_H$	P	BE	6096	6096

(table continues...)

Table 1500 (continued) Registers overview - PMS (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PMS_WAKEUP_CLR1	Standby and wake-up status clear register 1	$92C0_H$	P	SV, E, P	6098	6098
PMS_STANDBY_CON0	Standby configuration register 0	$92C4_H$	P	SV, E, P	See 6294	6294
PMS_STANDBY_CON1	Standby configuration register 1	$92C8_H$	P	SV, E, P	LVDPMS1 Reset	6100
PMS_WUT_CON	Wake-up timer configuration register	$92CC_H$	P	SV, E, P	6101	6101
PMS_WUT_STAT	Wake-up timer status register	$92D0_H$	P	BE	6102	6102
PMS_SCR_CON0	SCR configuration register 0	$92D4_H$	P	SV, E, P	6103	6103
PMS_SCR_CON1	SCR configuration register 1	$92D8_H$	P	SV, E, P, M	LVDPMS1 Reset	6105
PMS_SCR_STAT	SCR status register	$92DC_H$	P	SV, E, P, M	LVDPMS1 Reset	6106
PMS_EVRC_STAT0	EVRC status register 0	$92E0_H$	P	BE	LVDPMS2 Reset	6296
PMS_EVRC_STAT1	EVRC status register 1	$92E4_H$	P	BE	LVDPMS2 Reset	6108
PMS_EVRC_CON0	EVRC configuration register 0	$92EC_H$	P	SV, E, P	See 6298	6298
PMS_EVRC_CON1	EVRC configuration register 1	$92F0_H$	P	SV, E, P	6108	6108
PMS_EVRC_CON2	EVRC configuration register 2	$92F4_H$	P	SV, E, P	6111	6111
PMS_EVRC_CON3	EVRC configuration register 3	$92F8_H$	P	SV, E, P	6112	6112
PMS_EVRC_CON4	EVRC configuration register 4	$92FC_H$	P	SV, E, P	6114	6114
PMS_EVRC_COEFF0	EVRC SD coefficient register 0	9300_H	P	SV, E, P	6115	6115
PMS_EVRC_COEFF1	EVRC SD coefficient register 1	9304_H	P	SV, E, P	6117	6117
PMS_EVRC_COEFF2	EVRC SD coefficient register 2	9308_H	P	SV, E, P	6120	6120
PMS_EVRC_COEFF3	EVRC SD coefficient register 3	$930C_H$	P	SV, E, P	6122	6122
PMS_EVRC_COEFF4	EVRC SD coefficient register 4	9310_H	P	SV, E, P	6125	6125
PMS_EVRC_COEFF5	EVRC SD coefficient register 5	9314_H	P	SV, E, P	6127	6127
PMS_RTC_CON0	RTC configuration register 0	9318_H	P	SV, E, P	See 6300	6300
PMS_RTC_CON1	RTC configuration register 1	$931C_H$	P	SV, E, P	6128	6128

(table continues...)

Table 1500 (continued) Registers overview - PMS (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PMS_RTC_CON2	RTC configuration register 2	9320_H	P	SV, E, P	6129	6129
PMS_RTC_CMP0	RTC compare register 0	9324_H	P	SV, E, P	6130	6130
PMS_RTC_CMP1	RTC compare register 1	9328_H	P	SV, E, P	6131	6131
PMS_RTC_TIM0	RTC timer counter register 0	$932C_H$	P	BE	6131	6131
PMS_RTC_TIM1CAP	RTC timer capture register	9330_H	P	BE	LVDPMS2 Reset	6132
PMS_DTS_CON	Die temperature sensor control register	9334_H	U	E, P, SV	6132	6132
PMS_DTS_LIM1	Die temperature sensor limit register 1	9338_H	U	E, P, SV	LVDPMS1 Reset	6134
PMS_DTS_LIM2	Die temperature sensor limit register 2	$933C_H$	U	E, P, SV	LVDPMS1 Reset	6134
PMS_DTS_STAT	Die temperature sensor status register	9340_H	P	BE	LVDPMS1 Reset	6135
PMS_CLDRSTCON	Cold reset configuration register	9350_H	P	SV, E, P	LVDPMS1 Reset	6136
PMS_CLDRSTSTAT	Cold reset status register	9354_H	P	BE	See 6301	6301
PMS_CLDRSTRIG	Cold reset trigger register	9358_H	P	BE	LVDPMS2 Reset	6303
PMS_CLDRSTRIGL	Cold reset last trigger register	$935C_H$	P	BE	LVDPMS2 Reset	6306
PMS_CLDRSTSTATCLR	Cold reset status clear register	9360_H	P	SV, E, P	LVDPMS2 Reset	6137
PMS_OTSS	OCDS trigger set select register	9364_H	P	P	6137	6137
PMS_OTSC0	OCDS trigger set control 0 register	9368_H	P	P	See 6310	6310
PMS_OTSC1	OCDS trigger set control 1 register	$936C_H$	P	P	6138	6138
PMS_LVDPMS1USRINFO	Register to store user info in PMS1 domain and reset by LVDPMS1	9370_H	P	P	6139	6139
PMS_CLDPORSTUSRINFO	Register to store user info in PMS2 domain and reset by cold PORST	9374_H	P	P	6140	6140
PMS_WRMPORSTUSRINFO	Register to store user info in PMS2 domain and reset by warm PORST	9378_H	P	P	6140	6140

36.8.4.5 VDD primary voltage monitor configuration register

PMS_VMONP_VDDCON Offset address: 9044_H

VDD primary voltage monitor configuration register Reset values see: Table 1501

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0			OVENABLE	OVVAL											FILTER
r			rw	rw											rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FILTER				UVENABLE	UVVAL										
rw				rw	rw										

Field	Bits	Type	Description
UVVAL	10:0	rw	**Primary monitor alarm under-voltage threshold** This field defines the under-voltage monitoring threshold level. The LSB is documented for each supply rail in the user manual (primary voltage monitors section).
UVENABLE	11	rw	**Primary monitor under-voltage alarm enable** 0_B No alarm trigger signal is generated. 1_B An alarm trigger signal is generated and forwarded depending on the configured threshold value.
FILTER	16:12	rw	**Primary monitor alarm spike filter** 00_H Each conversion result is compared with threshold to generate alarm $1F_H$ A spike filter of consecutive 32 ADC results are used to generate alarm.
OVVAL	27:17	rw	**Primary monitor alarm over-voltage threshold** This field defines the over-voltage monitoring threshold level. The LSB is documented for each supply rail in the user manual (primary voltage monitors section).
OVENABLE	28	rw	**Primary monitor over-voltage alarm enable** 0_B No alarm trigger signal is generated. 1_B An alarm trigger signal is generated and forwarded depending on the configured threshold value.
0	31:29	r	**Reserved**

Table 1501 Reset values of PMS_VMONP_VDDCON

Reset type	Reset value	Note
LVDPMS2 Reset	$1351\ F927_H$	
Cold PORST	$1351\ F927_H$	
After Boot-FW Value	$032F\ F137_H$	

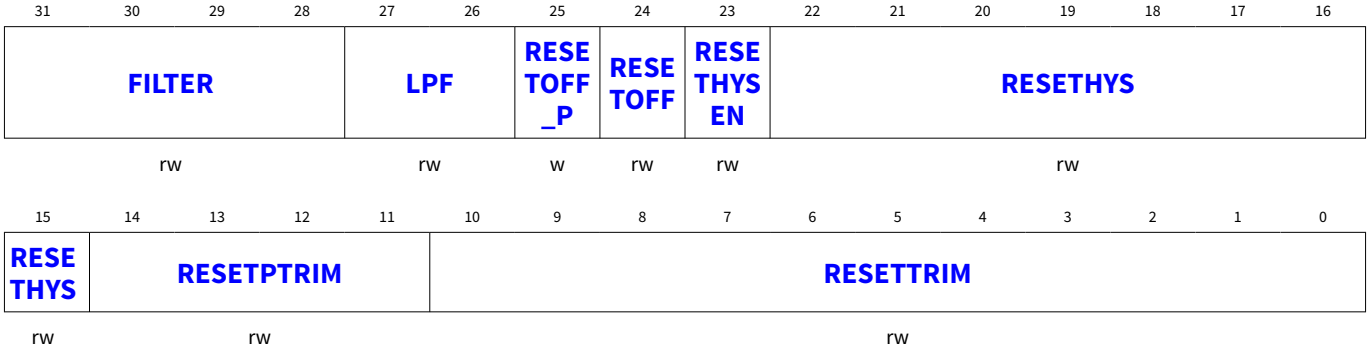
36.8.4.6 VDD primary voltage monitor reset configuration register

PMS_VMONP_VDDRST

VDD primary voltage monitor reset configuration register

Offset address: 9048_H

Reset values see: Table 1503



Field	Bits	Type	Description
RESETTRIM	10:0	rw	**Supply reset trim value** This bit field selects the hard reset generation level of the respective supply rail. This bitfield is trimmed by Firmware. The LSB is documented for each supply rail in the user manual (primary voltage monitors section).
RESETPTRIM	14:11	rw	**Signed reset trim value** This bit field allows fine trimming of the reset threshold. Binary signed two's complement representation.
RESETHYS	22:15	rw	**Supply reset hysteresis trim value** This bit field selects the reset hysteresis offset level of the primary voltage monitor. The LSB is documented for each supply rail in the user manual (primary voltage monitors section).
RESETHYSEN	23	rw	**Reset release hysteresis enable** 0_B Hysteresis for reset release is not active. 1_B Hysteresis for reset release is active.
RESETOFF	24	rw	**Reset enable** This bit can only be changed if bit RESETOFF_P is set in parallel. RESETOFF is intended to be used only for internal test purposes and the primary reset generation is not to be disabled in customer application. 0_B A reset trigger signal is generated and forwarded to the SCU by the primary monitor depending on the selected reset trim value. 1_B No reset trigger signal is generated and forwarded to the SCU by the primary monitor.
RESETOFF_P	25	w	**RESETOFF bit protection** Writing 1 into this bit enables that bit RESETOFF can be changed in this write operation. This bit is read as zero.

(table continues...)

(continued)

Field	Bits	Type	Description
LPF	27:26	rw	**Primary monitor reset LPF coefficient** This bitfield configures the coefficient of the low pass filter of the primary monitor tracking ADC counter value. The low pass filter is used for cold PORST generation and VMONP_VxxSTAT.RESULT value. y [k] = { y [k-1] * (1-a) } + { x [k] * a} ; y [k] is filter output; x [k] is ADC output a = {1 / (2 ^ LPF)}.If LPF = 0, the filter output is the same as ADC output.
FILTER	31:28	rw	**Primary monitor reset spike filter** 0_H Each conversion result is compared with threshold to generate alarm F_H A spike filter of consecutive 16 ADC results are used to generate reset.

Table 1502 Access mode restrictions of PMS_VMONP_VDDRST sorted by descending priority

Mode name	Access mode		Description
write 1 to .**RESETOFF_P**	rw	RESETOFF	
(default)	r	RESETOFF	

Table 1503 Reset values of PMS_VMONP_VDDRST

Reset type	Reset value	Note
LVDPMS2 Reset	0C83 0124$_H$	
Cold PORST	0C83 0124$_H$	
After Boot-FW Value	0000 1100 1000 0011 0––– –001 0011 0011$_B$	

36.8.4.7 Power domain cold reset configuration register

PMS_VMONP_VDDCLDRST Offset address: 904C$_H$

Power domain cold reset configuration register Reset values see: Table 1505

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0														FILTER	
r														rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FILTER			VDDPDCLDOFF_P	VDDPDCLDOFF	VDDPDCLDUV										
rw			w	rw	rw										

Field	Bits	Type	Description
VDDPDCLDUV	10:0	rw	**Hard PD reset generation level** Hard PD reset generation level, trimmed by FW. Additional comparator threshold for power domain switch and control logic reset. Voltage level = VDDPDCLDUV * LSB LSB = $V_{TRKCORELSB}$ data sheet parameter
VDDPDCLDOFF	11	rw	**Reset disable bit for test and debug purpose**
VDDPDCLDOFF_P	12	w	**Bit protection for VDDPDCLDOFF** Writing 1 into this bit enables that the protected bit can be changed in this write operation. This bit is read as zero.
FILTER	17:13	rw	**Primary monitor reset spike filter** 00_H Each conversion result is compared with threshold to generate alarm $1F_H$ A spike filter of consecutive 32 ADC results are used to generate reset.
0	31:18	r	**Reserved** Read as 0; should be written with 0.

Table 1504 Access mode restrictions of PMS_VMONP_VDDCLDRST sorted by descending priority

Mode name	Access mode		Description
write 1 to .**VDDPDCLDOFF_P**	rw	VDDPDCLDOFF	
(default)	r	VDDPDCLDOFF	

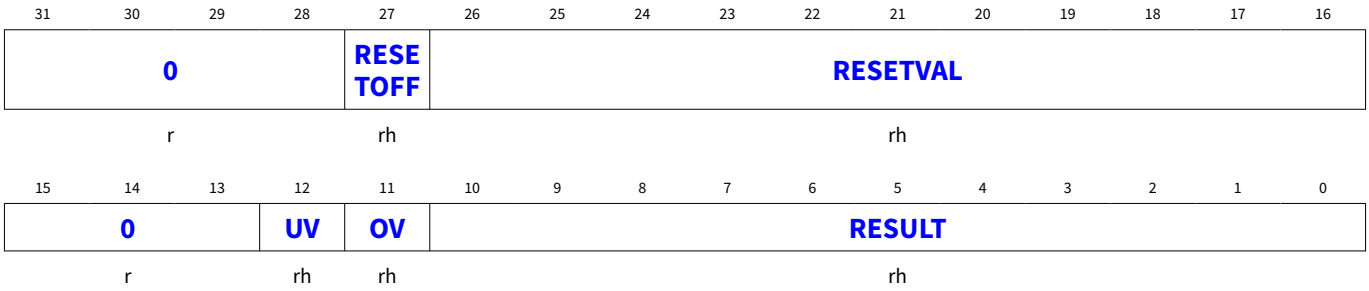
Table 1505 Reset values of PMS_VMONP_VDDCLDRST

Reset type	Reset value	Note
LVDPMS2 Reset	$0000\ 0100_H$	
Power Domain Cold Reset	$0000\ 0100_H$	
After Boot-FW Value	$0000\ 0110_H$	

36.8.4.8 VDD primary voltage monitor status register

PMS_VMONP_VDDSTAT Offset address: 9050_H

VDD primary voltage monitor status register Reset values see: Table 1506



Field	Bits	Type	Description
RESULT	10:0	rh	**Primary monitor ADC conversion result** This bit field contains the last conversion result of the ADC measurement of the primary voltage monitor. The LSB is documented for each supply rail in the user manual (primary voltage monitors section).
OV	11	rh	**Primary monitor over-voltage event flag** This bit is set if the primary voltage monitor recognizes an over-voltage event. An alarm is raised to the CSRM and SMU. 0_B No over-voltage condition happened. 1_B Over-voltage event indication as configured in the respective VMONP_VxxCON register.
UV	12	rh	**Primary monitor under-voltage event flag** This bit is set if the primary voltage monitor recognizes an under-voltage event. An alarm is raised to the CSRM and SMU. 0_B No under-voltage condition happened. 1_B Under-voltage event indication as configured in the respective VMONP_VxxCON register.
RESETVAL	26:16	rh	**Supply reset value status** This bit field indicates the actual cold PORST reset trim setpoint used by the primary voltage monitors.
RESETOFF	27	rh	**Reset enable status** 0_B A cold PORST is triggered in case of primary under-voltage event 1_B No cold PORST is generated in case of a primary under-voltage event.
0	15:13, 31:28	r	**Reserved**

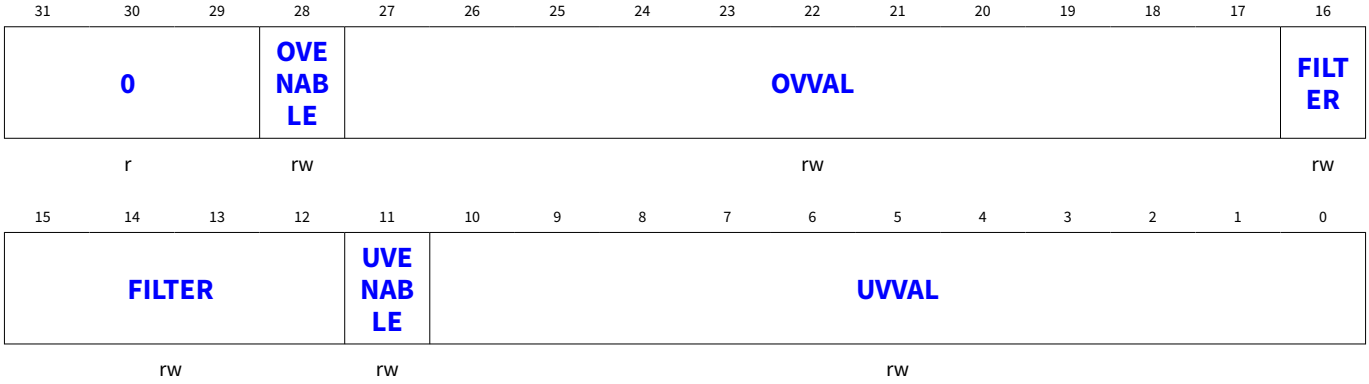
Table 1506 Reset values of PMS_VMONP_VDDSTAT

Reset type	Reset value	Note
LVDPMS2 Reset	$0124\ 1800_H$	
Cold PORST	$0124\ 1800_H$	

36.8.4.9 VDDEXT primary voltage monitor configuration register

PMS_VMONP_VDDEXTCON Offset address: 9054_H

VDDEXT primary voltage monitor configuration register Reset values see: Table 1507



Field	Bits	Type	Description
UVVAL	10:0	rw	**Primary monitor alarm under-voltage threshold** This field defines the under-voltage monitoring threshold level. The LSB is documented for each supply rail in the user manual (primary voltage monitors section).
UVENABLE	11	rw	**Primary monitor under-voltage alarm enable** 0_B No alarm trigger signal is generated. 1_B An alarm trigger signal is generated and forwarded depending on the configured threshold value.
FILTER	16:12	rw	**Primary monitor alarm spike filter** 00_H Each conversion result is compared with threshold to generate alarm $1F_H$ A spike filter of consecutive 32 ADC results are used to generate alarm.
OVVAL	27:17	rw	**Primary monitor alarm over-voltage threshold** This field defines the over-voltage monitoring threshold level. The LSB is documented for each supply rail in the user manual (primary voltage monitors section).
OVENABLE	28	rw	**Primary monitor over-voltage alarm enable** 0_B No alarm trigger signal is generated. 1_B An alarm trigger signal is generated and forwarded depending on the configured threshold value.
0	31:29	r	**Reserved**

Table 1507 Reset values of PMS_VMONP_VDDEXTCON

Reset type	Reset value	Note
LVDPMS2 Reset	13F5 F8F7$_H$	
Cold PORST	13F5 F8F7$_H$	
After Boot-FW Value	03E1 F101$_H$	

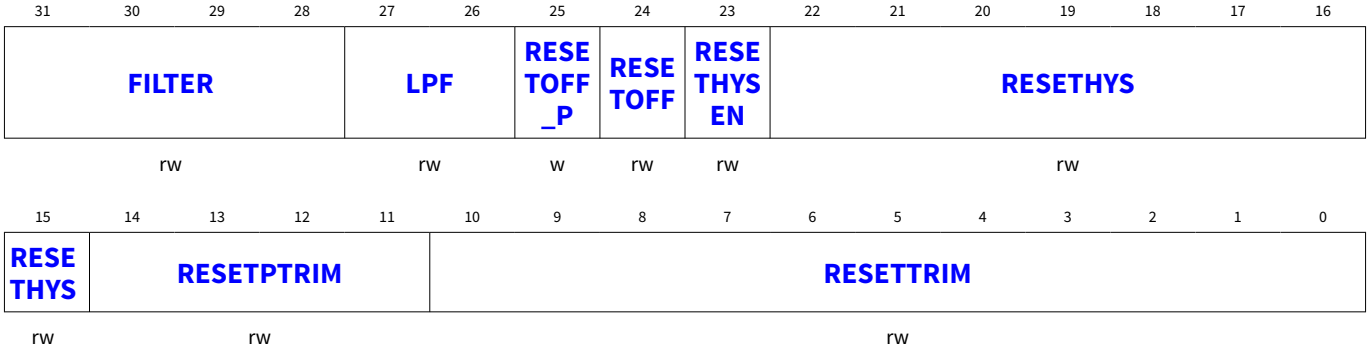
36.8.4.10 VDDEXT primary voltage monitor reset configuration register

PMS_VMONP_VDDEXTRST

VDDEXT primary voltage monitor reset configuration register

Offset address: 9058_H

Reset values see: Table 1509



Field	Bits	Type	Description
RESETTRIM	10:0	rw	**Supply reset trim value** This bit field selects the hard reset generation level of the respective supply rail. This bitfield is trimmed by Firmware. The LSB is documented for each supply rail in the user manual (primary voltage monitors section).
RESETPTRIM	14:11	rw	**Signed reset trim value** This bit field allows fine trimming of the reset threshold. Binary signed two's complement representation.
RESETHYS	22:15	rw	**Supply reset hysteresis trim value** This bit field selects the reset hysteresis offset level of the primary voltage monitor. The LSB is documented for each supply rail in the user manual (primary voltage monitors section).
RESETHYSEN	23	rw	**Reset release hysteresis enable** 0_B Hysteresis for reset release is not active. 1_B Hysteresis for reset release is active.
RESETOFF	24	rw	**Reset enable** This bit can only be changed if bit RESETOFF_P is set in parallel. RESETOFF is intended to be used only for internal test purposes and the primary reset generation is not to be disabled in customer application. 0_B A reset trigger signal is generated and forwarded to the SCU by the primary monitor depending on the selected reset trim value. 1_B No reset trigger signal is generated and forwarded to the SCU by the primary monitor.
RESETOFF_P	25	w	**RESETOFF bit protection** Writing 1 into this bit enables that bit RESETOFF can be changed in this write operation. This bit is read as zero.

(table continues...)

(continued)

Field	Bits	Type	Description
LPF	27:26	rw	**Primary monitor reset LPF coefficient** This bitfield configures the coefficient of the low pass filter of the primary monitor tracking ADC counter value. The low pass filter is used for cold PORST generation and VMONP_VxxSTAT.RESULT value. y [k] = { y [k-1] * (1-a) } + { x [k] * a} ; y [k] is filter output; x [k] is ADC output a = {1 / (2 ^ LPF)}.If LPF = 0, the filter output is the same as ADC output.
FILTER	31:28	rw	**Primary monitor reset spike filter** 0_H Each conversion result is compared with threshold to generate alarm F_H A spike filter of consecutive 16 ADC results are used to generate reset.

Table 1508 Access mode restrictions of PMS_VMONP_VDDEXTRST sorted by descending priority

Mode name	Access mode		Description
write 1 to .**RESETOFF_P**	rw	RESETOFF	
(default)	r	RESETOFF	

Table 1509 Reset values of PMS_VMONP_VDDEXTRST

Reset type	Reset value	Note
LVDPMS2 Reset	FC82 80F5$_H$	
Cold PORST	FC82 80F5$_H$	
After Boot-FW Value	1111 1100 1000 0010 1––– –000 1111 1110$_B$	

36.8.4.11 VDDEXT primary voltage monitor status register

PMS_VMONP_VDDEXTSTAT Offset address: 905C$_H$

VDDEXT primary voltage monitor status register Reset values see: Table 1510

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0				RESETOFF	RESETVAL										
r				rh	rh										

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			UV	OV	RESULT										
r			rh	rh	rh										

Field	Bits	Type	Description
RESULT	10:0	rh	**Primary monitor ADC conversion result** This bit field contains the last conversion result of the ADC measurement of the primary voltage monitor. The LSB is documented for each supply rail in the user manual (primary voltage monitors section).
OV	11	rh	**Primary monitor over-voltage event flag** This bit is set if the primary voltage monitor recognizes an over-voltage event. An alarm is raised to the CSRM and SMU. 0_B No over-voltage condition happened. 1_B Over-voltage event indication as configured in the respective VMONP_VxxCON register.
UV	12	rh	**Primary monitor under-voltage event flag** This bit is set if the primary voltage monitor recognizes an under-voltage event. An alarm is raised to the CSRM and SMU. 0_B No under-voltage condition happened. 1_B Under-voltage event indication as configured in the respective VMONP_VxxCON register.
RESETVAL	26:16	rh	**Supply reset value status** This bit field indicates the actual cold PORST reset trim setpoint used by the primary voltage monitors.
RESETOFF	27	rh	**Reset enable status** 0_B A cold PORST is triggered in case of primary under-voltage event 1_B No cold PORST is generated in case of a primary under-voltage event.
0	15:13, 31:28	r	**Reserved**

Table 1510 Reset values of PMS_VMONP_VDDEXTSTAT

Reset type	Reset value	Note
LVDPMS2 Reset	00F5 1800$_H$	
Cold PORST	00F5 1800$_H$	

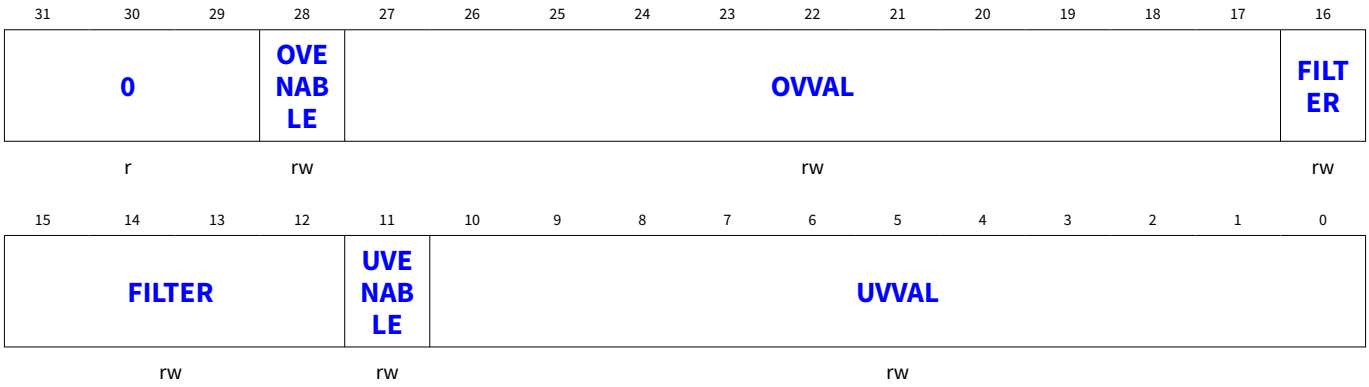
36.8.4.12 Primary voltage monitor configuration register for the (VDDEXTDC - VSSDCHS) voltage difference

PMS_VMONP_VSSDCHSCON

Primary voltage monitor configuration register for the [VDDEXTDC - VSSDCHS) voltage difference

Offset address: 9060$_H$

Reset values see: Table 1511



Field	Bits	Type	Description
UVVAL	10:0	rw	**Primary monitor alarm under-voltage threshold** This field defines the under-voltage monitoring threshold level. The LSB is documented for each supply rail in the user manual (primary voltage monitors section).
UVENABLE	11	rw	**Primary monitor under-voltage alarm enable** 0_B No alarm trigger signal is generated. 1_B An alarm trigger signal is generated and forwarded depending on the configured threshold value.
FILTER	16:12	rw	**Primary monitor alarm spike filter** 00_H Each conversion result is compared with threshold to generate alarm $1F_H$ A spike filter of consecutive 32 ADC results are used to generate alarm.
OVVAL	27:17	rw	**Primary monitor alarm over-voltage threshold** This field defines the over-voltage monitoring threshold level. The LSB is documented for each supply rail in the user manual (primary voltage monitors section).
OVENABLE	28	rw	**Primary monitor over-voltage alarm enable** 0_B No alarm trigger signal is generated. 1_B An alarm trigger signal is generated and forwarded depending on the configured threshold value.
0	31:29	r	**Reserved**

Table 1511 Reset values of PMS_VMONP_VSSDCHSCON

Reset type	Reset value	Note
LVDPMS2 Reset	13F4 F8F8$_H$	
Cold PORST	13F4 F8F8$_H$	
After Boot-FW Value	03E0 F101$_H$	

36.8.4.13 Primary voltage monitor reset configuration register for the (VDDEXTDC - VSSDCHS) voltage difference

PMS_VMONP_VSSDCHSRST

Offset address: 9064_H

Primary voltage monitor reset configuration register for the [VDDEXTDC - VSSDCHS) voltage difference

Reset values see: Table 1513

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FILTER				LPF		RESETOFF_P	RESETOFF	RESETHYSEN	RESETHYS						
rw				rw		w	rw	rw	rw						

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESETHYS	RESETPTRIM							RESETTRIM							
rw	rw							rw							

Field	Bits	Type	Description
RESETTRIM	10:0	rw	**Supply reset trim value** This bit field selects the hard reset generation level of the respective supply rail. This bitfield is trimmed by Firmware. The LSB is documented for each supply rail in the user manual (primary voltage monitors section).
RESETPTRIM	14:11	rw	**Signed reset trim value** This bit field allows fine trimming of the reset threshold. Binary signed two's complement representation.
RESETHYS	22:15	rw	**Supply reset hysteresis trim value** This bit field selects the reset hysteresis offset level of the primary voltage monitor. The LSB is documented for each supply rail in the user manual (primary voltage monitors section).
RESETHYSEN	23	rw	**Reset release hysteresis enable** 0_B Hysteresis for reset release is not active. 1_B Hysteresis for reset release is active.
RESETOFF	24	rw	**Reset enable** This bit can only be changed if bit RESETOFF_P is set in parallel. RESETOFF is intended to be used only for internal test purposes and the primary reset generation is not to be disabled in customer application. 0_B A reset trigger signal is generated and forwarded to the SCU by the primary monitor depending on the selected reset trim value. 1_B No reset trigger signal is generated and forwarded to the SCU by the primary monitor.
RESETOFF_P	25	w	**RESETOFF bit protection** Writing 1 into this bit enables that bit RESETOFF can be changed in this write operation. This bit is read as zero.

(table continues...)

(continued)

Field	Bits	Type	Description
LPF	27:26	rw	**Primary monitor reset LPF coefficient** This bitfield configures the coefficient of the low pass filter of the primary monitor tracking ADC counter value. The low pass filter is used for cold PORST generation and VMONP_VxxSTAT.RESULT value. y [k] = { y [k-1] * (1-a) } + { x [k] * a} ; y [k] is filter output; x [k] is ADC output a = {1 / (2 ^ LPF)}.If LPF = 0, the filter output is the same as ADC output.
FILTER	31:28	rw	**Primary monitor reset spike filter** 0_H Each conversion result is compared with threshold to generate alarm F_H A spike filter of consecutive 16 ADC results are used to generate reset.

Table 1512 Access mode restrictions of PMS_VMONP_VSSDCHSRST sorted by descending priority

Mode name	Access mode		Description
write 1 to .**RESETOFF_P**	rw	RESETOFF	
(default)	r	RESETOFF	

Table 1513 Reset values of PMS_VMONP_VSSDCHSRST

Reset type	Reset value	Note
LVDPMS2 Reset	8C83 00F5$_H$	
Cold PORST	8C83 00F5$_H$	
After Boot-FW Value	1000 1100 1000 0011 0––– –000 1111 1110$_B$	

36.8.4.14 Primary voltage monitor status register for the (VDDEXTDC - VSSDCHS) voltage difference

PMS_VMONP_VSSDCHSSTAT Offset address: 9068$_H$

Primary voltage monitor status register for the [VDDEXTDC - VSSDCHS) voltage difference Reset values see: Table 1514

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0				RESETOFF	RESETVAL										
r				rh	rh										

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			UV	OV	RESULT										
r			rh	rh	rh										

Field	Bits	Type	Description
RESULT	10:0	rh	**Primary monitor ADC conversion result** This bit field contains the last conversion result of the ADC measurement of the primary voltage monitor. The LSB is documented for each supply rail in the user manual (primary voltage monitors section).
OV	11	rh	**Primary monitor over-voltage event flag** This bit is set if the primary voltage monitor recognizes an over-voltage event. An alarm is raised to the CSRM and SMU. 0_B No over-voltage condition happened. 1_B Over-voltage event indication as configured in the respective VMONP_VxxCON register.
UV	12	rh	**Primary monitor under-voltage event flag** This bit is set if the primary voltage monitor recognizes an under-voltage event. An alarm is raised to the CSRM and SMU. 0_B No under-voltage condition happened. 1_B Under-voltage event indication as configured in the respective VMONP_VxxCON register.
RESETVAL	26:16	rh	**Supply reset value status** This bit field indicates the actual cold PORST reset trim setpoint used by the primary voltage monitors.
RESETOFF	27	rh	**Reset enable status** 0_B A cold PORST is triggered in case of primary under-voltage event 1_B No cold PORST is generated in case of a primary under-voltage event.
0	15:13, 31:28	r	**Reserved**

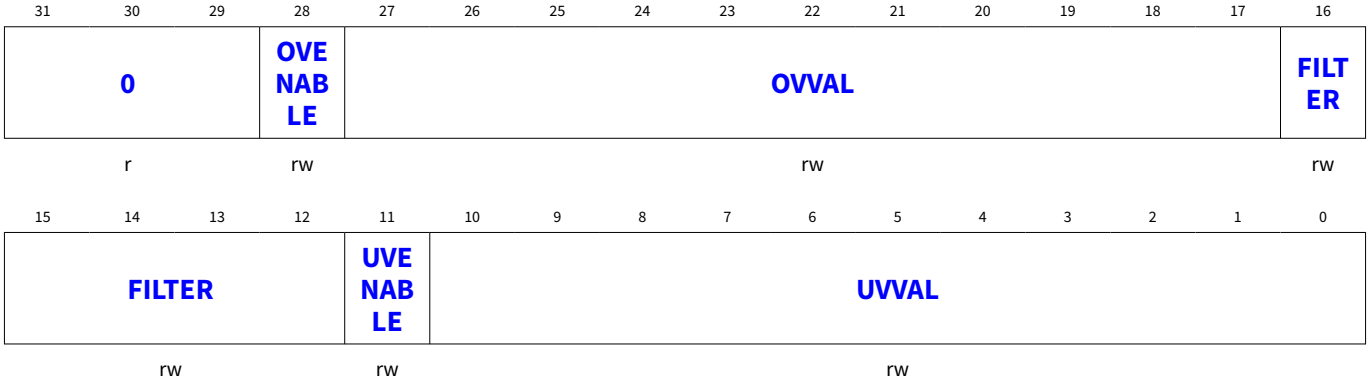
Table 1514 Reset values of PMS_VMONP_VSSDCHSSTAT

Reset type	Reset value	Note
LVDPMS2 Reset	00F5 1800$_H$	
Cold PORST	00F5 1800$_H$	

36.8.4.15 VDDEXTDC primary voltage monitor configuration register

PMS_VMONP_VDDEXTDCCON Offset address: 906C$_H$

VDDEXTDC primary voltage monitor configuration register Reset values see: Table 1515



Field	Bits	Type	Description
UVVAL	10:0	rw	**Primary monitor alarm under-voltage threshold** This field defines the under-voltage monitoring threshold level. The LSB is documented for each supply rail in the user manual (primary voltage monitors section).
UVENABLE	11	rw	**Primary monitor under-voltage alarm enable** 0_B No alarm trigger signal is generated. 1_B An alarm trigger signal is generated and forwarded depending on the configured threshold value.
FILTER	16:12	rw	**Primary monitor alarm spike filter** 00_H Each conversion result is compared with threshold to generate alarm $1F_H$ A spike filter of consecutive 32 ADC results are used to generate alarm.
OVVAL	27:17	rw	**Primary monitor alarm over-voltage threshold** This field defines the over-voltage monitoring threshold level. The LSB is documented for each supply rail in the user manual (primary voltage monitors section).
OVENABLE	28	rw	**Primary monitor over-voltage alarm enable** 0_B No alarm trigger signal is generated. 1_B An alarm trigger signal is generated and forwarded depending on the configured threshold value.
0	31:29	r	**Reserved**

Table 1515 Reset values of PMS_VMONP_VDDEXTDCCON

Reset type	Reset value	Note
LVDPMS2 Reset	13F4 F8F8$_H$	
LVDPMS2 Reset for VDDEXTDC high-voltage configuration (5.5V<=VDDEXTDC(Typ)<=6V5)	1508 F99C$_H$	Reset value with HWCFG[2:1]=11, 5V5<=VDDEXTDC<=6V5

(table continues...)

Table 1515 (continued) Reset values of PMS_VMONP_VDDEXTDCCON

Reset type	Reset value	Note
Cold PORST for VDDEXTDC high-voltage configuration (5.5V<=VDDEXTDC(Typ)<=6V5)	1508 F99C$_H$	Reset value with HWCFG[2:1]=11, 5V5<=VDDEXTDC<=6V5
Cold PORST	13F4 F8F8$_H$	
After Boot-FW Value	03E0 F101$_H$	
After Boot-FW Value for VDDEXTDC high-voltage configuration (5.5V<=VDDEXTDC(Typ)<=6V5)	04EE F1AD$_H$	Reset value with HWCFG[2:1]=11, 5V5<=VDDEXTDC<=6V5

36.8.4.16 VDDEXTDC primary voltage monitor reset configuration register

PMS_VMONP_VDDEXTDCRST Offset address: 9070$_H$

VDDEXTDC primary voltage monitor reset configuration register Reset values see: Table 1517

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FILTER				LPF		RESETOFF_P	RESETOFF	RESETHYSEN	RESETHYS						
rw				rw		w	rw	rw	rw						

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESETHYS	RESETPTRIM							RESETTRIM							
rw	rw							rw							

Field	Bits	Type	Description
RESETTRIM	10:0	rw	**Supply reset trim value** This bit field selects the hard reset generation level of the respective supply rail. This bitfield is trimmed by Firmware. The LSB is documented for each supply rail in the user manual (primary voltage monitors section).
RESETPTRIM	14:11	rw	**Signed reset trim value** This bit field allows fine trimming of the reset threshold. Binary signed two's complement representation.

(table continues...)

(continued)

Field	Bits	Type	Description
RESETHYS	22:15	rw	**Supply reset hysteresis trim value** This bit field selects the reset hysteresis offset level of the primary voltage monitor. The LSB is documented for each supply rail in the user manual (primary voltage monitors section).
RESETHYSEN	23	rw	**Reset release hysteresis enable** 0_B Hysteresis for reset release is not active. 1_B Hysteresis for reset release is active.
RESETOFF	24	rw	**Reset enable** This bit can only be changed if bit RESETOFF_P is set in parallel. RESETOFF is intended to be used only for internal test purposes and the primary reset generation is not to be disabled in customer application. 0_B A reset trigger signal is generated and forwarded to the SCU by the primary monitor depending on the selected reset trim value. 1_B No reset trigger signal is generated and forwarded to the SCU by the primary monitor.
RESETOFF_P	25	w	**RESETOFF bit protection** Writing 1 into this bit enables that bit RESETOFF can be changed in this write operation. This bit is read as zero.
LPF	27:26	rw	**Primary monitor reset LPF coefficient** This bitfield configures the coefficient of the low pass filter of the primary monitor tracking ADC counter value. The low pass filter is used for cold PORST generation and VMONP_VxxSTAT.RESULT value. y [k] = { y [k-1] * (1-a) } + { x [k] * a} ; y [k] is filter output; x [k] is ADC output a = {1 / (2 ^ LPF)}.If LPF = 0, the filter output is the same as ADC output.
FILTER	31:28	rw	**Primary monitor reset spike filter** 0_H Each conversion result is compared with threshold to generate alarm F_H A spike filter of consecutive 16 ADC results are used to generate reset.

Table 1516 Access mode restrictions of PMS_VMONP_VDDEXTDCRST sorted by descending priority

Mode name	Access mode		Description
write 1 to .**RESETOFF_P**	rw	RESETOFF	
(default)	r	RESETOFF	

Table 1517 Reset values of PMS_VMONP_VDDEXTDCRST

Reset type	Reset value	Note
LVDPMS2 Reset	8C83 00F5$_H$	

(table continues...)

Table 1517 (continued) Reset values of PMS_VMONP_VDDEXTDCRST

Reset type	Reset value	Note
LVDPMS2 Reset for VDDEXTDC high-voltage configuration (5.5V<=VDDEXTDC(Typ)<=6V5)	8C84 8198$_H$	Reset value with HWCFG[2:1]=11, 5V5<=VDDEXTDC<=6V5
Cold PORST for VDDEXTDC high-voltage configuration (5.5V<=VDDEXTDC(Typ)<=6V5)	8C84 8198$_H$	Reset value with HWCFG[2:1]=11, 5V5<=VDDEXTDC<=6V5
Cold PORST	8C83 00F5$_H$	
After Boot-FW Value	1000 1100 1000 0011 0––– –000 1111 1110$_B$	
After Boot-FW Value for VDDEXTDC high-voltage configuration (5.5V<=VDDEXTDC(Typ)<=6V5)	1000 1100 1000 0100 1––– –001 0011 1110$_B$	Reset value with HWCFG[2:1]=11, 5V5<=VDDEXTDC<=6V5

36.8.4.17 VDDEXTDC primary voltage monitor status register

PMS_VMONP_VDDEXTDCSTAT Offset address: 9074$_H$

VDDEXTDC primary voltage monitor status register Reset values see: Table 1518

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0				RESETOFF	RESETVAL										
r				rh	rh										

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			UV	OV	RESULT										
r			rh	rh	rh										

Field	Bits	Type	Description
RESULT	10:0	rh	**Primary monitor ADC conversion result** This bit field contains the last conversion result of the ADC measurement of the primary voltage monitor. The LSB is documented for each supply rail in the user manual (primary voltage monitors section).

(table continues...)

(continued)

Field	Bits	Type	Description
OV	11	rh	**Primary monitor over-voltage event flag** This bit is set if the primary voltage monitor recognizes an over-voltage event. An alarm is raised to the CSRM and SMU. 0_B No over-voltage condition happened. 1_B Over-voltage event indication as configured in the respective VMONP_VxxCON register.
UV	12	rh	**Primary monitor under-voltage event flag** This bit is set if the primary voltage monitor recognizes an under-voltage event. An alarm is raised to the CSRM and SMU. 0_B No under-voltage condition happened. 1_B Under-voltage event indication as configured in the respective VMONP_VxxCON register.
RESETVAL	26:16	rh	**Supply reset value status** This bit field indicates the actual cold PORST reset trim setpoint used by the primary voltage monitors.
RESETOFF	27	rh	**Reset enable status** 0_B A cold PORST is triggered in case of primary under-voltage event 1_B No cold PORST is generated in case of a primary under-voltage event.
0	15:13, 31:28	r	**Reserved**

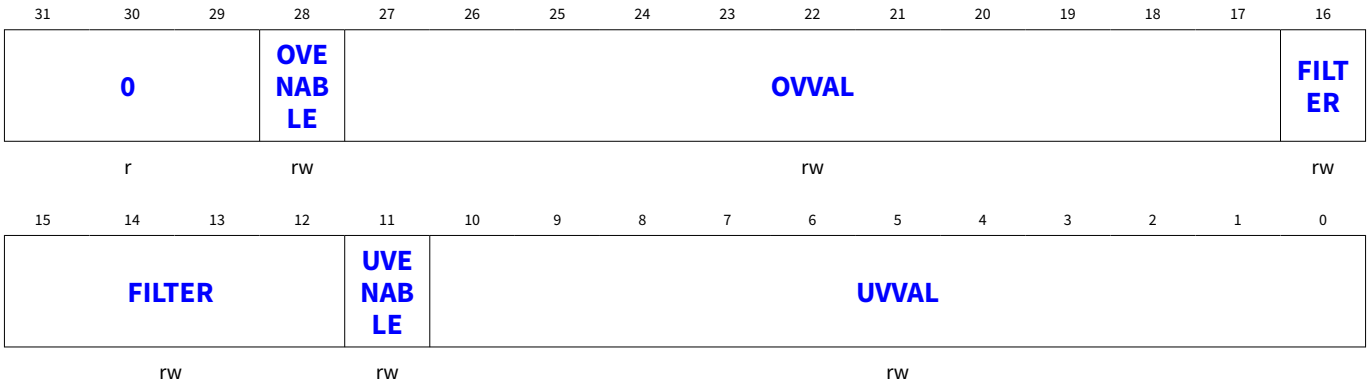
Table 1518 Reset values of PMS_VMONP_VDDEXTDCSTAT

Reset type	Reset value	Note
LVDPMS2 Reset	00F5 1800_H	
LVDPMS2 Reset for VDDEXTDC high-voltage configuration (5.5V<=VDDEXTDC(Typ)<=6V5)	0198 1800_H	Reset value with HWCFG[2:1]=11, 5V5<=VDDEXTDC<=6V5
Cold PORST for VDDEXTDC high-voltage configuration (5.5V<=VDDEXTDC(Typ)<=6V5)	0198 1800_H	Reset value with HWCFG[2:1]=11, 5V5<=VDDEXTDC<=6V5
Cold PORST	00F5 1800_H	

36.8.4.18 VDDDCLS primary voltage monitor configuration register

PMS_VMONP_VDDDCLSCON Offset address: 9078_H

VDDDCLS primary voltage monitor configuration register Reset values see: Table 1519



Field	Bits	Type	Description
UVVAL	10:0	rw	**Primary monitor alarm under-voltage threshold** This field defines the under-voltage monitoring threshold level. The LSB is documented for each supply rail in the user manual (primary voltage monitors section).
UVENABLE	11	rw	**Primary monitor under-voltage alarm enable** 0_B No alarm trigger signal is generated. 1_B An alarm trigger signal is generated and forwarded depending on the configured threshold value.
FILTER	16:12	rw	**Primary monitor alarm spike filter** 00_H Each conversion result is compared with threshold to generate alarm $1F_H$ A spike filter of consecutive 32 ADC results are used to generate alarm.
OVVAL	27:17	rw	**Primary monitor alarm over-voltage threshold** This field defines the over-voltage monitoring threshold level. The LSB is documented for each supply rail in the user manual (primary voltage monitors section).
OVENABLE	28	rw	**Primary monitor over-voltage alarm enable** 0_B No alarm trigger signal is generated. 1_B An alarm trigger signal is generated and forwarded depending on the configured threshold value.
0	31:29	r	**Reserved**

Table 1519 Reset values of PMS_VMONP_VDDDCLSCON

Reset type	Reset value	Note
LVDPMS2 Reset	13F4 F8F8$_H$	
Cold PORST	13F4 F8F8$_H$	
After Boot-FW Value	03E0 F101$_H$	

36.8.4.19 VDDDCLS primary voltage monitor reset configuration register

PMS_VMONP_VDDDCLSRST

VDDDCLS primary voltage monitor reset configuration register

Offset address: $907C_H$

Reset values see: Table 1521

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FILTER				LPF		RESETOFF_P	RESETOFF	RESETHYSEN	RESETHYS						
rw				rw		w	rw	rw	rw						

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESETHYS	RESETPTRIM								RESETTRIM						
rw	rw								rw						

Field	Bits	Type	Description
RESETTRIM	10:0	rw	**Supply reset trim value** This bit field selects the hard reset generation level of the respective supply rail. This bitfield is trimmed by Firmware. The LSB is documented for each supply rail in the user manual (primary voltage monitors section).
RESETPTRIM	14:11	rw	**Signed reset trim value** This bit field allows fine trimming of the reset threshold. Binary signed two's complement representation.
RESETHYS	22:15	rw	**Supply reset hysteresis trim value** This bit field selects the reset hysteresis offset level of the primary voltage monitor. The LSB is documented for each supply rail in the user manual (primary voltage monitors section).
RESETHYSEN	23	rw	**Reset release hysteresis enable** 0_B Hysteresis for reset release is not active. 1_B Hysteresis for reset release is active.
RESETOFF	24	rw	**Reset enable** This bit can only be changed if bit RESETOFF_P is set in parallel. RESETOFF is intended to be used only for internal test purposes and the primary reset generation is not to be disabled in customer application. 0_B A reset trigger signal is generated and forwarded to the SCU by the primary monitor depending on the selected reset trim value. 1_B No reset trigger signal is generated and forwarded to the SCU by the primary monitor.
RESETOFF_P	25	w	**RESETOFF bit protection** Writing 1 into this bit enables that bit RESETOFF can be changed in this write operation. This bit is read as zero.

(table continues...)

(continued)

Field	Bits	Type	Description
LPF	27:26	rw	**Primary monitor reset LPF coefficient** This bitfield configures the coefficient of the low pass filter of the primary monitor tracking ADC counter value. The low pass filter is used for cold PORST generation and VMONP_VxxSTAT.RESULT value. y [k] = { y [k-1] * (1-a) } + { x [k] * a} ; y [k] is filter output; x [k] is ADC output a = {1 / (2 ^ LPF)}.If LPF = 0, the filter output is the same as ADC output.
FILTER	31:28	rw	**Primary monitor reset spike filter** 0_H Each conversion result is compared with threshold to generate alarm F_H A spike filter of consecutive 16 ADC results are used to generate reset.

Table 1520 Access mode restrictions of PMS_VMONP_VDDDCLSRST sorted by descending priority

Mode name	Access mode		Description
write 1 to .**RESETOFF_P**	rw	RESETOFF	
(default)	r	RESETOFF	

Table 1521 Reset values of PMS_VMONP_VDDDCLSRST

Reset type	Reset value	Note
LVDPMS2 Reset	8C83 00F5$_H$	
Cold PORST	8C83 00F5$_H$	
After Boot-FW Value	1000 1100 1000 0011 0––– –000 1111 1110$_B$	

36.8.4.20 VDDDCLS primary voltage monitor status register

PMS_VMONP_VDDDCLSSTAT Offset address: 9080$_H$

VDDDCLS primary voltage monitor status register Reset values see: Table 1522

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0				RESETOFF	RESETVAL										
r				rh	rh										

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			UV	OV	RESULT										
r			rh	rh	rh										

Field	Bits	Type	Description
RESULT	10:0	rh	**Primary monitor ADC conversion result** This bit field contains the last conversion result of the ADC measurement of the primary voltage monitor. The LSB is documented for each supply rail in the user manual (primary voltage monitors section).
OV	11	rh	**Primary monitor over-voltage event flag** This bit is set if the primary voltage monitor recognizes an over-voltage event. An alarm is raised to the CSRM and SMU. 0_B No over-voltage condition happened. 1_B Over-voltage event indication as configured in the respective VMONP_VxxCON register.
UV	12	rh	**Primary monitor under-voltage event flag** This bit is set if the primary voltage monitor recognizes an under-voltage event. An alarm is raised to the CSRM and SMU. 0_B No under-voltage condition happened. 1_B Under-voltage event indication as configured in the respective VMONP_VxxCON register.
RESETVAL	26:16	rh	**Supply reset value status** This bit field indicates the actual cold PORST reset trim setpoint used by the primary voltage monitors.
RESETOFF	27	rh	**Reset enable status** 0_B A cold PORST is triggered in case of primary under-voltage event 1_B No cold PORST is generated in case of a primary under-voltage event.
0	15:13, 31:28	r	**Reserved**

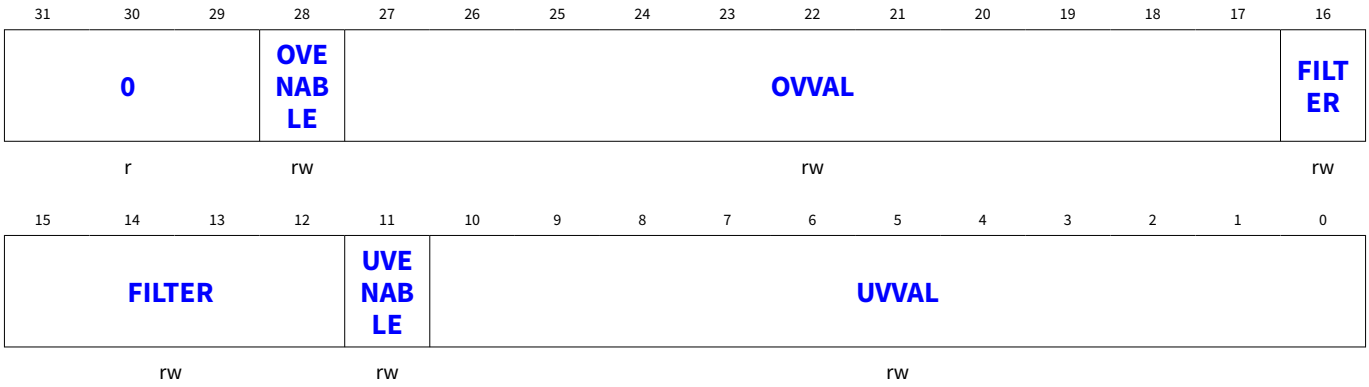
Table 1522 Reset values of PMS_VMONP_VDDDCLSSTAT

Reset type	Reset value	Note
LVDPMS2 Reset	00F5 1800$_H$	
Cold PORST	00F5 1800$_H$	

36.8.4.21 VDDEVRSB primary voltage monitor configuration register

PMS_VMONP_VDDEVRSBCON Offset address: 9084$_H$

VDDEVRSB primary voltage monitor configuration register Reset values see: Table 1523



Field	Bits	Type	Description
UVVAL	10:0	rw	**Primary monitor alarm under-voltage threshold** This field defines the under-voltage monitoring threshold level. The LSB is documented for each supply rail in the user manual (primary voltage monitors section).
UVENABLE	11	rw	**Primary monitor under-voltage alarm enable** 0_B No alarm trigger signal is generated. 1_B An alarm trigger signal is generated and forwarded depending on the configured threshold value.
FILTER	16:12	rw	**Primary monitor alarm spike filter** 00_H Each conversion result is compared with threshold to generate alarm $1F_H$ A spike filter of consecutive 32 ADC results are used to generate alarm.
OVVAL	27:17	rw	**Primary monitor alarm over-voltage threshold** This field defines the over-voltage monitoring threshold level. The LSB is documented for each supply rail in the user manual (primary voltage monitors section).
OVENABLE	28	rw	**Primary monitor over-voltage alarm enable** 0_B No alarm trigger signal is generated. 1_B An alarm trigger signal is generated and forwarded depending on the configured threshold value.
0	31:29	r	**Reserved**

Table 1523 Reset values of PMS_VMONP_VDDEVRSBCON

Reset type	Reset value	Note
LVDPMS2 Reset	1FD0 4BDA$_H$	
Cold PORST	1FD0 4BDA$_H$	
After Boot-FW Value	0F7E 4405$_H$	

36.8.4.22 VDDEVRSB primary voltage monitor reset configuration register

PMS_VMONP_VDDEVRSBRST

VDDEVRSB primary voltage monitor reset configuration register

Offset address: 9088_H

Reset values see: Table 1525

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FILTER				LPF		RESETOFF_P	RESETOFF	RESETHYSEN	RESETHYS						
rw				rw		w	rw	rw	rw						

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESETHYS	RESETPTRIM								RESETTRIM						
rw	rw								rw						

Field	Bits	Type	Description
RESETTRIM	10:0	rw	**Supply reset trim value** This bit field selects the hard reset generation level of the respective supply rail. This bitfield is trimmed by Firmware. The LSB is documented for each supply rail in the user manual (primary voltage monitors section).
RESETPTRIM	14:11	rw	**Signed reset trim value** This bit field allows fine trimming of the reset threshold. Binary signed two's complement representation.
RESETHYS	22:15	rw	**Supply reset hysteresis trim value** This bit field selects the reset hysteresis offset level of the primary voltage monitor. The LSB is documented for each supply rail in the user manual (primary voltage monitors section).
RESETHYSEN	23	rw	**Reset release hysteresis enable** 0_B Hysteresis for reset release is not active. 1_B Hysteresis for reset release is active.
RESETOFF	24	rw	**Reset enable** This bit can only be changed if bit RESETOFF_P is set in parallel. RESETOFF is intended to be used only for internal test purposes and the primary reset generation is not to be disabled in customer application. 0_B A reset trigger signal is generated and forwarded to the SCU by the primary monitor depending on the selected reset trim value. 1_B No reset trigger signal is generated and forwarded to the SCU by the primary monitor.
RESETOFF_P	25	w	**RESETOFF bit protection** Writing 1 into this bit enables that bit RESETOFF can be changed in this write operation. This bit is read as zero.

(table continues...)

(continued)

Field	Bits	Type	Description
LPF	27:26	rw	**Primary monitor reset LPF coefficient** This bitfield configures the coefficient of the low pass filter of the primary monitor tracking ADC counter value. The low pass filter is used for cold PORST generation and VMONP_VxxSTAT.RESULT value. y [k] = { y [k-1] * (1-a) } + { x [k] * a} ; y [k] is filter output; x [k] is ADC output a = {1 / (2 ^ LPF)}.If LPF = 0, the filter output is the same as ADC output.
FILTER	31:28	rw	**Primary monitor reset spike filter** 0_H Each conversion result is compared with threshold to generate alarm F_H A spike filter of consecutive 16 ADC results are used to generate reset.

Table 1524 Access mode restrictions of PMS_VMONP_VDDEVRSBRST sorted by descending priority

Mode name	Access mode		Description
write 1 to .**RESETOFF_P**	rw	RESETOFF	
(default)	r	RESETOFF	

Table 1525 Reset values of PMS_VMONP_VDDEVRSBRST

Reset type	Reset value	Note
LVDPMS2 Reset	2C89 83D3$_H$	
Cold PORST	2C89 83D3$_H$	
After Boot-FW Value	0010 1100 1000 1001 1––– –011 1111 1001$_B$	

36.8.4.23 VDDEVRSB primary voltage monitor status register

PMS_VMONP_VDDEVRSBSTAT Offset address: 908C$_H$

VDDEVRSB primary voltage monitor status register Reset values see: Table 1526

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0				RESETOFF	RESETVAL										
r				rh	rh										

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			UV	OV	RESULT										
r			rh	rh	rh										

Field	Bits	Type	Description
RESULT	10:0	rh	**Primary monitor ADC conversion result** This bit field contains the last conversion result of the ADC measurement of the primary voltage monitor. The LSB is documented for each supply rail in the user manual (primary voltage monitors section).
OV	11	rh	**Primary monitor over-voltage event flag** This bit is set if the primary voltage monitor recognizes an over-voltage event. An alarm is raised to the CSRM and SMU. 0_B No over-voltage condition happened. 1_B Over-voltage event indication as configured in the respective VMONP_VxxCON register.
UV	12	rh	**Primary monitor under-voltage event flag** This bit is set if the primary voltage monitor recognizes an under-voltage event. An alarm is raised to the CSRM and SMU. 0_B No under-voltage condition happened. 1_B Under-voltage event indication as configured in the respective VMONP_VxxCON register.
RESETVAL	26:16	rh	**Supply reset value status** This bit field indicates the actual cold PORST reset trim setpoint used by the primary voltage monitors.
RESETOFF	27	rh	**Reset enable status** 0_B A cold PORST is triggered in case of primary under-voltage event 1_B No cold PORST is generated in case of a primary under-voltage event.
0	15:13, 31:28	r	**Reserved**

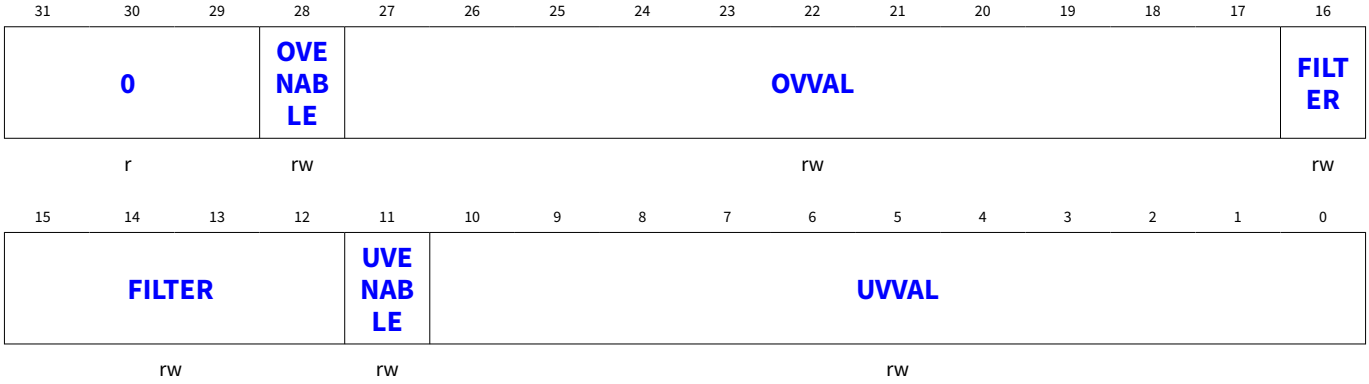
Table 1526 Reset values of PMS_VMONP_VDDEVRSBSTAT

Reset type	Reset value	Note
LVDPMS2 Reset	03D3 1800$_H$	
Cold PORST	03D3 1800$_H$	

36.8.4.24 VDDM primary voltage monitor configuration register

PMS_VMONP_VDDMCON Offset address: 9090$_H$

VDDM primary voltage monitor configuration register Reset values see: Table 1527



Field	Bits	Type	Description
UVVAL	10:0	rw	**Primary monitor alarm under-voltage threshold** This field defines the under-voltage monitoring threshold level. The LSB is documented for each supply rail in the user manual (primary voltage monitors section).
UVENABLE	11	rw	**Primary monitor under-voltage alarm enable** 0_B No alarm trigger signal is generated. 1_B An alarm trigger signal is generated and forwarded depending on the configured threshold value.
FILTER	16:12	rw	**Primary monitor alarm spike filter** 00_H Each conversion result is compared with threshold to generate alarm $1F_H$ A spike filter of consecutive 32 ADC results are used to generate alarm.
OVVAL	27:17	rw	**Primary monitor alarm over-voltage threshold** This field defines the over-voltage monitoring threshold level. The LSB is documented for each supply rail in the user manual (primary voltage monitors section).
OVENABLE	28	rw	**Primary monitor over-voltage alarm enable** 0_B No alarm trigger signal is generated. 1_B An alarm trigger signal is generated and forwarded depending on the configured threshold value.
0	31:29	r	**Reserved**

Table 1527 Reset values of PMS_VMONP_VDDMCON

Reset type	Reset value	Note
LVDPMS2 Reset	1FD0 43DA$_H$	
Cold PORST	1FD0 43DA$_H$	
After Boot-FW Value	0F7E 4405$_H$	

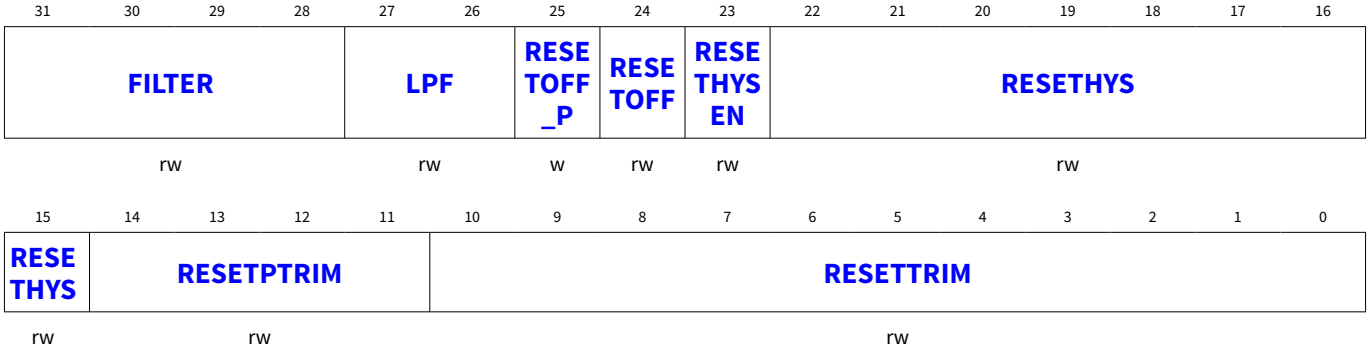
36.8.4.25 VDDM primary voltage monitor reset configuration register

PMS_VMONP_VDDMRST

VDDM primary voltage monitor reset configuration register

Offset address: 9094_H

Reset values see: Table 1529



Field	Bits	Type	Description
RESETTRIM	10:0	rw	**Supply reset trim value** This bit field selects the hard reset generation level of the respective supply rail. This bitfield is trimmed by Firmware. The LSB is documented for each supply rail in the user manual (primary voltage monitors section).
RESETPTRIM	14:11	rw	**Signed reset trim value** This bit field allows fine trimming of the reset threshold. Binary signed two's complement representation.
RESETHYS	22:15	rw	**Supply reset hysteresis trim value** This bit field selects the reset hysteresis offset level of the primary voltage monitor. The LSB is documented for each supply rail in the user manual (primary voltage monitors section).
RESETHYSEN	23	rw	**Reset release hysteresis enable** 0_B Hysteresis for reset release is not active. 1_B Hysteresis for reset release is active.
RESETOFF	24	rw	**Reset enable** This bit can only be changed if bit RESETOFF_P is set in parallel. RESETOFF is intended to be used only for internal test purposes and the primary reset generation is not to be disabled in customer application. 0_B A reset trigger signal is generated and forwarded to the SCU by the primary monitor depending on the selected reset trim value. 1_B No reset trigger signal is generated and forwarded to the SCU by the primary monitor.
RESETOFF_P	25	w	**RESETOFF bit protection** Writing 1 into this bit enables that bit RESETOFF can be changed in this write operation. This bit is read as zero.

(table continues...)

(continued)

Field	Bits	Type	Description
LPF	27:26	rw	**Primary monitor reset LPF coefficient** This bitfield configures the coefficient of the low pass filter of the primary monitor tracking ADC counter value. The low pass filter is used for cold PORST generation and VMONP_VxxSTAT.RESULT value. y [k] = { y [k-1] * (1-a) } + { x [k] * a} ; y [k] is filter output; x [k] is ADC output a = {1 / (2 ^ LPF)}.If LPF = 0, the filter output is the same as ADC output.
FILTER	31:28	rw	**Primary monitor reset spike filter** 0_H Each conversion result is compared with threshold to generate alarm F_H A spike filter of consecutive 16 ADC results are used to generate reset.

Table 1528 Access mode restrictions of PMS_VMONP_VDDMRST sorted by descending priority

Mode name	Access mode		Description
write 1 to .**RESETOFF_P**	rw	RESETOFF	
(default)	r	RESETOFF	

Table 1529 Reset values of PMS_VMONP_VDDMRST

Reset type	Reset value	Note
LVDPMS2 Reset	2C89 83D3$_H$	
Cold PORST	2C89 83D3$_H$	
After Boot-FW Value	0010 1100 1000 1001 1––– –011 1111 1001$_B$	

36.8.4.26 VDDM primary voltage monitor status register

PMS_VMONP_VDDMSTAT Offset address: 9098$_H$

VDDM primary voltage monitor status register Reset values see: Table 1530

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0				RESETOFF	RESETVAL										
r				rh	rh										

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			UV	OV	RESULT										
r			rh	rh	rh										

Field	Bits	Type	Description
RESULT	10:0	rh	**Primary monitor ADC conversion result** This bit field contains the last conversion result of the ADC measurement of the primary voltage monitor. The LSB is documented for each supply rail in the user manual (primary voltage monitors section).
OV	11	rh	**Primary monitor over-voltage event flag** This bit is set if the primary voltage monitor recognizes an over-voltage event. An alarm is raised to the CSRM and SMU. 0_B No over-voltage condition happened. 1_B Over-voltage event indication as configured in the respective VMONP_VxxCON register.
UV	12	rh	**Primary monitor under-voltage event flag** This bit is set if the primary voltage monitor recognizes an under-voltage event. An alarm is raised to the CSRM and SMU. 0_B No under-voltage condition happened. 1_B Under-voltage event indication as configured in the respective VMONP_VxxCON register.
RESETVAL	26:16	rh	**Supply reset value status** This bit field indicates the actual cold PORST reset trim setpoint used by the primary voltage monitors.
RESETOFF	27	rh	**Reset enable status** 0_B A cold PORST is triggered in case of primary under-voltage event 1_B No cold PORST is generated in case of a primary under-voltage event.
0	15:13, 31:28	r	**Reserved**

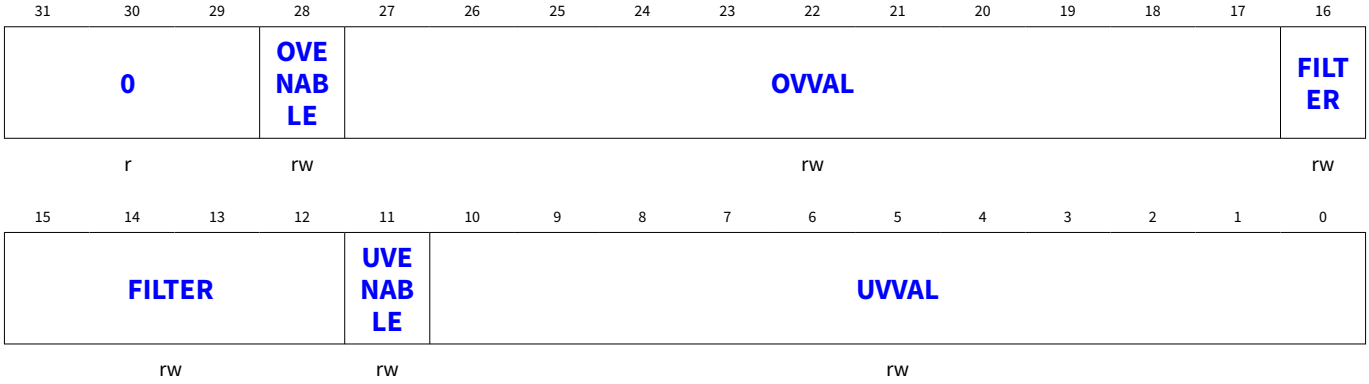
Table 1530 Reset values of PMS_VMONP_VDDMSTAT

Reset type	Reset value	Note
LVDPMS2 Reset	03D3 0800$_H$	
Cold PORST	03D3 0800$_H$	

36.8.4.27 VDDPHPHY0 primary voltage monitor configuration register

PMS_VMONP_VDDPHPHY0CON Offset address: 909C$_H$

VDDPHPHY0 primary voltage monitor configuration register Reset values see: Table 1531



Field	Bits	Type	Description
UVVAL	10:0	rw	**Primary monitor alarm under-voltage threshold** This field defines the under-voltage monitoring threshold level. The LSB is documented for each supply rail in the user manual (primary voltage monitors section).
UVENABLE	11	rw	**Primary monitor under-voltage alarm enable** 0_B No alarm trigger signal is generated. 1_B An alarm trigger signal is generated and forwarded depending on the configured threshold value.
FILTER	16:12	rw	**Primary monitor alarm spike filter** 00_H Each conversion result is compared with threshold to generate alarm $1F_H$ A spike filter of consecutive 32 ADC results are used to generate alarm.
OVVAL	27:17	rw	**Primary monitor alarm over-voltage threshold** This field defines the over-voltage monitoring threshold level. The LSB is documented for each supply rail in the user manual (primary voltage monitors section).
OVENABLE	28	rw	**Primary monitor over-voltage alarm enable** 0_B No alarm trigger signal is generated. 1_B An alarm trigger signal is generated and forwarded depending on the configured threshold value.
0	31:29	r	**Reserved**

Table 1531 Reset values of PMS_VMONP_VDDPHPHY0CON

Reset type	Reset value	Note
LVDPMS2 Reset	0594 4231$_H$	
Cold PORST	0594 4231$_H$	

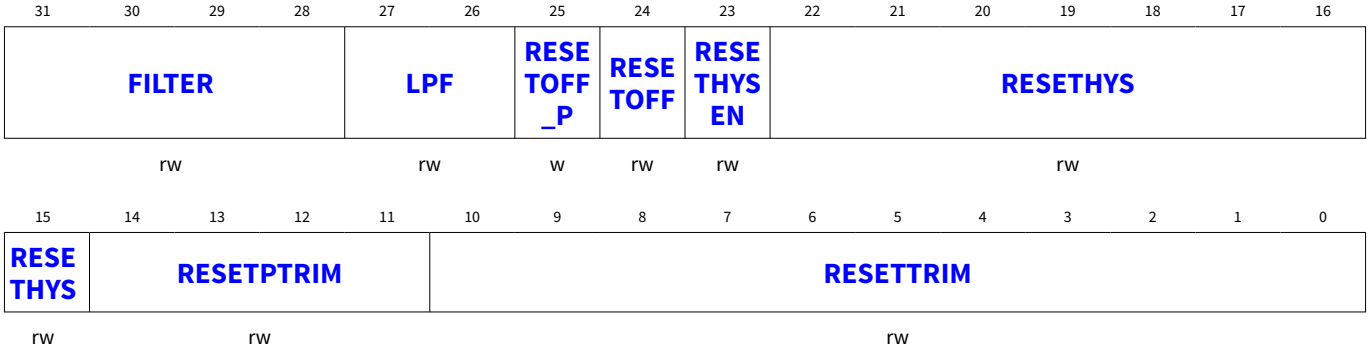
36.8.4.28 VDDPHPHY0 primary voltage monitor reset configuration register

PMS_VMONP_VDDPHPHY0RST

VDDPHPHY0 primary voltage monitor reset configuration register

Offset address: 90A0$_H$

Reset values see: Table 1533



Field	Bits	Type	Description
RESETTRIM	10:0	rw	**Supply reset trim value** This bit field selects the hard reset generation level of the respective supply rail. This bitfield is trimmed by Firmware. The LSB is documented for each supply rail in the user manual (primary voltage monitors section).
RESETPTRIM	14:11	rw	**Signed reset trim value** This bit field allows fine trimming of the reset threshold. Binary signed two's complement representation.
RESETHYS	22:15	rw	**Supply reset hysteresis trim value** This bit field selects the reset hysteresis offset level of the primary voltage monitor. The LSB is documented for each supply rail in the user manual (primary voltage monitors section).
RESETHYSEN	23	rw	**Reset release hysteresis enable** 0_B Hysteresis for reset release is not active. 1_B Hysteresis for reset release is active.
RESETOFF	24	rw	**Reset enable** This bit can only be changed if bit RESETOFF_P is set in parallel. RESETOFF is intended to be used only for internal test purposes and the primary reset generation is not to be disabled in customer application. 0_B A reset trigger signal is generated and forwarded to the SCU by the primary monitor depending on the selected reset trim value. 1_B No reset trigger signal is generated and forwarded to the SCU by the primary monitor.
RESETOFF_P	25	w	**RESETOFF bit protection** Writing 1 into this bit enables that bit RESETOFF can be changed in this write operation. This bit is read as zero.
LPF	27:26	rw	**Primary monitor reset LPF coefficient** This bitfield configures the coefficient of the low pass filter of the primary monitor tracking ADC counter value. The low pass filter is used for cold PORST generation and VMONP_VxxSTAT.RESULT value. y [k] = { y [k-1] * (1-a) } + { x [k] * a} ; y [k] is filter output; x [k] is ADC output a = {1 / (2 ^ LPF)}.If LPF = 0, the filter output is the same as ADC output.

(table continues...)

(continued)

Field	Bits	Type	Description
FILTER	31:28	rw	**Primary monitor reset spike filter** 0_H Each conversion result is compared with threshold to generate alarm F_H A spike filter of consecutive 16 ADC results are used to generate reset.

Table 1532 Access mode restrictions of PMS_VMONP_VDDPHPHY0RST sorted by descending priority

Mode name	Access mode		Description
write 1 to .**RESETOFF_P**	rw	RESETOFF	
(default)	r	RESETOFF	

Table 1533 Reset values of PMS_VMONP_VDDPHPHY0RST

Reset type	Reset value	Note
LVDPMS2 Reset	2D85 0220_H	
Cold PORST	2D85 0220_H	
After Boot-FW Value	0010 1101 1000 0101 0––– –010 0010 1011_B	

36.8.4.29 VDDPHPHY0 primary voltage monitor status register

PMS_VMONP_VDDPHPHY0STAT Offset address: $90A4_H$

VDDPHPHY0 primary voltage monitor status register Reset values see: Table 1534

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0				RESETOFF	RESETVAL										
r				rh	rh										

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			UV	OV	RESULT										
r			rh	rh	rh										

Field	Bits	Type	Description
RESULT	10:0	rh	**Primary monitor ADC conversion result** This bit field contains the last conversion result of the ADC measurement of the primary voltage monitor. The LSB is documented for each supply rail in the user manual (primary voltage monitors section).

(table continues...)

(continued)

Field	Bits	Type	Description
OV	11	rh	**Primary monitor over-voltage event flag** This bit is set if the primary voltage monitor recognizes an over-voltage event. An alarm is raised to the CSRM and SMU. 0_B No over-voltage condition happened. 1_B Over-voltage event indication as configured in the respective VMONP_VxxCON register.
UV	12	rh	**Primary monitor under-voltage event flag** This bit is set if the primary voltage monitor recognizes an under-voltage event. An alarm is raised to the CSRM and SMU. 0_B No under-voltage condition happened. 1_B Under-voltage event indication as configured in the respective VMONP_VxxCON register.
RESETVAL	26:16	rh	**Supply reset value status** This bit field indicates the actual cold PORST reset trim setpoint used by the primary voltage monitors.
RESETOFF	27	rh	**Reset enable status** 0_B A cold PORST is triggered in case of primary under-voltage event 1_B No cold PORST is generated in case of a primary under-voltage event.
0	15:13, 31:28	r	**Reserved**

Table 1534 Reset values of PMS_VMONP_VDDPHPHY0STAT

Reset type	Reset value	Note
LVDPMS2 Reset	0A20 0000$_H$	
Cold PORST	0A20 0000$_H$	

36.8.4.30 VDDPHPHY1 primary voltage monitor configuration register

PMS_VMONP_VDDPHPHY1CON Offset address: 90A8$_H$

VDDPHPHY1 primary voltage monitor configuration register Reset values see: Table 1535

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0			OVENABLE	OVVAL											FILTER
r			rw	rw											rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FILTER				UVENABLE	UVVAL										
rw				rw	rw										

Field	Bits	Type	Description
UVVAL	10:0	rw	**Primary monitor alarm under-voltage threshold** This field defines the under-voltage monitoring threshold level. The LSB is documented for each supply rail in the user manual (primary voltage monitors section).
UVENABLE	11	rw	**Primary monitor under-voltage alarm enable** 0_B No alarm trigger signal is generated. 1_B An alarm trigger signal is generated and forwarded depending on the configured threshold value.
FILTER	16:12	rw	**Primary monitor alarm spike filter** 00_H Each conversion result is compared with threshold to generate alarm $1F_H$ A spike filter of consecutive 32 ADC results are used to generate alarm.
OVVAL	27:17	rw	**Primary monitor alarm over-voltage threshold** This field defines the over-voltage monitoring threshold level. The LSB is documented for each supply rail in the user manual (primary voltage monitors section).
OVENABLE	28	rw	**Primary monitor over-voltage alarm enable** 0_B No alarm trigger signal is generated. 1_B An alarm trigger signal is generated and forwarded depending on the configured threshold value.
0	31:29	r	**Reserved**

Table 1535 Reset values of PMS_VMONP_VDDPHPHY1CON

Reset type	Reset value	Note
LVDPMS2 Reset	$0594\ 4231_H$	
Cold PORST	$0594\ 4231_H$	

36.8.4.31 VDDPHPHY1 primary voltage monitor reset configuration register

PMS_VMONP_VDDPHPHY1RST

VDDPHPHY1 primary voltage monitor reset configuration register

Offset address: $90AC_H$

Reset values see: Table 1537

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FILTER				LPF		RESETOFF_P	RESETOFF	RESETHYSEN	RESETHYS						
rw				rw		w	rw	rw	rw						

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESETHYS	RESETPTRIM							RESETTRIM							
rw	rw							rw							

Field	Bits	Type	Description
RESETTRIM	10:0	rw	**Supply reset trim value** This bit field selects the hard reset generation level of the respective supply rail. This bitfield is trimmed by Firmware. The LSB is documented for each supply rail in the user manual (primary voltage monitors section).
RESETPTRIM	14:11	rw	**Signed reset trim value** This bit field allows fine trimming of the reset threshold. Binary signed two's complement representation.
RESETHYS	22:15	rw	**Supply reset hysteresis trim value** This bit field selects the reset hysteresis offset level of the primary voltage monitor. The LSB is documented for each supply rail in the user manual (primary voltage monitors section).
RESETHYSEN	23	rw	**Reset release hysteresis enable** 0_B Hysteresis for reset release is not active. 1_B Hysteresis for reset release is active.
RESETOFF	24	rw	**Reset enable** This bit can only be changed if bit RESETOFF_P is set in parallel. RESETOFF is intended to be used only for internal test purposes and the primary reset generation is not to be disabled in customer application. 0_B A reset trigger signal is generated and forwarded to the SCU by the primary monitor depending on the selected reset trim value. 1_B No reset trigger signal is generated and forwarded to the SCU by the primary monitor.
RESETOFF_P	25	w	**RESETOFF bit protection** Writing 1 into this bit enables that bit RESETOFF can be changed in this write operation. This bit is read as zero.
LPF	27:26	rw	**Primary monitor reset LPF coefficient** This bitfield configures the coefficient of the low pass filter of the primary monitor tracking ADC counter value. The low pass filter is used for cold PORST generation and VMONP_VxxSTAT.RESULT value. y [k] = { y [k-1] * (1-a) } + { x [k] * a} ; y [k] is filter output; x [k] is ADC output a = {1 / (2 ^ LPF)}.If LPF = 0, the filter output is the same as ADC output.
FILTER	31:28	rw	**Primary monitor reset spike filter** 0_H Each conversion result is compared with threshold to generate alarm F_H A spike filter of consecutive 16 ADC results are used to generate reset.

Table 1536 Access mode restrictions of PMS_VMONP_VDDPHPHY1RST sorted by descending priority

Mode name	Access mode		Description
write 1 to .**RESETOFF_P**	rw	RESETOFF	

(table continues...)

Table 1536 (continued) Access mode restrictions of PMS_VMONP_VDDPHPHY1RST sorted by descending priority

Mode name	Access mode		Description
(default)	r	RESETOFF	

Table 1537 Reset values of PMS_VMONP_VDDPHPHY1RST

Reset type	Reset value	Note
LVDPMS2 Reset	2D85 0220_H	
Cold PORST	2D85 0220_H	
After Boot-FW Value	0010 1101 1000 0101 0––– –010 0010 1011_B	

36.8.4.32 VDDPHPHY1 primary voltage monitor status register

PMS_VMONP_VDDPHPHY1STAT Offset address: $90B0_H$

VDDPHPHY1 primary voltage monitor status register Reset values see: Table 1538

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0				RESETOFF	RESETVAL										
r				rh	rh										

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			UV	OV	RESULT										
r			rh	rh	rh										

Field	Bits	Type	Description
RESULT	10:0	rh	**Primary monitor ADC conversion result** This bit field contains the last conversion result of the ADC measurement of the primary voltage monitor. The LSB is documented for each supply rail in the user manual (primary voltage monitors section).
OV	11	rh	**Primary monitor over-voltage event flag** This bit is set if the primary voltage monitor recognizes an over-voltage event. An alarm is raised to the CSRM and SMU. 0_B No over-voltage condition happened. 1_B Over-voltage event indication as configured in the respective VMONP_VxxCON register.
UV	12	rh	**Primary monitor under-voltage event flag** This bit is set if the primary voltage monitor recognizes an under-voltage event. An alarm is raised to the CSRM and SMU. 0_B No under-voltage condition happened. 1_B Under-voltage event indication as configured in the respective VMONP_VxxCON register.

(table continues...)

(continued)

Field	Bits	Type	Description
RESETVAL	26:16	rh	**Supply reset value status** This bit field indicates the actual cold PORST reset trim setpoint used by the primary voltage monitors.
RESETOFF	27	rh	**Reset enable status** 0_B A cold PORST is triggered in case of primary under-voltage event 1_B No cold PORST is generated in case of a primary under-voltage event.
0	15:13, 31:28	r	**Reserved**

Table 1538 Reset values of PMS_VMONP_VDDPHPHY1STAT

Reset type	Reset value	Note
LVDPMS2 Reset	0A20 0000$_H$	
Cold PORST	0A20 0000$_H$	

36.8.4.33 VDDPHPHY2 primary voltage monitor configuration register

PMS_VMONP_VDDPHPHY2CON Offset address: 90B4$_H$

VDDPHPHY2 primary voltage monitor configuration register Reset values see: Table 1539

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0			OVENABLE	OVVAL											FILTER
r			rw	rw											rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FILTER				UVENABLE	UVVAL										
rw				rw	rw										

Field	Bits	Type	Description
UVVAL	10:0	rw	**Primary monitor alarm under-voltage threshold** This field defines the under-voltage monitoring threshold level. The LSB is documented for each supply rail in the user manual (primary voltage monitors section).
UVENABLE	11	rw	**Primary monitor under-voltage alarm enable** 0_B No alarm trigger signal is generated. 1_B An alarm trigger signal is generated and forwarded depending on the configured threshold value.

(table continues...)

(continued)

Field	Bits	Type	Description
FILTER	16:12	rw	**Primary monitor alarm spike filter** 00_H Each conversion result is compared with threshold to generate alarm $1F_H$ A spike filter of consecutive 32 ADC results are used to generate alarm.
OVVAL	27:17	rw	**Primary monitor alarm over-voltage threshold** This field defines the over-voltage monitoring threshold level. The LSB is documented for each supply rail in the user manual (primary voltage monitors section).
OVENABLE	28	rw	**Primary monitor over-voltage alarm enable** 0_B No alarm trigger signal is generated. 1_B An alarm trigger signal is generated and forwarded depending on the configured threshold value.
0	31:29	r	**Reserved**

Table 1539 Reset values of PMS_VMONP_VDDPHPHY2CON

Reset type	Reset value	Note
LVDPMS2 Reset	0594 4231$_H$	
Cold PORST	0594 4231$_H$	
After Boot-FW Value	0594 4231$_H$	

36.8.4.34 VDDPHPHY2 primary voltage monitor reset configuration register

PMS_VMONP_VDDPHPHY2RST

Offset address: $90B8_H$

VDDPHPHY2 primary voltage monitor reset configuration register

Reset values see: Table 1541

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FILTER				LPF		RESETOFF_P	RESETOFF	RESETHYSEN	RESETHYS						
rw				rw		w	rw	rw	rw						

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESETHYS	RESETPTRIM				RESETTRIM										
rw	rw				rw										

Field	Bits	Type	Description
RESETTRIM	10:0	rw	**Supply reset trim value** This bit field selects the hard reset generation level of the respective supply rail. This bitfield is trimmed by Firmware. The LSB is documented for each supply rail in the user manual (primary voltage monitors section).
RESETPTRIM	14:11	rw	**Signed reset trim value** This bit field allows fine trimming of the reset threshold. Binary signed two's complement representation.
RESETHYS	22:15	rw	**Supply reset hysteresis trim value** This bit field selects the reset hysteresis offset level of the primary voltage monitor. The LSB is documented for each supply rail in the user manual (primary voltage monitors section).
RESETHYSEN	23	rw	**Reset release hysteresis enable** 0_B Hysteresis for reset release is not active. 1_B Hysteresis for reset release is active.
RESETOFF	24	rw	**Reset enable** This bit can only be changed if bit RESETOFF_P is set in parallel. RESETOFF is intended to be used only for internal test purposes and the primary reset generation is not to be disabled in customer application. 0_B A reset trigger signal is generated and forwarded to the SCU by the primary monitor depending on the selected reset trim value. 1_B No reset trigger signal is generated and forwarded to the SCU by the primary monitor.
RESETOFF_P	25	w	**RESETOFF bit protection** Writing 1 into this bit enables that bit RESETOFF can be changed in this write operation. This bit is read as zero.
LPF	27:26	rw	**Primary monitor reset LPF coefficient** This bitfield configures the coefficient of the low pass filter of the primary monitor tracking ADC counter value. The low pass filter is used for cold PORST generation and VMONP_VxxSTAT.RESULT value. y [k] = { y [k-1] * (1-a) } + { x [k] * a} ; y [k] is filter output; x [k] is ADC output a = {1 / (2 ^ LPF)}.If LPF = 0, the filter output is the same as ADC output.
FILTER	31:28	rw	**Primary monitor reset spike filter** 0_H Each conversion result is compared with threshold to generate alarm F_H A spike filter of consecutive 16 ADC results are used to generate reset.

Table 1540 Access mode restrictions of PMS_VMONP_VDDPHPHY2RST sorted by descending priority

Mode name	Access mode		Description
write 1 to .**RESETOFF_P**	rw	RESETOFF	

(table continues...)

Table 1540 (continued) Access mode restrictions of PMS_VMONP_VDDPHPHY2RST sorted by descending priority

Mode name	Access mode		Description
(default)	r	RESETOFF	

Table 1541 Reset values of PMS_VMONP_VDDPHPHY2RST

Reset type	Reset value	Note
LVDPMS2 Reset	2D85 0220_H	
Cold PORST	2D85 0220_H	
After Boot-FW Value	0010 1101 1000 0101 0––– –010 0010 1011_B	

36.8.4.35 VDDPHPHY2 primary voltage monitor status register

PMS_VMONP_VDDPHPHY2STAT Offset address: $90BC_H$

VDDPHPHY2 primary voltage monitor status register Reset values see: Table 1542

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0				RESETOFF	RESETVAL										
r				rh	rh										

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			UV	OV	RESULT										
r			rh	rh	rh										

Field	Bits	Type	Description
RESULT	10:0	rh	**Primary monitor ADC conversion result** This bit field contains the last conversion result of the ADC measurement of the primary voltage monitor. The LSB is documented for each supply rail in the user manual (primary voltage monitors section).
OV	11	rh	**Primary monitor over-voltage event flag** This bit is set if the primary voltage monitor recognizes an over-voltage event. An alarm is raised to the CSRM and SMU. 0_B No over-voltage condition happened. 1_B Over-voltage event indication as configured in the respective VMONP_VxxCON register.
UV	12	rh	**Primary monitor under-voltage event flag** This bit is set if the primary voltage monitor recognizes an under-voltage event. An alarm is raised to the CSRM and SMU. 0_B No under-voltage condition happened. 1_B Under-voltage event indication as configured in the respective VMONP_VxxCON register.

(table continues...)

(continued)

Field	Bits	Type	Description
RESETVAL	26:16	rh	**Supply reset value status** This bit field indicates the actual cold PORST reset trim setpoint used by the primary voltage monitors.
RESETOFF	27	rh	**Reset enable status** 0_B A cold PORST is triggered in case of primary under-voltage event 1_B No cold PORST is generated in case of a primary under-voltage event.
0	15:13, 31:28	r	**Reserved**

Table 1542 Reset values of PMS_VMONP_VDDPHPHY2STAT

Reset type	Reset value	Note
LVDPMS2 Reset	0A20 0000$_H$	
Cold PORST	0A20 0000$_H$	

36.8.4.36 VDDHSIF primary voltage monitor configuration register

PMS_VMONP_VDDHSIFCON

VDDHSIF primary voltage monitor configuration register

Offset address: 90C0$_H$

Reset values see: Table 1543

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0			OVENABLE	OVVAL											FILTER
r			rw	rw											rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FILTER				UVENABLE	UVVAL										
rw				rw	rw										

Field	Bits	Type	Description
UVVAL	10:0	rw	**Primary monitor alarm under-voltage threshold** This field defines the under-voltage monitoring threshold level. The LSB is documented for each supply rail in the user manual (primary voltage monitors section).
UVENABLE	11	rw	**Primary monitor under-voltage alarm enable** 0_B No alarm trigger signal is generated. 1_B An alarm trigger signal is generated and forwarded depending on the configured threshold value.

(table continues...)

(continued)

Field	Bits	Type	Description
FILTER	16:12	rw	**Primary monitor alarm spike filter** 00_H Each conversion result is compared with threshold to generate alarm $1F_H$ A spike filter of consecutive 32 ADC results are used to generate alarm.
OVVAL	27:17	rw	**Primary monitor alarm over-voltage threshold** This field defines the over-voltage monitoring threshold level. The LSB is documented for each supply rail in the user manual (primary voltage monitors section).
OVENABLE	28	rw	**Primary monitor over-voltage alarm enable** 0_B No alarm trigger signal is generated. 1_B An alarm trigger signal is generated and forwarded depending on the configured threshold value.
0	31:29	r	**Reserved**

Table 1543 Reset values of PMS_VMONP_VDDHSIFCON

Reset type	Reset value	Note
LVDPMS2 Reset	0A3A 423F$_H$	
Cold PORST	0A3A 423F$_H$	

36.8.4.37 VDDHSIF primary voltage monitor reset configuration register

PMS_VMONP_VDDHSIFRST Offset address: 90C4$_H$

VDDHSIF primary voltage monitor reset configuration register Reset values see: Table 1545

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FILTER				LPF		RESETOFF_P	RESETOFF	RESETHYSEN	RESETHYS						
rw				rw		w	rw	rw	rw						

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESETHYS	RESETPTRIM							RESETTRIM							
rw	rw							rw							

Field	Bits	Type	Description
RESETTRIM	10:0	rw	**Supply reset trim value** This bit field selects the hard reset generation level of the respective supply rail. This bitfield is trimmed by Firmware. The LSB is documented for each supply rail in the user manual (primary voltage monitors section).

(table continues...)

(continued)

Field	Bits	Type	Description
RESETPTRIM	14:11	rw	**Signed reset trim value** This bit field allows fine trimming of the reset threshold. Binary signed two's complement representation.
RESETHYS	22:15	rw	**Supply reset hysteresis trim value** This bit field selects the reset hysteresis offset level of the primary voltage monitor. The LSB is documented for each supply rail in the user manual (primary voltage monitors section).
RESETHYSEN	23	rw	**Reset release hysteresis enable** 0_B Hysteresis for reset release is not active. 1_B Hysteresis for reset release is active.
RESETOFF	24	rw	**Reset enable** This bit can only be changed if bit RESETOFF_P is set in parallel. RESETOFF is intended to be used only for internal test purposes and the primary reset generation is not to be disabled in customer application. 0_B A reset trigger signal is generated and forwarded to the SCU by the primary monitor depending on the selected reset trim value. 1_B No reset trigger signal is generated and forwarded to the SCU by the primary monitor.
RESETOFF_P	25	w	**RESETOFF bit protection** Writing 1 into this bit enables that bit RESETOFF can be changed in this write operation. This bit is read as zero.
LPF	27:26	rw	**Primary monitor reset LPF coefficient** This bitfield configures the coefficient of the low pass filter of the primary monitor tracking ADC counter value. The low pass filter is used for cold PORST generation and VMONP_VxxSTAT.RESULT value. y [k] = { y [k-1] * (1-a) } + { x [k] * a} ; y [k] is filter output; x [k] is ADC output a = {1 / (2 ^ LPF)}.If LPF = 0, the filter output is the same as ADC output.
FILTER	31:28	rw	**Primary monitor reset spike filter** 0_H Each conversion result is compared with threshold to generate alarm F_H A spike filter of consecutive 16 ADC results are used to generate reset.

Table 1544 Access mode restrictions of PMS_VMONP_VDDHSIFRST sorted by descending priority

Mode name	Access mode		Description
write 1 to .**RESETOFF_P**	rw	RESETOFF	
(default)	r	RESETOFF	

Table 1545 Reset values of PMS_VMONP_VDDHSIFRST

Reset type	Reset value	Note
LVDPMS2 Reset	2D86 822E$_H$	
Cold PORST	2D86 822E$_H$	
After Boot-FW Value	0010 1101 1000 0110 1––– –010 0011 1001$_B$	

36.8.4.38 VDDHSIF primary voltage monitor status register

PMS_VMONP_VDDHSIFSTAT Offset address: 90C8$_H$

VDDHSIF primary voltage monitor status register Reset values see: Table 1546

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0				RESETOFF	RESETVAL										
r				rh	rh										

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			UV	OV	RESULT										
r			rh	rh	rh										

Field	Bits	Type	Description
RESULT	10:0	rh	**Primary monitor ADC conversion result** This bit field contains the last conversion result of the ADC measurement of the primary voltage monitor. The LSB is documented for each supply rail in the user manual (primary voltage monitors section).
OV	11	rh	**Primary monitor over-voltage event flag** This bit is set if the primary voltage monitor recognizes an over-voltage event. An alarm is raised to the CSRM and SMU. 0_B No over-voltage condition happened. 1_B Over-voltage event indication as configured in the respective VMONP_VxxCON register.
UV	12	rh	**Primary monitor under-voltage event flag** This bit is set if the primary voltage monitor recognizes an under-voltage event. An alarm is raised to the CSRM and SMU. 0_B No under-voltage condition happened. 1_B Under-voltage event indication as configured in the respective VMONP_VxxCON register.
RESETVAL	26:16	rh	**Supply reset value status** This bit field indicates the actual cold PORST reset trim setpoint used by the primary voltage monitors.

(table continues...)

(continued)

Field	Bits	Type	Description
RESETOFF	27	rh	**Reset enable status** 0_B A cold PORST is triggered in case of primary under-voltage event 1_B No cold PORST is generated in case of a primary under-voltage event.
0	15:13, 31:28	r	**Reserved**

Table 1546 Reset values of PMS_VMONP_VDDHSIFSTAT

Reset type	Reset value	Note
LVDPMS2 Reset	0A2E 0000$_H$	
Cold PORST	0A2E 0000$_H$	

36.8.4.39 VDDPAD primary voltage monitor configuration register

PMS_VMONP_VDDPADCON Offset address: 90D8$_H$

VDDPAD primary voltage monitor configuration register Reset values see: Table 1547

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0			OVENABLE	OVVAL											FILTER
r			rw	rw											rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FILTER				UVENABLE	UVVAL										
rw				rw	rw										

Field	Bits	Type	Description
UVVAL	10:0	rw	**Primary monitor alarm under-voltage threshold** This field defines the under-voltage monitoring threshold level. The LSB is documented for each supply rail in the user manual (primary voltage monitors section).
UVENABLE	11	rw	**Primary monitor under-voltage alarm enable** 0_B No alarm trigger signal is generated. 1_B An alarm trigger signal is generated and forwarded depending on the configured threshold value.
FILTER	16:12	rw	**Primary monitor alarm spike filter** 00_H Each conversion result is compared with threshold to generate alarm $1F_H$ A spike filter of consecutive 32 ADC results are used to generate alarm.

(table continues...)

(continued)

Field	Bits	Type	Description
OVVAL	27:17	rw	**Primary monitor alarm over-voltage threshold** This field defines the over-voltage monitoring threshold level. The LSB is documented for each supply rail in the user manual (primary voltage monitors section).
OVENABLE	28	rw	**Primary monitor over-voltage alarm enable** 0_B No alarm trigger signal is generated. 1_B An alarm trigger signal is generated and forwarded depending on the configured threshold value.
0	31:29	r	**Reserved**

Table 1547 Reset values of PMS_VMONP_VDDPADCON

Reset type	Reset value	Note
LVDPMS2 Reset	0F58 45DD$_H$	
Cold PORST	0F58 45DD$_H$	

36.8.4.40 VDDPAD primary voltage monitor reset configuration register

PMS_VMONP_VDDPADRST Offset address: 90DC$_H$

VDDPAD primary voltage monitor reset configuration register Reset values see: Table 1549

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FILTER				LPF		RESETOFF_P	RESETOFF	RESETHYSEN	RESETHYS						
rw				rw		w	rw	rw	rw						

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESETHYS	RESETPTRIM							RESETTRIM							
rw	rw							rw							

Field	Bits	Type	Description
RESETTRIM	10:0	rw	**Supply reset trim value** This bit field selects the hard reset generation level of the respective supply rail. This bitfield is trimmed by Firmware. The LSB is documented for each supply rail in the user manual (primary voltage monitors section).
RESETPTRIM	14:11	rw	**Signed reset trim value** This bit field allows fine trimming of the reset threshold. Binary signed two's complement representation.

(table continues...)

(continued)

Field	Bits	Type	Description
RESETHYS	22:15	rw	**Supply reset hysteresis trim value** This bit field selects the reset hysteresis offset level of the primary voltage monitor. The LSB is documented for each supply rail in the user manual (primary voltage monitors section).
RESETHYSEN	23	rw	**Reset release hysteresis enable** 0_B Hysteresis for reset release is not active. 1_B Hysteresis for reset release is active.
RESETOFF	24	rw	**Reset enable** This bit can only be changed if bit RESETOFF_P is set in parallel. RESETOFF is intended to be used only for internal test purposes and the primary reset generation is not to be disabled in customer application. 0_B A reset trigger signal is generated and forwarded to the SCU by the primary monitor depending on the selected reset trim value. 1_B No reset trigger signal is generated and forwarded to the SCU by the primary monitor.
RESETOFF_P	25	w	**RESETOFF bit protection** Writing 1 into this bit enables that bit RESETOFF can be changed in this write operation. This bit is read as zero.
LPF	27:26	rw	**Primary monitor reset LPF coefficient** This bitfield configures the coefficient of the low pass filter of the primary monitor tracking ADC counter value. The low pass filter is used for cold PORST generation and VMONP_VxxSTAT.RESULT value. y [k] = { y [k-1] * (1-a) } + { x [k] * a} ; y [k] is filter output; x [k] is ADC output a = {1 / (2 ^ LPF)}.If LPF = 0, the filter output is the same as ADC output.
FILTER	31:28	rw	**Primary monitor reset spike filter** 0_H Each conversion result is compared with threshold to generate alarm F_H A spike filter of consecutive 16 ADC results are used to generate reset.

Table 1548 Access mode restrictions of PMS_VMONP_VDDPADRST sorted by descending priority

Mode name	Access mode		Description
write 1 to .**RESETOFF_P**	rw	RESETOFF	
(default)	r	RESETOFF	

Table 1549 Reset values of PMS_VMONP_VDDPADRST

Reset type	Reset value	Note
LVDPMS2 Reset	2C8F 0580$_H$	
Cold PORST	2C8F 0580$_H$	
After Boot-FW Value	0010 1100 1000 1111 0––– –101 1100 1101$_B$	

36.8.4.41 VDDPAD primary voltage monitor status register

PMS_VMONP_VDDPADSTAT Offset address: $90E0_H$

VDDPAD primary voltage monitor status register Reset values see: Table 1550

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0				RESETOFF	RESETVAL										
r				rh	rh										

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			UV	OV	RESULT										
r			rh	rh	rh										

Field	Bits	Type	Description
RESULT	10:0	rh	**Primary monitor ADC conversion result** This bit field contains the last conversion result of the ADC measurement of the primary voltage monitor. The LSB is documented for each supply rail in the user manual (primary voltage monitors section).
OV	11	rh	**Primary monitor over-voltage event flag** This bit is set if the primary voltage monitor recognizes an over-voltage event. An alarm is raised to the CSRM and SMU. 0_B No over-voltage condition happened. 1_B Over-voltage event indication as configured in the respective VMONP_VxxCON register.
UV	12	rh	**Primary monitor under-voltage event flag** This bit is set if the primary voltage monitor recognizes an under-voltage event. An alarm is raised to the CSRM and SMU. 0_B No under-voltage condition happened. 1_B Under-voltage event indication as configured in the respective VMONP_VxxCON register.
RESETVAL	26:16	rh	**Supply reset value status** This bit field indicates the actual cold PORST reset trim setpoint used by the primary voltage monitors.
RESETOFF	27	rh	**Reset enable status** 0_B A cold PORST is triggered in case of primary under-voltage event 1_B No cold PORST is generated in case of a primary under-voltage event.
0	15:13, 31:28	r	**Reserved**

Table 1550 Reset values of PMS_VMONP_VDDPADSTAT

Reset type	Reset value	Note
LVDPMS2 Reset	$0580\ 0000_H$	
Cold PORST	$0580\ 0000_H$	

36.8.4.42 VDDLMU0 primary voltage monitor configuration register

PMS_VMONP_VDDLMU0CON

VDDLMU0 primary voltage monitor configuration register

Offset address: $90E4_H$

Reset values see: Table 1551

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0			OVENABLE	OVVAL											FILTER
r			rw	rw											rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FILTER				UVENABLE	UVVAL										
rw				rw	rw										

Field	Bits	Type	Description
UVVAL	10:0	rw	**Primary monitor alarm under-voltage threshold** This field defines the under-voltage monitoring threshold level. The LSB is documented for each supply rail in the user manual (primary voltage monitors section).
UVENABLE	11	rw	**Primary monitor under-voltage alarm enable** 0_B No alarm trigger signal is generated. 1_B An alarm trigger signal is generated and forwarded depending on the configured threshold value.
FILTER	16:12	rw	**Primary monitor alarm spike filter** 00_H Each conversion result is compared with threshold to generate alarm $1F_H$ A spike filter of consecutive 32 ADC results are used to generate alarm.
OVVAL	27:17	rw	**Primary monitor alarm over-voltage threshold** This field defines the over-voltage monitoring threshold level. The LSB is documented for each supply rail in the user manual (primary voltage monitors section).
OVENABLE	28	rw	**Primary monitor over-voltage alarm enable** 0_B No alarm trigger signal is generated. 1_B An alarm trigger signal is generated and forwarded depending on the configured threshold value.
0	31:29	r	**Reserved**

Table 1551 Reset values of PMS_VMONP_VDDLMU0CON

Reset type	Reset value	Note
LVDPMS2 Reset	$0F58\ 45DD_H$	
Cold PORST	$0F58\ 45DD_H$	

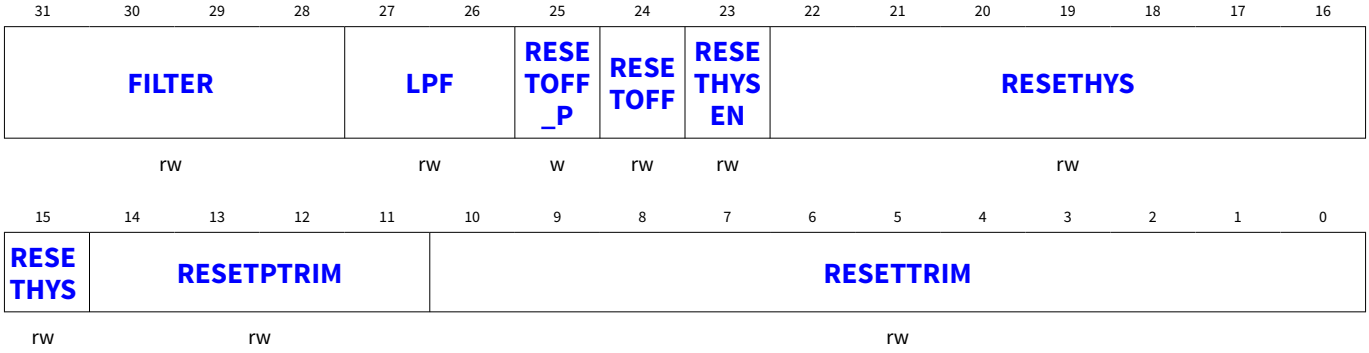
36.8.4.43 VDDLMU0 primary voltage monitor reset configuration register

PMS_VMONP_VDDLMU0RST

VDDLMU0 primary voltage monitor reset configuration register

Offset address: $90E8_H$

Reset values see: Table 1553



Field	Bits	Type	Description
RESETTRIM	10:0	rw	**Supply reset trim value** This bit field selects the hard reset generation level of the respective supply rail. This bitfield is trimmed by Firmware. The LSB is documented for each supply rail in the user manual (primary voltage monitors section).
RESETPTRIM	14:11	rw	**Signed reset trim value** This bit field allows fine trimming of the reset threshold. Binary signed two's complement representation.
RESETHYS	22:15	rw	**Supply reset hysteresis trim value** This bit field selects the reset hysteresis offset level of the primary voltage monitor. The LSB is documented for each supply rail in the user manual (primary voltage monitors section).
RESETHYSEN	23	rw	**Reset release hysteresis enable** 0_B Hysteresis for reset release is not active. 1_B Hysteresis for reset release is active.
RESETOFF	24	rw	**Reset enable** This bit can only be changed if bit RESETOFF_P is set in parallel. RESETOFF is intended to be used only for internal test purposes and the primary reset generation is not to be disabled in customer application. 0_B A reset trigger signal is generated and forwarded to the SCU by the primary monitor depending on the selected reset trim value. 1_B No reset trigger signal is generated and forwarded to the SCU by the primary monitor.
RESETOFF_P	25	w	**RESETOFF bit protection** Writing 1 into this bit enables that bit RESETOFF can be changed in this write operation. This bit is read as zero.

(table continues...)

(continued)

Field	Bits	Type	Description
LPF	27:26	rw	**Primary monitor reset LPF coefficient** This bitfield configures the coefficient of the low pass filter of the primary monitor tracking ADC counter value. The low pass filter is used for cold PORST generation and VMONP_VxxSTAT.RESULT value. y [k] = { y [k-1] * (1-a) } + { x [k] * a} ; y [k] is filter output; x [k] is ADC output a = {1 / (2 ^ LPF)}.If LPF = 0, the filter output is the same as ADC output.
FILTER	31:28	rw	**Primary monitor reset spike filter** 0_H Each conversion result is compared with threshold to generate alarm F_H A spike filter of consecutive 16 ADC results are used to generate reset.

Table 1552 Access mode restrictions of PMS_VMONP_VDDLMU0RST sorted by descending priority

Mode name	Access mode		Description
write 1 to .**RESETOFF_P**	rw	RESETOFF	
(default)	r	RESETOFF	

Table 1553 Reset values of PMS_VMONP_VDDLMU0RST

Reset type	Reset value	Note
LVDPMS2 Reset	2D8F 0580$_H$	
Cold PORST	2D8F 0580$_H$	
After Boot-FW Value	0010 1101 1000 1111 0––– –101 1100 1101$_B$	

36.8.4.44 VDDLMU0 primary voltage monitor status register

PMS_VMONP_VDDLMU0STAT Offset address: 90EC$_H$

VDDLMU0 primary voltage monitor status register Reset values see: Table 1554

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0				RESETOFF	RESETVAL										
r				rh	rh										

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			UV	OV	RESULT										
r			rh	rh	rh										

Field	Bits	Type	Description
RESULT	10:0	rh	**Primary monitor ADC conversion result** This bit field contains the last conversion result of the ADC measurement of the primary voltage monitor. The LSB is documented for each supply rail in the user manual (primary voltage monitors section).
OV	11	rh	**Primary monitor over-voltage event flag** This bit is set if the primary voltage monitor recognizes an over-voltage event. An alarm is raised to the CSRM and SMU. 0_B No over-voltage condition happened. 1_B Over-voltage event indication as configured in the respective VMONP_VxxCON register.
UV	12	rh	**Primary monitor under-voltage event flag** This bit is set if the primary voltage monitor recognizes an under-voltage event. An alarm is raised to the CSRM and SMU. 0_B No under-voltage condition happened. 1_B Under-voltage event indication as configured in the respective VMONP_VxxCON register.
RESETVAL	26:16	rh	**Supply reset value status** This bit field indicates the actual cold PORST reset trim setpoint used by the primary voltage monitors.
RESETOFF	27	rh	**Reset enable status** 0_B A cold PORST is triggered in case of primary under-voltage event 1_B No cold PORST is generated in case of a primary under-voltage event.
0	15:13, 31:28	r	**Reserved**

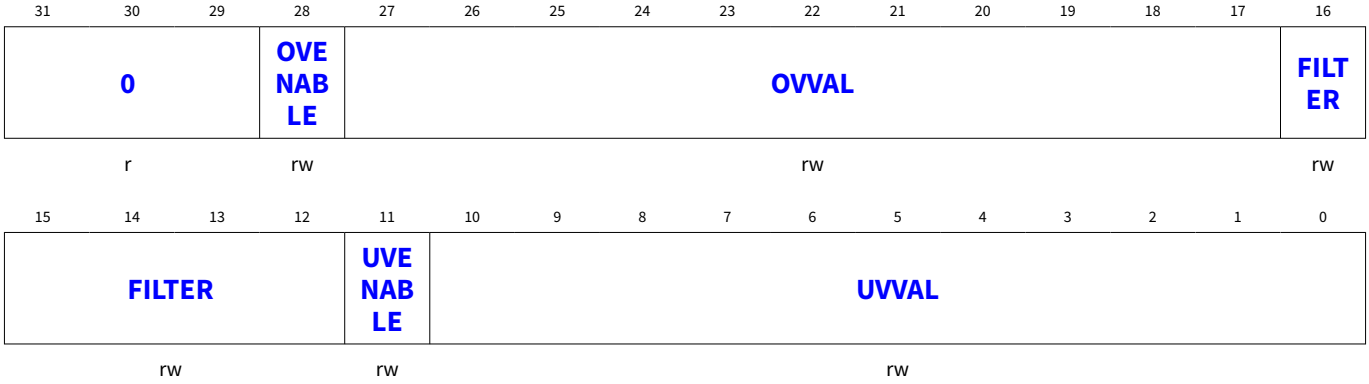
Table 1554 Reset values of PMS_VMONP_VDDLMU0STAT

Reset type	Reset value	Note
LVDPMS2 Reset	0D80 0000$_H$	
Cold PORST	0D80 0000$_H$	

36.8.4.45 VDDLMU1 primary voltage monitor configuration register

PMS_VMONP_VDDLMU1CON Offset address: 90F0$_H$

VDDLMU1 primary voltage monitor configuration register Reset values see: Table 1555



Field	Bits	Type	Description
UVVAL	10:0	rw	**Primary monitor alarm under-voltage threshold** This field defines the under-voltage monitoring threshold level. The LSB is documented for each supply rail in the user manual (primary voltage monitors section).
UVENABLE	11	rw	**Primary monitor under-voltage alarm enable** 0_B No alarm trigger signal is generated. 1_B An alarm trigger signal is generated and forwarded depending on the configured threshold value.
FILTER	16:12	rw	**Primary monitor alarm spike filter** 00_H Each conversion result is compared with threshold to generate alarm $1F_H$ A spike filter of consecutive 32 ADC results are used to generate alarm.
OVVAL	27:17	rw	**Primary monitor alarm over-voltage threshold** This field defines the over-voltage monitoring threshold level. The LSB is documented for each supply rail in the user manual (primary voltage monitors section).
OVENABLE	28	rw	**Primary monitor over-voltage alarm enable** 0_B No alarm trigger signal is generated. 1_B An alarm trigger signal is generated and forwarded depending on the configured threshold value.
0	31:29	r	**Reserved**

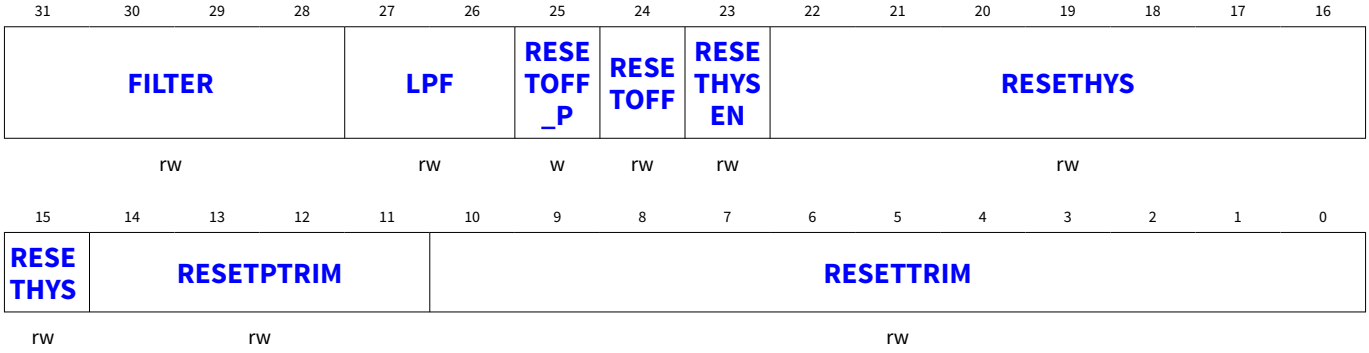
Table 1555 Reset values of PMS_VMONP_VDDLMU1CON

Reset type	Reset value	Note
LVDPMS2 Reset	0F58 45DD$_H$	
Cold PORST	0F58 45DD$_H$	

36.8.4.46 VDDLMU1 primary voltage monitor reset configuration register

PMS_VMONP_VDDLMU1RST Offset address: 90F4$_H$

VDDLMU1 primary voltage monitor reset configuration register Reset values see: Table 1557



Field	Bits	Type	Description
RESETTRIM	10:0	rw	**Supply reset trim value** This bit field selects the hard reset generation level of the respective supply rail. This bitfield is trimmed by Firmware. The LSB is documented for each supply rail in the user manual (primary voltage monitors section).
RESETPTRIM	14:11	rw	**Signed reset trim value** This bit field allows fine trimming of the reset threshold. Binary signed two's complement representation.
RESETHYS	22:15	rw	**Supply reset hysteresis trim value** This bit field selects the reset hysteresis offset level of the primary voltage monitor. The LSB is documented for each supply rail in the user manual (primary voltage monitors section).
RESETHYSEN	23	rw	**Reset release hysteresis enable** 0_B Hysteresis for reset release is not active. 1_B Hysteresis for reset release is active.
RESETOFF	24	rw	**Reset enable** This bit can only be changed if bit RESETOFF_P is set in parallel. RESETOFF is intended to be used only for internal test purposes and the primary reset generation is not to be disabled in customer application. 0_B A reset trigger signal is generated and forwarded to the SCU by the primary monitor depending on the selected reset trim value. 1_B No reset trigger signal is generated and forwarded to the SCU by the primary monitor.
RESETOFF_P	25	w	**RESETOFF bit protection** Writing 1 into this bit enables that bit RESETOFF can be changed in this write operation. This bit is read as zero.
LPF	27:26	rw	**Primary monitor reset LPF coefficient** This bitfield configures the coefficient of the low pass filter of the primary monitor tracking ADC counter value. The low pass filter is used for cold PORST generation and VMONP_VxxSTAT.RESULT value. y [k] = { y [k-1] * (1-a) } + { x [k] * a} ; y [k] is filter output; x [k] is ADC output a = {1 / (2 ^ LPF)}.If LPF = 0, the filter output is the same as ADC output.

(table continues...)

(continued)

Field	Bits	Type	Description
FILTER	31:28	rw	**Primary monitor reset spike filter** 0_H Each conversion result is compared with threshold to generate alarm F_H A spike filter of consecutive 16 ADC results are used to generate reset.

Table 1556 Access mode restrictions of PMS_VMONP_VDDLMU1RST sorted by descending priority

Mode name	Access mode		Description
write 1 to .**RESETOFF_P**	rw	RESETOFF	
(default)	r	RESETOFF	

Table 1557 Reset values of PMS_VMONP_VDDLMU1RST

Reset type	Reset value	Note
LVDPMS2 Reset	2D8F 0580$_H$	
Cold PORST	2D8F 0580$_H$	
After Boot-FW Value	0010 1101 1000 1111 0––– –101 1100 1101$_B$	

36.8.4.47 VDDLMU1 primary voltage monitor status register

PMS_VMONP_VDDLMU1STAT Offset address: 90F8$_H$

VDDLMU1 primary voltage monitor status register Reset values see: Table 1558

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0				RESETOFF	RESETVAL										
r				rh	rh										

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			UV	OV	RESULT										
r			rh	rh	rh										

Field	Bits	Type	Description
RESULT	10:0	rh	**Primary monitor ADC conversion result** This bit field contains the last conversion result of the ADC measurement of the primary voltage monitor. The LSB is documented for each supply rail in the user manual (primary voltage monitors section).

(table continues...)

(continued)

Field	Bits	Type	Description
OV	11	rh	**Primary monitor over-voltage event flag** This bit is set if the primary voltage monitor recognizes an over-voltage event. An alarm is raised to the CSRM and SMU. 0_B No over-voltage condition happened. 1_B Over-voltage event indication as configured in the respective VMONP_VxxCON register.
UV	12	rh	**Primary monitor under-voltage event flag** This bit is set if the primary voltage monitor recognizes an under-voltage event. An alarm is raised to the CSRM and SMU. 0_B No under-voltage condition happened. 1_B Under-voltage event indication as configured in the respective VMONP_VxxCON register.
RESETVAL	26:16	rh	**Supply reset value status** This bit field indicates the actual cold PORST reset trim setpoint used by the primary voltage monitors.
RESETOFF	27	rh	**Reset enable status** 0_B A cold PORST is triggered in case of primary under-voltage event 1_B No cold PORST is generated in case of a primary under-voltage event.
0	15:13, 31:28	r	**Reserved**

Table 1558 Reset values of PMS_VMONP_VDDLMU1STAT

Reset type	Reset value	Note
LVDPMS2 Reset	0D80 0000$_H$	
Cold PORST	0D80 0000$_H$	

36.8.4.48 VDDPPU primary voltage monitor configuration register

PMS_VMONP_VDDPPUCON Offset address: 90FC$_H$

VDDPPU primary voltage monitor configuration register Reset values see: Table 1559

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0			OVENABLE	OVVAL											FILTER
r			rw	rw											rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FILTER				UVENABLE	UVVAL										
rw				rw	rw										

Field	Bits	Type	Description
UVVAL	10:0	rw	**Primary monitor alarm under-voltage threshold** This field defines the under-voltage monitoring threshold level. The LSB is documented for each supply rail in the user manual (primary voltage monitors section).
UVENABLE	11	rw	**Primary monitor under-voltage alarm enable** 0_B No alarm trigger signal is generated. 1_B An alarm trigger signal is generated and forwarded depending on the configured threshold value.
FILTER	16:12	rw	**Primary monitor alarm spike filter** 00_H Each conversion result is compared with threshold to generate alarm $1F_H$ A spike filter of consecutive 32 ADC results are used to generate alarm.
OVVAL	27:17	rw	**Primary monitor alarm over-voltage threshold** This field defines the over-voltage monitoring threshold level. The LSB is documented for each supply rail in the user manual (primary voltage monitors section).
OVENABLE	28	rw	**Primary monitor over-voltage alarm enable** 0_B No alarm trigger signal is generated. 1_B An alarm trigger signal is generated and forwarded depending on the configured threshold value.
0	31:29	r	**Reserved**

Table 1559 Reset values of PMS_VMONP_VDDPPUCON

Reset type	Reset value	Note
LVDPMS2 Reset	0F58 45DD$_H$	
Cold PORST	0F58 45DD$_H$	

36.8.4.49 VDDPPU primary voltage monitor reset configuration register

PMS_VMONP_VDDPPURST

Offset address: 9100_H

VDDPPU primary voltage monitor reset configuration register

Reset values see: Table 1561

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FILTER				LPF		RESETOFF_P	RESETOFF	RESETHYSEN	RESETHYS						
rw				rw		w	rw	rw	rw						

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESETHYS	RESETPTRIM							RESETTRIM							
rw	rw							rw							

Field	Bits	Type	Description
RESETTRIM	10:0	rw	**Supply reset trim value** This bit field selects the hard reset generation level of the respective supply rail. This bitfield is trimmed by Firmware. The LSB is documented for each supply rail in the user manual (primary voltage monitors section).
RESETPTRIM	14:11	rw	**Signed reset trim value** This bit field allows fine trimming of the reset threshold. Binary signed two's complement representation.
RESETHYS	22:15	rw	**Supply reset hysteresis trim value** This bit field selects the reset hysteresis offset level of the primary voltage monitor. The LSB is documented for each supply rail in the user manual (primary voltage monitors section).
RESETHYSEN	23	rw	**Reset release hysteresis enable** 0_B Hysteresis for reset release is not active. 1_B Hysteresis for reset release is active.
RESETOFF	24	rw	**Reset enable** This bit can only be changed if bit RESETOFF_P is set in parallel. RESETOFF is intended to be used only for internal test purposes and the primary reset generation is not to be disabled in customer application. 0_B A reset trigger signal is generated and forwarded to the SCU by the primary monitor depending on the selected reset trim value. 1_B No reset trigger signal is generated and forwarded to the SCU by the primary monitor.
RESETOFF_P	25	w	**RESETOFF bit protection** Writing 1 into this bit enables that bit RESETOFF can be changed in this write operation. This bit is read as zero.
LPF	27:26	rw	**Primary monitor reset LPF coefficient** This bitfield configures the coefficient of the low pass filter of the primary monitor tracking ADC counter value. The low pass filter is used for cold PORST generation and VMONP_VxxSTAT.RESULT value. y [k] = { y [k-1] * (1-a) } + { x [k] * a} ; y [k] is filter output; x [k] is ADC output a = {1 / (2 ^ LPF)}.If LPF = 0, the filter output is the same as ADC output.
FILTER	31:28	rw	**Primary monitor reset spike filter** 0_H Each conversion result is compared with threshold to generate alarm F_H A spike filter of consecutive 16 ADC results are used to generate reset.

Table 1560 Access mode restrictions of PMS_VMONP_VDDPPURST sorted by descending priority

Mode name	Access mode		Description
write 1 to .**RESETOFF_P**	rw	RESETOFF	

(table continues...)

Table 1560 (continued) Access mode restrictions of PMS_VMONP_VDDPPURST sorted by descending priority

Mode name	Access mode		Description
(default)	r	RESETOFF	

Table 1561 Reset values of PMS_VMONP_VDDPPURST

Reset type	Reset value	Note
LVDPMS2 Reset	2D8F 0580$_H$	
Cold PORST	2D8F 0580$_H$	
After Boot-FW Value	0010 1101 1000 1111 0––– –101 1100 1101$_B$	

36.8.4.50 VDDPPU primary voltage monitor status register

PMS_VMONP_VDDPPUSTAT Offset address: 9104$_H$

VDDPPU primary voltage monitor status register Reset values see: Table 1562

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0				RESETOFF	RESETVAL										
r				rh	rh										

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			UV	OV	RESULT										
r			rh	rh	rh										

Field	Bits	Type	Description
RESULT	10:0	rh	**Primary monitor ADC conversion result** This bit field contains the last conversion result of the ADC measurement of the primary voltage monitor. The LSB is documented for each supply rail in the user manual (primary voltage monitors section).
OV	11	rh	**Primary monitor over-voltage event flag** This bit is set if the primary voltage monitor recognizes an over-voltage event. An alarm is raised to the CSRM and SMU. 0_B No over-voltage condition happened. 1_B Over-voltage event indication as configured in the respective VMONP_VxxCON register.
UV	12	rh	**Primary monitor under-voltage event flag** This bit is set if the primary voltage monitor recognizes an under-voltage event. An alarm is raised to the CSRM and SMU. 0_B No under-voltage condition happened. 1_B Under-voltage event indication as configured in the respective VMONP_VxxCON register.

(table continues...)

(continued)

Field	Bits	Type	Description
RESETVAL	26:16	rh	**Supply reset value status** This bit field indicates the actual cold PORST reset trim setpoint used by the primary voltage monitors.
RESETOFF	27	rh	**Reset enable status** 0_B A cold PORST is triggered in case of primary under-voltage event 1_B No cold PORST is generated in case of a primary under-voltage event.
0	15:13, 31:28	r	**Reserved**

Table 1562 Reset values of PMS_VMONP_VDDPPUSTAT

Reset type	Reset value	Note
LVDPMS2 Reset	0D80 0000$_H$	
Cold PORST	0D80 0000$_H$	

36.8.4.51 VDDPHY0 primary voltage monitor configuration register

PMS_VMONP_VDDPHY0CON

VDDPHY0 primary voltage monitor configuration register

Offset address: 9114$_H$

Reset values see: Table 1563

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0			OVENABLE	OVVAL											FILTER
r			rw	rw											rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FILTER				UVENABLE	UVVAL										
rw				rw	rw										

Field	Bits	Type	Description
UVVAL	10:0	rw	**Primary monitor alarm under-voltage threshold** This field defines the under-voltage monitoring threshold level. The LSB is documented for each supply rail in the user manual (primary voltage monitors section).
UVENABLE	11	rw	**Primary monitor under-voltage alarm enable** 0_B No alarm trigger signal is generated. 1_B An alarm trigger signal is generated and forwarded depending on the configured threshold value.

(table continues...)

(continued)

Field	Bits	Type	Description
FILTER	16:12	rw	**Primary monitor alarm spike filter** 00_H Each conversion result is compared with threshold to generate alarm $1F_H$ A spike filter of consecutive 32 ADC results are used to generate alarm.
OVVAL	27:17	rw	**Primary monitor alarm over-voltage threshold** This field defines the over-voltage monitoring threshold level. The LSB is documented for each supply rail in the user manual (primary voltage monitors section).
OVENABLE	28	rw	**Primary monitor over-voltage alarm enable** 0_B No alarm trigger signal is generated. 1_B An alarm trigger signal is generated and forwarded depending on the configured threshold value.
0	31:29	r	**Reserved**

Table 1563 Reset values of PMS_VMONP_VDDPHY0CON

Reset type	Reset value	Note
LVDPMS2 Reset	0F10 45DD$_H$	
Cold PORST	0F10 45DD$_H$	

36.8.4.52 VDDPHY0 primary voltage monitor reset configuration register

PMS_VMONP_VDDPHY0RST Offset address: 9118_H

VDDPHY0 primary voltage monitor reset configuration register Reset values see: Table 1565

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FILTER				LPF		RESETOFF_P	RESETOFF	RESETHYSEN	RESETHYS						
rw				rw		w	rw	rw	rw						

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESETHYS	RESETPTRIM							RESETTRIM							
rw	rw							rw							

Field	Bits	Type	Description
RESETTRIM	10:0	rw	**Supply reset trim value** This bit field selects the hard reset generation level of the respective supply rail. This bitfield is trimmed by Firmware. The LSB is documented for each supply rail in the user manual (primary voltage monitors section).

(table continues...)

(continued)

Field	Bits	Type	Description
RESETPTRIM	14:11	rw	**Signed reset trim value** This bit field allows fine trimming of the reset threshold. Binary signed two's complement representation.
RESETHYS	22:15	rw	**Supply reset hysteresis trim value** This bit field selects the reset hysteresis offset level of the primary voltage monitor. The LSB is documented for each supply rail in the user manual (primary voltage monitors section).
RESETHYSEN	23	rw	**Reset release hysteresis enable** 0_B Hysteresis for reset release is not active. 1_B Hysteresis for reset release is active.
RESETOFF	24	rw	**Reset enable** This bit can only be changed if bit RESETOFF_P is set in parallel. RESETOFF is intended to be used only for internal test purposes and the primary reset generation is not to be disabled in customer application. 0_B A reset trigger signal is generated and forwarded to the SCU by the primary monitor depending on the selected reset trim value. 1_B No reset trigger signal is generated and forwarded to the SCU by the primary monitor.
RESETOFF_P	25	w	**RESETOFF bit protection** Writing 1 into this bit enables that bit RESETOFF can be changed in this write operation. This bit is read as zero.
LPF	27:26	rw	**Primary monitor reset LPF coefficient** This bitfield configures the coefficient of the low pass filter of the primary monitor tracking ADC counter value. The low pass filter is used for cold PORST generation and VMONP_VxxSTAT.RESULT value. y [k] = { y [k-1] * (1-a) } + { x [k] * a} ; y [k] is filter output; x [k] is ADC output a = {1 / (2 ^ LPF)}.If LPF = 0, the filter output is the same as ADC output.
FILTER	31:28	rw	**Primary monitor reset spike filter** 0_H Each conversion result is compared with threshold to generate alarm F_H A spike filter of consecutive 16 ADC results are used to generate reset.

Table 1564 Access mode restrictions of PMS_VMONP_VDDPHY0RST sorted by descending priority

Mode name	Access mode		Description
write 1 to .**RESETOFF_P**	rw	RESETOFF	
(default)	r	RESETOFF	

Table 1565 Reset values of PMS_VMONP_VDDPHY0RST

Reset type	Reset value	Note
LVDPMS2 Reset	$2D8F\ 0580_H$	
Cold PORST	$2D8F\ 0580_H$	
After Boot-FW Value	$0010\ 1101\ 1000\ 1111$ $0\text{---}\ \text{–}101\ 1100\ 1101_B$	

36.8.4.53 VDDPHY0 primary voltage monitor status register

PMS_VMONP_VDDPHY0STAT Offset address: $911C_H$

VDDPHY0 primary voltage monitor status register Reset values see: Table 1566

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0				RESETOFF	RESETVAL										
r				rh	rh										

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			UV	OV	RESULT										
r			rh	rh	rh										

Field	Bits	Type	Description
RESULT	10:0	rh	**Primary monitor ADC conversion result** This bit field contains the last conversion result of the ADC measurement of the primary voltage monitor. The LSB is documented for each supply rail in the user manual (primary voltage monitors section).
OV	11	rh	**Primary monitor over-voltage event flag** This bit is set if the primary voltage monitor recognizes an over-voltage event. An alarm is raised to the CSRM and SMU. 0_B No over-voltage condition happened. 1_B Over-voltage event indication as configured in the respective VMONP_VxxCON register.
UV	12	rh	**Primary monitor under-voltage event flag** This bit is set if the primary voltage monitor recognizes an under-voltage event. An alarm is raised to the CSRM and SMU. 0_B No under-voltage condition happened. 1_B Under-voltage event indication as configured in the respective VMONP_VxxCON register.
RESETVAL	26:16	rh	**Supply reset value status** This bit field indicates the actual cold PORST reset trim setpoint used by the primary voltage monitors.

(table continues...)

(continued)

Field	Bits	Type	Description
RESETOFF	27	rh	**Reset enable status** 0_B A cold PORST is triggered in case of primary under-voltage event 1_B No cold PORST is generated in case of a primary under-voltage event.
0	15:13, 31:28	r	**Reserved**

Table 1566 Reset values of PMS_VMONP_VDDPHY0STAT

Reset type	Reset value	Note
LVDPMS2 Reset	0D80 0000$_H$	
Cold PORST	0D80 0000$_H$	

36.8.4.54 VDDPHY1 primary voltage monitor configuration register

PMS_VMONP_VDDPHY1CON Offset address: 9120$_H$

VDDPHY1 primary voltage monitor configuration register Reset values see: Table 1567

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0			OVENABLE	OVVAL											FILTER
r			rw	rw											rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FILTER				UVENABLE	UVVAL										
rw				rw	rw										

Field	Bits	Type	Description
UVVAL	10:0	rw	**Primary monitor alarm under-voltage threshold** This field defines the under-voltage monitoring threshold level. The LSB is documented for each supply rail in the user manual (primary voltage monitors section).
UVENABLE	11	rw	**Primary monitor under-voltage alarm enable** 0_B No alarm trigger signal is generated. 1_B An alarm trigger signal is generated and forwarded depending on the configured threshold value.
FILTER	16:12	rw	**Primary monitor alarm spike filter** 00_H Each conversion result is compared with threshold to generate alarm $1F_H$ A spike filter of consecutive 32 ADC results are used to generate alarm.

(table continues...)

(continued)

Field	Bits	Type	Description
OVVAL	27:17	rw	**Primary monitor alarm over-voltage threshold** This field defines the over-voltage monitoring threshold level. The LSB is documented for each supply rail in the user manual (primary voltage monitors section).
OVENABLE	28	rw	**Primary monitor over-voltage alarm enable** 0_B No alarm trigger signal is generated. 1_B An alarm trigger signal is generated and forwarded depending on the configured threshold value.
0	31:29	r	**Reserved**

Table 1567 Reset values of PMS_VMONP_VDDPHY1CON

Reset type	Reset value	Note
LVDPMS2 Reset	0F10 45DD$_H$	
Cold PORST	0F10 45DD$_H$	

36.8.4.55 VDDPHY1 primary voltage monitor reset configuration register

PMS_VMONP_VDDPHY1RST

Offset address: 9124$_H$

VDDPHY1 primary voltage monitor reset configuration register

Reset values see: Table 1569

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FILTER				LPF		RESETOFF_P	RESETOFF	RESETHYSEN	RESETHYS						
rw				rw		w	rw	rw	rw						

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESETHYS	RESETPTRIM							RESETTRIM							
rw	rw							rw							

Field	Bits	Type	Description
RESETTRIM	10:0	rw	**Supply reset trim value** This bit field selects the hard reset generation level of the respective supply rail. This bitfield is trimmed by Firmware. The LSB is documented for each supply rail in the user manual (primary voltage monitors section).
RESETPTRIM	14:11	rw	**Signed reset trim value** This bit field allows fine trimming of the reset threshold. Binary signed two's complement representation.

(table continues...)

(continued)

Field	Bits	Type	Description
RESETHYS	22:15	rw	**Supply reset hysteresis trim value** This bit field selects the reset hysteresis offset level of the primary voltage monitor. The LSB is documented for each supply rail in the user manual (primary voltage monitors section).
RESETHYSEN	23	rw	**Reset release hysteresis enable** 0_B Hysteresis for reset release is not active. 1_B Hysteresis for reset release is active.
RESETOFF	24	rw	**Reset enable** This bit can only be changed if bit RESETOFF_P is set in parallel. RESETOFF is intended to be used only for internal test purposes and the primary reset generation is not to be disabled in customer application. 0_B A reset trigger signal is generated and forwarded to the SCU by the primary monitor depending on the selected reset trim value. 1_B No reset trigger signal is generated and forwarded to the SCU by the primary monitor.
RESETOFF_P	25	w	**RESETOFF bit protection** Writing 1 into this bit enables that bit RESETOFF can be changed in this write operation. This bit is read as zero.
LPF	27:26	rw	**Primary monitor reset LPF coefficient** This bitfield configures the coefficient of the low pass filter of the primary monitor tracking ADC counter value. The low pass filter is used for cold PORST generation and VMONP_VxxSTAT.RESULT value. y [k] = { y [k-1] * (1-a) } + { x [k] * a} ; y [k] is filter output; x [k] is ADC output a = {1 / (2 ^ LPF)}.If LPF = 0, the filter output is the same as ADC output.
FILTER	31:28	rw	**Primary monitor reset spike filter** 0_H Each conversion result is compared with threshold to generate alarm F_H A spike filter of consecutive 16 ADC results are used to generate reset.

Table 1568 Access mode restrictions of PMS_VMONP_VDDPHY1RST sorted by descending priority

Mode name	Access mode		Description
write 1 to .**RESETOFF_P**	rw	RESETOFF	
(default)	r	RESETOFF	

Table 1569 Reset values of PMS_VMONP_VDDPHY1RST

Reset type	Reset value	Note
LVDPMS2 Reset	2D8F 0580$_H$	
Cold PORST	2D8F 0580$_H$	
After Boot-FW Value	0010 1101 1000 1111 0––– –101 1100 1101$_B$	

36.8.4.56 VDDPHY1 primary voltage monitor status register

PMS_VMONP_VDDPHY1STAT Offset address: 9128_H

VDDPHY1 primary voltage monitor status register Reset values see: Table 1570

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0				RESETOFF	RESETVAL										
r				rh	rh										

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			UV	OV	RESULT										
r			rh	rh	rh										

Field	Bits	Type	Description
RESULT	10:0	rh	**Primary monitor ADC conversion result** This bit field contains the last conversion result of the ADC measurement of the primary voltage monitor. The LSB is documented for each supply rail in the user manual (primary voltage monitors section).
OV	11	rh	**Primary monitor over-voltage event flag** This bit is set if the primary voltage monitor recognizes an over-voltage event. An alarm is raised to the CSRM and SMU. 0_B No over-voltage condition happened. 1_B Over-voltage event indication as configured in the respective VMONP_VxxCON register.
UV	12	rh	**Primary monitor under-voltage event flag** This bit is set if the primary voltage monitor recognizes an under-voltage event. An alarm is raised to the CSRM and SMU. 0_B No under-voltage condition happened. 1_B Under-voltage event indication as configured in the respective VMONP_VxxCON register.
RESETVAL	26:16	rh	**Supply reset value status** This bit field indicates the actual cold PORST reset trim setpoint used by the primary voltage monitors.
RESETOFF	27	rh	**Reset enable status** 0_B A cold PORST is triggered in case of primary under-voltage event 1_B No cold PORST is generated in case of a primary under-voltage event.
0	15:13, 31:28	r	**Reserved**

Table 1570 Reset values of PMS_VMONP_VDDPHY1STAT

Reset type	Reset value	Note
LVDPMS2 Reset	$0D80\ 0000_H$	
Cold PORST	$0D80\ 0000_H$	

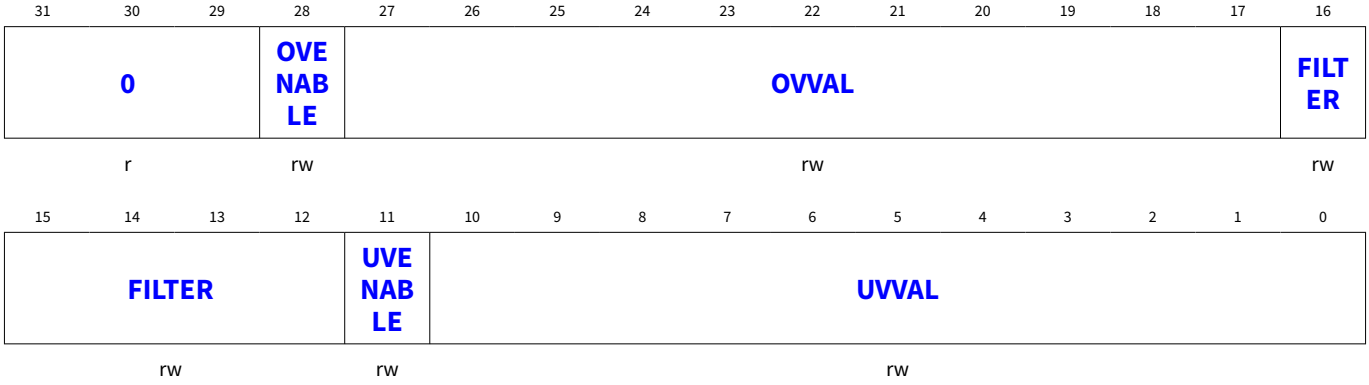
36.8.4.57 VDDPHY2 primary voltage monitor configuration register

PMS_VMONP_VDDPHY2CON

VDDPHY2 primary voltage monitor configuration register

Offset address: $912C_H$

Reset values see: Table 1571



Field	Bits	Type	Description
UVVAL	10:0	rw	**Primary monitor alarm under-voltage threshold** This field defines the under-voltage monitoring threshold level. The LSB is documented for each supply rail in the user manual (primary voltage monitors section).
UVENABLE	11	rw	**Primary monitor under-voltage alarm enable** 0_B No alarm trigger signal is generated. 1_B An alarm trigger signal is generated and forwarded depending on the configured threshold value.
FILTER	16:12	rw	**Primary monitor alarm spike filter** 00_H Each conversion result is compared with threshold to generate alarm $1F_H$ A spike filter of consecutive 32 ADC results are used to generate alarm.
OVVAL	27:17	rw	**Primary monitor alarm over-voltage threshold** This field defines the over-voltage monitoring threshold level. The LSB is documented for each supply rail in the user manual (primary voltage monitors section).
OVENABLE	28	rw	**Primary monitor over-voltage alarm enable** 0_B No alarm trigger signal is generated. 1_B An alarm trigger signal is generated and forwarded depending on the configured threshold value.
0	31:29	r	**Reserved**

Table 1571 Reset values of PMS_VMONP_VDDPHY2CON

Reset type	Reset value	Note
LVDPMS2 Reset	$0F10\ 45DD_H$	
Cold PORST	$0F10\ 45DD_H$	

36.8.4.58 VDDPHY2 primary voltage monitor reset configuration register

PMS_VMONP_VDDPHY2RST

VDDPHY2 primary voltage monitor reset configuration register

Offset address: 9130_H

Reset values see: Table 1573

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FILTER				LPF		RESETOFF_P	RESETOFF	RESETHYSEN	RESETHYS						
rw				rw		w	rw	rw	rw						

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESETHYS	RESETPTRIM								RESETTRIM						
rw	rw								rw						

Field	Bits	Type	Description
RESETTRIM	10:0	rw	**Supply reset trim value** This bit field selects the hard reset generation level of the respective supply rail. This bitfield is trimmed by Firmware. The LSB is documented for each supply rail in the user manual (primary voltage monitors section).
RESETPTRIM	14:11	rw	**Signed reset trim value** This bit field allows fine trimming of the reset threshold. Binary signed two's complement representation.
RESETHYS	22:15	rw	**Supply reset hysteresis trim value** This bit field selects the reset hysteresis offset level of the primary voltage monitor. The LSB is documented for each supply rail in the user manual (primary voltage monitors section).
RESETHYSEN	23	rw	**Reset release hysteresis enable** 0_B Hysteresis for reset release is not active. 1_B Hysteresis for reset release is active.
RESETOFF	24	rw	**Reset enable** This bit can only be changed if bit RESETOFF_P is set in parallel. RESETOFF is intended to be used only for internal test purposes and the primary reset generation is not to be disabled in customer application. 0_B A reset trigger signal is generated and forwarded to the SCU by the primary monitor depending on the selected reset trim value. 1_B No reset trigger signal is generated and forwarded to the SCU by the primary monitor.
RESETOFF_P	25	w	**RESETOFF bit protection** Writing 1 into this bit enables that bit RESETOFF can be changed in this write operation. This bit is read as zero.

(table continues...)

(continued)

Field	Bits	Type	Description
LPF	27:26	rw	**Primary monitor reset LPF coefficient** This bitfield configures the coefficient of the low pass filter of the primary monitor tracking ADC counter value. The low pass filter is used for cold PORST generation and VMONP_VxxSTAT.RESULT value. y [k] = { y [k-1] * (1-a) } + { x [k] * a} ; y [k] is filter output; x [k] is ADC output a = {1 / (2 ^ LPF)}.If LPF = 0, the filter output is the same as ADC output.
FILTER	31:28	rw	**Primary monitor reset spike filter** 0_H Each conversion result is compared with threshold to generate alarm F_H A spike filter of consecutive 16 ADC results are used to generate reset.

Table 1572 Access mode restrictions of PMS_VMONP_VDDPHY2RST sorted by descending priority

Mode name	Access mode		Description
write 1 to .**RESETOFF_P**	rw	RESETOFF	
(default)	r	RESETOFF	

Table 1573 Reset values of PMS_VMONP_VDDPHY2RST

Reset type	Reset value	Note
LVDPMS2 Reset	2D8F 0580$_H$	
Cold PORST	2D8F 0580$_H$	
After Boot-FW Value	0010 1101 1000 1111 0––– –101 1100 1101$_B$	

36.8.4.59 VDDPHY2 primary voltage monitor status register

PMS_VMONP_VDDPHY2STAT Offset address: 9134$_H$

VDDPHY2 primary voltage monitor status register Reset values see: Table 1574

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0				RESETOFF	RESETVAL										
r				rh	rh										

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			UV	OV	RESULT										
r			rh	rh	rh										

Field	Bits	Type	Description
RESULT	10:0	rh	**Primary monitor ADC conversion result** This bit field contains the last conversion result of the ADC measurement of the primary voltage monitor. The LSB is documented for each supply rail in the user manual (primary voltage monitors section).
OV	11	rh	**Primary monitor over-voltage event flag** This bit is set if the primary voltage monitor recognizes an over-voltage event. An alarm is raised to the CSRM and SMU. 0_B No over-voltage condition happened. 1_B Over-voltage event indication as configured in the respective VMONP_VxxCON register.
UV	12	rh	**Primary monitor under-voltage event flag** This bit is set if the primary voltage monitor recognizes an under-voltage event. An alarm is raised to the CSRM and SMU. 0_B No under-voltage condition happened. 1_B Under-voltage event indication as configured in the respective VMONP_VxxCON register.
RESETVAL	26:16	rh	**Supply reset value status** This bit field indicates the actual cold PORST reset trim setpoint used by the primary voltage monitors.
RESETOFF	27	rh	**Reset enable status** 0_B A cold PORST is triggered in case of primary under-voltage event 1_B No cold PORST is generated in case of a primary under-voltage event.
0	15:13, 31:28	r	**Reserved**

Table 1574 Reset values of PMS_VMONP_VDDPHY2STAT

Reset type	Reset value	Note
LVDPMS2 Reset	0D80 0000$_H$	
Cold PORST	0D80 0000$_H$	

36.8.4.60 Tracking ADC trim register 0

PMS_VMONP_TRKTRIM0 Offset address: 9138$_H$

Tracking ADC trim register 0 Reset values see: Table 1575

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				VDDEXTDCVMONSLPF		VDDDCLSALARMLPF		VSSDCHSALARMLPF		VDDEXTDCALARMLPF		VDDEXTALARMLPF		VDDALARMLPF	
r				rw		rw		rw		rw		rw		rw	

Field	Bits	Type	Description
VDDALARMLPF	1:0	rw	**VDD supply primary monitor alarm LPF coefficient** This bitfield configures the coefficient of the Low Pass Filter of the Primary Monitor tracking ADC counter value for SMU & CSRM alarms. y [k] = { y [k-1] * (1-a) } + { x [k] * a} ; y [k] is filter output; x [k] is ADC output a = {1 / (2 ^ LPF)}.If LPF = 0, the filter output is the same as ADC output. 00 : 2 clock delay 01 : 3 clock delay 10 : 5 clock delay 11 : 9 clock delay
VDDEXTALARMLPF	3:2	rw	**VDDEXT supply primary monitor alarm LPF coefficient** This bitfield configures the coefficient of the Low Pass Filter of the Primary Monitor tracking ADC counter value for SMU & CSRM alarms. y [k] = { y [k-1] * (1-a) } + { x [k] * a} ; y [k] is filter output; x [k] is ADC output a = {1 / (2 ^ LPF)}.If LPF = 0, the filter output is the same as ADC output. 00 : 2 clock delay 01 : 3 clock delay 10 : 5 clock delay 11 : 9 clock delay
VDDEXTDCALARMLPF	5:4	rw	**VDDEXTDC supply primary monitor alarm LPF coefficient** This bitfield configures the coefficient of the Low Pass Filter of the Primary Monitor tracking ADC counter value for SMU & CSRM alarms. y [k] = { y [k-1] * (1-a) } + { x [k] * a} ; y [k] is filter output; x [k] is ADC output a = {1 / (2 ^ LPF)}.If LPF = 0, the filter output is the same as ADC output. 00 : 2 clock delay 01 : 3 clock delay 10 : 5 clock delay 11 : 9 clock delay
VSSDCHSALARMLPF	7:6	rw	**VSSDCHS supply primary monitor alarm LPF coefficient** This bitfield configures the coefficient of the Low Pass Filter of the Primary Monitor tracking ADC counter value for SMU & CSRM alarms. y [k] = { y [k-1] * (1-a) } + { x [k] * a} ; y [k] is filter output; x [k] is ADC output a = {1 / (2 ^ LPF)}.If LPF = 0, the filter output is the same as ADC output. 00 : 2 clock delay 01 : 3 clock delay 10 : 5 clock delay 11 : 9 clock delay

(table continues...)

(continued)

Field	Bits	Type	Description
VDDDCLSALARMLPF	9:8	rw	**VDDDCLS supply primary monitor alarm LPF coefficient** This bitfield configures the coefficient of the Low Pass Filter of the Primary Monitor tracking ADC counter value for SMU & CSRM alarms. y [k] = { y [k-1] * (1-a) } + { x [k] * a} ; y [k] is filter output; x [k] is ADC output a = {1 / (2 ^ LPF)}.If LPF = 0, the filter output is the same as ADC output. 00 : 2 clock delay 01 : 3 clock delay 10 : 5 clock delay 11 : 9 clock delay
VDDEXTDCVMONSLPF	11:10	rw	**VDDEXTDC supply secondary monitor alarm LPF coefficient** This bitfield configures the coefficient of the Low Pass Filter of the Primary Monitor tracking ADC counter value for SMU & CSRM alarms. y [k] = { y [k-1] * (1-a) } + { x [k] * a} ; y [k] is filter output; x [k] is ADC output a = {1 / (2 ^ LPF)}.If LPF = 0, the filter output is the same as ADC output. 00 : 2 clock delay 01 : 3 clock delay 10 : 5 clock delay 11 : 9 clock delay
0	31:12	r	**Reserved** Read as 0; should be written with 0.

Table 1575 Reset values of PMS_VMONP_TRKTRIM0

Reset type	Reset value	Note
LVDPMS2 Reset	0000 0FFF$_H$	
Cold PORST	0000 0FFF$_H$	
After Boot-FW Value	0000 0FFF$_H$	

36.8.4.61 VDDEVRSB secondary monitor configuration register

PMS_VMONS_VDDEVRSBCON Offset address: 9144$_H$

VDDEVRSB secondary monitor configuration register Reset values see: Table 1576

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		FILTER				OVMOD		OVVAL							
r		rw				rw		rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OVVAL			UVMOD		UVVAL										
rw			rw		rw										

Field	Bits	Type	Description
UVVAL	10:0	rw	**Secondary monitor under-voltage threshold** This field defines the under-voltage threshold level of the supply monitor. The LSB is documented for each supply rail in the user manual (secondary voltage monitors section).
UVMOD	12:11	rw	**Under-voltage monitoring mode** 00_B Under-voltage monitoring inactive. 01_B Under-voltage alarm enabled. In addition, if the corresponding INT_CONx.UVVxx bit is enabled, an under-voltage interrupt is triggered as soon as the corresponding VMONS.VxxSTAT.UV status bit is set. 10_B Under-voltage alarm enabled. In addition, if the corresponding INT_CONx.UVVxx bit is enabled, an under-voltage interrupt is triggered as soon as the under-voltage alarm is de-asserted. 11_B Under-voltage alarm enabled. In addition, if the corresponding INT_CONx.UVVxx bit is enabled, an under-voltage interrupt is triggered as soon as either the corresponding VMONS.VxxSTAT.UV status bit is set or the under-voltage alarm is de-asserted.
OVVAL	23:13	rw	**Secondary monitor over-voltage threshold** This field defines the over-voltage threshold level of the supply monitor. The LSB is documented for each supply rail in the user manual (secondary voltage monitors section).
OVMOD	25:24	rw	**Over-voltage monitoring mode** 00_B Over-voltage monitoring inactive. 01_B Over-voltage alarm enabled. In addition, if the corresponding INT_CONx.OVVxx bit is enabled, an over-voltage interrupt is triggered as soon as the corresponding VMONS.VxxSTAT.OV status bit is set. 10_B Over-voltage alarm enabled. In addition, if the corresponding INT_CONx.OVVxx bit is enabled, an over-voltage interrupt is triggered as soon as the over-voltage alarm is de-asserted. 11_B Over-voltage alarm enabled. In addition, if the corresponding INT_CONx.OVVxx bit is enabled, an over-voltage interrupt is triggered as soon as either the corresponding VMONS.VxxSTAT.OV status bit is set or the over-voltage alarm is de-asserted.
FILTER	29:26	rw	**Secondary ADC supply filter** 0_H Each conversion result is compared with threshold to generate alarm F_H A spike filter of consecutive 16 ADC results are used to generate alarm to SMU.
0	31:30	r	**Reserved**

Table 1576 Reset values of PMS_VMONS_VDDEVRSBCON

Reset type	Reset value	Note
LVDPMS2 Reset	10F7 E405$_H$	
Cold PORST	10F7 E405$_H$	

36.8.4.62 VDDEVRSB secondary monitor status register

PMS_VMONS_VDDEVRSBSTAT Offset address: 9148_H

VDDEVRSB secondary monitor status register Reset values see: Table 1577

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ACTIVITY						0									
rh						r									

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			UV	OV	RESULT										
r			rh	rh	rh										

Field	Bits	Type	Description
RESULT	10:0	rh	**Secondary ADC conversion result (unfiltered)** This bit field contains the last conversion result of the ADC measurement of the secondary voltage monitor. The LSB is documented for each supply rail in the user manual (secondary voltage monitors section).
OV	11	rh	**Secondary monitor over-voltage event flag after filtering**
UV	12	rh	**Secondary monitor under-voltage event flag after filtering**
ACTIVITY	31:26	rh	**Secondary monitor activity counter** This bit field cumulatively counts the end of conversion signals in a single Secondary Monitor Background Scan over all channels and respective filter configurations of the secondary monitor ADC which monitors this voltage rail. The total number of conversions ConvTot = ∑ [ChX* ChXFIL]. The counter is reset to 0 on a ConvTot overflow.
0	25:13	r	**Reserved**

Table 1577 Reset values of PMS_VMONS_VDDEVRSBSTAT

Reset type	Reset value	Note
LVDPMS2 Reset	$0000\ 0000_H$	
Cold PORST	$0000\ 0000_H$	

36.8.4.63 VDDEXT secondary monitor configuration register

PMS_VMONS_VDDEXTCON Offset address: $914C_H$

VDDEXT secondary monitor configuration register Reset values see: Table 1578

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		FILTER				OVMOD		OVVAL							
r		rw				rw		rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OVVAL			UVMOD		UVVAL										
rw			rw		rw										

Field	Bits	Type	Description
UVVAL	10:0	rw	**Secondary monitor under-voltage threshold** This field defines the under-voltage threshold level of the supply monitor. The LSB is documented for each supply rail in the user manual (secondary voltage monitors section).
UVMOD	12:11	rw	**Under-voltage monitoring mode** 00_B Under-voltage monitoring inactive. 01_B Under-voltage alarm enabled. In addition, if the corresponding INT_CONx.UVVxx bit is enabled, an under-voltage interrupt is triggered as soon as the corresponding VMONS.VxxSTAT.UV status bit is set. 10_B Under-voltage alarm enabled. In addition, if the corresponding INT_CONx.UVVxx bit is enabled, an under-voltage interrupt is triggered as soon as the under-voltage alarm is de-asserted. 11_B Under-voltage alarm enabled. In addition, if the corresponding INT_CONx.UVVxx bit is enabled, an under-voltage interrupt is triggered as soon as either the corresponding VMONS.VxxSTAT.UV status bit is set or the under-voltage alarm is de-asserted.
OVVAL	23:13	rw	**Secondary monitor over-voltage threshold** This field defines the over-voltage threshold level of the supply monitor. The LSB is documented for each supply rail in the user manual (secondary voltage monitors section).
OVMOD	25:24	rw	**Over-voltage monitoring mode** 00_B Over-voltage monitoring inactive. 01_B Over-voltage alarm enabled. In addition, if the corresponding INT_CONx.OVVxx bit is enabled, an over-voltage interrupt is triggered as soon as the corresponding VMONS.VxxSTAT.OV status bit is set. 10_B Over-voltage alarm enabled. In addition, if the corresponding INT_CONx.OVVxx bit is enabled, an over-voltage interrupt is triggered as soon as the over-voltage alarm is de-asserted. 11_B Over-voltage alarm enabled. In addition, if the corresponding INT_CONx.OVVxx bit is enabled, an over-voltage interrupt is triggered as soon as either the corresponding VMONS.VxxSTAT.OV status bit is set or the over-voltage alarm is de-asserted.
FILTER	29:26	rw	**Secondary ADC supply filter** 0_H Each conversion result is compared with threshold to generate alarm F_H A spike filter of consecutive 16 ADC results are used to generate alarm to SMU.
0	31:30	r	**Reserved**

Table 1578 Reset values of PMS_VMONS_VDDEXTCON

Reset type	Reset value	Note
LVDPMS2 Reset	10F7 E405$_H$	
Cold PORST	10F7 E405$_H$	

36.8.4.64 VDDEXT secondary monitor status register

PMS_VMONS_VDDEXTSTAT Offset address: 9150_H

VDDEXT secondary monitor status register Reset values see: Table 1579

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ACTIVITY						0									
rh						r									

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			UV	OV	RESULT										
r			rh	rh	rh										

Field	Bits	Type	Description
RESULT	10:0	rh	**Secondary ADC conversion result (unfiltered)** This bit field contains the last conversion result of the ADC measurement of the secondary voltage monitor. The LSB is documented for each supply rail in the user manual (secondary voltage monitors section).
OV	11	rh	**Secondary monitor over-voltage event flag after filtering**
UV	12	rh	**Secondary monitor under-voltage event flag after filtering**
ACTIVITY	31:26	rh	**Secondary monitor activity counter** This bit field cumulatively counts the end of conversion signals in a single Secondary Monitor Background Scan over all channels and respective filter configurations of the secondary monitor ADC which monitors this voltage rail. The total number of conversions ConvTot = ∑ [ChX* ChXFIL]. The counter is reset to 0 on a ConvTot overflow.
0	25:13	r	**Reserved**

Table 1579 Reset values of PMS_VMONS_VDDEXTSTAT

Reset type	Reset value	Note
LVDPMS2 Reset	$0000\ 0000_H$	
Cold PORST	$0000\ 0000_H$	

36.8.4.65 VDDM secondary monitor configuration register

PMS_VMONS_VDDMCON Offset address: 9154_H

VDDM secondary monitor configuration register Reset values see: Table 1580

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		FILTER				OVMOD		OVVAL							
r		rw				rw		rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OVVAL			UVMOD		UVVAL										
rw			rw		rw										

Field	Bits	Type	Description
UVVAL	10:0	rw	**Secondary monitor under-voltage threshold** This field defines the under-voltage threshold level of the supply monitor. The LSB is documented for each supply rail in the user manual (secondary voltage monitors section).
UVMOD	12:11	rw	**Under-voltage monitoring mode** 00_B Under-voltage monitoring inactive. 01_B Under-voltage alarm enabled. In addition, if the corresponding INT_CONx.UVVxx bit is enabled, an under-voltage interrupt is triggered as soon as the corresponding VMONS.VxxSTAT.UV status bit is set. 10_B Under-voltage alarm enabled. In addition, if the corresponding INT_CONx.UVVxx bit is enabled, an under-voltage interrupt is triggered as soon as the under-voltage alarm is de-asserted. 11_B Under-voltage alarm enabled. In addition, if the corresponding INT_CONx.UVVxx bit is enabled, an under-voltage interrupt is triggered as soon as either the corresponding VMONS.VxxSTAT.UV status bit is set or the under-voltage alarm is de-asserted.
OVVAL	23:13	rw	**Secondary monitor over-voltage threshold** This field defines the over-voltage threshold level of the supply monitor. The LSB is documented for each supply rail in the user manual (secondary voltage monitors section).
OVMOD	25:24	rw	**Over-voltage monitoring mode** 00_B Over-voltage monitoring inactive. 01_B Over-voltage alarm enabled. In addition, if the corresponding INT_CONx.OVVxx bit is enabled, an over-voltage interrupt is triggered as soon as the corresponding VMONS.VxxSTAT.OV status bit is set. 10_B Over-voltage alarm enabled. In addition, if the corresponding INT_CONx.OVVxx bit is enabled, an over-voltage interrupt is triggered as soon as the over-voltage alarm is de-asserted. 11_B Over-voltage alarm enabled. In addition, if the corresponding INT_CONx.OVVxx bit is enabled, an over-voltage interrupt is triggered as soon as either the corresponding VMONS.VxxSTAT.OV status bit is set or the over-voltage alarm is de-asserted.
FILTER	29:26	rw	**Secondary ADC supply filter** 0_H Each conversion result is compared with threshold to generate alarm F_H A spike filter of consecutive 16 ADC results are used to generate alarm to SMU.
0	31:30	r	**Reserved**

Table 1580 Reset values of PMS_VMONS_VDDMCON

Reset type	Reset value	Note
LVDPMS2 Reset	10F7 E405$_H$	
Cold PORST	10F7 E405$_H$	

36.8.4.66 VDDM secondary monitor status register

PMS_VMONS_VDDMSTAT Offset address: 9158_H

VDDM secondary monitor status register Reset values see: Table 1581

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ACTIVITY						0									
rh						r									

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			UV	OV	RESULT										
r			rh	rh	rh										

Field	Bits	Type	Description
RESULT	10:0	rh	**Secondary ADC conversion result (unfiltered)** This bit field contains the last conversion result of the ADC measurement of the secondary voltage monitor. The LSB is documented for each supply rail in the user manual (secondary voltage monitors section).
OV	11	rh	**Secondary monitor over-voltage event flag after filtering**
UV	12	rh	**Secondary monitor under-voltage event flag after filtering**
ACTIVITY	31:26	rh	**Secondary monitor activity counter** This bit field cumulatively counts the end of conversion signals in a single Secondary Monitor Background Scan over all channels and respective filter configurations of the secondary monitor ADC which monitors this voltage rail. The total number of conversions ConvTot = ∑ [ChX* ChXFIL]. The counter is reset to 0 on a ConvTot overflow.
0	25:13	r	**Reserved**

Table 1581 Reset values of PMS_VMONS_VDDMSTAT

Reset type	Reset value	Note
LVDPMS2 Reset	$0000\ 0000_H$	
Cold PORST	$0000\ 0000_H$	

36.8.4.67 VDD secondary monitor configuration register

PMS_VMONS_VDDCON Offset address: 9164_H

VDD secondary monitor configuration register Reset values see: Table 1582

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		FILTER				OVMOD		OVVAL							
r		rw				rw		rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OVVAL			UVMOD		UVVAL										
rw			rw		rw										

Field	Bits	Type	Description
UVVAL	10:0	rw	**Secondary monitor under-voltage threshold** This field defines the under-voltage threshold level of the supply monitor. The LSB is documented for each supply rail in the user manual (secondary voltage monitors section).
UVMOD	12:11	rw	**Under-voltage monitoring mode** 00_B Under-voltage monitoring inactive. 01_B Under-voltage alarm enabled. In addition, if the corresponding INT_CONx.UVVxx bit is enabled, an under-voltage interrupt is triggered as soon as the corresponding VMONS.VxxSTAT.UV status bit is set. 10_B Under-voltage alarm enabled. In addition, if the corresponding INT_CONx.UVVxx bit is enabled, an under-voltage interrupt is triggered as soon as the under-voltage alarm is de-asserted. 11_B Under-voltage alarm enabled. In addition, if the corresponding INT_CONx.UVVxx bit is enabled, an under-voltage interrupt is triggered as soon as either the corresponding VMONS.VxxSTAT.UV status bit is set or the under-voltage alarm is de-asserted.
OVVAL	23:13	rw	**Secondary monitor over-voltage threshold** This field defines the over-voltage threshold level of the supply monitor. The LSB is documented for each supply rail in the user manual (secondary voltage monitors section).
OVMOD	25:24	rw	**Over-voltage monitoring mode** 00_B Over-voltage monitoring inactive. 01_B Over-voltage alarm enabled. In addition, if the corresponding INT_CONx.OVVxx bit is enabled, an over-voltage interrupt is triggered as soon as the corresponding VMONS.VxxSTAT.OV status bit is set. 10_B Over-voltage alarm enabled. In addition, if the corresponding INT_CONx.OVVxx bit is enabled, an over-voltage interrupt is triggered as soon as the over-voltage alarm is de-asserted. 11_B Over-voltage alarm enabled. In addition, if the corresponding INT_CONx.OVVxx bit is enabled, an over-voltage interrupt is triggered as soon as either the corresponding VMONS.VxxSTAT.OV status bit is set or the over-voltage alarm is de-asserted.
FILTER	29:26	rw	**Secondary ADC supply filter** 0_H Each conversion result is compared with threshold to generate alarm F_H A spike filter of consecutive 16 ADC results are used to generate alarm to SMU.
0	31:30	r	**Reserved**

Table 1582 Reset values of PMS_VMONS_VDDCON

Reset type	Reset value	Note
LVDPMS2 Reset	10F5 85DD$_H$	
Cold PORST	10F5 85DD$_H$	

36.8.4.68 VDD secondary monitor status register

PMS_VMONS_VDDSTAT Offset address: 9168_H

VDD secondary monitor status register Reset values see: Table 1583

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ACTIVITY						0									
rh						r									

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			UV	OV	RESULT										
r			rh	rh	rh										

Field	Bits	Type	Description
RESULT	10:0	rh	**Secondary ADC conversion result (unfiltered)** This bit field contains the last conversion result of the ADC measurement of the secondary voltage monitor. The LSB is documented for each supply rail in the user manual (secondary voltage monitors section).
OV	11	rh	**Secondary monitor over-voltage event flag after filtering**
UV	12	rh	**Secondary monitor under-voltage event flag after filtering**
ACTIVITY	31:26	rh	**Secondary monitor activity counter** This bit field cumulatively counts the end of conversion signals in a single Secondary Monitor Background Scan over all channels and respective filter configurations of the secondary monitor ADC which monitors this voltage rail. The total number of conversions ConvTot = ∑ [ChX* ChXFIL]. The counter is reset to 0 on a ConvTot overflow.
0	25:13	r	**Reserved**

Table 1583 Reset values of PMS_VMONS_VDDSTAT

Reset type	Reset value	Note
LVDPMS2 Reset	$0000\ 0000_H$	
Cold PORST	$0000\ 0000_H$	

36.8.4.69 VDDPMS0 secondary monitor configuration register

PMS_VMONS_VDDPMS0CON Offset address: $916C_H$

VDDPMS0 secondary monitor configuration register Reset values see: Table 1584

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		FILTER				OVMOD		OVVAL							
r		rw				rw		rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OVVAL			UVMOD		UVVAL										
rw			rw		rw										

Field	Bits	Type	Description
UVVAL	10:0	rw	**Secondary monitor under-voltage threshold** This field defines the under-voltage threshold level of the supply monitor. The LSB is documented for each supply rail in the user manual (secondary voltage monitors section).
UVMOD	12:11	rw	**Under-voltage monitoring mode** 00_B Under-voltage monitoring inactive. 01_B Under-voltage alarm enabled. In addition, if the corresponding INT_CONx.UVVxx bit is enabled, an under-voltage interrupt is triggered as soon as the corresponding VMONS.VxxSTAT.UV status bit is set. 10_B Under-voltage alarm enabled. In addition, if the corresponding INT_CONx.UVVxx bit is enabled, an under-voltage interrupt is triggered as soon as the under-voltage alarm is de-asserted. 11_B Under-voltage alarm enabled. In addition, if the corresponding INT_CONx.UVVxx bit is enabled, an under-voltage interrupt is triggered as soon as either the corresponding VMONS.VxxSTAT.UV status bit is set or the under-voltage alarm is de-asserted.
OVVAL	23:13	rw	**Secondary monitor over-voltage threshold** This field defines the over-voltage threshold level of the supply monitor. The LSB is documented for each supply rail in the user manual (secondary voltage monitors section).
OVMOD	25:24	rw	**Over-voltage monitoring mode** 00_B Over-voltage monitoring inactive. 01_B Over-voltage alarm enabled. In addition, if the corresponding INT_CONx.OVVxx bit is enabled, an over-voltage interrupt is triggered as soon as the corresponding VMONS.VxxSTAT.OV status bit is set. 10_B Over-voltage alarm enabled. In addition, if the corresponding INT_CONx.OVVxx bit is enabled, an over-voltage interrupt is triggered as soon as the over-voltage alarm is de-asserted. 11_B Over-voltage alarm enabled. In addition, if the corresponding INT_CONx.OVVxx bit is enabled, an over-voltage interrupt is triggered as soon as either the corresponding VMONS.VxxSTAT.OV status bit is set or the over-voltage alarm is de-asserted.
FILTER	29:26	rw	**Secondary ADC supply filter** 0_H Each conversion result is compared with threshold to generate alarm F_H A spike filter of consecutive 16 ADC results are used to generate alarm to SMU.
0	31:30	r	**Reserved**

Table 1584 Reset values of PMS_VMONS_VDDPMS0CON

Reset type	Reset value	Note
LVDPMS2 Reset	11F0 $0D21_H$	
Cold PORST	11F0 $0D21_H$	
After Boot-FW Value	10EB $253B_H$	

36.8.4.70 VDDPMS0 secondary monitor status register

PMS_VMONS_VDDPMS0STAT Offset address: 9170_H

VDDPMS0 secondary monitor status register Reset values see: Table 1585

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ACTIVITY						0									
rh						r									

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			UV	OV	RESULT										
r			rh	rh	rh										

Field	Bits	Type	Description
RESULT	10:0	rh	**Secondary ADC conversion result (unfiltered)** This bit field contains the last conversion result of the ADC measurement of the secondary voltage monitor. The LSB is documented for each supply rail in the user manual (secondary voltage monitors section).
OV	11	rh	**Secondary monitor over-voltage event flag after filtering**
UV	12	rh	**Secondary monitor under-voltage event flag after filtering**
ACTIVITY	31:26	rh	**Secondary monitor activity counter** This bit field cumulatively counts the end of conversion signals in a single Secondary Monitor Background Scan over all channels and respective filter configurations of the secondary monitor ADC which monitors this voltage rail. The total number of conversions ConvTot = ∑ [ChX* ChXFIL]. The counter is reset to 0 on a ConvTot overflow.
0	25:13	r	**Reserved**

Table 1585 Reset values of PMS_VMONS_VDDPMS0STAT

Reset type	Reset value	Note
LVDPMS2 Reset	$0000\ 1800_H$	
Cold PORST	$0000\ 1800_H$	

36.8.4.71 VDDPMS1 secondary monitor configuration register

PMS_VMONS_VDDPMS1CON Offset address: 9174_H

VDDPMS1 secondary monitor configuration register Reset values see: Table 1586

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		FILTER				OVMOD		OVVAL							
r		rw				rw		rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OVVAL			UVMOD		UVVAL										
rw			rw		rw										

Field	Bits	Type	Description
UVVAL	10:0	rw	**Secondary monitor under-voltage threshold** This field defines the under-voltage threshold level of the supply monitor. The LSB is documented for each supply rail in the user manual (secondary voltage monitors section).
UVMOD	12:11	rw	**Under-voltage monitoring mode** 00_B Under-voltage monitoring inactive. 01_B Under-voltage alarm enabled. In addition, if the corresponding INT_CONx.UVVxx bit is enabled, an under-voltage interrupt is triggered as soon as the corresponding VMONS.VxxSTAT.UV status bit is set. 10_B Under-voltage alarm enabled. In addition, if the corresponding INT_CONx.UVVxx bit is enabled, an under-voltage interrupt is triggered as soon as the under-voltage alarm is de-asserted. 11_B Under-voltage alarm enabled. In addition, if the corresponding INT_CONx.UVVxx bit is enabled, an under-voltage interrupt is triggered as soon as either the corresponding VMONS.VxxSTAT.UV status bit is set or the under-voltage alarm is de-asserted.
OVVAL	23:13	rw	**Secondary monitor over-voltage threshold** This field defines the over-voltage threshold level of the supply monitor. The LSB is documented for each supply rail in the user manual (secondary voltage monitors section).
OVMOD	25:24	rw	**Over-voltage monitoring mode** 00_B Over-voltage monitoring inactive. 01_B Over-voltage alarm enabled. In addition, if the corresponding INT_CONx.OVVxx bit is enabled, an over-voltage interrupt is triggered as soon as the corresponding VMONS.VxxSTAT.OV status bit is set. 10_B Over-voltage alarm enabled. In addition, if the corresponding INT_CONx.OVVxx bit is enabled, an over-voltage interrupt is triggered as soon as the over-voltage alarm is de-asserted. 11_B Over-voltage alarm enabled. In addition, if the corresponding INT_CONx.OVVxx bit is enabled, an over-voltage interrupt is triggered as soon as either the corresponding VMONS.VxxSTAT.OV status bit is set or the over-voltage alarm is de-asserted.
FILTER	29:26	rw	**Secondary ADC supply filter** 0_H Each conversion result is compared with threshold to generate alarm F_H A spike filter of consecutive 16 ADC results are used to generate alarm to SMU.
0	31:30	r	**Reserved**

Table 1586 Reset values of PMS_VMONS_VDDPMS1CON

Reset type	Reset value	Note
LVDPMS2 Reset	11FB 2D31$_H$	
Cold PORST	11FB 2D31$_H$	
After Boot-FW Value	10F6 054B$_H$	

36.8.4.72 VDDPMS1 secondary monitor status register

PMS_VMONS_VDDPMS1STAT Offset address: 9178_H

VDDPMS1 secondary monitor status register Reset values see: Table 1587

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ACTIVITY						0									
rh						r									

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			UV	OV	RESULT										
r			rh	rh	rh										

Field	Bits	Type	Description
RESULT	10:0	rh	**Secondary ADC conversion result (unfiltered)** This bit field contains the last conversion result of the ADC measurement of the secondary voltage monitor. The LSB is documented for each supply rail in the user manual (secondary voltage monitors section).
OV	11	rh	**Secondary monitor over-voltage event flag after filtering**
UV	12	rh	**Secondary monitor under-voltage event flag after filtering**
ACTIVITY	31:26	rh	**Secondary monitor activity counter** This bit field cumulatively counts the end of conversion signals in a single Secondary Monitor Background Scan over all channels and respective filter configurations of the secondary monitor ADC which monitors this voltage rail. The total number of conversions ConvTot = ∑ [ChX* ChXFIL]. The counter is reset to 0 on a ConvTot overflow.
0	25:13	r	**Reserved**

Table 1587 Reset values of PMS_VMONS_VDDPMS1STAT

Reset type	Reset value	Note
LVDPMS2 Reset	$0000\ 1800_H$	
Cold PORST	$0000\ 1800_H$	

36.8.4.73 VDDPMS2 secondary monitor configuration register

PMS_VMONS_VDDPMS2CON Offset address: $917C_H$

VDDPMS2 secondary monitor configuration register Reset values see: Table 1588

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		FILTER				OVMOD		OVVAL							
r		rw				rw		rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OVVAL			UVMOD		UVVAL										
rw			rw		rw										

Field	Bits	Type	Description
UVVAL	10:0	rw	**Secondary monitor under-voltage threshold** This field defines the under-voltage threshold level of the supply monitor. The LSB is documented for each supply rail in the user manual (secondary voltage monitors section).
UVMOD	12:11	rw	**Under-voltage monitoring mode** 00_B Under-voltage monitoring inactive. 01_B Under-voltage alarm enabled. In addition, if the corresponding INT_CONx.UVVxx bit is enabled, an under-voltage interrupt is triggered as soon as the corresponding VMONS.VxxSTAT.UV status bit is set. 10_B Under-voltage alarm enabled. In addition, if the corresponding INT_CONx.UVVxx bit is enabled, an under-voltage interrupt is triggered as soon as the under-voltage alarm is de-asserted. 11_B Under-voltage alarm enabled. In addition, if the corresponding INT_CONx.UVVxx bit is enabled, an under-voltage interrupt is triggered as soon as either the corresponding VMONS.VxxSTAT.UV status bit is set or the under-voltage alarm is de-asserted.
OVVAL	23:13	rw	**Secondary monitor over-voltage threshold** This field defines the over-voltage threshold level of the supply monitor. The LSB is documented for each supply rail in the user manual (secondary voltage monitors section).
OVMOD	25:24	rw	**Over-voltage monitoring mode** 00_B Over-voltage monitoring inactive. 01_B Over-voltage alarm enabled. In addition, if the corresponding INT_CONx.OVVxx bit is enabled, an over-voltage interrupt is triggered as soon as the corresponding VMONS.VxxSTAT.OV status bit is set. 10_B Over-voltage alarm enabled. In addition, if the corresponding INT_CONx.OVVxx bit is enabled, an over-voltage interrupt is triggered as soon as the over-voltage alarm is de-asserted. 11_B Over-voltage alarm enabled. In addition, if the corresponding INT_CONx.OVVxx bit is enabled, an over-voltage interrupt is triggered as soon as either the corresponding VMONS.VxxSTAT.OV status bit is set or the over-voltage alarm is de-asserted.
FILTER	29:26	rw	**Secondary ADC supply filter** 0_H Each conversion result is compared with threshold to generate alarm F_H A spike filter of consecutive 16 ADC results are used to generate alarm to SMU.
0	31:30	r	**Reserved**

Table 1588 Reset values of PMS_VMONS_VDDPMS2CON

Reset type	Reset value	Note
LVDPMS2 Reset	11FB 2D31$_H$	
Cold PORST	11FB 2D31$_H$	
After Boot-FW Value	10F6 054B$_H$	

36.8.4.74 VDDPMS2 secondary monitor status register

PMS_VMONS_VDDPMS2STAT Offset address: 9180_H

VDDPMS2 secondary monitor status register Reset values see: Table 1589

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ACTIVITY						0									
rh						r									

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			UV	OV	RESULT										
r			rh	rh	rh										

Field	Bits	Type	Description
RESULT	10:0	rh	**Secondary ADC conversion result (unfiltered)** This bit field contains the last conversion result of the ADC measurement of the secondary voltage monitor. The LSB is documented for each supply rail in the user manual (secondary voltage monitors section).
OV	11	rh	**Secondary monitor over-voltage event flag after filtering**
UV	12	rh	**Secondary monitor under-voltage event flag after filtering**
ACTIVITY	31:26	rh	**Secondary monitor activity counter** This bit field cumulatively counts the end of conversion signals in a single Secondary Monitor Background Scan over all channels and respective filter configurations of the secondary monitor ADC which monitors this voltage rail. The total number of conversions ConvTot = ∑ [ChX* ChXFIL]. The counter is reset to 0 on a ConvTot overflow.
0	25:13	r	**Reserved**

Table 1589 Reset values of PMS_VMONS_VDDPMS2STAT

Reset type	Reset value	Note
LVDPMS2 Reset	$0000\ 1800_H$	
Cold PORST	$0000\ 1800_H$	

36.8.4.75 VDDSBRAM secondary monitor configuration register

PMS_VMONS_VDDSBRAMCON Offset address: 9184_H

VDDSBRAM secondary monitor configuration register Reset values see: Table 1590

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		FILTER				OVMOD		OVVAL							
r		rw				rw		rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OVVAL			UVMOD		UVVAL										
rw			rw		rw										

Field	Bits	Type	Description
UVVAL	10:0	rw	**Secondary monitor under-voltage threshold** This field defines the under-voltage threshold level of the supply monitor. The LSB is documented for each supply rail in the user manual (secondary voltage monitors section).
UVMOD	12:11	rw	**Under-voltage monitoring mode** 00_B Under-voltage monitoring inactive. 01_B Under-voltage alarm enabled. In addition, if the corresponding INT_CONx.UVVxx bit is enabled, an under-voltage interrupt is triggered as soon as the corresponding VMONS.VxxSTAT.UV status bit is set. 10_B Under-voltage alarm enabled. In addition, if the corresponding INT_CONx.UVVxx bit is enabled, an under-voltage interrupt is triggered as soon as the under-voltage alarm is de-asserted. 11_B Under-voltage alarm enabled. In addition, if the corresponding INT_CONx.UVVxx bit is enabled, an under-voltage interrupt is triggered as soon as either the corresponding VMONS.VxxSTAT.UV status bit is set or the under-voltage alarm is de-asserted.
OVVAL	23:13	rw	**Secondary monitor over-voltage threshold** This field defines the over-voltage threshold level of the supply monitor. The LSB is documented for each supply rail in the user manual (secondary voltage monitors section).
OVMOD	25:24	rw	**Over-voltage monitoring mode** 00_B Over-voltage monitoring inactive. 01_B Over-voltage alarm enabled. In addition, if the corresponding INT_CONx.OVVxx bit is enabled, an over-voltage interrupt is triggered as soon as the corresponding VMONS.VxxSTAT.OV status bit is set. 10_B Over-voltage alarm enabled. In addition, if the corresponding INT_CONx.OVVxx bit is enabled, an over-voltage interrupt is triggered as soon as the over-voltage alarm is de-asserted. 11_B Over-voltage alarm enabled. In addition, if the corresponding INT_CONx.OVVxx bit is enabled, an over-voltage interrupt is triggered as soon as either the corresponding VMONS.VxxSTAT.OV status bit is set or the over-voltage alarm is de-asserted.
FILTER	29:26	rw	**Secondary ADC supply filter** 0_H Each conversion result is compared with threshold to generate alarm F_H A spike filter of consecutive 16 ADC results are used to generate alarm to SMU.
0	31:30	r	**Reserved**

Table 1590 Reset values of PMS_VMONS_VDDSBRAMCON

Reset type	Reset value	Note
LVDPMS2 Reset	$10EB\ 253B_H$	
Cold PORST	$10EB\ 253B_H$	

36.8.4.76 VDDSBRAM secondary monitor status register

PMS_VMONS_VDDSBRAMSTAT Offset address: 9188_H

VDDSBRAM secondary monitor status register Reset values see: Table 1591

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ACTIVITY						0									
rh						r									

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			UV	OV	RESULT										
r			rh	rh	rh										

Field	Bits	Type	Description
RESULT	10:0	rh	**Secondary ADC conversion result (unfiltered)** This bit field contains the last conversion result of the ADC measurement of the secondary voltage monitor. The LSB is documented for each supply rail in the user manual (secondary voltage monitors section).
OV	11	rh	**Secondary monitor over-voltage event flag after filtering**
UV	12	rh	**Secondary monitor under-voltage event flag after filtering**
ACTIVITY	31:26	rh	**Secondary monitor activity counter** This bit field cumulatively counts the end of conversion signals in a single Secondary Monitor Background Scan over all channels and respective filter configurations of the secondary monitor ADC which monitors this voltage rail. The total number of conversions ConvTot = ∑ [ChX* ChXFIL]. The counter is reset to 0 on a ConvTot overflow.
0	25:13	r	**Reserved**

Table 1591 Reset values of PMS_VMONS_VDDSBRAMSTAT

Reset type	Reset value	Note
LVDPMS2 Reset	$0000\ 0000_H$	
Cold PORST	$0000\ 0000_H$	

36.8.4.77 VDDPAD secondary monitor configuration register

PMS_VMONS_VDDPADCON Offset address: $918C_H$

VDDPAD secondary monitor configuration register Reset values see: Table 1592

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		FILTER				OVMOD		OVVAL							
r		rw				rw		rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OVVAL			UVMOD		UVVAL										
rw			rw		rw										

Field	Bits	Type	Description
UVVAL	10:0	rw	**Secondary monitor under-voltage threshold** This field defines the under-voltage threshold level of the supply monitor. The LSB is documented for each supply rail in the user manual (secondary voltage monitors section).
UVMOD	12:11	rw	**Under-voltage monitoring mode** 00_B Under-voltage monitoring inactive. 01_B Under-voltage alarm enabled. In addition, if the corresponding INT_CONx.UVVxx bit is enabled, an under-voltage interrupt is triggered as soon as the corresponding VMONS.VxxSTAT.UV status bit is set. 10_B Under-voltage alarm enabled. In addition, if the corresponding INT_CONx.UVVxx bit is enabled, an under-voltage interrupt is triggered as soon as the under-voltage alarm is de-asserted. 11_B Under-voltage alarm enabled. In addition, if the corresponding INT_CONx.UVVxx bit is enabled, an under-voltage interrupt is triggered as soon as either the corresponding VMONS.VxxSTAT.UV status bit is set or the under-voltage alarm is de-asserted.
OVVAL	23:13	rw	**Secondary monitor over-voltage threshold** This field defines the over-voltage threshold level of the supply monitor. The LSB is documented for each supply rail in the user manual (secondary voltage monitors section).
OVMOD	25:24	rw	**Over-voltage monitoring mode** 00_B Over-voltage monitoring inactive. 01_B Over-voltage alarm enabled. In addition, if the corresponding INT_CONx.OVVxx bit is enabled, an over-voltage interrupt is triggered as soon as the corresponding VMONS.VxxSTAT.OV status bit is set. 10_B Over-voltage alarm enabled. In addition, if the corresponding INT_CONx.OVVxx bit is enabled, an over-voltage interrupt is triggered as soon as the over-voltage alarm is de-asserted. 11_B Over-voltage alarm enabled. In addition, if the corresponding INT_CONx.OVVxx bit is enabled, an over-voltage interrupt is triggered as soon as either the corresponding VMONS.VxxSTAT.OV status bit is set or the over-voltage alarm is de-asserted.
FILTER	29:26	rw	**Secondary ADC supply filter** 0_H Each conversion result is compared with threshold to generate alarm F_H A spike filter of consecutive 16 ADC results are used to generate alarm to SMU.
0	31:30	r	**Reserved**

Table 1592 Reset values of PMS_VMONS_VDDPADCON

Reset type	Reset value	Note
LVDPMS2 Reset	$10F5\ 85DD_H$	
Cold PORST	$10F5\ 85DD_H$	

36.8.4.78 VDDPAD secondary monitor status register

PMS_VMONS_VDDPADSTAT Offset address: 9190_H

VDDPAD secondary monitor status register Reset values see: Table 1593

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ACTIVITY						0									
rh						r									

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			UV	OV	RESULT										
r			rh	rh	rh										

Field	Bits	Type	Description
RESULT	10:0	rh	**Secondary ADC conversion result (unfiltered)** This bit field contains the last conversion result of the ADC measurement of the secondary voltage monitor. The LSB is documented for each supply rail in the user manual (secondary voltage monitors section).
OV	11	rh	**Secondary monitor over-voltage event flag after filtering**
UV	12	rh	**Secondary monitor under-voltage event flag after filtering**
ACTIVITY	31:26	rh	**Secondary monitor activity counter** This bit field cumulatively counts the end of conversion signals in a single Secondary Monitor Background Scan over all channels and respective filter configurations of the secondary monitor ADC which monitors this voltage rail. The total number of conversions ConvTot = ∑ [ChX* ChXFIL]. The counter is reset to 0 on a ConvTot overflow.
0	25:13	r	**Reserved**

Table 1593 Reset values of PMS_VMONS_VDDPADSTAT

Reset type	Reset value	Note
LVDPMS2 Reset	$0000\ 0000_H$	
Cold PORST	$0000\ 0000_H$	

36.8.4.79 VDDPHPHY0 secondary monitor configuration register

PMS_VMONS_VDDPHPHY0CON Offset address: $919C_H$

VDDPHPHY0 secondary monitor configuration register Reset values see: Table 1594

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		FILTER				OVMOD		OVVAL							
r		rw				rw		rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OVVAL			UVMOD		UVVAL										
rw			rw		rw										

Field	Bits	Type	Description
UVVAL	10:0	rw	**Secondary monitor under-voltage threshold** This field defines the under-voltage threshold level of the supply monitor. The LSB is documented for each supply rail in the user manual (secondary voltage monitors section).
UVMOD	12:11	rw	**Under-voltage monitoring mode** 00_B Under-voltage monitoring inactive. 01_B Under-voltage alarm enabled. In addition, if the corresponding INT_CONx.UVVxx bit is enabled, an under-voltage interrupt is triggered as soon as the corresponding VMONS.VxxSTAT.UV status bit is set. 10_B Under-voltage alarm enabled. In addition, if the corresponding INT_CONx.UVVxx bit is enabled, an under-voltage interrupt is triggered as soon as the under-voltage alarm is de-asserted. 11_B Under-voltage alarm enabled. In addition, if the corresponding INT_CONx.UVVxx bit is enabled, an under-voltage interrupt is triggered as soon as either the corresponding VMONS.VxxSTAT.UV status bit is set or the under-voltage alarm is de-asserted.
OVVAL	23:13	rw	**Secondary monitor over-voltage threshold** This field defines the over-voltage threshold level of the supply monitor. The LSB is documented for each supply rail in the user manual (secondary voltage monitors section).
OVMOD	25:24	rw	**Over-voltage monitoring mode** 00_B Over-voltage monitoring inactive. 01_B Over-voltage alarm enabled. In addition, if the corresponding INT_CONx.OVVxx bit is enabled, an over-voltage interrupt is triggered as soon as the corresponding VMONS.VxxSTAT.OV status bit is set. 10_B Over-voltage alarm enabled. In addition, if the corresponding INT_CONx.OVVxx bit is enabled, an over-voltage interrupt is triggered as soon as the over-voltage alarm is de-asserted. 11_B Over-voltage alarm enabled. In addition, if the corresponding INT_CONx.OVVxx bit is enabled, an over-voltage interrupt is triggered as soon as either the corresponding VMONS.VxxSTAT.OV status bit is set or the over-voltage alarm is de-asserted.
FILTER	29:26	rw	**Secondary ADC supply filter** 0_H Each conversion result is compared with threshold to generate alarm F_H A spike filter of consecutive 16 ADC results are used to generate alarm to SMU.
0	31:30	r	**Reserved**

Table 1594 Reset values of PMS_VMONS_VDDPHPHY0CON

Reset type	Reset value	Note
LVDPMS2 Reset	$1059\ 4231_H$	
Cold PORST	$1059\ 4231_H$	

36.8.4.80 VDDPHPHY0 secondary monitor status register

PMS_VMONS_VDDPHPHY0STAT Offset address: $91A0_H$

VDDPHPHY0 secondary monitor status register Reset values see: Table 1595

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ACTIVITY						0									
rh						r									

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			UV	OV	RESULT										
r			rh	rh	rh										

Field	Bits	Type	Description
RESULT	10:0	rh	**Secondary ADC conversion result (unfiltered)** This bit field contains the last conversion result of the ADC measurement of the secondary voltage monitor. The LSB is documented for each supply rail in the user manual (secondary voltage monitors section).
OV	11	rh	**Secondary monitor over-voltage event flag after filtering**
UV	12	rh	**Secondary monitor under-voltage event flag after filtering**
ACTIVITY	31:26	rh	**Secondary monitor activity counter** This bit field cumulatively counts the end of conversion signals in a single Secondary Monitor Background Scan over all channels and respective filter configurations of the secondary monitor ADC which monitors this voltage rail. The total number of conversions ConvTot = ∑ [ChX* ChXFIL]. The counter is reset to 0 on a ConvTot overflow.
0	25:13	r	**Reserved**

Table 1595 Reset values of PMS_VMONS_VDDPHPHY0STAT

Reset type	Reset value	Note
LVDPMS2 Reset	$0000\ 0000_H$	
Cold PORST	$0000\ 0000_H$	

36.8.4.81 VDDPHPHY1 secondary monitor configuration register

PMS_VMONS_VDDPHPHY1CON Offset address: $91A4_H$

VDDPHPHY1 secondary monitor configuration register Reset values see: Table 1596

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		FILTER				OVMOD		OVVAL							
r		rw				rw		rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OVVAL			UVMOD		UVVAL										
rw			rw		rw										

Field	Bits	Type	Description
UVVAL	10:0	rw	**Secondary monitor under-voltage threshold** This field defines the under-voltage threshold level of the supply monitor. The LSB is documented for each supply rail in the user manual (secondary voltage monitors section).
UVMOD	12:11	rw	**Under-voltage monitoring mode** 00_B Under-voltage monitoring inactive. 01_B Under-voltage alarm enabled. In addition, if the corresponding INT_CONx.UVVxx bit is enabled, an under-voltage interrupt is triggered as soon as the corresponding VMONS.VxxSTAT.UV status bit is set. 10_B Under-voltage alarm enabled. In addition, if the corresponding INT_CONx.UVVxx bit is enabled, an under-voltage interrupt is triggered as soon as the under-voltage alarm is de-asserted. 11_B Under-voltage alarm enabled. In addition, if the corresponding INT_CONx.UVVxx bit is enabled, an under-voltage interrupt is triggered as soon as either the corresponding VMONS.VxxSTAT.UV status bit is set or the under-voltage alarm is de-asserted.
OVVAL	23:13	rw	**Secondary monitor over-voltage threshold** This field defines the over-voltage threshold level of the supply monitor. The LSB is documented for each supply rail in the user manual (secondary voltage monitors section).
OVMOD	25:24	rw	**Over-voltage monitoring mode** 00_B Over-voltage monitoring inactive. 01_B Over-voltage alarm enabled. In addition, if the corresponding INT_CONx.OVVxx bit is enabled, an over-voltage interrupt is triggered as soon as the corresponding VMONS.VxxSTAT.OV status bit is set. 10_B Over-voltage alarm enabled. In addition, if the corresponding INT_CONx.OVVxx bit is enabled, an over-voltage interrupt is triggered as soon as the over-voltage alarm is de-asserted. 11_B Over-voltage alarm enabled. In addition, if the corresponding INT_CONx.OVVxx bit is enabled, an over-voltage interrupt is triggered as soon as either the corresponding VMONS.VxxSTAT.OV status bit is set or the over-voltage alarm is de-asserted.
FILTER	29:26	rw	**Secondary ADC supply filter** 0_H Each conversion result is compared with threshold to generate alarm F_H A spike filter of consecutive 16 ADC results are used to generate alarm to SMU.
0	31:30	r	**Reserved**

Table 1596 Reset values of PMS_VMONS_VDDPHPHY1CON

Reset type	Reset value	Note
LVDPMS2 Reset	$1059\ 4231_H$	
Cold PORST	$1059\ 4231_H$	

36.8.4.82 VDDPHPHY1 secondary monitor status register

PMS_VMONS_VDDPHPHY1STAT Offset address: $91A8_H$

VDDPHPHY1 secondary monitor status register Reset values see: Table 1597

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ACTIVITY						0									
rh						r									

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			UV	OV	RESULT										
r			rh	rh	rh										

Field	Bits	Type	Description
RESULT	10:0	rh	**Secondary ADC conversion result (unfiltered)** This bit field contains the last conversion result of the ADC measurement of the secondary voltage monitor. The LSB is documented for each supply rail in the user manual (secondary voltage monitors section).
OV	11	rh	**Secondary monitor over-voltage event flag after filtering**
UV	12	rh	**Secondary monitor under-voltage event flag after filtering**
ACTIVITY	31:26	rh	**Secondary monitor activity counter** This bit field cumulatively counts the end of conversion signals in a single Secondary Monitor Background Scan over all channels and respective filter configurations of the secondary monitor ADC which monitors this voltage rail. The total number of conversions ConvTot = ∑ [ChX* ChXFIL]. The counter is reset to 0 on a ConvTot overflow.
0	25:13	r	**Reserved**

Table 1597 Reset values of PMS_VMONS_VDDPHPHY1STAT

Reset type	Reset value	Note
LVDPMS2 Reset	$0000\ 0000_H$	
Cold PORST	$0000\ 0000_H$	

36.8.4.83 VDDPHPHY2 secondary monitor configuration register

PMS_VMONS_VDDPHPHY2CON Offset address: $91AC_H$

VDDPHPHY2 secondary monitor configuration register Reset values see: Table 1598

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		FILTER				OVMOD		OVVAL							
r		rw				rw		rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OVVAL			UVMOD		UVVAL										
rw			rw		rw										

Field	Bits	Type	Description
UVVAL	10:0	rw	**Secondary monitor under-voltage threshold** This field defines the under-voltage threshold level of the supply monitor. The LSB is documented for each supply rail in the user manual (secondary voltage monitors section).
UVMOD	12:11	rw	**Under-voltage monitoring mode** 00_B Under-voltage monitoring inactive. 01_B Under-voltage alarm enabled. In addition, if the corresponding INT_CONx.UVVxx bit is enabled, an under-voltage interrupt is triggered as soon as the corresponding VMONS.VxxSTAT.UV status bit is set. 10_B Under-voltage alarm enabled. In addition, if the corresponding INT_CONx.UVVxx bit is enabled, an under-voltage interrupt is triggered as soon as the under-voltage alarm is de-asserted. 11_B Under-voltage alarm enabled. In addition, if the corresponding INT_CONx.UVVxx bit is enabled, an under-voltage interrupt is triggered as soon as either the corresponding VMONS.VxxSTAT.UV status bit is set or the under-voltage alarm is de-asserted.
OVVAL	23:13	rw	**Secondary monitor over-voltage threshold** This field defines the over-voltage threshold level of the supply monitor. The LSB is documented for each supply rail in the user manual (secondary voltage monitors section).
OVMOD	25:24	rw	**Over-voltage monitoring mode** 00_B Over-voltage monitoring inactive. 01_B Over-voltage alarm enabled. In addition, if the corresponding INT_CONx.OVVxx bit is enabled, an over-voltage interrupt is triggered as soon as the corresponding VMONS.VxxSTAT.OV status bit is set. 10_B Over-voltage alarm enabled. In addition, if the corresponding INT_CONx.OVVxx bit is enabled, an over-voltage interrupt is triggered as soon as the over-voltage alarm is de-asserted. 11_B Over-voltage alarm enabled. In addition, if the corresponding INT_CONx.OVVxx bit is enabled, an over-voltage interrupt is triggered as soon as either the corresponding VMONS.VxxSTAT.OV status bit is set or the over-voltage alarm is de-asserted.
FILTER	29:26	rw	**Secondary ADC supply filter** 0_H Each conversion result is compared with threshold to generate alarm F_H A spike filter of consecutive 16 ADC results are used to generate alarm to SMU.
0	31:30	r	**Reserved**

Table 1598 Reset values of PMS_VMONS_VDDPHPHY2CON

Reset type	Reset value	Note
LVDPMS2 Reset	$1059\ 4231_H$	
Cold PORST	$1059\ 4231_H$	

36.8.4.84 VDDPHPHY2 secondary monitor status register

PMS_VMONS_VDDPHPHY2STAT Offset address: $91B0_H$

VDDPHPHY2 secondary monitor status register Reset values see: Table 1599

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ACTIVITY						0									
rh						r									

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			UV	OV	RESULT										
r			rh	rh	rh										

Field	Bits	Type	Description
RESULT	10:0	rh	**Secondary ADC conversion result (unfiltered)** This bit field contains the last conversion result of the ADC measurement of the secondary voltage monitor. The LSB is documented for each supply rail in the user manual (secondary voltage monitors section).
OV	11	rh	**Secondary monitor over-voltage event flag after filtering**
UV	12	rh	**Secondary monitor under-voltage event flag after filtering**
ACTIVITY	31:26	rh	**Secondary monitor activity counter** This bit field cumulatively counts the end of conversion signals in a single Secondary Monitor Background Scan over all channels and respective filter configurations of the secondary monitor ADC which monitors this voltage rail. The total number of conversions ConvTot = ∑ [ChX* ChXFIL]. The counter is reset to 0 on a ConvTot overflow.
0	25:13	r	**Reserved**

Table 1599 Reset values of PMS_VMONS_VDDPHPHY2STAT

Reset type	Reset value	Note
LVDPMS2 Reset	$0000\ 0000_H$	
Cold PORST	$0000\ 0000_H$	

36.8.4.85 VDDHSIF secondary monitor configuration register

PMS_VMONS_VDDHSIFCON Offset address: $91B4_H$

VDDHSIF secondary monitor configuration register Reset values see: Table 1600

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		FILTER				OVMOD		OVVAL							
r		rw				rw		rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OVVAL			UVMOD		UVVAL										
rw			rw		rw										

Field	Bits	Type	Description
UVVAL	10:0	rw	**Secondary monitor under-voltage threshold** This field defines the under-voltage threshold level of the supply monitor. The LSB is documented for each supply rail in the user manual (secondary voltage monitors section).
UVMOD	12:11	rw	**Under-voltage monitoring mode** 00_B Under-voltage monitoring inactive. 01_B Under-voltage alarm enabled. In addition, if the corresponding INT_CONx.UVVxx bit is enabled, an under-voltage interrupt is triggered as soon as the corresponding VMONS.VxxSTAT.UV status bit is set. 10_B Under-voltage alarm enabled. In addition, if the corresponding INT_CONx.UVVxx bit is enabled, an under-voltage interrupt is triggered as soon as the under-voltage alarm is de-asserted. 11_B Under-voltage alarm enabled. In addition, if the corresponding INT_CONx.UVVxx bit is enabled, an under-voltage interrupt is triggered as soon as either the corresponding VMONS.VxxSTAT.UV status bit is set or the under-voltage alarm is de-asserted.
OVVAL	23:13	rw	**Secondary monitor over-voltage threshold** This field defines the over-voltage threshold level of the supply monitor. The LSB is documented for each supply rail in the user manual (secondary voltage monitors section).
OVMOD	25:24	rw	**Over-voltage monitoring mode** 00_B Over-voltage monitoring inactive. 01_B Over-voltage alarm enabled. In addition, if the corresponding INT_CONx.OVVxx bit is enabled, an over-voltage interrupt is triggered as soon as the corresponding VMONS.VxxSTAT.OV status bit is set. 10_B Over-voltage alarm enabled. In addition, if the corresponding INT_CONx.OVVxx bit is enabled, an over-voltage interrupt is triggered as soon as the over-voltage alarm is de-asserted. 11_B Over-voltage alarm enabled. In addition, if the corresponding INT_CONx.OVVxx bit is enabled, an over-voltage interrupt is triggered as soon as either the corresponding VMONS.VxxSTAT.OV status bit is set or the over-voltage alarm is de-asserted.
FILTER	29:26	rw	**Secondary ADC supply filter** 0_H Each conversion result is compared with threshold to generate alarm F_H A spike filter of consecutive 16 ADC results are used to generate alarm to SMU.
0	31:30	r	**Reserved**

Table 1600 Reset values of PMS_VMONS_VDDHSIFCON

Reset type	Reset value	Note
LVDPMS2 Reset	10A3 A23F$_H$	
Cold PORST	10A3 A23F$_H$	

36.8.4.86 VDDHSIF secondary monitor status register

PMS_VMONS_VDDHSIFSTAT Offset address: $91B8_H$

VDDHSIF secondary monitor status register Reset values see: Table 1601

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ACTIVITY						0									
rh						r									

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			UV	OV	RESULT										
r			rh	rh	rh										

Field	Bits	Type	Description
RESULT	10:0	rh	**Secondary ADC conversion result (unfiltered)** This bit field contains the last conversion result of the ADC measurement of the secondary voltage monitor. The LSB is documented for each supply rail in the user manual (secondary voltage monitors section).
OV	11	rh	**Secondary monitor over-voltage event flag after filtering**
UV	12	rh	**Secondary monitor under-voltage event flag after filtering**
ACTIVITY	31:26	rh	**Secondary monitor activity counter** This bit field cumulatively counts the end of conversion signals in a single Secondary Monitor Background Scan over all channels and respective filter configurations of the secondary monitor ADC which monitors this voltage rail. The total number of conversions ConvTot = ∑ [ChX* ChXFIL]. The counter is reset to 0 on a ConvTot overflow.
0	25:13	r	**Reserved**

Table 1601 Reset values of PMS_VMONS_VDDHSIFSTAT

Reset type	Reset value	Note
LVDPMS2 Reset	$0000\ 0000_H$	
Cold PORST	$0000\ 0000_H$	

36.8.4.87 VDDPHY0 secondary monitor configuration register

PMS_VMONS_VDDPHY0CON Offset address: $91C4_H$

VDDPHY0 secondary monitor configuration register Reset values see: Table 1602

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		FILTER				OVMOD		OVVAL							
r		rw				rw		rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OVVAL			UVMOD		UVVAL										
rw			rw		rw										

Field	Bits	Type	Description
UVVAL	10:0	rw	**Secondary monitor under-voltage threshold** This field defines the under-voltage threshold level of the supply monitor. The LSB is documented for each supply rail in the user manual (secondary voltage monitors section).
UVMOD	12:11	rw	**Under-voltage monitoring mode** 00_B Under-voltage monitoring inactive. 01_B Under-voltage alarm enabled. In addition, if the corresponding INT_CONx.UVVxx bit is enabled, an under-voltage interrupt is triggered as soon as the corresponding VMONS.VxxSTAT.UV status bit is set. 10_B Under-voltage alarm enabled. In addition, if the corresponding INT_CONx.UVVxx bit is enabled, an under-voltage interrupt is triggered as soon as the under-voltage alarm is de-asserted. 11_B Under-voltage alarm enabled. In addition, if the corresponding INT_CONx.UVVxx bit is enabled, an under-voltage interrupt is triggered as soon as either the corresponding VMONS.VxxSTAT.UV status bit is set or the under-voltage alarm is de-asserted.
OVVAL	23:13	rw	**Secondary monitor over-voltage threshold** This field defines the over-voltage threshold level of the supply monitor. The LSB is documented for each supply rail in the user manual (secondary voltage monitors section).
OVMOD	25:24	rw	**Over-voltage monitoring mode** 00_B Over-voltage monitoring inactive. 01_B Over-voltage alarm enabled. In addition, if the corresponding INT_CONx.OVVxx bit is enabled, an over-voltage interrupt is triggered as soon as the corresponding VMONS.VxxSTAT.OV status bit is set. 10_B Over-voltage alarm enabled. In addition, if the corresponding INT_CONx.OVVxx bit is enabled, an over-voltage interrupt is triggered as soon as the over-voltage alarm is de-asserted. 11_B Over-voltage alarm enabled. In addition, if the corresponding INT_CONx.OVVxx bit is enabled, an over-voltage interrupt is triggered as soon as either the corresponding VMONS.VxxSTAT.OV status bit is set or the over-voltage alarm is de-asserted.
FILTER	29:26	rw	**Secondary ADC supply filter** 0_H Each conversion result is compared with threshold to generate alarm F_H A spike filter of consecutive 16 ADC results are used to generate alarm to SMU.
0	31:30	r	**Reserved**

Table 1602 Reset values of PMS_VMONS_VDDPHY0CON

Reset type	Reset value	Note
LVDPMS2 Reset	10F1 05DD$_H$	
Cold PORST	10F1 05DD$_H$	

36.8.4.88 VDDPHY0 secondary monitor status register

PMS_VMONS_VDDPHY0STAT Offset address: $91C8_H$

VDDPHY0 secondary monitor status register Reset values see: Table 1603

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ACTIVITY						0									
rh						r									

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			UV	OV	RESULT										
r			rh	rh	rh										

Field	Bits	Type	Description
RESULT	10:0	rh	**Secondary ADC conversion result (unfiltered)** This bit field contains the last conversion result of the ADC measurement of the secondary voltage monitor. The LSB is documented for each supply rail in the user manual (secondary voltage monitors section).
OV	11	rh	**Secondary monitor over-voltage event flag after filtering**
UV	12	rh	**Secondary monitor under-voltage event flag after filtering**
ACTIVITY	31:26	rh	**Secondary monitor activity counter** This bit field cumulatively counts the end of conversion signals in a single Secondary Monitor Background Scan over all channels and respective filter configurations of the secondary monitor ADC which monitors this voltage rail. The total number of conversions ConvTot = ∑ [ChX* ChXFIL]. The counter is reset to 0 on a ConvTot overflow.
0	25:13	r	**Reserved**

Table 1603 Reset values of PMS_VMONS_VDDPHY0STAT

Reset type	Reset value	Note
LVDPMS2 Reset	$0000\ 0000_H$	
Cold PORST	$0000\ 0000_H$	

36.8.4.89 VDDPHY1 secondary monitor configuration register

PMS_VMONS_VDDPHY1CON Offset address: $91CC_H$

VDDPHY1 secondary monitor configuration register Reset values see: Table 1604

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		FILTER				OVMOD		OVVAL							
r		rw				rw		rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OVVAL			UVMOD		UVVAL										
rw			rw		rw										

Field	Bits	Type	Description
UVVAL	10:0	rw	**Secondary monitor under-voltage threshold** This field defines the under-voltage threshold level of the supply monitor. The LSB is documented for each supply rail in the user manual (secondary voltage monitors section).
UVMOD	12:11	rw	**Under-voltage monitoring mode** 00_B Under-voltage monitoring inactive. 01_B Under-voltage alarm enabled. In addition, if the corresponding INT_CONx.UVVxx bit is enabled, an under-voltage interrupt is triggered as soon as the corresponding VMONS.VxxSTAT.UV status bit is set. 10_B Under-voltage alarm enabled. In addition, if the corresponding INT_CONx.UVVxx bit is enabled, an under-voltage interrupt is triggered as soon as the under-voltage alarm is de-asserted. 11_B Under-voltage alarm enabled. In addition, if the corresponding INT_CONx.UVVxx bit is enabled, an under-voltage interrupt is triggered as soon as either the corresponding VMONS.VxxSTAT.UV status bit is set or the under-voltage alarm is de-asserted.
OVVAL	23:13	rw	**Secondary monitor over-voltage threshold** This field defines the over-voltage threshold level of the supply monitor. The LSB is documented for each supply rail in the user manual (secondary voltage monitors section).
OVMOD	25:24	rw	**Over-voltage monitoring mode** 00_B Over-voltage monitoring inactive. 01_B Over-voltage alarm enabled. In addition, if the corresponding INT_CONx.OVVxx bit is enabled, an over-voltage interrupt is triggered as soon as the corresponding VMONS.VxxSTAT.OV status bit is set. 10_B Over-voltage alarm enabled. In addition, if the corresponding INT_CONx.OVVxx bit is enabled, an over-voltage interrupt is triggered as soon as the over-voltage alarm is de-asserted. 11_B Over-voltage alarm enabled. In addition, if the corresponding INT_CONx.OVVxx bit is enabled, an over-voltage interrupt is triggered as soon as either the corresponding VMONS.VxxSTAT.OV status bit is set or the over-voltage alarm is de-asserted.
FILTER	29:26	rw	**Secondary ADC supply filter** 0_H Each conversion result is compared with threshold to generate alarm F_H A spike filter of consecutive 16 ADC results are used to generate alarm to SMU.
0	31:30	r	**Reserved**

Table 1604 Reset values of PMS_VMONS_VDDPHY1CON

Reset type	Reset value	Note
LVDPMS2 Reset	10F1 05DD$_H$	
Cold PORST	10F1 05DD$_H$	

36.8.4.90 VDDPHY1 secondary monitor status register

PMS_VMONS_VDDPHY1STAT Offset address: $91D0_H$

VDDPHY1 secondary monitor status register Reset values see: Table 1605

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ACTIVITY						0									
rh						r									

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			UV	OV	RESULT										
r			rh	rh	rh										

Field	Bits	Type	Description
RESULT	10:0	rh	**Secondary ADC conversion result (unfiltered)** This bit field contains the last conversion result of the ADC measurement of the secondary voltage monitor. The LSB is documented for each supply rail in the user manual (secondary voltage monitors section).
OV	11	rh	**Secondary monitor over-voltage event flag after filtering**
UV	12	rh	**Secondary monitor under-voltage event flag after filtering**
ACTIVITY	31:26	rh	**Secondary monitor activity counter** This bit field cumulatively counts the end of conversion signals in a single Secondary Monitor Background Scan over all channels and respective filter configurations of the secondary monitor ADC which monitors this voltage rail. The total number of conversions ConvTot = ∑ [ChX* ChXFIL]. The counter is reset to 0 on a ConvTot overflow.
0	25:13	r	**Reserved**

Table 1605 Reset values of PMS_VMONS_VDDPHY1STAT

Reset type	Reset value	Note
LVDPMS2 Reset	$0000\ 0000_H$	
Cold PORST	$0000\ 0000_H$	

36.8.4.91 VDDPHY2 secondary monitor configuration register

PMS_VMONS_VDDPHY2CON Offset address: $91D4_H$

VDDPHY2 secondary monitor configuration register Reset values see: Table 1606

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		FILTER				OVMOD		OVVAL							
r		rw				rw		rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OVVAL			UVMOD		UVVAL										
rw			rw		rw										

Field	Bits	Type	Description
UVVAL	10:0	rw	**Secondary monitor under-voltage threshold** This field defines the under-voltage threshold level of the supply monitor. The LSB is documented for each supply rail in the user manual (secondary voltage monitors section).
UVMOD	12:11	rw	**Under-voltage monitoring mode** 00_B Under-voltage monitoring inactive. 01_B Under-voltage alarm enabled. In addition, if the corresponding INT_CONx.UVVxx bit is enabled, an under-voltage interrupt is triggered as soon as the corresponding VMONS.VxxSTAT.UV status bit is set. 10_B Under-voltage alarm enabled. In addition, if the corresponding INT_CONx.UVVxx bit is enabled, an under-voltage interrupt is triggered as soon as the under-voltage alarm is de-asserted. 11_B Under-voltage alarm enabled. In addition, if the corresponding INT_CONx.UVVxx bit is enabled, an under-voltage interrupt is triggered as soon as either the corresponding VMONS.VxxSTAT.UV status bit is set or the under-voltage alarm is de-asserted.
OVVAL	23:13	rw	**Secondary monitor over-voltage threshold** This field defines the over-voltage threshold level of the supply monitor. The LSB is documented for each supply rail in the user manual (secondary voltage monitors section).
OVMOD	25:24	rw	**Over-voltage monitoring mode** 00_B Over-voltage monitoring inactive. 01_B Over-voltage alarm enabled. In addition, if the corresponding INT_CONx.OVVxx bit is enabled, an over-voltage interrupt is triggered as soon as the corresponding VMONS.VxxSTAT.OV status bit is set. 10_B Over-voltage alarm enabled. In addition, if the corresponding INT_CONx.OVVxx bit is enabled, an over-voltage interrupt is triggered as soon as the over-voltage alarm is de-asserted. 11_B Over-voltage alarm enabled. In addition, if the corresponding INT_CONx.OVVxx bit is enabled, an over-voltage interrupt is triggered as soon as either the corresponding VMONS.VxxSTAT.OV status bit is set or the over-voltage alarm is de-asserted.
FILTER	29:26	rw	**Secondary ADC supply filter** 0_H Each conversion result is compared with threshold to generate alarm F_H A spike filter of consecutive 16 ADC results are used to generate alarm to SMU.
0	31:30	r	**Reserved**

Table 1606 Reset values of PMS_VMONS_VDDPHY2CON

Reset type	Reset value	Note
LVDPMS2 Reset	$10F1\ 05DD_H$	
Cold PORST	$10F1\ 05DD_H$	

36.8.4.92 VDDPHY2 secondary monitor status register

PMS_VMONS_VDDPHY2STAT Offset address: $91D8_H$

VDDPHY2 secondary monitor status register Reset values see: Table 1607

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ACTIVITY						0									
rh						r									

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			UV	OV	RESULT										
r			rh	rh	rh										

Field	Bits	Type	Description
RESULT	10:0	rh	**Secondary ADC conversion result (unfiltered)** This bit field contains the last conversion result of the ADC measurement of the secondary voltage monitor. The LSB is documented for each supply rail in the user manual (secondary voltage monitors section).
OV	11	rh	**Secondary monitor over-voltage event flag after filtering**
UV	12	rh	**Secondary monitor under-voltage event flag after filtering**
ACTIVITY	31:26	rh	**Secondary monitor activity counter** This bit field cumulatively counts the end of conversion signals in a single Secondary Monitor Background Scan over all channels and respective filter configurations of the secondary monitor ADC which monitors this voltage rail. The total number of conversions ConvTot = ∑ [ChX* ChXFIL]. The counter is reset to 0 on a ConvTot overflow.
0	25:13	r	**Reserved**

Table 1607 Reset values of PMS_VMONS_VDDPHY2STAT

Reset type	Reset value	Note
LVDPMS2 Reset	$0000\ 0000_H$	
Cold PORST	$0000\ 0000_H$	

36.8.4.93 VDDLMU0 secondary monitor configuration register

PMS_VMONS_VDDLMU0CON Offset address: $91DC_H$

VDDLMU0 secondary monitor configuration register Reset values see: Table 1608

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		FILTER				OVMOD		OVVAL							
r		rw				rw		rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OVVAL			UVMOD		UVVAL										
rw			rw		rw										

Field	Bits	Type	Description
UVVAL	10:0	rw	**Secondary monitor under-voltage threshold** This field defines the under-voltage threshold level of the supply monitor. The LSB is documented for each supply rail in the user manual (secondary voltage monitors section).
UVMOD	12:11	rw	**Under-voltage monitoring mode** 00_B Under-voltage monitoring inactive. 01_B Under-voltage alarm enabled. In addition, if the corresponding INT_CONx.UVVxx bit is enabled, an under-voltage interrupt is triggered as soon as the corresponding VMONS.VxxSTAT.UV status bit is set. 10_B Under-voltage alarm enabled. In addition, if the corresponding INT_CONx.UVVxx bit is enabled, an under-voltage interrupt is triggered as soon as the under-voltage alarm is de-asserted. 11_B Under-voltage alarm enabled. In addition, if the corresponding INT_CONx.UVVxx bit is enabled, an under-voltage interrupt is triggered as soon as either the corresponding VMONS.VxxSTAT.UV status bit is set or the under-voltage alarm is de-asserted.
OVVAL	23:13	rw	**Secondary monitor over-voltage threshold** This field defines the over-voltage threshold level of the supply monitor. The LSB is documented for each supply rail in the user manual (secondary voltage monitors section).
OVMOD	25:24	rw	**Over-voltage monitoring mode** 00_B Over-voltage monitoring inactive. 01_B Over-voltage alarm enabled. In addition, if the corresponding INT_CONx.OVVxx bit is enabled, an over-voltage interrupt is triggered as soon as the corresponding VMONS.VxxSTAT.OV status bit is set. 10_B Over-voltage alarm enabled. In addition, if the corresponding INT_CONx.OVVxx bit is enabled, an over-voltage interrupt is triggered as soon as the over-voltage alarm is de-asserted. 11_B Over-voltage alarm enabled. In addition, if the corresponding INT_CONx.OVVxx bit is enabled, an over-voltage interrupt is triggered as soon as either the corresponding VMONS.VxxSTAT.OV status bit is set or the over-voltage alarm is de-asserted.
FILTER	29:26	rw	**Secondary ADC supply filter** 0_H Each conversion result is compared with threshold to generate alarm F_H A spike filter of consecutive 16 ADC results are used to generate alarm to SMU.
0	31:30	r	**Reserved**

Table 1608 Reset values of PMS_VMONS_VDDLMU0CON

Reset type	Reset value	Note
LVDPMS2 Reset	$10F5\ 85DD_H$	
Cold PORST	$10F5\ 85DD_H$	

36.8.4.94 VDDLMU0 secondary monitor status register

PMS_VMONS_VDDLMU0STAT Offset address: $91E0_H$

VDDLMU0 secondary monitor status register Reset values see: Table 1609

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ACTIVITY						0									
rh						r									

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			UV	OV	RESULT										
r			rh	rh	rh										

Field	Bits	Type	Description
RESULT	10:0	rh	**Secondary ADC conversion result (unfiltered)** This bit field contains the last conversion result of the ADC measurement of the secondary voltage monitor. The LSB is documented for each supply rail in the user manual (secondary voltage monitors section).
OV	11	rh	**Secondary monitor over-voltage event flag after filtering**
UV	12	rh	**Secondary monitor under-voltage event flag after filtering**
ACTIVITY	31:26	rh	**Secondary monitor activity counter** This bit field cumulatively counts the end of conversion signals in a single Secondary Monitor Background Scan over all channels and respective filter configurations of the secondary monitor ADC which monitors this voltage rail. The total number of conversions ConvTot = ∑ [ChX* ChXFIL]. The counter is reset to 0 on a ConvTot overflow.
0	25:13	r	**Reserved**

Table 1609 Reset values of PMS_VMONS_VDDLMU0STAT

Reset type	Reset value	Note
LVDPMS2 Reset	$0000\ 0000_H$	
Cold PORST	$0000\ 0000_H$	

36.8.4.95 VDDLMU1 secondary monitor configuration register

PMS_VMONS_VDDLMU1CON Offset address: $91E4_H$

VDDLMU1 secondary monitor configuration register Reset values see: Table 1610

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		FILTER				OVMOD		OVVAL							
r		rw				rw		rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OVVAL			UVMOD		UVVAL										
rw			rw		rw										

Field	Bits	Type	Description
UVVAL	10:0	rw	**Secondary monitor under-voltage threshold** This field defines the under-voltage threshold level of the supply monitor. The LSB is documented for each supply rail in the user manual (secondary voltage monitors section).
UVMOD	12:11	rw	**Under-voltage monitoring mode** 00_B Under-voltage monitoring inactive. 01_B Under-voltage alarm enabled. In addition, if the corresponding INT_CONx.UVVxx bit is enabled, an under-voltage interrupt is triggered as soon as the corresponding VMONS.VxxSTAT.UV status bit is set. 10_B Under-voltage alarm enabled. In addition, if the corresponding INT_CONx.UVVxx bit is enabled, an under-voltage interrupt is triggered as soon as the under-voltage alarm is de-asserted. 11_B Under-voltage alarm enabled. In addition, if the corresponding INT_CONx.UVVxx bit is enabled, an under-voltage interrupt is triggered as soon as either the corresponding VMONS.VxxSTAT.UV status bit is set or the under-voltage alarm is de-asserted.
OVVAL	23:13	rw	**Secondary monitor over-voltage threshold** This field defines the over-voltage threshold level of the supply monitor. The LSB is documented for each supply rail in the user manual (secondary voltage monitors section).
OVMOD	25:24	rw	**Over-voltage monitoring mode** 00_B Over-voltage monitoring inactive. 01_B Over-voltage alarm enabled. In addition, if the corresponding INT_CONx.OVVxx bit is enabled, an over-voltage interrupt is triggered as soon as the corresponding VMONS.VxxSTAT.OV status bit is set. 10_B Over-voltage alarm enabled. In addition, if the corresponding INT_CONx.OVVxx bit is enabled, an over-voltage interrupt is triggered as soon as the over-voltage alarm is de-asserted. 11_B Over-voltage alarm enabled. In addition, if the corresponding INT_CONx.OVVxx bit is enabled, an over-voltage interrupt is triggered as soon as either the corresponding VMONS.VxxSTAT.OV status bit is set or the over-voltage alarm is de-asserted.
FILTER	29:26	rw	**Secondary ADC supply filter** 0_H Each conversion result is compared with threshold to generate alarm F_H A spike filter of consecutive 16 ADC results are used to generate alarm to SMU.
0	31:30	r	**Reserved**

Table 1610 Reset values of PMS_VMONS_VDDLMU1CON

Reset type	Reset value	Note
LVDPMS2 Reset	10F5 85DD$_H$	
Cold PORST	10F5 85DD$_H$	

36.8.4.96 VDDLMU1 secondary monitor status register

PMS_VMONS_VDDLMU1STAT Offset address: $91E8_H$

VDDLMU1 secondary monitor status register Reset values see: Table 1611

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ACTIVITY						0									
rh						r									

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			UV	OV	RESULT										
r			rh	rh	rh										

Field	Bits	Type	Description
RESULT	10:0	rh	**Secondary ADC conversion result (unfiltered)** This bit field contains the last conversion result of the ADC measurement of the secondary voltage monitor. The LSB is documented for each supply rail in the user manual (secondary voltage monitors section).
OV	11	rh	**Secondary monitor over-voltage event flag after filtering**
UV	12	rh	**Secondary monitor under-voltage event flag after filtering**
ACTIVITY	31:26	rh	**Secondary monitor activity counter** This bit field cumulatively counts the end of conversion signals in a single Secondary Monitor Background Scan over all channels and respective filter configurations of the secondary monitor ADC which monitors this voltage rail. The total number of conversions ConvTot = ∑ [ChX* ChXFIL]. The counter is reset to 0 on a ConvTot overflow.
0	25:13	r	**Reserved**

Table 1611 Reset values of PMS_VMONS_VDDLMU1STAT

Reset type	Reset value	Note
LVDPMS2 Reset	$0000\ 0000_H$	
Cold PORST	$0000\ 0000_H$	

36.8.4.97 VDDPPU secondary monitor configuration register

PMS_VMONS_VDDPPUCON Offset address: $91EC_H$

VDDPPU secondary monitor configuration register Reset values see: Table 1612

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		FILTER				OVMOD		OVVAL							
r		rw				rw		rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OVVAL			UVMOD		UVVAL										
rw			rw		rw										

Field	Bits	Type	Description
UVVAL	10:0	rw	**Secondary monitor under-voltage threshold** This field defines the under-voltage threshold level of the supply monitor. The LSB is documented for each supply rail in the user manual (secondary voltage monitors section).
UVMOD	12:11	rw	**Under-voltage monitoring mode** 00_B Under-voltage monitoring inactive. 01_B Under-voltage alarm enabled. In addition, if the corresponding INT_CONx.UVVxx bit is enabled, an under-voltage interrupt is triggered as soon as the corresponding VMONS.VxxSTAT.UV status bit is set. 10_B Under-voltage alarm enabled. In addition, if the corresponding INT_CONx.UVVxx bit is enabled, an under-voltage interrupt is triggered as soon as the under-voltage alarm is de-asserted. 11_B Under-voltage alarm enabled. In addition, if the corresponding INT_CONx.UVVxx bit is enabled, an under-voltage interrupt is triggered as soon as either the corresponding VMONS.VxxSTAT.UV status bit is set or the under-voltage alarm is de-asserted.
OVVAL	23:13	rw	**Secondary monitor over-voltage threshold** This field defines the over-voltage threshold level of the supply monitor. The LSB is documented for each supply rail in the user manual (secondary voltage monitors section).
OVMOD	25:24	rw	**Over-voltage monitoring mode** 00_B Over-voltage monitoring inactive. 01_B Over-voltage alarm enabled. In addition, if the corresponding INT_CONx.OVVxx bit is enabled, an over-voltage interrupt is triggered as soon as the corresponding VMONS.VxxSTAT.OV status bit is set. 10_B Over-voltage alarm enabled. In addition, if the corresponding INT_CONx.OVVxx bit is enabled, an over-voltage interrupt is triggered as soon as the over-voltage alarm is de-asserted. 11_B Over-voltage alarm enabled. In addition, if the corresponding INT_CONx.OVVxx bit is enabled, an over-voltage interrupt is triggered as soon as either the corresponding VMONS.VxxSTAT.OV status bit is set or the over-voltage alarm is de-asserted.
FILTER	29:26	rw	**Secondary ADC supply filter** 0_H Each conversion result is compared with threshold to generate alarm F_H A spike filter of consecutive 16 ADC results are used to generate alarm to SMU.
0	31:30	r	**Reserved**

Table 1612 Reset values of PMS_VMONS_VDDPPUCON

Reset type	Reset value	Note
LVDPMS2 Reset	10F5 85DD$_H$	
Cold PORST	10F5 85DD$_H$	

36.8.4.98 VDDPPU secondary monitor status register

PMS_VMONS_VDDPPUSTAT Offset address: $91F0_H$

VDDPPU secondary monitor status register Reset values see: Table 1613

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ACTIVITY						0									
rh						r									

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			UV	OV	RESULT										
r			rh	rh	rh										

Field	Bits	Type	Description
RESULT	10:0	rh	**Secondary ADC conversion result (unfiltered)** This bit field contains the last conversion result of the ADC measurement of the secondary voltage monitor. The LSB is documented for each supply rail in the user manual (secondary voltage monitors section).
OV	11	rh	**Secondary monitor over-voltage event flag after filtering**
UV	12	rh	**Secondary monitor under-voltage event flag after filtering**
ACTIVITY	31:26	rh	**Secondary monitor activity counter** This bit field cumulatively counts the end of conversion signals in a single Secondary Monitor Background Scan over all channels and respective filter configurations of the secondary monitor ADC which monitors this voltage rail. The total number of conversions ConvTot = ∑ [ChX* ChXFIL]. The counter is reset to 0 on a ConvTot overflow.
0	25:13	r	**Reserved**

Table 1613 Reset values of PMS_VMONS_VDDPPUSTAT

Reset type	Reset value	Note
LVDPMS2 Reset	$0000\ 0000_H$	
Cold PORST	$0000\ 0000_H$	

36.8.4.99 VDDEXTDC secondary monitor configuration register

PMS_VMONS_VDDEXTDCCON Offset address: $91F4_H$

VDDEXTDC secondary monitor configuration register Reset values see: Table 1614

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		FILTER				OVMOD		OVVAL							
r		rw				rw		rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OVVAL			UVMOD		UVVAL										
rw			rw		rw										

Field	Bits	Type	Description
UVVAL	10:0	rw	**Secondary monitor under-voltage threshold** This field defines the under-voltage threshold level of the supply monitor. The LSB is documented for each supply rail in the user manual (secondary voltage monitors section).
UVMOD	12:11	rw	**Under-voltage monitoring mode** 00_B Under-voltage monitoring inactive. 01_B Under-voltage alarm enabled. In addition, if the corresponding INT_CONx.UVVxx bit is enabled, an under-voltage interrupt is triggered as soon as the corresponding VMONS.VxxSTAT.UV status bit is set. 10_B Under-voltage alarm enabled. In addition, if the corresponding INT_CONx.UVVxx bit is enabled, an under-voltage interrupt is triggered as soon as the under-voltage alarm is de-asserted. 11_B Under-voltage alarm enabled. In addition, if the corresponding INT_CONx.UVVxx bit is enabled, an under-voltage interrupt is triggered as soon as either the corresponding VMONS.VxxSTAT.UV status bit is set or the under-voltage alarm is de-asserted.
OVVAL	23:13	rw	**Secondary monitor over-voltage threshold** This field defines the over-voltage threshold level of the supply monitor. The LSB is documented for each supply rail in the user manual (secondary voltage monitors section).
OVMOD	25:24	rw	**Over-voltage monitoring mode** 00_B Over-voltage monitoring inactive. 01_B Over-voltage alarm enabled. In addition, if the corresponding INT_CONx.OVVxx bit is enabled, an over-voltage interrupt is triggered as soon as the corresponding VMONS.VxxSTAT.OV status bit is set. 10_B Over-voltage alarm enabled. In addition, if the corresponding INT_CONx.OVVxx bit is enabled, an over-voltage interrupt is triggered as soon as the over-voltage alarm is de-asserted. 11_B Over-voltage alarm enabled. In addition, if the corresponding INT_CONx.OVVxx bit is enabled, an over-voltage interrupt is triggered as soon as either the corresponding VMONS.VxxSTAT.OV status bit is set or the over-voltage alarm is de-asserted.
FILTER	29:26	rw	**Secondary ADC supply filter** 0_H Each conversion result is compared with threshold to generate alarm F_H A spike filter of consecutive 16 ADC results are used to generate alarm to SMU.
0	31:30	r	**Reserved**

Table 1614 Reset values of PMS_VMONS_VDDEXTDCCON

Reset type	Reset value	Note
LVDPMS2 Reset	$3C3E\ 0101_H$	

(table continues...)

Table 1614 (continued) Reset values of PMS_VMONS_VDDEXTDCCON

Reset type	Reset value	Note
LVDPMS2 Reset for VDDEXTDC high-voltage configuration (5.5V<=VDDEXTDC(Typ)<=6V5)	3C4E E1AD$_H$	Reset value with HWCFG[2:1]=11, 5V5<=VDDEXTDC<=6V5
Cold PORST for VDDEXTDC high-voltage configuration (5.5V<=VDDEXTDC(Typ)<=6V5)	3C4E E1AD$_H$	Reset value with HWCFG[2:1]=11, 5V5<=VDDEXTDC<=6V5
Cold PORST	3C3E 0101$_H$	

36.8.4.100 VDDEXTDC secondary monitor status register

PMS_VMONS_VDDEXTDCSTAT Offset address: 91F8$_H$

VDDEXTDC secondary monitor status register Reset values see: Table 1615

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ACTIVITY						0									
rh						r									

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			UV	OV	RESULT										
r			rh	rh	rh										

Field	Bits	Type	Description
RESULT	10:0	rh	**Secondary ADC conversion result (unfiltered)** This bit field contains the last conversion result of the ADC measurement of the secondary voltage monitor. The LSB is documented for each supply rail in the user manual (secondary voltage monitors section).
OV	11	rh	**Secondary monitor over-voltage event flag after filtering**
UV	12	rh	**Secondary monitor under-voltage event flag after filtering**
ACTIVITY	31:26	rh	**Secondary monitor activity counter** This bit field cumulatively counts the end of conversion signals in a single Secondary Monitor Background Scan over all channels and respective filter configurations of the secondary monitor ADC which monitors this voltage rail. The total number of conversions ConvTot = ∑ [ChX* ChXFIL]. The counter is reset to 0 on a ConvTot overflow.
0	25:13	r	**Reserved**

Table 1615 Reset values of PMS_VMONS_VDDEXTDCSTAT

Reset type	Reset value	Note
LVDPMS2 Reset	0000 0000$_H$	
Cold PORST	0000 0000$_H$	

36.8.4.101 Secondary monitor level selection register

PMS_VMONS_LVLSEL Offset address: 91FC$_H$

Secondary monitor level selection register Reset values see: Table 1616

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0					NVMMINSEL									VDDEXTLVLSEL	
r					r									rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VDDEXTLVLSEL							VDDMLVLSEL								
rw							rw								

Field	Bits	Type	Description
VDDMLVLSEL	8:0	rw	**VDDM level select** This field defines the under-voltage monitoring threshold level required by ADC (TMADC/DSADC) modules to differentiate between 5 V or 3.3 V VDDM supply level to adjust analog behavior to the actual voltage level. Threshold = VxxLVLSEL * LSB LSB = 4 * 2.83 mV
VDDEXTLVLSEL	17:9	rw	**VDDEXT level select** This field defines the under-voltage monitoring threshold level required by VDDEXT-supplied modules to differentiate between 5 V or 3.3 V VDDEXT supply level Threshold = VxxLVLSEL * LSB LSB = 4 * 2.83 mV
NVMMINSEL	26:18	r	**Minimum voltage level for stable NVM operation** This field defines the minimum V_{DDEXT} voltage threshold level required for NVM stable operation Threshold = NVMMINSEL * LSB LSB = 4 * 2.83 mV
0	31:27	r	**Reserved** Read as 0; should be written with 0.

Table 1616 Reset values of PMS_VMONS_LVLSEL

Reset type	Reset value	Note
LVDPMS2 Reset	0402 DF4F$_H$	
Cold PORST	0402 DF4F$_H$	

(table continues...)

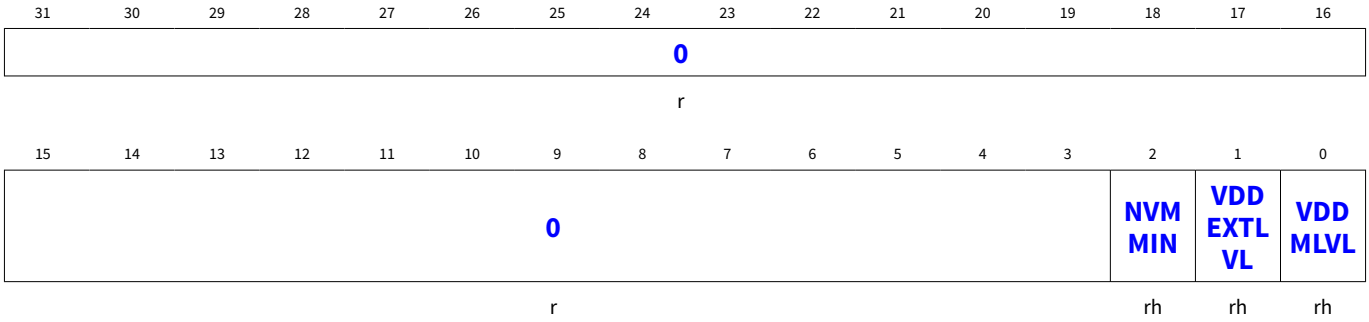
Table 1616 (continued) Reset values of PMS_VMONS_LVLSEL

Reset type	Reset value	Note
After Boot-FW Value	0402 DF4F$_H$	

36.8.4.102 Secondary monitor level selection status register

PMS_VMONS_LVLSTAT Offset address: 9200$_H$

Secondary monitor level selection status register Reset values see: Table 1617



Field	Bits	Type	Description
VDDMLVL	0	rh	**VDDM level status** 0_B V_{DDM} external supply is above the threshold. 1_B V_{DDM} external supply is below or equal to the threshold.
VDDEXTLVL	1	rh	**VDDEXT level status** 0_B V_{DDEXT} external supply is above the threshold. 1_B V_{DDEXT} external supply is below or equal to the threshold.
NVMMIN	2	rh	**VDDEXT voltage level status for NVM operation** 0_B V_{DDEXT} external supply is above the minimum threshold for stable NVM operation. 1_B V_{DDEXT} external supply is below or equal to the minimum threshold for stable NVM operation.
0	31:3	r	**Reserved** Read as 0.

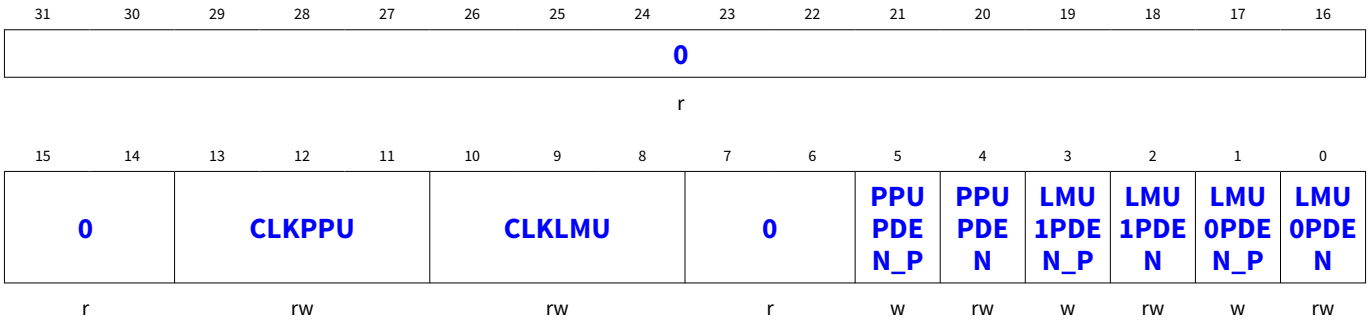
Table 1617 Reset values of PMS_VMONS_LVLSTAT

Reset type	Reset value	Note
LVDPMS2 Reset	0000 0000$_H$	
Cold PORST	XXXX XXXX XXXX XXXX XXXX XXXX XXXX X0––$_B$	

36.8.4.103 Power domain configuration register 0

PMS_PD_CON0 Offset address: 9260$_H$

Power domain configuration register 0 Reset values see: Table 1619



Field	Bits	Type	Description
LMU0PDEN	0	rw	**Enable to power down LMU0 power domain (power down enable)** 0_B Power-down of the power domain is disabled (domain is always on). 1_B Power-down of the power domain is enabled (domain can be powered down if requested).
LMU0PDEN_P	1	w	**Bit protection for LMU0PDEN** Writing 1 into this bit enables that the protected bit can be changed in this write operation. This bit is read as zero.
LMU1PDEN	2	rw	**Enable to power down LMU1 power domain (power down enable)** 0_B Power-down of the power domain is disabled (domain is always on). 1_B Power-down of the power domain is enabled (domain can be powered down if requested).
LMU1PDEN_P	3	w	**Bit protection for LMU1PDEN** Writing 1 into this bit enables that the protected bit can be changed in this write operation. This bit is read as zero.
PPUPDEN	4	rw	**Enable to power down PPU power domain (power down enable)** 0_B Power-down of the power domain is disabled (domain is always on). 1_B Power-down of the power domain is enabled (domain can be powered down if requested).
PPUPDEN_P	5	w	**Bit protection for PPUPDEN** Writing 1 into this bit enables that the protected bit can be changed in this write operation. This bit is read as zero.
CLKLMU	10:8	rw	**LMU power switch rate clock divider** Switch rate clock divider (0-7 range): CLK / 2^CLKLMU
CLKPPU	13:11	rw	**PPU power switch rate clock divider** Switch rate clock divider (0-7 range): CLK / 2^CLKPPU.
0	7:6, 31:14	r	**Reserved** Read as 0; should be written with 0.

Table 1618 Access mode restrictions of PMS_PD_CON0 sorted by descending priority

Mode name	Access mode		Description
write 1 to .**LMU0PDEN_P** and **PD_CON1.LMU0OFFREQ** == 0	rw	LMU0PDEN	
write 1 to .**LMU1PDEN_P** and **PD_CON1.LMU1OFFREQ** == 0	rw	LMU1PDEN	
write 1 to .**PPUPDEN_P** and **PD_CON1.PPUOFFREQ** == 0	rw	PPUPDEN	
(default)	r	LMU0PDEN, LMU1PDEN, PPUPDEN	

Table 1619 Reset values of PMS_PD_CON0

Reset type	Reset value	Note
LVDPMS2 Reset	0000 0000$_H$	
Power Domain Cold Reset	0000 0000$_H$	

36.8.4.104 Power domain configuration register 1

PMS_PD_CON1 Offset address: 9264$_H$

Power domain configuration register 1 Reset values see: Table 1621

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0										PPUOFFREQ_P	PPUOFFREQ	LMU1OFFREQ_P	LMU1OFFREQ	LMU0OFFREQ_P	LMU0OFFREQ
r										w	rw	w	rw	w	rw

Field	Bits	Type	Description
LMU0OFFREQ	0	rw	**Request LMU0 power domain switch off** 0_B Power domain switch off not requested 1_B Power domain switch off requested

(table continues...)

(continued)

Field	Bits	Type	Description
LMU0OFFREQ_P	1	w	**Bit protection for LMU0OFFREQ** Writing 1 into this bit enables that the protected bit can be changed in this write operation. This bit is read as zero.
LMU1OFFREQ	2	rw	**Request LMU1 power domain switch off** 0_B Power domain switch off not requested 1_B Power domain switch off requested
LMU1OFFREQ_P	3	w	**Bit protection for LMU1OFFREQ** Writing 1 into this bit enables that the protected bit can be changed in this write operation. This bit is read as zero.
PPUOFFREQ	4	rw	**Request PPU power domain switch off** 0_B Power domain switch off not requested 1_B Power domain switch off requested
PPUOFFREQ_P	5	w	**Bit protection for PPUOFFREQ** Writing 1 into this bit enables that the protected bit can be changed in this write operation. This bit is read as zero.
0	31:6	r	**Reserved** Read as 0; should be written with 0.

Table 1620 Access mode restrictions of PMS_PD_CON1 sorted by descending priority

Mode name	Access mode		Description
write 1 to .**LMU0OFFREQ_P** and **PD_CON0.LMU0PDEN** == 1	rw	LMU0OFFREQ	
write 1 to .**LMU1OFFREQ_P** and **PD_CON0.LMU1PDEN** == 1	rw	LMU1OFFREQ	
write 1 to .**PPUOFFREQ_P** and **PD_CON0.PPUPDEN** == 1	rw	PPUOFFREQ	
(default)	r	LMU0OFFREQ, LMU1OFFREQ, PPUOFFREQ	

Table 1621 Reset values of PMS_PD_CON1

Reset type	Reset value	Note
LVDPMS2 Reset	0000 0000_H	

(table continues...)

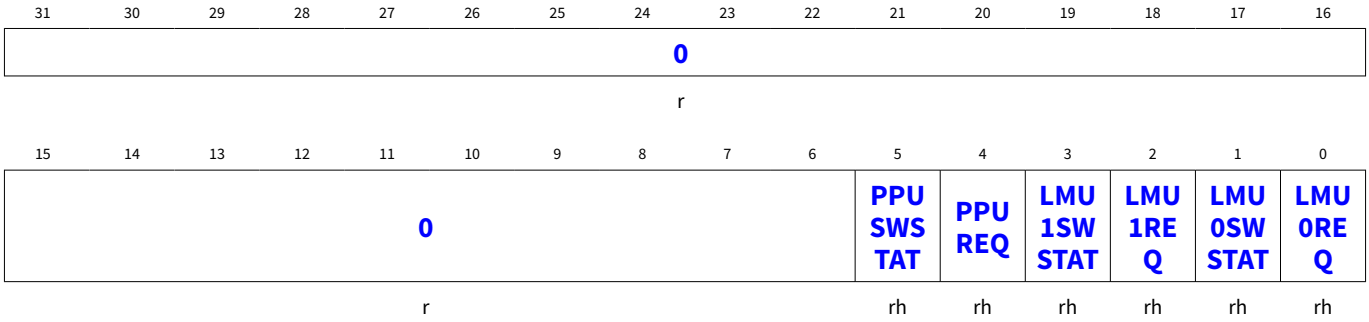
Table 1621 (continued) Reset values of PMS_PD_CON1

Reset type	Reset value	Note
Power Domain Cold Reset	0000 0000$_H$	

36.8.4.105 Power domain status register 0

PMS_PD_STAT0 Offset address: 9268$_H$

Power domain status register 0 Reset values see: Table 1622



Field	Bits	Type	Description
LMU0REQ	0	rh	**LMU0 power domain switch off was requested after last cold PORST** 0_B Power domain switch off not requested 1_B Power domain switch off requested
LMU0SWSTAT	1	rh	**LMU0 switch status** 0_B Power domain is switched on. 1_B Power domain is switched off.
LMU1REQ	2	rh	**LMU1 power domain switch off was requested after last cold PORST** 0_B Power domain switch off not requested 1_B Power domain switch off requested
LMU1SWSTAT	3	rh	**LMU1 switch status** 0_B Power domain is switched on. 1_B Power domain is switched off.
PPUREQ	4	rh	**PPU power domain switch off was requested after last cold PORST** 0_B Power domain switch off not requested 1_B Power domain switch off requested
PPUSWSTAT	5	rh	**PPU switch status** 0_B Power domain is switched on. 1_B Power domain is switched off.
0	31:6	r	**Reserved** Read as 0.

Table 1622 Reset values of PMS_PD_STAT0

Reset type	Reset value	Note
LVDPMS2 Reset	0000 0000$_H$	
Power Domain Cold Reset	0000 0000$_H$	

36.8.4.106 Interrupt configuration register 0

PMS_INT_CON0 Offset address: 926C$_H$

Interrupt configuration register 0 Reset values see: Table 1623

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								Res		UVVDDEXTDC	OVVDDEXTDC	UVVDDPAD	OVVDDPAD	UVVDDSBRAM	OVVDDSBRAM
r								r		rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UVVDDPMS2	OVVDDPMS2	UVVDDPMS1	OVVDDPMS1	UVVDDPMS0	OVVDDPMS0	UVVDD	OVVDD	0		UVVDDM	OVVDDM	UVVDDEXT	OVVDDEXT	UVVDDEVRSB	OVVDDEVRSB
rw	rw	rw	rw	rw	rw	rw	rw	rw		rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
OVVDDEVRSB	0	rw	**OVVDDEVRSB interrupt enable** Interrupt triggered on event as configured in the respective VMONS_VxxCON register. 0_B Interrupt is disabled. 1_B Interrupt is enabled.
UVVDDEVRSB	1	rw	**UVVDDEVRSB interrupt enable** Interrupt triggered on event as configured in the respective VMONS_VxxCON register. 0_B Interrupt is disabled. 1_B Interrupt is enabled.
OVVDDEXT	2	rw	**OVVDDEXT interrupt enable** Interrupt triggered on event as configured in the respective VMONS_VxxCON register. 0_B Interrupt is disabled. 1_B Interrupt is enabled.
UVVDDEXT	3	rw	**UVVDDEXT interrupt enable** Interrupt triggered on event as configured in the respective VMONS_VxxCON register. 0_B Interrupt is disabled. 1_B Interrupt is enabled.

(table continues...)

(continued)

Field	Bits	Type	Description
OVVDDM	4	rw	**OVVDDM interrupt enable** Interrupt triggered on event as configured in the respective VMONS_VxxCON register. 0_B Interrupt is disabled. 1_B Interrupt is enabled.
UVVDDM	5	rw	**UVVDDM interrupt enable** Interrupt triggered on event as configured in the respective VMONS_VxxCON register. 0_B Interrupt is disabled. 1_B Interrupt is enabled.
OVVDD	8	rw	**OVVDD interrupt enable** Interrupt triggered on event as configured in the respective VMONS_VxxCON register. 0_B Interrupt is disabled. 1_B Interrupt is enabled.
UVVDD	9	rw	**UVVDD interrupt enable** Interrupt triggered on event as configured in the respective VMONS_VxxCON register. 0_B Interrupt is disabled. 1_B Interrupt is enabled.
OVVDDPMS0	10	rw	**OVVDDPMS0 interrupt enable** Interrupt triggered on event as configured in the respective VMONS_VxxCON register. 0_B Interrupt is disabled. 1_B Interrupt is enabled.
UVVDDPMS0	11	rw	**UVVDDPMS0 interrupt enable** Interrupt triggered on event as configured in the respective VMONS_VxxCON register. 0_B Interrupt is disabled. 1_B Interrupt is enabled.
OVVDDPMS1	12	rw	**OVVDDPMS1 interrupt enable** Interrupt triggered on event as configured in the respective VMONS_VxxCON register. 0_B Interrupt is disabled. 1_B Interrupt is enabled.
UVVDDPMS1	13	rw	**UVVDDPMS1 interrupt enable** Interrupt triggered on event as configured in the respective VMONS_VxxCON register. 0_B Interrupt is disabled. 1_B Interrupt is enabled.

(table continues...)

(continued)

Field	Bits	Type	Description
OVVDDPMS2	14	rw	**OVVDDPMS2 interrupt enable** Interrupt triggered on event as configured in the respective VMONS_VxxCON register. 0_B Interrupt is disabled. 1_B Interrupt is enabled.
UVVDDPMS2	15	rw	**UVVDDPMS2 interrupt enable** Interrupt triggered on event as configured in the respective VMONS_VxxCON register. 0_B Interrupt is disabled. 1_B Interrupt is enabled.
OVVDDSBRAM	16	rw	**OVVDDSBRAM interrupt enable** Interrupt triggered on event as configured in the respective VMONS_VxxCON register. 0_B Interrupt is disabled. 1_B Interrupt is enabled.
UVVDDSBRAM	17	rw	**UVVDDSBRAM interrupt enable** Interrupt triggered on event as configured in the respective VMONS_VxxCON register. 0_B Interrupt is disabled. 1_B Interrupt is enabled.
OVVDDPAD	18	rw	**OVVDDPAD interrupt enable** Interrupt triggered on event as configured in the respective VMONS_VxxCON register. 0_B Interrupt is disabled. 1_B Interrupt is enabled.
UVVDDPAD	19	rw	**UVVDDPAD interrupt enable** Interrupt triggered on event as configured in the respective VMONS_VxxCON register. 0_B Interrupt is disabled. 1_B Interrupt is enabled.
OVVDDEXTDC	20	rw	**OVVDDEXTDC interrupt enable** Interrupt triggered on event as configured in the respective VMONS_VxxCON register. 0_B Interrupt is disabled. 1_B Interrupt is enabled.
UVVDDEXTDC	21	rw	**UVVDDEXTDC interrupt enable** Interrupt triggered on event as configured in the respective VMONS_VxxCON register. 0_B Interrupt is disabled. 1_B Interrupt is enabled.
0	7:6	rw	**Reserved** Should be written with 0.

(table continues...)

(continued)

Field	Bits	Type	Description
0	31:24	r	**Reserved** Read as 0; should be written with 0.

Table 1623 Reset values of PMS_INT_CON0

Reset type	Reset value	Note
LVDPMS2 Reset	0000 0000$_H$	
Cold PORST	0000 0000$_H$	

36.8.4.107 Interrupt Configuration Register 1

PMS_INT_CON1 Offset address: 9270$_H$

Interrupt Configuration Register 1 Reset values see: Table 1624

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0										UVVDDPPU	OVVDDPPU	UVVDDLMU1	OVVDDLMU1	UVVDDLMU0	OVVDDLMU0
r										rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UVVDDPHY2	OVVDDPHY2	UVVDDPHY1	OVVDDPHY1	UVVDDPHY0	OVVDDPHY0	Res		UVVDDHSIF	OVVDDHSIF	UVVDDPHPHY2	OVVDDPHPHY2	UVVDDPHPHY1	OVVDDPHPHY1	UVVDDPHPHY0	OVVDDPHPHY0
rw	rw	rw	rw	rw	rw	r		rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
OVVDDPHPHY0	0	rw	**OVVDDPHPHY0 interrupt enable** Interrupt triggered on event as configured in the respective VMONS_VxxCON register. 0_B Interrupt is disabled. 1_B Interrupt is enabled.
UVVDDPHPHY0	1	rw	**UVVDDPHPHY0 interrupt enable** Interrupt triggered on event as configured in the respective VMONS_VxxCON register. 0_B Interrupt is disabled. 1_B Interrupt is enabled.
OVVDDPHPHY1	2	rw	**OVVDDPHPHY1 interrupt enable** Interrupt triggered on event as configured in the respective VMONS_VxxCON register. 0_B Interrupt is disabled. 1_B Interrupt is enabled.

(table continues...)

(continued)

Field	Bits	Type	Description
UVVDDPHPHY1	3	rw	**UVVDDPHPHY1 interrupt enable** Interrupt triggered on event as configured in the respective VMONS_VxxCON register. 0_B Interrupt is disabled. 1_B Interrupt is enabled.
OVVDDPHPHY2	4	rw	**OVVDDPHPHY2 interrupt enable** Interrupt triggered on event as configured in the respective VMONS_VxxCON register. 0_B Interrupt is disabled. 1_B Interrupt is enabled.
UVVDDPHPHY2	5	rw	**UVVDDPHPHY2 interrupt enable** Interrupt triggered on event as configured in the respective VMONS_VxxCON register. 0_B Interrupt is disabled. 1_B Interrupt is enabled.
OVVDDHSIF	6	rw	**OVVDDHSIF interrupt enable** Interrupt triggered on event as configured in the respective VMONS_VxxCON register. 0_B Interrupt is disabled. 1_B Interrupt is enabled.
UVVDDHSIF	7	rw	**UVVDDHSIF interrupt enable** Interrupt triggered on event as configured in the respective VMONS_VxxCON register. 0_B Interrupt is disabled. 1_B Interrupt is enabled.
OVVDDPHY0	10	rw	**OVVDDPHY0 interrupt enable** Interrupt triggered on event as configured in the respective VMONS_VxxCON register. 0_B Interrupt is disabled. 1_B Interrupt is enabled.
UVVDDPHY0	11	rw	**UVVDDPHY0 interrupt enable** Interrupt triggered on event as configured in the respective VMONS_VxxCON register. 0_B Interrupt is disabled. 1_B Interrupt is enabled.
OVVDDPHY1	12	rw	**OVVDDPHY1 interrupt enable** Interrupt triggered on event as configured in the respective VMONS_VxxCON register. 0_B Interrupt is disabled. 1_B Interrupt is enabled.

(table continues...)

(continued)

Field	Bits	Type	Description
UVVDDPHY1	13	rw	**UVVDDPHY1 interrupt enable** Interrupt triggered on event as configured in the respective VMONS_VxxCON register. 0_B Interrupt is disabled. 1_B Interrupt is enabled.
OVVDDPHY2	14	rw	**OVVDDPHY2 interrupt enable** Interrupt triggered on event as configured in the respective VMONS_VxxCON register. 0_B Interrupt is disabled. 1_B Interrupt is enabled.
UVVDDPHY2	15	rw	**UVVDDPHY2 interrupt enable** Interrupt triggered on event as configured in the respective VMONS_VxxCON register. 0_B Interrupt is disabled. 1_B Interrupt is enabled.
OVVDDLMU0	16	rw	**OVVDDLMU0 interrupt enable** Interrupt triggered on event as configured in the respective VMONS_VxxCON register. 0_B Interrupt is disabled. 1_B Interrupt is enabled.
UVVDDLMU0	17	rw	**UVVDDLMU0 interrupt enable** Interrupt triggered on event as configured in the respective VMONS_VxxCON register. 0_B Interrupt is disabled. 1_B Interrupt is enabled.
OVVDDLMU1	18	rw	**OVVDDLMU1 interrupt enable** Interrupt triggered on event as configured in the respective VMONS_VxxCON register. 0_B Interrupt is disabled. 1_B Interrupt is enabled.
UVVDDLMU1	19	rw	**UVVDDLMU1 interrupt enable** Interrupt triggered on event as configured in the respective VMONS_VxxCON register. 0_B Interrupt is disabled. 1_B Interrupt is enabled.
OVVDDPPU	20	rw	**OVVDDPPU interrupt enable** Interrupt triggered on event as configured in the respective VMONS_VxxCON register. 0_B Interrupt is disabled. 1_B Interrupt is enabled.

(table continues...)

(continued)

Field	Bits	Type	Description
UVVDDPPU	21	rw	**UVVDDPPU interrupt enable** Interrupt triggered on event as configured in the respective VMONS_VxxCON register. 0_B Interrupt is disabled. 1_B Interrupt is enabled.
0	31:22	r	**Reserved** Read as 0; should be written with 0.

Table 1624 Reset values of PMS_INT_CON1

Reset type	Reset value	Note
LVDPMS2 Reset	0000 0000$_H$	
Cold PORST	0000 0000$_H$	

36.8.4.108 Pad status register

Before accessing, check the REGACCSTAT.FIFOEMPTY and REGACCSTAT.FIFOFULL status as described in the user manual (LVD reset section).

PMS_PAD_STAT Offset address: 9284$_H$

Pad status register Reset values see: Table 1625

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0							TEST MODE	SBTRIST	ALLTRIST	HWCFG5	HWCFG4	HWCFG3	HWCFG2	HWCFG1	HWCFGEVRC0
r							rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
HWCFGEVRC0	0	rh	**EVRC internal hardware configuration 0 status** This bit field indicates the value of HWCFGEVRC0, a hardware configuration input for EVRC, which is internally connected to a fixed value, either '0' or '1'.
HWCFG1	1	rh	**Hardware configuration pin 1 status** This bit field indicates the latched level of HWCFG[1] during a cold startup.
HWCFG2	2	rh	**Hardware configuration pin 2 status** This bit field indicates the latched level of HWCFG[2] during a cold startup.
HWCFG3	3	rh	**Hardware configuration pin 3 status** This bit field indicates the latched level of HWCFG[3] during a cold startup.

(table continues...)

(continued)

Field	Bits	Type	Description
HWCFG4	4	rh	**Hardware configuration pin 4 status** This bit field indicates the latched level of HWCFG[4] during a cold startup.
HWCFG5	5	rh	**Hardware configuration pin 5 status** This bit field indicates the latched level of HWCFG[5] during a cold startup.
ALLTRIST	6	rh	**Pad tristate / pull-up status** This bit indicates whether pads are configured as inputs with weak pull-up or as tristate during/after reset or after wake-up. At start-up, the value latched from HWCFG[6] pin decides the default state and is reflected in TRIST status bit. This bit may be later updated when PAD_CON.ALLTRIST is set to override initial latched status from HWCFG[6]. 0_B Pads configured as inputs with weak pull-up. 1_B Pads are in tristate.
SBTRIST	7	rh	**Standby pad tristate / pull-up status** This bit indicates whether pads are configured as inputs with weak pull-up or as tristate during/after reset or after wake-up. At start-up, the value latched from HWCFG[6] pin decides the default state and is reflected in TRIST status bit. This bit may be later updated when PAD_CON.SBTRIST is set to override initial latched status from HWCFG[6]. 0_B Pads configured as inputs with weak pull-up. 1_B Pads are in tristate.
TESTMODE	8	rh	**TESTMODE pin status** This bit field indicates the latched level of TESTMODE pin during a cold startup.
0	31:9	r	**Reserved** Read as 0.

Table 1625 Reset values of PMS_PAD_STAT

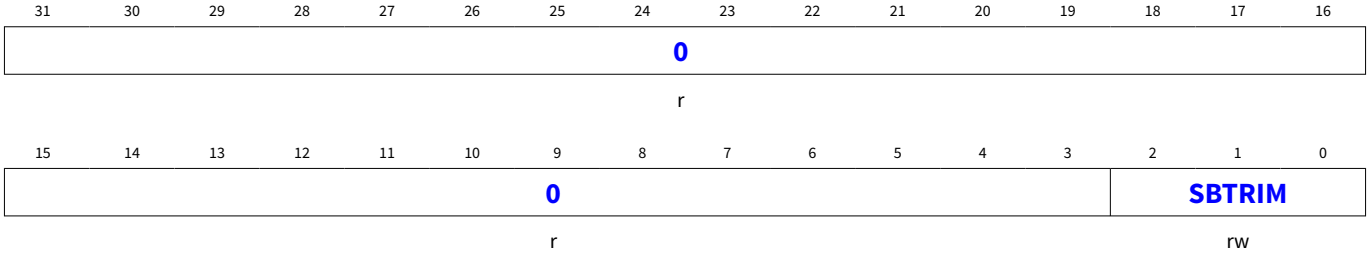
Reset type	Reset value	Note
LVDANA Reset	0000 013F$_H$	The register reset value gets overwritten by the HWCFG configuration during start-up.
After Boot-FW Value	0000 0000 0000 0000 0000 000– ---- ----$_B$	

36.8.4.109 Trim register for PMS0 components

Before accessing, check the REGACCSTAT.FIFOEMPTY and REGACCSTAT.FIFOFULL status as described in the user manual (LVD reset section).

PMS_PMS0TRIM Offset address: 92A8$_H$

Trim register for PMS0 components Reset values see: Table 1626



Field	Bits	Type	Description
SBTRIM	2:0	rw	**VDDSBRAM integrated regulator trim value** Trim value in two-complement representation.
0	31:3	r	**Reserved** Read as 0; should be written with 0.

Table 1626 Reset values of PMS_PMS0TRIM

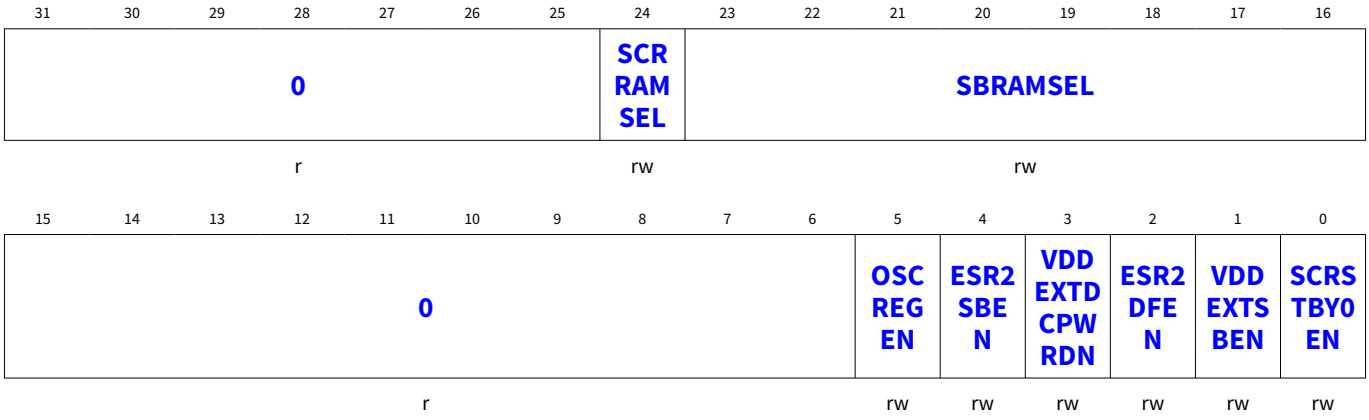
Reset type	Reset value	Note
LVDANA Reset	0000 0000_H	
After Boot-FW Value	0000 0000 0000 0000 0000 0000 0000 $0---_B$	

36.8.4.110 Standby configuration register 0

Before accessing, check the REGACCSTAT.FIFOEMPTY and REGACCSTAT.FIFOFULL status as described in the user manual (LVD reset section).

PMS_STANDBY_CON0 Offset address: $92C4_H$

Standby configuration register 0 Reset values see: Table 1627



Field	Bits	Type	Description
SCRSTBY0EN	0	rw	**Enable that SCR can trigger a transition from STBY1 to STBY0** 0_B Transition from STBY1 to STBY0 is disabled. 1_B Transition from STBY1 to STBY0 is enabled.

(table continues...)

(continued)

Field	Bits	Type	Description
VDDEXTSBEN	1	rw	**Standby entry on VDDEXT Supply ramp-down** This bit field enables Standby Entry on VDDEXT supply ramp-down. This is supported only in case Standby domain is supplied separately via VDDEVRSB supply pin and VDDEXT rail is switched off during Standby. The voltage threshold for entry is configured in VMONS_VDDEXTCON register. 0_B Standby Entry on VDDEXT supply ramp-down is disabled. 1_B Standby Entry triggered on a VDDEXT Supply undervoltage event (VDDEXTUV). Blanking filter active on Standby mode entry.
ESR2DFEN	2	rw	**ESR2 digital filter enable** This bit activates a digital filter for the standby entry trigger. If the digital filter (majority filter of 3 consecutive values, running on the 70 kHz standby clock) is enabled, then single pulses of less than 10 µs within a 60 µs window are suppressed and pulses longer than 50 µs will always result in a trigger. 0_B The filter is bypassed 1_B The filter is used
VDDEXTDCPWRDN	3	rw	**Enable powering down of VDDEXTDC during standby modes** This bit field enables the power-down of the $V_{DDEXTDC}$ external supply during standby modes. 0_B VDDEXTDC must be powered on during standby modes. 1_B VDDEXTDC can be powered off during standby modes.
ESR2SBEN	4	rw	**Standby entry on ESR2 trigger** 0_B Standby Entry on ESR2 trigger is disabled. 1_B Standby entry entry on ESR2 trigger is enabled.
OSCREGEN	5	rw	**Enable bit for the HPOSC pre-regulator during STANDBY0 and STANDBY1_70k** 0_B HPOSC pre-regulator disabled in STANDBY0 and STANDBY1_70k modes 1_B HPOSC pre-regulator enabled in STANDBY0 and STANDBY1_70k modes

(table continues...)

(continued)

Field	Bits	Type	Description
SBRAMSEL	23:16	rw	**Standby RAM supply in Standby Mode** This bit field configures the Standby RAM blocks to be kept supplied during Standby Mode. ***Note:*** *All other bit combinations are reserved.* 00_H CPU0 dLMU: 0kB supplied, CPU1 dLMU: 0kB supplied 01_H CPU0 dLMU: 64kB supplied, CPU1 dLMU: 0kB supplied 03_H CPU0 dLMU: 2x 64kB supplied, CPU1 dLMU: 0kB supplied 10_H CPU0 dLMU: 0kB supplied, CPU1 dLMU: 64kB supplied 11_H CPU0 dLMU: 64kB supplied, CPU1 dLMU: 64kB supplied 13_H CPU0 dLMU: 2x 64kB supplied, CPU1 dLMU: 64kB supplied 30_H CPU0 dLMU: 0kB supplied, CPU1 dLMU: 2x 64kB supplied 31_H CPU0 dLMU: 64kB supplied, CPU1 dLMU: 2x 64kB supplied 33_H CPU0 dLMU: 2x 64kB supplied, CPU1 dLMU: 2x 64kB supplied **others,** Reserved
SCRRAMSEL	24	rw	**SCR RAM supply in STANDBY0 mode** 0_B SCR RAM not supplied in STANDBY0 mode. 1_B SCR RAM supplied in STANDBY0 mode.
0	15:6, 31:25	r	**Reserved** Read as 0; should be written with 0.

Table 1627 Reset values of PMS_STANDBY_CON0

Reset type	Reset value	Note
LVDANA Reset	$0000\ 0000_H$	
LVDPMS0 Reset	XXXX XXX– ---- ---- $0000\ 0000\ 0000\ 0000_B$	

36.8.4.111 EVRC status register 0

The EVRC status registers EVRC_STATx are updated during start-up and operation with the actual EVRC status and therefore the read value may differ from the reset value.

PMS_EVRC_STAT0 Offset address: $92E0_H$

EVRC status register 0 LVDPMS2 Reset value: $016A\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0							SETPOINT								
r							rh								

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0							SUPPLYCFG		SDF REQ CFG	SDV OK	VDD EXTD CLVL	EVRC SHH V	EVRC SHLV	SYNC LCK	EVRC
r							rh		rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
EVRC	0	rh	**EVRC status** This bit is set if the internal EVRC regulator is currently active and no error occurred. 0_B EVRC is inactive. 1_B EVRC is active.
SYNCLCK	1	rh	**EVRC synchronization input locked status** This bitfield indicates the current synchronization status of the EVRC regulator to the external DCDCSYNCI input signal. When the EVRC switching frequency and edge is locked to the synchronization input, the SYNCLCK bit is set to HIGH indicating the locked state. When the synchronization is lost owing to frequency deviations beyond SYNCMAXDEV or the feature is disabled via SYNCIN, the SYNCLCK bit is set to LOW. 0_B EVRC regulator runs on internal configured switching frequency and is not currently synchronized to external DCDCSYNCI input signal. 1_B EVRC regulator switching frequency and VGATE output edge is currently synchronized to external DCDCSYNCI input signal.
EVRCSHLV	2	rh	**Short to ground** This bit is set if a short condition to ground has been detected. The measured EVRC output is below the operational supply range and the upper controller limits are reached.The feature is supported only during closed loop operation. 0_B No short to ground detected on VDD rail. 1_B Short to ground detected on VDD rail.
EVRCSHHV	3	rh	**Short to supply** This bit is set if a short condition to supply has been detected. The measured EVRC output exceeds the allowed supply range and the lower controller limits are reached. The feature is supported only during closed loop operation. 0_B No short to supply detected on VDD rail. 1_B Short to supply detected on VDD rail.
VDDEXTDCLVL	4	rh	**VDDEXTDC external supply level status** This bit indicates that the VDDEXTDC voltage has dropped below the threshold configured in EVRC_COEFF8.DRVFFTHRES ~4 V to indicate EVRC driver configuration switch to differentiate between 5V or 3.3V external supply. A hysteresis as configured in EVRC_COEFF8.DRVFFHYS is implemented on this detector. 0_B VDDEXTDC external supply is below the threshold, i.e. the 3.3V driver configuration is selected and used. 1_B VDDEXTDC external supply is above the threshold, i.e. the 5V driver configuration is selected and used.

(table continues...)

(continued)

Field	Bits	Type	Description
SDVOK	5	rh	**EVRC regulator voltage OK status** This bit is set by the EVRC voltage OK detector to indicate that the new regulator output value has been reached. This bit is reset whenever EVRC_CON0.SDVOUTSEL is changed (to adjust the V_{DD} voltage) and is set when the new output setpoint is reached. This bit is also reset in case droop compensation is requested before a load jump event. A time-out period of 20 µs shall be waited when polling the SDVOK bit. 0_B EVRC regulator setpoint voltage has not been reached. 1_B EVRC regulator setpoint voltage is reached and VDD voltage is ok.
SDFREQCFG	6	rh	**EVRC switching frequency configuration** This bit indicates the status of an EVRC configuration setting. 0_B Low-frequency configuration 1_B High-frequency configuration
SUPPLYCFG	8:7	rh	**EVRC supply configuration** This bit-field indicates the status of an EVRC configuration setting. 00_B EVRC disabled 01_B VDDEXTDC 3.3V supply mode 10_B VDDEXTDC 5.5V supply mode 11_B VDDEXTDC high-voltage supply mode
SETPOINT	24:16	rh	**EVRC target output voltage** The SETPOINT is the target output voltage of the EVRC including SDVOUTSEL, and droop compensation. Target voltage value = SETPOINT * LSB LSB = 2.83 mV
0	15:9, 31:25	r	**Reserved** Read as 0.

36.8.4.112 EVRC configuration register 0

PMS_EVRC_CON0 Offset address: $92EC_H$

EVRC configuration register 0 Reset values see: Table 1629

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UP								**0**							
rwh								r							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0							**SDVOUTSEL**								
r							rw								

Field	Bits	Type	Description
SDVOUTSEL	8:0	rw	**EVRC regulator output voltage target value** The V_{DD} output level of the EVRC. SDVOUTSEL = (V_{DD} / LSB); V_{DD} = LSB * SDVOUTSEL; LSB = 1.2V/424 (~=2.83 mV). This register bitfield requires a parameter update via EVRC_CON0.UP for transfer to EVRC SMPS shadow register. The reaching of the new target value is indicated by the EVRC_STAT0.SDVOK bit.
UP	31	rwh	**Update request for SMPS register values** This bitfield triggers the update of the current register values from PMS-FPI EVRC registers to the local SMPS module registers. It shall be ensured that all EVRC_CONx and EVRC_COEFFx registers have correct and coherent values across the various registers before the update request is issued. In case of singular register update, the other register values should match and be consistent. The parameter update via UP bit is not allowed in start-up mode. After setting the UP bit, the SW must wait for the UP bit to get de-asserted before performing further write accesses into EVRC registers which have shadow registers, otherwise the write accesses performed while the UP bit is asserted would be lost. 0_B No action is undertaken. 1_B A new complete EVRC parameter set is transferred to the SMPS module. All EVRC_CONx and EVRC_COEFFx register contents are transferred.
0	30:9	r	**Reserved** Read as 0; should be written with 0.

Table 1628 Access mode restrictions of PMS_EVRC_CON0 sorted by descending priority

Mode name	Access mode		Description
.UP == 0	rw	SDVOUTSEL	
(default)	r	SDVOUTSEL	

Table 1629 Reset values of PMS_EVRC_CON0

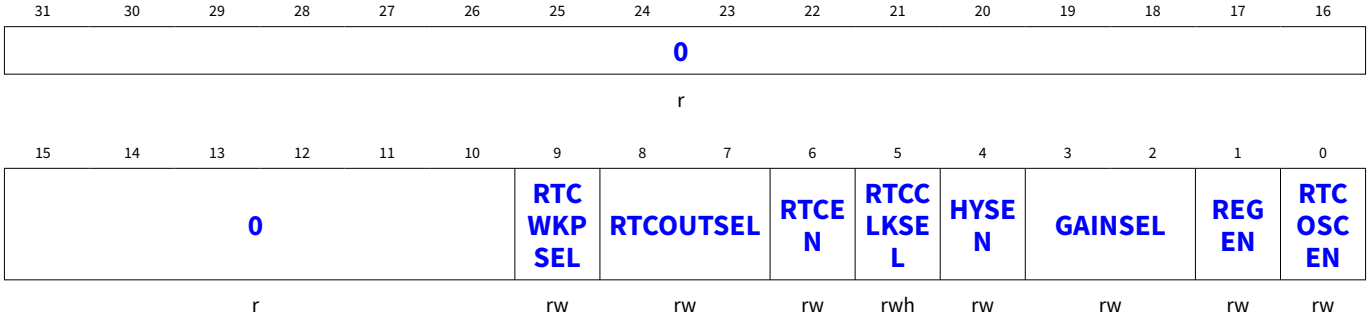
Reset type	Reset value	Note
LVDPMS2 reset for high-frequency configuration	8000 016A$_H$	
LVDPMS2 reset for low-frequency configuration	8000 016A$_H$	
Cold PORST	8000 016A$_H$	
After Boot-FW Value	0000 016A$_H$	

36.8.4.113 RTC configuration register 0

Before accessing, check the REGACCSTAT.FIFOEMPTY and REGACCSTAT.FIFOFULL status as described in the user manual (LVD reset section).

PMS_RTC_CON0 Offset address: 9318_H

RTC configuration register 0 Reset values see: Table 1631



Field	Bits	Type	Description
RTCOSCEN	0	rw	**RTC oscillator enable**
REGEN	1	rw	**Amplitude regulation** When REGEN=1, the oscillation amplitude at XTPADIN is approx. 0.5V, independent of the gain setting and on board elements.
GAINSEL	3:2	rw	**Gain control** 00_B 2x gain 01_B 4x gain (default) 10_B 6x gain 11_B 8x gain
HYSEN	4	rw	**Hysteresis enable**
RTCCLKSEL	5	rwh	**Clock source selection** Selection between 70kHz and 32kHz clock source, for the RTC. 0_B 70kHz clock source is selected (default). 1_B 32kHz clock source is selected.
RTCEN	6	rw	**Clock enable for RTC digital module**
RTCOUTSEL	8:7	rw	**RTC output to pins and timers** 00_B No output 01_B f_{RTC} 10_B CMP0 11_B CMP1
RTCWKPSEL	9	rw	**RTC wakeup selection** Selection whether the RTC wake-up occurs on a CMP0 or CMP1 match event. 0_B RTC wake-up occurs on a CMP0 match. 1_B RTC wake-up occurs on a CMP1 match.
0	31:10	r	**Reserved** Read as 0; should be written with 0.

Table 1630 Access mode restrictions of PMS_RTC_CON0 sorted by descending priority

Mode name	Access mode		Description
write 1 to .**RTCOSCEN**	rwh	RTCCLKSEL	
(default)	rh	RTCCLKSEL	

Table 1631 Reset values of PMS_RTC_CON0

Reset type	Reset value	Note
LVDANA Reset	0000 0000$_H$	
LVDPMS0 Reset	0000 0000$_H$	

36.8.4.114 Cold reset status register

Before accessing, check the REGACCSTAT.FIFOEMPTY and REGACCSTAT.FIFOFULL status as described in the user manual (LVD reset section).

PMS_CLDRSTSTAT Offset address: 9354$_H$

Cold reset status register Reset values see: Table 1632

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	Res			LBIST	PORST	PDCLDRST	CLDPORST	STBY1WKP	LVDPMS2	LVDPMS1	STBY0WKP	LVDPMS0	LVDANA
r	r	r	r			rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
LVDANA	0	rh	**LVD reset status for PMS0-Ana domain and PSSM0** 0_B This reset type has not occurred since the last clear (by CLDRSTSTATCLR.CLEAR). 1_B This reset type has occurred since the last clear (by CLDRSTSTATCLR.CLEAR).
LVDPMS0	1	rh	**LVD reset status for PMS0 domain and PSSM0** 0_B This reset type has not occurred since the last clear (by CLDRSTSTATCLR.CLEAR). 1_B This reset type has occurred since the last clear (by CLDRSTSTATCLR.CLEAR).
STBY0WKP	2	rh	**Transition from STANDBY0 to RUN mode for PSSM0** 0_B This reset trigger has not occurred since the last clear (by CLDRSTSTATCLR.CLEAR). 1_B This reset trigger has occurred since the last clear (by CLDRSTSTATCLR.CLEAR).

(table continues...)

(continued)

Field	Bits	Type	Description
LVDPMS1	3	rh	**LVD reset status for PMS1 domain and PSSM1** 0_B This reset type has not occurred since the last clear (by CLDRSTSTATCLR.CLEAR). 1_B This reset type has occurred since the last clear (by CLDRSTSTATCLR.CLEAR).
LVDPMS2	4	rh	**LVD reset status for PMS2 domain** 0_B This reset type has not occurred since the last clear (by CLDRSTSTATCLR.CLEAR). 1_B This reset type has occurred since the last clear (by CLDRSTSTATCLR.CLEAR).
STBY1WKP	5	rh	**Transition from STANDBY1 to RUN mode for PSSM1** 0_B This reset trigger has not occurred since the last clear (by CLDRSTSTATCLR.CLEAR). 1_B This reset trigger has occurred since the last clear (by CLDRSTSTATCLR.CLEAR).
CLDPORST	6	rh	**Cold PORST reset status** 0_B This reset type has not occurred since the last clear (by CLDRSTSTATCLR.CLEAR). 1_B This reset type has occurred since the last clear (by CLDRSTSTATCLR.CLEAR).
PDCLDRST	7	rh	**Power domain cold reset status** Power domain cold reset. Power switches and control logic, if available on the product variant, are reset. The $V_{DDPDCLDUV}$ cold reset threshold (deeper than $V_{DDPRIUV}$) has been crossed. 0_B This reset type has not occurred since the last clear (by CLDRSTSTATCLR.CLEAR). 1_B This reset type has occurred since the last clear (by CLDRSTSTATCLR.CLEAR).
PORST	8	rh	**Warm PORST status** Warm PORST reset triggered by PORST pin. 0_B This reset type has not occurred since the last clear (by CLDRSTSTATCLR.CLEAR). 1_B This reset type has occurred since the last clear (by CLDRSTSTATCLR.CLEAR).
LBIST	9	rh	**LBIST status** LBIST reset. This bitfield indicates that LBIST occured. 0_B This reset type has not occurred since the last clear (by CLDRSTSTATCLR.CLEAR). 1_B This reset type has occurred since the last clear (by CLDRSTSTATCLR.CLEAR).
0	13, 14, 31:15	r	**Reserved** Read as 0.

Table 1632 Reset values of PMS_CLDRSTSTAT

Reset type	Reset value	Note
LVDPMS1 Reset	0000 0000 0000 0000 0000 0000 0000 1XXX$_B$	
LVDANA Reset	XXXX XXXX XXXX XXXX XXXX XXXX XXXX X001$_B$	
LVDPMS0 Reset	XXXX XXXX XXXX XXXX XXX– ---- ---- -01-$_B$	

36.8.4.115 Cold reset trigger register

PMS_CLDRSTRIG Offset address: 9358$_H$

Cold reset trigger register LVDPMS2 Reset value: 0000 0000$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0							CPSWTRIG	PDCOLD	LBPORST	LBTERM	PORSTTO	VDDPHY2	VDDPHY1	VDDPHY0	Res
r							rh	rh	rh	rh	rh	rh	rh	rh	r

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VDDPPU	VDDLMU1	VDDLMU0	VDDPAD	Res	VDDHSIF	VDDPHPHY2	VDDPHPHY1	VDDPHPHY0	VDDM	VDDEVRSB	VDDDCLS	VDDEXTDC	VSSDCHS	VDDEXT	VDD
rh	rh	rh	rh	r	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
VDD	0	rh	**VDD primary monitor cold PORST indication** 0_B This reset trigger has not occurred since the last clear (by CLDRSTSTATCLR.CLEAR). 1_B This reset trigger has occurred since the last clear (by CLDRSTSTATCLR.CLEAR).
VDDEXT	1	rh	**VDDEXT primary monitor cold PORST indication** 0_B This reset trigger has not occurred since the last clear (by CLDRSTSTATCLR.CLEAR). 1_B This reset trigger has occurred since the last clear (by CLDRSTSTATCLR.CLEAR).
VSSDCHS	2	rh	**DCDC high-side supply primary monitor cold PORST indication** 0_B This reset trigger has not occurred since the last clear (by CLDRSTSTATCLR.CLEAR). 1_B This reset trigger has occurred since the last clear (by CLDRSTSTATCLR.CLEAR).

(table continues...)

(continued)

Field	Bits	Type	Description
VDDEXTDC	3	rh	**VDDEXTDC primary monitor cold PORST indication** 0_B This reset trigger has not occurred since the last clear (by CLDRSTSTATCLR.CLEAR). 1_B This reset trigger has occurred since the last clear (by CLDRSTSTATCLR.CLEAR).
VDDDCLS	4	rh	**DCDC low-side supply primary monitor cold PORST indication** 0_B This reset trigger has not occurred since the last clear (by CLDRSTSTATCLR.CLEAR). 1_B This reset trigger has occurred since the last clear (by CLDRSTSTATCLR.CLEAR).
VDDEVRSB	5	rh	**VDDEVRSB primary monitor cold PORST indication** 0_B This reset trigger has not occurred since the last clear (by CLDRSTSTATCLR.CLEAR). 1_B This reset trigger has occurred since the last clear (by CLDRSTSTATCLR.CLEAR).
VDDM	6	rh	**VDDM primary monitor cold PORST indication** 0_B This reset trigger has not occurred since the last clear (by CLDRSTSTATCLR.CLEAR). 1_B This reset trigger has occurred since the last clear (by CLDRSTSTATCLR.CLEAR).
VDDPHPHY0	7	rh	**VDDPHPHY0 primary monitor cold PORST indication** 0_B This reset trigger has not occurred since the last clear (by CLDRSTSTATCLR.CLEAR). 1_B This reset trigger has occurred since the last clear (by CLDRSTSTATCLR.CLEAR).
VDDPHPHY1	8	rh	**VDDPHPHY1 primary monitor cold PORST indication** 0_B This reset trigger has not occurred since the last clear (by CLDRSTSTATCLR.CLEAR). 1_B This reset trigger has occurred since the last clear (by CLDRSTSTATCLR.CLEAR).
VDDPHPHY2	9	rh	**VDDPHPHY2 primary monitor cold PORST indication** 0_B This reset trigger has not occurred since the last clear (by CLDRSTSTATCLR.CLEAR). 1_B This reset trigger has occurred since the last clear (by CLDRSTSTATCLR.CLEAR).
VDDHSIF	10	rh	**VDDHSIF primary monitor cold PORST indication** 0_B This reset trigger has not occurred since the last clear (by CLDRSTSTATCLR.CLEAR). 1_B This reset trigger has occurred since the last clear (by CLDRSTSTATCLR.CLEAR).

(table continues...)

(continued)

Field	Bits	Type	Description
VDDPAD	12	rh	**VDDPAD primary monitor cold PORST indication** 0_B This reset trigger has not occurred since the last clear (by CLDRSTSTATCLR.CLEAR). 1_B This reset trigger has occurred since the last clear (by CLDRSTSTATCLR.CLEAR).
VDDLMU0	13	rh	**VDDLMU0 primary monitor cold PORST indication** 0_B This reset trigger has not occurred since the last clear (by CLDRSTSTATCLR.CLEAR). 1_B This reset trigger has occurred since the last clear (by CLDRSTSTATCLR.CLEAR).
VDDLMU1	14	rh	**VDDLMU1 primary monitor cold PORST indication** 0_B This reset trigger has not occurred since the last clear (by CLDRSTSTATCLR.CLEAR). 1_B This reset trigger has occurred since the last clear (by CLDRSTSTATCLR.CLEAR).
VDDPPU	15	rh	**VDDPPU primary monitor cold PORST indication** 0_B This reset trigger has not occurred since the last clear (by CLDRSTSTATCLR.CLEAR). 1_B This reset trigger has occurred since the last clear (by CLDRSTSTATCLR.CLEAR).
VDDPHY0	17	rh	**VDDPHY0 primary monitor cold PORST indication** 0_B This reset trigger has not occurred since the last clear (by CLDRSTSTATCLR.CLEAR). 1_B This reset trigger has occurred since the last clear (by CLDRSTSTATCLR.CLEAR).
VDDPHY1	18	rh	**VDDPHY1 primary monitor cold PORST indication** 0_B This reset trigger has not occurred since the last clear (by CLDRSTSTATCLR.CLEAR). 1_B This reset trigger has occurred since the last clear (by CLDRSTSTATCLR.CLEAR).
VDDPHY2	19	rh	**VDDPHY2 primary monitor cold PORST indication** 0_B This reset trigger has not occurred since the last clear (by CLDRSTSTATCLR.CLEAR). 1_B This reset trigger has occurred since the last clear (by CLDRSTSTATCLR.CLEAR).
PORSTTO	20	rh	**Cold PORST reset triggered by a warm PORST time-out** 0_B This reset trigger has not occurred since the last clear (by CLDRSTSTATCLR.CLEAR). 1_B This reset trigger has occurred since the last clear (by CLDRSTSTATCLR.CLEAR).

(table continues...)

(continued)

Field	Bits	Type	Description
LBTERM	21	rh	**LBIST was properly terminated for respective domains** This bit-field indicates if the LBIST was terminated properly. This bit-field is cleared when CLDRSTSTATCLR.CLEAR is set. 0_B LBIST was not terminated properly. 1_B LBIST was terminated properly.
LBPORST	22	rh	**LBIST termination due to warm PORST** This bit-field indicates if the LBIST was early terminated due to the occurrence of a warm Power On Reset. If the status of this bitfield is 0, the application must still check the LBTERM to check if the LBIST was terminated properly. This bit-field is cleared when CLDRSTSTATCLR.CLEAR is set. 0_B LBIST was not terminated early due to a Power On Reset 1_B LBIST early termination due to the occurrence of Power On Reset.
PDCOLD	23	rh	**Cold reset due to VDDPDCLDUV threshold crossed** Power domain cold reset trigger occurrence ($V_{DDPDCLDUV}$ threshold). 0_B This reset trigger has not occurred since the last clear (by CLDRSTSTATCLR.CLEAR). 1_B This reset trigger has occurred since the last clear (by CLDRSTSTATCLR.CLEAR).
CPSWTRIG	24	rh	**Cold PORST software trigger** Cold PORST triggered by application software (through CLDRSTCON.CPTRIG). 0_B This reset trigger has not occurred since the last clear (by CLDRSTSTATCLR.CLEAR). 1_B This reset trigger has occurred since the last clear (by CLDRSTSTATCLR.CLEAR).
0	31:25	r	**Reserved** Read as 0.

36.8.4.116 Cold reset last trigger register

PMS_CLDRSTRIGL Offset address: $935C_H$

Cold reset last trigger register LVDPMS2 Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0							CPSWTRIG	PDCOLD	LBPORST	LBTERM	PORSTTO	VDDPHY2	VDDPHY1	VDDPHY0	Res
r							rh	rh	rh	rh	rh	rh	rh	rh	r

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VDDPPU	VDDLMU1	VDDLMU0	VDDPAD	Res	VDDHSIF	VDDPHPHY2	VDDPHPHY1	VDDPHPHY0	VDDM	VDDEVRSB	VDDDCLS	VDDEXTDC	VSSDCHS	VDDEXT	VDD
rh	rh	rh	rh	r	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
VDD	0	rh	**VDD primary monitor cold PORST indication** 0_B This reset trigger has not occurred since the last clear (by CLDRSTSTATCLR.CLEAR). 1_B This reset trigger has occurred since the last clear (by CLDRSTSTATCLR.CLEAR).
VDDEXT	1	rh	**VDDEXT primary monitor cold PORST indication** 0_B This reset trigger has not occurred since the last clear (by CLDRSTSTATCLR.CLEAR). 1_B This reset trigger has occurred since the last clear (by CLDRSTSTATCLR.CLEAR).
VSSDCHS	2	rh	**DCDC high-side supply primary monitor cold PORST indication** 0_B This reset trigger has not occurred since the last clear (by CLDRSTSTATCLR.CLEAR). 1_B This reset trigger has occurred since the last clear (by CLDRSTSTATCLR.CLEAR).
VDDEXTDC	3	rh	**VDDEXTDC primary monitor cold PORST indication** 0_B This reset trigger has not occurred since the last clear (by CLDRSTSTATCLR.CLEAR). 1_B This reset trigger has occurred since the last clear (by CLDRSTSTATCLR.CLEAR).
VDDDCLS	4	rh	**DCDC low-side supply primary monitor cold PORST indication** 0_B This reset trigger has not occurred since the last clear (by CLDRSTSTATCLR.CLEAR). 1_B This reset trigger has occurred since the last clear (by CLDRSTSTATCLR.CLEAR).
VDDEVRSB	5	rh	**VDDEVRSB primary monitor cold PORST indication** 0_B This reset trigger has not occurred since the last clear (by CLDRSTSTATCLR.CLEAR). 1_B This reset trigger has occurred since the last clear (by CLDRSTSTATCLR.CLEAR).
VDDM	6	rh	**VDDM primary monitor cold PORST indication** 0_B This reset trigger has not occurred since the last clear (by CLDRSTSTATCLR.CLEAR). 1_B This reset trigger has occurred since the last clear (by CLDRSTSTATCLR.CLEAR).
VDDPHPHY0	7	rh	**VDDPHPHY0 primary monitor cold PORST indication** 0_B This reset trigger has not occurred since the last clear (by CLDRSTSTATCLR.CLEAR). 1_B This reset trigger has occurred since the last clear (by CLDRSTSTATCLR.CLEAR).

(table continues...)

(continued)

Field	Bits	Type	Description
VDDPHPHY1	8	rh	**VDDPHPHY1 primary monitor cold PORST indication** 0_B This reset trigger has not occurred since the last clear (by CLDRSTSTATCLR.CLEAR). 1_B This reset trigger has occurred since the last clear (by CLDRSTSTATCLR.CLEAR).
VDDPHPHY2	9	rh	**VDDPHPHY2 primary monitor cold PORST indication** 0_B This reset trigger has not occurred since the last clear (by CLDRSTSTATCLR.CLEAR). 1_B This reset trigger has occurred since the last clear (by CLDRSTSTATCLR.CLEAR).
VDDHSIF	10	rh	**VDDHSIF primary monitor cold PORST indication** 0_B This reset trigger has not occurred since the last clear (by CLDRSTSTATCLR.CLEAR). 1_B This reset trigger has occurred since the last clear (by CLDRSTSTATCLR.CLEAR).
VDDPAD	12	rh	**VDDPAD primary monitor cold PORST indication** 0_B This reset trigger has not occurred since the last clear (by CLDRSTSTATCLR.CLEAR). 1_B This reset trigger has occurred since the last clear (by CLDRSTSTATCLR.CLEAR).
VDDLMU0	13	rh	**VDDLMU0 primary monitor cold PORST indication** 0_B This reset trigger has not occurred since the last clear (by CLDRSTSTATCLR.CLEAR). 1_B This reset trigger has occurred since the last clear (by CLDRSTSTATCLR.CLEAR).
VDDLMU1	14	rh	**VDDLMU1 primary monitor cold PORST indication** 0_B This reset trigger has not occurred since the last clear (by CLDRSTSTATCLR.CLEAR). 1_B This reset trigger has occurred since the last clear (by CLDRSTSTATCLR.CLEAR).
VDDPPU	15	rh	**VDDPPU primary monitor cold PORST indication** 0_B This reset trigger has not occurred since the last clear (by CLDRSTSTATCLR.CLEAR). 1_B This reset trigger has occurred since the last clear (by CLDRSTSTATCLR.CLEAR).
VDDPHY0	17	rh	**VDDPHY0 primary monitor cold PORST indication** 0_B This reset trigger has not occurred since the last clear (by CLDRSTSTATCLR.CLEAR). 1_B This reset trigger has occurred since the last clear (by CLDRSTSTATCLR.CLEAR).

(table continues...)

(continued)

Field	Bits	Type	Description
VDDPHY1	18	rh	**VDDPHY1 primary monitor cold PORST indication** 0_B This reset trigger has not occurred since the last clear (by CLDRSTSTATCLR.CLEAR). 1_B This reset trigger has occurred since the last clear (by CLDRSTSTATCLR.CLEAR).
VDDPHY2	19	rh	**VDDPHY2 primary monitor cold PORST indication** 0_B This reset trigger has not occurred since the last clear (by CLDRSTSTATCLR.CLEAR). 1_B This reset trigger has occurred since the last clear (by CLDRSTSTATCLR.CLEAR).
PORSTTO	20	rh	**Cold PORST reset triggered by a warm PORST time-out** 0_B This reset trigger has not occurred since the last clear (by CLDRSTSTATCLR.CLEAR). 1_B This reset trigger has occurred since the last clear (by CLDRSTSTATCLR.CLEAR).
LBTERM	21	rh	**LBIST was properly terminated for respective domains** This bit-field indicates if the LBIST was terminated properly. This bit-field is cleared when CLDRSTSTATCLR.CLEAR is set. 0_B LBIST was not terminated properly. 1_B LBIST was terminated properly.
LBPORST	22	rh	**LBIST termination due to warm PORST** This bit-field indicates if the LBIST was early terminated due to the occurrence of a warm Power On Reset. If the status of this bitfield is 0, the application must still check the LBTERM to check if the LBIST was terminated properly. This bit-field is cleared when CLDRSTSTATCLR.CLEAR is set. 0_B LBIST was not terminated early due to a Power On Reset 1_B LBIST early termination due to the occurrence of Power On Reset.
PDCOLD	23	rh	**Cold reset due to VDDPDCLDUV threshold crossed** Power domain cold reset trigger occurrence ($V_{DDPDCLDUV}$ threshold). 0_B This reset trigger has not occurred since the last clear (by CLDRSTSTATCLR.CLEAR). 1_B This reset trigger has occurred since the last clear (by CLDRSTSTATCLR.CLEAR).
CPSWTRIG	24	rh	**Cold PORST software trigger** Cold PORST triggered by application software (through CLDRSTCON.CPTRIG). 0_B This reset trigger has not occurred since the last clear (by CLDRSTSTATCLR.CLEAR). 1_B This reset trigger has occurred since the last clear (by CLDRSTSTATCLR.CLEAR).
0	31:25	r	**Reserved** Read as 0.

36.8.4.117 OCDS trigger set control 0 register

PMS_OTSC0 Offset address: 9368_H

OCDS trigger set control 0 register Reset values see: Table 1633

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								B1PMS							
r								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								B0PMS							
r								rw							

Field	Bits	Type	Description
B0PMS	7:0	rw	**OTGB0 PMS trigger set selection** 00_H No trigger selected 01_H VMONP_VDDSTAT.RESULT 02_H VMONP_VDDEXTSTAT.RESULT 03_H VMONP_VSSDCHSSTAT.RESULT 04_H VMONP_VDDEXTDCSTAT.RESULT 05_H VMONP_VDDDCLSSTAT.RESULT 06_H VMONP_VDDEVRSBSTAT.RESULT 07_H VMONP_VDDMSTAT.RESULT 08_H VMONP_VDDPHPHY0STAT.RESULT 09_H VMONP_VDDPHPHY1STAT.RESULT $0A_H$ VMONP_VDDPHPHY2STAT.RESULT $0B_H$ VMONP_VDDHSIFSTAT.RESULT $0D_H$ VMONP_VDDPADSTAT.RESULT $0E_H$ VMONP_VDDLMU0STAT.RESULT $0F_H$ VMONP_VDDLMU1STAT.RESULT 10_H VMONP_VDDPPUSTAT.RESULT 12_H VMONP_VDDPHY0STAT.RESULT 13_H VMONP_VDDPHY1STAT.RESULT 14_H VMONP_VDDPHY2STAT.RESULT 15_H VMONS_VDDPMS0STAT.RESULT 16_H VMONS_VDDPMS2STAT.RESULT 17_H DTS_STAT.RESULT 18_H PEAK_VDD.PEAKVOLT 19_H PEAK_VDDEXT.PEAKVOLT $1A_H$ PEAK_VDDEXTDC.PEAKVOLT $1B_H$ WUT_STAT.COUNTER[11:0] $1C_H$ WUT_STAT.COUNTER[23:12] $1D_H$ RTC Timer[15:0] $1E_H$ RTC Timer[31:16] $1F_H$ RTC Timer[47:32] 20_H RTC Timer[63:48] 21_H TCINT/SCRINT **others,** Reserved (no trigger selected)

(table continues...)

(continued)

Field	Bits	Type	Description
B1PMS	23:16	rw	**OTGB1 PMS trigger set selection** 00_H No trigger selected 01_H VMONP_VDDSTAT.RESULT 02_H VMONP_VDDEXTSTAT.RESULT 03_H VMONP_VSSDCHSSTAT.RESULT 04_H VMONP_VDDEXTDCSTAT.RESULT 05_H VMONP_VDDDCLSSTAT.RESULT 06_H VMONP_VDDEVRSBSTAT.RESULT 07_H VMONP_VDDMSTAT.RESULT 08_H VMONP_VDDPHPHY0STAT.RESULT 09_H VMONP_VDDPHPHY1STAT.RESULT $0A_H$ VMONP_VDDPHPHY2STAT.RESULT $0B_H$ VMONP_VDDHSIFSTAT.RESULT $0D_H$ VMONP_VDDPADSTAT.RESULT $0E_H$ VMONP_VDDLMU0STAT.RESULT $0F_H$ VMONP_VDDLMU1STAT.RESULT 10_H VMONP_VDDPPUSTAT.RESULT 12_H VMONP_VDDPHY0STAT.RESULT 13_H VMONP_VDDPHY1STAT.RESULT 14_H VMONP_VDDPHY2STAT.RESULT 15_H VMONS_VDDPMS0STAT.RESULT 16_H VMONS_VDDPMS2STAT.RESULT 17_H DTS_STAT.RESULT 18_H PEAK_VDD.PEAKVOLT 19_H PEAK_VDDEXT.PEAKVOLT $1A_H$ PEAK_VDDEXTDC.PEAKVOLT $1B_H$ WUT_STAT.COUNTER[11:0] $1C_H$ WUT_STAT.COUNTER[23:12] $1D_H$ RTC Timer[15:0] $1E_H$ RTC Timer[31:16] $1F_H$ RTC Timer[47:32] 20_H RTC Timer[63:48] 21_H TCINT/SCRINT **others,** Reserved (no trigger selected)
0	15:8, 31:24	r	**Reserved** Read as 0; must be written with 0.

Table 1633 Reset values of PMS_OTSC0

Reset type	Reset value	Note
LVDPMS2 Reset	$0000\ 0000_H$	
Debug Reset	$0000\ 0000_H$	

36.8.5 TC4Dx PMS connectivity

The table below lists interfaces to and from this functional block to other blocks in the device.

Table 1634 List of PMS interface signals

Interface signals	I/O	Description
EGTM_PMS_DCDCSYNCI	In	EVRC synchronization input.
PMS_PORTS_DCDCSYNCO	Out	EVRC synchronization output.
PMS_EGTM_DCDCSYNCO	Out	EVRC synchronization output.
PMS_EGTM_RTCOUT	Out	RTC output.
PMS_EGTM_WUTOUT	Out	WUT output.
PMS_SMM_ColdPORST	Out	Reset type triggered by PMS upon under-voltage conditions.
PMS_SMM_WarmPORST	Out	Warm PORST request from PORST pin.
PMS_SMM_RSTSTATy.z	Out	Reset status registers to indicate reset types and triggers.
SMM_PMS_VDROOP	In	Voltage droop request to PMS EVRC.
SMM_PMS_REQSLP_STANDBY	In	Standby request.
PMS_CLOCK_fBACK	Out	70KHz backup clock
PMS_DFT_fBACK	Out	70KHz backup clock
PMS_SCR_fBACK	Out	70KHz backup clock

36.8.6 TC4Dx PMS revision history

Reference	Description of change(s)
Date range: 2024-08-17 to 2024-11-07	
TC4Dx supply mode selection	• Image replaced for TC4Dx

37 System Mode Management (SMM)

The System Mode Management (SMM) and Reset Unit coordinate the system mode transitions including supply ramp-up, reset, start-up, operation, and ramp-down or standby states. The functional blocks PMS, PORTS, SMM, LBIST, SMU, processing units , CLOCKING and SSW interact to realize the various system states by triggering and propagating the transition events.

37.1 Feature list

Supported system modes

- RESET State: The device supports the following hierarchical reset states - Low Voltage Detector reset, Cold Power-On reset, Power Domain cold reset , Logical BIST reset, warm Power-On reset, System reset and Application reset
- RUN0 Mode: The device is in operation after the de-assertion of all resets and Boot Firmware is executed. CPU0 and CPUcs are active in this phase. All switchable power domains in VDD core domain are active
- RUN1 Mode: The device is in operation and user start-up software is executed. The switchable power domains which are not required are powered off
- IDLE Modes: The IDLE Mode is available for every processing unit such as CPUs where the respective cores are put into idle state
- SLEEP Modes: All CPUs are in Idle mode and only a few selected peripherals are active to realize wake-up functions. The SRI and SPB frequency can be reduced to reduce power consumption
- STANDBYx Modes: Only PMS0 and/or PMS1 standby domains are active. All other modules and VDD Core domain is switched off

Supported reset types

- Low Voltage Detector triggered resets: These resets are triggered by PMS in case of power fails of the standby supply domains or indicated after an initial supply ramp-up
- Power Domain Cold reset: This reset is triggered by PMS in the case of a VDD core voltage supply fail beyond which the power domain infrastructure and switch components need to be initialized
- Cold Power-On reset: This reset is triggered by PMS in the case of a power fail of critical IO and core supply rails below the operating minimum voltage limits . All modules are in reset state except parts of PMS module
- LBIST reset: This reset represents the device state when LBIST is active or being performed. After LBIST is completed or aborted, internal cold PORST is triggered. All modules are in reset state except PMS module
- Warm Power-on reset: This reset is triggered when the external power-on reset pin (PORST) is asserted low. This reset is triggered predominantly by external regulators in case the microcontroller does not service external watchdogs within a defined time period. All modules are in reset state except PMS, SCR if configured, Standby RAMs, and SMM and RRAM[45]. RAMs of processing units maintain data content during and after reset
- System reset: This reset is triggered by software, safety and security alarms, and watchdogs through the SMU unit, ESRx reset pins, system timers, test and debug modules. All modules are reset except PMS, SCR, SMM, RRAM[45], Standby RAMs, and parts of SMU and debug system, based on configuration. RAMs of processing units maintain data content during and after reset
- Application reset: This reset is triggered by software, safety and security alarms, and watchdogs through the SMU unit, ESRx reset pins, system timers and test and debug modules. All modules are reset except PMS, SCR, SMM, Standby RAMs, Clock, Flash[45],RRAM[45], parts of SMU, SCU and the debug system based on configuration. RAMs of processing units maintain data content during and after reset. This reset ensures a fast overall microcontroller reset with minimal reset time

45 Not available on all devices of the AURIX™ TC4xx family

- Module Group Resets: A predefined group of modules provide partitioning between multiple applications. These resets are triggered for such groups of modules which are bundled together to be reset together. The module reset behavior is the same as that after individual module resets
- Module reset: The respective module is reset. Module reset is not available for CPU0, CPUcs, Flash[45], RRAM[45], PMS, IR, SCU, SMM, and CCU. This reset is also used to specifically re-initialize the respective modules

External interfaces and features

- Bi-directional PORST pin: During cold resets, the PORST pin is asserted low internally therefore driving a reset output. The PORST pin is consequently de-asserted with additional voltage hysteresis once the supply is back in the operating region
- The external request pins (ESR0, ESR1 and ESR2) support reset input trigger requests
- The external request pins (ESR0 and ESR2) support reset output propagation with configurable delay to external driver and memory devices. This is to support open-drain star connected reset architectures and cascaded sequential reset architectures
- The external request pins (ESR0, ESR1 and ESR2) support trap requests, NMI inputs, and wake-up requests, depending on configuration and system modes
- Shut-down reset sequencing of modules after a reset request over a time period to ensure regulator friendly load transients

Start-up

- Coordination together with PMS of reset release and boot
- The startup configuration (HWCFG Pins) is latched into status register in PMS

Load jump management

- Adaption of core voltage set points voltage levels for load jump handling
- Shutdown sequencing for negative load jumps in case of reset events

37.2 Functional overview

The System Mode Management (SMM) functional block describes the device system modes and transitions between them. SMM co-ordinates supply ramp-up, reset, start-up, operation, and ramp-down or standby states.

The LVD resets, Power Domain Cold Reset, Cold PORST Reset, LBIST Reset, warm PORST Reset, System Reset, and Application Reset are hierarchical in nature with the higher reset triggering the lower reset class. The Module Group Resets and module resets are exceptions to this general rule.

The PMS.PORDET detectors release the low voltage detector resets once adequate supply voltage levels are reached for isolation release, ensuring the start of the basic infrastructure and power supplies.

Once the supply system, internal clock and reference voltages are up and running, the primary voltage monitors PMS.VMONPx release the cold PORST ensuring that all internal and external supply voltages are above their minimum operational level. The LBIST would manifest externally as a device being in reset state but the device having high power consumption and load transients. The PORST pin is externally asserted or de-asserted by the external power supply to reset the microcontroller. The release of PORST activates CPU0 and CPUcs into RUN mode and starts the start-up software, and consequently the user application.

System and application resets are triggered by software and hardware sources to debounce errors and failures of parts of the microcontroller. The choice of reset depends on the time available for reset, the reset trigger, and the application needs. The SMM collects the triggers from various reset sources and issues warm reset (warm PORST, system, or application resets). All resets are propagated to external devices using ESRx, FSPx, and PORST pins.

45 Not available on all devices of the AURIX™ TC4xx family

37 System Mode Management (SMM)

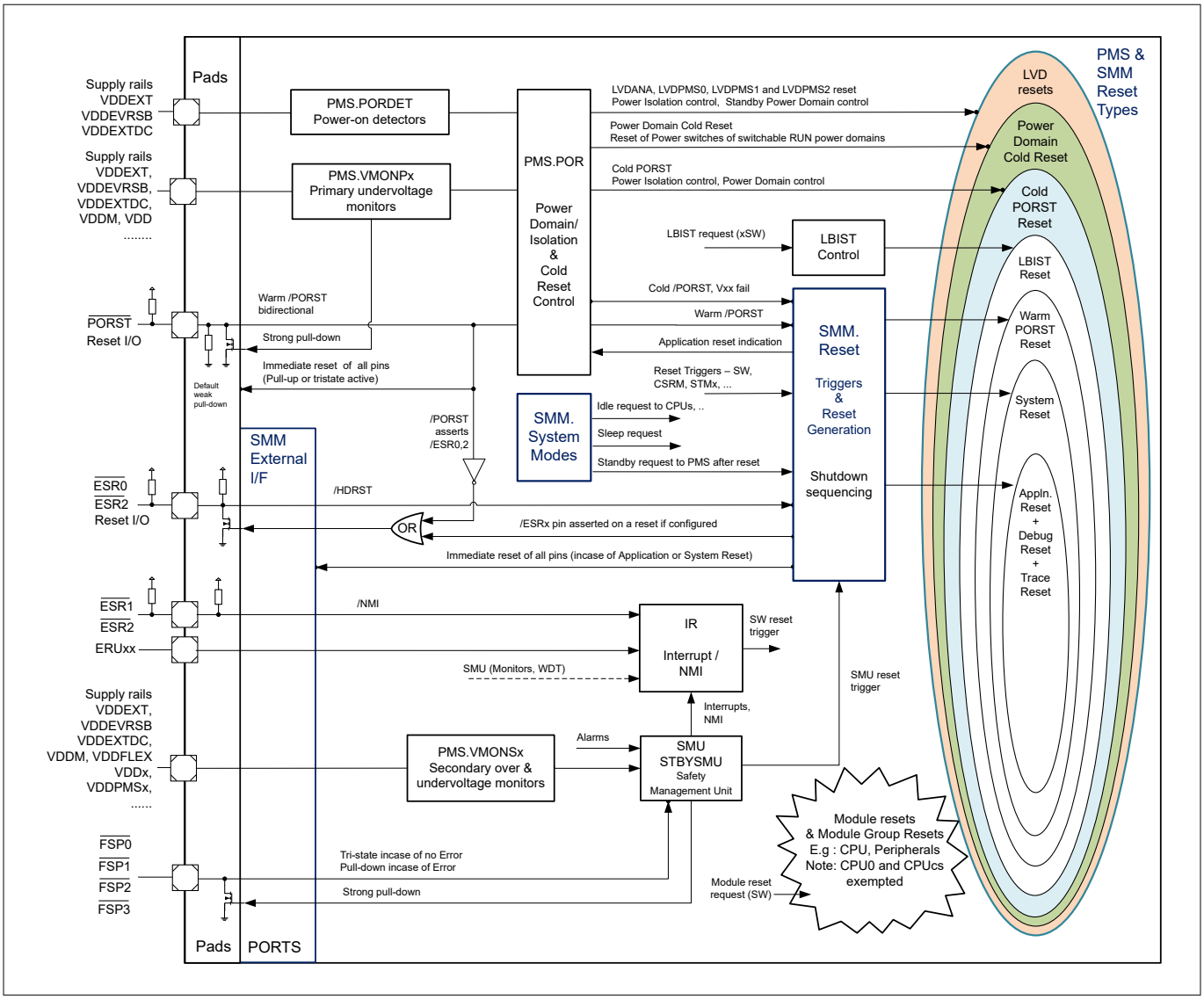


Figure 814 SMM block diagram

For the alarm description please refer to the alarm mapping tables in the SMU functional block and the Safety Manual.

Related information

TC4Dx SMU alarm mapping tables on page 7233

37.3 Functional description

The SMM functional block constitutes following sub-blocks:

- Start-up
  - The start-up block latches hardware and software configuration on a reset release, coordinates along with PMS the reset release to the modules and manages boot modes together with Boot Firmware
- System Mode control
  - The System Modes manages the triggering of Idle, Sleep, and Standby Modes to various modules in the device
- Load Jump Management
  - The Load Jump Management block adapts core voltage set point voltage levels of EVRC before load jump events are triggered to have adequate voltage headroom. The shut-down sequencing scheme manages negative load jumps on reset events and mode transitions
- External Interface
  - The External Interface block manages the IO interface of ESR and PORST reset pins
- Reset Generation
  - The Reset block evaluates reset triggers and generates the required reset types

37.3.1 Start-up

The start-up phase of the device is shown below with regards to supply ramp-up, reset release, and mode transitions.

Start-up on a supply ramp-up

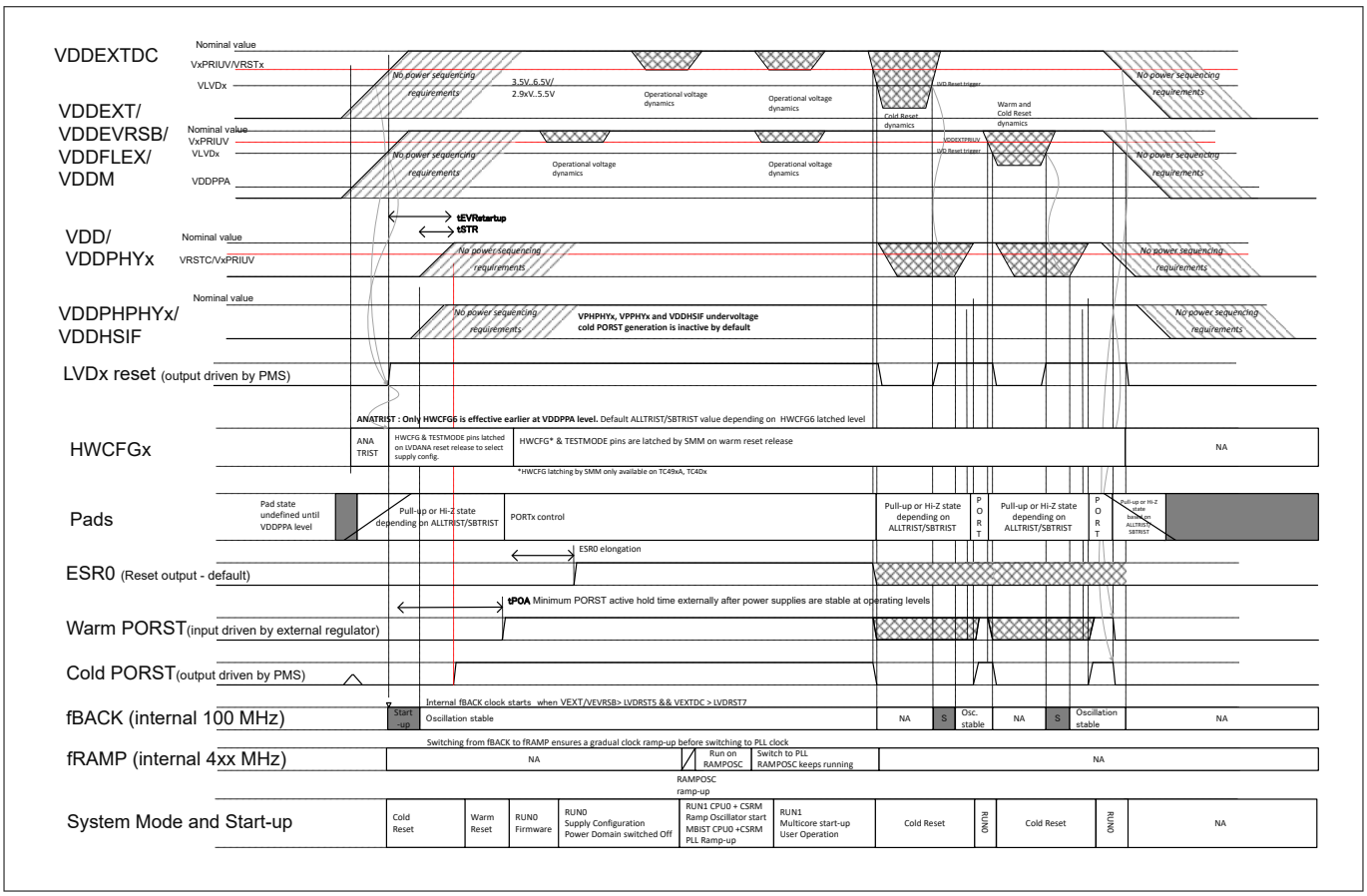


Figure 815 Cold start-up

All voltage supplies are generated externally except for the VDD supply. VDD supply may be generated externally or internally. All externally generated supplies may ramp-up and ramp-down independently with regards to start, rise and fall time(s).

If VDD (and VDDPHYx) is generated internally by the EVRC regulator, then tEVRstartup timings define EVRC start-up delay from the moment when all LVD resets are de-asserted. The EVRC start-up delay includes start-up of PMS infrastructure components including the start-up of fBACK internal 100 MHz back-up clock. The EVRC ramp-up timing is defined by the parameter tSTR.

A number of LVD detectors monitor external supply voltages VDDEVRSB, VDDEXT, and VDDEXTDC, and internal supply voltages (not present on this diagram) VDDPMS0, VDDSBRAM, VDDPMS1, and VDDPMS2. LVD reset is de-asserted when respective supplies are above the LVD limits. Refer to PMS chapter for more details.

HWCFG pads are latched directly by PMS when LVDANA reset is de-asserted and VDDEXT is supplied. Supply mode is selected based on the HWCFG[1, 2] and TESTMODE pins.

The states of general purpose pads depend on the state of the VDDEXT during ramp-up. Once the VDDPPA level is reached, the pull devices are activated. Therefore the default pad state before LVDANA reset release depends directly on the level of the HWCFG[6] pad after the moment when VDDEXT/VDDEVRSB = VDDPPA.

After LVDANA reset, the ALLTRIST/SBTRIST signal state of the PAD_CON register from PMS takes over the control of the default reset state of the pads based on the HWCFG[6] pin state latched at the moment when LVDANA reset is released. The ALLTRIST/SBTRIST signal differentiates the reset state of pads in the PORT unit in VDDEXT and standby domain respectively.

***Note**: For details about LVDANA reset refer to the PMS chapter*

Cold PORST (output) active means that the microcontroller asserts the reset internally and drives the PORST pin low thereby propagating the reset to external devices. The Cold PORST (output) is asserted by the microcontroller when at least one of the supply voltages is below the respective primary under voltage VxPRIUV or VRSTx thresholds. The Cold PORST (output) is de-asserted by the microcontroller when all supplies are above their respective primary reset thresholds, and PMS infrastructure including internal PMS supplies and fBACK clock are ready.

Cold PORST (output) generation is inactive for VDDPHYx, VDDPHPHYx, and VDDHSIFx after LVDPMS2 reset is de-asserted. Respective monitors can be activated by run-time configuration.

All operational voltage dynamics within operational limits (above cold reset limits) do not trigger the Cold PORST (output) generation.

Warm PORST (input) active means that the reset is held active by external agents by pulling the PORST pin low. It is recommended to keep Warm PORST (input) asserted until all the external supplies are above their primary reset thresholds for the time duration defined as the tPOA parameter. During warm reset release, a positive load jump on IDD rail is expected, as documented in the datasheet.

RUN0 phase starts on warm PORST de-assertion. The firmware and start-up software programs are executed on CPU0 and CPUcs. The power supply is initialized and any user configuration is applied at this step. User program enables the ramp-up oscillator RMPOSC, which allows for gradually ramping-up of the operating frequency of the modules (including CPUx) in order to limit the load current profile. At the same time, PLL ramp-up is enabled and once completed the internal clocks can be switched to the PLL clock output.

Start-up after warm PORST

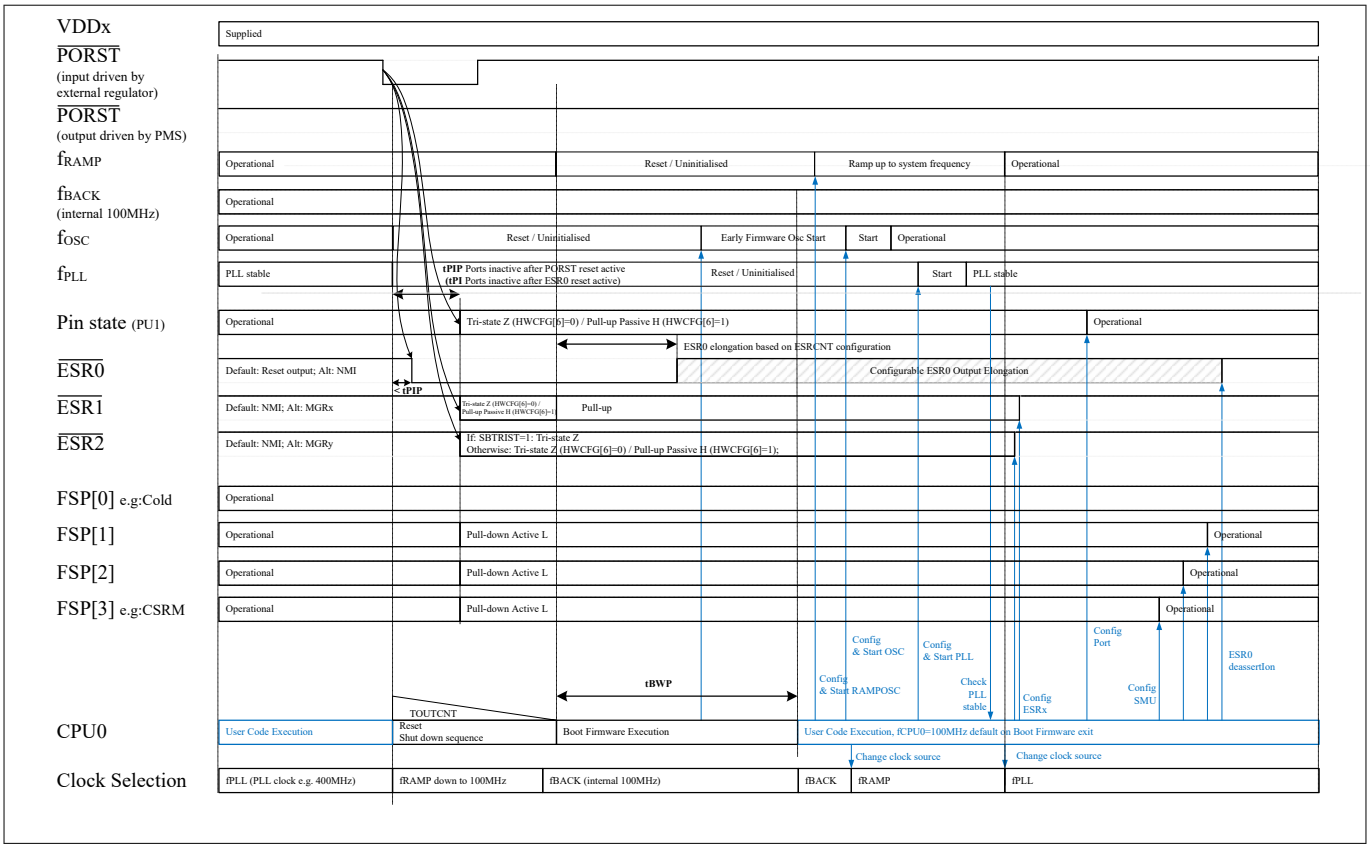


Figure 816 Start-up on warm power-on reset

The figure above illustrates the start-up of the microcontroller modules during and after warm PORST.

The warm PORST is driven low by the external agent, while internal PORST reset output remains released, power supplies remain stable within operational limits, and the primary voltage monitoring subsystem does not trigger Cold PORST.

37 System Mode Management (SMM)

The Power Management Subsystem (PMS) is operational through the reset assertion phase including internal clock fBACK.

Once PORST is asserted, shut down reset sequencing is activated which is completed within the time defined by TOUTCNT. After TOUTCNT expires, the reset is triggered.

The reset time of the PU1 or PU2 pins (whose default state is active) is defined by the time delay tPIP in the datasheet. The tPIP time also includes the delay for ESR0 and FSPx outputs if configured to be driven low. Default state of the common PU1 pads is defined by HWCFG[6] latched in the ALLTRIST/SBTRIST bit in register PAD_CON in PMS after initial cold start.

On PORST reset release after reset shut down sequence is completed, CPU0 and CPUcs start execution of the boot firmware for tBWP time duration. At boot firmware completion, CPU0 PC is set to the start-up address of the user program based on configuration and Oscillator and PLL configurations shall be reinitialized.

37.3.1.1 Start-up configuration

Hardware configuration (HWCFG) pins latch and start-up

The HWCFG pins are latched on LVDANA reset release into the PMS.PADSTAT register by PMS to determine the start-up hardware configuration and whether the internal EVRC regulator needs to be started.

***Note**: For supply start-up, cold PORST reset release and HWCFG latch on LVDANA reset release, please refer to the PMS chapter.*

Based on the STSTAT latched value, selection of Test Mode and Debug interface is decided.

Test mode and debug interface selection

On Power-On Reset release, the decision has to be made as to whether the device should operate in Test Mode or in RUN (User) Mode. The Test Mode is only for Infineon internal device testing and is not intended for an application. It is recommended to tie the $\overline{\text{TESTMODE}}$ pin permanently to an external pull-up in the application. The entry into Test Mode is made based on the latched value of the $\overline{\text{TESTMODE}}$ pin. The result is visible in the STSTAT.MODE bits. If device accidentally enters Test Mode after PORST, GPIO would be in reset state and pull devices would be activated as configured.

If the RUN(User) Mode was selected the next decision is which debug interface type shall be active for debugging until the next warm power-on reset. This decision is made based on the latched value of the pin $\overline{\text{TRST}}$ on $\overline{\text{PORST}}$ release.

Table 1635 Normal mode/test mode input selection

Pins	Description
$\overline{\text{TESTMODE}}$	Latched $\overline{\text{TESTMODE}}$ Pin on warm power-on reset release ($\overline{\text{PORST}}$ pin deassertion) The information if test mode is entered is visible in STSTAT.MODE1 and STSTAT.MODE2 bitfields. 0 A Test Mode can be selected 1 RUN (User) Mode is selected
$\overline{\text{TRST}}$	Latched $\overline{\text{TRST}}$ Pin on warm power-on reset release ($\overline{\text{PORST}}$ pin deassertion) The information about the latched value is visible in STSTAT.TRSTL bitfield. 0 The JTAG interface is active 1 The DAP interface is active

37.3.1.2 Software reset trigger

A software reset trigger can be initiated through SWRSTCON.SWRSTREQ bit-field. An application, system or Module Group Reset is triggered depending on the configuration in the RSTTRIGCTRLA.SW bit-field.

37.3.2 System modes

Different system modes ensure that the device operates with minimum required power for the corresponding device state. A progressive reduction in power consumption is achieved by invoking IDLE, SLEEP or STANDBY modes. The IDLE modes are specific to processing units where as SLEEP and STANDBY modes influence the complete system.

On start-up, the device first enters RUN0 mode with CPU0 and CPUcs running and all core power domains active. To reduce power consumption, the user may switch off power domains if present triggering a transition to RUN1 mode. Depending on configuration, the power domains shall again be switched off after every power domain cold reset type to transition to RUN1 mode. The clock is ramped-up thereafter and remaining CPUs and Peripherals are activated. The available power domains are documented in PMS chapter.

Processing units can be individually set into IDLE mode depending on application needs to be woken by interrupts, watchdog or other wake-up sources. In addition an arbitrary selection of peripherals and processing units can be clock gated using SLEEP modes depending on application needs to be woken up by interrupts, watchdog and other wake-up sources. The STANDBY modes are triggered to reach the lowest power states. Standby RAM is kept supplied and only SCR, RTC or WUT are kept active during STANDBY modes.

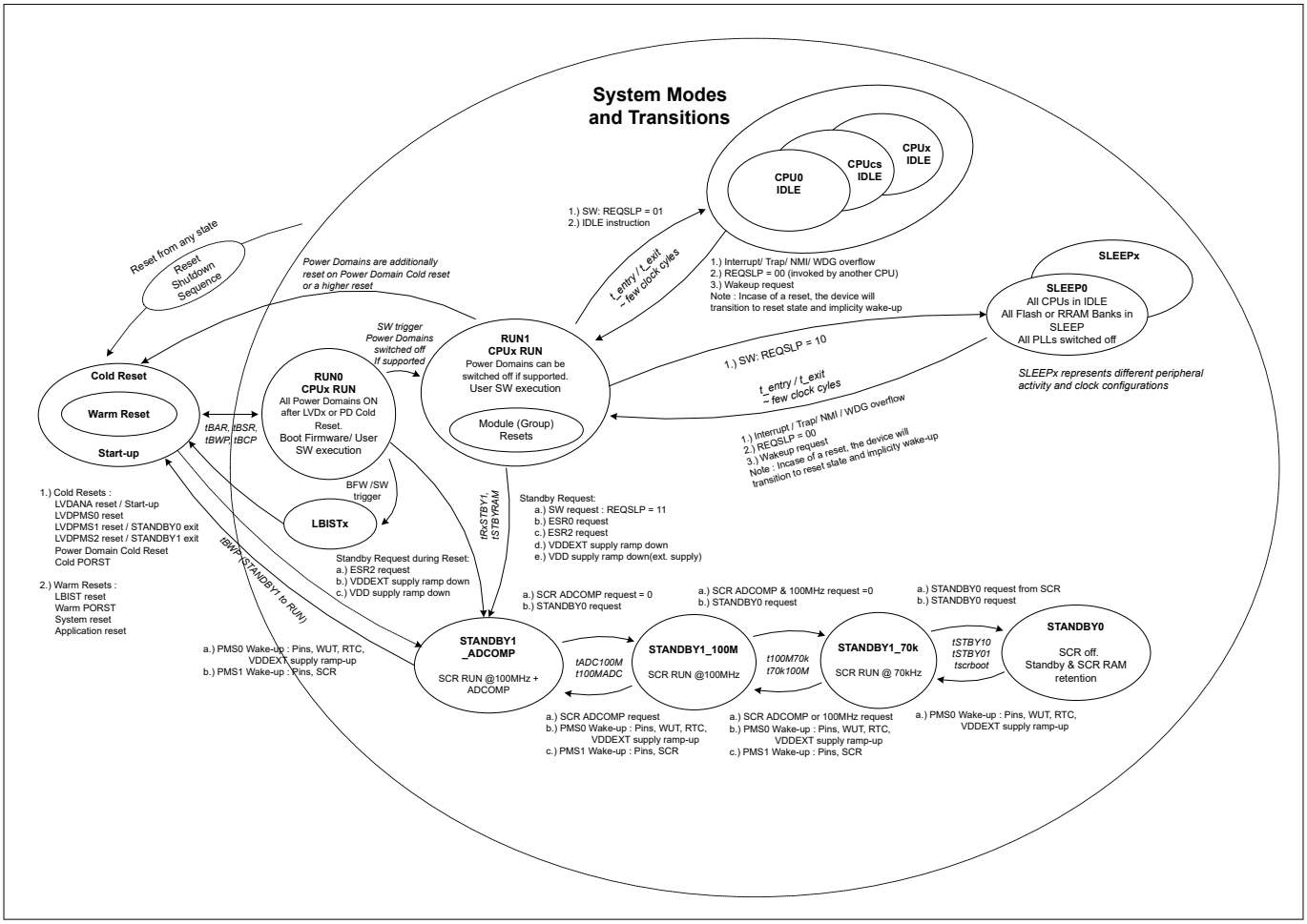


Figure 817 System states and transitions

As shown in the figure, there are two power modes available for each CPU:

- CPU Run Mode
- CPU Idle Mode

Table 1636 CPU modes

Mode	Description
Run mode	The CPU clock is active and code is being executed by CPU.
Idle mode	CPU may enter IDLE Mode in the following cases: • When a software Idle request is issued by setting register bits PMCSRn.REQSLP = 01_B because CPU has no active tasks to perform. • When a software Idle request (PMCSRx.REQSLP = 01_B) is issued by another CPU • WAIT instruction executed by CPU The CPU code execution is halted and CPU clock is disabled in Idle state. The peripherals continue to remain active. CPU RAM memories are accessible to other bus masters and peripherals. CPU may exit Idle Mode in the following cases: • When an interrupt occurs to a CPU returning the CPU to Run Mode • When a trap occurs like an NMI trap event • When the CPU watchdog or safety watchdog timer overflow events trigger an SMU alarm in turn leading to a CPU interrupt • When a MSB bit wrap of the CPU Watchdog counter takes place • When an application reset, system reset or any higher reset occurs • When a software run request (PMCSRx.REQSLP = 00_B is issued by another CPU

As shown in figure, there are three mode categories available for the system:

- Run Mode
- Sleep Mode
- Standby Mode

Furthermore, flexible reduction of power consumption is possible through the following measures:

- Reducing individual CPU power consumption by means of CPU clock scaling
- Disabling the module clock by setting bit DISR in respective module CLC register if the module need not be active at the current point of time
- Reducing the system frequency (e.g. fSPB; fSRI) without changing individual peripheral clocks
- Reducing individual peripheral clock frequency without changing system clock frequency. Main peripherals are provided with independent clocks separately from system SRI and SPB clocks

Table 1637 System modes

Mode	Description
Run mode	At least one CPU is not in Idle mode and Sleep or Standby mode has not been requested.

(table continues...)

Table 1637 (continued) System modes

Mode	Description
Sleep mode	System may enter Sleep Mode in the following cases: • When a software Sleep request is issued by setting PMCSRn.REQSLP = 10_B by the master CPU CPU code execution is halted and CPU Idle state is entered. Peripherals are set into sleep state if so configured in the respective CLCx.EDIS bit. Ports retain their earlier programmed state. System may exit Sleep Mode in the following cases: • When an interrupt or trap occurs on the CPU • When an NMI trap event takes place • When CPU watchdog or safety watchdog timer overflow events trigger an SMU alarm leading in turn to a CPU interrupt • When a MSB bit wrap of master CPU watchdog counter takes place • When an application reset, system reset or any higher reset occurs
Standby mode	System may enter Standby Mode STANDBY0 or STANDBY1 in the following cases if configured to do so: • Standby entry on a software standby request issued by setting PMCSRn.REQSLP= 11_B by the master CPU • Standby entry on an ESR1 (NMI) or ESR2 (implemented in PMS) assertion event • Standby entry on a primary or secondary under-voltage event during VDD supply ramp-down • Standby entry on a primary or secondary under-voltage event during VDDEXT supply ramp-down (Only VDDEVRSB supplied during STANDBY mode) The Standby domain constituting the Standby RAM, the 8 bit standby controller, standby ports including pin wake-up logic remain actively supplied. The power to the rest of the chip is completely switched off. VDD supply rail is switched off. SCR, WUT, RTC and Standby RAM supply maybe active or inactive during Standby mode (Refer PMS module chapter for details). System may exit Standby Mode in the following cases: • When a wake-up edge is detected on selected pins/ESRx • When a wake-up request is issued by the 8-bit Standby Controller (SCR) • When a wake-up request is issued by Wake-up timer or RTC • When PORST pin is asserted • When VDDEXT supply ramps up (Only VDDEVRSB supplied during STANDBY mode)

37.3.2.1 IDLE mode

In case there are no active tasks to perform, a CPU may be requested to enter Idle mode during run time by writing to the respective PMCSRx register and setting the bit-field REQSLP = 01_B.

Entering Idle mode

The following events can invoke a CPUx Idle request:

- CPUx setting itself in Idle by writing its own PMCSRx register: Idle mode may also be simultaneously triggered for additional CPUs based on PMSWCR1.CPUIDLSEL configuration
- Another CPU[y] setting CPUx into Idle: The Idle request is issued immediately on setting PMCSRx.REQSLP = 01_B

The CPU watchdog may be disabled or slowed down by reprogramming the timers before triggering Idle request through Software. On an Idle request, the CPU finishes its current operations and sends an acknowledge signal back to the SMM module. It then enters an inactive state in which the CPU clocks and the respective RAM memory units are clocked off.

State during Idle mode

During Idle Mode, memory accesses to the CPU memories from other bus masters cause these units to wake-up automatically to handle these transactions. When memory transactions are complete, the CPU memories return to Idle state again. Once Idle mode is entered, the state is reflected in the PMCSRx.PMST status bits.

Exiting Idle mode

In Idle mode, the CPU will return to Run mode in response to the following wake-up events:

- An interrupt request from a source mapped to the CPU.
- An NMI trap request is received to wake-up the corresponding CPUs
- A MSB bit wrap of the corresponding CPU Watchdog counter occurs. A wake-up signal is provided by the respective watchdog toward CPU on a timer-value transition $7FFF_H$ to 8000_H. Further details are documented in Watchdog Timer Unit chapter
- Setting the register bits PMCSRx.REQSLP = 00_B to set the CPUx into Run mode by another processing unit
- Exit owing to a reset event: The system enters reset state on an Application reset or any higher reset.

If it is woken by a watchdog timer overflow event routed through the SMU to the CPU, or by an NMI or an interrupt, the CPU will immediately vector to the appropriate interrupt or trap handler.

CPU module reset will not result in exit from Idle mode if it was already in Idle state. An explicit wake-up event has to happen before the CPU is in the Run state again.

Table 1638 CPU[x] Idle mode entry sequence, behavior and status indication

Condition	CPU sets only itself into Idle CPU[x] writes PMCSR[x].REQSLP = 01_B	Another CPU setting a CPU into Idle CPU[y] writes PMCSR[x].REQSLP = 01_B	CPU sets all CPUs into Idle CPU[y] writes PMCSR[y].REQSLP = 01_B
CPU entry event to idle mode	A CPU[x] should be able to set itself into Idle. CPUIDLSEL = 0	A CPU[y] or (other CPUs should be able to set CPU[x] into Idle if they have the access rights. CPUIDLSEL = 0	CPUIDLSEL = y+1 is already set by a CPU having access rights.

(table continues...)

Table 1638 (continued) CPU[x] Idle mode entry sequence, behavior and status indication

Condition	CPU sets only itself into Idle CPU[x] writes PMCSR[x].REQSLP = 01_B	Another CPU setting a CPU into Idle CPU[y] writes PMCSR[x].REQSLP = 01_B	CPU sets all CPUs into Idle CPU[y] writes PMCSR[y].REQSLP = 01_B
CPU status during idle mode	CPU[x] in IDLE PMCSR[x].REQSLP= 01_B PMCSR[x].PMST= 011_B PMSTAT0.CPU[x] & LS= 0_B		All CPUs in IDLE PMCSR[ALL].REQSLP = 01_B PMCSR[ALL].PMST= 011_B PMSTAT0.CPU[ALL] & LS= 0_B
CPU status after idle mode exit	PMCSR[x].REQSLP= 00_B PMCSR[x].PMST= 001_B PMSTAT0.CPU[x] & LS= 1_B		PMCSR[ALL].REQSLP= 00_B PMCSR[ALL].PMST= 001_B PMSTAT0.CPU[ALL] & LS= 1_B

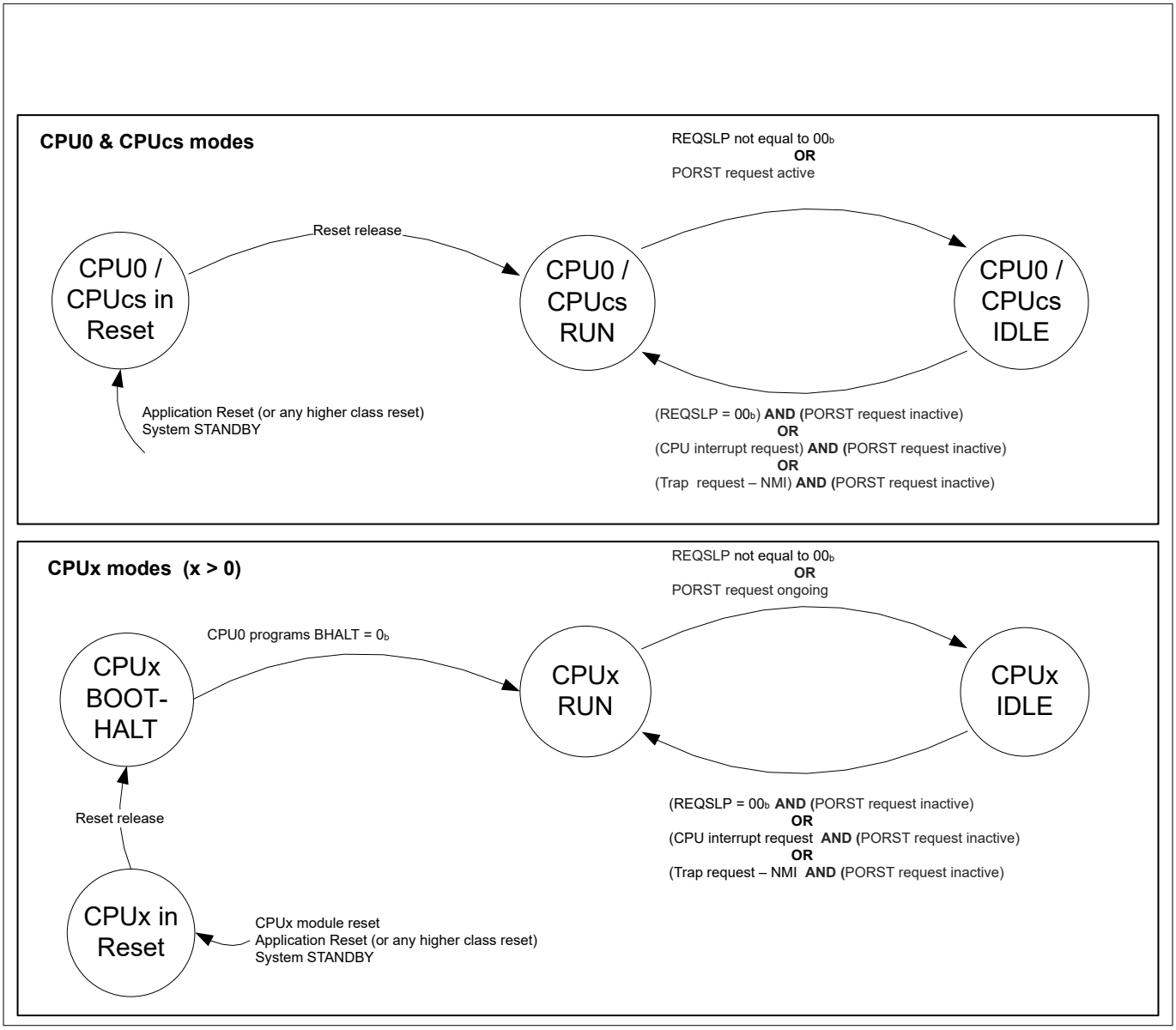


Figure 818 Idle modes and transitions

37.3.2.2 SLEEP mode

Entering SLEEP mode

System may be requested to enter Sleep mode via software by writing to the CPU's PMCSRx register and setting the bit field PMCSRx.REQSLP = 10_B.

An example sequence for Sleep mode is enumerated below:

- The Sleep mode entry shall be enabled in respective module configuration registers. The respective module CLCx.EDIS register bit shall be cleared for all peripherals intended to be inactive in Sleep mode
- The respective CPU watchdogs may be disabled or re-configured for slower modes
- Master CPU code execution maybe switched from Flash[46] or RRAM[46] to RAM if required. Individual Flash or RRAM banks may be explicitly set into (bank) sleep state as documented in NVM or NVMR and DMU sections
- The analog modules TMADC and DSADC[46] maybe switched off in Sleep state. The high speed communication modules and HSPHYx modules[46] maybe switched off in sequence if not required to be active in Sleep state
- It should be ensured to select the individual clocks from Clock Control Unit for peripherals which need to remain active during Sleep mode. Certain communication and timer peripherals have clocks independent from the system frequencies, namely SRI and SPB clocks, to allow the possibility to bypass the system PLL. In such cases, the System PLL is switched into bypass mode and consequently the DCO would be switched off. Peripheral clock will continue to run clocking the modules active during sleep mode. The system clock frequencies, namely SRI and SPB clocks, maybe then reduced to the minimum possible values through the low power divider and Kx divider to reduce the current consumption. In some cases the respective peripherals may be clocked directly from external crystal / resonator depending on application
- The interrupt control unit provides the infrastructure for wake-up from sleep state and therefore need to be kept active with a minimum SPB bus frequency. The respective module wake-up interrupts are routed to CPU to wake-up on an interrupt event
- Sleep Mode may be entered as a unanimous decision of all the CPUs when all of the PMCSRx registers in the system request it AND PMSWCR1.CPUSEL = 111_B. Sleep Mode may also be entered based on a singular decision of a master CPU based on the configuration of the CPUSEL register. The master CPU watchdog may also be disabled or slowed down before issuing a Sleep request

State during SLEEP mode

Sleep Mode is disabled for a module if the respective module CLCx.EDIS bit is set. The sleep request is ignored in this case and the corresponding unit continues operation as configured. If CLCx.EDIS is cleared, the clock of the module is gated. The current consumption during Sleep mode is documented in datasheet.

***Note**: The module list in the table below is representing the whole AURIX™ TC4xx family. If respective module or module function is not available in a given product then the information is not relevant*

Table 1639 Module activity and configuration during sleep mode

Module active during sleep mode	Module and clock state during sleep mode
PPU	PPU continues to run during Sleep mode. If required, it shall be ensured that PPU is set into Idle before Sleep mode is triggered
CPUcs	CPUcs is put into Idle state. CPUcs wakes-up on respective interrupt.
CPU	All CPUs are put into Idle state. CPU watchdogs maybe active with slower configuration. CPUs wake-up on respective interrupt

(table continues...)

[46] Not available on all devices of the AURIX™ TC4xx family

Table 1639 (continued) Module activity and configuration during sleep mode

Module active during sleep mode	Module and clock state during sleep mode
Flash	Flash banks may be selectively set into bank sleep state. ***Note**: For details about "deep sleep" refer to the NVM section of the user manual*
Ports	All Ports retain their earlier programmed state.
CAN ASCLIN QSPI I2C	Peripheral PLL maybe kept active providing the minimal required module clock for function (for example - f_{MOD} = 5 MHz - 40 MHz). Module may alternatively run on f_{OSC} allowing also complete switch off of the Peripheral PLL. System PLL may be switched into low power mode if supported. $f_{SRI}$and f_{SPB} clocks can be reduced (for example - ~1-2 MHz) via LPDIV and / or Kx dividers. System PLL may also be switched off and switched to Back-up clock depending on application. Module FIFO and DMA allows autonomous handling of messages without involvement of CPU. External communication may selectively remain active.. QSPI may need to service external watchdog if required by application. A timer module maybe used to trigger DMA or fill the FIFO allowing autonomous handling of messages without involvement of CPU. Input messages are monitored and wake-up interrupt is triggered. Wake-up on CAN wake-up message identifier via CAN interrupt or Wake-up on LIN wake-up frame via ASCLIN interrupt or Wake-up in case of fault diagnosis of external device via a QSPI interrupt
HSPHYx	HSPHY modules are inactive.
GTM STM eGTM	Peripheral PLL is disabled. System PLL is active and provides the module clock (for example -f_{MOD} ~ 1-2 MHz). $f_{SRI}$and f_{SPB} clocks are reduced to (for example - ~1-2 MHz) via LPDIV and / or Kx dividers. It is recommended to switch off the GTM/eGTM module completely and use smaller timer modules like RTC, STM or WUT during Sleep state to reduce power consumption. Wake-up on timer overflow or capture event via GTM/eGTM interrupt or Wake-up on timer overflow via STM interrupt
xADC	All ADC units are switched off. Peripheral PLL is disabled. System PLL is active and provides the module clock (for example - f MOD ~f SPB ~ 1-2 MHz). SCR.ADCOMP units is used to handle ADC conversions and wake-up interrupt is issued by the SCR
Pin Wake-up ESR1 ESR2	Infrastructure modules SMM, CLOCK and Interrupt Router is kept active. Peripheral PLL may be disabled. $f_{SRI}$and f_{SPB} clocks reduced to (for example - ~1-2 MHz) via LPDIV / Kx dividers. System PLL may also be switched off and switched to Back-up clock depending on application. Wake-up on Edge / Level detection on pin routed to ERUx, ESR1, ESR2 or PORT module via interrupts or polling Port registers on an active timer interrupt
PMS WUT RTC	PMS and SCR is kept active Peripheral PLL is disabled and System PLL is bypassed. $f_{SRI}$and f_{SPB}clocks reduced to (for example - ~1-2 MHz) via LPDIV / Kx dividers. System PLL may also be switched off as module clock is derived from Back-up clock Wake-up on timer overflow via WUT interrupt. Wake-up on RTC Compare 0 or Compare 1 interrupt

(table continues...)

Table 1639 (continued) Module activity and configuration during sleep mode

Module active during sleep mode	Module and clock state during sleep mode
RRAM	RRAM banks may be selectively set into sleep state.

Exiting SLEEP mode

The system will exit Sleep mode on any wake-up event that causes any master CPU to exit Idle Mode depending on CPUSEL configuration. Only the master CPU associated with the interrupt wake-up event would be set into Run mode (REQSLP = RUN, PMST = RUN). Other CPUs will remain in Idle (REQSLP = SLEEP, PMST = IDLE)[47]. An NMI trap event will wake-up the respective CPU as configured in TRAPS_CPU_DISx and TRAPS_CS_DIS registers. A MSB bit wrap of the corresponding master CPU Watchdog counter would also wake-up the master CPU. The response of the CPU to being woken up from Sleep Mode is also the same as for Idle Mode. Peripheral units that have entered Sleep Mode will switch back to their selected Run Mode operation. Wake-up latency from Sleep mode depends mainly on the extent of clock ramp-up required after wake-up keeping the load jump constraints. If PLL is switched off, the wake-up latency would include the time to ramp and lock the PLL. The sequence after wake-up is dependent on the entry sequence and mainly constitutes ramping back the clock system, activating analog and Flash or RRAM modules[46], switching from RAM to Flash or RRAM execution and activating additional CPUs. The time taken between interrupt trigger availability until CPU has woken up and is executing next instruction is less than 3 SPB + 20 SRI clock cycles.

Table 1640 System sleep mode entry sequence, behavior and status indication

Condition	Master CPU[x] writes PMCSR[x].REQSLP = 10_B	All CPUs write respective PMCSR[y].REQSLP = 10_B
System entry event to sleep mode	A master CPUx (any CPU except CPUcs) should be able to trigger SLEEP mode if CPUSEL = x+1_B is already set by a CPU having access rights. PPU Idle request is not triggered.	CPUSEL = 111_B is already set by a CPU having access rights before. PMCSR[y].REQSLP= 10_B System enters SLEEP mode if all CPUs have requested for SLEEP entry. PPU Idle request is not triggered.
Status during sleep mode	All CPUs in IDLE. PMCSR[x].REQSLP= 10_B PMCSR[x].PMST= 100_B PMSTAT0.CPU[x] & PMSTAT0.CPU[x]LS=0_B Status of other CPUs - CPU[y] PMCSR[y].REQSLP= 10_B PMCSR[y].PMST= 100_B PMSTAT0.CPU[y] & PMSTAT0.CPU[y]LS= 0_B	All CPUs in IDLE. PMCSR[y].REQSLP= 10_B PMCSR[y].PMST= 011_B PMSTAT0.CPU[y] & PMSTAT0.CPU[y]LS= 0_B

(table continues...)

47 In case the MSB of (non-master) WDT is already set during SLEEP mode the corresponding CPU transitions from IDLE to RUN instead of staying in IDLE

46 Not available on all devices of the AURIX™ TC4xx family

Table 1640 (continued) System sleep mode entry sequence, behavior and status indication

Condition	Master CPU[x] writes PMCSR[x].REQSLP = 10_B	All CPUs write respective PMCSR[y].REQSLP = 10_B
Status on exiting sleep mode	Wake-up on Master CPU leading to exit from its IDLE state leads to its transition to RUN state. PMCSR[x].REQSLP= 00_B PMCSR[x].PMST= 001_B PMSTAT0.CPU[x] & PMSTAT0.CPU[x]LS= 1_B Other CPU[y] remain in IDLE if not explicitly woken up[1)] PMCSR[y].REQSLP= 10_B PMCSR[y].PMST= 011_B PMSTAT0.CPU[y] & PMSTAT0.CPU[y]LS= 0_B	Wake-up event of respective CPU[x] sets it to RUN state PMCSR[x].REQSLP= 00_B PMCSR[x].PMST= 001_B PMSTAT0.CPU[x] & PMSTAT0.CPU[x]LS= 1_B Other CPU[y] remains in IDLE if not explicitly woken up PMCSR[y].REQSLP= 10_B PMCSR[y].PMST= 011_B PMSTAT0.CPU[y] & PMSTAT0.CPU[y]LS= 0_B

1) In case the MSB of (non-master) WDT is already set during SLEEP mode the corresponding CPU transitions from IDLE to RUN instead of staying in IDLE

37.3.2.3 Low frequency applications

If an application has the need for a reduced power consumption but does not want to go into a SLEEP mode the restrictions mentioned in this section need to be considered.

Certain applications may require reduced system frequencies, namely f_{SRI} , f_{SPB} and f_{CPB}[48)]. to a minimum allowed value according to the datasheet. If therefore the system frequency f_{pll0} is being configured below 100 MHz there are specific constraints applicable as described in this section.

***Note**: Radar devices of the AURIX™ TC4xx shall not run applications on system frequencies below 100 MHz*

Requirements for low frequency mode

For low frequency application the following conditions shall be met.

- The System PLL shall be configured using the N and P divider to the required target output frequency which shall not be divided further down for example for f_{SRI}
- The system frequency clock dividers shall not be configured to reduce the System PLL output f_{pll0} further down. This means a division by 1 shall be applied
- The Peripheral PLL shall be configured to nominal output values f_{pll1}=160 Mhz, f_{pll2}=200 Mhz, f_{pll3}=200 Mhz

Module restrictions for system frequencies below 100 MHz

A reduced frequency can have limiting effects on different IPs as well as on the clock ratios between different clock domains.

If the system frequency is being reduced below 100 MHz the SystemPLL shall be configured to the required output frequency f_{pll0} while all dividers for the system clocks are set to 1 to ensure a 1:1 ratio between all system frequencies. The respective dividers are SPBDIV, CPBDIV[48)], TPBDIV, SRIDIV, FSIDIV, FSI2DIV, STMDIV, , GETHDIV; RCBDIV, GTMDIV[48)]; eGTMDIV[48)], MCANHDIV, LETHDIV, CANXLHDIV.

If a clock is not needed in this case it shall be switched off.

For 400 MHz devices, if the CSRM subsystem is used, the minimum allowed system frequency f_{pll0} is 50 MHz to ensure the minimum frequency of CPU-CS is valid.

48 Not available on all devices of the AURIX™ TC4xx

37 System Mode Management (SMM)

The Peripheral clock tree (including peripheral PLL and Peripheral clock dividers) shall stay on nominal settings with the exception of f_{MCAN} and f_{CANXL} due to the dependency on the host frequency.

***Note**: The restrictions given by the CLOCKING chapter are still applicable as well as the related information given in the datasheet*

***Note**: The module list in the table below is representing the whole AURIX™ TC4xx family. If respective module or module function is not available in a given product then the information is not relevant*

Table 1641 Module restrictions for system frequencies below 100 MHz

Module	Restriction
HSSL	HSCT shall not be used for cases of f_{SRI} below 100 MHz
xSPI	The XiP Write feature[1)] shall only be used if sclk_out is lower or equal to f_{SRI}. To achieve this the xSPI internal clock dividers shall be used
CANXL	CANXL IP shall only be used with a minimum host frequency of f_{CANXLH} 80 MHz in case it is used for CANXL protocol
MSC	FAST mode shall not be used for system frequencies below 100 MHz Please refer to the baud rate configuration table of the MSC chapter for more details on baud rates for low frequencies
PSI5-S	PSI5-S shall not be used for cases of f_{SRI} below 100 MHz
NVM	The bank sleep feature shall not be used for f_{SRI} below 100 MHz
NVMR	For effects on writing operations for f_{FSI} below 100 MHz please refer to the datasheet of the respective derivative Reading operations can have a reduced performance up to 80%
HS communication including HSPHY LETH GETH PCIe xSPI	High speed communication which involves HSPHY shall not be used for cases of f_{SRI} and f_{SPB} below 100 MHz
DRE	DRE shall not be used for system frequencies f_{SRI} and f_{SPB} below 100 MHz

1) feature is not available on all derivatives of the AURIX™ TC4xx

37.3.3 External interface

This chapter describes the behavior of main external interface pins supporting reset function, interrupt/NMI function and error feedback function.

PORST bidirectional reset pin function

The PORST pin is the main reset input pin of the device and supports following functions:

- If PORST pin is externally asserted, a warm PORST reset type is triggered
- The PORST pin is driven low by PMS on supply fail detected by primary supply monitors leading to cold PORST reset state

ESR0 pin function

The ESR0 pin provides external indication of internal resets and can be used as a reset input or output pin. Following functions are supported by ESR0 pin:

- The pin provides reset indication of a PORST, System or Application Reset. The pin is driven low by default on an internal application reset. It is used to reset the external device immediately when microcontroller is set into reset state. The ESR0 pin de-assertion can be delayed to microcontroller reset release by configuring an elongation counter
- The pin can trigger a Trap request
- The pin can trigger an Application or System or Module Group Reset
- The pin can trigger a wake-up event from Standby Mode in PMS if VDDEXT is supplied during Standby Mode
- The pin can be used as a GPIO pin

***Note**: If configured as GPIO the ESR0 will keep its previously configured state until the reset assertion of the System Reset at the end of the shutdown sequence. In this case the ESR0 will start the reset indication as configured in ESR0CNTCTRL with System Reset assertion.*

ESR1 pin function

The ESR1 interface pin is configured as an Trap/Interrupt trigger by default. This is connected to external regulator or safety monitor to trigger communication with device to service an external alarm event. Following functions are supported by ESR1 pin:

- The pin is configured to trigger a trap request by default after reset. The trap/NMI request is used together with SPI communication with the external regulator/monitor to complete the communication transaction
- The pin can trigger a wake-up event from standby mode in PMS if VDDEXT is supplied during Standby Mode
- The pin can trigger an Application or System or Module Group Reset
- The pin can be used as a GPIO pin
- The pin can be used as trigger for standby entry

***Note**: In debug mode the standby entry by ESR1 can be blocked by CPUs being in debug halt mode. If in that case an unconditional standby entry is needed the PMSWCR1.IRADIS shall be used*

ESR2 pin function

The ESR2 interface pin is implemented in the standby domain to ensure reset, wake-up and trap functionality also in the lowest power-down modes. Following functions are supported by ESR2 pin:

- The pin can trigger a trap request
- The pin can trigger an Application or System or Module Group Reset
- The pin can trigger a wake-up event from Standby modes in PMS. ESR2 can be used to trigger ramp-up of external supply on a wake-up event
- External reset indication to external driver and memory devices implemented in PMS

- The pin can be used as a GPIO pin
- The pin can be used as trigger for standby entry

***Note**: GPIO registers implemented in PORTS module, instance P33*

Error pins - FSP0, FSP1, FSP2 and FSP3

Error pins are asserted low or stop toggling when there is a critical alarm which needs to be debounced externally through a warm PORST event or supply cycling of the microcontroller. Multiple FSP pins are provided to differentiate the external reaction of the monitors. FSP0 is in general configured to trigger warm PORST reaction. The other FSP pins may be configured to trigger a supply re-ramp and a cold PORST event if the alarm is supply related.

37.3.3.1 PORST pin

The PORST pin is the main reset input pin and triggers a warm power on reset of the device. It shall be connected to the external regulator or supply monitor. If this pin is left open then the default behavior is that the device remains in a reset state.

The PORST pin supports following functions:

- Warm PORST reset input function: It is expected that external monitor or regulator asserts PORST pin if operating conditions are violated, namely supply voltages. It is also expected that the external monitor asserts PORST pin if microcontroller is stuck and not responding or the external watchdog was not serviced by the microcontroller.
- If PORST pin is externally asserted, a warm PORST reset is triggered. It is ensured that all warm resets are sequenced internally and RAM contents are not affected. Reset is executed only after RAM write operations are completed and processing units have reached IDLE state.
- If the PORST pin is kept asserted externally after a system start-up and supply voltages are within operating range, then the warm PORST reset state will be kept extended until PORST pin is de-asserted.
- The PORST pin is driven low by PMS on a cold PORST reset trigger by activating a strong pull-down capable of sinking I_{PORST} current as documented in the datasheet. The strong pull-down is deactivated after system is back to RUN0 mode. During initial Boot strong pull-down is only released on reaching RUN0 (Operation) mode. During Standby state and during transition from Standby state to RUN0 (Operation) mode, strong pull-down is not activated.
- The PORST pin has an integrated weak pull-down function to ensure that device stays in reset state if pin connection or PCB routing to the PORST pin is broken. The weak pull-down is active in all system states except Standby mode. In Standby mode, the weak pull-down is deactivated to reduce standby current. The weak pull-down is activated by default during Supply Start-up phase and during Standby-exit.
- Multiple masters can trigger PORST if connected together in an open-drain circuit with an external pull-up. Depending on push-pull or open drain implementation of reset output by external regulator, care should be taken in the dimensioning of the pull-up and series resistances on PORST connection.
- The PORST pin has a basic analog spike filter to filter out noise as documented in the datasheet. It supports TTL input level for 5 V operation and degraded TTL levels for 3.3 V operations as documented in the datasheet.
- In addition to the inherent analog spike filter, PORST pin supports a digital filter that can be additionally activated via PMS.PAD_CON.PORSTDFEN register bitfield to filter higher noise at input.

37.3.3.2 External service request interface (ESRx)

The external service request ESRx pin functions are described in this section.

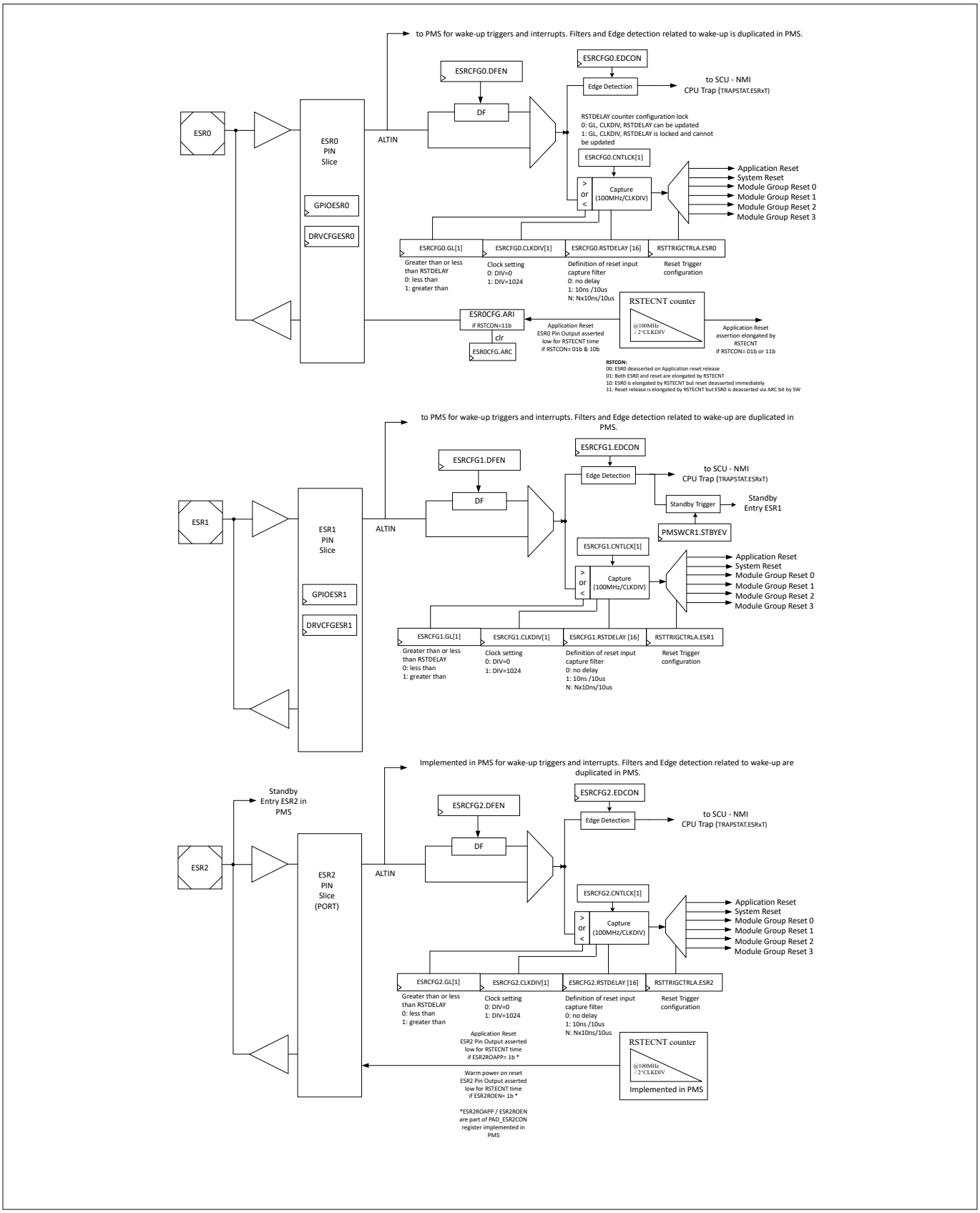


Figure 819 ESR pins connectivity and behavior

37.3.3.2.1 ESRx pin reset input

An $\overline{ESR0}$/$\overline{ESR1}$/$\overline{ESR2}$ pin reset request trigger can lead to a System Reset or an Application Reset or any Module Group Reset. The type of the reset to be triggered on an external ESRx pin low assertion is configured through RSTTRIGCTRLA.ESRx.

By default after a power-on reset, ESR0 is configured to trigger an application reset as shown in the following figure. An external device may extend an existing reset condition by holding the ESR0 pin active if RSTCON=00b or RSTCON=01b.

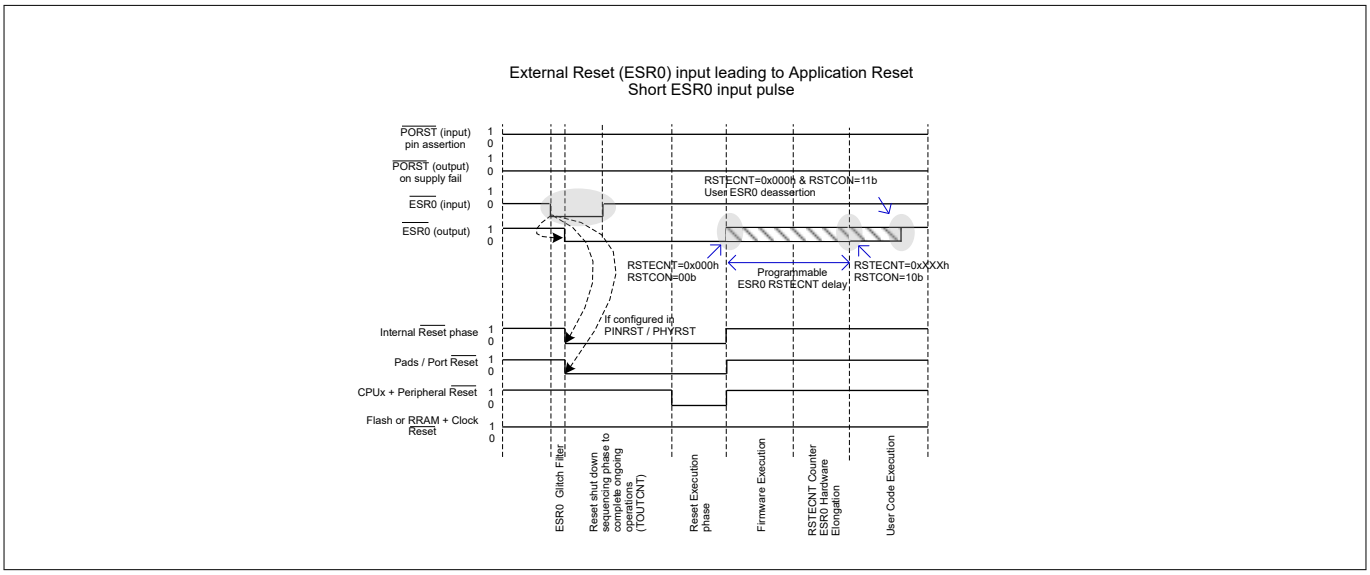


Figure 820 ESR0 reset input behavior with short input pulse

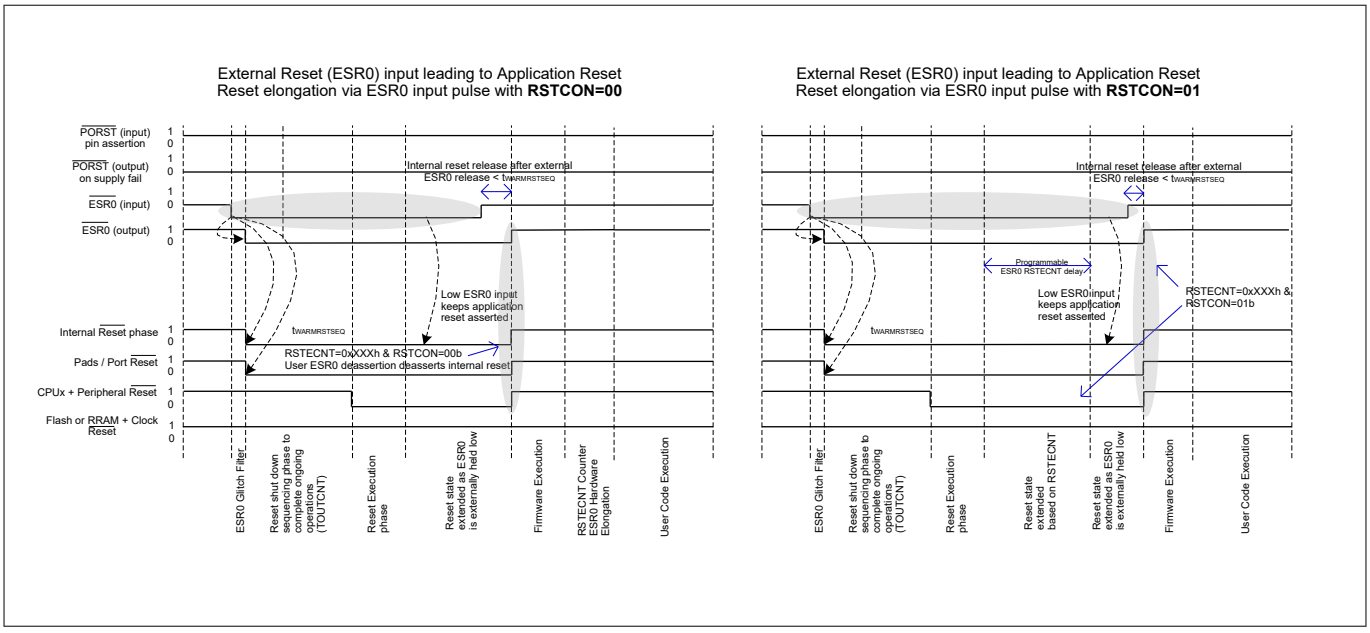


Figure 821 ESR0 reset input with long input pulse

The input signals $\overline{ESR0}$/$\overline{ESR1}$/$\overline{ESR2}$ can be filtered. The filter is enabled by register bit ESRCFGx.DFEN . If the filter is enabled, then pulses less than 10 ns cannot pass the filter and pulses longer than 100 ns will always result in forwarding the trigger to the capture block for fSPB ≥ 20 MHz in RUN mode.

The ESRCFG register defines the reset input capture delay behavior. The reset input low pulse duration is captured and compared to be greater than or less than RSTDELAY value and clocked as configured in CLKDIV bit-field. The CNTLCK bit-field allows update of the parameters and need to be explicitly locked for the updating

the parameters. The intention of this configurability is to use the same ESR pin to generate simultaneously NMI or Trap request and Reset by differentiating the time durations of the respective events.

By default after reset, ESRx pads have pull-ups but during LBIST,the ESRx pin is changed to a weak pull-down.

The GPIO pin functionality of $\overline{\text{ESR0}}$ and $\overline{\text{ESR1}}$ pins is configured through registers GPIOESRx and DRVCFGESRx. The GPIO pin functionality of $\overline{\text{ESR2}}$ is configured through the PORT module registers of instance P33. The pad control can be configured independently for each pin as described in the PORT chapter. The configuration comprises:

- Selection of driver type (open-drain or push-pull)
- Enabling of the output driver (input and output capability)
- Enabling of internal pull-up or pull-down resistance

***Note**: If ESR2 is used as reset input it shall not be used for any other functionality (e.g. reset output indication) in parallel to avoid a permanent reset state of the device*

37.3.3.2.2 ESRx pin reset output

The external pins $\overline{\text{ESR0}}$/$\overline{\text{ESR2}}$ can provide a reset output indication (strong pull-down) for Application Resets.

Some applications require a reset out signal with a configurable reset elongation, as longer initialization time may be required for certain devices. For example memory devices like external RAM, Flash, and EEPROM devices may require that reset be held longer and released with a delay after the main microcontroller reset has been released.

Register ESR0CFG.ARI determines the output of the ESR0 pin when configured as functional reset output. ARI is set automatically when an application reset trigger is detected and cleared after RSTECNT time expiry or by Application software. Note that an Application Reset also occurs on a System Reset or Power-On Reset.

While the ARI bit is set, the configured ESR0 pin drive active low. A subsequent write to ESR0CFG.ARC clears the ARI bit, which de-asserts the ESR0 output in case the pin needs to be cleared by user software depending on ESR0CNTCTRL.RSTCON configuration.

The minimum duration of the ESR0 pulse on cold power-on reset is indicated by datasheet parameter t_{EVRPOR} and the additional elongation is defined through the reset value of ESR0CNTCTRL.RSTECNT and CLKDIV registers.

The application output selection and the default pull behavior is configured in DRVCFGESR0.MODE bit-field. After a system reset, the ESR0 pin is configured as open drain reset output driving application reset with weak pull-up active. When the same ESR pin is configured as a PORT output, its state can instead be controlled directly by application software so that the application has the possibility to reset external devices.

On warm PORST assertion, the ESR0 output elongation behavior depends on the count value written earlier to the ESR0CNTCTRL.RSTECNT and CLKDIV registers, which controls ESR0 elongation duration depending on the ESR0CNTCTRL.RSTCON configuration. An external device may extend an existing reset condition by holding the ESR pin active if ESR0 is simultaneously configured as reset input depending on ESR0CNTCTRL.RSTCON configuration.

When ESR0 pin output is being asserted, ESR0 reset input function is inactive or masked irrespective of configuration.

The ESR0CNTCTRL.RSTECNT reset counter can be used for the following purposes:

- Controlling the length of the internal reset release
- Configuring the reset length in a way that the reset outputs through the ESRx pins match with the reset input requirements of external blocks connected with the reset outputs

The reset counter RSTECNT is a 12-bit counter counting down from the reload value defined by ESR0CNTCTRL.RSTECNT based on the divided 100 MHz back-up clock / 2^CLKDIV. The counter is started by the control block after a reset request trigger condition becomes active and as soon as the system has been

brought into a stable state (or TOUTCNT has expired). If a reset request trigger is asserted continuously longer than the complete count-down process then the reset will not be de-asserted until the reset request trigger is removed. In this case the counter is not restarted. The control block simply holds the reset asserted until the reset request trigger has been de-asserted depending on ESR0CNTCTRL.RSTCON configuration.

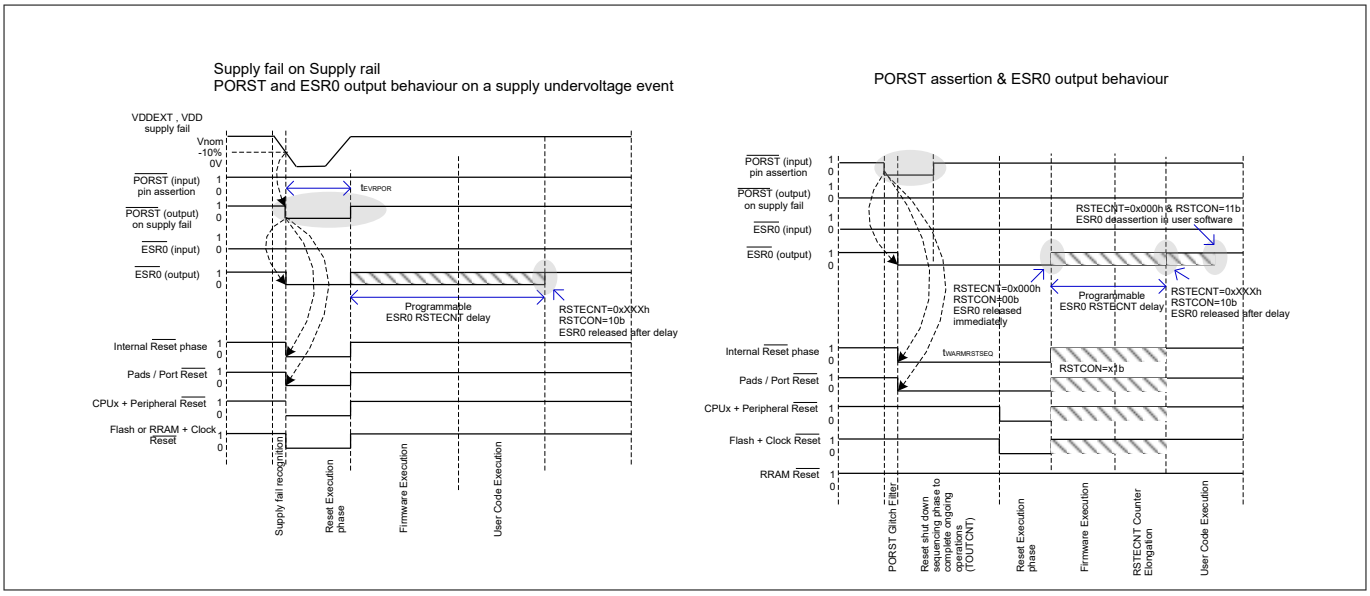


Figure 822 ESR0 reset output behavior

If ESR0 reset input and output function is configured simultaneously, it has to be ensured that the device does not go into a permanent reset state. This is realized by releasing the reset when ESR0 input is externally deasserted after ESR0 output buffer was disabled. This is only supported in specific RSTCON configurations as documented below. This acts as a time hysteresis as the pin will be kept low owing to external capacitance on the ESR0 pin for some more time after it is internally released.

Table 1642 ESR0 reset output behavior

Configuration	Reset release elongation	ESR0 pin output elongation	Simultaneous ESR0 reset input elongation	Update during start-up by Boot Firmware
RSTCON = 00	Immediate reset release	Immediate ESR0 deassertion	After reset sequencing and assertion, ESR0 output is disabled and later when ESR0 input goes high, reset is released	Firmware update of ESR0CNTCTRL effective only on next reset cycle.
RSTCON = 01	Elongation by RSTECNT time	Elongation by RSTECNT time	After reset sequencing and assertion and RSTECNT has expired, ESR0 output is disabled and later when ESR0 input goes high, reset is released	Firmware update of ESR0CNTCTRL effective only on next reset cycle.

(table continues...)

Table 1642 (continued) ESR0 reset output behavior

Configuration	Reset release elongation	ESR0 pin output elongation	Simultaneous ESR0 reset input elongation	Update during start-up by Boot Firmware
RSTCON = 10	Immediate reset release	Elongation by RSTECNT time	Simultaneous reset input feature is not supported	Firmware update of ESR0CNTCTRL effective on the current reset cycle.
RSTCON = 11	Elongation by RSTECNT time	ESR0 pin deasserted by user by writing to ARC bit	Simultaneous reset input feature is not supported	Firmware update of ESR0CNTCTRL effective only on next reset cycle.

The ESR2 pin can also provide a reset output indication for warm power-on reset (warm PORST) or for application reset as configured in PMS.PAD_ESR2CON.

***Note**: ESR2 reset output indication configuration in PMS.PAD_ESR2CON is reset on LVDPMS2 reset. For details about LVDPMS2 reset refer to PMS chapter*

By default the ESR2 pin configured as GPIO and not as reset output indication.

If needed an additional elongation for the warm power- on reset output indication is defined through the value of PMS.PAD_ESR2CON.RESTECNT and PMS.PAD_ESR2CON.CLKDIV as well as PMS.PAD_ESR2CON.ESR2ROEN register. For application reset the elongation is dependent on PMS.PAD_ESR2CON.ESR2ROAPP as well as PMS.PAD_ESR2CON.RESTECNT and PMS.PAD_ESR2CON.CLKDIV.

ESR1 and ESR2 behavior during reset execution

ESR1

The ESR1 pin will indicate a weak pull-up or tristate depending on HWCFG6/PMS.PAD_CON.ALLTRIST as defined as PU1 in the datasheet for the case of cold power-on resets and warm power-on resets. After successful execution of the reset shutdown sequence, the ESR1 behavior will be pull-up with system reset assertion.

In case of a system reset execution the ESR1 will always have internal pull-up active. During the execution of the reset shutdown sequence the ESR1 will keep its previously configured behavior.

The behavior for warm power-on reset and system reset execution can be seen in the figure below.

***Note**: If the duration of the application reset is elongated by configuration of the ESR0CNTCTRL the ESR1 will already behave in pull-up behavior at the beginning of the elongation, so the Internal reset elongation is without effect on the ESR1 behavior*

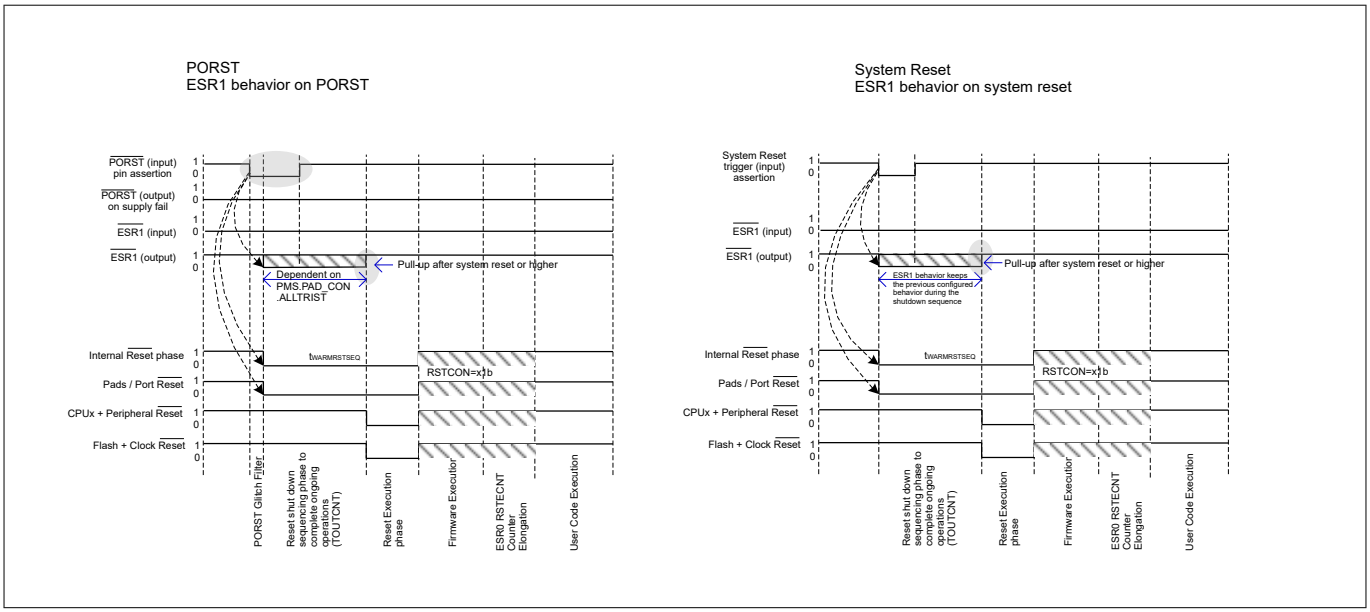


Figure 823 ESR1 behavior for warm power-on reset and system reset

ESR2

The ESR2 can be configured as reset output indication.

If configured as warm PORST output indication via PMS.PAD_ESR2CON.ROEN there are two cases to be considered

- cold power-on reset execution:

  In case of a cold power on reset the ESR2 will be asserted at the same time as the PORST Pad. The de-assertion is also on the same timing as the PORST Pad but can be elongated through PAD_ESR2CON.RSTECNT and PAD_ESR2CON.CLKDIV

  ***Note**: For cases of cold resets (except LVD) the PORST pad will be driven low for at least 80us by PMS*

- warm power-on reset execution:

  In case of a warm PORST triggered by assertion of the PORST Pad the ESR2 will be pulled low as long as the PORST Pad is asserted. The timing can be elongated though PAD_ESR2CON.RSTECNT and PAD_ESR2CON.CLKDIV

If configured as application reset output indication via PMS.PAD_ESR2CON.ROAPP=1 the ESR2 will be pulled low for the duration of the application reset phase. The shutdown sequence prior reset assertion does not have any effect on the ESR2 in that case.

***Note**: The duration of the application reset phase can be elongated by ESR0CNTCTRL configuration. Please refer to "ESRx pin reset output" section for details*

The scenario for warm power-on reset and application reset with respective configuration can be seen in the figure below.

***Note**: In case of activating the output indication for application reset and power-on reset simultaneously a superposition of the waveforms below can be observed.*

37 System Mode Management (SMM)

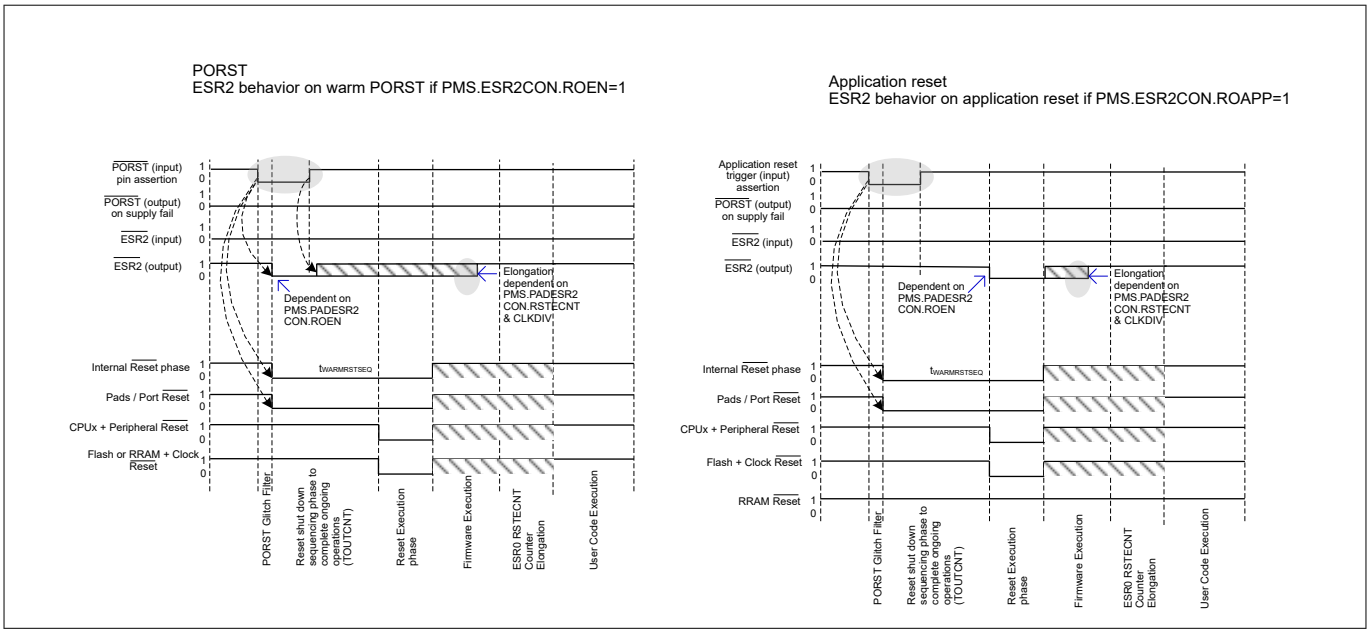


Figure 824 ESR2 behavior for warm PORST and application reset if configured as reset output

37.3.4 Reset

The AURIX™ TC4xx supports flexible reset concept to support open drain star reset architectures or cascaded reset architectures as shown in the figure. PORST and ESRx pins can be connected and configured flexibly to Power Management ICs, drivers, memory and communication devices depending on application needs. PORST supports bi-directional reset input and reset output function. ESR0/2 can be used as reset input or output. Error path can be configured in flexible manner to trigger immediate device reset or forwarded to PMIC. Hierarchical complete resets of the chip and partial selective resets of modules are supported in the reset architecture.

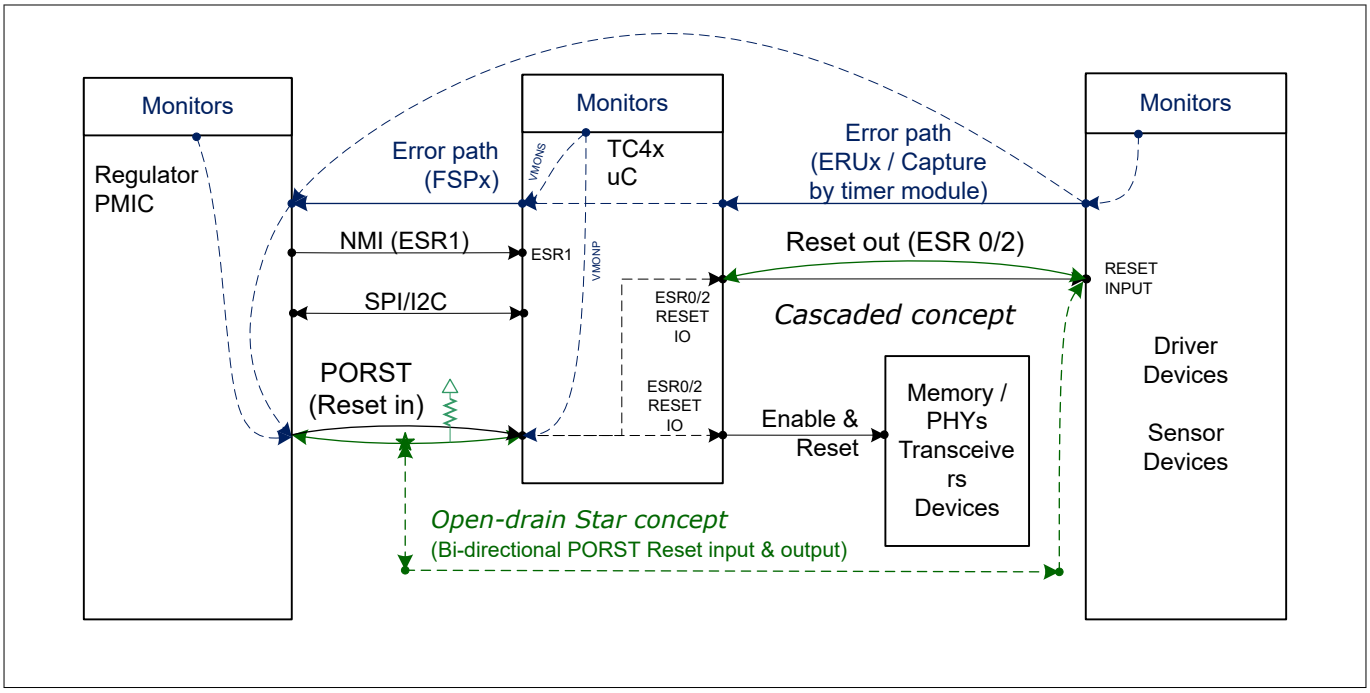


Figure 825 Reset connectivity

37.3.4.1 Overview

This section describes the conditions under which the AURIX™ TC4xx will be reset and the various reset types, triggers and effects.

37.3.4.1.1 Reset Types

The following list describes the different reset types and associated triggers.

Cold resets

- Low Voltage Detector reset (LVDPMS0, LVDPMS1 and LVDPMS2):This reset is triggered in case of power fail of standby supply domains and is the reset state after initial supply ramp-up. All modules are in reset state and all RAM memory content including Standby RAM memory need to be initialized. Refer to PMS chapter for more information
- Power Domain Cold Reset: This reset is triggered in case of power fail of the core supply rail beyond the lowest limits required by the power domain infrastructure. Refer PMS chapter for more information
- Cold Power-on reset (cold PORST): This reset is triggered in case of power fail of critical IO and core supply rails. All modules are in reset state except PMS0/1 and Standby RAMs. The PMS2 domain is partially reset. The individual PMS primary supply monitors are combined by a logical OR to trigger cold PORST which activates the strong pull-down of the PORST pad until supply under-voltage is active

Warm resets

- LBIST Reset: This reset portrays the device state when LBIST is running and after LBIST was successfully completed or aborted by an external power-on reset pin (PORST) assertion. The pins are held in reset state as configured by HWCFG6 pin
- Warm Power-on Reset (warm PORST): This reset sets the complete system into a defined reset state. This reset is triggered in case that the external power-on reset pin (PORST) is asserted low. This reset is triggered for example by external regulator when microcontroller fails to service external watchdogs or safety interfaces within a defined time period. All modules are in reset state except PMS, SMM, RRAM[49] and Standby RAMs. All RAMs are also required to maintain data content during and after the reset. A Power-on Reset also generates a Debug and Trace Reset and a System Reset and therefore also an Application Reset
- System Reset: This reset sets all modules in reset state except PMSx, SMM, RRAM[49] and Standby RAMs. A System Reset also generates an Application Reset. The system reset is triggered by software, safety alarms and watchdogs through the SMU unit, ESRx reset pins, system timers, TCU and debug modules
- Application Reset: This reset leads to an initialization into a defined state of the complete application system including peripherals and the processing units excluding the Clock system. All modules are reset except PMS, SMM, Clock, Flash[49], RRAM[49] and RAM contents are kept intact except for Cache/Tag memories. This reset is triggered by software, safety alarms and watchdogs through the SMU unit, ESRx reset pins, system timers, test and debug modules. This reset ensures a fast overall microcontroller reset with minimal reset time
  - Debug Reset: The debug system is also reset together with the application reset if OCDS is disabled. There is no explicit Debug Reset request possible but together with application reset
  - Trace Reset: This trace system is also reset with the application reset if trace is disabled
- Module Resets: Module resets result in individual modules being initialized into a defined state without any impact on the rest of the system. Module reset is not available for CPU0 and CSRM, Flash[49], RRAM[49] , PMS, IR, SCU, SMM, SMU and CLOCK. This reset is used to re-initialize the modules. The BPI part of the module is not reset in case of a module reset and only the module kernel part
- Module Group Resets: This reset leads to an initialization into a defined state for a configurable group of modules without any impact on the rest of the system

Warm PORST, system and application resets are triggered after the system is operational and the supplies are above the minimum operating conditions. On such a warm reset request, reset of pins will take place immediately, but the internal modules will only be reset after the system has been brought into a state where memory contents and debug trace data will not be corrupted and the load jumps have been sequentially ramped down. With regards to processing units, data integrity is only guaranteed for non-aligned write accesses from CPUs, PPU[49] and DMA. In the very unlikely event of a system hardware failure resulting in a timeout and reset but the memory transfer was not completed before the timeout, midway between two non-aligned bus transfers of a single bus transaction, possible corruption of memory may be unavoidable.

Cold Power-on Reset is triggered for the first time during a system power-up or in response to a temporary power failure of any of the main supply rails. During the power-up the PMS primary under-voltage monitors will trigger a complete reset of the system, placing the system into a known state. The PORST pin is de-asserted only when supplies are above the minimum operating conditions. The pins and modules are placed immediately into their reset state when the trigger is asserted. If the PORST pin is kept asserted after a system power-up or power fail event then the warm PORST will be extended after cold PORST release until the pin is de-asserted.

[49] Not available on all devices of the AURIX™ TC4xx family

37.3.4.1.2 Reset Trigger Sources

The reset trigger sources and the reset types which are generated are shown in the following table.

Table 1643 Reset Trigger Sources and generated Reset Type

Reset Request Trigger	Low Voltage Detector Resets	Cold Power on Reset	LBIST Reset	Warm Power on Reset	System Reset	Application Reset	Module Group Reset
VDDPMSx supply undervoltage below LVD level	Activated PMS LVD trigger	Activated by higher Reset	Not Activated	Activated by higher Reset	Activated by higher Reset	Activated by higher Reset	Not Activated
VDDEVRSB supply under voltage	Activated PMS LVD trigger	Activated PMS VMONP trigger	Not Activated	Activated by higher Reset	Activated by higher Reset	Activated by higher Reset	Not Activated
VDDEXT , VDDEXTDC supply under voltage	Activated PMS LVD trigger	Activated PMS VMONP trigger	Not Activated	Activated by higher Reset	Activated by higher Reset	Activated by higher Reset	Not Activated
VDDM, VDD supply under voltage	Not Activated	Activated PMS VMONP trigger	Not Activated	Activated by higher Reset	Activated by higher Reset	Activated by higher Reset	Not Activated
VDDHSIF, VDDPHPHYx, VDDPHYx supply under voltage	Not Activated	Configurable if HSPHYx or xSPI active PMS VMONP trigger	Not Activated	Configurable	Configurable	Configurable	Not Activated
VDDCPUx, VDDPPU power domains supply under voltage if present	Not Activated	Configurable if power domain active PMS VMONP trigger	Not Activated	Configurable	Configurable	Configurable	Not Activated
LBIST completion TCU trigger	Not Activated	Activated in SMM No PMS effect	Activated	Activated by higher Reset	Activated by higher Reset	Activated by higher Reset	Not Activated
LBIST abort by $\overline{\text{PORST}}$ pin	Not Activated	Activated in SMM No PMS effect	Activated	Activated by higher Reset	Activated by higher Reset	Activated by higher Reset	Not Activated
$\overline{\text{PORST}}$ pin assertion	Not Activated	Not Activated	Not Activated	Activated	Activated by higher Reset	Activated by higher Reset	Not Activated

(table continues...)

Table 1643 (continued) Reset Trigger Sources and generated Reset Type

Reset Request Trigger	Low Voltage Detector Resets	Cold Power on Reset	LBIST Reset	Warm Power on Reset	System Reset	Application Reset	Module Group Reset
$\overline{ESR0}$ pin assertion	Not Activated	Not Activated	Not Activated	Not Activated	Configurable	Configurable	Configurable
$\overline{ESR1}$ pin assertion	Not Activated	Not Activated	Not Activated	Not Activated	Configurable	Configurable	Configurable
$\overline{ESR2}$ pin assertion	Not Activated	Not Activated	Not Activated	Not Activated	Configurable	Configurable	Configurable
SMU0 Safe0 reset trigger	Not Activated	Not Activated	Not Activated	Not Activated	Configurable	Configurable	Configurable
SMU0 Safe1 reset trigger	Not Activated	Not Activated	Not Activated	Not Activated	Configurable	Configurable	Configurable
SMU1 Safe0 reset trigger	Not Activated	Not Activated	Not Activated	Not Activated	Configurable	Configurable	Configurable
SMU1 Safe1 reset trigger	Not Activated	Not Activated	Not Activated	Not Activated	Configurable	Configurable	Configurable
SMU Secure reset trigger	Not Activated	Not Activated	Not Activated	Not Activated	Configurable	Configurable	Configurable
CPU[x] STM[x] trigger	Not Activated	Not Activated	Not Activated	Not Activated	Configurable	Configurable	Configurable
Software (SW) trigger via SMM register	Not Activated	Not Activated	Not Activated	Not Activated	Configurable	Configurable	Configurable
Debug Tool System Reset - Cerberus RSTCL0	Not Activated	Not Activated	Not Activated	Not Activated	Activated	Activated by higher Reset	Not Activated
Debug Tool Application Reset - Cerberus RSTCL3	Not Activated	Not Activated	Not Activated	Not Activated	Not Activated	Activated.	Not Activated
Module Software Reset[1)]	Not Activated	Not Activated	Not Activated	Not Activated	Not Activated	Not Activated	Respective Module Kernel is reset but not BPI.

1) via KRST_CTRLx.KRST register bits)

***Note**: The cold power-on resets and configuration is implemented in PMS module. Activated by higher reset refers to the hierarchical trigger of the lower reset type by the higher reset type.*

***Note**: The JTAG interface and reset domain is controlled by the $\overline{TRST}$ pin.*

37.3.4.1.3 Reset Behavior of modules

A description of how the various modules, functions and pin of the device are affected by each reset type.

***Note**: In the following table, an "X" means that this block has at least some register bits that are affected by the corresponding reset. Please refer respective module register tables to ascertain which parts are affected by the corresponding reset.*

***Note**: The module list in the table below is representing the whole AURIX™ TC4xx family. If respective module or module function is not available in a given product then the information is not relevant*

Table 1644 Effect of Reset on Device Functions

Module/Function	Low Voltage Detector Resets	Cold Power-on Reset	LBIST	Warm Power-on Reset	System Reset	Application Reset
PMS0, PMS1	X	Not affected	Not affected	Not affected	Not affected	Not affected
SCR	X	configurable	Not affected	configurable	Not affected	Not affected
PMS2, EVRC, Standby SMU	X	X	Not affected	Not affected	Not affected	Not affected
PORST pin	driven low	driven low	Not affected.	Not affected. Reset trigger	Not affected	Not affected
Backup Clock	X	Not affected	Not affected	Not affected	Not affected	Not affected
SMM, CCU reset related parts	X	X	Not affected	Not affected	Not affected	Not affected
XTAL1/2, Oscillator, PLLs	X	X	X	X	X	Not affected
core SMU, SCU, WDT, VTMON, CSCU	X	X	X	X	X	X
Program and Data Flash	X	X	X	X	X	Not affected
DMU, FSI, PFRWB, DFRWB, IR, Interconnect, DMA, LMU logic	X	X	X	X	X	X
PRRAM, DRRAM	X	X	Not affected	Not affected	Not affected	Not affected
PMUR, DMUR	X	X	X	X	X	X

(table continues...)

Table 1644 (continued) Effect of Reset on Device Functions

Module/Function	Low Voltage Detector Resets	Cold Power-on Reset	LBIST	Warm Power-on Reset	System Reset	Application Reset
Debug	X	X	X	X	Reset if OSTATE.OEN=0 (OCDS is disabled)	Reset if OSTATE.OEN=0 (OCDS is disabled)
Trace, SGBT HSPHY used for Trace	X	X	X	X	Reset if OSTATE.TRACE_EN=0 (Trace is disabled)	Reset if OSTATE.TRACE_EN=0 (Trace is disabled)
JTAG Interface-reset additionally if TRST asserted	X	X	X	X	Not affected	Not affected
DAP Interface	X	X	X	X	Not affected	Not affected
TCU /TRI	X	X	Not affected	X	TRI & LBIST registers affected	Not affected
Pins $\overline{\text{ESR0}}$	X	X	weak pull-down	driven low	driven low depending on configuration	driven low depending on configuration
Pins $\overline{\text{ESR1}}$, $\overline{\text{ESR2}}$	X	X	weak pull-down	X	driven low depending on configuration	driven low depending on configuration
SMU Error pin FSP0	X	X	weak pull-down	X	Not affected	Not affected
CSRM, CPUcs, RNG, PKC, CSS	X	X	X	X	X	X
CPUx and Lockstep, PPU	X	X	X	X	X	X
Remaining Peripherals - ADC, GTM, FlexRay, CAN, QSPI, ASCLIN, SENT, I2C, PSI5, PSI5-S, Ethernet, PCIe, SDMMC, FCE, xSPI, HSPHYs not used for Trace.	X	X	X	X	X	X
Ports, Pins (supplied by VEXT)	X	X	X	X	X	X

(table continues...)

Table 1644 (continued) Effect of Reset on Device Functions

Module/Function	Low Voltage Detector Resets	Cold Power-on Reset	LBIST	Warm Power-on Reset	System Reset	Application Reset
Standby Port Pins (supplied by VEVRSB)	X	Reset depending on SCR config.	Reset depending on SCR config.	Reset depending on SCR config.	Reset depending on SCR config.	Reset depending on SCR config.
On-chip Static RAMs[1]						
dLMU (Standby RAMs)	reliable if no LVDANA reset [2]	Not affected by reset, reliable				
PSPRs, DSPRs, PPU CSM dLMU (non Standby) LMU RAM CSS RAM (keys & IVs)	Uninitialized. RAM content lost if VDD supply fails.			Not affected, reliable. Boot Firmware uses initial regions of PSPR (2kB RAM) and DSPR (8kB RAM) of CPU0 and CPUcs for the execution of start-up code and these parts are affected. Please refer Boot Firmware section for more details.		
PTAG, DTAG, CPU Cache, PPU Cache	Uninitialized. RAM content lost if VDD supply fails.			Cache invalidated. Initialization required after reset.		
Other RAMs- PKC RAM, CSS RAM (excluding keys & IVs), GTM RAM, CAN RAM, FlexRay RAM, PSI5 RAM, PSI5-S RAM, Ethernet RAM, HSPHY RAM	Uninitialized. RAM content lost if VDD supply fails.			Not reliable. After reset, the peripheral RAMs are filled/initialized with new data.		

1) Reliable here means that also the redundancy is not affected by the reset.
2) For Details about LVDANA please refer to the PMS chapter

The pins are reset depending on the PINRST configuration at the beginning when reset is triggered by default or after shut down sequencing. Likewise the HSPHY interface[50] is reset depending on the PHYRST configuration at the beginning when reset is triggered by default or after shut down sequencing.

50 Not available on all devices of the AURIX™ TC4xx family

37.3.4.1.4 Reset indication and flags

After a chip level reset release, the Reset Status registers provide information on the trigger(s) and the type(s) of the last reset(s).

All the reset types which happened since the last reset clear event through RSTSTATCLR.CLRC bit are cumulatively indicated in reset status register RSTSTAT. All warm reset triggers which happened since the last reset clear event through RSTSTATCLR.CLRC bit are accumulated and indicated in register WRMRSTTRIG. Likewise, all the cold reset triggers which happened since the last cold reset flag clear event through PMS.CLDRSTSTATCLR.CLEAR are indicated in PMS.CLDRSTRIG register. Detailed cold reset types and standby states are indicated in PMS.CLDRSTSTAT register bit field.

The highest reset type which happened since the last reset flag clear event is indicated in RSTSTATH register bits. A reset cycle denotes at least one reset trigger followed by the shut down sequence culminating in final reset. The corresponding reset trigger to trigger the highest warm reset type is indicated after any reset in WRMRSTTRIGH register reset bits. The RSTSTATH and WRMRSTTRIGH register can be cleared through CLRC bit by user software.

The RSTSTATCLR.CLRC bit allows clearing RSTSTATH, RSTSTAT, WRMRSTTRIG and WRMRSTTRIGH registers after the user software has completed reset evaluation. The PMS.CLDRSTSTATCLR.CLEAR clears PMS.CLDRSTSTAT and PMS.CLDRSTTRIG registers.

In case of a reset trigger from SMU as indicated in WRMRSTRIG and WRMRSTRIGH register flags, further information as to which alarm in SMU triggered the reset is indicated in SMURSTSTATA and SMURSTSTATB register bit fields. SMURSTSTATA indicates the corresponding alarms which triggered a reset for SMU SAFE0 unit. SMURSTSTATB indicates the corresponding alarms which triggered a reset for SMU SAFE1 unit. SMURSTSTATB indicates the corresponding alarms which triggered a reset for SMU SEC unit.

In case of consecutive watchdog timeout and permanent reset condition due to consecutive SMU resets which is recovered by external PORST pin assertion, DWR bit indicates that a a double watchdog permanent reset happened.

After a reset event the status bits in SHSEQCTRLD register can be used to determine whether the content of memory is still valid or whether corruption of data may have occurred.

Module level resets (Module Reset & Module Reset Group Reset) have no effect on the Reset Status registers and BPI part of the module.

***Note**: Even though application reset is not affecting the CLOCKING it can happen that after application reset the system PLL is not selected as clock source for the system clocks in case of a nested application reset scenario. Therefore the startup software shall check after application reset release that the CCUCON.CLKSELS bit field in the CLOCKING FB is selecting the expected system clock source*

37.3.4.2 Reset

The Reset Controller captures reset trigger requests and generates and distributes the different resets to all the modules .

A reset is generated if an enabled reset request trigger is asserted. Most reset triggers can be configured to initiate different types of reset depending on configuration.

37.3.4.2.1 Reset generation

The figure below shows the reset generation for the complete device

The Reset Generation detects the different reset requests, schedules a shutdown of the system, and then generates the appropriate internal reset pulse(s).

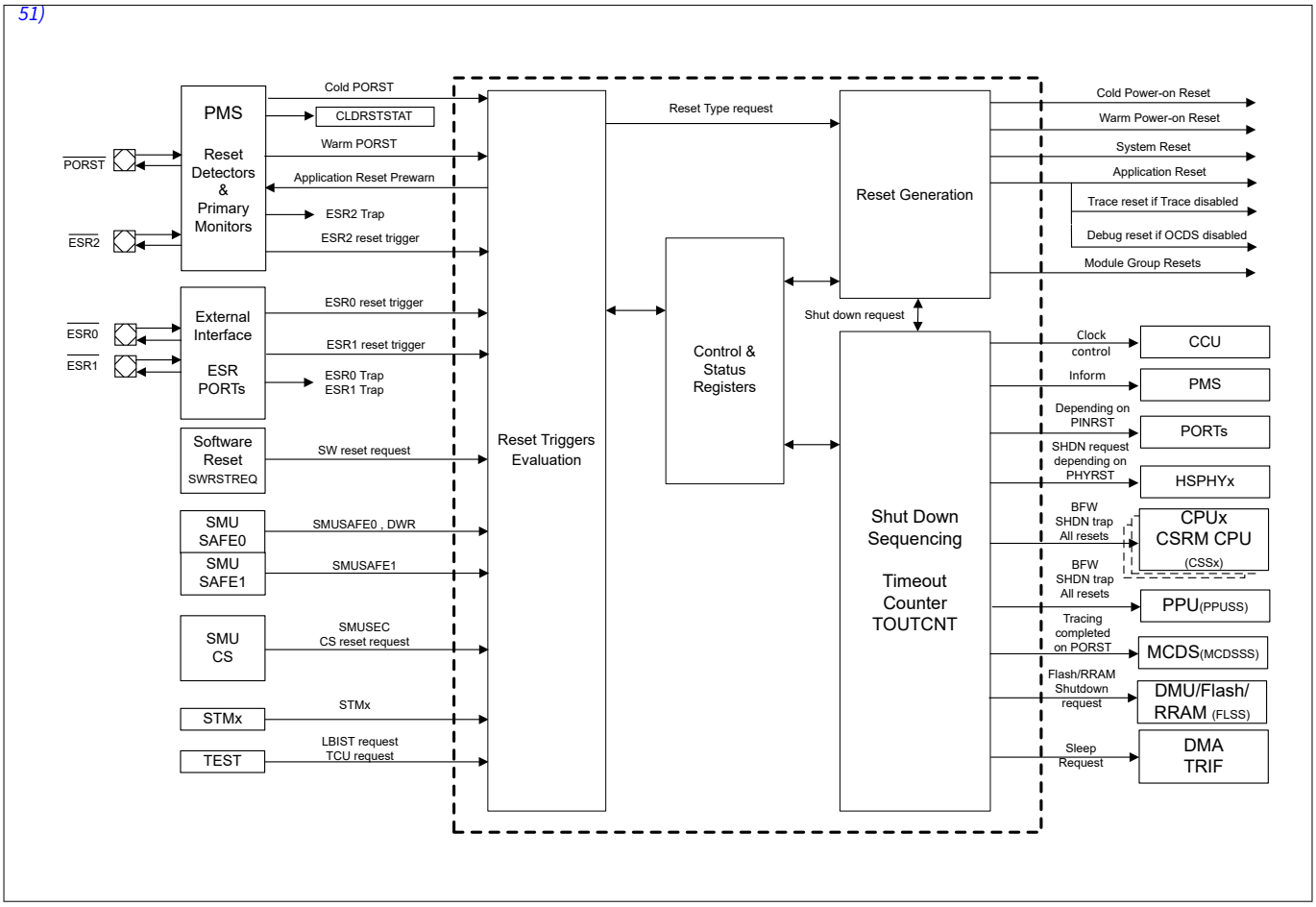


Figure 826 Reset overview

In case of safety-related faults during reset execution the SMM_RESET_ALARM or SMM_REG_ALARM is send to the SMU.

For details about the safety mechanisms please refer to the safety manual.

51 Flash, RRAM, PPU and HSPHYx are not available on all devices of the AURIX™ TC4xx family

37.3.4.2.2 Reset trigger request, assertion and de-assertion

Warm reset sequence

1. A Reset trigger requests a particular Reset Type, namely application reset, system reset or warm PORST
2. Shut-down sequencing is started for a time duration documented by datasheet parameter *t*WARMRSTSEQ. During this time, the modules are sequentially ramped down. An internal reset TOUTCNT counter counts down until the time duration has elapsed as indicated in SHSEQCTRLA register

   ***Note**: The DMA and RDMA[52] are excluded from the shutdown sequencing if their CLC.EDIS bit is disabled*

3. Pins or HSPHY modules[52] are set immediately into reset either when the reset request is taken or at a later time.
4. During the ramp-down phase, it might happen that further reset trigger requests are issued when counter is running, namely application reset, system reset or warm PORST. The additional reset requests are evaluated and highest reset type is executed among the reset requests. The shut-down sequencing is not restarted because of new reset requests as the ramp down phase of modules is already ongoing
5. After the shut-down sequence is completed or the reset counter has expired, the highest chosen reset type and the lower hierarchical resets are issued to all modules. The propagation of the reset to all modules takes less than 20 SPB clock cycles. Then the internal resets are de-asserted and boot firmware starts. The prerequisite for reset release is that no reset request trigger that is configured to generate the same reset is currently asserted and the reset counter has expired

   ***Note**: Elongated reset release only possible for PORST and ESR0*

6. If a reset trigger continues to be asserted externally even after reset issue (PORST or ESR0 reset pin input elongation, DWR), the internal reset type requested by the respective reset trigger is kept asserted until the reset trigger is externally de-asserted
7. ESR0 reset output and/or reset release are elongated for a configured time duration depending on configuration in ESRxCNTCTRL register

Cold reset sequence

1. A supply fail trigger requests a particular cold Reset Type namely, cold PORST, Power domain cold reset or LVD reset
2. Pins or HSPHY[52] modules are set immediately into reset and isolated
3. The cold PORST is immediately issued to all modules. The prerequisite for reset release is that the supply fail violation does not exist anymore
4. If the supply under voltage violation continues to exist, then the cold reset type is kept asserted until the supply has recovered and is in operational range
5. PORST and ESR0 pins are driven low during the cold PORST phase

37.3.4.2.3 Reset triggers and configurability

There are two types of reset triggers for the reset control logic:

- Triggers that lead to a specific reset
- Triggers that lead to a configurable reset

Specific reset triggers

Assertion of Debug tool (Cerberus) reset, TCU reset triggers and Supply fail leads to predefined resets and the triggers can not be enabled/disabled. All specific reset triggers are listed in the table below.

52 Not available on all devices of the AURIX™ TC4xx family

Table 1645 Specific reset triggers

Reset trigger	Reset type request
Cerberus 0 (CB0)	System reset
Cerberus 3 (CB3)	Application reset
VDDx	Cold power-on reset

Configurable reset triggers

Assertion of these triggers leads to a configurable reset type. These triggers can be enabled/disabled. The result of the reset triggers is defined by the corresponding bit-field in registers RSTTRIGCTRLA and RSTTRIGCTRLB.

SMU reset triggers

The SMU can request different resets in response to an alarm. The two Safety SMU partitions can request 2 different reset types each and the Security SMU partition has one available reset request. For detailed information about the possible configurations see Register description RSTTRIGCTRLA.

To indicate which Alarm was responsible for the triggered Reset the SMURSTSTATx provides information about Alarm Group and Alarm ID.

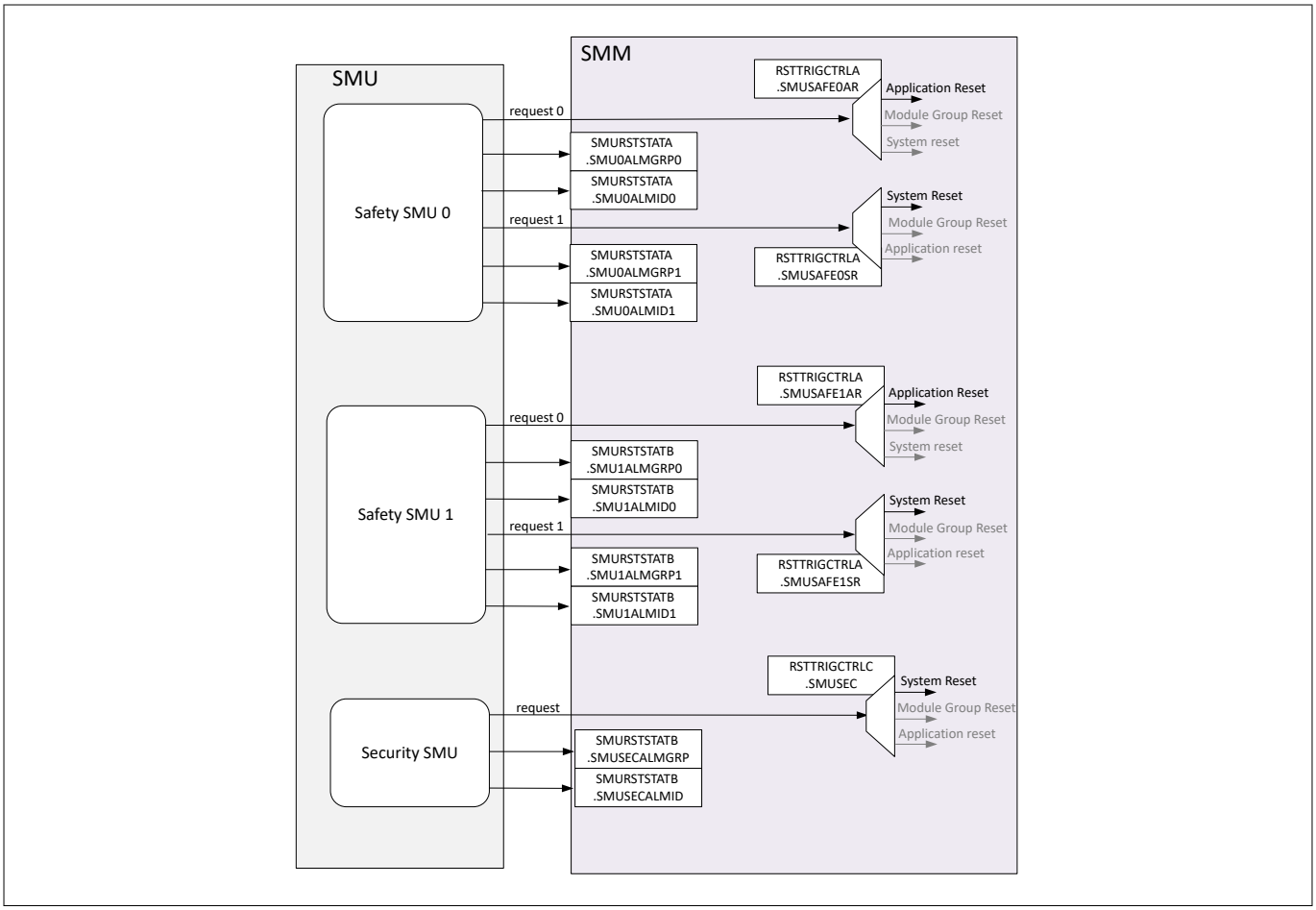


Figure 827 SMU reset request and status indication

Prevention of double SMU resets leading to permanent reset condition

The SMU can request a reset in response to an alarm. The type (Application or System Reset or Module Group Reset) is selected through register RSTTRIGCTRLA.SMUSAFEx or RSTTRIGCTRLC.SMUSEC. After boot firmware

initialization, the default behavior of the SMU WDT is that a Watchdog timeout alarm would result in an NMI followed by an application reset. If an SMU induced application reset occurs twice, a severe system malfunction is assumed and the AURIX™ TC4xx is held in permanent application reset until a Power-On Reset occurs. This prevents the device from being periodically reset if the application fails to service the watchdog correctly and the watchdog repeatedly times out.

An internal flag is set when the first SMU reset is requested. If a second reset is also requested by the SMU when the internal flag is already set, the double SMU reset event has occurred and a permanent reset request is automatically generated. This internal flag is only reset by a System Reset or when bit RSTSTATCLR.CLRSMUIRF is set, which is SMM_PROT_E.ENDINIT protected. A correct service of the WDT does not automatically clear this flag.

If this behavior is undesirable, then the RSTSTATCLR.CLRSMUIRF bit should be written by the application during initialization (before the watchdog times out), to clear the flag and prevent any subsequent occurrence of a permanent reset. The DWR reset flag is set in case of permanent reset event and will be indicated in WRMRSTTRIGx register after the system recovers from permanent Application Reset state after an external PORST assertion event.

Debug and trace resets

Debug reset behavior

It is required by the debugger that if the OCDS system is disabled (OSTATE.OEN=0), a Debug Reset is also asserted every time an Application Reset is asserted.

Trace reset behavior

It is required by the debugger that if the Tracing is disabled (OSTATE.TRACE_EN=0), a Trace Reset is triggered on an Application Reset.

JTAG interface reset behavior

During RUN mode, JTAG interface is reset with TRST pin assertion.

37.3.4.3 Module kernel reset

Many modules can be reset individually. The module or kernel reset has no effect outside the module itself. A module reset can only be triggered by writing '1' into both of the module reset registers RST_CTRLA.KRST and register RST_CTRLB.KRST. The module reset allows fast initialization of module registers to their reset values. It shall be noted that only the module kernel is reset on a module reset and the BPI interface is not reset. After the module reset is executed, the respective module RST_STAT indicates whether the module reset is done.

The table below shows the modules which support Module reset capability. Infrastructure modules are exempt from module resets as resetting them affects more than one module.

***Note**: The module list in the tables below is representing the whole AURIX™ TC4xx family. If respective module or module function is not available in a given product then the information is not relevant*

Table 1646 Functional block supporting module resets

Module	Module reset capability
CANx	Module reset implemented in CAN
DRE	Moduel reset implemented in DRE
ASCLINx	Module reset implemented in ASCLIN
GTM	Module reset implemented in GTM
ERAYx	Module reset implemented in ERAY
QSPIx	Module reset implemented in QSPI
HSSL	Module reset implemented in HSSL
GETH	Module reset implemented in GETH ***Note**: includes MII, RGMII and SGMII*
HSPHY	Module reset implemented in HSPHY
MSCx	Module reset implemented in MSC
SENTx	Module reset implemented in SENT
TMADC, DSADC, CDSPX	Module reset implemented independently for TMADC, DSADC and CDSPx in ADC
SCR	Module reset implemented in SCR ***Note**: SCU_RSTCON.SWRSTREQ to be used*
GPT12	Module reset implemented in GPT12
PSI5	Module reset implemented in PSI5
PSI5-S	Module reset implemented in PSI5-S
DMAx	Per channel reset implemented in DMA ***Note**: TSRc.RST to be used*
I2Cx	Module reset implemented in I2C

(table continues...)

Table 1646 (continued) Functional block supporting module resets

Module	Module reset capability
CPU1 and all higher CPUs	Individual CPU reset capability ***Note**: includes STMx*
SPU	Module reset implemented in SPU
RIF, CSI2	Module reset implemented in RIF, CSI2 if present
eGTM	Module reset implemented in eGTM ***Note**: includes reset of HRPWM*
SDMMC	Module reset implemented in SDMMC
PPU	Module reset implemented in PPU
PCIe	Module reset implemented in PCIe ***Note**: includes reset of the connected PHY in HSPHY*
LETH	Module reset implemented in Leth
xSPI	Module reset implemented in xSPI
VTMON	Module reset implemented in VTMON ***Note**: Module reset or Module Group Reset for VTMON shall only be triggered if there is no ongoing CLC clock disable or ongoing OCDS soft suspend*
FCE	Module reset implemented in FCE
PORTS	Module reset implemented in PORTS
RDMA	Module reset implemented in RDMA
CANXL	Module reset implemented in CANXL
AUDIO	Module reset implemented in AUDIO ***Note**: Module reset or Module Group Reset for AUDIO shall only be triggered if there is no clock of the AUDIO FB disabled with the CLC register*

Table 1647 Functional blocks not supporting module resets

Module	Module reset capability
PMS, SMM, SCU, CCU, CSCU, VMT, WTU	No module reset capability being infrastructure modules
SMU, Error pins	No module reset capability
NVM, PFLASH, DFLASH	No module reset capability as accessed by multiple masters

(table continues...)

Table 1647 (continued) Functional blocks not supporting module resets

Module	Module reset capability
NVMR, PRRAM, DRRAM	No module reset capability as accessed by multiple masters
LMU	No module reset capability as accessed by multiple masters
IR	No module reset capability as accessed by multiple modules
CPUcs	No module reset capability shall be used as CPUcs is the secure boot CPU
CSS, PKC,TRNG	No module reset capability
Bus System - SRI, FPI, LLI	No module reset capability
CPU0	No module reset capability shall be used as CPU0 is the boot CPU

Related information

CANXL Module kernel reset on page 6440

37.3.4.3.1 CPU module resets

A CPU Module reset has the same effect on an individual CPU as an Application Reset, but without a full reinitialization of other system elements outside the CPU:

- CPU0 and CPUcs - Module reset functionality shall not be used
- Other CPUs - Wait in HALT mode after module reset

The BPI register reset state can be used to determine whether the last CPUx reset had happened.

See CPU and Start-Up Chapters for more details.

***Note**: The individual CPU Module reset includes the respective STM*

37.3.4.3.2 PPU module resets

A PPU[53] Module reset has the same effect as an Application Reset, but without a full reinitialization of other system elements outside the PPU

PPU - Wait in HALT mode after Module Reset.

The BPI register reset state can be used to determine whether the last PPU reset had happened.

See PPU and Start-Up Chapters for more details.

53 Not available on all devices of the AURIX™ TC4xx family

37.3.4.4 Module Group Resets

Modules providing Module reset can be part of four possible Module reset groups. The configuration as to whether a module is part of a Reset Module Group is done in the respective module. When a respective trigger for one of these groups is broadcast by SMM, all modules in the group perform simultaneously a Module Reset.

***Note**: No Module Group Reset available for CPUs*

The triggers for Module Group Resets can be configured in RSTTRIGCTRLA and RSTTRIGCTRLB and may be triggered by the following warm reset triggers:

- ESR0, ESR1 and ESR2 pin triggers
- SMU SAFE0, SAFE1 & SMUSEC alarm triggers
- Software request for Module Group Reset
- STMx compare match triggers

Module Group Resets have no effect outside the configured module groups, since the behavior is similar to that of triggering individual module resets chained all together. Module Group Resets (through register bits in some modules) have no effect on infrastructure and BPI of that module. Modules which do not support module resets also do not support Module Group Reset.

The module participation in Module Group Reset is configured through respective module RST_CTRLA.GRSTENx (x=0-3) bit field. The module register RST_STAT.GRSTx (x=0-3) can be used to determine what kind of reset was ordered to be executed on the module. This bit-field can be cleared through RST_CTRLB.STATCLR.

If a module is assigned to multiple Module Group Resets, the respective flags would be cumulatively set to indicate that the module reset was requested from respective multiple Module Group Reset sources.

The SMM does not contain information about what was the trigger of the Module Group Reset in any status register.

***Note**: To avoid a cold power on reset on assertion of a Module Group Reset it is recommended to consider voltage droops when configuring a module group since all configured modules will reset at the same time without reset shut down sequencing.*

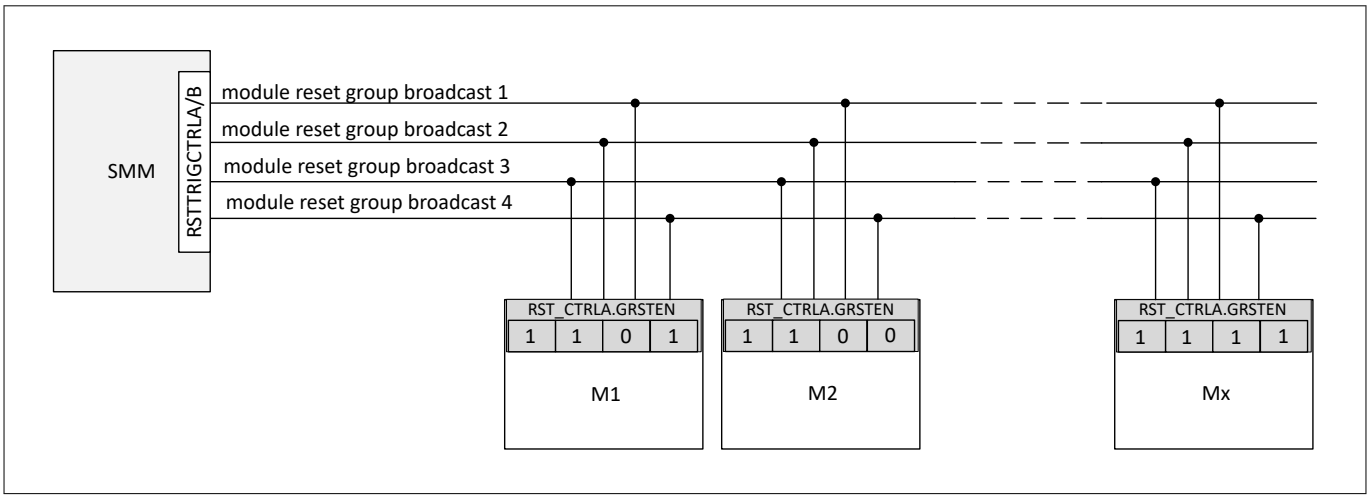


Figure 828 Module reset group overview

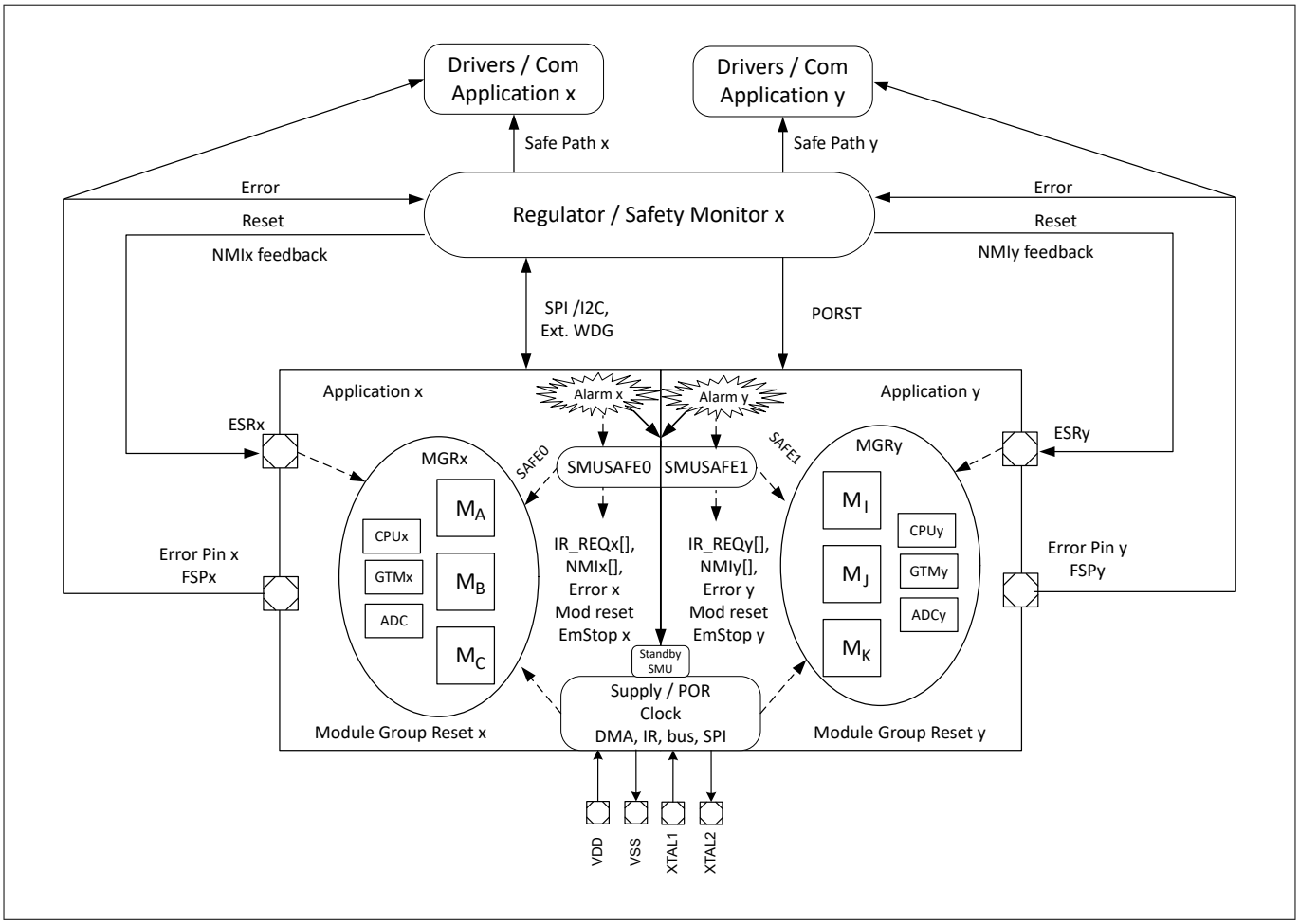


Figure 829 An example of application partitioning

Simultaneous Module Group Reset and NMI trigger by ESR pins

The figure illustrates an example case for MGRx assertion and the behavior of microcontroller modules. MGRx can be triggered by external or internal agent depending on configuration by the user program. In the described case, the ESRx pulse acts as an external MGRx request trigger. Same effects may be triggered by internal agents through MGR reset request to SMM if configured.

Flexibly mapped safety alarms in the SMU_SAFEx instances usually act as initial source of Module Group Reset to the SMM which is triggered directly through internal request from SMU_SAFEx or SMU_SAFEx output FSP[0-3] to external agents.

ESR[1-2], FSP[0-3], MGR[0-3] implement an isolated reset and restart of selected modules for flexible and fast recovery of only those parts of the application where fault is detected. If the fault does not interfere with the rest of application, the MGR can be triggered only on the affected application components, while other parts usually mapped to another group are able to maintain their dedicated application function. Application integrator decides which safety alarms trigger and what class of the reset request is to be configured. This configuration is usually applied by application Start-up initialization routine. Configurations are applied at module level in RST_CTRLA.GRSTEN registers to decide the connections to the respective Module Group Reset broadcasts line(-s).

ESR[1-2] input settings shall be initialized to define signal properties to generate the reaction to the SMM. Both configurations of edge and pulse detection are supported. It is possible to map pulse driven requests to SMM/ MGRx reset triggers. Pulses are captured by the ESR input capture timer and compared with ESRCFG1.RSTDELAY value through a greater than or less than compare filter to assert chosen request type.

At the same time, edge driven requests may still trigger NMI requests based on ESRCFG1.EDCON setting. For instance, the same ESR[1-2] input lines can be used for interrupt generation from the same external agent.

The figure below provides an example case where edge driven MGRx requests are configured on ESR2 input and pulse driven requests are configured on ESR1 input. ESR2 input accept pulses captured by ESRx timer for MGRx requests which are longer than pre-configured time window. Pulse driven requests do not hold modules in reset for the pulse duration time, but rather keep it for the same period as any pulse generated and broadcasted by SMM. Affected group of modules go into reset and restart once the respective module kernel reset is released. In this example FSP[1] trigger is mapped on ESR2 input and FSP[2] is mapped on ESR1 input on a system level. FSP[0-3] can be mapped to different reaction class such as Warm PORST or Cold Reset generation. This is typically done for severe faults that affects all application groups together, and thus require a higher class of recovery measure.

The blocks which are not affected by MGRx reset function include PORST, fBACK/fRAMP, fOSC, PLL, PU1/PU2 Pins and ESR0 function.

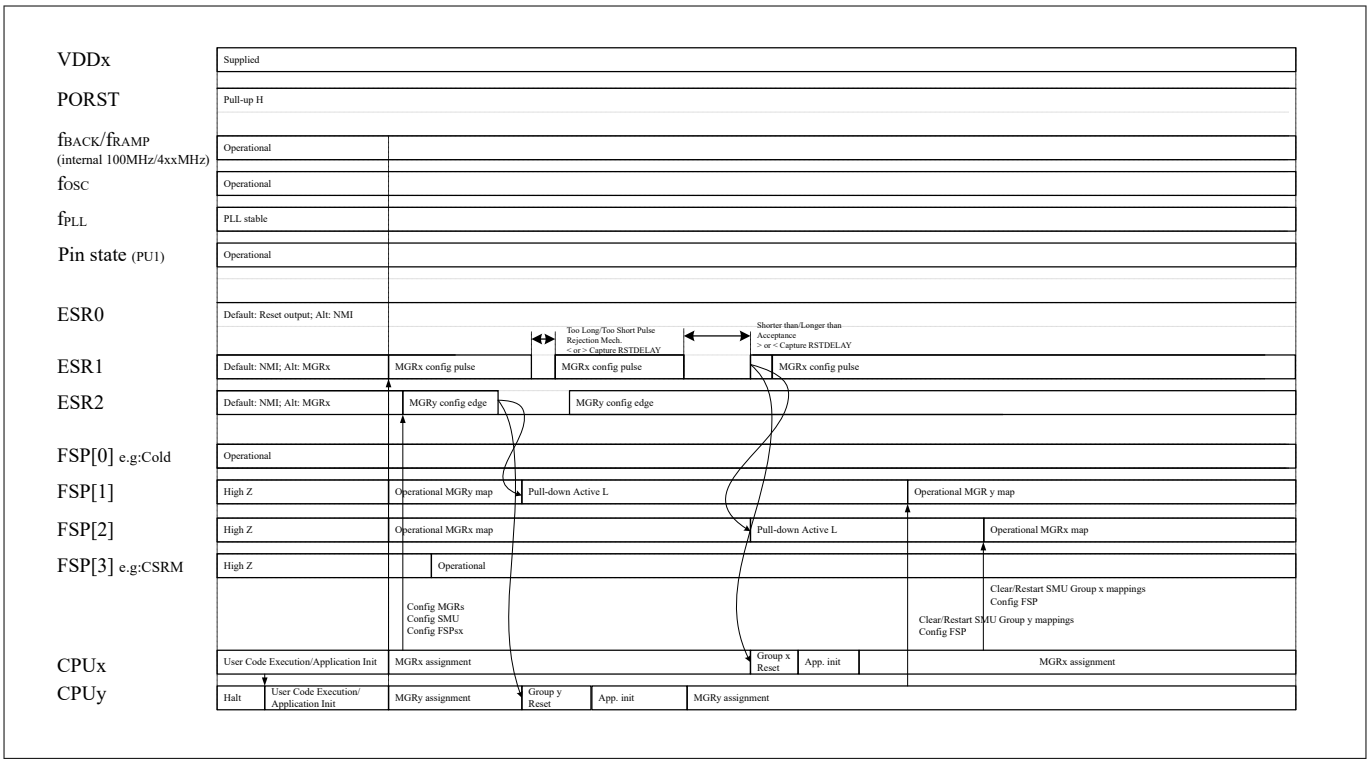


Figure 830 MGRx example using ESRx pins

37.3.5 Load jump management

Load jumps lead to voltage overshoots / undershoots which need to be limited within the regulator dynamic specification and operational bounds of the supply rail. The initial phase after the load jump is buffered immediately by external buffer and decoupling capacitors which consequently leads to a linear discharge of the capacitor. Consequently the regulator feedback loop recognizes the deviation in voltage and reacts to the jump by regulating the power supply unit. The dimensioning of the capacitor depends on the load jump amplitude, equivalent series resistance of the capacitor, the permissible voltage deviation and bandwidth of the regulator. The capacitor size in turn has a tangible impact on cost and PCB space. Minimum required capacitors and indicative load jump transients are documented in datasheet. Minimizing peak-to-peak voltage deviation in the face of large dynamic changes in load current need to be actively managed to avoid brown out reset events and large capacitors.

Load jumps may be triggered by software or user driven actions or asynchronous hardware events. During software triggered non reset events, it is recommended to limit the load jumps as documented in datasheet. For example, during clock ramp-up phase it is recommended to limit the clock switching steps so as not to violate this limit.

Asynchronous events and resulting load jump - resets and NMI

On typical load jump events triggered asynchronously, like Non Maskable Interrupts and reset events, measures are built in to ensure that voltage overshoots are kept within bounds by in-built load sequencing mechanisms.

In case of an NMI event, during RUN mode or waking up from SLEEP mode, the device would activate a large number of hitherto dormant circuits and wake-up the CPUs simultaneously resulting in a large change in load current. To avoid a large load jump on an NMI event, it needs to be ensured that only one CPU is triggered by the NMI or woken out of SLEEP mode and that other CPUs are still in IDLE mode. The other CPUs are consequently started one after another with adequate delay in between during start-up phase. The active CPU woken up on an NMI request is configurable in SCU.

In case of an warm Power-On Reset request, the port pins are set into reset state after t_{PIP} as specidifed in the datasheet. In case of a System or Application Reset request the timing of the reset state assertion for the port pins is configurable in RSTTRIGCTRLA.PINRST. After that the modules are ramped down in a grouped sequence over TOUTCNT period documented in datasheet. This serves to limit the maximum load jump to less than half of the total dynamic IDD current. It needs to be ensured that the supply voltages are above the minimum operational voltage limits during the reset phase after warm reset request not to trigger an additional cold power-fail reset. Larger overshoots are tolerable during and after reset phase for a certain accumulated time but must be limited to operational and absolute maximum voltage ratings as documented in the datasheet.

Load jump events caused by asynchronous failure events like PLL loss of lock or external oscillator watchdog event may lead to overshoots which cannot be sequenced owing to the inherent nature of failure.

37.3.5.1 Load jump sequencing unit A

Handling simultaneous load jump requests triggered by software

Software triggered load jump events include ramping up / down of various system clock frequencies, activating / deactivating processing units, power mode transitions, CPU idle requests, memory tests and so forth.

In case of software triggered events, it is possible to prepare by lowering or raising the voltage set point before the load jump is issued. A negative voltage droop may be done before a negative load jump and a positive voltage droop before a positive load jump respectively. Thus negative load jumps leading to voltage overshoots is compensated partly by the negative voltage droop and likewise positive load jumps leading to voltage undershoots is compensated partly by the positive voltage droop as shown in figure. The voltage droop is configured through PMTRCSR0.SDSTEP register bits. The voltage droop in positive or negative direction is issued via PMTRCSR3.VDROOPREQ register bits. In case a current Vdroop request is not active or the Voltage Droop Timer is not currently running indicated via PMTRCSR3.VDTRUN or the Load Jump Timer is not currently running indicated via PMTRCSR2.LJTRUN, a new Vdroop request is taken. Once a new voltage droop request is

issued, SDVOK is reset and device need to wait for a certain time till the regulator has settled on the new value which is realized using a Voltage Droop Timer. Once the regulator has settled on the new value, SDVOK status bit is set again indicating the end of the Voltage Droop transition and PMTRCSR3.VDTRUN and PMTRCSR3.VDTCNT is reset by hardware. If SDVOK status bit is set by EVRC before compare match of VDT has occurred, VDTOV overflow bit is not set and overrun interrupt is not generated. The Voltage Droop Timer compare value is configured in PMTRCSR1.VDTCV register bits and the current value is indicated in PMTRCSR3.VDTCNT register bits. In case of a compare match, the overflow PMTRCSR3.VDTOV bit is set if enabled via PMTRCSR0.VDTOVEN register bits. In this case, overflow bit has to be explicitly cleared via PMTRCSR3.VDTOVCLR before a new request can be taken to support a sequential polling based approach. Furthermore, interrupt maybe activated on an overflow if PMTRCSR0.VDTOVIEN is enabled.

Simultaneous software triggered load jump events can be likewise avoided by checking whether a Load Jump is ongoing or the Load Jump Timer is currently running. The Load Jump Request is issued by triggering a compare and swap operation on PMTRCSR2 register. A CPU will access data from PMTRCSR2 register and will compare the current value with an expected value. The expected value is that there is no load jump currently ongoing and VDTRUN bit state is 0. If there is a match, the CPU will make the swap by setting PMTRCSR2.LDJMPREQ variable and starting the timer. The idea is to prevent multiple CPUs from doing load jumps simultaneously by treating Load Jump as a critical section and ensuring that only a single CPU can check and issue a load jump request atomically. Once a load jump request is taken, a timer is started and other CPUs have to wait till the regulator output has been restored to the setpoint value and ensures adequate regulator reaction time. The other CPUs are not blocked during the waiting period instead they can continue with some other operations or try to make the request again at a later point of time.

Thus negative load jumps and positive load jumps are followed by a blanking period using a Load Jump Timer as shown in figure below. The Load Jump Timer is configured through PMTRCSR0 register. The load jump request is issued via PMTRCSR2.LDJMPREQ register bits. If a current Load Jump request is not active or the Load Jump Timer is not currently running indicated via PMTRCSR2.LJTRUN or the Voltage Droop Timer is not currently running indicated via PMTRCSR3.VDTRUN, a new Load Jump request is taken. Once a new Load Jump request is issued, the device needs to wait for a certain time till the regulator has reacted to the jump which is realized using a Load Jump Timer. The Load Jump Timer compare value is configured in PMTRCSR1.LJTCV register bits and the current value is indicated in PMTRCSR2.LJTCNT register bits. In case of a compare match, the overflow PMTRCSR2.LJTOV bit is set if enabled via PMTRCSR0.LJTOVEN register bits. In this case, overflow bit has to be explicitly cleared via PMTRCSR2.LJTOVCLR before a new request can be taken to support a sequential polling based approach. Furthermore, an interrupt maybe activated on an overflow if PMTRCSR0.LJTOVIEN is enabled. Overflow bit is routed to an OS interrupt to schedule the next current jump. Overflow bit maybe masked or used in the compare and swap operation if PMTRCSR0.LJTOVEN is set to ensure that explicit clear of the time out has happened before issuing a new request. PMTRCSR2.LJTOVCLR also clears PMTRCSR3.VDROOPREQ and PMTRCSR2.LDJMPREQ request.

***Note**: The droop can only be reversed to nominal when load jump timer is enabled as well as the droop timer. Only then the droop is reversed when LJTOVCLR bit is set*

37 System Mode Management (SMM)

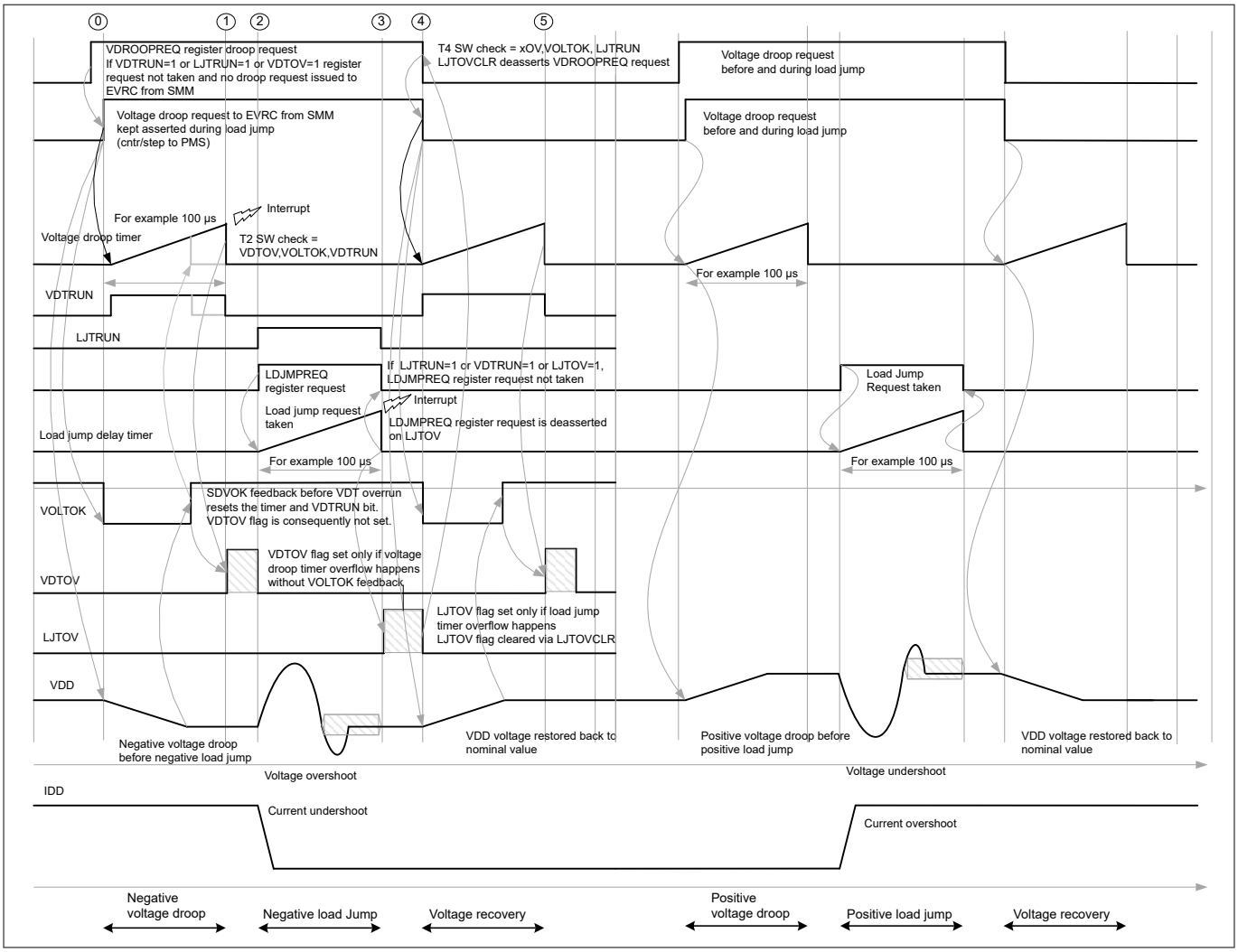


Figure 831 Load jump sequencing unit A

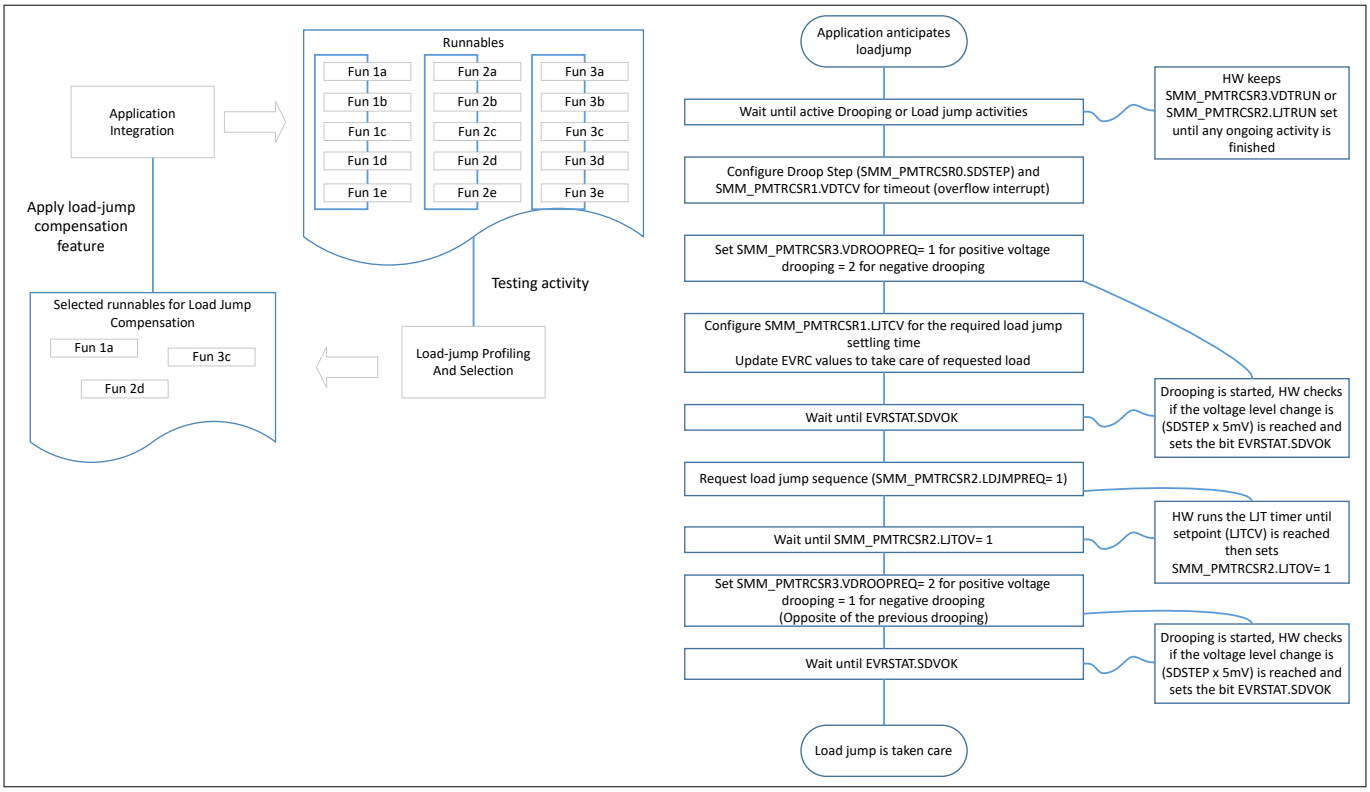


Figure 832 Load jump handling sequence

Interrupts

Following events may be configured to lead to interrupts routed to the interrupt router. If enabled by the related interrupt enable bits, an interrupt pulse is generated on one of the service request outputs (SRC_SMM0 and SRC_SMM1). Interrupts are forwarded from SMM to the IR module within 4 f_{SPB} clock cycles after the occurrence of the event.

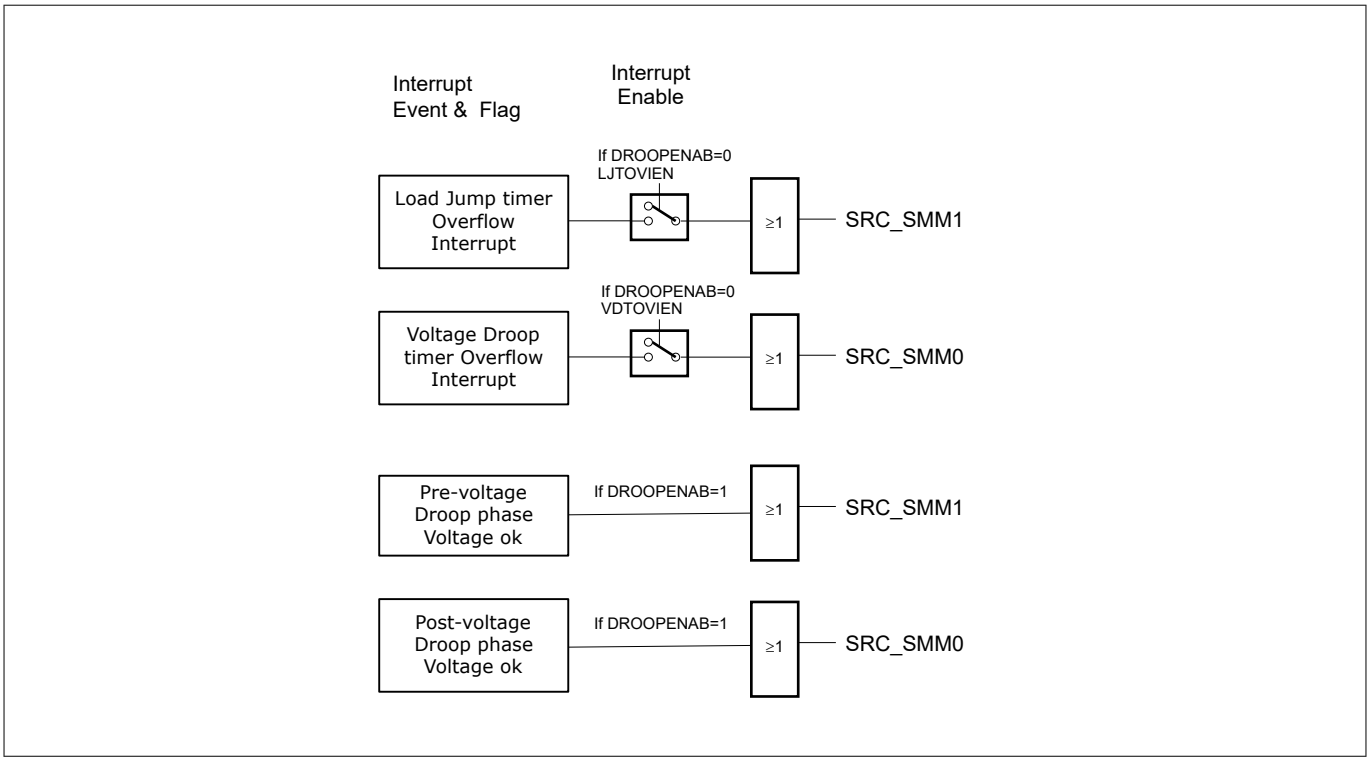


Figure 833 Interrupt sources and events

37.3.5.2 Load jump sequencing unit B

For a more hardware driven approach load jump sequencing unit B is provided.

The sequence of triggering voltage droop before load jump is as follows:

- T0: When droop request is triggered (REQ = 1), a voltage droop request is sent to PMS.EVRC module. The direction of voltage droop and amount of voltage step to be taken is also conveyed to EVRC. The BUSY flag is set immediately to indicate that no further droop requests can be taken until the ongoing activity is completed. V_NOM = 0 is set
- T1: The VDD voltage droop has started
- T2: After the new voltage set point is reached, PMS.EVRC confirms that voltage is ok. Then V_SP = 1 is set and an interrupt is triggered. DWELL counter is started to provide adequate time for software to trigger load jump. DWELL counter is running on the back-up clock
- T3: Tsw after T2 indicates the time when software triggers the load jump and set the LJT bit to confirm that load jump has happened
- T4: When dwell time has expired, the droop request is de-asserted to PMS.EVRC. V_SP = 0 is set on an acknowledge from PMS that droop request is de-asserted
- T5 = PMS.EVRC confirms that voltage is ok and V_NOM = 1 and BUSY = 0 is set and interrupt is triggered
- ERR is set if PMS.EVRC does not acknowledge to the droop request or software does not set LJT Load jump feedback bit before dwell time expiry

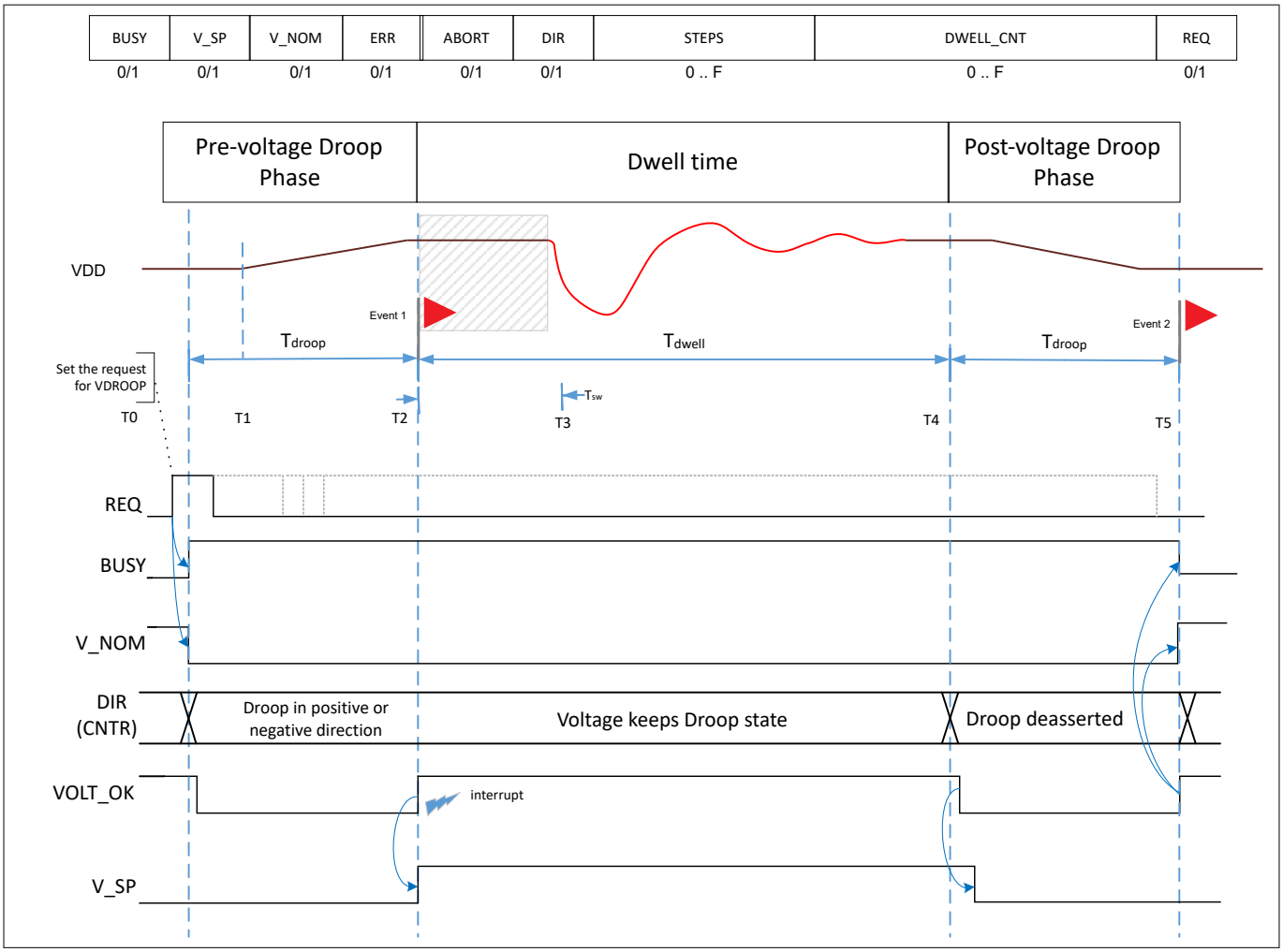


Figure 834 Load jump compensation with voltage droop

Software sequence and interface:

SMM module supports following software interface for voltage droop in register DRPCTRL:

REQ: Request Set to 1 to trigger new voltage droop compensation activity. It is a write only bit. The trigger is ignored if BUSY or ERR is set.

STEPS: Target number of steps to raise or lower voltage level. Control the target number of steps to bring VDD to either to higher or lower side of the nominal voltage level. Voltage is raised in 5 mV steps from nominal voltage level to target number of steps.

DIR: Direction of voltage change.

DWELL_CNT: Dwell count. This control field configures number of timer counts to keep the voltage level at compensation level.

LJT: Once load jump is triggered during Dwell time, the flag is set by user software to indicate that load jump was taken. If not, ERR will be set.

BUSY: Busy status of droop compensation activity. The bit is 1 during the time whole droop compensation activity is going-on.

V_SP: Voltage set point. Current voltage level is at voltage set point.

V_NOM: Voltage at nominal level. Current voltage level is at nominal supply voltage level.

ABORT: Abort control. If set, the voltage level at voltage compensation activity shall be aborted and supply voltage shall be brought back to the nominal voltage level (indicated by V_NOM).

ERR: Error indicator. Set if there is an error that has occurred during the voltage compensation activity.

ERRCLR: Error indicator clear. Clears ERR and LJT flag so that next REQ request can be taken.

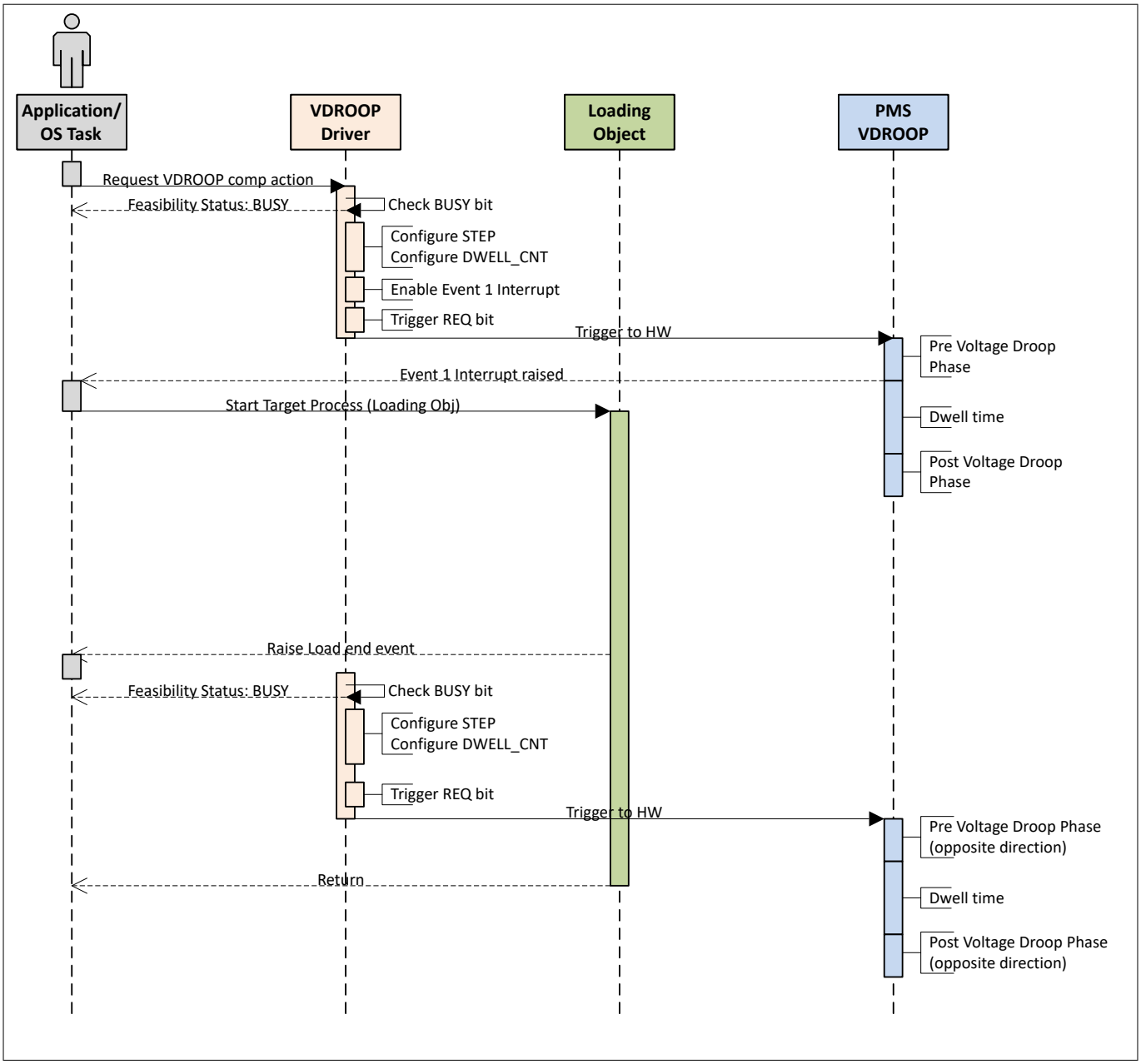


Figure 835 Load jump sequence unit B - software sequence

37.3.5.3 Shutdown sequencing on mode and reset transitions

Shut down sequencing ensures that modules are ramped down in a sequence before the internal reset(s) are issued so that RAM integrity is ensured on a warm reset and load currents are ramped down in a regulator friendly manner.

On detection of an Application Reset, System Reset or Power-On Reset trigger, the shut down sequence is started.

When the reset happens, status of all CPUs, PPU[54], Flash[54], RRAM[54] and other masters are latched in SHSEQCTRLD.

The CSSx bits indicate whether each CPU successfully flushed its write buffers and reached an idle state before the previous reset. The safe state status for PPU[54] is indicated respectively in the PPUSS. These bits can therefore be used to determine whether RAM content integrity can be trusted after the previous reset cycle.

54 not available on all devices of the AURIX™ TC4xx family

37.4 Registers

37.4.1 Register overview - access mode glossary

Table 1648 Register overview - access mode glossary

Keyword	Description
CE	Access protection using PROT register PROTCE.
CSE	Access protection using PROT register PROTCSE.
E	Access protection using PROT register PROTE.
ECPUn (n=0-5)	Access protection using PROT register PROTCPUn.
SE	Access protection using PROT register PROTSE.
APU-P	Protection group consisting of registers ACCEN_WRA, ACCEN_WRB, ACCEN_RDA, ACCEN_RDB, ACCEN_VM, ACCEN_PRS.
P	Access protection using *APU-P* registers.
APU-PCS	Protection group consisting of registers ACCENCS_WRA, ACCENCS_WRB, ACCENCS_RDA, ACCENCS_RDB, ACCENCS_VM, ACCENCS_PRS.
PCS	Access protection using *APU-PCS* registers.
BE	Always returns a Bus Error.
U	No access restrictions.
PROT	Access restrictions as defined in the PROT register access rules.
STP	Access only during start-up, before start-up protection is set (SMM_STCON.ST_PROT=1).

37.4.2 Registers overview - SMM (ascending offset address)

Table 1649 Registers overview - SMM (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ID	Module Identification Register	0008_H	P	BE	PowerOn Reset	6371
PROTE	PROT Register Endinit	$000C_H$	U	SV, PROT	Application Reset	6371
PROTSE	PROT Register Safe Endinit	0010_H	U	SV, PROT	Application Reset	6373
PROTCSE	PROT Register Cyber-Secure Endinit	0014_H	U	SV, PROT	Application Reset	6375
ACCEN_WRA	Write access enable register A	0018_H	U	SE, SV	Application Reset	6377
ACCEN_WRB	Write access enable register B	$001C_H$	U	SE, SV	Application Reset	6377
ACCEN_RDA	Read access enable register A	0020_H	U	SE, SV	Application Reset	6378

(table continues...)

Table 1649 (continued) Registers overview - SMM (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ACCEN_RDB	Read access enable register B	0024_H	U	SE, SV	Application Reset	6378
ACCEN_VM	VM access enable register	0028_H	U	SE, SV	Application Reset	6379
ACCEN_PRS	PRS access enable register	$002C_H$	U	SE, SV	Application Reset	6379
ACCENCS_WRA	ACCENCS write access enable register A	0038_H	U	CSE, SV	Application Reset	6380
ACCENCS_WRB	ACCENCS write access enable register B	$003C_H$	U	CSE, SV	Application Reset	6380
ACCENCS_RDA	ACCENCS read access enable register A	0040_H	U	CSE, SV	Application Reset	6381
ACCENCS_RDB	ACCENCS read access enable register B	0044_H	U	CSE, SV	Application Reset	6381
ACCENCS_VM	ACCENCS VM access enable register	0048_H	U	CSE, SV	Application Reset	6382
ACCENCS_PRS	ACCENCS PRS access enable register	$004C_H$	U	CSE, SV	Application Reset	6383
PROTCE	PROT Register Cyber-Secure Endinit	0058_H	U	SV, PROT	Application Reset	6383
PROTCPUn	PROT Register for PMCSRn Idle control	$005C_H$+n *4	U	SV, PROT	Application Reset	6385
PMCSRn	Power Management Control and Status Register for CPUn	0074_H+n *4	U	(SV, SE)| (ECPUn, SV)	Application Reset	6387
PMCSRCS	Power Management Control and Status Register for CPUcs in CSRM subystem	$008C_H$	U	CE, SV, PCS	Application Reset	6387
PMSWCR1	Standby and Wake-up Control Register 1	0094_H	U	SV, E, P	Cold PowerOn Reset	6389
PMTRCSR0	Power Management Transition Control and Status Register 0	0098_H	U	SV, E, P	Cold PowerOn Reset	6391
PMTRCSR1	Power Management Transition Control and Status Register 1	$009C_H$	U	SV, E, P	Cold PowerOn Reset	6394
PMTRCSR2	Power Management Transition Control and Status Register 2	$00A0_H$	U	SV, E, P	Cold PowerOn Reset	6394
PMTRCSR3	Power Management Transition Control and Status Register 3	$00A4_H$	U	SV, E, P	Cold PowerOn Reset	6396

(table continues...)

Table 1649 (continued) Registers overview - SMM (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PMSTAT0	Power Management Status Register 0	$00AC_H$	U	BE	See 6397	6397
DRPCTRL	Power Management Droop Control and Status Register	$00B0_H$	U	SV, E, P	Cold PowerOn Reset	6400
STSTAT	Start-Up Status Register	$00B4_H$	U	BE	See 6401	6401
SYSCON	System Control Register	$00B8_H$	P	P, SV, E	System Reset	6402
STCON	Start-up Configuration Register	$00BC_H$	P	STP, P	See 6403	6403
SWRSTCON	Software Reset Control Register	$00C0_H$	P	SV, E, P	System Reset	6404
STMEM1	Start-up Memory Register 1	$00C8_H$	P	STP, P	See 6404	6404
STMEM2	Start-up Memory Register 2	$00CC_H$	P	STP, P	See 6407	6407
SHSEQCTRLD	Shut Down Sequence Control Register D	$00F4_H$	U	BE	Cold PowerOn Reset	6407
WRMRSTTRIG	Reset Trigger Status Register for Warm Reset Events	$00FC_H$	U	BE	Cold PowerOn Reset	6408
WRMRSTTRIGH	Highest Reset Trigger Status Register for Warm Reset Events	0100_H	U	BE	Cold PowerOn Reset	6411
RSTSTAT	Reset Type Status Register	0104_H	U	BE	See 6414	6414
RSTSTATH	Highest Reset Type Status Register	0108_H	U	BE	See 6415	6415
RSTSTATCLR	Reset Status Clear Register	0110_H	P	SV, E, P	Cold PowerOn Reset	6417
RSTTRIGCTRLA	Reset Trigger Control Register A	0114_H	P	SV, E, P	PowerOn Reset	6418
RSTTRIGCTRLB	Reset Trigger Control Register B	0118_H	P	SV, E, P	PowerOn Reset	6421
RSTTRIGCTRLC	Reset Trigger Control Register C	$011C_H$	PCS	SV, CE, PCS	PowerOn Reset	6424
SMURSTSTATA	Reset Trigger Register for SMU related Trigger	0120_H	P	BE	Cold PowerOn Reset	6425
SMURSTSTATB	Reset Trigger Register for SMU related Trigger	0124_H	P	BE	Cold PowerOn Reset	6426
USRINFO	User Info Storage Register	0128_H	P	SV, E, P	Cold PowerOn Reset	6427
PDISC	Pad Disable Control Register	$012C_H$	P	SV, E, P	System Reset	6427
ESR0CFG	ESR0 Reset Output Configuration Register	0130_H	P	SV, E, P	System Reset	6428

(table continues...)

Table 1649 (continued) Registers overview - SMM (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ESRCFGx	ESRx Input Configuration Register	0134_H+x *4	P	SV, E, P	PowerOn Reset	6428
ESR0CNTCTRL	ESR0 Reset Counter Control Register	0140_H	P	SV, E, P	See 6430	6430
GPIOESR0	ESR0 GPIO Control Register	0144_H	U	P, E, SV	System Reset	6431
DRVCFGESR0	ESR0 Pad configuration Control Register	0148_H	P	E, SV, P	System Reset	6432
GPIOESR1	ESR1 GPIO Control Register	$014C_H$	U	P, E, SV	System Reset	6434
DRVCFGESR1	ESR1 Pad configuration Control Register	0150_H	P	E, SV, P	System Reset	6435
PROTINIT	PROT Initialisation Configuration Register	0154_H	U	SV, E, P	Application Reset	6437

37.4.3 Module Identification Register

ID Offset address: 0008_H

Module Identification Register PowerOn Reset value: $00C5\ C013_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MOD_NUM															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MOD_TYPE								MOD_REV							
r								r							

Field	Bits	Type	Description
MOD_REV	7:0	r	**Module Revision** Indicates the revision number of the implementation.
MOD_TYPE	15:8	r	**Module Type** This internal marker is fixed to $C0_H$.
MOD_NUM	31:16	r	**Module Number** Indicates the module identification number

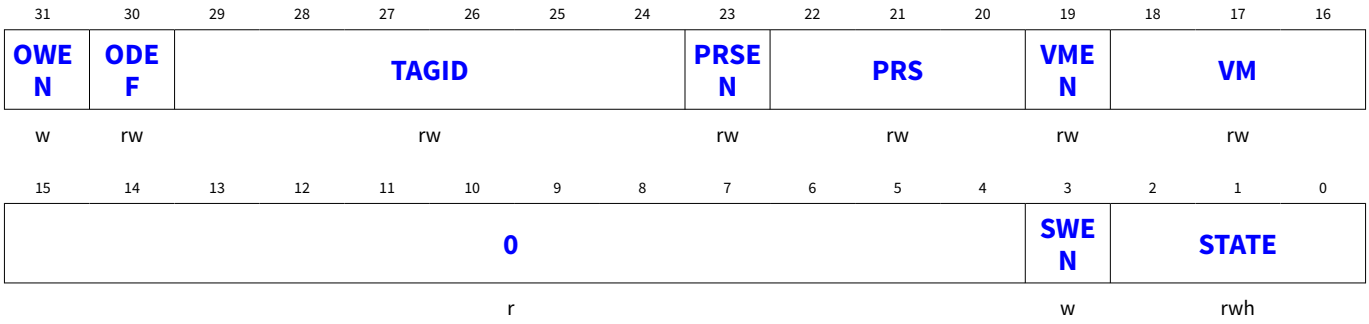
37.4.4 PROT Register Endinit

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner or CS master to update the PROT state as described in the PROT mechanism.

The PROTE register controls lock / unlock of the local Endinit (E) protected control registers.

PROTE Offset address: $000C_H$

PROT Register Endinit Application Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
VM	18:16	rw	**Virtual Machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual Machine definition Enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection Set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.
PRSEN	23	rw	**Protection Set definition Enable for PROT owner** 0_B **Disable**: PRS is not part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner

(table continues...)

(continued)

Field	Bits	Type	Description
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; should be written with 0.

Table 1650 Access mode restrictions of PROTE sorted by descending priority

Mode name	Access mode		Description
PROT_OWNER and write 1 to .**SWEN**	rwh	STATE	PROT owner can modify the PROT state (for allowed and valid transitions)
PROT_OWNER and write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	PROT owner can modify the existing PROT owner
SECURE_MASTER and write 1 to .**SWEN**	rwh	STATE	Secure master can modify the PROT state (for allowed and valid transitions)
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

37.4.5 PROT Register Safe Endinit

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner or CS master to update the PROT state as described in the PROT mechanism.

The PROTSE register controls lock / unlock of the local Safe Endinit (SE) protected control registers.

PROTSE Offset address: 0010_H

PROT Register Safe Endinit Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OWEN	**ODEF**	**TAGID**						**PRSEN**	**PRS**			**VMEN**	**VM**		
w	rw	rw						rw	rw			rw	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												**SWEN**	**STATE**		
r												w	rwh		

Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
VM	18:16	rw	**Virtual Machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual Machine definition Enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection Set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.
PRSEN	23	rw	**Protection Set definition Enable for PROT owner** 0_B **Disable**: PRS is not part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; should be written with 0.

Table 1651 Access mode restrictions of PROTSE sorted by descending priority

Mode name	Access mode		Description
PROT_OWNER and write 1 to .**SWEN**	rwh	STATE	PROT owner can modify the PROT state (for allowed and valid transitions)
PROT_OWNER and write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	PROT owner can modify the existing PROT owner
SECURE_MASTER and write 1 to .**SWEN**	rwh	STATE	Secure master can modify the PROT state (for allowed and valid transitions)
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

37.4.6 PROT Register Cyber-Secure Endinit

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner or CS master to update the PROT state as described in the PROT mechanism.

The PROT_CSE register controls lock / unlock of the local Cyber Secure Endinit (CSE) protected control registers. This protects the ACCENCS group

PROTCSE Offset address: 0014_H

PROT Register Cyber-Secure Endinit Application Reset value: $4C00\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OWEN	**ODEF**	**TAGID**						**PRSEN**	**PRS**			**VMEN**	**VM**		
w	rw	rw						rw	rw			rw	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												**SWEN**	**STATE**		
r												w	rwh		

Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)

(table continues...)

(continued)

Field	Bits	Type	Description
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
VM	18:16	rw	**Virtual Machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual Machine definition Enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection Set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.
PRSEN	23	rw	**Protection Set definition Enable for PROT owner** 0_B **Disable**: PRS is not part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; should be written with 0.

Table 1652 Access mode restrictions of PROTCSE sorted by descending priority

Mode name	Access mode		Description
PROT_OWNER and write 1 to .**SWEN**	rwh	STATE	PROT owner can modify the PROT state (for allowed and valid transitions)
PROT_OWNER and write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	PROT owner can modify the existing PROT owner
SECURE_MASTER and write 1 to .**SWEN**	rwh	STATE	Secure master can modify the PROT state (for allowed and valid transitions)
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

37.4.7 Write access enable register A

This register defines which master TAG ID encodings are enabled or disabled for write accesses to the access protected region.

ACCEN_WRA Offset address: 0018_H

Write access enable register A Application Reset value: $1000\ 0003_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Write access enable for Master TAG ID q** This bit enables write access to the access protected region for transactions with the Master TAG ID q 0_B Disabled for write access 1_B Enabled for write access

37.4.8 Write access enable register B

This register defines which master TAG ID encodings are enabled or disabled for write accesses to the access protected region.

ACCEN_WRB Offset address: $001C_H$

Write access enable register B Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Write access enable for Master TAG ID q** This bit enables write access to the access protected region for transactions with the Master TAG ID q 0_B Disabled for write access 1_B Enabled for write access
0	31:8	r	**Reserved** Read as 0; should be written with 0.

37.4.9 Read access enable register A

This register defines which master TAG ID encodings are enabled or disabled for read accesses to the access protected region.

ACCEN_RDA Offset address: 0020_H

Read access enable register A Application Reset value: FFFF FFFF$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Read access enable for Master TAG ID q** This bit enables read access to the access protected region for transactions with the Master TAG ID q 0_B Disabled for read access 1_B Enabled for read access

37.4.10 Read access enable register B

This register defines which master TAG ID encodings are enabled or disabled for read accesses to the access protected region.

ACCEN_RDB Offset address: 0024_H

Read access enable register B Application Reset value: 0000 00FF$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Read access enable for Master TAG ID q** This bit enables read access to the access protected region for transactions with the Master TAG ID q 0_B Disabled for read access 1_B Enabled for read access
0	31:8	r	**Reserved** Read as 0; should be written with 0.

37.4.11 VM access enable register

This register defines which virtual machine encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each VMx.

ACCEN_VM Offset address: 0028_H

VM access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for VM ID q** This bit enables read access to the access protected region for transactions with the VM ID q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for VM ID q** This bit enables write access to the access protected region for transactions with the VM ID q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

37.4.12 PRS access enable register

This register defines which protection register set encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each PRSx.

ACCEN_PRS Offset address: $002C_H$

PRS access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for PRS q** This bit enables read access to the access protected region for transactions with the PRS q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for PRS q** This bit enables write access to the access protected region for transactions with the PRS q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

37.4.13 ACCENCS write access enable register A

This register defines which master TAG ID encodings are enabled or disabled for write accesses to the access protected region.

ACCENCS_WRA Offset address: 0038_H

ACCENCS write access enable register A Application Reset value: $0000\ 1000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Write access enable for Master TAG ID q** This bit enables write access to the access protected region for transactions with the Master TAG ID q 0_B Disabled for write access 1_B Enabled for write access

37.4.14 ACCENCS write access enable register B

This register defines which master TAG ID encodings are enabled or disabled for write accesses to the access protected region.

ACCENCS_WRB Offset address: $003C_H$

ACCENCS write access enable register B Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Write access enable for Master TAG ID q** This bit enables write access to the access protected region for transactions with the Master TAG ID q 0_B Disabled for write access 1_B Enabled for write access
0	31:8	r	**Reserved** Read as 0; should be written with 0.

37.4.15 ACCENCS read access enable register A

This register defines which master TAG ID encodings are enabled or disabled for read accesses to the access protected region.

ACCENCS_RDA Offset address: 0040_H

ACCENCS read access enable register A Application Reset value: $0000\ 1000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Read access enable for Master TAG ID q** This bit enables read access to the access protected region for transactions with the Master TAG ID q 0_B Disabled for read access 1_B Enabled for read access

37.4.16 ACCENCS read access enable register B

This register defines which master TAG ID encodings are enabled or disabled for read accesses to the access protected region.

ACCENCS_RDB Offset address: 0044_H

ACCENCS read access enable register B Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Read access enable for Master TAG ID q** This bit enables read access to the access protected region for transactions with the Master TAG ID q 0_B Disabled for read access 1_B Enabled for read access
0	31:8	r	**Reserved** Read as 0; should be written with 0.

37.4.17 ACCENCS VM access enable register

This register defines which virtual machine encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each VMx.

ACCENCS_VM Offset address: 0048_H

ACCENCS VM access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for VM ID q** This bit enables read access to the access protected region for transactions with the VM ID q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for VM ID q** This bit enables write access to the access protected region for transactions with the VM ID q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

37.4.18 ACCENCS PRS access enable register

This register defines which protection register set encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each PRSx.

ACCENCS_PRS Offset address: $004C_H$

ACCENCS PRS access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for PRS q** This bit enables read access to the access protected region for transactions with the PRS q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for PRS q** This bit enables write access to the access protected region for transactions with the PRS q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

37.4.19 PROT Register Cyber-Secure Endinit

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner or CS master to update the PROT state as described in the PROT mechanism.

The PROT_CE register controls lock / unlock of the local Cyber Secure Endinit (CE) protected control registers.

PROTCE Offset address: 0058_H

PROT Register Cyber-Secure Endinit Application Reset value: $4C00\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OWEN	ODEF	TAGID						PRSEN	PRS			VMEN	VM		
w	rw	rw						rw	rw			rw	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												SWEN	STATE		
r												w	rwh		

Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
VM	18:16	rw	**Virtual Machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual Machine definition Enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection Set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.
PRSEN	23	rw	**Protection Set definition Enable for PROT owner** 0_B **Disable**: PRS is not part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; should be written with 0.

Table 1653 Access mode restrictions of PROTCE sorted by descending priority

Mode name	Access mode		Description
PROT_OWNER and write 1 to .**SWEN**	rwh	STATE	PROT owner can modify the PROT state (for allowed and valid transitions)
PROT_OWNER and write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	PROT owner can modify the existing PROT owner
SECURE_MASTER and write 1 to .**SWEN**	rwh	STATE	Secure master can modify the PROT state (for allowed and valid transitions)
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

37.4.20 PROT Register for PMCSRn Idle control

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner or CS master to update the PROT state as described in the PROT mechanism.

***Note**: If a CPU is not available in the device the respective register shall not be written.*

PROTCPUn (n=0-5) Offset address: $005C_H+n*4$

PROT Register for PMCSRn Idle control Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OWEN	**ODEF**	**TAGID**						**PRSEN**	**PRS**			**VMEN**	**VM**		
w	rw	rw						rw	rw			rw	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												**SWEN**	**STATE**		
r												w	rwh		

Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)

(table continues...)

(continued)

Field	Bits	Type	Description
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
VM	18:16	rw	**Virtual Machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual Machine definition Enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection Set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.
PRSEN	23	rw	**Protection Set definition Enable for PROT owner** 0_B **Disable**: PRS is not part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; should be written with 0.

Table 1654 Access mode restrictions of PROTCPUn (n=0-5) sorted by descending priority

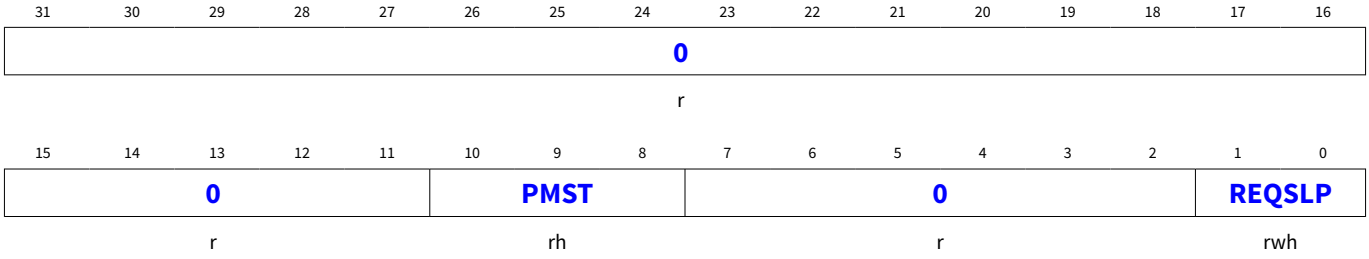
Mode name	Access mode		Description
PROT_OWNER and write 1 to .**SWEN**	rwh	STATE	PROT owner can modify the PROT state (for allowed and valid transitions)
PROT_OWNER and write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	PROT owner can modify the existing PROT owner
SECURE_MASTER and write 1 to .**SWEN**	rwh	STATE	Secure master can modify the PROT state (for allowed and valid transitions)
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

37.4.21 Power Management Control and Status Register for CPUn

***Note**: If a CPU is not available in the device the respective register shall not be written.*

PMCSRn (n=0-5) Offset address: 0074_H+n*4

Power Management Control and Status Register for CPUn Application Reset value: $0000\ 0100_H$



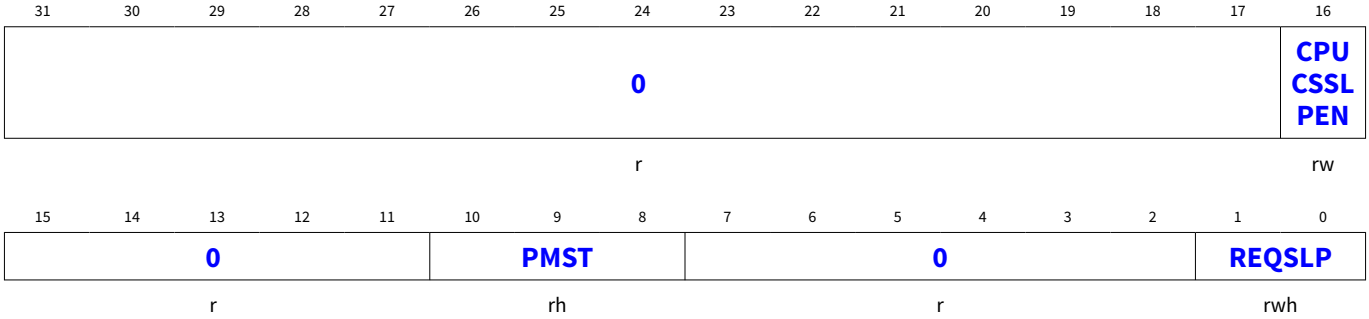
Field	Bits	Type	Description
REQSLP	1:0	rwh	**Idle Mode and Sleep Mode Request** In Idle Mode or Sleep Mode, these bits are cleared in response to an interrupt for the CPU, or when bit 15 of the corresponding CPU Watchdog Timer register (bit WDTCPUx_STAT.TIM[15]) changes from 0 to 1. In Standby Mode, these bits are cleared on wake-up. 00_B Request CPU Run Mode 01_B Request CPU Idle Mode 10_B Request System Sleep Mode 11_B Request System Standby Mode
PMST	10:8	rh	**Power management status** This bit field reflects the current status of the CPU. 000_B Reserved 001_B Normal Run Mode. Or Halt Mode for all CPUs except CPU0. 010_B CPU Idle Mode requested 011_B CPU Idle Mode acknowledged 100_B Sleep Mode requested 101_B Reserved, do not use this combination 110_B Standby Mode requested 111_B Reserved, do not use this combination
0	7:2, 31:11	r	**Reserved** Read as 0; should be written with 0.

37.4.22 Power Management Control and Status Register for CPUcs in CSRM subystem

PMCSRCS Offset address: $008C_H$

Power Management Control and Status Register for CPUcs in CSRM subystem Application Reset value: $0000\ 0100_H$

37 System Mode Management (SMM)



Field	Bits	Type	Description
REQSLP	1:0	rwh	**Idle Mode and Sleep Mode Request** In Idle Mode or Sleep Mode, these bits are cleared in response to an interrupt for the CPU, or when bit 15 of the corresponding CPU Watchdog Timer register (bit WDTCPUxSR.TIM[15]) changes from 0 to 1. In Standby Mode, these bits are cleared on wake-up. 00_B Request CPU Run Mode 01_B Request CPU Idle Mode 10_B Request System Sleep Mode 11_B Request System Standby Mode
PMST	10:8	rh	**Power management Status** This bit field reflects the current status of the CPU. 000_B Reserved 001_B Normal Run Mode. Or Halt Mode for CSRM CPU. 010_B CPU Idle Mode requested 011_B CPU Idle Mode acknowledged 100_B Sleep Mode requested 101_B Reserved, do not use this combination 110_B Standby Mode requested 111_B Reserved, do not use this combination
CPUCSSLPEN	16	rw	**Enable of CPUcs IDLE Mode on SLEEP request** Enable of CPUcs IDLE mode in case of a global SLEEP request or a global IDLE request based on CPUIDLSEL configuration. CPUcs is not set into idle on a SLEEP request by default reset value. In case of global STANDBY request, all CPUs including CPUcs are set into IDLE mode before going into STANDBY mode. In sleep mode, if CPUcs modifies this register, the change would be considered only for the next sleep request cycle. 0_B CPUcs IDLE request disabled on SLEEP request or global IDLE request based on CPUIDLSEL configuration. 1_B CPUcs IDLE request enabled on SLEEP request or global IDLE request based on CPUIDLSEL configuration
0	7:2, 15:11, 31:17	r	**Reserved** Read as 0; should be written with 0.

37.4.23 Standby and Wake-up Control Register 1

PMSWCR1 Offset address: 0094_H

Standby and Wake-up Control Register 1 Cold PowerOn Reset value: $0100\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	STBYEV			STBYEVEN	CPUSEL			0							
r	rw			w	rw			r							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			IRADIS	0	CPUIDLSEL			PROCMODEEN	0						
r			rw	r	rw			rw	r						

Field	Bits	Type	Description
PROCMODEEN	7	rw	**Global IDLE Mode Enable for Processing Units** Global Enable which enables entry to System Modes for all processing units and PMCSRx.REQSLP function. REQSLP request is taken only when this bit is set to 1. 0_B Processing Unit IDLE, SLEEP and STANDBY mode requests via REQSLP register bitfield is disabled 1_B Processing Unit IDLE, SLEEP and STANDBY mode requests via REQSLP register bitfield is enabled

(table continues...)

(continued)

Field	Bits	Type	Description
CPUIDLSEL	10:8	rw	**CPU selection for Idle mode** This bit field allows a CPUx to issue Idle request to other CPUs in addition to itself. A request for Idle via PMCSRx.REQSLP=01 by CPUx will also trigger Idle requests to all other CPUs. ***Note**: All other CPUIDLSEL bit combinations are reserved. CPUcs cannot be selected as master CPU.* *PPU is not available on all devices of the AURIX TC4x family.* *If a CPU is not available in the device the respective bitfield combination is shall not be written.* 000_B Entry to the respective Idle mode is decided by each individual CPU. 001_B CPU0 Idle request will send all CPUs including CPUcs and PPU in Idle. 010_B CPU1 Idle request will send all CPUs including CPUcs and PPU in Idle. 011_B CPU2 Idle request will send all CPUs including CPUcs and PPU in Idle. 100_B CPU3 Idle request will send all CPUs including CPUcs and PPU in Idle. 101_B CPU4 Idle request will send all CPUs including CPUcs and PPU in Idle. 110_B CPU5 Idle request will send all CPUs including CPUcs and PPU in Idle.
IRADIS	12	rw	**Idle-Request-Acknowledge Sequence Disable** This bit enables Idle Request to CPUs and CPUcs on Standby entry. PPU is not set into Idle in this case. Acknowledge is not awaited if bit is set. IRADIS bit has no effect in case of Standby entry triggered via PWRWKEN register bit. This bit shall be set before Standby entry to disable Idle request acknowledge sequence so that Standby request is not blocked by a pending Idle acknowledge feedback. ***Note**: PPU is not available on all devices of the AURIX TC4x family.* 0_B Idle-Request-Acknowledge Sequence issued on Standby entry. 1_B Idle-Request-Acknowledge Sequence skipped on Standby entry.

(table continues...)

(continued)

Field	Bits	Type	Description
CPUSEL	26:24	rw	**CPU selection for Sleep and Standby mode** ***Note**: All other CPUSEL bit combinations are reserved. CPUcs cannot be master CPU.* *If a CPU is not available in the device the respective bitfield combination is shall not be written.* 000_B CPU0 is taken as default master and can trigger power down modes. 001_B Only CPU0 can trigger power down modes. 010_B Only CPU1 can trigger power down modes. 011_B Only CPU2 can trigger power down modes. 100_B Only CPU3 can trigger power down modes. 101_B Only CPU4 can trigger power down modes. 110_B Only CPU5 can trigger power down modes. 111_B Entry to power down modes is unanimously decided by all the CPUs. CPUcs is not included in decision.
STBYEVEN	27	w	**STBYEV Bit protection** 0_B Bit STBYEV is not updated. 1_B Bit STBYEV can be updated.
STBYEV	30:28	rw	**Standby Entry Event Configuration** ***Note**: All other bit combinations are reserved.* 000_B Standby Entry triggered by setting PMCSRx.REQSLP register bit (Default). 100_B Standby Entry triggered on ESR1 / NMI assertion.
0	6:0, 11, 23:13, 31	r	**Reserved** Read as 0; should be written with 0.

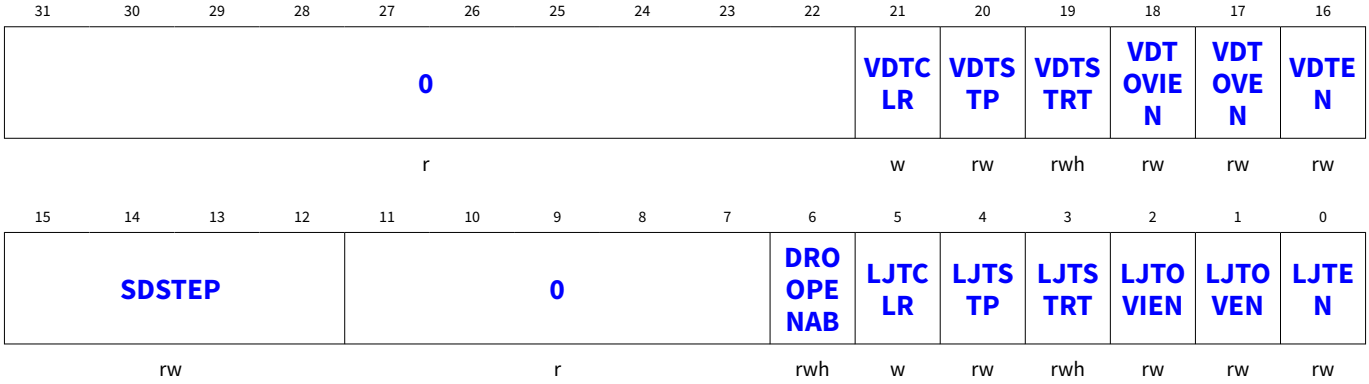
Table 1655 Access mode restrictions of PMSWCR1 sorted by descending priority

Mode name	Access mode		Description
write 1 to .**STBYEVEN**	rw	STBYEV	
(default)	r	STBYEV	

37.4.24 Power Management Transition Control and Status Register 0

PMTRCSR0 Offset address: 0098_H

Power Management Transition Control and Status Register 0 Cold PowerOn Reset value: $0000\ 0040_H$



Field	Bits	Type	Description
LJTEN	0	rw	**Load Jump Timer Enable** This bit field enables the usage of load jump timer. 0_B Load Jump Timer inactive 1_B Load Jump Timer active
LJTOVEN	1	rw	**Load Jump Timer Overflow Enable** This bit field enables the update of LJTOV status bit on timer overflow or time out. 0_B LJTOV bit is not updated on a Load Jump Timer overflow. 1_B LJTOV bit is updated on a Load Jump Timer overflow.
LJTOVIEN	2	rw	**Load Jump Timer Overflow Interrupt Enable** This bit field enables the activation of interrupt on timer overflow or time out. 0_B LJTOV interrupt is inactive. 1_B LJTOV interrupt is activated on a Load Jump Timer overflow.
LJTSTRT	3	rwh	**Load Jump Timer Start** This bit field starts Load jump timer. This is intended for test purposes. The LJTSTRT remains set on a write and is cleared when LJTOV bit is set if LJTOVEN bit is enabled. 0_B Load Jump Timer status not changed. 1_B Load Jump Timer started.
LJTSTP	4	rw	**Load Jump Timer Stop** This bit field stops Load jump timer. This is intended for test purposes. The LJTSTP remains set on a write and is to be explicitly cleared by software. The LJTSTP stops the counter at the current value and timer re-starts from that value when LJTSTP is cleared and LJTSTRT is set. 0_B Load Jump Timer status not changed. 1_B Load Jump Timer stopped.
LJTCLR	5	w	**Load Jump Timer Clear** This bit field clear Load jump timer count. This is intended for test purposes. This bit resets LJT and clears LJTRUN if LJTEN bit is set. 0_B Load Jump Count status not changed. 1_B Load Jump Timer Count cleared.

(table continues...)

(continued)

Field	Bits	Type	Description
DROOPENAB	6	rwh	**Selection between Load Jump Sequencing Units A or B** 0_B Load Jump Sequencing Unit A is active 1_B Load Jump Sequencing Unit B is active
SDSTEP	15:12	rw	**Droop Voltage Step(vdroop_step_i)** This bit field defines the voltage offset for droop compensation on a load jump to the EVRC setpoint value. The request is made via VDROOPREQ (sx_pms_evrc_droop_r_cntr_o) on an anticipated load jump with a voltage offset equal to the SDSTEP x LSB mV. The droop step is a positive offset if VDROOPREQ = 01b and is a negative offset if VDROOPREQ = 10b and no offset is applied if VDROOPREQ = 00b. Maximum Droop = 15 * VTRKCORELSB datasheet parameter ***Note**: Actual VDD voltage during droop time must be within the allowed voltage limits over lifetime as documented in datasheet section "Absolut maximum ratings"*
VDTEN	16	rw	**Voltage Droop Timer Enable** This bit field enables the usage of Voltage Droop timer. 0_B Voltage Droop Timer inactive 1_B Voltage Droop Timer active
VDTOVEN	17	rw	**Voltage Droop Timer Overflow Enable** This bit field enables the update of VDTOV status bit on timer overflow or time out. 0_B VDTOV bit is not updated on a Voltage Droop Timer overflow. 1_B VDTOV bit is updated on a Voltage Droop Timer overflow.
VDTOVIEN	18	rw	**Voltage Droop Timer Overflow Interrupt Enable** This bit field enables the activation of interrupt on timer overflow or time out. 0_B VDTOV interrupt is inactive. 1_B VDTOV interrupt is activated on a Voltage Droop Timer overflow.
VDTSTRT	19	rwh	**Voltage Droop Timer Start** This bit field starts Voltage Droop timer. This is intended for test purposes. The VDTSTRT remains set on a write and is cleared when VDTOV bit is set if VDTOVEN bit is enabled. 0_B Voltage Droop Timer status not changed. 1_B Voltage Droop Timer started.
VDTSTP	20	rw	**Voltage Droop Timer Stop** This bit field stops Voltage Droop timer. SCU cancels the droop request via signal sx_pms_evrc_droop_r_cntr_o = 00. This is intended for test purposes. The VDTSTP remains set on a write and is to be explicitly cleared by software. The VDTSTP stops the counter at the current value and timer re-starts from that value when VDTSTP is cleared and VDTSTRT is set. 0_B Voltage Droop Timer status not changed. 1_B Voltage Droop Timer stopped.

(table continues...)

(continued)

Field	Bits	Type	Description
VDTCLR	21	w	**Voltage Droop Timer Clear** This bit field clears Voltage Droop timer count. This is intended for test purposes. This bit resets VDT and clears VDTRUN if VDTEN bit is set. 0_B Voltage Droop Count status not changed. 1_B Voltage Droop Timer Count cleared.
0	11:7, 31:22	r	**Reserved** Read as 0; should be written with 0.

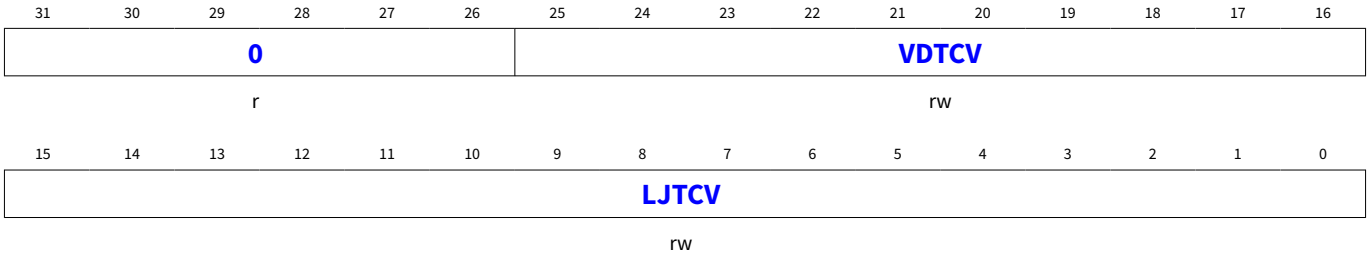
37.4.25 Power Management Transition Control and Status Register 1

PMTRCSR1

Offset address: $009C_H$

Power Management Transition Control and Status Register 1

Cold PowerOn Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
LJTCV	15:0	rw	**Load Jump Timer Compare Setpoint Value** This bit field defines the compare setpoint value of Load Jump timer. The compare event would lead to LJTOV bit being set and LJT interrupt being raised. The LJTRUN status bit, LDJMPREQ bit and LJTCNT value is reset to 0 on a compare event. X µs is the compare value. LSB = 1 µs. Total range = 65.5 ms.
VDTCV	25:16	rw	**Voltage Droop Timer Compare Setpoint Value** This bit field defines the compare setpoint value of Voltage Droop timer. The compare event would lead to VDTOV bit being set and VDT interrupt being raised. The VDTRUN status bit, VDROOPREQ bit and VDTCNT value is reset to 0 on a compare event. X µs is the compare value. LSB = 1 µs. Total range = 1023 µs.
0	31:26	r	**Reserved** Read as 0; should be written with 0.

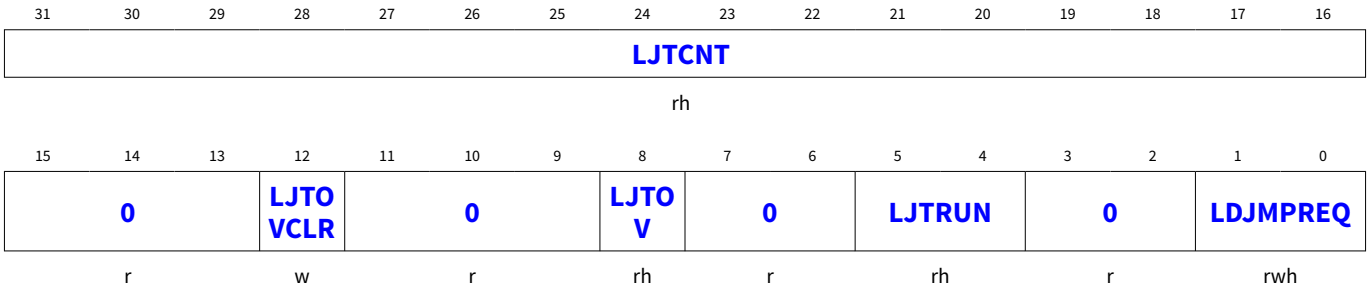
37.4.26 Power Management Transition Control and Status Register 2

PMTRCSR2

Offset address: $00A0_H$

Power Management Transition Control and Status Register 2

Cold PowerOn Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
LDJMPREQ	1:0	rwh	**Load Jump Request** This bit requests a Load Jump consequently leading to Load Jump Timer start and LJTRUN bit being set if LJTEN=1. The request is not taken if LJTRUN bit is already in set state and LJT is currently running. The request is not taken if VDTRUN bit is already in set state and VDT is currently running. The request is also not taken if (LJTOV bit is set AND LJTOVEN bit is enabled). The request is also not taken if (VDTOV bit is set AND VDTOVEN bit is enabled). The LDJMPREQ bit is cleared on a compare overflow. ***Note:*** *All other bit combinations are reserved.* 00_B Load Jump Timer inactive 01_B Load Jump Request made and taken. Load Jump Timer activated.
LJTRUN	5:4	rh	**Load Jump Timer Run Status** This status bit indicates that the Load Jump timer is currently running and a Load Jump is currently taking place. The LJTRUN bit is cleared on a compare overflow. ***Note:*** *All other bit combinations are reserved.* 00_B Load Jump and Load Jump Timer inactive 01_B A SW triggered Load Jump active and Load Jump Timer active
LJTOV	8	rh	**Load Jump Timer Overflow Status** This status bit indicates that the Load Jump timer compare match has happened. If LJTOVEN bit is enabled, then LJTOV can only be cleared explicitly via bit. If LJTOVEN bit is disabled, LJTOV is cleared on a taken Load Jump Request. LJTOV being set will lead to an interrupt if LJTOVIEN is enabled. 0_B Load Jump Timer compare overflow has not happened. 1_B Load Jump Timer compare overflow has happened.
LJTOVCLR	12	w	**Load Jump Timer Overflow Status Clear** This bit clears LJTOV status bit and sets VDROOPREQ and LDJMPREQ to 0 if LJTOVEN bit is enabled. This bit always reads as 0. 0_B This clear bit has no effect on Load Jump Timer overflow flag. 1_B Load Jump Timer overflow flag is cleared.

(table continues...)

(continued)

Field	Bits	Type	Description
LJTCNT	31:16	rh	**Load Jump Timer Value** This bit field reflects the current Load Jump timer value. LJTCNT value is cleared on timer overflow and on a taken Load Jump Request X µs is the compare value. LSB = 1 µs. Total range = 65.5 ms.
0	3:2, 7:6, 11:9, 15:13	r	**Reserved** Read as 0; should be written with 0.

37.4.27 Power Management Transition Control and Status Register 3

PMTRCSR3 Offset address: $00A4_H$

Power Management Transition Control and Status Register 3 Cold PowerOn Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0						VDTCNT									
r						rh									

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			VDTOVCLR	0			VDTOV	0		VDTRUN		0		VDROOPREQ	
r			w	r			rh	r		rh		r		rwh	

Field	Bits	Type	Description
VDROOPREQ	1:0	rwh	**Voltage Droop Request** This bit requests a Voltage Droop consequently leading to Voltage Droop Timer start and VDTRUN bit being set . The request is not taken if VDTRUN bit is already in set state and VDT is currently running. The request is also not taken if (VDTOV bit is set AND VDTOVEN bit is enabled). The droop step is a positive offset if sx_pms_evrc_droop_r_cntr_o = 01 and is a negative offset if sx_pms_evrc_droop_r_cntr_o = 10 and no offset is applied if sx_pms_evrc_droop_r_cntr_o = 00 and is applied immediately. 00_B Voltage Droop and Voltage Droop Timer inactive 01_B A Positive Voltage Droop Request made and taken. Voltage Droop Timer activated. 10_B A Negative Voltage Droop Request made and taken. Voltage Droop Timer activated. 11_B Voltage Droop and Voltage Droop Timer inactive

(table continues...)

(continued)

Field	Bits	Type	Description
VDTRUN	5:4	rh	**Voltage Droop Timer Run Status** This status bit indicates that the Voltage Droop timer is currently running and a Voltage Droop is currently taking place. The VDTRUN bit is cleared on a compare overflow. ***Note**: All other bit combinations are reserved.* 00_B Voltage Droop and Voltage Droop Timer inactive 01_B A SW triggered Voltage Droop active and Voltage Droop Timer active
VDTOV	8	rh	**Voltage Droop Timer Overflow Status** This status bit indicates that the Voltage Droop timer compare match has happened. If VDTOVEN bit is enabled, then VDTOV can only be cleared by explicitly via bit. if VDTOVEN bit is disabled, VDTOV is cleared on a taken Voltage Droop Request. VDTOV being set will lead to an interrupt if VDTOVIEN is enabled. If SDVOK is set by EVRC before VDT compare match, VDTOV bit is not set. 0_B Voltage Droop Timer compare overflow has not happened. 1_B Voltage Droop Timer compare overflow has happened.
VDTOVCLR	12	w	**Voltage Droop Timer Overflow Status Clear** This bit clears VDTOV status bit if VDTOVEN bit is enabled. If VDTOVEN bit is disabled, this bit has no effect. This bit always reads as 0. 0_B This clear bit has no effect on Voltage Droop Timer overflow flag. 1_B Voltage Droop Timer overflow flag is cleared.
VDTCNT	25:16	rh	**Voltage Droop Timer Value** This bit field reflects the current Voltage Droop timer value. VDTCNT value is cleared on timer overflow and on a taken Voltage Droop Request. X µs is the compare value. LSB = 1 µs. Total range = 65.5 ms.
0	3:2, 7:6, 11:9, 15:13, 31:26	r	**Reserved** Read as 0; should be written with 0.

37.4.28 Power Management Status Register 0

Bitfields regarding CPUs or PPUs which are not available on derviatives of the TC4x family are read as 0

PMSTAT0 Offset address: $00AC_H$

Power Management Status Register 0 Reset values see: Table 1656

37 System Mode Management (SMM)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0									Res	CPU 5LS	CPU 4LS	CPU 3LS	CPU 2LS	CPU 1LS	CPU 0LS
r									r	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								PPU	CSRM	CPU 5	CPU 4	CPU 3	CPU 2	CPU 1	CPU 0
r								rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
CPU0	0	rh	**CPU0 Status** This bit field reflects the current status of CPU0. 0_B CPU0 is in Halt or Idle Mode 1_B CPU0 is in Normal Run Mode
CPU1	1	rh	**CPU1 Status** This bit field reflects the current status of CPU1. 0_B CPU1 is in Halt or Idle Mode 1_B CPU1 is in Normal Run Mode
CPU2	2	rh	**CPU2 Status** This bit field reflects the current status of CPU2. 0_B CPU2 is in Halt or Idle Mode 1_B CPU2 is in Normal Run Mode
CPU3	3	rh	**CPU3 Status** This bit field reflects the current status of CPU3. 0_B CPU3 is in Halt or Idle Mode 1_B CPU3 is in Normal Run Mode
CPU4	4	rh	**CPU4 Status** This bit field reflects the current status of CPU4. 0_B CPU4 is in Halt or Idle Mode 1_B CPU4 is in Normal Run Mode
CPU5	5	rh	**CPU5 Status** This bit field reflects the current status of CPU5. 0_B CPU5 is in Halt or Idle Mode 1_B CPU5 is in Normal Run Mode
CSRM	6	rh	**CSRM CPUcs status** This bit field reflects the current status of CPUcs in Cyber Security Module. 0_B CPUcs is in Halt or Idle Mode 1_B CPUcs is in Normal Run Mode

(table continues...)

(continued)

Field	Bits	Type	Description
PPU	7	rh	**PPU Status** This bit field reflects the current status of PPU. ***Note**: PPU not available on all devices of the TC4x family.* 0_B PPU is in Halt or Idle Mode 1_B PPU is in Normal Run Mode
CPU0LS	16	rh	**CPU0LS Status** This bit field reflects the current status of CPU0 Lockstep Checker Core. 0_B CPU0LS is disabled 1_B CPU0LS is enabled
CPU1LS	17	rh	**CPU1LS Status** This bit field reflects the current status of CPU1 Lockstep Checker Core. 0_B CPU1LS is disabled 1_B CPU1LS is enabled
CPU2LS	18	rh	**CPU2LS Status** This bit field reflects the current status of CPU2 Lockstep Checker Core. 0_B CPU2LS is disabled 1_B CPU2LS is enabled
CPU3LS	19	rh	**CPU3LS Status** This bit field reflects the current status of CPU3 Lockstep Checker Core. 0_B CPU3LS is disabled 1_B CPU3LS is enabled
CPU4LS	20	rh	**CPU4LS Status** This bit field reflects the current status of CPU4 Lockstep Checker Core. 0_B CPU4LS is disabled 1_B CPU4LS is enabled
CPU5LS	21	rh	**CPU5LS Status** This bit field reflects the current status of CPU5 Lockstep Checker Core. 0_B CPU5LS is disabled 1_B CPU5LS is enabled
0	15:8, 31:23	r	**Reserved** Read as 0.

Table 1656 Reset values of PMSTAT0

Reset type	Reset value	Note
Application Reset	0041 0041_H	
After Boot-FW Value	0000 0000 X--- ---- 0000 0000 0100 0001_B	

37.4.29 Power Management Droop Control and Status Register

DRPCTRL Offset address: $00B0_H$

Power Management Droop Control and Status Register Cold PowerOn Reset value: $0000\ 1000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															LJT
r															rwh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ERRCLR	BUSY	V_SP	V_NOM	ERR	ABORT	DIR	STEPS				DWELL_CNT				REQ
w	rh	rh	rh	rh	rw	rw	rw				rw				w

Field	Bits	Type	Description
REQ	0	w	**Set to trigger new voltage droop compensation activity.** When REQ Droop request is set to 1, droop request, direction and voltage step information are transferred to EVRC. The request is ignored if BUSY or ERR bit is currently set. Read as 0. 0_B No droop control request 1_B Droop control request
DWELL_CNT	4:1	rw	**Dwell timer count delay keeping voltage before droop deassertion** The DWELL counter starts running once V_SP is set from 0 to 1. The Dwell counter provides a delay with a LSB granularity of ~ 5us. When DWELL time has expired, the droop request is deasserted to EVRC
STEPS	8:5	rw	**Target number of steps to raise or lower voltage level** Control the target number of steps to bring VDD to either to higher or lower side of the nominal voltage level. Voltage is raised in 5mV steps from nominal voltage level to target number of steps. ***Note**: Actual VDD voltage during droop time must be within the allowed voltage limits over lifetime as documented in datasheet section "Absolut maximum ratings"*
DIR	9	rw	**Definiton of the direction of the voltage (rising/reducing)** 0_B Voltage set point is higher than nominal 1_B Voltage set point is lower than nominal
ABORT	10	rw	**Abort control** 0_B Do not abort 1_B Voltage level at voltage compensation activity shall be aborted and supply voltage shall be brought back to the nominal voltage level (indicated by V_NOM).
ERR	11	rh	**Error indicator** 0_B No Error has occurred during the voltage compensation activity 1_B An Error has occurred during the voltage compensation activity

(table continues...)

(continued)

Field	Bits	Type	Description
V_NOM	12	rh	**Voltage at nominal level** On a new droop request to EVRC (REQ=1) , V_NOM is set to 0 on an acknowledge from EVRC that voltage droop has started. On the deassertion of the droop request after DWELL counter expiry, the V_NOM is set back to 1 when EVRC nominal voltage is restored. When V_NOM is set from 0 to 1, interrupt is triggered. During Voltage drooping phase and Load jump execution phase V_NOM = 0 0_B Current voltage is not at nominal supply voltage level. 1_B Current voltage is at nominal supply voltage level.
V_SP	13	rh	**Voltage Set Point** After the voltage droop phase is completed by EVRC, V_SP = 1 is set to indicate that new voltage set point has reached and load jump can be triggered. When V_SP=1 is set, interrupt is triggered. After expiry of DWELL counter, the droop request is deasserted and V_SP = 0 is set. 0_B Current voltage is not at voltage set point. 1_B Current voltage is at voltage set point.
BUSY	14	rh	**Busy status of droop compensation activity** When REQ request is taken, BUSY flag is set to indicate that no further request can be taken. The BUSY flag is deasserted when nominal voltage is reached again (V_NOM=1) 0_B No droop compensation activity currently running. 1_B Droop compensation activity is on-going.
ERRCLR	15	w	**Clear ERR** Write to ERRCLR bit clears ERR flag and LJT bit so that next Droop REQ can be given. Read as 0.
LJT	16	rwh	**Load Jump Taken flag set by SW during DWELL time** When load jump is triggered by user software, the flag shall be set before DWELL counter has expired. The flag is reset on ERRCLR or REQ write
0	31:17	r	**Reserved** Read as 0; should be written with 0.

37.4.30 Start-Up Status Register

STSTAT Offset address: $00B4_H$

Start-Up Status Register Reset values see: Table 1657

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res				0		MODE2	RAMINT	Res	0	Res	SPDEN	TRSTL	Res		
r				r		rh	rh	r	r	r	rh	rh	r		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MODE1	Res							RSVD_0							
rh	r							rh							

Field	Bits	Type	Description
RSVD_0	7:0	rh	**Reserved** Read as XXXX XXXX
MODE1	15	rh	**MODE** This bit indicates if the Test Mode is entered or not. Testmode is entered if both MODE1=0 AND MODE2=0 together. 0_B A Test Mode can be selected 1_B Normal Mode is selected
TRSTL	19	rh	**TRSTL Status** This bit simply displays the value of TRSTL.
SPDEN	20	rh	**Single Pin DAP Mode Enable** 0_B Single Pin DAP Mode is disabled 1_B Single Pin DAP Mode is enabled
RAMINT	24	rh	**RAM Content Security Integrity** In normal operation this bit can be set or cleared by the application (via SYSCON.RAMINTM). If a test boot mode is entered, the bit is automatically cleared (and cannot be set again in test mode) because the content may have been altered ***Note**: This bit is reset only by a cold power-on reset.* 0_B RAM Security Integrity cannot be guaranteed 1_B RAM Security Integrity maintained
MODE2	25	rh	**MODE** This bit indicates if the Test Mode is entered or not. Testmode is entered if both MODE1=0 AND MODE2=0. 0_B A Test Mode can be selected 1_B Normal Mode is selected
0	22, 27:26	r	**Reserved** Read as 0.

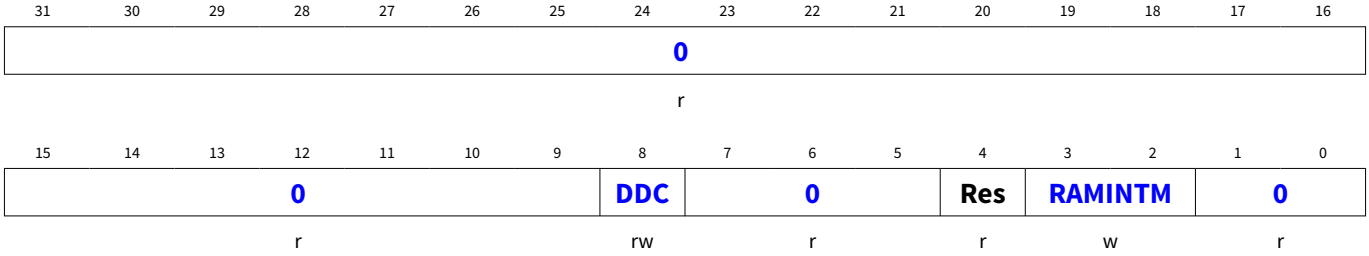
Table 1657 Reset values of STSTAT

Reset type	Reset value	Note
After Boot-FW Value	---- 0010 –X–0 ––00 1000 0000 XXXX $XXXX_B$	Bit-fields with unknown values depend on fuse or reset value.

37.4.31 System Control Register

SYSCON Offset address: $00B8_H$

System Control Register System Reset value: $0000\ 0000_H$

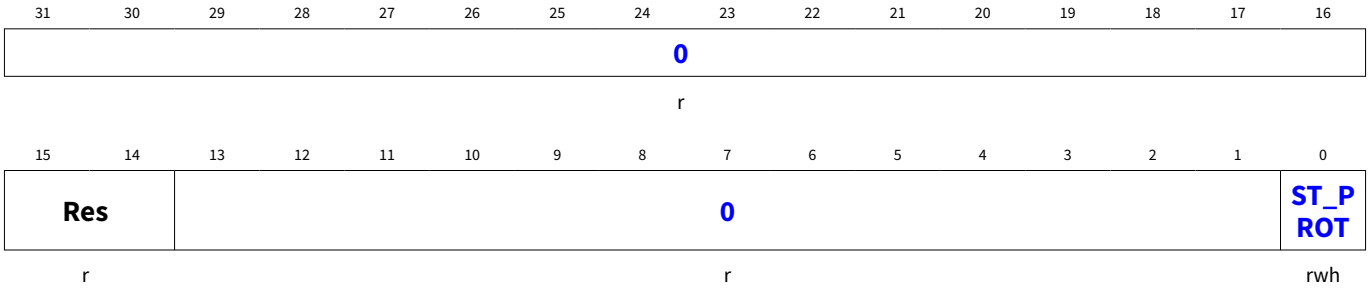


Field	Bits	Type	Description
RAMINTM	3:2	w	**RAM Integrity Modify** Read as 0. 00_B No effect 01_B Set STSTAT.RAMINT (No effect in test mode) 10_B Clear STSTAT.RAMINT 11_B No effect
DDC	8	rw	**Disable DXCPL** 0_B DXCPL not disabled 1_B DXCPL disabled
0	1:0, 7:5, 31:9	r	**Reserved** Read as 0; should be written with 0.

37.4.32 Start-up Configuration Register

STCON Offset address: $00BC_H$

Start-up Configuration Register Reset values see: Table 1658



Field	Bits	Type	Description
ST_PROT	0	rwh	**Start-up Protection Setting** This bit is also cleared by an Application Reset. ST_PROT is set by Boot Firmware before releasing the control to user software. ST_PROT is automatically set by hardware when a shutdown trap occurs. 0_B Start-up code is executed. Start-up protection is disabled. 1_B Start-up code protection is active

(table continues...)

(continued)

Field	Bits	Type	Description
0	13:1, 31:16	r	**Reserved** Read as 0; should be written with 0.

Table 1658 Reset values of STCON

Reset type	Reset value	Note
Application Reset	0000 0000$_H$	
After Boot-FW Value	0000 0001$_H$	

37.4.33 Software Reset Control Register

This register controls the Reset operation triggered by Software.

SWRSTCON Offset address: 00C0$_H$

Software Reset Control Register System Reset value: 0000 0000$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res								0						SWRSTREQ	Res
r								r						w	r

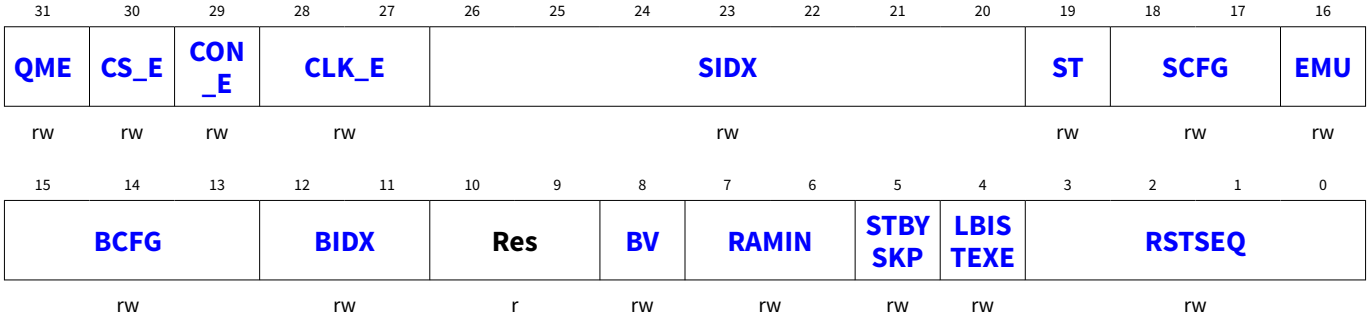
Field	Bits	Type	Description
SWRSTREQ	1	w	**Software Reset Request** This bit is automatically cleared and read always as zero. 0_B No SW Reset is requested 1_B A SW Reset request trigger is generated
0	7:2, 31:16	r	**Reserved** Read as 0; should be written with 0.

37.4.34 Start-up Memory Register 1

STMEM1 indicates the boot status to the user application SW (SYS_APPSW). The detailed bit encoding is shown below. The value is dynamically updated after Firmware execution.

STMEM1 Offset address: 00C8$_H$

Start-up Memory Register 1 Reset values see: Table 1659



Field	Bits	Type	Description
RSTSEQ	3:0	rw	**Processed reset sequence by FW** Processed reset sequence by FW. 1_H AppRST sequence was executed 2_H SysRST sequence was executed 4_H WarmPORST sequence was executed 8_H ColdPORST sequence was executed by FW
LBISTEXE	4	rw	**LBIST was executed with the current RSTSEQ** 0_B LBIST was not executed 1_B LBIST was executed
STBYSKP	5	rw	**Standby RAMs skipped during RAMINIT** 0_B Standby RAMs not skipped 1_B Standby RAMs skipped
RAMIN	7:6	rw	**Reserved for BFW usage** Reserved for BFW usage.
BV	8	rw	**BMHD valid flag** BMHD valid flag. 0_B invalid 1_B valid
BIDX	12:11	rw	**Index of the valid BMHD** Index of the valid BMHD.
BCFG	15:13	rw	**Start up mode effectively taken by SSW** Start up mode effectively taken by SSW. 011_B ASC Bootstrap loader 100_B Generic Bootstrap loader 101_B ABM (ASC BSL on fail, only selected via pins) 110_B ABM (Generic BSL on fail, when selected via pins) 111_B Boot from Flash
EMU	16	rw	**Device emulation indication** 0_B Emulation is not active 1_B Emulation is active

(table continues...)

(continued)

Field	Bits	Type	Description
SCFG	18:17	rw	**SWAP configuration** 00_B no SWAP configured by SSW (full address space active with the default map). 01_B SWAP A configured (address region A active, B inactive). 10_B SWAP B configured (address region B active, A inactive) 11_B reserved
ST	19	rw	**SWAP target** SWAP target. 0_B SWAP configuration done based on UCB_SWAP_ORIG 1_B SWAP configuration done based on UCB_SWAP_COPY.
SIDX	26:20	rw	**Index x of SWAP entry in UCB_SWAP_ORIG/COPY containing valid marker** Index x of SWAP entry in UCB_SWAP_ORIG/COPY containing valid marker has been installed.
CLK_E	28:27	rw	**Error during CLK handling during Application Reset only** 00_B No ramp performed 01_B Legacy behavior. No ramp performed. (RSTSEL = 0) 10_B Ramp performed successfully. 11_B Error during CLK handling. Fallback CLK selected.
CON_E	29	rw	**CON_E** User configuration errored flag, indicating that invalid set of configuration was provided to FW via UCB e.g. SOTA Marker 0_B no error detected 1_B error detected
CS_E	30	rw	**CSRM error flag indicating that CSRM boot was not successful** Register-write can only set the bitfield, but not reset it. 0_B No error 1_B Start-up error happened
QME	31	rw	**QM entered flag** QM indication. Register-write can only set the bitfield, but not reset it. 0_B No QM part entered. 1_B QM parts of SYS_FW were entered

Table 1659 Reset values of STMEM1

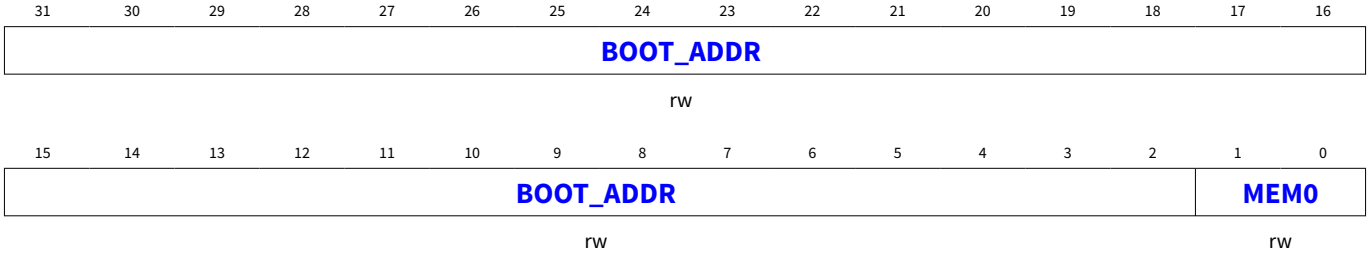
Reset type	Reset value	Note
System Reset	$0000\ 0000_H$	
After Boot-FW Value	$----\ ----_H$	

37.4.35 Start-up Memory Register 2

Please check this register description in the System Firmware chapter

STMEM2 Offset address: $00CC_H$

Start-up Memory Register 2 Reset values see: Table 1660



Field	Bits	Type	Description
MEM0	1:0	rw	**Reserved for BFW usage** Reserved for Boot Firmware usage.
BOOT_ADDR	31:2	rw	**Address of first user-code instruction** Address of the first user-code instruction, taken after SSW Note: BOOT_ADDR is always word-aligned, therefore bits [1:0] are effectively taken as zero.

Table 1660 Reset values of STMEM2

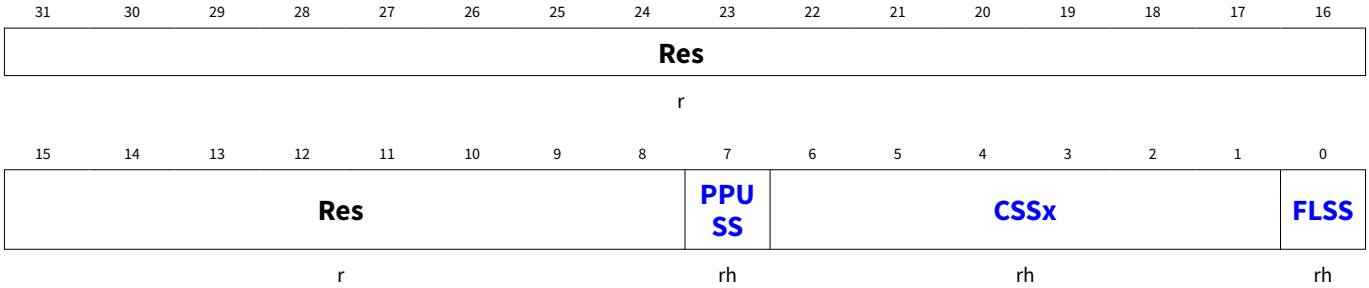
Reset type	Reset value	Note
System Reset	$0000\ 0000_H$	
After Boot-FW Value	$----\ ----_H$	

37.4.36 Shut Down Sequence Control Register D

This register indicates whether CPU, FLASH, PPU and all Bridges have been successfully put into shut down state before reset was issued to ensure that no ongoing RAM write operation was aborted by reset event.The state flags are cleared on a new reset request or on a CLRC register clear event

SHSEQCTRLD Offset address: $00F4_H$

Shut Down Sequence Control Register D Cold PowerOn Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
FLSS	0	rh	**Flash Shutdown State Reached and Acknowledge received** The state of Flash acknowledge after the last warm reset. The bitfield represents the state of the FSI before the last warm reset If this bit is zero after a System Reset (or higher) then it is possible that Flash reliability could have been affected by the reset. ***Note**: Flash is not available on all devices of the AURIX TC4x family. For RRAM devices this bitfield is always 1.* 0_B FSI flash shutdown state not reached before last reset 1_B FSI flash shutdown state reached before last reset
CSSx	6:1	rh	**CPUx Shutdown State Reached and Acknowledge received** The state of CPUx acknowledge after the last warm reset. If any bit is zero after an Application Reset (or higher) then it is possible that the SRAM content could have been corrupted by the reset. For products with fewer CPUs, only the LSBs are active and unused upper bits will always read '1'. 00_H CPU x shutdown state not achieved prior to last reset 01_H CPU x in shutdown state at last reset
PPUSS	7	rh	**PPU Shutdown State Reached and Acknowledge received** The state of PPU acknowledge after the last warm reset. If any bit is zero after an Application Reset (or higher) then it is possible that the SRAM content could have been corrupted by the reset. ***Note**: PPU is not available on all devices of the AURIX TC4x family. In these cases this bitfield is read as 1.* 0_B PPU shutdown state not reached before last reset 1_B PPU shutdown state reached before last reset

37.4.37 Reset Trigger Status Register for Warm Reset Events

After a reset has been executed, the WRMRSTTRIG provide information on all the triggers accumulated for the last reset types after the last CLRC event. Multiple resets triggers may have taken place which is captured in this register.

WRMRSTTRIG — Offset address: $00FC_H$

Reset Trigger Status Register for Warm Reset Events — Cold PowerOn Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	LBTERM	LBPORST	0	0					Res	DWR	CB3	0	CB0	0	Res
r	rh	rh	r	r					r	rh	rh	r	rh	r	r

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STM5	STM4	STM3	STM2	STM1	STM0	SW	SMUSEC	SMUSAFE1SR	SMUSAFE1AR	SMUSAFE0SR	SMUSAFE0AR	0	ESR2	ESR1	ESR0
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	r	rh	rh	rh

Field	Bits	Type	Description
ESR0	0	rh	**Reset Request Trigger Reset Status for ESR0 pin** 0_B This reset trigger has not occurred since the last clear (CLRC) 1_B This reset trigger has occurred since the last clear (CLRC)
ESR1	1	rh	**Reset Request Trigger Reset Status for ESR1 pin** 0_B This reset trigger has not occurred since the last clear (CLRC) 1_B This reset trigger has occurred since the last clear (CLRC)
ESR2	2	rh	**Reset Request Trigger Reset Status for ESR2 pin** 0_B This reset trigger has not occurred since the last clear (CLRC) 1_B This reset trigger has occurred since the last clear (CLRC)
SMUSAFE0AR	4	rh	**Application Reset Request Trigger Reset Status for SMU SAFE0** (See SMU section for SMU trigger sources, including Watchdog Timers) 0_B This reset trigger has not occurred since the last clear (CLRC) 1_B This reset trigger has occurred since the last clear (CLRC)
SMUSAFE0SR	5	rh	**System Reset Request Trigger Reset Status for SMU SAFE0** (See SMU section for SMU trigger sources, including Watchdog Timers) 0_B This reset trigger has not occurred since the last clear (CLRC) 1_B This reset trigger has occurred since the last clear (CLRC)
SMUSAFE1AR	6	rh	**Application Reset Request Trigger Reset Status for SMU SAFE1** (See SMU section for SMU trigger sources, including Watchdog Timers) 0_B This reset trigger has not occurred since the last clear (CLRC) 1_B This reset trigger has occurred since the last clear (CLRC)
SMUSAFE1SR	7	rh	**System Reset Request Trigger Reset Status for SMU SAFE1** (See SMU section for SMU trigger sources, including Watchdog Timers) 0_B This reset trigger has not occurred since the last clear (CLRC) 1_B This reset trigger has occurred since the last clear (CLRC)
SMUSEC	8	rh	**Reset Request Trigger Reset Status for Secure Safety Management Unit** (See SMU section for SMU trigger sources, including Watchdog Timers) 0_B This reset trigger has not occurred since the last clear (CLRC) 1_B This reset trigger has occurred since the last clear (CLRC)

(table continues...)

(continued)

Field	Bits	Type	Description
SW	9	rh	**Reset Request Trigger Reset Status for Software reset request** 0_B This reset trigger has not occurred since the last clear (CLRC) 1_B This reset trigger has occurred since the last clear (CLRC)
STM0	10	rh	**Reset Request Trigger Reset Status for STM0 Compare Match** 0_B This reset trigger has not occurred since the last clear (CLRC) 1_B This reset trigger has occurred since the last clear (CLRC)
STM1	11	rh	**Reset Request Trigger Reset Status for STM1 Compare Match** 0_B This reset trigger has not occurred since the last clear (CLRC) 1_B This reset trigger has occurred since the last clear (CLRC)
STM2	12	rh	**Reset Request Trigger Reset Status for STM2 Compare Match (If Product has STM2)** 0_B This reset trigger has not occurred since the last clear (CLRC) 1_B This reset trigger has occurred since the last clear (CLRC)
STM3	13	rh	**Reset Request Trigger Reset Status for STM3 Compare Match (If Product has STM3)** 0_B This reset trigger has not occurred since the last clear (CLRC) 1_B This reset trigger has occurred since the last clear (CLRC)
STM4	14	rh	**Reset Request Trigger Reset Status for STM4 Compare Match (If Product has STM4)** 0_B This reset trigger has not occurred since the last clear (CLRC) 1_B This reset trigger has occurred since the last clear (CLRC)
STM5	15	rh	**Reset Request Trigger Reset Status for STM5 Compare Match (If Product has STM5)** 0_B This reset trigger has not occurred since the last clear (CLRC) 1_B This reset trigger has occurred since the last clear (CLRC)
CB0	18	rh	**Reset Request Trigger Status for Cerberus System Reset from Debug tool** 0_B This reset trigger has not occurred since the last clear (CLRC) 1_B This reset trigger has occurred since the last clear (CLRC)
CB3	20	rh	**Reset Request Trigger Status for Cerberus Application Reset from Debug tool** 0_B This reset trigger has not occurred since the last clear (CLRC) 1_B This reset trigger has occurred since the last clear (CLRC)
DWR	21	rh	**Reset Request Trigger Reset Status for Double SMU Reset (Double Watchdog Reset)** This bit is set if double SMU resets happen and CLRSMUIRF clear did not happen before the consecutive second SMU reset. This bit shall be reset only with cold power on reset. As the DWR information need to be available after a warm PORST recovery of the permanent reset condition. 0_B This reset trigger has not occurred since the last clear (CLRC) 1_B This reset trigger has occurred since the last clear (CLRC)

(table continues...)

(continued)

Field	Bits	Type	Description
LBPORST	29	rh	**LBIST termination due to PORST** This bitfield indicates if the LBIST was early terminated due to the occurrence of a Power On Reset. If the status of this bitfield is 0, the application must still check the LBTERM to check if the LBIST was terminated properly. This bitfield is latched from PMS whenever a cold PORST happens. 0_B LBIST was not terminated early due to occurence of a PORST pin assertion This trigger has not occurred since the last clear (CLRC) 1_B LBIST early termination due to the occurrence of PORST pin assertion
LBTERM	30	rh	**LBIST was properly terminated** This bitfield indicates if the LBIST was terminated properly. This bitfield is cleared when CLRC is set. When LBIST is terminated and consequently LBTERM bit is set, an internal cold PORST is triggered to SMM by LBIST test unit and not from PMS, therefore the PORST pin is not driven in this case. This bitfield is latched from PMS whenever a cold PORST happens. 0_B LBIST was not terminated properly This trigger has not occurred since the last clear (CLRC) 1_B LBIST was terminated properly
0	3, 17, 19, 27:23, 28, 31	r	**Reserved** Read as 0.

37.4.38 Highest Reset Trigger Status Register for Warm Reset Events

After a reset has been executed, the WRMRSTTRIGH provide information on the reset trigger of the highest reset type among the last reset(s) after the last CLRC event. All the bitfields of this register are cleared when CLRC bit is set.

WRMRSTTRIGH Offset address: 0100_H

Highest Reset Trigger Status Register for Warm Reset Events Cold PowerOn Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	LBTERM	LBPORST	0						Res	DWR	CB3	0	CB0	0	Res
r	rh	rh	r						r	rh	rh	r	rh	r	r

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STM5	STM4	STM3	STM2	STM1	STM0	SW	SMUSEC	SMUSAFE1SR	SMUSAFE1AR	SMUSAFE0SR	SMUSAFE0AR	0	ESR2	ESR1	ESR0
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	r	rh	rh	rh

Field	Bits	Type	Description
ESR0	0	rh	**Reset Request Trigger Reset Status for ESR0 pin** 0_B This reset trigger has not occurred since the last clear (CLRC) 1_B This reset trigger has occurred since the last clear (CLRC)
ESR1	1	rh	**Reset Request Trigger Reset Status for ESR1 pin** 0_B This reset trigger has not occurred since the last clear (CLRC) 1_B This reset trigger has occurred since the last clear (CLRC)
ESR2	2	rh	**Reset Request Trigger Reset Status for ESR2 pin** 0_B This reset trigger has not occurred since the last clear (CLRC) 1_B This reset trigger has occurred since the last clear (CLRC)
SMUSAFE0AR	4	rh	**Application Reset Request Trigger Reset Status for SMU SAFE0** (See SMU section for SMU trigger sources, including Watchdog Timers) 0_B This reset trigger has not occurred since the last clear (CLRC) 1_B This reset trigger has occurred since the last clear (CLRC)
SMUSAFE0SR	5	rh	**System Reset Request Trigger Reset Status for SMU SAFE0** (See SMU section for SMU trigger sources, including Watchdog Timers) 0_B This reset trigger has not occurred since the last clear (CLRC) 1_B This reset trigger has occurred since the last clear (CLRC)
SMUSAFE1AR	6	rh	**Application Reset Request Trigger Reset Status for SMU SAFE1** (See SMU section for SMU trigger sources, including Watchdog Timers) 0_B This reset trigger has not occurred since the last clear (CLRC) 1_B This reset trigger has occurred since the last clear (CLRC)
SMUSAFE1SR	7	rh	**System Reset Request Trigger Reset Status for SMU SAFE1** (See SMU section for SMU trigger sources, including Watchdog Timers) 0_B This reset trigger has not occurred since the last clear (CLRC) 1_B This reset trigger has occurred since the last clear (CLRC)
SMUSEC	8	rh	**Reset Request Trigger Reset Status for Secure Safety Management Unit** (See SMU section for SMU trigger sources, including Watchdog Timers) 0_B This reset trigger has not occurred since the last clear (CLRC) 1_B This reset trigger has occurred since the last clear (CLRC)
SW	9	rh	**Reset Request Trigger Reset Status for Software reset request** 0_B This reset trigger has not occurred since the last clear (CLRC) 1_B This reset trigger has occurred since the last clear (CLRC)
STM0	10	rh	**Reset Request Trigger Reset Status for STM0 Compare Match** 0_B This reset trigger has not occurred since the last clear (CLRC) 1_B This reset trigger has occurred since the last clear (CLRC)
STM1	11	rh	**Reset Request Trigger Reset Status for STM1 Compare Match** 0_B This reset trigger has not occurred since the last clear (CLRC) 1_B This reset trigger has occurred since the last clear (CLRC)

(table continues...)

(continued)

Field	Bits	Type	Description
STM2	12	rh	**Reset Request Trigger Reset Status for STM2 Compare Match (If Product has STM2)** 0_B This reset trigger has not occurred since the last clear (CLRC) 1_B This reset trigger has occurred since the last clear (CLRC)
STM3	13	rh	**Reset Request Trigger Reset Status for STM3 Compare Match (If Product has STM3)** 0_B This reset trigger has not occurred since the last clear (CLRC) 1_B This reset trigger has occurred since the last clear (CLRC)
STM4	14	rh	**Reset Request Trigger Reset Status for STM4 Compare Match (If Product has STM4)** 0_B This reset trigger has not occurred since the last clear (CLRC) 1_B This reset trigger has occurred since the last clear (CLRC)
STM5	15	rh	**Reset Request Trigger Reset Status for STM5 Compare Match (If Product has STM5)** 0_B This reset trigger has not occurred since the last clear (CLRC) 1_B This reset trigger has occurred since the last clear (CLRC)
CB0	18	rh	**Reset Request Trigger Status for Cerberus System Reset from Debug tool** 0_B This reset trigger has not occurred since the last clear (CLRC) 1_B This reset trigger has occurred since the last clear (CLRC)
CB3	20	rh	**Reset Request Trigger Status for Cerberus Application Reset from Debug tool** 0_B This reset trigger has not occurred since the last clear (CLRC) 1_B This reset trigger has occurred since the last clear (CLRC)
DWR	21	rh	**Reset Request Trigger Reset Status for Double SMU Reset (Double Watchdog Reset)** This bit is set if double SMU resets happen and CLRSMUIRF clear did not happen before the consecutive second SMU reset. This bit shall be reset only with cold power on reset. As the DWR information need to be available after a warm PORST recovery of the permanent reset condition. 0_B This reset trigger has not occurred since the last clear (CLRC) 1_B This reset trigger has occurred since the last clear (CLRC)
LBPORST	29	rh	**LBIST termination due to PORST** This bitfield indicates if the LBIST was early terminated due to the occurrence of a Power On Reset. If the status of this bitfield is 0, the application must still check the LBTERM to check if the LBIST was terminated properly. This bitfield is latched from PMS whenever a cold PORST happens. 0_B LBIST was not terminated early due to occurence of a PORST pin assertion 1_B LBIST early termination due to the occurrence of PORST pin assertion

(table continues...)

(continued)

Field	Bits	Type	Description
LBTERM	30	rh	**LBIST was properly terminated** This bitfield indicates if the LBIST was terminated properly. This bitfield is cleared when CLRC is set. This bitfield is latched from PMS whenever a cold PORST happens. 0_B LBIST was not terminated properly 1_B LBIST was terminated properly
0	3, 17, 19, 28:23, 31	r	**Reserved** Read as 0.

37.4.39 Reset Type Status Register

After a reset has been executed, the RSTSTAT provide information on the type of the last reset(s) which happened accumulatively after the last CLRC event. Multiple resets types may have taken place which are captured in this register.

RSTSTAT Offset address: 0104_H

Reset Type Status Register Reset values see: Table 1661

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								TRRST	DBGRST	APPRST	SYSRST	PORST	CLDPORST	PDCLDRST	LVD
r								rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
LVD	0	rh	**Low Voltage Detector Resets** This bitfield indicates if LVDANA reset took place. This bitfield is latched from PMS whenever a cold PORST happens. 0_B This reset type did not appear since the last clear (CLRC) 1_B This reset type appeared since the last clear (CLRC)
PDCLDRST	1	rh	**Power Domain Cold Reset** This bitfield is latched from PMS whenever a cold PORST happens. 0_B This reset type did not appear since the last clear (CLRC) 1_B This reset type appeared since the last clear (CLRC)

(table continues...)

(continued)

Field	Bits	Type	Description
CLDPORST	2	rh	**Cold Power-On Reset** The register is by default reset on cold PORST which is indicated in this bitfield 0_B This reset type did not appear since the last clear (CLRC) 1_B This reset type appeared since the last clear (CLRC)
PORST	3	rh	**Warm Power-On Reset** This bitfield is latched from PMS whenever a cold PORST happens. On a warm PORST, this bitfield is updated by SMM itself. 0_B This reset type did not appear since the last clear (CLRC) 1_B This reset type appeared since the last clear (CLRC)
SYSRST	4	rh	**System Reset** 0_B This reset type did not appear since the last clear (CLRC) 1_B This reset type appeared since the last clear (CLRC)
APPRST	5	rh	**Application Reset** 0_B This reset type did not appear since the last clear (CLRC) 1_B This reset type appeared since the last clear (CLRC)
DBGRST	6	rh	**debug reset** 0_B This reset type did not appear since the last clear (CLRC) 1_B This reset type appeared since the last clear (CLRC)
TRRST	7	rh	**trace reset** 0_B This reset type did not appear since the last clear (CLRC) 1_B This reset type appeared since the last clear (CLRC)
0	31:8	r	**Reserved** Read as 0.

Table 1661 Reset values of RSTSTAT

Reset type	Reset value	Note
Reset	0000 00--$_H$	The value of this register can differ based on the given reset scenario as well as if the register was cleared prior the next reset event.

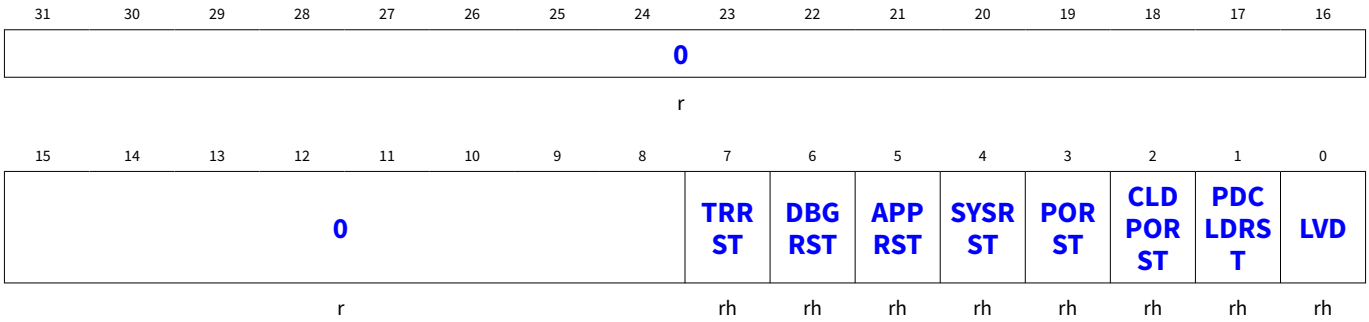
37.4.40 Highest Reset Type Status Register

After a reset has been executed, the RSTSTATH updates information on the highest reset type among the last reset(s) after the last CLRC event. All the bitfields of this register are cleared when CLRC bit is set.

RSTSTATH Offset address: 0108_H

Highest Reset Type Status Register Reset values see: Table 1662

37 System Mode Management (SMM)



Field	Bits	Type	Description
LVD	0	rh	**Low Voltage Detector Resets** This bitfield indicates whether LVDANA reset took place and is latched from PMS whenever a cold PORST happens. 0_B This reset type did not appear since the last clear (CLRC) 1_B This reset type appeared since the last clear (CLRC)
PDCLDRST	1	rh	**Power Domain Cold Reset** This bitfield is latched from PMS whenever a cold PORST happens. 0_B This reset type did not appear since the last clear (CLRC) or a higher reset took place 1_B This reset type appeared since the last clear (CLRC)
CLDPORST	2	rh	**Cold Power-On Reset** The register is by default reset on cold PORST which is indicated in this bitfield 0_B This reset type did not appear since the last clear (CLRC) or a higher reset took place 1_B This reset type appeared since the last clear (CLRC)
PORST	3	rh	**Warm Power-On Reset** This bitfield is latched from PMS whenever a cold PORST happens. Incase of warm PORST, this bitfield is updated by SMM itself. 0_B This reset type did not appear since the last clear (CLRC) or a higher reset took place 1_B This reset type appeared since the last clear (CLRC)
SYSRST	4	rh	**System Reset** 0_B This reset type did not happen during last reset or a higher reset took place 1_B This reset type did happen during last reset. This bitfield is cleared via CLRC bit.
APPRST	5	rh	**Application Reset** 0_B This reset type did not happen during last reset or a higher reset took place 1_B This reset type did happen during last reset. This bitfield is cleared via CLRC bit.

(table continues...)

(continued)

Field	Bits	Type	Description
DBGRST	6	rh	**Debug reset** 0_B This reset type did not happen during last reset or a higher reset took place 1_B This reset type did happen during last reset. This bitfield is cleared via CLRC bit.
TRRST	7	rh	**Trace reset** 0_B This reset type did not happen during last reset or a higher reset took place 1_B This reset type did happen during last reset. This bitfield is cleared via CLRC bit.
0	31:8	r	**Reserved** Read as 0.

Table 1662 Reset values of RSTSTATH

Reset type	Reset value	Note
Reset	0000 00--$_H$	The value of this register can differ based on the given reset scenario as well as if the register was cleared prior the next reset event.

37.4.41 Reset Status Clear Register

RSTSTATCLR Offset address: 0110_H

Reset Status Clear Register Cold PowerOn Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															CLRSMUIRF
r															w

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															CLRC
r															w

Field	Bits	Type	Description
CLRC	0	w	**Clear Reset Status** This bit simultaneously clears the sticky reset status bits which may indicate reset triggers or types. Read as 0. 0_B No effect 1_B Clear reset type and trigger status bits

(table continues...)

(continued)

Field	Bits	Type	Description
CLRSMUIRF	16	w	**Clear Internal SMU Reset Flag** This bit is used to request a clear of the internal flag which indicates whether a previous SMU reset has already been requested and determines the response to a further SMU reset request.Clearing this flag avoid a permanent reset condition in case of consecutive application reset triggered by watchdog timeouts. Read as 0. 0_B No action 1_B Request to clear the internal previous SMU-reset flag
0	15:1, 31:17	r	**Reserved** Read as 0; should be written with 0.

37.4.42 Reset Trigger Control Register A

These registers allow the configuration for the various reset trigger sources to trigger application, system or module group resets.

RSTTRIGCTRLA Offset address: 0114_H

Reset Trigger Control Register A PowerOn Reset value: $0001\ 0402_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PHYRST	PINRST	0			SW			0			SMUSAFE1SR			SMUSAFE1AR	
rw	rw	r			rw			r			rw			rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SMUSAFE1AR	SMUSAFE0SR			SMUSAFE0AR			ESR2			ESR1			ESR0		
rw	rw			rw			rw			rw			rw		

Field	Bits	Type	Description
ESR0	2:0	rw	**ESR0 Reset Trigger Configuration** This bit field defines which reset is generated by a reset request trigger from ESR0 reset. 000_B No reset is generated for a trigger of ESR0 001_B A System Reset is generated for a trigger of ESR0 reset 010_B An Application Reset is generated for a trigger of ESR0 reset 011_B A Module Reset Group Reset will be initiated for Module Reset Group 0 for a trigger of ESR0 100_B A Module Reset Group Reset will be initiated for Module Reset Group 1 for a trigger of ESR0 101_B A Module Reset Group Reset will be initiated for Module Reset Group 2 for a trigger of ESR0 110_B A Module Reset Group Reset will be initiated for Module Reset Group 3 for a trigger of ESR0 111_B Reserved, do not use this combination

(table continues...)

(continued)

Field	Bits	Type	Description
ESR1	5:3	rw	**ESR1 Reset Trigger Configuration** This bit field defines which reset is generated by a reset request trigger from ESR1 reset. 000_B No reset is generated for a trigger of ESR1 001_B A System Reset is generated for a trigger of ESR1 reset 010_B An Application Reset is generated for a trigger of ESR1 reset 011_B A Module Reset Group Reset will be initiated for Module Reset Group 0 for a trigger of ESR1 100_B A Module Reset Group Reset will be initiated for Module Reset Group 1 for a trigger of ESR1 101_B A Module Reset Group Reset will be initiated for Module Reset Group 2 for a trigger of ESR1 110_B A Module Reset Group Reset will be initiated for Module Reset Group 3 for a trigger of ESR1 111_B Reserved, do not use this combination
ESR2	8:6	rw	**ESR2 Reset Trigger Configuration** This bit field defines which reset is generated by a reset request trigger from ESR2 reset. 000_B No reset is generated for a trigger of ESR2 001_B A System Reset is generated for a trigger of ESR2 reset 010_B An Application Reset is generated for a trigger of ESR2 reset 011_B A Module Reset Group Reset will be initiated for Module Reset Group 0 for a trigger of ESR2 100_B A Module Reset Group Reset will be initiated for Module Reset Group 1 for a trigger of ESR2 101_B A Module Reset Group Reset will be initiated for Module Reset Group 2 for a trigger of ESR2 110_B A Module Reset Group Reset will be initiated for Module Reset Group 3 for a trigger of ESR2 111_B Reserved, do not use this combination
SMUSAFE0AR	11:9	rw	**SMU SAFE0 Reset0 Trigger Configuration** This bit field defines which reset is generated by a reset request trigger from SMU_SAFE0 reset. 000_B No reset is generated for an SMU_SAFE0 trigger 0 001_B Reserved, do not use this combination 010_B An Application Reset is generated for a trigger of SMU_SAFE0 reset 0 011_B A Module Reset Group Reset will be initiated for Module Reset Group 0 for a trigger of SMU_SAFE0 reset 0 100_B A Module Reset Group Reset will be initiated for Module Reset Group 1 for a trigger of SMU_SAFE0 reset 0 101_B A Module Reset Group Reset will be initiated for Module Reset Group 2 for a trigger of SMU_SAFE0 reset 0 110_B A Module Reset Group Reset will be initiated for Module Reset Group 3 for a trigger of SMU_SAFE0 reset 0 111_B Reserved, do not use this combination

(table continues...)

(continued)

Field	Bits	Type	Description
SMUSAFE0SR	14:12	rw	**SMU SAFE0 Reset1 Trigger Configuration** This bit field defines which reset is generated by a reset request trigger from SMU_SAFE0 reset. 000_B No reset is generated for an SMU_SAFE0 trigger 1 001_B A System Reset is generated for a trigger of SMU_SAFE0 reset 1 010_B Reserved, do not use this combination 011_B A Module Reset Group Reset will be initiated for Module Reset Group 0 for a trigger of SMU_SAFE0 reset 1 100_B A Module Reset Group Reset will be initiated for Module Reset Group 1 for a trigger of SMU_SAFE0 reset 1 101_B A Module Reset Group Reset will be initiated for Module Reset Group 2 for a trigger of SMU_SAFE0 reset 1 110_B A Module Reset Group Reset will be initiated for Module Reset Group 3 for a trigger of SMU_SAFE0 reset 1 111_B Reserved, do not use this combination
SMUSAFE1AR	17:15	rw	**SMU SAFE1 Reset0 Trigger Configuration** This bit field defines which reset is generated by a reset request trigger from SMU_SAFE1 reset. 000_B No reset is generated for an SMU_SAFE1 trigger 0 001_B Reserved, do not use this combination 010_B An Application Reset is generated for a trigger of SMU_SAFE1 reset 0 011_B A Module Reset Group Reset will be initiated for Module Reset Group 0 for a trigger of SMU_SAFE1 reset 0 100_B A Module Reset Group Reset will be initiated for Module Reset Group 1 for a trigger of SMU_SAFE1 reset 0 101_B A Module Reset Group Reset will be initiated for Module Reset Group 2 for a trigger of SMU_SAFE1 reset 0 110_B A Module Reset Group Reset will be initiated for Module Reset Group 3 for a trigger of SMU_SAFE1 reset 0 111_B Reserved, do not use this combination
SMUSAFE1SR	20:18	rw	**SMU SAFE1 Reset1 Trigger Configuration** This bit field defines which reset is generated by a reset request trigger from SMU_SAFE1 reset. 000_B No reset is generated for an SMU_SAFE1 trigger 1 001_B A System Reset is generated for a trigger of SMU_SAFE1 reset 1 010_B Reserved, do not use this combination 011_B A Module Reset Group Reset will be initiated for Module Reset Group 0 for a trigger of SMU_SAFE1 reset 1 100_B A Module Reset Group Reset will be initiated for Module Reset Group 1 for a trigger of SMU_SAFE1 reset 1 101_B A Module Reset Group Reset will be initiated for Module Reset Group 2 for a trigger of SMU_SAFE1 reset 1 110_B A Module Reset Group Reset will be initiated for Module Reset Group 3 for a trigger of SMU_SAFE1 reset 1 111_B Reserved, do not use this combination

(table continues...)

(continued)

Field	Bits	Type	Description
SW	26:24	rw	**SW Reset Request Trigger Reset Configuration** This bit field defines which reset is generated by a reset request trigger from software reset. 000_B No reset is generated for an SW trigger 001_B A System Reset is generated for a trigger of SW reset 010_B An Application Reset is generated for a trigger of SW reset 011_B A Module Reset Group Reset will be initiated for Module Reset Group 0 for a trigger of SW 100_B A Module Reset Group Reset will be initiated for Module Reset Group 1 for a trigger of SW 101_B A Module Reset Group Reset will be initiated for Module Reset Group 2 for a trigger of SW 110_B A Module Reset Group Reset will be initiated for Module Reset Group 3 for a trigger of SW 111_B Reserved, do not use this combination
PINRST	30	rw	**Pin Reset Configuration** This bit field defines whether the Pins are reset at the beginning of the reset shutdown sequence or at the end of the shutdown sequence. 0_B Pins (Pads) are reset instantly when a reset request occurs 1_B Pins (Pads) are reset at the end of the reset shutdown sequence
PHYRST	31	rw	**Phy Reset Configuration** This bit field defines whether HSPHY0, HSPHY1 and HSPHY2 (if present) are reset at the beginning of the reset sequence or towards the end of the shut down sequence. 0_B HSPHYx are reset instantly when a reset request occurs 1_B HSPHYx is reset at the end of the reset shutdown sequence
0	23:21, 29:27	r	**Reserved** Read as 0; Should be written with 0.

37.4.43 Reset Trigger Control Register B

RSTTRIGCTRLB Offset address: 0118_H

Reset Trigger Control Register B PowerOn Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0														STM5	
r														rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STM5		STM4		STM3			STM2			STM1			STM0		
rw		rw		rw			rw			rw			rw		

Field	Bits	Type	Description
STM0	2:0	rw	**STM0 Reset Request Trigger Reset Configuration** This bit field defines which reset is generated by a reset request trigger from STM0 compare match reset. 000_B No reset is generated for an STM0 trigger 001_B A System Reset is generated for a trigger of STM0 reset 010_B An Application Reset is generated for a trigger of STM0 reset 011_B A Module Reset Group Reset will be initiated for Module Reset Group 0 for a trigger of STM0 100_B A Module Reset Group Reset will be initiated for Module Reset Group 1 for a trigger of STM0 101_B A Module Reset Group Reset will be initiated for Module Reset Group 2 for a trigger of STM0 110_B A Module Reset Group Reset will be initiated for Module Reset Group 3 for a trigger of STM0 111_B Reserved, do not use this combination
STM1	5:3	rw	**STM1 Reset Request Trigger Reset Configuration (If Product has STM1)** This bit field defines which reset is generated by a reset request trigger from STM1 compare match reset. 000_B No reset is generated for an STM1 trigger 001_B A System Reset is generated for a trigger of STM1 reset 010_B An Application Reset is generated for a trigger of STM1 reset 011_B A Module Reset Group Reset will be initiated for Module Reset Group 0 for a trigger of STM1 100_B A Module Reset Group Reset will be initiated for Module Reset Group 1 for a trigger of STM1 101_B A Module Reset Group Reset will be initiated for Module Reset Group 2 for a trigger of STM1 110_B A Module Reset Group Reset will be initiated for Module Reset Group 3 for a trigger of STM1 111_B Reserved, do not use this combination
STM2	8:6	rw	**STM2 Reset Request Trigger Reset Configuration (If Product has STM2)** This bit field defines which reset is generated by a reset request trigger from STM2 compare match reset. 000_B No reset is generated for an STM2 trigger 001_B A System Reset is generated for a trigger of STM2 reset 010_B An Application Reset is generated for a trigger of STM2 reset 011_B A Module Reset Group Reset will be initiated for Module Reset Group 0 for a trigger of STM2 100_B A Module Reset Group Reset will be initiated for Module Reset Group 1 for a trigger of STM2 101_B A Module Reset Group Reset will be initiated for Module Reset Group 2 for a trigger of STM2 110_B A Module Reset Group Reset will be initiated for Module Reset Group 3 for a trigger of STM2 111_B Reserved, do not use this combination

(table continues...)

(continued)

Field	Bits	Type	Description
STM3	11:9	rw	**STM3 Reset Request Trigger Reset Configuration (If Product has STM3)** This bit field defines which reset is generated by a reset request trigger from STM3 compare match reset. 000_B No reset is generated for an STM3 trigger 001_B A System Reset is generated for a trigger of STM3 reset 010_B An Application Reset is generated for a trigger of STM3 reset 011_B A Module Reset Group Reset will be initiated for Module Reset Group 0 for a trigger of STM3 100_B A Module Reset Group Reset will be initiated for Module Reset Group 1 for a trigger of STM3 101_B A Module Reset Group Reset will be initiated for Module Reset Group 2 for a trigger of STM3 110_B A Module Reset Group Reset will be initiated for Module Reset Group 3 for a trigger of STM3 111_B Reserved, do not use this combination
STM4	14:12	rw	**STM4 Reset Request Trigger Reset Configuration (If Product has STM4)** This bit field defines which reset is generated by a reset request trigger from STM4 compare match reset. 000_B No reset is generated for an STM4 trigger 001_B A System Reset is generated for a trigger of STM4 reset 010_B An Application Reset is generated for a trigger of STM4 reset 011_B A Module Reset Group Reset will be initiated for Module Reset Group 0 for a trigger of STM4 100_B A Module Reset Group Reset will be initiated for Module Reset Group 1 for a trigger of STM4 101_B A Module Reset Group Reset will be initiated for Module Reset Group 2 for a trigger of STM4 110_B A Module Reset Group Reset will be initiated for Module Reset Group 3 for a trigger of STM4 111_B Reserved, do not use this combination

(table continues...)

(continued)

Field	Bits	Type	Description
STM5	17:15	rw	**STM5 Reset Request Trigger Reset Configuration (If Product has STM5)** This bit field defines which reset is generated by a reset request trigger from STM5 compare match reset. 000_B No reset is generated for an STM5 trigger 001_B A System Reset is generated for a trigger of STM5 reset 010_B An Application Reset is generated for a trigger of STM5 reset 011_B A Module Reset Group Reset will be initiated for Module Reset Group 0 for a trigger of STM5 100_B A Module Reset Group Reset will be initiated for Module Reset Group 1 for a trigger of STM5 101_B A Module Reset Group Reset will be initiated for Module Reset Group 2 for a trigger of STM5 110_B A Module Reset Group Reset will be initiated for Module Reset Group 3 for a trigger of STM5 111_B Reserved, do not use this combination
0	31:18	r	**Reserved** Read as 0; Should be written with 0.

37.4.44 Reset Trigger Control Register C

These registers allow the configuration for the various reset trigger sources to trigger application, system or module group resets.

RSTTRIGCTRLC Offset address: $011C_H$

Reset Trigger Control Register C PowerOn Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0													SMUSEC		
r													rw		

Field	Bits	Type	Description
SMUSEC	2:0	rw	**SMU SEC Reset Request Trigger Reset Configuration** This bit field defines which reset is generated by a reset request trigger from SMU_SEC reset. 000_B No reset is generated for an SMU_SEC trigger 001_B A System Reset is generated for a trigger of SMU_SEC reset 010_B An Application Reset is generated for a trigger of SMU_SEC reset 011_B A Module Reset Group Reset will be initiated for Module Reset Group 0 for a trigger of SMU_SEC reset 100_B A Module Reset Group Reset will be initiated for Module Reset Group 1 for a trigger of SMU_SEC reset 101_B A Module Reset Group Reset will be initiated for Module Reset Group 2 for a trigger of SMU_SEC reset 110_B A Module Reset Group Reset will be initiated for Module Reset Group 3 for a trigger of SMU_SEC reset 111_B Reserved, do not use this combination.
0	31:3	r	**Reserved** Read as 0; Should be written with 0.

37.4.45 Reset Trigger Register for SMU related Trigger

SMU provides information about the alarm group and the alarm id for the most recent SMU reset request. This way it can be identified which alarm triggered the last SMU related reset.

This register provides information of the last SMU reset alarm source. The register is cleared via CLRSMUIRF register bit

SMURSTSTATA Offset address: 0120_H

Reset Trigger Register for SMU related Trigger Cold PowerOn Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0												SMU0ALMID1			
r												rh			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SMU0ALMID1	SMU0ALMGRP1					SMU0ALMID0					SMU0ALMGRP0				
rh	rh					rh					rh				

Field	Bits	Type	Description
SMU0ALMGRP0	4:0	rh	**Reset Request Trigger Status for Safety related Trigger Group0 of SAFE_SMU0** This bit field defines which safety group triggered the last reset request
SMU0ALMID0	9:5	rh	**Reset Request Trigger Status for Safety related Trigger ID0 of SAFE_SMU0** This bit field defines which safety alarm triggered the last reset request

(table continues...)

(continued)

Field	Bits	Type	Description
SMU0ALMGRP1	14:10	rh	**Reset Request Trigger Status for Safety related Trigger Group1 of SAFE_SMU0** This bit field defines which safety group triggered the last reset request.
SMU0ALMID1	19:15	rh	**Reset Request Trigger Status for Safety related Trigger ID1 of SAFE_SMU0** This bit field defines which safety alarm triggered the last reset request
0	31:20	r	**Reserved** Read as 0.

37.4.46 Reset Trigger Register for SMU related Trigger

SMU provides information about the alarm group and the alarm id for the most recent SMU reset request. This way it can be identified which alarm triggered the last SMU related reset.

This register provides information of the last SMU reset alarm source. The register is cleared via CLRSMUIRF register bit

SMURSTSTATB Offset address: 0124_H

Reset Trigger Register for SMU related Trigger Cold PowerOn Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0				SMUSECALMID					SMUSECALMGRP			SMU1ALRMID1			
r				rh					rh			rh			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SMU1ALRMID1	SMU1ALRMGRP1					SMU1ALMID0					SMU1ALMGRP0				
rh	rh					rh					rh				

Field	Bits	Type	Description
SMU1ALMGRP0	4:0	rh	**Reset Request Trigger Status for Safety related Trigger Group0 of SAFE_SMU1** This bit field defines which safety group triggered the last reset request
SMU1ALMID0	9:5	rh	**Reset Request Trigger Status for Safety related Trigger ID0 of SAFE_SMU1** This bit field defines which safety alarm triggered the last reset request
SMU1ALRMGRP1	14:10	rh	**Reset Request Trigger Status for Safety related Trigger Group1 of SAFE_SMU1** This bit field defines which safety group triggered the last reset request
SMU1ALRMID1	19:15	rh	**Reset Request Trigger Status for Safety related Trigger Group of SAFE_SMU1** This bit field defines which safety alarm triggered the last reset request

(table continues...)

(continued)

Field	Bits	Type	Description
SMUSECALMGRP	22:20	rh	**Reset Request Trigger Status for Seucrity related Trigger Group of SEC_SMU** This bit field defines which security group triggered the last reset request
SMUSECALMID	27:23	rh	**Reset Request Trigger Status for Seucrity related Trigger Alarm of SEC_SMU** This bit field defines which security alarm triggered the last reset request
0	31:28	r	**Reserved** Read as 0.

37.4.47 User Info Storage Register

The user can store intermediate information in this register not affected by warm resets.

USRINFO Offset address: 0128_H

User Info Storage Register Cold PowerOn Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
USRINFO															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
USRINFO															
rw															

Field	Bits	Type	Description
USRINFO	31:0	rw	**User Information**

37.4.48 Pad Disable Control Register

The pad structure of the TC4Dx GPIO lines offers the possibility to disable pad. This feature can be controlled by individual bits in the pad disable control register PDISC.

PDISC Offset address: $012C_H$

Pad Disable Control Register System Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0														PDIS1	PDIS0
r														rwh	rwh

Field	Bits	Type	Description
PDISx (x=0-1)	x	rwh	**Pad Disable for ESR Pin x** This bit disables the pad. 0_B Pad Px is enabled 1_B Pad Px is disabled
0	31:2	r	**Reserved** Read as 0; should be written with 0.

37.4.49 ESR0 Reset Output Configuration Register

ESR0CFG Offset address: 0130_H

ESR0 Reset Output Configuration Register System Reset value: $0000\ 0001_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

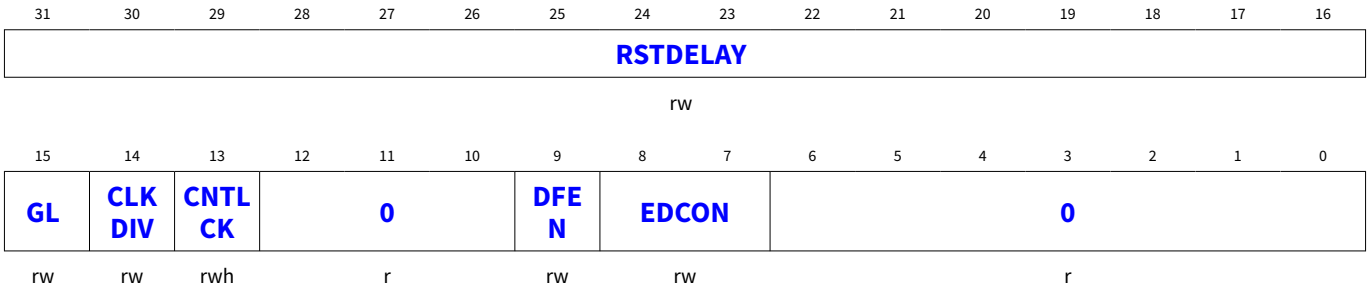
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0														ARC	ARI
r														w	rh

Field	Bits	Type	Description
ARI	0	rh	**Application ESR0 Reset Output Indicator** This bit is set when an Application Reset request trigger occurs and cleared by writing to ARC. When the ARI bit is set and an ESR pin is configured as a reset output, the corresponding ESR input will not re-trigger a reset. This prevents feedback of the reset indication causing a new reset request. ***Note**: Observed reset value after boot will depend upon ARI mode.* 0_B No application reset trigger detected (since last clear) 1_B Application reset trigger detected (since last clear)
ARC	1	w	**Application ESR0 Reset Output Indicator Clear** Read as 0 0_B No effect 1_B Clear Application Reset Indicator (ARI)
0	31:2	r	**Reserved** Read as 0; should be written with 0.

37.4.50 ESRx Input Configuration Register

ESRCFGx (x=0-2) Offset address: 0134_H+x*4

ESRx Input Configuration Register PowerOn Reset value: $0000\ A300_H$



Field	Bits	Type	Description
EDCON	8:7	rw	**Edge Detection Control** This bit field defines the edges that lead to an ESRx trigger of the synchronous path. 00_B No trigger is generated 01_B A trigger is generated upon a rising edge 10_B A trigger is generated upon a falling edge 11_B A trigger is generated upon a rising OR falling edge
DFEN	9	rw	**ESR Digital Filter Enable** This bit activates digital spike filter (majority filter of 3 consecutive values) during normal RUN mode. 0_B The filter is bypassed 1_B The filter is used
CNTLCK	13	rwh	**Counter configuration Lock** CNTLCK provides lock mechanism for RSTDELAY, GL and CLKDIV bitfields. CNTLCK is read and if it is 0, RSTDELAY, GL and CLKDIV bitfields are updated after which CNTLCK is set to 1 by software. The hardware updates all the values synchronously and sets CNTLCK back to 0. CNTLCK reset value is 1 to take ESR default configuration at reset but a later read by user would be 0 as hardware set the bit back. Following is the SW sequence to update these bitfields: - Wait until CNTLCK is 0 (required to check about any previous update is on-going) - Configure the RSTDELAY, GL and CLKDIV bitfields and set CNTLCK to 1 This would trigger the HW to process the new configured bitfield values, upon the completion of HW process the CNTLCK is automatically rest to 0 by HW. 0_B The bitfields GL, CLKDIV, RSTDELAY can be updated 1_B The bitfields GL, CLKDIV, RSTDELAY is locked by user software and hardware update is going on
CLKDIV	14	rw	**Backup Clock Divider - 1024** 0_B Counter is clocked by fBACK 1_B Counter is clocked by fBACK/1024
GL	15	rw	**Greater than or Less than Counter Event Selection** 0_B The event is triggered when the captured pulse is less than the RSTDELAY count value 1_B The event is triggered when the captured pulse is greater than the RSTDELAY count value

(table continues...)

(continued)

Field	Bits	Type	Description
RSTDELAY	31:16	rw	**Reset input pulse width compare value**
0	6:0, 12:10	r	**Reserved** Read as 0; should be written with 0.

37.4.51 ESR0 Reset Counter Control Register

This register controls the ESR0 strong pull down reset elongation time after Application reset release. This register controls the reset length setting for Application reset.

ESR0CNTCTRL Offset address: 0140_H

ESR0 Reset Counter Control Register Reset values see: Table 1663

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0						RSTCON		0				CLKDIV			
r						rw		r				rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				RSTECNT											
r				rw											

Field	Bits	Type	Description
RSTECNT	11:0	rw	**Reload Value for Reset and ESR Output Elongation Counter** On reset release, ESR0 output pin and or reset release is elongated for RSTECNT time duration. The reset release is delayed by the configured RSTECNT value depending on RSTCON bitfield
CLKDIV	19:16	rw	**Clock divider for 100MHz base clock** RSTECNT base clock = 100MHz / 2^CLKDIV
RSTCON	25:24	rw	**Reset Release and ESR0 Reset Output Elongation Behaviour** This bitfield configures the behaviour of both ESR0 output elongation and reset release based on counter value. 00_B ESR0 Reset output is deasserted immediately on reset release 01_B Both ESR0 Reset output and reset release are elongated by RSTECNT time after which both are released 10_B ESR0 Reset output is elongated by RSTECNT time but reset is released immediately. 11_B Reset release is elongated by RSTECNT time but ESR0 output is released by user software when ESR0CFG.ARC bit is released.
0	15:12, 23:20, 31:26	r	**Reserved** Read as 0; Should be written with 0.

Table 1663 Reset values of ESR0CNTCTRL

Reset type	Reset value	Note
Cold PowerOn Reset	$0208\ 030E_H$	Value on reset release.

(table continues...)

Table 1663 (continued) Reset values of ESR0CNTCTRL

Reset type	Reset value	Note
After Boot-FW Value	XXXX XX-- XXXX ---- XXXX ---- ---- ----$_B$	Value installed by Boot Firmware.

37.4.52 ESR0 GPIO Control Register

GPIO level control

If aPin was configured in DRVCFG to be a general purpose Input or Output this register may be used by Application Software to request a pin level change.

Note: When the Set and Clear Bits are set in the same write cycle to 1, the pin level request will toggle to the inverted value.

GPIOESR0 Offset address: 0144$_H$

ESR0 GPIO Control Register System Reset value: 0000 0000$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0							IN	0				CLR	SET	0	OUT
r							rh	r				w	w	r	rh

Field	Bits	Type	Description
OUT	0	rh	**Output value** The port output field reports the requested value of a GPIO pin when it is selected by DRVCFGESR0 as output. A 0 in GPIOESR0.OUT bit equals a low level requested at the ESR0 pin. A high level corresponds to the bit read as 1. Note that the bits GPIOESR0.OUT can be individually set or cleared by writing appropriate values into GPIOESR0.SET and GPIOESR0.CLR. 0_B Set the input level of ESR0 to the low state. 1_B Set the level of ESR0 to the high state.
SET	2	w	**Set** Writing a 1 to the SET field will set the Pin to a Voltage level equivalent to the High state. Read as 0.
CLR	3	w	**Clear** Writing a 1 to the CLR field will set the Pin to a Voltage level equivalent to the Low state. Read as 0.
IN	8	rh	**Read back Pin value** Reading the IN field always returns the current logical value at the pin independently from the pin direction or if the pin was set to be driven by GPIO, ALTSEL or HW_SEL. 0_B The input level of ESR0 is 0. 1_B The input level of ESR0 is 1.

(table continues...)

(continued)

Field	Bits	Type	Description
0	1, 7:4, 31:9	r	**Reserved** Read as 0; should be written with 0

37.4.53 ESR0 Pad configuration Control Register

Driver Configuration

The Pin Driver Configuration register contains the bit fields needed to set up the required pin configuration such as signal source and driver type selection.

DRVCFGESR0 Offset address: 0148_H

ESR0 Pad configuration Control Register System Reset value: $0000\ 2083_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	PL			0	PD			MODE				0		OD	DIR
r	rwh			r	rwh			rwh				r		rwh	rwh

Field	Bits	Type	Description
DIR	0	rwh	**Pad Signal direction** The DIR field defines the direction of the Pad in GPIO mode 0_B **IN**: Pad confligured as Input 1_B **OUT**: Pad confligured as Output
OD	1	rwh	**Output Buffer configuration** The bitfield OD defines the Output driver configuration when the Pin is configured in output mode. The Pad may be configured to Open Drain or Push/Pull driver types. 0_B **PP**: Push Pull 1_B **OD**: Open Drain

(table continues...)

(continued)

Field	Bits	Type	Description
MODE	7:4	rwh	**Input configuration or Alternative Output signal selection** If the Pad is configured as Input, the MODE field determinates the Input mode regarding pure tristate mode versus an connected pull device. Note: If the direction is set to input only the lower 2 bits of the Mode field will be evaluated. If the Pad is configured as Output, the MODE field selects the number of the alternative output function driving the value towards the pin number n. Values higher than 0 do select the alternative output function matching to the absolute number of the value (e.g. MODE=1 selects alternative function number 1) When set to 0 the Pin acts as GPIO port. There the Pin state is SW controllable via the register GPIOESR0 0_H **GPIO**: GPIO Pad configured in Tristate input mode or GPIO output mode 1_H **ALT01**: ALT01_PD Pad connected to input pull-down device 2_H **ALT02**: ALT02_PU Pad connected to input pull-up device 3_H **ALT03**: ALT03_TRI Pad connected to input tristate mode 4_H **ALT04**: reserved … 6_H **ALT06**: reserved 7_H **ALT07**: Weak Pull Down active with Application reset 8_H **ALT08**: Weak Pull Up active with Application reset. 9_H **ALT09**: TRISTATE with Application reset. A_H **ALT10**: reserved … F_H **ALT15**: reserved
PD	10:8	rwh	**Pad drive mode** Pin drive strength selection 000_B **SPEED0**: Speedgrade 0 … 111_B **SPEED7**: Speedgrade 7

(table continues...)

(continued)

Field	Bits	Type	Description
PL	14:12	rwh	**Pad level selection** **PLx Input Level control** 000_B **Const_0**: Automotive level "AL" Automotive level "AL" 001_B Automotive level "AL" Automotive level "AL" 010_B TTL level for 5V pad supply. Degraded TTL level used for CIF when pad supply is 3.3V. TTL level for 5V pad supply. Degraded TTL level used for CIF when pad supply is 3.3V. 011_B TTL level for 3.3V pad supply. TTL level for 3.3V pad supply. 100_B Automotive level "AL" Automotive level "AL" 101_B Automotive level "AL" Automotive level "AL" 110_B TTL level for 5V pad supply. Degraded TTL level used for CIF when pad supply is 3.3V. TTL level for 5V pad supply. Degraded TTL level used for CIF when pad supply is 3.3V. 111_B TTL level for 3.3V pad supply. TTL level for 3.3V pad supply.
0	3:2, 11, 31:15	r	**Reserved** Read as 0; should be written with 0

37.4.54 ESR1 GPIO Control Register

GPIO level control

If aPin was configured in DRVCFG to be a general purpose Input or Output this register may be used by Application Software to request a pin level change.

Note: When the Set and Clear Bits are set in the same write cycle to 1, the pin level request will toggle to the inverted value.

GPIOESR1 Offset address: $014C_H$

ESR1 GPIO Control Register System Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0							IN	0				CLR	SET	0	OUT
r							rh	r				w	w	r	rh

Field	Bits	Type	Description
OUT	0	rh	**Output value** The port output field reports the requested value of a GPIO pin when it is selected by DRVCFGESR1 as output. A 0 in GPIOESR1.OUT bit equals a low level requested at the ESR1 pin. A high level corresponds to the bit read as 1. Note that the bit GPIOESR1.OUT can be individually set or cleared by writing appropriate values into GPIOESR1.SET and GPIOESR1.CLR. 0_B Set the input level of ESR1 to the low state. 1_B Set the level of ESR1 to the high state.
SET	2	w	**Set** Writing a 1 to the SET field will set the Pin to a Voltage level equivalent to the High state. Read as 0.
CLR	3	w	**Clear** Writing a 1 to the CLR field will set the Pin to a Voltage level equivalent to the Low state. Read as 0.
IN	8	rh	**Read back Pin value** Reading the IN field always returns the current logical value at the pin independently from the pin direction or if the pin was set to be driven by GPIO, ALTSEL or HW_SEL. 0_B The input level of ESR1 is 0. 1_B The input level of ESR1 is 1.
0	1, 7:4, 31:9	r	**Reserved** Read as 0; should be written with 0

37.4.55 ESR1 Pad configuration Control Register

Driver Configuration

DRVCFGESR1 Offset address: 0150_H

ESR1 Pad configuration Control Register System Reset value: $0000\ 2020_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	PL			0	PD			MODE				0		OD	DIR
r	rwh			r	rwh			rwh				r		rwh	rwh

Field	Bits	Type	Description
DIR	0	rwh	**Pad Signal direction** The DIR field defines the direction of the Port Pad in GPIO mode 0_B **IN**: Pad configured as Input 1_B **OUT**: Pad configured as Output

(table continues...)

(continued)

Field	Bits	Type	Description
OD	1	rwh	**Output Buffer configuration** The bitfield OD defines the Output driver configuration when the Pin is configured in output mode. The Pad may be configured to Open Drain or Push/Pull driver types. 0_B **PP**: Push Pull 1_B **OD**: Open Drain
MODE	7:4	rwh	**Input configuration or Alternative Output signal selection** If the Pad is configured as Input, the MODE field determinates the Input mode regarding pure tristate mode versus an connected pull device. Note: If the direction is set to input only the lower 2 bits of the Mode field will be evaluated. If the Pad is configured as Output, the MODE field selects the number of the alternative output function driving the value towards the pin number n. Values higher than 0 do select the alternative output function matching to the absolute number of the value (e.g. MODE=1 selects alternative function number 1) When set to 0 the Pin acts as GPIO port. There the Pin state is SW controllable via the register GPIOESR1 0_H **GPIO**: GPIO Pad configured in Tristate input mode or GPIO output mode 1_H **ALT01**: ALT01_PD Pad connected to input pull-down device 2_H **ALT02**: ALT02_PU Pad connected to input pull-up device 3_H **ALT03**: ALT03_TRI Pad connected to input tristate mode 4_H **ALT04**: reserved … F_H **ALT15**: reserved
PD	10:8	rwh	**Pad drive mode** Pin drive strength selection 000_B **SPEED0**: Speedgrade 0 … 111_B **SPEED7**: Speedgrade 7

(table continues...)

(continued)

Field	Bits	Type	Description
PL	14:12	rwh	**Pad level selection** **PLx Input Level control** 000_B **Const_0**: Automotive level "AL" Automotive level "AL" 001_B Automotive level "AL" Automotive level "AL" 010_B TTL level for 5V pad supply. Degraded TTL level used for CIF when pad supply is 3.3V. TTL level for 5V pad supply. Degraded TTL level used for CIF when pad supply is 3.3V. 011_B TTL level for 3.3V pad supply. TTL level for 3.3V pad supply. 100_B Automotive level "AL" Automotive level "AL" 101_B Automotive level "AL" Automotive level "AL" 110_B TTL level for 5V pad supply. Degraded TTL level used for CIF when pad supply is 3.3V. TTL level for 5V pad supply. Degraded TTL level used for CIF when pad supply is 3.3V. 111_B TTL level for 3.3V pad supply. TTL level for 3.3V pad supply.
0	3:2, 11, 31:15	r	**Reserved** Read as 0; should be written with 0

37.4.56 PROT Initialisation Configuration Register

PROTINIT Offset address: 0154_H

PROT Initialisation Configuration Register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															PINIT_DONE
r															rw1s

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															
r															

Field	Bits	Type	Description
PINIT_DONE	16	rw1s	**INIT_DONE signal for locking unused PROTs** Setting this bit signals to all connected PROT state machines to change into RunLock state 0_B **INIT**: Initialization Phase 1_B **DONE**: Initialization Done
0	15:0, 31:17	r	**Reserved** Read as 0; should be written with 0.

37.5 Debug information

This section is not applicable for the module being described.

37.6 References

This section is not applicable for the module being described.

37.7 SMM revision history

Reference	Description of change(s)
Date range: 2024-08-17 to 2025-01-29	
Start-up configuration	• Removed information about HWCFG latching in SMM status register STSTAT throughout the whole SMM chapter
External service request interface (ESRx)	• Changed the wakeup connectivity of the ESRx pads and the standby entry connectivity for ESR2
Module kernel reset	• Added AUDIO IP to the overview table
Low frequency applications	• Added a new section about low frequency applications
System Mode Management (SMM)	Changed "CPU, PPU" to "processing units"

37.8 TC4Dx SMM information

37.8.1 TC4Dx SMM configuration

There are no deviations from the generic specification.

37.8.2 TC4Dx SMM features

There are no deviations from the generic specification.

37.8.3 TC4Dx SMM functional description

37.8.3.1 HWCFG latching on start-up

On a warm reset release, HWCFG port pin inputs are latched on the rising edge of an application reset and stored in register STSTAT.

It is recommended that the boot configuration is selected using the internal BMI configuration and not from pins owing to security concerns. Boot from HWCFG [3,4,5] is to be for test purposes and end of lineprogram ming functions and is not intended for productive usage. The STSTAT status register indication hasdepende nce on Port reset behavior and Software reset configurations. Comparison of latched HWCFG valuereflected in STSTAT register against the latched HWCFG value in PMS.PADSTAT register can be under all conditions ensured only if hardware configuration pins are pulled externally via resistors.

37.8.3.2 CANXL Module kernel reset

Table 1664 TC4Dx considerations for module resets

Module	Module reset capability
CANXL	Module reset implemented in CANXL ***Note**: TC4Dx CANXL Module reset requires special handling as the application software must ensure that X CAN node's Message Handler and Protocol Controller are stopped, before Kernel reset request via KRST or via Module Group Reset trigger (when enabled through MODULE_RST_CTRLA.GRSTEN bit-field)*

Related information

Module kernel reset on page 6354

37.8.4 TC4Dx SMM registers

There are no deviations from the generic specification.

37.8.4.1 Register address space - SMM

Table 1665 Registers address space - SMM

Module	Base address	End address	Note
SMM	$F0060000_H$	$F0063FFF_H$	SMM FPI Slave Interface

37.8.4.2 Register overview - access mode glossary

Table 1666 Register overview - access mode glossary

Keyword	Description
CE	Access protection using PROT register SMM_PROTCE .
CSE	Access protection using PROT register SMM_PROTCSE .
E	Access protection using PROT register SMM_PROTE .
ECPUn (n=0-5)	Access protection using PROT register SMM_PROTCPUn .
SE	Access protection using PROT register SMM_PROTSE .
APU-P	Protection group consisting of registers SMM_ACCEN_WRA , SMM_ACCEN_WRB , SMM_ACCEN_RDA , SMM_ACCEN_RDB , SMM_ACCEN_VM , SMM_ACCEN_PRS .
P	Access protection using *APU-P* registers.
APU-PCS	Protection group consisting of registers SMM_ACCENCS_WRA , SMM_ACCENCS_WRB , SMM_ACCENCS_RDA , SMM_ACCENCS_RDB , SMM_ACCENCS_VM , SMM_ACCENCS_PRS .
PCS	Access protection using *APU-PCS* registers.
BE	Always returns a Bus Error.
U	No access restrictions.
PROT	Access restrictions as defined in the PROT register access rules.
STP	Access only during start-up, before start-up protection is set (SMM_STCON.ST_PROT=1).

37.8.4.3 Registers overview - SMM (ascending offset address)

Table 1667 Registers overview - SMM (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
SMM_ID	Module Identification Register	0008_H	P	BE	PowerOn Reset	6371
SMM_PROTE	PROT Register Endinit	$000C_H$	U	SV, PROT	Application Reset	6371
SMM_PROTSE	PROT Register Safe Endinit	0010_H	U	SV, PROT	Application Reset	6373
SMM_PROTCSE	PROT Register Cyber-Secure Endinit	0014_H	U	SV, PROT	Application Reset	6375

(table continues...)

Table 1667 (continued) Registers overview - SMM (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
SMM_ACCEN_WRA	Write access enable register A	0018_H	U	SE, SV	Application Reset	6377
SMM_ACCEN_WRB	Write access enable register B	$001C_H$	U	SE, SV	Application Reset	6377
SMM_ACCEN_RDA	Read access enable register A	0020_H	U	SE, SV	Application Reset	6378
SMM_ACCEN_RDB	Read access enable register B	0024_H	U	SE, SV	Application Reset	6378
SMM_ACCEN_VM	VM access enable register	0028_H	U	SE, SV	Application Reset	6379
SMM_ACCEN_PRS	PRS access enable register	$002C_H$	U	SE, SV	Application Reset	6379
SMM_ACCENCS_WRA	ACCENCS write access enable register A	0038_H	U	CSE, SV	Application Reset	6380
SMM_ACCENCS_WRB	ACCENCS write access enable register B	$003C_H$	U	CSE, SV	Application Reset	6380
SMM_ACCENCS_RDA	ACCENCS read access enable register A	0040_H	U	CSE, SV	Application Reset	6381
SMM_ACCENCS_RDB	ACCENCS read access enable register B	0044_H	U	CSE, SV	Application Reset	6381
SMM_ACCENCS_VM	ACCENCS VM access enable register	0048_H	U	CSE, SV	Application Reset	6382
SMM_ACCENCS_PRS	ACCENCS PRS access enable register	$004C_H$	U	CSE, SV	Application Reset	6383
SMM_PROTCE	PROT Register Cyber-Secure Endinit	0058_H	U	SV, PROT	Application Reset	6383
SMM_PROTCPUn (n=0-5)	PROT Register for PMCSRn Idle control	$005C_H$+n *4	U	SV, PROT	Application Reset	6385
SMM_PMCSRn (n=0-5)	Power Management Control and Status Register for CPUn	0074_H+n *4	U	(SV, SE)| (ECPUn, SV)	Application Reset	6387
SMM_PMCSRCS	Power Management Control and Status Register for CPUcs in CSRM subystem	$008C_H$	U	CE, SV, PCS	Application Reset	6387
SMM_PMSWCR1	Standby and Wake-up Control Register 1	0094_H	U	SV, E, P	Cold PowerOn Reset	6389
SMM_PMTRCSR0	Power Management Transition Control and Status Register 0	0098_H	U	SV, E, P	Cold PowerOn Reset	6391
SMM_PMTRCSR1	Power Management Transition Control and Status Register 1	$009C_H$	U	SV, E, P	Cold PowerOn Reset	6394

(table continues...)

Table 1667 (continued) Registers overview - SMM (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
SMM_PMTRCSR2	Power Management Transition Control and Status Register 2	$00A0_H$	U	SV, E, P	Cold PowerOn Reset	6394
SMM_PMTRCSR3	Power Management Transition Control and Status Register 3	$00A4_H$	U	SV, E, P	Cold PowerOn Reset	6396
SMM_PMSTAT0	Power Management Status Register 0	$00AC_H$	U	BE	6397	6397
SMM_DRPCTRL	Power Management Droop Control and Status Register	$00B0_H$	U	SV, E, P	Cold PowerOn Reset	6400
SMM_STSTAT	Start-Up Status Register	$00B4_H$	U	BE	See 6444	6444
SMM_SYSCON	System Control Register	$00B8_H$	P	P, SV, E	System Reset	6402
SMM_STCON	Start-up Configuration Register	$00BC_H$	P	STP, P	6403	6403
SMM_SWRSTCON	Software Reset Control Register	$00C0_H$	P	SV, E, P	System Reset	6404
SMM_STMEM1	Start-up Memory Register 1	$00C8_H$	P	STP, P	6404	6404
SMM_STMEM2	Start-up Memory Register 2	$00CC_H$	P	STP, P	6407	6407
SMM_SHSEQCTRLD	Shut Down Sequence Control Register D	$00F4_H$	U	BE	Cold PowerOn Reset	6407
SMM_WRMRSTTRIG	Reset Trigger Status Register for Warm Reset Events	$00FC_H$	U	BE	Cold PowerOn Reset	6408
SMM_WRMRSTTRIGH	Highest Reset Trigger Status Register for Warm Reset Events	0100_H	U	BE	Cold PowerOn Reset	6411
SMM_RSTSTAT	Reset Type Status Register	0104_H	U	BE	6414	6414
SMM_RSTSTATH	Highest Reset Type Status Register	0108_H	U	BE	6415	6415
SMM_RSTSTATCLR	Reset Status Clear Register	0110_H	P	SV, E, P	Cold PowerOn Reset	6417
SMM_RSTTRIGCTRLA	Reset Trigger Control Register A	0114_H	P	SV, E, P	PowerOn Reset	6418
SMM_RSTTRIGCTRLB	Reset Trigger Control Register B	0118_H	P	SV, E, P	PowerOn Reset	6421
SMM_RSTTRIGCTRLC	Reset Trigger Control Register C	$011C_H$	PCS	SV, CE, PCS	PowerOn Reset	6424
SMM_SMURSTSTATA	Reset Trigger Register for SMU related Trigger	0120_H	P	BE	Cold PowerOn Reset	6425
SMM_SMURSTSTATB	Reset Trigger Register for SMU related Trigger	0124_H	P	BE	Cold PowerOn Reset	6426

(table continues...)

Table 1667 (continued) Registers overview - SMM (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
SMM_USRINFO	User Info Storage Register	0128_H	P	SV, E, P	Cold PowerOn Reset	6427
SMM_PDISC	Pad Disable Control Register	$012C_H$	P	SV, E, P	System Reset	6427
SMM_ESR0CFG	ESR0 Reset Output Configuration Register	0130_H	P	SV, E, P	System Reset	6428
SMM_ESRCFGx (x=0-2)	ESRx Input Configuration Register	0134_H+x *4	P	SV, E, P	PowerOn Reset	6428
SMM_ESR0CNTCTRL	ESR0 Reset Counter Control Register	0140_H	P	SV, E, P	6430	6430
SMM_GPIOESR0	ESR0 GPIO Control Register	0144_H	U	P, E, SV	System Reset	6431
SMM_DRVCFGESR0	ESR0 Pad configuration Control Register	0148_H	P	E, SV, P	System Reset	6432
SMM_GPIOESR1	ESR1 GPIO Control Register	$014C_H$	U	P, E, SV	System Reset	6434
SMM_DRVCFGESR1	ESR1 Pad configuration Control Register	0150_H	P	E, SV, P	System Reset	6435
SMM_PROTINIT	PROT Initialisation Configuration Register	0154_H	U	SV, E, P	Application Reset	6437

37.8.4.4 Start-Up Status Register

SMM_STSTAT Offset address: $00B4_H$

Start-Up Status Register Reset values see: Table 1668

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res				0		MODE2	RAMINT	Res	0	Res	SPDEN	TRSTL	Res		
r				r		rh	rh	r	r	r	rh	rh	r		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MODE1	Res							HWCFG							
rh	r							rh							

Field	Bits	Type	Description
HWCFG	7:0	rh	**Hardware Configuration Setting** This bit field contains the HWCFG latched value used by the boot firmware. STSTAT register latches HWCFG pin values with a delay of multiple clock cycles after reset release on every application reset. HWCFG[7:0] = [Res, P14.4, P10.6, P10.5, P14.3, P14.2, P14.5, Res] ***Note**: The observed reset value after boot depends upon the state of the HWCFG pins*
MODE1	15	rh	**MODE** This bit indicates if the Test Mode is entered or not. Testmode is entered if both MODE1=0 AND MODE2=0 together. 0_B A Test Mode can be selected 1_B Normal Mode is selected
TRSTL	19	rh	**TRSTL Status** This bit simply displays the value of TRSTL.
SPDEN	20	rh	**Single Pin DAP Mode Enable** 0_B Single Pin DAP Mode is disabled 1_B Single Pin DAP Mode is enabled
RAMINT	24	rh	**RAM Content Security Integrity** In normal operation this bit can be set or cleared by the application (via SYSCON.RAMINTM). If a test boot mode is entered, the bit is automatically cleared (and cannot be set again in test mode) because the content may have been altered ***Note**: This bit is reset only by a cold power-on reset.* 0_B RAM Security Integrity cannot be guaranteed 1_B RAM Security Integrity maintained
MODE2	25	rh	**MODE** This bit indicates if the Test Mode is entered or not. Testmode is entered if both MODE1=0 AND MODE2=0. 0_B A Test Mode can be selected 1_B Normal Mode is selected
0	22, 27:26	r	**Reserved** Read as 0.

Table 1668 Reset values of SMM_STSTAT

Reset type	Reset value	Note
After Boot-FW Value	---- 0010 -X-0 --00 1000 0000 ---- ----$_B$	Bit-fields with unknown values depend on fuse or reset value.

37.8.5 TC4Dx SMM connectivity

The table below lists interfaces to and from this functional block to other blocks in the device.

Table 1669 List of SMM interface signals

Interface signals	I/O	Description
SMM_IR_SRC_SMM0	Out	Interrupt Request
SMM_IR_SRC_SMM1	Out	Interrupt Request
PMS_SMM_RSTSTATy.z	In	Reset Status Registers to indicate Reset types and triggers
PMS_SMM_ColdPORST	In	Cold PORST request from PMS VMONP
PMS_SMM_WarmPORST	In	Warm PORST request from PORST pin
SMU_SMM_RESET[3:0]_REQ	In	Reset Request from SMU_SAFE0/1
SMU_SMM_CS_RESET_REQ	In	Reset Request from SMUCS
CPU0_SMM_STM_RST_REQ	In	Reset Request from STM
CPU1_SMM_STM_RST_REQ	In	Reset Request from STM
CPU2_SMM_STM_RST_REQ	In	Reset Request from STM
CPU3_SMM_STM_RST_REQ	In	Reset Request from STM
CPU4_SMM_STM_RST_REQ	In	Reset Request from STM
CPU5_SMM_STM_RST_REQ	In	Reset Request from STM
SMM_FB_ColdPowerOnReset	Out	Cold Reset Output to all connected Functional Blocks
SMM_FB_PowerOnReset	Out	Warm PORST Reset Output to all connected Functional Blocks
SMM_FB_SystemReset	Out	System Reset Output to all connected Functional Blocks
SMM_FB_ApplicationReset	Out	Application Reset Output to all connected Functional Blocks
SMM_FB_ModuleGroupReset[0:3]	Out	Module Group Reset Output to all connected Functional Blocks
SMM_DEBUG_DebugReset	Out	Debug Reset Output
PORTS_SMM_ESR2_PORT_IN	In	ESR2 input to SMM
SMM_SCU_ESR0	Out	ESR0 input to SCU ERU
SMM_SCU_ESR1	Out	ESR1 input to SCU ERU
SMM_SCU_ESR2	Out	ESR2 input to SCU ERU
SMM_PMS_VDROOP	Out	Voltage Droop request to PMS EVRC
SMM_PMS_REQSLP_STANDBY	Out	Standby Request
SMM_CPU0_REQSLP_IDLE	Out	Idle Request to CPU
SMM_CPU1_REQSLP_IDLE	Out	Idle Request to CPU
SMM_CPU2_REQSLP_IDLE	Out	Idle Request to CPU
SMM_CPU3_REQSLP_IDLE	Out	Idle Request to CPU

(table continues...)

Table 1669 (continued) List of SMM interface signals

Interface signals	I/O	Description
SMM_CPU4_REQSLP_IDLE	Out	Idle Request to CPU
SMM_CPU5_REQSLP_IDLE	Out	Idle Request to CPU
SMM_CPUCS_REQSLP_IDLE	Out	Idle Request to CPU
SMM_PPU_REQSLP_IDLE	Out	Idle Request to PPU
WTU_SMM_WAKEUP_CPU0	In	Wake-up request for CPU0 from WTU
WTU_SMM_WAKEUP_CPU1	In	Wake-up request for CPU1 from WTU
WTU_SMM_WAKEUP_CPU2	In	Wake-up request for CPU2 from WTU
WTU_SMM_WAKEUP_CPU3	In	Wake-up request for CPU3 from WTU
WTU_SMM_WAKEUP_CPU4	In	Wake-up request for CPU4 from WTU
WTU_SMM_WAKEUP_CPU5	In	Wake-up request for CPU5 from WTU
WTU_SMM_WAKEUP_CS	In	Wake-up request for CSRM-core from WTU
SMM_PPU_reset_req	Out	Shutdown request
PPU_SMM_ppu_safestate	In	PPU safe state indication
SMM_PPU_shutdown_addr	Out	Shutdown address for PPU
PPU_SMM_core_status	In	System mode indication from PPU
SMM_SCU_ESRxTRAPREQ	Out	ESR trap request x=[2:0]
SMU_SMM_CS_RESET_ALMGRP[2:0]	In	Alarm group information of alarm triggering the corresponding CS_RESET_REQ
SMU_SMM_CS_RESET_ALMID[4:0]	In	Alarm index information of alarm triggering the corresponding CS_RESET_REQ
SMU_SMM_RESET[3:0]_ALMGRP[4:0]	In	Alarm group information of alarm triggering the corresponding RESET_REQ
SMU_SMM_RESET[3:0]_ALMID[4:0]	In	Alarm index information of alarm triggering the corresponding RESET_REQ
DFT_SMM_LBIST_reset	In	Reset request at LBIST termination

37.8.6 TC4Dx SMM revision history

Reference	Description of change(s)
Date range: 2024-08-17 to 2025-01-29	
HWCFG latching on start-up	• Added the description about HWCFG latching in STSTAT

38 Voltage and Temperature Monitors (VTMON)

The VTMON module interfaces to distributed temperature monitors at different locations across the microcontroller.

38.1 Feature list

Die temperature sensor (DTS)

- Continuous Measurement of the actual silicon temperature by distributed integrated sensors reflecting the actual junction temperature and variation across the die
- Integrated Die Temperature Sensor (DTS) with an overall accuracy of measurement within ±3°C after trimming
- Multiple DTS instances placed at proximity of hot spots to cover spatial temperature variance (dT/ds inclusive hot spots), temporal temperature variations (dT/dt) within measurement interval, and application use-case variations
- The DTS shall generate SMU alarms when the upper or lower configurable temperature limits representing the operating temperature range of the application are violated
- Thermal shut down alarm which can trigger a reset through the SMU (Safety Management Unit) to detect cases of thermal runaway beyond the maximum allowed junction temperature
- Redundant temperature sensing units in different RUN and STANDBY power domains to ensure minimization of common cause faults

38.2 Functional overview

The VTMON provides access to the temperature measurement functions.

The building blocks are:

- Die Temperature Sensors (DTS)
- UBS-Serial bridge to all DTS instances

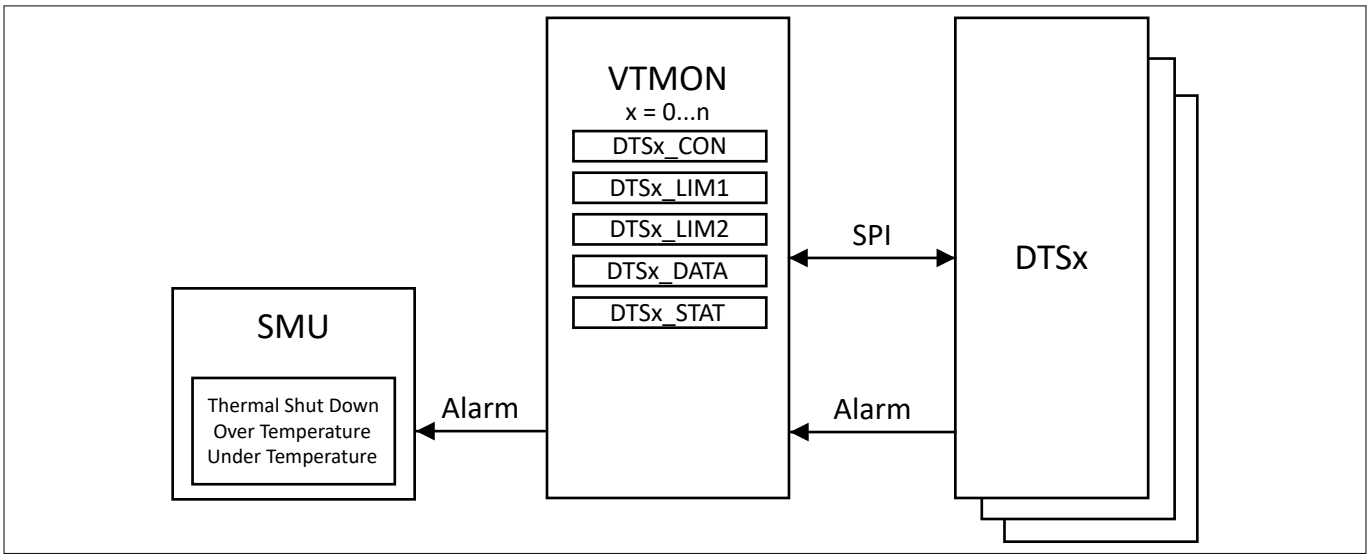


Figure 836 VTMON block diagram

For the alarm description please refer to the alarm mapping tables in the SMU functional block and the Safety Manual.

Related information

TC4Dx SMU alarm mapping tables on page 7233

38.3 Functional description

The VTMON connects to various DTS instances located at different module locations across the die.

Distributed DTS instances

The DTS sensors are instanced predominantly at hot spot areas like processing units. There is also one DTS instance in the Power Management System (PMS) to support temperature measurement during standby modes. The spatial separation of the sensors and instantiation in different power and reset domains allow redundancy from the safety point of view.

Table 1670 DTS instances and domains

DTS instances	Supply domain	Clock	Reset	SMU alarms
DTSPMS	VDDEVRSB	DTS clock: $\left(\frac{fBACK}{50}\right)$ MHz Interface clock: $\left(\frac{fBACK}{5}\right)$ MHz	LVDPMS1 reset	High, Low, Thermal shutdown
DTS0 (SRI0)	VDDEXTOSC		Application reset	High, Low, Thermal shutdown
DTS1 (SRI1)				High, Low, Thermal shutdown
DTS2 (SRI2)				High, Low, Thermal shutdown
DTS3				High, Low, Thermal shutdown
DTS4				High, Low, Thermal shutdown

38.3.1 VTMON serial interface

The VTMON interface is a serial interface connecting to various DTS located at different module locations across the die.

VTMON interface

The VTMON interface serves as a bridge that provides SPB/FPI access to the DTS sensor instances. To mitigate high routing overheads, all communication between the VTMON and the distributed sensor instances are serialized, with the VTMON serving as the serial master to convey read/write operations from the SPB / FPI to the distributed sensor instances. The serial interfaces are clocked by divided back-up clock for DTS instances ($\frac{100\ MHz}{5}$) The DTS module itself runs at a lower clock of $\left(\frac{fBACK}{5}\right)$ MHz .

All configuration registers come with a set of shadow registers located in the distributed sensor instances. The LCK bit in the configuration registers is used to indicate an ongoing serial transfer to the shadow register in the distributed sensor instance. It shall be ensured that the configuration register should not be written when the respective LCK bit is set. Data transfer from shadow configuration registers in the distributed sensor instances to the VTMON configuration registers can be triggered by setting the REQ bitfield. The VALID flag indicates the completion of the data transfer.

38.3.2 DTS measurement function

The Die Temperature Sensor (DTS) indicates directly the current junction temperature.

Each DTS instance periodically generates a measurement result that can be converted into the current die temperature in the region it is located. In the event of thermal limit violations, alarms can be generated to support thermal performance throttling operations. Each DTS operates within the temperature range of TSR, with an accuracy of (TNL+TCALACC), with corresponding parameters as documented in the datasheet. The

temperature sensing resolution is 0.22 °C/code. The maximum conversion time is the t_M parameter documented in the datasheet. Gain and offsets errors are compensated by trimming executed by start-up software. Systematic non-linearity is linearized through digital post-processing. Differential temperature monitoring between DTS instances correlated to application use case based thermal simulations can minimize the thermal margin required and allow tighter monitoring of thermal boundaries.

The analog circuitry for each DTS is enabled by setting the EN register bit. When the START register bit is set, temperature measurements will be carried out continuously during RUN/SLEEP modes, with the latest measurement result being updated in the RESULT bitfield. Both the EN and START can be set concurrently, though the first conversion will take slightly longer for the analog circuitry to be properly initialized. The DTSREADY register bit is set when the analog circuitry has finished the first conversion and only then the alarms are forwarded.

The DTSPMS registers are not affected by a warm PORST, system or application reset; consequently PMS_DTS_STAT temperature result from earlier conversion is available for immediate use after any warm reset as it is reset only on an LVD reset. All the other DTS instances in VTMON are reset on an application reset.

The DTS result RESULT can be converted to temperature by the following transfer function.

$$T[^oC] = \frac{\text{DTSx_STAT.RESULT}}{\text{gnom}} - 273.15$$ where the *gnom* is 4.781 Codes/K

A peak temperature detector records the highest temperature measurement result in PEAK since the last reset operation. This can be reset by writing a ‘1’ to the PEAKDETRST register field.

Upper and lower thermal warnings with adjustable threshold, hysteresis and filters are included and are targeted for thermal throttling and control functions. In addition, a thermal-shutdown warning is included to trigger a reset during thermal runaway scenarios. All warnings can be triggered only when the WRNEN register is set. WRNEN is configured so that warnings are disabled after start-up.

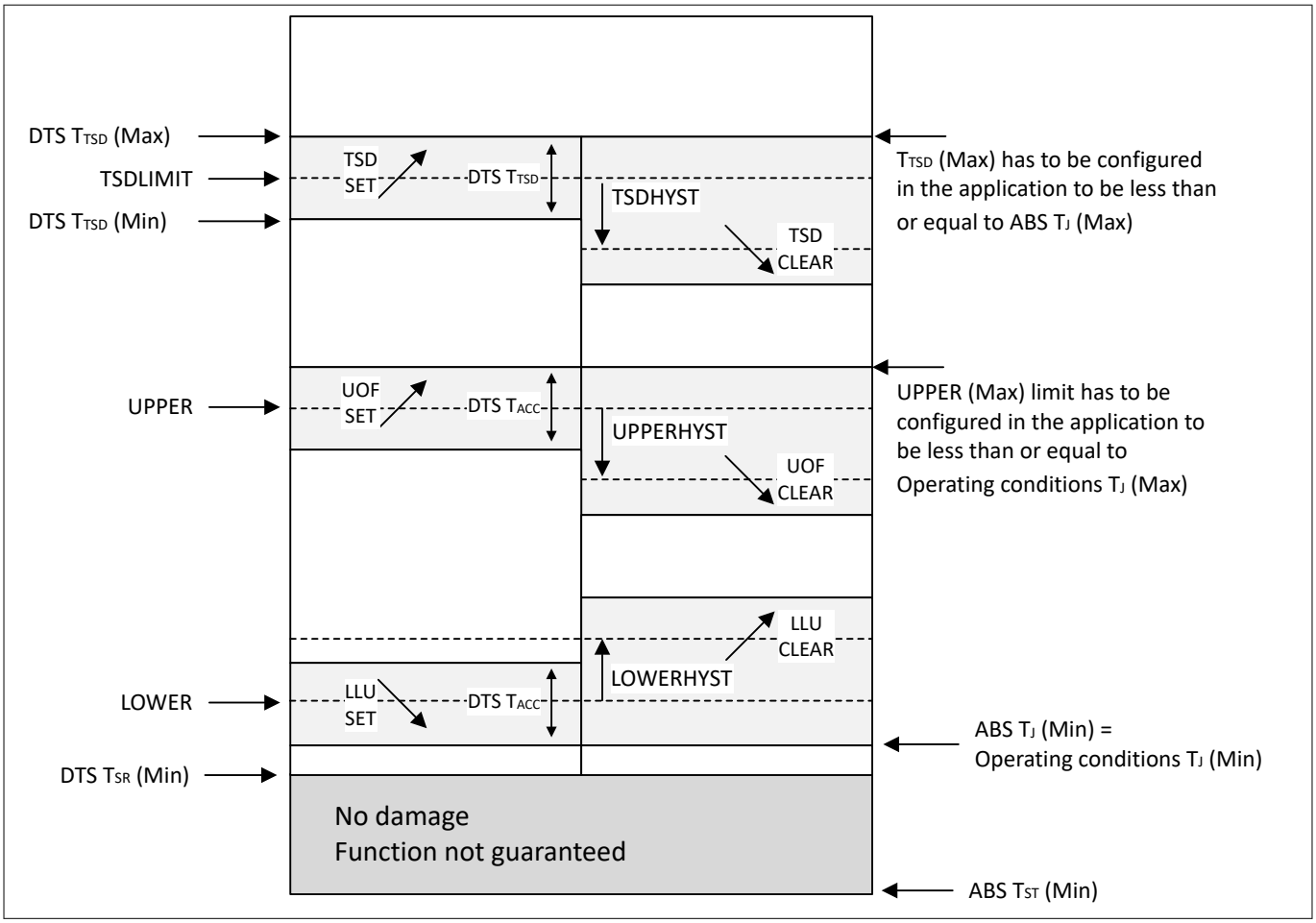


Figure 837 DTS warning generation

The functionality of all warnings are summarized in Table below. Die temperature thermal shutdown, upper and lower limits are configured in TSDLIMIT, UPPER and LOWER register bits respectively. On violation of these limits, TSD, UOF and LLU status bits are set. After start-up, the DTS limits have to be re-configured appropriately depending on the application before alarm reactions from SMU or CSRM are activated. Only when a new DTS conversion result is available, the DTS comparators are consequently triggered to check the actual RESULT against the thermal shutdown, upper and lower limits.

The set criteria for all warnings also allows implementation of a filter whereby the warning will be triggered only when the set limits are satisfied for a number of consecutive temperature measurement violations. The TSD warning is triggered when RESULT exceeds TSDLIMIT for (TSDFILT + 1) consecutive conversions. The UOF warning is triggered when RESULT exceeds UPPER for (UPPERFILT + 1) consecutive conversions. The LLU warning is triggered when RESULT is below LOWER for (LOWERFILT + 1) consecutive conversions.

All warnings are cleared when the corresponding clear criteria are satisfied, which includes an adjustable hysteresis in the clear limits. The clear criteria for TSD warning is satisfied when RESULT is below (TSDLIMIT - TSDHYST). The clear criteria for UOF warning is satisfied when RESULT is below (UPPER - UPPERHYST). Similarly, the clear criteria for LLU warning is satisfied when RESULT exceeds (LOWER + LOWERHYST).

When a warning is triggered, the associated alarm status bits (TSDALRM for thermal shutdown, UOFALRM for over temperature and LLUALRM for under temperature warnings) are set, together with the corresponding alarm being forwarded to the SMU and the CSRM. All alarms status bits are cleared by the user writing a '1' to the CLR bitfield. Unlike warnings, the alarm triggering functions are immediate and not subject to the serial transfer delay.

Table 1671 Alarms, warnings and triggers

Warning type	Warning signal (associated alarm)	Warning and alarm set limits	Warning and alarm set criteria	Warning and alarm clear limits	Warning and alarm clear criteria
Thermal shutdown warning	TSD (TSDALRM)	≥TSDLIMIT	Consecutive (TSDFILT+1) DTS conversions	< TSDLIMIT – TSDHYST	Immediate
Thermal upper warning	UOF (UOFALRM)	≥ UPPER	Consecutive (UPPERFILT+1) DTS conversions	< UPPER – UPPERHYST	Immediate
Thermal lower warning	LLU (LLUALRM)	≤ LOWER	Consecutive (LOWERFILT+1) DTS conversions	> LOWER + LOWERHYST	Immediate

38.4 Registers

38.4.1 Register overview - access mode glossary

Table 1672 Register overview - access mode glossary

Keyword	Description
E	Access protection using PROT register PROTE.
SE	Access protection using PROT register PROTSE.
APU-P	Protection group consisting of registers ACCEN_WRA, ACCEN_WRB, ACCEN_RDA, ACCEN_RDB, ACCEN_VM, ACCEN_PRS.
P	Access protection using *APU-P* registers.
BE	Always returns a Bus Error.
U	No access restrictions.
PROT	Access restrictions as defined in the PROT register access rules.

38.4.2 Registers overview - VTMON (ascending offset address)

Table 1673 Registers overview - VTMON (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CLC	Clock Control Register	000_H	P	P, SV, E	Application Reset	6455
OCS	OCDS Control and Status Register	004_H	P	SV, P	Debug Reset	6456
ID	Module Identification Register	008_H	P	BE	PowerOn Reset	6457
RST_CTRLA	Reset Control Register A	$00C_H$	P	P, SV, E	Application Reset	6457
RST_CTRLB	Reset Control Register B	010_H	P	P, SV, E	Application Reset	6458
RST_STAT	Reset Status Register	014_H	P	BE	Application Reset	6459
PROTE	PROT Register Endinit	020_H	U	SV, PROT	Application Reset	6460
PROTSE	PROT Register Safe Endinit	024_H	U	SV, PROT	Application Reset	6461
ACCEN_WRA	Write access enable register A	040_H	U	SE, SV	Application Reset	6463
ACCEN_WRB	Write access enable register B	044_H	U	SE, SV	Application Reset	6464
ACCEN_RDA	Read access enable register A	048_H	U	SE, SV	Application Reset	6464

(table continues...)

Table 1673 (continued) Registers overview - VTMON (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ACCEN_RDB	Read access enable register B	$04C_H$	U	SE, SV	Application Reset	6465
ACCEN_VM	VM access enable register	050_H	U	SE, SV	Application Reset	6465
ACCEN_PRS	PRS access enable register	054_H	U	SE, SV	Application Reset	6466
DTSx_CON	Die Temperature Sensor x Control Register	060_H+x* 20_H	U	E, P, SV	Kernel Reset	6466
DTSx_LIM1	Die Temperature Sensor x Limit Register 1	064_H+x* 20_H	U	E, P, SV	Kernel Reset	6468
DTSx_LIM2	Die Temperature Sensor x Limit Register 2	068_H+x* 20_H	U	E, P, SV	Kernel Reset	6469
DTSx_DATA	Die Temperature Sensor x Data Register	$06C_H$+x* 20_H	U	E, P, SV	Kernel Reset	6470
DTSx_STAT	Die Temperature Sensor x Status Register	070_H+x* 20_H	U	BE	Kernel Reset	6472

38.4.3 Clock Control Register

The Clock Control Register CLC allows the programmer to adapt the functionality and power consumption of the module to the requirements of the application.

Register CLC controls the module clock signal and the reactivity to the sleep signal.

CLC Offset address: 000_H

Clock Control Register Application Reset value: $0000\ 0003_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												EDIS	0	DISS	DISR
r												rw	r	rh	rw

Field	Bits	Type	Description
DISR	0	rw	**Module Disable Request Bit** Used for enable/disable control of the module. 0_B On request: enable the module clock 1_B Off request: stop the module clock
DISS	1	rh	**Module Disable Status Bit** 0_B Module clock is enabled 1_B Off: module is not clocked

(table continues...)

(continued)

Field	Bits	Type	Description
EDIS	3	rw	**Sleep Mode Enable Control** Used to control the module's reaction to sleep mode. 0_B Sleep mode request is enabled and functional 1_B Module disregards the sleep mode control signal
0	2, 31:4	r	**Reserved** Read as 0; should be written with 0.

38.4.4 OCDS Control and Status Register

The OCDS Control and Status register OCS controls the debug behavior by selecting suspend modes. When OCDS is disabled the suspend control is ineffective.

OCS Offset address: 004_H

OCDS Control and Status Register Debug Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		SUSSTA	SUS_P	SUS				0							
r		rh	w	rw				r							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												TG_P	TGB	0	
r												w	rw	r	

Field	Bits	Type	Description
TGB	2	rw	**OTGB0/1 Bus Select** 0_B Trigger Set is output on OTGB0 1_B Trigger Set is output on OTGB1
TG_P	3	w	**TGS, TGB Write Protection** TGS and TGB are only written when TG_P is 1, otherwise unchanged. Read as 0.
SUS	27:24	rw	**OCDS Suspend Control** Controls the sensitivity to the suspend signal coming from the OCDS Trigger Switch (OTGS) Not listed combinations lead to soft suspend mode. 0_H Will not suspend 1_H Hard suspend Clocks will be disabled immediately 2_H Soft suspend mode Clock will be disabled after kernel (VTMON) acknowledge.
SUS_P	28	w	**SUS Write Protection** SUS is only written when SUS_P is 1, otherwise unchanged. Read as 0.

(table continues...)

(continued)

Field	Bits	Type	Description
SUSSTA	29	rh	**Suspend State** 0_B Module is not (yet) suspended 1_B Module is suspended
0	1:0, 23:4, 31:30	r	**Reserved** Read as 0; should be written with 0.

Table 1674 Access mode restrictions of OCS sorted by descending priority

Mode name	Access mode		Description
write 1 to .**TG_P**	rw	TGB	Set TG_P during write access
write 1 to .**SUS_P**	rw	SUS	Set SUS_P during write access
(default)	r	SUS, TGB	

38.4.5 Module Identification Register

ID Offset address: 008_H

Module Identification Register PowerOn Reset value: $00C5\ C013_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MOD_NUM															
r															

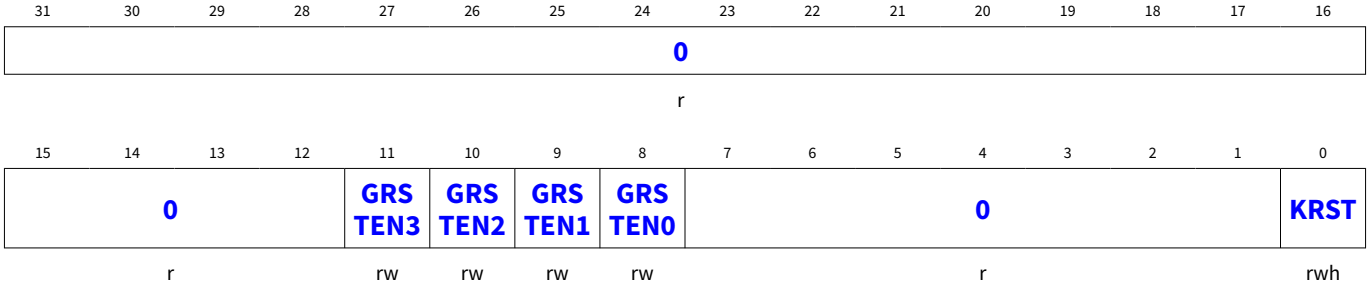
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MOD_TYPE								MOD_REV							
r								r							

Field	Bits	Type	Description
MOD_REV	7:0	r	**Module Revision** Indicates the revision number of the implementation.
MOD_TYPE	15:8	r	**Module Type** This internal marker is fixed to $C0_H$.
MOD_NUM	31:16	r	**Module Number** Indicates the module identification number (example: $00C5_H$ = SARADC).

38.4.6 Reset Control Register A

RST_CTRLA Offset address: $00C_H$

Reset Control Register A Application Reset value: $0000\ 0000_H$

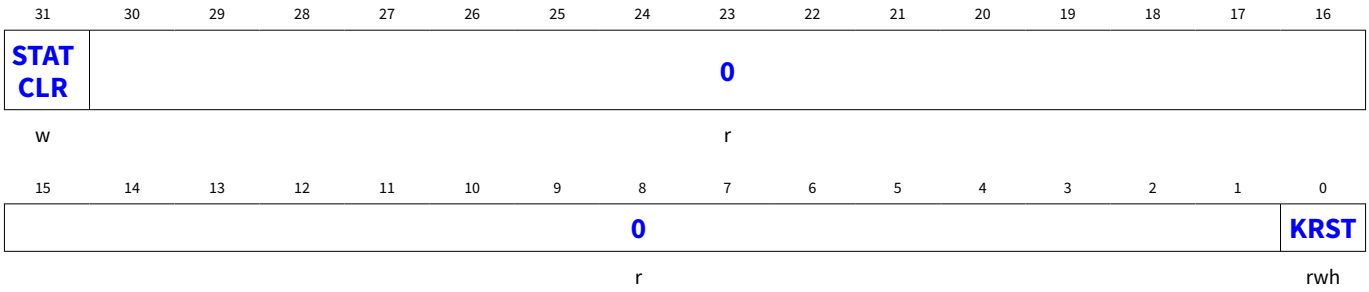


Field	Bits	Type	Description
KRST	0	rwh	**Kernel Reset** Request a kernel reset. The requested reset is executed if CTRLA.KRST=1 and CTRLB.KRST=1. KRST is cleared after the kernel reset was executed. Kernel reset shall not be used for VTMON module as this leads to the loss of trimming values which cannot be installed by the user. 0_B No action 1_B A kernel reset was requested
GRSTENx (x=0-3)	x+8	rw	**Enable for Global Module Reset Group x** 0_B Global module reset group x does not have any effect 1_B Global module reset group x results in a kernel reset
0	7:1, 31:12	r	**Reserved** Read as 0; should be written with 0.

38.4.7 Reset Control Register B

RST_CTRLB Offset address: 010_H

Reset Control Register B Application Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
KRST	0	rwh	**Kernel Reset** Request a kernel reset. The requested reset is executed if CTRLA.KRST=1 and CTRLB.KRST=1. KRST is cleared after the kernel reset was executed. Kernel reset shall not be used for VTMON module as this leads to the loss of trimming values which cannot be installed by the user. 0_B No action 1_B A kernel reset was requested
STATCLR	31	w	**Kernel Reset Status Clear** Clears all status bits in RST_STAT when 1 is written. Read will return 0. 0_B No action 1_B Write with ´1´ clears bits STAT.GRSTx and bit STAT.KRST .
0	30:1	r	**Reserved** Read as 0; should be written with 0.

38.4.8 Reset Status Register

The reset status register contains the status bits for kernel reset (KRST) and the global module reset groups (GRSTx).

RST_STAT Offset address: 014_H

Reset Status Register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				GRST3	GRST2	GRST1	GRST0	0							KRST
r				rh	rh	rh	rh	r							rh

Field	Bits	Type	Description
KRST	0	rh	**Kernel Reset Status** Indicates an executed kernel reset. RST_STAT.KRST is set after the execution of a kernel reset in the same clock cycle in which the reset bits are cleared. Clear KRST by setting bit STATCLR in register RST_CTRLB.
GRSTx (x=0-3)	x+8	rh	**Status for Global Module Reset Group x** 0_B Reset was not triggered by Global Reset Group x 1_B Reset was triggered by Global Reset Group x
0	7:1, 31:12	r	**Reserved** Read as 0.

38.4.9 PROT Register Endinit

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner or CS master to update the PROT state as described in the PROT mechanism.

The PROTE register controls lock / unlock of the local Endinit (E) protected control registers.

PROTE Offset address: 020_H

PROT Register Endinit Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OWEN	ODEF	TAGID						PRSEN	PRS			VMEN	VM		
w	rw	rw						rw	rw			rw	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												SWEN	STATE		
r												w	rwh		

Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
VM	18:16	rw	**Virtual Machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual Machine definition Enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection Set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.

(table continues...)

(continued)

Field	Bits	Type	Description
PRSEN	23	rw	**Protection Set definition Enable for PROT owner** 0_B **Disable**: PRS is not part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; should be written with 0.

Table 1675 Access mode restrictions of PROTE sorted by descending priority

Mode name	Access mode		Description
PROT_OWNER and write 1 to .**SWEN**	rwh	STATE	PROT owner can modify the PROT state (for allowed and valid transitions)
PROT_OWNER and write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	PROT owner can modify the existing PROT owner
SECURE_MASTER and write 1 to .**SWEN**	rwh	STATE	Secure master can modify the PROT state (for allowed and valid transitions)
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

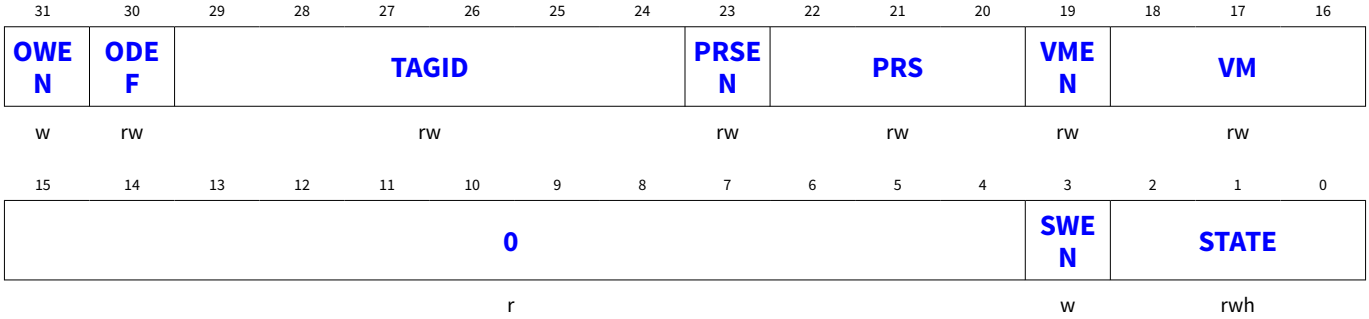
38.4.10 PROT Register Safe Endinit

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner or CS master to update the PROT state as described in the PROT mechanism.

The PROTSE register controls lock / unlock of the local Safe Endinit (SE) protected control registers.

PROTSE Offset address: 024_H

PROT Register Safe Endinit Application Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
VM	18:16	rw	**Virtual Machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual Machine definition Enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection Set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.
PRSEN	23	rw	**Protection Set definition Enable for PROT owner** 0_B **Disable**: PRS is not part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner

(table continues...)

(continued)

Field	Bits	Type	Description
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; should be written with 0.

Table 1676 Access mode restrictions of PROTSE sorted by descending priority

Mode name	Access mode		Description
PROT_OWNER and write 1 to .**SWEN**	rwh	STATE	PROT owner can modify the PROT state (for allowed and valid transitions)
PROT_OWNER and write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	PROT owner can modify the existing PROT owner
SECURE_MASTER and write 1 to .**SWEN**	rwh	STATE	Secure master can modify the PROT state (for allowed and valid transitions)
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

38.4.11 Write access enable register A

This register defines which master TAG ID encodings are enabled or disabled for write accesses to the access protected region.

ACCEN_WRA Offset address: 040_H

Write access enable register A Application Reset value: $1000\ 0003_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Write access enable for Master TAG ID q** This bit enables write access to the access protected region for transactions with the Master TAG ID q 0_B Disabled for write access 1_B Enabled for write access

38.4.12 Write access enable register B

This register defines which master TAG ID encodings are enabled or disabled for write accesses to the access protected region.

ACCEN_WRB Offset address: 044_H

Write access enable register B Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Write access enable for Master TAG ID q** This bit enables write access to the access protected region for transactions with the Master TAG ID q 0_B Disabled for write access 1_B Enabled for write access
0	31:8	r	**Reserved** Read as 0; should be written with 0.

38.4.13 Read access enable register A

This register defines which master TAG ID encodings are enabled or disabled for read accesses to the access protected region.

ACCEN_RDA Offset address: 048_H

Read access enable register A Application Reset value: $FFFF\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Read access enable for Master TAG ID q** This bit enables read access to the access protected region for transactions with the Master TAG ID q 0_B Disabled for read access 1_B Enabled for read access

38.4.14 Read access enable register B

This register defines which master TAG ID encodings are enabled or disabled for read accesses to the access protected region.

ACCEN_RDB Offset address: $04C_H$

Read access enable register B Application Reset value: $0000\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Read access enable for Master TAG ID q** This bit enables read access to the access protected region for transactions with the Master TAG ID q 0_B Disabled for read access 1_B Enabled for read access
0	31:8	r	**Reserved** Read as 0; should be written with 0.

38.4.15 VM access enable register

This register defines which virtual machine encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each VMx.

ACCEN_VM Offset address: 050_H

VM access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for VM ID q** This bit enables read access to the access protected region for transactions with the VM ID q. 0_B Disabled for read access 1_B Enabled for read access

(table continues...)

(continued)

Field	Bits	Type	Description
WRq (q=00-07)	q+16	rw	**Write access enable for VM ID q** This bit enables write access to the access protected region for transactions with the VM ID q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

38.4.16 PRS access enable register

This register defines which protection register set encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each PRSx.

ACCEN_PRS Offset address: 054_H

PRS access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

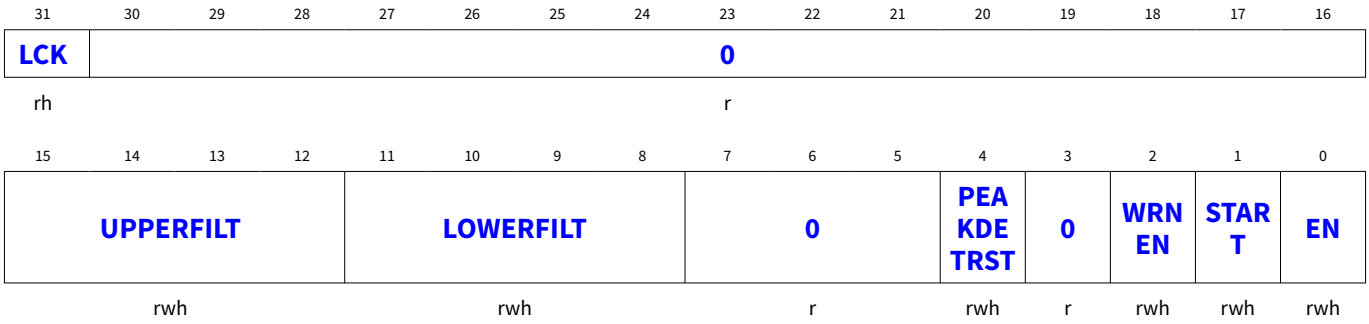
Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for PRS q** This bit enables read access to the access protected region for transactions with the PRS q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for PRS q** This bit enables write access to the access protected region for transactions with the PRS q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

38.4.17 Die Temperature Sensor x Control Register

DTSx_CON (x=0-4) Offset address: 060_H+x*20_H

Die Temperature Sensor x Control Register Kernel Reset value: $0000\ 0000_H$

38 Voltage and Temperature Monitors (VTMON)



Field	Bits	Type	Description
EN	0	rwh	**DTS enable** This field is used to enable the DTS analog circuitry. 0_B **DISABLED**: DTS is disabled 1_B **ENABLED**: DTS is enabled
START	1	rwh	**ADC conversion control** This field is used to start DTS conversions. 0_B **STOP**: DTS conversion is stopped or stopping 1_B **START**: DTS conversion is started
WRNEN	2	rwh	**Warning enable** This field is used as a global enable for all warnings/alarms. 0_B **DISABLED**: All warnings are disabled 1_B **ENABLED**: All warnings are enabled
PEAKDETRST	4	rwh	**DTS peak temperature detector reset** This field, when written with '1', resets the peak temperature detector. The bit is reset on LCK bit release after update in local DTS module instance. 0_B **NORESET**: Peak temperature detector is not reset 1_B **RESET**: Peak temperature detector is reset
LOWERFILT	11:8	rwh	**Temperature-sensing mode warning filter** This field configures the filter length for generation of LLU warning/ alarm. Warning/alarm is triggered after RESULT goes under LOWER by (LOWERFILT + 1) consecutive conversions.
UPPERFILT	15:12	rwh	**Temperature-sensing mode warning filter** This field configures the filter length for generation of UOF warning/ alarm. Warning/alarm is triggered after RESULT exceeds UPPER by (UPPERFILT + 1) consecutive conversions.

(table continues...)

(continued)

Field	Bits	Type	Description
LCK	31	rh	**Lock bit** This bit indicates if the register can be updated with a new value or if the register is locked and a write action from the bus side has no effect. Once the shadow register in local module instance is updated, the lock is released and LCK bit is set to 0. 0_B **UNLOCKED**: UNLOCKED The register is unlocked and can be updated. 1_B **LOCKED**: LOCKED The register is locked and cannot be updated.
0	3, 7:5, 30:16	r	**Reserved** Read as 0; should be written with 0.

Table 1677 Access mode restrictions of DTSx_CON (x=0-4) sorted by descending priority

Mode name	Access mode		Description
.LCK == 0	rwh	EN, LOWERFILT, PEAKDETRST, START, UPPERFILT, WRNEN	Write allowed when LCK='0'
(default)	rh	EN, LOWERFILT, PEAKDETRST, START, UPPERFILT, WRNEN	Write gated by LCK bit

38.4.18 Die Temperature Sensor x Limit Register 1

DTSx_LIM1 (x=0-4) Offset address: 064_H+x*20_H

Die Temperature Sensor x Limit Register 1 Kernel Reset value: $07FF\ 0469_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LCK	**UPPERHYST**			**UPPER**											
rh	rwh			rwh											

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	**LOWERHYST**			**LOWER**											
r	rwh			rwh											

Field	Bits	Type	Description
LOWER	11:0	rwh	**DTS lower warning threshold** This field configures the falling threshold for LLU warning/alarm. (Default at -37°C)
LOWERHYST	14:12	rwh	**Temperature-sensing mode warning hysteresis** This field configures the hysteresis for rising threshold of LLU warning/alarm, defined by (LOWER + LOWERHYST).
UPPER	27:16	rwh	**DTS upper warning threshold** This field configures the rising threshold for UOF warning/alarm. (Default at 155°C)

(table continues...)

(continued)

Field	Bits	Type	Description
UPPERHYST	30:28	rwh	**Temperature-sensing mode warning hysteresis** This field configures the hysteresis for falling threshold of UOF warning/ alarm, defined by (UPPER - UPPERHYST).
LCK	31	rh	**Lock bit** This bit indicates if the register can be updated with a new value or if the register is locked and a write action from the bus side has no effect. Once the shadow register in local module instance is updated, the lock is released and LCK bit is set to 0. 0_B **UNLOCKED**: UNLOCKED The register is unlocked and can be updated 1_B **LOCKED**: LOCKED The register is locked and cannot be updated
0	15	r	**Reserved** Read as 0; should be written with 0.

Table 1678 Access mode restrictions of DTSx_LIM1 (x=0-4) sorted by descending priority

Mode name	Access mode		Description
.LCK == 0	rwh	LOWER, LOWERHYST, UPPER, UPPERHYST	Write allowed when LCK='0'
(default)	rh	LOWER, LOWERHYST, UPPER, UPPERHYST	Write gated by LCK bit

38.4.19 Die Temperature Sensor x Limit Register 2

DTSx_LIM2 (x=0-4) Offset address: 068_H+x*20_H

Die Temperature Sensor x Limit Register 2 Kernel Reset value: $0000\ 0850_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LCK	0											TSDFILT			
rh	r											rwh			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	TSDHYST			TSDLIMIT											
r	rwh			rwh											

Field	Bits	Type	Description
TSDLIMIT	11:0	rwh	**Temperature shut down warning threshold** This field configures the rising threshold for Temperature Shut down warning/alarm. (Default at 172°C)
TSDHYST	14:12	rwh	**Temperature-sensing mode warning hysteresis** This field configures the hysteresis for falling threshold of TSD, defined by (TSDLIMIT - TSDHYST).

(table continues...)

(continued)

Field	Bits	Type	Description
TSDFILT	19:16	rwh	**Temperature-sensing mode warning filter** This field configures the filter length for generation of TSD warning/ alarm. Warning is triggered after RESULT exceeds TSDLIMIT by (TSDFILT + 1) consecutive conversions.
LCK	31	rh	**Lock bit** This bit indicates if the register can be updated with a new value or if the register is locked and a write action from the bus side has no effect. Once the shadow register in local module instance is updated, the lock is released and LCK bit is set to 0. 0_B **UNLOCKED**: UNLOCKED The register is unlocked and can be updated 1_B **LOCKED**: LOCKED The register is locked and cannot be updated
0	15, 30:20	r	**Reserved** Read as 0; should be written with 0.

Table 1679 Access mode restrictions of DTSx_LIM2 (x=0-4) sorted by descending priority

Mode name	Access mode		Description
.LCK == 0	rwh	TSDFILT, TSDHYST, TSDLIMIT	Write allowed when LCK='0'
(default)	rh	TSDFILT, TSDHYST, TSDLIMIT	Write gated by LCK bit

38.4.20 Die Temperature Sensor x Data Register

This register trigger data transfer from local instances to the VTMON registers. Also alarms going to SMU are latched in respective flags before being routed to SMU

DTSx_DATA (x=0-4) Offset address: $06C_H+x*20_H$

Die Temperature Sensor x Data Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0													TSDALRM	UOFALRM	LLUALRM
r													rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												BCAST	CLR	VALID	REQ
r												rw	w	rh	rwh

Field	Bits	Type	Description
REQ	0	rwh	**Data Transfer Request from DTSx to VTMON** Data transfer request is triggered from the all local DTS instances to transfer the local register configuration information. This field is cleared by hardware when the request has been serviced. 0_B **REQINACTIVE**: Request for data transfer inactive 1_B **REQACTIVE**: Request for data transfer active
VALID	1	rh	**Data Valid bit** This field indicates that the data transfer from the local shadow register of individual module instances to VTMON registers is complete 0_B **INVALID**: Data is invalid Data transfer ongoing and data in VTMON registers are invalid. 1_B **VALID**: Data is valid Data transfer completed and data in VTMON registers are valid.
CLR	2	w	**Clear Request** This field, when written with a '1', clears the LLUALRM, UOFALRM and TSDALRM alarm fields provided their respective temperature warnings are no longer existing. 0_B **NOCLEAR**: Status bits are not cleared 1_B **CLEAR**: Status bits are cleared
BCAST	3	rw	**Broadcast write enable** This field enables broadcast write function whereby register writes can be applied to all DTS instances simultaneously. 0_B **DISABLED**: Broadcase write is disabled. 1_B **ENABLED**: Broadcast write is enabled. A register write to any DTS register is broadcasted to the corresponding register of all DTS instances.
LLUALRM	16	rh	**Under Temperature Alarm** This field indicates if the last temperature-sensing conversion has triggered an under-temperature warning. A warning trigger would set this field, and an alarm is forwarded to the SMU. This field is cleared by writing '1' to the CLR field, provided that the warning is no longer existing. 0_B **INACTIVE**: Warning not active Last DTS result > LOWER. 1_B **TRIGGERED**: Warning triggered Last DTS result ≤ LOWER.

(table continues...)

(continued)

Field	Bits	Type	Description
UOFALRM	17	rh	**Over Temperature Alarm** This field indicates if the last temperature-sensing conversion has triggered an over-temperature warning. A warning trigger would set this field, and an alarm is forwarded to the SMU. This field is cleared by writing '1' to the CLR field, provided that the warning is no longer existing. 0_B **INACTIVE**: Warning not active Last DTS result < UPPER. 1_B **TRIGGERED**: Warning triggered Last DTS result ≥ UPPER.
TSDALRM	18	rh	**DTS thermal shutdown alarm** This field indicates if the last temperature-sensing conversion has triggered a thermal shutdown warning. A warning trigger would set this field, and an alarm is forwarded to the SMU. This field is cleared by writing '1' to the CLR field, provided that the warning is no longer existing. 0_B **INACTIVE**: Warning not active Last DTS result < TSDLIMIT. 1_B **TRIGGERED**: Warning triggered Last DTS result ≥ TSDLIMIT.
0	15:4, 31:19	r	**Reserved** Read as 0; should be written with 0.

38.4.21 Die Temperature Sensor x Status Register

The DTS status register value is updated regularly by a polling state machine which periodically polls the local DTS status registers and transfers the data every 200us interval. The status register is updated ~5us after the conversion is completed at the local DTS instance.

DTSx_STAT (x=0-4) Offset address: 070_H+x*20_H

Die Temperature Sensor x Status Register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DTSREADY	TSD	UOF	LLU	PEAK											
rh	rh	rh	rh	rh											

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				RESULT											
r				rh											

Field	Bits	Type	Description
RESULT	11:0	rh	**DTS result** This field contains the last temperature measurement result. This can be converted to temperature (in Kelvins) by dividing the readout by 4.781.

(table continues...)

(continued)

Field	Bits	Type	Description
PEAK	27:16	rh	**DTS peak temperature result** This field contains the peak temperature readout. This can be converted to temperature (Kelvins) by dividing by 4.781.
LLU	28	rh	**Low Temperature Warning** This field indicates if an under-temperature warning is active. 0_B **INACTIVE**: Warning not active Last DTS result > DTSxLIM1.LOWER. 1_B **TRIGGERED**: Warning triggered Last DTS result ≤ DTSxLIM1.LOWER.
UOF	29	rh	**High Temperature Warning** This field indicates if an over-temperature warning is active. 0_B **INACTIVE**: Warning not active Last DTS result < DTSxLIM1.UPPER. 1_B **TRIGGERED**: Warning triggered Last DTS result ≥ DTSxLIM1.UPPER.
TSD	30	rh	**Thermal Shutdown Warning** This field indicates if a thermal shutdown warning is active. 0_B **INACTIVE**: Warning not active Last DTS result < DTSxLIM2.TSDLIMIT. 1_B **TRIGGERED**: Warning triggered Last DTS result ≥ DTSxLIM2.TSDLIMIT.
DTSREADY	31	rh	**Ready indicator** This bit is cleared when DTS is disabled, and will be set only after we have the first DTS conversion result. Any alarm generation is gated with this bit, to avoid interrupts/alarms from triggering before the first conversion is complete. 0_B **BUSY**: DTS is disabled, or still starting up 1_B **DTSREADY**: DTS has completed startup
0	15:12	r	**Reserved** Read as 0.

38.5 Debug information

This section is not applicable for the module being described.

38.6 References

This section is not applicable for the module being described.

38.7 VTMON revision history

Initial release of the chapter.

38.8 TC4Dx VTMON information

38.8.1 TC4Dx VTMON configuration

There are no deviations from the generic specification.

38.8.2 TC4Dx VTMON features

There are no deviations from the generic specification.

38.8.3 TC4Dx VTMON functional description

The VTMON connects to various DTS instances located at different module locations across the die.

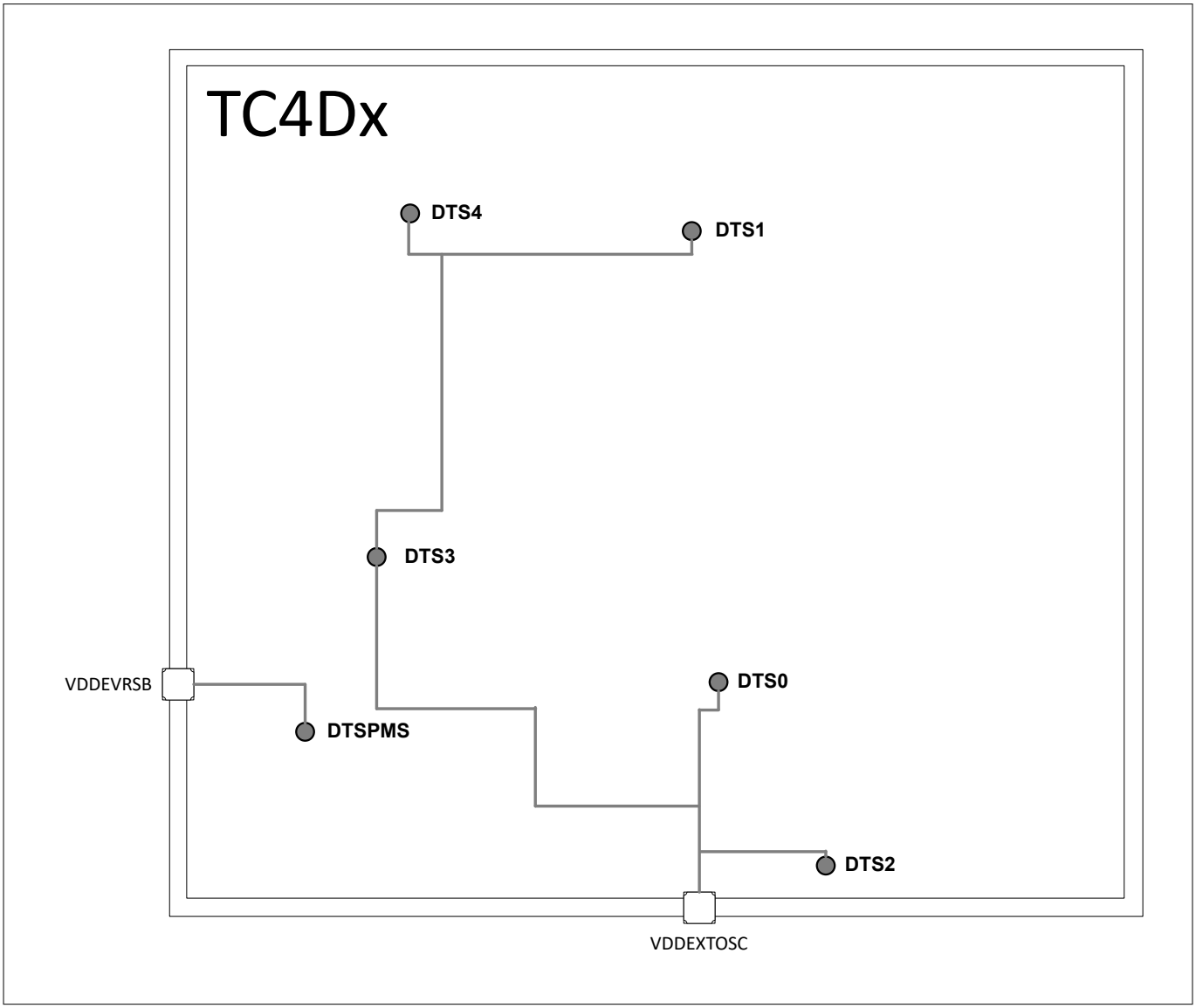


Figure 838 Location of DTS instances for TC4DX

38.8.4 TC4Dx VTMON registers

There are no deviations from the generic specification.

38.8.4.1 Register address space - VTMON

Table 1680 Registers address space - VTMON

Module	Base address	End address	Note
VTMON	$F006C000_H$	$F006FFFF_H$	

38.8.4.2 Register overview - access mode glossary

Table 1681 Register overview - access mode glossary

Keyword	Description
E	Access protection using PROT register VTMON_PROTE .
SE	Access protection using PROT register VTMON_PROTSE .
APU-P	Protection group consisting of registers VTMON_ACCEN_WRA , VTMON_ACCEN_WRB , VTMON_ACCEN_RDA , VTMON_ACCEN_RDB , VTMON_ACCEN_VM , VTMON_ACCEN_PRS .
P	Access protection using *APU-P* registers.
BE	Always returns a Bus Error.
U	No access restrictions.
PROT	Access restrictions as defined in the PROT register access rules.

38.8.4.3 Registers overview - VTMON (ascending offset address)

Table 1682 Registers overview - VTMON (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
VTMON_CLC	Clock Control Register	000_H	P	P, SV, E	Application Reset	6455
VTMON_OCS	OCDS Control and Status Register	004_H	P	SV, P	Debug Reset	6456
VTMON_ID	Module Identification Register	008_H	P	BE	PowerOn Reset	6457
VTMON_RST_CTRLA	Reset Control Register A	$00C_H$	P	P, SV, E	Application Reset	6457
VTMON_RST_CTRLB	Reset Control Register B	010_H	P	P, SV, E	Application Reset	6458
VTMON_RST_STAT	Reset Status Register	014_H	P	BE	Application Reset	6459
VTMON_PROTE	PROT Register Endinit	020_H	U	SV, PROT	Application Reset	6460
VTMON_PROTSE	PROT Register Safe Endinit	024_H	U	SV, PROT	Application Reset	6461

(table continues...)

Table 1682 (continued) Registers overview - VTMON (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
VTMON_ACCEN_WRA	Write access enable register A	040_H	U	SE, SV	Application Reset	6463
VTMON_ACCEN_WRB	Write access enable register B	044_H	U	SE, SV	Application Reset	6464
VTMON_ACCEN_RDA	Read access enable register A	048_H	U	SE, SV	Application Reset	6464
VTMON_ACCEN_RDB	Read access enable register B	$04C_H$	U	SE, SV	Application Reset	6465
VTMON_ACCEN_VM	VM access enable register	050_H	U	SE, SV	Application Reset	6465
VTMON_ACCEN_PRS	PRS access enable register	054_H	U	SE, SV	Application Reset	6466
VTMON_DTSx_CON (x=0-4)	Die Temperature Sensor x Control Register	060_H+x* 20_H	U	E, P, SV	Kernel Reset	6466
VTMON_DTSx_LIM1 (x=0-4)	Die Temperature Sensor x Limit Register 1	064_H+x* 20_H	U	E, P, SV	Kernel Reset	6468
VTMON_DTSx_LIM2 (x=0-4)	Die Temperature Sensor x Limit Register 2	068_H+x* 20_H	U	E, P, SV	Kernel Reset	6469
VTMON_DTSx_DATA (x=0-4)	Die Temperature Sensor x Data Register	$06C_H$+x* 20_H	U	E, P, SV	Kernel Reset	6470
VTMON_DTSx_STAT (x=0-4)	Die Temperature Sensor x Status Register	070_H+x* 20_H	U	BE	Kernel Reset	6472

38.8.4.4 Device specific registers

There are no device specific register changes.

38.8.5 TC4Dx VTMON connectivity

This section is not applicable for the module being described.

38.8.6 TC4Dx VTMON revision history

Initial release of the chapter.

39 Stand-by Controller (SCR)

The standby controller is intended to perform basic operations while the rest of the system is in standby, effectively reducing standby-power consumption while still supporting requested standby features. The SCR is based on an 8-bit XC800-core with a limited list of peripherals.

***Note**: SCR is classified as a quality management product, therefore there are no safety mechanisms.*

39.1 Feature list

- XC800 core
  - Based on the standard 8051 8-bit core
  - Two clocks per CPU cycle architecture, for memory access without wait state
  - Two data pointers
  - Maximum 100 MHz frequency
- On-chip memory
  - 4-Kbyte boot ROM for start-up and debug firmware
  - 256-byte internal memory; plus 64-byte Monitor memory
  - 32-Kbyte XRAM for program code and data
  - XRAM accessible by the TriCore™ domain CPUs through the SPB bus and XRAM arbiter
- Real-time clock (RTCSCR) to support periodic wake-up in standby mode
  - Support for crystal oscillator inputs - 32 kHz crystal from XTAL3/4 or external from SCR_P1.2 (P33.10)
  - Access to 64-bit Real-time Clock (RTC) and Die-Temperature Sensor (DTS) located in PMS address space
- Four general purpose I/O ports with up to 32 pins
  - Control of the shared pins in SCR Ports P0, P1, P2, and P3 (overlaid with Ports P32, P33, P34 and P35 of the TriCore™ domain)
- Two 16-bit timers T0 and T1 for general purpose usage (tightly coupled with the XC800 core)
- Two 16-bit timers with capture and compare unit (T2CCU0 and T2CCU1) for digital signal generation
- Watchdog timer (WDTSCR) with programmable window feature for refresh operation and warning prior to overflow
- Full duplex serial interface (UART)
- Synchronous serial channel (SSC)
- Inter-integrated circuit interface (I2CSCR)
- Wake-up CAN module (WCAN) to support Partial Networking (CAN ID filtering)
- ADC compare functionality (ADCOMP) with 16 channels and 11-bit resolution
- Local Interconnect Network (LIN) application support through extended UART features
- Interrupt supported from SCR to the main SoC interrupt system and vice-verse
- On-chip debug support through a single pin DAP interface (SPD)
- Power saving features
  - Idle mode of the SCR core
  - Clock gating control for each peripheral
  - Switch between STANDBY0 and STANDBY1 system modes without wake-up of the TriCore™ domain

***Note**: TriCore™ domain P35 (SCR Ports P2.5 - P2.7 and P3.5 - P3.7) are only available in the packages with extension STD*

39.2 Functional overview

The SCR is an 8-bit microcontroller that can continue to run while the TriCore™ domain is in the STANDBY state. It is based on the 8051 8-bit core which is a widely used industry standard processor. There is an embedded 32 kByte XRAM for program code and data.

Key features include two instances of a 16-bit general purpose timer with a capture/compare unit (T2CCU) for digital signal generation such as pulse generation, pulse width modulation, and pulse width measuring. A 41-bit Real Time Clock (RTCSCR) is included to support periodic wake-up in standby mode, and an On-Chip Debug Support (OCDS) unit for software development and debugging of XC800 based systems. Local Interconnect Network (LIN) applications are supported through extended UART features. SPI functionality is provided through the SSC module. Additionally an Inter-integrated circuit (I2CSCR) module is available. Wake-up CAN capabilities are provided with the WCAN module. An 11-bit SAR ADC is available for analog measurements in standby mode. For low-power applications, various power saving techniques are available. Control of the numerous on-chip peripheral functions is achieved by extending the Special Function Register (SFR) address range with a paging mechanism.

The following figure illustrates the available sub-modules within the SCR of the TC4xx device.

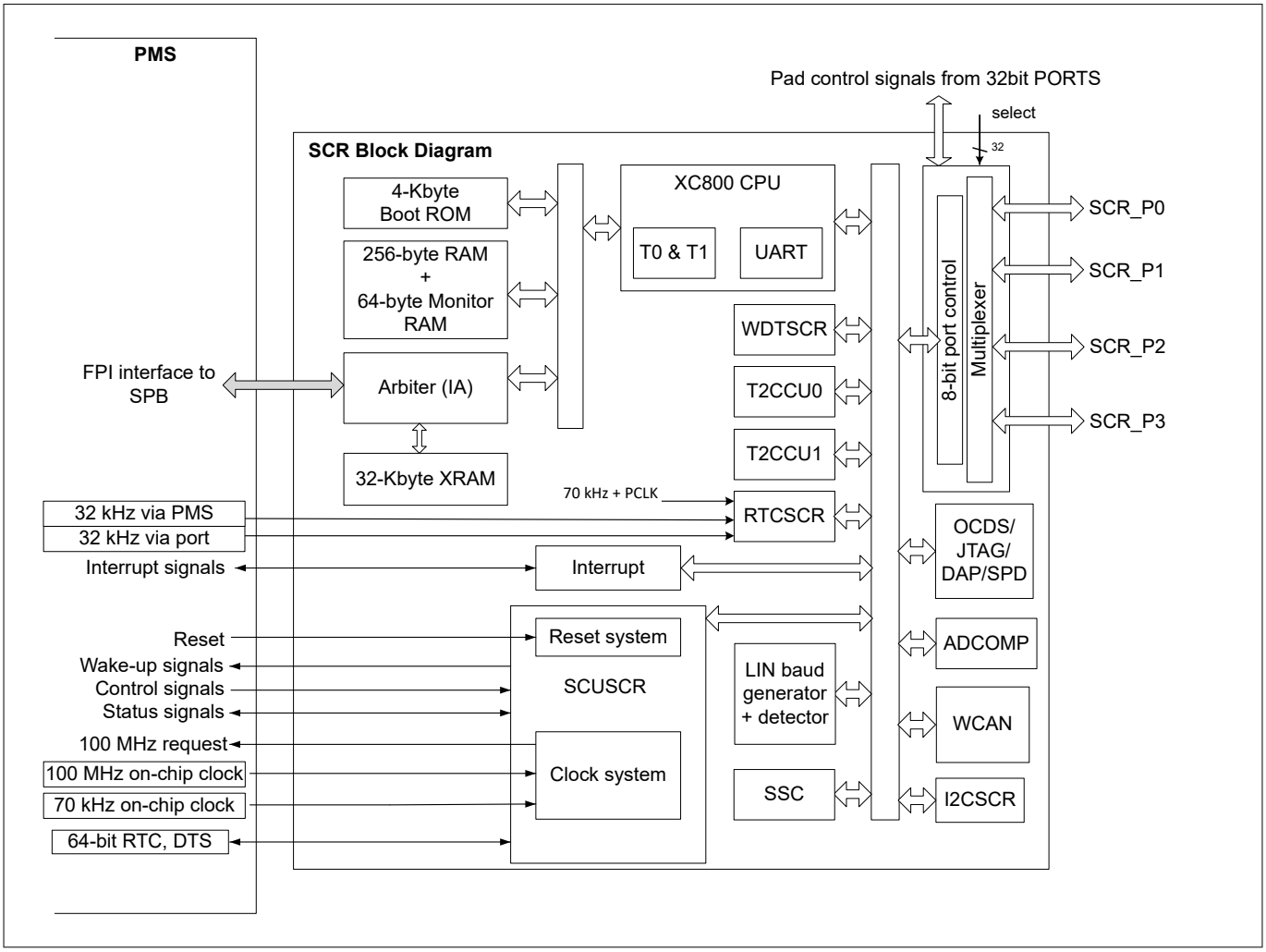


Figure 839 SCR block diagram

39.3 Functional description

The standby controller is a full sub-system consisting of a core, memory, internal bus, and dedicated peripherals.

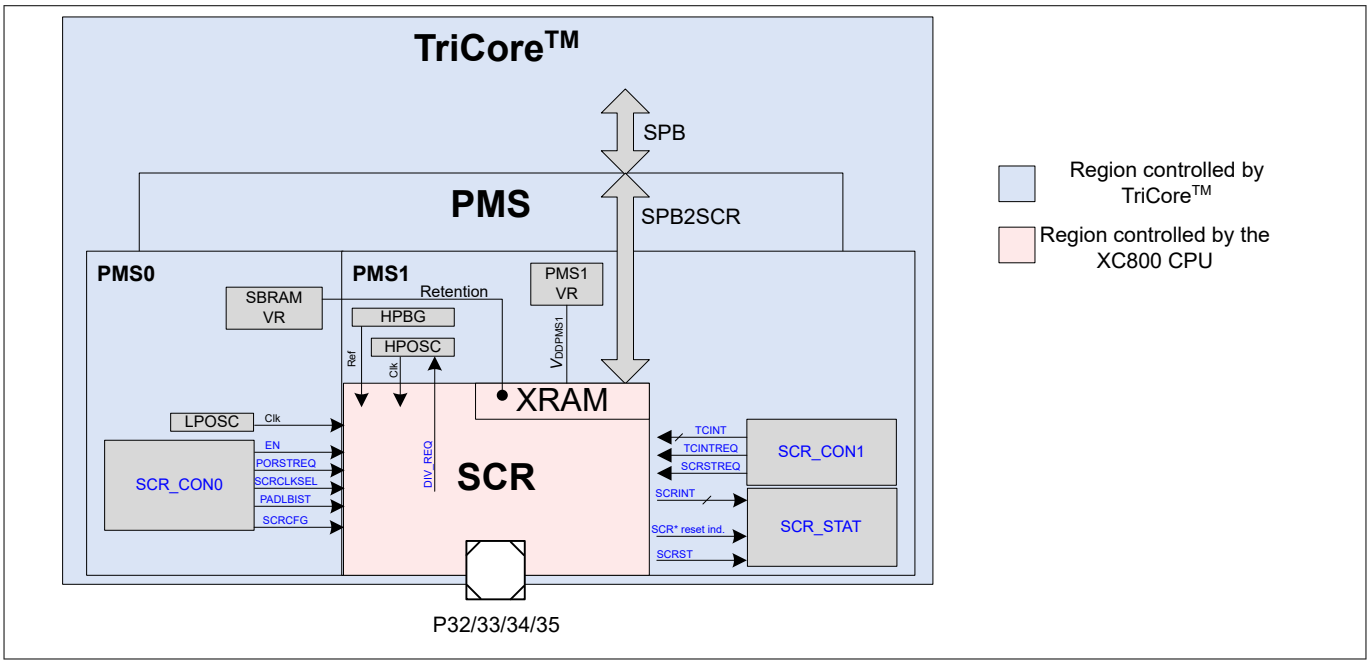


Figure 840 SCR position in the system

39.3.1 XC800 CPU

The SCR registers occupy direct Internal Data Memory space locations in the range 80_H to FF_H.

In the following sections main core and core-peripheral registers are briefly introduced.

***Note**: The SCR includes the XC800 core. Due to the special structure, this consists out of the XC800 CPU, Timer 0 and 1 as well as the UART.*

Related information

Registers on page 802

39.3.1.1 SFRs of the CPU

Stack pointer (SP, $D4_H$)

The SP register contains the stack pointer. The stack pointer is used to store the program counter into internal data memory during LCALL and ACALL instructions and to restore the program counter from memory during RET and RETI instructions. Data may also be saved on or retrieved from the stack using PUSH and POP instructions. Instructions that use the stack automatically pre-increment or post-decrement the stack pointer so that the stack pointer always points to the last byte written to the stack, that is the top of the stack. On reset, the stack pointer is reset to 07_H. This causes the stack to begin at a location = 08_H above register bank zero. The SP can be read or written under software control. The programmer must ensure that the location and size of the stack in internal data memory do not overlap with other application data.

Data pointer (DPTR = DPL-DPH, $D5\text{-}6_H$)

The data pointer (DPTR) consists of a high byte (DPH) and a low byte (DPL). Its intended function is to hold a 16-bit address. It may be manipulated as a 16-bit register or as two independent 8-bit registers. 16-bit addressing is used for:

- External Data Memory accesses (MOVX A,@DPTR and MOVX @DPTR,A)
- Program byte moves (MOVC A,@A+DPTR)
- Indirect program jumps (JMP @A+DPTR)

Two true 16-bit operations are allowed on the data pointer: load immediate (MOVDPTR,#data) and increment (INC DPTR).

Accumulator (ACC, $E0_H$)

This register provides one of the operands for most ALU operations. ACC is the symbol for the accumulator register. The mnemonics for accumulator-specific instructions refer to the accumulator simply as “A”.

B register (B, DA_H)

The B register is used during multiply and divide operations to provide the second operand. For other instructions, it can be treated as another scratch pad register.

Program status word (PSW, $D0_H$)

The PSW register contains program status information – parity, overflow, carry and general purpose flags together with register bank select control bits.

Extended operation (EO, $D7_H$)

The instruction set includes an additional instruction MOVC @(DPTR++),A which writes to program memory implemented as RAM. The instruction copies the contents of the accumulator to the code memory at the location pointed to by the current data pointer, then increments the data pointer.

The instruction uses the opcode $A5_H$, which is the same as the software break instruction TRAP (see also XC800 CPU instruction timing table for related information). Bit TRAP_EN in the Extended Operation (EO) register is used to select the instruction executed by the opcode $A5_H$. When bit TRAP_EN is 0 (default after reset), the $A5_H$ opcode executes the MOVC instruction. When bit TRAP_EN is 1, the $A5_H$ opcode executes the software break instruction TRAP, which switches the CPU to debug mode for breakpoint processing.

Register EO is also used to select the current data pointer through the DPSEL bit-field.

Power control (PCON, $D9_H$)

The PCON register provides:

- Control for entering idle mode
- Baudrate control for UART in mode 2 (for details see UART section)
- Two general purpose flags

In idle mode, the clock to the CPU (CCLK) is halted while all peripherals running on PCLK are still running. A wake-up is initiated by an interrupt.

Interrupt controller, serial port and core timers

Additional registers located inside the XC800 core:

- Interrupt controller (IR) registers
  - Interrupt enable register 0: IEN0
  - Interrupt enable register 1: IEN1
  - Interrupt priority register: IP
  - Interrupt priority 1 register: IP1
  - Interrupt priority high register: IPH
  - Interrupt priority 1 high register: IPH1
- Serial port/Universal Asynchronous Receiver/Transmitter (UART) registers
  - Serial channel control register: UART_SCON
  - Serial data buffer register: UART_SBUF
- Timer 0 and Timer 1 registers
  - Timer 0/1 control register: T01_TCON
  - Timer 0/1 mode register: T01_TMOD
  - Timer 0 low byte register: T01_TL0
  - Timer 1 low byte register: T01_TL1
  - Timer 0 high byte register: T01_TH0
  - Timer 1 high byte register: T01_TH1

These registers and the logic function around them are clocked with the peripheral clock (PCLK).

More details on these registers and the function associated to them is provided in the respective IR, UART, and T01 sections.

Related information

Instruction timing on page 6483

39.3.1.2 Instruction timing

A CPU cycle comprises two input clock periods, referred to as Phase 1 (P1) and Phase 2 (P2), that correspond to two different CPU states.

A CPU state within an instruction is referenced by the CPU cycle and state number. For example C2P1 means the first clock period within CPU cycle 2.

Memory access takes place during one or both phases of the CPU cycle.

SFR writes occur only at the end of P2.

Instructions are 1, 2, or 3 bytes long and can take 1, 2, or 4 CPU cycles to execute. Registers are generally updated and the next opcode pre-fetched at the end of P2 of the last CPU cycle for the current instruction.

The XC800 CPU supports access to slow memory by using wait cycle(s). Each wait cycle lasts one CPU cycle which is two clock periods. For example, in case of a memory requiring one or two wait state(s), the access time is increased by one CPU cycle for every byte of opcode or operand fetched.

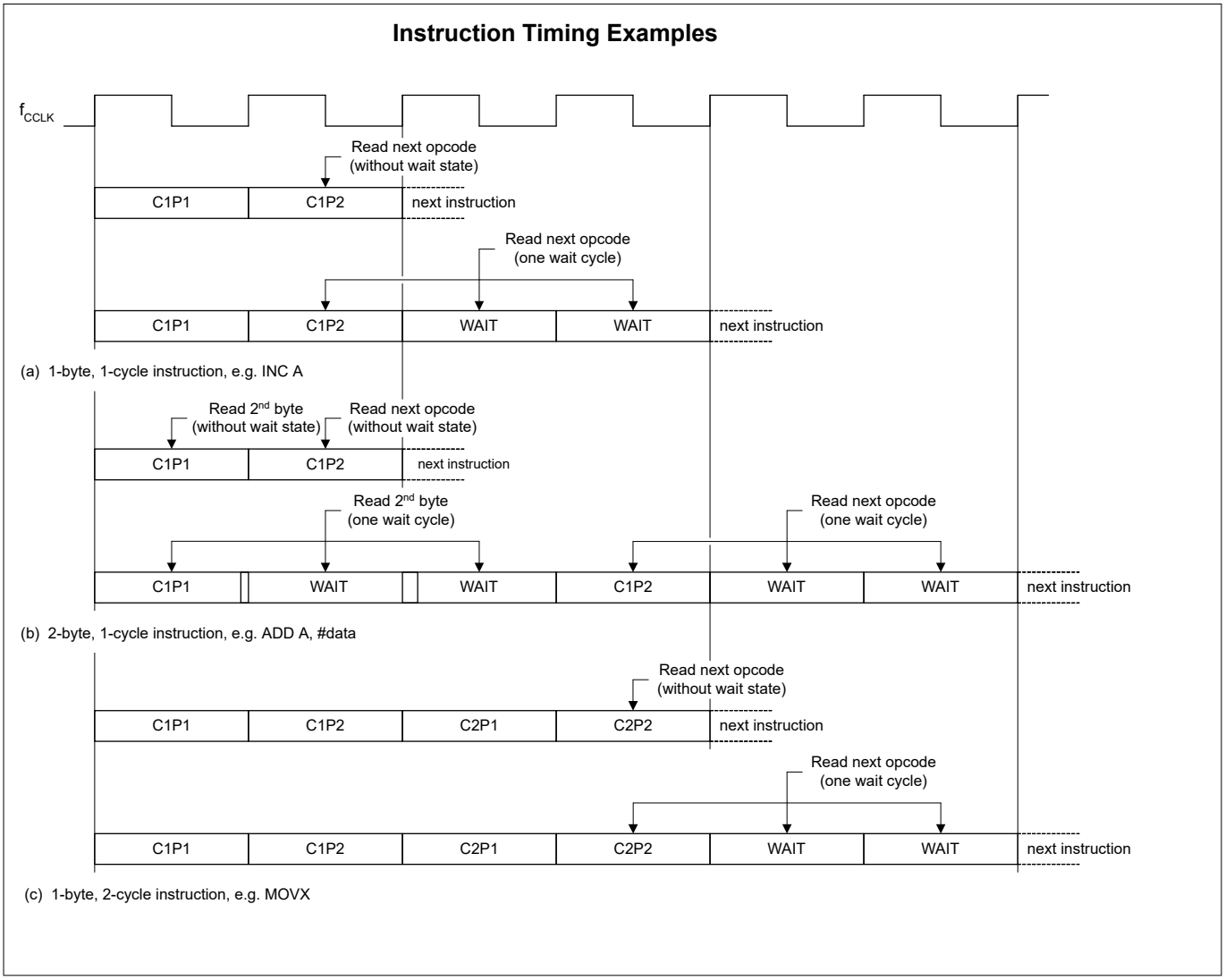


Figure 841 CPU instruction timing

The figure above shows the fetch/execute timing related to the internal states and phases. Execution of an instruction occurs at C1P1. For a 2-byte instruction, the second reading starts at C1P1.

Part (a) of the figure above shows two timing diagrams for a 1-byte, 1-cycle (1 x CPU cycle) instruction. The first diagram shows the instruction being executed within one CPU cycle since the opcode (C1P2) is fetched from a memory without wait state. The second diagram shows the corresponding states of the same instruction being executed over two CPU cycles (instruction time extended), with one wait cycle inserted for opcode fetching from the flash memory.

Part (b) of the figure above shows two timing diagrams for a 2-byte, 1-cycle (1 x CPU cycle) instruction. The first diagram shows the instruction being executed within one CPU cycle since the second byte (C1P1) and the opcode (C1P2) are fetched from a memory without wait state. The second diagram shows the corresponding states of the same instruction being executed over three CPU cycles (instruction time extended), with one wait cycle inserted for each access to the flash memory. In this case, two wait cycles are inserted in total.

Part (c) of the figure above shows two timing diagrams of a 1-byte, 2-cycle (2 x CPU cycle) instruction. The first diagram shows the instruction being executed over two CPU cycles with the opcode (C2P2) fetched from a memory without wait state. The second diagram shows the corresponding states of the same instruction being executed over three CPU cycles (instruction time extended), with one wait cycle inserted for opcode fetching from the slow memory requiring one/two wait state(s).

***Note**: For instructions that are executed over two or more CPU cycles, execution cycle may or may not be extended in case of access to slow memory with one or two wait states. The execution cycle is, nonetheless, guaranteed consistent for each instruction when accessed from slow memory with defined wait state(s).*

The time taken for each instruction includes:

- Decoding/executing the fetched opcode
- Fetching the operand/s (for instructions > 1 byte)
- Fetching the first byte (opcode) of the next instruction (due to CPU pipeline)

***Note**: The XC800 CPU fetches the opcode of the next instruction while executing the current instruction.*

Instructions are 1, 2, or 3 bytes long as indicted in the 'Bytes' column. Each instruction takes 1, 2 or 4 CPU cycles to execute (with no wait cycle). The table below gives two values for the number of CPU cycles required by each instruction. The first value applies to fetching operand/s and opcode from fast memory (for example Boot ROM and XRAM) without wait state. The second value applies to fetching operand/s and opcode (and in some cases accessing data) from slow memory (for example Flash) with wait cycles inserted due to memory requiring one or two wait state(s). One CPU cycle comprises two CCLK clock cycles.

Table 1683 XC800 CPU instruction timing table

Mnemonic	Hex code	Bytes	CPU cycles (no wait state)	CPU cycles (one wait state [1])
ARITHMETIC				
ADD A,Rn	28-2F	1	1	2
ADD A,dir	25	2	1	3
ADD A,@Ri	26-27	1	1	2
ADD A,#data	24	2	1	3
ADDC A,Rn	38-3F	1	1	2
ADDC A,dir	35	2	1	3
ADDC A,@Ri	36-37	1	1	2
ADDC A,#data	34	2	1	3
SUBB A,Rn	98-9F	1	1	2
SUBB A,dir	95	2	1	3
SUBB A,@Ri	96-97	1	1	2
SUBB A,#data	94	2	1	3
INC A	04	1	1	2
INC Rn	08-0F	1	1	2
INC dir	05	2	1	3

(table continues...)

Table 1683 (continued) XC800 CPU instruction timing table

Mnemonic	Hex code	Bytes	CPU cycles (no wait state)	CPU cycles (one wait state [1])
INC @Ri	06-07	1	1	2
DEC A	14	1	1	2
DEC Rn	18-1F	1	1	2
DEC dir	15	2	1	3
DEC @Ri	16-17	1	1	2
INC DPTR	A3	1	2	2
MUL AB	A4	1	4	4
DIV AB	84	1	4	4
DA A	D4	1	1	2
LOGICAL				
ANL A,Rn	58-5F	1	1	2
ANL A,dir	55	2	1	3
ANL A,@Ri	56-57	1	1	2
ANL A,#data	54	2	1	3
ANL dir,A	52	2	1	3
ANL dir,#data	53	3	2	5
ORL A,Rn	48-4F	1	1	2
ORL A,dir	45	2	1	3
ORL A,@Ri	46-47	1	1	2
ORL A,#data	44	2	1	3
ORL dir,A	42	2	1	3
ORL dir,#data	43	3	2	5
XRL A,Rn	68-6F	1	1	2
XRL A,dir	65	2	1	3
XRL A,@Ri	66-67	1	1	2
XRL A,#data	64	2	1	3
XRL dir,A	62	2	1	3
XRL dir,#data	63	3	2	5
CLR A	E4	1	1	2
CPL A	F4	1	1	2
SWAP A	C4	1	1	2
RL A	23	1	1	2
RLC A	33	1	1	2

(table continues...)

Table 1683 (continued) XC800 CPU instruction timing table

Mnemonic	Hex code	Bytes	CPU cycles (no wait state)	CPU cycles (one wait state [1])
RR A	03	1	1	2
RRC A	13	1	1	2
DATA TRANSFER				
MOV A,Rn	E8-EF	1	1	2
MOV A,dir	E5	2	1	3
MOV A,@Ri	E6-E7	1	1	2
MOV A,#data	74	2	1	3
MOV Rn,A	F8-FF	1	1	2
MOV Rn,dir	A8-AF	2	2	4
MOV Rn,#data	78-7F	2	1	3
MOV dir,A	F5	2	1	3
MOV dir,Rn	88-8F	2	2	4
MOV dir,dir	85	3	2	5
MOV dir,@Ri	86-87	2	2	4
MOV dir,#data	75	3	2	5
MOV @Ri,A	F6-F7	1	1	2
MOV @Ri,dir	A6-A7	2	2	4
MOV @Ri,#data	76-77	2	1	3
MOV DPTR,#data	90	3	2	5
MOVC A,@A+DPTR	93	1	2	3 or 4[2]
MOVC A,@A+PC	83	1	2	3 or 4[2]
MOVX A,@Ri	E2-E3	1	2	3
MOVX A,@DPTR	E0	1	2	3
MOVX @Ri,A	F2-F3	1	2	3
MOVX @DPTR,A	F0	1	2	3
PUSH dir	C0	2	2	4
POP dir	D0	2	2	4
XCH A,Rn	C8-CF	1	1	2
XCH A,dir	C5	2	1	3
XCH A,@Ri	C6-C7	1	1	2
XCHD A,@Ri	D6-D7	1	1	2
BOOLEAN				
CLR C	C3	1	1	2

(table continues...)

Table 1683 (continued) XC800 CPU instruction timing table

Mnemonic	Hex code	Bytes	CPU cycles (no wait state)	CPU cycles (one wait state [1])
CLR bit	C2	2	1	3
SETB C	D3	1	1	2
SETB bit	D2	2	1	3
CPL C	B3	1	1	2
CPL bit	B2	2	1	3
ANL C,bit	82	2	2	4
ANL C,/bit	B0	2	2	4
ORL C,bit	72	2	2	4
ORL C,/bit	A0	2	2	4
MOV C,bit	A2	2	1	3
MOV bit,C	92	2	2	4
BRANCHING[3]				
ACALL addr11	11->F1	2	2	4
LCALL addr16	12	3	2	5
RET	22	1	2	2 or 3[2]
RETI	32	1	2	2 or 3[2]
AJMP addr 11	01->E1	2	2	4
LJMP addr 16	02	3	2	5
SJMP rel	80	2	2	4
JC rel	40	2	2	4
JNC rel	50	2	2	4
JB bit,rel	20	3	2	5
JNB bit,rel	30	3	2	5
JBC bit,rel	10	3	2	5
JMP @A+DPTR	73	1	2	2 or 3[2]
JZ rel	60	2	2	4
JNZ rel	70	2	2	4
CJNE A,dir,rel	B5	3	2	5
CJNE A,#d,rel	B4	3	2	5
CJNE Rn,#d,rel	B8-BF	3	2	5
CJNE @Ri,#d,rel	B6-B7	3	2	5
DJNZ Rn,rel	D8-DF	2	2	4
DJNZ dir,rel	D5	3	2	5

(table continues...)

Table 1683 (continued) XC800 CPU instruction timing table

Mnemonic	Hex code	Bytes	CPU cycles (no wait state)	CPU cycles (one wait state [1])
MISCELLANEOUS				
NOP	00	1	1	2
ADDITIONAL INSTRUCTIONS				
MOVC @(DPTR++),A	A5	1	2	2 or 3[2]
TRAP	A5	1	1	–

1) In the case of a fetch from slow memory requiring only 1 wait state, no wait cycle may be required per the instruction fetched (normally for opcodes that do not require fetch in the next cycle).
2) Depending on whether the operation is accessing memory with zero or one/two wait state.
3) For branch instructions, the instruction time may vary depending on jump destination.

39.3.1.3 XRAM addressing modes

External Data Memory (XRAM) is accessed using MOVX instructions only. These use either 8-bit or 16-bit indirect address. The DPTR register is used for 16-bit addressing and either register R0 or R1 is used to form the 8-bit address.

39.3.1.3.1 Access to XRAM using the DPTR (16-bit addressing mode)

The external memory can be accessed by two read/write instructions, which use the 16-bit DPTR for indirect addressing. These instructions are:

- MOVX A, @DPTR (Read)
- MOVX @DPTR, A (Write)

39.3.1.3.2 Access to XRAM using the register R0/R1 (8-bit addressing mode)

The SCR provides also instructions for accesses to XRAM using the 8-bit address (indirect addressing with registers R0 or R1). These instructions are:

- MOVX A, @Ri (Read)
- MOVX @Ri, A (Write)

To access the XRAM properly, a higher order address must be put in SFR XADDRH . This write instruction must precede the MOVX instructions.

39.3.2 Memory organization

The XC800 CPU operates in the following four address spaces:

- 256 byte of internal data memory without ECC protection
- 32 Kbyte of XRAM memory (32-bit memory, accessible to TriCore™ through the System Peripheral Bus in PMS address space):
  - ECC protected
  - XRAM can be read and written as program memory and external data memory
- 128-byte Special Function Register area
- The available memory ECC protection is capable to correct single bit errors and to detect double bit errors (SEC/DED)
- 4 Kbyte of Boot ROM program memory

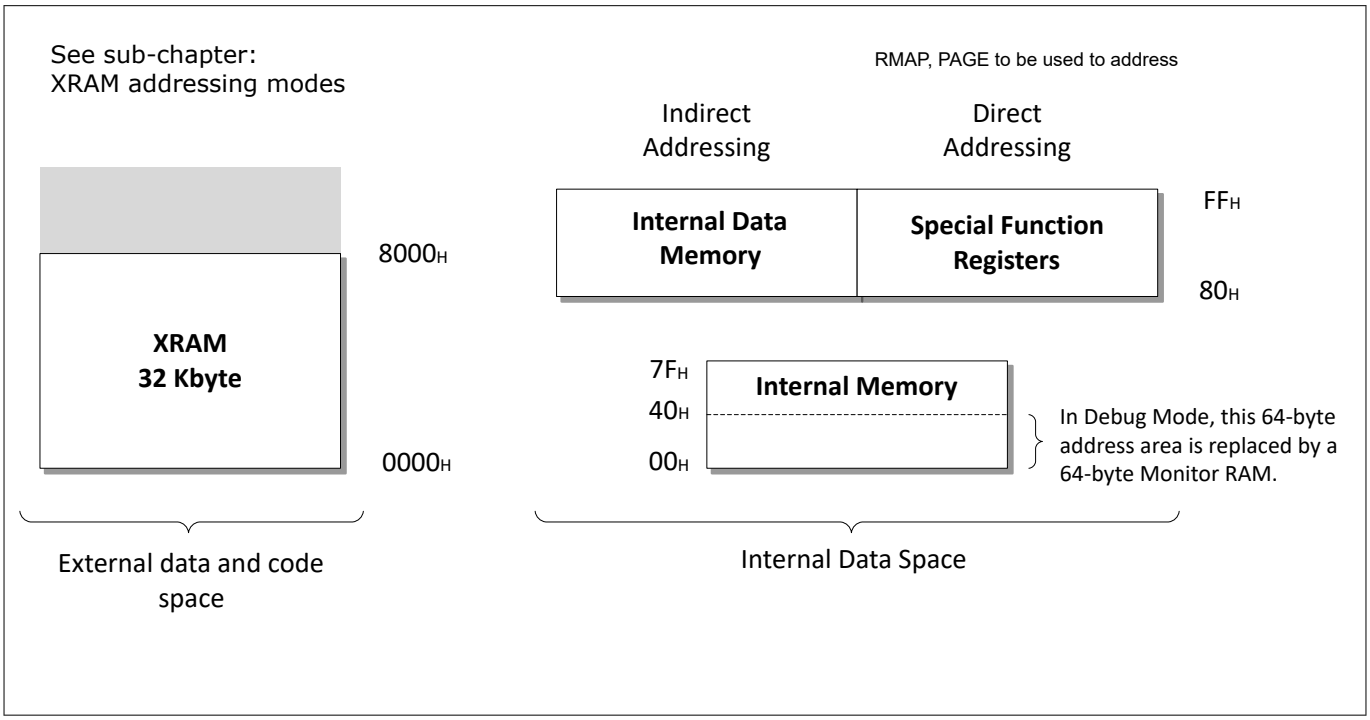


Figure 842 SCR memory map

39.3.2.1 Program memory

However, only access to defined program and data memory regions like XRAM (used for code and data) or Boot ROM is supported. For the memory organization (MemOrg), defined code and data space (XRAM) is occupied by on-chip memories.

Related information

Memory organization on page 6490

39.3.2.2 Data memory

The data space consists of an internal and external data space. Access to internal and external data space are distinguished by different sets of instruction opcodes. In the SCR, the on-chip XRAM is located in external data space and accessed by MOVX instructions.

SCR does not support access to external (off-chip) memory. Internal data space is occupied by Internal Memory and Special Function Registers (SFRs), distinguished by direct or indirect addressing.

39.3.2.2.1 Internal data memory

The internal data memory is divided into two physically separate and distinct blocks: the 256-byte Memory and the 128-byte Special Function Register (SFR) area. While the upper 128 bytes of Memory and the SFR area share the same address locations, they are accessed through different addressing modes.

The lower 128 bytes of Memory can be accessed through either direct or register indirect addressing, while the upper 128 bytes of Memory can be accessed through register indirect addressing only. The SFRs are accessible through direct addressing.

The 16 bytes of Memory that occupy addresses from 20_H to $2F_H$ are bit-addressable. Memory occupying direct addresses from 30_H to $7F_H$ can be used as scratch pad registers or used for the stack.

39.3.2.2.2 External data memory

The 32-Kbyte XRAM is mapped to both the external data memory area and the program memory area. The SCR XRAM can be accessed by both the SCR's XC800 core and the main TriCore™ CPU. The arbiter module determines which CPU has control of the XRAM at any one time. The TriCore™ CPU accesses the XRAM through the peripheral bus. An FPI slave interface is provided to the SCR to interface to the SPB bus. The arbiter module is responsible for arbitrating between the SCR core and the TriCore™ accesses to the XRAM through the SPB.

For memory access protection see PMS Standby controller extension RAM (XRAM) in related information.

SCR core accesses the XRAM using either the 'MOVX' or 'MOVC' instructions. The 'MOVX' instructions for XRAM access use either 8-bit or 16-bit indirect addresses, as shown in the following table.

Table 1684 Instructions for external data memory access

Type of access	16-bit addressing mode	8-bit addressing mode
Read	MOVX A, @DPTR	MOVX A, @Ri
Write	MOVX @DPTR, A	MOVX @Ri, A

While the DPTR register is used for 16-bit addressing, either register R0 or R1 is used to form the 8-bit address. The upper byte of the XRAM address during execution of the 8-bit accesses is defined by the value stored in register XADDRH. Hence, the write instruction for setting the higher order XRAM address in register XADDRH must precede the 'MOVX' instruction.

A write access to a undefined external data memory address has no effect while a read access returns undefined data.

Related information

Standby controller extension RAM (XRAM) on page 6039

39.3.2.3 Special function registers (SFRs)

The Special Function Registers (SFRs) occupy direct internal data memory space in the range 80_H to FF_H. All registers, except the program counter, reside in the SFR area. The SFRs include pointers and registers that provide an interface between the CPU and the on-chip peripherals. As the 128-SFR range is less than the total number of registers required, address extension mechanisms are required to increase the number of addressable SFRs. The address extension mechanisms include:

- Mapping
- Paging

39.3.2.3.1 Address extension by mapping

Address extension is performed at the system level by mapping. The SFR area is extended into two portions: the standard (non-mapped) SFR area and the mapped SFR area. Each portion supports the same address range 80_H

to FF_H, bringing the number of addressable SFRs to 256. The extended address range is not directly controlled by the CPU instruction itself, but is derived in the system control register from ROM bit SYSCON0.RMAP at address DF_H. To access SFRs in the mapped area, bit SYSCON0.RMAP must be set. However, the SFRs in the standard area can be accessed by clearing bit SYSCON0.RMAP. Figure below shows how the SFR area can be selected.

As long as bit SYSCON0.RMAP is set, the mapped SFR area can be accessed. This bit is not cleared automatically by hardware. Therefore, before standard/mapped registers are accessed, bit SYSCON0.RMAP must be cleared/set, respectively, by software.

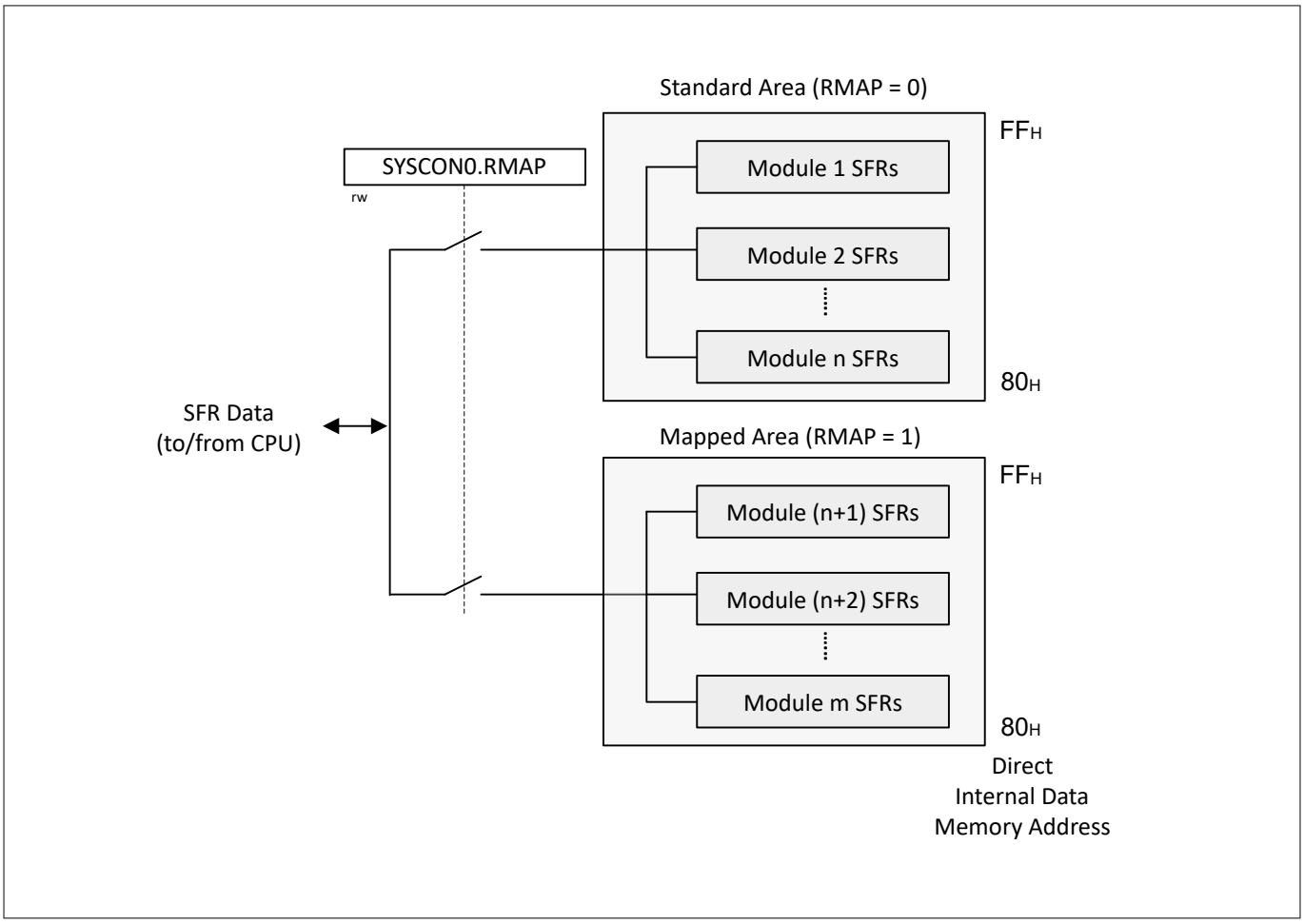


Figure 843 Address extension by mapping

39.3.2.3.2 Address extension by paging

Address extension is further performed at the module level by paging. Certain peripherals have a built-in local address extension mechanism for increasing the number of addressable SFRs. The extended address range is not directly controlled by the CPU instruction itself, but is derived from bit-field PAGE in the module page register MOD_PAGE. Hence, the bit-field PAGE must be programmed before accessing the SFRs of the target module. Each module may contain a different number of pages and a different number of SFRs per page, depending on the specific requirement. Besides setting the correct RMAP bit value to select the SFR area, the user must also ensure that a valid PAGE is selected to target the desired SFRs.

Figure below shows how a page inside the extended address range can be selected.

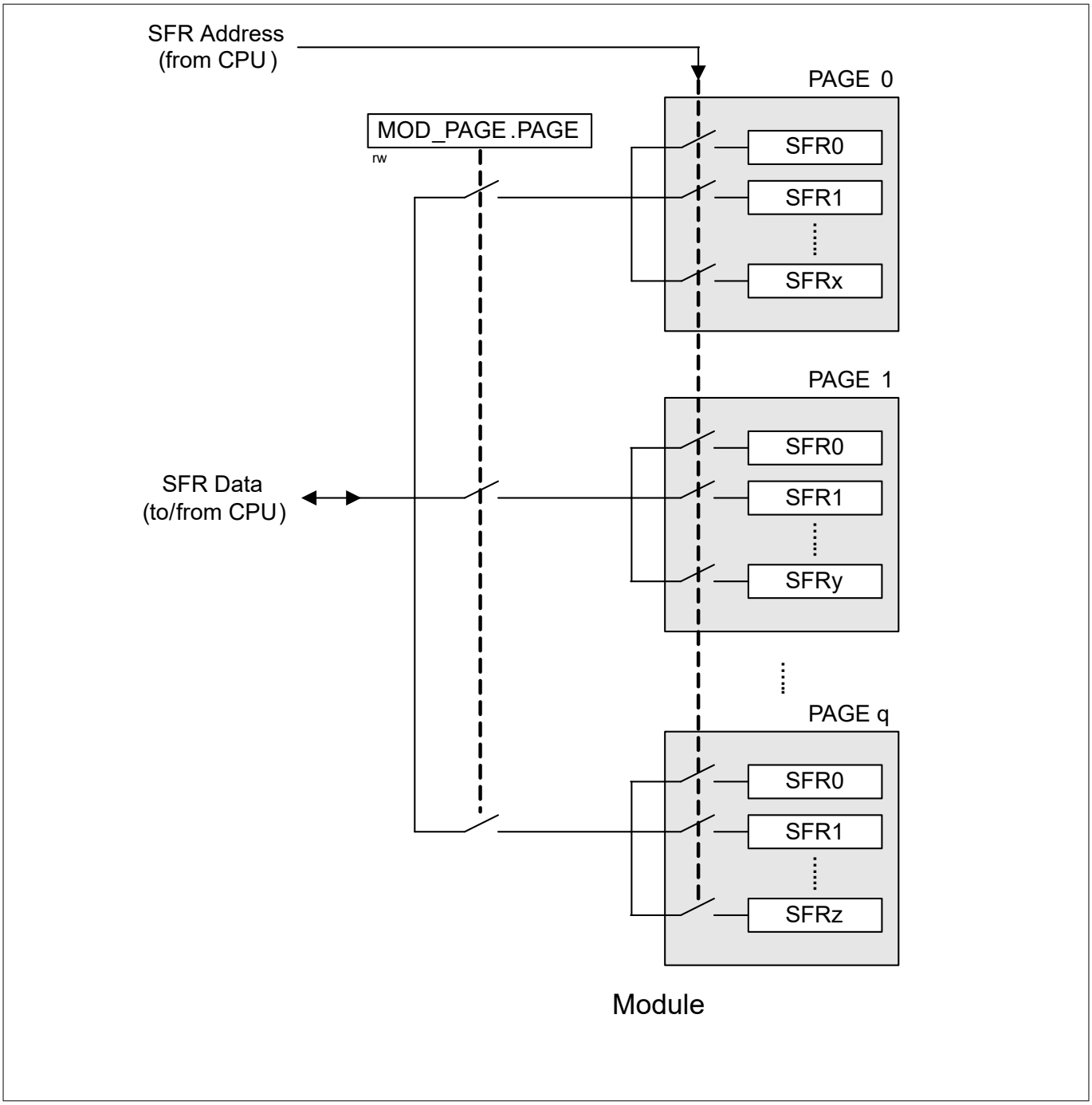


Figure 844 Address extension by paging

In order to access a register located in a page other than the current one, the current page must be exited. This is done by reprogramming the bit-field PAGE in the page register. Only then can the desired access be performed.

If an interrupt routine is initiated between the page register access and the module register access, and the interrupt needs to access a register located in another page, the current page setting can be saved, the new one programmed, and the old page setting restored. This is possible with the storage fields MOD_STx (x=0-3) for the save and restore action of the current page setting. By indicating which storage bit-field should be used in parallel with the new page value, a single write operation can either:

- Save the contents of PAGE in MOD_STx before overwriting with the new value

(this is done at the beginning of the interrupt routine to save the current page setting and program the new page number)

- Overwrite the contents of PAGE with the contents of MOD_STx, ignoring the value written to the bit positions of PAGE

  (this is done at the end of the interrupt routine to restore the previous page setting before the interrupt occurred)

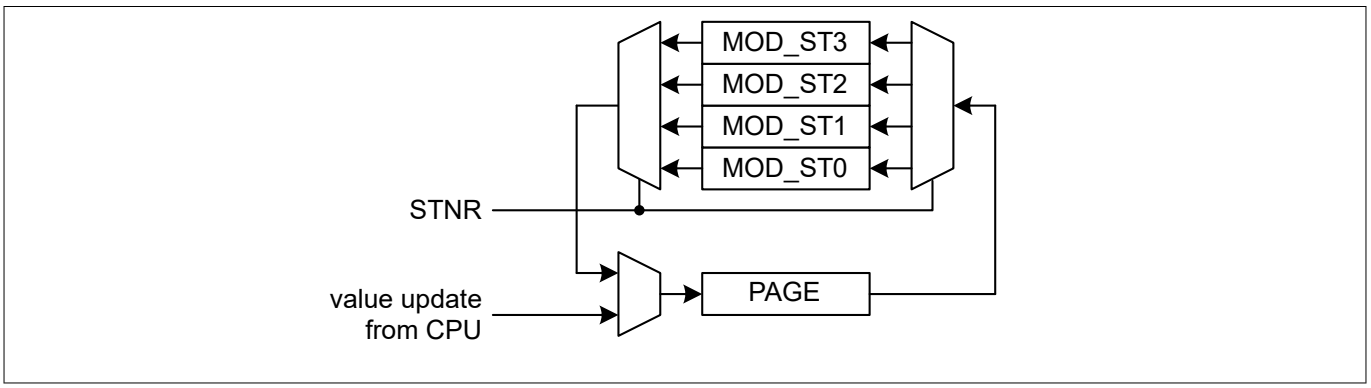


Figure 845 Storage elements for paging

With this mechanism, a certain number of interrupt routines (or other routines) can perform page changes without reading and storing the previously used page information. The use of only write operations makes the system simpler and faster. Consequently, this mechanism significantly improves the performance of short interrupt routines.

Page Register for module MOD

MOD_PAGE Address:

Page Register for module MOD : 00_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res								OP		STNR		0	PAGE		
r								w		w		r	rw		

Field	Bits	Type	Description
PAGE	2:0	rw	**Page Bits** When written, the value indicates the new page. When read, the value indicates the currently active page.

(table continues...)

(continued)

Field	Bits	Type	Description
STNR	5:4	w	**Storage Number** This number indicates which storage bit field is the target of the operation defined by bit-field OP. If OP=10_B, the contents of PAGE are saved in MOD_STx before being overwritten with the new value. If OP=11_B, the contents of PAGE are overwritten by the contents of MOD_STx. The value written to the bit positions of PAGE is ignored. 00_B MOD_ST0 is selected. 01_B MOD_ST1 is selected. 10_B MOD_ST2 is selected. 11_B MOD_ST3 is selected.
OP	7:6	w	**Operation** $0X_B$ Manual page mode. The value of STNR is ignored and PAGE is directly written. 10_B New page programming with automatic page saving. The value written to the bit positions of PAGE is stored. In parallel, the previous contents of PAGE are saved in the storage bit-field STx indicated by STNR. 11_B Automatic restore page action. The value written to the bit positions PAGE is ignored and instead, PAGE is overwritten by the contents of the storage bit-field STx indicated by STNR.
0	3	r	**Reserved** Returns 0 if read; shall be written with 0.

39.3.2.3.3 Bit-addressing

SFRs that have addresses in the form of $1XXXX000_B$ (for example,80_H, 88_H, 90_H,..., $F0_H$, $F8_H$) are bit addressable.

39.3.2.3.4 Bit protection scheme

The bit protection scheme prevents direct software writing of selected bits (protected bits) using the PASSWD register. When the bit PASSWD.MODE is 11_B, writing 10011_B to the bit-PASSWD.PASS opens access to writing of all protected bits, and writing 10101_B to the bit-PASSWD.PASS closes access to writing of all protected bits. In both cases, the value of the bit field MODE is not changed even if PASSWD register is written with 98_H or $A8_H$. It can only be changed when bit-field PASS is written with 11000_B, for example, writing $C0_H$ to PASSWD register disables the bit protection scheme.

Note that access is opened for maximum 34 (TC4Dx) / 32 (all other devices) CCLKs if the “close access” password is not written. If “open access” password is written again before the end of 34 (TC4Dx) / 32 (all other devices) CCLK cycles, there will be a recount of 34 (TC4Dx) / 32 (all other devices) CCLK cycles.

Table 1685 List of protected bits

Register	Bit-field(s)
SCUSCR_CLK	DIV_REQ
SCUSCR_RSTCON	SWRSTREQ

(table continues...)

Table 1685 (continued) List of protected bits

Register	Bit-field(s)
SCUSCR_STBY	PMSWKPCLR, STBY0REQ
WDTSCR_CON	WDTEN
RTCSCR_CNTn (n=0-3)	CNT_VAL bit-fields in all CNT0-3 registers

39.3.2.4 Memory control unit

The Memory Control Unit (MCU Module) is the interface between the CPU and the memories, providing three functions to the MemOrg:

- Active memory map selection
- Allows one physical memory (that is XRAM) to be accessed as both program and data memory at the same time
- Memory protection

39.3.2.4.1 Memory protection unit

The SCR consists of the following memory blocks:

- Boot ROM
- XRAM

The target of the memory protection scheme is to prevent unauthorized read out of critical data from the Boot ROM.

The memory protection scheme is hardware-based and is used to protect the Boot ROM. MOVC read instructions executed out of the “unsafe” program memory address range (for example XRAM) that target the Boot ROM are blocked when protection is enabled.

39.3.2.5 Arbiter

The arbiter serves the following purposes:

- Providing access to the XRAM through the TriCore™ domain System Peripheral Bus (SPB), using an FPI slave interface (over PMS address space)
- Arbitrates the XRAM access between XC800 CPU and TriCore™ CPU (access through the peripheral bus). SCR is given higher priority to access the XRAM

39.3.2.5.1 Arbiter logic

The Arbiter is clocked by either undivided 100 MHz clock (if it is available), or the 70 kHz clock.

The arbiter always grants the SCR higher priority when accessing the XRAM. During a read from the XC800 core, the output from the XRAM is latched until the value is used by the XC800 core. During a write, the data is simply written to the XRAM at the next Arbiter clock edge.

39.3.3 SCR firmware (SCR FW)

The SCR Boot ROM firmware provides the following features:

- Start-up procedure for stable operation of the SCR
- Basic debugging capability through OCDS mode

Start-up procedure initializes the SCR to a known state to ensure stable operation. The start-up procedure consists of all the settings and initialization to be done after SCR reset, before branching to various boot modes. On-Chip Debug Support provides basic debugging capability to the user.

Related information

Debug information on page 6847
Stand-by controller firmware on page 8797

39.3.4 System Control Unit (SCUSCR)

The System Control Unit (SCUSCR) supports all central control tasks in the SCR. The SCUSCR includes the following functions:

- SCR clock system and control
- SCR reset control
- SCR power management
- System wake-up and standby control
- Limited access to RTC, DTS and HPOSC blocks located inside PMS, for checking status and calibration

39.3.4.1 Boot and start-up

SCR boot mode is configured over PMS_SCR_CON0.SCRCFG bit-field. After first power-up, default value of SCR boot mode is 00_H– User mode, but SCR is not enabled (default value of PMS_SCR_CON0.SCREN=0). SCRCFG boot mode is applied when SCR is first enabled (by setting PMS_SCR_CON0.SCREN to 1 for the first time) and it is also applied after every consecutive SCR main reset while SCR remains enabled. A main reset to the SCR can be triggered by a PMS_SCR_CON1.SCRSTREQ bit-field or by PORST if enabled through PMS_SCR_CON0.PORSTREQ. However, in order to request the SCR main reset, SCR needs to be enabled first through PMS_SCR_CON0.SCREN.

In case SCR was enabled and then disabled, an internal clock sequencing is started to reduce the frequency of the system clock. During this sequencing the SCR continues with the code execution. Upon next enable SCR continues from the instruction, it has been stopped. If SCR was disabled during an ongoing SCR reset, the exact state of SCR cannot be guaranteed and SCR must be reset upon next enable.

Table 1686 SCR Boot modes

SCRCFG value	Boot mode	Descriptions
00_H (default)	User mode	After start-up routine, User Code in XRAM at address 0000_H will be executed. If user code is not loaded to XRAM before enabling SCR, the magic-pattern check during start-up routine will fail, SCUSCR_SCRINTEXCHG code 81_H will be sent to the TriCore™ domain, and the SCR will enter an endless loop.
01_H	OCDS modes	Debug mode using the TriCore™ domain - SPD or DAP interface
02_H		Debug mode using SCR DAP0_0 pin and DAP1_0 pins
03_H		Debug mode using SCR DAP0_1 pin and DAP1_1 pins
04_H		Debug mode using SCR SPD_0 pin
05_H		Debug mode using SCR SPD_1 pin
06_H-07_H	Reserved	-

At the end of the start-up routine by SCR firmware (SCR_FW), an interrupt is triggered to the main controller by setting bit IRSCR_NMICON.SCRINTTC0 to 1, and storing a value in SCUSCR_SCRINTEXCHG register:

- 80_H - there were no errors during start-up
- 81_H - in User-mode, magic-pattern is not matching (possibly due to missing user code in XRAM)

***Note**: Changing of boot mode during SCR operation or when SCR is disabled will have no effect. Changed value will only be applied on the next SCR main reset.*

Illustration of a normal boot and operation of SCR is given in the figure below. It shows several example interactions between the TriCore™ domain application software, SCR firmware and SCR application software.

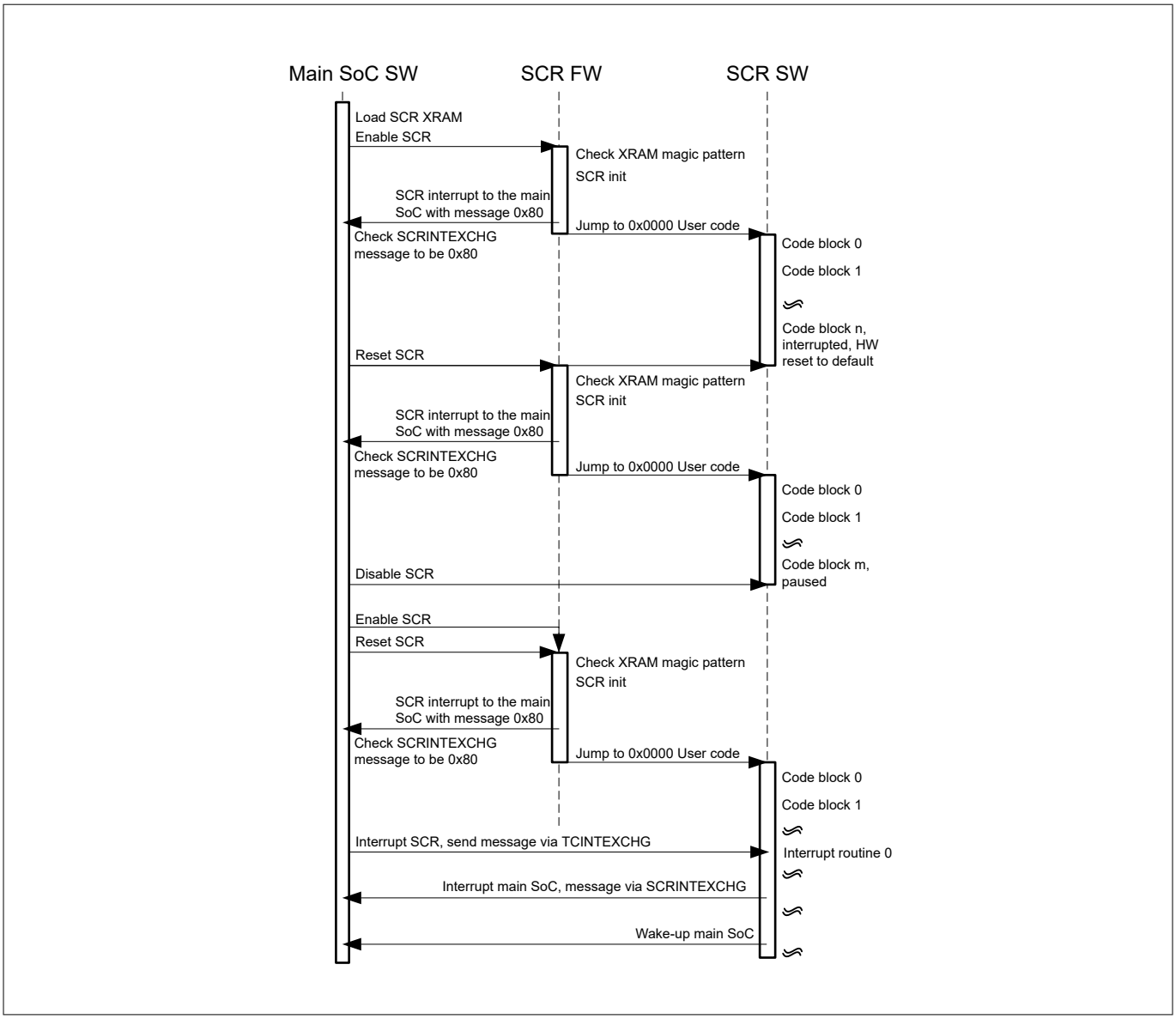


Figure 846 SCR boot and operation time diagram

39.3.4.1.1 Boot ROM operating mode

After a reset, the XC800 CPU will start to execute the Boot ROM code located at the memory address space 0000_H–$07FF_H$. The Boot ROM start-up procedure will first switch the address space for the Boot ROM to $D000_H$–$D7FF_H$. The remaining Boot ROM start-up procedure will be executed from $D60X_H$. The memory organization of the SCR shown in this document is after the Boot ROM address switch where the different operating modes are executed.

User mode

User Mode is the default boot mode after a Low-Voltage Detector (LVD) reset of SCR.

At XRAM addresses $7FF8_H$ through to $7FFF_H$ the "magic" pattern 55_H,AA_H,55_H,AA_H,55_H,AA_H,55_H,AA_H, must be stored. It is used by SCR_FW as check-condition that valid program code is available in the XRAM. Once the SCR starts-up in user mode:

- If the memory content does not match the "magic" pattern user code will not be executed and the SCR will enter an endless loop. Before entering the endless loop, the start-up code will trigger an interrupt to the

TriCore™ domain controller by setting bit IRSCR_NMICON.SCRINTTC0 to 1 with a value of 81_H in SCUSCR_SCRINTEXCHG register

- If there is a match, the same interrupt is triggered to the main controller with a value of 80_H in SCUSCR_SCRINTEXCHG register to indicate the successful start-up. Then SCR firmware ends and user code starts executing from address 0000_H

***Note**: In case of wake-up from STANDBY0 to STANDBY1 mode, the interrupt to the TriCore™ domain will still be issued but main TriCore™ domain controller will not be able to check it until the next RUN mode.*

OCDS mode with SCR DAP/SPD pin

If the OCDS mode with SCR DAP/SPD pin is selected, the debug mode will be entered for debugging the SCR program code. The OCDS hardware is initialized and a jump to program memory address 0000_H is executed. The user code in the XRAM memory is executed and the debugging process may be started.

***Note**: The DAP/SPD pin must be directly accessible by SCR. The control of these pins must be given to SCR before enabling this boot mode. The details on how to give control to the SCR are explained in the PORT8 GPIO section further below.*

OCDS mode with main TriCore™ domain DAP/SPD pin

If the OCDS mode with the TriCore™ domain DAP/SPD pin is selected, the debug mode will be entered for debugging the SCR code using the TriCore™ domain DAP/SPD pins that are also used by the main debug system. The OCDS hardware is initialized and a jump to program memory address 0000_H is executed. The user code in the XRAM memory is executed and the debugging process may be started.

39.3.4.2 Clock system and control

The figure below shows the block diagram of the SCR's clock system.

The two oscillators,

- a 100 MHz oscillator (High-Precision Oscillator, HPOSC)
- and a 70 kHz oscillator (Low-Power Oscillator, LPOSC),

are supplied from inside the PMS.

If SCR is not enabled (PMS_SCR_CON0.SCREN='0') these two clock sources are gated for SCR in both RUN and STANDBY1 mode.

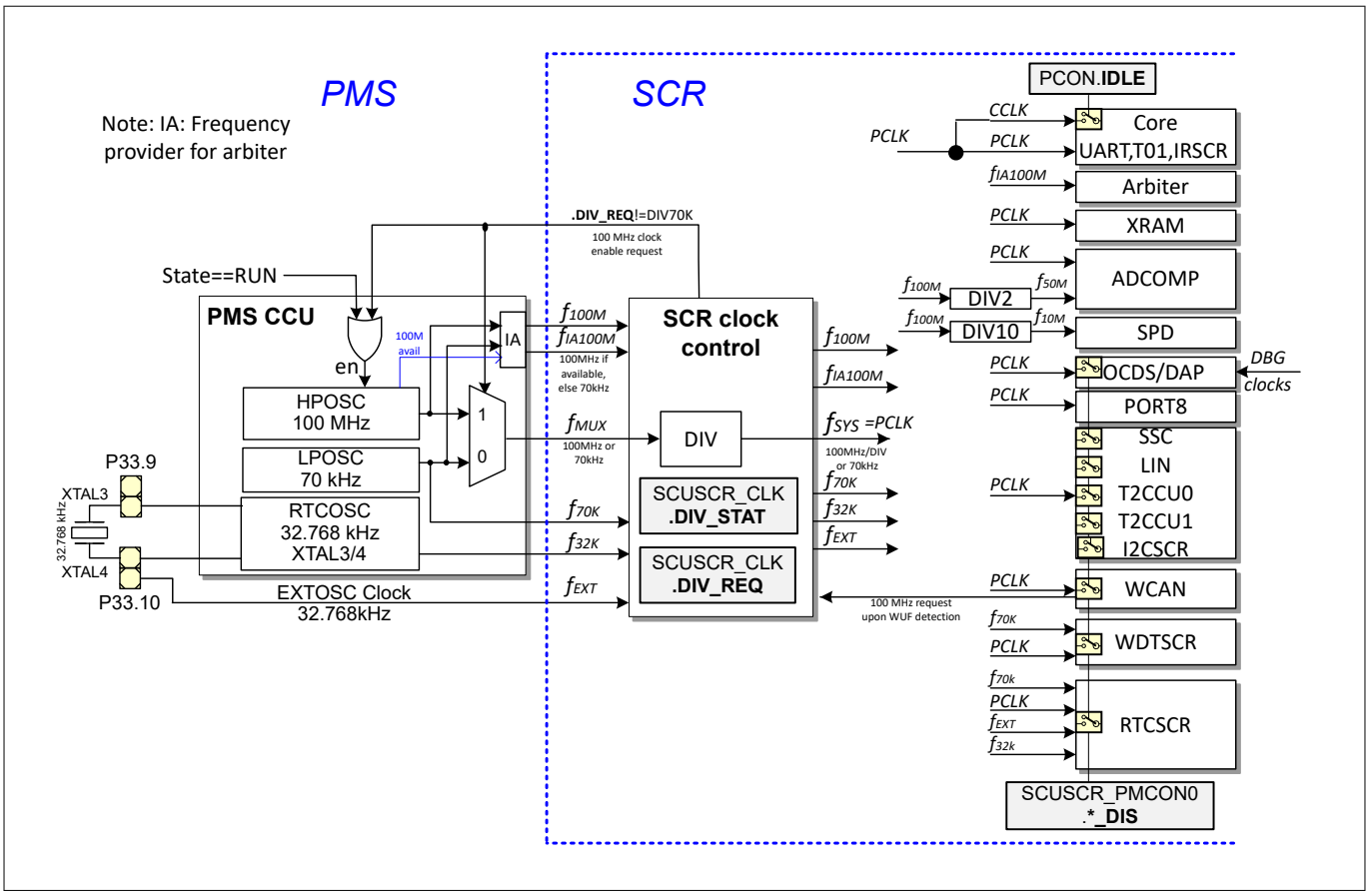


Figure 847 Clock system block diagram

During RUN mode, the 100 MHz clock from HPOSC is always enabled and serves as a backup clock (f_{BACK}) for the TriCore™ domain. During STANDBY0 mode, in order to save power, HPOSC is disabled and only the LPOSC 70 kHz clock is running. In STANDBY1 mode HPOSC is enabled if SCR_SCUSCR_CLK.DIV_REQ!=DIV70kHz. If SCR_SCUSCR_CLK.DIV_REQ==DIV70KHz, HPOSC will be switched off.

When enabled, SCR can select either 70 kHz or some derivative of 100 MHz clock (100 MHz divided by DIV) as its system clock f_{SYS}. This is independent from the TriCore™ status – even when system is in RUN state and 100 MHz oscillator is active, SCR can select 70 kHz as its system clock. However, if in STANDBY1 state HPOSC was not enabled, SCR request to use 100 MHz divided by DIV as a system clock triggers enabling the HPOSC. After an SCR reset default value of f_{SYS} is 20 MHz (DIV=5).

In addition to software request through DIV_REQ, WCAN module can also request 20 MHz clock upon detection of wake-up pattern (WUP).

Beside two internal oscillators (70 kHz and 100 MHz), two other clock sources are supplied to SCR RTCSCR unit:

- 32 kHz crystal oscillator XTAL3/4, forwarded from PMS (P33.9-10), or
- External oscillator coming from pin SCR_P1.2 (P33.10)

The figure above shows the clock source selection and ability of an independent clock gating for each sub-unit inside the SCR.

In active mode, the main module clocks are as follows:

- CPU clock (CCLK)=70 kHz or 100 MHz divided by DIV (default reset value DIV=5 results in 20 MHz clock) = f_{SYS}
- Peripheral clock (PCLK)= f_{SYS}

Most of the peripherals are running on the PCLK clock, for example UART/LIN, T0, T1, T2CCU. There are some exceptions:

- SPD Debug interface can only be supported when the 100 MHz oscillator is available. When it is available, SPD runs on 100 MHz/10 = 10 MHz clock frequency
- ADCOMP also requires that 100 MHz oscillator is available prior to starting measurements. When 100 MHz clock is available, the analog part of ADCOMP runs at 100 MHz/2 = 50 MHz clock frequency. Configuration and status registers of ADCOMP are clocked with PCLK
- Besides running on PCLK, clock watchdog and RTCSCR can select to run at 70 kHz at any time. Additionally, RTCSCR can also run on XTAL3/4 (P33.9-10, forwarded from PMS) or external clock provided from pin SCR_P1.2 (P33.10)
- Arbiter will run on undivided 100 MHz clock if HPOSC is enabled, otherwise it is running on 70 kHz. This is independent from divider DIV setting (for example, if DIV = 5 the arbiter still operates on 100 MHz)
- OCDS/DAP, besides running on PCLK clock, it can also run on the other clocks that are provided with the Debug system (clock from pads, SPD clock or external I/O-Client clock, depending on the mode)

Clock division and sequencing

If 100 MHz clock source is selected for f_{SYS}, the clock will be divided by the programmable factor DIV that is reflected through bit-field SCUSCR_CLK.DIV_STAT.

Transition from 70 kHz clock to higher frequencies is done step-wise to avoid sudden supply current fluctuations. First step is from 70 kHz to 10 MHz (DIV=10) followed by one DIV step at a time (one step every ~15.3 µs), if higher frequencies are requested. Same step-wise approach applies on the way back, or on any change between two frequencies, where at least one of them is higher than 10 MHz.

The following shows an exemplary time schedule, switching from 70 kHz clock to 100 MHz clock:

- Set division request value to divide clock by 1 (SCUSCR_CLK.DIV_REQ = DIV1), which corresponds to the frequency of 100 MHz
- Start of HPOSC is triggered. After around 200 µs, HPOSC is ready to be used
- Intermediate divided value of f_{SYS} is set to 10 MHz and status bit-field becomes SCUSCR_CLK.DIV_STAT = DIV10.
- Frequency is then increased step-wise: 10 MHz → 12.5 MHz → 20 MHz → 25 MHz → 33 MHz → 50 MHz → 100 MHz – total of 6 transitions, taking ~92 µs. Final value of SCUSCR_CLK.DIV_STAT = DIV1 which matches the DIV_REQ requested value

It can take around 300 µs to have SCR system clock reach target frequency. Nevertheless, XC800 CPU can keep executing further instructions without waiting for final clock frequency to settle.

When the SCR is disabled from TriCore™ software via SCR_CON0.SCREN=0 the SCR still remains enabled until the PCLK frequency is reduced to a frequency ≤ 20 MHz. This frequency reduction is done step-wise. For example if the PCLK is running at 100 MHz there will be 5 steps (100 MHz → 50 MHz → 33.3 MHz → 25 MHz → 20 MHz → 12.5 MHz) before the SCR gets disabled. Every step takes typically around 15.3 us. If the PCLK is running at any frequency < 20 MHz the SCR will be disabled immediately.

39.3.4.2.1 Module suspend control

During an active debug session timers in certain modules in the SCR can be suspended based on the settings of their corresponding module suspend bits in register SCUSCR_DBG_MODSUSP. When suspended, only the timer

stops counting as the counter input clock is gated off. The module is still clocked so that module registers are accessible.

39.3.4.3 Reset control

The SCR has three types of resets:

- LVD Reset – power reset caused by an Low-Voltage Detector (LVD) circuit
- Main Reset – SCR reset coming from the main SoC through PMS_SCR_CON1.SCRSTREQ or SCR-PORST pin or Debug reset
- Generated Reset – reset from sources within the SCR

The SCR Watchdog Timer Reset, the RAM double bit ECC Reset, the reset from Pin and the Soft Reset which have the source within the SCR are called Generated Resets. For the generated resets the following applies:

- The Watchdog Timer (WDT) module is capable of resetting the device if it detects a timer overflow (which represents malfunction in the system)
- A double bit ECC event in the SRAM can also trigger a reset
- A low value on the selected pin (after digital glitch filtering) can also trigger a reset
- A Software reset is also possible through write to designated bit-field

39.3.4.3.1 Types of reset

The following reset types affect the SCR:

- **LVD reset**, caused by an under-voltage detection
  - Triggered asynchronously during power failure (under-voltage on PMS1 domain)
  - OCDS/Debug system is reset when the LVD reset occurs
- **Main reset**, reset from the main SoC
  - SCR-PORST triggered by the PORST pin, if enabled using PMS_SCR_CON0.PORSTREQ triggering a main reset. Also OCDS/Debug system is reset by SCR PORST
  - This is a combined reset, consisting of all the Power reset sources and synchronous reset by the main SoC controller CPUx through bit PMS_SCR_CON1.SCRSTREQ
  - Debug reset is also triggering a main reset
  - Main reset also activates the generated reset
  - Resets all registers including the reset indication bits
- **Generated reset**, reset from sources within the SCR
  - Watchdog Timer (WDT) reset
    - Requested by WDT overflow or out of boundary refresh event if it is enabled through SCUSCR_RSTCON.WDTRSTEN
    - Resets all registers except the reset indication bits
    - Reset can also trigger an interrupt to the main SoC (interrupt feature is available independent from the configuration of this reset as a reset source)
  - RAM Double-Bit ECC reset
    - Requested by ECC event if it is enabled through SCUSCR_RSTCON.RAMECCRSTEN
    - Resets all registers except the reset indication bits
    - Reset can also trigger an interrupt to the main SoC (interrupt feature is available independent from the configuration of this reset as a reset source)
  - Reset from one of the reset pins:
    - Requested by low value of selected pin-reset input if it is enabled through SCUSCR_RSTCON.PINRSTEN

- Resets all registers except the reset indication bits
- If the reset is enabled it can also trigger an interrupt to main SoC

- Software triggered reset:
  - Requested by soft reset event through SCUSCR_RSTCON.SWRSTREQ
  - Resets all registers except the reset indication bits
  - Reset can also trigger an interrupt to the main SoC

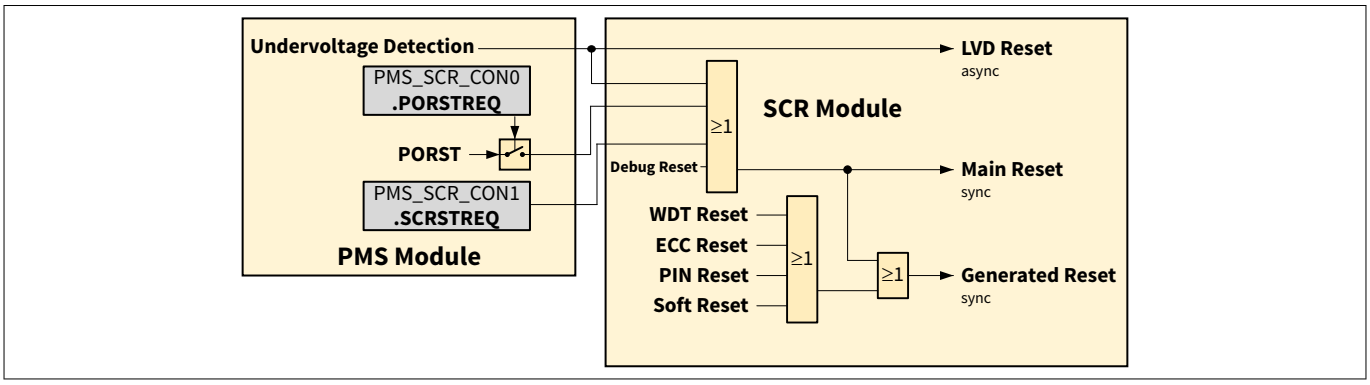


Figure 848 Simplified reset overview

LVD reset (under-voltage reset)

The LVD Reset, resets the complete SCR including the SRAMs and the Debug system.

An LVD Reset will also trigger a Main Reset, which will trigger a Generated Reset.

Main reset (external reset)

The main reset includes PORST and a software triggered reset from the main core (TriCore™) through the PMS_SCR_CON1.SCRSTREQ bit. A main reset will reset the digital logic around SRAMs. Main reset which is caused due to a PORST and will reset the debug system.

A main reset also triggers the generated reset.

If the value of PMS_SCR_CON0.SCRCFG bit-field is changed and, if SCR was already enabled prior to this change, triggering a main reset through the SCRSTREQ will re-initiate SCR from the newly given boot mode.

A debug reset is also triggering a main reset.

Generated reset (SCR reset)

Generated Reset is activated together with Main Reset, or on detection of a Watchdog Timer overflow event, an SRAM ECC error event, a PIN reset, a soft reset or a debug reset.

Same events that lead to generated SCR reset, except the Debug reset, are reported to the TriCore™ domain over PMS status flags which further generate an interrupt to the TriCore™ domain interrupt controller, as shown in following figure:

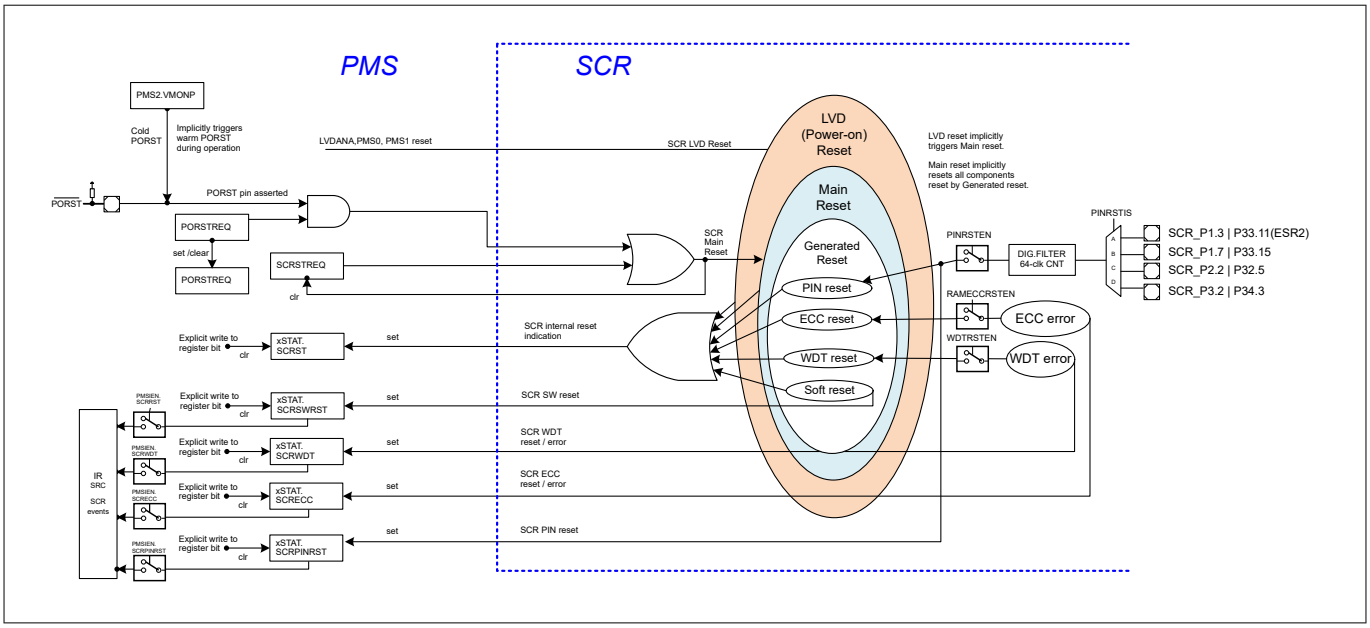


Figure 849 Detailed SCR reset diagram

Generated Reset will not change SCR boot mode, even if it was changed prior to this reset. Only Main Reset will trigger re-latching of PMS by the given SCR boot mode.

***Note**: SCR cannot change its own boot mode, this change has to come from the main CPU.*

Watchdog timer reset

The watchdog timer reset is an internal reset and can be disabled by bit SCUSCR_RSTCON.WDTRSTEN. The Watchdog Timer (WDT) maintains a counter that must be refreshed or cleared periodically. If the WDT is not serviced correctly and in time, it will generate an NMI request to the CPU and then reset the device (WDTSCR_RST) after a predefined time-out period if this reset type is enabled. When the reset is disabled, no reset will be triggered when the overflow event or out of boundary refresh event happens.

Bit SCUSCR_RSTSTAT.WDTRST is used to indicate the watchdog timer reset status.

***Note**: The event status SCUSCR_RSTSTAT.WDTRST can be cleared by Main Reset.*

An event signal will be generated as an output of SCR to indicate that a watchdog timer event happened. This signal is latched by the PMS and the error status can be read through PMS_SCR_STAT.SCRWDT bit.

Double-bit ECC reset

Double-bit ECC error can happen on read access to XRAM..

The double bit ECC reset is an internal reset and can be disabled by bit SCUSCR_RSTCON.RAMECCRSTEN (not advised). The data loaded by TriCore™ is encoded with an Error Correction Code (ECC) before being stored in the XRAM memory. During a read access, data is retrieved from the memory and decoded for dynamic error detection and correction. If a double-bit error is detected, it will generate a reset if this reset type is enabled. When the reset is disabled, no reset will be triggered when the ECC event happens.

Bit SCUSCR_RSTSTAT.RAMECCRST is used to indicate the ECC reset status.

***Note**: The event status SCUSCR_RSTSTAT.RAMECCRST can be cleared by Main Reset.*

An event signal will be generated as an output of SCR to indicate that an ECC event happened. This signal is latched by the main PMS and the error status can be read through PMS_SCR_STAT.SCRECC bit.

Reset from pin

A reset from pin is an internal reset that is caused by low level on selected PINRST pin and can be enabled by bit SCUSCR_RSTCON.PINRSTEN.

Digital glitch filtering is applied with simple 6-bit up-counter (63 clock cycles) that runs on PCLK clock – selected input signal has to be low for at least 63 consecutive PCLK cycles to trigger a reset request. This means If there is a bouncing signal on the input the low value shall be stable for at least 63 cycles to be recorded. This filtering feature is always enabled and cannot be disabled.

If a reset from pin request is detected, it will generate a reset if this reset type is enabled. Before enabling this reset make sure that selected input signal is not already low, as this can unintentionally trigger a reset soon after enabling.

Bit SCUSCR_RSTSTAT.PINRST is used to indicate the pin reset status.

***Note**: The event status SCUSCR_RSTSTAT.PINRST can be cleared by Main Reset.*

Bit-field SCUSCR_MODPISEL1.PINRSTIS is used to select one of the four possible pin reset inputs:

- PINRSTA - SCR_P1.3 (P33.11, ESR2) - this is the default option
- PINRSTB - SCR_P1.7 (P33.15)
- PINRSTC - SCR_P2.2 (P32.5)
- PINRSTD - SCR_P3.2 (P34.3)

An event signal will be generated as an output of SCR to indicate that a pin reset event happened. This signal is latched by the PMS and the reset status can be read through PMS_SCR_STAT.SCRPINRST bit.

Soft reset

Soft reset is an internal reset that is caused by a software set of the soft reset request bit inSCUSCR_RSTCON.SWRSTREQ.register. It can be triggered by application software when necessary.

Bit SCUSCR_RSTSTAT.SWRST indicates the soft reset status.

An event signal will be generated as an output of SCR to indicate that a soft reset event happened. This signal is latched by the PMS and the reset status can be read through PMS_SCR_STAT.SCRRST bit.

***Note**: The event status SCUSCR_RSTSTAT.SWRST can be cleared by Main Reset.*

Debug reset

A debug reset is a generated reset that is caused by debugger reset commands. Status on debug reset is not reported to the TriCore™ domain.

Debug reset is also triggered whenever the LVD reset or PIN reset is triggered. Main reset will trigger the debug reset only if the source of main reset is PORST. software triggered main reset from the TriCore™ will have no effect on the debug system.

39.3.4.3.2 Module reset behavior

Following table gives an overview on how the various modules or functions of the SCR are affected with respect to the reset type. An “X” means that the module/function is reset to its default state.

Table 1687 Effect of reset on modules/functions

Module/function	LVD reset	Main reset	Generated reset (WDT, ECC, PIN, SOFT)
XC800 core	X	X	X
SCUSCR	X	X	All logic is affected except the reset indication bits

(table continues...)

Table 1687 (continued) Effect of reset on modules/functions

Module/function	LVD reset	Main reset	Generated reset (WDT, ECC, PIN, SOFT)
Peripherals	X	X	X
Debug system	X	Affected only by PORST	Affected only by PIN reset
Port control (SCR PORT8 side)	X	X	X
On-chip static RAM	Affected, unreliable	Not affected, reliable	Not affected, reliable
Clock system	X	X	X

SCUSCR_RSTCON register contains enable bits for an ECC, WDT, PIN and soft reset. SCUSCR_RSTSTAT consists of reset indication bit-fields for the same four generated reset sources. SCUSCR_RSTSTAT can be cleared by software or by higher order reset (Main or LVD). Table below shows the value of SCUSCR_RSTSTAT register after these events. An "X" means value is not affected by that reset event.

Table 1688 SCUSCR_RSTSTAT value after reset events

Reset source	Reset value
RAM double-bit ECC reset	$000X_0XX\mathbf{1}_B$
WDT reset	$000X_0X\mathbf{1}X_B$
PIN reset	$000X_0\mathbf{1}XX_B$
Soft reset	$000\mathbf{1}_0XXX_B$
Main reset	$000\mathbf{0}_0\mathbf{000}_B$

39.3.4.4 SCR power management

This section describes the features and functionality provided for power management of the SCR.

Standby mode is one of the power saving modes in the main SoC. System Standby mode can be split in two distinctive modes, STANDBY1 and STANDBY0. During a STANDBY1 mode, SCR is the only CPU subsystem active (TriCore™ CPUs are switched off). During a STANDBY0 mode, even SCR is switched off and just a few peripherals in PMS are active waiting for a wake-up signal.

This section also describes:

- How to exit the Standby Mode
- How to operate the whole chip between two low power modes, STANDBY1 and STANDBY0

These points are described from SCR point of view.

More details on system Standby modes and transitions between different modes is presented in PMS chapter called "Standby Mode Management".

39.3.4.4.1 Overview

The SCR provides software programmable actions to minimize the power consumption.

There are generally two power modes of the SCR subsystem: Active Mode and Idle Mode.

The operation of the system components in each of these two modes can be configured by software. The power modes provide flexible reduction of power consumption through a combination of techniques, including:

- Stopping the CPU clock
- Stopping the clocks of other components individually
- Clock-speed reduction of some peripheral components

In Active mode, the CPU and peripherals are running at system frequency f_{SYS}.

In Idle mode, the SCR core clock (CCLK) is disabled and the Core halts. Peripherals with separate input clock (PCLK) are not disabled.

***Note**: The watchdog timer may generate an warning interrupt (followed by a reset) when an overflow is about to occur and this will disrupt the idle mode. The idle mode is terminated automatically when any enabled interrupt signal is detected.*

39.3.4.4.2 SCR idle mode

Software requests Idle Mode by setting the bit PCON.IDLE to 1.

Based on the IDLE request by the SCR software the CPU finishes its current operation and sends an acknowledge signal (IDLE) to the SCUSCR. Based on the feedback signal the SCU disables the CPU clock (CCLK).

The SCR core returns to the Active Mode when there is an incoming interrupt request to an enabled interrupt node.

An interrupt to the XC800 CPU causes the CCLK to recover. Upon RETI instruction, the CPU will return to execute the next instruction following the instruction which sets the PCON.IDLE bit to 1.

***Note**: The user is advised to disable watchdog timer prior to entering the Idle Mode and use the RTCSCR interrupt as preferred way to schedule wake-up from the Idle Mode.*

39.3.4.4.3 Peripheral clock disable

It is possible to disable the clock for some peripherals separately in order to save power. Each bit of SCUSCR_PMCON0 register controls one peripheral. When the bit is set, the request signal to gate off the clock to

the respective module is asserted. The SCUSCR will then synchronize the gating off of the clock to the peripheral. Upon power on reset (LVD reset) clock inputs of all peripherals as specified in the SCUSCR_PMCON0 register are gated.

39.3.4.4.4 Exit the TriCore™ domain standby mode through SCR

In the system STANDBY1 mode, SCR is operating in either Active Mode or Idle Mode. Therefore, SCR could be one of the wake-up sources for the TriCore™ domain to exit the system Standby mode and transition into a system Run mode. Before issuing the standby request and entering the system Standby mode, the followings steps related to SCR need to be set:

- Configure the wake-up signals related to SCR that trigger the exit from the Standby mode through PMS_WAKEUP_CON1.SCRWKEN bit-field
- Configure the clock selection for SCR in Standby Mode through SCUSCR_STBY.SCRCLKSEL bit

To exit from the Standby Mode, one of the following wake-up sources from SCR could be used:

- RTCSCR compare match interrupt (this is 41-bit RTCSCR located in SCR)
- WCAN Interrupts that include Wake up Frame, Wake up Pattern, System Error and CAN time-out
- Watchdog (WDTSCR) overflow event
- SRAM ECC double bit error event
- Software request through SCUSCR_WKPCON.SWWKPREQ

Besides the specified events such as WDT overflow event, SRAM ECC error, WCAN event or RTCSCR compare match event, other types of events from SSC, T0, T1, T2CCU and UART could be implicitly used to issue a wake-up request through the bit SWWKPREQ in SCUSCR_WKPCON register.

The status bit SCUSCR_STBY.STBYSTAT indicates to SCR software whether the TriCore™ domain is currently in Standby mode or not.

If SCR software and software running on the main cores (TriCore™) desire to exchange information upon wake-up it can be done in two ways:

- Through 8-bit exchange registers: SCUSCR_SCRINTEXCHG and SCUSCR_TCINTEXCHG
- Through XRAM memory that both SCR core and TriCore™ domain cores can access

39.3.4.4.5 Low power operation with STANDBY1 and STANDBY0 modes

In order to achieve low average power, while still supporting functional code to run, system can utilize switching between two low-power standby modes:

- STANDBY1 - low power mode where the main SoC is switched off and SCR subsystem is active
- STANDBY0 - low power mode where the whole system is switched off except few peripherals waiting for a wake-up signal

This represents a special low-power operation where the system is most of the time be in STANDBY0 state, and only waking up to STANDBY1 state for a short time, finishing necessary tasks (for example temperature calibration), and switching back to STANDBY0 state.

To enable this special low-power operation, the main SoC software needs to take several setup steps:

- Enable that STANDBY1 → STANDBY0 transitions can be requested by SCR software
  - This is done through PMS_STANDBY_CON0.SCRSTBY0EN = 1
- Configure which wake-up sources will trigger STANDBY0 → STANDBY1 transitions
  - More details on available wake-up sources is presented in PMS chapter Standby Mode Management

- Configure that chosen wake-up event will trigger a transition into the STANDBY1 mode and not into the RUN mode
  - This is done through SMM_PMCSRx.REQSLP = $1x_B$ and PMS_STANDBY_CON1.SBMODSEL = 1
- Enable XRAM memory retention while the system is in the STANDBY0 mode
  - This is done through PMS_STANDBY_CON0.SCRRAMSEL = 1

This settings have to be done first before transitioning from RUN to STANDBY1 mode.

Following diagram illustrates transitions between RUN, STANDBY1 and STANDBY0 mode from SCR control perspective.

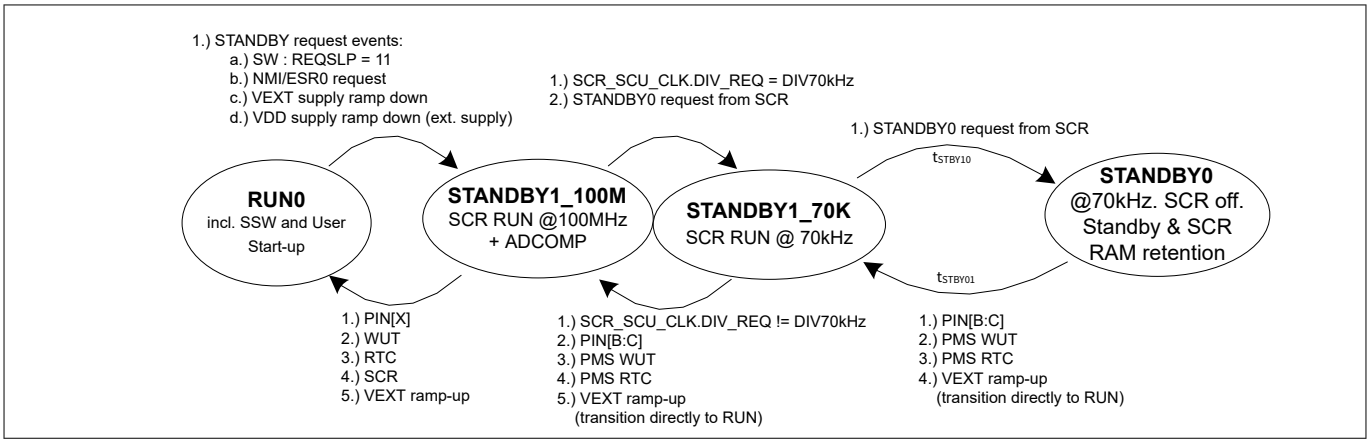


Figure 850 System mode transitions from SCR perspective

***Note**: In most use-cases, optimal power consumption is achieved when alternating between STANDBY1_100M and STANDBY0 states. STANDBY1_70K state has much slower clock and not proportionally lower power consumption comparing to STANDBY1_100M state. Which exact frequency derived from 100 MHz clock (10 MHz, 20 MHz, 50 MHz, 100 MHz or other) gives optimal power depends on exact use-case and is left on user to explore.*

***Note**: STANDBY1_70K mode is useful in case where use-case needs some SCR sources as a system wake-up (for example listening to SCR Port signals), where these sources are not available in STANDBY0 mode. Another useful case for STANDBY1_70K is when wake-ups happen frequently and amount of tasks to finish upon a wake-up is small. In these cases whole operation is alternating between STANDBY1_70K and STANDBY1_70K_IDLE modes, which gives better average power than alternating between STANDBY0 and STANDBY1_100M.*

Enter STANDBY0 mode

STBY0 mode can be entered from the RUN directly or from the STBY1 mode. Transition RUN → STBY0 does not involve SCR subsystem and is explained in PMS chapter "Standby Mode Management".

Transition from STBY1 to STBY0 involves SCR and is described below.

In order to trigger a STBY1 → STBY0 transition following conditions have to be fulfilled:

1. None of the PMS wake-up flags and over-run flags are set

2. There was a STBY1 → STBY0 transition trigger:

a. SCR software has triggered a STBY1 → STBY0 transition (through SCR_SCUSCR_STBY.STBY0REQ = 1, if enabled by PMS_STANDBY_CON0.SCRSTBY0EN = 1), or

b. Hardware condition has triggered a STBY1 → STBY0 transition (for example VDDEXT ramp-down if VDDEXTSBEN = 1)

In other words, if PMS wake-up flags are cleared both SCR software and hardware trigger will cause a transition of the system into STBY0 mode. Clearing of PMS wake-up and over-run flags can be done from SCR side through SCR_SCUSCR_STBY.PMSWKPCLR = 1

***Note**: If some hardware condition for STBY0 entry is already fulfilled (for example VDDEXT is already powered down) then clearing PMS flags from SCR side will directly be followed by transition into STBY0 - no need to explicitly trigger it by the SCR SW. Therefore, the SCR software routine should keep the instruction for clearing of the PMS wakeup-flags at the end of the processing loop.*

If shared ports were granted to the SCR (by setting respective Pn_PCSRSEL and SCREN = 1) while in RUN mode, the ownership will remain on the SCR side when transitioning in STANDBY1 and later in STANDBY0 state. More details on Shared Pads ownership is given in SCR "General Purpose I/O Ports (PORT8)" chapter.

Exit STANDBY0 to STANDBY1 mode

After a wake-up event the system wakes up from STANDBY0 into STANDBY1 if configured through SMM_PMCSRx registers.

After entering the STANDBY1 mode, and SCR boot start-up, SCR software can read the flags STBYSTAT and STBY0WKP in register SCUSCR_STBY to derive if the STANDBY1 entry was from RUN or STANDBY0 state. Based on this information, SCR software can differentiate between the cases:

- STANDBY1 mode was entered from STANDBY0 mode
- STANDBY1 mode was entered from RUN mode

Additionally perform different software routine if necessary.

XRAM keeps its content while in STANDBY0 state. If there is a need to share information between several STANDBY1 sessions which are alternated by STANDBY0 sessions, XRAM memory can be used.

If shared ports were granted to the SCR (by setting respective Pn_PCSRSEL and SCREN='1') while in RUN mode, the ownership will remain on the SCR when transitioning from STANDBY0 to STANDBY1. More details on Shared Pads ownership is given in SCR "General Purpose I/O Ports (PORT8)" chapter.

39.3.5 Interrupt system (IRSCR)

The XC800 Core supports one non-maskable interrupt (NMI) and 14 maskable interrupt requests. In addition to the standard interrupt functions supported by the core, for example, configurable interrupt priority and interrupt masking, the SCR interrupt system provides extended interrupt support capabilities such as the mapping of each interrupt vector to several interrupt sources to increase the number of interrupt sources supported, and additional status registers for detecting and identifying the interrupt source.

39.3.5.1 Interrupt sources

The SCR supports 14 interrupt vectors with four priority levels. These interrupt vectors are assigned to the on-chip peripherals:

- Timer 0, Timer 1, UART, RTCSCR and WCAN are each assigned one dedicated interrupt vector
- T2CCU0 and LIN interrupts have to share an interrupt vector
- SSC, ADC, T2CCU1 and PMS_RTC CMP0 also share one interrupt vector
- External interrupt 0 and 1 are each assigned one dedicated interrupt vector
- External interrupt 2 shares the interrupt vector with PMS_RTC CMP1 and I2CSCR
- remaining 13 external interrupts are shared among 4 interrupt vectors

A non-maskable interrupt (NMI) with the highest priority is shared by the following:

- Watchdog Timer, warning before overflow
- OCDS, on user IMEM event
- RAM ECC error
- External NMI (through rising edge on EXTNMI pin SCR_P1.6/P33.14)
- Main controller: This interrupt is triggered either when software sets PMS_SCR_CON1.TCINTREQ or when SCR_SCUSCR_MRSTST.APPRST or SCR_SCUSCR_MRSTST.SMURST are set
- Wake-up interrupt: This interrupt is triggered when any of the wake-up inputs were triggered, and the VEXT supply did not ramp up even after the blanking time interval. Please see PMS chapter for additional details

Following three figures give a general overview of the interrupt sources and nodes, and their corresponding control and status flags. Figure 'non-maskable Interrupt (NMI) Request Sources' below gives the corresponding overview for the NMI sources.

The details of the peripheral interrupt sources are described in the respective peripheral chapters, while the details of the external interrupt sources can be found in figures in related information.

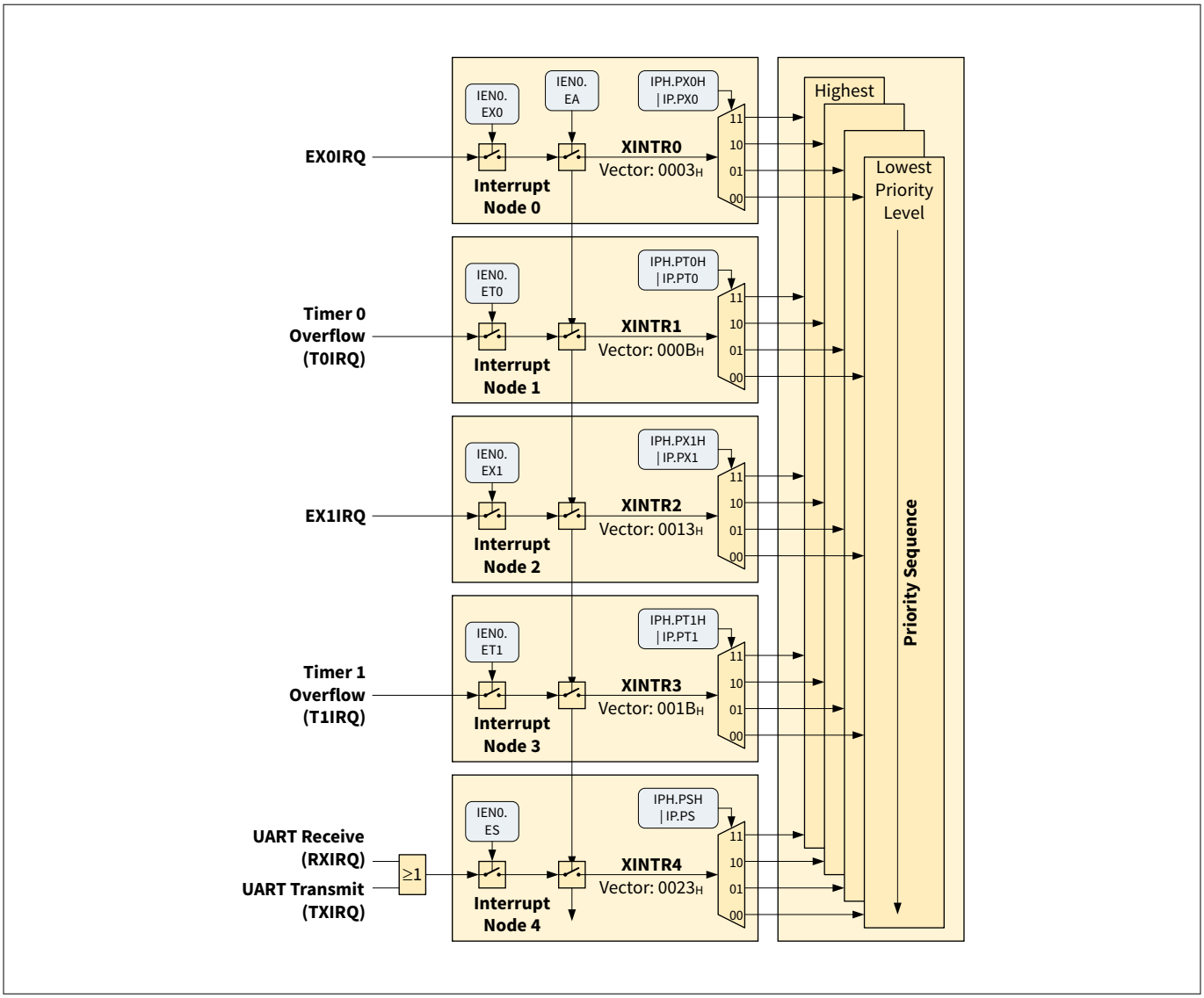


Figure 851 Interrupt request sources and interrupt nodes 0 through 4

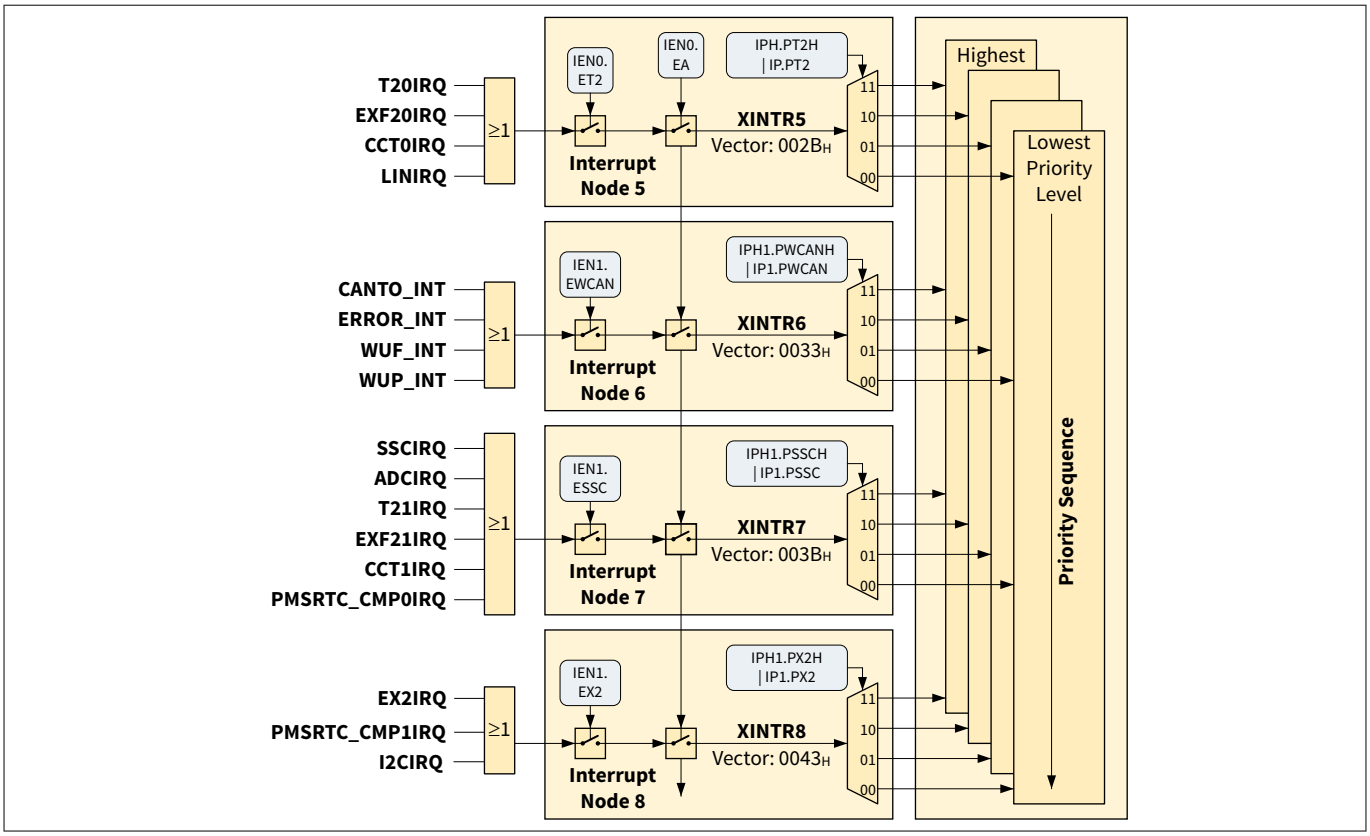


Figure 852 Interrupt request and interrupt nodes 5 through 8

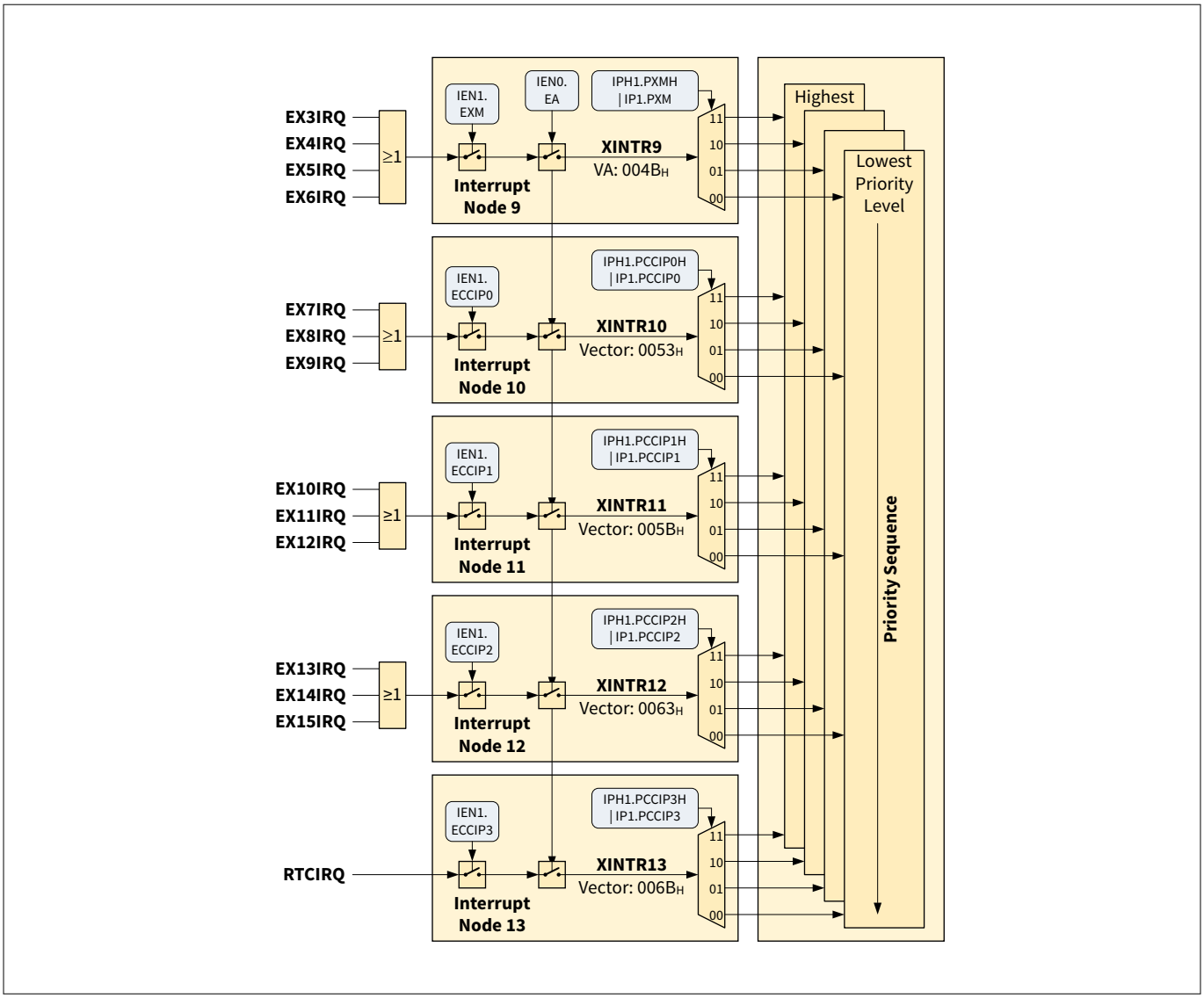


Figure 853 Interrupt request and interrupt nodes 9 through 13

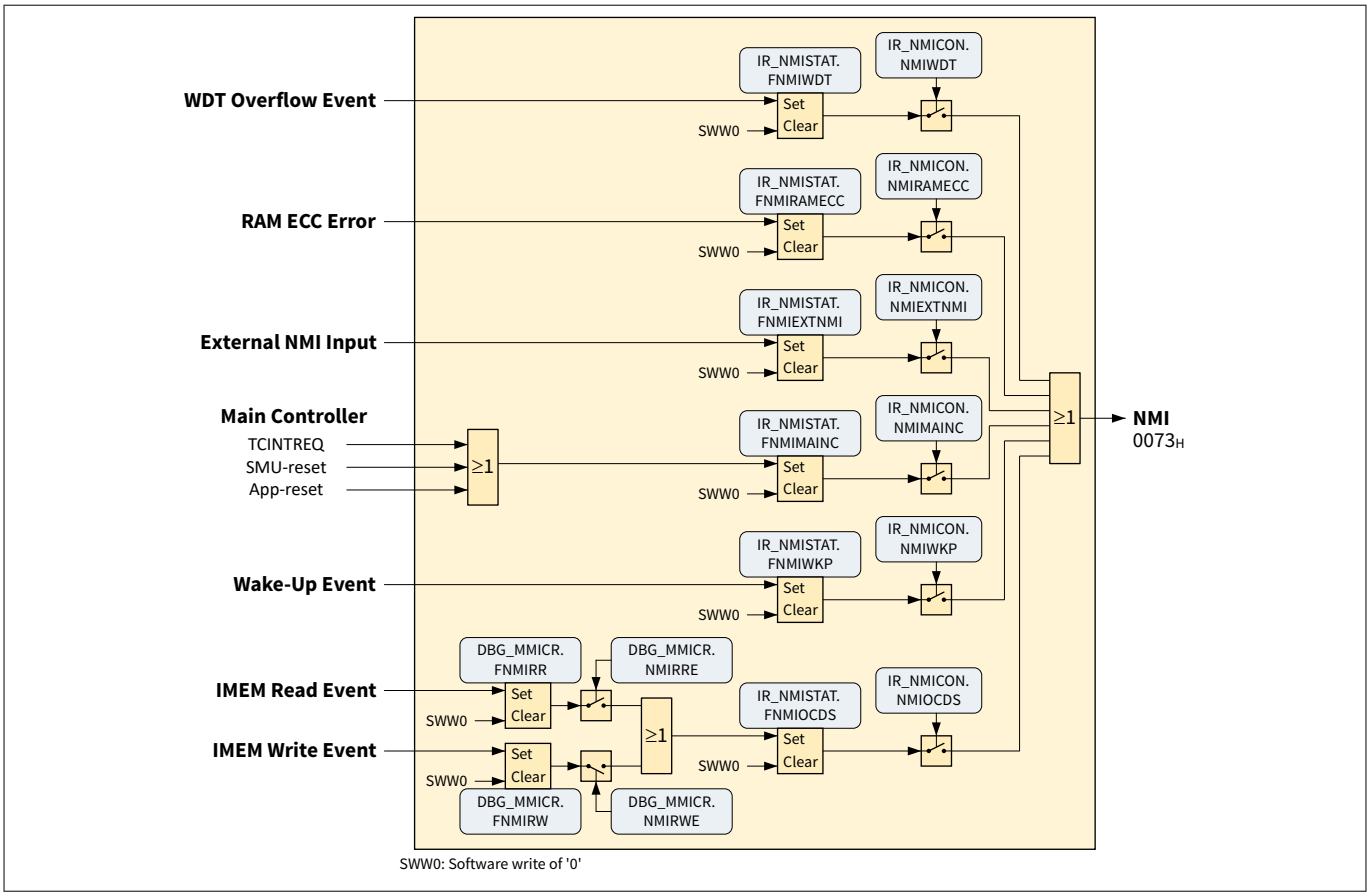


Figure 854 Non-maskable interrupt (NMI) request sources

Related information

Registers overview on page 6531

39.3.5.1.1 Interrupt source and vector of the SCR

Each interrupt event source has an associated interrupt vector address for the interrupt node it belongs to. This vector is accessed to service the corresponding interrupt node request. The interrupt service of each interrupt node can be individually enabled or disabled through an enable bit. The assignment of the SCR interrupt sources to the interrupt vector address and the corresponding interrupt node enable bits are summarized in the following table.

Table 1689 Interrupt vector address

Interrupt node	Vector address	Assignment for SCR	Enable bit	SFR
NMI	0073_H	Watchdog Timer NMI	NMIWDT	NMICON
		RAM Double Bit ECC Error NMI	NMIRAMECC	
		Wake-up event NMI	NMIWKP	
		External NMI through EXTNMI pin	NMIEXTNMI	
		OCDS NMI	NMIOCDS	
		Main Controller NMI	NMIMAINC	
XINTR0	0003_H	External Interrupt 0	EX0	IEN0
XINTR1	$000B_H$	Timer 0	ET0	
XINTR2	0013_H	External Interrupt 1	EX1	
XINTR3	$001B_H$	Timer 1	ET1	
XINTR4	0023_H	UART	ES	
XINTR5	$002B_H$	T2CCU0	ET2	
		LIN		
XINTR6	0033_H	Wake-up CAN filter interrupt (WCAN)	EWCAN	IEN1
XINTR7	$003B_H$	ADCOMP	ESSC	
		SSC		
		T2CCU1		
		PMSRTCSCR_CMP0		
XINTR8	0043_H	External Interrupt 2	EX2	
		PMSRTCSCR_CMP1		
		I2CSCR		
XINTR9	$004B_H$	External Interrupt 3	EXM	
		External Interrupt 4		
		External Interrupt 5		
		External Interrupt 6		
XINTR10	0053_H	External Interrupt 7	ECCIP0	
		External Interrupt 8		
		External Interrupt 9		
XINTR11	$005B_H$	External Interrupt 10	ECCIP1	

(table continues...)

Table 1689 (continued) Interrupt vector address

Interrupt node	Vector address	Assignment for SCR	Enable bit	SFR
		External Interrupt 11		
		External Interrupt 12		
XINTR12	0063_H	External Interrupt 13	ECCIP2	
		External Interrupt 14		
		External Interrupt 15		
XINTR13	$006B_H$	RTCSCR Interrupt	ECCIP3	

39.3.5.1.2 Interrupt source and priority

An interrupt that is currently being serviced can only be interrupted by a higher priority interrupt, but not by another interrupt of the same or lower priority. Hence, an interrupt of the highest priority cannot be interrupted by any other interrupt request.

If two or more requests of different priority levels are received simultaneously, the request with the highest priority is serviced first. If requests of the same priority are received simultaneously, an internal polling sequence determines which request is serviced first. Thus, within each priority level, there is a second priority structure determined by the polling sequence as shown in the following table.

Table 1690 Priority structure within interrupt level

Source	Level/interrupt node (XINTR)
Non-Maskable Interrupt (NMI)	NMI (highest)
External Interrupt 0	0
Timer 0	1
External Interrupt 1	2
Timer 1	3
UART	4
T2CCU0	5
LIN	5
Wake-Up CAN Filter	6
A/D Converter	7
SSC	7
T2CCU1	7
PMSRTC_CMP0	7
External Interrupt 2	8
PMSRTC_CMP1	8
I2C	8
External Interrupt [6:3]	9
External Interrupt [9:7]	10

(table continues...)

Table 1690 (continued) Priority structure within interrupt level

Source	Level/interrupt node (XINTR)
External Interrupt [12:10]	11
External Interrupt [15:13]	12
Real-Time Clock	13

39.3.5.2 Interrupt structure

An interrupt event source may be generated from the on-chip peripherals or from external event. The detection of interrupt events is controlled by the respective on-chip peripherals. The interrupt status flags are available for determining which interrupt event has occurred (see the table above: Priority structure within interrupt level), this is especially useful for an interrupt node which is shared by several event sources. Each interrupt node (except NMI) has a global enable/disable bit. In most cases, additional enable bits are provided for enabling/disabling particular interrupt events (provided for NMI events). No interrupt will be requested for any occurred event that has its interrupt enable bit disabled.

The SCR has, in general, two interrupt structures distinguished mainly by the manner in which the pending interrupt request (one per interrupt vector/node going directly to the core) is generated (due to the events) and cleared.

Common among these two interrupt structures is the interrupt masking bit, EA, which is used to globally enable or disable all interrupt requests (except NMI) to the core. Resetting bit EA to 0 only masks the pending interrupt requests from the core, but does not block the capture of incoming interrupt requests.

***Note**: The NMI node is similar to the other interrupt nodes, except for the exclusion of EA bit. Effectively, NMI node is non-maskable.*

***Note**: Please see also table: Location of the interrupt flags*

39.3.5.2.1 Interrupt structure 1

For interrupt structure 1 (see following figure), the interrupt event will set the interrupt status flag which doubles as a pending interrupt request to the core. An active pending interrupt request will interrupt the core only if its corresponding interrupt node is enabled. Once an interrupt node is serviced (interrupt acknowledged), its pending interrupt request (represented by the interrupt status flag) may be automatically cleared by hardware (the core).

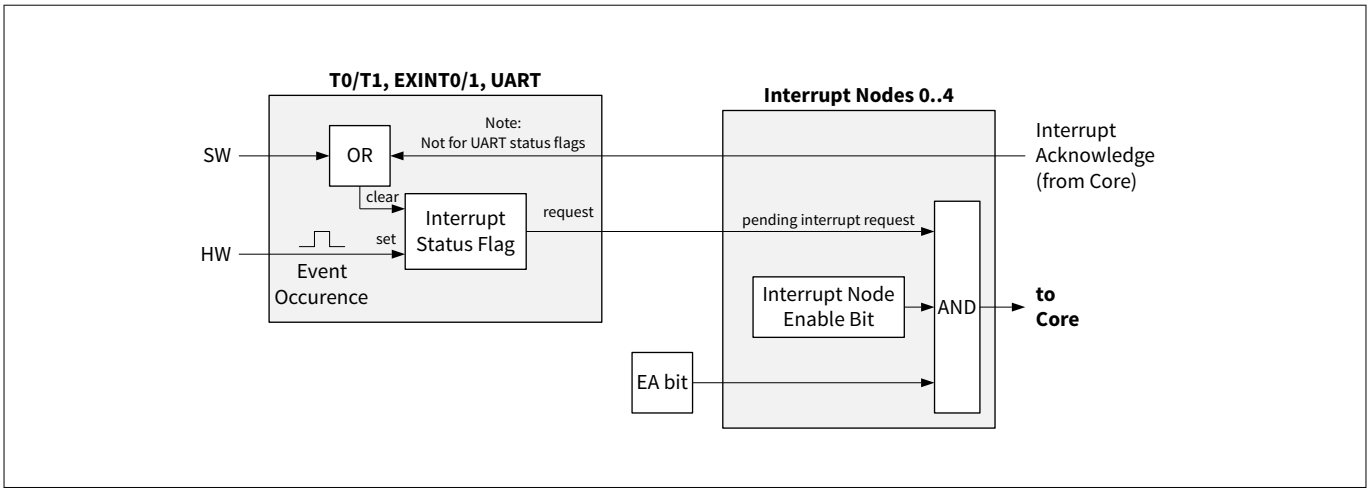


Figure 855 Interrupt structure 1

For the SCR, interrupt sources of Timer 0, Timer 1, external interrupt 0 and external interrupt 1 (each having a dedicated interrupt node) will have their respective interrupt status flags TF0, TF1, IE0 and IE1 in register T01_TCON cleared by the core, once their corresponding pending interrupt request is serviced. In the case that an interrupt node is disabled (for example software polling is used), its interrupt status flag must be cleared by software, since the core will not be interrupted (and therefore the interrupt acknowledge is not generated).

For the UART which has its dedicated interrupt node, interrupt status flags RI and TI in register SCON will NOT be cleared by the core, even when its pending interrupt request is serviced. The UART interrupt status flags (and hence the pending interrupt request) can only be cleared by software.

39.3.5.2.2 Interrupt structure 2

Interrupt structure 2 is implemented for interrupt nodes 5 through 13, relevant for the T2CCU0/1, LIN, external interrupts 2 to 15, WCAN, SSC, PMS RTC, I2CSCR and RTCSCR interrupt sources. For interrupt structure 2, the interrupt status flag does not directly drive the pending interrupt request. Instead, the interrupt nodes have a separate internal pending flag (FF).

An event generated by an associated interrupt source will set the internal pending flag. When the node is enabled, and the CPU is ready to service that interrupt request, the pending flag is automatically cleared by hardware (interrupt acknowledged) on entry into the service routine.

However, the status flag of the associated interrupt source is not automatically cleared; it must be cleared by software (preferably in the service routine).

Most of the interrupt nodes 5 through 13 have more than one associated interrupt source. The request signals of the sources are ORed for the set input of the pending flag. The clear input of the pending flag is either activated by the interrupt acknowledge, or when the status flags of ALL associated interrupt sources are cleared (inactive).

If one wants to clear the internal pending flag in an interrupt node with structure 2, special considerations need to be taken into account, depending on the associated interrupt source(s).

The interrupt sources differ slightly in the configuration of the request and the status lines. This is detailed in the following.

Related information

Registers overview on page 6531

Interrupt structure 2 for node 5

The following figure illustrates the configurations for interrupt node 5. Here, two different types of interrupt source configurations (indicated as Type b and Type c) are connected to the same node.

- For the LIN and the CCT0IRQ (of T2CCU0) interrupt sources, the request line as well as the status line are enabled/disabled through the associated interrupt enable bit. This means, when disabling the interrupt source, also the status line to the clear input of the interrupt node's pending flag is deactivated
- For the two interrupt sources T20IRQ and EXF20IRQ of the T2CCU0, only the request line is enabled/disabled through the associated interrupt enable bit. This means, when disabling the interrupt source, the status bit needs to be cleared by software in order to be able to clear the pending flag in the interrupt node

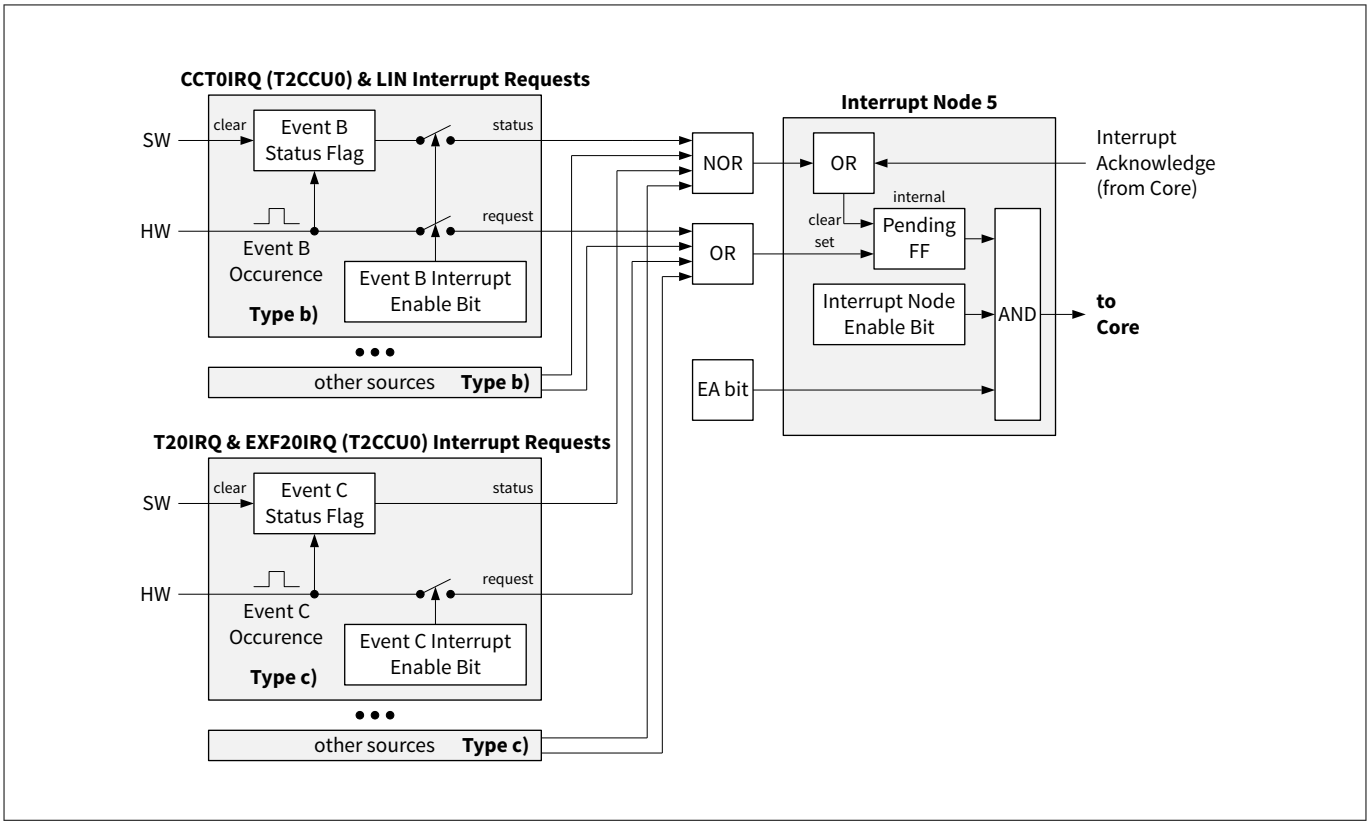


Figure 856 Interrupt structure 2 for node 5

Interrupt structure 2 for node 6

The following figure illustrates the configuration for interrupt node 6, which is connected to the WCAN interrupt sources. Only the request line is enabled/disabled through the associated interrupt enable bit. This means, when disabling the interrupt source, all active status bits of the WCAN module need to be cleared by software in order to be able to clear the pending flag in the interrupt node.

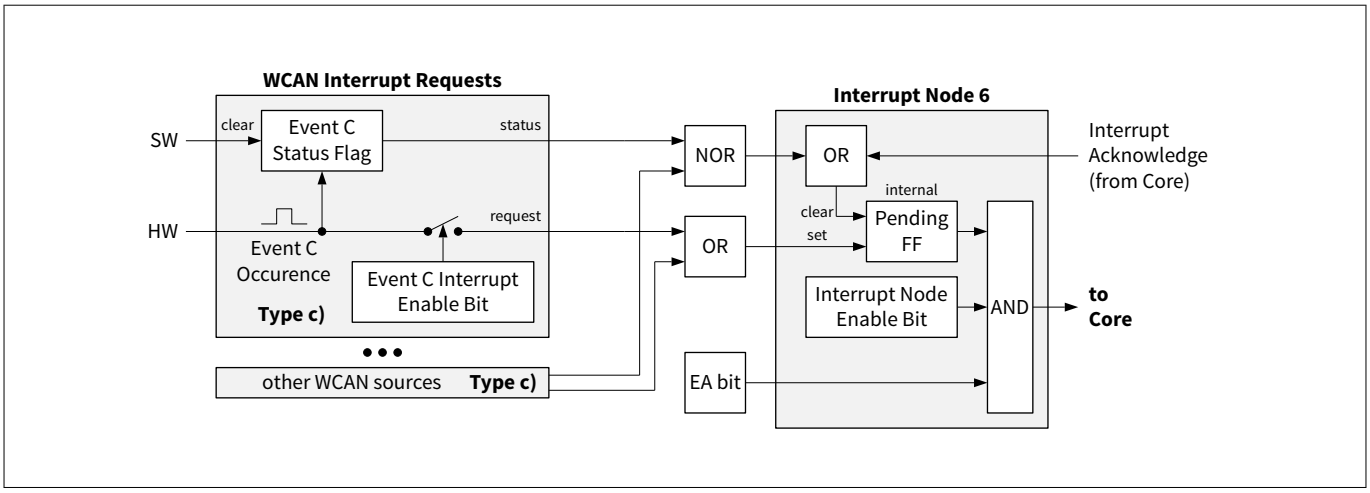


Figure 857 Interrupt structure 2 for node 6

Interrupt structure 2 for node 7

The following figure illustrates the configuration for interrupt node 7, which is connected to the SSC, ADCOMP, T2CCU1 and PMS RTC CMP0 interrupt sources. These interrupt sources are handled by a separate block in the interrupt module, controlled by registers IRSCR_MODIEN, IRSCR_CON1, and IEN1. Three different types of source configurations (indicated as type b, type c and type d) are connected to the same node.

- For the CCT1IRQ (of T2CCU1) interrupt source, the request line as well as the status line are enabled/ disabled through the associated interrupt enable bit. This means, when disabling the interrupt source, also the status line to the clear input of the interrupt node's pending flag is deactivated
- For the two interrupt sources T21IRQ and EXF21IRQ of the T2CCU1, only the request line is enabled/ disabled through the associated interrupt enable bit. This means, when disabling the interrupt source, status bit needs to be cleared by software in order to be able to clear the pending flag in the interrupt node
- Interrupt sources for ADCOMP, SSC and PMS RTC CMP0 are also connected to the interrupt node 7. For these interrupt sources, status flag is only set when the interrupt event is enabled. In order to clear the pending flag in the interrupt node, all the associated status flags for these interrupt sources must be cleared by software

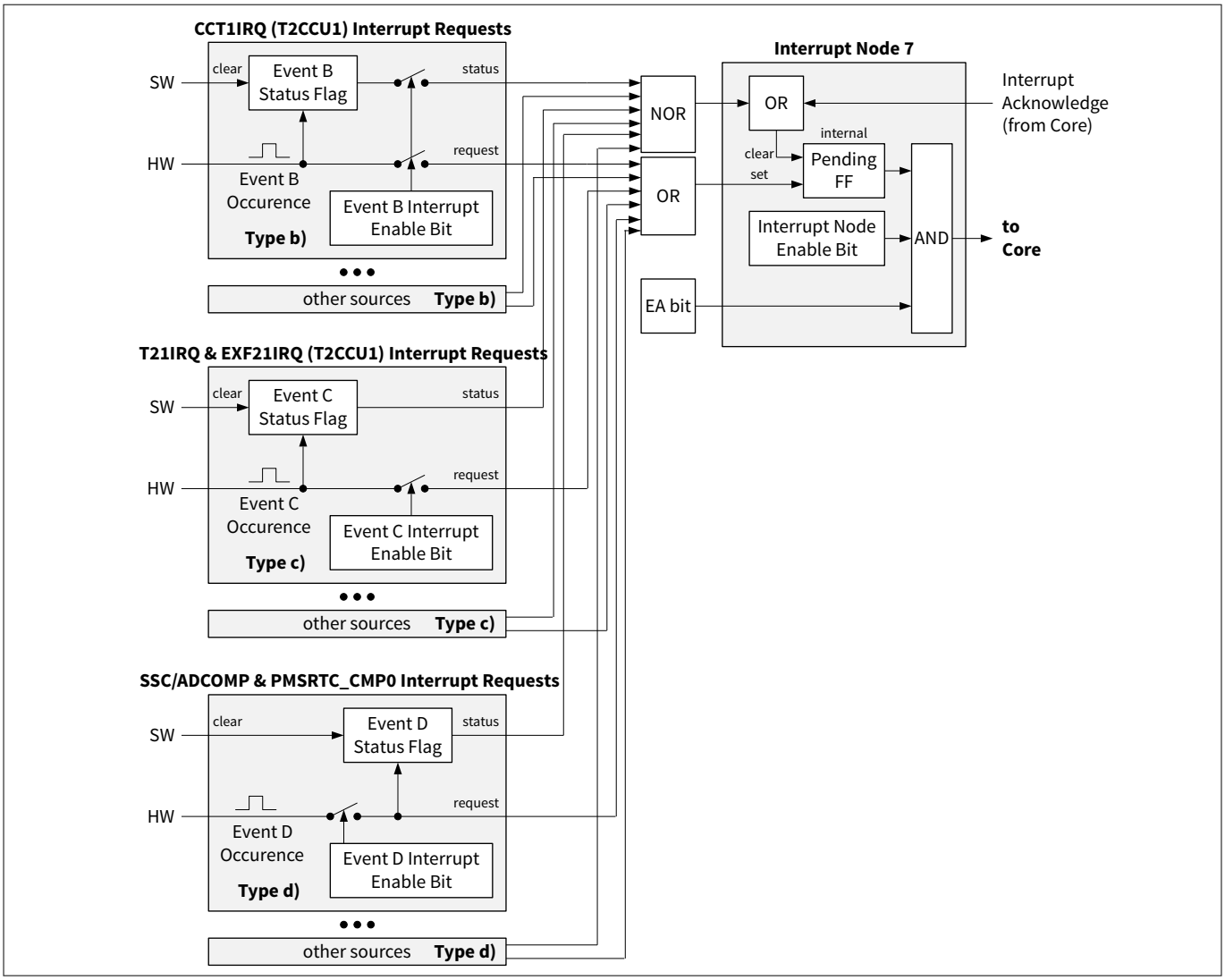


Figure 858 Interrupt structure 2 for node 7

Interrupt structure 2 for node 8

The following figure shows the configurations for interrupt node 8. Three different interrupt source configurations (indicated as Type a, Type b and Type d) are connected to this node.

- The external Interrupt EXINT2 does not have a separate interrupt enable control; instead, an event is disabled by selecting the '0'-input of its edge-detection multiplexer
- For the I2CSCR interrupt source, the request line as well as the status line are enabled/disabled through the associated interrupt enable bit. This means, when disabling the interrupt source, also the status line to the clear input of the interrupt node's pending flag is deactivated
- For interrupt sources PMS_RTC_CMP1, the request line is enabled/disabled through associated interrupt enable bit. Here, the status flag is only set when the interrupt event is enabled

In order to clear the pending flag in the interrupt node, all associated status flags connected to that node must be cleared by software.

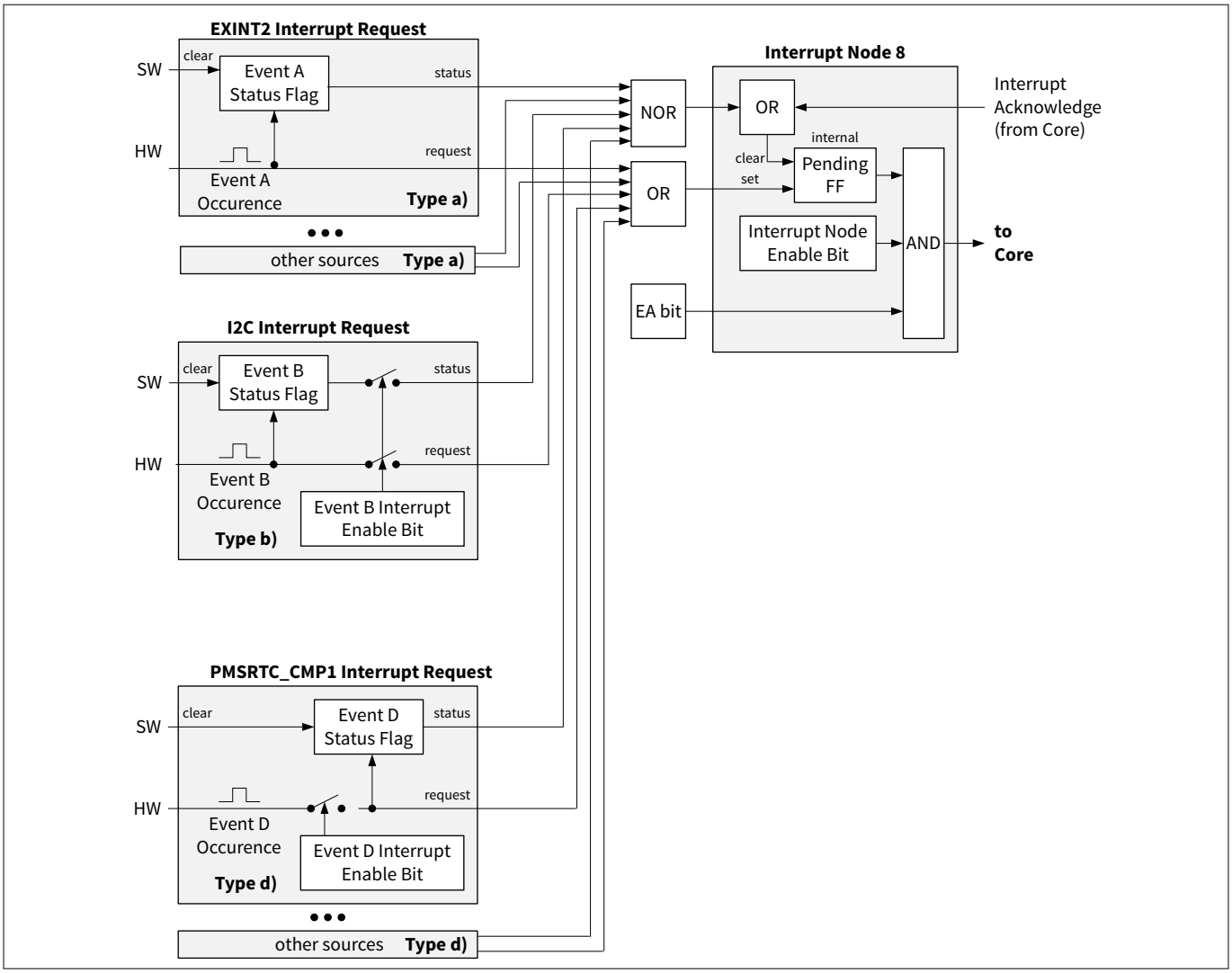


Figure 859 Interrupt structure 2 for node 8

***Note**: I2C here, is the I2CSCR*

Interrupt structure 2 for nodes 9 through 12

The following figure illustrates the configuration (indicated as Type a) for interrupt nodes 9 through 12, which are connected to the external interrupts EXINT3 through EXINT15. All interrupt sources connected to these nodes have the same configuration (Type a). These external interrupts do not have a separate interrupt enable control; instead, an event is disabled by selecting the '0'-input of its edge-detection multiplexer.

In order to clear the pending flag in the interrupt node, all associated status flags connected to that node must be cleared by software.

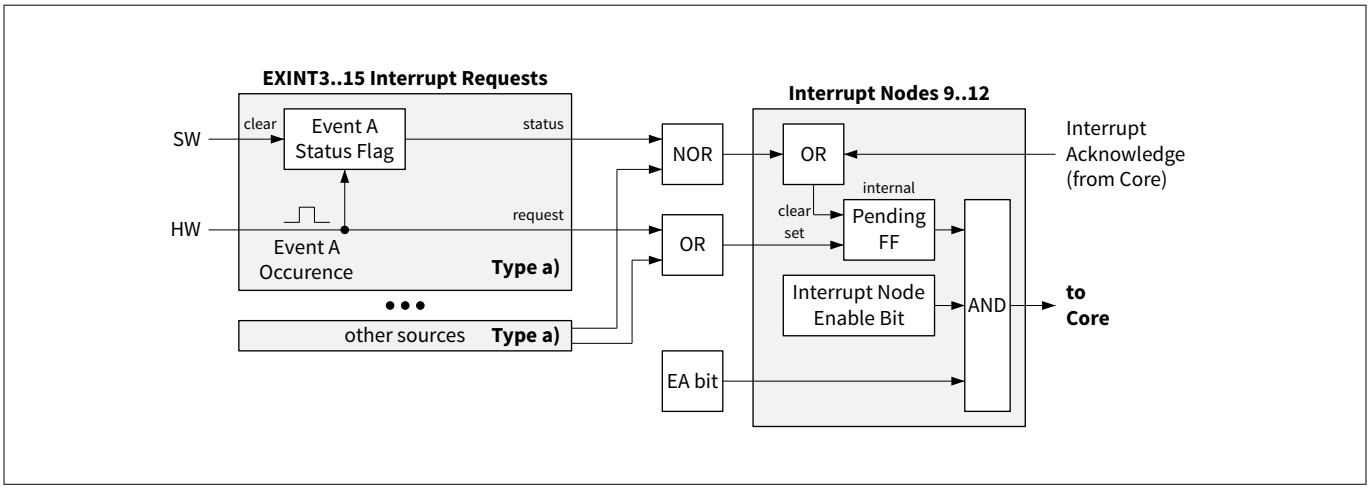


Figure 860 Interrupt structure 2 for nodes 9 through 12

Interrupt structure 2 for node 13

The following figure illustrates the configuration for interrupt node 13, which is solely connected to the RTCSCR interrupt source. Only the request line is enabled/disabled through the associated interrupt enable bit. This means, when disabling the interrupt source, the status bit needs to be cleared by software in order to be able to clear the pending flag in the interrupt node.

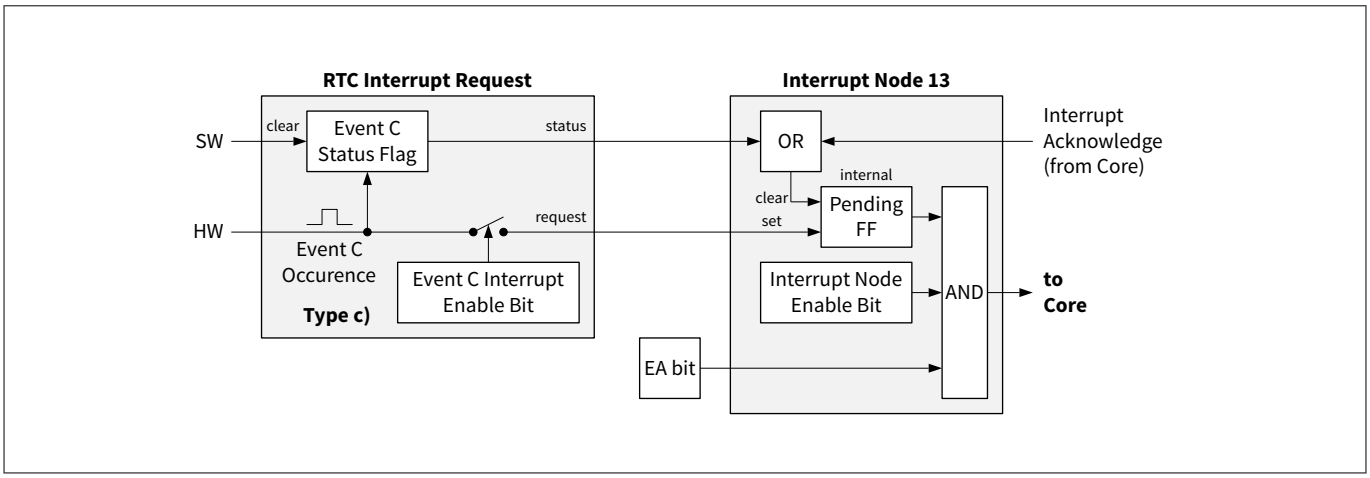


Figure 861 Interrupt structure 2 for node 13

39.3.5.3 Interrupt handling

The interrupt request signals are sampled at phase 2 in each CPU cycle. The sampled requests are then polled during the following CPU cycle. If one interrupt node request was active at phase 2 of the preceding cycle, the polling cycle will find it and the interrupt system will generate a LCALL to the node's service routine, provided this hardware-generated LCALL is not blocked by any of the following conditions:

1. An interrupt of equal or higher priority is already in progress
2. The current (polling) cycle is not in the final cycle of the instruction in progress
3. The instruction in progress is RETI or any write access to registers IEN0/IEN1, IP/IPH or IP1/IPH1

Any of these three conditions will block the generation of the LCALL to the interrupt service routine. Condition 2 ensures that the instruction in progress is completed before vectoring to any service routine. Condition 3 ensures that if the instruction in progress is RETI or any write access to registers IEN0/IEN1 or IP/IPH or IP1/

IPH1, then at least one more instruction will be executed before any interrupt is vectored to; this delay guarantees that changes of the interrupt status can be observed by the CPU.

The polling cycle is repeated with each CPU cycle, and the values polled are the values that were present at phase 2 of the previous CPU cycle. Note that if any interrupt flag is active but its node interrupt request was not responded to for one of the conditions already mentioned, and if the flag is no longer active at a later time when servicing the interrupt node, the corresponding interrupt source will not be serviced. In other words, the fact that the interrupt flag was once active but not serviced is not remembered. Every polling cycle interrogates only the pending interrupt requests.

The processor acknowledges an interrupt request by executing a hardware generated LCALL to the corresponding service routine. In some cases, hardware also clears the flag that generated the interrupt, while in other cases, the flag must be cleared by the user's software. The hardware-generated LCALL pushes the contents of the Program Counter (PC) onto the stack (but it does not save the PSW) and reloads the PC with an address that depends on the interrupt node being vectored to, as shown in the table 'Interrupt Vector Address' in related information.

Program execution returns to the next instruction after calling the interrupt when the RETI instruction is encountered. The RETI instruction informs the processor that the interrupt routine is no longer in progress, then pops the two top bytes from the stack and reloads the PC. Execution of the interrupted program continues from the point where it was stopped. Note that the RETI instruction is important because it informs the processor that the program has left the current interrupt priority level. A simple RET instruction resumes execution to the interrupted program, but it leaves the interrupt control system under the assumption that an interrupt is still in progress. In this case, no interrupt of the same or lower priority level will be acknowledged.

Related information

Interrupt source and vector of the SCR on page 6519

39.3.5.4 Interrupt response time

Due to an interrupt event of (the various sources of) an interrupt node, its corresponding request signal will be sampled active at phase 2 in every CPU cycle. The value is not polled by the circuitry until the next CPU cycle. If the request is active and conditions are right for it to be acknowledged, a hardware subroutine call to the requested service routine will be the next instruction to be executed. The call itself takes two CPU cycles. Therefore, a minimum of three complete CPU cycles will elapse from activation of the interrupt request to the beginning of execution of the first instruction of the service routine as shown in the following figure.

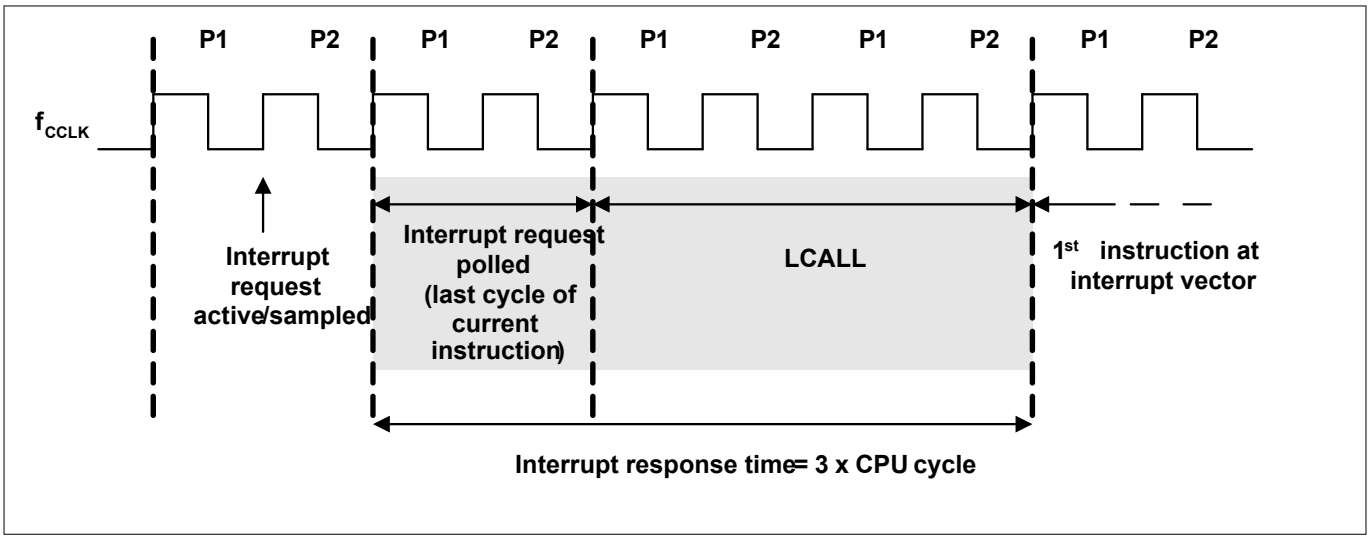


Figure 862 Minimum interrupt response time

A longer response time is obtained if the request is blocked by one of the three previously listed conditions:

1. If an interrupt of equal or higher priority is already in progress, the additional wait time will depend on the nature of the other interrupt's service routine
2. If the instruction in progress is not in its final cycle, the additional wait time cannot be more than three CPU cycles since the longest instructions (MUL and DIV) are only four CPU cycles long. See following figure
3. If the instruction in progress is RETI or a write access to registers IEN0, IEN1 or IP(H), IP1(H), the additional wait time cannot be more than five cycles (a maximum of one more CPU cycle to complete the instruction in progress, plus four CPU cycles to complete the next instruction, if the instruction is MUL or DIV). See figure 'Interrupt Response Time for Condition 3' below

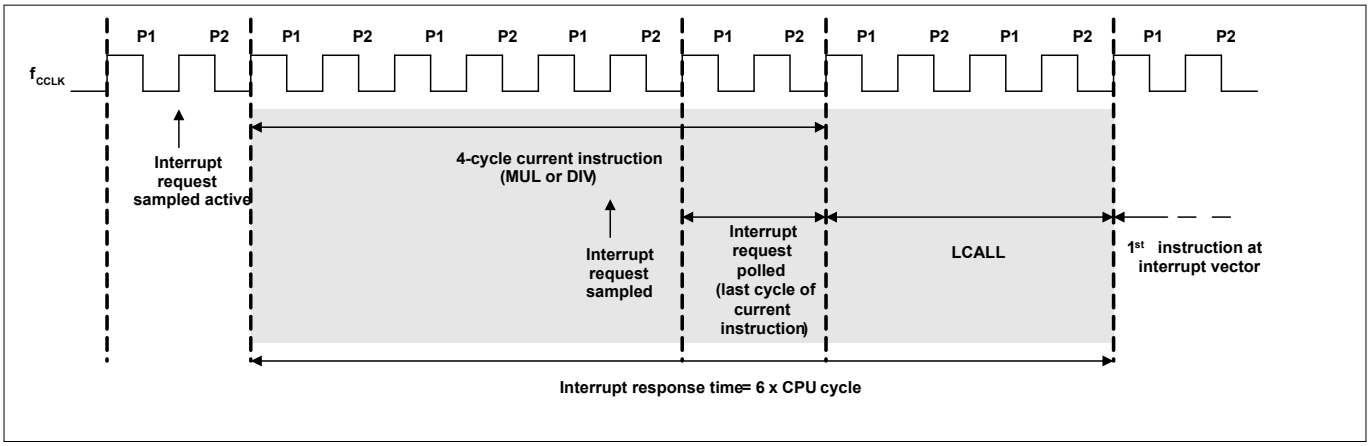


Figure 863 Interrupt response time for condition 2

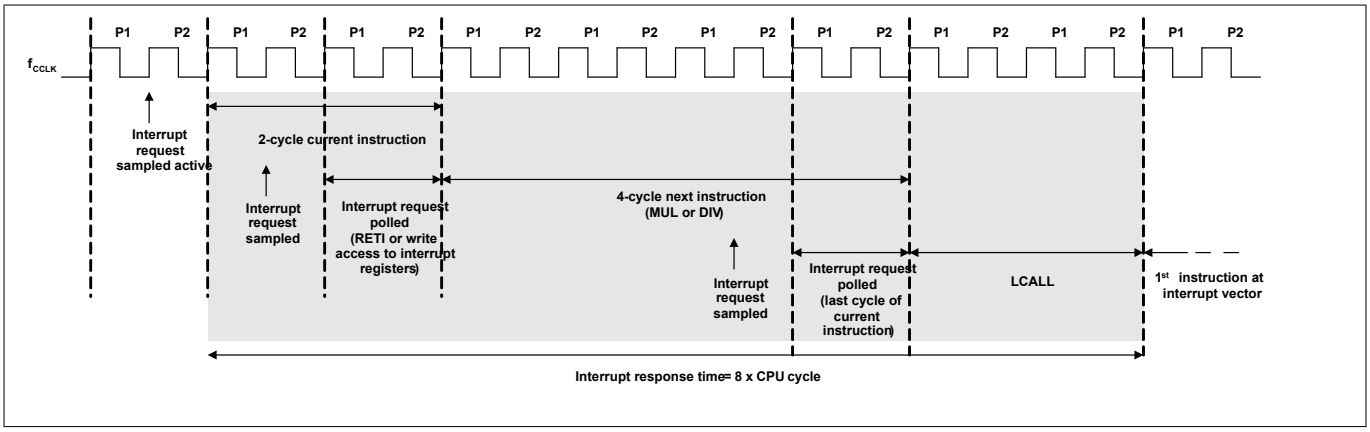


Figure 864 Interrupt response time for condition 3

Thus in a single interrupt system, the response time is always more than three CPU cycles and less than nine CPU cycles (wait states are not considered). When considering wait states, the interrupt response time will be extended depending on the user instructions (also the hardware generated LCALL) being executed during the interrupt response time (shaded region in the figures above).

39.3.5.5 Interrupts from or to TriCore™ CPUx

There is an interrupt signal from TriCore™ CPUx to trigger an SCR NMI event. It is triggered through PMS_SCR_CON1.TCINTREQ bit-field that is located in the PMS register space. The interrupt data exchange register, TCINTEXCHG, is used to indicate the additional message of events that causes this interrupt.

In SCR, there is also an interface to trigger interrupts to CPUx. This is done through bits IRSCR_NMICON.SCRINTTC0 and IRSCR_NMICON.SCRINTTC1 . The additional message of SCR events that causes this interrupt to CPUx can be stored in register SCRINTEXCHG. Interrupt triggered over IRSCR_NMICON.SCRINTTC0 is combined with other SCR-to-PMS interrupt sources (watchdog reset, ECC-error,

PIN-reset, Soft-reset) and routed to TriCore™ domain interrupt router. IRSCR_NMICON.SCRINTTC1 is assigned to a different PMS interrupt line to the main interrupt system. Having two separate interrupt lines to the main interrupt system allows configuring two different reactions – for example, one can be for error handling while other could trigger DMA transfers.

SCR software can send information to the TriCore™ domain through SCRINTEXCHG register. When written it will update the PMS_SCR_STAT.SCRINT bit-field. When read, the registerSCRINTEXCHG includes the data of the register PMS_SCR_CON1.TCINT as it is a virtual register. When IRSCR_NMICON.SCRINTTC0/1 are set, it will be cleared by the hardware with the next clock cycle.

To be able to issue an SCR NMI interrupt, the SCR has to be enabled, as well as respective NMI interrupt. It is set on the edge of the request. When PMS_SCR_CON0.SCREN=0 (SCR is not enabled), write access to PMS_SCR_CON1 register is blocked. (A trap will be generated.) It is not possible to request an interrupt of SCR from TC while SCR is not enabled. In addition to TCINTREQ, SCR NMI can be triggered from the TriCore™ domain, if it gets an SMU reset or an Application reset.

Interrupts triggered by the SCR to the TriCore™ domain have the following limitations:

- Interrupts triggered by the SCR in power down are lost for the TriCore™ domain
- Interrupts during an ongoing reset of the TriCore™ domain are delayed and applied as soon as the TriCore™ domain can service it

39.3.5.6 Registers overview

"Interrupt Special Function" Registers or bits are used for interrupt configuration such as node enable, external interrupt control, interrupt flags and interrupt priority setting.

The table below lists the SFRs and corresponding addresses.

Table 1691 Register map

Address	Register
RMAP = 0 or 1	
88_H	SYSCON0
$D8_H$	IEN0
$D1_H$	IEN1
DC_H	IP
$D2_H$	IPH
DB_H	IP1
$D3_H$	IPH1
$C9_H$	T01_TCON
BA_H	UART_SCON
RMAP = 0, IR Page = 0	
$F7_H$	IR_CON0
$F8_H$	IR_CON1
$F9_H$	IR_CON2
RMAP = 0, IR Page = 1	
$F7_H$	IR_NMICON
$F8_H$	IR_NMISTAT
$F9_H$	IR_MODIEN
RMAP = 0, IR Page = 2	
$F7_H$	IR_EXICON0
$F8_H$	IR_EXICON1
$F9_H$	IR_EXICON2
RMAP = 0, IR Page = 3	
$F7_H$	IR_EXICON3
RMAP = 0, IR Page = 4	
$F7_H$	IR_PISEL0
$F8_H$	IR_PISEL1
$F9_H$	IR_PISEL2
RMAP = 0, IR Page = 5	
$F7_H$	IR_PISEL3

Interrupt node enable registers

Each interrupt node can be individually enabled or disabled by setting or clearing the corresponding bit in the interrupt enable registers IEN0 and IEN1. Register IEN0 also contains the global interrupt masking bit (EA), which can be cleared to block all pending interrupt requests at once.

The NMI interrupt vector is shared by a number of sources, each of which can be enabled or disabled individually through register IR_NMICON.

After reset, the enable bits in IEN0, IEN1 and IR_NMICON are cleared to 0. This implies that all interrupt nodes are disabled by default.

External interrupt control registers

The 16 external interrupts, EXINT[15:0], are driven into the SCR from the ports. The external interrupt pin assignment for SCR is shown in the table below. Each external interrupt can be positive, negative, or double-edge triggered. Registers IR_EXICON0 through IR_EXICON3 specify the active edge for an external interrupt. Among the external interrupts, external interrupt 0 and external interrupt 1 can be selected to bypass edge detection for direct feed-through to the core. The signal to the core can be further programmed to either low-level or negative transition activated, by bits T01_TCON.IT0 and T01_TCON.IT1. However, for edge detection in IR, T01_TCON.IT0/1 must be set to falling edge triggered. A detected active edge event will generate internally two CCLK cycle low pulse for detection by the core.

If the external interrupt is positive (negative) edge triggered, the external source must hold the request pin low (high)for at least one CCLK cycle, and then hold it high (low) for at least one CCLK cycle to ensure that the transition is recognized (detection is done in the interrupt module outside of the core). If edge detection is bypassed for external interrupt 0 and/or external interrupt 1, the external source must hold the request pin “high” or “low” for at least two CCLK cycles (detection is done inside of the core) In the core there are two configuration possibilities:

T01_TCON.IT[0,1] = 0: low level triggered external interrupt is selected. In this case the request pin must be "low" until the interrupt is detected.

T01_TCON.IT[0,1] = 1: falling edge triggered external interrupt is selected. In this case the external source must hold the requested pin "high" or "low" for at least two CCLK cycles.

External interrupts 2 to 15 share 5 interrupt nodes. Therefore, in addition to the corresponding interrupt node enable, each external interrupt 2 to 15 may be disabled individually; they are disabled by default after reset.

Table 1692 External interrupt [15:0] pin functions and selection (attention SCR ports)

Pin	Function	Description	Selected by
SCR_P0.4 (P33.4)	EXINT0A	External Interrupt Input 0	IR_PISEL0.EXINT0IS = 00_B
SCR_P0.7 (P33.7)	EXINT0B		IR_PISEL0.EXINT0IS = 01_B
SCR_P1.5 (P33.13)	EXINT0C		IR_PISEL0.EXINT0IS = 10_B
SCR_P3.2 (P34.3)	EXINT0D		IR_PISEL0.EXINT0IS = 11_B
SCR_P0.5 (P33.5)	EXINT1A	External Interrupt Input 1	IR_PISEL0.EXINT1IS = 00_B
SCR_P1.0 (P33.8)	EXINT1B		IR_PISEL0.EXINT1IS = 01_B
SCR_P1.6 (P33.14)	EXINT1C		IR_PISEL0.EXINT1IS = 10_B
SCR_P3.3 (P34.4)	EXINT1D		IR_PISEL0.EXINT1IS = 11_B
SCR_P0.6 (P33.6)	EXINT2A	External Interrupt Input 2	IR_PISEL0.EXINT2IS = 00_B
SCR_P1.2 (P33.10)	EXINT2B		IR_PISEL0.EXINT2IS = 01_B
SCR_P1.7 (P33.15)	EXINT2C		IR_PISEL0.EXINT2IS = 10_B
SCR_P3.4 (P34.5)	EXINT2D		IR_PISEL0.EXINT2IS = 11_B

(table continues...)

Table 1692 (continued) External interrupt [15:0] pin functions and selection (attention SCR ports)

Pin	Function	Description	Selected by
SCR_P0.0 (P33.0)	EXINT3A	External Interrupt Input 3	IR_PISEL1.EXINT3_T2CCU0CH0_IS = 00_B
SCR_P1.1 (P33.9)	EXINT3B		IR_PISEL1.EXINT3_T2CCU0CH0_IS = 01_B
SCR_P1.7 (P33.15)	EXINT3C		IR_PISEL1.EXINT3_T2CCU0CH0_IS = 10_B
SCR_P2.6 (P35.1)	EXINT3D		IR_PISEL1.EXINT3_T2CCU0CH0_IS = 11_B
SCR_P0.1 (P33.1)	EXINT4A	External Interrupt Input 4	IR_PISEL1.EXINT4_T2CCU0CH1_IS = 00_B
SCR_P1.3 (P33.11)	EXINT4B		IR_PISEL1.EXINT4_T2CCU0CH1_IS = 01_B
SCR_P3.0 (P34.1)	EXINT4C		IR_PISEL1.EXINT4_T2CCU0CH1_IS = 10_B
SCR_P3.6 (P35.4)	EXINT4D		IR_PISEL1.EXINT4_T2CCU0CH1_IS = 11_B
SCR_P0.2 (P33.2)	EXINT5A	External Interrupt Input 5	IR_PISEL1.EXINT5_T2CCU0CH2_IS = 00_B
SCR_P1.4 (P33.12)	EXINT5B		IR_PISEL1.EXINT5_T2CCU0CH2_IS = 01_B
SCR_P3.1 (P34.2)	EXINT5C		IR_PISEL1.EXINT5_T2CCU0CH2_IS = 10_B
SCR_P3.7 (P35.5)	EXINT5D		IR_PISEL1.EXINT5_T2CCU0CH2_IS = 11_B
SCR_P0.3 (P33.3)	EXINT6A	External Interrupt Input 6	IR_PISEL1.EXINT6_T2CCU0CH3_IS = 00_B
SCR_P1.5 (P33.13)	EXINT6B		IR_PISEL1.EXINT6_T2CCU0CH3_IS = 01_B
SCR_P3.2 (P34.3)	EXINT6C		IR_PISEL1.EXINT6_T2CCU0CH3_IS = 10_B
SCR_P0.4 (P33.4)	EXINT6D		IR_PISEL1.EXINT6_T2CCU0CH3_IS = 11_B
SCR_P2.0 (P32.2)	EXINT7	External Interrupt Input 7	-
SCR_P2.1 (P32.4)	EXINT8A	External Interrupt Input 8	IR_PISEL2.EXINT8_T2CCU1CH0_IS = 00_B
SCR_P0.6 (P33.6)	EXINT8B		IR_PISEL2.EXINT8_T2CCU1CH0_IS = 01_B
SCR_P1.7 (P33.15)	EXINT8C		IR_PISEL2.EXINT8_T2CCU1CH0_IS = 10_B
SCR_P2.5 (P35.0)	EXINT8D		IR_PISEL2.EXINT8_T2CCU1CH0_IS = 11_B
SCR_P2.2 (P32.5)	EXINT9A	External Interrupt Input 9	IR_PISEL2.EXINT9_T2CCU1CH1_IS = 00_B
SCR_P0.7 (P33.7)	EXINT9B		IR_PISEL2.EXINT9_T2CCU1CH1_IS = 01_B
SCR_P3.0 (P34.1)	EXINT9C		IR_PISEL2.EXINT9_T2CCU1CH1_IS = 10_B
SCR_P2.7 (P35.2)	EXINT9D		IR_PISEL2.EXINT9_T2CCU1CH1_IS = 11_B
SCR_P2.3 (P32.6)	EXINT10A	External Interrupt Input 10	IR_PISEL2.EXINT10_T2CCU1CH2_IS = 00_B
SCR_P1.3 (P33.11)	EXINT10B		IR_PISEL2.EXINT10_T2CCU1CH2_IS = 01_B
SCR_P3.3 (P34.4)	EXINT10C		IR_PISEL2.EXINT10_T2CCU1CH2_IS = 10_B
SCR_P3.5 (P35.3)	EXINT10D		IR_PISEL2.EXINT10_T2CCU1CH2_IS = 11_B
SCR_P2.4 (P32.7)	EXINT11A	External Interrupt Input 11	IR_PISEL2.EXINT11_T2CCU1CH3_IS = 00_B
SCR_P1.4 (P33.12)	EXINT11B		IR_PISEL2.EXINT11_T2CCU1CH3_IS = 01_B
SCR_P3.4 (P34.5)	EXINT11C		IR_PISEL2.EXINT11_T2CCU1CH3_IS = 10_B
SCR_P1.0 (P33.8)	EXINT11D		IR_PISEL2.EXINT11_T2CCU1CH3_IS = 11_B
SCR_P2.1 (P32.4)	EXINT12A	External Interrupt Input 12	IR_PISEL3.EXINT12IS = 00_B
SCR_P0.0 (P33.0)	EXINT12B		IR_PISEL3.EXINT12IS = 01_B
SCR_P0.4 (P33.4)	EXINT12C		IR_PISEL3.EXINT12IS = 10_B

(table continues...)

Table 1692 (continued) External interrupt [15:0] pin functions and selection (attention SCR ports)

Pin	Function	Description	Selected by
SCR_P2.6 (P35.1)	EXINT12D		IR_PISEL3.EXINT12IS = 11_B
SCR_P2.2(P32.5)	EXINT13A	External Interrupt Input 13	IR_PISEL3.EXINT13IS = 00_B
SCR_P0.1 (P33.1)	EXINT13B		IR_PISEL3.EXINT13IS = 01_B
SCR_P0.5 (P33.5)	EXINT13C		IR_PISEL3.EXINT13IS = 10_B
SCR_P2.7 (P35.2)	EXINT13D		IR_PISEL3.EXINT13IS = 11_B
SCR_P2.3 (P32.6)	EXINT14A	External Interrupt Input 14	IR_PISEL3.EXINT14IS = 00_B
SCR_P0.2 (P33.2)	EXINT14B		IR_PISEL3.EXINT14IS = 01_B
SCR_P1.1 (P33.9)	EXINT14C		IR_PISEL3.EXINT14IS = 10_B
SCR_P3.5 (P35.3)	EXINT14D		IR_PISEL3.EXINT14IS = 11_B
SCR_P2.4 (P32.7)	EXINT15A	External Interrupt Input 15	IR_PISEL3.EXINT15IS = 00_B
SCR_P0.3 (P33.3)	EXINT15B		IR_PISEL3.EXINT15IS = 01_B
SCR_P1.2 (P33.10)	EXINT15C		IR_PISEL3.EXINT15IS = 10_B
SCR_P3.6 (P35.4)	EXINT15D		IR_PISEL3.EXINT15IS = 11_B

***Note**: Several external interrupts support alternative input signals. When switching inputs, the active edge/ level trigger select and the level on the associated pins should be considered to prevent unintentional interrupt generation.*

The following figures show block diagrams and connection overviews of External Interrupts 0 through 15. EXINT7 has only single input so there are no respective PISEL bits and multiplexer.

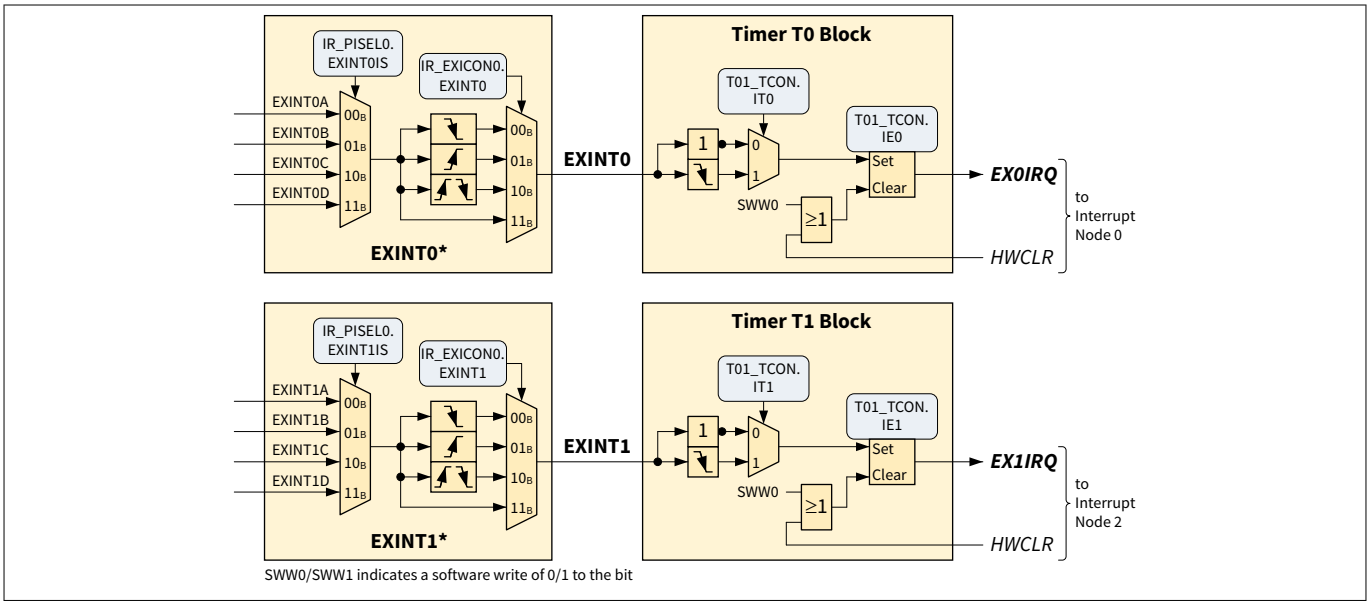
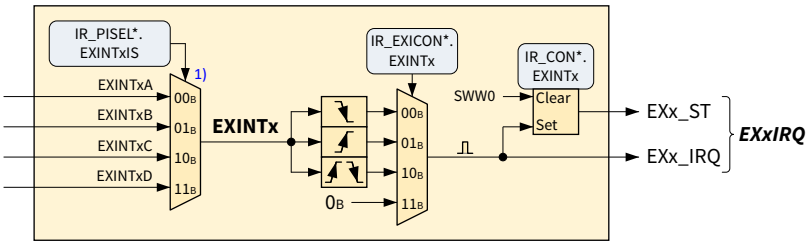


Figure 865 Block diagram and connections of external interrupts EXINT0 and EXINT1



SWW0 indicates a software write of 0 to the bit

Note 1) EXINT7 has only 1 input and no PISEL multiplexer

Figure 866 Generic block diagram of external interrupts EXINT2 through EXINT15. EXINT7 has 1 input and all other have 4 inputs (A-D)

Interrupt flag registers

The interrupt flags for the different interrupt sources are located in several "Special Function Registers". In case of software and hardware access to a flag bit at the same time, hardware will have higher priority.

***Note**: NMISTAT register can only be cleared by software or reset to 0 after a generated SCR_RESET.*

Interrupt priority registers

Each interrupt node can be individually programmed to one of the four priority levels available. Two pairs of Interrupt Priority Registers are available to program the priority level of the each interrupt vector. The first pair of Interrupt Priority Registers are SFRs IP and IPH. The second pair of Interrupt Priority Registers are SFRs IP1 and IPH1.

The corresponding bits in each pair of Interrupt Priority Registers select one of the four priority levels as shown in table below.

Table 1693 Interrupt priority level selection

IPH.x/IPH1.x	IP.x/IP1.x	Priority level
0	0	Level 0 (lowest)
0	1	Level 1
1	0	Level 2
1	1	Level 3 (highest)

***Note**: NMI always has the highest priority (above Level 3); it does not use the priority level given in table above.*

Interrupt event enable control

The five interrupt events of SSC module are of interrupt structure 2. As there is no enable/disable bit(s) for these interrupt events within the module, bits are defined in the IR register IR_MODIEN to provide for this purpose. Also modules like ADCOMP, PMS RTC CMPn are of interrupt structure 2 and their bits are available in the above mentioned register.

Interrupt data exchange register

Two registers are used to exchange interrupt data between main controller and SCR as described in related information.

Related information

Interrupts from or to TriCore CPUx on page 6529

39.3.5.7 Interrupt flag overview

The interrupt events have interrupt flags that are located in different SFRs. Next table shows the corresponding SFR to which each interrupt flag belongs. Detailed information on the interrupt flags is provided in the respective peripheral chapters.

Table 1694 Location of the interrupt flags

Interrupt event	Interrupt flag	SFR
Timer 0 Overflow	TF0	T01_TCON
Timer 1 Overflow	TF1	T01_TCON
Timer2 Overflow	TF2	T2CCU0/1_T2_CON
Timer2 External Event	EXF2	T2CCU0/1_T2_CON
T2CCU CCT Overflow	CCTOVF	T2CCU0/1_CCTCON
UART Receive	RI	UART_SCON
UART Transmit	TI	UART_SCON
LIN End of Synch Byte	EOFSYN	LIN_STAT
LIN Synch Byte Error	ERRSYN	LIN_STAT
External Interrupt 0	IE0	T01_TCON
External Interrupt 1	IE1	T01_TCON
External Interrupt 2	EXINT2	IRSCR_CON0
External Interrupt 3	EXINT3	IRSCR_CON0
External Interrupt 4	EXINT4	IRSCR_CON0
External Interrupt 5	EXINT5	IRSCR_CON0
External Interrupt 6	EXINT6	IRSCR_CON0
External Interrupt 7	EXINT7	IRSCR_CON0
External Interrupt 8	EXINT8	IRSCR_CON2
External Interrupt 9	EXINT9	IRSCR_CON2
External Interrupt 10	EXINT10	IRSCR_CON2
External Interrupt 11	EXINT11	IRSCR_CON2
External Interrupt 12	EXINT12	IRSCR_CON2
External Interrupt 13	EXINT13	IRSCR_CON2
External Interrupt 14	EXINT14	IRSCR_CON2
External Interrupt 15	EXINT15	IRSCR_CON2
WCAN WUF detection interrupt	WUF	WCAN_INTESTAT1
WCAN error overflow interrupt	SYSERR	WCAN_INTESTAT0
WCAN WUP detection interrupt	WUP	WCAN_INTESTAT0

(table continues...)

Table 1694 (continued) Location of the interrupt flags

Interrupt event	Interrupt flag	SFR
WCAN time-out interrupt	CANTO	WCAN_INTESTAT0
RTC compare/wakeup interrupt	CFRTC	RTCSCR_CON
ADCOMP Service Request	ADCIR	IRSCR_CON1
SSC Error	EIR	IRSCR_CON1
SSC Transmit	TIR	IRSCR_CON1
SSC Receive	RIR	IRSCR_CON1
SSC Receive FIFO Full	RFEIR	IRSCR_CON1
SSC Receive FIFO Empty	RFFIR	IRSCR_CON1
PMS RTC CMP0 OUT	CMP0IR	IRSCR_CON1
PMS RTC CMP1 OUT	CMP1IR	IRSCR_CON1
I2C Interrupt	IFLG	I2CSCR_CNTR
Watchdog Timer NMI	FNMIWDT	IRSCR_NMISTAT
RAM Double bit ECC Error NMI	FNMIRAMECC	IRSCR_NMISTAT
Wake-Up Event NMI	FNMIWKP	IRSCR_NMISTAT
External NMI (through EXTNMI pin SCR_P1.6/P33.14) NMI	FNMIEXTNMI	IRSCR_NMISTAT
OCDS NMI	FNMIOCDS	IRSCR_NMISTAT
Main Controller NMI	FNMIMAINC	IRSCR_NMISTAT

39.3.6 Watchdog timer (WDTSCR)

39.3.6.1 Overview

The Watchdog timer (WDTSCR) provides a highly reliable and secure way to detect and recover from software or hardware failures. The WDTSCR is reset at a regular interval that is predefined by the user. The CPU must service the WDTSCR within this interval to prevent the WDTSCR from causing generated reset. The routine service of the watchdog confirms that the system is functioning properly. This ensures that an accidental malfunction of the SCR will be aborted in a user-specified time period. The WDTSCR reset is default enabled. It can also be disabled through RSTCON register.

The WDTSCR is by default disabled through SCUSCR_PMCON0.WDT_DIS.

In debug mode, the watchdog is default suspended and stops counting. Its debug suspend bit is default set for example, SCR_SCUSCR_DBG_MODSUSP.WDTSUSP=1. Therefore during debugging, there is no need to refresh the watchdog.

Features

- 16-bit Watchdog timer
- Programmable reload value for upper 8 bits of timer
- Programmable window boundary
- Clock source from either the 70 kHz clock or the 100 MHz/DIV clock
- Selectable input frequency of $f_{WDTCLK}/2$ or $f_{WDTCLK}/128$

39.3.6.2 System information

This section consist of the system information required to use the WDTSCR.

39.3.6.2.1 Reset effects

The Watchdog Timer maintains a counter which must be refreshed or cleared periodically. Otherwise, the counter will overflow and the watchdog reset will be asserted. The assertion of the reset could be disabled by setting bit RSTCON.WDTRSTEN. The occurrence of a WDTSCR reset is indicated by the bit SCUSCR_RSTSTAT.WDTRST.

39.3.6.2.2 Clocking configuration

The WDT runs on either the 70 kHz clock or the 100 MHz/DIV clock. Bit WDTSCR_CON.WDTCLK is used to select the watchdog clock source, f_{WDTCLK}.

39.3.6.2.3 Interrupt events and assignment

The following table shows the non-maskable interrupt node assignment of the WDTSCR interrupt source.

Table 1695 WDTSCR events' non-maskable interrupt node control

Event	Interrupt node enable bit	Interrupt node flag bit	Vector address
WDTSCR Overflow	IRSCR_NMICON.NMIWDT	IRSCR_NMISTAT.FNMIWDT	73_H

39.3.6.2.4 Module suspend control

The timer in the WDTSCR is by default suspended on entering debug mode. The WDTSCR can be allowed to run in debug mode by clearing the bit MODSUSP.WDTSUSP to 0. Refer to the definition of register MODSUSP.

39.3.6.3 Functional description

The Watchdog Timer (WDTSCR) is a 16-bit timer, which is incremented by a count rate of $f_{WDTSCRCLK}/2$ or $f_{WDTSCRCLK}/128$. The f_{WDTCLK} clock input is selectable between the 70 kHz clock or the 100 MHz/DIV (PCLK, f_{SYS}) clock using the WDTCLK bit. This 16-bit timer is realized as two concatenated 8-bit timers. The upper 8 bits of the WDTSCR can be preset to a user-programmable value through a watchdog service access in order to vary the watchdog expire time. The lower 8 bits are reset on each service access. Next figure shows the block diagram of the WDTSCR unit.

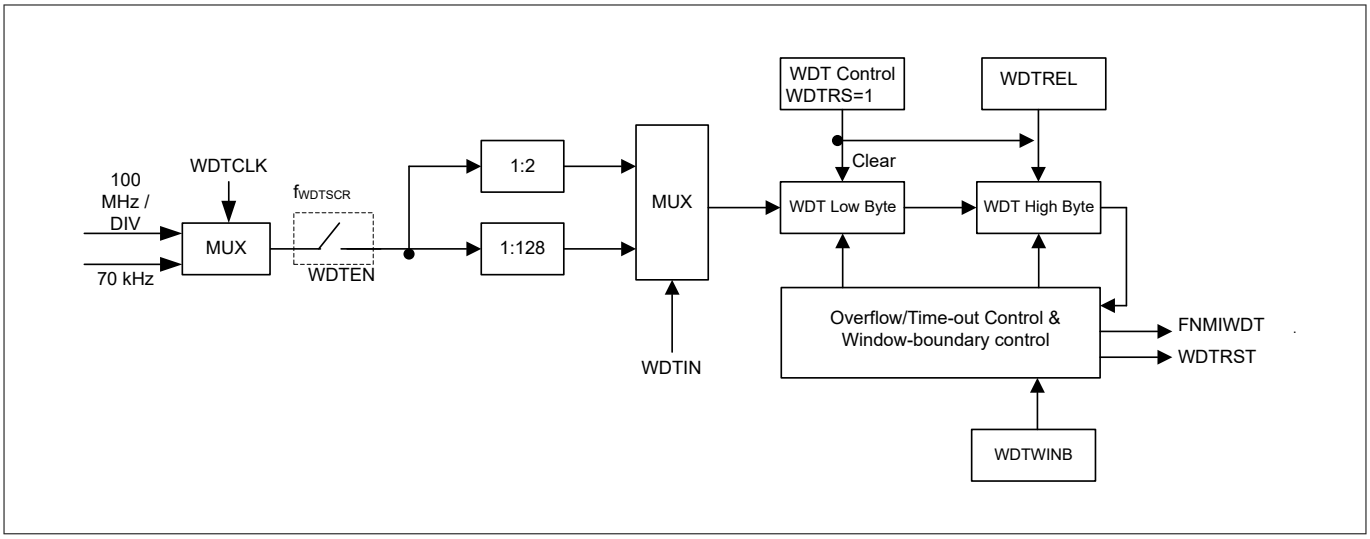


Figure 867 WDTSCR block diagram

***Note**: WDT is WDTSCR*

If the WDTSCR is enabled by setting WDTEN to 1, the timer is set to a user-defined start value (WDTSCR_REL) and begins counting up. It must be serviced before the counter overflows. Servicing is performed through refresh operation (setting bit WDTRS to 1, shown as WDT Control inside the graphic above). This reloads the timer with the start value, and normal operation continues.

If the WDTSCR is not serviced before the timer overflows, a system malfunction is assumed and normal mode is terminated. A WDTSCR NMI request (FNMIWDT) is then asserted and pre-warning is entered. The pre-warning lasts for 20_H count. During the pre-warning period, refreshing of the WDTSCR is possible in order to avoid WDTSCR generated reset. If WDTSCR is not refreshed before expiry of a pre-warning phase, a reset (WDTRST) of the SCR is imminent (if it is enabled by bit SCUSCR_RSTCON.WDTRSTEN). The occurrence of a WDTSCR reset is indicated by the bit SCUSCR_RSTSTAT.WDTRS. If refresh happens at the same time an overflow occurs, WDTSCR will not go into pre-warning period.

***Note**: In the SCR, it is possible to wake-up the main controller from standby mode when the WDTSCR overflow event happened by setting bit SCUSCR_WKPCON.WDTWKSEL to 1.*

The WDTSCR must be serviced periodically so that its count value will not overflow. Servicing the WDTSCR clears the low byte and reloads the high byte with the preset value in bit-field WDTREL. Servicing the WDTSCR also clears the bit WDTRS.

The WDTSCR has a “programmable window boundary”, which disallows any refresh during the WDTSCR’s count-up. A refresh during this window-boundary constitutes an invalid access to the WDTSCR and causes the WDTSCR to activate WDTRST, although no NMI request is generated in this instance. The window boundary is from 0000_H to the value obtained from the concatenation of WDTSCR_WINB and 00_H. This feature can be enabled by WINBEN.

After being serviced, the WDTSCR continues counting up from the value (<WDTREL> * 2^8). The time period for an overflow of the WDTSCR is programmable in two ways:

- The input frequency to the WDTSCR can be selected through bit WDTIN in register WDTSCR_CON to be either $f_{WDTCLK}/2$ or $f_{WDTCLK}/128$
- The reload value WDTREL for the high byte of WDTSCR can be programmed in register WDTSCR_REL

The period P_{WDTSCR} between servicing the watchdog and the next overflow can be determined by the following formula:

$$P_{WDT} = \frac{2^{(1+ <WDTIN> *6)} * \left(2^{16} - WDTREL * 2^{8}\right)}{f_{PCLK}}$$

If the Window-Boundary Refresh feature of the WDTSCR is enabled, the period during which a refresh is allowed is shortened if WDTSCR_WINB is greater than WDTSCR_REL. See also following figure. This period can be calculated by the same formula by replacing WDTSCR_REL WDTSCR_WINB. In order for this feature to be useful, WDTSCR_WINB cannot be smaller than WDTSCR_REL.

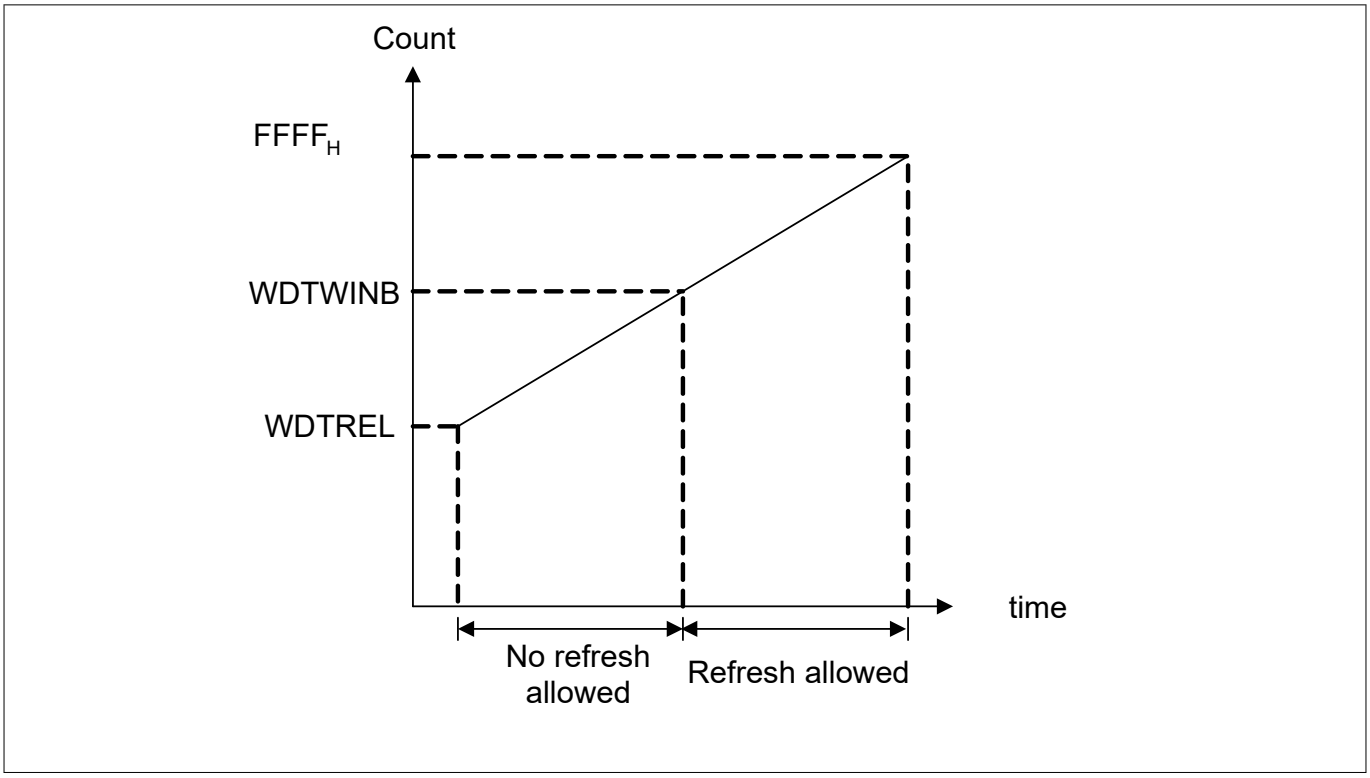


Figure 868 Watchdog timing diagram

Following table lists the possible ranges for the watchdog time which can be achieved using a certain module clock. Some numbers are rounded down to 2 significant digits.

Table 1696 Watchdog time ranges

Reload value in WDTREL	pre-scaler for f_{WDT}					
	2 (WDTIN=0)			128 (WDTIN=1)		
	20 MHz	100 MHz	70 kHz	20 MHz	100 MHz	70 kHz
FF_H	25.6 µs	5.12 µs	7.31 ms	1.63 ms	327.68 µs	486.11 ms
$7F_H$	3.3 ms	660.48 µs	943.54 ms	211.35 ms	42.27 ms	60.38 s
00_H	6.55 ms	1.31 ms	1.87 s	419.43 ms	83.88 ms	119.83 s

***Note**: The user is advised to rewrite WDTSCR_CON each time before the WDT is serviced.*

39 Stand-by Controller (SCR)

The Watchdog Timer Current Count Value is contained in the Watchdog Timer Register WDTSCR_H and WDTSCR_L, which are non-bit-addressable read-only register. The operation of the WDT is controlled by WDT Control Register WDTSCR_CON. This register also selects the input clock pre-scaling factor. The register WDTSCR_REL specifies the reload value for the high byte of the timer. WDTSCR_WINB specifies Watchdog Window-Boundary count value.

39.3.7 General purpose I/O Ports (PORT8)

The SCR has up to 32 general purpose I/O ports with 8-bit peripheral I/O lines that are shared with the 32-bit peripheral I/O lines. It is organized into four parallel ports. Internally within the SCR, these ports are always referred to as SCR_P0-SCR_P3. Same pads are mapped to pins in ports P32, P33, P34 and P35 of the TriCore™ domain. This particular pin assignment of these functions is shown in next table:

Table 1697 Pin assignment for ports in SCR

SCR Ports/pins	TriCore™ domain ports/pins
SCR_P0.0-SCR_P0.7	P33.0-P33.7
SCR_P1.0-SCR_P1.7	P33.8-P33.15
SCR_P2.0-SCR_P2.7	P32.2,P32.4-7,P35.0-2
SCR_P3.0-SCR_P3.7	P34.1-P34.5,P35.3-5

***Note**: Not all pins described in this chapter might be available for a specific device and package.*

During RUN mode, if SCR is enabled, the control of these shared Pads can be switched between SCR and the TriCore™ domain through 32-bit Ports register PCSRSEL in combination with SCREN value. If SCR is enabled, the shared Pads are controlled by the SCR after an Application reset (see default value of PCSRSEL). If SCR is disabled, independent from respective PCSRSEL settings, the 32-bit Ports module has the control over shared Pads.

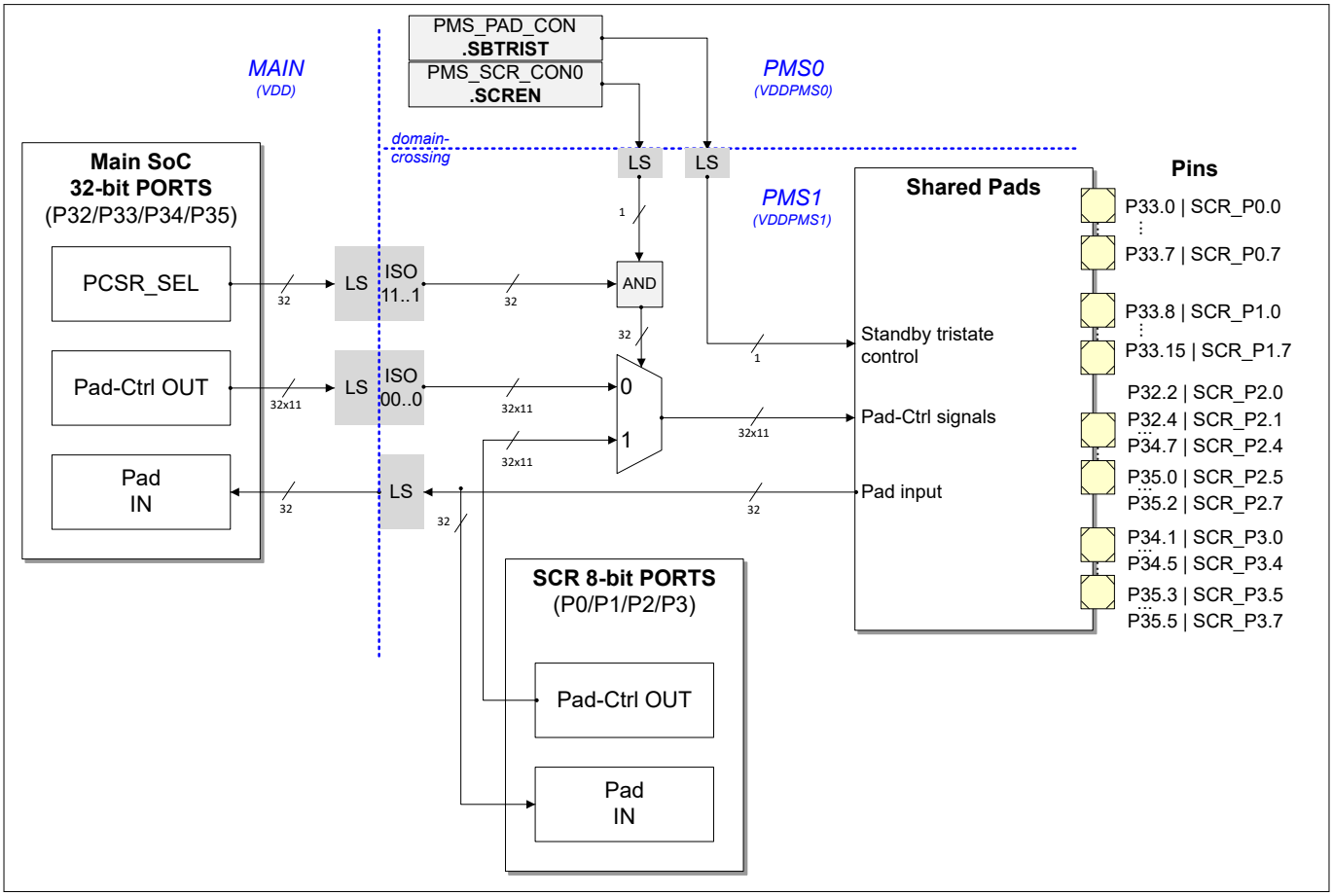


Figure 869 Block diagram for shared pads

After initial TC4xx start-up, the default value of "SCR enable signal" (PMS_SCR_CON0.SCREN) is '0'. Despite the value of PCSRSEL='1' the shared pads are in control of the TriCore™ domain 32-bit Ports module because SCR is not enabled. Dynamics of shared Pads control in different modes is shown in figure below.

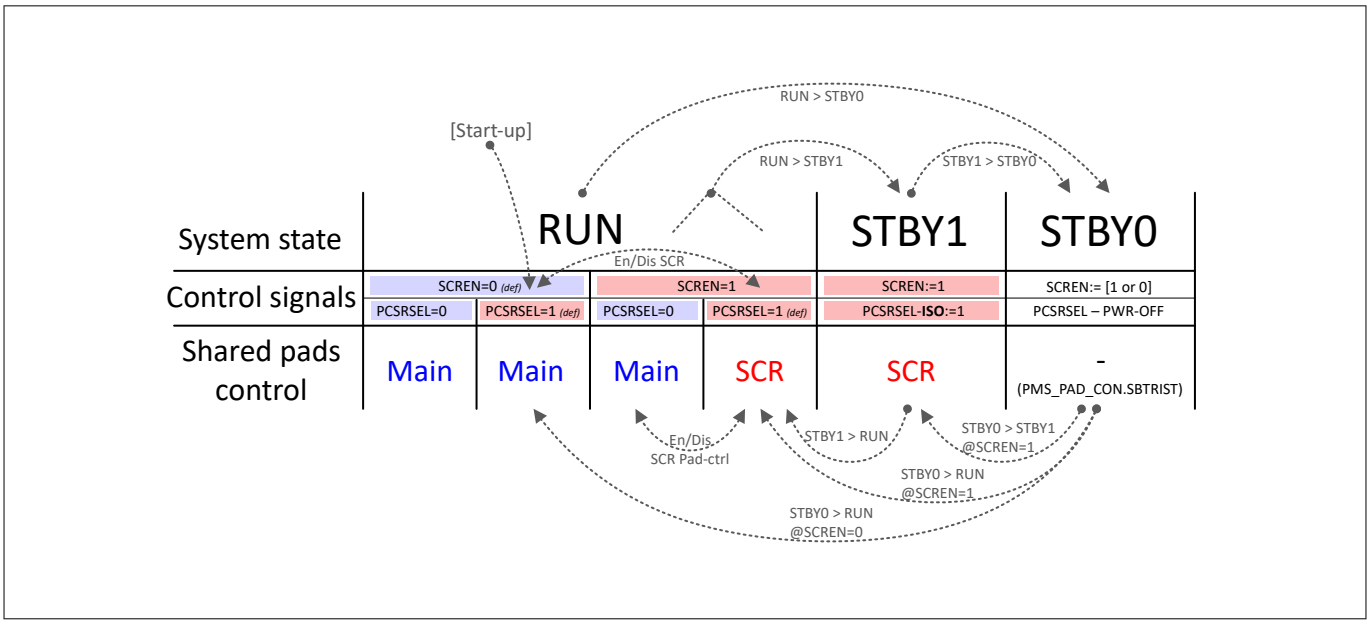


Figure 870 Dynamics of shared pads control

After start-up, and after enabling SCR all shared pads are under SCR control. Like this the system could transition from RUN into STANDBY1 mode and shared pads will remain under SCR control. After wake-up from STANDBY1 to RUN mode, the shared pads will always be controlled by SCR until software on TriCore™ domain takes control by either disabling SCR or by clearing PCSRSEL to 0.

In cases where system transitions from RUN to STANDBY0 mode, since neither 32-bit ports nor 8-bit ports are powered during STANDBY0, shared pads will take value as specified in PMS_PAD_CON.SBTRIST. On return from STANDBY0 to RUN, shared pads will be under SCR control if SCREN=1 or under TriCore™ domain control if SCREN=0. Similarly, transitions between STANDBY1 and STANDBY0 mode transfer the pad control between SCR and tri-state. For these transitions, special care was taken to avoid glitches.

Following timing diagram illustrates how can ownership of shared pads be exchanged between TriCore™ domain, SCR and tri-state. It shows which events lead to change of shared pads control.

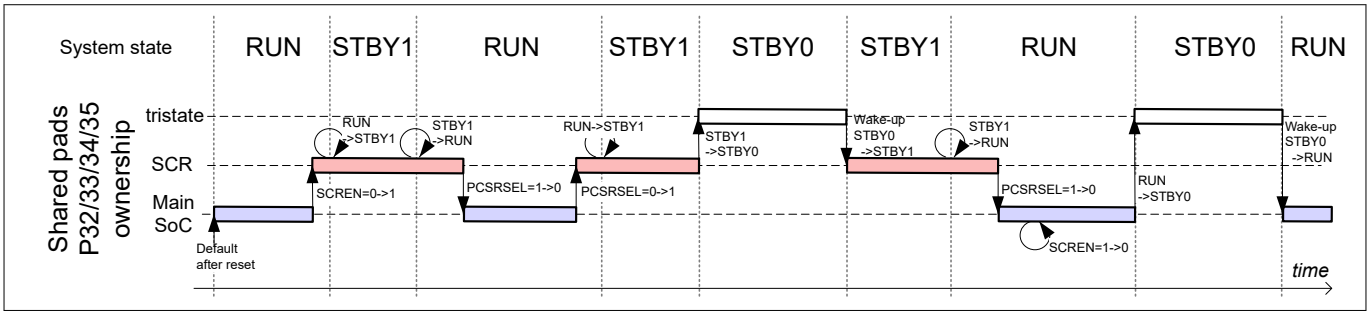


Figure 871 Shared pads control, timing diagram

***Note**: Each change of control between SCR and TriCore™ domain could lead to a sudden change of Pad configuration. Special care must be taken in TriCore™ domain code to avoid undesired transitions which could lead to glitches on shared Pads outputs.*

39.3.7.1 Basic port operation

Following figure illustrates a general block diagram of an SCR GPIO port slice.

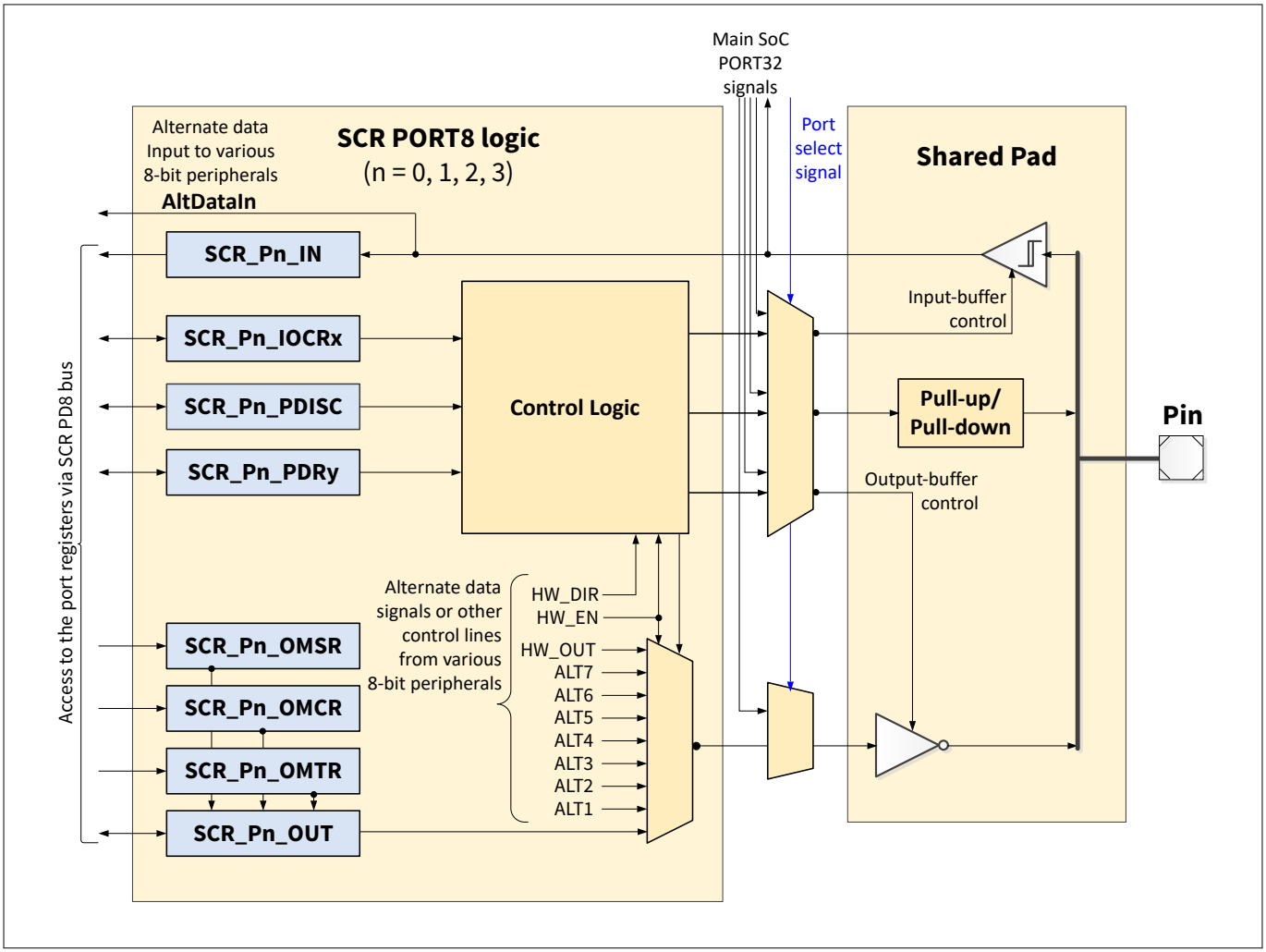


Figure 872 General structure of an SCR Port pin

Each port line has a number of control and data bits, enabling very flexible usage of the line. Each port pin can be configured for input or output operation. In input mode (default after reset), the output driver is switched off (high-impedance). The actual voltage level present at the port pin is translated into a logical 0 or 1 via a Schmitt-Trigger device and can be read via the read-only register SCR_Pn_IN (n=0, 1, 2, 3). Input signals are connected directly to the various inputs of the peripheral units (AltDataIn). The function of the input line from the pin to the input register SCR_Pn_IN and to AltDataIn is independent of whether the port pin operates as input or output.

This means that when the port is in output mode, the level of the pin can be read by software via SCR_Pn_IN or a peripheral can use the pin level as an input.

In output mode, the output driver is activated and drives the value supplied through the multiplexer to the port pin. Switching between input and output mode is selected through the SCR_Pn_IOCR register, which enables or disables the output driver. If a peripheral unit uses a GPIO port line as a bi-directional I/O line, register SCR_Pn_IOCR has to be written for input or output selection. The SCR_Pn_IOCR register further controls the driver type of the output driver, and determines whether an internal weak pull-up, pull-down, or without input pull device is alternatively connected to the pin when used as an input. This offers additional advantages in an application.

The output multiplexer in front of the output driver selects the signal source for the GPIO line when used as output. If the pin is used as general-purpose output, the multiplexer is switched by software (SCR_Pn_IOCR

register) to the Output Data Register SCR_Pn_OUT. Software can set, clear, or toggle the bit in SCR_Pn_OUT through separate SCR_Pn_OMSR, SCR_Pn_OMCR, or SCR_Pn_OMTR registers. The manipulation of the control bits in these registers can directly influence the state of the port pin. If the on-chip peripheral units uses the pin for output signals, the alternate output lines ALT1 to ALT7 can be switched via the multiplexer to the output driver. The data written into the output register SCR_Pn_OUT by software can be used as input data to an on-chip peripheral. This enables, for example, peripheral tests via software without external circuitry.

When selected as general-purpose output line, the logic state of each port pin can be changed individually by programming the pin-related bits in the Output Modification Set Register SCR_Pn_OMSR, Output Modification Clear Register SCR_Pn_OMCR, or the Output Modification Toggle Register SCR_Pn_OMTR.

When selected as general-purpose output line, the actual logic level at the pin can be examined through reading SCR_Pn_IN and compared against the applied output level (either applied through software via the output register SCR_Pn_OUT, or via an alternate output function of a peripheral unit). This can be used to detect some electrical failures at the pin caused through external circuitry. In addition, software-supported arbitration schemes can be implemented in this way using the open-drain configuration and an external wired-And circuitry. Collisions on the external communication lines can be detected when a high level (1) is output, but a low level (0) is seen when reading the pin value via the input register SCR_Pn_IN.

39.3.7.2 Port register overview

The individual control and data bits of each parallel port are implemented in a number of 8-bit registers. Bits with the same meaning and function are assembled together in the same register. The registers configure and use the port as general purpose I/O or alternate function input/output. For some ports, not all registers are implemented. The availability of the registers in the specific ports is described separately. The definition of these registers are similar to the 32-bit format.

The registers are shown below.

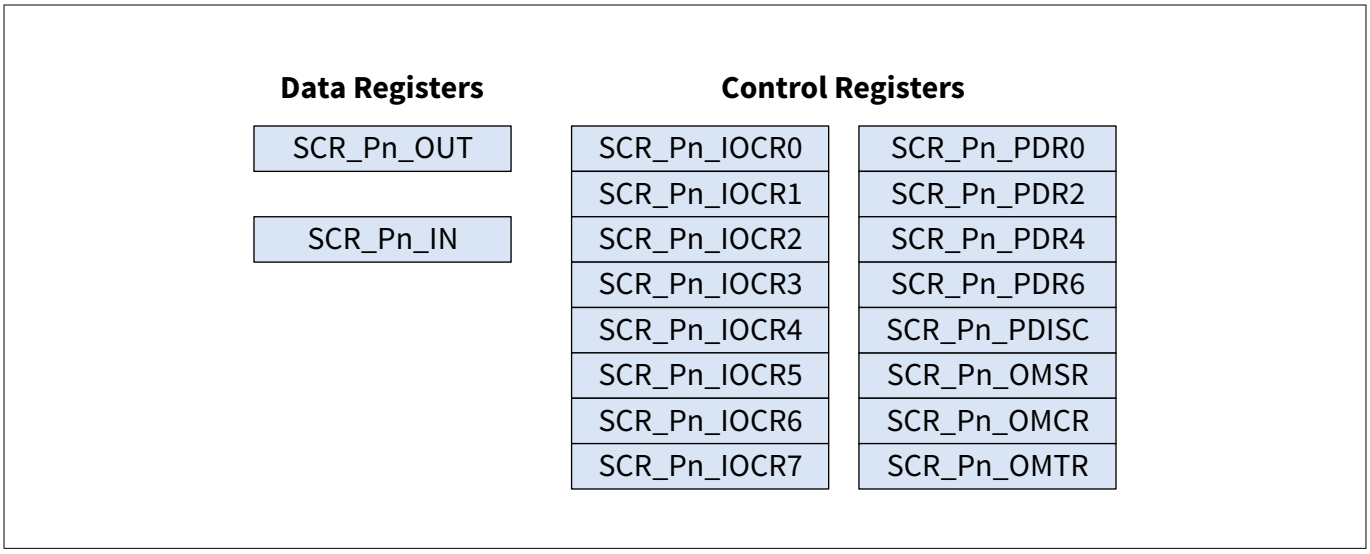


Figure 873 SCR port registers

The Port SFRs are located in the standard memory area (RMAP=0).

39.3.7.2.1 Port input/output control registers

The port input/output control registers select the digital output and input driver functionality and characteristics of a GPIO port pin. Port direction (input or output), pull-up, pull-down, or no pull devices for inputs, and push-pull or open-drain functionality for outputs can be selected by the corresponding bit field PC.

Each 8-bit wide port input/output control register controls one GPIO port line, for example, register SCR_P0_IOCR0 controls the SCR_P0.0 (P33.0) port line, register SCR_P1_IOCR0 controls the SCR_P1.0 port line (P33.8), register SCR_P3_IOCR4 controls the SCR_P3.4 (P34.5) port line.

The pictures below show the register layouts of the port input/output control registers with the PCm bit fields. One PCm bit field controls exactly one port line Pn.m.

***Note**: The post-reset values of 10_H and 00_H for SCR_Pn_IOCRm registers represents input pull-up and no input pull device (tri-state mode) being activated, respectively. The switching of the intended mode of the device is controlled by HWCFG6 and PMS_PAD_CON.SBTRIST. This values are applied after each SCR reset.*

Port control coding

Following table describes the coding of the PCm bit fields that determine the port line functionality.

Table 1698 PCm coding

PCm[4:0]	I/O	Output characteristics	Selected pull-up/pull-down/ selected output function
$0XX00_B$	Input		No input pull device connected, tri-state mode
$0XX01_B$			Input pull-down device connected
$0XX10_B$			Input pull-up device connected
$0XX11_B$			No input pull device connected, tri-state mode
10000_B	Output	Push-pull	General-purpose output
10001_B			Alternate output function 1
10010_B			Alternate output function 2
10011_B			Alternate output function 3
10100_B			Alternate output function 4
10101_B			Alternate output function 5
10110_B			Alternate output function 6
10111_B			Alternate output function 7
11000_B		Open-drain	General-purpose output
11001_B			Alternate output function 1
11010_B			Alternate output function 2
11011_B			Alternate output function 3
11100_B			Alternate output function 4
11101_B			Alternate output function 5
11110_B			Alternate output function 6
11111_B			Alternate output function 7

39.3.7.2.2 Pad driver mode register

Overview

The pad structure of the SCR GPIO lines offers the possibility to select the output driver strength and the slew rate. These two parameters are controlled by the PDm bit fields in the pad driver mode registers Pn_PDR0y (y=0, 2, 4, 6), independently from input/output and pull-up/pull-down control functionality as programmed in the Pn_IOCR register. SCR_Pn_PDRy registers are assigned to each port.

The assignment of each port pin to one of these pad classes is shown in the port configuration figures. Further details about pad driver classes that are available in the TC4Dx are summarized in the datasheet.

Depending on the assigned pad class, the 3-bit wide pad driver mode selection bit fields PDm in the pad driver mode registers SCR_Pn_PDRy make it possible to select the port line functionality as shown in the tables below.

Table 1699 Pad driver mode selection - Fast pads

PDm.1	PDm.0	Speed grade	Driver setting
0	0	1	Strong Driver, Sharp Edge (“ss”)

(table continues...)

Table 1699 (continued) Pad driver mode selection - Fast pads

PDm.1	PDm.0	Speed grade	Driver setting
0	1	2	Strong Driver, Medium Edge (“sm”)
1	X	3	Medium Driver (“m”)

Table 1700 Pad driver mode selection- slow pads

PDm.1	PDm.0	Speed grade	Driver setting
X	0	1	Medium Driver, Sharp Edge (“sm”)
X	1	2	Medium Driver (“m”)

***Note**: TC4Dx data-sheet describes the DC characteristics of all pad classes.*

TTL/Automotive input selection

The input function can operate with different VIH and VIL levels depending on the pad supply voltage and the selection done by the PLx bits of the Pn_PDRx registers, as shown in the following table.

PLx.1 additionally also changes the pull-up and pull-down resistors.

Table 1701 Pad level selection

PLx.1	PLx.0	Input levels
0	X	Automotive level “AL”
1	0	TTL level for 5V supply
1	1	TTL level for 3.3V supply

Pad driver mode registers

This is the general description of the PDR registers. Each port contains its own specific PDR registers, described additionally at each port, that can contain between one and eight PDm fields for SCR_Pn_PDRy y (y=0, 2, 4, 6) registers, respectively. Each PDm field controls one pin. For coding of PDm, see 'Pad Driver Mode Selection' tables. Similarly, each PLx bit controls one pin. For coding of PLx, see table above.

39.3.7.2.3 Pin function decision control register

The pad structure of the SCR GPIO lines offers the possibility to disable/enable port pad or select digital input. For the selection of digital input, the parameters defined for Class S pads must be met. This feature can be controlled by individual bits in the SCR_Pn_PDISC register, independently from input/output and pull-up/pull-down control functionality as programmed in the SCR_Pn_IOCR register. One SCR_Pn_PDISC register is assigned to each port.

***Note**: After reset, all SCR_Pn_PDISC registers have the reset value of FF_H.*

39.3.7.2.4 Port output register

The port output register determines the value of a GPIO pin when it is selected by SCR_Pn_IOCRm as output. Writing a 0 to a SCR_Pn_OUT.Pm (m=0-7) bit position delivers a low level at the corresponding output pin. A high level is output when the corresponding bit is written with a 1. Note that the bits of SCR_Pn_OUT.Pm can be individually set, cleared, or toggled by writing appropriate values into the port output modification set register SCR_Pn_OMSR, or port output modification clear register SCR_Pn_OMCR, or the port output modification toggle register SCR_Pn_OMTR, respectively.

39.3.7.2.5 Port output modification set register

The port output modification set register contains control bits that make it possible to individually set the logic state of a single port line by manipulating the output register.

39.3.7.2.6 Port output modification clear register

The port output modification clear register contains control bits that make it possible to individually clear the logic state of a single port line by manipulating the output register.

39.3.7.2.7 Port output modification toggle register

The port output modification toggle register contains control bits that make it possible to individually toggle the logic state of a single port line by manipulating the output register.

39.3.7.2.8 Port input register

The logic level of a GPIO pin can be read via the read-only port input register SCR_Pn_IN. Reading the SCR_Pn_IN register always returns the current logical value at the GPIO pin independently whether the pin is selected as input or output.

39.3.7.3 SCR functions in port SCR_P0

This section describes the SCR functions of Port SCR_P0, which is mapped on P33.0-P33.7.

The SCR has functions on Port lines SCR_P0.0 - SCR_P0.7, which can be alternatively used for UART/LIN, WCAN, SSC, T0/T1, T2CCU0/1, ADCOMP, Debug (DAP/SPD) and External Interrupt input and output functions.

Following table summarizes the SCR I/O control selection functions of the 8 Port SCR_P0 lines.

Table 1702 Port P0 functions

Port pin	I/O	Pin functionality	Associated reg./I/O line	Port I/O control selection	
				Reg./bit-field	Value
SCR_P0.0 (P33.0)	I	General-purpose input	SCR_P0_IN.P0	SCR_P0_IOCR0.PC	$0XXXX_B$
		T2CCU1 external input	T21EXB		
		External interrupt 3 / T2CCU0 input	EXINT3A/T2CCU0_0A		
		External interrupt 12	EXINT12B		
		ADCOMP input	ADCOMPCH0		
	O	General-purpose output	SCR_P0_OUT.P0		$1X000_B$
		UART output	UART_TXD		$1X001_B$
		Reserved	-		$1X010_B$
		Reserved	-		$1X011_B$
		Reserved	-		$1X100_B$
		T2CCU0 output	T2CCU0_0		$1X101_B$
		T2CCU1 output	T2CCU1_4		$1X110_B$
		Reserved	-		$1X111_B$

(table continues...)

Table 1702 (continued) Port P0 functions

Port pin	I/O	Pin functionality	Associated reg./I/O line	Port I/O control selection	
				Reg./bit-field	Value
SCR_P0.1 (P33.1)	I	General-purpose input	SCR_P0_IN.P1	SCR_P0_IOCR1.PC	$0XXXX_B$
		UART input	UART_RXDK		
		SSC input	SSC_SCLKD		
		T0 input	T0A		
		T2CCU0 external input	T20EXA		
		External interrupt 4 / T2CCU0 input	EXINT4A/T2CCU0_1A		
		External interrupt 13	EXINT13B		
		ADCOMP input	ADCOMPCH1		
	O	General-purpose output	SCR_P0_OUT.P1		$1X000_B$
		UART output	UART_RXD_O		$1X001_B$
		SSC output	SSC_SCLK		$1X010_B$
		Reserved	-		$1X011_B$
		T2CCU1 output	T21EXF2		$1X100_B$
		T2CCU0 output	T2CCU0_1		$1X101_B$
		T2CCU1 output	T2CCU1_5		$1X110_B$
		Reserved	-		$1X111_B$
SCR_P0.2 (P33.2)	I	General-purpose input	SCR_P0_IN.P2	SCR_P0_IOCR2.PC	$0XXXX_B$
		WCAN input	WCAN_RXDE		
		Debug input 0	DAP0_1		
		T2CCU1 input	T21B		
		External interrupt 5 / T2CCU0 input	EXINT5A/T2CCU0_2A		
		External interrupt 14	EXINT14B		
		ADCOMP input	ADCOMPCH2		
	O	General-purpose output	SCR_P0_OUT.P2		$1X000_B$
		UART output	UART_TXD		$1X001_B$
		Reserved	-		$1X010_B$
		T2CCU0 output	T20EXF2		$1X011_B$
		Reserved	-		$1X100_B$
		T2CCU0 output	T2CCU0_2		$1X101_B$
		Reserved	-		$1X110_B$
		Reserved	-		$1X111_B$

(table continues...)

Table 1702 (continued) Port P0 functions

Port pin	I/O	Pin functionality	Associated reg./I/O line	Port I/O control selection	
				Reg./bit-field	Value
SCR_P0.3 (P33.3)	I	General-purpose input	SCR_P0_IN.P3	SCR_P0_IOCR3.PC	$0XXXX_B$
		UART input	UART_RXDP		
		Debug input	DAP1_1/SPD_1		
		T2CCU0 input	T20A		
		External interrupt 6 / T2CCU0 input	EXINT6A/T2CCU0_3A		
		External interrupt 15	EXINT15B		
		ADCOMP input	ADCOMPCH3		
	O	General-purpose output	SCR_P0_OUT.P3		$1X000_B$
		UART output	UART_RXD_O		$1X001_B$
		Reserved	-		$1X010_B$
		Reserved	-		$1X011_B$
		Reserved	-		$1X100_B$
		T2CCU0 output	T2CCU0_3		$1X101_B$
		Reserved	-		$1X110_B$
		Reserved	-		$1X111_B$
	DIR	Debug output 1; ENx	DAP1_1/SPD_1	HW_OUT	
SCR_P0.4 (P33.4)	I	General-purpose input	SCR_P0_IN.P4	SCR_P0_IOCR4.PC	$0XXXX_B$
		UART input	UART_RXDC		
		WCAN input	WCAN_RXDP		
		T1 input	T1B		
		External interrupt 6 / T2CCU0 input	EXINT6D/T2CCU0_3D		
		External interrupt 0	EXINT0A		
		External interrupt 12	EXINT12C		
		ADCOMP input	ADCOMPCH4		
	O	General-purpose output	SCR_P0_OUT.P4		$1X000_B$
		UART output	UART_RXD_O		$1X001_B$
		Reserved	-		$1X010_B$
		Reserved	-		$1X011_B$
		T2CCU1 output	T21EXF2		$1X100_B$
		T2CCU0 output	T2CCU0_4		$1X101_B$

(table continues...)

Table 1702 (continued) Port P0 functions

Port pin	I/O	Pin functionality	Associated reg./I/O line	Port I/O control selection	
				Reg./bit-field	Value
		Reserved	-		$1X110_B$
		Reserved	-		$1X111_B$
SCR_P0.5 (P33.5)	I	General-purpose input	SCR_P0_IN.P5	SCR_P0_IOCR5.PC	$0XXXX_B$
		WCAN input	WCAN_RXDD		
		SSC input	SSC_MRSTA		
		T0 input	T0B		
		T2CCU0 external input	T20EXB		
		External interrupt 1	EXINT1A		
		External interrupt 13	EXINT13C		
		ADCOMP input	ADCOMPCH5		
	O	General-purpose output	SCR_P0_OUT.P5		$1X000_B$
		Reserved	-		$1X001_B$
		SSC output	SSC_MRST		$1X010_B$
		Reserved	-		$1X011_B$
		Reserved	-		$1X100_B$
		T2CCU0 output	T2CCU0_5		$1X101_B$
		Reserved	-		$1X110_B$
		Reserved	-		$1X111_B$
	DIR	PMS debugging power mode K output	PMS_DEBUG[0]	HWOUT	
SCR_P0.6 (P33.6)	I	General-purpose input	SCR_P0_IN.P6	SCR_P0_IOCR6.PC	$0XXXX_B$
		UART input	UART_RXDL		
		SSC input	SSC_MTSRA		
		Debug input	DAP0_0		
		External interrupt 8 / T2CCU1 input	EXINT8B/T2CCU1_0B		
		External interrupt 2	EXINT2A		
		ADCOMP input	ADCOMPCH6		
	O	General-purpose output	SCR_P0_OUT.P6		$1X000_B$
		UART output	UART_RXD_O		$1X001_B$
		SSC output	SSC_MTSR		$1X010_B$
		T2CCU0 output	T20EXF2		$1X011_B$
		Reserved	-		$1X100_B$

(table continues...)

Table 1702 (continued) Port P0 functions

Port pin	I/O	Pin functionality	Associated reg./I/O line	Port I/O control selection	
				Reg./bit-field	Value
		Reserved	-		$1X101_B$
		T2CCU1 output	T2CCU1_0		$1X110_B$
	DIR	PMS debugging power mode K output	PMS_DEBUG[1]	HW_OUT	
SCR_P0.7 (P33.7)	I	General-purpose input	SCR_P0_IN.P7	SCR_P0_IOCR7.PC	$0XXXX_B$
		WCAN input	WCAN_RXDB		
		SSC input	SSC_SCLKA		
		Debug input	DAP1_0/SPD_0		
		T2CCU0 input	T20B		
		T2CCU1 input	T21EXH		
		External interrupt 9 / T2CCU1 input	EXINT9B/T2CCU1_1B		
		External interrupt 0	EXINT0B		
		ADCOMP input	ADCOMPCH7		
	O	General-purpose output	SCR_P0_OUT.P7		$1X000_B$
		UART output	UART_TXD		$1X001_B$
		SSC output	SSC_SCLK		$1X010_B$
		Reserved	-		$1X011_B$
		Reserved	-		$1X100_B$
		Reserved	-		$1X101_B$
		T2CCU1 output	T2CCU1_1		$1X110_B$
		Reserved	-		$1X111_B$
	DIR	PMS debugging power mode K output	PMS_DEBUG[2]	HW_OUT	
	DIR	Debug output 1; ENx	DAP1_0/SPD_0	HW_OUT	

39.3.7.4 SCR functions in port SCR_P1

This section describes the SCR functions of Port SCR_P1, which is mapped on P33.8-P33.15.

The SCR has functions on Port lines SCR_P1.0 - SCR_P1.7, which can be alternatively used for I2C, UART, WCAN, SSC, T0/T1, T2CCU0/1, ADCOMP, Pin Reset, single pin 32 kHz clock input, External NMI and External Interrupt input and output functions.

Following table summarizes the SCR I/O control selection functions of the 8 Port SCR_P1 lines.

Table 1703 Port P1 functions

Port pin	I/O	Pin functionality	Associated reg./I/O line	Port I/O control selection	
				Reg./bit-field	Value
SCR_P1.0 (P33.8)	I	General-purpose input	SCR_P1_IN.P0	SCR_P1_IOCR0.PC	$0XXXX_B$
		UART input	UART_RXDF		
		WCAN input	WCAN_RXDO		
		T2CCU0 external input	T21EXN		
		External interrupt 11 / T2CCU1 input	EXINT11D/T2CCU1_3D		
		External interrupt 1	EXINT1B		
		I2C data input	I2CSDAD		
	O	General-purpose output	SCR_P1_OUT.P0		$1X000_B$
		Reserved	-		$1X001_B$
		Reserved	-		$1X010_B$
		Reserved	-		$1X011_B$
		Reserved	-		$1X100_B$
		Reserved	-		$1X101_B$
		Reserved	-		$1X110_B$
		I2C data output	I2CSDA		$1X111_B$
SCR_P1.1 (P33.9)	I	General-purpose input	SCR_P1_IN.P1	SCR_P1_IOCR1.PC	$0XXXX_B$
		T1 input	T1A		
		T2CCU0 external input	T20EXC		
		External interrupt 3 / T2CCU0 input	EXINT3B/T2CCU0_0B		
		External interrupt 14	EXINT14C		
		I2C clock input	I2CSCLD		
	O	General-purpose output	SCR_P1_OUT.P1		$1X000_B$
		UART output	UART_TXD		$1X001_B$
		Reserved	-		$1X010_B$
		Reserved	-		$1X011_B$
		Reserved	-		$1X100_B$
		T2CCU0 output	T2CCU0_0		$1X101_B$
		Reserved	-		$1X110_B$
		I2C clock output	I2CSCL		$1X111_B$
SCR_P1.2 (P33.10)	I	General-purpose input	SCR_P1_IN.P2	SCR_P1_IOCR2.PC	$0XXXX_B$
		UART input	UART_RXDA		

(table continues...)

Table 1703 (continued) Port P1 functions

Port pin	I/O	Pin functionality	Associated reg./I/O line	Port I/O control selection	
				Reg./bit-field	Value
		WCAN input	WCAN_RXDA		
		External clock input to RTC	RTC32_IN		
		T2CCU0 external input	T20EXP		
		External interrupt 2	EXINT2B		
		External interrupt 15	EXINT15C		
		T2CCU1 external input	T21CLK32K		
	O	General-purpose output	SCR_P1_OUT.P2		$1X000_B$
		UART output	UART_RXDO		$1X001_B$
		Reserved	-		$1X010_B$
		Reserved	-		$1X011_B$
		Reserved	-		$1X100_B$
		Reserved	-		$1X101_B$
		Reserved	-		$1X110_B$
		Reserved	-		$1X111_B$
SCR_P1.3 (P33.11)	I	General-purpose input	SCR_P1_IN.P3	SCR_P1_IOCR3.PC	$0XXXX_B$
		SSC input	SSC_SCLKB		
		Pin Reset input	PINRSTA		
		T2CCU0 input	T20C		
		External interrupt 4 / T2CCU0 input	EXINT4B/T2CCU0_1B		
		External interrupt 10 / T2CCU1 input	EXINT10B/T2CCU1_2B		
	O	General-purpose output	SCR_P1_OUT.P3		$1X000_B$
		Reserved	-		$1X001_B$
		SSC output	SSC_SCLK		$1X010_B$
		Reserved	-		$1X011_B$
		Reserved	-		$1X100_B$
		T2CCU0 output	T2CCU0_1		$1X101_B$
		T2CCU1 output	T2CCU1_2		$1X110_B$
		Reserved	-		$1X111_B$
SCR_P1.4 (P33.12)	I	General-purpose input	SCR_P1_IN.P4	SCR_P1_IOCR4.PC	$0XXXX_B$
		UART input	UART_RXDG		

(table continues...)

Table 1703 (continued) Port P1 functions

Port pin	I/O	Pin functionality	Associated reg./I/O line	Port I/O control selection	
				Reg./bit-field	Value
		WCAN input	WCAN_RXDG		
		SSC input	SSC_MTSRB		
		T1 input	T1C		
		T2CCU1 input	T21C		
		External interrupt 5 / T2CCU0 input	EXINT5B/T2CCU0_2B		
		External interrupt 11 / T2CCU1 input	EXINT11B/T2CCU1_3B		
	O	General-purpose output	SCR_P1_OUT.P4		$1X000_B$
		UART output	UART_TXD		$1X001_B$
		SSC output	SSC_MTSR		$1X010_B$
		Reserved	-		$1X011_B$
		Reserved	-		$1X100_B$
		T2CCU0 output	T2CCU0_2		$1X101_B$
		T2CCU1 output	T2CCU1_3		$1X110_B$
		Reserved	-		$1X111_B$
SCR_P1.5 (P33.13)	I	General-purpose input	SCR_P1_IN.P5	SCR_P1_IOCR5.PC	$0XXXX_B$
		UART input	UART_RXDB		
		WCAN input	WCAN_RXDC		
		SSC input	SSC_MRSTB		
		T0 input	T0C		
		T2CCU0 external input	T20EXD		
		External interrupt 6 / T2CCU0 input	EXINT6B/T2CCU0_3B		
		External interrupt 0	EXINT0C		
		ADCOMP input	ADCOMPCH8		
	O	General-purpose output	SCR_P1_OUT.P5		$1X000_B$
		UART output	UART_RXDO		$1X001_B$
		SSC output	SSC_MRST		$1X010_B$
		Reserved	-		$1X011_B$
		Reserved	-		$1X100_B$
		T2CCU0 output	T2CCU0_3		$1X101_B$
		T2CCU1 output	T2CCU1_4		$1X110_B$

(table continues...)

Table 1703 (continued) Port P1 functions

Port pin	I/O	Pin functionality	Associated reg./I/O line	Port I/O control selection	
				Reg./bit-field	Value
		Reserved	-		$1X111_B$
SCR_P1.6 (P33.14)	I	General-purpose input	SCR_P1_IN.P6	SCR_P1_IOCR6.PC	$0XXXX_B$
		SSC input	SSC_SCLKC		
		External NMI	EXTNMI		
		T2CCU1 external input	T21EXC		
		External interrupt 1	EXINT1C		
		ADCOMP input	ADCOMPCH9		
	O	General-purpose output	SCR_P1_OUT.P6		$1X000_B$
		UART output	UART_TXD		$1X001_B$
		SSC output	SSC_SCLK		$1X010_B$
		Reserved	-		$1X011_B$
		Reserved	-		$1X100_B$
		Reserved	-		$1X101_B$
		T2CCU1 output	T2CCU1_5		$1X110_B$
		Reserved	-		$1X111_B$
SCR_P1.7 (P33.15)	I	General-purpose input	SCR_P1_IN.P7	SCR_P1_IOCR7.PC	$0XXXX_B$
		UART input	UART_RXDE		
		Pin Reset input	PINRSTB		
		T2CCU0 external input	T20EXE		
		External interrupt 3 / T2CCU0 input	EXINT3C/T2CCU0_0C		
		External interrupt 8 / T2CCU1 input	EXINT8C/T2CCU1_0C		
		External interrupt 2	EXINT2C		
		ADCOMP input	ADCOMPCH10		
	O	General-purpose output	SCR_P1_OUT.P7		$1X000_B$
		UART output	UART_RXDO		$1X001_B$
		Reserved	-		$1X010_B$
		Reserved	-		$1X011_B$
		Reserved	-		$1X100_B$
		T2CCU0 output	T2CCU0_0		$1X101_B$
		T2CCU1 output	T2CCU1_0		$1X110_B$
		Reserved	-		$1X111_B$

39.3.7.5 SCR functions in port SCR_P2

This section describes the SCR functions of Port SCR_P2, which is mapped on P32.2, P32.4-P32.7 and P35.0-P35.2.

The SCR has functions on Port lines SCR_P2.0 - SCR_P2.7, which can be alternatively used for I2C, UART, WCAN, SSC, T0/T1, T2CCU0/1, Pin Reset, 32 kHz PMS RTC clock output and External Interrupt input and output functions.

Following table summarizes the SCR I/O control selection functions of the 8 Port SCR_P2 lines.

Table 1704 Port P2 functions

Port pin	I/O	Pin functionality	Associated reg./I/O line	Port I/O control selection	
				Reg./bit-field	Value
SCR_P2.0 (P32.2)	I	General-purpose input	SCR_P2_IN.P0	SCR_P2_IOCR0.PC	$0XXXX_B$
		UART input	UART_RXDH		
		WCAN input	WCAN_RXDF		
		T0 input	T0D		
		T2CCU1 external input	T21EXG		
		External interrupt 7	EXINT7		
	O	General-purpose output	SCR_P2_OUT.P0		$1X000_B$
		UART output	UART_TXD		$1X001_B$
		Reserved	-		$1X010_B$
		Reserved	-		$1X011_B$
		Reserved	-		$1X100_B$
		T2CCU0 output	T2CCU0_4		$1X101_B$
		Reserved	-		$1X110_B$
		Reserved	-		$1X111_B$
SCR_P2.1 (P32.4)	I	General-purpose input	SCR_P2_IN.P1	SCR_P2_IOCR1.PC	$0XXXX_B$
		SSC input	SSC_MRSTD		
		T2CCU1 external input	T21EXA		
		External interrupt 8 / T2CCU1 input	EXINT8A/T2CCU1_0A		
		External interrupt 12	EXINT12A		
		I2C clock input	I2CSCLA		
	O	General-purpose output	SCR_P2_OUT.P1		$1X000_B$
		Reserved	-		$1X001_B$
		SSC output	SSC_MRST		$1X010_B$
		T2CCU0 output	T20EXF2		$1X011_B$
		Reserved	-		$1X100_B$
		T2CCU0 output	T2CCU0_5		$1X101_B$

(table continues...)

Table 1704 (continued) Port P2 functions

Port pin	I/O	Pin functionality	Associated reg./I/O line	Port I/O control selection	
				Reg./bit-field	Value
		T2CCU1 output	T2CCU1_0		$1X110_B$
		I2C clock output	I2CSCL		$1X111_B$
SCR_P2.2 (P32.5)	I	General-purpose input	SCR_P2_IN.P2	SCR_P2_IOCR2.PC	$0XXXX_B$
		SSC input	SSC_MTSRD		
		Pin Reset input	PINRSTC		
		T2CCU0 external input	T20EXO		
		External interrupt 9 / T2CCU1 input	EXINT9A/T2CCU1_1A		
		External interrupt 13	EXINT13A		
	O	General-purpose output	SCR_P2_OUT.P2		$1X000_B$
		UART output	UART_TXD		$1X001_B$
		SSC output	SSC_MTSR		$1X010_B$
		Reserved	-		$1X011_B$
		T2CCU1 output	T21EXF2		$1X100_B$
		Reserved	-		$1X101_B$
		T2CCU1 output	T2CCU1_1		$1X110_B$
		Reserved	-		$1X111_B$
SCR_P2.3 (P32.6)	I	General-purpose input	SCR_P2_IN.P3	SCR_P2_IOCR3.PC	$0XXXX_B$
		UART input	UART_RXDI		
		T1 input	T1D		
		T2CCU1 input	T21A		
		External interrupt 10 / T2CCU1 input	EXINT10A/T2CCU1_2A		
		External interrupt 14	EXINT14A		
		I2C data input	I2CSDAA		
	O	General-purpose output	SCR_P2_OUT.P3		$1X000_B$
		UART output	UART_RXD_O		$1X001_B$
		Reserved	-		$1X010_B$
		Reserved	-		$1X011_B$
		Reserved	-		$1X100_B$
		Reserved	-		$1X101_B$
		T2CCU1 output	T2CCU1_2		$1X110_B$
		I2C data output	I2CSDA		$1X111_B$

(table continues...)

Table 1704 (continued) Port P2 functions

Port pin	I/O	Pin functionality	Associated reg./I/O line	Port I/O control selection	
				Reg./bit-field	Value
SCR_P2.4 (P32.7)	I	General-purpose input	SCR_P2_IN.P4	SCR_P2_IOCR4.PC	$0XXXX_B$
		WCAN input	WCAN_RXDH		
		T2CCU0 external input	T20EXH		
		External interrupt 11 / T2CCU1 input	EXINT11A/T2CCU1_3A		
		External interrupt 15	EXINT15A		
	O	General-purpose output	SCR_P2_OUT.P4		$1X000_B$
		UART output	UART_TXD		$1X001_B$
		Reserved	-		$1X010_B$
		Reserved	-		$1X011_B$
		Reserved	-		$1X100_B$
		Reserved	-		$1X101_B$
		T2CCU1 output	T2CCU1_3		$1X110_B$
		PMS RTC 32 kHz (XTA3/4) clock output	PMSRTC32_OUT		$1X111_B$
SCR_P2.5 (P35.0)	I	General-purpose input	SCR_P2_IN.P5	SCR_P2_IOCR5.PC	$0X000_B$
		WCAN input	WCAN_RXDK		
		T2CCU1 external input	T21EXL		
		External interrupt 8 / T2CCU1 input	EXINT8D/T2CCU1_0D		
		I2C data input	I2CSDAC		
	O	General-purpose output	SCR_P2_OUT.P5		$1X000_B$
		UART output	UART_TXD		$1X001_B$
		Reserved			$1X010_B$
		T2CCU0 output	T20EXF2		$1X011_B$
		Reserved			$1X100_B$
		T2CCU0 output	T2CCU0_1		$1X101_B$
		Reserved			$1X110_B$
		I2C data output	I2CSDA		$1X111_B$
SCR_P2.6 (P35.1)	I	General-purpose input	SCR_P2_IN.P6	SCR_P2_IOCR6.PC	$0X000_B$
		WCAN input	WCAN_RXDL		
		T2CCU1 external input	T21EXM		
		External interrupt 3 / T2CCU0 input	EXINT3D/T2CCU0_0D		

(table continues...)

Table 1704 (continued) Port P2 functions

Port pin	I/O	Pin functionality	Associated reg./I/O line	Port I/O control selection	
				Reg./bit-field	Value
		External interrupt 12	EXINT12D		
		I2C clock input	I2CSCLC		
	O	General-purpose output	SCR_P2_OUT.P6		$1X000_B$
		UART output	UART_TXD		$1X001_B$
		Reserved			$1X010_B$
		Reserved			$1X011_B$
		T2CCU1 output	T21EXF2		$1X100_B$
		T2CCU0 output	T2CCU0_2		$1X101_B$
		Reserved			$1X110_B$
		I2C clock output	I2CSCL		$1X1111_B$
SCR_P2.7 (P35.2)	I	General-purpose input	SCR_P2_IN.P7	SCR_P2_IOCR7.PC	$0X000_B$
		UART input	UART_RXDM		
		T2CCU0 external input	T20EXM		
		External interrupt 9 / T2CCU1 input	EXINT9D/T2CCU1_1D		
		External interrupt 13	EXINT13D		
	O	General-purpose output	SCR_P2_OUT.P7		$1X000_B$
		UART output	UART_RXD_O		$1X001_B$
		Reserved			$1X010_B$
		Reserved			$1X011_B$
		Reserved			$1X100_B$
		T2CCU0 output	T2CCU0_3		$1X101_B$
		Reserved			$1X110_B$
		Reserved			$1X111_B$

39.3.7.6 SCR functions in port SCR_P3

This section describes the SCR functions of Port SCR_P3, which is mapped on P34.1-P34.5 and P35.3-P35.5.

The SCR has functions on Port lines SCR_P3.0 - SCR_P3.7, which can be alternatively used for UART, WCAN, SSC, T0/T1, T2CCU0/1, ADCOMP, PMS power-mode debug and External Interrupt input and output functions.

Following table summarizes the SCR I/O control selection functions of the 8 Port SCR_P3 lines.

Table 1705 Port P3 functions

Port pin	I/O	Pin functionality	Associated reg./I/O line	Port I/O control selection	
				Reg./bit-field	Value
SCR_P3.0 (P34.1)	I	General-purpose input	SCR_P3_IN.P0	SCR_P3_IOCR0.PC	$0XXXX_B$
		UART input	UART_RXDD		
		T2CCU1 external input	T21EXD		
		External interrupt 4 / T2CCU0 input	EXINT4C/T2CCU0_1C		
		External interrupt 9 / T2CCU1 input	EXINT9C/T2CCU1_1C		
		ADCOMP input	ADCOMPCH11		
	O	General-purpose output	SCR_P3_OUT.P0		$1X000_B$
		UART output	UART_RXD_O		$1X001_B$
		Reserved	-		$1X010_B$
		T2CCU0 output	T20EXF2		$1X011_B$
		Reserved	-		$1X100_B$
		T2CCU0 output	T2CCU0_1		$1X101_B$
		T2CCU1 output	T2CCU1_1		$1X110_B$
		Reserved	-		$1X111_B$
SCR_P3.1 (P34.2)	I	General-purpose input	SCR_P3_IN.P1	SCR_P3_IOCR1.PC	$0XXXX_B$
		WCAN input	WCAN_RXDI		
		T2CCU0 external input	T20EXF		
		External interrupt 5 / T2CCU0 output	EXINT5C/T2CCU0_2C		
		ADCOMP input	ADCOMPCH12		
		I2C data input	I2CSDAB		
	O	General-purpose output	SCR_P3_OUT.P1		$1X000_B$
		Reserved	-		$1X001_B$
		Reserved	-		$1X010_B$
		Reserved	-		$1X011_B$
		T2CCU1 output	T21EXF2		$1X100_B$
		T2CCU0 output	T2CCU0_2		$1X101_B$
		T2CCU1 output	T2CCU1_2		$1X110_B$
		I2C data output	I2CSDA		$1X111_B$
SCR_P3.2 (P34.3)	I	General-purpose input	SCR_P3_IN.P2	SCR_P3_IOCR2.PC	$0XXXX_B$
		Pin Reset input	PINRSTD		

(table continues...)

Table 1705 (continued) Port P3 functions

Port pin	I/O	Pin functionality	Associated reg./I/O line	Port I/O control selection	
				Reg./bit-field	Value
		T2CCU1 external input	T21EXE		
		External interrupt 5 / T2CCU0 input	EXINT6C/T2CCU0_3C		
		External interrupt 0	EXINT0D		
		ADCOMP input	ADCOMPCH13		
		I2C clock input	I2CSCLB		
	O	General-purpose output	SCR_P3_OUT.P2		$1X000_B$
		Reserved	-		$1X001_B$
		Reserved	-		$1X010_B$
		T2CCU0 output	T20EXF2		$1X011_B$
		Reserved	-		$1X100_B$
		T2CCU0 output	T2CCU0_3		$1X101_B$
		T2CCU1 output	T2CCU1_3		$1X110_B$
		I2C clock output	I2CSCL		$1X111_B$
SCR_P3.3 (P34.4)	I	General-purpose input	SCR_P3_IN.P3	SCR_P3_IOCR3.PC	$0XXXX_B$
		UART input	UART_RXDJ		
		SSC input	SSC_MRSTC		
		T2CCU0 external input	T20EXG		
		External interrupt 10 / T2CCU1 input	EXINT10C/T2CCU1_2C		
		External interrupt 1	EXINT1D		
		ADCOMP input	ADCOMPCH14		
	O	General-purpose output	SCR_P3_OUT.P3		$1X000_B$
		UART output	UART_RXD_O		$1X001_B$
		SSC output	SSC_MRST		$1X010_B$
		Reserved	-		$1X011_B$
		T2CCU1 output	T21EXF2		$1X100_B$
		T2CCU0 output	T2CCU0_4		$1X101_B$
		T2CCU1 output	T2CCU1_4		$1X110_B$
		Reserved	-		$1X111_B$
SCR_P3.4 (P34.5)	I	General-purpose input	SCR_P3_IN.P4	SCR_P3_IOCR4.PC	$0XXXX_B$
		WCAN input	WCAN_RXDJ		
		SSC input	SSC_MTSRC		

(table continues...)

Table 1705 (continued) Port P3 functions

Port pin	I/O	Pin functionality	Associated reg./I/O line	Port I/O control selection	
				Reg./bit-field	Value
		T2CCU1 external input	T21EXF		
		External interrupt 11 / T2CCU1 input	EXINT11C/T2CCU1_3C		
		External interrupt 2	EXINT2D		
		ADCOMP input	ADCOMPCH15		
	O	General-purpose output	SCR_P3_OUT.P4		$1X000_B$
		UART output	UART_TXD		$1X001_B$
		SSC output	SSC_MTSR		$1X010_B$
		Reserved	-		$1X011_B$
		Reserved	-		$1X100_B$
		T2CCU0 output	T2CCU0_5		$1X101_B$
		T2CCU1 output	T2CCU1_5		$1X110_B$
		Reserved	-		$1X111_B$
SCR_P3.5 (P35.3)	I	General-purpose input	SCR_P3_IN.P5	SCR_P3_IOCR5.PC	$0XXXX_B$
		UART input	UART_RXDN		
		WCAN input	WCAN_RXDM		
		T2CCU0 external input	T20EXN		
		External interrupt 10 / T2CCU1 input	EXINT10D/T2CCU1_2D		
		External interrupt 14	EXINT14D		
	O	General-purpose output	SCR_P3_OUT.P5		$1X000_B$
		UART output	UART_RXD_O		$1X001_B$
		Reserved			$1X010_B$
		Reserved			$1X011_B$
		Reserved			$1X100_B$
		T2CCU0 output	T2CCU0_4		$1X101_B$
		Reserved			$1X110_B$
		Reserved			$1X111_B$
SCR_P3.6 (P35.4)	I	General-purpose input	SCR_P3_IN.P6	SCR_P3_IOCR6.PC	$0XXXX_B$
		T2CCU0 input	T20D		
		T2CCU1 external input	T21EXO		
		External interrupt 4 / T2CCU0 input	EXINT4D/T2CCU0_1D		

(table continues...)

Table 1705 (continued) Port P3 functions

Port pin	I/O	Pin functionality	Associated reg./I/O line	Port I/O control selection	
				Reg./bit-field	Value
		External interrupt 15	EXINT15D		
	O	General-purpose output	SCR_P3_OUT.P6		$1X000_B$
		UART output	UART_TXD		$1X001_B$
		Reserved			$1X010_B$
		Reserved			$1X011_B$
		Reserved			$1X100_B$
		T2CCU0 output	T2CCU0_5		$1X101_B$
		T2CCU1 output	T2CCU1_4		$1X110_B$
		Reserved			$1X111_B$
SCR_P3.7 (P35.5)	I	General-purpose input	SCR_P3_IN.P7	SCR_P3_IOCR7.PC	$0XXXX_B$
		UART input	UART_RXDO		
		WCAN input	WCAN_RXDN		
		T2CCU1 input	T21D		
		T2CCU1 external input	T21EXP		
		External interrupt 5 / T2CCU0 output	EXINT5D/T2CCU0_2D		
	O	General-purpose output	SCR_P3_OUT.P7		$1X000_B$
		UART output	UART_RXD_O		$1X001_B$
		Reserved			$1X010_B$
		Reserved			$1X011_B$
		Reserved			$1X100_B$
		Reserved			$1X101_B$
		T2CCU1 output	T2CCU1_5		$1X110_B$
		Reserved			$1X111_B$

39.3.7.7 General port control

There are a number of control registers for input pin selection:

- Selection of inputs for SSC, UART, T0, and T1 are made in:
  - SCUSCR_MODPISEL0
  - 1
- Inputs for T2 are selected in:
  - T2CCUx_T2_CON1.INSEL and .EXTINSEL
- WCAN input selection is made in:
  - WCAN_CFG.WCANRXDIS bit-field
- External interrupts are selected in:
  - IRSCR_PISEL0/1/2/3
- Inputs for I2CSCR are selected through:
  - I2CSCR_SRST.SCLSEL
  - I2CSCR_SRST.SDASEL

The following table provides an overview of these registers.

Table 1706 Registers overview - input pin selection (sorted by name)

Short name	Description	RMAP	PAGE	Offset address	Page number
IRSCR_PISEL0	External Interrupt Input Select Register 0	0	IRSCR_PAGE	$F7_H$	4
IRSCR_PISEL1	External Interrupt Input Select Register 1	0	IRSCR_PAGE	$F8_H$	4
IRSCR_PISEL2	External Interrupt Input Select Register 2	0	IRSCR_PAGE	$F9_H$	4
IRSCR_PISEL3	External Interrupt Input Select Register 3	0	IRSCR_PAGE	$F7_H$	5
SCUSCR_MODPISEL0	Peripheral Input Select Register 0 (UART,T0/1)	0	SCUSCR_PAGE	$F0_H$	2
SCUSCR_MODPISEL1	Peripheral Input Select Register 1 (PINRST, SSC)	0	SCUSCR_PAGE	$F1_H$	2
T2CCU0_T2_CON1.INSEL	Timer 2 Instance 0 Input Select	0	T2CCU0_PAGE	87_H	0
T2CCU0_T2_CON1.EXTINSEL	Timer 2 Instance 0 External Input Select	0	T2CCU0_PAGE	87_H	0
T2CCU1_T2_CON1.INSEL	Timer 2 Instance 1 Input Select	0	T2CCU1_PAGE	$C5_H$	0
T2CCU1_T2_CON1.EXTINSEL	Timer 2 Instance 1 External Input Select	0	T2CCU1_PAGE	$C5_H$	0
WCAN_CFG.WCANRXDIS	WCAN Receiver Input Select	0	WCAN_PAGE	$B0_H$	0
I2CSCR_SRST.SCLSEL	I2CSCR clock input select	0		FD_H	
I2CSCR_SRST.SDASEL	I2CSCR data input select	0		FD_H	

39.3.8 Real time clock (RTCSCR)

39.3.8.1 Overview

One of the SCR's peripherals is the Real-Time Clock (RTCSCR) that, once started, can work independently of the state of the rest of the microcontroller.

Features

- Periodic wake-up mode using several possible clock sources
- Wake-up source during standby mode

39.3.8.2 RTCSCR operation

The real-time clock consists of a 41-bit timer (including 9-bit pre-scaler) which count up. It contains a set of 4 count registers, collectively know at RTCSCR_CNT register, that shows the current count value or the current time of the real-time clock. Another set of registers, RTCSCR_CR register, that consists of 4 registers can be used for interrupt generation. It can also be used to wake-up device from standby mode. The RTCSCR_CR register is also used to store the capture value when a capture event is triggered.

The RTCSCR starts counting after 1 PCLK + 4 RTCSCR Clock cycles after the RTCC is set.

***Note**: Before starting the RTCSCR counter, the compare value in registers RTCSCR_CRn must be set to a value unequal 0 (for example $FFFF_H$). Otherwise, the RTCSCR will not start, if it is initially 0, or stop after an overflow to 0.*

39.3.8.2.1 Periodic wake-up mode

The figure below shows the real-time clock. It consists of a 41-bit general purposes timer. The f_{RTCSCR} is selected through RTCSCR_CON.RTCCLKSEL.

Before real-time clock is started, RTCSCR_CNT register of the real-time clock can be written with any value. The value written is used as an initial value for the real-time clock timer, when it is started by setting bit RTCSCR_CON.RTCC to 1. This bit also enables the input clock into the real-time clock timer. RTCSCR_CNT register is protected by the PASSWD bit protection scheme as described in the SCUSCR chapter. The initial count value can only be updated when the protection scheme is being disabled in the stop mode (RTCSCR_CON.RTCC=0). The real-time clock's lower 9 bits, which serve as a pre-scaler into the 32-bit counters, RTCSCR_CNT, are set to an initial value of 000000000_B when the RTCSCR starts to run. The pre-scaler can be bypassed by setting bit RTCSCR_CON.RTPBYP to 1.

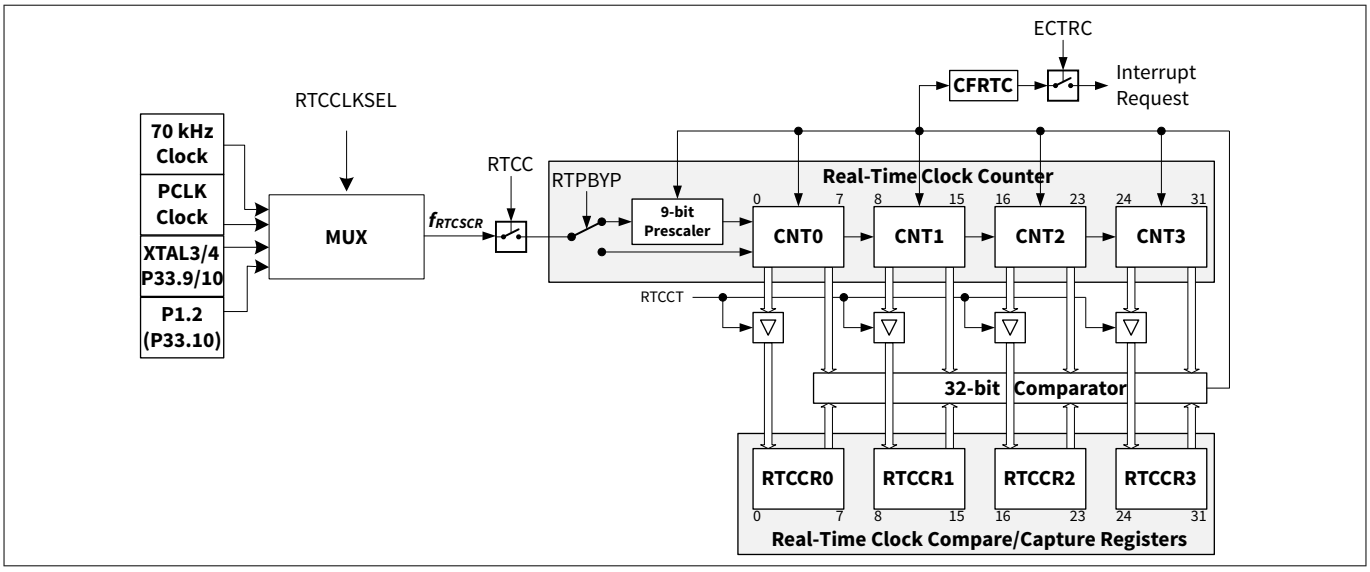


Figure 874 Real-time clock periodic wake-up mode with 9-bit pre-scaler

While the real-time clock is in operation, the contents of RTCSCR_CR register will be compared to the real-time clock counter (RTCSCR_CNT). An interrupt will be generated when the contents are equal if ECRTC is set to 1; bit RTCSCR_CON.CFRTCSCR will be set. The RTCSCR_CON.CFRTC flag can be monitored for the real-time clock wake-up request, but the flag has to be cleared to 0 by user software. In such situation, the real-time clock counter is reset and starts counting from zero again.

***Note**: Before starting the RTCSCR counter, the compare value in registers RTCSCR_CRn must be set to a value unequal 0 (for example $FFFF_H$). Otherwise, the RTCSCR will not start if it is initially 0, or stop after an overflow to 0.*

As RTCSCR still running in standby mode, it is able to wake-up at a fixed time by programming the number of count value that is equivalent to the length of time to wake-up in the RTCSCR_CR register. The real-time clock can generate a wake-up request to the main controller, if bit SCUSCR_WKPCON.RTCWKSEL is set to 1.

A capture event could be triggered by setting RTCSCR_CON.RTCCT bit to 1. The previous content in the RTCSCR_CR register will be overwritten with the captured RTCSCR_CNT values after a maximum of 9 f_{RTCSCR} cycles. An update of the actual compared value is necessary once a captured event is triggered. In SCR, it is recommended to trigger a capture event to read the value of the RTCSCR counter (RTCSCR_CNT). There is the potential of reading a wrong 32-bit real-time value while the RTCSCR is in running mode as only 8 bit of data could be fetched at one time.

The real-time clock stops counting and RTCSCR_CNT register holds the last value when the RTCSCR_CON.RTCC bit is set to 0. Setting this bit subsequently will start a new counting sequence which begin with the stop count.

***Note**: For the capture event to complete, up to 9 f_{RTCSCR} cycles are required. During this time window, the compare event is temporarily disabled and no compare match happens. Once, the capture is completed and the RTCSCR_CR is updated, the compare event is enabled again. In both cases- i.e.- pre-scaler bypassed or not bypassed - it is ensured that the next compare match happens only after the next overflow.*

***Note**: The RTCSCR single input pin for clock is limited to 32.768 kHz.*

39.3.8.2.2 Clock source

RTCSCR clock, f_{RTCSCR}, is selectable between 4 clock sources:

- 70 kHz clock
- (100 MHz)/DIV clock (PCLK)
- XTAL3/4 oscillator clock, forwarded from PMS
- Externally supplied clock on pin SCR_P1.2 (P33.10)

Selection is done through bit-field RTCSCR_CON.RTCCLKSEL, which is write-enable protected by bit-field RTCSCR_CON.RTCCLKSEL_P.

Once powered up, these clocks can operate irrespective of the state of the microcontroller. That is, it keeps running even when the SCR has entered idle or the device has entered standby mode.

In case of clock failure, the RTCSCR stops counting.

39.3.8.2.3 RTCSCR access delays and restrictions

When accessing the RTCSCR registers, the following delays and restrictions need to be obeyed:

- When the SCR is running and the clock is stable, RTCSCR_CON.RTCSCRC is allowed to be set/cleared
- When the RTCSCR clock is changed between available clocks, the waiting time of four old/current clock cycles and four clocks of the new/selected clock cycles need to be inserted before setting RTCSCR_CON.RTCSCRC. Upon each clock selection change, it is required to insert up to 9 RTCSCR Clock cycles to the waiting time mentioned in the previous sentence
- When an SCR reset is triggered, the RTCSCR clock is set to 70 kHz per default. After triggering the reset, software has to wait 2PCLK + 3RTCSCR clock cycles, before bit SCUSCR_PMCON0.RTC_DIS may be cleared
- When bit SCUSCR_PMCON0.RTC_DIS is cleared, a waiting time of four RTCSCR clock cycles needs to be inserted before any write access to RTCSCR registers may be performed

When bypassing the 9-bit pre-scaler (RTCSCR_CON.RTPBYP=1)and consecutively reading the RTCSCR count registers, the following delays need to be taken into account due to synchronization:

- When the RTCSCR is running with PCLK, a delay of 6 PCLK cycles is present between the read accesses, so the RTCSCR count value between two read operations may differ by 6 counts
- When the RTCSCR is running with 70 kHz, a delay of 5RTCSCR clock cycles is present between the read accesses, so the RTCSCR count value between two read operations will differ by 5 counts

39.3.8.2.4 Register overview

Overview of RTCSCR registers is given in the following table. More details are available in the "Registers" sub-chapter.

Table 1707 Overview of RTCSCR registers

Register short name	Description	Address
RTCSCR_CON	Control and status register	$E1_H$
RTCSCR_CNTn (n=0-3)	Upper 32-bits of the 41-bit counter value	$E2_H$-$E5_H$
RTCSCR_CRn (n=0-3)	Capture and compare register	$E6_H$-$E9_H$

39.3.8.3 RTCSCR interrupt request

Following table lists the interrupt event sources from the RTCSCR, and the corresponding event interrupt enable bit and flag bit.

Table 1708 RTCSCR interrupt events

Event	Event interrupt enable bit	Event flag bit
RTCSCR compare/wake-up	RTCSCR_CON.ECRTC	RTCSCR_CON.CFRTC

Next table shows the interrupt node assignment for each RTCSCR interrupt source.

Table 1709 RTC events' interrupt node control

Event	Interrupt node enable bit	Interrupt node flag bit	Vector address
RTC compare/wake-up	IEN1.ECCIP3	–	$6B_H$

The following figure illustrates the RTC interrupt request.

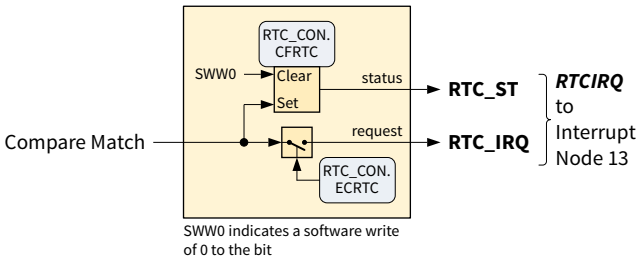


Figure 875 RTCSCR interrupt request

On a compare match event, if enabled through RTCSCR_CON.ECRTC, an RTC interrupt is signalized to the IRSCR through Interrupt Node 13. On the same event the status flag RTCSCR_CON.CFRTC is set. Once set, this flag should be cleared by software. For more details see also the section on "Interrupt Structure 2" in chapter "Interrupt System (IRSCR)".

39.3.8.4 Power saving mode

Once started, the real-time clock continues counting until the bit RTCSCR_CON.RTCC is cleared. The real-time clock is not affected by the idle mode of the SCR, and continues counting in standby mode provided the PCLK clock is available. In addition, the real-time clock will not automatically disable the operation when there is a clock failure.

39.3.8.4.1 Module suspend control

When the On-Chip Debug Support (OCDS) is in Monitor Mode (MMCR2.MMODE=1) and the Debug-Suspend signal is active (MMCR2.DSUSP=1), the timer/counter in RTCSCR module in the SCR can be suspended based on the settings of their corresponding module suspend bits in register SCUSCR_DBG_MODSUSP.

When suspended, only the timer stops counting as the counter input clock is gated off. The module is still clocked so that module registers are accessible.

39.3.9 PMS modules accessible from SCR

The SCR software can access certain sub-functions of the 64-bit Real-Time Clock (RTC) and Die-Temperature Sensor (DTS) located in PMS.

Besides RTC and DTS, the PMS HPOSC module also has the ability to be accessed by the SCR software through the SCU_PMSHPOSC_FTRIM registers (FTRIM7_0 and FTRIM8_8 are combined for 9-bits of value located inside PMS register PMS_HPOSCCON1.OSCFTRIM). This access can be used for temperature calibration of HPOSC from the SCR side, as a low-power alternative to doing calibration using software on the main SoC cores.

A detailed description of the 64-bit RTC as well as DTS modules are provided in respective PMS sub-chapter.

39.3.9.1 64-bit real-time clock

Besides the 41-bit RTCSCR located inside SCR sub-system, SCR software can also access certain subset functionality of 64-bit RTC located in the PMS module.

SCR can access PMS RTC only if the SCR is not in STANDBY1_70K state mode, meaning the SCR system clock (PCLK) is not 70 kHz. The SCR clock (PCLK) needs to be derived from the 100 MHz HPOSC clock. If the SCR clock is 70 kHz, reading to the below listed PMS RTC registers will always return zero and writing will be stalled until the 100 MHz clock is available. Shadow registers on the SCR side are available for write access even when SCR is operating at 70 kHz. However, PMS RTC registers will get updated only when the 100 MHz clock is available.

On the SCR side the following registers are available in the SCU address space:

Table 1710 PMS RTC bit-fields available for SCR software

PMS RTC bit-field name	SCR register name	Description	SCR software access
RTC_TIM1CAP[31:0] & RTC_TIM0[31:0]	SCUSCR_PMSRTC_TIM*	8 registers of 8-bits (TIM63_56, TIM55_48, …, TIM7_0) for a total of 64-bit timer value	Read-only
RTC_CON1.OFFSET[15:0]	SCUSCR_PMSRTC_OFFSET*	2 registers of 8-bits (OFFSET15_8 and OFFSET7_0) for a total of 16-bit offset value	Read and write
RTC_CMP1[31:0]	SCUSCR_PMSRTC_CMP1*	4 registers of 8-bits (CMP1_31_24, …, CMP1_7_0) for a total of 32-bits CMP1 value	Read and write
RTC_CON2.MSIZE1[4:0]	SCUSCR_PMSRTC_MSIZE1	5 least-significant bits of MSIZE1 value	Read and write
RTC_CON2.MSTART1[4:0]	SCUSCR_PMSRTC_MSTART1	5 least-significant bits of MSTART1 value	Read and write

Special notes on SCR access:

- While performing 8 read operations to read all 64-bits of TIM[63:0], the timer value can change in between those operations. It is recommended to read the most-significant byte first (SCUSCR_PMSRTC_TIM63_56) and then byte by byte toward least-significant byte (SCUSCR_PMSRTC_TIM7_0). If the value of the least significant byte is 00_H read again from SCUSCR_PMSRTC_TIM63_56 down to SCUSCR_PMSRTC_TIM15_8, as there is a chance that the lowest byte went from 0xFF to 0x00
- When writing MSIZE1 and MSTART1 values, the bit-fields in PMS RTC update with values provided by SCR software
- When writing OFFSET and CMP1 values, the bit-fields in PMS RTC update only after writing to the least-significant 8-bit register on SCR side
  - For example, writing SCUSCR_PMSRTC_CMP1_31_24 will be latched into SCR local shadow register, and it is copied to the respective RTC_CMP1[31:24] bit-field only after writing to

SCUSCR_PMSRTC_CMP1_7_0. It is not possible to read-back the value last written to local shadow register

***Note**: The PMS RTC registers are implemented in RTCSCR's clock domain (70 kHz/32 kHz), this is quite slow compared to the SCR's clock domain. Therefore, it takes some time for the PMS RTC registers to get updated with the new values. For example, if the RTC is clocked at 32 kHz it takes up to 62.5 µs until the registers are written and their new value can be read back. Therefore, the SCR software must wait until the PMS RTC registers are read back with their new value before a "standby 0" entry is requested. Otherwise the "standby 0" entry will invalidate the trimming of the RTC.*

PMS RTC outputs

Both CMP0 and CMP1 output lines (RTCCMP0 and RTCCMP1) are forwarded to the SCR as interrupt sources to interrupt node 7 and 8. The interrupt enable is located in IRSCR_MODIEN. The flags are located in the IRSCR_CON1 register. With these interrupt lines the SCR can fully utilize 64-bit PMS RTC as its own RTC.

In addition to the CMP outputs, the SCR also receives the 32 kHz clock from XTAL3/4 which also can be used by the PMS RTC.

39.3.9.2 Die-temperature sensor (DTS)

SCR software can also access some basic configuration and result registers of PMS DTS. Following registers are available in SCU address space:

Table 1711 PMS DTS bit-fields available for SCR software

PMS DTS bit-field name	SCR register name	Description	SCR software access
DTS_CON.START, DTS_CON.EN	SCUSCR_PMSDTS_CON.EN_START	DTS enable and start control is combined into a single bit	Read and write
DTS_STAT.RESULT[11:0]	SCUSCR_PMSDTS_RES*.VAL	Bit-fields in 2 registers combine for 12-bit result value	Read-only
DTS_STAT.READY	SCUSCR_PMSDTS_RES11_8.RES_READY	1-bit indication that temperature result is ready	Read-only

Special notes on SCR access:

***Note**: While performing 2 read operations to read all 12-bits of RESULT[11:0], the value can change in between those operations. It is recommended to read the lower 8-bits first, then the upper 4-bits and read again the lower 8-bits. If the new value of lower 8-bits is the same as on first read then keep the result, if not perform re-read of upper 4-bits.*

***Note**: Upon wake-up from STANDBY0, it is not possible to perform DTS calibration from SCR software side. This adds additional +-3LSB precision penalty on top of precision value listed in the datasheet.*

SCR only controls enabling and starting of the PMS DTS, and cannot explicitly disable it. But since DTS is located in the PMS1 domain it will be automatically switched off if the system transitions into STANDBY0 state.

SCR can enable and start the PMS DTS while in STANDBY1_70K mode. However, PMS DTS requires 100 MHz clock for its correct operation. Therefore, the SCR must select a PCLK faster than 70 kHz to use the DTS properly(SCR_SCUSCR_CLK.DIV_REQ!=DIV70kHz).

39.3.10 Timer 0 and timer 1 (T01)

39.3.10.1 Overview

Timer 0 and Timer 1 can function as both, timers or counters. When functioning as a timer, Timer 0 and Timer 1 are incremented at $f_{PCLK}/2$ (for example every 2 PCLKs). When functioning as a counter, Timer 0 and Timer 1 are incremented in response to a 1-to-0 transition (falling edge) at their respective external input pins, T0 or T1.

They are useful in many timing applications such as measuring the time interval between events, counting events and generating signals at regular intervals. In particular, Timer 1 can be used as the baudrate generator for the on-chip serial port.

Features:

- Four operational modes:
  - Mode 0: 13-bit timer/counter
  - Mode 1: 16-bit timer/counter
  - Mode 2: 8-bit timer/counter with auto-reload
  - Mode 3: Two 8-bit timers/counters

39.3.10.2 System information

This section provides system information relevant to the Timer 0 and 1.

39.3.10.2.1 Pinning

The following figure shows the connections and selections for the inputs to timers 0 and 1. Please see also the GPIO chapter (PORT8).

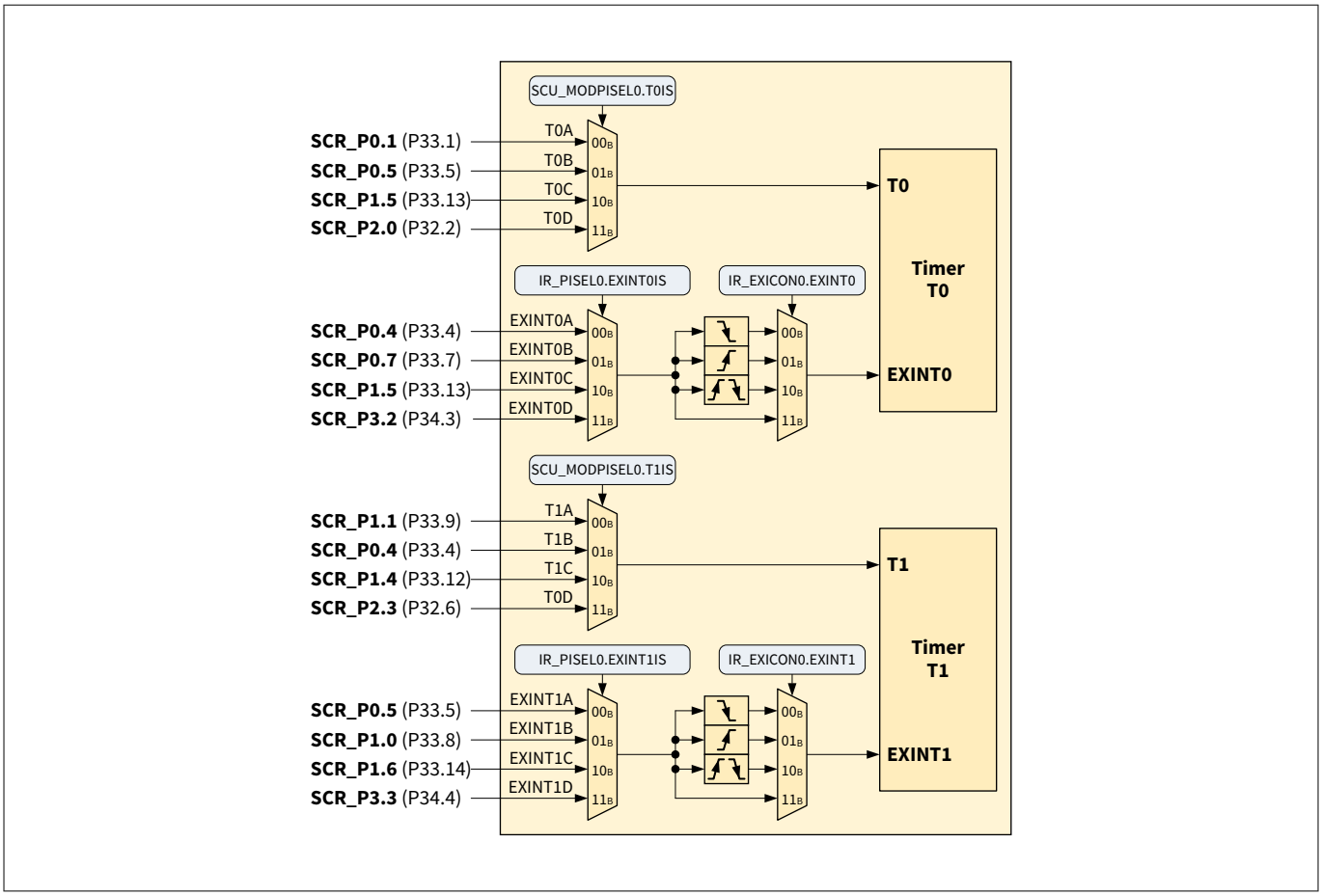


Figure 876 Timer 0 and 1 modules input signals

Related information

General purpose I/O Ports (PORT8) on page 6543

39.3.10.2.2 Clocking configuration

The Timer 0 and 1 run on the PCLK.

39.3.10.2.3 Interrupt events and assignment

The following table lists the interrupt event sources from Timer 0 and 1, and the interrupt node assignment for each Timer 0 and 1 interrupt source.

Table 1712 Timer 0 and 1 events' interrupt node control

Event	Interrupt node enable bit	Interrupt node flag bit	Vector address
External Interrupt 0	IEN0.EX0	T01_TCON.IE0	03_H
Timer 0 Overflow	IEN0.ET0	T01_TCON.TF0	$0B_H$
External Interrupt 1	IEN0.EX1	T01_TCON.IE1	13_H
Timer 1 Overflow	IEN0.ET1	T01_TCON.TF1	$1B_H$

39.3.10.3 Basic timer operations

The operations of the two timers are controlled using the Special Function Registers (SFRs) T01_TCON and T01_TMOD. To enable a timer, for example, to allow the timer to run, its control bit T01_TCON.TRx is set. To select the timer input to be either from internal system clock or external pin, the input selector bit T01_TMOD.TxS is used.

***Note**: The "x" (for example, T01_TCON.TRx) in this chapter denotes either 0 or 1.*

Each timer consists of two 8-bit registers - T01_TLx (low byte) and T01_THx (high byte) which defaults to $>00_H$ on reset. Setting or clearing T01_TCON.TRx does not affect the timer registers.

The timers 0 and 1 can be used as a counter, where the timer register is incremented in response to a 1-to-0 transition (falling edge) at the corresponding external input pin (T0 or T1). When the samples show a high for two PCLKs and a low in the next two PCLKs, the count is incremented. Since it takes four PCLKs to recognize a 1-to-0 transition, the maximum count rate is 1/4 of the PCLK frequency. There are no restrictions on the duty cycle of the external input signal, but it must be held at least two full PCLKs to ensure that a given level is sampled at least once before it changes.

Timer overflow

When a timer overflow occurs, the timer overflow flag, T01_TCON.TFx, is set, and an interrupt may be raised if the interrupt enable control bit, IEN0.ETx, is set. The overflow flag is automatically cleared when the interrupt service routine is entered. See also T0/T1 Interrupt Flags and Requests in related information..

When "Timer 0" operates in mode 3, the "Timer 1" control bits, TR1, TF1 and ET1 are reserved for TH0, see Mode 3 in related information.

External control

In addition to pure software control, the timers can also be enabled or disabled through external port control. When external port control is used, SFR EXICON0 must first be configured to bypass the edge detection circuitry for EXINTx to allow direct feed-through. When the timer is enabled (T01_TCON.TRx=1) and T01_TMOD.GATEx is set, the respective timer will only run if the core external interrupt EXINTx=1. This facilitates pulse width measurements. However, this is not applicable for "Timer 1" in mode 3.

If T01_TMOD.GATEx is cleared, the timer reverts to pure software control.

Related information

Mode 3 on page 6580

T0/T1 interrupt flags and requests on page 6581

39.3.10.4 Timer modes

Timers 0 and 1 are fully compatible and can be configured in four different operating modes, as shown in the following table. The bit-field TxM in register T01_TMOD selects the operating mode to be used for each timer.

In modes 0, 1 and 2, the two timers operate independently, but in mode 3, their functions are specialized.

Table 1713 Timer 0 and timer 1 modes

Mode	Operation
0	13-bit timer/counter The timer is essentially an 8-bit counter with a divide-by-32 prescaler. This mode is included solely for compatibility with Intel 8048 devices
1	16-bit timer/counter The timer registers, T01_TLx and T01_THx, are concatenated to form a 16-bit timer/ counter

(table continues...)

Table 1713 (continued) Timer 0 and timer 1 modes

Mode	Operation
2	8-bit timer/counter with auto-reload The timer register T01_TLx is reloaded with a user-defined 8-bit value in T01_THx upon overflow
3	Timer 0 operates as two 8-bit timers/counters The timer registers, T01_TL0 and T01_TH0, operate as two separate 8-bit counters. Timer 1 is halted and retains its count even if enabled

39.3.10.4.1 Mode 0

Putting either Timer 0 or Timer 1 into mode 0 configures it as an 8-bit timer/counter with a divide by 32 prescaler. The following figure shows the mode 0 operation.

In this mode, the timer register is configured as a 13-bit register. As the count rolls over from all '1's to all '0's, it sets the timer overflow flag TFx (see exception below). The overflow flag TFx can then be used to request an interrupt (see related information). The count input is enabled for the timer when TRx=1 and either GATEx=0 or EXINTx=1 (setting GATEx=1 allows the timer to be controlled by external input EXINTx to facilitate pulse width measurements). TRx is a control bit in the register TCON; bit GATEx is in register T01_TMOD.

The 13-bit register consists of all the 8 bits of T01_THx and the lower 5 bits of T01_TLx. The upper 3 bits of T01_TLx are indeterminate and should be ignored. Setting the run flag (TRx) does not clear the registers.

Mode 0 operation is the same for Timer 0 and Timer 1 (exception see below)

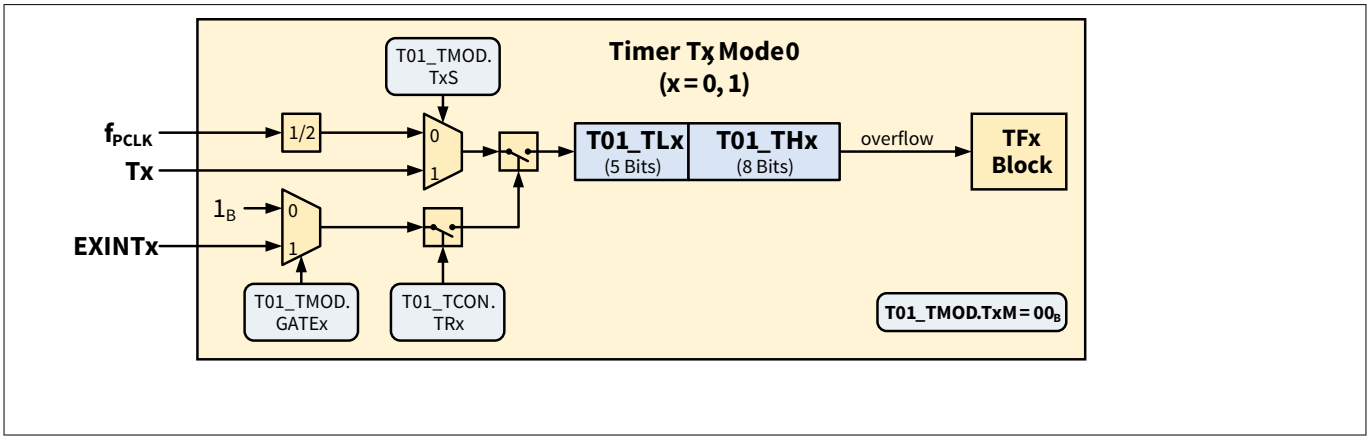


Figure 877 Timer x, mode 0: 13-bit timer/counter

The configuration shown in the figure above applies to Timer 1 as long as Timer 0 is NOT operating in mode 3. When Timer 0 is operating in mode 3, it uses resources from Timer 1; therefore, the mode 0 configuration of Timer 1 in this case is shown in the following figure.

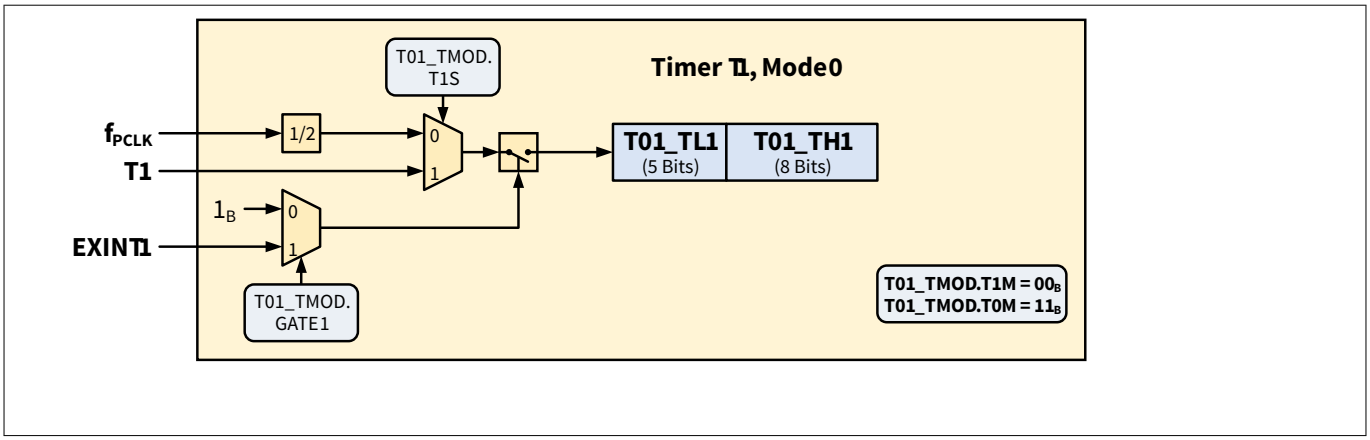


Figure 878 Timer 1 in mode 0 while timer 0 is in mode 3: 13-bit timer/counter

Related information

T0/T1 interrupt flags and requests on page 6581

39.3.10.4.2 Mode 1

Mode 1 operation is similar to that of mode 0, except that the timer register runs with all 16 bits. Mode 1 operation for Timer x is shown in the following figure.

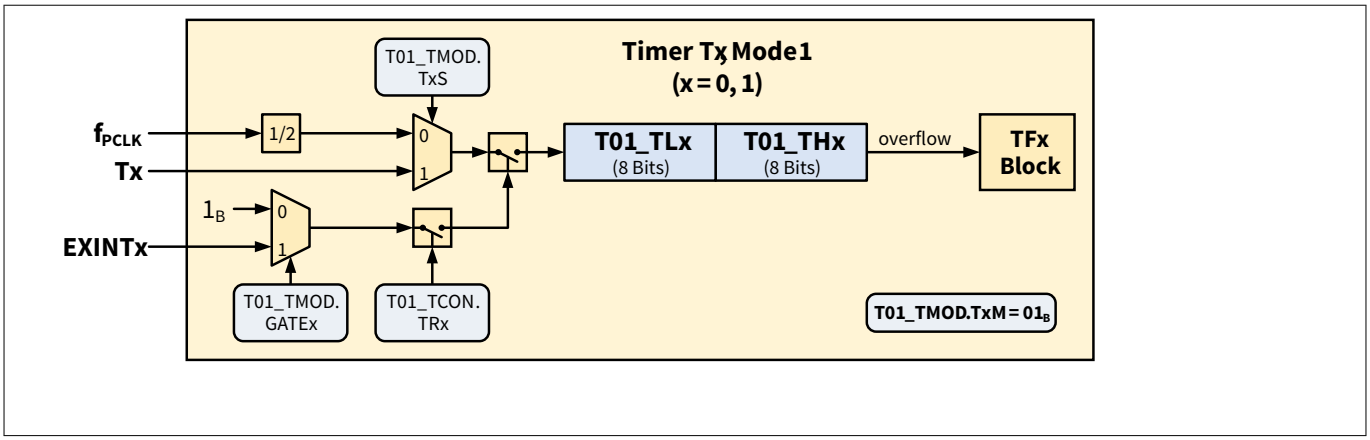


Figure 879 Timer x, mode 1: 16-bit timer/counter

The configuration shown in the figure above applies to Timer 1 as long as Timer 0 is NOT operating in mode 3. When Timer 0 is operating in mode 3, it uses resources from Timer 1; therefore, the mode 1 configuration of Timer 1 in this case is shown in the following figure.

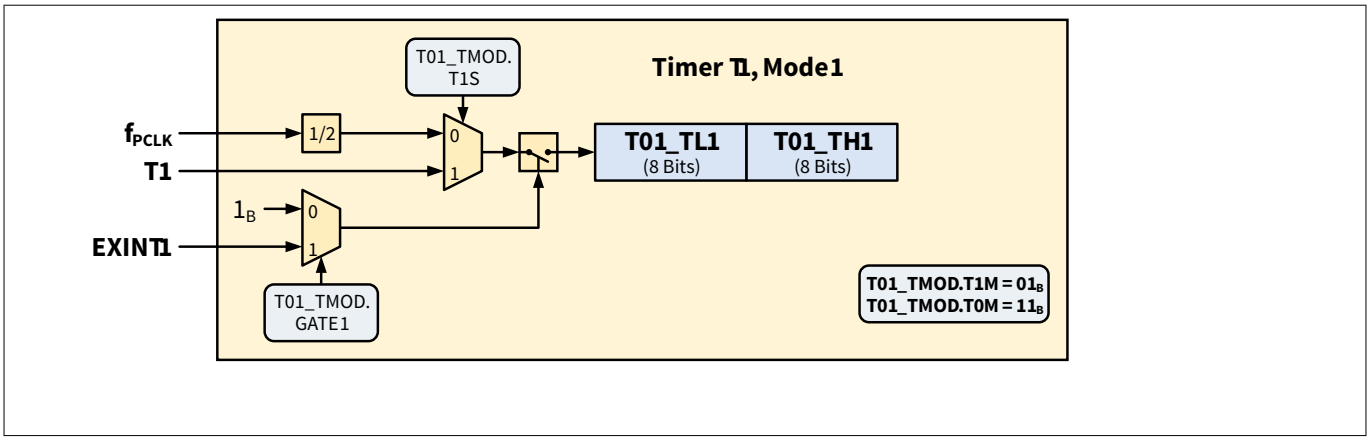


Figure 880 Timer 1 in mode 1 while timer 0 is in mode 3: 16-bit timer/counter

39.3.10.4.3 Mode 2

In Mode 2 operation, the timer is configured as an 8-bit counter (TLx) with automatic reload, as shown in the following figure.

An overflow from TLx not only sets TFx, but also reloads TLx with the contents of THx that has been preset by software. The reload leaves THx unchanged.

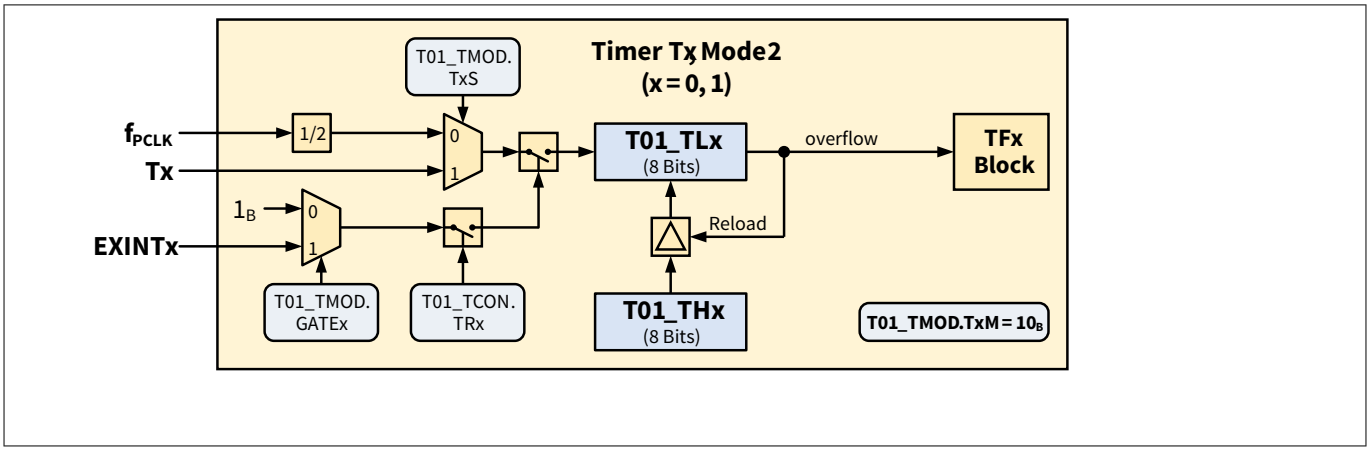


Figure 881 Timer x, mode 2: 8-bit timer/counter with auto-reload

The configuration shown in the figure above applies to Timer 1 as long as Timer 0 is NOT operating in mode 3. When Timer 0 is operating in mode 3, it uses resources from Timer 1; thus, the mode 2 configuration of Timer 1 in this case is shown in the following figure.

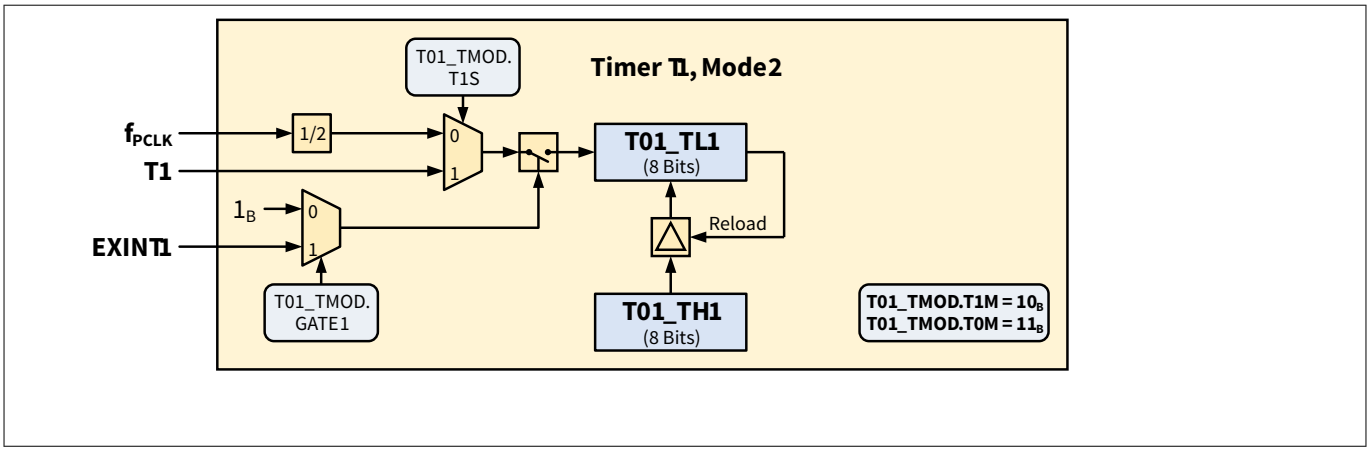


Figure 882 Timer 1 in mode 2 while timer 0 is in mode 3: 8-bit timer/counter

39.3.10.4.4 Mode 3

In mode 3, Timer 0 and Timer 1 behave differently. Timer 0 in mode 3 establishes T01_TL0 and T01_TH0 as two separate counters. Timer 1 in mode 3 simply holds its count. The effect is the same as setting TR1=0.

The logic for mode 3 operation for Timer 0 is shown in the following figure. T01_TL0 uses the Timer0 control bits GATE0, TR0 and TF0, while T01_TH0 is locked into a timer function (counting $f_{PCLK}/2$) and takes over the use of TR1 and TF1 from Timer 1 . T01_TH0 now sets TF1 upon overflow and generates an interrupt if ET1 is set (see also related information).

Mode 3 is provided for applications requiring an extra 8-bit timer. When Timer 0 is in mode 3 and TR1 is set, Timer 1 can be turned on by switching it to any of the other modes and turned off by switching it into mode 3.

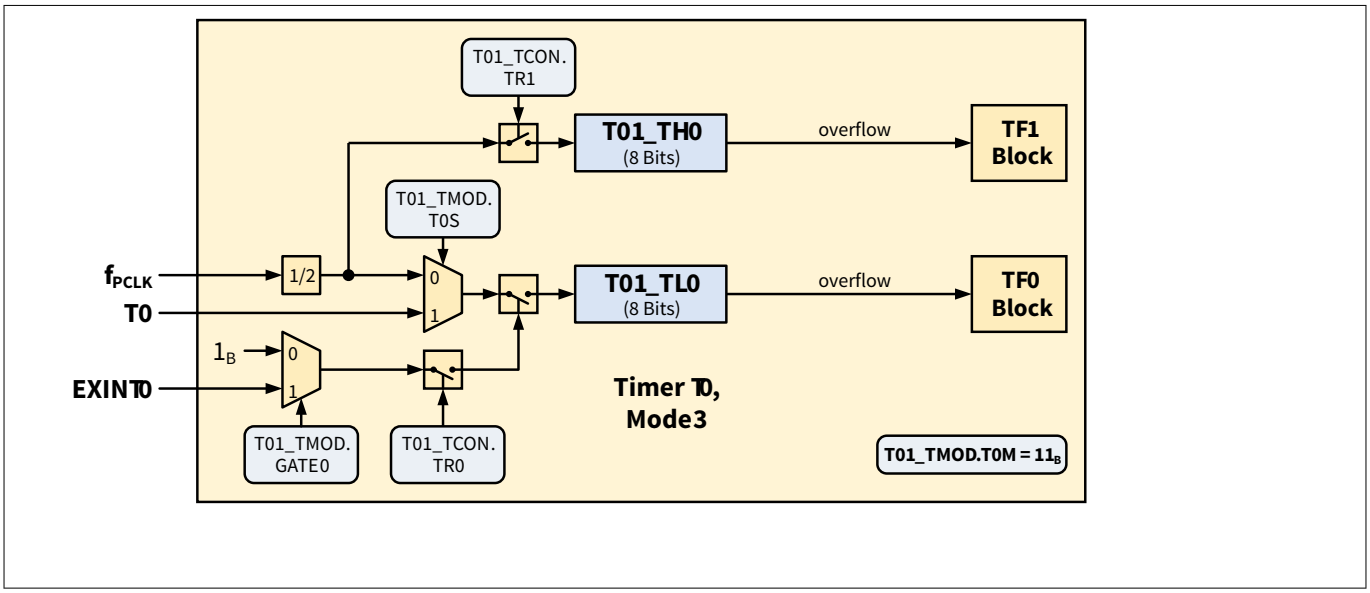


Figure 883 Timer0, mode 3: two 8-bit timers/counters

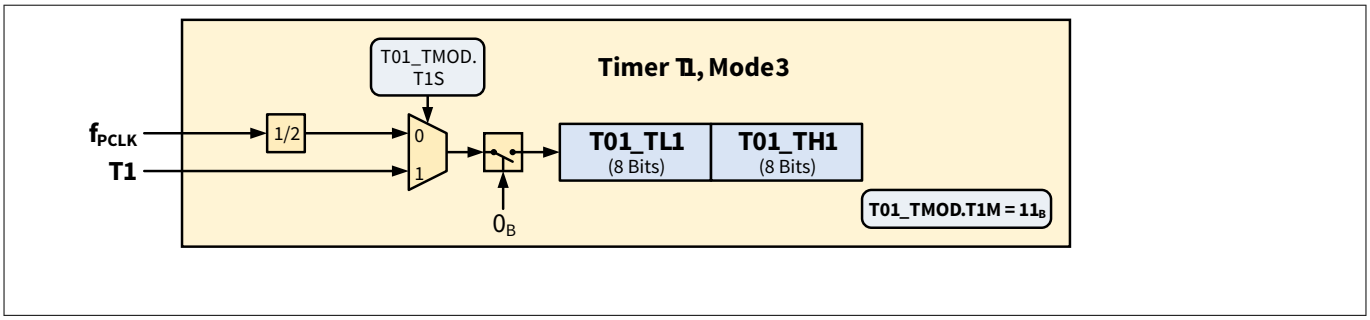


Figure 884 Timer 1 in mode 3: timer stops

Related information

T0/T1 interrupt flags and requests on page 6581

39.3.10.5 T0/T1 interrupt flags and requests

The following figure illustrates the T0/T1 interrupt requests. Besides the overflow interrupts, the T0/T1 module also handles the control for the two external interrupts EXINT0 and EXINT1.

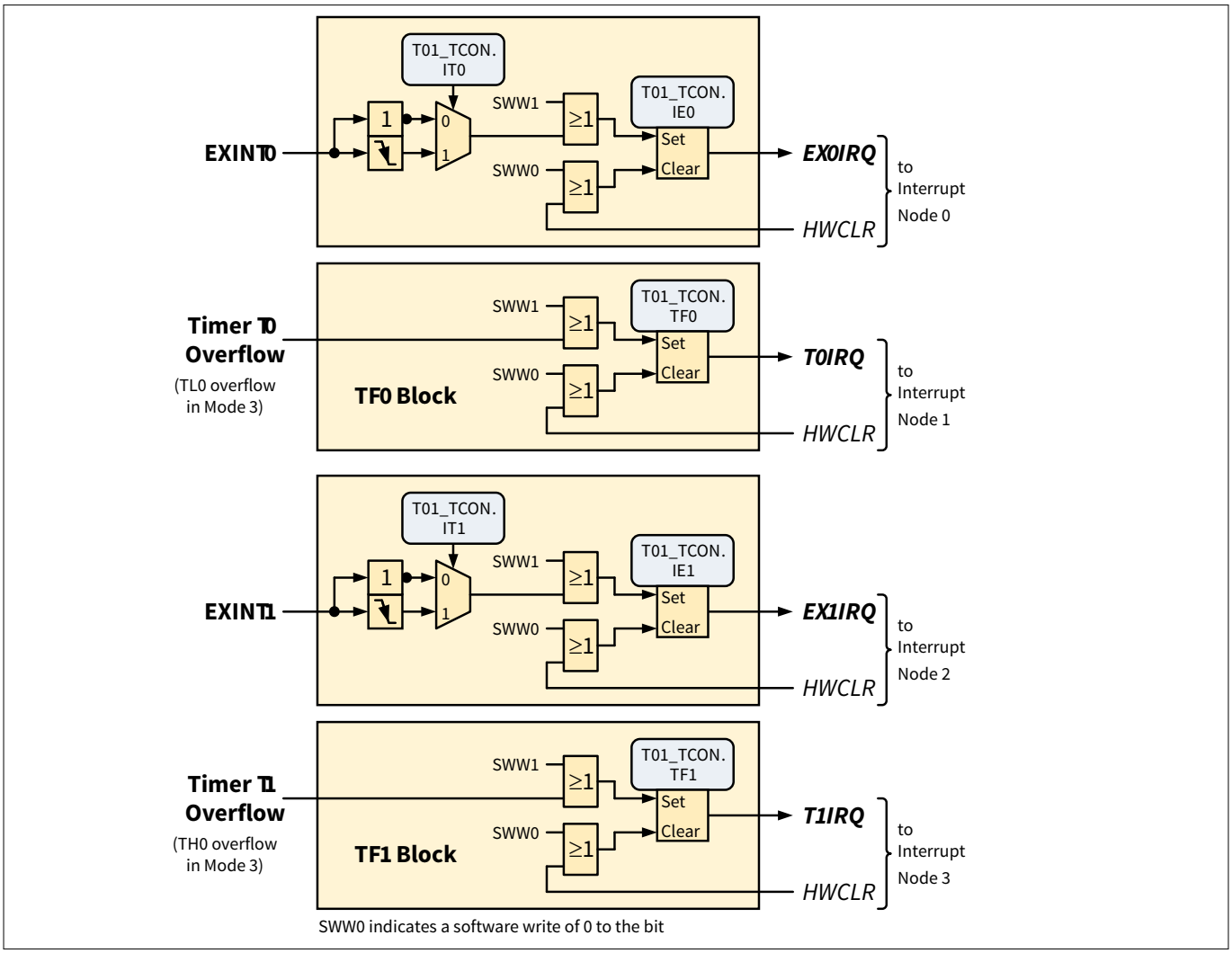


Figure 885 T0/T1 interrupt requests

Note that when timer T0 operates in Mode 3, flag TF0 is set by an overflow of the 8-bit timer part T01_TL0, while flag TF1 is set by an overflow of the 8-bit timer part T01_TH0. Besides being set by hardware, the flags can also be set by software (triggering an interrupt request).

The interrupt flags are automatically cleared by the CPU on entry into the respective service routine. They can also be cleared by software. See also the section on “Interrupt Structure 1” in “Interrupt System” chapter.

Related information

Interrupt structure 1 on page 6521

39.3.10.6 Register overview

Seven SFRs control the operations of Timer 0 and Timer 1. They can be accessed from both the standard (non-mapped) and mapped SFR area.

The following table lists the addresses of these SFRs.

Table 1714 Registers overview - T0/T1 (sorted by name)

Short name	Description	RMAP	PAGE	Offset address
IEN0	Interrupt Enable Register 0	X	X	$D8_H$
T01_TCON	Timer 0/1 Control Register	X	X	CA_H
T01_TMOD	Timer Mode Register	X	X	CB_H
T01_TL0	Timer 0 Low Byte	X	X	CC_H
T01_TL1	Timer 1 Low Byte	X	X	CD_H
T01_TH0	Timer 0 High Byte	X	X	CE_H
T01_TH1	Timer 1 High Byte	X	X	CF_H

The low bytes (T01_TL0,, T01_TL1) and high bytes (T01_TH0, T01_TH1) of both Timer 0 and Timer 1 can be combined to a one-timer configuration depending on the mode used. Register T01_TCON controls the operations of Timer 0 and Timer 1. The operating modes of both timers are selected using register T01_TMOD. Register IEN0 contains bits that enable interrupt operations in Timer 0 and Timer 1.

39.3.11 Timer 2 and capture-compare module (T2CCU0/1)

The T2CCU module is a flexible timing unit with two 16-bit timers and a 6-channel Capture/Compare Unit (CCU). There are two T2CCU modules instantiated in SCR: T2CCU0 and T2CCU1.

***Note**: In following sections if the index 0 or 1 is omitted in terms like T2CCU, T2EX, EXF2 then the statement applies generally to both instantiated T2CCU modules.*

39.3.11.1 Overview

The block diagram of the T2CCU module is shown in the following figure. It consists of the Timer 2, the Capture/Compare Timer (CCT), and a Capture/Compare Unit (CCU). Control is available in the CCU, for each of its 16-bit capture/compare channels, to individually select either Timer 2 or the Capture/Compare Timer (CCT) as the time base. Both timers have a resolution of 16 bits. The clock frequency of T2CCU, f_{T2CCU}, is set at PCLK frequency.

The T2CCU can be used for various digital signal generation and event capturing like pulse generation, pulse width modulation, pulse width measuring etc. Target applications include various automotive control as well as industrial applications (frequency generation, digital-to-analog conversion, process control etc.).

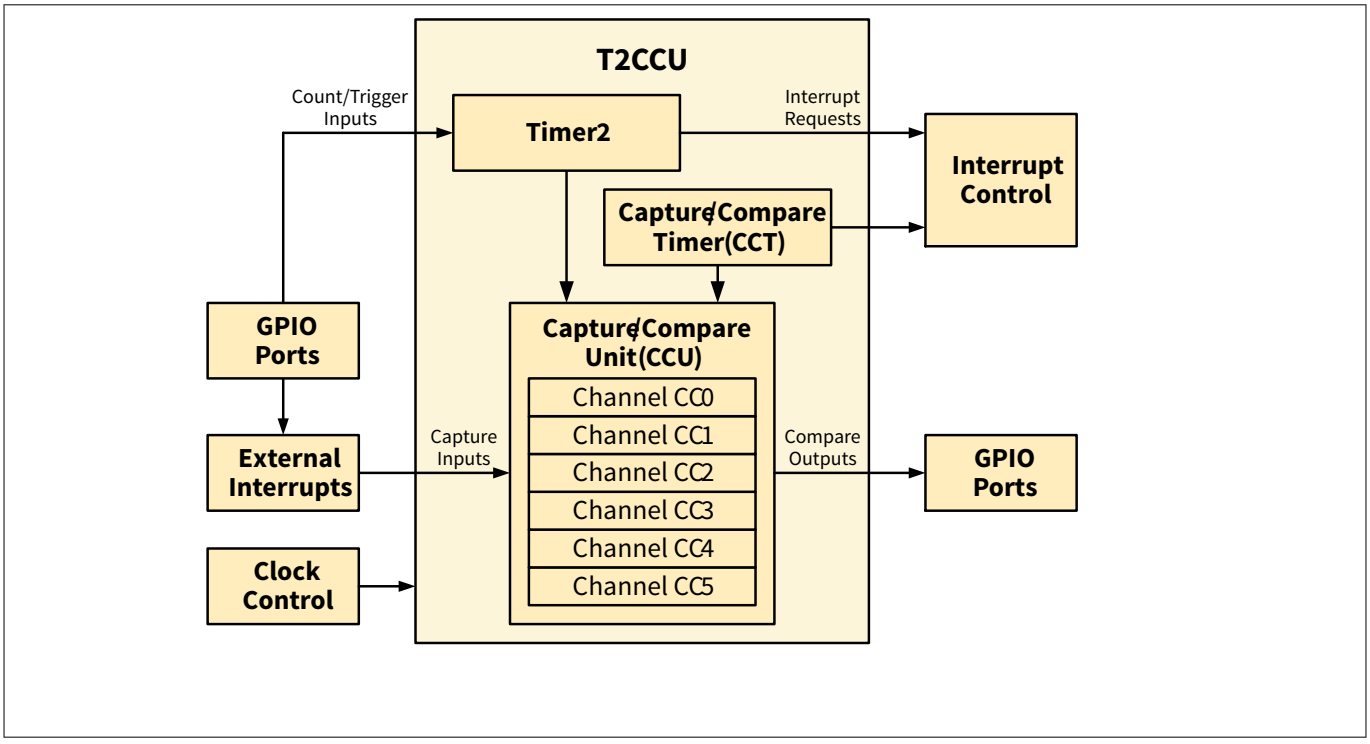


Figure 886 T2CCU module block diagram

39 Stand-by Controller (SCR)

The T2CCU has the following features:

Timer 2

- 16-bit resolution
- Timer or counter operation
- 16-bit auto-reload mode
- Selectable up or down counting
- One channel, 16-bit capture mode
- Baudrate generator for U(S)ART
- Usable as base timer for the CCU

Capture/Compare Timer (CCT)

- 16-bit resolution
- Timer operation
- 16-bit reload mode
- Extremely flexible Capture/Compare Timer count rate by cascading with Timer 2
- Capture/Compare Timer may be ‘reset’ immediately by triggering overflow event

Capture/Compare Unit (CCU)

- 16-bit resolution
- Option to individually select either Timer 2 or the CCT as base timer for a channel
- Six compare channels in total
  - Shadow register for each compare register: Transfer through software control or on timer overflow
  - Compare Mode 0: Compare output signal changes from the inactive level to active level on compare match. Returns to inactive level on timer overflow. Compare Mode 0 can be configured with or without dead-time generation. Active level can be defined by register bit for channel groups A and B. Support of 0% to 100% duty cycle in Compare Mode 0
  - Compare Mode 1: Full control of the software on the compare output signal level, for the next compare match
  - Concurrent Compare Mode with channel 0
- Four capture channels, shared with the compare channels
  - Capture Mode 0: Capture on any external event (rising/falling/both edge) at the 4 pins T2CC0 to T2CC3
  - Capture Mode 1: Capture upon writing to the low byte of the corresponding channel capture register
  - Capture Mode 0 or 1 can be established independently on the 4 capture channels

39.3.11.2 Timer 2

The Timer 2 of the T2CCU has the following features:

- 16-bit auto-reload mode
- Selectable up or down counting
- One channel, 16-bit capture mode
- Timer 2 can be used as a stand-alone timer, aside of the CCU, or can be used as a base timer for the CCU

39.3.11.2.1 Basic timer 2 operations

The following sections describe the basic functions of Timer 2.

Timer 2 start/stop and count control

The count input to Timer 2 is controlled by the run bit T2CCU1_T2_CON.TR2. timer 2 can be started by setting run bit T2CCU1_T2_CON.TR2 either through hardware or software. The timer can only be stopped through software by clearing bit T2CCU1_T2_CON.TR2.

The run bit T2CCU1_T2_CON.TR2 can be set by various events:

- By software: Write ’1’ to bit T2_CON.TR2
- By hardware:
  - A selectable transition at input T2EX. This input is enabled/disabled through bit T2_MOD.T2RHEN. The active transition of the input, either a rising or a falling edge, is selected through bit T2_MOD.REGS. Note that input T2EX can also be used for the capture/reload control of Timer 2 (see following sections)
  - Through the synchronous start signal from the capture/compare timer CCT. This event is enabled/ disabled through bit TIMSYN in register T2CCU_CCTCON. This option can be used to trigger a synchronous start of both timers, timer 2 and timer CCT. See Synchronized Start of Timer 2 and CCT Timer in related information for details

All of these actions set the run bit TR2 to 1. Its output controls the count input to the Timer 2. The state of the run bit can be read by software. To stop the timer, clear the run bit TR2 by writing a ‘0’ to it.

Following figure illustrates the start/stop and count control of timer 2. The count input selection is described in the next section. In the further diagrams of Timer 2, the start/stop control is shown as a block, together with the count input control. For the details, please refer to the following figure.

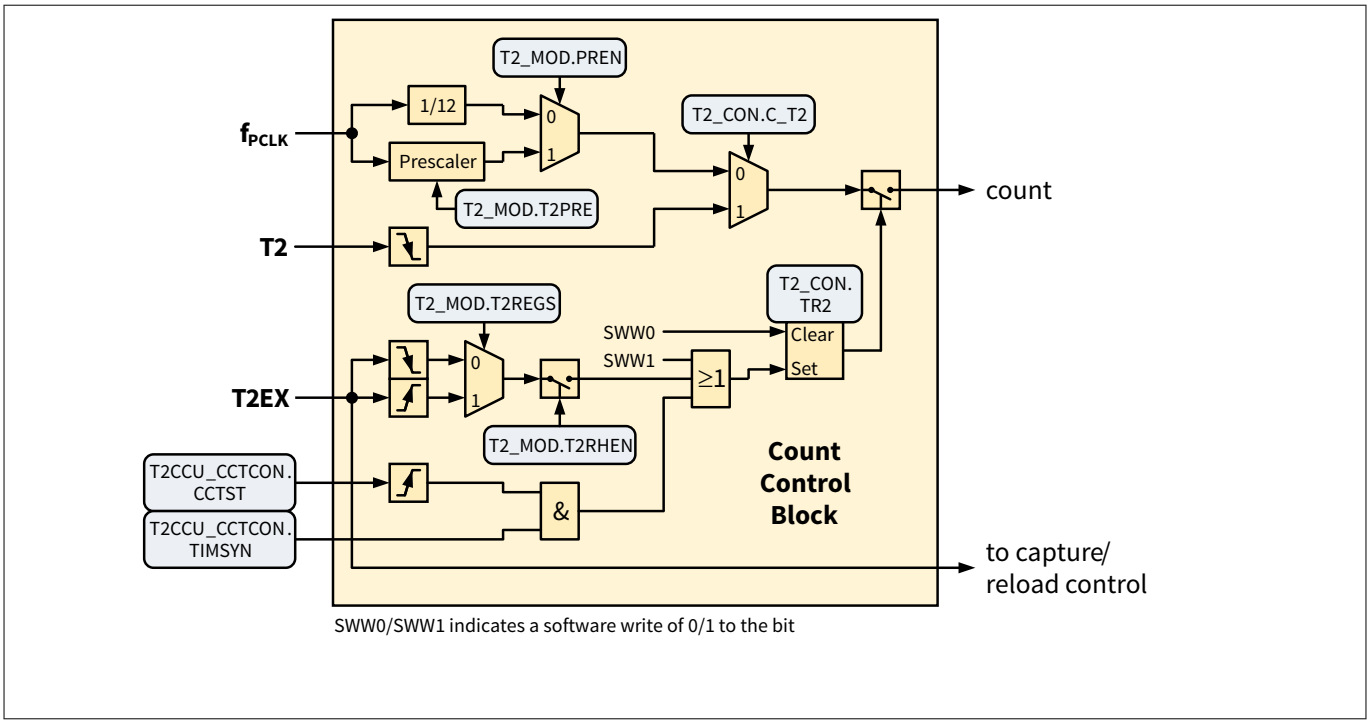


Figure 887 Timer 2 start/stop and count control

Related information

Synchronized start of timer 2 and CCT timer on page 6595

Count clock options

Timer 2 can either operate as a timer or as a counter. The selection is made through bit T2_CON.C_T2.

- Timer mode (C_T2=0):

  Either $f_{PCLK}/12$ or f_{PCLK} divided by a programmable pre-scaler can be selected as input clock to Timer 2. The selection is made through bit T2_MOD.PREN. The divide factor for the programmable pres-caler is chosen through bit-field T2_MOD.T2PRE
- Counter mode ((C_T2=1):

  Timer 2 counts in response to a 1-to-0 transition at input T2. Input T2 is sampled over 2 PCLK clock cycles. If a 1 was detected during the first clock and a 0 was detected in the following clock, then a count clock for Timer 2 is generated. Therefore, the input levels should be stable for at least 1 PCLK clock cycle

39.3.11.2.2 Timer 2 operating modes

The following sections describe the various operating modes of Timer 2.

***Note**: In order to simplify the diagrams, the connection of the Timer 2 contents to the Capture/Compare Unit (CCU) is not shown in the following figures.*

Reload Mode of timer 2

In reload mode of Timer 2 (bit T2_CON.CP_RL2 is 0), the RC2 register (register pair T2_RC2H|T2_T2_RC2L) is used to hold a reload/trigger value for Timer 2:

- When down counting is disabled, the value in RC2 is loaded into Timer 2 either on overflow or in response to an external signal transition
- When down counting is enabled, the value in RC2 is used as a trigger value to load $FFFF_H$ into Timer 2

In the following sections, these operations are described in detail.

***Note**: For a description of the interrupt request and flag logic "TF2 Block" and "EXF2 Block", please see related information.*

Related information

Timer 2 interrupt requests on page 6590

Reload operation in up-counting mode

When bit T2_MOD.DCEN is 0, Timer 2 can only count up, and the RC2 register is used to hold a reload value. The following figure shows an operational block diagram of Timer 2 in this configuration.

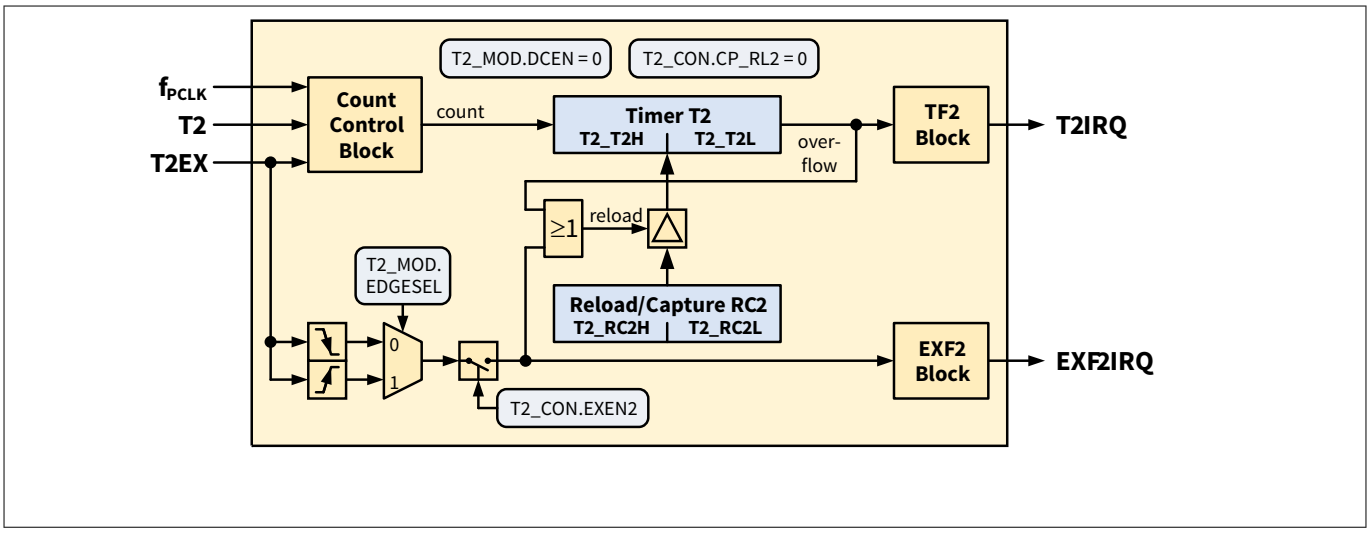


Figure 888 Reload operation in up-counting mode (T2_MOD.DCEN=0)

Depending on bit T2_CON.EXEN2, two options exist:

- EXEN2=0: Reload on overflow only (auto-reload):
  - Once the timer is started, it counts up to its maximum value $FFFF_H$
  - The next count clock pulse leads to an overflow condition: Timer 2 is loaded with the contents of register RC2, and flag T2_CON.TF2 is set to 1 (can be used to generate an interrupt request)
  - With the next count clock pulse(s), the timer continues to count up from the reloaded value
- EXEN2=1: Reload on overflow or on external signal edge:
  - Once the timer is started, it counts up
  - A reload of Timer 2 is performed either on an overflow condition (auto-reload, see above) or by a selected signal edge on input T2EX (external reload). With the next count clock pulse(s), the timer continues to count up from the reloaded value
  - The signal edge can either be a positive (0-to-1), or a negative (1-to-0) transition at input T2EX. The selection is done through bit T2_MOD.EDGESEL
  - The selected signal transition also sets flag T2_CON.EXF2 to 1 (can be used to generate an interrupt request)

Note that the input signal T2EX can be used for both, the timer count control and the external reload function.

When bit T2RHEN is set, Timer 2 is started by first falling edge/rising edge at pin T2EX, which is defined by bit T2REGS. If bit T2_CON.EXEN2 is set, bit T2_CON.EXF2 is also set at the same point when Timer 2 is started with the same falling edge/rising edge at pin T2EX, which is defined by bit T2_MOD.EDGESEL. The reload will happen with the following negative/positive transitions at pin T2EX, which is defined by bit T2_MOD.EDGESEL.

***Note**: In counter mode, if the reload via T2EX and the count clock T2 are detected simultaneously, the reload takes precedence over the count. The counter increments its value with the following T2 count clock.*

Reload operation in up/down-counting mode

When bit T2_MOD.DCEN is set to 1, the up/down count selection is enabled for Timer 2. The count direction is determined by the level at input pin T2EX. The following figure shows an operational block diagram of Timer 2 in this configuration.

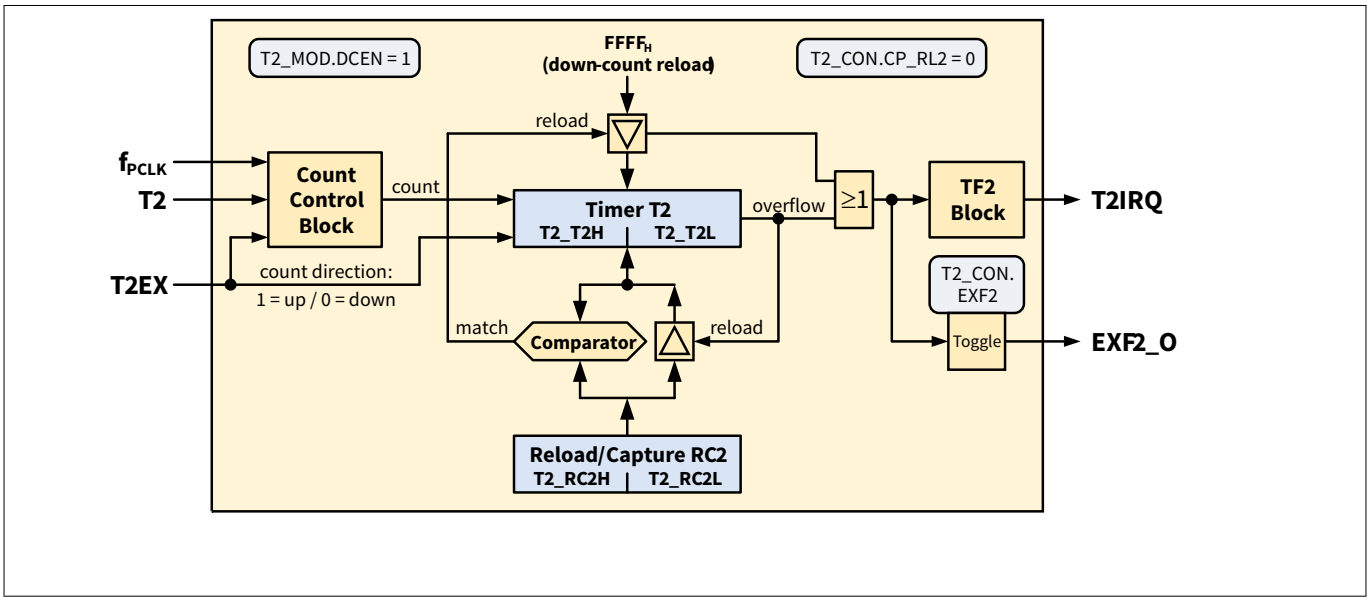


Figure 889 Reload operation in up/down-counting mode (T2_MOD.DCEN=1)

The operation in this mode is as follows:

- A logic 1 at pin T2EX sets the Timer 2 to up-counting mode:
  - Once the timer is started, it counts up to its maximum value FFFF$_H$
  - The next count clock pulse leads to an overflow condition: Timer 2 is loaded with the contents of register RC2, and flag T2_CON.TF2 is set to 1 (can be used to generate an interrupt request)
  - With the next count clock pulse(s), the timer continues to count up from the reloaded value
- A logic 0 at pin T2EX sets the Timer 2 to down-counting mode:
  - Once the timer is started, it counts down
  - When the timer contents match the value stored in register RC2, an underflow condition is generated. Timer 2 is loaded with the value FFFF$_H$, and flag T2_CON.TF2 is set to 1 (can be used to generate an interrupt request)
  - With the next count clock pulse(s), the timer continues to count down from FFFF$_H$

If bit T2RHEN is set, Timer 2 can be started either by rising edge (T2REGS=1) at pin T2EX and then proceed with the up counting, or be started by falling edge (T2REGS=0) at pin T2EX and then proceed with the down counting.

In this mode, bit T2_CON.EXF2 toggles whenever an overflow or an underflow condition is detected. This flag, however, does not generate an interrupt request.

Capture mode of timer 2

In capture mode of Timer 2 (bit T2_CON.CP_RL2 is 1), the RC2 register (register pair T2_RC2H|T2_RC2L) is used to latch the current contents of the timer in response to an external event.

The following figure shows an operational block diagram of Timer 2 in this configuration.

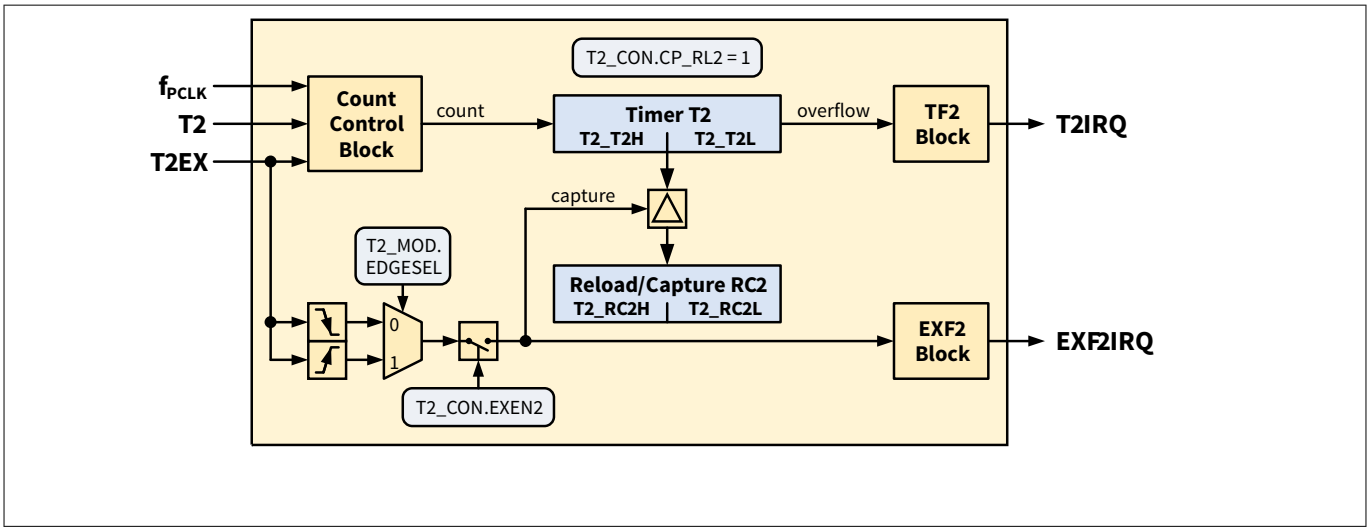


Figure 890 Capture mode of timer 2

To operate the timer in capture mode, the following settings must be made:

- Enable capture mode: Set bit T2_CON.CP_RL2 to 1
- Disable the down-count function: Clear bit T2_MOD.DCEN to 0
- Enable the external input T2EX: Set bit T2_CON.EXEN2 to 1

The timer functions as a 16-bit timer and always counts up to $FFFF_H$. With the next clock pulse, an overflow condition occurs; bit TF2 is set (which can generate an interrupt request), and the timer registers are loaded with 0000_H. The counting continues with the next clock pulse(s).

With a falling or rising edge (chosen by T2_MOD.EDGESEL) at input T2EX, the contents of the timer register are captured into the RC2 register. At the same time, bit T2_CON.EXF2 is set to 1. The external input is sampled in every PCLK clock cycle. When a sample shows a low (high) level in one PCLK clock cycle, and a high (low) in the next PCLK clock cycle, a transition is recognized. If the capture signal is detected while the timer is being incremented, the timer is first incremented before the capture operation is performed. This ensures that always the latest value of the timer register is captured.

Note that the input signal T2EX can be used for both, the timer count control and the external capture function.

If bit T2RHEN is set, Timer 2 is started by first falling edge/rising edge at pin T2EX, which is defined by bit T2REGS. If bit T2_CON.EXEN2 is set, bit T2_CON.EXF2 is also set at the same point when Timer 2 is started with the same falling edge/rising edge at pin T2EX, which is defined by bit T2_MOD.EDGESEL. The capture will happen with the following negative/positive transitions at pin T2EX, which is defined by bit T2_MOD.EDGESEL.

39.3.11.2.3 Timer 2 interrupt requests

Timer 2 can generate two different interrupt requests to the CPU:

- An interrupt request on timer overflow
- An interrupt request on an external transition at input T2EX

These interrupt requests are detailed in the following sections.

Timer overflow interrupt request

The following figure shows a functional diagram of the overflow interrupt request generation. The flag T2_CON.TF2 is set on an overflow of the timer (in certain modes), or it can be set by software. When enabled through bit T2_CON1.TF2EN, the signal which sets the T2_CON1.TF2EN flag will generate an interrupt request to the CPU.

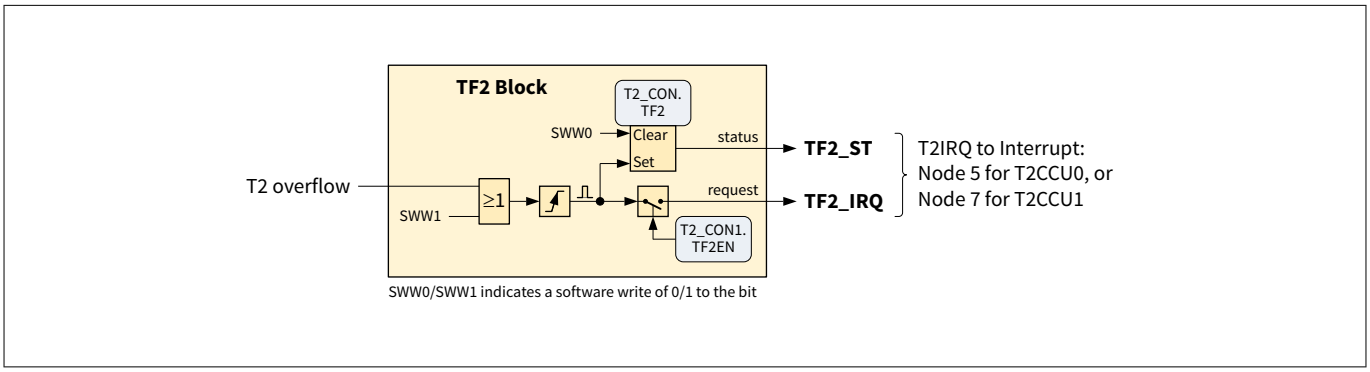


Figure 891 TF2 flag block

The overflow flag and the corresponding overflow interrupt request (if enabled) will be set/generated in the following cases:

- By software: By writing a 1 to flag T2_CON.TF2, the flag will be set and an interrupt request (if enabled) will be generated
- Reload Operation in Up-Counting Mode (T2_MOD.DCEN = 0, T2_CON.CP_RL2 = 0): When an overflow of Timer 2 occurs
- Reload Operation in Up/Down-Counting Mode (T2_MOD.DCEN = 1, T2_CON.CP_RL2 = 0): When an overflow of Timer 2 occurs, or when an underflow (timer contents match the value stored in register RC2) occurs
- Capture Mode of Timer 2 (T2_MOD.DCEN = 0, T2_CON.CP_RL2 = 1): When an overflow of Timer 2 occurs

Flag TF2 must be cleared by software.

***Note**: The overflow interrupt request logic of Timer 2 is basically of type “interrupt structure 2” (see section “Interrupt Structure” in chapter “Interrupt System”), however, the status signal does not have an individual enable control. Therefore, in order to clear the pending request flag in the interrupt system, it is necessary to clear all status flags assigned to that interrupt node, even if the respective interrupt request is not enabled.*

***Note**: Although two signals, TF2_IRQ and TF2_ST are connected to the respective interrupt node, the general signal name T2IRQ is used to represent both signals (interrupt structure 2).*

Related information

Interrupt structure on page 6521

Timer external interrupt request

The following figure shows a functional diagram of the external interrupt request generation. The flag T2_CON.EXF2 is set on an external transition at input T2EX (in certain modes), or it can be set by software. When enabled through bit T2_CON1.EXF2EN, the signal which sets the T2_CON.EXF2 flag will generate an interrupt request to the CPU.

The output of flag T2_CON.EXF2 is also connected to certain GPIO port lines.

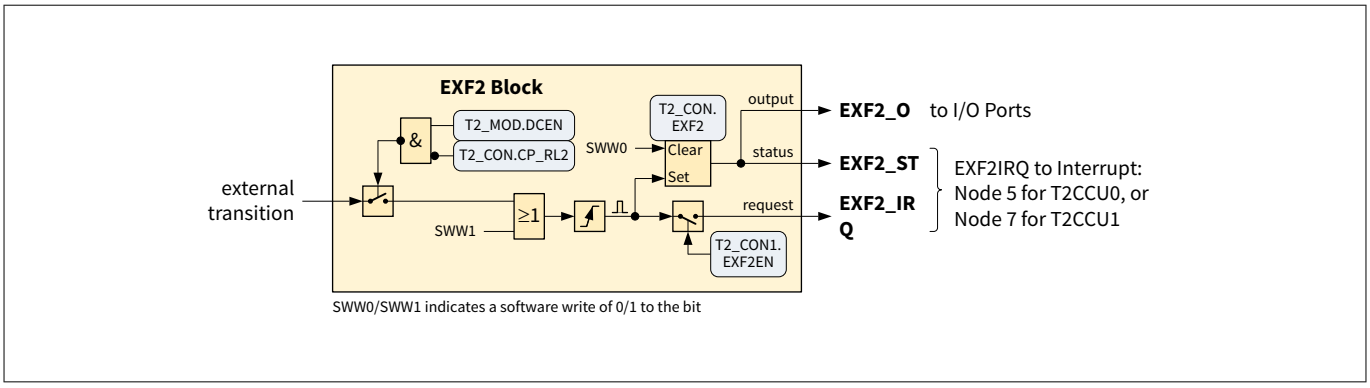


Figure 892 EXF2 flag block

The external transition flag T2_CON.EXF2 and the corresponding external interrupt request (if enabled) will be set/generated in the following cases:

- By software: By writing a 1 to flag T2_CON.EXF2, the flag will be set and an interrupt request (if enabled) will be generated
- Reload Operation in Up-Counting Mode (T2_MOD.DCEN = 0, T2_CON.CP_RL2 = 0): When bit T2_CON.EXEN2 is set and the selected transition at input T2EX occurs
- Reload Operation in Up/Down-Counting Mode (T2_MOD.DCEN = 1, T2_CON.CP_RL2 = 0): In this mode, flag T2_CON.EXF2 will toggle on each overflow or underflow (timer contents match the value stored in register RC2) of Timer 2. No interrupt request will be generated (this is not shown in the figure above)
- Capture Mode of Timer 2 (T2_MOD.DCEN = 0, T2_CON.CP_RL2 = 1): When bit T2_CON.EXEN2 is set and the selected transition at input T2EX occurs

Flag EXF2 must be cleared by software.

Note that while the timer/counter function is disabled (TR2=0), it is still possible to generate a Timer 2 interrupt to the core through an external event at input T2EX, as long as Timer 2 remains enabled (SCUSCR_PMCON0.T2CCU_DIS=0). To achieve this, bit T2_CON.EXEN2 must be set. As a result, any transition on T2EX will cause either a dummy reload or a dummy capture, depending on the T2_CON.CP_RL2 bit selection.

***Note**: The external interrupt request logic of Timer 2 is basically of type "interrupt structure 2" (see section "Interrupt Structure" in chapter "Interrupt System"), however, the status signal does not have an individual enable control. In order to clear the pending request flag in the interrupt system, it is necessary to clear all status flags assigned to that interrupt node, even if the respective interrupt request is not enabled.*

***Note**: Although two signals, EXF2_IRQ and EXF2_ST are connected to the respective interrupt node, the general signal name EXF2IRQ is used to represent both signals (interrupt structure 2).*

Related information

Interrupt structure on page 6521

39.3.11.2.4 Input selection for timer 2

Timer 2 can be used as an event counter which counts 1-to-0 transitions at the external input T2. This input is selected from four different sources, T2A, T2B, T2C and T2D. The selection is performed by the SFR bits T2_CON1.INSEL. In addition, there are sixteen sources for the input T2EX. They can be selected by respective T2CCUx_T2_CON1.EXTINSEL. In case of T2CCU0 12 are coming from SCR.P0-3 ports. For T2CCU1 13 are coming from SCR.P0-3 ports In addition:

- T20EX has four more sources:
  - PINA, ESR1, PMS RTC CMP0, PMS RTC CMP1
- T21EX has three more sources:
  - PMS RTC clock, 70 kHz clock, external clock from pin SCR_P1.2(P33.10)

Detailed pin assignment for both T2CCU modules is shown in the GPIO chapter of the SCR.

Related information

General purpose I/O Ports (PORT8) on page 6543

39.3.11.3 Capture/compare timer of T2CCU

The Capture/Compare Timer (CCT) is part of the T2CCU, and is a dedicated time base for the CCU capture/compare operations. The following figure shows a block diagram of the CCT timer.

***Note**: In order to simplify the diagrams, the connection of the CCT contents to the Capture/Compare Unit (CCU) is not shown in the following figures.*

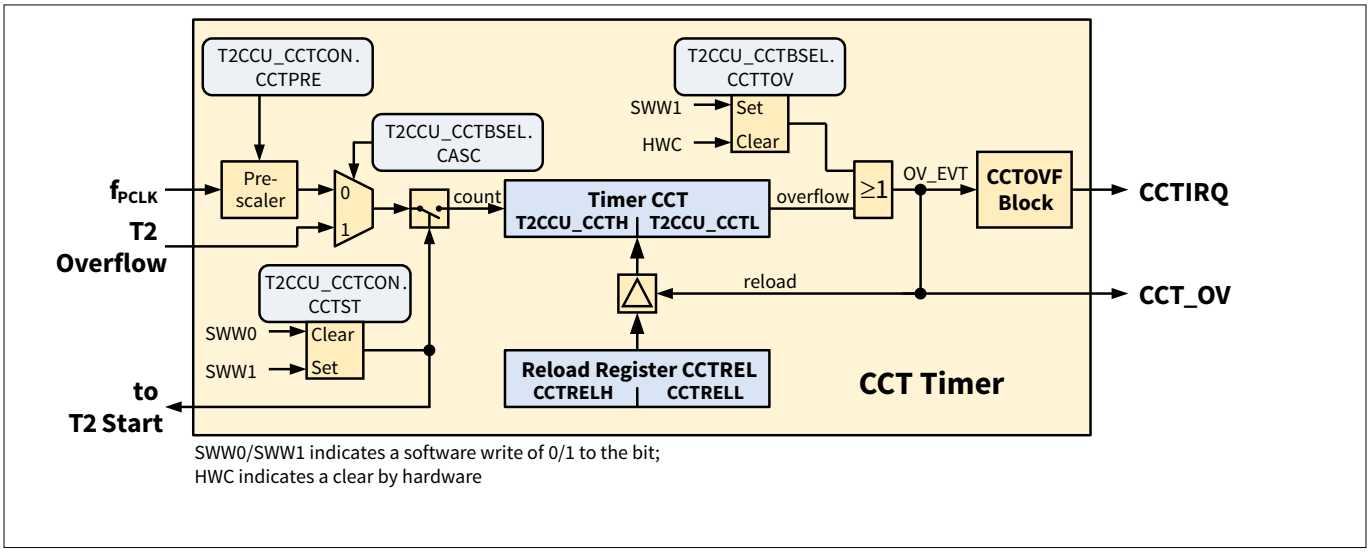


Figure 893 CCT timer block diagram

39.3.11.3.1 CCT timer basic operation

The features of the CCT are:

- 16-bit timer, comprised of register pair T2CCU_CCTH|T2CCU_CCTL (in the following simply called CCT)
- 16-bit reload register, comprised of register pair T2CCU_CCTRELH|T2CCU_CCTRELL (in the following simply called CCTREL)
- Start or stop controlled by software
- Input clock pre-scaler with 12 selectable stages (*input clock range from* f_{PCLK} to $f_{PCLK}/2048$)
- Once started, the CCT counts up from the initial value stored in its count register to $FFFF_H$. With the next clock pulse, it overflows and gets loaded with the value stored in register CCTREL. The counting continues from this value with the following clock pulse(s)
- CCT can be concatenated with Timer 2
- Optional synchronous start of CCT and Timer 2 possible
- CCT overflow event is generated either on CCT timer overflow or triggered by software. CCT overflow event performs the following:
  - Reloads CCT with contents of register CCTREL
  - Sets flag CCTOVF to 1 (can generate an interrupt request)
  - Pulses output line CCT_OV

When writing to the CCT timer register pair T2CCU_CCTH|T2CCU_CCTL, the following rules have to be obeyed:

- When the CCT is not running (CCTCON.CCTST=0), a write to the byte registers T2CCU_CCTH/T2CCU_CCTL updates the corresponding byte of the CCT timer
- When the CCT is running (CCTCON.CCTST=1), register T2CCU_CCTH has to be written first (this value is held in a shadow register). With the following write to register T2CCU_CCTL, both actual timer registers are updated, and the timer continues counting up from the written value

The CCT may be stopped at any time by clearing bit CCTST to 0. The timer registers will hold the last count value.

39.3.11.3.2 Software-triggered CCT timer overflow

A special feature of the CCT timer is the possibility to ‘reset’ the timer immediately by a Software-Triggered Timer Overflow event. By ‘reset’, the CCT timer is reloaded and starts counting from the value from the reload registers, while all actions associated with the operation mode of the timer channels on base timer overflow will be executed. When setting bit T2CCU_CCTBSEL.CCTTOV by software, the CCT timer overflow event is triggered asynchronously to the timer count operation. Bit T2CCU_CCTBSEL.CCTTOV is held at '1' for one PCLK cycle, then it is automatically cleared by hardware.

An example usage for a channel operating in compare mode 0 is illustrated in the following figure. For more details on compare mode 0, refer to Compare Operation in related information.

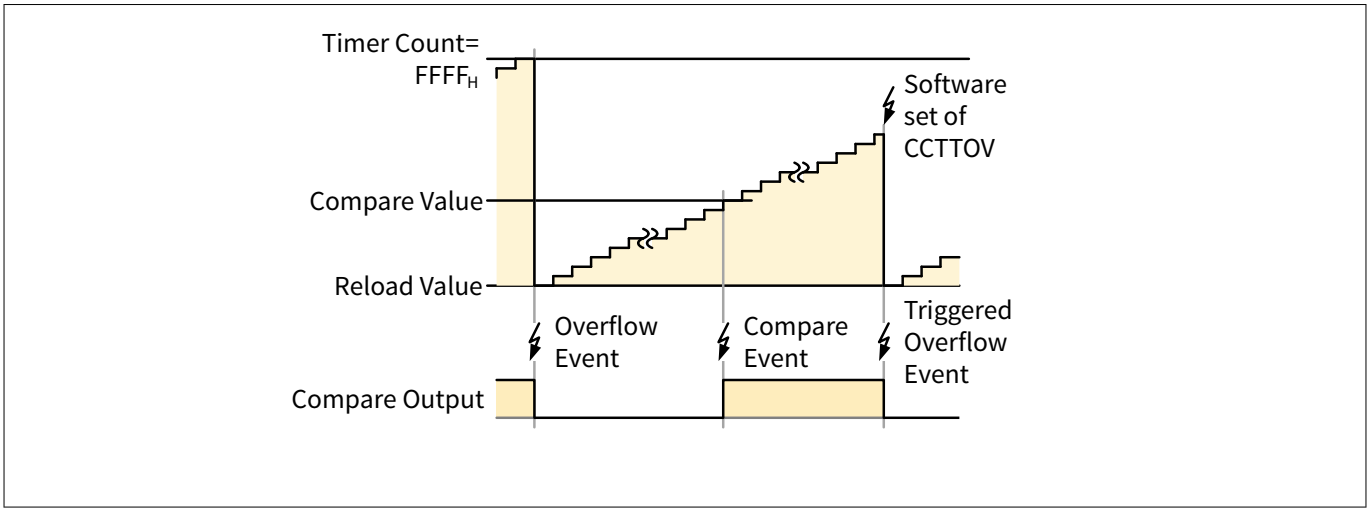


Figure 894 Example of channel in compare mode 0 on ‘triggered timer overflow’ event

Related information

Compare operation on page 6598

39.3.11.3.3 Synchronized start of timer 2 and CCT timer

As described in the previous sections, the two timers of the T2CCU are usually started or stopped individually: timer 2 through bit TR2, and the CCT through bit CCTCON.CCTST.

However, it is also possible to start both timers synchronously. The procedure required to perform this is listed below:

1. Ensure CCT timer is not already running
2. Clear bit TR2 (stop timer 2) and ensure timers are properly initialized
3. Set bit T2CCU0 capture/compare timer control register.TIMSYN to enable synchronized timer starts
4. Set bit CCTCON.CCTST; this will also set bit TR2

Steps 3 and 4 can be combined in a single write operation to SFR T2CCU_CCTCON.

To stop the timers, the respective timer bits need to be cleared independently.

39.3.11.3.4 Cascading timer 2 and CCT timer for flexible count rate

Especially for applications requiring the base timer to run at a flexible count rate, it is possible to cascade Timer 2 with the CCT timer. When bit T2CCU_CCTBSEL.CASC is set, the CCT timer will increment its count with every

overflow of Timer 2. In this mode, Timer 2 acts as a pre-scaler for timer CCT. See following figure for a block diagram of this option.

***Note**: Timer 2 may be configured in timer mode or as an external event counter through input T2.*

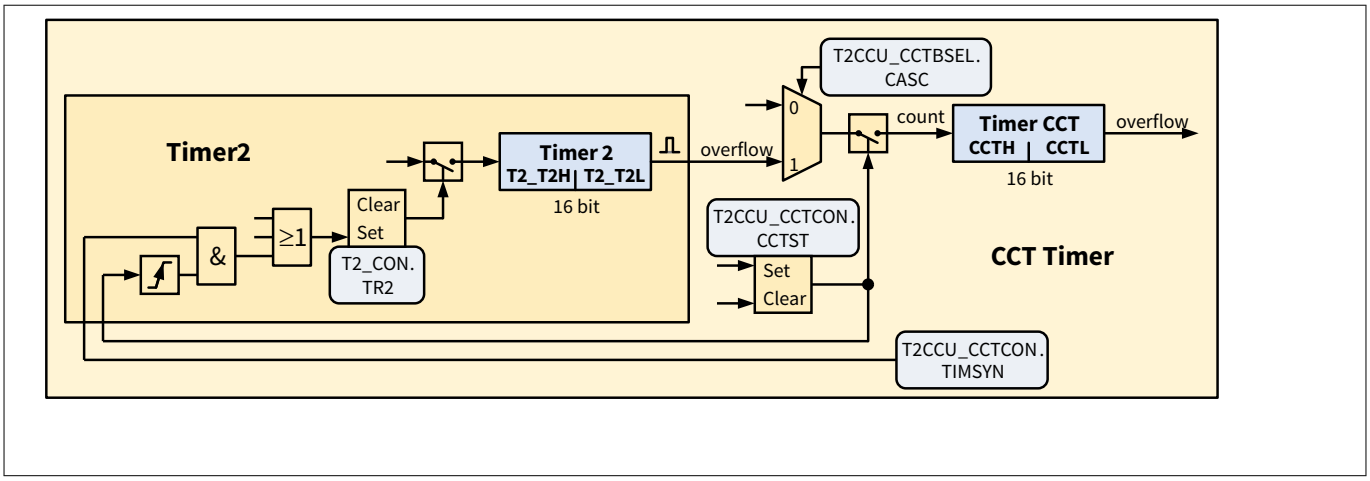


Figure 895 Cascading timer 2 with the CCT timer

39.3.11.3.5 CCT overflow flag and interrupt request

The overflow flag and the corresponding overflow interrupt request (if enabled) will be set/generated in the following cases:

- By software:
  - By writing a 1 to flag T2CCU_CCTCON.CCTOVF
  - By writing a 1 to bit T2CCU_CCTBSEL.CCTTOV (Software-Triggered Timer Overflow, see related information)
- By hardware, when an overflow of the CCT timer occurs

An interrupt request to the CPU is enabled when the enable bit T2CCU_CCTCON.CCTOVEN is set to 1.

Note that flag T2CCU_CCTCON.CCTOVF must be cleared by software.

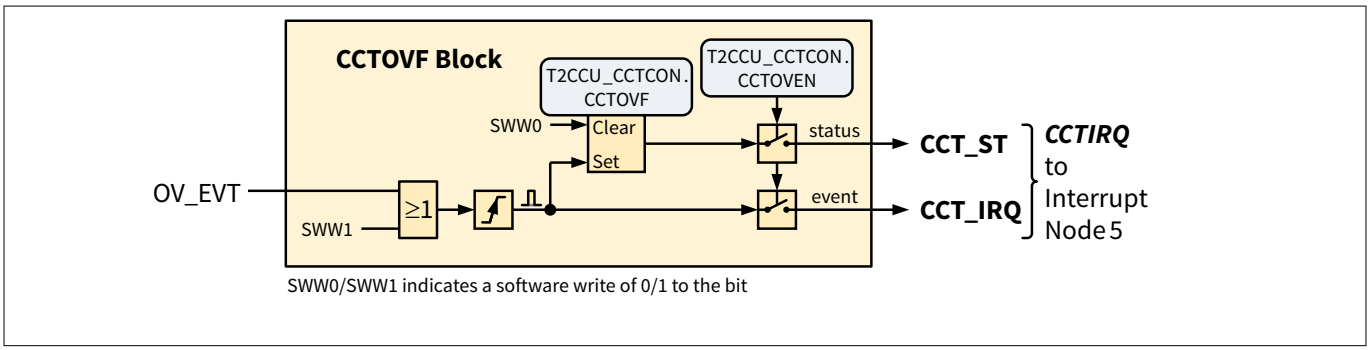


Figure 896 CCT overflow flag block

***Note**: Although two signals, CCT_IRQ and CCT_ST are connected to the respective interrupt node, the general signal name CCTIRQ is used to represent both signals (interrupt structure 2).*

Related information

Software-triggered CCT timer overflow on page 6595

39.3.11.4 Capture/Compare Unit (CCU) of the T2CCU

The Capture/Compare Unit (CCU) of the T2CCU consists of four 16-bit capture/compare channels (CC0-CC3) and two 16-bit compare-only channels (CC4-CC5), illustrated in the following figure.

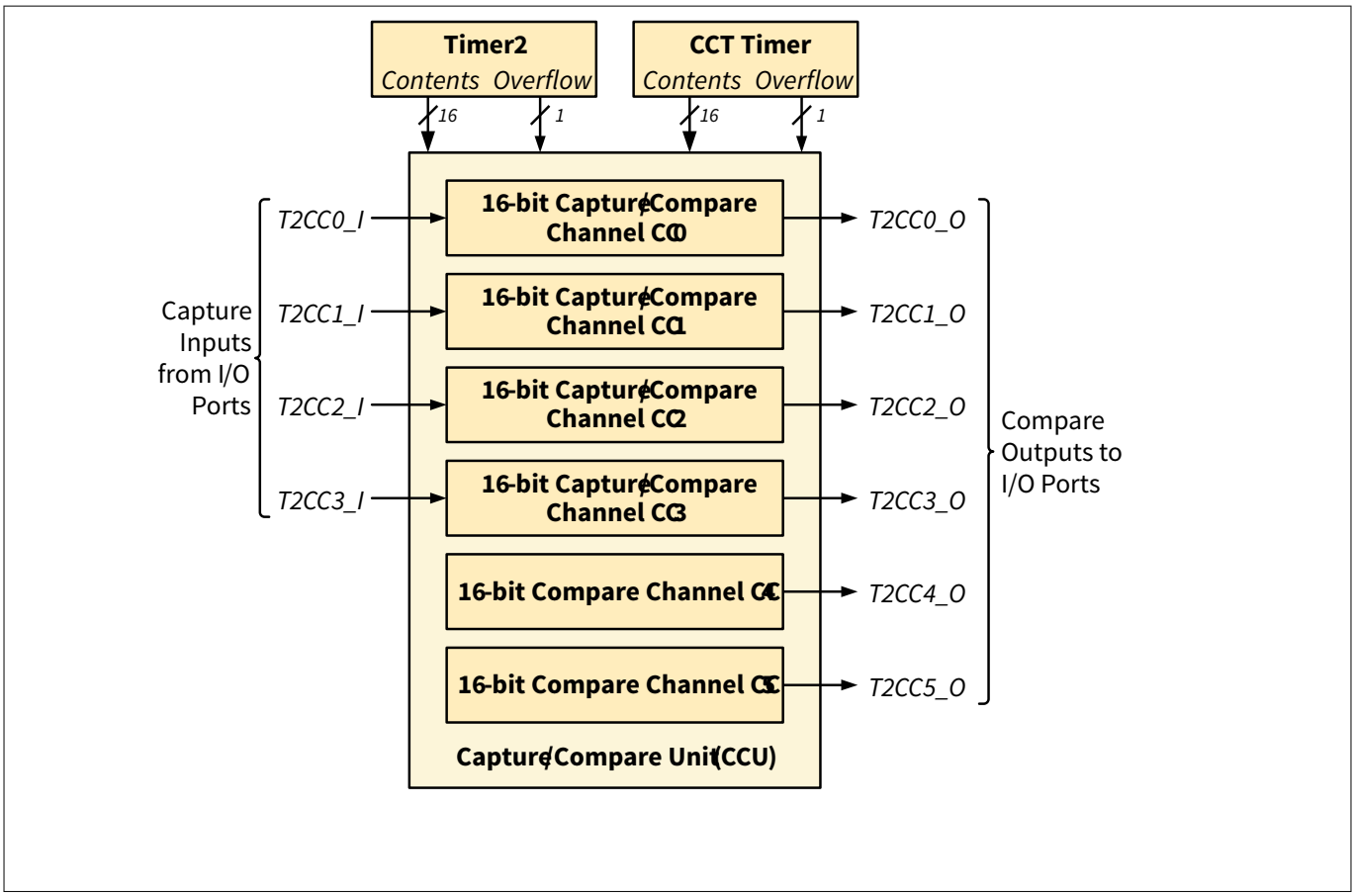


Figure 897 T2CCU Capture/Compare Unit (CCU) overview

The main features of the CCU are:

- 4 channels (CC0-CC3), which can be used for Capture or Compare Operation
- 2 channels (CC4-CC5), which can be used for Compare Operation
- Each capture channel (CC0-CC3) has a capture input T2CCx_I, coming from the I/O pins
- Each compare channel (CC0-CC5) has a compare output signal T2CCx_O, leading to the I/O pins
- Each channel can individually select either Timer 2 or the CCT as a base timer (using the contents and the overflow signal of the selected timer). The capture/compare resolution for a channel is determined by the operating frequency of the selected base timer
- Compare Operation: Various compare modes are available:
  - Compare Mode 0 without dead-time generation
  - Compare Mode 0 with dead-time generation
  - Compare Mode 1
  - Concurrent Compare Mode
- Capture Operation: Two modes are available:
  - Hardware-triggered capture
  - Software-triggered capture

The input/output signals of the CCU are assigned as alternate functions of general-purpose I/O ports. The compare outputs share the same pins as the capture inputs, and the ports are (partly) shared with External

Interrupt Inputs, see the following tables. The corresponding external interrupt request flags are set for the capture/compare function. See also CCU Interrupt and I/O Connections.

Table 1715 Input/output signals of the CCU0

Output/input	Function
T2CCU0_0_O / T2CCU0_0_I / EXINT3	Compare output/Capture input for channel 0
T2CCU0_1_O / T2CCU0_1_I / EXINT4	Compare output/Capture input for channel 1
T2CCU0_2_O / T2CCU0_2_I / EXINT5	Compare output/Capture input for channel 2
T2CCU0_3_O / T2CCU0_3_I / EXINT6	Compare output/Capture input for channel 3
T2CCU0_4_O	Compare output for channel 4
T2CCU0_5_O	Compare output for channel 5

Table 1716 Input/output signals of the CCU1

Output/input	Function
T2CCU1_0_O / T2CCU1_0_I / EXINT8	Compare output/Capture input for channel 0
T2CCU1_1_O / T2CCU1_1_I / EXINT9	Compare output/Capture input for channel 1
T2CCU1_2_O / T2CCU1_2_I / EXINT10	Compare output/Capture input for channel 2
T2CCU1_3_O / T2CCU1_3_I / EXINT11	Compare output/Capture input for channel 3
T2CCU1_4_O	Compare output for channel 4
T2CCU1_5_O	Compare output for channel 5

39.3.11.4.1 Capture/Compare operation

The following sections describe the compare and capture operations of the CCU in detail.

***Note**: For defined behavior, it is required of the user to initialize properly all the T2CCU related SFRs before enabling any T2CCU capture or compare mode.*

39.3.11.4.2 Compare operation

In compare operation, the contents of a compare register can be regarded as a “time stamp”, at which the corresponding output reacts in a predefined way (with either a positive or negative transition). Variation of this “time stamp” changes the waveform of a rectangular output signal at the pin. As a variation of the duty cycle of a periodic signal, this may be used for pulse width modulation as well as for a continually controlled generation of any kind of square waveforms. Various compare modes are available to cover a wide range of possible applications.

The value stored in a compare register is constantly compared to the current value of the selected base timer. When the timer value is equal to or higher than the compare value, a “compare match” is detected, and appropriate actions are taken.

The following figure illustrates the configuration of the CCU for compare operation. The following items are important for the compare operation:

- Each 16-bit compare register CCx (x=0..5) is represented by two 8-bit registers, named T2CCU_CCxH (high-byte register) and T2CCU_CCxL (low-byte register)

- The desired compare mode is selected by bit T2CCU_COCON.COMOD and applies to all enabled compare channels
- For output polarity control and the dead-time generation, the channels are organized into two groups. GroupA comprises channels CC0, CC2, and CC4, while GroupB comprises channels CC1, CC3, and CC5
- In all compare modes, the new output value arrives at the port pin within the same PCLK cycle in which the internal compare signal is activated
- The compare output is only actively driven when the compare function is enabled. When the compare function is enabled by writing to a channel's CCMx bits, the respective compare output is latched (as initial value) based on the settings of the channel's registers and bit
- The rising edge of the compare match signal is the active edge for all compare match operations
- Each compare register has a shadow transfer mechanism which allows glitch-free pulse width modulation; see detailed description below

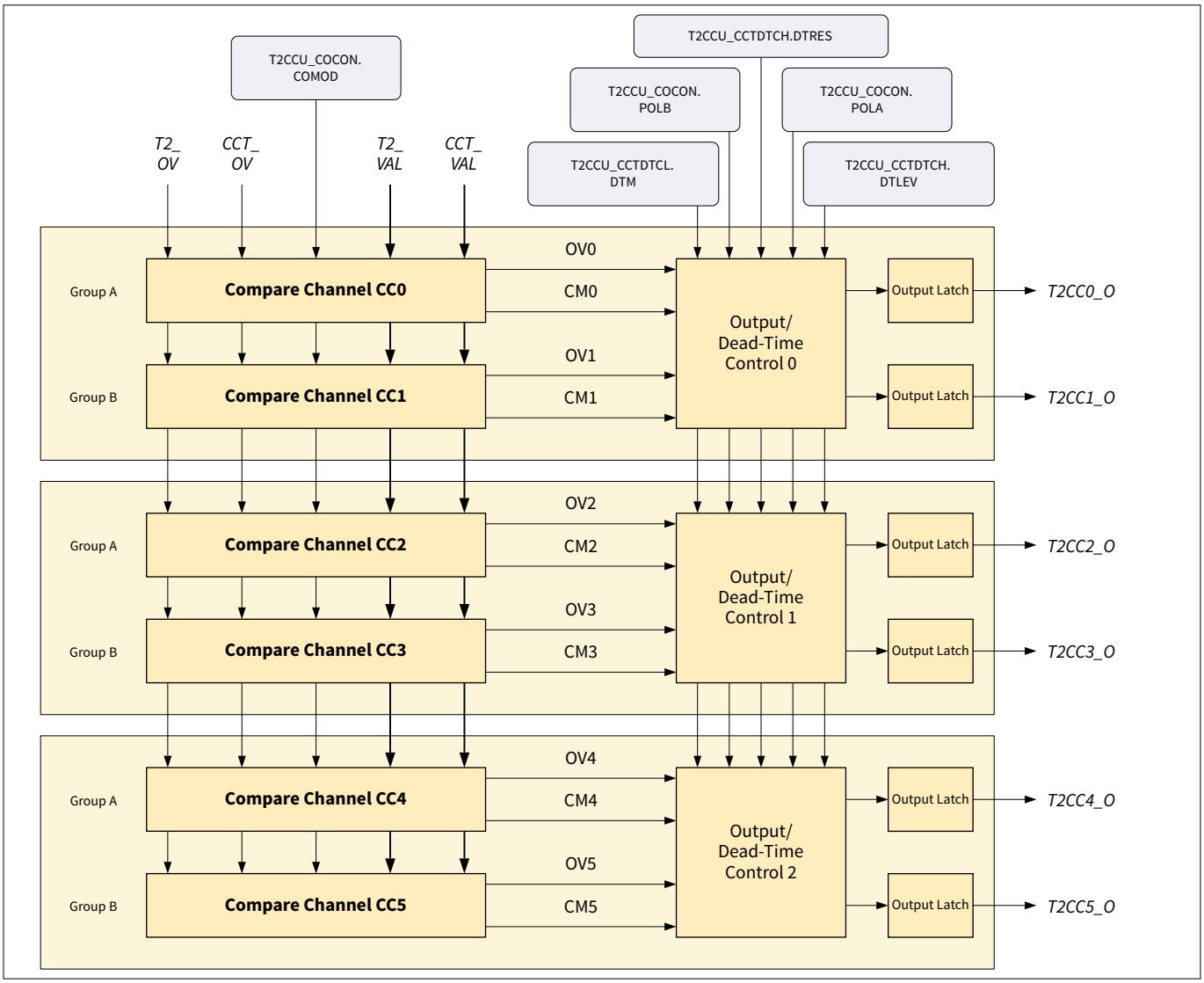


Figure 898 CCU compare channel configuration overview

Shadow registers for compare registers

To update a 16-bit compare value, two write accesses are required because of the two 8-bit registers of a compare channel. To allow for a glitch free update of a compare value without an intermediate "wrong" value, shadow registers are implemented. Software write accesses do not target the actual compare register, but the

respective shadow register. After the write access, the value is transferred from the shadow register into the actual compare register via one of two options:

1. Hardware Control: The transfer from the shadow register into the compare register is triggered by the overflow signal of the selected base timer. In this case, the update of the compare registers is controlled by hardware and is therefore independent from any service delay. This option is selected by setting bit T2CCU_COSHDW.TXOV to 1
2. Software Control: The transfer from the shadow register into the compare register is triggered by setting the corresponding bit T2CCU_COSHDW.COOUTx through software. Bit T2CCU_COSHDW.COOUTx will be cleared automatically by hardware at the next PCLK, after being set. This option is selected by setting bit T2CCU_COSHDW.ENSHDW to 1

In both cases, the rising edge of the event pulse signal is the active edge for the transfer from shadow register to compare register.

Note that it is not allowed to set both bits, T2CCU_COSHDW.TXOV and T2CCU_COSHDW.ENSHDW, at the same time. If bit T2CCU_COSHDW.TXOV is set to enable shadow transfer on timer overflow, bit T2CCU_COSHDW.ENSHDW will be cleared by internal hardware to disable shadow transfer by software.

The following figure illustrates the shadow transfer mechanism.

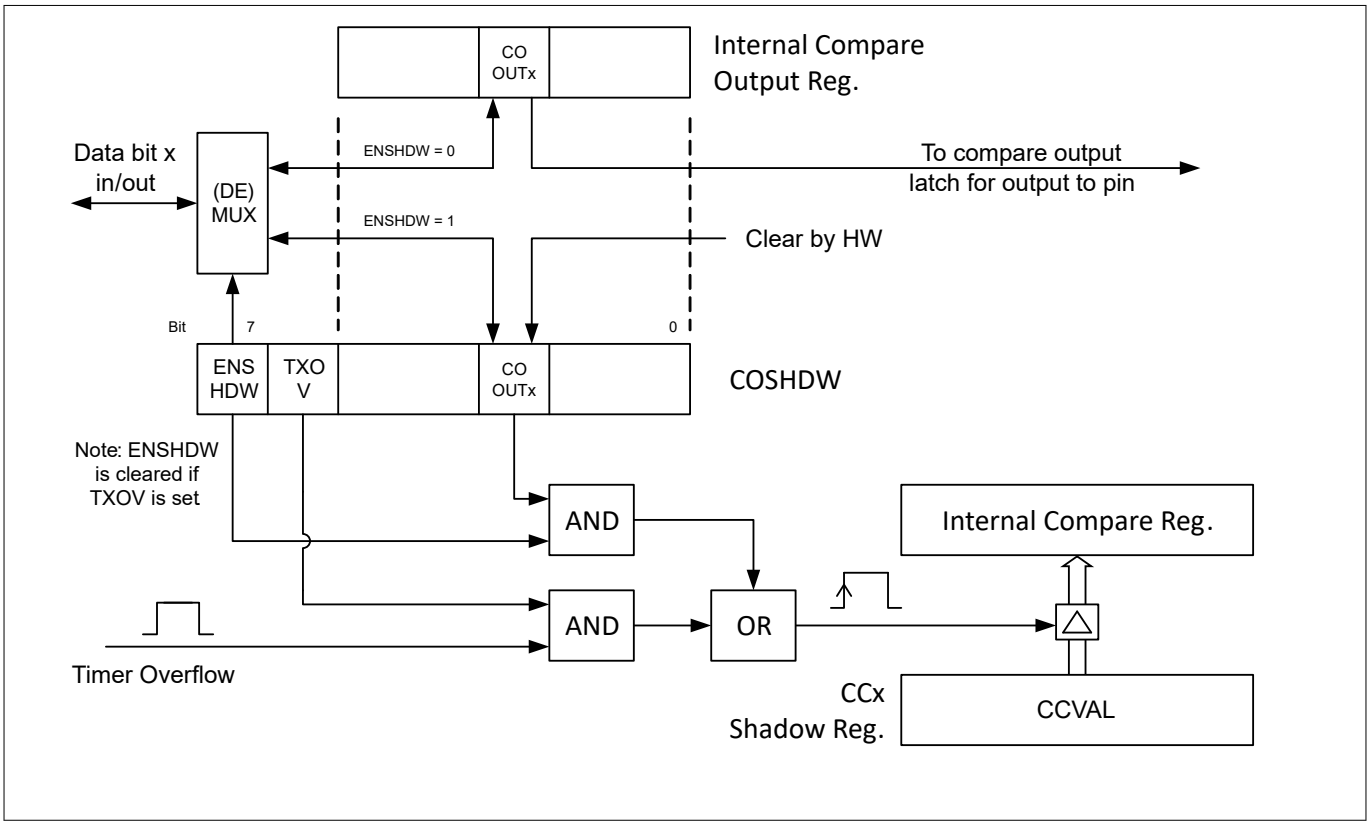


Figure 899 Compare value shadow transfer mechanism and the dual function of bits COOUTx

Mapped compare output bits

In 'Compare Mode 1' and 'Concurrent Compare Mode', bits T2CCU_COSHDW.COOUTx are used to set the polarity of the respective compare output. To select between the two functions of the T2CCU_COSHDW.COOUTx bits (shadow transfer versus output polarity), the state of bit T2CCU_COSHDW.ENSHDW is used (see figure above):

- When T2CCU_COSHDW.ENSHDW=1, bits T2CCU_COSHDW.COOUTx can be directly accessed to trigger the shadow transfer of the compare value
- When T2CCU_COSHDW.ENSHDW=0, writing or reading to the T2CCU_COSHDW.COOUTx bits actually accesses an internal compare output register instead of theT2CCU_COSHDW. COOUTx bits

The following compare modes are available, described in detail in the following sections:

Related information

Compare mode 0 (without dead-time control) on page 6601
Compare mode 0 with dead-time control on page 6603
Compare mode 1 on page 6606
Concurrent compare mode on page 6607

Compare mode 0 (without dead-time control)

In Compare Mode 0 without dead-time control, the output signal is affected by a compare match and by the overflow of the associated timer. A compare match sets the output to active level, while the timer overflow resets the output to the inactive level.

Compare Mode 0 is ideal for generating pulse width modulated output signals, which in turn can be used for digital-to-analog conversion through a filter network or by the controlled device (for example the inductance of a DC or AC motor). Compare Mode 0 may also be used for providing output clocks with initially defined period and duty cycle. This usage needs the least CPU time. Once set up, the output goes on oscillating without any CPU intervention.

The next figure shows a timing diagram example of Compare Mode 0, followed by a functional diagram of this mode. Please see figure Output & Dead-Time Generation Block for the output control.

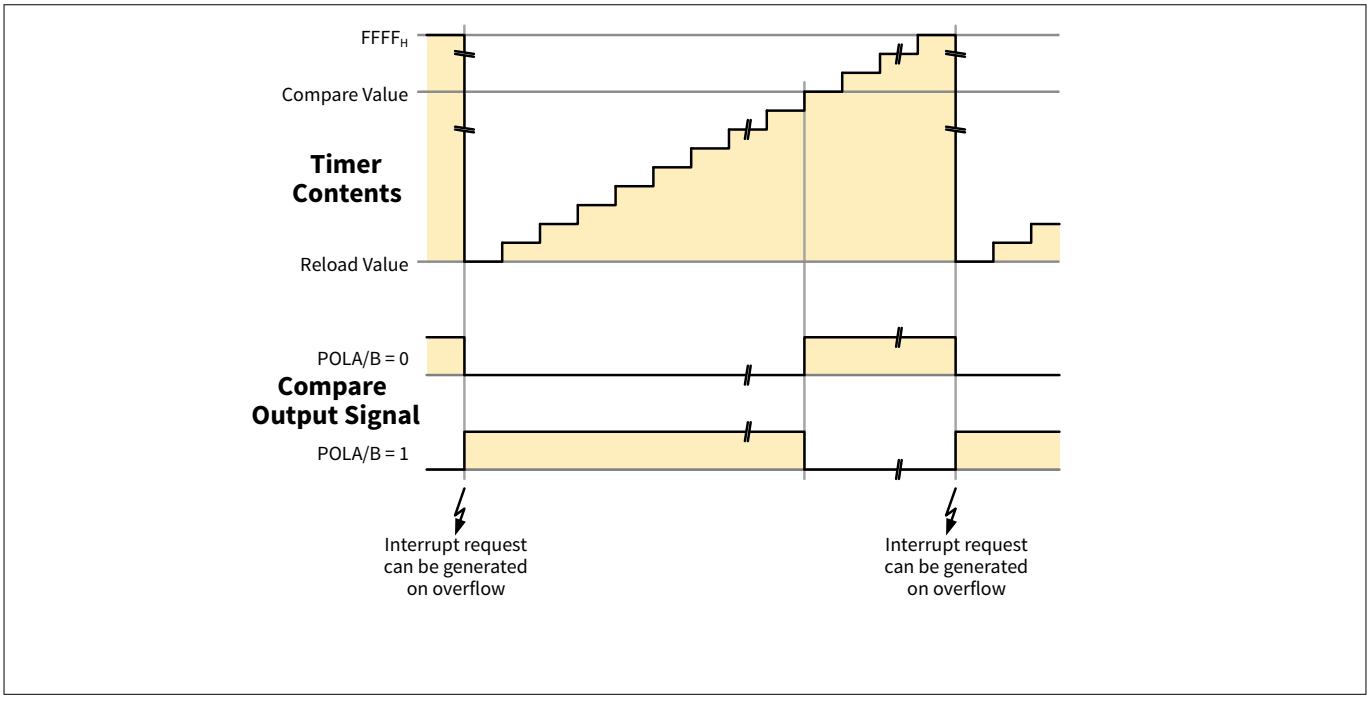


Figure 900 Channel output example in compare mode 0 (without dead-time control)

To configure Compare Mode 0, the following steps need to be done:

- Set bit-field T2CCU_COCON.COMOD to 00_B to select Compare Mode 0
- Set bit T2CCU_COSHDW.TXOV to 1_B (the usual way to perform the compare shadow value transfer in Compare Mode 0 is through the timer overflow)
- Configure the desired output polarity of the compare outputs through bits T2CCU_COCON.POLA and T2CCU_COCON.POLB. Bit T2CCU_COCON.POLA applies to channels 0, 2, and 4 (channel group A), while T2CCU_COCON.POLB applies to channels 1, 3, and 5 (channel group B)

***Note**: The two steps listed above can be done in one write access to register T2CCU_COCON. However, the enabling of compare channels 4 and 5 (see following step) must be done in a separate write access to register T2CCU_COCON, to ensure a glitch-free compare output level.*

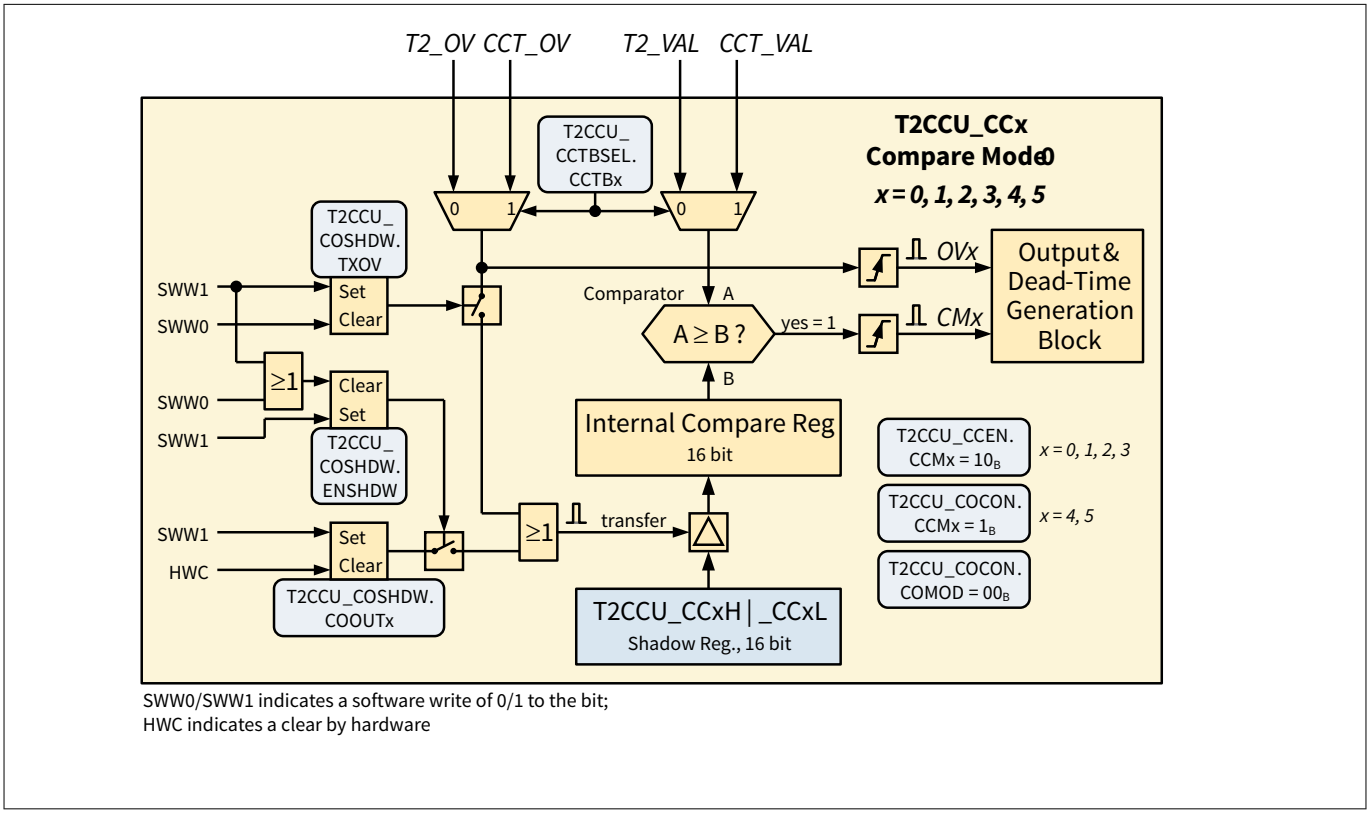
- For compare channels 0 through 3, individually enable the compare mode for the desired compare registers by setting bit T2CCU_CCEN.CCMx to 10_B. For compare channels 4 and 5, individually enable the compare mode for the desired compare registers by setting bit T2CCU_COCON.CCMx to 1_B

The compare output is immediately set to the appropriate level when the associated compare register is enabled through bit/bit-field T2CCU_COCON.CCMx:

- It is set to the active level if the base timer contents are equal to or greater than the compare register contents
- It is set to the inactive level if the base timer contents are less than the compare register contents

Either Timer 2 or the CCT timer can be used as base timer in Compare Mode 0. The shadow transfer of a new compare value is usually triggered by the overflow of the selected base timer (set bit T2CCU_COSHDW.TXOV to 1_B).

Note that no interrupt is generated when a compare match occurs. See CCU Interrupt and I/O Connections. for further information.



SWW0/SWW1 indicates a software write of 0/1 to the bit;
HWC indicates a clear by hardware

Figure 901 Functional diagram of compare mode 0 (without dead-time control)

Operation in the first timer period after initialization

The following needs to be regarded concerning the compare output level for the first timer period after initialization:

- Although a compare value might have been written to the compare shadow register before enabling the compare operation, the actual compare register will contain 0000_H (if no transfer from the shadow register to the actual compare register has been forced through software, see below)
- In this case, the value of the timer will be greater than or equal to the value in the compare register (whether the timer contains the initial value of 0x0 or has been programmed to another value)

- Thus, the condition of a “compare match” is true (timer ≥ compare value) and the compare output level will be set to the active level
- On the following timer overflow, the first compare value shadow transfer takes place (bit T2CCU_COSHDW.TXOV = 1_B) and the output will go to its inactive level if the programmed compare value is > 0000_H
- From this second timer period on, the output will switch from inactive to active level on compare match and vice verse on timer overflow

In order to get a proper compare operation already during the first timer period, a compare value shadow transfer must be forced by software. This can only be done after compare mode is enabled

- After enabling the compare mode and before starting the timer, a compare value shadow transfer must be forced by software
- Write a value greater than the timer value to the compare shadow register
- Set bit T2CCU_COSHDW.ENSHDW to 1_B (clear bit T2CCU_COSHDW.TXOV to 0_B with the same write access), and set T2CCU_COSHDW.COOUTx to 1_B
- After this, clear bit T2CCU_COSHDW.ENSHDW to 0_B and set bit T2CCU_COSHDW.TXOV to 1_B with the same write access (bit T2CCU_COSHDW.COOUTx is automatically cleared by hardware)
- Now the actual compare register contains the programmed value (> timer value) and the compare output goes to inactive level
- The timer can now be started for its first period

PWM signal from 0% to 100% duty cycle in compare mode 0

The active level of a PWM signal refers to the compare output signal level from the time of compare match until the next timer overflow. To generate signals with duty cycles of 100% or 0%, the following methods can be used:

- To support a 100% duty cycle for a compare channel with T2CCU_COCON.POLA/B=0 (active level = high), the compare value must be set equal to the timer reload value. In this case, the internal hardware will not cause the compare output signal to the inactive level on timer overflow and the output stays at the active level until the next timer overflow when the compare value is no more equal to the timer reload value
- To support a 0% duty cycle for a compare channel with T2CCU_COCON.POLA/B=0, the compare value must be smaller than the timer overflow reload value. In this case, the timer reload value must not be zero. The compare match event will not happen and the compare output stays at the inactive level (low)

The mechanisms described above apply correspondingly for the case that the chosen active level=low (T2CCU_COCON.POLA/B=1).

Compare mode 0 with dead-time control

Compare Mode 0 offers an additional feature for the generation of complementary signals with a so-called dead-time between the deactivation of one signal and the activation of the other signal.

The generation of (complementary) signals with dead-time for the high-side and the low-side switches of a power inverter phase is based on twin grouping of the consecutive compare channels – one from group A and the other from group B. The three compare channel pairs therefore are comprised of channel 0 with 1, channel 2 with 3, and channel 4 with 5.

The Compare Mode 0 with dead-time generation can be used under the following conditions:

- All involved compare channels must use the CCT timer
- Both channels of a channel pair must have the same compare value (CC0 = CC1; CC2 = CC3; CC4 = CC5)
- All used channels must have the dead-time generation enabled (bit T2CCU_CCTDTCH.DTEy = 1)

- All high-side and low-side switches must have the same active polarity. For example, if the high-side switch should be active while the CCT timer value is above the compare value, then the low-side switch should be active while the CCT timer value is below the compare value
- The active signal polarity is controlled by the bits T2CCU_COCON.POLA (for compare channel group A) and T2CCU_COCON.POLB (for compare channel group B). T2CCU_COCON.POLA and T2CCU_COCON.POLB must be set to complementary values

The following figure illustrates the dead-time generation. This logic is implemented three times, one for each compare channel pair. It mainly consists of an 8-bit down-counter (dead-time counter, clocked with the CCT count rate), and a logic to manipulate the output signals.

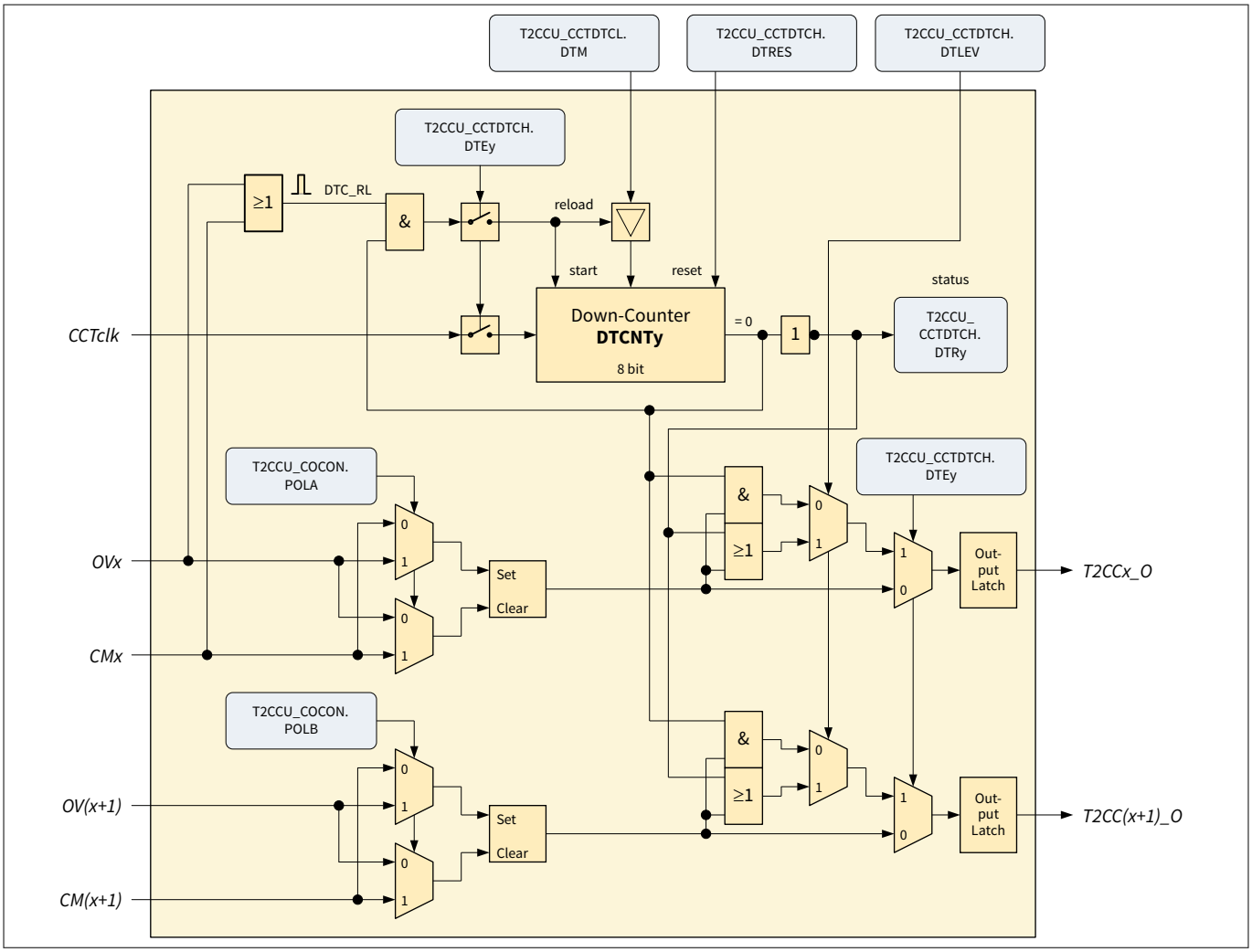


Figure 902 Output and dead-time generation block

The following list describes the detailed operation:

- If dead-time is not enabled for a channel (bit T2CCU_CCTDTCH.DTEy = 0), the output signal polarity is solely controlled by bits T2CCU_COCON.POLA/POLB
- If dead-time is enabled for a channel (bit T2CCU_CCTDTCH.DTEy = 1), the output signal polarity is controlled by bits T2CCU_COCON.POLA/POLB and the dead-time control logic:
  - As long as the dead-time counter is not running (signal '=0' is active, status flag T2CCU_CCTDTCH.DTRy=0), the output signals are determined by bits T2CCU_COCON.POLA/POLB
  - Each instance of the CCT timer overflow or a compare match event (of channel from group A, rising edge) triggers the corresponding dead-time counter. The trigger pulse (DTC_RL) reloads the dead-time

counter with the value programmed in bit-field T2CCU_CCTDTCL.DTM, and starts it. This can only take place if the dead-time feature is enabled by bit T2CCU_CCTDTCH.DTEx and while the counter is zero

- Once started, the dead-time counter counts down until it reaches 0. Then it stops counting
- As long as the dead-time counter is running, the output signals are forced to their programmed dead-time level. This level is determined by bit T2CCU_CCTDTCH.DTLEV. T2CCU_CCTDTCH.DTLEV must be set such that the outputs a forced to the inactive level during the dead-time

Please see also the section 'Operation in the first timer period after initialization' in related information, which applies in a similar manner for the dead-time operation.

The following figure illustrates the resulting signal waveforms for the dead-time operation (example with T2CCU_COCON.POLA= 0 and T2CCU_COCON.POLB=1).

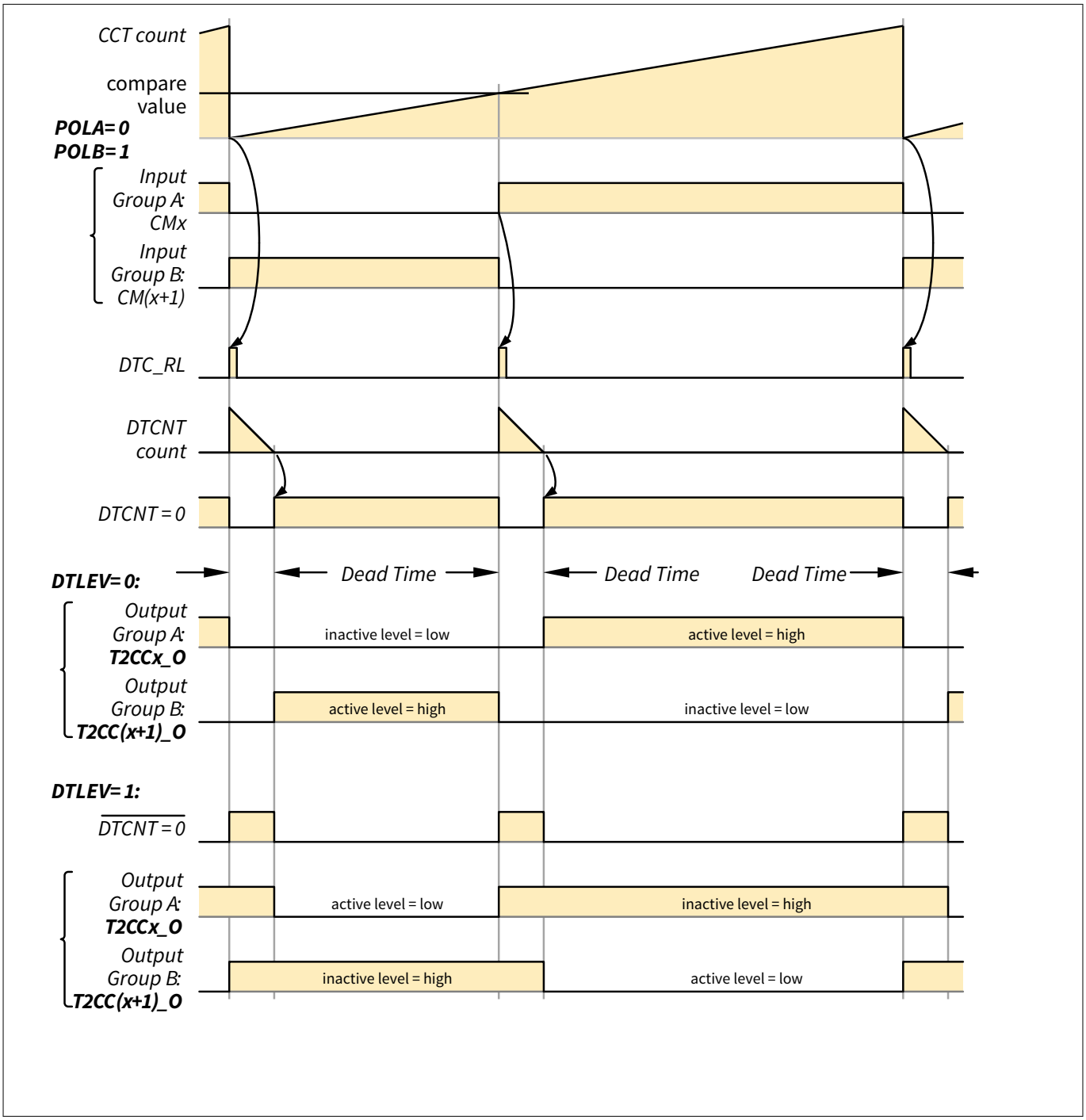


Figure 903 PWM-signals with dead-time generation (illustrated for one channel pair)

***Note**: While the channels pairs can be individually enabled or disabled for dead-time generation through their DTEy bits, the dead-time (DTM) and the dead-time level (DTLEV) apply to all channels. In addition, the reset control (DTRES) applies to all three dead-time counters.*

***Note**: The dead-time logic is designed for the application case in which one and only one compare event is generated within one timer period. If more than one compare event happens within one timer period or if two or more timer overflows happen without a compare event in between, proper functionality of the dead-time logic is not assured. In addition, the timer must not be disabled while the dead-time is in progress or between a timer overflow and a compare match.*

***Note**: When the dead-timeT2CCU_CCTDTCL. DTM is set to 0_H, no dead-time will be generated. This is equivalent as if the dead-time function is not enabled (for example DTEy = 0_B).*

Related information

Compare mode 0 (without dead-time control) on page 6601

Compare mode 1

In Compare Mode 1, the output signal is only updated on a compare match, but not by the respective timer overflow (as in Compare Mode 0). In Compare Mode 1, the software adaptively determines the transition of the output signal. It is commonly used when output signals are not related to a constant signal period (as in a standard PWM Generation), but must be controlled very precisely with high resolution and without jitter. In Compare Mode 1, the compare output signal level is under the full control of the software.

The following figure shows a functional diagram of Compare Mode 1.

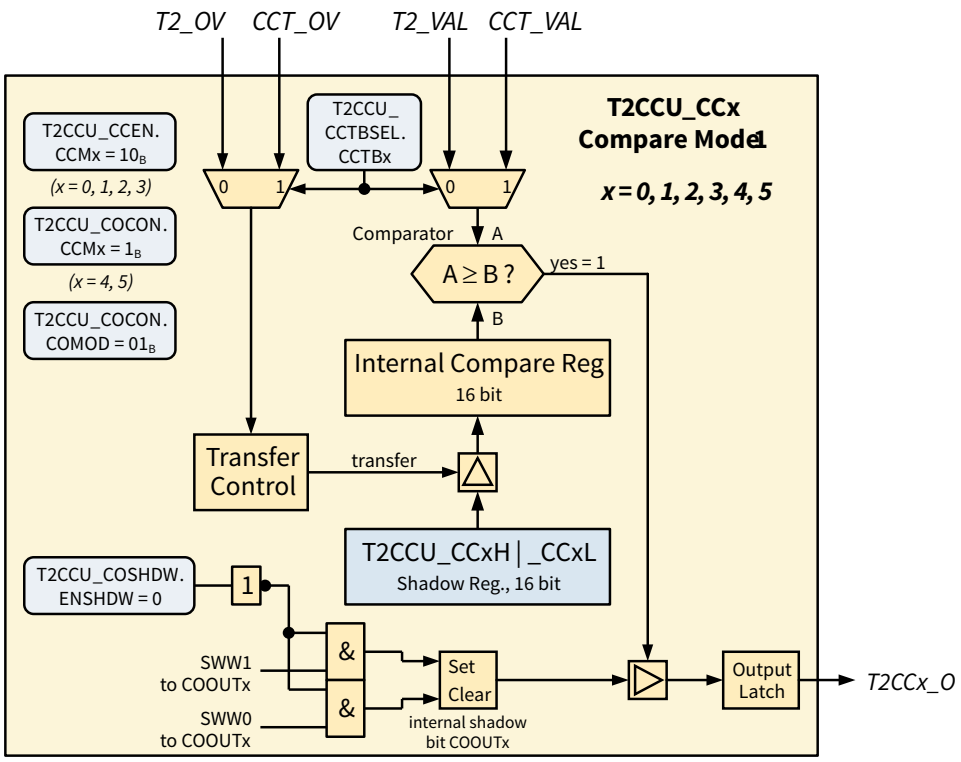


Figure 904 Functional diagram of compare mode 1

To configure Compare Mode 1, the following steps need to be done:

- Set bit-field T2CCU_COCON.COMOD to 01_B to select Compare Mode 1
- For compare channels 0 through 3, individually enable the compare mode for the desired compare registers by setting bit-field T2CCU_CCEN.CCMx to 10_B. For compare channels 4 and 5, individually enable the compare mode for the desired compare registers by setting bit T2CCU_COCON.CCMx to 1_B
- The shadow transfer of a new compare value is triggered by the overflow of the selected base timer (T2CCU_COSHDW.TXOV=1)
- The software writes the desired output level for the next compare match to the T2CCU_COSHDW.COOUTx bit(s) of the internal compare output register (in this case, bit T2CCU_COSHDW.ENSHDW must be cleared to 0). This new value will not appear at the output until the next compare match occurs. Thus, it is possible to choose, independently for each compare channel, whether the output signal is to toggle (1-to-0 or 0-to-1) or to remain at the same level, when a compare match occurs

The compare output is immediately set to the value initialized to bit T2CCU_COSHDW.COOUTx (of the internal compare output register), when the associated compare register is enabled through bitT2CCU_CCEN.CCMx.

Either Timer 2 or the CCT timer can be used as base timer in Compare Mode 0. The shadow transfer of a new compare value must be triggered by the overflow of the selected base timer.

Note that no interrupt is generated when a compare match occurs. See CCU Interrupt and I/O Connections for further information.

Concurrent compare mode

Concurrent Compare Mode allows to manipulate more than one output with the same compare event. Up to all six compare outputs can be updated concurrently when a compare channel 0 event occurs.

Concurrent compare is an ideal and effective mode for applications where more than one synchronous output signal is to be generated. Such applications could include, for example, a multiple-phase stepper motor control, or control of ignition coils of a car engine. These applications generally require predefined bit-patterns to be put to an output port at precisely predefined moments.

The following figure illustrates the function of Concurrent Compare Mode.

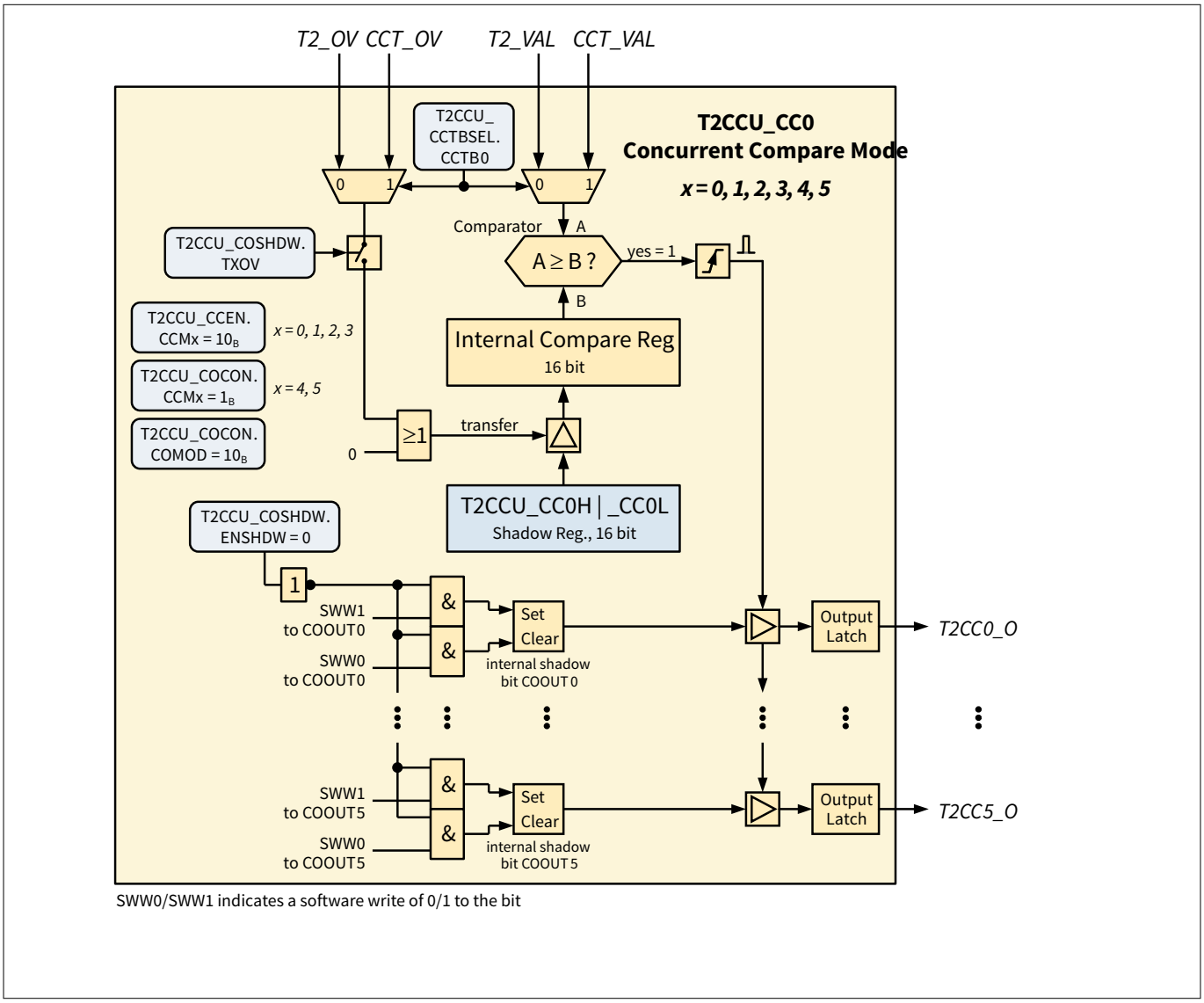


Figure 905 Functional diagram of concurrent compare mode

To operate the CCU in Concurrent Compare Mode, the following configurations must be made:

- Bit T2CCU_COSHDW.ENSHDW must be cleared to 0
- The shadow transfer is triggered by the overflow of the selected base timer; set bit T2CCU_COSHDW.TXOV to 1_B
- Set the compare mode to Concurrent Compare Mode, by setting bitfield T2CCU_COCON.COMOD to 10_B
- Write the initial output values to the internal compare output bits T2CCU_COSHDW.COOUTx of the desired outputs
- Only Compare Channel 0 is used for the outputs. Write the required compare value to T2CCU_CC0H and T2CCU_CC0L
- Enable the desired output signals through their individual enable bits (for channels 0 to 3: Set bit-field T2CCU_CCEN.CCMx to 10_B; for channels 4 and 5: Set bit T2CCU_COCON.CCMx to 1_B)

***Note**: The compare outputs are latched with the corresponding values initialized to the T2CCU_COSHDW.COOUTx bits of the internal compare output register when setting the CCMx enable control for the desired outputs (make sure to first select Concurrent Compare Mode through T2CCU_COCON.COMOD, and to program the desired values into the T2CCU_COSHDW.COOUTx bits).*

- During steady operation, write the new compare value to register pair T2CCU_CC0H and T2CCU_CC0L, and the new output values to bits T2CCU_COSHDW.COOUTx
- With the following timer overflow, the new compare value will be used during the new timer period, and the new output values will appear at the outputs when the compare match occurs

The following figure illustrates two examples of how four rectangular waveforms can be generated at the port using a pattern table. The patterns are written to the internal compare output register (with T2CCU_COSHDW.ENSHDW=0) before the corresponding timer count is reached. The (future) timer count at which the pattern shall be available at the port must be loaded to the compare shadow registers T2CCU_CC0H and T2CCU_CC0L, and transferred to the internal compare registers. To manage the waveform generation, one example uses a time schedule table with list of compare values, the other example makes use of a fixed compare value (such as $FFF0_H$) with shorter timer period. In these ways, each channel can be controlled at predefined time(s) by the user on whether to toggle or not.

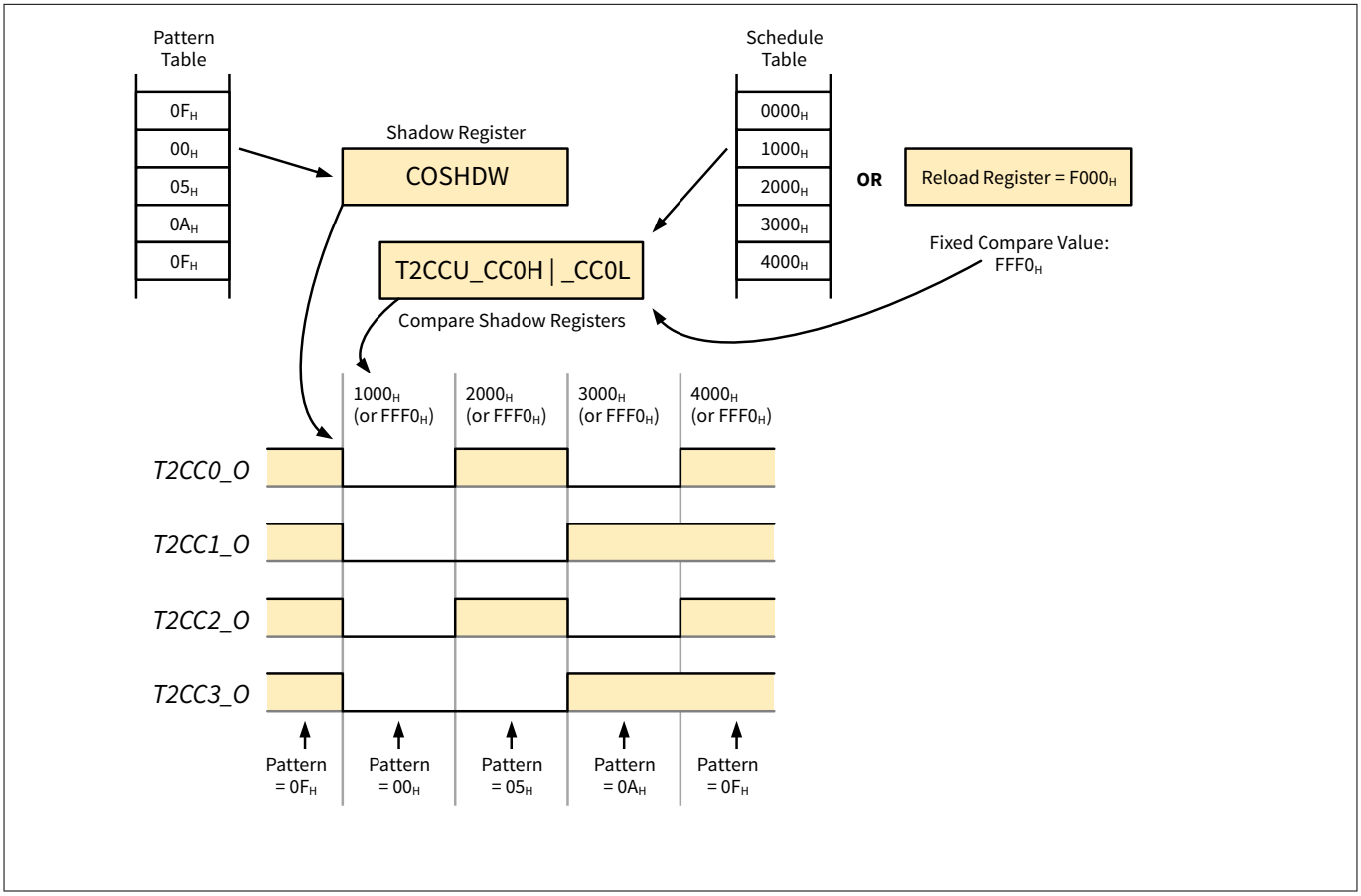


Figure 906 Two examples of waveform generation in concurrent compare mode (by schedule table of compare values or fixed compare value)

Using interrupts in combination with the compare function

There are no interrupt requests generated within the CCU on a compare match. Thus, it is not possible to directly trigger an interrupt service routine on a compare match to handle the programming of new compare and/or output values.

There are two options to overcome this:

- In many cases, the overflow interrupt request of the associated base timer can be used
- Since the compare outputs are connected to the same pins which serve as external interrupt inputs (and CCU capture inputs), these interrupts can be used

These options offer the following possibilities:

- Compare Mode 0 (with or without dead-time): In this mode, a compare match will always show a defined signal edge at the compare output. This signal edge will also appear on the associated input line for the external interrupt input. By appropriately programming the external interrupt configuration, an interrupt request to the CPU can be generated. In the associated service routine, the next compare value can be written to the compare shadow register
- Compare Mode 1 and Concurrent Compare Mode: In these modes, the shadow transfer of compare and output value is always triggered with the overflow of the selected base timer. So, the associated overflow interrupt of the timer is the right trigger for a respective service routine to set up new compare and output values

See also related information for more information for configuring the external interrupt input.

Related information

Configuring the external capture input on page 6611

39.3.11.4.3 Capture function

The capture modes of the CCU allow the measurement of external signal pulse widths or the latching of time-stamps via the capture/compare registers CC0 through CC3. Each of the 16-bit registers can be used to latch the current 16-bit count value of the selected base timer. Two capture modes are provided for this function, a hardware- and a software-triggered capture:

- In Capture Mode 0, an external event at input T2CCUx_I latches the contents of the selected base timer into the associated capture register
- In Capture Mode 1, a software write to the low byte of a capture register latches the contents of the selected base timer into the associated capture register

The two capture modes can be established individually for each capture channel. That means, in contrast to the compare modes, it is possible to simultaneously select Mode 0 for one capture channel and Mode 1 for another channel. The following figure shows the capture operation a capture/compare channel.

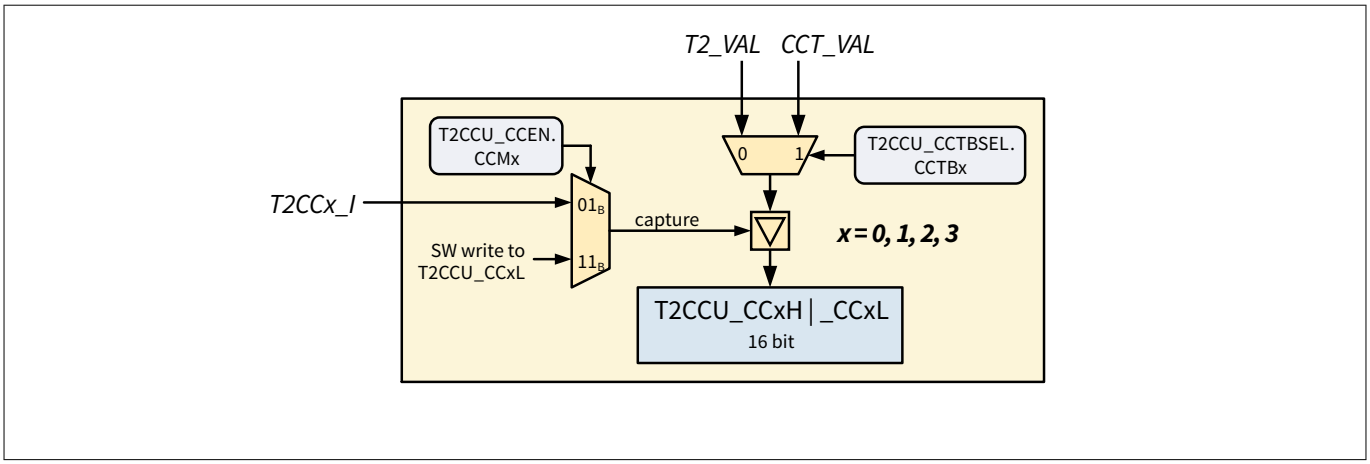


Figure 907 Functional diagram of capture mode

Capture mode 0

In Capture Mode 0, any selected external event (rising/falling/both edge) at the associated input (T2CC0_I through T2CC3_I) latches the channel timer contents to the corresponding channel capture register. The capture edge is defined by the corresponding external interrupt set up (each capture input shares the same pin with an external interrupt). The timer contents are latched into the corresponding capture register in the PCLK cycle following the one in which the external transition was recognized.

To configure Capture Mode 0, the following steps need to be done:

- Configure the external input signal (and interrupt request) associated with the desired capture channel (see related information)
- Select the base timer for the desired capture register through T2CCU_CCTBSEL.CCTBx
- Set bit-field T2CCU_CCEN.CCMx to 01_B (Capture Mode 0)

Related information

Configuring the external capture input on page 6611

Capture mode 1

In Capture Mode 1, a capture will occur upon writing to the low byte of the corresponding channel capture register. Thus, the software is able to read the contents of the channel timer “on-the-fly”. The value written to the capture register is irrelevant for this function. The timer contents will be latched into the channel capture register following the write instruction. In this mode, no interrupt request will be generated.

To configure Capture Mode 1, the following steps need to be done:

- Select the base timer for the desired capture register through T2CCU_CCTBSEL.CCTBx
- Set bit-field T2CCU_CCEN.CCMx to 11_B (Capture Mode 1)
- Write any value to the low-byte register T2CCU_CCxL to trigger the capture operation

Configuring the external capture input

Each capture input shares the same pin with an external interrupt. In fact, the capture input T2CCUx_I is derived from the external interrupt input such that it is connected from the edge selection multiplexer for the external interrupt to the CCU of the T2CCU. This is exemplary illustrated in next figure for capture input T2CCU0_0_I.

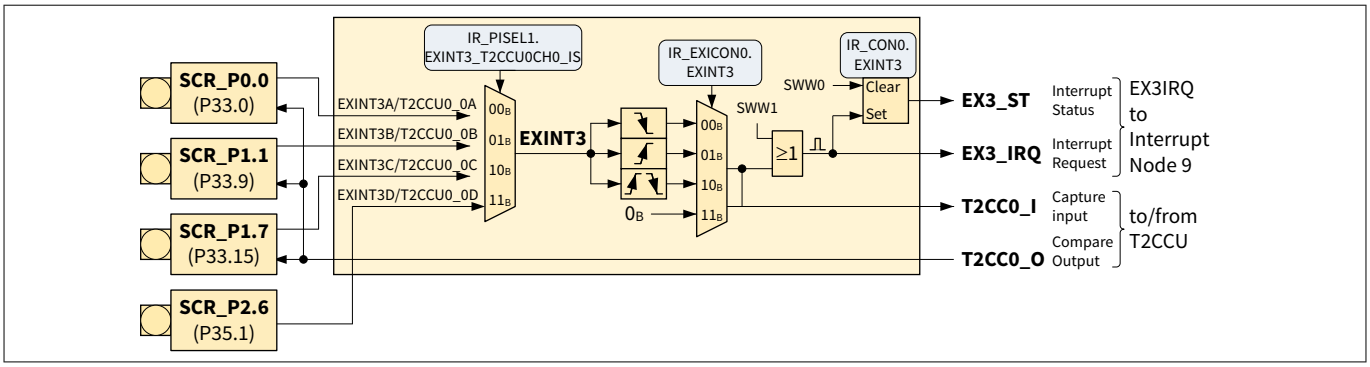


Figure 908 Capture input configuration

An external interrupt input is sampled in every PCLK cycle. When a sample shows a low (high) level in one PCLK cycle and a high (low) in the next PCLK cycle, a transition is recognized. The corresponding external interrupt request flag(s) (EXINT3 to EXINT6 for T2CCU0, or EXINT8 to EXINT11 for T2CCU1) is set. If the interrupt(s) is enabled, the CPU will vector to the appropriate interrupt service routine.

When a hardware capture mode is programmed for a CCU capture channel, the edge selected for the associated external interrupt input is also used as the relevant capture trigger. The timer contents are latched to the corresponding capture register in the PCLK cycle following the one in which the transition was identified.

When configuring an external capture input, the following points are relevant:

- The desired input pin is selected through bit IR_PISEL1.EXINTy_T2CCU0CHx_IS, or IR_PISEL2.EXINTz_T2CCU1CHx_IS
- The external interrupt inputs EXINT3 through EXINT6, share one interrupt node, XINTR9, with common enable bit for this interrupt node IEN1.EXM
- The external interrupt inputs EXINT8 through EXINT11, are on two interrupt nodes, XINTR10-11, with IEN1.ECCIP0-1 as enable bits
- The active edge is selected through bit-fields EXINTx (x=3 to 6, 8 to 11) in registers IR_EXICON0-2
- The interrupt status flags are: EXINT3 through EXINT6 in register IR_CON0, and EXINT8 through EXINT11 in register IR_CON2

Compare match interrupts

Since a compare match does not directly generate an interrupt request, the external interrupt inputs, which are shared with the compare outputs, can be used to generate an appropriate interrupt request, if required.

For this, the measures described in the section above for the capture inputs are taken. Since the channel is programmed for one of the compare modes, the capture input line has no effect.

The change of the compare output signal on a compare match is then used to trigger the associated interrupt request.

The main advantage in using compare match interrupts is to avoid unintentional overwriting of compare/ compare output registers before a match has been reached. This could happen when the CPU writes to the compare registers without knowing about the actual timer count.

***Note**: As an alternative, the timer overflow interrupt may be used in place of the compare match interrupt.*

39.3.11.5 CCU Interrupt and I/O Connections

This section describes the connections of the capture/compare channels to interrupts, as well as to the I/O ports (and other modules).

Usually, the interrupt request blocks are used which also provide the respective capture input T2CCx_I. However, since the capture/compare I/O lines are connected to more than on port pin, alternative interrupt requests can be used. In addition, the connection to other SCR modules is shown.

39.3.11.5.1 Capture/Compare Channel 0

The picture below illustrates (for both T2CCU0 and T2CCU1) the connections of capture/compare channel 0 to the ports, interrupt request blocks, and other SCR modules.

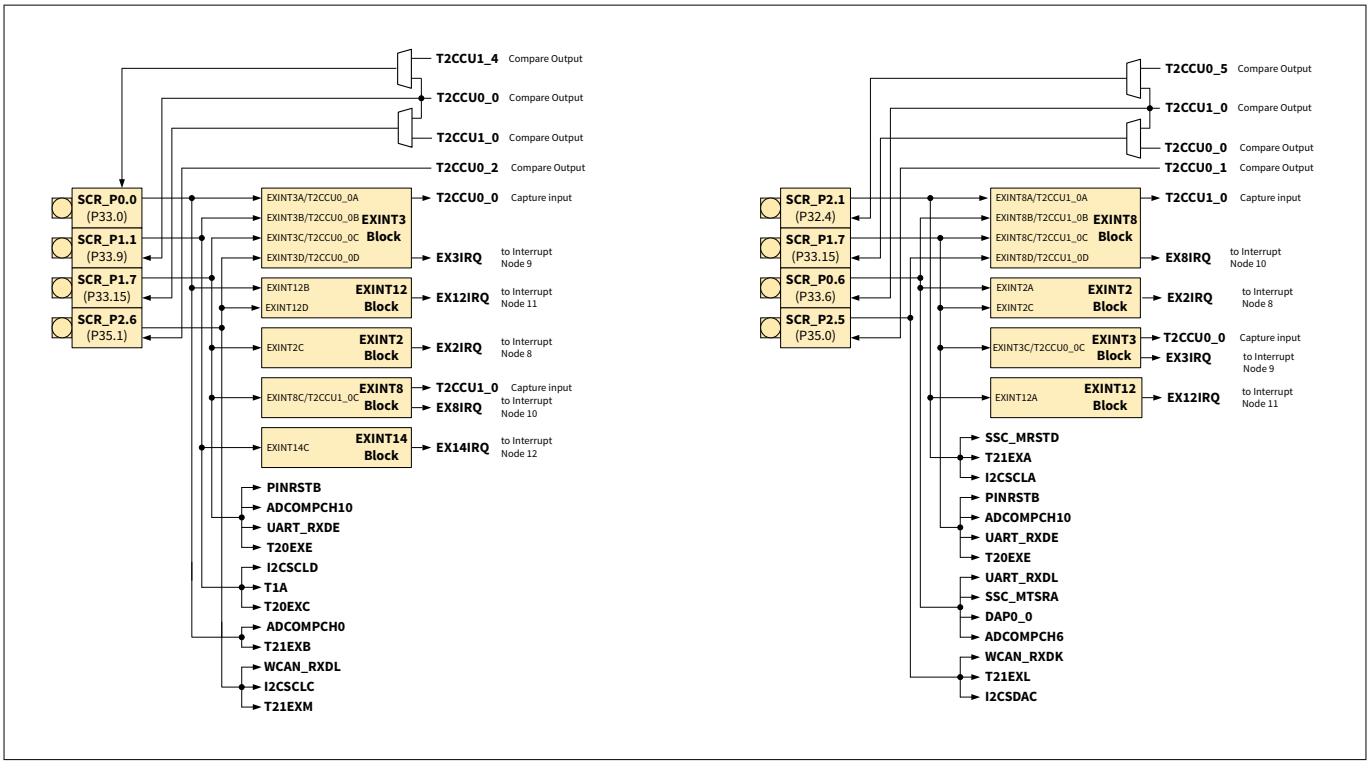


Figure 909 Interrupt and I/O Connections for Capture/Compare Channel 0

39.3.11.5.2 Capture/Compare Channel 1

The picture below illustrates (for both T2CCU0 and T2CCU1) the connections of capture/compare channel 1 to the ports, interrupt request blocks, and other SCR modules.

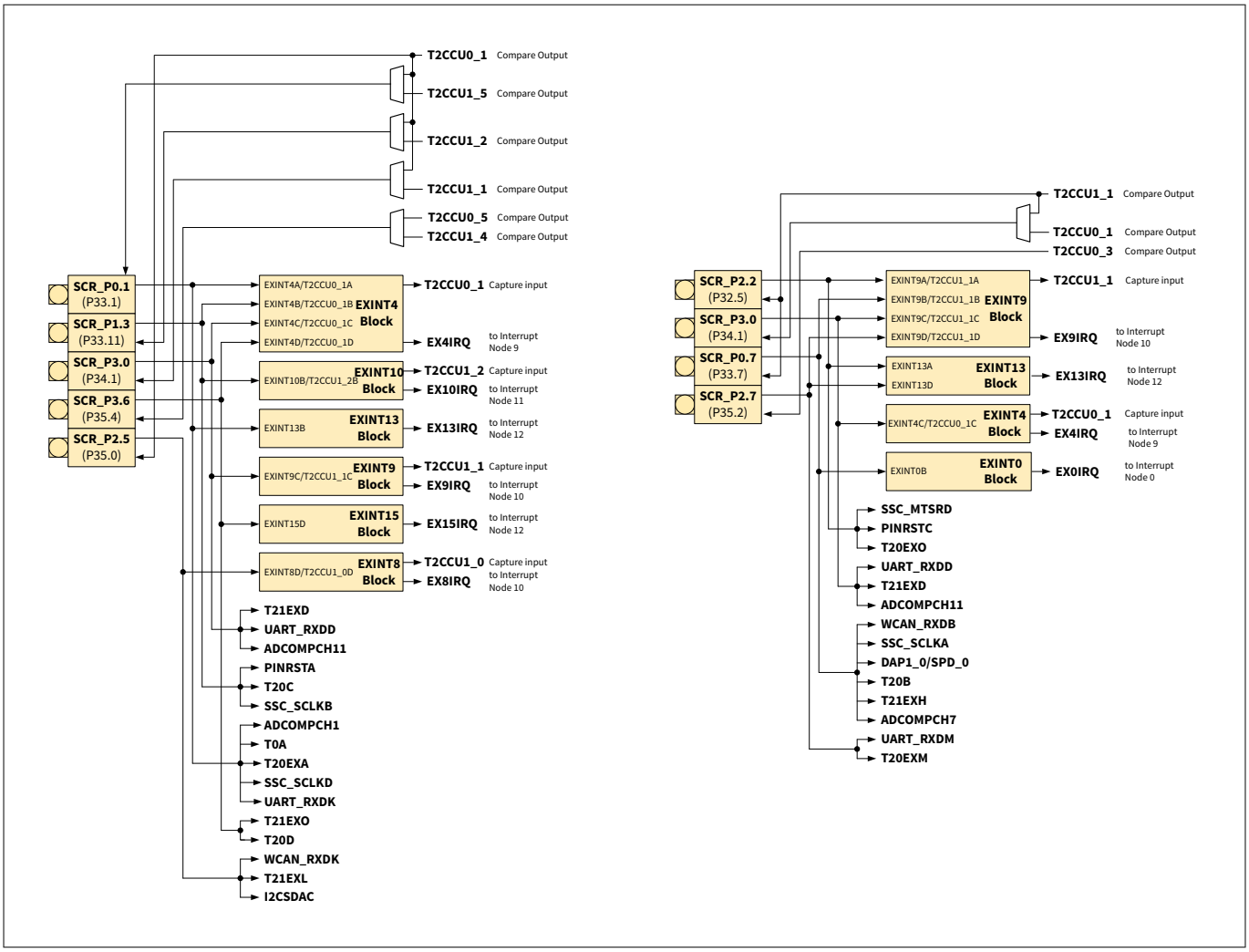


Figure 910 Interrupt and I/O Connections for Capture/Compare Channel 1

39.3.11.5.3 Capture/Compare Channel 2

The figure below illustrates (for both T2CCU0 and T2CCU1) the connections of capture/compare channel 2 to the ports, interrupt request blocks, and other SCR modules.

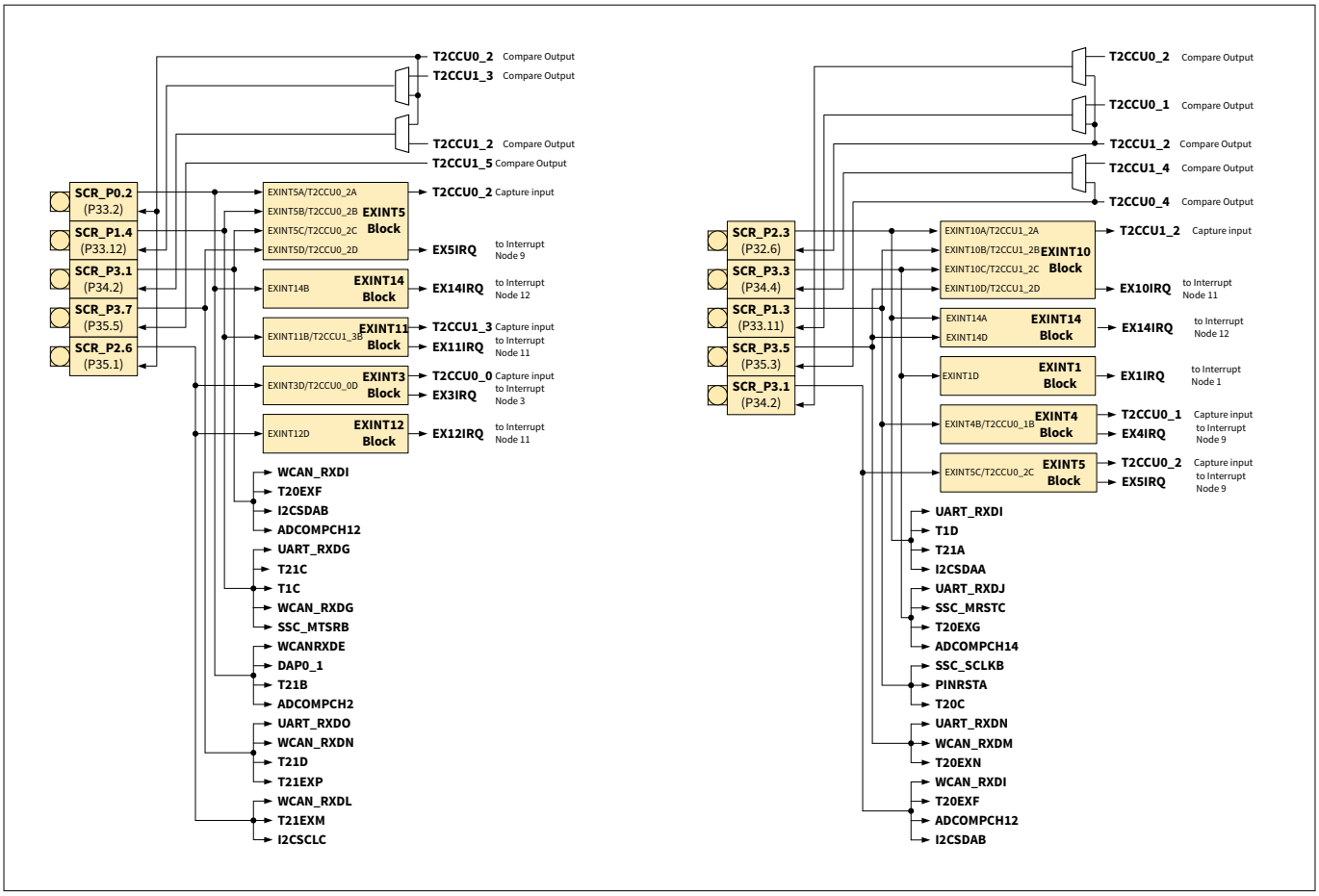


Figure 911 Interrupt and I/O Connections for Capture/Compare Channel 2

39.3.11.5.4 Capture/Compare Channel 3

The picture below illustrates (for both T2CCU0 and T2CCU1) the connections of capture/compare channel 3 to the ports, interrupt request blocks, and other SCR modules.

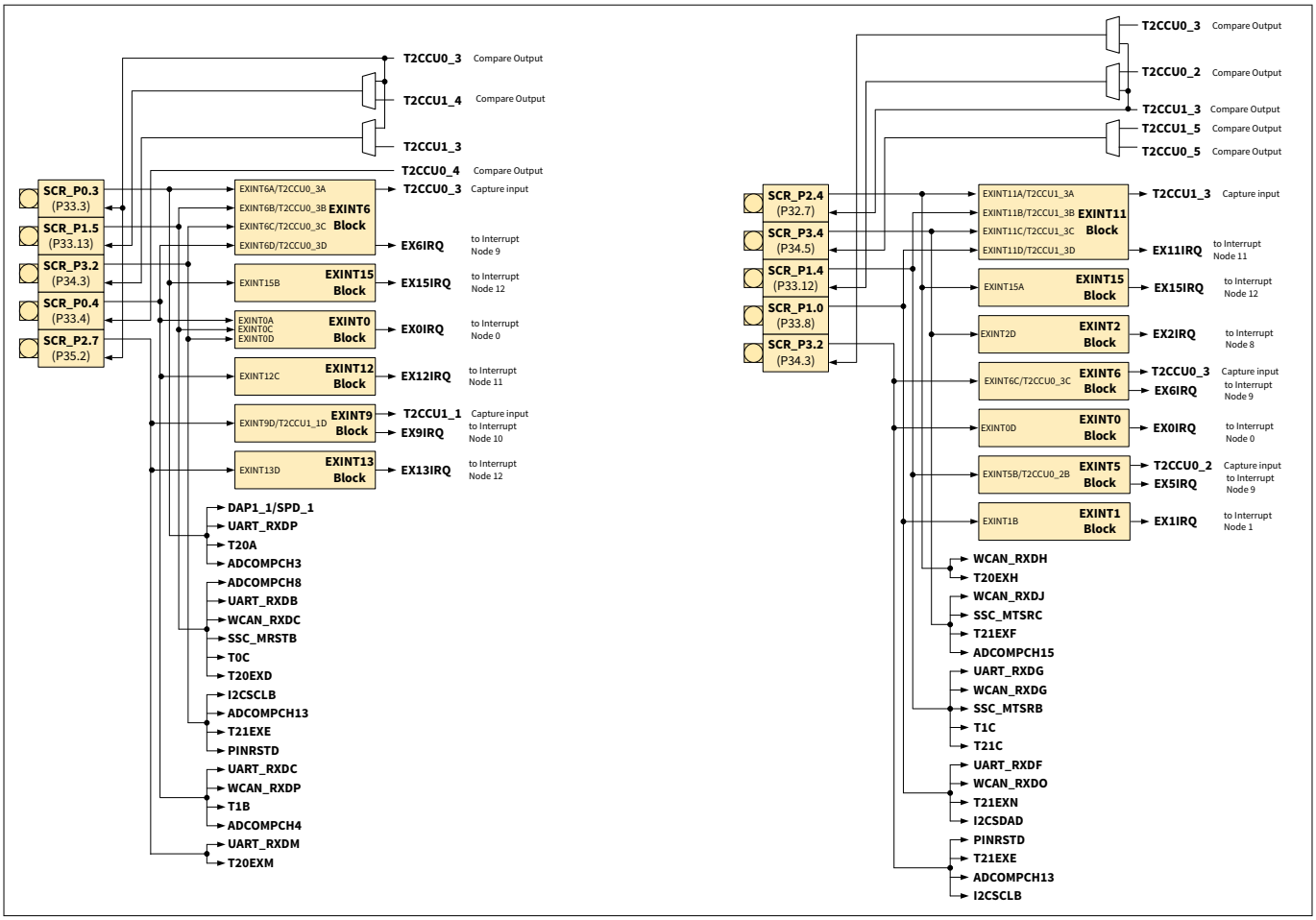


Figure 912 Interrupt and I/O Connections for Capture/Compare Channel 3

39.3.11.5.5 Compare Channel 4

The picture below illustrates (for both T2CCU0 and T2CCU1) the connections of compare channel 4 to the ports, interrupt request blocks, and other SCR modules.

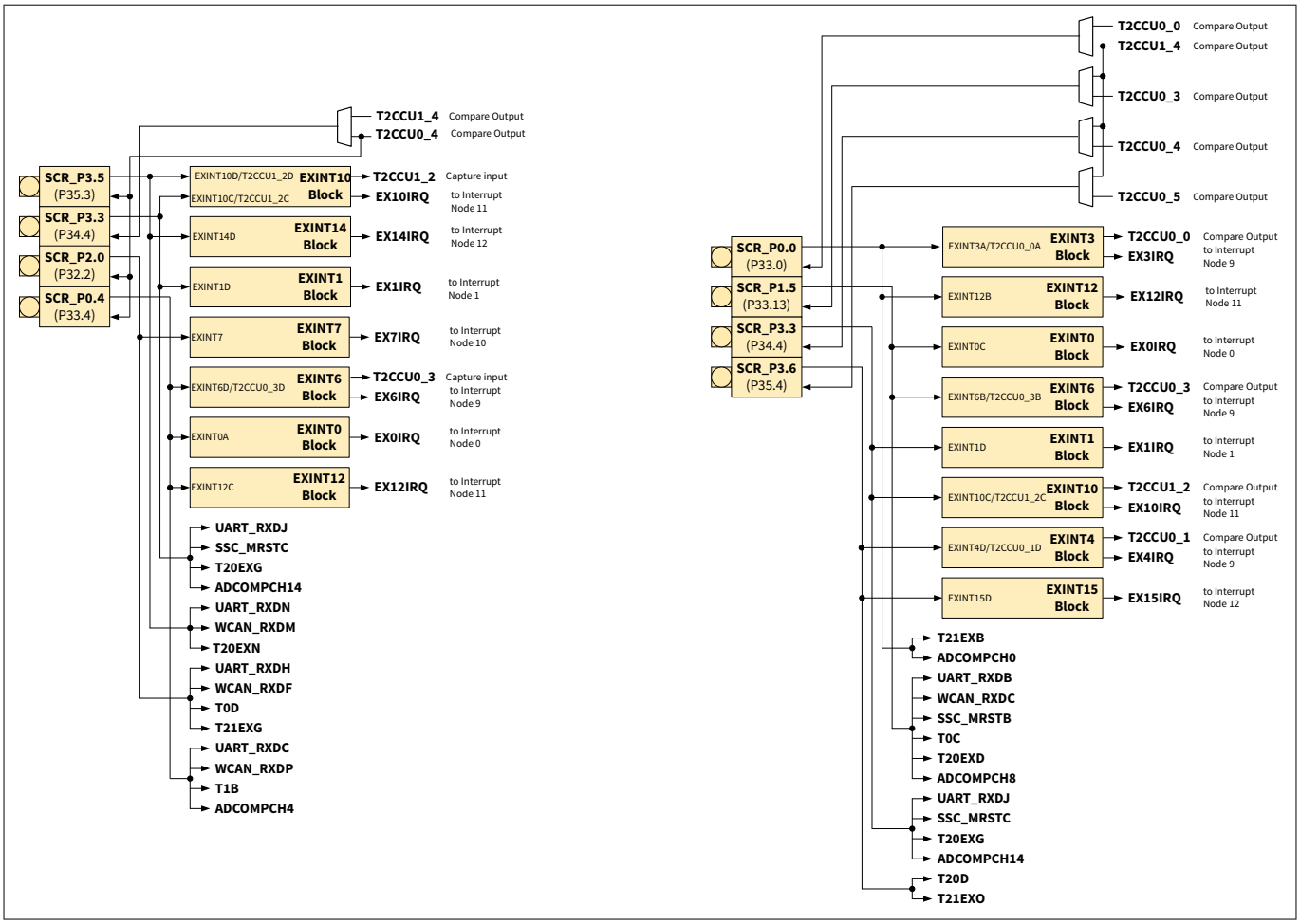


Figure 913 Interrupt and I/O Connections for Compare Channel 4

39.3.11.5.6 Compare Channel 5

The picture below illustrates (for both T2CCU0 and T2CCU1) the connections of compare channel 5 to the ports, interrupt request blocks, and other SCR modules.

39 Stand-by Controller (SCR)

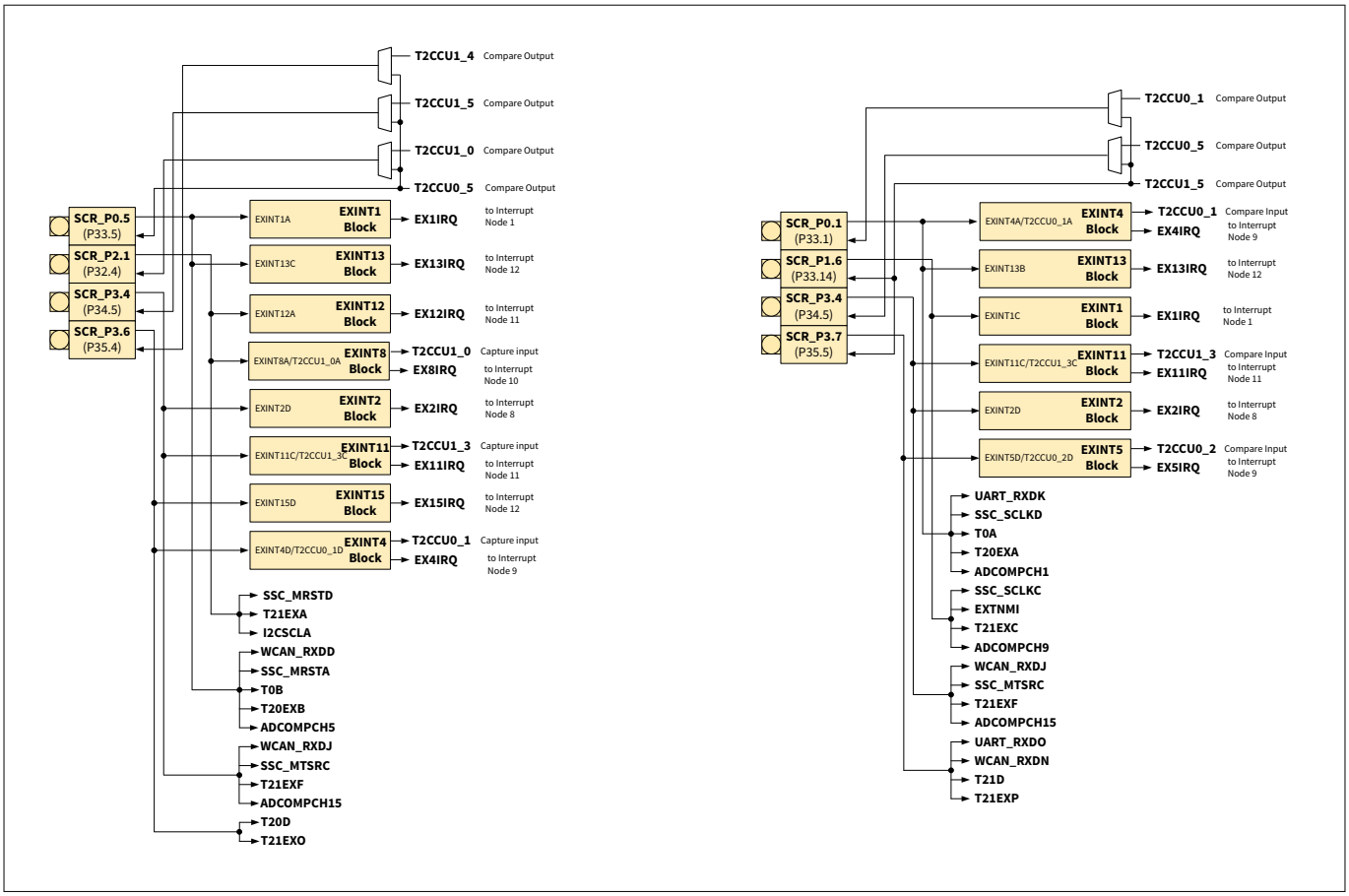


Figure 914 Interrupt and I/O Connections for Compare Channel 5

39.3.11.6 T2CCU clocking configuration

If the T2CCU0/1 functionality is not required at all, it can be completely disabled by gating off its clock input for maximal power reduction. This is done by setting bit SCUSCR_PMCON0.T2CCU0_DIS or SCUSCR_PMCON0.T2CCU1_DIS.

39.3.11.7 Interfaces of the T2CCU

The following figure gives an overview of the interconnections of the T2CCU.

The interrupt lines of the T2CCU0/1 are connected to the CPU interrupt controller.

The General Purpose IO (GPIO) Ports provide the interface from the T2CCU0/1 to the external world.

The T2CCU timers can be suspendedwhen OCDS enters Monitor Mode and has the Debug-Suspend signal activated, provided the respective timer suspend bit (SCUSCR_DBG_MODSUSP.T2CCUnSUSP) is set. Refer to the OCDS chapter.

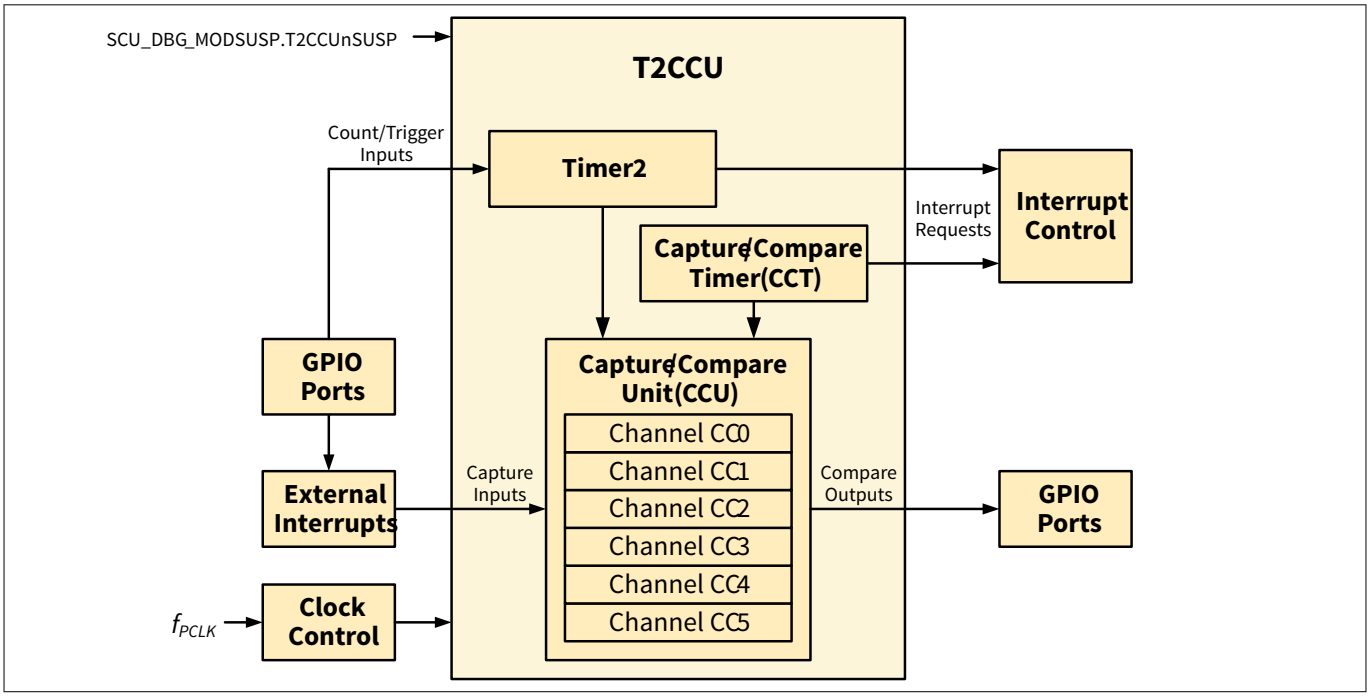


Figure 915 Interconnections of T2CCU

39.3.11.7.1 Interrupt events and assignment

The following table lists the interrupt event sources from the T2CCU, and the corresponding event interrupt enable bit and flag bit.

Table 1717 T2CCU interrupt events

Event	Event interrupt enable bit	Event flag bit
Timer 2 Overflow	T2_CON1.TF2EN	T2_CON.TF2
Timer 2 External Input	T2_CON1.EXF2EN	T2_CON.EXF2
CCT Timer Overflow	T2CCU_CCTCON.CCTOVEN	T2CCU_CCTCON.CCTOVF

Following tables show the interrupt node assignment for each T2CCU0/1 interrupt source.

Table 1718 T2CCU0 events' interrupt node control

Event	Interrupt node enable bit	Interrupt node flag bit	Vector address
Timer 2 Overflow	IEN0.ET2	–	$2B_H$
Timer 2 External Input			
CCT Timer Overflow			

Table 1719 T2CCU1 events' interrupt node control

Event	Interrupt node enable bit	Interrupt node flag bit	Vector address
Timer 2 Overflow	IEN1.ESSC	–	$3B_H$
Timer 2 External Input			
CCT Timer Overflow			

39.3.11.7.2 Module suspend control

When the On-Chip Debug Support (OCDS) is in Monitor Mode (DBG_MMCR2.MMODE=1) and the Debug-Suspend signal is active (DBG_MMCR2.DSUSP=1), the timers in the T2CCU module can be suspended based on the settings of their corresponding module suspend bits in register SCUSCR_DBG_MODSUSP. When suspended, only the timer stops counting, since the counter input clock is gated off. The module is still clocked, so that module registers are accessible.

39.3.12 Universal Asynchronous Receiver/Transmitter (UART)

39.3.12.1 Overview

The UART provides a full-duplex asynchronous receiver/transmitter, i.e., it can transmit and receive simultaneously. It is also receive-buffered, i.e., it can commence reception of a second byte before a previously received byte has been read from the receive register. However, if the first byte still has not been read by the time reception of the second byte is complete, one of the bytes will be lost.

***Note**: A module cycle of the UART consists of 2 PCLK cycles.*

***Note**: In this chapter, the input and output signals from/to the GPIO ports are referred to with their generic name, such as TXD and RXD. "RXD" assumes an input direction, "RXD_O" assumes an output direction, and "TXD" assumes an output direction.*

UART Features:

- Full-duplex asynchronous modes
  - 8-bit or 9-bit data frames, LSB first
  - fixed or variable baudrate
- Receive buffered
- Multiprocessor communication
- Interrupt generation on the completion of a data transmission or reception

39.3.12.2 UART modes

The UART can be used in four different modes. In mode 0, it operates as an 8-bit shift register. In mode 1, it operates as an 8-bit serial port. In modes 2 and 3, it operates as a 9-bit serial port. The only difference between mode 2 and mode 3 is the baud rate, which is fixed in mode 2 but variable in mode 3. The variable baudrate is set by the underflow rate on the dedicated baud rate generator.

The different modes are selected by setting bits UART_SCON.SM0 and UART_SCON.SM1, , to their corresponding values, as shown in following table:

Table 1720 UART modes

SM0	SM1	Operating mode	Baud rate
0	0	Mode 0: 8-bit shift register	$f_{PCLK}/2$
0	1	Mode 1: 8-bit shift UART	Variable
1	0	Mode 2: 9-bit shift UART	$f_{PCLK}/64$ or $f_{PCLK}/32$
1	1	Mode 3: 9-bit shift UART	Variable

39.3.12.2.1 Mode 0, 8-bit shift register, fixed baudrate

In mode 0, the serial port behaves as an 8-bit shift register. Data is shifted in through RXD, and out through RXD_O, while the TXD line is used to provide a shift clock which can be used by external devices to clock data in and out.

The transmission cycle is activated by a write to SBUF. One module cycle later, the data has been written to the transmit shift register with a 1 at the 9th bit position. For the next seven module cycles, the contents of the transmit shift register are shifted right one position and a zero shifted in from the left so that when the MSB of

the data byte is at the output position, it has a 1 and a sequence of zeros to its left. The control block then executes one last shift before setting the SCON.TI bit. If SCON.TI is set, the next data can be written to SBUF, without destroying the ongoing transmission. If SCON.TI is not set, the ongoing transmission maybe corrupted.

Reception is started by the condition REN=1 and SCON.RI=0. At the start of the reception cycle, 11111110_B is written to the receive shift register. In each module cycle that follows, the contents of the shift register are shifted left one position and the value sampled on the RXD line in the same module cycle is shifted in from the right. When the 0 of the initial byte reaches the leftmost position, the control block executes one last shift, loads SBUF and sets the SCON.RI bit.

The baudrate for the transfer is fixed at $f_{PCLK}/2$, where f_{PCLK} is the input clock frequency, that means one bit per module cycle.

***Note**: The bit SCON.RI is set at the beginning of the 8th bit.*

39.3.12.2.2 Mode 1, 8-bit UART, variable baudrate

In mode 1, the UART behaves as an 8-bit serial port. A start bit (0), 8 data bits, and a stop bit (1) are transmitted on TXD or received on RXD at a variable baudrate.

The transmission cycle is activated by a write to SBUF. The data is transferred to the transmit register and a 1 is loaded to the 9th bit position (as in mode 0). At phase 1 of the module cycle after the next rollover in the divide-by-16 counter, the start bit is copied to TXD, and data is activated one bit time later. One bit time after the data is activated, the data starts getting shifted right with zeros shifted in from the left. When the MSB gets to the output position, the control block executes one last shift and sets the SCON.TI bit.

Reception is started by a high to low transition on RXD (sampled at 16 times the baudrate). The divide-by-16 counter is then reset and 11111111_B is written to the receive register. If a valid start bit (0) is then detected (based on two out of three samples), it is shifted into the register followed by 8 data bits. If the transition is not followed by a valid start bit, the controller goes back to looking for a high to low transition on RXD. When the start bit reaches the leftmost position, the control block executes one last shift, then loads SBUF with the 8 data bits, loads RB8 (SCON.2) with the stop bit, and sets the SCON.RI bit, provided SCON.RI=0, and either SM2=0 (see related information) or the received stop bit=1. If none of these conditions is met, the received byte is lost.

The associated timings for transmit/receive in mode 1 are illustrated in the following figure.

***Note**: The bit SCON.RI is set at the end of the 8th bit.*

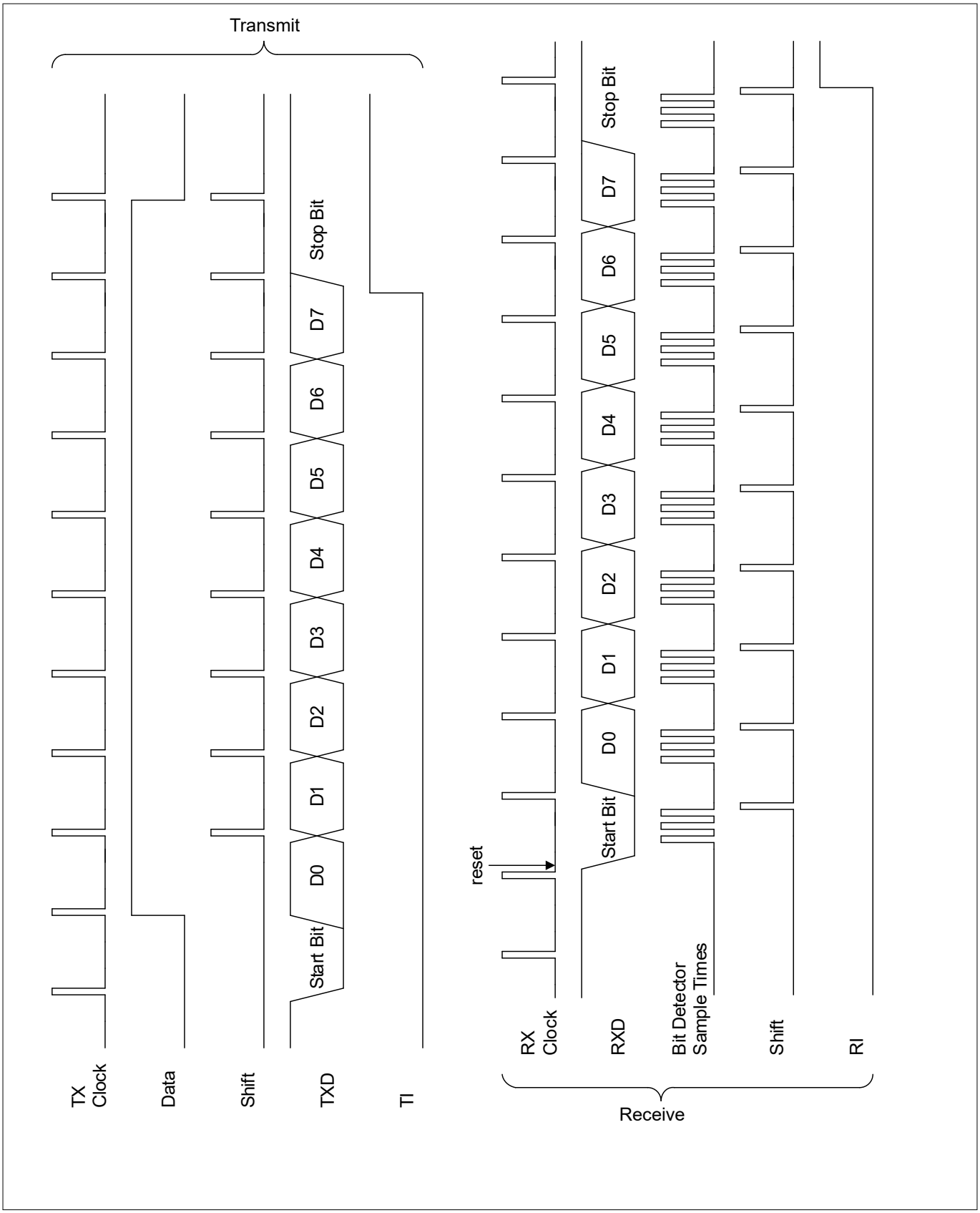


Figure 916 Serial interface, mode 1, timing diagram

Related information

Multiprocessor communication on page 6626

39.3.12.2.3 Mode 2, 9-bit UART, fixed baudrate

In mode 2, the UART behaves as a 9-bit serial port. A start bit (0), 8 data bits plus a programmable 9th bit and a stop bit (1) are transmitted on TXD or received on RXD. The 9th bit for transmission is taken from TB8 (SCON.3) while for reception, the 9th bit received is placed in RB8 (SCON.2).

The transmission cycle is activated by a write to SBUF. The data is transferred to the transmit register and TB8 is copied into the 9th bit position. At phase 1 of the module cycle following the next rollover in the divide-by-16 counter, the start bit is copied to TXD and data is activated one bit time later. One bit time after the data is activated, the data starts shifting right. For the first shift, a stop bit (1) is shifted in from the left and for subsequent shifts, zeros are shifted in. When the TB8 bit gets to the output position, the control block executes one last shift and sets the SCON.TI bit.

Reception is started by a high to low transition on RXD (sampled at 16 times the baudrate). The divide-by-16 counter is then reset and 11111111_B is written to the receive register. If a valid start bit (0) is then detected (based on two out of three samples), it is shifted into the register followed by 8 data bits. If the transition is not followed by a valid start bit, the controller goes back to looking for a high to low transition on RXD. When the start bit reaches the leftmost position, the control block executes one last shift, then loads SBUF with the 8 data bits, loads RB8 (SCON.2) with the 9th data bit, and sets the SCON.RI bit, provided SCON.RI=0, and either SM2=0 (see related information) or the 9th bit=1. If none of these conditions is met, the received byte is lost.

The baudrate for the transfer is either $f_{PCLK}/64$ or $f_{PCLK}/32$, depending on the setting of the top bit (PCON .SMOD) of the PCON (Power Control) register, which acts as a double baudrate selector.

***Note**: The bit SCON.RI is set at the end of the 9th bit.*

Related information

Multiprocessor communication on page 6626

39.3.12.2.4 Mode 3, 9-bit UART, variable baudrate

Mode 3 is the same as mode 2 in all respects except that the baudrate is variable.

In all modes, transmission is initiated by any instruction that uses SBUF as a destination register. A reception is initiated in the modes by the incoming start bit if REN=1.

The serial interface also provides interrupt requests when transmission or reception of the frames has been completed. The corresponding interrupt request flags are SCON.TI or SCON.RI, respectively. If the serial interrupt is not used (i.e., serial interrupt not enabled), SCON.TI and SCON.RI can also be used for polling the serial interface.

The associated timings for transmit/receive in modes 2 and 3 are illustrated in the following figure.

***Note**: The bit SCON.RI is set at the end of the 9th bit.*

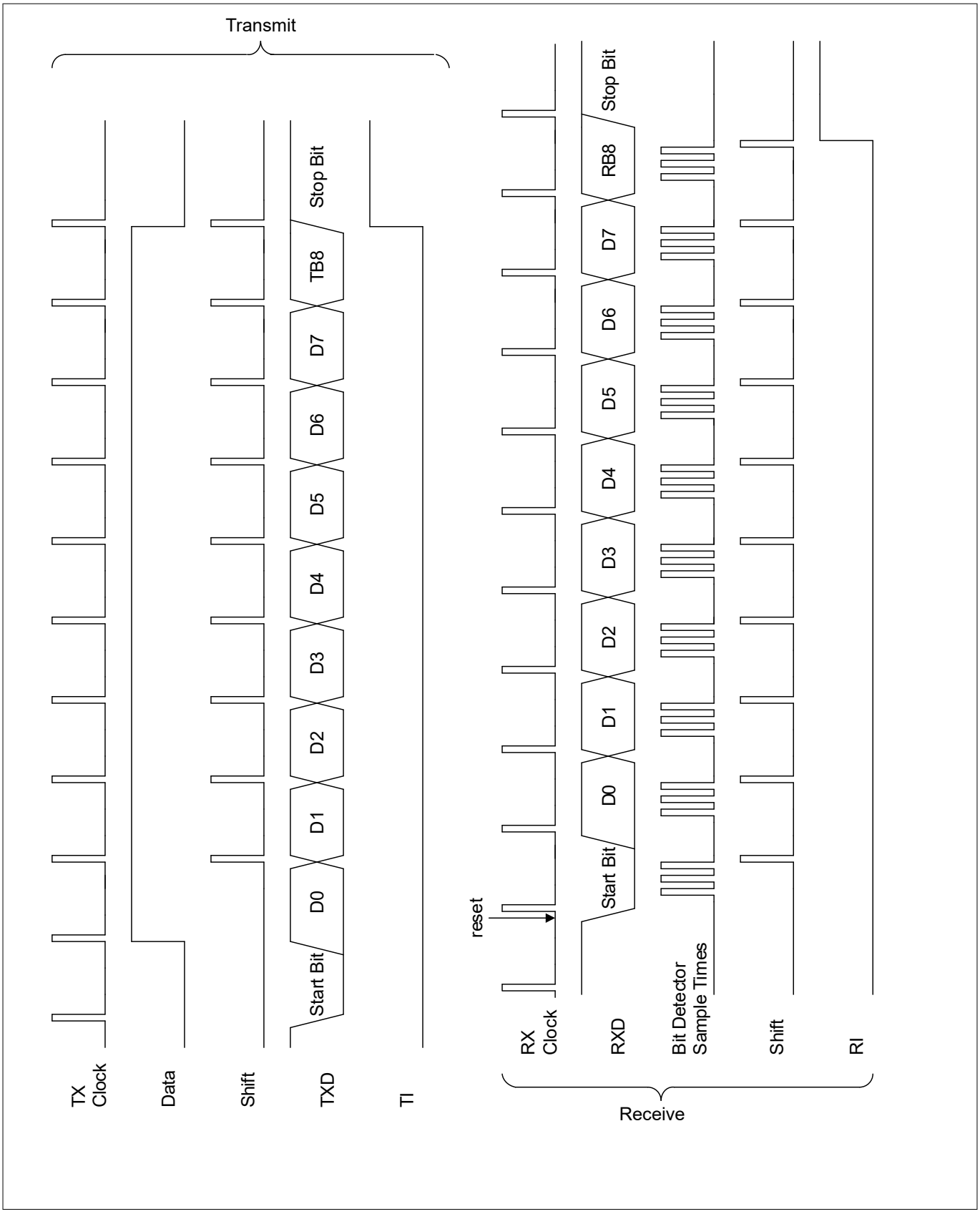


Figure 917 Serial interface, modes 2 and 3, timing diagram

39.3.12.3 Multiprocessor communication

Modes 2 and 3 have a special provision for multiprocessor communication using a system of address bytes with bit 9=1 and data bytes with bit 9=0. In these modes, 9data bits are received. The 9th data bit goes into RB8. The communication always ends with one stop bit. The port can be programmed such that when the stop bit is received, the serial port interrupt will be activated only if RB8=1.

This feature is enabled by setting bit SCON.SM2. One of the ways to use this feature in multiprocessor systems is described in the following paragraph.

When the master processor wants to transmit a block of data to one of several slaves, it first sends out an address byte that identifies the target slave. An address byte differs from a data byte in that the 9th bit is 1 in an address byte and 0 in a data byte. With SCON.SM2=1, no slave will be interrupted by a data byte. An address byte, however, will interrupt all slaves, so that each slave can examine the received byte and see if it is being addressed. The addressed slave will clear its SCON.SM2 bit and prepare to receive the data bytes that will be coming. The slaves that were not being addressed retain their SCON.SM2s as set and ignore the incoming data bytes.

Bit SCON.SM2 has no effect in mode 0. SCON.SM2 can be used in mode 1 to check the validity of the stop bit. In a mode 1 reception, if SCON.SM2=1, the receive interrupt will not be activated unless a valid stop bit is received.

39.3.12.4 Baudrate generation

There are several ways to generate the baudrate clock for the serial ports, depending on the mode in which they are operating.

The baudrates in modes 0 and 2 are fixed, so they use the fixed clock, (see Fixed Clock in related information). In modes 1 and 3, the variable baudrate is generated using the UART baudrate generator (see UART Baudrate Generator in related information).

“Baudrate clock” and “baudrate” must be distinguished from each other. The serial interface requires a clock rate that is 16 times the baudrate, to accommodate for the 16-times oversampling on reception. Therefore, the UART baudrate generator must provide a “baudrate clock” to the serial interface, where it is divided by 16 to obtain the actual “baudrate”. The abbreviation f_{PCLK} refers to the input clock frequency.

Related information

Fixed clock on page 6626

UART baudrate generator on page 6627

39.3.12.4.1 Fixed clock

The baudrates in modes 0 and 2 are fixed. However, while the baudrate in mode 0 can only be $f_{PCLK}/2$, the baudrate in mode 2 can be selected as either $f_{PCLK}/64$ or $f_{PCLK}/32$ depending on bit PCON.SMOD.

Bit PCON.SMOD acts as a double baudrate selector, as shown in the following figure. If PCON.SMOD=0 (value after reset), the baudrate is 1/64 of the input clock frequency f_{PCLK}. If PCON.SMOD=1, the baudrate is 1/32 of f_{PCLK}.

$$Mode\ 2\ Baudrate = \frac{2^{SMOD}}{64} \times f_{PCLK} \tag{63}$$

Mode 2 baudrate

39.3.12.4.2 UART baudrate generator

The UART baudrate generator is used to generate the variable baudrate for the UART in modes 1 and 3. It has programmable 11-bit reload value, 3-bit pre-scaler, and 5-bit fractional divider. It is configured through bit-fields stored in registers: LIN_BCON, LIN_BGL, LIN_BGH.

The baudrate generator is clocked derived through a pre-scaler (f_{DIV}) from the input clock f_{PCLK}. The baudrate timer counts downwards and can be started or stopped through the baudrate control run bit LIN_BCON.R. Each underflow of the timer provides one clock pulse to the serial channel. The timer is reloaded with the 11-bit BR_VALUE stored in its reload register LIN_BGH/LIN_BGL each time it underflows. The duration between underflows depends on the 'n' value in the fractional divider, which can be selected by the bits LIN_BGL.FD_SEL. 'n' times out of 32, the timer counts one cycle more than specified by BR_VALUE. The pre-scaler is selected by the bits LIN_BCON.BRPRE.

Register LIN_BGH/LIN_BGL is the dual-function baudrate Generator/Reload register. Reading LIN_BGH/LIN_BGL returns the contents of the timer, while writing to LIN_BGH/LIN_BGL always updates the reload register.

Registers LIN_BGH/LIN_BGL may be written only when LIN_BCON.R is 0. An auto-reload of the timer with the contents of the reload register is performed one instruction cycle after the next time LIN_BCON.R is set. Any write to LIN_BGH/LIN_BGL, while LIN_BCON.R is set, is not allowed.

The baudrate of the baudrate generator depends on the following bits and register values:

- Input clock f_{PCLK}
- Value of bit-field LIN_BCON.BRPRE
- Value of bit-field BG.FD_SEL
- Value of the 11-bit reload value BR_VALUE in registers LIN_BGH/LIN_BGL

The following figure shows a simplified block diagram of the baudrate generator. Note that f_{BR} is 16 times the configured baudrate, due to synchronization and oversampling needs.

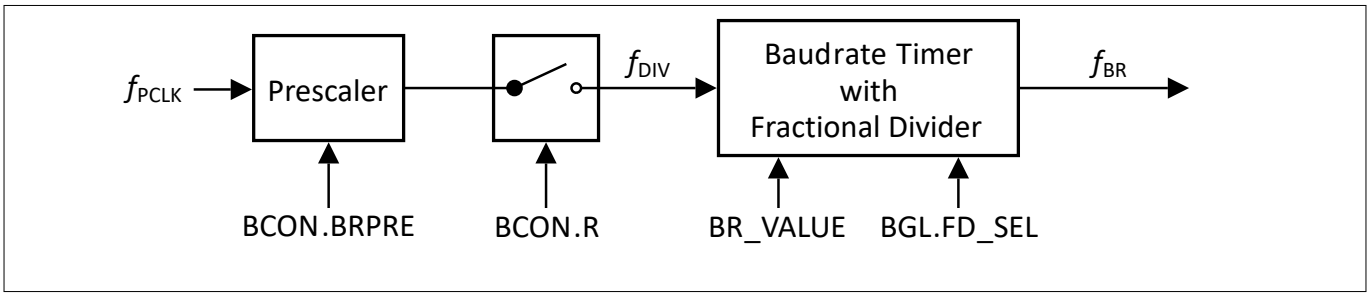


Figure 918 Simplified baudrate generator block diagram

The following equation calculates the final baudrate.

$$Baudrate = \frac{f_{BR}}{16} = \frac{f_{PCLK}}{16 \times \mathrm{Pr}\,escaler \times \left(BR_VALUE + \frac{n}{32}\right)} = \frac{f_{PCLK}}{16 \times \left(2^{BCON.BRPRE}\right) \times \left(BR_VALUE + \left(\frac{BGL.FD_SEL}{32}\right)\right)} \quad (64)$$

Baudrate calculation

The value of the pre-scaler is chosen by the bit-field LIN_BCON.BRPRE. BR_VALUE represents the contents of the reload value, taken as unsigned 11-bit integer from the two bit fields BR_VALUE in registers LIN_BGH and LIN_BGL. n/32 is defined by the fractional divider selection in bit-field LIN_BGL.FD_SEL.

Note that the maximum baudrate that can be generated in this implementation is limited to f_{PCLK}/48. For a module clock of 20 MHz, the maximum achievable baudrate is 416.67 kBaud.

The following table lists various commonly used baudrates together with their corresponding parameter settings and the deviation errors compared to the intended baudrate.

Table 1721 Typical baudrates of UART, example f_{PCLK} = 20 MHz

Baudrate (f_{PCLK} = 20 MHz)	BCON.BRPRE	Reload value (BR_VALUE)	Numerator of fractional value (BGL.FD_SEL)	BG value[1]	Deviation error
115.2 kBaud	000_B (pre-scaler = 1)	$0A_H$	$1B_H$	$015B_H$	+0.06%
20 kBaud	000_B (pre-scaler = 1)	$3E_H$	10_H	$07D0_H$	+0.00%
19.2 kBaud	000_B (pre-scaler = 1)	41_H	03_H	0823_H	+0.02%
9600 Baud	001_B (pre-scaler = 2)	65_H	03_H	0823_H	+0.02%
4800 Baud	010_B (pre-scaler = 4)	41_H	03_H	0823_H	+0.02%
2400 Baud	011_B (pre-scaler = 8)	41_H	03_H	0823_H	+0.02%

1) The BG value is obtained by concatenating the 11-bit BR_VALUE and 5-bit FD_SEL value to a 16-bit value, which represents the contents of the register pair BGH|BGL.

39.3.12.5 UART interrupt requests

The following figure illustrates the UART receive and transmit interrupt requests. Note that the SCON.RI/ SCON.TI flags are not automatically cleared by the CPU on entry into the respective service routine. They must be cleared by software. See also the section on “Interrupt Structure 1” in chapter “Interrupt System”.

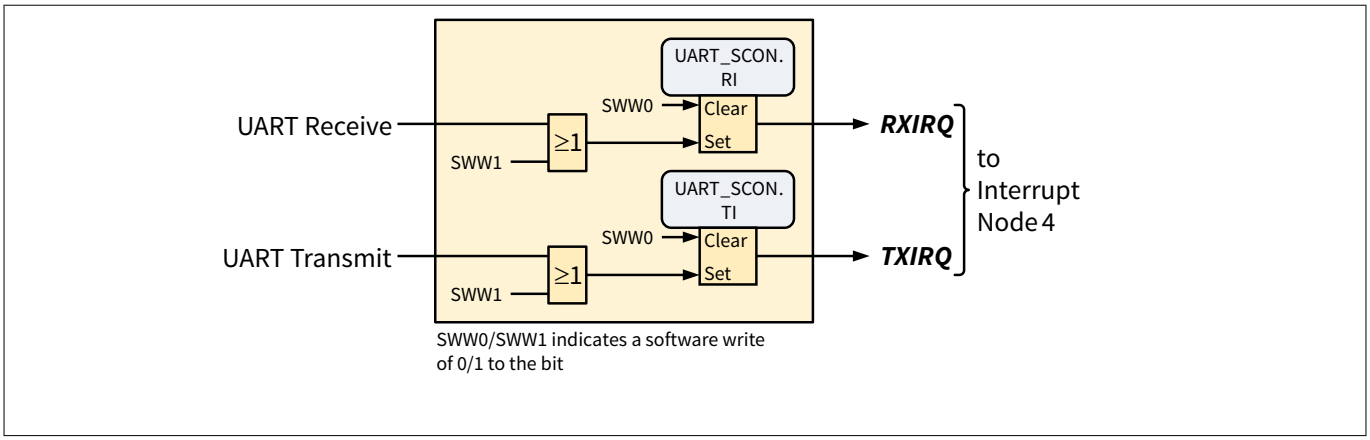


Figure 919 UART receive and transmit interrupt requests

Related information

Interrupt structure on page 6521

39.3.12.6 LIN support in UART

The UART module can be used to support the Local Interconnect Network (LIN) protocol for both master and slave operations. The LIN baudrate detection feature, which consists of the hardware logic for Break and Synch Field detection, provides the capability to detect the baudrate within LIN protocol using one of the available Timer 2 units. This allows the UART module to be synchronized to the LIN baudrate for data transmission and reception. Timer 2 will be used for BREAK detection and generation.

***Note**: Both T2CCU0 Timer 2 and T2CCU1 Timer 2 could be used for this feature. Which one of the two will be used depends on selected input pins - requirement is that UART/LIN RXD input is on the same pin as Timer 2 EX input (T20EX or T21EX).*

39.3.12.6.1 LIN protocol

LIN is a holistic communication concept for local interconnected networks in vehicles. The communication is based on the SCI (UART) data format, a single-master/multiple-slave concept, a clock synchronization for nodes without stabilized time base. An attractive feature of LIN is self-synchronization of the slave nodes without a crystal or ceramic resonator, which significantly reduces the cost of hardware platform. The baudrate must be calculated and returned with every message frame.

The structure of a LIN frame is shown in the following figure. The frame consists of the:

- Header, which comprises a Break (13-bit time low), Synch Byte (55_B), and ID field
- Response time
- Data bytes (according to UART protocol)
- Checksum

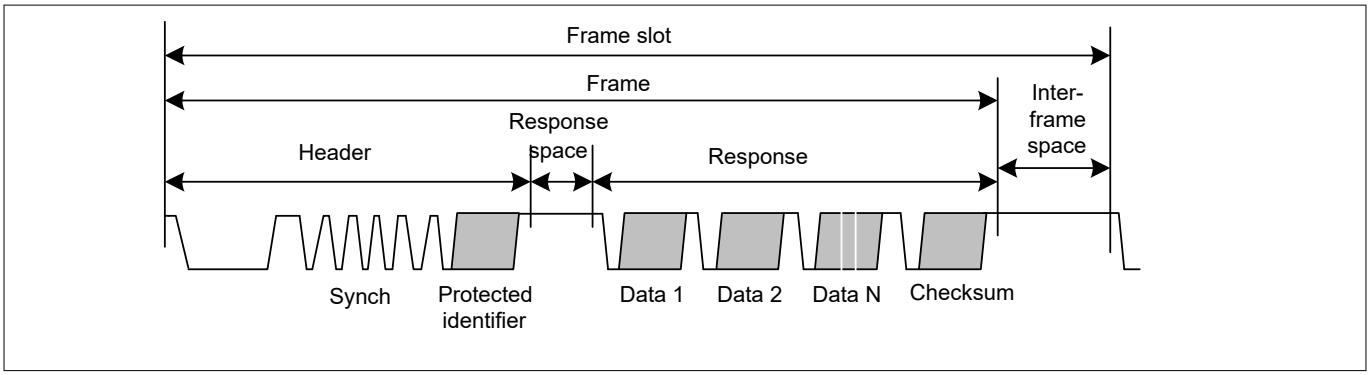


Figure 920 The structure of LIN frame

Each byte field is transmitted as a serial byte, as shown in the following figure. The LSB of the data is sent first and the MSB is sent last. The start bit is encoded as a bit with value zero (dominant) and the stop bit is encoded as a bit with value one (recessive).

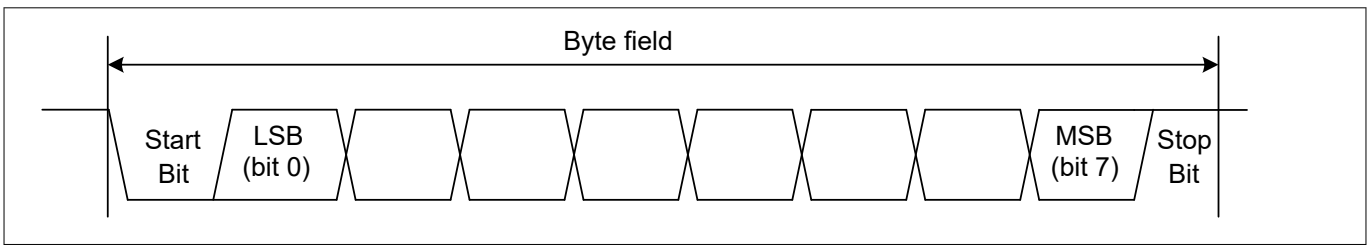


Figure 921 The structure of byte field

The break is used to signal the beginning of a new frame. It is the only field that does not comply with the previous figure. A break is always generated by the master task (in the master mode) and it must be at least

13 bits of dominant value, including the start bit, followed by a break delimiter, as shown in the following figure. The maximum length is 26.6 bit times. The break delimiter will be at least one nominal bit time long.

A slave node will use a break detection threshold of 11 nominal bit times.

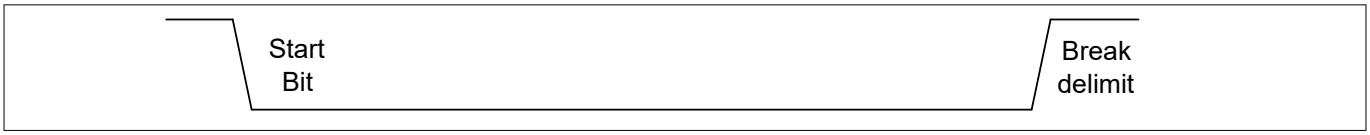


Figure 922 The break field

Synch Byte is a specific pattern for determination of time base. The byte field is with the data value 55_H, as shown in the following figure.

A slave task is always able to detect the Break/Synch sequence, even if it expects a byte field (assuming the byte fields are separated from each other). If this happens, detection of the Break/Synch sequence will abort the transfer in progress and processing of the new frame will commence.

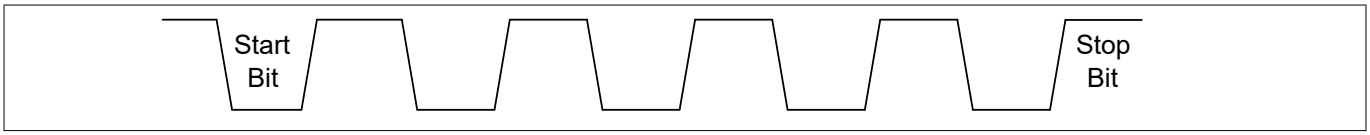


Figure 923 The synch byte field

The slave task will receive and transmit data when an appropriate ID is sent by the master:

1. Slave waits for Synch Break
2. Slave synchronizes on Synch Byte
3. Slave snoops for ID
4. According to ID, slave determines whether to receive or transmit data or to do nothing
5. When transmitting, the slave sends 2, 4 or 8 data bytes, followed by check byte

39.3.12.6.2 LIN header transmission

LIN header transmission is only applicable in master mode. In LIN communication, a master task decides when and which frame is to be transferred on the bus. It also identifies a slave task to provide the data transported by each frame. The information needed for the handshaking between the master and slave tasks is provided by the master task through the header portion of the frame.

The header consists of a "break and synch" pattern followed by an identifier. Among these three fields, only the break pattern cannot be transmitted as normal 8-bit UART data. The break must contain a dominant value of 13 bits or more to ensure proper synchronization of slave nodes.

In LIN communication, a slave task is required to be synchronized at the beginning of the protected identifier field of frame. For this purpose, every frame starts with a sequence consisting of a break field followed by a "synch" byte field. This sequence is unique and provides enough information for any slave task to detect the beginning of a new frame and be synchronized at the start of the identifier field.

39.3.12.6.3 Automatic synchronization to the host

Upon entering LIN communication, a connection is established and the transfer speed (baudrate) of the serial communication partner (host) is automatically synchronized in the following steps that are to be included in user software:

STEP 1: Initialize interface for reception and timer for baudrate measurement

STEP 2: Wait for an incoming LIN frame from host

STEP 3: Synchronize the baudrate to the host

STEP 4: Enter for Master Request Frame or for Slave Response Frame

The related section provides some hints on setting up the microcontroller for baudrate detection of LIN.

***Note**: Re-synchronization and setup of baudrate are always done for every Master Request Header or Slave Response Header LIN frame.*

Related information

Initialization of break/synch field detection logic on page 6631

39.3.12.6.4 Initialization of break/synch field detection logic

***Note**: Both T2CCU0 and T2CCU1 can be used to implement this feature. Which one of the two will be used depends on selected input pins - the requirement is that a UART/LIN RXD input is on the same pin as a Timer 2 EX input (T20EX or T21EX). In the text below it is only referred to as "Timer 2" without index 0 or 1.*

The LIN baudrate detection feature provides measures to detect the baudrate within the LIN protocol using Timer 2 (T2CCU) and UART, within the LIN user software. Initialization consists of:

- Serial port of the microcontroller to be configured to Mode 1 (8-bit UART, variable baudrate) for communication
- Provide the baudrate range by configuring the bit-field LIN_BCON.BGSEL
- Write LIN_BCON.BRDIS = 1
- Write LIN_BCON.BRDIS = 0 to initialize the Break / synch detection logic
- Clear all status flags LIN_STAT.BRK, LIN_STAT.EOFSYN and LIN_STAT.ERRSYN to 0
- Configure Timer 2 to capture mode with falling edge trigger input T2xEX (x can be 0 or 1). Bit T2CCUn_T2_MOD.EDGESEL is set to 0 by default, and bit T2CCUn_T2_CON.CP_RL2 is set to 1
- Enable external events for Timer 2 by setting T2CCUn_T2_CON. EXEN2 to 1 (EXF2 flag is set when a negative transition occurs at input T2xEX)

The baudrate calculation using the T2 result can be done with the following:

- The UART baudrate is given by the previous Equation 2
- The baudrate measured by the T2 is:

$$BRmeasured\ T2 = \frac{f_{PCLK} \times N_{measured\ bits}}{T2\ prescaler \times T2\ capture\ value} \quad (65)$$

- With the assumption T2 pre-scaler = UART pre-scaler, we can say:

$$BR_VALUE + \frac{BGL.FD_SEL}{32} = \frac{T2\ capture\ value}{16 \times N_{measured\ bits}} \quad (66)$$

With following bits and register values:

- Input clock f_{PCLK}
- Number of measured bits Nmeasured bits
- Value of bit-field T2 capture value, 16 bit value in registers T2CCUn_T2_RC2H/T2CCUn_T2_RC2L
- Value of bit-field LIN_BGL.FD_SEL
- Value of the UART pre-scaler is configured in LIN_BCON.BRPRE

- Value of the T2 pre-scaler is configured in T2CCUn_T2_MOD.T2PRE
- Value of the 11-bit reload value BR_VALUE in registers LIN_BGH/LIN_BGL

***Note**: f_{T2} can be configured by bit-field T2CCUn_T2_MOD.T2PRE*

***Note**: The baudrate must be calculated and returned with every message frame.*

***Note**: It is also recommended to toggle the LIN_BCON.BRDIS bit after the reception of each LIN frame (complete or erroneous frame) to avoid a wrong Break field detection in noisy environments (i.e. spikes on the LIN bus).*

***Note**: The baudrate is the number of times the signal can switch states in one second. To calculate the actual byte transfer rate simply divide the baudrate by the number of bits that must be transferred for each byte of data.*

39.3.12.6.5 Break detection

The Break/Synch Field detection logic supports a maximum number of bits in the Break field as defined by the conformance test 26.6 bits.

39.3.12.6.6 Baudrate range selection

$$Maximum\ number\ of\ bits = Baudrate \times \frac{7262}{SampleFrequency} \tag{67}$$

Maximum number of bits in break field

$$Sample\ frequency = \frac{f_{PCLK}}{8 \times 2^{BGSEL}} \tag{68}$$

Sample frequency equation

If the maximum number of bits in the Break field is exceeded, the internal counter will overflow, which results in a baudrate detection error. Therefore, an appropriate LIN_BCON.BGSEL value has to be selected for the required baudrate detection range.

The baudrate range defined by different LIN_BCON.BGSEL settings is shown in the next table. Each LIN_BCON.BGSEL setting supports a range of baudrate for detection. If the baudrate used is outside the defined range, the baudrate may not be detected correctly.

Table 1722 BGSEL bit-field definition for different input frequencies

f_{PCLK}	BGSEL	Baudrate selection for detection $f_{PCLK}/(2184*2^{BGSEL})$ to $f_{PCLK}/(74*2^{BGSEL})$
20 MHz	00_B	9.2 kHz to 270.2 kHz
	01_B	4.6 kHz to 135.1 kHz
	10_B	2.3 kHz to 67.5 kHz
	11_B	1.1 kHz to 33.7 kHz

(table continues...)

Table 1722 (continued) BGSEL bit-field definition for different input frequencies

f_{PCLK}	BGSEL	Baudrate selection for detection $f_{PCLK}/(2184*2^{BGSEL})$ to $f_{PCLK}/(74*2^{BGSEL})$
100 MHz	00_B	45.8 kHz to 1451.3 kHz
	01_B	22.9 kHz to 675.6 kHz
	10_B	11.5 kHz to 337.8 kHz
	11_B	5.8 kHz to 168.9 kHz

If f_{PCLK}=20 MHz, the baudrate range between 1.1 kHz to 270.2 kHz can be detected. The following examples serve as a guide to select the BGSEL value:

- If the baudrate falls in the range of 1.2 kHz to 2.3 kHz, the selected BGSEL value is 11_B
- If the baudrate falls in the range of 2.3 kHz to 4.6 kHz, the selected BGSEL value is 10_B
- If the baudrate falls in the range of 4.6 kHz to 9.2 kHz, the selected BGSEL value is 01_B
- If the baudrate falls in the range of 9.2 kHz to 270.2 kHz, the selected BGSEL value is 00_B. If the baudrate is 20 kHz, the possible values of BGSEL that can be selected are 00_B, 01_B, 10_B, and 11_B. However, it is advisable to select 00_B for better detection accuracy

For PCLK at 100 MHz, BGSEL can be chosen at 11_B.

39.3.12.6.7 LIN baudrate detection

The configuration has to take place in software. The baudrate detection for LIN is shown in the following figure, the Header LIN frame consists of:

- SYN(CH) Break (13-bit times low)
- SYN(CH) byte (55_H)
- Protected ID field

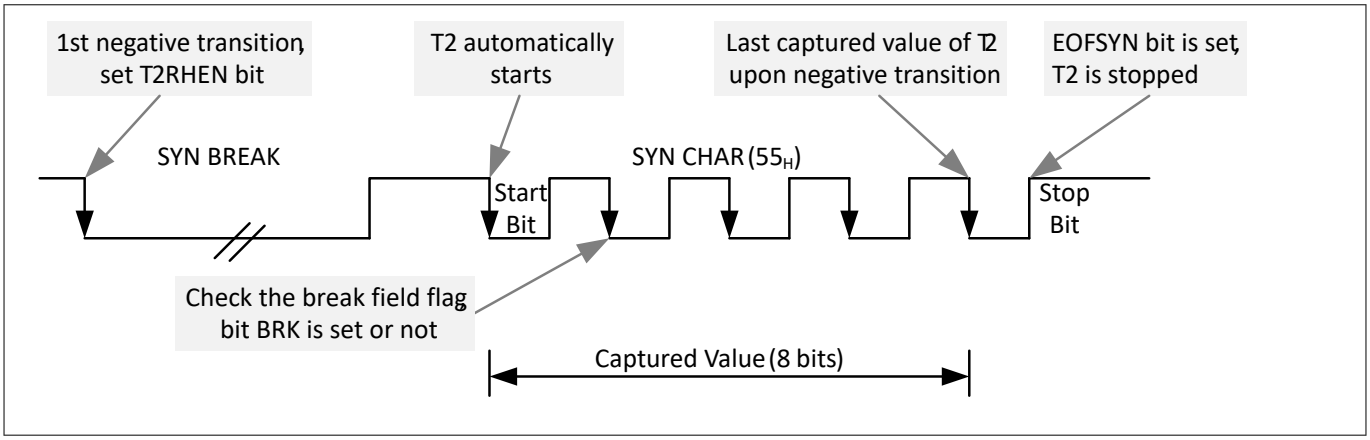


Figure 924 LIN baudrate detection

***Note**: Both T2CCU0 Timer 2 and T2CCU1 Timer 2 could be used for this feature. Which one of the two will be used depends on selected input pins - requirement is that UART/LIN RXD input is on the same pin as Timer 2 EX input (T20EX or T21EX). In the text below it is only referred to as "Timer 2" without index 0 or 1.*

With the first falling edge:

- The Timer 2 External Start Enable bit (T2CCU0_T2_MOD.T2RHEN) is set. The falling edge at input T2EX is selected by default for Timer 2 External Start (bit T2CCU0_T2_MOD.T2REGS is 0)

With the second falling edge:

- Start Timer 2 by the hardware

With the third falling edge:

- Timer 2 captures the timing of 2 bits of SYN byte
- Check the Break Field Flag bit LIN_STAT.BRK

If the Break Field Flag LIN_STAT.BRK is set, software may continue to capture 4/6/8 bits of SYN byte. The End of SYN Byte Flag (LIN_STAT.EOFSYN) is set, Timer 2 is stopped. T2 Reload/Capture register (T2CCUn_T2_RC2H/ T2CCUn_T2_RC2L) is the time taken for 2/4/6/8 bits according to the implementation. Then the LIN user software routine calculates the actual baudrate, sets the LIN_BCON.BRPRE and LIN_BGH/LIN_BGL values if the UART module uses the baudrate generator for baudrate generation.

After the third falling edge, the software may discard the current operation and continue to detect the next header LIN frame if the following conditions were detected:

- The Break Field Flag LIN_STAT.BRK is not set, or
- The SYN Byte Error Flag LIN_STAT.ERRSYN is set, or
- The Break Field Flag LIN_STAT.BRK is set, but the end of SYN Byte Flag LIN_STAT.EOFSYN and the SYN Byte Error Flag LIN_STAT.ERRSYN is not set

39.3.12.6.8 LIN interrupt requests

The following figure illustrates the LIN interrupt requests. Note that the flags must be cleared by software. See also the section on “Interrupt Structure 2” in chapter “Interrupt System”.

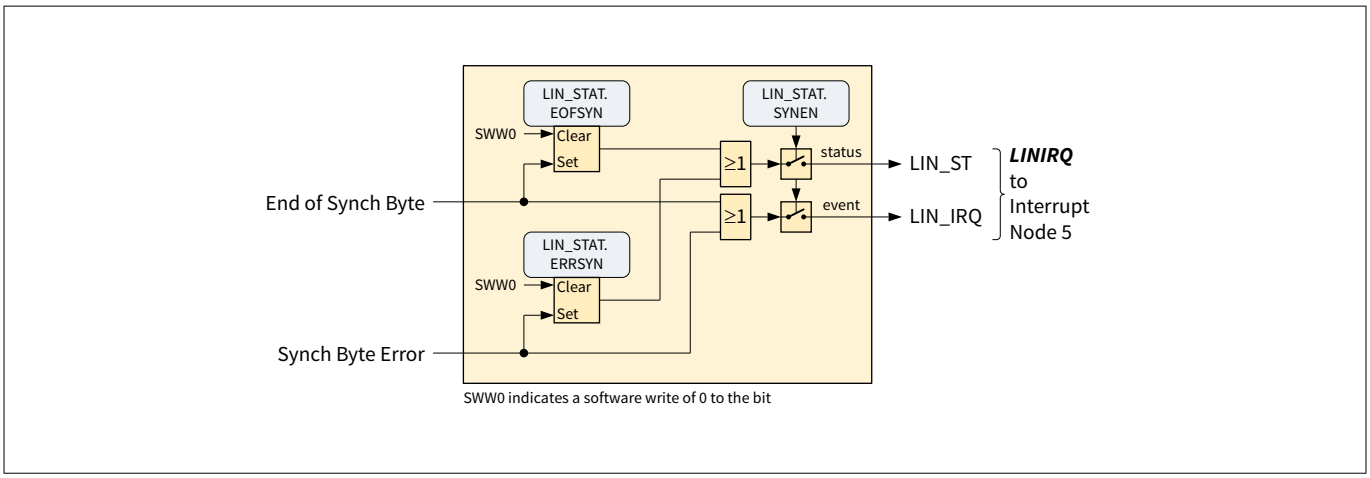


Figure 925 LIN interrupt requests

39.3.12.7 UART connections to GPIO

The following figure illustrates the signal connections of the UART to the GPIO ports.

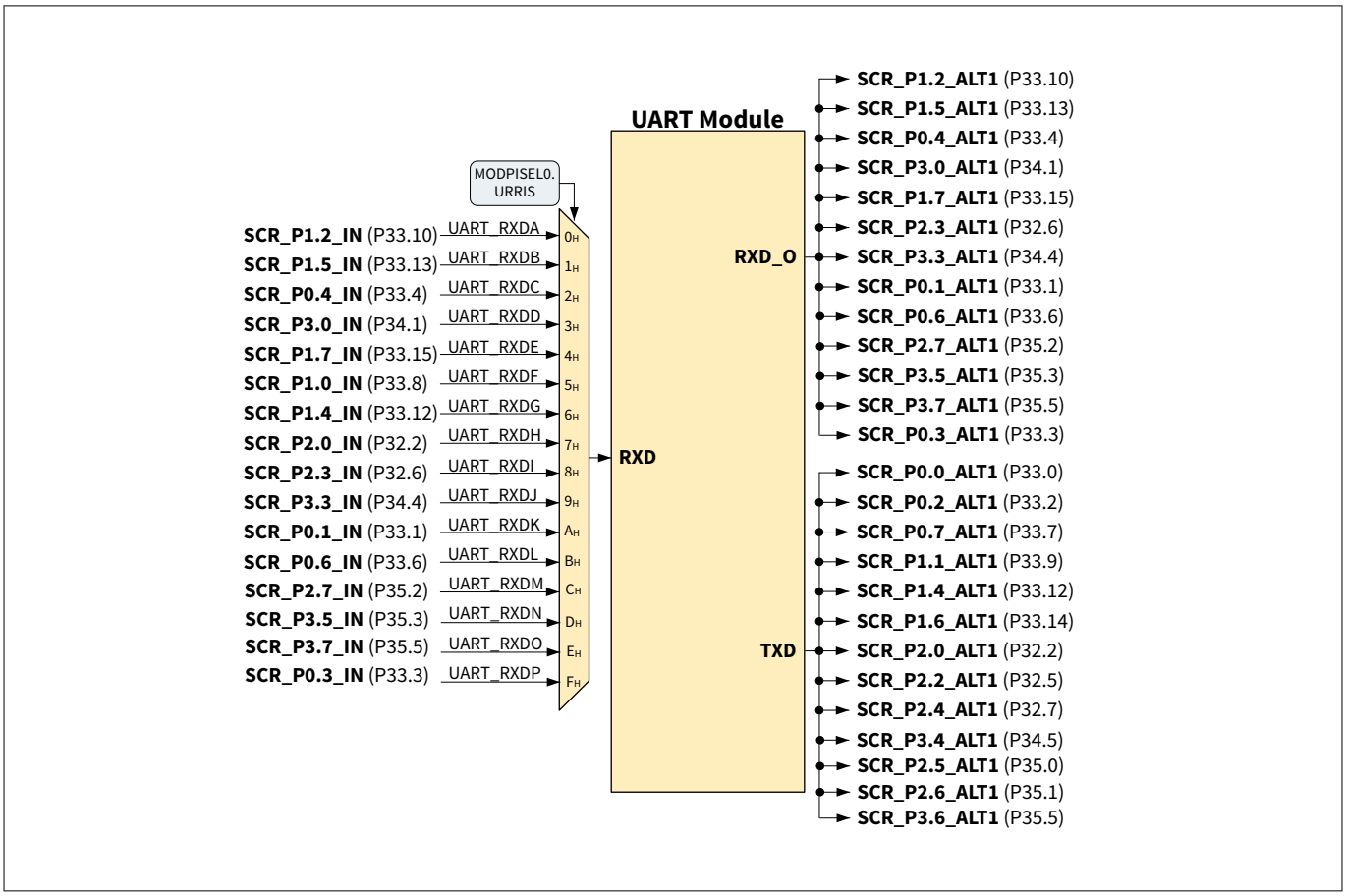


Figure 926 UART connections to GPIO ports

39.3.12.8 Register overview

The UART/LIN registers can be accessed from both the standard (non-mapped) and mapped SFR area.

UART contains the two Special Function Registers (SFRs), UART_SCON and UART_SBUF. UART_SCON is the control register and UART_SBUF is the data register. On reset, both UART_SCON and UART_SBUF return 00_H. The serial port control and status register is the UART_SCON . This register contains not only the mode selection bits, but also the 9th data bit for transmit and receive (TB8 and RB8) and the serial port interrupt bits (SCON.TI and SCON.RI).

UART_SBUF is the receive and transmit buffer of the serial interface. Writing to UART_SBUF loads the transmit register and initiates transmission. This register is used for both transmit and receive data. Transmit data is written to this location and receive data is read from this location, but the two paths are independent.

Reading out UART_SBUF accesses a physically separate receive register.

Other registers are prefixed with "LIN" even though they are used for both UART and LIN functionality: LIN_BCON, LIN_BGL/BGH, LIN_STAT.

The low (LIN_BGL) and high byte (LIN_BGH) of the baudrate timer/reload register BG (consisting out of high and low byte) contain the 11-bit reload value for the Baudrate timer and the 5-bit fractional divider selection. Reading the low byte of BG LIN_BGL returns the content of the lower three bits of the baudrate timer and the FD_SEL setting, while reading the high byte LIN_BGH returns the content of the upper 8 bits of the baudrate timer.

Writing to register BG loads the baudrate timer with the reload and fractional divider values from the BG registers, the first instruction cycle after LIN_BCON.R is set.

BG may only be written if BCON.R = 0.

39.3.13 Synchronous Serial Channel (SSC)

This chapter describes the Synchronous Serial Channel (SSC).

39.3.13.1 Overview

The Synchronous Serial Channel (SSC) supports both full-duplex and half-duplex serial synchronous communication. The serial clock signal can be generated by the SSC itself (Master Mode) through its own 16-bit baudrate generator, or can be received from an external master (Slave Mode). Data width, shift direction, clock polarity, and phase are programmable. This allows communication with SPI - compatible devices using other synchronous serial channel. Transmission and reception of data is double-buffered. In slave mode, the SSC is limited to 1 MBit/s.

Features

- Master and Slave Mode operation
  - Full-duplex or half-duplex operation
- Transmit and receive buffered
- Flexible data format
  - Programmable number of data bits: 4 to 8 bits (1, 2 and 3 bit transfers are programmable, but forbidden)
  - Programmable shift direction: Least Significant Bit (LSB) or Most Significant Bit (MSB) shift first
  - Programmable clock polarity: idle low or high state for the shift clock
  - Programmable clock/data phase: data shift with leading or trailing edge of the shift clock
- Compatible with Serial Peripheral Interface (SPI)
- Interrupt generation
  - On a transmitter empty condition
  - On a receive condition
  - On a receiver FIFO full and empty condition
  - On an error condition (receive, baudrate, transmit error)

The following figure shows the block diagram of the SSC.

***Note**: In this chapter, the input and output signals from/to the GPIO ports are referred to with their generic name, such as MTSR, MRST, or SCLK. However, the actual input and output signals use separate lines from/to the GPIO, since they are connected in different ways to the pin logic (pin input line; alternate data output line). Only where explicitly applicable, the generic signal names are extended with an "_I" (input line) or "_O" (output line) identifier.*

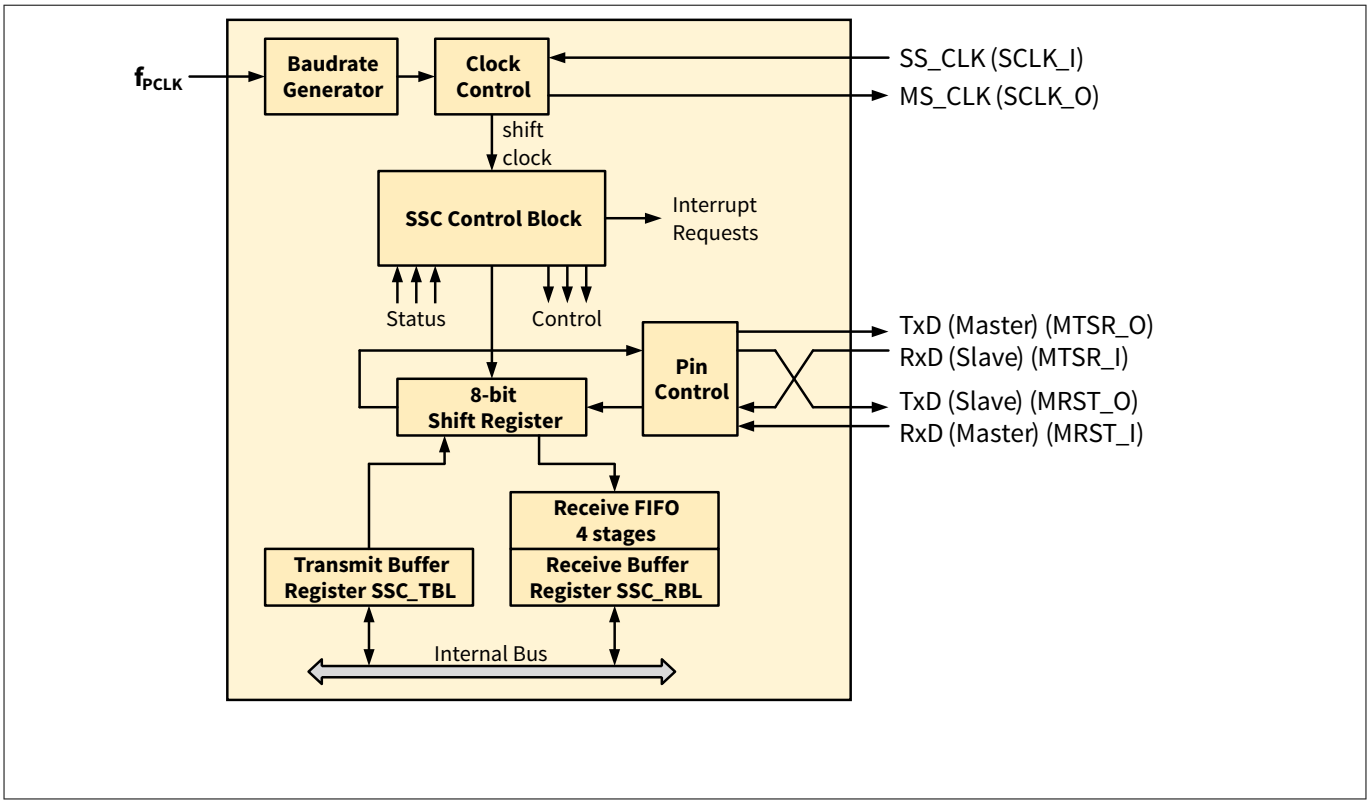


Figure 927 SSC block diagram

39.3.13.2 General operation

The SSC supports full-duplex and half-duplex synchronous communication of 3.33 MBaud (@20 MHz module clock). The serial clock signal can be generated by the SSC itself (Master Mode) or can be received from an external master (Slave Mode). Data width, shift direction, clock polarity, and phase are programmable. This allows communication with SPI devices. The data, which is transmitted and received is stored in is double-buffered registers. A 16-bit baudrate generator provides the SSC with a separate serial clock signal.

The SSC can be configured in a very flexible way, so it can be used with other synchronous serial channels, can serve for master/slave or multi-master interconnections, or can operate compatible with the popular SPI interface. Thus, the SSC can be used to communicate with shift registers (I/O expansion), peripherals (for example EEPROMs, etc.) or other controllers (networking). The SSC supports half-duplex and full-duplex communication. Data is transmitted or received on lines TXD and RXD, normally connected with pins MTSR (Master Transmit/Slave Receive) and MRST (Master Receive/Slave Transmit). The clock signal is output through line MS_CLK (Master Serial Shift Clock) or input through line SS_CLK (Slave Serial Shift Clock). Both lines are normally connected to pin SCLK.

39.3.13.2.1 Operating mode selection

The operating mode of the serial channel SSC is controlled by two register pairs, sharing the same addresses:

- During programming (SSC disabled SSC_CONPH.EN=0), access to register pair SSC_CONPH/SSC_CONPL is available, providing access to a set of control bits
- During operation (SSC enabled SSC_CONOH.EN=1), access to register pair SSC_CONOH/SSC_CONOL is available, providing access to a set of control/status flags

The two register pairs share the same two addresses. The selection between the registers is depending on the state of the enable bit SSC_CONPH.EN prior to an access to the register pairs.

The shift register of the SSC is connected to both the transmit lines and the receive lines via the pin control logic. Transmission and reception of serial data are synchronized and take place at the same time, that is the same number of transmitted bits is also received. Transmit data is written into the Transmit Buffer (SSC_TBL) and it is moved, on the next shifting edge, to the shift register as soon as this is empty. An SSC master (SSC_CONOH.MS=1) immediately begins transmitting, while an SSC slave (SSC_CONOH.MS=0) will wait for an active shift clock. When the transfer starts, the busy flag SSC_CONOH.BSY is set and the Transmit Interrupt Request line TIR will be activated within the timeframe of the first bit to indicate that register SSC_TBL may be reloaded again. When the programmed number of bits (4…8) has been transferred, the contents of the shift register are moved to the Receive FIFO and the Receive Interrupt Request line RIR will be activated for the first received frame. If no further transfer is to take place (SSC_TBL is empty), SSC_CONOH.BSY will be cleared (Master Mode). In Slave Mode, SSC_CONOH.BSY will always be cleared after a single transfer. Software shall not modify SSC_CONOH.BSY, as this flag is hardware controlled. If the mode of SSC is changed from operating to programming after the Transmit Interrupt Request line is set, the last byte of SSC data will not be transmitted. It is important to wait for SSC_CONOH.BSY to get cleared before changing the mode from operating to programming.

***Note**:* *The SSC starts transmission and sets SSC_CONOH.BSY minimum two clock cycles after transmit data is written into SSC_TBL. Therefore, it is not recommended to poll SSC_CONOH.BSY to check for the start and end of a single transmission. Instead, interrupt service routines should be used if interrupts are enabled, or the interrupt flags IRSCR_IRCON1.TIR and IRSCR_IRCON1.RIR should be polled if interrupts are disabled.*

***Note**:* *Only one serial interface can be master at a given time.*

The transfer of serial data bits can be programmed in many respects:

- The data width can be specified from 4 bits to 8 bits
- A transfer may start with either the LSB or the MSB
- The shift clock may be idle low or idle high
- The data bits may be shifted with the leading edge or the trailing edge of the shift clock signal
- The baudrate may be set from 152.59 Baud up to 3.33 MBaud (@20 MHz module clock)
- The shift clock can be generated (MS_CLK) or can be received (SS_CLK)

These features allow the adaptation of the SSC to a wide range of applications requiring serial data transfer.

The Data Width Selection supports the transfer of frames of any data length, from 4-bit “characters” up to 8-bit “characters”. Starting with the LSB (SSC_CONPL.HB=0) allows communication with SSC devices in Synchronous Mode or with 8051-like serial interfaces, for example. Starting with the MSB (SSC_CONPL.HB=1) allows operation compatible with the SPI interface.

Regardless of the data width selected and whether the MSB or the LSB is transmitted first, the transfer data is always right-aligned in registers SSC_TBL and SSC_RBL, with the LSB of the transfer data in bit 0 of these registers. The data bits are rearranged for transfer by the internal shift register logic. The unselected bits of SSC_TBL are ignored; the unselected bits of SSC_RBL will not be valid and should be ignored by the receiver service routine.

The Clock Control allows the adaptation of transmit and receive behavior of the SSC to a variety of serial interfaces. A specific shift clock edge (rising or falling) is used to shift out transmit data, while the other shift clock edge is used to latch in receive data. Bit SSC_CONPL.PH selects the leading edge or the trailing edge for each function. Bit SSC_CONPL.PO selects the level of the shift clock line in the idle state. Therefore, for an idle-high clock, the leading edge is a falling one, a 1-to-0 transition (see following figure).

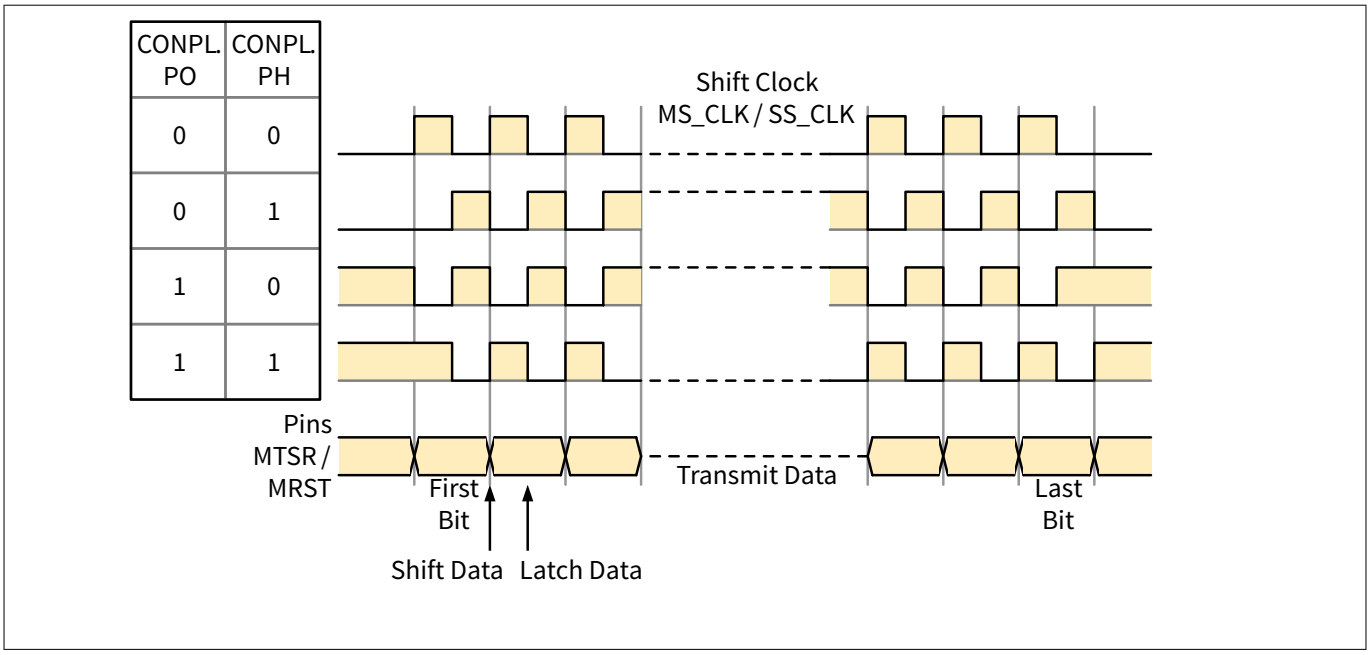


Figure 928 Serial clock phase and polarity options

39.3.13.2.2 Full-duplex operation

The various devices are connected through three lines. The definition of these lines is always determined by the master: the line connected to the master's data output line TXD is the transmit line; the receive line is connected to its data input line RXD; the shift clock line is either MS_CLK or SS_CLK. Only the device selected for master operation generates and outputs the shift clock on line MS_CLK. Since all slaves receive this clock, their pin SCLK must be switched to input mode. The output of the master's shift register is connected to the external transmit line, which in turn is connected to the slaves' shift register input. The output of the slaves' shift register is connected to the external receive line in order to enable the master to receive the data shifted out of the slave. The external connections are hard-wired, the function and direction of these pins is determined by the master or slave operation of the individual device.

***Note**: The shift direction shown in the figure applies for MSB-first operation as well as for LSB-first operation.*

When initializing the devices in this configuration, one device must be selected for master operation while all other devices must be programmed for slave operation. Initialization includes the operating mode of the device's SSC and also the function of the respective port lines.

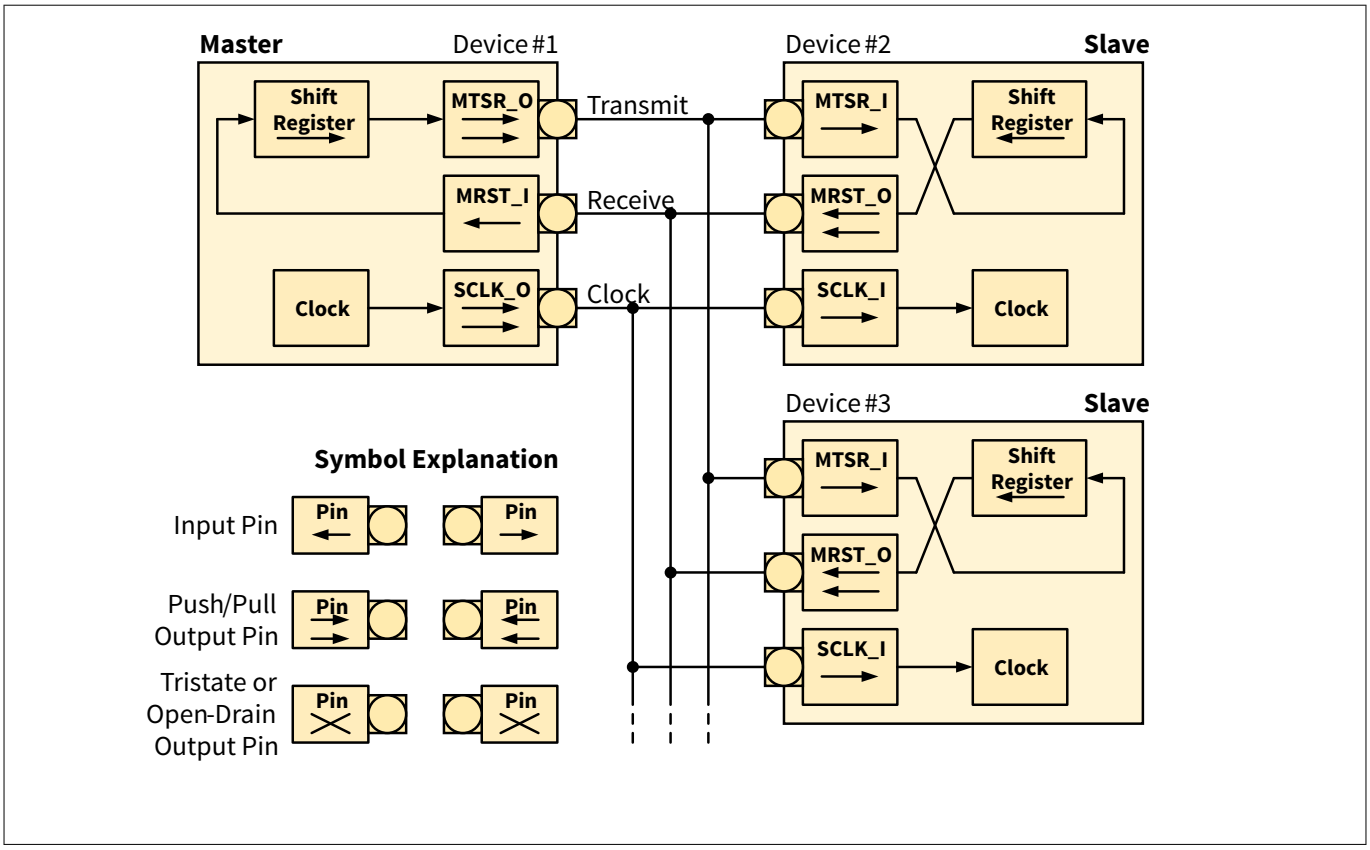


Figure 929 SSC full-duplex configuration

The data output pins MRST of all slave devices are connected together onto the one receive line in the configuration shown in the figure above. During a transfer, each slave shifts out data from its shift register. There are two ways to avoid collisions on the receive line due to different slave data:

- Only one slave drives the line, that is to say it enables the driver of its MRST pin. All the other slaves must have their MRST pins programmed as input so only one slave can put its data onto the master's receive line. It is possible to only receive data from the master. The master selects the slave device from which it expects data either by separate select lines or by sending a special command to this slave. The selected slave then switches its MRST line to output until it gets a de-selection signal or command. This option is applicable only if the pin has output driver disabling capability
- The slaves use open-drain output on MRST. This forms a wired-AND connection. The receive line needs an external pull-up in this case. Corruption of the data on the receive line sent by the selected slave is avoided when all slaves not selected for transmission to the master only send ones (1 s). Because this high level is not actively driven onto the line, but only held through the pull-up device, the selected slave can pull this line actively to a low-level when transmitting a zero bit. The master selects the slave device from which it expects data either by separate select lines or by sending a special command to this slave

After performing the necessary initialization of the SSC, the serial interfaces can be enabled. For a master device, the alternate clock line will now go to its programmed polarity. The alternate data line will go to either 0 or 1 until the first transfer starts.

When the serial interfaces are enabled, the master device can initiate the first data transfer by writing the transmit data into register SSC_TBL. This value is copied into the shift register (assumed to be empty at this time), and the selected first bit of the transmit data will be placed onto the TXD line before the next clock from the baudrate generator (transmission starts only if SSC_CONPH.EN=1).

Depending on the selected clock phase, a clock pulse will also be generated on the MS_CLK line. At the same time, with the opposite clock edge, the master latches and shifts in the data detected at its input line RXD. This “exchanges” the transmit data with the receive data. Because the clock line is connected to all slaves, their shift

registers will be shifted synchronously with the master's shift register — shifting out the data contained in the registers, and shifting in the data detected at the input line. After the pre-programmed number of clock pulses (via the data width selection), the data transmitted by the master is contained in all the slaves' shift registers, while the master's shift register holds the data of the selected slave. In the master and all slaves, the contents of the shift register are copied into the receive FIFO and the receive interrupt line RIR is activated for the first received frame.

When a slave device is transmitting its data, the bit SSC_CONOH.BSY is not set until the first clock edge at SS_CLK appears. The slave device will not wait for the next clock from the baudrate generator, as the master does. The reason for this is that, depending on the selected clock phase, the first clock edge generated by the master may already be used to clock in the first data bit. Thus, the slave's first data bit must already be valid at this time.

***Note**: For Master and Slave Mode the following applies: In the case SSC_CONPL.PH=1, the value is placed immediately on the bus, while for SSC_CONPL.PH=0, it is placed on the output with the first shifting edge. Besides, the value is not immediately copied into the shift register, but with the first shifting edge.*

***Note**: On the SSC, a transmission and a reception takes place at the same time, regardless of whether valid data has been transmitted or received.*

***Note**: The initialization of the SCLK pin on the master requires some attention in order to avoid undesired clock transitions, which may disturb the other devices. Before the clock pin is switched to output mode, the clock output level should be selected in the control register SSC_CONPL, and the alternate output be prepared via the appropriate GPIO register, or the output latch must be loaded with the clock idle level.*

39.3.13.2.3 Half-duplex operation

In a Half-Duplex Mode, only one data line is necessary for both receiving and transmitting of data (see following figure).

The master device controls the data transfer by generating the shift clock, while the slave devices receive it. Due to the fact that all transmit and receive pins are connected to the one data exchange line, serial data may be moved between arbitrary stations.

Similar to Full-Duplex Mode, there are two ways to avoid collisions on the data exchange line:

- Only the transmitting device may enable its transmit pin driver. This option is applicable only if the pin has output driver disabling capability
- The non-transmitting devices use open-drain output and send only ones

Because the data inputs and outputs are connected together, a transmitting device will clock in its own data at the input pin (MRST for a master device, MTSR for a slave). By this method, any corruptions on the common data exchange line are detected if the received data is not equal to the transmitted data.

The Half-Duplex Mode port configuration using three pins is shown in the following figure.

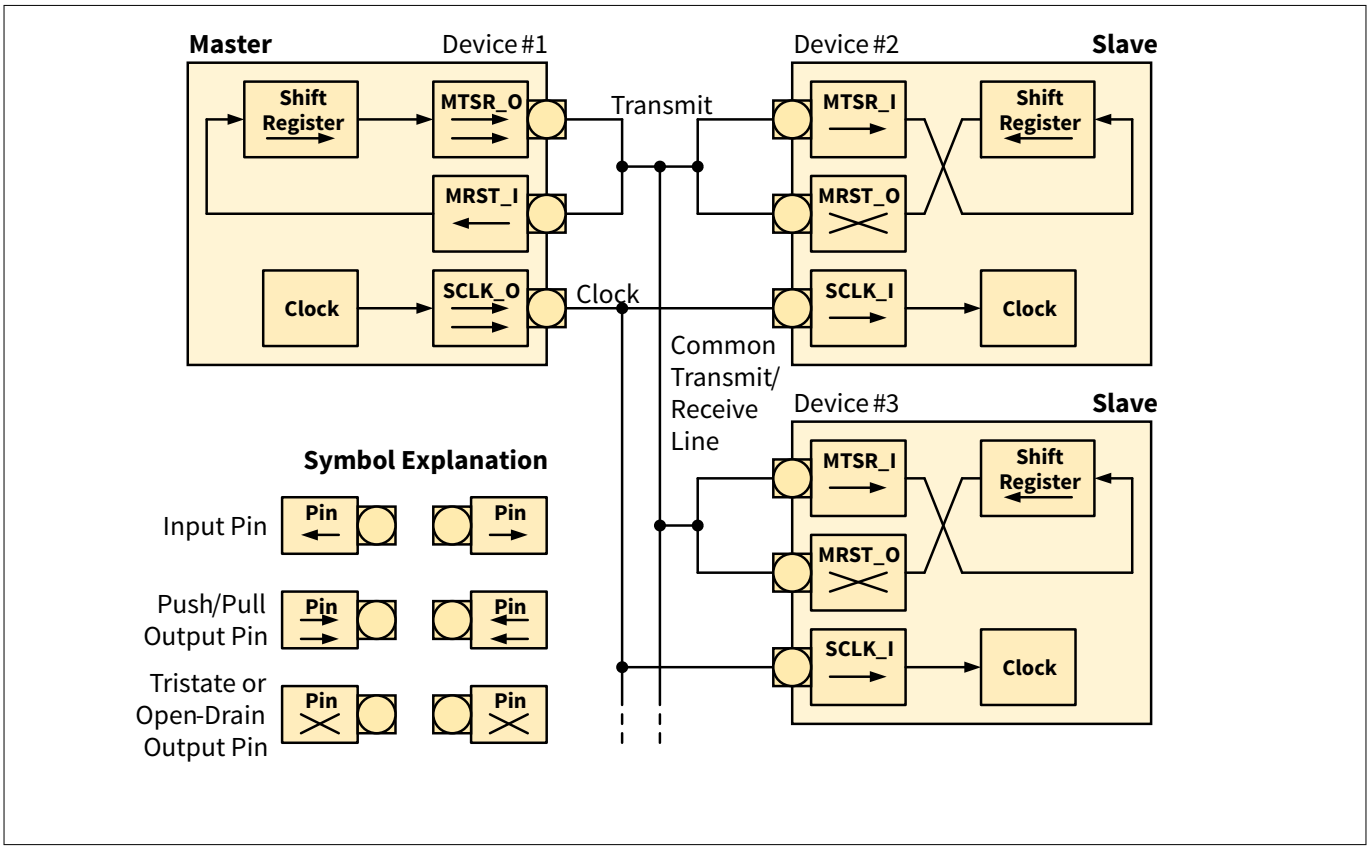


Figure 930 SSC half-duplex configuration

39.3.13.2.4 Loop back mode

To aid debugging, it is useful to receive back the data which is sent out. When SSC_CONPL.LB is set, the receive input is connected to the transmit output. Hence, the data which is sent out is automatically received back. Loop-back mode is supported only in master mode.

39.3.13.2.5 Continuous transfers (master mode only)

When the transmit interrupt request flag is set, it indicates that the transmit buffer SSC_TBL is empty and ready to be loaded with the next transmit data. If SSC_TBL has been reloaded by the time the current transmission is finished (at least 2 PCLK cycles before the last latching edge), the data is immediately transferred to the shift register and the next transmission will start without any additional delay. In such a case, there is no gap between the two successive frames on the data line. For example, two byte transfers looks the same as one word transfer. This feature can be used to interface with devices that can operate with or require more than 8 data bits per transfer. It is just a matter of software, how long a total data frame length can be. This option can also be used to interface to byte-wide and word-wide devices on the same serial bus, for instance.

***Note**: Of course, this can happen only in multiples of the selected basic data width, because it requires disabling/enabling of the SSC to reprogram the basic data width on-the-fly. Additionally, if the next data is now written any time after 2 PCLK cycles before the last latching edge, then there may be up to a half SCLK clock delay to start transmission of the new word.*

***Note**: In slave mode, when SSC_CONPL.PH = 0, if the data is written into the transmit buffer at least 4 PCLK cycles before the first SS_CLK edge of the next transfer (whether shifting or latching), then continuous transfer occurs and no transmit error is generated. Otherwise, a transmit error is generated. If SSC_CONPL.PH = 1, then the data has to be provided 7 PCLK cycles before the first SS_CLK edge of the next transfer. Otherwise, a transmit error is generated.*

39.3.13.2.6 Baudrate generation

The serial channel SSC has its own dedicated 16-bit Baudrate generator with 16-bit reload capability, allowing Baudrate generation independent of the timers. Next figure shows the Baudrate generator.

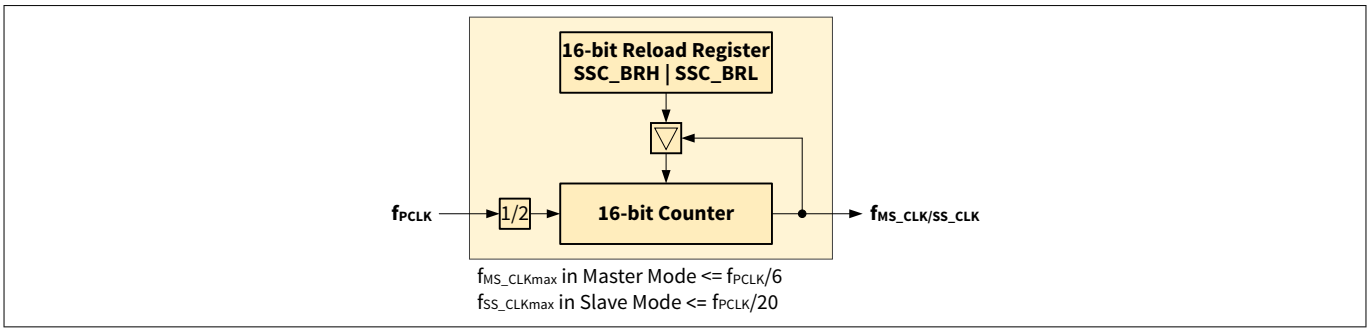


Figure 931 SSC baud-rate generator

The Baudrate generator is clocked with the module clock f_{PCLK}. The timer counts downwards. On underflow, it is reloaded with the 16-bit value BR contained in register pair SSC_BRH|SSC_BRL.

***Note**: Never write to BR while the SSC is enabled. The desired value shall be written to BR only when SSC is disabled.*

The formulas below calculate either the resulting Baudrate for a given reload value, or the required reload value for a given Baudrate:

***Note**: For the SSC module in the SCR, f_{hw_clk} is the PCLK (f_{PCLK}).*

$$Baudrate = \frac{f_{hw_clk}}{2 \times (< BR > + 1)}$$
$$BR = \frac{f_{hw_clk}}{2 \times Baudrate} - 1 \quad (69)$$

SSC baud rate calculation

BR represents the contents of the reload register pair, taken as an unsigned 16-bit integer, while Baudrate is equal to f_{MS_CLK/SS_CLK} as shown in the equation above.

The maximum Baudrate that can be achieved when using a module clock of 100 MHz is 16.67 MBaud in Master Mode (with BR=0002_H) (the theoretical value is 50 MBaud, but not guaranteed) or 5 MBaud (the theoretical value is 25 MBaud, but this value is not guaranteed) in Slave Mode (with BR=0009_H).

The table below lists some possible Baudrates together with the required reload values and the resulting bit times, assuming a module clock of 20 MHz.

Table 1723 Typical Baudrates of the SSC (f_{hw_clk}=20 MHz)

Reload value	Baudrate (= $f_{MS_CLK}/_{ss_clk}$)	Deviation
0000_H	Not allowed	0.0%
0001_H	Not allowed	0.0%
0004_H	2 MBaud (only in master mode)	0.0%
0009_H	1 MBaud	0.0%
0013_H	500 kBaud	0.0%
0063_H	100 kBaud	0.0%
$FFFF_H$	152.59 Baud	0.0%

39.3.13.2.7 Error detection mechanisms

The SSC is able to detect three different error conditions. Receive Error is detected in all modes; "Transmit Error" and "Baudrate Error" apply only to Slave Mode. When an error is detected, the respective error flag is/can be set and an error interrupt request will be generated if enabled (see following figure). The error interrupt handler may then check the error flags to determine the cause of the error interrupt. The error flags are not reset automatically, but must be cleared by software after servicing. This allows servicing of some error conditions via interrupt, while the others may be polled by software.

***Note**: The error interrupt handler must clear the associated (enabled) error flag(s) to prevent repeated interrupt requests.*

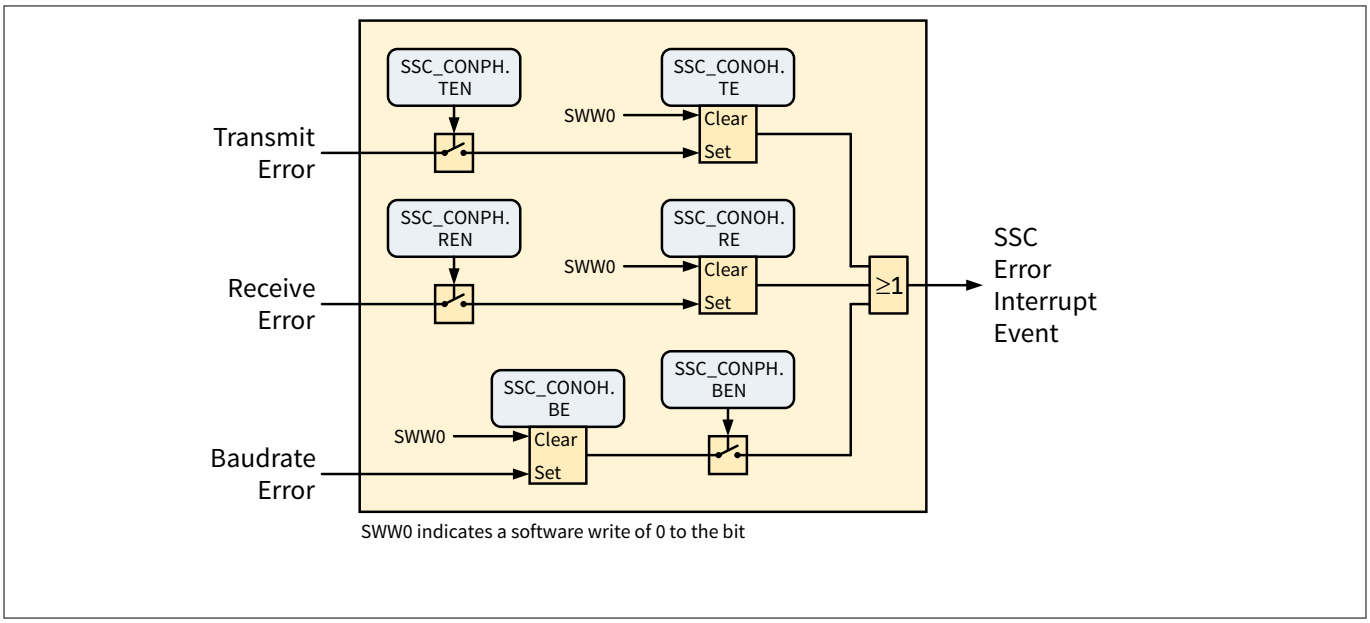


Figure 932 SSC error interrupt control (interrupt pulse no signal)

A Receive Error (Master or Slave Mode) is detected when the FIFO is full (FIFO full already set) and another frame is completely received, but the previous data was not read out of the receive FIFO. This condition sets the error flag SSC_CONOH.RE and the error interrupt request line, when enabled via SSC_CONPH.REN. The FIFO has four elements, afterwards, the reception is stopped. FIFO full will be raised. When Receive Error is raised, an additional reception was intended, but as the FIFO is full it is lost.

A "Baudrate Error" (Slave Mode) is detected when the incoming clock signal deviates from the programmed Baudrate by more than 100%, that means it is either more than double or less than half the expected baudrate.

This condition sets the error flag SSC_CONOH.BE and, when enabled via SSC_CONPH.BEN , the error interrupt request line. Using this error detection capability requires that the slave's baudrate generator is programmed to the same baudrate as the master device. This feature detects false additional, or missing pulses on the clock line (within a certain frame).

A Transmit Error (Slave Mode) is detected when a transfer was initiated by the master (SS_CLK gets active) but the transmit buffer SSC_TBL of the slave was not updated since the last transfer. This condition sets the error flag SSC_CONOH.TE and the error interrupt request line, when enabled via SSC_CONPH.TEN. If a transfer starts while the transmit buffer is not updated, the slave may shift out invalid data. This may lead to corruption of the data on the transmit/receive line in half-duplex mode (open-drain configuration) if this slave is not selected for transmission. This mode requires that slaves not selected for transmission only shift out ones; that is, their transmit buffers must be loaded with $FFFF_H$ prior to any transfer.

***Note**: In order to avoid possible conflicts or misinterpretations, it is recommended to always load the slave's transmit buffer prior to any transfer.*

The cause of an error interrupt request (receive, baudrate, transmit error) can be identified by the error status flags in control register SSC_CONOH.

***Note**: The error status flags SSC_CONOH.TE, SSC_CONOH.RE, and SSC_CONOH.BE are not reset automatically upon entry into the error interrupt service routine, but must be cleared by software.*

39.3.13.3 SSC interrupts

The five SSC interrupts can be separately enabled or disabled by setting or clearing their corresponding enable bits in SFR IR_MODIEN.

For a detailed description of the various interrupts see General Operation in related information. An overview is given in the following table.

Table 1724 SSC interrupt sources

Interrupt	Signal	Description
Transmission starts	TIR	Indicates that the transmit buffer can be reloaded with new data
Transmission ends	RIR	The configured number of bits has been transmitted and shifted to the receive buffer. This is only the case for the first reception after FIFO empty
Receive FIFO full	RFFIR	The receive FIFO is full and will stop reception. Further information is lost
Receive FIFO empty	RFEIR	The receive FIFO is empty, further access will provide junk data
Receive Error	EIR	This interrupt occurs if a new data frame is completely received while the FIFO is full
Baud-Rate Error (Slave Mode only)	EIR	This interrupt is generated when the incoming clock signal deviates from the programmed baud-rate by more than 100%
Transmit Error (Slave Mode only)	EIR	TE is generated in case corrupted data is sent. This can be an empty transmit buffer or corrupted data on the serial bus (in case the transmit buffer is overwritten round about transmission start)

Following figure shows a diagram of the interrupt request handling of the SSC.

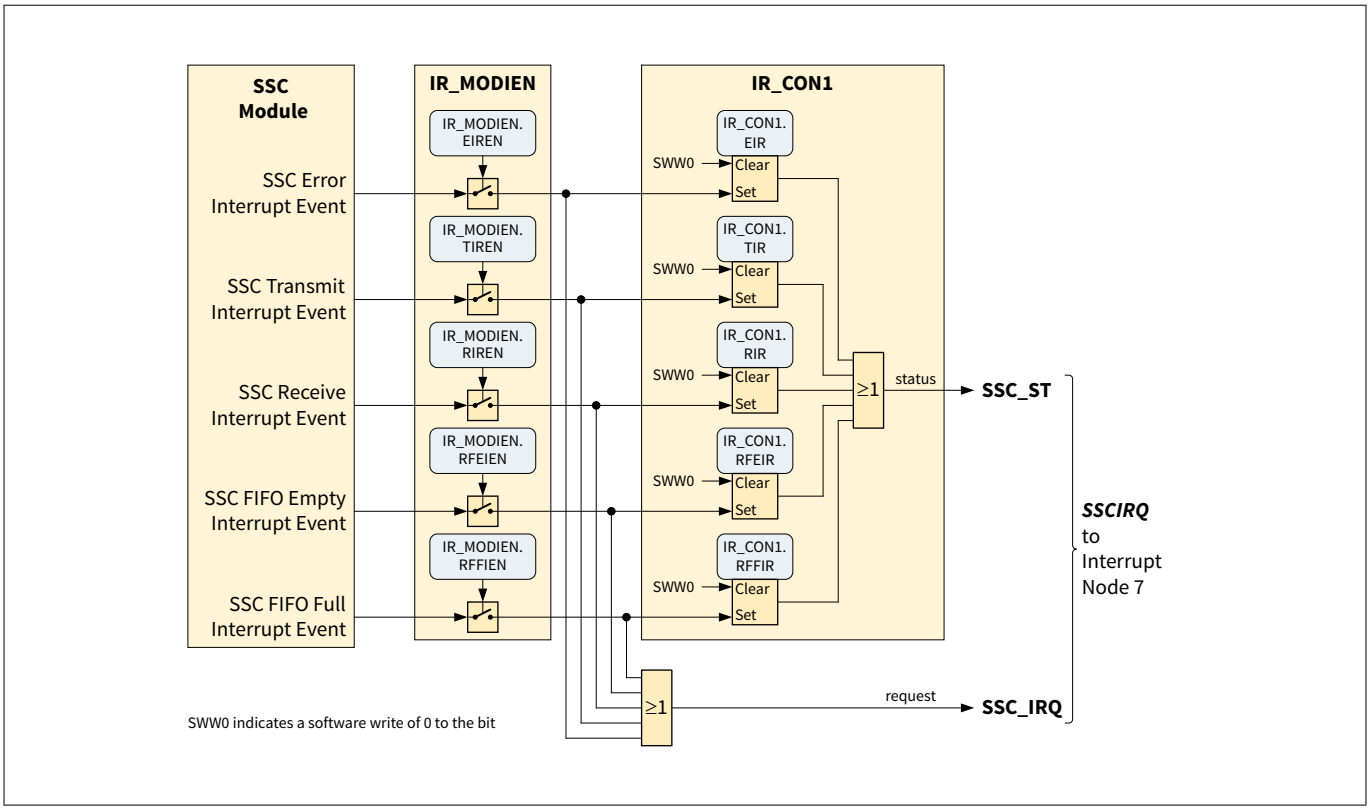


Figure 933 SSC interrupt request overview

Related information

General operation on page 6638

39.3.13.4 SSC connections to GPIO

The following figure illustrates the signal connections of the SSC to the GPIO ports.

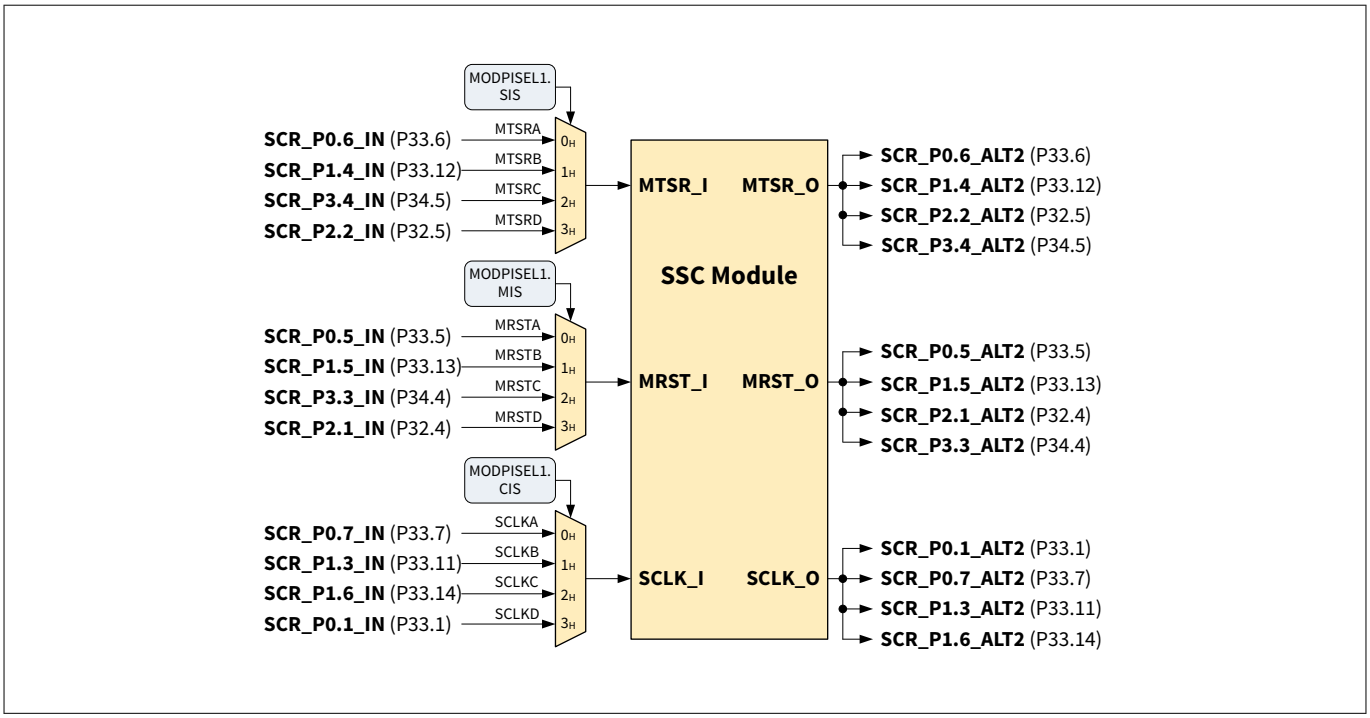


Figure 934 SSC connections to GPIO ports

39.3.13.5 Register overview

The SSC Special Function Registers are accessed from the standard (non-mapped) SFR area.

Configuration register

The operating mode of the serial channel SSC is controlled by two register pairs, sharing the two same addresses:

- Register pair SSC_CONPH/SSC_CONPL provides access to a set of control bits and is accessible in Programming Mode
- Register pair SSC_CONOH/SSC_CONOL provides access to a set of control/status flags and is accessible in Operating Mode

The selection between the register pairs is depending on the state of the enable bit EN prior to an access to the register. This bit is available at the MSB position in both registers, SSC_CONPH and SSC_CONOH:

- When EN=0 (Programming Mode), registers SSC_CONPH and SSC_CONPL are accessible
- When EN=1 (Operating Mode), registers SSC_CONOH and SSC_CONOL are accessible

Since the selection is dependent on the state of the EN bit prior to an access, writing, for example, 11000000_B to SSC_CONPH in programming mode (SSC_CONPH.EN = 0) will initialize the SSC (SSC_CONPH.EN was 0) and then turn it on. Then, register pair SSC_CONOH|SSC_CONOL is accessible, with SSC_CONOH.EN = 1.

***Note**: Between programming SSC_CONPH.EN=1 and writing into SSC_TBL a minimum delay of 4 cycles has to be kept. (see SSC_TBL)*

Baud rate timer reload register

The SSC baudrate timer reload register pair SSC_BRH|SSC_BRL contains the 16-bit reload value BR for the baudrate timer.

Transmit buffer register

***Note**: Before writing to SSC_TBL a minimum delay of 4 PCLK cycles has to be regarded after writing,SSC_CONPH.EN=1. for example:*

SSC_CONPH.EN=1;
for (i=0;i<4;i++)
nop();
SSC_TBL=Byte to be transferred.
Instead of nop() more useful code can be used, as long as a delay of 4 cycles is guaranteed.

It is recommended to use the transmit interrupt as signal to write the next piece of information to the transmit buffer register. This way, it is guaranteed, that the module has the information on time (in slave mode) to guarantee the transmission.

Receive buffer register

The SSC receive buffer register SSC_RBL contains the received data value. It is the access point to the FIFO. The FIFO depth is 4 bytes. The FIFO stops receiving, when full. Warning flags for FIFO full (RFF) and FIFO empty (RFE) exist in SSC_CONOH.

39.3.14 ADC Comparator Unit (ADCOMP)

The ADCOMP constitutes an 11-bit successive approximation (SAR) Analog-to-Digital converter, supporting up to 16 channels. The ADCOMP unit can be periodically activated by the SCR to measure certain analog inputs. The ADCOMP unit can be quickly activated, followed by few measurements, after which the unit is powered off to keep the overall Standby current to the minimum.

The conversion control involves enabling the ADC Comparator unit, selecting the channel, selecting sampling time and subsequently issuing start of conversion through software. After conversion is completed, end of conversion event is indicated and an interrupt may be generated.

39.3.14.1 Features

The ADCOMP has the following features:

- 11-bit resolution
- Periodic monitoring of up to 16 ADC channels
- Operating frequency 50 MHz (100 MHz/2)
- Configurable sampling times

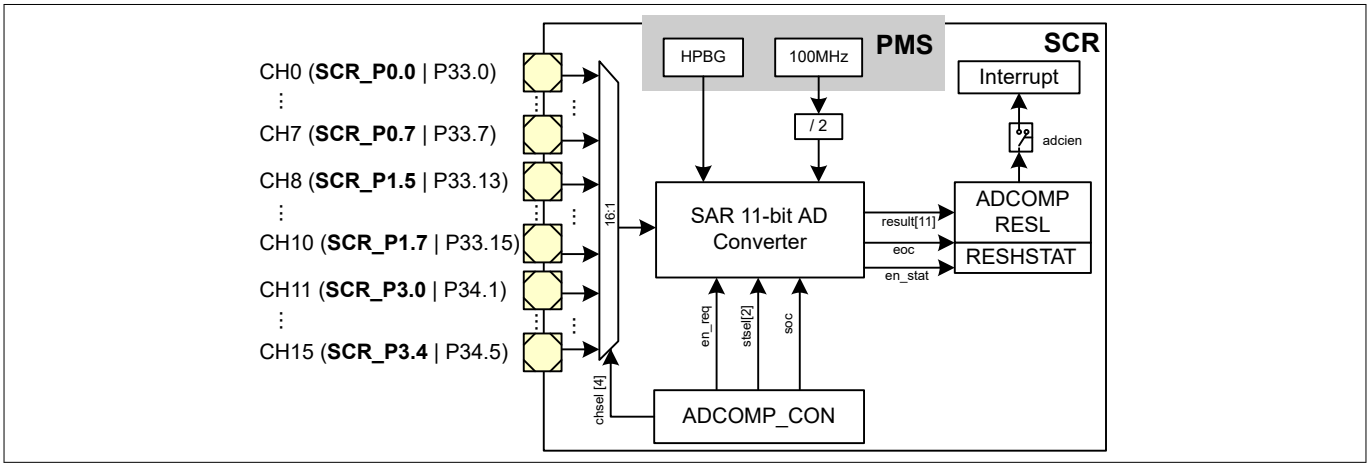


Figure 935 ADC comparator overview

For detailed mapping of SCR Ports to ADCOMP input channels, please refer to the SCR GPIO chapter.

***Note**: A pad should be in a tristate configuration in order to be used as an analog input.*

For detailed electrical characteristics of ADCOMP please refer to device datasheet.

39.3.14.2 Overview

The ADC unit is clocked with 50 MHz clock derived from the HPOSC 100 MHz clock. The reference voltage for the conversion is derived from the high-precision bandgap HPBG. The analog channels are selected through a 16:1 multiplexer. The ADCOMP unit is enabled through ADCOMP_CON.EN_REQ. The conversion control and result handling part is encapsulated in the SCR digital domain, and is clocked by PCLK.

39.3.14.3 ADC comparator (ADCOMP) operation

This single ADC comparator can be operated by SCR independent from PMS.

Before an ADC conversion is triggered, HPBG band-gap, ADC unit and 100 MHz back-up clock have to be enabled:

- If backup clock is disabled (by having SCR_SCUSCR_CLK.DIV_REQ = DIV70K) the clock source has to be enabled from SCR side by setting SCUSCR_CLK.DIV_REQ to a different value than DIV70K
- ADCOMP is by default disabled, so starting ADC is triggered by setting ADCOMP_CON.EN_REQ
- The activation of the ADCOMP unit triggers the activation of the band gap HPBG

After enabling ADCOMP unit a certain stabilization time needs to be waited – time will be in range of 50 µs, and it will depend if HPBG was enable or not before enabling ADCOMP.

In most cases,

- if the system was in "RUN mode" prior to setting ADCOMP_CON.EN_REQ, HPBG is enabled and this stabilization time is shorter (around 15 µs),
- and if the system was in STBY0 or STBY1_70kHz mode prior to setting ADCOMP_CON.EN_REQ, HPBG is disabled and it will take longer (around 65 µs)
- Once the ADCOMP components are initialized and ready to use, the bit ADCOMP_RESHSTAT.EN_STAT is set showing that the ADCOMP is ready. This allows a start of conversion (SOC) to be triggered, as shown in the figure below
- Start of conversion (SOC) can be requested even if ADCOMP_RESHSTAT.EN_STAT is not set. In this case, the conversion will be started as soon as the ADCOMP_RESHSTAT.EN_STAT is set and the ADCOMP is up and running

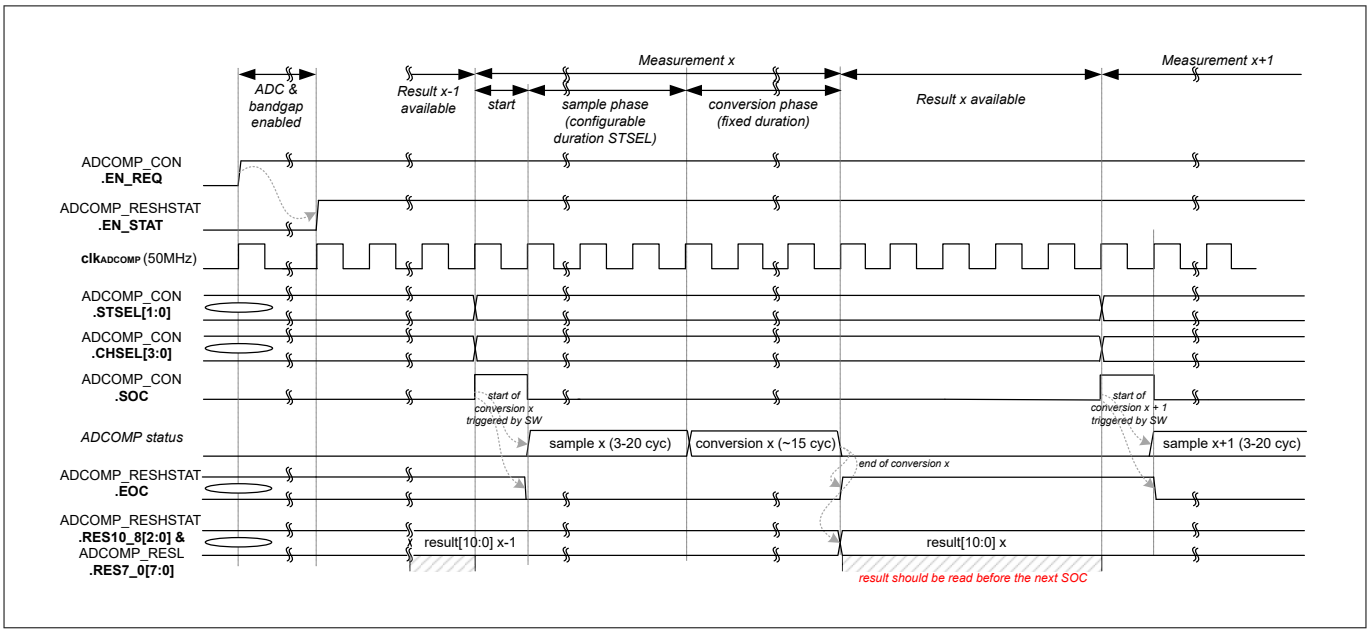


Figure 936 ADC comparator signal diagram

After ADCOMP is enabled each measurement is carried with following steps:

1. Start of conversion (SOC) is initiated by SCR software through ADCOMP_CON.SOC bit

- When the "Start of Conversion" is requested, the "End of Conversion" (EOC) bit is cleared. End of Conversion is set, when the requested conversion is finished. Start of Conversion can be set at anytime irrespective of the "EOC" flag. If Start of Conversion is requested while the conversion is ongoing that means End of Conversion flag is not set, the next conversion will be started immediately after the current conversion has ended. End of Conversion flag will be set only when the second conversion has finished

- The sampling and conversion phases are carried out back to back
- The required channel and sample time duration are selected through ADCOMP_CON.CHSEL (channel-select) and ADCOMP_CON.STSEL (sample-time-select)
  - Both ADCOMP_CON.CHSEL and ADCOMP_CON.STSEL can be set after the ADCOMP is enabled, and even in the same write access as when SOC is set to '1'
  - The sample time duration ADCOMP_CON.STSEL can be one of the four predefined values Conversion time is not configurable
  - In order to get a valid measurement the ADCOMP_CON.CHSEL value shall not be changed before end of conversion is reached

2. After each conversion phase is completed, bit ADCOMP_RESHSTAT.EOC is set

- An interrupt is triggered if enabled through IRSCR_MODIEN.ADCIEN enable bit
- The converted 11-bit result is transferred to ADCOMP_RESL.RES7_0 (lower 8 bits) and ADCOMP_RESHSTAT.RES10_8 (upper 3 bits). Result should be read before the subsequent SOC conversion event
- EOC bit serves also as a valid indication

Once the sample and conversion time (Conversion time = 15 cycles) have elapsed, a new measurement can be triggered by software. If a measurement is triggered before the end of conversion is reached, the EOC will not be signaled.

After the conversion is completed, ADCOMP unit may be disabled by clearing ADCOMP_CON.EN_REQ register bit. If ADCOMP_CON.EN_REQ is cleared during ongoing conversion, then the ADC is disabled only after the currently running conversion has ended, but the result is not saved.

In case of an LVD reset event during STANDBY1, the ADCOMP operation is aborted. Consequently, the SCR is switched off and ADC is disabled after that reset. In case of an SCR-only reset, the ADCOMP is disabled after finished conversion, similar to clearing ADCOMP_CON.EN_REQ.

39.3.14.4 ADCOMP interrupt request

Figure below illustrates the interrupt request handling of the ADCOMP module.

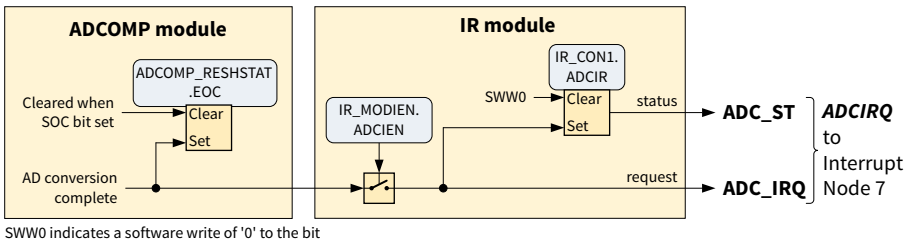


Figure 937 ADCOMP interrupt request overview

Upon completion of a conversion an interrupt is signaled to Interrupt Node 7, if ADC interrupt is enabled through IR_MODIEN.ADCIR bit. Information on interrupt request is also stored in IR_CON1.ADCIR, which should be cleared by software if set. Besides triggering an interrupt, end of conversion signal is also stored in a local ADC bit ADCOMP_RESHSTAT.EOC until the next start of conversion.

39.3.14.5 ADCOMP register overview

ADCOMP contains only 3 registers, as shown in the table below. Beside those, an interrupt enable bit ADCIEN is located inside IRSCR_MODIEN, and interrupt status flag ADCIR inside IRSCR_CON1 register.

Table 1725 Register overview - ADCOMP (sorted by name)

Short name	Description	Offset address RMAP=0, PAGE=X
ADCOMP_CON	ADCOMP control register	$C7_H$
ADCOMP_RESL	ADCOMP result register, lower 8-bits	$C8_H$
ADCOMP_RESHSTAT	ADCOMP result and status register, higher 3-bits of result	$C9_H$

39.3.15 Wake-up CAN filter (WCAN)

This chapter describes the WCAN controller of the SCR. It contains the following sections:

- Overview on the Wake-Up capabilities
- Functional description of the WCAN kernel
- WCAN Kernel register description
- Functional description on the CAN based kernel
- SCR implementation specific details and registers of the WCAN (port connections and control, interrupt control, address decoding, clock control)

39.3.15.1 Definitions and abbreviations

- WUF - Wake-Up Frame
- WUP - Wake-Up Pattern; Wake-Up Pattern detection has to be supported by external CAN transceiver and WUP detection has to be notified by transceiver to WCAN by a falling edge on RXD. To keep the context simple, the term Wake-Up Pattern is used in WCAN specification

39.3.15.2 Wake-up CAN filter implementation

39.3.15.2.1 Overview

This section describes the ISO 11898-6 datalink layer. The Wake-Up CAN module contains 1 CAN node, representing only the receive part of the communication interface. Furthermore, the Wake-Up CAN filter functionality and interfaces in TC4Dx.

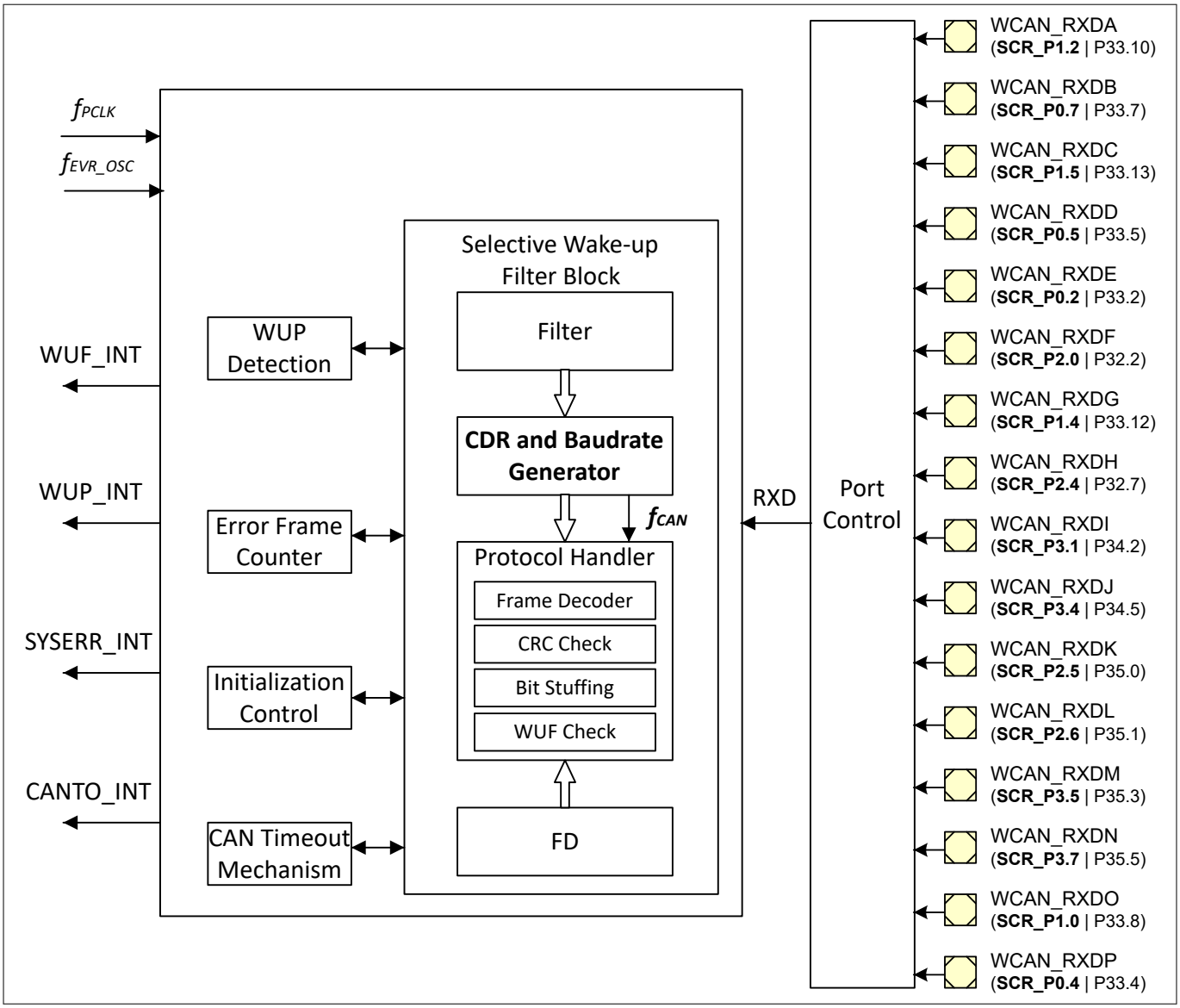


Figure 938 Wake-up CAN filter interfaces

39.3.15.2.2 WCAN filter features

The following are the feature list for WCAN module in SCR:

- Wake-Up CAN functionality similar to ISO 11898-6
- 1 Wake-Up CAN node
- Guaranteed data transfer rate up to 500 KBit/s. Default settings for 125 KBit/s, 250 KBit/s and 500 KBit/s are according to the given propagation delays
- 1 Wake-up frame configuration registers to the wake-up node:
  - Set up to handle frames with 11-bit and/or 29-bit identifiers
  - Programmable acceptance mask register for filtering
  - Programmable Wake-Up message ID, DLC and data
- Wake-Up CAN message time-out feature
- Power saving through Standby Mode on CAN message time-out
- WUP detection by monitoring falling edge on RXD when in Standby Mode

- Tolerant to CAN FD frames
- Error counter to de-bounce CAN protocol and Wake-Up function errors
- Interrupt generation on:
  - Successful Wake-Up message frame reception
  - Error recognition
  - Wake-Up CAN timeout
  - WUP (falling edge on RXD) detection

39.3.15.2.3 Introduction

Feature overview

Reception and evaluation of CAN frames is handled in like in CAN specification ISO 11898-2 (formally known as ISO11898-6) data link layer. The received messages will be compared against the programmed wake-up pattern and in case of pattern match, a wake-up will be issued. In case of an erroneous understanding or errors on the bus, the receive error counter can also reach a wake-up threshold. The node will be purely listening to the bus and will not communicate to the outside. The CAN node has Wake-Up Frame (WUF) configuration registers. In these WUF registers, the identifier including a bit-wise mask, the corresponding DLC as well as a data mask is included. Only (outside the erroneous case), if all criteria match, a wake-up will be initiated. The bit timings for the CAN nodes are derived from the CAN clock (f_{CAN}) and are programmable up to a data rate of 1 MBit/s, nonetheless guaranteed wake-up is only given for special configurations according to the propagation delay defined by the German OEMs.

Features:

- Similar to ISO 11898-2
- Receive operation similar to ISO 11898-2
- Tolerant to CAN FD frames
- Dedicated control registers for the CAN node. Data transfer rates are guaranteed until 500 kBit/s, in case of higher baudrates, more wake-up frames might be needed. Transmitter tolerances up to 0.4% are calculated and guaranteed with CDR enabled. Higher transmitter tolerances are not pretested
  - The message buffer can be configured to accept only standard or only extended Classical frames as valid Wake-Up frame
- Four interrupt sources do exist, wake-up frame found, $t_{SILENCE}$ expired, error overflow and WUP detected

39.3.15.2.4 WCAN initialization sequence

In order to initialize the WCAN module, the following sequence has to be performed:

- Enable WCAN module by setting SCUSCR_PMCON0.WCAN_DIS = 1
- Enable WCAN input pin SCR_Pn_PDISC.PDISm = 0
- Set WCAN_CFG.WCAN_EN = 1
- Set WCAN_CFG.CCE = 1
- Configure CDR,CAN FD and baudrate configuration registers
- Reset WCAN_CFG.CCE = 0
- Reset WCAN_CFG.SELWK_EN = 0
- Configure WUF configuration registers
- Set WCAN_CFG.SELWK_EN = 1
- Wait until WCAN_INTESTAT0.SWACK = 1

In order to perform only reconfiguration of the WUF configuration registers, the following sequence has to be performed

- Set WCAN_CFG.WCAN_EN = 1
- Reset WCAN_CFG.SELWK_EN = 0
- Wait until WCAN_INTESTAT0.SWACK = 0
- Configure WUF configuration registers
- Set WCAN_CFG.SELWK_EN = 1b
- Wait until WCAN_INTESTAT0.SWACK = 1

After the initialization/WUF reconfiguration sequence is performed, the application software has to check the WCAN_INTESTAT1.SWKSET bit-field to verify if the Selective Wake-Up feature is successfully enabled or not. The WCAN_INTESTAT1.SWKSET bit-field is set to 1 if all of the below mentioned conditions are true, and it is cleared if any of the conditions is not met.

- WCAN_CFG.WCAN_EN = 1b
- WCAN_CFG.CCE = 0b
- WCAN_INTESTAT0.SWACK = 1b
- WCAN_INTESTAT0.MODE = 0b
- f_{BACK} is switched on

***Note**: If application software writes either WCAN_CFG.CCE = 1 or WCAN_CFG.SELWK_EN = 0b when WCAN is already active, then any ongoing reception will be lost. If WCAN_CFG.WCAN_EN = 0 when WCAN is already active, then the selective wake-Up filter block will be in halt state and the application software has to perform re-initialization based on the sequence given above.*

39.3.15.2.5 Wake-up frame configuration and detection

The received frame is compared to the configured wake-up frame and in case a match is detected a wake-up request is issued. The wake-up request sets the WCAN_INTESTAT1.WUF flag and issues WUF_INT interrupt if not masked by WCAN_INTMRSLT.WUFMASK. Only a matched Classical CAN frame can lead to WUF detection.

***Note**: In addition to the message object configuration registers (i.e. WCAN_DLC_CTRL, IDn_CTRL(n=0-3), MASK_IDn_CTRL(n=0-3), WCAN_DATAn_CTRL) the node bit timing configuration is configured in the BTLn_CTRL(n=1-2) register.*

Configuration of wake-up frame filter

The message object configuration of the wake-up frame is done in the following registers:

- Register IDn_CTRL(n=0-3)) contains the CAN identifier filter bits
- Register MASK_IDn_CTRL(n=0-3) contains the ID mask bits
- Register WCAN_DLC_CTRL are the data length code filter bits
- Register WCAN_DATAn_CTRL are the data filter bits

Detection of wake-up frame filter

The compare mechanisms are different according to the information field inside the CAN frame:

- The identifier programmed inside IDn_CTRL(n=0-3) and MASK_IDn_CTRL(n=0-3) has to match the data in the received frame and the logic seen in Figure 939, a wake-up condition is detected
- The DLC programmed in WCAN_DLC_CTRL together with the data filter bits in WCAN_DATAn_CTRL will be compared to the DLC and data inside the received frame, in case a single ‘1’ match in WCAN_DATAn_CTRL,

a wake-up condition within the data is detected. DLC values greater than 8 will be treated as 8. See following figure

If and only if, identifier and masking, the WCAN_DLC_CTRL and the WCAN_DATAn_CTRL will lead to a match, a valid wake-up Frame (WUF) is detected and a wake-up interrupt will be issued if WCAN_CFG.SELWK_EN is enabled.

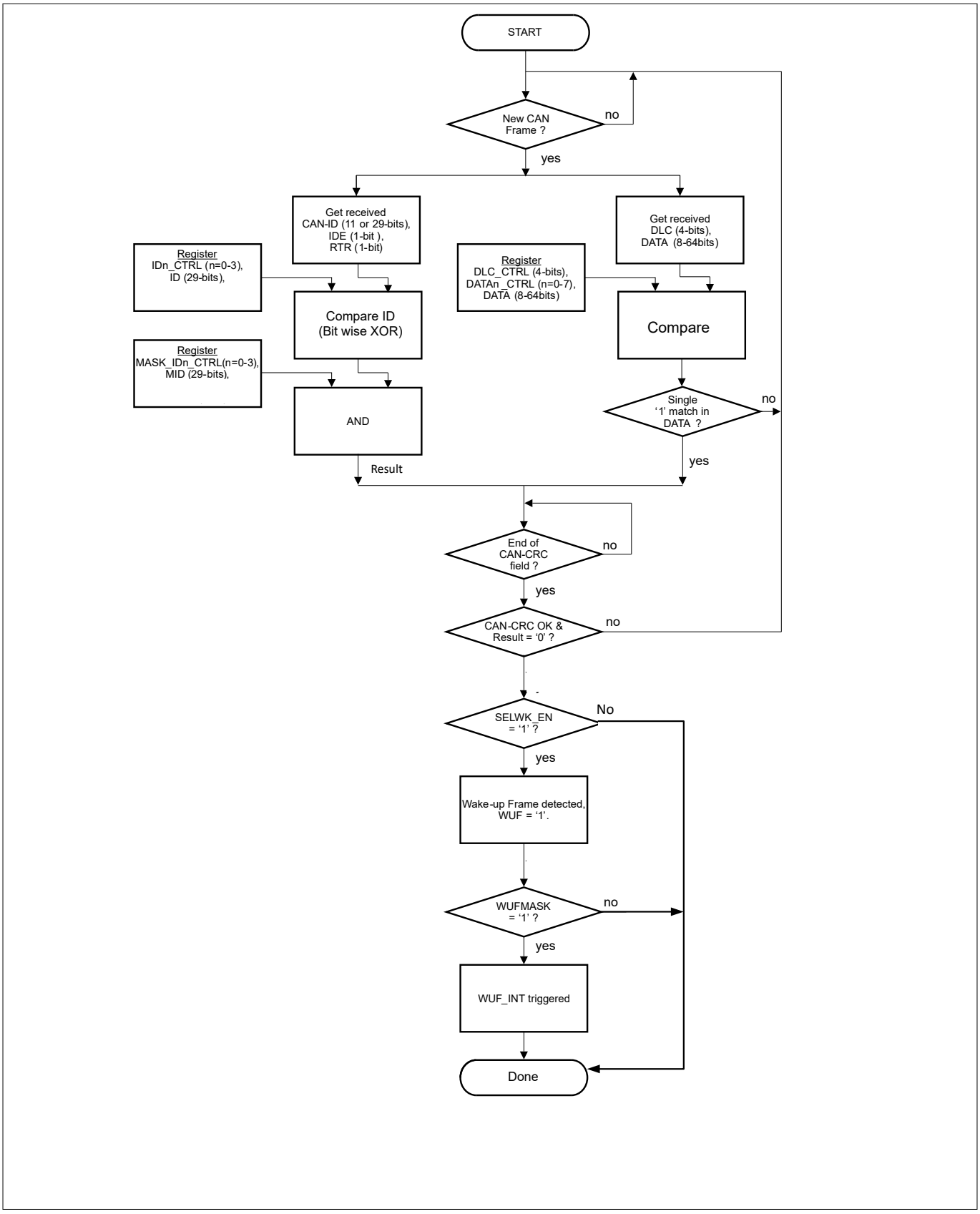


Figure 939 Wake-up frame detection

39.3.15.2.6 CAN message timeout

A timeout mechanism to monitor CAN bus activity is available on the WCAN. The timeout mechanism monitors the RXDC0 pin. When the RXDC0 pin is inactive for a $t_{SILENCE}$ time of 750 ms, both CAN bus silent bit (WCAN_INTESTAT1.CANSIL) and if selective Wake-Up is enabled, CAN timeout bit (WCAN_INTESTAT0.CANTO) will be set with a CAN time out interrupt (CANTO_INT). An occurrence of CAN Timeout will clear the error counter.

The CAN bus silent bit (WCAN_INTESTAT1.CANSIL) is reset when there is activity on the CAN bus where else the CAN timeout bit can be reset by setting CAN time out clear bit (WCAN_INTESCLR0.CANTOCLR). When WCAN_CFG.WCAN_EN = 0, the timeout counter remains in the reset state. The WCAN_INTESTAT1.CANSIL can also be cleared by setting WCAN_INTESCLR1.CANSILCLR.

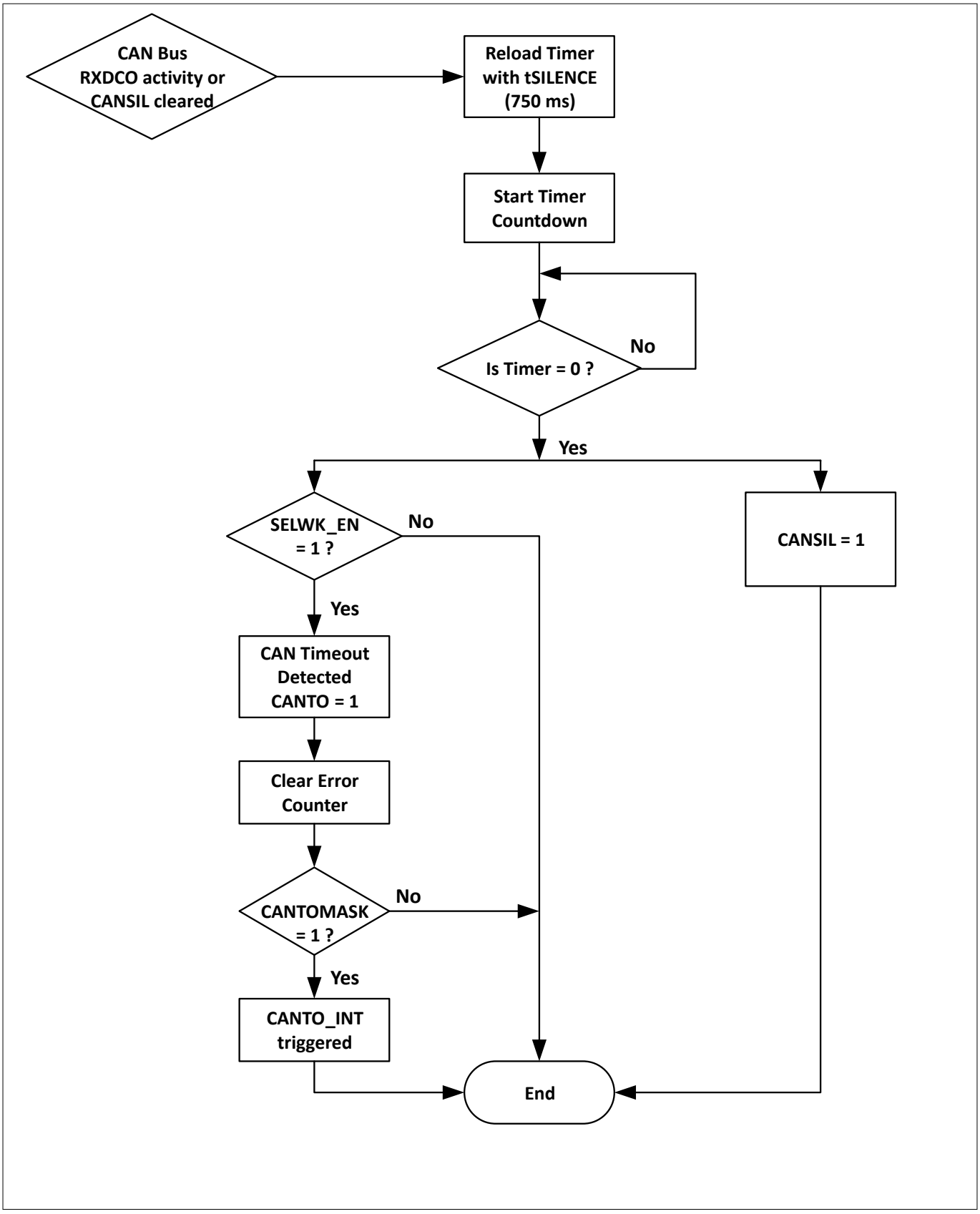


Figure 940 CAN timeout

39.3.15.2.7 Error counter

The error counting mechanism in ISO 11898-6 is different to ISO 11898-1. As the device is only receiving messages and never transmitting it could never get into a bus-off situation. Therefore an ISO11898-6 implementation is incrementing its error counter on a Form, Stuff and a CRC error. In case of a correct message, the error counter will be decremented, if it is greater than 0. In case the error counter is greater than 31 an overflow occurs and a wake-up will be initiated. This shall enable the software to correct the situation. In case of overflow the WCAN_INTESTAT0.SYSERR and WCAN_INTESTAT0.ECOFL flags are set. The error counter is cleared when timeout has occurred or when WCAN_CFG.SELWK_EN is re-enabled (from '0' to '1'). If the CAN FD tolerance feature is enabled and a CAN FD frame is received, the error monitoring is done only until WCAN_INTESCLR1.FDF bit-field. The behavior of the error counter for CAN FD frames is based on WCAN_FRMERRCNT.FDEC_DIS bit-field. The conditions at which error counter is incremented or decremented is given in the table below. The WCAN_INTESTAT0.ECOFL flags can be cleared by writing WCAN_INTESCLR0.ECOFLCLR to '1'.

***Note**: If ECNTCLR is written to '1' and at the same time an error in received frame has occurred, the error counter will remain cleared and it will not be incremented.*

Table 1726 Error counter conditions

WCAN_FD_CTRL.**FDEN**	WCAN_FRMERRCNT.**FDEC_DIS**	**Behavior of error counter**
0	X	Decrement on: • Valid classical CAN frame Increment on: • Corrupt classical CAN frame • CAN FD frame
1	0	Decrement on: • Valid classical or CAN FD frame Increment on: • Corrupt CAN FD frame • Corrupt classical CAN frame
1	1	Decrement on: • Valid classical CAN frame Increment on: • Corrupt classical CAN frame No action: • CAN FD frame

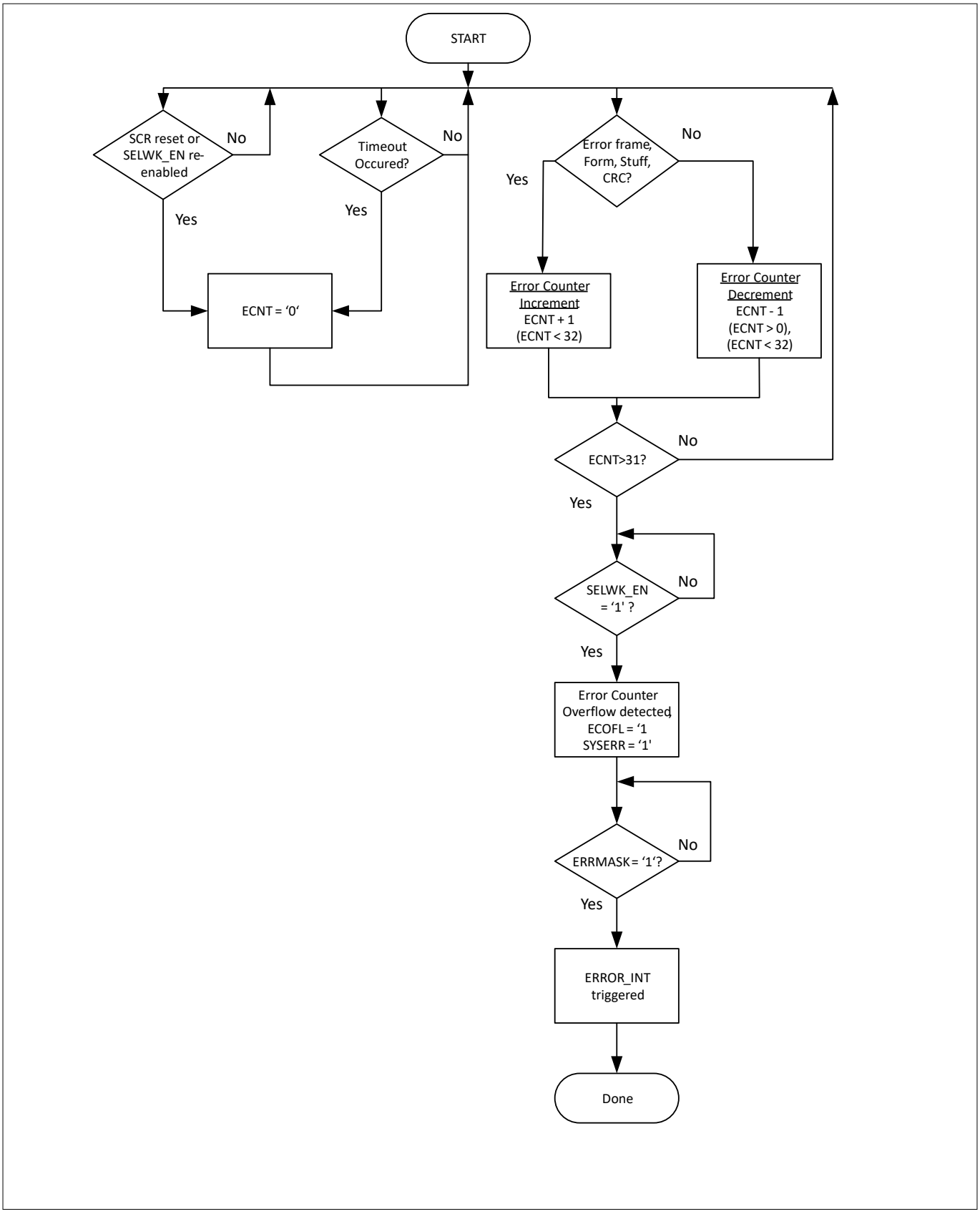


Figure 941 Error counter

39.3.15.3 WUP detection

WCAN supports WUP detection by waking up on a low level detected on RXD when a wake-up is sent on the bus. The WUP is detected by the CAN Transceiver. In case the CAN Transceiver does not support Wake-Up Pattern detection, then WCAN considers any low level on RXD as WUP. In WCAN, the WUP detection is active only when the Selective Wake-Up feature is enabled and CAN timeout has occurred. On occurrence of "CAN Timeout", WCAN enters into a Standby mode.

When WCAN is in Standby Mode:

- The Protocol Handler is reset and Selective Wake-Up feature is disabled
- The Clock to the Protocol Handler is switched off to save power
- WUP detection is active, waiting for a low level on RXD (decoding of the WUP itself should be done by the CAN Transceiver)
- WCAN_INTESTAT0.MODE = 1 and WCAN_INTESTAT1.SWKSET = 0

***Note**: In WCAN Standby mode, the WUP detection is active even when 100 MHz f_{BACK} is switched-off. In such a case, the WUP detection will be performed with 70 kHz clock, but with a risk that the low level might be missed. In-order to save power when WCAN in standby mode, the application software may disable 100 MHz f_{BACK}, and only 70 kHz clock can be provided. The 100 MHz f_{BACK} is automatically turned-on when WUP is detected.*

When WUP is detected in Standby Mode, the WCAN enters into the Normal Mode, where:

- Selective Wake-Up feature is re-enabled
- The clock to the protocol handler is active
- Error Counter is reset to 0
- 100 MHz f_{BACK} enable is requested (SCUSCR_CLK.DIV_REQ is set to 4 (DIV5, 20 MHz)), when 100 MHz f_{BACK} is disabled in Standby Mode
- WCAN_INTESTAT0.MODE = 0, WCAN_INTESTAT1.SWKSET = 1 and WCAN_INTESTAT0.WUP = 1
- WUP_INT is triggered when it is not masked through WCAN_INTMRSLT.WUPMASK

The application software can make WCAN to leave the Standby Mode by SCR Reset or by disabling WCAN (WCAN_CFG.WCAN_EN = 0). The figure below shows the state diagram on transition of WCAN between different modes of operation.

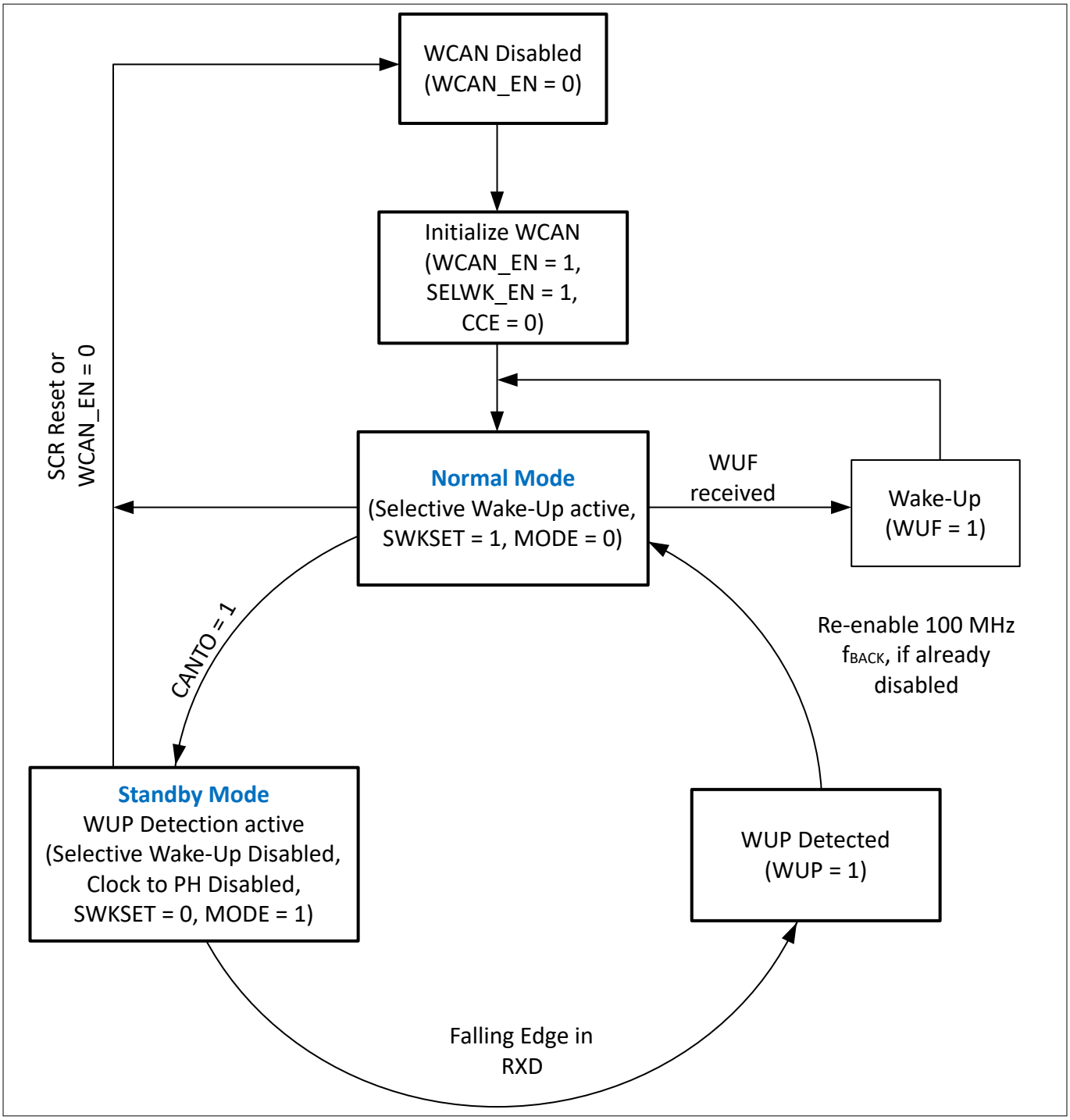


Figure 942 WCAN modes of operation

39.3.15.4 CAN functional description

39.3.15.4.1 CAN node control

The CAN node is equipped with an individual set of SFR registers to control and to monitor the CAN node.

Baudrate pre-scaler

The WCAN has an integer divider with a scaling factor defined by the WCAN_BTL2_CTRL.BRP bit-field. The f_{BACK} is the input frequency and the f_{CAN} is the scaled output frequency of the integer divider. The WCAN_BTL2_CTRL.BRP bit-field is recommended to be configured based on the baudrates required as shown in table 'Bit Timing and RXD Filter configuration - Recommended' (see related information), because WCAN_BTL2_CTRL.BRP values influences the receive input filter time and CAN FD filter time. The f_{CAN} frequency is fed to the Receive Input Filter and CDR (to measure the sample point). When 100 MHz f_{BACK} is switched off, the Selective Wake-Up filter block will be reset and then f_{CAN} is also switched off.

***Note**: The Selective wake-up feature cannot be used for example, if WCAN_INTESTAT1.SWKSET = 0.*

Related information

Filter and sample point calculation on page 6669

Receive input filter

The RXD signal coming from the transceiver, is filtered to eliminate the jitter effects from the CAN bus, The filter time is dependent on the recommended baud rate configuration and the WCAN_BTL2_CTRL.BRP values (table 'Bit Timing and RXD Filter configuration - Recommended' in related information). The filtering is done based on f_{WCAN} frequency. The filtering time T_filt is based on the END COUNT value for different WCAN_BTL2_CTRL.BRP as shown in the related table. The filter has an asymmetrical behavior as shown in the following figure.

The first recessive to dominant edge is detected asynchronously, and the event is used for the starting of the filter counter. Once the counter reaches the END COUNT value, the RXD is checked again if it is still ‘0’, then the synchronization edge is considered valid. Therefore, the filter introduces a delay of T_filt.

At dominant to recessive edge, RXD is checked at each f_{WCAN} clock cycle. The RXD_FILT output is ‘1’ only if the input signal is a stable recessive for entire filtering time (T_filt). The recessive bits shorter than t_glitch are filtered out and only recessive bits longer that t-detect are considered valid.

When CAN FD tolerance is enable, the received input signal RXD is used directly without filtering for the data phase, to avoid the additional filter delay.

The logic level of the filtered input RXD_FILT is shown in WCAN_INTESTAT1.RXDF.

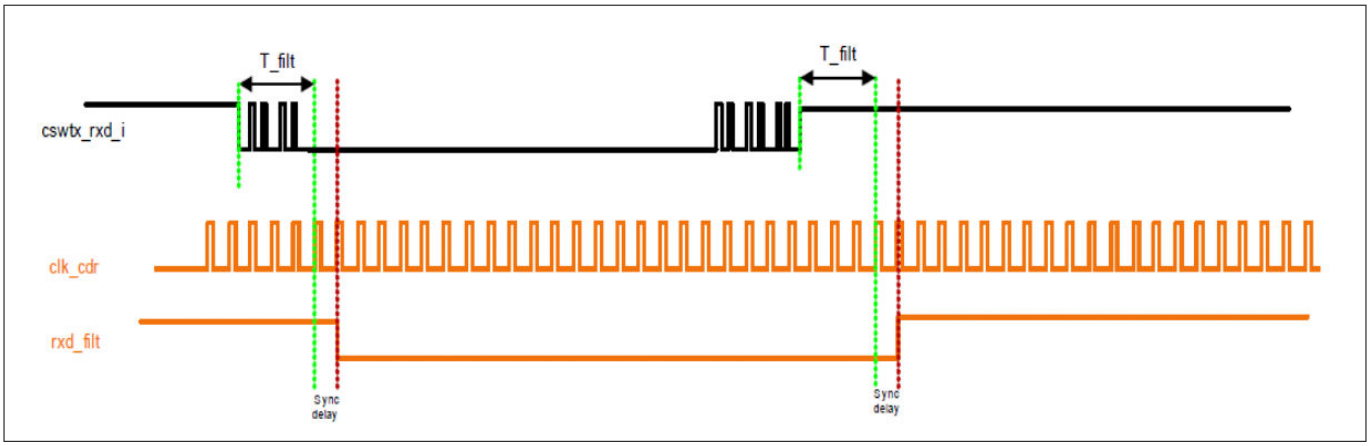


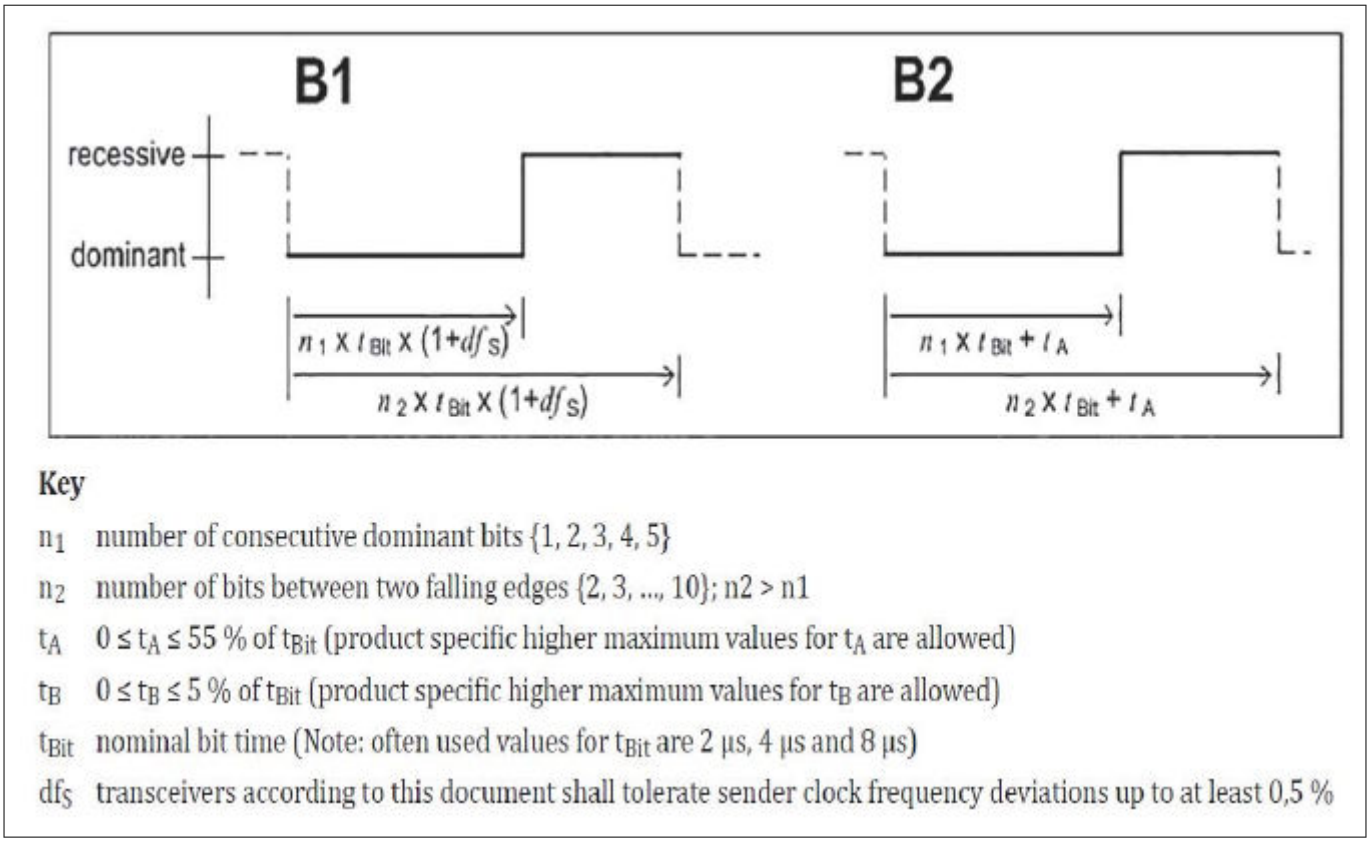
Figure 943 RXD filtering

Related information

Filter and sample point calculation on page 6669

Bit timing

According to ISO 11898-2 (here the ISO11898-6 part) standard, a received classical CAN frame can be decoded correctly when the timing of the CAN bit fields complies with multiple instances of signal shape B1 and one instance of signal shape B2 (to handle the sender clock tolerance and loss of arbitration). In order to compensate possible deviations of the CAN oscillator frequency, integrated clock and data recovery is used (CDR). The CDR can be optionally enabled or disabled using WCAN_CDR_CTRL.CDREN bit-field. It is recommended to always enable the CDR feature. If CDR is disabled, the required oscillator tolerance is 1% or else it is 4%. When CDR is disabled, the sample point position is calculated directly based on BTLn_CTRL (n=1,2) values. Before enabling the CDR, the bit timings parameters have to be properly configured as shown in table 'Bit Timing and RXD Filter configuration - Recommended' in related information. When CAN FD tolerance is enabled WCAN_FD_CTRL.FDEN = 1, the CDR is stopped during CAN FD frame reception, in order to avoid calculation based on high baudrate data phase. The CDR is kept at last calculated value before CAN FD has been detected.



Key

n_1	number of consecutive dominant bits {1, 2, 3, 4, 5}
n_2	number of bits between two falling edges {2, 3, ..., 10}; n2 > n1
t_A	$0 \le t_A \le 55$ % of t_{Bit} (product specific higher maximum values for t_A are allowed)
t_B	$0 \le t_B \le 5$ % of t_{Bit} (product specific higher maximum values for t_B are allowed)
t_{Bit}	nominal bit time (Note: often used values for t_{Bit} are 2 µs, 4 µs and 8 µs)
df_S	transceivers according to this document shall tolerate sender clock frequency deviations up to at least 0,5 %

Figure 944 CAN bit timing signal shape

Clock and Data Recovery (CDR)

The CDR consists of three building blocks as shown in the following figure. The CDR uses the f_{WCAN} clock. This clock period defines the duration of one time quanta. For ISO 11898-2 some settings are guaranteed to be working according to the given propagation delay times (see Network Propagation Delays in related information). Working settings for 125 kBit/s, 250 kBit/s and 500 kBit/s will be provided. The acquisition counter of CDR measures the CAN bit timings in terms of number of time-quanta. The sample point calculation evaluates the new sample point based on the computed CAN bit length and the sample point position configured in WCAN_BTL2_CTRL. The filter block is used to compensate the error introduced during internal measurement of number of time-quanta in one bit time.

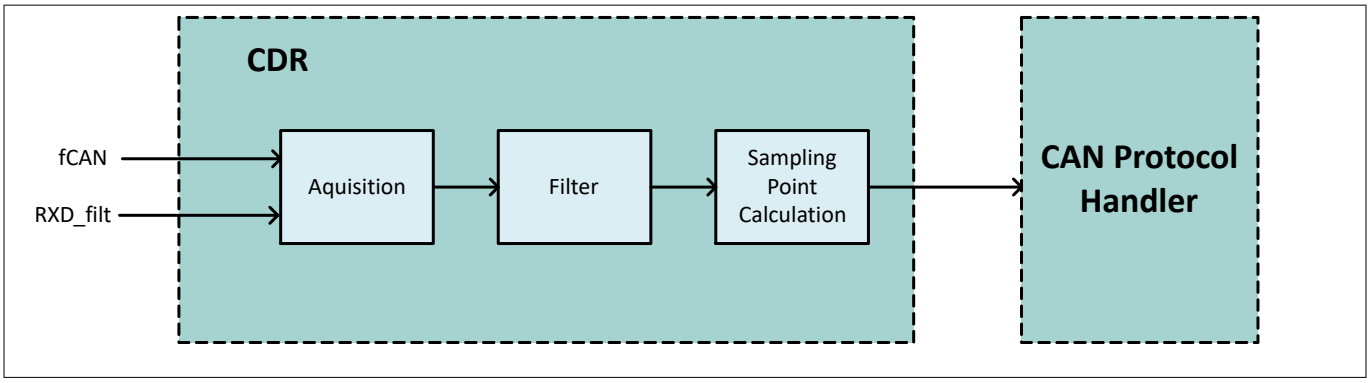


Figure 945 CDR block diagram

The configuration change enable, WCAN_CFG.CCE has to be enabled before a valid CAN bit timing can be written to the CAN Node Bit Timing Register, BTLn_CTRL (n=1-2). This is then followed by resetting WCAN_CFG.CCE bit before enabling the operation of the CAN node.

***Note**: The proposed sample point is at 80% of the bit time.*

Related information

Filter and sample point calculation on page 6669
Network propagation delays on page 6670

Acquisition counter

The acquisition counter measures the bit timing based on the number of time-quanta between two synchronization events (i.e., recessive to dominant transition of the filtered RX signal). The counter is active only if WCAN_CDR_CTRL.CDREN=1 and enabled by the protocol handler. The protocol handler enables the acquisition outside the arbitration phase, where CAN bit shortenings and extension are likely to occur. The only exception to that scenario is when an error is detected in the arbitration phase. In that case, the acquisition is enabled for preventing deadlock situation when the oscillator is outside the range and error occurs during Arbitration. The synchronization edges occurring within an error frame are canceled by the windowing of acquisition. Based on the counter value, the number of CAN bits received between two synchronization events can be determined. This value is between 2 and 10, in accordance to the bit-stuffing rule.

Filter and sample point calculation

The sample point calculation block evaluates the sampling position of the CAN bus bit based on the information from the "Acquisition Counter" and from the CDR configuration registers. Depending on the selected frequency and the baudrate, the default number of actual clock pulses expected during a CAN bus bit time is configured in WCAN_BTL1_CTRL register. The sampling position of the CAN bit is defined in BTL2_CTRL.SP bit-field. It can be calculated as follows

BTL2_CTRL.SP = (64*sampling point)/100

***Note**: For example: A sampling point at 80% of bit time corresponds to WCAN_BTL2_CTRL.SP = 33_H.*

The number of time-quanta in one CAN bit time (tbit_calc) is measured internally based on the output from the acquisition counter and the expected number of time-quanta in one CAN bit time as provided by WCAN_BTL1_CTRL. This measured value can be read from the WCAN_CDR_MEAS_HIGH and WCAN_CDR_MEAS_LOW. The error introduced during the measurement is compensated using the Error scaling factor. During successive CDR measurements, the new bit timing value is evaluated by an averaging algorithm:

tbit_new = tbit_calc + error / SEL_FILT

where SEL_FILT is the error scaling factor (for example the value over how many acquisition cycles the average will be built) as described in Table below.

Table 1727 Number of acquisitions based on ESF

ESF	SEL_FILT
00_B	8
01_B	16 (recommended)
10_B	32
11_B	Adaptive averaging (based on number of bits between two consecutive synchronization edges) 2, 3 or 4 CAN bits = 32 5 or 6 CAN bits = 16 7, 8 or 9 CAN bits = 8

The sample point is then calculated based on WCAN_BTL2_CTRL and tbit_new. The sampling point is calculated only when tbit_new falls within the range defined by WCAN_CDR_LOWER_CTRL.LOWER and WCAN_CDR_UPPER_CTRL.UPPER. If not, the sample point is calculated directly based on the t_{bit} value defined in WCAN_BTL1_CTRL.

Table 1728 Bit-timing and RXD filter configuration - recommended

Baudrate (kBit/s)	BRP	BTL1_CTRL	RXD Filter END COUNT	t_glitch (ns)	t-detect (ns)
1000	00_B	64_H	24	200	400
500	01_B	64_H	16	250	500
250	10_B	64_H	8	250	500
125	11_B	64_H	8	400	1400

Network propagation delays

The “worst case” network propagation delay is the network propagation delay after the 5th bit for the dominant to recessive edge on the bus. (See following figure)

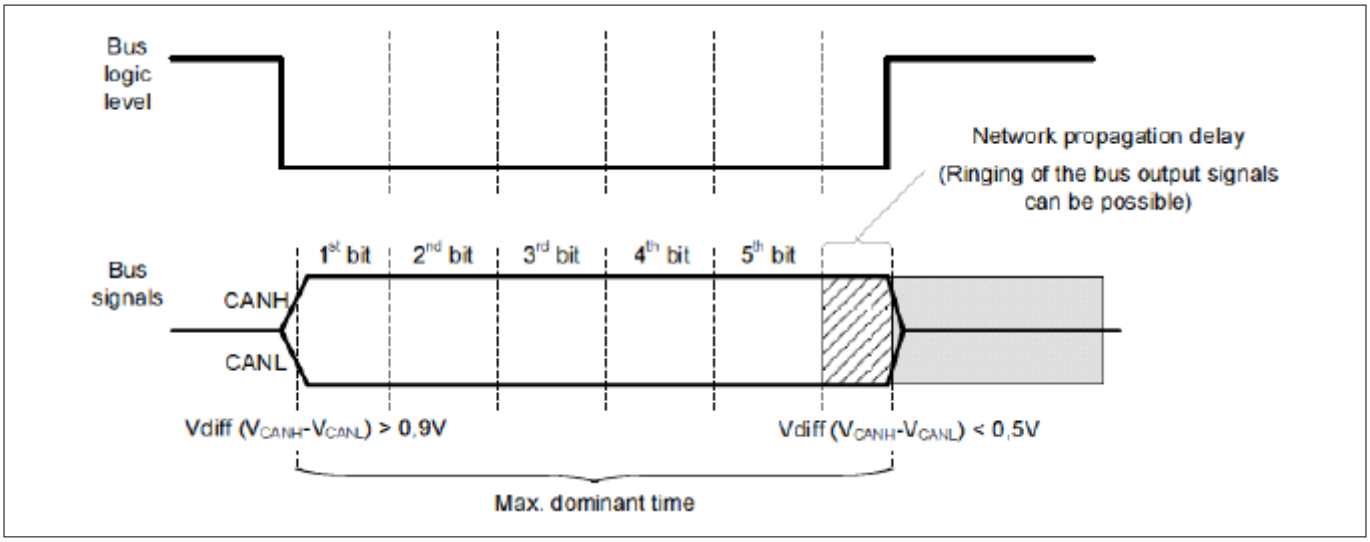


Figure 946 Network propagation delay after 5th bit

The WCAN module will be capable of sampling the recessive stuff bit after 5 consecutive dominant bits to ensure a safe wake-up, even if the dominant to recessive edge is delayed for the value of the network propagation delay shown in the following table. The network propagation delay describes the maximal occurrence propagation delay which can be seen on the bus. This maximum network propagation delay does not include any EMC jitter (i.e no HF injection).

Table 1729 Network propagation delay

Baudrate (Baud)	Network propagation delay (ns)	Maximum dominant time (ns) (sender oscillator tolerance included)
1 MBaud	550	5550
500 kBaud	1350	11350
250 kBaud	2675	22675
125 kBaud	5450	45450

For HF injection the WCAN module is able to handle EMC network propagation delay as shown in following table.

Table 1730 EMC network propagation delay

Baudrate (Baud)	Network propagation delay (ns)	Maximum dominant time (ns) (sender oscillator tolerance included)
1 MBaud	450	5450
500 kBaud	1250	11250
250 kBaud	2575	22575
125 kBaud	5350	45350

Also the WCAN module is able to decode frames that contain sequences of 5 consecutive dominant bits followed by 5 consecutive recessive bits as shown in the previous figure.

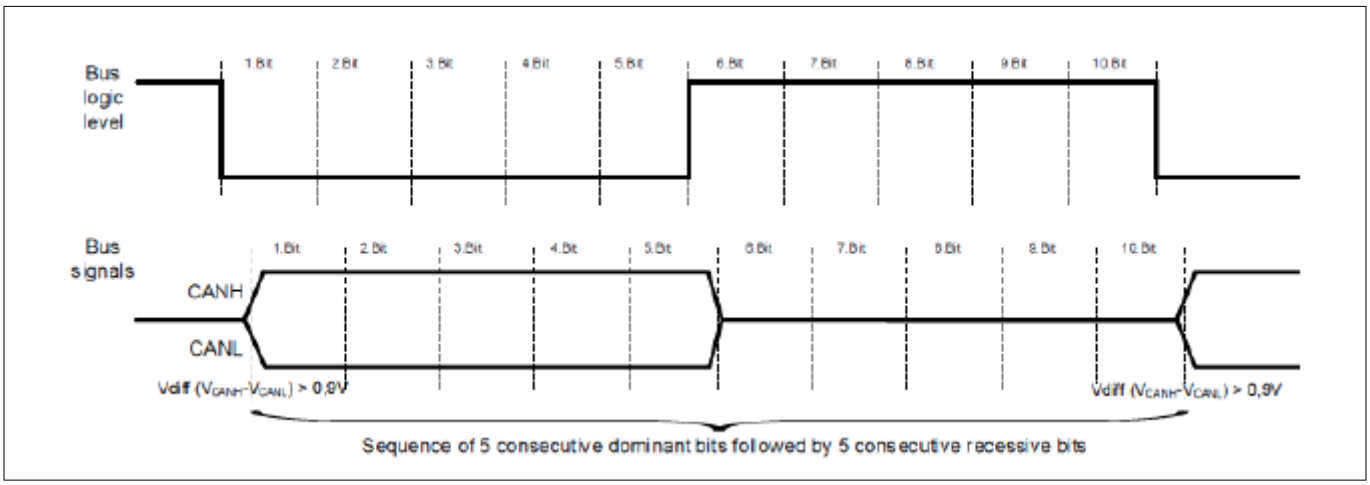


Figure 947 Network propagation delay after 10th bit

The transmitting CAN node on the bus has to guarantee a maximum baudrate tolerance of +/- 0.4% for correct detection of the CAN frames (See following table)

Table 1731 Worst case length of 10 bits

Baudrate (Baud)	Worst case time for length of 10 bits by sender oscillator tolerance (-0.4%)	Worst case time for length of 10 bits by sender oscillator tolerance (+0.4%)
1 MBaud	9960 ns	10040 ns
500 kBaud	19920 ns	20080 ns
250 kBaud	39840 ns	40160 ns
125 kBaud	79680 ns	80320 ns

CAN FD tolerance

The WCAN can be configured as CAN FD tolerant (WCAN_FD_CTRL.FDEN = 1), which means that protocol handler does not consider received CAN FD frames as error condition. When bit WCAN_INTESTAT1.FDF is found dominant, the decoding phase is stopped and the protocol handler goes to the bus integration state waiting for 5 consecutive recessive bits. The protocol handler has an "End Of Frame" (EOF) bit counter to detect 5 consecutive recessive bits. The FD block generates a reset signal (active low) for the EOF bit counter, when a dominant bit in the high bit-rate phase is detected. The FD block directly uses the RXD signal and operates at

f_{BACK} to avoid losing a dominant bit during the high bit-rate phase. Consequently a filter to suppress the dominant glitches is introduced in FD block. Since only dominant detection acts as the reset for EOF bit counter, the filter is an up-counter incremented when the RXD signal is dominant and reset when the RXD is recessive. So the filtered RXD is considered dominant when the counter end value (FD End Count) is reached. The glitch length (t_FD_glitch) to filter out and the minimum dominant pulse to detect (t_FD_detect) are based on the baudrate and the FD factor as shown in the table below.

Table 1732 CAN FD dominant bit filter time

Baudrate (kBaud)	FD factor	FD_FILT	FD end count	t_FD_glitch (ns)	t_FD_detect (ns)
125	4	011	90	400	1400
250	4	010	48	200	700
500	4	001	25	100	350
125	10	010	48	200	700
250	10	001	25	100	350
500	10	000	13	50	175

39.3.15.4.2 WCAN kernel registers

Register address map

The register set of the WCAN module consists of the below distinct subsets:

1. The Configuration, Miscellaneous Registers exist in one instance and apply for the entire WCAN module
2. The Wake-Up CAN registers consist of the following:
   - CAN Node Registers (WCAN_BTL1_CTRL, WCAN_BTL2_CTRL, WCAN_CDR_CTRL, WCAN_CDR_UPPER_CTRL, WCAN_CDR_LOWER_CTRL, WCAN_CDR_MEAS_HIGH, WCAN_CDR_MEAS_LOW and WCAN_FD_CTRL), which applies to a single CAN node
   - WUF configuration registers (DLC_CTRL, IDx_CTRL, MASK_IDx_CTRL and DATAx_CTRL) are defined for a single message object and exist once for each message object

Configuration and status registers

The configuration and status registers of a WCAN node are accessed from standard (non-mapped) SFR area. The following table shows the address as seen on the Standby Controller (SCR).

Table 1733 WCAN configuration and status registers address map

Register	Addr.	Page	SCR reset value	Full name of register
WCAN_CFG	$B0_H$	0	08_H	Configuration Register
WCAN_INTMRSLT	$B1_H$	0	00_H	Interrupt Mask and Reference Select Register
WCAN_INTESTAT0	$B0_H$	1	00_H	Interrupt and Event Status 0 Register
WCAN_INTESTAT1	$B1_H$	1	60_H	Interrupt and Event Status 1 Register
WCAN_FRMERRCNT	$B2_H$	1	00_H	Frame Error Counter Register
WCAN_INTESCLR0	$B3_H$	1	00_H	Interrupt and Event Status Clear Register 0
WCAN_INTESCLR1	$B4_H$	1	00_H	Interrupt and Event Status Clear Register 1

CAN node registers

The registers of a WCAN node are accessed from standard (non-mapped) SFR area. The following table shows the address of registers as seen on the Standby Controller (SCR).

Table 1734 CAN node registers address map

Register	Addr.	Page	SCR reset value	Full name of register
WCAN_BTL1_CTRL	$B5_H$	1	64_H	Bit Timing Logic 1 Control Register
WCAN_BTL2_CTRL	$B6_H$	1	73_H	Bit Timing Logic 2 Control Register
WCAN_CDR_CTRL	$B2_H$	0	05_H	CDR Control Register
WCAN_CDR_UPPER_CTRL	$B3_H$	0	69_H	CDR Upper Limit Control Register
WCAN_CDR_LOWER_CTRL	$B4_H$	0	$5F_H$	CDR Lower Limit Control Register
WCAN_CDR_MEAS_HIGH	$B5_H$	0	64_H	CDR Measured High Register
WCAN_CDR_MEAS_LOW	$B6_H$	0	00_H	CDR Measured Low Register
WCAN_FD_CTRL	$B7_H$	0	00_H	CAN FD Control Register

WUF configuration registers

The registers to configure WUF are accessed from standard (non-mapped) SFR area. The following table shows the address of the WUF configuration Registers as seen on the Standby Controller (SCR).

Table 1735 WUF configuration registers address map

Register	Addr.	Page	SCR reset value	Full name of register
WCAN_DLC_CTRL	$B7_H$	1	00_H	Message Data Length Code Control Register
WCAN_ID0_CTRL	$B0_H$	2	00_H	Message Identifier Control Register 0
WCAN_ID1_CTRL	$B1_H$	2	00_H	Message Identifier Control Register 1
WCAN_ID2_CTRL	$B2_H$	2	00_H	Message Identifier Control Register 2
WCAN_ID3_CTRL	$B3_H$	2	00_H	Message Identifier Control Register 3
WCAN_MASK_ID0_CTRL	$B4_H$	2	00_H	Message Identifier Acceptance Mask Register 0
WCAN_MASK_ID1_CTRL	$B5_H$	2	00_H	Message Identifier Acceptance Mask Register 1
WCAN_MASK_ID2_CTRL	$B6_H$	2	00_H	Message Identifier Acceptance Mask Register 2
WCAN_MASK_ID3_CTRL	$B7_H$	2	00_H	Message Identifier Acceptance Mask Register 3
DATA0_CTRL	$B0_H$	3	00_H	Message Data Control Register 0
DATA1_CTRL	$B1_H$	3	00_H	Message Data Control Register 1
DATA2_CTRL	$B2_H$	3	00_H	Message Data Control Register 2
DATA3_CTRL	$B3_H$	3	00_H	Message Data Control Register 3
DATA4_CTRL	$B4_H$	3	00_H	Message Data Control Register 4
DATA5_CTRL	$B5_H$	3	00_H	Message Data Control Register 5
DATA6_CTRL	$B6_H$	3	00_H	Message Data Control Register 6
DATA7_CTRL	$B7_H$	3	00_H	Message Data Control Register 7

CAN node registers

The CAN node specific registers contain information that is directly related to the operation of the CAN node. The Node Control Register contains basic settings that define the bit timing for the operation of the CAN node.

The Bit Timing Logic Control Registers contain all parameters to setup the bit timing for the CAN transfer.

***Note**: The SCR Firmware (SRC-FW) overwrites the BTL1_CTRL register with 64_H and BTL2_CTRL register with 73_H after reset, to obtain a 500Kbits/s baud rate. Please use the values specified in table 'Bit Timing and RXD Filter configuration - Recommended' in related information for 1Mbits/s (BTL1_CTRL1, BTL2_CTRL), 250Kbits/s (BTL1_CTRL, BTL2_CTRL) and 125Kbits/s (BTL1_CTRL, BTL2_CTRL) baud rate. The Configuration Change Enable, WCAN_CFG.CCE has to be enabled before the BTLn_CTRL register can be written.*

Related information

Filter and sample point calculation on page 6669

39.3.15.5 WCAN connections to GPIO

The port input/output control registers contain the bit fields that selects the digital output and input driver characteristics such as pull-up/down devices, port direction (input/output), open-drain, and alternate output selections. The I/O lines for the WCAN module are controlled by the port input/output control registers shown below.

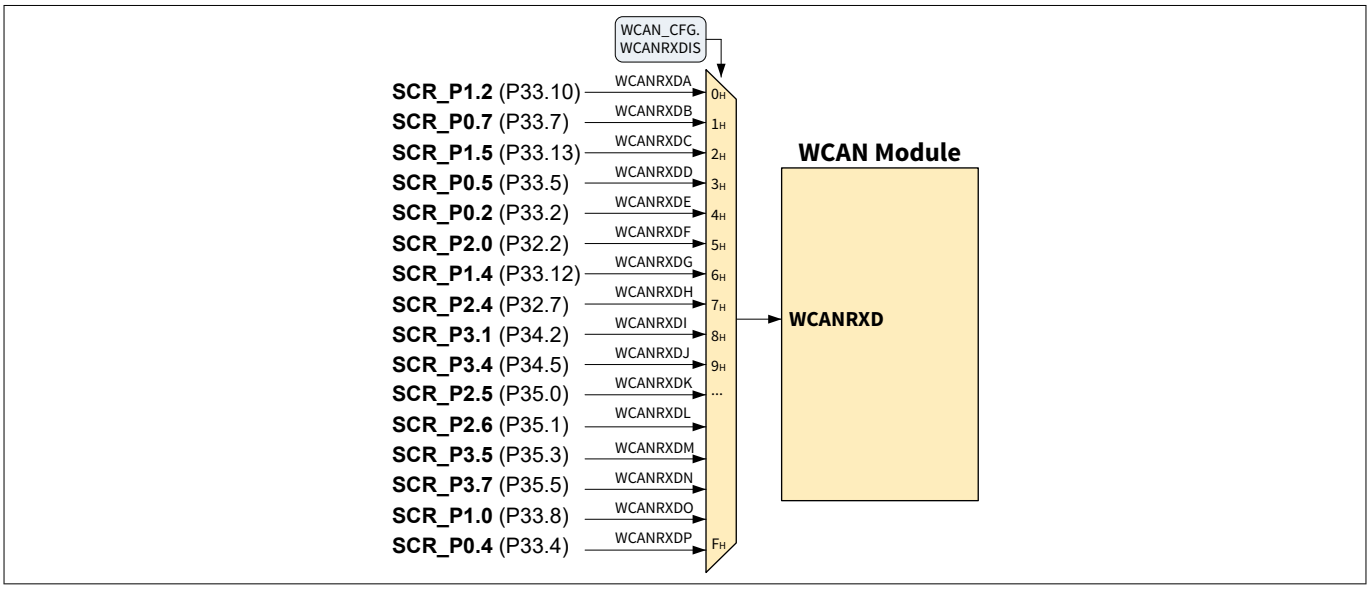


Figure 948 Wake-up CAN connections to GPIO ports

39.3.15.6 WCAN interrupts

The interrupt control logic in the Wake-Up CAN module consists of the following interrupts shown in the table and figure below.

Table 1736 WCAN interrupt outputs

Interrupt inputs	Connected to WCAN interrupt output
SCR Interrupt Node 6	WUF_INT (Wake-Up Frame Interrupt)
	ERROR_INT (System Error Interrupt)

(table continues...)

Table 1736 (continued) WCAN interrupt outputs

Interrupt inputs	Connected to WCAN interrupt output
	CANTO_INT (CAN Timeout Interrupt)
	WUP_INT (WUP Detected Interrupt)

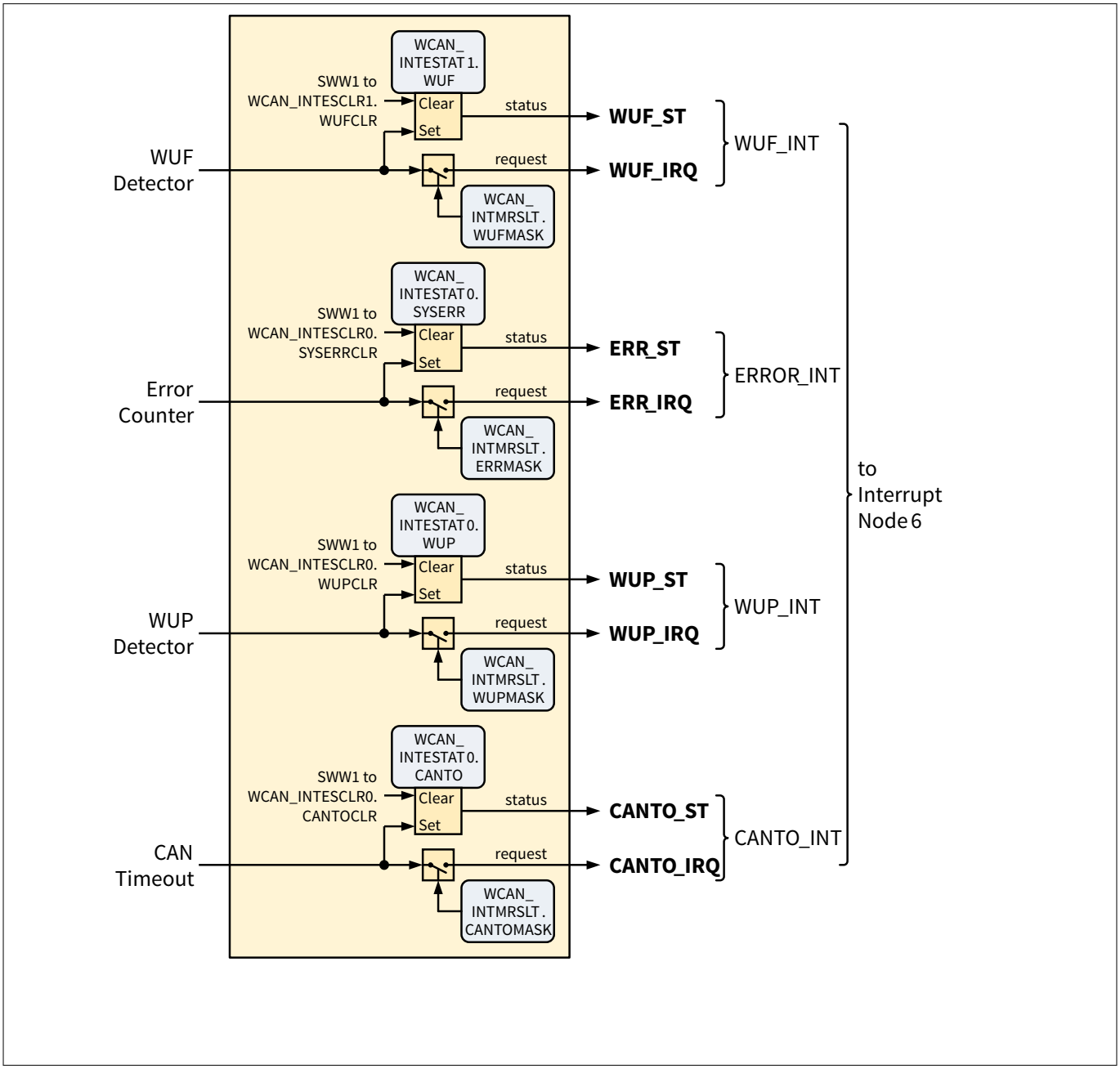


Figure 949 Wake-up CAN interrupt outputs

39.3.16 Inter-Integrated Circuit (I2CSCR)

The I2CSCR provides an interface between the microcontroller and an I2C bus that conforms to the I2C Bus Protocol 1.0.

The can operate in either master or slave mode. It performs arbitration in master mode to allow it to operate in multi-master systems. The I2CSCR provides communication at data rates of up to 400 KBit/s and features 7-bit addressing as well as 10-bit addressing.

39.3.16.1 Features

- Operates in Master or slave mode
- Supports multi-master systems
- Performs arbitration and clock synchronization
- Supports 7-bit and 10-bit addressing modes
- Selectable baud rate generation of up to 400 kBaud (fast mode)
- Flexible control through interrupt service routines or polling
- Clock stretching

39.3.16.2 System information

This section provides system information relevant to the I2CSCR.

39.3.16.2.1 Pinning

The I2CSCR pin assignment for SCR is shown in the following table. Please also see the General purpose I/O ports (PORT8) chapter.

Table 1737 I2C pin functions in SCR

Pin	Function	Description	Selected by
SCR_P1.0 (P33.8)	I2CSDAD	I2C data input	I2CSCR_SRST.SDASEL
SCR_P1.0 (P33.8)	I2CSDA	I2C data output	SCR_P1_IOCR0.PC
SCR_P1.1 (P33.9)	I2CSCLD	I2C clock input	I2CSCR_SRST.SCLSEL
SCR_P1.1 (P33.9)	I2CSCL	I2C clock output	SCR_P1_IOCR1.PC
SCR_P2.1 (P32.4)	I2CSCLA	I2C clock input	I2CSCR_SRST.SCLSEL
SCR_P2.1 (P32.4)	I2CSCL	I2CSCR clock output	SCR_P2_IOCR1.PC
SCR_P2.3 (P32.6)	I2CSDAA	I2C data input	I2CSCR_SRST.SDASEL
SCR_P2.3 (P32.6)	I2CSDA	I2CSCR data output	SCR_P2_IOCR3.PC
SCR_P2.5 (P35.0)	I2CSDAC	I2CSCR data input	I2CSCR_SRST.SDASEL
SCR_P2.5 (P35.0)	I2CSDA	I2CSCR data output	SCR_P2_IOCR5.PC
SCR_P2.6 (P35.1)	I2CSCLC	I2CSCR clock input	I2CSCR_SRST.SCLSEL
SCR_P2.6 (P35.1)	I2CSCL	I2CSCR clock output	SCR_P2_IOCR6.PC
SCR_P3.1 (P34.2)	I2CSDAB	I2CSCR data input	I2CSCR_SRST.SDASEL
SCR_P3.1 (P34.2)	I2CSDA	I2CSCR data output	SCR_P3_IOCR1.PC
SCR_P3.2 (P34.3)	I2CSCLB	I2CSCR clock input	I2CSCR_SRST.SCLSEL
SCR_P3.2 (P34.3)	I2CSCL	I2CSCR clock output	SCR_P3_IOCR2.PC

39.3.16.2.2 Output pin configuration

The pin drivers that are assigned to the I2C lines provide open drain outputs. This ensures that the I2CSCR module does not put any load on the I2C bus lines while the I2CSCR module is not powered. Therefore, it is necessary to have external pull-up resistors (approximately 10 KΩ for operation at 100 KBaud, 2 KΩ for operation at 400 KBaud) on the I2C bus lines.

All pins of the I2CSCR module that are to be used for I2C bus communication must be switched to output and their alternate function must be enabled, before any communication can be established.

If not driven by the I2CSCR module, the pins will switch off their drivers (i.e. driving 1 to an open drain output). Due to the external pull-up devices, the respective bus levels will then be 1, which is idle.

39.3.16.2.3 Clocking configuration

The I2CSCR runs on the PCLK. If the I2CSCR functionality is not required at all, it can be completely disabled by gating off its clock input for maximal power reduction.

This is done by setting bit SCUSCR_PMCON1.I2C_DIS.

The bit SCUSCR_PAGE.PAGE must be programmed before accessing the SCUSCR_PMCON1 register.

39.3.16.2.4 Interrupt events and assignment

Following table lists the interrupt event source from the SCR_I2C, and the corresponding event interrupt enable and flag bit. See chapter Interrupt system for more information related to the interrupts.

Table 1738 SCR_I2C Interrupt Events

Event	Event interrupt enable bit	Event flag bit
SCR_I2C interrupt	SCR_I2C_CNTR.IEN	SCR_I2C_CNTR.IFLG

The table below shows the interrupt node assignment for the SCR_I2C interrupt source.

Table 1739 SCR_I2C Events' Interrupt Node Control

Event	Interrupt node	Vector address	Interrupt node enable bit
SCR_I2C Interrupt	XINTR8	0043_H	IEN1.EX2

Status code

The state of the I2CSCR module is defined by the 5-bit status code in SCR_I2CSCR_STAT register.

When SCR_I2CSCR_STAT contains the status code $F8_H$, no relevant status information is available, no interrupt is generated and the I2CSCR_CNTR.IFLG bit is not set. All other status codes correspond to a defined state of the I2CSCR module. When any of these states is entered, the corresponding status code appears in this register and the I2CSCR_CNTR.IFLG bit is set. When the I2CSCR_CNTR.IFLG bit is cleared, the status code returns to $F8_H$.

Table 1740 Status code

Value of I2CSCR_STAT register	Status
00_H	Bus error
08_H	START condition transmitted
10_H	Repeated START condition transmitted

(table continues...)

Table 1740 (continued) Status code

Value of I2CSCR_STAT register	Status
18_H	Address and write bit transmitted, ACK received
20_H	Address and write bit transmitted, ACK not received
28_H	Data byte transmitted in master mode, ACK received
30_H	Data byte transmitted in master mode, ACK not received
38_H	Arbitration lost in address or data byte
40_H	Address and read bit transmitted, ACK received
48_H	Address and read bit transmitted, ACK not received
50_H	Data byte received in master mode, ACK transmitted
58_H	Data byte received in master mode, ACK not transmitted
60_H	Slave address and write bit received, ACK transmitted
68_H	Arbitration lost in address as master, slave address and write bit received, ACK transmitted
70_H	General call address received, ACK transmitted
78_H	Arbitration lost in address as master, general call address received, ACK transmitted
80_H	Data byte received after slave address received, ACK transmitted
88_H	Data byte received after slave address received, ACK not transmitted
90_H	Data byte received after general call address received, ACK transmitted
98_H	Data byte received after general call address received, ACK not transmitted
$A0_H$	STOP or repeated START mode received in slave mode
$A8_H$	Slave address and read bit received, ACK transmitted
$B0_H$	Arbitration lost in address as master, slave address and read bit received, ACK transmitted
$B8_H$	Data byte transmitted in slave mode, ACK received
$C0_H$	Data byte transmitted in slave mode, ACK not received
$C8_H$	Last byte transmitted in slave mode, ACK received
$D0_H$	Second address byte and write bit transmitted, ACK received
$D8_H$	Second address byte and write bit transmitted, ACK not received
$E0_H$	Unused

(table continues...)

Table 1740 (continued) Status code

Value of I2CSCR_STAT register	Status
$E8_H$	Unused
$F0_H$	Unused
$F8_H$	No relevant status information, IFLG = 0

If an illegal condition occurs on the I2C bus, the bus error state is entered (status code 00_H). To recover from this state, the I2CSCR_CNTR.STP bit must be set and the I2CSCR_CNTR.IFLG bit cleared. The I2CSCR module will then return to idle state (status code $F8_H$): no STOP condition will be transmitted on the I2C bus.

To request resumption of transmission, set the I2CSCR_CNTR.STA bit to 1 at the same time as the I2CSCR_CNTR.STP bit is set. The I2CSCR module will then send a START on recovery from the bus error.

39.3.16.2.5 Baudrate generation

Baudrate is measured in KBit/s. Therefore all units will be called KBit/s.

To support a wide range of input clock frequencies (f_{PCLK}), two dividers are implemented to generate the required bit-rate on the I2C bus in master mode. The two dividers are defined by the I2CSCR_BRCR.PREDIV and I2CSCR_BRCR.BRP bits.

***Note**: An operation of the I2CSCR module with the SCR 70 kHz clock source is not recommended as due to the low module operating frequency a proper bus communication cannot be guaranteed.*

The bit-rate is calculated by the formula:

$$f_{OCSL} = \frac{f_{PCLK}}{2^{PREDIV} \times (BRP + 1) \times 10} \quad (70)$$

Bit-rate

***Notes**:*
Due to internal synchronization mechanisms this formula is not valid for the following settings of PREDIV and BRP. Instead the following formulas have to be used:

1. *For PREDIV = 0 and BRP = 0 formula is: $f_{OCSL} = f_{PCLK}/15$*
2. *For PREDIV = 1 and BRP = 0 formula is: $f_{OCSL} = f_{PCLK}/26$*
3. *For PREDIV = 0 and BRP = 1 formula is: $f_{OCSL} = f_{PCLK}/24$*
4. *For PREDIV = 1 and BRP = 1 formula is: $f_{OCSL} = f_{PCLK}/48$*
5. *For PREDIV = 0 and BRP = 2 formula is: $f_{OCSL} = f_{PCLK}/36$*

The following table shows some examples of the bit-rate together with the required dividers setting.

Table 1741 Bit-rate Selection

	Standard Mode			Fast Mode		
f_{PCLK}	PREDIV	BRP+1	f_{SCL}	PREDIV	BRP+1	f_{SCL}
20 MHz	2	5	100 kHz	1	3	333.3 kHz
33.3 MHz	2	9	92.5 kHz	1	5	333 kHz
50 MHz	2	13	96.15 kHz	1	7	357.14 kHz
100 MHz	4	7	89.2 kHz	1	13	384.16 kHz

The frequency at which the I2C bus is sampled is calculated by the formula:

$$f_{SAMP} = \frac{f_{PCLK}}{2^{PREDIV}} \quad (71)$$

Sampling frequency

To ensure correct detection of START and STOP conditions on the bus, the I2CSCR module must sample the I2C bus at least twelve times faster than the bus clock speed of the fastest master on the bus. The sampling frequency should therefore be at least 1.2 MHz in standard mode or 4.8 MHz in fast mode to guarantee correct operation with other bus masters. In order to ensure a functional communication following per-requisites have to be fulfilled:

1. SCL low ≥ 6 x T_{SAMP}
2. SCL high ≥ 6 x T_{SAMP}
3. min setup/hold times ≥ 2 x T_{SAMP}

39.3.16.2.6 Clock synchronization

If another device on the I2C bus drives the clock line when the SCR_I2C is in master mode, the SCR_I2C will synchronize its clock to the I2C bus clock. The high period of the clock will be determined by the device that generates the shortest high clock period. The low period of the clock will be determined by the device that generates the longest low clock period.

When the I2CSCR module is in master mode and is communicating with a slow slave, the slave may stretch each bit period by holding the SCL line low until it is ready for the next bit. The I2CSCR module will automatically re-synchronize as described above.

When the I2CSCR module is in slave mode, it will hold the SCL line low after each byte has been transferred until I2CSCR_CNTR.IFLG has been cleared.

39.3.16.2.7 Bus arbitration

In master mode, the SCR_I2C will check that each transmitted logic 1 appears on the SCR_I2C bus as a logic 1. If another device on the bus over-rules and pulls the SDA line low, arbitration is lost. If arbitration is lost during the transmission of a data byte or a Not-Acknowledge bit, the SCR_I2C will return to idle state. If arbitration is lost during the transmission of an address, the SCR_I2C will switch to slave mode so that it can recognize its own slave address or the general call address.

39.3.16.2.8 Software reset

A software reset may be applied to the I2CSCR module by writing '1' to the bit-field I2CSCR_SRST.SRST. This sets the I2CSCR module back to idle (I2CSCR_STAT set to $F8_H$) and sets the STP, STA and IFLG bits of the I2CSCR_CNTR register to 0. Also the I2CSCR_BRCR is reset by a software reset.

39.3.16.2.9 Operating modes

All operating modes of the I2C module require the I2CSCR_CNTR.ENAB bit to be set to 1.

Master Transmit

In the master transmit mode, the I2CSCR module transmits a number of bytes to a slave receiver. The master transmit mode is entered by setting the I2CSCR_CNTR.STA bit to ‘1’. The I2CSCR module will then test the I2C bus and will transmit a START condition when the bus is free. When a START condition has been transmitted,

the I2CSCR_CNTR.IFLG bit will be set and the status code in the I2CSCR_STAT register will be 08_H. Before this interrupt is serviced, the I2CSCR_DATA register must be loaded with either a 7-bit slave address or the first part (containing 5 fixed bits that is 11110_B and 2 uppermost bits of the 10-bit address) of a 10-bit slave address, with the LSB cleared to '0' (for example with an added Write bit) to specify transmit mode. The I2CSCR_CNTR.IFLG bit should now be cleared to '0' to prompt the transfer to continue.

After the 7-bit slave address (or the first part of a 10-bit address) plus the write bit have been transmitted, I2CSCR_CNTR.IFLG will be set again. A number of status codes is possible in the I2CSCR_STAT register, see below:

Table 1742 Status code after address is transmitted in master transmit mode

Code	I2C (bus) state	CPU response	Next I2C module action
18_H	Addr + W, ACK received	**For 7-bit address:** Write byte to I2C_DATA, clear IFLG Or set STA, clear IFLG Or set STP, clear IFLG Or set STA and STP, clear IFLG **For 10-bit address:** Write extended address byte to I2C_DATA, clear IFLG	 Transmit data byte, receives ACK Transmit repeated START Transmit STOP Transmit STOP then START Transmit extended address byte
20_H	Addr + W, ACK not received	Set STA, clear IFLG Or set STP, clear IFLG Or set STA and STP, clear IFLG	Transmit repeated START Transmitted STOP Transmit repeated START then STOP
38_H	Arbitration lost	Clear IFLG Or set STA, clear IFLG	Return to idle state Transmit START when bus is free
68_H	Arbitration lost, SLA + W received, ACK transmitted	Clear IFLG, AAK=0 Or clear IFLG, AAK=1	Receive data byte, transmit not ACK Receive data byte, transmit ACK
78_H	Arbitration lost, general call address received, ACK transmitted	Same as for code 68_H	Same as for code 68_H
$B0_H$	Arbitration lost, SLA + R received, ACK transmitted	Write byte to I2C_DATA, clear IFLG, AAK=0 Or write byte to I2C_DATA, clear IFLG, AAK=1	Transmit last byte, receive ACK Transmit data byte receive ACK

If 10-bit addressing is being used, then after the first part of a 10-bit address plus the Write bit have been successfully transmitted, the status code will be 18_H or 20_H.

After this interrupt has been serviced and the second part (lower byte) of the address transmitted, the I2CSCR_STAT register will contain one of the following codes:

Table 1743 Status code after second address byte (for 10-bit addressing) is transmitted in master transmit mode

Code	I2C(bus) state	CPU response	Next I2C module action
38_H	Arbitration lost	Clear IFLG Or set STA, clear IFLG	Return to idle state Transmit START when bus is free
68_H	Arbitration lost, SLA + W received, ACK transmitted	Clear IFLG, AAK = 0 Or clear IFLG, AAK = 1	Receive data byte, transmit not ACK Receive data byte, transmit ACK
$B0_H$	Arbitration lost, SLA + R received, ACK transmitted	Write byte to I2C_DATA, clear IFLG, AAK = 0 Or write byte to I2C_DATA, clear IFLG, AAK=1	Transmit last byte, receive ACK Transmit data byte, receive ACK
$D0_H$	Second address byte + W transmitted, ACK received	Write byte to I2C_DATA, clear IFLG Or set STA, clear IFLG Or set STP, clear IFLG Or set STA and STP, clear IFLG	Transmit data byte, receives ACK Transmit repeated START Transmitted STOP Transmitted STOP then START
$D8_H$	Second address byte + W transmitted, ACK not received	Set STA, clear IFLG Or set STP, clear IFLG Or set STA and STP, clear IFLG	Transmit repeated START Transmitted STOP Transmit repeated START then STOP

If a repeated START condition has been transmitted, the status code will be 10_H instead of 08_H. After each data byte has been transmitted, I2CSCR_CNTR.IFLG will be set and one of three status codes will be in the I2CSCR_STAT register:

Table 1744 Status code after data is transmitted in master transmit mode

Code	I2C (bus) state	CPU response	Next I2C module action
28_H	Data byte transmitted, ACK received	Write byte to I2C_DATA, clear IFLG Or set STA, clear IFLG Or set STP, clear IFLG Or set STA and STP, clear IFLG	Transmit data byte, receives ACK Transmit repeated START Transmit STOP Transmit STOP then START
30_H	Data byte transmitted, ACK not received	Same as for code 28_H	Same as for code 28_H
38_H	Arbitration lost	Clear IFLG Or set STA, clear IFLG	Return to idle state Transmit START when bus is free

When all bytes have been transmitted, the STP bit should be set by writing a 1 to this bit in the I2CSCR_CNTR register. The I2CSCR module will then transmit a STOP condition, clear the I2CSCR_CNTR.STP bit and return to idle state (status code $F8_H$).

Master Receive

In the master receive mode, the SCR_I2C will receive a number of bytes from a slave transmitter. After the START condition has been transmitted, the I2CSCR_CNTR.IFLG bit will be set and status code 08_H will be in the I2CSCR_STAT register. The I2CSCR_DATA register should now be loaded with the slave address (or the first part of a 10-bit slave address), with the LSB set to '1' to signify a Read. The I2CSCR_CNTR.IFLG bit should now be cleared to '0' to prompt the transfer to continue. When the 7-bit slave address (or the first part of a 10-bit address) and the Read bit have been transmitted, the I2CSCR_CNTR.IFLG bit will be set again. A number of status codes are possible in the I2CSCR_STAT register:

Table 1745 Status code after address is transmitted in master receive mode

Code	I2C (bus) state	CPU response	Next I2CSCR module action
40_H	Addr + R transmitted, ACK received	Clear IFLG, AAK = 0 Or clear IFLG, AAK = 1	Receive data byte, transmit NACK Receive data byte, transmit ACK
48_H	Addr + R transmitted, ACK not received	Set STA, clear IFLG Or set STP, clear IFLG Or set STA and STP, clear IFLG	Transmit repeated START Transmit STOP Transmit STOP then START
38_H	Arbitration lost	Clear IFLG Or set STA, clear IFLG	Return to idle state Transmit START when bus is free
68_H	Arbitration lost, SLA + W received, ACK transmitted	Clear IFLG, AAK = 0 Or clear IFLG, AAK = 1	Receive data byte, transmit NACK Receive data byte, transmit ACK
78_H	Arbitration lost, general call address received, ACK transmitted	Same as for code 68_H	Same as for code 68_H
$B0_H$	Arbitration lost, SLA + R received, ACK transmitted	Write byte to DATA, clear IFLG, AAK = 0 Or write byte to DATA,clear IFLG, AAK = 1	Transmit last byte, receive ACK Transmit data byte, receive ACK

If 10-bit addressing is being used, the slave is first addressed using the full 10-bit address plus the Write bit. The master then issues a restart followed the first part of the 10-bit address again, but plus the Read bit – after which the status code will be 40_H or 48_H. It is the responsibility of the slave to remember that it had been selected prior to the restart.

If a repeated START condition has been transmitted, the status code will be 10_H instead of 08_H.

After each data byte has been received, I2CSCR_CNTR.IFLG will be set and one of three status codes will be in the I2CSCR_STAT register:

Table 1746 Status code after data is received in master receive mode

Code	I2C (bus) state	CPU response	Next I2CSCR module action
50_H	Data byte received,ACK transmitted	Read DATA, clear IFLG,AAK = 0 Or read DATA, clear IFLG, AAK = 1	Receive data byte, transmit NACK Receive data byte, transmit ACK
58_H	Data byte received, NACK transmitted	Read DATA, set STA, clear IFLG Or read DATA, set STP, clear IFLG Or read DATA, set STA and STP, clear IFLG	Transmit repeated START Transmit STOP Transmit STOP then START
38_H	Arbitration lost in NACK bit	Clear IFLG Or set STA, clear IFLG	Return to idle state Transmit START when bus is free

When all bytes have been received, a NACK should be transmitted then the I2CSCR_CNTR.STP bit should be set by writing a '1' to this bit. The SCR_I2C will transmit a STOP condition, clear the I2CSCR_CNTR.STP bit and return to idle state (status code $F8_H$).

Slave transmit

In the slave transmit mode, a number of bytes are transmitted to a master receiver. For the I2CSCR module to respond, the I2CSCR_CNTR.AAK bit needs to be set.

The I2CSCR module will enter slave transmit mode when it receives its own slave address and a "Read bit" after a START condition. The I2CSCR module will then transmit an acknowledge bit and set the I2CSCR_CNTR.IFLG bit. The I2CSCR_STAT register will contain the status code $A8_H$.

***Note**: If the I2CSCR module has an extended slave address (signified by 11110_B in I2CSCR_ADDR[7:3]), it will first be selected, then there will be a restart followed by another address byte. If this address byte matches the value stored in I2CSCR_ADDR, the I2CSCR module will transmit an acknowledge after this address byte is received. An interrupt will be generated, I2CSCR_CNTR.IFLG will be set and the status will be $A8_H$. No second address byte will be sent by the master: it is up to the slave to remember that it had been selected prior to the restart.*

The "Slave transmit mode" can also be entered directly from a master mode if arbitration is lost in master mode during the transmission of an address and the slave address and "Read bit" are received. The status code in the I2CSCR_STAT register will then be $B0_H$.

The data byte to be transmitted should then be loaded into the I2CSCR_DATA register and I2CSCR_CNTR.IFLG cleared. When the I2CSCR module has transmitted the byte and received an acknowledge, I2CSCR_CNTR.IFLG will be set and the I2CSCR_STAT register will contain $B8_H$. Once the last byte to be transmitted has been loaded into the I2CSCR_DATA register, the I2CSCR_CNTR.AAK bit should be cleared when I2CSCR_CNTR.IFLG is cleared. After the last byte has been transmitted, I2CSCR_CNTR.IFLG will be set and the I2CSCR_STAT register will contain $C8_H$. The I2CSCR module will then return to idle state (status code $F8_H$). The I2CSCR_CNTR.AAK bit must be set to 1 before slave mode can be entered again.

If no acknowledge is received after transmitting a byte, I2CSCR_CNTR.IFLG will be set and the I2CSCR_STAT register will contain $C0_H$. The I2CSCR module will then return to idle state.

If the STOP condition is detected after an acknowledge bit, the I2CSCR module will return to idle state.

Slave receive

In the slave receive mode, a number of data bytes are received from a master transmitter.

The I2CSCR module will enter slave receive mode when it receives its own slave address and a Write bit (LSB=0) after a START condition. The I2CSCR module will then transmit an acknowledge bit and set the I2CSCR_CNTR.IFLG bit: the I2CSCR_STAT register will then contain status code 60_H. The I2CSCR module will also enter slave receive mode when it receives the general call address 00_H (if the I2CSCR_ADDR.GCE bit is set). The status code will then be 70_H.

Note: Where the I2CSCR module has an extended slave address (signified by 11110_B in I2CSCR_ADDR[7:3]), it will transmit an acknowledge after the first address byte is received but no interrupt will be generated, I2CSCR_CNTR.IFLG will not be set and the status will not change. Only after the second address byte has been received the I2CSCR module will generate an interrupt, set the I2CSCR_CNTR.IFLG bit and the status code in I2CSCR_STAT as described above.

Slave receive mode can also be entered directly from a master mode if arbitration is lost in master mode during the transmission of an address and the slave address and Write bit (or the general call address if bit I2CSCR_ADDR.GCE is set to 1) are received. The status code in the I2CSCR_STAT register will be 68_H if the slave address was received or 78_H if the general call address was received. The I2CSCR_CNTR.IFLG bit must be cleared to '0' to allow the data transfer to continue.

If the I2CSCR_CNTR.AAK bit is set to 1, then after each byte is received, an acknowledge bit (low level on SDA) is transmitted and the I2CSCR_CNTR.IFLG bit is set: the I2CSCR_STAT register will then contain status code 80_H (or 90_H if slave receive mode was entered with the general call address). The received data byte can be read from the I2CSCR_DATA register and the I2CSCR_CNTR.IFLG bit must be cleared to allow the transfer to continue. When the STOP condition or a repeated START condition is detected after the acknowledge bit, then the I2CSCR_CNTR.IFLG bit is set and the I2CSCR_STAT register will contain status code $A0_H$.

***Note**: In case of a general call, when the general call address was received the receive input buffer is reset.*

If the I2CSCR_CNTR.AAK bit is cleared to 0 during a transfer, a NAK (not acknowledge; high level on SDA) is sent after receiving the next byte. The I2CSCR_CNTR.IFLG bit is set. The I2CSCR_STAT register will contain status code 88_H (or 98_H if slave receive mode was entered with the general call address). When the I2CSCR_CNTR.IFLG bit has been cleared to 0, the I2CSCR module will return to idle state (status code $F8_H$).

39.3.17 Interface between PMS and SCR

The 8 bit Standby controller (SCR) sub-system is integrated in the standby domain of the PMS subsystem and several control functions are available through PMS registers.

Main functionality

The SCR is enabled through the PMS SCR_CON0.SCREN.

If SCR is disabled through PMS SCR_CON0.SCREN bit, it is ensured that the SCR 100 MHz clock request, pending requests from SCR wake-up sources, and other SCR interfaces do not have any effect on the main system.

After start-up, before enabling SCR, CPU0 must program the SCR by copying code into internal SCR XRAM memory.

The SCR XRAM is accessible from the TriCore™ domain through the FPI interface. Simultaneous access to XRAM through FPI interface and the SCR is arbitrated with SCR having default priority for XRAM access. In the case of wake-up from Standby, it is ensured that SCR XRAM is not re-initialized.

Clock

The 20 MHz stand-by clock source, derived from the 100 MHz back-up clock, is the default SCR clock active in Standby Mode. The SCR clock source may be switched to the internal low-power 70 kHz clock source by the SCR subsystem through the SCUSCR_CLK.DIV_REQ bit. A watchdog timer unit ensures that the clock received after the request is adequate for reliable operation.

The following clock interfaces are forwarded from PMS to SCR:

- 70 kHz clock input
- 100 MHz clock input
- XTAL3/4 32 kHz crystal oscillator clock to be used as SCR RTCSCR input

A signal with information indicating if a 100 MHz clock is available or not, is forwarded and is used to generate the clock for the SCR Arbiter unit.

Reset

SCR Main Reset is asserted in the case of warm PORST, depending on PORSTREQ configuration, or on a SCRSTREQ request. In the case of cold PORST, the PORST pin is asserted which in turn triggers SCR Main Reset depending on the PORSTREQ configuration. For LVD reset, a complete power-on reset of SCR is carried out.

***Note**: When VDDEXT is switched off during standby, the PORST signal is isolated to 0 (active). Therefore if VDDEXT is not supplied in STANDBY mode then PORSTREQ must be disabled to avoid the spurious reset trigger due to PORST assertion.*

After an SCR reset release, the SCR boot firmware initializes the SCR subsystem based on the hardware configuration programmed in PMS SCR_CON0.SCRCFG bits.

Critical SCR errors and events such as XRAM ECC errors, an SCR watchdog overflow event, SCR reset from pin, or SCR software reset, are communicated back to the main core domain through the following bits of the PMS SCR_STAT register:

- .SCRECC
- .SCRWDT
- .SCRPINRST
- .SCRRST

These events may additionally trigger internal SCR reset if configured in the SCR_SCUSCR_RSTCON register.

The occurrence of SCRECC, SCRWDT, SCRPINRST, and SCRRST events is routed to the TriCore™ domain interrupt controller if enabled through PMS_INT_CON2.SCRECC, .SCRWDT, SCRPINRST, and .SCRRST register bits.

Resets of the TriCore™ domain, namely Application, System, and Power-on Reset, are communicated to the SCR through SCUSCR_MRSTST.APPRST register bit.

SMU issued reset is communicated to the SCR through SCUSCR_MRSTST.SMURST register bit.

An NMI interrupt may be generated in the SCR subsystem when SCUSCR_MRSTST register bits are set.

Interrupt

To exchange data and facilitate status handshake between the SCR and the TriCore™ domain cores a register interface with interrupt support is provided.

The SCR can make a direct interrupt request to any CPUx by writing to the SCR_IRSCR_NMICON.SCRINTTC0 or .SCRINTTC1 register bit. An additional 8-bit information may be written into the SCR_SCUSCR_SCRINTEXCHG register, which is also transferred to the PMS_SCR_STAT.SCRINT register bit-field available to the TriCore™ domain cores. The routing of the interrupt to the TriCore™ domain interrupt request node can be enabled through PMS_INT_CON2.SCRINT0 and .SCRINT1 register bits.

Likewise, any CPUx may also trigger a direct interrupt request to the SCR by writing into the PMS_SCR_CON1.TCINTREQ register bit. An additional 8-bit information maybe written in the PMS_SCR_CON1.TCINT bit-field, which is transferred to the SCR_SCUSCR_TCINTEXCHG.TCINT register bit-field available to the SCR software.

An SCR NMI from the main controller is also triggered on SMURST or application reset.

Wake-up and mode transitions

The SCR may wake-up the TriCore™ domain from STANDBY1 to RUN state if configured in the PMS_WAKEUP_CON1.SCRWKEN register bit.

A wake-up request from SCR is issued by the SCR software through the SCR_SCUSCR_WKPCON.SWWKPREQ bit as documented in the SCR SCUSCR section. After wake-up of the TriCore™ domain, the PMS_WAKEUP_STAT1.SCRWKP event flag is set, which shall be cleared through the PMS_WAKEUP_CLR1.SCRWKPCLR register bit.

PMS wake-up flags can also be cleared from SCR side through SCUSCR_STBY.PMSWKPCLR='1'. Any pending STBY0 entry condition leads the system into a STBY0 mode after all PMS flags are cleared. If there is no pending STBY0 entry condition SCR software can trigger the transition to STBY0 through SCUSCR_STBY.STBY0REQ = 1

SCR has access to the following PMS information in order to execute different software routines if required:

- SCUSCR_STBY.STBYSTAT
  - Information whether the main system is in RUN or STANDBY1 mode
- SCUSCR_STBY.STBY0WKP
  - Information if the main system was previously in STANDBY0 mode
- SCUSCR_STBY.PMSRTCWKP
  - Information if the source of latest system wake-up was PMS RTC. This is useful when calibration of PMS RTC is done through SCR software
- SCUSCR_STBY.PMS0WKP
  - Information if the source of latest system wake-up was any PMS0 source combined as logical OR

Calibration

SCR has access to PMS RTC, DTS, HPOSC and HPBG register in order to provide calibration (trimming) capability during standby mode.

The following information about PMS RTC is made accessible to SCR software:

Table 1747 PMS RTC bit-fields available for SCR software

PMS RTC bit-field name	SCR register name	Description	SCR software access
RTC_TIM1CAP[31:0] & RTC_TIM0[31:0]	SCUSCR_PMSRTC_TIM*	8 registers of 8-bits (TIM63_56, TIM55_48, …, TIM7_0) for a total of 64-bit timer value	Read-only

(table continues...)

Table 1747 (continued) PMS RTC bit-fields available for SCR software

PMS RTC bit-field name	SCR register name	Description	SCR software access
RTC_CON1.OFFSET[15:0]	SCUSCR_PMSRTC_OFFSET*	2 registers of 8-bits (OFFSET15_8 and OFFSET7_0) for a total of 64-bit offset value	Read and write
RTC_CMP1[31:0]	SCUSCR_PMSRTC_CMP1*	4 registers of 8-bits (CMP1_31_24, …, CMP1_7_0) for a total of 32-bits CMP1 value	Read and write
RTC_CON2.MSIZE1[4:0]	SCUSCR_PMSRTC_MSIZE1	5 least-significant bits of MSIZE1 value	Read and write
RTC_CON2.MSTART1[4:0]	SCUSCR_PMSRTC_MSTART1	5 least-significant bits of MSTART1 value	Read and write

For PMS DTS there is following interface with prefix "SCUSCR_PMSDTS_":

Table 1748 PMS DTS bit-fields available for SCR software

PMS DTS bit-field name	SCR register name	Description	SCR software access
DTS_CON.START, DTS_CON.EN	SCUSCR_PMSDTS_CON.EN_START	DTS enable and start control is combined into a single bit	Read and write
DTS_STAT.RESULT[11:0]	SCUSCR_PMSDTS_RES*.VAL	Bit-fields in 2 registers combine for 12-bit result value	Read-only
DTS_STAT.READY	SCUSCR_PMSDTS_RES11_8.RES_READY	1-bit indication that temperature result is ready	Read-only

The PMS HPOSC trimming register is available for SCR software:

Table 1749 PMS HPOSC bit-fields available for SCR software

PMS HPOSC bit-field name	SCR register name	Description	SCR software access
PMS_HPOSCCON1.OSCFTRIM[8:0]	SCUSCR_PMSHPOSC_FTRIM*	2 registers (FTRIM8_8, FTRIM7_0) for a total of 9-bit OSCFTRIM value	Read and write

The PMS HPBG trimming register is available for SCR software:

Table 1750 PMS HPBG bit-fields available for SCR software

PMS HPBG bit-field name	SCR register name	Description	SCR software access
PMS_HPBGCON0.FINETRIM[6:0]	SCUSCR_PMSHPBG_BGTRIM*	7 least-significant bits of FINETRIM value	Read and write

Ports

SCR 8-bit Ports module (SCR_P0-3) shares 32 Pads with the TriCore™ domain 32-bit Port module (P32.2, P32.4-7, P33.0-15, P34.1-5, and P35.0-5).

The SCR ports are supplied by V_{DDPMS1} and $V_{DDEVRSB}$ standby supply. The control to these pins is by default allocated to the SCR through port configuration Pn_PCSRSEL register, if SCR is enabled. Unused wake-up pins may be configured as tri-state in Standby mode through PMS_PAD_CON.SBTRIST.

During a STANDBY1 to RUN mode transition the ownership of shared Pads is not changed. Isolation cells retain the programmed value of Pn_PCSRSELx value after RUN to STANDBY transition. This is to ensure that SCR continues to have control over the respective P32, P33, P34, and P35 port pins during and after exit from STANDBY, although the Port register Pn_PCSRSELx value is reset. Only after a consequent explicit write to the Pn_PCSRSELx value will be taken to switch the P32, P33, P34, and P35 control.

39 Stand-by Controller (SCR)

The wake-up pins *ESR1* and *PINA*(P14.1) are also routed to the SCR subsystem as external inputs to T2CCU0 Timer 2 module.

39 Stand-by Controller (SCR)

Summary

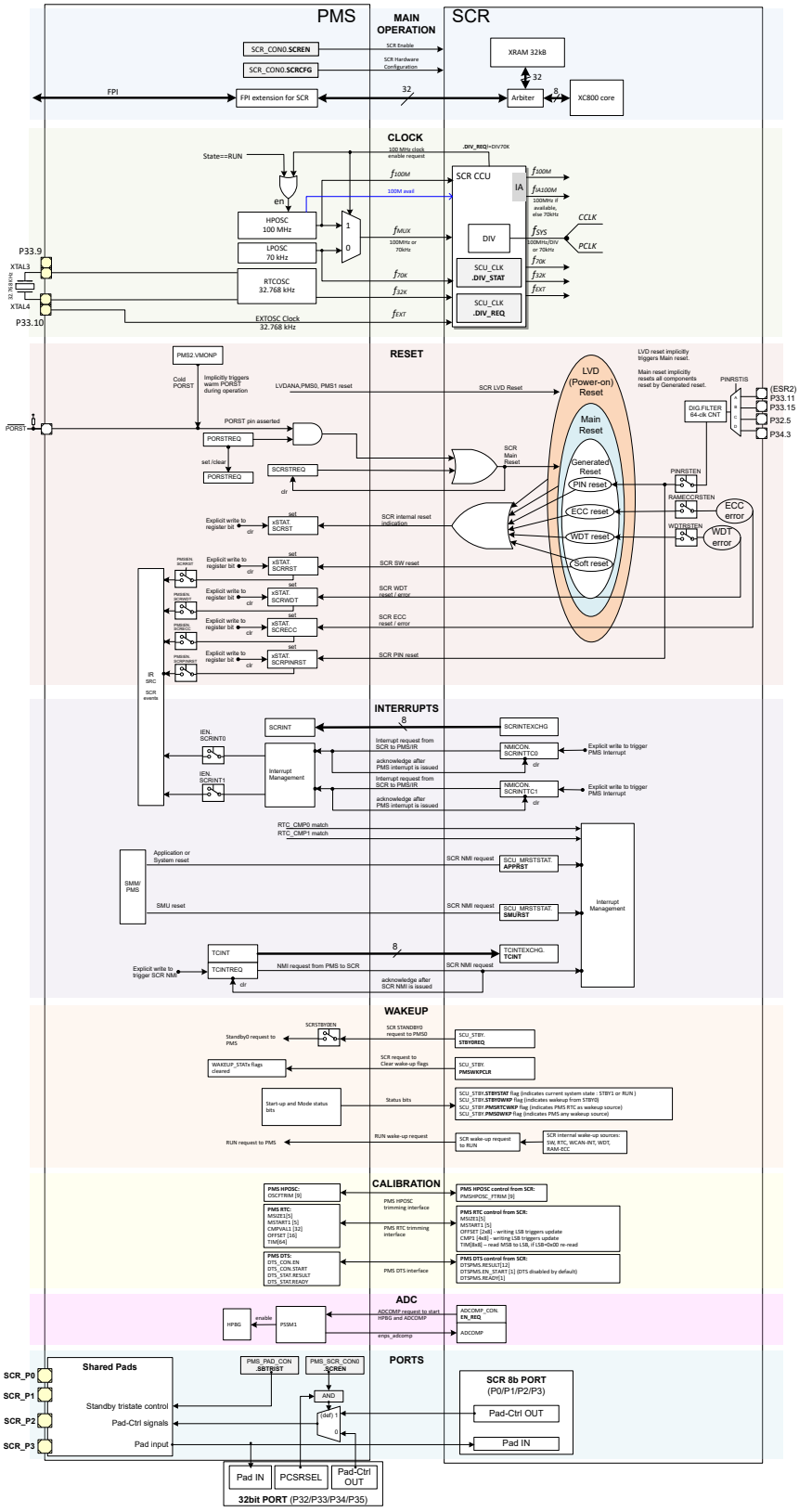


Figure 950 Overview of connections between PMS and SCR

39.4 Registers

39.4.1 Register overview - access mode glossary

Table 1751 Register overview - access mode glossary

Keyword	Description
U	No access restrictions.
nBE	Indicates that no Bus Error is generated when accessing this address range, even though it is either an access to an undefined address or the access does not follow the given rules.

39.4.2 Register overview - SCR domain SCR (ascending offset address)

Table 1752 Register overview - SCR domain SCR (ascending offset address)

Short name	Description	RMAP	PAGE	Offset address	See
T2CCU0_T2_CON	Timer 2 instance 0 control register	0	T2CCU0_PAGE=0	081_H	6699
T2CCU0_CCEN	T2CCU0 capture/compare enable register	0	T2CCU0_PAGE=1	081_H	6705
T2CCU0_COSHDW	T2CCU0 compare shadow register	0	T2CCU0_PAGE=2	081_H	6710
T2CCU0_COCON	T2CCU0 compare control register	0	T2CCU0_PAGE=3	081_H	6713
T2CCU0_T2_MOD	Timer 2 instance 0 mode register	0	T2CCU0_PAGE=0	082_H	6700
T2CCU0_CCTBSEL	T2CCU0 capture/compare time base select register	0	T2CCU0_PAGE=1	082_H	6706
T2CCU0_CCxL	T2CCU0 capture/compare register x low	0	T2CCU0_PAGE=2	082_H+x*2	6712
T2CCU0_CCyL	T2CCU0 capture/compare register y low	0	T2CCU0_PAGE=3	082_H+(y-3)*2	6715
T2CCU0_T2_RC2L	Timer 2 instance 0 reload/capture register, low byte	0	T2CCU0_PAGE=0	083_H	6702
T2CCU0_CCTRELL	T2CCU0 capture/compare timer reload register low	0	T2CCU0_PAGE=1	083_H	6706
T2CCU0_CCxH	T2CCU0 capture/compare register x high	0	T2CCU0_PAGE=2	083_H+x*2	6712
T2CCU0_CCyH	T2CCU0 capture/compare register y high	0	T2CCU0_PAGE=3	083_H+(y-3)*2	6715
T2CCU0_CCTDTCL	T2CCU0 capture/compare timer dead-time control register low	0	T2CCU0_PAGE=4	083_H	6716
T2CCU0_T2_RC2H	Timer 2 instance 0 reload/capture register, high byte	0	T2CCU0_PAGE=0	084_H	6702
T2CCU0_CCTRELH	T2CCU0 capture/compare timer reload register high	0	T2CCU0_PAGE=1	084_H	6707

(table continues...)

Table 1752 (continued) Register overview - SCR domain SCR (ascending offset address)

Short name	Description	RMAP	PAGE	Offset address	See
T2CCU0_CCTDTCH	T2CCU0 capture/compare timer dead-time control register high	0	T2CCU0_PAGE =4	084_H	6717
T2CCU0_T2_T2L	Timer 2 instance 0, low byte	0	T2CCU0_PAGE =0	085_H	6703
T2CCU0_CCTL	T2CCU0 capture/compare timer register low	0	T2CCU0_PAGE =1	085_H	6707
T2CCU0_T2_T2H	Timer 2 instance 0, high byte	0	T2CCU0_PAGE =0	086_H	6703
T2CCU0_CCTH	T2CCU0 capture/compare timer register high	0	T2CCU0_PAGE =1	086_H	6708
T2CCU0_T2_CON1	Timer 2 instance 0 control register 1	0	T2CCU0_PAGE =0	087_H	6703
T2CCU0_CCTCON	T2CCU0 capture/compare timer control register	0	T2CCU0_PAGE =1	087_H	6709
T2CCU0_PAGE	Page register for T2CCU0	0	X	088_H	6717
SSC_CONPL	Control register low [programming mode]	0	X	089_H	6719
SSC_CONOL	Control register low [operation mode]	0	X	089_H	6720
SSC_CONPH	Control register high [programming mode]	0	X	$08A_H$	6720
SSC_CONOH	Control register high [operation mode]	0	X	$08A_H$	6721
SSC_TBL	Transmitter buffer register	0	X	$08B_H$	6722
SSC_RBL	Receiver buffer register	0	X	$08C_H$	6723
SSC_BRL	Baud rate timer reload register low	0	X	$08D_H$	6723
SSC_BRH	Baud rate timer reload register high	0	X	$08E_H$	6724
IO_PAGE	Page register for port SFRs	0	X	$08F_H$	6724
Pn_OUT	Port Pn output register	0	IO_PAGE=0	090_H+n*8	6726
Pn_IOCRm	Port Pn input/output control register m	0	IO_PAGE=1	090_H+n*8+m	6728
Pn_PDR0	Port Pn pad driver mode 0 register	0	IO_PAGE=2	090_H+n*8	6729
Pn_WKEN	Port Pn wake enable request register	0	IO_PAGE=3	090_H+n*8	6732
Pn_IN	Port Pn input register	0	IO_PAGE=0	091_H+n*8	6726
Pn_PDR2	Port Pn pad driver mode 2 register	0	IO_PAGE=2	091_H+n*8	6730
Pn_WKENSTS	Port Pn wake enable status register	0	IO_PAGE=3	091_H+n*8	6732

(table continues...)

Table 1752 (continued) Register overview - SCR domain SCR (ascending offset address)

Short name	Description	RMAP	PAGE	Offset address	See
Pn_OMSR	Port Pn output modification set register	0	IO_PAGE=0	092_H+n*8	6727
Pn_PDR4	Port Pn pad driver mode 4 register	0	IO_PAGE=2	092_H+n*8	6730
Pn_WKSTS	Wakeup status register Port Pn	0	IO_PAGE=3	092_H+n*8	6733
Pn_OMCR	Port Pn output modification clear register	0	IO_PAGE=0	093_H+n*8	6727
Pn_PDR6	Port Pn pad driver mode 6 register	0	IO_PAGE=2	093_H+n*8	6731
Pn_OMTR	Port Pn output modification toggle register	0	IO_PAGE=0	094_H+n*8	6728
Pn_PDISC	Port Pn pin function decision control register	0	IO_PAGE=2	095_H+n*8	6731
WCAN_CFG	Wake-Up CAN configuration register	0	WCAN_PAGE=0	$0B0_H$	6733
WCAN_INTESTAT0	Wake-Up CAN interrupt and event status register 0	0	WCAN_PAGE=1	$0B0_H$	6740
WCAN_ID0_CTRL	Message identifier control register 0	0	WCAN_PAGE=2	$0B0_H$	6748
WCAN_DATAn_CTRL	DATA control register n	0	WCAN_PAGE=3	$0B0_H$+n	6752
WCAN_INTMRSLT	Wake-Up CAN interrupt mask register	0	WCAN_PAGE=0	$0B1_H$	6735
WCAN_INTESTAT1	Wake-Up CAN interrupt and event status register 1	0	WCAN_PAGE=1	$0B1_H$	6742
WCAN_ID1_CTRL	Message identifier control register 1	0	WCAN_PAGE=2	$0B1_H$	6748
WCAN_CDR_CTRL	CDR control register	0	WCAN_PAGE=0	$0B2_H$	6736
WCAN_FRMERRCNT	Wake-Up CAN error counter register	0	WCAN_PAGE=1	$0B2_H$	6743
WCAN_ID2_CTRL	Message identifier control register 2	0	WCAN_PAGE=2	$0B2_H$	6749
WCAN_CDR_UPPER_CTRL	CDR upper limit control register	0	WCAN_PAGE=0	$0B3_H$	6737
WCAN_INTESCLR0	Wake-Up CAN interrupt and event status clear register 0	0	WCAN_PAGE=1	$0B3_H$	6744
WCAN_ID3_CTRL	Message identifier control register 3	0	WCAN_PAGE=2	$0B3_H$	6749
WCAN_CDR_LOWER_CTRL	CDR lower limit control register	0	WCAN_PAGE=0	$0B4_H$	6738
WCAN_INTESCLR1	Wake-Up CAN interrupt and event status clear register 1	0	WCAN_PAGE=1	$0B4_H$	6745
WCAN_MASK_ID0_CTRL	Message identifier mask register 0	0	WCAN_PAGE=2	$0B4_H$	6750

(table continues...)

Table 1752 (continued) Register overview - SCR domain SCR (ascending offset address)

Short name	Description	RMAP	PAGE	Offset address	See
WCAN_CDR_MEAS_HIGH	CDR measured high register	0	WCAN_PAGE=0	$0B5_H$	6738
WCAN_BTL1_CTRL	Bit timing logic 1 control register	0	WCAN_PAGE=1	$0B5_H$	6746
WCAN_MASK_ID1_CTRL	Message identifier mask register 1	0	WCAN_PAGE=2	$0B5_H$	6750
WCAN_CDR_MEAS_LOW	CDR measured low register	0	WCAN_PAGE=0	$0B6_H$	6739
WCAN_BTL2_CTRL	Bit timing logic 2 control register	0	WCAN_PAGE=1	$0B6_H$	6746
WCAN_MASK_ID2_CTRL	Message identifier mask register 2	0	WCAN_PAGE=2	$0B6_H$	6751
WCAN_FD_CTRL	Wake-Up CAN FD control register	0	WCAN_PAGE=0	$0B7_H$	6739
WCAN_DLC_CTRL	Message data length code control register	0	WCAN_PAGE=1	$0B7_H$	6747
WCAN_MASK_ID3_CTRL	Message identifier mask register 3	0	WCAN_PAGE=2	$0B7_H$	6751
WCAN_PAGE	Page register for WCAN	0	X	$0B8_H$	6753
UART_SCON	Serial channel control register	0	X	$0B9_H$	6754
UART_SBUF	Serial data buffer	0	X	$0BA_H$	6755
LIN_BGL	Baudrate timer/reload register, low byte	0	X	$0BB_H$	6756
LIN_BCON	Baudrate control register	0	X	$0BC_H$	6756
LIN_STAT	LIN status register	0	X	$0BD_H$	6758
LIN_BGH	Baudrate timer/reload register, high byte	0	X	$0BE_H$	6758
T2CCU1_T2_CON	Timer 2 instance 1 control register	0	T2CCU1_PAGE=0	$0BF_H$	6759
T2CCU1_CCEN	T2CCU1 capture/compare enable register	0	T2CCU1_PAGE=1	$0BF_H$	6765
T2CCU1_COSHDW	T2CCU1 compare shadow register	0	T2CCU1_PAGE=2	$0BF_H$	6770
T2CCU1_COCON	T2CCU1 compare control register	0	T2CCU1_PAGE=3	$0BF_H$	6773
T2CCU1_T2_MOD	Timer 2 instance 1 mode register	0	T2CCU1_PAGE=0	$0C0_H$	6760
T2CCU1_CCTBSEL	T2CCU1 capture/compare time base select register	0	T2CCU1_PAGE=1	$0C0_H$	6766
T2CCU1_CCxL	T2CCU1 capture/compare register x low	0	T2CCU1_PAGE=2	$0C0_H$+x*2	6772
T2CCU1_CCxL	T2CCU1 capture/compare register x low	0	T2CCU1_PAGE=3	$0C0_H$+(x-3)*2	6775

(table continues...)

Table 1752 (continued) Register overview - SCR domain SCR (ascending offset address)

Short name	Description	RMAP	PAGE	Offset address	See
T2CCU1_T2_RC2L	Timer 2 instance 1 reload/capture register, low byte	0	T2CCU1_PAGE =0	$0C1_H$	6762
T2CCU1_CCTRELL	T2CCU1 capture/compare timer reload register low	0	T2CCU1_PAGE =1	$0C1_H$	6766
T2CCU1_CCxH	T2CCU1 capture/compare register x high	0	T2CCU1_PAGE =2	$0C1_H$+x*2	6772
T2CCU1_CCxH	T2CCU1 capture/compare register x high	0	T2CCU1_PAGE =3	$0C1_H$+(x-3)*2	6775
T2CCU1_CCTDTCL	T2CCU1 capture/compare timer dead-time control register low	0	T2CCU1_PAGE =4	$0C1_H$	6776
T2CCU1_T2_RC2H	Timer 2 instance 1 reload/capture register, high byte	0	T2CCU1_PAGE =0	$0C2_H$	6762
T2CCU1_CCTRELH	T2CCU1 capture/compare timer reload register high	0	T2CCU1_PAGE =1	$0C2_H$	6767
T2CCU1_CCTDTCH	T2CCU1 capture/compare timer dead-time control register high	0	T2CCU1_PAGE =4	$0C2_H$	6777
T2CCU1_T2_T2L	Timer 2 instance 1, low byte	0	T2CCU1_PAGE =0	$0C3_H$	6763
T2CCU1_CCTL	T2CCU1 capture/compare timer register low	0	T2CCU1_PAGE =1	$0C3_H$	6767
T2CCU1_T2_T2H	Timer 2 instance 1, high byte	0	T2CCU1_PAGE =0	$0C4_H$	6763
T2CCU1_CCTH	T2CCU1 capture/compare timer register high	0	T2CCU1_PAGE =1	$0C4_H$	6768
T2CCU1_T2_CON1	Timer 2 instance 1 control register 1	0	T2CCU1_PAGE =0	$0C5_H$	6763
T2CCU1_CCTCON	T2CCU1 capture/compare timer control register	0	T2CCU1_PAGE =1	$0C5_H$	6769
T2CCU1_PAGE	Page register for T2CCU1	0	X	$0C6_H$	6777
ADCOMP_CON	ADCOMP control register	0	X	$0C7_H$	6779
ADCOMP_RESL	ADCOMP result-low register	0	X	$0C8_H$	6780
ADCOMP_RESHSTAT	ADCOMP result-high and status register	0	X	$0C9_H$	6780
T01_TCON	Timer 0/1 control register	X	X	$0CA_H$	6781
T01_TMOD	Timer mode register	X	X	$0CB_H$	6782
T01_TL0	Timer 0 low byte	X	X	$0CC_H$	6783
T01_TL1	Timer 1 low byte	X	X	$0CD_H$	6784
T01_TH0	Timer 0 high byte	X	X	$0CE_H$	6784
T01_TH1	Timer 1 high byte	X	X	$0CF_H$	6785
PSW	Program status word register	X	X	$0D0_H$	6785

(table continues...)

Table 1752 (continued) Register overview - SCR domain SCR (ascending offset address)

Short name	Description	RMAP	PAGE	Offset address	See
IEN1	Interrupt enable register 1	X	X	$0D1_H$	6786
IPH	Interrupt priority high register	X	X	$0D2_H$	6787
IPH1	Interrupt priority 1 high register	X	X	$0D3_H$	6787
SP	Stack pointer	X	X	$0D4_H$	6788
DPL	Data pointer low byte	X	X	$0D5_H$	6788
DPH	Data pointer high byte	X	X	$0D6_H$	6788
EO	Extended operation register	X	X	$0D7_H$	6789
IEN0	Interrupt enable register 0	X	X	$0D8_H$	6789
PCON	Power control register	X	X	$0D9_H$	6790
B	B register	X	X	$0DA_H$	6791
IP	Interrupt priority register	X	X	$0DB_H$	6791
IP1	Interrupt priority 1 register	X	X	$0DC_H$	6792
PASSWD	Password register	X	X	$0DD_H$	6792
XADDRH	On-chip XRAM address higher order	X	X	$0DE_H$	6793
SYSCON0	System control register 0	X	X	$0DF_H$	6793
ACC	Accumulator	X	X	$0E0_H$	6792
RTCSCR_CON	Real-time clock control register	0	X	$0E1_H$	6794
RTCSCR_CNTn	Count clock register n	0	X	$0E2_H$+n	6796
RTCSCR_CRn	Real-time clock compare/capture register n	0	X	$0E6_H$+n	6797
WDTSCR_CON	Watchdog timer control register	0	X	$0EA_H$	6797
WDTSCR_REL	Watchdog timer reload register	0	X	$0EB_H$	6799
WDTSCR_WINB	Watchdog window-boundary register	0	X	$0EC_H$	6799
WDTSCR_L	Watchdog timer low byte	0	X	$0ED_H$	6800
WDTSCR_H	Watchdog timer high byte	0	X	$0EE_H$	6800
SCUSCR_PAGE	Page register for SCUSCR	0	X	$0EF_H$	6801
SCUSCR_RSTCON	Reset control register	0	SCUSCR_PAGE=0	$0F0_H$	6802
SCUSCR_SCRINTEXCHG	SCR interrupt data exchange register	0	SCUSCR_PAGE=1	$0F0_H$	6810
SCUSCR_MODPISEL0	Peripheral input select register 0	0	SCUSCR_PAGE=2	$0F0_H$	6812
SCUSCR_PMSRTC_OFFSET7_0	SCR access to PMS_RTC_CON1.OFFSET[7:0] bits and update trigger for complete PMS_RTC_CON1.OFFSET	0	SCUSCR_PAGE=3	$0F0_H$	6816

(table continues...)

Table 1752 (continued) Register overview - SCR domain SCR (ascending offset address)

Short name	Description	RMAP	PAGE	Offset address	See
SCUSCR_PMSRTC_CMP1_7_0	SCR access to PMS_RTC_CMP1[7:0] bits and update trigger for complete PMS_RTC_CMP1	0	SCUSCR_PAGE=4	$0F0_H$	6818
SCUSCR_PMSDTS_CON	Control for PMS DTS	0	SCUSCR_PAGE=5	$0F0_H$	6821
SCUSCR_PMS1IVRCON	PMS1 iVR control	0	SCUSCR_PAGE=7	$0F0_H$	6824
SCUSCR_RSTSTAT	SCR reset status register	0	SCUSCR_PAGE=0	$0F1_H$	6803
SCUSCR_TCINTEXCHG	TriCore interrupt data exchange register	0	SCUSCR_PAGE=1	$0F1_H$	6810
SCUSCR_MODPISEL1	Peripheral input select register 1	0	SCUSCR_PAGE=2	$0F1_H$	6813
SCUSCR_PMSRTC_OFFSET15_8	SCR shadow register for PMS_RTC_CON1.OFFSET[15:8] bits	0	SCUSCR_PAGE=3	$0F1_H$	6816
SCUSCR_PMSRTC_CMP1_15_8	SCR shadow register for PMS_RTC_CMP1[15:8] bits	0	SCUSCR_PAGE=4	$0F1_H$	6818
SCUSCR_PMSDTS_RES7_0	Die temperature sensor result register - lower 8-bits	0	SCUSCR_PAGE=5	$0F1_H$	6821
SCUSCR_DBG_MODSUSP	Module suspend control register	0	SCUSCR_PAGE=7	$0F1_H$	6825
SCUSCR_MRSTSTAT	Main reset status register	0	SCUSCR_PAGE=0	$0F2_H$	6804
SCUSCR_PMSRTC_TIM39_32	SCR access to PMS_RTC_TIM[39:32] bits	0	SCUSCR_PAGE=2	$0F2_H$	6814
SCUSCR_PMSRTC_TIM7_0	SCR access to PMS_RTC_TIM[7:0] bits	0	SCUSCR_PAGE=3	$0F2_H$	6817
SCUSCR_PMSRTC_CMP1_23_16	SCR shadow register for PMS_RTC_CMP1[23:16] bits	0	SCUSCR_PAGE=4	$0F2_H$	6819
SCUSCR_PMSDTS_RES11_8	Die temperature sensor result register - upper 4-bits	0	SCUSCR_PAGE=5	$0F2_H$	6822
SCUSCR_STBY	SCUSCR status register	0	SCUSCR_PAGE=0	$0F3_H$	6804
SCUSCR_PMCON0	Peripheral management control register 0	0	SCUSCR_PAGE=1	$0F3_H$	6810
SCUSCR_PMSRTC_TIM47_40	SCR access to PMS_RTC_TIM[47:40] bits	0	SCUSCR_PAGE=2	$0F3_H$	6815
SCUSCR_PMSRTC_TIM15_8	SCR access to PMS_RTC_TIM[15:8] bits	0	SCUSCR_PAGE=3	$0F3_H$	6817
SCUSCR_PMSRTC_CMP1_31_24	SCR shadow register for PMS_RTC_CMP1[31:24] bits	0	SCUSCR_PAGE=4	$0F3_H$	6819

(table continues...)

Table 1752 (continued) Register overview - SCR domain SCR (ascending offset address)

Short name	Description	RMAP	PAGE	Offset address	See
SCUSCR_PMSHPOSC_FTRIM7_0	SCR access to PMS_HPOSCCON1.OSCFTRIM[7:0] bits and update trigger for complete PMS_HPOSCCON1.OSCFTRIM	0	SCUSCR_PAGE=5	$0F3_H$	6823
SCUSCR_CLK	Clock control and status register	0	SCUSCR_PAGE=0	$0F4_H$	6806
SCUSCR_PMCON1	Peripheral management control register 1	0	SCUSCR_PAGE=1	$0F4_H$	6812
SCUSCR_PMSRTC_TIM55_48	SCR access to PMS_RTC_TIM[55:48] bits	0	SCUSCR_PAGE=2	$0F4_H$	6815
SCUSCR_PMSRTC_TIM23_16	SCR access to PMS_RTC_TIM[23:16] bits	0	SCUSCR_PAGE=3	$0F4_H$	6817
SCUSCR_PMSRTC_MSIZE1	SCR access to PMS_RTC_CON2.MSIZE1 bits	0	SCUSCR_PAGE=4	$0F4_H$	6820
SCUSCR_PMSHPOSC_FTRIM8_8	SCR shadow register for PMS_HPOSCCON1.OSCFTRIM[8:8] bits	0	SCUSCR_PAGE=5	$0F4_H$	6823
SCUSCR_WKPCON	Control register for system wake-up via SCR sources	0	SCUSCR_PAGE=0	$0F5_H$	6809
SCUSCR_PMSRTC_TIM63_56	SCR access to PMS_RTC_TIM[63:56] bits	0	SCUSCR_PAGE=2	$0F5_H$	6815
SCUSCR_PMSRTC_TIM31_24	SCR access to PMS_RTC_TIM[31:24] bits	0	SCUSCR_PAGE=3	$0F5_H$	6818
SCUSCR_PMSRTC_MSTART1	SCR access to PMS_RTC_CON2.MSTART1 bits	0	SCUSCR_PAGE=4	$0F5_H$	6820
SCUSCR_PMSHPBG_BGTRIM	Trimming register for PMS High Precision BandGap (HPBG)	0	SCUSCR_PAGE=5	$0F5_H$	6824
IRSCR_PAGE	Page register for SFRs	0	X	$0F6_H$	6826
IRSCR_CON0	Interrupt request register 0	0	IRSCR_PAGE=0	$0F7_H$	6827
IRSCR_NMICON	NMI control register	0	IRSCR_PAGE=1	$0F7_H$	6829
IRSCR_EXICON0	External interrupt control register 0	0	IRSCR_PAGE=2	$0F7_H$	6832
IRSCR_EXICON3	External interrupt control register 3	0	IRSCR_PAGE=3	$0F7_H$	6835
IRSCR_PISEL0	External interrupt input select register 0	0	IRSCR_PAGE=4	$0F7_H$	6836
IRSCR_PISEL3	External interrupt input select register 3	0	IRSCR_PAGE=5	$0F7_H$	6840
IRSCR_CON1	Interrupt request register 1	0	IRSCR_PAGE=0	$0F8_H$	6828
IRSCR_NMISTAT	NMI status register	0	IRSCR_PAGE=1	$0F8_H$	6830

(table continues...)

Table 1752 (continued) Register overview - SCR domain SCR (ascending offset address)

Short name	Description	RMAP	PAGE	Offset address	See
IRSCR_EXICON1	External interrupt control register 1	0	IRSCR_PAGE=2	$0F8_H$	6833
IRSCR_PISEL1	External interrupt input select register 1	0	IRSCR_PAGE=4	$0F8_H$	6837
IRSCR_CON2	Interrupt request register 2	0	IRSCR_PAGE=0	$0F9_H$	6829
IRSCR_MODIEN	Peripheral interrupt enable register	0	IRSCR_PAGE=1	$0F9_H$	6831
IRSCR_EXICON2	External interrupt control register 2	0	IRSCR_PAGE=2	$0F9_H$	6834
IRSCR_PISEL2	External interrupt input select register 2	0	IRSCR_PAGE=4	$0F9_H$	6838
I2CSCR_CNTR	I2C control register	0	X	$0FA_H$	6840
I2CSCR_DATA	Data register	0	X	$0FB_H$	6842
I2CSCR_STAT	Status register	0	X	$0FC_H$	6842
I2CSCR_BRCR	Baud rate control register [write mode]	0	X	$0FC_H$	6843
I2CSCR_SRST	Software reset register	0	X	$0FD_H$	6844
I2CSCR_ADDR	Slave address register	0	X	$0FE_H$	6845
I2CSCR_ADDRX	Extended slave address register	0	X	$0FF_H$	6846

39.4.3 Timer 2 instance 0 control register

T2CCU0_T2_CON Offset address: 081_H

Timer 2 instance 0 control register Reset values see: Table 1753

PAGE: T2CCU0_PAGE=0, RMAP: 0

7	6	5	4	3	2	1	0
TF2	**EXF2**	**0**		**EXEN2**	**TR2**	**C_T2**	**CP_RL2**
rwh	rwh	r		rw	rwh	rw	rw

Field	Bits	Type	Description
CP_RL2	0	rw	**Capture/Reload select** 0_B Reload upon overflow or upon negative/positive transition at pin T2EX (when EXEN2 = 1) 1_B Capture timer 2 data register contents on the negative/positive transition at pin T2EX, provided EXEN2 = 1. The negative or positive transition at pin T2EX is selected by bit EDGESEL in register T2MOD.

(table continues...)

(continued)

Field	Bits	Type	Description
C_T2	1	rw	**Timer or counter select** 0_B Timer function selected 1_B Count upon negative edge at pin T2
TR2	2	rwh	**Timer 2 run control** This bit can be set or cleared by software to start or stop the timer T2. It can also be set by hardware via input T2EX, if bit T2MOD.T2RHEN is set to 1, or via a synchronized start with the CCT timer of the T2CCU0. ***Note**: Writing a one to this bit while the timer is running has no effect.* 0_B Timer T2 is halted 1_B Timer T2 is running
EXEN2	3	rw	**Timer 2 external enable control** 0_B External events are disabled 1_B External events are enabled in capture/reload mode
EXF2	6	rwh	**Timer 2 external flag** In capture/reload mode, this bit is set by hardware when a negative/ positive transition occurs at pin T2EX, if bit EXEN2 = 1. This bit must be cleared by software. ***Note**: When bit DCEN = 1 in auto-reload mode, no interrupt request to the core is generated.*
TF2	7	rwh	**Timer 2 overflow/underflow flag** Set by a timer 2 overflow/underflow. Must be cleared by software.
0	5:4	r	**Reserved** must be written with 0.

Table 1753 Reset values of T2CCU0_T2_CON

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.4 Timer 2 instance 0 mode register

T2CCU0_T2_MOD Offset address: 082_H

Timer 2 instance 0 mode register Reset values see: Table 1754

PAGE: T2CCU0_PAGE=0, RMAP: 0

7	6	5	4	3	2	1	0
T2REGS	T2RHEN	EDGESEL	PREN	T2PRE			DCEN
rw	rw	rw	rw	rw			rw

Field	Bits	Type	Description
DCEN	0	rw	**Up/Down counter enable** 0_B Up/Down counter function is disabled 1_B Up/Down counter function is enabled and controlled by pin T2EX (Up = 1, Down = 0)
T2PRE	3:1	rw	**Timer 2 pre-scaler bit** Selects the input clock for Timer 2 which is derived from the peripheral clock. 000_B **PCLK**: 1 $f_{T2} = f_{PCLK}$ 001_B **PCLK_DIV2**: 2 $f_{T2} = f_{PCLK} / 2$ 010_B **PCLK_DIV4**: 4 $f_{T2} = f_{PCLK} / 4$ 011_B **PCLK_DIV8**: 8 $f_{T2} = f_{PCLK} / 8$ 100_B **PCLK_DIV16**: 16 $f_{T2} = f_{PCLK} / 16$ 101_B **PCLK_DIV32**: 32 $f_{T2} = f_{PCLK} / 32$ 110_B **PCLK_DIV64**: 64 $f_{T2} = f_{PCLK} / 64$ 111_B **PCLK_DIV128**: 128 $f_{T2} = f_{PCLK} / 128$
PREN	4	rw	**pre-scaler enable** 0_B pre-scaler is disabled, and the 2 or 12 divider takes effect 1_B pre-scaler is enabled (see T2PRE bit), and the 2 or 12 divider is bypassed
EDGESEL	5	rw	**Edge select in capture mode/reload mode** 0_B The falling edge at pin T2EX is selected 1_B The rising edge at pin T2EX is selected
T2RHEN	6	rw	**Timer 2 external start enable** 0_B Timer 2 external start is disabled 1_B Timer 2 external start is enabled
T2REGS	7	rw	**Edge select for timer 2 external start** 0_B The falling edge at pin T2EX is selected 1_B The rising edge at pin T2EX is selected

Table 1754 Reset values of T2CCU0_T2_MOD

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.5 Timer 2 instance 0 reload/capture register, low byte

T2CCU0_T2_RC2L Offset address: 083_H

Timer 2 instance 0 reload/capture register, low byte Reset values see: Table 1755

PAGE: T2CCU0_PAGE=0, RMAP: 0

7	6	5	4	3	2	1	0
RC2							
rwh							

Field	Bits	Type	Description
RC2	7:0	rwh	**Reload/Capture value [7:0]** If T2_CON.CP_RL2 = 0, these contents are loaded into the timer register upon an overflow condition. If T2_CON.CP_RL2 = 1, this register is loaded with the current timer count upon a negative/positive transition at pin T2EX when T2_CON.EXEN2 = 1.

Table 1755 Reset values of T2CCU0_T2_RC2L

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.6 Timer 2 instance 0 reload/capture register, high byte

T2CCU0_T2_RC2H Offset address: 084_H

Timer 2 instance 0 reload/capture register, high byte Reset values see: Table 1756

PAGE: T2CCU0_PAGE=0, RMAP: 0

7	6	5	4	3	2	1	0
RC2							
rwh							

Field	Bits	Type	Description
RC2	7:0	rwh	**Reload/Capture value [15:8]** If T2_CON.CP_RL2 = 0, these contents are loaded into the timer register upon an overflow condition. If T2_CON.CP_RL2 = 1, this register is loaded with the current timer count upon a negative/positive transition at pin T2EX when T2_CON.EXEN2 = 1.

Table 1756 Reset values of T2CCU0_T2_RC2H

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.7 Timer 2 instance 0, low byte

T2CCU0_T2_T2L Offset address: 085_H

Timer 2 instance 0, low byte Reset values see: Table 1757

PAGE: T2CCU0_PAGE=0, RMAP: 0

7	6	5	4	3	2	1	0
THL2							
rwh							

Field	Bits	Type	Description
THL2	7:0	rwh	**Timer 2 value [7:0]** These bits indicate the lower 8 bits of the current 16-bit timer value.

Table 1757 Reset values of T2CCU0_T2_T2L

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.8 Timer 2 instance 0, high byte

T2CCU0_T2_T2H Offset address: 086_H

Timer 2 instance 0, high byte Reset values see: Table 1758

PAGE: T2CCU0_PAGE=0, RMAP: 0

7	6	5	4	3	2	1	0
THL2							
rwh							

Field	Bits	Type	Description
THL2	7:0	rwh	**Timer 2 value [15:8]** These bits indicate the upper 8 bits of the current 16-bit timer value.

Table 1758 Reset values of T2CCU0_T2_T2H

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.9 Timer 2 instance 0 control register 1

Register T20_CON1 (T2CCU0_PAGE) is used to enable the external interrupt and the overflow interrupt.

T2CCU0_T2_CON1 Offset address: 087_H

Timer 2 instance 0 control register 1 Reset values see: Table 1759

PAGE: T2CCU0_PAGE=0, RMAP: 0

7	6	5	4	3	2	1	0
EXTINSEL				INSEL		TF2EN	EXF2EN
rw				rw		rw	rw

Field	Bits	Type	Description
EXF2EN	0	rw	**External interrupt enable** 0_B External interrupt is disabled 1_B External interrupt is enabled
TF2EN	1	rw	**Overflow/Underflow interrupt enable** 0_B Overflow/underflow interrupt is disabled 1_B Overflow/underflow interrupt is enabled
INSEL	3:2	rw	**Timer 2 instance 0 input select** 00_B **T20A**: Timer 2 instance 0 input T20A is selected 01_B **T20B**: Timer 2 instance 0 input T20B is selected 10_B **T20C**: Timer 2 instance 0 input T20C is selected 11_B **T20D**: Timer 2 instance 0 input T20D is selected
EXTINSEL	7:4	rw	**Timer 2 instance 0 external input select** 0_H **T20EXA**: Timer 2 instance 0 input T20EXA is selected 1_H **T20EXB**: Timer 2 instance 0 input T20EXB is selected 2_H **T20EXC**: Timer 2 instance 0 input T20EXC is selected 3_H **T20EXD**: Timer 2 instance 0 input T20EXD is selected 4_H **T20EXE**: Timer 2 instance 0 input T20EXE is selected 5_H **T20EXF**: Timer 2 instance 0 input T20EXF is selected 6_H **T20EXG**: Timer 2 instance 0 input T20EXG is selected 7_H **T20EXH**: Timer 2 instance 0 input T20EXH is selected 8_H **T20ESR1**: Timer 2 instance 0 input ESR1 is selected 9_H **T20PINA**: Timer 2 instance 0 input PINA is selected A_H **T20CMP0**: Timer 2 instance 0 input PMS RTC CMP0OUT is selected B_H **T20CMP1**: Timer 2 instance 0 input PMS RTC CMP1OUT is selected C_H **T20EXM**: Timer 2 instance 0 input T20EXM is selected D_H **T20EXN**: Timer 2 instance 0 input T20EXN is selected E_H **T20EXO**: Timer 2 instance 0 input T20EXO is selected F_H **T20EXP**: Timer 2 instance 0 input T20EXP is selected

Table 1759 Reset values of T2CCU0_T2_CON1

Reset type	Reset value	Note
LVD Reset	03_H	
Generated Reset	03_H	

39.4.10 T2CCU0 capture/compare enable register

The register is addressable via T2CCU0_PAGE.

***Note**: The T2CCU0 output is only actively driven on enabling the compare function.*

T2CCU0_CCEN Offset address: 081_H

T2CCU0 capture/compare enable register Reset values see: Table 1760

PAGE: T2CCU0_PAGE=1, RMAP: 0

7	6	5	4	3	2	1	0
CCM3		**CCM2**		**CCM1**		**CCM0**	
rw		rw		rw		rw	

Field	Bits	Type	Description
CCM0	1:0	rw	**Capture/Compare enable for channel 0** 00_B **CC_DIS**: Capture/Compare disabled 01_B **CAPTURE_ACT_EDGE**: Capture on active edge at pin T2CC0 10_B **COMP_EN**: Compare enabled 11_B **CAPTURE_ON_WRITE**: Capture on write operation into register CC0L
CCM1	3:2	rw	**Capture/Compare enable for channel 1** 00_B **CC_DIS**: Capture/Compare disabled 01_B **CAPTURE_ACT_EDGE**: Capture on active edge at pin T2CC1 10_B **COMP_EN**: Compare enabled 11_B **CAPTURE_ON_WRITE**: Capture on write operation into register CC1L
CCM2	5:4	rw	**Capture/Compare enable for channel 2** 00_B **CC_DIS**: Capture/Compare disabled 01_B **CAPTURE_ACT_EDGE**: Capture on active edge at pin T2CC2 10_B **COMP_EN**: Compare enabled 11_B **CAPTURE_ON_WRITE**: Capture on write operation into register CC2L
CCM3	7:6	rw	**Capture/Compare enable for channel 3** 00_B **CC_DIS**: Capture/Compare disabled 01_B **CAPTURE_ACT_EDGE**: Capture on active edge at pin T2CC3 10_B **COMP_EN**: Compare enabled 11_B **CAPTURE_ON_WRITE**: Capture on write operation into register CC3L

Table 1760 Reset values of T2CCU0_CCEN

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.11 T2CCU0 capture/compare time base select register

The register is addressable via T2CCU0_PAGE.

T2CCU0_CCTBSEL Offset address: 082_H

T2CCU0 capture/compare time base select register Reset values see: Table 1761

PAGE: T2CCU0_PAGE=1, RMAP: 0

7	6	5	4	3	2	1	0
CASC	CCTTOV	CCTB5	CCTB4	CCTB3	CCTB2	CCTB1	CCTB0
rw	rwh	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
CCTBx (x=0-5)	x	rw	**Channel x time base select** 0_B Selects timer 2 1_B Selects capture/compare timer (CCT)
CCTTOV	6	rwh	**Trigger CCT timer overflow event** On set, this bit is held for one PCLK, then cleared by hardware. Reading this bit always returns 0. 0_B No action 1_B CCT overflow event is triggered immediately
CASC	7	rw	**Cascade timers** 0_B Timer 2 and CCT timer operate independently 1_B Timer 2 is cascaded with CCT timer

Table 1761 Reset values of T2CCU0_CCTBSEL

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.12 T2CCU0 capture/compare timer reload register low

The register is addressable via T2CCU0_PAGE.

T2CCU0_CCTRELL Offset address: 083_H

T2CCU0 capture/compare timer reload register low Reset values see: Table 1762

PAGE: T2CCU0_PAGE=1, RMAP: 0

7	6	5	4	3	2	1	0
CCTREL							
rw							

Field	Bits	Type	Description
CCTREL	7:0	rw	**Capture/Compare timer low byte reload value [7:0]** The contents of the registers are loaded into the CCT timer low byte register upon an overflow condition.

Table 1762 Reset values of T2CCU0_CCTRELL

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.13 T2CCU0 capture/compare timer reload register high

The register is addressable via T2CCU0_PAGE.

T2CCU0_CCTRELH Offset address: 084_H

T2CCU0 capture/compare timer reload register high Reset values see: Table 1763

PAGE: T2CCU0_PAGE=1, RMAP: 0

7	6	5	4	3	2	1	0
CCTREL							
rw							

Field	Bits	Type	Description
CCTREL	7:0	rw	**Capture/Compare timer high byte reload value [15:8]** The contents of the registers are loaded into the CCT timer high byte register upon an overflow condition.

Table 1763 Reset values of T2CCU0_CCTRELH

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.14 T2CCU0 capture/compare timer register low

The register is addressable via T2CCU0_PAGE.

T2CCU0_CCTL Offset address: 085_H

T2CCU0 capture/compare timer register low Reset values see: Table 1764

PAGE: T2CCU0_PAGE=1, RMAP: 0

7	6	5	4	3	2	1	0
CCT							
rwh							

Field	Bits	Type	Description
CCT	7:0	rwh	**Capture/Compare timer low byte value [7:0]** The register represents the low byte counter value of the capture/compare timer. When CCT is not running, a write to CCTL updates the lower byte of timer. To (re-)initialize CCT when it is running, CCTH has to be written first. On write to CCTL, the values CCTH and CCTL will be updated to the timer, which then counts from this value. ***Note**:*

Table 1764 Reset values of T2CCU0_CCTL

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.15 T2CCU0 capture/compare timer register high

The register is addressable via T2CCU0_PAGE.

T2CCU0_CCTH Offset address: 086_H

T2CCU0 capture/compare timer register high Reset values see: Table 1765

PAGE: T2CCU0_PAGE=1, RMAP: 0

7	6	5	4	3	2	1	0
CCT							
rwh							

Field	Bits	Type	Description
CCT	7:0	rwh	**Capture/Compare timer high byte value [15:8]** The register represents the high byte counter value of the capture/compare timer. When CCT is not running, a write to CCTH updates the upper byte of timer. To (re-)initialize CCT when it is running, CCTH has to be written first. On write to CCTL, the values CCTH and CCTL will be updated to the timer, which then counts from this value. ***Note**:*

Table 1765 Reset values of T2CCU0_CCTH

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.16 T2CCU0 capture/compare timer control register

The register is addressable via T2CCU0_PAGE.

T2CCU0_CCTCON Offset address: 087_H

T2CCU0 capture/compare timer control register Reset values see: Table 1766

PAGE: T2CCU0_PAGE=1, RMAP: 0

7	6	5	4	3	2	1	0
CCTPRE				CCTOVF	CCTOVEN	TIMSYN	CCTST
rw				rwh	rw	rw	rw

Field	Bits	Type	Description
CCTST	0	rw	**Capture/Compare timer start/stop control** ***Note:*** *Setting CCTST while the bit was already set will not disturb the running CCT timer operation.* 0_B Stop capture/compare timer 1_B Start capture/compare timer. If bit TIMSYN = 1 and CCTST was 0, start also timer 2 (if it was not already running).
TIMSYN	1	rw	**Enable synchronized timer starts** 0_B Disable synchronized timer starts 1_B Enable synchronized timer starts
CCTOVEN	2	rw	**Capture/Compare timer overflow interrupt enable** Set to enable CCT overflow interrupt.
CCTOVF	3	rwh	**Capture/Compare timer overflow flag** Set by hardware on CCT overflow. This bit has to be cleared by software. ***Note:*** *In case of software and hardware access to the flag at the same time, hardware will have higher priority.*

(table continues...)

(continued)

Field	Bits	Type	Description
CCTPRE	7:4	rw	**Compare/Compare timer pre-scaler bitfield** Selects the input clock for CCT, which is derived from the peripheral clock. ***Note:*** *It is not recommended to change CCTPRE when CCTST = 1.* 0_H **DIV**: 1 $f_{CCT} = f_{PCLK}$ 1_H **DIV2**: 2 $f_{CCT} = {}^{fPCLK}/_{2}$ 2_H **DIV4**: 4 $f_{CCT} = {}^{fPCLK}/_{4}$ 3_H **DIV8**: 8 $f_{CCT} = {}^{fPCLK}/_{8}$ 4_H **DIV16**: 16 $f_{CCT} = {}^{fPCLK}/_{16}$ 5_H **DIV32**: 32 $f_{CCT} = {}^{fPCLK}/_{32}$ 6_H **DIV64**: 64 $f_{CCT} = {}^{fPCLK}/_{64}$ 7_H **DIV128**: 128 $f_{CCT} = {}^{fPCLK}/_{128}$ 8_H **DIV256**: 256 $f_{CCT} = {}^{fPCLK}/_{256}$ 9_H **DIV512**: 512 $f_{CCT} = {}^{fPCLK}/_{512}$ A_H **DIV1024**: 1024 $f_{CCT} = {}^{fPCLK}/_{1024}$ B_H **DIV2048**: 2048 $f_{CCT} = {}^{fPCLK}/_{2048}$

Table 1766 Reset values of T2CCU0_CCTCON

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.17 T2CCU0 compare shadow register

The register is addressable via T2CCU0_PAGE.

T2CCU0_COSHDW Offset address: 081_H

T2CCU0 compare shadow register Reset values see: Table 1767

PAGE: T2CCU0_PAGE=2, RMAP: 0

7	6	5	4	3	2	1	0
ENSHDW	TXOV	COOUT5	COOUT4	COOUT3	COOUT2	COOUT1	COOUT0
rwh	rw	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
COOUTx (x=0-5)	x	rwh	**Compare channel x output** **ENSHDW = 0:** Any access to bit COOUTx accesses the corresponding bit of the internal 6-bit compare output register. Writing to bit COOUTx will change the corresponding bit of the internal compare output register. Reading from bit COOUTx returns the value of the bit in the internal compare output register. The content of the internal compare output register is output on the corresponding T2CCx pins when a valid compare match occurs for a channel in compare mode 1 or concurrent compare mode. ***Note**: A concurrent write operation to clear bit ENSHDW and set/ clear bit(s) COOUTx is effective for initializing the internal compare output register.* **ENSHDW = 1:** In this case, accessing bit COOUTx accesses the actual SFR bits. If COOUTx is set, the contents of register CCx will be transferred to corresponding channel x internal compare register. On set, COOUTx is held for one PCLK, then cleared automatically by hardware. ***Note**: A concurrent write operation to set bit ENSHDW and bit(s) COOUTx (and clear bit TXOV) is effective for starting shadow transfer(s).*
TXOV	6	rw	**Enable shadow transfer on timer overflow** Set to enable for all compare channels, transfer of shadow value in T2CCU0_CCx to the corresponding internal compare registers on overflow of their respective timer base.
ENSHDW	7	rwh	**Enable shadow transfer with COOUTx** Clear to enable access of the internal compare output register via the COOUTx bits. Set to enable use of COOUTx bits for transfer of shadow value in T2CCU0_CCx to the corresponding channel x internal compare register. ***Note**: When setting bit TXOV, bit ENSHDW will be cleared by internal hardware. To ensure bit ENSHDW is set, user must ensure bit TXOV is cleared.*

Table 1767 Reset values of T2CCU0_COSHDW

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.18 T2CCU0 capture/compare register x low

The register is addressable via T2CCU0_PAGE.

T2CCU0_CCxL (x=0-2) Offset address: 082_H+x*2

T2CCU0 capture/compare register x low Reset values see: Table 1768

PAGE: T2CCU0_PAGE=2, RMAP: 0

7	6	5	4	3	2	1	0
CCVALL							
rwh							

Field	Bits	Type	Description
CCVALL	7:0	rwh	**Capture/Compare low byte value for channel x** **Capture:** On read, CCVALL returns the low byte of the last captured value. **Compare:** CCVALL holds the low byte written. CCx acts as a shadow register to the channel x internal compare register. The value written is only transferred to the internal compare register when the corresponding bit T2CCU0_COSHDW.COOUTx is being set (with ENSHDW = 1), or on overflow of the channel timer base. The read value always return the last written value.

Table 1768 Reset values of T2CCU0_CCxL (x=0-2)

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.19 T2CCU0 capture/compare register x high

The register is addressable via T2CCU0_PAGE.

T2CCU0_CCxH (x=0-2) Offset address: 083_H+x*2

T2CCU0 capture/compare register x high Reset values see: Table 1769

PAGE: T2CCU0_PAGE=2, RMAP: 0

7	6	5	4	3	2	1	0
CCVALH							
rwh							

Field	Bits	Type	Description
CCVALH	7:0	rwh	**Capture/Compare high byte value for channel x** **Capture:** On read, CCVALH returns the high byte of the last captured value. **Compare:** CCVALH holds the high byte written. CCx acts as a shadow register to the channel x internal compare register. The value written is only transferred to the internal compare register when the corresponding bit T2CCU0_COSHDW.COOUTx is being set (with ENSHDW = 1), or on overflow of the channel timer base. The read value always return the last written value.

Table 1769 Reset values of T2CCU0_CCxH (x=0-2)

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.20 T2CCU0 compare control register

The register is addressable via T2CCU0_PAGE.

T2CCU0_COCON Offset address: 081_H

T2CCU0 compare control register Reset values see: Table 1770

PAGE: T2CCU0_PAGE=3, RMAP: 0

7	6	5	4	3	2	1	0
CCM5	CCM4	CM5F	CM4F	POLB	POLA	COMOD	
rw	rw	rwh	rwh	rw	rw	rw	

Field	Bits	Type	Description
COMOD	1:0	rw	**Compare mode control (for compare channels)** 00_B Compare mode 0 selected 01_B Compare mode 1 selected 10_B Concurrent compare mode selected 11_B Reserved
POLA	2	rw	**Compare active level for channel group A** In compare mode 0, the bit defines the active level of the outputs of the compare channels belonging to the group. 0_B High on compare match 1_B Low on compare match
POLB	3	rw	**Compare active level for channel group B** In compare mode 0, the bit defines the active level of the outputs of the compare channels belonging to the group. 0_B High on compare match 1_B Low on compare match

(table continues...)

(continued)

Field	Bits	Type	Description
CM4F	4	rwh	**Compare channel 4 interrupt flag** Set on a compare match event on compare channel 4. This bit has to be cleared by software. ***Note:*** *In case of software and hardware access to a flag bit at the same time, hardware will have higher priority.*
CM5F	5	rwh	**Compare channel 5 interrupt flag** Set on a compare match event on compare channel 5. This bit has to be cleared by software. ***Note:*** *In case of software and hardware access to a flag bit at the same time, hardware will have higher priority.*
CCM4	6	rw	**Compare enable for channel 4** ***Note:*** *Compare channels 4 and 5 should be enabled in a separate step after setting the compare mode via bits COMOD (and active level POLA/B if compare mode 0). This is to ensure defined compare output according to the selected mode on starting the respective compare channel.* ***Note:*** *The interrupt enable for compare channel 4 is controlled by the respective bit of SCU SFR MODIEN.CM4EN.* 0_B Compare disabled 1_B Compare enabled
CCM5	7	rw	**Compare enable for channel 5** ***Note:*** *Compare channels 4 and 5 should be enabled in a separate step after setting the compare mode via bits COMOD (and active level POLA/B if compare mode 0). This is to ensure defined compare output according to the selected mode on starting the respective compare channel.* ***Note:*** *The interrupt enable for compare channel 5 is controlled by the respective bit of SCU SFR MODIEN.CM5EN.* 0_B Compare disabled 1_B Compare enabled

Table 1770 Reset values of T2CCU0_COCON

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.21 T2CCU0 capture/compare register y low

The register is addressable via T2CCU0_PAGE.

T2CCU0_CCyL (y=3-5) Offset address: 082_H+(y-3)*2

T2CCU0 capture/compare register y low Reset values see: Table 1771

PAGE: T2CCU0_PAGE=3, RMAP: 0

rwh

Field	Bits	Type	Description
CCVALL	7:0	rwh	**Capture/Compare low byte value for channel y** **Capture:** On read, CCVALL returns the low byte of the last captured value. **Compare:** CCVALL holds the low byte written. CCy acts as a shadow register to the channel x internal compare register. The value written is only transferred to the internal compare register when the corresponding bit T2CCU0_COSHDW.COOUTx is being set (with ENSHDW = 1), or on overflow of the channel timer base. The read value always return the last written value. ***Note**: For T2CCU0_CC4L/H and T2CCU0_CC5L/H, only the compare function is available.*

Table 1771 Reset values of T2CCU0_CCyL (y=3-5)

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.22 T2CCU0 capture/compare register y high

The register is addressable via T2CCU0_PAGE.

T2CCU0_CCyH (y=3-5) Offset address: 083_H+(y-3)*2

T2CCU0 capture/compare register y high Reset values see: Table 1772

PAGE: T2CCU0_PAGE=3, RMAP: 0

rwh

Field	Bits	Type	Description
CCVALH	7:0	rwh	**Capture/Compare high byte value for channel y** **Capture:** On read, CCVALH returns the high byte of the last captured value. **Compare:** CCVALH holds the high byte written. CCx acts as a shadow register to the channel x internal compare register. The value written is only transferred to the internal compare register when the corresponding bit T2CCU0_COSHDW.COOUTx is being set (with ENSHDW = 1), or on overflow of the channel timer base. The read value always return the last written value. ***Note**: For T2CCU0_CC4L/H and T2CCU0_CC5L/H, only the compare function is available.*

Table 1772 Reset values of T2CCU0_CCyH (y=3-5)

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.23 T2CCU0 capture/compare timer dead-time control register low

The register is addressable via T2CCU0_PAGE.

T2CCU0_CCTDTCL Offset address: 083_H

T2CCU0 capture/compare timer dead-time control register low

Reset values see: Table 1773

PAGE: T2CCU0_PAGE=4, RMAP: 0

7	6	5	4	3	2	1	0
DTM							
rw							

Field	Bits	Type	Description
DTM	7:0	rw	**Dead-time value** This 8-bit value defines the programmable delay between switching from the passive state to the active state of the selected outputs. The switching from the active state to the passive state is not delayed.

Table 1773 Reset values of T2CCU0_CCTDTCL

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.24 T2CCU0 capture/compare timer dead-time control register high

The register is addressable via T2CCU0_PAGE.

T2CCU0_CCTDTCH Offset address: 084_H

T2CCU0 capture/compare timer dead-time control register high Reset values see: Table 1774

PAGE: T2CCU0_PAGE=4, RMAP: 0

7	6	5	4	3	2	1	0
DTRES	**DTR2**	**DTR1**	**DTR0**	**DTLEV**	**DTE2**	**DTE1**	**DTE0**
rwh	rh	rh	rh	rw	rw	rw	rw

Field	Bits	Type	Description
DTEx (x=0-2)	x	rw	**Dead-time enable for channel pair x** ***Note**: This bit shall only be set after enabling the corresponding channels in compare mode 0 with CCT timer as the time base.* 0_B Dead-time generation is disabled 1_B Dead-time generation is enabled for channel pair x. The corresponding outputs switch from the passive state to the active state (according to compare status) with the delay programmed in DTM.
DTLEV	3	rw	**Dead-time level** This bit defines the active level of all DTCx during dead-time counting. 0_B Low (switches are active high) 1_B High (switches are active low)
DTRx (x=0-2)	x+4	rh	**Dead-time run indication bits** 0_B The value of the corresponding dead-time counter is 0 1_B The value of the corresponding dead-time counter is not 0
DTRES	7	rwh	**Reset dead-time counters** On set, this bit is held for one PCLK, then cleared by hardware. Reading this bit always returns 0. 0_B No action 1_B The three dead-time counters are reset to 0

Table 1774 Reset values of T2CCU0_CCTDTCH

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.25 Page register for T2CCU0

T2CCU0_PAGE Offset address: 088_H

Page register for T2CCU0 Reset values see: Table 1775

PAGE: X, RMAP: 0

7	6	5	4	3	2	1	0
OP		STNR		0	PAGE		
w		w		r	rwh		

Field	Bits	Type	Description
PAGE	2:0	rwh	**Page bits** When written, the value indicates the new page address. When read, the value indicates the currently active page = addr[y:x+1]. 000_B **PAGE0**: Selection is PAGE0 001_B **PAGE1**: Selection is PAGE1 010_B **PAGE2**: Selection is PAGE2 011_B **PAGE3**: Selection is PAGE3 100_B **PAGE4**: Selection is PAGE4 101_B **PAGE5**: Selection is PAGE5 110_B **PAGE6**: Selection is PAGE6 111_B **PAGE7**: Selection is PAGE7
STNR	5:4	w	**Storage number** This number indicates which storage bitfield is the target of the operation defined by bit OP. If OP = 10_B, the contents of PAGE are saved in STx before being overwritten with the new value. If OP = 11_B, the contents of PAGE are overwritten by the contents of STx. The value written to bitfield PAGE is ignored. 00_B **MOD_ST0**: ST0 is selected 01_B **MOD_ST1**: ST1 is selected 10_B **MOD_ST2**: ST2 is selected 11_B **MOD_ST3**: ST3 is selected
OP	7:6	w	**Operation** 00_B **PAGE_MANUAL0**: Manual page mode. The value of STNR is ignored and PAGE is directly written. 01_B **PAGE_MANUAL1**: Manual page mode. The value of STNR is ignored and PAGE is directly written 10_B **PAGE_SAVE**: New page programming with automatic page saving. The value written to the bit positions of PAGE is stored. In parallel, the former contents of PAGE are saved in the storage bit field STx indicated by STNR. 11_B **PAGE_RESTORE**: Automatic restore page action. The value written to the bit positions PAGE is ignored and instead, PAGE is overwritten by the contents of the storage bit field STx indicated by STNR.
0	3	r	**Reserved** Read as 0; must be written with 0.

Table 1775 Reset values of T2CCU0_PAGE

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.26 Control register low [programming mode]

***Note**: When initializing the SSC, register SSC_CONPL must be written first, before writing to register SSC_CONPH and setting bit SSC_CONPH.EN to 1.*

SSC_CONPL Offset address: 089_H

Control register low [programming mode] Reset values see: Table 1776

PAGE: X, RMAP: 0

7	6	5	4	3	2	1	0
LB	**PO**	**PH**	**HB**	**0**	**BM**		
rw	rw	rw	rw	r	rw		

Field	Bits	Type	Description
BM	2:0	rw	**Data width selection** ***Note**: The transfer and receive data width is maximum 8 bits. Transfer data width is 4...8 bits (<BM>+1).* 000_B Reserved. Do not use this combination. ... 010_B Reserved. Do not use this combination. 011_B Transfer data width is 4 bits 100_B Transfer data width is 5 bits 101_B Transfer data width is 6 bits 110_B Transfer data width is 7 bits 111_B Transfer data width is 8 bits
HB	4	rw	**Heading control** 0_B **LSB_FIRST**: Transmit/Receive LSB first 1_B **MSB_FIRST**: Transmit/Receive MSB first
PH	5	rw	**Clock phase control** 0_B **LATCH_ON_TRAILING_EDGE**: Shift transmit data on the leading clock edge, latch on trailing edge. 1_B **LATCH_ON_LEADING_EDGE**: Latch receive data on leading clock edge, shift on trailing edge.
PO	6	rw	**Clock polarity control** 0_B **IDLE_LOW**: Idle clock line is low, leading clock edge is low-to-high transition. 1_B **IDLE_HIGH**: Idle clock line is high, leading clock edge is high-to-low transition.
LB	7	rw	**Loop back control** This bit shall be set by software only in master mode. 0_B Normal output 1_B Receive input is connected with transmit output (half-duplex mode)
0	3	r	**Reserved** Read as 0; must be written with 0.

Table 1776 Reset values of SSC_CONPL

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.27 Control register low [operation mode]

SSC_CONOL Offset address: 089_H

Control register low [operation mode] Reset values see: Table 1777

PAGE: X, RMAP: 0

7	6	5	4	3	2	1	0
0				BC			
r				rh			

Field	Bits	Type	Description
BC	3:0	rh	**Bit count field** This field is loaded with the programmed data width value (BM) when the SSC is enabled. It then counts down with every shift bit. ***Note:*** *This bit field is not to be written to.*
0	7:4	r	**Reserved**

Table 1777 Reset values of SSC_CONOL

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.28 Control register high [programming mode]

SSC_CONPH Offset address: $08A_H$

Control register high [programming mode] Reset values see: Table 1778

PAGE: X, RMAP: 0

7	6	5	4	3	2	1	0
EN	MS	0	0	BEN	0	REN	TEN
rw	rw	r	r	rw	r	rw	rw

Field	Bits	Type	Description
TEN	0	rw	**Transmit error enable** 0_B Ignore transmit errors 1_B Check transmit errors

(table continues...)

(continued)

Field	Bits	Type	Description
REN	1	rw	**Receive error enable** 0_B Ignore receive errors 1_B Check receive errors
BEN	3	rw	**Baudrate error enable** 0_B Ignore baudrate errors 1_B Check baudrate errors
MS	6	rw	**Master select** This bit shall not be changed during operation mode. 0_B **SLAVE**: Slave mode. Operate on shift clock received via SCLK. 1_B **MASTER**: Master mode. Generate shift clock and output it via SCLK.
EN	7	rw	**Enable bit = 0 (needed for programming mode)** Transmission and reception disabled. Access to control bits.
0	2, 4, 5	r	**Reserved** Read as 0; must be written with 0.

Table 1778 Reset values of SSC_CONPH

Reset type	Reset value	Note
LVD Reset	40_H	
Generated Reset	40_H	

39.4.29 Control register high [operation mode]

***Note**: Before disabling the SSC by clearing bit SSC_CONOH.EN during operation, received data has to be read out of the FIFO, as otherwise the software can read corrupted data.*

SSC_CONOH Offset address: $08A_H$

Control register high [operation mode] Reset values see: Table 1779

PAGE: X, RMAP: 0

7	6	5	4	3	2	1	0
EN	**MS**	**RFF**	**BSY**	**BE**	**RFE**	**RE**	**TE**
rw	rw	rh	rh	rwh	rh	rwh	rwh

Field	Bits	Type	Description
TE	0	rwh	**Transmit error flag** 0_B No error 1_B Transfer starts with the slave's transmit buffer not being updated

(table continues...)

(continued)

Field	Bits	Type	Description
RE	1	rwh	**Receive error flag** 0_B No error 1_B Reception in FIFO is blocked
RFE	2	rh	**Receive FIFO buffer empty flag** 0_B Receive FIFO contains data 1_B Receive FIFO is empty
BE	3	rwh	**Baudrate error flag** 0_B No error 1_B More than factor 2 or 0.5 between slave's actual and expected baudrate
BSY	4	rh	**Busy flag** Set while a transfer is in progress. Note: This bit is not to be written to. 0_B No transfer in progress 1_B Transfer in progress
RFF	5	rh	**Receive FIFO buffer full flag** ***Note:*** *FIFO full will be deleted as soon as the receive buffer is read by software.* 0_B Receive FIFO is not full 1_B Receive FIFO is full
MS	6	rw	**Master select** This bit shall not be modified during operation mode. 0_B Slave mode. Operate on shift clock received via SCLK. 1_B Master mode. Generate shift clock and output it via SCLK.
EN	7	rw	**Enable bit = 1 (needed for operating mode)** Transmission and reception enabled. Access to status flags and M/S control.

Table 1779 Reset values of SSC_CONOH

Reset type	Reset value	Note
LVD Reset	40_H	
Generated Reset	40_H	

39.4.30 Transmitter buffer register

SSC_TBL Offset address: $08B_H$

Transmitter buffer register Reset values see: Table 1780

PAGE: X, RMAP: 0

7	6	5	4	3	2	1	0
TB_VALUE							
rw							

Field	Bits	Type	Description
TB_VALUE	7:0	rw	**Transmit data register value** TB_VALUE is the data value to be transmitted. Unselected bits of TB_VALUE are ignored during transmission.

Table 1780 Reset values of SSC_TBL

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.31 Receiver buffer register

***Note**: Before disabling the SSC by clearing bit SSC_CONOH.EN during operation, received data has to be read out of the FIFO, as otherwise the software can read corrupted data.*

SSC_RBL Offset address: $08C_H$

Receiver buffer register Reset values see: Table 1781

PAGE: X, RMAP: 0

7	6	5	4	3	2	1	0
RB_VALUE							
rh							

Field	Bits	Type	Description
RB_VALUE	7:0	rh	**Receive data register value** SSC_RBL contains the received data value RB_VALUE. Unselected bits of RB_VALUE will be not valid and should be ignored. Access point to FIFO. In case of FIFO overflow, the receive FIFO is blocked.

Table 1781 Reset values of SSC_RBL

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.32 Baud rate timer reload register low

SSC_BRL Offset address: $08D_H$

Baud rate timer reload register low Reset values see: Table 1782

PAGE: X, RMAP: 0

7	6	5	4	3	2	1	0
BR_VALUE							
rw							

Field	Bits	Type	Description
BR_VALUE	7:0	rw	**Baud rate timer reload register value, low byte** Represents the lower 8 bits of the 16-bit baudrate reload value.

Table 1782 Reset values of SSC_BRL

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.33 Baud rate timer reload register high

SSC_BRH Offset address: $08E_H$

Baud rate timer reload register high Reset values see: Table 1783

PAGE: X, RMAP: 0

7	6	5	4	3	2	1	0
BR_VALUE							
rw							

Field	Bits	Type	Description
BR_VALUE	7:0	rw	**Baud rate timer reload register value, high byte** Represents the upper 8 bits of the 16-bit baudrate reload value.

Table 1783 Reset values of SSC_BRH

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.34 Page register for port SFRs

Page register for Port registers of SCU.

IO_PAGE Offset address: $08F_H$

Page register for port SFRs Reset values see: Table 1784

PAGE: X, RMAP: 0

7	6	5	4	3	2	1	0
OP		STNR		0	PAGE		
w		w		r	rwh		

Field	Bits	Type	Description
PAGE	2:0	rwh	**Page bits** When written, the value indicates the new page address. When read, the value indicates the currently active page = addr[y:x+1]. 000_B **PAGE0**: Selection is PAGE0 001_B **PAGE1**: Selection is PAGE1 010_B **PAGE2**: Selection is PAGE2 011_B **PAGE3**: Selection is PAGE3 100_B **PAGE4**: Selection is PAGE4 101_B **PAGE5**: Selection is PAGE5 110_B **PAGE6**: Selection is PAGE6 111_B **PAGE7**: Selection is PAGE7
STNR	5:4	w	**Storage number** This number indicates which storage bitfield is the target of the operation defined by bit OP. If OP = 10_B, the contents of PAGE are saved in STx before being overwritten with the new value. If OP = 11_B, the contents of PAGE are overwritten by the contents of STx. The value written to bitfield PAGE is ignored. 00_B **MOD_ST0**: ST0 is selected 01_B **MOD_ST1**: ST1 is selected 10_B **MOD_ST2**: ST2 is selected 11_B **MOD_ST3**: ST3 is selected
OP	7:6	w	**Operation** 00_B **PAGE_MANUAL0**: Manual page mode. The value of STNR is ignored and PAGE is directly written. 01_B **PAGE_MANUAL1**: Manual page mode. The value of STNR is ignored and PAGE is directly written 10_B **PAGE_SAVE**: New page programming with automatic page saving. The value written to the bit positions of PAGE is stored. In parallel, the former contents of PAGE are saved in the storage bit field STx indicated by STNR. 11_B **PAGE_RESTORE**: Automatic restore page action. The value written to the bit positions PAGE is ignored and instead, PAGE is overwritten by the contents of the storage bit field STx indicated by STNR.
0	3	r	**Reserved** Read as 0; must be written with 0.

Table 1784 Reset values of IO_PAGE

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.35 Port Pn output register

Pn_OUT (n=0-3) Offset address: 090_H+n*8

Port Pn output register Reset values see: Table 1785

PAGE: IO_PAGE=0, RMAP: 0

7	6	5	4	3	2	1	0
P7	**P6**	**P5**	**P4**	**P3**	**P2**	**P1**	**P0**
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
Pm (m=0-7)	m	rwh	**Port Pn output bit m** This bit determines the level at the output pin Pn.m if the output is selected as GPIO output. Pn.m can also be set or cleared by control bits of the SCR_Pn_OMSR, SCR_Pn_OMCR or SCR_Pn_OMTR registers. 0_B The output level of Pn.m is 0 1_B The output level of Pn.m is 1

Table 1785 Reset values of Pn_OUT (n=0-3)

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.36 Port Pn input register

Pn_IN (n=0-3) Offset address: 091_H+n*8

Port Pn input register Reset values see: Table 1786

PAGE: IO_PAGE=0, RMAP: 0

7	6	5	4	3	2	1	0
P7	**P6**	**P5**	**P4**	**P3**	**P2**	**P1**	**P0**
rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
Pm (m=0-7)	m	rh	**Port Pn input bit m** This bit reflects the level at the input pin Pn.m. 0_B The input level of Pn.m is 0 1_B The input level of Pn.m is 1

Table 1786 Reset values of Pn_IN (n=0-3)

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.37 Port Pn output modification set register

Pn_OMSR (n=0-3) Offset address: 092_H+n*8

Port Pn output modification set register Reset values see: Table 1787

PAGE: IO_PAGE=0, RMAP: 0

7	6	5	4	3	2	1	0
PS7	PS6	PS5	PS4	PS3	PS2	PS1	PS0
w	w	w	w	w	w	w	w

Field	Bits	Type	Description
PSm (m=0-7)	m	w	**Port Pn set bit m** Setting this bit will set the corresponding bit in the port output register SCR_Pn_OUT. Read as 0. 0_B No operation 1_B Sets SCR_Pn_OUT.Pm

Table 1787 Reset values of Pn_OMSR (n=0-3)

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.38 Port Pn output modification clear register

Pn_OMCR (n=0-3) Offset address: 093_H+n*8

Port Pn output modification clear register Reset values see: Table 1788

PAGE: IO_PAGE=0, RMAP: 0

7	6	5	4	3	2	1	0
PCL7	PCL6	PCL5	PCL4	PCL3	PCL2	PCL1	PCL0
w	w	w	w	w	w	w	w

Field	Bits	Type	Description
PCLm (m=0-7)	m	w	**Port Pn clear bit m** Setting this bit will clear the corresponding bit in the port output register SCR_Pn_OUT. Read as 0. 0_B No operation 1_B Clears SCR_Pn_OUT.Pm

Table 1788 Reset values of Pn_OMCR (n=0-3)

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.39 Port Pn output modification toggle register

Pn_OMTR (n=0-3) Offset address: 094_H+n*8

Port Pn output modification toggle register Reset values see: Table 1789

PAGE: IO_PAGE=0, RMAP: 0

7	6	5	4	3	2	1	0
PTL7	PTL6	PTL5	PTL4	PTL3	PTL2	PTL1	PTL0
w	w	w	w	w	w	w	w

Field	Bits	Type	Description
PTLm (m=0-7)	m	w	**Port Pn toggle bit m** Setting this bit will toggle the corresponding bit in the port output register SCR_Pn_OUT. Read as 0. 0_B No operation 1_B Toggles SCR_Pn_OUT.Pm

Table 1789 Reset values of Pn_OMTR (n=0-3)

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.40 Port Pn input/output control register m

Pn_IOCRm (m=0-7;n=0-3) Offset address: 090_H+n*8+m

Port Pn input/output control register m Reset values see: Table 1790

PAGE: IO_PAGE=1, RMAP: 0

7	6	5	4	3	2	1	0
PC					0		
rwh					r		

Field	Bits	Type	Description
PC	7:3	rwh	**Port control for port Pn pin m** This bit field determines the port n pin m functionality according to the coding table (see Table "PCm Coding"). 10_H **PP_GPO**: Push-pull general purpose output 11_H **PP_ALT_FUNC1**: Push-pull alternate function 1 12_H **PP_ALT_FUNC2**: Push-pull alternate function 2 13_H **PP_ALT_FUNC3**: Push-pull alternate function 3 14_H **PP_ALT_FUNC4**: Push-pull alternate function 4 15_H **PP_ALT_FUNC5**: Push-pull alternate function 5 16_H **PP_ALT_FUNC6**: Push-pull alternate function 6 17_H **PP_ALT_FUNC7**: Push-pull alternate function 7 18_H **OD_GPO**: Open-drain general purpose output 19_H **OD_ALT_FUNC1**: Open-drain alternate function 1 $1A_H$ **OD_ALT_FUNC2**: Open-drain alternate function 2 $1B_H$ **OD_ALT_FUNC3**: Open-drain alternate function 3 $1C_H$ **OD_ALT_FUNC4**: Open-drain alternate function 4 $1D_H$ **OD_ALT_FUNC5**: Open-drain alternate function 5 $1E_H$ **OD_ALT_FUNC6**: Open-drain alternate function 6 $1F_H$ **OD_ALT_FUNC7**: Open-drain alternate function 7
0	2:0	r	**Reserved** Read as 0; must be written with 0.

Table 1790 Reset values of Pn_IOCRm (m=0-7;n=0-3)

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	
After Boot-FW Value	$000–\ 0000_B$	Post-reset value of this register is either 0x10 or 0x00 depending on PMS_PAD_CON.SBTRIST value

39.4.41 Port Pn pad driver mode 0 register

Pn_PDR0 (n=0-3) Offset address: 090_H+n*8

Port Pn pad driver mode 0 register Reset values see: Table 1791

PAGE: IO_PAGE=2, RMAP: 0

7	6	5	4	3	2	1	0
PL1		**PD1**		**PL0**		**PD0**	
rw		rw		rw		rw	

Field	Bits	Type	Description
PDm (m=0-1)	4*m+1:4*m	rw	**Pad driver mode for port Pn pin m**
PLm (m=0-1)	4*m+3:4*m +2	rw	**Pad level selection for port Pn pin m**

Table 1791 Reset values of Pn_PDR0 (n=0-3)

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.42 Port Pn pad driver mode 2 register

Pn_PDR2 (n=0-3) Offset address: 091_H+n*8

Port Pn pad driver mode 2 register Reset values see: Table 1792

PAGE: IO_PAGE=2, RMAP: 0

7	6	5	4	3	2	1	0
PL3		PD3		PL2		PD2	
rw		rw		rw		rw	

Field	Bits	Type	Description
PDm (m=2-3)	4*m-7:4*m-8	rw	**Pad driver mode for port Pn pin m**
PLm (m=2-3)	4*m-5:4*m-6	rw	**Pad level selection for port Pn pin m**

Table 1792 Reset values of Pn_PDR2 (n=0-3)

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.43 Port Pn pad driver mode 4 register

Pn_PDR4 (n=0-3) Offset address: 092_H+n*8

Port Pn pad driver mode 4 register Reset values see: Table 1793

PAGE: IO_PAGE=2, RMAP: 0

7	6	5	4	3	2	1	0
PL5		PD5		PL4		PD4	
rw		rw		rw		rw	

Field	Bits	Type	Description
PDm (m=4-5)	4*m-15:4*m-16	rw	**Pad driver mode for port Pn pin m**
PLm (m=4-5)	4*m-13:4*m-14	rw	**Pad level selection for port Pn pin m**

Table 1793 Reset values of Pn_PDR4 (n=0-3)

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.44 Port Pn pad driver mode 6 register

Pn_PDR6 (n=0-3) Offset address: 093_H+n*8

Port Pn pad driver mode 6 register Reset values see: Table 1794

PAGE: IO_PAGE=2, RMAP: 0

7	6	5	4	3	2	1	0
PL7		PD7		PL6		PD6	
rw		rw		rw		rw	

Field	Bits	Type	Description
PDm (m=6-7)	4*m-23:4*m-24	rw	**Pad driver mode for port Pn pin m**
PLm (m=6-7)	4*m-21:4*m-22	rw	**Pad level selection for port Pn pin m**

Table 1794 Reset values of Pn_PDR6 (n=0-3)

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.45 Port Pn pin function decision control register

Pn_PDISC (n=0-3) Offset address: 095_H+n*8

Port Pn pin function decision control register Reset values see: Table 1795

PAGE: IO_PAGE=2, RMAP: 0

7	6	5	4	3	2	1	0
PDIS7	PDIS6	PDIS5	PDIS4	PDIS3	PDIS2	PDIS1	PDIS0
rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
PDISm (m=0-7)	m	rw	**Pin function decision control for port Pn pin m** This bit selects the function of the port pad. 0_B Pad Pn.m is enabled 1_B Pad Pn.m is disabled

Table 1795 Reset values of Pn_PDISC (n=0-3)

Reset type	Reset value	Note
LVD Reset	FF_H	
Generated Reset	FF_H	

39.4.46 Port Pn wake enable request register

Pn_WKEN: Setting a bit field in this register will set the corresponding bit in the PMS register

- P0_WKEN[7:0] is mapped to PMS_PAD_WKEN[7:0]
- P1_WKEN[7:0] is mapped to PMS_PAD_WKEN[15:8]
- P2_WKEN[4:0] is mapped to PMS_PAD_WKEN[20:16]
- P2_WKEN[7:5] is mapped to PMS_PAD_WKEN[28:26]
- P3_WKEN[4:0] is mapped to PMS_PAD_WKEN[25:21]
- P3_WKEN[7:5] is mapped to PMS_PAD_WKEN[31:29]

Pn_WKEN (n=0-3) Offset address: 090_H+n*8

Port Pn wake enable request register LVD Reset value: 00_H

PAGE: IO_PAGE=3, RMAP: 0

7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
Pm (m=0-7)	m	rwh	**Port Pn pin m enable wakeup function** This port pin can be used to wakeup the SCR. 0_B Writing 0 has no effect. 1_B Port Pn pin m wakeup function is requested to be enabled

39.4.47 Port Pn wake enable status register

The wake enable status reflects the status of the wakeup feature of the corresponding port pin. It is set when the corresponding WKEN is set to one and indicates that the wakeup feature for the port pin is available and STANDBY mode can be entered.

Note: This register is cleared by setting SCUSCR_STDBY.PMSWKPCLR.

- P0_WKENSTS[7:0] is mapped to PMS_PAD_WKENSTS[7:0]
- P1_WKENSTS[7:0] is mapped to PMS_PAD_WKENSTS[15:8]
- P2_WKENSTS[4:0] is mapped to PMS_PAD_WKENSTS[20:16]
- P2_WKENSTS[7:5] is mapped to PMS_PAD_WKENSTS[28:26]
- P3_WKENSTS[4:0] is mapped to PMS_PAD_WKENSTS[25:21]
- P3_WKENSTS[7:5] is mapped to PMS_PAD_WKENSTS[31:29]

Pn_WKENSTS (n=0-3) Offset address: 091_H+n*8

Port Pn wake enable status register LVD Reset value: 00_H

PAGE: IO_PAGE=3, RMAP: 0

7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0
rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
Pm (m=0-7)	m	rh	**Port Pn input pin m is enabled for wakeup** This input pin Pn.m, is enabled for wakeup. 0_B Port Pn.m wake feature is disabled 1_B Port Pn.m wake feature is enabled

39.4.48 Wakeup status register Port Pn

The wake status register contains the wakeup information provided by the wakeup circuitry of a dedicated pad. The read value is the same as PMS register PAD_WKSTS.

- P0_WKSTS[7:0] is mapped to PMS_PAD_WKSTS[7:0]
- P1_WKSTS[7:0] is mapped to PMS_PAD_WKSTS[15:8]
- P2_WKSTS[4:0] is mapped to PMS_PAD_WKSTS[20:16]
- P2_WKSTS[7:5] is mapped to PMS_PAD_WKSTS[28:26]
- P3_WKSTS[4:0] is mapped to PMS_PAD_WKSTS[25:21]
- P3_WKSTS[7:5] is mapped to PMS_PAD_WKSTS[31:29]

Pn_WKSTS (n=0-3) Offset address: 092_H+n*8

Wakeup status register Port Pn LVD Reset value: 00_H

PAGE: IO_PAGE=3, RMAP: 0

7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0
rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
Pm (m=0-7)	m	rh	**Port Pn pin m has caused the wakeup** This pin has caused the wakeup of the device. 0_B No wakeup trigger on port Pn.m 1_B Wakeup triggered on port Pn.m

39.4.49 Wake-Up CAN configuration register

WCAN_CFG Offset address: $0B0_H$

Wake-Up CAN configuration register Reset values see: Table 1796

PAGE: WCAN_PAGE=0, RMAP: 0

7	6	5	4	3	2	1	0
WCANRXDIS				CCE	SELWK_EN	0	WCAN_EN
rw				rw	rwh	r	rw

Field	Bits	Type	Description
WCAN_EN	0	rw	**WCAN enable** This bit enables or disables the clock to WCAN selective wake-up filter block. 0_B Disable clock to WCAN selective wake-up filter block 1_B Enable clock to WCAN selective wake-up filter block
SELWK_EN	2	rwh	**Selective wake-up enable** The selective wake-up enable function enables the protocol handler. The status of the protocol handler is also acknowledged through INTESTAT0.SWACK. The status of the Selective wakeup active is given in INTESTAT1.SWKSET. It also serves several functions when enabled. Allows wake-up source WUF to generate WUF_INT (see Figure "Wake-up frame detection"). Allows error counter to generate ERROR_INT (see Figure "Error counter"), indicated by INTESTAT0.ECOFL. Triggers ERROR_INT and wake-up if there are any changes to the message object registers, indicated by INTESTAT0.MORC. If a SYSERR occurs while SELWK_EN is enabled, then the Selective wake-up function shall be disabled (this bit field will be reset to 0). It is also indicated by INTESTAT0.ERSEL. 0_B Selective wake disabled 1_B Selective wake enabled
CCE	3	rw	**Configuration change enable** This bit field enables the configuration of the bit timing registers. It also acts as soft-reset to the selective wake-up filter block. It has to be set back to '0' once the bit timing registers are modified for desired baudrate. 0_B The bit timing registers (BTLn_CTRL), CDR configuration registers (CDR_CTRL, CDR_UPPER_CTRL, CDR_LOWER_CTRL) and CAN_FD_CTRL may only be read. All attempts to modify them will be ignored. 1_B The bit timing registers and CDR configuration registers may be read and written

(table continues...)

(continued)

Field	Bits	Type	Description
WCANRXDIS	7:4	rw	**WCAN receiver input select** 0_H **WCANRXDA**: WCAN receiver input WCANRXDA is selected 1_H **WCANRXDB**: WCAN receiver input WCANRXDB is selected 2_H **WCANRXDC**: WCAN receiver input WCANRXDC is selected 3_H **WCANRXDD**: WCAN receiver input WCANRXDD is selected 4_H **WCANRXDE**: WCAN receiver input WCANRXDE is selected 5_H **WCANRXDF**: WCAN receiver input WCANRXDF is selected 6_H **WCANRXDG**: WCAN receiver input WCANRXDG is selected 7_H **WCANRXDH**: WCAN receiver input WCANRXDH is selected 8_H **WCANRXDI**: WCAN receiver input WCANRXDI is selected 9_H **WCANRXDJ**: WCAN receiver input WCANRXDJ is selected A_H **WCANRXDK**: WCAN receiver input WCANRXDK is selected B_H **WCANRXDL**: WCAN receiver input WCANRXDL is selected C_H **WCANRXDM**: WCAN receiver input WCANRXDM is selected D_H **WCANRXDN**: WCAN receiver input WCANRXDN is selected E_H **WCANRXDO**: WCAN receiver input WCANRXDO is selected F_H **WCANRXDP**: WCAN receiver input WCANRXDP is selected
0	1	r	**Reserved** Read as 0; must be written with 0.

Table 1796 Reset values of WCAN_CFG

Reset type	Reset value	Note
LVD Reset	08_H	
Generated Reset	08_H	

39.4.50 Wake-Up CAN interrupt mask register

WCAN_INTMRSLT Offset address: $0B1_H$

Wake-Up CAN interrupt mask register Reset values see: Table 1797

PAGE: WCAN_PAGE=0, RMAP: 0

7	6	5	4	3	2	1	0
0				WUPMASK	WUFMASK	ERRMASK	CANTOMASK
r				rw	rw	rw	rw

Field	Bits	Type	Description
CANTOMASK	0	rw	**CAN time-out masking** This bit controls the interrupt masking for CANTO_INT interrupt. 0_B CAN time-out is masked - no interrupt is triggered 1_B CAN time-out is signaled as an interrupt

(table continues...)

(continued)

Field	Bits	Type	Description
ERRMASK	1	rw	**SYSERR masking** This bit controls the interrupt masking for SYSERR interrupt. 0_B SYSERR is masked - no interrupt is triggered 1_B SYSERR is signaled as an interrupt
WUFMASK	2	rw	**Wake-Up frame interrupt masking** This bit controls the interrupt masking for WUF_INT interrupt. 0_B WUF_INT is masked - no interrupt is triggered 1_B WUF_INT is signaled as an interrupt
WUPMASK	3	rw	**Wake-Up pattern detected interrupt masking** This bit controls the interrupt masking for WUP_INT interrupt. 0_B WUP_INT is masked - no interrupt is triggered 1_B WUP_INT is signaled as an interrupt
0	7:4	r	**Reserved** Read as 0; must be written with 0.

Table 1797 Reset values of WCAN_INTMRSLT

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.51 CDR control register

WCAN_CDR_CTRL Offset address: $0B2_H$

CDR control register Reset values see: Table 1799

PAGE: WCAN_PAGE=0, RMAP: 0

7	6	5	4	3	2	1	0
0				**ESF**		**0**	**CDREN**
r				rw		r	rw

Field	Bits	Type	Description
CDREN	0	rw	**Enable CDR** 0_B Disable CDR internal bit time measurement 1_B Enable CDR internal bit time measurement
ESF	3:2	rw	**Error scaling factor** This bitfield defines the number of acquisition cycles on which average on error is calculated. 00_B **ERR_SCALE_FACT8**: 8 01_B **ERR_SCALE_FACT16**: 16 (recommended) 10_B **ERR_SCALE_FACT32**: 32 11_B **ADAPT_AVRGE**: Adaptive averaging

(table continues...)

(continued)

Field	Bits	Type	Description
0	1, 7:4	r	**Reserved** Read as 0; must be written with 0.

Table 1798 Access mode restrictions of WCAN_CDR_CTRL sorted by descending priority

Mode name	Access mode		Description
WCAN_CFG.CCE == 1	rw	CDREN, ESF	
(default)	r	CDREN, ESF	

Table 1799 Reset values of WCAN_CDR_CTRL

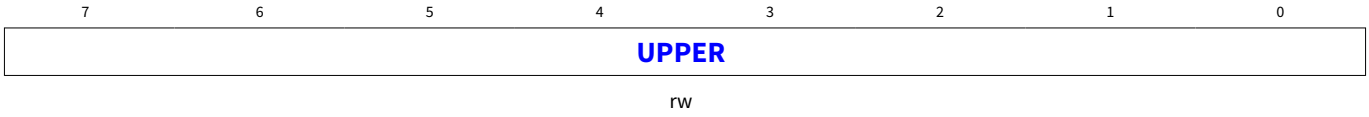
Reset type	Reset value	Note
LVD Reset	05_H	
Generated Reset	05_H	

39.4.52 CDR upper limit control register

WCAN_CDR_UPPER_CTRL Offset address: $0B3_H$

CDR upper limit control register Reset values see: Table 1801

PAGE: WCAN_PAGE=0, RMAP: 0



Field	Bits	Type	Description
UPPER	7:0	rw	**CDR upper limit** This bitfield defines the upper limit for CDR internal bit time measurement. It should be configured as +5% of BTL1_CTRL value (integer part).

Table 1800 Access mode restrictions of WCAN_CDR_UPPER_CTRL sorted by descending priority

Mode name	Access mode		Description
WCAN_CFG.CCE == 1	rw	UPPER	
(default)	r	UPPER	

Table 1801 Reset values of WCAN_CDR_UPPER_CTRL

Reset type	Reset value	Note
LVD Reset	69_H	
Generated Reset	69_H	

39.4.53 CDR lower limit control register

WCAN_CDR_LOWER_CTRL Offset address: $0B4_H$

CDR lower limit control register Reset values see: Table 1803

PAGE: WCAN_PAGE=0, RMAP: 0

7	6	5	4	3	2	1	0
LOWER							
rw							

Field	Bits	Type	Description
LOWER	7:0	rw	**CDR lower limit** This bitfield defines the lower limit for CDR internal bit time measurement. It should be configured as -5% of BTL1_CTRL value (integer part).

Table 1802 Access mode restrictions of WCAN_CDR_LOWER_CTRL sorted by descending priority

Mode name	Access mode		Description
WCAN_CFG.CCE == 1	rw	LOWER	
(default)	r	LOWER	

Table 1803 Reset values of WCAN_CDR_LOWER_CTRL

Reset type	Reset value	Note
LVD Reset	$5F_H$	
Generated Reset	$5F_H$	

39.4.54 CDR measured high register

WCAN_CDR_MEAS_HIGH Offset address: $0B5_H$

CDR measured high register Reset values see: Table 1804

PAGE: WCAN_PAGE=0, RMAP: 0

7	6	5	4	3	2	1	0
HIGH							
rh							

Field	Bits	Type	Description
HIGH	7:0	rh	**CDR internally measured high value** This bitfield shows the internally measured duration of a CAN bus bit as number of f_{CAN} clock cycles (integer part).

Table 1804 Reset values of WCAN_CDR_MEAS_HIGH

Reset type	Reset value	Note
LVD Reset	64_H	
Generated Reset	64_H	

39.4.55 CDR measured low register

WCAN_CDR_MEAS_LOW Offset address: $0B6_H$

CDR measured low register Reset values see: Table 1805

PAGE: WCAN_PAGE=0, RMAP: 0

7	6	5	4	3	2	1	0
0				**LOW**			
r				rh			

Field	Bits	Type	Description
LOW	5:0	rh	**CDR internally measured low value** This bitfield shows the internally measured duration of a CAN bus bit as number of f_{CAN} clock cycles (fractional part).
0	7:6	r	**Reserved** Read as 0; must be written with 0.

Table 1805 Reset values of WCAN_CDR_MEAS_LOW

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.56 Wake-Up CAN FD control register

WCAN_FD_CTRL Offset address: $0B7_H$

Wake-Up CAN FD control register Reset values see: Table 1807

PAGE: WCAN_PAGE=0, RMAP: 0

7	6	5	4	3	2	1	0
0				**FDFILT**			**FDEN**
r				rw			rw

Field	Bits	Type	Description
FDEN	0	rw	**CAN FD tolerant enable** This bitfield enables or disables the CAN FD tolerant feature. 0_B Disabled 1_B Enabled

(table continues...)

(continued)

Field	Bits	Type	Description
FDFILT	3:1	rw	**CAN FD filter time** This bitfield configures the filter time to detect the dominant bit field in FD phase. Refer to Table "CAN FD dominant bit filter time" for the FD filter time configuration for various baudrates. 000_B 13 (130 ns) 001_B 25 (250 ns) 010_B 48 (480 ns) 011_B 90 (900 ns) **others,** Reserved
0	7:4	r	**Reserved** Read as 0; must be written with 0.

Table 1806 Access mode restrictions of WCAN_FD_CTRL sorted by descending priority

Mode name	Access mode		Description
WCAN_CFG.CCE == 1	rw	FDEN, FDFILT	
(default)	r	FDEN, FDFILT	

Table 1807 Reset values of WCAN_FD_CTRL

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.57 Wake-Up CAN interrupt and event status register 0

WCAN_INTESTAT0 Offset address: $0B0_H$

Wake-Up CAN interrupt and event status register 0 Reset values see: Table 1808

PAGE: WCAN_PAGE=1, RMAP: 0

7	6	5	4	3	2	1	0
WUP	**ERSEL**	**MORC**	**ECOFL**	**CANTO**	**SYSERR**	**MODE**	**SWACK**
rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
SWACK	0	rh	**Selective wake-up enable acknowledge** This bit field indicates the acknowledgment of protocol handler enabled/disabled. 0_B Protocol handler is disabled and configuration of WUF is possible 1_B Protocol handler is enabled and configuration of WUF should not be done

(table continues...)

(continued)

Field	Bits	Type	Description
MODE	1	rh	**WCAN mode of operation** This bitfield indicates the WCAN mode of operation. 0_B Normal mode 1_B Standby mode
SYSERR	2	rh	**CAN wake-up system error** The SYSERR indicates a configuration error (MORC), and an error counter overflow condition (ECOFL). 0_B Selective wake mode is possible 1_B System error detected, WCAN enabling not possible. It also leads to SYSERR_INT interrupt trigger if enabled through ERRMASK.
CANTO	3	rh	**CAN time-out detection** This bit indicates that there was an inactive CAN bus on RXDCO pin within $t_{SILENCE}$ of 750 ms. See section "CAN Message Timeout". A software write to INTESTATCLR0.CANTOCLR is required to reset this bit. 0_B Normal operation 1_B CAN time-out detected. It also leads to CANTO_INT interrupt if enabled through CANTOMASK.
ECOFL	4	rh	**Error counter overflow** This bit field shows occurrence of error counter overflow. A set condition of this flag by hardware has higher priority than a clear through ECOFLCLR. 0_B Error counter has not overflowed (<=31) 1_B Error counter overflowed (>31)
MORC	5	rh	**Message object register changed** A set condition of this flag by hardware has higher priority than a clear through MORCCLR. 0_B Message object registers not modified after SELWK_EN is set 1_B Message object registers was modified after SELWK_EN is set
ERSEL	6	rh	**Error select** A set condition of this flag by hardware has higher priority than a clear through ERSELCLR. 0_B SYSERR is not set while SELWK_EN is enabled 1_B SYSERR is set while SELWK_EN is enabled
WUP	7	rh	**Wake-Up pattern detected** This bit field is set when a falling edge on RXD (or WUP) is detected, when WCAN is in standby mode. A software write to INTESTATCLR0.WUPCLR is required to clear this bit field. 0_B WUP not detected 1_B WUP detected. It also triggers WUP_INT interrupt when enabled through INTMRSLT.WUPMASK

Table 1808 Reset values of WCAN_INTESTAT0

Reset type	Reset value	Note
LVD Reset	00_H	

(table continues...)

Table 1808 (continued) Reset values of WCAN_INTESTAT0

Reset type	Reset value	Note
Generated Reset	00_H	

39.4.58 Wake-Up CAN interrupt and event status register 1

WCAN_INTESTAT1 Offset address: $0B1_H$

Wake-Up CAN interrupt and event status register 1 Reset values see: Table 1809

PAGE: WCAN_PAGE=1, RMAP: 0

7	6	5	4	3	2	1	0
0	RXDS	RXDF	FDF	CANSIL	SWKSET	SYNC	WUF
r	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
WUF	0	rh	**Wake-Up frame detection (as defined in ISO)** This bit shows the status whether a wake-up frame has been detected. 0_B No wake-up frame detected 1_B Wake-Up frame detected. It also leads to WUF_INT interrupt trigger if enabled through WUFMASK.
SYNC	1	rh	**Synchronization (at least one CAN frame without fail must have been received)** This bit is set when a valid classical CAN frame is detected and is updated after every received classical CAN frame. This flag is not updated when a CAN FD frame is received. Clearing this flag through SYNCCLR has higher priority than a set condition by hardware. 0_B WCAN function not working or not synchronous to CAN bus 1_B Valid CAN frame received, WCAN function is synchronous to CAN bus
SWKSET	2	rh	**Selective wake activity** 0_B Selective wake is not active 1_B Selective wake is activated
CANSIL	3	rh	**CAN bus silent time during selective wake operation** This bit is set if there is no activity on RXDC0 pin on the CAN bus after $t_{SILENCE}$ of 750 ms expired. If any activity is detected on the bus, this bit is reset automatically. $t_{SILENCE}$ is fixed at 750 ms. This bit can also be cleared through bit WCAN_INTESCLR.CANSILCLR. A set condition of this bit has higher priority than a clear through CANSILCLR. 0_B not exceeded $t_{SILENCE}$ not exceeded 1_B is exceeded $t_{SILENCE}$ is exceeded

(table continues...)

(continued)

Field	Bits	Type	Description
FDF	4	rh	**FD frame reception** This bit indicates reception of CAN FD frame. After reception of the CAN FD frame is completed, this bit is cleared to '0'. 0_B No action 1_B FD frame is being received
RXDF	5	rh	**Filtered receive data input** This bit shows the logic state of the filtered received data input signal (RXD_filt). 0_B Dominant 1_B Recessive
RXDS	6	rh	**Sampled receive data input** This bit shows the logic state of the sampled received data input signal by the protocol handler. 0_B Dominant 1_B Recessive
0	7	r	**Reserved** Read as 0; must be written with 0.

Table 1809 Reset values of WCAN_INTESTAT1

Reset type	Reset value	Note
LVD Reset	60_H	
Generated Reset	60_H	

39.4.59 Wake-Up CAN error counter register

WCAN_FRMERRCNT Offset address: $0B2_H$

Wake-Up CAN error counter register Reset values see: Table 1810

PAGE: WCAN_PAGE=1, RMAP: 0

7	6	5	4	3	2	1	0
ECNTCLR	**FDEC_DIS**	**ECNT**					
w	rw	rh					

Field	Bits	Type	Description
ECNT	5:0	rh	**SWK CAN frame error counter** If WCAN_FD_CTRL.FDEN = 0, the CAN frame error counter is incremented when an error message or CAN FD frame is received, and decremented on a reception of a correct classical CAN frame, if ECNT>0. If FDEN = 1, the behavior of the error counter is based on FDEC_DIS bit configuration. See section "Error Counter".

(table continues...)

(continued)

Field	Bits	Type	Description
FDEC_DIS	6	rw	**Error count disable on CAN FD Frame** This bit is used to describe the behavior of the error counter for FD frames. The value of this bit has no effect if WCAN_FD_CTRL.FDEN = 0. ***Note**: When an FD frame is detected, the error is checked only until the FDF bit of the FD frame.* 0_B Error counter is incremented or decremented based on correctness of the bitfields in arbitration phase of the FD frame 1_B Error counter is active only for classical CAN frames, and it is frozen for CAN FD frames
ECNTCLR	7	w	**Error counter clear bit** Writing a 1 to this bit clears the error counter FRMERRCNT.ECNT.

Table 1810 Reset values of WCAN_FRMERRCNT

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.60 Wake-Up CAN interrupt and event status clear register 0

WCAN_INTESCLR0 Offset address: $0B3_H$

Wake-Up CAN interrupt and event status clear register 0 Reset values see: Table 1811

PAGE: WCAN_PAGE=1, RMAP: 0

7	6	5	4	3	2	1	0
WUPCLR	ERSELCLR	MORCCLR	ECOFLCLR	CANTOCLR	SYSERRCLR	0	
w	w	w	w	w	w	r	

Field	Bits	Type	Description
SYSERRCLR	2	w	**CAN WUP system error clear bit** Writing a 1 to this bit clears the corresponding status bit INTESTAT0.SYSERR.
CANTOCLR	3	w	**CAN time-out detection clear bit** Writing a 1 to this bit clears the corresponding status bit INTESTAT0.CANTO.
ECOFLCLR	4	w	**Error counter overflow clear bit** Writing a 1 to this bit clears the corresponding status bit INTESTAT0.ECOFL.
MORCCLR	5	w	**Message object register changed clear bit** Writing a 1 to this bit clears the corresponding status bit INTESTAT0.MORC.

(table continues...)

(continued)

Field	Bits	Type	Description
ERSELCLR	6	w	**Error select clear bit** Writing a 1 to this bit clears the corresponding status bit INTESTAT0.ERSEL.
WUPCLR	7	w	**WUP clear bit** Writing a 1 to this bit clears the corresponding status bit INTESTAT0.WUP.
0	1:0	r	**Reserved** Read as 0; must be written with 0.

Table 1811 Reset values of WCAN_INTESCLR0

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.61 Wake-Up CAN interrupt and event status clear register 1

WCAN_INTESCLR1

Offset address: $0B4_H$

Wake-Up CAN interrupt and event status clear register 1

Reset values see: Table 1812

PAGE: WCAN_PAGE=1, RMAP: 0

7	6	5	4	3	2	1	0
0				CANSILCLR	0	SYNCCLR	WUFCLR
r				w	r	w	w

Field	Bits	Type	Description
WUFCLR	0	w	**Wake-up frame detection flag clear bit** Writing a 1 to this bit clears the corresponding status bit INTESTAT1.WUF.
SYNCCLR	1	w	**Synchronization clear bit** Writing a 1 to this bit clears the corresponding status bit INTESTAT1.SYNC.
CANSILCLR	3	w	**CAN silent time during SWK operation clear bit** Writing a 1 to this bit clears the corresponding status bit INTESTAT1.CANSIL.
0	2, 7:4	r	**Reserved** Read as 0; must be written with 0.

Table 1812 Reset values of WCAN_INTESCLR1

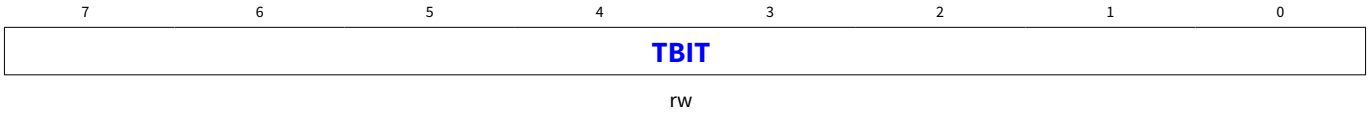
Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.62 Bit timing logic 1 control register

WCAN_BTL1_CTRL Offset address: $0B5_H$

Bit timing logic 1 control register Reset values see: Table 1814

PAGE: WCAN_PAGE=1, RMAP: 0



Field	Bits	Type	Description
TBIT	7:0	rw	**Number of time quanta in a bit time** This bitfield represents the number of time quanta in a bit time. The time quanta is based on f_{CAN} frequency.

Table 1813 Access mode restrictions of WCAN_BTL1_CTRL sorted by descending priority

Mode name	Access mode		Description
WCAN_CFG.CCE == 1	rw	TBIT	
(default)	r	TBIT	

Table 1814 Reset values of WCAN_BTL1_CTRL

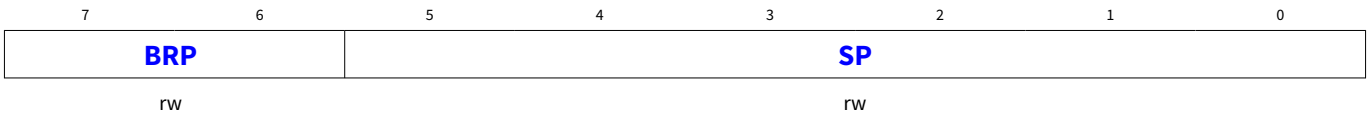
Reset type	Reset value	Note
LVD Reset	64_H	
Generated Reset	64_H	

39.4.63 Bit timing logic 2 control register

WCAN_BTL2_CTRL Offset address: $0B6_H$

Bit timing logic 2 control register Reset values see: Table 1816

PAGE: WCAN_PAGE=1, RMAP: 0



Field	Bits	Type	Description
SP	5:0	rw	**Sample point position** Represents the sample point position. The sample point position is calculated as given in section "Filter and Sample Point Calculation". 33_H represents the sample point position of 79.68 % (~80 %). The value shall be within $2C_H$ to 38_H (this means: 67.5 % to 87.5 %).

(table continues...)

(continued)

Field	Bits	Type	Description
BRP	7:6	rw	**Baudrate pre-scaler** This bitfield defines the scaling factor to derive one time-quanta. 00_B **DIV1**: Division by 1 (100 MHz) 01_B **DIV2**: Division by 2 (50 MHz) 10_B **DIV4**: Division by 4 (25 MHz) 11_B **DIV8**: Division by 8 (12.5 MHz)

Table 1815 Access mode restrictions of WCAN_BTL2_CTRL sorted by descending priority

Mode name	Access mode		Description
WCAN_CFG.CCE == 1	rw	BRP, SP	
(default)	r	BRP, SP	

Table 1816 Reset values of WCAN_BTL2_CTRL

Reset type	Reset value	Note
LVD Reset	73_H	
Generated Reset	73_H	

39.4.64 Message data length code control register

WCAN_DLC_CTRL Offset address: $0B7_H$

Message data length code control register Reset values see: Table 1817

PAGE: WCAN_PAGE=1, RMAP: 0

7	6	5	4	3	2	1	0
0				**DLC**			
r				rw			

Field	Bits	Type	Description
DLC	3:0	rw	**Data length code** Valid values for the data length are 0 to 8. DLC > 8 will be considered as 8 data bytes, but the truncation will not be visible in this bitfield upon read.
0	7:4	r	**Reserved** Read as 0; must be written with 0.

Table 1817 Reset values of WCAN_DLC_CTRL

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.65 Message identifier control register 0

WCAN_ID0_CTRL Offset address: $0B0_H$

Message identifier control register 0 Reset values see: Table 1818

PAGE: WCAN_PAGE=2, RMAP: 0

7	6	5	4	3	2	1	0
ID							
rw							

Field	Bits	Type	Description
ID	7:0	rw	**CAN Identifier** Identifier of a standard message (ID[28:18]) or an extended message (ID[28:0]). For standard identifiers, bits ID[17:0] are "don't care". This bitfield specifies ID7 to ID0.

Table 1818 Reset values of WCAN_ID0_CTRL

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.66 Message identifier control register 1

WCAN_ID1_CTRL Offset address: $0B1_H$

Message identifier control register 1 Reset values see: Table 1819

PAGE: WCAN_PAGE=2, RMAP: 0

7	6	5	4	3	2	1	0
ID							
rw							

Field	Bits	Type	Description
ID	7:0	rw	**CAN identifier** Identifier of a standard message (ID[28:18]) or an extended message (ID[28:0]). For standard identifiers, bits ID[17:0] are "don't care". This bitfield specifies ID15 to ID8.

Table 1819 Reset values of WCAN_ID1_CTRL

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.67 Message identifier control register 2

WCAN_ID2_CTRL Offset address: $0B2_H$

Message identifier control register 2 Reset values see: Table 1820

PAGE: WCAN_PAGE=2, RMAP: 0

7	6	5	4	3	2	1	0
ID							
rw							

Field	Bits	Type	Description
ID	7:0	rw	**CAN identifier** Identifier of a standard message (ID[28:18]) or an extended message (ID[28:0]). For standard identifiers, bits ID[17:0] are "don't care". This bitfield specifies ID23 to ID16.

Table 1820 Reset values of WCAN_ID2_CTRL

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.68 Message identifier control register 3

WCAN_ID3_CTRL Offset address: $0B3_H$

Message identifier control register 3 Reset values see: Table 1821

PAGE: WCAN_PAGE=2, RMAP: 0

7	6	5	4	3	2	1	0
0	ID					0	IDE
r	rw					r	rw

Field	Bits	Type	Description
IDE	0	rw	**Identified extension bit** 0_B Standard frame with 11 bits identified 1_B Extended frame with 29 bits identified
ID	6:2	rw	**CAN identifier** Identifier of a standard message (ID[28:18]) or an extended message (ID[28:0]). For standard identifiers, bits ID[17:0] are "don't care". This bitfield specifies ID28 to ID24.
0	1, 7	r	**Reserved** Read as 0; must be written with 0.

Table 1821 Reset values of WCAN_ID3_CTRL

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.69 Message identifier mask register 0

WCAN_MASK_ID0_CTRL Offset address: $0B4_H$

Message identifier mask register 0 Reset values see: Table 1822

PAGE: WCAN_PAGE=2, RMAP: 0

7	6	5	4	3	2	1	0
MID							
rw							

Field	Bits	Type	Description
MID	7:0	rw	**Mask for message identifier** Mask to filter incoming messages with standard identifiers (MID[28:18]) or extended identifiers (MID[28:0]). For standard identifiers, bits MID[17:0] are "don't care". If a specific bit in MID is set, the corresponding bit in the ID is considered for the ID matching evaluation, otherwise it is considered as "don't care". This bitfield specifies MID7 to MID0.

Table 1822 Reset values of WCAN_MASK_ID0_CTRL

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.70 Message identifier mask register 1

WCAN_MASK_ID1_CTRL Offset address: $0B5_H$

Message identifier mask register 1 Reset values see: Table 1823

PAGE: WCAN_PAGE=2, RMAP: 0

7	6	5	4	3	2	1	0
MID							
rw							

Field	Bits	Type	Description
MID	7:0	rw	**Mask for message identifier** Mask to filter incoming messages with standard identifiers (MID[28:18]) or extended identifiers (MID[28:0]). For standard identifiers, bits MID[17:0] are "don't care". If a specific bit in MID is set, the corresponding bit in the ID is considered for the ID matching evaluation, otherwise it is considered as "don't care". This bitfield specifies MID15 to MID8.

Table 1823 Reset values of WCAN_MASK_ID1_CTRL

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.71 Message identifier mask register 2

WCAN_MASK_ID2_CTRL Offset address: $0B6_H$

Message identifier mask register 2 Reset values see: Table 1824

PAGE: WCAN_PAGE=2, RMAP: 0

7	6	5	4	3	2	1	0
MID							
rw							

Field	Bits	Type	Description
MID	7:0	rw	**Mask for message identifier** Mask to filter incoming messages with standard identifiers (MID[28:18]) or extended identifiers (MID[28:0]). For standard identifiers, bits MID[17:0] are "don't care". If a specific bit in MID is set, the corresponding bit in the ID is considered for the ID matching evaluation, otherwise it is considered as "don't care". This bitfield specifies MID23 to MID16.

Table 1824 Reset values of WCAN_MASK_ID2_CTRL

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.72 Message identifier mask register 3

WCAN_MASK_ID3_CTRL Offset address: $0B7_H$

Message identifier mask register 3 Reset values see: Table 1825

PAGE: WCAN_PAGE=2, RMAP: 0

7	6	5	4	3	2	1	0
0	MID					0	
r	rw					r	

Field	Bits	Type	Description
MID	6:2	rw	**Mask for message identifier** Mask to filter incoming messages with standard identifiers (MID[28:18]) or extended identifiers (MID[28:0]). For standard identifiers, bits MID[17:0] are "don't care". If a specific bit in MID is set, the corresponding bit in the ID is considered for the ID matching evaluation, otherwise it is considered as "don't care". This bitfield specifies MID28 to MID24.
0	1:0, 7	r	**Reserved** Read as 0; must be written with 0.

Table 1825 Reset values of WCAN_MASK_ID3_CTRL

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.73 DATA control register n

Data control registers, DATAn_CTRL (n = 0-7) contain the data filter bits. See figure “wake-up frame detection” for flow of acceptance filtering it determines if a wake-up frame has been detected. It is a match if a single bit is equally set at the same bit position in the incoming data and DATAn_CTRL (n = 0-7) register.

WCAN_DATAn_CTRL (n=0-7) Offset address: $0B0_H$+n

DATA control register n Reset values see: Table 1826

PAGE: WCAN_PAGE=3, RMAP: 0

7	6	5	4	3	2	1	0
DATA							
rw							

Field	Bits	Type	Description
DATA	7:0	rw	**CAN Data Byte** These bitfields specify the data bits 7 to 0 that will be compared with corresponding bits of $(n+1)^{th}$ data byte of the received CAN frame.

Table 1826 Reset values of WCAN_DATAn_CTRL (n=0-7)

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.74 Page register for WCAN

WCAN_PAGE Offset address: $0B8_H$

Page register for WCAN Reset values see: Table 1827

PAGE: X, RMAP: 0

7	6	5	4	3	2	1	0
OP		**STNR**		**0**	**PAGE**		
w		w		r	rwh		

Field	Bits	Type	Description
PAGE	2:0	rwh	**Page bits** When written, the value indicates the new page address. When read, the value indicates the currently active page = addr[y:x+1]. 000_B **PAGE0**: Selection is PAGE0 001_B **PAGE1**: Selection is PAGE1 010_B **PAGE2**: Selection is PAGE2 011_B **PAGE3**: Selection is PAGE3 100_B **PAGE4**: Selection is PAGE4 101_B **PAGE5**: Selection is PAGE5 110_B **PAGE6**: Selection is PAGE6 111_B **PAGE7**: Selection is PAGE7
STNR	5:4	w	**Storage number** This number indicates which storage bitfield is the target of the operation defined by bit OP. If OP = 10_B, the contents of PAGE are saved in STx before being overwritten with the new value. If OP = 11_B, the contents of PAGE are overwritten by the contents of STx. The value written to bitfield PAGE is ignored. 00_B **MOD_ST0**: ST0 is selected 01_B **MOD_ST1**: ST1 is selected 10_B **MOD_ST2**: ST2 is selected 11_B **MOD_ST3**: ST3 is selected
OP	7:6	w	**Operation** 00_B **PAGE_MANUAL0**: Manual page mode. The value of STNR is ignored and PAGE is directly written. 01_B **PAGE_MANUAL1**: Manual page mode. The value of STNR is ignored and PAGE is directly written 10_B **PAGE_SAVE**: New page programming with automatic page saving. The value written to the bit positions of PAGE is stored. In parallel, the former contents of PAGE are saved in the storage bit field STx indicated by STNR. 11_B **PAGE_RESTORE**: Automatic restore page action. The value written to the bit positions PAGE is ignored and instead, PAGE is overwritten by the contents of the storage bit field STx indicated by STNR.
0	3	r	**Reserved** Read as 0; must be written with 0.

Table 1827 Reset values of WCAN_PAGE

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.75 Serial channel control register

UART_SCON Offset address: $0B9_H$

Serial channel control register Kernel Reset value: 00_H

PAGE: X, RMAP: 0

7	6	5	4	3	2	1	0
SM01		SM2	REN	TB8	RB8	TI	RI
rw		rw	rw	rw	rwh	rwh	rwh

Field	Bits	Type	Description
RI	0	rwh	**Receive interrupt flag** This is set by hardware at the end of the 8th bit on mode 0, or at the half point of the stop bit in modes 1, 2, and 3. This bit can also be set by software. Must be cleared by software. ***Note**: In case of an ongoing reception and a mode change at the very same time, the flag RI will have an undefined behavior, as it is dependent from the point in time of the mode change.*
TI	1	rwh	**Transmit interrupt flag** This is set by hardware at the end of the 8th bit in mode 0, or at the beginning of the stop bit in modes 1, 2, and 3. This bit can also be set by software. Must be cleared by software. ***Note**: If a transmission has been started and TI is still 0, a further write to UART_SBUF will destroy the ongoing transmission. If TI is set to 1, the next transmission can be started.*
RB8	2	rwh	**Serial port receiver bit 9** In modes 2 and 3, this is the 9th data bit received. In mode 1, this is the stop bit received. In mode 0, this bit is not used.
TB8	3	rw	**Serial port transmitter bit 9** In modes 2 and 3, this is the 9th data bit sent.
REN	4	rw	**Enable receiver of serial port** ***Note**: The operating mode of the UART must not be changed while bit REN is set (or while a transmission is in progress). Before changing the mode of the UART, bit REN shall be cleared to 0.* 0_B Serial reception is disabled 1_B Serial reception is enabled

(table continues...)

(continued)

Field	Bits	Type	Description
SM2	5	rw	**Enable serial port multiprocessor communication in modes 2 and 3** In mode 2 or 3, if SM2 is set to 1, RI will not be activated if the received 9th data bit (RB8) is 0. In mode 1, if SM2 is set to 1, RI will not be activated if a valid stop bit (RB8) was not received. In mode 0, SM2 should be 0. ***Note**: The operating mode of the UART must not be changed while bit REN is set or while a transmission is in progress. Before changing the mode of the UART, make sure that no transmissions are in progress, and that bit REN is cleared to 0.*
SM01	7:6	rw	**Serial port operating mode selection** ***Note**: The operating mode of the UART must not be changed while bit REN is set or while a transmission is in progress. Before changing the mode of the UART, make sure that no transmissions are in progress, and that bit REN is cleared to 0.* 00_B **MODE_0**: 8-bit shift register, fixed baud rate ($^{fPCLK}/_{2}$) 01_B **MODE_1**: 8-bit UART, variable baud rate 10_B **MODE_2**: 9-bit UART, fixed baud rate ($^{fPCLK}/_{64}$ or $^{fPCLK}/_{32}$) 11_B **MODE_3**: 9-bit UART, variable baud rate

39.4.76 Serial data buffer

***Note**: Register UART_SBUF may only be written to when no transmission is in progress (a previously commenced transmission is finished when bit UART_SCON.TI is set to 1). Otherwise, an ongoing transmission may be corrupted.*

UART_SBUF Offset address: $0BA_H$

Serial data buffer Kernel Reset value: 00_H

PAGE: X, RMAP: 0

7	6	5	4	3	2	1	0
VAL							
rwh							

Field	Bits	Type	Description
VAL	7:0	rwh	**Serial interface buffer register**

39.4.77 Baudrate timer/reload register, low byte

LIN_BGL Offset address: $0BB_H$

Baudrate timer/reload register, low byte Reset values see: Table 1828

PAGE: X, RMAP: 0

7	6	5	4	3	2	1	0
BR_VALUE			FD_SEL				
rwh			rw				

Field	Bits	Type	Description
FD_SEL	4:0	rw	**Fractional divider selection** Selects the fractional divider to be n/32, where n is the value of FD_SEL, and is in the range of 0 to 31. For example, writing 0001_B to FD_SEL selects the fractional divider to be 1/32. ***Note**: Fractional divider has no effect if the 11-bit BR_VALUE = 000_H.*
BR_VALUE	7:5	rwh	**Baudrate timer/reload value** The lower three bits of the 11-bit baudrate timer/reload value. See also description in BGH register. ***Note**: BR_VALUE must be written if, and only if, BCON.R = 0.* *Writing to this register updates the reload value, while reading this register returns current timer value. Therefore it is not possible to directly read back the written value.*

Table 1828 Reset values of LIN_BGL

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.78 Baudrate control register

LIN_BCON Offset address: $0BC_H$

Baudrate control register Reset values see: Table 1829

PAGE: X, RMAP: 0

7	6	5	4	3	2	1	0
BGSEL		TTDIS	BRDIS	BRPRE			R
rw		rw	rw	rw			rw

Field	Bits	Type	Description
R	0	rw	**Baudrate generator run control bit** ***Note**: BR_VALUE must be written if, and only if, R = 0.* 0_B Baudrate generator disabled 1_B Baudrate generator enabled
BRPRE	3:1	rw	**pre-scaler bit** Selects the input clock for f_{DIV}, which is derived from the peripheral clock. 000_B **DIV**: 1 $f_{DIV} = f_{PCLK}$ 001_B **DIV2**: 2 $f_{DIV} = {}^{fPCLK}/_{2}$ 010_B **DIV4**: 4 $f_{DIV} = {}^{fPCLK}/4$ 011_B **DIV8**: 8 $f_{DIV} = {}^{fPCLK}/_{8}$ 100_B **DIV16**: 16 $f_{DIV} = {}^{fPCLK}/_{16}$ 101_B **DIV32**: 32 $f_{DIV} = {}^{fPCLK}/_{32}$ **others,** Reserved
BRDIS	4	rw	**Baudrate detection disable** ***Note**: It is recommended to toggle bit BRDIS after the reception of each LIN frame (complete or erroneous frame) to avoid a wrong Break field detection in noisy environments (i.e. spikes on the LIN bus).* 0_B Break/Sync detection is enabled 1_B Break/Sync detection is disabled
TTDIS	5	rw	**Temperature Trimming of Peripheral Clock Source Disable** 0_B Software signal to PMS, trimming is allowed 1_B Software signal to PMS, communication is still ongoing, trimming of oscillator is not allowed.
BGSEL	7:6	rw	**Baudrate select for detection** For different values of BGSEL, the baud rate range for detection is defined by the following formula: ${}^{fPCLK}/_{(2184*2\text{^}BGSEL)}$ < baud rate range < ${}^{fPCLK}/_{(74*2\text{^}BGSEL)}$; with BGSEL element of {00_B, 01_B, 10_B, 11_B} See table "BGSEL Bit Field Definition for Different Input Frequencies" for bit field BGSEL definition for different input frequencies.

Table 1829 Reset values of LIN_BCON

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.79 LIN status register

LIN_STAT Offset address: $0BD_H$

LIN status register Reset values see: Table 1830

PAGE: X, RMAP: 0

7	6	5	4	3	2	1	0
0	**SYNEN**	**ERRSYN**	**EOFSYN**	**BRK**	**0**		
r	rw	rwh	rwh	rwh	r		

Field	Bits	Type	Description
BRK	3	rwh	**Break field flag** This bit is set by hardware and can only be cleared by software. 0_B Break field is not detected 1_B Break field is detected
EOFSYN	4	rwh	**End of SYN byte interrupt flag** This bit is set by hardware and can only be cleared by software. 0_B End of SYN byte is not detected 1_B End of SYN byte is detected
ERRSYN	5	rwh	**SYN byte error interrupt flag** This bit is set by hardware and can only be cleared by software. 0_B Error is not detected in SYN byte 1_B Error is detected in SYN byte
SYNEN	6	rw	**End of SYN byte and SYN byte error interrupts enable** 0_B End of SYN byte and SYN byte error interrupts are not enabled 1_B End of SYN byte and SYN byte error interrupts are enabled
0	2:0, 7	r	**Reserved** Read as 0; must be written with 0.

Table 1830 Reset values of LIN_STAT

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.80 Baudrate timer/reload register, high byte

LIN_BGH Offset address: $0BE_H$

Baudrate timer/reload register, high byte Reset values see: Table 1831

PAGE: X, RMAP: 0

7	6	5	4	3	2	1	0
BR_VALUE							
rwh							

Field	Bits	Type	Description
BR_VALUE	7:0	rwh	**Baudrate timer/reload value** The upper 8 bits of the 11-bit baudrate timer/reload value. When the 11-bit BR_VALUE is 000_H, the baudrate timer is bypassed. ***Note**: BR_VALUE must be written if, and only if, BCON.R = 0.* *Writing to this register updates the reload value, while reading this register returns current timer value. Therefore it is not possible to directly read back the written value.*

Table 1831 Reset values of LIN_BGH

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.81 Timer 2 instance 1 control register

T2CCU1_T2_CON Offset address: $0BF_H$

Timer 2 instance 1 control register Reset values see: Table 1832

PAGE: T2CCU1_PAGE=0, RMAP: 0

7	6	5	4	3	2	1	0
TF2	**EXF2**	**0**		**EXEN2**	**TR2**	**C_T2**	**CP_RL2**
rwh	rwh	r		rw	rwh	rw	rw

Field	Bits	Type	Description
CP_RL2	0	rw	**Capture/Reload select** 0_B Reload upon overflow or upon negative/positive transition at pin T2EX (when EXEN2 = 1) 1_B Capture timer 2 data register contents on the negative/positive transition at pin T2EX, provided EXEN2 = 1. The negative or positive transition at pin T2EX is selected by bit EDGESEL in register T2MOD.
C_T2	1	rw	**Timer or counter select** 0_B Timer function selected 1_B Count upon negative edge at pin T2

(table continues...)

(continued)

Field	Bits	Type	Description
TR2	2	rwh	**Timer 2 run control** This bit can be set or cleared by software to start or stop the timer T2. It can also be set by hardware via input T2EX, if bit T2MOD.T2RHEN is set to 1, or via a synchronized start with the CCT timer of the T2CCU1. ***Note**: Writing a one to this bit while the timer is running has no effect.* 0_B Timer T2 is halted 1_B Timer T2 is running
EXEN2	3	rw	**Timer 2 external enable control** 0_B External events are disabled 1_B External events are enabled in capture/reload mode
EXF2	6	rwh	**Timer 2 external flag** In capture/reload mode, this bit is set by hardware when a negative/ positive transition occurs at pin T2EX, if bit EXEN2 = 1. This bit must be cleared by software. ***Note**: When bit DCEN = 1 in auto-reload mode, no interrupt request to the core is generated.*
TF2	7	rwh	**Timer 2 overflow/underflow flag** Set by a timer 2 overflow/underflow. Must be cleared by software.
0	5:4	r	**Reserved** Read as 0; must be written with 0.

Table 1832 Reset values of T2CCU1_T2_CON

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.82 Timer 2 instance 1 mode register

T2CCU1_T2_MOD Offset address: $0C0_H$

Timer 2 instance 1 mode register Reset values see: Table 1833

PAGE: T2CCU1_PAGE=0, RMAP: 0

7	6	5	4	3	2	1	0
T2REGS	T2RHEN	EDGESEL	PREN	T2PRE			DCEN
rw	rw	rw	rw	rw			rw

Field	Bits	Type	Description
DCEN	0	rw	**Up/Down counter enable** 0_B Up/Down counter function is disabled 1_B Up/Down counter function is enabled and controlled by pin T2EX (Up = 1, Down = 0)
T2PRE	3:1	rw	**Timer 2 pre-scaler bit** Selects the input clock for timer 2 which is derived from the peripheral clock. 000_B **PCLK**: 1 $f_{T2} = f_{PCLK}$ 001_B **PCLK_DIV2**: 2 $f_{T2} = {}^{fPCLK}/_{2}$ 010_B **PCLK_DIV4**: 4 $f_{T2} = {}^{fPCLK}/_{4}$ 011_B **PCLK_DIV8**: 8 $f_{T2} = {}^{fPCLK}/_{8}$ 100_B **PCLK_DIV16**: 16 $f_{T2} = {}^{fPCLK}/_{16}$ 101_B **PCLK_DIV32**: 32 $f_{T2} = {}^{fPCLK}/_{32}$ 110_B **PCLK_DIV64**: 64 $f_{T2} = {}^{fPCLK}/_{64}$ 111_B **PCLK_DIV128**: 128 $f_{T2} = {}^{fPCLK}/_{128}$
PREN	4	rw	**pre-scaler enable** 0_B pre-scaler is disabled, and the 2 or 12 divider takes effect 1_B pre-scaler is enabled (see T2PRE bit), and the 2 or 12 divider is bypassed
EDGESEL	5	rw	**Edge select in capture mode/reload mode** 0_B The falling edge at pin T2EX is selected 1_B The rising edge at pin T2EX is selected
T2RHEN	6	rw	**Timer 2 external start enable** 0_B Timer 2 external start is disabled 1_B Timer 2 external start is enabled
T2REGS	7	rw	**Edge select for timer 2 external start** 0_B The falling edge at pin T2EX is selected 1_B The rising edge at pin T2EX is selected

Table 1833 Reset values of T2CCU1_T2_MOD

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.83 Timer 2 instance 1 reload/capture register, low byte

T2CCU1_T2_RC2L Offset address: $0C1_H$

Timer 2 instance 1 reload/capture register, low byte Reset values see: Table 1834

PAGE: T2CCU1_PAGE=0, RMAP: 0

Field	Bits	Type	Description
RC2	7:0	rwh	**Reload/Capture value [7:0]** If T2_CON.CP_RL2 = 0, these contents are loaded into the timer register upon an overflow condition. If T2_CON.CP_RL2 = 1, this register is loaded with the current timer count upon a negative/positive transition at pin T2EX when EXEN2 = 1.

Table 1834 Reset values of T2CCU1_T2_RC2L

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.84 Timer 2 instance 1 reload/capture register, high byte

T2CCU1_T2_RC2H Offset address: $0C2_H$

Timer 2 instance 1 reload/capture register, high byte Reset values see: Table 1835

PAGE: T2CCU1_PAGE=0, RMAP: 0

Field	Bits	Type	Description
RC2	7:0	rwh	**Reload/Capture value [15:8]** If T2_CON.CP_RL2 = 0, these contents are loaded into the timer register upon an overflow condition. If T2_CON.CP_RL2 = 1, this register is loaded with the current timer count upon a negative/positive transition at pin T2EX when EXEN2 = 1.

Table 1835 Reset values of T2CCU1_T2_RC2H

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.85 Timer 2 instance 1, low byte

T2CCU1_T2_T2L Offset address: $0C3_H$

Timer 2 instance 1, low byte Reset values see: Table 1836

PAGE: T2CCU1_PAGE=0, RMAP: 0

7	6	5	4	3	2	1	0
THL2							
rwh							

Field	Bits	Type	Description
THL2	7:0	rwh	**Timer 2 value [7:0]** These bits indicate the lower 8 bits of the current 16-bit timer value.

Table 1836 Reset values of T2CCU1_T2_T2L

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.86 Timer 2 instance 1, high byte

T2CCU1_T2_T2H Offset address: $0C4_H$

Timer 2 instance 1, high byte Reset values see: Table 1837

PAGE: T2CCU1_PAGE=0, RMAP: 0

7	6	5	4	3	2	1	0
THL2							
rwh							

Field	Bits	Type	Description
THL2	7:0	rwh	**Timer 2 value [15:8]** These bits indicate the upper 8 bits of the current 16-bit timer value.

Table 1837 Reset values of T2CCU1_T2_T2H

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.87 Timer 2 instance 1 control register 1

Register T21_CON1 (T2CCU1_PAGE) is used to enable the external interrupt and the overflow interrupt.

T2CCU1_T2_CON1 Offset address: $0C5_H$

Timer 2 instance 1 control register 1 Reset values see: Table 1838

PAGE: T2CCU1_PAGE=0, RMAP: 0

7	6	5	4	3	2	1	0
EXTINSEL				INSEL		TF2EN	EXF2EN
rw				rw		rw	rw

Field	Bits	Type	Description
EXF2EN	0	rw	**External interrupt enable** 0_B External interrupt is disabled 1_B External interrupt is enabled
TF2EN	1	rw	**Overflow/Underflow interrupt enable** 0_B Overflow/underflow interrupt is disabled 1_B Overflow/underflow interrupt is enabled
INSEL	3:2	rw	**Timer 2 instance 1 input select** 00_B **T21A**: Timer 2 instance 1 input T21A is selected 01_B **T21B**: Timer 2 instance 1 input T21B is selected 10_B **T21C**: Timer 2 instance 1 input T21C is selected 11_B **T21D**: Timer 2 instance 1 input T21D is selected
EXTINSEL	7:4	rw	**Timer 2 instance 1 external input select** 0_H **T21EXA**: Timer 2 instance 1 input T21EXA is selected 1_H **T21EXB**: Timer 2 instance 1 input T21EXB is selected 2_H **T21EXC**: Timer 2 instance 1 input T21EXC is selected 3_H **T21EXD**: Timer 2 instance 1 input T21EXD is selected 4_H **T21EXE**: Timer 2 instance 1 input T21EXE is selected 5_H **T21EXF**: Timer 2 instance 1 input T21EXF is selected 6_H **T21EXG**: Timer 2 instance 1 input T21EXG is selected 7_H **T21EXH**: Timer 2 instance 1 input T21EXH is selected 8_H **T21CLKRTC**: Timer 2 instance 1 input PMS RTC clock is selected 9_H **T21CLK70K**: Timer 2 instance 1 input 70 kHz is selected A_H **T21CLK32K**: Timer 2 instance 1 input from SCR_P1.2(P33.10) is selected B_H **T21EXL**: Timer 2 instance 1 input T21EXL is selected C_H **T21EXM**: Timer 2 instance 1 input T21EXM is selected D_H **T21EXN**: Timer 2 instance 1 input T21EXN is selected E_H **T21EXO**: Timer 2 instance 1 input T21EXO is selected F_H **T21EXP**: Timer 2 instance 1 input T21EXP is selected

Table 1838 Reset values of T2CCU1_T2_CON1

Reset type	Reset value	Note
LVD Reset	03_H	
Generated Reset	03_H	

39.4.88 T2CCU1 capture/compare enable register

The register is addressable via T2CCU1_PAGE.

***Note**: The T2CCU1 output is only actively driven on enabling the compare function.*

T2CCU1_CCEN Offset address: $0BF_H$

T2CCU1 capture/compare enable register Reset values see: Table 1839

PAGE: T2CCU1_PAGE=1, RMAP: 0

7	6	5	4	3	2	1	0
CCM3		**CCM2**		**CCM1**		**CCM0**	
rw		rw		rw		rw	

Field	Bits	Type	Description
CCM0	1:0	rw	**Capture/Compare enable for channel 0** 00_B **CC_DIS**: Capture/Compare disabled 01_B **CAPTURE_ACT_EDGE**: Capture on active edge at pin T2CC0 10_B **COMP_EN**: Compare enabled 11_B **CAPTURE_ON_WRITE**: Capture on write operation into register CC0L
CCM1	3:2	rw	**Capture/Compare enable for channel 1** 00_B **CC_DIS**: Capture/Compare disabled 01_B **CAPTURE_ACT_EDGE**: Capture on active edge at pin T2CC1 10_B **COMP_EN**: Compare enabled 11_B **CAPTURE_ON_WRITE**: Capture on write operation into register CC1L
CCM2	5:4	rw	**Capture/Compare enable for channel 2** 00_B **CC_DIS**: Capture/Compare disabled 01_B **CAPTURE_ACT_EDGE**: Capture on active edge at pin T2CC2 10_B **COMP_EN**: Compare enabled 11_B **CAPTURE_ON_WRITE**: Capture on write operation into register CC2L
CCM3	7:6	rw	**Capture/Compare enable for channel 3** 00_B **CC_DIS**: Capture/Compare disabled 01_B **CAPTURE_ACT_EDGE**: Capture on active edge at pin T2CC3 10_B **COMP_EN**: Compare enabled 11_B **CAPTURE_ON_WRITE**: Capture on write operation into register CC3L

Table 1839 Reset values of T2CCU1_CCEN

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.89 T2CCU1 capture/compare time base select register

The register is addressable via T2CCU1_PAGE.

T2CCU1_CCTBSEL Offset address: $0C0_H$

T2CCU1 capture/compare time base select register Reset values see: Table 1840

PAGE: T2CCU1_PAGE=1, RMAP: 0

7	6	5	4	3	2	1	0
CASC	CCTTOV	CCTB5	CCTB4	CCTB3	CCTB2	CCTB1	CCTB0
rw	rwh	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
CCTBx (x=0-5)	x	rw	**Channel x time base select** 0_B Selects timer 2 1_B Selects capture/compare timer (CCT)
CCTTOV	6	rwh	**Trigger CCT timer overflow event** On set, this bit is held for one PCLK, then cleared by hardware. Reading this bit always returns 0. 0_B No action 1_B CCT overflow event is triggered immediately
CASC	7	rw	**Cascade timers** 0_B Timer 2 and CCT timer operate independently 1_B Timer 2 is cascaded with CCT timer

Table 1840 Reset values of T2CCU1_CCTBSEL

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.90 T2CCU1 capture/compare timer reload register low

The register is addressable via T2CCU1_PAGE.

T2CCU1_CCTRELL Offset address: $0C1_H$

T2CCU1 capture/compare timer reload register low Reset values see: Table 1841

PAGE: T2CCU1_PAGE=1, RMAP: 0

7	6	5	4	3	2	1	0
CCTREL							
rw							

Field	Bits	Type	Description
CCTREL	7:0	rw	**Capture/Compare timer low byte reload value [7:0]** The contents of the registers are loaded into the CCT timer low byte register upon an overflow condition.

Table 1841 Reset values of T2CCU1_CCTRELL

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.91 T2CCU1 capture/compare timer reload register high

The register is addressable via T2CCU1_PAGE.

T2CCU1_CCTRELH Offset address: $0C2_H$

T2CCU1 capture/compare timer reload register high Reset values see: Table 1842

PAGE: T2CCU1_PAGE=1, RMAP: 0

7	6	5	4	3	2	1	0
CCTREL							
rw							

Field	Bits	Type	Description
CCTREL	7:0	rw	**Capture/Compare timer high byte reload value [15:8]** The contents of the registers are loaded into the CCT timer high byte register upon an overflow condition.

Table 1842 Reset values of T2CCU1_CCTRELH

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.92 T2CCU1 capture/compare timer register low

The register is addressable via T2CCU1_PAGE.

T2CCU1_CCTL Offset address: $0C3_H$

T2CCU1 capture/compare timer register low Reset values see: Table 1843

PAGE: T2CCU1_PAGE=1, RMAP: 0

7	6	5	4	3	2	1	0
CCT							
rwh							

Field	Bits	Type	Description
CCT	7:0	rwh	**Capture/Compare timer low byte value [7:0]** The register represents the low byte counter value of the Capture/ Compare Timer. When CCT is not running, a write to CCTL updates the lower byte of timer. To (re-)initialize CCT when it is running, CCTH has to be written first. On write to CCTL, the values CCTH and CCTL will be updated to the timer, which then counts from this value.

Table 1843 Reset values of T2CCU1_CCTL

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.93 T2CCU1 capture/compare timer register high

The register is addressable via T2CCU1_PAGE.

T2CCU1_CCTH Offset address: $0C4_H$

T2CCU1 capture/compare timer register high Reset values see: Table 1844

PAGE: T2CCU1_PAGE=1, RMAP: 0

7	6	5	4	3	2	1	0
CCT							
rwh							

Field	Bits	Type	Description
CCT	7:0	rwh	**Capture/Compare timer high byte value [15:8]** The register represents the high byte counter value of the Capture/ Compare Timer. When CCT is not running, a write to CCTH updates the upper byte of timer. To (re-)initialize CCT when it is running, CCTH has to be written first. On write to CCTL, the values CCTH and CCTL will be updated to the timer, which then counts from this value.

Table 1844 Reset values of T2CCU1_CCTH

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.94 T2CCU1 capture/compare timer control register

The register is addressable via T2CCU1_PAGE.

T2CCU1_CCTCON Offset address: $0C5_H$

T2CCU1 capture/compare timer control register Reset values see: Table 1845

PAGE: T2CCU1_PAGE=1, RMAP: 0

7	6	5	4	3	2	1	0
CCTPRE				CCTOVF	CCTOVEN	TIMSYN	CCTST
rw				rwh	rw	rw	rw

Field	Bits	Type	Description
CCTST	0	rw	**Capture/Compare timer start/stop control** ***Note:*** *Setting CCTST while the bit was already set will not disturb the running CCT timer operation.* 0_B Stop capture/compare timer 1_B Start capture/compare timer. If bit TIMSYN = 1 and CCTST was 0, start also timer 2 (if it was not already running).
TIMSYN	1	rw	**Enable synchronized timer starts** 0_B Disable synchronized timer starts 1_B Enable synchronized timer starts
CCTOVEN	2	rw	**Capture/Compare timer overflow interrupt enable** Set to enable CCT overflow interrupt.
CCTOVF	3	rwh	**Capture/Compare timer overflow flag** Set by hardware on CCT overflow. This bit has to be cleared by software. ***Note:*** *In case of software and hardware access to the flag at the same time, hardware will have higher priority.*

(table continues...)

(continued)

Field	Bits	Type	Description
CCTPRE	7:4	rw	**Compare/Compare timer pre-scaler bitfield** Selects the input clock for CCT, which is derived from the peripheral clock. ***Note:*** *It is not recommended to change CCTPRE when CCTST = 1.* 0_H **DIV**: 1 $f_{CCT} = f_{PCLK}$ 1_H **DIV2**: 2 $f_{CCT} = f_{PCLK} / 2$ 2_H **DIV4**: 4 $f_{CCT} = f_{PCLK} / 4$ 3_H **DIV8**: 8 $f_{CCT} = f_{PCLK} / 8$ 4_H **DIV16**: 16 $f_{CCT} = f_{PCLK} / 16$ 5_H **DIV32**: 32 $f_{CCT} = f_{PCLK} / 32$ 6_H **DIV64**: 64 $f_{CCT} = f_{PCLK} / 64$ 7_H **DIV128**: 128 $f_{CCT} = f_{PCLK} / 128$ 8_H **DIV256**: 256 $f_{CCT} = f_{PCLK} / 256$ 9_H **DIV512**: 512 $f_{CCT} = f_{PCLK} / 512$ A_H **DIV1024**: 1024 $f_{CCT} = f_{PCLK} / 1024$ B_H **DIV2048**: 2048 $f_{CCT} = f_{PCLK} / 2048$

Table 1845 Reset values of T2CCU1_CCTCON

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.95 T2CCU1 compare shadow register

The register is addressable via T2CCU1_PAGE.

T2CCU1_COSHDW Offset address: $0BF_H$

T2CCU1 compare shadow register Reset values see: Table 1846

PAGE: T2CCU1_PAGE=2, RMAP: 0

7	6	5	4	3	2	1	0
ENSHDW	**TXOV**	**COOUT5**	**COOUT4**	**COOUT3**	**COOUT2**	**COOUT1**	**COOUT0**
rwh	rw	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
COOUTx (x=0-5)	x	rwh	**Compare channel x output** **ENSHDW = 0:** Any access to bit COOUTx accesses the corresponding bit of the internal 6-bit compare output register. Writing to bit COOUTx will change the corresponding bit of the internal compare output register. Reading from bit COOUTx returns the value of the bit in the internal compare output register. The content of the internal compare output register is output on the corresponding T2CCx pins when a valid compare match occurs for a channel in compare mode 1 or concurrent compare mode. ***Note**: A concurrent write operation to clear bit ENSHDW and set/ clear bit(s) COOUTx is effective for initializing the internal compare output register.* **ENSHDW = 1:** In this case, accessing bit COOUTx accesses the actual SFR bits. If COOUTx is set, the contents of register CCx will be transferred to corresponding channel x internal compare register. On set, COOUTx is held for one PCLK, then cleared automatically by hardware. ***Note**: A concurrent write operation to set bit ENSHDW and bit(s) COOUTx (and clear bit TXOV) is effective for starting shadow transfer(s).*
TXOV	6	rw	**Enable shadow transfer on timer overflow** Set to enable for all compare channels, transfer of shadow value in T2CCU1_CCx to the corresponding internal compare registers on overflow of their respective timer base.
ENSHDW	7	rwh	**Enable shadow transfer with COOUTx** Clear to enable access of the internal compare output register via the COOUTx bits. Set to enable use of COOUTx bits for transfer of shadow value in T2CCU1_CCx to the corresponding channel x internal compare register. ***Note**: When setting bit TXOV, bit ENSHDW will be cleared by internal hardware. To ensure bit ENSHDW is set, user must ensure bit TXOV is cleared.*

Table 1846 Reset values of T2CCU1_COSHDW

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.96 T2CCU1 capture/compare register x low (x=0-2)

The register is addressable via T2CCU1_PAGE.

T2CCU1_CCxL (x=0-2) Offset address: $0C0_H$+x*2

T2CCU1 capture/compare register x low Reset values see: Table 1847

PAGE: T2CCU1_PAGE=2, RMAP: 0

7	6	5	4	3	2	1	0
CCVALL							
rwh							

Field	Bits	Type	Description
CCVALL	7:0	rwh	**Capture/Compare low byte value for channel x** **Capture:** On read, CCVALL returns the low byte of the last captured value. **Compare:** CCVALL holds the low byte written. CCx acts as a shadow register to the channel x internal compare register. The value written is only transferred to the internal compare register when the corresponding bit T2CCU1_COSHDW.COOUTx is being set (with ENSHDW = 1), or on overflow of the channel timer base. The read value always return the last written value.

Table 1847 Reset values of T2CCU1_CCxL (x=0-2)

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.97 T2CCU1 capture/compare register x high (x=0-2)

The register is addressable via T2CCU1_PAGE.

T2CCU1_CCxH (x=0-2) Offset address: $0C1_H$+x*2

T2CCU1 capture/compare register x high Reset values see: Table 1848

PAGE: T2CCU1_PAGE=2, RMAP: 0

7	6	5	4	3	2	1	0
CCVALH							
rwh							

Field	Bits	Type	Description
CCVALH	7:0	rwh	**Capture/Compare high byte value for channel x** **Capture:** On read, CCVALH returns the high byte of the last captured value. **Compare:** CCVALH holds the high byte written. CCx acts as a shadow register to the channel x internal compare register. The value written is only transferred to the internal compare register when the corresponding bit T2CCU1_COSHDW.COOUTx is being set (with ENSHDW = 1), or on overflow of the channel timer base. The read value always return the last written value.

Table 1848 Reset values of T2CCU1_CCxH (x=0-2)

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.98 T2CCU1 compare control register

The register is addressable via T2CCU1_PAGE.

T2CCU1_COCON Offset address: $0BF_H$

T2CCU1 compare control register Reset values see: Table 1849

PAGE: T2CCU1_PAGE=3, RMAP: 0

7	6	5	4	3	2	1	0
CCM5	CCM4	CM5F	CM4F	POLB	POLA	COMOD	
rw	rw	rwh	rwh	rw	rw	rw	

Field	Bits	Type	Description
COMOD	1:0	rw	**Compare mode control (for compare channels)** 00_B Compare mode 0 selected 01_B Compare mode 1 selected 10_B Concurrent compare mode selected 11_B Reserved
POLA	2	rw	**Compare active level for channel group A** In compare mode 0, the bit defines the active level of the outputs of the compare channels belonging to the group. 0_B High on compare match 1_B Low on compare match
POLB	3	rw	**Compare active level for channel group B** In compare mode 0, the bit defines the active level of the outputs of the compare channels belonging to the group. 0_B High on compare match 1_B Low on compare match

(table continues...)

(continued)

Field	Bits	Type	Description
CM4F	4	rwh	**Compare channel 4 interrupt flag** Set on a compare match event on compare channel 4. This bit has to be cleared by software. ***Note**: In case of software and hardware access to a flag bit at the same time, hardware will have higher priority.*
CM5F	5	rwh	**Compare channel 5 interrupt flag** Set on a compare match event on compare channel 5. This bit has to be cleared by software. ***Note**: In case of software and hardware access to a flag bit at the same time, hardware will have higher priority.*
CCM4	6	rw	**Compare enable for channel 4** ***Note**: Compare channels 4 and 5 should be enabled in a separate step after setting the compare mode via bits COMOD (and active level POLA/B if compare mode 0). This is to ensure defined compare output according to the selected mode on starting the respective compare channel.* ***Note**: The interrupt enable for compare channel 4 is controlled by the respective bit of SCU SFR MODIEN.CM4EN.* 0_B Compare disabled 1_B Compare enabled
CCM5	7	rw	**Compare enable for channel 5** ***Note**: Compare channels 4 and 5 should be enabled in a separate step after setting the compare mode via bits COMOD (and active level POLA/B if compare mode 0). This is to ensure defined compare output according to the selected mode on starting the respective compare channel.* ***Note**: The interrupt enable for compare channel 5 is controlled by the respective bit of SCU SFR MODIEN.CM5EN.* 0_B Compare disabled 1_B Compare enabled

Table 1849 Reset values of T2CCU1_COCON

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.99 T2CCU1 capture/compare register x low (x=3-5)

The register is addressable via T2CCU1_PAGE.

T2CCU1_CCxL (x=3-5) Offset address: $0C0_H$+(x-3)*2

T2CCU1 capture/compare register x low Reset values see: Table 1850

PAGE: T2CCU1_PAGE=3, RMAP: 0

7	6	5	4	3	2	1	0
CCVALL							
rwh							

Field	Bits	Type	Description
CCVALL	7:0	rwh	**Capture/Compare low byte value for channel x** **Capture:** On read, CCVALL returns the low byte of the last captured value. **Compare:** CCVALL holds the low byte written. CCx acts as a shadow register to the channel x internal compare register. The value written is only transferred to the internal compare register when the corresponding bit T2CCU1_COSHDW.COOUTx is being set (with ENSHDW = 1), or on overflow of the channel timer base. The read value always return the last written value. ***Note**: For T2CCU1_CC4L/H and T2CCU1_CC5L/H, only the compare function is available.*

Table 1850 Reset values of T2CCU1_CCxL (x=3-5)

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.100 T2CCU1 capture/compare register x high (x=3-5)

The register is addressable via T2CCU1_PAGE.

T2CCU1_CCxH (x=3-5) Offset address: $0C1_H$+(x-3)*2

T2CCU1 capture/compare register x high Reset values see: Table 1851

PAGE: T2CCU1_PAGE=3, RMAP: 0

7	6	5	4	3	2	1	0
CCVALH							
rwh							

Field	Bits	Type	Description
CCVALH	7:0	rwh	**Capture/Compare high byte value for channel x** **Capture:** On read, CCVALH returns the high byte of the last captured value. **Compare:** CCVALH holds the high byte written. CCx acts as a shadow register to the channel x internal compare register. The value written is only transferred to the internal compare register when the corresponding bit T2CCU1_COSHDW.COOUTx is being set (with ENSHDW = 1), or on overflow of the channel timer base. The read value always return the last written value. ***Note**: For T2CCU1_CC4L/H and T2CCU1_CC5L/H, only the compare function is available.*

Table 1851 Reset values of T2CCU1_CCxH (x=3-5)

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.101 T2CCU1 capture/compare timer dead-time control register low

The register is addressable via T2CCU1_PAGE.

T2CCU1_CCTDTCL Offset address: $0C1_H$

T2CCU1 capture/compare timer dead-time control register low

Reset values see: Table 1852

PAGE: T2CCU1_PAGE=4, RMAP: 0

7	6	5	4	3	2	1	0
DTM							
rw							

Field	Bits	Type	Description
DTM	7:0	rw	**Dead-time value** This 8-bit value defines the programmable delay between switching from the passive state to the active state of the selected outputs. The switching from the active state to the passive state is not delayed.

Table 1852 Reset values of T2CCU1_CCTDTCL

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.102 T2CCU1 capture/compare timer dead-time control register high

The register is addressable via T2CCU1_PAGE.

T2CCU1_CCTDTCH Offset address: $0C2_H$

T2CCU1 capture/compare timer dead-time control register high Reset values see: Table 1853

PAGE: T2CCU1_PAGE=4, RMAP: 0

7	6	5	4	3	2	1	0
DTRES	**DTR2**	**DTR1**	**DTR0**	**DTLEV**	**DTE2**	**DTE1**	**DTE0**
rwh	rh	rh	rh	rw	rw	rw	rw

Field	Bits	Type	Description
DTEx (x=0-2)	x	rw	**Dead-time enable for channel pair x** ***Note**: This bit shall only be set after enabling the corresponding channels in compare mode 0 with CCT timer as the time base.* 0_B Dead-time generation is disabled 1_B Dead-time generation is enabled for channel pair x. The corresponding outputs switch from the passive state to the active state (according to compare status) with the delay programmed in DTM.
DTLEV	3	rw	**Dead-time level** This bit defines the active level of all DTCx during dead-time counting. 0_B Low (switches are active high) 1_B High (switches are active low)
DTRx (x=0-2)	x+4	rh	**Dead-time run indication bits** 0_B The value of the corresponding dead-time counter is 0 1_B The value of the corresponding dead-time counter is not 0
DTRES	7	rwh	**Reset dead-time counters** On set, this bit is held for one PCLK, then cleared by hardware. Reading this bit always returns 0. 0_B No action 1_B The three dead-time counters are reset to 0

Table 1853 Reset values of T2CCU1_CCTDTCH

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.103 Page register for T2CCU1

T2CCU1_PAGE Offset address: $0C6_H$

Page register for T2CCU1 Reset values see: Table 1854

PAGE: X, RMAP: 0

7	6	5	4	3	2	1	0
OP		STNR		0	PAGE		
w		w		r	rwh		

Field	Bits	Type	Description
PAGE	2:0	rwh	**Page bits** When written, the value indicates the new page address. When read, the value indicates the currently active page = addr[y:x+1]. 000_B **PAGE0**: Selection is PAGE0 001_B **PAGE1**: Selection is PAGE1 010_B **PAGE2**: Selection is PAGE2 011_B **PAGE3**: Selection is PAGE3 100_B **PAGE4**: Selection is PAGE4 101_B **PAGE5**: Selection is PAGE5 110_B **PAGE6**: Selection is PAGE6 111_B **PAGE7**: Selection is PAGE7
STNR	5:4	w	**Storage number** This number indicates which storage bitfield is the target of the operation defined by bit OP. If OP = 10_B, the contents of PAGE are saved in STx before being overwritten with the new value. If OP = 11_B, the contents of PAGE are overwritten by the contents of STx. The value written to bitfield PAGE is ignored. 00_B **MOD_ST0**: ST0 is selected 01_B **MOD_ST1**: ST1 is selected 10_B **MOD_ST2**: ST2 is selected 11_B **MOD_ST3**: ST3 is selected
OP	7:6	w	**Operation** 00_B **PAGE_MANUAL0**: Manual page mode. The value of STNR is ignored and PAGE is directly written. 01_B **PAGE_MANUAL1**: Manual page mode. The value of STNR is ignored and PAGE is directly written 10_B **PAGE_SAVE**: New page programming with automatic page saving. The value written to the bit positions of PAGE is stored. In parallel, the former contents of PAGE are saved in the storage bit field STx indicated by STNR. 11_B **PAGE_RESTORE**: Automatic restore page action. The value written to the bit positions PAGE is ignored and instead, PAGE is overwritten by the contents of the storage bit field STx indicated by STNR.
0	3	r	**Reserved** Read as 0; must be written with 0.

Table 1854 Reset values of T2CCU1_PAGE

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.104 ADCOMP control register

ADCOMP_CON Offset address: $0C7_H$

ADCOMP control register Reset values see: Table 1855

PAGE: X, RMAP: 0

7	6	5	4	3	2	1	0
SOC	**STSEL**		**CHSEL**				**EN_REQ**
rwh	rw		rw				rw

Field	Bits	Type	Description
EN_REQ	0	rw	**ADC comparator enable request** The ADC clock source is NOT automatically enabled upon setting of this bit. It can enabled in SCUSCR_CLK register by setting DIV_REQ to any other value than 0xF to activate the 100 MHz HPOSC clock. It is required to wait some time for the ADCOMP unit to start after the request. The activation status can be read from ADCOMP_RESHSTAT.EN_STAT. Note: The time can vary from around 15 to 65 µs, depending if system was in RUN or in STANDBY0 or in STB1_70KHz mode prior to setting EN_REQ. In any case the application can poll on ADCOMP_RESHSTAT.EN_STAT to know, if the ADCOMP is ready or not. In case EN_REQ is cleared during ongoing conversion, then the ADC is disabled only after the currently running conversion is ended. 0_B ADCOMP unit is disabled 1_B ADCOMP unit is enabled
CHSEL	4:1	rw	**ADC channel select** This bit-field selects the channel and configures the 16:1 ADC multiplexer before a conversion is triggered. During an ongoing measurement (sampling and conversion) CHSEL shall be kept unchanged. If CHSEL is changed during an ongoing conversion, the result of the conversion is invalid. 0_H **CH0**: CH0 is selected … F_H **CH15**: CH15 is selected
STSEL	6:5	rw	**Sample time selection** Sampling time value is taken into account when SOC signal is set and latched locally inside ADC. If the bit is changed during an ongoing measurement, the new value will be taken into account at the next start of conversion. 00_B Minimal sample time is selected (80ns, default) 01_B Sample time of 100 ns is selected 10_B Sample time of 200 ns is selected 11_B Sample time of 400 ns is selected

(table continues...)

(continued)

Field	Bits	Type	Description
SOC	7	rwh	**ADC start of conversion request** This bit activates a new conversion. This bit is automatically cleared by hardware once the conversion has started. If EOC is set, setting SOC clears EOC. 0_B ADC conversion is not started 1_B ADC conversion is started

Table 1855 Reset values of ADCOMP_CON

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.105 ADCOMP result-low register

ADCOMP_RESL Offset address: $0C8_H$

ADCOMP result-low register Reset values see: Table 1856

PAGE: X, RMAP: 0

7	6	5	4	3	2	1	0
RES7_0							
rh							

Field	Bits	Type	Description
RES7_0	7:0	rh	**ADC conversion result** This register shows the current converted ADC result. Software should ensure that the result is read before starting the next conversion.

Table 1856 Reset values of ADCOMP_RESL

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.106 ADCOMP result-high and status register

ADCOMP_RESHSTAT Offset address: $0C9_H$

ADCOMP result-high and status register Reset values see: Table 1857

PAGE: X, RMAP: 0

7	6	5	4	3	2	1	0
0			EOC	EN_STAT	RES10_8		
r			rh	rh	rh		

Field	Bits	Type	Description
RES10_8	2:0	rh	**ADC conversion result** This register shows the current converted ADC result. Software should ensure that the result is read before starting the next conversion.
EN_STAT	3	rh	**ADC comparator enable status** This bit indicates that the ADC unit, the associated bandgap HPBG, and the clock required for the ADC unit are active, and a subsequent conversion can be triggered. It is set after some time after EN_REQ is set - time can vary from around 15 to 65 µs, depending if HPBG was enabled or not prior to setting EN_REQ. 0_B ADCOMP unit is disabled 1_B ADCOMP unit is enabled
EOC	4	rh	**ADC end of conversion / sampling indication** This bit indicates that a conversion is completed and the result can be read. A new conversion via SOC can be started only when the EOC from the previous conversion has been set, except in case of the first measurement where SOC can be started even if EOC is not set. This bit is automatically cleared when SOC is set. 0_B ADC conversion is currently on-going 1_B ADC conversion is completed.
0	7:5	r	**Reserved** Read as 0; must be written with 0.

Table 1857 Reset values of ADCOMP_RESHSTAT

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.107 Timer 0/1 control register

T01_TCON Offset address: $0CA_H$

Timer 0/1 control register Kernel Reset value: 00_H

PAGE: X, RMAP: X

7	6	5	4	3	2	1	0
TF1	**TR1**	**TF0**	**TR0**	**IE1**	**IT1**	**IE0**	**IT0**
rwh	rw	rwh	rw	rwh	rw	rwh	rw

Field	Bits	Type	Description
IT0	0	rw	**External interrupt 0 level/edge trigger control** 0_B Low level triggered external interrupt 0 is selected 1_B Falling edge triggered external interrupt 0 is selected

(table continues...)

(continued)

Field	Bits	Type	Description
IE0	1	rwh	**External interrupt 0 flag** Set by hardware when external interrupt 0 event is detected. Cleared by hardware when the processor vectors to interrupt routine. Can also be set/cleared by software. ***Note**:* *Note: If the interrupt is set to be level-sensitive, IE0 reflects the inverted logic level on EXINT0*
IT1	2	rw	**External interrupt 1 level/edge trigger control** 0_B Low level triggered external interrupt 1 is selected 1_B Falling edge triggered external interrupt 1 is selected
IE1	3	rwh	**External interrupt 1 flag** Set by hardware when external interrupt 1 event is detected. Cleared by hardware when the processor vectors to interrupt routine. Can also be set/cleared by software. ***Note**:* *Note: If the interrupt is set to be level-sensitive, IE1 reflects the inverted logic level on EXINT1*
TR0	4	rw	**Timer 0 run control** 0_B Timer is halted 1_B Timer runs
TF0	5	rwh	**Timer 0 overflow flag** Set by hardware on timer/counter 0 overflow. Cleared by hardware when processor vectors to interrupt routine. Can also be set/cleared by software.
TR1	6	rw	**Timer 1 run control** ***Note**:* *Timer 1 run control also affects TH0, if timer 0 operates in mode 3.* 0_B Timer is halted 1_B Timer runs
TF1	7	rwh	**Timer 1 overflow flag** Set by hardware on timer/counter 1 overflow. Cleared by hardware when processor vectors to interrupt routine. Can also be set/cleared by software. ***Note**:* *TF1 is set by TH0 instead, if timer 0 operates in mode 3.*

39.4.108 Timer mode register

T01_TMOD Offset address: $0CB_H$

Timer mode register Kernel Reset value: 00_H

PAGE: X, RMAP: X

7	6	5	4	3	2	1	0
GATE1	T1S	T1M		GATE0	T0S	T0M	
rw	rw	rw		rw	rw	rw	

Field	Bits	Type	Description
T0M	1:0	rw	**Timer 0 mode select bits** 00_B **TIMER_13B**: 13-bit timer (M8048 compatible mode) 01_B **TIMER_16B**: 16-bit timer 10_B **TIMER_8B_AUTO_REL**: 8-bit auto-reload timer 11_B **TIMER0_SPLIT**: Timer 0 is split into two halves. TL0 is an 8-bit timer controlled by the standard Timer 0 control bits, and TH0 is the other 8-bit timer controlled by the standard timer 1 control bits. TH1 and TL1 of timer 1 are held (timer 1 is stopped).
T0S	2	rw	**Timer 0 input selector** 0_B Input is from internal system clock 1_B Input is from T0 pin
GATE0	3	rw	**Timer 0 gate control** 0_B Timer 0 will only run if TCON.TR0 = 1 (software control) 1_B Timer 0 will only run if EXINT0 pin = 1 (hardware control) and TCON.TR0 is set
T1M	5:4	rw	**Timer 1 mode select bits** 00_B **TIMER_13B**: 13-bit timer (M8048 compatible mode) 01_B **TIMER_16B**: 16-bit timer 10_B **TIMER_8B_AUTO_REL**: 8-bit auto-reload timer 11_B **TIMER0_SPLIT**: Timer 0 is split into two halves. TL0 is an 8-bit timer controlled by the standard timer 0 control bits, and TH0 is the other 8-bit timer controlled by the standard timer 1 control bits. TH1 and TL1 of timer 1 are held (timer 1 is stopped).
T1S	6	rw	**Timer 1 input selector** 0_B Input is from internal system clock 1_B Input is from T1 pin
GATE1	7	rw	**Timer 1 gate control** 0_B Timer 1 will only run if TCON.TR1 = 1 (software control) 1_B Timer 1 will only run if EXINT1 pin = 1 (hardware control) and TCON.TR1 is set

39.4.109 Timer 0 low byte

T01_TL0

Timer 0 low byte

PAGE: X, RMAP: X

Offset address: $0CC_H$

Kernel Reset value: 00_H

7	6	5	4	3	2	1	0
VAL							
rwh							

Field	Bits	Type	Description
VAL	7:0	rwh	**Timer 0 low byte value** ***Note**: If TMOD.T0M is equal to:* *00_B: TL0 holds the 5-bit pre-scaler value.* *01_B: TL0 holds the lower 8-bit part of the 16-bit timer value.* *10_B: TL0 holds the 8-bit timer value.* *11_B: TL0 holds the 8-bit timer value.*

39.4.110 Timer 1 low byte

T01_TL1 Offset address: $0CD_H$

Timer 1 low byte Kernel Reset value: 00_H

PAGE: X, RMAP: X

7	6	5	4	3	2	1	0
VAL							
rwh							

Field	Bits	Type	Description
VAL	7:0	rwh	**Timer 1 low byte value** ***Note**: If TMOD.T1M is equal to:* *00_B: TL1 holds the 5-bit pre-scaler value.* *01_B: TL1 holds the lower 8-bit part of the 16-bit timer value.* *10_B: TL1 holds the 8-bit timer value.* *11_B: TL1 is not used.*

39.4.111 Timer 0 high byte

T01_TH0 Offset address: $0CE_H$

Timer 0 high byte Kernel Reset value: 00_H

PAGE: X, RMAP: X

7	6	5	4	3	2	1	0
VAL							
rwh							

Field	Bits	Type	Description
VAL	7:0	rwh	**Timer 0 high byte value** ***Note**: If TMOD.T0M is equal to:* *00_B: TH0 holds the 8-bit pre-scaler value.* *01_B: TH0 holds the higher 8-bit part of the 16-bit timer value.* *10_B: TH0 holds the 8-bit reload value.* *11_B: TH0 holds the 8-bit timer value.*

39.4.112 Timer 1 high byte

T01_TH1 Offset address: $0CF_H$

Timer 1 high byte Kernel Reset value: 00_H

PAGE: X, RMAP: X

7	6	5	4	3	2	1	0
VAL							
rwh							

Field	Bits	Type	Description
VAL	7:0	rwh	**Timer 1 high byte value** ***Note**: If TMOD.T1M is equal to:* *00_B: TH1 holds the 8-bit pre-scaler value.* *01_B: TH1 holds the higher 8-bit part of the 16-bit timer value.* *10_B: TH1 holds the 8-bit reload value.* *11_B: TH1 is not used.*

39.4.113 Program status word register

PSW Offset address: $0D0_H$

Program status word register Kernel Reset value: 00_H

PAGE: X, RMAP: X

7	6	5	4	3	2	1	0
CY	AC	F0	RS		OV	F1	P
rwh	rwh	rw	rw		rwh	rw	rh

Field	Bits	Type	Description
P	0	rh	**Parity flag** Set/cleared by hardware after each instruction to indicate an odd/even number of "one" bits in the accumulator, i.e. even parity.
F1	1	rw	**General purpose flag**

(table continues...)

(continued)

Field	Bits	Type	Description
OV	2	rwh	**Overflow flag** Used by arithmetic instructions.
RS	4:3	rw	**Register bank select** These bits are used to select one of the four register banks. 00_B **BANK0**: Bank 0 selected, data address 0x00 - 0x07 01_B **BANK1**: Bank 1 selected, data address 0x08 - 0x0F 10_B **BANK2**: Bank 2 selected, data address 0x10 - 0x17 11_B **BANK3**: Bank 3 selected, data address 0x18 - 0x1F
F0	5	rw	**General purpose flag**
AC	6	rwh	**Auxiliary carry flag** Used by instructions which execute BCD operations.
CY	7	rwh	**Carry flag** Used by arithmetic instructions.

39.4.114 Interrupt enable register 1

IEN1 Offset address: $0D1_H$

Interrupt enable register 1 Kernel Reset value: 00_H

PAGE: X, RMAP: X

7	6	5	4	3	2	1	0
ECCIP3	ECCIP2	ECCIP1	ECCIP0	EXM	EX2	ESSC	EWCAN
rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
EWCAN	0	rw	**Interrupt node XINTR6 enable** 0_B XINTR6 is disabled 1_B XINTR6 is enabled
ESSC	1	rw	**Interrupt node XINTR7 enable** 0_B XINTR7 is disabled 1_B XINTR7 is enabled
EX2	2	rw	**Interrupt node XINTR8 enable** 0_B XINTR8 is disabled 1_B XINTR8 is enabled
EXM	3	rw	**Interrupt node XINTR9 enable** 0_B XINTR9 is disabled 1_B XINTR9 is enabled
ECCIP0	4	rw	**Interrupt node XINTR10 enable** 0_B XINTR10 is disabled 1_B XINTR10 is enabled

(table continues...)

(continued)

Field	Bits	Type	Description
ECCIP1	5	rw	**Interrupt node XINTR11 enable** 0_B XINTR11 is disabled 1_B XINTR11 is enabled
ECCIP2	6	rw	**Interrupt node XINTR12 enable** 0_B XINTR12 is disabled 1_B XINTR12 is enabled
ECCIP3	7	rw	**Interrupt node XINTR13 enable** 0_B XINTR13 is disabled 1_B XINTR13 is enabled

39.4.115 Interrupt priority high register

IPH Offset address: $0D2_H$

Interrupt priority high register Kernel Reset value: 00_H

PAGE: X, RMAP: X

7	6	5	4	3	2	1	0
0		PT2H	PSH	PT1H	PX1H	PT0H	PX0H
r		rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
PX0H	0	rw	**Priority level high bit for interrupt node XINTR0**
PT0H	1	rw	**Priority level high bit for interrupt node XINTR1**
PX1H	2	rw	**Priority level high bit for interrupt node XINTR2**
PT1H	3	rw	**Priority level high bit for interrupt node XINTR3**
PSH	4	rw	**Priority level high bit for interrupt node XINTR4**
PT2H	5	rw	**Priority level high bit for interrupt node XINTR5**
0	7:6	r	**Reserved** Read as 0; must be written with 0.

39.4.116 Interrupt priority 1 high register

IPH1 Offset address: $0D3_H$

Interrupt priority 1 high register Kernel Reset value: 00_H

PAGE: X, RMAP: X

7	6	5	4	3	2	1	0
PCCIP3H	PCCIP2H	PCCIP1H	PCCIP0H	PXMH	PX2H	PSSCH	PWCANH
rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
PWCANH	0	rw	**Priority level high bit for interrupt node XINTR6**
PSSCH	1	rw	**Priority level high bit for interrupt node XINTR7**
PX2H	2	rw	**Priority level high bit for interrupt node XINTR8**
PXMH	3	rw	**Priority level high bit for interrupt node XINTR9**
PCCIP0H	4	rw	**Priority level high bit for interrupt node XINTR10**
PCCIP1H	5	rw	**Priority level high bit for interrupt node XINTR11**
PCCIP2H	6	rw	**Priority level high bit for interrupt node XINTR12**
PCCIP3H	7	rw	**Priority level high bit for interrupt node XINTR13**

39.4.117 Stack pointer

SP Offset address: $0D4_H$

Stack pointer Kernel Reset value: 07_H

PAGE: X, RMAP: X

7	6	5	4	3	2	1	0
SP							
rwh							

Field	Bits	Type	Description
SP	7:0	rwh	**Stack pointer value**

39.4.118 Data pointer low byte

DPL Offset address: $0D5_H$

Data pointer low byte Kernel Reset value: 00_H

PAGE: X, RMAP: X

7	6	5	4	3	2	1	0
DPL							
rwh							

Field	Bits	Type	Description
DPL	7:0	rwh	**Data pointer low byte value**

39.4.119 Data pointer high byte

DPH Offset address: $0D6_H$

Data pointer high byte Kernel Reset value: 00_H

PAGE: X, RMAP: X

7	6	5	4	3	2	1	0
DPH							
rwh							

Field	Bits	Type	Description
DPH	7:0	rwh	**Data pointer high byte value**

39.4.120 Extended operation register

EO Offset address: $0D7_H$

Extended operation register Kernel Reset value: 00_H

PAGE: X, RMAP: X

7	6	5	4	3	2	1	0
0			TRAP_EN	0	DPSEL		
r			rw	r	rw		

Field	Bits	Type	Description
DPSEL	2:0	rw	**Data pointer select** These bits are used to select the current data pointer. 000_B **DTPR0**: DPTR0 selected 001_B **DTPR1**: DPTR1 selected **others,** Reserved
TRAP_EN	4	rw	**TRAP enable** 0_B **SEL_MOVC**: Select MOVC @(DPTR++),A 1_B **SEL_TRAP**: Select software TRAP instruction
0	3, 7:5	r	**Reserved** Read as 0; must be written with 0.

39.4.121 Interrupt enable register 0

IEN0 Offset address: $0D8_H$

Interrupt enable register 0 Kernel Reset value: 00_H

PAGE: X, RMAP: X

7	6	5	4	3	2	1	0
EA	0	ET2	ES	ET1	EX1	ET0	EX0
rw	r	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
EX0	0	rw	**Interrupt node XINTR0 enable** 0_B XINTR0 is disabled 1_B XINTR0 is enabled
ET0	1	rw	**Interrupt node XINTR1 enable** 0_B XINTR1 is disabled 1_B XINTR1 is enabled
EX1	2	rw	**Interrupt node XINTR2 enable** 0_B XINTR2 is disabled 1_B XINTR2 is enabled
ET1	3	rw	**Interrupt node XINTR3 enable** 0_B XINTR3 is disabled 1_B XINTR3 is enabled
ES	4	rw	**Interrupt node XINTR4 enable** 0_B XINTR4 is disabled 1_B XINTR4 is enabled
ET2	5	rw	**Interrupt node XINTR5 enable** 0_B XINTR5 is disabled 1_B XINTR5 is enabled
EA	7	rw	**Global interrupt mask** 0_B All pending interrupt requests (except NMI) are blocked from the core 1_B Pending interrupt requests are not blocked from the core
0	6	r	**Reserved** Read as 0; must be written with 0.

39.4.122 Power control register

PCON Offset address: $0D9_H$

Power control register Kernel Reset value: 00_H

PAGE: X, RMAP: X

7	6	5	4	3	2	1	0
SMOD	**0**			**GF1**	**GF0**	**0**	**IDLE**
rw	r			rw	rw	r	rw

Field	Bits	Type	Description
IDLE	0	rw	**Idle mode enable** 0_B Do not enter idle mode 1_B Enter idle mode
GF0	2	rw	**General purpose flag bit 0**

(table continues...)

(continued)

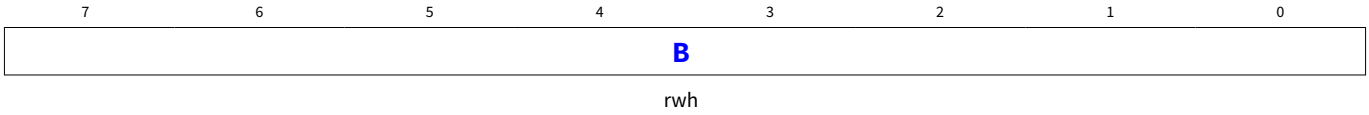
Field	Bits	Type	Description
GF1	3	rw	**General purpose flag bit 1**
SMOD	7	rw	**Double baud rate enable** 0_B Do not double the baud rate of serial interface in mode 2 1_B Double the baud rate of serial interface in mode 2
0	1, 6:4	r	**Reserved** Read as 0; must be written with 0.

39.4.123 B register

B Offset address: $0DA_H$

B register Kernel Reset value: 00_H

PAGE: X, RMAP: X



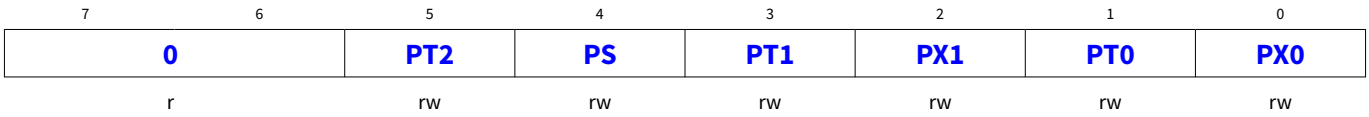
Field	Bits	Type	Description
B	7:0	rwh	**B register value**

39.4.124 Interrupt priority register

IP Offset address: $0DB_H$

Interrupt priority register Kernel Reset value: 00_H

PAGE: X, RMAP: X



Field	Bits	Type	Description
PX0	0	rw	**Priority level low bit for interrupt node XINTR0**
PT0	1	rw	**Priority level low bit for interrupt node XINTR1**
PX1	2	rw	**Priority level low bit for interrupt node XINTR2**
PT1	3	rw	**Priority level low bit for interrupt node XINTR3**
PS	4	rw	**Priority level low bit for interrupt node XINTR4**
PT2	5	rw	**Priority level low bit for interrupt node XINTR5**
0	7:6	r	**Reserved** Read as 0; must be written with 0.

39.4.125 Interrupt priority 1 register

IP1 Offset address: $0DC_H$

Interrupt priority 1 register Kernel Reset value: 00_H

PAGE: X, RMAP: X

7	6	5	4	3	2	1	0
PCCIP3	PCCIP2	PCCIP1	PCCIP0	PXM	PX2	PSSC	PWCAN
rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
PWCAN	0	rw	**Priority level low bit for interrupt node XINTR6**
PSSC	1	rw	**Priority level low bit for interrupt node XINTR7**
PX2	2	rw	**Priority level low bit for interrupt node XINTR8**
PXM	3	rw	**Priority level low bit for interrupt node XINTR9**
PCCIP0	4	rw	**Priority level low bit for interrupt node XINTR10**
PCCIP1	5	rw	**Priority level low bit for interrupt node XINTR11**
PCCIP2	6	rw	**Priority level low bit for interrupt node XINTR12**
PCCIP3	7	rw	**Priority level low bit for interrupt node XINTR13**

39.4.126 Accumulator

ACC Offset address: $0E0_H$

Accumulator Kernel Reset value: 00_H

PAGE: X, RMAP: X

7	6	5	4	3	2	1	0
ACC							
rwh							

Field	Bits	Type	Description
ACC	7:0	rwh	**Accumulator value**

39.4.127 Password register

PASSWD Offset address: $0DD_H$

Password register Kernel Reset value: 07_H

PAGE: X, RMAP: X

7	6	5	4	3	2	1	0
PASS					PROTECT_S	MODE	
w					rh	rwh	

Field	Bits	Type	Description
MODE	1:0	rwh	**Bit-protection scheme control bit** These two bits cannot be written directly. To change the value between 11_B and 00_B, the bitfield PASS must be written with 11000_B, only then the MODE[1:0] will be registered. 00_B **SCHEME_DIS**: Scheme disabled 11_B **SCHEME_EN3**: Scheme enabled **others,** Reserved; do not use
PROTECT_S	2	rh	**Bit-protection signal status bit** This bit shows the status of the protection. 0_B Software is able to write to all protected bits 1_B Software is unable to write to any protected bits (default)
PASS	7:3	w	**Password bits** The Bit-Protection Scheme only recognizes three patterns. 13_H **PROT_OPEN**: Opens access to writing of all protected bits 15_H **PROT_CLOSE**: Closes access to writing of all protected bits 18_H **MODE_EN**: Enables writing of the bit field MODE **others,** Reserved; do not use

39.4.128 On-chip XRAM address higher order

XADDRH Offset address: $0DE_H$

On-chip XRAM address higher order Kernel Reset value: 00_H

PAGE: X, RMAP: X

7	6	5	4	3	2	1	0
ADDRH							
rw							

Field	Bits	Type	Description
ADDRH	7:0	rw	**Higher order of on-chip XRAM address** The range for SCR is from 00_H.

39.4.129 System control register 0

SYSCON0 Offset address: $0DF_H$

System control register 0 Kernel Reset value: 04_H

PAGE: X, RMAP: X

7	6	5	4	3	2	1	0
0					AMSEL	0	RMAP
r					rw	r	rw

Field	Bits	Type	Description
RMAP	0	rw	**Special function register map control** 0_B Access to non-mapped (standard) special function register area 1_B Access to mapped special function register area (this setting is not used for applications; therefore, RMAP should be programmed with 0)
AMSEL	2	rw	**Active memory map select** 0_B Active memory map 0: Boot ROM is only mapped from 0xD000 1_B Active memory map 1: Last 512 bytes of boot ROM are mapped to both, 0x0000 and 0xD600
0	1, 7:3	r	**Reserved** Read as 0; must be written with 0.

39.4.130 Real-time clock control register

RTCSCR_CON Offset address: $0E1_H$

Real-time clock control register Reset values see: Table 1859

PAGE: X, RMAP: 0

7	6	5	4	3	2	1	0
CFRTC	**ECRTC**	**RTCCT**	**RTPBYP**	**RTCCLKSEL_P**	**RTCCLKSEL**		**RTCC**
rwh	rw	rwh	rw	w	rw		rw

Field	Bits	Type	Description
RTCC	0	rw	**Real-time clock start/stop control** This bit enables or disables the RTCSCR. After software writes to this bit, it takes 6 RTC clock cycles for the RTC to be actually enabled or disabled. 0_B Stop real-time clock operation 1_B Start real-time clock operation

(table continues...)

(continued)

Field	Bits	Type	Description
RTCCLKSEL	2:1	rw	**Real-time clock input clock selection** ***Note**: Real-time clock must be in stop operation before the clock setting can be changed.* ***Note**: Setting RTCCLKSEL to 1 (PCLK) shall be used with caution, as PCLK clock can change during RTC operation, based on settings in SCUSCR_CLK register.* *RTCCLKSEL shall not be set to 1 (PCLK) before transitioning into STANDBY1_70k mode, as 100 MHz HPOSC is disabled in this mode.* External clock input from pin SCR_P1.2 (P33.10). Writing to this bit-field is protected by bit RTCCLKSEL_P in this register. 00_B 70 kHz clock is selected 01_B PCLK clock is selected 10_B External clock oscillator input from pin SCR_P1.2 (P33.10) is selected 11_B External clock oscillator input from PMS (fOSC32k) is selected
RTCCLKSEL_P	3	w	**RTCCLKSEL protection bit** This is the protection bit for bit RTCCLKSEL. To write RTCCLKSEL, RTCCLKSEL_P needs to be set as well in the same access. 0_B RTCCLKSEL not writable 1_B RTCCLKSEL is writable
RTPBYP	4	rw	**Real-time clock 9-bit pre-scaler bypass** ***Note**: Real-time clock must be in stop operation before the clock setting can be changed.* 0_B 9-bit pre-scaler is not bypassed 1_B 9-bit pre-scaler is bypassed
RTCCT	5	rwh	**Real-time clock capture event trigger** When set, this bit is held until the capture occurs , and then cleared by hardware. Software has to poll this bit until it is cleared to ensure that the capture has occurred. 0_B No action 1_B Capture event is triggered
ECRTC	6	rw	**Real-time clock compare interrupt enable** 0_B Disable real-time clock compare interrupt 1_B Enable real-time clock compare interrupt
CFRTC	7	rwh	**Real-time clock compare flag** This bit is set by hardware when there is a compare match, and can only be cleared by software. A wake-up request is generated only if the system is in standby mode with SCR running. Writing 0 clears CFRTC. Writing 1 has no effect.

Table 1858 Access mode restrictions of RTCSCR_CON sorted by descending priority

Mode name	Access mode		Description
write 1 to .**RTCCLKSEL_P**	rw	RTCCLKSEL	
(default)	r	RTCCLKSEL	

Table 1859 Reset values of RTCSCR_CON

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.131 Count clock register n

4 count registers, RTCSCR_CNT0 - RTCSCR_CNT3, are used to represent the most significant 32 bits of the 41-bit real-time clock timer.

***Note**: When RTCSCR_CNTn is written, a minimum delay of 8 RTC clock cycles is required, until the next write access to the register is allowed. Otherwise, the second write access will be lost.*

RTCSCR_CNTn (n=0-3) Offset address: $0E2_H$+n

Count clock register n Reset values see: Table 1861

PAGE: X, RMAP: 0

7	6	5	4	3	2	1	0
CNT_VAL							
rwh							

Field	Bits	Type	Description
CNT_VAL	7:0	rwh	**Real-time clock count value** These bits represent the current 32-bit real-time clock value. The assignment is as follows: • RTCSCR_CNT3: Bits [31:24] • RTCSCR_CNT2: Bits [23:16] • RTCSCR_CNT1: Bits [15:8] • RTCSCR_CNT0: Bits [7:0]

Table 1860 Access mode restrictions of RTCSCR_CNTn (n=0-3) sorted by descending priority

Mode name	Access mode		Description
PASSWD.PROTECT_S=0	rwh	CNT_VAL	
(default)	rh	CNT_VAL	

Table 1861 Reset values of RTCSCR_CNTn (n=0-3)

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.132 Real-time clock compare/capture register n

4 compare and capture registers, RTCSCR_CR0 - RTCSCR_CR3, are used to represent 32 bits of compare value, which will generate a compare event when it matches the counter value as specified in the RTCSCR_CNT register.

When a capture event is triggered by setting RTCCT bit to 1, the content in the RTCSCR_CRn register will be overwritten with the captured RTCSCR_CNT value after a maximum of 8 f_{RTC} clock cycles. In order to take effect of the new capture value inside RTCSCR_CRn, another delay of 9 RTC clock cycles is required. During this time, the compare interrupt is disabled.

If the pre-scaler is bypassed, then the captured value inside the register is 2 or 3 clock cycles higher in relation to the point in time of the trigger set in RTCCT. Otherwise, the value is exact or 1 clock cycle higher than the point in time of the trigger.

***Note**: When RTCSCR_CRn is written, a minimum delay of 8 RTC clock cycles is required, until the next write access to the register is allowed. Otherwise, the second write access is not guaranteed.*

***Note**: Before starting the RTC counter, the compare value in registers RTCSCR_CRn should be set to a value different to 0 (e.g. 0xFFFF). Otherwise, the RTC will not start if it is initially 0, or stop after an overflow to 0.*

RTCSCR_CRn (n=0-3) Offset address: $0E6_H$+n

Real-time clock compare/capture register n Reset values see: Table 1862

PAGE: X, RMAP: 0

rwh

Field	Bits	Type	Description
CC_VAL	7:0	rwh	**Compare/Capture value** These bits represent the 32-bit compare/capture value. The assignment is as follows: • RTCSCR_CR3: Bits [31:24] • RTCSCR_CR2: Bits [23:16] • RTCSCR_CR1: Bits [15:8] • RTCSCR_CR0: Bits [7:0]

Table 1862 Reset values of RTCSCR_CRn (n=0-3)

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.133 Watchdog timer control register

WDTSCR_CON Offset address: $0EA_H$

Watchdog timer control register Reset values see: Table 1864

PAGE: X, RMAP: 0

7	6	5	4	3	2	1	0
0		WINBEN	WDTPR	WDTCLK	WDTEN	WDTRS	WDTIN
r		rw	rh	rw	rw	rwh	rw

Field	Bits	Type	Description
WDTIN	0	rw	**Watchdog timer input frequency selection** 0_B Input frequency is $f_{WDTCLK}/2$ 1_B Input frequency is $f_{WDTCLK}/128$
WDTRS	1	rwh	**WDT refresh start** Active high. Set to start refresh operation on the watchdog timer. Cleared automatically by hardware after it is set by software. The refresh operation is possible at any operation mode of the watchdog timer, as long as the request is evaluated before the watchdog generated NMI takes place.
WDTEN	2	rw	**WDT enable** WDTEN is a protected bit. If the protection scheme is activated, then this bit cannot be written directly. See section "Miscellaneous control" for details on the protection scheme. ***Note**: Clearing WDTEN bit to 0 during the pre-warning mode (WDTPR = 1) has no effect.* 0_B WDT is disabled 1_B WDT is enabled
WDTCLK	3	rw	**Watchdog timer clock source selection** 0_B 70 kHz clock is selected 1_B PCLK (100 MHz/DIV) clock is selected
WDTPR	4	rh	**Watchdog pre-warning mode flag** This bit is set to 1 when a watchdog error is detected. The watchdog timer has issued an NMI trap and is in pre-warning mode. A reset of the chip occurs, after the pre-warning period has expired. 0_B Normal mode (default after reset) 1_B The watchdog is operating in pre-warning mode
WINBEN	5	rw	**Watchdog window-boundary enable** 0_B Watchdog window-boundary feature is disabled (default) 1_B Watchdog window-boundary feature is enabled
0	7:6	r	**Reserved** Read as 0; must be written with 0.

Table 1863 Access mode restrictions of WDTSCR_CON sorted by descending priority

Mode name	Access mode		Description
PASSWD.PROTECT_S=0	rw	WDTEN	
(default)	r	WDTEN	

Table 1864 Reset values of WDTSCR_CON

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.134 Watchdog timer reload register

WDTSCR_REL Offset address: $0EB_H$

Watchdog timer reload register Reset values see: Table 1865

PAGE: X, RMAP: 0

Field	Bits	Type	Description
WDTREL	7:0	rw	**Watchdog timer reload value** Reload value for the high byte of the WDTSCR. The delay has to be regarded for WDTSCR enable and WDTSCR refresh operation, as otherwise the previous WDTREL value is still active.

Table 1865 Reset values of WDTSCR_REL

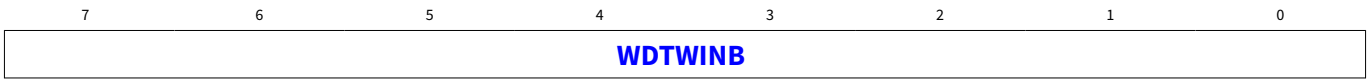
Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.135 Watchdog window-boundary register

WDTSCR_WINB Offset address: $0EC_H$

Watchdog window-boundary register Reset values see: Table 1866

PAGE: X, RMAP: 0



Field	Bits	Type	Description
WDTWINB	7:0	rw	**Watchdog window-boundary value** This value is programmable. Within this window-boundary range from 0000_H to (WDTSCR_WINB|00_H), the WDTSCR cannot do a refresh, else it will cause a WDTRST to be asserted. WDTSCR_WINB is compared to WDTSCR_H.

Table 1866 Reset values of WDTSCR_WINB

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.136 Watchdog timer low byte

WDTSCR_L Offset address: $0ED_H$

Watchdog timer low byte Reset values see: Table 1867

PAGE: X, RMAP: 0

7	6	5	4	3	2	1	0
WDT							
rh							

Field	Bits	Type	Description
WDT	7:0	rh	**Watchdog timer current value, low byte**

Table 1867 Reset values of WDTSCR_L

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.137 Watchdog timer high byte

WDTSCR_H Offset address: $0EE_H$

Watchdog timer high byte Reset values see: Table 1868

PAGE: X, RMAP: 0

7	6	5	4	3	2	1	0
WDT							
rh							

Field	Bits	Type	Description
WDT	7:0	rh	**Watchdog timer current value, high byte**

Table 1868 Reset values of WDTSCR_H

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.138 Page register for SCUSCR

SCUSCR_PAGE

Page register for SCUSCR

PAGE: X, RMAP: 0

Offset address: $0EF_H$

Reset values see: Table 1869

7	6	5	4	3	2	1	0
OP		**STNR**		**0**	**PAGE**		
w		w		r	rwh		

Field	Bits	Type	Description
PAGE	2:0	rwh	**Page bits** When written, the value indicates the new page address. When read, the value indicates the currently active page = addr[y:x+1]. 000_B **PAGE0**: Selection is PAGE0 001_B **PAGE1**: Selection is PAGE1 010_B **PAGE2**: Selection is PAGE2 011_B **PAGE3**: Selection is PAGE3 100_B **PAGE4**: Selection is PAGE4 101_B **PAGE5**: Selection is PAGE5 110_B **PAGE6**: Selection is PAGE6 111_B **PAGE7**: Selection is PAGE7
STNR	5:4	w	**Storage number** This number indicates which storage bitfield is the target of the operation defined by bit OP. If OP = 10_B, the contents of PAGE are saved in STx before being overwritten with the new value. If OP = 11_B, the contents of PAGE are overwritten by the contents of STx. The value written to bitfield PAGE is ignored. 00_B **MOD_ST0**: ST0 is selected 01_B **MOD_ST1**: ST1 is selected 10_B **MOD_ST2**: ST2 is selected 11_B **MOD_ST3**: ST3 is selected
OP	7:6	w	**Operation** 00_B **PAGE_MANUAL0**: Manual page mode. The value of STNR is ignored and PAGE is directly written. 01_B **PAGE_MANUAL1**: Manual page mode. The value of STNR is ignored and PAGE is directly written 10_B **PAGE_SAVE**: New page programming with automatic page saving. The value written to the bit positions of PAGE is stored. In parallel, the former contents of PAGE are saved in the storage bit field STx indicated by STNR. 11_B **PAGE_RESTORE**: Automatic restore page action. The value written to the bit positions PAGE is ignored and instead, PAGE is overwritten by the contents of the storage bit field STx indicated by STNR.
0	3	r	**Reserved** Read as 0; must be written with 0.

Table 1869 Reset values of SCUSCR_PAGE

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.139 Reset control register

SCUSCR_RSTCON Offset address: $0F0_H$

Reset control register Reset values see: Table 1871

PAGE: SCUSCR_PAGE=0, RMAP: 0

7	6	5	4	3	2	1	0
0			SWRSTREQ	0	PINRSTEN	WDTRSTEN	RAMECCRSTEN
r			rwh	r	rw	rw	rw

Field	Bits	Type	Description
RAMECCRSTEN	0	rw	**Double bit ECC reset enable bit** 0_B ECC reset is disabled 1_B ECC reset is enabled
WDTRSTEN	1	rw	**Watchdog reset enable bit** 0_B Watchdog reset is disabled 1_B Watchdog reset is enabled
PINRSTEN	2	rw	**Pin reset enable bit** If enabled PINRST will trigger main reset of SCR only after 64 consecutive PCLK clocks cycles of low value on selected input. Input selection is done through SCUSCR_MODPISEL1.PINRSTIS bit-field. 0_B Reset from dedicated pin is disabled 1_B Reset from dedicated pin is enabled
SWRSTREQ	4	rwh	**Software reset request** This bit is automatically cleared by hardware. The SWRSTREQ bit is a protected bit. When the protection scheme is activated, this bit cannot be written directly. For more information on protection scheme, see Section "Miscellaneous Control". 0_B No action 1_B On set, soft reset is requested
0	3, 7:5	r	**Reserved** Read as 0; must be written with 0.

Table 1870 Access mode restrictions of SCUSCR_RSTCON sorted by descending priority

Mode name	Access mode		Description
PASSWD.PROTECT_S=0	rwh	SWRSTREQ	
(default)	rh	SWRSTREQ	

Table 1871 Reset values of SCUSCR_RSTCON

Reset type	Reset value	Note
LVD Reset	03_H	
Generated Reset	02_H	

39.4.140 SCR reset status register

SCUSCR_RSTSTAT Offset address: $0F1_H$

SCR reset status register Reset values see: Table 1872

PAGE: SCUSCR_PAGE=0, RMAP: 0

7	6	5	4	3	2	1	0
0			SWRST	0	PINRST	WDTRST	RAMECCRST
r			rwh	r	rwh	rwh	rwh

Field	Bits	Type	Description
RAMECCRST	0	rwh	**Double bit ECC reset indication bit** This bit can only be set by hardware and cleared by software. 0_B No ECC reset has occurred 1_B ECC reset has occurred
WDTRST	1	rwh	**Watchdog timer reset indication bit** This bit can only be set by hardware and cleared by software. 0_B No watchdog reset has occurred 1_B Watchdog reset has occurred
PINRST	2	rwh	**Pin reset indication bit** 0_B No reset from pin has occurred 1_B Reset from pin has occurred
SWRST	4	rwh	**Software reset indication bit** 0_B No software reset has occurred 1_B Software reset has occurred
0	3, 7:5	r	**Reserved** Read as 0; must be written with 0.

Table 1872 Reset values of SCUSCR_RSTSTAT

Reset type	Reset value	Note
LVD Reset	00_H	
Main Reset	00_H	

39.4.141 Main reset status register

SCUSCR_MRSTSTAT Offset address: $0F2_H$

Main reset status register LVD Reset value: 00_H

PAGE: SCUSCR_PAGE=0, RMAP: 0

7	6	5	4	3	2	1	0
0						APPRST	SMURST
r						rwh	rwh

Field	Bits	Type	Description
SMURST	0	rwh	**SMU reset status bit** This bit can only be set by hardware, and cleared by software. 0_B No reset issued by SMU since the last clearing of the bit 1_B Reset issued by SMU
APPRST	1	rwh	**Application reset status bit** This bit can only be set by hardware, and cleared by software. 0_B Application reset is not triggered since the last clearing of the bit 1_B Application reset triggered
0	7:2	r	**Reserved** Read as 0; must be written with 0.

39.4.142 SCUSCR status register

SCUSCR_STBY Offset address: $0F3_H$

SCUSCR status register Reset values see: Table 1874

PAGE: SCUSCR_PAGE=0, RMAP: 0

7	6	5	4	3	2	1	0
0		STBY0REQ	PMSWKPCLR	PMS0WKP	PMSRTCWKP	STBY0WKP	STBYSTAT
r		rw	w	rh	rh	rh	rh

Field	Bits	Type	Description
STBYSTAT	0	rh	**System standby mode status** Wired in value from PMS. This bit-field is used to check if main system is in RUN or STANDBY1 state. 0_B System is in RUN mode 1_B System is in STANDBY1 mode

(table continues...)

(continued)

Field	Bits	Type	Description
STBY0WKP	1	rh	**Standby0 wake-up flag** Wired in value from PMS_CLDRSTSTAT.STBY0WKP flag. This bit-field is used to check if main system was in STANDBY0 state. 0_B Transition from STANDBY0 state has not occurred since the last clear (by PMS_CLDRSTSTATCLR.CLEAR). 1_B Transition from STANDBY0 state has occurred since the last clear (by PMS_CLDRSTSTATCLR.CLEAR).
PMSRTCWKP	2	rh	**PMS RTC wake-up flag** Wired in value from PMS_WAKEUP_STAT0.RTCWKP flag. This bit-field is used to check if source of latest system wakeup was PMS RTC, so that SCR software can run different routine in that case. 0_B PMS RTC has not triggered a wake-up since the last clear of PMS_WAKEUP_STAT0.RTCWKP 1_B PMS RTC has triggered a wake-up since the last clear of PMS_WAKEUP_STAT0.RTCWKP
PMS0WKP	3	rh	**PMS0 combined wake-up flags** Wired in OR-ed value from all PMS_WAKEUP_STAT0 WKP flags. 0_B None of the sources in PMS_WAKEUP_STAT0 has triggered a wake-up since the last clear of PMS_WAKEUP_STAT0 flags 1_B At least one of the sources in PMS_WAKEUP_STAT0 has triggered a wake-up since the last clear of PMS_WAKEUP_STAT0 flags
PMSWKPCLR	4	w	**Clear wake-up flags in PMS** To allow system transition into STANDBY0 state all wake-up flags on PMS side must be cleared first. Clearing will take part only if enabled inside PMS with PMS_WAKEUP_CON0. Reading this bit-field returns '0'. 0_B No action. 1_B Clear PMS wakeup flags. All PMS wakeup bits will be cleared.
STBY0REQ	5	rw	**Request transition from STANDBY1 to STANDBY0 system state (can only be set during STANDBY1)** Transition will only take part if: • System is in STANDBY1 state. • STANDBY1 to STANDBY0 transition is enabled inside PMS via PMS_STANDBY_CON0.SCRSTBY0EN='1'. • All PMS wake-up flags are cleared, by writing SCUSCR_STBY.PMSWKPCLR='1'. If successful, SCR is switched off and the request and also the bit field is implicitly cleared. Also during a transition from STANDBY1 to RUN mode the bit field STBY0REQ is cleared. 0_B No request 1_B Request transition from STANDBY1 to STANDBY0
0	7:6	r	**Reserved** Read as 0; must be written with 0.

Table 1873 Access mode restrictions of SCUSCR_STBY sorted by descending priority

Mode name	Access mode		Description
PASSWD.PROTECT_S=0	rw	STBY0REQ	
	w	PMSWKPCLR	
(default)	r	PMSWKPCLR, STBY0REQ	

Table 1874 Reset values of SCUSCR_STBY

Reset type	Reset value	Note
LVD Reset	$0X_H$	
Generated Reset	$0-_H$	Wired in signals from PMS are not affected by SCR kernel reset.

39.4.143 Clock control and status register

SCUSCR_CLK Offset address: $0F4_H$

Clock control and status register Reset values see: Table 1876

PAGE: SCUSCR_PAGE=0, RMAP: 0

7	6	5	4	3	2	1	0
DIV_STAT				DIV_REQ			
rh				rwh			

Field	Bits	Type	Description
DIV_REQ	3:0	rwh	**Clock divider request** Effect on enabling and disabling of HPOSC 100 MHz oscillator from this bit-field is only considered when the main system is in standby mode. In all other modes SCR has no control over HPOSC. Setting this bit-field triggers either an immediate or a stepwise clock update: • If both DIV_STAT and the new DIV_REQ value represent frequency <= 10 MHz (DIV10 or higher DIV values) the transition to the new value, immediately. • If at least one of the DIV_REQ or DIV_STAT values represent frequency >10 MHz (DIV is one of DIV1-5,DIV8) the transition is stepwise. In case of stepwise transition, both directions (lower frequency to higher, and higher to lower) are affected. Steps are performed one DIV increment/decrement at a time between DIV1 and DIV10, and a direct transition between DIV10 and DIV70kHz. Each steps takes ~15.3 us. If f_start=70 kHz and HPOSC was not enabled prior to setting DIV_REQ to a new value, it takes additional ~200 us to start the HPOSC oscillator. This bit-field is automatically set to its default value (DIV5, 20 MHz) upon reset and upon detection of WCAN wake-up signal. 0_H **DIV1**: DIV = 1; f_{SYS} = 100 MHz 1_H **DIV2**: DIV = 2; f_{SYS} = 50 MHz 2_H **DIV3**: DIV = 3; f_{SYS} = 33.3 MHz 3_H **DIV4**: DIV = 4; f_{SYS} = 25 MHz 4_H **DIV5**: DIV = 5; f_{SYS} = 20 MHz (default) 5_H **DIV8**: DIV = 8; f_{SYS} = 12.5 MHz 6_H **DIV10**: DIV = 10; f_{SYS} = 10 MHz 7_H **DIV16**: DIV = 16; f_{SYS} = 6.25 MHz 8_H **DIV20**: DIV = 20; f_{SYS} = 5 MHz 9_H **DIV32**: DIV = 32; f_{SYS} = 3.125 MHz A_H **DIV50**: DIV = 50; f_{SYS} = 2 MHz B_H **DIV80**: DIV = 80; f_{SYS} = 1.25 MHz C_H **DIV100**: DIV = 100; f_{SYS} = 1 MHz D_H **DIV160**: DIV = 160;

(table continues...)

(continued)

Field	Bits	Type	Description
			f_{SYS} = 0.625 MHz E_H **DIV200**: DIV = 200; f_{SYS} = 0.5 MHz F_H **DIV70kHz**: 70 kHz f_{SYS} = 70 kHz
DIV_STAT	7:4	rh	**Clock divider status** Represents current clock frequency. If DIV_STAT is different than DIV_REQ it means that a stepwise frequency update is on-going. 0_H **DIV1**: DIV = 1; f_{SYS} = 100 MHz 1_H **DIV2**: DIV = 2; f_{SYS} = 50 MHz 2_H **DIV3**: DIV = 3; f_{SYS} = 33.3 MHz 3_H **DIV4**: DIV = 4; f_{SYS} = 25 MHz 4_H **DIV5**: DIV = 5; f_{SYS} = 20 MHz 5_H **DIV8**: DIV = 8; f_{SYS} = 12.5 MHz 6_H **DIV10**: DIV = 10; f_{SYS} = 10 MHz 7_H **DIV16**: DIV = 16; f_{SYS} = 6.25 MHz 8_H **DIV20**: DIV = 20; f_{SYS} = 5 MHz 9_H **DIV32**: DIV = 32; f_{SYS} = 3.125 MHz A_H **DIV50**: DIV = 50; f_{SYS} = 2 MHz B_H **DIV80**: DIV = 80; f_{SYS} = 1.25 MHz C_H **DIV100**: DIV = 100; f_{SYS} = 1 MHz D_H **DIV160**: DIV = 160; f_{SYS} = 0.625 MHz E_H **DIV200**: DIV = 200; f_{SYS} = 0.5 MHz F_H **DIV70kHz**: 70 kHz f_{SYS} = 70 kHz

Table 1875 Access mode restrictions of SCUSCR_CLK sorted by descending priority

Mode name	Access mode		Description
PASSWD.PROTECT_S=0	rwh	DIV_REQ	
(default)	rh	DIV_REQ	

Table 1876 Reset values of SCUSCR_CLK

Reset type	Reset value	Note
LVD Reset	$F4_H$	
Generated Reset	-4_H	DIV_STAT is reset to 0xF, but since DIV_REQ reset value is 0x4 the clock update starts after reset and exact DIV_STAT value depends when it is read. Therefore the reset value can be seen as undefined for the application.

39.4.144 Control register for system wake-up via SCR sources

SCUSCR_WKPCON Offset address: $0F5_H$

Control register for system wake-up via SCR sources Reset values see: Table 1877

PAGE: SCUSCR_PAGE=0, RMAP: 0

7	6	5	4	3	2	1	0
0			SWWKPREQ	RAMECCWKPSEL	WDTWKPSEL	WCANINTWKPSEL	RTCWKPSEL
r			rwh	rw	rw	rw	rw

Field	Bits	Type	Description
RTCWKPSEL	0	rw	**RTC wake-up select** 0_B Wake-up from standby mode via RTC is not selected 1_B Wake-up from standby mode via RTC is selected
WCANINTWKPSEL	1	rw	**WCAN interrupt wake-up select** 0_B Wake-up from standby mode via WCAN Interrupt is not selected 1_B Wake-up from standby mode via WCAN Interrupt is selected
WDTWKPSEL	2	rw	**WDT wake-up select** 0_B Wake-up from standby mode is not selected when WDT event occurred 1_B Wake-up from standby mode is selected when WDT event occurred
RAMECCWKPSEL	3	rw	**RAM ECC wake-up select** 0_B Wake-up from standby mode is not selected when RAM ECC event occurred 1_B Wake-up from standby mode is selected when RAM ECC event occurred
SWWKPREQ	4	rwh	**Software wake-up request** This bit is automatically cleared by hardware. 0_B No action 1_B Request a wake-up from standby mode
0	7:5	r	**Reserved** Read as 0; must be written with 0.

Table 1877 Reset values of SCUSCR_WKPCON

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.145 SCR interrupt data exchange register

SCUSCR_SCRINTEXCHG Offset address: $0F0_H$

SCR interrupt data exchange register LVD Reset value: 00_H

PAGE: SCUSCR_PAGE=1, RMAP: 0

7	6	5	4	3	2	1	0
SCRINT							
rw							

Field	Bits	Type	Description
SCRINT	7:0	rw	**Data exchange from SCR to main controller** Data exchange between SCR and main controller. The data can be read via PMS_SCR_STAT.SCRINT.

39.4.146 TriCore interrupt data exchange register

SCUSCR_TCINTEXCHG Offset address: $0F1_H$

TriCore interrupt data exchange register LVD Reset value: 00_H

PAGE: SCUSCR_PAGE=1, RMAP: 0

7	6	5	4	3	2	1	0
TCINT							
rh							

Field	Bits	Type	Description
TCINT	7:0	rh	**Data exchange from main controller to SCR** Data exchange between main controller and SCR. The data written to PMS_SCR_CON1.TCINT appears as read-only here.

39.4.147 Peripheral management control register 0

SCUSCR_PMCON0 Offset address: $0F3_H$

Peripheral management control register 0 Reset values see: Table 1878

PAGE: SCUSCR_PAGE=1, RMAP: 0

7	6	5	4	3	2	1	0
OCDS_DIS	T2CCU1_DIS	T2CCU0_DIS	LIN_DIS	WDT_DIS	WCAN_DIS	RTC_DIS	SSC_DIS
rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
SSC_DIS	0	rw	**SSC disable request (active high)** 0_B SSC is in normal operation 1_B Request to disable the SSC (default)
RTC_DIS	1	rw	**RTC disable request (active high)** 0_B RTC is in normal operation 1_B Request to disable the RTC (default)
WCAN_DIS	2	rw	**WCAN disable request (active high)** 0_B WCAN is in normal operation 1_B Request to disable the WCAN (default)
WDT_DIS	3	rw	**Watchdog disable request (active high)** 0_B Watchdog is in normal operation 1_B Request to disable the watchdog (default)
LIN_DIS	4	rw	**LIN disable request (active high)** 0_B LIN baudrate generator and detector are in normal operation 1_B Request to disable the LIN baudrate generator and detector (default)
T2CCU0_DIS	5	rw	**T2CCU0 disable request (active high)** 0_B T2CCU0 is in normal operation 1_B Request to disable the T2CCU0 (default)
T2CCU1_DIS	6	rw	**T2CCU1 disable request (active high)** 0_B T2CCU1 is in normal operation 1_B Request to disable the T2CCU1 (default)
OCDS_DIS	7	rw	**OCDS disable request (active high)** 0_B OCDS and debug system is in normal operation (this is the default value, and the clock to OCDS and debug system is enabled by default) 1_B Request to disable the OCDS and debug system (during normal operation, application may choose to gate the OCDS and debug system clock to save power)

Table 1878 Reset values of SCUSCR_PMCON0

Reset type	Reset value	Note
LVD Reset	$7F_H$	
Generated Reset	$7F_H$	

39.4.148 Peripheral management control register 1

SCUSCR_PMCON1 Offset address: $0F4_H$

Peripheral management control register 1 Reset values see: Table 1879

PAGE: SCUSCR_PAGE=1, RMAP: 0

7	6	5	4	3	2	1	0
0							I2C_DIS
r							rw

Field	Bits	Type	Description
I2C_DIS	0	rw	**I2C disable request (active high)** 0_B I2C is in normal operation 1_B Request to disable the I2C (default)
0	7:1	r	**Reserved** Read as 0; must be written with 0.

Table 1879 Reset values of SCUSCR_PMCON1

Reset type	Reset value	Note
LVD Reset	01_H	
Generated Reset	01_H	

39.4.149 Peripheral input select register 0

This register is used for input pin selection of the UART and timer 0/1 modules. The register is on SCUSCR_PAGE.

SCUSCR_MODPISEL0 Offset address: $0F0_H$

Peripheral input select register 0 Reset values see: Table 1880

PAGE: SCUSCR_PAGE=2, RMAP: 0

7	6	5	4	3	2	1	0
T1IS		T0IS		URRIS			
rw		rw		rw			

Field	Bits	Type	Description
URRIS	3:0	rw	**UART receiver input select** 0_H **UART_RXDA**: UART receiver input RXDA is selected 1_H **UART_RXDB**: UART receiver input RXDB is selected 2_H **UART_RXDC**: UART receiver input RXDC is selected 3_H **UART_RXDD**: UART receiver input RXDD is selected 4_H **UART_RXDE**: UART receiver input RXDE is selected 5_H **UART_RXDF**: UART receiver input RXDF is selected 6_H **UART_RXDG**: UART receiver input RXDG is selected 7_H **UART_RXDH**: UART receiver input RXDH is selected 8_H **UART_RXDI**: UART receiver input RXDI is selected 9_H **UART_RXDJ**: UART receiver input RXDJ is selected A_H **UART_RXDK**: UART receiver input RXDK is selected B_H **UART_RXDL**: UART receiver input RXDL is selected C_H **UART_RXDM**: UART receiver input RXDM is selected D_H **UART_RXDN**: UART receiver input RXDN is selected E_H **UART_RXDO**: UART receiver input RXDO is selected F_H **UART_RXDP**: UART receiver input RXDP is selected
T0IS	5:4	rw	**Timer 0 input select** 00_B **T0A**: Timer 0 input T0A is selected 01_B **T0B**: Timer 0 input T0B is selected 10_B **T0C**: Timer 0 input T0C is selected 11_B **T0D**: Timer 0 input T0D is selected
T1IS	7:6	rw	**Timer 1 input select** 00_B **T1A**: Timer 1 input T1A is selected 01_B **T1B**: Timer 1 input T1B is selected 10_B **T1C**: Timer 1 input T1C is selected 11_B **T1D**: Timer 1 input T1D is selected

Table 1880 Reset values of SCUSCR_MODPISEL0

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.150 Peripheral input select register 1

This register is used for input pin selection of the reset-from-pin functionality and for SSC module. The register is on SCUSCR_PAGE.

SCUSCR_MODPISEL1 Offset address: $0F1_H$

Peripheral input select register 1 Reset values see: Table 1881

PAGE: SCUSCR_PAGE=2, RMAP: 0

7	6	5	4	3	2	1	0
CIS		SIS		MIS		PINRSTIS	
rw		rw		rw		rw	

Field	Bits	Type	Description
PINRSTIS	1:0	rw	**Reset pin input select** 00_B **PINRSTA**: Reset from pin PINRSTA is selected 01_B **PINRSTB**: Reset from pin PINRSTB is selected 10_B **PINRSTC**: Reset from pin PINRSTC is selected 11_B **PINRSTD**: Reset from pin PINRSTD is selected
MIS	3:2	rw	**Master mode receive input select** 00_B **MRSTA**: SSC master receiver input MRSTA is selected 01_B **MRSTB**: SSC master receiver input MRSTB is selected 10_B **MRSTC**: SSC master receiver input MRSTC is selected 11_B **MRSTD**: SSC master receiver input MRSTD is selected
SIS	5:4	rw	**Slave mode receive input select** 00_B **MTSRA**: SSC slave receiver input MTSRA is selected 01_B **MTSRB**: SSC slave receiver input MTSRB is selected 10_B **MTSRC**: SSC slave receiver input MTSRC is selected 11_B **MTSRD**: SSC slave receiver input MTSRD is selected
CIS	7:6	rw	**Slave mode clock input select** 00_B **SCLKA**: SSC slave clock input SCLKA is selected 01_B **SCLKB**: SSC slave clock input SCLKB is selected 10_B **SCLKC**: SSC slave clock input SCLKC is selected 11_B **SCLKD**: SSC slave clock input SCLKD is selected

Table 1881 Reset values of SCUSCR_MODPISEL1

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

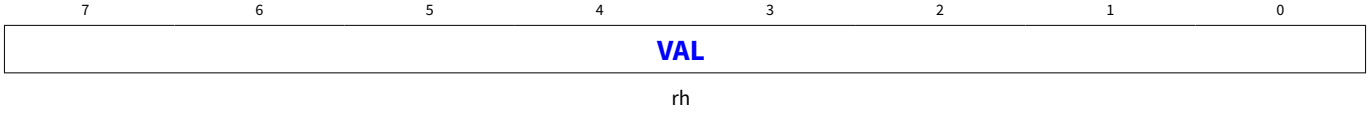
39.4.151 SCR access to PMS_RTC_TIM[39:32] bits

Reading this register returns value from respective PMS_RTC bit-field.

SCUSCR_PMSRTC_TIM39_32 Offset address: $0F2_H$

SCR access to PMS_RTC_TIM[39:32] bits LVD Reset value: 00_H

PAGE: SCUSCR_PAGE=2, RMAP: 0



Field	Bits	Type	Description
VAL	7:0	rh	**TIM[39:32] value**

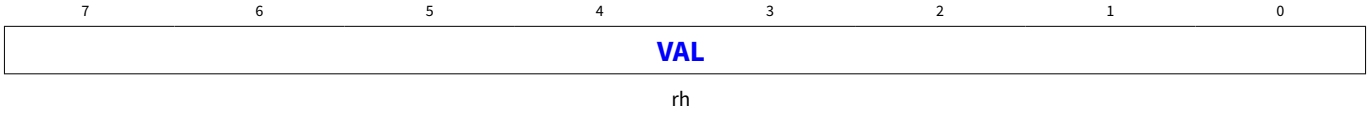
39.4.152 SCR access to PMS_RTC_TIM[47:40] bits

Reading this register returns value from respective PMS_RTC bit-field.

SCUSCR_PMSRTC_TIM47_40 Offset address: $0F3_H$

SCR access to PMS_RTC_TIM[47:40] bits LVD Reset value: 00_H

PAGE: SCUSCR_PAGE=2, RMAP: 0



Field	Bits	Type	Description
VAL	7:0	rh	**TIM[47:40] value**

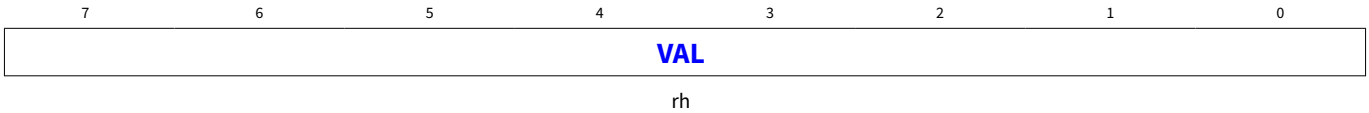
39.4.153 SCR access to PMS_RTC_TIM[55:48] bits

Reading this register returns value from respective PMS_RTC bit-field.

SCUSCR_PMSRTC_TIM55_48 Offset address: $0F4_H$

SCR access to PMS_RTC_TIM[55:48] bits LVD Reset value: 00_H

PAGE: SCUSCR_PAGE=2, RMAP: 0



Field	Bits	Type	Description
VAL	7:0	rh	**TIM[55:48] value**

39.4.154 SCR access to PMS_RTC_TIM[63:56] bits

Reading this register returns value from respective PMS_RTC bit-field if system is not in STANDBY1_70K mode.

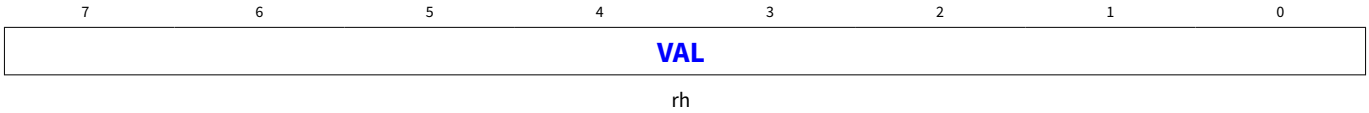
Since the TIM counter is running while reading 8-bits at a time, it is recommended to read TIM[63:56] first, and then proceed with TIM[55:48] etc., and last read TIM[7:0].

In case TIM[7:0]==0x00, perform re-read of other 7 registers to be sure the counter didn't tip over during reading.

SCUSCR_PMSRTC_TIM63_56 Offset address: $0F5_H$

SCR access to PMS_RTC_TIM[63:56] bits LVD Reset value: 00_H

PAGE: SCUSCR_PAGE=2, RMAP: 0



Field	Bits	Type	Description
VAL	7:0	rh	**TIM[63:56] value**

39.4.155 SCR access to PMS_RTC_CON1.OFFSET[7:0] bits and update trigger for complete PMS_RTC_CON1.OFFSET

Writing to this register updates PMS_RTC_CON1.OFFSET[15:0] bit-field with combined content of PMSRTC_OFFSET15_8 and this register if system is not in STANDBY1_70K mode. It is important to update the content of PMSRTC_OFFSET15_8 along with this register. Writing only to this register will assume the value of PMSRTC_OFFSET15_8 to be 0x00. Therefore, to ensure that all the bits of PMS_RTC_CON1.OFFSET[15:0] are correctly written, it is important to update both this register and PMSRTC_OFFSET15_8.

Reading this register returns value from respective PMS_RTC bit-field.

SCUSCR_PMSRTC_OFFSET7_0 Offset address: $0F0_H$

SCR access to PMS_RTC_CON1.OFFSET[7:0] bits and update trigger for complete PMS_RTC_CON1.OFFSET

LVD Reset value: 00_H

PAGE: SCUSCR_PAGE=3, RMAP: 0

7	6	5	4	3	2	1	0
VAL							
rwh							

Field	Bits	Type	Description
VAL	7:0	rwh	**OFFSET[7:0] value**

39.4.156 SCR shadow register for PMS_RTC_CON1.OFFSET[15:8] bits

Writing to this register updates local shadow register PMSRTC_OFFSET15_8. Update of PMSRTC_CON1.OFFSET bit-field is triggered with write to PMSRTC_OFFSET7_0.

Reading this register returns value from PMS_RTC_CON1.OFFSET[15:8] bit-field. Therefore, it is not possible to read back value last written to local shadow register.

SCUSCR_PMSRTC_OFFSET15_8 Offset address: $0F1_H$

SCR shadow register for PMS_RTC_CON1.OFFSET[15:8] bits

LVD Reset value: 00_H

PAGE: SCUSCR_PAGE=3, RMAP: 0

7	6	5	4	3	2	1	0
VAL							
rwh							

Field	Bits	Type	Description
VAL	7:0	rwh	**OFFSET[15:8] value**

39.4.157 SCR access to PMS_RTC_TIM[7:0] bits

Reading this register returns value from respective PMS_RTC bit-field if system is not in STANDBY1_70K mode.

Since the TIM counter is running while reading 8-bits at a time, it is recommended to read TIM[63:56] first, and then proceed with TIM[55:48], and last TIM[7:0].

In case TIM[7:0]==0x00, perform re-read of all 8 registers to be sure the counter didn't tip over during reading.

SCUSCR_PMSRTC_TIM7_0 Offset address: $0F2_H$

SCR access to PMS_RTC_TIM[7:0] bits LVD Reset value: 00_H

PAGE: SCUSCR_PAGE=3, RMAP: 0

Field	Bits	Type	Description
VAL	7:0	rh	**TIM[7:0] value**

39.4.158 SCR access to PMS_RTC_TIM[15:8] bits

Reading this register returns value from respective PMS_RTC bit-field if system is not in STANDBY1_70K mode.

SCUSCR_PMSRTC_TIM15_8 Offset address: $0F3_H$

SCR access to PMS_RTC_TIM[15:8] bits LVD Reset value: 00_H

PAGE: SCUSCR_PAGE=3, RMAP: 0

Field	Bits	Type	Description
VAL	7:0	rh	**TIM[15:8] value**

39.4.159 SCR access to PMS_RTC_TIM[23:16] bits

Reading this register returns value from respective PMS_RTC bit-field if system is not in STANDBY1_70K mode.

SCUSCR_PMSRTC_TIM23_16 Offset address: $0F4_H$

SCR access to PMS_RTC_TIM[23:16] bits LVD Reset value: 00_H

PAGE: SCUSCR_PAGE=3, RMAP: 0

Field	Bits	Type	Description
VAL	7:0	rh	**TIM[23:16] value**

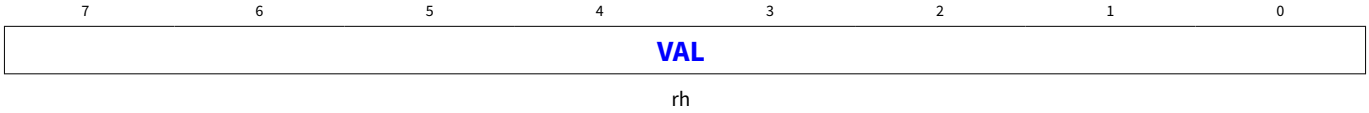
39.4.160 SCR access to PMS_RTC_TIM[31:24] bits

Reading this register returns value from respective PMS_RTC bit-field, if system is not in STANDBY1_70K mode.

SCUSCR_PMSRTC_TIM31_24 Offset address: $0F5_H$

SCR access to PMS_RTC_TIM[31:24] bits LVD Reset value: 00_H

PAGE: SCUSCR_PAGE=3, RMAP: 0



Field	Bits	Type	Description
VAL	7:0	rh	**TIM[31:24] value**

39.4.161 SCR access to PMS_RTC_CMP1[7:0] bits and update trigger for complete PMS_RTC_CMP1

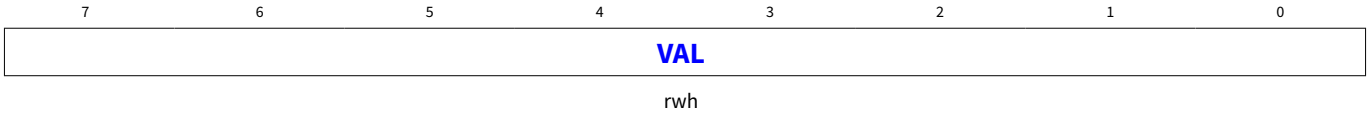
Writing to this register updates PMS_RTC_CMP1[31:0] register with combined content of PMSRTC_CMP1_31_24, PMSRTC_CMP1_23_16, PMSRTC_CMP1_15_8 and this register, if system is not in STANDBY1_70K mode. In order to ensure that all the bits of PMS_RTC_CMP1[31:0] are correctly updated it is important to update all the relevant shadow registers before writing to this register. In case, one of the register is not updated its value will be assumed to be 0x00.

Reading this register returns value from respective PMS_RTC bit-field.

SCUSCR_PMSRTC_CMP1_7_0 Offset address: $0F0_H$

SCR access to PMS_RTC_CMP1[7:0] bits and update trigger for complete PMS_RTC_CMP1 LVD Reset value: 00_H

PAGE: SCUSCR_PAGE=4, RMAP: 0



Field	Bits	Type	Description
VAL	7:0	rwh	**CMP1[7:0] value**

39.4.162 SCR shadow register for PMS_RTC_CMP1[15:8] bits

Writing to this register updates local shadow register PMSRTC_CMP1_15_8. Update of PMS_RTC_CMP1 register is triggered by writing to PMSRTC_CMP1_7_0 register.

Reading this register returns value from PMS_RTC_CMP1[15:8] bit-field. Therefore, it is not possible to read back value last written to local shadow register.

SCUSCR_PMSRTC_CMP1_15_8 Offset address: $0F1_H$

SCR shadow register for PMS_RTC_CMP1[15:8] bits LVD Reset value: 00_H

PAGE: SCUSCR_PAGE=4, RMAP: 0

7	6	5	4	3	2	1	0
VAL							
rwh							

Field	Bits	Type	Description
VAL	7:0	rwh	**CMP1[15:8] value**

39.4.163 SCR shadow register for PMS_RTC_CMP1[23:16] bits

Writing to this register updates local shadow register PMSRTC_CMP1_23_16. Update of PMS_RTC_CMP1 register is triggered by writing to PMSRTC_CMP1_7_0 register.

Reading this register returns value from PMS_RTC_CMP1[23:16] bit-field. Therefore, it is not possible to read back value last written to local shadow register.

SCUSCR_PMSRTC_CMP1_23_16 Offset address: $0F2_H$

SCR shadow register for PMS_RTC_CMP1[23:16] bits LVD Reset value: 00_H

PAGE: SCUSCR_PAGE=4, RMAP: 0

7	6	5	4	3	2	1	0
VAL							
rwh							

Field	Bits	Type	Description
VAL	7:0	rwh	**CMP1[23:16] value**

39.4.164 SCR shadow register for PMS_RTC_CMP1[31:24] bits

Writing to this register updates local shadow register PMSRTC_CMP1_31_24. Update of PMS_RTC_CMP1 register is triggered by writing to PMSRTC_CMP1_7_0 register.

Reading this register returns value from PMS_RTC_CMP1[31:24] bit-field. Therefore, it is not possible to read back value last written to local shadow register.

SCUSCR_PMSRTC_CMP1_31_24 Offset address: $0F3_H$

SCR shadow register for PMS_RTC_CMP1[31:24] bits LVD Reset value: 00_H

PAGE: SCUSCR_PAGE=4, RMAP: 0

7	6	5	4	3	2	1	0
VAL							
rwh							

Field	Bits	Type	Description
VAL	7:0	rwh	**CMP1[31:24] value**

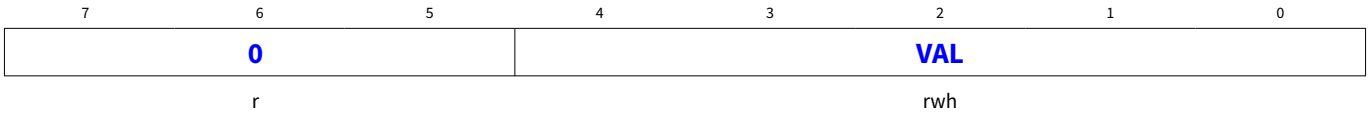
39.4.165 SCR access to PMS_RTC_CON2.MSIZE1 bits

Writing to this register updates PMS_RTC_CON2.MSIZE1 bit-field if system is not in STANDBY1_70K mode. Reading this register returns value from respective PMS_RTC bit-field.

SCUSCR_PMSRTC_MSIZE1 Offset address: $0F4_H$

SCR access to PMS_RTC_CON2.MSIZE1 bits LVD Reset value: 00_H

PAGE: SCUSCR_PAGE=4, RMAP: 0



Field	Bits	Type	Description
VAL	4:0	rwh	**MSIZE1 value**
0	7:5	r	**Reserved** Read as 0; must be written with 0.

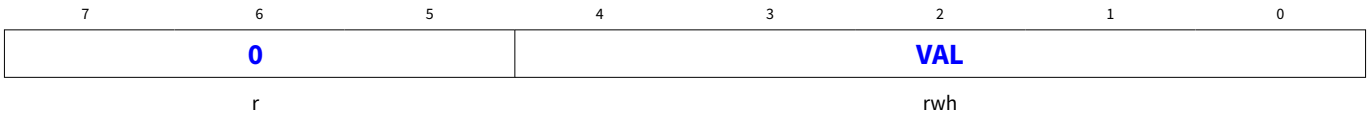
39.4.166 SCR access to PMS_RTC_CON2.MSTART1 bits

Writing to this register updates PMS_RTC_CON2.MSTART1 bit-field if system is not in STANDBY1_70K mode. Reading this register returns value from respective PMS_RTC bit-field.

SCUSCR_PMSRTC_MSTART1 Offset address: $0F5_H$

SCR access to PMS_RTC_CON2.MSTART1 bits LVD Reset value: 00_H

PAGE: SCUSCR_PAGE=4, RMAP: 0



Field	Bits	Type	Description
VAL	4:0	rwh	**MSTART1 value**
0	7:5	r	**Reserved** Read as 0; must be written with 0.

39.4.167 Control for PMS DTS

PMS DTS 12-bit result can be read from two SCR registers. It is slow changing, but to avoid result change between two reads, it is recommended to perform following sequence:

1. Read PMSDTS_RES11_8 first time - save as res_h1
2. Read PMSDTS_RES7_0 - save as res_l1
3. Read PMSDTS_RES11_8 second time - save as res_h2
4. If values res_h1 and res_h2 are equal:
   - then: proceed using res_h1 and res_l1.
   - else: read PMSDTS_RES7_0 as res_l2, and proceed using res_h2 and res_l2.

SCUSCR_PMSDTS_CON Offset address: $0F0_H$

Control for PMS DTS Reset values see: Table 1882

PAGE: SCUSCR_PAGE=5, RMAP: 0

7	6	5	4	3	2	1	0
0							EN_START
r							w

Field	Bits	Type	Description
EN_START	0	w	**DTS enable and start signal** Writing '1' triggers enable and start of the PMS DTS. Reading returns '0'. 0_B No action 1_B Enable and start PMS DTS from SCR
0	7:1	r	**Reserved** Read as 0; must be written with 0.

Table 1882 Reset values of SCUSCR_PMSDTS_CON

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.168 Die temperature sensor result register - lower 8-bits

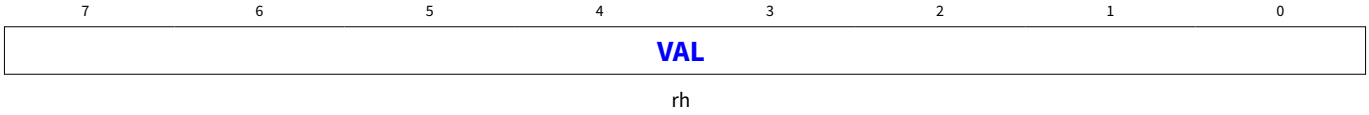
PMS DTS 12-bit result can be read from two SCR registers. It is slow changing, but to avoid result change between two reads, it is recommended to perform following sequence:

1. Read PMSDTS_RES11_8 first time - save as res_h1
2. Read PMSDTS_RES7_0 - save as res_l1
3. Read PMSDTS_RES11_8 second time - save as res_h2
4. If values res_h1 and res_h2 are equal:
   - then: proceed using res_h1 and res_l1.
   - else: read PMSDTS_RES7_0 as res_l2, and proceed using res_h2 and res_l2.

SCUSCR_PMSDTS_RES7_0 Offset address: $0F1_H$

Die temperature sensor result register - lower 8-bits LVD Reset value: 00_H

PAGE: SCUSCR_PAGE=5, RMAP: 0



Field	Bits	Type	Description
VAL	7:0	rh	**Temperature-sensing mode readout** Wired in value from PMS_DTS_STAT.RESULT[7:0] bits. After combining with upper 4-bits it can be converted to temperature (in Kelvins) by dividing the value by 4.781.

39.4.169 Die temperature sensor result register - upper 4-bits

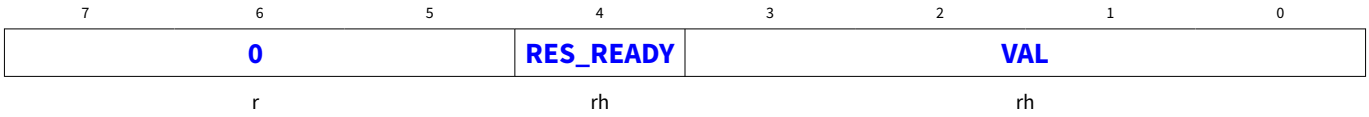
PMS DTS 12-bit result can be read from two SCR registers. It is slow changing, but to avoid result change between two reads, it is recommended to perform following sequence:

1. Read PMSDTS_RES11_8 first time - save as res_h1
2. Read PMSDTS_RES7_0 - save as res_l1
3. Read PMSDTS_RES11_8 second time - save as res_h2
4. If values res_h1 and res_h2 are equal:
   - then: proceed using res_h1 and res_l1.
   - else: read PMSDTS_RES7_0 as res_l2, and proceed using res_h2 and res_l2.

SCUSCR_PMSDTS_RES11_8 Offset address: $0F2_H$

Die temperature sensor result register - upper 4-bits LVD Reset value: 00_H

PAGE: SCUSCR_PAGE=5, RMAP: 0



Field	Bits	Type	Description
VAL	3:0	rh	**Temperature-sensing mode readout** Wired in value from PMS_DTS_STAT.RESULT[11:8] bits. After combining with lower 8-bits it can be converted to temperature (in Kelvins) by dividing the value by 4.781.
RES_READY	4	rh	**DTS results ready status** Wired in value form PMS_DTS_STAT.READY 0_B **BUSY**: PMS DTS is disabled, or still starting up 1_B **READY**: PMS DTS has completed startup
0	7:5	r	**Reserved** Returns 0 when read.

39.4.170 SCR access to PMS_HPOSCCON1.OSCFTRIM[7:0] bits and update trigger for complete PMS_HPOSCCON1.OSCFTRIM

Writing to this register updates PMS_HPOSCCON1_OSCFTRIM[8:0] bit-field with combined content of PMSHPOSC_FTRIM8_8 and this register. In case, only this register is updated and no write operation is made on PMSHPOSC_FTRIM8_8 then bit PMS_HPOSCCON1_OSCFTRIM[8:8] will retain its previous value.

Reading this register returns value from respective PMS_HPOSCCON1 bit-field.

SCUSCR_PMSHPOSC_FTRIM7_0 Offset address: $0F3_H$

SCR access to PMS_HPOSCCON1.OSCFTRIM[7:0] bits and update trigger for complete PMS_HPOSCCON1.OSCFTRIM

LVD Reset value: 00_H

PAGE: SCUSCR_PAGE=5, RMAP: 0

7	6	5	4	3	2	1	0
VAL							
rwh							

Field	Bits	Type	Description
VAL	7:0	rwh	**OSCFTRIM[7:0] value**

39.4.171 SCR shadow register for PMS_HPOSCCON1.OSCFTRIM[8:8] bits

Writing to this register updates local shadow register PMSHPOSC_FTRIM8_8. Update of PMS_HPOSCCON1.OSCFTRIM bit-field is triggered with write to PMSHPOSC_FTRIM7_0.

Reading this register returns value from PMS_HPOSCCON1.OSCFTRIM[8:8] bit-field. Therefore, it is not possible to read back value last written to local shadow register.

SCUSCR_PMSHPOSC_FTRIM8_8 Offset address: $0F4_H$

SCR shadow register for PMS_HPOSCCON1.OSCFTRIM[8:8] bits

LVD Reset value: 00_H

PAGE: SCUSCR_PAGE=5, RMAP: 0

7	6	5	4	3	2	1	0
0							VAL
r							rwh

Field	Bits	Type	Description
VAL	0	rwh	**OSCFTRIM[8:8] value**
0	7:1	r	**Reserved** Returns 0 when read.

39.4.172 Trimming register for PMS High Precision BandGap (HPBG)

This register is used for the trimming the High Precision BandGap (HPBG) of the PMS.

SCUSCR_PMSHPBG_BGTRIM Offset address: $0F5_H$

Trimming register for PMS High Precision BandGap [HPBG) Reset values see: Table 1883

PAGE: SCUSCR_PAGE=5, RMAP: 0

7	6	5	4	3	2	1	0
RES7	**VAL**						
r	rwh						

Field	Bits	Type	Description
VAL	6:0	rwh	**BGTRIM[6:0] value**
RES7	7	r	**reserved bitfield** Reserved bitfield.

Table 1883 Reset values of SCUSCR_PMSHPBG_BGTRIM

Reset type	Reset value	Note
LVD Reset	$X000\ 0000_B$	

39.4.173 PMS1 iVR control

SCUSCR_PMS1IVRCON Offset address: $0F0_H$

PMS1 iVR control LVD Reset value: 00_H

PAGE: SCUSCR_PAGE=7, RMAP: 0

7	6	5	4	3	2	1	0
0	**0**	**SINKSTS**			**SINKCTRLEN**	**SINKCTRL**	
r	r	rh			rw	rw	

Field	Bits	Type	Description
SINKCTRL	1:0	rw	**PMS iVR sink control value** Set power saving level in STANDBY1_100 and STANDBY1_ADCOMP. Note: The higher the level, the less power is consumed, but the more critical load jumps due to clock switch or module reconfiguration are to the regulator. This could lead to a system reset. 00_B Powersaving level 0 Lowest level of power saving 01_B Powersaving level 1 10_B Powersaving level 2 11_B Powersaving level 3 Highest level of power saving

(table continues...)

(continued)

Field	Bits	Type	Description
SINKCTRLEN	2	rw	**Enable Powermode Control from SCR side** **Note:** If SINKCTRLEN=1, it is required that SCUSCR_CLK.DIV.REQ >= 5 (max. SCR frequency is limited 20 MHz in the medium power mode), otherwise there is a risk of triggering under-voltage reset or alarms. 0_B The SCR does not control its power modes 1_B The SCR controls its power modes
SINKSTS	5:3	rh	**Sink disable status from SCR side** 000_B Power saving mode is off 001_B Powersaving level 0 is active 010_B Powersaving level 1 is active 011_B Powersaving level 2 is active 100_B Powersaving level 3 is active
0	6, 7	r	**Reserved** Read as 0; must be written with 0.

39.4.174 Module suspend control register

SCUSCR_DBG_MODSUSP Offset address: $0F1_H$

Module suspend control register Reset values see: Table 1884

PAGE: SCUSCR_PAGE=7, RMAP: 0

7	6	5	4	3	2	1	0
0				T2CCU1SUSP	T2CCU0SUSP	RTCSUSP	WDTSUSP
r				rw	rw	rw	rw

Field	Bits	Type	Description
WDTSUSP	0	rw	**SCU watchdog timer debug suspend bit** 0_B WDT will not be suspended 1_B WDT will be suspended
RTCSUSP	1	rw	**Real-time clock debug suspend bit** 0_B Real-time clock will not be suspended 1_B Real-time clock will be suspended
T2CCU0SUSP	2	rw	**T2CCU0 debug suspend bit** 0_B T2CCU0 will not be suspended 1_B T2CCU0 will be suspended
T2CCU1SUSP	3	rw	**T2CCU1 debug suspend bit** 0_B T2CCU1 will not be suspended 1_B T2CCU1 will be suspended
0	7:4	r	**Reserved** Read as 0; must be written with 0.

Table 1884 Reset values of SCUSCR_DBG_MODSUSP

Reset type	Reset value	Note
LVD Reset	01_H	
Generated Reset	01_H	

39.4.175 Page register for SFRs

IRSCR_PAGE Offset address: $0F6_H$

Page register for SFRs Reset values see: Table 1885

PAGE: X, RMAP: 0

7	6	5	4	3	2	1	0
OP		**STNR**		**0**	**PAGE**		
w		w		r	rwh		

Field	Bits	Type	Description
PAGE	2:0	rwh	**Page bits** When written, the value indicates the new page address. When read, the value indicates the currently active page = addr[y:x+1]. 000_B **PAGE0**: Selection is PAGE0 001_B **PAGE1**: Selection is PAGE1 010_B **PAGE2**: Selection is PAGE2 011_B **PAGE3**: Selection is PAGE3 100_B **PAGE4**: Selection is PAGE4 101_B **PAGE5**: Selection is PAGE5 110_B **PAGE6**: Selection is PAGE6 111_B **PAGE7**: Selection is PAGE7
STNR	5:4	w	**Storage number** This number indicates which storage bitfield is the target of the operation defined by bit OP. If OP = 10_B, the contents of PAGE are saved in STx before being overwritten with the new value. If OP = 11_B, the contents of PAGE are overwritten by the contents of STx. The value written to bitfield PAGE is ignored. 00_B **MOD_ST0**: ST0 is selected 01_B **MOD_ST1**: ST1 is selected 10_B **MOD_ST2**: ST2 is selected 11_B **MOD_ST3**: ST3 is selected

(table continues...)

(continued)

Field	Bits	Type	Description
OP	7:6	w	**Operation** 00_B **PAGE_MANUAL0**: Manual page mode. The value of STNR is ignored and PAGE is directly written. 01_B **PAGE_MANUAL1**: Manual page mode. The value of STNR is ignored and PAGE is directly written 10_B **PAGE_SAVE**: New page programming with automatic page saving. The value written to the bit positions of PAGE is stored. In parallel, the former contents of PAGE are saved in the storage bit field STx indicated by STNR. 11_B **PAGE_RESTORE**: Automatic restore page action. The value written to the bit positions PAGE is ignored and instead, PAGE is overwritten by the contents of the storage bit field STx indicated by STNR.
0	3	r	**Reserved** Read as 0; must be written with 0.

Table 1885 Reset values of IRSCR_PAGE

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.176 Interrupt request register 0

IRSCR_CON0 Offset address: $0F7_H$

Interrupt request register 0 Reset values see: Table 1886

PAGE: IRSCR_PAGE=0, RMAP: 0

7	6	5	4	3	2	1	0
EXINT7	**EXINT6**	**EXINT5**	**EXINT4**	**EXINT3**	**EXINT2**	**0**	
rwh	rwh	rwh	rwh	rwh	rwh	r	

Field	Bits	Type	Description
EXINTm (m=2-7)	m	rwh	**Interrupt flag for external interrupt x** This bit is set by hardware and can only be cleared by software. 0_B Interrupt request has not occurred 1_B Interrupt request has occurred
0	1:0	r	**Reserved** Read as 0; must be written with 0.

Table 1886 Reset values of IRSCR_CON0

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.177 Interrupt request register 1

IRSCR_CON1 Offset address: $0F8_H$

Interrupt request register 1 Reset values see: Table 1887

PAGE: IRSCR_PAGE=0, RMAP: 0

7	6	5	4	3	2	1	0
CMP1IR	CMP0IR	ADCIR	RFFIR	RFEIR	RIR	TIR	EIR
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
EIR	0	rwh	**Error interrupt flag for SSC** This bit is set by hardware, and can only be cleared by software. 0_B Interrupt request has not occurred 1_B Interrupt request has occurred
TIR	1	rwh	**Transmit interrupt flag for SSC** This bit is set by hardware, and can only be cleared by software. 0_B Interrupt request has not occurred 1_B Interrupt request has occurred
RIR	2	rwh	**Receive interrupt flag for SSC** This bit is set by hardware, and can only be cleared by software. 0_B Interrupt request has not occurred 1_B Interrupt request has occurred
RFEIR	3	rwh	**Receive FIFO empty interrupt flag for SSC** This bit is set by hardware, and can only be cleared by software. 0_B Interrupt request has not occurred 1_B Interrupt request has occurred
RFFIR	4	rwh	**Receive FIFO full interrupt flag for SSC** This bit is set by hardware, and can only be cleared by software. 0_B Interrupt request has not occurred 1_B Interrupt request has occurred
ADCIR	5	rwh	**Interrupt flag for ADCOMP unit** This bit is set by hardware, and can only be cleared by software. 0_B Interrupt request has not occurred 1_B Interrupt request has occurred
CMP0IR	6	rwh	**PMS RTC CMP0 OUT interrupt flag** 0_B Interrupt request has not occurred 1_B Interrupt request has occurred
CMP1IR	7	rwh	**PMS RTC CMP1 OUT interrupt flag** 0_B Interrupt request has not occurred 1_B Interrupt request has occurred

Table 1887 Reset values of IRSCR_CON1

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.178 Interrupt request register 2

IRSCR_CON2 Offset address: $0F9_H$

Interrupt request register 2 Reset values see: Table 1888

PAGE: IRSCR_PAGE=0, RMAP: 0

7	6	5	4	3	2	1	0
EXINT15	**EXINT14**	**EXINT13**	**EXINT12**	**EXINT11**	**EXINT10**	**EXINT9**	**EXINT8**
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
EXINTm (m=8-15)	m-8	rwh	**Interrupt flag for external interrupt x** This bit is set by hardware and can only be cleared by software. 0_B Interrupt request has not occurred 1_B Interrupt request has occurred

Table 1888 Reset values of IRSCR_CON2

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.179 NMI control register

IRSCR_NMICON Offset address: $0F7_H$

NMI control register Reset values see: Table 1889

PAGE: IRSCR_PAGE=1, RMAP: 0

7	6	5	4	3	2	1	0
SCRINTTC1	**SCRINTTC0**	**NMIWKP**	**NMIMAINC**	**NMIOCDS**	**NMIEXTNMI**	**NMIRAMECC**	**NMIWDT**
rwh	rwh	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
NMIWDT	0	rw	**Watchdog timer NMI enable** 0_B WDT NMI is disabled 1_B WDT NMI is enabled

(table continues...)

(continued)

Field	Bits	Type	Description
NMIRAMECC	1	rw	**RAM double bit ECC error NMI enable** 0_B RAM double bit ECC error NMI is disabled 1_B RAM double bit ECC error NMI is enabled
NMIEXTNMI	2	rw	**External NMI (via EXTNMI pin) NMI enable** ***Note:*** *It is recommended to clear the IRSCR_NMISTAT.FNMIEXTNMI flag before enabling external NMI.* 0_B External NMI (via EXTNMI pin) is disabled 1_B External NMI (via EXTNMI pin) is enabled
NMIOCDS	3	rw	**OCDS NMI enable** Note: In case the bit field NMIOCDS is not set and breakpoint event occurs (OCDS inserts a TRAP) no reliable software execution after this event can be ensured. It is highly recommended not to disable this NMIOCDS in order to ensure a reliable software execution during debug mode. 0_B OCDS NMI is disabled 1_B OCDS NMI is enabled
NMIMAINC	4	rw	**Main controller (to TriCore) NMI enable** 0_B Main controller (to TriCore) NMI is disabled 1_B Main controller (to TriCore) NMI is enabled
NMIWKP	5	rw	**Wake-up NMI enable** 0_B Wake-up NMI is disabled 1_B Wake-up NMI is enabled
SCRINTTC0	6	rwh	**Main IR interrupt request 0** Trigger an interrupt to CPUx. This bit will be reset automatically to 0. 0_B No interrupt event is triggered 1_B An SCR interrupt 0 is triggered toward the main interrupt router
SCRINTTC1	7	rwh	**Main IR interrupt request 1** Trigger an interrupt to CPUx. This bit will be reset automatically to 0. 0_B No interrupt event is triggered 1_B An SCR interrupt 1 is triggered toward the main interrupt router (over PMS INT4 line)

Table 1889 Reset values of IRSCR_NMICON

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.180 NMI status register

IRSCR_NMISTAT Offset address: $0F8_H$

NMI status register Reset values see: Table 1890

PAGE: IRSCR_PAGE=1, RMAP: 0

7	6	5	4	3	2	1	0
0	FNMIWKP	0	FNMIMAINC	FNMIOCDS	FNMIEXTNMI	FNMIRAMECC	FNMIWDT
r	rwh	r	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
FNMIWDT	0	rwh	**Watchdog timer NMI flag** 0_B No watchdog NMI has occurred 1_B WDT pre-warning has occurred
FNMIRAMECC	1	rwh	**RAM double bit ECC error NMI flag** 0_B No RAM ECC NMI has occurred 1_B RAM ECC NMI has occurred
FNMIEXTNMI	2	rwh	**External NMI (via EXTNMI pin) flag** ***Note**: It is recommended to clear the IR_NMISTAT.FNMIEXTNMI flag before enabling external NMI.* 0_B No external NMI (via EXTNMI pin) has occurred 1_B External NMI (via EXTNMI pin) has occurred
FNMIOCDS	3	rwh	**OCDS NMI flag** 0_B No OCDS NMI has occurred 1_B JTAG-receiving or user interrupt request in monitor mode has occurred
FNMIMAINC	4	rwh	**Main controller NMI flag** 0_B No main controller NMI has occurred 1_B Main controller NMI has occurred
FNMIWKP	6	rwh	**Wake-up NMI flag** 0_B No wake-up NMI has occurred 1_B Wake-up NMI has occurred
0	5, 7	r	**Reserved** Read as 0; must be written with 0.

Table 1890 Reset values of IRSCR_NMISTAT

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.181 Peripheral interrupt enable register

IRSCR_MODIEN Offset address: $0F9_H$

Peripheral interrupt enable register Reset values see: Table 1891

PAGE: IRSCR_PAGE=1, RMAP: 0

7	6	5	4	3	2	1	0
CMP1IEN	CMP0IEN	ADCIEN	RFFIEN	RFEIEN	RIREN	TIREN	EIREN
rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
EIREN	0	rw	**SSC error interrupt enable** 0_B Error interrupt is disabled 1_B Error interrupt is enabled
TIREN	1	rw	**SSC transmit interrupt enable** 0_B Transmit interrupt is disabled 1_B Transmit interrupt is enabled
RIREN	2	rw	**SSC receive interrupt enable** 0_B Receive interrupt is disabled 1_B Receive interrupt is enabled
RFEIEN	3	rw	**SSC receive FIFO empty interrupt enable** 0_B Receive FIFO empty interrupt is disabled 1_B Receive FIFO empty interrupt is enabled
RFFIEN	4	rw	**SSC receive FIFO full interrupt enable** 0_B Receive FIFO full interrupt is disabled 1_B Receive FIFO full interrupt is enabled
ADCIEN	5	rw	**ADC comparator interrupt enable** 0_B No interrupt is generated 1_B Interrupt is generated after conversion
CMP0IEN	6	rw	**PMS RTC CMP0 OUT interrupt enable** 0_B Interrupt is disabled 1_B Interrupt is enabled
CMP1IEN	7	rw	**PMS RTC CMP1 OUT interrupt enable** 0_B Interrupt is disabled 1_B Interrupt is enabled

Table 1891 Reset values of IRSCR_MODIEN

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.182 External interrupt control register 0

IRSCR_EXICON0 Offset address: $0F7_H$

External interrupt control register 0 Reset values see: Table 1892

PAGE: IRSCR_PAGE=2, RMAP: 0

7	6	5	4	3	2	1	0
EXINT3		EXINT2		EXINT1		EXINT0	
rw		rw		rw		rw	

Field	Bits	Type	Description
EXINT0	1:0	rw	**External interrupt 0 trigger select** 00_B **INT_FALL_EDGE**: Interrupt on falling edge 01_B **INT_RISE_EDGE**: Interrupt on rising edge 10_B **INT_BOTH_EDGE**: Interrupt on both, rising and falling edge 11_B **BYPASS**: Bypass the edge detection in SCU. The input signal directly feeds to the core.
EXINT1	3:2	rw	**External interrupt 1 trigger select** 00_B **INT_FALL_EDGE**: Interrupt on falling edge 01_B **INT_RISE_EDGE**: Interrupt on rising edge 10_B **INT_BOTH_EDGE**: Interrupt on both, rising and falling edge 11_B **BYPASS**: Bypass the edge detection in SCU. The input signal directly feeds to the core.
EXINT2	5:4	rw	**External interrupt 2 trigger select** 00_B **INT_FALL_EDGE**: Interrupt on falling edge 01_B **INT_RISE_EDGE**: Interrupt on rising edge 10_B **INT_BOTH_EDGE**: Interrupt on both, rising and falling edge 11_B **DISABLE**: External interrupt 2 is disabled
EXINT3	7:6	rw	**External interrupt 3 trigger select** 00_B **INT_FALL_EDGE**: Interrupt on falling edge 01_B **INT_RISE_EDGE**: Interrupt on rising edge 10_B **INT_BOTH_EDGE**: Interrupt on both, rising and falling edge 11_B **DISABLE**: External interrupt 3 is disabled

Table 1892 Reset values of IRSCR_EXICON0

Reset type	Reset value	Note
LVD Reset	$F0_H$	
Generated Reset	$F0_H$	

39.4.183 External interrupt control register 1

IRSCR_EXICON1 Offset address: $0F8_H$

External interrupt control register 1 Reset values see: Table 1893

PAGE: IRSCR_PAGE=2, RMAP: 0

7	6	5	4	3	2	1	0
EXINT7		EXINT6		EXINT5		EXINT4	
rw		rw		rw		rw	

Field	Bits	Type	Description
EXINT4	1:0	rw	**External interrupt 4 trigger select** 00_B **INT_FALL_EDGE**: Interrupt on falling edge 01_B **INT_RISE_EDGE**: Interrupt on rising edge 10_B **INT_BOTH_EDGE**: Interrupt on both, rising and falling edge 11_B **DISABLE**: External interrupt 4 is disabled
EXINT5	3:2	rw	**External interrupt 5 trigger select** 00_B **INT_FALL_EDGE**: Interrupt on falling edge 01_B **INT_RISE_EDGE**: Interrupt on rising edge 10_B **INT_BOTH_EDGE**: Interrupt on both, rising and falling edge 11_B **DISABLE**: External interrupt 5 is disabled
EXINT6	5:4	rw	**External interrupt 6 trigger select** 00_B **INT_FALL_EDGE**: Interrupt on falling edge 01_B **INT_RISE_EDGE**: Interrupt on rising edge 10_B **INT_BOTH_EDGE**: Interrupt on both, rising and falling edge 11_B **DISABLE**: External interrupt 6 is disabled
EXINT7	7:6	rw	**External interrupt 7 trigger select** 00_B **INT_FALL_EDGE**: Interrupt on falling edge 01_B **INT_RISE_EDGE**: Interrupt on rising edge 10_B **INT_BOTH_EDGE**: Interrupt on both, rising and falling edge 11_B **DISABLE**: External interrupt 7 is disabled

Table 1893 Reset values of IRSCR_EXICON1

Reset type	Reset value	Note
LVD Reset	FF_H	
Generated Reset	FF_H	

39.4.184 External interrupt control register 2

IRSCR_EXICON2 Offset address: $0F9_H$

External interrupt control register 2 Reset values see: Table 1894

PAGE: IRSCR_PAGE=2, RMAP: 0

7	6	5	4	3	2	1	0
EXINT11		**EXINT10**		**EXINT9**		**EXINT8**	
rw		rw		rw		rw	

Field	Bits	Type	Description
EXINT8	1:0	rw	**External interrupt 8 trigger select** 00_B **INT_FALL_EDGE**: Interrupt on falling edge 01_B **INT_RISE_EDGE**: Interrupt on rising edge 10_B **INT_BOTH_EDGE**: Interrupt on both, rising and falling edge 11_B **DISABLE**: External interrupt 8 is disabled

(table continues...)

(continued)

Field	Bits	Type	Description
EXINT9	3:2	rw	**External interrupt 9 trigger select** 00_B **INT_FALL_EDGE**: Interrupt on falling edge 01_B **INT_RISE_EDGE**: Interrupt on rising edge 10_B **INT_BOTH_EDGE**: Interrupt on both, rising and falling edge 11_B **DISABLE**: External interrupt 9 is disabled
EXINT10	5:4	rw	**External interrupt 10 trigger select** 00_B **INT_FALL_EDGE**: Interrupt on falling edge 01_B **INT_RISE_EDGE**: Interrupt on rising edge 10_B **INT_BOTH_EDGE**: Interrupt on both, rising and falling edge 11_B **DISABLE**: External interrupt 10 is disabled
EXINT11	7:6	rw	**External interrupt 11 trigger select** 00_B **INT_FALL_EDGE**: Interrupt on falling edge 01_B **INT_RISE_EDGE**: Interrupt on rising edge 10_B **INT_BOTH_EDGE**: Interrupt on both, rising and falling edge 11_B **DISABLE**: External interrupt 11 is disabled

Table 1894 Reset values of IRSCR_EXICON2

Reset type	Reset value	Note
LVD Reset	FF_H	
Generated Reset	FF_H	

39.4.185 External interrupt control register 3

IRSCR_EXICON3 Offset address: $0F7_H$

External interrupt control register 3 Reset values see: Table 1895

PAGE: IRSCR_PAGE=3, RMAP: 0

7	6	5	4	3	2	1	0
EXINT15		**EXINT14**		**EXINT13**		**EXINT12**	
rw		rw		rw		rw	

Field	Bits	Type	Description
EXINT12	1:0	rw	**External interrupt 12 trigger select** 00_B **INT_FALL_EDGE**: Interrupt on falling edge 01_B **INT_RISE_EDGE**: Interrupt on rising edge 10_B **INT_BOTH_EDGE**: Interrupt on both, rising and falling edge 11_B **DISABLE**: External interrupt 12 is disabled
EXINT13	3:2	rw	**External interrupt 13 trigger select** 00_B **INT_FALL_EDGE**: Interrupt on falling edge 01_B **INT_RISE_EDGE**: Interrupt on rising edge 10_B **INT_BOTH_EDGE**: Interrupt on both, rising and falling edge 11_B **DISABLE**: External interrupt 13 is disabled

(table continues...)

(continued)

Field	Bits	Type	Description
EXINT14	5:4	rw	**External interrupt 14 trigger select** 00_B **INT_FALL_EDGE**: Interrupt on falling edge 01_B **INT_RISE_EDGE**: Interrupt on rising edge 10_B **INT_BOTH_EDGE**: Interrupt on both, rising and falling edge 11_B **DISABLE**: External interrupt 14 is disabled
EXINT15	7:6	rw	**External interrupt 15 trigger select** 00_B **INT_FALL_EDGE**: Interrupt on falling edge 01_B **INT_RISE_EDGE**: Interrupt on rising edge 10_B **INT_BOTH_EDGE**: Interrupt on both, rising and falling edge 11_B **DISABLE**: External interrupt 15 is disabled

Table 1895 Reset values of IRSCR_EXICON3

Reset type	Reset value	Note
LVD Reset	FF_H	
Generated Reset	FF_H	

39.4.186 External interrupt input select register 0

This register is used for input pin selection of the external interrupt inputs 0 to 2.

IRSCR_PISEL0 Offset address: $0F7_H$

External interrupt input select register 0 Reset values see: Table 1896

PAGE: IRSCR_PAGE=4, RMAP: 0

7	6	5	4	3	2	1	0
EXINT2IS		**EXINT1IS**		**EXINT0IS**		**0**	
rw		rw		rw		r	

Field	Bits	Type	Description
EXINT0IS	3:2	rw	**External interrupt input 0 select** 00_B **EXINT0A**: External interrupt input EXINT0A is selected 01_B **EXINT0B**: External interrupt input EXINT0B is selected 10_B **EXINT0C**: External interrupt input EXINT0C is selected 11_B **EXINT0D**: External interrupt input EXINT0D is selected
EXINT1IS	5:4	rw	**External interrupt input 1 select** 00_B **EXINT1A**: External interrupt input EXINT1A is selected 01_B **EXINT1B**: External interrupt input EXINT1B is selected 10_B **EXINT1C**: External interrupt input EXINT1C is selected 11_B **EXINT1D**: External interrupt input EXINT1D is selected

(table continues...)

(continued)

Field	Bits	Type	Description
EXINT2IS	7:6	rw	**External interrupt input 2 select** 00_B **EXINT2A**: External interrupt input EXINT2A is selected 01_B **EXINT2B**: External interrupt input EXINT2B is selected 10_B **EXINT2C**: External interrupt input EXINT2C is selected 11_B **EXINT2D**: External interrupt input EXINT2D is selected
0	1:0	r	**Reserved** Read as 0; must be written with 0.

Table 1896 Reset values of IRSCR_PISEL0

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.187 External interrupt input select register 1

This register is used for input pin selection of the external interrupt inputs 3 to 6 and for T2CCU0 module.

IRSCR_PISEL1 Offset address: $0F8_H$

External interrupt input select register 1 Reset values see: Table 1897

PAGE: IRSCR_PAGE=4, RMAP: 0

7	6	5	4	3	2	1	0
EXINT6_T2CCU0CH3_IS		EXINT5_T2CCU0CH2_IS		EXINT4_T2CCU0CH1_IS		EXINT3_T2CCU0CH0_IS	
rw		rw		rw		rw	

Field	Bits	Type	Description
EXINT3_T2CCU0CH0_IS	1:0	rw	**External interrupt 3 / T2CCU0 capture channel 0 input select** 00_B External interrupt 3 input EXINT3A / T2CCU0 capture channel 0 input T2CCU0_CH0A is selected 01_B External interrupt 3 input EXINT3B / T2CCU0 capture channel 0 input T2CCU0_CH0B is selected 10_B External interrupt 3 input EXINT3C / T2CCU0 capture channel 0 input T2CCU0_CH0C is selected 11_B External interrupt 3 input EXINT3D / T2CCU0 capture channel 0 input T2CCU0_CH0D is selected
EXINT4_T2CCU0CH1_IS	3:2	rw	**External interrupt 4 / T2CCU0 capture channel 1 input select** 00_B External interrupt 4 input EXINT4A / T2CCU0 capture channel 1 input T2CCU0_CH1A is selected 01_B External interrupt 4 input EXINT4B / T2CCU0 capture channel 1 input T2CCU0_CH1B is selected 10_B External interrupt 4 input EXINT4C / T2CCU0 capture channel 1 input T2CCU0_CH1C is selected 11_B External interrupt 4 input EXINT4D / T2CCU0 capture channel 1 input T2CCU0_CH1D is selected

(table continues...)

(continued)

Field	Bits	Type	Description
EXINT5_T2CCU0CH2_IS	5:4	rw	**External interrupt 5 / T2CCU0 capture channel 2 input select** 00_B External Interrupt 5 input EXINT5A / T2CCU0 capture channel 2 input T2CCU0_CH2A is selected 01_B External interrupt 5 input EXINT5B / T2CCU0 capture channel 2 input T2CCU0_CH2B is selected 10_B External interrupt 5 input EXINT5C / T2CCU0 capture channel 2 input T2CCU0_CH2C is selected 11_B External interrupt 5 input EXINT5D / T2CCU0 capture channel 2 input T2CCU0_CH2D is selected
EXINT6_T2CCU0CH3_IS	7:6	rw	**External interrupt 6 / T2CCU0 capture channel 3 input select** 00_B External interrupt 6 input EXINT6A / T2CCU0 capture channel 3 input T2CCU0_CH3A is selected 01_B External interrupt 6 input EXINT6B / T2CCU0 capture channel 3 input T2CCU0_CH3B is selected 10_B External interrupt 6 input EXINT6C / T2CCU0 capture channel 3 input T2CCU0_CH3C is selected 11_B External interrupt 6 input EXINT6D / T2CCU0 capture channel 3 input T2CCU0_CH3D is selected

Table 1897 Reset values of IRSCR_PISEL1

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.188 External interrupt input select register 2

This register is used for input pin selection of the external interrupt inputs 8 to 11 and for T2CCU1 module.

IRSCR_PISEL2 Offset address: $0F9_H$

External interrupt input select register 2 Reset values see: Table 1898

PAGE: IRSCR_PAGE=4, RMAP: 0

7	6	5	4	3	2	1	0
EXINT11_T2CCU1CH3_IS		**EXINT10_T2CCU1CH2_IS**		**EXINT9_T2CCU1CH1_IS**		**EXINT8_T2CCU1CH0_IS**	
rw		rw		rw		rw	

Field	Bits	Type	Description
EXINT8_T2CCU1CH0_IS	1:0	rw	**External interrupt 8 / T2CCU1 capture channel 0 input select** 00_B External interrupt 8 input EXINT8A / T2CCU1 capture channel 0 input T2CCU1_CH0A is selected 01_B External interrupt 8 input EXINT8B / T2CCU1 capture channel 0 input T2CCU1_CH0B is selected 10_B External interrupt 8 input EXINT8C / T2CCU1 capture channel 0 input T2CCU1_CH0C is selected 11_B External interrupt 8 input EXINT8D / T2CCU1 capture channel 0 input T2CCU1_CH0D is selected
EXINT9_T2CCU1CH1_IS	3:2	rw	**External interrupt 9 / T2CCU1 capture channel 1 input select** 00_B External interrupt 9 input EXINT9A / T2CCU1 capture channel 1 input T2CCU1_CH1A is selected 01_B External interrupt 9 input EXINT9B / T2CCU1 capture channel 1 input T2CCU1_CH1B is selected 10_B External interrupt 9 input EXINT9C / T2CCU1 capture channel 1 input T2CCU1_CH1C is selected 11_B External interrupt 9 input EXINT9D / T2CCU1 capture channel 1 input T2CCU1_CH1D is selected
EXINT10_T2CCU1CH2_IS	5:4	rw	**External interrupt 10 / T2CCU1 capture channel 2 input select** 00_B External interrupt 10 input EXINT10A / T2CCU1 capture channel 2 input T2CCU1_CH2A is selected 01_B External interrupt 10 input EXINT10B / T2CCU1 capture channel 2 input T2CCU1_CH2B is selected 10_B External interrupt 10 input EXINT10C / T2CCU1 capture channel 2 input T2CCU1_CH2C is selected 11_B External interrupt 10 input EXINT10D / T2CCU1 capture channel 2 input T2CCU1_CH2D is selected
EXINT11_T2CCU1CH3_IS	7:6	rw	**External interrupt 11 / T2CCU1 capture channel 3 input select** 00_B External interrupt 11 input EXINT11A / T2CCU1 capture channel 3 input T2CCU1_CH3A is selected 01_B External interrupt 11 input EXINT11B / T2CCU1 capture channel 3 input T2CCU1_CH3B is selected 10_B External interrupt 11 input EXINT11C / T2CCU1 capture channel 3 input T2CCU1_CH3C is selected 11_B External interrupt 11 input EXINT11D / T2CCU1 capture channel 3 input T2CCU1_CH3D is selected

Table 1898 Reset values of IRSCR_PISEL2

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.189 External interrupt input select register 3

This register is used for input pin selection of the External Interrupt inputs 12 to 15.

IRSCR_PISEL3 Offset address: $0F7_H$

External interrupt input select register 3 Reset values see: Table 1899

PAGE: IRSCR_PAGE=5, RMAP: 0

7	6	5	4	3	2	1	0
EXINT15IS		EXINT14IS		EXINT13IS		EXINT12IS	
rw		rw		rw		rw	

Field	Bits	Type	Description
EXINT12IS	1:0	rw	**External interrupt 12 input select** 00_B External interrupt 12 input EXINT12A is selected 01_B External interrupt 12 input EXINT12B is selected 10_B External interrupt 12 input EXINT12C is selected 11_B External interrupt 12 input EXINT12D is selected
EXINT13IS	3:2	rw	**External interrupt 13 input select** 00_B External interrupt 13 input EXINT13A is selected 01_B External interrupt 13 input EXINT13B is selected 10_B External interrupt 13 input EXINT13C is selected 11_B External interrupt 13 input EXINT13D is selected
EXINT14IS	5:4	rw	**External interrupt 14 input select** 00_B External interrupt 14 input EXINT14A is selected 01_B External interrupt 14 input EXINT14B is selected 10_B External interrupt 14 input EXINT14C is selected 11_B External interrupt 14 input EXINT14D is selected
EXINT15IS	7:6	rw	**External interrupt 15 input select** 00_B External interrupt 15 input EXINT15A is selected 01_B External interrupt 15 input EXINT15B is selected 10_B External interrupt 15 input EXINT15C is selected 11_B External interrupt 15 input EXINT15D is selected

Table 1899 Reset values of IRSCR_PISEL3

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.190 I2C control register

This register is used to configure the I2CSCR and generate START and STOP conditions. It also contains the interrupt status flag.

I2CSCR_CNTR Offset address: $0FA_H$

I2C control register Reset values see: Table 1900

PAGE: X, RMAP: 0

7	6	5	4	3	2	1	0
IEN	ENAB	STA	STP	IFLG	AAK	0	
rw	rw	rwh	rwh	rwh	rw	r	

Field	Bits	Type	Description
AAK	2	rw	**Assert acknowledge** ***Note**: If the AAK bit is cleared to 0 in slave transmitter mode, the byte in the DATA register is assumed to be the last byte.* 0_B A NACK is sent when a data byte is received in master or slave mode. 1_B An ACK is sent if one of the following has been received: 1) A matching 7-bit or either byte of the 10-bit slave address 2) General call address 3) A databyte in master or slave mode
IFLG	3	rwh	**Interrupt flag** The bit is set by hardware and can only be cleared through software by writing a 0. 0_B No interrupt event has occurred. 1_B An interrupt event has occurred.
STP	4	rwh	**Master mode stop** When this bit is set to 1, the I2CSCR will transmit a STOP condition. If the STP bit is set while the I2CSCR is in slave mode, no STOP condition will be transmitted on the I2C bus but the I2CSCR behaves as if a STOP condition has been received. If both STA and STP bits are set, the I2CSCR will first transmit a STOP condition (if in master mode) before transmitting a START condition. ***Note**: The STP bit is cleared automatically after the STOP condition has been sent; writing 0 to this bit has no effect.* 0_B The I2C does not transmit any STOP condition. 1_B The I2C transmits a STOP condition on the I2C bus.
STA	5	rwh	**Master Mode Start** When the STA bit is set to 1, the I2CSCR will enter master mode and transmit a START condition once the I2C bus is free. If the I2CSCR is already in master mode and one or more bytes have been transmitted, a repeated START condition will be transmitted. If the I2CSCR is still being accessed in slave mode, the I2CSCR will complete the data transfer in slave mode and enter master mode once the I2C bus has been released. ***Note**: The STA bit is cleared automatically after the START condition has been sent; writing 0 to this bit has no effect.* 0_B The I2C does not enter master mode. 1_B The I2C enters master mode and transmits a START condition on the I2C bus.

(table continues...)

(continued)

Field	Bits	Type	Description
ENAB	6	rw	**I2C enable** 0_B The I2CSCR is disabled. 1_B The I2CSCR is enabled.
IEN	7	rw	**Interrupt enable** 0_B The interrupt is disabled. 1_B The interrupt is enabled.
0	1:0	r	**Reserved** Read as 0; must be written with 0.

Table 1900 Reset values of I2CSCR_CNTR

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	
Kernel Reset	$--00\ 0-XX_B$	

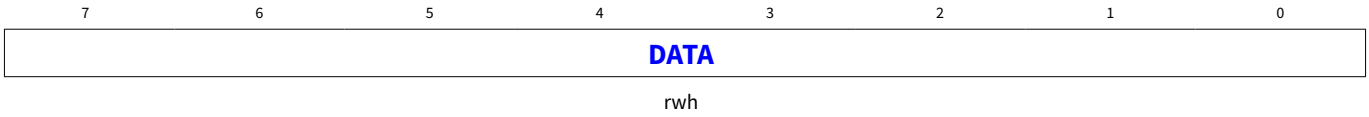
39.4.191 Data register

This register contains the data byte or slave address to be transmitted or the data byte that has just been received. Note: This register shall only be written, if CNTR.IFLG==1.

I2CSCR_DATA Offset address: $0FB_H$

Data register Reset values see: Table 1901

PAGE: X, RMAP: 0



Field	Bits	Type	Description
DATA	7:0	rwh	**Data byte** Data to be sent or received.

Table 1901 Reset values of I2CSCR_DATA

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.192 Status register

The read-only I2CSCR_STAT register stores the 5-bit status code of the I2CSCR.

I2CSCR_STAT Offset address: $0FC_H$

Status register Reset values see: Table 1902

PAGE: X, RMAP: 0

7	6	5	4	3	2	1	0
STAT					0		
r					r		

Field	Bits	Type	Description
STAT	7:3	r	**Status code** 5-bit status code (see Table in Status Code Section)
0	2:0	r	**Reserved** Read as 0; must be written with 0.

Table 1902 Reset values of I2CSCR_STAT

Reset type	Reset value	Note
LVD Reset	$F8_H$	
Generated Reset	$F8_H$	
Kernel Reset	$F8_H$	

39.4.193 Baud rate control register [write mode]

The write-only I2CSCR_BRCR register controls the sampling frequency of the I2CSCR and the baudrate of the I2CSCR in master mode.

I2CSCR_BRCR Offset address: $0FC_H$

Baud rate control register [write mode] Reset values see: Table 1903

PAGE: X, RMAP: 0

7	6	5	4	3	2	1	0
0	BRP				PREDIV		
r	w				w		

Field	Bits	Type	Description
PREDIV	2:0	w	**Pre-devider for baud rate generation** 000_B Pre-devider factor of 1 is used. 001_B Pre-devider factor of 2 is used. 010_B Pre-devider factor of 4 is used. 011_B Pre-devider factor of 8 is used. 100_B Pre-devider factor of 16 is used. 101_B Pre-devider factor of 32 is used. 110_B Pre-devider factor of 64 is used. 111_B Pre-devider factor of 128 is used.
BRP	6:3	w	**Baud rate pre-scaler** Determines the baudrate for the I2CSCR in master mode.
0	7	r	**Reserved** Read as 0; must be written with 0.

Table 1903 Reset values of I2CSCR_BRCR

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	
Kernel Reset	00_H	

39.4.194 Software reset register

The I2CSCR_SRST register provides a software reset on the I2CSCR. Moreover, this register is also used to do the selection for the SCL and SDA signals coming from the input ports.

I2CSCR_SRST Offset address: $0FD_H$

Software reset register Reset values see: Table 1905

PAGE: X, RMAP: 0

7	6	5	4	3	2	1	0
SDASEL		**SDASEL_P**	**SCLSEL**		**SCLSEL_P**	**0**	**SRST**
rw		w	rw		w	r	w

Field	Bits	Type	Description
SRST	0	w	**Software reset** 0_B No effect. 1_B Triggers a soft reset on the I2C.
SCLSEL_P	2	w	**SCLSEL write protection** Protects bitfield SCLSEL against unintended changes. 0_B SCLSEL is not changed 1_B SCLSEL is updated by this write access
SCLSEL	4:3	rw	**SCL selector** Selects one out of four SCL inputs (I2CSCLA/I2CSCLB/I2CSCLC/I2CSCLD) coming from the input ports. 00_B I2CSCLA is selected 01_B I2CSCLB is selected 10_B I2CSCLC is selected 11_B I2CSCLD is selected
SDASEL_P	5	w	**SDASEL write protection** Protects bitfield SDASEL against unintended changes. 0_B SDASEL is not changed 1_B SDASEL is updated by this write access

(table continues...)

(continued)

Field	Bits	Type	Description
SDASEL	7:6	rw	**SDA selector** Selects one out of four SDA inputs (I2CSDAA/I2CSDAB/I2CSDAC/ I2CSDAD) coming from the input ports. 00_B I2CSDAA is selected 01_B I2CSDAB is selected 10_B I2CSDAC is selected 11_B I2CSDAD is selected
0	1	r	**Reserved** Read as 0; must be written with 0.

Table 1904 Access mode restrictions of I2CSCR_SRST sorted by descending priority

Mode name	Access mode		Description
write 1 to .**SCLSEL_P** and write 1 to .**SDASEL_P**	rw	SCLSEL, SDASEL	
(default)	r	SCLSEL, SDASEL	

Table 1905 Reset values of I2CSCR_SRST

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.195 Slave address register

This register contains a bit to enable the general call address option and the device address of the I2CSCR in 7-bit addressing mode. For 10-bit addressing, part of the address bits is located in I2CSCR_ADDRX register.

I2CSCR_ADDR Offset address: $0FE_H$

Slave address register Reset values see: Table 1906

PAGE: X, RMAP: 0

7	6	5	4	3	2	1	0
SLA							**GCE**
rw							rw

Field	Bits	Type	Description
GCE	0	rw	**General call enable** 0_B General call address option is disabled 1_B General call address option is enabled

(table continues...)

(continued)

Field	Bits	Type	Description
SLA	7:1	rw	**Slave address** For 7-bit addressing, SLA contains the 7-bit address of the I2CR when in slave mode. For 10-bit addressing, the uppermost five bits SLA[6:2] are fixed at 11110_B, while SLA[1:0] contains the uppermost two bits of the 10-bit slave address. The remaining lower 8 bits are located in I2CSCR_ADDRX register.

Table 1906 Reset values of I2CSCR_ADDR

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.4.196 Extended slave address register

This register contains the lower 8-bit slave address of the device in 10-bit addressing mode.

I2CSCR_ADDRX Offset address: $0FF_H$

Extended slave address register Reset values see: Table 1907

PAGE: X, RMAP: 0

7	6	5	4	3	2	1	0
SLAX							
rw							

Field	Bits	Type	Description
SLAX	7:0	rw	**Extended slave address** For 10-bit addressing, SLAX contains the lower 8-bit address of the I2CSCR when in slave mode.

Table 1907 Reset values of I2CSCR_ADDRX

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.5 Debug information

39.5.1 Features

The main debug functionality supported are:

- Set breakpoints on instruction address and on address range within the program memory
- Set breakpoints on Internal Memory (IMEM) address range for read and write
- Support unlimited software breakpoints in Flash/RAM code region
- Process external breaks through debug interface
- Stepping through the program code
- User handling of OCDS NMI in case of IMEM read/write breakpoint

In summary, up to four hardware breakpoints can be set for the current implementation. It can be used for detection specific Instruction Pointer (IP), a range of IP and specific range of read/write of internal Memory (IMEM),
unlimited software breakpoint (only possible in writable region) and external break. When any of the above is encountered, that corresponds to a debug event which will then trigger the execution of Monitor ROM.

39.5.2 Components of the debug system

The components of the debug system are briefly described here.

39.5.2.1 Debug interface

The debug interface allows to access the device, such as for data read or write. The debug interface supports:

- Dual-pin Device Access Port (DAP)
- Single-pin DAP (SPD)

The DAP interface refers to the "Dual-pin Device Access Port" standardized for the Infineon micro-controllers. It offers high noise immunity and robustness, and requires only two pins DAP0 (clock), DAP1 (serial data in/out).

The SPD interface refers to the "Single-pin Device Access Port" standardized for the Infineon micro-controllers. It requires only one pin DAP1 for serial data in/out.

39.5.2.2 Monitor program

Part of the Boot ROM is occupied by the OCDS monitor program (or OCDS firmware). On the command from the debugger, the monitor program executes the actual instructions for accessing the addressable locations such as memories and SFRs of the device.

39.5.2.3 On-Chip Debug System unit (OCDS)

The OCDS hardware is the core of the debug system, controlling the debugging with interfaces to the XC800 CPU.

39.5.3 Product specific OCDS information

39.5.3.1 OCDS pin functions

Table 1908 OCDS pin functions

Function	Pin connection	Description
DAP0_1	SCR_P0.2 (P33.2)	Dual-pin debug access port: clock
DAP0_0	SCR_P0.6 (P33.6)	
DAP1_1/SPD_1	SCR_P0.3 (P33.3)	Single-pin debug access port: data
DAP1_0/SPD_0	SCR_P0.7 (P33.7)	

The pin will be set up through hardware once the OCDS mode is selected. A detailed pin-out is given in the GPIO chapter.

39.5.3.2 Clocking configuration

The OCDS runs at the peripheral frequency, PCLK.

39.5.3.3 JTAG ID

The following assignment of JTAG ID are used for SCR:

VERSION = 1_H, PART_NUMBER = $01EC_H$, i.e. JTAG ID Register = $101EC083_H$

39.6 References

This section is not applicable for the module being described.

39.7 SCR revision history

Reference	Description of change(s)
Date range: 2025-02-26 to 2025-03-31	
Revision history	• Updated Revision history table according latest template
Debug information	• Added section with debug information
External data memory	• Added XRAM access protection reference to PMS
Initialization of break/ synch field detection logic	• Description provided with more details
SCR functions in port SCR_P2	• Corrected generic mapping between register bit fields and pin
SCR functions in port SCR_P3	• Corrected generic mapping between register bit fields and pin
Date range: 2024-10-05 to 2025-02-25	
UART connections to GPIO	• Corrected typo in UART connections to GPIO ports figure: P35.7 on RXD0 means to be P35.5
WCAN initialization sequence	• New description added for WCAN initialization sequence

39 Stand-by Controller (SCR)

Reference	Description of change(s)
Initialization of break/ synch field detection logic	• New description added for LIN auto baud detection feature
Date range: 2024-08-17 to 2024-10-04	
Stand-by Controller (SCR)	• The SCR specification has been completely reworked and shows the TC4xx silicon changes relevant for TC4Dx

39.8 TC4Dx SCR information

39.8.1 TC4Dx SCR configuration

The TC4Dx has the base SCR configuration. All registers deviating among the devices are listed in this paragraph.

39.8.2 TC4Dx SCR features

The additional IO page does not exist in this device. The following registers are for future extension of the IO page:

The IO_PAGE=3 is not existing on this device, it includes the following registers

- Pn_WKEN, Pn_WKENSTS and Pn_WKSTS

ADCOMP_CTR registers

- The ADCOMP_CTR register described in the ADCOMP paragraph

SCUSCR_PMS1IVRCON register

- The register SCUSCR_PMS1IVRCON enables the application to switch among the power modes
- To enable the SCR to switch its power modes the bit SCUSCR_PMS1IVRCON.SINKCTRLEN has to be set to 1. When enabling the power mode control via SCR the following has to be regarded: If SCUSCR_PMS1IVRCON.SINKCTRLEN=1, it is required that SCUSCR_CLK.DIV.REQ >= 5 (max. SCR frequency is limited 20 MHz in the medium power mode), otherwise there is a risk of triggering under-voltage reset or alarms
- The SCUSCR_PMS1IVRCON.SINKCTR regulates the power consumption. A 0_B represents the lowest power saving level and therefore the highest power consumption, dependent on the wished power saving this lowest power consumption can be reached with programming a 3_B

39.8.3 TC4Dx SCR functional description

The device specific registers are listed below.

39.8.3.1 WKEN: Reserved for future extension - not accessible - page does not exist

Pn_WKEN0 (n=0-3)

Offset address: 090_H+n*8

WKEN: Reserved for future extension - not accessible - page does not exist

Reset values see: Table 1909

PAGE: IO_PAGE=3, RMAP: 0

7	6	5	4	3	2	1	0
Res							R0
r							r

Field	Bits	Type	Description
R0	0	r	**Reserved for future extension - not accessible - trap will be generated**

Table 1909 Reset values of Pn_WKEN0 (n=0-3)

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.8.3.2 WKENSTS: Reserved for future extension - not accessible - page does not exist

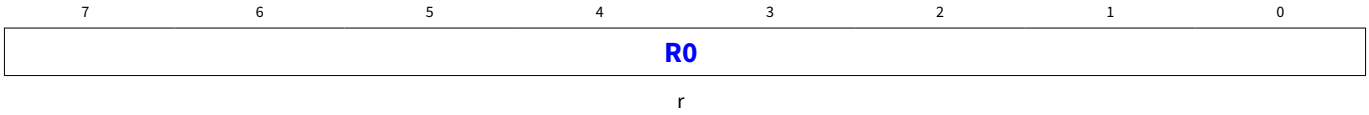
Pn_WKENSTS0 (n=0-3)

Offset address: 091_H+n*8

WKENSTS: Reserved for future extension - not accessible - page does not exist

Reset values see: Table 1910

PAGE: IO_PAGE=3, RMAP: 0



Field	Bits	Type	Description
R0	7:0	r	**Reserved for future extension - not accessible - trap will be generated**

Table 1910 Reset values of Pn_WKENSTS0 (n=0-3)

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.8.3.3 WKSTS: Reserved for future extension - not accessible - page does not exist

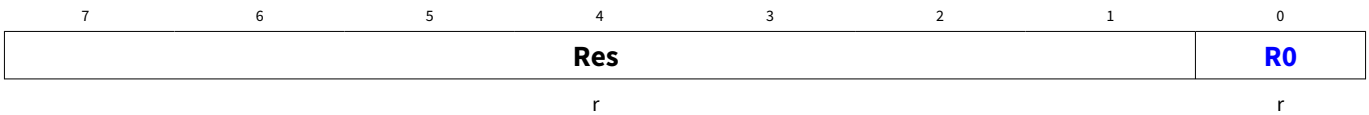
Pn_WKSTS0 (n=0-3)

Offset address: 092_H+n*8

WKSTS: Reserved for future extension - not accessible - page does not exist

Reset values see: Table 1911

PAGE: IO_PAGE=3, RMAP: 0



Field	Bits	Type	Description
R0	0	r	**Reserved for future extension**

Table 1911 Reset values of Pn_WKSTS0 (n=0-3)

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.8.3.4 ADCOMP control register

ADCOMP_CON Offset address: $0C7_H$

ADCOMP control register Reset values see: Table 1912

PAGE: X, RMAP: 0

7	6	5	4	3	2	1	0
SOC	**STSEL**		**CHSEL**				**EN_REQ**
rwh	rw		rw				rw

Field	Bits	Type	Description
EN_REQ	0	rw	**ADC comparator enable request** The ADC clock source is NOT automatically enabled upon setting of this bit. It can enabled in SCUSCR_CLK register by setting DIV_REQ to any other value than 0xF to activate the 100 MHz HPOSC clock. It is required to wait some time for the ADCOMP unit to start after the request. The activation status can be read from ADCOMP_RESHSTAT.EN_STAT. Note: The time can vary from around 15 to 65 µs, depending if system was in RUN or in STANDBY0 or in STB1_70KHz mode prior to setting EN_REQ. In any case the application can poll on ADCOMP_RESHSTAT.EN_STAT to know, if the ADCOMP is ready or not. In case EN_REQ is cleared during ongoing conversion, then the ADC is disabled only after the currently running conversion is ended. 0_B ADCOMP unit is disabled 1_B ADCOMP unit is enabled
CHSEL	4:1	rw	**ADC channel select** This bit-field selects the channel and configures the 16:1 ADC multiplexer before a conversion is triggered. During an ongoing measurement (sampling and conversion) CHSEL shall be kept unchanged. If CHSEL is changed during an ongoing conversion, the result of the conversion is invalid. 0_H **CH0**: CH0 is selected … F_H **CH15**: CH15 is selected

(table continues...)

(continued)

Field	Bits	Type	Description
STSEL	6:5	rw	**Sample time selection** Sampling time value is taken into account when SOC signal is set and latched locally inside ADC. If the bit is changed during an ongoing measurement, the new value will be taken into account at the next start of conversion. 00_B Minimal sample time is selected (80ns, default) 01_B Sample time of 100 ns is selected 10_B Sample time of 200 ns is selected 11_B Sample time of 400 ns is selected
SOC	7	rwh	**ADC start of conversion request** This bit activates a new conversion. This bit is automatically cleared by hardware once the conversion has started. If EOC is set, setting SOC clears EOC. 0_B ADC conversion is not started 1_B ADC conversion is started

Table 1912 Reset values of ADCOMP_CON

Reset type	Reset value	Note
LVD Reset	00_H	
Generated Reset	00_H	

39.8.3.5 PMS1 iVR control

SCUSCR_PMS1IVRCON Offset address: $0F0_H$

PMS1 iVR control LVD Reset value: 00_H

PAGE: SCUSCR_PAGE=7, RMAP: 0

7	6	5	4	3	2	1	0
0	**0**	**SINKSTS**			**SINKCTRLEN**	**SINKCTRL**	
r	r	rh			rw	rw	

Field	Bits	Type	Description
SINKCTRL	1:0	rw	**PMS iVR sink control value** Set power saving level in STANDBY1_100 and STANDBY1_ADCOMP. Note: The higher the level, the less power is consumed, but the more critical load jumps due to clock switch or module reconfiguration are to the regulator. This could lead to a system reset. 00_B Powersaving level 0 Lowest level of power saving 01_B Powersaving level 1 10_B Powersaving level 2 11_B Powersaving level 3 Highest level of power saving

(table continues...)

(continued)

Field	Bits	Type	Description
SINKCTRLEN	2	rw	**Enable Powermode Control from SCR side** **Note:** If SINKCTRLEN=1, it is required that SCUSCR_CLK.DIV.REQ >= 5 (max. SCR frequency is limited 20 MHz in the medium power mode), otherwise there is a risk of triggering under-voltage reset or alarms. 0_B The SCR does not control its power modes 1_B The SCR controls its power modes
SINKSTS	5:3	rh	**Sink disable status from SCR side** 000_B Power saving mode is off 001_B Powersaving level 0 is active 010_B Powersaving level 1 is active 011_B Powersaving level 2 is active 100_B Powersaving level 3 is active
0	6, 7	r	**Reserved** Read as 0; must be written with 0.

39.8.4 TC4Dx SCR connectivity

There are no deviations from the generic specification.
The connectivity of the SCR module can be found within the corresponding device datasheet.

39.8.5 TC4Dx SCR revision history

Initial release of the chapter.

40 Interrupt Router (IR)

The interrupt router receives interrupts from external resources, internal functional blocks and from application software and forwards them as service requests to the interrupt targets (CPUs, DMAs, GTM, and the PPU). These interrupt targets are referred to as interrupt service providers (ISP).

The IR converts the interrupt events from functional blocks and application software into service requests (SR) by adding configurable information, such as which interrupt target, service request priority number, and service request security information.

***Note**: Unless otherwise stated, ´CPUs´ means ´all TriCore™ CPUs´, which includes CPUx and the CPUCS.*

40.1 Feature list

- Interrupt system with support for up to 2048 service requests
- Support for up to 256 service request priority levels per Interrupt Control Unit (ICU) and Interrupt Service Provider (ISP)
- Support for up to 15 ICUs
- A dedicated ICU for each implemented ISP. The ISPs are the functional blocks CPUs, DMAs, PPU, and the GTM
- Low latency arbitration
  - Four fSPB clock cycles from interrupt trigger to service request signaling to ISP
  - Interrupt latency may increase for an ICU that operates in hardware virtualization mode
- Each interrupt with a dedicated Service Request Node (SRN)
  - Each SRN with a programmable 8-bit Service Request Priority Number (SRPN)
  - Each SRN can be mapped to one of the implemented ISP
  - Each SRN can be mapped to exactly one virtual machine
  - SRNs are cleared by hardware on an interrupt acknowledge by the configured ISP if no EDC error is detected
- Interrupt system with integrity support
  - The IR is implemented with a lockstep mechanism
  - End-to-end EDC protection of the interface between IR and the ISPs
- Software Interrupts
  - One group of 8 General Purpose Service Requests (GPSR) per CPU and for the PPU
- Ability to signal software interrupts simultaneously to multiple ISPs through service request broadcast registers (SRB)
- Ability to transfer additional 16-bit information through software interrupt control register (GPSRG_SWC.DATA) to ISPs, protected by a hardware handshake
- External interrupts with filter and trigger modes
  - External interrupt logic and related control registers are described in the System Control Unit (SCU) chapter, External Request Unit (ERU)
- The IR provides wake-up support for the CPUs and the PPU that can cause the respective function block that is currently in idle or sleep to enter run mode

40.2 Functional overview

The Interrupt router schedules interrupts, called service requests, from external resources, internal functional blocks, and from application software to the interrupt targets. The interrupt targets are the functional blocks CPUs, DMAs, GTM and the PPU. These service request targets are referred to as Interrupt service providers (ISP). The IR converts the interrupt events from functional blocks and application software into service requests by adding configurable information about which interrupt target, service request priority number, service request and security information.

The Interrupt router includes:

40 Interrupt Router (IR)

- The Service request nodes (SRNs)
- The Software interrupt functions that includes
  - General purpose service request nodes (GPSR)
  - Broadcast control registers (SRB)
  - Software control registers (SWC)
- The Interrupt control units (ICU) with its control registers
- Additional functionality for software development support like access protection and debug support

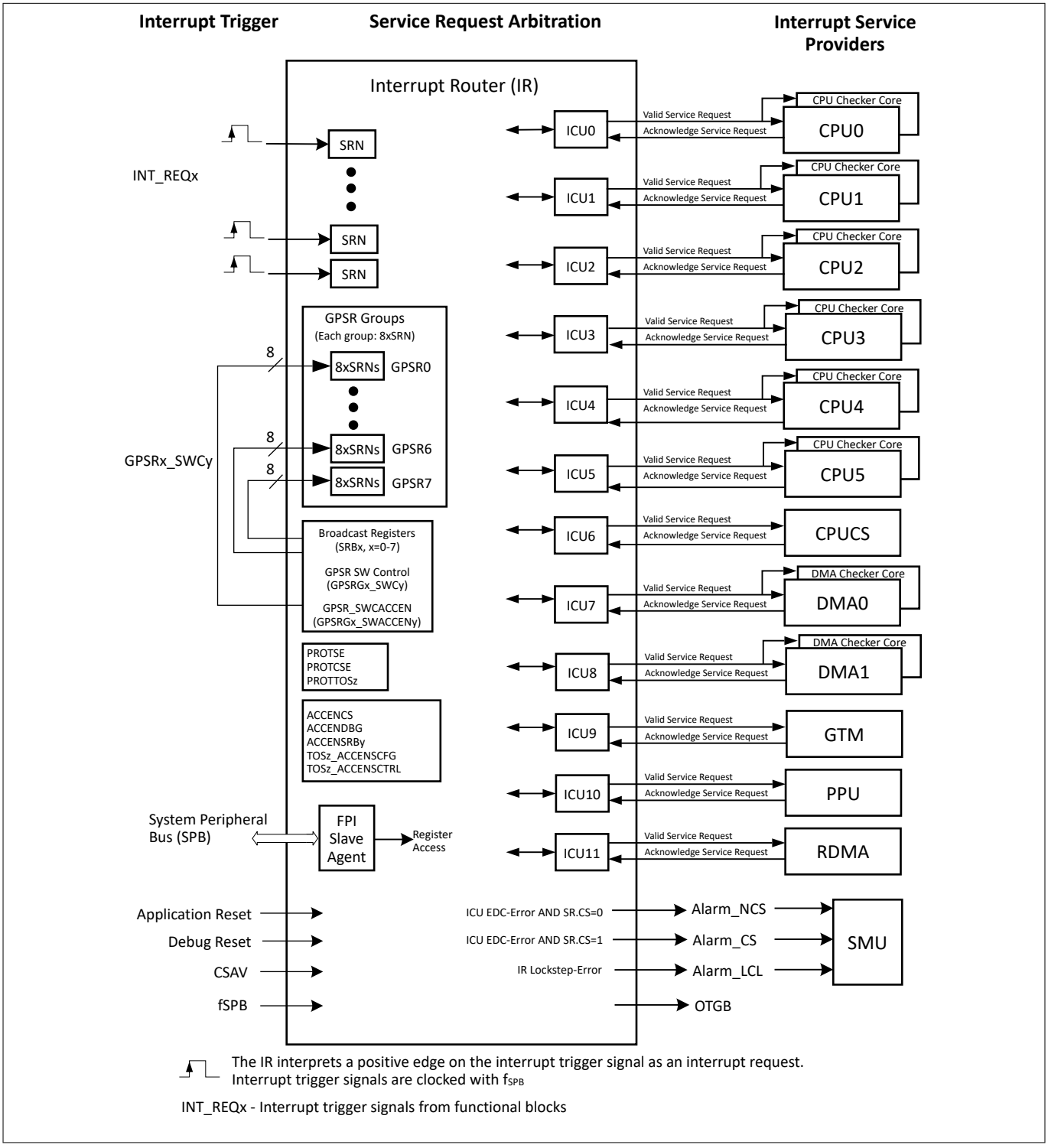


Figure 951 IR block diagram

40 Interrupt Router (IR)

For the alarm description please refer to the alarm mapping tables in the SMU functional block and the Safety Manual.

Related information

TC4Dx SMU alarm mapping tables on page 7233

40.3 Functional description

As shown in the figure ´IR block diagram´ , each functional block that can generate interrupt requests is connected to one or more Service Request Nodes (SRNs) in the Interrupt Router (IR).

Each SRN contains a Service request control register (SRC) to configure the Service request (SR) regarding priority and mapping to one of the available Interrupt service provider (ISP).

The IR module contains also several groups of General purpose service request nodes (GPSR) that can be used for software (SW) triggered service requests. There is a Broadcast register for each GPSR group which allows to trigger multiple nodes of the related GPSR group with one software access.

Additionally for each GPSR there is a GPSR control register (GPSRGx_SWC) and a GPSR access protection register (GPSRGx_SWCACCENy). These registers enable application software to map each GPSR individually to master function(s), control it from the mapped master function(s) and enable the master function(s) to signal additional information to the configured ISP.

The IR has a dedicated interface, called interrupt control unit (ICU), for each implemented physical interrupt service provider (ISP) which are CPUs, DMAs, PPU and GTM.

For each ICU the IR arbitrates among all pending service requests mapped to an ISP and passes the current arbitration round winner to the corresponding ICU.

The ICU signals the latest winning service request to the related interrupt service provider (ISP). The ISP signals back to the ICU if it has taken a service request (SR) for further processing.

The IR offers the possibility to protect the group of security related SRNs against modification by other master function and to protect the group of safety related SRNs against modification by the security function (SRC.CS, PROT_CSE, PROT_SE).

40.3.1 Service request nodes (SRN)

Each service request node (SRN) inside the IR contains a service request control register (SRC) and interface logic that connects it to the interrupt trigger source outside the IR.

40.3.1.1 Service request control registers (SRC)

The description given in this chapter characterizes the Service Request Control registers (SRC).

All SRC registers in the Interrupt Router module have the same format where each register contains the following control and status information:

- Service Request Priority Number (SRPN)
- Service Request Virtual Machine Number (VM)
- Service Request Cyber Security configuration (CS)
- Service Request destination, Interrupt Service Provider (TOS)
- Enable and disable the Service Request Node (SRE)
- Service Request Set bit and Service Request Clear bit (SETR, CLRR)
- Service request status bit (SRR)
- Interrupt Overflow bit (IOV)
- Interrupt Overflow Clear bit (IOVCLR)

Besides being activated by the associated triggering unit through hardware, each SRN can also be set or reset by software through two software-initiated service request control bits.

If a service request node is disabled with SRE=0 it can not generate a service request. Nevertheless, the SRR and IOV bits will be set in case of an interrupt and interrupt overflow event. As long SRE=0 the SRR=1 information will not be taken into account for the IR arbitration nor will it be signaled as pending service request to the SMM for the IR wake-up support for CPUs and PPU.

***Note**: Several modules have additional interrupt related control registers on module level, for example interrupt status, set clear or enable register. These module registers are described in the corresponding sections of the module chapters.*

Related information

Request set and clear bits (SETR, CLRR) on page 6861
Enable bit (SRE) on page 6861
Service request flag (SRR) on page 6862
Type-of-service (TOS) on page 6862
Service request priority number (SRPN) on page 6862
Virtual machine number (VM) on page 6863
Cyber security bit (CS) on page 6864
Interrupt trigger overflow bit (IOV) on page 6864
Interrupt trigger overflow clear bit (IOVCLR) on page 6865

40.3.1.1.1 Request set and clear bits (SETR, CLRR)

The SETR and CLRR bits allow software to set or clear the service request bit SRR.

- Writing a 1 to SETR causes bit SRR to be set to 1
- Writing a 1 to CLRR causes bit SRR to be cleared to 0
- Writing a 1 to SETR and CLRR at the same time, SRR is not changed
- Writing a 1 to SETR and a hardware clear after an ISP Acknowledge at the same time causes SRR to be set to 1
- Writing a 1 to CLRR and a hardware trigger at the same time causes SRR to be set to 1
- The value written to SETR or CLRR is not stored
- Writing a 0 to these bits has no effect
- These bits always return 0 when read

40.3.1.1.2 Enable bit (SRE)

The SRE bit enables an interrupt to take part in the arbitration for the selected ISP. It does not enable or disable the setting of the request flag SRR; the request flag can be set by hardware or by software (through SETR) independent of the state of the SRE bit. This allows service requests to be handled automatically by hardware or through software polling.

SRE = 1:

If SRE=1, pending service requests are passed on to the designated ISP for interrupt arbitration. The SRR bit is automatically set to 0 by hardware when the service request is acknowledged by the ISP. It is recommended that in this case, software should not modify the SRR bit to avoid unexpected behavior due to the hardware controlling this bit.

SRE = 0:

If SRE=0, pending service requests are not passed on to ISPs. Software can poll the SRR bit to check whether a service request is pending. To acknowledge the service request, the SRR bit must then be reset by software by writing a 1 to CLRR.

***Note**: In this document, 'active source' means an SRN whose service request control register has its request enable bit SRE set to 1 to allow its service requests to participate in interrupt arbitration.*

40.3.1.1.3 Service request flag (SRR)

When set, the SRR flag indicates that a service request is pending. It can be set or reset directly by hardware or indirectly through software using the SETR and CLRR bits. Writing directly to this bit through software has no effect.

SRR can be set or cleared either by hardware or by software regardless of the state of the enable bit SRE. However, the request is only forwarded to an interrupt service provider if the enable bit is set.

For a disabled service request node (SRE = 0)

- SRR can be set by software (SETR) and hardware
- Pending service request (SRR = 1) is excluded from interrupt arbitration since SRE = 0
- Software can poll a SRR in a disabled service request node
- Software can reset the SRR by writing a 1 to CLRR
- See also "Request Set and Clear Bits (SETR, CLRR)"

For an enabled service request node (SRE = 1):

- SRR can be set by software (SETR) and hardware
- Pending service request (SRR = 1) takes part in the interrupt arbitration of the service provider selected by the TOS bit-field
- SRR is reset by hardware on acknowledge of the service request by the Interrupt Service Provider
- See also "Request Set and Clear Bits (SETR, CLRR)"

***Note**: Clearing a pending service request flag SRR and enabling the corresponding service request node (SRN) should be done in two steps which means two writes: first clear the SRR flag (SRC.CLRR), then enable the (SRC.SRE).*

Related information

Request set and clear bits (SETR, CLRR) on page 6861

40.3.1.1.4 Type-of-service (TOS)

CPU´s, PPU, GTM and DMA´s can act as an interrupt service provider (ISP).

A service request node can be mapped through the TOS bit-field to exact one ISP.

The TOS bit-field configuration with TOS = z maps a service request to:

- The IR interrupt control unit ICUz and through this to the related ISP
- To the two access protection register sets TOSz_ACCENSCTRL and TOSz_ACCENSCFG. The IR provides a dedicated access protection register set for each TOS coding used, that is for each implemented ICU

40.3.1.1.5 Service request priority number (SRPN)

The Service request priority number (SRPN) defines the priority of a service request with respect to other sources requesting service from the same ISP, and with respect to the priority of the ISP itself.

Multiple active SRN mapped the same ISP by using the same TOS configuration:

- Can have a unique SRPN value
- Can have a non-unique SRPN to give a group of service requests the same priority
- For a group of service requests with the same SRPN and VM configuration, the sequence of execution can not be defined
- If an SRN is not active – meaning its SRE bit is 0 – no restrictions are applied to the SRPN configuration

Interrupt service provider is a TriCore™ CPU (CPUs)

Service requests are associated through the SRPN by an interrupt vector table located in each CPU. This means that the CPU interrupt vector table is ordered by priority number. This is unlike traditional interrupt CPU architectures in which their interrupt vector tables are ordered by the source of the interrupt.

***Notes**:*

1. *For a CPU, a service request should not be configured with SRPN = 0. This is because a service request with the SRPN = 0 is ignored by the CPU and will not be acknowledged and therefore, the related SRN will not be cleared. Priority 0 is the lowest service request priority and therefore, it will not hide a SR with a higher priority but it prevents arbitration from being stopped if there is no valid SR (SRPN > 0) pending for the CPU*
2. *TriCore™ CPUs are providing a flexible interrupt table alignment with a configurable vector spacing of 8 byte or 32 byte. Please see also CPU chapter*

Interrupt service provider is a DMA

Service requests are associated through the SRPN to DMA channel numbers:

- SRPN = x will trigger DMA channel x, if DMA channel x is implemented

Only the SRPN numbers from 0 up to the max_channel_number will result in a trigger of the related DMA channel.

SRPN numbers > max_channel number will be ignored by the DMA. This means that the DMA will acknowledge the service request but it will not trigger a DMA channel nor is there any notification signaled to the application.

Examples

- In case of a 128 channel DMA, a service request with SRPN 00_H will trigger DMA channel 0, SRPN number $7F_H$ will trigger the channel 127. Service requests with SRPN > $7F_H$ will be acknowledged by the DMA and otherwise ignored
- In case of a 64 channel DMA, a service request with SRPN 00_H will trigger DMA channel 0, SRPN 17_H will trigger channel 23 and SRPN number $3E_H$ will trigger channel 62. Service requests with SRPN > $3F_H$ will be acknowledged by the DMA and otherwise ignored

Interrupt service provider is GTM

The mechanism how the GTM module maps received service requests to the GTM internal MCS functions is described in the GTM chapter.

Interrupt service provider is PPU

The mechanism how the PPU module handles received service requests is described in the PPU chapter.

40.3.1.1.6 Virtual machine number (VM)

Each TriCore™ CPU in hardware virtualization mode represents up to 8 Virtual machines (VMs) where each VM represents a virtual interrupt service providers (virtual ISP´s).

A SRN can be mapped to a physical ISP through SRC.TOS configuration. For a CPU in hardware virtualization mode, an SRN can be mapped to one of the VM through SRC.VM.

- SRC.VM = 000 -> Service request is mapped to CPU virtual machine 0 (Hypervisor)
- SRC.VM = 001 -> Service request is mapped to CPU virtual machine 1
- SRC.VM = 010 -> Service request is mapped to CPU virtual machine 2
- …
- SRC.VM = 111 -> Service request is mapped to CPU virtual machine 7

If a CPU in hardware virtualization support mode receives a service request for a VM that is not enabled on the CPU, CPU reacts to the Service Request and the Hypervisor or System software is expected to signal an ALARM to the SMU. For more details, see chapter CPU, "Interrupt corner cases".

Each ICUz provides a VMEN register, VMENz:

- The VMENz register is implemented hardwired to 0000_0000 if either a CPU without hardware virtualization support or any other type of ISP (DMA, PPU, GTM) is connected to ICUz
- If a CPUz with hardware virtualization support is connected to ICUz, then VMENz[7:0] bits are implemented as register bits. The bits VMENz[7:0] can be used by software to configure which VMs are used on the corresponding CPUz. VMENz[0] corresponds to VM0, VMENz[1] corresponds to VM1, VMENz[7] corresponds to VM7

For a CPUz with hardware virtualization support:

- For all VMs that need to be used on CPUz, the corresponding bits in VMENz[7:0] must be set to 1_B. The ICU then arbitrates separately for each VM enabled through the VMENz[7:0] bits, ensuring that the CPU sees always the pending interrupt with the highest priority for each used VM
- Interrupts mapped to a non-enabled VM with SRC.VM are ignored during arbitration
- If the CPU is used without virtualization by the software, VMENz[7:0] must be set to 00_H0. With this the ICUz will ignore the SRC.VM information during arbitration so the CPU sees always the pending interrupt with the highest priority for the CPU

VM configuration restrictions:

If a CPU is used in hardware virtualization mode, the hardware virtualization mode must be activated in the corresponding ICU through the register VMEN. VMEN[7:0] defines for which of the virtual machines the ICU will arbitrate.

VM configuration restriction for an ISP in Hardware virtualization support mode (VMEN is not 0, at least one VMEN bit is set to 1):

- SRNs shall only be mapped using SRC.VM to the VMs that are enabled in the respective VMEN register. Background: a pending SR to a not enabled VM will result in a permanent running arbitration for this ICU because it will never be arbitrated by the ICU
- SRNs shall not be mapped to a VM that is not enabled in the respective CPU. Background: A CPU with hardware virtualization support enabled that receives an SR for an unused VM will only acknowledge it if it has been configured by the application SW, e.g., by a hypervisor that has installed dummy VMs that respond to it and reports an ALARM to the SMU

VM configuration restriction for an ISP not in hardware virtualization support mode, VMEN=00_H:

- The IR will ignore the SRCx.VM configuration for the arbitration. All service requests for this ISP will be signaled by the ICU with VM=0_H to it

40.3.1.1.7 Cyber security bit (CS)

The SRCx.CS bit can be used by the application software to assign an SRCx to the security or to the non-security application. For further information please see chapter Access protection of the SRCx register (SRCx[15:0], SRCx[31:16]).

Related information

Access protection of the SRCx register (SRCx[15:0], SRCx[31:16]) on page 6884

40.3.1.1.8 Interrupt trigger overflow bit (IOV)

The IOV bit is set by hardware if both conditions are true:

- Service request is pending
- A new service request is triggered through hardware interrupt trigger or SETR bit

40.3.1.1.9 Interrupt trigger overflow clear bit (IOVCLR)

The IOVCLR bit can be used to clear the IOV bit. The IOV bit is cleared by writing a ´1´ to the IOVCLR bit.

40.3.2 Changing an SRN configuration

All service request nodes (SRN) are disabled per default. To use a SRN, it has to be configured and enabled by setting the SRCx.SRE bit=´1´.

The service request nodes can be configured regarding the interrupt service provider target (SRCx.TOS) the virtual machine target (SRCx.VM) regarding the service request priority number (SRCx.SRPN) and if it is assigned the security application (SRCx.CS).

Once an SRN is enabled, the TOS, VM and SRPN bit fields can be changed by using the following sequence.

SRCx reconfiguration: assumptions for the enabled SRCx configuration

- SRCx with IDX = x
- SRCx.TOS = z
- SRCx.VM = y

  ***Note**: For VMENz = 0, y = 0*

SRCx reconfiguration: sequence to reconfigure SRCx

- Disable SRCx by setting SRCx.SRE = 0
- Check if SRCx.SRR = 1
- IF SRCx.SRR = 1, check if a service request may have been pending when the SRCx was switched off
  - IF SRCx.TOS = PPU, check in the PPU interrupt interface logic if this service request is pending. IF true, acknowledge it in the PPU interrupt interface logic
  - END IF
-
  - LOOP1 (check ICUz_LASR until ENTER = 0)
    - IF ICUz_LASR.ENTER = 0, EXIT LOOP1
  - END LOOP1
  - LOOP2 (check ICUz_LWSRy until either STAT = 0 or LWSRy.IDX != x)
    - IF ICUz_LWSRy.STAT=0 OR ICUz_LWSRy.IDX != 0 -> EXIT LOOP2
  - END LOOP2
- END IF
- Re-configure SRCx
- Enable re-configured SRCx

40.3.3 Changing the VMEN configuration

Assumptions

- ICUz is implemented with hardware virtualization support
- ICUz is connected to CPUz

Enabling a VMx

It is possible to activate a VMx through VMENz at any time. Before activating the VMx in the VMENz you should activate VMx in the corresponding CPU.

Disabling a VMx

Before disabling a VMx in a CPU, software shall first disable the VMx in the ICUz by setting VMENz[x]=0. This prevents that further SR are signaled to VMx. Software shall check LWSRx.STAT until LWSRx.STAT=0. Then software can disable VMx in the CPU.

Having disabled a VM through VMENz, any enabled SRN's that are mapped to that VM should be disabled, because if they are triggered they will force the ICU to arbitrate even if no other service requests are available. This is true as long VMEN[7:0] is not 0x00.

Enabling hardware virtualization support (VMENx!=0)

When changing VMENx[7:0]=00_H to VMENx[7:0]!=00_H, which means that the hardware virtualization support is enabled, the application has to ensure that no service request for ICUx is pending.

Disabling hardware virtualization support (VMEN=0)

When changing VMENx[7:0]!=00_H to VMENx[7:0]=00_H, which means that the hardware virtualization support is disabled, the application has to ensure that no service request for ICUx is pending.

40.3.4 Mapping of interrupt request triggers signals to SRNs

All functional block interrupt trigger signals are mapped to Service Request Nodes (SRN) in the Interrupt Router.

There is one dedicated Service request node (SRN) for each module interrupt request.

Each SRN has one unique SRN index number (IDX) within the IR. The IDX numbers are not required for the functionality of the interrupt router itself. The IDX numbers can be used to select a SRN for observation through the OTGB feature.

The IDX number of a service request control (SRC) register can be directly calculated out of its address offset in the SRC address range.

The IR has one interrupt trigger input vector. Each interrupt trigger input vector bit [x] is related to one SRN with its unique IDX number x[55]. This means that a trigger pulse on interrupt trigger input vector bit [x] will trigger the SRN [x].

***Note**: A positive edge on a interrupt trigger signal is interpreted as interrupt trigger. See also Timing characteristics of service request trigger signals.*

[55] The Interrupt Router register overview table shows for all module service requests the SRC registers with its address offset and the related SRN Index number

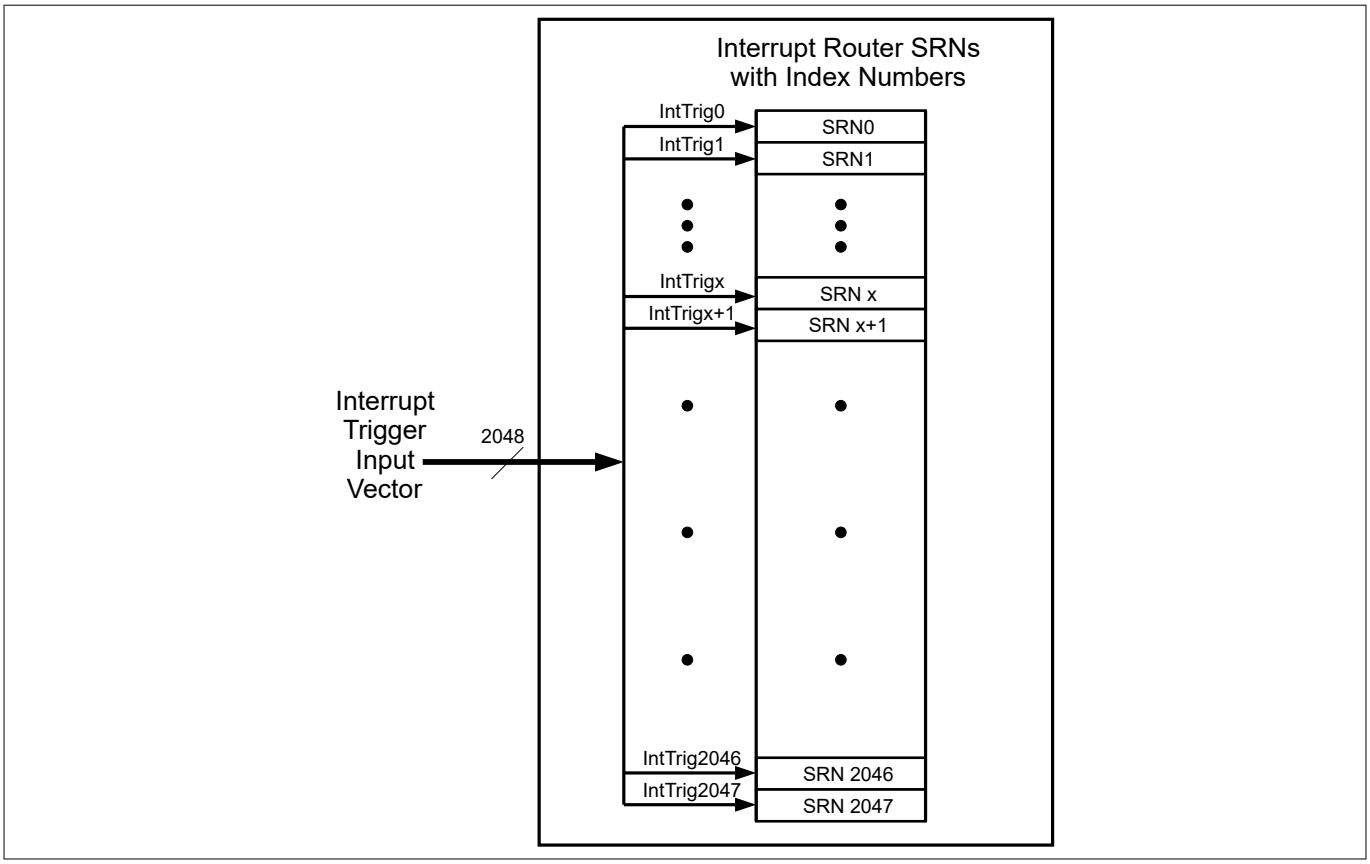


Figure 952 Mapping of interrupt trigger signals to SRN index numbers

Related information

SRC index number (IDX) on page 6868
Interrupt trigger signals - default value, trigger and timing characteristics on page 6869

40.3.4.1 SRC index number (IDX)

Each SRN can be configured and controlled through its dedicated service request control register (SRC). The implemented SRC registers are described in the product specific IR SRC register overview table.

Each SRC has a unique IDX number inside the IR.

The IDX number of a SRC can be directly calculated out if its address offset in the SRC address range:

- $Index\left(SRC\right) = \frac{< SRC\ Adress\ Offset >}{4}$

Example:

- An SRC register is shown in the product specific SRC register overview table with the offset $1E0_H$
- The index of this SRC register is than calculated: $\frac{1E0_H}{4} = 78H$. This means the index number is 78_H and 120_D

Related information

Module implementation on page 6896

40.3.4.2 Interrupt trigger signals - default value, trigger and timing characteristics

The interrupt trigger signals from the functional blocks to the IR, the IR itself and the interfaces from the IR to the Interrupt Service are clocked with the System Peripheral Bus clock (f_{SPB}). The following rules apply to the modules interrupt trigger signals from the modules to the Interrupt router:

- Trigger signals must be synchronous to the SPB interconnect clock (f_{SPB})
- The IR interrupt trigger inputs are edge sensitive, an interrupt trigger signal change from 0 → 1 is interpreted as interrupt event
- The minimal high and low pulse width of interrupt trigger signals is 1 f_{SPB} clock cycle, high pulse length can be > 1 f_{SPB} clock cycle

Default value of interrupt trigger input signals

The IR expects the interrupt trigger signals during and after reset with a low signal value (0).

A high signal value (1) during and after reset at an interrupt input is interpreted by the IR as a first interrupt trigger event, the SRR bit in the corresponding SRC is set to 1_B'.

***Note**: The SRC.SRR and SRC.IOV bits are set independent of the SRC.SRE (enable) bit setting. This to enable a customer to detect missed interrupts during a potential SRC re-configuration.*

40.3.5 Interrupt control unit (ICU)

The IR contains one ICU for each implemented physical ISP (CPUs, DMAs, PPU, GTM). Each ICU is assigned to one ISP. Each SRN can be mapped to exactly one ISP and its related ICU through SRCx.TOS configuration.

The ICU:

- ICU arbitrates among the pending SRs that are mapped to the ICU and to the related ISP, VM
- ICU signals the SR information of the arbitration round winner to the ISP by using the VALID signals
- ICU signals in special situations that an already transferred SR is no longer pending by using the INVALID signal
- On ACKNOWLEDGE of a SR by the ISP
  - ICU receives information of the acknowledged SR from the ISP
  - ICU does an ECC check on the received Service request information
  - ICU signals detected EDC errors to the SMU
  - ICU clears the SRN acknowledged by the ISP if no EDC error is detected. This means SRC.SRR is set to 0

***Note**: The ECC code is only used for error detection. Detected errors are reported to the SMU but not corrected.*

***Note**: An ICU where the Hardware virtualization support (HVS) is disabled or not available (in both cases: VMEN=00_H) always signals the service request to the ISP with VM=0, regardless of the SRC.VM configuration.*

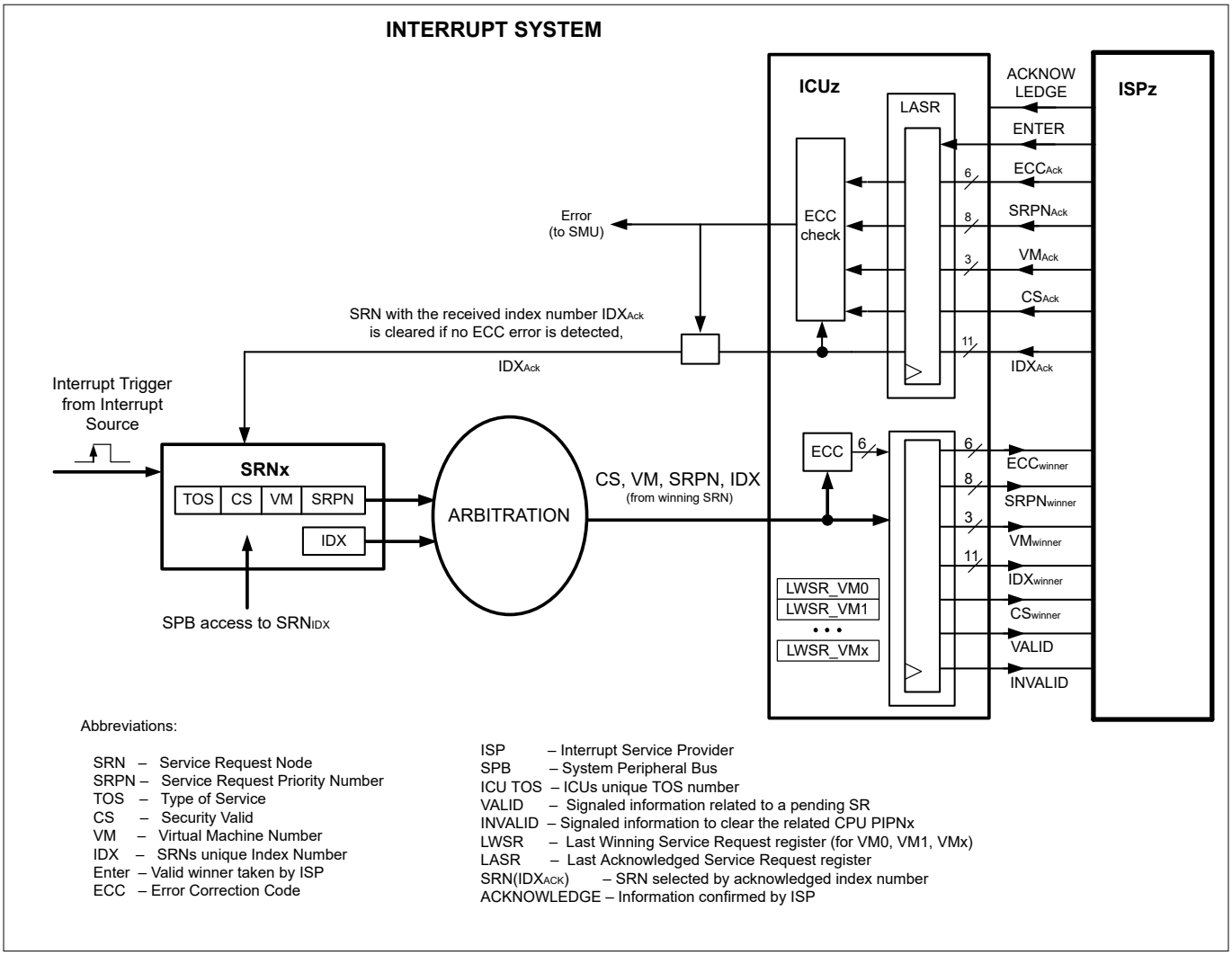


Figure 953 Interface to an Interrupt service provider

40.3.5.1 ICU control registers

This section describes the Interrupt Control Unit (ICU) registers. Each ICUz includes a dedicated set of control registers:

- Last Winning Service Request Register (LWSRz_VMy)
- Last Acknowledged Service Request Register (LASRz)
- Virtual Machine Control Register (VMENz)
- Error Capture Register (ECRz)
- Error Control Register (ECTRLz)
- Error Generation Register (EGENz)

***Note**: An ICUz for a CPU with Hardware Virtualization support has 8 LWSRz_VMy (y=0-7) and a VMENz with 8 readable and writable bits. An ICUx for a ISP without Hardware Virtualization support has only one LWSR register (LWSRz_VM0) and a read only VMENz register that is hardwired to 0.*

40.3.5.2 ICU interface

This chapter describes the EDC protected ICU interface to an ISP in general but also the differences for the different ISP types as well as a detailed description of the EDC protection and the EDC error injection.

40.3.5.2.1 VALID, INVALID and ACKNOWLEDGE

The interface between ICU and ISP are controlled through four control signals

- VALID: ICU signals with the assertion of the VALID signal that a pending service request is available at the ICU interface (winner of the last arbitration round). An ICU where the hardware virtualization support (HVS) is disabled or not available (in both cases: VMEN = 00_H) always signals the service request to the ISP with VM = 0, regardless of the SRC.VM configuration
- INVALID: ICUy signals with the assertion of the INVALID signal to a CPU that there was a pending SR visible in a used LWSRz_VMy register and now no SR is pending anymore because the last SR was cleared by software. The INVALID is only used for CPUs to clear the related PIPNx register in the CPU (set priority = 0). INVALID is not acknowledged
- ENTER: ISP CPU asserts the ENTER signal when it starts to prepare itself for the execution of a received Service Request . ICU stops re-arbitration as long the ENTER signal is asserted. See also ICU Abort condition
- ACKNOWLEDGE: ISP signals with a assertion of the ACKNOWLEDGE signal that it has taken over the service request information and will start with the execution of the related ISR. With the ACKNOWLEDGE signal de-assertion, the ISP de-asserts also the ENTER signal. See also ICU SRN clear condition

***Note**: While an ISP DMA , PPU and GTM has a fixed timing between Enter and Acknowledge, it can take many cycles for an CPU ISP before an Entered Service Request is Acknowledged or Enter is aborted (see ICU Abort condition below).*

40.3.5.2.2 Service request VALID

The ICU signals information about the pending Service Request that won the arbitration round to the ISP by asserting the Valid signal. Re-arbitration will be done until the ISP asserts the Enter signal.

The ICU signals with each VALID=1 and INVALID=0 cycle the winner service request of a new arbitration round to the ISP.

40.3.5.2.3 Service request ENTER

By setting the Enter signal, the Interrupt Service Provider signals to the ICU that it is preparing to start and execute the corresponding interrupt service routine. The Enter signal can be reset by the ISP

- With acknowledge because the Interrupt Service Routine has started
- Without acknowledge because the CPU got a trap or an exception before it was able to start the Interrupt Service Routine

40.3.5.2.4 Service request INVALID

The INVALID signal has only an effect on an ISP of type CPU.

With the INVALID signal an already sent SR information is neutralized in the CPU if required. This will be done in two situations:

- the only pending service request for a VMy that was already sent to the CPU was cleared by software in the IR
- the only pending service request for a CPU without or with disabled hardware virtualization support that was already sent to the CPU was cleared by software in the IR

The service request INVALID mechanism ensures that the CPU will not acknowledge the already cleared SR later as this may affect the already re-configured and newly triggered SRN. The mechanism is not used if more than one SR is pending for a VM or CPU because a SR that is already sent to the CPU and cleared by application software before acknowledged is replaced by the SR that wins the next IR arbitration round for the VM or CPU.

If the ICU has sent a SR through a VALID pulse to VMx in the CPU, as also captured in the IR LWSRz_VMy register, but the new arbitration round for VMx shows that there is no pending SR for VMx anymore although the previously sent SR for VMx was not ACKNOWLEDGE , the ICU sends an INVALID pulse for this VMx to the CPU. The INVALID is sent out with the respective VMx value to the CPU to enable it to identify the VMx. The corresponding CPU.PIPNx register will then be cleared by the CPU and set to priority 0.

An INVALID is not answered by the ISP with ACKNOWLEDGE and therefore not ECC checked in the ICU.

When the ICU signals an INVALID pulse for VMy to the ISP it clears in parallel the LWSRz_VMy.STATUS and sets the LWSRz_VMy.INVALID bit.

40.3.5.2.5 Service request ACKNOWLEDGE

An ISP signals an acknowledge to the ICU together with the information about the service request that is acknowledged (SRPN, VM, ECC, CS, SRN Index Number). The ISPs (CPU, DMA, GTM, PPU) have different definitions when they signal an SR acknowledge. Further information can be found in the chapter DMA to ICU interface, GTM to ICU interface and PPU to ICU interface.

The IR clears the acknowledged SRN if no ECC failure is detected and takes care that it is not signaled again unless it is newly triggered by hardware or software.

The ICU clears the LWSRz_VMy.STATUS bit if the acknowledged SR is for VMy and the SR IDX and SRPN information matches the LWSRz_VMx.VM and LWSRz_VMx.IDX information. This comparison is done because the acknowledged SR for VMx does not have to be the last SR that was signaled for VMx to the ISP which is now in the LWSRx register. This may be due to the latency of VALID pulse -> ENTER=1 or to the fact that a CPU in hardware virtualization mode itself decides in which order SR will be processed for the different VMs.

***Note**: On detection of an ECC failure the SRN is not cleared.*

40.3.5.2.6 ICU SRN clear condition

- ISP acknowledges a service request by signaling the service request information (SRPN, VM, CS, IDX) with an ACKNOWLEGE cycle to the ICU
- ICU performs an ECC check of the service request information received with the ACKNOWLEDGE
- If no ECC failure is detected on the received service request information the ICU clears the SRN with the received index number (IDX, part of the service request information)

  ***Note**: On detection of an ECC failure the SRN is not cleared.*

40.3.5.2.7 ICU abort condition

ICU abort condition

- Definition of the abort condition: Interrupt Service Provider de-asserts the ENTER signal without signaling ACKNOWLEDGE
- The abort condition does not clear the pending service request
- The abort condition enables the ICU to start re-arbitration again
- A CPU as ISP can generate the abort condition. DMAs, PPU and GTM as ISP are not able to generate the abort condition

Abort conditions generated by CPU as ISP

The CPU can generate the abort condition in the following situations:

- If the software task that is currently executed by the CPU raises the CPU priority threshold to a value greater than the entered interrupt. In this situation the abort condition enables the ICU to re-arbitrate
- If the CPU gets into the situation by a trap or a context switch so it cannot acknowledge the service request it de-asserts ENTER signal to allow a re arbitration in the IR
- In situation when an service request can not be taken, for example when a context operation is ongoing. After the previous context is restored then the CPU could be in a situation that ICR.IE bit is disabled or ICU.PIPN is less than the ICR.CCPN. In those cases, the ENTER signal is set from 1 to 0 without ACKNOWLEDGE because the conditions to take an interrupt are not met anymore
- If no service request can be taken by the CPU when there is an instruction in the pipeline updating ICR (BISR, DISABLE, RESTORE,….), the ENTER signal can be asserted and at the end of the execution of those instruction the CPU can be in a situation where the conditions for the CPU to take an service request are not met anymore. In that case, the ENTER signal will be de-asserted as well

40.3.5.2.8 ICU interface default values

The service request information signaled from ICU to ISP with VALID = 1_B and signaled from ISP back to ICU with an ACKNOWLEDGE = $1_B$are ECC protected.

In special situations the ICU signals information with INVALID = 1_B to the ISP. This is not signaled back from ISP to ICU therefore is not ECC checked.

To add robustness to ICU - ISP interface the ICU and ISP outputs are set to default values while no information is signaled.

As long the ICU does not signal information to the connected ISP (VALID = 0_B AND INVALID = 0_B), the ICU output signals are set to the following default values:

- IDX = 000_H
- PIPN and SRPN = 00_H
- VM = 0_H
- CS = 0_B
- ECC = 00_H

As long the ISP does not acknowledge a service request to the related ICU (ACKNOWLEDGE = 0_B), the ISP output signals are set to the following default values:

- IDX = 000_H
- PIPN and SRPN = 00_H
- VM = 0_H
- CS=0_B
- ECC = 00_H

This means:

In case of a stuck-at-one error on the VALID signal, a CPU will ignore the default information (SRPN= 00_H) while a DMA will trigger its DMA Channel 0 and ACKNOWLEDGE the default information to the ICU where it is detected by the ECC protection and an Alarm_NCS or Alarm_CS is signaled to the SMU.

In case of a stuck-at-one error on the ACKNOWLEDGE signal, the ICU will detect an ECC error. On detection of the EDC error an Alarm is signaled and no SRN is cleared.

In case of stuck-at-one-error on the ENTER, the IR will stop the arbitration for the respective ICU.

40.3.5.2.9 ICU to DMA interface

The interrupt router has dedicated Interrupt control unit (ICU) for each DMA (DMA0, DMA1 and RDMA).

This chapter gives an overview of how a DMA processes service requests and acknowledges them back to the ICU. A detailed description can be found in the DMA chapter.

A DMA takes over the service request information from the ICU, triggers the related DMA channel and signals the service request immediately with ACKNOWLEDGE and ENTER to the ICU where it is ECC checked and, if no error is detected, the related SRN is cleared.

The DMA interface consists of a register set where it takes over the information of a service request provided by the ICU (SRPN, CS, VM, SRN Index, ECC). The DMA sends the service request information it received with VALID=1 back to the ICU in the next clock cycle with an ACKNOWLEDGE (see also chapter Service request priority number (SRPN)).

The DMA ignores the INVALID signal.

The CS information signaled to the DMA will be used within the DMA to ensure that a:

- CS related service request (CS = 1) is mapped to a DMA channel on a CSRM related DMA resource partition (see DMA, register MODEr.CS)
- Not-CSRM related service request (CS = 0) is mapped to a DMA channel on a not-CS related DMA resource partition (see DMA, register MODEr.CS)

The DMA channel priority scheme is identical to the SRPN priority scheme:

- Lowest priority within the DMA: channel 0
- Lowest priority within the Interrupt Router: SRPN = 0

Related information

Service request priority number (SRPN) on page 6862
Access protection of the SRCx register (SRCx[15:0], SRCx[31:16]) on page 6884

40.3.5.2.10 ICU to GTM interface

The interrupt router has a dedicated Interrupt control unit (ICU) for the GTM.

This chapter gives an overview of how the GTM processes service requests and acknowledges them back to the ICU. A detailed description can be found in the GTM chapter.

The GTM takes over the service request information from the ICU, sets the GTM internal MSC interrupt trigger bit, and signals the service request immediately with ACKNOWLEDGE and ENTER to the ICU where it is ECC checked. If no ECC error is detected, the related SRN is cleared.

The GTM interface consists of a register set where it takes over the information of a service request provided by the ICU (SRPN, CS, VM, SRN Index, ECC, VALID, INVALID). The GTM sends the service request information it received with VALID=1 back to the ICU with an ACKNOWLEDGE and ENTER. Further details of the GTM ICU interface and the GTM handling of interrupts can be found in the GTM chapter.

The GTM ignores the INVALID signal.

Related information

Service request priority number (SRPN) on page 6862

40.3.5.2.11 ICU to PPU interface

The interrupt router has dedicated Interrupt control unit (ICU) for the PPU.

This chapter gives an overview of how the PPU processes service requests and acknowledges them back to the ICU. A detailed description can be found in the PPU chapter.

The PPU takes the service request information from the ICU. The PPU has control registers with which application software can trigger that the service request information is signaled with an ACKNOWLEDGE and ENTER to the ICU where it is ECC checked. If no ECC error is detected, the related SRN is cleared. The service request which is currently pending in the PPU for processing and waiting for software acknowledge can a be read there.

The PPU interface consists of a register set where it takes over the information of a service request provided by the ICU (SRPN, CS, VM, SRN Index, ECC). The PPU sends the service request information it received with VALID=1 back to the ICU with an ACKNOWLEDGE and ENTER. Further details of the PPU ICU interface and the PPU handling of interrupts can be found in the PPU chapter.

The PPU ignores the INVALID signal.

Related information

Service request priority number (SRPN) on page 6862

40.3.5.3 ECC protection of the ICU interface

The interface between ICU and the interrupt service provider is ECC protected by the ICU.

The ICU uses an ECC encoding of the SR information which is only used for error detection.

This means:

- The ICU generates an ECC value for the information signaled to the ISP
- The ICU signals the information and the ECC value with VALID and INVALID to the ISP
- On ACKNOWLEDGE by the ISP, the ICU generates an ECC value for the received information
- The ICU compares the ECC value for the received information with the received ECC value
- On detection of a ECC mismatch which means a received ECC and the ECC value generated for the received information does not match, an Alarm_NCS or Alarm_CS is signaled to the SMU and the received SR information is captured in the ECR register
- ICU signals Alarm_NCS to SMU_SAFE if received SR information shows CS = 0 and an Alarm_CS to the SMU_CS if CS = 1.

ECC protection covers the following information:

- SRPN[7:0], Service request priority number
- VM[2:0], SR Virtual machine information
- IDX[10:0], SR Index number of the related SRN
- CS, SR Cyber security information

***Note**: ICU signals information to the ISP with either VALID = 1 or INVALID = 1. The information signaled through INVALID = 1 will not be acknowledged by the CPU and therefore not ECC checked in the ICU.*

ECC code

The ECC code used for the interrupt router error detection mechanism is a 24_6 code with DED (double error detection) capability. Data being ECC´d is {1´b0, IDX, VM, CS, SRPN} = 24 bits. The ECC codes are:

```
var h5 : uint(bits:24) = 24'b000000000000111111111111;
```

```
var h4 : uint(bits:24) = 24'b000011111110000000111111;
```

```
var h3 : uint(bits:24) = 24'b011100011110001111000111;
```

```
var h2 : uint(bits:24) = 24'b101101100110110011001001;
```

```
var h1 : uint(bits:24) = 24'b110110101011010101010010;
```

```
var h0 : uint(bits:24) = 24'b111011010011101001100011;
```

40.3.5.3.1 Handling of detected ECC errors

The ICU does an ECC check of the SR information it receives with the ACKNOWLEDGE from the ISP. The ECC information that is signaled by the ICU to the ISP is covering the service request information: SRPN, VM, CS and IDX.

Whenever the ICU receives an ACKNOWLEGE from the ISP it does an ECC check. The check is done on the information received with the acknowledge from the ISP (SRPN, VM, CS, Index, ECC).

On detection of an ECC error in ICUz:

- ICUz captures the related service request information in the Error capture registers (ECTRLz), sets the ECTRLz.STAT bit and the ECTRLz.EOV bit if the ECTRLz.STAT bit is still set
- IR signals an Alarm to the Safety and Security Error Management Unit (SMU). The SMU forwards this information to a CPU (if enabled)
- ICUx does not clear the service request in the SRN (selected by the index)
- The application software can identify the ICU where an ECC error was detected through the Error status bits (ECRx.STAT=1). It can read out the related service request information and clear the ECRx.STAT bit by writing with 1. The application software can also identify if one or multiple ECC errors where detected by this ICU through the Error overflow bit (ECRx.EOV=1)

40.3.5.3.2 Default ECC output values

The path from the ICU to an ISP and back to the ICU is also protected if no service request is signaled:

- As long as an ICU does not intend to signal service request information with valid or invalid to the ISP, it has invalid ECC values in the ICU output registers
- As long as an ISP does not intend signal service request information with an acknowledge to the ICU, it has invalid ECC values in the ISP output registers

See also related information in the chapter ´ICU interface default values´.

Related information

ICU interface default values on page 6874

40.3.5.3.3 ECC error injection, alarm test

The ECC error detection mechanism and the related alarm signals can be checked by the application software for each ICUz separately through the EGENz register:

- Select an SRNx which is enabled and configured with SRCx.TOS = z (means: SNRx is mapped to ICUz)
- Write the Index of the SRNx in EGENz.ID and set EGENz.AE = 1
- Check the ECTRLz.STAT and ECTRLz.EOV bits, clear the bits if necessary to ensure that the injected error can be captured by the ECRz register and possible second error can be detected through the ECTRLz.EOV bit
- Inject one SR in SRNx (through SRNx related hardware interrupt trigger condition or through writing to SRCx.SETR)
- On detection of the related IR Alarm_NCS or Alarm_CS signal, check the ECRz register to ensure that the SMU alarm is related to the injected error and not a real error from a different SR acknowledged to ICUz
- On detection of the related IR Alarm_NCS and Alarm_CS signal, check the ECTRLz.EOV bit is not set
- Disable the EGENz mechanism by setting EGENz.AE = 0

***Note**: A CS related (CS_VALID=´1´) write to an EGENz register is only executed if it writes with CS = 1. A non CS related (CS_VALID = 0) write to an EGENz register is only executed if it writes with CS = 0.*

***Note**: An error is only injected if the CS bit of selected service request is equal to the EGEN CS bit (for EGENz.ID = x, SRCx.CS = EGENz.CS).*

40.3.6 Software interrupts (GPSR)

The IR provides multiple groups of general purpose service requests (GPSRGx) and a mechanism to trigger multiple Service Requests of a GPSRGx group in parallel, by software.

General purpose service requests are intended for software interrupts. These SRNs are not connected to internal or external hardware trigger signals and can only be used as software interrupts also called software initiated service requests.

***Note**: Any Service request node can be used by software for software interrupts by writing to the service request bit (SRCx.SETR) where it has write access rights. After the SRCx.SRR bit is set in an SRN, there is no way to distinguish between a software initiated service request and a hardware initiated service request. Therefore: If software uses an SRN for software interrupts that can also be initiated by hardware, it should use SRNs that are not used by the application for hardware-initiated service requests.*

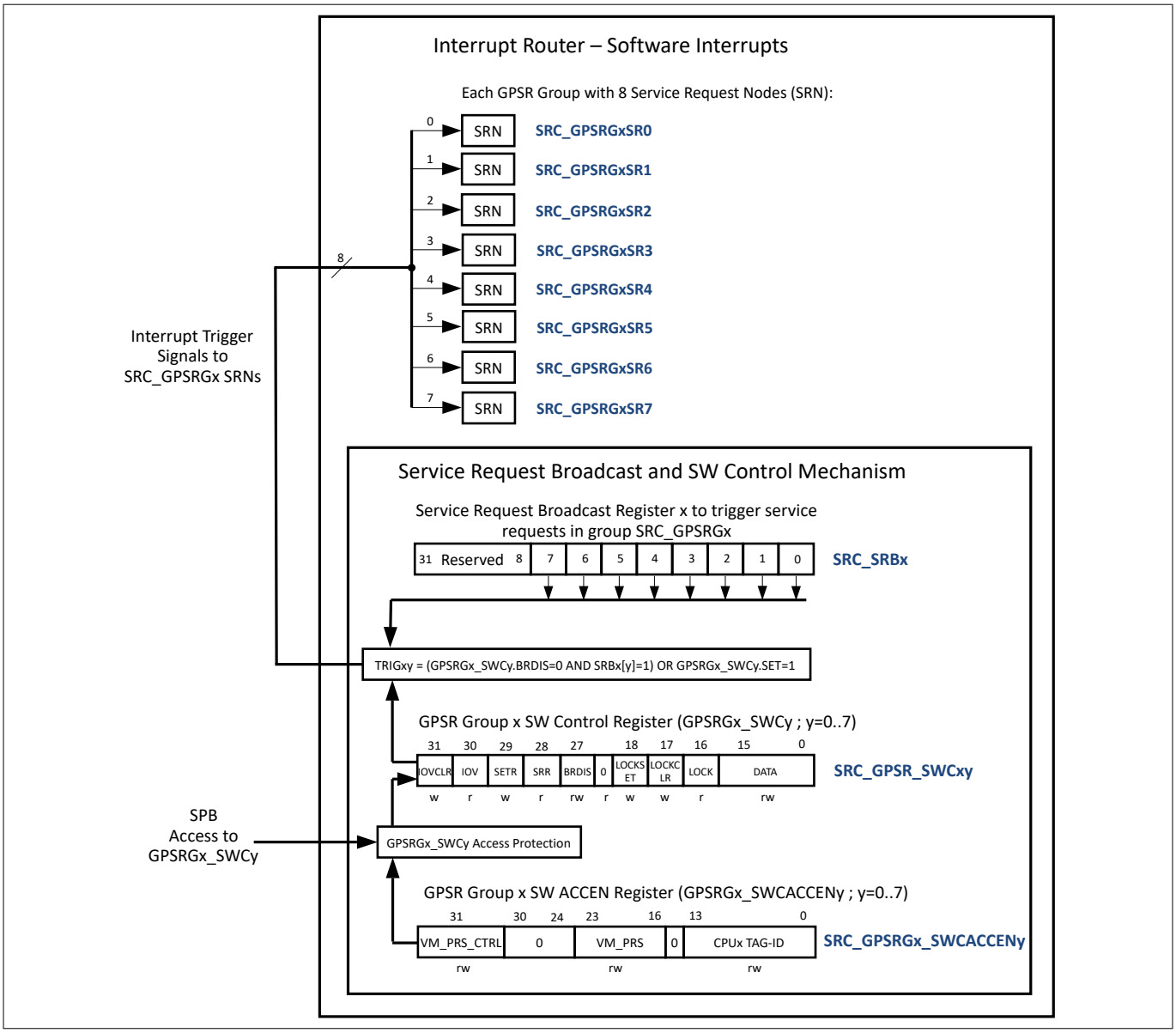


Figure 954 Structure of a general purpose service request group (GPSRx) and its related registers SRBx, SRBx_ACCEN, GPSRGx_SWCy, GPSRGx_SWCACCENy

Related information

Service request broadcast registers (SRBx) on page 6880

40.3.6.1 General purpose service requests (GPSR)

The Interrupt Router provides multiple groups of general purpose service request (GPSR) nodes:

- The GPSR nodes are implemented in groups
- Each GPSR group consists of eight service request nodes and its related service request control registers
- The GPSR SRC registers are named SRC_GPSRGxSRy; x = group number; y = number of interrupt within the group with y = 0:7
- The GPSR nodes are intended for software interrupts and not mapped to hardware service request triggers
- Each GPSR node can be assigned to one interrupt service provider (ISP)
- For the access to the GPSR SRC registers, the same access rules apply as for all other SRCx registers

Related information

Module implementation on page 6896

40.3.6.2 Triggering a general purpose service request node (GPSRx)

Any GPSR node can be triggered by software through 3 different ways:

Writing to the related service request control (GPSRGxSRy.SETR)

The register is access protected by the TOSz_ACCENCTRL where TOSz is selected by the register GPSRGxSRy.TOS=z configuration. TOSz_ACCENCTRL controls access to all SRN mapped to TOSz.

***Note**: This way is only suitable for software tasks that have access to all SRCs that are mapped to the ISP related to the TOSz encoding.*

Writing to the related service request broadcast register (SRBx)

This mechanism enables software to trigger multiple GPSR nodes of a group with one write access. Each SRBx register is protected by dedicated SRBx_ACCEN. The SRBx mechanism enables software to trigger multiple GPSRxy of the GPSR group in parallel.

***Note**: The broadcast mechanism can be disabled for each GPSRGxSRy individually by its related software control register (GPSRGx_SWCy).*

Writing to the related GPSR software control register (GPSRGx_SWCy)

GPSRGx_SWCy is protected by GPSRGx_SWCACCENy. The GPSRGx_SWCACCENy. allows application software to control for each GPSRGxSRy which CPU master agent functions are enabled for write access to the related GPSRGx_SWCy and can inject software interrupts into the GPSRGxSRy. The master agent functions are enabled through its transaction identifiers, TAG-IDs and VM or PRS codings.

The mechanism provides a separation between the software interrupt sources that can initiate software interrupts into a GPSR node and the interrupt destination that retains control of the corresponding GPSR node.

40.3.6.3 Service request broadcast registers (SRBx)

Service request broadcast registers (SRBx) can be used to assign service requests to multiple ISP´s in parallel.

There is one service request broadcast register (SRBx) implemented for each general purpose service request group (GPSRGx).

A (SRBx) register can be used to trigger multiple service requests within the GPSRx group in parallel.

- A SRBx register is always read as 0
- Writing 1 to SRBx.TRIGy triggers the service request GPSRGxSRy (if not disabled through GPSRGx_SWCy
- Writing 1 to SRBx[31:8] has no effect

40.3.6.4 Access protection of SRBx registers (ACCENSRBx)

Each SRBx register is write protected through a dedicated SRBx_ACCEN register set:

- Each SRBx register has a related ACCENSRBx register
- The configuration of the ACCENSRBx registers defines which master agent functions are allowed to write to the related SRBx register
- In the case of an access protection violation, the access is not executed, IR signals bus error to the SPB

All ACCENSRBx are write protected by PROTSE.

40.3.6.5 GPSR software control registers (GPSRGx_SWCy)

In comparison to the Broadcast register (SRBx), the GPSR software control registers offers more flexible and extended functions for setting and controlling software interrupts.

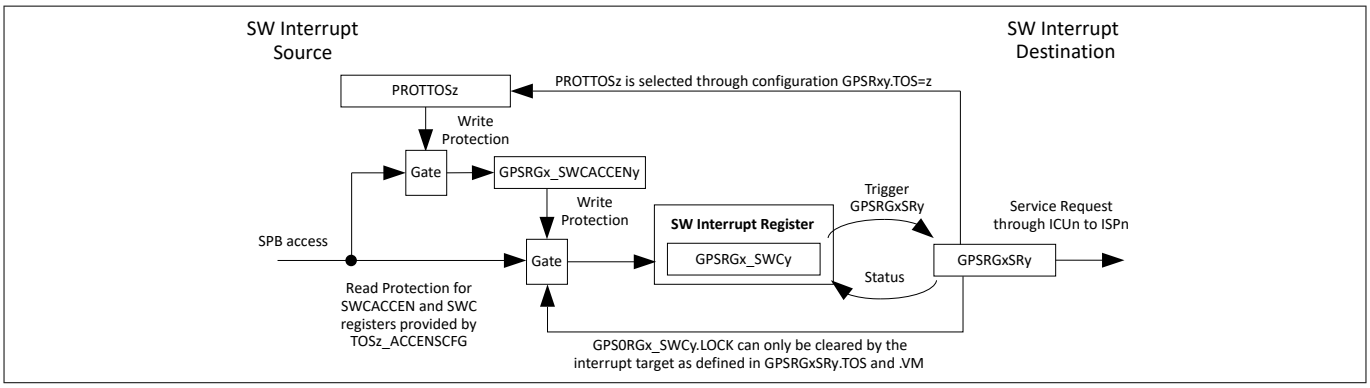


Figure 955 Structure of the GPSRGx_SWCy access protection and its relation to the GPSRx configuration

The GPSR Software Control functions includes:

- Dedicated GPSRGx_SWCy register for each GPSRGxSRy
- Reading the service request status (SRR) and interrupt overflow status (IOV) of related GPSRGxSRy (SRR, IOV)
- Set of a service request (SETR) and clear of the interrupt overflow (IOVCLR) in the related GPSRGxSRy (IOVCLR, SETR)
- Disable the SRBx trigger mechanism for GPSRGxSRy (BRDIS)
- The DATA register allows to attach additional information to the service request that the application software can read from the GPSRGxSRy.DATA register
- Hardware handshake mechanism for the DATA information (LOCKSTAT, LOCKCLR, LOCKSET)
- GPSRGx_SWCy.LOCKSTAT can only be set by a master agent function enabled for write access to GPSRGx_SWCy
- GPSRGx_SWCy.LOCKSTAT can only be cleared by the configured interrupt target (ISP). The target ISP is defined by GPSRGxSR.TOS and GPSRGxSR.VM

Protection: there is a dedicated GPSRGx_SWCACCENy register for each GPSRGx_SWCy that defines which master agent functions are to read and write to GPSRGx_SWCy.

40.3.6.6 GPSRGx_SWCy access protection (GPSRGx_SWCACCENy)

For each GPSRGx_SWCy there is a corresponding GPSRGx_SWCACCENy register. The GPSRGx_SWCACCENy enables application software to control write access permissions to the GPSRGx_SWCy for the implemented CPU´s and the Cerberus. This controls which CPUs and the Cerberus are enabled to inject software interrupts through GPSRGx_SWCy into the corresponding SRN GPSRGxSRy.

A GPSRGx_SWCACCENy register itself is write protected through the PROTTOSz and read protected by the TOSz_ACCENSCFG, as defined through the GPSRGxSRy.TOS=z configuration.

The control of the related GPSRGxSRy is assigned through the GPSRxSRy.TOS and its related TOSz_ACCENSCTRL configuration to a different set of master agent functions.

The GPSR SWC Access Protection function includes:

- Dedicated read and write access protection for each GPSRGx_SWCy
- Enable and disable TAG-IDs of CPUx and Cerberus for access to the related GPSRGx_SWCy
- Enable and disable of VM or PRS codings for read and write access to the related GPSRGx_SWCy

***Note**: If VM check is selected but the transaction id does not contain any valid VM information (VM_VALID = 1) then the VM access condition is considered to be fulfilled for the write to the protected GPSRGx_SWCy register . The same applies if PRS check is selected but no valid PRS information is transmitted (PRS_VALID = 0).*

40.3.6.7 Configuration sequence of GPSRGxSRy, GPSRGx_SWCy and GPSRGx_SWCACCENy

To ensure that the GPSRGx_SWCy register can only be used by the desired interrupt sources, the following sequence can be used:

- TOSz_ACCENSCTRL should be configured before GPSRGxSRy.TOS is configured with z and assigned to ICUz
- ACCENCS shall be enabled before GPSRGxSRy is to be marked as CSRM related by setting GPSRGxSRy.CS = 1
- With the configuration of GPSRGxSRy.TOS = z the register GPSRGx_SWCACCENy is:
  - Write protected by the register PROTTOSz
  - Read protected by TOSz_ACCENSCFG

***Note**: Because register GPSRGx_SWC is not write protected after a reset by default, the GPSRGx_SWCACCENy register should be configured after the above sequence was executed and before the GPSRGxSRy is enabled (SRE = 1) to prevent illegal software interrupts.*

40.3.7 External interrupts

The eight SRNs SRC_SCUERU[7:0] are reserved to handle external interrupts. The setup for external GPIO port input signals that are used by the application to generate an interrupt request is controlled in the External Request Unit (ERU). The ERU functionality is described in detail in the SCU chapter and allows to configure the type of the respective external interrupt signal.

40.3.8 Access protection of interrupt router registers (PROT, APU)

The IR provides a master agent function identifier based access protection. Each on chip resource with a interconnect master capability has one or multiple unique master agent function identifier that can be used to identify the master agent function that initiated a transaction on the interconnect system.

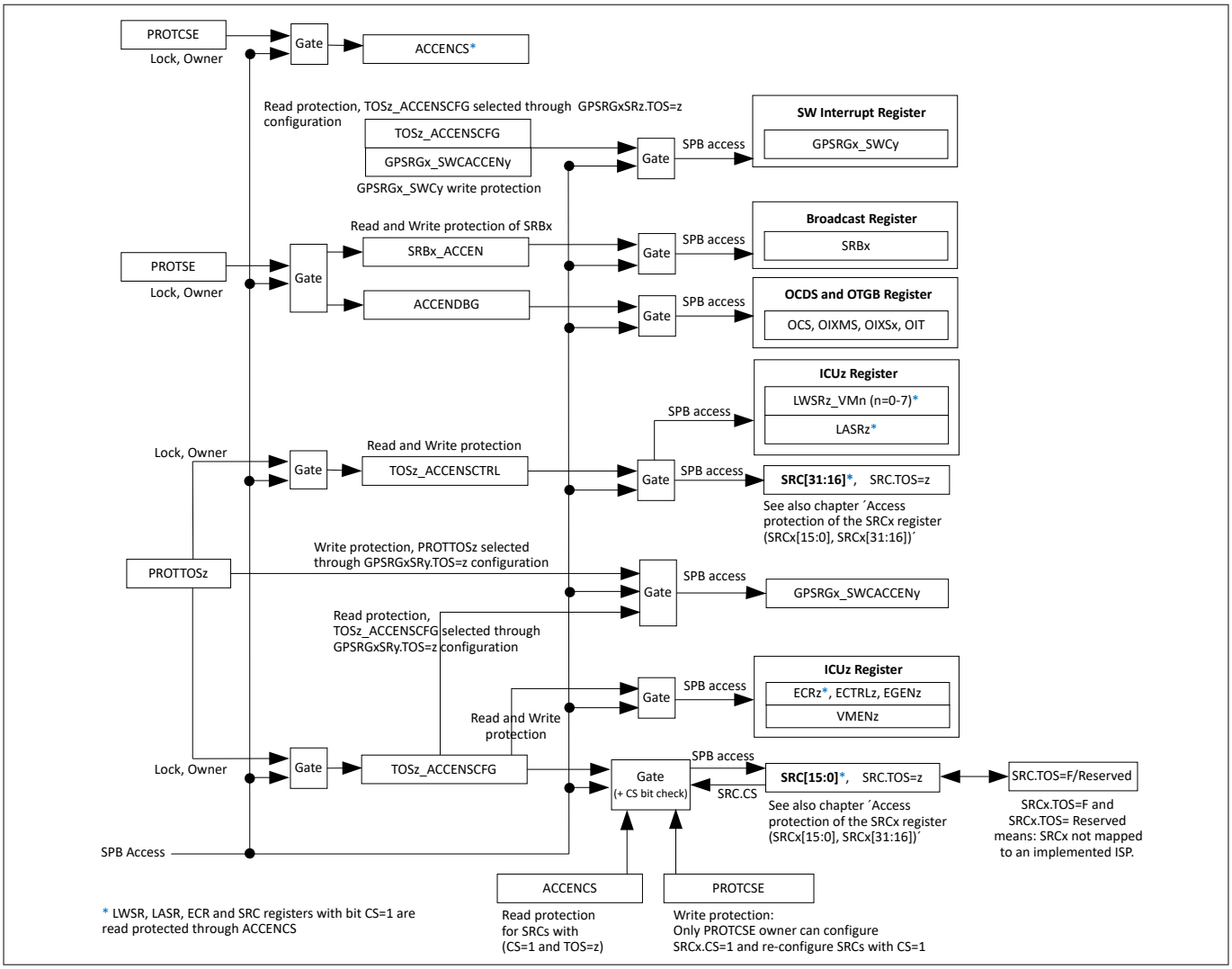


Figure 956 Access protection of interrupt router registers

PROT function implementation

The IR provides various PROT functions that write protect the APUs and SRC registers.

The IR provides five access enable register sets for the access control of the protected IR registers:

Access protection by SRBx_ACCEN

The SRBx_ACCEN registers provides access protection for the related SRBx register.

The IR provides a dedicated set of SRBx_ACCEN registers for each SRBx register. The SRBx_ACCEN registers are protected by the PROTSE.

Access protection by GPSRx_SWCACCENy

The GPSRGx_SWCACCENy registers provides write access protection for the related GPSRGx_SWCy register.

The IR provides for each GPSRGxSRy a dedicated set of SWC register (GPSRGx_SWCACCENy and GPSRGx_SWCy).

A GPSRGx_SWCACCENy register is access protected through the following registers:

- Write protected by PROTTOSz; PROTTOSz is selected through GPSRGxSRy.TOS=z configuration
- Read protected by TOSz_ACCENSCFG; TOSz_ACCENSCFG is selected through GPSRGxSRy.TOS=z configuration

Access protection by TOSz_ACCENSCTRL

The TOSz_ACCENSCTRL provides access protection for the following registers:

- SRCx[31:16], also called SRCx control information. The access protection for SRCx[31:16] is described in detail in the chapter ´Access protection of the SRCx register (SRCx[15:0], SRCx[31:16])´
- LWSRz_VMn (Last Winning Service Request, ICUz, Virtual Machine n). LWSRz_VMn registers with CS=1 are in addition read protected through ACCENCS
- LASRz (Last Acknowledged Service Request, ICUz). LASRz registers with CS=1 are in addition read protected through ACCENCS

Access protection by TOSz_ACCENSCFG

The TOSz_ACCENSCFG provides access protection for the following registers:

- SRCx[15:0], also called SRCx configuration information. The access protection for SRCx[15:0] is described in detail in the chapter ´Access protection of the SRCx register (SRCx[15:0], SRCx[31:16])´
- ECRz (Error Control Register, ICUz). ECRz registers with CS=1 are in addition read protected through ACCENCS
- ECTRLz (Error Capture Register, ICUz)
- EGENz (Error Generation Register, ICUz)

Access protection by ACCENDBG

The ACCENDBG provides access protection for the debug related registers:

- OCS
- LCLTEST
- OIXMS, OIXS0, OIXS1, OIT

Access protection by ACCENCS

The ACCENCS provides read access protection for the following registers in case these registers are configured with CS=1_B or when they have captured service request information marked with CS=1_B.

If CS=1_B is set in the following registers, ACCENCS defines which master agent function is enabled to read the information of the respective register with CS=1_B.

- LWSRz_VMy
- ECR
- SRC

Related information

Access protection of SRBx registers (ACCENSRBx) on page 6880

Access protection of the SRCx register (SRCx[15:0], SRCx[31:16]) on page 6884

40.3.8.1 Access protection of the SRCx register (SRCx[15:0], SRCx[31:16])

The IR contains all hardware and software service request nodes. It allows a flexible assignment of the service requests to the implemented ISPs (CPUx, CPUCS, DMAs, PPU, GTM) through the configuration of the service request nodes (SRN). So the IR is a shared resource that can be used by the CSRM for secure tasks as well as by other CPUs for not security related tasks (for DMAs see also related information). After assigning the SRNs to the ISPs, the IR allows to distinguish between these groups:

SRCx.TOS=z - SRNx is assigned to an implemented ICUz

TOS=z is used in the following description as a synonym for mapping an SRC to an implemented ICUz and so to an implemented interrupt service provider (ICUz).

Within the group of SRNs that are assigned or mapped to an implemented ISPz, a distinction is made between

- The group of SRN marked as security related (SRC.TOS=z and SRC.CS=1)
- The group of SRN marked as non-security related (SRC.TOS=z and SRC.CS=0)

All these groups can be protected against unauthorized configuration and reconfiguration by other master functions. This is made possible by the respective PROTTOSz, ACCENCS and PROTCSE and TOSz_ACCENSCFG.

SRCx.TOS=F and SRCx.TOS=Reserved - SRNx is not assigned to an implemented ICU

TOS=F_H or TOS=Reserved is used in the following description as a synonym for mapping an SRC to the group of unused SRCs. The SRCs configured in this way are not read protected and write protection only takes effect if they are to be mapped to an implemented ICUz, see TOS=z.

***Note**: PROTTOSz should be enabled (ODEF=1) before TOSz_ACCENSCFG and TOSz_ACCENCTRL are configured to prevent unintended modification of these registers that might enabled unauthorized master agent functions for SRC access.*

***Note**: PROTCSE is enabled (ODEF=1) after reset, configured PROTCSE owner is CPUCS. Disabling PROTCSE by setting ODEF=0 enable unauthorized master agent functions to configure SRC.CS=1 and trigger DMA channels that are assigned to the secure DMA Resource Partition (RP).*

SRCx[15:0] assignment status after reset

After reset all Service Request Nodes (SRN) are assigned to the group of not used SRNs (SRCx.TOS=F_H) and marked as not secure (SRCx.CS=0). The TOS encoding F_H is not assigned to an Interrupt Service Provider in the AURIX™ TC4xx family.

Application software can assign SRN with SRCx.TOS=F_H to an existing ISPz (TOS=z) and mark them as security related or not security related.

For a word write to an SRC currently configured with SRC.TOS=F_H: if the write does not satisfy access conditions to SRC[15:0], SRC[31:16] will be updated. No FPI error or alarm will be triggered.

Write access to SRCx[15:0], assignment to TOS=z or SRCx.TOS=F, not-CS related (CS=0)

Software can configure SRCx[15:0] with TOS=z and CS=0 or re-configure a SRCx that is already configured with SRCx.TOS=z and SRCx.CS=0 if following conditions are met:

- Assumption: SRCx configuration is (SRCx.TOS=z and SRCx.CS=0) or SRCx.TOS=F_H
- The writing master agent function must be enabled in TOSz_ACCENSCFG for write
- If PROTCSE.ODEF=1, which means that the IR PROTCSE owner is enabled:
  - the writing master agent function must not be the PROTCSE owner
- The writing master agent function can write:
  - (SRC.CS=0 AND SRC.TOS= z) to assign an SRC with SRC.TOS=F to the SRC TOS=z with SRC.CS=0
  - (SRC.CS=0 AND SRC.TOS= z) to re-configure an SRC that is already configured with SRC.CS=0 AND SRC.TOS= z
  - (SRC.CS=0 AND SRC.TOS=F_H) to assign an SRC already configured with SRC.CS=0 AND SRC.TOS= z back to the group of not used SRC

Write to SRCx[15:0], assignment to TOS=z or SRCx.TOS=F, CS related (CS=1)

Software can configure SRCx[15:0] with TOS=z and CS=1 or re-configure a SRC that is already configured with SRCx.TOS=z and SRCx.CS=1 if following conditions are met:

- Assumption: SRCx configuration is (SRCx.TOS=z and SRCx.CS=1) or SRCx.TOS=F_H
- The writing master agent function must be enabled in TOSz_ACCENSCFG for write
- IF PROTCSE.ODEF=1, which means that the PROTCSE owner is enabled:

- The writing master agent must be equal to the PROTCSE owner
- The writing master agent function writes:
  - (SRC.CS=1 AND SRC.TOS= z) to assign an SRC with SRC.TOS=F to the SRC TOS=z with SRC.CS=1
  - (SRC.CS=1 AND SRC.TOS= z) to re-configure an SRC that is already configured with SRC.CS=1 AND SRC.TOS= z
  - (SRC.CS=0 AND SRC.TOS=F_H) to assign an SRC already configured with SRC.CS=1 AND SRC.TOS= z back to the group of not used SRC

Write to SRCx[31:16], SRCx is assigned to TOS=z, CS=0

Software can write to SRCx[31:16] if following conditions are met:
- Writing master agent function enabled in TOSz_ACCENSCTRL for write
- Writing master agent function writes with VM0 or VM=SRCx.VM or it writes with VM_VALID=0
- Write access with CS_VALID=0

Write to SRCx[31:16], SRCx is assigned with TOS=z, CS=1

Software can write to SRCx[31:16] if following conditions are met:
- Writing master agent function enabled in TOSz_ACCENSCTRL for write
- Writing master agent function writes with VM0 or VM=SRCx.VM or it writes with VM_VALID=0
- Write access with CS_VALID=1

Write Access to SRCx[31:0]

Software can write to SRCx[31:0] if the writing master agent function is at least enabled for write to SRCx[31:16] or SRCx[15:0]:
- If write condition for SRCx[31:16] is met, SRCx[31:16] is updated
- If write condition for SRCx[15:0] is met, SRCx[15:0] is updated

If the access master agent function ID of a write access is neither write enabled for SRCx[31:16] nor write enabled for SRCx[15:0] it is an access protection violation.

This also means that for a word write to an SRC currently configured with SRC.TOS=$F_H$or another reserved TOS encoding: if the write does not satisfy access conditions to SRC[15:0], SRC[31:16] will be updated. No FPI error or alarm will be triggered. Therefore, if a SRN is configured from a reserved TOS encoding to a non-reserved TOS encoding with a word access, the configuration should be checked by reading it back.

Reading SRCx configured with SRCx.CS=0

Software can read a SRCx that is configured with SRCx.TOS=z and SRCx.CS=0 if the following condition is met:
- Reading master agent function is at least enabled for reading in TOSz_ACCENSCTRL or TOSz_ACCENSCFG for read

If the reading master agent function is neither enabled for reading in TOSz_ACCENSCTRL nor TOSz_ACCENSCFG it is an access protection violation.

If SRCx is configured with a reserved TOS encoding, for example SRCx.TOS=F_H, all master agent functions are enabled to read. A reserved TOS encoding is not related to an implemented ISP.

Half word and byte access have the same condition as word read accesses.

Reading SRCx configured with SRCx.CS=1

Software can read a SRCx that is configured with SRCx.TOS=z and SRCx.CS=1 if the following condition is met:
- Reading master agent function must be enabled in ACCENCS for read

Otherwise it is an access protection violation.

Half word and byte access have the same condition as word read accesses.

SRCx[15:0] and SRCx[31:16] access with protection violation

If an access protection violation occurs, the access is not executed and a FPI error is generated. On detection of a FPI error on the FPI interconnect the Bus Control Unit (here: SBCU because the IR is connected to the SPB interconnect) signals an Alarm_NCS (CS_VALID=0) to the SMU_SAFE or an Alarm_CS (access with CS_VALID=1) to the SMU_CS.

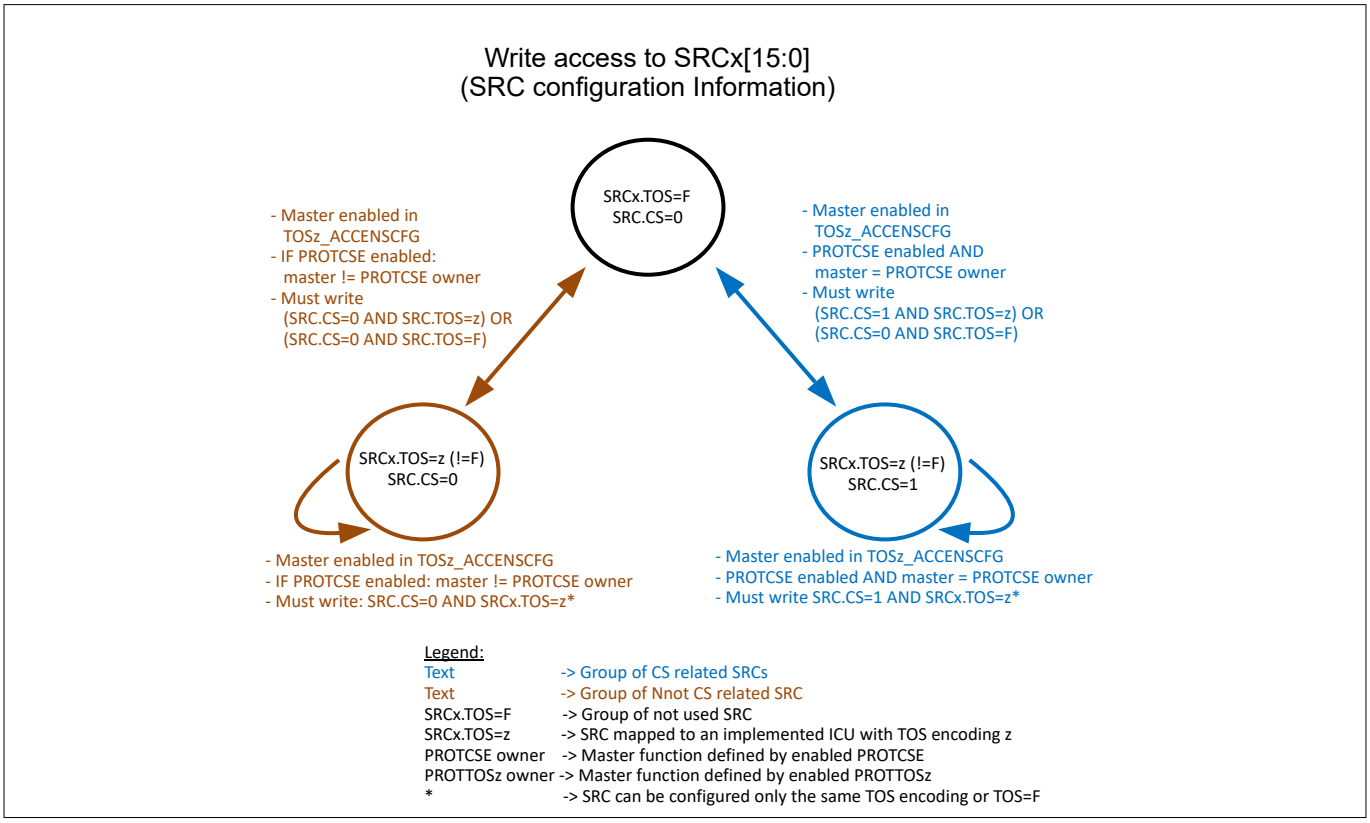


Figure 957 Write access to SRC configuration information (SRC[15:0])

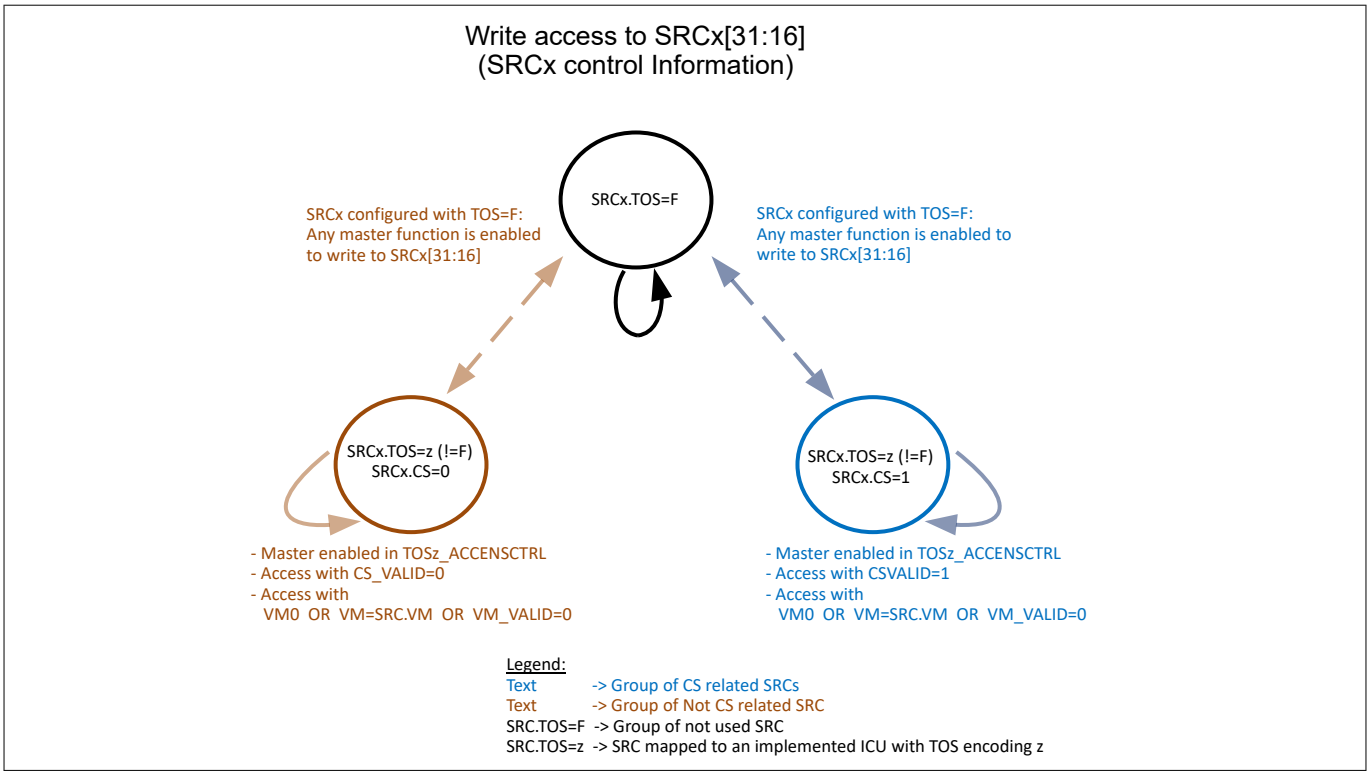


Figure 958 Write access to SRC control information (SRC[31:16]).

Related information

ICU to DMA interface on page 6874

40.3.9 Arbitration process

The IR provides a dedicated ICUz for each interrupt service provider (ISP) and a dedicated arbitration mechanism for each ICU which arbitrates among the pending SRs for the related ICUz and ISP.

Each service request node (SRNx) can be directed to one ISP by mapping it through the SRCx.TOS bit-field setting to the related ICUz.

As soon as an SR is pending for an ICUz the IR will start with the arbitration sequence for this ICU.

The ICUz will signal every cycle the winning service request information of the current arbitration round.

The arbitration sequence for ICUz will stop if either no SR is pending anymore of ICUx or as long the ´Enter´ signal is asserted by the related ISP.

***Note**: In the current implementation the ECC code is only used for error detection. Detected errors are reported to the SMU but not corrected.*

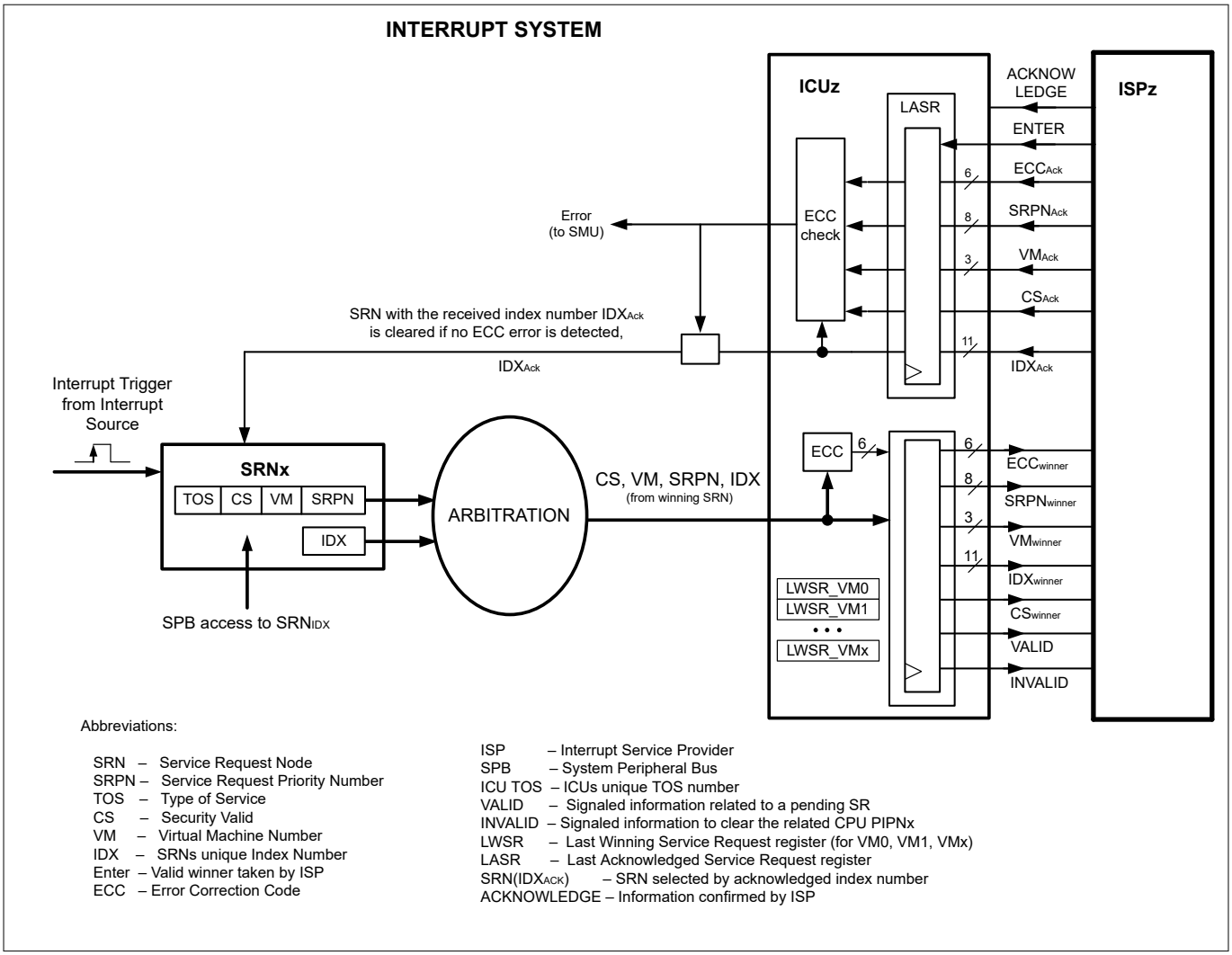


Figure 959 Interrupt system arbitration scheme overview

40.3.9.1 Arbitration mechanism

Each interrupt node can be mapped through its SRCx.TOS bit field to one ICUz and so to the related interrupt service provider. For a CPU with hardware virtualization support it can be mapped to one VM on the CPU through the SRCx.VM bit field.

When arbitrating for an ISP, the IR selects the enabled and pending service request with the highest service request priority number (SRPN) assigned to that ISP.

Arbitration for the implemented ISPs that are CPUs with hardware virtualization support is done in parallel. Arbitration for the active VMs on a CPU is done in a fixed sequence.

Arbitration for ISPs that are not CPUs with hardware virtualization support is done in time division multiplexing. Here, a maximum of two of these ISPs are arbitrated in a single clock time-division multiplexing procedure.

For a CPU with hardware virtualization support, service request arbitration is performed for each active VM individually.

In an arbitration process for ICUz, the arbitration mechanism compares the SRCx.SRPN bit-fields of all pending service request nodes that are mapped to this ICUz (SRCx.TOS=z) and to the VM it is just arbitrating for on this ICUz (SRCx.VM configuration).

During the arbitration process for one ISP with only one VM enabled, the pending service request with the highest priority number is identified as winner and the related SRN service request control register bit-field values SRPN, VM, ECC and the Index of the SRN are provided to the ICU.

When multiple VMs are enabled for a CPU the IR arbitrates for enabled VMs on that CPU in a fixed sequence. During the arbitration sequences for the activated VMs, the ICU signals for each enabled VM the highest SRPN for this VM (VALID=1 AND INVALID=0).

For enabled VMs without a pending service request the ICU signals no information (VALID=0 AND INVALID=0).

The ICU signals these service request information to the ISP (SRPN, VM, ECC, CS, IDX) to the ISP. The ICU does an ECC check when it gets these information back from the service provider with an acknowledge.

The ECC check is done with the received values: SRPN, VM, ECC, CS, IDX and the TOS information related to the respective ICU. On detected ECC error, IR does not clear the SRN related to the received IDX and signals an alarm to the SMU.

The INVALID is used in a special situation: When the last pending SR for VMx on a CPUy was deleted by SW. Then the SR in the IR is deleted, but the SR information in the CPUy can still exist in the CPUy.PIPNx register. To clear this information the IR signals an INVALID with the VMx information to the CPUy which causes the CPUy to set the register bit field PIPNx.SRPN=0.

Example situation in which INVALID is used by the ICU: VMx is enabled, only one SR is pending for VMx. The SR has been signaled to VMx but not yet acknowledged which can be seen by software through LWSRx.STAT=1 and LWSRx.VALID=1. Now software clears the SR in the interrupt router. After this, the ICU does not see a new winner in the next arbitration round for VMx but LWSRx shows the last pending service request for VMx that was cleared by SW. In this situation the ICU signals an INVALID for VMx to the CPU, clears LWSRx.STAT bit and updates the other LWSRx bits with the signaled information.

40.3.9.2 IR hardware virtualization support (VMENz)

Each ICUz has one VMENz register. An ICU for an ISP CPU with hardware virtualization support has VMENz [7:0] implemented with read and write register bits, in addition 8 LWSRz registers are implemented. These VMENz[7:0] bit numbers correspond to the VM numbers on the CPU and must be configured according to VMs used on the CPU. Thereby the ICU supports two different arbitration mechanisms which are controlled by the VMENz configuration.

- VMENz[7:0] is 0 . This configuration should be selected if the CPU is not used in hardware virtualization mode. The SRCx.VM information is ignored by ICUz for arbitration, alll service requests are signaled with VM=0_H to the ISP
- VMENz[7:0] is not 0. If the CPU is used in hardware virtualization mode, the used virtual machines (VM) must be activated in VMENz[7:0] and the corresponding SRNs must be assigned to the VMs through SRNx.VM. Any VMENz[7:0] configuration is allowed and supported by the IR, including VMEN[0]=0 which means that VM0 disabled and does not make sense for a CPU with hardware virtualization support enabled

In case of ICUz related to an ISP CPU and at least one VM is enabled in register VMENz the IR uses the SRCx.VM information for an arbitration mechanism that is specifically adapted to the TriCore™ hardware virtualization mode. For all enabled VMs (VMENz) the IR arbitrates among all pending service requests that are mapped to this ICUz through SRCx.TOS, and to the virtual Machine through SRCx.VM.

- Case 1: VMENz shows all VM disabled. SRCx.VM information will be ignored. The winner of an arbitration round is the pending service request with the highest SRCx.SRPN number. Service requests are signaled with VM=0_H to the ISP
- Case 2: VMENz shows one or more VMs enabled. The IR arbitrates for the enabled VMs in a fixed sequence. The winner of an arbitration round for a VM is the pending service request with the highest SRCx.SRPN mapped to this VM through the SRCx.VM configuration. The arbitration sequence is fixed even when there is no service request pending for an enabled VM

***Note**: If an ICU related to CPUx is reconfigured from VMEN != 0 (hardware virtualization enabled) to VMEN=0 during operation, it must be ensured that no service requests are pending for VM1-VM7. Otherwise the register CPUx.PIPN0 could be wrongly set to 0 due to a last INVALID information for VMy (y=1-7).*

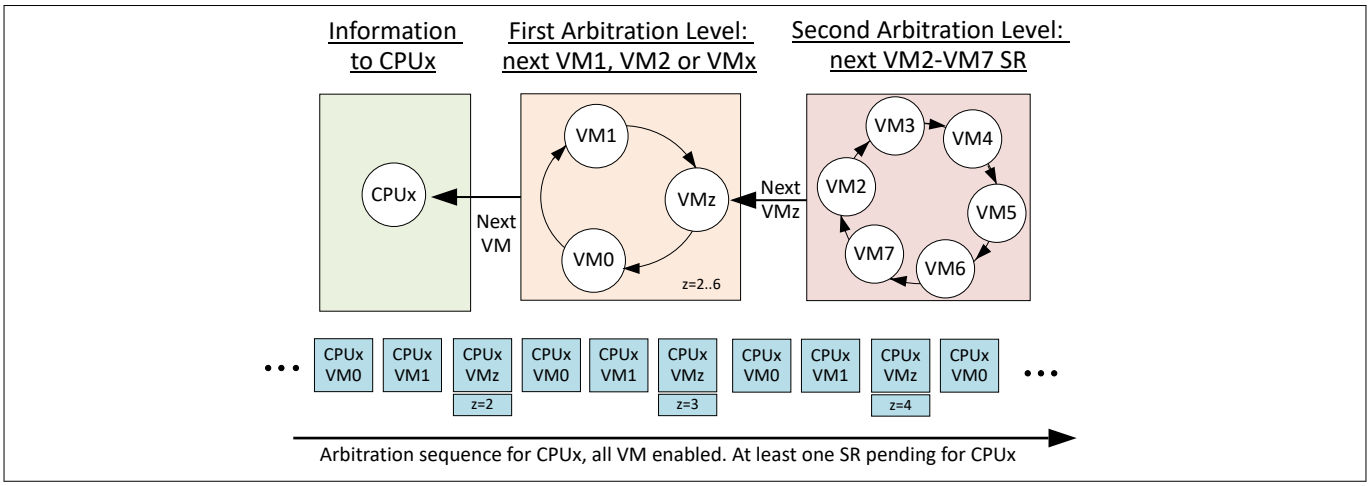


Figure 960 Example for CPUx. All VMs enabled through related ICU VMEN. At least one SR pending for CPUx, Enter signal not asserted (otherwise arbitration would stop until Enter signal is de-asserted)

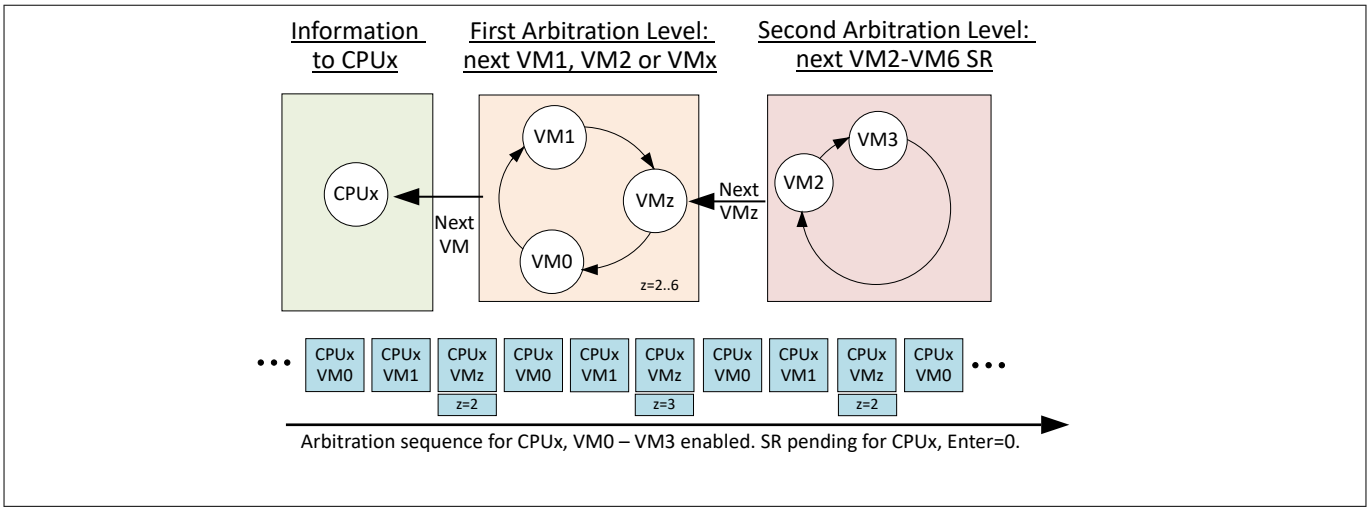


Figure 961 Example for CPUx. Only VM0 - VM3 enabled through related ICU VMEN. At least one SR pending for CPUx. CPUx Enter signal not asserted

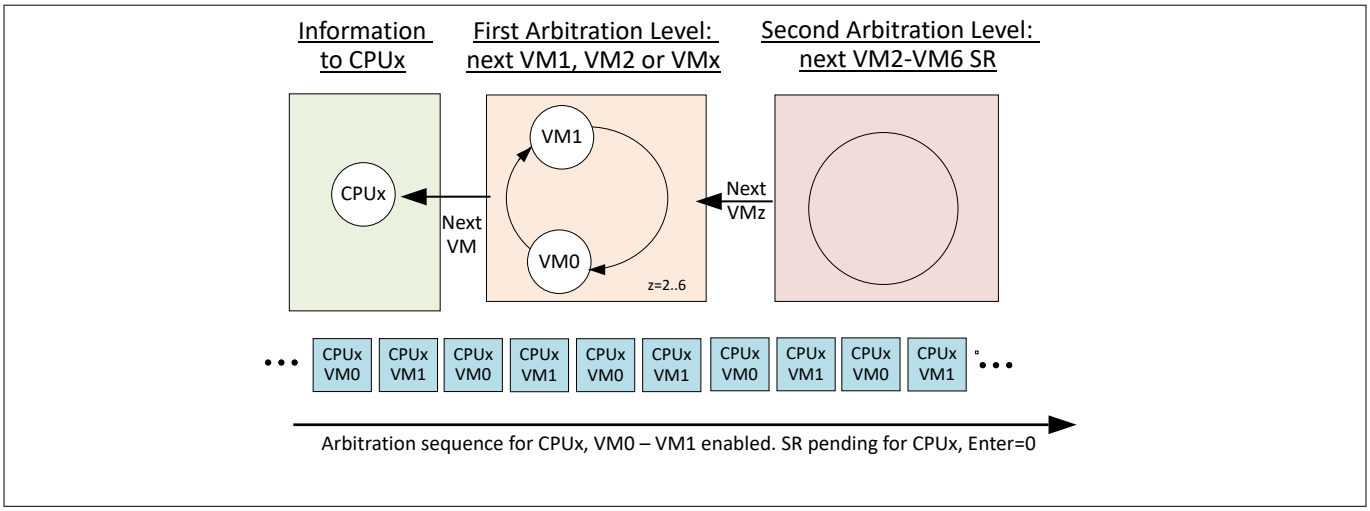


Figure 962 **Example for CPUx. Only VM0 and VM1 enabled through related ICU VMEN. At least one SR pending for CPUx. CPUx Enter signal not asserted**

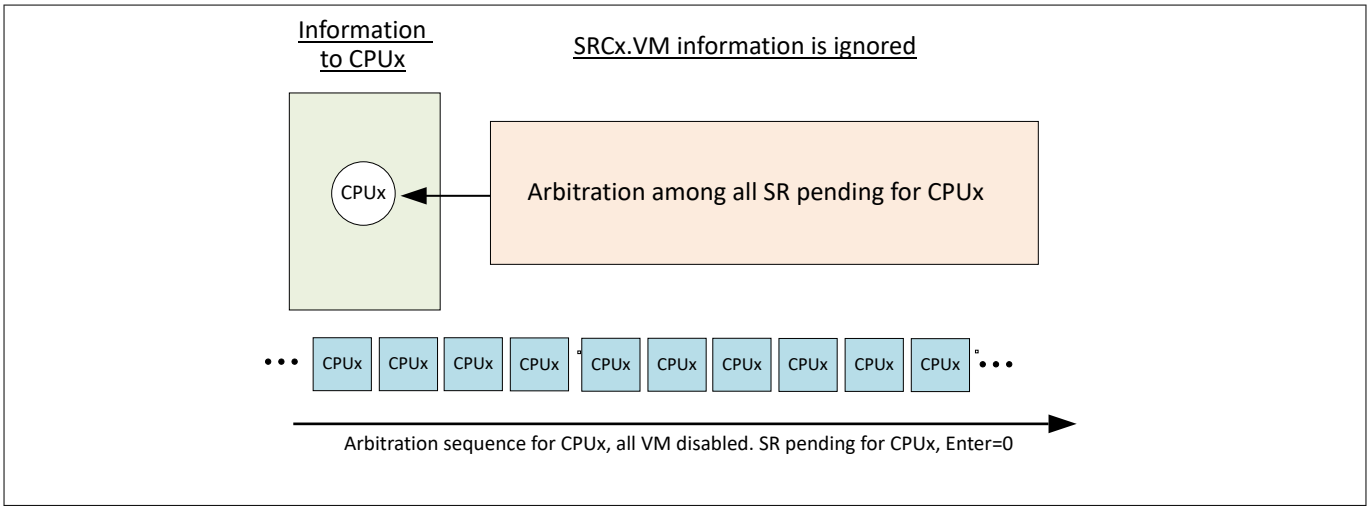


Figure 963 **Example for CPUx (GTM, DMA, PPU) with all VMs disabled through related ICU VMEN. At least one SR pending for CPUx, Enter signal not asserted**

40.3.9.3 SRCx information used for the arbitration

To determine the arbitration round winner for an ISP, the IR uses the following information:

- Service Request Enable (SRCx.SRE) and Service Request Flag (SRCx.SRR) to identify the enabled service request nodes with a pending service request
- Type of Service (SRCx.TOS) to identify the group of pending SR mapped to a physical ISP (CPUs, DMAs, PPU, GTM)

In case of an ISP for CPU with hardware virtualization support the IR uses the following information in addition:

- Register VMENz to identify the enabled VMs on that CPU. The VMENy configuration defines the executed arbitration sequence
- If a VMENz is configured to show all VM as disabled the SRCx.VMx information will be ignored for the arbitration (can be used if CPU is not used in Hardware Virtualization mode)
- SRC Virtual Machine Information (SRCx.VM) to identify per ISP the group of pending SR per VM encoding that is enabled through VMENz

In case of an ISP DMA, PPU, GTM or CPU without hardware virtualization support:

- VMENz is implemented hardwired, all bits are 0
- For interrupt nodes mapped to these ISPs the SRCx.VM information has no effect

40.3.9.4 Number of clock cycles per arbitration

The maximum interruption latency can be calculated with the following formulas. The formulas take into account the type and number of VMs that are activated through the VMEN register. The interrupt latency is calculated in f_{SPB} clock cycles and contains the interrupt path from the interrupt source where the functional block interrupt trigger output is signaled in the first cycle, to the clock cycle at which the service request is signaled to the ISP.

Latency for VM0 and VM1 interrupts

$Latency_VM_{01} = LBV + VM01 - 1$

Latency for VMx interrupts (VMx -> VM2 ... VM7)

$Latency_VM_{2..7} = LBV + VMx * VM01 - 1$

Table 1913 Description of variables used in interrupt latency formula

Variable	Description
VM01	VM01 indicates the number of enabled VM in the group VM0, VM1 and VMz. VM01 to be replaced in the formula with: • 1 -> if exactly one out of the group VM0, VM1, VMz is enabled • 2 -> if exactly two out of the group VM0, VM1, VMz are enabled • 3 -> if VM0, VM1 and VMz are enabled ***Note**: VMz indicates here if at least one VM of the group VM2-VM7 is enabled through the VMEN register. Example: VM0 is enabled, VM1 is disabled and one or more of VM2-VM7 are enabled then VM01=2. Example: VM0 and VM1 are enabled, VM2-VM7 are disabled then VM01=2*
VMx	VMx indicates the number of activated VM in the group VM2-VM7. VMx to be replaced in the formula with: • 0 -> if VM2 - VM7 are disabled in register VMEN • 1 -> if exactly one VM in the group is enabled • … • 5 -> if exactly five VM in the group are enabled • 6 -> if all VM in the group are enabled in register VMEN
LBV	Latency Basic Value -> 4 The LBV is a fix value and describes the best case latency from the interrupt trigger event of the interrupt source (fSPB clock cycle 2) through the IR arbitration for ICUx to the signaling of the SR to the ISPx (see following figure, ICU_VALID=´1´, f_{SPB} clock cycle 5).The best case latency is reached when a new interrupt is immediately arbitrated by the arbitration tree. This is always the case if no SR was pending before, then the VM arbitration sequence will start with VM0, or if all VMs are disabled and a new interrupt has the highest priority of all pending SR

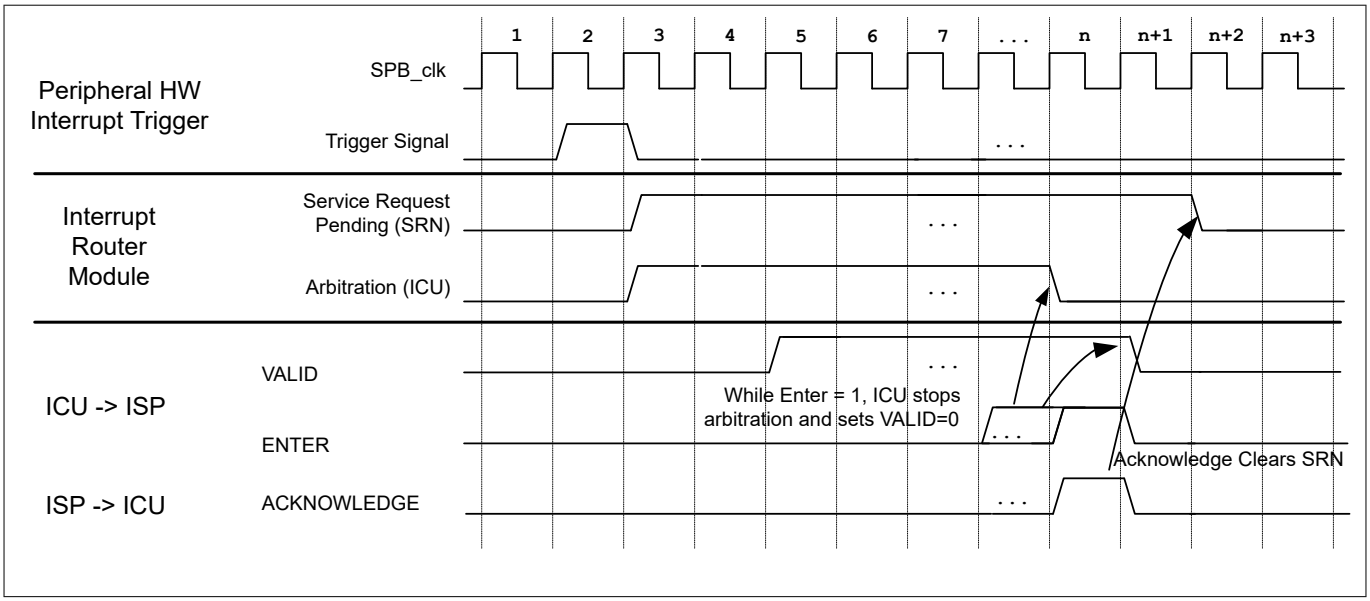


Figure 964 Interrupt system timing - example 1: single SR pending

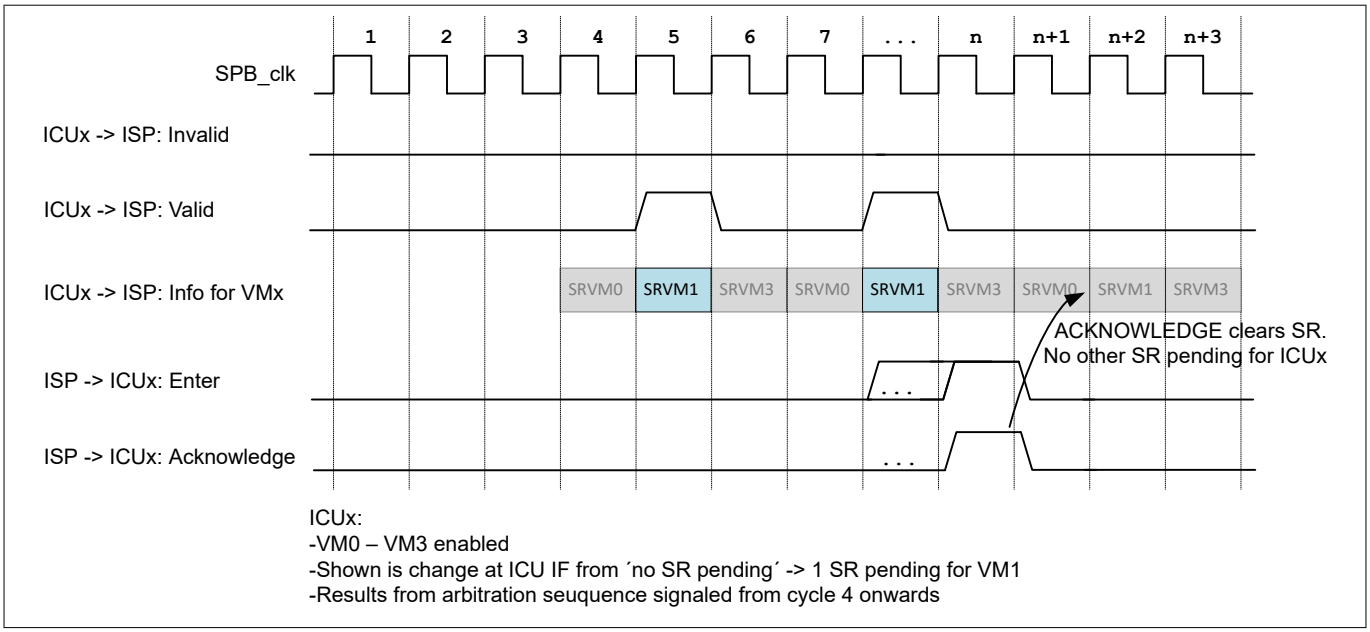


Figure 965 Interrupt system timing - example 2: ICUx with ISP=CPU interface timing

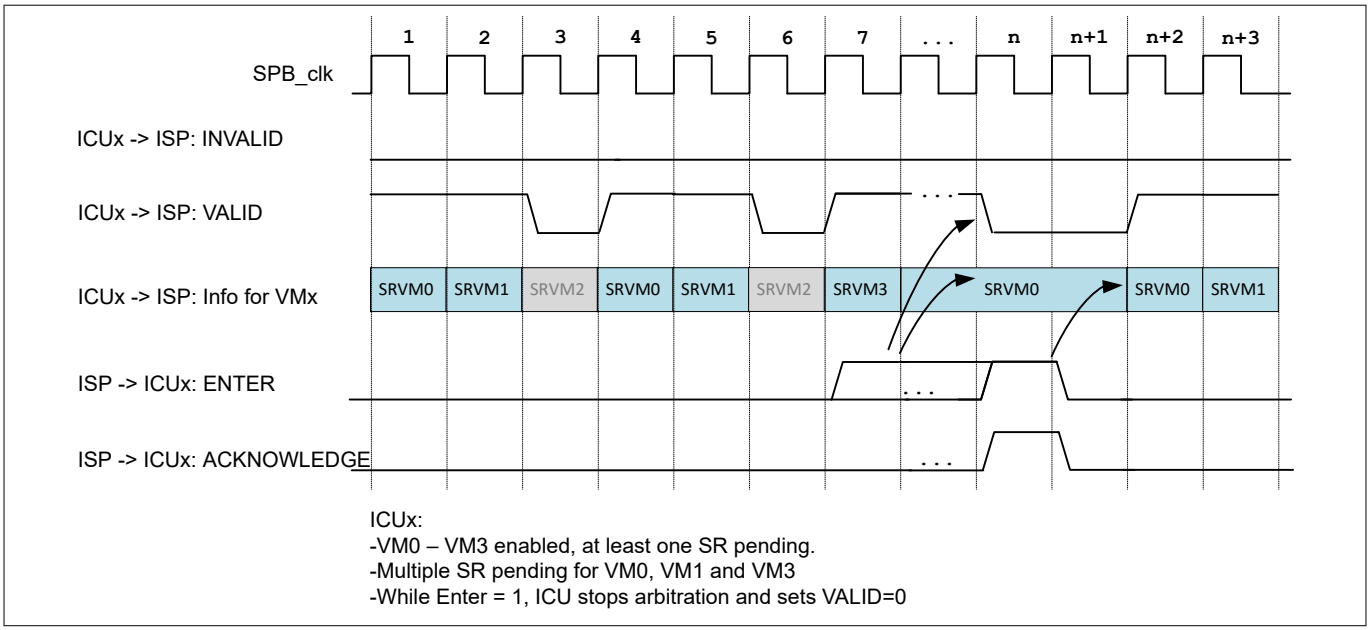


Figure 966 Interrupt system timing - example 3: ICU to CPU interface timing

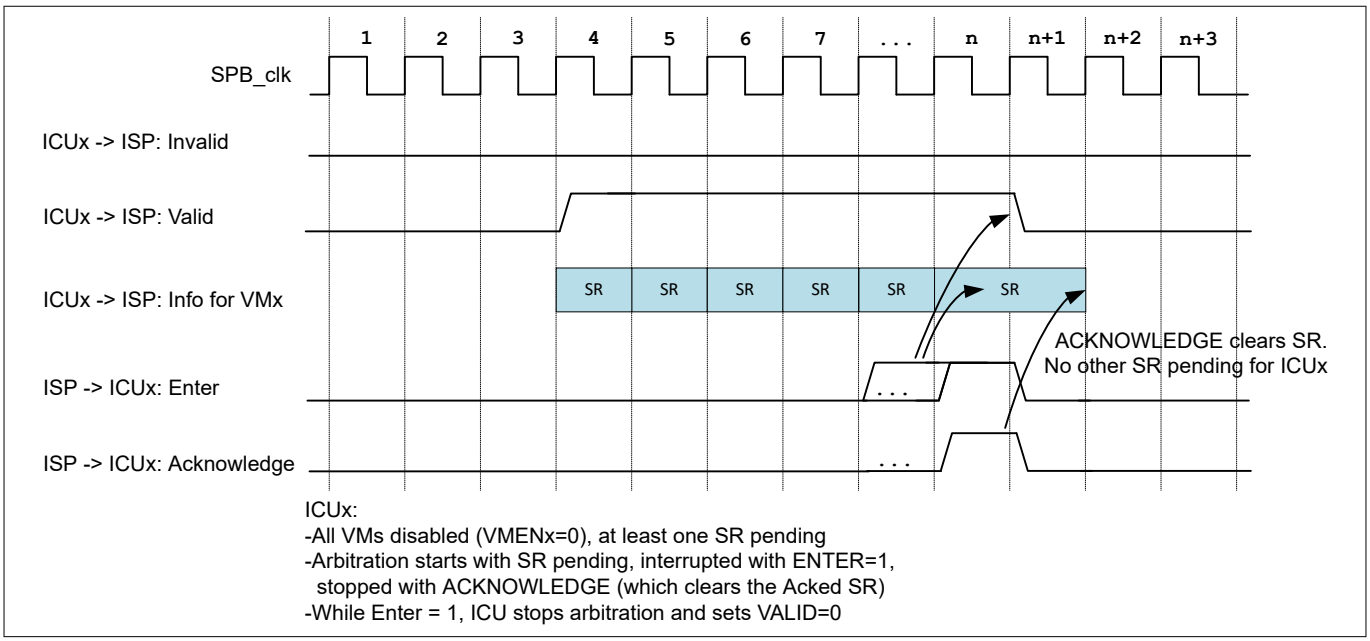


Figure 967 Interrupt system timing - example 4: ICU to CPU interface timing

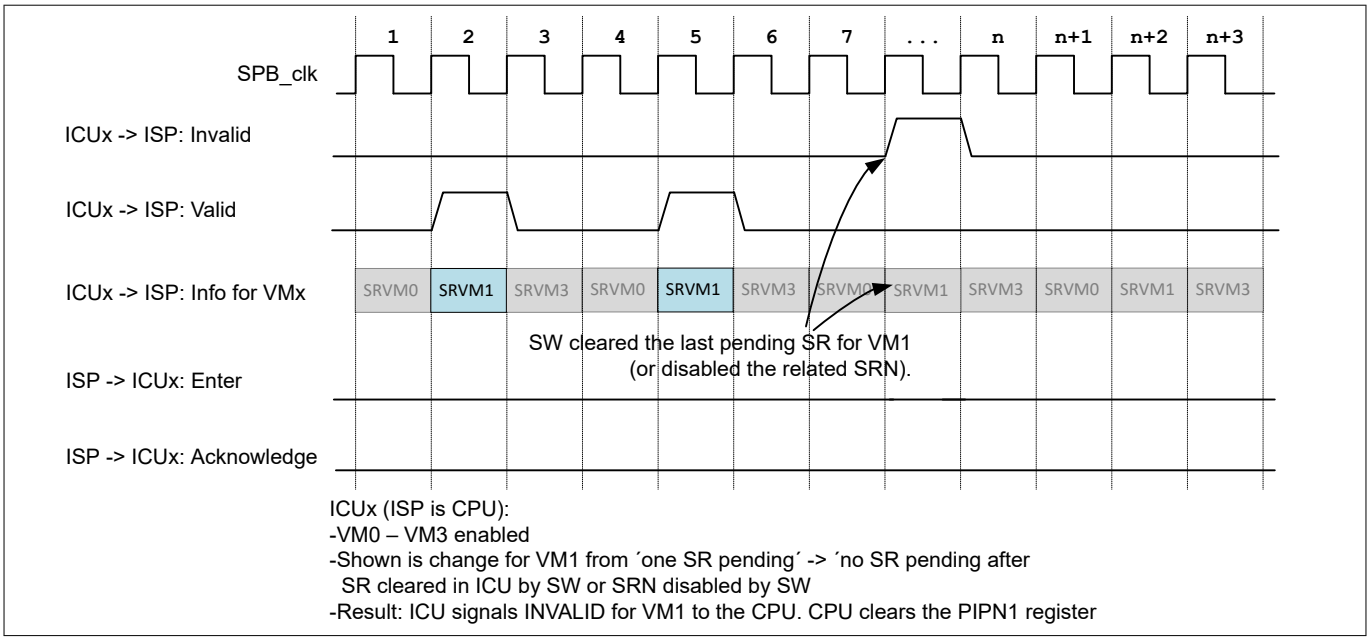


Figure 968 Interrupt system timing - example 5: ICU to CPU interface timing

40.3.10 IR wake-up support for CPUs and PPU

The IR provides wake-up support for the functional blocks CPUs and the PPU. The wake-up function can cause a functional block in IDLE or SLEEP to enter RUN mode. For this purpose the IR signals for each CPU and PPU separately if a service request is pending. A pending service request means that SRR=1 and SRE=1. This information is signaled to the functional block SMM.

40.3.11 IR alarm signals to SMU

IR alarm signals to SMU

The IR provides three alarm signals:

- ALARM_CS on detection of an ECC error in an acknowledged service request information where the service request CS information is 1. This alarm is mapped to the SMU_CS
- ALARM_NCS on detection of an ECC error in an acknowledged service request information where the service request CS information is 0. This alarm is mapped to the SMU_SAFE
- ALARM_LCL is signaled to the SMU_SAFE on detection of an IR lockstep error

Each ICU has implemented an independent ECC check mechanism where it checks the service request information it received through Acknowledge by the related ISP. The alarm_cs and alarm_ncs events from all implemented ICUs are combined to a single ALARM_CS and a single ALARM_NCS. Each ICU provides a dedicated error capture register (ECR) that captures the information of the first acknowledged service request where an ECC failure was detected. Each ICU provides a dedicated error generation register (EGEN) that enables application software to inject ECC errors in order to test the ECC check mechanism an the ALARM_CS and ALARM_NCS signal path to the SMU.

The IR lockstep compare mechanism and the ALARM_LCL signal path to the SMU can be tested through the lockstep test register (LCLTEST).

40.3.12 Module implementation

This chapter describes the IR ICU configurations and the number of General purpose service request Groups (GPSR) for the AURIX™ TC4xx family.

The product-specific IR ICU and SRC registers are described in the respective product-specific UM chapter.

40.3.12.1 Characteristics of the interrupt router module

The following tables show the Interrupt router configurations of the TC4xx devices.

The tables enable the customer to write generic software that can be easily ported between the targeted derivatives and to recognize the differences between the derivatives.

ICU hardware virtualization support (HVS):

The table also shows which ICU provides the hardware virtualization support (HVS).

- ICU marked with ´*´: ICU is implemented with HVS support. It provideshas eight LWSR registers (LWSR0-LWSR7) and the ICUs´VMEN[7:0] register bits are implemented with read and write bits to control which VM encoding to be considered. If all VMs are disabled in the VMEN register, hardware virtualization support is disabled, SRCx.VM information is ignored by the ICU arbitration and only LWSR0 is updated by hardware
- ICU marked without ´*´ : ICU is implemented without HVS support. It provides one LWSR register (LWSR0) and the VMEN register implemented hardwired 0 (HVS disabled). The SRCx.VM information is ignored by the ICU arbitration. Only LWSR0 is updated by hardware

Table 1914 Mapping of interrupt service provider to ICUs and implemented SRNs

	CPU0	CPU1	CPU2	CPU3	CPU4	CPU5	CPUCS	SDMA0	SDMA1	GTM	PPU	RDMA
TC49xN	ICU0*	ICU1*	ICU2*	ICU3*	ICU4*	-	ICU6	ICU7	ICU8	ICU9	ICU10	-
TC46x	ICU0*	ICU1*	ICU2*	ICU3*	-	-	ICU6	ICU7	-	ICU9	ICU10	-
TC4Ex	ICU0*	ICU1*	ICU2*	-	-	-	ICU6	ICU7	-	ICU9	ICU10	-
TC4Px	ICU0*	ICU1*	-	-	-	-	ICU6	ICU7	-	-	ICU10	-
TC4Dx	ICU0*	ICU1*	ICU2*	ICU3*	ICU4*	ICU5*	ICU6	ICU7	ICU8	-	ICU10	-
TC4Zx	ICU0*	ICU1*	ICU2*	ICU3*	ICU4*	ICU5*	ICU6	ICU7	ICU8	-	-	-
TC48x	ICU0*	ICU1*	ICU2*	ICU3*	-	-	ICU6	ICU7	-	-	-	-
TC44x	ICU0*	ICU1*	ICU2*	-	-	-	ICU6	ICU7	-	-	-	-
TC42x	ICU0*	ICU1*	-	-	-	-	ICU6	ICU7	-	-	-	-
TC45x	ICU0	ICU1	-	-	-	-	ICU6	ICU7	-	-	ICU10	ICU11

Software interrupt groups (GPSRGxSRy, SRBx, SRBx_ACCEN, GPSRGx_SWCy, GPSRGx_SWCACCENy)

The following table shows the relation between available ICUz and the implemented general purpose software interrupt groups where each group includes the registers GPRGxSRy, SRBx, GPSRGx_SWCy and GPSRGx_SWCACCENy.

The GPSRG groups can be used freely by the application software, GPSRs can be freely assigned to any implemented ISPz.

The information can help to decide: which GPSRGx should software use for a particular ISP to create a portable SW.

- Example: GPSRG7 and related registers are available in all products with a PPU
- Example: GPSRG5 and related registers is available in all products with a CPU5

Table 1915 Relationship between available ICUz and implemented GPSRGx resources

ISPz	CPU0	CPU1	CPU2	CPU3	CPU4	CPU5	CPUCS	SDMA 0	SDMA 1	GTM	PPU	RDMA
GPSRGx	x=0	x=1	x=2	x=3	x=4	x=5	x=6	-	-	-	x=7	-

Dedicated and shared ICU arbitration

The following table is showing which ICU is implemented with a dedicated arbitration (marked with D in the table below) and which ICU pair is implemented with a shared arbitration of the pending service requests assigned to the two respective ICUs:

- D means that there is a dedicated arbitration for this one ICU
- Sx means that the two ICUs with the same x value share an arbitration mechanism

An ICU related to a CPU is always implemented with a dedicated arbitration mechanism each.

ICU´s related to DMA functions, PPU and GTM are implemented with shared arbitration mechanisms where possible. Because DMA´s and the GTM acknowledge an arbitration round winner immediately the side effect in between the paired ICU´s is limited.

Shared arbitration for an ICU couple means:

- as long as service requests pending for ICUx and ICUy the mechanism arbitrates alternately for ICUx and ICUy in each f_{SPB} clock cycle, means that the worst case arbitration latency per arbitration round is delayed by one f_{SPB} clock cycle
- as long service requests pending only for ICUx the mechanism arbitrates permanently for ICUx in each cycle, same true for ICUy
- as long ICUx asserts its ENTER signal, the mechanism does not arbitrates anymore for ICUx and can permanently arbitrate for ICUy

Table 1916 ICU with dedicated and shared arbitration

	CPU0	CPU1	CPU2	CPU3	CPU4	CPU5	CPUCS	SDMA0	SDMA1	GTM	PPU	RDMA
TC49xN	D	D	D	D	D	-	S0	S1	D	S1	S0	-
TC46x	D	D	D	D	-	-	S0	S1	-	S1	S0	-
TC4Ex	D	D	D	-	-	-	S0	S1	-	S1	S0	-
TC4Px	D	D	-	-	-	-	S0	D	-	-	S0	-
TC4Dx	D	D	D	D	D	D	S0	S1	S1	-	S0	-
TC4Zx	D	D	D	D	D	D	S0	D	S0	-	-	-
TC48x	D	D	D	D	-	-	S0	S0	-	-	-	-
TC44x	D	D	D	-	-	-	S0	S0	-	-	-	-
TC42x	D	D	-	-	-	-	S0	S0	-	-	-	-
TC45x	D	D	-	-	-	-	S0	S1	-	-	S0	S1

40.4 Registers

40.4.1 Registers of INT

40.4.1.1 Register overview - access mode glossary

Table 1917 Register overview - access mode glossary

Keyword	Description
CSE	Access protection using PROT register PROTCSE.
SE	Access protection using PROT register PROTSE.
TOSz (z=0-14)	Access protection using PROT register PROTTOSz.
APU-PCS	Protection group consisting of registers ACCENCS_WRA, ACCENCS_WRB, ACCENCS_RDA, ACCENCS_RDB, ACCENCS_VM, ACCENCS_PRS.
PCS	Access protection using *APU-PCS* registers.
APU-PDBG	Protection group consisting of registers ACCENDBG_WRA, ACCENDBG_WRB, ACCENDBG_RDA, ACCENDBG_RDB, ACCENDBG_VM, ACCENDBG_PRS.
PDBG	Access protection using *APU-PDBG* registers.
APU-PSCFGz (z=0-14)	Protection group consisting of registers TOSz_ACCENSCFG_WRA, TOSz_ACCENSCFG_WRB, TOSz_ACCENSCFG_RDA, TOSz_ACCENSCFG_RDB, TOSz_ACCENSCFG_VM, TOSz_ACCENSCFG_PRS.
PSCFGz	Access protection using *APU-PSCFGz* registers.
APU-PSCTRLz (z=0-14)	Protection group consisting of registers TOSz_ACCENSCTRL_WRA, TOSz_ACCENSCTRL_WRB, TOSz_ACCENSCTRL_RDA, TOSz_ACCENSCTRL_RDB, TOSz_ACCENSCTRL_VM, TOSz_ACCENSCTRL_PRS.
PSCTRLz	Access protection using *APU-PSCTRLz* registers.
APU-PSRBy (y=0-15)	Protection group consisting of registers ACCENSRBy_WRA, ACCENSRBy_WRB, ACCENSRBy_RDA, ACCENSRBy_RDB, ACCENSRBy_VM, ACCENSRBy_PRS.
PSRBy	Access protection using *APU-PSRBy* registers.
BE	Always returns a Bus Error.
U	No access restrictions.
PROT	Access restrictions as defined in the PROT register access rules.
M	Indicates a module specific access condition. Refer to the register description for details of the specific access condition.

40.4.1.2 Registers overview - INT (ascending offset address)

Table 1918 Registers overview - INT (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
OCS	OCDS control and status register	0004_H	PDBG	SV, PDBG	Debug Reset	6902

(table continues...)

Table 1918 (continued) Registers overview - INT (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ID	Module identification register	0008_H	PDBG	BE	Application Reset	6903
LCLTEST	LCL test register	$000C_H$	PDBG	SV, SE, PDBG	Application Reset	6903
OIXMS	OTGB IRQ MUX missed IRQ select	0010_H	PDBG	SV, PDBG	Debug Reset	6904
OIXS0	OTGB IRQ MUX select 0	0014_H	PDBG	SV, PDBG	Debug Reset	6904
OIXS1	OTGB IRQ MUX select 1	0018_H	PDBG	SV, PDBG	Debug Reset	6905
OIT	OTGB IRQ trace	$001C_H$	PDBG	SV, PDBG	Debug Reset	6905
PROTSE	PROT register safe endinit	0020_H	U	SV, PROT	Application Reset	6907
PROTCSE	PROT register cyber security endinit	0024_H	U	SV, PROT	Application Reset	6909
PROTTOSz	PROT register for TOS=z	0030_H+z *4	U	SV, PROT	Application Reset	6910
ACCENCS_WRA	CS write access enable register A	0080_H	U	CSE, SV	Application Reset	6912
ACCENCS_WRB	CS write access enable register B	0084_H	U	CSE, SV	Application Reset	6913
ACCENCS_RDA	CS read access enable register A	0088_H	U	CSE, SV	Application Reset	6913
ACCENCS_RDB	CS read access enable register B	$008C_H$	U	CSE, SV	Application Reset	6914
ACCENCS_VM	CS VM access enable register	0090_H	U	CSE, SV	Application Reset	6914
ACCENCS_PRS	CS PRS access enable register	0094_H	U	CSE, SV	Application Reset	6915
ACCENDBG_WRA	DBG write access enable register A	$00A0_H$	U	SE, SV	Application Reset	6915
ACCENDBG_WRB	DBG write access enable register B	$00A4_H$	U	SE, SV	Application Reset	6916
ACCENDBG_RDA	DBG read access enable register A	$00A8_H$	U	SE, SV	Application Reset	6916
ACCENDBG_RDB	DBG read access enable register B	$00AC_H$	U	SE, SV	Application Reset	6917
ACCENDBG_VM	DBG VM access enable register	$00B0_H$	U	SE, SV	Application Reset	6917
ACCENDBG_PRS	DGB PRS access enable register	$00B4_H$	U	SE, SV	Application Reset	6918
ACCENSRBy_WRA	SRBy write access enable register A	0100_H+y *18_H	U	SE, SV	Application Reset	6919

(table continues...)

Table 1918 (continued) Registers overview - INT (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ACCENSRBy_WRB	SRBy write access enable register B	0104_H+y *18_H	U	SE, SV	Application Reset	6919
ACCENSRBy_RDA	SRBy read access enable register A	0108_H+y *18_H	U	SE, SV	Application Reset	6920
ACCENSRBy_RDB	SRBy read access enable register B	$010C_H$+y *18_H	U	SE, SV	Application Reset	6920
ACCENSRBy_VM	SRBy VM access enable register	0110_H+y *18_H	U	SE, SV	Application Reset	6921
ACCENSRBy_PRS	SRBy PRS access enable register	0114_H+y *18_H	U	SE, SV	Application Reset	6921
TOSz_ACCENSCFG_WRA	SRC[15:0] write access enable register A for TOS=z	0300_H+z *30_H	U	TOSz, SV	Application Reset	6922
TOSz_ACCENSCFG_WRB	SRC[15:0] write access enable register B for TOS=z	0304_H+z *30_H	U	TOSz, SV	Application Reset	6923
TOSz_ACCENSCFG_RDA	SRC[15:0] read access enable register A for TOS=z	0308_H+z *30_H	U	TOSz, SV	Application Reset	6923
TOSz_ACCENSCFG_RDB	SRC[15:0] read access enable register B for TOS=z	$030C_H$+z *30_H	U	TOSz, SV	Application Reset	6924
TOSz_ACCENSCFG_VM	SRC[15:0] VM access enable register for TOS=z	0310_H+z *30_H	U	TOSz, SV	Application Reset	6924
TOSz_ACCENSCFG_PRS	SRC[15:0] PRS access enable register for TOS=z	0314_H+z *30_H	U	TOSz, SV	Application Reset	6925
TOSz_ACCENSCTRL_WRA	SRC[31:16] write access enable register A for TOS=z	0318_H+z *30_H	U	TOSz, SV	Application Reset	6925
TOSz_ACCENSCTRL_WRB	SRC[31:16] write access enable register B for TOS=z	$031C_H$+z *30_H	U	TOSz, SV	Application Reset	6926
TOSz_ACCENSCTRL_RDA	SRC[31:16] read access enable register A for TOS=z	0320_H+z *30_H	U	TOSz, SV	Application Reset	6926
TOSz_ACCENSCTRL_RDB	SRC[31:16] read access enable register B for TOS=z	0324_H+z *30_H	U	TOSz, SV	Application Reset	6927
TOSz_ACCENSCTRL_VM	SRC[31:16] VM access enable register for TOS=z	0328_H+z *30_H	U	TOSz, SV	Application Reset	6927
TOSz_ACCENSCTRL_PRS	SRC[31:16] PRS access enable register for TOS=z	$032C_H$+z *30_H	U	TOSz, SV	Application Reset	6928
GPSRGx_SWCACCENy	GPRSGx_SWCy write access protection register	0700_H+x *40_H+y* 4	M	SV, M	Application Reset	6929
GPSRGx_SWCy	SW control register for GPSRGxSRy	0720_H+x *40_H+y* 4	M	SV, M	Application Reset	6931
SRBx	Service request broadcast register x	$0B00_H$+x *4	PSRBx	SV, PSRBx	Application Reset	6933

(table continues...)

Table 1918 (continued) Registers overview - INT (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
LWSRz_VMy	ICU latest service request information signaled for VMy	$0C00_H$+z *34_H+y* 4	PSCTRLz	BE	Application Reset	6933
LASRz	ICU Last acknowledged service request register	$0C20_H$+z *34_H	PSCTRLz	BE	Application Reset	6934
ECRz	ICU error capture register	$0C24_H$+z *34_H	PSCFGz	BE	Application Reset	6935
ECTRLz	ICU error control register	$0C28_H$+z *34_H	PSCFGz	SV, PSCFGz	Application Reset	6936
EGENz	ICU error generation register	$0C2C_H$+z *34_H	PSCFGz	SV, PSCFGz	Application Reset	6937
VMENz	ICU VM control register	$0C30_H$+z *34_H	PSCFGz	SV, PSCFGz	Application Reset	6938

40.4.1.3 OCDS control and status register

The OCDS Control and Status register OCS controls the debug and trace behavior by selecting suspend modes and OTGB trigger sets. When OCDS is disabled the suspend control is ineffective.

OCS Offset address: 0004_H

OCDS control and status register Debug Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												TG_P	TGB	TGS	
r												w	rw	rw	

Field	Bits	Type	Description
TGS	1:0	rw	**Trigger Set for OTGB0 and OTGB1** 00_B No Trigger Set output 01_B Trigger Set 1: ´TS16_IS Interrupt Selection´ selected 10_B Trigger Set 2: ´TS16_SPA Service Provider Activity´ selected 11_B Trigger Set 3: ´TS16_SP Service Provider´ selected
TGB	2	rw	**OTGB0 and OTGB1 Bus Select** 0_B Trigger Set is output on OTGB0 1_B Trigger Set is output on OTGB1
TG_P	3	w	**TGS, TGB Write Protection** TGS and TGB are only written when TG_P is 1, otherwise unchanged. Read as 0.
0	31:4	r	**Reserved-0** Read as 0; should be written with 0.

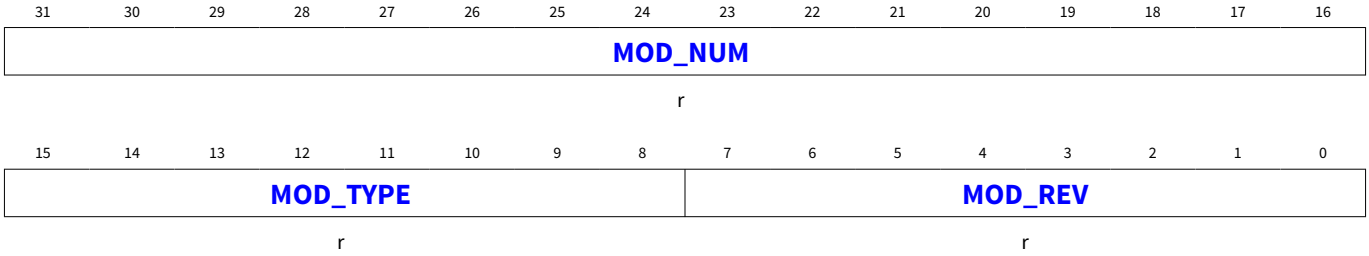
Table 1919 Access mode restrictions of OCS sorted by descending priority

Mode name	Access mode		Description
write 1 to .**TG_P**	rw	TGB, TGS	Set TG_P during write access
(default)	r	TGB, TGS	

40.4.1.4 Module identification register

ID Offset address: 0008_H

Module identification register Application Reset value: $00B9\ C013_H$



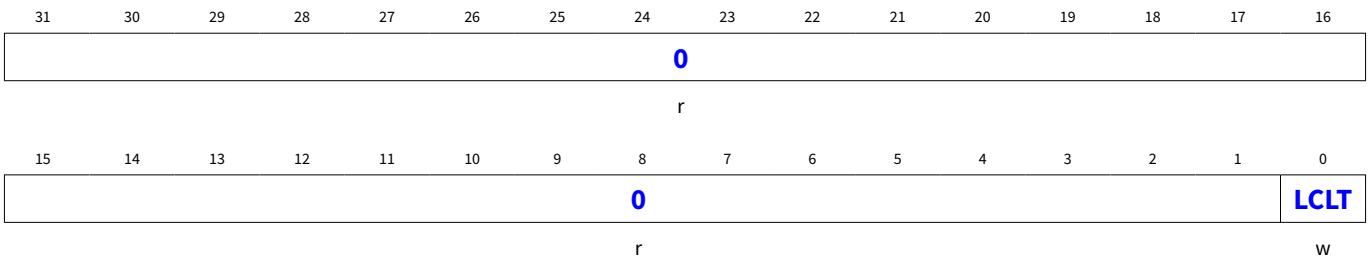
Field	Bits	Type	Description
MOD_REV	7:0	r	**Module Revision** Indicates the revision number of the implementation.
MOD_TYPE	15:8	r	**Module Type** This internal marker is fixed to $C0_H$.
MOD_NUM	31:16	r	**Module Number** Indicates the module identification number.

40.4.1.5 LCL test register

Provides the capability for software to inject a fault condition into the comparators of each Lockstep Comparator Logic block. The implementation should generate a single cycle fault each time the bit is written with '1'.

LCLTEST Offset address: $000C_H$

LCL test register Application Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
LCLT	0	w	**LCL Lockstep Test** Fault injection for LCL. Reads as zero. 0_B No action 1_B Inject single fault in the LCL
0	31:1	r	**Reserved** Read as 0; should be written with 0.

40.4.1.6 OTGB IRQ MUX missed IRQ select

OIXMS Offset address: 0010_H

OTGB IRQ MUX missed IRQ select Debug Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0					MIRQ										
r					rw										

Field	Bits	Type	Description
MIRQ	10:0	rw	**SRN Index for Missed Interrupt Trigger**
0	31:11	r	**Reserved-0** Read as 0; must be written with 0.

40.4.1.7 OTGB IRQ MUX select 0

OIXS0 Offset address: 0014_H

OTGB IRQ MUX select 0 Debug Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0					IRQ1										
r					rw										

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0					IRQ0										
r					rw										

Field	Bits	Type	Description
IRQ0	10:0	rw	**SRN Index for Interrupt Trigger 0**
IRQ1	26:16	rw	**SRN Index for Interrupt Trigger 1**
0	15:11, 31:27	r	**Reserved** Read as 0; must be written with 0.

40.4.1.8 OTGB IRQ MUX select 1

OIXS1 Offset address: 0018_H

OTGB IRQ MUX select 1 Debug Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0					IRQ3										
r					rw										

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0					IRQ2										
r					rw										

Field	Bits	Type	Description
IRQ2	10:0	rw	**SRN Index for Interrupt Trigger 2**
IRQ3	26:16	rw	**SRN Index for Interrupt Trigger 3**
0	15:11, 31:27	r	**Reserved-0** Read as 0; must be written with 0.

40.4.1.9 OTGB IRQ trace

OIT Offset address: $001C_H$

OTGB IRQ trace Debug Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OE1	0			TOS1				OE0	0			TOS0			
rw	r			rw				rw	r			rw			

Field	Bits	Type	Description
TOS0	3:0	rw	**Selection of the ISP for observation through OTGB0.** Trigger Set TS16_SP Selection of the ISP for observation through OTGB1 whether SRs are pending for it and the observation of acknoweledged SRs. Family concept encoding, compatible with SRC.TOS Note: In products where the CPUx, DMA1, GTM, PPU, RDMA is not implemented, the related TOS encoding will not be used and treated as RESERVED. 0_H CPU0 is selected 1_H CPU1 is selected 2_H CPU2 is selected 3_H CPU3 is selected 4_H CPU4 is selected 5_H CPU5 is selected 6_H CPUCS is selected 7_H DMA0 is selected 8_H DMA1 is selected 9_H GTM is selected A_H PPU is selected B_H RDMA is selected **others,** Reserved
OE0	7	rw	**Output Enable for OTGB0** 0_B Disabled 1_B Enabled
TOS1	11:8	rw	**Selection of the ISP for observation through OTGB1.** Trigger Set TS16_SP Selection of the ISP for observation through OTGB1 whether SRs are pending for it and the observation of acknoweledged SRs. Family concept encoding, compatible with SRC.TOS Note: In products where the CPUx, DMA1, GTM, PPU, RDMA is not implemented, the related TOS encoding will not be used and treated as RESERVED. 0_H CPU0 is selected 1_H CPU1 is selected 2_H CPU2 is selected 3_H CPU3 is selected 4_H CPU4 is selected 5_H CPU5 is selected 6_H CPUCS is selected 7_H DMA0 is selected 8_H DMA1 is selected 9_H GTM is selected A_H PPU is selected B_H RDMA is selected **others,** Reserved

(table continues...)

(continued)

Field	Bits	Type	Description
OE1	15	rw	**Output Enable for OTGB1** 0_B Disabled 1_B Enabled
0	6:4, 14:12, 31:16	r	**Reserved-0** Read as 0; should be written with 0.

40.4.1.10 PROT register safe endinit

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner and in some PROT states, the secure master to update the PROT state as described in the PROT mechanism.

The PROTSE register controls lock and unlock of the local safe endinit (SE) protected control registers.

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner and in some PROT states, the secure master to update the PROT state as described in the PROT mechanism

PROTSE Offset address: 0020_H

PROT register safe endinit Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OWEN	ODEF	TAGID						PRSEN	PRS			VMEN	VM		
w	rw	rw						rw	rw			rw	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												SWEN	STATE		
r												w	rwh		

Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write

(table continues...)

(continued)

Field	Bits	Type	Description
VM	18:16	rw	**Virtual machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual machine definition enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.
PRSEN	23	rw	**Protection set definition enable for PROT owner** 0_B **Disable**: PRS is not part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG-ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; should be written with 0.

Table 1920 Access mode restrictions of PROTSE sorted by descending priority

Mode name	Access mode		Description
write 1 to .**SWEN**	rwh	STATE	
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

40.4.1.11 PROT register cyber security endinit

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner and in some PROT states, the secure master to update the PROT state as described in the PROT mechanism.

The PROTCSE register controls write access to the local Cyber Secuirty Endinit (CSE) protected control registers.

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner and in some PROT states, the secure master to update the PROT state as described in the PROT mechanism

PROTCSE Offset address: 0024_H

PROT register cyber security endinit Application Reset value: $4C00\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OWEN	ODEF	TAGID						PRSEN	PRS			VMEN	VM		
w	rw	rw						rw	rw			rw	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												SWEN	STATE		
r												w	rwh		

Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
VM	18:16	rw	**Virtual machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual machine definition enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.

(table continues...)

(continued)

Field	Bits	Type	Description
PRSEN	23	rw	**Protection set definition enable for PROT owner** 0_B **Disable**: PRS is not part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG-ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; should be written with 0.

Table 1921 Access mode restrictions of PROTCSE sorted by descending priority

Mode name	Access mode		Description
write 1 to .**SWEN**	rwh	STATE	
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

40.4.1.12 PROT register for TOS=z

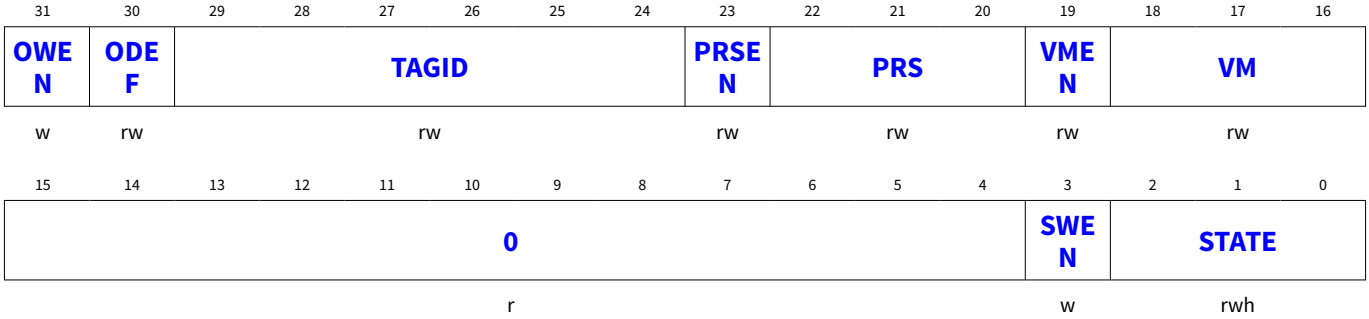
The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner and in some PROT states, the secure master to update the PROT state as described in the PROT mechanism.

The PROTTOSz register controls lock and unlock of the TOS control and configuration access protection registers.

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner and in some PROT states, the secure master to update the PROT state as described in the PROT mechanism

PROTTOSz (z=0-14) Offset address: 0030_H+z*4

PROT register for TOS=z Application Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
VM	18:16	rw	**Virtual machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual machine definition enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.
PRSEN	23	rw	**Protection set definition enable for PROT owner** 0_B **Disable**: PRS is not part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG-ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner

(table continues...)

(continued)

Field	Bits	Type	Description
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; should be written with 0.

Table 1922 Access mode restrictions of PROTTOSz (z=0-14) sorted by descending priority

Mode name	Access mode		Description
write 1 to .**SWEN**	rwh	STATE	
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

40.4.1.13 CS write access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

ACCENCS_WRA Offset address: 0080_H

CS write access enable register A Application Reset value: $0000\ 1000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	**EN30**	**EN29**	**EN28**	**EN27**	**EN26**	**EN25**	**EN24**	**EN23**	**EN22**	**EN21**	**EN20**	**EN19**	**EN18**	**EN17**	**EN16**
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	**EN14**	**EN13**	**EN12**	**EN11**	**EN10**	**EN09**	**EN08**	**EN07**	**EN06**	**EN05**	**EN04**	**EN03**	**EN02**	**EN01**	**EN00**
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q 0_B Disabled for write access 1_B Enabled for write access

40.4.1.14 CS write access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

ACCENCS_WRB Offset address: 0084_H

CS write access enable register B Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q 0_B Disabled for write access 1_B Enabled for write access
0	31:8	r	**Reserved** Read as 0; should be written with 0.

40.4.1.15 CS read access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

ACCENCS_RDA Offset address: 0088_H

CS read access enable register A Application Reset value: $0000\ 1000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q 0_B Disabled for read access 1_B Enabled for read access

40.4.1.16 CS read access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

ACCENCS_RDB — Offset address: $008C_H$

CS read access enable register — Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q 0_B Disabled for read access 1_B Enabled for read access
0	31:8	r	**Reserved** Read as 0; should be written with 0.

40.4.1.17 CS VM access enable register

This register defines which virtual machine encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each VMx.

ACCENCS_VM — Offset address: 0090_H

CS VM access enable register — Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for VM ID q** This bit enables read access to the access protected region for transactions with the VM ID q. 0_B Disabled for read access 1_B Enabled for read access

(table continues...)

(continued)

Field	Bits	Type	Description
WRq (q=00-07)	q+16	rw	**Write access enable for VM ID q** This bit enables write access to the access protected region for transactions with the VM ID q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

40.4.1.18 CS PRS access enable register

This register defines which protection register set encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each PRSx.

ACCENCS_PRS Offset address: 0094_H

CS PRS access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for PRS q** This bit enables read access to the access protected region for transactions with the PRS q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for PRS q** This bit enables write access to the access protected region for transactions with the PRS q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

40.4.1.19 DBG write access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

ACCENDBG_WRA Offset address: $00A0_H$

DBG write access enable register A Application Reset value: $1000\ 0003_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q 0_B Disabled for write access 1_B Enabled for write access

40.4.1.20 DBG write access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

ACCENDBG_WRB Offset address: $00A4_H$

DBG write access enable register B Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q 0_B Disabled for write access 1_B Enabled for write access
0	31:8	r	**Reserved** Read as 0; should be written with 0.

40.4.1.21 DBG read access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

ACCENDBG_RDA Offset address: $00A8_H$

DBG read access enable register A Application Reset value: $FFFF\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q 0_B Disabled for read access 1_B Enabled for read access

40.4.1.22 DBG read access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

ACCENDBG_RDB Offset address: $00AC_H$

DBG read access enable register B Application Reset value: $0000\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q 0_B Disabled for read access 1_B Enabled for read access
0	31:8	r	**Reserved** Read as 0; should be written with 0.

40.4.1.23 DBG VM access enable register

This register defines which virtual machine encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each VMx.

ACCENDBG_VM Offset address: $00B0_H$

DBG VM access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for VM ID q** This bit enables read access to the access protected region for transactions with the VM ID q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for VM ID q** This bit enables write access to the access protected region for transactions with the VM ID q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

40.4.1.24 DGB PRS access enable register

This register defines which protection register set encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each PRSx.

ACCENDBG_PRS Offset address: $00B4_H$

DGB PRS access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for PRS q** This bit enables read access to the access protected region for transactions with the PRS q. 0_B Disabled for read access 1_B Enabled for read access

(table continues...)

(continued)

Field	Bits	Type	Description
WRq (q=00-07)	q+16	rw	**Write access enable for PRS q** This bit enables write access to the access protected region for transactions with the PRS q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

40.4.1.25 SRBy write access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

ACCENSRBy_WRA (y=0-15) Offset address: 0100_H+y*18_H

SRBy write access enable register A Application Reset value: $1000\ 0003_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q 0_B Disabled for write access 1_B Enabled for write access

40.4.1.26 SRBy write access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

ACCENSRBy_WRB (y=0-15) Offset address: 0104_H+y*18_H

SRBy write access enable register B Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q 0_B Disabled for write access 1_B Enabled for write access
0	31:8	r	**Reserved** Read as 0; should be written with 0.

40.4.1.27 SRBy read access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

ACCENSRBy_RDA (y=0-15) Offset address: 0108_H+y*18_H

SRBy read access enable register A Application Reset value: FFFF FFFF$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q 0_B Disabled for read access 1_B Enabled for read access

40.4.1.28 SRBy read access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

ACCENSRBy_RDB (y=0-15) Offset address: $010C_H+y*18_H$

SRBy read access enable register B Application Reset value: 0000 00FF$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q 0_B Disabled for read access 1_B Enabled for read access
0	31:8	r	**Reserved** Read as 0; should be written with 0.

40.4.1.29 SRBy VM access enable register

This register defines which virtual machine encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each VMx.

ACCENSRBy_VM (y=0-15) Offset address: 0110_H+y*18_H

SRBy VM access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for VM ID q** This bit enables read access to the access protected region for transactions with the VM ID q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for VM ID q** This bit enables write access to the access protected region for transactions with the VM ID q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

40.4.1.30 SRBy PRS access enable register

This register defines which protection register set encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each PRSx.

ACCENSRBy_PRS (y=0-15) Offset address: 0114_H+y*18_H

SRBy PRS access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for PRS q** This bit enables read access to the access protected region for transactions with the PRS q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for PRS q** This bit enables write access to the access protected region for transactions with the PRS q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

40.4.1.31 SRC[15:0] write access enable register A for TOS=z

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

TOSz_ACCENSCFG_WRA (z=0-14) Offset address: 0300_H+z*30_H

SRC[15:0] write access enable register A for TOS=z Application Reset value: $1000\ 0003_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q 0_B Disabled for write access 1_B Enabled for write access

40.4.1.32 SRC[15:0] write access enable register B for TOS=z

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

TOSz_ACCENSCFG_WRB (z=0-14) Offset address: 0304_H+z*30_H

SRC[15:0] write access enable register B for TOS=z Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q 0_B Disabled for write access 1_B Enabled for write access
0	31:8	r	**Reserved** Read as 0; should be written with 0.

40.4.1.33 SRC[15:0] read access enable register A for TOS=z

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

TOSz_ACCENSCFG_RDA (z=0-14) Offset address: 0308_H+z*30_H

SRC[15:0] read access enable register A for TOS=z Application Reset value: $FFFF\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q 0_B Disabled for read access 1_B Enabled for read access

40.4.1.34 SRC[15:0] read access enable register B for TOS=z

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

TOSz_ACCENSCFG_RDB (z=0-14) Offset address: $030C_H+z*30_H$

SRC[15:0] read access enable register B for TOS=z Application Reset value: $0000\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q 0_B Disabled for read access 1_B Enabled for read access
0	31:8	r	**Reserved** Read as 0; should be written with 0.

40.4.1.35 SRC[15:0] VM access enable register for TOS=z

This register defines which virtual machine encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each VMx.

TOSz_ACCENSCFG_VM (z=0-14) Offset address: 0310_H+z*30_H

SRC[15:0] VM access enable register for TOS=z Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for VM ID q** This bit enables read access to the access protected region for transactions with the VM ID q. 0_B Disabled for read access 1_B Enabled for read access

(table continues...)

(continued)

Field	Bits	Type	Description
WRq (q=00-07)	q+16	rw	**Write access enable for VM ID q** This bit enables write access to the access protected region for transactions with the VM ID q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

40.4.1.36 SRC[15:0] PRS access enable register for TOS=z

This register defines which protection register set encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each PRSx.

TOSz_ACCENSCFG_PRS (z=0-14) Offset address: 0314_H+z*30_H

SRC[15:0] PRS access enable register for TOS=z Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for PRS q** This bit enables read access to the access protected region for transactions with the PRS q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for PRS q** This bit enables write access to the access protected region for transactions with the PRS q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

40.4.1.37 SRC[31:16] write access enable register A for TOS=z

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

TOSz_ACCENSCTRL_WRA (z=0-14) Offset address: 0318_H+z*30_H

SRC[31:16] write access enable register A for TOS=z Application Reset value: $1000\ 0003_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q 0_B Disabled for write access 1_B Enabled for write access

40.4.1.38 SRC[31:16] write access enable register B for TOS=z

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

TOSz_ACCENSCTRL_WRB (z=0-14) Offset address: $031C_H+z*30_H$

SRC[31:16] write access enable register B for TOS=z Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q 0_B Disabled for write access 1_B Enabled for write access
0	31:8	r	**Reserved** Read as 0; should be written with 0.

40.4.1.39 SRC[31:16] read access enable register A for TOS=z

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

TOSz_ACCENSCTRL_RDA (z=0-14) Offset address: 0320_H+z*30_H

SRC[31:16] read access enable register A for TOS=z Application Reset value: $FFFF\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q 0_B Disabled for read access 1_B Enabled for read access

40.4.1.40 SRC[31:16] read access enable register B for TOS=z

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

TOSz_ACCENSCTRL_RDB (z=0-14) Offset address: 0324_H+z*30_H

SRC[31:16] read access enable register B for TOS=z Application Reset value: $0000\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q 0_B Disabled for read access 1_B Enabled for read access
0	31:8	r	**Reserved** Read as 0; should be written with 0.

40.4.1.41 SRC[31:16] VM access enable register for TOS=z

This register defines which virtual machine encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each VMx.

TOSz_ACCENSCTRL_VM (z=0-14) Offset address: 0328_H+z*30_H

SRC[31:16] VM access enable register for TOS=z Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for VM ID q** This bit enables read access to the access protected region for transactions with the VM ID q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for VM ID q** This bit enables write access to the access protected region for transactions with the VM ID q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

40.4.1.42 SRC[31:16] PRS access enable register for TOS=z

This register defines which protection register set encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each PRSx.

TOSz_ACCENSCTRL_PRS (z=0-14) Offset address: $032C_H+z*30_H$

SRC[31:16] PRS access enable register for TOS=z Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for PRS q** This bit enables read access to the access protected region for transactions with the PRS q. 0_B Disabled for read access 1_B Enabled for read access

(table continues...)

(continued)

Field	Bits	Type	Description
WRq (q=00-07)	q+16	rw	**Write access enable for PRS q** This bit enables write access to the access protected region for transactions with the PRS q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

40.4.1.43 GPRSGx_SWCy write access protection register

GPSRGx_SWCACCENy controls the write access protection for the related GPSRGx_SWCy register.

It defines which CPU TAG-IDs, Debug TAG-ID, VM or PRS encodings are enabled for write access to GPSRGx_SWCy.

Bits referring to TAG-IDs of unimplemented CPUs have no effect as the TAG-IDs are not used in the system.

Access Protection for access to the GPSRGx_SWCACCENy register:

- it is write protected by PROTTOSz.
- it is read protected TOSz_ACCENSCFG.
- TOSz_ACCENSCFG and PROTTOSz are selected by the GPSRGxSRy.TOSz configuration.

For more details see IR chapter General Purpose Service Requests (GPSRGxSRy).

GPSRGx_SWCACCENy (x=0-15;y=0-7) Offset address: 0700_H+x*40_H+y*4

GPRSGx_SWCy write access protection register Application Reset value: $00FF\ 3FFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
VM_PRS_CTRL	0							VM_PRS7	VM_PRS6	VM_PRS5	VM_PRS4	VM_PRS3	VM_PRS2	VM_PRS1	VM_PRS0
rw	r							rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	CBS_IOCE	CPUCS_DS	CPUCS_D	CPU5_DS	CPU5_D	CPU4_DS	CPU4_D	CPU3_DS	CPU3_D	CPU2_DS	CPU2_D	CPU1_DS	CPU1_D	CPU0_DS	CPU0_D
r	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
CPUz_D (z=0-5)	2*z	rw	**Access enable for CPUz DMI non-safe TAG-ID** This bit enables read and write access to the GPSR_SWCxy for the DMI-D TAG-ID of CPUz 0_B Access will not be executed 1_B Access will be executed

(table continues...)

(continued)

Field	Bits	Type	Description
CPUz_DS (z=0-5)	2*z+1	rw	**Access enable for CPUz DMI Safe TAG-ID** This bit enables read and write access to the GPSR_SWCxy for the DMI-DS TAG-ID of CPUz 0_B Access will not be executed 1_B Access will be executed
CPUCS_D	12	rw	**Access enable for CSRM_CPU DMI non-safe TAG-ID** This bit enables read and write access to the GPSR_SWCxy for the DMI-D TAG-ID of the CSRM_CPU 0_B Access will not be executed 1_B Access will be executed
CPUCS_DS	13	rw	**Access enable for CSRM_CPU DMI Safe TAG-ID** This bit enables read and write access to the GPSR_SWCxy for the DMI-DS TAG-ID of CPUz 0_B Access will not be executed 1_B Access will be executed
CBS_IOCE	14	rw	**Access enable for Cerberus and IOCE TAG-ID** This bit enables read and write access to the GPSR_SWCxy for Cerberus 0_B Access will not be executed 1_B Access will be executed
VM_PRSi (i=0-7)	i+16	rw	**Access enable for VM or PRS encoding i** Access enable for VM or PRS coding i (read and write). Whether the VM_PRS register is used for the VM check (for access with VM_VALID=1) or for the PRS check (for access with PRS_VALID_1) is configured through the VM_PRS_CTRL bit. 0_B Disabled for read and write access 1_B Enabled for read and write access
VM_PRS_CTRL	31	rw	**VM_PRS control bit** This bit controls whether the VM_PRS register is used for the VM check (for access with VM_VALID=1) or for the PRS check (for access with PRS_VALID=1) 0_B VM_PRS register is used for the VM check 1_B VM_PRS register is used for the PRS check
0	15, 30:24	r	**Reserved-0** Read as 0; should be written with 0.

40.4.1.44 SW control register for GPSRGxSRy

The GPSRx_SWCy can be used to control the associated GPSRGxSRy (GPSR group x, node y) and to to disable GPSRGxSRy interrupt trigger through SRBx.TRIGx.

The GPSRx_SWCy register is write protected by the related GPSRGx_SWCACCENy register and read protected by the TOSz_ACCENCFG where 'z' is determined by the configuration of the corresponding GPSRGxSRy node (GPSRGxSRy.TOS=z).

The GPSRx_SWCy.LOCKSTAT bit can only be cleared by a master agent function enabled for write access to GPSRGx.SRy and enabled for write access in GPSRGx_SWCACCENy.

GPSRGx_SWCy (x=0-15;y=0-7) Offset address: 0720_H+x*40_H+y*4

SW control register for GPSRGxSRy Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
IOVCLR	IOV	SETR	SRR	BRDIS	0								LOCKSTAT	LOCKCLR	LOCKSET
w	rh	w	rh	rw	r								rh	w	w

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA															
rw															

Field	Bits	Type	Description
DATA	15:0	rw	**DATA** Data can be used to transfer additional information from the SW interrupt initiator to the ISP. DATA is protected against unintentional overwriting through LOCKSET. The ISP can read the information from the DATA register and then release the register again via LOCKCLR. Write: DATA is updated with the written value Read: Returns DATA Note: DATA can only be updated if LOCKSTAT=0 and LOCKSET is written with 1 in parallel. Data can only be updated by writing to the complete DATA 16-bit field. Otherwise DATA will not be updated and there will be no notification.
LOCKSET	16	w	**Sets LOCK bit** The LOCKSET bit can be used to set the DATA lock status 0_B No action 1_B Sets LOCKSTAT=1; DATA is updated by this write if LOCKSTAT=0; no action if LOCKCLR is written with 1 at the same time; Read always returns 0
LOCKCLR	17	w	**Clears LOCK bit** The LOCKCLR bit can be used to reset the DATA lock status. 0_B No action 1_B Sets LOCKSTAT=0 if the master agent function writes that is defined in the related GPSRGxSRy SRC register; no action if LOCKSET is written with 1 at the same time; write from any other master has no effect; read always returns 0

(table continues...)

(continued)

Field	Bits	Type	Description
LOCKSTAT	18	rh	**Status of DATA lock** 0_B DATA is not locked, can be updated 1_B DATA is locked, can not be updated
BRDIS	27	rw	**Broadcast disable** The BRDIS bit can be used to disable the GPRS node control through the broadcast register. If disabled (GPSRGx_SWCy.BRDIS=1), SRBx.TRIGy can not trigger GPSRGxSRy (GPSR group x, SRN y). 0_B SRBx.TRIGy register bit is enabled 1_B SRBx.TRIGy register bit is disable
SRR	28	rh	**Service request status** Signals status of register bit GPSRGxSRy.SRR 0_B Service request not pending (SRR=0) 1_B Service request pending (SRR=1)
SETR	29	w	**Service request set** Sets register bit GPSRxSRy.SRR. 0_B No action 1_B Sets register bit GPSRxSRy.SRR=1; if already GPSRxSRy.SRR=1 then SRC_GPSRxSRy.IOV is set to 1; read always returns 0
IOV	30	rh	**Service request interupt overflow status** Signals the current status of register bit SRC_GPSRxSRy.IOV 0_B No interrupt overflow detected (status of register bit GPSRxSRy.IOV=0) 1_B Interrupt overflow detected (status of register bit is GPSRxSRy.IOV=1)
IOVCLR	31	w	**Service request interupt overflow clear** Resets the register bit SRC_GPSRxSRy.IOV 0_B No action 1_B Sets register bit GPSRGxSRy.IOV=0 but only if the master agent function writes that is defined in the related SRC_GPSRGySRy register; write from any other master has no effect; no action in case of an IOV condition for register bit GPSRGySRy at the same time; read always returns 0
0	26:19	r	**Reserved-0** Read as 0; should be written with 0.

Table 1923 Access mode restrictions of GPSRGx_SWCy (x=0-15;y=0-7) sorted by descending priority

Mode name	Access mode		Description
write 1 to .**LOCKSET** and .**LOCKSTAT** == 0	rw	DATA	Lockset must be written with 1
(default)	r	DATA	

40.4.1.45 Service request broadcast register x

Interrupt service request broadcast registers can be used to trigger multiple of the eight service request nodes in the related GPSRx group in parallel.

SRBx (x=0-15) Offset address: $0B00_H+x*4$

Service request broadcast register x Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								TRIG 7	TRIG 6	TRIG 5	TRIG 4	TRIG 3	TRIG 2	TRIG 1	TRIG 0
r								w	w	w	w	w	w	w	w

Field	Bits	Type	Description
TRIGi (i=0-7)	i	w	**General purpose service request trigger i** This bit is always read as 0. 0_B No effect 1_B Trigger the general purpose service request i in group x (GPSRxi)
0	31:8	r	**Reserved-0** Read as 0; should be written with 0.

40.4.1.46 ICU latest service request information signaled for VMy

LWSRz_VMy register contains the information of the Last winning service request (LWSR) for VMy that was sent last from ICUz to the connected interrupt service provider. Service request information in the LWSRz_VMy register is only valid if VALID=1 and STAT=1.

LWSRz_VMy (y=0-7;z=0-14) Offset address: $0C00_H+z*34_H+y*4$

ICU latest service request information signaled for VMy Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
STAT	0			CS	ID										
rh	r			rh	rh										

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		INVALID	VALID	0				PN							
r		rh	rh	r				rh							

Field	Bits	Type	Description
PN	7:0	rh	**Priority number information**
VALID	12	rh	**Status of signaled VALID** 0_B Signaled information not related to a pending service request. 1_B Signaled information related to a pending service request.

(table continues...)

(continued)

Field	Bits	Type	Description
INVALID	13	rh	**Status of signaled INVALID** 0_B Signaled information not related to a not pending service request. 1_B Received information related to a not pending service request.
ID	26:16	rh	**Index number information**
CS	27	rh	**Cyber security enable** 0_B Service request is not assigned to the security application 1_B Service request is assigned to the security application
STAT	31	rh	**LWSR register status** The STAT bit is set to 1 whenever ICU signals an SR with VALID. The LWSRx.STAT bit is cleared when the ISP ACKNOWLEDGE an service request for VMx with the ID and VM information matching LWSRx values. The LWSRx.STAT bit is also cleared if the ICU signals and INVALID for VMx. 0_B LWSR bit fields are not valid 1_B LWSR bit fields are valid
0	11:8, 15:14, 30:28	r	**Reserved-0** Read as 0; should be written with 0.

40.4.1.47 ICU Last acknowledged service request register

The Last acknowledged service request (LASR) register shows the information received with last Acknowledge by the interrupt service provider.

Exception: ENTER shows always the current state of the ISPs ENTER signal.

LASRz (z=0-14) Offset address: $0C20_H+z*34_H$

ICU Last acknowledged service request register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ENTER		VM		CS	ID										
rh		rh		rh	rh										

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		ECC						PN							
r		rh						rh							

Field	Bits	Type	Description
PN	7:0	rh	**Service request priority number** This bit field shows the service request priority number received with the last Acknowledge.

(table continues...)

(continued)

Field	Bits	Type	Description
ECC	13:8	rh	**Received ECC information** This bit field shows the ECC information received with the last Acknowledge.
ID	26:16	rh	**Service request node index number** This bit field shows the SRN index number received with the last Acknowledge.
CS	27	rh	**Cyber security information** 0_B Service request is not assigned to the security application 1_B Service request is assigned to the security application
VM	30:28	rh	**Virtual machine number**
ENTER	31	rh	**ENTER signal state** This bit field shows the current status of the ENTER signal. 0_B ISP is not in ENTER state. 1_B ISP is in ENTER state.
0	15:14	r	**Reserved-0** Read as 0; should be written with 0.

40.4.1.48 ICU error capture register

On detection of an ECC error at ICUz, the ECR register captures the related acknowledged service request information. This is done by updating the ECR with the content of the Last Acknowledged Service Request (LASR) register. The ECR shows always the last acknowledged information where an ECC error was detected.

***Note**: ECC is used here for error detection only. Detected errors are signaled through Alarm_CS to SMU_CS or through Alarm_NCS to SMU_SAFE.*

ECRz (z=0-14) Offset address: $0C24_H+z*34_H$

ICU error capture register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		VM		CS						ID					
r		rh		rh						rh					

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		ECC						PN							
r		rh						rh							

Field	Bits	Type	Description
PN	7:0	rh	**Service request priority number** This bit field shows the priority number of the last service request where an error was detected. Bit field can be modified by writing to it.
ECC	13:8	rh	**Received ECC information** This bit field shows the ECC of the last Acknowledged information where an error was detected.

(table continues...)

(continued)

Field	Bits	Type	Description
ID	26:16	rh	**Service request node index number** This bit field shows the ID of the last service request where an error was detected. Bit field can be modified by writing to it
CS	27	rh	**Cyber security information** 0_B Service request is not assigned to the security application 1_B Service request is assigned to the security application
VM	30:28	rh	**Virtual machine number**
0	15:14, 31	r	**Reserved-0** Read as 0; should be written with 0.

40.4.1.49 ICU error control register

Error Status (STAT) and error overflow (EOV) bits can be used as error handling mechanism and indication that error information was lost.

If EOV is cleared by SW, it shall be cleared together with the STAT bit.

Clearing STAT by SW has precedence over a new STAT set situation. The clearing of the EOV by SW has precedence over a simultaneous EOV situation. Both corner cases are not relevant to the application because it cannot be controlled whether a STAT and EOV event occurs one cycle before or after the respective SW clear.

ECTRLz (z=0-14) Offset address: $0C28_H+z*34_H$

ICU error control register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												EOV CLR	EOV	STAT CLR	STAT
r												w	rh	w	rh

Field	Bits	Type	Description
STAT	0	rh	**Error status bit** The error status bit is set whenever an ECC error was detected by the ICU. 0_B No ECC error detected 1_B ECC error detected
STATCLR	1	w	**Error status bit** The STATCLR bit is used to clear the STAT bit. Note: the STAT bit is not cleard in case of an error event that happens in parallel. 0_B No action 1_B Clear STAT bit; read always returns 0; no action if STAT bit is set in parallel by hardware to 1.

(table continues...)

(continued)

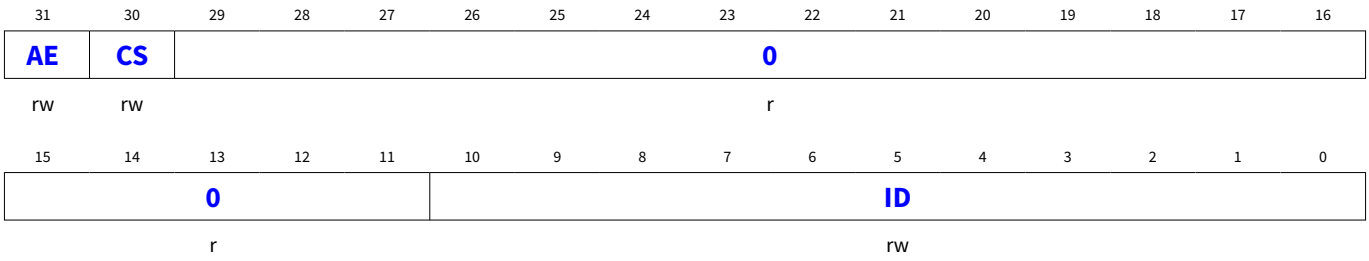
Field	Bits	Type	Description
EOV	2	rh	**Error overflow bit** The bit is set if an ECC error was detected by the ICU while ECTRL.STAT= ´1´ (error overflow situation). 0_B No error overflow situation detected 1_B Error overflow situation detected
EOVCLR	3	w	**Error overflow bit** The EOVCLR bit is used to clear the EOV bit. The EOV bit should be cleared together with the STAT bit. Note: the EOV bit is not cleard in case of an EOV event that happens in parallel. 0_B No action 1_B Clear EOV bit; read always returns 0; no action if EOV bit is set in parallel.
0	31:4	r	**Reserved** Read as 0; should be written with 0.

40.4.1.50 ICU error generation register

The EGEN register enables application software to check the ICU ECC error detection mechanism and the ALARM signal to the SMU.

EGENz (z=0-14) Offset address: $0C2C_H+z*34_H$

ICU error generation register Application Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
ID	10:0	rw	**Service request node ID** The bitfield describes the index number (ID) of the SRN whose SR is to be transmitted with an ECC error.
CS	30	rw	**Cyber security** A CS write access (CS_VALID=´1´) to this register is only executed if it writes with CS=´1´. A non CS write access (CS_VALID=´0´) to this register is only executed if it writes with CS=´0´.: 0_B Error is only injected if service request selected through IDX is not security related (SCRx.CS=´0´). 1_B Error is only injected if service request selected through IDX is security related (SCRx.CS=´1´).

(table continues...)

(continued)

Field	Bits	Type	Description
AE	31	rw	**Activate error enable** Activates the selective inversion of ECC bit 0 of a service request from the selected selected SRNx (EGEN.ID=x). Note: The ICU only injects the ECC error if SRCx.CS = EGEN.CS. 0_B Not enabled 1_B Enabled
0	29:11	r	**Reserved-0** Read as 0; should be written with 0.

40.4.1.51 ICU VM control register

Controls which VM is considered in the ICU Arbitration sequence. Arbitration slots of disabled VMs are skipped in the arbitration sequence.

***Note**: Hardware virtualization support is disabled for this ICU if all VM bits in this register are disabled. This means that only the SRCx.SRPN information is considered for arbitration, the SRCx.VM information will be ignored. All pending SRs for this ICU are arbitrated and the winner SR is passed to the ISP.*

VMENz (z=0-14) Offset address: $0C30_H+z*34_H$

ICU VM control register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								VM7	VM6	VM5	VM4	VM3	VM2	VM1	VM0
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
VMy (y=0-7)	y	rw	**VMy control** 0_B VMy is disabled for ICU arbitration 1_B VMy is enabled for ICU arbitration
0	31:8	r	**Reserved-0** Read as 0; should be written with 0.

40.4.2 Registers of SRC

40.4.2.1 Register overview - access mode glossary

Table 1924 Register overview - access mode glossary

Keyword	Description
M	Indicates a module specific access condition. Refer to the register description for details of the specific access condition.
16	Access when using 16-bit width or higher.

40.4.2.2 Registers overview - SRC (ascending offset address)

Table 1925 Registers overview - SRC (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
SRCi	Service request control register i	00000_H+i *4	M	SV, M, 16	Application Reset	6939

40.4.2.3 Service request control register i

The access protection mechanism for the SRCx register is described in chapter IR, access protection of SRCx registers.

SRCi (i=0-2047) Offset address: $00000_H + i*4$

Service request control register i Application Reset value: $0000\ F000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0			IOVCLR	IOV	SETR	CLRR	SRR	SRE	0						
r			w	rh	w	w	rh	rw	r						

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TOS				CS	VM			SRPN							
rw				rw	rw			rw							

Field	Bits	Type	Description
SRPN	7:0	rw	**Service request priority number** The SRPN bit field defines the priority of a service request. The larger SRPN number has the higher priority.
VM	10:8	rw	**Virtual machine** The VM bit field configuration maps a service request to a virtual machine. ***Note**: SRNs mapped to DMAs, PPU, GTM and CPUs in non-hardware virtualization mode shall be configured with SRCx.VM=000_B.* 000_B Service request is mapped to VM0 001_B Service request is mapped to VM1 010_B Service request is mapped to VM2 011_B Service request is mapped to VM3 100_B Service request is mapped to VM4 101_B Service request is mapped to VM5 110_B Service request is mapped to VM6 111_B Service request is mapped to VM7
CS	11	rw	**Cyber security** 0_B Service request is not assigned to the security application 1_B Service request is assigned to the security application

(table continues...)

(continued)

Field	Bits	Type	Description
TOS	15:12	rw	**Type of service** The TOS bit field configuration maps a service request to an interrupt service provider: ***Note**: In products where DMA1, GTM, PPU, RDMA or CPUx is not implemented, the related TOS encoding will not be used and treated as RESERVED.* 0_H Service request is mapped to CPU0 1_H Service request is mapped to CPU1 2_H Service request is mapped to CPU2 3_H Service request is mapped to CPU3 4_H Service request is mapped to CPU4 5_H Service request is mapped to CPU5 6_H Service request is mapped to CPUCS 7_H Service request is mapped to DMA0 8_H Service request is mapped to DMA1 9_H Service request is mapped to GTM A_H Service request is mapped to PPU B_H Service request is mapped to RDMA F_H SRN not used **others,** Reserved
SRE	23	rw	**Service request enable** 0_B Service request is disabled 1_B Service request is enabled
SRR	24	rh	**Service request status** The SRR bit shows the status of the service request. 0_B No service request is pending 1_B Service request is pending
CLRR	25	w	**Service request clear** The CLRR bit is required to reset SRR. 0_B No action 1_B Write with 1 leads to SRR=0; no action if SETR is written with 1 at the same time; no clear of SRR if a hardware interrupt trigger happens at the same time. read always returns 0.
SETR	26	w	**Service Request set** The SETR bit is required to set SRR. 0_B No action 1_B Write with 1 leads to SRR=1; if SRR is already =1 then IOV is set to 1; no action if CLRR is written with 1 at the same time; read always returns 0.

(table continues...)

(continued)

Field	Bits	Type	Description
IOV	27	rh	**Interrupt overflow status** The IOV bit is set by HW when a new service request is triggered by interrupt trigger or write to the SETR bit with 1 while SRR=1. 0_B No interrupt overflow detected 1_B Interrupt overflow detected
IOVCLR	28	w	**Interrupt overflow clear** IOVCLR is required to reset IOV. 0_B No action 1_B Write with 1 lead to IOV=0; no action in case of an IOV condition at the same time; read always returns 0.
0	22:16, 31:29	r	**Reserved-0** Read as 0; should be written with 0.

40.5 Debug information

IR debug information.

40.5.1 OCDS trigger bus interface (OTGB)

The IR module has three 16-bit trigger sets which are selected with the OCS register.

The IR module provides

- two OTGB 16 bit interfaces to the MCDS module, OTGB0 and OTGB1
- three 16 bit trigger sets as described below TS16_IS, TS16_SPA and TS_16_SP
- Standard OCS register:
  - OCS.TGS to select which trigger set is mapped to OTGB0 and OTGB1
  - OCS.TGB to select which of the two OTGB interfaces, OTGB0 OTGB1 is used. TGB is ignored if TC16_SP is selected

If TS16_SP is selected, OCS.TGB is ignored and there is the option for two instances in parallel on OTGB0 and OTGB1 (control by OIT register).

Table 1926 IR trigger sets selected by OCS.TGS

Trigger set	Description	Details
TS16_IS Interrupt Selection	Request state of selected interrupt request signals	• 16 bit trigger set, bit mapping described in TS16_IS table • set selected through OCS.TGS • mapped to OTGB0 or OTGB1 through OCS.TGB
TS16_SPA Service Provider Activity	Parallel view of requests waiting for all service providers	• 16 bit trigger set, bit mapping described in TS16_SPA table • set selected through OCS.TGS • mapped to OTGB0 or OTGB1 through OCS.TGB
TS16_SP Service Provider	Interrupt activities of a selected service provider. Option for two Service Requests in parallel on OTGB0 and OTBG1	• 16 bit trigger set, bit mapping described in TS16_SP table • set selected through OCS.TGS • OCS.TGB is ignored if TS16_SP is selected • 16 bit trigger set can be mapped to OTGB0 and OTGB1 with two configurations to observe service requests to two different ISPs in parallel. Configuration controlled through the register OIT

40.5.1.1 Request state of selected interrupt request signals

Table 1927 TS16_IS trigger set interrupt selection

Bits	Name	Description
0	SI0	Selected (see OIXS0) interrupt 0

(table continues...)

Table 1927 (continued) TS16_IS trigger set interrupt selection

Bits	Name	Description
1	SI1	Selected (see OIXS0) interrupt 1
2	SI2	Selected (see OIXS1) interrupt 2
3	SI3	Selected (see OIXS1) interrupt 3
[14:4]		Reserved (value is 0)
15	SIL	Selected (see OIXMS) interrupt trigger lost

***Note**: The selected interrupt reflects the service requests by hardware and by software.*

CyberSecurity activity visibility (CSAV) control

Only for CSAV=Full, TS16_IS will show the status for SRN with SRC.CS=1. This applies to all TS16_IS bits: SIx and SIL.

For CSAV!=Full and a the selected SRC is configured with CS=1, the respective bit is set to 0. This applies to all TS16_IS bits: SIx and SIL.

40.5.1.2 Parallel view for all service providers of requests waiting

Table 1928 TS16_SPA trigger set interrupt selection

Bits	Name	Description
0	CPU0	CPU0: One or more service requests waiting
1	CPU1	CPU1: One or more service requests waiting
2	CPU2	CPU2: One or more service requests waiting
3	CPU3	CPU3: One or more service requests waiting
4	CPU4	CPU4: One or more service requests waiting
5	CPU5	CPU5: One or more service requests waiting
6	CPUCS	CPUCS: One or more service requests waiting (only visible if CSAV=Full)
7	DMA0	DMA0: One or more service requests waiting (only visible if CSAV=Limited or Full)
8	DMA1	DMA1: One or more service requests waiting (only visible if CSAV=Limited or Full)
9	GTM	GTM: One or more service requests waiting
10	PPU	PPU: One or more service requests waiting
11	RDMA	RDMA: One or more service requests waiting
[14:12]		Reserved (value is 0)
15	SIL	Selected (see OIXMS) interrupt trigger lost

The encoding in the above TS16_SPA trigger set table covers the full device family. The timing resolution of 'One or more requests waiting' is one ICU arbitration round. Also immediately taken requests are visible in TS16_SPA

***Note**: For all TS16_IS signals, ´1´ means that one or more SR are pending, 0 means that non SR is pending.*

CyberSecurity Activity Visibility (CSAV) control

- Bit CPUCS: Only for CSAV=Full, TS16_SPA will show the CPUcs activity
- Bits DMA0 and DMA1: Only for CSAV=Limited or CSAV=Full, TS16_SPA will show the DMA0 and DMA1 activities
- SIL: Only with CSAV=Full the status of SRNs with SRC.CS=1 is shown in TS16_SPA SIL. For CSAV!=Full and SRC configured with CS=1, SIL is set to 0

40.5.1.3 Interrupt activities of a selected service provider

The interrupt system provides the ability to observe up to two service providers in parallel on OTGB0 and OTGB1. This is configured with the OIT register (OCS.TGB is ignored in this case).

Table 1929 TS16_SP trigger set interrupt selection

Bits	Name	Description
0	A	Request acknowledged
1	W	One or more service requests waiting for arbitration by the service providers as selected through the OIT register
[3:2]		Reserved (value is 0)
[14:4]	ASR	Accepted Service Request index (status of the selected ICU LASR.IDX)
15	SIL	Selected (see OIXMS) interrupt trigger lost

***Note**: The timing resolution of A and W is one ICU arbitration round.*

TS16_SP is implemented in a way which minimizes the value changes. The following rules apply:

- A is a single cycle pulse
- ASR will only change with the pulse on A (when A has a positive edge)

CyberSecurity Activity Visibility (CSAV) control

- Selected ICU is CPUCS: Only for CSAV=Full, TS16_SP will show the CPUcs activity for A, W, ASR. For CSAV! =Full and SRC configured with CS=1, A, W, ASR are set to 0
- Selected ICU is DMA0 or DMA1: Only for CSAV=Limited or CSAV=Full, TS16_SP will show the DMA0 and DMA1 activities for A, W, ASR. Otherwise A, W, ASR are set to 0
- SIL: Only with CSAV=Full the status of SRNs with SRC.CS=1 is shown in TS16_SP SIL. For CSAV!=Full and SRC configured with CS=1, SIL is set to 0

40.5.1.4 Example for TS16_SP

shows an example sequence of two SR from SRN 123 with one SR from SRN 178. The SRN number are unique SRN index numbers and do not present the configured priority of the SR.

Table 1930 TS16_SP trigger set interrupt selection

ASR	A	W	Description
xxx	0	0	No SR pending. Last acknowledged SR was related to SRN xxx
xxx	0	1	One or more service requests pending. Last acknowledged SR was related to SRN xxx
123	1	x	SR from SRN 123 is accepted by ISP. W is updated with delay

(table continues...)

Table 1930 (continued) TS16_SP trigger set interrupt selection

ASR	A	W	Description
123	0	1	One or more interrupts are pending. Last acknowledged SR was related to SRN 123
178	1	x	SR from SRN 178 is accepted by ISP. W is updated with delay
178	0	1	One or more interrupts are pending. Last acknowledged SR was related to SRN 178
123	1	x	New SR from SRN 123 is accepted by ISP. W is updated with delay
123	0	0	No SR pending. Last acknowledged SR was related to SRN 123

Table 1931 TS16_SP example with a CPU

ASR	A	W	Description
123	0	0	Last service request was 123. (Info: CPU is executing user code and interrupts are enabled)
123	1	1	New service request from SRN 123 is accepted
123	1	0	No service request pending
123	0	1	New service request(s) pending. (Info: SRN 456 and 789 in parallel)CPU is still executing ISR 123 and - or interrupts are disabled
456	1	1	CPU accepts SRN 456 (Info: SRN 456 has higher priority than 789)SRN 456 has either higher priority than SRN123 or ISR 123 is finished
456	0	1	Service request(s) pending. CPU is executing ISR 456
789	1	1	CPU acknowledges service request 789
789	0	0	No service request pending

40.6 References

This section is not applicable for the module being described.

40.7 IR revision history

Reference	Description of change(s)
Date range: 2024-08-17 to 2025-01-20	
Changing the VMEN configuration	• Equations updated
Register overview - access mode glossary	• Keywords updated
VMENz	• Register updated

40.8 TC4Dx IR information

40.8.1 TC4Dx IR configuration

The table below shows the Interrupt Router configuration as implemented in this AURIX™ device.

The IR slave interface that provides access to the IR registers is connected to the COMPB.

ICU hardware virtualization support (HVS):

The table also shows which ICU provides the hardware virtualization support (HVS).

- HVS = ´Yes´ - ICU has 8 LWSR registers (LWSR0-LWSR7) and the VMEN register is implemented with 8 bits to control which VM encoding to be considered. If all VMs are disabled in the VMEN register, hardware virtualization support is disabled, SRCx.VM information is ignored by the ICU arbitration and only LWSR0 is updated by hardware
- HVS = ´No´ - ICU has 1 LWSR register (LWSR0) and the VMEN register implemented hardwired 0 (HVS disabled). The SRCx.VM information is ignored by the ICU arbitration. Only LWSR0 is updated by hardware

Software interrupt groups (GPSRx, SRBx, SRBx_ACCEN, GPSRx_SWC, GPSRx_ACCEN)

Table line GPSRx shows the configuration relation between available ICUx and the available general purpose software interrupt groups where each group includes the registers GPRSxy, SRBx, SWCxy and SWCACCENxy.

There is no fixed assignment of GPSRx groups to ISPs but this information can be used to make the correct selection of GPSR groups for the application

- Example: GPRS7 and related registers are available in all products with a PPU
- Example: GPRS1 and related registers is available in all products with a CPU1

Table 1932 Mapping of interrupt service provider to ICUs and implemented SRNs

	CPU0	CPU1	CPU2	CPU3	CPU4	CPU5	CPUCS	DMA0	DMA1	-	PPU	-
HVS	Yes	Yes	Yes	Yes	Yes	Yes	No	No	No	-	No	-
ICU index	ICU0	ICU1	ICU2	ICU3	ICU4	ICU5	ICU6	ICU7	ICU8	-	ICU10	-
GPSRx	0	1	2	3	4	5	6	-	-	-	7	-

40.8.2 TC4Dx IR features

There are no deviations from the generic specification.

40.8.3 TC4Dx IR functional description

There are no deviations from the generic specification.

40.8.4 TC4Dx IR registers

40.8.4.1 Registers of INT

40.8.4.1.1 Register address space - INT

Table 1933 Registers address space - INT

Module	Base address	End address	Note
INT	$F4430000_H$	$F4431FFF_H$	

40.8.4.1.2 Register overview - access mode glossary

Table 1934 Register overview - access mode glossary

Keyword	Description
CSE	Access protection using PROT register INT_PROTCSE .
SE	Access protection using PROT register INT_PROTSE .
TOSz (z=0-10)	Access protection using PROT register INT_PROTTOSz .
APU-PCS	Protection group consisting of registers INT_ACCENCS_WRA , INT_ACCENCS_WRB , INT_ACCENCS_RDA , INT_ACCENCS_RDB , INT_ACCENCS_VM , INT_ACCENCS_PRS .
PCS	Access protection using *APU-PCS* registers.
APU-PDBG	Protection group consisting of registers INT_ACCENDBG_WRA , INT_ACCENDBG_WRB , INT_ACCENDBG_RDA , INT_ACCENDBG_RDB , INT_ACCENDBG_VM , INT_ACCENDBG_PRS .
PDBG	Access protection using *APU-PDBG* registers.
APU-PSCFGz (z=0-10)	Protection group consisting of registers INT_TOSz_ACCENSCFG_WRA , INT_TOSz_ACCENSCFG_WRB , INT_TOSz_ACCENSCFG_RDA , INT_TOSz_ACCENSCFG_RDB , INT_TOSz_ACCENSCFG_VM , INT_TOSz_ACCENSCFG_PRS .
PSCFGz	Access protection using *APU-PSCFGz* registers.
APU-PSCTRLz (z=0-10)	Protection group consisting of registers INT_TOSz_ACCENSCTRL_WRA , INT_TOSz_ACCENSCTRL_WRB , INT_TOSz_ACCENSCTRL_RDA , INT_TOSz_ACCENSCTRL_RDB , INT_TOSz_ACCENSCTRL_VM , INT_TOSz_ACCENSCTRL_PRS .
PSCTRLz	Access protection using *APU-PSCTRLz* registers.
APU-PSRBy (y=0-7)	Protection group consisting of registers INT_ACCENSRBy_WRA , INT_ACCENSRBy_WRB , INT_ACCENSRBy_RDA , INT_ACCENSRBy_RDB , INT_ACCENSRBy_VM , INT_ACCENSRBy_PRS .
PSRBy	Access protection using *APU-PSRBy* registers.
BE	Always returns a Bus Error.
U	No access restrictions.
PROT	Access restrictions as defined in the PROT register access rules.
M	Indicates a module specific access condition. Refer to the register description for details of the specific access condition.

40.8.4.1.3 Registers overview - INT (ascending offset address)

Table 1935 Registers overview - INT (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
INT_OCS	OCDS control and status register	0004_H	PDBG	SV, PDBG	Debug Reset	6902
INT_ID	Module identification register	0008_H	PDBG	BE	Application Reset	6903
INT_LCLTEST	LCL test register	$000C_H$	PDBG	SV, SE, PDBG	Application Reset	6903
INT_OIXMS	OTGB IRQ MUX missed IRQ select	0010_H	PDBG	SV, PDBG	Debug Reset	6904
INT_OIXS0	OTGB IRQ MUX select 0	0014_H	PDBG	SV, PDBG	Debug Reset	6904
INT_OIXS1	OTGB IRQ MUX select 1	0018_H	PDBG	SV, PDBG	Debug Reset	6905
INT_OIT	OTGB IRQ trace	$001C_H$	PDBG	SV, PDBG	Debug Reset	6905
INT_PROTSE	PROT register safe endinit	0020_H	U	SV, PROT	Application Reset	6907
INT_PROTCSE	PROT register cyber security endinit	0024_H	U	SV, PROT	Application Reset	6909
INT_PROTTOSz (z=0-8,10)	PROT register for TOS=z	0030_H+z *4	U	SV, PROT	Application Reset	6910
INT_ACCENCS_WRA	CS write access enable register A	0080_H	U	CSE, SV	Application Reset	6912
INT_ACCENCS_WRB	CS write access enable register B	0084_H	U	CSE, SV	Application Reset	6913
INT_ACCENCS_RDA	CS read access enable register A	0088_H	U	CSE, SV	Application Reset	6913
INT_ACCENCS_RDB	CS read access enable register B	$008C_H$	U	CSE, SV	Application Reset	6914
INT_ACCENCS_VM	CS VM access enable register	0090_H	U	CSE, SV	Application Reset	6914
INT_ACCENCS_PRS	CS PRS access enable register	0094_H	U	CSE, SV	Application Reset	6915
INT_ACCENDBG_WRA	DBG write access enable register A	$00A0_H$	U	SE, SV	Application Reset	6915
INT_ACCENDBG_WRB	DBG write access enable register B	$00A4_H$	U	SE, SV	Application Reset	6916
INT_ACCENDBG_RDA	DBG read access enable register A	$00A8_H$	U	SE, SV	Application Reset	6916
INT_ACCENDBG_RDB	DBG read access enable register B	$00AC_H$	U	SE, SV	Application Reset	6917
INT_ACCENDBG_VM	DBG VM access enable register	$00B0_H$	U	SE, SV	Application Reset	6917

(table continues...)

Table 1935 (continued) Registers overview - INT (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
INT_ACCENDBG_PRS	DGB PRS access enable register	$00B4_H$	U	SE, SV	Application Reset	6918
INT_ACCENSRBy_WRA (y=0-7)	SRBy write access enable register A	0100_H+y *18_H	U	SE, SV	Application Reset	6919
INT_ACCENSRBy_WRB (y=0-7)	SRBy write access enable register B	0104_H+y *18_H	U	SE, SV	Application Reset	6919
INT_ACCENSRBy_RDA (y=0-7)	SRBy read access enable register A	0108_H+y *18_H	U	SE, SV	Application Reset	6920
INT_ACCENSRBy_RDB (y=0-7)	SRBy read access enable register B	$010C_H$+y *18_H	U	SE, SV	Application Reset	6920
INT_ACCENSRBy_VM (y=0-7)	SRBy VM access enable register	0110_H+y *18_H	U	SE, SV	Application Reset	6921
INT_ACCENSRBy_PRS (y=0-7)	SRBy PRS access enable register	0114_H+y *18_H	U	SE, SV	Application Reset	6921
INT_TOSz_ACCENSCFG_WRA (z=0-8,10)	SRC[15:0] write access enable register A for TOS=z	0300_H+z *30_H	U	TOSz, SV	Application Reset	6922
INT_TOSz_ACCENSCFG_WRB (z=0-8,10)	SRC[15:0] write access enable register B for TOS=z	0304_H+z *30_H	U	TOSz, SV	Application Reset	6923
INT_TOSz_ACCENSCFG_RDA (z=0-8,10)	SRC[15:0] read access enable register A for TOS=z	0308_H+z *30_H	U	TOSz, SV	Application Reset	6923
INT_TOSz_ACCENSCFG_RDB (z=0-8,10)	SRC[15:0] read access enable register B for TOS=z	$030C_H$+z *30_H	U	TOSz, SV	Application Reset	6924
INT_TOSz_ACCENSCFG_VM (z=0-8,10)	SRC[15:0] VM access enable register for TOS=z	0310_H+z *30_H	U	TOSz, SV	Application Reset	6924
INT_TOSz_ACCENSCFG_PRS (z=0-8,10)	SRC[15:0] PRS access enable register for TOS=z	0314_H+z *30_H	U	TOSz, SV	Application Reset	6925
INT_TOSz_ACCENSCTRL_WRA (z=0-8,10)	SRC[31:16] write access enable register A for TOS=z	0318_H+z *30_H	U	TOSz, SV	Application Reset	6925

(table continues...)

Table 1935 (continued) Registers overview - INT (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
INT_TOSz_ACCENSCTRL_WRB (z=0-8,10)	SRC[31:16] write access enable register B for TOS=z	$031C_H$+z*30_H	U	TOSz, SV	Application Reset	6926
INT_TOSz_ACCENSCTRL_RDA (z=0-8,10)	SRC[31:16] read access enable register A for TOS=z	0320_H+z*30_H	U	TOSz, SV	Application Reset	6926
INT_TOSz_ACCENSCTRL_RDB (z=0-8,10)	SRC[31:16] read access enable register B for TOS=z	0324_H+z*30_H	U	TOSz, SV	Application Reset	6927
INT_TOSz_ACCENSCTRL_VM (z=0-8,10)	SRC[31:16] VM access enable register for TOS=z	0328_H+z*30_H	U	TOSz, SV	Application Reset	6927
INT_TOSz_ACCENSCTRL_PRS (z=0-8,10)	SRC[31:16] PRS access enable register for TOS=z	$032C_H$+z*30_H	U	TOSz, SV	Application Reset	6928
INT_GPSRGx_SWCACCENy (x=0-7;y=0-7)	GPRSGx_SWCy write access protection register	0700_H+x*40_H+y*4	M	SV, M	Application Reset	6929
INT_GPSRGx_SWCy (x=0-7;y=0-7)	SW control register for GPSRGxSRy	0720_H+x*40_H+y*4	M	SV, M	Application Reset	6931
INT_SRBx (x=0-7)	Service request broadcast register x	$0B00_H$+x*4	PSRBx	SV, PSRBx	Application Reset	6933
INT_LWSRz_VMy (y=0-7;z=0-5) (y=0;z=6-8,10)	ICU latest service request information signaled for VMy	$0C00_H$+z*34_H+y*4	PSCTRLz	BE	Application Reset	6933
INT_LASRz (z=0-8,10)	ICU Last acknowledged service request register	$0C20_H$+z*34_H	PSCTRLz	BE	Application Reset	6934
INT_ECRz (z=0-8,10)	ICU error capture register	$0C24_H$+z*34_H	PSCFGz	BE	Application Reset	6935
INT_ECTRLz (z=0-8,10)	ICU error control register	$0C28_H$+z*34_H	PSCFGz	SV, PSCFGz	Application Reset	6936
INT_EGENz (z=0-8,10)	ICU error generation register	$0C2C_H$+z*34_H	PSCFGz	SV, PSCFGz	Application Reset	6937
INT_VMENz (z=6-8,10)	ICU VM control register	$0C30_H$+z*34_H	PSCFGz	BE	Application Reset	
INT_VMENz (z=0-5)	ICU VM control register	$0C30_H$+z*34_H	PSCFGz	SV, PSCFGz	Application Reset	6938

40.8.4.1.4 Device specific registers

There are no device specific register changes.

40.8.4.2 Registers of SRC

40.8.4.2.1 Register address space - SRC

Table 1936 Registers address space - SRC

Module	Base address	End address	Note
SRC	$F4432000_H$	$F4435FFF_H$	IR service request Control Registers (SRC)

40.8.4.2.2 Register overview - access mode glossary

Table 1937 Register overview - access mode glossary

Keyword	Description
M	Indicates a module specific access condition. Refer to the register description for details of the specific access condition.
16	Access when using 16-bit width or higher.

40.8.4.2.3 Registers overview - SRC (ascending offset address)

Table 1938 Registers overview - SRC (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
SRC_CPUwSB (w=0-5)	CPUw software breakpoint service request	00000_H+ w*4	M	SV, M, 16	Application Reset	6958
SRC_CPUCSSB	CPUCS software breakpoint service request	00018_H	M	SV, M, 16	Application Reset	6958
SRC_STMCPUwSRx (w=0-5;x=0-15)	CPUw system timer service requests SRx	00020_H+ w*40_H+x *4	M	SV, M, 16	Application Reset	6960
SRC_STMCPUCSSR2	STM1 CPUCS service request 0	$001A8_H$	M	SV, M, 16	Application Reset	6960
SRC_STMCPUCSSR3	STM1 CPUCS service request 1	$001AC_H$	M	SV, M, 16	Application Reset	6960
SRC_SBCUNCS	SBCU NCS service request	$001E0_H$	M	SV, M, 16	Application Reset	6962
SRC_SBCUCS	SBCU CS service request	$001E4_H$	M	SV, M, 16	Application Reset	6965
SRC_SBCUQOSC	SBCU QOS control service request	$001E8_H$	M	SV, M, 16	Application Reset	6962
SRC_COMBCUNCS	COMBCU NCS service request	$001F8_H$	M	SV, M, 16	Application Reset	6958

(table continues...)

Table 1938 (continued) Registers overview - SRC (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
SRC_COMBCUCS	COMBCU CS service request	$001FC_H$	M	SV, M, 16	Application Reset	6958
SRC_COMBCUQOSC	COMBCU QOS control service request	00200_H	M	SV, M, 16	Application Reset	6958
SRC_TBCUNCS	TBCU NCS service request	00210_H	M	SV, M, 16	Application Reset	6960
SRC_TBCUCS	TBCU CS service request	00214_H	M	SV, M, 16	Application Reset	6960
SRC_CSBCUNCS	CSBCU NCS service request	$0021C_H$	M	SV, M, 16	Application Reset	6958
SRC_CSBCUCS	CSBCU CS service request	00220_H	M	SV, M, 16	Application Reset	6958
SRC_QOSARPw (w=0-7)	QOS ARP w service request	00230_H+ w*4	M	SV, M, 16	Application Reset	6965
SRC_TRIF	TRIF status service request	00250_H	M	SV, M, 16	Application Reset	6960
SRC_SRIw (w=0-7)	SRI domain w service request	00260_H+ w*8	M	SV, M, 16	Application Reset	6962
SRC_SRIwSM (w=0-7)	SRI domain w special master service request	00264_H+ w*8	M	SV, M, 16	Application Reset	6960
SRC_CERBERUSw (w=0-1)	Cerberus service request w	$002A0_H$+ w*4	M	SV, M, 16	Application Reset	6967
SRC_ASCLINwTX (w=0-27)	ASCLINw transmit service request	$002B0_H$+ w*12	M	SV, M, 16	Application Reset	6967
SRC_ASCLINwRX (w=0-27)	ASCLINw receive service request	$002B4_H$+ w*12	M	SV, M, 16	Application Reset	6967
SRC_ASCLINwERR (w=0-27)	ASCLINw error service request	$002B8_H$+ w*12	M	SV, M, 16	Application Reset	6967
SRC_VMTwSR (w=0-6)	VMTw service request	00410_H+ w*4	M	SV, M, 16	Application Reset	6960
SRC_QSPIwTX (w=0-7)	QSPIw transmit service request	00460_H+ w*14_H	M	SV, M, 16	Application Reset	6965
SRC_QSPIwRX (w=0-7)	QSPIw receive service request	00464_H+ w*14_H	M	SV, M, 16	Application Reset	6965
SRC_QSPIwERR (w=0-7)	QSPIw error service request	00468_H+ w*14_H	M	SV, M, 16	Application Reset	6965
SRC_QSPIwPT (w=0-7)	QSPIw phase transition service request	$0046C_H$+ w*14_H	M	SV, M, 16	Application Reset	6965

(table continues...)

Table 1938 (continued) Registers overview - SRC (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
SRC_QSPIwU (w=0-7)	QSPIw user defined service request	00470_H+ w*14_H	M	SV, M, 16	Application Reset	6965
SRC_HSCTw (w=0-1)	HSCTw service request	00540_H+ w*4	M	SV, M, 16	Application Reset	6970
SRC_HSSLwCOKx (w=0-1;x=0-3)	HSSLw channel x OK service request	00550_H+ w*44_H+x *10_H	M	SV, M, 16	Application Reset	6972
SRC_HSSLwRDIx (w=0-1;x=0-3)	HSSLw channel x read data service request	00554_H+ w*44_H+x *10_H	M	SV, M, 16	Application Reset	6972
SRC_HSSLwERRx (w=0-1;x=0-3)	HSSLw channel x error service request	00558_H+ w*44_H+x *10_H	M	SV, M, 16	Application Reset	6972
SRC_HSSLwTRGx (w=0-1;x=0-3)	HSSLw channel x trigger interrupt service request	$0055C_H$+ w*44_H+x *10_H	M	SV, M, 16	Application Reset	6972
SRC_HSSLwEXI (w=0-1)	HSSw exception service request	00590_H+ w*44_H	M	SV, M, 16	Application Reset	6972
SRC_I2CwDTR (w=0-2)	I2Cw data transfer request	$005E0_H$+ w*10_H	M	SV, M, 16	Application Reset	6972
SRC_I2CwERR (w=0-2)	I2Cw error service request	$005E4_H$+ w*10_H	M	SV, M, 16	Application Reset	6972
SRC_I2CwP (w=0-2)	I2Cw Protocol service request	$005E8_H$+ w*10_H	M	SV, M, 16	Application Reset	6972
SRC_SENTwSRx (w=0-1;x=0-9)	SENTw service request x	00610_H+ w*28_H+x *4	M	SV, M, 16	Application Reset	6962
SRC_MSC0SRx (w=0;x=0-4)	MSC0 service request x	00660_H+ x*4	M	SV, M, 16	Application Reset	6975
SRC_FCE	FCE error service request	$006D0_H$	M	SV, M, 16	Application Reset	6970
SRC_DMA0ERRx (x=0-15)	DMA0 RPx error service request	$006E0_H$+ x*4	M	SV, M, 16	Application Reset	6958
SRC_DMA0GERR	DMA0 global error service request	00720_H	M	SV, M, 16	Application Reset	6977
SRC_DMA0CHx (x=0-127)	DMA0 channel x service request	00730_H+ x*4	M	SV, M, 16	Application Reset	6958
SRC_DMA1ERRx (x=0-7)	DMA1 RPx error service request	00930_H+ x*4	M	SV, M, 16	Application Reset	6977
SRC_DMA1GERR	DMA1 global error service request	00970_H	M	SV, M, 16	Application Reset	6977

(table continues...)

Table 1938 (continued) Registers overview - SRC (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
SRC_DMA1CHx (x=0-63)	DMA1 channel x service request	00980_H+x*4	M	SV, M, 16	Application Reset	6977
SRC_SDMMC0ERR (w=0)	SDMMC0 error service request	$00B80_H$	M	SV, M, 16	Application Reset	6962
SRC_SDMMC0DMA (w=0)	SDMMC0 DMA ready service request	$00B84_H$	M	SV, M, 16	Application Reset	6962
SRC_GETH0DMAx (w=0;x=0-15)	GETH0 DMAx service request	$00B90_H$+x*4	M	SV, M, 16	Application Reset	6970
SRC_GETH0PPSx (w=0;x=0-3)	GETH0 PPSx service request	$00BD0_H$+x*4	M	SV, M, 16	Application Reset	6970
SRC_GETH0INTR (w=0)	GETH0 INTR service request	$00BE0_H$	M	SV, M, 16	Application Reset	6970
SRC_LETH0DMATXx (w=0;x=0-7)	LETH0 DMA TXx service request	$00BF0_H$+x*4	M	SV, M, 16	Application Reset	6972
SRC_LETH0DMARXx (w=0;x=0-7)	LETH0 DMA RXx service request	$00C10_H$+x*4	M	SV, M, 16	Application Reset	6972
SRC_LETH0PPSx (w=0;x=0-3)	LETH0 Px_PPS service request	$00C30_H$+x*4	M	SV, M, 16	Application Reset	6975
SRC_LETH0INTR (w=0)	LETH0 INTR service request	$00C40_H$	M	SV, M, 16	Application Reset	6975
SRC_CANwINTx (w=0-4;x=0-15)	CANw service request x	$00C50_H$+w*40_H+x*4	M	SV, M, 16	Application Reset	6967
SRC_CANXLwFUNC (w=0-3)	CANXLw functional service request	$00DD0_H$+w*12	M	SV, M, 16	Application Reset	6967
SRC_CANXLwERR (w=0-3)	CANXLw error service request	$00DD4_H$+w*12	M	SV, M, 16	Application Reset	6967
SRC_CANXLwSAFETY (w=0-3)	CANXLw safety service request	$00DD8_H$+w*12	M	SV, M, 16	Application Reset	6967
SRC_TMADCwSRx (w=0-3;x=0-5)	TMADCw service request x	$00E10_H$+w*18_H+x*4	M	SV, M, 16	Application Reset	6960

(table continues...)

Table 1938 (continued) Registers overview - SRC (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
SRC_ADCSRx (x=0-11)	ADC service request x	$00ED0_H$+x*4	M	SV, M, 16	Application Reset	6967
SRC_CDSPwSRx (w=0-1;x=0-2)	CDSPw service request x	$010D0_H$+w*12+x*4	M	SV, M, 16	Application Reset	6967
SRC_ERAYwINT0 (w=0-1)	E-RAYw service request 0	$011B0_H$+w*30_H	M	SV, M, 16	Application Reset	6980
SRC_ERAYwINT1 (w=0-1)	E-RAYw service request 1	$011B4_H$+w*30_H	M	SV, M, 16	Application Reset	6980
SRC_ERAYwTINT0 (w=0-1)	E-RAYw timer interrupt 0 service request	$011B8_H$+w*30_H	M	SV, M, 16	Application Reset	6970
SRC_ERAYwTINT1 (w=0-1)	E-RAYw timer interrupt 1 service request	$011BC_H$+w*30_H	M	SV, M, 16	Application Reset	6970
SRC_ERAYwNDAT0 (w=0-1)	E-RAYw new data 0 service request	$011C0_H$+w*30_H	M	SV, M, 16	Application Reset	6980
SRC_ERAYwNDAT1 (w=0-1)	E-RAYw new data 1 service request	$011C4_H$+w*30_H	M	SV, M, 16	Application Reset	6980
SRC_ERAYwMBSC0 (w=0-1)	E-RAYw message buffer status changed 0 service request	$011C8_H$+w*30_H	M	SV, M, 16	Application Reset	6980
SRC_ERAYwMBSC1 (w=0-1)	E-RAYw message buffer status changed 1 service request	$011CC_H$+w*30_H	M	SV, M, 16	Application Reset	6980
SRC_ERAYwOBUSY (w=0-1)	E-RAYw output buffer busy	$011D0_H$+w*30_H	M	SV, M, 16	Application Reset	6980
SRC_ERAYwIBUSY (w=0-1)	E-RAYw input buffer busy	$011D4_H$+w*30_H	M	SV, M, 16	Application Reset	6980
SRC_NVMHOST	NVM HOST service request	01210_H	M	SV, M, 16	Application Reset	6975
SRC_NVMCSRM	NVM CSRM service request	01214_H	M	SV, M, 16	Application Reset	6975
SRC_CSS0SRx (w=0;x=0-25)	CSS0 service request x	01220_H+x*4	M	SV, M, 16	Application Reset	6958
SRC_RNG	RNG service request	01290_H	M	SV, M, 16	Application Reset	6965

(table continues...)

Table 1938 (continued) Registers overview - SRC (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
SRC_PKC	PKC service request	01298_H	M	SV, M, 16	Application Reset	6982
SRC_SCUERUw (w=0-7)	SCU ERU service request w	$012A0_H$+ w*4	M	SV, M, 16	Application Reset	6962
SRC_PMSSRw (w=0-7)	Power management system service request w	$012D0_H$+ w*4	M	SV, M, 16	Application Reset	6982
SRC_SMMw (w=0-1)	SMM system service request w	$012F0_H$+ w*4	M	SV, M, 16	Application Reset	6962
SRC_SMUIRQx (x=0-5)	SMU_Safety service request x	01300_H+ x*4	M	SV, M, 16	Application Reset	6962
SRC_SMUCSIRQ	SMU_Security service request	01318_H	M	SV, M, 16	Application Reset	6962
SRC_PSI5SRx (x=0-7)	PSI5 service request x	01320_H+ x*4	M	SV, M, 16	Application Reset	6965
SRC_PSI5S0SRx (w=0;x=0-7)	PSI5S0 service request x	01340_H+ x*4	M	SV, M, 16	Application Reset	6965
SRC_PPUSTUDONE	PPU STU-DMI done service request	01380_H	M	SV, M, 16	Application Reset	6982
SRC_PPUSTUEVENT	PPU STU-DMI event service request	01384_H	M	SV, M, 16	Application Reset	6982
SRC_PPUSTUERR	PPU STU-DMI error service request	01388_H	M	SV, M, 16	Application Reset	6982
SRC_PPUICI	PPU inter-core interrupt (ICI) service request	$0138C_H$	M	SV, M, 16	Application Reset	6982
SRC_PCIEwMSIx (w=0-1;x=0-7)	PCIEw MSI and MSI-X service request x	01390_H+ w*60_H+x *4	M	SV, M, 16	Application Reset	6975
SRC_PCIEwWCx (w=0-1;x=0-3)	PCIEw write channel x service request	$013B0_H$+ w*60_H+x *4	M	SV, M, 16	Application Reset	6982
SRC_PCIEwRCx (w=0-1;x=0-3)	PCIEw read channel x service request	$013C0_H$+ w*60_H+x *4	M	SV, M, 16	Application Reset	6982
SRC_PCIEwERR (w=0-1)	PCIEw general error service request	$013D0_H$+ w*60_H	M	SV, M, 16	Application Reset	6975
SRC_PCIEwNFAT (w=0-1)	PCIEw non-fatal error and fatal error service request	$013D4_H$+ w*60_H	M	SV, M, 16	Application Reset	6982
SRC_PCIEwPM (w=0-1)	PCIEw power management service request	$013D8_H$+ w*60_H	M	SV, M, 16	Application Reset	6982

(table continues...)

Table 1938 (continued) Registers overview - SRC (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
SRC_PCIEwLNK (w=0-1)	PCIEw link state change / link reset service request	$013DC_H$+ w*60_H	M	SV, M, 16	Application Reset	6975
SRC_PCIEwMSG (w=0-1)	PCIEw general message received service request	$013E0_H$+ w*60_H	M	SV, M, 16	Application Reset	6975
SRC_PCIEwMISC (w=0-1)	PCIEw miscellaneous service request	$013E4_H$+ w*60_H	M	SV, M, 16	Application Reset	6975
SRC_XSPIFIFO	xSPI FIFO service request	01450_H	M	SV, M, 16	Application Reset	6985
SRC_XSPIERROR	xSPI interface error service request	01454_H	M	SV, M, 16	Application Reset	6960
SRC_GPSRwSRx (w=0-5;x=0-7)	General purpose group w service request x	01460_H+ w*20_H+x *4	M	SV, M, 16	Application Reset	6970
SRC_GPSR6SRx (x=0-7)	General purpose group 6 service request x	01520_H+ x*4	M	SV, M, 16	Application Reset	6970
SRC_GPSR7SRx (x=0-7)	General purpose group 7 service request x	01540_H+ x*4	M	SV, M, 16	Application Reset	6970
SRC_DREw (w=0-15)	DRE service request w	01560_H+ w*4	M	SV, M, 16	Application Reset	6977
SRC_EGTMAEIIRQ	AEI service request	$01E10_H$	M	SV, M, 16	Application Reset	6977
SRC_EGTMCMPIRQ	CMP service request	$01E1C_H$	M	SV, M, 16	Application Reset	6977
SRC_EGTMSPExIRQ (x=0-2)	SPEx service request	$01E20_H$+ x*4	M	SV, M, 16	Application Reset	6977
SRC_EGTMERR	Error service request	$01E50_H$	M	SV, M, 16	Application Reset	6977
SRC_EGTMTIMxSRy (x=0-2;y=0-7)	TIMx service request y	$01E60_H$+ x*20_H+y*4	M	SV, M, 16	Application Reset	6980
SRC_EGTMTOMxSRy (x=0-2;y=0-7)	TOMx service request y	$01F20_H$+ x*20_H+y*4	M	SV, M, 16	Application Reset	6980
SRC_EGTMATOMxSRy (x=0-2;y=0-3)	ATOMx service request y	$01FA0_H$+ x*10_H+y*4	M	SV, M, 16	Application Reset	6977

40.8.4.2.4 CPUw software breakpoint service request

The access protection mechanism for the SRCx register is described in chapter IR, access protection of SRCx registers.

SRC_CPUwSB (w=0-5)	Offset address:	00000_H + w*4
CPUw software breakpoint service request	Application Reset value:	$0000\ 0000_H$
SRC_CPUCSSB	Offset address:	00018_H
CPUCS software breakpoint service request	Application Reset value:	$0000\ 0000_H$
SRC_COMBCUNCS	Offset address:	$001F8_H$
COMBCU NCS service request	Application Reset value:	$0000\ 0000_H$
SRC_COMBCUCS	Offset address:	$001FC_H$
COMBCU CS service request	Application Reset value:	$0000\ 0000_H$
SRC_COMBCUQOSC	Offset address:	00200_H
COMBCU QOS control service request	Application Reset value:	$0000\ 0000_H$
SRC_CSBCUNCS	Offset address:	$0021C_H$
CSBCU NCS service request	Application Reset value:	$0000\ 0000_H$
SRC_CSBCUCS	Offset address:	00220_H
CSBCU CS service request	Application Reset value:	$0000\ 0000_H$
SRC_DMA0ERRx (x=0-15)	Offset address:	$006E0_H$+x*4
DMA0 RPx error service request	Application Reset value:	$0000\ 0000_H$
SRC_DMA0CHx (x=0-127)	Offset address:	00730_H+x*4
DMA0 channel x service request	Application Reset value:	$0000\ 0000_H$
SRC_CSS0SRx (x=0-25)	Offset address:	01220_H+x*4
CSS0 service request x	Application Reset value:	$0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0			IOVCLR	IOV	SETR	CLRR	SRR	SRE	0						
r			w	rh	w	w	rh	rw	r						

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TOS				CS	VM			SRPN							
rw				rw	rw			rw							

Field	Bits	Type	Description
SRPN	7:0	rw	**Service request priority number** The SRPN bit field defines the priority of a service request. The larger SRPN number has the higher priority.

(table continues...)

(continued)

Field	Bits	Type	Description
VM	10:8	rw	**Virtual machine** The VM bit field configuration maps a service request to a virtual machine. 000_B Service request is mapped to VM0 001_B Service request is mapped to VM1 010_B Service request is mapped to VM2 011_B Service request is mapped to VM3 100_B Service request is mapped to VM4 101_B Service request is mapped to VM5 110_B Service request is mapped to VM6 111_B Service request is mapped to VM7
CS	11	rw	**Cyber security** 0_B Service request is not assigned to the security application 1_B Service request is assigned to the security application
TOS	15:12	rw	**Type of service** The TOS bit field configuration maps a service request to an interrupt service provider: 0_H Service request is mapped to CPU0 1_H Service request is mapped to CPU1 2_H Service request is mapped to CPU2 3_H Service request is mapped to CPU3 4_H Service request is mapped to CPU4 5_H Service request is mapped to CPU5 6_H Service request is mapped to CPUCS 7_H Service request is mapped to DMA0 8_H Service request is mapped to DMA1 A_H Service request is mapped to PPU F_H SRN not used **others,** Reserved
SRE	23	rw	**Service request enable** 0_B Service request is disabled 1_B Service request is enabled
SRR	24	rh	**Service request status** The SRR bit shows the status of the service request. 0_B No service request is pending 1_B Service request is pending
CLRR	25	w	**Service request clear** The CLRR bit is required to reset SRR. 0_B No action 1_B Write with 1 leads to SRR=0; no action if SETR is written with 1 at the same time; no clear of SRR if a hardware interrupt trigger happens at the same time. read always returns 0.

(table continues...)

(continued)

Field	Bits	Type	Description
SETR	26	w	**Service Request set** The SETR bit is required to set SRR. 0_B No action 1_B Write with 1 leads to SRR=1; if SRR is already =1 then IOV is set to 1; no action if CLRR is written with 1 at the same time; read always returns 0.
IOV	27	rh	**Interrupt overflow status** The IOV bit is set by HW when a new service request is triggered by interrupt trigger or write to the SETR bit with 1 while SRR=1. 0_B No interrupt overflow detected 1_B Interrupt overflow detected
IOVCLR	28	w	**Interrupt overflow clear** IOVCLR is required to reset IOV. 0_B No action 1_B Write with 1 lead to IOV=0; no action in case of an IOV condition at the same time; read always returns 0.
0	22:16, 31:29	r	**Reserved-0** Read as 0; should be written with 0.

40.8.4.2.5 CPUw system timer service requests SRx

SRC_STMCPUwSRx (w=0-5;x=0-15)	Offset address:	00020_H+w*40_H+x*4
CPUw system timer service requests SRx	Application Reset value:	0000 0000_H
SRC_STMCPUCSSR2	Offset address:	$001A8_H$
STM1 CPUCS service request 0	Application Reset value:	0000 0000_H
SRC_STMCPUCSSR3	Offset address:	$001AC_H$
STM1 CPUCS service request 1	Application Reset value:	0000 0000_H
SRC_TBCUNCS	Offset address:	00210_H
TBCU NCS service request	Application Reset value:	0000 0000_H
SRC_TBCUCS	Offset address:	00214_H
TBCU CS service request	Application Reset value:	0000 0000_H
SRC_TRIF	Offset address:	00250_H
TRIF status service request	Application Reset value:	0000 0000_H
SRC_SRIwSM (w=0-7)	Offset address:	00264_H+w*8
SRI domain w special master service request	Application Reset value:	0000 0000_H
SRC_VMTwSR (w=0-6)	Offset address:	00410_H+w*4
VMTw service request	Application Reset value:	0000 0000_H
SRC_TMADCwSRx (w=0-3;x=0-5)	Offset address:	$00E10_H$+w*18_H+x*4
TMADCw service request x	Application Reset value:	0000 0000_H
SRC_XSPIERROR	Offset address:	01454_H
xSPI interface error service request	Application Reset value:	0000 0000_H

Field	Bits	Type	Description
SRPN	7:0	rw	**Service request priority number** The SRPN bit field defines the priority of a service request. The larger SRPN number has the higher priority.
VM	10:8	rw	**Virtual machine** The VM bit field configuration maps a service request to a virtual machine. 000_B Service request is mapped to VM0 001_B Service request is mapped to VM1 010_B Service request is mapped to VM2 011_B Service request is mapped to VM3 100_B Service request is mapped to VM4 101_B Service request is mapped to VM5 110_B Service request is mapped to VM6 111_B Service request is mapped to VM7
CS	11	rw	**Cyber security** 0_B Service request is not assigned to the security application 1_B Service request is assigned to the security application
TOS	15:12	rw	**Type of service** The TOS bit field configuration maps a service request to an interrupt service provider: 0_H Service request is mapped to CPU0 1_H Service request is mapped to CPU1 2_H Service request is mapped to CPU2 3_H Service request is mapped to CPU3 4_H Service request is mapped to CPU4 5_H Service request is mapped to CPU5 6_H Service request is mapped to CPUCS 7_H Service request is mapped to DMA0 8_H Service request is mapped to DMA1 A_H Service request is mapped to PPU F_H SRN not used **others,** Reserved
SRE	23	rw	**Service request enable** 0_B Service request is disabled 1_B Service request is enabled

(table continues...)

(continued)

Field	Bits	Type	Description
SRR	24	rh	**Service request status** The SRR bit shows the status of the service request. 0_B No service request is pending 1_B Service request is pending
CLRR	25	w	**Service request clear** The CLRR bit is required to reset SRR. 0_B No action 1_B Write with 1 leads to SRR=0; no action if SETR is written with 1 at the same time; no clear of SRR if a hardware interrupt trigger happens at the same time. read always returns 0.
SETR	26	w	**Service Request set** The SETR bit is required to set SRR. 0_B No action 1_B Write with 1 leads to SRR=1; if SRR is already =1 then IOV is set to 1; no action if CLRR is written with 1 at the same time; read always returns 0.
IOV	27	rh	**Interrupt overflow status** The IOV bit is set by HW when a new service request is triggered by interrupt trigger or write to the SETR bit with 1 while SRR=1. 0_B No interrupt overflow detected 1_B Interrupt overflow detected
IOVCLR	28	w	**Interrupt overflow clear** IOVCLR is required to reset IOV. 0_B No action 1_B Write with 1 lead to IOV=0; no action in case of an IOV condition at the same time; read always returns 0.
0	22:16, 31:29	r	**Reserved-0** Read as 0; should be written with 0.

40.8.4.2.6 SBCU NCS service request

SRC_SBCUNCS	Offset address:	$001E0_H$
SBCU NCS service request	Application Reset value:	$0000\ 0000_H$
SRC_SBCUQOSC	Offset address:	$001E8_H$
SBCU QOS control service request	Application Reset value:	$0000\ 0000_H$
SRC_SRIw (w=0-7)	Offset address:	00260_H+w*8
SRI domain w service request	Application Reset value:	$0000\ 0000_H$
SRC_SENTwSRx (w=0-1;x=0-9)	Offset address:	00610_H+w*28_H+x*4
SENTw service request x	Application Reset value:	$0000\ 0000_H$
SRC_SDMMC0ERR	Offset address:	$00B80_H$
SDMMC0 error service request	Application Reset value:	$0000\ 0000_H$

SRC_SDMMC0DMA Offset address: $00B84_H$

SDMMC0 DMA ready service request Application Reset value: $0000\ 0000_H$

SRC_SCUERUw (w=0-7) Offset address: $012A0_H+w*4$

SCU ERU service request w Application Reset value: $0000\ 0000_H$

SRC_SMMw (w=0-1) Offset address: $012F0_H+w*4$

SMM system service request w Application Reset value: $0000\ 0000_H$

SRC_SMUIRQx (x=0-5) Offset address: 01300_H+x*4

SMU_Safety service request x Application Reset value: $0000\ 0000_H$

SRC_SMUCSIRQ Offset address: 01318_H

SMU_Security service request Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0			IOVCLR	IOV	SETR	CLRR	SRR	SRE	0						
r			w	rh	w	w	rh	rw	r						

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TOS				CS	VM			SRPN							
rw				rw	rw			rw							

Field	Bits	Type	Description
SRPN	7:0	rw	**Service request priority number** The SRPN bit field defines the priority of a service request. The larger SRPN number has the higher priority.
VM	10:8	rw	**Virtual machine** The VM bit field configuration maps a service request to a virtual machine. 000_B Service request is mapped to VM0 001_B Service request is mapped to VM1 010_B Service request is mapped to VM2 011_B Service request is mapped to VM3 100_B Service request is mapped to VM4 101_B Service request is mapped to VM5 110_B Service request is mapped to VM6 111_B Service request is mapped to VM7
CS	11	rw	**Cyber security** 0_B Service request is not assigned to the security application 1_B Service request is assigned to the security application

(table continues...)

(continued)

Field	Bits	Type	Description
TOS	15:12	rw	**Type of service** The TOS bit field configuration maps a service request to an interrupt service provider: 0_H Service request is mapped to CPU0 1_H Service request is mapped to CPU1 2_H Service request is mapped to CPU2 3_H Service request is mapped to CPU3 4_H Service request is mapped to CPU4 5_H Service request is mapped to CPU5 6_H Service request is mapped to CPUCS 7_H Service request is mapped to DMA0 8_H Service request is mapped to DMA1 A_H Service request is mapped to PPU F_H SRN not used **others,** Reserved
SRE	23	rw	**Service request enable** 0_B Service request is disabled 1_B Service request is enabled
SRR	24	rh	**Service request status** The SRR bit shows the status of the service request. 0_B No service request is pending 1_B Service request is pending
CLRR	25	w	**Service request clear** The CLRR bit is required to reset SRR. 0_B No action 1_B Write with 1 leads to SRR=0; no action if SETR is written with 1 at the same time; no clear of SRR if a hardware interrupt trigger happens at the same time. read always returns 0.
SETR	26	w	**Service Request set** The SETR bit is required to set SRR. 0_B No action 1_B Write with 1 leads to SRR=1; if SRR is already =1 then IOV is set to 1; no action if CLRR is written with 1 at the same time; read always returns 0.
IOV	27	rh	**Interrupt overflow status** The IOV bit is set by HW when a new service request is triggered by interrupt trigger or write to the SETR bit with 1 while SRR=1. 0_B No interrupt overflow detected 1_B Interrupt overflow detected

(table continues...)

(continued)

Field	Bits	Type	Description
IOVCLR	28	w	**Interrupt overflow clear** IOVCLR is required to reset IOV. 0_B No action 1_B Write with 1 lead to IOV=0; no action in case of an IOV condition at the same time; read always returns 0.
0	22:16, 31:29	r	**Reserved-0** Read as 0; should be written with 0.

40.8.4.2.7 SBCU CS service request

SRC_SBCUCS Offset address: $001E4_H$
SBCU CS service request Application Reset value: $0000\ 0000_H$
SRC_QOSARPw (w=0-7) Offset address: 00230_H+w*4
QOS ARP w service request Application Reset value: $0000\ 0000_H$
SRC_QSPIwTX (w=0-7) Offset address: 00460_H+w*14_H
QSPIw transmit service request Application Reset value: $0000\ 0000_H$
SRC_QSPIwRX (w=0-7) Offset address: 00464_H+w*14_H
QSPIw receive service request Application Reset value: $0000\ 0000_H$
SRC_QSPIwERR (w=0-7) Offset address: 00468_H+w*14_H
QSPIw error service request Application Reset value: $0000\ 0000_H$
SRC_QSPIwPT (w=0-7) Offset address: $0046C_H+w*14_H$
QSPIw phase transition service request Application Reset value: $0000\ 0000_H$
SRC_QSPIwU (w=0-7) Offset address: 00470_H+w*14_H
QSPIw user defined service request Application Reset value: $0000\ 0000_H$
SRC_RNG Offset address: 01290_H
RNG service request Application Reset value: $0000\ 0000_H$
SRC_PSI5SRx (x=0-7) Offset address: 01320_H+x*4
PSI5 service request x Application Reset value: $0000\ 0000_H$
SRC_PSI5S0SRx (x=0-7) Offset address: 01340_H+x*4
PSI5S0 service request x Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0			IOVCLR	IOV	SETR	CLRR	SRR	SRE	0						
r			w	rh	w	w	rh	rw	r						

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TOS				CS	VM			SRPN							
rw				rw	rw			rw							

Field	Bits	Type	Description
SRPN	7:0	rw	**Service request priority number** The SRPN bit field defines the priority of a service request. The larger SRPN number has the higher priority.
VM	10:8	rw	**Virtual machine** The VM bit field configuration maps a service request to a virtual machine. 000_B Service request is mapped to VM0 001_B Service request is mapped to VM1 010_B Service request is mapped to VM2 011_B Service request is mapped to VM3 100_B Service request is mapped to VM4 101_B Service request is mapped to VM5 110_B Service request is mapped to VM6 111_B Service request is mapped to VM7
CS	11	rw	**Cyber security** 0_B Service request is not assigned to the security application 1_B Service request is assigned to the security application
TOS	15:12	rw	**Type of service** The TOS bit field configuration maps a service request to an interrupt service provider: 0_H Service request is mapped to CPU0 1_H Service request is mapped to CPU1 2_H Service request is mapped to CPU2 3_H Service request is mapped to CPU3 4_H Service request is mapped to CPU4 5_H Service request is mapped to CPU5 6_H Service request is mapped to CPUCS 7_H Service request is mapped to DMA0 8_H Service request is mapped to DMA1 A_H Service request is mapped to PPU F_H SRN not used **others,** Reserved
SRE	23	rw	**Service request enable** 0_B Service request is disabled 1_B Service request is enabled
SRR	24	rh	**Service request status** The SRR bit shows the status of the service request. 0_B No service request is pending 1_B Service request is pending

(table continues...)

(continued)

Field	Bits	Type	Description
CLRR	25	w	**Service request clear** The CLRR bit is required to reset SRR. 0_B No action 1_B Write with 1 leads to SRR=0; no action if SETR is written with 1 at the same time; no clear of SRR if a hardware interrupt trigger happens at the same time. read always returns 0.
SETR	26	w	**Service Request set** The SETR bit is required to set SRR. 0_B No action 1_B Write with 1 leads to SRR=1; if SRR is already =1 then IOV is set to 1; no action if CLRR is written with 1 at the same time; read always returns 0.
IOV	27	rh	**Interrupt overflow status** The IOV bit is set by HW when a new service request is triggered by interrupt trigger or write to the SETR bit with 1 while SRR=1. 0_B No interrupt overflow detected 1_B Interrupt overflow detected
IOVCLR	28	w	**Interrupt overflow clear** IOVCLR is required to reset IOV. 0_B No action 1_B Write with 1 lead to IOV=0; no action in case of an IOV condition at the same time; read always returns 0.
0	22:16, 31:29	r	**Reserved-0** Read as 0; should be written with 0.

40.8.4.2.8 Cerberus service request w

SRC_CERBERUSw (w=0-1)	Offset address:	$002A0_H$+w*4
Cerberus service request w	Application Reset value:	0000 0000_H
SRC_ASCLINwTX (w=0-27)	Offset address:	$002B0_H$+w*12
ASCLINw transmit service request	Application Reset value:	0000 0000_H
SRC_ASCLINwRX (w=0-27)	Offset address:	$002B4_H$+w*12
ASCLINw receive service request	Application Reset value:	0000 0000_H
SRC_ASCLINwERR (w=0-27)	Offset address:	$002B8_H$+w*12
ASCLINw error service request	Application Reset value:	0000 0000_H
SRC_CANwINTx (w=0-4;x=0-15)	Offset address:	$00C50_H$+w*40_H+x*4
CANw service request x	Application Reset value:	0000 0000_H
SRC_CANXLwFUNC (w=0-3)	Offset address:	$00DD0_H$+w*12
CANXLw functional service request	Application Reset value:	0000 0000_H
SRC_CANXLwERR (w=0-3)	Offset address:	$00DD4_H$+w*12
CANXLw error service request	Application Reset value:	0000 0000_H

SRC_CANXLwSAFETY (w=0-3) Offset address: $00DD8_H+w*12$

CANXLw safety service request Application Reset value: $0000\ 0000_H$

SRC_ADCSRx (x=0-11) Offset address: $00ED0_H+x*4$

ADC service request x Application Reset value: $0000\ 0000_H$

SRC_CDSPwSRx (w=0-1;x=0-2) Offset address: $010D0_H+w*12+x*4$

CDSPw service request x Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0			IOVCLR	IOV	SETR	CLRR	SRR	SRE	0						
r			w	rh	w	w	rh	rw	r						

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TOS				CS	VM			SRPN							
rw				rw	rw			rw							

Field	Bits	Type	Description
SRPN	7:0	rw	**Service request priority number** The SRPN bit field defines the priority of a service request. The larger SRPN number has the higher priority.
VM	10:8	rw	**Virtual machine** The VM bit field configuration maps a service request to a virtual machine. 000_B Service request is mapped to VM0 001_B Service request is mapped to VM1 010_B Service request is mapped to VM2 011_B Service request is mapped to VM3 100_B Service request is mapped to VM4 101_B Service request is mapped to VM5 110_B Service request is mapped to VM6 111_B Service request is mapped to VM7
CS	11	rw	**Cyber security** 0_B Service request is not assigned to the security application 1_B Service request is assigned to the security application

(table continues...)

(continued)

Field	Bits	Type	Description
TOS	15:12	rw	**Type of service** The TOS bit field configuration maps a service request to an interrupt service provider: 0_H Service request is mapped to CPU0 1_H Service request is mapped to CPU1 2_H Service request is mapped to CPU2 3_H Service request is mapped to CPU3 4_H Service request is mapped to CPU4 5_H Service request is mapped to CPU5 6_H Service request is mapped to CPUCS 7_H Service request is mapped to DMA0 8_H Service request is mapped to DMA1 A_H Service request is mapped to PPU F_H SRN not used **others,** Reserved
SRE	23	rw	**Service request enable** 0_B Service request is disabled 1_B Service request is enabled
SRR	24	rh	**Service request status** The SRR bit shows the status of the service request. 0_B No service request is pending 1_B Service request is pending
CLRR	25	w	**Service request clear** The CLRR bit is required to reset SRR. 0_B No action 1_B Write with 1 leads to SRR=0; no action if SETR is written with 1 at the same time; no clear of SRR if a hardware interrupt trigger happens at the same time. read always returns 0.
SETR	26	w	**Service Request set** The SETR bit is required to set SRR. 0_B No action 1_B Write with 1 leads to SRR=1; if SRR is already =1 then IOV is set to 1; no action if CLRR is written with 1 at the same time; read always returns 0.
IOV	27	rh	**Interrupt overflow status** The IOV bit is set by HW when a new service request is triggered by interrupt trigger or write to the SETR bit with 1 while SRR=1. 0_B No interrupt overflow detected 1_B Interrupt overflow detected

(table continues...)

(continued)

Field	Bits	Type	Description
IOVCLR	28	w	**Interrupt overflow clear** IOVCLR is required to reset IOV. 0_B No action 1_B Write with 1 lead to IOV=0; no action in case of an IOV condition at the same time; read always returns 0.
0	22:16, 31:29	r	**Reserved-0** Read as 0; should be written with 0.

40.8.4.2.9 HSCTw service request

SRC_HSCTw (w=0-1) Offset address: 00540_H+w*4
HSCTw service request Application Reset value: 0000 0000_H
SRC_FCE Offset address: $006D0_H$
FCE error service request Application Reset value: 0000 0000_H
SRC_GETH0DMAx (x=0-15) Offset address: $00B90_H$+x*4
GETH0 DMAx service request Application Reset value: 0000 0000_H
SRC_GETH0PPSx (x=0-3) Offset address: $00BD0_H$+x*4
GETH0 PPSx service request Application Reset value: 0000 0000_H
SRC_GETH0INTR Offset address: $00BE0_H$
GETH0 INTR service request Application Reset value: 0000 0000_H
SRC_ERAYwTINT0 (w=0-1) Offset address: $011B8_H$+w*30_H
E-RAYw timer interrupt 0 service request Application Reset value: 0000 0000_H
SRC_ERAYwTINT1 (w=0-1) Offset address: $011BC_H$+w*30_H
E-RAYw timer interrupt 1 service request Application Reset value: 0000 0000_H
SRC_GPSRwSRx (w=0-5;x=0-7) Offset address: 01460_H+w*20_H+x*4
General purpose group w service request x Application Reset value: 0000 0000_H
SRC_GPSR6SRx (x=0-7) Offset address: 01520_H+x*4
General purpose group 6 service request x Application Reset value: 0000 0000_H
SRC_GPSR7SRx (x=0-7) Offset address: 01540_H+x*4
General purpose group 7 service request x Application Reset value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0			IOVCLR	IOV	SETR	CLRR	SRR	SRE	0						
r			w	rh	w	w	rh	rw	r						

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TOS				CS	VM			SRPN							
rw				rw	rw			rw							

Field	Bits	Type	Description
SRPN	7:0	rw	**Service request priority number** The SRPN bit field defines the priority of a service request. The larger SRPN number has the higher priority.
VM	10:8	rw	**Virtual machine** The VM bit field configuration maps a service request to a virtual machine. 000_B Service request is mapped to VM0 001_B Service request is mapped to VM1 010_B Service request is mapped to VM2 011_B Service request is mapped to VM3 100_B Service request is mapped to VM4 101_B Service request is mapped to VM5 110_B Service request is mapped to VM6 111_B Service request is mapped to VM7
CS	11	rw	**Cyber security** 0_B Service request is not assigned to the security application 1_B Service request is assigned to the security application
TOS	15:12	rw	**Type of service** The TOS bit field configuration maps a service request to an interrupt service provider: 0_H Service request is mapped to CPU0 1_H Service request is mapped to CPU1 2_H Service request is mapped to CPU2 3_H Service request is mapped to CPU3 4_H Service request is mapped to CPU4 5_H Service request is mapped to CPU5 6_H Service request is mapped to CPUCS 7_H Service request is mapped to DMA0 8_H Service request is mapped to DMA1 A_H Service request is mapped to PPU F_H SRN not used **others,** Reserved
SRE	23	rw	**Service request enable** 0_B Service request is disabled 1_B Service request is enabled
SRR	24	rh	**Service request status** The SRR bit shows the status of the service request. 0_B No service request is pending 1_B Service request is pending

(table continues...)

(continued)

Field	Bits	Type	Description
CLRR	25	w	**Service request clear** The CLRR bit is required to reset SRR. 0_B No action 1_B Write with 1 leads to SRR=0; no action if SETR is written with 1 at the same time; no clear of SRR if a hardware interrupt trigger happens at the same time. read always returns 0.
SETR	26	w	**Service Request set** The SETR bit is required to set SRR. 0_B No action 1_B Write with 1 leads to SRR=1; if SRR is already =1 then IOV is set to 1; no action if CLRR is written with 1 at the same time; read always returns 0.
IOV	27	rh	**Interrupt overflow status** The IOV bit is set by HW when a new service request is triggered by interrupt trigger or write to the SETR bit with 1 while SRR=1. 0_B No interrupt overflow detected 1_B Interrupt overflow detected
IOVCLR	28	w	**Interrupt overflow clear** IOVCLR is required to reset IOV. 0_B No action 1_B Write with 1 lead to IOV=0; no action in case of an IOV condition at the same time; read always returns 0.
0	22:16, 31:29	r	**Reserved-0** Read as 0; should be written with 0.

40.8.4.2.10 HSSLw channel x OK service request

SRC_HSSLwCOKx (w=0-1;x=0-3) Offset address: 00550_H+w*44_H+x*10 H

HSSLw channel x OK service request Application Reset value: $0000\ 0000_H$

SRC_HSSLwRDIx (w=0-1;x=0-3) Offset address: 00554_H+w*44_H+x*10 H

HSSLw channel x read data service request Application Reset value: $0000\ 0000_H$

SRC_HSSLwERRx (w=0-1;x=0-3) Offset address: 00558_H+w*44_H+x*10 H

HSSLw channel x error service request Application Reset value: $0000\ 0000_H$

SRC_HSSLwTRGx (w=0-1;x=0-3) Offset address: $0055C_H$+w*44_H+x*10 H

HSSLw channel x trigger interrupt service request Application Reset value: $0000\ 0000_H$

SRC_HSSLwEXI (w=0-1) Offset address: 00590_H+w*44_H

HSSw exception service request Application Reset value: $0000\ 0000_H$

SRC_I2CwDTR (w=0-2) Offset address: $005E0_H$+w*10_H

I2Cw data transfer request Application Reset value: $0000\ 0000_H$

SRC_I2CwERR (w=0-2) Offset address: $005E4_H+w*10_H$

I2Cw error service request Application Reset value: $0000\ 0000_H$

SRC_I2CwP (w=0-2) Offset address: $005E8_H+w*10_H$

I2Cw Protocol service request Application Reset value: $0000\ 0000_H$

SRC_LETH0DMATXx (x=0-7) Offset address: $00BF0_H+x*4$

LETH0 DMA TXx service request Application Reset value: $0000\ 0000_H$

SRC_LETH0DMARXx (x=0-7) Offset address: $00C10_H+x*4$

LETH0 DMA RXx service request Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0			IOVCLR	IOV	SETR	CLRR	SRR	SRE	0						
r			w	rh	w	w	rh	rw	r						

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TOS				CS	VM			SRPN							
rw				rw	rw			rw							

Field	Bits	Type	Description
SRPN	7:0	rw	**Service request priority number** The SRPN bit field defines the priority of a service request. The larger SRPN number has the higher priority.
VM	10:8	rw	**Virtual machine** The VM bit field configuration maps a service request to a virtual machine. 000_B Service request is mapped to VM0 001_B Service request is mapped to VM1 010_B Service request is mapped to VM2 011_B Service request is mapped to VM3 100_B Service request is mapped to VM4 101_B Service request is mapped to VM5 110_B Service request is mapped to VM6 111_B Service request is mapped to VM7
CS	11	rw	**Cyber security** 0_B Service request is not assigned to the security application 1_B Service request is assigned to the security application

(table continues...)

(continued)

Field	Bits	Type	Description
TOS	15:12	rw	**Type of service** The TOS bit field configuration maps a service request to an interrupt service provider: 0_H Service request is mapped to CPU0 1_H Service request is mapped to CPU1 2_H Service request is mapped to CPU2 3_H Service request is mapped to CPU3 4_H Service request is mapped to CPU4 5_H Service request is mapped to CPU5 6_H Service request is mapped to CPUCS 7_H Service request is mapped to DMA0 8_H Service request is mapped to DMA1 A_H Service request is mapped to PPU F_H SRN not used **others,** Reserved
SRE	23	rw	**Service request enable** 0_B Service request is disabled 1_B Service request is enabled
SRR	24	rh	**Service request status** The SRR bit shows the status of the service request. 0_B No service request is pending 1_B Service request is pending
CLRR	25	w	**Service request clear** The CLRR bit is required to reset SRR. 0_B No action 1_B Write with 1 leads to SRR=0; no action if SETR is written with 1 at the same time; no clear of SRR if a hardware interrupt trigger happens at the same time. read always returns 0.
SETR	26	w	**Service Request set** The SETR bit is required to set SRR. 0_B No action 1_B Write with 1 leads to SRR=1; if SRR is already =1 then IOV is set to 1; no action if CLRR is written with 1 at the same time; read always returns 0.
IOV	27	rh	**Interrupt overflow status** The IOV bit is set by HW when a new service request is triggered by interrupt trigger or write to the SETR bit with 1 while SRR=1. 0_B No interrupt overflow detected 1_B Interrupt overflow detected

(table continues...)

(continued)

Field	Bits	Type	Description
IOVCLR	28	w	**Interrupt overflow clear** IOVCLR is required to reset IOV. 0_B No action 1_B Write with 1 lead to IOV=0; no action in case of an IOV condition at the same time; read always returns 0.
0	22:16, 31:29	r	**Reserved-0** Read as 0; should be written with 0.

40.8.4.2.11 MSC0 service request x

SRC_MSC0SRx (x=0-4) Offset address: 00660_H+x*4
MSC0 service request x Application Reset value: $0000\ 0000_H$
SRC_LETH0PPSx (x=0-3) Offset address: $00C30_H$+x*4
LETH0 Px_PPS service request Application Reset value: $0000\ 0000_H$
SRC_LETH0INTR Offset address: $00C40_H$
LETH0 INTR service request Application Reset value: $0000\ 0000_H$
SRC_NVMHOST Offset address: 01210_H
NVM HOST service request Application Reset value: $0000\ 0000_H$
SRC_NVMCSRM Offset address: 01214_H
NVM CSRM service request Application Reset value: $0000\ 0000_H$
SRC_PCIEwMSIx (w=0-1;x=0-7) Offset address: 01390_H+w*60_H+x*4
PCIEw MSI and MSI-X service request x Application Reset value: $0000\ 0000_H$
SRC_PCIEwERR (w=0-1) Offset address: $013D0_H$+w*60_H
PCIEw general error service request Application Reset value: $0000\ 0000_H$
SRC_PCIEwLNK (w=0-1) Offset address: $013DC_H$+w*60_H
PCIEw link state change / link reset service request Application Reset value: $0000\ 0000_H$
SRC_PCIEwMSG (w=0-1) Offset address: $013E0_H$+w*60_H
PCIEw general message received service request Application Reset value: $0000\ 0000_H$
SRC_PCIEwMISC (w=0-1) Offset address: $013E4_H$+w*60_H
PCIEw miscellaneous service request Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0			IOVCLR	IOV	SETR	CLRR	SRR	SRE	0						
r			w	rh	w	w	rh	rw	r						

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TOS				CS	VM			SRPN							
rw				rw	rw			rw							

Field	Bits	Type	Description
SRPN	7:0	rw	**Service request priority number** The SRPN bit field defines the priority of a service request. The larger SRPN number has the higher priority.
VM	10:8	rw	**Virtual machine** The VM bit field configuration maps a service request to a virtual machine. 000_B Service request is mapped to VM0 001_B Service request is mapped to VM1 010_B Service request is mapped to VM2 011_B Service request is mapped to VM3 100_B Service request is mapped to VM4 101_B Service request is mapped to VM5 110_B Service request is mapped to VM6 111_B Service request is mapped to VM7
CS	11	rw	**Cyber security** 0_B Service request is not assigned to the security application 1_B Service request is assigned to the security application
TOS	15:12	rw	**Type of service** The TOS bit field configuration maps a service request to an interrupt service provider: 0_H Service request is mapped to CPU0 1_H Service request is mapped to CPU1 2_H Service request is mapped to CPU2 3_H Service request is mapped to CPU3 4_H Service request is mapped to CPU4 5_H Service request is mapped to CPU5 6_H Service request is mapped to CPUCS 7_H Service request is mapped to DMA0 8_H Service request is mapped to DMA1 A_H Service request is mapped to PPU F_H SRN not used **others,** Reserved
SRE	23	rw	**Service request enable** 0_B Service request is disabled 1_B Service request is enabled
SRR	24	rh	**Service request status** The SRR bit shows the status of the service request. 0_B No service request is pending 1_B Service request is pending

(table continues...)

(continued)

Field	Bits	Type	Description
CLRR	25	w	**Service request clear** The CLRR bit is required to reset SRR. 0_B No action 1_B Write with 1 leads to SRR=0; no action if SETR is written with 1 at the same time; no clear of SRR if a hardware interrupt trigger happens at the same time. read always returns 0.
SETR	26	w	**Service Request set** The SETR bit is required to set SRR. 0_B No action 1_B Write with 1 leads to SRR=1; if SRR is already =1 then IOV is set to 1; no action if CLRR is written with 1 at the same time; read always returns 0.
IOV	27	rh	**Interrupt overflow status** The IOV bit is set by HW when a new service request is triggered by interrupt trigger or write to the SETR bit with 1 while SRR=1. 0_B No interrupt overflow detected 1_B Interrupt overflow detected
IOVCLR	28	w	**Interrupt overflow clear** IOVCLR is required to reset IOV. 0_B No action 1_B Write with 1 lead to IOV=0; no action in case of an IOV condition at the same time; read always returns 0.
0	22:16, 31:29	r	**Reserved-0** Read as 0; should be written with 0.

40.8.4.2.12 DMA0 global error service request

SRC_DMA0GERR	Offset address:	00720_H
DMA0 global error service request	Application Reset value:	$0000\ 0000_H$
SRC_DMA1ERRx (x=0-7)	Offset address:	00930_H+x*4
DMA1 RPx error service request	Application Reset value:	$0000\ 0000_H$
SRC_DMA1GERR	Offset address:	00970_H
DMA1 global error service request	Application Reset value:	$0000\ 0000_H$
SRC_DMA1CHx (x=0-63)	Offset address:	00980_H+x*4
DMA1 channel x service request	Application Reset value:	$0000\ 0000_H$
SRC_DREw (w=0-15)	Offset address:	01560_H+w*4
DRE service request w	Application Reset value:	$0000\ 0000_H$
SRC_EGTMAEIIRQ	Offset address:	$01E10_H$
AEI service request	Application Reset value:	$0000\ 0000_H$
SRC_EGTMCMPIRQ	Offset address:	$01E1C_H$
CMP service request	Application Reset value:	$0000\ 0000_H$

SRC_EGTMSPExIRQ (x=0-2) Offset address: $01E20_H+x*4$

SPEx service request Application Reset value: $0000\ 0000_H$

SRC_EGTMERR Offset address: $01E50_H$

Error service request Application Reset value: $0000\ 0000_H$

SRC_EGTMATOMxSRy (x=0-2;y=0-3) Offset address: $01FA0_H+x*10_H+y*4$

ATOMx service request y Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0			IOVCLR	IOV	SETR	CLRR	SRR	SRE	0						
r			w	rh	w	w	rh	rw	r						

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TOS				CS	VM			SRPN							
rw				rw	rw			rw							

Field	Bits	Type	Description
SRPN	7:0	rw	**Service request priority number** The SRPN bit field defines the priority of a service request. The larger SRPN number has the higher priority.
VM	10:8	rw	**Virtual machine** The VM bit field configuration maps a service request to a virtual machine. 000_B Service request is mapped to VM0 001_B Service request is mapped to VM1 010_B Service request is mapped to VM2 011_B Service request is mapped to VM3 100_B Service request is mapped to VM4 101_B Service request is mapped to VM5 110_B Service request is mapped to VM6 111_B Service request is mapped to VM7
CS	11	rw	**Cyber security** 0_B Service request is not assigned to the security application 1_B Service request is assigned to the security application

(table continues...)

(continued)

Field	Bits	Type	Description
TOS	15:12	rw	**Type of service** The TOS bit field configuration maps a service request to an interrupt service provider: 0_H Service request is mapped to CPU0 1_H Service request is mapped to CPU1 2_H Service request is mapped to CPU2 3_H Service request is mapped to CPU3 4_H Service request is mapped to CPU4 5_H Service request is mapped to CPU5 6_H Service request is mapped to CPUCS 7_H Service request is mapped to DMA0 8_H Service request is mapped to DMA1 A_H Service request is mapped to PPU F_H SRN not used **others,** Reserved
SRE	23	rw	**Service request enable** 0_B Service request is disabled 1_B Service request is enabled
SRR	24	rh	**Service request status** The SRR bit shows the status of the service request. 0_B No service request is pending 1_B Service request is pending
CLRR	25	w	**Service request clear** The CLRR bit is required to reset SRR. 0_B No action 1_B Write with 1 leads to SRR=0; no action if SETR is written with 1 at the same time; no clear of SRR if a hardware interrupt trigger happens at the same time. read always returns 0.
SETR	26	w	**Service Request set** The SETR bit is required to set SRR. 0_B No action 1_B Write with 1 leads to SRR=1; if SRR is already =1 then IOV is set to 1; no action if CLRR is written with 1 at the same time; read always returns 0.
IOV	27	rh	**Interrupt overflow status** The IOV bit is set by HW when a new service request is triggered by interrupt trigger or write to the SETR bit with 1 while SRR=1. 0_B No interrupt overflow detected 1_B Interrupt overflow detected

(table continues...)

(continued)

Field	Bits	Type	Description
IOVCLR	28	w	**Interrupt overflow clear** IOVCLR is required to reset IOV. 0_B No action 1_B Write with 1 lead to IOV=0; no action in case of an IOV condition at the same time; read always returns 0.
0	22:16, 31:29	r	**Reserved-0** Read as 0; should be written with 0.

40.8.4.2.13 E-RAYw service request 0

SRC_ERAYwINT0 (w=0-1) Offset address: $011B0_H+w*30_H$

E-RAYw service request 0 Application Reset value: $0000\ 0000_H$

SRC_ERAYwINT1 (w=0-1) Offset address: $011B4_H+w*30_H$

E-RAYw service request 1 Application Reset value: $0000\ 0000_H$

SRC_ERAYwNDAT0 (w=0-1) Offset address: $011C0_H+w*30_H$

E-RAYw new data 0 service request Application Reset value: $0000\ 0000_H$

SRC_ERAYwNDAT1 (w=0-1) Offset address: $011C4_H+w*30_H$

E-RAYw new data 1 service request Application Reset value: $0000\ 0000_H$

SRC_ERAYwMBSC0 (w=0-1) Offset address: $011C8_H+w*30_H$

E-RAYw message buffer status changed 0 service request Application Reset value: $0000\ 0000_H$

SRC_ERAYwMBSC1 (w=0-1) Offset address: $011CC_H+w*30_H$

E-RAYw message buffer status changed 1 service request Application Reset value: $0000\ 0000_H$

SRC_ERAYwOBUSY (w=0-1) Offset address: $011D0_H+w*30_H$

E-RAYw output buffer busy Application Reset value: $0000\ 0000_H$

SRC_ERAYwIBUSY (w=0-1) Offset address: $011D4_H+w*30_H$

E-RAYw input buffer busy Application Reset value: $0000\ 0000_H$

SRC_EGTMTIMxSRy (x=0-2;y=0-7) Offset address: $01E60_H+x*20_H+y*4$

TIMx service request y Application Reset value: $0000\ 0000_H$

SRC_EGTMTOMxSRy (x=0-2;y=0-7) Offset address: $01F20_H+x*20_H+y*4$

TOMx service request y Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0			IOVCLR	IOV	SETR	CLR R	SRR	SRE	0						
r			w	rh	w	w	rh	rw	r						

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TOS				CS	VM			SRPN							
rw				rw	rw			rw							

Field	Bits	Type	Description
SRPN	7:0	rw	**Service request priority number** The SRPN bit field defines the priority of a service request. The larger SRPN number has the higher priority.
VM	10:8	rw	**Virtual machine** The VM bit field configuration maps a service request to a virtual machine. 000_B Service request is mapped to VM0 001_B Service request is mapped to VM1 010_B Service request is mapped to VM2 011_B Service request is mapped to VM3 100_B Service request is mapped to VM4 101_B Service request is mapped to VM5 110_B Service request is mapped to VM6 111_B Service request is mapped to VM7
CS	11	rw	**Cyber security** 0_B Service request is not assigned to the security application 1_B Service request is assigned to the security application
TOS	15:12	rw	**Type of service** The TOS bit field configuration maps a service request to an interrupt service provider: 0_H Service request is mapped to CPU0 1_H Service request is mapped to CPU1 2_H Service request is mapped to CPU2 3_H Service request is mapped to CPU3 4_H Service request is mapped to CPU4 5_H Service request is mapped to CPU5 6_H Service request is mapped to CPUCS 7_H Service request is mapped to DMA0 8_H Service request is mapped to DMA1 A_H Service request is mapped to PPU F_H SRN not used **others,** Reserved
SRE	23	rw	**Service request enable** 0_B Service request is disabled 1_B Service request is enabled
SRR	24	rh	**Service request status** The SRR bit shows the status of the service request. 0_B No service request is pending 1_B Service request is pending

(table continues...)

(continued)

Field	Bits	Type	Description
CLRR	25	w	**Service request clear** The CLRR bit is required to reset SRR. 0_B No action 1_B Write with 1 leads to SRR=0; no action if SETR is written with 1 at the same time; no clear of SRR if a hardware interrupt trigger happens at the same time. read always returns 0.
SETR	26	w	**Service Request set** The SETR bit is required to set SRR. 0_B No action 1_B Write with 1 leads to SRR=1; if SRR is already =1 then IOV is set to 1; no action if CLRR is written with 1 at the same time; read always returns 0.
IOV	27	rh	**Interrupt overflow status** The IOV bit is set by HW when a new service request is triggered by interrupt trigger or write to the SETR bit with 1 while SRR=1. 0_B No interrupt overflow detected 1_B Interrupt overflow detected
IOVCLR	28	w	**Interrupt overflow clear** IOVCLR is required to reset IOV. 0_B No action 1_B Write with 1 lead to IOV=0; no action in case of an IOV condition at the same time; read always returns 0.
0	22:16, 31:29	r	**Reserved-0** Read as 0; should be written with 0.

40.8.4.2.14 PKC service request

SRC_PKC	Offset address:	01298_H
PKC service request	Application Reset value:	$0000\ 0000_H$
SRC_PMSSRw (w=0-7)	Offset address:	$012D0_H$+w*4
Power management system service request w	Application Reset value:	$0000\ 0000_H$
SRC_PPUSTUDONE	Offset address:	01380_H
PPU STU-DMI done service request	Application Reset value:	$0000\ 0000_H$
SRC_PPUSTUEVENT	Offset address:	01384_H
PPU STU-DMI event service request	Application Reset value:	$0000\ 0000_H$
SRC_PPUSTUERR	Offset address:	01388_H
PPU STU-DMI error service request	Application Reset value:	$0000\ 0000_H$
SRC_PPUICI	Offset address:	$0138C_H$
PPU inter-core interrupt [ICI) service request	Application Reset value:	$0000\ 0000_H$
SRC_PCIEwWCx (w=0-1;x=0-3)	Offset address:	$013B0_H$+w*60_H+x*4
PCIEw write channel x service request	Application Reset value:	$0000\ 0000_H$

SRC_PCIEwRCx (w=0-1;x=0-3) Offset address: $013C0_H+w*60_H+x*4$

PCIEw read channel x service request Application Reset value: $0000\ 0000_H$

SRC_PCIEwNFAT (w=0-1) Offset address: $013D4_H+w*60_H$

PCIEw non-fatal error and fatal error service request Application Reset value: $0000\ 0000_H$

SRC_PCIEwPM (w=0-1) Offset address: $013D8_H+w*60_H$

PCIEw power management service request Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0			IOVCLR	IOV	SETR	CLRR	SRR	SRE	0						
r			w	rh	w	w	rh	rw	r						

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TOS				CS	VM			SRPN							
rw				rw	rw			rw							

Field	Bits	Type	Description
SRPN	7:0	rw	**Service request priority number** The SRPN bit field defines the priority of a service request. The larger SRPN number has the higher priority.
VM	10:8	rw	**Virtual machine** The VM bit field configuration maps a service request to a virtual machine. 000_B Service request is mapped to VM0 001_B Service request is mapped to VM1 010_B Service request is mapped to VM2 011_B Service request is mapped to VM3 100_B Service request is mapped to VM4 101_B Service request is mapped to VM5 110_B Service request is mapped to VM6 111_B Service request is mapped to VM7
CS	11	rw	**Cyber security** 0_B Service request is not assigned to the security application 1_B Service request is assigned to the security application

(table continues...)

(continued)

Field	Bits	Type	Description
TOS	15:12	rw	**Type of service** The TOS bit field configuration maps a service request to an interrupt service provider: 0_H Service request is mapped to CPU0 1_H Service request is mapped to CPU1 2_H Service request is mapped to CPU2 3_H Service request is mapped to CPU3 4_H Service request is mapped to CPU4 5_H Service request is mapped to CPU5 6_H Service request is mapped to CPUCS 7_H Service request is mapped to DMA0 8_H Service request is mapped to DMA1 A_H Service request is mapped to PPU F_H SRN not used **others,** Reserved
SRE	23	rw	**Service request enable** 0_B Service request is disabled 1_B Service request is enabled
SRR	24	rh	**Service request status** The SRR bit shows the status of the service request. 0_B No service request is pending 1_B Service request is pending
CLRR	25	w	**Service request clear** The CLRR bit is required to reset SRR. 0_B No action 1_B Write with 1 leads to SRR=0; no action if SETR is written with 1 at the same time; no clear of SRR if a hardware interrupt trigger happens at the same time. read always returns 0.
SETR	26	w	**Service Request set** The SETR bit is required to set SRR. 0_B No action 1_B Write with 1 leads to SRR=1; if SRR is already =1 then IOV is set to 1; no action if CLRR is written with 1 at the same time; read always returns 0.
IOV	27	rh	**Interrupt overflow status** The IOV bit is set by HW when a new service request is triggered by interrupt trigger or write to the SETR bit with 1 while SRR=1. 0_B No interrupt overflow detected 1_B Interrupt overflow detected

(table continues...)

(continued)

Field	Bits	Type	Description
IOVCLR	28	w	**Interrupt overflow clear** IOVCLR is required to reset IOV. 0_B No action 1_B Write with 1 lead to IOV=0; no action in case of an IOV condition at the same time; read always returns 0.
0	22:16, 31:29	r	**Reserved-0** Read as 0; should be written with 0.

40.8.4.2.15 xSPI FIFO service request

SRC_XSPIFIFO Offset address: 01450_H

xSPI FIFO service request Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0			IOVCLR	IOV	SETR	CLRR	SRR	SRE	0						
r			w	rh	w	w	rh	rw	r						

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TOS				CS	VM			SRPN							
rw				rw	rw			rw							

Field	Bits	Type	Description
SRPN	7:0	rw	**Service request priority number** The SRPN bit field defines the priority of a service request. The larger SRPN number has the higher priority.
VM	10:8	rw	**Virtual machine** The VM bit field configuration maps a service request to a virtual machine. 000_B Service request is mapped to VM0 001_B Service request is mapped to VM1 010_B Service request is mapped to VM2 011_B Service request is mapped to VM3 100_B Service request is mapped to VM4 101_B Service request is mapped to VM5 110_B Service request is mapped to VM6 111_B Service request is mapped to VM7
CS	11	rw	**Cyber security** 0_B Service request is not assigned to the security application 1_B Service request is assigned to the security application

(table continues...)

(continued)

Field	Bits	Type	Description
TOS	15:12	rw	**Type of service** The TOS bit field configuration maps a service request to an interrupt service provider: 0_H Service request is mapped to CPU0 1_H Service request is mapped to CPU1 2_H Service request is mapped to CPU2 3_H Service request is mapped to CPU3 4_H Service request is mapped to CPU4 5_H Service request is mapped to CPU5 6_H Service request is mapped to CPUCS 7_H Service request is mapped to DMA0 8_H Service request is mapped to DMA1 A_H Service request is mapped to PPU F_H SRN not used **others,** Reserved
SRE	23	rw	**Service request enable** 0_B Service request is disabled 1_B Service request is enabled
SRR	24	rh	**Service request status** The SRR bit shows the status of the service request. 0_B No service request is pending 1_B Service request is pending
CLRR	25	w	**Service request clear** The CLRR bit is required to reset SRR. 0_B No action 1_B Write with 1 leads to SRR=0; no action if SETR is written with 1 at the same time; no clear of SRR if a hardware interrupt trigger happens at the same time. read always returns 0.
SETR	26	w	**Service Request set** The SETR bit is required to set SRR. 0_B No action 1_B Write with 1 leads to SRR=1; if SRR is already =1 then IOV is set to 1; no action if CLRR is written with 1 at the same time; read always returns 0.
IOV	27	rh	**Interrupt overflow status** The IOV bit is set by HW when a new service request is triggered by interrupt trigger or write to the SETR bit with 1 while SRR=1. 0_B No interrupt overflow detected 1_B Interrupt overflow detected

(table continues...)

(continued)

Field	Bits	Type	Description
IOVCLR	28	w	**Interrupt overflow clear** IOVCLR is required to reset IOV. 0_B No action 1_B Write with 1 lead to IOV=0; no action in case of an IOV condition at the same time; read always returns 0.
0	22:16, 31:29	r	**Reserved-0** Read as 0; should be written with 0.

40.8.5 TC4Dx IR connectivity

The table below lists interfaces to and from this functional block to other blocks in the device.

Table 1939 List of IR interface signals

Interface signals	I/O	Description
SMM_IR_ApplicationReset	In	Application reset
SMM_IR_DebugReset	In	Debug reset
CLOCK_IR_fSPB	In	SPB clock input
IR_DEBUG_OTGB	Out	OTGB interface to DEBUG
IR_SMU_ALARM_NCS	Out	IR non-secure alarm
IR_SMU_ALARM_LCL	Out	IR lockstep alarm
IR_SMU_ALARM_CS	Out	IR secure alarm
IR_IR_GPSR0_SWCy	In/Out	Software interrupts from SRB0[y] and GPSRG0_SWCy triggering GPSRG0SRy
IR_IR_GPSR1_SWCy	In/Out	Software interrupts from SRB1[y] and GPSRG1_SWCy triggering GPSRG1SRy
IR_IR_GPSR2_SWCy	In/Out	Software interrupts from SRB2[y] and GPSRG2_SWCy triggering GPSRG2SRy
IR_IR_GPSR3_SWCy	In/Out	Software interrupts from SRB3[y] and GPSRG3_SWCy triggering GPSRG3SRy
IR_IR_GPSR4_SWCy	In/Out	Software interrupts from SRB4[y] and GPSRG4_SWCy triggering GPSRG4SRy
IR_IR_GPSR5_SWCy	In/Out	Software interrupts from SRB5[y] and GPSRG5_SWCy triggering GPSRG5SRy
IR_IR_GPSR6_SWCy	In/Out	Software interrupts from SRB6[y] and GPSRG6_SWCy triggering GPSRG6SRy
IR_IR_GPSR7_SWCy	In/Out	Software interrupts from SRB7[y] and GPSRG7_SWCy triggering GPSRG7SRy
FB_IR_INT_REQx	In	FB interrupt trigger signals mapped to SRNx
IR_CPU0_ICU0	Out	Interrupt control unit connected to CPU0

(table continues...)

Table 1939 (continued) List of IR interface signals

Interface signals	I/O	Description
IR_CPU1_ICU1	Out	Interrupt control unit connected to CPU1
IR_CPU2_ICU2	Out	Interrupt control unit connected to CPU2
IR_CPU3_ICU3	Out	Interrupt control unit connected to CPU3
IR_CPU4_ICU4	Out	Interrupt control unit connected to CPU4
IR_CPU5_ICU5	Out	Interrupt control unit connected to CPU5
IR_CPUCS_ICU6	Out	Interrupt control unit connected to CPUCS
IR_DMA0_ICU7	Out	Interrupt control unit connected to SDMA0
IR_DMA1_ICU8	Out	Interrupt control unit connected to SDMA1
IR_PPU_ICU10	Out	Interrupt control unit connected to PPU
FPI_IR_FPI_SIF	In	IR software interface
CSRM.CSCU_IR_CSAV[1:0]	In	CS activity visibility, trace enable for security related service requests

40.8.6 TC4Dx IR revision history

Initial release of the chapter.

41 Watchdog Timer Unit (WTU)

Watchdog timer unit (WTU) provides several independent watchdog timer (WDT) functions. The main system use of the WDT functions is program flow monitoring (PFM), for example, tracking whether the software tasks are executed in the correct sequence and do not get stuck in a specific process.

41.1 Feature list

- A dedicated WDT for each implemented CPU, a dedicated WDT for cyber-security CPU (CPUcs) and a system WDT that can be freely used by the application
- Programmable 16-bit timer
- Programmable password protection with configurable automatic password sequencing
- Programmable timestamp checking with programmable window
- Alarm is triggered if the WDT is not serviced before timer overflows
- Alarm is triggered if wrong password is provided during unlock or check access
- Configurable unlock-restriction if the respective SMU state-machine is not in RUN state
- Wake-up signal to respective CPU on timer-value transition from $7FFF_H$ to 8000_H
- WDT functions can be suspended during test and debug

41.2 Functional overview

Watchdog functions can be used by the application software to monitor the sequence and timing of software tasks and to trigger a software independent hardware reaction if errors are detected. Following figure shows the structure of the WTU.

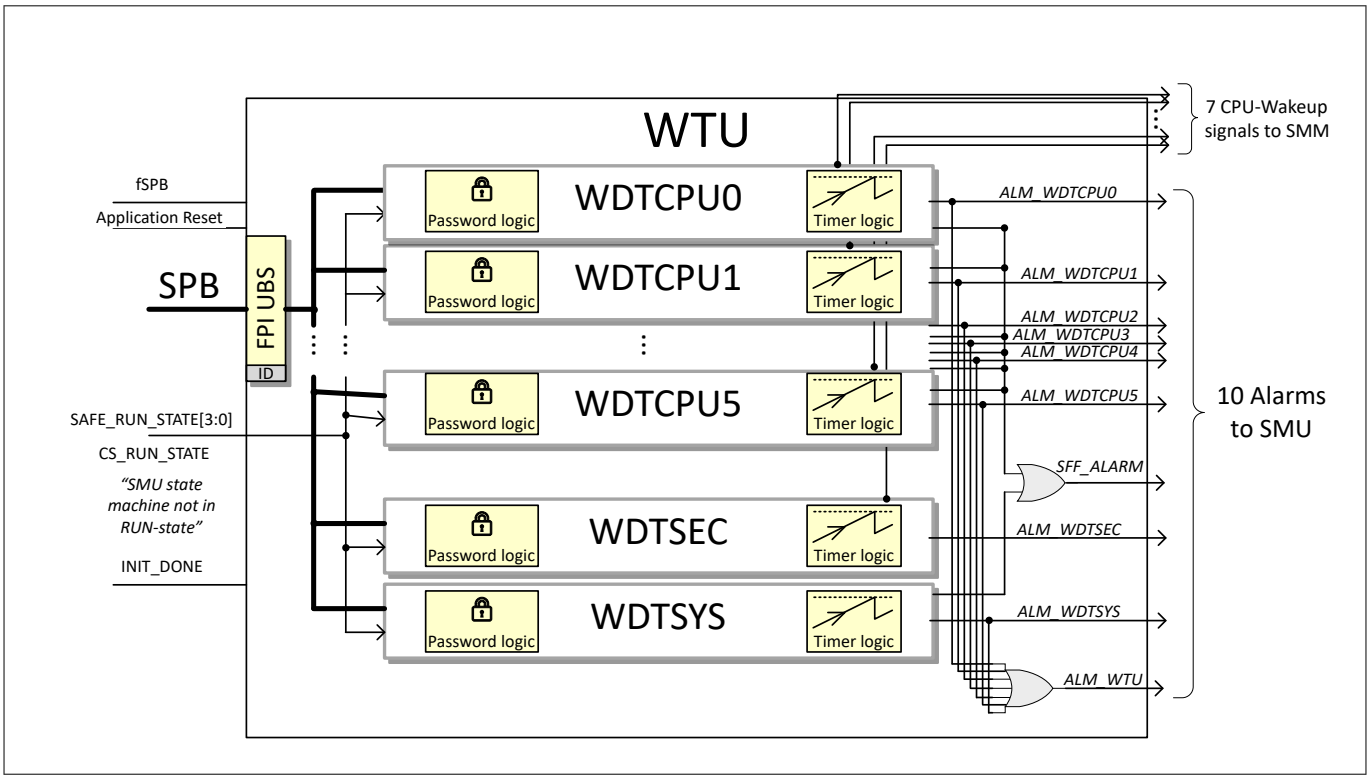


Figure 969 Internal structure of the WTU

For the alarm description please refer to the alarm mapping tables in the SMU functional block and the Safety Manual.

WTU consists of Universal Bus Slave (UBS) on the SPB FPI interface and a total of 8 independent Watchdog Timer (WDT) modules:

- One watchdog timer per lock-stepped CPU - WDTCPU0-5 (6 total)

- One watchdog timer for CSRM security CPU - WDTSEC (1 total)
- One system watchdog timer - WDTSYS (1 total)

Module ID register is a part of UBS.

In addition to the SPB bus interface, each WDT receives SMU state machine signals used for Unlock Restriction (UR) feature which is explained in the following sections. Each of WDT functions provides dedicated register interface, password and timer logic, and a separate alarm signal to the SMU. There is also a 9th alarm output which represents logical "OR" combined alarm output of WDTCPU0-5 and WDTSYS.

More details on internal mechanisms inside each of the WDT is provided in the following sections.

***Note**: Regarding links to the registers which appear in the further text in this chapter: Every WDT contains same set of registers, so wherever in text there is a link to a register, and if link text does not include WDT prefix (for example link text is just "STAT"), the link would point to the WDTSYS register description, but it represents all registers with that name in all WDTs. If the link contains WDT prefix (for example WDTSEC_STAT) then the link will point to the register description of the respective WDT, and represents only that one register.*

Related information

TC4Dx SMU alarm mapping tables on page 7233

41.3 Functional description

The main function of a WDT is program flow monitoring (PFM):

- Temporal supervision:
  - Purpose: Check that software is finishing tasks in a defined time-frame (for example, not hanging in an infinite loop)
  - How: Alarm is signaled if the application software is unable to service its WDT before a certain time-deadline or inside a certain time-window
- Logical supervision
  - Purpose: Check that software tasks are executed in the expected order
  - How: The WDT access password is configured to auto-increment after each access. Alarm is signaled if wrong password is supplied

Overflow of any of the watchdog timers leads to an SMU alarm. Each WDT also supports options to check code execution sequence and intermediate code execution time. If enabled, it can be used by the application to detect incorrect sequences or out-of-limit execution time will also result in an SMU alarm.

41.3.1 Features of a watchdog timer (WDT) function

- Programmable 16-bit watchdog timer
- Configurable input frequency: $f_{SPB}/64$, $f_{SPB}/256$ or $f_{SPB}/16384$
- Password protection mechanism with user-definable password and configurable auto-sequencing
- Temporal and logical monitoring capabilities:
  - Code sequence checking - an incorrect code sequence signature identification triggers an alarm to the SMU
  - Code execution time checking - code execution times out of expected limits triggers an alarm to the SMU
- Error indication for inspection:
  - Overflow error detection - an overflow of the WDT counter sets STAT.OE bit and triggers an alarm
  - Access error detection - an invalid password or input-timer-value access sets STAT.AE bit and triggers an alarm
- Window-watchdog feature can be emulated with the use of estimated-timer-value - CTRLA.TCVI
- Wake-up signal for corresponding CPU to SMM on timer-value transition from $7FFF_H$ to 8000_H
- Configurable Unlock Restriction (UR) mechanism to prevent WDT servicing after a WDT event that leads to an alarm to SMU

41.3.2 Different watchdog timers

The control register set of each WDT function is protected through a dedicated and partially configurable access protection, each consisting of a PROT function and an access protection unit (APU).

***Note**: For CPUs implemented with Hardware Virtualization Support the WDT can be assigned to one or multiple of its Virtual Machines (VMs) through its access protection unit register ACCEN_VM. In case where several VMs run on a single CPU, to achieve freedom from interference, respective WDTCPUy should be assigned to only one of those VMs.*

Watchdog timers for CPUs with lock-step support, WDTCPUy (y = 0-5)

The individual CPU WDT functions provide the ability to monitor separate CPU execution threads without the need for software to coordinate the shared use of a common system watchdog. Periodic servicing of a WDTCPU

and providing correct password confirms that the corresponding CPU is executing a software sequence as expected. Each CPU WDT has separate SMU alarm signaling, and reaction to each is configurable within the SMU.

Each WDTCPUy has a designated WDTCPUy_PROTSE register that controls access to the access protection unit registers (ACCEN). Write access registers (WDTCPUy_ACCEN_WRA, WDTCPUy_ACCEN_WRB) are set to:

- The respective CPUy is always enabled for write access to its related WDTCPUy function
- CERBERUS write access is configurable and enabled by default
- All other tag IDs are always disabled (non-configurable)

Read access, VM and PRS protections are fully configurable by respective WDTCPUy_PROTSE owner.

Application reset disables all WDTCPUs, but during Boot-FW WDTCPU0 gets enabled in normal mode. Other CPUs are initially in a HALT state and therefore their corresponding WDTCPUy functions are not enabled by Boot-FW.

***Note**: In case certain CPU is not available in the device (smaller devices can have less than 6 CPU cores), its respective WDTCPUx function will still be there, but it will remain disabled at all times.*

Watchdog timer for CPUcs, WDTSEC

Similar to WDTCPUy, WDTSEC is designated to monitor temporal and logical code execution of CPUcs. Write access (WDTSEC_ACCEN_WRA, WDTSEC_ACCEN_WRB):

- CPUcs is always enabled for write access to its related WDTSEC function
- CERBERUS write access is configurable, and disabled by default
- All other tag IDs are always disabled (non-configurable)

Read access is fully configurable, and enabled for only CPUcs by default. VM and PRS protections are fully configurable by respective WDTSEC_PROTCSE owner.

After an application reset, WDTSEC is disabled.

***Note**: In case CPUcs is not available in the device its respective WDTSEC function will still be there, but it will remain disabled at all times.*

System watchdog timer, WDTSYS

The WTU function block provides also a system watchdog. In comparison to the CPU and CPUcs WDT functions, the system WDT is not fixed to certain CPU and access protection registers (ACCEN) are freely configurable through WDTSYS_PROTSE and Bus-APU (WDTSYS_ACCEN_WRA, WDTSYS_ACCEN_WRB) protection mechanisms.

Application reset disables WDTSYS, but during Boot-FW it gets enabled in normal mode.

***Note**: WDTSYS is available in all derivatives of TC4xx.*

41.3.3 WDT functionality

Each WDT has an architecture as shown in the following figure.

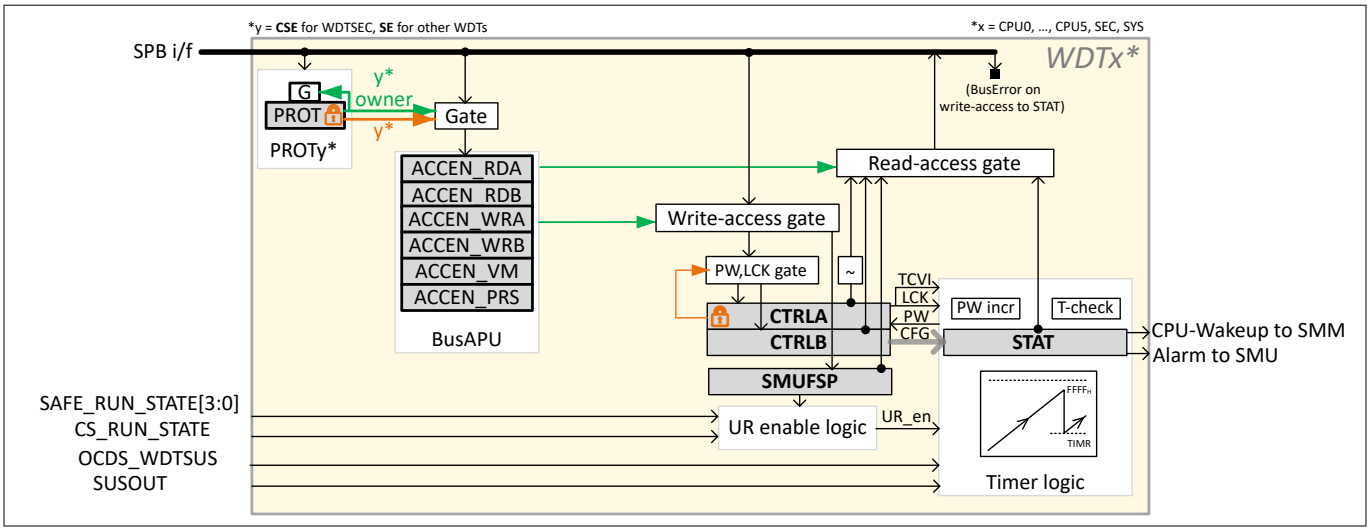


Figure 970 Single watchdog timer block diagram

Each WDT has a dedicated protection scheme. PROT controls write access to APU registers (ACCEN*), while APU registers provide read and write access control to the registers CTRLA, CTRLB, STAT and SMUFSP. Registers CTRLA and CTRLB are additionally write protected by password and lock mechanism. Symbol '~' used in the figure indicates that some of the CTRLA bits are returned bit-wise inverted on a read access in order to unintended unlock accesses.

Timer logic comprises of 16-bit up counting timer, frequency division and selection logic, logic for password auto-sequencing feature, unlock-restriction feature and timer-check feature.

Related information

Suspend mode support on page 7044

41.3.3.1 Main function registers

A WDT function provides three main control and status registers:

- STAT as a read-only status register
- CTRLA and CTRLB as control registers

CTRLA implements password protection and configurable estimated-timer-value protection. Its lock bit (LCK) represents an additional gate for write access to the CTRLA and CTRLB registers. Any write access with a wrong password or an incorrect timer-check value, if this feature is enabled, will not be executed and results in a watchdog alarm. In addition, with a correct write access that clears the LCK, the WDT function starts an internal timeout counter. If the LCK bit has not been properly set before this timer expires, the system is assumed to have malfunctioned.

CTRLB register contains shadow configuration for the watchdog function.

STAT-registers show active configuration and error-status indication.

In order to avoid accidental servicing of a WDT, a two-access sequence with password and lock mechanism is put in place. This is to ensure that only intended and expected code-blocks can service the timer. Besides password protection, timer-value checks can also be applied and could be deployed to ensure that:

- Software execution was taking the expected code-branches
- Software execution was not going faster than expected (in emulated window-watchdog-mode)

41.3.3.2 Error types and alarm signals

Each WDT function can detect two types of errors:

- Access Error (AE) - invalid access to CTRLA or CTRLB registers
- Overflow Error (OE) - timer value has overflown without being serviced in time

On detection of a WDT overflow error, an alarm is signaled to the SMU, STAT.OE bit set, the timer value is reloaded with CTRLB.TIMR timer-base value and the timer is started again.

Both AE and OE errors are combined into one alarm signal sent to the SMU. The resulting alarm reaction is configurable within the SMU.

41.3.3.3 Access types and state transitions

CTRLA register has a lock control function (LCK). When a WDT becomes unlocked it automatically starts the time-out mode (TOM). This represents a timed and password protected write-access to the vital configuration registers CTRLA and CTRLB.

As shown in the following figure, there are two states of the watchdog access mechanism - locked (L) and unlocked (U) - directly represented by the value of the LCK bit:

- Locked state provides the ability to check timer and password value without exiting the state, and also provides a gateway to the unlocked state
- Unlocked state is always followed by an automatic TOM, providing a time-limit for potential updates made during unlocked state. This means additional safety while the configuration registers CTRLA and CTRLB are open for writing

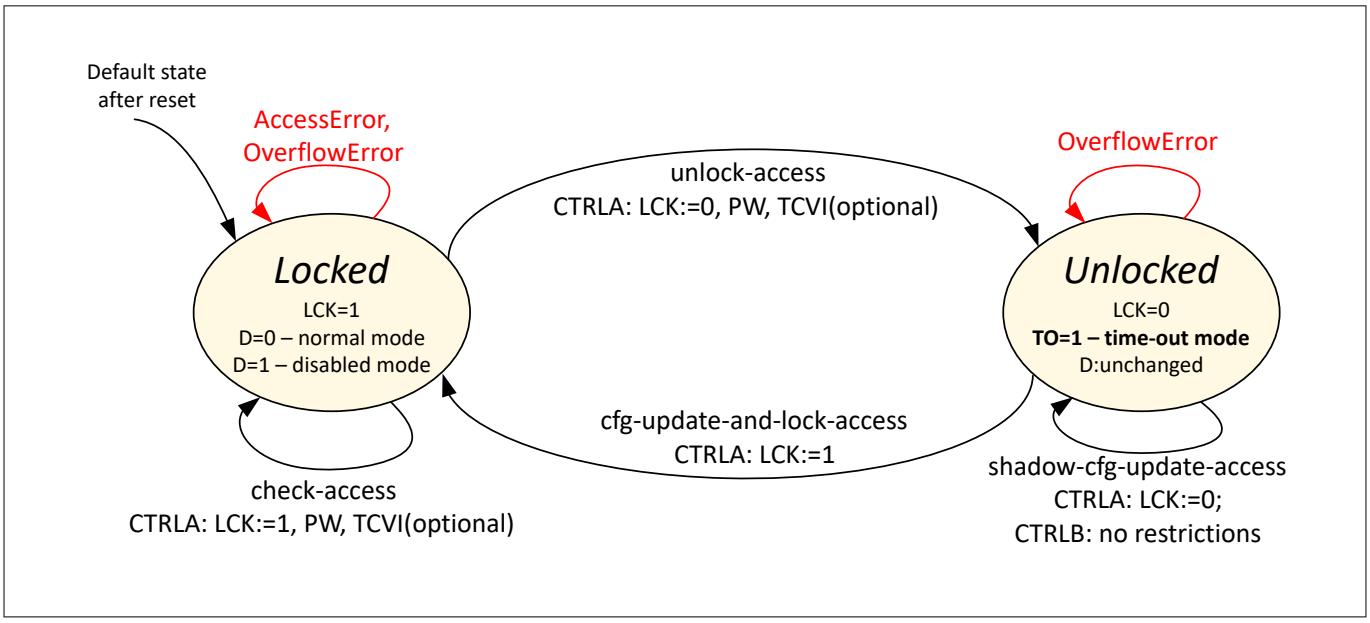


Figure 971 State diagram of locking mechanism

There are four different types of access to CTRLA and CTRLB registers:

1. Unlock access, applies to CTRLA while in locked state
2. Check access, applies to CTRLA while in locked state
3. Shadow-configuration-update access, applies to CTRLA, CTRLB while in unlocked state
4. Configuration-update-and-lock access, applies to CTRLA while in unlocked state

***Note**: Since CTRLA.LCK bit controls write access to both CTRLA and CTRLB, a write access to CTRLB with LCK bit set to 1 (meaning lock state is active) will result in an access error (AE).*

Unlock access

Used to switch the WDT from locked to unlocked. Successful if correct password and timer-value check is applied to CTRLA register, and if input LCK bit is set to 0. If unsuccessful, an access error flag is set and an alarm is sent to the SMU.

Check access

A check access is similar to the unlock access except that the LCK bit is not cleared and a subsequent shadow-config-update access is therefore not possible. A check access may only be performed when the WDT is in locked state.

If successful, the WDT stays locked and no error is reported. If unsuccessful, an access error flag is set and an alarm is sent to the SMU.

The check access can be used for intermediate checkpoints between WDT services. This may be used for task sequence or window-watchdog monitoring.

Shadow configuration update access

When a WDT is in unlocked state, access to CTRLB and access to CTRLA with LCK bit set to '0' represent shadow configuration update access. There is no limit of how many consecutive accesses of this type can be applied, but the timer is in the TOM and WDT must be switched back to locked state before the timer overflows.

During this access control bit-fields in CTRLA and CTRLB can be assigned with new values.

Configuration update and lock access

This access is applicable only in unlocked state. If CTRLA is accessed with LCK bit set to '1', PW and TCVI fields are ignored and the WDT is switched from unlocked to locked state. At the same time, shadow configuration from CTRLB register is applied to STAT register and made active.

41.3.3.4 Password protection

The 15-bit user-definable password, CTRLA.PW, can be used for logical program monitoring: to detect unexpected software loops, or to monitor the execution sequence of sub-routines. The following table summarizes the requirements for the password for unlock and check accesses. As an additional protection, **while reading password field PW of register CTRLA, the lowest 7 bits are bit-wise inverted** and cannot be identically written back.

Table 1940 Unlock and check access to WDTx_CTRLA

Bit position	Required value
[0]	Must be written with 0_B for unlock access and must be written with 1_B for check access.
[15:1]	If STAT.PA = 0: • PW[6:0] (= CTRLA[7:1]) must be written with current value stored in those bits. If PW is read prior to write, read value of PW[6:0] (CTRLA[7:1]) must be bit-wise inverted before writing • PW[14:7] (= CTRLA[15:8]) must be written with current (non-inverted) value stored in those bits If STAT.PA = 1: • PW[14:0] (= CTRLA[15:1]) must be written with expected next sequence password

(table continues...)

Table 1940 (continued) Unlock and check access to WDTx_CTRLA

Bit position	Required value
[31:16]	If STAT.TC = 0: • This bit-field is ignored If STAT.TC = 1: • Must be written with bit-wise inverted estimate of the WDT timer value. For the valid access, written value must be within +/- STAT.TCT of the actual STAT.TIM value

If an incorrect value is written to CTRLA during an unlock or check access, a watchdog access error condition exists. Bit STAT.AE is set and an alarm is sent to the SMU.

Static password check

In the static password mode (STAT.PA=0) the password can only be changed by a valid shadow config update access to CTRLA register. The unlock access has been designed such that it is not possible simply to read the register and re-write it. The lowest 7 password read bits must be bit-wise inverted (toggled) before being re-written. This prevents a simple malfunction from accidentally unlocking the WDT through a simple read/write sequence.

Automatic password sequencing check

If automatic password sequencing is enabled (STAT.PA=1) then the password changes automatically after each password check (unlock or check access). The expected next password is calculated using a pseudo-random sequence based on a 15-bit Fibonacci LFSR (linear feedback shift register) with characteristic polynomial. $x_{15} \oplus x_{14} \oplus x_{13} \oplus x_3 \oplus 1$ An initial password (or subsequent manual password updates) can also be provided by shadow-config-update access to CTRLA.

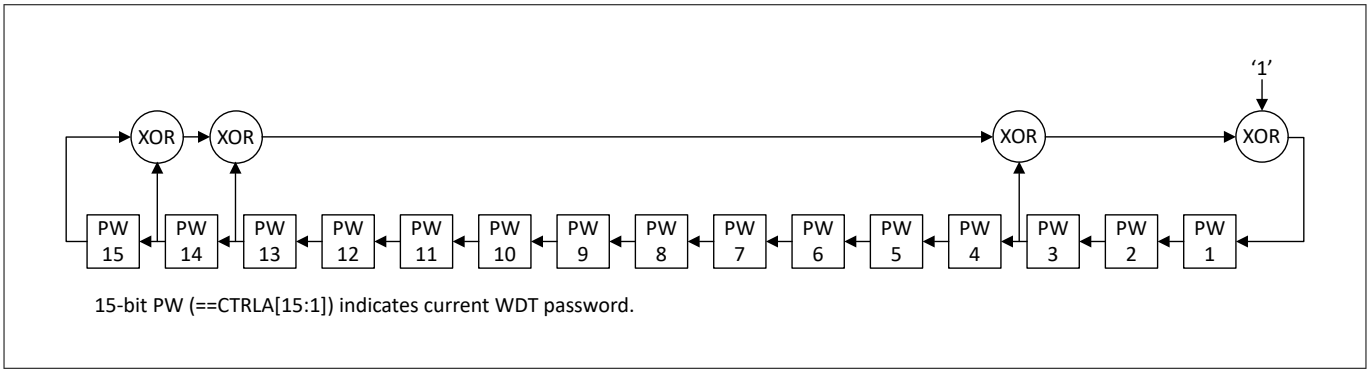


Figure 972 LFSR used for WDT password auto-sequencing

41.3.3.5 Additional timer-value protection

Estimated-timer-value check can be used for:

- Check of the elapsed program execution time since the last WDT timer service
- Emulating a window-watchdog mode

Timer-value checking is not enabled

If timer-value checking is not enabled (STAT.TC==0) then the input value of the CTRLA.TCVI field is ignored.

Timer-value checking is enabled

If timer-value checking is enabled (STAT.TC==1) then on each unlock- and check-access, the value of input field TCVI is compared with actual timer value STAT.TIM with regard to tolerance value STAT.TCT:

- If bit-wise inverted value of TCVI is inside the window: [max(TIM-TCT,0000_H), min(TIM+TCT,$FFFF_H$)], then the time-value check is correct
- If bit-wise inverted value of TCVI falls outside the window, then the access is incorrect independent from correctness of password check

Incorrect access would set STAT.AE bit and trigger an alarm to the SMU.

***Note**: The timer-value checking is done in addition to a password check.*

41.3.3.6 Timer operation

Each WDT has a 16-bit up-counter timer. Timer can be in three different modes:

- Disabled
- Enabled in normal counting mode
- Automatically enabled in time-out mode (TOM)

Disabled and normal mode are indicated through STAT.D bit-field (which is configurable via CTRLB.DR) – value '1' represents Disabled, value '0' Normal mode.

TOM is automatically started on every successful unlock-access, with the transition from locked to unlocked state. Upon entering TOM, timer base value is set to $FFFC_H$ and all other settings remain the same as in previous state (frequency IFS, PA, TC,…). TOM is ended with correct transition back to locked state.

The WDTSYS and WDTCPU0 are started in Normal mode by Boot-FW after an application reset.

Determining WDT periods

All WDTs use the SPB clock f_{SPB}. A clock divider in front of each WDT provides three WDT counter frequencies:f_{SPB} / 64, f_{SPB} / 256 or f_{SPB} / 16384. The following table shows the encoding:

Table 1941 WDT timer frequency - STAT.IFS and CTRLB.IFSR bit-fields

IFS/IFSR	Watchdog timer rate (source clock divider setting)
00_B	f_{SPB}/16384
01_B	f_{SPB}/256
10_B	f_{SPB}/64
11_B	Reserved. Do not use

The general formula to calculate a watchdog timeout period is:

$$\text{period} = \frac{\left(2^{16} - \text{start_value}\right) \times \text{divider}}{f_{SPB}} \quad (72)$$

where parameter start_value represents the fixed value $FFFC_H$ for the calculation of the time-out period, and the user-programmable reload value CTRLB.TIMR for the calculation of the normal period. Parameter divider takes values 64, 256 or 16384 indicated by STAT.IFS.

Table 1942 Example period values

Start_value	Divider	f_{SPB}	Period	1-counter-step time
$FFFC_H$	16384	100 MHz	~0.66 ms (=655.36 µs)	163.84 µs (=~0.16 ms)
0000_H	16384	100 MHz	~10737.4 ms (=~10.7 s)	163.84 µs (=~0.16 ms)

(table continues...)

Table 1942 (continued) Example period values

Start_value	Divider	f_{SPB}	Period	1-counter-step time
0000_H	256	100 MHz	~167.8 ms	2.56 µs
0000_H	64	100 MHz	~41.9 ms	0.64 µs
$FFFC_H$	16384	112.5 MHz	~0.58 ms (=582.54 µs)	145.64 µs (=~0.15 ms)
0000_H	16384	112.5 MHz	~9544.4 ms (=~9.5 s)	145.64 µs (=~0.15 ms)
0000_H	256	112.5 MHz	~149.1 ms	2.28 µs
0000_H	64	112.5 MHz	~37.3 ms	0.57 µs

Time-out mode

WDT changes automatically in time-out mode (TOM) with a valid unlock-access to register CTRLA. Additionally, for WDTCPU0 and WDTSYS TOM is entered by Boot-FW after an application reset. At the TOM entry, bit STAT.TOM is set to '1' and the timer STAT.TIM is set to $FFFC_H$ and starts counting upwards. All other bit-fields of STAT registers are the same as before entering TOM. TOM can only be exited properly by setting CTRLA.LCK = '1'. If the timer overflows before LCK is set, STAT.OE error flag is set and an alarm is sent to the SMU.

A proper exit from TOM can either be to the normal or to the disable mode, depending on the state of the disable request bit CTRLB.DR which gets applied to STAT.D.

Normal mode

With STAT.D='0' and STAT.TOM='0' the WDT operates in normal mode. Upon entry to Normal mode the timer STAT.TIM value is set to CTRLB.TIMR value, and timer begins counting up. It has to be serviced before the counter overflows. Servicing is performed through a valid access sequence to the control register CTRLA: unlock-access which moves counter into TOM, followed by config-update-and-lock-access which brings timer back to normal mode with reloaded timer value CTRLB.TIMR.

If the WDT is not serviced by the application software before a timer overflows, STAT.OE error flag is set and an alarm is signaled to the SMU. In that case, the timer is reloaded with CTRLB.TIMR value and continues counting up.

Disabled mode

With STAT.D='1' and STAT.TOM='0' the WDT is disabled. If a watchdog timer was enabled, it can be disabled again with CTRLB.DR bit set to '1' and valid config update and lock access.

In Disabled mode the timer STAT.TIM is set to CTRLB.TIMR and stopped. If valid unlock access to register CTRLA is performed in disabled mode,TOM is entered and timer is started. If valid check access is performed in disabled mode, password-auto-sequencing (if enabled) is operating normally, while the timer is still in disabled mode.

All WDTs except WDTCPU0 and system watchdog WDTSYS are in disabled mode after Boot-FW execution.

41.3.4 Operation in special cases

Operation during power-saving modes

If a CPU is in idle mode it cannot service a WDT by software. There are two ways to prevent unwanted WDT overflows and alarms in this situation. Firstly, the watchdog can be disabled before setting the CPU to idle. The disadvantage of this is that the system will no longer be monitored during the idle period.

A better approach to this problem relies upon the wake-up feature of the WDTs. Whenever CPUy is put in idle or sleep mode and the respective WDT is not disabled, the CPUy is woken up at regular intervals. When WDTx (x=CPU0-5,SEC) changes its count value (STAT.TIM) from $7FFF_H$ to 8000_H, respective CPU is woken up and continues execution at the instruction following the instruction that was executed before entering the idle or

sleep mode. The wake-up signal coming out of WDT remains high as long as its timer MSB is '1' (timer value greater or equal than 8000_H).

***Note**: Before switching into a non-running power-management mode, software should perform a watchdog service sequence. At the shadow-config-update access, the watchdog refresh request value, CTRLB.TIMR, should be programmed such that the wake-up occurs after a period which best meets application requirements. The maximum period between two CPU wake-ups is one-half of the maximum WDT period.*

Behavior in case of SMU FAULT

It is possible to configure a WDT so that it can no longer be refreshed or modified if the corresponding SMU state machine is not in RUN state - so it is either in START or FAULT state. This is to prevent software from interfering with potential ongoing SMU reaction to an alarm. This feature may be enabled via bit-field STAT.UR.

SMU supports total of 5 separate signaling lines - 1 for security partition and up to 4 for other non-secure resource partitions. Each non-secure WDT can be part of any of the 4 non-secure partitions, and this is configurable for each watchdog independently through SMUFSP register.

If STAT.UR=='1' and respective SMU state machine (as selected by SMUFSP bits) is not in RUN state then an unlock-access to CTRLA will NOT lead to the unlock state of the WDT, as indicated by CTRLA.LCK=='1'.

As long as bit STAT.UR is '0', the Unlock-Restriction feature is ignored, and valid unlock-access would result in an unlocked watchdog timer.

41.4 Registers

41.4.1 Register overview - access mode glossary

Table 1943 Register overview - access mode glossary

Keyword	Description
CSE	Access protection using PROT register WDTSEC_PROTCSE.
SEy (y=0-5)	Access protection using PROT register WDTCPUy_PROTSE.
SSE	Access protection using PROT register WDTSYS_PROTSE.
APU-PCS	Protection group consisting of registers WDTSEC_ACCEN_WRA, WDTSEC_ACCEN_WRB, WDTSEC_ACCEN_RDA, WDTSEC_ACCEN_RDB, WDTSEC_ACCEN_VM, WDTSEC_ACCEN_PRS.
PCS	Access protection using *APU-PCS* registers.
APU-PSYS	Protection group consisting of registers WDTSYS_ACCEN_WRA, WDTSYS_ACCEN_WRB, WDTSYS_ACCEN_RDA, WDTSYS_ACCEN_RDB, WDTSYS_ACCEN_VM, WDTSYS_ACCEN_PRS.
PSYS	Access protection using *APU-PSYS* registers.
APU-Py (y=0)	Protection group consisting of registers WDTCPUy_ACCEN_WRA, WDTCPUy_ACCEN_WRB, WDTCPUy_ACCEN_RDA, WDTCPUy_ACCEN_RDB, WDTCPUy_ACCEN_VM, WDTCPUy_ACCEN_PRS.
Py	Access protection using *APU-Py* registers.
APU-Py (y=1)	Protection group consisting of registers WDTCPUy_ACCEN_WRA, WDTCPUy_ACCEN_WRB, WDTCPUy_ACCEN_RDA, WDTCPUy_ACCEN_RDB, WDTCPUy_ACCEN_VM, WDTCPUy_ACCEN_PRS.
APU-Py (y=2)	Protection group consisting of registers WDTCPUy_ACCEN_WRA, WDTCPUy_ACCEN_WRB, WDTCPUy_ACCEN_RDA, WDTCPUy_ACCEN_RDB, WDTCPUy_ACCEN_VM, WDTCPUy_ACCEN_PRS.
APU-Py (y=3)	Protection group consisting of registers WDTCPUy_ACCEN_WRA, WDTCPUy_ACCEN_WRB, WDTCPUy_ACCEN_RDA, WDTCPUy_ACCEN_RDB, WDTCPUy_ACCEN_VM, WDTCPUy_ACCEN_PRS.
APU-Py (y=4)	Protection group consisting of registers WDTCPUy_ACCEN_WRA, WDTCPUy_ACCEN_WRB, WDTCPUy_ACCEN_RDA, WDTCPUy_ACCEN_RDB, WDTCPUy_ACCEN_VM, WDTCPUy_ACCEN_PRS.
APU-Py (y=5)	Protection group consisting of registers WDTCPUy_ACCEN_WRA, WDTCPUy_ACCEN_WRB, WDTCPUy_ACCEN_RDA, WDTCPUy_ACCEN_RDB, WDTCPUy_ACCEN_VM, WDTCPUy_ACCEN_PRS.
BE	Always returns a Bus Error.
U	No access restrictions.
PROT	Access restrictions as defined in the PROT register access rules.
32	Access only when using 32-bit width.

41.4.2 Registers overview - WTU (ascending offset address)

Table 1944 Registers overview - WTU (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ID	Module identification register	008_H	PSYS	BE	PowerOn Reset	7003
WDTCPUy_PROTSE	CPUy WDT PROT register safe endinit - SEy	018_H+y* 30_H	U	SV, PROT	Application Reset	7004
WDTCPUy_ACCEN_WRA	CPUy WDT write access enable register A	$01C_H$+y* 30_H	U	SEy, SV	Application Reset	7006
WDTCPUy_ACCEN_WRB	CPUy WDT write access enable register B	020_H+y* 30_H	U	SEy, SV	Application Reset	7011
WDTCPUy_ACCEN_RDA	CPUy WDT read access enable register A	024_H+y* 30_H	U	SEy, SV	Application Reset	7012
WDTCPUy_ACCEN_RDB	CPUy WDT read access enable register B	028_H+y* 30_H	U	SEy, SV	Application Reset	7012
WDTCPUy_ACCEN_VM	CPUy WDT VM access enable register	$02C_H$+y* 30_H	U	SEy, SV	Application Reset	7013
WDTCPUy_ACCEN_PRS	CPUy WDT PRS access enable register	030_H+y* 30_H	U	SEy, SV	Application Reset	7014
WDTCPUy_SMUFSP	CPUy WDT SMU partitions register	038_H+y* 30_H	Py	Py, SV	Application Reset	7014
WDTCPUy_CTRLA	CPUy WDT control register A	$03C_H$+y* 30_H	Py	Py, 32	See 7015	7015
WDTCPUy_CTRLB	CPUy WDT control register B	040_H+y* 30_H	Py	Py	See 7017	7017
WDTCPUy_STAT	CPUy WDT status register	044_H+y* 30_H	Py	BE	See 7018	7018
WDTSEC_PROTCSE	Security WDT PROT register cyber-secure endinit - CSE	150_H	U	SV, PROT	Application Reset	7020
WDTSEC_ACCEN_WRA	Security WDT write access enable register A	154_H	U	CSE, SV	Application Reset	7022
WDTSEC_ACCEN_WRB	Security WDT write access enable register B	158_H	U	CSE, SV	Application Reset	7023
WDTSEC_ACCEN_RDA	Security WDT read access enable register A	$15C_H$	U	CSE, SV	Application Reset	7024
WDTSEC_ACCEN_RDB	Security WDT read access enable register B	160_H	U	CSE, SV	Application Reset	7024
WDTSEC_ACCEN_VM	Security WDT VM access enable register	164_H	U	CSE, SV	Application Reset	7025
WDTSEC_ACCEN_PRS	Security WDT PRS access enable register	168_H	U	CSE, SV	Application Reset	7025
WDTSEC_SMUFSP	Security WDT SMU partitions register	170_H	PCS	PCS, SV	Application Reset	7026

(table continues...)

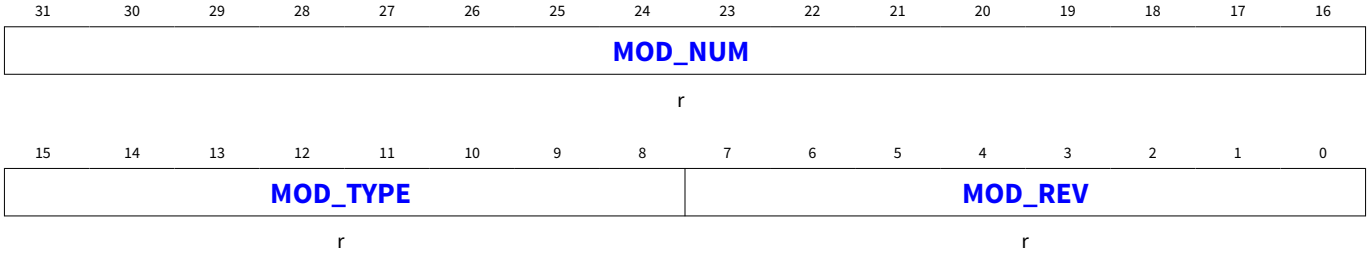
Table 1944 (continued) Registers overview - WTU (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
WDTSEC_CTRLA	Security WDT control register A	174_H	PCS	PCS, 32	See 7027	7027
WDTSEC_CTRLB	Security WDT control register B	178_H	PCS	PCS	Application Reset	7028
WDTSEC_STAT	Security WDT status register	$17C_H$	PCS	BE	Application Reset	7030
WDTSYS_PROTSE	System WDT PROT register safe endinit - SSE	184_H	U	SV, PROT	Application Reset	7032
WDTSYS_ACCEN_WRA	System WDT write access enable register A	188_H	U	SSE, SV	Application Reset	7034
WDTSYS_ACCEN_WRB	System WDT write access enable register B	$18C_H$	U	SSE, SV	Application Reset	7034
WDTSYS_ACCEN_RDA	System WDT read access enable register A	190_H	U	SSE, SV	Application Reset	7035
WDTSYS_ACCEN_RDB	System WDT read access enable register B	194_H	U	SSE, SV	Application Reset	7035
WDTSYS_ACCEN_VM	System WDT VM access enable register	198_H	U	SSE, SV	Application Reset	7036
WDTSYS_ACCEN_PRS	System WDT PRS access enable register	$19C_H$	U	SSE, SV	Application Reset	7036
WDTSYS_SMUFSP	System WDT SMU partitions register	$1A4_H$	PSYS	PSYS, SV	Application Reset	7037
WDTSYS_CTRLA	System WDT control register A	$1A8_H$	PSYS	PSYS, 32	See 7038	7038
WDTSYS_CTRLB	System WDT control register B	$1AC_H$	PSYS	PSYS	See 7040	7040
WDTSYS_STAT	System WDT status register	$1B0_H$	PSYS	BE	See 7041	7041

41.4.3 Module identification register

ID Offset address: 008_H

Module identification register PowerOn Reset value: $00ED\ C001_H$



Field	Bits	Type	Description
MOD_REV	7:0	r	**Module revision number** This bit-field defines the module revision number. The value of a module revision starts with 01_H (first revision).
MOD_TYPE	15:8	r	**Module type** The bit-field is set to $C0_H$ which defines the module as a 32-bit module.
MOD_NUM	31:16	r	**Module number value** This bit-field defines a module identification number. The value for the watchdog timer module is $00ED_H$.

41.4.4 CPUy WDT PROT register safe endinit - SEy

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner and in some PROT states, the secure master to update the PROT state as described in the PROT mechanism.

The PROTSE register controls lock / unlock of the local Safe Endinit protected control registers.

WDTCPUy_PROTSE (y=0-5) Offset address: 018_H+y*30_H

CPUy WDT PROT register safe endinit - SEy Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OWEN	ODEF	TAGID						PRSEN	PRS			VMEN	VM		
w	rw	rw						rw	rw			rw	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												SWEN	STATE		
r												w	rwh		

Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write

(table continues...)

(continued)

Field	Bits	Type	Description
VM	18:16	rw	**Virtual Machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual Machine definition Enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection Set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.
PRSEN	23	rw	**Protection Set definition Enable for PROT owner** 0_B **Disable**: PRS is not part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; should be written with 0.

Table 1945 Access mode restrictions of WDTCPUy_PROTSE (y=0-5) sorted by descending priority

Mode name	Access mode		Description
write 1 to .**SWEN**	rwh	STATE	
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

41.4.5 CPUy WDT write access enable register A (y=0)

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

D and DS TAG-IDs of corresponding CPU are enabled after reset and can be configured, TAG-ID of Cerberus is configurable with default value EN='1'.

All others are always disabled (non-configurable).

WDTCPUy_ACCEN_WRA (y=0) Offset address: $01C_H+y*30_H$

CPUy WDT write access enable register A Application Reset value: $1000\ 0003_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
r	r	r	rw	r	r	r	r	r	r	r	r	r	r	r	r

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
r	r	r	r	r	r	r	r	r	r	r	r	r	r	rw	rw

Field	Bits	Type	Description
ENq (q=00-01)	q	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access
ENq (q=02-27)	q	r	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access
ENq (q=28)	q	rw	**Write access enable for TAG-ID of CERBERUS - EN28** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access
ENq (q=29-31)	q	r	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access

41.4.6 CPUy WDT write access enable register A (y=1)

WDTCPUy_ACCEN_WRA (y=1) Offset address: $01C_H+y*30_H$

CPUy WDT write access enable register A Application Reset value: $1000\ 000C_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
r	r	r	rw	r	r	r	r	r	r	r	r	r	r	r	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
r	r	r	r	r	r	r	r	r	r	r	r	rw	rw	r	r

Field	Bits	Type	Description
ENq (q=00-01)	q	r	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access
ENq (q=02-03)	q	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access
ENq (q=04-27)	q	r	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access
ENq (q=28)	q	rw	**Write access enable for TAG-ID of CERBERUS - EN28** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access
ENq (q=29-31)	q	r	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access

41.4.7 CPUy WDT write access enable register A (y=2)

WDTCPUy_ACCEN_WRA (y=2) Offset address: $01C_H+y*30_H$

CPUy WDT write access enable register A Application Reset value: $1000\ 0030_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
r	r	r	rw	r	r	r	r	r	r	r	r	r	r	r	r

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
r	r	r	r	r	r	r	r	r	r	rw	rw	r	r	r	r

Field	Bits	Type	Description
ENq (q=00-03)	q	r	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access
ENq (q=04-05)	q	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access
ENq (q=06-27)	q	r	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access
ENq (q=28)	q	rw	**Write access enable for TAG-ID of CERBERUS - EN28** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access
ENq (q=29-31)	q	r	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access

41.4.8 CPUy WDT write access enable register A (y=3)

WDTCPUy_ACCEN_WRA (y=3) Offset address: $01C_H+y*30_H$

CPUy WDT write access enable register A Application Reset value: $1000\ 00C0_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
r	r	r	rw	r	r	r	r	r	r	r	r	r	r	r	r

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
r	r	r	r	r	r	r	r	rw	rw	r	r	r	r	r	r

Field	Bits	Type	Description
ENq (q=00-05)	q	r	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access
ENq (q=06-07)	q	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access
ENq (q=08-27)	q	r	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access
ENq (q=28)	q	rw	**Write access enable for TAG-ID of CERBERUS - EN28** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access
ENq (q=29-31)	q	r	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access

41.4.9 CPUy WDT write access enable register A (y=4)

WDTCPUy_ACCEN_WRA (y=4) Offset address: $01C_H+y*30_H$

CPUy WDT write access enable register A Application Reset value: $1000\ 0300_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
r	r	r	rw	r	r	r	r	r	r	r	r	r	r	r	r

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
r	r	r	r	r	r	rw	rw	r	r	r	r	r	r	r	r

Field	Bits	Type	Description
ENq (q=00-07)	q	r	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access
ENq (q=08-09)	q	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access
ENq (q=10-27)	q	r	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access
ENq (q=28)	q	rw	**Write access enable for TAG-ID of CERBERUS - EN28** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access
ENq (q=29-31)	q	r	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access

41.4.10 CPUy WDT write access enable register A (y=5)

WDTCPUy_ACCEN_WRA (y=5) Offset address: $01C_H+y*30_H$

CPUy WDT write access enable register A Application Reset value: $1000\ 0C00_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
r	r	r	rw	r	r	r	r	r	r	r	r	r	r	r	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
r	r	r	r	rw	rw	r	r	r	r	r	r	r	r	r	r

Field	Bits	Type	Description
ENq (q=00-09)	q	r	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access
ENq (q=10-11)	q	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access
ENq (q=12-27)	q	r	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access
ENq (q=28)	q	rw	**Write access enable for TAG-ID of CERBERUS - EN28** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access
ENq (q=29-31)	q	r	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access

41.4.11 CPUy WDT write access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

Writing rights are 'hard-coded' to just accept D and DS TAG-IDs of corresponding CPU, and TAG-ID of Cerberus.

WDTCPUy_ACCEN_WRB (y=0-5) Offset address: 020_H+y*30_H

CPUy WDT write access enable register B Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															
r															

Field	Bits	Type	Description
0	31:0	r	**Reserved** Read as 0; should be written with 0.

41.4.12 CPUy WDT read access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

WDTCPUy_ACCEN_RDA (y=0-5) Offset address: 024_H+y*30_H

CPUy WDT read access enable register A Application Reset value: FFFF FFFF$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access

41.4.13 CPUy WDT read access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

WDTCPUy_ACCEN_RDB (y=0-5) Offset address: 028_H+y*30_H

CPUy WDT read access enable register B Application Reset value: 0000 00FF$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access
0	31:8	r	**Reserved** Read as 0; should be written with 0.

41.4.14 CPUy WDT VM access enable register

This register defines which virtual machine encodings are enabled or disabled for accesses to the access protected region. There is one RDq and WRq bit for each VMq.

WDTCPUy_ACCEN_VM (y=0-5) Offset address: $02C_H+y*30_H$

CPUy WDT VM access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for VM ID q** This bit enables read access to the access protected region for transactions with the VM ID q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for VM ID q** This bit enables write access to the access protected region for transactions with the VM ID q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

41.4.15 CPUy WDT PRS access enable register

This register defines which protection register set encodings are enabled or disabled for accesses to the access protected region. There is one RDq and WRq bit for each PRSq.

WDTCPUy_ACCEN_PRS (y=0-5) Offset address: 030_H+y*30_H

CPUy WDT PRS access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for PRS q** This bit enables read access to the access protected region for transactions with the PRS q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for PRS q** This bit enables write access to the access protected region for transactions with the PRS q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

41.4.16 CPUy WDT SMU partitions register

This register is to specify output of which SMU state machines will be considered for the SMU unlock-restriction (UR) feature. If more than one bit-field is set, their values are combined with OR logic.

WDTCPUy_SMUFSP (y=0-5) Offset address: 038_H+y*30_H

CPUy WDT SMU partitions register Application Reset value: $0000\ 0001_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0											CSPART	PART3	PART2	PART1	PART0
r											rw	rw	rw	rw	rw

Field	Bits	Type	Description
PARTx (x=0-3)	x	rw	**Consider signals from SMU state machine partition x** This bit enables that the outputs from SMU safe state machine partition x will be considered in unlock restriction feature (see also bit WDTx_STAT.UR). 0_B Outputs of corresponding SMU state machine are ignored inside this watchdog unit. 1_B Outputs of corresponding SMU state machine are considered inside this watchdog unit.
CSPART	4	rw	**Consider signals from SMU CS-related partition** This bit enables that the outputs from SMU CS-related partition will be considered in unlock restriction feature (see also bit WDTx_STAT.UR). 0_B Outputs of corresponding SMU state machine are ignored inside this watchdog unit. 1_B Outputs of corresponding SMU state machine are considered inside this watchdog unit.
0	31:5	r	**Reserved** Read as 0; should be written with 0.

41.4.17 CPUy WDT control register A

This register is written with the expected password, the inverted estimated timer value and a lock-bit value.

The meaning of the individual values depends on the access type and active configuration fields STAT.PA and STAT.TC.

This register stores the current watchdog password value (PW) and lock control bit (LCK).

WDTCPUy_CTRLA (y=0-5) Offset address: $03C_H+y*30_H$

CPUy WDT control register A Reset values see: Table 1946

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TCVI															
w															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PW															LCK
rwh															rwh

Field	Bits	Type	Description
LCK	0	rwh	**Lock bit** LCK bit is cleared after a valid unlock-access to WDTx_CTRLA when unlock is not disabled. (Note: unlock can be disabled by UR-feature or by debug) LCK bit is set after a valid configuration-update-and-lock-access to WDTx_CTRLA. A valid check-access or shadow-config-update-access does not trigger a change of the LCK value. 0_B Registers WDTx_CTRLA and WDTx_CTRLB are unlocked 1_B Registers WDTx_CTRLA and WDTx_CTRLB are locked (default after reset)
PW	15:1	rwh	**Password** This bit-field is written with a password value during an unlock- or check-access to WDTx_CTRLA: • If corresponding WDTx_STAT.PA == 0 then this bit-field must be written with its current contents. • If corresponding WDTx_STAT.PA == 1 then this bit-field must be written with the next password in the LFSR sequence. It is written with a new password value during a shadow-cfg-update-access to WDTx_CTRLA, and it is ignored during cfg-update-and-lock-access to WDTx_CTRLA. A read from this bit-field returns current password, but bits CTRLA[7:1] are bitwise inverted to ensure that a simple read-write is not sufficient to service the WDTx. The default password after reset is $007C_H$=000_0000_0111_1100$_B$ (due to inversion, SW will read it as 0003_H=000_0000_0000_0011$_B$).
TCVI	31:16	w	**Timer check value inverted** The timer-check value inverted (TCVI) is used only during unlock- and check-access and only if configured for use (WDTx_STAT.TC==1). In all other cases provided value is ignored. The value written to these bits will be bitwise-inverted and compared with the current timer value, with respect to the tolerance margin WDTx_STAT.TCT. If written value is outside defined margins it will result in an access-error. A read from this bit-field always returns 0000_H.

Table 1946 Reset values of WDTCPUy_CTRLA (y=0-5)

Reset type	Reset value	Note
Application Reset	0000 00F9$_H$	Default password value and locked.
After reset read value	0000 0007$_H$	Bits [7:1] (PW[6:0]) are read as bit-wise inverted

41.4.18 CPUy WDT control register B

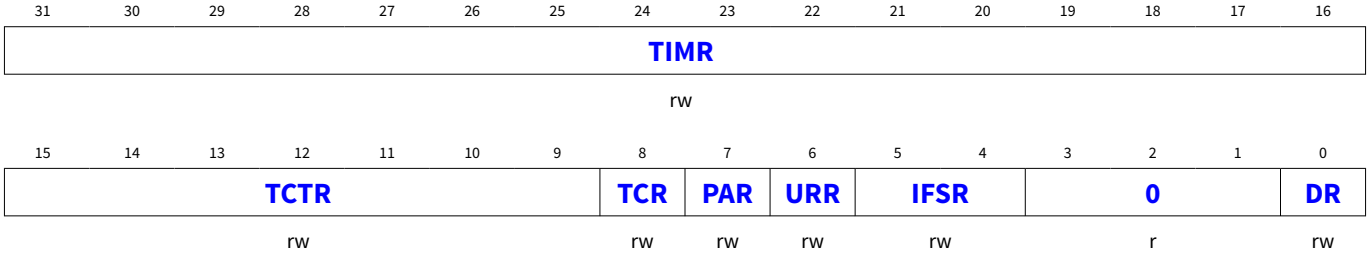
This register represents a shadow configuration for the watchdog timer.

WDTx_CTRLB bits can only be modified if corresponding WDTx_CTRLA.LCK is cleared - after a valid unlock-access.

Active configuration bits in WDTx_STAT are updated by these shadow/request bits only when LCK is set again - after a valid config-update-and-lock-access.

WDTCPUy_CTRLB (y=0-5) Offset address: 040_H+y*30_H

CPUy WDT control register B Reset values see: Table 1947



Field	Bits	Type	Description
DR	0	rw	**Disable request bit** This bit represents a request (shadow) to disable the watchdog Timer. 0_B Request to enable the WDTx 1_B Request to disable the WDTx
IFSR	5:4	rw	**Input frequency selection request bits** Two bits of IFSR represent a request (shadow) to configure the watchdog timer frequency. 00_B Request to set input frequency to fSPB/16384. 01_B Request to set input frequency to fSPB/256. 10_B Request to set input frequency to fSPB/64. 11_B Reserved. Should not be used.
URR	6	rw	**Unlock restriction request bit** This bit represents a request (shadow) to enable unlock-restriction feature. If the unlock-restriction is set (WDTx_STAT.UR=='1') then WDTx cannot be unlocked in case respective SMU state machine is not in RUN state. 0_B Request to disable unlock-restriction feature 1_B Request to enable unlock-restriction feature
PAR	7	rw	**Password auto-sequence request bit** This bit represents a request (shadow) to configure a password-autosequencing feature. If the active password-autosequencing is set (WDTx_STAT.PA==1), after each unlock- and check-access, the password WDTx_CTRLA.PW will increment using built-in LFSR and store the new value. 0_B Request to disable automatic change of password after each valid unlock- or check-access 1_B Request to enable automatic sequence of password after each valid unlock- or check-access

(table continues...)

(continued)

Field	Bits	Type	Description
TCR	8	rw	**Timer check request bit** This bit represents a request (shadow) to configure a timer-check feature. If the active timer-check configuration is set (WDTx_STAT.TC==1), on each unlock- and check-access, the input TCVI will be compared to the current timer value WDTx_STAT.TIM with respect to the tolerance margin WDTx_STAT.TCT. 0_B Request to disable timer check feature during unlock- or check-access. 1_B Request to enable timer check feature during unlock- or check-access.
TCTR	15:9	rw	**Timer check tolerance request** This bit-field represents a request (shadow) to configure the timer-check-tolerance value.
TIMR	31:16	rw	**Timer base request value** This bit-field represents a request (shadow) to set the timer reload value. A read from this bit-field returns the timer reload value.
0	3:1	r	**Reserved** Read as 0; should be written with 0.

Table 1947 Reset values of WDTCPUy_CTRLB (y=0-5)

Reset type	Reset value	Note
Application Reset	FFFC 0001_H	Request to enable with timer set to 0xFFFC.
After Boot-FW Value	FFFC 0000_H	Only WDTCPU0 is enabled during Boot-FW execution.

41.4.19 CPUy WDT status register

This register represents the active configuration of the watchdog timer.

These bits are read-only from the outside and they get updated by shadow-configuration bits from WDTx_CTRLB after a valid config-update-and-lock-access.

All WDTx are disabled by reset, and only WDTCPU0 and WDTSYS are enabled by FW before application SW starts.

WDTCPUy_STAT (y=0-5) Offset address: 044_H+y*30_H

CPUy WDT status register Reset values see: Table 1948

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TIM															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TCT							TC	PA	UR	IFS		AE	OE	TOM	D
rh							rh	rh	rh	rh		rh	rh	rh	rh

Field	Bits	Type	Description
D	0	rh	**Watchdog enable/disable status Flag** This bit represents watchdog timer's enable/disable state. This bit is cleared when time-out mode (unlock-state) is entered - after a valid unlock-access to corresponding CTRLA register. 0_B WDTx is enabled 1_B WDTx is disabled.
TOM	1	rh	**Watchdog time-out mode flag** This bit is set when time-out mode (unlock-state) is entered. It is automatically cleared when time-out mode (unlock-state) is left. 0_B The watchdog is not operating in time-out mode 1_B The watchdog is operating in time-out mode.
OE	2	rh	**Watchdog overflow error status flag** This bit is set when the WDTx timer value overflows from $FFFF_H$ to 0000_H. This bit is only cleared when WDTx_CTRLA.LCK changes state from 0 to 1 (after a valid config-update-and-lock-access). 0_B No watchdog overflow error. 1_B A watchdog overflow error has occurred.
AE	3	rh	**Watchdog access error status flag** This bit is set when: • An invalid unlock- or check-access to WDTx_CTRLA register was attempted. • A write access to WDTx_CTRLB register was attempted while WDTx_CTRLA.LCK is set. This bit is only cleared when WDTx_CTRLA.LCK changes state from 0 to 1 (after a valid config-update-and-lock-access). 0_B No watchdog access error. 1_B A watchdog access error has occurred.
IFS	5:4	rh	**Watchdog internal frequency selection status flag** These bits indicate the current WDTx clock rate. 00_B WDTx counter frequency is fSPB/16384 (default). 01_B WDTx counter frequency is fSPB/256. 10_B WDTx counter frequency is fSPB/64. 11_B Reserved. Should not be used.
UR	6	rh	**SMU unlock restriction status flag** This bit represents status of unlock-restriction feature for the Watchdog Timer. 0_B Unlocking of WDTx is not from SMU FSP side. 1_B Unlocking of WDTx not possible when corresponding SMU FSP state machine is not in RUN state

(table continues...)

(continued)

Field	Bits	Type	Description
PA	7	rh	**Password auto-sequence status flag** This bit represents control for password-autosequencing. 0_B No change of password after each unlock- or check-access. 1_B Automatically sequence the password after each valid unlock- or check-access using built-in LFSR.
TC	8	rh	**Timer check status flag** This bit represents control for timer-check - whether an additional check (next to password check) will be performed during unlock- or check-access. 0_B Additional timer-check during unlock- or check-access is disabled. 1_B Additional timer-check during unlock- or check-access is enabled.
TCT	15:9	rh	**Timer check tolerance** This bit-field represents the tolerance value of the timer check during unlock- or check-access. If timer-check is enabled, the timer value is combined with this tolerance value to create an upper and lower comparison boundary [max(TIM-TCT,0), min(TIM+TCT,$FFFF_H$)]. If time-check is enabled and provided value falls outside these boundaries an access error will be signaled to the SMU.
TIM	31:16	rh	**Timer value** This bit-field reflects the current value of the watchdog timer counter. Performing sequence of two accesses: a valid unlock-access followed by a valid config-update-and-lock-access, is servicing of the WDTx and the WDTx_STAT.TIM value gets reloaded by the value stored in WDTx_CTRLB.TIMR. The value reload into WDTx_STAT.TIM is not done in suspend mode.

Table 1948 Reset values of WDTCPUy_STAT (y=0-5)

Reset type	Reset value	Note
Application Reset	FFFC 0001$_H$	WDTCPUy (y=0-5) are disabled with default timer value of 0xFFFC.
After Boot-FW Value	XXXX 0000$_H$	Only WDTCPU0 is enabled during Boot-FW execution. Exact timer value depends at which moment timer is read.

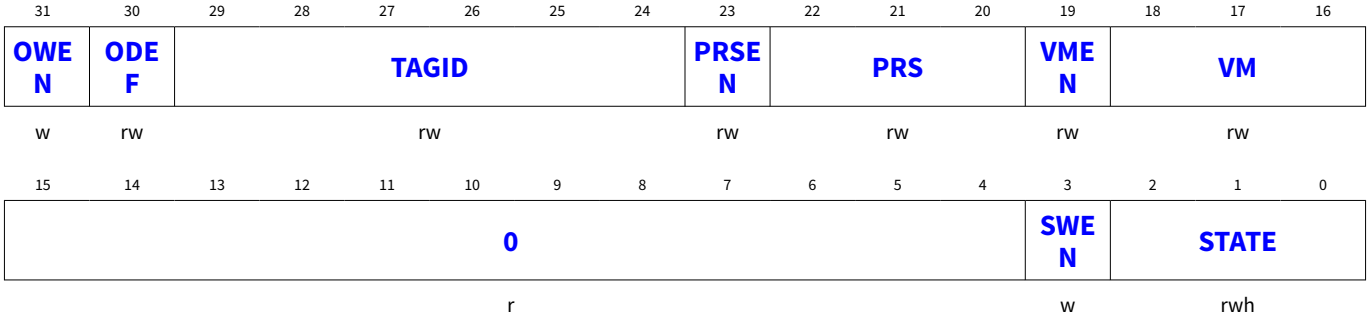
41.4.20 Security WDT PROT register cyber-secure endinit - CSE

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner and in some PROT states, the secure master to update the PROT state as described in the PROT mechanism.

The PROTCSE register controls lock / unlock of the local Cyber-Security Endinit protected registers.

WDTSEC_PROTCSE Offset address: 150_H

Security WDT PROT register cyber-secure endinit - CSE Application Reset value: 4C00 0000$_H$



Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
VM	18:16	rw	**Virtual Machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual Machine definition Enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection Set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.
PRSEN	23	rw	**Protection Set definition Enable for PROT owner** 0_B **Disable**: PRS is not part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner

(table continues...)

(continued)

Field	Bits	Type	Description
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; should be written with 0.

Table 1949 Access mode restrictions of WDTSEC_PROTCSE sorted by descending priority

Mode name	Access mode		Description
write 1 to .**SWEN**	rwh	STATE	
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

41.4.21 Security WDT write access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

D and DS TAG-IDs of CPUcs are enabled after reset and can be configured, TAG-ID of Cerberus is configurable with default value EN='0'.

All others are always disabled (non-configurable).

WDTSEC_ACCEN_WRA Offset address: 154_H

Security WDT write access enable register A Application Reset value: $0000\ 3000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
r	r	r	rw	r	r	r	r	r	r	r	r	r	r	r	r

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
r	r	rw	rw	r	r	r	r	r	r	r	r	r	r	r	r

Field	Bits	Type	Description
ENq (q=00-11)	q	r	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access

(table continues...)

(continued)

Field	Bits	Type	Description
ENq (q=12-13)	q	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access
ENq (q=14-27)	q	r	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access
ENq (q=28)	q	rw	**Write access enable for TAG-ID of CERBERUS - EN28** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access
ENq (q=29-31)	q	r	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access

41.4.22 Security WDT write access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

Writing rights are 'hard-coded' to just accept D and DS TAG-IDs of corresponding CPU, and TAG-ID of Cerberus.

WDTSEC_ACCEN_WRB Offset address: 158_H

Security WDT write access enable register B Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															
r															

Field	Bits	Type	Description
0	31:0	r	**Reserved** Read as 0; should be written with 0.

41.4.23 Security WDT read access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

WDTSEC_ACCEN_RDA Offset address: $15C_H$

Security WDT read access enable register A Application Reset value: $0000\ 3000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access

41.4.24 Security WDT read access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

WDTSEC_ACCEN_RDB Offset address: 160_H

Security WDT read access enable register B Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access
0	31:8	r	**Reserved** Read as 0; should be written with 0.

41.4.25 Security WDT VM access enable register

This register defines which virtual machine encodings are enabled or disabled for accesses to the access protected region. There is one RDq and WRq bit for each VMq.

WDTSEC_ACCEN_VM Offset address: 164_H

Security WDT VM access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for VM ID q** This bit enables read access to the access protected region for transactions with the VM ID q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for VM ID q** This bit enables write access to the access protected region for transactions with the VM ID q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

41.4.26 Security WDT PRS access enable register

This register defines which protection register set encodings are enabled or disabled for accesses to the access protected region. There is one RDq and WRq bit for each PRSq.

WDTSEC_ACCEN_PRS Offset address: 168_H

Security WDT PRS access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for PRS q** This bit enables read access to the access protected region for transactions with the PRS q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for PRS q** This bit enables write access to the access protected region for transactions with the PRS q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

41.4.27 Security WDT SMU partitions register

This register is to specify output of which SMU state machines will be considered for the SMU unlock-restriction (UR) feature. If more than one bit-field is set, their values are combined with OR logic.

WDTSEC_SMUFSP Offset address: 170_H

Security WDT SMU partitions register Application Reset value: $0000\ 0010_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0											CSPART	PART3	PART2	PART1	PART0
r											rw	rw	rw	rw	rw

Field	Bits	Type	Description
PARTx (x=0-3)	x	rw	**Consider signals from SMU state machine partition x** This bit enables that the outputs from SMU safe state machine partition x will be considered in unlock restriction feature (see also bit WDTx_STAT.UR). 0_B Outputs of corresponding SMU state machine are ignored inside this watchdog unit. 1_B Outputs of corresponding SMU state machine are considered inside this watchdog unit.

(table continues...)

(continued)

Field	Bits	Type	Description
CSPART	4	rw	**Consider signals from SMU CS-related partition** This bit enables that the outputs from SMU CS-related partition will be considered in unlock restriction feature (see also bit WDTx_STAT.UR). 0_B Outputs of corresponding SMU state machine are ignored inside this watchdog unit. 1_B Outputs of corresponding SMU state machine are considered inside this watchdog unit.
0	31:5	r	**Reserved** Read as 0; should be written with 0.

41.4.28 Security WDT control register A

This register is written with the expected password, the inverted estimated timer value and a lock-bit value.

The meaning of the individual values depends on the access type and active configuration fields STAT.PA and STAT.TC.

This register stores the current watchdog password value (PW) and lock control bit (LCK).

WDTSEC_CTRLA Offset address: 174_H

Security WDT control register A Reset values see: Table 1950

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TCVI															
w															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PW															LCK
rwh															rwh

Field	Bits	Type	Description
LCK	0	rwh	**Lock bit** LCK bit is cleared after a valid unlock-access to WDTx_CTRLA when unlock is not disabled. (Note: unlock can be disabled by UR-feature or by debug) LCK bit is set after a valid configuration-update-and-lock-access to WDTx_CTRLA. A valid check-access or shadow-config-update-access does not trigger a change of the LCK value. 0_B Registers WDTx_CTRLA and WDTx_CTRLB are unlocked 1_B Registers WDTx_CTRLA and WDTx_CTRLB are locked (default after reset)

(table continues...)

(continued)

Field	Bits	Type	Description
PW	15:1	rwh	**Password** This bit-field is written with a password value during an unlock- or check-access to WDTx_CTRLA: • If corresponding WDTx_STAT.PA == 0 then this bit-field must be written with its current contents. • If corresponding WDTx_STAT.PA == 1 then this bit-field must be written with the next password in the LFSR sequence. It is written with a new password value during a shadow-cfg-update-access to WDTx_CTRLA, and it is ignored during cfg-update-and-lock-access to WDTx_CTRLA. A read from this bit-field returns current password, but bits CTRLA[7:1] are bitwise inverted to ensure that a simple read-write is not sufficient to service the WDTx. The default password after reset is $007C_H$=000_0000_0111_1100$_B$ (due to inversion, SW will read it as 0003_H=000_0000_0000_0011$_B$).
TCVI	31:16	w	**Timer check value inverted** The timer-check value inverted (TCVI) is used only during unlock- and check-access and only if configured for use (WDTx_STAT.TC==1). In all other cases provided value is ignored. The value written to these bits will be bitwise-inverted and compared with the current timer value, with respect to the tolerance margin WDTx_STAT.TCT. If written value is outside defined margins it will result in an access-error. A read from this bit-field always returns 0000_H.

Table 1950 Reset values of WDTSEC_CTRLA

Reset type	Reset value	Note
Application Reset	$0000\ 00F9_H$	Default password value and locked.
After reset read value	$0000\ 0007_H$	Bits [7:1] (PW[6:0]) are read as bit-wise inverted

41.4.29 Security WDT control register B

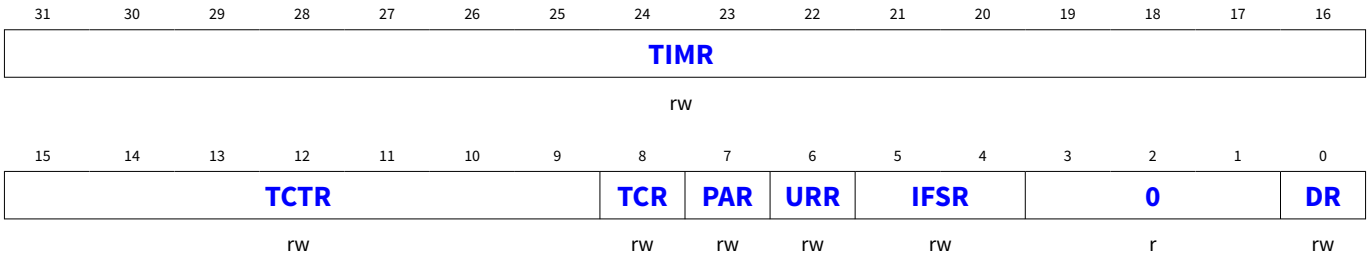
This register represents a shadow configuration for the watchdog timer.

WDTx_CTRLB bits can only be modified if corresponding WDTx_CTRLA.LCK is cleared - after a valid Unlock-Access.

Active configuration bits in WDTx_STAT are updated by these shadow/request bits only when LCK is set again - after a valid Config-Update-and-Lock-Access.

WDTSEC_CTRLB Offset address: 178_H

Security WDT control register B Application Reset value: $FFFC\ 0001_H$



Field	Bits	Type	Description
DR	0	rw	**Disable request bit** This bit represents a request (shadow) to disable the watchdog Timer. 0_B Request to enable the WDTx 1_B Request to disable the WDTx
IFSR	5:4	rw	**Input frequency selection request bits** Two bits of IFSR represent a request (shadow) to configure the watchdog timer frequency. 00_B Request to set input frequency to fSPB/16384. 01_B Request to set input frequency to fSPB/256. 10_B Request to set input frequency to fSPB/64. 11_B Reserved. Should not be used.
URR	6	rw	**Unlock restriction request bit** This bit represents a request (shadow) to enable unlock-restriction feature. If the unlock-restriction is set (WDTx_STAT.UR=='1') then WDTx cannot be unlocked in case respective SMU state machine is not in RUN state. 0_B Request to disable unlock-restriction feature 1_B Request to enable unlock-restriction feature
PAR	7	rw	**Password auto-sequence request bit** This bit represents a request (shadow) to configure a password-autosequencing feature. If the active password-autosequencing is set (WDTx_STAT.PA==1), after each unlock- and check-access, the password WDTx_CTRLA.PW will increment using built-in LFSR and store the new value. 0_B Request to disable automatic change of password after each valid unlock- or check-access 1_B Request to enable automatic sequence of password after each valid unlock- or check-access

(table continues...)

(continued)

Field	Bits	Type	Description
TCR	8	rw	**Timer check request bit** This bit represents a request (shadow) to configure a timer-check feature. If the active timer-check configuration is set (WDTx_STAT.TC==1), on each unlock- and check-access, the input TCVI will be compared to the current timer value WDTx_STAT.TIM with respect to the tolerance margin WDTx_STAT.TCT. 0_B Request to disable timer check feature during unlock- or check-access. 1_B Request to enable timer check feature during unlock- or check-access.
TCTR	15:9	rw	**Timer check tolerance request** This bit-field represents a request (shadow) to configure the timer-check-tolerance value.
TIMR	31:16	rw	**Timer base request value** This bit-field represents a request (shadow) to set the timer reload value. A read from this bit-field returns the timer reload value.
0	3:1	r	**Reserved** Read as 0; should be written with 0.

41.4.30 Security WDT status register

This register represents the active configuration of the watchdog timer.

These bits are read-only from the outside and they get updated by shadow-configuration bits from WDTx_CTRLB after a valid Config-Update-and-Lock-Access.

All WDTx are disabled by reset, and WDTSEC is not enabled before security SW starts.

WDTSEC_STAT Offset address: $17C_H$

Security WDT status register Application Reset value: $FFFC\ 0001_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TIM															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TCT							TC	PA	UR	IFS		AE	OE	TOM	D
rh							rh	rh	rh	rh		rh	rh	rh	rh

Field	Bits	Type	Description
D	0	rh	**Watchdog enable/disable status Flag** This bit represents watchdog timer's enable/disable state. This bit is cleared when time-out mode (unlock-state) is entered - after a valid unlock-access to corresponding CTRLA register. 0_B WDTx is enabled 1_B WDTx is disabled.
TOM	1	rh	**Watchdog time-out mode flag** This bit is set when time-out mode (unlock-state) is entered. It is automatically cleared when time-out mode (unlock-state) is left. 0_B The watchdog is not operating in time-out mode 1_B The watchdog is operating in time-out mode.
OE	2	rh	**Watchdog overflow error status flag** This bit is set when the WDTx timer value overflows from $FFFF_H$ to 0000_H. This bit is only cleared when WDTx_CTRLA.LCK changes state from 0 to 1 (after a valid config-update-and-lock-access). 0_B No watchdog overflow error. 1_B A watchdog overflow error has occurred.
AE	3	rh	**Watchdog access error status flag** This bit is set when: • An invalid unlock- or check-access to WDTx_CTRLA register was attempted. • A write access to WDTx_CTRLB register was attempted while WDTx_CTRLA.LCK is set. This bit is only cleared when WDTx_CTRLA.LCK changes state from 0 to 1 (after a valid config-update-and-lock-access). 0_B No watchdog access error. 1_B A watchdog access error has occurred.
IFS	5:4	rh	**Watchdog internal frequency selection status flag** These bits indicate the current WDTx clock rate. 00_B WDTx counter frequency is fSPB/16384 (default). 01_B WDTx counter frequency is fSPB/256. 10_B WDTx counter frequency is fSPB/64. 11_B Reserved. Should not be used.
UR	6	rh	**SMU unlock restriction status flag** This bit represents status of unlock-restriction feature for the Watchdog Timer. 0_B Unlocking of WDTx is not from SMU FSP side. 1_B Unlocking of WDTx not possible when corresponding SMU FSP state machine is not in RUN state

(table continues...)

(continued)

Field	Bits	Type	Description
PA	7	rh	**Password auto-sequence status flag** This bit represents control for password-autosequencing. 0_B No change of password after each unlock- or check-access. 1_B Automatically sequence the password after each valid unlock- or check-access using built-in LFSR.
TC	8	rh	**Timer check status flag** This bit represents control for timer-check - whether an additional check (next to password check) will be performed during unlock- or check-access. 0_B Additional timer-check during unlock- or check-access is disabled. 1_B Additional timer-check during unlock- or check-access is enabled.
TCT	15:9	rh	**Timer check tolerance** This bit-field represents the tolerance value of the timer check during unlock- or check-access. If timer-check is enabled, the timer value is combined with this tolerance value to create an upper and lower comparison boundary [max(TIM-TCT,0), min(TIM+TCT,$FFFF_H$)]. If time-check is enabled and provided value falls outside these boundaries an access error will be signaled to the SMU.
TIM	31:16	rh	**Timer value** This bit-field reflects the current value of the watchdog timer counter. Performing sequence of two accesses: a valid unlock-access followed by a valid config-update-and-lock-access, is servicing of the WDTx and the WDTx_STAT.TIM value gets reloaded by the value stored in WDTx_CTRLB.TIMR. The value reload into WDTx_STAT.TIM is not done in suspend mode.

41.4.31 System WDT PROT register safe endinit - SSE

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner and in some PROT states, the secure master to update the PROT state as described in the PROT mechanism.

The PROTSE register controls lock / unlock of the local Safe Endinit protected control registers.

WDTSYS_PROTSE Offset address: 184_H

System WDT PROT register safe endinit - SSE Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OWEN	ODEF	TAGID						PRSEN	PRS			VMEN	VM		
w	rw	rw						rw	rw			rw	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												SWEN	STATE		
r												w	rwh		

Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
VM	18:16	rw	**Virtual Machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual Machine definition Enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection Set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.
PRSEN	23	rw	**Protection Set definition Enable for PROT owner** 0_B **Disable**: PRS is not part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; should be written with 0.

Table 1951 Access mode restrictions of WDTSYS_PROTSE sorted by descending priority

Mode name	Access mode		Description
write 1 to .**SWEN**	rwh	STATE	
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

41.4.32 System WDT write access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

WDTSYS_ACCEN_WRA Offset address: 188_H

System WDT write access enable register A Application Reset value: FFFF FFFF$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	**EN30**	**EN29**	**EN28**	**EN27**	**EN26**	**EN25**	**EN24**	**EN23**	**EN22**	**EN21**	**EN20**	**EN19**	**EN18**	**EN17**	**EN16**
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	**EN14**	**EN13**	**EN12**	**EN11**	**EN10**	**EN09**	**EN08**	**EN07**	**EN06**	**EN05**	**EN04**	**EN03**	**EN02**	**EN01**	**EN00**
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access

41.4.33 System WDT write access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

WDTSYS_ACCEN_WRB Offset address: $18C_H$

System WDT write access enable register B Application Reset value: 0000 00FF$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								**EN39**	**EN38**	**EN37**	**EN36**	**EN35**	**EN34**	**EN33**	**EN32**
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access
0	31:8	r	**Reserved** Read as 0; should be written with 0.

41.4.34 System WDT read access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

WDTSYS_ACCEN_RDA Offset address: 190_H

System WDT read access enable register A Application Reset value: FFFF $FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access

41.4.35 System WDT read access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

WDTSYS_ACCEN_RDB Offset address: 194_H

System WDT read access enable register B Application Reset value: 0000 $00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access
0	31:8	r	**Reserved** Read as 0; should be written with 0.

41.4.36 System WDT VM access enable register

This register defines which virtual machine encodings are enabled or disabled for accesses to the access protected region. There is one RDq and WRq bit for each VMq.

WDTSYS_ACCEN_VM Offset address: 198_H

System WDT VM access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for VM ID q** This bit enables read access to the access protected region for transactions with the VM ID q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for VM ID q** This bit enables write access to the access protected region for transactions with the VM ID q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

41.4.37 System WDT PRS access enable register

This register defines which protection register set encodings are enabled or disabled for accesses to the access protected region. There is one RDq and WRq bit for each PRSq.

WDTSYS_ACCEN_PRS Offset address: $19C_H$

System WDT PRS access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for PRS q** This bit enables read access to the access protected region for transactions with the PRS q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for PRS q** This bit enables write access to the access protected region for transactions with the PRS q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

41.4.38 System WDT SMU partitions register

This register is to specify output of which SMU state machines will be considered for the SMU unlock-restriction (UR) feature. If more than one bit-field is set, their values are combined with OR logic.

For WDTSYS these bits have default (reset) value of 11111_B, meaning all SMU outputs are combined with OR logic for usage in UR feature.

WDTSYS_SMUFSP Offset address: $1A4_H$

System WDT SMU partitions register Application Reset value: $0000\ 001F_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0											CSPART	PART3	PART2	PART1	PART0
r											rw	rw	rw	rw	rw

Field	Bits	Type	Description
PARTx (x=0-3)	x	rw	**Consider signals from SMU state machine partition x** This bit enables that the outputs from SMU safe state machine partition x will be considered in unlock restriction feature (see also bit WDTx_STAT.UR). 0_B Outputs of corresponding SMU state machine are ignored inside this watchdog unit. 1_B Outputs of corresponding SMU state machine are considered inside this watchdog unit.
CSPART	4	rw	**Consider signals from SMU CS-related partition** This bit enables that the outputs from SMU CS-related partition will be considered in unlock restriction feature (see also bit WDTx_STAT.UR). 0_B Outputs of corresponding SMU state machine are ignored inside this watchdog unit. 1_B Outputs of corresponding SMU state machine are considered inside this watchdog unit.
0	31:5	r	**Reserved** Read as 0; should be written with 0.

41.4.39 System WDT control register A

This register is written with the expected password, the inverted estimated timer value and a lock-bit value.

The meaning of the individual values depends on the access type and active configuration fields STAT.PA and STAT.TC.

This register stores the current watchdog password value (PW) and lock control bit (LCK).

WDTSYS_CTRLA Offset address: $1A8_H$

System WDT control register A Reset values see: Table 1952

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TCVI															
w															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PW															LCK
rwh															rwh

Field	Bits	Type	Description
LCK	0	rwh	**Lock bit** LCK bit is cleared after a valid unlock-access to WDTx_CTRLA when unlock is not disabled. (Note: unlock can be disabled by UR-feature or by debug) LCK bit is set after a valid configuration-update-and-lock-access to WDTx_CTRLA. A valid check-access or shadow-config-update-access does not trigger a change of the LCK value. 0_B Registers WDTx_CTRLA and WDTx_CTRLB are unlocked 1_B Registers WDTx_CTRLA and WDTx_CTRLB are locked (default after reset)
PW	15:1	rwh	**Password** This bit-field is written with a password value during an unlock- or check-access to WDTx_CTRLA: • If corresponding WDTx_STAT.PA == 0 then this bit-field must be written with its current contents. • If corresponding WDTx_STAT.PA == 1 then this bit-field must be written with the next password in the LFSR sequence. It is written with a new password value during a shadow-cfg-update-access to WDTx_CTRLA, and it is ignored during cfg-update-and-lock-access to WDTx_CTRLA. A read from this bit-field returns current password, but bits CTRLA[7:1] are bitwise inverted to ensure that a simple read-write is not sufficient to service the WDTx. The default password after reset is $007C_H$=000_0000_0111_1100$_B$ (due to inversion, SW will read it as 0003_H=000_0000_0000_0011$_B$).
TCVI	31:16	w	**Timer check value inverted** The timer-check value inverted (TCVI) is used only during unlock- and check-access and only if configured for use (WDTx_STAT.TC==1). In all other cases provided value is ignored. The value written to these bits will be bitwise-inverted and compared with the current timer value, with respect to the tolerance margin WDTx_STAT.TCT. If written value is outside defined margins it will result in an access-error. A read from this bit-field always returns 0000_H.

Table 1952 Reset values of WDTSYS_CTRLA

Reset type	Reset value	Note
Application Reset	0000 00F9$_H$	Default password value and locked.
After reset read value	0000 0007$_H$	Bits [7:1] (PW[6:0]) are read as bit-wise inverted

41.4.40 System WDT control register B

This register represents a shadow configuration for the watchdog timer.

WDTx_CTRLB bits can only be modified if corresponding WDTx_CTRLA.LCK is cleared - after a valid Unlock-Access.

Active configuration bits in WDTx_STAT are updated by these shadow/request bits only when LCK is set again - after a valid Config-Update-and-Lock-Access.

WDTSYS_CTRLB Offset address: $1AC_H$

System WDT control register B Reset values see: Table 1953

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TIMR															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TCTR							TCR	PAR	URR	IFSR		0			DR
rw							rw	rw	rw	rw		r			rw

Field	Bits	Type	Description
DR	0	rw	**Disable request bit** This bit represents a request (shadow) to disable the watchdog Timer. 0_B Request to enable the WDTx 1_B Request to disable the WDTx
IFSR	5:4	rw	**Input frequency selection request bits** Two bits of IFSR represent a request (shadow) to configure the watchdog timer frequency. 00_B Request to set input frequency to fSPB/16384. 01_B Request to set input frequency to fSPB/256. 10_B Request to set input frequency to fSPB/64. 11_B Reserved. Should not be used.
URR	6	rw	**Unlock restriction request bit** This bit represents a request (shadow) to enable unlock-restriction feature. If the unlock-restriction is set (WDTx_STAT.UR=='1') then WDTx cannot be unlocked in case respective SMU state machine is not in RUN state. 0_B Request to disable unlock-restriction feature 1_B Request to enable unlock-restriction feature
PAR	7	rw	**Password auto-sequence request bit** This bit represents a request (shadow) to configure a password-autosequencing feature. If the active password-autosequencing is set (WDTx_STAT.PA==1), after each unlock- and check-access, the password WDTx_CTRLA.PW will increment using built-in LFSR and store the new value. 0_B Request to disable automatic change of password after each valid unlock- or check-access 1_B Request to enable automatic sequence of password after each valid unlock- or check-access

(table continues...)

(continued)

Field	Bits	Type	Description
TCR	8	rw	**Timer check request bit** This bit represents a request (shadow) to configure a timer-check feature. If the active timer-check configuration is set (WDTx_STAT.TC==1), on each unlock- and check-access, the input TCVI will be compared to the current timer value WDTx_STAT.TIM with respect to the tolerance margin WDTx_STAT.TCT. 0_B Request to disable timer check feature during unlock- or check-access. 1_B Request to enable timer check feature during unlock- or check-access.
TCTR	15:9	rw	**Timer check tolerance request** This bit-field represents a request (shadow) to configure the timer-check-tolerance value.
TIMR	31:16	rw	**Timer base request value** This bit-field represents a request (shadow) to set the timer reload value. A read from this bit-field returns the timer reload value.
0	3:1	r	**Reserved** Read as 0; should be written with 0.

Table 1953 Reset values of WDTSYS_CTRLB

Reset type	Reset value	Note
Application Reset	FFFC 0001$_H$	Request to enable with timer set to 0xFFFC.
After Boot-FW Value	FFFC 0000$_H$	WDTSYS is enabled after Boot-FW execution.

41.4.41 System WDT status register

This register represents the active configuration of the watchdog timer.

These bits are read-only from the outside and they get updated by shadow-configuration bits from WDTx_CTRLB after a valid Config-Update-and-Lock-Access.

All WDTx are disabled by reset, and WDTCPU0 and WDTSYS are enabled by FW before application SW starts.

WDTSYS_STAT Offset address: 1B0$_H$

System WDT status register Reset values see: Table 1954

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TIM															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TCT							TC	PA	UR	IFS		AE	OE	TOM	D
rh							rh	rh	rh	rh		rh	rh	rh	rh

Field	Bits	Type	Description
D	0	rh	**Watchdog enable/disable status Flag** This bit represents watchdog timer's enable/disable state. This bit is cleared when time-out mode (unlock-state) is entered - after a valid unlock-access to corresponding CTRLA register. 0_B WDTx is enabled 1_B WDTx is disabled.
TOM	1	rh	**Watchdog time-out mode flag** This bit is set when time-out mode (unlock-state) is entered. It is automatically cleared when time-out mode (unlock-state) is left. 0_B The watchdog is not operating in time-out mode 1_B The watchdog is operating in time-out mode.
OE	2	rh	**Watchdog overflow error status flag** This bit is set when the WDTx timer value overflows from $FFFF_H$ to 0000_H. This bit is only cleared when WDTx_CTRLA.LCK changes state from 0 to 1 (after a valid config-update-and-lock-access). 0_B No watchdog overflow error. 1_B A watchdog overflow error has occurred.
AE	3	rh	**Watchdog access error status flag** This bit is set when: • An invalid unlock- or check-access to WDTx_CTRLA register was attempted. • A write access to WDTx_CTRLB register was attempted while WDTx_CTRLA.LCK is set. This bit is only cleared when WDTx_CTRLA.LCK changes state from 0 to 1 (after a valid config-update-and-lock-access). 0_B No watchdog access error. 1_B A watchdog access error has occurred.
IFS	5:4	rh	**Watchdog internal frequency selection status flag** These bits indicate the current WDTx clock rate. 00_B WDTx counter frequency is fSPB/16384 (default). 01_B WDTx counter frequency is fSPB/256. 10_B WDTx counter frequency is fSPB/64. 11_B Reserved. Should not be used.
UR	6	rh	**SMU unlock restriction status flag** This bit represents status of unlock-restriction feature for the Watchdog Timer. 0_B Unlocking of WDTx is not from SMU FSP side. 1_B Unlocking of WDTx not possible when corresponding SMU FSP state machine is not in RUN state

(table continues...)

(continued)

Field	Bits	Type	Description
PA	7	rh	**Password auto-sequence status flag** This bit represents control for password-autosequencing. 0_B No change of password after each unlock- or check-access. 1_B Automatically sequence the password after each valid unlock- or check-access using built-in LFSR.
TC	8	rh	**Timer check status flag** This bit represents control for timer-check - whether an additional check (next to password check) will be performed during unlock- or check-access. 0_B Additional timer-check during unlock- or check-access is disabled. 1_B Additional timer-check during unlock- or check-access is enabled.
TCT	15:9	rh	**Timer check tolerance** This bit-field represents the tolerance value of the timer check during unlock- or check-access. If timer-check is enabled, the timer value is combined with this tolerance value to create an upper and lower comparison boundary [max(TIM-TCT,0), min(TIM+TCT,$FFFF_H$)]. If time-check is enabled and provided value falls outside these boundaries an access error will be signaled to the SMU.
TIM	31:16	rh	**Timer value** This bit-field reflects the current value of the watchdog timer counter. Performing sequence of two accesses: a valid unlock-access followed by a valid config-update-and-lock-access, is servicing of the WDTx and the WDTx_STAT.TIM value gets reloaded by the value stored in WDTx_CTRLB.TIMR. The value reload into WDTx_STAT.TIM is not done in suspend mode.

Table 1954 Reset values of WDTSYS_STAT

Reset type	Reset value	Note
After Boot-FW Value	XXXX 0000_H	WDTSYS is enabled after Boot-FW execution, exact timer value depends at which moment timer is read.

41.5 Debug information

41.5.1 Suspend mode support

During a debug session, the watchdog timer functionality might lead to some unintended alarms.

By default, the WDTs are disabled when OCDS is enabled. However, it is possible to enable WDTs even when OCDS is enabled if CBS_OSTATE.WDTSUS=1.

WDTCPUy may only be re-enabled if the corresponding CPUy_DBGSR.SUSOUT is inactive. Similarly WDTSEC may be enabled if CPU6_DBGSR.SUSOUT is inactive. In WDTSYS CBS_TLS.TL1 is used for this purpose.

For safety reasons it is essential that WDTCPUy ignores CPUy_DBGSR.SUSOUT, WDTSEC ignores CPU6_DBGSR.SUSOUT, and WDTSYS ignores CBS_TLS.TL1 when OCDS is disabled.

Table 1955 CPUy WDT suspend state

WDTx_STAT.D (x=CPUy, SEC,SYS)	CBS_OSTATE.OEN	CBS_OSTATE.WDTSUS	WDTCPUy → CPUy_DBGSR.SUSOUT WDTSEC → CPU6_DBGSR.SUSOUT WDTSYS → CBS_TLS.TL1	WDTx state
1	X	X	X	Stopped
0	0	X	X	Running
0	1	0	X	Stopped
0	1	1	0	Running
0	1	1	1	Stopped

41.6 References

This section is not applicable for the module being described.

41.7 WTU revision history

Initial release of the chapter.

41.8 TC4Dx WTU information

41.8.1 TC4Dx WTU configuration

There are no deviations from the generic specification.

41.8.2 TC4Dx WTU features

There are no deviations from the generic specification.

41.8.3 TC4Dx WTU functional description

There are no deviations from the generic specification.

41.8.4 TC4Dx WTU registers

41.8.4.1 Register address space - WTU

Table 1956 Registers address space - WTU

Module	Base address	End address	Note
WTU	$F0000000_H$	$F00007FF_H$	WDT: Connections to FPI/BPI bus

41.8.4.2 Register overview - access mode glossary

Table 1957 Register overview - access mode glossary

Keyword	Description
CSE	Access protection using PROT register WTU_WDTSEC_PROTCSE .
SEy (y=0-5)	Access protection using PROT register WTU_WDTCPUy_PROTSE .
SSE	Access protection using PROT register WTU_WDTSYS_PROTSE .
APU-PCS	Protection group consisting of registers WTU_WDTSEC_ACCEN_WRA , WTU_WDTSEC_ACCEN_WRB , WTU_WDTSEC_ACCEN_RDA , WTU_WDTSEC_ACCEN_RDB , WTU_WDTSEC_ACCEN_VM , WTU_WDTSEC_ACCEN_PRS .
PCS	Access protection using *APU-PCS* registers.
APU-PSYS	Protection group consisting of registers WTU_WDTSYS_ACCEN_WRA , WTU_WDTSYS_ACCEN_WRB , WTU_WDTSYS_ACCEN_RDA , WTU_WDTSYS_ACCEN_RDB , WTU_WDTSYS_ACCEN_VM , WTU_WDTSYS_ACCEN_PRS .
PSYS	Access protection using *APU-PSYS* registers.
APU-Py (y=0-5)	Protection group consisting of registers WTU_WDTCPUy_ACCEN_WRA , WTU_WDTCPUy_ACCEN_WRB , WTU_WDTCPUy_ACCEN_RDA , WTU_WDTCPUy_ACCEN_RDB , WTU_WDTCPUy_ACCEN_VM , WTU_WDTCPUy_ACCEN_PRS .
Py	Access protection using *APU-Py* registers.
BE	Always returns a Bus Error.
U	No access restrictions.
PROT	Access restrictions as defined in the PROT register access rules.
32	Access only when using 32-bit width.

41.8.4.3 Registers overview - WTU (ascending offset address)

Table 1958 Registers overview - WTU (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
WTU_ID	Module identification register	008_H	PSYS	BE	PowerOn Reset	7003
WTU_WDTCPUy_PROTSE (y=0-5)	CPUy WDT PROT register safe endinit - SEy	018_H+y* 30_H	U	SV, PROT	Application Reset	7004

(table continues...)

Table 1958 (continued) Registers overview - WTU (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
WTU_WDTCPUy_ACCEN_WRA (y=0-5)	CPUy WDT write access enable register A	$01C_H$+y* 30_H	U	SEy, SV	Application Reset	7006
WTU_WDTCPUy_ACCEN_WRB (y=0-5)	CPUy WDT write access enable register B	020_H+y* 30_H	U	SEy, SV	Application Reset	7011
WTU_WDTCPUy_ACCEN_RDA (y=0-5)	CPUy WDT read access enable register A	024_H+y* 30_H	U	SEy, SV	Application Reset	7012
WTU_WDTCPUy_ACCEN_RDB (y=0-5)	CPUy WDT read access enable register B	028_H+y* 30_H	U	SEy, SV	Application Reset	7012
WTU_WDTCPUy_ACCEN_VM (y=0-5)	CPUy WDT VM access enable register	$02C_H$+y* 30_H	U	SEy, SV	Application Reset	7013
WTU_WDTCPUy_ACCEN_PRS (y=0-5)	CPUy WDT PRS access enable register	030_H+y* 30_H	U	SEy, SV	Application Reset	7014
WTU_WDTCPUy_SMUFSP (y=0-5)	CPUy WDT SMU partitions register	038_H+y* 30_H	Py	Py, SV	Application Reset	7014
WTU_WDTCPUy_CTRLA (y=0-5)	CPUy WDT control register A	$03C_H$+y* 30_H	Py	Py, 32	7015	7015
WTU_WDTCPUy_CTRLB (y=0-5)	CPUy WDT control register B	040_H+y* 30_H	Py	Py	See 7049	7049
WTU_WDTCPUy_STAT (y=0-5)	CPUy WDT status register	044_H+y* 30_H	Py	BE	See 7051	7051
WTU_WDTSEC_PROTCSE	Security WDT PROT register cyber-secure endinit - CSE	150_H	U	SV, PROT	Application Reset	7020
WTU_WDTSEC_ACCEN_WRA	Security WDT write access enable register A	154_H	U	CSE, SV	Application Reset	7022
WTU_WDTSEC_ACCEN_WRB	Security WDT write access enable register B	158_H	U	CSE, SV	Application Reset	7023
WTU_WDTSEC_ACCEN_RDA	Security WDT read access enable register A	$15C_H$	U	CSE, SV	Application Reset	7024
WTU_WDTSEC_ACCEN_RDB	Security WDT read access enable register B	160_H	U	CSE, SV	Application Reset	7024
WTU_WDTSEC_ACCEN_VM	Security WDT VM access enable register	164_H	U	CSE, SV	Application Reset	7025

(table continues...)

Table 1958 (continued) Registers overview - WTU (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
WTU_WDTSEC_ACCEN_PRS	Security WDT PRS access enable register	168_H	U	CSE, SV	Application Reset	7025
WTU_WDTSEC_SMUFSP	Security WDT SMU partitions register	170_H	PCS	PCS, SV	Application Reset	7026
WTU_WDTSEC_CTRLA	Security WDT control register A	174_H	PCS	PCS, 32	7027	7027
WTU_WDTSEC_CTRLB	Security WDT control register B	178_H	PCS	PCS	Application Reset	7028
WTU_WDTSEC_STAT	Security WDT status register	$17C_H$	PCS	BE	Application Reset	7030
WTU_WDTSYS_PROTSE	System WDT PROT register safe endinit - SSE	184_H	U	SV, PROT	Application Reset	7032
WTU_WDTSYS_ACCEN_WRA	System WDT write access enable register A	188_H	U	SSE, SV	Application Reset	7034
WTU_WDTSYS_ACCEN_WRB	System WDT write access enable register B	$18C_H$	U	SSE, SV	Application Reset	7034
WTU_WDTSYS_ACCEN_RDA	System WDT read access enable register A	190_H	U	SSE, SV	Application Reset	7035
WTU_WDTSYS_ACCEN_RDB	System WDT read access enable register B	194_H	U	SSE, SV	Application Reset	7035
WTU_WDTSYS_ACCEN_VM	System WDT VM access enable register	198_H	U	SSE, SV	Application Reset	7036
WTU_WDTSYS_ACCEN_PRS	System WDT PRS access enable register	$19C_H$	U	SSE, SV	Application Reset	7036
WTU_WDTSYS_SMUFSP	System WDT SMU partitions register	$1A4_H$	PSYS	PSYS, SV	Application Reset	7037
WTU_WDTSYS_CTRLA	System WDT control register A	$1A8_H$	PSYS	PSYS, 32	7038	7038
WTU_WDTSYS_CTRLB	System WDT control register B	$1AC_H$	PSYS	PSYS	7040	7040
WTU_WDTSYS_STAT	System WDT status register	$1B0_H$	PSYS	BE	7041	7041

41.8.4.4 CPUy WDT control register B

This register represents a shadow configuration for the watchdog timer.

WDTx_CTRLB bits can only be modified if corresponding WDTx_CTRLA.LCK is cleared - after a valid unlock-access.

Active configuration bits in WDTx_STAT are updated by these shadow/request bits only when LCK is set again - after a valid config-update-and-lock-access.

WTU_WDTCPUy_CTRLB (y=0) — Offset address: 040_H+y*30_H

CPUy WDT control register B — Reset values see: Table 1959

WTU_WDTCPUy_CTRLB (y=1-5) — Offset address: 040_H+y*30_H

CPUy WDT control register B — Application Reset value: $FFFC\ 0001_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TIMR															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TCTR							TCR	PAR	URR	IFSR		0			DR
rw							rw	rw	rw	rw		r			rw

Field	Bits	Type	Description
DR	0	rw	**Disable request bit** This bit represents a request (shadow) to disable the watchdog Timer. 0_B Request to enable the WDTx 1_B Request to disable the WDTx
IFSR	5:4	rw	**Input frequency selection request bits** Two bits of IFSR represent a request (shadow) to configure the watchdog timer frequency. 00_B Request to set input frequency to fSPB/16384. 01_B Request to set input frequency to fSPB/256. 10_B Request to set input frequency to fSPB/64. 11_B Reserved. Should not be used.
URR	6	rw	**Unlock restriction request bit** This bit represents a request (shadow) to enable unlock-restriction feature. If the unlock-restriction is set (WDTx_STAT.UR=='1') then WDTx cannot be unlocked in case respective SMU state machine is not in RUN state. 0_B Request to disable unlock-restriction feature 1_B Request to enable unlock-restriction feature

(table continues...)

(continued)

Field	Bits	Type	Description
PAR	7	rw	**Password auto-sequence request bit** This bit represents a request (shadow) to configure a password-autosequencing feature. If the active password-autosequencing is set (WDTx_STAT.PA==1), after each unlock- and check-access, the password WDTx_CTRLA.PW will increment using built-in LFSR and store the new value. 0_B Request to disable automatic change of password after each valid unlock- or check-access 1_B Request to enable automatic sequence of password after each valid unlock- or check-access
TCR	8	rw	**Timer check request bit** This bit represents a request (shadow) to configure a timer-check feature. If the active timer-check configuration is set (WDTx_STAT.TC==1), on each unlock- and check-access, the input TCVI will be compared to the current timer value WDTx_STAT.TIM with respect to the tolerance margin WDTx_STAT.TCT. 0_B Request to disable timer check feature during unlock- or check-access. 1_B Request to enable timer check feature during unlock- or check-access.
TCTR	15:9	rw	**Timer check tolerance request** This bit-field represents a request (shadow) to configure the timer-check-tolerance value.
TIMR	31:16	rw	**Timer base request value** This bit-field represents a request (shadow) to set the timer reload value. A read from this bit-field returns the timer reload value.
0	3:1	r	**Reserved** Read as 0; should be written with 0.

Table 1959 Reset values of WTU_WDTCPUy_CTRLB (y=0)

Reset type	Reset value	Note
Application Reset	FFFC 0001$_H$	Request to enable with timer set to 0xFFFC.
After Boot-FW Value	FFFC 0000$_H$	Only WDTCPU0 is enabled during Boot-FW execution.

41.8.4.5 CPUy WDT status register

This register represents the active configuration of the watchdog timer.

These bits are read-only from the outside and they get updated by shadow-configuration bits from WDTx_CTRLB after a valid config-update-and-lock-access.

All WDTx are disabled by reset, and only WDTCPU0 and WDTSYS are enabled by FW before application SW starts.

WTU_WDTCPUy_STAT (y=0) — Offset address: 044_H+y*30_H

CPUy WDT status register — Reset values see: Table 1960

WTU_WDTCPUy_STAT (y=1-5) — Offset address: 044_H+y*30_H

CPUy WDT status register — Application Reset value: $FFFC\ 0001_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TIM															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TCT							TC	PA	UR	IFS		AE	OE	TOM	D
rh							rh	rh	rh	rh		rh	rh	rh	rh

Field	Bits	Type	Description
D	0	rh	**Watchdog enable/disable status Flag** This bit represents watchdog timer's enable/disable state. This bit is cleared when time-out mode (unlock-state) is entered - after a valid unlock-access to corresponding CTRLA register. 0_B WDTx is enabled 1_B WDTx is disabled.
TOM	1	rh	**Watchdog time-out mode flag** This bit is set when time-out mode (unlock-state) is entered. It is automatically cleared when time-out mode (unlock-state) is left. 0_B The watchdog is not operating in time-out mode 1_B The watchdog is operating in time-out mode.
OE	2	rh	**Watchdog overflow error status flag** This bit is set when the WDTx timer value overflows from $FFFF_H$ to 0000_H. This bit is only cleared when WDTx_CTRLA.LCK changes state from 0 to 1 (after a valid config-update-and-lock-access). 0_B No watchdog overflow error. 1_B A watchdog overflow error has occurred.

(table continues...)

(continued)

Field	Bits	Type	Description
AE	3	rh	**Watchdog access error status flag** This bit is set when: • An invalid unlock- or check-access to WDTx_CTRLA register was attempted. • A write access to WDTx_CTRLB register was attempted while WDTx_CTRLA.LCK is set. This bit is only cleared when WDTx_CTRLA.LCK changes state from 0 to 1 (after a valid config-update-and-lock-access). 0_B No watchdog access error. 1_B A watchdog access error has occurred.
IFS	5:4	rh	**Watchdog internal frequency selection status flag** These bits indicate the current WDTx clock rate. 00_B WDTx counter frequency is fSPB/16384 (default). 01_B WDTx counter frequency is fSPB/256. 10_B WDTx counter frequency is fSPB/64. 11_B Reserved. Should not be used.
UR	6	rh	**SMU unlock restriction status flag** This bit represents status of unlock-restriction feature for the Watchdog Timer. 0_B Unlocking of WDTx is not from SMU FSP side. 1_B Unlocking of WDTx not possible when corresponding SMU FSP state machine is not in RUN state
PA	7	rh	**Password auto-sequence status flag** This bit represents control for password-autosequencing. 0_B No change of password after each unlock- or check-access. 1_B Automatically sequence the password after each valid unlock- or check-access using built-in LFSR.
TC	8	rh	**Timer check status flag** This bit represents control for timer-check - whether an additional check (next to password check) will be performed during unlock- or check-access. 0_B Additional timer-check during unlock- or check-access is disabled. 1_B Additional timer-check during unlock- or check-access is enabled.
TCT	15:9	rh	**Timer check tolerance** This bit-field represents the tolerance value of the timer check during unlock- or check-access. If timer-check is enabled, the timer value is combined with this tolerance value to create an upper and lower comparison boundary [max(TIM-TCT,0), min(TIM+TCT,$FFFF_H$)]. If time-check is enabled and provided value falls outside these boundaries an access error will be signaled to the SMU.

(table continues...)

(continued)

Field	Bits	Type	Description
TIM	31:16	rh	**Timer value** This bit-field reflects the current value of the watchdog timer counter. Performing sequence of two accesses: a valid unlock-access followed by a valid config-update-and-lock-access, is servicing of the WDTx and the WDTx_STAT.TIM value gets reloaded by the value stored in WDTx_CTRLB.TIMR. The value reload into WDTx_STAT.TIM is not done in suspend mode.

Table 1960 Reset values of WTU_WDTCPUy_STAT (y=0)

Reset type	Reset value	Note
Application Reset	FFFC 0001$_H$	WDTCPUy (y=0-5) are disabled with default timer value of 0xFFFC.
After Boot-FW Value	XXXX 0000$_H$	Only WDTCPU0 is enabled during Boot-FW execution. Exact timer value depends at which moment timer is read.

41.8.5 TC4DX WTU connectivity

The table below lists interfaces to and from this functional block to other blocks in the device.

Table 1961 List of WTU interface signals

Interface signals	I/O	Description
CLOCK_WTU_fSPB	In	CLOCK
FPI_WTU_SPB	In	System Peripheral Bus
DEBUG_WTU_OCDS_ENABLE	In	Cerberus OCDS enable signal
DEBUG_WTU_OCDS_WDTSUS	In	Cerberus OCDS WDT suspend signal
DEBUG_WTU_OCDS_TLS_TL1	In	Cerberus OCDS TSL TL1 suspend signal
CPU0_WTU_CPUxSUSOUT	In	Non-security CPUs suspend state signal x=[5:0]
CPU1_WTU_CPUxSUSOUT	In	Non-security CPUs suspend state signal x=[5:0]
CPU2_WTU_CPUxSUSOUT	In	Non-security CPUs suspend state signal x=[5:0]
CPU3_WTU_CPUxSUSOUT	In	Non-security CPUs suspend state signal x=[5:0]
CPU4_WTU_CPUxSUSOUT	In	Non-security CPUs suspend state signal x=[5:0]
CPU5_WTU_CPUxSUSOUT	In	Non-security CPUs suspend state signal x=[5:0]
CPUCS_WTU_CPUxSUSOUT	In	Non-security CPUs suspend state signal x=[5:0]
CPUCS_WTU_SUSPEND	In	Security CPU suspend state signal
SMU_WTU_SAFE_RUN_STATE[3:0]	In	SMU_SAFE0/1 State Machines SSM0/1 RUN State information to WTU
SMU_WTU_CS_RUN_STATE	In	SMU_CS State Machine SSM RUN State information to WTU
SMM_WTU_INIT_DONE	In	Access protection initialization done
WTU_SMM_WAKEUP_CPU0	Out	Wake-up request for CPU0 to SMM

(table continues...)

Table 1961 (continued) List of WTU interface signals

Interface signals	I/O	Description
WTU_SMM_WAKEUP_CPU1	Out	Wake-up request for CPU1 to SMM
WTU_SMM_WAKEUP_CPU2	Out	Wake-up request for CPU2 to SMM
WTU_SMM_WAKEUP_CPU3	Out	Wake-up request for CPU3 to SMM
WTU_SMM_WAKEUP_CPU4	Out	Wake-up request for CPU4 to SMM
WTU_SMM_WAKEUP_CPU5	Out	Wake-up request for CPU5 to SMM
WTU_SMM_WAKEUP_CS	Out	Wake-up request for CSRM-core to SMM
WTU_SMU_ALM_WDTCPU0	Out	Access and Overflow Error alarms from WDTCPU0 to SMU
WTU_SMU_ALM_WDTCPU1	Out	Access and Overflow Error alarms from WDTCPU1 to SMU
WTU_SMU_ALM_WDTCPU2	Out	Access and Overflow Error alarms from WDTCPU2 to SMU
WTU_SMU_ALM_WDTCPU3	Out	Access and Overflow Error alarms from WDTCPU3 to SMU
WTU_SMU_ALM_WDTCPU4	Out	Access and Overflow Error alarms from WDTCPU4 to SMU
WTU_SMU_ALM_WDTCPU5	Out	Access and Overflow Error alarms from WDTCPU5 to SMU
WTU_SMU_ALM_WDTSEC	Out	Access and Overflow Error alarms from WDTSEC to SMU
WTU_SMU_ALM_WDTSYS	Out	Access and Overflow Error alarms from WDTSYS to SMU
WTU_SMU_ALM_WTU	Out	Combined alarm to SMU
WTU_SMU_SFF_ALARM	Out	WTU SFF uncorrectable error alarm

41.8.6 TC4Dx WTU revision history

Initial release of the chapter.

42 Safety and security alarm Management Unit (SMU)

As a central component of the safety and security architecture, the SMU provides a generic interface to manage the behavior of the microcontroller under the presence of safety critical faults and security events. These faults and security events are signaled to the SMU as alarms. Each alarm can be individually configured to trigger internal reactions and notify externally through a fault signaling protocol.

42.1 Feature list

- Collect every alarm signal generated from safety mechanisms and security mechanisms
- Process safety alarms with redundant safety alarm handlers
- Each safety alarm handler can be configured to activate a number of internal reactions:
  - Generate an interrupt request to any of the CPUs. Concurrent interrupts to several CPUs can be configured
  - Generate a Non-Maskable Interrupt (NMI) request to the System Control Unit (SCU)
  - Generate a reset request to the System Mode Management (SMM)
  - Activate the Port Emergency Stop signal controlling the safe state of output pads
  - Generate a CPU module reset request
  - Generate a PPU module reset request
- Each safety alarm handler can be configured to report internal faults to the external environment using the Fault Signaling Protocol (FSP). The following FSP modes are available:
  - Bi-stable single pin output, also called ErrorPin
  - Timed dual rail coding using two inverted values on the ErrorPins
  - Single-bit timed protocol using the ErrorPin
- Each safety alarm handler can be configured to trigger emergency stop request to the PORTS
- Diverse processing of safety alarms generated by certain safety mechanisms (voltage, clock, and temperature monitors) detecting potential common cause faults
- Process security alarms with security alarm handler
- Security alarm handler can be configured to activate internal reactions:
  - Generate an interrupt request to any of the CPUs including CPUcs if present
  - Generate a NMI request to the SCU
  - Generate a reset request to the SCU
  - Locking KEYS request to the Cyber Security Satellite (CSS)
- A subset of safety and security alarms can be configured to be processed either by safety alarm handler or security alarm handler
- Monitor correct internal reaction execution by watchdog timers
- Test alarm handlers reaction generation by software based alarm emulation
- Control the test mechanism available for safety flip-flop instances within the microcontroller and capture the error condition during the test
- Protect configuration registers against unintended accesses using a diverse hardware access protection mechanism

42.2 Functional overview

The SMU is a central component for handling safety and security alarms. Based on the safety concept several hardware and software-based safety mechanisms are implemented within the microcontroller to detect presence of faults. These faults are signaled to SMU as safety alarms. Please refer to the Safety Manual for a detailed description of which safety mechanisms trigger which alarms.

Based on the security concept, several hardware and software-based security mechanisms are implemented within the microcontroller to detect presence of security events. These security events are signaled to SMU as

security alarms. Certain alarms can be categorized to be safety or security relevant only based on usage of microcontroller. These alarms are categorized as shared alarms and configured during start-up.

SMU_CS is a security alarm handler. Within SMU_CS each security alarm can be individually configured to trigger certain reactions. These reactions will be discussed in detail in section SMU_CS. An independent access protection mechanism is present to protect the configuration of SMU_CS. SMU_CS is implemented within core domain, that is, provided by core voltage and f_{SPB} clock (refer to the PMS and Clocking chapter of the UM for more details).

Two safety alarm handlers instances SMU_SAFE0 and SMU_SAFE1 are also present in core domain. The architectural design of both of the instances are exactly the same. All safety alarms are mapped to both instances. Within each instance each safety alarm can be individually configured to trigger certain reactions. These reactions will be discussed in detail in section SMU_SAFE. A separate access protection mechanism is present to protect the configuration of SMU_SAFE0 and SMU_SAFE1 individually.

A shared alarm selector is present which can be configured during start-up to map shared alarm to either Security Alarm Handler SMU_CS or to Safety Alarm Handlers SMU_SAFE0 and SMU_SAFE1.

In order to mitigate the potential common cause faults inside the MCU, an independent alarm handler SMU_STDBY is present in standby domain. SMU_STDBY is provided by back-up voltage and f_{BACK} clock (refer to the PMS and Clocking chapter of the user manual for more details). Each safety alarm which is mapped to SMU_STDBY can be individually configured to trigger error notification on the external Fault Signaling Protocol (FSP) error pins. All power and temperature related alarms are processed in a diverse way since they are mapped to core domain alarm handlers (SMU_SAFE0 and SMU_SAFE1) and standby domain alarm handler SMU_STDBY. The clock alive monitor alarms are processed in the same clock domain as they are generated. Moreover in order to detect errors in the core domain alarm handlers, SMU_SAFE0 Alive Alarm, SMU_SAFE1 Alive Alarm and SMU_CS Alive Alarm are sent from the corresponding SMU_SAFE0, SMU_SAFE1 and SMU_CS to the SMU_STDBY upon the detection of error.

SMU Global Control and Configuration (SMU_GCC) is present to handle certain configuration of the SMU globally which cannot be performed by individual alarm handlers. SMU_GCC takes care of global configuration for SMU alarm handlers (SMU_SAFE0, SMU_SAFE1 and SMU_CS) in the core domain. Emergency Stop Unit, which triggers port emergency stop request to port, and Register Monitor Unit, to test the safety flip-flops, are also present in SMU_GCC. SMU_GCC is also having independent access protection mechanism.

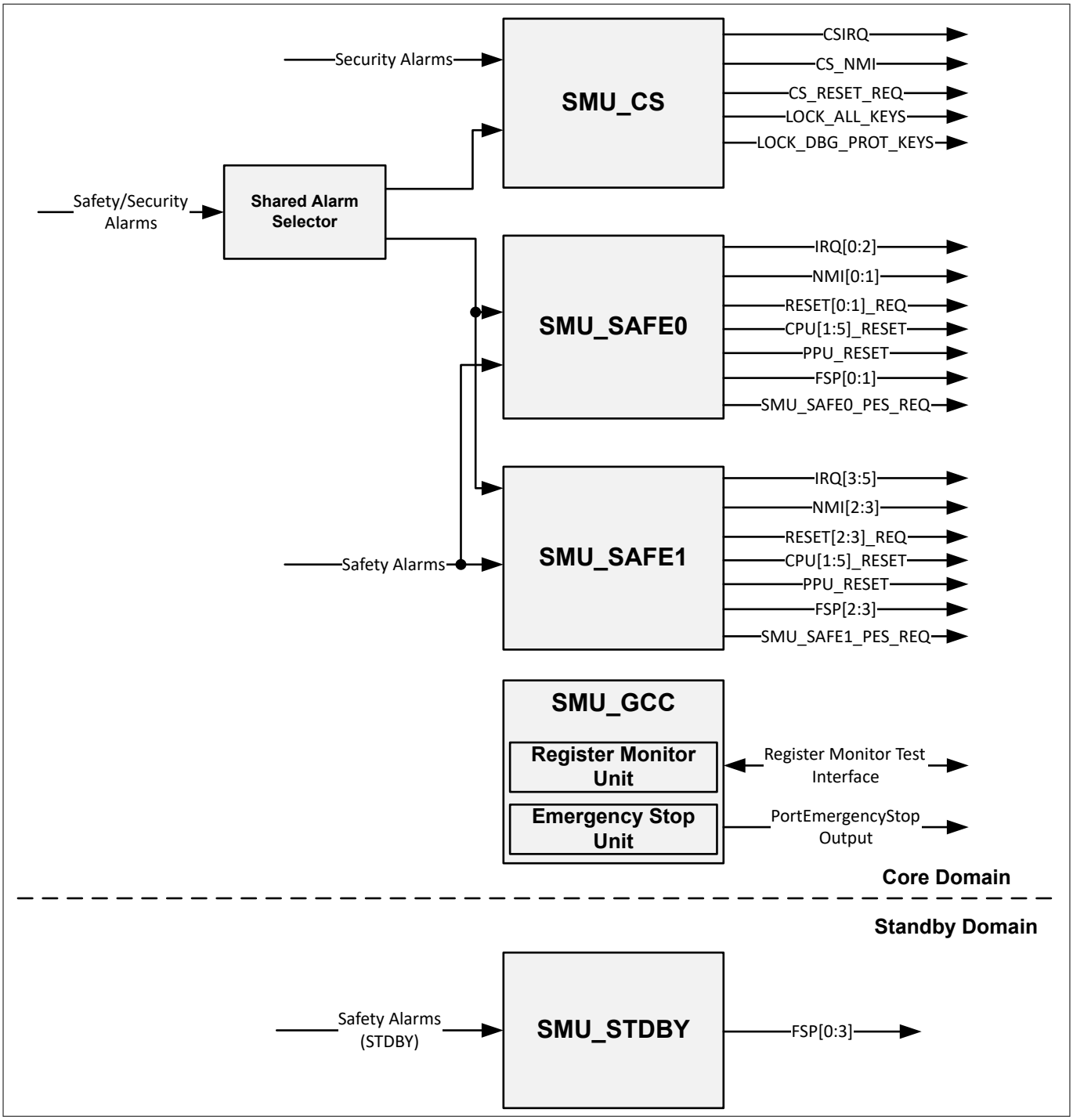


Figure 973 SMU block diagram

For the alarm description please refer to the 'TC4xx SMU alarm mapping tables' subchapter and to the Safety Manual.

Related information

CLOCK on page 5525
Power Management System (PMS) on page 5946
TC4Dx SMU alarm mapping tables on page 7233

42.3 Functional description

The following sections describe SMU_SAFE0, SMU_SAFE1, SMU_CS and SMU_STDBY functionality in detail.

42.3.1 Alarm mapping

Please refer to the product specific section of this document to find the product specific Alarm Mapping Tables.

All the incoming alarms are classified into alarm groups. The alarm groups define a logical mapping used to map alarm signals to internal configuration and status registers. Hence, a maximum of 32 alarm signals can be mapped to a single alarm group. Within the context of SMU, an alarm group is specified by using the three letters 'ALM' followed by alarm type and then index. For example ALMCS0 refers to Alarm Group 0 for cyber security alarms. The following table gives an overview about the alarm groups and their mapping to Alarm Handler.

Table 1962 Alarm group to alarm handler mapping

Alarm group	Alarm type	Alarm handler
ALMCS0 - ALMCS1	Security Alarms	SMU_CS
ALMSF0 - ALMSF16	Safety Alarms	SMU_SAFE0 and SMU_SAFE1
ALMSH0 - ALMSH2	Shared (Security or safety) Alarms	SMU_CS, SMU_SAFE0 and SMU_SAFE1
ALMSTDBY0 - ALMSTDBY4	Safety (Standby) Alarms	SMU_STDBY

42.3.2 Shared alarm selector

A selector which can be configured to map shared alarm group alarms (ALMSHi) to either SMU_CS or SMU_SAFE instances (SMU_SAFE0 and SMU_SAFE1).

As introduced earlier, some alarms can be freely mapped to either safety or security alarm handler depending on application use case. For that a Shared Alarm Selector is present to map the shared alarm group alarms (ALMSHi) to either SMU_CS or SMU_SAFE instances (SMU_SAFE0 and SMU_SAFE1). The configuration of the Shared Alarm Selector is done during start-up where Security and Application software configure their appropriate registers for shared alarm group alarms (ALMSHi) mapping.

The Shared Alarm Selector can be configured by registers CS_AGSHi_SEL and GCC_AGSHi_SEL. The register CS_AGSHi_SEL is used to map the alarms to SMU_CS. Correspondingly, GCC_AGSHi_SEL is used to map the alarms to SMU_SAFE instances (SMU_SAFE0 and SMU_SAFE1). The status register CS_AGSHi_SS is used to indicate which alarms in shared alarm group is mapped to SMU_SAFE instances (SMU_SAFE0 and SMU_SAFE1). Correspondingly, the status register GCC_AGSHi_SS is used to indicate which alarms in shared alarm group is mapped to SMU_CS. The Shared Alarm Selector illustration is shown below :

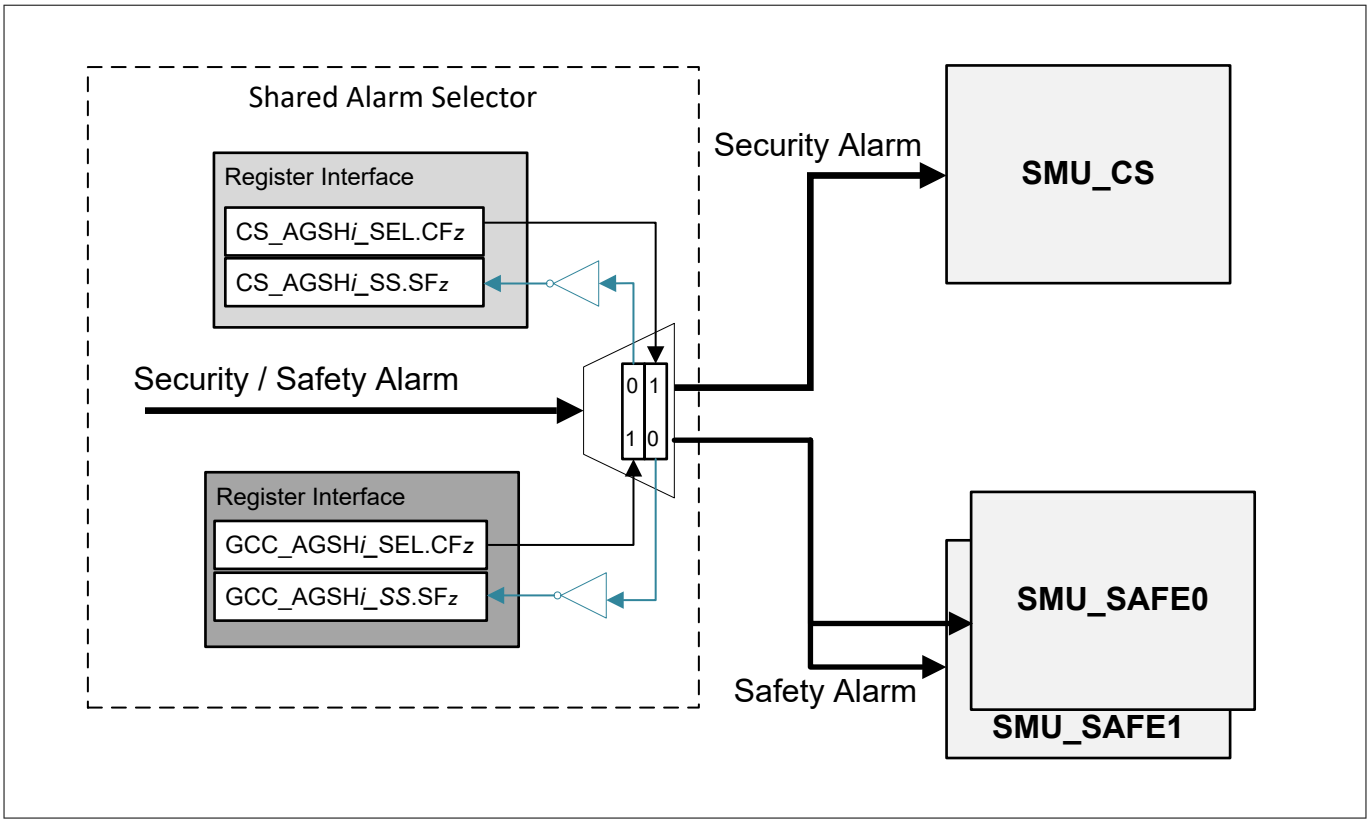


Figure 974 Shared alarm selector: An overview

A bit level locking mechanism is provided for configuration registers CS_AGSHi_SEL and GCC_AGSHi_SEL. The CS_AGSHi_SEL.CFz flag can only be written when the corresponding flag in GCC_AGSHi_SEL.CFz is set to '0' and the GCC_AGSHi_SEL.CFz flag can only be written when the corresponding flag in CS_AGSHi_SEL.CFz is set to '0'. This behavior is explained by an example shown in the following figure :

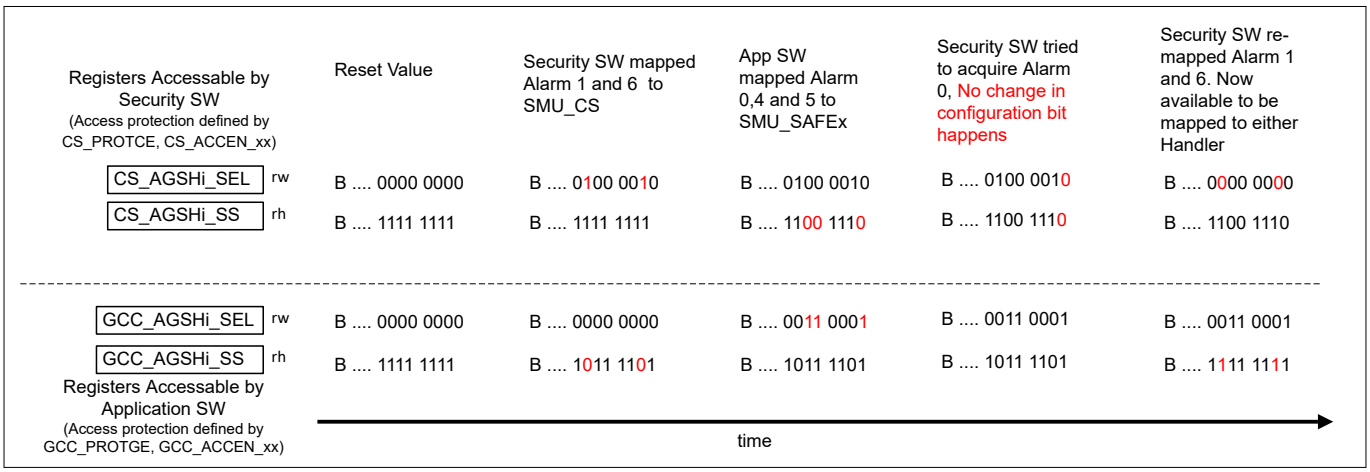


Figure 975 Bit level locking mechanism example for CS_AGSHi_SEL and GCC_AGSHi_SEL registers

The select status registers SAFEx_AGSHi_SS, where x = 0 and 1, are present inside SMU_SAFE instances and hold the exact copy of GCC_AGSHi_SEL. These registers are present to notify corresponding SMU_SAFE0 and SMU_SAFE1 instances about shared alarm group mapping.

42.3.3 SMU_SAFE

Two identical instances SMU_SAFE0 and SMU_SAFE1 are present for handling safety alarms in the core domain. The functionality of both instances is exactly the same. Both SMU_SAFE0 and SMU_SAFE1 instances share input alarms but have their own configuration registers and functional interfaces on outputs side. This is explained in the figure below :

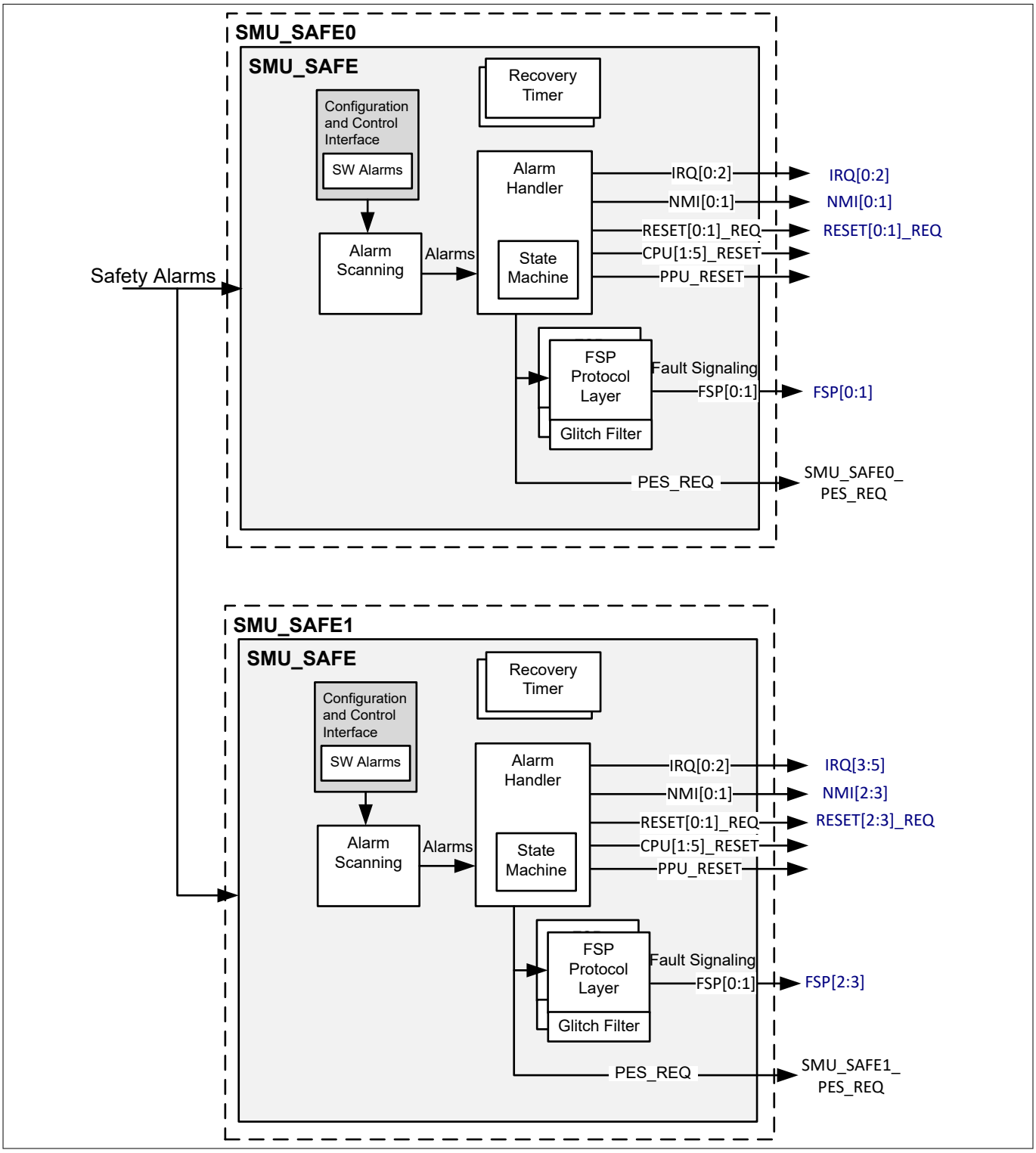


Figure 976 SMU_SAFE0 and SMU_SAFE1 overview

The rest of the sections explain the functionality of the SMU_SAFE instance in detail. Within SMU_SAFE each safety alarm can be individually configured to trigger internal or (and) external reactions. In the Alarm Handling section it is explained how incoming alarms are processed. The Fault Signaling Protocol and its relevant configuration is explained in the section Fault Signaling Protocol (FSP). SMU_SAFE control interface and SMU_SAFE state machine sections explain the usage of software control interface and its impact on SMU_SAFE state machines. An independent access protection mechanism is present to protect the configuration of the SMU_SAFE0 and SMU_SAFE1. SMU_SAFE0 and SMU_SAFE1 are implemented within core domain, that is, provided by core voltage and f_{SPB} clock (refer to the PMS and Clocking chapter of the UM for more details).

Related information

Alarm handling on page 7083
Fault signaling protocol (FSP) on page 7074
SMU_SAFE control interface on page 7064
SMU_SAFE state machine on page 7063
CLOCK on page 5525
Power Management System (PMS) on page 5946

42.3.3.1 SMU_SAFE state machine

The figures below describe the behavior of the SMU_SAFE state machines (SSM0 and SSM1). The SSM0 controls the Fault Signaling Protocol (FSP) Layer 0 and the SSM1 the FSP Layer 1, as explained in the chapter Fault Signaling Protocol (FSP). The SSM0 and SSM1 can be controlled using control interface defined in the chapter SMU_SAFE control interface and alarm handling is also dependent upon the SSM0 and SSM1 as explained in the chapter Alarm Handling. The SMU_SAFE will process input alarm events and therefore trigger alarm reactions according to their configuration if at least one of their state machines (SSM0 or SSM1) is in the RUN state. By default SSM1 is disabled and stays in START state after Power on Reset. It has to be enabled using bitfield SAFEx_AGC.ESSM1.

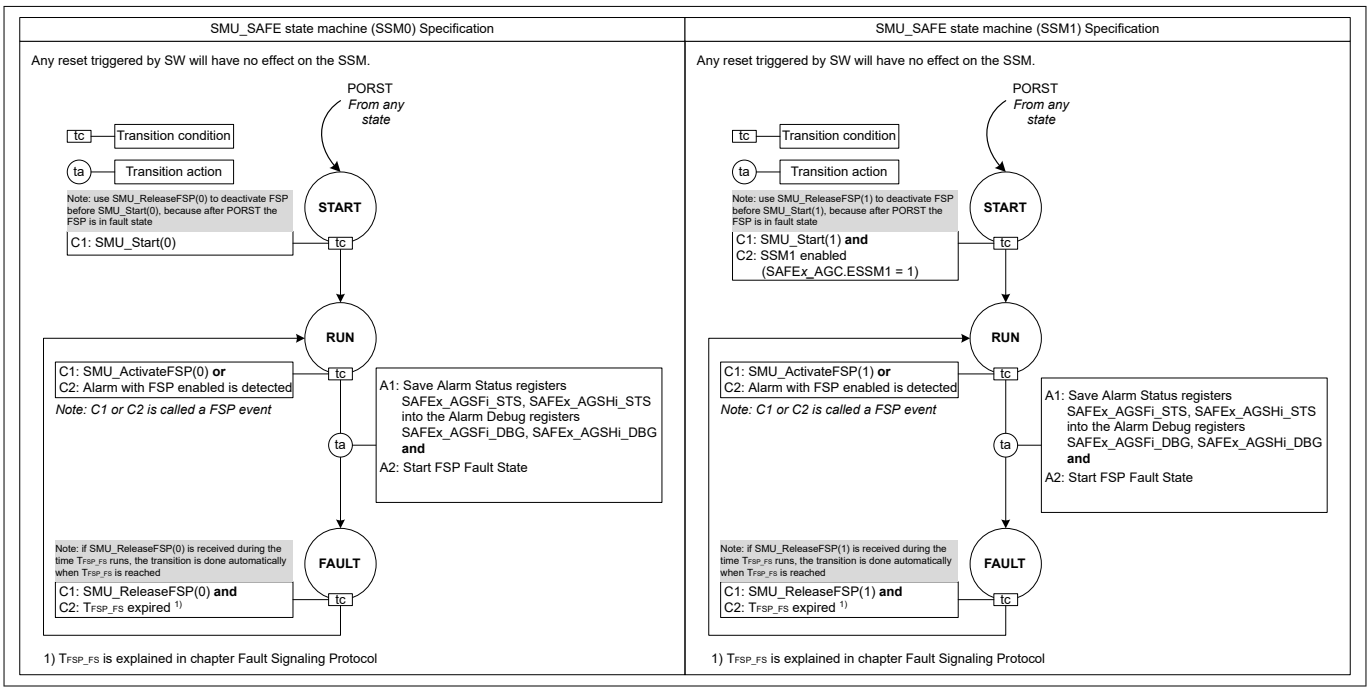


Figure 977 SMU_SAFE state machines (SSM0 and SSM1): Transition conditions and actions

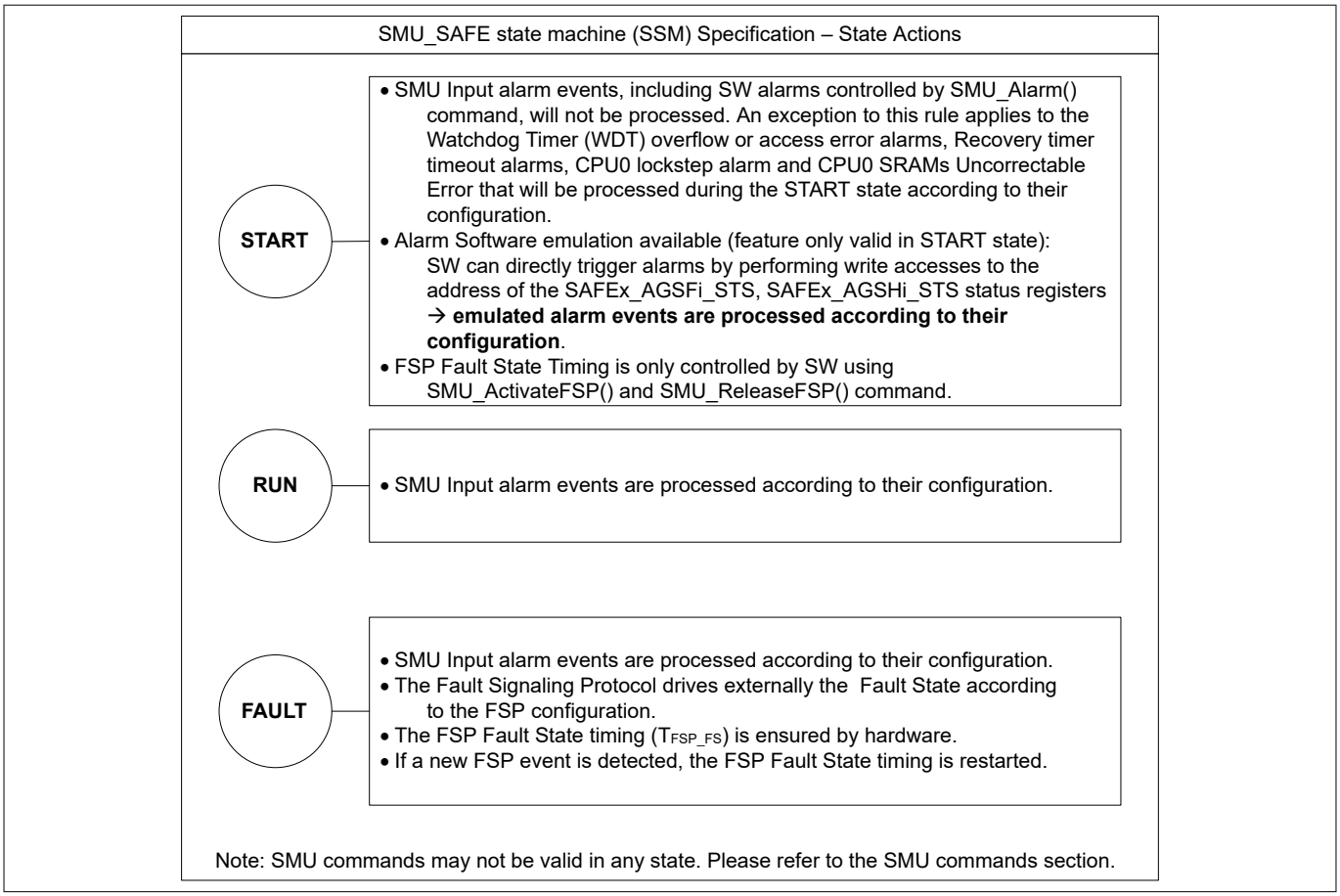


Figure 978 SMU_SAFE state machine: State actions

The current state of the state machines SSM0 and SSM1 can be read out of the SAFEx_DBG register. The SMU_SAFE also implements a Fault Counter SAFEx_AFCNT that counts the number of transitions from the RUN state to the FAULT state. The Fault Counter register is only reset by a warm power-on-reset.

Related information

Fault signaling protocol (FSP) on page 7074
SMU_SAFE control interface on page 7064
Alarm handling on page 7083

42.3.3.2 SMU_SAFE control interface

The core functionality of the SMU_SAFE is introduced through its control interface. The control interface defines how the SMU_SAFE can be controlled by software, as summarized in the table below. The control interface is directly linked to the SMU_SAFE state machines (SSM0 and SSM1) operation described in SMU_SAFE state machine. The control interface is implemented by the SAFEx_CMD register using the CMD and ARG fields. The command completion status is available in the SAFEx_STS register.

Table 1963 SMU_SAFE commands

Command	Description	CMD bit-field
SMU_Start(ARG)	Forces the SSM to go to the RUN state from the START state. For SSM0 : Argument ARG shall be set to 0. For SSM1 : Argument ARG shall be set to 1	0x0
SMU_ActivateFSP(ARG)	Activates the Fault Signaling Protocol. This action is possible in any state of the SSM. For SSM0 : Argument ARG shall be set to 0. For SSM1 : Argument ARG shall be set to 1	0x1
SMU_ReleaseFSP(ARG)	Turns the FSP into the inactive Fault-Free State. In the START state, SMU_ActivateFSP() and SMU_ReleaseFSP() can be called as many times as necessary to perform self tests of every alarm source. For SSM0 : Argument ARG shall be set to 0. For SSM1 : Argument ARG shall be set to 1	0x2
SMU_ActivatePES(ARG)	Triggers the activation of the Port Emergency Stop (PES). The PES is also directly controlled by the SMU_SAFE when entering the FAULT state. Argument ARG shall be set to 0	0x3
SMU_RTStop(ARG)	Stop the recovery Timer. For RT0 : Argument ARG shall be set to 0. For RT1 : Argument ARG shall be set to 1	0x4
SMU_ASCE(ARG)	Alarm Status Clear Enable Command. Software shall execute this command prior to clearing a SAFEx_AGSFi_STS or SAFEx_AGSHi_STS alarm status bit. This command sets the ASCE bit in the SAFEx_STS register. Argument ARG shall be set to 0	0x5
SMU_Alarm(ARG)	Triggers a software based alarm. ARG specifies the alarm index according to the mapping defined in Alarm Mapping. A software alarm has the same properties as a hardware alarm	0x6
SMU_AliveTest(ARG)	Enables the testing of the corresponding SMU_SAFEx Alive Alarm (where x is 0 or 1). Sending this command will trigger SMU_SAFEx Alive Alarm and forwarded to the SMU_STDBY. Argument ARG shall be set to 0x5 to start the test and to 0xA to end the test	0x7

***Note**: If the argument does not comply with the specification, the command is ignored and returns an error code.*

The next table provides the legal conditions for the execution of the commands. The conditions depend on the SMU_SAFE state machines (SSM0 and SSM1) states. Any situation not specified leads to an error code in the SAFEx_STS register.

Table 1964 SMU_SAFE commands and valid conditions

Command	SSM state	Conditions
SMU_Start(ARG)	START	SSM0 : ARG == 0 SSM1 : ARG == 1
SMU_AliveTest(ARG)	START	ARG == 0x5 to start the test and 0xA to end the test
SMU_ActivateFSP(ARG)	Any	SSM0 : ARG == 0 SSM1 : ARG == 1
SMU_ReleaseFSP(ARG)	START	SSM0 : ARG == 0 SSM1 : ARG == 1
SMU_ReleaseFSP(ARG)	FAULT	SSM0 : ARG == 0 & SAFEx_AGC.EFRST0 == 1[1)] SSM1 : ARG == 0 & SAFEx_AGC.EFRST1 == 1
SMU_ActivatePES(ARG)	Any	ARG == 0
SMU_RTStop(ARG)	Any	ARG >= 0 and ARG <= Number of Recovery Timer Instances and Recovery Timer Enabled
SMU_ASCE(ARG)	Any	ARG == 0
SMU_Alarm(ARG)	RUN, FAULT[2)]	ARG >= 0

1) See also Fault Signaling Protocol (FSP).
2) In the START state of the SMU_SAFE, input alarms are not processed with the exceptions specified in Figure 978, therefore software triggered alarms will have no effect.

Related information

SMU_SAFE state machine on page 7063
Fault signaling protocol (FSP) on page 7074
Port emergency stop on page 7071
Recovery timer on page 7072
Alarm status registers on page 7070
Alarm mapping on page 7059

42.3.3.3 Alarm handling

This section specifies the hardware and software alarm processes.

42.3.3.3.1 Alarm configuration

Upon reception of an alarm event the SMU_SAFE decodes the reactions to be performed. The reaction can be classified into an internal reaction and an external reaction. Both the internal and external reactions can be configured for every alarm.

The external reaction is related to the Fault Signaling Protocol. The external reaction can be configured with the following registers:

- SAFEx_AGSFi_FSPj
- SAFEx_AGSHi_FSPj

The internal reaction of the SMU_SAFE under the presence of an alarm can be configured with the following registers:

- SAFEx_AGSFi_CON0, SAFEx_AGSFi_CON1, SAFEx_AGSFi_CON2, SAFEx_AGSFi_CON3
- SAFEx_AGSHi_CON0, SAFEx_AGSHi_CON1, SAFEx_AGSHi_CON2, SAFEx_AGSHi_CON3

The internal reaction is specified by a 4-bit code. As an example for any alarm ALMSFi[z], the 4 bit code is formed as shown in the illustration below:

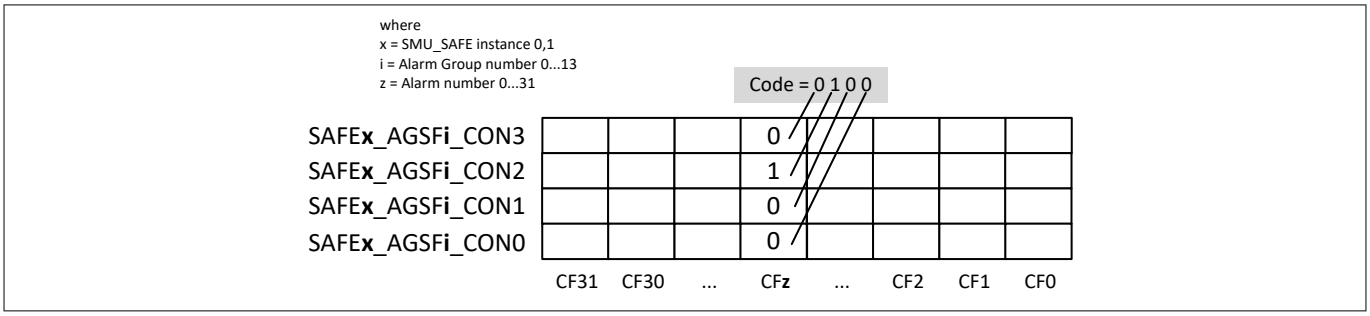


Figure 979 SMU_SAFE alarm configuration code

The table below describes the reaction of SMU_SAFE for each corresponding code:

Table 1965 SMU_SAFE alarm configuration

Code	Name	Reaction
0000	NA	No Reaction. Reset value. Alarm disabled
0001	RSVD	Reserved. No Reaction. Alarm disabled
0010	IGCS0_REQ	Sends an interrupt request to the interrupt system according to the Interrupt Generation Configuration Set 0 from the SAFEx_AGC register
0011	IGCS1_REQ	Sends an interrupt request to the interrupt system according to the Interrupt Generation Configuration Set 1 from the SAFEx_AGC register
0100	IGCS2_REQ	Sends an interrupt request to the interrupt system according to the Interrupt Generation Configuration Set 2 from the SAFEx_AGC register
0101	NMI0_REQ	Sends an NMI0 request to the SCU
0110	RESET0_REQ	Sends a RESET0 request to the SMM. The SMM can be configured to generate an application or system reset

(table continues...)

Table 1965 (continued) SMU_SAFE alarm configuration

Code	Name	Reaction
0111	CPU_RESET0_REQ	Triggers a CPU module reset request using CPU Reset Configuration Set 0 from the SAFEx_AGC register
1000	RSVD	Reserved. No Reaction. Alarm disabled
1001	PPU_RESET	Triggers a PPU module reset request
1010	RSVD	Reserved. No Reaction. Alarm disabled
1011	RSVD	Reserved. No Reaction. Alarm disabled
1100	RSVD	Reserved. No Reaction. Alarm disabled
1101	NMI1_REQ	Sends an NMI1 request to the SCU
1110	RESET1_REQ	Sends a RESET1 request to the SMM. The SMM can be configured to generate an application or system reset
1111	CPU_RESET1_REQ	Triggers a CPU module reset request using CPU Reset Configuration Set 1 from the SAFEx_AGC register

Related information

Fault signaling protocol (FSP) on page 7074

42.3.3.3.2 Alarm operation

Whenever an input alarm event is detected and the SMU_SAFE state machines are in the RUN or FAULT state, the SMU checks for the corresponding internal and the external reactions to be triggered. If an input alarm event is detected and no reaction is specified for the alarm, the corresponding status flag (Refer to the registers SAFEx_AGSFi_STS, SAFEx_AGSHi_STS and section Alarm Status Registers) is set to 1 by hardware although no reaction takes place.

The processing of the incoming alarm events is performed as follows:

- All alarm groups and every alarm are scanned at the same time
- The execution of pending alarms is done concurrently
- The processing of an alarm within SMU_SAFE may take up to 4 f_{SPB} cycles
- If an internal or external reaction is triggered, the corresponding bit in SAFEx_AEX register is set. As long as a bit is set, the corresponding alarm reaction is blocked. This bit needs to be reset by software after fault handling is done in order to enable the alarm reaction upon the next alarm detection
  - If the alarm execution bit in SAFEx_AEX register related to a pending alarm is already set, then the alarm event is ignored but the corresponding status flag in registers SAFEx_AGSFi_STS and SAFEx_AGSHi_STS as well as the corresponding alarm event missed (AEM) bit in SAFEx_AEX register will be set[56)]

[56] The behavior of the AEM bit is different for level alarms. When a reaction to an alarm of this type is triggered by the SMU, the corresponding request status bit xSTS in the SAFEx_AEX register (for example IRQ0STS) is set as expected. However, the corresponding alarm missed event bit xAEM is also set 1 f_{SPB} cycle after the xSTS bit is set for the same alarm event. Therefore, it is recommended that while clearing the xSTS bit the corresponding xAEM bit should also be cleared for the alarm event. Level alarms, as opposed to pulse alarms, remain active until their driving alarm flag is cleared in the source functional block. To find out which alarms need to be cleared in the source functional block, please refer to the alarm tables in the device's safety manual or in the source functional block chapter of the user manual where alarm information is described.

- If the status flag in the registers SAFEx_AGSFi_STS and SAFEx_AGSHi_STS related to an alarm event is already set to 1, the alarm event is ignored
- Whenever an alarm event is processed, the corresponding status bit is set to 1 by hardware in the SAFEx_AGSFi_STS and SAFEx_AGSHi_STS registers. If an internal SMU_SAFE reaction is configured and executed, the alarm reaction counter (ACNT) in the SAFEx_AFCNT register increments

Related information

SMU_SAFE state machine on page 7063
Alarm status registers on page 7070

42.3.3.3.3 Alarm status registers

The status flags in the alarm status registers SAFEx_AGSFi_STS and SAFEx_AGSHi_STS can be set and cleared by software. Setting the bits by software can be used to test alarm reactions and handling. Clearing of an alarm status bit must be performed after an alarm handling reaction has been successfully executed. The software can interact with the alarm status registers through 2 mechanisms:

- By writing a software command 'SMU_ASCE(0)' in the SAFEx_CMD register, as explained in the chapter SMU_SAFE control interface. Then the alarm status flag can be cleared by writing data to SAFEx_AGSFi_STS or SAFEx_AGSHi_STS register. Note that only 32-bit write operations are allowed for clearing an alarm
- By writing data directly to the SAFEx_AGSFi_STS or SAFEx_AGSHi_STS register to test the alarm reactions and handling, also called as alarm emulation

The ability to set and clear the alarm status registers by software depends on the current state of the SMU_SAFE state machine and on the current value of a particular alarm signal. The table below shows examples of the software actions required to set or clear the SAFEx_AGSFi_STS under different conditions (for the state machine and alarm signals), where:

- DATA[j] is a 32-bit data word being written by software and 'j' is the index to an individual word-bit
- In AGSFi[j] 'i' is the index to a particular alarm group and 'j' is the index to an alarm signal within the alarm group

These examples are based on the SAFEx_AGSFi_STS registers but the same principles apply for the SAFEx_AGSHi_STS registers.

Table 1966 Handling of alarm status

SMU_SAFE state machine state	SW action	Effect on SAFEx_AGSFi_STS

START (Both SSM0 and SSM1)	SMU_ASCE(0) command
	and then
	Write DATA at SAFEx_AGSFi_STS address
		If (DATA[j] == 1 AND (Alarm Signal ALMSFi[j][1] == inactive))
		then
		SAFEx_AGSFi_STS.SFj = 0
		else
		“no effect”

START (Both SSM0 and SSM1)	Write DATA at SAFEx_AGSFi_STS Address	If (DATA[j] == 1)
		then
		SAFEx_AGSFi_STS.SFj = 1
		else
		“no effect”

RUN or FAULT (any SSM0 or SSM1)	SMU_ASCE(0) command
	and then
	Write DATA at SAFEx_AGSFi_STS address
		If (DATA[j] == 1 AND Alarm Signal ALMSFi[j] == inactive)
		then
		SAFEx_AGSFi_STS.SFj = 0
		else
		“no effect”

RUN or FAULT (any SSM0 or SSM1)	Write DATA at SAFEx_AGSFi_STS Address	No Effect

1) Watchdog timer (WDT) overflow or access error alarms ALMSF8[0:6] and Recovery timer alarms ALMSF10[18:19] are not clearable in START with default alarm configuration. See Recovery Timer and Watchdog Alarms chapters for more detail about recovery timers and Watchdog alarms default configuration.

To clear individual alarm flags, use only 32-bit writes.

***Note**: After setting an alarm status bit by software in the SAFEx_AGSFi_STS or SAFEx_AGSHi_STS registers, the software shall read back the register value to check if the operation has been completed. Also, after clearing an alarm, the software shall re-check the alarm status bit in case alarms occurred during the time window of the clearing.*

Related information

SMU_SAFE control interface on page 7064
SMU_SAFE state machine on page 7063
Recovery timer on page 7072
Watchdog alarms on page 7072

42.3.3.3.4 Alarm diagnosis registers

The alarm diagnosis registers enable the application to improve the diagnosis of the root cause that led to a malfunction. In that context they may help to implement recovery strategies, if allowed by the application. The SAFEx_AGSFi_DBG and SAFEx_AGSHi_DBG diagnosis registers copy the contents of the corresponding SAFEx_AGSFi_STS and SAFEx_AGSHi_STS registers when:

- The action to be executed by the SMU is a reset when any SMU state machine SSM0 or SSM1 is in the RUN or in the FAULT state
- A condition which switches SMU_SAFE state machine (SSM0 or SSM1) to the FAULT state (RUN -> FAULT, FAULT -> FAULT) takes place, either controlled by the SMU hardware or a software command

The SAFEx_AGSFi_DBG, SAFEx_AGSHi_DBG registers can be cleared by a warm power-on reset (see System Mode Management (SMM) chapter for description about the reset types).

***Note**: After every condition which triggers the SMU to make a copy of the SAFEx_AGSFi_STS and SAFEx_AGSHi_STS registers, the corresponding SAFEx_AGSFi_DBG and SAFEx_AGSHi_DBG diagnosis registers are overwritten. This is also valid if the SMU is already in FAULT state and the FSP is activated again.*

42.3.3.3.5 Port emergency stop

The port emergency stop feature enables forcing a pad into General Purpose Input Mode. The port emergency stop request to the SMU_GCC can be triggered by any of the following situations:

- A SMU_ActivatePES() software command
- An alarm event with SAFEx_AGSFi_FSPj enabled and SAFEx_PES.FSPj enabled
- An alarm event with an internal reaction configured in SAFEx_AGSFi_CONj registers and SAFEx_PES enabled for that reaction. For example for an alarm mapped to index AGSF0[0] in SMU_SAFE0, SAFE0_AGSF0_CON0.CF0 and SAFE0_AGSF0_CON2.CF0 are set to 1 to trigger NMI0_REQ and SAFE0_PES.NMI0 is also set to 1, then upon the occurrence of alarm event mapped to AGSF0[0] the NMI0_REQ as well as port emergency stop request will be triggered

Related information

Emergency stop unit on page 7090
SMU_SAFE control interface on page 7064

42.3.3.3.6 Recovery timer

A recovery timer (RT) is available to enable the monitoring of the response time of an internal reaction such as IRQ[0:2], NMI[0:1], RESET[0:1]_REQ, CPU[1:5]_RESET, and PPU_RESET triggered by an alarm. In SMU_SAFE two independent instances (RT0 and RT1) are available. The recovery timer duration (identical for all instances) is configured in the register SAFEx_RTC. It is possible to enable or disable each instance, however both instances are enabled by default as it is required for the operation of the CPU watchdogs (refer to the chapter Watchdog Alarms). Besides SAFEx_RTC, additional configuration registers (SAFEx_RT0ACA, SAFEx_RT0ACB, SAFEx_RT1ACA and SAFEx_RT1ACB) are available to configure alarms for usage of the recovery timer.

The alarm mapping consists of a pair of parameters {GIDi, ALIDi} (with i = 0..3) present in the registers SAFEx_RT0ACA, SAFEx_RT0ACB, SAFEx_RT1ACA and SAFEx_RT1ACB, where GIDi is a group identifier and ALIDi is the alarm identifier belonging to the group. Four {GIDi, ALIDi} pairs can be configured per recovery timer instance, for example for RT0 instance these four pairs are present in registers SAFEx_RT0ACA and SAFEx_RT0ACB. It is possible to configure the same group identifier several times. If less than four alarms need to be mapped to a recovery timer, the same {GIDi, ALIDi} shall be configured several times. The table below shows the Alarm Groups ALMSF and ALMSH to GIDi mapping.

Table 1967 Alarm group mapping to group index GIDi

Alarm group	GIDi
ALMSF0 - ALMSF13	0-16
ALMSH0 - ALMSH2	17-19

***Note**: The use of the recovery timer only makes sense if the internal reaction is an interrupt or NMI. It is the system integrator's responsibility to configure the SMU_SAFE in the appropriate way.*

A recovery timer is automatically started by hardware, provided that the following conditions are met:

- The recovery timer is enabled
- An alarm event occurs for any of the {GIDi, ALIDi} pairs and the corresponding alarm status bit in the SAFEx_AGSFi_STS or SAFEx_AGSHi_STS register is set by this particular alarm event

Such a situation is called as recovery timer event. Once a recovery timer event has occurred, the recovery timer starts and counts until software stops it with the SMU_RTStop(). If the timer expires, an internal SMU alarm (SMU_SAFE*x* RT*y* Time-out Alarm, where x=0,1 and y=0,1) is issued. During the time the recovery timer is running, any other reaction that requests the recovery timer is ignored. If such event happens, the bit RTME0 (Recovery Timer Missed Event) or RTME1 is set to ‘1’ by hardware in the SAFEx_STS register. The bit RTME0 or RTME1 can only be cleared by software by writing 0 to the corresponding bit. The bit RTS0 (Recovery Timer Status) or RTS1 is set to ‘1’ by hardware in the SAFEx_STS register during the time the recovery timer is running: from the timer activation until a SMU_RTStop() is received or when the timer expires. The bit RTS is cleared by hardware upon reception of SMU_RTStop() or when the timer expires.

If a SMU_RTStop() command is received when the recovery timer is not active, the command returns error state in the bitfield SAFEx_STS.RES.

Related information

Watchdog alarms on page 7072
SMU_SAFE control interface on page 7064

42.3.3.3.7 Watchdog alarms

The watchdog timer (WDT) overflow or access error alarms require a special processing in order to ensure a correct microcontroller behavior if the watchdogs are not serviced by software or firmware. The default reset configuration of these alarms ensures that the microcontroller is reset after a pre-warning phase, where

software can still perform some critical actions. The following properties are implemented as reset values for the WDT overflow or access error alarms and for the recovery timer 0 and 1 in the SMU_SAFE0.

- Every WDT overflow or access error alarms is configured to activate an NMI0
- The Recovery Timer 0 is configured to service WDT overflow or access error alarms for CPU0 WDT, CPU1 WDT, CPU2 WDT and System WDT
- The Recovery Timer 1 is configured to service WDT overflow or access error alarms for CPU3 WDT, CPU4 WDT and CPU5 WDT
- The Recovery Timer 0 and Recovery Timer 1 timeout alarms are configured to issue a reset request (RESET0_REQ) and activate the Fault Signaling Protocol (FSP0)

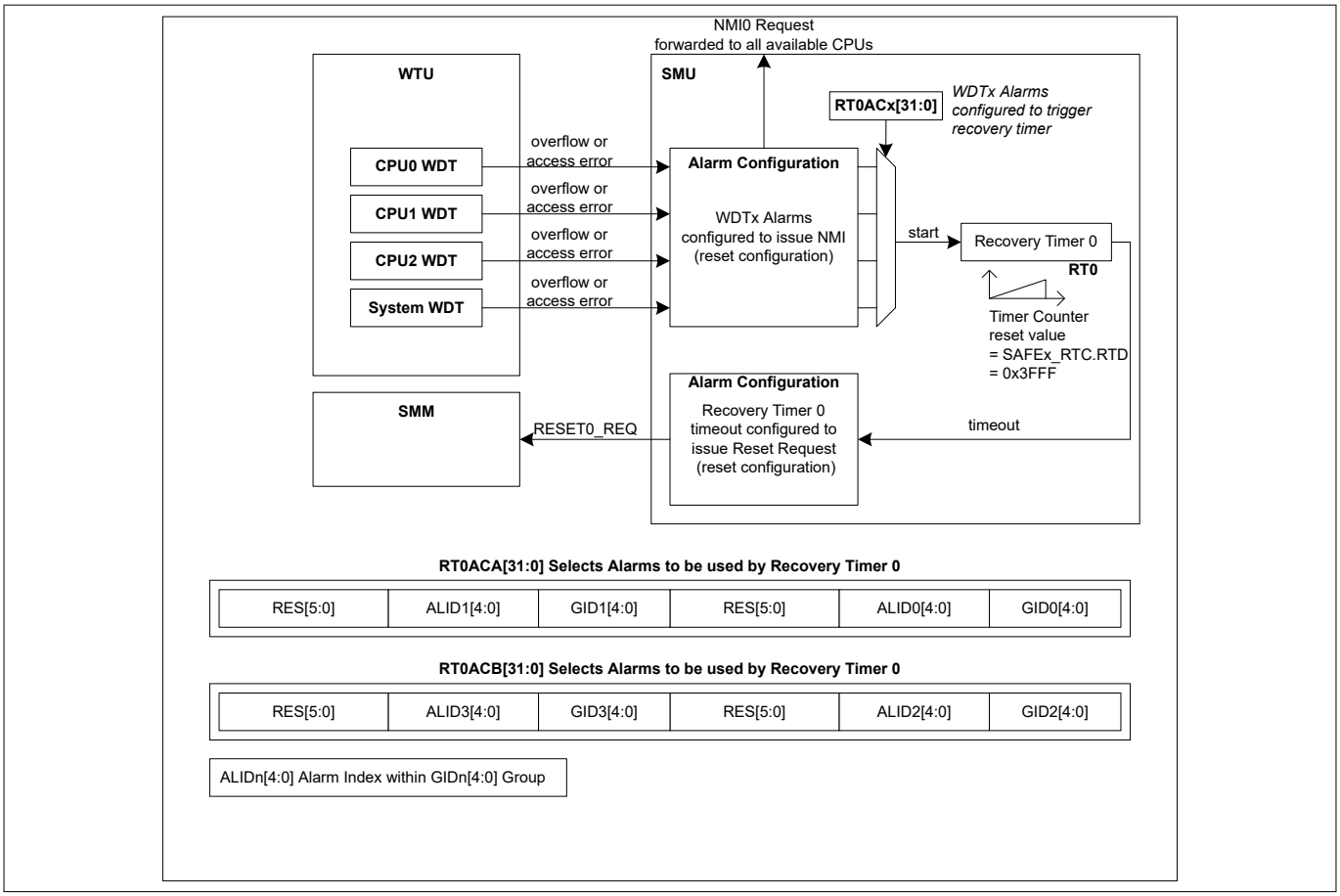


Figure 980 WDT overflow or access error alarm configuration (RT0ACA, RT0ACB)

The figure shows only the example for SAFEx_RT0ACA, SAFEx_RT0ACB and the related four WDT. In the same way the SAFEx_RT1ACA and SAFEx_RT1ACB with the related CPU3 WDT, CPU4 WDT and CPU5 WDT is set up.

Because the WDToverflow or access error detection is also required from the very first instruction executed by a CPU, the SMU is configured to process any WDToverflow or access error alarm during the START state of its internal state machines.

***Note**: If the same behavior is expected for all WDT alarms, it is recommended to use the global WDT timeout alarm that implements a logical OR among all WDT timeout alarms, thus freeing some {GIDi, ALIDi} configuration pairs in SAFEx_RT0ACA, SAFEx_RT0ACB, SAFEx_RT1ACA, and SAFEx_RT1ACB, for other purposes.*

42.3.3.4 Fault signaling protocol (FSP)

The Fault Signaling Protocol enables the microcontroller to report a critical situation to an external safety controller device in order to control the safe state of the safety system.

Each SMU_SAFE has two independent FSP drivers controlled by independent state machines SSM0 and SSM1 respectively. Each FSP layer drives its independent signal FSP[0] and FSP[1] with the exception that FSP Dynamic dual-rail protocol is only supported by FSP Layer 0. Therefore when FSP Layer 0 is used as FSP Dynamic dual-rail protocol FSP Layer 1 and corresponding state machine SSM1 should be disabled (SAFEx_AGC.ESSM1).

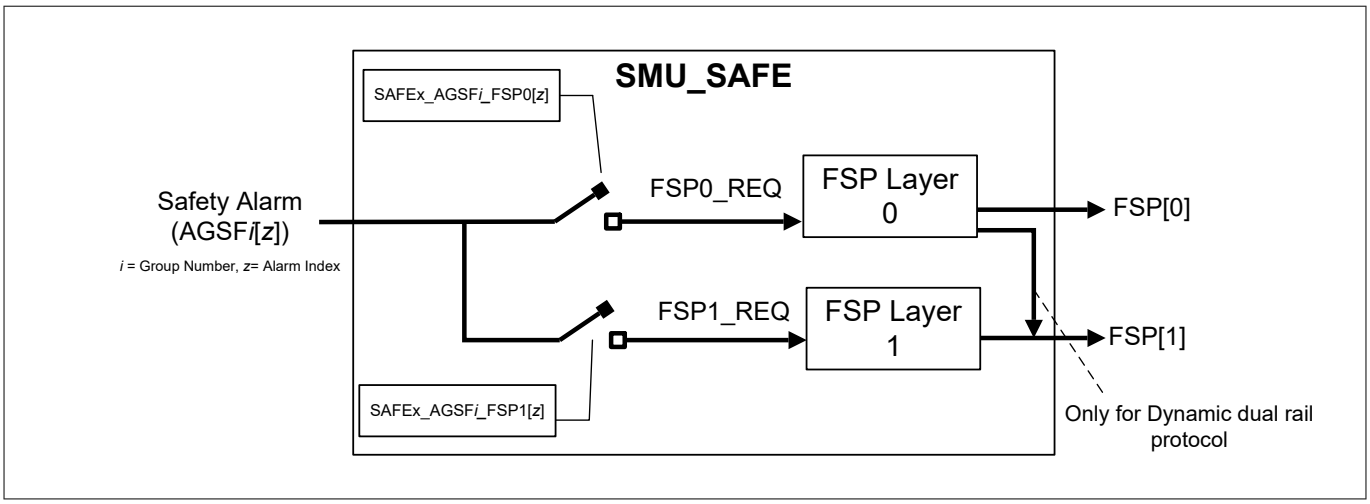


Figure 981 SMU_SAFE fault signaling protocol (FSP)

As Both FSP Layers have similar functionality, with the only exception that Dynamic Dual-rail protocol is only supported by FSP Layer 0, the following sections describe the Fault Signaling Protocol (FSP) by considering FSP Layer 0 as reference. For more information see the chapter in related information.

Related information

Interface to the ports pad on page 7103

42.3.3.4.1 Introduction

The Fault Signaling Protocol is configured with the SAFEx_FSPi_CON configuration register. The FSP status is indicated by the FSP flag in the SAFEx_STS register. The FSP has three states:

- The power-on reset state. After warm power-on reset the SMU is disconnected from the ports and the SMU FSP output assumes the Fault State
- The Fault-Free State. Whenever the Fault-Free State is controlled by a timing, the timing will be called TFSP_FFS and is controlled by the SAFEx_FSPi_CON register
- The Fault State. The timing of the Fault State is controlled by the SAFEx_FSPi_CON register. The minimum active Fault State time is called TFSP_FS

The Fault-Free and Fault State behavior can be configured with the following protocols:

- Bi-stable protocol (default)
- Dynamic dual-rail protocol
- Time-switching protocol

The FSP can be controlled by:

- Software using the SMU_ActivateFSP() and SMU_ReleaseFSP() commands using the SAFEx_CMD register
- Hardware based on the SAFEx_AGSFi_FSPj and SAFEx_AGSHi_FSPj configuration registers as shown in Figure 981

The figure below specifies the intermediate clocks to generate the TFSP_FFS and TFSP_FS timings.

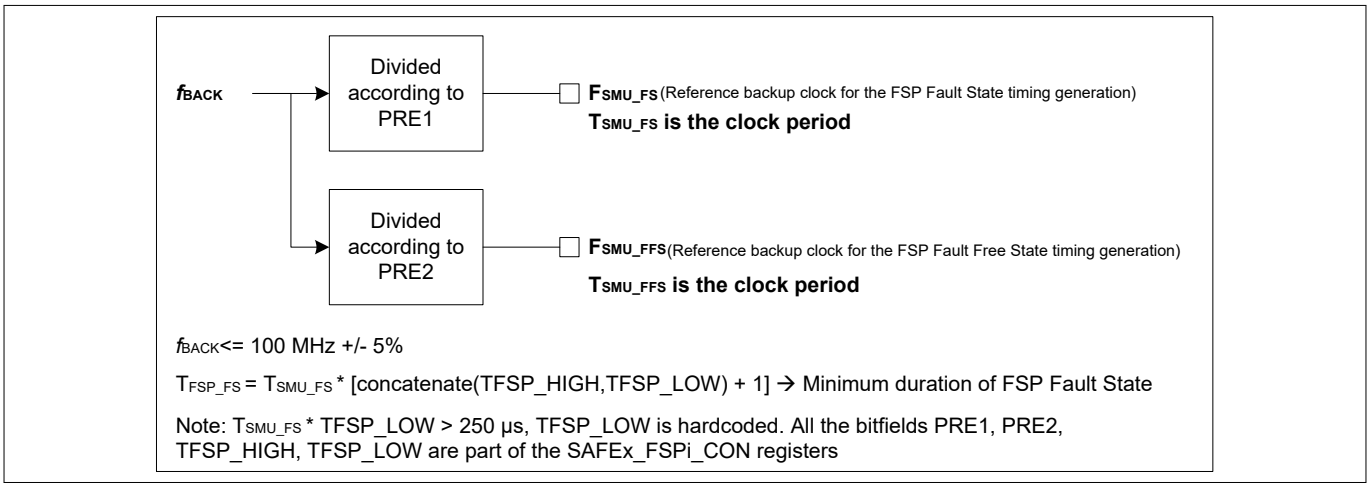


Figure 982 Reference clocks for FSP timings

42.3.3.4.2 Bistable fault signaling protocol

Operation

- During power-on-reset FSP[0] = 0 (Fault State)
- After power-on reset FSP[0] stays in the Fault State
- FSP[0] must be set to the Fault-Free State (FSP[0] = 1) by software (SMU_ReleaseFSP())
- Upon detection of an alarm event configured to activate the FSP, FSP[0] goes to the Fault State and remains in this state until a SMU_ReleaseFSP() command is received and TFSP_FS is satisfied
- While in the Fault State, if a new alarm event configured to activate the FSP is received and the TFSP_FS has not yet been reached, the TFSP_FS timing will be restarted
- While in the Fault State, if a new alarm event configured to activate the FSP is received and the TFSP_FS has already been reached, the TFSP_FS timing will be started

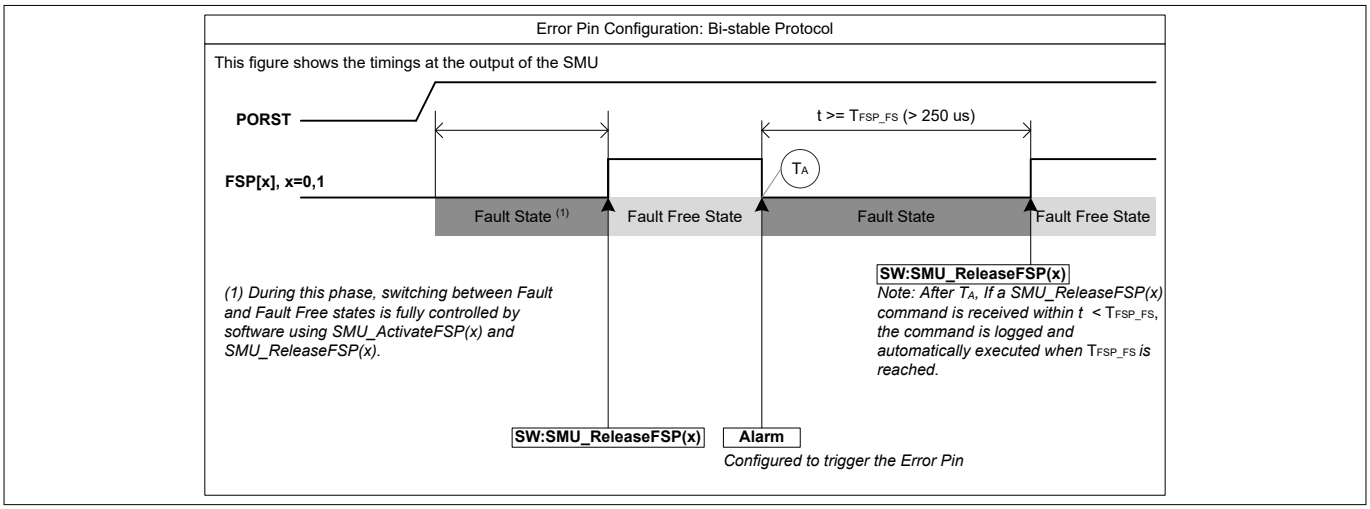


Figure 983 Bistable fault signaling protocol

42.3.3.4.3 Dynamic dual-rail protocol

Operation

Dual-rail encoding is an alternative method for encoding bits. Dual-rail codes use two signals to define a logical state.

- During power-on-reset FSP[1:0] = 00b (Fault State)
- After power-on-reset FSP[1:0] stays in the Fault State
- FSP[1:0] must be set to the Fault-Free State per software(SMU_ReleaseFSP())
- The Fault-Free State is defined by FSP[1:0] oscillating between 2'b01 and 2'b10 with a defined frequency configured by the SAFEx_FSPi_CON register and with a duty cycle of 50% (see figure below)
- Upon detection of an alarm event configured to activate the FSP, FSP[1:0] goes immediately to the Fault State and remains in this state until a SMU_ReleaseFSP() command is received and TFSP_FS is satisfied

Table 1968 Dual-rail coding

Code	Description
{0,1}	Fault-Free State
{1,0}	Fault-Free State
{0,0}	Fault State
{1,1}	Fault State (encoding not used)

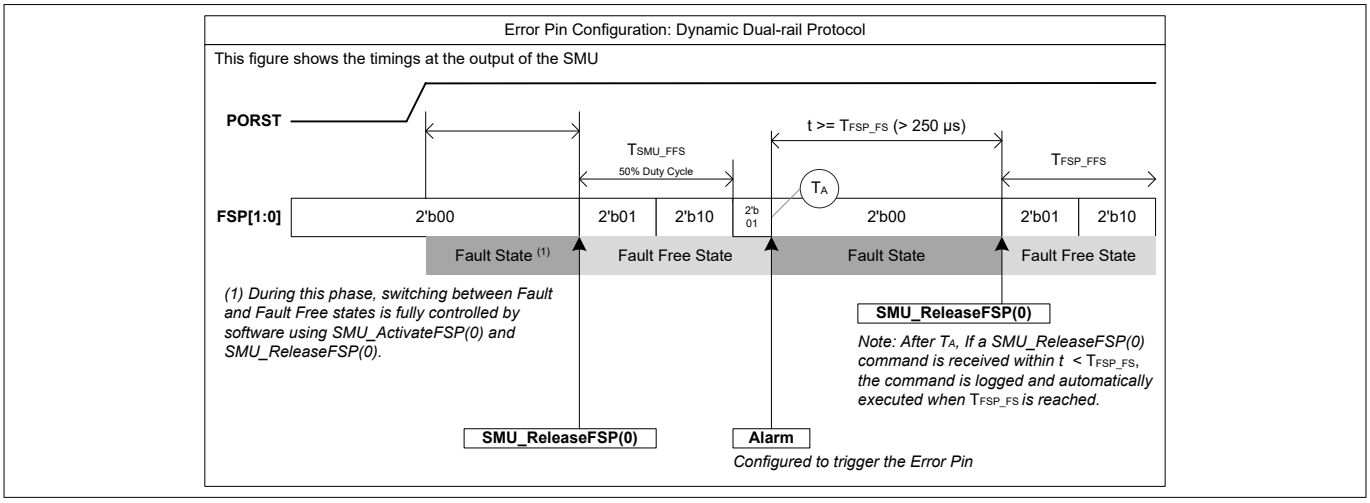


Figure 984 Dynamic dual-rail fault signaling protocol

42.3.3.4.4 Time switching protocol

Operation

FSP[0] is toggled between logic level 0 and logic level 1 with a defined frequency. This frequency modulation protocol is violated when the SMU_SAFE enters the FAULT State.

- During Power-on Reset FSP[0] = 0 (Fault State)
- After Power-on Reset FSP[0] stays in the Fault State
- FSP[0] must be set to the Fault-Free State per software (SMU_ReleaseFSP())
- In the Fault-Free State, FSP[0] oscillates between logic level 0 and logic level 1 with the frequency configured by the SAFEx_FSPi_CON register (see figure below)
- Upon detection of an alarm event configured to activate the FSP, FSP[0] goes immediately to the Fault State and remains in this state until a SMU_ReleaseFSP() command is received and TFSP_FS is satisfied

42 Safety and security alarm Management Unit (SMU)

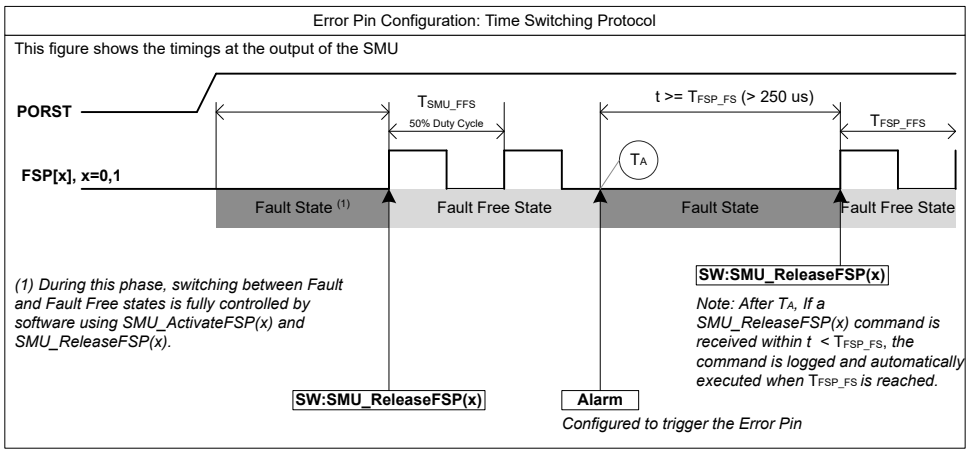


Figure 985 Time switching protocol

42.3.3.4.5 FSP control during Fault State

When the FSP is triggered, the SMU_SAFE state machine automatically switches to the FAULT state. During this time it is also possible for the software to potentially analyze the root cause (provided that the microcontroller is still operational) and decide the severity of the error. As the FSP is active for at least TFSP_FS, it is ensured that the safe state of the system is entered through external safety mechanisms independent from the microcontroller (besides FSP itself). During the time FSP is in the Fault State (TFSP_FS), the software may have concluded that the fault is uncritical and decides to issue a SMU_ReleaseFSP() command, notifying the SMU_SAFE that it can return to the RUN state. Note that the run-time of the software error handler is not directly correlated with the TFSP_FS duration and in practice could be much shorter.

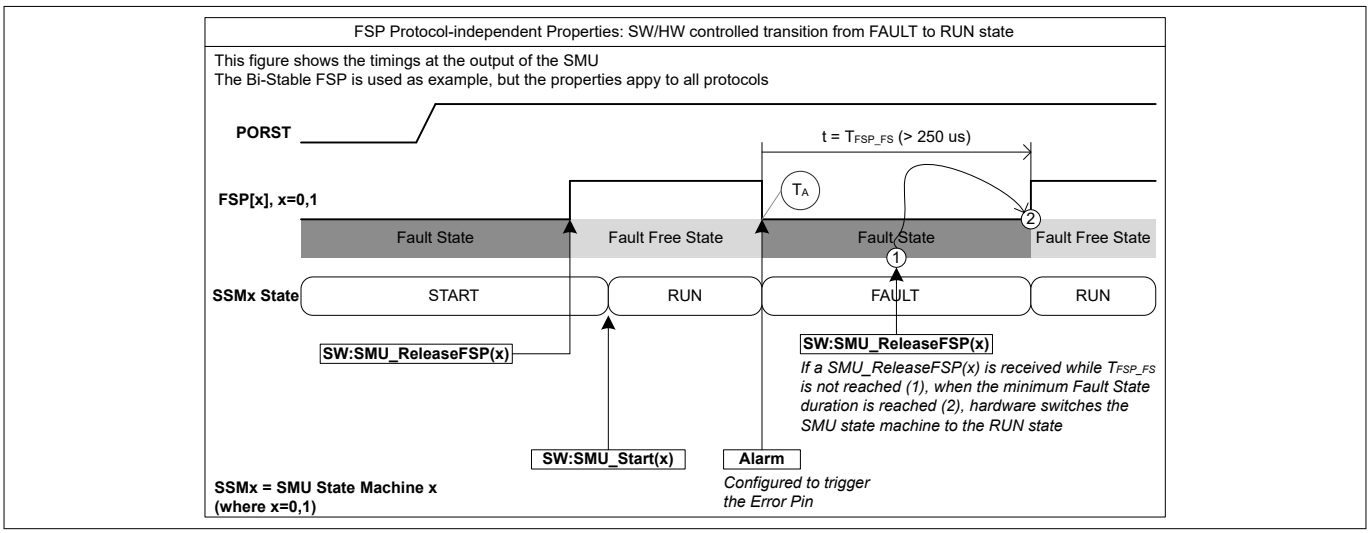


Figure 986 FSP: Fault State to Fault-Free State transition

This feature is per default disabled and can be configured with the EFRST (Enable FAULT to RUN State Transition) field in the SAFEx_AGC register.

42.3.3.4.6 FSP control during START state

The figure below shows a typical use case where the FSP transitions between the Fault State and Fault-Free State are controlled by software using the SMU_ReleaseFSP(0) and SMU_ActivateFSP(0) commands during the SSM0 START State.

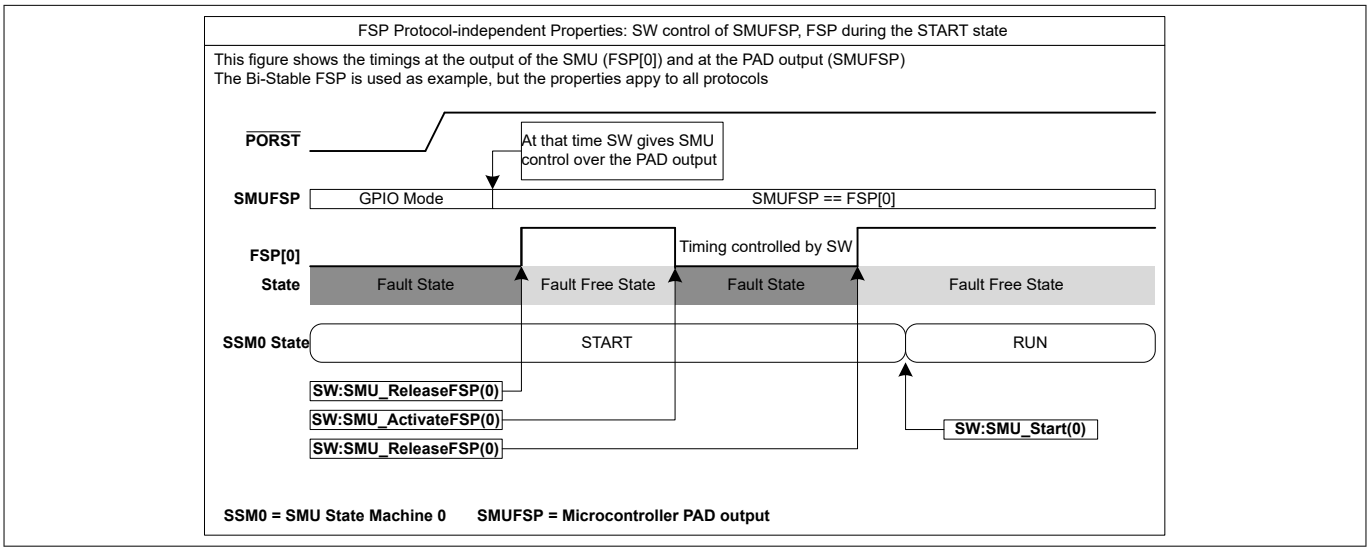


Figure 987 Software control of FSP during SMU_SAFE START state

Conditions of use

Software shall ensure that the FSP is in the Fault-Free State before entering the RUN state with the SMU_Start() command.

42.3.3.5 Access protection

The SMU_SAFE registers are accessed by SPB bus and are access protected based on PROT and BUS Access Protection Unit (APU) mechanism. More explanation for the PROT and APU based access protection can be found in the FPI chapter (section Access Protection) in the user manual.

Each SMU_SAFE instance has a Safe Endinit PROT Register (SAFEx_PROTSSE) that controls access to the Bus Access Protection Unit (APU) registers (SAFEx_ACCEN_WRA, SAFEx_ACCEN_WRB, SAFEx_ACCEN_RDA, SAFEx_ACCEN_RDB, SAFEx_ACCEN_VM, SAFEx_ACCEN_PRS). Only the PROT owner defined SAFEx_PROTSSE can configure the Bus APU registers. Bus APU registers define the access rule for the SMU_SAFE configuration and status register. The SMU_SAFE configuration and status register are read- or write- access protected as defined by BUS APU registers of protection group PSAFEx in the register overview.

For certain configuration registers, the write access is controlled by the PROT owner defined in SMU_SAFE Endinit PROT Register (SAFEx_PROTSE). These registers are marked with SxE in the Register Overview table.

Besides PROT and APU based access protection, the SMU_SAFE has an internal locking mechanism. This locking mechanism is controlled by SAFEx_KEYS register and is explained in the Safety Manual chapter SM[HW]:SMU:LOCK. Please always refer to the device specific safety manual to find out which safety mechanisms of this functional block need to be considered for safety relevant applications.

Related information

Registers overview - SMU (ascending offset address) on page 7109

42.3.3.6 Reset types

The System Mode Management (SMM) implements different reset types that affect the SMU_SAFE. The reset types that are relevant for SMU_SAFE are:

- Warm power-on reset
- Debug reset
- Application reset

The table below specifies the scope of each reset type to the SMU_SAFE control configuration and logic.

Table 1969 Effect of reset types to SMU_SAFE functionality

SMU function	Application reset	Debug reset	Warm power-on reset
SMU_SAFE FSP function	Not affected	Not affected	Reset
SMU_SAFE state machine function	Not affected	Not affected	Reset
SMU_SAFE debug function	Not affected	Reset	Reset
SMU_SAFE alarm diagnosis registers SAFEx_AGSFi_DBG, SAFEx_AGSHi_DBG	Not affected	Not affected	Reset
Port control register SAFEx_PCTL	Not affected	Not affected	Reset
SMU_SAFE alive monitor 1)	Reset	Not affected	Reset
SMU_SAFE glitch filter 2)	Reset	Not affected	Reset
SMU_SAFE other functions	Reset	Not affected	Reset

1) The description of Alive Monitor is present in Safety Manual SM[HW]:SMU:ALIVE_MONITOR.
2) The description of Glitch Filter is present in the chapter Interface to the Ports Pad.

Related information

System Mode Management (SMM) on page 6315
Fault signaling protocol (FSP) on page 7074
SMU_SAFE state machine on page 7063
OCDS trigger bus (OTGB) interface on page 7220
Interface to the ports pad on page 7103

42.3.4 SMU_CS

SMU_CS is a security alarm handler. Within SMU_CS each security alarm can be individually configured to trigger internal reactions. In the Alarm Handling section it is explained how incoming alarms are processed while SMU_CS control interface and SMU_CS state machine explain the usage of the software control interface and its impact on the SMU_CS state machine. An independent access protection mechanism is present to protect the configuration of SMU_CS. The SMU_CS is implemented within core domain, that is, provided by core voltage and f_{SPB} clock (refer to the PMS and Clocking chapter of the UM for more details).

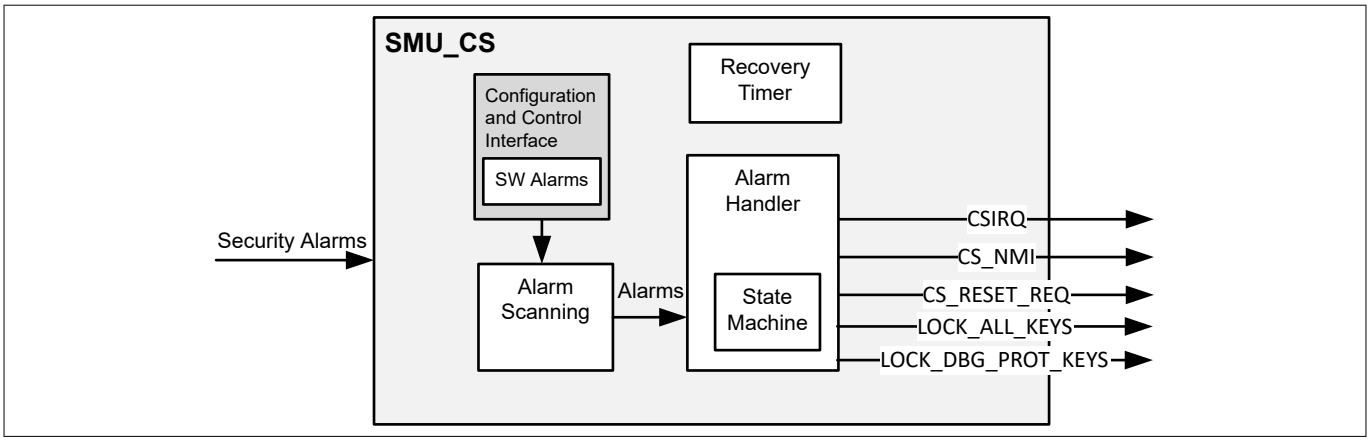


Figure 988 SMU_CS overview

Related information

Alarm handling on page 7083
SMU_CS control interface on page 7082
SMU_CS state machine on page 7081
CLOCK on page 5525
Power Management System (PMS) on page 5946

42.3.4.1 SMU_CS state machine

The figures below describes the behavior of the SMU_CS state machine (SSM0). The SSM0 can be controlled using control interface defined in the chapter SMU_CS control interface and alarm handling is also dependent upon the SSM0 as explained in the chapter Alarm Handling. The current state of the SMU_CS state machine can be read out from the CS_DBG register.

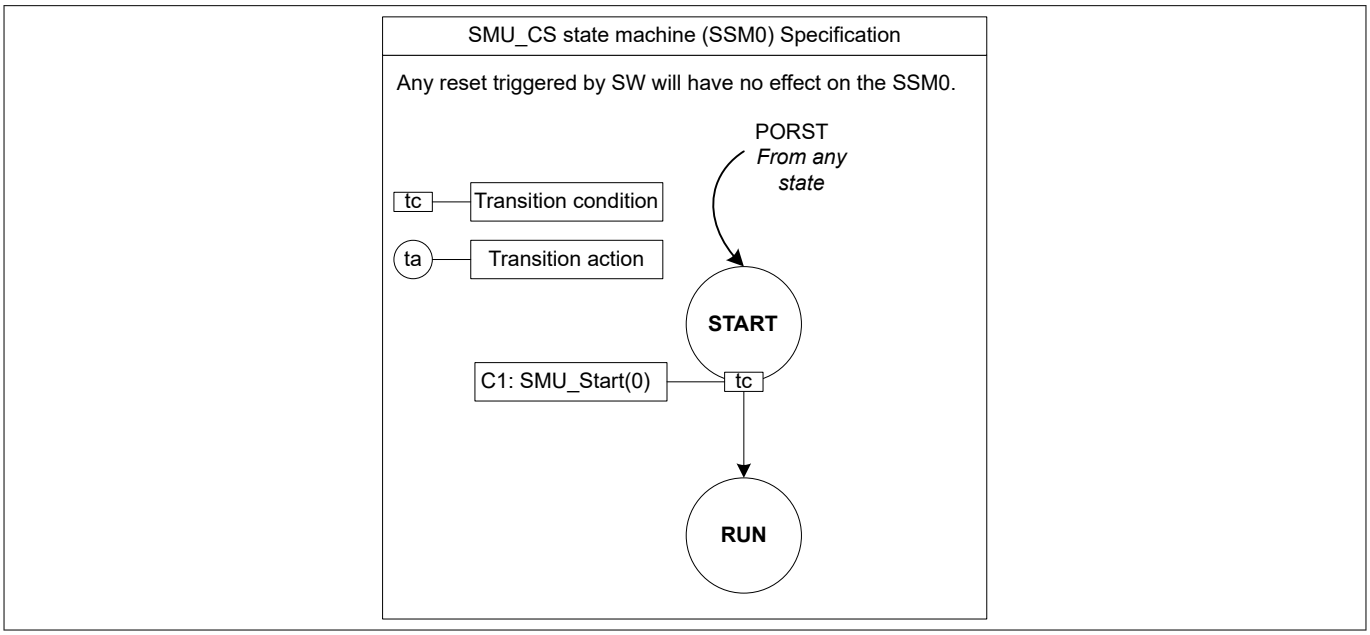


Figure 989 SMU_CS state machine (SSM0): Transition conditions and actions

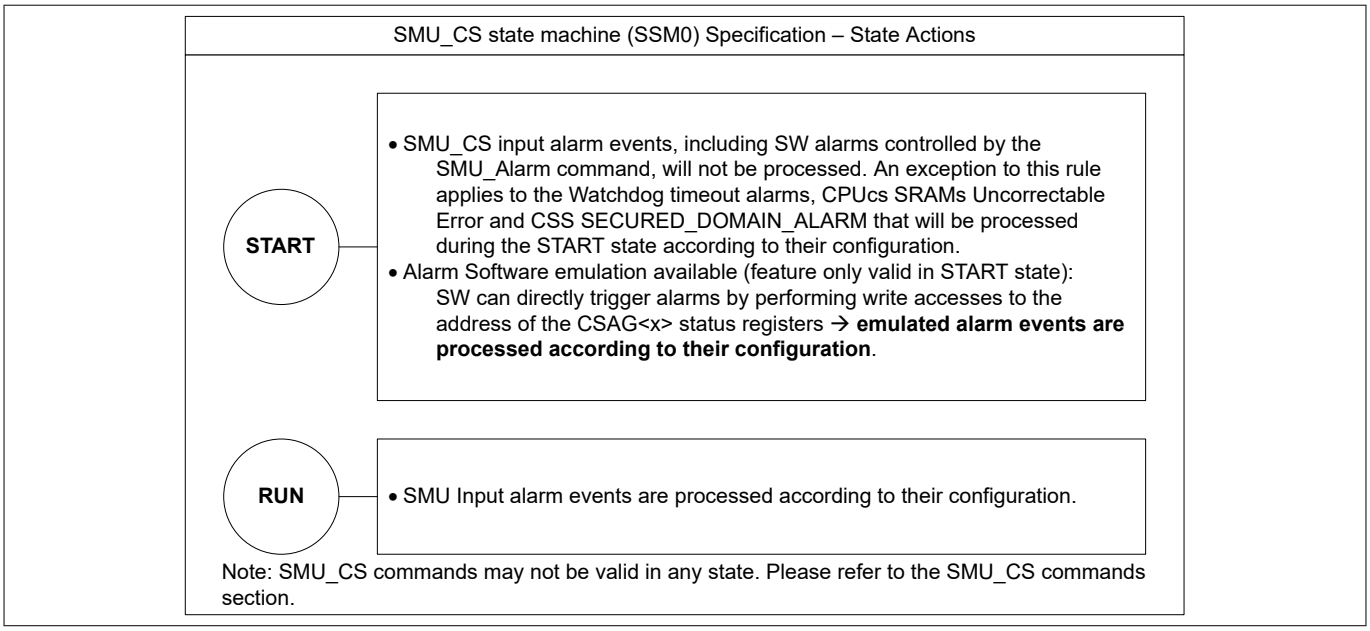


Figure 990 SMU_CS state machine: State actions

Related information

SMU_CS control interface on page 7082
Alarm handling on page 7083

42.3.4.2 SMU_CS control interface

The core functionality of the SMU_CS is controlled through its control interface. The control interface defines how the SMU_CS can be controlled by software, as summarized in the following table. The control interface is directly linked to the SMU_CS state machine (SSM0) operation described in SMU_CS state machine. The control interface is implemented by the CS_CMD register using the CMD and ARG fields. The command completion status is available in the CS_STS register.

Table 1970 SMU_CS commands

Command	Description	CMD bit-field
SMU_Start(ARG)	Forces the SSM to go to the RUN state from the START state. For SSM0 : Argument ARG shall be set to 0	0x0
Reserved	Reserved	0x1
Reserved	Reserved	0x2
Reserved	Reserved	0x3
SMU_RTStop(ARG)	Stop the recovery Timer. For RT0 : Argument ARG shall be set to 0	0x4
SMU_ASCE(ARG)	Alarm Status Clear Enable Command. Software shall execute this command prior to clear a CS_AGCSi_STS alarm status bit. This command sets the ASCE bit in the CS_STS register. Argument ARG shall be set to 0	0x5

(table continues...)

Table 1970 (continued) SMU_CS commands

Command	Description	CMD bit-field
SMU_Alarm(ARG)	Triggers a software based alarm. ARG specifies the alarm index according to the mapping defined in Alarm Mapping. A software alarm has the same properties as a hardware alarm	0x6
SMU_AliveTest(ARG)	Enables the testing of the SMU_CS Alive Alarm. Sending this command will trigger SMU_CS Alive Alarm and forwarded to the SMU_STDBY. Argument ARG shall be set to 0x5 to start the test and to 0xA to end the test	0x7

***Note**: If the argument does not comply with the specification, the command is ignored and returns an error code.*

The next table provides the legal conditions for the execution of the commands. The conditions depend on the SMU_CS state machine (SSM0) states. Any situation not specified leads to an error code in the CS_STS register.

Table 1971 SMU_CS commands and valid conditions

Command	SSM state	Other conditions
SMU_Start(ARG)	START	SSM0 : ARG == 0
SMU_AliveTest(ARG)	START	ARG == 0x5 to start the test and 0xA to end the test
SMU_RTStop(ARG)	Any	ARG >= 0 and ARG <= Number of Recovery Timer Instances and Recovery Timer Enabled
SMU_ASCE(ARG)	Any	ARG == 0
SMU_Alarm(ARG)	RUN[1)]	ARG >= 0

1) In the START state of the SMU_CS input alarms are not processed, therefore software triggered alarms will have no effect.

Related information

SMU_CS state machine on page 7081
Recovery timer on page 7087
Alarm status registers on page 7085
Alarm mapping on page 7059

42.3.4.3 Alarm handling

This section specifies the hardware and software alarm processes.

42.3.4.3.1 Alarm configuration

Upon reception of an alarm event the SMU_CS decodes the reactions to be performed. These reactions can be configured for every alarm through the following registers :

- CS_AGCSi_CON0, CS_AGCSi_CON1 and CS_AGCSi_CON2
- CS_AGSHi_CON0, CS_AGSHi_CON1 and CS_AGSHi_CON2

The reaction is specified by a 3-bit code. As an example for any alarm ALMCSi[z], the 3 bit code is formed as shown in the illustration below:

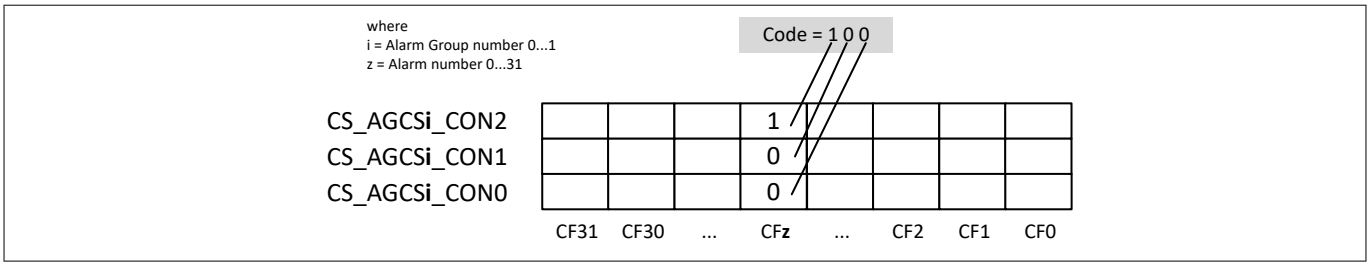


Figure 991 SMU_SAFE alarm configuration code

Table 1972 SMU_CS alarm configuration

Code	Name	Reaction
000	NA	No Reaction. Reset value. Alarm disabled
001	LOCK_ALL_KEYS & CSIRQ	Sends a request to the CSS for locking all keys and an interrupt request to IR
010	LOCK_ALL_KEYS & CS_NMI	Sends a request to the CSS for locking all keys and a NMI request to the SCU
011	LOCK_DBG_PROT_KEYS	Sends a request to the CSS for locking Debug Protected Keys
100	LOCK_DBG_PROT_KEYS & CSIRQ	Sends a request to the CSS for locking Debug Protected Keys and an interrupt request to IR
101	CS_NMI	Sends an NMI request to the SCU
110	CS_RESET_REQ	Sends a RESET request to the SMM. The SMM can be configured to generate an application or system reset
111	CSIRQ	Sends an interrupt request to IR

42.3.4.3.2 Alarm operation

Whenever an input alarm event is detected and the SMU_CS state machine is in the RUN state, the SMU checks for the corresponding reactions to be triggered. If an input alarm event is detected and no reaction is specified for the alarm, the corresponding status flag (Refer to the registers CS_AGCSi_STS, CS_AGSHi_STS and section Alarm Status Registers) is set to 1 by hardware although no reaction takes place.

The processing of the incoming alarm events is performed as follows:

- All alarm groups and every alarm are scanned at the same time
- The execution of pending alarms is done concurrently
- The processing of an alarm within SMU_CS may take up to four f_{SPB} cycles
- If an reaction is triggered, the corresponding bit in CS_AEX register is set. As long as a bit is set, the corresponding alarm reaction is blocked. This bit needs to be reset by software after fault handling is done in order to enable the alarm reaction upon the next alarm detection
  - If the alarm execution bit in CS_AEX register related to a pending alarm is already set, the alarm event is ignored but the corresponding status flag in registers CS_AGCSi_STS and CS_AGSHi_STS as well as the corresponding alarm event missed (AEM) bit in CS_AEX register will be set[57)]

57 The behavior of the AEM bit is different for level alarms. When a reaction to an alarm of this type is triggered by the SMU, the corresponding request status bit xSTS in the CS_AEX register (for example IRQ0STS) is set as expected. However, the corresponding alarm missed event bit xAEM is also set 1 f_{SPB} cycle after the xSTS bit is set for the same alarm event. Therefore, it is recommended that while clearing the xSTS bit the corresponding xAEM bit should also be cleared for the alarm event. Level alarms, as opposed to pulse alarms, remain active until their driving alarm flag is cleared in the source functional block. To find

- If the status flag in registers CS_AGCSi_STS and CS_AGSHi_STS related to an alarm event is already set to 1, the alarm event is ignored
- Whenever an alarm event is processed, the corresponding status bit is set to 1 by hardware in the CS_AGCSi_STS and CS_AGSHi_STS register. If an internal SMU_CS reaction is configured and executed, the alarm reaction counter (ACNT) in the CS_AFCNT register increments

Related information

SMU_CS state machine on page 7081

Alarm status registers on page 7085

42.3.4.4.3 Alarm status registers

The status flags in the alarm status registers CS_AGCSi_STS and CS_AGSHi_STS can be set and cleared by software. Setting the bits by software can be used to test alarm reactions and handling. Clearing of an alarm status bit must be performed after an alarm handling reaction has been successfully executed. The software can interact with the alarm status registers through 2 mechanisms:

- By writing a software command 'SMU_ASCE(0)' in the CS_CMD register, as explained in the chapter SMU_CS control interface. Then the alarm status flag can be cleared by writing data to CS_AGCSi_STS or CS_AGSHi_STS register. Note that only 32-bit write operations are allowed to clear an alarm
- By writing data directly to the CS_AGCSi_STS or CS_AGSHi_STS register to test the alarm reactions and handling, also called alarm emulation

The ability to set and clear the alarm status registers by software depends on the current state of the SMU_CS state machine and on the current value of a particular alarm signal. The table below shows examples of the software actions required to set or clear the CS_AGCSi_STS under different conditions (for the state machine and alarm signals), where:

- DATA[j] is a 32-bit data word being written by software and 'j' is the index to an individual word-bit
- In AGCSi[j] 'i' is the index to a particular alarm group and 'j' is the index to an alarm signal within the alarm group

These examples are based on the CS_AGCSi_STS registers but the same principles apply for the CS_AGSHi_STS registers.

Table 1973 Handling of alarm status

SMU_CS state machine state	SW action	Effect on CS_AGCSi_STS

START	SMU_ASCE(0) command
	and then
	Write DATA at CS_AGCSi_STS address
		If (DATA[j] == 1 AND (Alarm Signal ALMCSi[j][1] == inactive))
		then
		CS_AGCSi_STS.SF[j] = 0
		else
		“no effect”

(table continues...)

out which alarms need to be cleared in the source functional block, please refer to the alarm tables in the device's safety manual or in the source functional block chapter of the user manual where alarm information is described.

Table 1973 (continued) Handling of alarm status

SMU_CS state machine state	SW action	Effect on CS_AGCSi_STS

START	Write DATA at CS_AGCSi_STS address	If (DATA[j] == 1)
		then
		CS_AGCSi_STS.SF[j] = 1
		else
		“no effect”

RUN	SMU_ASCE(0) command
	Write DATA at CS_AGCSi_STS address
		If (DATA[j] == 1 AND Alarm Signal ALMCSi[j] == inactive)
		then
		CS_AGCSi_STS.SF[j] = 0
		else
		“no effect”

RUN	Write DATA at CS_AGCSi_STS address	No effect

1) Watchdog timer (WDT) overflow or access error alarm ALMSH0[31] and Recovery timer alarm ALMCS1[18] are not clearable in START with default alarm configuration. See Recovery Timer and Watchdog alarms chapters for more details about Recovery timers' and Watchdog alarms' default configurations.

To clear individual alarm flags, use only 32bit writes.

***Note**: After setting an alarm status bit by software in the CS_AGCSi_STS or CS_AGSHi_STS registers, the software shall read back the register value to check if the operation has been completed. Also, after clearing an alarm, the software shall re-check the alarm status bit to check if alarms occurred during the time window of the clearing.*

Related information

SMU_CS control interface on page 7082
SMU_CS state machine on page 7081
Recovery timer on page 7087
Watchdog alarms on page 7087

42.3.4.3.4 Alarm diagnosis registers

The alarm diagnosis registers enable the application to improve the diagnosis of the root cause that led to a malfunction. In that context they may help to implement recovery strategies, if allowed by the application. The CS_AGCSi_DBG and CS_AGSHi_DBG diagnosis registers copy the contents of the corresponding CS_AGCSi_STS and CS_AGSHi_STS registers when:

- The action to be executed by the SMU_CS is a reset when any SMU_CS state machine SSM0 is in the RUN state

The CS_AGCSi_DBG and CS_AGSHi_DBG registers can be cleared by a warm power-on reset (see System Mode Management (SMM) chapter for description about the reset types).

42.3.4.3.5 Recovery timer

A recovery timer (RT) is available to enable the monitoring of the response time of an internal reaction such as CSIRQ, CS_NMI, CS_RESET_REQ, LOCK_ALL_KEYS, and LOCK_DBG_PROT_KEYS triggered by an alarm. In SMU_CS one instance (RT0) is available. The recovery timer duration is configured in the register CS_RTC. It is possible to enable or disable the RT0, however RT0 is enabled by default as it is required for the operation of the CPUcs watchdog. Besides CS_RTC, additional configuration registers (CS_RT0ACA and CS_RT0ACB) are available to configure alarms for usage of the recovery timer.

The alarm mapping consists of a pair of parameters {GIDi, ALIDi} (with i = 0..3), where GIDi is a group identifier and ALIDi is the alarm identifier belonging to the group. Four {GIDi, ALIDi} pairs can be configured per recovery timer instance. It is possible to configure the same group identifier several times. If less than four alarms need to be mapped to a recovery timer, the same {GIDi, ALIDi} shall be configured several times. The table below shows the Alarm Groups ALMCS and ALMSH to GIDi mapping.

Table 1974 Alarm group mapping to group index GIDi

Alarm group	GIDi
ALMCS0 - ALMCS1	0-1
ALMSH0 - ALMSH2	2-4

***Note**: The use of the recovery timer only makes sense if the internal reaction is an interrupt or NMI. It is the system integrator's responsibility to configure the SMU_CS in the appropriate way.*

A recovery timer is automatically started by hardware, provided that the following conditions are met:

- The recovery timer is enabled
- An alarm event occurs for any of the {GIDi, ALIDi} pairs and the corresponding alarm status bit in the CS_AGCSi_STS or CS_AGSHi_STS register is set by this particular alarm event

Such a situation is called as recovery timer event. Once a recovery timer event has occurred, the recovery timer starts and counts until software stops it with the SMU_RTStop(). If the timer expires, an internal SMU_CS alarm (SMU_CS RT0 Time-out Alarm) is issued. During the time the recovery timer is running, any other reaction that requests the recovery timer is ignored. If such event happens, the bit RTME0 (Recovery Timer Missed Event) is set to '1' by hardware in the CS_STS register. The bit RTME0 can only be cleared by software by writing 0 to the corresponding bit. The bit RTS0 (Recovery Timer Status) is set to '1' by hardware in the CS_STS register during the time the recovery timer is running: from the timer activation until a SMU_RTStop() is received or when the timer expires. The bit RTS0 is cleared by hardware upon reception of SMU_RTStop() or when the timer expires.

If a SMU_RTStop() command is received when the recovery timer is not active, the command returns error state in the bitfield CS_STS.RES.

Related information

SMU_CS control interface on page 7082

42.3.4.3.6 Watchdog alarms

The CPUcs watchdog timer (WDT) overflow or access error alarm require a special processing in order to ensure a correct microcontroller behavior if the watchdogs are not serviced by software or firmware. The default reset configuration of these alarms ensures that the microcontroller is reset after a pre-warning phase, where software can still perform some critical actions. The following properties are implemented as reset values for the CPUcs WDT overflow or access error alarm and for the recovery timer 0:

- The CPUcs WDT overflow or access error alarm is configured to activate a CS_NMI

- The Recovery Timer 0 is configured to service CPUcs WDT overflow or access error alarm
- The Recovery Timer 0 timeout alarm is configured to issue a reset request CS_RESET_REQ

Because the CPUcs WDToverflow or access error detection is also required from the very first instruction executed by a CPUcs, the SMU_CS is set up to process any CPUcs WDToverflow or access error alarm during the START state of its internal state machine.

42.3.4.4 Access protection

The SMU_CS registers are accessed by SPB bus and are access protected based on PROT and BUS Access Protection Unit (APU) mechanism. More explanation for the PROT and APU based access protection can be found in the FPI chapter (section Access Protection) in the user manual.

The SMU_CS has a Safe Endinit PROT Register (CS_PROTCSE) that controls access to the Bus Access Protection Unit (APU) registers (CS_ACCEN_WRA, CS_ACCEN_WRB, CS_ACCEN_RDA, CS_ACCEN_RDB, CS_ACCEN_VM, CS_ACCEN_PRS). Only the PROT owner defined CS_PROTCSE can configure the Bus APU registers. Bus APU registers define the access rule for the SMU_CS configuration and status register. The SMU_CS configuration and status register are read- or write- access protected as defined by BUS APU registers of protection group PCS in the register overview.

For certain configuration registers, the write access is controlled by the PROT owner defined in SMU_CS Endinit PROT Register (CS_PROTCE). These registers are marked with CE in the Register Overview table.

Besides PROT and APU based access protection, the SMU_CS has an internal locking mechanism. This locking mechanism is controlled by CS_KEYS register and is explained in the Safety Manual chapter SM[HW]:SMU:LOCK. Please always refer to the device specific safety manual to find out which safety mechanisms of this functional block need to be considered for safety relevant applications.

Related information

Registers overview - SMU (ascending offset address) on page 7109

42.3.4.5 Reset types

The System Mode Management (SMM) implements different reset types that affect the SMU_CS. The reset types that are relevant for SMU_CS are:

- Warm power-on reset
- Debug reset
- Application reset

The table below specifies the scope of each reset type to the SMU_CS control configuration and logic.

Table 1975 Effect of reset types to SMU_CS functionality

SMU function	Application reset	Debug reset	Warm power-on reset
SMU_CS state machine function	Not affected	Not affected	Reset
SMU_CS debug function	Not affected	Reset	Reset
SMU_CS alarm diagnosis registers CS_AGCSi_DBG, CS_AGSHi_DBG	Not affected	Not affected	Reset
SMU_CS alive monitor[1)]	Reset	Not affected	Reset
SMU_CS shared alarm selection and status registers CS_AGSHi_SEL, CS_AGSHi_SS	Not affected	Not affected	Reset
SMU_CS other functions	Reset	Not affected	Reset

1) The description of Alive Monitor is present in Safety Manual SM[HW]:SMU:ALIVE_MONITOR.

Related information

System Mode Management (SMM) on page 6315
SMU_CS state machine on page 7081
OCDS trigger bus (OTGB) interface on page 7220

42.3.5 SMU_GCC

SMU Global Control and Configuration (SMU_GCC) takes care of global configuration for SMU alarm handlers and other safety relevant functionality mapped to SMU functional block. This includes:

- Register Monitor Unit
- Emergency Stop Unit

The SMU_GCC is implemented within core domain, that is, provided by core voltage and f_{SPB} clock (refer to the PMS and Clocking chapter of the UM for more details).

42.3.5.1 Register monitor unit

If a register is monitored by a safety flip-flop, a self-test mechanism is also available for the safety flip-flop. This self-test mechanism is explained in the Safety Manual chapter SM[HW]:<FB>:REG_MONITOR, where <FB> refers to the Functional Block where safety flip-flop is implemented. GCC_RMi_CTL (i=0-4) Registers are used to start the self-test while the Registers GCC_RMi_STS (i=0-4) are used to capture the status of self-test. The failures are reported back to the Registers GCC_RMi_EF (i=0-4).

Please refer to the product specific section of this document to find the product specific Register Monitor Mapping tables.

42.3.5.2 Emergency stop unit

The Emergency Stop unit (ES) provides a fast reaction to an emergency event without the intervention of software. In response to the emergency event, selected output ports can be immediately placed into a defined state (for more information refer to the PORTS chapter in the User Manual).

An Emergency Stop may be triggered by either of the following emergency events:

- A transition on the port which is configured as the Emergency Stop input
- An SMU alarm event or SMU command which is enabled and configured to generate an Port Emergency Stop (PES)

The figure below shows a diagram of the emergency stop input logic. This logic is controlled by the Emergency Stop Register GCC_EMSR. There is also a possibility of using a Glitch filter, implemented in the SMU_SAFE0, for the Port A. For Port B the Glitch filter is implemented within SMU_GCC.

Attention**: **When a fault signaling protocol (FSP) is used on the Port A, the Port A cannot be chosen as input trigger for the Emergency Stop

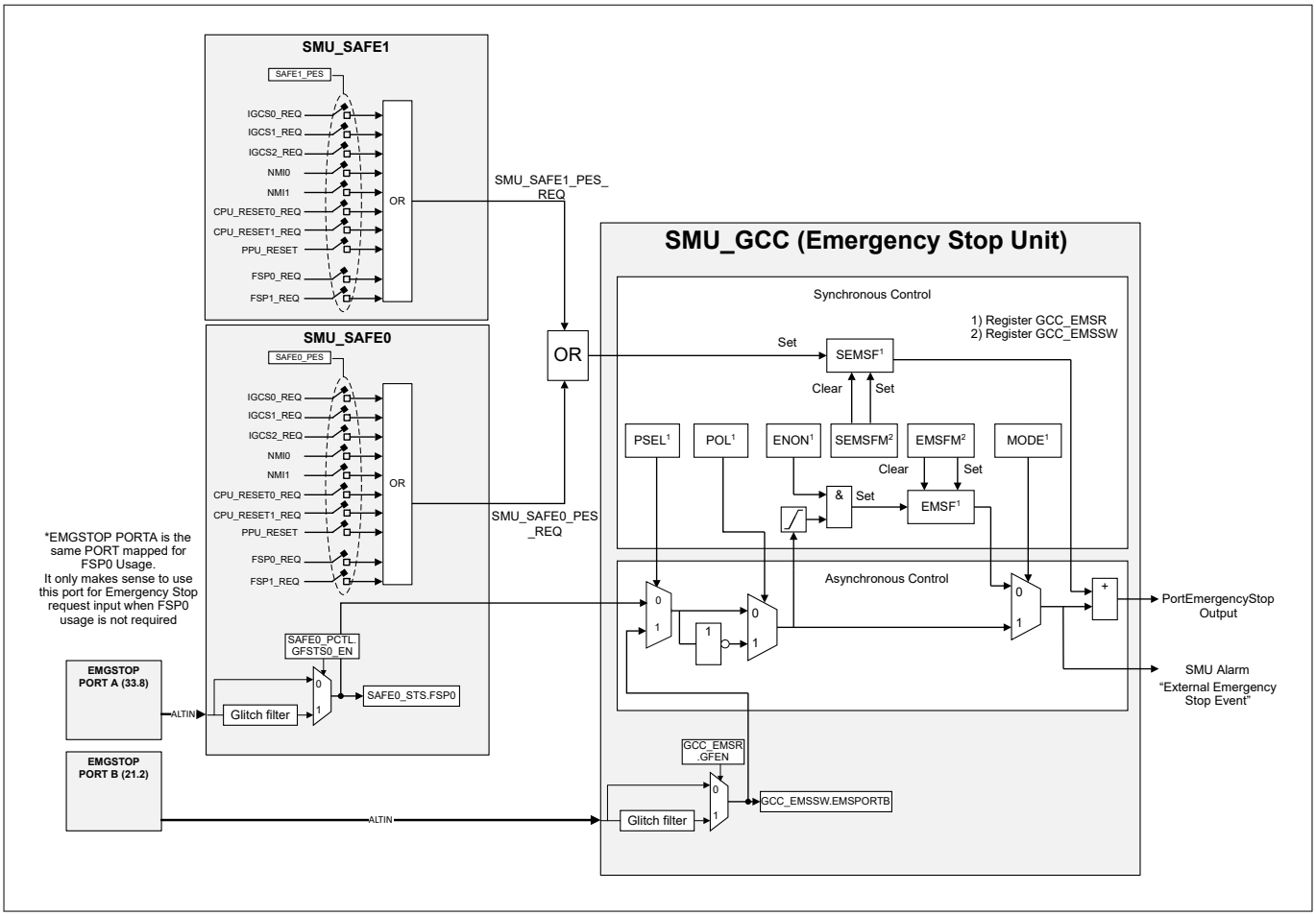


Figure 992 Emergency stop control

42.3.5.2.1 Port triggered emergency stop

The SMU can be configured to trigger an emergency stop on either transition edge of either one of two ports.

The two input port options are (PORT A) P 33.8 and (PORT B) P21.2.

The emergency stop control logic for the ports can basically operate in two modes:

- Synchronous Mode (default after reset): Emergency case is activated by hardware and released by software
- Asynchronous Mode: Emergency case is activated and released by hardware

In Synchronous Mode (selected by GCC_EMSR.MODE = 0), the port signal is sampled for an inactive-to-active level transition, and an emergency stop flag GCC_EMSR.EMSF is set if the transition is detected. The setting of GCC_EMSR.EMSF activates the emergency stop. A port triggered emergency state can only be terminated by clearing GCC_EMSR.EMSF by software (Write GCC_EMSR.EMSFM with 10_B). The synchronous control logic is clocked by the system bus clock f_{SPB}. This results in a small delay between the port signal and emergency stop signal generation.

In Asynchronous Mode (selected by GCC_EMSR.MODE = 1), the occurrence of an active level at the port input immediately activates the emergency stop signal. Of course, a valid-to-invalid transition of the port input (emergency case is released) also immediately deactivates the emergency stop signal.

The GCC_EMSR.POL bit determines the active level of the input signal from the port. The GCC_EMSR.MODE bit selects Synchronous or Asynchronous Mode for emergency stop signal generation and the GCC_EMSR.PSEL bit selects which of the two ports is used as the emergency stop trigger.

42.3.5.2.2 SMU_SAFE0 or SMU_SAFE1 event triggered emergency stop

The Safety Alarm(s) which can trigger an Emergency Stop are configured and enabled within the SMU_SAFE0 or SMU_SAFE1. All SMU_SAFE0 or SMU_SAFE1 triggered Emergency Stop cases are in Synchronous Mode, regardless of the state of GCC_EMSR.MODE. The safety emergency stop flag GCC_EMSR.SEMSF is set when a configured and enabled SMU_SAFE0 or SMU_SAFE1 Safety Alarm occurs. The setting of GCC_EMSR.SEMSF activates the emergency stop. An SMU_SAFE0 or SMU_SAFE1 triggered emergency state can only be terminated by clearing the GCC_EMSR.SEMSF by software (Write SEMSFM with 10_B). The synchronous control logic is clocked by the system bus clock f_{SPB}.

42.3.5.3 Access protection

The SMU_GCC registers are accessed by SPB bus and are access protected based on PROT and BUS Access Protection Unit (APU) mechanism. More explanation for the PROT and APU based access protection can be found in the FPI chapter (section Access Protection) in the user manual.

The SMU_GCC has a Safe Endinit PROT Register (GCC_PROTGSE) that controls access to the Bus Access Protection Unit (APU) registers (GCC_ACCEN_WRA, GCC_ACCEN_WRB, GCC_ACCEN_RDA, GCC_ACCEN_RDB, GCC_ACCEN_VM, GCC_ACCEN_PRS). Only the PROT owner defined GCC_PROTGSE can configure the Bus APU registers. Bus APU registers define the access rule for the SMU_GCC configuration and status register. The SMU_GCC configuration and status register are read- or write- access protected as defined by BUS APU registers of protection group PGCC in the register overview.

For certain configuration registers, the write access is controlled by the PROT owner defined in SMU_GCC Endinit PROT Register (GCC_PROTGE). These registers are marked with GE in the Register Overview table.

Besides PROT and APU based access protection, the SMU_GCC has an internal locking mechanism. This locking mechanism is controlled by GCC_KEYS register and is explained in the Safety Manual chapter SM[HW]:SMU:LOCK. Please always refer to the device specific safety manual to find out which safety mechanisms of this functional block need to be considered for safety relevant applications.

Related information

Registers overview - SMU (ascending offset address) on page 7109

42.3.5.4 Reset types

The System Mode Management (SMM) implements different reset types that affect the SMU_GCC. The reset types that are relevant for SMU_GCC are:

- Warm power-on reset
- Debug reset
- Application reset

The following table specifies the scope of each reset type to the SMU_GCC control configuration and logic.

Table 1976 Effect of reset types to SMU_GCC functionality

SMU function	Application reset	Debug reset	Warm power-on reset
Emergency stop unit	Reset	Not affected	Reset
Register monitor unit	Reset	Not affected	Reset
SMU_SAFE0/1 Shared Alarm Selection and Status Registers GCC_AGSHi_SEL, GCC_AGSHi_SS	Not affected	Not affected	Reset

(table continues...)

Table 1976 (continued) Effect of reset types to SMU_GCC functionality

SMU function	Application reset	Debug reset	Warm power-on reset
SMU OCDS control and status register OCS	Not affected	Reset	Reset
SMU_GCC other functions	Reset	Not affected	Reset

Related information

System Mode Management (SMM) on page 6315
Emergency stop unit on page 7090
Register monitor unit on page 7090
Shared alarm selector on page 7059

42.3.6 SMU_STDBY

SMU_STDBY is a redundant safety alarm handler that can monitor the failure of the SMU_SAFE0, SMU_SAFE1, SMU_CS and other common cause failure initiators of the microcontroller. Within SMU_STDBY each safety alarm can be individually configured to trigger external reactions on FSP independently from SMU_SAFE0, SMU_SAFE1. In the Alarm Handling section it is explained how incoming alarms are processed. SMU_STDBY is provided with back-up voltage and f_{BACK} clock (refer to the PMS and Clocking chapter of the UM for more details) in order to achieve independence from SMU_SAFE0, SMU_SAFE1 and SMU_CS.

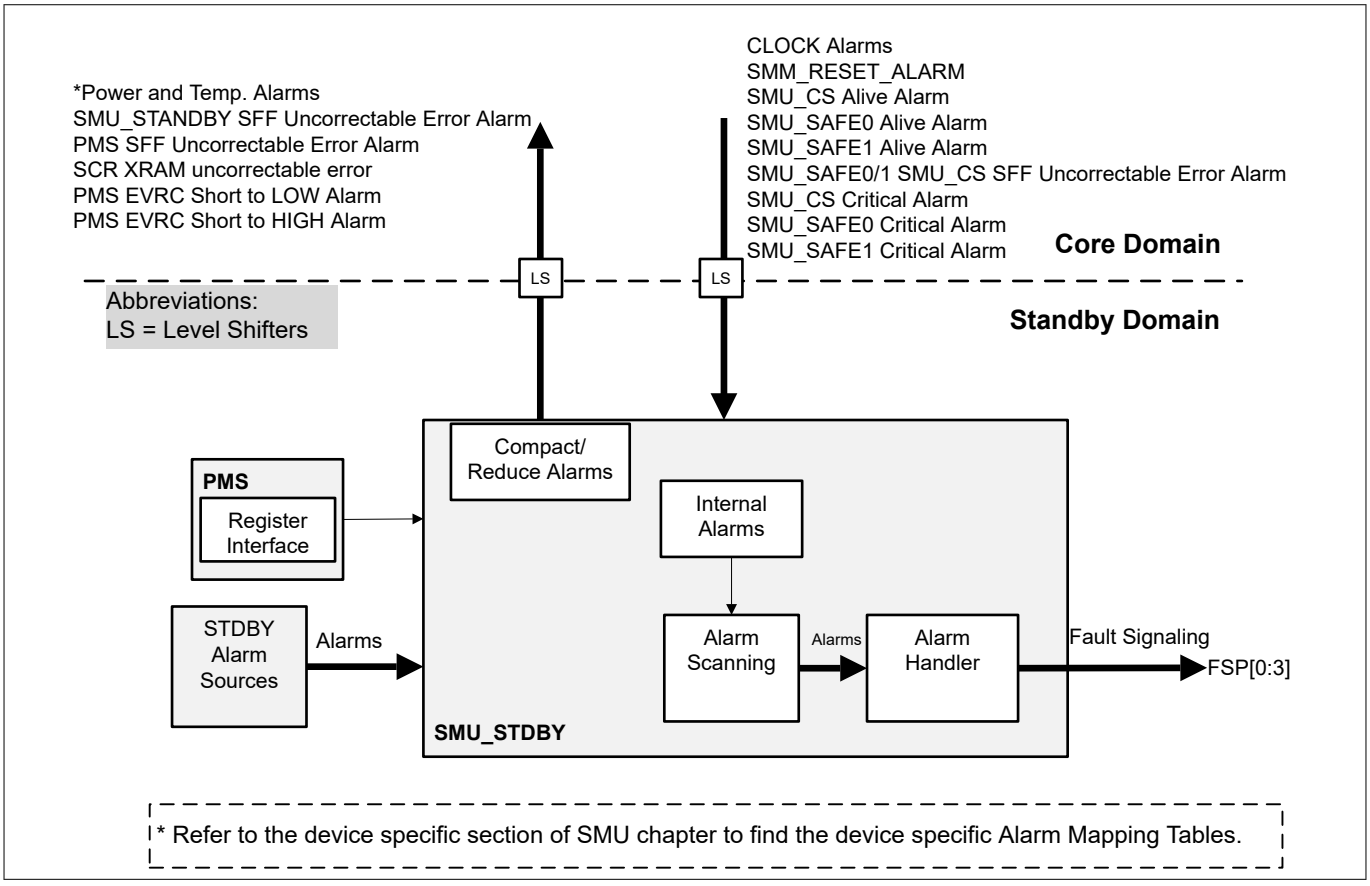


Figure 993 SMU_STDBY overview

42 Safety and security alarm Management Unit (SMU)

Related information

Alarm handling on page 7095
CLOCK on page 5525
Power Management System (PMS) on page 5946

42.3.6.1 Alarm handling

This section specifies the hardware and software alarm processes.

42.3.6.1.1 Alarm configuration

Upon reception of an alarm event, the SMU_STDBY either does not generate any reaction, or sets FSP[0...3] in high impedance state (tristate) or drives FSP[0...3] to low with internal pull-down circuit. All of these behaviors can be configured indvidually for each alarm.

The external behavior for any incoming alarm ALMSTDBY*i*[*z*], where *i*=0-4 indicates the alarm group and *z*=0-31 indicates the alarm number, is configured by enabling the corresponding bit-fields of the following registers:

- AGSTDBYi_FSP0.FEz
- AGSTDBYi_FSP1.FEz
- AGSTDBYi_FSP2.FEz
- AGSTDBYi_FSP3.FEz

For example for ALMSTDBY**0**[**5**] the intended reaction is to generate Fault State on FSP0 and FSP1, then AGSTDBY0_FSP0.FE5 and AGSTDBY0_FSP1.FE5 needs to be configured as 1.

CMDSTDBY.SMUEN shall also be set to enable the FSP reaction on FSP[0...3]. Moreover, through the bit-fields in the register CMDSTDBY.FSPxEN, with x = 0-3, it is possible to enable or disable the reaction on the corresponding FSP. These bit-fields need to be enabled besides the alarm mapping register. The FSP Fault State can be configured for corresponding FSP pins by bit-fields CMDSTDBY.FSPxPD where *x* = 0-3. The Fault State on the FSP pin can be high impedance state (tristate) or low driven by internal pull-down circuit.

The status flag in the alarm status registers AGSTDBYi_STS (where i=0-4) is set to 1 by hardware whenever an alarm event is received by SMU_STDBY. Software can clear this status flag by first writing 1 to bitfield CMDSTDBY.ASCE and then writing 1 to the corresponding status flags in the AGSTDBYi_STS (where i=0-4) register. Afterwards the bit CMDSTDBY.ASCE is reset to 0 by hardware. Therefore, for clearing status flags from multiple status registers AGSTDBYi_STS (where i=0-4), the complete sequence described before needs to be performed for each individual status register AGSTDBYi_STS (where i=0-4). However multiple status flags within the same register can be cleared by first writing 1 to bitfield CMDSTDBY.ASCE and then writing 1 to all the corresponding status flags in the alarm status register AGSTDBYi_STS (where i=0-4) in a single write operation.

***Note**: In order to recognize the high impedance state of the FSP Pins (ErrorPins) as the Fault State, an external pull-down device might be necessary, whereas with CMDSTDBY.FSPxPD set to be 1, this is not required.*

Attention**: **When the SMU_STDBY, sets the FSP[0] or FSP[1] in high impedance state as an alarm reaction, the corresponding status field SAFE0_STS.FSP0 or SAFE0_STS.FSP1 will be set to 1 even if no external pull-down device is connected to the respective ErrorPin. The same condition is true for FSP[2] and FSP[3] and its corresponding status fields SAFE1_STS.FSP0 and SAFE1_STS.FSP1.

42.3.6.2 Interface to the pads (ErrorPin)

The SMU_STDBY has the ability to signal an error to the external world by the FSP pin (ErrorPin) independently from SMU_SAFE0 or SMU_SAFE1.

The figure below specifies the data path connectivity of the SMU_STDBY with the FSP0 pin (ErrorPin0). The ErrorPin0 is connected to the SMU_STDBY by standard ENPS signal and PD_EN signal. The ENPS signal can be driven low by the SMU_STDBY which would set FSP0 in high impedance state regardless of the port configuration and SMU_SAFE0 reactions, if an alarm event reaction is mapped to FSP0. Moreover, if the bitfield CMDSTDBY.FSP0PD is set to 1, PD_EN signal drives the ErrorPin0 to low state with internal pull down circuit.

Otherwise, an external pull-down device might be needed to make the high impedance state correspond to the FSP Fault State.

The behavior of all of the rest of the FSP[1..3] pins is also the same as explained above, which can be configured by their corresponding bitfields in CMDSTDBY register and separate signals controlling the corresponding pins.

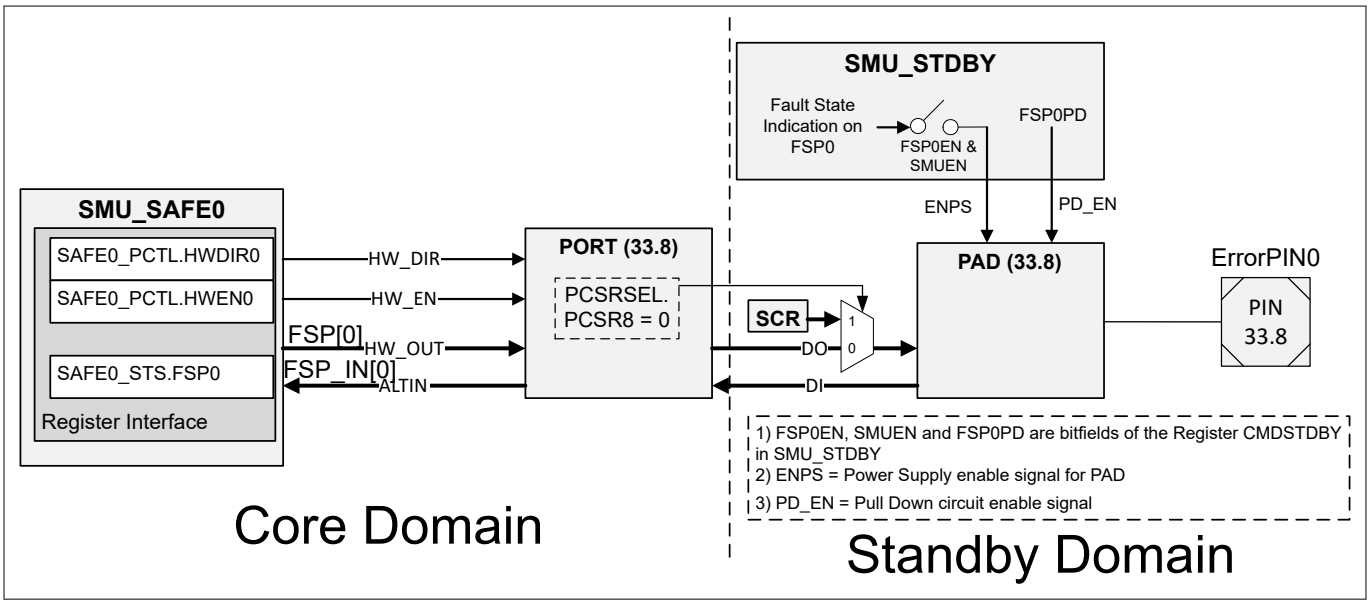


Figure 994 SMU_STDBY data path interfaces with the ErrorPins

For more information refer to the Related information.

Related information

Interface to the ports pad on page 7103

42.3.6.3 Interdependency between SMU_SAFE0 (or SMU_SAFE1) and SMU_STDBY

The SMU_SAFE0 (or SMU_SAFE1) and SMU_STDBY are designed to function together. SMU_STDBY can only be used to react to alarms when the SMU_SAFE0 (or the SMU_SAFE1) is enabled and their corresponding state machine (SSM0 or SSM1) is in RUN state. The reasons are listed below:

- Fault Signaling Protocol is generated by the SMU_SAFE0 (or SMU_SAFE1)
- If the SMU_SAFE0 (or SMU_SAFE1) is not enabled and their corresponding state machine is not in RUN State, the ErrorPin remains in their default state, which is the FSP Fault State
- Therefore, the reaction of the SMU_STDBY to any alarm (setting the ErrorPin in Fault State) will not be noticeable

On the other hand the application software can configure the SMU to enable SMU_SAFE0 (or SMU_SAFE1) only and disable alarm processing in SMU_STDBY if the diverse processing of safety alarms generated by certain safety mechanisms (voltage, clock and temperature monitors) detecting potential common cause faults is not required. SMU_STDBY alarm status registers, AGSTDBYi_STS (i=0-4), are accessible to application software even when the SMU_STDBY is disabled. The application software can read SMU_STDBY alarm status registers periodically to check the occurrence of an alarm.

42.3.6.4 Access protection

The SMU_STDBY registers are accessed by SPB bus and are access protected based on PROT and BUS Access Protection Unit (APU) mechanism. More explanation for the PROT and APU based access protection can be found in the FPI chapter (section Access Protection) in the user manual.

The SMU_STDBY has a Safe Endinit PROT Register (PROTSTDBYSE) that controls access to the Bus Access Protection Unit (APU) registers (ACCENSTDBY_WRA, ACCENSTDBY_WRB, ACCENSTDBY_RDA, ACCENSTDBY_RDB, ACCENSTDBY_VM, ACCENSTDBY_PRS). Only the PROT owner defined PROTSTDBYSE can configure the Bus APU registers. Bus APU registers define the access rule for the SMU_STDBY configuration and status register. The SMU_STDBY configuration and status register are read- or write- access protected as defined by BUS APU registers of protection group P in the register overview.

For certain configuration registers, the write access is controlled by the PROT owner defined in SMU_STDBY Endinit PROT Register (PROTSTDBYE). These registers are marked with E in the Register Overview table.

Besides PROT and APU based access protection, the configuration registers (MONBISTCTRL, CMDSTDBY, AGSTDBYi_FSPj) can only be updated during the register write operation if the BITPROT field of the corresponding registers is written with 1 by the software during the same register write operation. Please always refer to the device specific safety manual to find out which safety mechanisms of this functional block need to be considered for safety relevant applications.

42.3.6.5 Reset types

The System Mode Management (SMM) implements different reset types that affect the SMU_STDBY. The reset types that are relevant for SMU_STDBY are:

- Warm PORST
- Cold PORST
- LVDPMS2 reset

The following table specifies the scope of each reset type to the SMU_STDBY functions (a function includes the control and configuration registers and the related logic).

Table 1977 Effect of reset types to SMU_STDBY functionality

SMU function	Warm PORST	Cold PORST	LVDPMS2 reset
SMU_STDBY alarm status registers (AGSTDBYi_STS)	Not affected	Reset	Reset
SMU_STDBY alarm configuration functionalities	Reset	Reset	Reset
SMU_STDBY BIST functionality	Reset	Reset	Reset
SMU_STDBY other functions	Reset	Reset	Reset

Related information

System Mode Management (SMM) on page 6315

42.3.7 SMU_SAFE0, SMU_SAFE1 and SMU_CS interaction overview

This section describes the interaction between all the core domain SMU instances (SMU_CS, SMU_SAFE0, SMU_SAFE1) to other Functional Blocks.

42.3.7.1 Interface to the system mode management (SMM)

All the reset requests from the SMU are forwarded to the functional block SMM. The interface signals used are listed below:

- CS_RESET_REQ
- RESET0_REQ
- RESET1_REQ
- RESET2_REQ
- RESET3_REQ

Along with each reset request, alarm information (containing Alarm Group ALMgrp and Alarm ID ALMid), is also forwarded to SMM. The alarm information indicates which alarm has triggered reset request. The same alarm information is also captured by corresponding registers CS_RST and SAFEx_RST in SMU_CS, SMU_SAFE0 and SMU_SAFE1. In case of multiple simultaneous alarms mapped to generate a reset request, ALMgrp and ALMid indicate the alarm which is mapped to the lowest alarm group and assigned with the lowest alarm index within that alarm group. The following figure shows the interaction between SMU_CS, SMU_SAFE0 and SMU_SAFE1 to SMM.

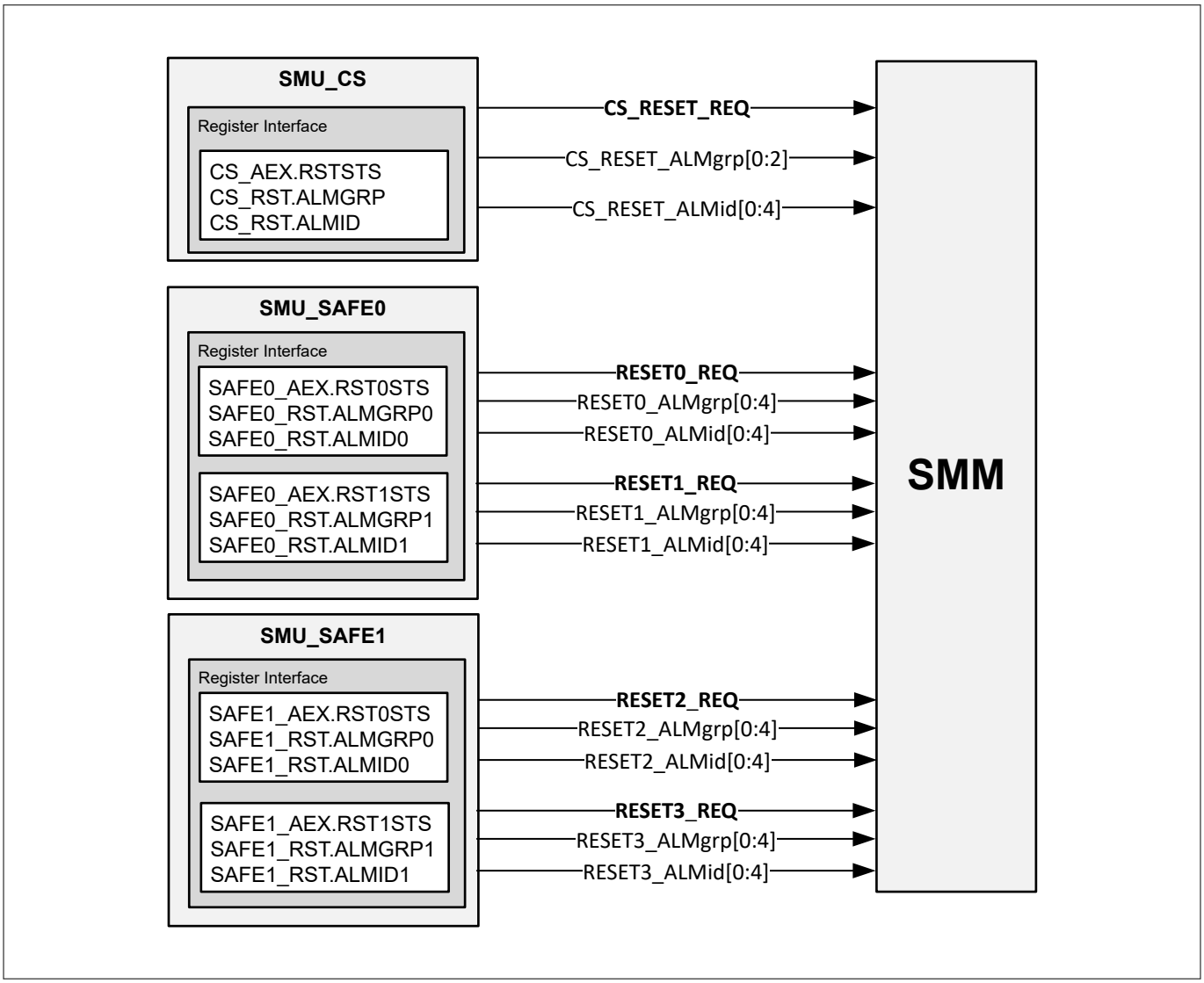


Figure 995 Interface to SMM

ALMgrp also contains information about shared alarm group ALMSH0 - ALMSH2. For detail alarm group mapping refer to table below.

Table 1978 Alarm group mapping to ALMgrp signal

Alarm group	CS_RESET_ALMgrp value	RESETx_ALMgrp value (x = 0-4)
ALMCS0 - ALMCS1	0 - 1	NA
ALMSF0 - ALMSF16	NA	0-16
ALMSH0 - ALMSH2	2-4	17-19

42.3.7.2 Interfaces to the Interrupt Router (IR)

All the interrupt service requests from the SMU are forwarded to the functional block IR. The interface signals used are listed below:

- CS_IRQ
- IRQ0
- IRQ1
- IRQ2
- IRQ3
- IRQ4
- IRQ5

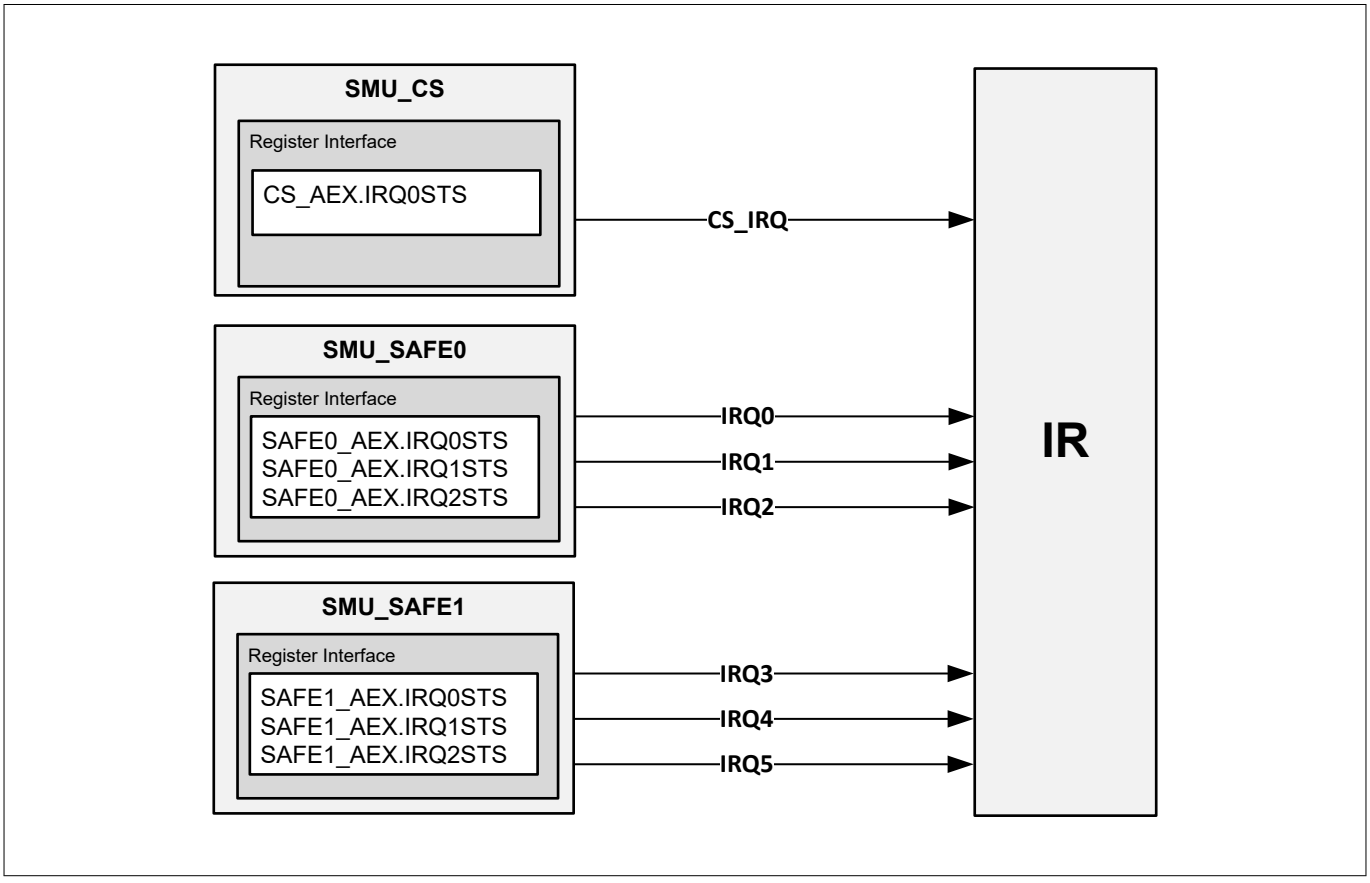


Figure 996 Interface to IR

The SAFEx_AGC.IGCSi, i={0,1,2} register fields provide the software interface to control how the SMU_SAFE0 and SMU_SAFE1 trigger interrupt requests to the interrupt router.

Each SAFEx_AGC.IGCSii is a 3-bits bit-field. For SMU_SAFE0:

- SAFE0_AGC.IGCSi[0] shall be set to 1 to trigger Interrupt Service Request 0 IRQ0
- SAFE0_AGC.IGCSi[1] shall be set to 1 to trigger Interrupt Service Request 1 IRQ1
- SAFE0_AGC.IGCSi[2] shall be set to 1 to trigger Interrupt Service Request 2 IRQ2

For SMU_SAFE1:

- SAFE1_AGC.IGCSi[0] shall be set to 1 to trigger Interrupt Service Request 3 IRQ3
- SAFE1_AGC.IGCSi[1] shall be set to 1 to trigger Interrupt Service Request 4 IRQ4
- SAFE1_AGC.IGCSi[2] shall be set to 1 to trigger Interrupt Service Request 5 IRQ5

The usage of the SAFEx_AGC.IGCSi bit-fields is defined in the figure below

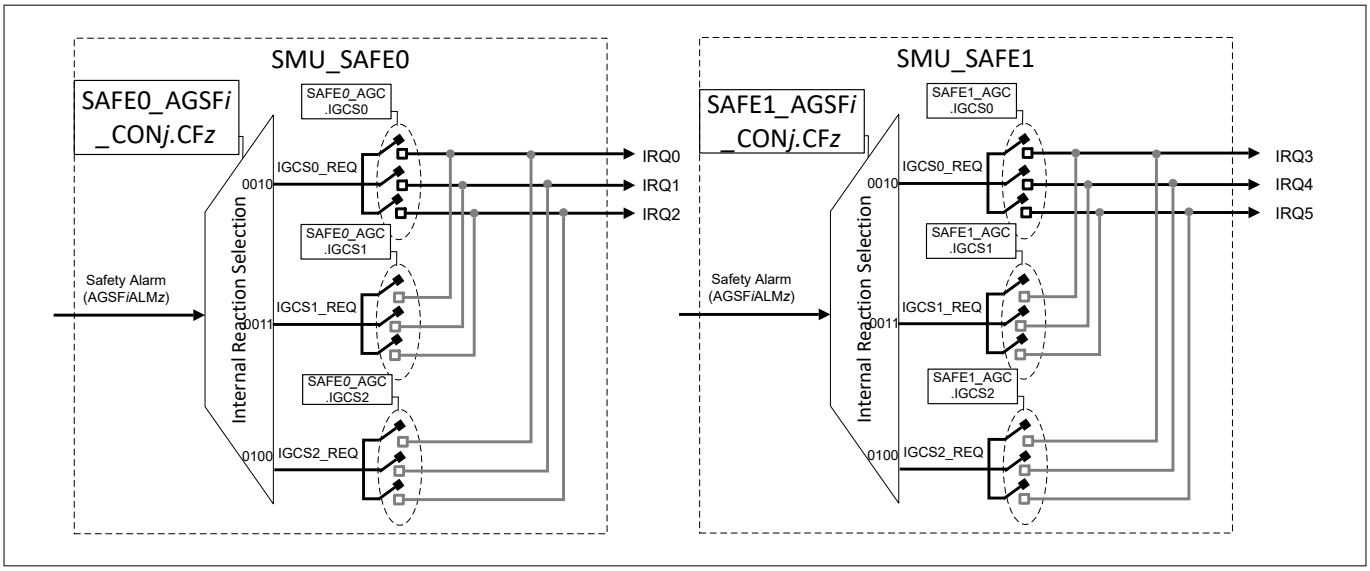


Figure 997 SMU_SAFE0 and SMU_SAFE1 interrupt service request generation

42.3.7.3 Interfaces to the System Control Unit (SCU)

All the Non-Maskable Interrupts (NMI) from the SMU are forwarded to the functional block SCU. The interface signals used are listed below:

- CS_NMI
- NMI0
- NMI1
- NMI2
- NMI3

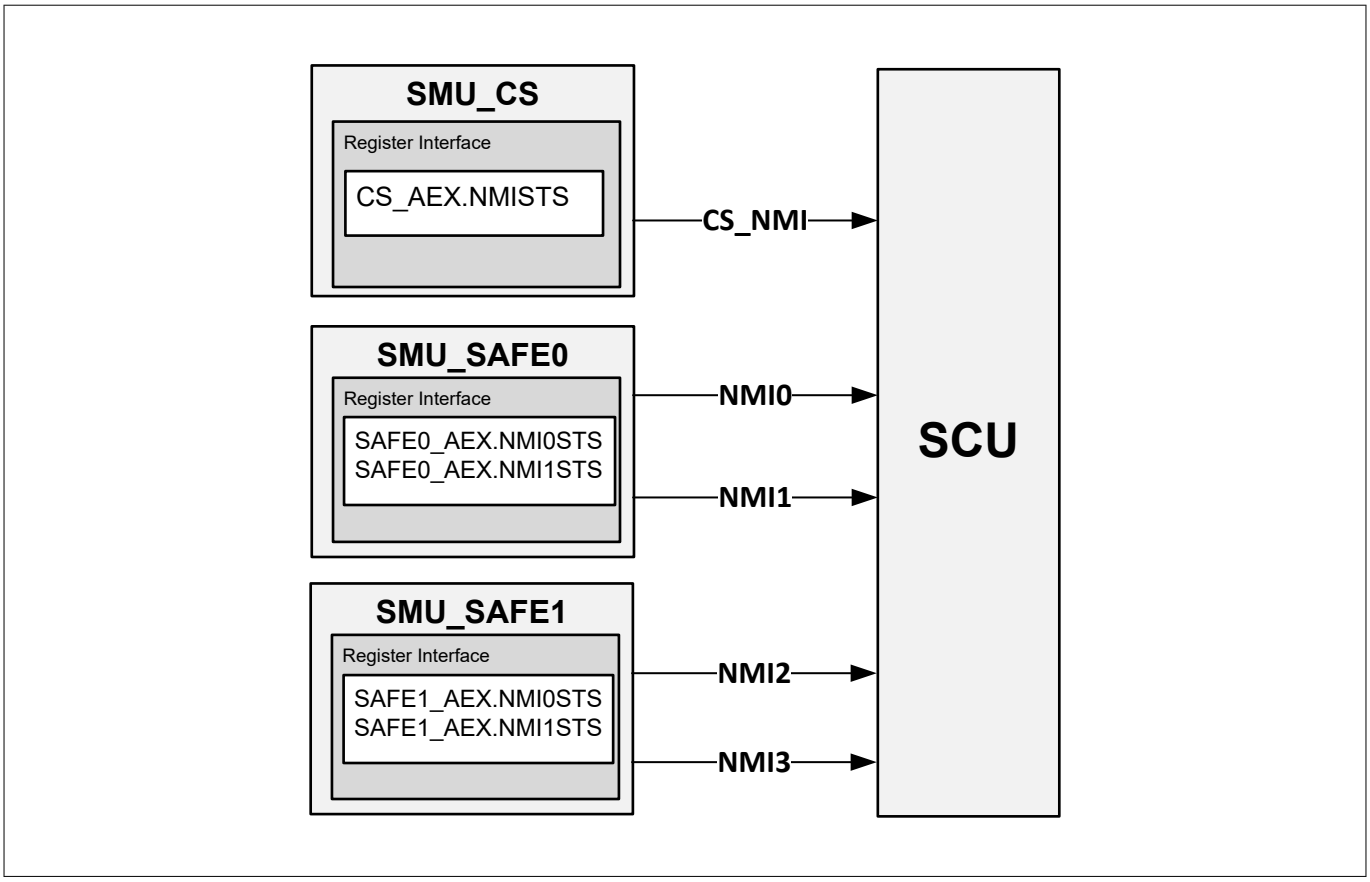


Figure 998 Interface to SCU

42.3.7.4 Interface to the Watchdog Timer Unit (WTU)

Signals indicating the RUN state of SMU_SAFE0, SMU_SAFE1 and SMU_CS state machines are forwarded to Watchdog Timer Unit (WTU). The WTU can be configured to lock certain configuration registers depending upon the state of SMU state machines (refer to the WTU chapter in user manual for more details). If any of the state machines is in the RUN state, the corresponding signal is driven to '1' otherwise in START and FAULT state the signal is driven to '0' (for SMU_CS only on case of START state since there is no FAULT state). All of the signals are listed below :

- SMU_CS state machine SSM will generate SMU_CS_RUN_STATE
- SMU_SAFE0 state machine SSM0 and SSM1 will generate two independent signals SMU_SAFE_RUN_STATE0 and SMU_SAFE_RUN_STATE1
- SMU_SAFE1 state machine SSM0 and SSM1 will generate two independent signals SMU_SAFE_RUN_STATE2 and SMU_SAFE_RUN_STATE3

42.3.7.5 Interfaces to CPU

CPU module reset request from the SMU are forwarded to the corresponding CPUs. The interface signals used are listed below:

- CPU[1:5]_RESET

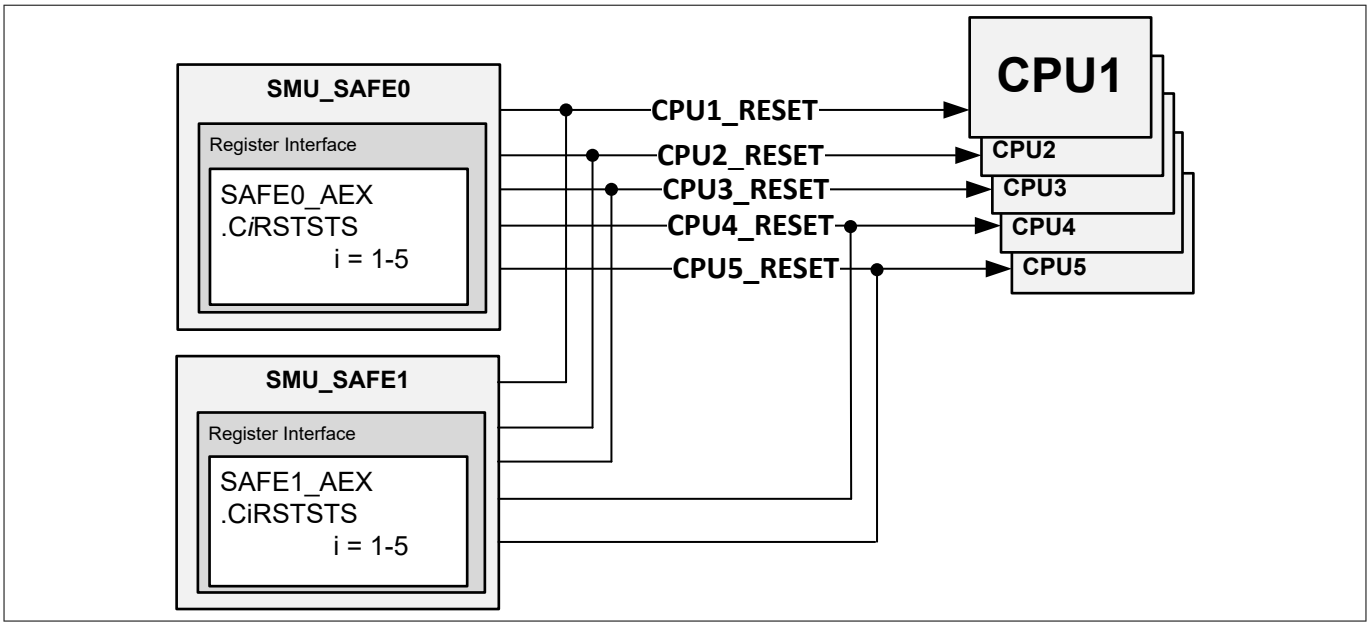


Figure 999 Interface to CPUs

As shown in the figure above, as soon as CPUi_RESET (where i = 1-5) request has been triggered, the corresponding status bits will be set. For example, if only CPU1_RESET request is triggered from SMU_SAFE0 then SAFE0_AEX.C1RSTSTS bitfield will be set.

The SAFEx_AGC.RCSi, i={0,1} register fields provide the software interface to control how the SMU_SAFE0 and SMU_SAFE1 triggers CPU module reset requests.

Each SAFEx_AGC.RCSi is a 5-bits bit-field. For both SMU_SAFE0 and SMU_SAFE1:

- SAFEx_AGC.RCSi[0] shall be set to 1 to trigger CPU1 Module Reset Request CPU1_RESET
- SAFEx_AGC.RCSi[1] shall be set to 1 to trigger CPU2 Module Reset Request CPU2_RESET
- SAFEx_AGC.RCSi[2] shall be set to 1 to trigger CPU3 Module Reset Request CPU3_RESET
- SAFEx_AGC.RCSi[3] shall be set to 1 to trigger CPU4 Module Reset Request CPU4_RESET
- SAFEx_AGC.RCSi[4] shall be set to 1 to trigger CPU5 Module Reset Request CPU5_RESET

The usage of the SAFEx_AGC.RCSi bit-fields is defined in the figure below:

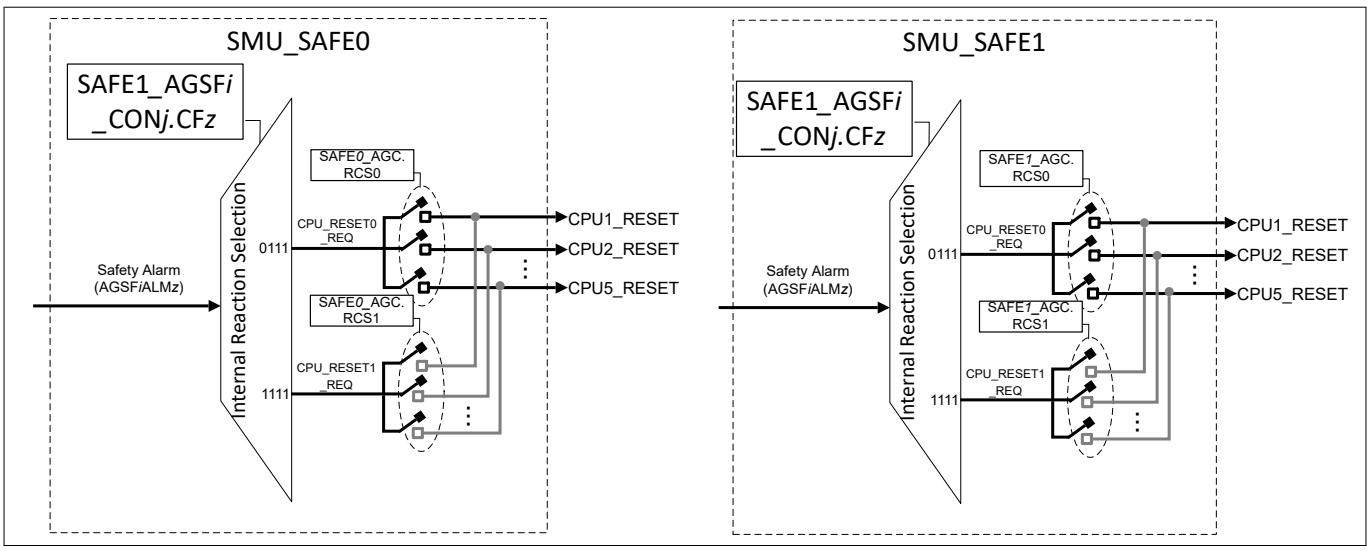


Figure 1000 SMU_SAFE0 and SMU_SAFE1 CPU reset request generation

42.3.7.6 Interface to Cyber Security Satellite (CSS)

All the locking key requests from the SMU are forwarded to the functional block CSS. The interface signals used are listed below:

- LOCK_ALL_KEYS
- LOCK_DBG_PROT_KEYS

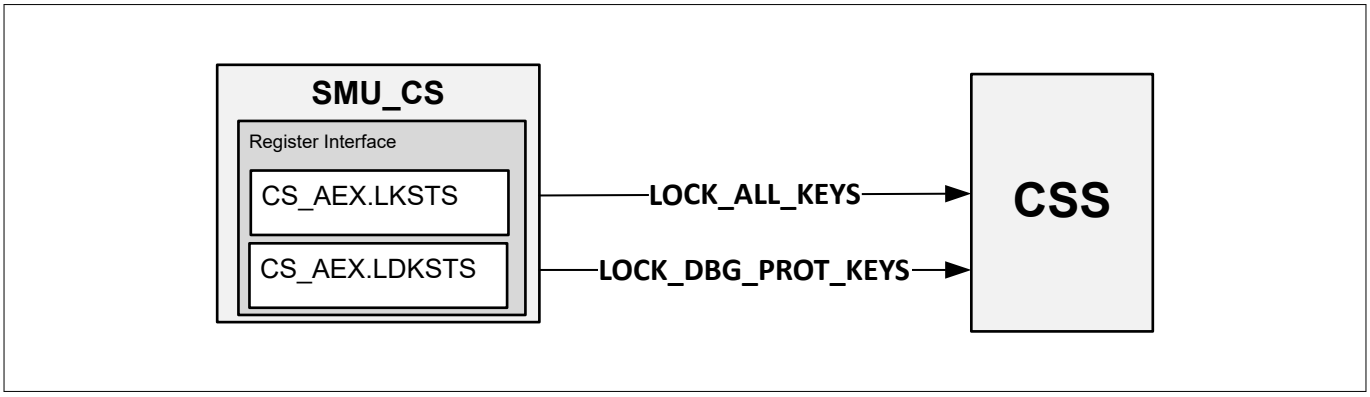


Figure 1001 Interface to CSS

42.3.7.7 Interface to the Parallel Processing Unit (PPU)

All the PPU module reset requests from the SMU are forwarded to the functional block PPU. The interface signals used are listed below:

- PPU_RESET

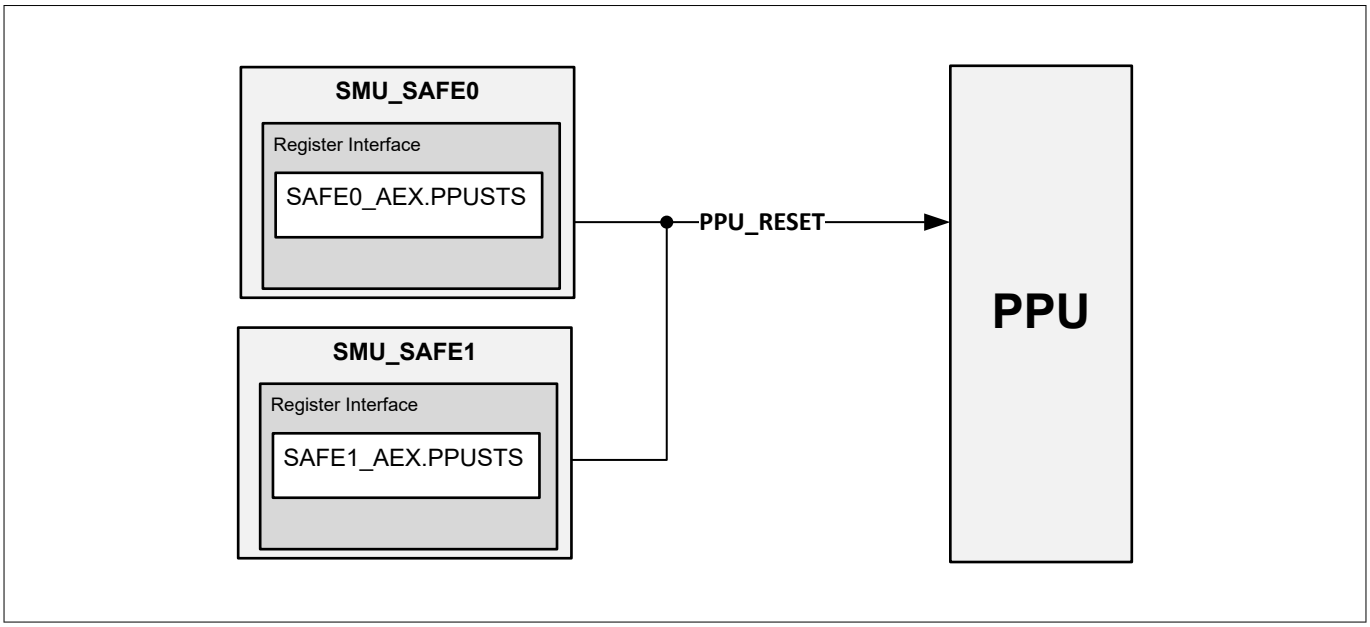


Figure 1002 Interface to PPU

42.3.7.8 Interface to the ports pad

The generic port structure is presented below:

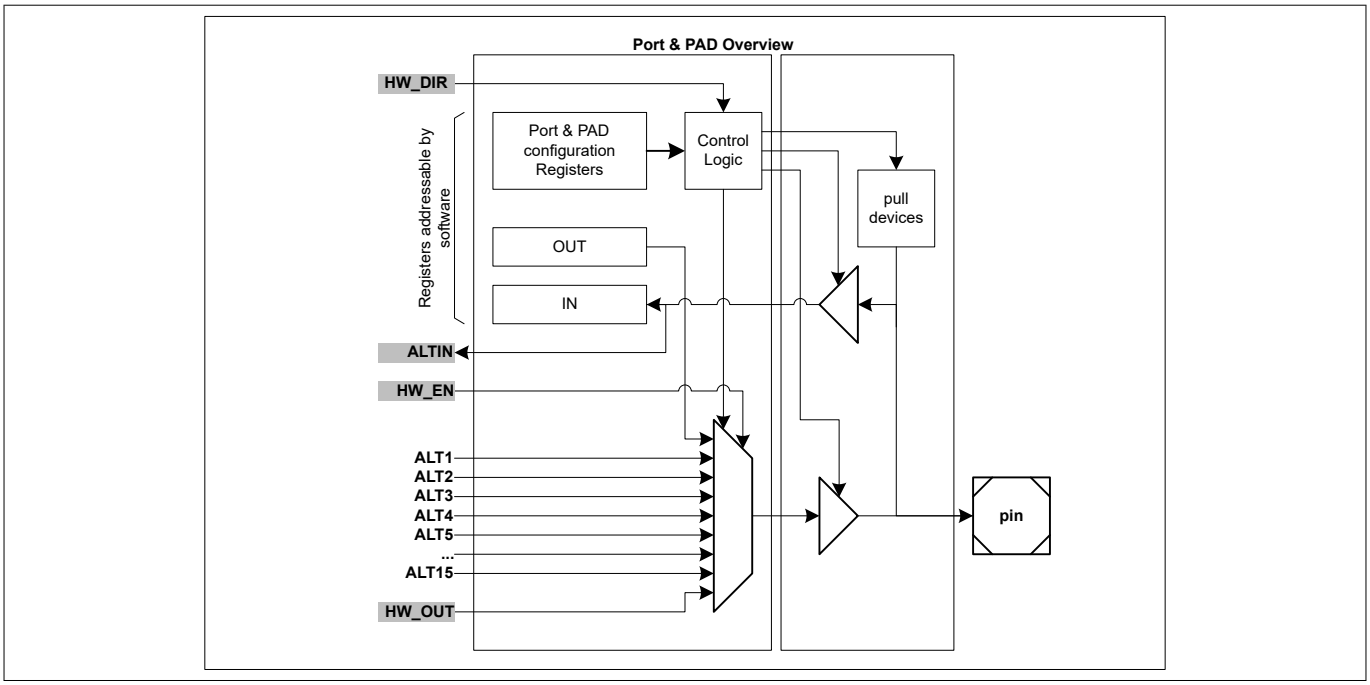


Figure 1003 Generic port structure

The port pin can be connected to peripheral with the ALTx output lines. This is the default state of the port after power-on reset (refer to the section Functional description of PORTS). The SMU_SAFE0 and SMU_SAFE1 connect to PORTS using the HW_DIR, ALTIN, HW_EN, HW_OUT signals. When the HW_EN port input is driven active by the SMU_SAFE0 or SMU_SAFE1, SMU_SAFE0 or SMU_SAFE1 gets full control over the port, bypassing any other software configuration related to the usage of the ALTx inputs.

The figure below provides a more detailed overview of the port structure and highlights the signals involved in the SMU_SAFE0 and SMU_SAFE1 connectivity. Moreover as shown in the figure, the PORT mapping to the corresponding FSP is also fixed, for example FSP0 is always mapped to PORT (33.8) and so on.

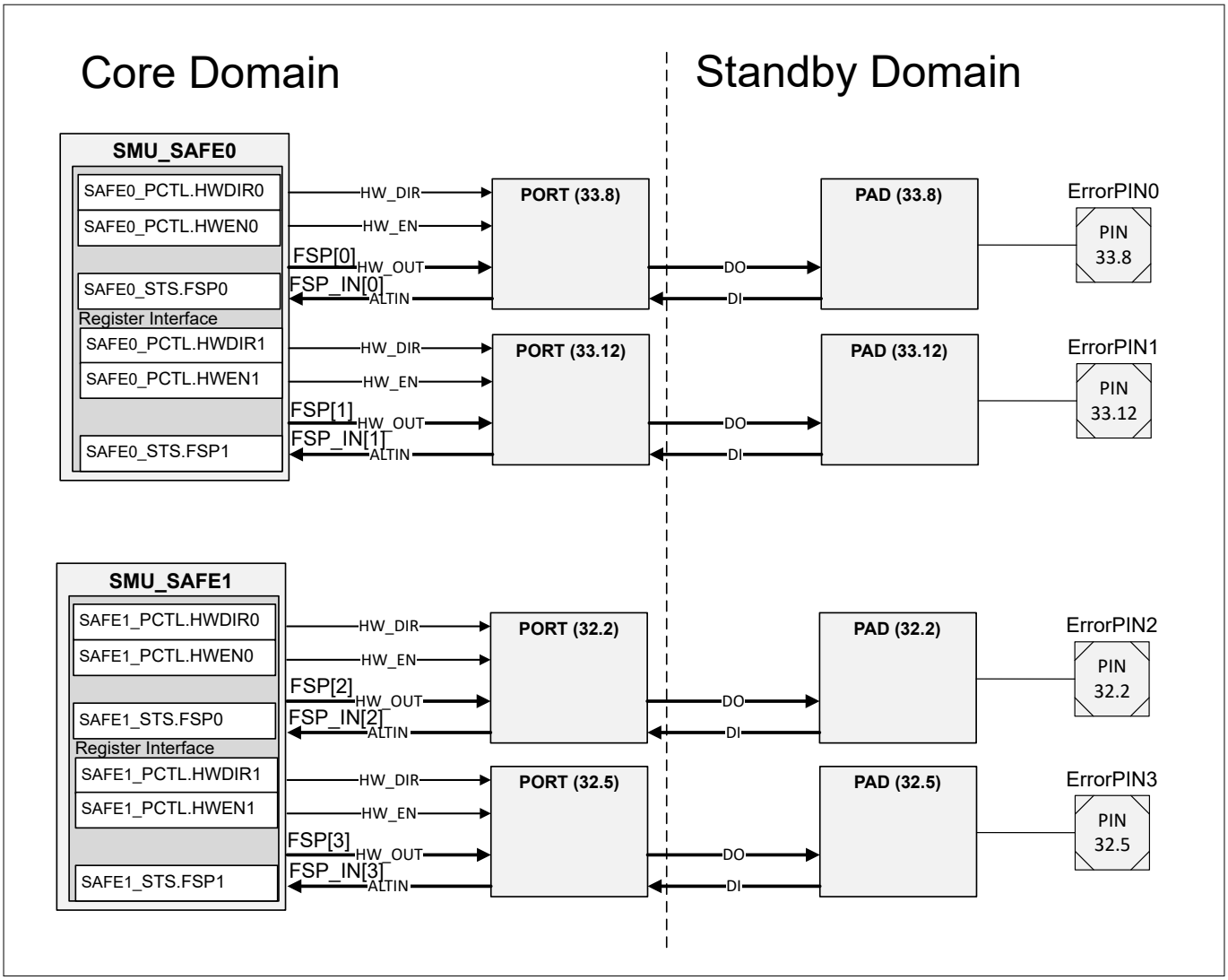


Figure 1004 SMU/PAD control interface to the PADs

All the HW_DIR and HW_EN signals are controlled by software as shown in the figure above. FSP[0], FSP[1], FSP[2] and FSP[3] in the figure above are controlled by hardware. This is shown in the figure below:

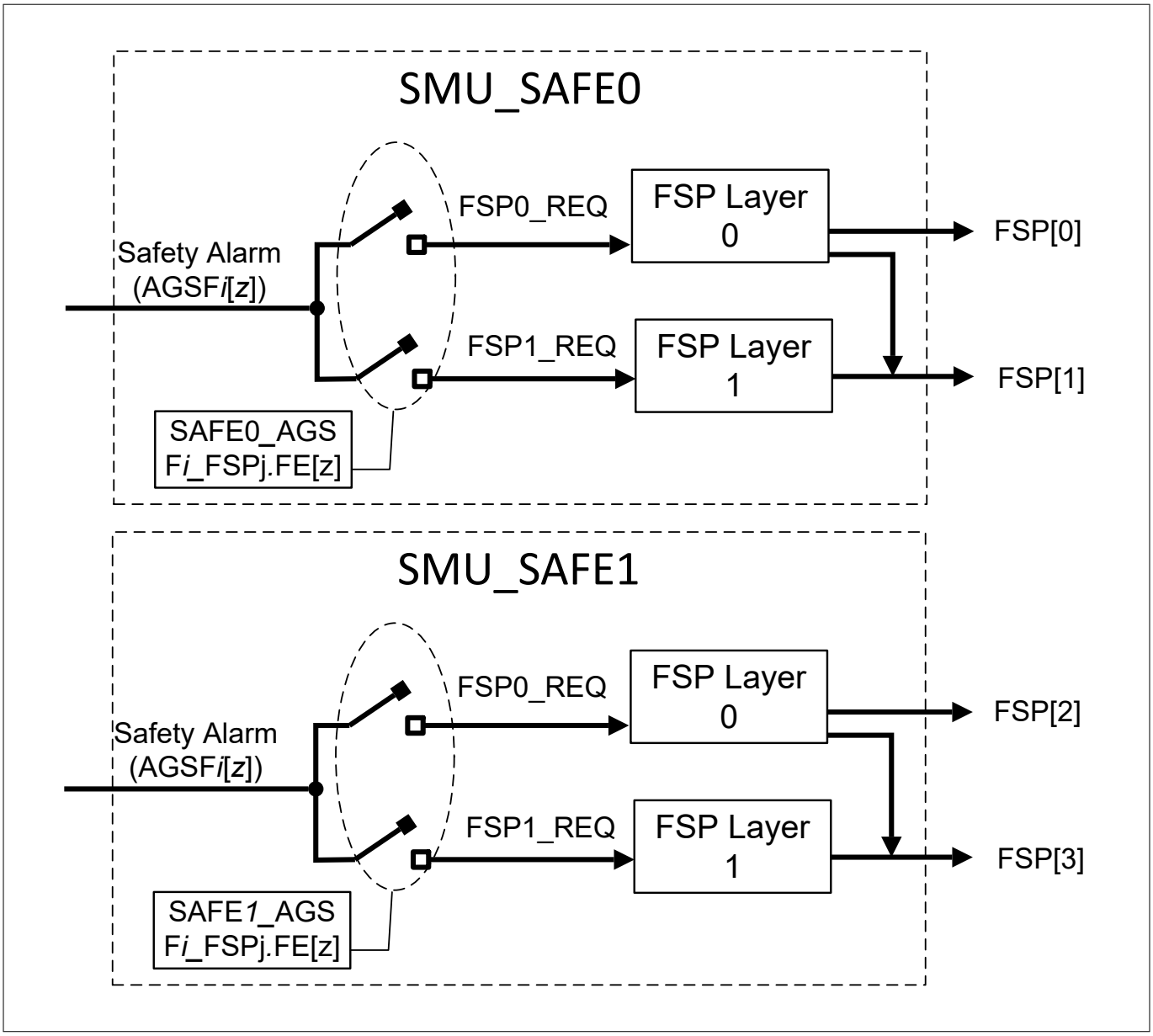


Figure 1005 SMU_SAFE0 and SMU_SAFE1 FSP to PORTS

ErrorPin usage guidelines

This section provides additional information for the usage of the ErrorPins in combination with other input/output (GPIO) functions of the microcontroller. ErrorPin0 (33.8) is used as an example below, for ErrorPins[1:3] the configuration steps remain the same and the corresponding configuration registers for ErrorPins[1:3] shall be used.

***Note**: The PAD properties (push-pull, open-drain, drive strength,...) are configured in the registers of the PORT to which the SMU connects to. These registers are described in the PORTS chapter in this UM.*

- During power-on-reset, the ErrorPin is in high impedance: the pull devices are disabled
- After power-on-reset the default mode of the PORT to which the ErrorPin is connected is GPIO
- Before changing the ownership of the PORT/PAD to SMU, software shall configure the PORT registers including the following:
  - Disable the pull-up and pull-down devices if GPIO is not used

- Program the GPIO registers of the ErrorPin with SPEED0 drive strength and set the low level output on ErrorPin. (Set P33_PADCFG8_DRVCFG.PD = 000_B and set P33_OUT.P8 = 0)
- Set P33_PCSRSEL.PCSR8 to 1

Also, the following steps need to be followed to reconfigure the Fault Signaling Protocol (FSP) settings:

- While in Fault Free State and the Bi-stable protocol is in use:
  - Write desired value to PRE1, PRE2 or TFSP_HIGH in SAFEx_FSPi_CON
  - Switch SAFEx_FSPi_CON.MODE to the desired protocol (if required)
- While in Fault Free State and the Time Switching OR Dual Rail protocol is in use:
  - Write desired value to PRE1, PRE2 or TFSP_HIGH in SAFEx_FSPi_CON

    ***Note**:* *Changing the values of SAFEx_FSPi_CON.PRE1, PRE2, or TFSP_HIGH while the FSP is active and monitored, may result in the violation of the timing expected by the monitoring function.*

  - Switch SAFEx_FSPi_CON.MODE to the desired protocol (if required)

Glitch filter

When ErrorPins are used for Fault Signaling protocol, the logic level driven on the ErrorPins can be read from corresponding status bits implemented in SMU_SAFE0 and SMU_SAFE1. Glitch filters are present in the input paths from ErrorPins to these status bits, which can filter out glitches up to 1.2 µs. The following list and figure describes each path and usage of the filter:

- FSP_IN[0] : ErrorPin0 to SAFEx_STS.FSP0 (where x = 0)
  - For this path the filter can be switched on/off with SAFEx_PCTL.GFSTS0_EN (where x = 0) bit
- FSP_IN[1] : ErrorPin1 to SAFEx_STS.FSP1 (where x = 0)
  - For this path the filter can be switched on/off with SAFEx_PCTL.GFSTS1_EN (where x = 0) bit
- FSP_IN[2] : ErrorPin2 to SAFEx_STS.FSP0 (where x = 1)
  - For this path the filter can be switched on/off with SAFEx_PCTL.GFSTS0_EN (where x = 1) bit
- FSP_IN[3] : ErrorPin3 to SAFEx_STS.FSP1 (where x = 1)
  - For this path the filter can be switched on/off with SAFEx_PCTL.GFSTS1_EN (where x = 1) bit

Glitch filter is also present in the path from EMGSTOP PORT B to Emergency Stop unit. For this path the filter can be switched on/off with GCC_EMSR.GFEN.

Related information

Ports on page 8599
Fault signaling protocol (FSP) on page 7074
Emergency stop unit on page 7090

42.3.7.9 Interface to ports (emergency stop)

One single Port Emergency Stop signal is going from SMU Emergency Stop Unit to all the PORT instances supporting emergency stop feature.

Related information

Emergency stop unit on page 7090

42.3.7.10 Interfaces to SMU_STDBY

The following figure shows an overview of the interaction between the core domain SMU instance (SMU_CS, SMU_SAFE0 and SMU_SAFE1) and the standby domain SMU instance (SMU_STDBY). Please refer to the product specific section of this document to find the product specific alarm information as well as Safety Manual for the description of safety mechanisms triggering these alarms. For SMU related alarms, refer to the sections

SM[HW]:SMU:ALIVE_MONITOR, SM[HW]:SMU:CRITICAL_ALARM and SM[HW]:SMU:REG_MONITOR in Safety Manual for corresponding Alive Alarms, Critical Alarms and SFF Uncorrectable Error Alarm.

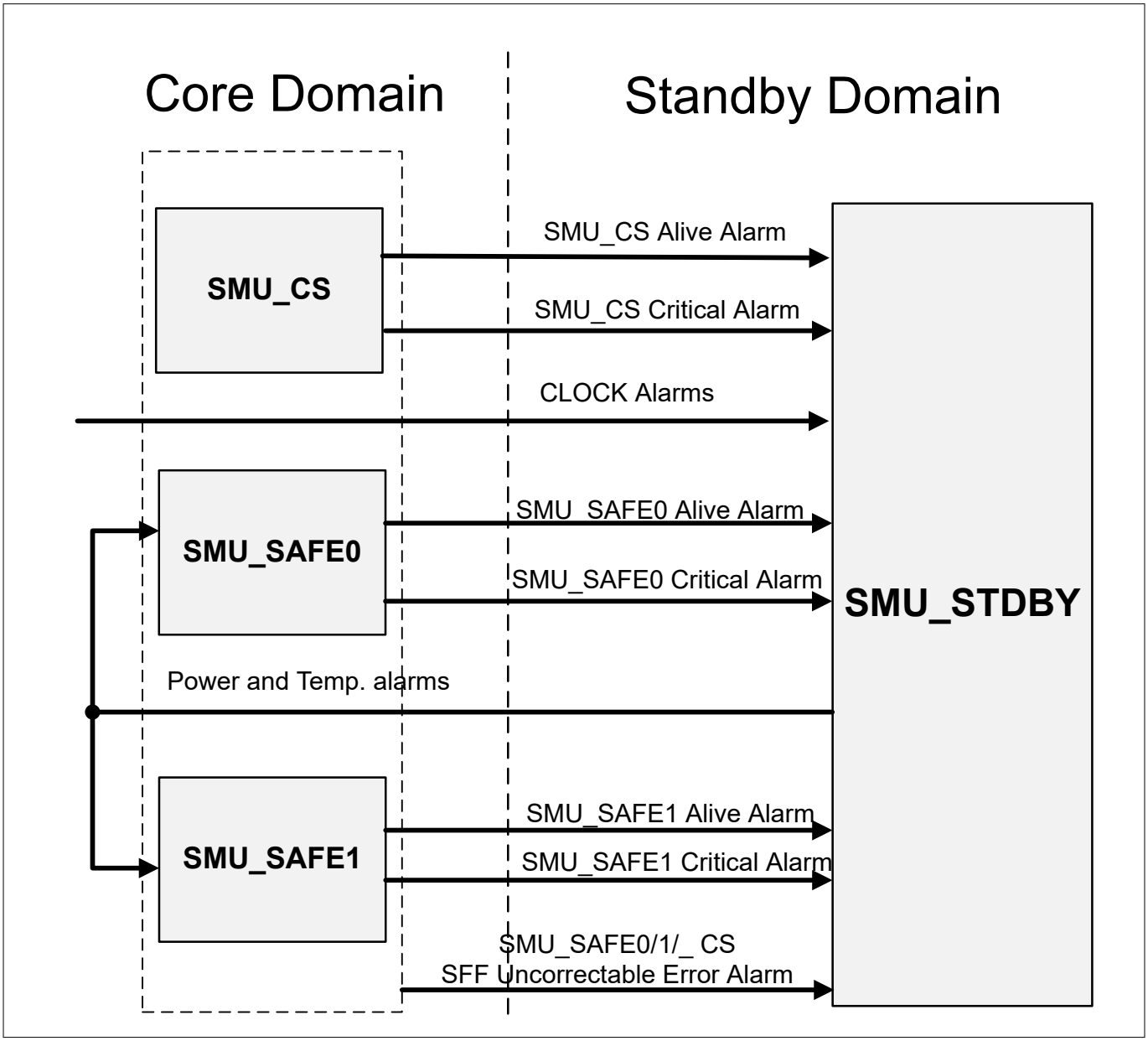


Figure 1006 Interface to SMU_STDBY

42.4 Registers

42.4.1 Registers of SMU

This chapter shows specific information related to the SMU registers excluding the SMU_STDBY registers. The SMU_STDBY registers information is present in the next section.

Related information

Registers of SMU_STANDBY on page 7200

42.4.1.1 Register overview - access mode glossary

Table 1979 Register overview - access mode glossary

Keyword	Description
CE	Access protection using PROT register CS_PROTCE.
CSE	Access protection using PROT register CS_PROTCSE.
GE	Access protection using PROT register GCC_PROTGE.
GSE	Access protection using PROT register GCC_PROTGSE.
SxE (x=0-1)	Access protection using PROT register SAFEx_PROTSE.
SxSE (x=0-1)	Access protection using PROT register SAFEx_PROTSSE.
APU-PCS	Protection group consisting of registers CS_ACCEN_WRA, CS_ACCEN_WRB, CS_ACCEN_RDA, CS_ACCEN_RDB, CS_ACCEN_VM, CS_ACCEN_PRS.
PCS	Access protection using *APU-PCS* registers.
APU-PGCC	Protection group consisting of registers GCC_ACCEN_WRA, GCC_ACCEN_WRB, GCC_ACCEN_RDA, GCC_ACCEN_RDB, GCC_ACCEN_VM, GCC_ACCEN_PRS.
PGCC	Access protection using *APU-PGCC* registers.
APU-PSAFEx (x=0-1)	Protection group consisting of registers SAFEx_ACCEN_WRA, SAFEx_ACCEN_WRB, SAFEx_ACCEN_RDA, SAFEx_ACCEN_RDB, SAFEx_ACCEN_VM, SAFEx_ACCEN_PRS.
PSAFEx	Access protection using *APU-PSAFEx* registers.
BE	Always returns a Bus Error.
32	Access only when using 32-bit width.
U	No access restrictions.
PROT	Access restrictions as defined in the PROT register access rules.

42.4.1.2 Registers overview - SMU (ascending offset address)

Table 1980 Registers overview - SMU (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
OCS	OCDS Control and Status	0004_H	PGCC	SV, PGCC	Debug Reset	7114
ID	Module Identification Register	0008_H	PGCC	BE	Application Reset	7115

(table continues...)

Table 1980 (continued) Registers overview - SMU (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
GCC_KEYS	SMU_GCC Key Register	$000C_H$	PGCC	SV, PGCC, GE, 32	Application Reset	7116
GCC_EMSR	Emergency Stop Register	0010_H	PGCC	SV, PGCC, GE	Application Reset	7116
GCC_EMSSW	Emergency Stop Software set and clear register	0014_H	PGCC	PGCC, SV	Application Reset	7118
GCC_RMi_CTL	Register Monitor Control i	0018_H+i* 12	PGCC	SV, PGCC, GE, 32	Application Reset	7119
GCC_RMi_EF	Register Monitor Error Flags i	$001C_H$+i* 12	PGCC	SV, PGCC, GE, 32	Application Reset	7119
GCC_RMi_STS	Register Monitor Self Test Status i	0020_H+i* 12	PGCC	SV, PGCC, GE, 32	Application Reset	7120
GCC_ALMTST	Alarm Signal Test Register	0058_H	PGCC	SV, PGCC, GE	Application Reset	7120
GCC_AGSHi_SEL	SMU_GCC Shared Alarm Group i Selection Register	$005C_H$+i* 8	PGCC	SV, PGCC, GE, 32	PowerOn Reset	7121
GCC_AGSHi_SS	SMU_GCC Shared Alarm Group i Selection Status	0060_H+i* 8	PGCC	BE	See 7122	7122
GCC_PROTGE	SMU_GCC PROT Register Endinit	0078_H	U	SV, PROT	Application Reset	7123
GCC_PROTGSE	SMU_GCC PROT Register Safe Endinit	$007C_H$	U	SV, PROT	Application Reset	7124
GCC_ACCEN_WRA	SMU_GCC Write access enable register A	0080_H	U	GSE, SV	Application Reset	7126
GCC_ACCEN_WRB	SMU_GCC Write access enable register B	0084_H	U	GSE, SV	Application Reset	7126
GCC_ACCEN_RDA	SMU_GCC Read access enable register A	0088_H	U	GSE, SV	Application Reset	7127
GCC_ACCEN_RDB	SMU_GCC Read access enable register B	$008C_H$	U	GSE, SV	Application Reset	7127
GCC_ACCEN_VM	SMU_GCC VM access enable register	0090_H	U	GSE, SV	Application Reset	7128
GCC_ACCEN_PRS	SMU_GCC PRS access enable register	0094_H	U	GSE, SV	Application Reset	7129
CS_CMD	SMU_CS Command Register	0200_H	PCS	SV, PCS, 32	Application Reset	7129
CS_STS	SMU_CS Status Register	0204_H	PCS	SV, PCS, 32	Application Reset	7130
CS_RTC	SMU_CS Recovery Timer Configuration	0208_H	PCS	SV, PCS, CE, 32	Application Reset	7131
CS_KEYS	SMU_CS Key Register	$020C_H$	PCS	SV, PCS, CE, 32	Application Reset	7132

(table continues...)

Table 1980 (continued) Registers overview - SMU (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CS_DBG	SMU_CS Debug Register	0210_H	PCS	BE	PowerOn Reset	7133
CS_AFCNT	SMU_CS Alarm and Fault Counter	0214_H	PCS	BE	PowerOn Reset	7133
CS_RT0ACA	SMU_CS Recovery Timer 0 Alarm Configuration A	0218_H	PCS	SV, PCS, CE, 32	See 7134	7134
CS_RT0ACB	SMU_CS Recovery Timer 0 Alarm Configuration B	$021C_H$	PCS	SV, PCS, CE, 32	Application Reset	7135
CS_AEX	SMU_CS Alarm Executed Status Register	0220_H	PCS	BE	Application Reset	7136
CS_AEXCLR	SMU_CS Alarm Executed Status Clear Register	0224_H	PCS	SV, PCS, 32	Application Reset	7138
CS_RST	CS_RESET Request Triggering Alarm Information Register	0228_H	PCS	BE	PowerOn Reset	7140
CS_AGCSi_CONj	SMU_CS Group i Alarm Configuration Register j	$022C_H$+i* 18_H+j*4	PCS	SV, PCS, CE, 32	See 7140	7140
CS_AGCSi_STS	SMU_CS Group i Alarm Status Register	0238_H+i* 18_H	PCS	SV, PCS, 32	Application Reset	7141
CS_AGCSi_DBG	SMU_CS Group i Alarm Debug Register	$023C_H$+i* 18_H	PCS	BE	PowerOn Reset	7142
CS_AGCSi_CAS	SMU_CS Group i Critical Alarm Selection Flag	0240_H+i* 18_H	PCS	SV, PCS, CE, 32	Application Reset	7142
CS_AGSHi_CONj	SMU_CS Group i Shared Alarm Configuration Register j	0300_H+i* 20_H+j*4	PCS	SV, PCS, CE, 32	See 7143	7143
CS_AGSHi_STS	SMU_CS Group i Shared Alarm Status Register	$030C_H$+i* 20_H	PCS	SV, PCS, 32	Application Reset	7144
CS_AGSHi_DBG	SMU_CS Group i Shared Alarm Debug Register	0310_H+i* 20_H	PCS	BE	PowerOn Reset	7145
CS_AGSHi_CAS	SMU_CS Group i Critical Shared Alarm Selection Flag	0314_H+i* 20_H	PCS	SV, PCS, CE, 32	Application Reset	7145
CS_AGSHi_SEL	SMU_CS Group i Shared Alarm Group Selection Register	0318_H+i* 20_H	PCS	SV, CE, PCS, 32	See 7146	7146
CS_AGSHi_SS	SMU_CS Group i Shared Alarm Group Selection Status	$031C_H$+i* 20_H	PCS	BE	PowerOn Reset	7147
CS_ALMTST	Alarm Signal Test Register	$03F8_H$	PCS	SV, PCS, CE	Application Reset	7148
CS_PROTCE	SMU_CS PROT Register Endinit	$03FC_H$	U	SV, PROT	Application Reset	7148

(table continues...)

Table 1980 (continued) Registers overview - SMU (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CS_PROTCSE	SMU_CS PROT Register Safe Endinit	0400_H	U	SV, PROT	Application Reset	7150
CS_ACCEN_WRA	SMU_CS Write access enable register A	0404_H	U	CSE, SV	Application Reset	7152
CS_ACCEN_WRB	SMU_CS Write access enable register B	0408_H	U	CSE, SV	Application Reset	7152
CS_ACCEN_RDA	SMU_CS Read access enable register A	$040C_H$	U	CSE, SV	Application Reset	7153
CS_ACCEN_RDB	SMU_CS Read access enable register B	0410_H	U	CSE, SV	Application Reset	7153
CS_ACCEN_VM	SMU_CS VM access enable register	0414_H	U	CSE, SV	Application Reset	7154
CS_ACCEN_PRS	SMU_CS PRS access enable register	0418_H	U	CSE, SV	Application Reset	7154
SAFEx_CMD	SMU_SAFE x Command Register	1000_H+x *500_H	PSAFEx	SV, PSAFEx, 32	Application Reset	7155
SAFEx_STS	SMU_SAFE x Status Register	1004_H+x *500_H	PSAFEx	SV, PSAFEx, 32	Application Reset	7156
SAFEx_FSPi_CON	SMU_SAFE x Fault Signaling Protocol Driver Timing Register i	1008_H+x *500_H+i* 4	PSAFEx	SV, PSAFEx, SxE, 32	PowerOn Reset	7158
SAFEx_PES	SMU_SAFE x Port Emergency Stop Enable Register	1010_H+x *500_H	PSAFEx	SV, PSAFEx, SxE, 32	Application Reset	7162
SAFEx_AGC	SMU_SAFE x Alarm Global Configuration	1014_H+x *500_H	PSAFEx	SV, PSAFEx, SxE, 32	Application Reset	7163
SAFEx_RTC	SMU_SAFE x Recovery Timer Configuration	1018_H+x *500_H	PSAFEx	SV, PSAFEx, SxE, 32	See 7165	7165
SAFEx_KEYS	SMU_SAFE x Key Register	$101C_H$+x *500_H	PSAFEx	SV, PSAFEx, SxE, 32	Application Reset	7166
SAFEx_DBG	SMU_SAFE x Debug Register	1020_H+x *500_H	PSAFEx	BE	PowerOn Reset	7167
SAFEx_PCTL	SMU_SAFE x Port Control	1024_H+x *500_H	PSAFEx	SV, PSAFEx, SxE, 32	PowerOn Reset	7167
SAFEx_AFCNT	SMU_SAFE x Alarm and Fault Counter	1028_H+x *500_H	PSAFEx	BE	PowerOn Reset	7169
SAFEx_RT0ACA	SMU_SAFE x Recovery Timer 0 Alarm Configuration A	1030_H+x *500_H	PSAFEx	SV, PSAFEx, SxE, 32	See 7170	7170
SAFEx_RT0ACB	SMU_SAFE x Recovery Timer 0 Alarm Configuration B	1034_H+x *500_H	PSAFEx	SV, PSAFEx, SxE, 32	See 7171	7171

(table continues...)

Table 1980 (continued) Registers overview - SMU (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
SAFEx_RT1ACA	SMU_SAFE x Recovery Timer 1 Alarm Configuration A	1038_H+x *500_H	PSAFEx	SV, PSAFEx, SxE, 32	See 7173	7173
SAFEx_RT1ACB	SMU_SAFE x Recovery Timer 1 Alarm Configuration B	$103C_H$+x *500_H	PSAFEx	SV, PSAFEx, SxE, 32	See 7174	7174
SAFEx_AEX	SMU_SAFE x Alarm Executed Status Register	1050_H+x *500_H	PSAFEx	BE	Application Reset	7175
SAFEx_AEXCLR	SMU_SAFE x Alarm Executed Status Clear Register	1054_H+x *500_H	PSAFEx	SV, PSAFEx, 32	Application Reset	7180
SAFEx_RST	SMU_SAFE x RESET0/1 Request Triggering Alarm Information Register	1058_H+x *500_H	PSAFEx	BE	PowerOn Reset	7183
SAFEx_AGSFi_CONj	SMU_SAFE x Group i Alarm Configuration Register j	1080_H+x *500_H+i* 24_H+j*4	PSAFEx	SV, PSAFEx, SxE, 32	See 7185	7185
SAFEx_AGSFi_FSPj	SMU_SAFE x Group i FSP j Alarm Configuration Register	1090_H+x *500_H+i* 24_H+j*4	PSAFEx	SV, PSAFEx, SxE, 32	See 7187	7187
SAFEx_AGSFi_STS	SMU_SAFE x Group i Alarm Status Register	1098_H+x *500_H+i* 24_H	PSAFEx	SV, PSAFEx, 32	Application Reset	7188
SAFEx_AGSFi_DBG	SMU_SAFE x Group i Alarm Debug Register	$109C_H$+x *500_H+i* 24_H	PSAFEx	BE	PowerOn Reset	7189
SAFEx_AGSFi_CAS	SMU_SAFE x Group i Critical Alarm Selection Flag	$10A0_H$+x *500_H+i* 24_H	PSAFEx	SV, PSAFEx, SxE, 32	Application Reset	7189
SAFEx_AGSHi_CONj	SMU_SAFE x Group i Shared Alarm Configuration Register j	1300_H+x *500_H+i* 28_H+j*4	PSAFEx	SV, PSAFEx, SxE, 32	Application Reset	7190
SAFEx_AGSHi_FSPj	SMU_SAFE x Group i FSP j Shared Alarm Configuration Register	1310_H+x *500_H+i* 28_H+j*4	PSAFEx	SV, PSAFEx, SxE, 32	Application Reset	7190
SAFEx_AGSHi_STS	SMU_SAFE x Group i Shared Alarm Status Register	1318_H+x *500_H+i* 28_H	PSAFEx	SV, PSAFEx, 32	Application Reset	7191
SAFEx_AGSHi_DBG	SMU_SAFE x Group i Shared Alarm Debug Register	$131C_H$+x *500_H+i* 28_H	PSAFEx	BE	PowerOn Reset	7192
SAFEx_AGSHi_CAS	SMU_SAFE x Group i Critical Shared Alarm Selection Flag	1320_H+x *500_H+i* 28_H	PSAFEx	SV, PSAFEx, SxE, 32	Application Reset	7192

(table continues...)

Table 1980 (continued) Registers overview - SMU (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
SAFEx_AGSHi_SS	SMU_SAFE x Group i Shared Alarm Group Selection Status	1324_H+x *500_H+i* 28_H	PSAFEx	BE	PowerOn Reset	7193
SAFEx_PROTSE	SMU_SAFE x PROT Register Endinit	$13FC_H$+x *500_H	U	SV, PROT	Application Reset	7193
SAFEx_PROTSSE	SMU_SAFE x PROT Register Safe Endinit	1400_H+x *500_H	U	SV, PROT	Application Reset	7195
SAFEx_ACCEN_WRA	SMU_SAFE x Write access enable register A	1404_H+x *500_H	U	SxSE, SV	Application Reset	7197
SAFEx_ACCEN_WRB	SMU_SAFE x Write access enable register B	1408_H+x *500_H	U	SxSE, SV	Application Reset	7197
SAFEx_ACCEN_RDA	SMU_SAFE x Read access enable register A	$140C_H$+x *500_H	U	SxSE, SV	Application Reset	7198
SAFEx_ACCEN_RDB	SMU_SAFE x Read access enable register B	1410_H+x *500_H	U	SxSE, SV	Application Reset	7198
SAFEx_ACCEN_VM	SMU_SAFE x VM access enable register	1414_H+x *500_H	U	SxSE, SV	Application Reset	7199
SAFEx_ACCEN_PRS	SMU_SAFE x PRS access enable register	1418_H+x *500_H	U	SxSE, SV	Application Reset	7199

42.4.1.3 OCDS Control and Status

The OCDS Control and Status register OCS controls the debug and trace behavior by selecting suspend modes and OTGB trigger sets. When OCDS is disabled, the suspend control is ineffective.

OCS Offset address: 0004_H

OCDS Control and Status Debug Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		SUSSTA	SUS_P	SUS				0							
r		rh	w	rw				r							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												TG_P	TGB	TGS	
r												w	rw	rw	

Field	Bits	Type	Description
TGS	1:0	rw	**Trigger Set for OTGB0/1** 00_B No Trigger Set output 01_B TS16_SMU_SAFE0 10_B TS16_SMU_SAFE1 11_B TS16_SMU_CS

(table continues...)

(continued)

Field	Bits	Type	Description
TGB	2	rw	**OTGB0/1 Bus Select** 0_B Trigger Set is output on OTGB0 1_B Trigger Set is output on OTGB1
TG_P	3	w	**TGS, TGB Write Protection** TGS and TGB are only written when TG_P is 1, otherwise unchanged. Read as 0.
SUS	27:24	rw	**OCDS Suspend Control** Controls the sensitivity to the suspend signal coming from the OCDS Trigger Switch (OTGS) 0_H Will not suspend 1_H Hard suspend. Clock is switched off immediately. 2_H Soft suspend. **others,** reserved
SUS_P	28	w	**SUS Write Protection** SUS is only written when SUS_P is 1, otherwise unchanged. Read as 0.
SUSSTA	29	rh	**Suspend State** 0_B Module is not (yet) suspended 1_B Module is suspended
0	23:4, 31:30	r	**Reserved** Read as 0; should be written with 0.

Table 1981 Access mode restrictions of OCS sorted by descending priority

Mode name	Access mode		Description
write 1 to .**TG_P**	rw	TGB, TGS	Set TG_P during write access
write 1 to .**SUS_P**	rw	SUS	Set SUS_P during write access
(default)	r	SUS, TGB, TGS	

42.4.1.4 Module Identification Register

ID Offset address: 0008_H

Module Identification Register Application Reset value: $0089\ C001_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MOD_NUM															
r															

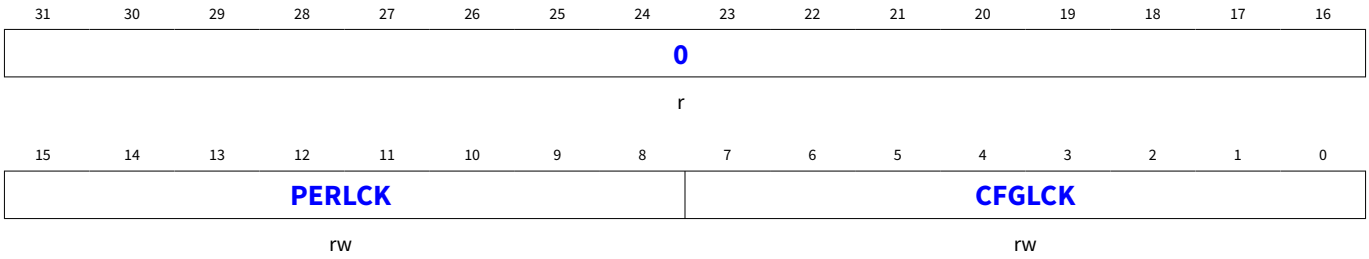
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MOD_TYPE								MOD_REV							
r								r							

Field	Bits	Type	Description
MOD_REV	7:0	r	**Module Revision** Indicates the revision number of the implementation.
MOD_TYPE	15:8	r	**Module Type** The bit field is set to C0H, which defines the module as a 32-bit module.
MOD_NUM	31:16	r	**Module Number Value** This bit field defines a module identification number. The value for the SMU module is 0089H.

42.4.1.5 SMU_GCC Key Register

GCC_KEYS Offset address: $000C_H$

SMU_GCC Key Register Application Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
CFGLCK	7:0	rw	**Configuration Lock** The SMU_GCC configuration is only possible if this field is set to 0xBC.
PERLCK	15:8	rw	**Permanent Lock** If this field is set to 0xFF, no further configuration of the SMU_CS is possible before next application reset.
0	31:16	r	**Reserved** Read as 0; should be written with 0

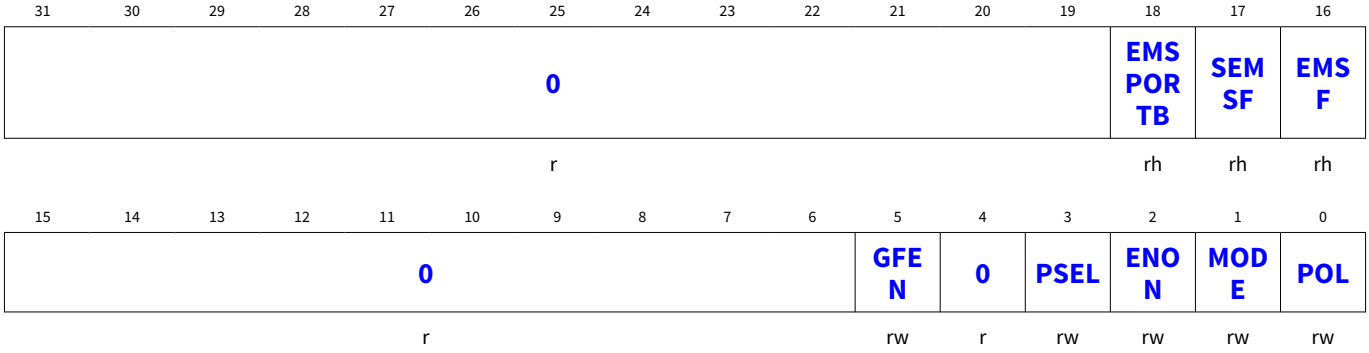
Table 1982 Access mode restrictions of GCC_KEYS sorted by descending priority

Mode name	Access mode		Description
GCC_KEYS.PERLCK != 0xFF	rw	CFGLCK, PERLCK	
(default)	r	CFGLCK, PERLCK	

42.4.1.6 Emergency Stop Register

GCC_EMSR Offset address: 0010_H

Emergency Stop Register Application Reset value: $0000\ 0001_H$



Field	Bits	Type	Description
POL	0	rw	**Input Polarity** This bit determines the polarity of the configured Emergency Stop input. 0_B Input is active high 1_B Input is active low
MODE	1	rw	**Mode Selection** This bit determines the operating mode of the emergency stop signal. 0_B Synchronous Mode selected; emergency stop is derived from the state of flag EMSF 1_B Asynchronous Mode selected; emergency stop is directly derived from the state of the input signal
ENON	2	rw	**Enable ON** This bit enables the setting of flag EMSF by an inactive-to-active level transition of input signal. 0_B Setting of EMSF is disabled 1_B Setting of EMSF is enabled
PSEL	3	rw	**PORT Select** This bit selects which one of the Emergency Stop port options is monitored. 0_B Port A is used as Emergency Stop input 1_B Port B is used as Emergency Stop input.
GFEN	5	rw	**Glitch Filter for PORT B (21.2) to register GCC_EMSSW.EMSPORTB enable** 0_B Glitch Filter disabled 1_B Glitch FIlter enabled
EMSF	16	rh	**Emergency Stop Flag** This bit indicates that a synchronous mode port-triggered emergency stop condition has occurred. 0_B An emergency stop has not occurred 1_B An emergency stop has occurred and emergency stop state becomes active (if MODE = 0)

(table continues...)

(continued)

Field	Bits	Type	Description
SEMSF	17	rh	**SMU Emergency Stop Flag** This bit indicates that an SMU_SAFE0 or SMU_SAFE1 has triggered emergency stop condition. 0_B An emergency stop has not occurred 1_B An emergency stop has occurred and emergency stop state becomes active
EMSPORTB	18	rh	**Emergency Stop External Request Port B (21.2) status** This field capture the value driven on Emergency Stop External Request input PORT B (21.2). This field is updated by hardware every clock cycle, therefore a software clear on write is not meaningful for this field.
0	4, 15:6, 31:19	r	**Reserved** Read as 0; should be written with 0.

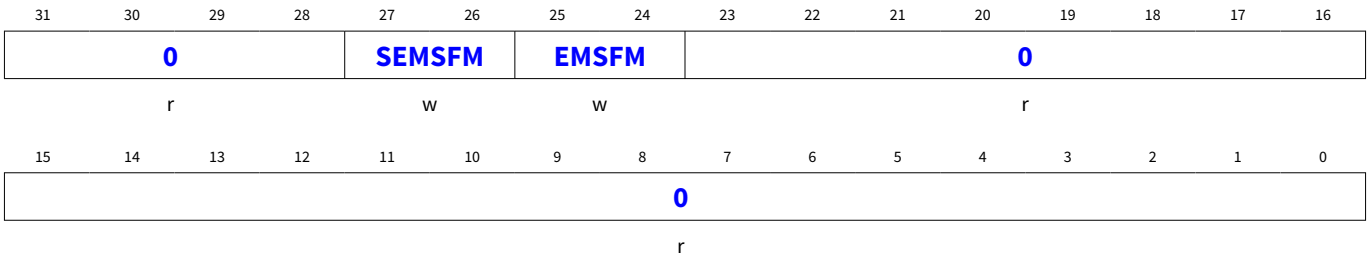
Table 1983 Access mode restrictions of GCC_EMSR sorted by descending priority

Mode name	Access mode		Description
GCC_KEYS.CFGLCK == 0xBC and **GCC_KEYS.PERLCK** != 0xFF	rw	ENON, GFEN, MODE, POL, PSEL	
(default)	r	ENON, GFEN, MODE, POL, PSEL	

42.4.1.7 Emergency Stop Software set and clear register

GCC_EMSSW Offset address: 0014_H

Emergency Stop Software set and clear register Application Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
EMSFM	25:24	w	**Emergency Stop Flag Modification** This bit field sets or clears flag EMSF. EMSFM is always read as 00_B. 00_B EMSF remains unchanged 01_B EMSF becomes set 10_B EMSF becomes cleared 11_B EMSF remains unchanged

(table continues...)

(continued)

Field	Bits	Type	Description
SEMSFM	27:26	w	**SMU Emergency Stop Flag Modification** This bit field sets or clears flag SEMSF. SEMSFM is always read as 00_B. 00_B SEMSF remains unchanged 01_B SEMSF becomes set 10_B SEMSF becomes cleared 11_B SEMSF remains unchanged
0	23:0, 31:28	r	**Reserved** Read as 0; should be written with 0.

42.4.1.8 Register Monitor Control i

GCC_RMi_CTL (i=0-1) Offset address: 0018_H+i*12

Register Monitor Control i Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TE31	TE30	TE29	TE28	TE27	TE26	TE25	TE24	TE23	TE22	TE21	TE20	TE19	TE18	TE17	TE16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TE15	TE14	TE13	TE12	TE11	TE10	TE9	TE8	TE7	TE6	TE5	TE4	TE3	TE2	TE1	TE0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
TEz (z=0-31)	z	rw	**Test Enable** **This bit controls the test of the register monitor safety mechanism. Setting this bit starts the selft-test of the safety flip-flops in the corresponding functional block instance.** 0_B 0 Test mode disabled 1_B 1 Test mode enabled

Table 1984 Access mode restrictions of GCC_RMi_CTL (i=0-1) sorted by descending priority

Mode name	Access mode		Description
GCC_KEYS.CFGLCK == 0xBC and **GCC_KEYS.PERLCK** != 0xFF	rw	TEz (z=0-31)	
(default)	r	TEz (z=0-31)	

42.4.1.9 Register Monitor Error Flags i

GCC_RMi_EF (i=0-1) Offset address: $001C_H$+i*12

Register Monitor Error Flags i Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EF31	EF30	EF29	EF28	EF27	EF26	EF25	EF24	EF23	EF22	EF21	EF20	EF19	EF18	EF17	EF16
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EF15	EF14	EF13	EF12	EF11	EF10	EF9	EF8	EF7	EF6	EF5	EF4	EF3	EF2	EF1	EF0
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
EFz (z=0-31)	z	rwh	**Error flag** **A logical '1' in the error flag reports a real flip flop failure during the register monitor test in the corresponding functional block instance.** This flag can only be cleared by software, a set by software has no effect 0_B Error flag z does not report a fault condition 1_B Error flag z reports a fault condition

42.4.1.10 Register Monitor Self Test Status i

GCC_RMi_STS (i=0-1) Offset address: 0020_H+i*12

Register Monitor Self Test Status i Application Reset value: $0000\ 0000_H$

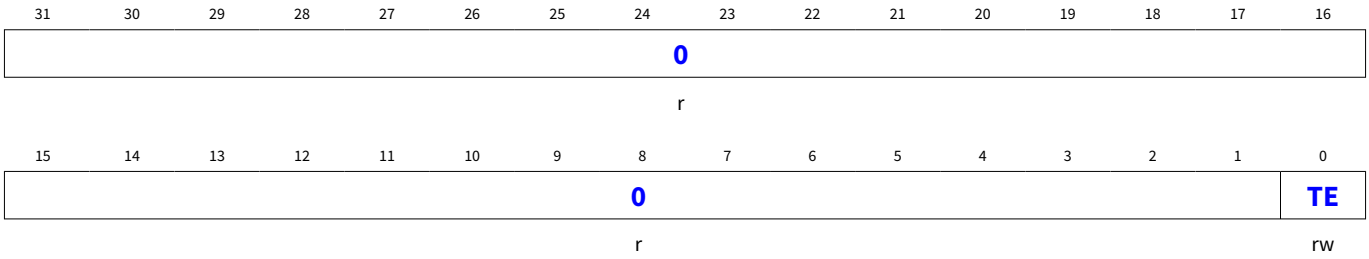
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
STS31	STS30	STS29	STS28	STS27	STS26	STS25	STS24	STS23	STS22	STS21	STS20	STS19	STS18	STS17	STS16
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STS15	STS14	STS13	STS12	STS11	STS10	STS9	STS8	STS7	STS6	STS5	STS4	STS3	STS2	STS1	STS0
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
STSz (z=0-31)	z	rwh	**Status flag** A logical '1' of this bit indicates that the register monitor test has been executed in the corresponding functional block instance. This bit can only be cleared by software, a set by software has no effect. 0_B Self-test has not completed 1_B Self-test has completed

42.4.1.11 Alarm Signal Test Register

GCC_ALMTST Offset address: 0058_H

Alarm Signal Test Register Application Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
TE	0	rw	**Alarm Signal Test Enable** 0_B Test disabled 1_B Test enabled
0	31:1	r	**Reserved** Read as 0; should be written with 0.

42.4.1.12 SMU_GCC Shared Alarm Group i Selection Register

GCC_AGSHi_SEL (i=0-2) Offset address: $005C_H+i*8$

SMU_GCC Shared Alarm Group i Selection Register PowerOn Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CF31	CF30	CF29	CF28	CF27	CF26	CF25	CF24	CF23	CF22	CF21	CF20	CF19	CF18	CF17	CF16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CF15	CF14	CF13	CF12	CF11	CF10	CF9	CF8	CF7	CF6	CF5	CF4	CF3	CF2	CF1	CF0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
CFz (z=0-31)	z	rw	**Shared Alarm Group configuration flag for Alarm Group AGSHi.** The configuration flag is used to map the alarm in the shared group to SMU_SAFE0/1. ***Note**: This flag can only be written when the corresponding bit in CS_AGSHi_SEL.CFz Register is set to '0'.* 0_B Configuration flag z indicates Alarm z is not mapped 1_B Configuration flag z indicates Alarm z is mapped

Table 1985 Access mode restrictions of GCC_AGSHi_SEL (i=0-2) sorted by descending priority

Mode name	Access mode		Description
CS_AGSHi_SEL.CFz == 0 and **GCC_KEYS.CFGLCK** == 0xBC and **GCC_KEYS.PERLCK** != 0xFF	rw	CFz (z=0-31)	
(default)	r	CFz (z=0-31)	

42.4.1.13 SMU_GCC Shared Alarm Group i Selection Status

***Note**: Writing to this register has no effect*

GCC_AGSHi_SS (i=0) Offset address: 0060_H+i*8

SMU_GCC Shared Alarm Group i Selection Status Reset values see: Table 1988

GCC_AGSHi_SS (i=1) Offset address: 0060_H+i*8

SMU_GCC Shared Alarm Group i Selection Status Reset values see: Table 1986

GCC_AGSHi_SS (i=2) Offset address: 0060_H+i*8

SMU_GCC Shared Alarm Group i Selection Status Reset values see: Table 1987

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SF31	**SF30**	**SF29**	**SF28**	**SF27**	**SF26**	**SF25**	**SF24**	**SF23**	**SF22**	**SF21**	**SF20**	**SF19**	**SF18**	**SF17**	**SF16**
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SF15	**SF14**	**SF13**	**SF12**	**SF11**	**SF10**	**SF9**	**SF8**	**SF7**	**SF6**	**SF5**	**SF4**	**SF3**	**SF2**	**SF1**	**SF0**
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
SFz (z=0-31)	z	rh	**Shared Alarm Group status flag for Alarm Group 3i. - CFz** The status flag is used to indicate which alarm in shared alarm group is mapped to SMU_CS. 0_B Status flag z indicates Alarm z is mapped 1_B Status flag z indicates Alarm z is not mapped

Table 1986 Reset values of GCC_AGSHi_SS (i=1)

Reset type	Reset value	Note
PowerOn Reset	FFFF FFFF$_H$	Reset value except GCC_AGSH0_SS, GCC_AGSH2_SS

Table 1987 Reset values of GCC_AGSHi_SS (i=2)

Reset type	Reset value	Note
PowerOn Reset	FFFF BFFF$_H$	Reset value for GCC_AGSH2_SS (VMT4-VMT Uncorrectable Error mapped to SMU_CS)

Table 1988 Reset values of GCC_AGSHi_SS (i=0)

Reset type	Reset value	Note
PowerOn Reset	7FFF FFFF$_H$	Reset value for GCC_AGSH0_SS (CPUcs WDT overflow or access error alarm mapped to SMU_CS)

42.4.1.14 SMU_GCC PROT Register Endinit

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner and in some PROT states, the secure master to update the PROT state as described in the PROT mechanism.

The PROTGE register controls lock / unlock of the local Endinit (GE) protected control registers.

GCC_PROTGE Offset address: 0078$_H$

SMU_GCC PROT Register Endinit Application Reset value: 0000 0000$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OWEN	ODEF	TAGID						PRSEN	PRS			VMEN	VM		
w	rw	rw						rw	rw			rw	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												SWEN	STATE		
r												w	rwh		

Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
VM	18:16	rw	**Virtual Machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual Machine definition Enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition

(table continues...)

(continued)

Field	Bits	Type	Description
PRS	22:20	rw	**Protection Set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.
PRSEN	23	rw	**Protection Set definition Enable for PROT owner** 0_B **Disable**: PRS is part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner
OWEN	31	w	**Owner write enable** 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; should be written with 0.

Table 1989 Access mode restrictions of GCC_PROTGE sorted by descending priority

Mode name	Access mode		Description
write 1 to .**SWEN**	rwh	STATE	
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

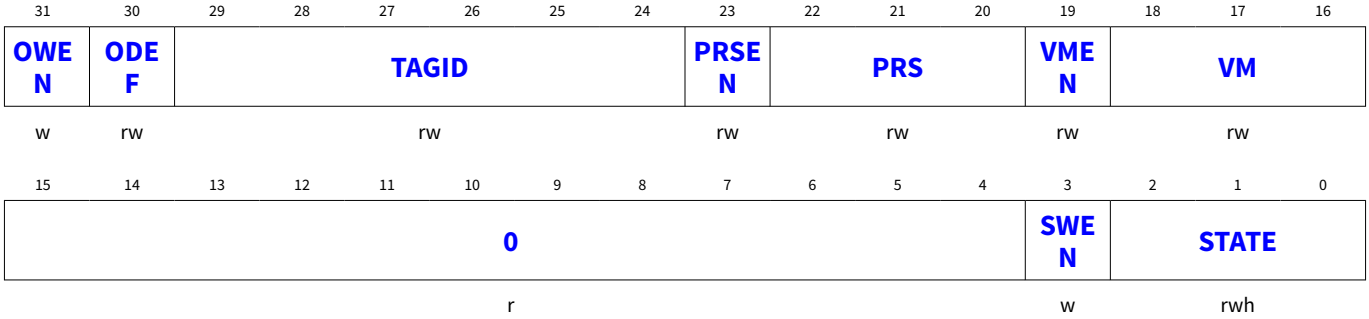
42.4.1.15 SMU_GCC PROT Register Safe Endinit

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner and in some PROT states, the secure master to update the PROT state as described in the PROT mechanism.

The PROTGSE register controls lock / unlock of the local Safe Endinit (GSE) protected control registers.

GCC_PROTGSE Offset address: $007C_H$

SMU_GCC PROT Register Safe Endinit Application Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
VM	18:16	rw	**Virtual Machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual Machine definition Enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection Set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.
PRSEN	23	rw	**Protection Set definition Enable for PROT owner** 0_B **Disable**: PRS is part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner

(table continues...)

(continued)

Field	Bits	Type	Description
OWEN	31	w	**Owner write enable** 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; should be written with 0.

Table 1990 Access mode restrictions of GCC_PROTGSE sorted by descending priority

Mode name	Access mode		Description
write 1 to .**SWEN**	rwh	STATE	
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

42.4.1.16 SMU_GCC Write access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

GCC_ACCEN_WRA Offset address: 0080_H

SMU_GCC Write access enable register A Application Reset value: $1000\ 0003_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	**EN30**	**EN29**	**EN28**	**EN27**	**EN26**	**EN25**	**EN24**	**EN23**	**EN22**	**EN21**	**EN20**	**EN19**	**EN18**	**EN17**	**EN16**
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	**EN14**	**EN13**	**EN12**	**EN11**	**EN10**	**EN09**	**EN08**	**EN07**	**EN06**	**EN05**	**EN04**	**EN03**	**EN02**	**EN01**	**EN00**
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access

42.4.1.17 SMU_GCC Write access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

GCC_ACCEN_WRB Offset address: 0084_H

SMU_GCC Write access enable register B Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access
0	31:8	r	**Reserved** Read as 0; should be written with 0.

42.4.1.18 SMU_GCC Read access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

GCC_ACCEN_RDA Offset address: 0088_H

SMU_GCC Read access enable register A Application Reset value: FFFF FFFF$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

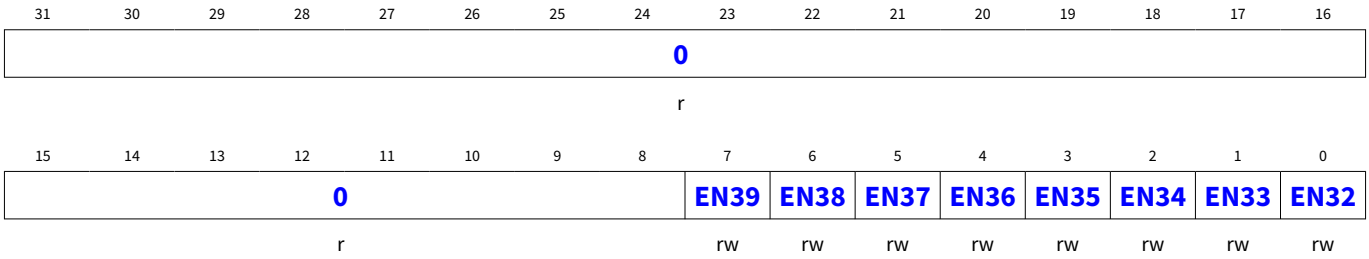
Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access

42.4.1.19 SMU_GCC Read access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

GCC_ACCEN_RDB Offset address: $008C_H$

SMU_GCC Read access enable register B Application Reset value: 0000 00FF$_H$



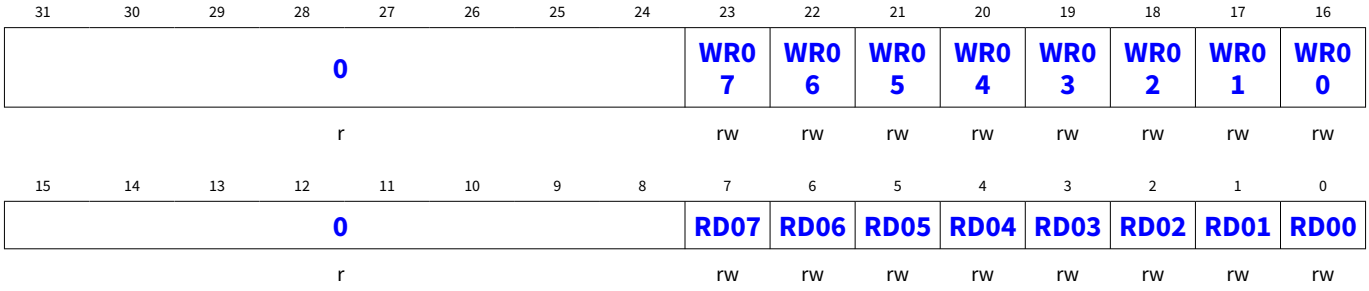
Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access
0	31:8	r	**Reserved** Read as 0; should be written with 0.

42.4.1.20 SMU_GCC VM access enable register

This register defines which virtual machine encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each VMx.

GCC_ACCEN_VM Offset address: 0090_H

SMU_GCC VM access enable register Application Reset value: $00FF\ 00FF_H$



Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for VM ID q** This bit enables read access to the access protected region for transactions with the VM ID q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for VM ID q** This bit enables write access to the access protected region for transactions with the VM ID q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

42.4.1.21 SMU_GCC PRS access enable register

This register defines which protection register set encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each PRSx.

GCC_ACCEN_PRS Offset address: 0094_H

SMU_GCC PRS access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for PRS q** This bit enables read access to the access protected region for transactions with the PRS q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for PRS q** This bit enables write access to the access protected region for transactions with the PRS q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

42.4.1.22 SMU_CS Command Register

CS_CMD Offset address: 0200_H

SMU_CS Command Register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								ARG				CMD			
r								w				w			

Field	Bits	Type	Description
CMD	3:0	w	**Implements the SMU_CS Command Interface.** Command Encoding defined in SMU_CS control interface Read as 0.
ARG	7:4	w	**Implements the SMU_CS Command Interface.** Argument to be used with the command. Command Encoding defined in SMU_CS control interface. Read as 0.
0	31:8	r	**Reserved** Read as 0; should be written with 0

42.4.1.23 SMU_CS Status Register

***Note**: A write to this register (regardless of the data written) clears all the fields with the rwh type. If on the same cycle a software write event and hardware event is detected, the hardware action wins.*

CS_STS Offset address: 0204_H

SMU_CS Status Register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0														RTME0	RTS0
r														rwh	rwh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0						ASCE	RES	ARG				CMD			
r						rwh	rwh	rwh				rwh			

Field	Bits	Type	Description
CMD	3:0	rwh	**Last command received** Same encoding as CMD field of CS_CMD register
ARG	7:4	rwh	**Last command argument received** Same encoding as ARG field of CS_CMD register
RES	8	rwh	**Result of last received command** 0_B Command was successful 1_B Command failed

(table continues...)

(continued)

Field	Bits	Type	Description
ASCE	9	rwh	**Alarm Status Clear Enable** This bit controls if a status flag set in an CS_AGCSi_STS register upon detection of an alarm event can be cleared by software or not. When ASCE is enabled software shall write a 1 to the bit position in CS_AGCSi_STS to clear the bit. When CS_AGCSi_STS is cleared, the ASCE bit is automatically cleared to 0 by hardware and software shall set the ASCE bit again by using the SMU_ASCE() command. 0_B Alarm status bits CSAG3*i* can not be cleared 1_B Alarm status bits CSAG3*i* can be cleared.
RTS0	16	rwh	**Recovery Timer 0 Status** State of the recovery timer 0 0_B Recovery Timer not running 1_B Recovery Timer running
RTME0	17	rwh	**Recovery Timer 0 Missed Event** Recovery timer trigger request missed 0_B Recovery Timer event not detected 1_B Recovery Timer event detected
0	15:10, 31:18	r	**Reserved** Read as 0; should be written with 0.

42.4.1.24 SMU_CS Recovery Timer Configuration

This register controls the timing duration of the recovery timer.

CS_RTC Offset address: 0208_H

SMU_CS Recovery Timer Configuration Application Reset value: 003F FF01$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RTD															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RTD								0			PRE			0	RT0E
rw								r			rw			r	rw

Field	Bits	Type	Description
RT0E	0	rw	**RT0 Enable Bit** 0_B Recovery Timer 0 is disabled 1_B Recovery Timer 0 is enabled

(table continues...)

(continued)

Field	Bits	Type	Description
PRE	4:2	rw	**Prescaler For Recovery Timer Clock** Dividing factor to apply to the reference clock f_{BACK}. It is assumed that the maximum value for f_{BACK} is 100 Mhz with a precision of 5%. The divided clock is used as reference clock for recovery timer. The frequency of the divided clock is defined as follows: 000_B reference clock frequency divided by 2 001_B reference clock frequency divided by 4 010_B reference clock frequency divided by 8 011_B reference clock frequency divided by 16 100_B reference clock frequency divided by 32 101_B reference clock frequency divided by 64 110_B reference clock frequency divided by 128 111_B reference clock frequency divided by 256
RTD	31:8	rw	**Recovery Timer Duration** This field specifies the maximum duration of the recovery timer. When the timer counter reaches the programmed value, the internal time-out alarm is triggered. The timer is stopped by a SMU_RTStop() command before the recovery timer timeout occurs. RTD shall be specified as a number of the f_{BACK}/CS_RTC.PRE clock ticks. ***Note**: It is not allowed to write to the RTD when at least one recovery timer is running.*
0	1, 7:5	r	**Reserved** Read as 0; should be written with 0

Table 1991 Access mode restrictions of CS_RTC sorted by descending priority

Mode name	Access mode		Description
CS_KEYS.CFGLCK == 0xBC and **CS_KEYS.PERLCK** ! = 0xFF	rw	PRE, RT0E, RTD	
(default)	r	PRE, RT0E, RTD	

42.4.1.25 SMU_CS Key Register

CS_KEYS Offset address: $020C_H$

SMU_CS Key Register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PERLCK								CFGLCK							
rw								rw							

Field	Bits	Type	Description
CFGLCK	7:0	rw	**Configuration Lock** The SMU_CS configuration is only possible if this field is set to 0xBC.
PERLCK	15:8	rw	**Permanent Lock** If this field is set to 0xFF, no further configuration of the SMU_CS is possible before next application reset.
0	31:16	r	**Reserved** Read as 0; should be written with 0

Table 1992 Access mode restrictions of CS_KEYS sorted by descending priority

Mode name	Access mode		Description
CS_KEYS.PERLCK != 0xFF	rw	CFGLCK, PERLCK	
(default)	r	CFGLCK, PERLCK	

42.4.1.26 SMU_CS Debug Register

This register enables to observe some internal states of the SMU_CS hardware.

CS_DBG Offset address: 0210_H

SMU_CS Debug Register PowerOn Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

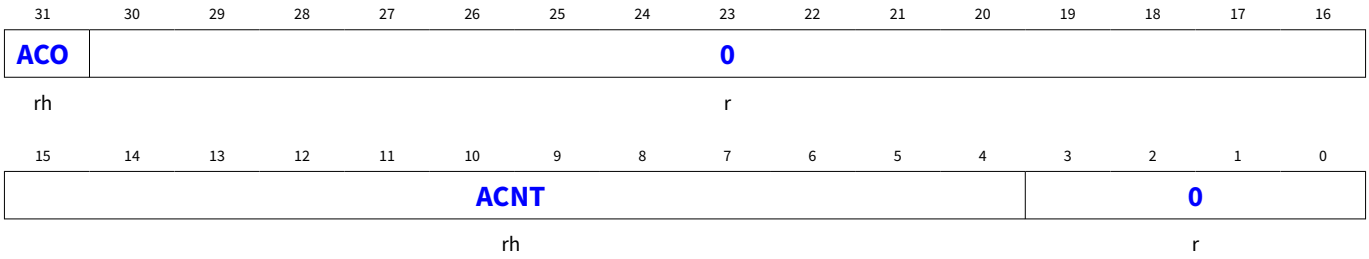
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															SSM
r															rh

Field	Bits	Type	Description
SSM	0	rh	**Running state of the SMU_CS state machine** 0_B START state 1_B RUN state
0	31:1	r	**Reserved** Read as 0; should be written with 0

42.4.1.27 SMU_CS Alarm and Fault Counter

CS_AFCNT Offset address: 0214_H

SMU_CS Alarm and Fault Counter PowerOn Reset value: $0000\ 0000_H$



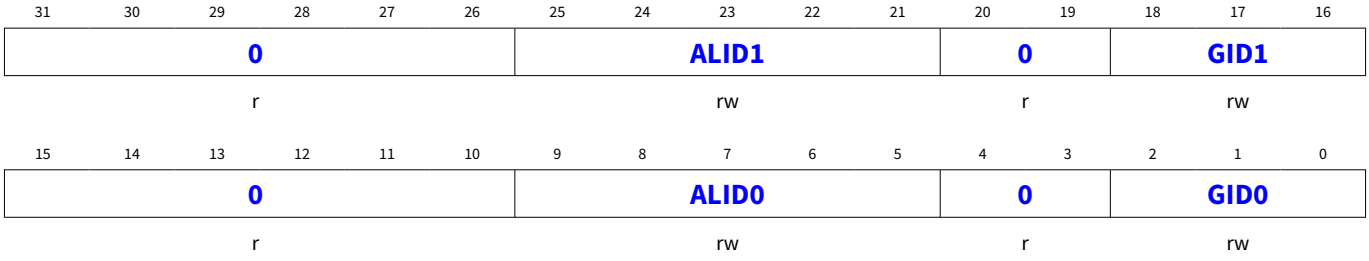
Field	Bits	Type	Description
ACNT	15:4	rh	**Alarm Counter.** This field is incremented by hardware when the SMU_CS processes an internal reaction related to an alarm event. The counter value is kept stable when the maximum value is reached.
ACO	31	rh	**Alarm Counter Overflow.** This bit is set by hardware if the ACNT counter reached the maximum value and an increment condition is present.
0	3:0, 30:16	r	**Reserved** Read as 0; should be written with 0.

42.4.1.28 SMU_CS Recovery Timer 0 Alarm Configuration A

***Note**: It is possible to configure multiple times the same group identifier in GID0/1 fields.*

CS_RT0ACA Offset address: 0218_H

SMU_CS Recovery Timer 0 Alarm Configuration A Reset values see: Table 1994



Field	Bits	Type	Description
GID0	2:0	rw	**Group Index 0.** This field enables to specify if an alarm from this alarm group can use the recovery timer 0.
ALID0	9:5	rw	**Alarm Identifier 0.** This field specifies the alarm index related to the group index specified in GID0.
GID1	18:16	rw	**Group Index 1.** This field enables to specify if an alarm from this alarm group can use the recovery timer 0.

(table continues...)

(continued)

Field	Bits	Type	Description
ALID1	25:21	rw	**Alarm Identifier 1.** This field specifies the alarm index related to the group index specified in GID1.
0	4:3, 15:10, 20:19, 31:26	r	**Reserved** Read as 0; should be written with 0.

Table 1993 Access mode restrictions of CS_RT0ACA sorted by descending priority

Mode name	Access mode		Description
CS_KEYS.CFGLCK == 0xBC and **CS_KEYS.PERLCK** ! = 0xFF	rw	ALID0, ALID1, GID0, GID1	
(default)	r	ALID0, ALID1, GID0, GID1	

Table 1994 Reset values of CS_RT0ACA

Reset type	Reset value	Note
Application Reset	03E2 03E2$_H$	Recovery Timer 0 to CPUcs WDT overflow or access error alarm

42.4.1.29 SMU_CS Recovery Timer 0 Alarm Configuration B

***Note**: It is possible to configure multiple times the same group identifier in GID2/3 fields.*

CS_RT0ACB Offset address: 021C$_H$

SMU_CS Recovery Timer 0 Alarm Configuration B Application Reset value: 0000 0000$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0						ALID3					0		GID3		
r						rw					r		rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0						ALID2					0		GID2		
r						rw					r		rw		

Field	Bits	Type	Description
GID2	2:0	rw	**Group Index 2.** This field enables to specify if an alarm from this alarm group can use the recovery timer 0.
ALID2	9:5	rw	**Alarm Identifier 0.** This field specifies the alarm index related to the group index specified in GID2.

(table continues...)

(continued)

Field	Bits	Type	Description
GID3	18:16	rw	**Group Index 3.** This field enables to specify if an alarm from this alarm group can use the recovery timer 3.
ALID3	25:21	rw	**Alarm Identifier 1.** This field specifies the alarm index related to the group index specified in GID3.
0	4:3, 15:10, 20:19, 31:26	r	**Reserved** Read as 0; should be written with 0.

Table 1995 Access mode restrictions of CS_RT0ACB sorted by descending priority

Mode name	Access mode		Description
CS_KEYS.CFGLCK == 0xBC and **CS_KEYS.PERLCK** ! = 0xFF	rw	ALID2, ALID3, GID2, GID3	
(default)	r	ALID2, ALID3, GID2, GID3	

42.4.1.30 SMU_CS Alarm Executed Status Register

The Alarm Executed Status Register is used to show, which alarm mechanisms are executed.

CS_AEX Offset address: 0220_H

SMU_CS Alarm Executed Status Register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LKAEM	**LDKAEM**	**0**			**Res**	**NMIAEM**	**0**								**IRQ0AEM**
rh	rh	r			r	rh	r								rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LKSTS	**LDKSTS**	**0**			**RSTSTS**	**NMISTS**	**0**								**IRQ0STS**
rh	rh	r			rh	rh	r								rh

Field	Bits	Type	Description
IRQ0STS	0	rh	**CSIRQ Request Status** This bit indicates whether a CSIRQ request was serviced or not. This bit is set by the SMU_CS after an alarm configured for CSIRQ is detected. This bit can be cleared by writing with ´1´ to the IRQ0C bit in the CS_AEXCLR register. 0_B NO CSIRQ request was serviced 1_B CSIRQ request was serviced

(table continues...)

(continued)

Field	Bits	Type	Description
NMISTS	9	rh	**CS_NMI Request Status** This bit indicates whether a CS_NMI request was serviced or not. This bit is set by the SMU_CS after an alarm configured for CS_NMI is detected. This bit can be cleared by writing with ´1´ to the NMIC bit in the related CS_AEXCLR register. 0_B NO CS_NMI request was serviced 1_B CS_NMI request was serviced
RSTSTS	10	rh	**CS_RESET Request Status** This bit indicates whether a CS_RESET request was serviced or not. This bit is set by the SMU_CS after an alarm configured for CS_RESET is detected. 0_B NO CS_RESET request was serviced 1_B CS_RESET request was serviced
LDKSTS	14	rh	**LOCK_DBG_PROT_KEYS Request Status** This bit indicates whether a LOCK_DBG_PROT_KEYS request was serviced or not. This bit is set by the SMU_CS after an alarm configured for LOCK_DBG_PROT_KEYS is detected. This bit can be cleared by writing with ´1´ to the LDKC bit in the CS_AEXCLR register. 0_B NO LOCK_DBG_PROT_KEYS request was serviced 1_B LOCK_DBG_PROT_KEYS request was serviced
LKSTS	15	rh	**LOCK_ALL_KEYS Request Status** This bit indicates whether a LOCK_ALL_KEYS request was serviced or not. This bit is set by the SMU_CS after an alarm configured for LOCK_ALL_KEYS is detected. This bit can be cleared by writing with ´1´ to the LKC bit in the related CS_AEXCLR register. 0_B NO LOCK_ALL_KEYS request was serviced 1_B LOCK_ALL_KEYS request was serviced
IRQ0AEM	16	rh	**CSIRQ AEM** This bit indicates that an alarm event is missed. That means that an alarm has occurred with configuration for CSIRQ Request where this alarm handler was blocked because CS_AEX.IRQ0STS was already set. This bit can be cleared by writing with ´1´ to the IRQ0AEMC bit in the CS_AEXCLR register. 0_B No alarm is missed 1_B At least one alarm is missed

(table continues...)

(continued)

Field	Bits	Type	Description
NMIAEM	25	rh	**CS_NMI AEM** This bit indicates that an alarm event is missed. That means that an alarm has occurred with configuration for CS_NMI Request where this alarm handler was blocked because CS_AEX.NMISTS was already set. This bit can be cleared by writing with ´1´ to the NMIAEMC bit in the CS_AEXCLR register. 0_B No alarm is missed 1_B At least one alarm is missed
LDKAEM	30	rh	**LOCK_DBG_PROT_KEYS AEM** This bit indicates that an alarm event is missed. That means that an alarm has occurred with configuration for LOCK_DBG_PROT_KEYS Request where this alarm handler was blocked because CS_AEX.LDKSTS was already set. This bit can be cleared by writing with ´1´ to the LDKAEMC bit in the CS_AEXCLR register. 0_B No alarm is missed 1_B At least one alarm is missed
LKAEM	31	rh	**LOCK_ALL_KEYS AEM** This bit indicates that an alarm event is missed. That means that an alarm has occurred with configuration for LOCK_ALL_KEYS Request where this alarm handler was blocked because CS_AEX.LKSTS was already set. This bit can be cleared by writing with ´1´ to the LKAEMC bit in the CS_AEXCLR register. 0_B No alarm is missed 1_B At least one alarm is missed
0	8:1, 13:11, 24:17, 29:27	r	**Reserved** Read as 0; should be written with 0.

42.4.1.31 SMU_CS Alarm Executed Status Clear Register

The Alarm Executed Status Clear register is used to clear the Alarm Executed Status(CS_AEX) register.

CS_AEXCLR Offset address: 0224_H

SMU_CS Alarm Executed Status Clear Register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LKAEMC	LDKAEMC	0			Res	NMIAEMC	0								IRQ0AEMC
w	w	r			r	w	r								w

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LKC	LDKC	0			Res	NMIC	0								IRQ0C
w	w	r			r	w	r								w

Field	Bits	Type	Description
IRQ0C	0	w	**IRQ0STS Clear** Read always as 0. 0_B No action 1_B Clear CS_AEX.IRQ0STS status bit.
NMIC	9	w	**NMISTS Clear** Read always as 0. 0_B No action 1_B Clear CS_AEX.NMISTS status bit.
LDKC	14	w	**LOCK_DBG_PROT_KEYS Request Status Clear** Read always as 0. 0_B No action 1_B Clear CS_AEX.LDKSTS status bit.
LKC	15	w	**LOCK_ALL_KEYS Request Status Clear** Read always as 0. 0_B No action 1_B Clear CS_AEX.LKSTS status bit.
IRQ0AEMC	16	w	**IRQ0AEM Clear** Read always as 0. 0_B No action 1_B Clear CS_AEX.IRQ0AEM status bit.
NMIAEMC	25	w	**NMIAEM Clear** Read always as 0. 0_B No action 1_B Clear CS_AEX.NMIAEM status bit.
LDKAEMC	30	w	**LDKAEM Clear** Read always as 0. 0_B No action 1_B Clear CS_AEX.LDKAEM status bit.
LKAEMC	31	w	**LKAEM Clear** Read always as 0. 0_B No action 1_B Clear CS_AEX.LKAEM status bit.
0	8:1, 13:11, 24:17, 29:27	r	**Reserved** Read as 0; should be written with 0.

42.4.1.32 CS_RESET Request Triggering Alarm Information Register

***Note**: Writing to this register has no effect*

CS_RST Offset address: 0228_H

CS_RESET Request Triggering Alarm Information Register PowerOn Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			ALMID					0					ALMGRP		
r			rh					r					rh		

Field	Bits	Type	Description
ALMGRP	2:0	rh	**Alarm Group** This field indicates the alarm group of the alarm that triggered the CS_RESET request.
ALMID	12:8	rh	**Alarm Index** This field specifies the alarm index related to the alarm group index ALMGRP which has triggered CS_RESET request.
0	7:3, 31:13	r	**Reserved** Read as 0; should be written with 0.

42.4.1.33 SMU_CS Group i Alarm Configuration Register j

CS_AGCSi_CONj (i=0;j=0,2) Offset address: $022C_H+i*18_H+j*4$

SMU_CS Group i Alarm Configuration Register j Reset values see: Table 1999

CS_AGCSi_CONj (i=0;j=1) Offset address: $022C_H+i*18_H+j*4$

SMU_CS Group i Alarm Configuration Register j Reset values see: Table 1997

CS_AGCSi_CONj (i=1;j=0) Offset address: $022C_H+i*18_H+j*4$

SMU_CS Group i Alarm Configuration Register j Reset values see: Table 1997

CS_AGCSi_CONj (i=1;j=1-2) Offset address: $022C_H+i*18_H+j*4$

SMU_CS Group i Alarm Configuration Register j Reset values see: Table 1998

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CF31	CF30	CF29	CF28	CF27	CF26	CF25	CF24	CF23	CF22	CF21	CF20	CF19	CF18	CF17	CF16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CF15	CF14	CF13	CF12	CF11	CF10	CF9	CF8	CF7	CF6	CF5	CF4	CF3	CF2	CF1	CF0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
CFz (z=0-31)	z	rw	**Configuration flag j (j=0-2) for alarm z in alarm group AGCSi.** The configuration flags 0, 1 and 2 must be used together to define the behavior of the SMU_CS when a fault state is reported by the alarm z belonging to this group. 0_B Configuration flag j (j=0-2) is set to 0 1_B Configuration flag j (j=0-2) is set to 1

Table 1996 Access mode restrictions sorted by descending priority

Applies to CS_AGCSi_CONj (i=0;j=0,2)
Applies to CS_AGCSi_CONj (i=0;j=1)
Applies to CS_AGCSi_CONj (i=1;j=0)
Applies to CS_AGCSi_CONj (i=1;j=1-2)

Mode name	Access mode		Description
CS_KEYS.CFGLCK == 0xBC and **CS_KEYS.PERLCK** ! = 0xFF	rw	CFz (z=0-31)	
(default)	r	CFz (z=0-31)	

Table 1997 Reset values variant 1

Applies to CS_AGCSi_CONj (i=0;j=1)
Applies to CS_AGCSi_CONj (i=1;j=0)

Reset type	Reset value	Note
Application Reset	0000 0000_H	Reset value for all alarm configuration registers except CS_AGCS1_CON1, CS_AGCS1_CON2, CS_AGCS0_CON0 and CS_AGCS0_CON2

Table 1998 Reset values of CS_AGCSi_CONj (i=1;j=1-2)

Reset type	Reset value	Note
Application Reset	0004 0000_H	Reset value for CS_AGCS1_CON1 and CS_AGCS1_CON2 (CS_RESET_REQ to Recovery Timer Timeout alarm)

Table 1999 Reset values of CS_AGCSi_CONj (i=0;j=0,2)

Reset type	Reset value	Note
Application Reset	0000 2000_H	Reset value for CS_AGCS0_CON0 and CS_AGCS0_CON2 (CS_NMI to CSS SECURED_DOMAIN_ALARM)

42.4.1.34 SMU_CS Group i Alarm Status Register

Refer to SMU_CS Alarm Status Registers section for the conditions to set and reset the status flag by software.

CS_AGCSi_STS (i=0-1) Offset address: 0238_H+i*18_H

SMU_CS Group i Alarm Status Register Application Reset value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SF31	SF30	SF29	SF28	SF27	SF26	SF25	SF24	SF23	SF22	SF21	SF20	SF19	SF18	SF17	SF16
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SF15	SF14	SF13	SF12	SF11	SF10	SF9	SF8	SF7	SF6	SF5	SF4	SF3	SF2	SF1	SF0
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
SFz (z=0-31)	z	rwh	**Status flag for alarm z in alarm group AGCSi.** 0_B Status flag z does not report a fault condition 1_B Status flag z reports a fault condition

42.4.1.35 SMU_CS Group i Alarm Debug Register

***Note**: Writing to this register has no effect*

CS_AGCSi_DBG (i=0-1) Offset address: $023C_H+i*18_H$

SMU_CS Group i Alarm Debug Register PowerOn Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DF31	DF30	DF29	DF28	DF27	DF26	DF25	DF24	DF23	DF22	DF21	DF20	DF19	DF18	DF17	DF16
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DF15	DF14	DF13	DF12	DF11	DF10	DF9	DF8	DF7	DF6	DF5	DF4	DF3	DF2	DF1	DF0
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
DFz (z=0-31)	z	rh	**Diagnosis flag for alarm z in alarm group AGCSi.** The diagnosis registers make a snapshot of the alarm group status registers when either the executed alarm reaction is a reset or a state machine transition to FAULT state takes place. 0_B Status flag z does not report a fault condition 1_B Status flag z reports a fault condition

42.4.1.36 SMU_CS Group i Critical Alarm Selection Flag

CS_AGCSi_CAS (i=0-1) Offset address: 0240_H+i*18_H

SMU_CS Group i Critical Alarm Selection Flag Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CF31	CF30	CF29	CF28	CF27	CF26	CF25	CF24	CF23	CF22	CF21	CF20	CF19	CF18	CF17	CF16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CF15	CF14	CF13	CF12	CF11	CF10	CF9	CF8	CF7	CF6	CF5	CF4	CF3	CF2	CF1	CF0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
CFz (z=0-31)	z	rw	**Critical Alarm Selection flag for alarm z in alarm group AGCSi.** 0_B Critical alarm disabled for this alarm event 1_B Critical alarm enabled for this alarm event

Table 2000 Access mode restrictions of CS_AGCSi_CAS (i=0-1) sorted by descending priority

Mode name	Access mode		Description
CS_KEYS.CFGLCK == 0xBC and **CS_KEYS.PERLCK** ! = 0xFF	rw	CFz (z=0-31)	
(default)	r	CFz (z=0-31)	

42.4.1.37 SMU_CS Group i Shared Alarm Configuration Register j

CS_AGSHi_CONj (i=0;j=0,2) Offset address: 0300_H+i*20_H+j*4

SMU_CS Group i Shared Alarm Configuration Register j Reset values see: Table 2004

CS_AGSHi_CONj (i=0,2;j=1) Offset address: 0300_H+i*20_H+j*4

SMU_CS Group i Shared Alarm Configuration Register j Reset values see: Table 2002

CS_AGSHi_CONj (i=1;j=0-2) Offset address: 0300_H+i*20_H+j*4

SMU_CS Group i Shared Alarm Configuration Register j Reset values see: Table 2002

CS_AGSHi_CONj (i=2;j=0,2) Offset address: 0300_H+i*20_H+j*4

SMU_CS Group i Shared Alarm Configuration Register j Reset values see: Table 2003

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CF31	CF30	CF29	CF28	CF27	CF26	CF25	CF24	CF23	CF22	CF21	CF20	CF19	CF18	CF17	CF16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CF15	CF14	CF13	CF12	CF11	CF10	CF9	CF8	CF7	CF6	CF5	CF4	CF3	CF2	CF1	CF0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
CFz (z=0-31)	z	rw	**Configuration flag j (j=0-2) for alarm z in alarm group AGSHi.** The configuration flags 0, 1 and 2 must be used together to define the behavior of the SMU_CS when a fault state is reported by the alarm z belonging to this group. 0_B Configuration flag j (j=0-2) is set to 0 1_B Configuration flag j (j=0-2) is set to 1

Table 2001 Access mode restrictions sorted by descending priority

Applies to CS_AGSHi_CONj (i=0;j=0,2)
Applies to CS_AGSHi_CONj (i=0,2;j=1)
Applies to CS_AGSHi_CONj (i=1;j=0-2)
Applies to CS_AGSHi_CONj (i=2;j=0,2)

Mode name	Access mode		Description
CS_KEYS.CFGLCK == 0xBC and **CS_KEYS.PERLCK** != 0xFF	rw	CFz (z=0-31)	
(default)	r	CFz (z=0-31)	

Table 2002 Reset values variant 1

Applies to CS_AGSHi_CONj (i=0,2;j=1)
Applies to CS_AGSHi_CONj (i=1;j=0-2)

Reset type	Reset value	Note
Application Reset	0000 0000_H	Reset value for all alarm configuration registers except CS_AGSH0_CON0, CS_AGSH0_CON2, CS_AGSH2_CON0 and CS_AGSH2_CON2

Table 2003 Reset values of CS_AGSHi_CONj (i=2;j=0,2)

Reset type	Reset value	Note
Application Reset	0000 4000_H	Reset value for CS_AGSH2_CON0 and CS_AGSH2_CON2 (CS_NMI to VMT4-VMT Uncorrectable Error)

Table 2004 Reset values of CS_AGSHi_CONj (i=0;j=0,2)

Reset type	Reset value	Note
Application Reset	8000 0000_H	Reset value for CS_AGSH0_CON0 and CS_AGSH0_CON2 (CS_NMI to CPUcs WDT overflow or access error alarm)

42.4.1.38 SMU_CS Group i Shared Alarm Status Register

Refer to SMU_CS Alarm Status Registers section for the conditions to set and reset the status flag by software.

CS_AGSHi_STS (i=0-2) Offset address: $030C_H+i*20_H$

SMU_CS Group i Shared Alarm Status Register Application Reset value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SF31	SF30	SF29	SF28	SF27	SF26	SF25	SF24	SF23	SF22	SF21	SF20	SF19	SF18	SF17	SF16
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SF15	SF14	SF13	SF12	SF11	SF10	SF9	SF8	SF7	SF6	SF5	SF4	SF3	SF2	SF1	SF0
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
SFz (z=0-31)	z	rwh	**Status flag for alarm z in alarm group AGSHi.** 0_B Status flag z does not report a fault condition 1_B Status flag z reports a fault condition

42.4.1.39 SMU_CS Group i Shared Alarm Debug Register

***Note**: Writing to this register has no effect*

CS_AGSHi_DBG (i=0-2) Offset address: 0310_H+i*20_H

SMU_CS Group i Shared Alarm Debug Register PowerOn Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DF31	DF30	DF29	DF28	DF27	DF26	DF25	DF24	DF23	DF22	DF21	DF20	DF19	DF18	DF17	DF16
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DF15	DF14	DF13	DF12	DF11	DF10	DF9	DF8	DF7	DF6	DF5	DF4	DF3	DF2	DF1	DF0
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
DFz (z=0-31)	z	rh	**Diagnosis flag for alarm z in alarm group AGSHi.** The diagnosis registers make a snapshot of the alarm group status registers when either the executed alarm reaction is a reset or a state machine transition to FAULT state takes place. 0_B Status flag z does not report a fault condition 1_B Status flag z reports a fault condition

42.4.1.40 SMU_CS Group i Critical Shared Alarm Selection Flag

CS_AGSHi_CAS (i=0-2) Offset address: 0314_H+i*20_H

SMU_CS Group i Critical Shared Alarm Selection Flag Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CF31	CF30	CF29	CF28	CF27	CF26	CF25	CF24	CF23	CF22	CF21	CF20	CF19	CF18	CF17	CF16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CF15	CF14	CF13	CF12	CF11	CF10	CF9	CF8	CF7	CF6	CF5	CF4	CF3	CF2	CF1	CF0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
CFz (z=0-31)	z	rw	**Critical Alarm Selection flag for alarm z in alarm group AGSHi.** 0_B Critical alarm disabled for this alarm event 1_B Critical alarm enabled for this alarm event

Table 2005 Access mode restrictions of CS_AGSHi_CAS (i=0-2) sorted by descending priority

Mode name	Access mode		Description
CS_KEYS.CFGLCK == 0xBC and **CS_KEYS.PERLCK** ! = 0xFF	rw	CFz (z=0-31)	
(default)	r	CFz (z=0-31)	

42.4.1.41 SMU_CS Group i Shared Alarm Group Selection Register

CS_AGSHi_SEL (i=0) Offset address: 0318_H+i*20_H

SMU_CS Group i Shared Alarm Group Selection Register Reset values see: Table 2009

CS_AGSHi_SEL (i=1) Offset address: 0318_H+i*20_H

SMU_CS Group i Shared Alarm Group Selection Register Reset values see: Table 2007

CS_AGSHi_SEL (i=2) Offset address: 0318_H+i*20_H

SMU_CS Group i Shared Alarm Group Selection Register Reset values see: Table 2008

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CF31	CF30	CF29	CF28	CF27	CF26	CF25	CF24	CF23	CF22	CF21	CF20	CF19	CF18	CF17	CF16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CF15	CF14	CF13	CF12	CF11	CF10	CF9	CF8	CF7	CF6	CF5	CF4	CF3	CF2	CF1	CF0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
CFz (z=0-31)	z	rw	**Shared Alarm Group configuration flag for Alarm Group AGSHi.** The configuration flag is used to map the alarm in shared group to Security Alarm Handler. ***Note**: This flag can only be written when the corresponding bit in GCC_AGSHi_SEL.CFz Register is set to '0'.* 0_B Configuration flag z indicates Alarm z is not mapped 1_B Configuration flag z indicates Alarm z is mapped

Table 2006 Access mode restrictions sorted by descending priority

Applies to CS_AGSHi_SEL (i=0)

Applies to CS_AGSHi_SEL (i=1)

Applies to CS_AGSHi_SEL (i=2)

Mode name	Access mode		Description
CS_KEYS.CFGLCK == 0xBC and **CS_KEYS.PERLCK** != 0xFF and **GCC_AGSHi_SEL.CFz** == 0	rw	CFz (z=0-31)	
(default)	r	CFz (z=0-31)	

Table 2007 Reset values of CS_AGSHi_SEL (i=1)

Reset type	Reset value	Note
PowerOn Reset	$0000\ 0000_H$	Reset value except CS_AGSH0_SEL, CS_AGSH2_SEL

Table 2008 Reset values of CS_AGSHi_SEL (i=2)

Reset type	Reset value	Note
PowerOn Reset	$0000\ 4000_H$	Reset value for CS_AGSH2_SEL (VMT4-VMT Uncorrectable Error mapped to SMU_CS)

Table 2009 Reset values of CS_AGSHi_SEL (i=0)

Reset type	Reset value	Note
PowerOn Reset	$8000\ 0000_H$	Reset value for CS_AGSH0_SEL (CPUcs WDT overflow or access error alarm mapped to SMU_CS)

42.4.1.42 SMU_CS Group i Shared Alarm Group Selection Status

***Note**: Writing to this register has no effect*

CS_AGSHi_SS (i=0-2) Offset address: $031C_H+i*20_H$

SMU_CS Group i Shared Alarm Group Selection Status PowerOn Reset value: $FFFF\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SF31	SF30	SF29	SF28	SF27	SF26	SF25	SF24	SF23	SF22	SF21	SF20	SF19	SF18	SF17	SF16
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SF15	SF14	SF13	SF12	SF11	SF10	SF9	SF8	SF7	SF6	SF5	SF4	SF3	SF2	SF1	SF0
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
SFz (z=0-31)	z	rh	**Shared Alarm Group status flag for Alarm Group AGSHi. - CFz** The status flag is used to indicate which alarm in shared alarm group is mapped to Safety Alarm Handler. 0_B Status flag z indicates Alarm z is mapped 1_B Status flag z indicates Alarm z is not mapped

42.4.1.43 Alarm Signal Test Register

CS_ALMTST Offset address: $03F8_H$

Alarm Signal Test Register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															TE
r															rw

Field	Bits	Type	Description
TE	0	rw	**Alarm Signal Test Enable** 0_B Test disabled 1_B Test enabled
0	31:1	r	**Reserved** Read as 0; should be written with 0.

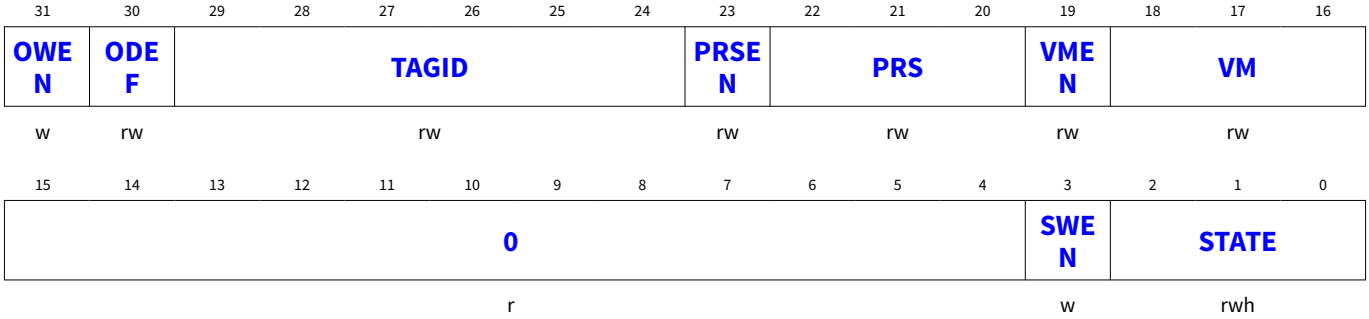
42.4.1.44 SMU_CS PROT Register Endinit

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner and in some PROT states, the secure master to update the PROT state as described in the PROT mechanism.

The PROTCE register controls lock / unlock of the local Endinit (CE) protected control registers.

CS_PROTCE Offset address: $03FC_H$

SMU_CS PROT Register Endinit Application Reset value: $4C00\ 0000_H$



Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
VM	18:16	rw	**Virtual Machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual Machine definition Enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection Set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.
PRSEN	23	rw	**Protection Set definition Enable for PROT owner** 0_B **Disable**: PRS is part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner

(table continues...)

(continued)

Field	Bits	Type	Description
OWEN	31	w	**Owner write enable** 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; should be written with 0.

Table 2010 Access mode restrictions of CS_PROTCE sorted by descending priority

Mode name	Access mode		Description
write 1 to .**SWEN**	rwh	STATE	
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

42.4.1.45 SMU_CS PROT Register Safe Endinit

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner and in some PROT states, the secure master to update the PROT state as described in the PROT mechanism.

The PROTCSE register controls lock / unlock of the local Safe Endinit (CSE) protected control registers.

CS_PROTCSE Offset address: 0400_H

SMU_CS PROT Register Safe Endinit Application Reset value: $4C00\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OWEN	**ODEF**	**TAGID**						**PRSEN**	**PRS**			**VMEN**	**VM**		
w	rw	rw						rw	rw			rw	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												**SWEN**	**STATE**		
r												w	rwh		

Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
VM	18:16	rw	**Virtual Machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual Machine definition Enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection Set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.
PRSEN	23	rw	**Protection Set definition Enable for PROT owner** 0_B **Disable**: PRS is part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner
OWEN	31	w	**Owner write enable** 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; should be written with 0.

Table 2011 Access mode restrictions of CS_PROTCSE sorted by descending priority

Mode name	Access mode		Description
write 1 to .**SWEN**	rwh	STATE	
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

42.4.1.46 SMU_CS Write access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

CS_ACCEN_WRA Offset address: 0404_H

SMU_CS Write access enable register A Application Reset value: $0000\ 3000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	**EN30**	**EN29**	**EN28**	**EN27**	**EN26**	**EN25**	**EN24**	**EN23**	**EN22**	**EN21**	**EN20**	**EN19**	**EN18**	**EN17**	**EN16**
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	**EN14**	**EN13**	**EN12**	**EN11**	**EN10**	**EN09**	**EN08**	**EN07**	**EN06**	**EN05**	**EN04**	**EN03**	**EN02**	**EN01**	**EN00**
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access

42.4.1.47 SMU_CS Write access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

CS_ACCEN_WRB Offset address: 0408_H

SMU_CS Write access enable register B Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								**EN39**	**EN38**	**EN37**	**EN36**	**EN35**	**EN34**	**EN33**	**EN32**
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access
0	31:8	r	**Reserved** Read as 0; should be written with 0.

42.4.1.48 SMU_CS Read access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

CS_ACCEN_RDA Offset address: $040C_H$

SMU_CS Read access enable register A Application Reset value: $0000\ 3000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access

42.4.1.49 SMU_CS Read access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

CS_ACCEN_RDB Offset address: 0410_H

SMU_CS Read access enable register B Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access
0	31:8	r	**Reserved** Read as 0; should be written with 0.

42.4.1.50 SMU_CS VM access enable register

This register defines which virtual machine encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each VMx.

CS_ACCEN_VM Offset address: 0414_H

SMU_CS VM access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for VM ID q** This bit enables read access to the access protected region for transactions with the VM ID q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for VM ID q** This bit enables write access to the access protected region for transactions with the VM ID q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

42.4.1.51 SMU_CS PRS access enable register

This register defines which protection register set encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each PRSx.

CS_ACCEN_PRS Offset address: 0418_H

SMU_CS PRS access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for PRS q** This bit enables read access to the access protected region for transactions with the PRS q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for PRS q** This bit enables write access to the access protected region for transactions with the PRS q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

42.4.1.52 SMU_SAFE x Command Register

SAFEx_CMD (x=0-1) Offset address: $1000_H+x*500_H$

SMU_SAFE x Command Register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								ARG				CMD			
r								w				w			

Field	Bits	Type	Description
CMD	3:0	w	**Implements the SMU_SAFE Command Interface.** Command Encoding defined in SMU_SAFE control interface . Read as 0.
ARG	7:4	w	**Implements the SMU_SAFE Command Interface.** Argument to be used with the command. Command Encoding defined in SMU_SAFE control interface. . Read as 0.

(table continues...)

(continued)

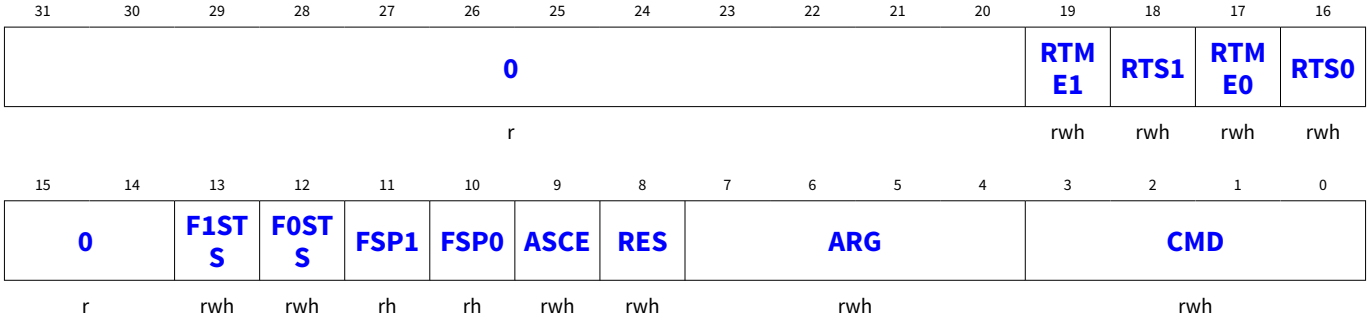
Field	Bits	Type	Description
0	31:8	r	**Reserved** Read as 0; should be written with 0

42.4.1.53 SMU_SAFE x Status Register

***Note**: A write to this register (regardless of the data written) clears all the fields with the rwh type. If on the same cycle a software write event and hardware event is detected, the hardware action wins.*

SAFEx_STS (x=0-1) Offset address: $1004_H+x*500_H$

SMU_SAFE x Status Register Reset values see: Table 2012



Field	Bits	Type	Description
CMD	3:0	rwh	**Last command received** Same encoding as CMD field of SAFEx_CMD register
ARG	7:4	rwh	**Last command argument received** Same encoding as ARG field of SAFEx_CMD register
RES	8	rwh	**Result of last received command** 0_B Command was successful 1_B Command failed
ASCE	9	rwh	**Alarm Status Clear Enable** This bit controls if a status flag set in an SAFEx_AGSFi_STS register upon detection of an alarm event can be cleared by software or not. When ASCE is enabled software shall write a 1 to the bit position in SAFEx_AGSFi_STS to clear the bit. When SAFEx_AGSFi_STS is cleared, the ASCE bit is automatically cleared to 0 by hardware and software shall set the ASCE bit again by using the SMU_ASCE() command. 0_B Alarm status bits AG<x> can not be cleared 1_B Alarm status bits AG<x> can be cleared.

(table continues...)

(continued)

Field	Bits	Type	Description
FSP0	10	rh	**Fault Signaling Protocol status** FSP0 = FSP_STS[0] input signal This field is updated by hardware every clock cycle. Note: When the FSP[0] is set in Fault State by the SMU_STDBY this bitfield does not reflect the actual state of the ErrorPins if the SMU_STDBY sets the ErrorPins in high impedence state to indicate the presence of a fault.
FSP1	11	rh	**Fault Signaling Protocol status** FSP1 = FSP_STS[1] input signal This field is updated by hardware every clock cycle. Note: When the FSP[1] is set in Fault State by the SMU_STDBY this bitfield does not reflect the actual state of the ErrorPins if the SMU_STDBY sets the ErrorPins in high impedence state to indicate the presence of a fault.
F0STS	12	rwh	**Fault State Timing Status 0** This bit indicates if the minimum timing duration of the FSP Fault State (TFSP_FS) on FSP[0] has been reached or not. The bit is cleared by hardware when the Fault State is entered. 0_B Minimum timing duration not reached 1_B Minimum timing duration reached
F1STS	13	rwh	**Fault State Timing Status 1** This bit indicates if the minimum timing duration of the FSP Fault State (TFSP_FS) on FSP[1] has been reached or not. The bit is cleared by hardware when the Fault State is entered. 0_B Minimum timing duration not reached 1_B Minimum timing duration reached
RTS0	16	rwh	**Recovery Timer 0 Status** State of the Recovery Timer 0. 0_B Recovery Timer not running 1_B Recovery Timer running
RTME0	17	rwh	**Recovery Timer 0 Missed Event** Recovery timer 0 trigger request missed. 0_B Recovery Timer event not detected 1_B Recovery Timer event detected
RTS1	18	rwh	**Recovery Timer 1 Status** State of the Recovery Timer 1. 0_B Recovery Timer not running 1_B Recovery Timer running
RTME1	19	rwh	**Recovery Timer 1 Missed Event** Recovery timer 1 trigger request missed. . 0_B Recovery Timer event not detected 1_B Recovery Timer event detected

(table continues...)

(continued)

Field	Bits	Type	Description
0	15:14, 31:20	r	**Reserved** Read as 0; should be written with 0.

Table 2012 Reset values of SAFEx_STS (x=0-1)

Reset type	Reset value	Note
Application Reset	0000 0000 0000 0000 0000 XX00 0000 0000_B	

42.4.1.54 SMU_SAFE x Fault Signaling Protocol Driver Timing Register i (i=0;x=0-1)

This register controls the timing of the fault signaling protocol.

SAFEx_FSPi_CON (i=0;x=0-1) Offset address: $1008_H+x*500_H+i*4$

SMU_SAFE x Fault Signaling Protocol Driver Timing Register i PowerOn Reset value: $003F\ FF00_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TFSP_HIGH										TFSP_LOW					
rw										r					

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TFSP_LOW								MODE		PRE2			PRE1		
r								rw		rw			rw		

Field	Bits	Type	Description
PRE1	2:0	rw	**Prescaler1** Dividing factor to apply to the reference clock f_{BACK}. It is assumed that the maximum value for f_{BACK} is 100 Mhz with a precision of 5%. The divided clock is used as reference to generate the timing of the fault signaling protocol Fault State. ***Note**: Changing the FSP timing while it is active and monitored, may result in the violation of the monitored timings.* The frequency of the divided clock (called FSMU_FS) is defined as follows: 000_B reference clock frequency divided by 2 001_B reference clock frequency divided by 4 010_B reference clock frequency divided by 8 011_B reference clock frequency divided by 16 100_B reference clock frequency divided by 32 101_B reference clock frequency divided by 64 110_B reference clock frequency divided by 128 111_B reference clock frequency divided by 256

(table continues...)

(continued)

Field	Bits	Type	Description
PRE2	5:3	rw	**Prescaler2** Dividing factor to apply to the reference clock f_{BACK} in order to generate the timing of the Fault-Free State for the dynamic dual rail and time switching modes of the fault signaling protocol. ***Note**: Changing the FSP timing while it is active and monitored, may result in the violation of the monitored timings.* The frequency of the divided clock (called FSMU_FFS) is defined as follows: 000_B reference clock frequency divided by 32 001_B reference clock frequency divided by 64 010_B reference clock frequency divided by 128 011_B reference clock frequency divided by 256 100_B reference clock frequency divided by 512 101_B reference clock frequency divided by 1024 110_B reference clock frequency divided by 2048 111_B reference clock frequency divided by 4096
MODE	7:6	rw	**Fault Signaling Protocol configuration** 00_B Bi-stable protocol 01_B Dual Rail protocol 10_B Time switching protocol 11_B Reserved
TFSP_LOW	21:8	r	**Specifies the FSP Fault State duration** $T_{FSP_FS} = T_{SMU_FS} * [concatenate(TFSP_HIGH,TPSP_LOW) + 1]$, where T_{SMU_FS} is the clock period defined by F_{SMU_FS} ticks. TFSP_LOW is defined so that the minimum duration is greater than 250 µs. It can not be changed by software.
TFSP_HIGH	31:22	rw	**Specifies the FSP Fault State duration** $T_{FSP_FS} = T_{SMU_FS} * [concatenate(TFSP_HIGH,TPSP_LOW) + 1]$, where T_{SMU_FS} is the clock period defined by F_{SMU_FS} ticks. TFSP_HIGH and PRE1 shall enable to configure a fault state duration of 500 ms. ***Note**: Changing the FSP timing while it is active and monitored, may result in the violation of the monitored timings.*

Table 2013 Access mode restrictions of SAFEx_FSPi_CON (i=0;x=0-1) sorted by descending priority

Mode name	Access mode		Description
SAFEx_KEYS.CFGLCK == 0xBC and **SAFEx_KEYS.PERLCK** != 0xFF	rw	MODE, PRE1, PRE2, TFSP_HIGH	
(default)	r	MODE, PRE1, PRE2, TFSP_HIGH	

42.4.1.55 SMU_SAFE x Fault Signaling Protocol Driver Timing Register i (i=1;x=0-1)

SAFEx_FSPi_CON (i=1;x=0-1) Offset address: $1008_H+x*500_H+i*4$

SMU_SAFE x Fault Signaling Protocol Driver Timing Register i PowerOn Reset value: $003F\ FF00_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TFSP_HIGH										TFSP_LOW					
rw										r					

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TFSP_LOW								MODE		PRE2			PRE1		
r								rw		rw			rw		

Field	Bits	Type	Description
PRE1	2:0	rw	**Prescaler1** Dividing factor to apply to the reference clock f_{BACK}. It is assumed that the maximum value for f_{BACK} is 100 Mhz with a precision of 5%. The divided clock is used as reference to generate the timing of the fault signaling protocol Fault State. ***Note**: Changing the FSP timing while it is active and monitored, may result in the violation of the monitored timings.* The frequency of the divided clock (called FSMU_FS) is defined as follows: 000_B reference clock frequency divided by 2 001_B reference clock frequency divided by 4 010_B reference clock frequency divided by 8 011_B reference clock frequency divided by 16 100_B reference clock frequency divided by 32 101_B reference clock frequency divided by 64 110_B reference clock frequency divided by 128 111_B reference clock frequency divided by 256

(table continues...)

(continued)

Field	Bits	Type	Description
PRE2	5:3	rw	**Prescaler2** Dividing factor to apply to the reference clock f_{BACK} in order to generate the timing of the Fault-Free State for the dynamic dual rail and time switching modes of the fault signaling protocol. ***Note**: Changing the FSP timing while it is active and monitored, may result in the violation of the monitored timings.* The frequency of the divided clock (called FSMU_FFS) is defined as follows: 000_B reference clock frequency divided by 32 001_B reference clock frequency divided by 64 010_B reference clock frequency divided by 128 011_B reference clock frequency divided by 256 100_B reference clock frequency divided by 512 101_B reference clock frequency divided by 1024 110_B reference clock frequency divided by 2048 111_B reference clock frequency divided by 4096
MODE	7:6	rw	**Fault Signaling Protocol configuration** 00_B Bi-stable protocol 01_B Reserved 10_B Time switching protocol 11_B Reserved
TFSP_LOW	21:8	r	**Specifies the FSP Fault State duration** $T_{FSP_FS} = T_{SMU_FS} * [concatenate(TFSP_HIGH,TPSP_LOW) + 1]$, where T_{SMU_FS} is the clock period defined by F_{SMU_FS} ticks. TFSP_LOW is defined so that the minimum duration is greater than 250 µs. It can not be changed by software.
TFSP_HIGH	31:22	rw	**Specifies the FSP Fault State duration** $T_{FSP_FS} = T_{SMU_FS} * [concatenate(TFSP_HIGH,TPSP_LOW) + 1]$, where T_{SMU_FS} is the clock period defined by F_{SMU_FS} ticks. TFSP_HIGH and PRE1 shall enable to configure a fault state duration of 500 ms. ***Note**: Changing the FSP timing while it is active and monitored, may result in the violation of the monitored timings.*

Table 2014 Access mode restrictions of SAFEx_FSPi_CON (i=1;x=0-1) sorted by descending priority

Mode name	Access mode		Description
SAFEx_KEYS.CFGLCK == 0xBC and **SAFEx_KEYS.PERLCK** != 0xFF	rw	MODE, PRE1, PRE2, TFSP_HIGH	
(default)	r	MODE, PRE1, PRE2, TFSP_HIGH	

42.4.1.56 SMU_SAFE x Port Emergency Stop Enable Register

SAFEx_PES (x=0-1) Offset address: $1010_H+x*500_H$

SMU_SAFE x Port Emergency Stop Enable Register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0														FSP1	FSP0
r														rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								CPU RST1	NMI1	PPU RST0	CPU RST0	NMI0	IGCS 2	IGCS 1	IGCS 0
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
IGCS0	0	rw	**Enable IGCS0_REQ to trigger port emergency stop** 0_B PES disabled 1_B PES enabled
IGCS1	1	rw	**Enable IGCS1_REQ to trigger port emergency stop** 0_B PES disabled 1_B PES enabled
IGCS2	2	rw	**Enable IGCS2_REQ to trigger port emergency stop** 0_B PES disabled 1_B PES enabled
NMI0	3	rw	**Enable NMI0_REQ to trigger port emergency stop** 0_B PES disabled 1_B PES enabled
CPURST0	4	rw	**Enable CPU_RESET0_REQ to trigger port emergency stop** 0_B PES disabled 1_B PES enabled
PPURST0	5	rw	**Enable PPU_RESET to trigger port emergency stop** 0_B PES disabled 1_B PES enabled
NMI1	6	rw	**Enable NMI1_REQ to trigger port emergency stop** 0_B PES disabled 1_B PES enabled
CPURST1	7	rw	**Enable CPU_RESET1_REQ to trigger port emergency stop** 0_B PES disabled 1_B PES enabled
FSP0	16	rw	**Enable FSP0_REQ to trigger port emergency stop** 0_B PES disabled 1_B PES enabled

(table continues...)

(continued)

Field	Bits	Type	Description
FSP1	17	rw	**Enable FSP1_REQ to trigger port emergency stop** 0_B PES disabled 1_B PES enabled
0	15:8, 31:18	r	**Reserved** Read as 0; should be written with 0

Table 2015 Access mode restrictions of SAFEx_PES (x=0-1) sorted by descending priority

Mode name	Access mode		Description
SAFEx_KEYS.CFGLCK == 0xBC and **SAFEx_KEYS.PERLCK** != 0xFF	rw	CPURST0, CPURST1, FSP0, FSP1, IGCS0, IGCS1, IGCS2, NMI0, NMI1, PPURST0	
(default)	r	CPURST0, CPURST1, FSP0, FSP1, IGCS0, IGCS1, IGCS2, NMI0, NMI1, PPURST0	

42.4.1.57 SMU_SAFE x Alarm Global Configuration

This register controls some properties related to the behavior of the SMU_safe to alarm.

SAFEx_AGC (x=0-1) Offset address: $1014_H+x*500_H$

SMU_SAFE x Alarm Global Configuration Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ESSM1	EFRST1	EFRST0	0					RCS1					0	RCS0	
rw	rw	rw	r					rw					r	rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RCS0			0		IGCS2			0	IGCS1			0	IGCS0		
rw			r		rw			r	rw			r	rw		

Field	Bits	Type	Description
IGCS0	2:0	rw	**Interrupt Generation Configuration Set 0** Defines the output value of the interrupt request vector when the alarm configuration flag selects the interrupt configuration set 0 (IGCS0_REQ). Enables to issue an interrupt request to several CPUs.
IGCS1	6:4	rw	**Interrupt Generation Configuration Set 1** Defines the output value of the interrupt request vector when the alarm configuration flag selects the interrupt configuration set 1 (IGCS1_REQ). Enables to issue an interrupt request to several CPUs.
IGCS2	10:8	rw	**Interrupt Generation Configuration Set 2** Defines the output value of the interrupt request vector when the alarm configuration flag selects the interrupt configuration set 2 (IGCS2_REQ). Enables to issue an interrupt request to several CPUs.

(table continues...)

(continued)

Field	Bits	Type	Description
RCS0	17:13	rw	**CPU Reset Configuration Set 0** Defines the output value of the CPU reset request vector when the alarm configuration flag selects the CPU Reset Configuration Set (CPU_RESET0_REQ). Enables to issue an reset request to several CPUs if required. More complex reset scenarios can be handled by using software interrupts. Setting the bit (n-1) to 1 enables issuing a reset request to CPUn with n = 1-5.
RCS1	23:19	rw	**CPU Reset Configuration Set 1** Defines the output value of the CPU reset request vector when the alarm configuration flag selects the CPU Reset Configuration Set (CPU_RESET1_REQ). Enables to issue an reset request to several CPUs if required. More complex reset scenarios can be handled by using software interrupts. Setting the bit (n-1) to 1 enables issuing a reset request to CPUn with n = 1-5.
EFRST0	29	rw	**Enable FAULT to RUN State Transition in SSM0** 0_B FAULT to RUN State Transition disabled 1_B FAULT to RUN State Transition enabled
EFRST1	30	rw	**Enable FAULT to RUN State Transition in SSM1** 0_B FAULT to RUN State Transition disabled 1_B FAULT to RUN State Transition enabled
ESSM1	31	rw	**Enable START to RUN State Transition in SSM1** 0_B FAULT to RUN State Transition disabled 1_B FAULT to RUN State Transition enabled
0	3, 7, 12:11, 18, 28:24	r	**Reserved** Read as 0; should be written with 0

Table 2016 Access mode restrictions of SAFEx_AGC (x=0-1) sorted by descending priority

Mode name	Access mode		Description
SAFEx_KEYS.CFGLCK == 0xBC and **SAFEx_KEYS.PERLCK** != 0xFF	rw	EFRST0, EFRST1, ESSM1, IGCS0, IGCS1, IGCS2, RCS0, RCS1	
(default)	r	EFRST0, EFRST1, ESSM1, IGCS0, IGCS1, IGCS2, RCS0, RCS1	

42.4.1.58 SMU_SAFE x Recovery Timer Configuration

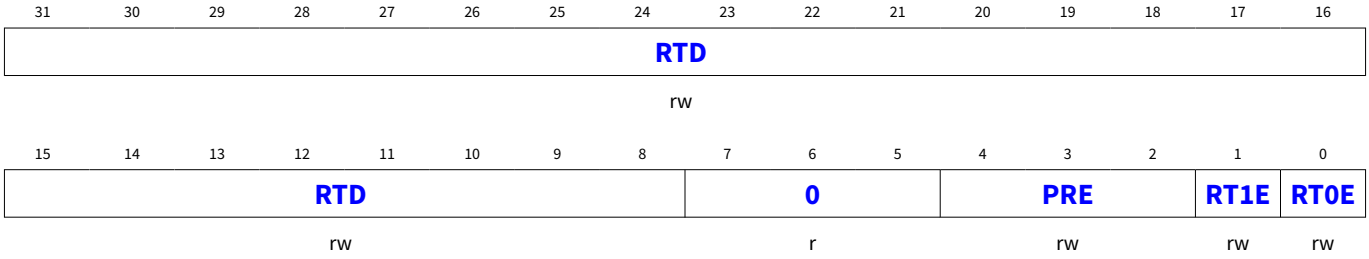
This register controls the timing duration of the recovery timer.

SAFEx_RTC (x=0) Offset address: $1018_H+x*500_H$

SMU_SAFE x Recovery Timer Configuration Reset values see: Table 2019

SAFEx_RTC (x=1) Offset address: $1018_H+x*500_H$

SMU_SAFE x Recovery Timer Configuration Reset values see: Table 2018



Field	Bits	Type	Description
RT0E	0	rw	**RT0 Enable Bit** 0_B Recovery Timer 0 is disabled 1_B Recovery Timer 0 is enabled
RT1E	1	rw	**RT1 Enable Bit** 0_B Recovery Timer 1 is disabled 1_B Recovery Timer 1 is enabled
PRE	4:2	rw	**Prescaler For Recovery Timer Clock** Dividing factor to apply to the reference clock f_{BACK}. It is assumed that the maximum value for f_{BACK} is 100 Mhz with a precision of 5%. The divided clock is used as reference clock for recovery timer. The frequency of the divided clock is defined as follows: 000_B reference clock frequency divided by 2 001_B reference clock frequency divided by 4 010_B reference clock frequency divided by 8 011_B reference clock frequency divided by 16 100_B reference clock frequency divided by 32 101_B reference clock frequency divided by 64 110_B reference clock frequency divided by 128 111_B reference clock frequency divided by 256
RTD	31:8	rw	**Recovery Timer Duration** This field specifies the maximum duration of the recovery timer. When the timer counter reaches the programmed value, the internal time-out alarm is triggered. The timer is stopped by a SMU_RTStop() command before the recovery timer. RTD shall be specified as a number of the fBACK/SAFEx_RTC.PRE clock ticks. ***Note**: It is not allowed to write to the RTD when at least one recovery timer is running.*

(table continues...)

(continued)

Field	Bits	Type	Description
0	7:5	r	**Reserved** Read as 0; should be written with 0

Table 2017 Access mode restrictions sorted by descending priority

Applies to SAFEx_RTC (x=0)

Applies to SAFEx_RTC (x=1)

Mode name	Access mode		Description
SAFEx_KEYS.CFGLCK == 0xBC and **SAFEx_KEYS.PERLCK** != 0xFF	rw	PRE, RT0E, RT1E, RTD	
(default)	r	PRE, RT0E, RT1E, RTD	

Table 2018 Reset values of SAFEx_RTC (x=1)

Reset type	Reset value	Note
Application Reset	003F FF00$_H$	SMU_SAFE1 RT0 RT1 Disable

Table 2019 Reset values of SAFEx_RTC (x=0)

Reset type	Reset value	Note
Application Reset	003F FF03$_H$	SMU_SAFE0 RT0 RT1 Enable

42.4.1.59 SMU_SAFE x Key Register

SAFEx_KEYS (x=0-1) Offset address: 101C$_H$+x*500$_H$

SMU_SAFE x Key Register Application Reset value: 0000 0000$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PERLCK								CFGLCK							
rw								rw							

Field	Bits	Type	Description
CFGLCK	7:0	rw	**Configuration Lock** The SMU_SAFE configuration is only possible if this field is set to 0xBC.
PERLCK	15:8	rw	**Permanent Lock** If this field is set to 0xFF, no further configuration of the SMU_SAFE is possible before next application reset.
0	31:16	r	**Reserved** Read as 0; should be written with 0

Table 2020 Access mode restrictions of SAFEx_KEYS (x=0-1) sorted by descending priority

Mode name	Access mode		Description
SAFEx_KEYS.PERLCK != 0xFF	rw	CFGLCK, PERLCK	
(default)	r	CFGLCK, PERLCK	

42.4.1.60 SMU_SAFE x Debug Register

This register enables to observe some internal states of the SMU_safe hardware.

SAFEx_DBG (x=0-1) Offset address: $1020_H+x*500_H$

SMU_SAFE x Debug Register PowerOn Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												SSM1		SSM0	
r												rh		rh	

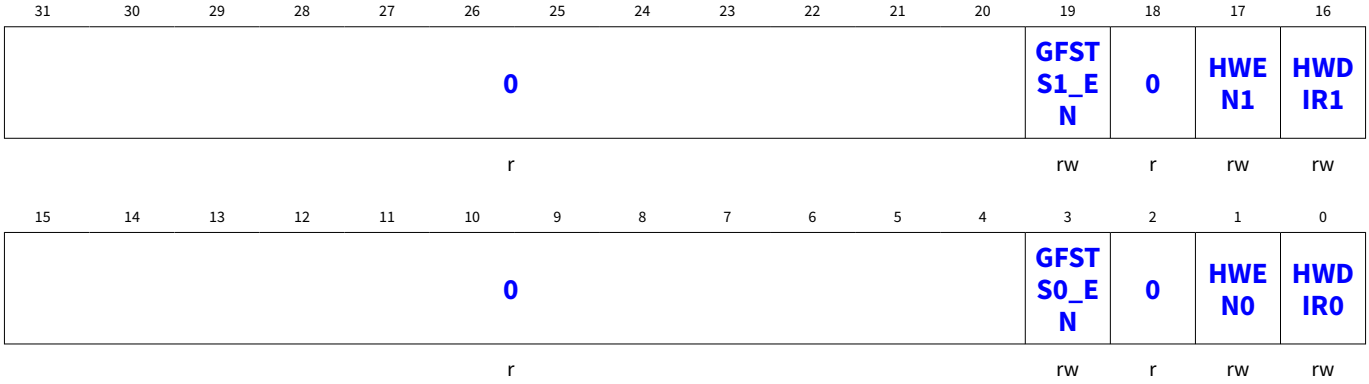
Field	Bits	Type	Description
SSM0	1:0	rh	**Running state of the SMU_SAFE state machine 0** 00_B START state 01_B RUN state 10_B FAULT state 11_B unspecified state
SSM1	3:2	rh	**Running state of the SMU_SAFE state machine 1** 00_B START state 01_B RUN state 10_B FAULT state 11_B unspecified state
0	31:4	r	**Reserved** Read as 0; should be written with 0

42.4.1.61 SMU_SAFE x Port Control

This register controlls the connectivity with the Ports

SAFEx_PCTL (x=0-1) Offset address: $1024_H+x*500_H$

SMU_SAFE x Port Control PowerOn Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
HWDIR0	0	rw	**Port Direction for FSP[0] or FSP[2]** This bitfield directly controls the value of the FSP_DIR output signal. Also refer to the Interface to the Ports Pad section of SMU chapter. ***Note**: To avoid glitches it is recommended to update HWDIR0 and HWEN0 bitfields with separate write access.* 0_B sets FSP[0] or FSP[2] to input state 1_B sets FSP[0] or FSP[2] to output state
HWEN0	1	rw	**Port Enable for FSP[0] or FSP[2]- HWEN** This bitfied directly controls the value of the FSP_EN output signal. When set to 1b the port output is directly driven by SMU_SAFE. Also refer to the Interface to the Ports Pad section of SMU chapter. ***Note**: To avoid glitches it is recommended to update HWDIR0 and HWEN0 bitfields with separate write access.* 0_B FSP[0] or FSP[2] port output is not driven by SMU_SAFE. 1_B FSP[0] or FSP[2] port output is directly driven by SMU_SAFE.
GFSTS0_EN	3	rw	**Glitch Filter for ErrorPin FSP[0] or FSP[2] to SAFEx_STS.FSP0 enable - GFSTS_EN** 0_B Glitch Filter disabled 1_B Glitch FIlter enabled
HWDIR1	16	rw	**Port Direction for FSP[1] or FSP[3]** This bitfield directly controls the value of the FSP_DIR output signal. Also refer to the Interface to the Ports Pad section of SMU chapter. ***Note**: To avoid glitches it is recommended to update HWDIR1 and HWEN1 bitfields with separate write access.* 0_B sets FSP[1] or FSP[3] to input state 1_B sets FSP[1] or FSP[3] to output state

(table continues...)

(continued)

Field	Bits	Type	Description
HWEN1	17	rw	**Port Enable for FSP[1] or FSP[3]- HWEN** This bitfied directly controls the value of the FSP_EN output signal. When set to 1b the port output is directly driven by SMU_SAFE. Also refer to the Interface to the Ports Pad section of SMU chapter. ***Note**: To avoid glitches it is recommended to update HWDIR1 and HWEN1 bitfields with separate write access.* 0_B FSP[1] or FSP[3] port output is not driven by SMU_SAFE. 1_B FSP[1] or FSP[3] port output is directly driven by SMU_SAFE.
GFSTS1_EN	19	rw	**Glitch Filter for ErrorPin FSP[1] or FSP[3] to SAFEx_STS.FSP0 enable - GFSTS_EN** 0_B Glitch Filter disabled 1_B Glitch FIlter enabled
0	2, 15:4, 18, 31:20	r	**Reserved** Read as 0; should be written with 0

Table 2021 Access mode restrictions of SAFEx_PCTL (x=0-1) sorted by descending priority

Mode name	Access mode		Description
SAFEx_KEYS.CFGLCK == 0xBC and **SAFEx_KEYS.PERLCK** != 0xFF	rw	GFSTS0_EN, GFSTS1_EN, HWDIR0, HWDIR1, HWEN0, HWEN1	
(default)	r	GFSTS0_EN, GFSTS1_EN, HWDIR0, HWDIR1, HWEN0, HWEN1	

42.4.1.62 SMU_SAFE x Alarm and Fault Counter

SAFEx_AFCNT (x=0-1) Offset address: $1028_H+x*500_H$

SMU_SAFE x Alarm and Fault Counter PowerOn Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ACO	**F0CO**	**0**	**F1CO**	**F1CNT**				**0**							
rh	rh	r	rh	rh				r							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ACNT												**F0CNT**			
rh												rh			

Field	Bits	Type	Description
F0CNT	3:0	rh	**Fault Counter - FCNT** This field is incremented by hardware when the SMU_SAFE state machine SSM0 goes from the RUN state to the FAULT state. The counter value holds if the maximum value is reached.
ACNT	15:4	rh	**Alarm Counter** This field is incremented by hardware when the SMU_SAFE processes an internal reaction related to an alarm event. The counter value is kept stable when the maximum value is reached.
F1CNT	27:24	rh	**Fault Counter** This field is incremented by hardware when the SMU_SAFE state machine SSM1 goes from the RUN state to the FAULT state. The counter value holds if the maximum value is reached.
F1CO	28	rh	**Fault Counter Overflow** This bit is set by hardware if the F1CNT counter reached the maximum value and an increment condition is present.
F0CO	30	rh	**Fault Counter Overflow** This bit is set by hardware if the FCNT counter reached the maximum value and an increment condition is present.
ACO	31	rh	**Alarm Counter Overflow** This bit is set by hardware if the ACNT counter reached the maximum value and an increment condition is present.
0	23:16, 29	r	**Reserved** Read as 0; should be written with 0.

42.4.1.63 SMU_SAFE x Recovery Timer 0 Alarm Configuration A

***Note**: It is possible to configure multiple times the same group identifier in GID0/1 fields.*

SAFEx_RT0ACA (x=0) Offset address: $1030_H+x*500_H$

SMU_SAFE x Recovery Timer 0 Alarm Configuration A Reset values see: Table 2024

SAFEx_RT0ACA (x=1) Offset address: $1030_H+x*500_H$

SMU_SAFE x Recovery Timer 0 Alarm Configuration A Reset values see: Table 2023

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0						ALID1					GID1				
r						rw					rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0						ALID0					GID0				
r						rw					rw				

Field	Bits	Type	Description
GID0	4:0	rw	**Group Index 0** This field enables to specify if an alarm from this alarm group can use the recovery timer 0.
ALID0	9:5	rw	**Alarm Identifier 0** This field specifies the alarm index related to the group index specified in GID0.
GID1	20:16	rw	**Group Index 1** This field enables to specify if an alarm from this alarm group can use the recovery timer 0.
ALID1	25:21	rw	**Alarm Identifier 1** This field specifies the alarm index related to the group index specified in GID1.
0	15:10, 31:26	r	**Reserved** Read as 0; should be written with 0.

Table 2022 Access mode restrictions sorted by descending priority

Applies to SAFEx_RT0ACA (x=0)

Applies to SAFEx_RT0ACA (x=1)

Mode name	Access mode		Description
SAFEx_KEYS.CFGLCK == 0xBC and **SAFEx_KEYS.PERLCK** != 0xFF	rw	ALID0, ALID1, GID0, GID1	
(default)	r	ALID0, ALID1, GID0, GID1	

Table 2023 Reset values of SAFEx_RT0ACA (x=1)

Reset type	Reset value	Note
Application Reset	0000 0000$_H$	SMU_SAFE1 Default Mapping

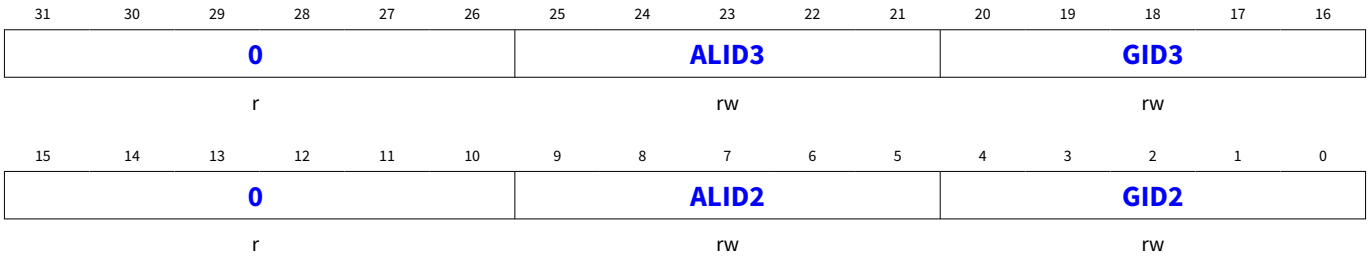
Table 2024 Reset values of SAFEx_RT0ACA (x=0)

Reset type	Reset value	Note
Application Reset	0028 0008$_H$	SMU_SAFE0 Recovery Timer 0 to CPU0 WDT, CPU1 WDT overflow or access error alarm

42.4.1.64 SMU_SAFE x Recovery Timer 0 Alarm Configuration B

***Note**: It is possible to configure multiple times the same group identifier in GID2/3 fields.*

SAFEx_RT0ACB (x=0)	Offset address:	$1034_H+x*500_H$
SMU_SAFE x Recovery Timer 0 Alarm Configuration B	Reset values see:	Table 2027
SAFEx_RT0ACB (x=1)	Offset address:	$1034_H+x*500_H$
SMU_SAFE x Recovery Timer 0 Alarm Configuration B	Reset values see:	Table 2026



Field	Bits	Type	Description
GID2	4:0	rw	**Group Index 2** This field enables to specify if an alarm from this alarm group can use the recovery timer 0.
ALID2	9:5	rw	**Alarm Identifier 2** This field specifies the alarm index related to the group index specified in GID2.
GID3	20:16	rw	**Group Index 3** This field enables to specify if an alarm from this alarm group can use the recovery timer 0.
ALID3	25:21	rw	**Alarm Identifier 3** This field specifies the alarm index related to the group index specified in GID3.
0	15:10, 31:26	r	**Reserved** Read as 0; should be written with 0.

Table 2025 Access mode restrictions sorted by descending priority

Applies to SAFEx_RT0ACB (x=0)

Applies to SAFEx_RT0ACB (x=1)

Mode name	Access mode		Description
SAFEx_KEYS.CFGLCK == 0xBC and **SAFEx_KEYS.PERLCK** != 0xFF	rw	ALID2, ALID3, GID2, GID3	
(default)	r	ALID2, ALID3, GID2, GID3	

Table 2026 Reset values of SAFEx_RT0ACB (x=1)

Reset type	Reset value	Note
Application Reset	$0000\ 0000_H$	SMU_SAFE1 Default Mapping

Table 2027 Reset values of SAFEx_RT0ACB (x=0)

Reset type	Reset value	Note
Application Reset	$00C8\ 0048_H$	SMU_SAFE0 Recovery Timer 0 to CPU2 WDT, System WDT overflow or access error alarm

42.4.1.65 SMU_SAFE x Recovery Timer 1 Alarm Configuration A

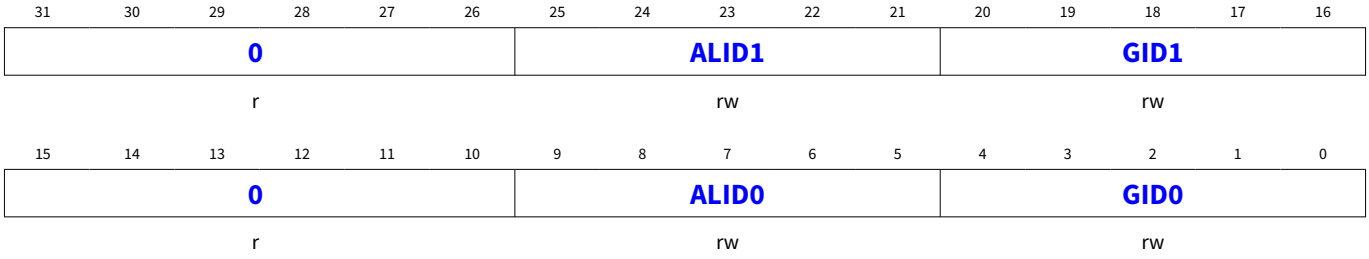
***Note**: It is possible to configure multiple times the same group identifier in GID0/1 fields.*

SAFEx_RT1ACA (x=0) Offset address: $1038_H+x*500_H$

SMU_SAFE x Recovery Timer 1 Alarm Configuration A Reset values see: Table 2030

SAFEx_RT1ACA (x=1) Offset address: $1038_H+x*500_H$

SMU_SAFE x Recovery Timer 1 Alarm Configuration A Reset values see: Table 2029



Field	Bits	Type	Description
GID0	4:0	rw	**Group Index 0.** This field enables to specify if an alarm from this alarm group can use the recovery timer 1.
ALID0	9:5	rw	**Alarm Identifier 0.** This field specifies the alarm index related to the group index specified in GID0.
GID1	20:16	rw	**Group Index 1.** This field enables to specify if an alarm from this alarm group can use the recovery timer 1.
ALID1	25:21	rw	**Alarm Identifier 1.** This field specifies the alarm index related to the group index specified in GID1.
0	15:10, 31:26	r	**Reserved** Read as 0; should be written with 0.

Table 2028 Access mode restrictions sorted by descending priority

Applies to SAFEx_RT1ACA (x=0)

Applies to SAFEx_RT1ACA (x=1)

Mode name	Access mode		Description
SAFEx_KEYS.CFGLCK == 0xBC and **SAFEx_KEYS.PERLCK** != 0xFF	rw	ALID0, ALID1, GID0, GID1	
(default)	r	ALID0, ALID1, GID0, GID1	

Table 2029 Reset values of SAFEx_RT1ACA (x=1)

Reset type	Reset value	Note
Application Reset	$0000\ 0000_H$	SMU_SAFE1 Default Mapping

Table 2030 Reset values of SAFEx_RT1ACA (x=0)

Reset type	Reset value	Note
Application Reset	0088 0068$_H$	SMU_SAFE0 Recovery Timer 1 to CPU3 WDT, CPU4 WDT overflow or access error alarm

42.4.1.66 SMU_SAFE x Recovery Timer 1 Alarm Configuration B

***Note**: It is possible to configure multiple times the same group identifier in GID2/3 fields.*

SAFEx_RT1ACB (x=0) Offset address: 103C$_H$+x*500$_H$

SMU_SAFE x Recovery Timer 1 Alarm Configuration B Reset values see: Table 2033

SAFEx_RT1ACB (x=1) Offset address: 103C$_H$+x*500$_H$

SMU_SAFE x Recovery Timer 1 Alarm Configuration B Reset values see: Table 2032

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0						ALID3					GID3				
r						rw					rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0						ALID2					GID2				
r						rw					rw				

Field	Bits	Type	Description
GID2	4:0	rw	**Group Index 2.** This field enables to specify if an alarm from this alarm group can use the recovery timer 1.
ALID2	9:5	rw	**Alarm Identifier 2.** This field specifies the alarm index related to the group index specified in GID2.
GID3	20:16	rw	**Group Index 3.** This field enables to specify if an alarm from this alarm group can use the recovery timer 1.
ALID3	25:21	rw	**Alarm Identifier 3.** This field specifies the alarm index related to the group index specified in GID3.
0	15:10, 31:26	r	**Reserved** Read as 0; should be written with 0.

Table 2031 Access mode restrictions sorted by descending priority

Applies to SAFEx_RT1ACB (x=0)

Applies to SAFEx_RT1ACB (x=1)

Mode name	Access mode		Description
SAFEx_KEYS.CFGLCK == 0xBC and **SAFEx_KEYS.PERLCK** != 0xFF	rw	ALID2, ALID3, GID2, GID3	
(default)	r	ALID2, ALID3, GID2, GID3	

Table 2032 Reset values of SAFEx_RT1ACB (x=1)

Reset type	Reset value	Note
Application Reset	0000 0000_H	SMU_SAFE1 Default Mapping

Table 2033 Reset values of SAFEx_RT1ACB (x=0)

Reset type	Reset value	Note
Application Reset	00A8 $00A8_H$	SMU_SAFE0 Recovery Timer 1 to CPU5 WDT overflow or access error alarm

42.4.1.67 SMU_SAFE x Alarm Executed Status Register

The Alarm Executed Status Register is used to show, which alarm mechanisms are executed.

SAFEx_AEX (x=0-1) Offset address: $1050_H+x*500_H$

SMU_SAFE x Alarm Executed Status Register Application Reset value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	PRSTAEM	Res	NMI1AEM	EMSAEM	Res	NMI0AEM	C5RSTAEM	C4RSTAEM	C3RSTAEM	C2RSTAEM	C1RSTAEM	0	IRQ2AEM	IRQ1AEM	IRQ0AEM
r	rh	r	rh	rh	r	rh	rh	rh	rh	rh	rh	r	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	PRSTSTS	Res	NMI1STS	EMSSTS	Res	NMI0STS	C5RSTSTS	C4RSTSTS	C3RSTSTS	C2RSTSTS	C1RSTSTS	0	IRQ2STS	IRQ1STS	IRQ0STS
r	rh	r	rh	rh	r	rh	rh	rh	rh	rh	rh	r	rh	rh	rh

Field	Bits	Type	Description
IRQ0STS	0	rh	**IRQ0 Request Status** This bit indicates whether a IRQ0 request was serviced or not. This bit is set by the SMU_SAFE after an alarm configured for IRQ0 is detected. This bit can be cleared by writing with ´1´ to the IRQ0C bit in the SAFEx_AEXCLR register. 0_B NO IRQ0 request was serviced 1_B IRQ0 request was serviced

(table continues...)

(continued)

Field	Bits	Type	Description
IRQ1STS	1	rh	**IRQ1 Request Status** This bit indicates whether a IRQ1 request was serviced or not. This bit is set by the SMU_SAFE after an alarm configured for IRQ1 is detected. This bit can be cleared by writing with ´1´ to the IRQ1C bit in the SAFEx_AEXCLR register. 0_B NO IRQ1 request was serviced 1_B IRQ1 request was serviced
IRQ2STS	2	rh	**IRQ2 Request Status** This bit indicates whether a IRQ2 request was serviced or not. This bit is set by the SMU_SAFE after an alarm configured for IRQ2 is detected. This bit can be cleared by writing with ´1´ to the IRQ2C bit in the SAFEx_AEXCLR register. 0_B NO IRQ2 request was serviced 1_B IRQ2 request was serviced
C1RSTSTS	4	rh	**CPU1 RESET Request Status** This bit indicates whether a CPU1_RESET request was serviced or not. This bit is set by the SMU_SAFE after an alarm configured for CPU1_RESET is detected. This bit can be cleared by writing with ´1´ to the C1RSTC bit in the SAFEx_AEXCLR register. 0_B NO CPU1_RESET request was serviced 1_B CPU1_RESET request was serviced
C2RSTSTS	5	rh	**CPU2 RESET Request Status** This bit indicates whether a CPU2_RESET request was serviced or not. This bit is set by the SMU_SAFE after an alarm configured for CPU2_RESET is detected. This bit can be cleared by writing with ´1´ to the C2RSTC bit in the SAFEx_AEXCLR register. 0_B NO CPU2_RESET request was serviced 1_B CPU2_RESET request was serviced
C3RSTSTS	6	rh	**CPU3 RESET Request Status** This bit indicates whether a CPU3_RESET request was serviced or not. This bit is set by the SMU_SAFE after an alarm configured for CPU3_RESET is detected. This bit can be cleared by writing with ´1´ to the C3RSTC bit in the SAFEx_AEXCLR register. 0_B NO CPU3_RESET request was serviced 1_B CPU3_RESET request was serviced

(table continues...)

(continued)

Field	Bits	Type	Description
C4RSTSTS	7	rh	**CPU4 RESET Request Status** This bit indicates whether a CPU4_RESET request was serviced or not. This bit is set by the SMU_SAFE after an alarm configured for CPU4_RESET is detected. This bit can be cleared by writing with ´1´ to the C4RSTC bit in the SAFEx_AEXCLR register. 0_B NO CPU4_RESET request was serviced 1_B CPU4_RESET request was serviced
C5RSTSTS	8	rh	**CPU5 RESET Request Status** This bit indicates whether a CPU5_RESET request was serviced or not. This bit is set by the SMU_SAFE after an alarm configured for CPU5_RESET is detected. This bit can be cleared by writing with ´1´ to the C5RSTC bit in the SAFEx_AEXCLR register. 0_B NO CPU5_RESET request was serviced 1_B CPU5_RESET request was serviced
NMI0STS	9	rh	**NMI0 Request Status** This bit indicates whether a NMI0 request was serviced or not. This bit is set by the SMU_SAFE after an alarm configured for NMI0 is detected. This bit can be cleared by writing with ´1´ to the NMI0C bit in the SAFEx_AEXCLR register. 0_B NO NMI0 request was serviced 1_B NMI0 request was serviced
EMSSTS	11	rh	**EMS Request Status** This bit indicates whether a emergency stop (EMS) request, triggered by an alarm was serviced or not. This bit is set by the SMU_SAFE after an alarm configured for emergency stop is detected. Further emergency stop requests will be blocked unless this bit is cleared. This bit can be cleared by writing with ´1´ to the EMSC bit in the SAFEx_AEXCLR register. ***Note**: This bit is also set by EMS request using SMU_ActivatePES(). However, SMU_ActivatePES() always trigger EMS request irrespective of the situation where this bit is already set.* 0_B NO EMS request was serviced 1_B EMS request was serviced
NMI1STS	12	rh	**NMI1 Request Status** This bit indicates whether a NMI1 request was serviced or not. This bit is set by the SMU_SAFE after an alarm configured for NMI1 is detected. This bit can be cleared by writing with ´1´ to the NMI1C bit in the SAFEx_AEXCLR register. 0_B NO NMI1 request was serviced 1_B NMI1 request was serviced

(table continues...)

(continued)

Field	Bits	Type	Description
PRSTSTS	14	rh	**PPU RESET Request Status** This bit indicates whether a PPU_RESET request was serviced or not. This bit is set by the SMU_SAFE after an alarm configured for PPU_RESET is detected. This bit can be cleared by writing with ´1´ to the PRSTC bit in the SAFEx_AEXCLR register. 0_B NO PPU_RESET request was serviced 1_B PPU_RESET request was serviced
IRQ0AEM	16	rh	**IRQ0 AEM** This bit indicates that an alarm event is missed. That means that an alarm has occurred with configuration for IRQ0 Request where this alarm handler was blocked because SAFEx_AEX.IRQ0STS was already set. This bit can be cleared by writing with ´1´ to the IRQ0AEMC bit in the SAFEx_AEXCLR register. 0_B No alarm is missed 1_B At least one alarm is missed
IRQ1AEM	17	rh	**IRQ1 AEM** This bit indicates that an alarm event is missed. That means that an alarm has occurred with configuration for IRQ1 Request where this alarm handler was blocked because SAFEx_AEX.IRQ1STS was already set. This bit can be cleared by writing with ´1´ to the IRQ1AEMC bit in the SAFEx_AEXCLR register. 0_B No alarm is missed 1_B At least one alarm is missed
IRQ2AEM	18	rh	**IRQ2 AEM** This bit indicates that an alarm event is missed. That means that an alarm has occurred with configuration for IRQ2 Request where this alarm handler was blocked because SAFEx_AEX.IRQ2STS was already set. This bit can be cleared by writing with ´1´ to the IRQ2AEMC bit in the SAFEx_AEXCLR register. 0_B No alarm is missed 1_B At least one alarm is missed
C1RSTAEM	20	rh	**C1RST AEM** This bit indicates that an alarm event is missed. That means that an alarm has occurred with configuration for CPU1_RESET Request where this alarm handler was blocked because SAFEx_AEX.C1RSTSTS was already set. This bit can be cleared by writing with ´1´ to the C1RSTAEMC bit in the SAFEx_AEXCLR register. 0_B No alarm is missed 1_B At least one alarm is missed

(table continues...)

(continued)

Field	Bits	Type	Description
C2RSTAEM	21	rh	**C2RST AEM** This bit indicates that an alarm event is missed. That means that an alarm has occurred with configuration for CPU2_RESET Request where this alarm handler was blocked because SAFEx_AEX.C2RSTSTS was already set. This bit can be cleared by writing with ´1´ to the C2RSTAEMC bit in the SAFEx_AEXCLR register. 0_B No alarm is missed 1_B At least one alarm is missed
C3RSTAEM	22	rh	**C3RST AEM** This bit indicates that an alarm event is missed. That means that an alarm has occurred with configuration for CPU3_RESET Request where this alarm handler was blocked because SAFEx_AEX.C3RSTSTS was already set. This bit can be cleared by writing with ´1´ to the C3RSTAEMC bit in the SAFEx_AEXCLR register. 0_B No alarm is missed 1_B At least one alarm is missed
C4RSTAEM	23	rh	**C4RST AEM** This bit indicates that an alarm event is missed. That means that an alarm has occurred with configuration for CPU4_RESET Request where this alarm handler was blocked because SAFEx_AEX.C4RSTSTS was already set. This bit can be cleared by writing with ´1´ to the C4RSTAEMC bit in the SAFEx_AEXCLR register. 0_B No alarm is missed 1_B At least one alarm is missed
C5RSTAEM	24	rh	**C5RST AEM** This bit indicates that an alarm event is missed. That means that an alarm has occurred with configuration for CPU5_RESET Request where this alarm handler was blocked because SAFEx_AEX.C5RSTSTS was already set. This bit can be cleared by writing with ´1´ to the C5RSTAEMC bit in the SAFEx_AEXCLR register. 0_B No alarm is missed 1_B At least one alarm is missed
NMI0AEM	25	rh	**NMI0 AEM** This bit indicates that an alarm event is missed. That means that an alarm has occurred with configuration for NMI0 Request where this alarm handler was blocked because SAFEx_AEX.NMI0STS was already set. This bit can be cleared by writing with ´1´ to the NMI0AEMC bit in the SAFEx_AEXCLR register. 0_B No alarm is missed 1_B At least one alarm is missed

(table continues...)

(continued)

Field	Bits	Type	Description
EMSAEM	27	rh	**EMS AEM** This bit indicates that an alarm event is missed. That means that an alarm has occurred with configuration for EMS Request where this alarm handler was blocked because SAFEx_AEX.EMSSTS was already set. This bit can be cleared by writing with ´1´ to the EMSAEMC bit in the SAFEx_AEXCLR register. 0_B No alarm is missed 1_B At least one alarm is missed
NMI1AEM	28	rh	**NMI1 AEM** This bit indicates that an alarm event is missed. That means that an alarm has occurred with configuration for NMI1 Request where this alarm handler was blocked because SAFEx_AEX.NMI1STS was already set. This bit can be cleared by writing with ´1´ to the NMI1AEMC bit in the SAFEx_AEXCLR register. 0_B No alarm is missed 1_B At least one alarm is missed
PRSTAEM	30	rh	**PRST AEM** This bit indicates that an alarm event is missed. That means that an alarm has occurred with configuration for PPU_RESET Request where this alarm handler was blocked because SAFEx_AEX.PRSTSTS was already set. This bit can be cleared by writing with ´1´ to the PRSTAEMC bit in the SAFEx_AEXCLR register. 0_B No alarm is missed 1_B At least one alarm is missed
0	3, 15, 19, 31	r	**Reserved** Read as 0; should be written with 0.

42.4.1.68 SMU_SAFE x Alarm Executed Status Clear Register

The Alarm Executed Status Clear register is used to clear the Alarm Executed Status (SMUx_AEX) register.

SAFEx_AEXCLR (x=0-1) Offset address: $1054_H+x*500_H$

SMU_SAFE x Alarm Executed Status Clear Register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	PRSTAEMC	Res	NMI1AEMC	EMSAEMC	Res	NMI0AEMC	C5RSTAEMC	C4RSTAEMC	C3RSTAEMC	C2RSTAEMC	C1RSTAEMC	0	IRQ2AEMC	IRQ1AEMC	IRQ0AEMC
r	w	r	w	w	r	w	w	w	w	w	w	r	w	w	w

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	PRSTC	Res	NMI1C	EMSC	Res	NMI0C	C5RSTC	C4RSTC	C3RSTC	C2RSTC	C1RSTC	0	IRQ2C	IRQ1C	IRQ0C
r	w	r	w	w	r	w	w	w	w	w	w	r	w	w	w

Field	Bits	Type	Description
IRQ0C	0	w	**IRQ0STS Clear** Read always as 0. 0_B No action 1_B Clear SAFEx_AEX.IRQ0STS status bit.
IRQ1C	1	w	**IRQ1STS Clear** Read always as 0. 0_B No action 1_B Clear SAFEx_AEX.IRQ1STS status bit.
IRQ2C	2	w	**IRQ2STS Clear** Read always as 0. 0_B No action 1_B Clear SAFEx_AEX.IRQ2STS status bit.
C1RSTC	4	w	**C1RSTSTS Clear** Read always as 0. 0_B No action 1_B Clear SAFEx_AEX.C1RSTSTS status bit.
C2RSTC	5	w	**C2RSTSTS Clear** Read always as 0. 0_B No action 1_B Clear SAFEx_AEX.C2RSTSTS status bit.
C3RSTC	6	w	**C3RSTSTS Clear** Read always as 0. 0_B No action 1_B Clear SAFEx_AEX.C3RSTSTS status bit.
C4RSTC	7	w	**C4RSTSTS Clear** Read always as 0. 0_B No action 1_B Clear SAFEx_AEX.C4RSTSTS status bit.

(table continues...)

(continued)

Field	Bits	Type	Description
C5RSTC	8	w	**C5RSTSTS Clear** Read always as 0. 0_B No action 1_B Clear SAFEx_AEX.C5RSTSTS status bit.
NMI0C	9	w	**NMI0STS Clear** Read always as 0. 0_B No action 1_B Clear SAFEx_AEX.NMI0STS status bit
EMSC	11	w	**EMSSTS Clear** Read always as 0. 0_B No action 1_B Clear SAFEx_AEX.EMSSTS status bit.
NMI1C	12	w	**NMI1STS Clear** Read always as 0. 0_B No action 1_B Clear SAFEx_AEX.NMI1STS status bit.
PRSTC	14	w	**PRSTSTS Clear** Read always as 0. 0_B No action 1_B Clear SAFEx_AEX.PRSTSTS status bit.
IRQ0AEMC	16	w	**IRQ0AEM Clear** Read always as 0. 0_B No action 1_B Clear SAFEx_AEX.IRQ0AEM status bit.
IRQ1AEMC	17	w	**IRQ1AEM Clear** Read always as 0. 0_B No action 1_B Clear SAFEx_AEX.IRQ1AEM status bit.
IRQ2AEMC	18	w	**IRQ2AEM Clear** Read always as 0. 0_B No action 1_B Clear SAFEx_AEX.IRQ2AEM status bit.
C1RSTAEMC	20	w	**C1RSTAEM Clear** Read always as 0. 0_B No action 1_B Clear SAFEx_AEX.C1RSTAEM status bit.
C2RSTAEMC	21	w	**C2RSTAEM Clear** Read always as 0. 0_B No action 1_B Clear SAFEx_AEX.C2RSTAEM status bit.

(table continues...)

(continued)

Field	Bits	Type	Description
C3RSTAEMC	22	w	**C3RSTAEM Clear** Read always as 0. 0_B No action 1_B Clear SAFEx_AEX.C3RSTAEM status bit.
C4RSTAEMC	23	w	**C4RSTAEM Clear** Read always as 0. 0_B No action 1_B Clear SAFEx_AEX.C4RSTAEM status bit.
C5RSTAEMC	24	w	**C5RSTAEM Clear** Read always as 0. 0_B No action 1_B Clear SAFEx_AEX.C5RSTAEM status bit.
NMI0AEMC	25	w	**NMI0AEM Clear** Read always as 0. 0_B No action 1_B Clear SAFEx_AEX.NMI0AEM status bit.
EMSAEMC	27	w	**EMSAEM Clear** Read always as 0. 0_B No action 1_B Clear SAFEx_AEX.EMSAEM status bit.
NMI1AEMC	28	w	**NMI1AEM Clear** Read always as 0. 0_B No action 1_B Clear SAFEx_AEX.NMI1AEM status bit.
PRSTAEMC	30	w	**PRSTAEM Clear** Read always as 0. 0_B No action 1_B Clear SAFEx_AEX.PRSTAEM status bit.
0	3, 15, 19, 31	r	**Reserved** Read as 0; should be written with 0.

42.4.1.69 SMU_SAFE x RESET0/1 Request Triggering Alarm Information Register

***Note**: Writing to this register has no effect*

SAFEx_RST (x=0-1)

SMU_SAFE x RESET0/1 Request Triggering Alarm Information Register

Offset address: $1058_H+x*500_H$

PowerOn Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0			ALMID1					0			ALMGRP1				
r			rh					r			rh				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			ALMID0					0			ALMGRP0				
r			rh					r			rh				

Field	Bits	Type	Description
ALMGRP0	4:0	rh	**Alarm Group 0** This field indicates the alarm group of the alarm that triggered the RESET0 request
ALMID0	12:8	rh	**Alarm Index 0** This field specifies the alarm index related to the alarm group index ALMGRP which has triggered RESET0 request.
ALMGRP1	20:16	rh	**Alarm Group 1** This field indicates the alarm group of the alarm that triggered the RESET1 request
ALMID1	28:24	rh	**Alarm Index 1** This field specifies the alarm index related to the alarm group index ALMGRP which has triggered RESET1 request.
0	7:5, 15:13, 23:21, 31:29	r	**Reserved** Read as 0; should be written with 0.

42.4.1.70 SMU_SAFE x Group i Alarm Configuration Register j

SAFEx_AGSFi_CONj (i=0;j=0,2;x=0) Offset address: 1080_H+x*500_H+i*24_H +j*4

SMU_SAFE x Group i Alarm Configuration Register j Reset values see: Table 2039

SAFEx_AGSFi_CONj (i=0-1,8;j=1,3;x=0) Offset address: 1080_H+x*500_H+i*24_H +j*4

SMU_SAFE x Group i Alarm Configuration Register j Reset values see: Table 2035

SAFEx_AGSFi_CONj (i=1;j=0,2;x=0) Offset address: 1080_H+x*500_H+i*24_H +j*4

SMU_SAFE x Group i Alarm Configuration Register j Reset values see: Table 2038

SAFEx_AGSFi_CONj (i=2-7,9,11-16;j=0-3;x=0) Offset address: 1080_H+x*500_H+i*24_H +j*4

SMU_SAFE x Group i Alarm Configuration Register j Reset values see: Table 2035

SAFEx_AGSFi_CONj (i=8;j=0,2;x=0) Offset address: 1080_H+x*500_H+i*24_H +j*4

SMU_SAFE x Group i Alarm Configuration Register j Reset values see: Table 2036

SAFEx_AGSFi_CONj (i=10;j=0,3;x=0) Offset address: 1080_H+x*500_H+i*24_H +j*4

SMU_SAFE x Group i Alarm Configuration Register j Reset values see: Table 2035

SAFEx_AGSFi_CONj (i=10;j=1-2;x=0) Offset address: 1080_H+x*500_H+i*24_H +j*4

SMU_SAFE x Group i Alarm Configuration Register j Reset values see: Table 2037

SAFEx_AGSFi_CONj (i=0-16;j=0-3;x=1) Offset address: 1080_H+x*500_H+i*24_H +j*4

SMU_SAFE x Group i Alarm Configuration Register j Reset values see: Table 2035

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CF31	CF30	CF29	CF28	CF27	CF26	CF25	CF24	CF23	CF22	CF21	CF20	CF19	CF18	CF17	CF16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CF15	CF14	CF13	CF12	CF11	CF10	CF9	CF8	CF7	CF6	CF5	CF4	CF3	CF2	CF1	CF0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
CFz (z=0-31)	z	rw	**Configuration flag j (j=0-3) for alarm z in alarm group AGSFi.** The configuration flags 0, 1, 2 and 3 must be used together to define the behavior of the SMU_SAFE when a fault state is reported by the alarm z belonging to this group. 0_B Configuration flag j (j=0-3) is set to 0 1_B Configuration flag j (j=0-3) is set to 1

Table 2034 Access mode restrictions sorted by descending priority

Applies to SAFEx_AGSFi_CONj (i=0;j=0,2;x=0)
Applies to SAFEx_AGSFi_CONj (i=0-1,8;j=1,3;x=0)
Applies to SAFEx_AGSFi_CONj (i=2-7,9,11-16;j=0-3;x=0)
Applies to SAFEx_AGSFi_CONj (i=10;j=0,3;x=0)
Applies to SAFEx_AGSFi_CONj (i=0-16;j=0-3;x=1)
Applies to SAFEx_AGSFi_CONj (i=1;j=0,2;x=0)
Applies to SAFEx_AGSFi_CONj (i=8;j=0,2;x=0)
Applies to SAFEx_AGSFi_CONj (i=10;j=1-2;x=0)

Mode name	Access mode		Description
SAFEx_KEYS.CFGLCK == 0xBC and **SAFEx_KEYS.PERLCK** != 0xFF	rw	CFz (z=0-31)	
(default)	r	CFz (z=0-31)	

Table 2035 Reset values variant 1

Applies to SAFEx_AGSFi_CONj (i=0-1,8;j=1,3;x=0)
Applies to SAFEx_AGSFi_CONj (i=2-7,9,11-16;j=0-3;x=0)
Applies to SAFEx_AGSFi_CONj (i=10;j=0,3;x=0)
Applies to SAFEx_AGSFi_CONj (i=0-16;j=0-3;x=1)

Reset type	Reset value	Note
Application Reset	0000 0000$_H$	Reset value for all alarm configuration registers except SAFE0_AGSF8_CON0, SAFE0_AGSF8_CON2, SAFE0_AGSF10_CON1, SAFE0_AGSF10_CON2, SAFE0_AGSF0_CON0, SAFE0_AGSF0_CON2, SAFE0_AGSF1_CON0 and SAFE0_AGSF1_CON2

Table 2036 Reset values of SAFEx_AGSFi_CONj (i=8;j=0,2;x=0)

Reset type	Reset value	Note
Application Reset	0000 007F$_H$	Reset value for SAFE0_AGSF8_CON0 and SAFE0_AGSF8_CON2 (NMI0 to WDT overflow or access error alarms)

Table 2037 Reset values of SAFEx_AGSFi_CONj (i=10;j=1-2;x=0)

Reset type	Reset value	Note
Application Reset	000C 0000$_H$	Reset value for SAFE0_AGSF10_CON1 and SAFE0_AGSF10_CON2 (RESET0_REQ to Recovery Timers Timeout alarms)

Table 2038 Reset values of SAFEx_AGSFi_CONj (i=1;j=0,2;x=0)

Reset type	Reset value	Note
Application Reset	0000 0800$_H$	Reset value for SAFE0_AGSF1_CON0 and SAFE0_AGSF1_CON2 (NMI0 to VMT0-VMT Uncorrectable Error)

Table 2039 Reset values of SAFEx_AGSFi_CONj (i=0;j=0,2;x=0)

Reset type	Reset value	Note
Application Reset	0000 0001$_H$	Reset value for SAFE0_AGSF0_CON0 and SAFE0_AGSF0_CON2 (NMI0 to CPU0 Lockstep alarm)

42.4.1.71 SMU_SAFE x Group i FSP j Alarm Configuration Register

SAFEx_AGSFi_FSPj (i=0-9,11-16;j=0-1;x=0) Offset address: 1090_H+x*500_H+i*24_H +j*4

SMU_SAFE x Group i FSP j Alarm Configuration Register Reset values see: Table 2042

SAFEx_AGSFi_FSPj (i=10;j=0;x=0) Offset address: 1090_H+x*500_H+i*24_H +j*4

SMU_SAFE x Group i FSP j Alarm Configuration Register Reset values see: Table 2041

SAFEx_AGSFi_FSPj (i=10;j=1;x=0) Offset address: 1090_H+x*500_H+i*24_H +j*4

SMU_SAFE x Group i FSP j Alarm Configuration Register Reset values see: Table 2042

SAFEx_AGSFi_FSPj (i=0-16;j=0-1;x=1) Offset address: 1090_H+x*500_H+i*24_H +j*4

SMU_SAFE x Group i FSP j Alarm Configuration Register Reset values see: Table 2042

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FE31	**FE30**	**FE29**	**FE28**	**FE27**	**FE26**	**FE25**	**FE24**	**FE23**	**FE22**	**FE21**	**FE20**	**FE19**	**FE18**	**FE17**	**FE16**
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FE15	**FE14**	**FE13**	**FE12**	**FE11**	**FE10**	**FE9**	**FE8**	**FE7**	**FE6**	**FE5**	**FE4**	**FE3**	**FE2**	**FE1**	**FE0**
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
FEz (z=0-31)	z	rw	**FSP configuration flag j (j=0-1) for alarm z in alarm group AGSFi** The configuration flags 0 and 1 must be used to trigger corresponding FSP[0] and FSP[1] when a fault state is reported by the alarm z belonging to this group. 0_B FSP[j] (j=0-1) disabled for this alarm event 1_B FSP[j] (j=0-1) enabled for this alarm event

Table 2040 Access mode restrictions sorted by descending priority

Applies to SAFEx_AGSFi_FSPj (i=0-9,11-16;j=0-1;x=0)
Applies to SAFEx_AGSFi_FSPj (i=10;j=1;x=0)
Applies to SAFEx_AGSFi_FSPj (i=0-16;j=0-1;x=1)
Applies to SAFEx_AGSFi_FSPj (i=10;j=0;x=0)

Mode name	Access mode		Description
SAFEx_KEYS.CFGLCK == 0xBC and **SAFEx_KEYS.PERLCK** != 0xFF	rw	FEz (z=0-31)	
(default)	r	FEz (z=0-31)	

Table 2041 Reset values of SAFEx_AGSFi_FSPj (i=10;j=0;x=0)

Reset type	Reset value	Note
Application Reset	000C 0000$_H$	Reset value for SAFE0_AGSF10_FSP0 (FSP0 to Recovery Timers Timeout alarms)

Table 2042 Reset values variant 1

Applies to SAFEx_AGSFi_FSPj (i=0-9,11-16;j=0-1;x=0)
Applies to SAFEx_AGSFi_FSPj (i=10;j=1;x=0)
Applies to SAFEx_AGSFi_FSPj (i=0-16;j=0-1;x=1)

Reset type	Reset value	Note
Application Reset	0000 0000$_H$	Reset value for all alarm FSP configuration registers except SAFE0_AGSF10_FSP0

42.4.1.72 SMU_SAFE x Group i Alarm Status Register

Refer to SMU_SAFE Alarm Status Registers section for the conditions to set and reset the status flag by software.

SAFEx_AGSFi_STS (i=0-16;x=0-1) Offset address: 1098$_H$+x*500$_H$+i*24$_H$

SMU_SAFE x Group i Alarm Status Register Application Reset value: 0000 0000$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SF31	**SF30**	**SF29**	**SF28**	**SF27**	**SF26**	**SF25**	**SF24**	**SF23**	**SF22**	**SF21**	**SF20**	**SF19**	**SF18**	**SF17**	**SF16**
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SF15	**SF14**	**SF13**	**SF12**	**SF11**	**SF10**	**SF9**	**SF8**	**SF7**	**SF6**	**SF5**	**SF4**	**SF3**	**SF2**	**SF1**	**SF0**
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
SFz (z=0-31)	z	rwh	**Status flag for alarm z in alarm group AGSFi.** 0_B Status flag z does not report a fault condition 1_B Status flag z reports a fault condition

42.4.1.73 SMU_SAFE x Group i Alarm Debug Register

***Note**: Writing to this register has no effect*

SAFEx_AGSFi_DBG (i=0-16;x=0-1) Offset address: $109C_H+x*500_H+i*24_H$

SMU_SAFE x Group i Alarm Debug Register PowerOn Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DF31	DF30	DF29	DF28	DF27	DF26	DF25	DF24	DF23	DF22	DF21	DF20	DF19	DF18	DF17	DF16
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DF15	DF14	DF13	DF12	DF11	DF10	DF9	DF8	DF7	DF6	DF5	DF4	DF3	DF2	DF1	DF0
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
DFz (z=0-31)	z	rh	**Diagnosis flag for alarm z in alarm group AGSFi.** The diagnosis registers make a snapshot of the alarm group status registers when either the executed alarm reaction is a reset or a state machine transition to FAULT state takes place. 0_B Status flag z does not report a fault condition 1_B Status flag z reports a fault condition

42.4.1.74 SMU_SAFE x Group i Critical Alarm Selection Flag

SAFEx_AGSFi_CAS (i=0-16;x=0-1) Offset address: $10A0_H+x*500_H+i*24_H$

SMU_SAFE x Group i Critical Alarm Selection Flag Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CF31	CF30	CF29	CF28	CF27	CF26	CF25	CF24	CF23	CF22	CF21	CF20	CF19	CF18	CF17	CF16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CF15	CF14	CF13	CF12	CF11	CF10	CF9	CF8	CF7	CF6	CF5	CF4	CF3	CF2	CF1	CF0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
CFz (z=0-31)	z	rw	**Critical Alarm Selection flag for alarm z in alarm group AGSFi.** 0_B Critical alarm disabled for this alarm event 1_B Critical alarm enabled for this alarm event

Table 2043 Access mode restrictions of SAFEx_AGSFi_CAS (i=0-16;x=0-1) sorted by descending priority

Mode name	Access mode		Description
SAFEx_KEYS.CFGLCK == 0xBC and **SAFEx_KEYS.PERLCK** != 0xFF	rw	CFz (z=0-31)	
(default)	r	CFz (z=0-31)	

42.4.1.75 SMU_SAFE x Group i Shared Alarm Configuration Register j

SAFEx_AGSHi_CONj (i=0-2;j=0-3;x=0-1)

Offset address: $1300_H+x*500_H+i*28_H+j*4$

SMU_SAFE x Group i Shared Alarm Configuration Register j

Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CF31	**CF30**	**CF29**	**CF28**	**CF27**	**CF26**	**CF25**	**CF24**	**CF23**	**CF22**	**CF21**	**CF20**	**CF19**	**CF18**	**CF17**	**CF16**
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CF15	**CF14**	**CF13**	**CF12**	**CF11**	**CF10**	**CF9**	**CF8**	**CF7**	**CF6**	**CF5**	**CF4**	**CF3**	**CF2**	**CF1**	**CF0**
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
CFz (z=0-31)	z	rw	**Configuration flag j (j=0-3) for alarm z in alarm group AGSHi.** The configuration flags 0, 1, 2 and 3 must be used together to define the behavior of the SMU_SAFE when a fault state is reported by the alarm z belonging to this group. 0_B Configuration flag j (j=0-3) is set to 0 1_B Configuration flag j (j=0-3) is set to 1

Table 2044 Access mode restrictions of SAFEx_AGSHi_CONj (i=0-2;j=0-3;x=0-1) sorted by descending priority

Mode name	Access mode		Description
SAFEx_KEYS.CFGLCK == 0xBC and **SAFEx_KEYS.PERLCK** != 0xFF	rw	CFz (z=0-31)	
(default)	r	CFz (z=0-31)	

42.4.1.76 SMU_SAFE x Group i FSP j Shared Alarm Configuration Register

SAFEx_AGSHi_FSPj (i=0-2;j=0-1;x=0-1)

Offset address: $1310_H+x*500_H+i*28_H+j*4$

SMU_SAFE x Group i FSP j Shared Alarm Configuration Register

Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FE31	FE30	FE29	FE28	FE27	FE26	FE25	FE24	FE23	FE22	FE21	FE20	FE19	FE18	FE17	FE16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FE15	FE14	FE13	FE12	FE11	FE10	FE9	FE8	FE7	FE6	FE5	FE4	FE3	FE2	FE1	FE0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
FEz (z=0-31)	z	rw	**FSP configuration flag for alarm z in alarm group AGSHi.** The configuration flags 0 and 1 must be used to trigger corresponding FSP[0] and FSP[1] when a fault state is reported by the alarm z belonging to this group. 0_B FSP[j] (j=0-1) disabled for this alarm event 1_B FSP[j] (j=0-1) enabled for this alarm event

Table 2045 Access mode restrictions of SAFEx_AGSHi_FSPj (i=0-2;j=0-1;x=0-1) sorted by descending priority

Mode name	Access mode		Description
SAFEx_KEYS.CFGLCK == 0xBC and **SAFEx_KEYS.PERLCK** != 0xFF	rw	FEz (z=0-31)	
(default)	r	FEz (z=0-31)	

42.4.1.77 SMU_SAFE x Group i Shared Alarm Status Register

Refer to SMU_SAFE Alarm Status Registers section for the conditions to set and reset the status flag by software.

SAFEx_AGSHi_STS (i=0-2;x=0-1) Offset address: $1318_H+x*500_H+i*28_H$

SMU_SAFE x Group i Shared Alarm Status Register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SF31	SF30	SF29	SF28	SF27	SF26	SF25	SF24	SF23	SF22	SF21	SF20	SF19	SF18	SF17	SF16
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SF15	SF14	SF13	SF12	SF11	SF10	SF9	SF8	SF7	SF6	SF5	SF4	SF3	SF2	SF1	SF0
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
SFz (z=0-31)	z	rwh	**Status flag for alarm z in alarm group AGSHi.** 0_B Status flag z does not report a fault condition 1_B Status flag z reports a fault condition

42.4.1.78 SMU_SAFE x Group i Shared Alarm Debug Register

***Note**: Writing to this register has no effect*

SAFEx_AGSHi_DBG (i=0-2;x=0-1) Offset address: $131C_H+x*500_H+i*28_H$

SMU_SAFE x Group i Shared Alarm Debug Register PowerOn Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DF31	DF30	DF29	DF28	DF27	DF26	DF25	DF24	DF23	DF22	DF21	DF20	DF19	DF18	DF17	DF16
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DF15	DF14	DF13	DF12	DF11	DF10	DF9	DF8	DF7	DF6	DF5	DF4	DF3	DF2	DF1	DF0
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
DFz (z=0-31)	z	rh	**Diagnosis flag for alarm z in alarm group AGSHi.** The diagnosis registers make a snapshot of the alarm group status registers when either the executed alarm reaction is a reset or a state machine transition to FAULT state takes place. 0_B Status flag z does not report a fault condition 1_B Status flag z reports a fault condition

42.4.1.79 SMU_SAFE x Group i Critical Shared Alarm Selection Flag

SAFEx_AGSHi_CAS (i=0-2;x=0-1) Offset address: $1320_H+x*500_H+i*28_H$

SMU_SAFE x Group i Critical Shared Alarm Selection Flag Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CF31	CF30	CF29	CF28	CF27	CF26	CF25	CF24	CF23	CF22	CF21	CF20	CF19	CF18	CF17	CF16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CF15	CF14	CF13	CF12	CF11	CF10	CF9	CF8	CF7	CF6	CF5	CF4	CF3	CF2	CF1	CF0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
CFz (z=0-31)	z	rw	**Critical Alarm Selection flag for alarm z in alarm group AGSHi.** 0_B Critical alarm disabled for this alarm event 1_B Critical alarm enabled for this alarm event

Table 2046 Access mode restrictions of SAFEx_AGSHi_CAS (i=0-2;x=0-1) sorted by descending priority

Mode name	Access mode		Description
SAFEx_KEYS.CFGL CK == 0xBC and **SAFEx_KEYS.PERL CK** != 0xFF	rw	CFz (z=0-31)	
(default)	r	CFz (z=0-31)	

42.4.1.80 SMU_SAFE x Group i Shared Alarm Group Selection Status

***Note**: Holds exact copy of Register GCC_AGSHi_SEL. Writing to this register has no effect*

SAFEx_AGSHi_SS (i=0-2;x=0-1) Offset address: $1324_H+x*500_H+i*28_H$

SMU_SAFE x Group i Shared Alarm Group Selection Status PowerOn Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SF31	**SF30**	**SF29**	**SF28**	**SF27**	**SF26**	**SF25**	**SF24**	**SF23**	**SF22**	**SF21**	**SF20**	**SF19**	**SF18**	**SF17**	**SF16**
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SF15	**SF14**	**SF13**	**SF12**	**SF11**	**SF10**	**SF9**	**SF8**	**SF7**	**SF6**	**SF5**	**SF4**	**SF3**	**SF2**	**SF1**	**SF0**
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
SFz (z=0-31)	z	rh	**Shared Alarm Group status flag for Alarm Group AGSHi. - CFz** The status flag is used to indicate which alarm in the shared alarm group is mapped to SMU_SAFE. 0_B Status flag z indicates Alarm z is not mapped 1_B Status flag z indicates Alarm z is mapped

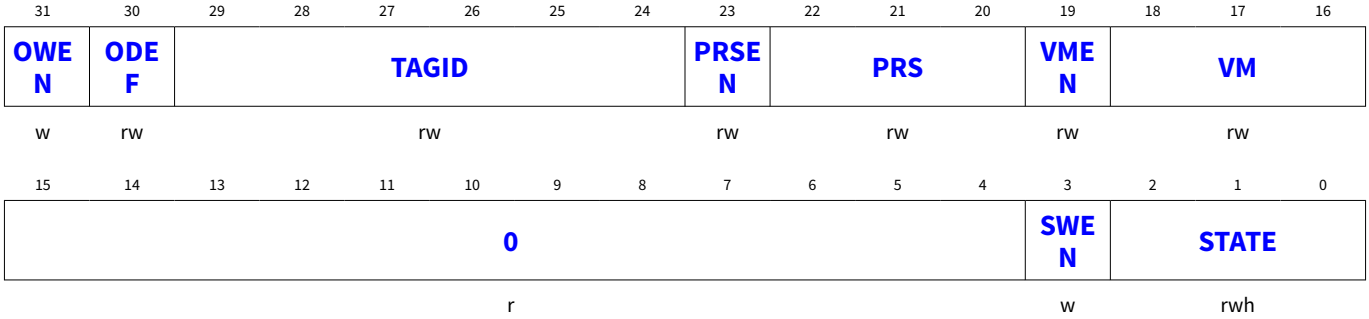
42.4.1.81 SMU_SAFE x PROT Register Endinit

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner and in some PROT states, the secure master to update the PROT state as described in the PROT mechanism.

The PROTSE register controls lock / unlock of the local Endinit (SxE) protected control registers.

SAFEx_PROTSE (x=0-1) Offset address: $13FC_H+x*500_H$

SMU_SAFE x PROT Register Endinit Application Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
VM	18:16	rw	**Virtual Machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual Machine definition Enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection Set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.
PRSEN	23	rw	**Protection Set definition Enable for PROT owner** 0_B **Disable**: PRS is part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner

(table continues...)

(continued)

Field	Bits	Type	Description
OWEN	31	w	**Owner write enable** 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; should be written with 0.

Table 2047 Access mode restrictions of SAFEx_PROTSE (x=0-1) sorted by descending priority

Mode name	Access mode		Description
write 1 to .**SWEN**	rwh	STATE	
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

42.4.1.82 SMU_SAFE x PROT Register Safe Endinit

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner and in some PROT states, the secure master to update the PROT state as described in the PROT mechanism.

The PROTSSE register controls lock / unlock of the local Safe Endinit (SxSE) protected control registers.

SAFEx_PROTSSE (x=0-1) Offset address: $1400_H+x*500_H$

SMU_SAFE x PROT Register Safe Endinit Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OWEN	**ODEF**	**TAGID**						**PRSEN**	**PRS**			**VMEN**	**VM**		
w	rw	rw						rw	rw			rw	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												**SWEN**	**STATE**		
r												w	rwh		

Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
VM	18:16	rw	**Virtual Machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual Machine definition Enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection Set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.
PRSEN	23	rw	**Protection Set definition Enable for PROT owner** 0_B **Disable**: PRS is part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner
OWEN	31	w	**Owner write enable** 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; should be written with 0.

Table 2048 Access mode restrictions of SAFEx_PROTSSE (x=0-1) sorted by descending priority

Mode name	Access mode		Description
write 1 to .**SWEN**	rwh	STATE	
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

42.4.1.83 SMU_SAFE x Write access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

SAFEx_ACCEN_WRA (x=0-1) Offset address: $1404_H+x*500_H$

SMU_SAFE x Write access enable register A Application Reset value: $1000\ 0003_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	**EN30**	**EN29**	**EN28**	**EN27**	**EN26**	**EN25**	**EN24**	**EN23**	**EN22**	**EN21**	**EN20**	**EN19**	**EN18**	**EN17**	**EN16**
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	**EN14**	**EN13**	**EN12**	**EN11**	**EN10**	**EN09**	**EN08**	**EN07**	**EN06**	**EN05**	**EN04**	**EN03**	**EN02**	**EN01**	**EN00**
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access

42.4.1.84 SMU_SAFE x Write access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

SAFEx_ACCEN_WRB (x=0-1) Offset address: $1408_H+x*500_H$

SMU_SAFE x Write access enable register B Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								**EN39**	**EN38**	**EN37**	**EN36**	**EN35**	**EN34**	**EN33**	**EN32**
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access
0	31:8	r	**Reserved** Read as 0; should be written with 0.

42.4.1.85 SMU_SAFE x Read access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

SAFEx_ACCEN_RDA (x=0-1) Offset address: $140C_H+x*500_H$

SMU_SAFE x Read access enable register A Application Reset value: FFFF FFFF$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access

42.4.1.86 SMU_SAFE x Read access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

SAFEx_ACCEN_RDB (x=0-1) Offset address: $1410_H+x*500_H$

SMU_SAFE x Read access enable register B Application Reset value: 0000 00FF$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access
0	31:8	r	**Reserved** Read as 0; should be written with 0.

42.4.1.87 SMU_SAFE x VM access enable register

This register defines which virtual machine encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each VMx.

SAFEx_ACCEN_VM (x=0-1) Offset address: $1414_H+x*500_H$

SMU_SAFE x VM access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for VM ID q** This bit enables read access to the access protected region for transactions with the VM ID q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for VM ID q** This bit enables write access to the access protected region for transactions with the VM ID q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

42.4.1.88 SMU_SAFE x PRS access enable register

This register defines which protection register set encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each PRSx.

SAFEx_ACCEN_PRS (x=0-1) Offset address: $1418_H+x*500_H$

SMU_SAFE x PRS access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for PRS q** This bit enables read access to the access protected region for transactions with the PRS q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for PRS q** This bit enables write access to the access protected region for transactions with the PRS q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

42.4.2 Registers of SMU_STANDBY

This chapter shows specific information related to the SMU_STDBY registers.

42.4.2.1 Register overview - access mode glossary

Table 2049 Register overview - access mode glossary

Keyword	Description
E	Access protection using PROT register PROTSTDBYE.
SE	Access protection using PROT register PROTSTDBYSE.
APU-P	Protection group consisting of registers ACCENSTDBY_WRA, ACCENSTDBY_WRB, ACCENSTDBY_RDA, ACCENSTDBY_RDB, ACCENSTDBY_VM, ACCENSTDBY_PRS.
P	Access protection using *APU-P* registers.
BE	Always returns a Bus Error.
U	No access restrictions.
PROT	Access restrictions as defined in the PROT register access rules.

42.4.2.2 Registers overview - SMU_STDBY (ascending offset address)

Table 2050 Registers overview - SMU_STDBY (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
MONBISTSTAT	SMU_STDBY BIST Status Register	9814_H	P	BE	See 7201	7201
MONBISTCTRL	SMU_STDBY BIST Control Register	9818_H	P	SV, P, E	See 7202	7202
CMDSTDBY	SMU_STDBY Command Register	$981C_H$	P	SV, P, E	See 7204	7204
AGSTDBYi_FSPj	SMU_STDBY Group i FSP j Configuration Register	9820_H+i* 14_H+j*4	P	SV, P, E	See 7206	7206
AGSTDBYi_STS	SMU_STDBY Group i Alarm Status Register	9830_H+i* 14_H	P	SV, P	See 7208	7208
PROTSTDBYE	SMU_STDBY PROT Register Endinit	$989C_H$	U	SV, PROT	Application Reset	7213
PROTSTDBYSE	SMU_STDBY PROT Register Safe Endinit	$98A0_H$	U	SV, PROT	Application Reset	7214
ACCENSTDBY_WRA	SMU_STDBY Write access enable register A	$98A4_H$	U	SE, SV	Application Reset	7216
ACCENSTDBY_WRB	SMU_STDBY Write access enable register B	$98A8_H$	U	SE, SV	Application Reset	7216
ACCENSTDBY_RDA	SMU_STDBY Read access enable register A	$98AC_H$	U	SE, SV	Application Reset	7217
ACCENSTDBY_RDB	SMU_STDBY Read access enable register B	$98B0_H$	U	SE, SV	Application Reset	7217
ACCENSTDBY_VM	SMU_STDBY VM access enable register	$98B4_H$	U	SE, SV	Application Reset	7218
ACCENSTDBY_PRS	SMU_STDBY PRS access enable register	$98B8_H$	U	SE, SV	Application Reset	7218

42.4.2.3 SMU_STDBY BIST Status Register

Refer to MONBIST section in PMS chapter.

MONBISTSTAT Offset address: 9814_H

SMU_STDBY BIST Status Register Reset values see: Table 2051

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0							SMUERR3	SMUERR2	SMUERR1	SMUERR0	PMSERR	TSTDONE	TSTRUN	0	TSTOK
r							rh	rh	rh	rh	rh	rh	rh	r	rh

Field	Bits	Type	Description
TSTOK	0	rh	**SMU_stdby BIST ok bit** This status bit indicates that MONBIST test was successful and the result is as expected. 0_B The MONBIST test result is not ok. 1_B The MONBIST test result is ok.
TSTRUN	2	rh	**SMU_stdby BIST run bit** The status bit indicates that MONBIST test is ongoing. 0_B The MONBIST test is currently inactive. 1_B The MONBIST test is active.
TSTDONE	3	rh	**SMU_stdby BIST done bit** The status bit indicates that MONBIST test is completed. 0_B The MONBIST test is not started. 1_B The MONBIST test is done.
PMSERR	4	rh	**Error found in PMS SARADC by SMU_stdby BIST** This status bit indicates that SMU_stdby BIST found an error in the PMS SARADC. 0_B No error happened in PMS module during MONBIST test. 1_B This bit is set if MONBIST test failed and the error occurred in Secondary voltage monitor and alarm generation in PMS. This status bit is cleared by setting TSTCLR bit.
SMUERRz (z=0-3)	z+5	rh	**Error found in SMU_stdby found by SMU_stdby BIST** This status bit indicates that the MONBIST test found an error in the SMU_stdby. 0_B No error happened in SMU_STDBY module during MONBIST test. 1_B This bit is set if MONBIST test failed and the error occurred in SMU_STDBY module and alarm processing path. This status bit is cleared by setting TSTCLR bit..
0	1, 31:9	r	**Reserved** Read as 0; should be written with 0.

Table 2051 Reset values of MONBISTSTAT

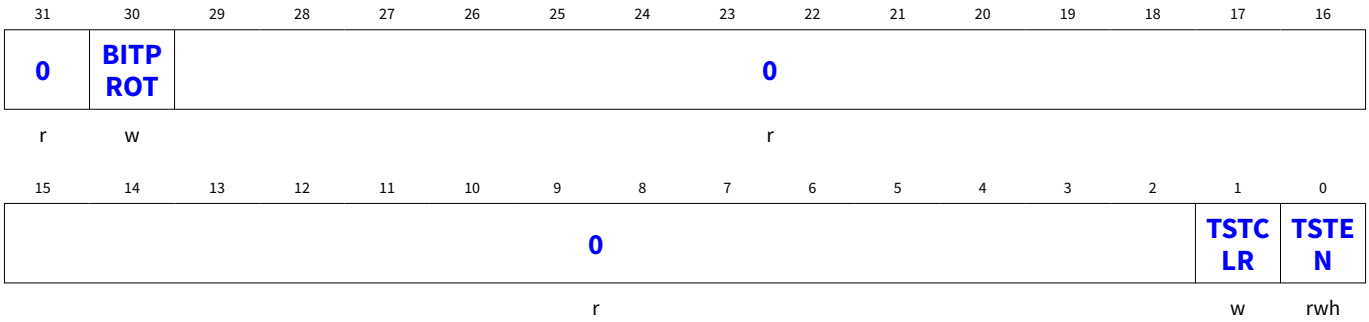
Reset type	Reset value	Note
LVDPMS2 Reset	$0000\ 0000_H$	
Warm PORST	$0000\ 0000_H$	

42.4.2.4 SMU_STDBY BIST Control Register

Refer to MONBIST section in PMS chapter.

MONBISTCTRL Offset address: 9818_H

SMU_STDBY BIST Control Register Reset values see: Table 2053



Field	Bits	Type	Description
TSTEN	0	rwh	**SMU_stdby alarm BIST enable** If SMUEN in the CMD_stdby register is set to 1, setting the TSTEN bit triggers SMU_stdby BIST tests to test the alarm path and PMS components. This bit can only be changed if bit BITPROT is set in parallel. This bit is cleared by hardware at the end of the SMU_stdby BIST operation. 0_B SMU_stdby BIST test is inactive 1_B SMU_stdby BIST test triggered. When the SMU_stdby BIST is activated, the alarm signals tested by the bist are not propagated to the SMU_SAFE0/1.
TSTCLR	1	w	**SMU_stdby BIST flag clear** Setting this bit enables the clearing of the bitfields TSTOK, TSTDONE, TSTRUN, SMUERR and PMSERR of register MONBISTSTAT. 0_B No action 1_B The bitfields TSTOK, TSTDONE, TSTRUN, SMUERR and PMSERR of register MONBISTSTAT are cleared. Before triggering SMU_stdby BIST mechanism with TSTEN, previous test results need to be cleared by TSTCLR.
BITPROT	30	w	**Bit Protection TSTEN** Setting this bit enables that bit TSTEN can be changed in this write operation. This bit is read as zero.
0	29:2, 31	r	**Reserved** Read as 0; should be written with 0.

Table 2052 Access mode restrictions of MONBISTCTRL sorted by descending priority

Mode name	Access mode		Description
write 1 to .**BITPROT**	rwh	TSTEN	
(default)	rh	TSTEN	

Table 2053 Reset values of MONBISTCTRL

Reset type	Reset value	Note
LVDPMS2 Reset	$0000\ 0000_H$	
Warm PORST	$0000\ 0000_H$	

42.4.2.5 SMU_STDBY Command Register

CMDSTDBY Offset address: $981C_H$

SMU_STDBY Command Register Reset values see: Table 2055

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	BITPROT	0													
r	w	r													

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0						ASCE	FSP3PD	FSP2PD	FSP1PD	FSP0PD	FSP3EN	FSP2EN	FSP1EN	FSP0EN	SMUEN
r						rwh	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
SMUEN	0	rw	**SMU_stdby Module Enable** This bit enables SMU_stdby to issue a FSP reaction when an alarm is received. Also, SMUEN needs to be set to enter the SMU_stdby BIST mode. 0_B SMU_stdby disabled. 1_B SMU_stdby enabled.
FSP0EN	1	rw	**SMU_stdby FSP0 Error pin enable** This bit enables SMU_stdby Error pin function to be able set P33.8 to fault state. 0_B SMU_stdby Error Pin fault indication function on P33.8 inactive. 1_B SMU_stdby Error Pin fault indication function on P33.8 active.
FSP1EN	2	rw	**SMU_stdby FSP1 Error pin enable** This bit enables SMU_stdby Error pin function to be able set P33.12 to fault state. 0_B SMU_stdby Error Pin fault indication function on P33.12 inactive. 1_B SMU_stdby Error Pin fault indication function on P33.12 active.
FSP2EN	3	rw	**SMU_stdby FSP2 Error pin enable** This bit enables SMU_stdby Error pin function to be able set P32.2 to fault state. 0_B SMU_stdby Error Pin fault indication function on P32.2 inactive. 1_B SMU_stdby Error Pin fault indication function on P32.2 active.
FSP3EN	4	rw	**SMU_stdby FSP3 Error pin enable** This bit enables SMU_stdby Error pin function to be able set P32.5 to fault state. 0_B SMU_stdby Error Pin fault indication function on P32.5 inactive. 1_B SMU_stdby Error Pin fault indication function on P32.5 active.

(table continues...)

(continued)

Field	Bits	Type	Description
FSP0PD	5	rw	**SMU_stdby FSP0 Error pull-down** This bit defines the error reaction of SMU_stdby ErrorPin0 (P33.8). 0_B Pulled down state inactive. 1_B *Pulled down state active.*
FSP1PD	6	rw	**SMU_stdby FSP1 Error pull-down** This bit defines the error reaction of SMU_stdby ErrorPin1 (P33.12). 0_B Pulled down state inactive. 1_B *Pulled down state active.*
FSP2PD	7	rw	**SMU_stdby FSP2 Error pull-down** This bit defines the error reaction of SMU_stdby ErrorPin2 (P32.7). 0_B Pulled down state inactive. 1_B *Pulled down state active.*
FSP3PD	8	rw	**SMU_stdby FSP3 Error pull-down** This bit defines the error reaction of SMU_stdby ErrorPin3 (P32.5). 0_B Pulled down state inactive. 1_B *Pulled down state active.*
ASCE	9	rwh	**SMU_stdby alarm status clear enable** This bit controls if a status flag set in an AGSTDBYi_STS register upon detection of the alarm event can be cleared by software or not. When ASCE is enabled, software shall write a 1 to bit position in AGSTDBYi_STS to clear the bit (W1C). When a W1C action takes place the ASCE bit is automatically cleared to 0 by hardware and software shall set the ASCE bit again. 0_B SMU_stdby alarm status bits in AGSTDBYi_STS cannot be cleared. 1_B SMU_stdby alarm status bits in AGSTDBYi_STS can be cleared
BITPROT	30	w	**CMDSTDBY register bits protection** Setting this bit enables the other bits in the registers (SMUEN, FSP0EN, FSP1EN, FSP2EN, FSP3EN, FSP0PD, FSP1PD, FSP2PD, FSP3PD, ASCE) to be changed in the same write operation. This bit is read as zero.
0	29:10, 31	r	**Reserved** Read as 0; should be written with 0.

Table 2054 Access mode restrictions of CMDSTDBY sorted by descending priority

Mode name	Access mode		Description
write 1 to .**BITPROT**	rw	FSP0EN, FSP0PD, FSP1EN, FSP1PD, FSP2EN, FSP2PD, FSP3EN, FSP3PD, SMUEN	
	rwh	ASCE	
(default)	r	FSP0EN, FSP0PD, FSP1EN, FSP1PD, FSP2EN, FSP2PD, FSP3EN, FSP3PD, SMUEN	
	rh	ASCE	

Table 2055 Reset values of CMDSTDBY

Reset type	Reset value	Note
LVDPMS2 Reset	0000 0000$_H$	
Warm PORST	0000 0000$_H$	

42.4.2.6 SMU_STDBY Group i FSP j Configuration Register (i=0-1;j=0-3)

AGSTDBYi_FSPj (i=0-1;j=0-3) Offset address: 9820$_H$+i*14$_H$+j*4

SMU_STDBY Group i FSP j Configuration Register Reset values see: Table 2057

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	**BITPROT**	**0**		**FE27**	**FE26**	**FE25**	**FE24**	**FE23**	**FE22**	**FE21**	**FE20**	**FE19**	**FE18**	**FE17**	**FE16**
r	w	r		rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FE15	**FE14**	**FE13**	**FE12**	**FE11**	**FE10**	**FE9**	**FE8**	**FE7**	**FE6**	**FE5**	**FE4**	**FE3**	**FE2**	**FE1**	**FE0**
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
FEz (z=0-19)	z	rw	**FSP configuration flag for alarm z in alarm group AGSTDBYi.** 0_B FSP disabled for this alarm event. 1_B FSP enabled for this alarm event.
FEz (z=20-27)	z	rw	**FSP configuration flag for alarm z in alarm group AGSTDBYi.** 0_B FSP disabled for this alarm event. 1_B FSP enabled for this alarm event.
BITPROT	30	w	**AG2iFSP_STDBY register bits protection** Setting this bit enables the other bits in the registers (FEz) to be changed in the same write operation. This bit is read as zero.
0	27, 26, 25, 24, 23, 22, 21, 20, 29:28, 31	r	**Reserved** Read as 0; should be written with 0.

Table 2056 Access mode restrictions of AGSTDBYi_FSPj (i=0-1;j=0-3) sorted by descending priority

Mode name	Access mode		Description
write 1 to .**BITPROT**	rw	FEz (z=0-19), FEz (z=20-27)	
(default)	r	FEz (z=0-19), FEz (z=20-27)	

Table 2057 Reset values of AGSTDBYi_FSPj (i=0-1;j=0-3)

Reset type	Reset value	Note
LVDPMS2 Reset	0000 0000$_H$	
Warm PORST	0000 0000$_H$	

42.4.2.7 SMU_STDBY Group i FSP j Configuration Register (i=2-3;j=0-3)

AGSTDBYi_FSPj (i=2-3;j=0-3) Offset address: 9820_H+i*14_H+j*4

SMU_STDBY Group i FSP j Configuration Register Reset values see: Table 2059

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	**BITPROT**	**0**		**FE27**	**FE26**	**FE25**	**FE24**	**FE23**	**FE22**	**FE21**	**FE20**	**FE19**	**FE18**	**FE17**	**FE16**
r	w	r		rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FE15	**FE14**	**FE13**	**FE12**	**FE11**	**FE10**	**FE9**	**FE8**	**FE7**	**FE6**	**FE5**	**FE4**	**FE3**	**FE2**	**FE1**	**FE0**
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
FEz (z=0-22)	z	rw	**FSP configuration flag for alarm z in alarm group AGSTDBYi.** 0_B FSP disabled for this alarm event. 1_B FSP enabled for this alarm event.
FEz (z=23-27)	z	rw	**FSP configuration flag for alarm z in alarm group AGSTDBYi.** 0_B FSP disabled for this alarm event. 1_B FSP enabled for this alarm event.
BITPROT	30	w	**AG2iFSP_STDBY register bits protection** Setting this bit enables the other bits in the registers (FEz) to be changed in the same write operation. This bit is read as zero.
0	27, 26, 25, 24, 23, 29:28, 31	r	**Reserved** Read as 0; should be written with 0.

Table 2058 Access mode restrictions of AGSTDBYi_FSPj (i=2-3;j=0-3) sorted by descending priority

Mode name	Access mode		Description
write 1 to .**BITPROT**	rw	FEz (z=0-22), FEz (z=23-27)	
(default)	r	FEz (z=0-22), FEz (z=23-27)	

Table 2059 Reset values of AGSTDBYi_FSPj (i=2-3;j=0-3)

Reset type	Reset value	Note
LVDPMS2 Reset	$0000\ 0000_H$	
Warm PORST	$0000\ 0000_H$	

42.4.2.8 SMU_STDBY Group i FSP j Configuration Register (i=4;j=0-3)

AGSTDBYi_FSPj (i=4;j=0-3) Offset address: 9820_H+i*14_H+j*4

SMU_STDBY Group i FSP j Configuration Register Reset values see: Table 2061

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	**BITPROT**	**0**		**FE27 0**	**FE26 0**	**FE25 0**	**FE24 0**	**FE23 0**	**FE22 0**	**FE21 0**	**FE20 0**	**FE19 0**	**FE18**	**FE17**	**FE16**
r	w	r		rw r	rw r	rw r	rw r	rw r	rw r	rw r	rw r	rw r	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FE15	**FE14**	**FE13**	**FE12**	**FE11**	**FE10**	**FE9**	**FE8**	**FE7**	**FE6**	**FE5**	**FE4**	**FE3**	**FE2**	**FE1**	**FE0**
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
FEz (z=0-18)	z	rw	**FSP configuration flag for alarm z in alarm group AGSTDBYi.** 0_B FSP disabled for this alarm event. 1_B FSP enabled for this alarm event.
FEz (z=19-27)	z	rw	**FSP configuration flag for alarm z in alarm group AGSTDBYi.** 0_B FSP disabled for this alarm event. 1_B FSP enabled for this alarm event.
BITPROT	30	w	**AG2iFSP_STDBY register bits protection** Setting this bit enables the other bits in the registers (FEz) to be changed in the same write operation. This bit is read as zero.
0	27, 26, 25, 24, 23, 22, 21, 20, 19, 29:28, 31	r	**Reserved** Read as 0; should be written with 0.

Table 2060 Access mode restrictions of AGSTDBYi_FSPj (i=4;j=0-3) sorted by descending priority

Mode name	Access mode		Description
write 1 to .**BITPROT**	rw	FEz (z=0-18), FEz (z=19-27)	
(default)	r	FEz (z=0-18), FEz (z=19-27)	

Table 2061 Reset values of AGSTDBYi_FSPj (i=4;j=0-3)

Reset type	Reset value	Note
LVDPMS2 Reset	0000 0000_H	
Warm PORST	0000 0000_H	

42.4.2.9 SMU_STDBY Group i Alarm Status Register (i=0)

AGSTDBYi_STS (i=0) Offset address: 9830_H+i*14_H

SMU_STDBY Group i Alarm Status Register Reset values see: Table 2063

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FSP3 ERR	**FSP2 ERR**	**FSP1 ERR**	**FSP0 ERR**	**SF27 0**	**SF26 0**	**SF25 0**	**SF24 0**	**SF23 0**	**SF22 0**	**SF21 0**	**SF20 0**	**SF19**	**SF18**	**SF17**	**SF16**
rwh	rwh	rwh	rwh	rwh r	rwh r	rwh r	rwh r	rwh r	rwh r	rwh r	rwh r	rwh	rwh	rwh	rwh
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SF15	**SF14**	**SF13**	**SF12**	**SF11**	**SF10**	**SF9**	**SF8**	**SF7**	**SF6**	**SF5**	**SF4**	**SF3**	**SF2**	**SF1**	**SF0**
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
SFz (z=0-19)	z	rwh	**Status flag for alarm z belonging to alarm group i.** 0_B Status flag z does not report a fault condition 1_B Status flag z reports a fault condition
SFz (z=20-27)	z	rwh	**Status flag for alarm z belonging to alarm group i.** 0_B Status flag z does not report a fault condition 1_B Status flag z reports a fault condition
FSP0ERR	28	rwh	**FSP0 Error Pin Fault State Status Bit** The bit indicates that Error pin was set into fault state by the SMU_stdby. Reset by setting to 1. If the Error Pins were set in fault state by the SMU_stdby, reseting this bit sets the Error Pins back in fault free state 0_B Error pin was not set into fault state 1_B The Error pin was set into fault state.
FSP1ERR	29	rwh	**FSP1 Error Pin Fault State Status Bit** The bit indicates that Error pin was set into fault state by the SMU_stdby. Reset by setting to 1. If the Error Pins were set in fault state by the SMU_stdby, reseting this bit sets the Error Pins back in fault free state 0_B Error pin was not set into fault state 1_B The Error pin was set into fault state.
FSP2ERR	30	rwh	**FSP2 Error Pin Fault State Status Bit** The bit indicates that Error pin was set into fault state by the SMU_stdby. Reset by setting to 1. If the Error Pins were set in fault state by the SMU_stdby, reseting this bit sets the Error Pins back in fault free state 0_B Error pin was not set into fault state 1_B The Error pin was set into fault state.
FSP3ERR	31	rwh	**FSP3 Error Pin Fault State Status Bit** The bit indicates that Error pin was set into fault state by the SMU_stdby. Reset by setting to 1. If the Error Pins were set in fault state by the SMU_stdby, reseting this bit sets the Error Pins back in fault free state 0_B Error pin was not set into fault state 1_B The Error pin was set into fault state.
0	27, 26, 25, 24, 23, 22, 21, 20	r	**Reserved** Read as 0; should be written with 0.

Table 2062 Access mode restrictions of AGSTDBYi_STS (i=0) sorted by descending priority

Mode name	Access mode		Description
CMDSTDBY.ASCE == 1	rwh	FSP0ERR, FSP1ERR, FSP2ERR, FSP3ERR, SFz (z=0-19), SFz (z=20-27)	
(default)	rh	FSP0ERR, FSP1ERR, FSP2ERR, FSP3ERR, SFz (z=0-19), SFz (z=20-27)	

Table 2063 Reset values of AGSTDBYi_STS (i=0)

Reset type	Reset value	Note
LVDPMS2 Reset	$0000\ 0000_H$	
Cold PORST	$0000\ 0000_H$	

42.4.2.10 SMU_STDBY Group i Alarm Status Register (i=1)

AGSTDBYi_STS (i=1) Offset address: 9830_H+i*14_H

SMU_STDBY Group i Alarm Status Register Reset values see: Table 2065

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	**0**	**0**	**0**	**SF27 0**	**SF26 0**	**SF25 0**	**SF24 0**	**SF23 0**	**SF22 0**	**SF21 0**	**SF20 0**	**SF19**	**SF18**	**SF17**	**SF16**
r	r	r	r	rwh r	rwh r	rwh r	rwh r	rwh r	rwh r	rwh r	rwh r	rwh	rwh	rwh	rwh
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SF15	**SF14**	**SF13**	**SF12**	**SF11**	**SF10**	**SF9**	**SF8**	**SF7**	**SF6**	**SF5**	**SF4**	**SF3**	**SF2**	**SF1**	**SF0**
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
SFz (z=0-19)	z	rwh	**Status flag for alarm z belonging to alarm group i.** 0_B Status flag z does not report a fault condition 1_B Status flag z reports a fault condition
SFz (z=20-27)	z	rwh	**Status flag for alarm z belonging to alarm group i.** 0_B Status flag z does not report a fault condition 1_B Status flag z reports a fault condition
0	27, 26, 25, 24, 23, 22, 21, 20, 28, 29, 30, 31	r	**Reserved** Read as 0; should be written with 0.

Table 2064 Access mode restrictions of AGSTDBYi_STS (i=1) sorted by descending priority

Mode name	Access mode		Description
CMDSTDBY.ASCE == 1	rwh	SFz (z=0-19), SFz (z=20-27)	
(default)	rh	SFz (z=0-19), SFz (z=20-27)	

Table 2065 Reset values of AGSTDBYi_STS (i=1)

Reset type	Reset value	Note
LVDPMS2 Reset	0000 0000$_H$	
Cold PORST	0000 0000$_H$	

42.4.2.11 SMU_STDBY Group i Alarm Status Register (i=2-3)

AGSTDBYi_STS (i=2-3) Offset address: 9830$_H$+i*14$_H$

SMU_STDBY Group i Alarm Status Register Reset values see: Table 2067

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	**0**	**0**	**0**	**SF27 0**	**SF26 0**	**SF25 0**	**SF24 0**	**SF23 0**	**SF22**	**SF21**	**SF20**	**SF19**	**SF18**	**SF17**	**SF16**
r	r	r	r	rwh r	rwh r	rwh r	rwh r	rwh r	rwh	rwh	rwh	rwh	rwh	rwh	rwh
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SF15	**SF14**	**SF13**	**SF12**	**SF11**	**SF10**	**SF9**	**SF8**	**SF7**	**SF6**	**SF5**	**SF4**	**SF3**	**SF2**	**SF1**	**SF0**
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
SFz (z=0-22)	z	rwh	**Status flag for alarm z belonging to alarm group i.** 0_B Status flag z does not report a fault condition 1_B Status flag z reports a fault condition
SFz (z=23-27)	z	rwh	**Status flag for alarm z belonging to alarm group i.** 0_B Status flag z does not report a fault condition 1_B Status flag z reports a fault condition
0	27, 26, 25, 24, 23, 28, 29, 30, 31	r	**Reserved** Read as 0; should be written with 0.

Table 2066 Access mode restrictions of AGSTDBYi_STS (i=2-3) sorted by descending priority

Mode name	Access mode		Description
CMDSTDBY.ASCE == 1	rwh	SFz (z=0-22), SFz (z=23-27)	
(default)	rh	SFz (z=0-22), SFz (z=23-27)	

Table 2067 Reset values of AGSTDBYi_STS (i=2-3)

Reset type	Reset value	Note
LVDPMS2 Reset	0000 0000$_H$	
Cold PORST	0000 0000$_H$	

42.4.2.12 SMU_STDBY Group i Alarm Status Register (i=4)

AGSTDBYi_STS (i=4) Offset address: 9830$_H$+i*14$_H$

SMU_STDBY Group i Alarm Status Register Reset values see: Table 2069

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	**0**	**0**	**0**	**SF27 0**	**SF26 0**	**SF25 0**	**SF24 0**	**SF23 0**	**SF22 0**	**SF21 0**	**SF20 0**	**SF19 0**	**SF18**	**SF17**	**SF16**
r	r	r	r	rwh r	rwh r	rwh r	rwh r	rwh r	rwh r	rwh r	rwh r	rwh r	rwh	rwh	rwh
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SF15	**SF14**	**SF13**	**SF12**	**SF11**	**SF10**	**SF9**	**SF8**	**SF7**	**SF6**	**SF5**	**SF4**	**SF3**	**SF2**	**SF1**	**SF0**
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
SFz (z=0-18)	z	rwh	**Status flag for alarm z belonging to alarm group i.** 0_B Status flag z does not report a fault condition 1_B Status flag z reports a fault condition
SFz (z=19-27)	z	rwh	**Status flag for alarm z belonging to alarm group i.** 0_B Status flag z does not report a fault condition 1_B Status flag z reports a fault condition
0	27, 26, 25, 24, 23, 22, 21, 20, 19, 28, 29, 30, 31	r	**Reserved** Read as 0; should be written with 0.

Table 2068 Access mode restrictions of AGSTDBYi_STS (i=4) sorted by descending priority

Mode name	Access mode		Description
CMDSTDBY.ASCE == 1	rwh	SFz (z=0-18), SFz (z=19-27)	
(default)	rh	SFz (z=0-18), SFz (z=19-27)	

Table 2069 Reset values of AGSTDBYi_STS (i=4)

Reset type	Reset value	Note
LVDPMS2 Reset	0000 0000$_H$	
Cold PORST	0000 0000$_H$	

42.4.2.13 SMU_STDBY PROT Register Endinit

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner and in some PROT states, the secure master to update the PROT state as described in the PROT mechanism.

The PROTSTDBYE register controls lock / unlock of the local Endinit (E) protected control registers.

PROTSTDBYE Offset address: $989C_H$

SMU_STDBY PROT Register Endinit Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OWEN	ODEF	TAGID						PRSEN	PRS			VMEN	VM		
w	rw	rw						rw	rw			rw	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												SWEN	STATE		
r												w	rwh		

Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
VM	18:16	rw	**Virtual Machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual Machine definition Enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection Set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.

(table continues...)

(continued)

Field	Bits	Type	Description
PRSEN	23	rw	**Protection Set definition Enable for PROT owner** 0_B **Disable**: PRS is not part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; should be written with 0.

Table 2070 Access mode restrictions of PROTSTDBYE sorted by descending priority

Mode name	Access mode		Description
write 1 to .**SWEN**	rwh	STATE	
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

42.4.2.14 SMU_STDBY PROT Register Safe Endinit

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner and in some PROT states, the secure master to update the PROT state as described in the PROT mechanism.

The PROTSTDBYSE register controls lock / unlock of the local Safe Endinit (SE) protected control registers.

PROTSTDBYSE Offset address: $98A0_H$

SMU_STDBY PROT Register Safe Endinit Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OWEN	ODEF	TAGID						PRSEN	PRS			VMEN	VM		
w	rw	rw						rw	rw			rw	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												SWEN	STATE		
r												w	rwh		

Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
VM	18:16	rw	**Virtual Machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual Machine definition Enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection Set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.
PRSEN	23	rw	**Protection Set definition Enable for PROT owner** 0_B **Disable**: PRS is not part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; should be written with 0.

Table 2071 Access mode restrictions of PROTSTDBYSE sorted by descending priority

Mode name	Access mode		Description
write 1 to .**SWEN**	rwh	STATE	
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

42.4.2.15 SMU_STDBY Write access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

ACCENSTDBY_WRA Offset address: $98A4_H$

SMU_STDBY Write access enable register A Application Reset value: $1000\ 0003_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	**EN30**	**EN29**	**EN28**	**EN27**	**EN26**	**EN25**	**EN24**	**EN23**	**EN22**	**EN21**	**EN20**	**EN19**	**EN18**	**EN17**	**EN16**
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	**EN14**	**EN13**	**EN12**	**EN11**	**EN10**	**EN09**	**EN08**	**EN07**	**EN06**	**EN05**	**EN04**	**EN03**	**EN02**	**EN01**	**EN00**
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q 0_B Disabled for write access 1_B Enabled for write access

42.4.2.16 SMU_STDBY Write access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

ACCENSTDBY_WRB Offset address: $98A8_H$

SMU_STDBY Write access enable register B Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								**EN39**	**EN38**	**EN37**	**EN36**	**EN35**	**EN34**	**EN33**	**EN32**
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q 0_B Disabled for write access 1_B Enabled for write access
0	31:8	r	**Reserved** Read as 0; should be written with 0.

42.4.2.17 SMU_STDBY Read access enable register A

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

ACCENSTDBY_RDA Offset address: $98AC_H$

SMU_STDBY Read access enable register A Application Reset value: $FFFF\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q 0_B Disabled for read access 1_B Enabled for read access

42.4.2.18 SMU_STDBY Read access enable register B

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

ACCENSTDBY_RDB Offset address: $98B0_H$

SMU_STDBY Read access enable register B Application Reset value: $0000\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q 0_B Disabled for read access 1_B Enabled for read access
0	31:8	r	**Reserved** Read as 0; should be written with 0.

42.4.2.19 SMU_STDBY VM access enable register

This register defines which virtual machine encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each VMx.

ACCENSTDBY_VM Offset address: $98B4_H$

SMU_STDBY VM access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for VM ID q** This bit enables read access to the access protected region for transactions with the VM ID q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for VM ID q** This bit enables write access to the access protected region for transactions with the VM ID q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

42.4.2.20 SMU_STDBY PRS access enable register

This register defines which protection register set encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each PRSx.

ACCENSTDBY_PRS Offset address: $98B8_H$

SMU_STDBY PRS access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for PRS q** This bit enables read access to the access protected region for transactions with the PRS q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for PRS q** This bit enables write access to the access protected region for transactions with the PRS q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

42.5 Debug information

The SMU supports the standard OTGB Interface. The trigger sets are described in the following section.

42.5.1 OCDS trigger bus (OTGB) interface

The SMU concentrates all failure indicator signals (alarms) of the device. By using them as MCDS trace information (failure type) and trace control (stop trace recording), the analysis of a failure's root cause is supported. This is even the case, when the alarm handling within the system includes a PORST, since the content of the trace memory will be still valid after the PORST.

The alarms handled in the SMU are very rare and sporadic. So it's acceptable that they are visible on the Trigger Bus with a small delay and not all in parallel with a cycle accurate timing resolution. The SMU Trigger Sets are shown on tables (Table 2072, Table 2073, Table 2074) below. It is output on OTGB0 or OTGB1 controlled by the OCS register.

Table 2072 TS16_SMU_SAFE0 trigger set SMU

Bits	Name	Description
0	AA	Any ALM bit active
[5:1]	AGI	ALM Group Index
[7:6]	ABI	ALM Byte Index, selecting the byte within the ALM group
[15:8]	ALM	Selected byte of the ALM group

Table 2073 TS16_SMU_SAFE1 trigger set SMU

Bits	Name	Description
0	AA	Any ALM bit active
[5:1]	AGI	ALM Group Index
[7:6]	ABI	ALM Byte Index, selecting the byte within the ALM group
[15:8]	ALM	Selected byte of the ALM group

Table 2074 TS16_SMU_CS trigger set SMU

Bits	Name	Description
0	AA	Any ALM bit active
[2:1]		Reserved
[5:3]	AGI	ALM Group Index
[7:6]	ABI	ALM Byte Index, selecting the byte within the ALM group
[15:8]	ALM	Selected byte of the ALM group

The table below shows the mapping of alarm groups and the value appeared in AGI fields for TS16_SMU_CS and TS16_SMU_SAFE0/1 Trigger Sets. The values shown in the table are all decimal values.

Table 2075 ALM group index (AGI) mapping

Alarm Group	TS16_SMU_CS AGI	TS16_SMU_SAFE0/1 AGI
ALMCS0 - ALMCS1	0 - 1	NA

(table continues...)

Table 2075 (continued) ALM group index (AGI) mapping

Alarm Group	TS16_SMU_CS AGI	TS16_SMU_SAFE0/1 AGI
ALMSF0 - ALMSF16	NA	0-16
ALMSH0 - ALMSH2	2-4	17-19

For any Trigger Set, If no alarm is active, TS16_SMU_SAFE0/SAFE1/CS will be all zero. If alarms are only active within one ALM Byte, TS16_SMU_SAFE0/SAFE1/CS will show statically this byte. If alarms are active in two or more ALM Bytes, TS16_SMU_SAFE0/SAFE1/CS will change between these bytes.

Note that due to the implementation, all the ALM Bytes are being scanned once per clock cycle, the TS16_SMU_SAFE0/SAFE1 ALM Byte information on OTGB0/1 can be delayed by up to 71 clock cycles for the second alarm. On the other hand TS16_SMU_CS ALM Byte information on OTGB0/1 can be delayed by up to 23 clock cycles for the second alarm. TS16_SMU_SAFE0/SAFE1/CS.AA will however become active (and inactive) immediately. The implementation will make sure, that the ALM Byte with the first causing alarm will be traced with MCDS before a reset as safety action clears this information.

Moreover depending upon the global CSAV level the behavior of TS16_SMU_CS is described below:

Table 2076 CSAV levels and TS16_SMU_CS behavior

CSAV	TS16_SMU_CS behavior
Full	Normal behavior as described above
Limited	No information about incoming alarm to SMU_CS is provided, always appear zeros on trigger set
No	No information about incoming alarm to SMU_CS is provided, always appear zeros on trigger set

***Note**: In some cases, it might be possible to clear an alarm faster than it is captured on the OTGB. Therefore software needs to take care that the time from an incoming alarm until it is cleared is longer than 68 f_{SPB} clock cycles for SMU_SAFE0/1 and 20 f_{SPB} clock cycles for SMU_CS to ensure that the alarm is captured on the OTGB.*

42.6 References

This section is not applicable for the module being described.

42.7 SMU revision history

Initial release of the chapter.

42.8 TC4Dx SMU information

42.8.1 TC4Dx SMU configuration

Table 2077 Device specific parameters

Parameter	Value
Number of SMU instances	1

42.8.2 TC4Dx SMU features

There are no deviations from the generic specification.

42.8.3 TC4Dx SMU functional description

There are no deviations from the generic specification.

42.8.4 TC4Dx SMU registers

This following chapters shows specific information related to the SMU registers in the TC4Dx device.

42.8.4.1 Registers of SMU

This chapter shows specific information related to the SMU registers excluding the SMU_STDBY registers. The SMU_STDBY registers information is present in the next section.

42.8.4.1.1 Register address space - SMU

Table 2078 Registers address space - SMU

Module	Base address	End address	Note
SMU	$F0028000_H$	$F0029FFF_H$	FPI slave interface

42.8.4.1.2 Register overview - access mode glossary

Table 2079 Register overview - access mode glossary

Keyword	Description
CE	Access protection using PROT register SMU_CS_PROTCE .
CSE	Access protection using PROT register SMU_CS_PROTCSE .
GE	Access protection using PROT register SMU_GCC_PROTGE .
GSE	Access protection using PROT register SMU_GCC_PROTGSE .
SxE (x=0-1)	Access protection using PROT register SMU_SAFEx_PROTSE .
SxSE (x=0-1)	Access protection using PROT register SMU_SAFEx_PROTSSE .
APU-PCS	Protection group consisting of registers SMU_CS_ACCEN_WRA , SMU_CS_ACCEN_WRB , SMU_CS_ACCEN_RDA , SMU_CS_ACCEN_RDB , SMU_CS_ACCEN_VM , SMU_CS_ACCEN_PRS .
PCS	Access protection using *APU-PCS* registers.
APU-PGCC	Protection group consisting of registers SMU_GCC_ACCEN_WRA , SMU_GCC_ACCEN_WRB , SMU_GCC_ACCEN_RDA , SMU_GCC_ACCEN_RDB , SMU_GCC_ACCEN_VM , SMU_GCC_ACCEN_PRS .
PGCC	Access protection using *APU-PGCC* registers.
APU-PSAFEx (x=0-1)	Protection group consisting of registers SMU_SAFEx_ACCEN_WRA , SMU_SAFEx_ACCEN_WRB , SMU_SAFEx_ACCEN_RDA , SMU_SAFEx_ACCEN_RDB , SMU_SAFEx_ACCEN_VM , SMU_SAFEx_ACCEN_PRS .
PSAFEx	Access protection using *APU-PSAFEx* registers.
BE	Always returns a Bus Error.
32	Access only when using 32-bit width.
U	No access restrictions.
PROT	Access restrictions as defined in the PROT register access rules.

42.8.4.1.3 Registers overview - SMU (ascending offset address)

Table 2080 Registers overview - SMU (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
SMU_OCS	OCDS Control and Status	0004_H	PGCC	SV, PGCC	Debug Reset	7114
SMU_ID	Module Identification Register	0008_H	PGCC	BE	Application Reset	7115
SMU_GCC_KEYS	SMU_GCC Key Register	$000C_H$	PGCC	SV, PGCC, GE, 32	Application Reset	7116
SMU_GCC_EMSR	Emergency Stop Register	0010_H	PGCC	SV, PGCC, GE	Application Reset	7116
SMU_GCC_EMSSW	Emergency Stop Software set and clear register	0014_H	PGCC	PGCC, SV	Application Reset	7118
SMU_GCC_RMi_CTL (i=0-1)	Register Monitor Control i	0018_H+i*12	PGCC	SV, PGCC, GE, 32	Application Reset	7119
SMU_GCC_RMi_EF (i=0-1)	Register Monitor Error Flags i	$001C_H$+i*12	PGCC	SV, PGCC, GE, 32	Application Reset	7119
SMU_GCC_RMi_STS (i=0-1)	Register Monitor Self Test Status i	0020_H+i*12	PGCC	SV, PGCC, GE, 32	Application Reset	7120
SMU_GCC_ALMTST	Alarm Signal Test Register	0058_H	PGCC	SV, PGCC, GE	Application Reset	7120
SMU_GCC_AGSHi_SEL (i=0-2)	SMU_GCC Shared Alarm Group i Selection Register	$005C_H$+i*8	PGCC	SV, PGCC, GE, 32	PowerOn Reset	7121
SMU_GCC_AGSHi_SS (i=0-2)	SMU_GCC Shared Alarm Group i Selection Status	0060_H+i*8	PGCC	BE	7122	7122
SMU_GCC_PROTGE	SMU_GCC PROT Register Endinit	0078_H	U	SV, PROT	Application Reset	7123
SMU_GCC_PROTGSE	SMU_GCC PROT Register Safe Endinit	$007C_H$	U	SV, PROT	Application Reset	7124
SMU_GCC_ACCEN_WRA	SMU_GCC Write access enable register A	0080_H	U	GSE, SV	Application Reset	7126
SMU_GCC_ACCEN_WRB	SMU_GCC Write access enable register B	0084_H	U	GSE, SV	Application Reset	7126
SMU_GCC_ACCEN_RDA	SMU_GCC Read access enable register A	0088_H	U	GSE, SV	Application Reset	7127
SMU_GCC_ACCEN_RDB	SMU_GCC Read access enable register B	$008C_H$	U	GSE, SV	Application Reset	7127
SMU_GCC_ACCEN_VM	SMU_GCC VM access enable register	0090_H	U	GSE, SV	Application Reset	7128

(table continues...)

Table 2080 (continued) Registers overview - SMU (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
SMU_GCC_ACCEN_PRS	SMU_GCC PRS access enable register	0094_H	U	GSE, SV	Application Reset	7129
SMU_CS_CMD	SMU_CS Command Register	0200_H	PCS	SV, PCS, 32	Application Reset	7129
SMU_CS_STS	SMU_CS Status Register	0204_H	PCS	SV, PCS, 32	Application Reset	7130
SMU_CS_RTC	SMU_CS Recovery Timer Configuration	0208_H	PCS	SV, PCS, CE, 32	Application Reset	7131
SMU_CS_KEYS	SMU_CS Key Register	$020C_H$	PCS	SV, PCS, CE, 32	Application Reset	7132
SMU_CS_DBG	SMU_CS Debug Register	0210_H	PCS	BE	PowerOn Reset	7133
SMU_CS_AFCNT	SMU_CS Alarm and Fault Counter	0214_H	PCS	BE	PowerOn Reset	7133
SMU_CS_RT0ACA	SMU_CS Recovery Timer 0 Alarm Configuration A	0218_H	PCS	SV, PCS, CE, 32	7134	7134
SMU_CS_RT0ACB	SMU_CS Recovery Timer 0 Alarm Configuration B	$021C_H$	PCS	SV, PCS, CE, 32	Application Reset	7135
SMU_CS_AEX	SMU_CS Alarm Executed Status Register	0220_H	PCS	BE	Application Reset	7136
SMU_CS_AEXCLR	SMU_CS Alarm Executed Status Clear Register	0224_H	PCS	SV, PCS, 32	Application Reset	7138
SMU_CS_RST	CS_RESET Request Triggering Alarm Information Register	0228_H	PCS	BE	PowerOn Reset	7140
SMU_CS_AGCSi_CONj (i=0;j=0-2) (i=1;j=0-2)	SMU_CS Group i Alarm Configuration Register j	$022C_H$+i*18_H+j*4	PCS	SV, PCS, CE, 32	7140	7140
SMU_CS_AGCSi_STS (i=0-1)	SMU_CS Group i Alarm Status Register	0238_H+i*18_H	PCS	SV, PCS, 32	Application Reset	7141
SMU_CS_AGCSi_DBG (i=0-1)	SMU_CS Group i Alarm Debug Register	$023C_H$+i*18_H	PCS	BE	PowerOn Reset	7142
SMU_CS_AGCSi_CAS (i=0-1)	SMU_CS Group i Critical Alarm Selection Flag	0240_H+i*18_H	PCS	SV, PCS, CE, 32	Application Reset	7142

(table continues...)

Table 2080 (continued) Registers overview - SMU (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
SMU_CS_AGSHi_CONj (i=0;j=0,2) (i=0,2;j=1) (i=1;j=0-2) (i=2;j=0,2)	SMU_CS Group i Shared Alarm Configuration Register j	0300_H+i* 20_H+j*4	PCS	SV, PCS, CE, 32	7143	7143
SMU_CS_AGSHi_STS (i=0-2)	SMU_CS Group i Shared Alarm Status Register	$030C_H$+i* 20_H	PCS	SV, PCS, 32	Application Reset	7144
SMU_CS_AGSHi_DBG (i=0-2)	SMU_CS Group i Shared Alarm Debug Register	0310_H+i* 20_H	PCS	BE	PowerOn Reset	7145
SMU_CS_AGSHi_CAS (i=0-2)	SMU_CS Group i Critical Shared Alarm Selection Flag	0314_H+i* 20_H	PCS	SV, PCS, CE, 32	Application Reset	7145
SMU_CS_AGSHi_SEL (i=0-2)	SMU_CS Group i Shared Alarm Group Selection Register	0318_H+i* 20_H	PCS	SV, CE, PCS, 32	7146	7146
SMU_CS_AGSHi_SS (i=0-2)	SMU_CS Group i Shared Alarm Group Selection Status	$031C_H$+i* 20_H	PCS	BE	PowerOn Reset	7147
SMU_CS_ALMTST	Alarm Signal Test Register	$03F8_H$	PCS	SV, PCS, CE	Application Reset	7148
SMU_CS_PROTCE	SMU_CS PROT Register Endinit	$03FC_H$	U	SV, PROT	Application Reset	7148
SMU_CS_PROTCSE	SMU_CS PROT Register Safe Endinit	0400_H	U	SV, PROT	Application Reset	7150
SMU_CS_ACCEN_WRA	SMU_CS Write access enable register A	0404_H	U	CSE, SV	Application Reset	7152
SMU_CS_ACCEN_WRB	SMU_CS Write access enable register B	0408_H	U	CSE, SV	Application Reset	7152
SMU_CS_ACCEN_RDA	SMU_CS Read access enable register A	$040C_H$	U	CSE, SV	Application Reset	7153
SMU_CS_ACCEN_RDB	SMU_CS Read access enable register B	0410_H	U	CSE, SV	Application Reset	7153
SMU_CS_ACCEN_VM	SMU_CS VM access enable register	0414_H	U	CSE, SV	Application Reset	7154
SMU_CS_ACCEN_PRS	SMU_CS PRS access enable register	0418_H	U	CSE, SV	Application Reset	7154
SMU_SAFEx_CMD (x=0-1)	SMU_SAFE x Command Register	1000_H+x *500_H	PSAFEx	SV, PSAFEx, 32	Application Reset	7155

(table continues...)

Table 2080 (continued) Registers overview - SMU (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
SMU_SAFEx_STS (x=0-1)	SMU_SAFE x Status Register	1004_H+x *500_H	PSAFEx	SV, PSAFEx, 32	Application Reset	7156
SMU_SAFEx_FSPi_CON (i=0-1;x=0-1)	SMU_SAFE x Fault Signaling Protocol Driver Timing Register i	1008_H+x *500_H+i*4	PSAFEx	SV, PSAFEx, SxE, 32	PowerOn Reset	7158
SMU_SAFEx_PES (x=0-1)	SMU_SAFE x Port Emergency Stop Enable Register	1010_H+x *500_H	PSAFEx	SV, PSAFEx, SxE, 32	Application Reset	7162
SMU_SAFEx_AGC (x=0-1)	SMU_SAFE x Alarm Global Configuration	1014_H+x *500_H	PSAFEx	SV, PSAFEx, SxE, 32	Application Reset	7163
SMU_SAFEx_RTC (x=0-1)	SMU_SAFE x Recovery Timer Configuration	1018_H+x *500_H	PSAFEx	SV, PSAFEx, SxE, 32	7165	7165
SMU_SAFEx_KEYS (x=0-1)	SMU_SAFE x Key Register	$101C_H$+x *500_H	PSAFEx	SV, PSAFEx, SxE, 32	Application Reset	7166
SMU_SAFEx_DBG (x=0-1)	SMU_SAFE x Debug Register	1020_H+x *500_H	PSAFEx	BE	PowerOn Reset	7167
SMU_SAFEx_PCTL (x=0-1)	SMU_SAFE x Port Control	1024_H+x *500_H	PSAFEx	SV, PSAFEx, SxE, 32	PowerOn Reset	7167
SMU_SAFEx_AFCNT (x=0-1)	SMU_SAFE x Alarm and Fault Counter	1028_H+x *500_H	PSAFEx	BE	PowerOn Reset	7169
SMU_SAFEx_RT0ACA (x=0-1)	SMU_SAFE x Recovery Timer 0 Alarm Configuration A	1030_H+x *500_H	PSAFEx	SV, PSAFEx, SxE, 32	7170	7170
SMU_SAFEx_RT0ACB (x=0-1)	SMU_SAFE x Recovery Timer 0 Alarm Configuration B	1034_H+x *500_H	PSAFEx	SV, PSAFEx, SxE, 32	7171	7171
SMU_SAFEx_RT1ACA (x=0-1)	SMU_SAFE x Recovery Timer 1 Alarm Configuration A	1038_H+x *500_H	PSAFEx	SV, PSAFEx, SxE, 32	7173	7173
SMU_SAFEx_RT1ACB (x=0-1)	SMU_SAFE x Recovery Timer 1 Alarm Configuration B	$103C_H$+x *500_H	PSAFEx	SV, PSAFEx, SxE, 32	7174	7174
SMU_SAFEx_AEX (x=0-1)	SMU_SAFE x Alarm Executed Status Register	1050_H+x *500_H	PSAFEx	BE	Application Reset	7175
SMU_SAFEx_AEXCLR (x=0-1)	SMU_SAFE x Alarm Executed Status Clear Register	1054_H+x *500_H	PSAFEx	SV, PSAFEx, 32	Application Reset	7180

(table continues...)

Table 2080 (continued) Registers overview - SMU (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
SMU_SAFEx_RST (x=0-1)	SMU_SAFE x RESET0/1 Request Triggering Alarm Information Register	1058_H+x *500_H	PSAFEx	BE	PowerOn Reset	7183
SMU_SAFEx_AGSFi_CONj (i=0;j=0,2;x=0) (i=0-1,8;j=1,3;x=0) (i=2-7,9,11-16;j=0-3;x=0) (i=10;j=0-3;x=0) (i=1;j=0,2;x=0) (i=8;j=0,2;x=0) (i=0-16;j=0-3;x=1)	SMU_SAFE x Group i Alarm Configuration Register j	1080_H+x *500_H+i* 24_H+j*4	PSAFEx	SV, PSAFEx, SxE, 32	7185	7185
SMU_SAFEx_AGSFi_FSPj (i=0-9,11-16;j=0-1;x=0) (i=10;j=0-1;x=0) (i=0-16;j=0-1;x=1)	SMU_SAFE x Group i FSP j Alarm Configuration Register	1090_H+x *500_H+i* 24_H+j*4	PSAFEx	SV, PSAFEx, SxE, 32	7187	7187
SMU_SAFEx_AGSFi_STS (i=0-16;x=0-1)	SMU_SAFE x Group i Alarm Status Register	1098_H+x *500_H+i* 24_H	PSAFEx	SV, PSAFEx, 32	Application Reset	7188
SMU_SAFEx_AGSFi_DBG (i=0-16;x=0-1)	SMU_SAFE x Group i Alarm Debug Register	$109C_H$+x *500_H+i* 24_H	PSAFEx	BE	PowerOn Reset	7189
SMU_SAFEx_AGSFi_CAS (i=0-16;x=0-1)	SMU_SAFE x Group i Critical Alarm Selection Flag	$10A0_H$+x *500_H+i* 24_H	PSAFEx	SV, PSAFEx, SxE, 32	Application Reset	7189
SMU_SAFEx_AGSHi_CONj (i=0-2;j=0-3;x=0-1)	SMU_SAFE x Group i Shared Alarm Configuration Register j	1300_H+x *500_H+i* 28_H+j*4	PSAFEx	SV, PSAFEx, SxE, 32	Application Reset	7190
SMU_SAFEx_AGSHi_FSPj (i=0-2;j=0-1;x=0-1)	SMU_SAFE x Group i FSP j Shared Alarm Configuration Register	1310_H+x *500_H+i* 28_H+j*4	PSAFEx	SV, PSAFEx, SxE, 32	Application Reset	7190
SMU_SAFEx_AGSHi_STS (i=0-2;x=0-1)	SMU_SAFE x Group i Shared Alarm Status Register	1318_H+x *500_H+i* 28_H	PSAFEx	SV, PSAFEx, 32	Application Reset	7191
SMU_SAFEx_AGSHi_DBG (i=0-2;x=0-1)	SMU_SAFE x Group i Shared Alarm Debug Register	$131C_H$+x *500_H+i* 28_H	PSAFEx	BE	PowerOn Reset	7192

(table continues...)

Table 2080 (continued) Registers overview - SMU (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
SMU_SAFEx_AGSHi_CAS (i=0-2;x=0-1)	SMU_SAFE x Group i Critical Shared Alarm Selection Flag	1320_H+x *500_H+i* 28_H	PSAFEx	SV, PSAFEx, SxE, 32	Application Reset	7192
SMU_SAFEx_AGSHi_SS (i=0-2;x=0-1)	SMU_SAFE x Group i Shared Alarm Group Selection Status	1324_H+x *500_H+i* 28_H	PSAFEx	BE	PowerOn Reset	7193
SMU_SAFEx_PROTSE (x=0-1)	SMU_SAFE x PROT Register Endinit	$13FC_H$+x *500_H	U	SV, PROT	Application Reset	7193
SMU_SAFEx_PROTSSE (x=0-1)	SMU_SAFE x PROT Register Safe Endinit	1400_H+x *500_H	U	SV, PROT	Application Reset	7195
SMU_SAFEx_ACCEN_WRA (x=0-1)	SMU_SAFE x Write access enable register A	1404_H+x *500_H	U	SxSE, SV	Application Reset	7197
SMU_SAFEx_ACCEN_WRB (x=0-1)	SMU_SAFE x Write access enable register B	1408_H+x *500_H	U	SxSE, SV	Application Reset	7197
SMU_SAFEx_ACCEN_RDA (x=0-1)	SMU_SAFE x Read access enable register A	$140C_H$+x *500_H	U	SxSE, SV	Application Reset	7198
SMU_SAFEx_ACCEN_RDB (x=0-1)	SMU_SAFE x Read access enable register B	1410_H+x *500_H	U	SxSE, SV	Application Reset	7198
SMU_SAFEx_ACCEN_VM (x=0-1)	SMU_SAFE x VM access enable register	1414_H+x *500_H	U	SxSE, SV	Application Reset	7199
SMU_SAFEx_ACCEN_PRS (x=0-1)	SMU_SAFE x PRS access enable register	1418_H+x *500_H	U	SxSE, SV	Application Reset	7199

42.8.4.1.4 Device specific registers

There are no device specific register changes.

42.8.4.2 Registers of SMU_STDBY

42.8.4.2.1 Register address space - SMU_STDBY

Table 2081 Registers address space - SMU_STDBY

Module	Base address	End address	Note
SMUSTDBY	$F0240000_H$	$F0249FFF_H$	FPI slave interface

42.8.4.2.2 Register overview - access mode glossary

Table 2082 Register overview - access mode glossary

Keyword	Description
E	Access protection using PROT register SMUSTDBY_PROTSTDBYE .
SE	Access protection using PROT register SMUSTDBY_PROTSTDBYSE .
APU-P	Protection group consisting of registers SMUSTDBY_ACCENSTDBY_WRA , SMUSTDBY_ACCENSTDBY_WRB , SMUSTDBY_ACCENSTDBY_RDA , SMUSTDBY_ACCENSTDBY_RDB , SMUSTDBY_ACCENSTDBY_VM , SMUSTDBY_ACCENSTDBY_PRS .
P	Access protection using *APU-P* registers.
BE	Always returns a Bus Error.
U	No access restrictions.
PROT	Access restrictions as defined in the PROT register access rules.

42.8.4.2.3 Registers overview - SMUSTDBY (ascending offset address)

Table 2083 Registers overview - SMUSTDBY (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
SMUSTDBY_MONBISTSTAT	SMU_STDBY BIST Status Register	9814_H	P	BE		
SMUSTDBY_MONBISTCTRL	SMU_STDBY BIST Control Register	9818_H	P	SV, P, E		
SMUSTDBY_CMDSTDBY	SMU_STDBY Command Register	$981C_H$	P	SV, P, E		
SMUSTDBY_AGSTDBYi_FSPj (i=0-1;j=0-3) (i=2-3;j=0-3) (i=4;j=0-3)	SMU_STDBY Group i FSP j Configuration Register	9820_H+i* 14_H+j*4	P	SV, P, E		

(table continues...)

Table 2083 (continued) Registers overview - SMUSTDBY (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
SMUSTDBY_AGSTDBYi_STS (i=0-1,4) (i=2-3)	SMU_STDBY Group i Alarm Status Register	9830_H+i*14_H	P	SV, P		
SMUSTDBY_PROTSTDBYE	SMU_STDBY PROT Register Endinit	$989C_H$	U	SV, PROT	Application Reset	
SMUSTDBY_PROTSTDBYSE	SMU_STDBY PROT Register Safe Endinit	$98A0_H$	U	SV, PROT	Application Reset	
SMUSTDBY_ACCENSTDBY_WRA	SMU_STDBY Write access enable register A	$98A4_H$	U	SE, SV	Application Reset	
SMUSTDBY_ACCENSTDBY_WRB	SMU_STDBY Write access enable register B	$98A8_H$	U	SE, SV	Application Reset	
SMUSTDBY_ACCENSTDBY_RDA	SMU_STDBY Read access enable register A	$98AC_H$	U	SE, SV	Application Reset	
SMUSTDBY_ACCENSTDBY_RDB	SMU_STDBY Read access enable register B	$98B0_H$	U	SE, SV	Application Reset	
SMUSTDBY_ACCENSTDBY_VM	SMU_STDBY VM access enable register	$98B4_H$	U	SE, SV	Application Reset	
SMUSTDBY_ACCENSTDBY_PRS	SMU_STDBY PRS access enable register	$98B8_H$	U	SE, SV	Application Reset	

42.8.4.2.4 Device specific registers

There are no device specific register changes.

42.8.5 TC4Dx SMU alarm mapping tables

The Alarm Mapping tables specify the mapping between the alarms implemented within the TC4Dx device and the assigned Alarm Index within SMU. Please refer to the Safety Manual for a detailed description of which safety mechanisms trigger which alarms.

There is a direct relationship between an Alarm Index and their corresponding configuration and status registers. The examples below show how the alarm index and group numbers of a particular alarm are related to the register names, as explained also in the Alarm Mapping chapter. Since the same naming principle applies for all alarms within an alarm group, only one example per alarm group is shown below.

For safety alarm groups ALMSF, the following registers and bit-fields can be used to configure the alarm reaction and getting the status related to alarm mapped to Alarm Index ALMSF1[10] :

- SAFEx_AGSFi_CONj.**CF10** (where x=0-1;**i = 1**;j=0-3)
- SAFEx_AGSFi_FSPj.**FE10** (where x=0-1;**i = 1**;j=0-1)
- SAFEx_AGSFi_STS.**SF10** (where x=0-1;**i = 1**)
- SAFEx_AGSFi_DBG.**DF10** (where x=0-1;**i = 1**)
- SAFEx_AGSFi_CAS.**CF10** (where x=0-1;**i = 1**)

For security alarm groups ALMCS, the following registers and bit-fields can be used to configure the alarm reaction and getting the status related to alarm mapped to Alarm Index ALMCS1[9] :

- CS_AGCSi_CONj.**CF9** (where **i = 1**;j=0-2)
- CS_AGCSi_STS.**SF9** (where **i = 1**)
- CS_AGCSi_DBG.**DF9** (where **i = 1**)
- CS_AGCSi_CAS.**CF9** (where **i = 1**)

For shared alarm groups ALMSH , the following registers and bit-fields can be used to configure the alarm reaction and getting the status related to alarm mapped to Alarm Index ALMSH1[10] :

- GCC_AGSHi_SEL.**CF10** (where **i = 1**)
- GCC_AGSHi_SS.**SF10** (where **i = 1**)
- CS_AGSHi_SEL.**CF10** (where **i = 1**)
- CS_AGSHi_SS.**SF10** (where **i = 1**)
- CS_AGSHi_CONj.**CF10** (where **i = 1**;j=0-2)
- CS_AGSHi_STS.**SF10** (where **i = 1**)
- CS_AGSHi_DBG.**DF10** (where **i = 1**)
- CS_AGSHi_CAS.**CF10** (where **i = 1**)
- SAFEx_AGSHi_CONj.**CF10** (where x=0-1;**i = 1**;j=0-3)
- SAFEx_AGSHi_FSPj.**FE10** (where x=0-1;**i = 1**;j=0-1)
- SAFEx_AGSHi_STS.**SF10** (where x=0-1;**i = 1**)
- SAFEx_AGSHi_DBG.**DF10** (where x=0-1;**i = 1**)
- SAFEx_AGSHi_CAS.**CF10** (where x=0-1;**i = 1**)

For safety alarm groups ALMSTDBY , the following registers and bit-fields can be used to configure the alarm reaction and getting the status related to alarm mapped to Alarm Index AGSTDBY1[8] :

- SMUSTDBY_AGSTDBYi_FSPj.**FE8** (where **i = 1**;j=0-3)
- SMUSTDBY_AGSTDBYi_STS.**SF8** (where **i = 1**)

For any Alarm Index having the corresponding Functional Block and Alarm Name columns marked as 'Reserved', it means that no alarm is mapped to that particular Alarm Index in the current device. As a consequence, the reading or writing to the corresponding bit-fields in the configuration and status registers as explained above will not have any impact.

42.8.5.1 Safety alarm mapping tables

Table 2084 Alarm mapping table for ALMSF0

Alarm index	Functional block instance	Alarm name
ALMSF0[0]	CPU0	CPU lockstep mismatch alarm CPU CFI lockstep mismatch alarm
ALMSF0[1]	CPU0	CPU access protection alarm
ALMSF0[2]	CPU0	CPU SOTA select redundancy error alarm
ALMSF0[3]	CPU0	CPU CFI access protection alarm
ALMSF0[4]	CPU0	CPU CFI wait state error
ALMSF0[5]	CPU0	CPU instruction fetch EDC error alarm
ALMSF0[6]	CPU0	CPU data side access EDC error alarm
ALMSF0[7]	NVM.PFRWB0A	PFLASH ECC double bit error
ALMSF0[8]	NVM.PFRWB0B	PFLASH ECC double bit error
ALMSF0[9]	NVM.FSI0	NVM incorrect bank monitoring
ALMSF0[10]	NVM.FSI0	NVM static configurations MISR error
ALMSF0[11]	NVM.DFRWB0	DFLASH static configurations MISR error
ALMSF0[12]	NVM.DMU	NVM SRI Address Phase EDC Error
ALMSF0[13]	NVM.DMU	NVM SRI Write Data Phase EDC Error
ALMSF0[14]	NVM.DMU	NVM incorrect bank monitoring
ALMSF0[15]	Reserved	Reserved
ALMSF0[16]	NVM.PFRWB0A	PFLASH dynamic configurations SFF error
ALMSF0[17]	NVM.PFRWB0B	PFLASH dynamic configurations SFF error
ALMSF0[20]	NVM.PFRWB0A	PFLASH ECC error
ALMSF0[21]	NVM.PFRWB0A	PFLASH EDC error
ALMSF0[22]	NVM.PFRWB0A	PFLASH ECC single bit error address buffer (SBAB) full
ALMSF0[23]	NVM.PFRWB0A	PFLASH ECC double bit error address buffer (DBAB) full
ALMSF0[24]	NVM.PFRWB0A	PFLASH ECC uncorrectable error address buffer (UBAB) full
ALMSF0[25]	NVM.PFRWB0A	PFLASH static configurations MISR error
ALMSF0[26]	NVM.PFRWB0B	PFLASH ECC error

(table continues...)

Table 2084 (continued) Alarm mapping table for ALMSF0

Alarm index	Functional block instance	Alarm name
ALMSF0[27]	NVM.PFRWB0B	PFLASH EDC error
ALMSF0[28]	NVM.PFRWB0B	PFLASH ECC single bit error address buffer (SBAB) full
ALMSF0[29]	NVM.PFRWB0B	PFLASH ECC double bit error address buffer (DBAB) full
ALMSF0[30]	NVM.PFRWB0B	PFLASH ECC uncorrectable error address buffer (UBAB) full
ALMSF0[31]	NVM.PFRWB0B	PFLASH static configurations MISR error

Table 2085 Alarm mapping table for ALMSF1

Alarm index	Functional block instance	Alarm name
ALMSF1[0]	CPU1	CPU lockstep mismatch alarm CPU CFI lockstep mismatch alarm
ALMSF1[1]	CPU1	CPU access protection alarm
ALMSF1[2]	CPU1	CPU SOTA select redundancy error alarm
ALMSF1[3]	CPU1	CPU CFI access protection alarm
ALMSF1[4]	CPU1	CPU CFI wait state error
ALMSF1[5]	CPU1	CPU instruction fetch EDC error alarm
ALMSF1[6]	CPU1	CPU data side access EDC error alarm
ALMSF1[7]	NVM.PFRWB1A	PFLASH ECC double bit error
ALMSF1[8]	NVM.PFRWB1B	PFLASH ECC double bit error
ALMSF1[9]	VMT0	VMT Miscellaneous Error
ALMSF1[11]	VMT0	VMT Uncorrectable Error
ALMSF1[12]	VMT1	VMT Miscellaneous Error
ALMSF1[14]	VMT1	VMT Uncorrectable Error
ALMSF1[15]	VMT0 VMT1	VMT SFF Uncorrectable Error VMT SFF Uncorrectable Error
ALMSF1[16]	NVM.PFRWB1A	PFLASH dynamic configurations SFF error
ALMSF1[17]	NVM.PFRWB1B	PFLASH dynamic configurations SFF error
ALMSF1[20]	NVM.PFRWB1A	PFLASH ECC error
ALMSF1[21]	NVM.PFRWB1A	PFLASH EDC error
ALMSF1[22]	NVM.PFRWB1A	PFLASH ECC single bit error address buffer (SBAB) full
ALMSF1[23]	NVM.PFRWB1A	PFLASH ECC double bit error address buffer (DBAB) full

(table continues...)

Table 2085 (continued) Alarm mapping table for ALMSF1

Alarm index	Functional block instance	Alarm name
ALMSF1[24]	NVM.PFRWB1A	PFLASH ECC uncorrectable error address buffer (UBAB) full
ALMSF1[25]	NVM.PFRWB1A	PFLASH static configurations MISR error
ALMSF1[26]	NVM.PFRWB1B	PFLASH ECC error
ALMSF1[27]	NVM.PFRWB1B	PFLASH EDC error
ALMSF1[28]	NVM.PFRWB1B	PFLASH ECC single bit error address buffer (SBAB) full
ALMSF1[29]	NVM.PFRWB1B	PFLASH ECC double bit error address buffer (DBAB) full
ALMSF1[30]	NVM.PFRWB1B	PFLASH ECC uncorrectable error address buffer (UBAB) full
ALMSF1[31]	NVM.PFRWB1B	PFLASH static configurations MISR error

Table 2086 Alarm mapping table for ALMSF2

Alarm index	Functional block instance	Alarm name
ALMSF2[0]	CPU2	CPU lockstep mismatch alarm CPU CFI lockstep mismatch alarm
ALMSF2[1]	CPU2	CPU access protection alarm
ALMSF2[2]	CPU2	CPU SOTA select redundancy error alarm
ALMSF2[3]	CPU2	CPU CFI access protection alarm
ALMSF2[4]	CPU2	CPU CFI wait state error
ALMSF2[5]	CPU2	CPU instruction fetch EDC error alarm
ALMSF2[6]	CPU2	CPU data side access EDC error alarm
ALMSF2[7]	NVM.PFRWB2A	PFLASH ECC double bit error
ALMSF2[8]	NVM.PFRWB2B	PFLASH ECC double bit error
ALMSF2[9]	VMT2	VMT Miscellaneous Error
ALMSF2[11]	VMT2	VMT Uncorrectable Error
ALMSF2[12]	VMT3	VMT Miscellaneous Error
ALMSF2[14]	VMT3	VMT Uncorrectable Error
ALMSF2[15]	VMT2	VMT SFF Uncorrectable Error
ALMSF2[16]	NVM.PFRWB2A	PFLASH dynamic configurations SFF error
ALMSF2[17]	NVM.PFRWB2B	PFLASH dynamic configurations SFF error
ALMSF2[20]	NVM.PFRWB2A	PFLASH ECC error
ALMSF2[21]	NVM.PFRWB2A	PFLASH EDC error
ALMSF2[22]	NVM.PFRWB2A	PFLASH ECC single bit error address buffer (SBAB) full

(table continues...)

Table 2086 (continued) Alarm mapping table for ALMSF2

Alarm index	Functional block instance	Alarm name
ALMSF2[23]	NVM.PFRWB2A	PFLASH ECC double bit error address buffer (DBAB) full
ALMSF2[24]	NVM.PFRWB2A	PFLASH ECC uncorrectable error address buffer (UBAB) full
ALMSF2[25]	NVM.PFRWB2A	PFLASH static configurations MISR error
ALMSF2[26]	NVM.PFRWB2B	PFLASH ECC error
ALMSF2[27]	NVM.PFRWB2B	PFLASH EDC error
ALMSF2[28]	NVM.PFRWB2B	PFLASH ECC single bit error address buffer (SBAB) full
ALMSF2[29]	NVM.PFRWB2B	PFLASH ECC double bit error address buffer (DBAB) full
ALMSF2[30]	NVM.PFRWB2B	PFLASH ECC uncorrectable error address buffer (UBAB) full
ALMSF2[31]	NVM.PFRWB2B	PFLASH static configurations MISR error

Table 2087 Alarm mapping table for ALMSF3

Alarm index	Functional block instance	Alarm name
ALMSF3[0]	CPU3	CPU lockstep mismatch alarm CPU CFI lockstep mismatch alarm
ALMSF3[1]	CPU3	CPU access protection alarm
ALMSF3[2]	CPU3	CPU SOTA select redundancy error alarm
ALMSF3[3]	CPU3	CPU CFI access protection alarm
ALMSF3[4]	CPU3	CPU CFI wait state error
ALMSF3[5]	CPU3	CPU instruction fetch EDC error alarm
ALMSF3[6]	CPU3	CPU data side access EDC error alarm
ALMSF3[7]	NVM.PFRWB3A	PFLASH ECC double bit error
ALMSF3[8]	NVM.PFRWB3B	PFLASH ECC double bit error
ALMSF3[12]	VMT5	VMT Miscellaneous Error
ALMSF3[14]	VMT5	VMT Uncorrectable Error
ALMSF3[16]	NVM.PFRWB3A	PFLASH dynamic configurations SFF error
ALMSF3[17]	NVM.PFRWB3B	PFLASH dynamic configurations SFF error
ALMSF3[20]	NVM.PFRWB3A	PFLASH ECC error
ALMSF3[21]	NVM.PFRWB3A	PFLASH EDC error
ALMSF3[22]	NVM.PFRWB3A	PFLASH ECC single bit error address buffer (SBAB) full

(table continues...)

Table 2087 (continued) Alarm mapping table for ALMSF3

Alarm index	Functional block instance	Alarm name
ALMSF3[23]	NVM.PFRWB3A	PFLASH ECC double bit error address buffer (DBAB) full
ALMSF3[24]	NVM.PFRWB3A	PFLASH ECC uncorrectable error address buffer (UBAB) full
ALMSF3[25]	NVM.PFRWB3A	PFLASH static configurations MISR error
ALMSF3[26]	NVM.PFRWB3B	PFLASH ECC error
ALMSF3[27]	NVM.PFRWB3B	PFLASH EDC error
ALMSF3[28]	NVM.PFRWB3B	PFLASH ECC single bit error address buffer (SBAB) full
ALMSF3[29]	NVM.PFRWB3B	PFLASH ECC double bit error address buffer (DBAB) full
ALMSF3[30]	NVM.PFRWB3B	PFLASH ECC uncorrectable error address buffer (UBAB) full
ALMSF3[31]	NVM.PFRWB3B	PFLASH static configurations MISR error

Table 2088 Alarm mapping table for ALMSF4

Alarm index	Functional block instance	Alarm name
ALMSF4[0]	CPU4	CPU lockstep mismatch alarm CPU CFI lockstep mismatch alarm
ALMSF4[1]	CPU4	CPU access protection alarm
ALMSF4[2]	CPU4	CPU SOTA select redundancy error alarm
ALMSF4[3]	CPU4	CPU CFI access protection alarm
ALMSF4[4]	CPU4	CPU CFI wait state error
ALMSF4[5]	CPU4	CPU instruction fetch EDC error alarm
ALMSF4[6]	CPU4	CPU data side access EDC error alarm
ALMSF4[7]	NVM.PFRWB4A	PFLASH ECC double bit error
ALMSF4[8]	NVM.PFRWB4B	PFLASH ECC double bit error
ALMSF4[10]	Reserved	Reserved
ALMSF4[12:14]	Reserved	Reserved
ALMSF4[16]	NVM.PFRWB4A	PFLASH dynamic configurations SFF error
ALMSF4[17]	NVM.PFRWB4B	PFLASH dynamic configurations SFF error
ALMSF4[20]	NVM.PFRWB4A	PFLASH ECC error
ALMSF4[21]	NVM.PFRWB4A	PFLASH EDC error
ALMSF4[22]	NVM.PFRWB4A	PFLASH ECC single bit error address buffer (SBAB) full

(table continues...)

Table 2088 (continued) Alarm mapping table for ALMSF4

Alarm index	Functional block instance	Alarm name
ALMSF4[23]	NVM.PFRWB4A	PFLASH ECC double bit error address buffer (DBAB) full
ALMSF4[24]	NVM.PFRWB4A	PFLASH ECC uncorrectable error address buffer (UBAB) full
ALMSF4[25]	NVM.PFRWB4A	PFLASH static configurations MISR error
ALMSF4[26]	NVM.PFRWB4B	PFLASH ECC error
ALMSF4[27]	NVM.PFRWB4B	PFLASH EDC error
ALMSF4[28]	NVM.PFRWB4B	PFLASH ECC single bit error address buffer (SBAB) full
ALMSF4[29]	NVM.PFRWB4B	PFLASH ECC double bit error address buffer (DBAB) full
ALMSF4[30]	NVM.PFRWB4B	PFLASH ECC uncorrectable error address buffer (UBAB) full
ALMSF4[31]	NVM.PFRWB4B	PFLASH static configurations MISR error

Table 2089 Alarm mapping table for ALMSF5

Alarm index	Functional block instance	Alarm name
ALMSF5[0]	CPU5	CPU lockstep mismatch alarm CPU CFI lockstep mismatch alarm
ALMSF5[1]	CPU5	CPU access protection alarm
ALMSF5[2]	CPU5	CPU SOTA select redundancy error alarm
ALMSF5[3]	CPU5	CPU CFI access protection alarm
ALMSF5[4]	CPU5	CPU CFI wait state error
ALMSF5[5]	CPU5	CPU instruction fetch EDC error alarm
ALMSF5[6]	CPU5	CPU data side access EDC error alarm
ALMSF5[7]	NVM.PFRWB5A	PFLASH ECC double bit error
ALMSF5[8]	NVM.PFRWB5B	PFLASH ECC double bit error
ALMSF5[9]	Reserved	Reserved
ALMSF5[10]	CSS	SAFETY_DOMAIN_ALARM0
ALMSF5[11]	CSS	SAFETY_DOMAIN_ALARM1
ALMSF5[12]	CSS	SAFETY_DOMAIN_ALARM2
ALMSF5[13]	CSS	SAFETY_DOMAIN_ALARM3
ALMSF5[14]	CSS	SAFETY_DOMAIN_ALARM4
ALMSF5[15]	CSS	SAFETY_DOMAIN_ALARM5
ALMSF5[16]	NVM.PFRWB5A	PFLASH dynamic configurations SFF error

(table continues...)

Table 2089 (continued) Alarm mapping table for ALMSF5

Alarm index	Functional block instance	Alarm name
ALMSF5[17]	NVM.PFRWB5B	PFLASH dynamic configurations SFF error
ALMSF5[20]	NVM.PFRWB5A	PFLASH ECC error
ALMSF5[21]	NVM.PFRWB5A	PFLASH EDC error
ALMSF5[22]	NVM.PFRWB5A	PFLASH ECC single bit error address buffer (SBAB) full
ALMSF5[23]	NVM.PFRWB5A	PFLASH ECC double bit error address buffer (DBAB) full
ALMSF5[24]	NVM.PFRWB5A	PFLASH ECC uncorrectable error address buffer (UBAB) full
ALMSF5[25]	NVM.PFRWB5A	PFLASH static configurations MISR error
ALMSF5[26]	NVM.PFRWB5B	PFLASH ECC error
ALMSF5[27]	NVM.PFRWB5B	PFLASH EDC error
ALMSF5[28]	NVM.PFRWB5B	PFLASH ECC single bit error address buffer (SBAB) full
ALMSF5[29]	NVM.PFRWB5B	PFLASH ECC double bit error address buffer (DBAB) full
ALMSF5[30]	NVM.PFRWB5B	PFLASH ECC uncorrectable error address buffer (UBAB) full
ALMSF5[31]	NVM.PFRWB5B	PFLASH static configurations MISR error

Table 2090 Alarm mapping table for ALMSF6

Alarm index	Functional block instance	Alarm name
ALMSF6[0:19]	Reserved	Reserved
ALMSF6[20]	HSSL0	HSSL SRI Read Data EDC Error
ALMSF6[21]	HSSL1	HSSL SRI Read Data EDC Error
ALMSF6[23]	XSPI	XSPI data address error alarm
ALMSF6[24]	Reserved	Reserved
ALMSF6[25]	HSPHY	HSPHY SFF uncorrectable error alarm
ALMSF6[27:28]	Reserved	Reserved
ALMSF6[29]	PCIE0	PCIE configuration uncorrectable error alarm
ALMSF6[30]	Reserved	Reserved
ALMSF6[31]	PCIE0	PCIE data uncorrectable error alarm

Table 2091 Alarm mapping table for ALMSF7

Alarm index	Functional block instance	Alarm name
ALMSF7[0]	LMU0 LMU4	LMU internal data transfer EDC error alarm LMU internal data transfer EDC error alarm
ALMSF7[1]	LMU0 LMU4	APU Alarm APU Alarm
ALMSF7[2]	Reserved	Reserved
ALMSF7[3]	LMU0 LMU4	LMU SRI Address Phase EDC Error LMU SRI Address Phase EDC Error
ALMSF7[4]	LMU0 LMU4	LMU SRI Write Data EDC Error LMU SRI Write Data EDC Error
ALMSF7[5]	LMU0 LMU4	LMU control redundancy mismatch LMU control redundancy mismatch
ALMSF7[6]	LMU0 LMU4	LMU control redundancy self-test mismatch LMU control redundancy self-test mismatch
ALMSF7[7]	LMU1 LMU5	LMU internal data transfer EDC error alarm LMU internal data transfer EDC error alarm
ALMSF7[8]	LMU1 LMU5	APU Alarm APU Alarm
ALMSF7[9]	Reserved	Reserved
ALMSF7[10]	LMU1 LMU5	LMU SRI Address Phase EDC Error LMU SRI Address Phase EDC Error
ALMSF7[11]	LMU1 LMU5	LMU SRI Write Data EDC Error LMU SRI Write Data EDC Error
ALMSF7[12]	LMU1 LMU5	LMU control redundancy mismatch LMU control redundancy mismatch
ALMSF7[13]	LMU1 LMU5	LMU control redundancy self-test mismatch LMU control redundancy self-test mismatch
ALMSF7[14]	LMU2 LMU6	LMU internal data transfer EDC error alarm LMU internal data transfer EDC error alarm
ALMSF7[15]	LMU2 LMU6	APU Alarm APU Alarm
ALMSF7[16]	Reserved	Reserved
ALMSF7[17]	LMU2 LMU6	LMU SRI Address Phase EDC Error LMU SRI Address Phase EDC Error
ALMSF7[18]	LMU2 LMU6	LMU SRI Write Data EDC Error LMU SRI Write Data EDC Error

(table continues...)

Table 2091 (continued) Alarm mapping table for ALMSF7

Alarm index	Functional block instance	Alarm name
ALMSF7[19]	LMU2 LMU6	LMU control redundancy mismatch LMU control redundancy mismatch
ALMSF7[20]	LMU2 LMU6	LMU control redundancy self-test mismatch LMU control redundancy self-test mismatch
ALMSF7[21]	LMU3 LMU7	LMU internal data transfer EDC error alarm LMU internal data transfer EDC error alarm
ALMSF7[22]	LMU3 LMU7	APU Alarm APU Alarm
ALMSF7[23]	Reserved	Reserved
ALMSF7[24]	LMU3 LMU7	LMU SRI Address Phase EDC Error LMU SRI Address Phase EDC Error
ALMSF7[25]	LMU3 LMU7	LMU SRI Write Data EDC Error LMU SRI Write Data EDC Error
ALMSF7[26]	LMU3 LMU7	LMU control redundancy mismatch LMU control redundancy mismatch
ALMSF7[27]	LMU3 LMU7	LMU control redundancy self-test mismatch LMU control redundancy self-test mismatch
ALMSF7[28]	IR	IR lockstep mismatch alarm
ALMSF7[29]	IR	IR EDC error alarm (Alarm_NCS)
ALMSF7[31]	DEBUG	OEN safety FF test fail

Table 2092 Alarm mapping table for ALMSF8

Alarm index	Functional block instance	Alarm name
ALMSF8[0]	WTU	WTU CPU0 WDT overflow or access error alarm
ALMSF8[1]	WTU	WTU CPU1 WDT overflow or access error alarm
ALMSF8[2]	WTU	WTU CPU2 WDT overflow or access error alarm
ALMSF8[3]	WTU	WTU CPU3 WDT overflow or access error alarm
ALMSF8[4]	WTU	WTU CPU4 WDT overflow or access error alarm
ALMSF8[5]	WTU	WTU CPU5 WDT overflow or access error alarm
ALMSF8[6]	WTU	WTU System WDT overflow or access error alarm
ALMSF8[7]	WTU	WTU combined overflow or access error alarm
ALMSF8[8]	WTU	WTU SFF uncorrectable error alarm
ALMSF8[9]	LETH0	LETH Data Address Error Alarm
ALMSF8[10]	Reserved	Reserved

(table continues...)

Table 2092 (continued) Alarm mapping table for ALMSF8

Alarm index	Functional block instance	Alarm name
ALMSF8[15]	CLOCKING	CCU safety flip-flop uncorrectable error
ALMSF8[16]	CLOCKING	System-PLL crystal clock frequency out of range
ALMSF8[17]	CLOCKING	Peripheral-PLL crystal clock frequency out of range
ALMSF8[18]	CLOCKING	EVR oscillator clock frequency out of range
ALMSF8[19]	CLOCKING	EVR oscillator clock alive error
ALMSF8[20]	CLOCKING	Ramp oscillator clock alive error
ALMSF8[21]	CLOCKING	Peripheral-PLL clock alive error
ALMSF8[22]	CLOCKING	System-PLL loss of lock event
ALMSF8[23]	CLOCKING	Peripheral-PLL loss of lock event
ALMSF8[24]	CLOCKING	Peripheral-PLL K-pre-divider K-divider error
ALMSF8[26]	CLOCKING	Peripheral clock divider error
ALMSF8[27]	CLOCKING	Ramp oscillator loss of lock event
ALMSF8[28]	CLOCKING	SYSPLL safety flip-flop uncorrectable error
ALMSF8[29]	CLOCKING	PERPLL safety flip-flop uncorrectable error
ALMSF8[30]	DRE	DRE SRI Address Phase EDC Error Alarm
ALMSF8[31]	DRE	DRE SRI Data Phase EDC Error Alarm

Table 2093 Alarm mapping table for ALMSF9

Alarm index	Functional block instance	Alarm name
ALMSF9[1]	VTMON	VTMON.DTS0 over temperature alarm
ALMSF9[3]	VTMON	VTMON.DTS1 over temperature alarm
ALMSF9[5]	VTMON	VTMON.DTS2 over temperature alarm
ALMSF9[7]	VTMON	VTMON.DTS3 over temperature alarm
ALMSF9[9]	VTMON	VTMON.DTS4 over temperature alarm
ALMSF9[11]	VTMON	VTMON SFF Uncorrectable Error alarm
ALMSF9[12]	PMS.DTS	PMS.DTS over temperature alarm
ALMSF9[16:19]	Reserved	Reserved
ALMSF9[28]	SCU	SCU SFF uncorrectable error alarm
ALMSF9[29]	SMM	SMM_REG_ALARM
ALMSF9[30]	PSI5	PSI5 safe protocol error alarm
ALMSF9[31]	Reserved	Reserved

Table 2094 Alarm mapping table for ALMSF10

Alarm index	Functional block instance	Alarm name
ALMSF10[0]	SMU	SMU_SAFE0 Software Alarm 0 or SMU_SAFE1 Software Alarm 0
ALMSF10[1]	SMU	SMU_SAFE0 Software Alarm 1 or SMU_SAFE1 Software Alarm 1
ALMSF10[2]	SMU	SMU_SAFE0 Software Alarm 2 or SMU_SAFE1 Software Alarm 2
ALMSF10[3]	SMU	SMU_SAFE0 Software Alarm 3 or SMU_SAFE1 Software Alarm 3
ALMSF10[4]	SMU	SMU_SAFE0 Software Alarm 4 or SMU_SAFE1 Software Alarm 4
ALMSF10[5]	SMU	SMU_SAFE0 Software Alarm 5 or SMU_SAFE1 Software Alarm 5
ALMSF10[6]	SMU	SMU_SAFE0 Software Alarm 6 or SMU_SAFE1 Software Alarm 6
ALMSF10[7]	SMU	SMU_SAFE0 Software Alarm 7 or SMU_SAFE1 Software Alarm 7
ALMSF10[8]	SMU	SMU_SAFE0 Software Alarm 8 or SMU_SAFE1 Software Alarm 8
ALMSF10[9]	SMU	SMU_SAFE0 Software Alarm 9 or SMU_SAFE1 Software Alarm 9
ALMSF10[10]	SMU	SMU_SAFE0 Software Alarm 10 or SMU_SAFE1 Software Alarm 10
ALMSF10[11]	SMU	SMU_SAFE0 Software Alarm 11 or SMU_SAFE1 Software Alarm 11
ALMSF10[12]	SMU	SMU_SAFE0 Software Alarm 12 or SMU_SAFE1 Software Alarm 12
ALMSF10[13]	SMU	SMU_SAFE0 Software Alarm 13 or SMU_SAFE1 Software Alarm 13
ALMSF10[14]	SMU	SMU_SAFE0 Software Alarm 14 or SMU_SAFE1 Software Alarm 14
ALMSF10[15]	SMU	SMU_SAFE0 Software Alarm 15 or SMU_SAFE1 Software Alarm 15
ALMSF10[16]	SMU	SMU_SAFE0 FSP0 Fault State Activation Alarm or SMU_SAFE1 FSP2 Fault State Activation Alarm

(table continues...)

Table 2094 (continued) Alarm mapping table for ALMSF10

Alarm index	Functional block instance	Alarm name
ALMSF10[17]	SMU	SMU_SAFE0 FSP1 Fault State Activation Alarm or SMU_SAFE1 FSP3 Fault State Activation Alarm
ALMSF10[18]	SMU	SMU_SAFE0 RT0 Time-out Alarm or SMU_SAFE1 RT0 Time-out Alarm
ALMSF10[19]	SMU	SMU_SAFE0 RT1 Time-out Alarm or SMU_SAFE1 RT1 Time-out Alarm
ALMSF10[20]	SMU_SAFE0 SMU_SAFE1 SMU_CS	SMU_SAFE0/1 SMU_CS SFF Uncorrectable Error Alarm
ALMSF10[21]	SMU_STDBY	SMU_STANDBY SFF Uncorrectable Error Alarm
ALMSF10[29]	PMS.VMONS0	VDDEVRSB OVER Voltage Alarm
	PMS.VMONS0	VDDEXT OVER Voltage Alarm
	PMS.VMONS0	VDDM OVER Voltage Alarm
	PMS.VMONS0	VDD OVER Voltage Alarm
	PMS.VMONS0	VDDPMS0 OVER Voltage Alarm
	PMS.VMONS0	VDDPMS1 OVER Voltage Alarm
	PMS.VMONS0	VDDPMS2 OVER Voltage Alarm
	PMS.VMONS0	VDDSBRAM OVER Voltage Alarm
	PMS.VMONS0	VDDPAD OVER Voltage Alarm
	PMS.VMONS1	VDDPHPHY0 OVER Voltage Alarm
	PMS.VMONS1	VDDPHPHY1 OVER Voltage Alarm
	PMS.VMONS1	VDDPHPHY2 OVER Voltage Alarm
	PMS.VMONS1	VDDHSIF OVER Voltage Alarm
	PMS.VMONS1	VDDPHY0 OVER Voltage Alarm
	PMS.VMONS1	VDDPHY1 OVER Voltage Alarm
	PMS.VMONS1	VDDPHY2 OVER Voltage Alarm
	PMS.VMONS1	VDDCPULMU0 OVER Voltage Alarm
	PMS.VMONS1	VDDCPULMU1 OVER Voltage Alarm
	PMS.VMONS1	VDDPPU OVER Voltage Alarm
	PMS.VMONP3	VDDEXTDC OVER Voltage Alarm

(table continues...)

Table 2094 (continued) Alarm mapping table for ALMSF10

Alarm index	Functional block instance	Alarm name
ALMSF10[30]	PMS.VMONS0	VDDEVRSB Under Voltage Alarm
	PMS.VMONS0	VDDEXT Under Voltage Alarm
	PMS.VMONS0	VDDM Under Voltage Alarm
	PMS.VMONS0	VDD Under Voltage Alarm
	PMS.VMONS0	VDDPMS0 Under Voltage Alarm
	PMS.VMONS0	VDDPMS1 Under Voltage Alarm
	PMS.VMONS0	VDDPMS2 Under Voltage Alarm
	PMS.VMONS0	VDDSBRAM Under Voltage Alarm
	PMS.VMONS0	VDDPAD Under Voltage Alarm
	PMS.VMONS1	VDDPHPHY0 Under Voltage Alarm
	PMS.VMONS1	VDDPHPHY1 Under Voltage Alarm
	PMS.VMONS1	VDDPHPHY2 Under Voltage Alarm
	PMS.VMONS1	VDDHSIF Under Voltage Alarm
	PMS.VMONS1	VDDPHY0 Under Voltage Alarm
	PMS.VMONS1	VDDPHY1 Under Voltage Alarm
	PMS.VMONS1	VDDPHY2 Under Voltage Alarm
	PMS.VMONS1	VDDCPULMU0 Under Voltage Alarm
	PMS.VMONS1	VDDCPULMU1 Under Voltage Alarm
	PMS.VMONS1	VDDPPU Under Voltage Alarm
	PMS.VMONP3	VDDEXTDC Under Voltage Alarm

Table 2095 Alarm mapping table for ALMSF11

Alarm index	Functional block instance	Alarm name
ALMSF11[0]	SRI0	SRI XBAR SOTA Swap Configuration Alarm
ALMSF11[1]	SRI0	SRI XBAR Address Phase EDC Error Alarm
ALMSF11[2]	SRI0	SRI XBAR Write Data EDC Error Alarm
ALMSF11[3]	SRI0	SRI XBAR Bus Error Event Alarm
ALMSF11[4]	SRI1	SRI XBAR SOTA Swap Configuration Alarm
ALMSF11[5]	SRI1	SRI XBAR Address Phase EDC Error Alarm
ALMSF11[6]	SRI1	SRI XBAR Write Data EDC Error Alarm
ALMSF11[7]	SRI1	SRI XBAR Bus Error Event Alarm
ALMSF11[8]	SRI2	SRI XBAR SOTA Swap Configuration Alarm
ALMSF11[9]	SRI2	SRI XBAR Address Phase EDC Error Alarm
ALMSF11[10]	SRI2	SRI XBAR Write Data EDC Error Alarm
ALMSF11[11]	SRI2	SRI XBAR Bus Error Event Alarm

(table continues...)

Table 2095 (continued) Alarm mapping table for ALMSF11

Alarm index	Functional block instance	Alarm name
ALMSF11[12]	TC4Dx: S2F_3COMPB_0	SRI S2F Address Phase EDC Error Alarm
ALMSF11[13]	TC4Dx: S2F_3COMPB_0	SRI S2F Write Data EDC Error Alarm
ALMSF11[14]	A2S PPU.STU	SRI A2S Address Phase ECC Error Alarm
ALMSF11[15]	A2S PPU.CBU	SRI A2S Address Phase ECC Error Alarm
ALMSF11[16]	TC4Dx: F2S_TPB7_0	SRI F2S Read Data EDC Error Alarm
ALMSF11[17]	S2A PPU.CSM	SRI S2A Address Phase EDC Error Alarm
ALMSF11[18]	S2A PPU.CSM	SRI S2A Write Data EDC Error Alarm
ALMSF11[19]	A2S PPU.STU	SRI A2S Read Data EDC Error Alarm
ALMSF11[20]	S2L	SRI S2L Address Phase EDC Error Alarm
ALMSF11[21]	S2L	SRI S2L Write Data EDC Error Alarm
ALMSF11[22]	Reserved	Reserved
ALMSF11[23]	A2S PPU.CBU	SRI A2S Read Data EDC Error Alarm
ALMSF11[24]	SRI4	SRI XBAR SOTA Swap Configuration Alarm
ALMSF11[25]	SRI4	SRI XBAR Address Phase EDC Error Alarm
ALMSF11[26]	SRI4	SRI XBAR Write Data EDC Error Alarm
ALMSF11[27]	SRI4	SRI XBAR Bus Error Event Alarm
ALMSF11[28]	S2F_4CSPB_0	SRI S2F Address Phase EDC Error Alarm
ALMSF11[29]	S2F_4CSPB_0	SRI S2F Write Data EDC Error Alarm
ALMSF11[30]	Reserved	Reserved
ALMSF11[31]	FPI.COMBCU	FPI error alarm (Alarm_NCS) FPI EDC error alarm (Alarm_NCS)

Table 2096 Alarm mapping table for ALMSF12

Alarm index	Functional block instance	Alarm name
ALMSF12[0:6]	Reserved	Reserved
ALMSF12[7]	PSI5	PSI5 SFF uncorrectable error alarm
ALMSF12[8:12]	Reserved	Reserved
ALMSF12[13]	FPI.SBCU	FPI error alarm (Alarm_NCS) FPI EDC error alarm (Alarm_NCS)
ALMSF12[14]	Reserved	Reserved
ALMSF12[15]	FPI.TBCU	FPI error alarm (Alarm_NCS) FPI EDC error alarm (Alarm_NCS)
ALMSF12[17:19]	Reserved	Reserved

(table continues...)

Table 2096 (continued) Alarm mapping table for ALMSF12

Alarm index	Functional block instance	Alarm name
ALMSF12[20]	PORT32 PORT33	PORT SFF Uncorrectable Error Alarm PORT SFF Uncorrectable Error Alarm
ALMSF12[21:23]	Reserved	Reserved
ALMSF12[24]	ADC	ADC_SMU_ALARM 0
ALMSF12[25]	ADC	ADC_SMU_ALARM 1
ALMSF12[26]	ADC	ADC_SMU_ALARM 2
ALMSF12[27]	ADC	ADC_SMU_ALARM 3
ALMSF12[28]	ADC	ADC_SMU_ALARM 4
ALMSF12[29]	ADC	ADC_SMU_ALARM 5
ALMSF12[30]	ADC	ADC_SMU_ALARM 6
ALMSF12[31]	ADC	ADC_SMU_ALARM 7

Table 2097 Alarm mapping table for ALMSF13

Alarm index	Functional block instance	Alarm name
ALMSF13[0:7]	Reserved	Reserved
ALMSF13[8]	PMS	PMS SFF Uncorrectable Error Alarm
ALMSF13[9]	Reserved	Reserved
ALMSF13[10]	LLI	LLI PARITY/ECC decoding alarm
ALMSF13[11]	DMA0	DMA lockstep mismatch alarm
ALMSF13[12]	DMA0	DMA SRI Read Data EDC Error
ALMSF13[13]	PPU	PPU SFF register monitor alarm
ALMSF13[14]	Reserved	Reserved
ALMSF13[15]	DMA1	DMA lockstep mismatch alarm
ALMSF13[16]	DMA1	DMA SRI Read Data EDC Error
ALMSF13[21]	DFT	Test mode Alarm
ALMSF13[22]	DFT	LBIST Alarm
ALMSF13[23:26]	Reserved	Reserved
ALMSF13[27]	PPU	PPU_WDT_TO (PPU watchdog timer timeout)
ALMSF13[28]	PPU	PPU_HSSI_ERR (PPU safety monitor alarm)
ALMSF13[30]	PPU	PPU SFF register monitor alarm
ALMSF13[31]	PPU	PPU shutdown handling alarm

Table 2098 Alarm mapping table for ALMSF14

Alarm index	Functional block instance	Alarm name
ALMSF14[0]	SRI3	SRI XBAR SOTA Swap Configuration Alarm
ALMSF14[1]	SRI3	SRI XBAR Address Phase EDC Error Alarm
ALMSF14[2]	SRI3	SRI XBAR Write Data EDC Error Alarm
ALMSF14[3]	SRI3	SRI XBAR Bus Error Event Alarm
ALMSF14[4]	SRI5	SRI XBAR SOTA Swap Configuration Alarm
ALMSF14[5]	SRI5	SRI XBAR Address Phase EDC Error Alarm
ALMSF14[6]	SRI5	SRI XBAR Write Data EDC Error Alarm
ALMSF14[7]	SRI5	SRI XBAR Bus Error Event Alarm
ALMSF14[8]	SRI6	SRI XBAR SOTA Swap Configuration Alarm
ALMSF14[9]	SRI6	SRI XBAR Address Phase EDC Error Alarm
ALMSF14[10]	SRI6	SRI XBAR Write Data EDC Error Alarm
ALMSF14[11]	SRI6	SRI XBAR Bus Error Event Alarm
ALMSF14[12]	SRI7	SRI XBAR SOTA Swap Configuration Alarm
ALMSF14[13]	SRI7	SRI XBAR Address Phase EDC Error Alarm
ALMSF14[14]	SRI7	SRI XBAR Write Data EDC Error Alarm
ALMSF14[15]	SRI7	SRI XBAR Bus Error Event Alarm
ALMSF14[16]	TC4Dx: S2F_3SPB_0	SRI S2F Address Phase EDC Error Alarm
ALMSF14[17]	TC4Dx: S2F_3SPB_0	SRI S2F Write Data EDC Error Alarm
ALMSF14[18]	TC4Dx: S2F_7TPB_0	SRI S2F Address Phase EDC Error Alarm
ALMSF14[19]	TC4Dx: S2F_7TPB_0	SRI S2F Write Data EDC Error Alarm
ALMSF14[20:23]	Reserved	Reserved
ALMSF14[24]	S2A PPU.VMEM	SRI S2A Address Phase EDC Error Alarm
ALMSF14[25]	S2A PPU.VMEM	SRI S2A Write Data EDC Error Alarm
ALMSF14[26:31]	Reserved	Reserved

Table 2099 Alarm mapping table for ALMSF15

Alarm index	Functional block instance	Alarm name
ALMSF15[0]	LMU8 LMU9	LMU internal data transfer EDC error alarm LMU internal data transfer EDC error alarm
ALMSF15[1]	LMU8 LMU9	APU Alarm APU Alarm
ALMSF15[2]	Reserved	Reserved

(table continues...)

Table 2099 (continued) Alarm mapping table for ALMSF15

Alarm index	Functional block instance	Alarm name
ALMSF15[3]	LMU8 LMU9	LMU SRI Address Phase EDC Error LMU SRI Address Phase EDC Error
ALMSF15[4]	LMU8 LMU9	LMU SRI Write Data EDC Error LMU SRI Write Data EDC Error
ALMSF15[5]	LMU8 LMU9	LMU control redundancy mismatch LMU control redundancy mismatch
ALMSF15[6]	LMU8 LMU9	LMU control redundancy self-test mismatch LMU control redundancy self-test mismatch
ALMSF15[7:24]	Reserved	Reserved
ALMSF15[25]	S2L eGTM	SRI S2L Address Phase EDC Error Alarm
ALMSF15[26]	S2L eGTM	SRI S2L Write Data EDC Error Alarm
ALMSF15[27]	Reserved	Reserved
ALMSF15[28]	CANXL	CANXL SRI Read Data EDC Error
ALMSF15[29]	PCIE1	PCIE configuration uncorrectable error alarm
ALMSF15[30]	Reserved	Reserved
ALMSF15[31]	PCIE1	PCIE data uncorrectable error alarm

Table 2100 Alarm mapping table for ALMSF16

Alarm index	Functional block instance	Alarm name
ALMSF16[0:31]	Reserved	Reserved

42.8.5.2 Safety (Standby) alarm mapping tables

Table 2101 Alarm mapping table for ALMSTDBY0

Alarm index	Functional block instance	Alarm name
ALMSTDBY0[11]	Reserved	Reserved
ALMSTDBY0[16]	Reserved	Reserved
ALMSTDBY0[20:31]	Reserved	Reserved

Table 2102 Alarm mapping table for ALMSTDBY1

Alarm index	Functional block instance	Alarm name
ALMSTDBY1[11]	Reserved	Reserved
ALMSTDBY1[16]	Reserved	Reserved

(table continues...)

Table 2102 (continued) Alarm mapping table for ALMSTDBY1

Alarm index	Functional block instance	Alarm name
ALMSTDBY1[20:31]	Reserved	Reserved

Table 2103 Alarm mapping table for ALMSTDBY2

Alarm index	Functional block instance	Alarm name
ALMSTDBY2[0]	PMS.VMONS0	VDDEVRSB Under Voltage Alarm
ALMSTDBY2[1]	PMS.VMONS0	VDDEXT Under Voltage Alarm
ALMSTDBY2[2]	PMS.VMONS0	VDDM Under Voltage Alarm
ALMSTDBY2[3]	Reserved	Reserved
ALMSTDBY2[4]	PMS.VMONS0	VDD Under Voltage Alarm
ALMSTDBY2[5]	PMS.VMONS0	VDDPMS0 Under Voltage Alarm
ALMSTDBY2[6]	PMS.VMONS0	VDDPMS1 Under Voltage Alarm
ALMSTDBY2[7]	PMS.VMONS0	VDDPMS2 Under Voltage Alarm
ALMSTDBY2[8]	PMS.VMONS0	VDDSBRAM Under Voltage Alarm
ALMSTDBY2[9]	PMS.VMONS0	VDDPAD Under Voltage Alarm
ALMSTDBY2[10]	Reserved	Reserved
ALMSTDBY2[11]	PMS.VMONS1	VDDPHPHY0 Under Voltage Alarm
ALMSTDBY2[12]	PMS.VMONS1	VDDPHPHY1 Under Voltage Alarm
ALMSTDBY2[13]	PMS.VMONS1	VDDPHPHY2 Under Voltage Alarm
ALMSTDBY2[14]	PMS.VMONS1	VDDHSIF Under Voltage Alarm
ALMSTDBY2[15]	Reserved	Reserved
ALMSTDBY2[16]	PMS.VMONS1	VDDPHY0 Under Voltage Alarm
ALMSTDBY2[17]	PMS.VMONS1	VDDPHY1 Under Voltage Alarm
ALMSTDBY2[18]	PMS.VMONS1	VDDPHY2 Under Voltage Alarm
ALMSTDBY2[19]	PMS.VMONS1	VDDCPULMU0 Under Voltage Alarm
ALMSTDBY2[20]	PMS.VMONS1	VDDCPULMU1 Under Voltage Alarm
ALMSTDBY2[21]	PMS.VMONS1	VDDPPU Under Voltage Alarm
ALMSTDBY2[22]	PMS.VMONP3	VDDEXTDC Under Voltage Alarm
ALMSTDBY2[23:31]	Reserved	Reserved

Table 2104 Alarm mapping table for ALMSTDBY3

Alarm index	Functional block instance	Alarm name
ALMSTDBY3[0]	PMS.VMONS0	VDDEVRSB OVER Voltage Alarm

(table continues...)

Table 2104 (continued) Alarm mapping table for ALMSTDBY3

Alarm index	Functional block instance	Alarm name
ALMSTDBY3[1]	PMS.VMONS0	VDDEXT OVER Voltage Alarm
ALMSTDBY3[2]	PMS.VMONS0	VDDM OVER Voltage Alarm
ALMSTDBY3[3]	Reserved	Reserved
ALMSTDBY3[4]	PMS.VMONS0	VDD OVER Voltage Alarm
ALMSTDBY3[5]	PMS.VMONS0	VDDPMS0 OVER Voltage Alarm
ALMSTDBY3[6]	PMS.VMONS0	VDDPMS1 OVER Voltage Alarm
ALMSTDBY3[7]	PMS.VMONS0	VDDPMS2 OVER Voltage Alarm
ALMSTDBY3[8]	PMS.VMONS0	VDDSBRAM OVER Voltage Alarm
ALMSTDBY3[9]	PMS.VMONS0	VDDPAD OVER Voltage Alarm
ALMSTDBY3[10]	Reserved	Reserved
ALMSTDBY3[11]	PMS.VMONS1	VDDPHPHY0 OVER Voltage Alarm
ALMSTDBY3[12]	PMS.VMONS1	VDDPHPHY1 OVER Voltage Alarm
ALMSTDBY3[13]	PMS.VMONS1	VDDPHPHY2 OVER Voltage Alarm
ALMSTDBY3[14]	PMS.VMONS1	VDDHSIF OVER Voltage Alarm
ALMSTDBY3[15]	Reserved	Reserved
ALMSTDBY3[16]	PMS.VMONS1	VDDPHY0 OVER Voltage Alarm
ALMSTDBY3[17]	PMS.VMONS1	VDDPHY1 OVER Voltage Alarm
ALMSTDBY3[18]	PMS.VMONS1	VDDPHY2 OVER Voltage Alarm
ALMSTDBY3[19]	PMS.VMONS1	VDDCPULMU0 OVER Voltage Alarm
ALMSTDBY3[20]	PMS.VMONS1	VDDCPULMU1 OVER Voltage Alarm
ALMSTDBY3[21]	PMS.VMONS1	VDDPPU OVER Voltage Alarm
ALMSTDBY3[22]	PMS.VMONP3	VDDEXTDC OVER Voltage Alarm
ALMSTDBY3[23:31]	Reserved	Reserved

Table 2105 Alarm mapping table for ALMSTDBY4

Alarm index	Functional block instance	Alarm name
ALMSTDBY4[0]	SMU.SMU_CS	SMU_CS Critical Alarm
ALMSTDBY4[1]	SMU.SMU_CS	SMU_CS Alive Alarm
ALMSTDBY4[2]	SMU.SMU_SAFE0	SMU_SAFE0 Alive Alarm
ALMSTDBY4[3]	SMU.SMU_SAFE1	SMU_SAFE1 Alive Alarm
ALMSTDBY4[4]	SMU.SMU_SAFE0	SMU_SAFE0 Critical Alarm
ALMSTDBY4[5]	SMU.SMU_SAFE1	SMU_SAFE1 Critical Alarm
ALMSTDBY4[8]	PMS	PMS SFF Uncorrectable Error Alarm

(table continues...)

Table 2105 (continued) Alarm mapping table for ALMSTDBY4

Alarm index	Functional block instance	Alarm name
ALMSTDBY4[11]	PMS.DTS	PMS.DTS over temperature alarm
ALMSTDBY4[12]	CLOCKING	System-PLL clock alive error
ALMSTDBY4[16]	CLOCKING	System-PLL K-pre-divider K-divider error System clock divider error
ALMSTDBY4[17]	SMU_SAFE0 SMU_SAFE1 SMU_CS	SMU_SAFE0/1 SMU_CS SFF Uncorrectable Error Alarm
ALMSTDBY4[18]	SMM	SMM_RESET_ALARM
ALMSTDBY4[19:31]	Reserved	Reserved

42.8.5.3 System trigger alarm mapping tables

System trigger alarms originate from safety mechanisms that are not used in the safety concept. Therefore, they are not relevant to the safety integrity of the device and the system integrator is not required to configure a safe reaction upon their occurrence. However, they can still be used in the system if it is considered as useful, for example to generate a fast reaction on a specific event. In such a case, it is the responsibility of the system integrator to ensure that the use of these alarms meets the needs of the application. Similar to any safety alarm, system trigger alarms can be configured within SMU_SAFE0, SMU_SAFE1 or SMU_STDBY to trigger a specific reaction.

Table 2106 System trigger alarm mapping table for ALMSF0

Alarm index	Functional block instance	Alarm name
ALMSF0[18]	NVM.PFRWB0A	PFLASH ECC single bit error
ALMSF0[19]	NVM.PFRWB0B	PFLASH ECC single bit error

Table 2107 System trigger alarm mapping table for ALMSF1

Alarm index	Functional block instance	Alarm name
ALMSF1[10]	VMT0	VMT Correctable Error
ALMSF1[13]	VMT1	VMT Correctable Error
ALMSF1[18]	NVM.PFRWB1A	PFLASH ECC single bit error
ALMSF1[19]	NVM.PFRWB1B	PFLASH ECC single bit error

Table 2108 System trigger alarm mapping table for ALMSF2

Alarm index	Functional block instance	Alarm name
ALMSF2[10]	VMT2	VMT Correctable Error
ALMSF2[13]	VMT3	VMT Correctable Error

(table continues...)

Table 2108 (continued) System trigger alarm mapping table for ALMSF2

Alarm index	Functional block instance	Alarm name
ALMSF2[18]	NVM.PFRWB2A	PFLASH ECC single bit error
ALMSF2[19]	NVM.PFRWB2B	PFLASH ECC single bit error

Table 2109 System trigger alarm mapping table for ALMSF3

Alarm index	Functional block instance	Alarm name
ALMSF3[9]	VMT4	VMT Miscellaneous Error
ALMSF3[10]	VMT4	VMT Correctable Error
ALMSF3[11]	VMT4	VMT Uncorrectable Error
ALMSF3[13]	VMT5	VMT Correctable Error
ALMSF3[15]	VMT4	VMT SFF Uncorrectable Error
ALMSF3[18]	NVM.PFRWB3A	PFLASH ECC single bit error
ALMSF3[19]	NVM.PFRWB3B	PFLASH ECC single bit error

Table 2110 System trigger alarm mapping table for ALMSF4

Alarm index	Functional block instance	Alarm name
ALMSF4[9]	VMT6	VMT Miscellaneous Error
ALMSF4[11]	VMT6	VMT Uncorrectable Error
ALMSF4[15]	VMT6	VMT SFF Uncorrectable Error
ALMSF4[18]	NVM.PFRWB4A	PFLASH ECC single bit error
ALMSF4[19]	NVM.PFRWB4B	PFLASH ECC single bit error

Table 2111 System trigger alarm mapping table for ALMSF5

Alarm index	Functional block instance	Alarm name
ALMSF5[18]	NVM.PFRWB5A	PFLASH ECC single bit error
ALMSF5[19]	NVM.PFRWB5B	PFLASH ECC single bit error

Table 2112 System trigger alarm mapping table for ALMSF6

Alarm index	Functional block instance	Alarm name
ALMSF6[22]	XSPI	XSPI FSM alarm
ALMSF6[26]	HSPHY	HSPHY data uncorrectable error alarm

Table 2113 System trigger alarm mapping table for ALMSF7

Alarm index	Functional block instance	Alarm name
ALMSF7[30]	DEBUG	ocds_mode - Debug active mode

Table 2114 System trigger alarm mapping table for ALMSF8

Alarm index	Functional block instance	Alarm name
ALMSF8[11]	PORT.PAD	OK_PADS_BOT (Bottom PADS functionality not reliable)
ALMSF8[12]	PORT.PAD	OK_PADS_RTH (Right PADS functionality not reliable)
ALMSF8[13]	PORT.PAD	OK_PADS_TOP (Top PADS functionality not reliable)
ALMSF8[14]	PORT.PAD	OK_PADS_LEF (Left PADS functionality not reliable)
ALMSF8[25]	PPU	PPU_PERSONALITY_MODE_ERR

Table 2115 System trigger alarm mapping table for ALMSF9

Alarm index	Functional block instance	Alarm name
ALMSF9[0]	VTMON	VTMON.DTS0 Under Temperature Alarm
	VTMON	VTMON.DTS1 Under Temperature Alarm
	VTMON	VTMON.DTS2 Under Temperature Alarm
	VTMON	VTMON.DTS3 Under Temperature Alarm
	VTMON	VTMON.DTS4 Under Temperature Alarm
ALMSF9[2]	VTMON	VTMON.DTS0 Thermal Shut down Alarm
ALMSF9[4]	VTMON	VTMON.DTS1 Thermal Shut down Alarm
ALMSF9[6]	VTMON	VTMON.DTS2 Thermal Shut down Alarm
ALMSF9[8]	VTMON	VTMON.DTS3 Thermal Shut down Alarm
ALMSF9[10]	VTMON	VTMON.DTS4 Thermal Shut down Alarm
ALMSF9[13]	PMS.DTS	PMS.DTS Under Temperature Alarm
ALMSF9[14]	PMS.DTS	PMS.DTS Thermal Shut down Alarm
ALMSF9[15]	PMS	DCDC Sync Alarm
ALMSF9[20]	SCU	SCU External Request Unit alarm 0
ALMSF9[21]	SCU	SCU External Request Unit alarm 1
ALMSF9[22]	SCU	SCU External Request Unit alarm 2
ALMSF9[23]	SCU	SCU External Request Unit alarm 3
ALMSF9[24]	SCU	SCU External Request Unit alarm 4
ALMSF9[25]	SCU	SCU External Request Unit alarm 5
ALMSF9[26]	SCU	SCU External Request Unit alarm 6
ALMSF9[27]	SCU	SCU External Request Unit alarm 7

Table 2116 System trigger alarm mapping table for ALMSF10

Alarm index	Functional block instance	Alarm name
ALMSF10[22]	SCR	XRAM uncorrectable error
ALMSF10[23]	PMS.VMONP0	VDD OVER Voltage Alarm
ALMSF10[24]	PMS.VMONP0	VDD Under Voltage Alarm
ALMSF10[25]	PMS.VMONP1	VDDEXT OVER Voltage Alarm
	PMS.VMONP2	VDDEXTDC OVER Voltage Alarm
	PMS.VMONP4	VDDEVRSB OVER Voltage Alarm
ALMSF10[26]	PMS.VMONP1	VDDEXT Under Voltage Alarm
	PMS.VMONP2	VDDEXTDC Under Voltage Alarm
	PMS.VMONP4	VDDEVRSB Under Voltage Alarm
ALMSF10[27]	PMS.VMONP2	(VDDEXTDC-VSSDCHS) OVER Voltage Alarm
	PMS.VMONP3	VDDDCLS OVER Voltage Alarm
	PMS.VMONP4	VDDM OVER Voltage Alarm
	PMS.VMONP4	VDDPHPHY0 OVER Voltage Alarm
	PMS.VMONP4	VDDPHPHY1 OVER Voltage Alarm
	PMS.VMONP4	VDDPHPHY2 OVER Voltage Alarm
	PMS.VMONP4	VDDHSIF OVER Voltage Alarm
	PMS.VMONP4	VDDPAD OVER Voltage Alarm
	PMS.VMONP4	VDDCPULMU0 OVER Voltage Alarm
	PMS.VMONP4	VDDCPULMU1 OVER Voltage Alarm
	PMS.VMONP4	VDDPPU OVER Voltage Alarm
	PMS.VMONP4	VDDPHY0 OVER Voltage Alarm
	PMS.VMONP4	VDDPHY1 OVER Voltage Alarm
	PMS.VMONP4	VDDPHY2 OVER Voltage Alarm
ALMSF10[28]	PMS.VMONP2	(VDDEXTDC-VSSDCHS) Under Voltage Alarm
	PMS.VMONP3	VDDDCLS Under Voltage Alarm
	PMS.VMONP4	VDDM Under Voltage Alarm
	PMS.VMONP4	VDDPHPHY0 Under Voltage Alarm
	PMS.VMONP4	VDDPHPHY1 Under Voltage Alarm
	PMS.VMONP4	VDDPHPHY2 Under Voltage Alarm
	PMS.VMONP4	VDDHSIF Under Voltage Alarm
	PMS.VMONP4	VDDPAD Under Voltage Alarm
	PMS.VMONP4	VDDCPULMU0 Under Voltage Alarm
	PMS.VMONP4	VDDCPULMU1 Under Voltage Alarm
	PMS.VMONP4	VDDPPU Under Voltage Alarm
	PMS.VMONP4	VDDPHY0 Under Voltage Alarm
	PMS.VMONP4	VDDPHY1 Under Voltage Alarm
	PMS.VMONP4	VDDPHY2 Under Voltage Alarm

(table continues...)

Table 2116 (continued) System trigger alarm mapping table for ALMSF10

Alarm index	Functional block instance	Alarm name
ALMSF10[31]	PMS	EVRC Short to LOW Alarm EVRC Short to HIGH Alarm

Table 2117 System trigger alarm mapping table for ALMSF12

Alarm index	Functional block instance	Alarm name
ALMSF12[16]	PORT00	PORT Access Protection Violation Alarm
	PORT01	PORT Access Protection Violation Alarm
	PORT02	PORT Access Protection Violation Alarm
	PORT03	PORT Access Protection Violation Alarm
	PORT04	PORT Access Protection Violation Alarm
	PORT10	PORT Access Protection Violation Alarm
	PORT13	PORT Access Protection Violation Alarm
	PORT14	PORT Access Protection Violation Alarm
	PORT15	PORT Access Protection Violation Alarm
	PORT16	PORT Access Protection Violation Alarm
	PORT20	PORT Access Protection Violation Alarm
	PORT21	PORT Access Protection Violation Alarm
	PORT22	PORT Access Protection Violation Alarm
	PORT23	PORT Access Protection Violation Alarm
	PORT25	PORT Access Protection Violation Alarm
	PORT30	PORT Access Protection Violation Alarm
	PORT31	PORT Access Protection Violation Alarm
	PORT32	PORT Access Protection Violation Alarm
	PORT33	PORT Access Protection Violation Alarm
	PORT34	PORT Access Protection Violation Alarm
	PORT35	PORT Access Protection Violation Alarm
	PORT40	PORT Access Protection Violation Alarm

Table 2118 System trigger alarm mapping table for ALMSF13

Alarm index	Functional block instance	Alarm name
ALMSF13[17]	GETH	SRC_GETHCE (Safety alarm for correctable errors on MAC0 or MAC1)
ALMSF13[18]	GETH	SRC_GETHSE (SRI ECC Error alarm on any SRI master or slave of GETH)
ALMSF13[19]	GETH	SRC_GETHNCE (Safety alarm for uncorrectable errors on MAC0 or MAC1)
ALMSF13[20]	SMU	External Emergency Stop Event

(table continues...)

Table 2118 (continued) System trigger alarm mapping table for ALMSF13

Alarm index	Functional block instance	Alarm name
ALMSF13[29]	PPU	PPU_HS_SBE (PPU single bit error alarm)

Table 2119 System trigger alarm mapping table for ALMSH0

Alarm index	Functional block instance	Alarm name
ALMSH0[18]	NVM.PFRWBCS	PFLASH ECC single bit error

Table 2120 System trigger alarm mapping table for ALMSH2

Alarm index	Functional block instance	Alarm name
ALMSH2[1]	VMT0	VMT Correctable Error
ALMSH2[4]	VMT1	VMT Correctable Error
ALMSH2[7]	VMT2	VMT Correctable Error
ALMSH2[10]	VMT3	VMT Correctable Error
ALMSH2[13]	VMT4	VMT Correctable Error
ALMSH2[16]	VMT5	VMT Correctable Error
ALMSH2[19]	VMT6	VMT Correctable Error

Table 2121 System trigger alarm mapping table for ALMSTDBY0

Alarm index	Functional block instance	Alarm name
ALMSTDBY0[0]	PMS.VMONP0	VDD Under Voltage Alarm
ALMSTDBY0[1]	PMS.VMONP1	VDDEXT Under Voltage Alarm
ALMSTDBY0[2]	PMS.VMONP2	(VDDEXTDC-VSSDCHS) Under Voltage Alarm
ALMSTDBY0[3]	PMS.VMONP2	VDDEXTDC Under Voltage Alarm
ALMSTDBY0[4]	PMS.VMONP3	VDDDCLS Under Voltage Alarm
ALMSTDBY0[5]	PMS.VMONP4	VDDEVRSB Under Voltage Alarm
ALMSTDBY0[6]	PMS.VMONP4	VDDM Under Voltage Alarm
ALMSTDBY0[7]	PMS.VMONP4	VDDPHPHY0 Under Voltage Alarm
ALMSTDBY0[8]	PMS.VMONP4	VDDPHPHY1 Under Voltage Alarm
ALMSTDBY0[9]	PMS.VMONP4	VDDPHPHY2 Under Voltage Alarm
ALMSTDBY0[10]	PMS.VMONP4	VDDHSIF Under Voltage Alarm
ALMSTDBY0[12]	PMS.VMONP4	VDDPAD Under Voltage Alarm
ALMSTDBY0[13]	PMS.VMONP4	VDDCPULMU0 Under Voltage Alarm
ALMSTDBY0[14]	PMS.VMONP4	VDDCPULMU1 Under Voltage Alarm

(table continues...)

Table 2121 (continued) System trigger alarm mapping table for ALMSTDBY0

Alarm index	Functional block instance	Alarm name
ALMSTDBY0[15]	PMS.VMONP4	VDDPPU Under Voltage Alarm
ALMSTDBY0[17]	PMS.VMONP4	VDDPHY0 Under Voltage Alarm
ALMSTDBY0[18]	PMS.VMONP4	VDDPHY1 Under Voltage Alarm
ALMSTDBY0[19]	PMS.VMONP4	VDDPHY2 Under Voltage Alarm

Table 2122 System trigger alarm mapping table for ALMSTDBY1

Alarm index	Functional block instance	Alarm name
ALMSTDBY1[0]	PMS.VMONP0	VDD OVER Voltage Alarm
ALMSTDBY1[1]	PMS.VMONP1	VDDEXT OVER Voltage Alarm
ALMSTDBY1[2]	PMS.VMONP2	(VDDEXTDC-VSSDCHS) OVER Voltage Alarm
ALMSTDBY1[3]	PMS.VMONP2	VDDEXTDC OVER Voltage Alarm
ALMSTDBY1[4]	PMS.VMONP3	VDDDCLS OVER Voltage Alarm
ALMSTDBY1[5]	PMS.VMONP4	VDDEVRSB OVER Voltage Alarm
ALMSTDBY1[6]	PMS.VMONP4	VDDM OVER Voltage Alarm
ALMSTDBY1[7]	PMS.VMONP4	VDDPHPHY0 OVER Voltage Alarm
ALMSTDBY1[8]	PMS.VMONP4	VDDPHPHY1 OVER Voltage Alarm
ALMSTDBY1[9]	PMS.VMONP4	VDDPHPHY2 OVER Voltage Alarm
ALMSTDBY1[10]	PMS.VMONP4	VDDHSIF OVER Voltage Alarm
ALMSTDBY1[12]	PMS.VMONP4	VDDPAD OVER Voltage Alarm
ALMSTDBY1[13]	PMS.VMONP4	VDDCPULMU0 OVER Voltage Alarm
ALMSTDBY1[14]	PMS.VMONP4	VDDCPULMU1 OVER Voltage Alarm
ALMSTDBY1[15]	PMS.VMONP4	VDDPPU OVER Voltage Alarm
ALMSTDBY1[17]	PMS.VMONP4	VDDPHY0 OVER Voltage Alarm
ALMSTDBY1[18]	PMS.VMONP4	VDDPHY1 OVER Voltage Alarm
ALMSTDBY1[19]	PMS.VMONP4	VDDPHY2 OVER Voltage Alarm

Table 2123 System trigger alarm mapping table for ALMSTDBY4

Alarm index	Functional block instance	Alarm name
ALMSTDBY4[6]	SCR	XRAM uncorrectable error
ALMSTDBY4[7]	PMS.PORDET	VDDLVDEVRSB DET Under Voltage Alarm
ALMSTDBY4[9]	PMS.DTS	PMS.DTS Thermal Shut down Alarm
ALMSTDBY4[10]	PMS.DTS	PMS.DTS Under Temperature Alarm

(table continues...)

Table 2123 (continued) System trigger alarm mapping table for ALMSTDBY4

Alarm index	Functional block instance	Alarm name
ALMSTDBY4[13]	PMS	EVRC Short to LOW Alarm
ALMSTDBY4[14]	PMS	EVRC Short to HIGH Alarm
ALMSTDBY4[15]	PORT.PAD	OK_PADS_BOT (Bottom PADS functionality not reliable) OK_PADS_RTH (Right PADS functionality not reliable) OK_PADS_TOP (Top PADS functionality not reliable) OK_PADS_LEF (Left PADS functionality not reliable)

42.8.5.4 Shared (safety or security) alarm mapping tables

Table 2124 Alarm mapping table for ALMSH0

Alarm index	Functional block instance	Alarm name
ALMSH0[0]	Reserved	Reserved
ALMSH0[1]	CPUCS	CPU access protection alarm
ALMSH0[2]	CPUCS	CPU SOTA select redundancy error alarm
ALMSH0[3]	CPUCS	CPU CFI access protection alarm
ALMSH0[4]	CPUCS	CPU CFI wait state error
ALMSH0[5]	CPUCS	CPU instruction fetch EDC error alarm
ALMSH0[6]	CPUCS	CPU data side access EDC error alarm
ALMSH0[7]	NVM.PFRWBCS	PFLASH ECC double bit error
ALMSH0[8:13]	Reserved	Reserved
ALMSH0[14]	NVM.FSI1	NVM incorrect bank monitoring
ALMSH0[15]	NVM.FSI1	NVM static configurations MISR error
ALMSH0[16]	NVM.DFRWB1	DFLASH static configurations MISR error
ALMSH0[17]	NVM.PFRWBCS	PFLASH dynamic configurations SFF error
ALMSH0[19]	Reserved	Reserved
ALMSH0[20]	NVM.PFRWBCS	PFLASH ECC error
ALMSH0[21]	NVM.PFRWBCS	PFLASH EDC error
ALMSH0[22]	NVM.PFRWBCS	PFLASH ECC single bit error address buffer (SBAB) full
ALMSH0[23]	NVM.PFRWBCS	PFLASH ECC double bit error address buffer (DBAB) full
ALMSH0[24]	NVM.PFRWBCS	PFLASH ECC uncorrectable error address buffer (UBAB) full
ALMSH0[25]	NVM.PFRWBCS	PFLASH static configurations MISR error

(table continues...)

Table 2124 (continued) Alarm mapping table for ALMSH0

Alarm index	Functional block instance	Alarm name
ALMSH0[26:30]	Reserved	Reserved
ALMSH0[31]	WTU	WTU CPUcs WDT overflow or access error alarm

Table 2125 Alarm mapping table for ALMSH1

Alarm index	Functional block instance	Alarm name
ALMSH1[0:6]	Reserved	Reserved
ALMSH1[7]	CSS	SFF_ALARM
ALMSH1[8:9]	Reserved	Reserved
ALMSH1[10]	CSRM.CSCU	CSCU SFF Uncorrectable Error
ALMSH1[11:15]	Reserved	Reserved
ALMSH1[16]	FPI.CSBCU	FPI error alarm (Alarm_NCS) FPI EDC error alarm (Alarm_NCS)
ALMSH1[17:31]	Reserved	Reserved

Table 2126 Alarm mapping table for ALMSH2

Alarm index	Functional block instance	Alarm name
ALMSH2[0]	VMT0	VMT Miscellaneous Error
ALMSH2[2]	VMT0	VMT Uncorrectable Error
ALMSH2[3]	VMT1	VMT Miscellaneous Error
ALMSH2[5]	VMT1	VMT Uncorrectable Error
ALMSH2[6]	VMT2	VMT Miscellaneous Error
ALMSH2[8]	VMT2	VMT Uncorrectable Error
ALMSH2[9]	VMT3	VMT Miscellaneous Error
ALMSH2[11]	VMT3	VMT Uncorrectable Error
ALMSH2[12]	VMT4	VMT Miscellaneous Error
ALMSH2[14]	VMT4	VMT Uncorrectable Error
ALMSH2[15]	VMT5	VMT Miscellaneous Error
ALMSH2[17]	VMT5	VMT Uncorrectable Error
ALMSH2[18]	VMT6	VMT Miscellaneous Error
ALMSH2[20]	VMT6	VMT Uncorrectable Error
ALMSH2[21:31]	Reserved	Reserved

42.8.5.5 Security alarm mapping tables

Table 2127 Alarm mapping table for ALMCS0

Alarm index	Functional block instance	Alarm name
ALMCS0[0]	PMS.VMONP0	VDD OVER Voltage Alarm
ALMCS0[1]	PMS.VMONP0	VDD Under Voltage Alarm
ALMCS0[2]	PMS.VMONP1	VDDEXT OVER Voltage Alarm
	PMS.VMONP2	VDDEXTDC OVER Voltage Alarm
	PMS.VMONP4	VDDEVRSB OVER Voltage Alarm
ALMCS0[3]	PMS.VMONP1	VDDEXT Under Voltage Alarm
	PMS.VMONP2	VDDEXTDC Under Voltage Alarm
	PMS.VMONP4	VDDEVRSB Under Voltage Alarm
ALMCS0[4]	PMS.VMONP2	(VDDEXTDC-VSSDCHS) OVER Voltage Alarm
	PMS.VMONP3	VDDDCLS OVER Voltage Alarm
	PMS.VMONP4	VDDM OVER Voltage Alarm
	PMS.VMONP4	VDDPHPHY0 OVER Voltage Alarm
	PMS.VMONP4	VDDPHPHY1 OVER Voltage Alarm
	PMS.VMONP4	VDDPHPHY2 OVER Voltage Alarm
	PMS.VMONP4	VDDHSIF OVER Voltage Alarm
	PMS.VMONP4	VDDPAD OVER Voltage Alarm
	PMS.VMONP4	VDDCPULMU0 OVER Voltage Alarm
	PMS.VMONP4	VDDCPULMU1 OVER Voltage Alarm
	PMS.VMONP4	VDDPPU OVER Voltage Alarm
	PMS.VMONP4	VDDPHY0 OVER Voltage Alarm
	PMS.VMONP4	VDDPHY1 OVER Voltage Alarm
	PMS.VMONP4	VDDPHY2 OVER Voltage Alarm
ALMCS0[5]	PMS.VMONP2	(VDDEXTDC-VSSDCHS) Under Voltage Alarm
	PMS.VMONP3	VDDDCLS Under Voltage Alarm
	PMS.VMONP4	VDDM Under Voltage Alarm
	PMS.VMONP4	VDDPHPHY0 Under Voltage Alarm
	PMS.VMONP4	VDDPHPHY1 Under Voltage Alarm
	PMS.VMONP4	VDDPHPHY2 Under Voltage Alarm
	PMS.VMONP4	VDDHSIF Under Voltage Alarm
	PMS.VMONP4	VDDPAD Under Voltage Alarm
	PMS.VMONP4	VDDCPULMU0 Under Voltage Alarm
	PMS.VMONP4	VDDCPULMU1 Under Voltage Alarm
	PMS.VMONP4	VDDPPU Under Voltage Alarm
	PMS.VMONP4	VDDPHY0 Under Voltage Alarm
	PMS.VMONP4	VDDPHY1 Under Voltage Alarm
	PMS.VMONP4	VDDPHY2 Under Voltage Alarm

(table continues...)

Table 2127 (continued) Alarm mapping table for ALMCS0

Alarm index	Functional block instance	Alarm name
ALMCS0[6:9]	Reserved	Reserved
ALMCS0[10]	CSS	KEY_USAGE_ERROR
ALMCS0[11]	CSS	PHYSICAL_ERROR
ALMCS0[12]	CSS	AUTH_FAIL_ERROR
ALMCS0[13]	CSS	SECURED_DOMAIN_ALARM
ALMCS0[14]	Reserved	Reserved
ALMCS0[15]	DEBUG	DAP_active - DAP interface activated
ALMCS0[16]	DEBUG	trace_en - Trace logic enabled
ALMCS0[17]	DEBUG	DEBUG_IF_UNLOCKED - Debug interface unlocked
ALMCS0[18:19]	Reserved	Reserved
ALMCS0[20]	CSRM.CSROM	CSROM Access Violation Error
ALMCS0[21:24]	Reserved	Reserved
ALMCS0[25]	FPI.SBCU	FPI error alarm (Alarm_CS)
ALMCS0[26]	Reserved	Reserved
ALMCS0[27]	FPI.TBCU	FPI error alarm (Alarm_CS)
ALMCS0[28]	FPI.CSBCU	FPI error alarm (Alarm_CS)
ALMCS0[29]	FPI.COMBCU	FPI error alarm (Alarm_CS)
ALMCS0[30]	CLOCKING	SRI Clock Security error
ALMCS0[31]	IR	IR EDC error alarm (Alarm_CS)

Table 2128 Alarm mapping table for ALMCS1

Alarm index	Functional block instance	Alarm name
ALMCS1[0]	SMU	SMU_CS Software Alarm 0
ALMCS1[1]	SMU	SMU_CS Software Alarm 1
ALMCS1[2]	SMU	SMU_CS Software Alarm 2
ALMCS1[3]	SMU	SMU_CS Software Alarm 3
ALMCS1[4]	SMU	SMU_CS Software Alarm 4
ALMCS1[5]	SMU	SMU_CS Software Alarm 5
ALMCS1[6]	SMU	SMU_CS Software Alarm 6
ALMCS1[7]	SMU	SMU_CS Software Alarm 7
ALMCS1[8]	SRI0	SRI XBAR Bus Error Event Security Alarm
ALMCS1[9]	SRI1	SRI XBAR Bus Error Event Security Alarm
ALMCS1[10]	SRI2	SRI XBAR Bus Error Event Security Alarm

(table continues...)

Table 2128 (continued) Alarm mapping table for ALMCS1

Alarm index	Functional block instance	Alarm name
ALMCS1[11]	SRI3	SRI XBAR Bus Error Event Security Alarm
ALMCS1[12]	SRI4	SRI XBAR Bus Error Event Security Alarm
ALMCS1[13]	SRI5	SRI XBAR Bus Error Event Security Alarm
ALMCS1[14]	SRI6	SRI XBAR Bus Error Event Security Alarm
ALMCS1[15]	SRI7	SRI XBAR Bus Error Event Security Alarm
ALMCS1[16:17]	Reserved	Reserved
ALMCS1[18]	SMU	SMU_CS RT0 Time-out Alarm
ALMCS1[19]	Reserved	Reserved
ALMCS1[20]	CPU0	CPU lockstep mismatch alarm CPU CFI lockstep mismatch alarm
ALMCS1[21]	CPU0	CPU instruction fetch EDC error alarm
ALMCS1[22]	CPU0	CPU data side access EDC error alarm
ALMCS1[23]	Reserved	Reserved
ALMCS1[24]	SRI0	SRI XBAR Address Phase EDC Error Alarm
ALMCS1[25]	SRI0	SRI XBAR Write Data EDC Error Alarm
ALMCS1[26]	SRI1	SRI XBAR Address Phase EDC Error Alarm
ALMCS1[27]	SRI1	SRI XBAR Write Data EDC Error Alarm
ALMCS1[28]	SRI4	SRI XBAR Address Phase EDC Error Alarm
ALMCS1[29]	SRI4	SRI XBAR Write Data EDC Error Alarm
ALMCS1[30:31]	Reserved	Reserved

42.8.5.6 Alarm signal test mapping tables

Alarm signal test mapping table specify the alarms which can be tested as described in Safety Manual chapter ESM[SW]:SMU:ALARM_SIGNAL_TEST

Table 2129 Alarm signal test mapping table for ALMSF

Alarm index	Functional block instance	Alarm name
ALMSF5[10]	CSS	SAFETY_DOMAIN_ALARM0
ALMSF5[11]	CSS	SAFETY_DOMAIN_ALARM1
ALMSF5[12]	CSS	SAFETY_DOMAIN_ALARM2
ALMSF5[13]	CSS	SAFETY_DOMAIN_ALARM3
ALMSF5[14]	CSS	SAFETY_DOMAIN_ALARM4
ALMSF5[15]	CSS	SAFETY_DOMAIN_ALARM5

(table continues...)

Table 2129 (continued) Alarm signal test mapping table for ALMSF

Alarm index	Functional block instance	Alarm name
ALMSF8[25]	PPU	PPU_PERSONALITY_MODE_ERR
ALMSF13[31]	PPU	PPU shutdown handling alarm

Table 2130 Alarm signal test mapping table for ALMSTDBY

Alarm index	Functional block instance	Alarm name
ALMSTDBY4[18]	SMM	SMM_RESET_ALARM

Table 2131 Alarm signal test mapping table for ALMCS

Alarm index	Functional block instance	Alarm name
ALMCS0[10]	CSS	KEY_USAGE_ERROR
ALMCS0[11]	CSS	PHYSICAL_ERROR
ALMCS0[12]	CSS	AUTH_FAIL_ERROR
ALMCS0[13]	CSS	SECURED_DOMAIN_ALARM

42.8.6 TC4Dx SMU register monitor mapping tables

The Register Monitor Mapping tables specify the mapping between the Register Monitor interfaces implemented within the TC4Dx device and the assigned Register Monitor (RM) Index within SMU.

There is a direct relationship between the numbering of an RM Index and the corresponding configuration and status registers. For any RM index RM**i**[**z**], the same numbers **i** and **z** shall be used to locate the corresponding registers and bit-field as shown below:

- GCC_RMi_CTL.TE**z**
- GCC_RMi_EF.EF**z**
- GCC_RMi_STS.STS**z**

The following example shows the particular case with i=1 and z=10, but the same principle applies for all RM indexes and registers. For RM**1**[**10**] the following registers and bit-fields apply:

- GCC_RM1_CTL.TE**10**
- GCC_RM1_EF.EF**10**
- GCC_RM1_STS.STS**10**

For any RM Index having the corresponding Functional Block column marked as 'Reserved', it means that no Register Monitor Interface is mapped to that particular RM Index in the current device. As a consequence, the reading or writing to the corresponding bit-fields in the configuration and status registers as explained above will not have any impact.

42.8.6.1 Register monitor mapping tables

Table 2132 Register monitor mapping table for RM0

RM index	Functional block instance
RM0[0]	NVM.PFRWB0A
RM0[1]	NVM.PFRWB0B
RM0[2]	NVM.PFRWB1A
RM0[3]	NVM.PFRWB1B
RM0[4]	NVM.PFRWB2A
RM0[5]	NVM.PFRWB2B
RM0[6]	NVM.PFRWB3A
RM0[7]	NVM.PFRWB3B
RM0[8]	NVM.PFRWB4A
RM0[9]	NVM.PFRWB4B
RM0[10]	NVM.PFRWB5A
RM0[11]	NVM.PFRWB5B
RM0[12]	NVM.PFRWBCS
RM0[13]	CLOCK.SYSPLL
RM0[14]	CLOCK.PERPLL
RM0[15]	HSPHY
RM0[16]	PSI5
RM0[17]	VTMON
RM0[18]	WTU
RM0[19]	SCU
RM0[20]	CLOCK
RM0[21]	SMU.SMU_SAFE0/SMU_SAFE1, SMU.SMU_CS, SMU.SMU_GCC
RM0[22]	CSS
RM0[23]	PMS, SMU.SMU_STDBY
RM0[24]	VMT0
RM0[25]	VMT1
RM0[26]	VMT2
RM0[27]	Reserved
RM0[28]	VMT4
RM0[29]	Reserved
RM0[30]	VMT6
RM0[31]	SMM

Table 2133 Register monitor mapping table for RM1

RM index	Functional block instance
RM1[0]	DEBUG
RM1[1]	PORT32
RM1[2]	PORT33
RM1[3:6]	Reserved
RM1[7]	PPU
RM1[8:31]	Reserved

42.8.7 TC4Dx SMU connectivity

The table below lists interfaces to and from this functional block to other blocks in the device.

Table 2134 List of SMU interface signals

Interface signals	I/O	Description
CLOCK_SMU_fSPB	In	SPB clock input
SMU_DEBUG_TS16_SMU_SAFE0[15:0]	Out	SMU_SAFE0 trigger set mapped on OTGB0 or OTGB1
SMU_DEBUG_TS16_SMU_SAFE1[15:0]	Out	SMU_SAFE1 trigger set mapped on OTGB0 or OTGB1
SMU_DEBUG_TS16_SMU_CS[15:0]	Out	SMU_CS trigger set mapped on OTGB0 or OTGB1
SMU_IR_CSIRQ	Out	Interrupt Service Request from SMU_CS
SMU_IR_IRQ[5:0]	Out	Interrupt Service Request from SMU_SAFE0/1
SMU_SCU_CSTRAPREQ	Out	NMI Request from SMU_CS
SMU_SCU_PxTRAPREQ	Out	NMI Request from SMU_SAFE0/1
SMU_SMM_CS_RESET_REQ	Out	Reset Request from SMU_CS to SMM
SMU_SMM_CS_RESET_ALMGRP[2:0]	Out	Alarm Group information of alarm triggering the corresponding CS_RESET_REQ
SMU_SMM_CS_RESET_ALMID[4:0]	Out	Alarm index information of alarm triggering the corresponding CS_RESET_REQ
SMU_SMM_RESET[3:0]_REQ	Out	Reset Request from SMU_SAFE0/1 to SMM
SMU_SMM_RESET[3:0]_ALMGRP[4:0]	Out	Alarm Group information of alarm triggering the corresponding RESET[0:3]_REQ
SMU_SMM_RESET[3:0]_ALMID[4:0]	Out	Alarm index information of alarm triggering the corresponding RESET[0:3]_REQ
SMU_CPU1_RESET	Out	Reset Request to CPUs
SMU_CPU2_RESET	Out	Reset Request to CPUs
SMU_CPU3_RESET	Out	Reset Request to CPUs
SMU_CPU4_RESET	Out	Reset Request to CPUs
SMU_CPU5_RESET	Out	Reset Request to CPUs

(table continues...)

Table 2134 (continued) List of SMU interface signals

Interface signals	I/O	Description
SMU_PPU_PPU_RESET	Out	Reset Request to PPU
SMU_CSS_LOCK_ALL_KEYS	Out	Lock all keys request to CSS
SMU_CSS_LOCK_DBG_PROT_KEYS	Out	Lock debug protected keys request to CSS
SMU_PORTS_FSP[3:0]	Out	FSP Output Signal
SMU_PORTS_HW_DIR[3:0]	Out	FSP PORTS Direction
SMU_PORTS_HW_EN[3:0]	Out	FSP PORTS Enable - Enables SMU to control FSP PORTS
SMU_PORTS_PORTSEmergencyStop	Out	Emergency Stop Request Output
SMU_MSC_PORTSEmergencyStop	Out	Emergency Stop Request Output
PORTS_SMU_EXT_EMERGENCYSTOP_REQ	In	External Emergency Stop Request Input
PORTS_SMU_FSP_IN[3:0]	In	FSP Status Input - Shows the actual state of the FSP Output Pin
SMU_WTU_SAFE_RUN_STATE[3:0]	Out	SMU_SAFE0/1 State Machines SSM0/1 RUN State information to WTU
SMU_WTU_CS_RUN_STATE	Out	SMU_CS State Machine SSM RUN State information to WTU
CSS_SMU_KEY_USAGE_ERROR	In	Wrong usage of a key
CSS_SMU_PHYSICAL_ERROR	In	Detected hardware fault alarm with potential security risk
CSS_SMU_AUTH_FAIL_ERROR	In	Authentications Fail Overflow
CSRM.CSCU_SMU_CSAV[1:0]	In	filters alarms from OTGB

42.8.8 TC4Dx SMU revision history

Reference	Description of change(s)
Date range: 2024-08-17 to 2025-01-28	
Safety (Standby) alarm mapping tables	• Fix ALMSTDBY1[11,16,20:31] alarms which were wrongly listed as ALMSTDBY0[11,16,20:31]
Safety alarm mapping tables System trigger alarm mapping tables	• VMT6 has no correctable alarm, ALMSF4[10] is reserved

43 Flexible CRC Engine (FCE)

The FCE provides a parallel implementation of cyclic redundancy code (CRC) algorithms, implementing the IEEE 802.3 Ethernet CRC32, AUTOSAR CRC32P4, CCITT CRC16, and the SAE J1850 CRC8 polynomials. The FCE is targeted as a hardware acceleration engine for software applications or for operating system services that use CRC signatures.

43.1 Feature list

- A range of cyclic redundancy code (CRC) polynomials are implemented
  - CRC kernel 0: IEEE 802.3 CRC32 ethernet polynomial: 0x04C11DB7 - $x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$
  - CRC kernel 1: Autosar safety polynomial CRC32P4: 0xF4ACFB13 - $x^{32} + x^{31} + x^{30} + x^{29} + x^{28} + x^{26} + x^{23} + x^{21} + x^{19} + x^{18} + x^{15} + x^{14} + x^{13} + x^{12} + x^{11} + x^9 + x^8 + x^4 + x + 1$
  - CRC kernel 2: CCITT CRC16 polynomial: 0x1021 - $x^{16} + x^{12} + x^5 + 1$
  - CRC kernel 3: SAE J1850 CRC8 polynomial: 0x1D - $x^8 + x^4 + x^3 + x^2 + 1$

  ***Note**: The polynomial hexadecimal representation covers the coefficients (degree - 1) down to 0.*
- Parallel CRC computation
- Parallel channels of CRC context
  - Up to 8 parallel channels of CRC context are supported
  - The channels can be configured to use any one of the implemented algorithms
  - Different channels can use the same or different CRC algorithms concurrently
- Register Interface for CRC computation
  - Input register
  - CRC register
  - Configuration registers that enable control of the CRC operation and perform automatic checksum validation at the end of a message
  - Extended register interface to control the reliability of FCE execution in safety applications
  - Supports endianness conversion
- Independent module reset by software. The reset affects all CRC channels
- Error notification scheme through dedicated interrupt node for:
  - Transient error detection
    - Error interrupt generation (maskable) with local status register (cleared by software)
  - Checksum failure
    - Error interrupt generation (maskable) with local status register (cleared by software)
- Provides one interrupt line to the interrupt system

43.2 Functional overview

The FCE is a standard peripheral slave module which is controlled over a set of memory mapped registers. The FCE is fully synchronous with the peripheral bus clock and runs with a 1:1 clock ratio.

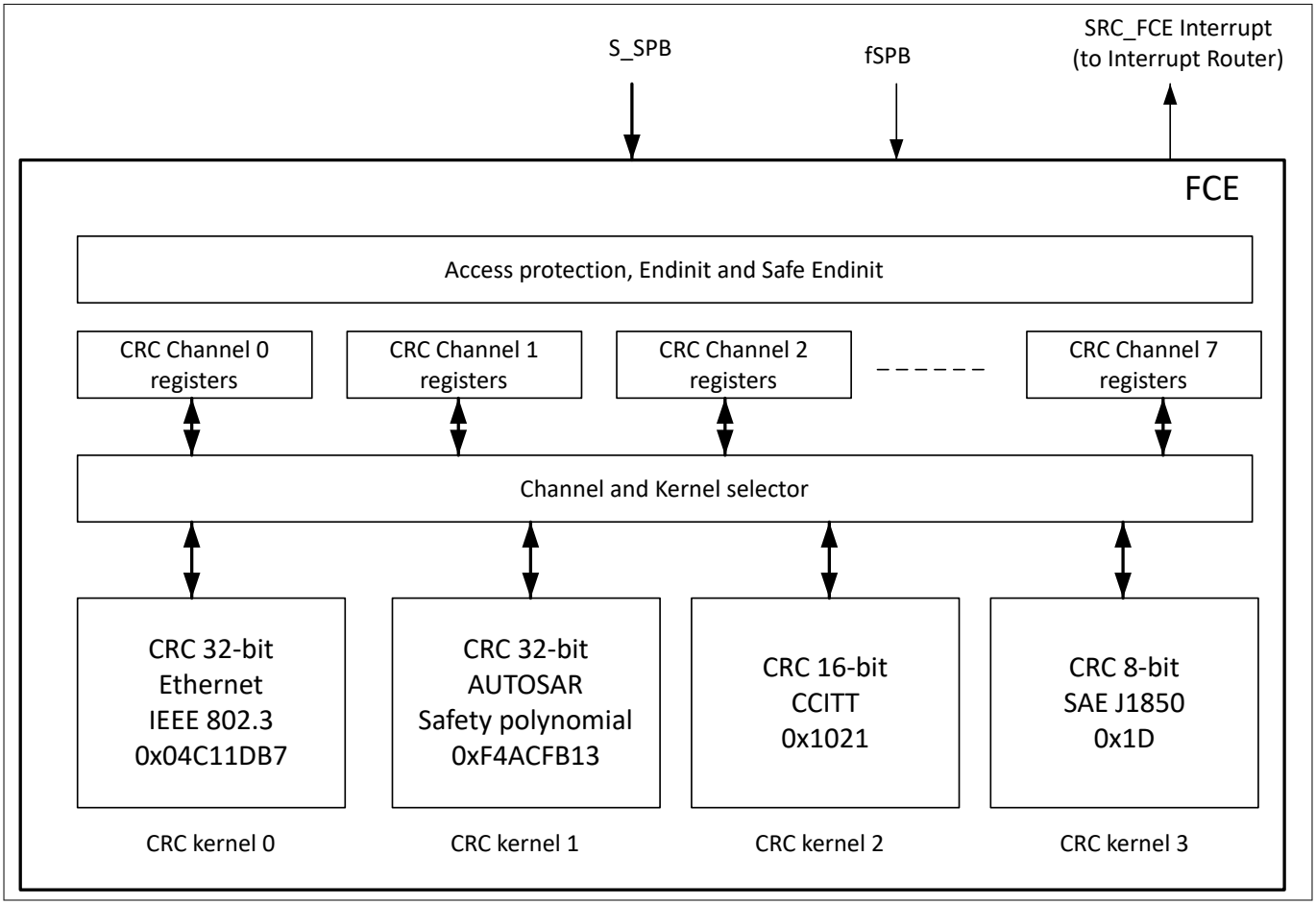


Figure 1007 FCE block diagram

An FCE kernel is the instantiation of the CRC algorithm as a parallel matrix. Each FCE kernel implements a different CRC algorithm.

The FCE module contains 8 channels (Channel 0 - Channel 7). Each channel is provided with its own set of registers which allows the software to configure the channel, choose the algorithm kernel used, write the inputs, and read the results and status from the CRC computation.

Each channel has an interrupt associated with it. A status register is associated with each channel that enables the software to identify which interrupt source is active.

To aid fast interrupt handling in software, a channels status register is provided, which ORs together the status from each channel. The software can read this one register to get information regarding the pending status from each channel.

Each FCE channel presents the same hardware and software architecture for computing the CRC. The following chapters only focus on the description of the generic CRC computational flow, since each channel can be independently configured to use any CRC kernel.

Related information

TC4Dx SMU alarm mapping tables on page 7233

43.3 Functional description

The following section describes the features of FCE in detail.

The generic CRC computational flow is shown in figure below.

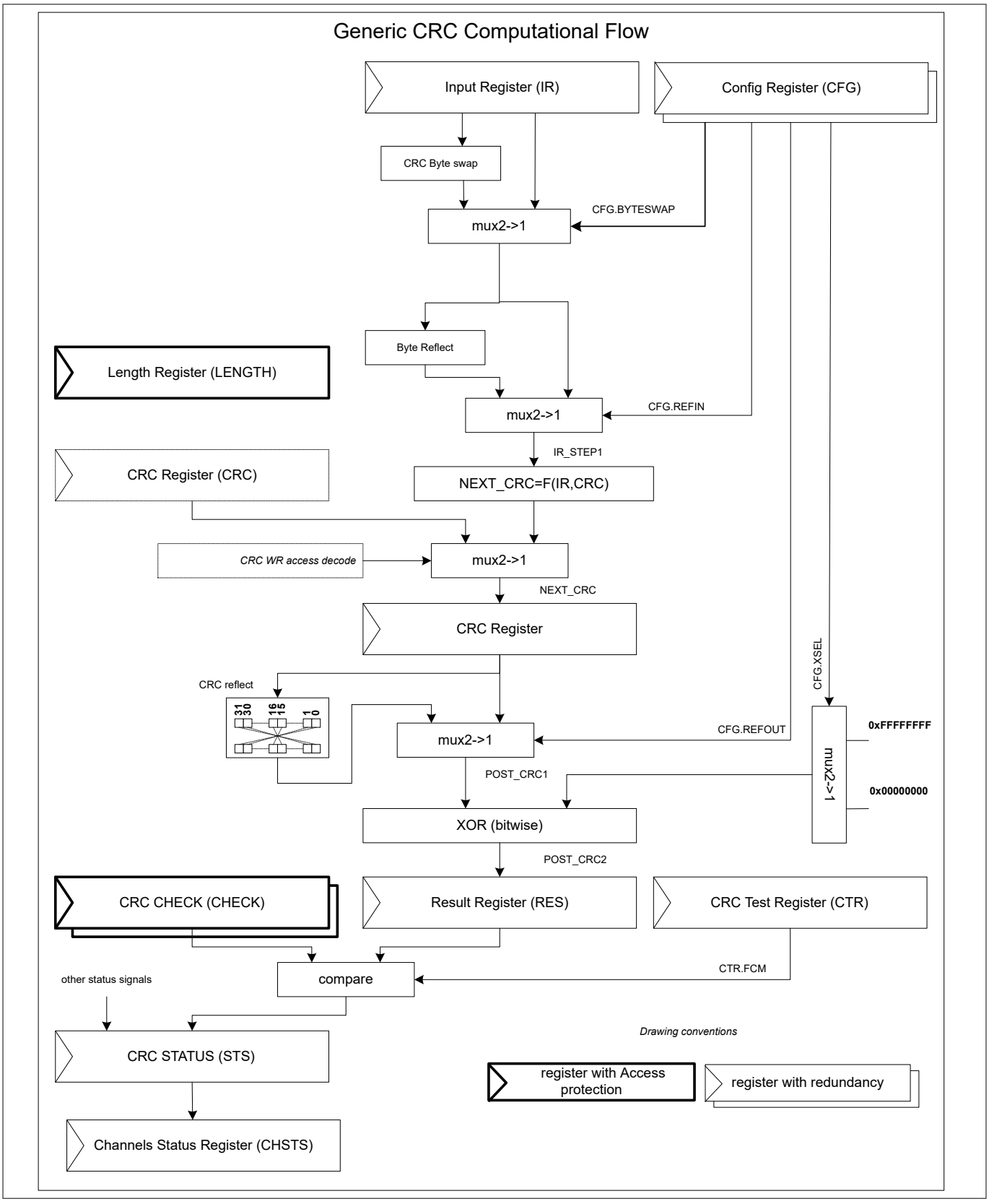


Figure 1008 CRC Computational flow

43 Flexible CRC Engine (FCE)

A checksum algorithm based on CRC polynomial division is characterized by the following properties:

- polynomial degree (example. 32, that represents the highest power of two of the polynomial)
- polynomial (example. $04C11DB7_H$: the 33rd bit is omitted because it is always equal to 1)
- init value: the initial value of the CRC register
- input data reflected: indicates if each byte of the input parallel data is reflected before being used to compute the CRC
- result data reflected: indicates if the final CRC value is reflected or not
- XOR value: indicates if a final XOR operation is done before returning the CRC result
- BYTESWAP: If set, the order of bytes in the input register(when 16- or 32-bit CRC is used) is swapped before the CRC conversion (For endianness conversion).

All the properties are fixed once a polynomial has been chosen. However the FCE provides the capability to control the two reflection steps and the final XOR as depicted in figure above through the CFG register. The reset values are compatible with the implemented algorithm. The final XOR control enables to select either $FFFFFFFF_H$ or 00000000_H to be XORed with the POST_CRC1 (see figure above) value. These two values are those used by the most common CRC polynomials.

As shown in the figure above, when the software writes a value to the IR register, the CRC register is updated in the next clock cycle. The RES register is updated after the output operations (REFOUT, XSEL) still one more cycle later. (that is, there are 2 cycles from a write to the IR register until the result gets updated in the RES register). In case of back to back write accesses to IR register and read access to the RES register, the read access to the RES register is delayed accordingly by hardware so that the software always reads the final computed CRC value from RES register.

It shall be ensured by the programmer that the CFG register is configured before starting the CRC computation, and is not changed until the final result is obtained.

Note that the CRC computation of the FCE is equivalent to a bit-wise shift CRC computation considering that the MSB bit is shifted in first.

***Note**: The reflection steps and final XOR do not modify the properties of the CRC algorithm in terms of error detection, only the CRC final signature is affected.*

The next two figures provides an overview of the control and status features of a CRC channel.

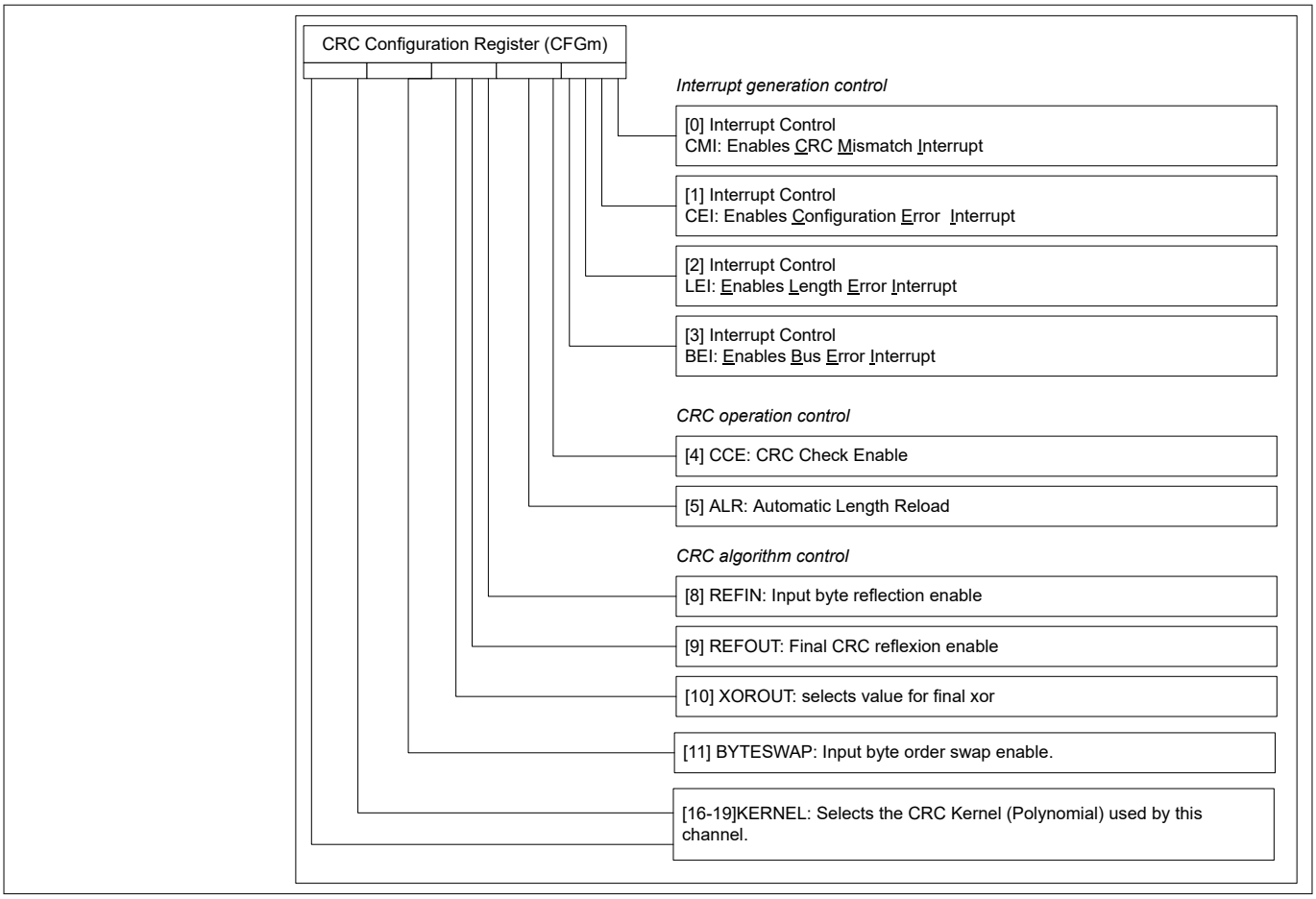


Figure 1009 CRC channel configuration register

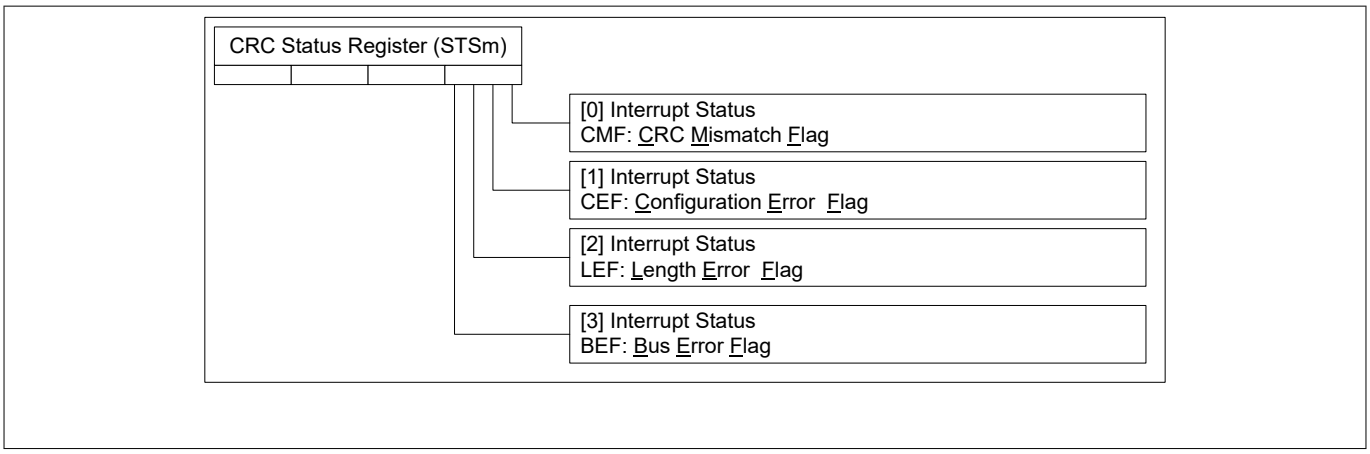


Figure 1010 CRC kernel status register

43.3.1 Initialization

The FCE is enabled by writing 0x0 to the CLC register. After enabling, software can initialize CRC register value with an initial value and CFG register with the CRC kernel selection, input and result reflection, and the final XORed value.

43.3.2 Basic operation

The software must ensure proper initialization of the CRC channels, especially

1. The configuration, including the CRC kernel used in the CFG register
2. Length of the data stream in the LENGTH register
3. The initial CRC value is written through the CRC register

The software can then write as many times as necessary into the IR register according to the length of the message. The resulting signature is stored in the result register RES, which can be read by the software.

Depending on the CRC kernel that is configured to be used by the channel, the following rules apply:

- When using a CRC kernel of degree <N>, only the bits $N-1$ down to 0 are used
- The upper bits are ignored on write
- When reading from a CRC kernel register the unused upper bits are set to 0

Each of the 8 channels can be configured to use any of the implemented CRC algorithm kernels. Multiple channels may be configured to use the same kernel, so enabling concurrent access of the same CRC algorithm by software.

When software writes to an IR register, the registers of the corresponding channel are selected to be used in the CRC computation, based on the bus address of the IR register. The CRC kernel used by this channel is selected based on the CFG.KERNEL bits of the selected channel.

In the following figure the logical representation of the channel and kernel selection based on the bus address of the IR register is shown.

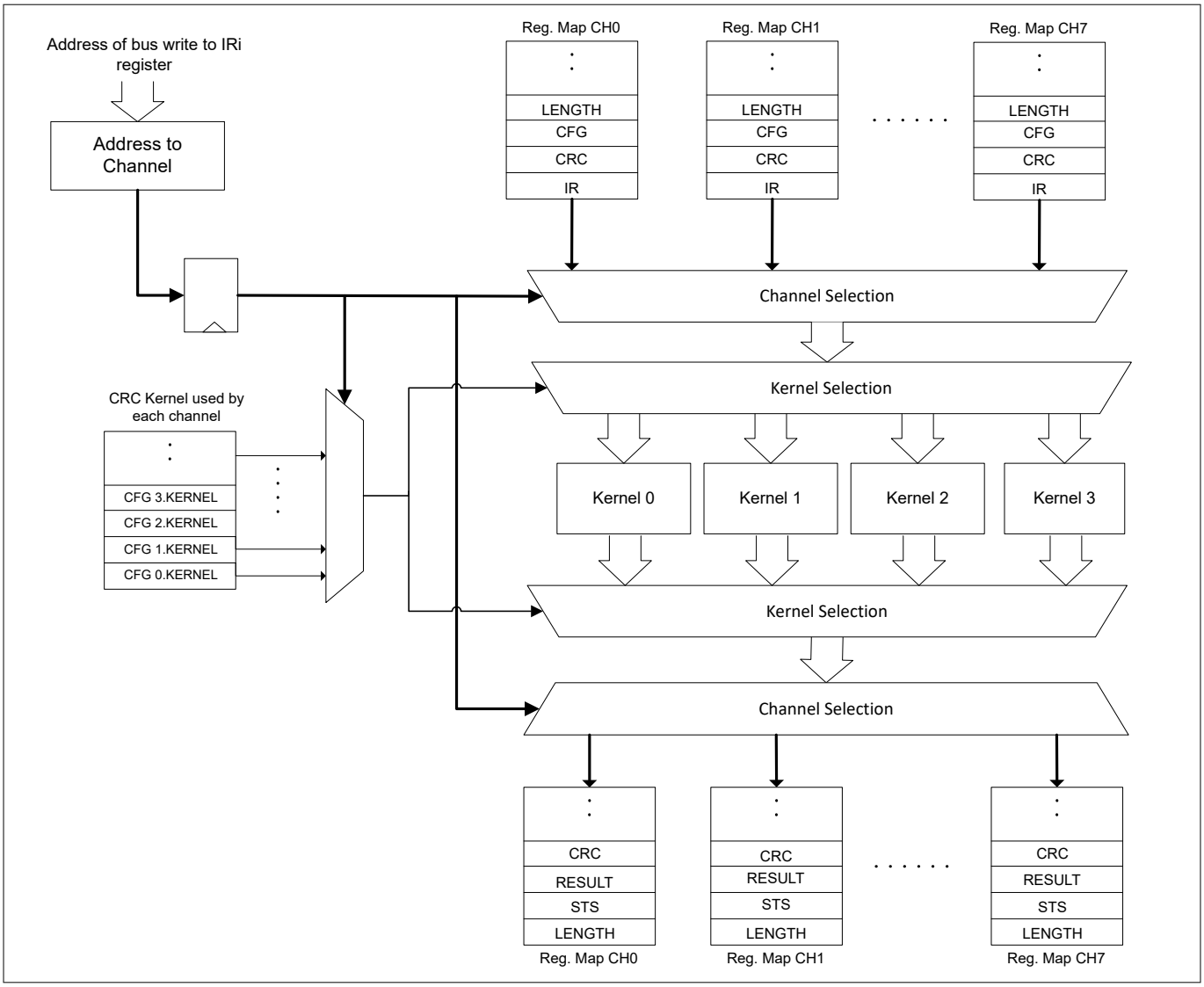


Figure 1011 FCE channel and kernel selection

43.3.3 Automatic signature check

The automatic signature check compares the signature at the end of a message with the expected signature configured in the CHECK register. In case of a mismatch, an event is generated (see related information). This feature is enabled by the CFG.CCE bit-field.

If the software wishes to use this feature, the LENGTH register and CHECK registers of the channel must be configured with:

- The length as number of words of the message
  - The word length is defined by the degree of the polynomial used
- The expected signature (CHECK)
  - The CHECK value takes the final CRC reflection and XOR operation into account

When the CFG.CCE bit-field is set, every time the IR register is written, the LENGTH register is decremented by one until it reaches zero. The hardware monitors the transition of the LENGTH register from 1 to 0 to detect the end of the message and proceeds with the comparison of the result register RES value with the CHECK register value. If the automatic length reload feature is enabled by the CFG.ALR bit-field, the LENGTH register is re-

initialized with the previously configured value. This feature is especially suited when the FCE is used in combination with a DMA engine.

In the case where the automatic length reload feature is not enabled, if LENGTH is already at zero but software still writes to IR (by mistake) then every bit of the LENGTH is set to 1. This value is held until software initializes it again to process a new message. In such cases the STS.LEF (Length Error Flag) is set and an interrupt generated. If the CFG.LEI (Length Error Interrupt) is set, the corresponding written value is completely ignored, and no CRC calculation update is done. In such an error case, any further decrement and reload of the length as well as CRC check is disabled. Software shall clear the LEF error flag for further correct operation.

Usually the CRC signature of a message M0 is computed and appended to M0 to form the message M1 which is transmitted. One interesting property of CRCs is that the CRC signature of M1 shall be zero. This property is particularly useful when automatically checking the signature of data blocks of fixed length with the automatic length reload enabled. LENGTH should be loaded with the length of M1 and CHECK with 0.

Related information

Service request generation on page 7280

43.3.4 Register protection and monitoring methods

Register monitoring applies only to CFG and CHECK registers.

Since CFG and CHECK registers are critical to the CRC operation, some mechanisms to detect and log transient errors are provided. Early detection of transient failures enables to improve the failure detection time and assess the severity of the failure. The monitoring mechanisms are implemented using two redundant instances as presented in the following figure.

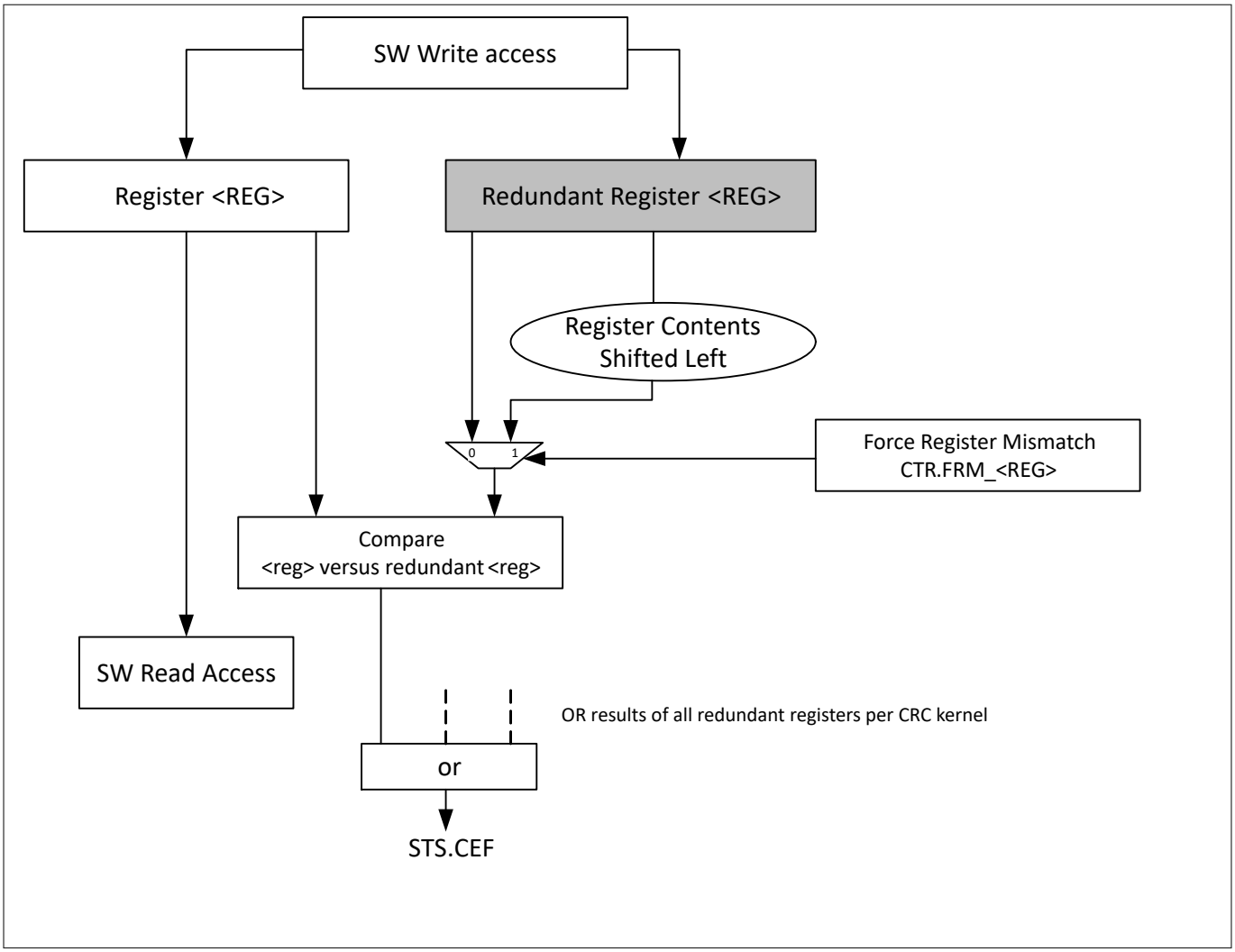


Figure 1012 Register monitoring scheme

Let <REG> designate either CFG or CHECK registers. When a write to <REG> takes place the redundant register is also updated. Redundant registers are not visible to software. Bits of <REG> reserved have no storage and are not used for redundancy. A compare logic continuously compares the two stored values and provides a signal that indicates if the compare is successful or not. The result of all compare blocks are ORed together to provide a single flag information. If a mismatch is detected the STS.CEF (Configuration Error Flag) bit is set. For run-time validation of the compare logic a Force Register Mismatch bit-field (CTR.FRM_<REG>) is provided. When set to 1 by software the contents of the redundant register is shifted left by one bit position (redundant bit 0 position is always replaced by a logical 0 value) and is given to the compare logic instead of the redundant register value. This enables to check if the compare logic is functional. Using a walking bit pattern, the software can completely check the full operation of the compare logic. Software needs to clear the CTR.FRM_<REG> bit to ‘0’ to be able to trigger again a new comparison error interrupt.

Register access protection applies to only LENGTH and CHECK registers.

In order to reduce the probability of a mis-configuration of the CHECK and LENGTH registers (in the case the automatic check is used), the write access to the CHECK and LENGTH registers must follow a procedure:

This procedure is depicted in the following figure:

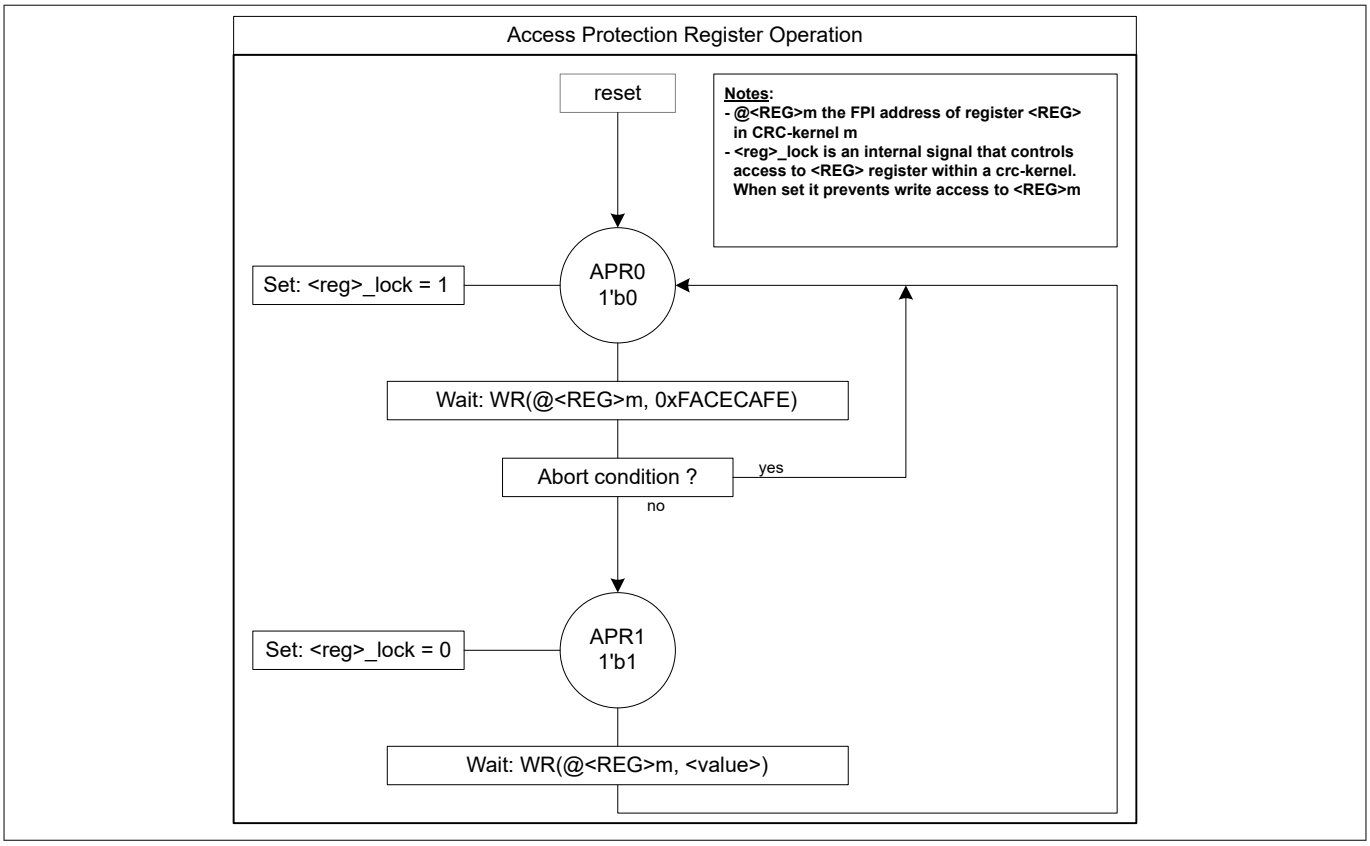


Figure 1013 Access control to CHECK register

Let <REG> designate CHECK or LENGTH registers. Before being able to configure a new <value> into the <REG> register of a CRC kernel, software must first write the 0xFACECAFE value to the <REG> address. The 0xFACECAFE is not written into the <REG> register. The next write access will proceed as a normal bus write access. The write accesses shall use full 32-bit access only. This procedure will then be repeated every time software wants to configure a new <REG> value. If software reads the CHECK register just after writing 0xFACECAFE it returns the current <REG> contents and not 0xFACECAFE. A read access to <REG> has no effect on the protection mechanism.

43.3.5 Power, reset and clock

The FCE is inside the core power domain and therefore no special considerations with regards to power-up or power-down sequences need to be taken. For an explanation about the different power domains, please see the SMM (System Mode Management) chapter.

A power-down mode can be achieved by disabling the module using the CLC register.

The FCE module can be reset through Application Reset or module group resets or through kernel reset request by software. Refer to RST_CTRLA and RST_CTRLB SFRs.

After reset, the complete IP is set to a default configuration. The default configuration for each register field is addressed in the registers overview table.

The module clock can be enabled using the CLC register.

43.3.6 Properties of CRC code

This section describes the Hamming distance for various CRC polynomials supported by FCE.

Hamming distance

The Hamming distance defines the error detection capability of a CRC polynomial. A cyclic code with a Hamming distance of D can detect all D-1 bit errors. The following table shows the dependency of the Hamming distance with the length of the message.

Table 2135 Hamming distance as a function of message length (bits)

Hamming distance	IEEE-802.3 CRC32	CCITT CRC16	J1850 CRC8
15	8 - 10	Information not available	Information not available
14	8 - 10		
13	8 - 10		
12	11 - 12		
11	13 - 21		
10	22 - 34		
9	35 - 57		
8	58 - 91		
7	92 - 171		
6	172 - 268		
5	269 - 2974		
4	2975 - 91606		
3	91607 - 131072		

43.3.7 Service request generation

Each FCE CRC channel provides one internal interrupt source. The interrupt lines from each CRC channel are ORed together to be sent to the interrupt system. The system interrupt is an active high pulse with the duration of one cycle (of the peripheral clock). The FCE interrupt handler can use the status information located within the STS status register of each CRC channel.

Each CRC channel provides the following interrupt sources:

- CRC mismatch interrupt controlled by CFG.CMI bit-field and observable through the status bit-field STS.CMF (CRC Mismatch Flag)
- Configuration error interrupt controlled by CFG.CEI bit-field and observable through the status bit-field STS.CEF (Configuration Error Flag)
- Length error interrupt controlled by CFG.LEI bit-field and observable through the status bit-field STS.LEF (Length Error Flag)
- Bus error interrupt controlled by CFG.BEI bit-field and observable through the status bit-field STS.BEF (Bus Error Flag). This error is triggered by a write to the IR register with write width less than the selected kernel's polynomial width

Interrupt generation rules are as follows:

- A status flag shall be cleared by software by writing a 0 to the corresponding bit position in the STS register
- If a status flag is set and a new hardware condition occurs, no new interrupt is generated by the kernel. A set STS.<FLAG> bit-field masks the generation of a new interrupt from the FCE module. If a software access to clear the interrupt status bit takes place and in the same cycle the hardware wants to set the bit, the hardware condition wins the arbitration

To aid faster interrupt handling in the software, the status flags from each channels are ORed together and provided as individual bits in a Channels Status Register (see figure below). The software can read this single register to learn the channel status, rather than reading the registers of each channel separately.

Since there is only a single interrupt from the FCE, the software has to clear the status flags from all the channels to forward a further FCE interrupt.

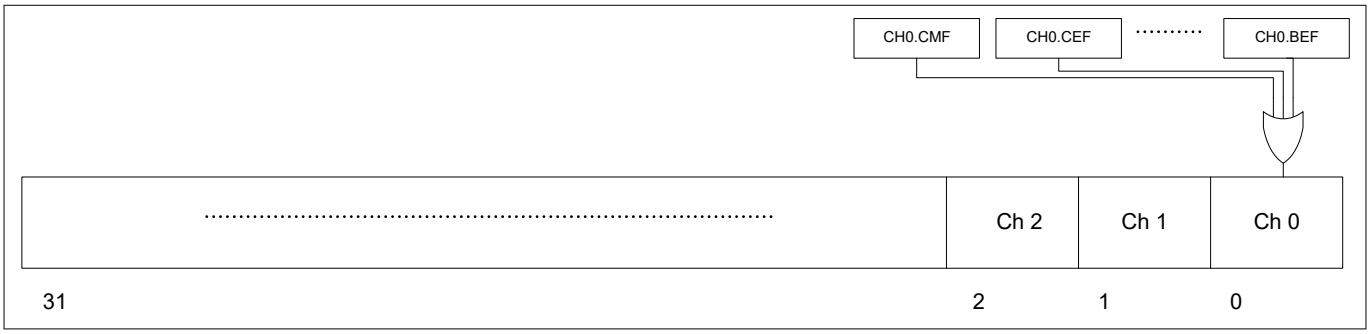


Figure 1014 Channels status register

43.4 Registers

43.4.1 Register overview - access mode glossary

Table 2136 Register overview - access mode glossary

Keyword	Description
E	Access protection using PROT register PROTE.
SE	Access protection using PROT register PROTSE.
APU-Py (y=0-2)	Protection group consisting of registers ACCENy_WRA, ACCENy_WRB, ACCENy_RDA, ACCENy_RDB, ACCENy_VM, ACCENy_PRS.
Py	Access protection using *APU-Py* registers.
PCHz (z=0-7)	Access protection using one APU from *APU-P[y] (y=0-2)*. The protection source to be used is selected by bit-field: z=0-7: RESALLOC_CH.CHi
U	No access restrictions.
BE	Always returns a Bus Error.
PROT	Access restrictions as defined in the PROT register access rules.

43.4.2 Registers overview - FCE (ascending offset address)

Table 2137 Registers overview - FCE (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CLC	Clock control register	000_H	P0	P0, SV, E	Application Reset	7283
ID	Module identification register	008_H	U	BE	PowerOn Reset	7283

(table continues...)

Table 2137 (continued) Registers overview - FCE (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
RST_CTRLA	Reset control register A	$00C_H$	P0	P0, SV, E	Application Reset	7284
RST_CTRLB	Reset control register B	010_H	P0	P0, SV, E	Application Reset	7284
RST_STAT	Reset status register	014_H	P0	BE	Application Reset	7285
PROTE	PROT register endinit	020_H	U	SV, PROT	Application Reset	7286
PROTSE	PROT register safe endinit	024_H	U	SV, PROT	Application Reset	7287
RESALLOC_CH	Resource allocation register for FCE channels	028_H	U	SE, SV	Application Reset	7289
ACCENy_WRA	Write access enable y register A	044_H+y* 20_H	U	SE, SV	Application Reset	7290
ACCENy_WRB	Write access enable y register B	048_H+y* 20_H	U	SE, SV	Application Reset	7290
ACCENy_RDA	Read access enable y register A	$04C_H$+y* 20_H	U	SE, SV	Application Reset	7291
ACCENy_RDB	Read access enable y register B	050_H+y* 20_H	U	SE, SV	Application Reset	7291
ACCENy_VM	VM access enable y register	054_H+y* 20_H	U	SE, SV	Application Reset	7292
ACCENy_PRS	PRS access enable y register	058_H+y* 20_H	U	SE, SV	Application Reset	7293
CHSTS	Channels status register	$0C0_H$	U	BE	Kernel Reset	7293
IRi	Input register i	100_H+i*2 0_H	PCHi	PCHi	Kernel Reset	7294
RESi	CRC result register i	104_H+i*2 0_H	PCHi	BE	Kernel Reset	7295
CFGi	CRC configuration register i	108_H+i*2 0_H	PCHi	E, SV, PCHi	Kernel Reset	7295
STSi	CRC status register i	$10C_H$+i*2 0_H	PCHi	PCHi	Kernel Reset	7297
LENGTHi	CRC length register i	110_H+i*2 0_H	PCHi	PCHi	Kernel Reset	7298
CHECKi	CRC check register i	114_H+i*2 0_H	PCHi	PCHi	Kernel Reset	7298
CRCi	CRC register i	118_H+i*2 0_H	PCHi	PCHi	Kernel Reset	7299
CTRi	CRC test register i	$11C_H$+i*2 0_H	PCHi	PCHi	Kernel Reset	7299

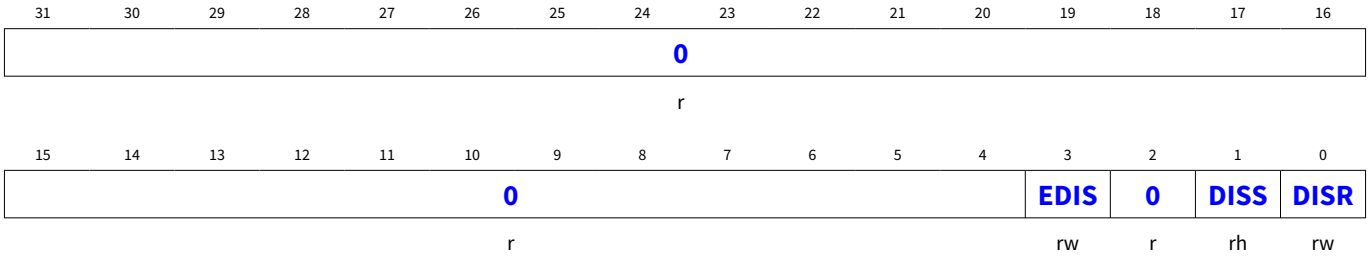
43.4.3 Clock control register

The Clock Control Register CLC allows the programmer to adapt the functionality and power consumption of the module to the requirements of the application.

Register CLC controls the module clock signal and the reactivity to the sleep signal.

CLC Offset address: 000_H

Clock control register Application Reset value: $0000\ 0003_H$

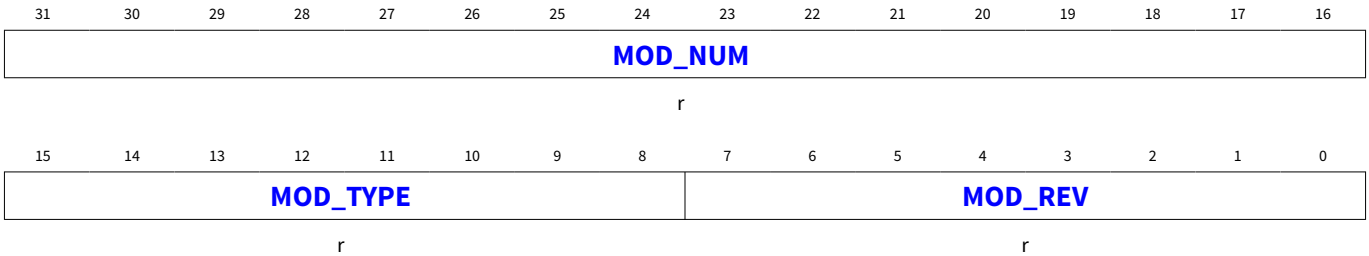


Field	Bits	Type	Description
DISR	0	rw	**Module Disable Request Bit** Used for enable or disable control of the module. Also the analog section is disabled by clearing ANONS. 0_B On request: enable the module clock 1_B Off request: stop the module clock
DISS	1	rh	**Module Disable Status Bit** 0_B Module clock is enabled 1_B Off: module is not clocked
EDIS	3	rw	**Sleep Mode Enable Control** Used to control the module's reaction to sleep mode. 0_B Sleep mode request is enabled and functional 1_B Module disregards the sleep mode control signal
0	2, 31:4	r	**Reserved** Read as 0; should be written with 0.

43.4.4 Module identification register

ID Offset address: 008_H

Module identification register PowerOn Reset value: $00CA\ C005_H$



Field	Bits	Type	Description
MOD_REV	7:0	r	**Module Revision** Indicates the revision number of the implementation.
MOD_TYPE	15:8	r	**Module Type** This internal marker is fixed to $C0_H$.
MOD_NUM	31:16	r	**Module Number** Indicates the module identification number.

43.4.5 Reset control register A

RST_CTRLA Offset address: $00C_H$

Reset control register A Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

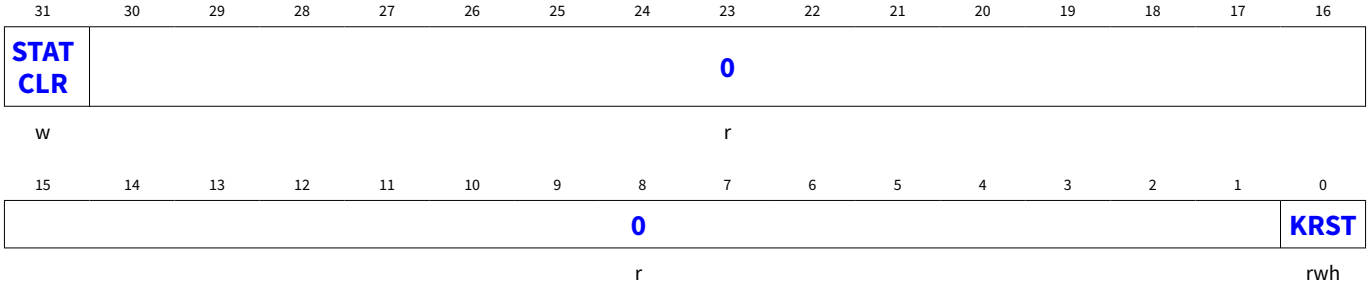
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				GRSTEN3	GRSTEN2	GRSTEN1	GRSTEN0	0							KRST
r				rw	rw	rw	rw	r							rwh

Field	Bits	Type	Description
KRST	0	rwh	**Kernel Reset** Request a kernel reset. The requested reset is executed if CTRLA.KRST=1 and CTRLB.KRST=1. KRST is cleared after the kernel reset was executed. 0_B No action 1_B A kernel reset was requested
GRSTENx (x=0-3)	x+8	rw	**Enable for Global Module Reset Group x** 0_B Global module reset group x does not have any effect 1_B Global module reset group x results in a kernel reset
0	7:1, 31:12	r	**Reserved** Read as 0; should be written with 0.

43.4.6 Reset control register B

RST_CTRLB Offset address: 010_H

Reset control register B Application Reset value: $0000\ 0000_H$



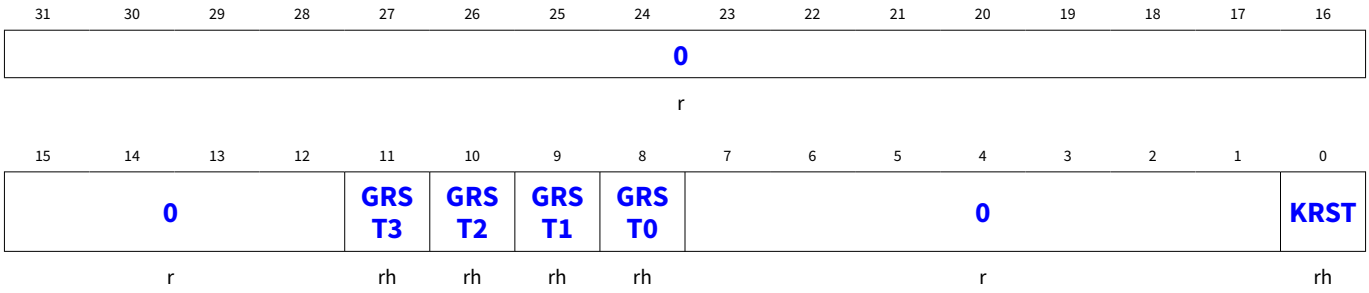
Field	Bits	Type	Description
KRST	0	rwh	**Kernel Reset** Request a kernel reset. The requested reset is executed if CTRLA.KRST=1 and CTRLB.KRST=1. KRST is cleared after the kernel reset was executed. 0_B No action 1_B A kernel reset was requested
STATCLR	31	w	**Kernel Reset Status Clear** Clears all status bits in RST_STAT when 1 is written. Read will return 0. 0_B No action 1_B Write with 1 clears bits STAT.GRSTx and bit STAT.KRST .
0	30:1	r	**Reserved** Read as 0; should be written with 0.

43.4.7 Reset status register

The reset status register contains the status bits for kernel reset (KRST) and the global module reset groups (GRSTx).

RST_STAT Offset address: 014_H

Reset status register Application Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
KRST	0	rh	**Kernel Reset Status** Indicates an executed kernel reset. RST_STAT.KRST is set after the execution of a kernel reset in the same clock cycle in which the reset bits are cleared. Clear KRST by setting bit STATCLR in register RST_CTRLB.

(table continues...)

(continued)

Field	Bits	Type	Description
GRSTx (x=0-3)	x+8	rh	**Status for Global Module Reset Group x** 0_B Reset was not triggered by Global Reset Group x 1_B Reset was triggered by Global Reset Group x
0	7:1, 31:12	r	**Reserved** Read as 0; should be written with 0.

43.4.8 PROT register endinit

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner and in some PROT states, the secure master to update the PROT state as described in the PROT mechanism.

The PROTE register controls lock or unlock of the local Endinit (E) protected control registers.

PROTE Offset address: 020_H

PROT register endinit Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OWEN	ODEF	TAGID						PRSEN	PRS			VMEN	VM		
w	rw	rw						rw	rw			rw	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												SWEN	STATE		
r												w	rwh		

Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
VM	18:16	rw	**Virtual machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.

(table continues...)

(continued)

Field	Bits	Type	Description
VMEN	19	rw	**Virtual machine definition enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.
PRSEN	23	rw	**Protection set definition enable for PROT owner** 0_B **Disable**: PRS is not part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG-ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; should be written with 0.

Table 2138 Access mode restrictions of PROTE sorted by descending priority

Mode name	Access mode		Description
write 1 to .**SWEN**	rwh	STATE	
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

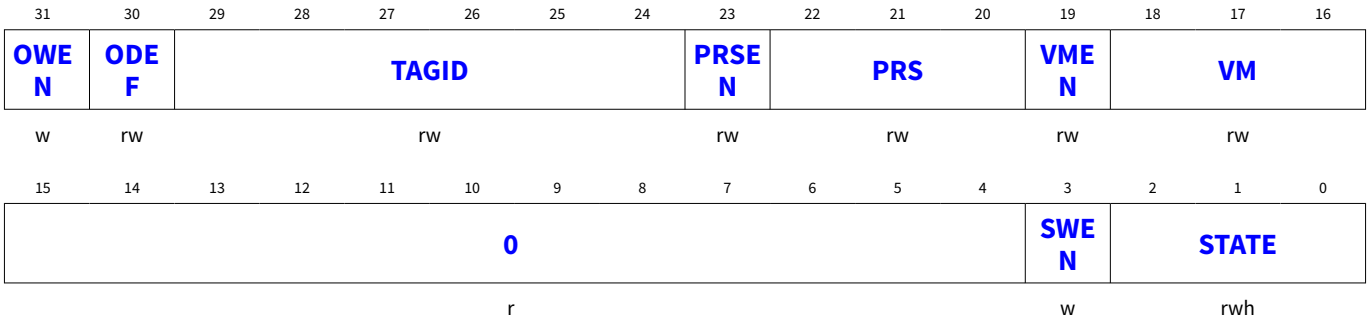
43.4.9 PROT register safe endinit

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner and in some PROT states, the secure master to update the PROT state as described in the PROT mechanism.

The PROTSE register controls lock or unlock of the local Safe Endinit (SE) protected control registers.

PROTSE Offset address: 024_H

PROT register safe endinit Application Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
VM	18:16	rw	**Virtual machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual machine definition enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.
PRSEN	23	rw	**Protection set definition enable for PROT owner** 0_B **Disable**: PRS is not part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG-ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner

(table continues...)

(continued)

Field	Bits	Type	Description
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; should be written with 0.

Table 2139 Access mode restrictions of PROTSE sorted by descending priority

Mode name	Access mode		Description
write 1 to .**SWEN**	rwh	STATE	
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

43.4.10 Resource allocation register for FCE channels

RESALLOC_CH Offset address: 028_H

Resource allocation register for FCE channels Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		CH7		0		CH6		0		CH5		0		CH4	
r		rw		r		rw		r		rw		r		rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		CH3		0		CH2		0		CH1		0		CH0	
r		rw		r		rw		r		rw		r		rw	

Field	Bits	Type	Description
CHi (i=0-7)	4*i+1:4*i	rw	**APU Assignment for FCE channel i** Used to select one of the APU-Py (y=0-2) for the corresponding FCE channel i SFRs 00_B Protected by APU-P0 01_B Protected by APU-P1 10_B Protected by APU-P2 **others,** Reserved - Protected by APU-P0

(table continues...)

(continued)

Field	Bits	Type	Description
0	31:30, 27:26, 23:22, 19:18, 15:14, 11:10, 7:6, 3:2	r	**Reserved** Read as 0; should be written with 0.

43.4.11 Write access enable y register A

Protection for FCE Kernel registers and Channel specific registers.

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

ACCENy_WRA (y=0-2) Offset address: 044_H+y*20_H

Write access enable y register A Application Reset value: $1000\ 0003_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access

43.4.12 Write access enable y register B

Protection for FCE Kernel registers and Channel specific registers.

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

ACCENy_WRB (y=0-2) Offset address: 048_H+y*20_H

Write access enable y register B Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for write access 1_B Enabled for write access
0	31:8	r	**Reserved** Read as 0; should be written with 0.

43.4.13 Read access enable y register A

Protection for FCE Kernel registers and Channel specific registers.

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

ACCENy_RDA (y=0-2) Offset address: $04C_H+y*20_H$

Read access enable y register A Application Reset value: FFFF $FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access

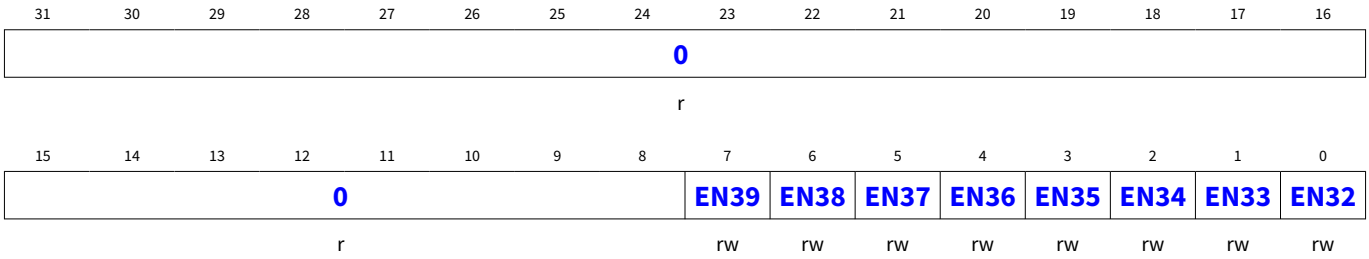
43.4.14 Read access enable y register B

Protection for FCE Kernel registers and Channel specific registers.

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

ACCENy_RDB (y=0-2) Offset address: 050_H+y*20_H

Read access enable y register B Application Reset value: 0000 $00FF_H$



Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q. 0_B Disabled for read access 1_B Enabled for read access
0	31:8	r	**Reserved** Read as 0; should be written with 0.

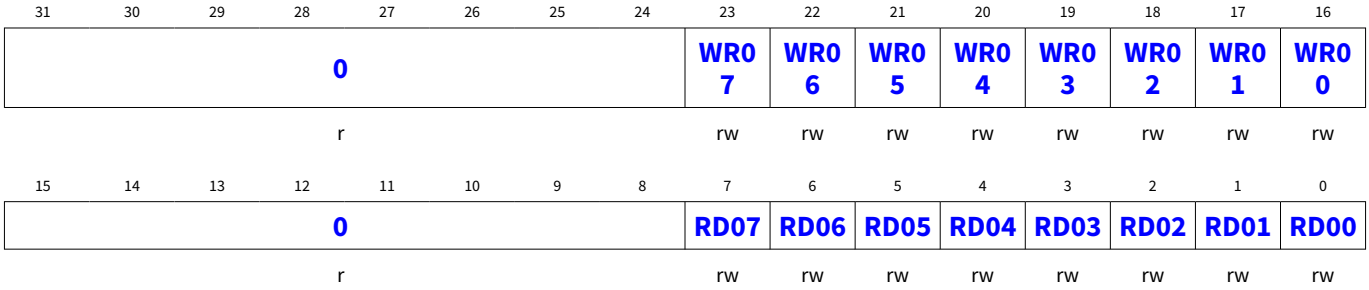
43.4.15 VM access enable y register

Protection for FCE Kernel registers and Channel specific registers.

This register defines which virtual machine encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each VMx.

ACCENy_VM (y=0-2) Offset address: 054_H+y*20_H

VM access enable y register Application Reset value: $00FF\ 00FF_H$



Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for VM ID q** This bit enables read access to the access protected region for transactions with the VM ID q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for VM ID q** This bit enables write access to the access protected region for transactions with the VM ID q. 0_B Disabled for write access 1_B Enabled for write access

(table continues...)

(continued)

Field	Bits	Type	Description
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

43.4.16 PRS access enable y register

Protection for FCE Kernel registers and Channel specific registers.

This register defines which protection register set encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each PRSx.

ACCENy_PRS (y=0-2) Offset address: 058_H+y*20_H

PRS access enable y register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

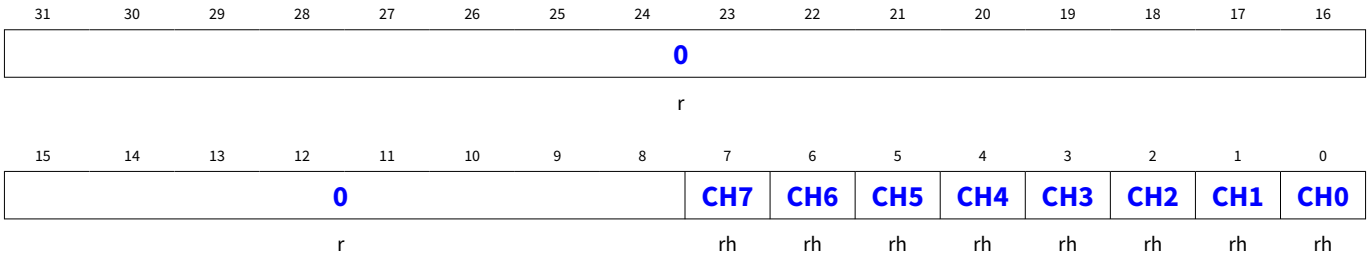
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for PRS q** This bit enables read access to the access protected region for transactions with the PRS q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for PRS q** This bit enables write access to the access protected region for transactions with the PRS q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

43.4.17 Channels status register

CHSTS Offset address: $0C0_H$

Channels status register Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
CH0	0	rh	**Channel0 Status** This bit is the result of an OR operation of the various status bits of channel 0 (see STS register).
CH1	1	rh	**Channel1 Status** This bit is the result of an OR operation of the various status bits of channel 1 (see STS register).
CH2	2	rh	**Channel2 Status** This bit is the result of an OR operation of the various status bits of channel 2 (see STS register).
CH3	3	rh	**Channel3 Status** This bit is the result of an OR operation of the various status bits of channel 3 (see STS register).
CH4	4	rh	**Channel4 Status** This bit is the result of an OR operation of the various status bits of channel 4 (see STS register).
CH5	5	rh	**Channel5 Status** This bit is the result of an OR operation of the various status bits of channel 5 (see STS register).
CH6	6	rh	**Channel6 Status** This bit is the result of an OR operation of the various status bits of channel 6 (see STS register).
CH7	7	rh	**Channel7 Status** This bit is the result of an OR operation of the various status bits of channel 7 (see STS register).
0	31:8	r	**Reserved** Read as 0. Should be written with 0.

43.4.18 Input register i

IRi (i=0-7)

Input register i

Offset address: 100_H+i*20_H

Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
IR															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IR															
rw															

Field	Bits	Type	Description
IR	31:0	rw	**Input Register** This bit-field holds the input data to be computed. In case the channel is configured to use 16-bit or 8-bit CRC, only the LSB 16 or 8-bits will be used as input.

43.4.19 CRC result register i

RESi (i=0-7) Offset address: 104_H+i*20_H

CRC result register i Kernel Reset value: FFFF FFFF$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES															
rh															

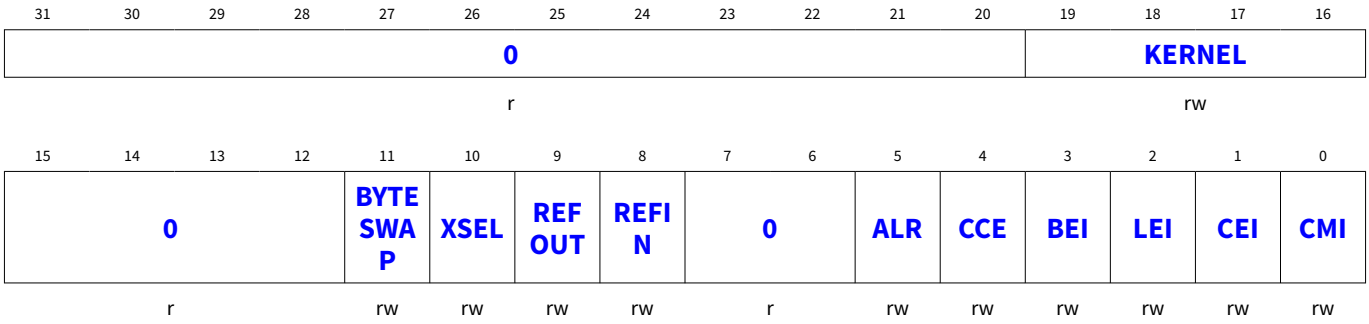
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES															
rh															

Field	Bits	Type	Description
RES	31:0	rh	**Result Register** Returns the final CRC value including CRC reflection and final XOR according to the CFG register configuration. If the channel is configured to use 16-bit or 8-bit CRC, the MSB 16 or 24 bits respectively shall be read as 0.

43.4.20 CRC configuration register i

CFGi (i=0-7) Offset address: 108_H+i*20_H

CRC configuration register i Kernel Reset value: 0000 0700$_H$



Field	Bits	Type	Description
CMI	0	rw	**CRC Mismatch Interrupt** 0_B CRC Mismatch Interrupt is disabled 1_B CRC Mismatch Interrupt is enabled
CEI	1	rw	**Configuration Error Interrupt** When enabled, a Configuration Error Interrupt is generated whenever a mismatch is detected in the CFG and CHECK redundant registers. 0_B Configuration Error Interrupt is disabled 1_B Configuration Error Interrupt is enabled
LEI	2	rw	**Length Error Interrupt** When enabled, a Length Error Interrupt is generated if software writes to IR register with LENGTH equal to 0 and CFG.CCE is set to 1. 0_B Length Error Interrupt is disabled 1_B Length Error Interrupt is enabled
BEI	3	rw	**Bus Error Interrupt** When enabled, an interrupt (BEF) is generated if a bus write transaction with an access width smaller than the kernel width is issued to the input register. In this case, the corresponding value written to the IR is discarded and no CRC computation takes place. 0_B Bus Error Interrupt is disabled 1_B Bus Error Interrupt is enabled
CCE	4	rw	**CRC Check Comparison** 0_B CRC check comparison at the end of a message is disabled 1_B CRC check comparison at the end of a message is enabled. In this case, if length error is set (STS.LEF = 1) and IR is written to, then length is set to all ones, independent of the previous length value. The ALR bit is also ignored in such a condition.
ALR	5	rw	**Automatic Length Reload** 0_B Disables automatic reload of the LENGTH field. 1_B Enables automatic reload of the LENGTH field at the end of a message.
REFIN	8	rw	**IR Byte Wise Reflection** 0_B IR Byte Wise Reflection is disabled 1_B IR Byte Wise Reflection is enabled

(table continues...)

(continued)

Field	Bits	Type	Description
REFOUT	9	rw	**CRC Bit Wise Reflection** The alignment of the reflection is the same as the kernel polynomial width. For example: 32-bit kernel has bitwise reflection by 32-bits. 0_B CRC-bit wise is disabled 1_B CRC-bit wise is enabled
XSEL	10	rw	**Selects the value to be XORed with the final CRC** 0_B 0x00000000 1_B 0xFFFFFFFF
BYTESWAP	11	rw	**Swaps the order of the bytes in the IR input register.** 0_B The order of bytes in IR register are not swapped before CRC computation. 1_B The order of bytes in the IR register are swapped before CRC computation. Big-endian input is converted to Little-endian and vice versa. (When 8-bit CRC is chosen, this has no effect).
KERNEL	19:16	rw	**Selects the CRC Kernel (Polynomial Engine) used by this channel.** Other possible values are reserved for additional kernels that may be added in the future. If these values are used, then KERNEL3 is used by default. 0_H Kernel 0 is used. 1_H Kernel 1 is used. 2_H Kernel 2 is used. 3_H Kernel 3 is used.
0	7:6, 15:12, 31:20	r	**Reserved** Read as 0; should be written with 0.

43.4.21 CRC status register i

STSi (i=0-7) Offset address: $10C_H+i*20_H$

CRC status register i Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												BEF	LEF	CEF	CMF
r												rwh	rwh	rwh	rwh

Field	Bits	Type	Description
CMF	0	rwh	**CRC Mismatch Flag** This bit is set per hardware only. To clear this bit, software must write a 0 to this bit-field location. Writing 1 to this bit has no effect.

(table continues...)

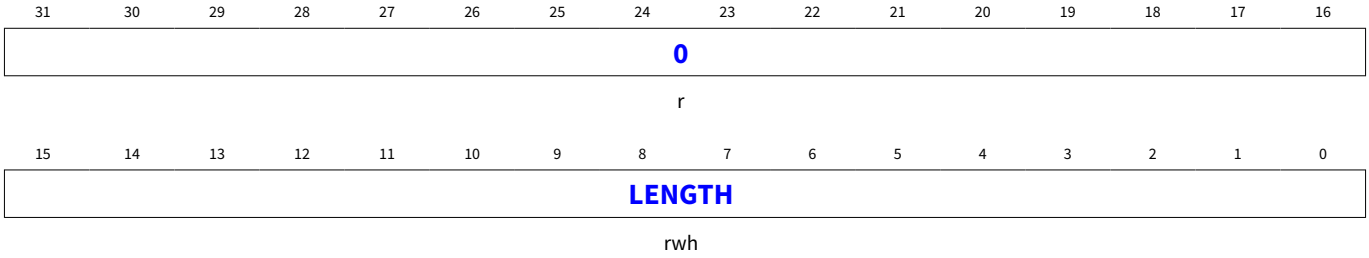
(continued)

Field	Bits	Type	Description
CEF	1	rwh	**Configuration Error Flag** This bit is set per hardware only. To clear this bit, software must write a 0 to this bit-field location. Writing a 1 has no effect.
LEF	2	rwh	**Length Error Flag** This bit is set per hardware only. To clear this bit, software must write a 0 to this bit-field location. Writing 1 has no effect.
BEF	3	rwh	**Bus Error Flag** This bit is set per hardware only. To clear this bit, software must write a 0 to this bit-field location. Writing 1 has no effect.
0	31:4	r	**Reserved** Read as 0; should be written with 0.

43.4.22 CRC length register i

LENGTHi (i=0-7) Offset address: 110_H+i*20_H

CRC length register i Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
LENGTH	15:0	rwh	**Message Length Register** Number of words (bit width of each word in terms of KERNEL polynomial width) building the message over which the CRC checksum is calculated. This bit-field is modified by the hardware: every write to the IR register decrements the value of the LENGTH bit-field. If the CFG.ALR field is set to 1, the LENGTH field shall be reloaded with its configuration value at the end of the cycle where LENGTH reaches 0.
0	31:16	r	**Reserved** Read as 0; should be written with 0.

43.4.23 CRC check register i

CHECKi (i=0-7) Offset address: 114_H+i*20_H

CRC check register i Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CHECK															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CHECK															
rw															

Field	Bits	Type	Description
CHECK	31:0	rw	**CHECK Register** Expected CRC value to be checked by the hardware upon detection of a 1 to 0 transition of the LENGTH register. The comparison is enabled by the CFG.CCE bit-field

43.4.24 CRC register i

CRCi (i=0-7) Offset address: 118_H+i*20_H

CRC register i Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CRC															
rwh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CRC															
rwh															

Field	Bits	Type	Description
CRC	31:0	rwh	**CRC Register** This register enables to directly access the internal CRC register

43.4.25 CRC test register i

CTRi (i=0-7) Offset address: $11C_H+i*20_H$

CRC test register i Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0													FRM_CHECK	FRM_CFG	FCM
r													rw	rw	rw

Field	Bits	Type	Description
FCM	0	rw	**Force CRC Mismatch** Forces the CRC compare logic to issue an error regardless of the CHECK and CRC values. The hardware detects a 0 to 1 transition of this bit-field and triggers a CRC Mismatch interrupt
FRM_CFG	1	rw	**Force CFG Register Mismatch** This field is used to control the error injection mechanism used to check the compare logic of the redundant CFG registers. This is a one shot operation. When the hardware detects a 0 to 1 transition of this bit-field it triggers a Configuration Mismatch interrupt (if enabled by the corresponding CFGm register).
FRM_CHECK	2	rw	**Force Check Register Mismatch** This field is used to control the error injection mechanism used to check the compare logic of the redundant CHECK registers. This is a one shot operation. The hardware detects a 0 to 1 transition of this bit-field and triggers a Check Register Mismatch interrupt (if enabled by the corresponding CFGm register).
0	31:3	r	**Reserved** Read as 0; should be written with 0.

43.5 Debug information

This section is not applicable for the module being described.

43.6 References

1. “32-Bit Cyclic Redundancy Codes for Internet Applications” by Philip Koopman, Carnegie Mellon University, 2002

43.7 FCE revision history

Initial release of the chapter.

43.8 TC4Dx FCE information

43.8.1 TC4Dx FCE configuration

There are no deviations from the generic specification.

43.8.2 TC4Dx FCE features

There are no deviations from the generic specification.

43.8.3 TC4Dx FCE functional description

Slave interface

In TC4Dx device, the FCE slave interface is connected to ComPB bus interface.

Functional limitations

Software must ensure that FCE kernel reset (either via KRST or module group reset) is not initiated while a concurrent write access to IR register and read access to RES register is performed. Alternatively, the software must ensure there is always a 2 fSPB clock cycles wait between the last write to IR register and read to RES register.

43.8.4 TC4Dx FCE registers

There are no deviations from the generic specification.

43.8.4.1 Register address space - FCE

Table 2140 Registers address space - FCE

Module	Base address	End address	Note
FCE	$F4402E00_H$	$F4402FFF_H$	FPI slave interface

43.8.4.2 Register overview - access mode glossary

Table 2141 Register overview - access mode glossary

Keyword	Description
E	Access protection using PROT register FCE_PROTE .
SE	Access protection using PROT register FCE_PROTSE .
APU-Py (y=0-2)	Protection group consisting of registers FCE_ACCENy_WRA , FCE_ACCENy_WRB , FCE_ACCENy_RDA , FCE_ACCENy_RDB , FCE_ACCENy_VM , FCE_ACCENy_PRS .
Py	Access protection using *APU-Py* registers.
PCHz (z=0-7)	Access protection using one APU from *APU-P[y] (y=0-2)*. The protection source to be used is selected by bit-field: z=0-7: RESALLOC_CH.CHi
U	No access restrictions.
BE	Always returns a Bus Error.
PROT	Access restrictions as defined in the PROT register access rules.

43.8.4.3 Registers overview - FCE (ascending offset address)

Table 2142 Registers overview - FCE (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
FCE_CLC	Clock control register	000_H	P0	P0, SV, E	Application Reset	7283
FCE_ID	Module identification register	008_H	U	BE	PowerOn Reset	7283
FCE_RST_CTRLA	Reset control register A	$00C_H$	P0	P0, SV, E	Application Reset	7284
FCE_RST_CTRLB	Reset control register B	010_H	P0	P0, SV, E	Application Reset	7284
FCE_RST_STAT	Reset status register	014_H	P0	BE	Application Reset	7285
FCE_PROTE	PROT register endinit	020_H	U	SV, PROT	Application Reset	7286

(table continues...)

Table 2142 (continued) Registers overview - FCE (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
FCE_PROTSE	PROT register safe endinit	024_H	U	SV, PROT	Application Reset	7287
FCE_RESALLOC_CH	Resource allocation register for FCE channels	028_H	U	SE, SV	Application Reset	7289
FCE_ACCENy_WRA (y=0-2)	Write access enable y register A	044_H+y* 20_H	U	SE, SV	Application Reset	7290
FCE_ACCENy_WRB (y=0-2)	Write access enable y register B	048_H+y* 20_H	U	SE, SV	Application Reset	7290
FCE_ACCENy_RDA (y=0-2)	Read access enable y register A	$04C_H$+y* 20_H	U	SE, SV	Application Reset	7291
FCE_ACCENy_RDB (y=0-2)	Read access enable y register B	050_H+y* 20_H	U	SE, SV	Application Reset	7291
FCE_ACCENy_VM (y=0-2)	VM access enable y register	054_H+y* 20_H	U	SE, SV	Application Reset	7292
FCE_ACCENy_PRS (y=0-2)	PRS access enable y register	058_H+y* 20_H	U	SE, SV	Application Reset	7293
FCE_CHSTS	Channels status register	$0C0_H$	U	BE	Kernel Reset	7293
FCE_IRi (i=0-7)	Input register i	100_H+i*2 0_H	PCHi	PCHi	Kernel Reset	7294
FCE_RESi (i=0-7)	CRC result register i	104_H+i*2 0_H	PCHi	BE	Kernel Reset	7295
FCE_CFGi (i=0-7)	CRC configuration register i	108_H+i*2 0_H	PCHi	E, SV, PCHi	Kernel Reset	7295
FCE_STSi (i=0-7)	CRC status register i	$10C_H$+i*2 0_H	PCHi	PCHi	Kernel Reset	7297
FCE_LENGTHi (i=0-7)	CRC length register i	110_H+i*2 0_H	PCHi	PCHi	Kernel Reset	7298
FCE_CHECKi (i=0-7)	CRC check register i	114_H+i*2 0_H	PCHi	PCHi	Kernel Reset	7298
FCE_CRCi (i=0-7)	CRC register i	118_H+i*2 0_H	PCHi	PCHi	Kernel Reset	7299
FCE_CTRi (i=0-7)	CRC test register i	$11C_H$+i*2 0_H	PCHi	PCHi	Kernel Reset	7299

43.8.4.4 Device specific registers

There are no device specific register changes.

43.8.5 TC4Dx FCE connectivity

The table below lists interfaces to and from this functional block to other blocks in the device.

Table 2143 List of FCE interface signals

Interface signals	I/O	Description
CLOCK_FCE_fSPB	In	SPB clock input
FCE_IR_SRC_FCE	Out	Interrupt request
FPI_FCE_SIF	In	FCE FPI Slave Interface

43.8.6 TC4Dx FCE revision history

Initial release of the chapter.

44 Enhanced Generic Timer Module (eGTM)

eGTM is an enhanced version of the GTM, a universal timer architecture provided by Bosch AE.

The eGTM is capable of generating complex PWM/Pulses as for example motor control or power conversion. The module can be used for input signal characterization and for many other use cases.

This specification is a generic specification. Registers for non-existing modules (as communication modules are differing in number) are not functional and they will be simply not connected.

***Note**: TC4Dx is an exception as registers for non-existing modules are removed.*

44.1 Feature list

- GTM v4.1, designed by Bosch AE, running at the maximum CPU frequency
- Trigger outputs to: ADC, SENT, PSI5(S), PMS, ASCLIN, QSPI, CAN(XL), SCU, and MSC
- Trigger inputs from: CAN(XL), Ethernet, SCU, PMS, GPT12, ADC, and FlexRay™
- Filtered outputs can be provided from eGTM Timer Input Modules (TIMs) to GPT12 modules (No GPT12 on TC4Dx, TC4Zx)
- Debug interface
- ADC boundary flags are provided as input to TIMs and to DTMs (auxiliary interface)
- High Resolution Support on ATOMs (No HRPWM on TC4Dx, TC4Zx)

44.2 Functional overview

The eGTM module is capable of driving different kinds of pulse-width-modulation (PWM) with and without dead-time. eGTM can be considered a scaled-down version of GTM Classic with some additional features to support electric drive application requirements (for example "High resolution PWM" support) and reduced access latency.

It is able to capture signals from ports. It is possible to get signals from other function blocks of the device and to provide control signals to other function blocks of the device.

***Note**: eGTM-IP is the GTM-IP block from Bosch AE, which is chosen in the corresponding configuration for electrical drive application.*

***Note**: Inside the block diagram all resources which are possible, are part of the picture. Please check in the device specific section, if the resources marked on the right side are existing in the device.*

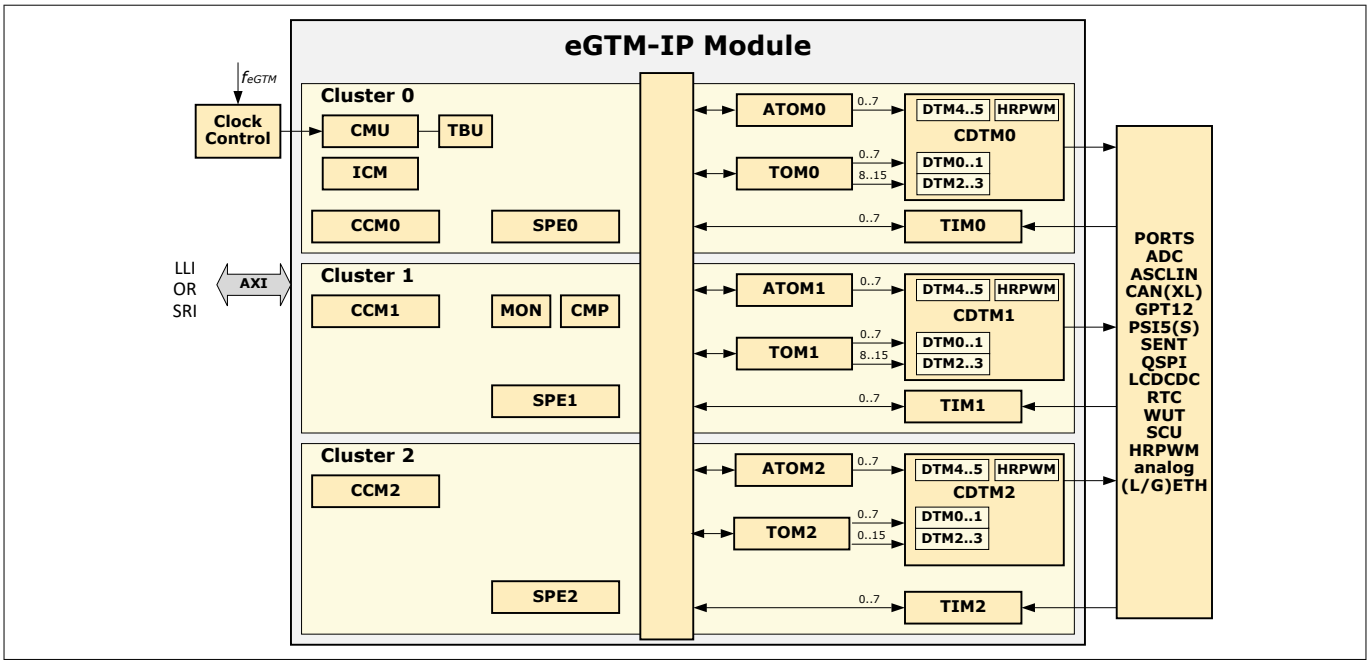


Figure 1015 eGTM block diagram

***Note**: This figure is a general figure. The exact configuration can be found in the device specific configuration.*

For the alarm description please refer to the alarm mapping tables in the SMU functional block and the Safety Manual.

Related information

eGTM implementation on page 7572
eGTM control registers on page 7575
Document structure on page 7309

44.3 Functional description

Information to the IP specification and implementation you will find in the respective sub chapters.

In this sub-chapter the eGTM module is documented. It includes out of the following sub-parts:

- eGTM IP specification (specification from Bosch AE)
- eGTM implementation (the integration and additional registers)
- and if applicable a HRPWM sub-chapter

Related information

eGTM implementation on page 7572

44.3.1 eGTM Specification

- Version: GTM4.1-V1.40 patched
- Date: 2024-06-26

44.3.1.1 Introduction

44.3.1.1.1 Overview

This document is the specification for the enhanced Generic Timer Module (**eGTM**).

Nevertheless, the **eGTM** is designed to unload the CPU or a peripheral core from a high interrupt load. Most of the tasks inside the **eGTM** run (once setup by an external CPU) independently and in parallel to the CPU software. There may be special situations, where the CPU has to take action but the goal of the **eGTM** design was to reduce these situations to a minimum.

The hardware sub-modules have dedicated functionalities, for example, there are timer input modules where incoming signals are captured and characterized together with a notion of time.

The sub-modules are categorized into the following groups:

Interface components: The slave communication from CPU to **eGTM** is covered by the sub-module (E)GTM_AEI.

Infrastructure components: These sub-modules are needed to define common functionality in the **eGTM** and ensure that data storage and exchange between other sub-modules is possible. They are in use to implement complex functions which have to pass data over multiple sub-modules to ensure the user application. These components are present in all **eGTM** devices. However, the number of these components may vary from device to device.

IO Modules: They have connections with input or output signals to communicate from and to the **eGTM**. They are built based on a regular architecture to fulfill typical timer functions, for example, usable as PWM generation units, input measurement units.

Dedicated functionality: These sub-modules are those fulfilling a dedicated functionality for a certain application domain.

Interrupt management: The module **ICM** is responsible for interrupt services.

Safety features: These sub-modules are able to support the implementation of safety functions to fulfill a defined safety level.

44.3.1.1.2 Document structure

The structure of this document is motivated by the aforementioned sub-module groups. The Overview describes the dedicated **eGTM** architecture. It gives an overview of the structuring of the implemented sub-modules.

The following chapters up to Time base unit (**TBU**) deal with the so-called infrastructure components for clock management and common time base functions. Timer input module (**TIM**) to Dead-time module (**DTM**) describe the signal input and output modules. The next chapters up to Sensor pattern evaluation (**SPE**) provide detailed descriptions of application-specific modules like the **SPE**. Interrupt concentrator module (**ICM**) describes a module that bundles several interrupts coming from the other sub-modules and connect them to the outside world. The last Output compare unit (**CMP**) to Monitor unit (**MON**) provide information on the safety related module.

Table 2144 Sub-module groups

Chapter	Sub-module	Group
eGTM architecture	Slave configuration interface	Interface components
Clock management unit (**CMU**)	Clock Management Unit (**CMU**)	Infrastructure components
Cluster configuration module (**CCM**)	Cluster Configuration Module (**CCM**)	Infrastructure components
Time base unit (**TBU**)	Time Base Unit (**TBU**)	Infrastructure components
Timer input module (**TIM**)	Timer Input Module (**TIM**)	IO Modules
Timer output module (**TOM**)	Timer Output Module (**TOM**)	IO Modules
ARU-connected timer output module (**ATOM**)	**ARU**-connected Timer Output Module (**ATOM**)	IO Modules
Dead-time module (**DTM**)	Dead-Time Module (**DTM**)	IO Modules
Sensor pattern evaluation (**SPE**)	Sensor Pattern Evaluation Module (**SPE**)	Dedicated functionality BLDC support
Interrupt concentrator module (**ICM**)	Interrupt Concentrator Module (**ICM**)	Infrastructure components
Output compare unit (**CMP**)	Output Compare Unit (**CMP**)	Safety features
Monitor unit (**MON**)	Monitoring Unit (**MON**)	Safety features

44.3.1.2 eGTM architecture

44.3.1.2.1 Overview

Find here a list of indices and their range which are used in all chapters with the same semantic:

- i:={0, 1,..., NCCM-1} index of cluster or module instance in the **eGTM** device
- j in case the index refers to a cluster/module j!=i
- x:={0, 1,..., 15} index of a channel inside a module
- d:={0, 1,..., 5} index of **DTM** unit after **TOM** / **ATOM**
- g:={0, 1} index of group of resources/functions

In addition, more indices might be introduced at the beginning of the following chapters. These additional indices with their corresponding ranges are used throughout the respective chapters they are defined in.

As already mentioned in Introduction the **eGTM** forms a generic timer platform that serves different application domains and different classes within these application domains. Depending on these multiple requirements of application domains, multiple device configurations with different number of sub-modules (i.e. **ATOM**, **SPE**, **TIM**, **TOM** and **DTM**) and different number of channels per sub-module (if applicable) are possible.

The device dependent configuration (i.e. the number of sub-modules) is listed in the device-specific documentation.

The Cluster Dead-time Module (**CDTM**) is also a virtual hierarchy and consists of up to twelve **DTM** modules. It depends on the **eGTM** device configuration which of the twelve **DTM** instances are available. In general, the first four **DTM** modules (d∈{0, 1, 2, 3}) inside a CDTM[i] hierarchy are connected to the outputs of the **TOM** instance i of the cluster i. The **DTM** instances (d∈{4, 5}) are connected to the outputs of the **ATOM** instance i of this cluster i.

The cluster view of a **eGTM** architecture is depicted in Figure 1016. This is a generic figure which shows an exemplary **eGTM** device configuration.

The device dependent configuration (i.e. the count of sub-modules and channels per sub-module) is listed in the device-specific part of this document.

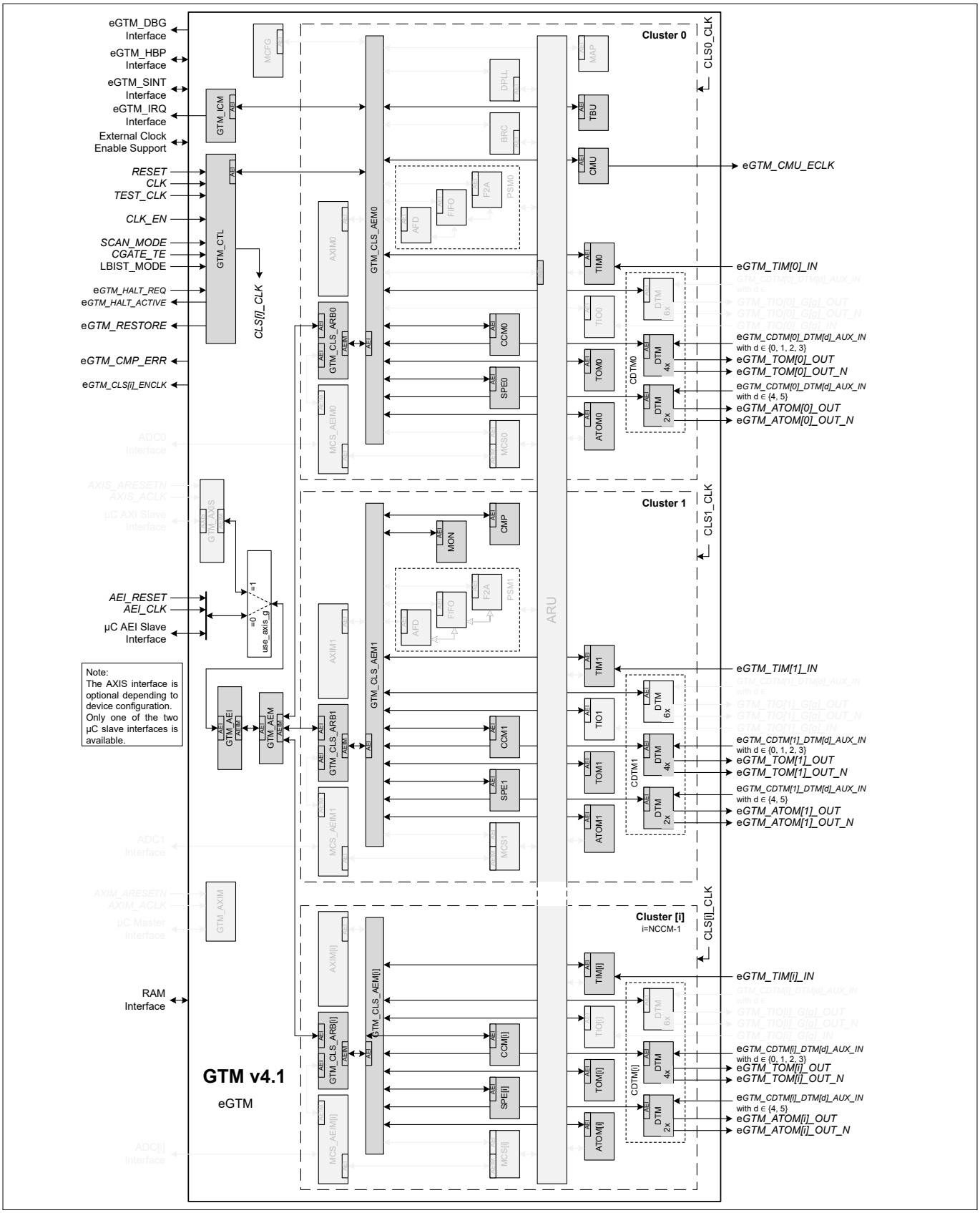


Figure 1016 eGTM architecture block diagram

***Note**: Faded content is not existing on the device.*

44 Enhanced Generic Timer Module (eGTM)

The **eGTM** is divided in multiple clusters 0...n. A certain number of modules exist in each cluster. The operating frequency of a cluster can be configured to OFF, CLK or CLK / 2 by appropriately configuring CLSi_ARCH_CLK_CFG**.**CLSi_ARCH_CLK_CFG **(j=i)**. The clock enable generation can be implemented internal to the **eGTM** or external. In case of an external enable generation, CLK_EN is used to generate the internal clocks. In addition, an enable watchdog is implemented to monitor the correctness of the externally applied enable signals CLK_EN. More about clocking in the **eGTM** follows in Introduction to clocking.

Signals are transferred into the **eGTM** at the Timer Input Modules (**TIM**). These modules are able to filter the input signals and annotate additional information. Each channel is, for example, able to measure pulse high or low times and the period of a PWM signal in parallel . The internal operation registers of the **TIM** sub-module are 24-bit wide.

The Clock Management Unit (**CMU**) which generates up to 14 clock resolutions for the sub-modules of the **eGTM** and up to three **eGTM** external clocks. The clock resolutions are CMU_CLK_RES[x:x] and CMU_FXCLK_RES[y:y], where $x \in \{0, 1, ..., 8\}$ and $y \in \{0, 1, ..., 3\}$. These signals are routed to all clusters via ICPAs. Within a cluster, they are used as clock enables regulating the resolution (or rate) at which actions are performed. The external clocks are CMU_ECLK[z:z], where $z \in \{0, 1, 2\}$. In contrast to CMU_CLK_RES[x:x], these are approximately 50% duty cycle clocks. The primary clock source for this sub-module is cluster 0 clock signal (CLS[j]_CLK) which can be configured via CLSi_ARCH_CLK_CFG**.**CLSi_ARCH_CLK_CFG **(j=0)** to OFF, CLK or CLK /2. For a detailed description of the **CMU** functionality and clocks please refer to Clock management unit (**CMU**).

The **TBU** provides up to three independent common time bases for the **eGTM**. In general, the number of time bases depends on the implemented device.

Signal outputs are generated with the Dead-time Module (**DTM**), Timer Output Modules (**TOM**) and the **ARU**-connected **TOM**s (**ATOM**). Each **TOM** channel is able to generate a PWM signal at its output. Because of the integrated shadow register (see here) even the generation of complex PWM outputs is possible with the **TOM** channels by serving the parameters with the CPU. It is possible to trigger **TOM** channels for a successor **TOM** sub-module through a trigger line between **TOM[i]** channel 15 and **TOM[i+1]** channel 0. But to avoid long trigger paths the **eGTM** integrator configures, after which **TOM** sub-module instance, a pipeline register is placed into the trigger signal chain. Each pipeline register results in one CLK cycle delay of the trigger signal. Please refer to the device-specification of the silicon vendor for unregistered trigger chain length.

In addition, each **TOM** sub-module integrates functions to drive one BLDC engine. This BLDC support is established together with the **TIM** and Sensor Pattern Evaluation (**SPE**) sub-module.

The **ATOM**s offer the additional functionality to generate complex output signals without CPU interaction . While the internal operation and shadow registers of the **TOM** channels are 16 bits wide, the operation and shadow registers of the **ATOM** channels are 24 bits wide to have a higher resolution and to have the opportunity to compare against time base values coming from the **TBU**.

It is possible to trigger **ATOM** channels for a successor **ATOM** sub-module through a trigger line between **ATOM[i]** channel 7 and **ATOM[i+1]** channel 0. But to avoid long trigger paths the **eGTM** integrator can configure after which **ATOM** sub-module instance a pipeline register is placed into the trigger signal chain. Each pipeline register results in one CLK cycle delay of the trigger signal. Please refer to the device-specification of the silicon vendor for unregistered trigger chain length.

The two modules Compare Module (**CMP**) and Monitor Module (**MON**) implement safety related features. The **CMP** compares two output channels of the **DTM** and sends the result to the **MON** sub-module, where the error is signaled to the CPU. The **MON** module is also able to monitor the **CMU** activities.

In the described implementation, the sub-modules of the **eGTM** have a huge number of different interrupt sources. These interrupt sources are grouped and concentrated by the Interrupt Concentrator Module (**ICM**) to form a much easily manageable bunch of interrupts that are visible outside of the **eGTM**.

44.3.1.2.2 Introduction to clocking

The **eGTM** offers many possibilities on different granularity levels to reduce the power budget during run-time. The possibilities include shutting off a particular cluster/module or clocking them at a reduced frequency, among others.

The first layer of configurability is on the cluster level. A cluster clock can be shut off completely or configured to use the **eGTM** global clock or the **eGTM** global clock halved by appropriately configuring CLSi_ARCH_CLK_CFG. In the **eGTM**, cluster clocks can be configured differently. The **CCM** module also offers the possibility to enable/disable different modules in the cluster by appropriately configuring CLSi_CCM_CMU_CLK_CFG**.**CLKy_SRC (y=0-7), where 0 <= i <= NCCM - 1 and 0 <= y <= 7.

The second layer of configurability is offered by the **CMU** module (exists only in cluster 0). This module takes the cluster clock as input and uses the clock dividers within to generate different clock resolutions (CMU_CLK_RES[x:x] (x ∈ {0, 1,..., 8})) that are routed through ICPA (refer to Inter-cluster pulse adapter) to the **CCM** modules in all the clusters. These signals give the user the possibility to further regulate the rate (or the resolution) at which the functions in different modules run at (see Figure 1036). A CMU_CLK_RES[x:x] signal can be enabled/disabled by appropriately configuring CLSi_CMU_CLK_EN**.**EN_CLKx (x=0-7).

The third layer of configurability is offered by the **CCM** module (exists in every cluster). Different clusters can be clocked differently. This module samples the CMU_CLK_RES[x:x] resolution signals at the cluster clock. The sampled clocks are then routed to different modules within the cluster in which they are used as clock enables. This allows different modules within a cluster to run at different resolutions.

Attention: The **eGTM** offers various debugging mechanisms (more about that in **eGTM** software debugger support). When debugging is active and (e)GTM_HALT_REQ is asserted, the cluster clocks are disabled in order to stall the internal operations in the **eGTM** during the debugging session.

44.3.1.2.3 Connectivity between modules

On the **eGTM** top-level there are some configurable signal connections from the signal output of the **DTM** modules to the input signals of the **TIM** modules.

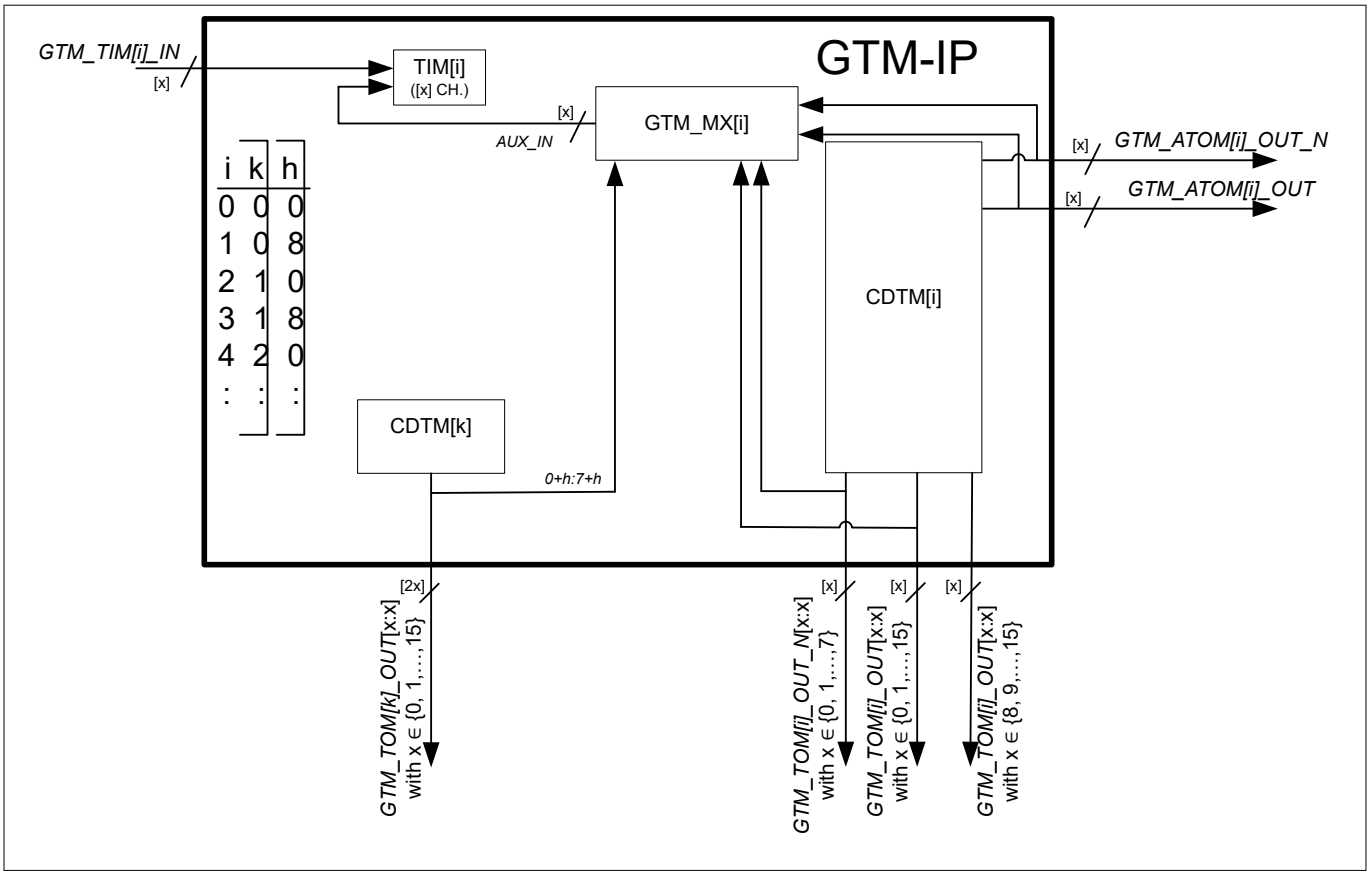


Figure 1017 eGTM signal multiplex

The next diagram gives an overview of the connectivity for different configuration of bit field CLSi_ARCH_CFG.SRC_IN_MUX and the cluster configuration register CLSi_CCM_TIM_AUX_IN_SRC. The source selection is defined per channel with the bit fields CLSi_CCM_TIM_AUX_IN_SRC.SRC_CH0, CLSi_CCM_TIM_AUX_IN_SRC.SRC_CH1, CLSi_CCM_TIM_AUX_IN_SRC.SRC_CH2, CLSi_CCM_TIM_AUX_IN_SRC.SRC_CH3, CLSi_CCM_TIM_AUX_IN_SRC.SRC_CH4, CLSi_CCM_TIM_AUX_IN_SRC.SRC_CH5, CLSi_CCM_TIM_AUX_IN_SRC.SRC_CH6, CLSi_CCM_TIM_AUX_IN_SRC.SRC_CH7 and CLSi_CCM_TIM_AUX_IN_SRC.SEL_OUT_N_CHx (x=0-7).

The calculation of the index k, h and the connectivity of the next figure covers devices with 8 **TIM** modules.

In case a device is equipped with more **TIM** instances as **ATOM** instances (NTIM > NATOM), the signals (e)GTM_ATOM[i]_OUT[x:x], (e)GTM_ATOM[i]_OUT_N[x:x] are tied to 0; i>NTIM.

CLSi_ARCH_CFG.SRC_IN_MUX=1: In case a device is equipped with more **TIM** instances as **TOM** instances (NTIM > NTOM), the signals (e)GTM_TOM[i]_OUT[x:x], (e)GTM_TOM[i]_OUT_N[x:x] are tied to 0; i>NTIM.

CLSi_ARCH_CFG.SRC_IN_MUX=0: In case a device is equipped with lesser **TIM** instances as **TOM** instances (2* NTIM < NTOM), the signals (e)GTM_TOM[i]_OUT[x+h:x+h] are tied to 0; i>NTIM.

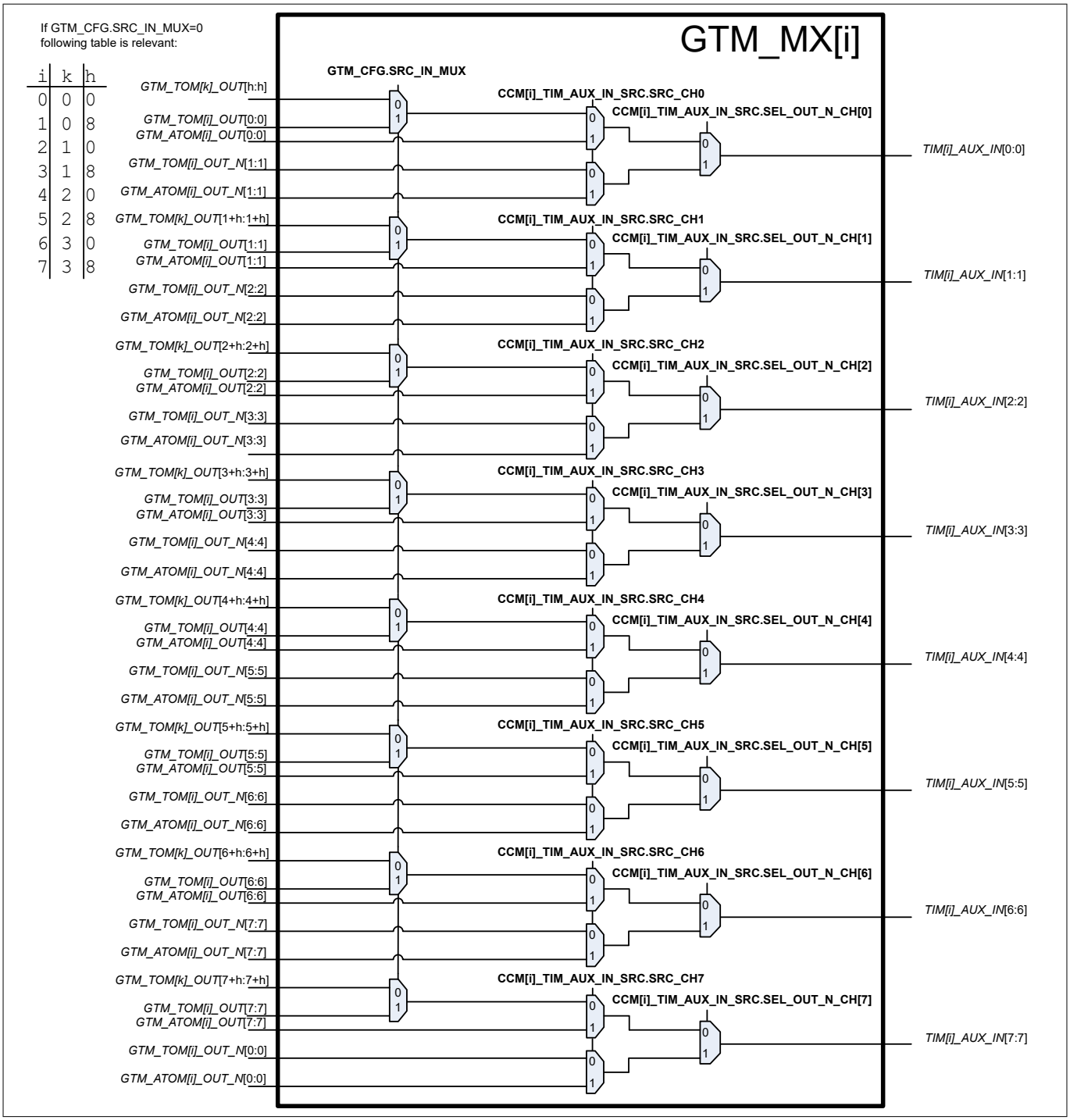


Figure 1018 TIM auxiliary input multiplexing

The trigger out of **TIM** (i.e. the signals TIM_EXT_CAPTURE[7:0] of each **TIM** instance i) are routed to **ATOM** instance i and **TOM** instance i with i∈{0, 1,..., NTIM-1} (NTIM defines the number of available **TIM** instances).

This **TIM** trigger is used to trigger inside the **ATOM** or **TOM** instance either a channel or the global control register of AGC or TGC0/TGC1 unit.

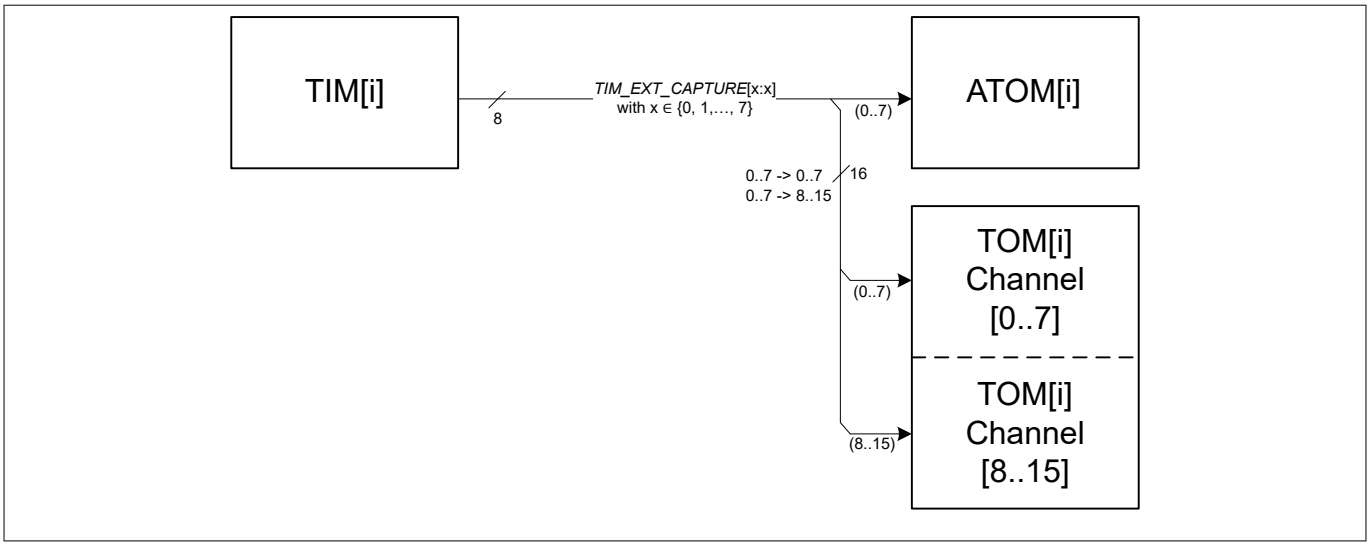


Figure 1019 TIM external capture forwarding to TOM and ATOM

The following two figures show, how signals are routed to the **DTM** module.

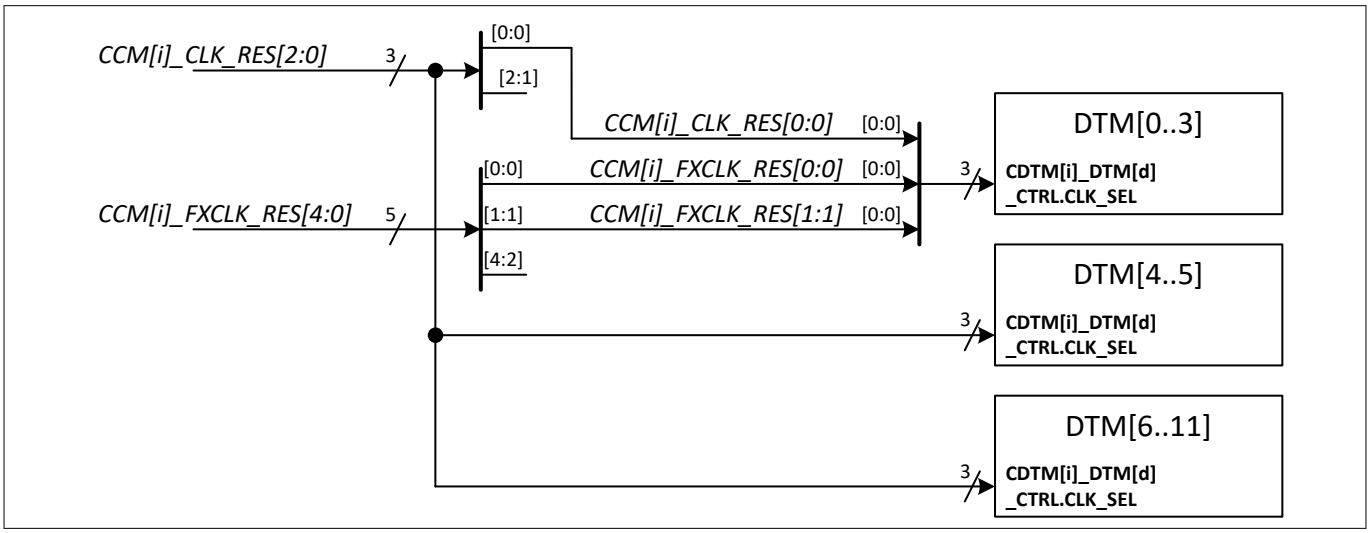


Figure 1020 Routing of CMU clock signals to DTM modules

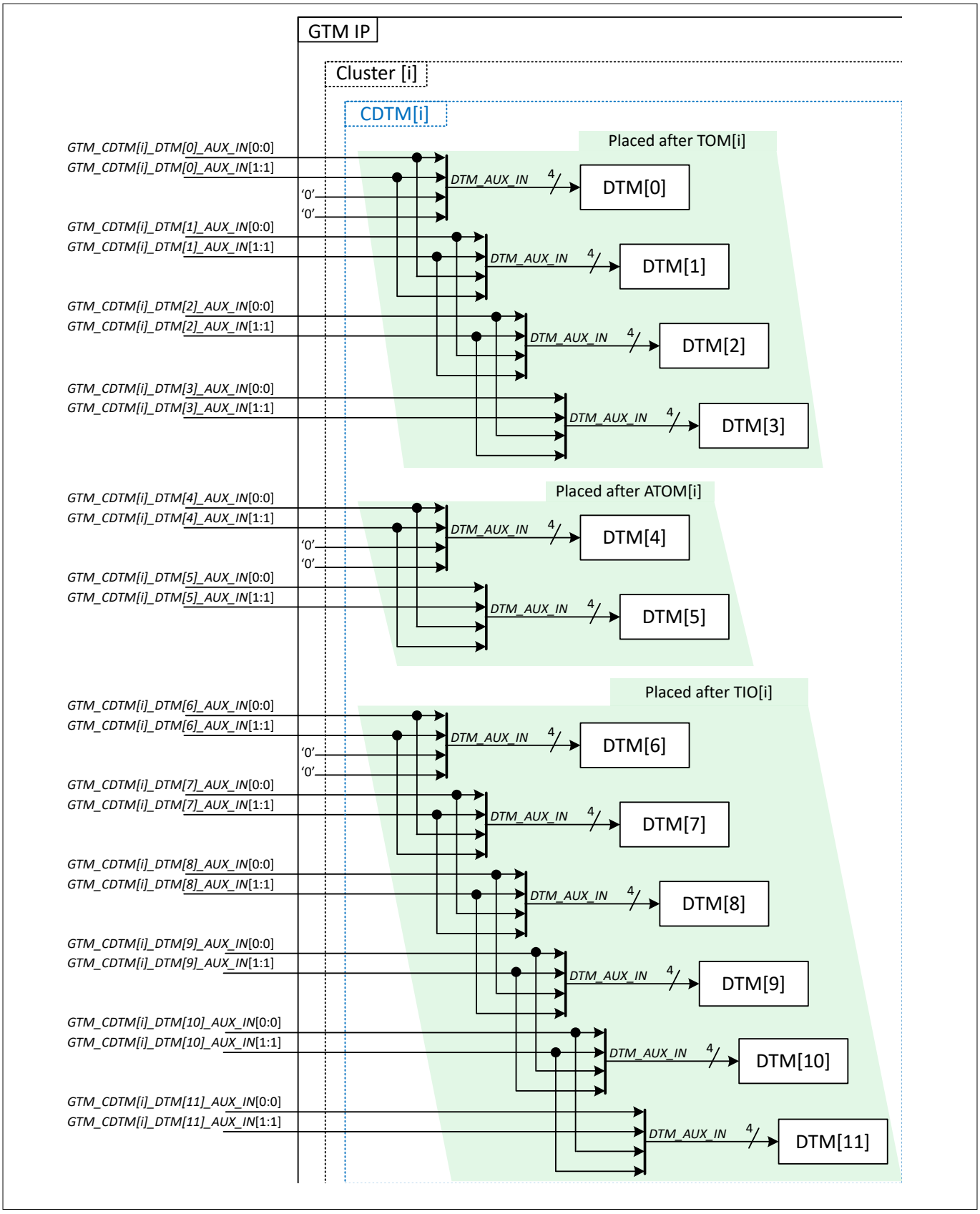


Figure 1021 Routing of DTM AUX input signals

***Note**: The figure is a general picture from the IP vendor. The exact configuration can be found in the device specific configuration.*

44.3.1.2.4 eGTM interfaces

In general, the **eGTM** is divided into multiple interface groups. Two interface groups represent the ports of the **eGTM** where incoming signals are assembled and outgoing signals are created. These interfaces are therefore connected to the **eGTM** input sub-module **TIM** and to the **eGTM** output sub-modules **DTM**.

Another interface is the bus master interface where the **eGTM** is connected to the system bus and actively drive read and write operations onto the system bus. This generic bus interface is described in more detail in **eGTM** generic bus interface (**AEI**).

Another interface is the bus slave interface where the **eGTM** is connected to the system bus to allow access to the **eGTM** internal resources described in more detail in **eGTM** generic bus interface (**AEI**).

The last interface is the interrupt controller interface. The **eGTM** provides several interrupt lines coming from the various sub-modules. These interrupt lines are concentrated inside the **ICM** and have to be adapted to the dedicated microcontroller environment where each interrupt handling can look different. The interrupt concept is described in more detail in **eGTM** interrupt concept.

eGTM generic bus interface (AEI)

The **eGTM** is equipped with a generic bus interface. This generic bus interface is called AE_Interface (**AEI**). .

Bit	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	0	0	0	cluster index				m/I	m/I	m/I	m/I	m/I	m/I	m/I	m/I	m/I	m/I	m/I	m/I	m/I	m/I	m/I	0	0

23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		Address range	Accessibility
0	0	0	a				0	0	0	0	0	0	0	0	0	0	0	I	I	I	I	0	0	ARCH	0x000000 - 0x00003C	CPU and MCS[0]
0	0	0	a				0	0	0	0	0	0	0	0	0	0	1	0	0	I	I	0	0	AEI	0x000040 - 0x00004C	CPU
0	0	0	a				0	0	0	0	0	0	0	0	0	1	I	I	I	I	I	0	0	CMU	0x000080 - 0x0000FC	CPU and MCS[0]
0	0	0	a				0	0	0	0	0	0	0	0	1	0	I	I	I	I	I	0	0	TBU	0x000100 - 0x00017C	CPU and MCS[0]
0	0	0	a				0	0	0	0	0	0	0	0	1	1	I	I	I	I	I	0	0	ARU	0x000180 - 0x0001FC	CPU and MCS[0]
0	0	0	a				0	0	0	0	0	0	0	1	0	0	I	I	I	I	I	0	0	BRC	0x000200 - 0x00027C	CPU and MCS[0]
0	0	0	a				0	0	0	0	0	0	1	0	I	I	I	I	I	I	I	0	0	ICM	0x000400 - 0x0005FC	CPU and MCS[0]
0	0	0	a				0	0	0	0	0	0	1	1	0	0	0	I	I	I	I	0	0	MCFG	0x000600 - 0x00063C	CPU and MCS[0]
0	0	0	a				0	0	0	0	0	0	1	1	0	0	1	I	I	I	I	0	0	MAP	0x000640 - 0x00067C	CPU and MCS[0]
0	0	0	b				0	0	0	0	0	0	1	1	0	1	0	I	I	I	I	0	0	MON	0x000680 - 0x0006BC	CPU and MCS[1]
0	0	0	b				0	0	0	0	0	0	1	1	0	1	1	I	I	I	I	0	0	CMP	0x0006C0 - 0x0006FC	CPU and MCS[1]
0	0	0	d				0	0	0	0	0	1	I	I	I	I	I	I	I	I	I	0	0	TIM[i]	0x000800 - 0x000FFC	CPU and MCS[i]
0	0	0	f				0	0	0	0	1	0	I	I	I	I	I	I	I	I	I	0	0	TOM[i]	0x001000 - 0x0017FC	CPU and MCS[i]
0	0	0	f				0	0	0	0	1	1	I	I	I	I	I	I	I	I	I	0	0	ATOM[i]	0x001800 - 0x001FFC	CPU and MCS[i]
0	0	0	e				0	0	0	1	0	I	I	I	I	I	I	I	I	I	I	0	0	MCS[i]	0x002000 - 0x002FFC	CPU and MCS[i]
0	0	0	f				0	0	0	1	1	I	I	I	I	I	I	I	I	I	I	0	0	TIO[i]	0x003000 - 0x003FFC	CPU and MCS[i]
0	0	0	f				0	0	1	0	0	0	0	0	I	I	I	I	I	I	I	0	0	CCM[i]	0x004000 - 0x0041FC	CPU and MCS[i]
0	0	0	f				0	0	1	0	0	0	1	I	I	I	I	I	I	I	I	0	0	CDTM[i]	0x004400 - 0x0047FC	CPU and MCS[i]
0	0	0	c				0	0	1	0	0	1	0	0	0	0	I	I	I	I	I	0	0	F2A[i] (PSM[i])	0x004800 - 0x00487C	CPU and MCS[i]
0	0	0	c				0	0	1	0	0	1	0	0	0	1	I	I	I	I	I	0	0	AFD[i] (PSM[i])	0x004880 - 0x0048FC	CPU and MCS[i]
0	0	0	c				0	0	1	0	0	1	0	1	I	I	I	I	I	I	I	0	0	FIFO[i] (PSM[i])	0x004A00 - 0x004BFC	CPU and MCS[i]
0	0	0	d				0	0	1	0	0	1	1	0	0	0	I	I	I	I	I	0	0	SPE[i]	0x004C00 - 0x004C7C	CPU and MCS[i]
0	0	0	e				0	0	1	0	1	0	0	I	I	I	I	I	I	I	I	0	0	AXIM[i]	0x005000 - 0x0053FC	CPU and MCS[i]
0	0	0	e				0	0	1	0	1	0	1	0	I	I	I	I	I	I	I	0	0	ADC[i]	0x005400 - 0x0055FC	MCS[i]
0	0	0	a				0	0	1	0	1	0	1	1	0	0	0	I	I	I	I	0	0	MCS2DPLL	0x005600 - 0x00563C	MCS[0]
0	0	0	c				0	0	1	1	0	I	I	I	I	I	I	I	I	I	I	0	0	FIFO[i]_MEM	0x006000 - 0x006FFC	CPU and MCS[i]
0	0	0	a				0	1	0	I	I	I	I	I	I	I	I	I	I	I	I	0	0	DPLL	0x008000- 0x00BFFC	CPU and MCS[0]
0	0	0	a				0	1	1	I	I	I	I	I	I	I	I	I	I	I	I	0	0	DPLL_RR2	0x00C000 - 0x00FFFC	CPU and MCS[0]
0	0	0	e				1	I	I	I	I	I	I	I	I	I	I	I	I	I	I	0	0	MCS[i]_MEM	0x010000 - 0x01FFFC	CPU and MCS[i]

Bits 20:17	Availability (cluster range)
a	0000
b	0001
c	0000 .. 0010
d	0000 .. 0111
e	0000 .. 1001
f	0000 .. 1011

Figure 1022 Overview of eGTM address space

***Note**: The figure is a general picture from the IP vendor. The exact configuration can be found in the device specific configuration.*

The modules (**ARCH**, **AEI**, **CMU**, **TBU**, **ICM**, **CMP**) may only be available in cluster 0 address space.

The modules (**MON**, **CMP**) may only be available in cluster 1 address space.

The (**TIM**, **TOM**, **ATOM**, **CCM**, **SPE**, **DTM**) modules may only fill out the clusters starting from cluster 0 onwards.

The **ARCH** configuration registers and the **AEI** configuration registers share the following properties:

- They are responsible for configuring functionalities spanning/impacting multiple clusters (CLSi_ARCH_RST, CLSi_ARCH_CFG, CLSi_ARCH_CLK_CFG).
- They are responsible for atomically switching on/off functionalities across multiple clusters (CLSi_ARCH_RST, CLSi_ARCH_CFG, CLSi_ARCH_CLK_CFG).
- They are responsible for configuring and providing status of the interfaces (CLSi_ARCH_CTRL, CLSi_ARCH_IRQ_NOTIFY, CLSi_ARCH_AEI_ADDR_XPT, CLSi_ARCH_AEI_STA_XPT, CLSi_ARCH_RST).
- They are still accessible when cluster 0 is switched off.

The **AEI** status signal may drive one of the following values:

Table 2145 AEI status signal

AEI_STATUS coding	Description
00_B	No Error
01_B	Illegal Byte-addressing
10_B	Illegal Address Access
11_B	Unsupported Address

The evaluation of the status of a read-/write-access is executed based on a priority scheme. The ordering of priorities is defined in the following:

1. PRIO1 checks are validated first, if a check condition matches, the corresponding AEI_STATUS is returned, otherwise checking continues with PRIO2.
2. PRIO2 checks are validated next, if a check condition matches, the corresponding AEI_STATUS is returned, otherwise checking continues with PRIO3.
3. PRIO3 checks are validated next, if a check condition matches, the corresponding AEI_STATUS is returned, otherwise checking continues with PRIO4.
4. PRIO4 checks are validated next, if a check condition matches, the corresponding AEI_STATUS is returned, otherwise AEI_STATUS = 00_B is returned.

The signal value 01_B is returned:

- PRIO1: if the bus address is not an integer multiple of 4 (byte-addressing).

For cluster or module clock disable functionality, the following address range definitions exist:

Table 2146 Cluster address range access is disabled by CLSi_ARCH_CLK_CFG.CLSj_CLK_DIV (j=0-2) (j=i)=00_B

Cluster i	Address range
0	000050_H - $0003FF_H$; 000600_H - $01FFFF_H$
1	020000_H - $03FFFF_H$
2	040000_H - $05FFFF_H$
3	060000_H - $07FFFF_H$
4	080000_H - $09FFFF_H$
5	$0A0000_H$ - $0BFFFF_H$
6	$0C0000_H$ - $0DFFFF_H$
7	$0E0000_H$ - $0FFFFF_H$
8	100000_H - $11FFFF_H$
9	120000_H - $13FFFF_H$
10	140000_H - $15FFFF_H$
11	160000_H - $17FFFF_H$

Table 2147 Module address range access is disabled by module clock enable = 0_B

Module clock enable	Implemented in cluster i if condition is true	Module name	Module address offset range
CLSi_CCM_CFG.EN_TIM	i < NTIM	**TIM**	000800_H - $000FFF_H$
CLSi_CCM_CFG.EN_TOM_SPE_TDTM	i < NTOM	**TOM**	001000_H - $017FFF_H$
CLSi_CCM_CFG.EN_TOM_SPE_TDTM	i < NTOM & CDTM[i][d] (d∈{0, 1, 2, 3})	TDTM	004400_H - $0044FF_H$
CLSi_CCM_CFG.EN_TOM_SPE_TDTM	i < NSPE	**SPE**	$004C00_H$ - $004C7F_H$
CLSi_CCM_CFG.EN_ATOM_ADTM	i < NATOM	**ATOM**	001800_H - $01FFFF_H$
CLSi_CCM_CFG.EN_ATOM_ADTM	i < NATOM & CDTM[i][d] (d∈{4, 5})	ADTM	004500_H - $00457F_H$
CLSi_CCM_CFG.EN_CMP_MON	i == NMON & i == 1	**MON**	000680_H - $0006BF_H$
CLSi_CCM_CFG.EN_CMP_MON	i == NCMP & i == 1	**CMP**	$0006C0_H$ - $0006FF_H$

The signal value 10_B is returned for an access where the target address is not accessible (cluster / module shutoff or internal operation on-going):

- PRIO2: on a read or write, if the address is located in the cluster i address range, which is disabled by CLSi_ARCH_CLK_CFG.CLSj_CLK_DIV (j=0-2) **(j=i)** = 00_B
- PRIO2: on a read or write, if the address is located in the module **TIM[i]** address range, which is disabled by CLSi_CCM_CFG.EN_TIM = 0_B
- PRIO2: on a read or write, if the address is located in the module **TOM[i]**, **SPE[i]**, CDTM[i]_DTM[3:0] address range, which is disabled by CLSi_CCM_CFG.EN_TOM_SPE_TDTM = 0_B
- PRIO2: on a read or write, if the address is located in the module **ATOM[i]**, CDTM[i]_DTM[5:4] address range, which is disabled by CLSi_CCM_CFG.EN_ATOM_ADTM = 0_B
- PRIO2: on a read or write, if the address is located in the module **CMP**, **MON** address range, which is disabled by CLSi_CCM_CFG.EN_CMP_MON = 0_B

The signal value 10_B is returned for a protected write-access:

- PRIO4: if for the written register all writable bit fields are equipped with the same write-protection condition and the protection is active (independently of the written value).

In case of an illegal write-access signaled by status 10_B the register will not be modified.

The signal value 11_B is returned:

- PRIO3: if for the requested address no **eGTM** register or **eGTM** memory location is defined
- PRIO3: if the requested address is not implemented in the actual device (device dependency attribute of that address evaluates to false)

The signal value 00_B is returned:

- If no error occurred during **AEI** access.
- If all register bit fields are defined as read only, a write to this register will not modify any content.
- If a register bit field is defined as read only, a write to this register bit field will not modify the content. All bit fields of the same register which are writable will be written as requested.

- If a register contains a protected bit field and the protection is active the bit field will not be modified. All bit fields of the same register which have no protection will be written as requested.
- If a bit field of a register is defined as reserved, any write-access to this bit field has no side-effect and a read-access to this bit field will always return the value 0.
- If a bit field of a register is not implemented in the actual device (device dependency attribute of that bit field evaluates to false) this is treated as reserved, any write-access to this bit field has no side-effect and a read-access to this bit field will always return the value 0.
- If one or more encoding values of a bit field are defined as reserved values and nothing else is specified, the following default behavior is defined: The written value can be read back and the entire behavior of the circuit is undefined.

eGTM AEI bridge software reset

The configuration bit field CLSi_AEI_BRIDGE_MODE.BRG_RST is used for software reset of the (E)GTM_AEI module.

Writing the value 1_B to the bit field CLSi_AEI_BRIDGE_MODE.BRG_RST via the configuration interface, will immediately reset the content of the following registers to the initial hardware reset state.

- CLSi_AEI_BRIDGE_MODE: Details about which bit fields are reset follow in CLSi_AEI_BRIDGE_MODE register description.
- CLSi_AEI_BRIDGE_PTR1
- CLSi_AEI_BRIDGE_PTR2
- CLSi_ARCH_AEI_ADDR_XPT
- CLSi_ARCH_AEI_STA_XPT
- Internal registers inside the (E)GTM_AEI module which are not memory mapped are reset too.

44.3.1.2.5 AEI bus system

Table 2148 Functionality of AEI bus system modules

Module	Functionality
(E)GTM_AEI	Implements the top level bus bridge. It allows to operate the **eGTM AEI** bus interface on an individual clock frequency. It supports 3 different **AEI** protocol versions which have different performance properties. The semiconductor vendor will choose a protocol variant which suits best for the integration of the **eGTM** . In the module a transaction buffer is implemented to store incoming transactions and support for postponed transactions.
(E)GTM_AEM	Decodes the **eGTM** internal address space into the cluster address ranges. Transactions will only be forwarded to the cluster which is addressed.
(E)GTM_CLS_ARB	Performs arbitration of the 2 possible cluster **AEI** masters (MCS_AEIM, (E)GTM_AEM). Arbitration will be performed on each single **AEI** access.
MCS_AEIM	This signal has no impact on **eGTM** .

(table continues...)

Table 2148 (continued) Functionality of AEI bus system modules

Module	Functionality
(E)GTM_CLS_AEM	Decodes the cluster internal address space into the module address ranges. Transactions will only be forwarded to the module which is addressed.
AEI	Interface of a module which acts as **AEI** slave.
AEIM	Interface of a module which acts as **AEI** master.

(E)GTM_AEM module

This module receives accesses from the (E)GTM_AEI bridge and decodes (based on the provided address AEI_ADDR to which cluster i) the access has to be forwarded. The cluster index is decoded by i = AEI_ADDR[10:17]. In case the targeted cluster is switched off (CLSi_ARCH_CLK_CFG**.**CLSj_CLK_DIV (j=0-2) **(j=i)** = 0), the access is not forwarded and AEI_STATUS = 10_B with AEI_RDATA = 0_H is responded for this access. In case the targeted cluster i is not implemented in this device, the access is responded with AEI_STATUS = 11_B and AEI_RDATA = 0_H.

(E)GTM_CLS_ARB module

***Note**: The EGTM does not include an MCS nor a PSM.*

The **eGTM** is equipped with the possibility to introduce pipeline stages in the **AEI** bus system for accesses from the CPU. Details how to control the pipeline stage insertion can be found in the module integration guide. In big devices, the decoding of the combinatorial address space for the register accesses and the Rdata multiplexing will not be possible (due to limited performance of technology) in one clock cycle.

- AEI_ADDR_PIPELINE_STAGE: Pipeline registers are included for all **AEI** request signals in the **AEI** address decoding path at the input of (E)GTM_CLS_ARB for the signals which are driven by the (E)GTM_AEM. The signals (AEI_SEL, AEI_ADDR, AEI_WDATA, AEI_W1R0, AEI_DEBUG_ACCESS) will be delayed by 1 **eGTM** clock (CLK).
- AEI_RDATA_PIPELINE_STAGE: Pipeline registers are included for all **AEI** response signals in the **AEI** access multiplexing path at the output of the (E)GTM_CLS_ARB for the signals which are received by (E)GTM_AEM. The signals (AEI_RDATA, AEI_READY, AEI_STATUS) will be delayed by 1 **eGTM** clock (CLK). As a consequence, the latency of bus transaction from the CPU will increase in case pipeline registers are put in place.

(E)GTM_CLS_AEM module

This module receives accesses from the (E)GTM_CLS_ARB module and decodes to which module the access has to be forwarded based on the provided address AEI_ADDR. The sub module decoding can be found in Figure 1022. In case the targeted module is switched off (corresponding enable bit in CLSi_CCM_CFG is set to 0), the access is not forwarded and AEI_STATUS = 10_B with AEI_RDATA = 0_H is responded for this access. In case the targeted module is not implemented in this device, the access is responded with AEI_STATUS = 11_B and AEI_RDATA = 0_H.

44.3.1.2.6 GTM_CTL module

AEI bus system transaction status monitoring for transactions applied to the eGTM

GTM_CTL allows monitoring of transactions initiated from **eGTM** external bus masters (CPU, DMA,..). Monitoring takes place on the signals after the (E)GTM_AEI bridge and operates on the CLK clock.

Each transaction response on the **AEI** bus system will indicate (with the AEI_STATUS signal) the current state of execution of the transaction (see Table 2145). To allow an application software to react on unexpected state of transactions (AEI_STATUS != 00_B), a transaction monitoring and signaling with an interrupt on (e)GTM_AEI_IRQ is implemented.

For monitoring the first transaction which occurs with AEI_STATUS != 0, GTM_CTL stores the transaction type in CLSi_ARCH_AEI_STA_XPT**.**W1R0 and the transaction address in CLSi_ARCH_AEI_STA_XPT**.**ADDR. Following transactions with (AEI_STATUS != 00_B) will not change the content of CLSi_ARCH_AEI_STA_XPT. Furthermore, for each type of AEI_STATUS an individual interrupt unit (see CLSi_ARCH_IRQ_NOTIFY**.**AEI_USP_ADDR, CLSi_ARCH_IRQ_NOTIFY**.**AEI_IM_ADDR, and CLSi_ARCH_IRQ_NOTIFY**.**AEI_USP_BE) is implemented. This allows applications to individually select for which kind of transaction response exception, an interrupt (e)GTM_AEI_IRQ (see CLSi_ARCH_IRQ_EN**.**AEI_USP_ADDR_IRQ_EN, CLSi_ARCH_IRQ_EN**.**AEI_IM_ADDR_IRQ_EN, and CLSi_ARCH_IRQ_EN**.**AEI_USP_BE_IRQ_EN) or/and error interrupt (e)GTM_ERR_IRQ (CLSi_ARCH_EIRQ_EN**.**AEI_USP_ADDR_EIRQ_EN, CLSi_ARCH_EIRQ_EN**.**AEI_IM_ADDR_EIRQ_EN, and CLSi_ARCH_EIRQ_EN**.**AEI_USP_BE_EIRQ_EN) can be generated.

Any write access of value 1 to one of the flags CLSi_ARCH_IRQ_NOTIFY**.**AEI_USP_ADDR, CLSi_ARCH_IRQ_NOTIFY**.**AEI_USP_BE, CLSi_ARCH_IRQ_NOTIFY**.**AEI_IM_ADDR will clear the flags and the transaction status exception register CLSi_ARCH_AEI_STA_XPT.

Additional transaction status monitoring in a device with an AXI slave

The AXI slave is able to detect invalid transactions which cannot be operated correctly. In these cases, the AXIS module initiates an error signaling read transaction to the (E)GTM_AEI. The address in use will be FC_H. This address is an unimplemented address in the GTM_CTL. Any transaction to this address can trigger the (e)GTM_AEI_IRQ interrupt or store the exception in CLSi_ARCH_AEI_STA_XPT.

AEI bus system transaction length monitoring

GTM_CTL allows monitoring of transactions initiated from **eGTM** external bus masters (CPU, DMA,..). The monitoring takes place on the signals AEI_SEL, AEI_READY of the (E)GTM_AEI bridge master, the monitoring operates on the CLK clock. The transaction length monitoring can be enabled by writing CLSi_ARCH_CTRL**.**TO_VAL to a value unequal to 0. For each transaction the length will be counted in clock resolutions. The length timeout counter will increment on CLK as long as AEI_SEL = 1 and AEI_READY = 0. Each end of a transaction will reset the timeout counter. As soon as the length timeout counter is equal to CLSi_ARCH_CTRL**.**TO_VAL, one of the 4 possible reactions configurable with CLSi_ARCH_CTRL**.**TO_MODE will be triggered:

- b00: Observe once. On the first trigger, store the transaction information in CLSi_ARCH_AEI_ADDR_XPT. There is no impact on the ongoing transaction. Further triggers will be ignored.
- b01: Abort once. On the first trigger, store the transaction information in CLSi_ARCH_AEI_ADDR_XPT and immediately abort the ongoing transaction. The (E)GTM_AEI bridge master will abort the ongoing transaction by AEI_SEL = 0. In the transaction buffer, the access is terminated and the data and status are assumed to be 0. The subsequent triggers will not impact the transactions. Further triggers will be ignored.
- b10: Retry. On the first trigger, store the transaction information in CLSi_ARCH_AEI_ADDR_XPT and immediately stop and repeat the ongoing transaction. The (E)GTM_AEI bridge master will stop the ongoing transaction by applying AEI_SEL = 0 for 1 clock cycle, afterwards with AEI_SEL = 1 the transaction is continued. This will be applied for every occurring trigger.
- b11: Abort. On the first trigger, store the transaction information in CLSi_ARCH_AEI_ADDR_XPT and immediately abort the ongoing transaction. The (E)GTM_AEI bridge master will abort the ongoing transaction by AEI_SEL = 0. In the transaction buffer, the access is terminated and the data and status are assumed to be 0. This will be applied for every occurring trigger.

The first occurring trigger will store the transaction address AEI_ADDR in CLSi_ARCH_AEI_ADDR_XPT.TO_ADDR and the transaction type AEI_W1R0 in CLSi_ARCH_AEI_ADDR_XPT.TO_W1R0. Following timeout triggers will not change the content of CLSi_ARCH_AEI_ADDR_XPT. Furthermore, an individual interrupt unit (see **eGTM** interrupt concept) is implemented. This allows applications to receive an interrupt based on the transaction access length. The interrupt (e)GTM_AEI_IRQ or/and error interrupt (e)GTM_ERR_IRQ can be generated.

Any write access of value 1 to the flag CLSi_ARCH_IRQ_NOTIFY.AEI_TO_XPT will clear the flag and the transaction length monitoring register CLSi_ARCH_AEI_ADDR_XPT. This will re-enable the timeout trigger generation.

44.3.1.2.7 High-resolution PWM support

To realize a high-resolution PWM a suitable circuit is needed, e.g. a delay chain, which additionally delays the PWM output signal by a defined number of micro-steps.

This circuit is not part of the **eGTM**, but to control this circuit, **eGTM** delivers output ports (e)GTM_ATOM[i]_OUT_HRES[x] in addition to the PWM output ports (e)GTM_TOM[i]_OUT[x:x] and (e)GTM_ATOM[i]_OUT[x:x].

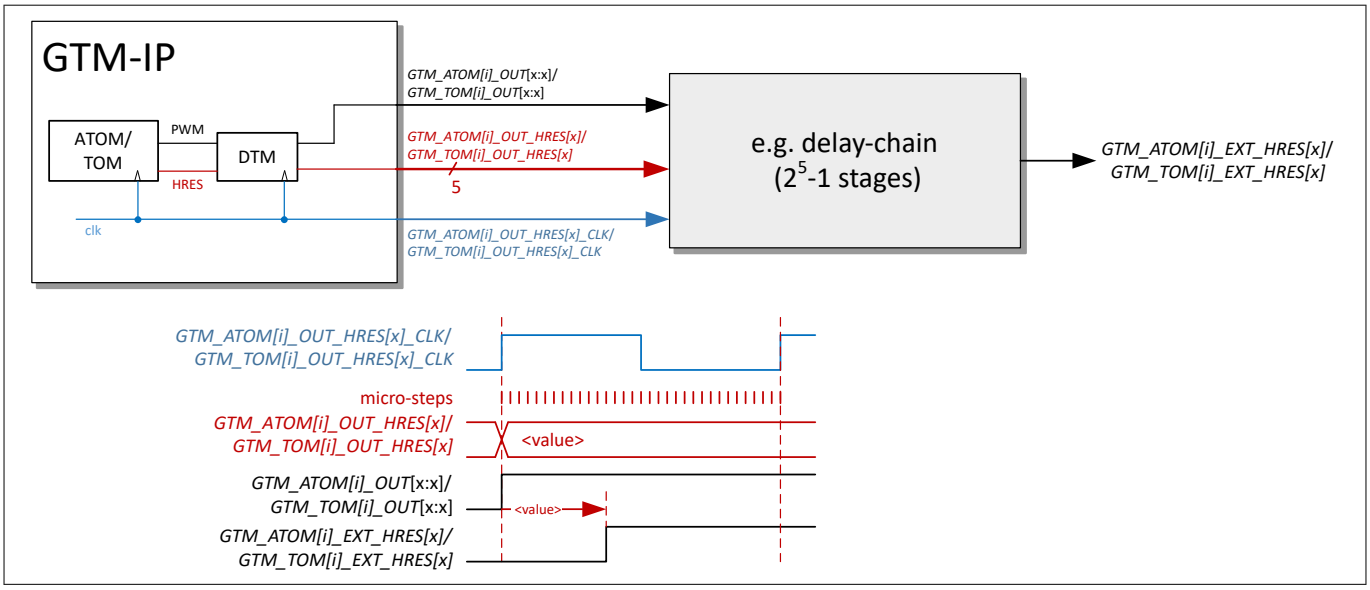


Figure 1023 High-resolution PWM support

The width of (e)GTM_ATOM[i]_OUT_HRES[x] is defined to 5. Therefore, the number of micro-steps is 2^5- 1 = 31.

The width of one micro-step depends on the clock period: micro-step$_{width}$ = CLK $_{period}$/2^5

A clock period of 5 ns (200 MHz) results in a micro-step width of: 5 ns / 2^5 = 156.25 ps.

The additional output signal (e)GTM_ATOM[i]_OUT_HRES[x] will be generated in **ATOM** sub-modules beside the PWM output signal (e)GTM_TOM[i]_OUT[x:x] and (e)GTM_ATOM[i]_OUT[x:x]. Afterwards the signal is connected through the **DTM** sub-module before being directed outside of **eGTM**.

Furthermore, the internal clock signal (clk) with which the **eGTM** output signals are generated will be delivered by **eGTM**.

The **DTM** sub-module provides the possibility to input dead-time on the output signal and the inverse output signal. Based on the high-resolution feature it is possible to adjust the dead-time length on micro-step level.

For detailed information about the high-resolution PWM support please refer to the chapter of the sub-modules **TOM**, **ATOM** and **DTM**.

44.3.1.2.8 Inter-cluster pulse adapter

Clusters can be configured to use different clock frequencies. For that, an additional control scheme (called here inter-cluster pulse adapter (ICPA)) is needed to coordinate the exchange of signals between two clusters. The concept of ICPA is demonstrated in Figure 1024.

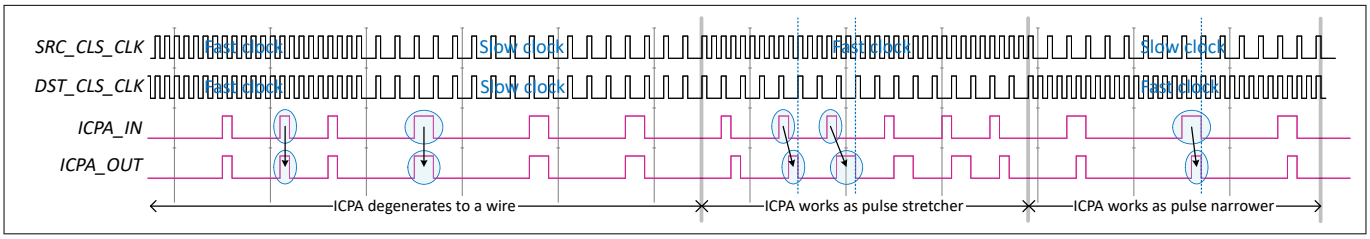


Figure 1024 Demonstration of the ICPA behavior handling pulses at different source and destination cluster frequencies

ICPA attempts to transform a pulse (of one cycle duration) generated in the source cluster in order to make it sampled exactly once in the destination cluster (regardless of the clocks used in both).

When the clocks in the two clusters are identical, the ICPA degenerates into a wire.

When a pulse is communicated from a cluster running at a slower clock rate to a faster clock rate, the ICPA narrows the pulse appropriately so that it is sampled once in the destination cluster.

When a pulse is communicated from a cluster running at a faster clock rate to a cluster running at a slower clock rate, the ICPA stretches it appropriately so that it is sampled once in the destination cluster.

It is worth mentioning that in the latter case, some pulses might get missed. This happens for example when the rate of pulse generation at the source cluster is higher than the clock rate in the destination cluster.

44.3.1.2.9 eGTM clock and time base management (CTBM)

Inside the **eGTM**, several sub-modules are involved in the clock and time base management of the whole **eGTM**. Figure 1025 shows the connections and sub-modules involved in these tasks. The sub-modules involved are called Clock and Time Base Management (CTBM) modules further on.

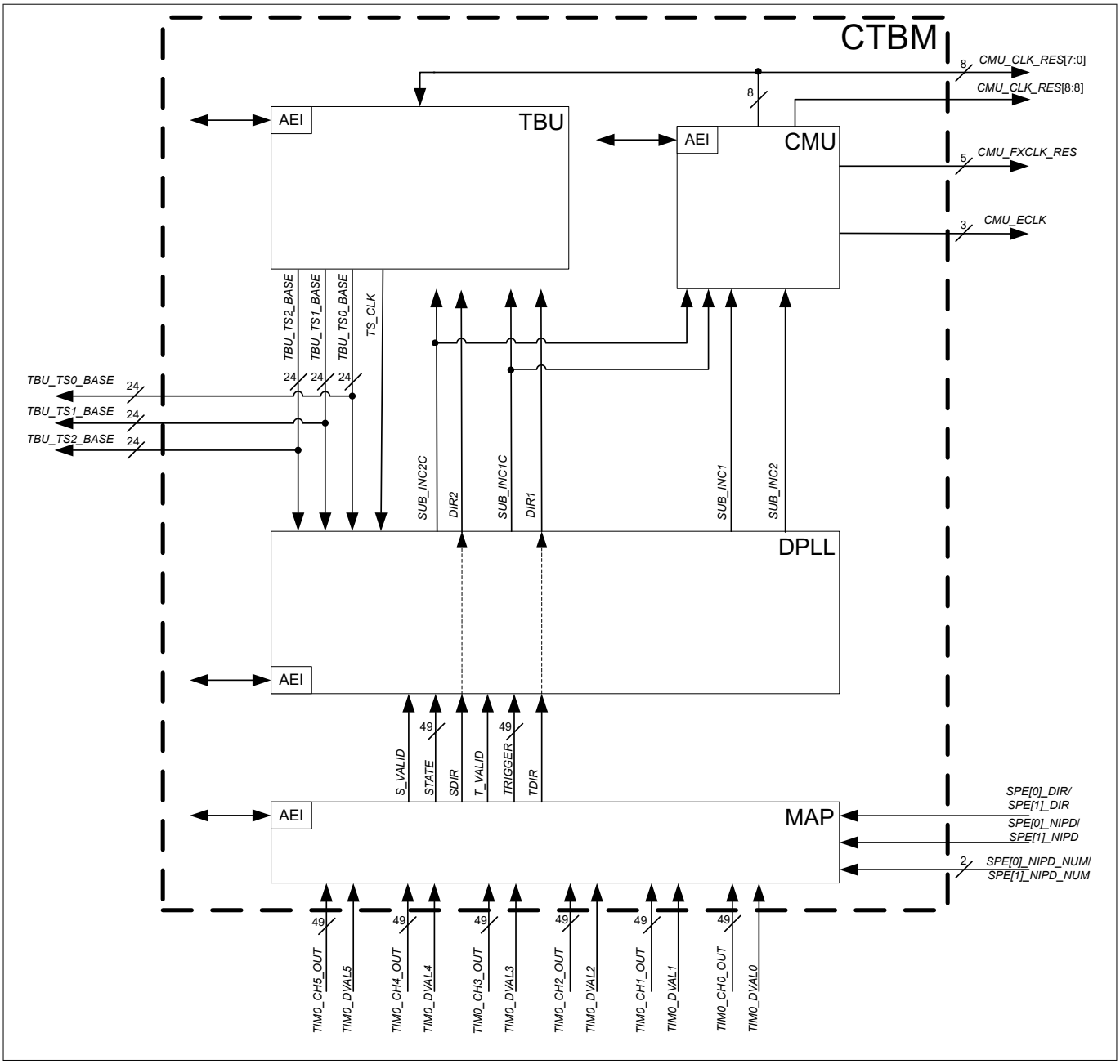


Figure 1025 eGTM clock and time base management architecture

***Note**: The figure is a general picture from the IP vendor. The exact configuration can be found in the device specific configuration.*

One important module of the CTBM is the Clock Management Unit (**CMU**) which generates up to 14 clock resolutions for the sub-modules of the **eGTM** and up to three **eGTM** external clocks. The clock resolutions are CMU_CLK_RES[x:x] and CMU_FXCLK_RES[y:y], where x ∈ {0, 1, ..., 8} and y ∈ {0, 1, ..., 4}. These signals are routed to all clusters via ICPAs. Within a cluster, they are used as clock enables regulating the resolution (or rate) at which actions are performed. The external clocks are CMU_ECLK[z:z], where z ∈ {0, 1, 2}. In contrast to CMU_CLK_RES[x:x], these are approximately 50% duty cycle clocks. For a detailed description of the **CMU** functionality and clocks please refer to Clock management unit (**CMU**).

The five CMU_FXCLK_RES[y:y] (y ∈ {0, 1, ..., 4}) clock resolutions are used by the **TOM** sub-module for PWM generation.

A maximum of nine CMU_CLK_RES[x:x] (x∈{0, 1,..., 8}) clocks are used by other sub-modules of the **eGTM** for signal generation.

Inside the Time Base Unit (**TBU**) one of CMU_CLK_RES[x:x] (x ∈ {0, 1, ..., 7}) clock resolutions is used per channel to generate a common time base for the **eGTM**. The **TBU** functionality is described in Time base unit (**TBU**).

The **TBU** sub-module generates the three time base signals TBU_TS0_BASE, TBU_TS1_BASE and TBU_TS2_BASE which are widely used inside the **eGTM** as common time bases for signal characterization and generation.

Besides the time base 1 and 2 which may represent a relative angle clock for an engine management system it is helpful to have an absolute angle clock for CPU internal angle algorithm calculations. This absolute angle clock is represented by the **TBU** base 3. The **TBU** channel 0 up to 2 are widely used inside the **eGTM** as common time (channel 0, 1 and/or 2) or angle (channel 1 and/or 2) bases for signal characterization and generation. The **TBU** channel 3 is only configurable and readable by CPU.

Cyclic event compare

With the time base module (**TBU**) the **eGTM** provides three counters, where the counter of **TBU** channel 0 represents a time and the counter of **TBU** channel 1 or 2 may represent a time.

Deciding if an event is to happen in the future (or has happened in the past) is vital in many applications. Using conventional relational operators to determine if an event has happened in the past (or is to happen in the future) is not reliable. The reason is attributed to the inevitable wrap-arounds associated with using limited word length representations for the time bases. In order to avoid this, cyclic event compares have been introduced in **eGTM** (see Figure 1026). The dynamic range of tbu ∈ {TBU_TS0_BASE, TBU_TS1_BASE, TBU_TS2_BASE} is represented as a circle with a diameter starting from the current value of tbu and splitting the circle into two regions (future and past). The two regions rotate with tbu (clockwise when up-counting and counter-clockwise when down-counting). Depending on the direction of counting and the time/position associated with an event, it can be readily determined if an event has happened in the past or is to happen in the future. The definitions for the cyclic compare operators are provided in the following algorithm:

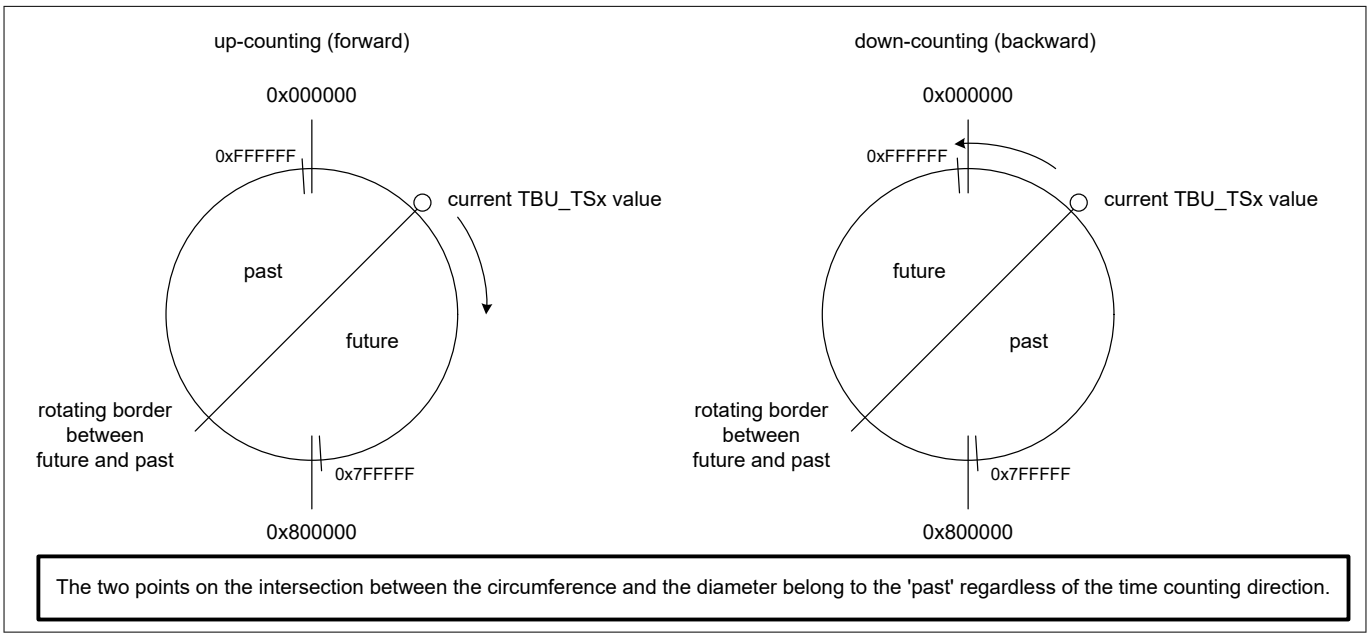


Figure 1026 Cyclic event counter representing time or angle

Inputs: tbu: 24−bit unsigned (time/angle value of the timebase); cmp: 24−bit unsigned (the time/position associated with an event); dir ∈ {FORWARD, BACKWARD}.

If (cmp = tbu or |tbu - cmp| = 2^{23})

tbu is (cyclically) >= cmp

Else
IF (tbu < cmp)
tbu_cmp = 2^{24} + tbu − cmp
ELSE
tbu_cmp = tbu − cmp
END IF
IF (tbu_cmp > 2^{23})
If (dir = FORWARD)
tbu is (cyclically) < cmp
Else
tbu is (cyclically) >= cmp
END IF
ELSE
If (dir = FORWARD)
tbu is (cyclically) >= cmp
Else
tbu is (cyclically) < cmp
END IF
END IF
END IF

An event is assumed to have happened in the past if tbu is 'cyclically greater than or equal to' the time associated with the event. Otherwise, the event is assumed to happen in the future.

Figure 1027 demonstrates how cyclic event compares in conjunction with the counting direction of tbu can be used to identify if an event has happened in the past or is to happen in the future. In the figure, both tbu and cmp are assumed to be 3-bit unsigned.

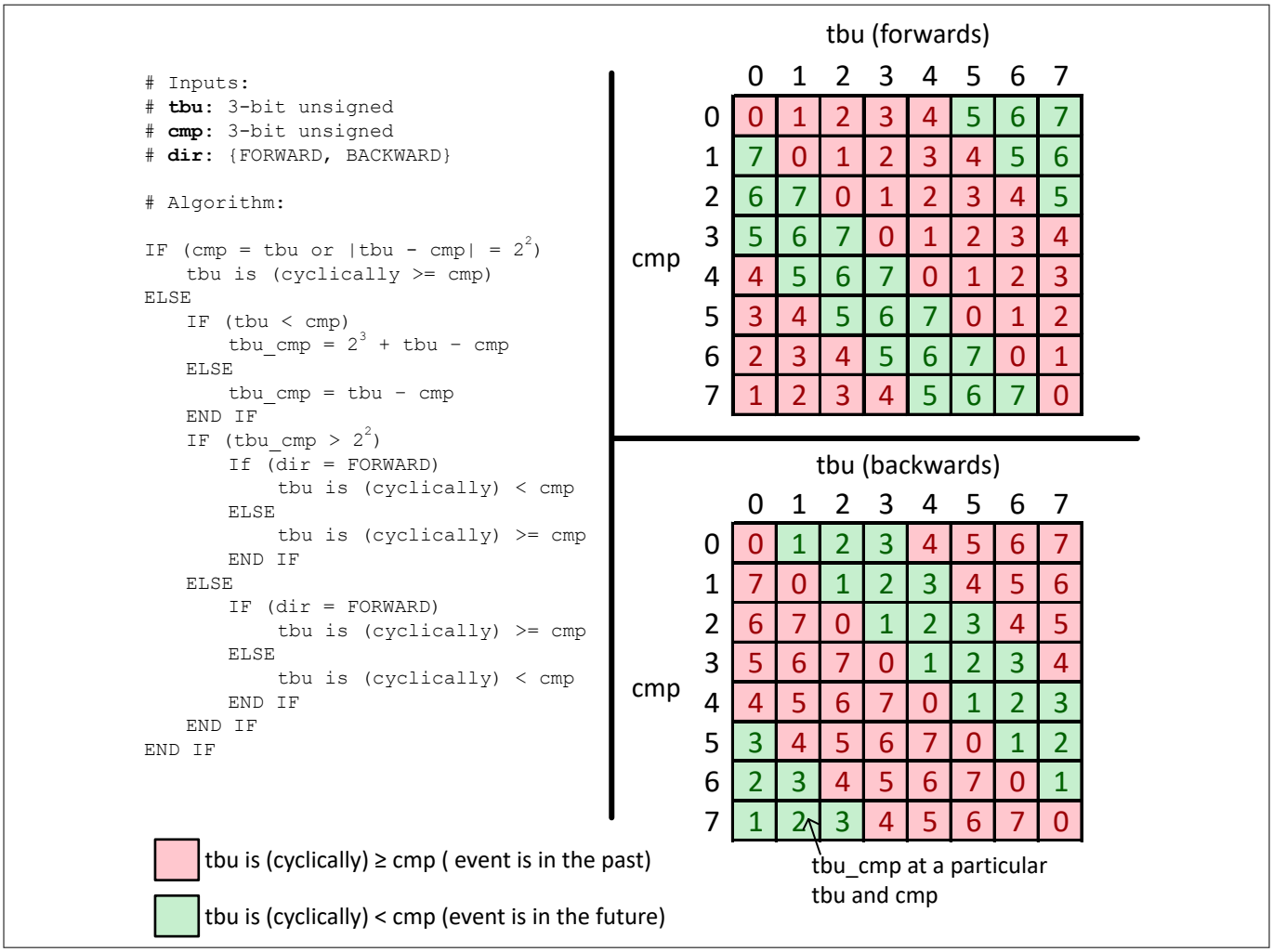


Figure 1027 Demonstration of cyclic event compares

By adapting the cyclic event compares and taking the counting direction of tbu (up-counting or down-counting) into account, it is possible to identify if an event is to happen in the future or has happened in the past reliably as long as |tbu-cmp| < 800000_H. Having a difference larger than that should be avoided as it erroneously pushes an event from 'past' to 'future' (or the other way around) without crossing 'now' (the current timestamp on tbu). For example, if an event is to be scheduled in the far future, cmp should be carefully selected such that the difference does not exceed $7FFFFF_H$. Having a difference larger than that will result in classifying the associated event as 'has happened in the past'.

44.3.1.2.10 eGTM interrupt concept

The sub-modules of the **eGTM** can generate thousands of interrupts on behalf of internal events. This high number of interrupts is combined inside the Interrupt Concentrator Module (**ICM**) into interrupt groups. In these interrupt groups the **eGTM** sub-module interrupt signals are bundled to a smaller set of interrupts. From these interrupt sets, a smaller number of interrupt signals is created and signaled outside of the **eGTM** on a signal with the name "GTM_<MOD>[i]_**IRQ**", where <MOD>[i] identifies the name of the corresponding **eGTM** sub-module.

In the following subchapters the interrupt signals are described on the basis of the **TIM** instance i. **TIM[i]** in the name of the signals can be replaced by the name of the other sub-modules of the **eGTM** and are valid for all following modules.

- **TOM[i]**
- **ATOM[i]**

- **SPE[i]**
- **CMP**

Moreover, each output signal (e)GTM_TIM[i]_IRQ has a corresponding input signal (e)GTM_TIM[i]_IRQ_CLR that can be used for clearing the interrupts. These input signals can be used by the surrounding microcontroller system as:

- Acknowledge signal from a DMA controller
- Validation signal from ADC
- Clear signal from a **eGTM**-external interrupt controller to do an atomic clear while entering an ISR routine

The individual interrupts are controlled inside the sub-modules. If a sub-module consists of several sub-module channels that are most likely to work independent from each other (like **TIM**, **TOM**, **ATOM**), each sub-module channel has its own interrupt control and status register set, named as interrupt set IRQ_SET= <MOD>[i]_CH[x:x] in the following. Other sub-modules (**SPE**, **CMP** and global **eGTM** functionality) have a common interrupt set IRQ_SET= <MOD>[i] for the whole sub-module.

From this point onwards (including the subsequent subsections), TIM[i]_CH *[x]* will be used in an exemplary way as a placeholder for:

- TOM[i]_CH[x]
- ATOM[i]_CH[x]
- **SPE[i]**
- **CMP**
- **eGTM**

The interrupt set consists of four registers: The CLSi_TIM_CHx_IRQ_EN register, the CLSi_TIM_CHx_IRQ_NOTIFY register, the CLSi_TIM_CHx_IRQ_FORCINT register, and the CLSi_TIM_CHx_IRQ_MODE register. While the registers CLSi_TIM_CHx_IRQ_EN, CLSi_TIM_CHx_IRQ_NOTIFY, and CLSi_TIM_CHx_IRQ_FORCINT signalize the status and allow controlling of each individual interrupt source within an interrupt set, the register CLSi_TIM_CHx_IRQ_MODE configures the interrupt mode that is applied to all interrupts that belong to the same interrupt set.

In order to support a wide variety of microcontroller architectures and interrupt systems with different interrupt signal output characteristics and internal interrupt handling the following four modes can be configured:

- Level mode
- Pulse mode
- Pulse-Notify mode
- Single-Pulse mode

These interrupt modes are described in more details in the following subsections.

The register CLSi_TIM_CHx_IRQ_EN allows the enabling and disabling of an individual interrupt within an interrupt set. Independent of the configured mode, only enabled interrupts can signalize an interrupt on their signal (e)GTM_TIM[i]_IRQ.

The register CLSi_TIM_CHx_IRQ_NOTIFY collects the occurrence of interrupt events. The behavior for setting a bit in this register depends on the configured mode and thus it is described later on in the mode descriptions.

Independent of the configured mode any write-access with value '1' to a bit in the register CLSi_TIM_CHx_IRQ_NOTIFY always clears the corresponding CLSi_TIM_CHx_IRQ_NOTIFY bit.

Moreover, the enabling of a disabled interrupt source with a write-access to the register CLSi_TIM_CHx_IRQ_EN also clears the corresponding bit in the CLSi_TIM_CHx_IRQ_NOTIFY register but only if the error interrupt source CLSi_TIM_CHx_EIRQ_EN is disabled. However, if the enabling of a disabled interrupt is simultaneous to an incoming interrupt event, the interrupt event is dominant and the register CLSi_TIM_CHx_IRQ_NOTIFY is not cleared.

Attention**: **Each write-access to** CLSi_TIM_CHx_IRQ_MODE **clears all bits in** CLSi_TIM_CHx_IRQ_NOTIFY**. Write-access to** CLSi_TIM_CHx_IRQ_MODE **shall only be executed if all interrupts of the corresponding channel are disabled. Otherwise, interrupt events can get lost or interrupt logic can go into an unpredictable state.

To secure a way for reconfiguring the interrupt mode of an interrupt set, proceed as follows:

1. Disable all interrupts via CLSi_TIM_CHx_IRQ_EN and all error interrupts via CLSi_TIM_CHx_EIRQ_EN
2. Define the new interrupt mode via CLSi_TIM_CHx_IRQ_MODE
3. Enable the desired interrupts via CLSi_TIM_CHx_IRQ_EN and desired error interrupts via CLSi_TIM_CHx_EIRQ_EN

The register CLSi_TIM_CHx_IRQ_FORCINT is used by software for triggering individual interrupts with a write-access with value '1'. Since a write-access to CLSi_TIM_CHx_IRQ_FORCINT only generates a single pulse, CLSi_TIM_CHx_IRQ_FORCINT is not implemented as a true register and thus any read-access to CLSi_TIM_CHx_IRQ_FORCINT always results with a value of '0'. A write-access with a one to particular bit field in this register shall force the corresponding bit field in CLSi_TIM_CHx_IRQ_NOTIFY to one.

The mechanism for triggering interrupts with CLSi_TIM_CHx_IRQ_FORCINT is globally disabled after reset. It has to be explicitly enabled by clearing the bit CLSi_ARCH_CTRL.RF_PROT.

For the modules **AEI**-bridge, **TIM**, **SPE** and **CMP** the interrupt set is extended by the register CLSi_TIM_CHx_EIRQ_EN. Each interrupt can be configured to trigger an error interrupt instead of the normal interrupt if it is enabled by the corresponding error interrupt enable bit in the CLSi_TIM_CHx_EIRQ_EN register.

It is possible for one source to enable the normal interrupt and the error interrupt in parallel. Because both interrupt clear signals can reset the notify bit, this is expected to cause problems in a system and it is strongly recommended not to enable both interrupt types at the same time.

Similar to enabling an interrupt, the enabling of a disabled error interrupt source with a write-access to the register CLSi_TIM_CHx_EIRQ_EN also clears the corresponding bit in the CLSi_TIM_CHx_IRQ_NOTIFY register only if the interrupt source CLSi_TIM_CHx_IRQ_EN is disabled. However, if the enabling of a disabled error interrupt is simultaneous to an incoming interrupt event, the interrupt event is dominant and the register CLSi_TIM_CHx_IRQ_NOTIFY is not cleared.

All enabled error interrupts are **OR**-combined inside the **ICM** and assigned to the dedicated **eGTM** port (e)GTM_ERR_IRQ. A corresponding input (e)GTM_ERR_IRQ_CLR allows the reset of this error interrupt from outside the **eGTM** (hardware clear).

To be able to detect the module source of the error interrupt the **ICM** provides the register CLSi_ICM_IRQG_MEI.

By evaluating the **ICM** register CLSi_ICM_IRQG_CEI1, the channel causing the error interrupt can be determined for the module **TIM**.

Level interrupt mode

The default interrupt mode is the Level Interrupt Mode. In this mode each occurred interrupt event is collected in the register CLSi_TIM_CHx_IRQ_NOTIFY, independent of the corresponding enable bit of register CLSi_TIM_CHx_IRQ_EN and CLSi_TIM_CHx_EIRQ_EN.

An interrupt event, which is defined as a pulse on the signal INT_OUT of Figure 1028, may be triggered by the interrupt source of the sub-module or by software performing a write-access to the corresponding register CLSi_TIM_CHx_IRQ_FORCINT, with a cleared bit CLSi_ARCH_CTRL.RF_PROT.

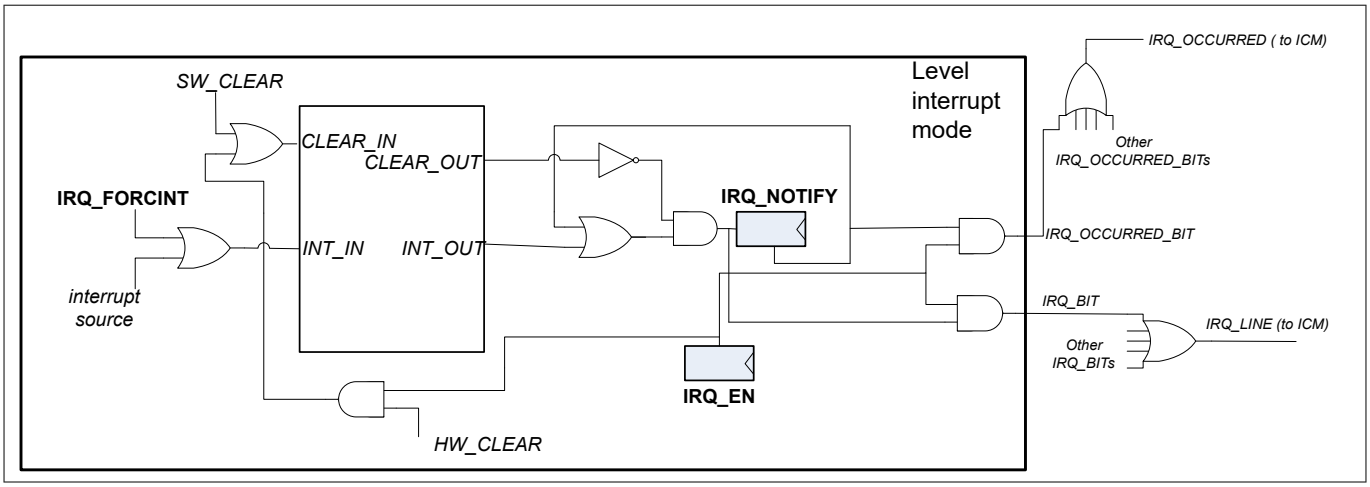


Figure 1028 Level interrupt mode scheme

A collected interrupt bit in register CLSi_TIM_CHx_IRQ_NOTIFY may be cleared by a clear event, which is defined as a pulse on signal CLEAR_OUT of Figure 1028. A clear event can be performed by writing '1' to the corresponding bit in the register CLSi_TIM_CHx_IRQ_NOTIFY leading to a pulse on signals SW_CLEAR. A clear event may also result from an externally connected signal (e)GTM_TIM[i]_IRQ_CLR, which is routed to the signal HW_CLEAR of Figure 1028. However, the hardware clear mechanism is only possible, if the corresponding interrupt is enabled by register CLSi_TIM_CHx_IRQ_EN.

As Table 2149 shows, interrupt events are dominant in the case of a simultaneous interrupt event and clear event.

Table 2149 Priority of interrupt events and clear events

INT_IN	CLEAR_IN	INT_OUT	CLEAR_OUT
0	0	0	0
0	1	0	1
1	0	1	0
1	1	1	0

As shown in Figure 1028 an occurred interrupt event is signaled as a constant signal level with value 1 to the signal IRQ_BIT, if the corresponding interrupt is enabled in register CLSi_TIM_CHx_IRQ_EN. Moreover, the enabling of a disabled interrupt source with a write-access to the register CLSi_TIM_CHx_IRQ_EN also clears the signal IRQ_BIT, but only if the error interrupt source CLSi_TIM_CHx_EIRQ_EN is disabled. However, if the enabling of a disabled interrupt is simultaneous to an incoming interrupt event, the interrupt event is dominant and signal IRQ_BIT is not cleared.

The signal IRQ_BIT is **OR**-combined with the neighboring IRQ_BIT signals of the same interrupt set and they are routed as a signal IRQ_LINE to the interrupt concentrator module (**ICM**). In some cases (sub-modules **TOM** and **ATOM**) the **ICM** may further **OR**-combine several IRQ_LINE signals to an outgoing interrupt signal (e)GTM_TIM[i]_IRQ. In the other cases the IRQ_LINE signals are directly connected to the outgoing signals (e)GTM_TIM[i]_IRQ, within the sub-module **ICM**.

The signal IRQ_OCCURRED is connected in a similar way as the signal IRQ_LINE, however this signal is used for monitoring the interrupt state of the register CLSi_TIM_CHx_IRQ_NOTIFY in the registers of the **ICM**.

The additional error interrupt enable mechanism for level interrupt is shown below.

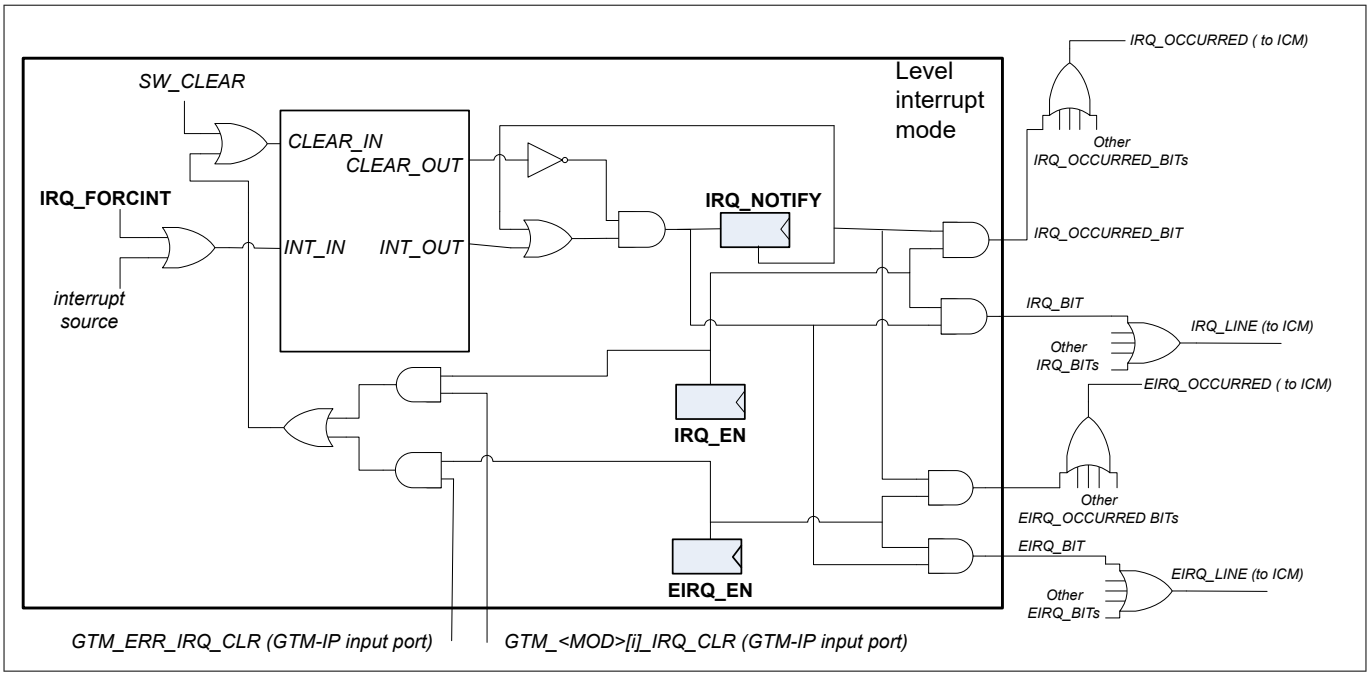


Figure 1029 Level interrupt scheme for modules AEI-bridge, TIM, SPE, CMP

A collected interrupt bit in register CLSi_TIM_CHx_IRQ_NOTIFY may be cleared by a clear event, which is defined as a pulse on signal CLEAR_OUT of Figure 1029. A clear event can be performed by writing '1' to the corresponding bit in the register CLSi_TIM_CHx_IRQ_NOTIFY leading to a pulse on signals SW_CLEAR. A clear event may also result from the externally connected signal (e)GTM_TIM[i]_IRQ_CLR or (e)GTM_ERR_IRQ_CLR, which is routed as an HW_CLEAR to CLEAR_IN of Figure 1029. However, the hardware clear mechanism is only possible, if the corresponding interrupt or error interrupt is enabled by register CLSi_TIM_CHx_IRQ_EN or CLSi_TIM_CHx_EIRQ_EN.

As it can be seen from the Figure 1029 an occurred interrupt event is signaled as a constant signal level with value 1 to the signal IRQ_BIT, if the corresponding interrupt is enabled in register CLSi_TIM_CHx_IRQ_EN.

Pulse interrupt mode

The Pulse interrupt mode behavior can be observed from Figure 1030.

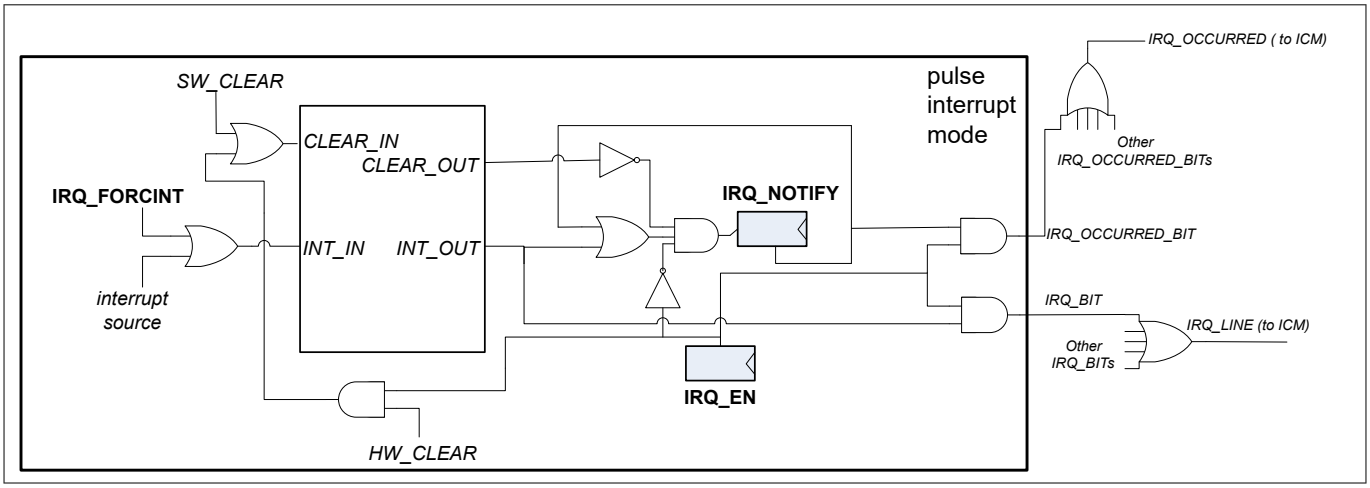


Figure 1030 Pulse interrupt mode scheme

In Pulse Interrupt Mode each Interrupt Event will generate a pulse on the IRQ_BIT signal if CLSi_TIM_CHx_IRQ_EN is enabled.

As it can be seen from the figure, the interrupt bit in CLSi_TIM_CHx_IRQ_NOTIFY register is always cleared if CLSi_TIM_CHx_IRQ_EN is enabled.

However, if an interrupt is disabled in the register CLSi_TIM_CHx_IRQ_EN, an occurred interrupt event is captured in the register CLSi_TIM_CHx_IRQ_NOTIFY, in order to allow polling for disabled interrupts by software.

Disabled interrupts may be cleared by an interrupt clear event.

In case of a simultaneous interrupt event and clear event, the interrupt events are dominant (see details Table 2149).

In Pulse Interrupt Mode, the signal IRQ_OCCURRED is always 0.

The additional error interrupt enable mechanism for pulse interrupt is shown below.

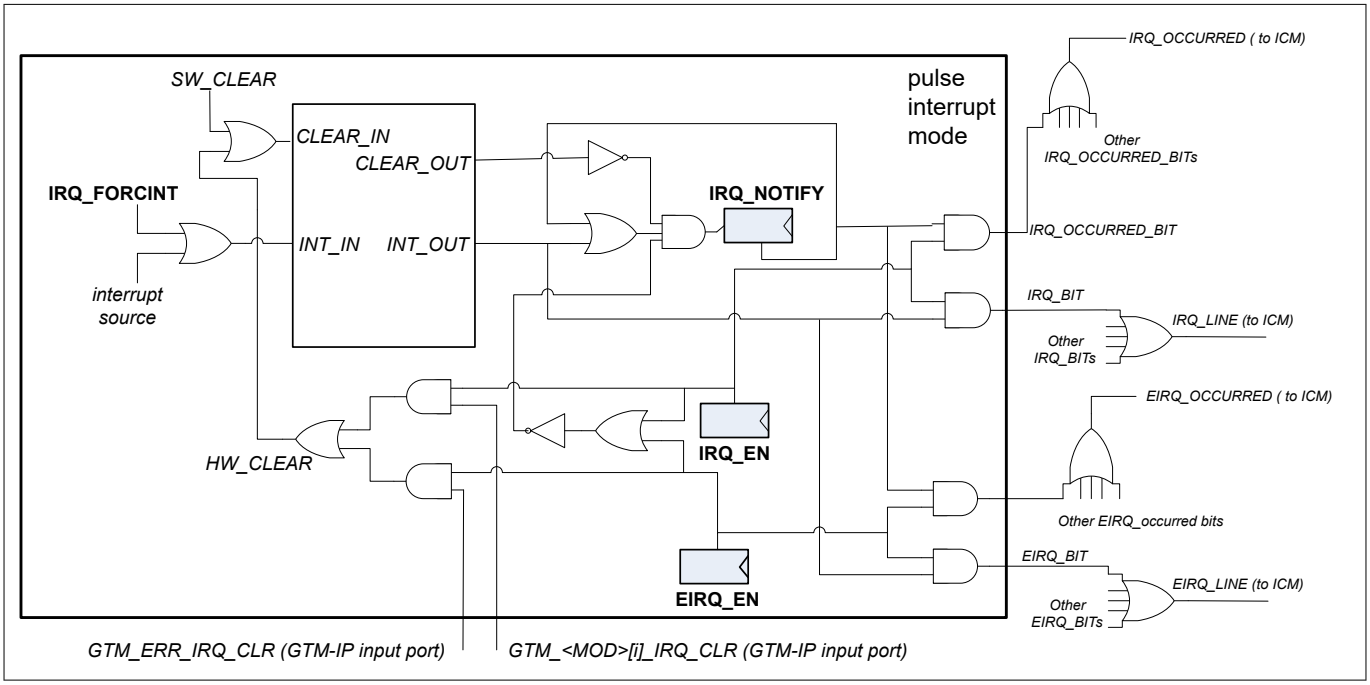


Figure 1031 Pulse interrupt scheme for modules AEI-bridge, TIM, SPE, CMP

In Pulse Interrupt Mode each Interrupt Event will generate a pulse on the EIRQ_BIT signal if CLSi_TIM_CHx_EIRQ_EN is enabled.

As it can be seen from the figure, the interrupt bit in CLSi_TIM_CHx_IRQ_NOTIFY register is always cleared if CLSi_TIM_CHx_EIRQ_EN or CLSi_TIM_CHx_IRQ_EN are enabled.

However, if an error interrupt is disabled in the register CLSi_TIM_CHx_EIRQ_EN, an occurred error interrupt event is captured in the register CLSi_TIM_CHx_IRQ_NOTIFY, in order to allow polling for disabled error interrupts by software.

Disabled error interrupts may be cleared by an error interrupt clear event.

In Pulse interrupt mode, the signal EIRQ_OCCURRED is always 0.

Pulse-notify interrupt mode

In Pulse-notify Interrupt Mode, all interrupt events are captured in the register CLSi_TIM_CHx_IRQ_NOTIFY. If an interrupt is enabled by the register CLSi_TIM_CHx_IRQ_EN, each interrupt event will also generate a pulse on the IRQ_BIT signal. The signal IRQ_OCCURRED will be high if interrupt is enabled in register CLSi_TIM_CHx_IRQ_EN and the corresponding bit of register CLSi_TIM_CHx_IRQ_NOTIFY is set. The Pulse-notify interrupt mode is shown in Figure 1032.

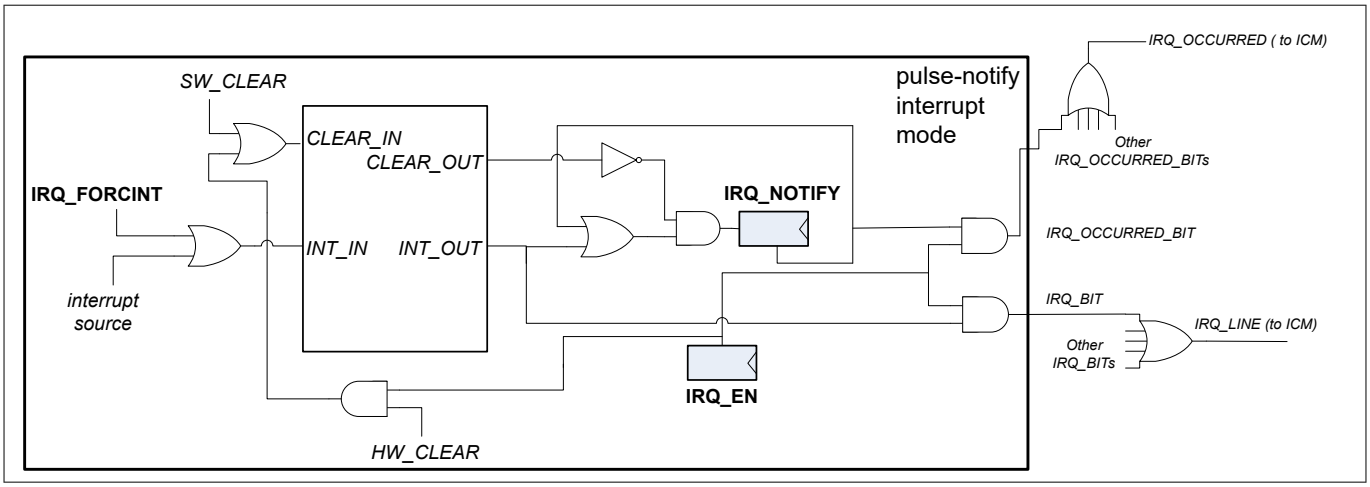


Figure 1032 Pulse-notify interrupt mode scheme

In case of a simultaneous interrupt event and clear event, the interrupt events are dominant (see details Table 2149).

The additional error interrupt enable mechanism for pulse-notify interrupt is shown below

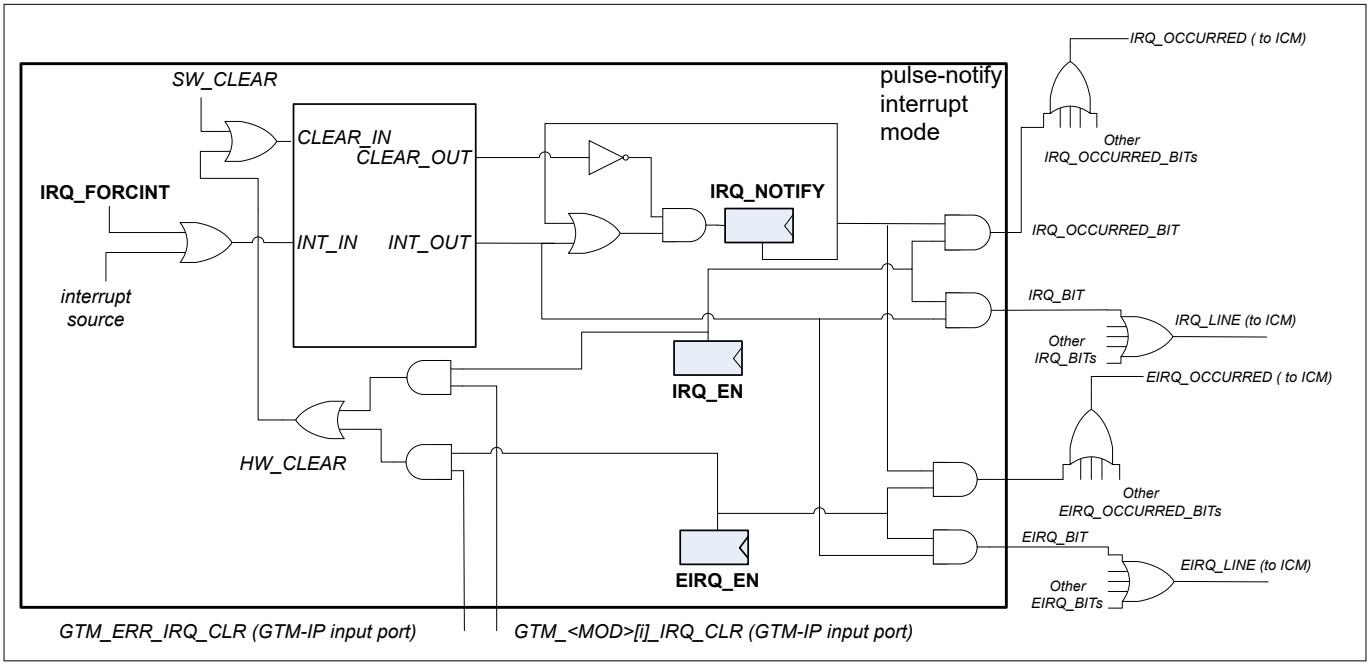


Figure 1033 Pulse-notify interrupt scheme for modules AEI-bridge, TIM, SPE, CMP

In Pulse-notify Interrupt mode, all error interrupt events are captured in the register CLSi_TIM_CHx_IRQ_NOTIFY. If an error interrupt is enabled by the register CLSi_TIM_CHx_EIRQ_EN, each error interrupt event will also generate a pulse on the EIRQ_BIT signal. The signal EIRQ_OCCURRED will be high if error interrupt is enabled in register CLSi_TIM_CHx_EIRQ_EN and the corresponding bit of register CLSi_TIM_CHx_IRQ_NOTIFY is set. The Pulse-notify interrupt mode for error interrupts is shown in Figure 1033.

Single-pulse interrupt mode

In Single-pulse Interrupt Mode, an interrupt event is always captured in the register CLSi_TIM_CHx_IRQ_NOTIFY, independent of the state of CLSi_TIM_CHx_IRQ_EN. However, only the first interrupt event of an enabled interrupt within a common interrupt set is forwarded to signal IRQ_LINE. Additional interrupt events of the same interrupt set cannot generate pulses on the signal IRQ_LINE, until the

corresponding bits in register CLSi_TIM_CHx_IRQ_NOTIFY of enabled interrupts are cleared by a clear event. This requirement is not valid when one of the following two cases is fulfilled. The first case is fulfilled if there is a simultaneous occurrence of interrupt event and an event on HW_CLEAR (or/and SW_CLEAR) input. The second case is fulfilled if an interrupt event occurs in the same clock-cycle in which the corresponding disabled interrupt is enabled and the corresponding bit in CLSi_TIM_CHx_IRQ_NOTIFY is at 1. That means, the interrupt event in these cases will result in setting of IRQ_LINE output, even when register CLSi_TIM_CHx_IRQ_NOTIFY was previously not cleared. The IRQ_OCCURRED signal line will be high, if the CLSi_TIM_CHx_IRQ_EN and the CLSi_TIM_CHx_IRQ_NOTIFY register bits are set. The single−pulse interrupt mode is shown in Figure 1034.

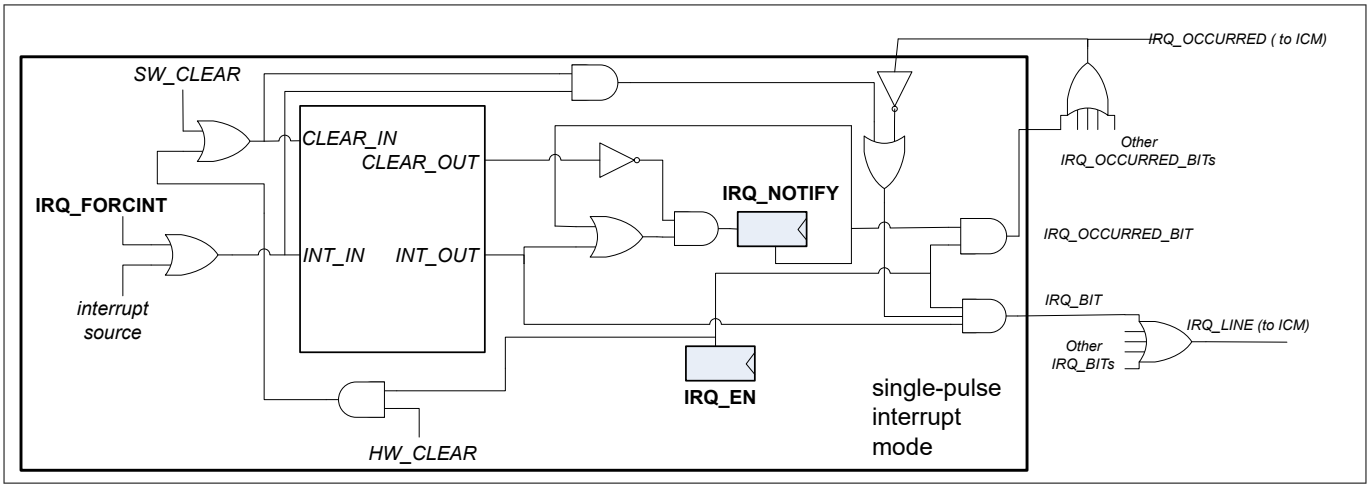


Figure 1034 Single-pulse interrupt mode scheme

In case of a simultaneous interrupt event and clear event, the interrupt events are dominant (see details Table 2149).

To avoid unexpected **IRQ** behavior in the single pulse mode, all desired interrupt sources should be enabled by a single write-access to CLSi_TIM_CHx_IRQ_EN and the notification bits should be cleared by a single write-access to the register CLSi_TIM_CHx_IRQ_NOTIFY.

The additional error interrupt enable mechanism for single-pulse interrupt is shown below

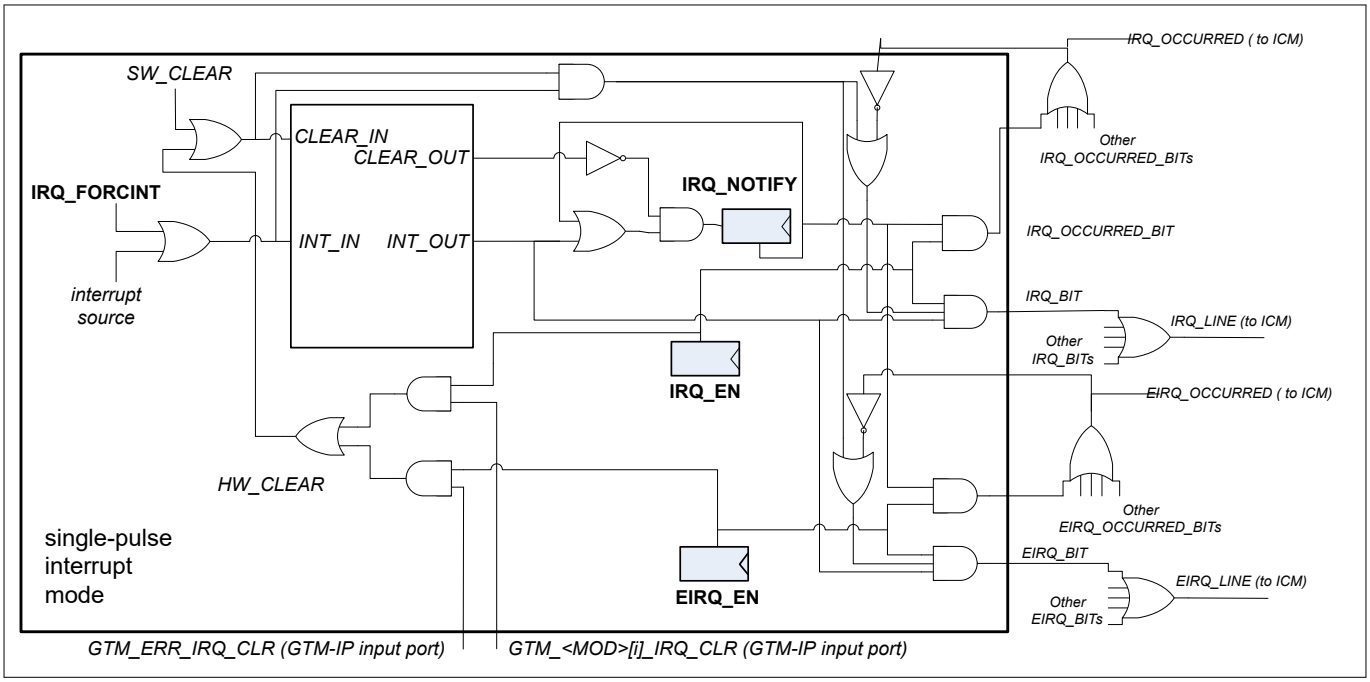


Figure 1035 Single-pulse interrupt scheme for modules AEI-bridge, TIM, SPE, CMP

In Single-pulse Interrupt Mode, an error interrupt event is always captured in the register CLSi_TIM_CHx_IRQ_NOTIFY, independent of the state of CLSi_TIM_CHx_EIRQ_EN. However, only the first error interrupt event of an enabled error interrupt within a common error interrupt set is forwarded to signal EIRQ_LINE. Additional error interrupt events of the same error interrupt set cannot generate pulses on the signal EIRQ_LINE, until the corresponding bits in register CLSi_TIM_CHx_IRQ_NOTIFY of enabled error interrupts are cleared by a clear event. The EIRQ_OCCURRED signal line will be high, if the CLSi_TIM_CHx_EIRQ_EN and the CLSi_TIM_CHx_IRQ_NOTIFY register bits are set. The Single-pulse interrupt mode for error interrupts is shown in Figure 1035.

To avoid unexpected EIRQ behavior in the single pulse mode, all desired error interrupt sources should be enabled by a single write-access to CLSi_TIM_CHx_EIRQ_EN and the notification bits should be cleared by a single write-access to the register CLSi_TIM_CHx_IRQ_NOTIFY.

eGTM interrupt concentration method

Because of the grouping of interrupts inside the **ICM**, it can be necessary for the software to access the **ICM** sub-module first to determine the interrupt set that is responsible for an interrupt. A second access to the responsible register CLSi_TIM_CHx_IRQ_NOTIFY is then necessary to identify the interrupt source, serve it and to reset the interrupt flag in register CLSi_TIM_CHx_IRQ_NOTIFY afterwards. The interrupt flags are never reset by an access to the **ICM**. For a detailed description of the **ICM** sub-module please refer to Interrupt concentrator module (**ICM**).

44.3.1.2.11 eGTM software debugger support

For software debugger support the **eGTM** comes with several features. E.g. status register bits must not be altered by a read-access from a software debugger. To avoid this behavior to reset a status register bit by software, the CPU has to write a '1' explicitly to the register bit to reset its content.

The Table 2150 describes the behavior of some **eGTM** registers with special functionality on behalf of read-accesses from the **AEI** bus interface.

Table 2150 Register behavior in case of software debugger accesses

Module	Register	Description
TIM	CLSi_TIM_CHx_GPR0 / CLSi_TIM_CHx_GPR1	The overflow bit is not altered in case of a debugger read-access to this register
TIM	CLSi_TIM_CHx_ECNT	While CLSi_TIM_CHx_CTRL.TIM_EN = 0, the register is not cleared in case of a debugger read-access to this register.
ATOM	CLSi_ATOM_CHx_SR0 / CLSi_ATOM_CHx_SR1	In SOMC mode a read-access to this register by the debugger does not release the channel for a new compare/match event.

The **eGTM** offers several debugging possibilities. For an exhaustive survey the user is referred to the module integration guide, and to the modules' respective chapters. It is up to the silicon vendor how to integrate/use these debugging possibilities and to which extent.

Upon the assertion of the (e)GTM_HALT_REQ signal the cluster clocks are disabled stalling the internal operations within the **eGTM**. Once the cluster clocks are disabled, (e)GTM_HALT_ACTIVE is asserted after a few clock-cycles and the debugger can perform read-/write-access over the **AEI** interface to all memory mapped

registers within the **eGTM**. When the external debugger de-asserts (e)GTM_HALT_REQ signal to mark the end of the debug accesses, the **eGTM** asserts (e)GTM_RESTORE for a few clock-cycles before it de-asserts it again. During the assertion of (e)GTM_RESTORE, the **eGTM** restores all transactions which were active on the internal **AEI** interface and were canceled when (e)GTM_HALT_REQ was asserted. This restoration phase is introduced to make sure, that the **eGTM** can resume working appropriately as if there was no debugger access at all.

In addition to enabling debug access to the memory addressable registers, the **eGTM** makes some important internal signals from different modules (**TIM**, **SPE**, **TOM**, **ATOM**, **TBU**) available to the outside. An external debugger can monitor such signals and, for example, trace them or halt the **eGTM** when the signals fulfill a predefined condition. Again, it is up to the silicon vendor to which extent to use these debugging possibilities.

44.3.1.2.12 eGTM programming conventions

To serve different application domains the **eGTM** is a highly configurable module with many configuration modes. In principle the sub-modules of the **eGTM** are intended to be configured at system startup to fulfill certain functionality for the application domain the microcontroller runs in.

For example, a **TIM** input channel can be used to monitor an application-specific external signal, and this signal has to be filtered. Therefore, the configuration of the **TIM** channel filter mode will be specific to the external signal characteristic. While it can be necessary to adapt the filter thresholds during runtime an adaptation of the filter mode during runtime is not reasonable. Thus, the change of the filter mode during runtime can lead to an unexpected behavior.

In general, the programmer has to be careful when reprogramming configuration registers of the **eGTM** sub-modules during runtime. It is recommended to disable the channels before reconfiguration takes place to avoid unexpected behavior of the **eGTM**.

44.3.1.2.13 eGTM Architecture Configuration Registers Description

Sub-chapter Architecture Configuration Registers Description

Table 2151

Register Shortname	Description
CLSi_ARCH_REV	eGTM version control register (i=0)
CLSi_ARCH_RST	eGTM global reset register (i=0)
CLSi_ARCH_CTRL	eGTM global control register (i=0)
CLSi_ARCH_CFG	eGTM configuration register (i=0)
CLSi_ARCH_AEI_ADDR_XPT	eGTM AEI timeout exception address register (i=0)
CLSi_ARCH_AEI_STA_XPT	eGTM AEI non-zero status register (i=0)
CLSi_ARCH_IRQ_NOTIFY	eGTM Interrupt notification register (i=0)
CLSi_ARCH_IRQ_EN	eGTM interrupt enable register (i=0)
CLSi_ARCH_EIRQ_EN	eGTM error interrupt enable register (i=0)
CLSi_ARCH_IRQ_FORCINT	eGTM Software interrupt generation register (i=0)
CLSi_ARCH_IRQ_MODE	eGTM toplevel interrupts mode selection (i=0)
CLSi_ARCH_CLK_CFG	eGTM Cluster Clock Configuration (i=0)
CLSi_AEI_BRIDGE_MODE	eGTM AEI bridge mode register (i=0)
CLSi_AEI_BRIDGE_PTR1	eGTM AEI bridge pointer 1 register (i=0)
CLSi_AEI_BRIDGE_PTR2	eGTM AEI bridge pointer 2 register (i=0)

44.3.1.2.14 eGTM toplevel port description

Table 2152 eGTM clock/reset/infrastructure interface

Name	Description
CLK	**eGTM** main clock
RESET	**eGTM** reset
SCAN_MODE	Signal to bypass reset_mux in scan mode
TEST_CLK	Testclock for clusters running on divided clock only; in use with scan_mode=1
CGATE_TE	Enable clock passing in clock gating cells for ATPG test purpose

Table 2153 eGTM clock enable interface

Name	Description
CLK_EN	**eGTM** main clock enable
(e)GTM_WDG_ERR	Watchdog error indicating an invalid CLK_EN

Table 2154 eGTM toplevel AEI interface

Name	Description
AEI_CLK	**AEI** interface clock
AEI_RESET	**AEI** interface reset (de-assertion synchronized to AEI_CLK)
AEI_SEL	Indicates active **AEI** transfer
AEI_PIPE	Indicates **AEI** protocol and valid address phase
AEI_ADDR	**AEI** access address
AEI_W1R0	**AEI** access type (read/write)
AEI_WDATA	**AEI** write data
AEI_RDATA	**AEI** read data
AEI_READY	**AEI** access ready
AEI_STATUS	**AEI** access status
AEI_SPLIT	Indicates **AEI** split transaction protocol type
AEI_RESPONSE_REQ	Indicates request for **AEI** response data
AEI_RESPONSE_READY	**AEI** response data ready
AEI_RESPONSE_TRANSACTION_CNT	Number of ready response transfers
AEI_FREE_BUFFER_CNT	Number of free transfer buffer entries
AEI_DEBUG_ACCESS	Sideband signal controlling debugger access

Table 2155 eGTM toplevel TIM[i] signal interface

Name	Description
(e)GTM_TIM[i]_IN	Input signals for **TIM[i]**

Table 2156 eGTM toplevel TOM[i] signal interface

Name	Description
(e)GTM_TOM[i]_OUT	Timer output signals for **TOM[i]**
(e)GTM_TOM[i]_OUT_N	Inverted timer output signals for **TOM[i]**

Table 2157 eGTM toplevel ATOM[i] signal interface

Name	Description
(e)GTM_ATOM[i]_OUT	Timer output signals for **ATOM[i]**
(e)GTM_ATOM[i]_OUT_N	Inverted timer output signals for **ATOM[i]**

Table 2158 eGTM toplevel ATOM[i] HRES output interface

Name	Description
(e)GTM_ATOM[i]_OUT_HRES[x]_CLK	Output signal (e)GTM_ATOM[i]_OUT *[x:x]* high-resolution clock
(e) (e)GTM_ATOM[i]_OUT_HRES[x]	Output signal (e)GTM_ATOM[i]_OUT *[x:x]* high-resolution delay value
(e)GTM_ATOM[i]_OUT_N_HRES[x]_CLK	Inverted output signal (e)GTM_ATOM[i]_OUT_N *[x:x]* high-resolution clock
(e)GTM_ATOM[i]_OUT_N_HRES[x]	Inverted output signal (e)GTM_ATOM[i]_OUT_N *[x:x]* high-resolution delay value

Table 2159 eGTM toplevel TIM[i] interrupt interface

Name	Description
(e)GTM_TIM[i]_IRQ	Shared interrupt from module **TIM[i]**
(e)GTM_TIM[i]_IRQ_CLR	Interrupt clear to module **TIM[i]**

Table 2160 eGTM toplevel TOM[i] interrupt interface

Name	Description
(e)GTM_TOM[i]_IRQ	Shared interrupt from module **TOM[i]**
(e)GTM_TOM[i]_IRQ_CLR	Interrupt clear to module **TOM[i]**

Table 2161 eGTM toplevel ATOM[i] interrupt interface

Name	Description
(e)GTM_ATOM[i]_IRQ	Shared interrupt from module **ATOM[i]**
(e)GTM_ATOM[i]_IRQ_CLR	Interrupt clear to module **ATOM[i]**

Table 2162 eGTM toplevel DTM[i] signal interface

Name	Description
(e)GTM_CDTM[i]_DTM[d]_AUX_IN	Input signals for **DTM[d]** in cluster [i].

Table 2163 (E)GTM_AEI interrupt interface

Name	Description
(e)GTM_AEI_IRQ	Shared interrupt from module GTM_CTL
(e)GTM_AEI_IRQ_CLR	Interrupt clear to module GTM_CTL

Table 2164 eGTM toplevel CMP interrupt interface

Name	Description
(e)GTM_CMP_IRQ	Shared interrupt from module **CMP**
(e)GTM_CMP_IRQ_CLR	Interrupt clear to module **CMP**

Table 2165 eGTM toplevel SPE[i] interrupt interface

Name	Description
(e)GTM_SPE[i]_IRQ	Shared interrupt from module **SPE[i]**
(e)GTM_SPE[i]_IRQ_CLR	Interrupt clear to module **SPE[i]**

Table 2166 eGTM toplevel ERR interrupt interface

Name	Description
(e)GTM_ERR_IRQ	Error interrupt from module **ICM**
(e)GTM_ERR_IRQ_CLR	Error interrupt clear to module **ICM**

Table 2167 eGTM toplevel CMU signal interface

Name	Description
(e)GTM_CMU_ECLK	Clock output signal from **CMU**

Table 2168 eGTM Top-Level Update Indicators

Name	Description
(e) (e)GTM_DBG_TBU_UP0	Indication of update debug port
(e) (e)GTM_DBG_TBU_UP1	Indication of update of TBU_TS1_BASE debug port
(e)GTM_DBG_TBU_UP2	Indication of update of TBU_TS2_BASE debug port
(e)GTM_DBG_TBU_UP3	Indication of update of debug port

Table 2169 eGTM Halt Interface

Name	Description
(e)GTM_HALT_REQ	Control signal which requests that **eGTM** operation has to be halted
(e)GTM_HALT_ACTIVE	Signal indicates that **eGTM** operation is halted
(e)GTM_RESTORE	Indicates restore phase of functional memory accesses when leaving **eGTM** halt state and continue with regular operation

***Note**: In this paragraph the signals are written as (e)GTM. Inside the original Bosch AE graphics, these signals are called GTM_ therefore the (e) is only in brackets.*

44.3.1.2.15 eGTM toplevel signal description

Table 2170 eGTM toplevel interrupt signals

Name	Description
AEI_IM_ADDR_IRQ	Interrupt signal **AEI** illegal Module address access
AEI_IM_ADDR_EIRQ	Error interrupt signal **AEI** illegal Module address access
AEI_TO_XPT_IRQ	Interrupt signal **AEI** timeout exception occurred.
AEI_TO_XPT_EIRQ	Error interrupt signal **AEI** timeout exception occurred.
AEI_USP_ADDR_IRQ	Interrupt signal **AEI** unsupported address
AEI_USP_ADDR_EIRQ	Error interrupt signal **AEI** unsupported address
AEI_USP_BE_IRQ	Interrupt signal **AEI** unsupported byte enable
AEI_USP_BE_EIRQ	Error interrupt signal **AEI** unsupported byte enable
CLK_EN_ERR_IRQ	Interrupt signal clock enable error
CLK_EN_ERR_EIRQ	Error interrupt signal clock enable error
CLK_PER_ERR_IRQ	Interrupt signal clock period error
CLK_PER_ERR_EIRQ	Error interrupt signal clock period error

Table 2171 CLS[j]_CLK_ENABLE

Name	Description
CLS[j]_CLK_ENABLE	Cluster [j] clocks are enabled; physical net gtm_core_i/gtm_cls[j]_i/en_clk_zz
CLS[j]_CLK	Cluster [j] clock signal

Table 2172 CLK_EN

Name	Description
INT_CLK_EN	Internal clock enable signal

Table 2173 RF_PROT AND RESETS

Name	Description
RF_PROT	rf_prot signal; physical net gtm_core_i/rf_prot_zz
HW_RESET	**eGTM** global reset active signal for clk clock domain; physical net gtm_core_i/reset_int_zz
(e)GTM_RESET	**eGTM** global reset active signal; physical net gtm_core_i/reset_int_zz
HW_CPU_IF_RESET	Reset active signal for CPU interface clock domain; physical net
(e)GTM_AEI_RESET	Software reset of gtm_aei bridge; physical net

Table 2174 CLS[j]_AEI_ARB

Name	Description
CLS[j]_AEI_ARB_STATE	Cluster [j] **AEI** bus arbiter state: 1=CPU
CLS[j]_AEI_ARB_WDATA	Cluster [j] **AEI** bus write data of arbiter; physical net gtm_core_i/gtm_cls[j]_i/arb_aei_wdata_zz
CLS[j]_CPU_AEI_WDATA	Cluster [j] **AEI** CPU bus write data; physical net gtm_core_i/gtm_cls[j]_i/aei_wdata_cls
CLS_CPU_AEI_SEL[j]	Cluster [j] CPU bus read-/write-access active; physical net

Table 2175 Toplevel internal signals

Name	Description
TIM[i]_AUX_IN	Selected source for module **TIM[i]** AUX_IN
TIM[i]_EXT_CAPTURE	Signal vector: driven by output port TIM_EXT_CAPTURE of **TIM** instance i, channel x:={0, 1,..., 7}

Table 2176 Interrupt unit internal signals

Name	Description
HW_CLEAR	Interrupt unit: Hardware request to clear any registered interrupt
SW_CLEAR	Interrupt unit: Software request to clear any registered interrupt
IRQ_BIT	Interrupt unit: Interrupt output signal, behaving as defined by corresponding **IRQ** mode
IRQ_OCCURRED_BIT	Interrupt unit: Interrupt output signal, 1 is signaling that an interrupt occurred
IRQ_LINE	Interrupt unit: Bundled output signal which combines multiple IRQ_BIT signals
IRQ_OCCURRED	Interrupt unit: Bundled output signal which combines multiple IRQ_OCCURRED_BIT signals
INT_IN	Interrupt priority decoding: Interrupt event input
CLEAR_IN	Interrupt priority decoding: Interrupt clear event input
INT_OUT	Interrupt priority decoding: Interrupt event output
CLEAR_OUT	Interrupt priority decoding: Interrupt clear event output

Table 2177 Error Interrupt Unit Internal Signals

Name	Description
EIRQ_BIT	Error interrupt unit: Interrupt output signal, behaving as defined by corresponding **IRQ** mode
EIRQ_OCCURRED_BIT	Error interrupt unit: Interrupt output signal, 1 is signaling that an interrupt occurred

(table continues...)

Table 2177 (continued) Error Interrupt Unit Internal Signals

Name	Description
EIRQ_LINE	Error interrupt unit: Bundled output signal which combines multiple EIRQ_BIT signals
EIRQ_OCCURRED	Error interrupt unit: Bundled output signal which combines multiple EIRQ_OCCURRED_BIT signals

44.3.1.3 Clock management unit (CMU)

44.3.1.3.1 Overview

The Clock Management Unit (**CMU**) only exists in cluster 0 and is tasked with the generation of 14 clock resolution signals and 3 **eGTM** external clocks. The clock resolution signals are routed to all clusters via ICPAs. Within a cluster, they are used as clock enables regulating the resolution (or rate) at which actions are performed. The 3 **eGTM** external clocks are approximately true 50% duty cycle clocks. The primary clock source used in this module is the cluster 0 clock signal CLS[j]_CLK. The cluster clock (CLS[j]_CLK) can be turned off, or use the **eGTM** clock, or use the **eGTM** clock halved depending on the configuration of CLSi_ARCH_CLK_CFG**.**CLSj_CLK_DIV (j=0-2) **(j=0)**. The clock resolution signals and the external clocks are generated with reference to it. Figure 1036 shows a block diagram of the **CMU**.

Indices and their range as used inside this chapter are:

- x:={0, 1,..., 7} index for clock resolution channel CMU_CLK_RES[x:x]
- y:={0, 1,..., 4} index for fixed clock for CMU_FXCLK_RES[y:y]
- z:={0, 1, 2} index for external clocks CMU_ECLK[z:z]

The Configurable Resolution Generation (CRG) sub-unit generates eight dedicated clock resolutions (CMU_CLK_RES[x:x], where x ∈ {0, 1, ..., 7}) for the following **eGTM** modules: **TIM**, **ATOM**, **TBU**, **CDTM** and **MON**. Every resolution signal has its own configuration register (CLSi_CMU_CLKx_CTRL) to configure its clock resolution generator. The input to a clock resolution generator can either be CMU_ECLK1_EN or CMU_GCLK_EN depending on the configuration of CLSi_CMU_CLK_CTRL**.**CLKx_EXT_DIVIDER (x=0-7).

The Fixed Clock Resolution Generation (FCRG) sub-unit generates predefined non-configurable clock resolutions CMU_FXCLK_RES[y:y] (y ∈ {0, 1, ..., 4}) for the **TOM**, **CDTM**, and **MON** modules. In the sub-unit there is a multiplexer that can be configured to select CMU_GCLK_EN or one of the eight CMU_CLK_RES[x:x] (x ∈ {0, 1, ..., 7}) as input to the FCR generators within. The dividing factors for a CMU_FXCLK_RES[y:y] clock resolution is non-configurable and is fixed to 2^{4*y}.

The External Clock Generation (EGU) sub-unit is able to generate up to three chip external clock signals visible at CMU_ECLK[z:z] (z ∈ {0, 1, 2}) with a duty cycle of approximately 50%.

The External Clock Generation (EGU) sub-unit is able to generate clock CMU_CLK_RES[8:8] for module **CCM** to manage 2 clock domains.

The clock source signals CMU_CLK_RES[x:x] (x∈{0, 1,..., 7}) and CMU_FXCLK_RES[y:y] are implemented in the form of enable signals for the corresponding registers, which means that the actual clock signal of all registers always use CLS[j]_CLK signal.

The four configurable clock resolution signals CMU_CLK_RES[0:0], CMU_CLK_RES[1:1], CMU_CLK_RES[6:6] and CMU_CLK_RES[7:7] are used for the **TIM** filter counters.

Although the clock resolutions and external clocks (*CMU_CLK_EN*, CMU_FXCLK_RES, CMU_ECLK) are run-time configurable, any change during run-time may result in an unpredictable behavior in the **eGTM**. It is strongly recommended to configure them only during initialization phase.

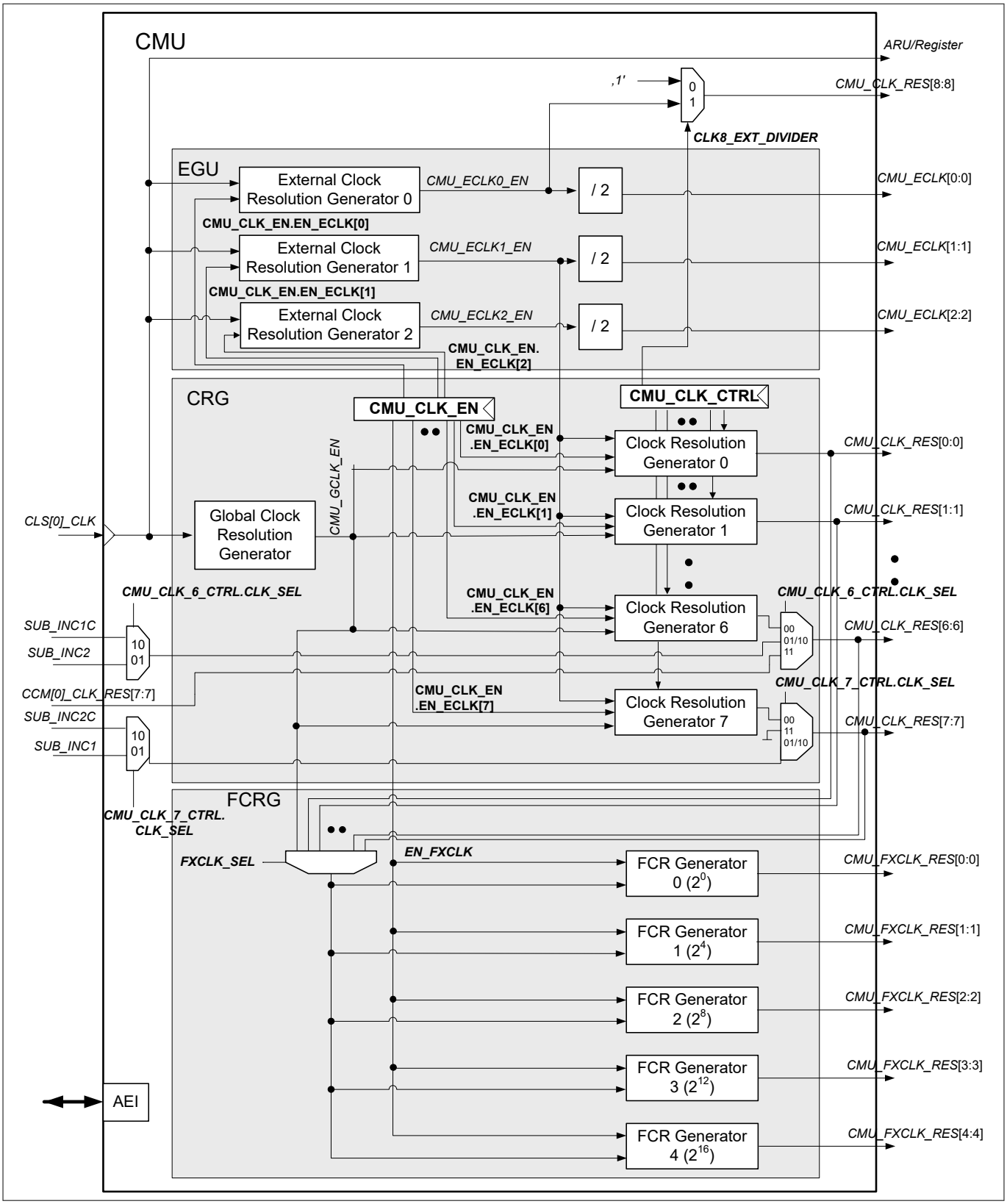


Figure 1036 CMU block diagram

***Note**: The figure is a general picture from the IP vendor. The exact configuration can be found in the device specific configuration.*

44.3.1.3.2 Global clock resolution generator

The subblock Global Clock Resolution Generator can be used to generate a common clock resolution signal (CMU_GCLK_EN) with reference to cluster 0 clock (CLS[j]_CLK).

The common clock resolution signal is generated by the Global Clock Resolution Generator. Depending on the configuration of the numerator CLSi_CMU_GCLK_NUM.GCLK_NUM and the denominator CLSi_CMU_GCLK_DEN.GCLK_DEN, the Global Clock Resolution Generator attempts to assert CMU_GCLK_EN for CLSi_CMU_GCLK_DEN.GCLK_DEN clock-cycles distributed at regular intervals (when possible) in every CLSi_CMU_GCLK_NUM.GCLK_NUM clock-cycles. Depending on the configuration, the ratio between the numerator to the denominator might result in a fractional number leading to the introduction of some irregularities in these intervals.

In theory, if the common clock resolution signal is used as a clock enable in conjunction with the cluster clock, it is possible to regulate the effective clock frequency (denoted as $1/T_{CMU_GCLK_EN}$) at which the registers are clocked. The effective clock frequency can range between 0% (when CMU_GCLK_EN is low all the time) and 100% (when CMU_GCLK_EN is high all the time) of the cluster clock frequency depending on the configuration. It is important to emphasize that $T_{CMU_GCLK_EN}$ is only a period of a virtual clock whose frequency is equivalent to the effective clock frequency resulting from using CMU_GCLK_EN as a clock enable to the registers the cluster clock is connected to. The following algorithm describes how to achieve a particular $T_{CMU_GCLK_EN} = (Z/N) * T_{CLS[j]_CLK}$, where Z = CLSi_CMU_GCLK_NUM.GCLK_NUM; N = CLSi_CMU_GCLK_DEN.GCLK_DEN; Z>= N > 0:

1. INIT-phase: set remainder (R), operand1 (OP1) and operand2 (OP2) register during INIT-phase (with implicit conversion to signed): R=Z, OP1=N, O-P2=N-Z.
2. After leaving INIT-phase (at least one CMU_CLK_RES[x:x] has been enabled) the sign of remainder R for each CLS[j]_CLK cycle is checked:
3. If R>=0: keep CMU_GCLK_EN='0', and update R=R-OP1.
4. If R<0: set CMU_GCLK_EN='1', and update R=R-OP2.

After at most (Z/N+1) subtractions (3. in the list above) there will be a negative R and an active phase of the generated clock enable (for one cycle) will be triggered (4. in the list above). The remainder R is a measure for the distance to a real Z/N clock and will be considered for the next generated clock enable cycle phase. The new R value will be R=R+(Z-N). In the worst case, the remainder R will sum up to an additional cycle in the generated clock enable period after Z-cycles. In other cases, equally distributed additional cycles will be inserted for the generated clock enable. If Z is an integer multiple of N, no additional cycles will be included for the generated clock enable at all.

***Note**: For a better resource sharing all arithmetic has been reduced to subtractions and the initialization of the remainder R uses the complement of (Z-N).*

44.3.1.3.3 Configurable clock resolution generation (CRG)

The CRG sub-unit generates up to eight configurable clock resolution signals with reference to an input signal. The input signal can either be the common clock resolution signal CMU_GCLK_EN or CMU_ECLK1_EN depending on the configuration of CLSi_CMU_CLK_CTRL.CLKx_EXT_DIVIDER (x=0-7).

Each clock resolution signal CMU_CLK_RES[x:x] has its own configuration register CLSi_CMU_CLKx_CTRL.

The value specified in CLSi_CMU_CLKx_CTRL.CLK_CNT determines how many pulses (of one CLS[j]_CLK clock-cycle duration) from the input signal are suppressed before one is wired to CMU_CLK_RES[x:x].

When a CMU_CLK_RES[x:x] is used as a clock enable in conjunction with cluster clock, it is possible to regulate the effective clock frequency (denoted as $f_{CMU_CLK_RES}[x:x] = 1/T_{CMU_CLK_RES}[x:x]$) at which the registers are clocked. It is important to emphasize that $T_{CMU_CLK_RES}[x:x]$ is only a period of a virtual clock whose frequency is equivalent to the effective clock frequency resulting from using CMU_CLK_RES[x:x] as a clock enable to the registers the cluster clock is connected to. From that, $T_{CMU_CLK_RES}[x:x]$ can be expressed as: $T_{CMU_CLK_RES}[x:x] =$

(CLSi_CMU_CLKx_CTRL.CLK_CNT +1) * $T_{CMU_GCLK_EN}$. Depending on the configuration of CLSi_CMU_CLK_CTRL.CLKx_EXT_DIVIDER (x=0-7), $T_{CMU_GCLK_EN}$ can be substituted with $T_{CMU_ECLK1_EN}$. The corresponding waveform is shown in Figure 1037

Each clock resolution signal CMU_CLK_RES[z:z] (z ∈ {0, 1, ..., 5} can be enabled individually by appropriately configuring the corresponding CLSi_CMU_CLK_EN.EN_CLKx (x=0-7) **(x=z)** bit field. Enabling of CMU_CLK_RES[6:6] is only possible if the bit field CLSi_CMU_CLK_EN.EN_CLKx (x=0-7) **(x=6)** is appropriately configured and the bit field CLSi_CMU_CLK_6_CTRL.CLK_SEL is set to 0. Similarly, enabling of CMU_CLK_RES[7:7] is only possible if the bit field CLSi_CMU_CLK_EN.EN_CLKx (x=0-7) **(x=7)** is appropriately configured and the bit field CLSi_CMU_CLK_7_CTRL.CLK_SEL is set to 0.

The clock resolution CMU_CLK_RES[8:8] can be configured by CLSi_CMU_CLK_CTRL.CLKx_EXT_DIVIDER (x=0-7) **(x=8)** to select between a constant '1' and CMU_ECLK0_EN.

The input to a clock source divider can either be CMU_ECLK1_EN or CMU_GCLK_EN depending on the configuration of CLSi_CMU_CLK_CTRL.CLKx_EXT_DIVIDER (x=0-7).

To avoid unexpected behavior of the hardware, the configuration of register CLSi_CMU_CLKx_CTRL and CLSi_CMU_CLK_CTRL can only be changed, when the corresponding clock resolution signal CMU_CLK_RES[x:x] and CMU_ECLK[1:1] are disabled.

Further, any changes to the registers CLSi_CMU_GCLK_NUM and CLSi_CMU_GCLK_DEN can only be performed, when all clock resolution signals (CMU_CLK_RES[x:x] and CMU_FXCLK_RES[y:y]) are disabled.

The clock resolution signals CMU_CLK_RES[x:x] and CMU_FXCLK_RES[y:y] are implemented in the form of enable signals for the corresponding registers, which means that the actual clock signal of all registers always use the CLS[j]_CLK signal.

The hardware guarantees that all clock resolution signals CMU_CLK_RES[x:x], which are enabled simultaneously, remain synchronous to each other. Synchronous enabling here means that the bit fields CLSi_CMU_CLK_EN.EN_CLKx (x=0-7) in the register CLSi_CMU_CLK_EN are set by the same write-access.

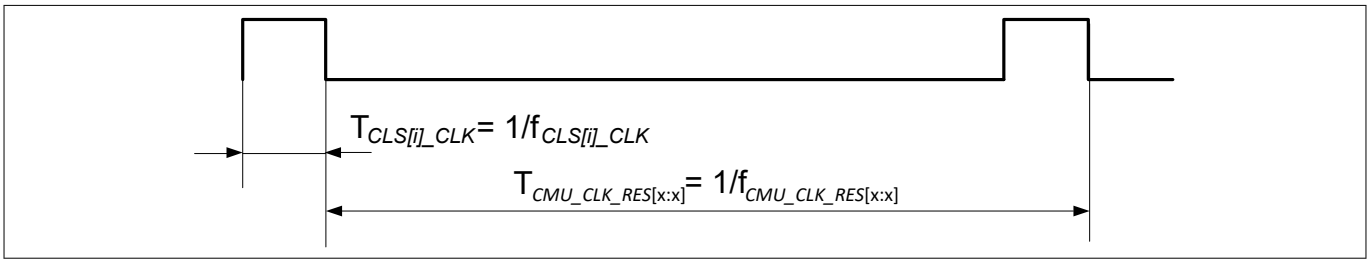


Figure 1037 Wave form of generated clock signal

44.3.1.3.4 Fixed clock resolution generation (FCRG)

The FCRG sub-unit generates fixed clock resolutions signals based on the CMU_GCLK_EN signal or one of the eight CMU_CLK_RES[x:x] resolution signals. The selection of which signal is configurable via CLSi_CMU_FXCLK_CTRL.FXCLK_SEL bit field. These clock resolutions are used for the PWM generation inside the **TOM** modules.

All clock resolutions CMU_FXCLK_RES[y:y] can be enabled/disabled simultaneously by appropriately configuring bit field CLSi_CMU_CLK_EN.EN_FXCLK.

The dividing factors are defined as 2^0, 2^4, 2^8, 2^{12}, and 2^{16}. An FCR Generator with a dividing factor configured as 2^q shall only send one pulse (of one CLS[j]_CLK clock-cycle) from every 2^q pulses to the output. The remaining pulses (2^q-1) are suppressed.

44.3.1.3.5 External generation unit (EGU)

The EGU sub-unit generates up to three separate clock output signals CMU_ECLK[z:z].

Each of these clock signals is derived from the corresponding External Clock Resolution Generator subblock, which generates a clock signal derived from the **eGTM** input clock CLS[j]_CLK.

In contrast to the signals CMU_CLK_RES[x:x] and CMU_FXCLK_RES[y:y], which are clock resolution signals effectively regulating the clock frequency when used as clock enables in conjunction with the cluster clock, the signals CMU_ECLK[z:z] have a duty cycle of approximately 50%. A CMU_CLK_RES[x:x] can be used as a true clock signal for external peripheral components.

The signal CMU_ECLK1_EN could be selected (instead of CMU_GCLK_EN) as input for the clock resolution generators to be used as a reference in the generation of the CMU_CLK_RES[x:x] signals by appropriately configuring CLSi_CMU_CLK_CTRL**.**CLKx_EXT_DIVIDER (x=0-7). In addition, CMU_ECLK0_EN could be wired (instead of '1') to the all-time enabled CMU_CLK_RES[8:8] by appropriately configuring CLSi_CMU_CLK_CTRL**.**CLKx_EXT_DIVIDER (x=0-7) **(x=8)**.

Each of the external clock dividers is enabled/disabled by appropriately configuring bit field CLSi_CMU_CLK_EN**.**EN_ECLKz (z=0-2).

Similar to 'Global Clock Resolution Generator', an 'External Clock Resolution Generator' generates a clock resolution signal (let CMU_ECLK_EN ∈ {CMU_ECLK0_EN, CMU_ECLK1_EN, CMU_ECLK2_EN}). In theory, if a clock resolution signal CMU_ECLK_EN is used as a clock enable in conjunction with the cluster clock, it is possible to regulate the effective clock frequency (denoted as $1/T_{CMU_ECLK_EN}$) at which the registers are clocked. It is important to emphasize that $T_{CMU_ECLK_EN}$ is only a period of a virtual clock whose frequency is equivalent to the effective clock frequency resulting from using CMU_ECLK_EN as a clock enable to the registers the cluster clock is connected to. The following algorithm describes how to achieve a particular $T_{CMU_ECLK_EN}$ and the corresponding CMU_ECLK[z:z] by appropriately configuring CLSi_CMU_ECLKz_NUM and CLSi_CMU_ECLKz_DEN. Let:

$T_{CMU_ECLK0_EN}$ =(CLSi_CMU_ECLKz_NUM **(z=0)**/CLSi_CMU_ECLKz_DEN **(z=0)**)*$T_{CLS[j]_CLK}$

$T_{CMU_ECLK1_EN}$ =(CLSi_CMU_ECLKz_NUM **(z=1)**/CLSi_CMU_ECLKz_DEN **(z=1)**)*$T_{CLS[j]_CLK}$

$T_{CMU_ECLK2_EN}$ =(CLSi_CMU_ECLKz_NUM **(z=2)**/CLSi_CMU_ECLKz_DEN **(z=2)**)*$T_{CLS[j]_CLK}$

and is implemented according to the following algorithm

(Z: CLSi_CMU_ECLKz_NUM**.**ECLK_NUM ; N: CLSi_CMU_ECLKz_DEN**.**ECLK_DEN ; Z>=N >0 ; Z>=N ; CMU_ECLK[z:z]='0'):

1. INIT-phase: Set remainder (R), operand1 (OP1) and operand2 (OP2) register during INIT-phase (with implicit conversion to signed): R=Z, OP1=N, OP2=N-Z.
2. After leaving INIT-phase (CMU_ECLK[z:z] has been enabled) the sign of remainder R will be checked every (CLS[j]_CLK clock-cycle):
3. If R>=0: keep the corresponding CMU_ECLK_EN at '0', keep CMU_ECLK[z:z], and update R=R-OP1.
4. If R<0: set corresponding CMU_ECLK_EN to '1', toggle CMU_ECLK[z:z], and update R=R-OP2.

After at most (Z/N+1) subtractions (3. in the list above), there will be a negative R and an active phase of the generated clock enable (for one cycle) will be triggered (4. in the list above). The remainder R is a measure for the distance to a real Z/N clock and will be considered for the next generated clock toggle phase. The new R value will be R=R+(Z-N). In the worst case, the remainder R will sum up to an additional cycle in the generated clock toggle period after Z-cycles. In other cases equally distributed additional cycles will be inserted for the generated clock toggle. If Z is an integer multiple of N, no additional cycles will be included for the generated clock toggle at all.

Attention: an external clock signal (CMU_ECLK[z:z]) shall have exactly 50% duty cycle only when the ratio CLSi_CMU_ECLKz_NUM**.**ECLK_NUM / CLSi_CMU_ECLKz_DEN**.**ECLK_DEN results in an integer value.

***Note**: For a better resource sharing, all arithmetic has been reduced to subtractions and the initialization of the remainder R uses the complement of (Z-N).*

The initial value of the CMU_ECLK[z:z] output is low.

44.3.1.3.6 eGTM CMU Configuration Registers Description

Sub-chapter CMU Configuration Registers Description

Table 2178

Register Shortname	Description
CLSi_CMU_CLK_EN	CMU clock enable (i=0)
CLSi_CMU_GCLK_NUM	The numerator for CMU global clock resolution generator (i=0)
CLSi_CMU_GCLK_DEN	The denominator for CMU global clock resolution generator (i=0)
CLSi_CMU_CLKx_CTRL	CMU control for clock resolution generator [x] (x=0-5;i=0)
CLSi_CMU_CLK_6_CTRL	CMU control for clock resolution generator 6 (i=0)
CLSi_CMU_CLK_7_CTRL	CMU control for clock resolution generator 7 (i=0)
CLSi_CMU_ECLKz_NUM	The numerator for the external clock resolution generator [z] (z=0-2;i=0)
CLSi_CMU_ECLKz_DEN	The denominator for the external clock resolution generator [z] (z=0-2;i=0)
CLSi_CMU_FXCLK_CTRL	CMU control for selection of FCR subblock input (i=0)
CLSi_CMU_CLK_CTRL	CMU control for clock resolution generator (i=0)

44.3.1.3.7 CMU port description

Table 2179 CMU Ports

Name	Description
CMU_FXCLK_RES	**CMU** fixed clocks with predefined and non-configurable clock dividers
CMU_ECLK	**CMU** external clocks
CMU_CLK_RES	**CMU** clock resolutions

44.3.1.3.8 CMU signal description

Table 2180 CMU Signals

Name	Description
EN_FXCLK	Enables/disables the clock source for FCR generators
CMU_GCLK_EN	Common clock resolution signal generated by Global Clock Resolution Generator
CMU_ECLK0_EN	Clock resolution signal generated by External Clock Resolution Generator 0

(table continues...)

Table 2180 (continued) CMU Signals

Name	Description
CMU_ECLK1_EN	Clock resolution signal generated by External Clock Resolution Generator 1
CMU_ECLK2_EN	Clock resolution signal generated by External Clock Resolution Generator 2
CMU_ECLK_EN	Refers to CMU_ECLK0_EN or CMU_ECLK1_EN or CMU_ECLK2_EN depending on which External Clock Resolution Generator is under consideration

44.3.1.4 Cluster configuration module (CCM)

44.3.1.4.1 Overview

As already mentioned in **eGTM** architecture, each sub-module of the **eGTM** is aligned explicitly to a cluster. The Cluster Configuration Module (**CCM**) enables the configuration of several cluster-specific options namely:

- cluster's clock frequency,
- module clock gating,
- address range protection, and
- global architecture configuration.

Indices and their range as used inside this chapter are:

- i:={0, 1,..., NCMM-1} instance index of cluster / **CCM** module
- a:={0, 1,..., NARP-1} index of address range protectors

44.3.1.4.2 Cluster CMU clock resolution

The registers CLSi_CCM_CMU_CLK_CFG and CLSi_CCM_CMU_FXCLK_CFG allow the configuration of various cluster clock frequencies.

Figure 1038 shows important details about the wiring of the cluster's local clock signals. It is worth mentioning that CCM[i]_CLK_RES[7:7] is routed to the **CMU** module in which it can be used for clock resolution generation.

Clock resolution signals (CMU_CLK_RES[x:x], x ∈ {0, 1, ..., 8}) are routed from cluster 0 to the remaining clusters through ICPAs Inter-cluster pulse adapter. When a clock resolution signal with an effective clock frequency Global clock resolution generator greater than CLK / 2 of is routed from cluster 0 with CLS[j]_CLK = CLK to cluster z ∈ {1, ..., NCMM-1} with CLS[j]_CLK = CLK / 2, the effective clock frequency of the resolution signal at the output of ICPA is limited to CLK / 2.

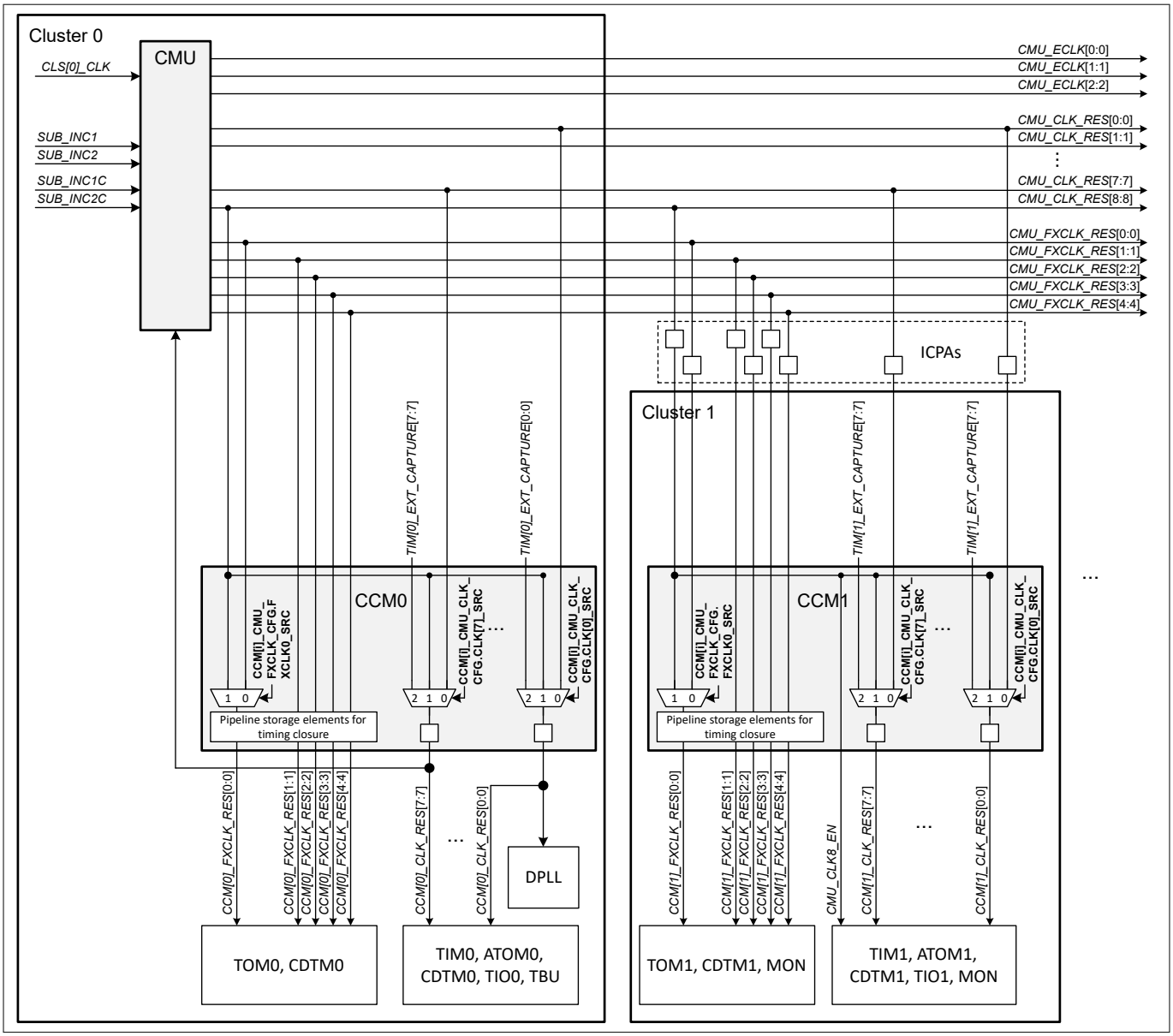


Figure 1038 Cluster clock resolution wiring

***Note**: The figure is a general picture from the IP vendor. The exact configuration can be found in the device specific configuration.*

44.3.1.4.3 Cluster timebase distribution

Figure 1039 depicts the wiring of the time bases to the clusters. The pipeline stages shown in the figure are only present for devices with CFG_CLOCK_RATE = 1.

In cluster 0, the time bases (TBU_TS0X, TBU_TS1_BASE, TBU_TS2_BASE, TBU_TS2_BASE) as well as the respective update indicators (TBU_UP0, TBU_UP1, TBU_UP2) are generated in **TBU** and routed to **CCM[0]** in which a pipeline stage is introduced in order to reach timing closure. The 27-bit time base (TBU_TS0X) is wired to **TIM[0]** and the **DPLL**. The lower 24 bits or the higher 24 bits of TBU_TS0X are selected based on CLSi_TBU_CH0_CTRL.LOW_RES and forwarded to CCM[i]_TBU_TS0. The time bases (CCM[i]_TBU_TS0, CCM[i]_TBU_TS1, CCM[i]_TBU_TS2) are routed to the appropriate modules. The times bases (CCM[i]_TBU_TS0, CCM[i]_TBU_TS1, CCM[i]_TBU_TS2) as well as the update indicators ((e)GTM_DBG_TBU_UP0, (e)GTM_DBG_TBU_UP1, (e)GTM_DBG_TBU_UP2) are exposed as debug ports.

The time bases (TBU_TS0_BASE, TBU_TS1_BASE, TBU_TS2_BASE) are generated in **TBU** and routed to other clusters via the **CCM**s. In a **CCM**, a pipeline stage for the times bases is introduced in order to reach timing closure. The lower 24 bits or the higher 24 bits of TBU_TS0X are selected based on CLSi_TBU_CH0_CTRL**.**LOW_RES and forwarded to CCM[i]_TBU_TS0. The time bases at the output of the pipeline stage (CCM[i]_TBU_TS0, CCM[i]_TBU_TS1, CCM[i]_TBU_TS2) are routed to the appropriate modules.

***Note**:* *If cluster 0 is clocked with CLK whereas another cluster (say cluster 1) is clocked with CLK/2, not all the values of the time bases generated in cluster 0 are visible/sampled in cluster 1. Therefore, it is strongly advised that the user configures the compare operations in such clusters to be based on the 'greater than or equal to' operator - not 'equal to' operator.*

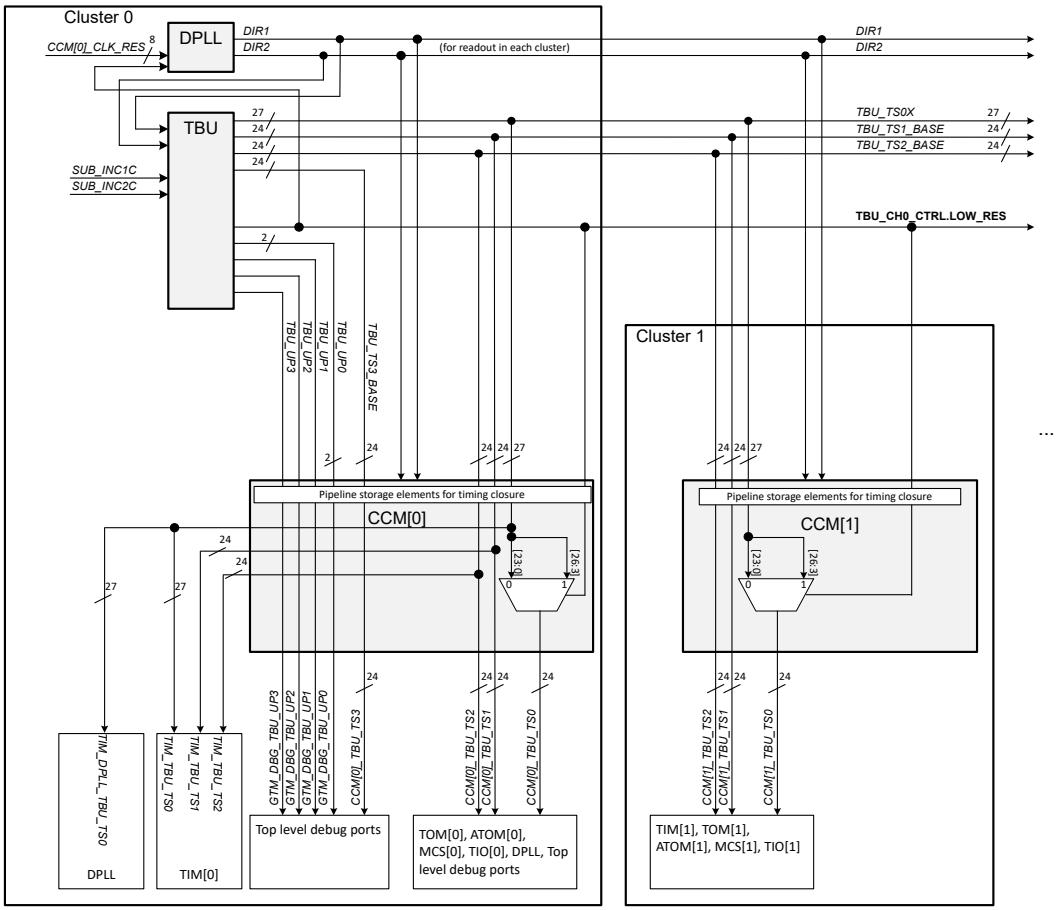


Figure 1039 Cluster timebase wiring

***Note**: The figure is a general picture from the IP vendor. The exact configuration can be found in the device specific configuration.*

44.3.1.4.4 Address range protection

The **CCM** also provides so-called address range protectors (ARPs) up to NARP. An ARP can be used to define a configurable write-protected address range in order to support enhanced safety features.

44.3.1.4.5 eGTM CCM Configuration Registers Description

Sub-chapter CCM Configuration Registers Description

Table 2181

Register Shortname	Description
CLSi_CCM_PROT	CCM[i] protection register (i=0-2)
CLSi_CCM_CFG	CCM[i] configuration register (i=0,2)
CLSi_CCM_CFG	CCM[i] configuration register (i=1)
CLSi_CCM_CMU_CLK_CFG	CCM[i] CMU clock configuration register (i=0-2)
CLSi_CCM_CMU_FXCLK_CFG	CCM[i] CMU fixed clock configuration register (i=0-2)
CLSi_CCM_HW_CONF	CCM[i] hardware configuration register (i=0-2)
CLSi_CCM_TIM_AUX_IN_SRC	CCM[i] TIM AUX input source register (i=0-2)
CLSi_CCM_TOM_OUT	CCM[i] TOM output register (i=0-2)
CLSi_CCM_ATOM_OUT	CCM[i] ATOM output register (i=0-1)
CLSi_CCM_ATOM_OUT	CCM[i] ATOM output register (i=2)
CLSi_CCM_HW_CONF2	CCM[i] 2. hardware configuration register (i=0-2)

44.3.1.4.6 CCM port description

Table 2182 CCM clock / reset/ infrastructure interface

Name	Description
CCM_RESET	Asynchronous reset, active reset level defined by constant RESET_ACTIVE_C
CCM_ENCLK	Enable signal of CCM_SYS_CLK/CLK
CCM_CLK	Clock signal, possible divider values 0,1,2, only gated by gtm_halt
CCM_ATOM_OUT	Value of register CLSi_CCM_ATOM_OUT.ATOM_I_OUTx (x=0-7)
CCM_ATOM_OUT_N	Value of register CLSi_CCM_ATOM_OUT.ATOM_I_OUT_Nx (x=0-7)
CCM_TOM_OUT	Value of register CLSi_CCM_TOM_OUT.TOM_OUTx (x=0-15)
CCM_TOM_OUT_N	Value of register CLSi_CCM_TOM_OUT.TOM_OUT_Nx (x=0-15)

Table 2183 CCM TBU interface

Name	Description
CCM_CMU_CLK_EN	Enable signal of the **CMU** clocks
CLSi_CMU_CLK_EN	Enable signal vector of **CMU** clocks 0 .. 8
CMU_FXCLK_EN	Enable signal of the **CMU** fixed clocks

Table 2184 CCM CLS interface

Name	Description
CLS_CLK_0_EN	Cluster clock enable signals to switch on/off the clock for the cluster sub-modules (see register CLSi_CCM_CFG)
CLS_CLK_1_EN	Cluster clock enable signals to switch on/off the clock for the cluster sub-modules (see register CLSi_CCM_CFG)
CLS_CLK_2_EN	Cluster clock enable signals to switch on/off the clock for the cluster sub-modules (see register CLSi_CCM_CFG)
CLS_CLK_3_EN	Cluster clock enable signals to switch on/off the clock for the cluster sub-modules (see register CLSi_CCM_CFG)
CLS_CLK_4_EN	Cluster clock enable signals to switch on/off the clock for the cluster sub-modules (see register CLSi_CCM_CFG)
CLS_CLK_5_EN	Cluster clock enable signals to switch on/off the clock for the cluster sub-modules (see register CLSi_CCM_CFG)
CLS_CLK_6_EN	Cluster clock enable signals to switch on/off the clock for the cluster sub-modules (see register CLSi_CCM_CFG)
CLS_CLK_7_EN	Cluster clock enable signals to switch on/off the clock for the cluster sub-modules (see register CLSi_CCM_CFG)
CLK_CLK_DIV	Value of register CLSi_CCM_CFG.CLS_CLK_DIV

Table 2185 CCM Ports

Name	Description
CCM_AUX_IN	Auxiliary input source of **TIM[i]**
CCM[i]_CLK_RES	Resolution in use, selected k-th clock output of i-th **CCM** instance
CCM[i]_FXCLK_RES	Resolution in use, selected input fix clock of instance i-th
CCM[i]_TBU_TS0	Timestamp of **TBU** channel 0, 24 bits of time base register CLSi_TBU_CH0_BASE (26:3/23:0 dependent on CLSi_TBU_CH0_CTRL.LOW_RES)
CCM[i]_TBU_TS1	Time/Angle value of **TBU** channel 1, time/angle base register CLSi_TBU_CHx_BASE
CCM[i]_TBU_TS2	Time/Angle value of tbu_CH2, time/angle base register CLSi_TBU_CHx_BASE
GTM_HALT	Global debug signal for halting the entire **eGTM**

44.3.1.4.7 CCM signal description

The register CLSi_CCM_CFG allows disabling the system clock signal for unused sub-modules of the i-th cluster.

Table 2186 CCM[j]_CLK_ENABLE

Name	Description
CCM[j]_CLK_ENABLE	**CCM[j]** interface clock is enabled

Table 2187 CMU_CLK_ENABLE

Name	Description
CMU_CLK_ENABLE	**CMU** interface clock is enabled

Table 2188 TBU_CLK_ENABLE

Name	Description
TBU_CLK_ENABLE	**TBU** interface clock is enabled

Table 2189 ARCH_CLK_ENABLE

Name	Description
ARCH_CLK_ENABLE	**ARCH** interface clock is enabled

Table 2190 ICM_CLK_ENABLE

Name	Description
ICM_CLK_ENABLE	**ICM** interface clock is enabled

Table 2191 TIM[j]_CLK_ENABLE

Name	Description
TIM[j]_CLK_ENABLE	**TIM[j]** interface clock is enabled

Table 2192 TOM[j]_CLK_ENABLE

Name	Description
TOM[j]_CLK_ENABLE	**TOM[j]** interface clock is enabled

Table 2193 SPE[j]_CLK_ENABLE

Name	Description
SPE[j]_CLK_ENABLE	**SPE[j]** interface clock is enabled

Table 2194 ATOM[j]_CLK_ENABLE

Name	Description
ATOM[j]_CLK_ENABLE	**ATOM[j]** interface clock is enabled

Table 2195 PSM[j]_CLK_ENABLE

Name	Description
PSM[j]_CLK_ENABLE	PSM[j] interface clock is enabled

Table 2196 CMP_CLK_ENABLE

Name	Description
CMP_CLK_ENABLE	**CMP** interface clock is enabled

Table 2197 MON_CLK_ENABLE

Name	Description
MON_CLK_ENABLE	**MON** interface clock is enabled

Table 2198 **CDTM[j]_CLK_ENABLE**

Name	Description
CDTM[j]_DTM[d]_CLK_ENABLE	CDTM[j]_DTM[d] interface clock is enabled

44.3.1.5 Time base unit (TBU)

44.3.1.5.1 Overview

The Time Base Unit **TBU** provides common time bases for the **eGTM**. The **TBU** sub-module is organized in channels, where the number of channels is device-dependent. There are up to four channels implemented inside the **TBU**.

Indices and their range as used inside this chapter are:

- x:={1, 2} index for time base channel 1 and channel 2
- y:={0, 1, 2} index of **TBU** channel

The time base register CLSi_TBU_CH0_BASE of **TBU** channel 0 is 27 bits wide and it is configurable whether the lower 24 bits or the upper 24 bits are provided to the **eGTM** as signal TBU_TS0_BASE.

The two **TBU** channels 1 and 2 have a time base register CLSi_TBU_CHx_BASE (i=0;x=1-2) (x∈{1, 2}) of 24 bits length. The time base register values TBU_TS1_BASE and TBU_TS2_BASE are provided to subsequent sub-modules of the **eGTM**.

***Note**: The right-hand sum is limited to 24 bits.*

The TBU_UP1, TBU_UP2 signals are set to high for a single cls_clk period, whenever the corresponding signals TBU_TS1_BASE, TBU_TS2_BASE are updated. The signal TBU_UP0_L is set to high for a single cls_clk period, if the signal TBU_TS0_BASE and TBU_TS0X are updated and TBU_UP0_H is set to high for a single cls_clk period, whenever the upper 24 bits of TBU_TS0_BASE are updated.

The time base channels can run independently of each other and can be enabled and disabled synchronously by control bits in a global **TBU** channel enable register CLSi_TBU_CHEN. Figure 1040 shows a block diagram of the Time Base Unit.

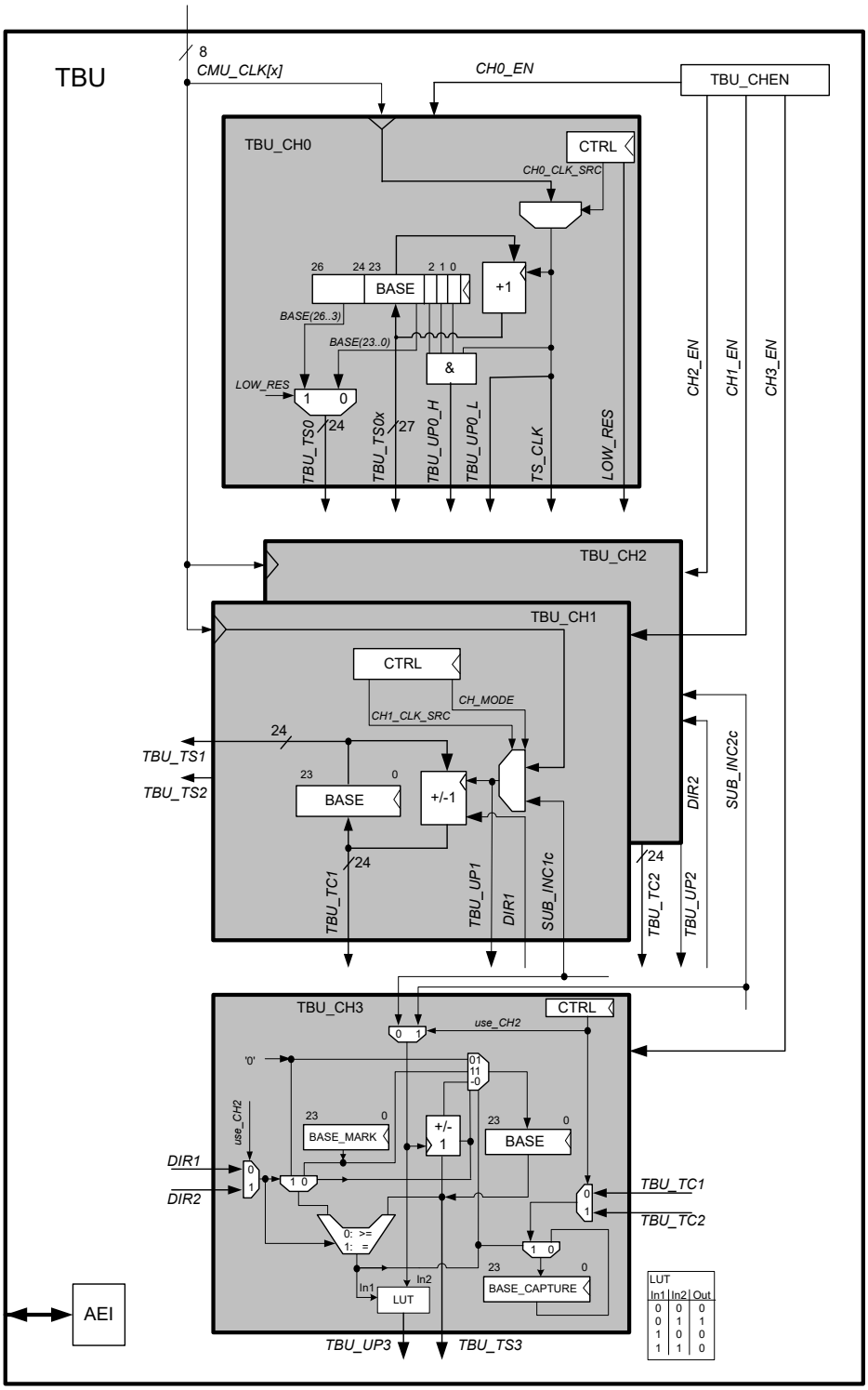


Figure 1040 TBU block diagram

***Note**: The figure is a general picture from the IP vendor. The exact configuration can be found in the device specific configuration.*

Depending on the device a third **TBU** channel exists which offers the same functionality as the time base channel 1.

The configuration of the independent time base channels CLSi_TBU_CHx_BASE (i=0;x=1-2) is done via the **AEI** interface. The **TBU** channel 0 to channel 2 may select one of the eight CCM[i]_CLK_RES[y:y] (y∈{0, 1,..., 7}) signals coming from the **CMU** sub-module.

44.3.1.5.2 TBU channels

The time base values are generated within the **TBU** time base channels in two independent and one dependent operation modes.

In all modes, the time base registers CLSi_TBU_CH0_BASE, CLSi_TBU_CHx_BASE (i=0;x=1-2) can be initialized with a start value just before enabling the corresponding **TBU** channel.

Moreover, the time base registers CLSi_TBU_CH0_BASE, CLSi_TBU_CHx_BASE (i=0;x=1-2) can always be read in order to determine the actual value of the counter.

Independent modes

Free running counter mode

TBU channel 0 provides a 27-bit counter in a free running counter mode. Depending on the bit field CLSi_TBU_CH0_CTRL.LOW_RES, the lower 24 bits CLSi_TBU_CH0_BASE.BASE[23:0] or the upper 24 bits CLSi_TBU_CH0_BASE.BASE[26:3] are provided to the **eGTM** sub-modules.

TBU channel 1 and channel 2 provides a 24-bit counter in a free running counter mode enabled by reset of CLSi_TBU_CH1_CTRL.CH_MODE and CLSi_TBU_CH2_CTRL.CH_MODE.

In **TBU** free running counter mode, the time base registers CLSi_TBU_CH0_BASE and CLSi_TBU_CHx_BASE (i=0;x=1-2) are updated on every specified incoming clock event by the selected signal CCM[i]_CLK_RES[y:y] (y∈{0, 1,..., 7}) (depending on CLSi_TBU_CH0_CTRL, CLSi_TBU_CH1_CTRL and CLSi_TBU_CH2_CTRL register). In general, the time base register CLSi_TBU_CHx_BASE (i=0;x=1-2) is incremented on every CCM[i]_CLK_RES[y:y] clock tick.

Forward/backward counter mode

TBU channels 1 and 2 provide a 24-bit forward/backward counter enabled by set of CLSi_TBU_CH1_CTRL.CH_MODE and CLSi_TBU_CH2_CTRL.CH_MODE.

44.3.1.5.3 eGTM TBU Configuration Registers Description

Sub-chapter TBU Configuration Registers Description

Table 2199

Register Shortname	Description
CLSi_TBU_CHEN	TBU global channel enable (i=0)
CLSi_TBU_CH0_CTRL	TBU channel 0 control (i=0)
CLSi_TBU_CH0_BASE	TBU channel 0 base (i=0)
CLSi_TBU_CH1_CTRL	TBU channel 1 control (i=0)

(table continues...)

Table 2199 (continued)

Register Shortname	Description
CLSi_TBU_CH2_CTRL	TBU channel 2 control (i=0)
CLSi_TBU_CHx_BASE	TBU channel [x] base (x=1-2;i=0)

44.3.1.5.4 TBU port description

Table 2200 TBU Ports

Name	Description
TBU_TS0_BASE	**TBU** channel 0, lower 24 bits of CLSi_TBU_CH0_BASE**[23:0]**
TBU_TS0X	**TBU** channel 0 to 27 bits of CLSi_TBU_CH0_BASE**[26:0]**
TBU_TS1_BASE	**TBU** channel 1, CLSi_TBU_CHx_BASE**[23:0]**
TBU_TS2_BASE	**TBU** channel 2, CLSi_TBU_CHx_BASE**[23:0]**
TBU_UP0	Indication of update. The LSB reflects TBU_UP0_L and the MSB reflects TBU_UP0_H.
TBU_UP0_H	Indication of update of TBU_TS0_BASE and TBU_TS0X
TBU_UP0_L	Indication of update of TBU_TS0X *[23:3]*
TBU_UP1	Indication of update of TBU_TS1_BASE
TBU_UP2	Indication of update of TBU_TS2_BASE

44.3.1.6 Timer input module (TIM)

44.3.1.6.1 Overview

The Timer Input Module (**TIM**) is responsible for filtering and capturing input signals of the **eGTM**. Several characteristics of the input signals can be measured inside the **TIM** channels.

Input characteristics mean either timestamp values of detected input rising or falling edges together with the new signal level or the number of edges received since channel enable together with the actual timestamp or PWM signal duration for a whole PWM period.

Indices and their range as used inside this chapter are:

- NTIM: number of **TIM**
- i:={0, 1,..., NTIM-1} instance index of cluster / module
- x:={0, 1,..., 7} channel index in a **TIM** instance
- m: number of channels available in the **TIM** sub-module
- y:={0, 1,..., m-1} index for the number **TIM** sub-module channels

The number of channels m inside a **TIM** sub-module depends on the device.

The architecture of **TIM** is shown in Figure 1041.

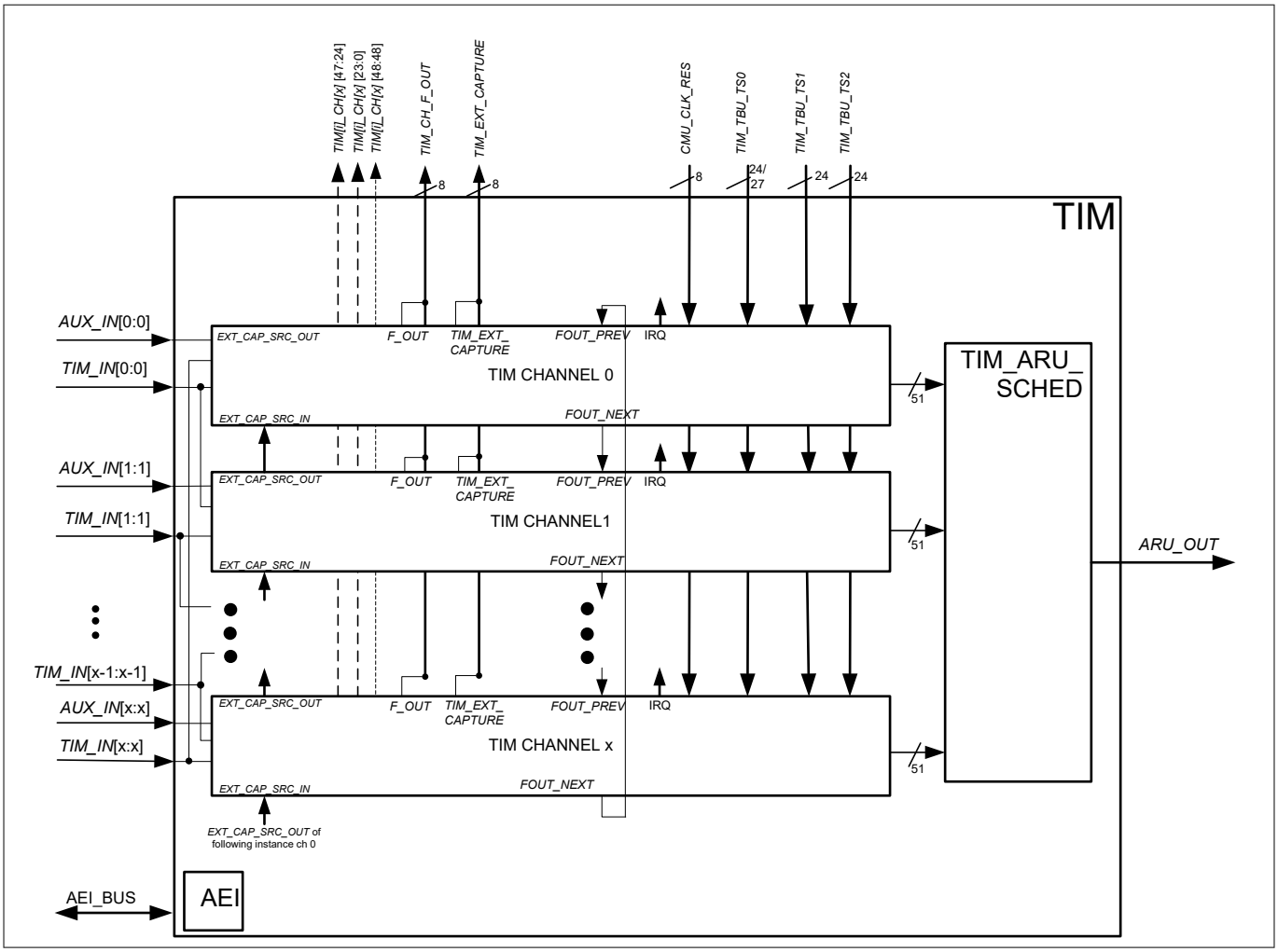


Figure 1041 TIM block diagram

Each of the m dedicated input signals are filtered inside the FLT_CHx sub-unit of the **TIM** module.

***Note**: The input signals TIM_IN[x:x]/TIM_IN[x-1:x-1] are driven by external logic and has no defined relationship to the internal cluster clock. To avoid metastability it is necessary to synchronize TIM_IN[x:x]/TIM_IN[x-1:x-1] via two register stages, which leads to a delay of two cls_clk periods for these incoming signals. On the other hand the input signals AUX_IN[x:x] are not synchronized as these are generated already synchronously within the **eGTM**.*

The measurement values can be read by the CPU directly via the **AEI**-Bus .

The two (three) time bases coming from the **TBU** are connected to the **TIM** channels to annotate timestamps to incoming signals. For TIM0, the extended 27 bits wide time base TIM_TBU_TS0 is connected to the **TIM** channels, and the user has to select if the lower 24 bits (TIM_TBU_TS0[23:0]) or the higher 24 bits (TIM_TBU_TS0[26:3]) are stored inside the CLSi_TIM_CHx_GPR0 and CLSi_TIM_CHx_GPR1 registers.

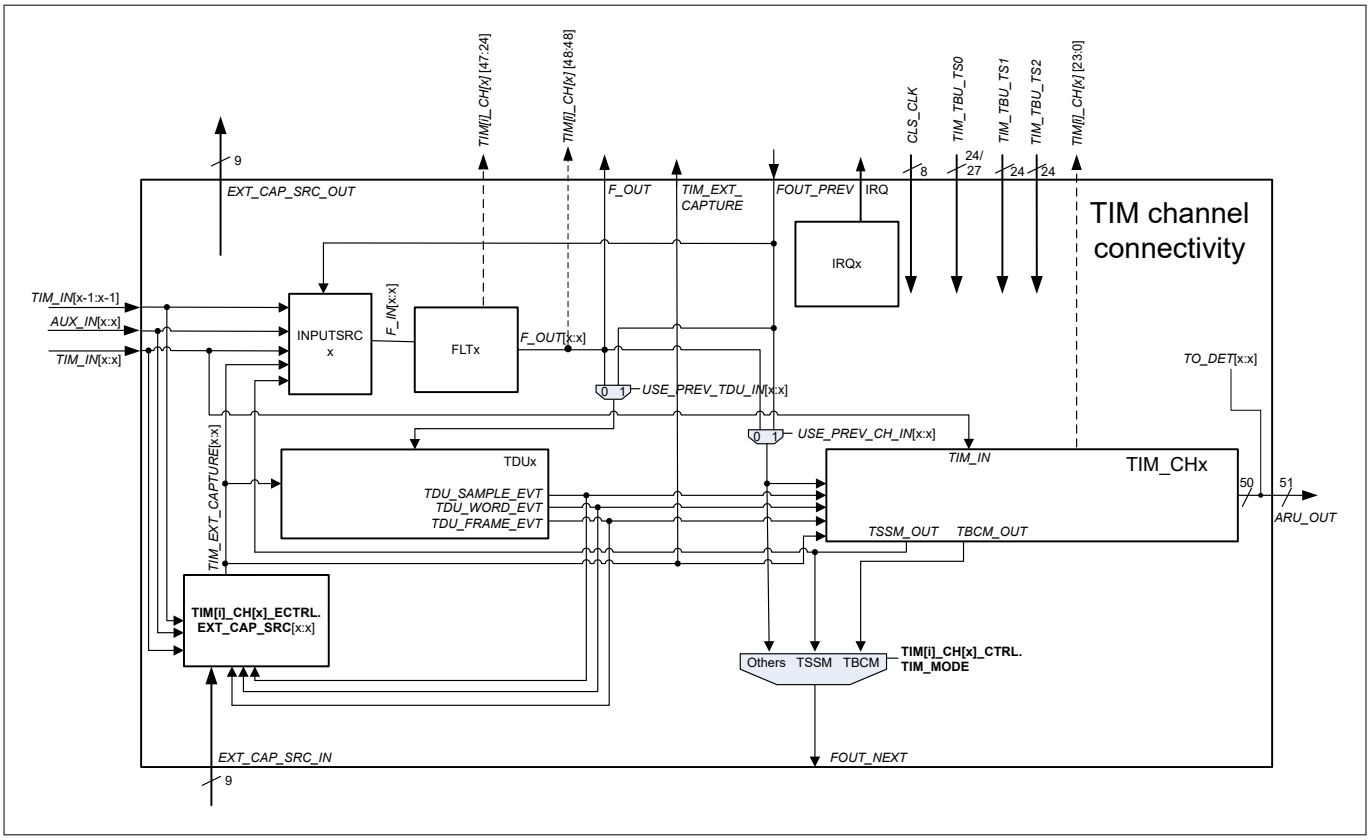


Figure 1042 TIM channel internal connectivity

- Above figure gives an overview of the channel internal connectivity of the sub-units. The sub-units with the major functionality are listed next:
- INPUTSRC_CHx: Select signal for processing by the filter unit FLT_CHx
- FLT_CHx: The filter unit provides different filter mechanisms described in more detail in **TIM** filter functionality (FLT).
- TDU_CHx: Timeout detection unit (no subsequent edge detected during a specified duration)
- TIM_CHx: Measurement unit; different measurements strategies configurable on the filtered signal
- IRQ_CHx: Local interrupt controller (enabling, status, ..)
- CLSi_TIM_CHx_ECTRL.EXT_CAP_SRC: Selects a local signal TIM_EXT_CAPTURE[x:x] which is needed by certain functions
- Details are given in the next chapters.

***Note**: Depending on the values of the configuration bit fields CLSi_TIM_CHx_ECTRL.USE_PREV_TDU_IN , CLSi_TIM_CHx_ECTRL.USE_PREV_CH_IN, it is possible to operate on the signal of the local channel x or the previous channel x-1.*

Depending on the value of the configuration bit field CLSi_TIM_CHx_CTRL.TIM_MODE it is possible to provide different signals (via FOUT_NEXT) to the next channel.

In TBCM mode each capture event selected by the sensitive edges (CLSi_TIM_CHx_CNTS.CNTS) will be forwarded with the value of CLSi_TIM_CHx_ECNT[0] to the following channel (via FOUT_NEXT).

Input source selection INPUTSRCx

It can be configured which source is used for processing in the FLT,TDU,TIM_CH units. It can be selected by the bit fields CLSi_TIM_CHx_CTRL.CICTRL and CLSi_TIM_IN_SRC.MODEx, CLSi_TIM_IN_SRC.VALx in the register CLSi_TIM_IN_SRC which source is in use.

Alternatively, the signal F_IN[x:x] can be generated by an 8-bit lookup-table, which allows to define any function of 3 input sources.

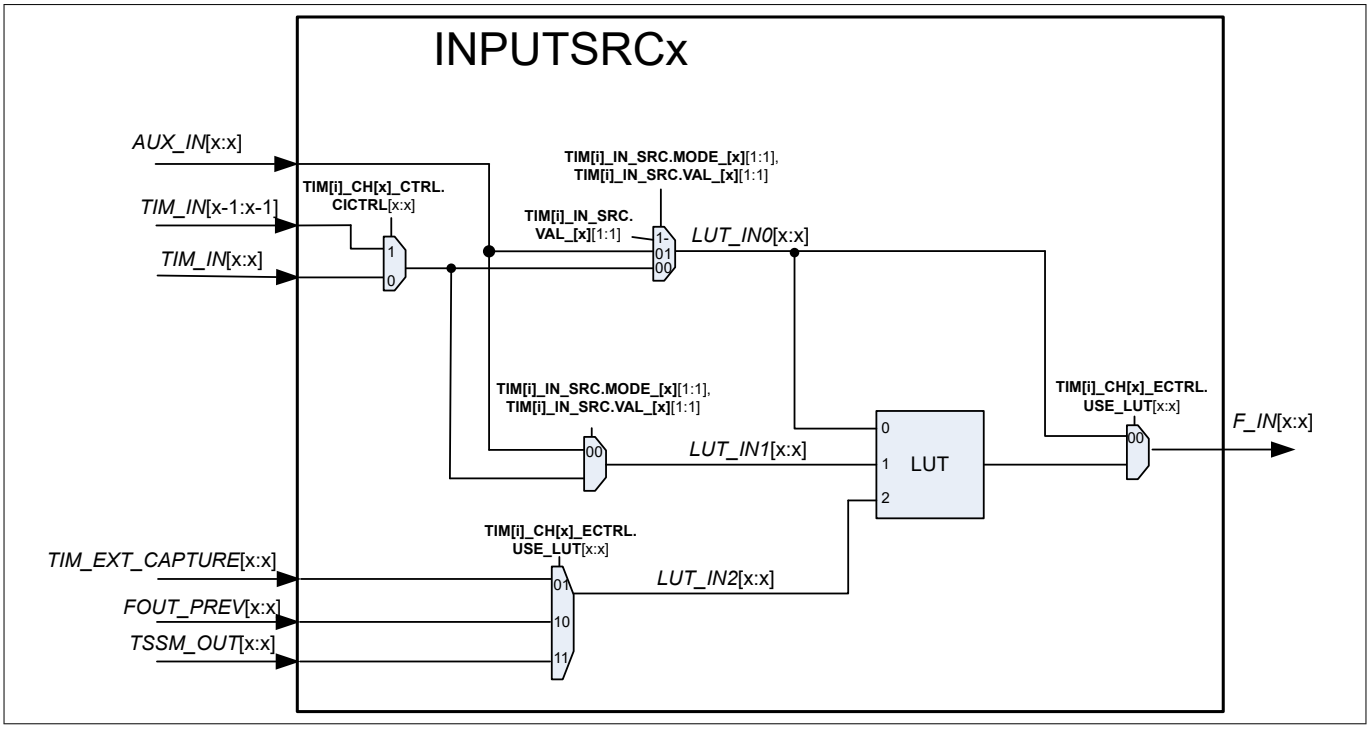


Figure 1043 INPUTSRC block diagram

If CLSi_TIM_CHx_ECTRL.USE_LUT=00_B is set, the lookup-table signal generation is bypassed and the signal is selected as follows:

In a certain CLSi_TIM_IN_SRC.MODEx, CLSi_TIM_IN_SRC.VALx combination the input signal F_IN[x:x] can be driven by CLSi_TIM_IN_SRC.VALx[1:1] with 0 or 1 directly.

Due to the fact that all 8 channels are bundled in the register CLSi_TIM_IN_SRC a synchronous control of all 8 input channels is possible.

Two adjacent channels can be combined by setting the CLSi_TIM_CHx_CTRL.CICTRL bit field. This allows combining of complex measurements on one input signal with two **TIM** channels.

The additional signal AUX_IN[x:x] can be selected as an input signal. The source of this signal is defined in the Figure 1018.

If CLSi_TIM_CHx_ECTRL.USE_LUT !=00_B is set, the lookup-table signal generation with following inputs is in use. See Figure 1043:

Input LUT_IN0[x:x] selection:

- CLSi_TIM_INP_VAL.TIM_INx (x=0-7) if CLSi_TIM_CHx_CTRL.CICTRL = 0 and CLSi_TIM_IN_SRC.MODEx[1:1] = 0 and CLSi_TIM_IN_SRC.VALx[1:1] = 0
- CLSi_TIM_INP_VAL.TIM_INx (x=0-7) **(x=x-1)** if CLSi_TIM_CHx_CTRL.CICTRL = 1 and CLSi_TIM_IN_SRC.MODEx[1:1] = 0 and CLSi_TIM_IN_SRC.VALx[1:1] = 0
- AUX_IN[x:x] if CLSi_TIM_IN_SRC.MODEx[1:1]=0 and CLSi_TIM_IN_SRC.VALx[1:1] = 1
- CLSi_TIM_IN_SRC.VALx if CLSi_TIM_IN_SRC.MODEx[1:1] = 1

Input LUT_IN1[x:x] selection:

- AUX_IN[x:x] if CLSi_TIM_IN_SRC.MODEx[1:1] = 0 and CLSi_TIM_IN_SRC.VALx[1:1] = 0
- CLSi_TIM_INP_VAL.TIM_INx (x=0-7) if CLSi_TIM_CHx_CTRL.CICTRL= 0
- CLSi_TIM_INP_VAL.TIM_INx (x=0-7) **(x=x-1)** if CLSi_TIM_CHx_CTRL.CICTRL= 1

Input LUT_IN2[x:x] selection:

- TIM_EXT_CAPTURE[x:x] if CLSi_TIM_CHx_ECTRL.USE_LUT=01_B
- FOUT_PREV[x:x] if CLSi_TIM_CHx_ECTRL.USE_LUT=10_B
- TSSM_OUT[x:x] if CLSi_TIM_CHx_ECTRL.USE_LUT=11_B

The lookup-table is defined by the contents of the bit field CLSi_TIM_CHx_TDUC.TO_CNT2. The lookup_table_index is defined by LUT_IN2[x:x] & LUT_IN1[x:x] & LUT_IN0[x:x]. The signal F_IN[x:x] is generated by CLSi_TIM_CHx_TDUC.TO_CNT2[lookup_table_index].

If CLSi_TIM_CHx_ECTRL.USE_LUT !=00_B is set, only limited functionality is available in the TDU. See bit field CLSi_TIM_CHx_TDUV.SLICING in the register CLSi_TIM_CHx_TDUV.

Input observation

It is possible to observe for all channels of one instance by reading CLSi_TIM_INP_VAL the actual signal values of the following processing stages:

- TIM_IN[7:0] signals after **TIM** input synchronization
- F_IN[7:0] signals after **TIM** INPUTSRC selection (input to FLT)
- F_OUT[7:0] signals after **TIM** filter functionality (output of FLT)

External capture source selection EXTCAPSRCx

Each channel can operate on an external capture signal TIM_EXT_CAPTURE. The source to use for this signal can be configured by the bit field CLSi_TIM_CHx_ECTRL.EXT_CAP_SRC.

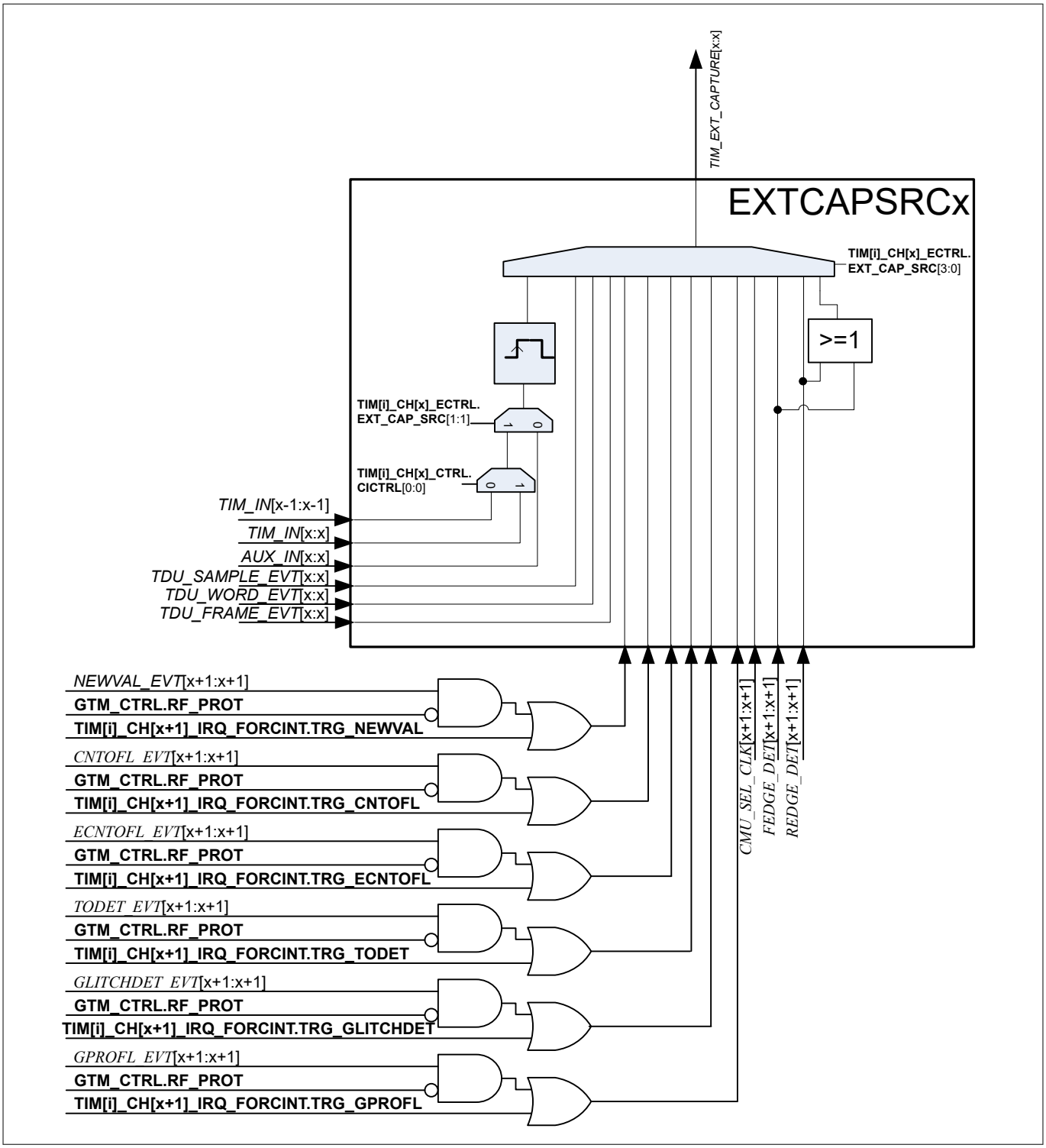


Figure 1044 EXTCAPSRC block diagram

A pulse generation for each rising edge of the selected input signal CLSi_TIM_INP_VAL.TIM_INx (x=0-7), CLSi_TIM_INP_VAL.TIM_INx (x=0-7) **(x=x-1)** and AUX_IN[x:x] is applied. These pulses can be selected and used as TIM_EXT_CAPTURE[x:x] signal.

Alternatively, one of the events TDU_SAMPLE_EVT[x:x], TDU_WORD_EVT[x:x], TDU_FRAME_EVT[x:x] generated in the TDU can be selected.

Furthermore, one of the six interrupt sources of channel x+1 can be selected. Alternatively, they can be issued by a soft trigger using the corresponding bits in the register CLSi_TIM_CHx_IRQ_FORCINT **(x=x+1)**. The pulses

on CLSi_TIM_CHx_IRQ_FORCINT **(x=x+1)** are allowed to pass through only when CLSi_ARCH_CTRL**.**RF_PROT = 0.

Additional sources of channel x+1 can be selected (rise/ fall/ both input edges or **CMU** clock resolution in use in channel x+1).

44.3.1.6.2 TIM filter functionality (FLT)

Overview

The **TIM** sub-module provides a configurable filter mechanism for each input signal. These filter mechanisms are provided inside the FLT sub-unit.

FLT architecture is shown in Figure 1045.

The filter includes a clock synchronization unit (CSU), an edge detection unit (EDU), and a filter counter associated with the filter unit (FLTU).

The CSU synchronizes the incoming signal F_IN with the selected filter clock frequency, which is controlled with the bit field CLSi_TIM_CHx_CTRL**.**FLT_CNT_FRQ.

The synchronized input signal F_IN_SYNC is used for further processing within the filter.

Within the FLT sub-module there is a two cluster-clock-cycle delay between F_IN and F_OUT, one cycle in the CSU sub-block and one cycle in the output signal generation sub-block (see Figure 1045).

***Note**: Glitches with a duration go less than the selected **CMU** clock period is lost.*

The filter modes can be applied individually to the falling and rising edges of an input signal. The following filter modes are available:

- immediate edge propagation mode,
- individual deglitch time mode (up/down counter), and
- individual deglitch time mode (hold counter).
- individual deglitch time mode (reset counter).

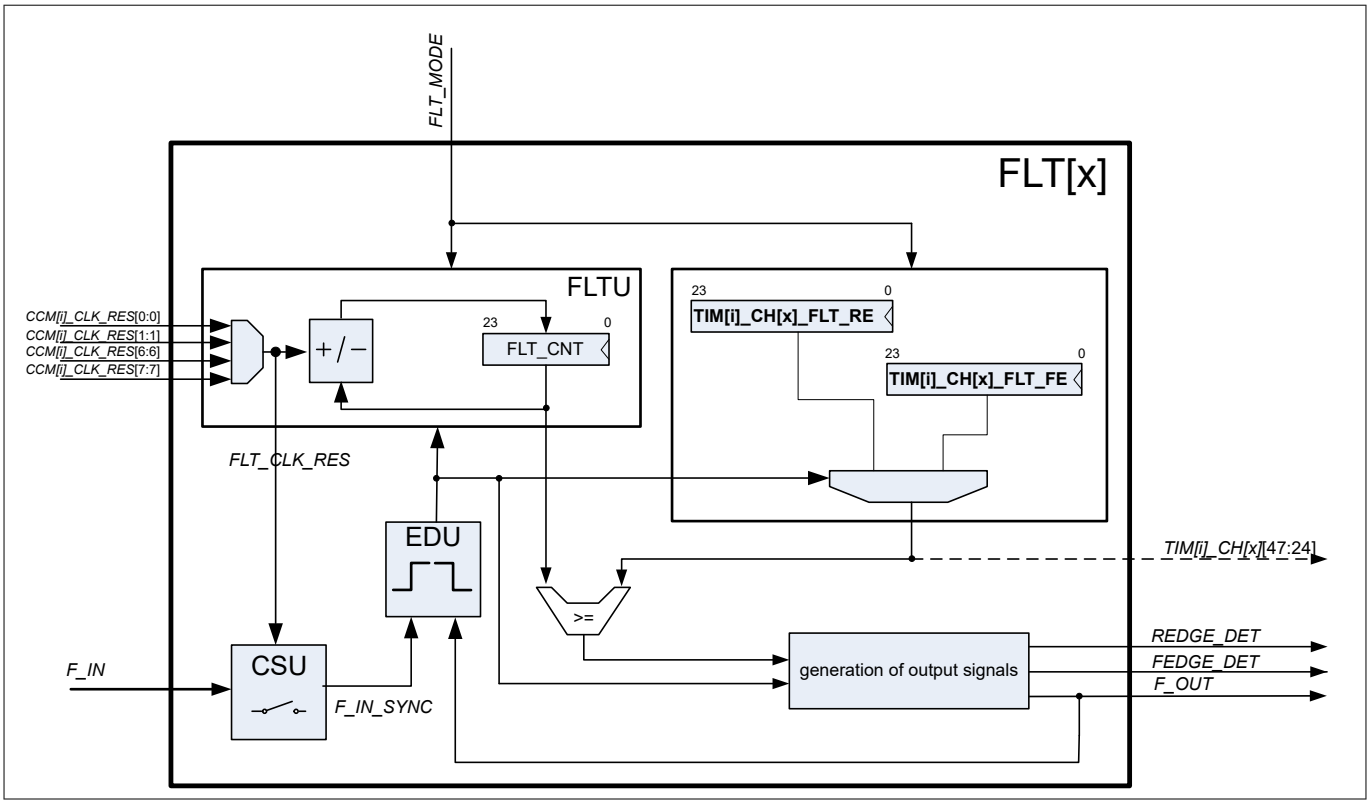


Figure 1045 FLT architecture

The filter parameters (deglitch and acceptance time) for the rising and falling edge can be configured inside the two filter parameter registers CLSi_TIM_CHx_FLT_RE (rising edge) and CLSi_TIM_CHx_FLT_FE (falling edge). The exact meaning of the parameter depends on the filter mode.

However, the delay time T of both filter parameters CLSi_TIM_CHx_FLT_FE, CLSi_TIM_CHx_FLT_RE can always be determined by:

$T=(CLSi_TIM_CHx_FLT_RE.FLT_RE+1)*T_{FLT_CLK_RES}$,

$T=(CLSi_TIM_CHx_FLT_FE.FLT_FE+1)*T_{FLT_CLK_RES}$,

Whereas $T_{FLT_CLK_RES}$ is the clock period of the selected **CMU** clock signal in bit field CLSi_TIM_CHx_CTRL.FLT_CNT_FRQ of register CLSi_TIM_CHx_CTRL.

When a glitch is detected on an input signal, a pulse with the length of one cluster clock on the interrupt event signal *GLITCHDET_IRQ* is raised and a status flag CLSi_TIM_CHx_IRQ_NOTIFY.GLITCHDET is set.

Table 2201 gives an overview of the meanings for the registers CLSi_TIM_CHx_FLT_RE and CLSi_TIM_CHx_FLT_FE.

Table 2201 Filter parameter summary for the different filter modes

Filter mode	Meaning of CLSi_TIM_CHx_FLT_RE	Meaning of CLSi_TIM_CHx_FLT_FE
Immediate edge propagation	Acceptance time for rising edge	Acceptance time for falling edge
Individual deglitch time (up/down counter)	Deglitch time for rising edge	Deglitch time for falling edge
Individual deglitch time (hold counter)	Deglitch time for rising edge	Deglitch time for falling edge
Individual deglitch time (reset counter)	Deglitch time for rising edge	Deglitch time for falling edge

A counter FLT_CNT is used to measure the glitch and acceptance times.

The frequency of the FLT_CNT counter is configurable in bit field CLSi_TIM_CHx_CTRL.FLT_CNT_FRQ.

The counter FLT_CNT is operating on FLT_CLK_RES clock resolution. The source for FLT_CLK_RES can be selected with CLSi_TIM_CHx_CTRL.FLT_CNT_FRQ from CCM[i]_CLK_RES[0:0], CCM[i]_CLK_RES[1:1], CCM[i]_CLK_RES[6:6] or the CCM[i]_CLK_RES[7:7] signals which were provided by the **CCM[i]** module.

The counter FLT_CNT and the registers CLSi_TIM_CHx_FLT_FE and CLSi_TIM_CHx_FLT_RE are 24 bits wide. For example, when the resolution of the CCM[i]_CLK_RES[0:0] signal is 50ns this allows maximum deglitch and acceptance times of about 838ms for the filter.

TIM filter modes

Immediate edge propagation mode

In immediate edge propagation mode after detection of an edge, the new signal level on F_IN_SYNC is propagated to F_OUT with a delay of one $T_{FLT_CLK_RES}$ period and the new signal level remains unchanged until the configured acceptance time expires.

For each edge type the acceptance time can be specified separately in the CLSi_TIM_CHx_FLT_RE and CLSi_TIM_CHx_FLT_FE registers.

Each signal change on the input F_IN_SYNC during the duration of the acceptance time has no effect on the output signal level F_OUT of the filter but it sets the glitch CLSi_TIM_CHx_IRQ_NOTIFY.GLITCHDET bit.

After it expires an acceptance time the input signal F_IN_SYNC is observed and on signal level change the filter raises a new detected edge and the new signal level is propagated to F_OUT.

Independent of a signal level change the value of F_OUT is always set to F_IN_SYNC, when the acceptance time expires (see also Figure 1047).

Figure 1046 shows an example for the immediate edge propagation mode, in the case of rising edge detection. Both, the signal before filtering (F_IN) and after filtering (F_OUT) are shown. The acceptance time at1 is specified in the register CLSi_TIM_CHx_FLT_RE.

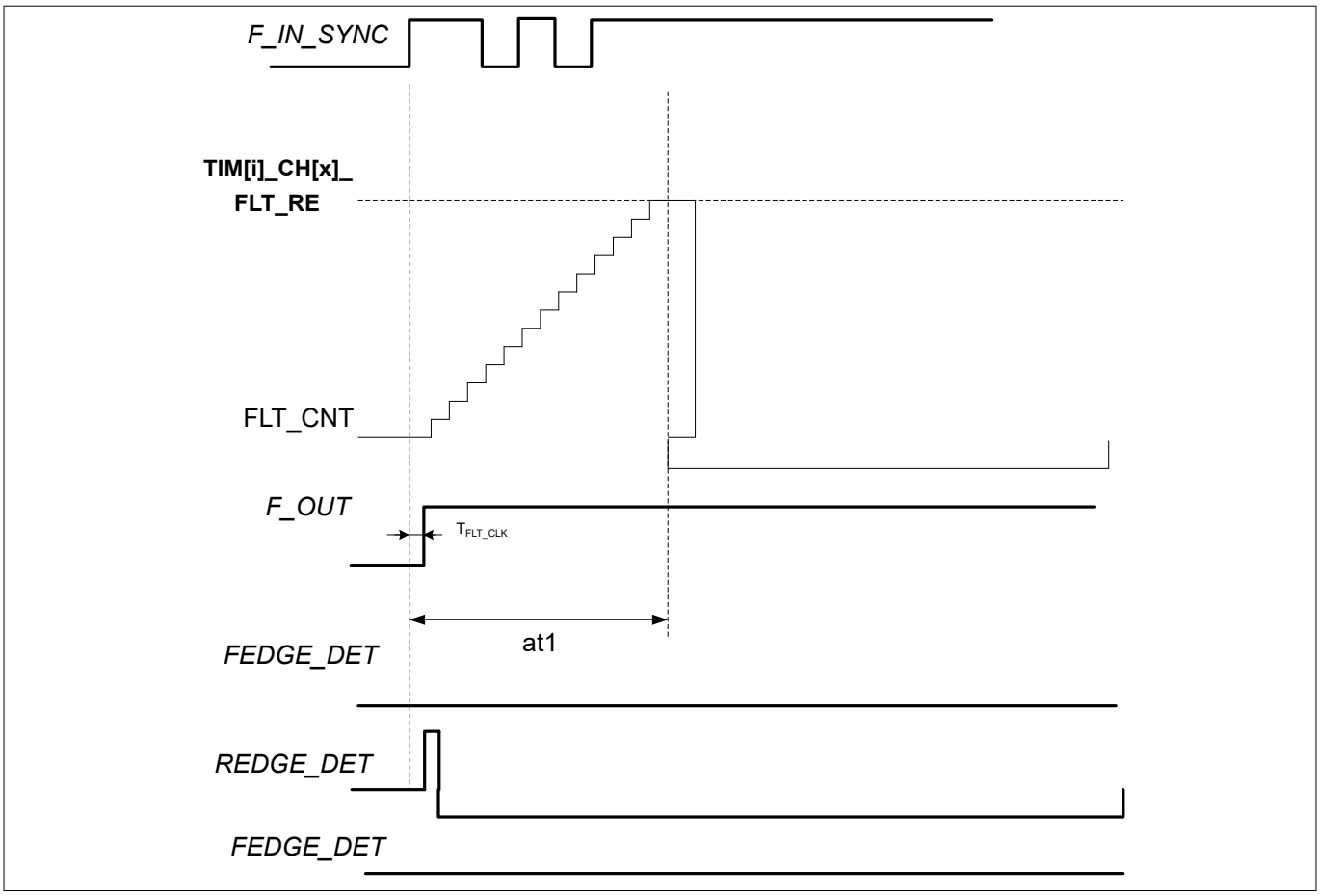


Figure 1046 Immediate edge propagation mode in the case of a rising edge

In immediate edge propagation mode, the glitch measurement mechanism is not applied to the edge detection. Detected edges on F_IN_SYNC are transferred directly to F_OUT.

The counter FLT_CNT is incremented until the acceptance time threshold is reached.

Figure 1047 shows a more complex example of the **TIM** filter, in which both, rising and falling edges are configured in immediate edge propagation mode.

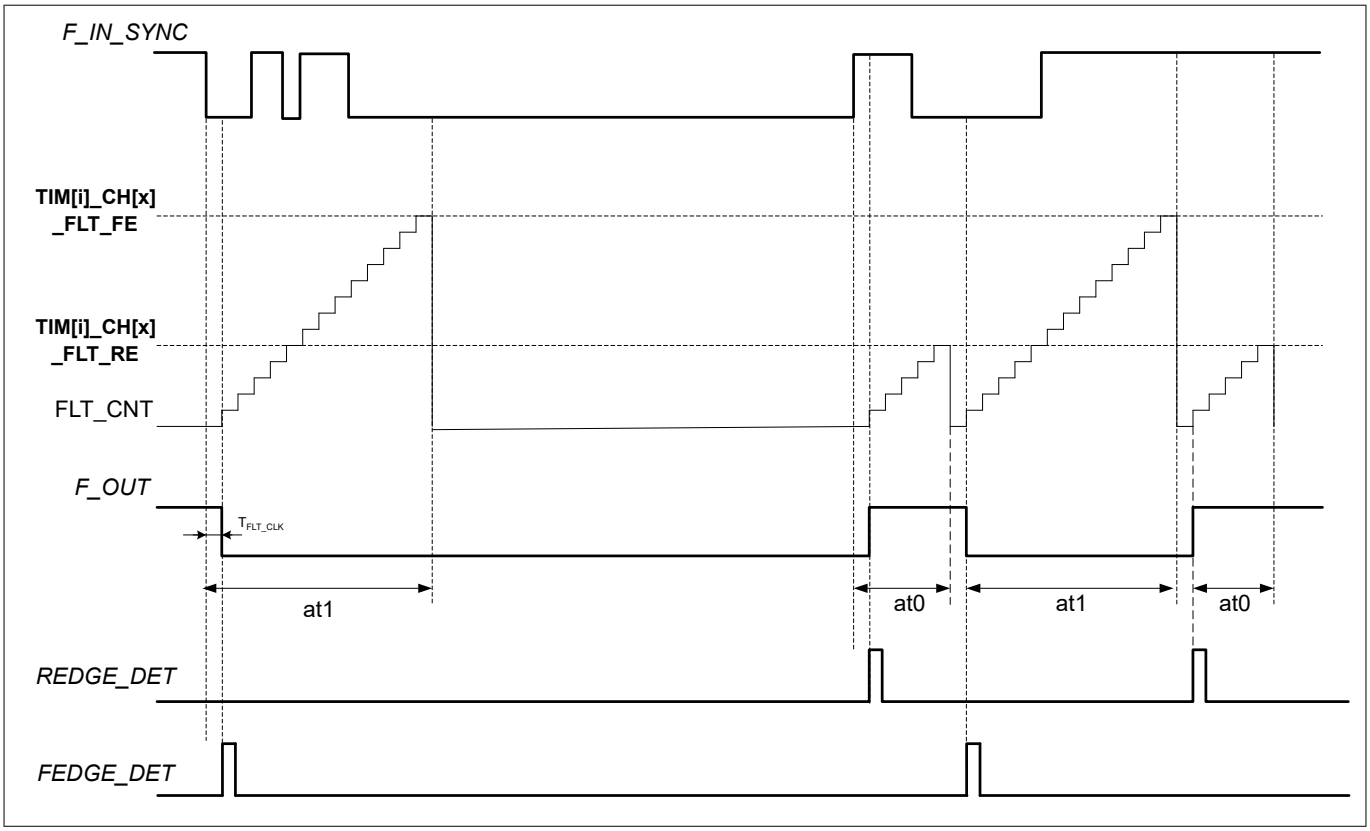


Figure 1047 Immediate edge propagation mode in the case of a rising and falling edge

If the FLT_CNT has reached the acceptance time for a specific signal edge and the signal F_IN_SYNC has already changed to the opposite level of F_OUT, the opposite signal level is set to F_OUT and the acceptance time measurement is started immediately. Figure 1047 shows this scenario at the detection of the first rising edge and the second falling edge.

Individual deglitch time mode (up/down counter)

In individual deglitch time mode (up/down counter) each edge of an input signal can be filtered with an individual deglitch threshold filter value mentioned in the registers CLSi_TIM_CHx_FLT_RE and CLSi_TIM_CHx_FLT_FE, respectively.

The filter counter register FLT_CNT is incremented when the signal level on F_IN_SYNC is unequal to the signal level on F_OUT and decremented if F_IN_SYNC equals F_OUT.

If FLT_CNT reaches zero due to decrementing, the counter is stopped immediately.

If a glitch is detected a glitch detection bit CLSi_TIM_CHx_IRQ_NOTIFY.GLITCHDET is set.

The detected edge signal together with the new signal level is propagated to F_OUT after the individual deglitch threshold is reached. Figure 1048 shows the behavior of the filter in individual deglitch time (up/down counter) mode in the case of the rising edge detection.

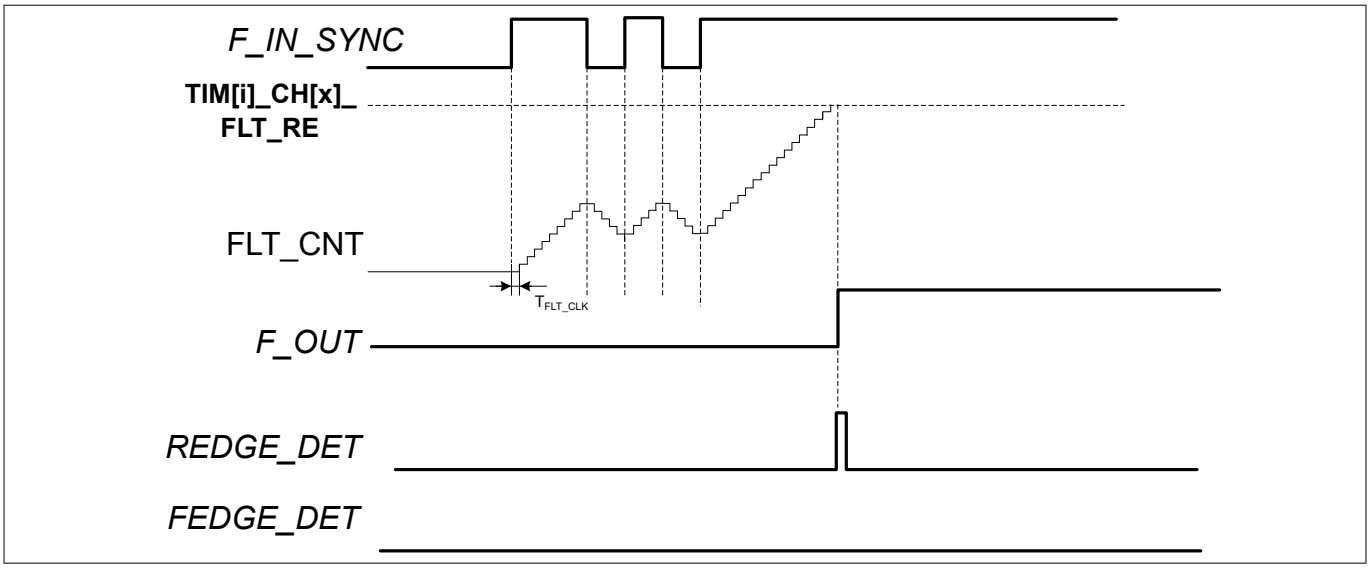


Figure 1048 Individual deglitch time mode (up/down counter) in the case of a rising edge

Individual deglitch time mode (hold counter)

In individual deglitch time mode (hold counter) each edge of an input signal can be filtered with an individual deglitch threshold filter value mentioned in the registers CLSi_TIM_CHx_FLT_RE and CLSi_TIM_CHx_FLT_FE, respectively.

The filter counter register FLT_CNT is incremented when the signal level on F_IN_SYNC is unequal to the signal level on F_OUT and the counter value of FLT_CNT is hold if F_IN equals F_OUT.

If a glitch is detected the glitch detection bit CLSi_TIM_CHx_IRQ_NOTIFY.GLITCHDET is set.

The detected edge signal together with the new signal level is propagated to F_OUT after the individual deglitch threshold is reached. Figure 1049 shows the behavior of the filter in individual deglitch time (hold counter) mode in the case of the rising edge detection.

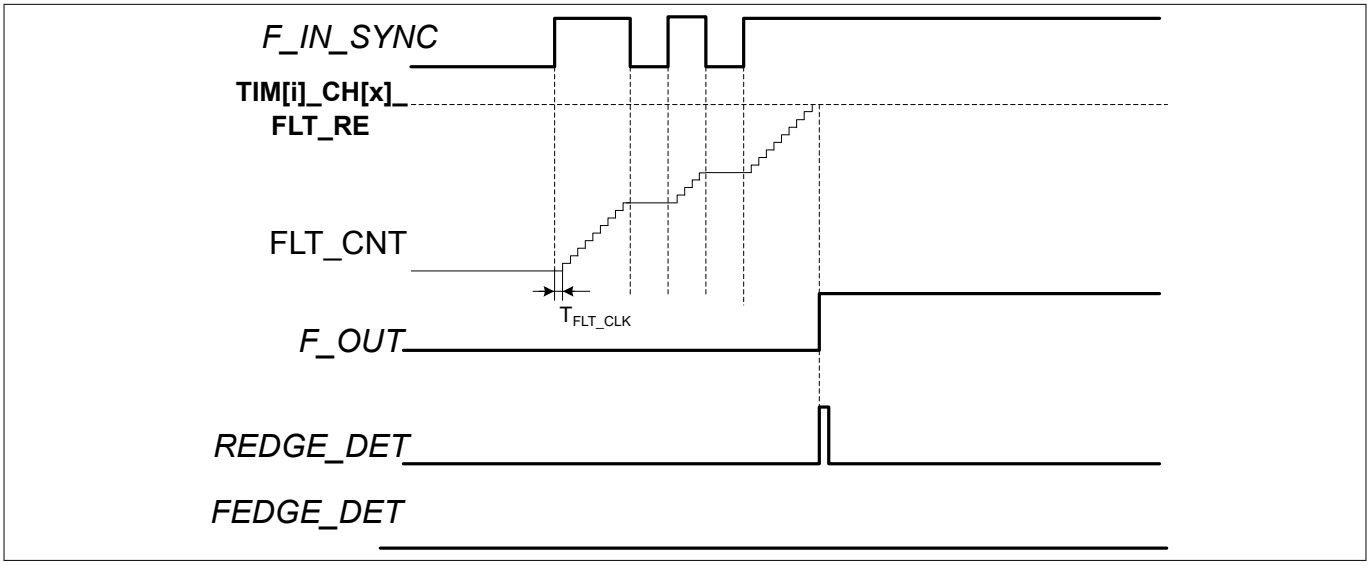


Figure 1049 Individual deglitch time mode (hold counter) in the case of a rising edge

Individual deglitch time mode (reset counter)

In individual deglitch time mode (reset counter) each edge of an input signal can be filtered with an individual deglitch threshold filter value mentioned in the registers CLSi_TIM_CHx_FLT_RE and CLSi_TIM_CHx_FLT_FE, respectively.

The filter counter register FLT_CNT is incremented when the signal level on F_IN_SYNC is unequal to the signal level on F_OUT and the counter value of FLT_CNT is reset to 000000_H if F_IN equals F_OUT.

If a glitch is detected the glitch detection bit CLSi_TIM_CHx_IRQ_NOTIFY.GLITCHDET is set.

The detected edge signal together with the new signal level is propagated to F_OUT after the individual deglitch threshold is reached. Figure 1050 shows the behavior of the filter in individual deglitch time (reset counter) mode in the case of the rising edge detection.

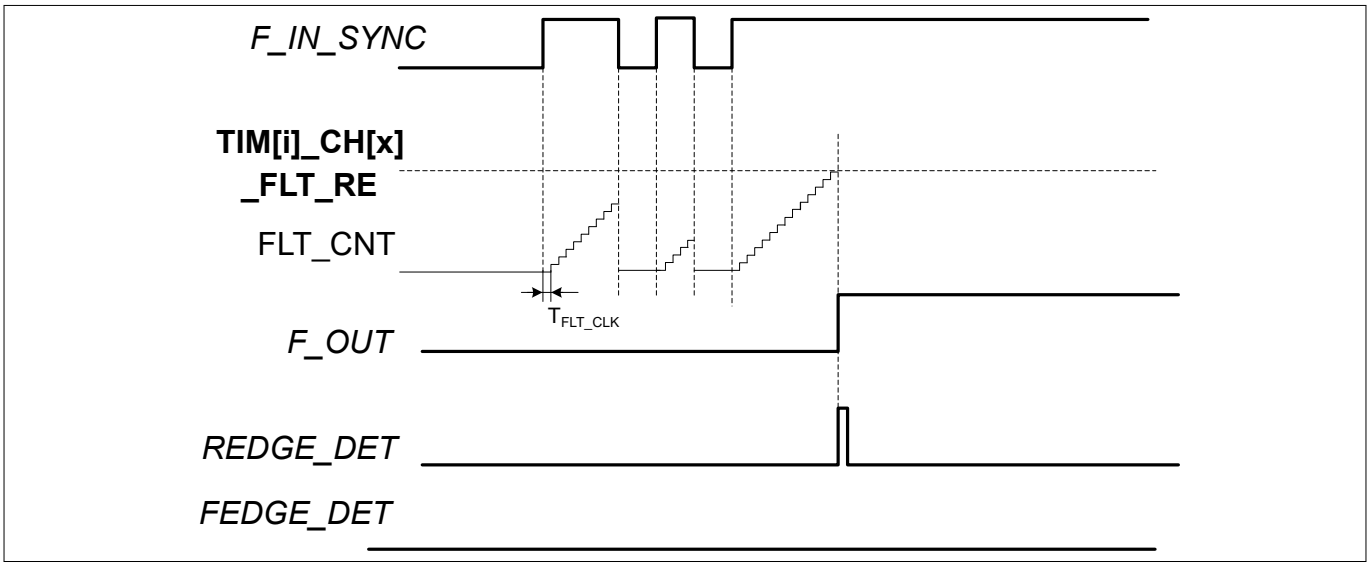


Figure 1050 Individual deglitch time mode (reset counter) in the case of a rising edge

Immediate edge propagation and individual deglitch mode

As already mentioned, the four different filter modes can be applied individually to each edge of the measured signal.

However, if one edge is configured with immediate edge propagation and the other edge with an individual deglitch mode (whether up/down counter, hold counter or reset counter) a special consideration has to be applied.

Assume that the rising edge is configured for immediate edge propagation and the falling edge with individual deglitch mode (up/down counter) as shown in Figure 1051.

If the falling edge of the incoming signal already occurs during the measuring of the acceptance time of the rising edge, the measurement of the deglitch time on the falling edge is started with delay, but immediately after the acceptance time measurement phase of the rising edge has finished.

Consequently, the deglitch counter cannot measure the time T_{ERROR}, as shown in Figure 1051.

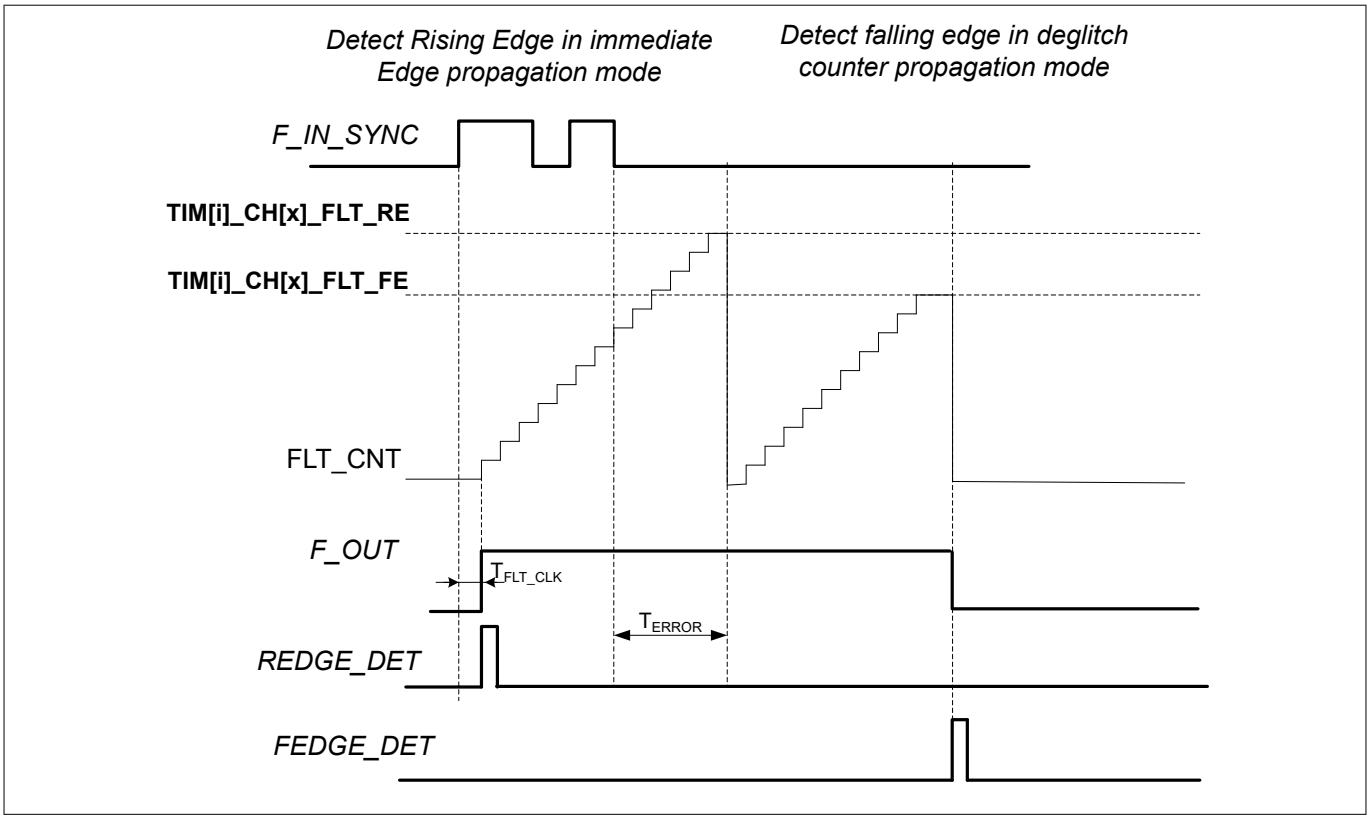


Figure 1051 Mixed mode measurement

TIM filter reconfiguration

If CLSi_TIM_CHx_CTRL.FLT_EN=1 a change of CLSi_TIM_CHx_FLT_RE or CLSi_TIM_CHx_FLT_FE will take place immediately.

If CLSi_TIM_CHx_CTRL.FLT_EN=1 a change of CLSi_TIM_CHx_CTRL.FLT_MODE_RE or CLSi_TIM_CHx_CTRL.FLT_MODE_FE will be used with the next occurring corresponding edge. If the mode is changed while the filter unit is processing a certain mode, it will end this edge filtering in the mode as started.

If CLSi_TIM_CHx_CTRL.FLT_EN=1 a change of CLSi_TIM_CHx_CTRL.FLT_CTR_RE, CLSi_TIM_CHx_CTRL.FLT_CTR_FE, CLSi_TIM_CHx_ECTRL.EFLT_CTR_RE or CLSi_TIM_CHx_ECTRL.EFLT_CTR_FE will take place immediately.

44.3.1.6.3 Timeout detection unit (TDU)

The Timeout Detection Unit (TDU) is responsible for timeout detection of the **TIM** input signals.

Each channel of the **TIM** sub-module has its own Timeout Detection Unit (TDU) where a timeout event can be set up on the filtered input signal of the corresponding channel.

In each timeout unit exist three 8-bit counter/comparator slices. A counter/comparator slice is shown below. The counter CLSi_TIM_CHx_TDUC.TO_CNT will increment by signal INC. The counter can be loaded with the value LOAD_VAL if LOAD = 1. GT_EVT will be 1 if CLSi_TIM_CHx_TDUC.TO_CNT > CLSi_TIM_CHx_TDUV.TOV is fulfilled. EQ_EVT will be 1 if CLSi_TIM_CHx_TDUC.TO_CNT = CLSi_TIM_CHx_TDUV.TOV is fulfilled.

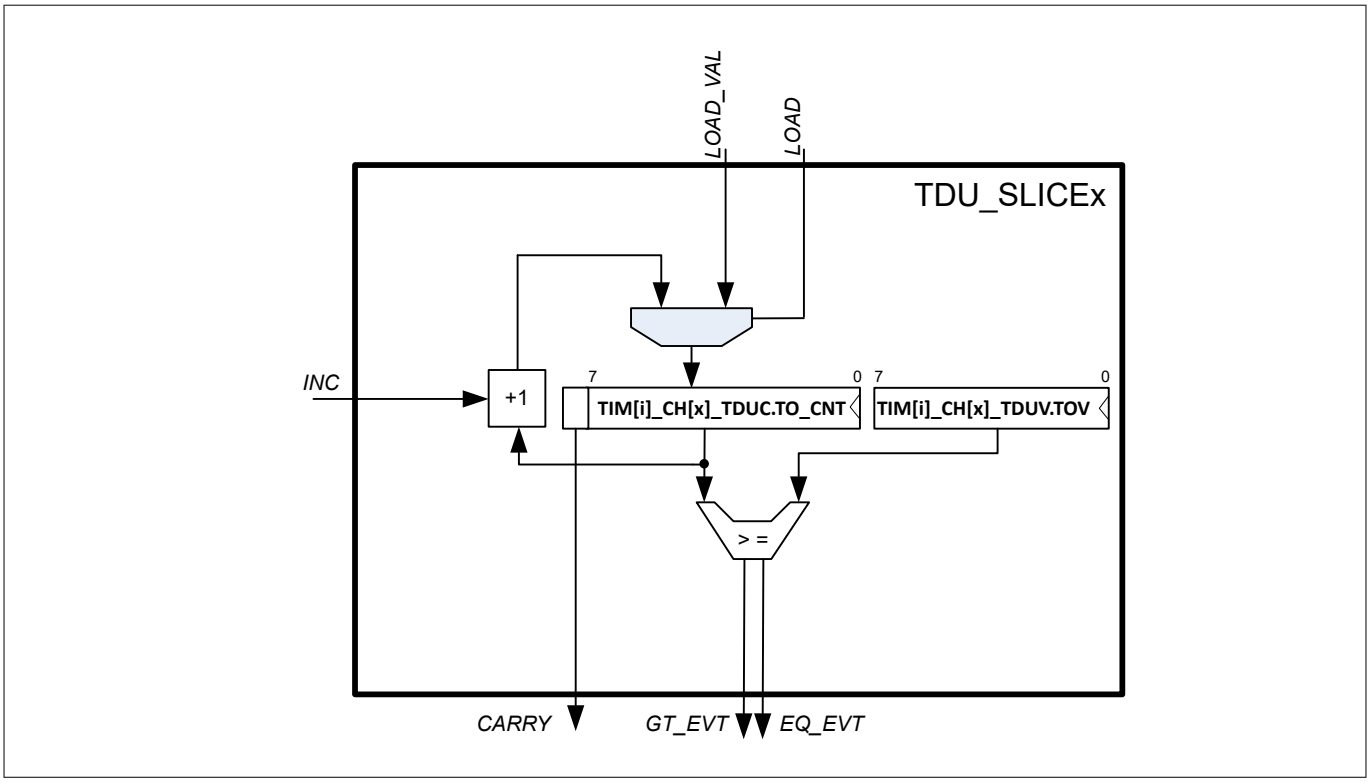


Figure 1052 Counter/comparator slice

The counter/comparator slices can be cascaded depending on the application needs to operate as:
three 8-bit counters
one 16-bit counter and one 8-bit counter
one 24-bit counter
two 8-bit counters.
This allows the user to use the functions
timeout on input signals
local **CMU** clock prescaler 8-bit
trigger event generation 8-bit (external capture, TIM_TODET_IRQ)
in parallel.
With usage of the three 8-bit counters it is possible to define different timeout values for the two signal levels.
Following table shows which functions can be used in parallel.

Table 2202 Used parallel functions

Counter type	Timeout functionality	Generate local TIM CMU clk	Source for external capture to previous channel	Source for TIM_TODET_IRQ
24-bit	24-bit	no	TDU_TIMEOUT_EVT, TDU_SAMPLE_EVT	TDU_TIMEOUT_EVT, TDU_SAMPLE_EVT
1 x 8-bit 1 x 16-bit	16-bit local clk TDU_SAMPLE_EVT usable	yes	TDU_TIMEOUT_EVT, TDU_FRAME_EVT, TDU_SAMPLE_EVT	TDU_TIMEOUT_EVT, TDU_FRAME_EVT, TDU_SAMPLE_EVT

(table continues...)

Table 2202 (continued) Used parallel functions

Counter type	Timeout functionality	Generate local TIM CMU clk	Source for external capture to previous channel	Source for TIM_TODET_IRQ
3 x 8-bit	8-bit local clk TDU_SAMPLE_EVT usable	yes	TDU_TIMEOUT_EVT, TDU_SAMPLE_EVT, TDU_WORD_EVT, TDU_FRAME_EVT	TDU_TIMEOUT_EVT, TDU_SAMPLE_EVT, TDU_WORD_EVT, TDU_FRAME_EVT
3 x 8-bit	no	yes	TDU_TIMEOUT_EVT, TDU_SAMPLE_EVT, TDU_WORD_EVT, TDU_FRAME_EVT	TDU_TIMEOUT_EVT, TDU_SAMPLE_EVT, TDU_WORD_EVT, TDU_FRAME_EVT
2 x 8-bit	no	no	TDU_TIMEOUT_EVT, TDU_WORD_EVT, TDU_FRAME_EVT	TDU_TIMEOUT_EVT, TDU_WORD_EVT, TDU_FRAME_EVT

Next table shows which of the available 8-bit resources are cascaded with a chosen CLSi_TIM_CHx_TDUV.SLICING.

Table 2203 Which of the available 8-bit resources are cascaded with a chosen SLICING

Counter type	Counters count on	Counter resource generates	CLK selection
24-bit	CNT on TCS	CLSi_TIM_CHx_CNT.CNT= CLSi_TIM_CHx_TDUC.TO_CNT2 & CLSi_TIM_CHx_TDUC.TO_CNT1 & CLSi_TIM_CHx_TDUC.TO_CNT; TCMP = CLSi_TIM_CHx_TDUV.TOV2 & CLSi_TIM_CHx_TDUV.TOV1 & CLSi_TIM_CHx_TDUV.TOV ; CLSi_TIM_CHx_CNT.CNT >= TCMP generates TDU_SAMPLE_EVT TDU_TIMEOUT_EVT = TDU_SAMPLE_EVT; TDU_FRAME_EVT = 0 TDU_WORD_EVT = 0	TCS selected

(table continues...)

Table 2203 (continued) Which of the available 8-bit resources are cascaded with a chosen SLICING

Counter type	Counters count on	Counter resource generates	CLK selection
3 x 8-bit	CLSi_TIM_CHx_TDUC.TO_CNT2 on TCS; CLSi_TIM_CHx_TDUC.TO_CNT on TDU_SAMPLE_EVT CLSi_TIM_CHx_TDUC.TO_CNT1 on TDU_WORD_EVT	CLSi_TIM_CHx_TDUC.TO_CNT2 >= CLSi_TIM_CHx_TDUV.TOV2 generates TDU_SAMPLE_EVT; CLSi_TIM_CHx_TDUC.TO_CNT >= CLSi_TIM_CHx_TDUV.TOV generates TDU_WORD_EVT CLSi_TIM_CHx_TDUC.TO_CNT1 >= CLSi_TIM_CHx_TDUV.TOV1 generates TDU_FRAME_EVT TDU_TIMEOUT_EVT = TDU_WORD_EVT	CLSi_TIM_CHx_TDUC.TO_CNT2: CLSi_TIM_CHx_TDUV.TCS selected; CLSi_TIM_CHx_TDUC.TO_CNT: TDU_SAMPLE_EVT selected with CLSi_TIM_CHx_TDUV.TCS_USE_SAMPLE_EVT=1 CLSi_TIM_CHx_TDUC.TO_CNT1: TDU_WORD_EVT selected with CLSi_TIM_CHx_TDUV.TDU_SAME_CNT_CLK=0
3 x 8-bit	CLSi_TIM_CHx_TDUC.TO_CNT2 on TCS; CLSi_TIM_CHx_TDUC.TO_CNT on TDU_SAMPLE_EVT; CLSi_TIM_CHx_TDUC.TO_CNT1 on TDU_SAMPLE_EVT	CLSi_TIM_CHx_TDUC.TO_CNT2 >= CLSi_TIM_CHx_TDUV.TOV2 generates TDU_SAMPLE_EVT; CLSi_TIM_CHx_TDUC.TO_CNT >= CLSi_TIM_CHx_TDUV.TOV generates TDU_WORD_EVT; CLSi_TIM_CHx_TDUC.TO_CNT1 >= CLSi_TIM_CHx_TDUV.TOV1 generates TDU_FRAME_EVT; TDU_TIMEOUT_EVT = TDU_WORD_EVT or TDU_FRAME_EVT	CLSi_TIM_CHx_TDUC.TO_CNT2: CLSi_TIM_CHx_TDUV.TCS selected; CLSi_TIM_CHx_TDUC.TO_CNT: TDU_SAMPLE_EVT selected with CLSi_TIM_CHx_TDUV.TCS_USE_SAMPLE_EVT=1; CLSi_TIM_CHx_TDUC.TO_CNT1: TDU_SAMPLE_EVT selected with CLSi_TIM_CHx_TDUV.TDU_SAME_CNT_CLK=1

(table continues...)

Table 2203 (continued) Which of the available 8-bit resources are cascaded with a chosen SLICING

Counter type	Counters count on	Counter resource generates	CLK selection
3 x 8-bit	CLSi_TIM_CHx_TDUC.TO_CNT2 on TCS; CLSi_TIM_CHx_TDUC.TO_CNT on TCS; CLSi_TIM_CHx_TDUC.TO_CNT1 on TDU_WORD_EVT	CLSi_TIM_CHx_TDUC.TO_CNT2 >= CLSi_TIM_CHx_TDUV.TOV2 generates TDU_SAMPLE_EVT; CLSi_TIM_CHx_TDUC.TO_CNT >= CLSi_TIM_CHx_TDUV.TOV generates TDU_WORD_EVT; CLSi_TIM_CHx_TDUC.TO_CNT1 >= CLSi_TIM_CHx_TDUV.TOV1 generates TDU_FRAME_EVT; TDU_TIMEOUT_EVT = TDU_WORD_EVT	CLSi_TIM_CHx_TDUC.TO_CNT2: CLSi_TIM_CHx_TDUV.TCS selected; CLSi_TIM_CHx_TDUC.TO_CNT: CLSi_TIM_CHx_TDUV.TCS selected with CLSi_TIM_CHx_TDUV.TCS_USE_SAMPLE_EVT=0; CLSi_TIM_CHx_TDUC.TO_CNT1: TDU_WORD_EVT selected with CLSi_TIM_CHx_TDUV.TDU_SAME_CNT_CLK=0
3 x 8-bit	CLSi_TIM_CHx_TDUC.TO_CNT2 on TCS; CLSi_TIM_CHx_TDUC.TO_CNT on TCS; CLSi_TIM_CHx_TDUC.TO_CNT1 on TCS	CLSi_TIM_CHx_TDUC.TO_CNT2 >= CLSi_TIM_CHx_TDUV.TOV2 generates TDU_SAMPLE_EVT; CLSi_TIM_CHx_TDUC.TO_CNT >= CLSi_TIM_CHx_TDUV.TOV generates TDU_WORD_EVT; CLSi_TIM_CHx_TDUC.TO_CNT1 >= CLSi_TIM_CHx_TDUV.TOV1 generates TDU_FRAME_EVT; TDU_TIMEOUT_EVT = TDU_WORD_EVT or TDU_FRAME_EVT	CLSi_TIM_CHx_TDUC.TO_CNT2: CLSi_TIM_CHx_TDUV.TCS selected; CLSi_TIM_CHx_TDUC.TO_CNT: CLSi_TIM_CHx_TDUV.TCS selected with CLSi_TIM_CHx_TDUV.TCS_USE_SAMPLE_EVT=0; CLSi_TIM_CHx_TDUC.TO_CNT1: CLSi_TIM_CHx_TDUV.TCS selected with CLSi_TIM_CHx_TDUV.TDU_SAME_CNT_CLK=1

(table continues...)

Table 2203 (continued) Which of the available 8-bit resources are cascaded with a chosen SLICING

Counter type	Counters count on	Counter resource generates	CLK selection
2 x 8-bit	CLSi_TIM_CHx_TDUC.TO_CNT on TCS; CLSi_TIM_CHx_TDUC.TO_CNT1 on TDU_WORD_EVT	CLSi_TIM_CHx_TDUC.TO_CNT >= CLSi_TIM_CHx_TDUV.TOV generates TDU_WORD_EVT; CLSi_TIM_CHx_TDUC.TO_CNT1 >= CLSi_TIM_CHx_TDUV.TOV 1 generates TDU_FRAME_EVT; TDU_TIMEOUT_EVT = TDU_WORD_EVT; TDU_SAMPLE_EVT = 0	CLSi_TIM_CHx_TDUC.TO_CNT: CLSi_TIM_CHx_TDUV.TCS selected; CLSi_TIM_CHx_TDUC.TO_CNT1: TDU_WORD_EVT selected with CLSi_TIM_CHx_TDUV.TDU_SAME_CNT_CLK =0
2 x 8-bit	CLSi_TIM_CHx_TDUC.TO_CNT on TCS; CLSi_TIM_CHx_TDUC.TO_CNT1 on TCS	CLSi_TIM_CHx_TDUC.TO_CNT >= CLSi_TIM_CHx_TDUV.TOV generates TDU_WORD_EVT; CLSi_TIM_CHx_TDUC.TO_CNT1 >= CLSi_TIM_CHx_TDUV.TOV 1 generates TDU_FRAME_EVT; TDU_TIMEOUT_EVT = TDU_WORD_EVT; TDU_SAMPLE_EVT = 0	CLSi_TIM_CHx_TDUC.TO_CNT: CLSi_TIM_CHx_TDUV.TCS selected; CLSi_TIM_CHx_TDUC.TO_CNT1: CLSi_TIM_CHx_TDUV.TCS selected with CLSi_TIM_CHx_TDUV.TDU_SAME_CNT_CLK=1
1 x 8-bit 1 x 16-bit	CLSi_TIM_CHx_TDUC.TO_CNT2 on TCS; CLSi_TIM_CHx_CNT.CNT on TCS	CLSi_TIM_CHx_TDUC.TO_CNT2 >= CLSi_TIM_CHx_TDUV.TOV 2 generates TDU_SAMPLE_EVT; CNT = CLSi_TIM_CHx_TDUC.TO_CNT1 & CLSi_TIM_CHx_TDUC.TO_CNT; TCMP = CLSi_TIM_CHx_TDUV.TOV 1 & CLSi_TIM_CHx_TDUV.TOV ; CNT >= TCMP generates TDU_FRAME_EVT; TDU_TIMEOUT_EVT = TDU_FRAME_EVT; TDU_WORD_EVT = 0	CLSi_TIM_CHx_TDUC.TO_CNT2: CLSi_TIM_CHx_TDUV.TCS selected; CLSi_TIM_CHx_CNT.CNT: CLSi_TIM_CHx_TDUV.TCS selected with CLSi_TIM_CHx_TDUV.TCS_USE_SAMPLE_EVT=0

(table continues...)

Table 2203 (continued) Which of the available 8-bit resources are cascaded with a chosen SLICING

Counter type	Counters count on	Counter resource generates	CLK selection
1 x 8-bit 1 x 16-bit	CLSi_TIM_CHx_TDUC.TO_CNT2 on TCS; CLSi_TIM_CHx_CNT.CNT on TDU_SAMPLE_EVT	CLSi_TIM_CHx_TDUC.TO_CNT2 >= CLSi_TIM_CHx_TDUV.TOV2 generates TDU_SAMPLE_EVT; CNT = CLSi_TIM_CHx_TDUC.TO_CNT1 & CLSi_TIM_CHx_TDUC.TO_CNT; TCMP = CLSi_TIM_CHx_TDUV.TOV1 & CLSi_TIM_CHx_TDUV.TOV ; CNT >= TCMP generates TDU_FRAME_EVT; TDU_TIMEOUT_EVT = TDU_FRAME_EVT; TDU_WORD_EVT = 0	CLSi_TIM_CHx_TDUC.TO_CNT2: CLSi_TIM_CHx_TDUV.TCS selected; CLSi_TIM_CHx_CNT.CNT: TDU_SAMPLE_EVT selected with CLSi_TIM_CHx_TDUV.TCS_USE_SAMPLE_EVT=1

Based on a chosen counter configuration by CLSi_TIM_CHx_TDUV.SLICING it is possible to control the start behavior of the counters by CLSi_TIM_CHx_ECTRL.TDU_START in multiple ways. In addition, the stopping of the counters can be controlled by CLSi_TIM_CHx_ECTRL.TDU_STOP. Depending on the application needs it can be decided how the individual counter slices can be reset/reloaded by the configuration field CLSi_TIM_CHx_ECTRL.TDU_RESYNC.

Depending on the counter configuration, up to 4 internal compare events TDU_TIMEOUT_EVT, TDU_SAMPLE_EVT, TDU_WORD_EVT, TDU_FRAME_EVT can be generated out of the 3 comparator slices. CLSi_TIM_CHx_ECTRL.TODET_IRQ_SRC can choose, which event to be used as TIM_TODET_IRQ[x:x] signal which will be accessible by the CLSi_TIM_CHx_IRQ_NOTIFY.TODET bit.

The TDU architecture is shown in Figure 1053.

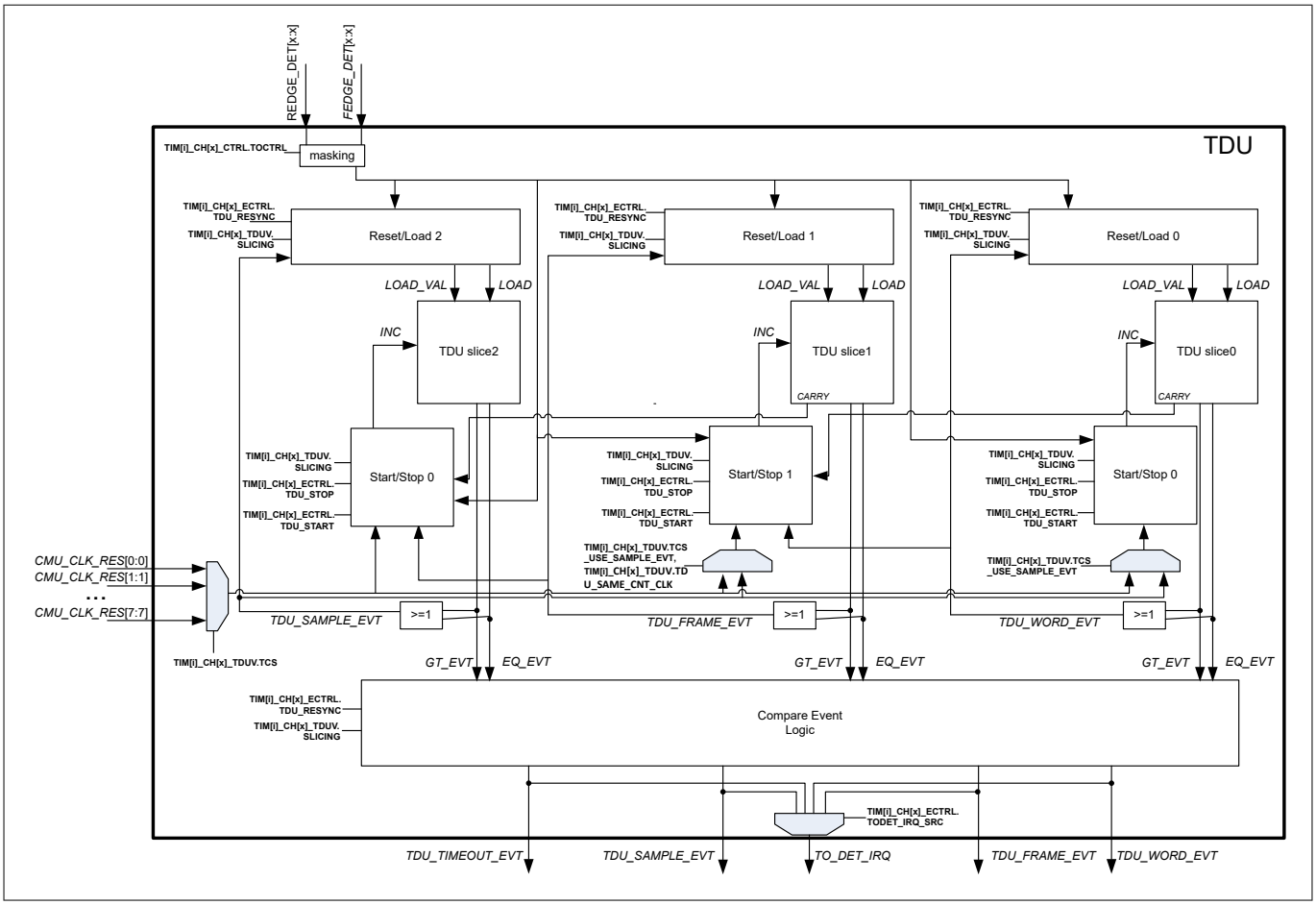


Figure 1053 Architecture of the TDU sub-unit

Each TDU_slice has its own start/stop control, based on the chosen configuration it will decide if the counter inside the TDU slice will increment on the resolution of the applied clock/event. The reset/load control decides based on the configuration settings and the compare result of the TDU slices those the counters CLSi_TIM_CHx_TDUC.TO_CNT, CLSi_TIM_CHx_TDUC.TO_CNT1, CLSi_TIM_CHx_TDUC.TO_CNT2 have to be reloaded. Depending on the chosen counter/compare configuration the compare event logic will generate based on the compare results of the 3 TDU slices and the chosen resolution the events TDU_SAMPLE_EVT, TDU_WORD_EVT, TDU_FRAME_EVT.

The primary resolution on which the TDU is working can be specified with the bit field CLSi_TIM_CHx_TDUV.TCS of the register CLSi_TIM_CHx_TDUV. The corresponding input signal CCM[i]_CLK_RES[x:x] will be used to clock the TDU. The individual timeout/counter values have to be specified in number of ticks of the selected input clock signal in the fields CLSi_TIM_CHx_TDUV.TOV, CLSi_TIM_CHx_TDUV.TOV1, CLSi_TIM_CHx_TDUV.TOV2 of the timeout value register CLSi_TIM_CHx_TDUV of the **TIM** channel x.

In case of cascading the bit slices by usage of CLSi_TIM_CHx_TDUV.SLICING and CLSi_TIM_CHx_TDUV.TCS_USE_SAMPLE_EVT and CLSi_TIM_CHx_TDUV.TDU_SAME_CNT_CLK the resolution for counting can be switched to the events TDU_SAMPLE_EVT or TDU_WORD_EVT. For more details see table above.

The counter compare units start operation on occurrence of the first "start event" configured by CLSi_TIM_CHx_ECTRL.TDU_START. They continue their operation until the first "stop event" configured by CLSi_TIM_CHx_ECTRL.TDU_STOP occurs.

In case of occurrence of a start event and a compare/count resolution event in the same clock-cycle, the counters will increment or reload/reset based on CLSi_TIM_CHx_ECTRL.TDU_RESYNC immediately. No TDU_SAMPLE_EVT, TDU_WORD_EVT, TDU_FRAME_EVT will be generated.

In case of occurrence of a stop event the counters will not change their values. In case of occurrence of a stop event and a compare/count resolution event in the same clock-cycle the corresponding events TDU_SAMPLE_EVT, TDU_WORD_EVT, TDU_FRAME_EVT will be generated.

In case of occurrence of a start event and a stop event in the same clock-cycle the counters will not change their values. No TDU_SAMPLE_EVT, TDU_WORD_EVT, TDU_FRAME_EVT will be generated.

The function of the timeout unit (configured to CLSi_TIM_CHx_ECTRL.TDU_RESYNC=0000_B, CLSi_TIM_CHx_ECTRL.TDU_START=000_B) can be started or stopped by setting/resetting the CLSi_TIM_CHx_CTRL.TOCTRL bit.

Timeout detection can be enabled to be sensitive to falling, rising or both edges of the input signal by writing the corresponding values to the bit field CLSi_TIM_CHx_CTRL.TOCTRL.

***Note**: The used signal names REDGE_DET[x:x] and FEDGE_DET[x:x] are related to the local channel x or the previous channel x-1, depending on the value of the configuration bit field CLSi_TIM_CHx_ECTRL.USE_PREV_TDU_IN.*

The TDU generates an interrupt signal TIM_TODET_IRQ[x:x] whenever a timeout is detected for an individual input signal and the CLSi_TIM_CHx_IRQ_NOTIFY.TODET bit.

Since the **TIM** channel still monitors its input pin although the timeout happened, a valid edge could occur at the input pin while the timeout information is still valid. In that case, the new edge associated data is stored inside the registers CLSi_TIM_CHx_GPR0 and CLSi_TIM_CHx_GPR1.

44.3.1.6.4 TIM channel architecture

Overview

Each **TIM** channel consists of an input edge counter CLSi_TIM_CHx_ECNT, a Signal Measurement Unit (SMU) with a counter CLSi_TIM_CHx_CNT, a counter shadow register CLSi_TIM_CHx_CNTS for SMU counter and two general-purpose registers CLSi_TIM_CHx_GPR0 and CLSi_TIM_CHx_GPR1 for value storage.

The value CLSi_TIM_CHx_TDUV.TOV of the timeout register CLSi_TIM_CHx_TDUV is provided to TDU sub-unit of each individual channel for timeout measurement. The architecture of the **TIM** channel is depicted in Figure 1054.

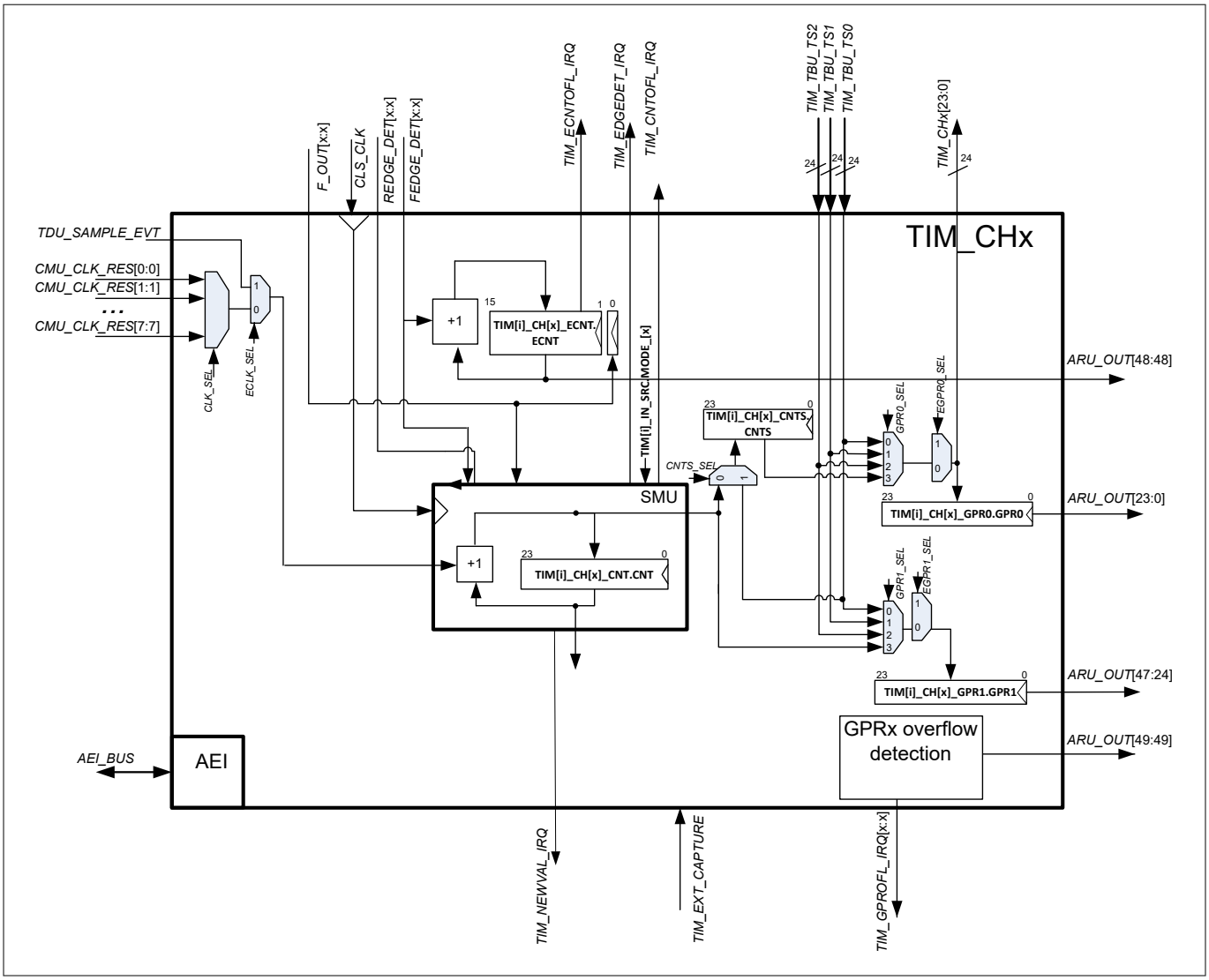


Figure 1054 TIM channel architecture

Each **TIM** channel receives both input trigger signals REDGE_DET[x:x] and FEDGE_DET[x:x], generated by the corresponding filter module in order to signalize a detected echo of the input signal F_IN[x:x]. The signal F_OUT[x:x] shows the filtered signal of the channel's input signal F_IN[x:x].

***Note**: In the following chapters for the different channel modes - except the TBCM mode - the used signal names F_OUT[x:x], REDGE_DET[x:x] and FEDGE_DET[x:x] are related to the local channel x or the previous channel x-1, depending on the value of the configuration bit field CLSi_TIM_CHx_ECTRL.USE_PREV_CH_IN. In TBCM mode, the signals do not depend on the value of the configuration bit field CLSi_TIM_CHx_ECTRL.USE_PREV_CH_IN.*

The edge counter CLSi_TIM_CHx_ECNT counts every incoming filtered edge (rising and falling). The counter value is uneven in case of detected rising and even in case of detected falling edge. Thus, the input signal level is part of the counter and can be obtained by bit 0 of CLSi_TIM_CHx_ECNT. (However, the actual counter implementation counts only falling edges on CLSi_TIM_CHx_ECNT[n:1] bits. It generates CLSi_TIM_CHx_ECNT.ECNT by composing the CLSi_TIM_CHx_ECNT.ECNT[n:1] bits with F_OUT[x:x] as bit 0).

Thus, the whole CLSi_TIM_CHx_ECNT counter value is always odd, when a positive edge was received and always even, when a negative edge was received.

The current CLSi_TIM_CHx_ECNT[7:0] register content is made visible on the bits 31 down to 24 of the registers CLSi_TIM_CHx_GPR0, CLSi_TIM_CHx_GPR1, and CLSi_TIM_CHx_CNTS. This allows the software to detect

inconsistent read-accesses to registers CLSi_TIM_CHx_GPR0, CLSi_TIM_CHx_GPR1, and CLSi_TIM_CHx_CNTS. However, the update strategy of these registers depends on the selected **TIM** modes, and thus the consistency check has to be adapted carefully.

It can be chosen with the bit field CLSi_TIM_CHx_CTRL.FR_ECNT_OFL when an overflow is signaled on CLSi_TIM_CHx_IRQ_NOTIFY.ECNTOFL. An CLSi_TIM_CHx_ECNT overflow can be signaled on 8-bit or full range resolution (wrap around FF_H or $FFFF_H$ to 0_H due to increment)

While reading the bit CLSi_TIM_CHx_ECNT.ECNT[0:0] shows the input signal value F_OUT[x:x] independent of the state (enabled / disabled) of the channel. If a channel gets disabled (OSM mode or resetting CLSi_TIM_CHx_CTRL.TIM_EN) the content of CLSi_TIM_CHx_ECNT will be frozen until a read of the register takes place. This read will reset the CLSi_TIM_CHx_ECNT counter. Continuing reads will show the input signal value in bit CLSi_TIM_CHx_ECNT.ECNT[0:0] again.

If a channel gets enabled (CLSi_TIM_CHx_CTRL.TIM_EN = 1), the registers CLSi_TIM_CHx_ECNT.ECNT[15:1], CLSi_TIM_CHx_CNT, CLSi_TIM_CHx_GPR0 and CLSi_TIM_CHx_GPR1 are set to the initial values. In case an incoming signal edge occurs in the same cluster-clock cycle as the channel enabling, the registers CLSi_TIM_CHx_ECNT.ECNT[15:1], CLSi_TIM_CHx_GPR0 and CLSi_TIM_CHx_GPR1 are set to the initial value and the register CLSi_TIM_CHx_CNT is increased or a new value is shifted (in TSSM mode) based on the initial value.

When new data is written into CLSi_TIM_CHx_GPR0 and CLSi_TIM_CHx_GPR1 the CLSi_TIM_CHx_IRQ_NOTIFY.NEWVAL bit is set and depending on corresponding enable bit value the TIM_NEWVAL_IRQ[x:x] interrupt is raised.

To guarantee a consistent delivery of data from the CLSi_TIM_CHx_GPR0 and CLSi_TIM_CHx_GPR1 registers to the CPU each **TIM** channel has to ensure that the data is consumed before it is overwritten with new values.

If new data was produced by the **TIM** channel (bit CLSi_TIM_CHx_IRQ_NOTIFY.NEWVAL is set) while the old data is not consumed , the **TIM** channel sets the CLSi_TIM_CHx_IRQ_NOTIFY.GPROFL bit and it overwrites the data inside the registers CLSi_TIM_CHx_GPR0 and CLSi_TIM_CHx_GPR1.

The acknowledge for reading out data is performed by a read-access to the register CLSi_TIM_CHx_GPR0. Thus, register CLSi_TIM_CHx_GPR1 should be read always before CLSi_TIM_CHx_GPR0.

TIM channel modes

The **TIM** provides seven different measurement modes that can be configured with the bit field CLSi_TIM_CHx_CTRL.TIM_MODE of register CLSi_TIM_CHx_CTRL. The measurement modes are described in the following subsections. Besides these different basic measurement modes, there exist distinct configuration bits in the register CLSi_TIM_CHx_CTRL for a more detailed controlling of each mode. The meanings of these bits are as follows:

CLSi_TIM_CHx_CTRL.DSL: control the signal level for the measurement modes (e.g. if a measurement is started with rising edge or falling edge, or if high or low level pulses are measured.

CLSi_TIM_CHx_CTRL.EGPR0_SEL, CLSi_TIM_CHx_CTRL.GPR0_SEL and CLSi_TIM_CHx_CTRL.EGPR1_SEL, CLSi_TIM_CHx_CTRL.GPR1_SEL: control the actual content of the registers CLSi_TIM_CHx_GPR0 and CLSi_TIM_CHx_GPR1 after a measurement has finished.

CLSi_TIM_CHx_CTRL.CNTS_SEL: control the content of the registers CLSi_TIM_CHx_CNTS. The actual time for updating the CLSi_TIM_CHx_CNTS register is mode dependent.

CLSi_TIM_CHx_CTRL.OSM: activate measurement in one-shot mode or continuous mode. In one-shot mode only one measurement cycle is performed and after that the channel is disabled.

CLSi_TIM_CHx_IRQ_NOTIFY.NEWVAL: The NEWVAL **IRQ** interrupt is triggered at the end of a measurement cycle, signaling that the registers CLSi_TIM_CHx_GPR0 and CLSi_TIM_CHx_GPR1 are updated.

CLSi_TIM_CHx_CTRL.EXT_CAP_EN: forces an update of the registers CLSi_TIM_CHx_GPR0 and CLSi_TIM_CHx_GPR1 and CLSi_TIM_CHx_CNTS (**TIM** channel mode dependent) only on each rising edge of the TIM_EXT_CAPTURE[x:x] signal and triggers a TIM_NEWVAL_IRQ[x:x] interrupt. If this mode is disabled the TIM_NEWVAL_IRQ[x:x] interrupt is triggered at the end of each measurement cycle.

For each channel, the source of the TIM_EXT_CAPTURE signal can be configured with the bit fields CLSi_TIM_CHx_ECTRL.EXT_CAP_SRC in the register CLSi_TIM_CHx_ECTRL.

TIM PWM measurement mode (TPWM)

In **TIM** PWM Measurement Mode, the **TIM** channel measures duty cycle and period of an incoming PWM signal. The CLSi_TIM_CHx_CTRL.DSL bit defines the polarity of the PWM signal to be measured.

When measurement of pulse high time and period is requested (PWM with a level high duty cycle, CLSi_TIM_CHx_CTRL.DSL=1) and CLSi_TIM_CHx_ECTRL.IMM_START=0, the channel starts measuring after the first rising edge is detected by the filter.

If CLSi_TIM_CHx_ECTRL.IMM_START=1 the measurement starts immediately after activating the channel by CLSi_TIM_CHx_CTRL.TIM_EN=1.

Measurement is done with the CLSi_TIM_CHx_CNT register counting with the configured clock coming from CCM[i]_CLK_RES[x:x] until a falling edge is detected.

Assume: CLSi_TIM_CHx_ECTRL.SWAP_CAPTURE=0, CLSi_TIM_CHx_CTRL.ECNT_RESET=0

Then the counter value is stored inside the shadow register CLSi_TIM_CHx_CNTS (if CLSi_TIM_CHx_CTRL.CNTS_SEL = 0) and the counter CLSi_TIM_CHx_CNT counts continuously until the next rising edge is reached.

On this following rising edge the content of the CLSi_TIM_CHx_CNTS register is transferred to CLSi_TIM_CHx_GPR0 and the content of CLSi_TIM_CHx_CNT register is transferred to CLSi_TIM_CHx_GPR1, assuming settings for the selectors CLSi_TIM_CHx_CTRL.EGPR0_SEL=0, CLSi_TIM_CHx_CTRL.GPR0_SEL=11_B and CLSi_TIM_CHx_CTRL.EGPR1_SEL=0, CLSi_TIM_CHx_CTRL.GPR1_SEL=11_B. By this, CLSi_TIM_CHx_GPR0 contains the pulse width length and CLSi_TIM_CHx_GPR1 contains the period.

***Note**: The bits CLSi_TIM_CHx_ECNT[7:1] may be used to check data consistency of the registers CLSi_TIM_CHx_GPR0 and CLSi_TIM_CHx_GPR1.*

In addition, the CLSi_TIM_CHx_CNT register is cleared CLSi_TIM_CHx_IRQ_NOTIFY.NEWVAL status bit and depending on corresponding interrupt enable condition TIM_NEWVAL_IRQ[x:x] interrupt is raised.

The CLSi_TIM_CHx_CNTS register update is not performed until the measurement is started. Afterwards each edge leaving the level defined by CLSi_TIM_CHx_CTRL.DSL is performing a CLSi_TIM_CHx_CNTS register update.

If a PWM with a level low duty cycle should be measured (CLSi_TIM_CHx_CTRL.DSL = 0) and CLSi_TIM_CHx_ECTRL.IMM_START=0, the channel waits for a falling edge until measurement is started. On this edge the level low duty cycle time is stored first in CLSi_TIM_CHx_CNTS and then finally in CLSi_TIM_CHx_GPR0 and the period is stored in CLSi_TIM_CHx_GPR1.

When a PWM period was successfully measured, the data in the registers CLSi_TIM_CHx_GPR0 and CLSi_TIM_CHx_GPR1 is marked as valid for reading when the CLSi_TIM_CHx_IRQ_NOTIFY.NEWVAL bit is set and a new measurement is started.

If the preceding PWM values were not consumed the **TIM** channel set CLSi_TIM_CHx_IRQ_NOTIFY.GPROFL status bit and depending on corresponding interrupt enable bit value raises a TIM_GPROFL_IRQ[x:x] and overwrites the old values in CLSi_TIM_CHx_GPR0 and CLSi_TIM_CHx_GPR1. A new measurement is started afterwards.

If the register CLSi_TIM_CHx_CNT produces an overflow (wrap around $FFFFFF_H$ to 0_H due to increment) during the measurement, the bit CLSi_TIM_CHx_IRQ_NOTIFY.CNTOFL is set and interrupt TIM_CNTOFL_IRQ[x:x] is raised depending on corresponding interrupt enable condition.

If the register CLSi_TIM_CHx_ECNT produces an overflow during the measurement, the bit CLSi_TIM_CHx_IRQ_NOTIFY.ECNTOFL is set and interrupt TIM_ECNTOFL_IRQ[x:x] is raised depending on corresponding interrupt enable condition.

If CLSi_TIM_CHx_CTRL.ECNT_RESET=0 the counter CLSi_TIM_CHx_CNT will be reset to 0 on active edge (defined by CLSi_TIM_CHx_CTRL.DSL) of the input signal. If CLSi_TIM_CHx_CTRL.ECNT_RESET=1 the counter CLSi_TIM_CHx_CNT will be reset to 0 on each edge of the input signal.

Assume CLSi_TIM_CHx_CTRL.EXT_CAP_EN=0 and CLSi_TIM_CHx_ECTRL.SWAP_CAPTURE=0:

On every input edge to the active level defined by CLSi_TIM_CHx_CTRL.DSL will capture the data selected by CLSi_TIM_CHx_CTRL.EGPR1_SEL, CLSi_TIM_CHx_CTRL.GPR1_SEL to the registers CLSi_TIM_CHx_GPR1. Every edge to the inactive level will capture the data selected by CLSi_TIM_CHx_CTRL.CNTS_SEL to the registers CLSi_TIM_CHx_CNTS.

Assume CLSi_TIM_CHx_CTRL.EXT_CAP_EN=0 and CLSi_TIM_CHx_ECTRL.SWAP_CAPTURE=1:

On every input edge to the inactive level defined by CLSi_TIM_CHx_CTRL.DSL will capture the data selected by CLSi_TIM_CHx_CTRL.EGPR1_SEL, CLSi_TIM_CHx_CTRL.GPR1_SEL to the registers CLSi_TIM_CHx_GPR1. Every edge to the active level will capture the data selected by CLSi_TIM_CHx_CTRL.CNTS_SEL to the registers CLSi_TIM_CHx_CNTS.

External capture TIM PWM measurement mode (TPWM)

If external capture is enabled CLSi_TIM_CHx_CTRL.EXT_CAP_EN=1, the PWM measurement is done continuously. The actual measurement values are captured to CLSi_TIM_CHx_GPR0, CLSi_TIM_CHx_GPR1 if an external capture event occurs.

On every external capture event the data selected by CLSi_TIM_CHx_CTRL.CNTS_SEL, CLSi_TIM_CHx_CTRL.EGPR0_SEL, CLSi_TIM_CHx_CTRL.GPR0_SEL will be captured to the registers CLSi_TIM_CHx_CNTS, CLSi_TIM_CHx_GPR0.

If CLSi_TIM_CHx_ECTRL.SWAP_CAPTURE=0 every external capture event will capture the data selected by CLSi_TIM_CHx_CTRL.EGPR1_SEL, CLSi_TIM_CHx_CTRL.GPR1_SEL to the registers CLSi_TIM_CHx_GPR1. Every input edge to the level != CLSi_TIM_CHx_CTRL.DSL will capture the data selected by CLSi_TIM_CHx_CTRL.CNTS_SEL to the registers CLSi_TIM_CHx_CNTS.

If CLSi_TIM_CHx_ECTRL.SWAP_CAPTURE=1 every input edge to the inactive level != CLSi_TIM_CHx_CTRL.DSL will capture the data selected by CLSi_TIM_CHx_CTRL.EGPR1_SEL, CLSi_TIM_CHx_CTRL.GPR1_SEL to the registers CLSi_TIM_CHx_GPR1.

Assume CLSi_TIM_CHx_ECTRL.SWAP_CAPTURE=0:

Operation is done depending on **CMU** clock, CLSi_TIM_CHx_CTRL.ISL, CLSi_TIM_CHx_CTRL.DSL bit and the input signal value defined in next table (Assume CLSi_TIM_CHx_CTRL.CNTS_SEL= 0):

Table 2204 Operation depending on CMU clock, CLSi_TIM_CHx_CTRL.ISL, CLSi_TIM_CHx_CTRL.DSL and the input signal value (assume CLSi_TIM_CHx_CTRL.CNTS_SEL= 0)

Input signal F_OUT[x:x]	Selected CMU clock	External capture	CLSi_TIM_CHx_CTRL.ISL	CLSi_TIM_CHx_CTRL.DSL	Action description
0	1	0	-	0	CLSi_TIM_CHx_CNT++
1	1	0	-	0	No
Rising edge	-	0	0	0	Capture CLSi_TIM_CHx_CNT value in CLSi_TIM_CHx_CNTS
Falling edge	-	0	0	0	CLSi_TIM_CHx_CNT=0

(table continues...)

Table 2204 (continued) Operation depending on CMU clock, CLSi_TIM_CHx_CTRL.ISL, CLSi_TIM_CHx_CTRL.DSL and the input signal value (assume CLSi_TIM_CHx_CTRL.CNTS_SEL= 0)

Input signal F_OUT[x:x]	Selected CMU clock	External capture	CLSi_TIM_CHx_CTRL.ISL	CLSi_TIM_CHx_CTRL.DSL	Action description
Rising edge	-	0	1	0	No
Falling edge	-	0	1	0	Capture CLSi_TIM_CHx_CNT value in CLSi_TIM_CHx_CNTS; CLSi_TIM_CHx_CNT=0
1	1	0	-	1	CLSi_TIM_CHx_CNT++
0	1	0	-	1	No
Falling edge	-	0	0	1	Capture CLSi_TIM_CHx_CNT value in CLSi_TIM_CHx_CNTS
Rising edge	-	0	0	1	CLSi_TIM_CHx_CNT=0
Falling edge	-	0	1	1	No
Rising edge	-	0	1	1	Capture CLSi_TIM_CHx_CNT value in CLSi_TIM_CHx_CNTS; CLSi_TIM_CHx_CNT=0
-	-	Rising edge	-	-	Do CLSi_TIM_CHx_GPR0, CLSi_TIM_CHx_GPR1 capture ; issue TIM_NEWVAL_IRQ
-	0	0	-	-	No

The CLSi_TIM_CHx_CNTS register update is not performed until the measurement is started (first edge defined by CLSi_TIM_CHx_CTRL.DSL is detected). Afterwards the update of the CLSi_TIM_CHx_CNTS register is defined by CLSi_TIM_CHx_CTRL.ISL, CLSi_TIM_CHx_CTRL.DSL combinations in the table above.

TIM pulse integration mode (TPIM)

In **TIM** Pulse Integration Mode each **TIM** channel is able to measure a sum of pulse high or low times on an input signal, depending on the selected signal level bit CLSi_TIM_CHx_CTRL.DSL of register CLSi_TIM_CHx_CTRL register.

If CLSi_TIM_CHx_ECTRL.IMM_START=0 the pulse integration measurement is started with occurrence of the first edge defined by CLSi_TIM_CHx_CTRL.DSL on the input signal. If CLSi_TIM_CHx_ECTRL.IMM_START=1 the measurement starts immediately after activating the channel by CLSi_TIM_CHx_CTRL.TIM_EN=1.

The pulse times are measured by incrementing the **TIM** channel counter CLSi_TIM_CHx_CNT until the counter is stopped with occurrence of an input signal edge to the opposite signal level defined by CLSi_TIM_CHx_CTRL.DSL.

The counter CLSi_TIM_CHx_CNT counts with the CCM[i]_CLK_RES[x:x] clock specified by the CLSi_TIM_CHx_CTRL.CLK_SEL bit field of the CLSi_TIM_CHx_CTRL register.

If the register CLSi_TIM_CHx_CNT produces an overflow (wrap around $FFFFFF_H$ to 0_H due to increment) during the measurement, the bit CLSi_TIM_CHx_IRQ_NOTIFY.CNTOFL is set and interrupt TIM_CNTOFL_IRQ[x:x] is raised depending on corresponding interrupt enable condition.

The CLSi_TIM_CHx_CNT register is reset at the time the channel is activated (enabling via **AEI** write-access) and it accumulates pulses while the channel is staying enabled.

Assume CLSi_TIM_CHx_CTRL.EXT_CAP_EN=0 and CLSi_TIM_CHx_ECTRL.SWAP_CAPTURE=0:

After measurement is started, every falling(CLSi_TIM_CHx_CTRL.DSL=1) or rising(CLSi_TIM_CHx_CTRL.DSL=0) input edge will issue a TIM_NEWVAL_IRQ[x:x] interrupt, and the registers CLSi_TIM_CHx_CNTS, CLSi_TIM_CHx_GPR0 and CLSi_TIM_CHx_GPR1 are updated according to settings of its corresponding input multiplexers, using the bits CLSi_TIM_CHx_CTRL.EGPR0_SEL, CLSi_TIM_CHx_CTRL.EGPR1_SEL, CLSi_TIM_CHx_CTRL.GPR0_SEL, CLSi_TIM_CHx_CTRL.GPR1_SEL and CLSi_TIM_CHx_CTRL.CNTS_SEL.

***Note**: The bits 1 to 7 of the CLSi_TIM_CHx_ECNT may be used to check data consistency of the registers CLSi_TIM_CHx_GPR0 and CLSi_TIM_CHx_GPR1.*

Assume CLSi_TIM_CHx_CTRL.EXT_CAP_EN=0 and CLSi_TIM_CHx_ECTRL.SWAP_CAPTURE=1:

After measurement is started, every falling (CLSi_TIM_CHx_CTRL.DSL=1) or rising (CLSi_TIM_CHx_CTRL.DSL=0) input edge will issue a TIM_NEWVAL_IRQ[x:x] interrupt, and the registers CLSi_TIM_CHx_CNTS, CLSi_TIM_CHx_GPR0 are updated according to settings of its corresponding input multiplexers, using the bits CLSi_TIM_CHx_CTRL.EGPR0_SEL, CLSi_TIM_CHx_CTRL.GPR0_SEL and CLSi_TIM_CHx_CTRL.CNTS_SEL.

Every input edge to the active level defined by CLSi_TIM_CHx_CTRL.DSL (rising CLSi_TIM_CHx_CTRL.DSL=1; falling CLSi_TIM_CHx_CTRL.DSL=0) will capture the data selected by CLSi_TIM_CHx_CTRL.EGPR1_SEL, CLSi_TIM_CHx_CTRL.GPR1_SEL to the registers CLSi_TIM_CHx_GPR1.

External capture TIM pulse integration mode (TPIM)

If external capture is enabled CLSi_TIM_CHx_CTRL.EXT_CAP_EN=1, the pulse integration is done until next external capture event occurs.

On every external capture event the data selected by CLSi_TIM_CHx_CTRL.CNTS_SEL, CLSi_TIM_CHx_CTRL.EGPR0_SEL, CLSi_TIM_CHx_CTRL.GPR0_SEL will be captured to the registers CLSi_TIM_CHx_CNTS, CLSi_TIM_CHx_GPR0.

If CLSi_TIM_CHx_ECTRL.SWAP_CAPTURE=0 every external capture event will capture the data selected by CLSi_TIM_CHx_CTRL.EGPR1_SEL, CLSi_TIM_CHx_CTRL.GPR1_SEL to the registers CLSi_TIM_CHx_GPR1.

If CLSi_TIM_CHx_ECTRL.SWAP_CAPTURE=1 every input edge to the inactive level != CLSi_TIM_CHx_CTRL.DSL will capture the data selected by CLSi_TIM_CHx_CTRL.EGPR1_SEL, CLSi_TIM_CHx_CTRL.GPR1_SEL to the registers CLSi_TIM_CHx_GPR1.

Assume CLSi_TIM_CHx_ECTRL.SWAP_CAPTURE=0; CLSi_TIM_CHx_ECTRL.IMM_START=0:

Operation is done depending on **CMU** clock, CLSi_TIM_CHx_CTRL.DSL bit and the input signal value defined in next table (inc_cnt = false if **TIM** channel is enabled) :

Table 2205 Operation depending on CMU clock, CLSi_TIM_CHx_CTRL.DSL and the input signal value (inc_cnt = false if TIM channel is enabled)

Input signal F_OUT[x:x]	Selected CMU Clock	External capture	CLSi_TIM_CHx_CTRL.ISL	CLSi_TIM_CHx_CTRL.DSL	Action description
Falling edge	-	0	-	0	inc_cnt = true
Rising edge	-	0	-	0	inc_cnt = false
Rising edge	-	0	-	1	inc_cnt = true
Falling edge	-	0	-	1	inc_cnt = false
-	1	0	-	-	If inc_cnt == true then CLSi_TIM_CHx_CNT++;
-	-	Rising edge	-	-	Do capture CLSi_TIM_CHx_GPR0, CLSi_TIM_CHx_GPR1, CLSi_TIM_CHx_CNTS ; issue TIM_NEWVAL_IRQ; CLSi_TIM_CHx_CNT=0
-	0	0	-	-	No

TIM input event mode (TIEM)

In **TIM** Input Event Mode, the **TIM** channel is able to count edges.

It is configurable if rising, falling or both edges should be counted. This can be done with the bit fields CLSi_TIM_CHx_CTRL.DSL and CLSi_TIM_CHx_CTRL.ISL in CLSi_TIM_CHx_CTRL register.

In addition, a TIM_NEWVAL_IRQ[x:x] interrupt is raised when the configured edge was received and this interrupt was enabled.

The counter register CLSi_TIM_CHx_CNT is used to count the number of edges, and the bit fields CLSi_TIM_CHx_CTRL.EGPR0_SEL, CLSi_TIM_CHx_CTRL.EGPR1_SEL, CLSi_TIM_CHx_CTRL.EGPR0_SEL, CLSi_TIM_CHx_CTRL.GPR1_SEL, and CLSi_TIM_CHx_CTRL.CNTS_SEL can be used to configure the desired update values for the registers CLSi_TIM_CHx_GPR0, CLSi_TIM_CHx_GPR1 and CLSi_TIM_CHx_CNTS. These registers are updated whenever the edge counter CLSi_TIM_CHx_CNT is incremented due to the arrival of a desired edge.

If the preceding data was not consumed , the **TIM** channel sets CLSi_TIM_CHx_IRQ_NOTIFY.GPROFL status bit and raises a TIM_GPROFL_IRQ[x:x] if it was enabled in CLSi_TIM_CHx_IRQ_EN register and overwrites the old values in CLSi_TIM_CHx_GPR0 and CLSi_TIM_CHx_GPR1 with the new ones.

If the register CLSi_TIM_CHx_CNT produces an overflow (wrap around $FFFFFF_H$ to 0_H due to increment) during the measurement, the bit CLSi_TIM_CHx_IRQ_NOTIFY.CNTOFL is set and interrupt TIM_CNTOFL_IRQ[x:x] is raised depending on corresponding interrupt enable condition.

If the register CLSi_TIM_CHx_ECNT produces an overflow during the measurement, the bit CLSi_TIM_CHx_IRQ_NOTIFY.ECNTOFL is set and interrupt TIM_ECNTOFL_IRQ[x:x] is raised depending on corresponding interrupt enable condition.

The **TIM** Input Event Mode does not depend on the bit field CLSi_TIM_CHx_CTRL.CLK_SEL of register CLSi_TIM_CHx_CTRL.

External capture TIM input event mode (TIEM)

If external capture is enabled, capturing is done depending on the CLSi_TIM_CHx_CTRL.DSL, CLSi_TIM_CHx_CTRL.ISL bit and the input signal value defined in next table:

Table 2206 Capturing dependent on the CLSi_TIM_CHx_CTRL.DSL, CLSi_TIM_CHx_CTRL.ISL and the input signal value, if External Capture is enabled

Input signal F_OUT[x:x]	External capture	CLSi_TIM_CHx_CTRL.ISL	CLSi_TIM_CHx_CTRL.DSL	Action description
-	Rising edge	1	-	Do capture; issue TIM_NEWVAL_IRQ; CLSi_TIM_CHx_CNT ++
-	0	1	-	No
1	Rising edge	0	1	Do capture; issue TIM_NEWVAL_IRQ; CLSi_TIM_CHx_CNT ++
0	-	0	1	No
0	Rising edge	0	0	Do capture; issue TIM_NEWVAL_IRQ; CLSi_TIM_CHx_CNT ++
1	-	0	0	No

TIM input prescaler mode (TIPM)

In the **TIM** Input Prescaler Mode, the number of edges which should be detected before a TIM_NEWVAL_IRQ[x:x] is raised is programmable. In this mode it must be specified in the CLSi_TIM_CHx_CNTS register after how many edges the interrupt has to be raised.

A value of 0 in CLSi_TIM_CHx_CNTS means that after one edge an interrupt is raised and a value of 1 means that after two edges an interrupt is raised, and so on.

The edges to be counted can be selected by the bit fields CLSi_TIM_CHx_CTRL.DSL and CLSi_TIM_CHx_CTRL.ISL.

With each triggered interrupt, the registers CLSi_TIM_CHx_GPR0 and CLSi_TIM_CHx_GPR1 are updated according to bits CLSi_TIM_CHx_CTRL.EGPR0_SEL, CLSi_TIM_CHx_CTRL.EGPR1_SEL, CLSi_TIM_CHx_CTRL.GPR0_SEL and CLSi_TIM_CHx_CTRL.GPR1_SEL.

If the register CLSi_TIM_CHx_ECNT produces an overflow during the measurement, the bit CLSi_TIM_CHx_IRQ_NOTIFY.ECNTOFL is set and interrupt TIM_ECNTOFL_IRQ[x:x] is raised depending on corresponding interrupt enable condition.

The register CLSi_TIM_CHx_CNT never produces an overflow during this mode, the bit CLSi_TIM_CHx_IRQ_NOTIFY.CNTOFL is never set.

The **TIM** Input Prescaler Mode does not depend on the bit field CLSi_TIM_CHx_CTRL.CLK_SEL of register CLSi_TIM_CHx_CTRL.

External capture TIM input prescaler mode (TIPM)

If external capture is enabled, the external capture events are counted instead of the input signal edges.

Operation is done depending on the external capture signal, CLSi_TIM_CHx_CTRL.DSL, CLSi_TIM_CHx_CTRL.ISL bit and the input signal value defined in next table:

Table 2207 Operation depending on the external capture signal, CLSi_TIM_CHx_CTRL.DSL, CLSi_TIM_CHx_CTRL.ISL and the input signal value

Input signal F_OUT[x:x]	External capture	CLSi_TIM_CHx_CTRL.ISL	DSL	Action description

-	Rising edge	1	-	If CLSi_TIM_CHx_CNT >= CLSi_TIM_CHx_CNTS then
				do capture; issue TIM_NEWVAL_IRQ; CLSi_TIM_CHx_CNT=0
				else
				CLSi_TIM_CHx_CNT++
				endif

-	0	1	-	No

1	Rising edge	0	1	If CLSi_TIM_CHx_CNT >= CLSi_TIM_CHx_CNTS then
				do capture; issue TIM_NEWVAL_IRQ; CLSi_TIM_CHx_CNT=0
				else
				CLSi_TIM_CHx_CNT++
				endif

0	-	0	1	No

(table continues...)

Table 2207 (continued) Operation depending on the external capture signal, CLSi_TIM_CHx_CTRL.DSL, CLSi_TIM_CHx_CTRL.ISL and the input signal value

Input signal F_OUT[x:x]	External capture	CLSi_TIM_CHx_CTRL.ISL	DSL	Action description

0	Rising edge	0	0	If CLSi_TIM_CHx_CNT >= CLSi_TIM_CHx_CNTS then
				do capture; issue TIM_NEWVAL_IRQ; CLSi_TIM_CHx_CNT =0
				else
				CLSi_TIM_CHx_CNT ++
				endif

1	-	0	0	No

TIM bit compression mode (TBCM)

The **TIM** Bit Compression Mode can be used to combine all filtered input signals of a **TIM** sub-module to a parallel m bit data word, where m is the number of channels available in the **TIM** sub-module.

Figure 1055 gives an overview of the **TIM** bit compression mode.

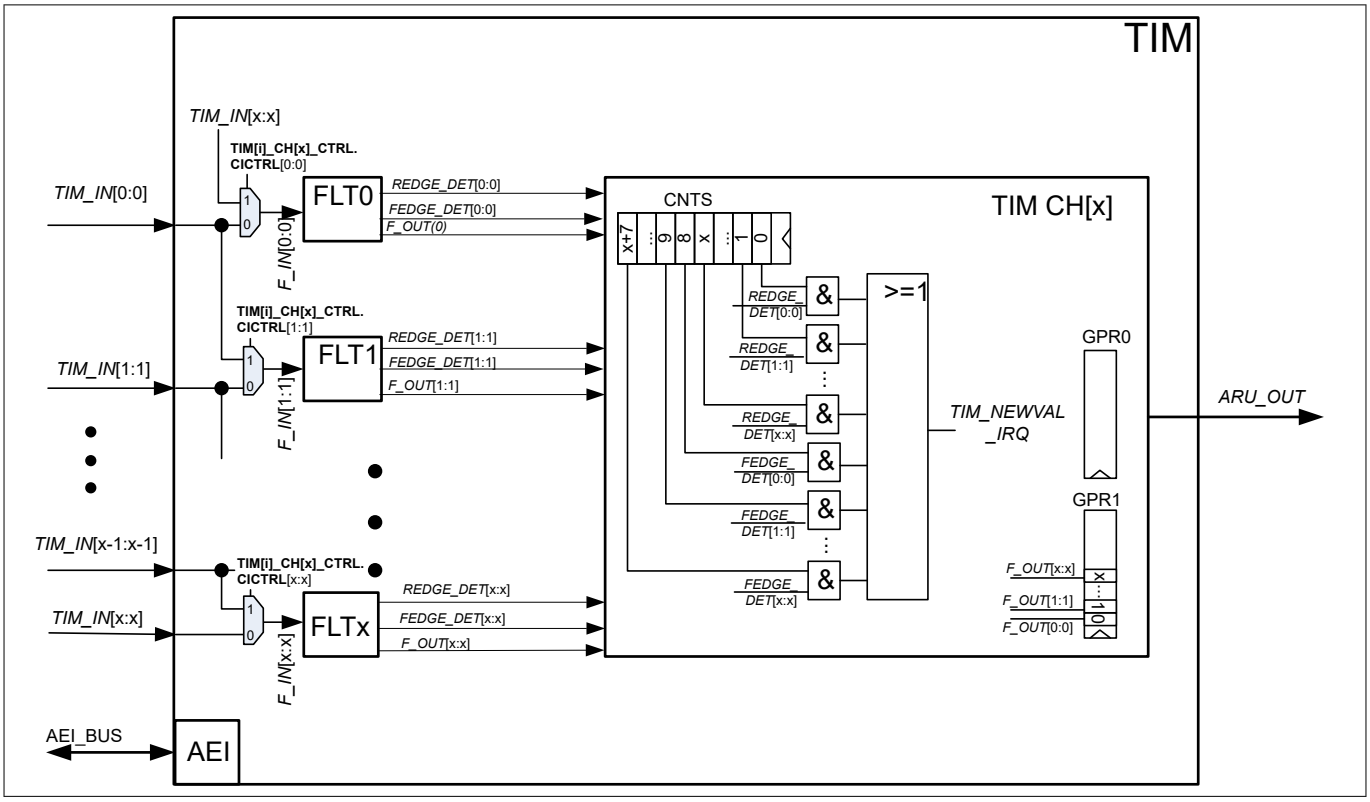


Figure 1055 TIM bit compression mode

The register CLSi_TIM_CHx_CNTS of a channel is used to configure the event that releases the TIM_NEWVAL_IRQ and samples the input signals F_IN[0:0] to F_IN[7:7] of instance [i-1] in ascending order as a parallel data word in CLSi_TIM_CHx_GPR1.

The bits 0 to m-1 of the CLSi_TIM_CHx_CNTS register are used to select the REDGE_DET signals of the **TIM** filters 0 to m-1 as a sampling event, and the bits 8 to (7+m) are used to select the FEDGE_DET signals of the **TIM** filters 0 to m-1, respectively. If multiple events are selected, the events are **OR**-combined (see also Figure 1055).

In **TIM** Bit compression mode, the register CLSi_TIM_CHx_ECNT increments with each TIM_NEWVAL_IRQ, which means that the value of CLSi_TIM_CHx_ECNT may depend on all m input signals. Consequently, the LSB of CLSi_TIM_CHx_ECNT does not reflect the actual level of the input signal CLSi_TIM_INP_VAL.TIM_INx (x=0-7).

If the register CLSi_TIM_CHx_ECNT produces an overflow during the measurement, the bit CLSi_TIM_CHx_IRQ_NOTIFY.ECNTOFL is set and interrupt TIM_ECNTOFL_IRQ[x:x] is raised depending on corresponding interrupt enable condition.

The **TIM** Bit Compression Mode does not depend on the bit field CLSi_TIM_CHx_CTRL.CLK_SEL of register CLSi_TIM_CHx_CTRL.

External capture bit compression mode (TBCM)

If external capture is enabled, capturing is done for CLSi_TIM_CHx_CTRL.ISL=1 as defined in next table. The value 0 for CLSi_TIM_CHx_CTRL.ISL is prohibited.

Table 2208 Capturing dependent on the CLSi_TIM_CHx_CTRL.DSL, CLSi_TIM_CHx_CTRL.ISL and the input signal value, if external capture is enabled

Input signal F_OUT[x:x]	External capture	CLSi_TIM_CHx_CTRL.ISL	CLSi_TIM_CHx_CTRL.DSL	Action description
-	Rising edge	1	-	Do capture; issue TIM_NEWVAL_IRQ;
-	0	1	-	No
-	-	0 - prohibited	-	-

TIM gated periodic sampling mode (TGPS)

In the **TIM** Gated Periodic Sampling Mode, the number of **CMU** clock-cycles which should elapse before capturing and raising TIM_NEWVAL_IRQ[x:x] is programmable. In this mode it must be specified in the CLSi_TIM_CHx_CNTS register after how many **CMU** clock-cycles the interrupt has to be raised.

A value of 0 in CLSi_TIM_CHx_CNTS means that after one CLSi_TIM_CHx_CTRL.CLK_SEL edge a trigger/interrupt is raised, and a value of 1 means that after two edges a trigger/interrupt is raised, and so on.

In the CLSi_TIM_CHx_CNT register the elapsed cycles were incremented and compared against CLSi_TIM_CHx_CNTS. If CLSi_TIM_CHx_CNT is greater or equal to CLSi_TIM_CHx_CNTS a trigger will be raised. This allows by writing a value to CLSi_TIM_CHx_CNTS that the actual period time can be changed on the fly.

Operation is done depending on **CMU** clock, CLSi_TIM_CHx_CTRL.DSL, CLSi_TIM_CHx_CTRL.ISL bit and the input signal value defined in next table:

The register CLSi_TIM_CHx_CNT never produces an overflow during this mode, the bit CLSi_TIM_CHx_IRQ_NOTIFY.CNTOFL is never set.

Table 2209 Operation depending on CMU clock, CLSi_TIM_CHx_CTRL.DSL, CLSi_TIM_CHx_CTRL.ISL and the input signal value

Input signal F_OUT *[x:x]*	Selected CMU clock	External capture	CLSi_TIM_CHx_CTRL.ISL	DSL	Action description

-	1	0	1	-	If CLSi_TIM_CHx_CNT >= CLSi_TIM_CHx_CNTS then
					do capture; issue TIM_NEWVAL_IRQ; CLSi_TIM_CHx_CNT=0
					else
					CLSi_TIM_CHx_CNT++
					endif

0	0	0	0	1	No

1	1	0	0	1	If CLSi_TIM_CHx_CNT >= CLSi_TIM_CHx_CNTS then
					do capture; issue TIM_NEWVAL_IRQ; CLSi_TIM_CHx_CNT=0
					else
					CLSi_TIM_CHx_CNT++
					endif

0	0	-	0	1	No

(table continues...)

Table 2209 (continued) Operation depending on CMU clock, CLSi_TIM_CHx_CTRL.DSL, CLSi_TIM_CHx_CTRL.ISL and the input signal value

Input signal F_OUT *[x:x]*	Selected CMU clock	External capture	CLSi_TIM_CHx_CTRL.ISL	DSL	Action description

0	1	0	0	0	If CLSi_TIM_CHx_CNT >= CLSi_TIM_CHx_CNTS then
					do capture; issue TIM_NEWVAL_IRQ; CLSi_TIM_CHx_CNT=0
					else
					CLSi_TIM_CHx_CNT++
					endif

1	0	0	0	0	No
-	0	0	-	-	No

In this mode the CLSi_TIM_CHx_GPR1 operates as a shadow register for CLSi_TIM_CHx_CNTS. This would allow that the period for the next sampling period could be specified. The update of CLSi_TIM_CHx_CNTS will only take place once on a trigger if the CLSi_TIM_CHx_GPR1 was written by the CPU. This means that the captured value from the previous trigger can be read by the CPU from CLSi_TIM_CHx_GPR1 and afterwards the new sampling period for the next sampling period (the one after the actual sampling period) could be written.

With each triggered interrupt, the registers CLSi_TIM_CHx_GPR0 and CLSi_TIM_CHx_GPR1 are updated according to bits CLSi_TIM_CHx_CTRL.GPR0_SEL, CLSi_TIM_CHx_CTRL.GPR1_SEL, CLSi_TIM_CHx_CTRL.EGPR0_SEL and CLSi_TIM_CHx_CTRL.EGPR1_SEL.

When selecting CLSi_TIM_CHx_ECNT as a source for the capture registers, CLSi_TIM_CHx_GPR0, CLSi_TIM_CHx_GPR1 will show the edge count and the input signal value at point of capture. Selecting CLSi_TIM_CHx_CTRL.GPR0_SEL = '11' and CLSi_TIM_CHx_CTRL.EGPR0_SEL = '0' for **TIM** channel 0 all 8 **TIM** input signals will be captured to CLSi_TIM_CHx_GPR0[7:0].

In the TGPS Mode, the bit field CLSi_TIM_CHx_CTRL.CLK_SEL of register CLSi_TIM_CHx_CTRL will define the selected **CMU** clock which will be used.

The behavior of the CLSi_TIM_CHx_ECNT counter is configurable by CLSi_TIM_CHx_CTRL.ECNT_RESET. If set to 1 on each interrupt (period expired), the CLSi_TIM_CHx_ECNT will be reset. Otherwise, it operates in wrap around mode.

If the register CLSi_TIM_CHx_ECNT produces an overflow during the measurement, the bit CLSi_TIM_CHx_IRQ_NOTIFY.ECNTOFL is set and interrupt TIM_ECNTOFL_IRQ[x:x] is raised depending on corresponding interrupt enable condition.

External capture TIM gated periodic sampling mode (TGPS)

If external capture is enabled, the external capture events will capture actual values in the register CLSi_TIM_CHx_GPR0 and CLSi_TIM_CHx_GPR1, reset the counter CLSi_TIM_CHx_CNT and issue a TIM_NEWVAL_IRQ.

Operation is done depending on the **CMU** clock, external capture signal, CLSi_TIM_CHx_CTRL.DSL, CLSi_TIM_CHx_CTRL.ISL bit and the input signal value defined in next table:

Table 2210 Operation depending on the CMU clock, external capture signal, CLSi_TIM_CHx_CTRL.DSL, CLSi_TIM_CHx_CTRL.ISL and the input signal value

Input signal F_OUT[x:x]	Selected CMU clock	External capture	CLSi_TIM_CHx_CTRL.ISL	CLSi_TIM_CHx_CTRL.DSL	Action description

-	1	0	1	-	If CLSi_TIM_CHx_CNT >= CLSi_TIM_CHx_CNTS then
					do capture; issue TIM_NEWVAL_IRQ; CLSi_TIM_CHx_CNT=0
					else
					CLSi_TIM_CHx_CNT++
					endif

0	0	0	0	1	No

1	1	0	0	1	If CLSi_TIM_CHx_CNT >= CLSi_TIM_CHx_CNTS then
					do capture; issue TIM_NEWVAL_IRQ; CLSi_TIM_CHx_CNT=0
					else
					CLSi_TIM_CHx_CNT++
					endif

0	0	-	0	1	No

(table continues...)

Table 2210 (continued) Operation depending on the CMU clock, external capture signal, CLSi_TIM_CHx_CTRL.DSL, CLSi_TIM_CHx_CTRL.ISL and the input signal value

Input signal F_OUT[x:x]	Selected CMU clock	External capture	CLSi_TIM_CHx_CTRL.ISL	CLSi_TIM_CHx_CTRL.DSL	Action description

0	1	0	0	0	If CLSi_TIM_CHx_CNT >= CLSi_TIM_CHx_CNTS then
					do capture; issue TIM_NEWVAL_IRQ; CLSi_TIM_CHx_CNT=0
					else
					CLSi_TIM_CHx_CNT++
					endif

1	0	0	0	0	No
-	0	0	-	-	No
-	-	Rising edge	-	-	Do capture; issue TIM_NEWVAL_IRQ; CLSi_TIM_CHx_CNT =0

TIM serial shift mode (TSSM)

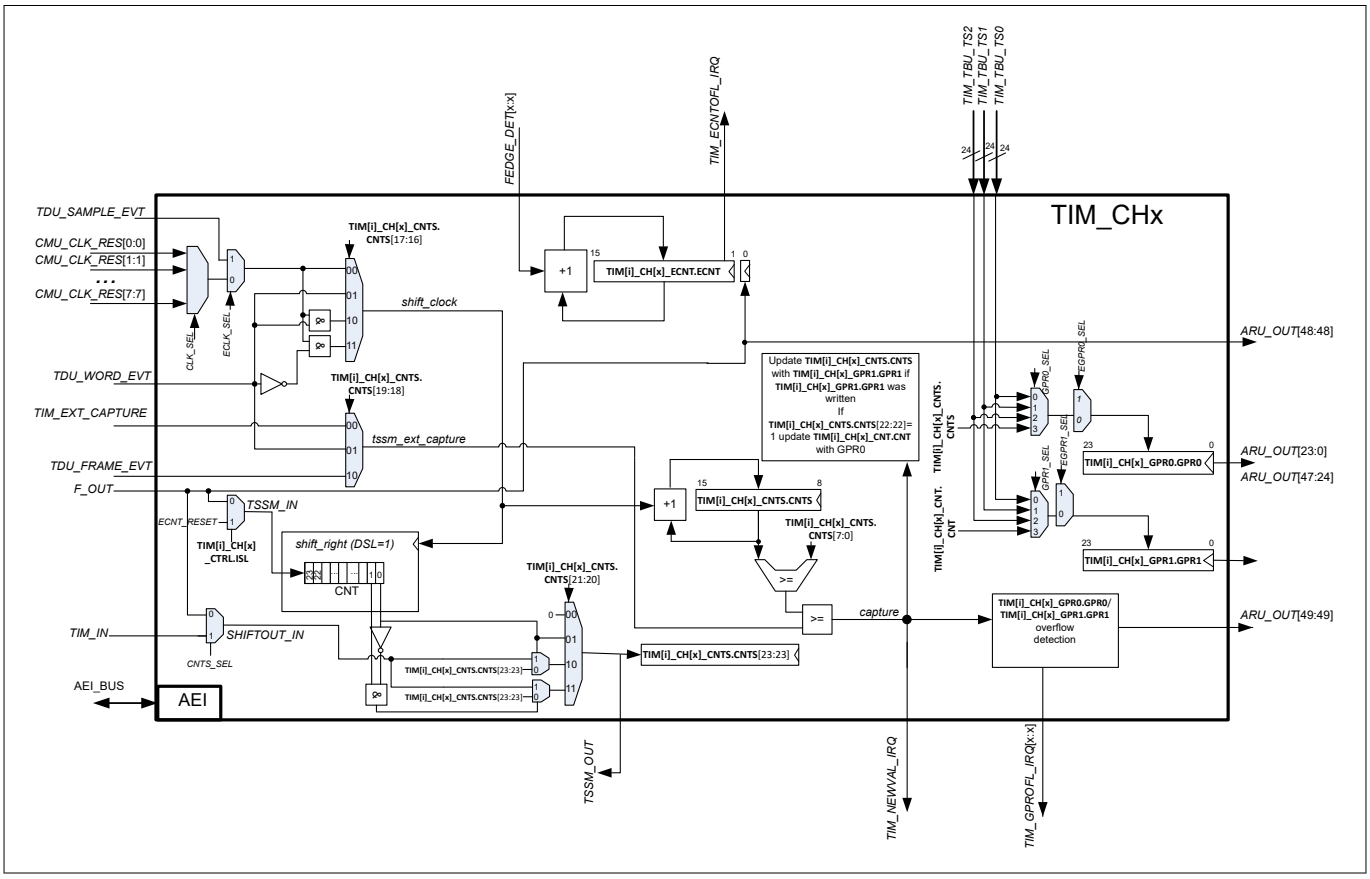


Figure 1056 TIM serial shift mode

In the **TIM** Serial Shift Mode on each shift clock event, the actual value of the input signal TSSM_IN[x:x] will be registered in dependence of CLSi_TIM_CHx_CTRL.DSL in the register CLSi_TIM_CHx_CNT.

If CLSi_TIM_CHx_CTRL.ISL=0 is set F_OUT[x:x] will be used as shift in value TSSM_IN[x:x] , with CLSi_TIM_CHx_CTRL.ISL=1 the bit field CLSi_TIM_CHx_CTRL.ECNT_RESET defines the value for TSSM_IN[x:x].

With CLSi_TIM_CHx_CTRL.DSL=0 TSSM_IN[x:x] will be stored in CLSi_TIM_CHx_CNT[0:0] and CLSi_TIM_CHx_CNT[22:0] will be shifted left. With CLSi_TIM_CHx_CTRL.DSL=1 TSSM_IN[x:x] will be stored in CLSi_TIM_CHx_CNT[23:23] and CLSi_TIM_CHx_CNT[23:1] will be shifted right.

Operation is done depending on the shift clock, external capture signal, CLSi_TIM_CHx_CTRL.DSL, CLSi_TIM_CHx_CTRL.ISL bit and the input signal value defined in next table:

The register CLSi_TIM_CHx_CNT never produces an overflow during this mode, the bit CLSi_TIM_CHx_IRQ_NOTIFY.CNTOFL is never set.

Table 2211 Operation depending on the shift clock, external capture signal, CLSi_TIM_CHx_CTRL.DSL, CLSi_TIM_CHx_CTRL.ISL and the input signal value

Input signal TSSM_IN[x:x]	Shift clock	tssm_ext_capture	CLSi_TIM_CHx_CTRL.ISL	DSL	Action description
-	0	0	-	-	No

(table continues...)

Table 2211 (continued) Operation depending on the shift clock, external capture signal, CLSi_TIM_CHx_CTRL.DSL, CLSi_TIM_CHx_CTRL.ISL and the input signal value

Input signal TSSM_IN[x:x]	Shift clock	tssm_ext_capture	CLSi_TIM_CHx_CTRL.ISL	DSL	Action description

-	-	1	-	-	If CLSi_TIM_CHx_CTRL.EXT_CAP_EN=1 then
					see function table in next chapter
					else
					no
					endif

value	1	0	0	0	CLSi_TIM_CHx_CNT[23:1]= CLSi_TIM_CHx_CNT[22:0];
					CLSi_TIM_CHx_CNT[0:0]= value
					if CLSi_TIM_CHx_CNTS[15:8] >= CLSi_TIM_CHx_CNTS[7:0] then
					do capture; issue TIM_NEWVAL_IRQ;
					CLSi_TIM_CHx_CNTS[15:8]=0
					else
					CLSi_TIM_CHx_CNTS[15:8]++
					endif

(table continues...)

Table 2211 (continued) Operation depending on the shift clock, external capture signal, CLSi_TIM_CHx_CTRL.DSL, CLSi_TIM_CHx_CTRL.ISL and the input signal value

Input signal TSSM_IN[x:x]	Shift clock	tssm_ext_capture	CLSi_TIM_CHx_CTRL.ISL	DSL	Action description

value	1	0	0	1	CLSi_TIM_CHx_CNT[22:0]= CLSi_TIM_CHx_CNT[23:1]; CLSi_TIM_CHx_CNT[23:23]= value
					if CLSi_TIM_CHx_CNTS[15:8] >= CLSi_TIM_CHx_CNTS[7:0] then
					do capture; issue TIM_NEWVAL_IRQ; CLSi_TIM_CHx_CNTS[15:8]=0
					else
					CLSi_TIM_CHx_CNTS[15:8]++
					endif

(table continues...)

Table 2211 (continued) Operation depending on the shift clock, external capture signal, CLSi_TIM_CHx_CTRL.DSL, CLSi_TIM_CHx_CTRL.ISL and the input signal value

Input signal TSSM_IN[x:x]	Shift clock	tssm_ext_capture	CLSi_TIM_CHx_CTRL.ISL	DSL	Action description

value	1	0	1	0	CLSi_TIM_CHx_CNT[23:1]= CLSi_TIM_CHx_CNT[22:0]; CLSi_TIM_CHx_CNT[0:0]= value
					if CLSi_TIM_CHx_CNTS[15:8] >= CLSi_TIM_CHx_CNTS[7:0] then
					do capture; issue TIM_NEWVAL_IRQ; CLSi_TIM_CHx_CNTS[15:8]=0
					CLSi_TIM_CHx_CNT[23:0]=CLSi_TIM_CHx_CTRL.ECNT_RESET
					else
					CLSi_TIM_CHx_CNTS[15:8]++
					endif

(table continues...)

Table 2211 (continued) Operation depending on the shift clock, external capture signal, CLSi_TIM_CHx_CTRL.DSL, CLSi_TIM_CHx_CTRL.ISL and the input signal value

Input signal TSSM_IN[x:x]	Shift clock	tssm_ext_capture	CLSi_TIM_CHx_CTRL.ISL	DSL	Action description

value	1	0	1	1	CLSi_TIM_CHx_CNT[22:0]= CLSi_TIM_CHx_CNT[23:1]; CLSi_TIM_CHx_CNT[23:23]= value
					if CLSi_TIM_CHx_CNTS[15:8] >= CLSi_TIM_CHx_CNTS[7:0] then
					do capture; issue TIM_NEWVAL_IRQ; CLSi_TIM_CHx_CNTS[15:8]=0
					CLSi_TIM_CHx_CNT[23:0]=CLSi_TIM_CHx_CTRL.ECNT_RESET
					else
					CLSi_TIM_CHx_CNTS[15:8]++
					endif

The register CLSi_TIM_CHx_CNTS[7:0] defines the number of bits which will be stored inside CLSi_TIM_CHx_CNT.

Each shift clock will increment the register CLSi_TIM_CHx_CNTS[15:8]. If the condition CLSi_TIM_CHx_CNTS[15:8] >= CLSi_TIM_CHx_CNTS[7:0] is met a capture event is raised and TIM_NEWVAL_IRQ[x:x] is asserted.

With each capture event the registers CLSi_TIM_CHx_GPR0 and CLSi_TIM_CHx_GPR1 are updated according to bits CLSi_TIM_CHx_CTRL.GPR0_SEL, CLSi_TIM_CHx_CTRL.GPR1_SEL, CLSi_TIM_CHx_CTRL.EGPR0_SEL and CLSi_TIM_CHx_CTRL.EGPR1_SEL.

If the bit field CLSi_TIM_CHx_CTRL.ISL is set to 1 the register bits CLSi_TIM_CHx_CNT are set to the value defined by CLSi_TIM_CHx_CTRL.ECNT_RESET in case of a capture event.

In this mode the CLSi_TIM_CHx_GPR1 operates as a shadow register for CLSi_TIM_CHx_CNTS. This allows that the number of bits to sample can be specified. The update of CLSi_TIM_CHx_CNTS will only take place once on a trigger if the CLSi_TIM_CHx_GPR1 was written by the CPU. This means that the captured value from the previous trigger can be read by the CPU from CLSi_TIM_CHx_GPR1 and afterwards the new number of bits to sample for the next sampling period (the one after the actual sampling period) could be written.

TSSM using as serial shift in/out: CLSi_TIM_CHx_CTRL.ISL=0

- CLSi_TIM_CHx_CNTS.CNTS[22:22]=0: Shifted in data stored in CLSi_TIM_CHx_CNT.CNT can be captured on a capture event in CLSi_TIM_CHx_GPR1.GPR1, all other CLSi_TIM_CHx_CTRL.GPR1_SEL/ CLSi_TIM_CHx_CTRL.EGPR1_SEL settings can be used too.

- CLSi_TIM_CHx_CNTS.CNTS[22:22]=1: The CLSi_TIM_CHx_GPR0.GPR0 can never be used to capture data.
- The CLSi_TIM_CHx_GPR0.GPR0 can be used as a shadow register to provide the next data which will be shifted out. An update of CLSi_TIM_CHx_CNT.CNT with the content of CLSi_TIM_CHx_GPR0.GPR0 will happen on every capture event.

TSSM using as serial shift out: CLSi_TIM_CHx_CTRL.ISL=1

- TIM_IN data will not be stored in CLSi_TIM_CHx_CNT.CNT; CLSi_TIM_CHx_CTRL.ECNT_RESET will be taken instead.
- CLSi_TIM_CHx_CNTS.CNTS[22:22]=1:CLSi_TIM_CHx_GPR0.GPR0 can never be used to capture data.
- CLSi_TIM_CHx_GPR0.GPR0 can be used as a shadow register to provide the next data which will be shifted out. An update of CLSi_TIM_CHx_CNT.CNT with the content of CLSi_TIM_CHx_GPR0.GPR0 will happen on every capture event.
- CLSi_TIM_CHx_CNTS.CNTS[22:22]=0: Shifted-in data stored in CLSi_TIM_CHx_CNT.CNT can be captured on a capture event in CLSi_TIM_CHx_GPR1.GPR1, all other CLSi_TIM_CHx_CTRL.GPR1_SEL/ CLSi_TIM_CHx_CTRL.EGPR1_SEL settings can be used too.
- The shift clock which will be in use is selectable by CLSi_TIM_CHx_CNTS[17:16]
- 00_B = USED_CLK_RES is in use. It can be set to any CCM[i]_CLK_RES source or to the local TDU sample clock TDU_SAMPLE_EVT.
- 01_B = the TDU_WORD_EVT signal will be used as shift clock source.
- 10_B = USED_CLK_RES is gated with TDU_WORD_EVT. If TDU_WORD_EVT=0 then shift clock will be 0.
- 11_B = USED_CLK_RES is gated with inverted TDU_WORD_EVT. If TDU_WORD_EVT=1 then shift clock will be 0.

Signal generation with TIM serial shift mode

If CLSi_TIM_CHx_CNTS[22:22] is 1 the CLSi_TIM_CHx_GPR0 operates as a shadow register for CLSi_TIM_CHx_CNT. This allows that the bits for shifting out can be specified. The update of CLSi_TIM_CHx_CNT will only take place once on a trigger if the CLSi_TIM_CHx_GPR0 was written by the CPU. This means that the captured value from the previous trigger can be read by the CPU from CLSi_TIM_CHx_GPR0 and afterwards the new bits to shift out could be written.

In addition, the **TIM** Serial Shift Mode is able to generate a signal TSSM_OUT which can be used internally to the **TIM** channel.

On each system clock the value for TSSM_OUT is generated as defined next. The actual value can be read by the register bit CLSi_TIM_CHx_CNTS[23:23].

- Following functionality for TSSM_OUT[x:x] is selectable by CLSi_TIM_CHx_CNTS[21:20]
- 00_B: Constant output; TSSM_OUT[x:x] = 0.
- 01_B: Shift output; If CLSi_TIM_CHx_CTRL.DSL=0 (shift left) then TSSM_OUT[x:x] = CLSi_TIM_CHx_CNT[23:23] else (shift right) TSSM_OUT[x:x] = CLSi_TIM_CHx_CNT[0:0].
- 10_B: Latched output; If CLSi_TIM_CHx_CTRL.DSL=0 and CLSi_TIM_CHx_CNT[23:23]=1 then TSSM_OUT[x:x] =SHIFTOUT_IN[x:x] elsif CLSi_TIM_CHx_CTRL.DSL=1 and CLSi_TIM_CHx_CNT[0:0]=1 then TSSM_OUT[x:x] = SHIFTOUT_IN[x:x].
- 11_B: Registered output; If CLSi_TIM_CHx_CTRL.DSL=0 and CLSi_TIM_CHx_CNT[23:22]=01_B then TSSM_OUT[x:x] = SHIFTOUT_IN[x:x] elsif CLSi_TIM_CHx_CTRL.DSL=1 and CLSi_TIM_CHx_CNT[1:0]=10_B then TSSM_OUT[x:x] = SHIFTOUT_IN[x:x].

In case of registered or latched output mode the signal SHIFTOUT_IN[x:x] is selectable by CLSi_TIM_CHx_CTRL.CNTS_SEL.

If CLSi_TIM_CHx_CTRL.CNTS_SEL=0 is set F_OUT[x:x] will be used for SHIFTOUT_IN[x:x], with CLSi_TIM_CHx_CTRL.CNTS_SEL=1 the signal TIM_IN[x:x] is in µs for SHIFTOUT_IN[x:x].

External capture TIM serial shift mode (TSSM)

If external capture is enabled (CLSi_TIM_CHx_CTRL.EXT_CAP_EN=1), the external capture events will capture the actual values in CLSi_TIM_CHx_GPR0 and CLSi_TIM_CHx_GPR1, reset the counter CLSi_TIM_CHx_CNT depending on CLSi_TIM_CHx_CTRL.ISL and issue a TIM_NEWVAL_IRQ. Functionality from previous table will be applied.

- The source which will be used as external capture event for TSSM mode is selectable by CLSi_TIM_CHx_CNTS[19:18]
- 00_B = source selection by CLSi_TIM_CHx_ECTRL.EXT_CAP_SRC is in use.
- 01_B = TDU_WORD_EVT signal will be used as source.
- 10_B = TDU_FRAME_EVT signal will be used as source.
- 11_B = reserved

Operation is done depending on the shift clock, external capture signal, CLSi_TIM_CHx_CTRL.DSL, CLSi_TIM_CHx_CTRL.ISL bit and the input signal value defined in next table:

Table 2212 Operation depending on the shift clock, external capture signal, CLSi_TIM_CHx_CTRL.DSL, CLSi_TIM_CHx_CTRL.ISL and the input signal value

Input signal F_OUT[x:x]	Shift clock	tssm_ext_capture	CLSi_TIM_CHx_CTRL.ISL	CLSi_TIM_CHx_CTRL.DSL	Action description
-	0	1	1	-	Do capture; issue TIM_NEWVAL_IRQ; CLSi_TIM_CHx_CNTS[15:8]=0 CLSi_TIM_CHx_CNT[23:0]=CLSi_TIM_CHx_CTRL.ECNT_RESET
-	0	1	0	-	Do capture; issue TIM_NEWVAL_IRQ; CLSi_TIM_CHx_CNTS[15:8]=0
value	1	1	1	0	CLSi_TIM_CHx_CNT[23:1]= CLSi_TIM_CHx_CNT[22:0]; CLSi_TIM_CHx_CNT[0:0]= value do capture; issue TIM_NEWVAL_IRQ; CLSi_TIM_CHx_CNTS[15:8]=0 CLSi_TIM_CHx_CNT[23:0]=CLSi_TIM_CHx_CTRL.ECNT_RESET

(table continues...)

Table 2212 (continued) Operation depending on the shift clock, external capture signal, CLSi_TIM_CHx_CTRL.DSL, CLSi_TIM_CHx_CTRL.ISL and the input signal value

Input signal F_OUT[x:x]	Shift clock	tssm_ext_capture	CLSi_TIM_CHx_CTRL.ISL	CLSi_TIM_CHx_CTRL.DSL	Action description
value	1	1	1	1	CLSi_TIM_CHx_CNT[22:0]= CLSi_TIM_CHx_CNT[23:1]; CLSi_TIM_CHx_CNT[23:23]= value do capture; issue TIM_NEWVAL_IRQ; CLSi_TIM_CHx_CNTS[15:8]=0 CLSi_TIM_CHx_CNT[23:0]=CLSi_TIM_CHx_CTRL.ECNT_RESET
value	1	1	0	0	CLSi_TIM_CHx_CNT[23:1]= CLSi_TIM_CHx_CNT[22:0]; CLSi_TIM_CHx_CNT[0:0]= value do capture; issue TIM_NEWVAL_IRQ; CLSi_TIM_CHx_CNTS[15:8]=0
value	1	1	0	1	CLSi_TIM_CHx_CNT[22:0]= CLSi_TIM_CHx_CNT[23:1]; CLSi_TIM_CHx_CNT[23:23]= value do capture; issue TIM_NEWVAL_IRQ; CLSi_TIM_CHx_CNTS[15:8]=0

44.3.1.6.5 TIM software reset of channels

Each channel x is assigned a unique bit x in the configuration register CLSi_TIM_RST for software reset of the **TIM[i]** channel block.

Writing via the configuration interface the value 1_B to the bit field CLSi_TIM_RST.RST_CHx (x=0-7), will immediately reset the content of the following registers to the initial hardware reset state.

- CLSi_TIM_CHx_CTRL

- CLSi_TIM_CHx_FLT_RE
- CLSi_TIM_CHx_FLT_FE
- CLSi_TIM_CHx_GPR0
- CLSi_TIM_CHx_GPR1
- CLSi_TIM_CHx_CNT
- CLSi_TIM_CHx_CNTS
- CLSi_TIM_CHx_IRQ_NOTIFY
- CLSi_TIM_CHx_IRQ_EN
- CLSi_TIM_CHx_IRQ_MODE
- CLSi_TIM_CHx_EIRQ_EN
- CLSi_TIM_CHx_TDUV
- CLSi_TIM_CHx_TDUC
- CLSi_TIM_CHx_ECNT
- CLSi_TIM_CHx_ECTRL

Atomic reset of multiple channels is possible by writing a 1 on each associated bit. E.g.: Writing $0F_H$ to register CLSi_TIM_RST (with i=0) will reset the TIM0 channel x (x∈{0, 1, 2, 3}).

44.3.1.6.6 eGTM TIM Configuration Registers Description

Sub-chapter TIM Configuration Registers Description

Table 2213

Register Shortname	Description
CLSi_TIM_CHx_CTRL	TIM[i] channel [x] control register (x=0-7;i=0)
CLSi_TIM_CHx_CTRL	TIM[i] channel [x] control register (x=0-7;i=1-2)
CLSi_TIM_CHx_FLT_RE	TIM[i] channel [x] filter parameter 0 register (x=0-7;i=0-2)
CLSi_TIM_CHx_FLT_FE	TIM[i] channel [x] filter parameter 1 register (x=0-7;i=0-2)
CLSi_TIM_CHx_GPR0	TIM[i] channel [x] general-purpose 0 register (x=0-7;i=0-2)
CLSi_TIM_CHx_GPR1	TIM[i] channel [x] general-purpose 1 register (x=0-7;i=0-2)
CLSi_TIM_CHx_CNT	TIM[i] channel [x] SMU counter register (x=0-7;i=0-2)
CLSi_TIM_CHx_CNTS	TIM[i] channel [x] SMU shadow counter register (x=0-7;i=0-2)
CLSi_TIM_CHx_IRQ_NOTIFY	TIM[i] channel [x] interrupt notification register (x=0-7;i=0-2)
CLSi_TIM_CHx_IRQ_EN	TIM[i] channel [x] interrupt enable register (x=0-7;i=0-2)
CLSi_TIM_CHx_IRQ_FORCINT	TIM[i] channel [x] force interrupt register (x=0-7;i=0-2)
CLSi_TIM_CHx_IRQ_MODE	TIM[i] channel [x] interrupt mode configuration register (x=0-7;i=0-2)

(table continues...)

Table 2213 (continued)

Register Shortname	Description
CLSi_TIM_RST	TIM[i] global software reset register (i=0-2)
CLSi_TIM_IN_SRC	TIM[i] AUX IN source selection register (i=0-2)
CLSi_TIM_CHx_EIRQ_EN	TIM[i] channel [x] error interrupt enable register (x=0-7;i=0-2)
CLSi_TIM_CHx_TDUV	TIM[i]_CH[x]_TDUV (x=0-7;i=0-2)
CLSi_TIM_CHx_TDUC	TIM[i]_CH[x]_TDUC (x=0-7;i=0-2)
CLSi_TIM_CHx_ECNT	TIM[i] channel [x] SMU edge counter register (x=0-7;i=0-2)
CLSi_TIM_CHx_ECTRL	TIM[i] channel [x] extended control register (x=0-7;i=0-2)
CLSi_TIM_INP_VAL	TIM[i] input value observation register (i=0-2)

44.3.1.6.7 TIM signal description

Table 2214 TIM Reset

Name	Description
TIM[j]_RESET_CH[x]	**TIM[j]** channel [x] reset active signal

Table 2215 TIM Interrupt Signals

Name	Description
TIM_GPROFL_IRQ	CLSi_TIM_CHx_GPR0 and CLSi_TIM_CHx_GPR1 data overflow, old data was not read out before new data has arrived at input pin of channel x
TIM_TODET_IRQ	Time out reached for input signal of channel x
TIM_NEWVAL_IRQ	New measurement value detected by SMU of channel x
TIM_ECNTOFL_IRQ	ECNT counter overflow of channel x
TIM_CNTOFL_IRQ	SMU CLSi_TIM_CHx_CNT counter overflow of channel x
TIM_GLITCHDET_IRQ	A glitch was detected by the **TIM** filter of channel x

Table 2216 TIM Signals

Name	Description
LUT_IN0	INPUTSRC unit: Input signal 0 for lookup-table processing
LUT_IN1	INPUTSRC unit: Input signal 1 for lookup-table processing
LUT_IN2	INPUTSRC unit: Input signal 2 for lookup-table processing
F_IN	Output signal of the INPUTSRC units of each channels
FLT_CLK_RES	FLT unit: Clock resolution in use

(table continues...)

Table 2216 (continued) TIM Signals

Name	Description
F_IN_SYNC	FLT unit: FLT input signal F_IN synchronized based on the chosen clock resolution
FEDGE_DET	FLT unit: Falling edge detection signal
REDGE_DET	FLT unit: Rising edge detection signal
INC	Input signal to TDU_SLICE; on 1 the counter will increase
LOAD	Input signal to TDU_SLICE; on 1 the counter will be loaded with LOAD_VAL
LOAD_VAL	Input signal to TDU_SLICE; defines the value which will be loaded into the counter in case LOAD=1
EQ_EVT	Output signal of TDU_SLICE; 1 in case counter and compare values are identical
GT_EVT	Output signal of TDU_SLICE; 1 in case counter value is greater than compare value
TDU_TIMEOUT_EVT	Output signal of TDU; signal behavior depends on TDU configuration
TDU_WORD_EVT	Output signal of TDU; signal behavior depends on TDU configuration
TDU_FRAME_EVT	Output signal of TDU; signal behavior depends on TDU configuration
TDU_SAMPLE_EVT	Output signal of TDU; signal behavior depends on TDU configuration
TDU_TOCTRL_0_EVENT	TDU unit: single-cycle pulse indicating that the timeout functionality was disabled
TDU_START_000_EVENT	TDU unit: single-cycle pulse indicating that the timeout functionality was enabled
USED_CLK_RES	TIM_CH unit: Internal clock resolution signal selected depending on the configuration in TSSM mode;
TSSM_IN	TIM_CH unit: Internal input signal selected depending on the configuration in TSSM mode;
TSSM_OUT	TIM_CH unit: Output signal in use by TSSM mode
SHIFTOUT_IN	TIM_CH unit: Internal input signal selected depending on configuration in TSSM mode;
FOUT_NEXT	TIM_CH unit: Output signal the next channel instance
FOUT_PREV	TIM_CH unit: Input signal from the previous channel instance
NEWVAL	TIM_CH unit: Signal which indicates that a capture event of the defined channel mode has occurred
CNTOFL	TIM_CH unit: Signal which indicates that an overflow of the channel counter has occurred

(table continues...)

Table 2216 (continued) TIM Signals

Name	Description
ECNTOFL	TIM_CH unit: Signal which indicates that an overflow of the edge counter has occurred
GPROFL	TIM_CH unit: Signal which indicates that captured data was overwritten by new capture event
EXT_CAP_SRC_IN	TIM_CH unit: Input port vector 9 bits , providing all local capture source events from following channel
EXT_CAP_SRC_OUT	TIM_CH unit: Output port vector 9 bits, providing all local capture source events from this channel for the previous channel
NEWVAL_EVT	Interrupt event - triggered when a new measurement value is detected by SMU of channel x
CNTOFL_EVT	Interrupt event - triggered when SMU CLSi_TIM_CHx_CNT counter of channel x overflows
ECNTOFL_EVT	Interrupt event - triggered when ECNT counter of channel x overflows
TODET_EVT	Interrupt event - triggered when input signal of channel x times out
GLITCHDET_EVT	Interrupt event - triggered when a glitch is detected by the **TIM** filter in channel x
GPROFL_EVT	Interrupt event - triggered when CLSi_TIM_CHx_GPR0 and CLSi_TIM_CHx_GPR1 data overflow, old data was not read out before new data has arrived at input pin of channel x

44.3.1.6.8 TIM port description

Table 2217 TIM Ports

Name	Description
AUX_IN	Auxiliary input for **TIM** channels (wired to TIM[i]_AUX_IN)
TIM_EXT_CAPTURE	External capture signal of each channel
TIMCH0_EXT_CAP_SRC_OUT	External capture sources signal of channel 0
TIM_TBU_TS0	Time base channel 0 signal
TIM_TBU_TS1	Time base channel 1 signal
TIM_TBU_TS2	Time base channel 2 signal
TIM_IN	External input for **TIM** channels
F_OUT	FLT unit: Output signal of each filter channel

44.3.1.7 Timer output module (TOM)

44.3.1.7.1 Overview

The Timer Output Module (**TOM**) offers up to 16 independent channels (index x) to generate simple PWM signals at each output port TOM_OUT[x:x].

Additionally, at **TOM** output TOM_OUT[15:15] a pulse count modulated signal can be generated.

The architecture of the **TOM** sub-module is depicted in Figure 1057.

Indices and their range as used inside this chapter are:

- NTOM= number of **TOM** instances in the device
- i:={0, 1,..., NTOM-1} instance index of cluster / **TOM** module
- x:={0, 1,..., 15} index of channels
- g:={0, 1} index of global control unit
- c:={0, 1,..., 7} index of channel in the context of a global control unit
- y:={0, 1, 2} index of used time base

The register and bit field names used inside this chapter have the following meaning:

***Note**: All figures in this chapter, which show block diagrams or gate schematics are drawn in principle to show specific functionality. They cannot completely represent the real implementation of the design.*

In the following sections the terms "100% duty cycle" and "0% duty cycle" will be used. 100% duty cycle is defined as a constant output level on CLSi_TOM_CHx_CTRL.SL of TOM_OUT*[x]* over the whole period. 0% duty cycle is defined as a constant output level on !CLSi_TOM_CHx_CTRL.SL of TOM_OUT*[x]* over the whole period.

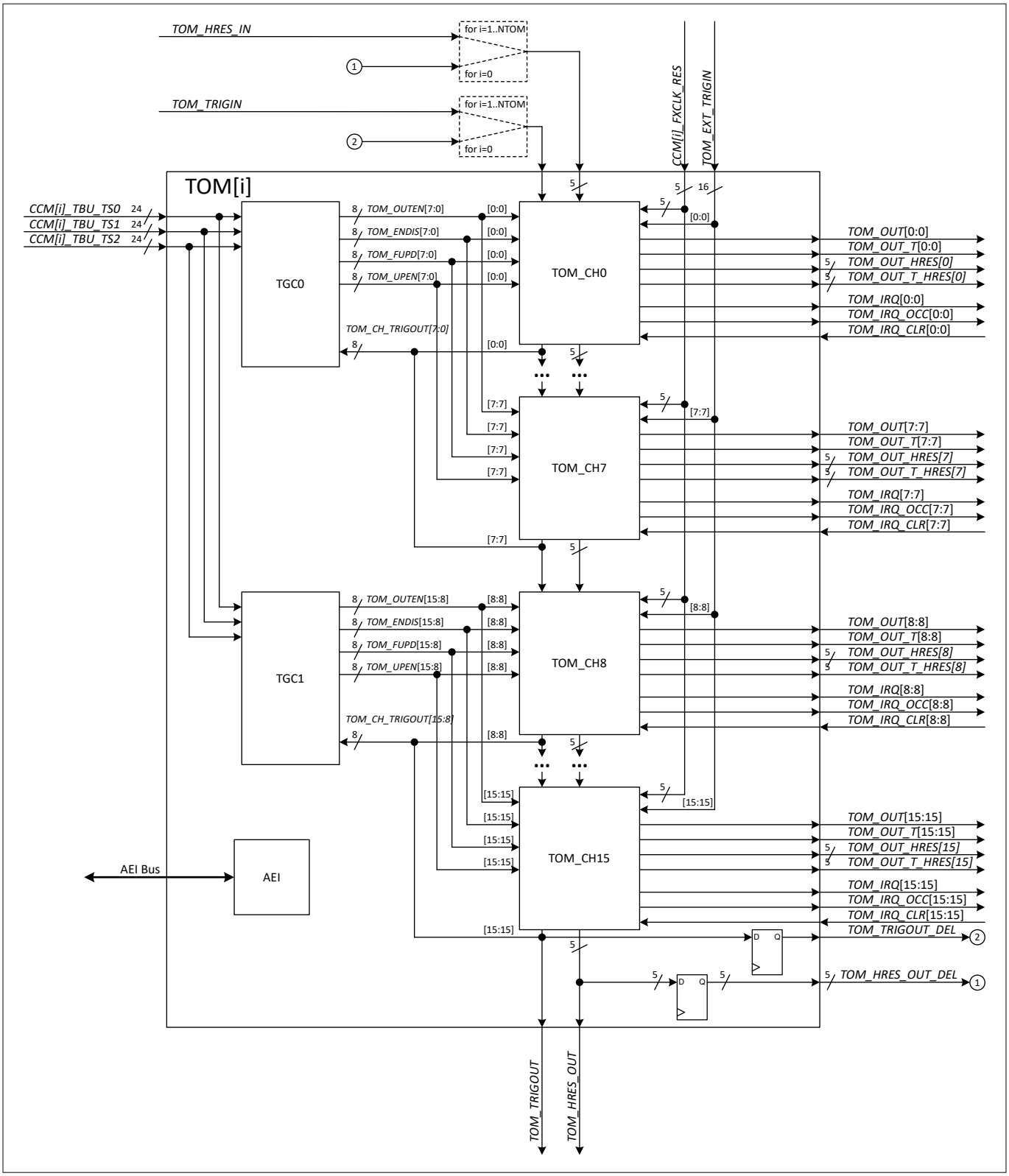


Figure 1057 TOM block diagram

As shown in Figure 1057, each **TOM** module integrates 16 channels with outputs TOM_OUT and TOM_OUT_T. The two sub-modules TGC0 and TGC1 are global channel control units that control the enabling/disabling of the channels and their outputs as well as the update of their period and pulse width register.

The input clocks for the **TOM** channels come from the cluster clock CLS[j]_CLK with the configurable clock resolution CCM[i]_FXCLK_RES signals of the **CCM** sub-module. This gives the freedom to select a programmable input clock for the **TOM** channel counters.

The module **TOM** receives two (three) timestamp values CCM[i]_TBU_TS0, CCM[i]_TBU_TS1 (and CCM[i]_TBU_TS2) in order to realize synchronized output behavior on behalf of a common time base.

As shown in Figure 1057, NTOM-1 **TOM** sub-modules are connected together through a trigger chain. The trigger signal TOM_TRIGIN of **TOM** instance i is coming from TOM_TRIGOUT of **TOM** instance i-1 and the trigger signal TOM_TRIGOUT of **TOM** instance i is routed to TOM_TRIGIN of **TOM** instance i+1. The trigger chain goes across all channels one by one. The trigger input signal TOM_TRIGIN of **TOM** instance i is connected to the trigger input signal TOM_CH_TRIGIN[0:0] of channel 0. The channel 0 can generate a trigger output signal TOM_CH_TRIGOUT[0:0] that is connected to TOM_CH_TRIGIN[1:1] to trigger channel 1. In this way, each channel x is triggered from the preceding channel and then can trigger the next channel (please refer to Figure 1060 and Internal trigger interface for the internal trigger chain details). Channel 15 of **TOM** instance i is routed to trigger channel 0 of the next instance i+1.

The trigger input signal of **TOM** instance 0 is connected to its own output TOM_TRIGOUT_DEL, i.e. the last channel of **TOM** instance 0 can trigger the first channel of **TOM** instance 0 (this path is registered, which means delayed by one cluster clock period).

The function of the interrupt signals TOM_IRQ, TOM_IRQ_OCC and TOM_IRQ_CLR is described in more detail in **eGTM** interrupt concept.

In Figure 1058 the **TOM SPE** interface signals are depicted. Please refer to Sensor pattern evaluation (**SPE**) for more details about the **TOM SPE** interface signals (TOM[i]_SPE_OUT, TOM[i]_SPE_NIPD, TOM[i]_CH[x]_SOUR, TOM[i]_CH[x]_SOUR, TOM[i]_CH[x]_TRIG_CCU0, TOM[i]_CH[x]_TRIG_CCU1, TOM[i]_CH[x]_TRIG_CCU1, TOM[i]_CH[x]_TRIG_CCU1, TOM[i]_CH[x]_TRIG_CCU1, TOM[i]_CH[x]_TRIG_CCU1, TOM[i]_CH[x]_TRIG_CCU1).

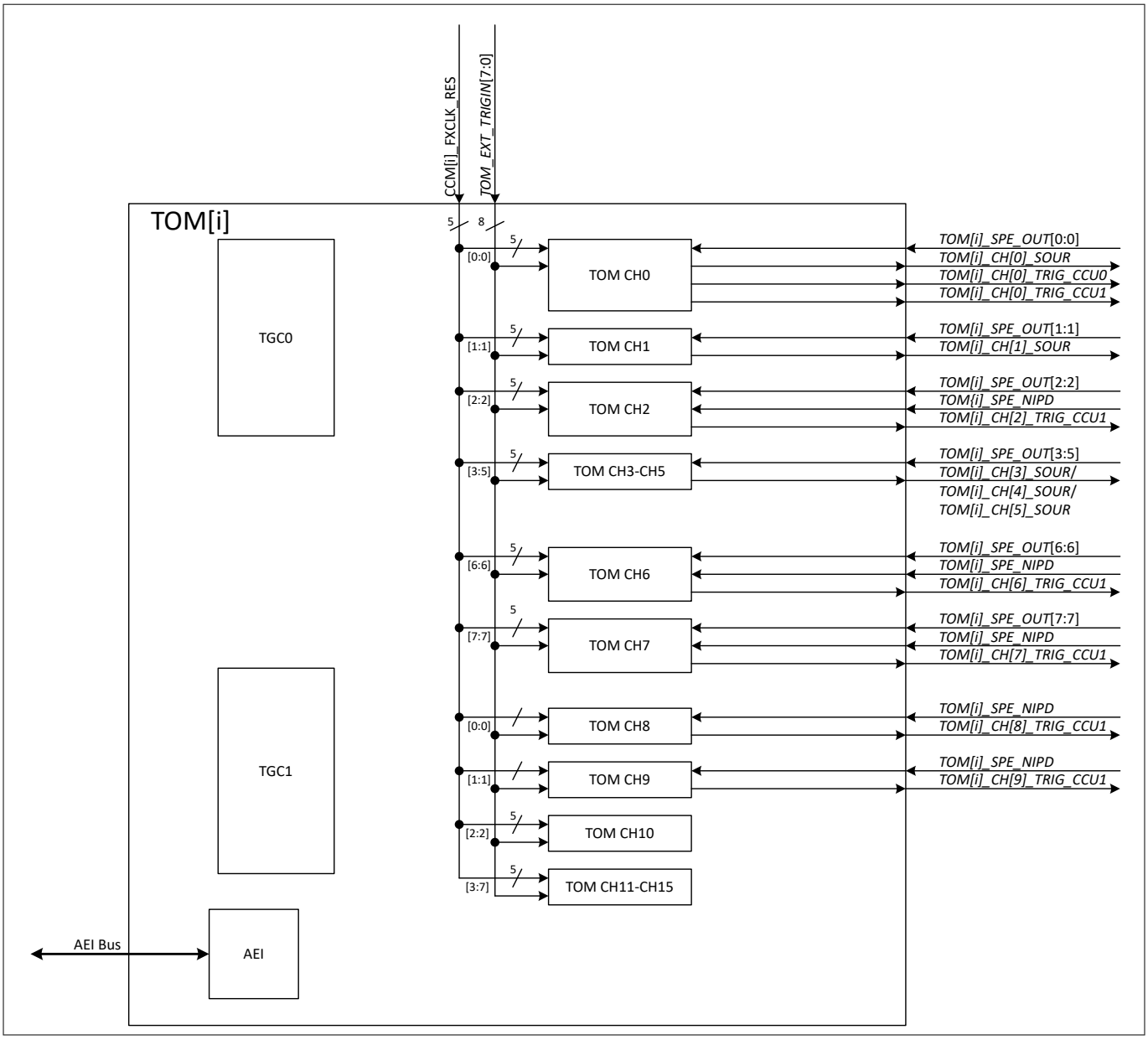


Figure 1058 TOM-SPE interface signals

***Note**: The figure is a general picture from the IP vendor. The exact configuration can be found in the device specific configuration.*

44.3.1.7.2 TOM global channel control (TGC[0], TGC[1])

Overview

Two global channel control units (TGC[0] and TGC[1]) exist to drive a number of individual **TOM** channels synchronously by external or internal events.

Each TGC[g] can drive up to eight **TOM** channels where TGC[0] controls **TOM** channels x (x∈{0, 1,..., 7}) and TGC[1] controls **TOM** channels x (x∈{8, 9,..., 15}).

The **TOM** sub-module supports four different kinds of signaling mechanisms:

- Global enable/disable mechanism for each **TOM** channel with control register CLSi_TOM_TGCg_ENDIS_CTRL and status register CLSi_TOM_TGCg_ENDIS_STAT.
- Global output enable mechanism for each **TOM** channel x with control register CLSi_TOM_TGCg_OUTEN_CTRL and status register CLSi_TOM_TGCg_OUTEN_STAT.
- Global force update mechanism for each **TOM** channel and optionally reset the counter CLSi_TOM_CHx_CN0 with control register CLSi_TOM_TGCg_FUPD_CTRL.
- Global update enable of the register CLSi_TOM_CHx_CM0.CM0, CLSi_TOM_CHx_CM1.CM1 and CLSi_TOM_CHx_CTRL.CLK_SRC for each **TOM** channel x with the control bit field CLSi_TOM_TGCg_GLB_CTRL.UPEN_CTRLc (c=0-7) (x= c+8*g).

TGC sub-unit

Each of the first three individual mechanisms (enable/disable of the channel, output enable and force update) can be driven by three different trigger sources.

The three trigger sources are:

1. Over the configuration interface (bit CLSi_TOM_TGCg_GLB_CTRL.HOST_TRIG
2. The **TBU** timestamp (signal CCM[i]_TBU_TS0, CCM[i]_TBU_TS1, CCM[i]_TBU_TS2 if available)
3. The trigger signal TOM_CH_TRIGOUT[7:0] for TGC[0] or TOM_CH_TRIGOUT[15:8] for TCG[1] (bunch of trigger signals TOM_CH_TRIGOUT[x:x]) which can be the trigger TRIG_CCU0 of channel x, the trigger of preceding channel x-1 (i.e. signal TOM_CH_TRIGOUT[x-1:x-1]) or the external trigger TOM_EXT_TRIGIN[x:x] from **TIM**.

The first way (1.) is to trigger the control mechanism by a direct register write-access via configuration interface (bit CLSi_TOM_TGCg_GLB_CTRL.HOST_TRIG).

The second way (2.) is provided by a compare match trigger on behalf of a specified time base coming from the module **TBU** (selected by bit field CLSi_TOM_TGCg_ACT_TB.TBU_SEL) and the timestamp compare value defined in the bit field CLSi_TOM_TGCg_ACT_TB.ACT_TB.

In this case, the selected **TBU** time base CCM[i]_TBU_TS0/CCM[i]_TBU_TS1/CCM[i]_TBU_TS2 should be configured to be up-counting. A cyclic event compare of CLSi_TOM_TGCg_ACT_TB.ACT_TB and selected CCM[i]_TBU_TS0, CCM[i]_TBU_TS1, CCM[i]_TBU_TS2 is performed. Please refer to Cyclic event compare for more details about cyclic event compare strategy.

The third way (3.) is the input TOM_CH_TRIGOUT (bunch of trigger signals TOM_CH_TRIGOUT[x:x]) coming from the **TOM** channel x, x∈{0, 1,..., 7} for TGC[0] or x∈{8, 9,..., 15} for TGC[1].

The corresponding trigger signal TOM_CH_TRIGOUT[x:x] coming from channel x can be masked by the register CLSi_TOM_TGCg_INT_TRIG.

To enable or disable each individual **TOM** channel, the register CLSi_TOM_TGCg_ENDIS_CTRL and/or CLSi_TOM_TGCg_ENDIS_STAT have to be used.

The register CLSi_TOM_TGCg_ENDIS_STAT controls directly the signals TOM_ENDIS[7:0] if g=0 or TOM_ENDIS[15:8] if g=1 (see Figure 1059). A write-access to this register is possible.

The register CLSi_TOM_TGCg_ENDIS_CTRL is a shadow register that overwrites the value of register CLSi_TOM_TGCg_ENDIS_STAT if one of the three trigger conditions matches.

The update mechanism within the channel is not affected by the TOM_ENDIS control signal. This means that the update of the working register can be executed - depending on the update condition - as soon as it is enabled by setting CLSi_TOM_TGCg_GLB_CTRL.UPEN_CTRLc (c=0-7) $=10_B$.

To ensure a clean restart of a **TOM** channel x, it is recommended to raise a channel reset by setting of bit CLSi_TOM_TGCg_GLB_CTRL.RST_CHc (c=0-7) (x = c + 8*g) after a channel is disabled (TOM_ENDIS[x:x]=0) and before it is enabled again (TOM_ENDIS[x:x]=1).

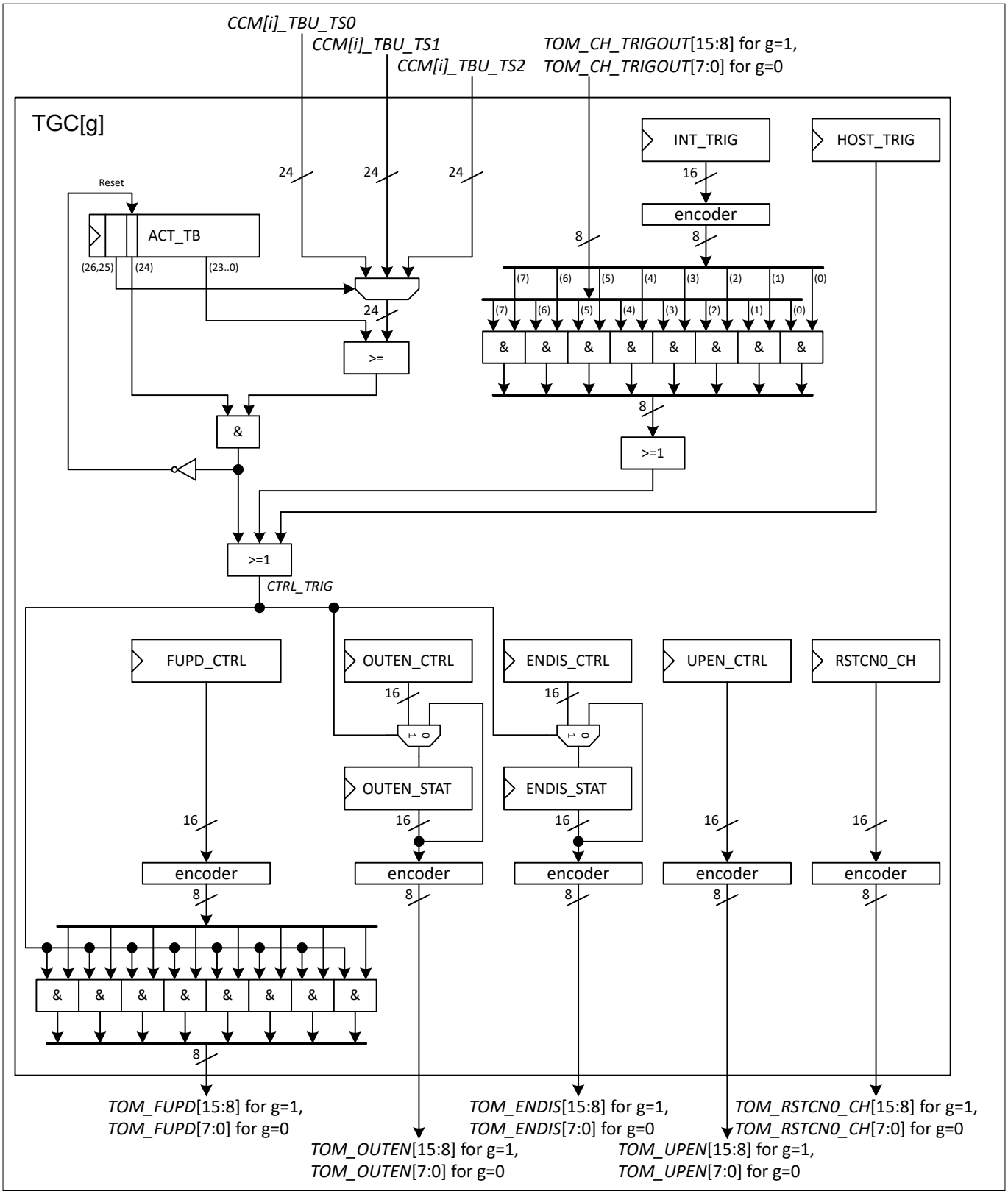


Figure 1059 TOM global channel control mechanism

The output of the individual **TOM** channels can be controlled using the register CLSi_TOM_TGCg_OUTEN_CTRL and CLSi_TOM_TGCg_OUTEN_STAT.

The register CLSi_TOM_TGCg_OUTEN_STAT controls directly the signals TOM_OUTEN[7:0] if g=0 or TOM_OUTEN[15:8] if g=1. A write-access to this register is possible.

CLSi_TOM_TGCg_OUTEN_CTRL is a shadow register that overwrites the value of register CLSi_TOM_TGCg_OUTEN_STAT if one of the three trigger conditions matches.

The trigger condition has always priority over the bus write-access to the CLSi_TOM_TGCg_OUTEN_STAT and CLSi_TOM_TGCg_ENDIS_STAT registers, even if CLSi_TOM_TGCg_OUTEN_CTRL.OUTEN_CTRLc (c=0-7) **(c=k)**/ CLSi_TOM_TGCg_OUTEN_CTRL.**ENDIS_CTRL[k]** is set to 00_B. This means that the bus write-access to CLSi_TOM_TGCg_OUTEN_STAT and CLSi_TOM_TGCg_ENDIS_STAT register is ignored in the clock cycle when the trigger condition is active.

The register CLSi_TOM_TGCg_FUPD_CTRL defines which of the **TOM** channels receive a force update event if the trigger signal CTRL_TRIG is raised while CLSi_TOM_TGCg_FUPD_CTRL.RSTCN0_CHc (c=0-7) determines whether the force update event also reset counter CLSi_TOM_CHx_CN0.CN0 of the channel. If in continuous counting up mode with CLSi_TOM_CHx_CTRL.RST_CCU0=0, the force update mechanism is configured together with CLSi_TOM_TGCg_FUPD_CTRL.RSTCN0_CHc (c=0-7)=10_B and CLSi_TOM_TGCg_FUPD_CTRL.FUPD_CTRLc (c=0-7)=10_B, the force update event will simultaneously update the operation registers, reset the counter and also set the output to be CLSi_TOM_CHx_CTRL.SL. That means the force update mechanism can be used to stop the current PWM signal generation and restart a new PWM signal generation with new PWM parameters.

The force update request is synchronized to the selected CCM[i]_FXCLK_RES as shown in Figure 1060 and then executed. The execution of the force update does not depend on the state of CLSi_TOM_TGCg_ENDIS_STAT.ENDIS_STATc (c=0-7).

The TOM_RSTCN0_CH signal is not synchronized with the selected CCM[i]_CLK_RES. In order to ensure that the reset function is executed, the bit field CLSi_TOM_TGCg_FUPD_CTRL.RSTCN0_CHc (c=0-7) may only be set to 01_B after the selected CCM[i]_CLK_RES has occured.

The register bit CLSi_TOM_TGCg_GLB_CTRL.UPEN_CTRLc (c=0-7) defines for which **TOM** channel the update of the working register CLSi_TOM_CHx_CM0.CM0, CLSi_TOM_CHx_CM1.CM1, CLSi_TOM_CHx_CTRL.SL and CLSi_TOM_CHx_CTRL.CLK_SRC by the corresponding shadow register CLSi_TOM_CHx_SR0.SR0, CLSi_TOM_CHx_SR1.SR1, CLSi_TOM_CHx_CTRL_SR.SL_SR and CLSi_TOM_CHx_CTRL_SR.CLK_SRC_SR is enabled. If update is enabled, the registers CLSi_TOM_CHx_CM0.CM0, CLSi_TOM_CHx_CM1.CM1 and CLSi_TOM_CHx_CTRL.CLK_SRC will be updated on reset of counter CLSi_TOM_CHx_CN0.CN0 (see Figure 1060). An exception is the configuration of CLSi_TOM_CHx_CTRL.SR0_TRIG=1 which enables the trigger generation defined by CLSi_TOM_CHx_SR0.SR0 then CLSi_TOM_CHx_CM0 is not updated with CLSi_TOM_CHx_SR0 (please refer to here).

44.3.1.7.3 TOM channel architecture

Each individual **TOM** channel comprises a Counter Compare Unit 0 (CCU0) which contains the counter CLSi_TOM_CHx_CN0.CN0 and the period register CLSi_TOM_CHx_CM0.CM0, a Counter Compare Unit 1 (CCU1) which contains the pulse width register CLSi_TOM_CHx_CM1.CM1 and the Signal Output Generation Unit (SOU) which contains the output register SOUR (see Figure 1060). The architecture is depicted in Figure 1060.

***Note**:* Figure 1060 *doesn't reflect the real implementation of the circuit but is intended to explain the functionality of the channel in a better manner.*

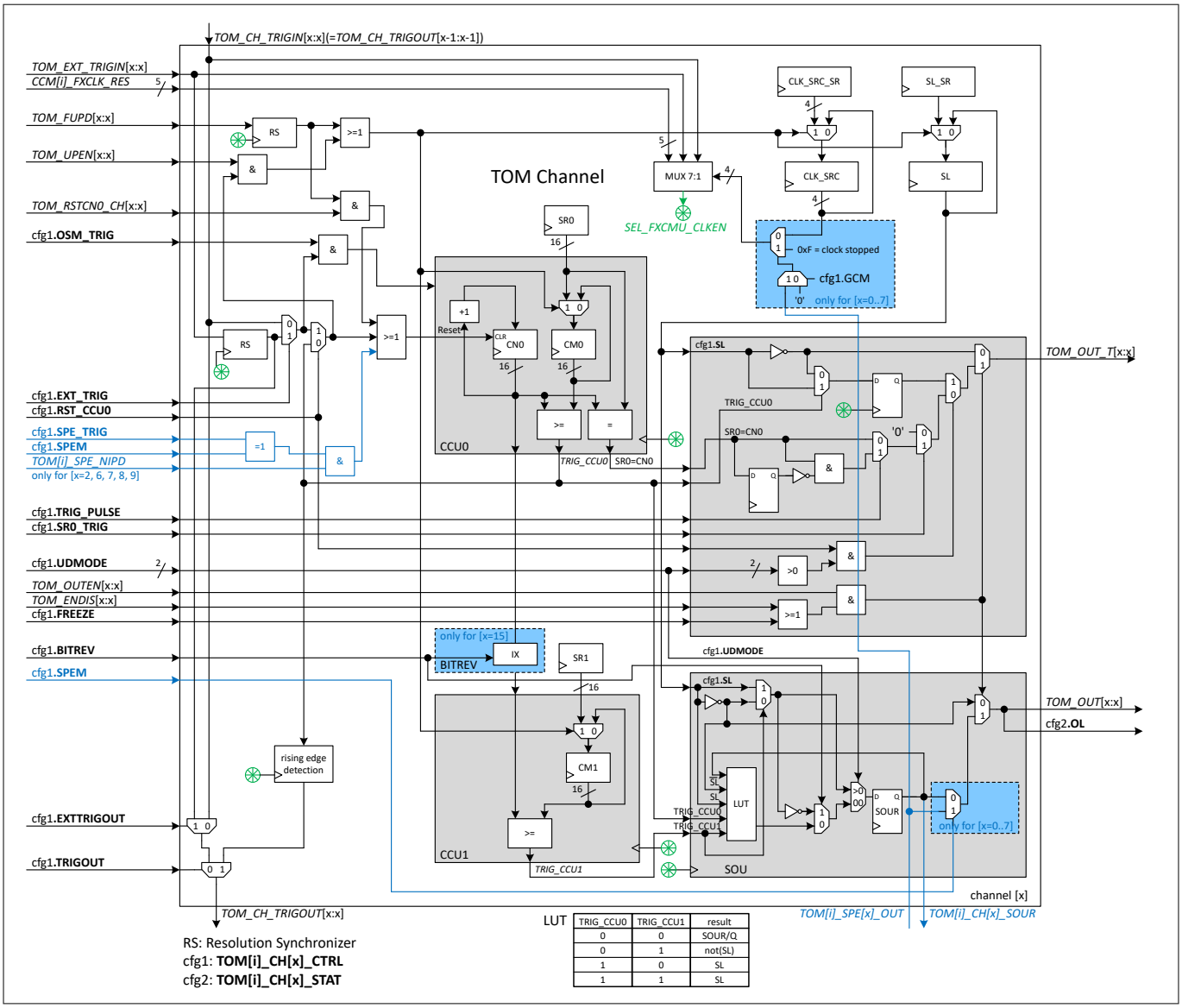


Figure 1060 TOM channel architecture

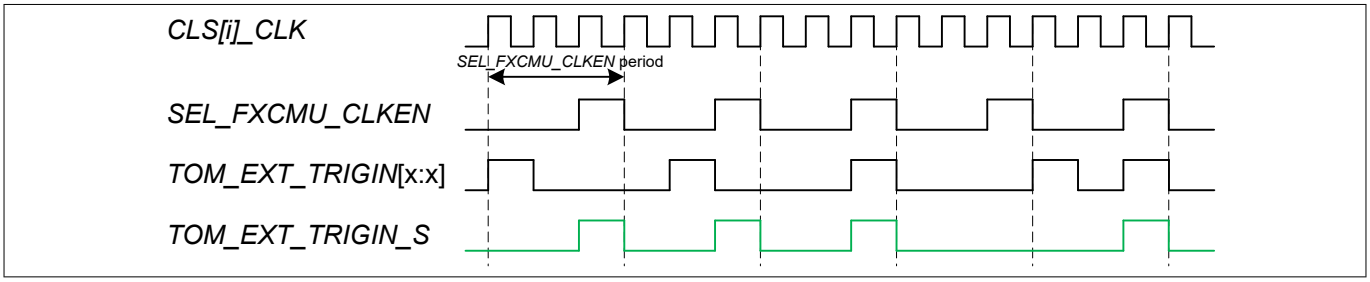


Figure 1061 Timing diagram of RS unit

Most functionalities (e.g. the counter CLSi_TOM_CHx_CN0.CN0 in CCU0 unit) are synchronized with the selected clock enable signal SEL_FXCMU_CLKEN that is configured by CLSi_TOM_CHx_CTRL.CLK_SRC. This configuration register can select the clock source from fixed clock enable signals CCM[i]_FXCLK_RES that are provided from **CCM** sub-module. The signal SEL_FXCMU_CLKEN looks like clock enable signal that is shown in Figure 1061. Most functionality modes are only active when SEL_FXCMU_CLKEN = 1 and the consequent operation registers and outputs are always synchronously updated at the falling edge of SEL_FXCMU_CLKEN

pulse. The functionalities synchronous to the selected clock resolution SEL_FXCMU_CLKEN are depicted with the green round symbol in Figure 1060.

In all **TOM** channels the operation registers CLSi_TOM_CHx_CN0.CN0, CLSi_TOM_CHx_CM0.CM0 and CLSi_TOM_CHx_CM1.CM1 and the shadow registers CLSi_TOM_CHx_SR0.SR0 and CLSi_TOM_CHx_SR1.SR1 are 16 bits wide. As shown in Figure 1060, the unit CCU0 provides different counting up/down functionalities in various modes. The comparators inside CCU0 and CCU1 perform selectable signed greater-equal or less-equal comparison against different parameters in different modes and then generate TRIG_CCU0=1 when the compare match event in CCU0 occurs or generate TRIG_CCU1=1 when the compare match event in CCU1 occurs.

The Signal Output Unit (SOU) generates the output signal for each **TOM** channel. The output signal level depends on the bit CLSi_TOM_CHx_CTRL.SL in combination with the counter behavior and its compare events in CCU0 and CCU1. The initial signal output level for the channel is the reverse value of the bit CLSi_TOM_CHx_CTRL.SL.

In continuous counting up mode, CLSi_TOM_CHx_CM0.CM0 represents PWM period while CLSi_TOM_CHx_CM1.CM1 is PWM pulse width. In order to generate various PWM signals, CCU0 counts up CLSi_TOM_CHx_CN0.CN0 and compares it with CLSi_TOM_CHx_CM0.CM0 while CCU1 compares it with CLSi_TOM_CHx_CM1.CM1. Depending on the configuration bit CLSi_TOM_CHx_CTRL.RST_CCU0, the counter CLSi_TOM_CHx_CN0.CN0 can be reset, when the counter value is equal to the compare value CLSi_TOM_CHx_CM0.CM0 (i.e. CLSi_TOM_CHx_CN0.CN0 counts only from 0 to CLSi_ATOM_CHx_CM0.CM0-1 and is then reset to 0) or when it is triggered by the **TOM[i]** trigger signal TOM_CH_TRIGIN[x:x] (TOM_CH_TRIGOUT[x-1:x-1]) of the preceding channel x-1 which can also be the last channel of preceding instance **TOM[i-1]**) or when it is triggered by the external trigger signal TOM_EXT_TRIGIN[x:x] from **TIM** module i.

As an exception, the input TOM_TRIGIN of instance TOM0 is triggered by its own last channel 15 via signal TOM_CH_TRIGOUT[15:15].

The external trigger signal TOM_EXT_TRIGIN[x:x] is synchronous to the clock source selected by RS (Resolution Synchronizer) unit and then taken into use.

With such a RS unit, in each SEL_FXCMU_CLKEN period (i.e. selected clock resolution signal), any event (or pulses) of TOM_EXT_TRIGIN will be synchronized by RS unit on TOM_EXT_TRIGIN_S that looks exactly the same as SEL_FXCMU_CLKEN. Figure 1061 shows the timing diagram of RS unit. If TOM_EXT_TRIGIN has one or more pulses in a SEL_FXCMU_CLKEN period, the synchronized output TOM_EXT_TRIGIN_S will have 1 pulse at the end of this period. Otherwise, TOM_EXT_TRIGIN_S is always 0.

The trigger signal TOM_CH_TRIGIN[x:x] coming from the preceding channel may not be synchronous to the selected clock resolution of channel x. If the counter CLSi_TOM_CHx_CN0.CN0 is reset by the trigger signal TOM_CH_TRIGIN[x:x] of a preceding channel with CLSi_TOM_CHx_CTRL.RST_CCU0=1, the preceding triggering channel should be configured with the same CCM[i]_FXCLK_RES as the triggered channel.

In continuous counting up mode, once the comparison event of CLSi_TOM_CHx_CN0.CN0 >= CLSi_TOM_CHx_CM0.CM0-1 in unit CCU0 occurs, TRIG_CCU0 = 1 will be set to trigger the output generation of TOM_OUT[x:x] in the SOU sub-unit together with the corresponding interrupt signals. In order to trigger the next channel, the rising edge detection unit can detect each rising edge of TRIG_CCU0 (i.e. each comparison event) and generate a single cluster clock pulse on TOM_CH_TRIGOUT[x:x] when the selected clock resolution signal SEL_FXCMU_CLKEN =1 (i.e. synchronous to the selected clock resolution). In addition, TOM_CH_TRIGOUT[x:x] can also be configured to be TOM_CH_TRIGIN[x:x] (TOM_CH_TRIGOUT[x-1:x-1] of the preceding channel x-1) or the external trigger signal TOM_EXT_TRIGIN[x:x] from **TIM**.

Similarly, once the comparison event of CLSi_TOM_CHx_CN0.CN0 >= CLSi_TOM_CHx_CM1.CM1-1 in unit CCU1 occurs, TRIG_CCU1 = 1 will be set to trigger the output generation of TOM_OUT[x:x] in the SOU sub-unit together with the corresponding interrupt signals.

When CLSi_TOM_CHx_CN0.CN0 < CLSi_TOM_CHx_CM0.CM0-1, TRIG_CCU0 will be reset as 0. Similarly, when CLSi_TOM_CHx_CN0.CN0 < CLSi_TOM_CHx_CM1.CM1-1, TRIG_CCU1 will be reset to 0.

As shown in Figure 1060, in continuous counting up mode according to the LUT with TRIG_CCU0, TRIG_CCU1 and the signal level configuration CLSi_TOM_CHx_CTRL.SL, SOU unit can generate TOM_OUT[x:x] with various PWM timing characteristics when the channel and the output are both enabled (TOM_OUTEN[x:x]=1 and TOM_ENDIS[x:x]=1). Please refer to the chapter of different modes for more details on PWM signal generation.

The output level on the **TOM** channel output pin TOM_OUT[x:x] is captured in bit CLSi_TOM_CHx_STAT.OL (see Figure 1060)

If a **TOM** channel is disabled by the register CLSi_TOM_TGCg_OUTEN_STAT, the actual value of the channel output at TOM_OUT[x:x] is defined by the signal level bit (CLSi_TOM_CHx_CTRL.SL).

If the output is enabled, the output at TOM_OUT[x:x] depends on value of register SOUR (see Figure 1060).

In addition to the bit CLSi_TOM_CHx_CTRL.SL, a shadow register exists in CLSi_TOM_CHx_CTRL_SR.SL_SR. With this shadow register it is possible to change the signal level value synchronous to the update of the operation registers CLSi_TOM_CHx_CM0.CM0 and CLSi_TOM_CHx_CM1.CM1 from its shadow registers CLSi_TOM_CHx_SR0.SR0 and CLSi_TOM_CHx_SR1.SR1.

The operating clock resolution from **CMU** is specified in the bit field CLSi_TOM_CHx_CTRL.CLK_SRC. In addition, a shadow register CLSi_TOM_CHx_CTRL_SR.CLK_SRC_SR exists. With this shadow register it is possible to change the selected clock resolution from **CMU** synchronously to the update of the operation registers CLSi_TOM_CHx_CM0.CM0 and CLSi_TOM_CHx_CM1.CM1 from its shadow registers CLSi_TOM_CHx_SR0.SR0 and CLSi_TOM_CHx_SR1.SR1.

Due to the fixed clock resolution scheme from **CMU**, the first clock resolution period may differ from expected value, depending on the point of time, the CLSi_TOM_CHx_CTRL.CLK_SRC bit field is updated from its shadow register bit field CLSi_TOM_CHx_CTRL_SR.CLK_SRC_SR.

When the output is disabled (TOM_OUTEN[x:x]=0), the output signals TOM_OUT[x:x] and TOM_OUT_T[x:x] are the inverse value of the CLSi_TOM_CHx_CTRL.SL bit. When the output is enabled (TOM_OUTEN[x:x]=1), the output signals TOM_OUT[x:x] and TOM_OUT_T[x:x] are dependent on different operation modes, the channel enable signal TOM_ENDIS[x:x] and the freeze mode enable register CLSi_TOM_CHx_CTRL.FREEZE. (Please refer to register description of CLSi_TOM_CHx_CTRL.SL for more details.) Especially, in some modes the output signal TOM_OUT_T[x:x] is not supported. So it is always clamped to 0 if the output and the channel are both enabled (i.e. TOM_OUTEN[x:x]=1 and TOM_ENDIS[x:x]=1). When the output is enabled (TOM_OUTEN[x:x]=1) but the channel is disabled (TOM_ENDIS[x:x]=0), if CLSi_TOM_CHx_CTRL.FREEZE=0, TOM_OUT_T[x:x] is ! CLSi_TOM_CHx_CTRL.SL. When the output is enabled (TOM_OUTEN[x:x]=1) but the channel is disabled (TOM_ENDIS[x:x]=0), if CLSi_TOM_CHx_CTRL.FREEZE=1, TOM_OUT_T[x:x] is clamped to 0.

When the channel is disabled (TOM_ENDIS[x:x]=0), the counting and comparing functionalities in CCU0 and CCU1 are deactivated in different modes and thus TRIG_CCU0 = 0, TRIG_CCU1 = 0. Therefore, TOM_CH_TRIGOUT[x:x] is not generated if CLSi_TOM_CHx_CTRL.TRIGOUT = 1, regardless of TOM_OUTEN[x:x].

External trigger interface

Each channel x of **TOM** instance i has an input signal TOM_EXT_TRIGIN[x:x] that is connected to TIM_EXT_CAPTURE[x:x] of **TIM** instance i if x ∈ {0, 1, ..., 7} and TIM_EXT_CAPTURE[x-8:x-8] of **TIM** instance i if x ∈ {8, 9, ..., 15} (see Figure 1019). Via this signal, events (high pulses with the length of 1 CLS[j]_CLK period) generated by the **TIM** module i can be used to control the functionality of the **TOM** channel in multiple ways:

- TOM_EXT_TRIGIN[x:x] events can be used to define the clock source of the selected channel mode (CLSi_TOM_CHx_CTRL.CLK_SRC = 1110_B).
- TOM_EXT_TRIGIN[x:x] events can be used to reset the counter register CLSi_TOM_CHx_CN0 in continuous counting up mode or change the counting direction in counting up-down mode or trigger the start of counting in one-shot mode (activated by CLSi_TOM_CHx_CTRL.EXT_TRIG = 1 && CLSi_TOM_CHx_CTRL.RST_CCU0 = 1). In this case, if CLSi_TOM_TGCg_GLB_CTRL.UPEN_CTRLc (c=0-7) = 10_B is configured, TOM_EXT_TRIGIN[x:x] events can also trigger the update of the operation registers from

their shadow registers. Any TOM_EXT_TRIGIN[x:x] event is synchronized in the **TOM** channel as specified in Figure 1061.

- TOM_EXT_TRIGIN[x:x] can be used as a trigger event for the following channels (activated by CLSi_TOM_CHx_CTRL.EXTTRIGOUT = 1 && CLSi_TOM_CHx_CTRL.TRIGOUT = 0). Any TOM_EXT_TRIGIN[x:x] event is synchronized in the **TOM** channel as specified in Figure 1061.

In certain configurations of the **TIM** module, the TOM_EXT_TRIGIN[x:x] signal pulses can have a length of more than 1 cluster clock-cycle. In this case, the functionality of the **TOM** is not specified and it is highly recommended to prevent this configuration in applications.

Before triggering the functionalities mentioned above, the TOM_EXT_TRIGIN[x:x] signal is always synchronized with the current selected clock enable signal in RS unit as shown in Figure 1061.

Internal trigger interface

Each **TOM** channel x has an input signal TOM_CH_TRIGIN[x:x] that is coming from TOM_CH_TRIGOUT[x-1:x-1] from the preceding channel. Via this signal, events (high pulses with the length of one CLS[j]_CLK period) generated by the preceding **TOM** channel, can be used to control the functionality of **TOM** channel in multiple ways:

- TOM_CH_TRIGIN[x:x] events can be used to define the clock source of the selected channel mode (CLSi_TOM_CHx_CTRL.CLK_SRC = 1101_B).
- TOM_CH_TRIGIN[x:x] events can be used to reset the counter register CLSi_TOM_CHx_CN0 in continuous counting up mode or change the counting direction in counting up-down mode or trigger the start of counting in one-shot mode (activated by CLSi_TOM_CHx_CTRL.EXT_TRIG = 0 && CLSi_ATOM_CHx_CTRL.RST_CCU0 = 1). In this case, if CLSi_TOM_TGCg_GLB_CTRL.UPEN_CTRLc (c=0-7) = 10_B is configured, TOM_CH_TRIGIN[x:x] events can also trigger the update of the operation registers from their shadow registers.
- TOM_CH_TRIGIN[x:x] can be used as a trigger event for following channels (activated by CLSi_TOM_CHx_CTRL.EXTTRIGOUT = 0 && CLSi_TOM_CHx_CTRL.TRIGOUT = 0).

Especially, for channel 0 of **TOM** module i, i ∈ {1, 2, ..., NTOM-1}, TOM_CH_TRIGIN[0:0] comes from TOM_CH_TRIGOUT[15:15] of the preceding **TOM** instance [i-1] and may be delayed by a register with the cluster clock CLS[j]_CLK in channel 0 before triggering the functionalities as mentioned above. This delay is controlled by the device configuration variable TOM_TRIG_CHAIN (indicated by CLSi_CCM_HW_CONF.TOM_TRIG_CHAIN). The delay exists after every N cluster (N = TOM_TRIG_CHAIN). For TOM0, TOM_CH_TRIGIN[0:0] is connected to TOM_TRIGOUT_DEL via the signal TOM_CH_TRIGOUT[15:15] of TOM0 itself as shown in Figure 1057. Furthermore, the cluster i can operate on faster **eGTM** clock CLK, if the device is defined with FAST_CLS_CLK[i] = 1. In this case, it is allowed to have one more register that is clocked with CLK to delay TOM_CH_TRIGIN[8:8] in channel 8 before triggering the functionalities. This is controlled by another **ATOM** device configuration parameter TOM_TRIG_INTCHAIN (it can only be valid if FAST_CLS_CLK[i] = 1 and indicated by CLSi_CCM_HW_CONF.TOM_TRIG_INTCHAIN). The registers for these special cases are not shown in Figure 1057.

Unlike TOM_EXT_TRIGIN[x:x], TOM_CH_TRIGIN[x:x] is not synchronized with the selected clock enable signal of channel x. For the application of 2 channels generating PWM signals synchronously, CLSi_TOM_CHx_CTRL.CLK_SRC of the triggering channel and the triggered channel should be configured with the same clock resolution and the same cluster clock.

Internal trigger interface

As shown in Figure 1060, each **TOM** channel x has an input signal TOM_CH_TRIGIN[x:x] that is coming from TOM_CH_TRIGOUT[x-1:x-1] from the preceding channel. Via this signal, events (high pulses with the length of 1 CLS[j]_CLK period) generated by the preceding **TOM** channel, can be used to control the functionality of **TOM** channel in multiple ways:

- *TOM_AGC_TRIGIN*[x:x] events can be used to define the clock source of the selected channel mode (CLSi_TOM_CHx_CTRL.CLK_SRC = 1101_B).

- *TOM_AGC_TRIGIN*[x:x] events can be used to reset the counter register CLSi_TOM_CHx_CN0 in continuous counting up mode or change the counting direction in counting up-down mode or trigger the start of counting in one-shot mode (activated by CLSi_TOM_CHx_CTRL.EXT_TRIG = 0 && CLSi_TOM_CHx_CTRL.RST_CCU0 = 1). In this case, if CLSi_TOM_TGCg_GLB_CTRL.UPEN_CTRLc (c=0-7) = 10_B is configured, TOM_CH_TRIGIN[x:x] events can also trigger the update of the operation registers from their shadow registers.
- TOM_CH_TRIGIN[x:x] can be used as a trigger event for following channels (activated by CLSi_TOM_CHx_CTRL.EXTTRIGOUT = 0 && CLSi_TOM_CHx_CTRL.TRIGOUT = 0).

Especially, for channel 0 of **TOM** module i, i ∈ {1, 2, ..., NTOM-1}, TOM_CH_TRIGIN[0:0] is connected to TOM_TRIGIN of instance [i] that comes from TOM_CH_TRIGOUT[15:15] of the preceding **TOM** instance [i-1] and may be delayed by a register with the cluster clock CLS[j]_CLK in channel 0 before triggering the functionalities as mentioned above. This delay is controlled by the device configuration variable TOM_TRIG_CHAIN (indicated by CLSi_CCM_HW_CONF.TOM_TRIG_CHAIN). The delay exists after every N cluster (N = TOM_TRIG_CHAIN). For TOM0, TOM_CH_TRIGIN[0:0] is connected to TOM_TRIGOUT_DEL via the signal TOM_CH_TRIGOUT[15:15] of TOM0 itself as shown in Figure 1057. Furthermore, the cluster i can operate on faster **eGTM** clock CLK, if the device is defined with FAST_CLS_CLK[i] = 1. In this case, it is allowed to have one more register that is clocked with CLK to delay TOM_CH_TRIGIN[8:8] in channel 8 before triggering the functionalities. This is controlled by another **TOM** device configuration parameter TOM_TRIG_INTCHAIN (it can only be valid if FAST_CLS_CLK[i] = 1 and indicated by CLSi_CCM_HW_CONF.TOM_TRIG_INTCHAIN). The registers for these special cases are not shown in Figure 1060.

Unlike TOM_EXT_TRIGIN[x:x], TOM_CH_TRIGIN[x:x] is not synchronized with the selected clock enable signal of channel x. For the application of 2 channels generating PWM signals synchronously, CLSi_TOM_CHx_CTRL.CLK_SRC of the triggering channel and the triggered channel should be configured with the same clock resolution and the same cluster clock.

Pulse width, period, signal level and clock frequency update mechanisms

The two action registers CLSi_TOM_CHx_CM0.CM0 and CLSi_TOM_CHx_CM1.CM1 can be reloaded with the content of the shadow registers CLSi_TOM_CHx_SR0.SR0 and CLSi_TOM_CHx_SR1.SR1. The bit CLSi_TOM_CHx_CTRL.CLK_SRC that determines the clock frequency of the counter CLSi_TOM_CHx_CN0.CN0 can be reloaded with its shadow register CLSi_TOM_CHx_CTRL_SR.CLK_SRC_SR. The signal level defined in bit field CLSi_TOM_CHx_CTRL.SL can be reloaded with its shadow register CLSi_TOM_CHx_CTRL_SR.SL_SR.

With the update of the register CLSi_TOM_CHx_CTRL.CLK_SRC at the end of a period a new counter CLSi_TOM_CHx_CN0.CN0 clock frequency can be easily adjusted.

The update of the configuration registers CLSi_TOM_CHx_CM0.CM0, CLSi_TOM_CHx_CM1.CM1, CLSi_TOM_CHx_CTRL.SL and CLSi_TOM_CHx_CTRL.CLK_SRC with the content of their shadow registers can be performed under the following conditions:

- In case CLSi_TOM_CHx_CTRL.RST_CCU0 =0_B when the specified CLSi_TOM_CHx_CTRL.UDMODE compare condition is fulfilled, the update will be performed. Additional information is given in the bit field description of CLSi_TOM_CHx_CTRL.UDMODE. The update mechanism must be enabled by configuring CLSi_TOM_TGCg_GLB_CTRL.UPEN_CTRLc (c=0-7) =10_B.
- In case CLSi_TOM_CHx_CTRL.RST_CCU0 =1_B when the channel [x] is triggered by the signal TOM_CH_TRIGIN*[x:x]* from another **TOM** channel or by the external trigger signal TOM_EXT_TRIGIN of the assigned **TIM** channel, after it is synchronized to the selected CCM[i]_FXCLK_RES of channel [x], the update will also be performed. The update mechanism must be enabled by configuring CLSi_TOM_TGCg_GLB_CTRL.UPEN_CTRLc (c=0-7) =10_B.
- These configuration registers can be force-updated via a trigger signal via TOM_FUPD or via TOM_EXT_TRIGIN.
- If CLSi_TOM_CHx_CTRL.RST_CCU0 =0_B, the operating registers are continuously updated as long as CLSi_TOM_CHx_CM0.CM0 =0_H. This update condition must be enabled by configuring CLSi_TOM_TGCg_GLB_CTRL.UPEN_CTRLc (c=0-7) =10_B.

Based on the update mechanisms above, these registers CLSi_TOM_CHx_CM0.CM0, CLSi_TOM_CHx_CM1.CM1, CLSi_TOM_CHx_CTRL.SL, CLSi_TOM_CHx_CTRL.CLK_SRC can be updated synchronously and asynchronously. Synchronous update means that the pulse width, period duration, signal level and the counter operating clock frequency can be changed at the end of the running PWM period. Asynchronous update means that the pulse width, period duration, signal level and the counter operating clock frequency can also be changed during the actual running PWM period. The following two Synchronous update of pulse width only and Asynchronous update of pulse width only provide examples of synchronous update and asynchronous update of pulse width in continuous up mode with CLSi_TOM_CHx_CTRL.RST_CCU0 = 0.

An update of pulse width, period and counter CLSi_TOM_CHx_CN0.CN0 clock frequency that takes effect synchronous to the start of a new period can be easily reached by performing following steps:

1. Disable the update of the action register with the content of the corresponding shadow register by setting the channel-specific configuration bit CLSi_TOM_TGCg_GLB_CTRL.UPEN_CTRLc (c=0-7) = 01_B. (Configuring CLSi_TOM_CHx_CTRL.UDMODE=01_B if in counting up-down mode.)
2. Write new desired values to CLSi_TOM_CHx_SR0, CLSi_TOM_CHx_SR1, CLSi_TOM_CHx_CTRL.SL, CLSi_TOM_CHx_CTRL_SR.CLK_SRC_SR.
3. Enable update of the action register by setting the channel-specific configuration bit CLSi_TOM_TGCg_GLB_CTRL.UPEN_CTRLc (c=0-7)= 01_B.

Asynchronous update of the operation registers can be realized by hot reconfiguring directly the registers CLSi_TOM_CHx_CM0.CM0, CLSi_TOM_CHx_CM1.CM1, CLSi_TOM_CHx_CTRL.SL, CLSi_TOM_CHx_CTRL.CLK_SRC or hot reconfiguring their shadow registers and then using the force update mechanism (configuring CLSi_TOM_TGCg_FUPD_CTRL.FUPD_CTRLc (c=0-7) = 01_B, CLSi_TOM_TGCg_FUPD_CTRL.RSTCN0_CHc (c=0-7) = 01_B and then writing bit CLSi_TOM_TGCg_GLB_CTRL.HOST_TRIG = 1). Such asynchronous update mechanism is sometimes helpful to change the PWM parameters of the current running PWM signal. For example, in continuous counting up mode, the current running PWM period can be reduced by asynchronously updating the value of CLSi_TOM_CHx_CM0.CM0 with a smaller value before the end of the current period. However, since the behavior depends on the access or update time of registers and the current running operation phase of the channel, which is often difficult to control, then the behavior after an asynchronous update can be unpredictable and unexpected. Therefore, such an asynchronous update must be carefully performed in real application. The following steps are recommended to apply the force update mechanism on channel x:

1. Disable the update of the action register with the content of the corresponding shadow register by setting the channel-specific configuration bit field CLSi_TOM_TGCg_GLB_CTRL.UPEN_CTRLc (c=0-7) = 01_B.
2. Write new desired values to CLSi_TOM_CHx_SR0.SR0, CLSi_TOM_CHx_SR1.SR1, CLSi_TOM_CHx_CTRL_SR.SL_SR, CLSi_TOM_CHx_CTRL_SR.CLK_SRC_SR.
3. Enable the force update of the operation registers by setting the channel-specific configuration bit field CLSi_TOM_TGCg_FUPD_CTRL.FUPD_CTRLc (c=0-7) = 10_B, CLSi_TOM_TGCg_FUPD_CTRL.RSTCN0_CHc (c=0-7) = 01_B.
4. Write CLSi_TOM_TGCg_GLB_CTRL.HOST_TRIG = 1 to trigger the force update.

Synchronous update of pulse width only

A synchronous update of the pulse width only can be done by simply writing the desired new value to register CLSi_TOM_CHx_SR1.SR1 without prior disable of the update mechanism (as described in the chapter above). The new pulse width is then applied in the period following the period where the update of register CLSi_TOM_CHx_SR1.SR1 is done.

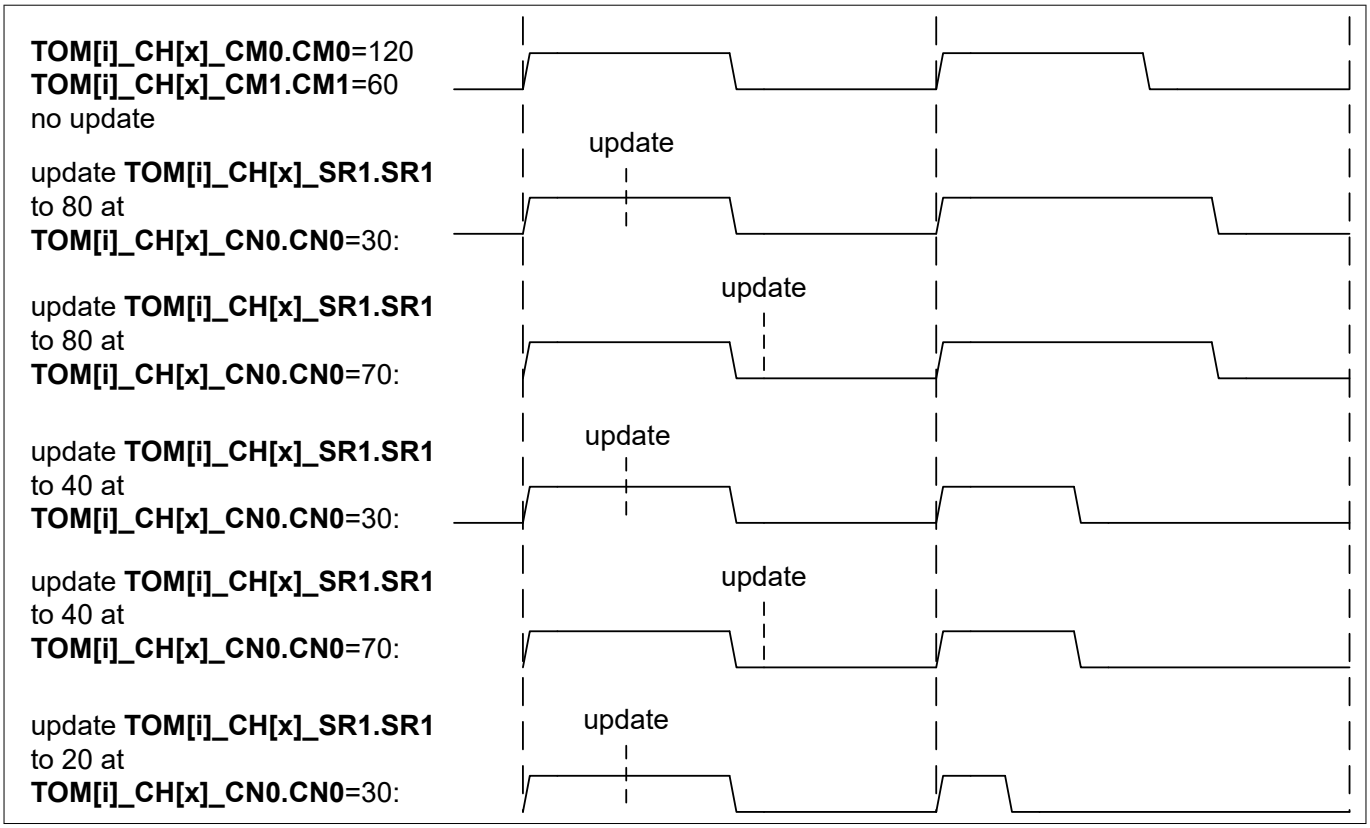


Figure 1062 Synchronous update of pulse width only

Asynchronous update of pulse width only

If the update of the pulse width should be performed independent of the start of a new period (asynchronous), it is mandatory to additionally disable the synchronous update mechanism as a whole (i.e. configuring bits CLSi_TOM_TGCg_GLB_CTRL.UPEN_CTRLc (c=0-7)=01_B of corresponding channel x) and then update the operation register. On the one hand, the desired new value can be written directly to register CLSi_TOM_CHx_CM1.CM1. On the other hand, it can be realized by writing the new value to CLSi_TOM_CHx_SR1.SR1 and afterwards updating the value of CLSi_TOM_CHx_CM1.CM1 by using the asynchronous force update mechanism (configuring CLSi_TOM_TGCg_FUPD_CTRL.FUPD_CTRLc (c=0-7) = 10_B, CLSi_TOM_TGCg_FUPD_CTRL.RSTCN0_CHc (c=0-7) = 01_B and then writing bit CLSi_TOM_TGCg_GLB_CTRL.HOST_TRIG = 1).

Depending on the time of update of CLSi_TOM_CHx_CM1.CM1 in relation to the actual value of CLSi_TOM_CHx_CN0.CN0 and CLSi_TOM_CHx_CM1.CM1, the new pulse width is applied in the current period or the following period. The behavior of the output signal due to the different possibilities of an asynchronous update during a PWM period is shown in Figure 1063. The new pulse width may jitter from update to update by maximum one period (given by CLSi_TOM_CHx_CM0.CM0). However, the period remains unchanged. In any case the creation of glitches is avoided.

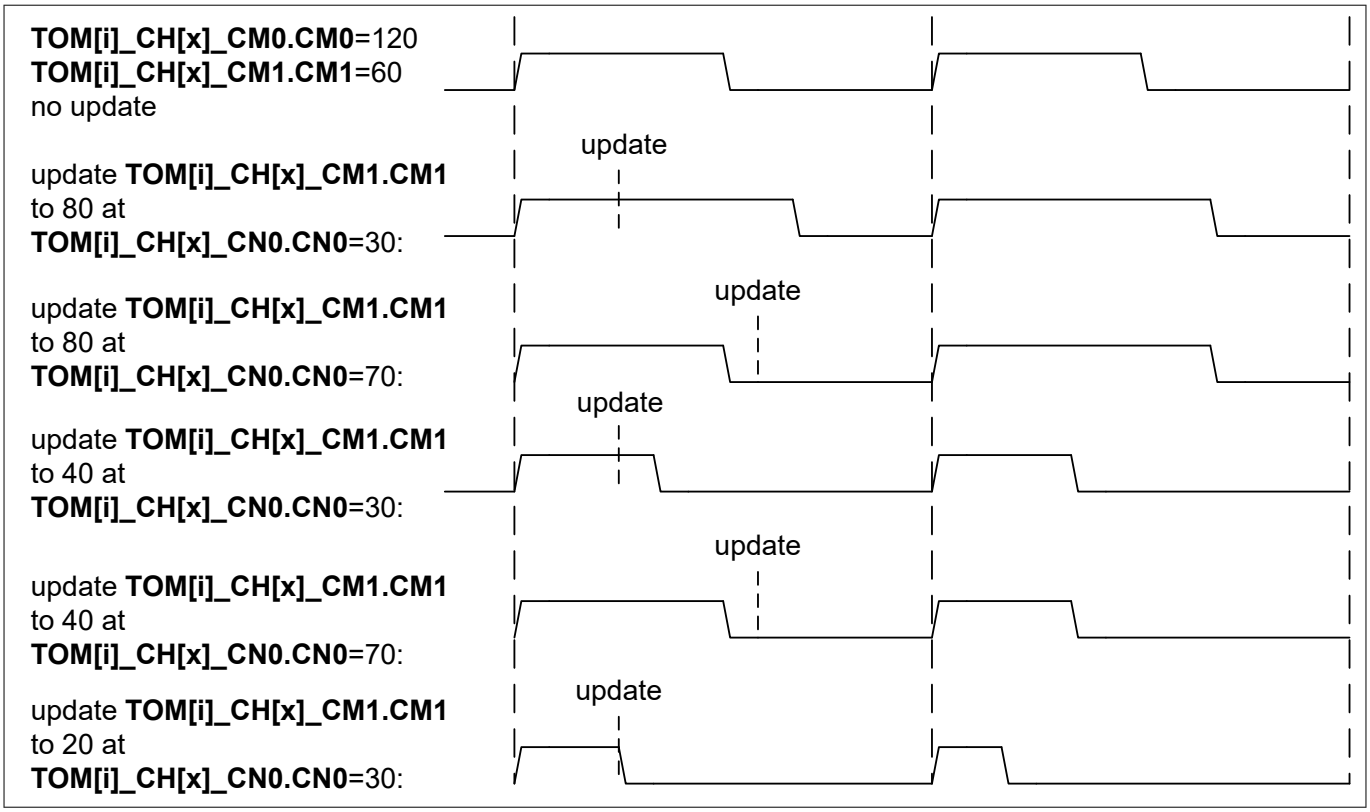


Figure 1063 Asynchronous update of pulse width only

44.3.1.7.4 Continuous counting up mode

In continuous counting up mode, the **TOM** channel starts incrementing the counter CLSi_TOM_CHx_CN0.CN0 once it is enabled by setting the corresponding bits in register CLSi_TOM_TGCg_ENDIS_STAT (refer to TGC sub-unit for details of enabling a **TOM** channel). In this mode with CLSi_TOM_CHx_CTRL.UDMODE=00_B (i.e. CLSi_TOM_CHx_CN0.CN0 counts only up), depending on configuration bits CLSi_TOM_CHx_CTRL.RST_CCU0 the counter register CLSi_TOM_CHx_CN0.CN0 can be reset either when the counter value is equal to the compare value CLSi_TOM_CHx_CM0.CM0 (i.e. CLSi_TOM_CHx_CN0.CN0 counts only from 0 to CLSi_TOM_CHx_CM0.CM0-1 and is then reset to 0) or when the trigger signal TOM_CH_TRIGOUT[x-1:x-1] of the preceding channel x-1 (which can also be the last channel of preceding instance **TOM[i-1]**) is captured by the **TOM[i]** or when there is trigger signal TOM_EXT_TRIGIN[x:x] from **TIM[i]** after it was synchronized to the selected clock resolution signal.

In this case, if CLSi_TOM_TGCg_GLB_CTRL.UPEN_CTRLc (c=0-7)=10_B, also the operation register CLSi_TOM_CHx_CM0.CM0, CLSi_TOM_CHx_CM1.CM1, CLSi_TOM_CHx_CTRL.SL and CLSi_TOM_CHx_CTRL.CLK_SRC are updated from their shadow registers together with CLSi_TOM_CHx_CN0.CN0 reset.

The duration of the pulse high or low time and period is measured with the counter in sub-unit CCU0. The trigger of the counter is one of the eight fixed **CMU** clock signals configurable in the channel control register CLSi_TOM_CHx_CTRL.CLK_SRC. The register CLSi_TOM_CHx_CM0.CM0 holds the duration of the period and the register CLSi_TOM_CHx_CM1.CM1 holds the duration of the pulse width in clock ticks of the selected **CMU** clock SEL_FXCMU_CLKEN. The signal level of the generated output signal can be configured with the configuration bit CLSi_TOM_CHx_CTRL.SL.

If counter CLSi_TOM_CHx_CN0.CN0 of channel x is reset by its own CCU0 unit (i.e. the compare match of CLSi_TOM_CHx_CN0.CN0>=CLSi_TOM_CHx_CM0.CM0-1 configured by CLSi_TOM_CHx_CTRL.RST_CCU0=0), the PWM output behavior is shown in Figure 1064 and the following statements are valid:

- After **TOM** i channel x is enabled, CLSi_TOM_CHx_CN0.CN0 counts firstly from its initial value to CLSi_TOM_CHx_CM0.CM0-1 and is then reset to 0. This phase is defined as initial delay that can be flexibly configured as by the value of (CLSi_TOM_CHx_CM0.CM0-CLSi_TOM_CHx_CN0.CN0) multiplied with the selected fixed **CMU** clock period.
- When CLSi_TOM_CHx_CN0.CN0 is reset from CLSi_TOM_CHx_CM0.CM0-1 to 0, an edge to CLSi_TOM_CHx_CTRL.SL is generated.
- When CLSi_TOM_CHx_CN0.CN0 is incrementing and reaches CLSi_TOM_CHx_CM1.CM1 (CLSi_TOM_CHx_CN0.CN0>=CLSi_TOM_CHx_CM1.CM1), an edge to !CLSi_TOM_CHx_CTRL.SL is generated.
- CLSi_TOM_CHx_CN0.CN0 always counts from 0 to CLSi_TOM_CHx_CM0.CM0-1 and is then reset to 0.
- If CLSi_TOM_CHx_CM0.CM0=0 or CLSi_TOM_CHx_CM0.CM0=1, the counter CLSi_TOM_CHx_CN0.CN0 is constant 0. It is highly recommended to configure CLSi_TOM_CHx_CM0.CM0>1.
- If CLSi_TOM_CHx_CM1.CM1=0 and CLSi_TOM_CHx_CM0.CM0>1, the output is !CLSi_TOM_CHx_CTRL.SL (i.e. 0% duty cycle PWM signal).
- If CLSi_TOM_CHx_CM1.CM1 >= CLSi_TOM_CHx_CM0.CM0 and CLSi_TOM_CHx_CM0.CM0>1, the output is CLSi_TOM_CHx_CTRL.SL (i.e. 100% duty cycle PWM signal).

Since the TOM_OUT[x:x] signal level is defined as !CLSi_TOM_CHx_CTRL.SL when the **TOM** channel is disabled without setting freeze mode, the first PWM period can be shifted earlier by writing an initial offset value to CLSi_TOM_CHx_CN0.CN0 register. By doing this, the **TOM** channel first counts until CLSi_TOM_CHx_CN0.CN0 reaches CLSi_TOM_CHx_CM0.CM0 and then it toggles the output signal at TOM_OUT[x:x].

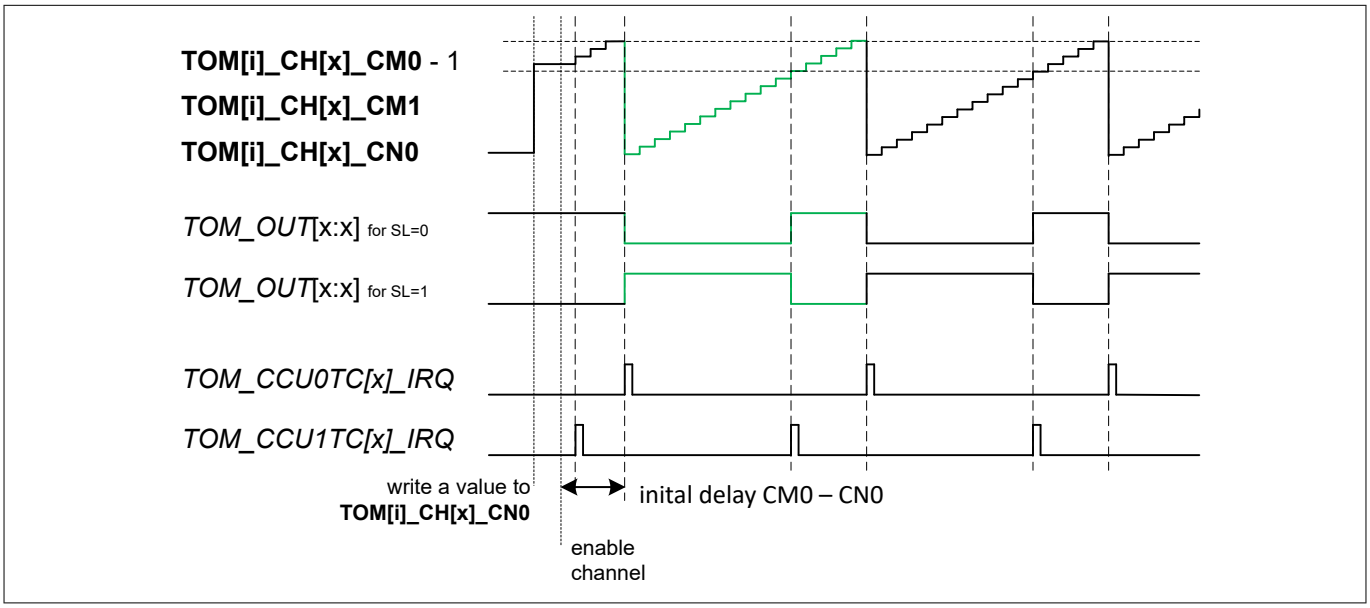


Figure 1064 PWM **output behavior with respect to the** CLSi_TOM_CHx_CTRL.SL **bit in continuous counting up mode if** CLSi_TOM_CHx_CTRL.RST_CCU0**=0**

In this case, as shown in Figure 1064, 2 interrupt signals TOM_CCU0TC[x]_IRQ and TOM_CCU1TC[x]_IRQ of channel x can be generated together with output generation TOM_OUT[x:x] and both of them are shown on TOM_IRQ[x:x] if CLSi_TOM_CHx_IRQ_EN=11_B. When CLSi_TOM_CHx_CN0.CN0 is reset from CLSi_TOM_CHx_CM0.CM0 -1 to 0, TOM_CCU0TC[x]_IRQ of channel x is generated. When CLSi_TOM_CHx_CN0.CN0 is incrementing to reach CLSi_TOM_CHx_CM1.CM1, TOM_CCU1TC[x]_IRQ of channel x is generated. Especially, if the initial configuration before enabling the channel fulfills the condition CLSi_TOM_CHx_CN0.CN0 >= CLSi_TOM_CHx_CM1.CM1 - 1, an interrupt is generated after 1 fixed **CMU** clock period on TOM_CCU1TC[x]_IRQ. Similarly but not shown in the figure, if the initial configuration before enabling the channel fulfills the condition CLSi_TOM_CHx_CN0.CN0 >= CLSi_TOM_CHx_CM0.CM0 - 1, an interrupt is generated after 1 selected **CMU** clock period on TOM_CCU0TC[x]_IRQ. The interrupt signals are always aligned with the edges of the output TOM_OUT[x:x].

If the counter register CLSi_TOM_CHx_CN0.CN0 of channel x is reset by the trigger signal coming from another channel or the assigned **TIM** module (configured by CLSi_TOM_CHx_CTRL.RST_CCU0=1), the PWM output behavior is shown in Figure 1065 and the following statements are valid (with MAX=number of selected CCM[i]_FXCLK_RES cycle between two trigger signals):

- CLSi_TOM_CHx_CN0.CN0 counts from 0 to MAX-1 and is then reset to 0 with the falling edge of TOM_CH_TRIGIN[x:x] or falling edge of TOM_EXT_TRIGIN_S that is synchronized signal TOM_EXT_TRIGIN[x:x] (see External trigger interface).
- If CLSi_TOM_CHx_CN0.CN0 reaches CLSi_TOM_CHx_CM1.CM1, an edge to !CLSi_TOM_CHx_CTRL.SL is generated.
- If CLSi_TOM_CHx_CN0.CN0 reaches CLSi_TOM_CHx_CM0.CM0, an edge to CLSi_TOM_CHx_CTRL.SL is generated.
- As soon as CLSi_TOM_CHx_CN0.CN0 reaches the value of CLSi_TOM_CHx_CM0.CM0 while CLSi_TOM_CHx_CM1.CM1 is equal to CLSi_TOM_CHx_CM0.CM0, an edge to CLSi_TOM_CHx_CTRL.SL is generated at the output or the output remains at CLSi_TOM_CHx_CTRL.SL level depending on the former level of the output (CLSi_TOM_CHx_CM0.CM0 has higher priority). Please note that this configuration is not suitable for generating 100% duty cycle.
- If CLSi_TOM_CHx_CM0.CM0=0 and CLSi_TOM_CHx_CM1.CM1>MAX, the output is a pulse of CLSi_TOM_CHx_CTRL.SL for the period MAX (i.e. 100% duty cycle PWM signal).
- If CLSi_TOM_CHx_CM0.CM0 > MAX and CLSi_TOM_CHx_CM1.CM1=0, the output is ! CLSi_TOM_CHx_CTRL.SL (i.e. 0% duty cycle PWM signal).
- If CLSi_TOM_CHx_CM0.CM0 is set to MAX, an edge to CLSi_TOM_CHx_CTRL.SL is generated at the output at the very end of the current period or the output remains at the CLSi_TOM_CHx_CTRL.SL level, depending on the previous level of the output. Be aware that this applies even if the value of CLSi_TOM_CHx_CM1.CM1 is updated to 0 for the new period (CLSi_TOM_CHx_CM0.CM0 has higher priority).

In this case, as shown in Figure 1065, 2 interrupt signals TOM_CCU0TC[x]_IRQ and TOM_CCU1TC[x]_IRQ of channel x can be generated together with output generation TOM_OUT[x:x] and both of them can be shown on TOM_IRQ[x:x] if CLSi_TOM_CHx_IRQ_EN=11_B. TOM_CCU0TC[x]_IRQ of channel x is generated when CLSi_TOM_CHx_CN0.CN0 reaches CLSi_TOM_CHx_CM0.CM0 while TOM_CCU1TC[x]_IRQ of channel x is generated when CLSi_TOM_CHx_CN0.CN0 reaches CLSi_TOM_CHx_CM1.CM1.

The hardware ensures that for both 0% and 100% duty cycle no glitch occurs at the output of the **TOM** channel.

In case of CLSi_TOM_TGCg_GLB_CTRL.UPEN_CTRLc (c=0-7)=10_B, when the counter value CLSi_TOM_CHx_CN0.CN0 reaches the compare value in register CLSi_TOM_CHx_CM0.CM0 (in fact CLSi_TOM_CHx_CM0.CM0-1) if CLSi_TOM_CHx_CTRL.RST_CCU0=0 or the channel captures the falling edge of the trigger signal if CLSi_TOM_CHx_CTRL.RST_CCU0=1, the operation registers CLSi_TOM_CHx_CM0.CM0, CLSi_TOM_CHx_CM1.CM1, CLSi_TOM_CHx_CTRL.SL and CLSi_TOM_CHx_CTRL.CLK_SRC are updated with the content of their shadow registers CLSi_TOM_CHx_SR0.SR0, CLSi_TOM_CHx_SR1.SR1, CLSi_TOM_CHx_CTRL_SR.SL_SR and CLSi_TOM_CHx_CTRL_SR.CLK_SRC_SR.

The operation registers CLSi_TOM_CHx_CM0.CM0, CLSi_TOM_CHx_CM1.CM1, CLSi_TOM_CHx_CTRL.SL and CLSi_TOM_CHx_CTRL.CLK_SRC can also be asynchronously updated from their shadow registers with the force update mechanism. Please refer to Pulse width, period, signal level and clock frequency update mechanisms for the force update configuration steps and more details. In continuous counting up mode, the force update mechanism can be also used to stop the current running PWM signal generation and restart a new PWM period with new PWM parameters by additionally setting CLSi_TOM_TGCg_FUPD_CTRL.RSTCN0_CHc (c=0-7) = 10_B together with CLSi_TOM_TGCg_FUPD_CTRL.FUPD_CTRLc (c=0-7) = 10_B. That means the force update event will not only update the operation registers from their shadow registers, but also simultaneously reset the counter to restart the next new PWM period with setting the output as CLSi_TOM_CHx_CTRL.SL if CLSi_TOM_CHx_CTRL.RST_CCU0=0 or without changing the output if CLSi_TOM_CHx_CTRL.RST_CCU0=1.

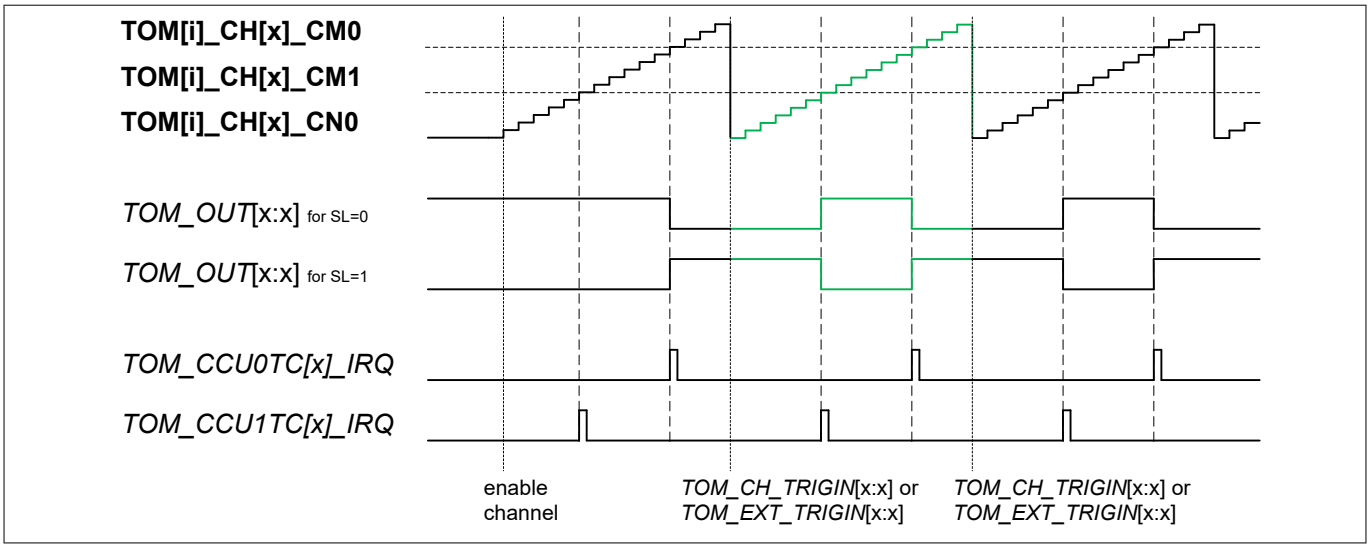


Figure 1065 PWM **output behavior with respect to the** CLSi_TOM_CHx_CTRL.SL **bit in continuous counting up mode if** CLSi_TOM_CHx_CTRL.RST_CCU0**=1**

44.3.1.7.5 Continuous counting up-down mode

In continuous counting up-down mode, if CLSi_TOM_CHx_CN0.CN0 counts up and down (CLSi_TOM_CHx_CTRL.UDMODE != 00_B). Depending on configuration bits CLSi_TOM_CHx_CTRL.RST_CCU0, the counter CLSi_TOM_CHx_CN0.CN0 changes the direction either when the counter value is equal to the compare value CLSi_TOM_CHx_CM0.CM0-1, or when has counted down to 0 or when triggered by the internal trigger signal TOM_CH_TRIGIN[x:x] (i.e. TOM_CH_TRIGOUT[x-1:x-1] from the preceding channel [x-1]) or the external trigger signal TOM_EXT_TRIGIN[x:x] from **TIM** module [i].

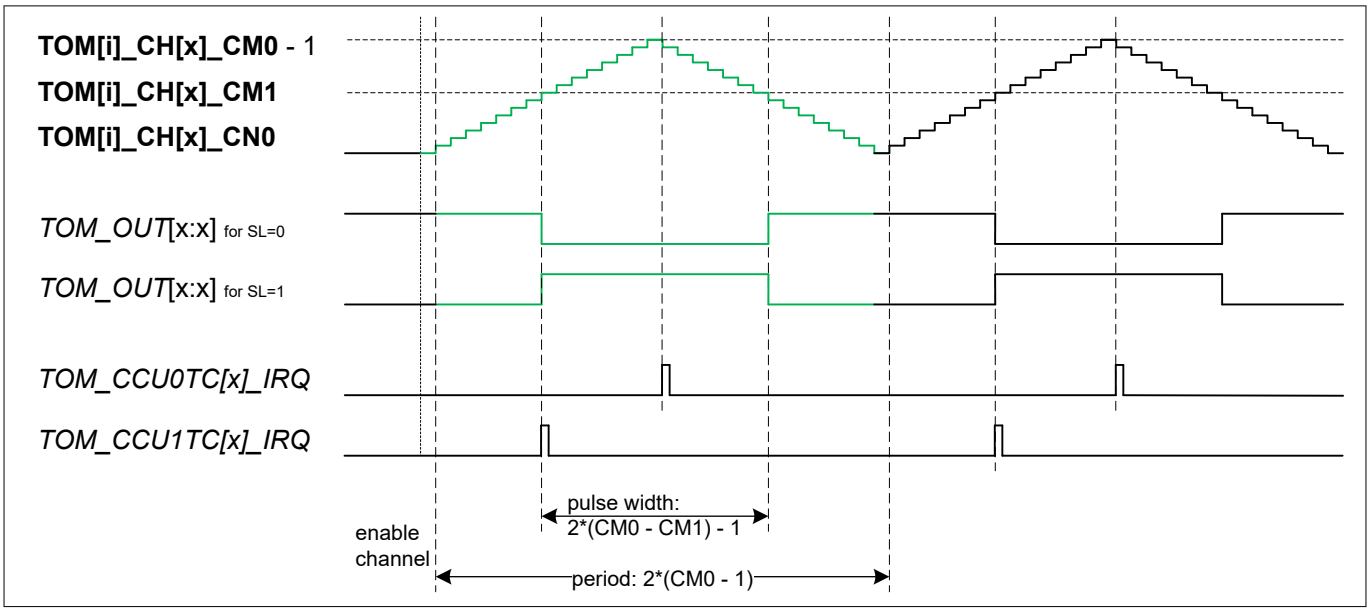


Figure 1066 PWM **output behavior with respect to the** CLSi_TOM_CHx_CTRL.SL **bit in continuous counting up-down mode if** CLSi_TOM_CHx_CTRL.RST_CCU0**=0**

The clock of the counter CLSi_TOM_CHx_CN0.CN0 can be one of the **CMU** clocks CCM[i]_FXCLK_RES.

The clock for CLSi_TOM_CHx_CN0.CN0 is defined by CLSi_TOM_CHx_CTRL.CLK_SRC value in register CLSi_TOM_CHx_CTRL.

If CLSi_TOM_CHx_CTRL.RST_CCU0=0, the counter register CLSi_TOM_CHx_CN0.CN0 of channel x will change its counting direction by the compare match of CLSi_TOM_CHx_CN0.CN0>=CLSi_TOM_CHx_CM0.CM0-1 in its own CCU0 unit and the following statements are valid:

- In each PWM period, CLSi_TOM_CHx_CN0.CN0 firstly counts continuously up from 1 to CLSi_TOM_CHx_CM0.CM0-1 and then down to 0. So period = 2*(CLSi_TOM_CHx_CM0.CM0-1).
- If CLSi_TOM_CHx_CN0.CN0 >= CLSi_TOM_CHx_CM1.CM1, the output is TOM_OUT[x:x] set to CLSi_TOM_CHx_CTRL.SL.
- If CLSi_TOM_CHx_CN0.CN0 < CLSi_TOM_CHx_CM1.CM1, the output is TOM_OUT[x:x] set to ! CLSi_TOM_CHx_CTRL.SL. So pulse width = 2*(CLSi_TOM_CHx_CM0.CM0-CLSi_TOM_CHx_CM1.CM1)-1.
- If CLSi_TOM_CHx_CM0.CM0 <= 1, the counter behavior of CLSi_TOM_CHx_CN0.CN0 is unexpected (not recommended).
- If CLSi_TOM_CHx_CM1.CM1 = 0 and also CLSi_TOM_CHx_CM0.CM0 > 1, the output TOM_OUT[x:x] is CLSi_TOM_CHx_CTRL.SL (i.e. 100% duty cycle).
- If CLSi_TOM_CHx_CM1.CM1 >= CLSi_TOM_CHx_CM0.CM0 and also CLSi_TOM_CHx_CM0.CM0 > 1, the output TOM_OUT[x:x] is !CLSi_TOM_CHx_CTRL.SL (i.e. 0% duty cycle).

This behavior is depicted in Figure 1066. In this case, if CLSi_TOM_TGCg_GLB_CTRL.UPEN_CTRLc (c=0-7) **(c=x)** =1, the operation registers CLSi_TOM_CHx_CM0.CM0, CLSi_TOM_CHx_CM1.CM1, CLSi_TOM_CHx_CTRL.SL and CLSi_TOM_CHx_CTRL.CLK_SRC can be updated when CLSi_TOM_CHx_CN0.CN0 = 0 (i.e. counting direction changes from down to up) or CLSi_TOM_CHx_CN0.CN0 reaches CLSi_TOM_CHx_CM0.CM0-1 (i.e. counting direction changes from up to down) depending on various configurations of CLSi_TOM_CHx_CTRL.UDMODE (see the register description for more details). It is particularly not recommended to change the value of CLSi_TOM_CHx_CM0.CM0, CLSi_TOM_CHx_CTRL.SL and CLSi_TOM_CHx_CTRL.CLK_SRC when CLSi_TOM_CHx_CN0.CN0 reaches CLSi_TOM_CHx_CM0.CM0-1. Otherwise, the behavior may be unexpected.

Like continuous counting up mode, interrupt signals TOM_CCU0TC[x]_IRQ and TOM_CCU1TC[x]_IRQ of channel x are generated together with PWM signal output edges. When CLSi_TOM_CHx_CN0.CN0 reaches CLSi_TOM_CHx_CM1.CM1 at the first time and the first edge is generated, an interrupt on TOM_CCU1TC[x]_IRQ is generated. When CLSi_TOM_CHx_CN0.CN0 changes its counting direction from up to down, an interrupt on TOM_CCU0TC[x]_IRQ is generated.

If CLSi_TOM_CHx_CTRL.RST_CCU0=1, the counter register CLSi_TOM_CHx_CN0.CN0 of channel x will change its counting direction by the trigger signal coming from another channel or the assigned **TIM** module and the following statements are valid:

- CLSi_TOM_CHx_CN0.CN0 firstly counts continuously up. At the falling edge of the trigger signal pulse TOM_CH_TRIGIN[x:x] or TOM_EXT_TRIGIN_S (the synchronized signal TOM_EXT_TRIGIN[x:x], see External trigger interface), the counter switches to count down mode. If CLSi_TOM_CHx_CN0.CN0 reaches 0, it will count up again.
- If CLSi_TOM_CHx_CN0.CN0 >= CLSi_TOM_CHx_CM1.CM1, the output TOM_OUT[x:x] is set to CLSi_TOM_CHx_CTRL.SL.
- If CLSi_TOM_CHx_CN0.CN0 < CLSi_TOM_CHx_CM1.CM1, the output is TOM_OUT[x:x] set to ! CLSi_TOM_CHx_CTRL.SL.
- If CLSi_TOM_CHx_CM1.CM1 = 0, the output TOM_OUT[x:x] is CLSi_TOM_CHx_CTRL.SL (i.e. 100% duty cycle).
- If CLSi_TOM_CHx_CN0.CN0 >= CLSi_TOM_CHx_CM0.CM0, the output TOM_OUT_T[x:x] is set to CLSi_TOM_CHx_CTRL.SL.
- If CLSi_TOM_CHx_CN0.CN0 < CLSi_TOM_CHx_CM0.CM0, the output is TOM_OUT_T[x:x] set to ! CLSi_TOM_CHx_CTRL.SL.
- If CLSi_TOM_CHx_CM0.CM0 = 0, the output TOM_OUT_T[x:x] is CLSi_TOM_CHx_CTRL.SL (i.e. 100% duty cycle).

This behavior is depicted in Figure 1067. In this case, if CLSi_TOM_TGCg_GLB_CTRL.UPEN_CTRLc (c=0-7) **(c=x)**

=1, the operation registers CLSi_TOM_CHx_CM0.CM0, CLSi_TOM_CHx_CM1.CM1, CLSi_TOM_CHx_CTRL.SL and CLSi_TOM_CHx_CTRL.CLK_SRC can also be updated when CLSi_TOM_CHx_CN0.CN0 = 0 (i.e. counting direction changes from down to up) or CLSi_TOM_CHx_CN0.CN0 reaches its maximum value (i.e. counting direction changes from up to down) depending on various configurations of CLSi_TOM_CHx_CTRL.UDMODE (see the register description for more details). Especially, it is not recommended to change the value of CLSi_TOM_CHx_CTRL.SL and CLSi_TOM_CHx_CTRL.CLK_SRC when CLSi_TOM_CHx_CN0.CN0 its maximum value (i.e. counting direction changes from up to down). Otherwise, the behavior may be unexpected.

If CLSi_TOM_CHx_IRQ_EN=11_B, interrupt signals TOM_CCU0TC[x]_IRQ and TOM_CCU1TC[x]_IRQ of channel x are generated together with PWM signal output. When CLSi_TOM_CHx_CN0.CN0 reaches CLSi_TOM_CHx_CM1.CM1 at the first time and the first edge of TOM_OUT[x:x] is generated, an interrupt on TOM_CCU1TC[x]_IRQ is generated. When CLSi_TOM_CHx_CN0.CN0 reaches CLSi_TOM_CHx_CM0.CM0 at the first time and the first edge of TOM_OUT_T[x:x] is generated, an interrupt on TOM_CCU0TC[x]_IRQ is generated.

In case of continuous counting up-down mode and CLSi_TOM_CHx_CTRL.RST_CCU0=1, it is recommended to configure the **TOM** channel according to the following statements:

- The triggering channel and the triggered channel are both running in up-down mode.
- The time between two trigger signals is equal to the time needed for CLSi_TOM_CHx_CN0.CN0 of triggered channel to count back to 0 and again up to the same upper value.

The second recommendation can be fulfilled by synchronizing the start of the triggering channel and the triggered channel, i.e. let both channels start with CLSi_TOM_CHx_CN0.CN0 = 0 and same clock source.

If there is a pipeline register in the trigger chain, the additional delay of one clock period has to be taken into account by starting the triggering channel with CLSi_TOM_CHx_CN0.CN0 = 1 (i.e. 1 greater than CLSi_TOM_CHx_CN0.CN0 of the triggered channel).

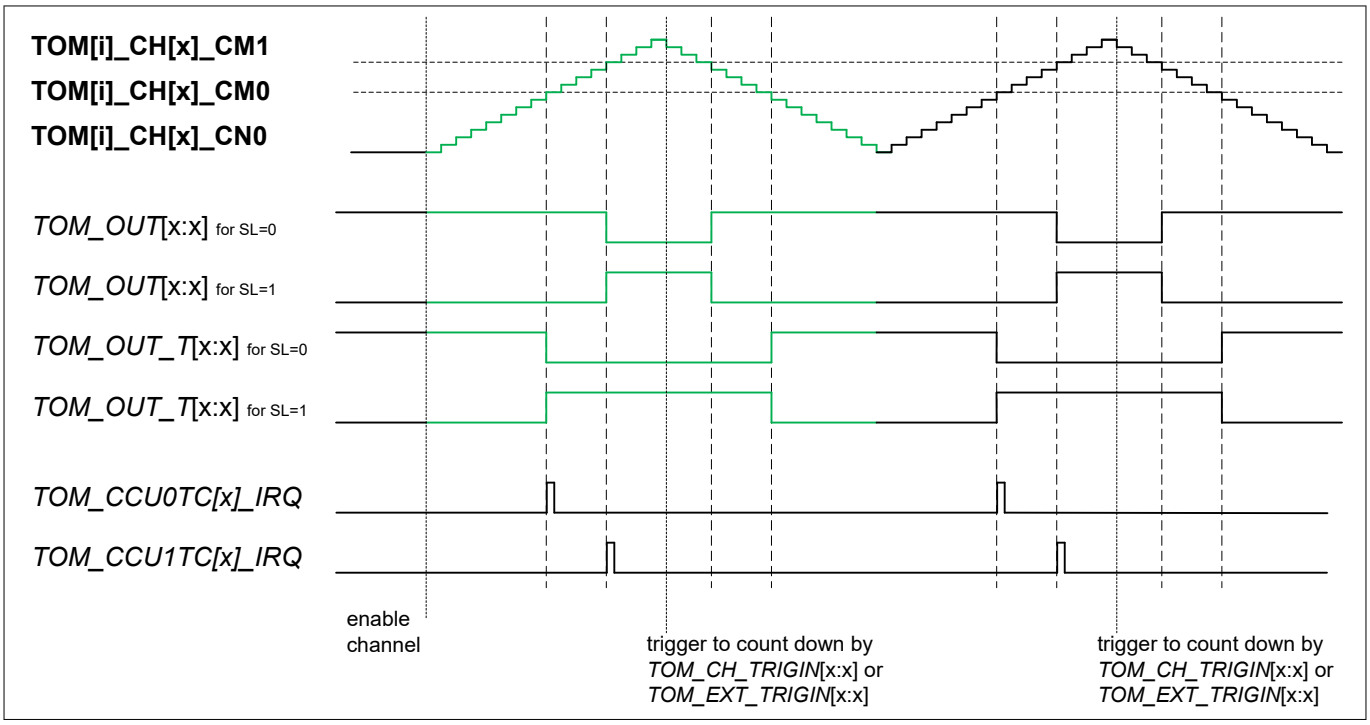


Figure 1067 PWM output behavior with respect to the CLSi_TOM_CHx_CTRL.SL bit in continuous counting up-down mode if CLSi_TOM_CHx_CTRL.RST_CCU0=1

44.3.1.7.6 One-shot counting up mode

The **TOM** channel can operate in one-shot mode when the CLSi_TOM_CHx_CTRL.OSM bit is set. One-shot mode means that a single pulse with the pulse level defined in bit CLSi_TOM_CHx_CTRL.SL is generated on the output signal.

Firstly, the channel has to be enabled by setting the corresponding CLSi_TOM_TGCg_ENDIS_STAT.ENDIS_STATc (c=0-7).

Unlike in the continuous counting modes, the counter CLSi_TOM_CHx_CN0.CN0 will not be incremented once the channel is enabled. Instead, according to the configuration bit CLSi_TOM_CHx_CTRL.OSM_TRIG, the start of counting and pulse generation is triggered by a write-access of CLSi_TOM_CHx_CN0.CN0 or the internal trigger signal TOM_CH_TRIGIN[x:x] or the external trigger signal TOM_EXT_TRIGIN[x:x].

If **SPE** mode of **TOM[i]** channel x (x∈{2, 6, 7, 8, 9}) is enabled (set bit CLSi_TOM_CHx_CTRL.SPEM (x∈{2, 6, 7, 8, 9})), also the trigger signal SPE[i]_NIPD can trigger the reset of register CLSi_TOM_CHx_CN0.CN0 to zero and a start of the pulse generation.

When CLSi_TOM_CHx_CTRL.OSM_TRIG=0, a write-access to the register CLSi_TOM_CHx_CN0.CN0 triggers the start of incrementing the counter register CLSi_TOM_CHx_CN0.CN0 and pulse generation.

If the time between the write-access and the first edge generation of one-shot PWM signal is defined as initial delay, the initial delay is determined by the value of (CLSi_TOM_CHx_CM0.CM0-CLSi_TOM_CHx_CN0.CN0) multiplied with the selected **CMU** clock period.

Especially, if writing a value of greater than or equal to CLSi_TOM_CHx_CM0.CM0-1 to CLSi_TOM_CHx_CN0.CN0 and configuring the initial clock source as cluster clock (the clock source at reset state), the initial delay is minimum, i.e. 1 cluster clock period CLS[j]_CLK. That means after the write-access of CLSi_TOM_CHx_CN0.CN0, the first edge of the output with the interrupt will be set in 1 CLS[j]_CLK period.

If the counter CLSi_TOM_CHx_CN0.CN0 is reset from CLSi_TOM_CHx_CM0.CM0-1 back to zero, the first edge at TOM_OUT[x:x] is generated. And the operation phase of the pulse generation starts incrementing CLSi_TOM_CHx_CN0.CN0 from 0 to CLSi_TOM_CHx_CM0.CM0-1 again.

The second edge in the operation phase is generated if CLSi_TOM_CHx_CN0.CN0 increments until it reaches CLSi_TOM_CHx_CM1.CM1.

If the counter CLSi_TOM_CHx_CN0.CN0 reaches the value of CLSi_TOM_CHx_CM0.CM0-1 for the second time, the counter will stop.

Figure 1068 depicts the pulse generation process mentioned above. The generation of the interrupt signals during the generation of the PWM signal is similar as described in continuous counting up mode. Furthermore, if the value written to CLSi_TOM_CHx_CN0.CN0 for triggering or retriggering the counter is greater than or equal to CLSi_TOM_CHx_CM1.CM1-1, an interrupt on TOM_CCU1TC[x]_IRQ is generated after 1 selected **CMU** clock period. Similarly but not shown in the figure, if the value written to CLSi_TOM_CHx_CN0.CN0 for triggering or retriggering the counter is greater than or equal to CLSi_TOM_CHx_CM0.CM0 - 1, an interrupt on TOM_CCU0TC[x]_IRQ is generated after 1 selected **CMU** clock period. Afterwards when CLSi_TOM_CHx_CN0.CN0 is reset from CLSi_TOM_CHx_CM0.CM0 -1 to 0, TOM_CCU0TC[x]_IRQ of channel x is generated. When CLSi_TOM_CHx_CN0.CN0 is incrementing to reach CLSi_TOM_CHx_CM1.CM1, TOM_CCU1TC[x]_IRQ of channel x is generated.

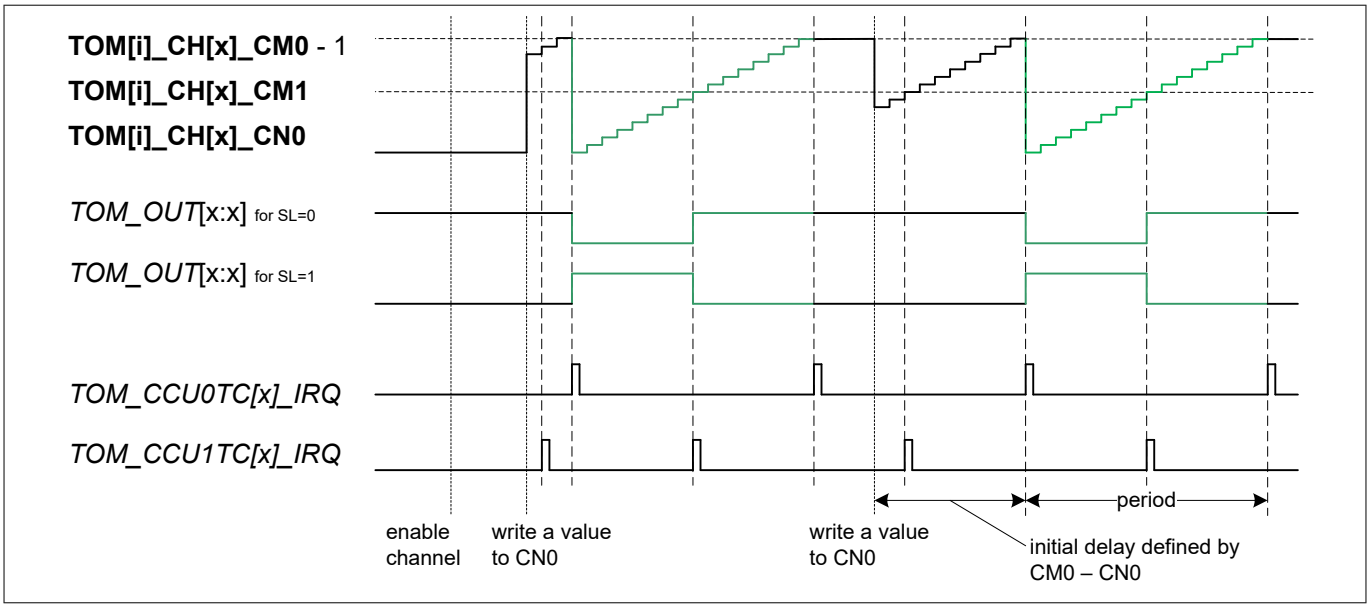


Figure 1068 PWM output with respect to configuration bit CLSi_TOM_CHx_CTRL.SL in one-shot counting up mode if TOM[i]_CH[x]_CRTL.OSM_TRIG=0

Retriggering an one-shot cycle (while CLSi_TOM_CHx_CN0.CN0 is already incrementing) by writing a value to CLSi_TOM_CHx_CN0.CN0 is possible but depends on the phase of CLSi_TOM_CHx_CN0.CN0:

- Phase 1: update of CLSi_TOM_CHx_CN0.CN0 before CLSi_TOM_CHx_CN0.CN0 reaches CLSi_TOM_CHx_CM0.CM0 for the first time (initial phase)
- Phase 2: update of CLSi_TOM_CHx_CN0.CN0 after CLSi_TOM_CHx_CN0.CN0 has reached CLSi_TOM_CHx_CM0.CM0 for the first time until the current single PWM pulse is finished (operation phase)

In phase 1 (initial phase): writing a value CN0(new) to counter CLSi_TOM_CHx_CN0.CN0 (CN0(new)< CLSi_TOM_CHx_CM0.CM0) leads to a shift of the first edge by the time CLSi_TOM_CHx_CM0.CM0-CN0(new), i.e. the first edge is generated when CLSi_TOM_CHx_CN0.CN0 is reset from CLSi_TOM_CHx_CM0.CM0-1 back to 0 for the first time.

In phase 2 (operation phase): writing to counter CLSi_TOM_CHx_CN0.CN0 is not possible and protected by the hardware.

A failed write-access to CLSi_TOM_CHx_CN0.CN0 in phase 2 (operation phase) is indicated by the status register flag CLSi_TOM_CHx_STAT.OSM_RTF.

After one PWM pulse is ended, other single pulses can be further triggered by a write-access to register CLSi_TOM_CHx_CN0.CN0.

It is recommended to enable the update mechanism by setting of CLSi_TOM_TGCg_GLB_CTRL.UPEN_CTRLc (c=0-7) = 10_B for retriggering the second one-shot PWM signal generation with different initial delay and different PWM parameters. Similar to the continuous counting up mode with CLSi_TOM_CHx_CTRL.RST_CCU0=0, the operation registers CLSi_TOM_CHx_CM0.CM0/ CLSi_TOM_CHx_CM1.CM1/CLSi_TOM_CHx_CTRL.SL/CLSi_TOM_CHx_CTRL.CLK_SRC are always synchronously updated when CLSi_TOM_CHx_CN0.CN0 reaches CLSi_TOM_CHx_CM0.CM0-1. Firstly, after the first one-shot PWM signal is generated, the new values for initial delay (phase 1) can be written into the shadow registers CLSi_TOM_CHx_SR0.SR0/CLSi_TOM_CHx_SR1.SR1/CLSi_TOM_CHx_CTRL_SR.SL_SR/ CLSi_TOM_CHx_CTRL_SR.CLK_SRC_SR before the retriggering is executed by writing of register CLSi_TOM_CHx_CN0.CN0. The shadow registers should be immediately loaded to the operation registers CLSi_TOM_CHx_CM0.CM0/CLSi_TOM_CHx_CM1.CM1/CLSi_TOM_CHx_CTRL.SL/CLSi_TOM_CHx_CTRL.CLK_SRC, because CLSi_TOM_CHx_CN0.CN0 keeps as CLSi_TOM_CHx_CM0.CM0-1. When CLSi_TOM_CHx_CN0.CN0 is written to be retriggered afterwards, the initial delay is defined according to the updated operation registers. Secondly, during the initial delay phase (phase 1), the new parameters for PWM signal can be written into the shadow registers again. The operational registers will be updated with the shadow registers when

CLSi_TOM_CHx_CN0.CN0 is reset from CLSi_TOM_CHx_CM0.CM0-1 at the end of phase 1 and phase 2 is started to generate the second one-shot PWM signal with the new updated operation registers.

If a channel is configured to one-shot mode and configuration bit CLSi_TOM_CHx_CTRL.OSM_TRIG is set to 1, the trigger signal only triggers start of one pulse generation.

If a channel is configured to one-shot mode and configuration bit CLSi_TOM_CHx_CTRL.OSM_TRIG is set to 1, the trigger signal TOM_CH_TRIGIN[x:x] or TOM_EXT_TRIGIN[x:x] triggers start of pulse generation as shown in Figure 1069. In this case, it is not allowed to write CLSi_TOM_CHx_CN0.CN0 and so the register CLSi_TOM_CHx_STAT.OSM_RTF is not relevant. Otherwise, a write-access to CLSi_TOM_CHx_CN0.CN0 can make an unpredictable output behavior. Furthermore, as shown in Figure 1069, CLSi_TOM_CHx_CN0.CN0 should be initially configured as CLSi_TOM_CHx_CM0.CM0-1 or greater value before enabling the channel. Otherwise, it is hard to define the counting behavior and the initial delay behavior. As shown in Figure 1069, when the channel is enabled, after the falling edge of the trigger pulse TOM_CH_TRIGIN[x:x] or TOM_EXT_TRIGIN_S (the synchronized signal TOM_EXT_TRIGIN[x:x], see External trigger interface), CLSi_TOM_CHx_CN0.CN0 will be triggered to be reset that is synchronous with the next enable high pulse of the selected **CMU** clock source (i.e. synchronous to the next falling edge of the selected **CMU** enable signal). Then it counts from 0 to CLSi_TOM_CHx_CM0.CM0-1. This process is defined as initial phase. If the internal trigger signal TOM_TRIGIN[x:x] is in use and it is synchronous to the selected **CMU** clock source or the external trigger signal TOM_EXT_TRIGIN[x:x] is in use and it must be synchronized to the selected **CMU** clock source before triggering, the initial delay can be defined as the time between the falling edge of the synchronized trigger signal to the point of time of starting the generation of the PWM signal (green phase in the figure), i.e. CLSi_TOM_CHx_CM0.CM0+1 selected **CMU** clock periods. After that, CLSi_TOM_CHx_CN0.CN0 will be reset and counts from 0 to CLSi_TOM_CHx_CM0.CM0-1 at the second time to generate one single PWM signal and then stops. This second counting is defined as operation phase.

After the first OSM signal is generated, the second trigger signal is allowed to be applied to start the counter and generate the next OSM signal after a certain initial delay. Figure 1069 shows such an application of generating 2 OSM signals without changing PWM parameters. In the case, where the second trigger signal is generated, the counter exhibits the same behavior as described above for the first OSM signal generation. That means the initial delay is also CLSi_TOM_CHx_CM0.CM0+1 selected **CMU** clock periods and the OSM signal is the same as the first one.

However, in some real applications of generating various single PWM signals, it is necessary to synchronously update the parameters in CLSi_TOM_CHx_CM0.CM0/CLSi_TOM_CHx_CM1.CM1/CLSi_TOM_CHx_CTRL.SL/CLSi_TOM_CHx_CTRL.CLK_SRC by setting CLSi_TOM_TGCg_GLB_CTRL.UPEN_CTRLc (c=0-7) **(c=x)** = 10_B. In this case, these registers are updated from their shadow registers when CLSi_TOM_CHx_CN0.CN0 is matching CLSi_TOM_CHx_CM0.CM0-1. After the first OSM PWM signal, if CLSi_TOM_CHx_CM0.CM0 is updated with a new value CM0(new) before the second trigger signal comes, the initial delay of the second OSM signal depends on CM0(new) and the value CN0(old) of CLSi_TOM_CHx_CN0.CN0 when the counter stopped at the end of the first OSM signal. The value CN0(old) is actually the value of CLSi_TOM_CHx_CM0.CM0-1 of the first OSM signal before the synchronous update, i.e. CN0(old)=CM0(old)-1. Like the first OSM signal generation, with the falling edge of the second trigger pulse signal, the counter will be retriggered synchronous to the next enable high pulse of the selected **CMU** clock source (i.e. synchronous to the next falling edge of the selected **CMU** enable signal). Assuming the internal trigger signal TOM_TRIGIN[x:x] is in use and it is synchronous to the selected **CMU** clock source or the external trigger signal TOM_EXT_TRIGIN[x:x] is in use and it will be synchronized to the selected **CMU** clock source before triggering, the initial delay can be defined as the time between the falling edge of the second synchronized trigger signal to the point of time of starting the generation of the PWM signal (green phase in the figure). On one hand, if CN0(old) is greater than or equal to updated CM0(new)-1, the counter will be reset and counts up to CM0(new)-1 and the initial delay is CM0(new)+1 selected **CMU** clock periods, i.e. CLSi_TOM_CHx_CM0.CM0+1 selected **CMU** clock periods. The behavior of the counter is the same as described above for the initial delay of the first OSM signal. On the other hand, if CN0(old) is smaller than updated CM0(new)-1, the initial phase or the initial delay is longer. In the initial phase, the counter will count up from CN0(old) to CM0(new)-1 and reset, and then increment again to CM0(new)-1. That means the initial delay is no longer CM0(new)+1, but the larger value of 2*CM0(new)-CN0(old) **CMU** clock periods, i.e. 2*CM0(new)-CM0(old)+1 selected **CMU** clock periods.

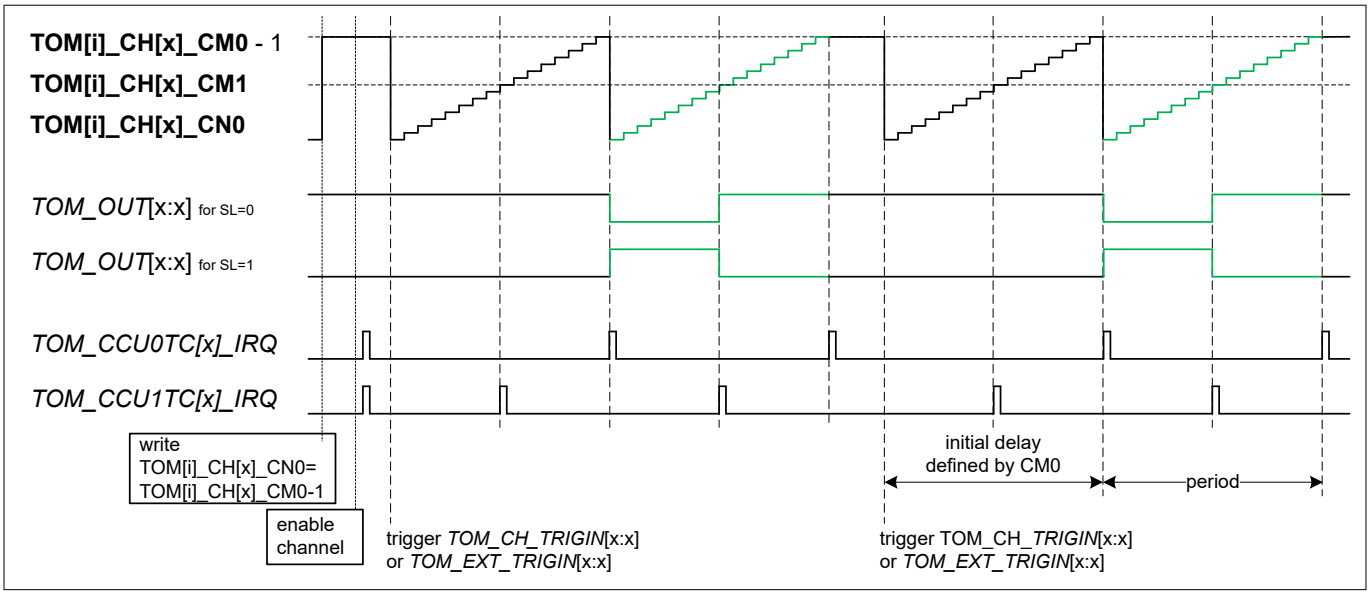


Figure 1069 PWM output with respect to configuration bit CLSi_TOM_CHx_CTRL.SL in one-shot counting mode if CLSi_TOM_CHx_CTRL.OSM_TRIG=1

As shown in Figure 1069, the generation of the interrupt signals during the generation of the PWM signal is similar as described in continuous counting up mode. As before enabling the channel the initial value written to CLSi_TOM_CHx_CN0.CN0 is greater than or equal to CLSi_TOM_CHx_CM0.CM0 - 1 and CLSi_TOM_CHx_CM1.CM1 - 1, both interrupt signals TOM_CCU0TC[x]_IRQ and TOM_CCU1TC[x]_IRQ are generated after 1 selected **CMU** clock period. Afterwards when CLSi_TOM_CHx_CN0.CN0 is reset from CLSi_TOM_CHx_CM0.CM0 -1 to 0, TOM_CCU0TC[x]_IRQ of channel x is generated. When CLSi_TOM_CHx_CN0.CN0 is incrementing to reach CLSi_TOM_CHx_CM1.CM1, TOM_CCU1TC[x]_IRQ of channel x is generated.

In one-shot mode, it is not recommended to configure CLSi_TOM_CHx_CM0.CM0<2. Otherwise, the behavior may be unexpected.

44.3.1.7.7 One-shot counting up-down mode

The **TOM** channel can operate in one-shot counting up-down mode when the bit CLSi_TOM_CHx_CTRL.OSM = 1 and the CLSi_TOM_CHx_CTRL.UDMODE != 00_B. One-shot mode means that a single pulse with the pulse level defined in bit CLSi_TOM_CHx_CTRL.SL is generated on the output signal.

Firstly, the channel has to be enabled by setting the corresponding CLSi_TOM_TGCg_ENDIS_STAT.ENDIS_STATc (c=0-7).

In one-shot mode the counter CLSi_TOM_CHx_CN0.CN0 will not be incremented once the channel is enabled.

When CLSi_TOM_CHx_CTRL.OSM_TRIG=0, a write-access to the register CLSi_TOM_CHx_CN0.CN0 triggers the start of incrementing the counter register CLSi_TOM_CHx_CN0.CN0 and pulse generation.

If the counter CLSi_TOM_CHx_CN0.CN0 is greater than or equal to CLSi_TOM_CHx_CM1.CM1, the output TOM_OUT[x:x] is set to CLSi_TOM_CHx_CTRL.SL value.

If the counter CLSi_TOM_CHx_CN0.CN0 is less than CLSi_TOM_CHx_CM1.CM1, the output TOM_OUT[x:x] is set to !CLSi_TOM_CHx_CTRL.SL value.

If the counter CLSi_TOM_CHx_CN0.CN0 has reached the value 0 (by counting down), it will stop.

The written value of CLSi_TOM_CHx_CN0.CN0 determines the start delay of the first edge. The delay time of the first edge is given by (CLSi_TOM_CHx_CM1.CM1 - CLSi_TOM_CHx_CN0.CN0) multiplied with the period defined by the current value of CLSi_TOM_CHx_CTRL.CLK_SRC.

Figure 1070 depicts the pulse generation in one-shot counting up-down mode. The interrupt signals are generated similarly as described in one-shot counting up mode and continuous counting up-down mode. The write-access of CLSi_TOM_CHx_CN0.CN0 can cause interrupts if the write value is greater than or equal to CLSi_TOM_CHx_CM1.CM1 - 1 or CLSi_TOM_CHx_CM0.CM0 -1. If the value written to CLSi_TOM_CHx_CN0.CN0 for triggering or retriggering the counter is greater than or equal to CLSi_TOM_CHx_CM1.CM1 -1, the output TOM_OUT[x:x] will be set to CLSi_TOM_CHx_CTRL.SL after 1 selected **CMU** clock period together with an interrupt on TOM_CCU1TC[x]_IRQ. Similarly but not shown in the figure, if the value written to CLSi_TOM_CHx_CN0.CN0 for triggering or retriggering the counter is greater than or equal to CLSi_TOM_CHx_CM0.CM0 - 1, an interrupt on TOM_CCU0TC[x]_IRQ will be generated after 1 selected **CMU** clock period. Afterwards when CLSi_TOM_CHx_CN0.CN0 is incrementing to reach CLSi_TOM_CHx_CM1.CM1, an interrupt on TOM_CCU1TC[x]_IRQ is generated. When CLSi_TOM_CHx_CN0.CN0 changes its counting direction from up to down, an interrupt on TOM_CCU0TC[x]_IRQ is generated.

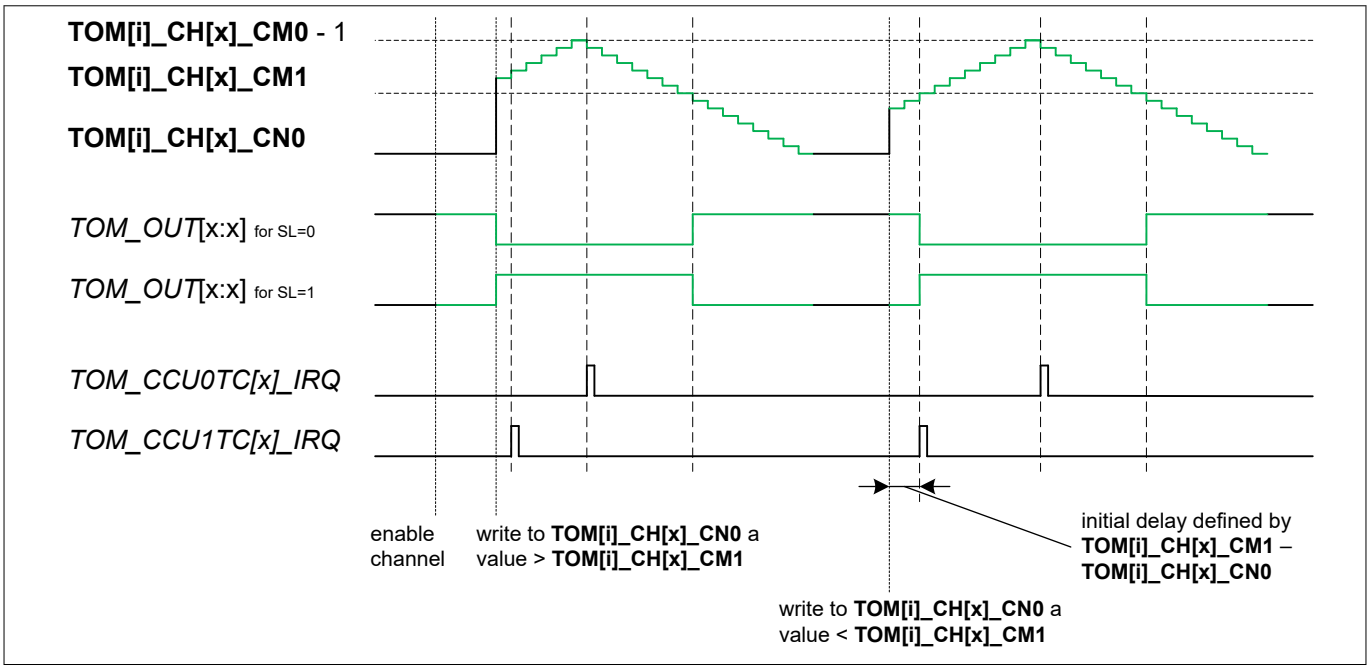


Figure 1070 PWM output with respect to configuration bit CLSi_TOM_CHx_CTRL.SL in one-shot counting up-down mode if TOM[i]_CH[x]_CRTL.OSM_TRIG= 0

Further output of single pulses can be started by writing to register CLSi_TOM_CHx_CN0.CN0.

If a channel is configured to one-shot counting up-down mode and configuration bit CLSi_TOM_CHx_CTRL.OSM_TRIG is set to 1, the falling edge of the trigger signal TOM_CH_TRIGIN[x:x] or TOM_EXT_TRIGIN_S (i.e. in fact the synchronized signal of TOM_EXT_TRIGIN[x:x]) triggers start of incrementing CLSi_TOM_CHx_CN0.CN0 and pulse generation. The output generation with interrupt signals is similar as mentioned above. If the counter CLSi_TOM_CHx_CN0.CN0 is greater than or equal to CLSi_TOM_CHx_CM1.CM1, the output TOM_OUT is set to CLSi_TOM_CHx_CTRL.SL value. Otherwise, the output TOM_OUT is set to ! CLSi_TOM_CHx_CTRL.SL value. When CLSi_TOM_CHx_CN0.CN0 is incrementing to reach CLSi_TOM_CHx_CM1.CM1, an interrupt is generated on TOM_CCU1TC[x]_IRQ. When CLSi_TOM_CHx_CN0.CN0 changes its counting direction from up to down, an interrupt is generated on TOM_CCU0TC[x]_IRQ.

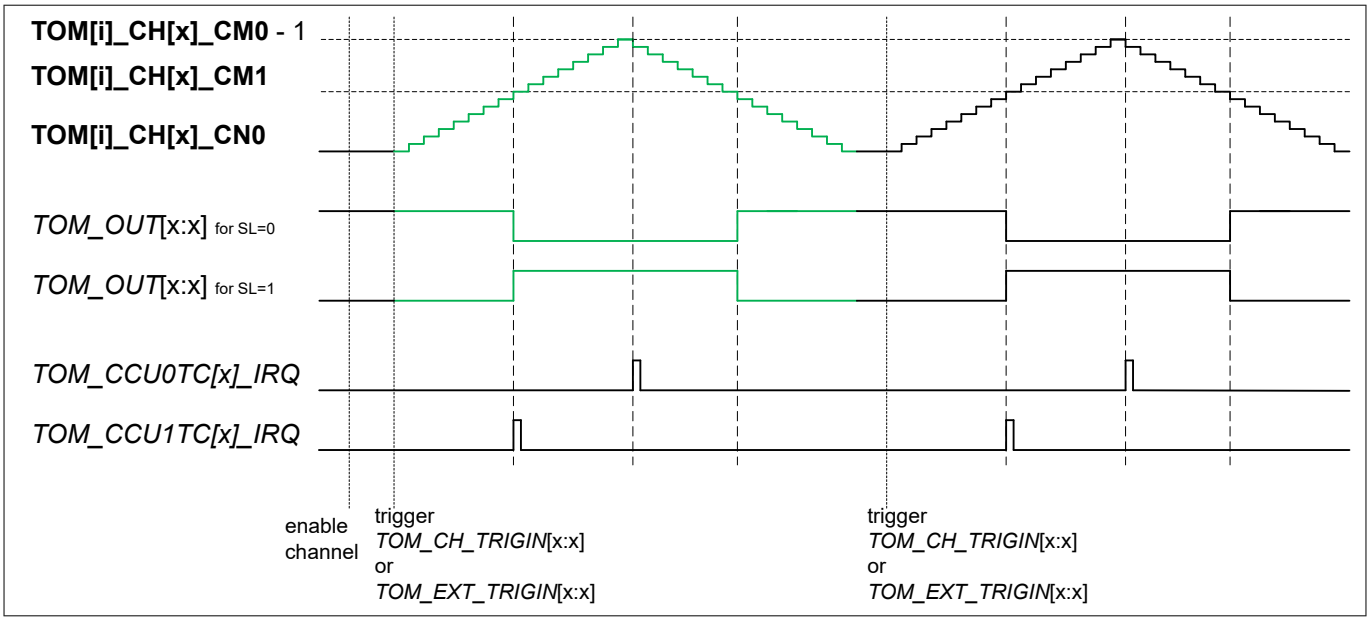


Figure 1071 PWM **output with respect to configuration bit** CLSi_TOM_CHx_CTRL.SL **in one-shot counting up-down mode if TOM[i]_CH[x]_CRTL.OSM_TRIG = 1**

44.3.1.7.8 Pulse count modulation mode

At the output TOM_OUT[15:15] a pulse count modulated signal can be generated instead of the simple PWM output signal. This specific behavior is called pulse count modulation (PCM) mode. PCM mode is only available when CLSi_TOM_CHx_CTRL.UDMODE=0.

Figure 1060 outlines the circuit for Pulse Count Modulation.

The PCM mode is enabled by setting bit CLSi_TOM_CHx_CTRL.BITREV to 1.

The PCM mode is only available for **TOM** channel 15.

With the configuration bit CLSi_TOM_CHx_CTRL.BITREV=1 a bit reversing of the counter output CLSi_TOM_CHx_CN0.CN0 is configured. In this case the bits are swapped, for example, LSB and MSB, LSB+1 and MSB-1, LSB+2 and MSB-2 and so on.

The effect of bit-reversing of the CLSi_TOM_CHx_CN0.CN0 register value is shown in the following Figure 1072.

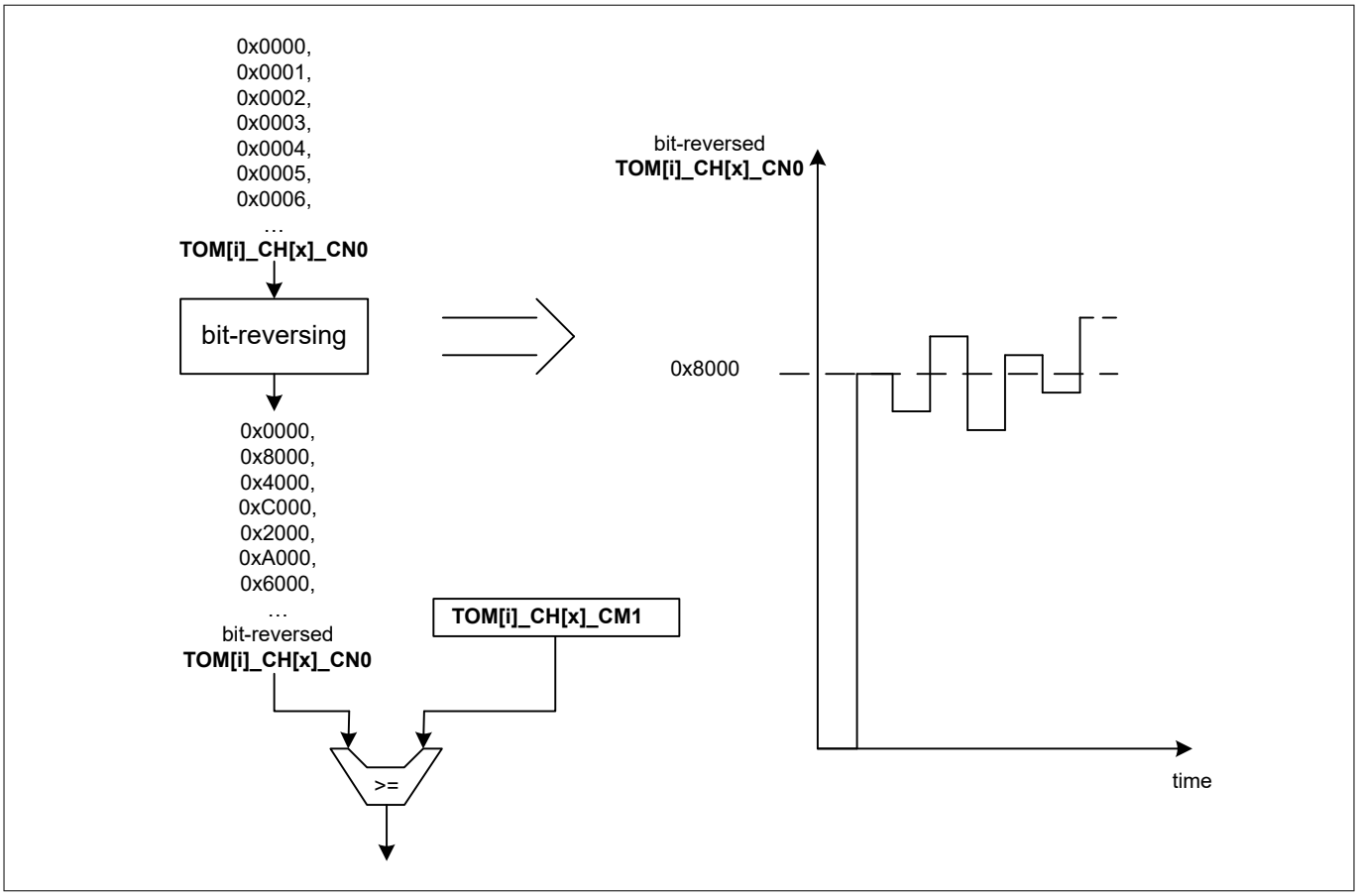


Figure 1072 Bit reversing of counter CLSi_TOM_CHx_CN0.CN0 output

In the PCM mode the counter CLSi_TOM_CHx_CN0.CN0 is incremented by every clock tick depending on configured **CMU** clock (CCM[i]_FXCLK_RES).

The output of counter CLSi_TOM_CHx_CN0.CN0 is first bit-reversed and then compared with the configured register value CLSi_TOM_CHx_CM1.CM1.

When the bit-reversed value of register CLSi_TOM_CHx_CN0.CN0 is greater than or equal to CLSi_TOM_CHx_CM1.CM1, the output TOM_OUT[15:15] is set to !CLSi_TOM_CHx_CTRL.SL and otherwise set to CLSi_TOM_CHx_CTRL.SL.

In PCM mode the CLSi_TOM_CHx_CM0.CM0 register, in which the period is defined, normally has to be set to its maximum value $FFFF_H$.

To reduce time period of updating pulse width value in CLSi_TOM_CHx_CM1.CM1 register, it is additionally possible to setup period value in CLSi_TOM_CHx_CM0.CM0 register to smaller values than the maximum value as described before.

Possible values for CLSi_TOM_CHx_CM0.CM0 register are each even numbered values to the power of 2 e.g. 8000_H, 4000_H, 2000_H ... .

In this case the pulse width has to be configured in the following manner.

Depending on how much the period in CLSi_TOM_CHx_CM0.CM0 register is decreased, means shifted right starting from 10000_H - the pulse width in CLSi_TOM_CHx_CM1.CM1 register has to be shifted left (= rotated: shift MSB back into LSB) with same value, e.g. :

Period CLSi_TOM_CHx_CM0.CM0 = 0100_H -> shifted 8 bits right from 10000_H

--> so pulse width has to be shifted 8 bits left:

e.g. 50% pulse width = 00080_H -> shift 8 bits left -> CLSi_TOM_CHx_CM1.CM1 = 8000_H

Table 2218 PCM examples

CM0 (period)	CM1 (pulse width)	Shift	CM1 (configured value)
$FFFF_H$	8000_H	No shift	8000_H
8000_H	4000_H	Shift 1 bit left	8000_H
4000_H	1000_H	Shift 2 bits left	4000_H
2000_H	$0FFF_H$	Shift 3 bits left	$7FF8_H$
1000_H	0333_H	Shift 4 bits left	3330_H
0800_H	0055_H	Shift 5 bits left	$0AA0_H$
0020_H	0008_H	Shift 11 bits left	4000_H
0010_H	0005_H	Shift 12 bits left	5000_H

In this mode the CLSi_TOM_CHx_IRQ_NOTIFY.CCU1TC (see register CLSi_TOM_CHx_IRQ_NOTIFY) is set every time if bit reverse value of CLSi_TOM_CHx_CN0.CN0 is greater than or equal to CLSi_TOM_CHx_CM1.CM1 which may be multiple times during one period. Therefore, from application point of view it is not useful to enable this interrupt.

44.3.1.7.9 Trigger generation

For applications with constant PWM period defined by CLSi_TOM_CHx_CM0.CM0, it is not necessary to update the CLSi_TOM_CHx_CM0.CM0 register regularly with CLSi_TOM_CHx_SR0.SR0 register. For these applications, the CLSi_TOM_CHx_SR0.SR0 register can be used to define an additional output signal and interrupt trigger event.

If bit CLSi_TOM_CHx_CTRL.SR0_TRIG is set to 1, the register CLSi_TOM_CHx_SR0.SR0 is no longer used as a shadow register for register CLSi_TOM_CHx_CM0.CM0. Instead, CLSi_TOM_CHx_SR0.SR0 is compared with CLSi_TOM_CHx_CN0.CN0. If both are equal, a pulse of signal level '1' is generated at the output TOM_OUT_T[x:x] of instance i, channel x. In this case, CLSi_TOM_CHx_CM1.CM1 will still be synchronously updated from its shadow register CLSi_TOM_CHx_SR1.SR1 if the update is enabled by setting CLSi_TOM_TGCg_GLB_CTRL.UPEN_CTRLc (c=0-7)=10_B.

The bit CLSi_TOM_CHx_CTRL.SR0_TRIG should only be set if bit CLSi_TOM_CHx_CTRL.RST_CCU0 of this channel is 0.

If bit CLSi_TOM_CHx_CTRL.SR0_TRIG=1 is set, the interrupt on TOM_CCU1TC[x]_IRQ is set at the same time with pulse generation on TOM_OUT_T in case of a compare equal match of CLSi_TOM_CHx_SR0.SR0 = CLSi_TOM_CHx_CN0.CN0. If bit CLSi_TOM_CHx_CTRL.SR0_TRIG is set the interrupt notify flag CLSi_TOM_CHx_IRQ_NOTIFY.CCU1TC is no longer set on a compare match of CLSi_TOM_CHx_CM1.CM1 and CLSi_TOM_CHx_CN0.CN0. Instead, the CLSi_TOM_CHx_IRQ_NOTIFY.CCU1TC interrupt notify flag is set in case of a compare equal match of CLSi_TOM_CHx_SR0.SR0 and CLSi_TOM_CHx_CN0.CN0.

With configuration bit CLSi_TOM_CHx_CTRL.TRIG_PULSE one can select if the output TOM_OUT_T[x:x] is high as long as CLSi_TOM_CHx_CN0.CN0= CLSi_TOM_CHx_SR0.SR0 (CLSi_TOM_CHx_CTRL.TRIG_PULSE=0) or if there will be only one pulse of length one cluster clock period when CLSi_TOM_CHx_CN0.CN0 becomes SR0 (CLSi_TOM_CHx_CTRL.TRIG_PULSE=1). Figure 1073 shows the case of CLSi_TOM_CHx_CTRL.TRIG_PULSE=1.

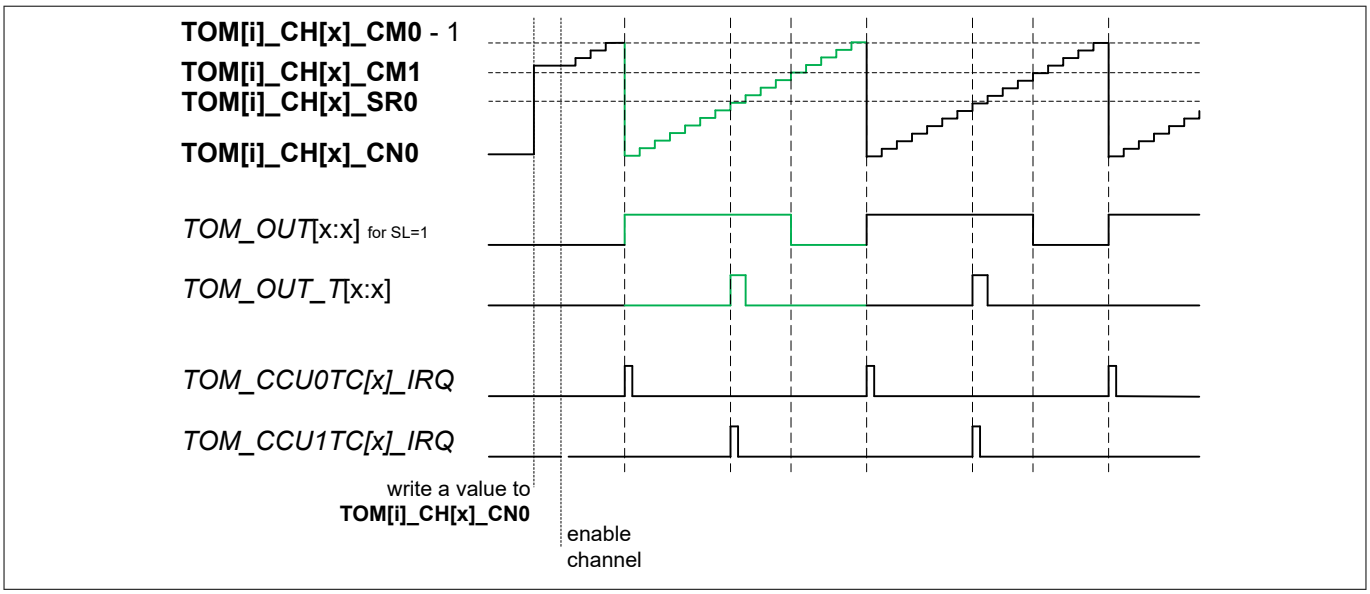


Figure 1073 Output behavior with respect to the CLSi_TOM_CHx_CTRL.TRIG_PULSE=1 in continuous counting up mode if CLSi_TOM_CHx_CTRL.SR0_TRIG=1

TOM/**SPE** interface must not be used when CLSi_TOM_CHx_CTRL.SR0_TRIG is set to 1.

The **TOM** output signal routing to **DTM** or **eGTM** toplevel is described in **DTM** connections on **eGTM** toplevel

44.3.1.7.10 TOM BLDC support

The **TOM** sub-module offers a BLDC support in combination with the **SPE** sub-module. **TOM** channels 0 to 7 can be used to drive a BLDC engine.

The BLDC support can be configured by setting the bit CLSi_TOM_CHx_CTRL.SPEM = 1. If this bit is set, the **TOM** channel output will be controlled through the TOM[i]_SPE_OUT[x:x] signal coming from SPE[i]_OUT[x:x] of the **SPE** sub-module (see Figure 1060). Please refer to Sensor pattern evaluation (**SPE**) for a detailed description of the **SPE** sub-module.

The **TOM[i]** channel [x] (x∈{2, 6, 7, 8, 9}) can be used together with the **SPE** module to trigger a delayed update of CLSi_SPE_OUT_CTRL register (i.e. commutation delay) after new input pattern detected by **SPE** (signaled by SPE[i]_NIPD). This feature is configured on **TOM[i]** channel[x] (x∈{2, 6, 7, 8, 9}) by setting CLSi_TOM_CHx_CTRL.SPE_TRIG=1, CLSi_TOM_CHx_CTRL.SPEM=0 or CLSi_TOM_CHx_CTRL.SPE_TRIG=0, CLSi_TOM_CHx_CTRL.SPEM=1 with CLSi_TOM_CHx_CTRL.OSM=1. With this configuration the **TOM[i]** channel [x] (x∈{2, 6, 7, 8, 9}) generates one single PWM pulse when receiving trigger signal TOM[i]_SPE_NIPD. Then the update of CLSi_SPE_OUT_CTRL register can be triggered by TOM[i]_CH[x]_TRIG_CCU1 that are generated by compare event in CCU1 unit. That means the update of CLSi_SPE_OUT_CTRL can be synchronized to the second edge of the one shot PWM signal.

For details please refer to chapter of **SPE** sub-module description.

Normally the trigger signals TOM[i]_CH[x]_TRIG_CCU0/TOM[i]_CH[x]_TRIG_CCU1 to **SPE** sub-module are single-cycle pulses with the length of one cluster clock-cycle. Depending on the configuration, it may happen that the trigger signal TOM[i]_CH[x]_TRIG_CCU0/TOM[i]_CH[x]_TRIG_CCU1 to **SPE** sub-module is high for two system clock-cycles once at the starting condition.

44.3.1.7.11 TOM gated counter mode

Each **TOM SPE** module combination provides also the feature of a gated counter mode. This is reached by using the *FSOI[i]* input of the **SPE** module i (see Figure 1119) to gate the clock of a CCU0 sub-module.

To configure this mode, register of module **SPE** should be set as following:

- The **SPE** should be enabled (bit CLSi_SPE_CTRL_STAT.EN = 1)
- All three **TIM** inputs should be disabled (CLSi_SPE_CTRL_STAT.SIEk (k=0-2) **(k=0)** = CLSi_SPE_CTRL_STAT.SIEk (k=0-2) **(k=1)** = CLSi_SPE_CTRL_STAT.SIEk (k=0-2) **(k=2)** = 0) according to Figure 1124,
- CLSi_SPE_OUT_CTRL.SPE_OUT_CTRLo (o=0-7) **(o=x)** should be set to 10_B (set TOM[i]_SPE_OUT[x:x] to '0'),
- Mode FSOM should be enabled (CLSi_SPE_CTRL_STAT.FSOM=1), CLSi_SPE_CTRL_STAT.FSOL[x:x] if channel x of module **TOM** i is chosen for gated counter mode

Additionally in module **TOM**:

- The **SPE** mode should be disabled (CLSi_TOM_CHx_CTRL.SPEM=0)
- The gated counter mode should be enabled (CLSi_TOM_CHx_CTRL.GCM=1)

As a result of this configuration, the counter CLSi_TOM_CHx_CN0.CN0 in sub-module CCU0 of **TOM** channel x counts as long as input *FSOI[i]* of the connected **SPE** module is '0'.

44.3.1.7.12 TOM FREEZE mode

It is possible, to disable the **TOM** channel (TOM_ENDIS=0) without changing internal and output register states by setting of CLSi_TOM_CHx_CTRL.FREEZE to 1. After re-enabling again (TOM_ENDIS=1), the channel is started from the state from which it was deactivated (TOM_ENDIS=0).

The following sequence has to be realized to use the FREEZE mode in a reliable manner:

1. **TOM** channel setup and enable by setting TOM_ENDIS=1
2. Setting of CLSi_TOM_CHx_CTRL.FREEZE to 1
3. **TOM** channel disable by setting TOM_ENDIS=0
4. **TOM** channel re-enable by setting TOM_ENDIS=1
5. Setting of CLSi_TOM_CHx_CTRL.FREEZE to 0

***Note**:* CLSi_TOM_CHx_CTRL.FREEZE *must always be set, while the channel is enabled (*TOM_ENDIS*=1).*

44.3.1.7.13 TOM software reset of channels

Each channel x (x=g*2+c) is assigned a unique bit c in the configuration register CLSi_TOM_TGCg_GLB_CTRL for software reset of the **TOM[i]** channel x.

Writing the value 1_B to the bit field CLSi_TOM_TGCg_GLB_CTRL.RST_CHc (c=0-7) via the configuration interface will immediately reset the content of the following channel x registers to the initial hardware reset state. x = g*2+c

- CLSi_TOM_CHx_CTRL
- CLSi_TOM_CHx_CTRL_SR
- CLSi_TOM_CHx_CN0
- CLSi_TOM_CHx_CM0
- CLSi_TOM_CHx_CM1
- CLSi_TOM_CHx_SR0
- CLSi_TOM_CHx_SR1
- CLSi_TOM_CHx_STAT
- CLSi_TOM_CHx_IRQ_NOTIFY
- CLSi_TOM_CHx_IRQ_EN
- CLSi_TOM_CHx_IRQ_MODE
- Internal signals inside the **TOM[i]** channel [x]:

Atomic reset of multiple channels is possible by writing a 1 on each associated bit. E.g.: Writing $0F_H$ to register CLSi_TOM_TGCg_GLB_CTRL **(i=0;g=0)** will reset the TOM0 channel .

44.3.1.7.14 eGTM TOM Configuration Registers Description

Sub-chapter TOM Configuration Registers Description

Table 2219

Register Shortname	Description
CLSi_TOM_TGCg_GLB_CTRL	TOM[i] TGC [g] global control register (g=0-1;i=0-2)
CLSi_TOM_TGCg_ENDIS_CTRL	TOM[i] TGC [g] enable/disable control register (g=0-1;i=0-2)
CLSi_TOM_TGCg_ENDIS_STAT	TOM[i] TGC [g] enable/disable status register (g=0-1;i=0-2)
CLSi_TOM_TGCg_ACT_TB	TOM[i] TGC [g] action time base register (g=0-1;i=0-2)
CLSi_TOM_TGCg_OUTEN_CTRL	TOM[i] TGC [g] output enable control register (g=0-1;i=0-2)
CLSi_TOM_TGCg_OUTEN_STAT	TOM[i] TGC [g] output enable status register (g=0-1;i=0-2)
CLSi_TOM_TGCg_FUPD_CTRL	TOM[i] TGC [g] force update control register (g=0-1;i=0-2)
CLSi_TOM_TGCg_INT_TRIG	TOM[i] TGC [g] internal trigger control register (g=0-1;i=0-2)
CLSi_TOM_CHx_CTRL	TOM[i] channel [x] control register (x=0-1,3-5;i=0-2)
CLSi_TOM_CHx_CTRL	TOM[i] channel [x] control register (x=8-9;i=0-2)
CLSi_TOM_CHx_CTRL	TOM[i] channel [x] control register (x=10-14;i=0-2)
CLSi_TOM_CHx_CTRL	TOM[i] channel [x] control register (x=15;i=0-2)
CLSi_TOM_CHx_CTRL_SR	TOM[i] channel [x] control shadow register (x=0-15;i=0-2)
CLSi_TOM_CHx_CN0	TOM[i] channel [x] CCU0 counter (x=0-15;i=0-2)
CLSi_TOM_CHx_CM0	TOM[i] channel [x] CCU0 compare register (x=0-15;i=0-2)
CLSi_TOM_CHx_SR0	TOM[i] channel [x] CCU0 compare shadow register (x=0-15;i=0-2)
CLSi_TOM_CHx_CM1	TOM[i] channel [x] CCU1 compare register (x=0-15;i=0-2)
CLSi_TOM_CHx_SR1	TOM[i] channel [x] CCU1 compare shadow register (x=0-15;i=0-2)
CLSi_TOM_CHx_STAT	TOM[i] channel [x] status register (x=0-15;i=0-2)
CLSi_TOM_CHx_IRQ_NOTIFY	TOM[i] channel [x] interrupt notification register (x=0-15;i=0-2)
CLSi_TOM_CHx_IRQ_EN	TOM[i] channel [x] interrupt enable register (x=0-15;i=0-2)

(table continues...)

Table 2219 (continued)

Register Shortname	Description
CLSi_TOM_CHx_IRQ_FORCINT	TOM[i] channel [x] force interrupt register (x=0-15;i=0-2)
CLSi_TOM_CHx_IRQ_MODE	TOM[i] channel [x] interrupt mode register (x=0-15;i=0-2)

44.3.1.7.15 TOM signal description

Table 2220 TOM Reset

Name	Description
TOM[j]_RESET_CH[x]	**TOM[j]** channel [x] reset active signal

Table 2221 TOM Interrupt Signals

Name	Description
TOM_CCU1TC[x]_IRQ	CCU1 Trigger condition interrupt for channel x of each **TOM** instance i (i:={0, 1,..., NTOM-1})
TOM_CCU0TC[x]_IRQ	CCU0 Trigger condition interrupt for channel x of each **TOM** instance (i:={0, 1,..., NTOM-1})

Table 2222 TOM Signals

Name	Description
CTRL_TRIG	Trigger signal to update the TGC[g] registers CLSi_TOM_TGCg_OUTEN_STAT, CLSi_TOM_TGCg_ENDIS_STAT and to activate TOM_FUPD signals
TOM_EXT_TRIGIN_S	synchronized TOM_EXT_TRIGIN *[x:x]* (x:={0, 1,..., 15}) signal to selected **CMU** clock source
TOM_ENDIS	A bundle of channel enable signals controlled by TGC[g] (g:={0, 1}) unit
TOM_OUTEN	A bundle of channel enable signals controlled by TGC[g] (g:={0, 1}) unit
TOM_FUPD	A bundle of channel force update event signals controlled by TGC[g] (g:={0, 1}) unit
TOM_RSTCN0_CH	A bundle of channel counter reset trigger signal at force update event that are controlled by TGC[g] (g:={0, 1}) unit
TOM_UPEN	A bundle of channel update enable signals controlled by TGC[g] (g:={0, 1}) unit
SEL_FXCMU_CLKEN	The selected fixed **CMU** clock enable signal
TRIG_CCU0	Trigger signal for CCU0 compare match
TRIG_CCU1	Trigger signal for CCU1 compare match

(table continues...)

Table 2222 (continued) TOM Signals

Name	Description
TOM_CH_TRIGIN	Trigger input signal for each channel x. It comes from the preceding channel x-1 or channel 15 of the preceding instance.
TOM_CH_TRIGOUT	Trigger output signal for each channel x. It is connected to the trigger input signal of the next channel x+1 or channel 0 of the next instance.

44.3.1.7.16 TOM port description

Table 2223 TOM Signal Interface

Name	Description
TOM_EXT_TRIGIN	TOM_EXT_TRIGIN[7:0] of **TOM** instance i are connected to TIM_EXT_CAPTURE *[7:0]* signal from **TIM** instance i while TOM_EXT_TRIGIN[15:08] of **TOM** instance i are connected to TIM_EXT_CAPTURE *[7:0]* signal from **TIM** instance i
TOM_TRIGIN	**TOM** module trigger chain input
TOM_TRIGOUT	**TOM** module trigger chain output
TOM_TRIGOUT_DEL	**TOM** module trigger chain output delayed by 1 cluster clock; will be used only in **TOM** [0] instance as TOM_TRIGIN
TOM_OUT	Output signal TOM_OUT
TOM_OUT_T	Output signal TOM_OUT_T

Table 2224 TOM Interrupt Interface

Name	Description
TOM_IRQ	**TOM IRQ** line signals
TOM_IRQ_CLR	**TOM IRQ** clear signal, hardware clear of CLSi_ATOM_CHx_IRQ_NOTIFY register
TOM_IRQ_OCC	**TOM IRQ** occurred signal, corresponds to CLSi_ATOM_CHx_IRQ_NOTIFY register

Table 2225 TOM SPE Interface

Name	Description
TOM[i]_SPE_NIPD	Special interface signal connected to SPE[i]_NIPD of **SPE** module i. That means new input pattern detected by **SPE** module i.
TOM[i]_SPE_OUT	Special interface signal connected to SPE[i]_OUT of **SPE** module i.
TOM[i]_CH[x]_SOUR	Special interface signals to **SPE** instance i; driven by **TOM** instance i, channel x.

(table continues...)

Table 2225 (continued) TOM SPE Interface

Name	Description
TOM[i]_CH[x]_TRIG_CCU1	Special interface signal to **SPE** instance i; connected to the internal signal TRIG_CCU1 of **TOM** instance i, channel x (with x∈{0, 2, 6, 7, 8, 9}.
TOM[i]_CH[x]_TRIG_CCU0	Special interface signals to **SPE** instance i; connected to the internal signal TRIG_CCU0 of driven by **TOM** instance i, channel x (with x = 0) .

44.3.1.8 ARU-connected timer output module (ATOM)

44.3.1.8.1 Overview

The **ARU**-connected Timer Output Module (**ATOM**) is able to generate complex output signals without interaction via the configuration interface . Each **ATOM** sub-module contains eight output channels which can operate independently from each other in several configurable operation modes. A block diagram of the **ATOM** sub-module is depicted in Figure 1074.

Indices and their ranges as used inside this chapter are:

- i:={0, 1,..., NATOM-1} instance index of cluster / module
- NATOM= number of **ATOM** instances in the device
- x:={0, 1,..., 7} index of channels in each **ATOM** instance
- k:={0, 1,..., 7} index of sub-channels

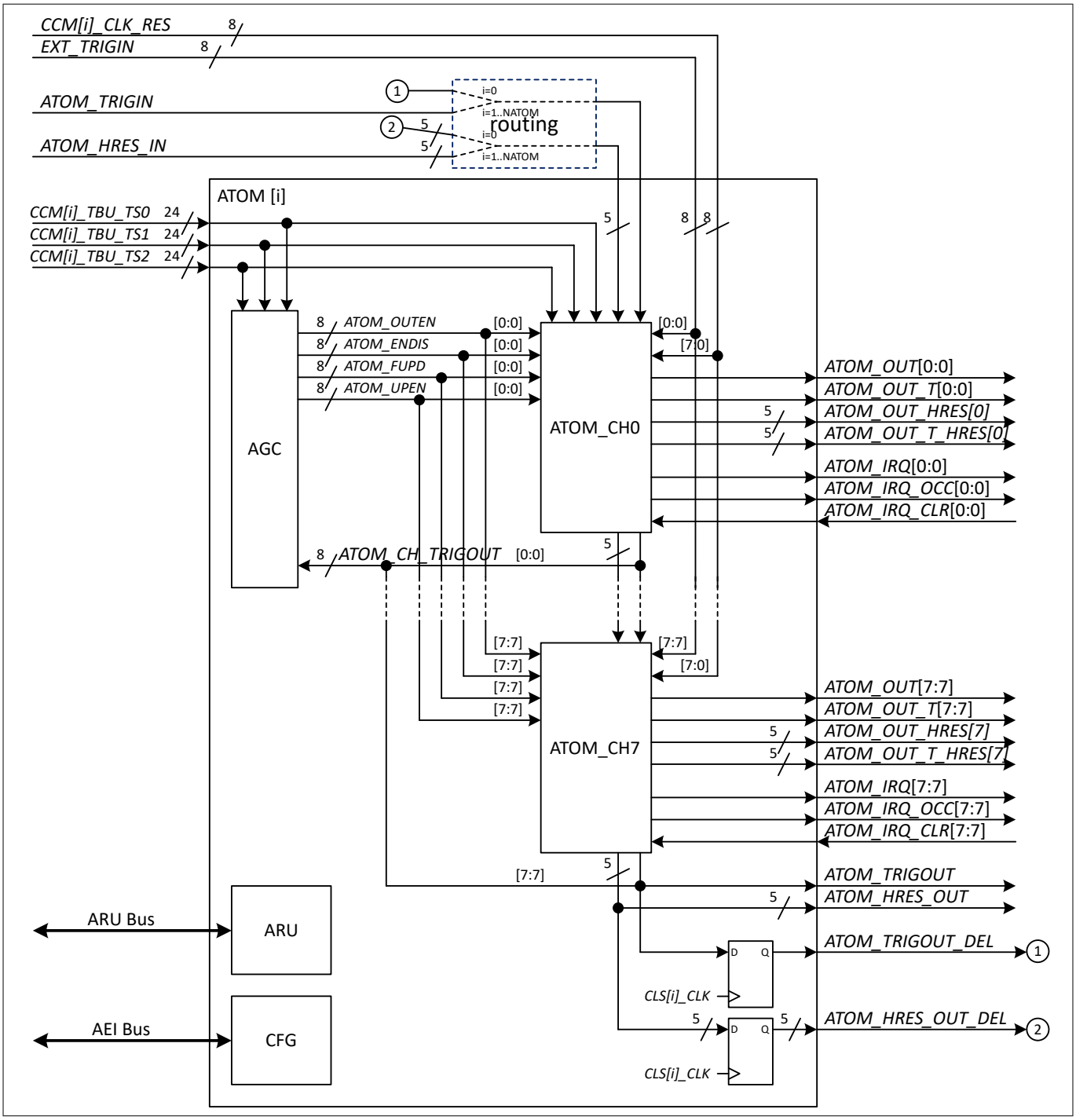


Figure 1074 ATOM block diagram

***Note**: The figure is a general picture from the IP vendor. The exact configuration can be found in the device specific configuration.*

The architecture of **ATOM** sub-module is similar to **TOM** sub-module with some differences. Firstly, each **ATOM** integrates only eight channels with outputs ATOM_OUT and ATOM_OUT_T. There is one **ATOM** Global Control sub-unit (AGC) for the **ATOM** channels. Furthermore, **ATOM** channels are able to generate various PWM signals in different modes.

The **ATOM** output signal routing to **DTM** or **eGTM** toplevel is described in **DTM** connections on **eGTM** toplevel

Each **ATOM** channel provides five modes of operation:

- **ATOM** Signal Output Mode Immediate (SOMI)
- **ATOM** Signal Output Mode Compare (SOMC)
- **ATOM** Signal Output Mode PWM (SOMP)
- **ATOM** Signal Output Mode Serial (SOMS)
- **ATOM** Signal Output Mode Buffered Compare (SOMB)

These modes are described in more detail in **ATOM** channel modes.

The input clocks for the **ATOM** channels come from the cluster clock CLS[j]_CLK with the configurable clock resolution CCM[i]_CLK_RES signals of the **CCM** sub-module. This gives the freedom to select a programmable input clock for the **ATOM** channel counters. The module **ATOM** receives three timestamp values CCM[i]_TBU_TS0, CCM[i]_TBU_TS1 and CCM[i]_TBU_TS2 in order to realize synchronized output behavior on behalf of a common time base.

The operation registers (e.g. counter, compare registers) of the **ATOM** channel are 24 bits wide. Each **ATOM** channel provides a so-called operation register set CLSi_ATOM_CHx_CM0, CLSi_ATOM_CHx_CM1 and shadow register set CLSi_ATOM_CHx_SR0, CLSi_ATOM_CHx_SR1. With this architecture it is possible to work with the operation register set, while the shadow register set can be reloaded with new parameters over configuration interface.

As shown in Figure 1074, NATOM-1 **ATOM** sub-modules are connected together through a trigger chain. The trigger signal ATOM_TRIGIN of **ATOM** instance i is coming from ATOM_TRIGOUT of **ATOM** instance i-1 and the trigger signal ATOM_TRIGOUT of **ATOM** instance i is routed to ATOM_TRIGIN of **ATOM** instance i+1. This trigger chain goes across all channels one by one. The trigger input signal ATOM_TRIGIN of **ATOM** instance i is connected to the trigger input signal ATOM_CH_TRIGIN[0:0] of channel 0. The channel 0 can generate a trigger output signal ATOM_CH_TRIGOUT[0:0] that is connected to ATOM_CH_TRIGIN[1:1] to trigger channel 1. In this way, each channel x is triggered from the preceding channel and then can trigger the next channel (please refer to Figure 1076 and Internal trigger interface for the internal trigger chain details). Channel 7 of **ATOM** instance i is routed to trigger channel 0 of the next instance i+1.

***Note**: The following paragraph is only relevant for devices with High Resolution.*

Similarly, there is also a HRES value chain that goes across all channels of each instance along with the trigger chain. The high-resolution value input ATOM_HRES_IN of **ATOM** instance i is connected to ATOM_HRES_OUT of **ATOM** instance i-1 and the high-resolution value output ATOM_HRES_OUT of **ATOM** instance i is routed to ATOM_HRES_IN of **ATOM** instance i+1. Please refer to High-resolution PWM support (HRES mode) for more details.

Especially, the trigger input signal **ATOM** instance 0 is connected to its own output ATOM_TRIGOUT_DEL, i.e. the last channel of **ATOM** instance 0 can trigger the first channel of **ATOM** instance 0 (this path is registered, which means delayed by one cluster clock period).

***Note**: The following paragraph is only relevant for devices with High Resolution.*

Similarly, the high-resolution value input ATOM_HRES_IN of **ATOM** instance 0 is connected to its own output ATOM_HRES_OUT_DEL, i.e. the high-resolution value output of the last channel of **ATOM** instance 0 is also fed to the first channel of **ATOM** instance 0, also with one pipeline register delay.

The function of the interrupt signals ATOM_IRQ, ATOM_IRQ_OCC and ATOM_IRQ_CLR is described in more details in **eGTM** interrupt concept.

ATOM global control (AGC)

Synchronous start, stop and update of work register of up to 8 channels is possible with the AGC sub-unit. This sub-unit has the same functionality as the TGC sub-unit of the **TOM** sub-module.

Overview

One global channel control unit (AGC) exists to drive a number of individual **ATOM** channels synchronously by external or internal events.

An AGC can drive up to eight **ATOM** channels.

The **ATOM** sub-module supports four different kinds of signaling mechanisms:

- Global enable/disable mechanism for each **ATOM** channel with control register CLSi_ATOM_AGC_ENDIS_CTRL and status register CLSi_ATOM_AGC_ENDIS_STAT
- Global output enable mechanism for each **ATOM** channel with control register CLSi_ATOM_AGC_OUTEN_CTRL and status register CLSi_ATOM_AGC_OUTEN_STAT
- Global force update mechanism of the register CLSi_ATOM_CHx_CM0.CM0, CLSi_ATOM_CHx_CM1.CM1, CLSi_ATOM_CHx_CTRL.SL and CLSi_ATOM_CHx_CTRL.CLK_SRC for each **ATOM** channel and optionally reset the counter CLSi_ATOM_CHx_CN0 with control register CLSi_ATOM_AGC_FUPD_CTRL
- Update enable of the register CLSi_ATOM_CHx_CM0.CM0, CLSi_ATOM_CHx_CM1.CM1, CLSi_ATOM_CHx_CTRL.SL and CLSi_ATOM_CHx_CTRL.CLK_SRC for each **ATOM** channel with the control bit field CLSi_ATOM_AGC_GLB_CTRL.UPEN_CTRLk (k=0-7) **(k=x)**

AGC sub-unit

Each of the first three individual mechanisms (enable/disable of the channel, output enable and force update) can be driven by three different trigger sources.

The three trigger sources are:

1. over the configuration interface (bit CLSi_ATOM_AGC_GLB_CTRL.HOST_TRIG)
2. the **TBU** timestamp (signal CCM[i]_TBU_TS0, CCM[i]_TBU_TS1 and CCM[i]_TBU_TS2 if available)
3. the internal trigger signal ATOM_CH_TRIGOUT (bunch of trigger signals ATOM_CH_TRIGOUT[x:x]) which can be either the trigger ATOM_TRIG_CCU0 of channel x, the trigger of preceding channel x-1 (i.e. signal ATOM_CH_TRIGOUT[x-1:x-1]) or the external trigger EXT_TRIGIN[x:x] of assigned **TIM** channel x.

The first way (1.) is to trigger the control mechanism by a direct register write-access via configuration interface (bit CLSi_ATOM_AGC_GLB_CTRL.HOST_TRIG).

The second way (2.) is provided by a compare match trigger on behalf of a specified time base coming from the module **TBU** (selected by bits CLSi_ATOM_AGC_ACT_TB.TBU_SEL) and the timestamp compare value defined in the bit field CLSi_ATOM_AGC_ACT_TB.ACT_TB.

In this case, the selected **TBU** time base CCM[i]_TBU_TS0/CCM[i]_TBU_TS1/CCM[i]_TBU_TS2 should be configured to be up-counting. A cyclic event compare of CLSi_ATOM_AGC_ACT_TB.ACT_TB and selected CCM[i]_TBU_TS0, CCM[i]_TBU_TS1, CCM[i]_TBU_TS2 is performed. Please refer to Cyclic event compare for more information about cyclic event compare strategy.

The third way (3.) is the input ATOM_CH_TRIGOUT (bunch of trigger signals ATOM_CH_TRIGOUT[x:x]) coming from **ATOM** channels.

The corresponding trigger signal ATOM_CH_TRIGOUT[x:x] coming from channel x can be masked by the register CLSi_ATOM_AGC_INT_TRIG.

To enable or disable each individual **ATOM** channel, the registers CLSi_ATOM_AGC_ENDIS_CTRL and/or CLSi_ATOM_AGC_ENDIS_STAT have to be used.

The register CLSi_ATOM_AGC_ENDIS_STAT directly controls the signal ATOM_ENDIS[x:x]. A write-access to this register is possible.

The register CLSi_ATOM_AGC_ENDIS_CTRL is a shadow register that overwrites the value of register CLSi_ATOM_AGC_ENDIS_STAT if one of the three trigger conditions matches.

The update mechanism within the channel is not affected by the ATOM_ENDIS control signal. This means that the update of the working register can be executed - depending on the update condition - as soon as it is enabled by setting CLSi_ATOM_AGC_GLB_CTRL.UPEN_CTRLk (k=0-7) **(k=x)** $=10_B$.

To ensure a clean restart of an **ATOM** channel, it is recommended to raise a channel reset by setting of bit CLSi_ATOM_AGC_GLB_CTRL.RST_CHk (k=0-7) **(k=x)** after a channel was disabled (ATOM_ENDIS[x:x]=0) and before it is enabled again (ATOM_ENDIS[x:x]=1).

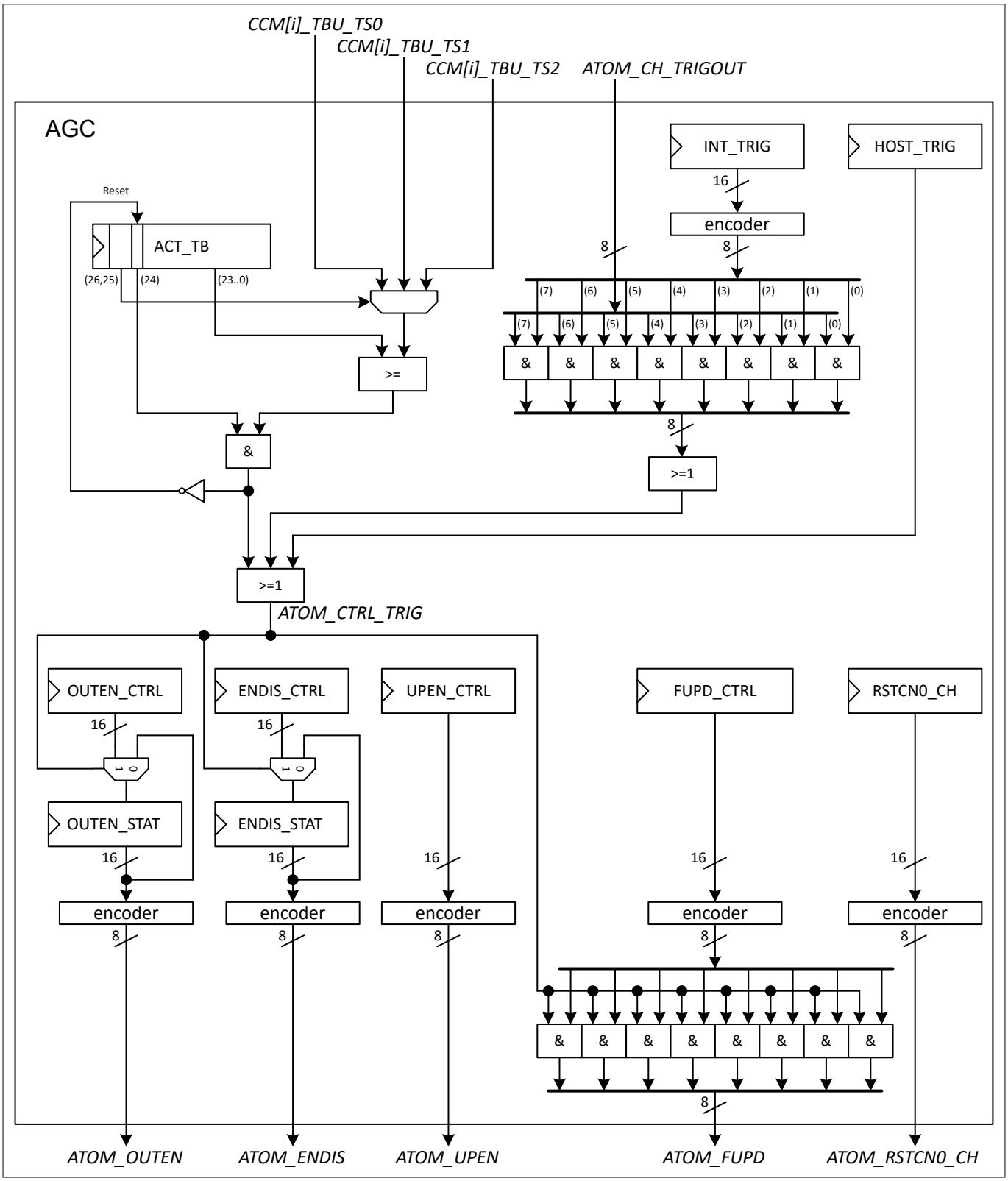


Figure 1075 ATOM global channel control mechanism

The output of the individual **ATOM** channels can be controlled using the registers CLSi_ATOM_AGC_OUTEN_CTRL and CLSi_ATOM_AGC_OUTEN_STAT.

The register CLSi_ATOM_AGC_OUTEN_STAT directly controls the signal ATOM_OUTEN. A write-access to this register is possible.

The register CLSi_ATOM_AGC_OUTEN_CTRL is a shadow register that overwrites the value of register CLSi_ATOM_AGC_OUTEN_STAT if one of the three trigger conditions matches.

The trigger condition has always priority over the bus write-access to the CLSi_ATOM_AGC_OUTEN_STAT and CLSi_ATOM_AGC_ENDIS_STAT registers, even if CLSi_ATOM_AGC_OUTEN_CTRL.OUTEN_CTRLk (k=0-7)/ CLSi_ATOM_AGC_ENDIS_CTRL.ENDIS_CTRLk (k=0-7) is set to 00_B. This means that the bus write-access to CLSi_ATOM_AGC_OUTEN_STAT and CLSi_ATOM_AGC_ENDIS_STAT register is ignored in the clock cycle when the trigger condition is active.

If an **ATOM** output is disabled by the register CLSi_ATOM_AGC_OUTEN_STAT, the actual value of the channel output at ATOM_OUT[x:x] is set as !CLSi_ATOM_CHx_CTRL.SL.

If the output is enabled, the output at ATOM_OUT[x:x] depends on the current operation of different mode, CLSi_ATOM_AGC_ENDIS_STAT and CLSi_ATOM_CHx_CTRL.FREEZE. (Please check CLSi_ATOM_CHx_CTRL_SOMP.SL/CLSi_ATOM_CHx_CTRL_SOMI.SL/CLSi_ATOM_CHx_CTRL_SOMB.SL/ CLSi_ATOM_CHx_CTRL_SOMS.SL for more information)

The register CLSi_ATOM_AGC_FUPD_CTRL.FUPD_CTRLk (k=0-7) defines which of the **ATOM** channels receive a force update event if the trigger signal ATOM_CTRL_TRIG is raised while CLSi_ATOM_AGC_FUPD_CTRL.RSTCN0_CHk (k=0-7) determines whether the force update event also reset counter CLSi_ATOM_CHx_CN0.CN0 of the channel. If in SOMP continuous counting up mode with CLSi_ATOM_CHx_CTRL_SOMP.RST_CCU0=0 the force update mechanism is configured together with CLSi_ATOM_AGC_FUPD_CTRL.RSTCN0_CHk (k=0-7)=10_B and CLSi_ATOM_AGC_FUPD_CTRL.FUPD_CTRLk (k=0-7)=10_B, the force update event will simultaneously update the operation registers, reset the counter and also set the output to be CLSi_ATOM_CHx_CTRL_SOMP.SL. That means the force update mechanism can be used to stop the current PWM signal generation and restart a new PWM signal generation with new PWM parameters.

In SOMP, SOMI and SOMS modes, the force update request is synchronized to the selected CCM[i]_CLK_RES as shown in Figure 1076 and then executed. In other modes, the force update request is executed immediately. The execution of the force update does not depend on the state of CLSi_ATOM_AGC_ENDIS_STAT.ENDIS_STATk (k=0-7).

The ATOM_RSTCN0_CH signal is not synchronized with the selected CCM[i]_CLK_RES. In order to ensure that the reset function is executed, the bit field CLSi_ATOM_AGC_FUPD_CTRL.RSTCN0_CHk (k=0-7) **(k=x)** may only be set to 01_B after the selected CCM[i]_CLK_RES has occured.

The register bits CLSi_ATOM_AGC_GLB_CTRL.UPEN_CTRLk (k=0-7) **(k=x)** define, for which **ATOM** channel the update of the working register CLSi_ATOM_CHx_CM0.CM0, CLSi_ATOM_CHx_CM1.CM1, CLSi_ATOM_CHx_CTRL.SL and CLSi_ATOM_CHx_CTRL.CLK_SRC is enabled by the corresponding shadow register CLSi_ATOM_CHx_SR0.SR0, CLSi_ATOM_CHx_SR1.SR1, CLSi_ATOM_CHx_CTRL_SR.SL_SR and CLSi_ATOM_CHx_CTRL_SR.CLK_SRC_SR. If update is enabled in SOMP continuous counting up mode, the register CLSi_ATOM_CHx_CM0.CM0, CLSi_ATOM_CHx_CM1.CM1, CLSi_ATOM_CHx_CTRL.SL and CLSi_ATOM_CHx_CTRL.CLK_SRC will be updated on reset of counter register CLSi_ATOM_CHx_CN0.CN0 (see Figure 1076).

ATOM channel mode overview

Each **ATOM** channel offers the following different operation modes:

In **ATOM** Signal Output Mode Immediate (SOMI), the **ATOM** channels generate an output signal according to CLSi_ATOM_CHx_CTRL_SOMI.ACB0.

In **ATOM** Signal Output Mode Compare (SOMC), the **ATOM** channel generates an output signal on behalf of timestamps that are located in the **ATOM** operation registers CLSi_ATOM_CHx_CM0 and CLSi_ATOM_CHx_CM1. The output generates edges when the **TBU** timestamps reach the timestamps stored in CLSi_ATOM_CHx_CM0 and CLSi_ATOM_CHx_CM1. The **ATOM** is able to receive new timestamps by configuration interface. The new

timestamps are directly loaded into the channel's operation register. The shadow registers are used as capture registers for two time base values, when a compare match of the channel's operation registers occurs.

In **ATOM** Signal Output Mode PWM (SOMP), the **ATOM** channel is able to generate simple and complex PWM output signals like the **TOM** sub-module by comparing its operation registers with a sub-module internal counter. Unlike the **TOM**, the **ATOM** shadow registers can be reloaded over the configuration interface , while the channel operates on the operation registers. Please refer to **ATOM** signal output mode PWM (SOMP) for further details.

In **ATOM** Signal Output Mode Serial (SOMS), the **ATOM** channel generates a serial output bit stream on behalf of a shift register. The number of bits shifted and the shift direction is configurable. The shift frequency is determined by one of the CCM[i]_CLK_RES clock resolution signals. Please refer to **ATOM** signal output mode serial (SOMS) for further details.

In **ATOM** Signal Output Buffered Compare (SOMB), the **ATOM** channel generates an output signal on behalf of timestamps that are located in the **ATOM** operation registers CLSi_ATOM_CHx_CM0 and CLSi_ATOM_CHx_CM1. The output generates edges when the **TBU** timestamps reach the timestamps stored in CLSi_ATOM_CHx_CM0 and CLSi_ATOM_CHx_CM1. The **ATOM** is able to receive new compare values over the configuration interface. The new compare values are stored first in the shadow register. Only if previous compare match has occurred, the operation registers are updated with the content of the shadow registers. Please refer to **ATOM** signal output mode buffered compare (SOMB) for further details.

44.3.1.8.2 ATOM channel architecture

Each **ATOM** channel is able to generate output signals according to five operation modes. The architecture of the **ATOM** channels is similar to the architecture of the **TOM** channels. The architecture of an **ATOM** channel in SOMP mode is depicted in Figure 1076.

***Note**:* Figure 1076 *doesn't reflect the real implementation of the circuit but is intended to clearly explain the functionality of the channel, especially for SOMP mode.*

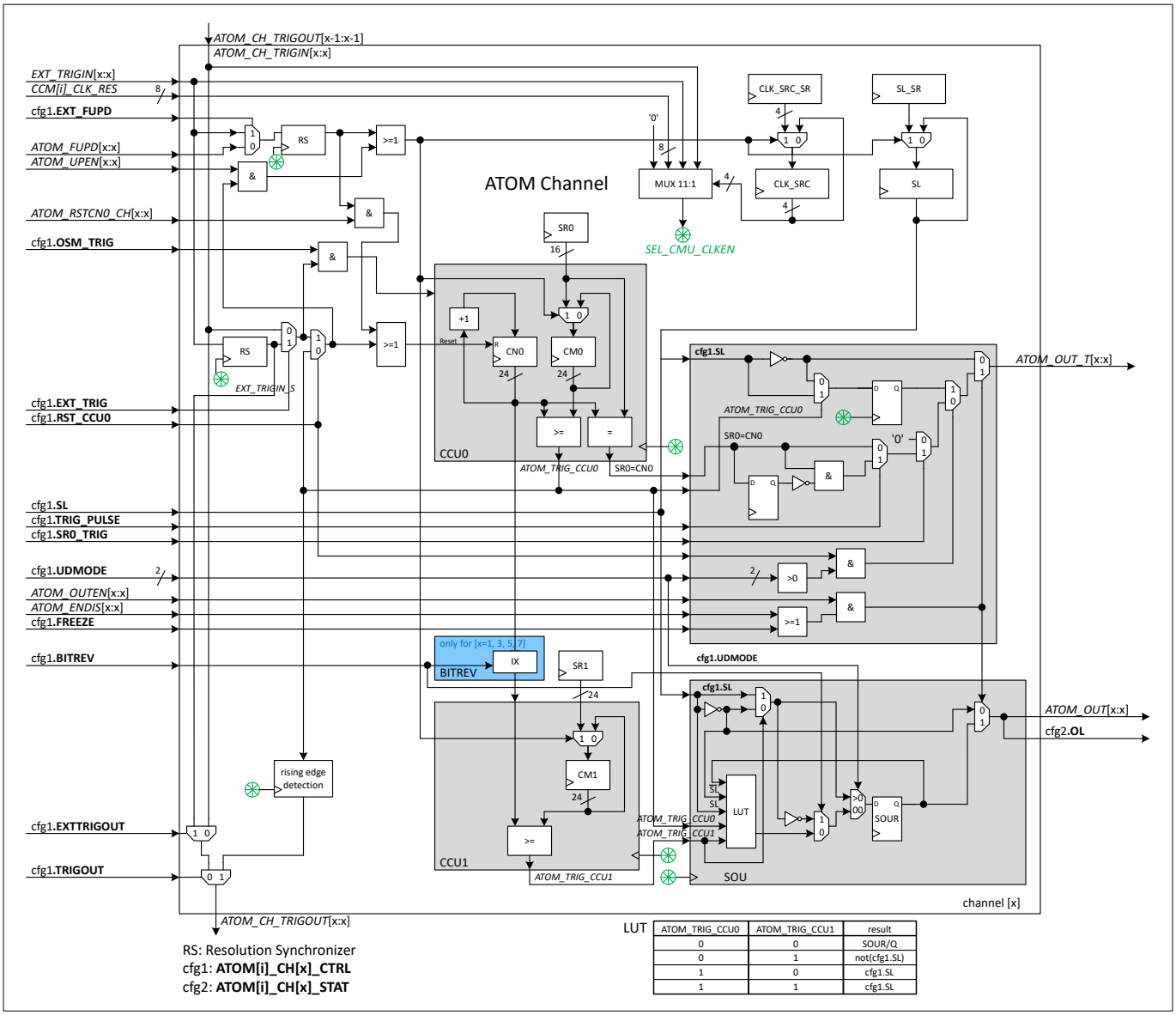


Figure 1076 ATOM channel architecture in SOMP mode

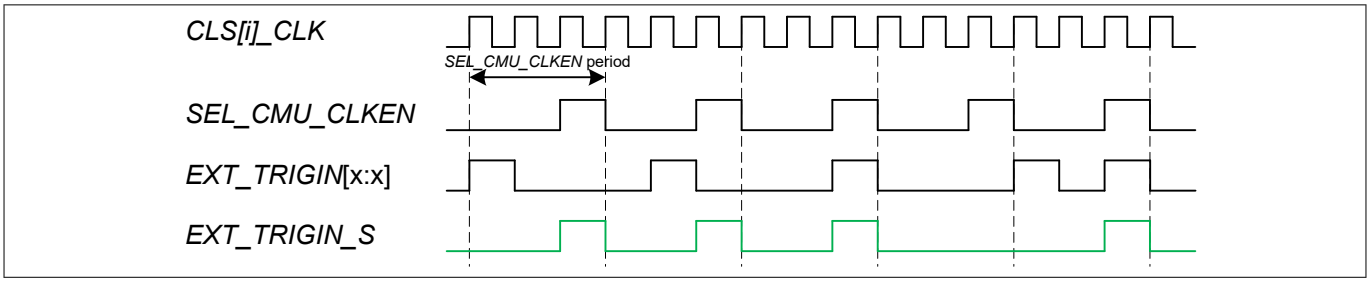


Figure 1077 Timing diagram of RS unit

As shown in Figure 1076, the operating clock resolution signal SEL_CMU_CLKEN in SOMP and SOMS mode can be configured by CLSi_ATOM_CHx_CTRL.CLK_SRC. This configuration register can select the clock source from **CMU** clock enable signals CCM[i]_CLK_RES that are provided from **CCM** sub-module.

In addition, with the shadow register CLSi_ATOM_CHx_CTRL_SR.CLK_SRC_SR, it is possible to change the selected clock resolution from **CMU** under different conditions. The signal SEL_CMU_CLKEN looks like clock enable signal that is shown in Figure 1077. Most functionalities in SOMP and SOMS modes are only active when

SEL_CMU_CLKEN = 1 and the consequent operation registers and outputs are always synchronously updated at the falling edge of SEL_CMU_CLKEN pulse. The functionalities synchronous to the selected clock resolution SEL_CMU_CLKEN are depicted with the green round symbol in Figure 1076.

Due to a fixed clock resolution scheme from **CMU**, the first clock resolution period may differ from the expected value, depending on the point of time, where the CLSi_ATOM_CHx_CTRL.CLK_SRC bit field is updated from its shadow register bit field CLSi_ATOM_CHx_CTRL_SR.CLK_SRC_SR.

In all **ATOM** channels the operation registers CLSi_ATOM_CHx_CN0.CN0, CLSi_ATOM_CHx_CM0.CM0 and CLSi_ATOM_CHx_CM1.CM1 and the shadow registers CLSi_ATOM_CHx_SR0.SR0 and CLSi_ATOM_CHx_SR1.SR1 are 24 bits wide. As shown in Figure 1076, the unit CCU0 provides different counting up/down functionalities in various modes. The comparators inside CCU0 and CCU1 perform selectable signed greater-equal or less-equal comparison against different parameters depending on different **ATOM** modes and then generate ATOM_TRIG_CCU0=1 when the compare match event in CCU0 occurs or generate ATOM_TRIG_CCU1=1 when the compare match event in CCU1 occurs.

The Signal Output Unit (SOU) generates the output signal for each **ATOM** channel. In SOMI mode, this output signal level depends on the bit CLSi_ATOM_CHx_CTRL_SOMI.SL in combination with two control bits CLSi_ATOM_CHx_CTRL_SOMI.ACB0.

The Signal Output Unit (SOU) generates the output signal for each **ATOM** channel. In SOMC mode, this output signal level depends on the bit CLSi_ATOM_CHx_CTRL_SOMC.SL in combination with five control bits CLSi_ATOM_CHx_CTRL_SOMC.ACB10 and CLSi_ATOM_CHx_CTRL_SOMC.ACB42.

The Signal Output Unit (SOU) generates the output signal for each **ATOM** channel. In SOMS mode, this output signal level depends on the bit CLSi_ATOM_CHx_CTRL_SOMS.SL in combination with two control bits CLSi_ATOM_CHx_CTRL_SOMS.ACB0.

The Signal Output Unit (SOU) generates the output signal for each **ATOM** channel. In SOMP mode, this output signal level depends on the bit CLSi_ATOM_CHx_CTRL_SOMP.SL in combination with the counter behavior and its compare events in CCU0 and CCU1.

The Signal Output Unit (SOU) generates the output signal for each **ATOM** channel. In SOMB mode, this output signal level depends on the bit CLSi_ATOM_CHx_CTRL_SOMB.SL in combination with five control bits CLSi_ATOM_CHx_CTRL_SOMB.ACB10 and CLSi_ATOM_CHx_CTRL_SOMC.ACB42.

In SOMP continuous counting up mode, CLSi_ATOM_CHx_CM0.CM0 represents PWM period while CLSi_ATOM_CHx_CM1.CM1 is PWM pulse width. In order to generate various PWM signals, CCU0 counts up CLSi_ATOM_CHx_CN0.CN0 and compares it with CLSi_ATOM_CHx_CM0.CM0 while CCU1 compares it with CLSi_ATOM_CHx_CM1.CM1. Depending on the configuration bit CLSi_ATOM_CHx_CTRL_SOMP.RST_CCU0, the counter CLSi_ATOM_CHx_CN0.CN0 can be reset, when the counter value is equal to the compare value CLSi_ATOM_CHx_CM0.CM0 (i.e. CLSi_ATOM_CHx_CN0.CN0 counts only from 0 to CLSi_ATOM_CHx_CM0.CM0-1 and is then reset to 0) or when it is triggered by the **ATOM[i]** trigger signal ATOM_CH_TRIGIN[x:x] (ATOM_CH_TRIGOUT[x-1:x-1]) of the preceding channel x-1 which can also be the last channel of preceding instance **ATOM[i-1]**) or when it is triggered by the external trigger signal EXT_TRIGIN[x:x] of the assigned **TIM** channel x.

The external trigger signal EXT_TRIGIN[x:x] is synchronous to the clock source selected by RS (Resolution Synchronizer) unit and then taken into use.

With such a RS unit, in each SEL_CMU_CLKEN period (i.e. selected clock resolution signal), any event (or pulses) of EXT_TRIGIN will be synchronized by RS unit on EXT_TRIGIN_S that looks exactly the same as SEL_CMU_CLKEN. Figure 1077 shows the timing diagram of RS unit. If EXT_TRIGIN has one or more pulses in a SEL_CMU_CLKEN period, the synchronized output EXT_TRIGIN_S will have 1 pulse at the end of this period. Otherwise, EXT_TRIGIN_S is always 0.

The trigger signal ATOM_CH_TRIGIN[x:x] coming from the preceding channel may not be synchronous to the selected clock resolution of channel x.

In SOMP continuous counting up mode, once the comparison event of CLSi_ATOM_CHx_CN0.CN0 >= CLSi_ATOM_CHx_CM0.CM0-1 in unit CCU0 occurs, ATOM_TRIG_CCU0 = 1 will be set to trigger the output generation of ATOM_OUT[x:x] in the SOU sub-unit together with the corresponding interrupt signals. In order to

trigger the next channel, the rising edge detection unit can detect each rising edge of ATOM_TRIG_CCU0 (i.e. each comparison event) and generate a single cluster clock pulse on ATOM_CH_TRIGOUT[x:x] when the selected clock resolution signal SEL_CMU_CLKEN =1 (i.e. synchronous to the selected clock resolution). In addition, ATOM_CH_TRIGOUT[x:x] can also be configured to be ATOM_CH_TRIGIN[x:x] (ATOM_CH_TRIGOUT[x-1:x-1] of the preceding channel x-1) or the external trigger signal EXT_TRIGIN[x:x] of the assigned **TIM** channel x. Similarly, once the comparison event of CLSi_ATOM_CHx_CN0.CN0 >= CLSi_ATOM_CHx_CM1.CM1-1 in unit CCU1 occurs, ATOM_TRIG_CCU1 = 1 will be set to trigger the output generation of ATOM_OUT[x:x] in the SOU sub-unit together with the corresponding interrupt signals.

When CLSi_ATOM_CHx_CN0.CN0 < CLSi_ATOM_CHx_CM0.CM0-1, ATOM_TRIG_CCU0 will be reset as 0. Similarly, when CLSi_ATOM_CHx_CN0.CN0 < CLSi_ATOM_CHx_CM1.CM1-1, ATOM_TRIG_CCU1 will be reset to 0.

As shown in Figure 1077, in SOMP continuous counting up mode according to the LUT with ATOM_TRIG_CCU0, ATOM_TRIG_CCU1 and the signal level configuration CLSi_ATOM_CHx_CTRL.SL, SOU unit can generate ATOM_OUT[x:x] with various PWM timing characteristics when the channel and the output are both enabled (ATOM_OUTEN[x:x]=1 and ATOM_ENDIS[x:x]=1). Please refer to the chapter of different modes for more details on PWM signal generation.

In SOMC/SOMB mode, CCU0 and CCU1 compare CLSi_ATOM_CHx_CM0.CM0/CLSi_ATOM_CHx_CM1.CM1 against the **eGTM** time bases CCM[i]_TBU_TS0, CCM[i]_TBU_TS1, and if available, CCM[i]_TBU_TS2 generates the compare events ATOM_TRIG_CCU0 and ATOM_TRIG_CCU1. Please refer to **TBU** Time base unit (**TBU**) for further details. In SOMC/SOMB mode, the two compare units CCU0/CCU1 can be used in combination with each other. When used in combination, the signals ATOM_TRIG_CCU0 and ATOM_TRIG_CCU1 can be used to enable/disable the other compare unit when a match event occurs. Different compare strategy can be configured in CLSi_ATOM_CHx_CTRL_SOMC.ACB42/CLSi_ATOM_CHx_CTRL_SOMB.ACB42. Based on the signal level register bit CLSi_ATOM_CHx_CTRL.SL and the two control bits CLSi_ATOM_CHx_CTRL_SOMC.ACB10/ CLSi_ATOM_CHx_CTRL_SOMB.ACB10, the output ATOM_OUT[x:x] reflects the configured compare strategy. This behavior is not really illustrated in Figure 1076. Please refer to the **ATOM** signal output mode compare (SOMC) and **ATOM** signal output mode buffered compare (SOMB) for further details.

In SOMC/SOMB mode, CCU0/CCU1 unit always performs cyclic event compare between the selected time base CCM[i]_TBU_TS0/CCM[i]_TBU_TS1/CCM[i]_TBU_TS2 and the compare value CLSi_ATOM_CHx_CM0.CM0/ CLSi_ATOM_CHx_CM1.CM1. When the selected time base is "cyclically greater than or equal to" the compare value of CLSi_ATOM_CHx_CM0.CM0/CLSi_ATOM_CHx_CM1.CM1, the compare match event specified in CLSi_ATOM_CHx_CM0.CM0/ CLSi_ATOM_CHx_CM1.CM1 is considered to have occurred. Please refer to Cyclic event compare for more information about cyclic event compare strategy.

For a correct behavior of cyclic event compare in SOMC/SOMB mode, when a compare match event with the specified compare value CLSi_ATOM_CHx_CM0.CM0/ CLSi_ATOM_CHx_CM1.CM1 occurs, the new compare value in CLSi_ATOM_CHx_CM0.CM0/ CLSi_ATOM_CHx_CM1.CM1 for the next cyclic event compare must be specified in the range of the half total time base value (i.e. $7FFFFF_H$) greater/smaller than the old compare value.

In SOMI mode, the output signal level depends on CLSi_ATOM_CHx_CTRL.SL and CLSi_ATOM_CHx_CTRL_SOMI.ACB0. In SOMS mode, the output signal level is defined by the bit pattern that has to be shifted out by the **ATOM** channel. The bit pattern is located inside the CLSi_ATOM_CHx_CM1.CM1 register.

When the output is disabled (ATOM_OUTEN[x:x]=0), the output signals ATOM_OUT[x:x] and ATOM_OUT_T[x:x] are the inverse value of the CLSi_ATOM_CHx_CTRL.SL bit. When the output is enabled (ATOM_OUTEN[x:x]=1), the output signals ATOM_OUT[x:x] and ATOM_OUT_T[x:x] are dependent on different operation modes, the channel enable signal ATOM_ENDIS[x:x] and the freeze mode enable register CLSi_ATOM_CHx_CTRL.FREEZE. (Please refer to register description of CLSi_ATOM_CHx_CTRL_SOMI.SL/CLSi_ATOM_CHx_CTRL_SOMB.SL/ CLSi_ATOM_CHx_CTRL_SOMC.SL/CLSi_ATOM_CHx_CTRL_SOMP.SL/CLSi_ATOM_CHx_CTRL_SOMS.SL for more details.) Especially, in SOMB, SOMC and SOMI modes, the output signal ATOM_OUT_T[x:x] is not supported. So it is always clamped to 0 if the output and the channel are both enabled (i.e. ATOM_OUTEN[x:x]=1 and ATOM_ENDIS[x:x]=1). When the output is enabled (ATOM_OUTEN[x:x]=1) but the channel is disabled (ATOM_ENDIS[x:x]=0), if CLSi_ATOM_CHx_CTRL.FREEZE=0, ATOM_OUT_T[x:x] is !CLSi_ATOM_CHx_CTRL.SL.

When the output is enabled (ATOM_OUTEN[x:x]=1) but the channel is disabled (ATOM_ENDIS[x:x]=0), if CLSi_ATOM_CHx_CTRL.FREEZE=1, ATOM_OUT_T[x:x] is clamped to 0.

When the channel is disabled (ATOM_ENDIS[x:x]=0), the counting and comparing functionalities in CCU0 and CCU1 are deactivated in different modes and thus ATOM_TRIG_CCU0 = 0, ATOM_TRIG_CCU1 = 0. Therefore, ATOM_CH_TRIGOUT[x:x] is not generated if CLSi_ATOM_CHx_CTRL.TRIGOUT = 1, regardless of ATOM_OUTEN[x:x].

Similar to CLSi_ATOM_CHx_CTRL.CLK_SRC, CLSi_ATOM_CHx_CTRL.SL, CLSi_ATOM_CHx_CM0.CM0 and CLSi_ATOM_CHx_CM1.CM1 also have their shadow registers CLSi_ATOM_CHx_CTRL_SR.SL_SR, CLSi_ATOM_CHx_SR0.SR0 and CLSi_ATOM_CHx_SR1.SR1. In SOMP and SOMS mode, it is possible to synchronously update the clock resolution register CLSi_ATOM_CHx_CTRL.CLK_SRC and the operation registers CLSi_ATOM_CHx_CM0.CM0 and CLSi_ATOM_CHx_CM1.CM1 from their shadow registers (please refer to here). In addition, synchronous updating of the signal level register CLSi_ATOM_CHx_CTRL.SL is possible in SOMP mode. There are different update mechanisms in SOMP and SOMS mode. Please refer to the chapters SOMP and SOMS modes for more details.

External trigger interface

Each channel x of **ATOM** instance i has an input signal EXT_TRIGIN[x:x] that is connected to TIM_EXT_CAPTURE[x:x] of **TIM** instance i. Via this signal, events (high pulses with the length of 1 CLS[j]_CLK period) generated by the **TIM** module can be used to control the functionality of the **ATOM** channel in multiple ways:

- EXT_TRIGIN[x:x] events can be used to define the clock source of the selected channel mode (CLSi_ATOM_CHx_CTRL_SOMP.CLK_SRC = 1110_B; CLSi_ATOM_CHx_CTRL_SOMS.CLK_SRC = 1110_B).
- EXT_TRIGIN[x:x] events can be used to update the operation registers with the content of the shadow registers in SOMP, SOMI, SOMS, SOMB mode (activated by CLSi_ATOM_CHx_CTRL_SOMP.EXT_FUPD = 1; CLSi_ATOM_CHx_CTRL_SOMI.EXT_FUPD = 1; CLSi_ATOM_CHx_CTRL_SOMS.EXT_FUPD = 1; CLSi_ATOM_CHx_CTRL_SOMB.EXT_FUPD = 1). Any EXT_TRIGIN[x:x] event is synchronized in the **ATOM** channel as specified in Figure 1077.
- EXT_TRIGIN[x:x] events can be used to reset the counter register CLSi_ATOM_CHx_CN0 in SOMP continuous counting up mode or change the counting direction in counting up-down mode or trigger the start of counting in one-shot mode (activated by CLSi_ATOM_CHx_CTRL_SOMP.EXT_TRIG = 1 && CLSi_ATOM_CHx_CTRL_SOMP.RST_CCU0 = 1). In this case, if CLSi_ATOM_AGC_GLB_CTRL.UPEN_CTRLk (k=0-7) **(k=x)** = 10_B is configured, EXT_TRIGIN[x:x] events can also trigger the update of the operation registers from their shadow registers. Any EXT_TRIGIN[x:x] event is synchronized in the **ATOM** channel as specified in Figure 1077.
- EXT_TRIGIN[x:x] can be used as a trigger event for the following channels (activated by CLSi_ATOM_CHx_CTRL_SOMI.EXTTRIGOUT = 1 && CLSi_ATOM_CHx_CTRL_SOMI.TRIGOUT = 0; CLSi_ATOM_CHx_CTRL_SOMS.EXTTRIGOUT = 1 && CLSi_ATOM_CHx_CTRL_SOMS.TRIGOUT = 0; CLSi_ATOM_CHx_CTRL_SOMP.EXTTRIGOUT = 1 && CLSi_ATOM_CHx_CTRL_SOMP.TRIGOUT = 0; CLSi_ATOM_CHx_CTRL_SOMC.EXTTRIGOUT = 1 && CLSi_ATOM_CHx_CTRL_SOMC.TRIGOUT = 0; CLSi_ATOM_CHx_CTRL_SOMB.EXTTRIGOUT = 1 && CLSi_ATOM_CHx_CTRL_SOMB.TRIGOUT = 0). Any EXT_TRIGIN[x:x] event is synchronized in the **ATOM** channel as specified in Figure 1077.

In certain configurations of the **TIM** module, the EXT_TRIGIN[x:x] signal pulses can have a length of more than 1 cluster clock-cycle. In this case, the functionality of the **ATOM** is not specified and it is highly recommended to prevent this configuration in applications.

Before triggering the functionalities mentioned above, the EXT_TRIGIN[x:x] signal is always synchronized with the current selected clock enable signal in RS unit as shown in Figure 1077.

Internal trigger interface

As shown in Figure 1076, each **ATOM** channel x has an input signal ATOM_CH_TRIGIN[x:x] that is coming from ATOM_CH_TRIGOUT[x-1:x-1] from the preceding channel. Via this signal, events (high pulses with the length of 1 CLS[j]_CLK period) generated by the preceding **ATOM** channel, can be used to control the functionality of **ATOM** channel in multiple ways:

- ATOM_CH_TRIGIN[x:x] events can be used to define the clock source of the selected channel mode (CLSi_ATOM_CHx_CTRL_SOMP**.**CLK_SRC = 1101_B; CLSi_ATOM_CHx_CTRL_SOMS**.**CLK_SRC = 1101_B).
- ATOM_CH_TRIGIN[x:x] events can be used to reset the counter register CLSi_ATOM_CHx_CN0 in SOMP continuous counting up mode or change the counting direction in counting up-down mode or trigger the start of counting in one-shot mode (activated by CLSi_ATOM_CHx_CTRL_SOMP**.**EXT_TRIG = 0 && CLSi_ATOM_CHx_CTRL_SOMP**.**RST_CCU0 = 1). In this case, if CLSi_ATOM_AGC_GLB_CTRL**.**UPEN_CTRLk (k=0-7) = 10_B is configured, ATOM_CH_TRIGIN[x:x] events can also trigger the update of the operation registers from their shadow registers.
- ATOM_CH_TRIGIN[x:x] can be used as a trigger event for following channels (activated by CLSi_ATOM_CHx_CTRL_SOMI**.**EXTTRIGOUT = 0 && CLSi_ATOM_CHx_CTRL_SOMI**.**TRIGOUT = 0; CLSi_ATOM_CHx_CTRL_SOMS**.**EXTTRIGOUT = 0 && CLSi_ATOM_CHx_CTRL_SOMS**.**TRIGOUT = 0; CLSi_ATOM_CHx_CTRL_SOMP**.**EXTTRIGOUT = 0 && CLSi_ATOM_CHx_CTRL_SOMP**.**TRIGOUT = 0; CLSi_ATOM_CHx_CTRL_SOMC**.**EXTTRIGOUT = 0 && CLSi_ATOM_CHx_CTRL_SOMC**.**TRIGOUT = 0; CLSi_ATOM_CHx_CTRL_SOMB**.**EXTTRIGOUT = 0 && CLSi_ATOM_CHx_CTRL_SOMB**.**TRIGOUT = 0).

Especially, for channel 0 of **ATOM** module i, i ∈ {1, 2, ..., NATOM-1}, ATOM_CH_TRIGIN[0:0] is connected to ATOM_TRIGIN of instance [i] that comes from ATOM_CH_TRIGOUT[7:7] of the preceding **ATOM** instance [i-1] and may be delayed by a register with the cluster clock CLS[j]_CLK in channel 0 before triggering the functionalities as mentioned above. This delay is controlled by the device configuration variable ATOM_TRIG_CHAIN (indicated by CLSi_CCM_HW_CONF**.**ATOM_TRIG_CHAIN). The delay exists after every N clusters (N = ATOM_TRIG_CHAIN). For ATOM0, ATOM_CH_TRIGIN[0:0] is connected to ATOM_TRIGOUT_DEL via the signal ATOM_CH_TRIGOUT[7:7] of ATOM0 itself as shown in Figure 1074. Furthermore, the cluster i can operate on faster **eGTM** clock CLK, if the device is defined with FAST_CLS_CLK[i] = 1. In this case, it is allowed to have one more register that is clocked with CLK to delay ATOM_CH_TRIGIN[4:4] in channel 4 before triggering the functionalities. This is controlled by another **ATOM** device configuration parameter ATOM_TRIG_INTCHAIN (it can only be valid if FAST_CLS_CLK[i] = 1 and indicated by CLSi_CCM_HW_CONF**.**ATOM_TRIG_INTCHAIN). The registers for these special cases are not shown in Figure 1076.

Unlike EXT_TRIGIN[x:x], ATOM_CH_TRIGIN[x:x] is not synchronized with the selected clock enable signal of channel x. For the application of 2 channels generating PWM signals synchronously, CLSi_ATOM_CHx_CTRL_SOMP**.**CLK_SRC of the triggering channel and the triggered channel should be configured with the same clock resolution and the same cluster clock.

44.3.1.8.3 ATOM channel modes

As described above, each **ATOM** channel can operate independently from each other in one of five dedicated output modes:

- **ATOM** Signal Output Mode Immediate (SOMI)
- **ATOM** Signal Output Mode Compare (SOMC)
- **ATOM** Signal Output Mode PWM (SOMP)
- **ATOM** Signal Output Mode Serial (SOMS)
- **ATOM** Signal Output Mode Buffered Compare (SOMB)

The Signal Output Mode PWM (SOMP) is principally the same as the output mode for the **TOM** sub-module. The other modes provide additional functionality for signal output control. All operation modes are described in more detail in the following sections.

In any output mode, if a channel is enabled, one-shot mode is disabled (CLSi_ATOM_CHx_CTRL.OSM=0; only used in modes SOMP and SOMS) and CLSi_ATOM_CHx_CM0.CM0 >= CLSi_ATOM_CHx_CN0.CN0, the counter CLSi_ATOM_CHx_CN0.CN0 is incrementing until it reaches CLSi_ATOM_CHx_CM0.CM0.

To avoid unintended counting of CLSi_ATOM_CHx_CN0.CN0 after enabling a channel, it is recommended to reset a channel (or at least CLSi_ATOM_CHx_CN0.CN0 and CLSi_ATOM_CHx_CM0.CM0) before any change on the mode bits CLSi_ATOM_CHx_CTRL.MODE and CLSi_ATOM_CHx_CTRL.OSM.

In the following sections the terms "100% duty cycle" and "0% duty cycle" will be used. 100% duty cycle is defined as a constant output level on CLSi_ATOM_CHx_CTRL_SOMP.SL of ATOM_OUT*[x]* over the whole period. 0% duty cycle is defined as a constant output level on !CLSi_ATOM_CHx_CTRL_SOMP.SL of ATOM_OUT*[x]* over the whole period.

ATOM signal output mode immediate (SOMI)

In **ATOM** Signal Output Mode Immediate (SOMI), the **ATOM** channel generates output signals on the ATOM_OUT of instance i and channel x output port immediately after the update of the bit CLSi_ATOM_CHx_STAT.ACBI[0:0] or CLSi_ATOM_CHx_CTRL_SOMI.ACB0 bit.

The update of the output ATOM_OUT of instance i and channel x depends on the bit CLSi_ATOM_CHx_CTRL_SOMI.ACB0 and the bit CLSi_ATOM_CHx_CTRL_SOMI.SL. The bit CLSi_ATOM_CHx_STAT.ACBI[0:0] has no effect in this case.

The initial **ATOM** channel port pin ATOM_OUT signal level has to be specified by the CLSi_ATOM_CHx_CTRL_SOMI.SL bit when CLSi_ATOM_AGC_OUTEN_CTRL.OUTEN_CTRLk (k=0-7) (x = k) is disabled (see chapter CLSi_ATOM_AGC_OUTEN_CTRL) for details.

In SOMI mode the output behavior depends on the CLSi_ATOM_CHx_CTRL_SOMI.SL bit and the bit CLSi_ATOM_CHx_STAT.ACBI[0:0] or the bit CLSi_ATOM_CHx_CTRL_SOMI.ACB0:

Table 2226 Output behavior in SOMI mode

CLSi_ATOM_CHx_CTRL_SOMI.SL	CLSi_ATOM_CHx_STAT.ACBI**[0:0]**/ CLSi_ATOM_CHx_CTRL_SOMI.ACB0	**Output behavior**
0	0	Set output to inverse of CLSi_ATOM_CHx_CTRL_SOMI.SL (1)
0	1	Set output to CLSi_ATOM_CHx_CTRL_SOMI.SL (0)
1	0	Set output to inverse of CLSi_ATOM_CHx_CTRL_SOMI.SL (0)
1	1	Set output to CLSi_ATOM_CHx_CTRL_SOMI.SL (1)

The signal level bit CLSi_ATOM_CHx_STAT.ACBI[0:0] is transferred to the SOU sub-unit of the **ATOM** and made visible at the output port according to the table above immediately after the data was received by the ACI.

ATOM signal output mode compare (SOMC)

Overview

In **ATOM** Signal Output Mode Compare (SOMC) the output action is performed depending on the comparison between input values located in CLSi_ATOM_CHx_CM0.CM0 and/or CLSi_ATOM_CHx_CM1.CM1 registers and the two (three) time base values CCM[i]_TBU_TS0 or CCM[i]_TBU_TS1 (or CCM[i]_TBU_TS2) provided by the **TBU**. For a description of the time base generation please refer to the **TBU** specification in Time base unit (**TBU**

). It is configurable, which of the two (three) time bases is to be compared with one or both values in CLSi_ATOM_CHx_CM0.CM0 and CLSi_ATOM_CHx_CM1.CM1.

The behavior of the two compare units CCU0 and CCU1 is controlled with the bits CLSi_ATOM_CHx_CTRL_SOMC.ACB42.

Based on the configured compare strategy CLSi_ATOM_CHx_CTRL_SOMC.ACB42 or CLSi_ATOM_CHx_STAT.ACBI[4:2], CCU0 unit always performs cyclic event compare between CLSi_ATOM_CHx_CM0.CM0 and the selected **TBU** time base CCM[i]_TBU_TS0/CCM[i]_TBU_TS1/ CCM[i]_TBU_TS2 while CCU1 unit always performs cyclic event compare between CLSi_ATOM_CHx_CM1.CM1 and the selected **TBU** time base CCM[i]_TBU_TS0/CCM[i]_TBU_TS1/CCM[i]_TBU_TS2. Please refer to Cyclic event compare for more information about cyclic event compare strategy. When the selected time base is "cyclically greater than or equal to" the compare value of CLSi_ATOM_CHx_CM0.CM0, CCU0 compare match event is considered to have occurred. When the selected time base is "cyclically greater than or equal to" the compare value of CLSi_ATOM_CHx_CM1.CM1, CCU1 compare match event is considered to have occurred. Whenever CCU0 or CCU1 compare match event has occurred, an edge is generated on output ATOM_OUT[x:x], depending on the predefined signal level in CLSi_ATOM_CHx_CTRL_SOMC.SL bit in combination with two control bits CLSi_ATOM_CHx_CTRL_SOMC.ACB10 or CLSi_ATOM_CHx_STAT.ACBI[1:0].

In SOMC mode the channel is always disabled after the specified compare match event has occurred. The shadow registers are used to store two timestamp values at the match time. The channel compare can be re-enabled by first reading the shadow registers and by providing new data for CLSi_ATOM_CHx_CM0.CM0/ CLSi_ATOM_CHx_CM1.CM1 registers. For a detailed description please refer to the SOMC mode controlled by configuration interface.

If three time bases exist for the **eGTM** there must be a preselection between CCM[i]_TBU_TS1 and CCM[i]_TBU_TS2 for the **ATOM** channel. This can be done with CLSi_ATOM_CHx_CTRL_SOMC.TB12_SEL bit.

If CCM[i]_TBU_TS1/CCM[i]_TBU_TS2 is selected for cyclic event compare in CCU0 or CCU1 unit, CLSi_ATOM_CHx_CTRL_SOMC.CMP_CTRL must be configured to assure correct cyclic event compare. If the selected **TBU** time base CCM[i]_TBU_TS1/CCM[i]_TBU_TS2 performs up-counting, CLSi_ATOM_CHx_CTRL_SOMC.CMP_CTRL must be configured as 0. If the selected **TBU** time base CCM[i]_TBU_TS1/CCM[i]_TBU_TS2 performs down-counting, CLSi_ATOM_CHx_CTRL_SOMC.CMP_CTRL must be configured as 1. The bit CLSi_ATOM_CHx_CTRL_SOMC.CMP_CTRL has no impact on cyclic event compare against the **TBU** time base CCM[i]_TBU_TS0.

When configured in SOMC mode, the channel port pin has to be initialized to an initial signal level. This initial level after enabling the **ATOM** channel is determined by the CLSi_ATOM_CHx_CTRL_SOMC.SL bit. If the output is disabled, the signal level is set to the inverse level of the CLSi_ATOM_CHx_CTRL_SOMC.SL bit.

If the channel is disabled and the output is still enabled with CLSi_ATOM_CHx_CTRL_SOMC.FREEZE=0, the register SOUR and the output ATOM_OUT[x:x]is set to the CLSi_ATOM_CHx_CTRL_SOMC.SL bit.

On a compare match event the shadow register CLSi_ATOM_CHx_SR0.SR0 and CLSi_ATOM_CHx_SR1.SR1 are used to capture the **TBU** timestamp values. CLSi_ATOM_CHx_SR0.SR0 always holds CCM[i]_TBU_TS0 and CLSi_ATOM_CHx_SR1.SR1 either holds CCM[i]_TBU_TS1 or CCM[i]_TBU_TS2 depending on the CLSi_ATOM_CHx_CTRL_SOMC.TB12_SEL bit.

When the channel is disabled and the compare registers are written, the compare registers CLSi_ATOM_CHx_CM0.CM0/CLSi_ATOM_CHx_CM1.CM1 are loaded with the written value and the channel starts with the comparison against these values, when the channel is enabled.

SOMC mode controlled by configuration interface

The output of the **ATOM** channel is set at a compare match event depending on the CLSi_ATOM_CHx_CTRL_SOMC.ACB10 in combination with the CLSi_ATOM_CHx_CTRL_SOMC.SL bit. The comparison is performed according to different compare strategy specified by CLSi_ATOM_CHx_CTRL_SOMC.ACB42. The output behavior according to CLSi_ATOM_CHx_CTRL_SOMC.ACB10 bit field in the control register is shown in the following table:

Table 2227 Output behavior according to the CLSi_ATOM_CHx_CTRL_SOMC.ACB10 bit field in the control register

CLSi_ATOM_CHx_CTRL.SL	CLSi_ATOM_CHx_CTRL_SOMC.ACB10[1:1]	CLSi_ATOM_CHx_CTRL_SOMC.ACB10[0:0]	Output behavior
0	0	0	No signal level change at output (exception in Table 2230 mode CLSi_ATOM_CHx_CTRL_SOMC.ACB42=001_B)
0	0	1	Set output signal level to 1 (exception in Table 2230 mode CLSi_ATOM_CHx_CTRL_SOMC.ACB42=001_B)
0	1	0	Set output signal level to 0 (exception in Table 2230 mode CLSi_ATOM_CHx_CTRL_SOMC.ACB42=001_B)
0	1	1	Toggle output signal level (exception in Table 2230 mode CLSi_ATOM_CHx_CTRL_SOMC.ACB42=001_B)
1	0	0	No signal level change at output (exception in Table 2230 mode CLSi_ATOM_CHx_CTRL_SOMC.ACB42=001_B)
1	0	1	Set output signal level to 0 (exception in Table 2230 mode CLSi_ATOM_CHx_CTRL_SOMC.ACB42=001_B)
1	1	0	Set output signal level to 1 (exception in Table 2230 mode CLSi_ATOM_CHx_CTRL_SOMC.ACB42=001_B)
1	1	1	Toggle output signal level (exception in Table 2230 mode CLSi_ATOM_CHx_CTRL_SOMC.ACB42 =001_B)

The capture/compare strategy of the two units CCU0 and CCU1 can be controlled with the CLSi_ATOM_CHx_CTRL_SOMC.ACB42 bit field. The CCU0 unit always compares CLSi_ATOM_CHx_CM0.CM0 with the selected **TBU** time base. The CCU1 unit always compares CLSi_ATOM_CHx_CM1.CM1 with the selected **TBU**

time base. Different CCU0/CCU1 comparison strategies controlled by CLSi_ATOM_CHx_CTRL_SOMC.ACB42 are shown in the following table:

Table 2228 Capture/compare strategy of the two CCUx units controlled by CLSi_ATOM_CHx_CTRL_SOMC.ACB42 bit field

CLSi_ATOM_CHx_CTRL_SOMC.ACB42[2:2]	CLSi_ATOM_CHx_CTRL_SOMC.ACB42[1:1]	CLSi_ATOM_CHx_CTRL_SOMC.ACB42[0:0]	CCU0/CCU1 comparison control
0	0	0	Serve First: Compare in CCU0 using CCM[i]_TBU_TS0 and in parallel in CCU1 using CCM[i]_TBU_TS1 or CCM[i]_TBU_TS2. Disable the other comparison at one compare match event. Output signal level at the compare match event is defined by combination of CLSi_ATOM_CHx_CTRL_SOMC.SL and CLSi_ATOM_CHx_CTRL_SOMC.ACB10. Details see Table 2229.
0	0	1	Serve First: Compare in CCU0 using CCM[i]_TBU_TS0 and in parallel in CCU1 using CCM[i]_TBU_TS1 or CCM[i]_TBU_TS2. Disable the other comparison at one compare match event. Output signal level at the compare match event is defined by combination of CLSi_ATOM_CHx_CTRL_SOMC.SL and CLSi_ATOM_CHx_CTRL_SOMC.ACB10. Details see Table 2230.

(table continues...)

Table 2228 (continued) Capture/compare strategy of the two CCUx units controlled by CLSi_ATOM_CHx_CTRL_SOMC.ACB42 bit field

CLSi_ATOM_CHx_CTRL_SOMC.ACB42[2:2]	CLSi_ATOM_CHx_CTRL_SOMC.ACB42[1:1]	CLSi_ATOM_CHx_CTRL_SOMC.ACB42[0:0]	CCU0/CCU1 comparison control
0	1	0	Compare in CCU0 only, use time base CCM[i]_TBU_TS0. Output signal level at the compare match event is defined by combination of CLSi_ATOM_CHx_CTRL_SOMC.SL and CLSi_ATOM_CHx_CTRL_SOMC.ACB10 (details see Table 2227).
0	1	1	Compare in CCU1 only, use time base CCM[i]_TBU_TS1 or CCM[i]_TBU_TS2. Output signal level at the compare match event is defined by combination of CLSi_ATOM_CHx_CTRL_SOMC.SL and CLSi_ATOM_CHx_CTRL_SOMC.ACB10 (details see Table 2227).
1	0	0	Serve Last: Compare in CCU0 and then in CCU1 using CCM[i]_TBU_TS0. Output signal level at the CCU0 compare match event is defined by combination of CLSi_ATOM_CHx_CTRL_SOMC.SL and CLSi_ATOM_CHx_CTRL_SOMC.ACB10 (details see Table 2227). At the CCU1 compare match event the output level is toggled.

(table continues...)

Table 2228 (continued) Capture/compare strategy of the two CCUx units controlled by CLSi_ATOM_CHx_CTRL_SOMC.ACB42 bit field

CLSi_ATOM_CHx_CTRL_SOMC.ACB42[2:2]	CLSi_ATOM_CHx_CTRL_SOMC.ACB42[1:1]	CLSi_ATOM_CHx_CTRL_SOMC.ACB42[0:0]	CCU0/CCU1 comparison control
1	0	1	Serve Last: Compare in CCU0 and then in CCU1 using CCM[i]_TBU_TS1 or CCM[i]_TBU_TS2. Output signal level at the CCU0 compare match event is defined by combination of CLSi_ATOM_CHx_CTRL_SOMC.SL and CLSi_ATOM_CHx_CTRL_SOMC.ACB10 (details see Table 2227). At the CCU1 compare match event the output level is toggled.
1	1	0	Serve Last: Compare in CCU0 using CCM[i]_TBU_TS0 and then in CCU1 using CCM[i]_TBU_TS1 or CCM[i]_TBU_TS2. Output signal level at the CCU1 compare match event is defined by combination of CLSi_ATOM_CHx_CTRL_SOMC.SL and CLSi_ATOM_CHx_CTRL_SOMC.ACB10 (details see Table 2227).
1	1	1	Cancels pending comparison

The behavior of the CLSi_ATOM_CHx_STAT.ACBI[4:2]/CLSi_ATOM_CHx_CTRL_SOMC.ACB42 bit combinations 000_B and 001_B is described in more detail in Table 2229 and Table 2230.

Table 2229 ATOM serve first definition CLSi_ATOM_CHx_CTRL_SOMC.ACB42 = 000_B or CLSi_ATOM_CHx_STAT.ACBI = 000_B

CLSi_ATOM_CHx_CTRL_SOMC.ACB42[2:2] / CLSi_ATOM_CHx_STAT.ACBI[4:4]	CLSi_ATOM_CHx_CTRL_SOMC.ACB42[1:1] / CLSi_ATOM_CHx_STAT.ACBI[3:3]	CLSi_ATOM_CHx_CTRL_SOMC.ACB42[0:0] / CLSi_ATOM_CHx_STAT.ACBI[2:2]	CLSi_ATOM_CHx_CTRL_SOMC.ACB10[1:1] / CLSi_ATOM_CHx_STAT.ACBI[1:1]	CLSi_ATOM_CHx_CTRL_SOMC.ACB10[0:0] / CLSi_ATOM_CHx_STAT.ACBI[0:0]	CLSi_ATOM_CHx_CTRL_SOMC.SL	CCU0 match	CCU1 match	Pin level new
0	0	0	0	0	0	0	1	hold
0	0	0	0	0	0	1	0	hold
0	0	0	0	0	0	1	1	hold
0	0	0	0	1	0	0	1	1
0	0	0	0	1	0	1	0	1
0	0	0	0	1	0	1	1	1
0	0	0	1	0	0	0	1	0
0	0	0	1	0	0	1	0	0
0	0	0	1	0	0	1	1	0
0	0	0	1	1	0	0	1	toggle
0	0	0	1	1	0	1	0	toggle
0	0	0	1	1	0	1	1	toggle
0	0	0	0	0	1	0	1	hold
0	0	0	0	0	1	1	0	hold
0	0	0	0	0	1	1	1	hold
0	0	0	0	1	1	0	1	0
0	0	0	0	1	1	1	0	0
0	0	0	0	1	1	1	1	0
0	0	0	1	0	1	0	1	1
0	0	0	1	0	1	1	0	1
0	0	0	1	0	1	1	1	1
0	0	0	1	1	1	0	1	toggle
0	0	0	1	1	1	1	0	toggle
0	0	0	1	1	1	1	1	toggle

Table 2230 ATOM serve first definition CLSi_ATOM_CHx_CTRL_SOMC.ACB42 = 001_B or CLSi_ATOM_CHx_STAT.ACBI = 001_B

CLSi_ATOM_CHx_CTRL_SOMC.ACB42[2:2] / CLSi_ATOM_CHx_STAT.ACBI[4:4]	CLSi_ATOM_CHx_CTRL_SOMC.ACB42[1:1] / CLSi_ATOM_CHx_STAT.ACBI[3:3]	CLSi_ATOM_CHx_CTRL_SOMC.ACB42[0:0] / CLSi_ATOM_CHx_STAT.ACBI[2:2]	CLSi_ATOM_CHx_CTRL_SOMC.ACB10[1:1] / CLSi_ATOM_CHx_STAT.ACBI[1:1]	CLSi_ATOM_CHx_CTRL_SOMC.ACB10[0:0] / CLSi_ATOM_CHx_STAT.ACBI[0:0]	CLSi_ATOM_CHx_CTRL_SOMC.SL	CCU0 match	CCU1 match	Pin level new
0	0	1	0	0	0	0	1	hold
0	0	1	0	0	0	1	0	toggle
0	0	1	0	0	0	1	1	hold
0	0	1	0	1	0	0	1	0
0	0	1	0	1	0	1	0	1
0	0	1	0	1	0	1	1	0
0	0	1	1	0	0	0	1	1
0	0	1	1	0	0	1	0	0
0	0	1	1	0	0	1	1	1
0	0	1	1	1	0	0	1	toggle
0	0	1	1	1	0	1	0	hold
0	0	1	1	1	0	1	1	toggle
0	0	1	0	0	1	0	1	hold
0	0	1	0	0	1	1	0	toggle
0	0	1	0	0	1	1	1	hold
0	0	1	0	1	1	0	1	1
0	0	1	0	1	1	1	0	0
0	0	1	0	1	1	1	1	1
0	0	1	1	0	1	0	1	0
0	0	1	1	0	1	1	0	1
0	0	1	1	0	1	1	1	0
0	0	1	1	1	1	0	1	toggle
0	0	1	1	1	1	1	0	hold
0	0	1	1	1	1	1	1	toggle

If the **ATOM** channel is enabled, the CLSi_ATOM_CHx_CM0.CM0 and/or CLSi_ATOM_CHx_CM1.CM1 registers and the CLSi_ATOM_CHx_CTRL_SOMC.ACB42 bit field can be updated over the configuration interface as long as the first match event occurs in case of a 'serve first' compare strategy or as long as the overall match event occurs in case of the other compare strategies.

After a compare match event that causes an update of the shadow registers CLSi_ATOM_CHx_SR0.SR0/ CLSi_ATOM_CHx_SR1.SR1 and before reading the CLSi_ATOM_CHx_SR0.SR0 and/or CLSi_ATOM_CHx_SR1.SR1 register, the update of the registers CLSi_ATOM_CHx_CM0.CM0 and/or CLSi_ATOM_CHx_CM1.CM1 is possible but has no effect.

To set up a new compare action, first the CLSi_ATOM_CHx_SR0.SR0 and/or CLSi_ATOM_CHx_SR1.SR1 register containing captured values have to be read and then new compare values have to be written into the register CLSi_ATOM_CHx_CM0.CM0 and/or CLSi_ATOM_CHx_CM1.CM1.

Which register CLSi_ATOM_CHx_CM0.CM0 or CLSi_ATOM_CHx_CM1.CM1 has to be updated depends on the compare strategy defined in the CLSi_ATOM_CHx_CTRL_SOMC.ACB42 bit field of the channel control register. Since the channel immediately starts with the comparison after the CLSi_ATOM_CHx_CM0.CM0/ CLSi_ATOM_CHx_CM1.CM1 register was/were written, the compare strategy has to be updated before the CLSi_ATOM_CHx_CM0.CM0/CLSi_ATOM_CHx_CM1.CM1 registers are written.

For the 'serve last' compare strategies, if the register CLSi_ATOM_CHx_CM0.CM0 and CLSi_ATOM_CHx_CM1.CM1 are updated, it can happen that one or both compare values are already located in the past. In any case, the **ATOM** channel will first wait until both compare values are written before it starts the time base comparisons to avoid a deadlock.

The configuration interface can be used to check at any time, whether at least one of the capture compare registers of the **ATOM** channel contains valid data, and to wait for a compare event to occur. This is signaled by the CLSi_ATOM_CHx_STAT.DV bit inside the CLSi_ATOM_CHx_STAT register.

***Note**: For 'serve last' compare strategies, if CLSi_ATOM_CHx_STAT.DV bit is currently not set, writing to CLSi_ATOM_CHx_CM0.CM0 or CLSi_ATOM_CHx_CM1.CM1 immediately sets the CLSi_ATOM_CHx_STAT.DV bit although the compare is only started if both values are written.*

An exception for update of register CLSi_ATOM_CHx_CM0.CM0/CLSi_ATOM_CHx_CM1.CM1 exists in SOMC mode and CCUx control mode 'serve last'. If in this mode the CCU0 compare match event occurred, the update of register CLSi_ATOM_CHx_CM0.CM0/CLSi_ATOM_CHx_CM1.CM1 over the configuration interface is blocked until the CCU1 compare match event.

In the 'serve last' mode (CLSi_ATOM_CHx_CTRL_SOMC.ACB42= 100_B or CLSi_ATOM_CHx_CTRL_SOMC.ACB42=101_B) it is possible to generate very small spikes on the output pin by loading CLSi_ATOM_CHx_CM0.CM0 and CLSi_ATOM_CHx_CM1.CM1 with two timestamp values for CCM[i]_TBU_TS0, CCM[i]_TBU_TS1 or CCM[i]_TBU_TS2 close together. The output pin will then be set or reset depending on the CLSi_ATOM_CHx_CTRL_SOMC.SL bit and CLSi_ATOM_CHx_CTRL_SOMC.ACB10 bit field on the first match event and the output will toggle on the second compare event in the CCU1 compare unit.

***Note**: Since the CCU0 will enable the CCU1 once it reaches its comparison timestamp, the bigger (smaller) timestamp has to be loaded into the CLSi_ATOM_CHx_CM1.CM1 register. The order of the comparison timestamps depends on the defined greater-equal or less-equal comparison of the CCU0/CCU1 units.*

In addition to storing the captured timestamps in the shadow registers, the **ATOM** channel provides the result of the compare match event in the CLSi_ATOM_CHx_STAT.ACBO[4:4] and CLSi_ATOM_CHx_STAT.ACBO[3:3] bits. The meaning of the bits is shown in the following table:

Table 2231 Compare match event CLSi_ATOM_CHx_STAT.ACBO[4:4] and CLSi_ATOM_CHx_STAT.ACBO[3:3] bits of

CLSi_ATOM_CHx_STAT.ACBO[4:4]	CLSi_ATOM_CHx_STAT.ACBO[3:3]	Indication
0	1	CCU0 compare match occurred
1	0	CCU1 compare match occurred

The CLSi_ATOM_CHx_STAT.ACBO bit field is reset, when the CLSi_ATOM_CHx_STAT.DV bit is set.

Depending on the capture compare unit where the compare match event occurred, the interrupt ATOM_CCU0TC[x]_IRQ or ATOM_CCU1TC[x]_IRQ is raised.

In case of 'serve first' compare strategy, if both events CCU0 and CCU1 occur at the same point in time, both interrupts will be raised and CLSi_ATOM_CHx_STAT.ACBO will be 10_B due to higher priority.

The behavior of an **ATOM** channel in SOMC mode with CLSi_ATOM_CHx_CTRL_SOMC.EUPM=0 controlled via configuration interface is depicted in Figure 1078.

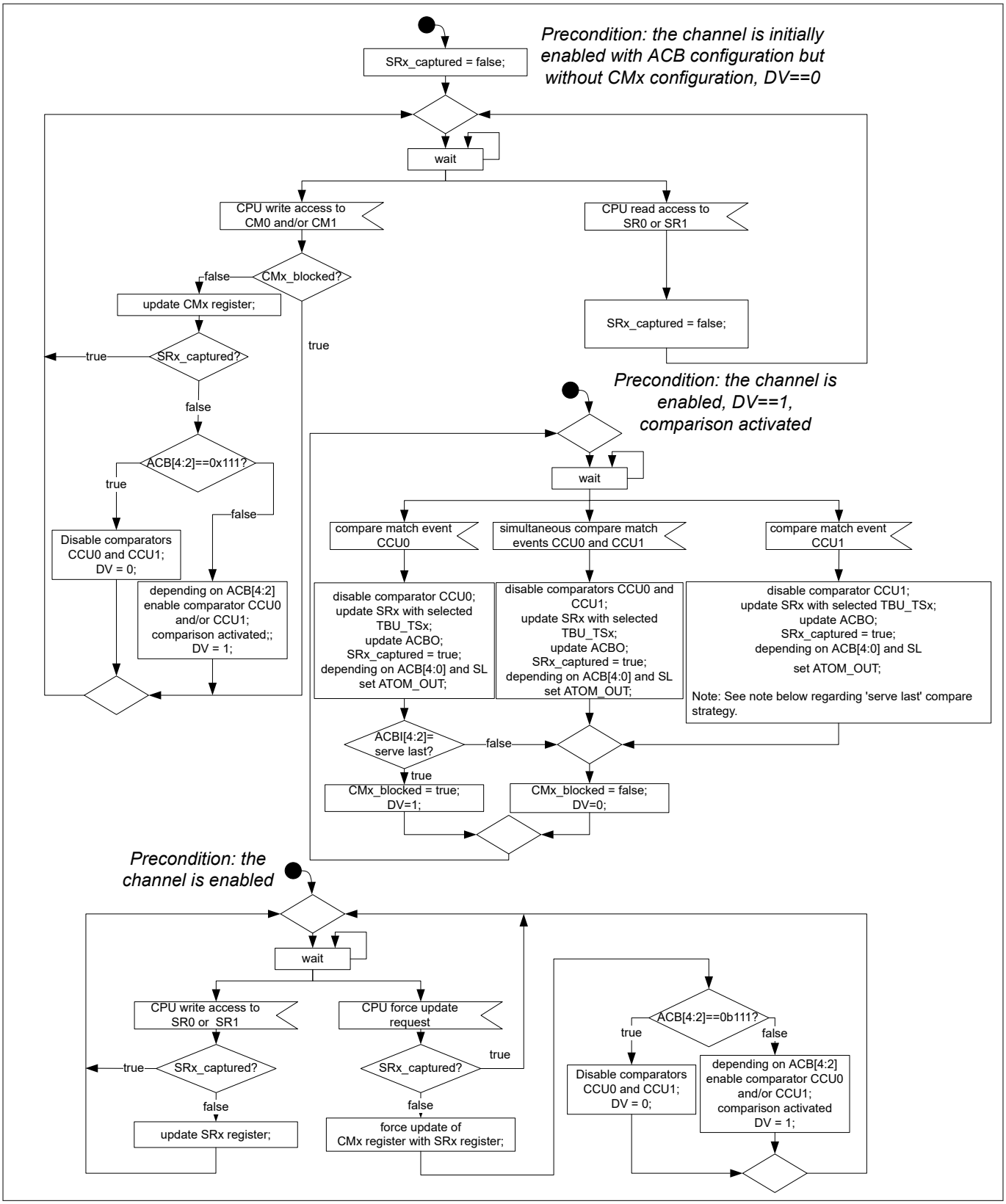


Figure 1078 SOMC state diagram for channel controlled via configuration interface if CLSi_ATOM_CHx_CTRL_SOMC.EUPM=0

***Note**: In case of 'serve last' compare strategy the compare match in CCU0 is expected before the compare match in CCU1. But if the CCU1 compare match occurs first, CLSi_ATOM_CHx_SR0.SR0, CLSi_ATOM_CHx_SR1.SR1 as well as CLSi_ATOM_CHx_STAT.ACBO will not be updated and also the output signal will not be set. After first occurance of CCU0 match and afterwards a CCU1 match the ordering is as expected and captures and signal level output changes will be performed on both matches.*

ATOM signal output mode PWM (SOMP)

In **ATOM** Signal Output Mode PWM (SOMP), the **ATOM** sub-module channel is able to generate complex PWM signals with different pulse widths and periods by configuring different operation registers CLSi_ATOM_CHx_CM0.CM0, CLSi_ATOM_CHx_CM1.CM1 and the control register CLSi_ATOM_CHx_CTRL_SOMP. **ATOM** SOMP mode has almost the same functions as the **TOM** module.

The signal level of the pulse generated inside the period can be configured inside the channel control register (CLSi_ATOM_CHx_CTRL_SOMP.SL). The initial signal output level for the channel is the inverse pulse level defined by the CLSi_ATOM_CHx_CTRL_SOMP.SL bit. Figure 1081 depicts this behavior.

The counter CLSi_ATOM_CHx_CN0.CN0 of each channel can run in two different modes depending on configuration of CLSi_ATOM_CHx_CTRL_SOMP.UDMODE.

By default, the counter counts only up until it reaches a specified value and it is then reset to 0.

In the up down counter mode CLSi_ATOM_CHx_CN0.CN0 switches between counting up and counting down.

Like **TOM** module, in SOMP mode, each **ATOM** channel can further work in one-shot mode by configuring CLSi_ATOM_CHx_CTRL_SOMP.OSM=1. Pulse count modulation mode and high-resolution mode are also available depending on the configuration of the control register CLSi_ATOM_CHx_CTRL_SOMP.

Pulse width, period, signal level and clock frequency update mechanisms

Since **ATOM** shows a behavior similar to **TOM** module in SOMP mode, their update mechanisms of pulse width, period, signal level and clock frequency are also almost same. The two operation registers CLSi_ATOM_CHx_CM0.CM0 and CLSi_ATOM_CHx_CM1.CM1 can be updated with the content of the shadow register CLSi_ATOM_CHx_SR0.SR0 and CLSi_ATOM_CHx_SR1.SR1. The register CLSi_ATOM_CHx_CTRL_SOMP.CLK_SRC that determines the clock resolution of the counter CLSi_ATOM_CHx_CN0.CN0 can be updated with its shadow register CLSi_ATOM_CHx_CTRL_SR.CLK_SRC_SR. The signal level defined in bit field CLSi_ATOM_CHx_CTRL_SOMP.SL can be updated with its shadow register CLSi_ATOM_CHx_CTRL_SR.SL_SR.

The update of the configuration register CLSi_ATOM_CHx_CM0.CM0, CLSi_ATOM_CHx_CM1.CM1, CLSi_ATOM_CHx_CTRL_SOMP.SL and CLSi_ATOM_CHx_CTRL_SOMP.CLK_SRC with the content of their shadow registers can be performed under the following conditions:

- In case CLSi_ATOM_CHx_CTRL_SOMP.RST_CCU0 =0_B when the specified CLSi_ATOM_CHx_CTRL_SOMP.UDMODE compare condition is fulfilled the update will be performed. Additional information is given in the bit field description of CLSi_ATOM_CHx_CTRL_SOMP.UDMODE. The update mechanism must be enabled by configuring CLSi_ATOM_AGC_GLB_CTRL.UPEN_CTRLk (k=0-7) =10_B.
- In case CLSi_ATOM_CHx_CTRL_SOMP.RST_CCU0 =1_B when the channel [x] is triggered by the signal ATOM_CH_TRIGIN*[x:x]* from another **ATOM** channel or by the external trigger signal EXT_TRIGIN of the assigned **TIM** channel, after it is synchronized to the selected CCM[i]_CLK_RES of channel [x], the update will also be performed. The update mechanism must be enabled by configuring CLSi_ATOM_AGC_GLB_CTRL.UPEN_CTRLk (k=0-7) =10_B.
- These configuration registers can be force-updated via a trigger signal via ATOM_FUPD or via EXT_TRIGIN.
- In SOMP mode, if CLSi_ATOM_CHx_CTRL_SOMP.RST_CCU0 =0_B, the operating register is continuously updated as long as CLSi_ATOM_CHx_CM0.CM0 =0_H. This update condition must be enabled by configuring

CLSi_ATOM_AGC_GLB_CTRL.UPEN_CTRLk (k=0-7) =10_B regardless of the source for the shadow registers. The source can be the configuration interface or the **ARU**.

- When the stop condition of the clock is configured (CLSi_ATOM_CHx_CTRL_SOMP.CLK_SRC =1100_B), new data is received via the **ARU** interface, until the clock stop condition is released. This update case does not depend on the state of CLSi_ATOM_AGC_GLB_CTRL.UPEN_CTRLk (k=0-7).

Based on the update mechanisms above, these registers CLSi_ATOM_CHx_CM0.CM0, CLSi_ATOM_CHx_CM1.CM1, CLSi_ATOM_CHx_CTRL_SOMP.SL, CLSi_ATOM_CHx_CTRL_SOMP.CLK_SRC can be updated synchronously and asynchronously. Synchronous update means that the pulse width, period duration, signal level and the counter operating clock frequency can be changed at the end of the running PWM period. Asynchronous update means that the pulse width, period duration, signal level and the counter operating clock frequency can also be changed during the actual running PWM period. The following two sections provide examples of synchronous update and asynchronous update of pulse width in continuous up mode with CLSi_ATOM_CHx_CTRL_SOMP.RST_CCU0 = 0.

An update of pulse width, period and the counter clock frequency becoming effective synchronously with start of a new period can easily be reached by performing the below steps:

1. Disable the update of the action register with the content of the corresponding shadow register by setting the channel-specific configuration bit field CLSi_ATOM_AGC_GLB_CTRL.UPEN_CTRLk (k=0-7) **(k=x)** = 01_B. (Configuring CLSi_ATOM_CHx_CTRL_SOMP.UDMODE=01_B if in counting up-down mode.)
2. Write new desired values to CLSi_ATOM_CHx_SR0.SR0, CLSi_ATOM_CHx_SR1.SR1, CLSi_ATOM_CHx_CTRL_SR.SL_SR, CLSi_ATOM_CHx_CTRL_SR.CLK_SRC_SR.
3. Enable update of the action register by setting the channel-specific configuration bit field CLSi_ATOM_AGC_GLB_CTRL.UPEN_CTRLk (k=0-7) **(k=x)** = 10_B.

Asynchronous update of the operation registers can be realized by hot reconfiguring directly the registers CLSi_ATOM_CHx_CM0.CM0, CLSi_ATOM_CHx_CM1.CM1, CLSi_ATOM_CHx_CTRL_SOMP.SL, CLSi_ATOM_CHx_CTRL_SOMP.CLK_SRC or hot reconfiguring their shadow registers and then using the force update mechanism. Such asynchronous update mechanism is sometimes helpful to change the PWM parameters of the current running PWM signal. For example, in continuous counting up mode, the current running PWM period can be reduced by asynchronously updating the value of CLSi_ATOM_CHx_CM0.CM0 with a smaller value before the end of the current period. However, since the behavior depends on the access or update time of registers and the current running operation phase of the channel, which is often difficult to control, then the behavior after an asynchronous update can be unpredictable and unexpected. Therefore, such an asynchronous update must be carefully performed in real application. The following steps are recommended to apply the force update mechanism on channel x:

1. Disable the update of the action register with the content of the corresponding shadow register by setting the channel-specific configuration bit field CLSi_ATOM_AGC_GLB_CTRL.UPEN_CTRLk (k=0-7) **(k=x)** = 01_B.
2. Write new desired values to CLSi_ATOM_CHx_SR0.SR0, CLSi_ATOM_CHx_SR1.SR1, CLSi_ATOM_CHx_CTRL_SR.SL_SR, CLSi_ATOM_CHx_CTRL_SR.CLK_SRC_SR.
3. Enable the force update of the operation registers by setting the channel-specific configuration bit field CLSi_ATOM_AGC_FUPD_CTRL.FUPD_CTRLk (k=0-7) **(k=x)** = 10_B, CLSi_ATOM_AGC_FUPD_CTRL.RSTCN0_CHk (k=0-7) **(k=x)** = 01_B.
4. Write CLSi_ATOM_AGC_GLB_CTRL.HOST_TRIG = 1 to trigger the force update.

Synchronous update of pulse width only

A synchronous update of the pulse width only can be done by simply writing the desired new value to register CLSi_ATOM_CHx_SR1.SR1 without preceding disable of the update mechanism (as described in the chapter above). As an example in SOMP continuous counting up mode with CLSi_ATOM_CHx_CTRL.RST_CCU0=0 is shown in Figure 1079, the new pulse width is then applied in the period following the period where the update of register CLSi_ATOM_CHx_SR1.SR1 is done.

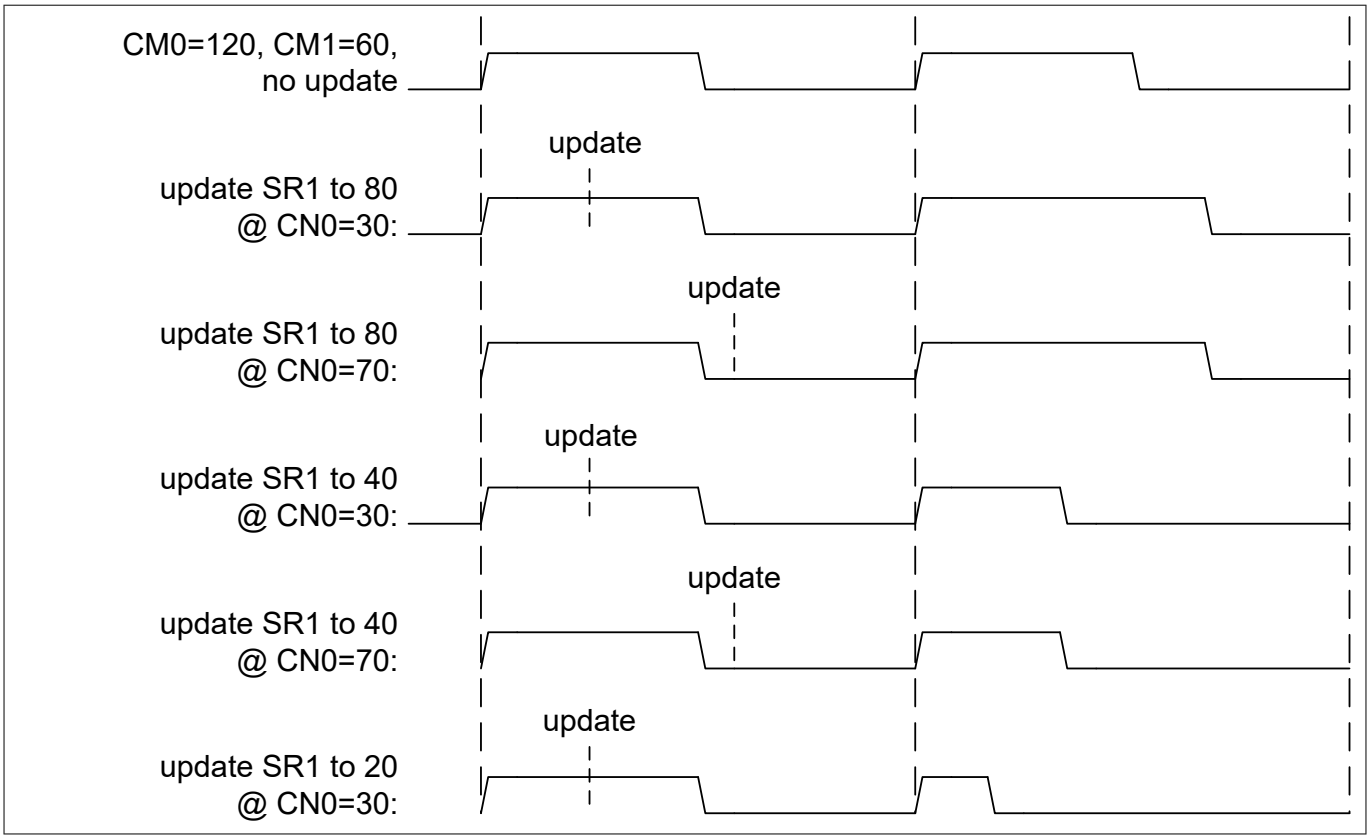


Figure 1079 Synchronous update of pulse width only in SOMP continuous counting up mode with CLSi_ATOM_CHx_CTRL.RST_CCU0=0

Asynchronous update of pulse width only

If the update of the pulse width should be performed independent of the start of a new period (asynchronous), it is mandatory to additionally disable the synchronous update mechanism as a whole (i.e. configuring bits CLSi_ATOM_AGC_GLB_CTRL.UPEN_CTRLk (k=0-7) **(k=x)**=01_B of corresponding channel x) and then update the operation register. On the one hand, the desired new value can be written directly to register CLSi_ATOM_CHx_CM1.CM1. On the other hand, it can be realized by writing the new value to CLSi_ATOM_CHx_SR1.SR1 and afterwards updating the value of CLSi_ATOM_CHx_CM1.CM1 by using the asynchronous force update mechanism (configuring CLSi_ATOM_AGC_FUPD_CTRL.FUPD_CTRLk (k=0-7) **(k=x)** = 10_B, CLSi_ATOM_AGC_FUPD_CTRL.RSTCN0_CHk (k=0-7) **(k=x)** = 01_B and then writing bit CLSi_ATOM_AGC_GLB_CTRL.HOST_TRIG = 1).

Depending on the point of time of the update of CLSi_ATOM_CHx_CM1.CM1 in relation to the actual value of CLSi_ATOM_CHx_CN0.CN0 and CLSi_ATOM_CHx_CM1.CM1, the new pulse width is applied in the current period or the following period. That means asynchronous update of CLSi_ATOM_CHx_CM1.CM1 in the current period can at least change the pulse width in the next PWM signal. The behavior of the output signal due to the different possibilities of an asynchronous update during a PWM period in SOMP continuous counting up mode with CLSi_ATOM_CHx_CTRL.RST_CCU0=0 is shown in Figure 1080. The new pulse width may jitter from update to update by maximum one period (given by CLSi_ATOM_CHx_CM0.CM0). However, the period remains unchanged. On an asynchronous update, it is guaranteed, that no spike occurs at the output port of the channel due to a too late update of the operation registers.

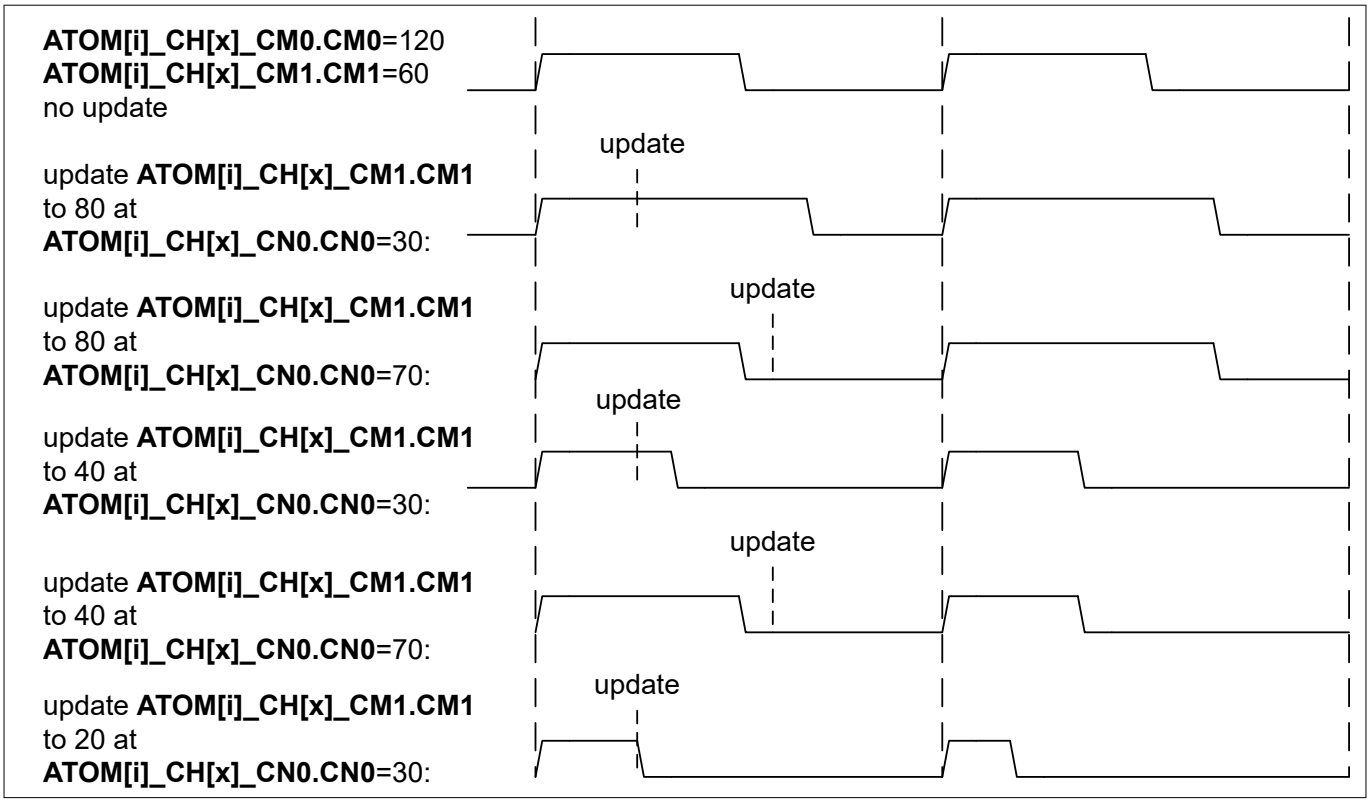


Figure 1080 Asynchronous update of pulse width only in SOMP continuous counting up mode with CLSi_ATOM_CHx_CTRL.RST_CCU0=0

Update controlled by configuration interface

Update of operation registers CLSi_ATOM_CHx_CM0.CM0, CLSi_ATOM_CHx_CM1.CM1, CLSi_ATOM_CHx_CTRL_SOMP.SL and CLSi_ATOM_CHx_CTRL_SOMP.CLK_SRC can also be controlled by the **AEI** bus interface. On the one hand, they can be synchronously updated from their shadow registers CLSi_ATOM_CHx_SR0.SR0, CLSi_ATOM_CHx_SR1.SR1, CLSi_ATOM_CHx_CTRL_SR.SL_SR and CLSi_ATOM_CHx_CTRL_SR.CLK_SRC_SR when ATOM_UPEN[x:x] = 1 in the AGC sub-unit. Normally, synchronous update happens at the end of PWM period when the counter is reset. On the other hand, they can be asynchronously updated from the shadow registers triggered by ATOM_FUPD[x:x] signal in AGC sub-unit. Furthermore, CLSi_ATOM_CHx_CM0.CM0, CLSi_ATOM_CHx_CM1.CM1 and CLSi_ATOM_CHx_CTRL_SOMP.SL can be directly asynchronously written via the configuration interface. Such an asynchronous update happens during one PWM period that should be carefully applied to avoid unexpected and unpredicted behavior. For asynchronous update, the synchronous update mechanism has to be locked via the CLSi_ATOM_AGC_GLB_CTRL register with the ATOM_UPEN[x:x] signal in the AGC sub-unit.

On a compare match of CLSi_ATOM_CHx_CN0.CN0 and CLSi_ATOM_CHx_CM0.CM0 or CLSi_ATOM_CHx_CM1.CM1, the output signal level of ATOM_OUT is toggled according to the signal level output bit CLSi_ATOM_CHx_CTRL_SOMP.SL. Please refer to different mode sections for more output generation details.

Thus, the output level of PWM pulse width can be changed during runtime by synchronous update or asynchronous update of CLSi_ATOM_CHx_CTRL_SOMP.SL bit of the channel configuration register.

Continuous counting up mode

In SOMP continuous counting up mode the **ATOM** channel starts incrementing the counter CLSi_ATOM_CHx_CN0.CN0 once it is enabled by setting the corresponding bits in register CLSi_ATOM_AGC_ENDIS_STAT (refer to Figure 1075 for details of enabling an **ATOM** channel).

In this mode with CLSi_ATOM_CHx_CTRL_SOMP.UDMODE=00_B (i.e. CLSi_ATOM_CHx_CN0.CN0 counts only up), depending on configuration bits CLSi_ATOM_CHx_CTRL_SOMP.RST_CCU0 the counter register CLSi_ATOM_CHx_CN0.CN0 can be reset either when the counter value is equal to the compare value CLSi_ATOM_CHx_CM0.CM0 (i.e. CLSi_ATOM_CHx_CN0.CN0 counts only from 0 to CLSi_ATOM_CHx_CM0.CM0-1 and is then reset to 0) or when the trigger signal ATOM_CH_TRIGOUT[x-1:x-1] of the preceding channel x-1 (which can also be the last channel of preceding instance **ATOM[i-1]**) is captured by the **ATOM[i]** or when there is trigger signal EXT_TRIGIN[x:x] of the assigned **TIM** channel x after it was synchronized to the selected clock resolution signal.

In this case, if CLSi_ATOM_AGC_GLB_CTRL.UPEN_CTRLk (k=0-7) **(k=x)**=10_B, also the operation register CLSi_ATOM_CHx_CM0.CM0, CLSi_ATOM_CHx_CM1.CM1, CLSi_ATOM_CHx_CTRL_SOMP.SL and CLSi_ATOM_CHx_CTRL_SOMP.CLK_SRC are updated from their shadow registers together with CLSi_ATOM_CHx_CN0.CN0 reset.

As an exception, the input ATOM_CH_TRIGIN[0:0] of instance ATOM0 is triggered by its own last channel 7 via signal ATOM_CH_TRIGOUT[7:7].

The duration of the pulse high or low time and period is measured with the counter in sub-unit CCU0. The trigger of the counter is one of the eight **CMU** clock signals configurable in the channel control register CLSi_ATOM_CHx_CTRL_SOMP.CLK_SRC. The register CLSi_ATOM_CHx_CM0.CM0 holds the duration of the period and the register CLSi_ATOM_CHx_CM1.CM1 holds the duration of the pulse width in clock ticks of the selected **CMU** clock SEL_CMU_CLKEN. The signal level of the generated output signal can be configured with the configuration bit CLSi_ATOM_CHx_CTRL_SOMP.SL.

If counter register CLSi_ATOM_CHx_CN0.CN0 of channel x is reset by its own CCU0 unit (i.e. the compare match of CLSi_ATOM_CHx_CN0.CN0>=CLSi_ATOM_CHx_CM0.CM0-1 configured by CLSi_ATOM_CHx_CTRL_SOMP.RST_CCU0=0), the PWM output behavior is shown in Figure 1081 and the following statements are valid:

- After **ATOM** i channel x is enabled, CLSi_ATOM_CHx_CN0.CN0 counts firstly from its initial value to CLSi_ATOM_CHx_CM0.CM0-1 and is then reset to 0. This phase is defined as initial delay that can be flexibly configured as by the value of (CLSi_ATOM_CHx_CM0.CM0-CLSi_ATOM_CHx_CN0.CN0) multiplied with the selected **CMU** clock period.
- When CLSi_ATOM_CHx_CN0.CN0 is reset from CLSi_ATOM_CHx_CM0.CM0-1 to 0, an edge to CLSi_ATOM_CHx_CTRL_SOMP.SL is generated.
- When CLSi_ATOM_CHx_CN0.CN0 is incrementing and reaches CLSi_ATOM_CHx_CM1.CM1 (CLSi_ATOM_CHx_CN0.CN0>=CLSi_ATOM_CHx_CM1.CM1), an edge to !CLSi_ATOM_CHx_CTRL_SOMP.SL is generated.
- CLSi_ATOM_CHx_CN0.CN0 always counts from 0 to CLSi_ATOM_CHx_CM0.CM0-1 and is then reset to 0.
- If CLSi_ATOM_CHx_CM0.CM0=0 or CLSi_ATOM_CHx_CM0.CM0=1, the counter CLSi_ATOM_CHx_CN0.CN0 is constant 0. It is highly recommended to configure CLSi_ATOM_CHx_CM0.CM0>1.
- If CLSi_ATOM_CHx_CM1.CM1=0 and CLSi_ATOM_CHx_CM0.CM0>1, the output is ! CLSi_ATOM_CHx_CTRL_SOMP.SL (i.e. 0% duty cycle PWM signal).
- If CLSi_ATOM_CHx_CM1.CM1 >= CLSi_ATOM_CHx_CM0.CM0 and CLSi_ATOM_CHx_CM0.CM0 >1, the output is CLSi_ATOM_CHx_CTRL_SOMP.SL (i.e. 100% duty cycle PWM signal).

Since the ATOM_OUT signal level is defined as !CLSi_ATOM_CHx_CTRL_SOMP.SL when the **ATOM** channel is disabled without setting freeze mode, the first PWM period can be shifted earlier by writing an initial offset value to CLSi_ATOM_CHx_CN0.CN0 register. By doing this, the **ATOM** channel first counts until CLSi_ATOM_CHx_CN0.CN0 reaches CLSi_ATOM_CHx_CM0.CM0 and then it toggles the output signal at ATOM_OUT.

In this case, as shown in Figure 1081, 2 interrupt signals ATOM_CCU0TC[x]_IRQ and ATOM_CCU1TC[x]_IRQ of channel x can be generated together with output generation ATOM_OUT[x:x] and both of them are shown on ATOM_IRQ[x:x] if CLSi_ATOM_CHx_IRQ_EN=11_B. When CLSi_ATOM_CHx_CN0.CN0 is reset from CLSi_ATOM_CHx_CM0.CM0 -1 to 0, ATOM_CCU0TC[x]_IRQ of channel x is generated. When CLSi_ATOM_CHx_CN0.CN0 is incrementing to reach CLSi_ATOM_CHx_CM1.CM1, ATOM_CCU1TC[x]_IRQ of

channel x is generated. Especially, if the initial configuration before enabling the channel fulfills the condition CLSi_ATOM_CHx_CN0.CN0 >= CLSi_ATOM_CHx_CM1.CM1 - 1, an interrupt is generated after 1 selected **CMU** clock period on ATOM_CCU1TC[x]_IRQ. Similarly but not shown in the figure, if the initial configuration before enabling the channel fulfills the condition CLSi_ATOM_CHx_CN0.CN0 >= CLSi_ATOM_CHx_CM0.CM0 - 1, an interrupt is generated after 1 selected **CMU** clock period on ATOM_CCU0TC[x]_IRQ. The interrupt signals are always aligned with the edges of the output ATOM_OUT[x:x].

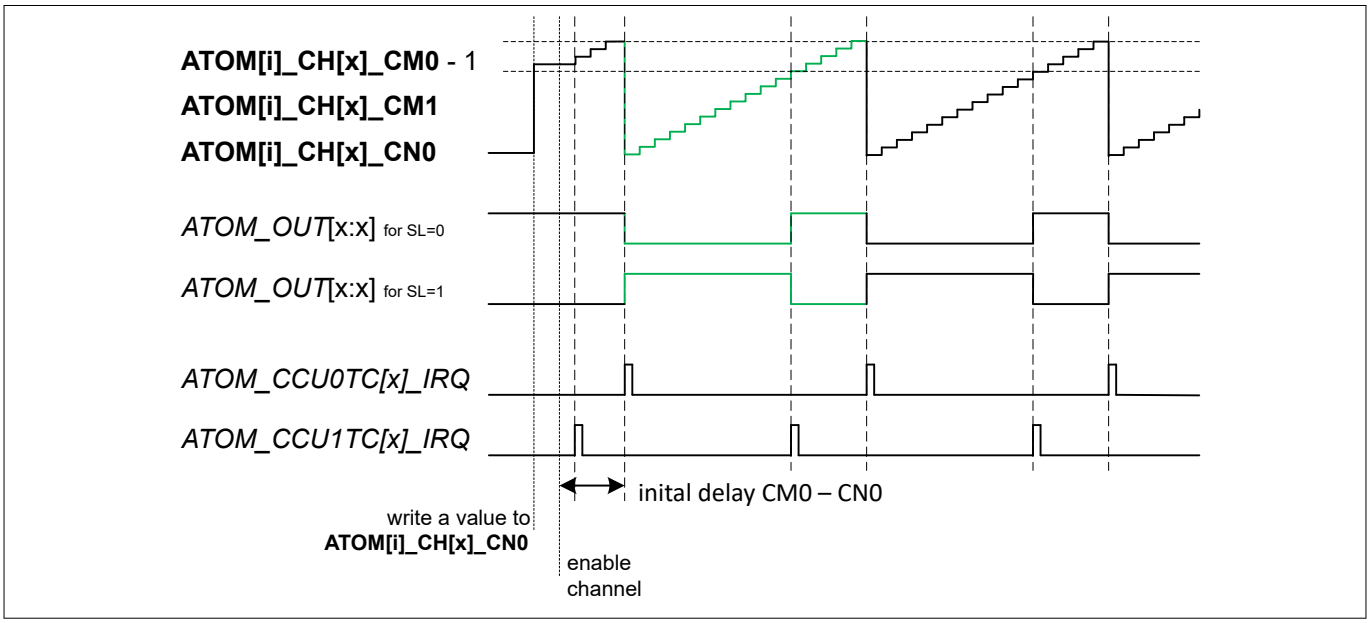


Figure 1081 PWM **output behavior with respect to the** CLSi_ATOM_CHx_CTRL_SOMP.SL **bit in continuous counting up mode if** CLSi_ATOM_CHx_CTRL_SOMP.RST_CCU0**=0**

If the counter register CLSi_ATOM_CHx_CN0.CN0 of channel x is reset by the trigger signal coming from another channel or the assigned **TIM** module (configured by CLSi_ATOM_CHx_CTRL_SOMP.RST_CCU0=1), the PWM output behavior is shown in Figure 1082 and the following statements are valid (with MAX=number of selected CCM[i]_CLK_RES cycle between two trigger signals):

- CLSi_ATOM_CHx_CN0.CN0 counts from 0 to MAX-1 and is then reset to 0 at the falling edge of ATOM_CH_TRIGIN[x:x] or at falling edge of EXT_TRIGIN_S (the synchronized signal of EXT_TRIGIN[x:x], see External trigger interface).
- If CLSi_ATOM_CHx_CN0.CN0 reaches CLSi_ATOM_CHx_CM1.CM1, an edge to ! CLSi_ATOM_CHx_CTRL_SOMP.SL is generated.
- If CLSi_ATOM_CHx_CN0.CN0 reaches CLSi_ATOM_CHx_CM0.CM0, an edge to CLSi_ATOM_CHx_CTRL_SOMP.SL is generated.
- As soon as CLSi_ATOM_CHx_CN0.CN0 reaches the value of CLSi_ATOM_CHx_CM0.CM0 while CLSi_ATOM_CHx_CM1.CM1 is equal to CLSi_ATOM_CHx_CM0.CM0, an edge to CLSi_ATOM_CHx_CTRL_SOMP.SL is generated at the output or the output remains at CLSi_ATOM_CHx_CTRL_SOMP.SL level depending on the former level of the output (CLSi_ATOM_CHx_CM0.CM0 has higher priority). Please note that this configuration is not suitable for generating 100% duty cycle.
- If CLSi_ATOM_CHx_CM0.CM0=0 and CLSi_ATOM_CHx_CM1.CM1>MAX, the output is a pulse of CLSi_ATOM_CHx_CTRL_SOMP.SL for the period MAX (i.e. 100% duty cycle PWM signal).
- If CLSi_ATOM_CHx_CM0.CM0 > MAX and CLSi_ATOM_CHx_CM1.CM1=0, the output is ! CLSi_ATOM_CHx_CTRL_SOMP.SL (i.e. 0% duty cycle PWM signal).
- If CLSi_ATOM_CHx_CM0.CM0 is set to MAX, an edge to CLSi_ATOM_CHx_CTRL_SOMP.SL is generated at the output at the very end of the current period or the output remains at the CLSi_ATOM_CHx_CTRL_SOMP.SL level, depending on the previous level of the output. Be aware that this applies even if the value of

CLSi_ATOM_CHx_CM1.CM1 is updated to 0 for the new period (CLSi_ATOM_CHx_CM0.CM0 has higher priority).

In this case, as shown in Figure 1082, 2 interrupt signals ATOM_CCU0TC[x]_IRQ and ATOM_CCU1TC[x]_IRQ of channel x can be generated together with output generation ATOM_OUT[x:x] and both of them are shown on ATOM_IRQ[x:x] if CLSi_ATOM_CHx_IRQ_EN=11_B. When CLSi_ATOM_CHx_CN0.CN0 is incrementing to reach CLSi_ATOM_CHx_CM0.CM0, ATOM_CCU0TC[x]_IRQ of channel x is generated. When CLSi_ATOM_CHx_CN0.CN0 is incrementing to reach CLSi_ATOM_CHx_CM1.CM1, ATOM_CCU1TC[x]_IRQ of channel x is generated. The interrupt signals are always aligned with the edges of the output ATOM_OUT[x:x].

According to internal trigger interface specification Internal trigger interface, more than one channel (maybe also in different clusters) can be triggered by a common preceding channel to synchronously generate consistent PWM signals. In such applications, the channels and instances between the triggering channel to the triggered channels build up a trigger chain and the triggered channels should be configured with same PWM parameters (i.e. period, duty cycle, signal level and clock source). However, possible pipeline registers according to Internal trigger interface may introduce delay of several cluster clock periods CLS[j]_CLK or **eGTM** clock periods CLK in the trigger chain. As a result, the triggered channels in the trigger chain may be asynchronously triggered and have different behavior even though they are triggered by a common channel and they are configured with same PWM parameters. Therefore, if in real applications there are pipeline registers in the trigger chain, it is necessary to configure the selected **CMU** clock source to be a slower frequent clock enable signal to avoid the influence of the introduced delay so that the triggered channels can be synchronously triggered and have consistent behavior. The period between two enable pulses of the selected **CMU** clock source (i.e. the distance between 2 falling edges of SEL_CMU_CLKEN) should be greater than the total delay caused by all the pipeline registers in the trigger chain. Especially, if the triggering channel and other triggered channels are in a common instance i where CLS[j]_CLK is enabled with clock divider of 2 (CLSi_ARCH_CLK_CFG.CLSj_CLK_DIV (j=0-2) **(j=i)**=2) and there is only one pipeline register (with delay of one CLK period) between channel 3 and channel 4 in the trigger chain, then such a delay less than CLS[j]_CLK period can be ignored regardless of the selection of **CMU** clock source.

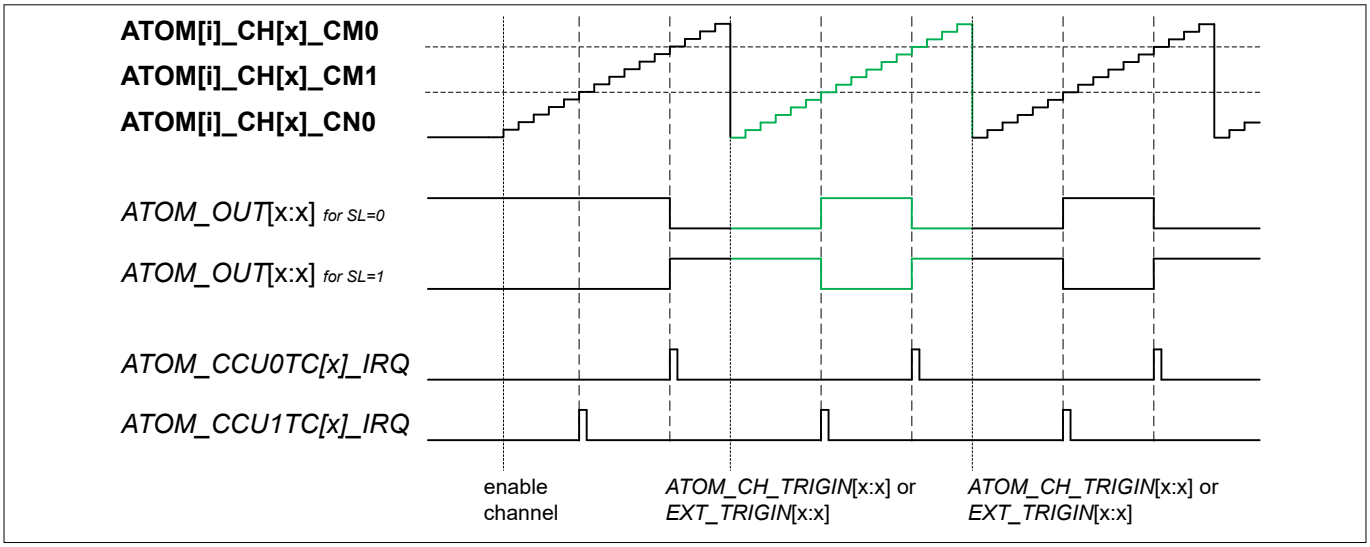


Figure 1082 PWM output behavior with respect to the CLSi_ATOM_CHx_CTRL_SOMP.SL bit in continuous counting up mode if CLSi_ATOM_CHx_CTRL_SOMP.RST_CCU0=1

In case of CLSi_ATOM_AGC_GLB_CTRL.UPEN_CTRLk (k=0-7) **(k=x)**=10_B, when the counter value CLSi_ATOM_CHx_CN0.CN0 reaches the compare value in register CLSi_ATOM_CHx_CM0.CM0 (in fact CLSi_ATOM_CHx_CM0.CM0-1) if CLSi_ATOM_CHx_CTRL_SOMP.RST_CCU0=0 or the channel captures the falling edge of the trigger signal if CLSi_ATOM_CHx_CTRL_SOMP.RST_CCU0=1, the operation registers CLSi_ATOM_CHx_CM0.CM0, CLSi_ATOM_CHx_CM1.CM1, CLSi_ATOM_CHx_CTRL_SOMP.SL and CLSi_ATOM_CHx_CTRL_SOMP.CLK_SRC are updated with the content of their shadow registers

CLSi_ATOM_CHx_SR0.SR0, CLSi_ATOM_CHx_SR1.SR1, CLSi_ATOM_CHx_CTRL_SR.SL_SR and CLSi_ATOM_CHx_CTRL_SR.CLK_SRC_SR.

The operation registers CLSi_ATOM_CHx_CM0.CM0, CLSi_ATOM_CHx_CM1.CM1, CLSi_ATOM_CHx_CTRL_SOMP.SL and CLSi_ATOM_CHx_CTRL_SOMP.CLK_SRC can also be asynchronously updated from their shadow registers with the force update mechanism. Please refer to Pulse width, period, signal level and clock frequency update mechanisms for the force update configuration steps and more details. In continuous counting up mode, the force update mechanism can be also used to stop the current running PWM signal generation and restart a new PWM period with new PWM parameters by additionally setting CLSi_ATOM_AGC_FUPD_CTRL.RSTCN0_CHk (k=0-7) **(k=x)** = 10_B together with CLSi_ATOM_AGC_FUPD_CTRL.FUPD_CTRLk (k=0-7) **(k=x)** = 10_B. That means the force update event will not only update the operation registers from their shadow registers, but also simultaneously reset the counter to restart the next new PWM period with setting the output as CLSi_ATOM_CHx_CTRL_SOMP.SL if CLSi_ATOM_CHx_CTRL_SOMP.RST_CCU0=0 or without changing the output if CLSi_ATOM_CHx_CTRL_SOMP.RST_CCU0=1.

Continuous counting up-down mode

In SOMP mode, if CLSi_ATOM_CHx_CTRL_SOMP.UDMODE != 00_B, the counter CLSi_ATOM_CHx_CN0.CN0 will count up and down. Depending on the configuration bit CLSi_ATOM_CHx_CTRL_SOMP.RST_CCU0, the counter register CLSi_ATOM_CHx_CN0.CN0 changes the direction either when the counter value is equal to the compare value CLSi_ATOM_CHx_CM0.CM0-1 or when triggered by the internal trigger signal ATOM_CH_TRIGIN[x:x] (i.e. ATOM_CH_TRIGOUT[x-1:x-1] from the preceding channel [x-1]) or the external trigger signal EXT_TRIGIN[x:x] from **TIM** module [i].

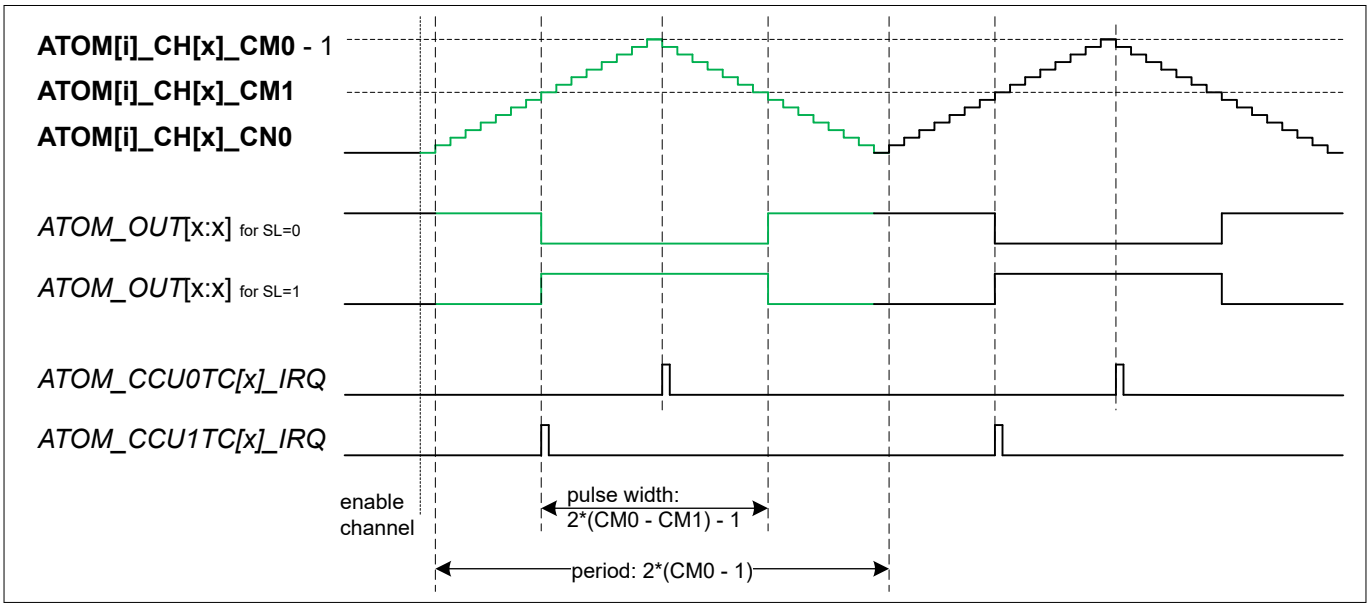


Figure 1083 PWM **output behavior with respect to the** CLSi_ATOM_CHx_CTRL_SOMP.SL **bit in continuous counting up-down mode if** CLSi_ATOM_CHx_CTRL_SOMP.RST_CCU0**=0**

The clock of the counter register CLSi_ATOM_CHx_CN0.CN0 can be one of the **CMU** enable clock resolutions CCM[i]_CLK_RES.

The clock for CLSi_ATOM_CHx_CN0.CN0 is defined by CLSi_ATOM_CHx_CTRL_SOMP.CLK_SRC value.

If CLSi_ATOM_CHx_CTRL_SOMP.RST_CCU0=0, the counter register CLSi_ATOM_CHx_CN0.CN0 of channel x will change its counting direction by the compare match of CLSi_ATOM_CHx_CN0.CN0>=CLSi_ATOM_CHx_CM0.CM0-1 in its own CCU0 unit and the following statements are valid:

- In each PWM period, CLSi_ATOM_CHx_CN0.CN0 firstly counts continuously up from 1 to CLSi_ATOM_CHx_CM0.CM0-1 and then down to 0. So period = 2*(CLSi_ATOM_CHx_CM0.CM0-1).
- If CLSi_ATOM_CHx_CN0.CN0 >= CLSi_ATOM_CHx_CM1.CM1, the output is ATOM_OUT[x:x] set to CLSi_ATOM_CHx_CTRL_SOMP.SL.
- If CLSi_ATOM_CHx_CN0.CN0 < CLSi_ATOM_CHx_CM1.CM1, the output is ATOM_OUT[x:x] set to !CLSi_ATOM_CHx_CTRL_SOMP.SL. So pulse width = 2*(CLSi_ATOM_CHx_CM0.CM0 - CLSi_ATOM_CHx_CM1.CM1)-1.
- If CLSi_ATOM_CHx_CM0.CM0 <= 1, the counter behavior of CLSi_ATOM_CHx_CN0.CN0 is unexpected (not recommended).
- If CLSi_ATOM_CHx_CM1.CM1 = 0 and also CLSi_ATOM_CHx_CM0.CM0 > 1, the output ATOM_OUT[x:x] is CLSi_ATOM_CHx_CTRL_SOMP.SL (i.e. 100% duty cycle).
- If CLSi_ATOM_CHx_CM1.CM1 >= CLSi_ATOM_CHx_CM0.CM0 and also CLSi_ATOM_CHx_CM0.CM0 > 1, the output ATOM_OUT[x:x] is !CLSi_ATOM_CHx_CTRL_SOMP.SL (i.e. 0% duty cycle).

This behavior is depicted in Figure 1083. In this case, if CLSi_ATOM_AGC_GLB_CTRL.UPEN_CTRLk (k=0-7) **(k=x)**=1, the operation registers CLSi_ATOM_CHx_CM0.CM0, CLSi_ATOM_CHx_CM1.CM1, CLSi_ATOM_CHx_CTRL_SOMP.SL and CLSi_ATOM_CHx_CTRL_SOMP.CLK_SRC can be updated when CLSi_ATOM_CHx_CN0.CN0 = 0 (i.e. counting direction changes from down to up) or CLSi_ATOM_CHx_CN0.CN0 reaches CLSi_ATOM_CHx_CM0.CM0-1 (i.e. counting direction changes from up to down) depending on various configurations of CLSi_ATOM_CHx_CTRL_SOMP.UDMODE (see the register description for more details). Especially, it is not recommended to change the value of CLSi_ATOM_CHx_CM0.CM0, CLSi_ATOM_CHx_CTRL_SOMP.SL and CLSi_ATOM_CHx_CTRL_SOMP.CLK_SRC when CLSi_ATOM_CHx_CN0.CN0 reaches CLSi_ATOM_CHx_CM0.CM0-1. Otherwise, the behavior may be unexpected.

Like continuous counting up mode, interrupt signals ATOM_CCU0TC[x]_IRQ and ATOM_CCU1TC[x]_IRQ of channel x are generated together with PWM signal output edges. When CLSi_ATOM_CHx_CN0.CN0 reaches CLSi_ATOM_CHx_CM1.CM1 at the first time and the first edge is generated, an interrupt on ATOM_CCU1TC[x]_IRQ is generated. When CLSi_ATOM_CHx_CN0.CN0 changes its counting direction from up to down, an interrupt on ATOM_CCU0TC[x]_IRQ is generated.

If CLSi_ATOM_CHx_CTRL_SOMP.RST_CCU0=1, the counter register CLSi_ATOM_CHx_CN0.CN0 of channel x will change its counting direction by the trigger signal coming from another channel or the assigned **TIM** module and the following statements are valid:

- CLSi_ATOM_CHx_CN0.CN0 firstly counts continuously up. At the falling edge of the trigger signal pulse ATOM_CH_TRIGIN[x:x] or EXT_TRIGIN_S (the synchronized signal of EXT_TRIGIN[x:x], see External trigger interface), the counter switches to count down mode. If CLSi_ATOM_CHx_CN0.CN0 reaches 0, it will count up again.
- If CLSi_ATOM_CHx_CN0.CN0 >= CLSi_ATOM_CHx_CM1.CM1, the output ATOM_OUT[x:x] is set to CLSi_ATOM_CHx_CTRL_SOMP.SL.
- If CLSi_ATOM_CHx_CN0.CN0 < CLSi_ATOM_CHx_CM1.CM1, the output ATOM_OUT[x:x] is set to ! CLSi_ATOM_CHx_CTRL_SOMP.SL.
- If CLSi_ATOM_CHx_CM1.CM1 = 0, the output ATOM_OUT[x:x] is CLSi_ATOM_CHx_CTRL_SOMP.SL (i.e. 100% duty cycle).
- If CLSi_ATOM_CHx_CN0.CN0 >= CLSi_ATOM_CHx_CM0.CM0, the output ATOM_OUT_T[x:x] is set to CLSi_ATOM_CHx_CTRL_SOMP.SL.
- If CLSi_ATOM_CHx_CN0.CN0 < CLSi_ATOM_CHx_CM0.CM0, the output ATOM_OUT_T[x:x] is set to ! CLSi_ATOM_CHx_CTRL_SOMP.SL.
- If CLSi_ATOM_CHx_CM0.CM0 = 0, the output ATOM_OUT_T[x:x] is CLSi_ATOM_CHx_CTRL_SOMP.SL (i.e. 100% duty cycle).

This behavior is depicted in Figure 1084. In this case, if CLSi_ATOM_AGC_GLB_CTRL.UPEN_CTRLk (k=0-7) **(k=x)**=1, the operation registers CLSi_ATOM_CHx_CM0.CM0, CLSi_ATOM_CHx_CM1.CM1, CLSi_ATOM_CHx_CTRL_SOMP.SL and CLSi_ATOM_CHx_CTRL_SOMP.CLK_SRC can also be updated when

CLSi_ATOM_CHx_CN0.CN0 = 0 (i.e. counting direction changes from down to up) or CLSi_ATOM_CHx_CN0.CN0 reaches its maximum value (i.e. counting direction changes from up to down) depending on various configurations of CLSi_ATOM_CHx_CTRL_SOMP.UDMODE (see the register description for more details). Especially, it is not recommended to change the value of CLSi_ATOM_CHx_CTRL_SOMP.SL and CLSi_ATOM_CHx_CTRL_SOMP.CLK_SRC when CLSi_ATOM_CHx_CN0.CN0 reaches its maximum value (i.e. counting direction changes from up to down). Otherwise, the behavior may be unexpected.

If CLSi_ATOM_CHx_IRQ_EN=11_B, interrupt signals ATOM_CCU0TC[x]_IRQ and ATOM_CCU1TC[x]_IRQ of channel x are generated together with PWM signal output. When CLSi_ATOM_CHx_CN0.CN0 reaches CLSi_ATOM_CHx_CM1.CM1 at the first time and the first edge of ATOM_OUT[x:x] is generated, an interrupt on ATOM_CCU1TC[x]_IRQ is generated. When CLSi_ATOM_CHx_CN0.CN0 reaches CLSi_ATOM_CHx_CM0.CM0 at the first time and the first edge of ATOM_OUT_T[x:x] is generated, an interrupt on ATOM_CCU0TC[x]_IRQ is generated.

In case of continuous counting up-down mode and CLSi_ATOM_CHx_CTRL_SOMP.RST_CCU0=1, it is recommended to configure the **ATOM** channel according to the following statements in order to realize several cyclic central aligned PWM signals:

- The triggering channel and the triggered channel are both running in up-down mode.
- The time between two trigger signals is equal to the time needed for CLSi_ATOM_CHx_CN0.CN0 of triggered channel to count back to 0 and again up to the same upper value.

The second recommendation can be fulfilled by synchronizing the start of the triggering channel and the triggered channel, i.e. let both channels start with CLSi_ATOM_CHx_CN0.CN0 = 0 and same clock source.

If there is a pipeline register in the trigger chain, the additional delay of one clock period has to be taken into account by starting the triggering channel with CLSi_ATOM_CHx_CN0.CN0 = 1 (i.e. 1 greater than CLSi_ATOM_CHx_CN0.CN0 of the triggered channel).

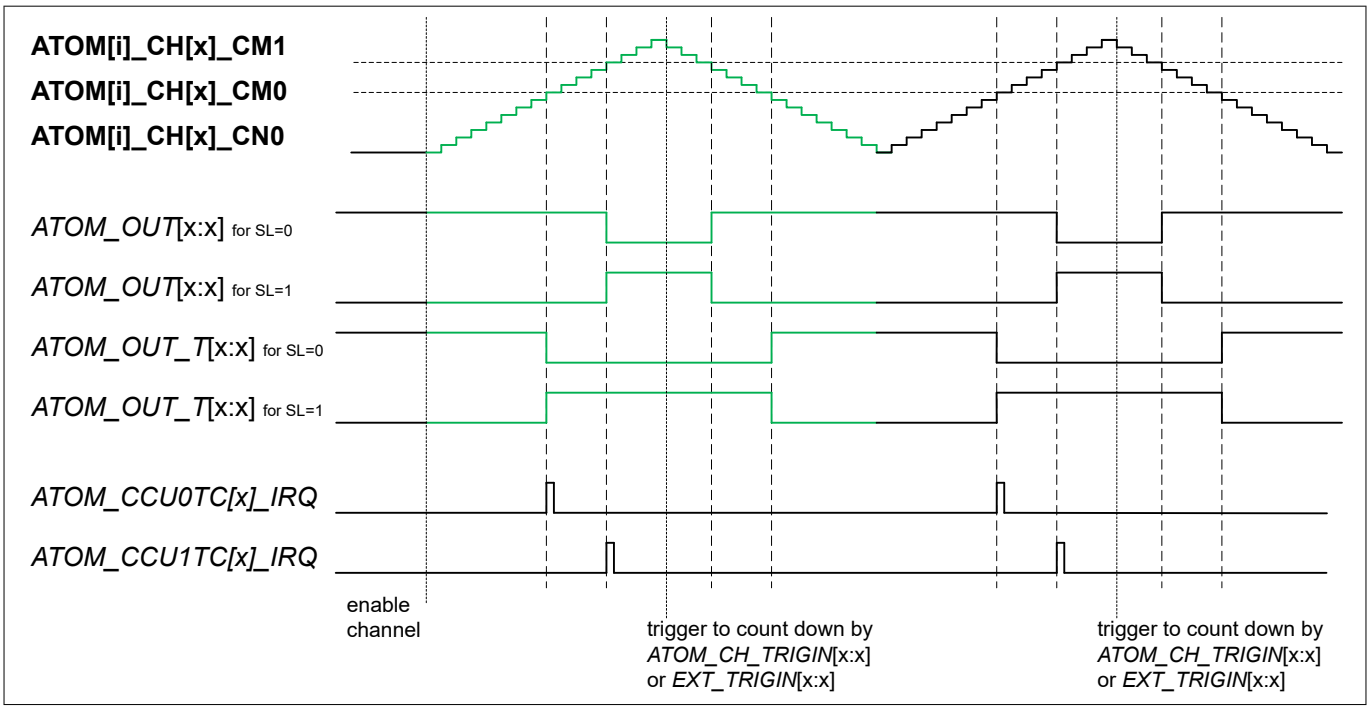


Figure 1084 PWM **output behavior with respect to the** CLSi_ATOM_CHx_CTRL_SOMP.SL **bit in continuous counting up-down mode if** CLSi_ATOM_CHx_CTRL_SOMP.RST_CCU0**=1**

One-shot counting up mode

The **ATOM** channel can operate in one-shot mode when the CLSi_ATOM_CHx_CTRL_SOMP.OSM bit is set. One-shot mode means that a single pulse with the pulse level defined in bit CLSi_ATOM_CHx_CTRL_SOMP.SL is generated on the output signal.

Firstly, the channel has to be enabled by setting the corresponding CLSi_ATOM_AGC_ENDIS_STAT.ENDIS_STATk (k=0-7).

Unlike in the continuous counting modes, the counter CLSi_ATOM_CHx_CN0.CN0 will not be incremented once the channel is enabled. Instead, according to the configuration bit CLSi_ATOM_CHx_CTRL_SOMP.OSM_TRIG, the start of counting and pulse generation is triggered by a write-access of CLSi_ATOM_CHx_CN0.CN0 or the internal trigger signal ATOM_CH_TRIGIN[x:x] or the external trigger signal EXT_TRIGIN[x:x].

When CLSi_ATOM_CHx_CTRL_SOMP.OSM_TRIG=0, a write-access to the register CLSi_ATOM_CHx_CN0.CN0 triggers the start of incrementing the counter register CLSi_ATOM_CHx_CN0.CN0 and pulse generation. If the time between the write-access and the first edge generation of one-shot PWM signal is defined as initial delay, the initial delay is determined by the value of (CLSi_ATOM_CHx_CM0.CM0-CLSi_ATOM_CHx_CN0.CN0) multiplied with the selected **CMU** clock period. Especially, if writing a value of greater than or equal to CLSi_ATOM_CHx_CM0.CM0-1 to CLSi_ATOM_CHx_CN0.CN0 and configuring the initial clock source as cluster clock (the clock source at reset state), the initial delay is minimum, i.e. 1 cluster clock period CLS[j]_CLK. That means after the write-access of CLSi_ATOM_CHx_CN0.CN0, the first edge of the output with the interrupt will be set in 1 CLS[j]_CLK period.

If the counter CLSi_ATOM_CHx_CN0.CN0 is reset from CLSi_ATOM_CHx_CM0.CM0-1 back to zero, the first edge at ATOM_OUT is generated. And the operation phase of the pulse generation starts incrementing of CLSi_ATOM_CHx_CN0.CN0 from 0 to CLSi_ATOM_CHx_CM0.CM0-1 again.

The second edge is generated, if CLSi_ATOM_CHx_CN0.CN0 increments until it reaches CLSi_ATOM_CHx_CM1.CM1.

If the counter CLSi_ATOM_CHx_CN0.CN0 reaches the value of CLSi_ATOM_CHx_CM0.CM0-1 at the second time, the counter will stop.

Figure 1085 depicts the pulse generation process mentioned above. The interrupt signals generation during PWM signal generation are similar as described in continuous counting up mode. If the value written to CLSi_ATOM_CHx_CN0.CN0 for triggering or retriggering the counter is greater than or equal to CLSi_ATOM_CHx_CM1.CM1 -1, an interrupt on ATOM_CCU1TC[x]_IRQ is generated after 1 selected **CMU** clock period. Similarly but not shown in the figure, if the value written to CLSi_ATOM_CHx_CN0.CN0 for triggering or retriggering the counter is greater than or equal to CLSi_ATOM_CHx_CM0.CM0 - 1, an interrupt on ATOM_CCU0TC[x]_IRQ is generated after 1 selected **CMU** clock period. Afterwards when CLSi_ATOM_CHx_CN0.CN0 is reset from CLSi_ATOM_CHx_CM0.CM0 -1 to 0, ATOM_CCU0TC[x]_IRQ of channel x is generated. When CLSi_ATOM_CHx_CN0.CN0 is incrementing to reach CLSi_ATOM_CHx_CM1.CM1, ATOM_CCU1TC[x]_IRQ of channel x is generated.

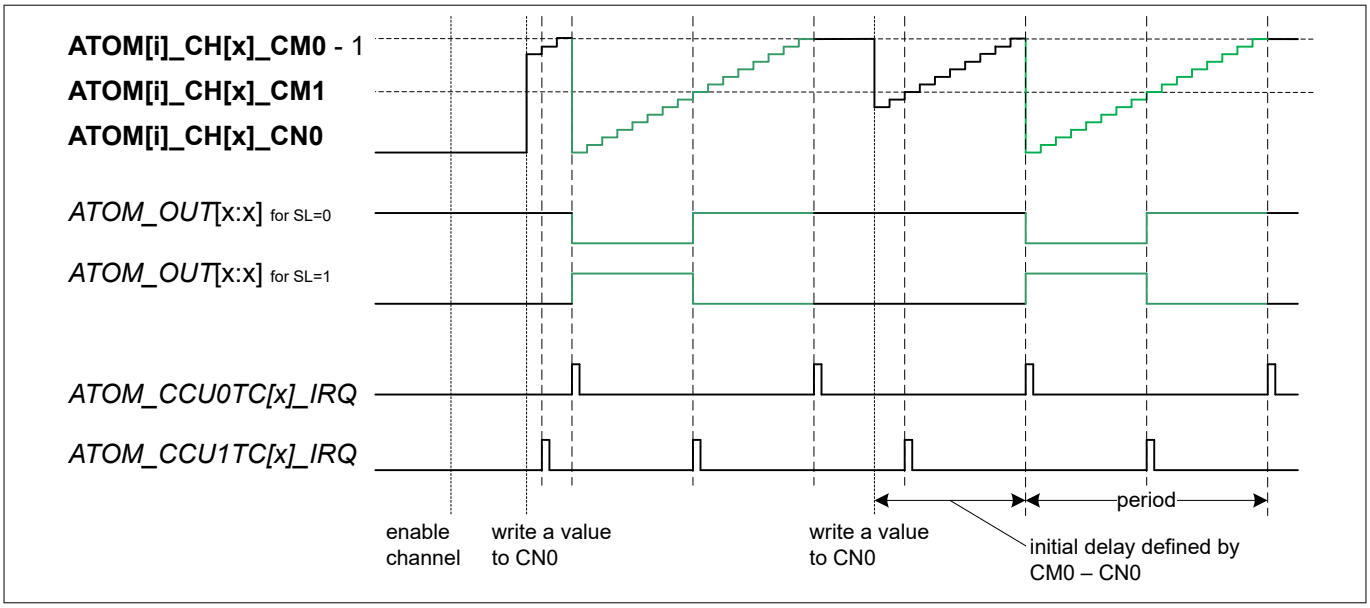


Figure 1085 PWM output with respect to configuration bit CLSi_ATOM_CHx_CTRL_SOMP.SL in one-shot counting up mode if CLSi_ATOM_CHx_CTRL_SOMP.OSM_TRIG= 0

Retriggering an one-shot cycle (while CLSi_ATOM_CHx_CN0.CN0 is already incrementing) by writing a value to CLSi_ATOM_CHx_CN0.CN0 is possible but depends on the phase of CLSi_ATOM_CHx_CN0.CN0:

- phase 1: update of CLSi_ATOM_CHx_CN0.CN0 before CLSi_ATOM_CHx_CN0.CN0 reaches CLSi_ATOM_CHx_CM0.CM0 for the first time (initial phase)
- phase 2: update of CLSi_ATOM_CHx_CN0.CN0 after CLSi_ATOM_CHx_CN0.CN0 has reached CLSi_ATOM_CHx_CM0.CM0 for the first time until the current single PWM pulse is finished (operation phase)

In phase 1 (initial phase), writing a value CN0(new) to counter CLSi_ATOM_CHx_CN0.CN0 (CN0(new)< CLSi_ATOM_CHx_CM0.CM0) leads to a shift of the first edge by the time CLSi_ATOM_CHx_CM0.CM0-CN0(new), i.e. the first edge is generated when CLSi_ATOM_CHx_CN0.CN0 is reset from CLSi_ATOM_CHx_CM0.CM0-1 back to 0 for the first time.

In phase 2 (operation phase): writing to counter CLSi_ATOM_CHx_CN0.CN0 is not possible and is protected by hardware.

A failed write-access to CLSi_ATOM_CHx_CN0.CN0 in phase 2 (operation phase) is indicated by the status register flag CLSi_ATOM_CHx_STAT.OSM_RTF.

After one PWM pulse is ended, other single pulses can be further triggered by a write-access to register CLSi_ATOM_CHx_CN0.CN0.

It is recommended to enable the update mechanism by setting CLSi_ATOM_AGC_GLB_CTRL.UPEN_CTRLk (k=0-7) **(k=x)** = 10_B for retriggering the second one-shot PWM signal generation with different initial delay and different PWM parameters. Similarly like continuous counting up mode with CLSi_ATOM_CHx_CTRL.RST_CCU0=0, the operation registers CLSi_ATOM_CHx_CM0.CM0/ CLSi_ATOM_CHx_CM1.CM1/CLSi_ATOM_CHx_CTRL.SL/CLSi_ATOM_CHx_CTRL.CLK_SRC are always synchronously updated when CLSi_ATOM_CHx_CN0.CN0 reaches CLSi_ATOM_CHx_CM0.CM0-1. Firstly, after the first one-shot PWM signal is generated, the new values for initial delay (phase 1) can be written into the shadow registers CLSi_ATOM_CHx_SR0.SR0/CLSi_ATOM_CHx_SR1.SR1/CLSi_ATOM_CHx_CTRL_SR.SL_SR/ CLSi_ATOM_CHx_CTRL_SR.CLK_SRC_SR before the retriggering is executed by writing of register CLSi_ATOM_CHx_CN0.CN0. The shadow registers should be loaded to the operation registers CLSi_ATOM_CHx_CM0.CM0/CLSi_ATOM_CHx_CM1.CM1/CLSi_ATOM_CHx_CTRL.SL/ CLSi_ATOM_CHx_CTRL.CLK_SRC at the next falling edge of the selected **CMU** clock resolution signal, because CLSi_ATOM_CHx_CN0.CN0 keeps as CLSi_ATOM_CHx_CM0.CM0-1. When CLSi_ATOM_CHx_CN0.CN0 is written to be retriggered afterwards, the initial delay is defined according to the updated operation registers. Secondly,

during the initial delay phase (phase 1), the new parameters for PWM signal can be written into the shadow registers again. The operational registers will be updated with the shadow registers when CLSi_ATOM_CHx_CN0.CN0 is reset from CLSi_ATOM_CHx_CM0.CM0-1 at the end of phase 1 and phase 2 is started to generate the second one-shot PWM signal with the new updated operation registers.

If a channel is configured to one-shot mode and configuration bit CLSi_ATOM_CHx_CTRL_SOMP.OSM_TRIG is set to 1, the trigger signal pulse ATOM_CH_TRIGIN[x:x] or EXT_TRIGIN[x:x] triggers start of pulse generation as shown in Figure 1086. In this case, it is not allowed to write CLSi_ATOM_CHx_CN0.CN0 and so the register CLSi_ATOM_CHx_STAT.OSM_RTF is not relevant. Otherwise, a write-access to CLSi_ATOM_CHx_CN0.CN0 can make an unpredictable output behavior. Furthermore, as shown in Figure 1086, CLSi_ATOM_CHx_CN0.CN0 should be initially configured as CLSi_ATOM_CHx_CM0.CM0-1 or greater value before enabling the channel. Otherwise, it is hard to define the counting behavior and the initial delay behavior. As shown in Figure 1086, when the channel is enabled, after the falling edge of the trigger pulse ATOM_CH_TRIGIN[x:x] or EXT_TRIGIN_S (the synchronized signal of EXT_TRIGIN[x:x], see External trigger interface), CLSi_ATOM_CHx_CN0.CN0 will be triggered to be reset that is synchronous with the next enable high pulse of the selected **CMU** clock source (i.e. synchronous to the next falling edge of the selected **CMU** enable signal). Then it counts from 0 to CLSi_ATOM_CHx_CM0.CM0-1. This process is defined as initial phase. If the internal trigger signal ATOM_CH_TRIGIN[x:x] is in use and it is synchronous to the selected **CMU** clock source or the external trigger signal EXT_TRIGIN[x:x] is in use and it must be synchronized to the selected **CMU** clock source before triggering, the initial delay can be defined as the time between the falling edge of the synchronized trigger signal to the time point of starting generating the PWM signal (green phase in the figure), i.e. CLSi_ATOM_CHx_CM0.CM0+1 selected **CMU** clock periods. Then CLSi_ATOM_CHx_CN0.CN0 will be reset again and count from 0 to CLSi_ATOM_CHx_CM0.CM0-1 at the second time to generate one signal PWM signal and then stop. This second counting is defined as operation phase.

After the first OSM signal is generated, the second trigger signal is allowed to be applied to start the counter and generate the next OSM signal after a certain initial delay. Figure 1086 shows such an application of generating 2 OSM signals without changing PWM parameters. In the case, after the second trigger signal is coming, the counter has the same behavior as described above for the first OSM signal generation. That means the initial delay is also CLSi_ATOM_CHx_CM0.CM0+1 selected **CMU** clock periods and the OSM signal is the same as the first one.

However, in some real applications of generating various single PWM signals, it is necessary to synchronously update the parameters in CLSi_ATOM_CHx_CM0.CM0/CLSi_ATOM_CHx_CM1.CM1/CLSi_ATOM_CHx_CTRL.SL/CLSi_ATOM_CHx_CTRL.CLK_SRC by setting CLSi_ATOM_AGC_GLB_CTRL.UPEN_CTRLk (k=0-7) **(k=x)** = 10_B. In this case, these registers are updated from their shadow registers when CLSi_ATOM_CHx_CN0.CN0 is matching CLSi_ATOM_CHx_CM0.CM0-1. After the first OSM PWM signal, if CLSi_ATOM_CHx_CM0.CM0 is updated with a new value CM0(new) before the second trigger signal comes, the initial delay of the second OSM signal depends on CM0(new) and the value CN0(old) of CLSi_ATOM_CHx_CN0.CN0 when the counter stopped at the end of the first OSM signal. The value CN0(old) is actually the value of CLSi_ATOM_CHx_CM0.CM0-1 of the first OSM signal before the synchronous update, i.e. CN0(old)=CM0(old)-1. Like the first OSM signal generation, with the falling edge of the second trigger pulse signal, the counter will be retriggered synchronously to the next enable high pulse of the selected **CMU** clock source (i.e. synchronous to the next falling edge of the selected **CMU** enable signal). Assuming the internal trigger signal ATOM_CH_TRIGIN[x:x] is in use and it is synchronous to the selected **CMU** clock source or the external trigger signal EXT_TRIGIN[x:x] is in use and it will be synchronized to the selected **CMU** clock source before triggering, the initial delay can be defined as the time between the falling edge of the second synchronized trigger signal to the time point of starting generating the PWM signal (green phase in the figure). On the one hand, if CN0(old) is greater than or equal to updated CM0(new)-1, the counter will be reset and counts up to CM0(new)-1 and the initial delay is CM0(new)+1 selected **CMU** clock periods, i.e. CLSi_ATOM_CHx_CM0.CM0+1 selected **CMU** clock periods. The behavior of the counter is the same as described above for the initial delay of the first OSM signal. On the other hand, if CN0(old) is smaller than updated CM0(new)-1, the initial phase or the initial delay is longer. In the initial phase, the counter will count up from CN0(old) to CM0(new)-1 and reset, and then increment again to CM0(new)-1. That means the initial delay is no longer CM0(new)+1, but the larger value of 2*CM0(new)-CN0(old) **CMU** clock periods, i.e. 2*CM0(new)-CM0(old) +1 selected **CMU** clock periods.

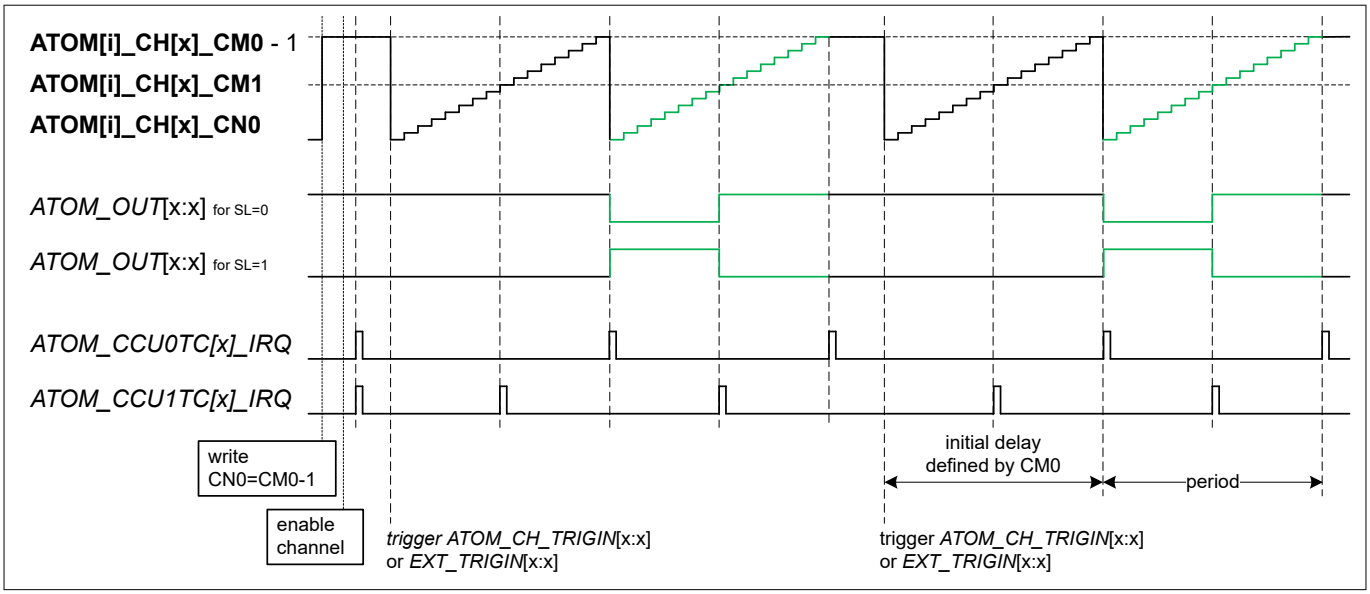


Figure 1086 PWM output with respect to configuration bit CLSi_ATOM_CHx_CTRL_SOMP.SL in one-shot counting up mode if CLSi_ATOM_CHx_CTRL_SOMP.OSM_TRIG= 1

As shown in Figure 1086, the interrupt signals generation during PWM signal generation are similar as described in continuous counting up mode. As before enabling the channel the initial value written to CLSi_ATOM_CHx_CN0.CN0 is greater than or equal to CLSi_ATOM_CHx_CM0.CM0 - 1 and CLSi_ATOM_CHx_CM1.CM1 - 1, both interrupt signals ATOM_CCU0TC[x]_IRQ and ATOM_CCU1TC[x]_IRQ are generated after 1 selected **CMU** clock period. Afterwards when CLSi_ATOM_CHx_CN0.CN0 is reset from CLSi_ATOM_CHx_CM0.CM0 -1 to 0, ATOM_CCU0TC[x]_IRQ of channel x is generated. When CLSi_ATOM_CHx_CN0.CN0 is incrementing to reach CLSi_ATOM_CHx_CM1.CM1, ATOM_CCU1TC[x]_IRQ of channel x is generated.

In one-shot mode, it is not recommended to configure CLSi_ATOM_CHx_CM0.CM0<2. Otherwise, the behavior may be unexpected.

One-shot counting up-down mode

The **ATOM** channel can operate in one-shot counting up-down mode when the bit CLSi_ATOM_CHx_CTRL_SOMP.OSM = 1 and the CLSi_ATOM_CHx_CTRL_SOMP.UDMODE != 00_B. One-shot mode means that a single pulse with the pulse level defined in bit CLSi_ATOM_CHx_CTRL_SOMP.SL is generated on the output signal.

Firstly, the channel has to be enabled by setting the corresponding CLSi_ATOM_AGC_ENDIS_STAT.ENDIS_STATk (k=0-7).

In one-shot mode the counter CLSi_ATOM_CHx_CN0.CN0 will not be incremented once the channel is enabled.

When CLSi_ATOM_CHx_CTRL_SOMP.OSM_TRIG=0, a write-access to the register CLSi_ATOM_CHx_CN0.CN0 triggers the start of incrementing the counter register CLSi_ATOM_CHx_CN0.CN0 and pulse generation.

If the counter CLSi_ATOM_CHx_CN0.CN0 is greater than or equal to CLSi_ATOM_CHx_CM1.CM1, the output ATOM_OUT is set to CLSi_ATOM_CHx_CTRL_SOMP.SL value.

If the counter CLSi_ATOM_CHx_CN0.CN0 is less than CLSi_ATOM_CHx_CM1.CM1, the output ATOM_OUT is set to !CLSi_ATOM_CHx_CTRL_SOMP.SL value.

If the counter CLSi_ATOM_CHx_CN0.CN0 has reached the value 0 (by counting down), it will stop.

The written value of CLSi_ATOM_CHx_CN0.CN0 determines the start delay of the first edge. The delay time of the first edge is given by (CLSi_ATOM_CHx_CM1.CM1-CLSi_ATOM_CHx_CN0.CN0) multiplied with the period defined by current value of CLSi_ATOM_CHx_CTRL_SOMP.CLK_SRC.

Figure 1087 depicts the pulse generation in SOMP one-shot counting up-down mode. The interrupt signals are generated similarly as described in one-shot counting up mode and continuous counting up-down mode. The write-access of CLSi_ATOM_CHx_CN0.CN0 can cause interrupts if the write value is greater than or equal to CLSi_ATOM_CHx_CM1.CM1 - 1 or CLSi_ATOM_CHx_CM0.CM0 -1. If the value written to CLSi_ATOM_CHx_CN0.CN0 for triggering or retriggering the counter is greater than or equal to CLSi_ATOM_CHx_CM1.CM1 -1, the output ATOM_OUT[x:x] will be set to CLSi_ATOM_CHx_CTRL_SOMP.SL after 1 selected **CMU** clock period together with an interrupt on ATOM_CCU1TC[x]_IRQ. Similarly but not shown in the figure, if the value written to CLSi_ATOM_CHx_CN0.CN0 for triggering or retriggering the counter is greater than or equal to CLSi_ATOM_CHx_CM0.CM0 - 1, an interrupt on ATOM_CCU0TC[x]_IRQ will be generated after 1 selected **CMU** clock period. Afterwards when CLSi_ATOM_CHx_CN0.CN0 is incrementing to reach CLSi_ATOM_CHx_CM1.CM1, an interrupt on ATOM_CCU1TC[x]_IRQ is generated. When CLSi_ATOM_CHx_CN0.CN0 changes its counting direction from up to down, an interrupt on ATOM_CCU0TC[x]_IRQ is generated.

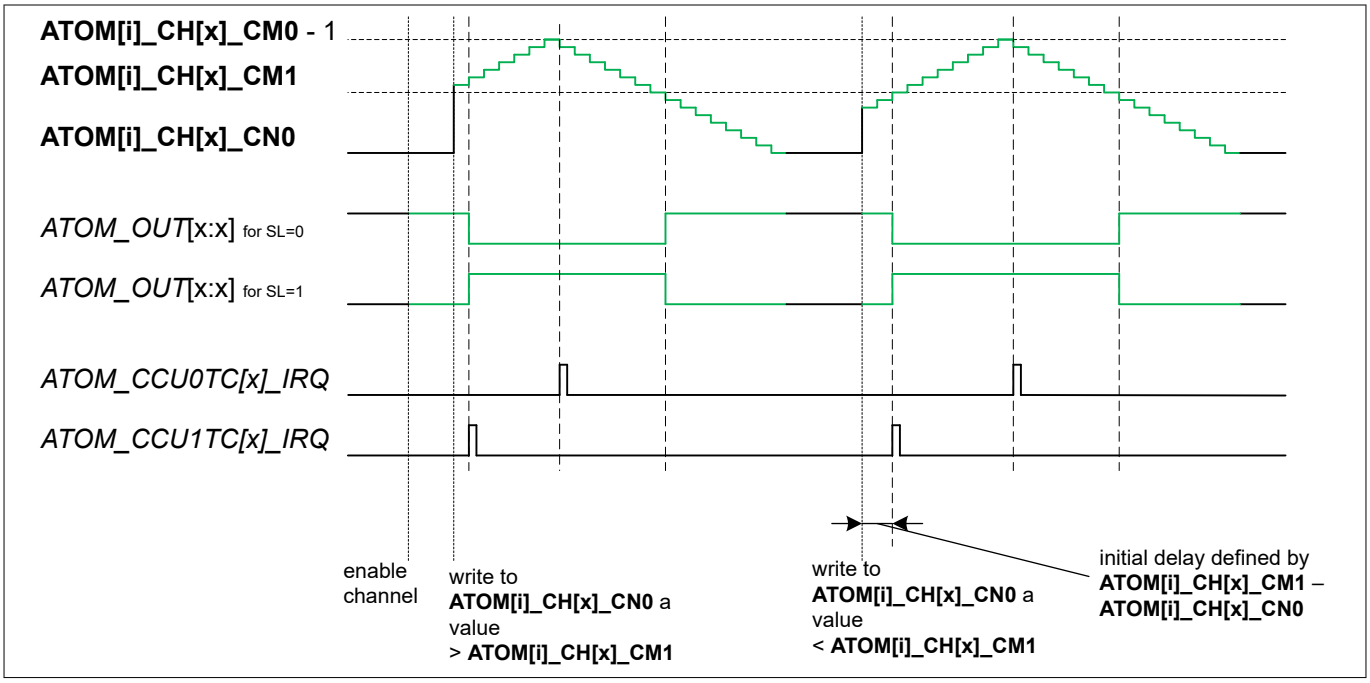


Figure 1087 PWM output with respect to configuration bit CLSi_ATOM_CHx_CTRL_SOMP.SL in one-shot counting up-down mode if CLSi_ATOM_CHx_CTRL_SOMP.OSM_TRIG= 0

Further output of single pulses can be started by writing to register CLSi_ATOM_CHx_CN0.CN0.

If a channel is configured to one-shot counting up-down mode and configuration bit CLSi_ATOM_CHx_CTRL_SOMP.OSM_TRIG is set to 1, the falling edge of the trigger signal ATOM_CH_TRIGIN[x:x] or EXT_TRIGIN_S (the synchronized signal of EXT_TRIGIN[x:x]) triggers start of incrementing CLSi_ATOM_CHx_CN0.CN0 and pulse generation. The output generation with interrupt signals is similar as mentioned above and shown in Figure 1088. If the counter CLSi_ATOM_CHx_CN0.CN0 is greater than or equal to CLSi_ATOM_CHx_CM1.CM1, the output ATOM_OUT is set to CLSi_ATOM_CHx_CTRL_SOMP.SL value. Otherwise, the output ATOM_OUT is set to !CLSi_ATOM_CHx_CTRL_SOMP.SL value. When CLSi_ATOM_CHx_CN0.CN0 is incrementing to reach CLSi_ATOM_CHx_CM1.CM1, an interrupt on ATOM_CCU1TC[x]_IRQ is generated. When CLSi_ATOM_CHx_CN0.CN0 changes its counting direction from up to down, an interrupt on ATOM_CCU0TC[x]_IRQ is generated.

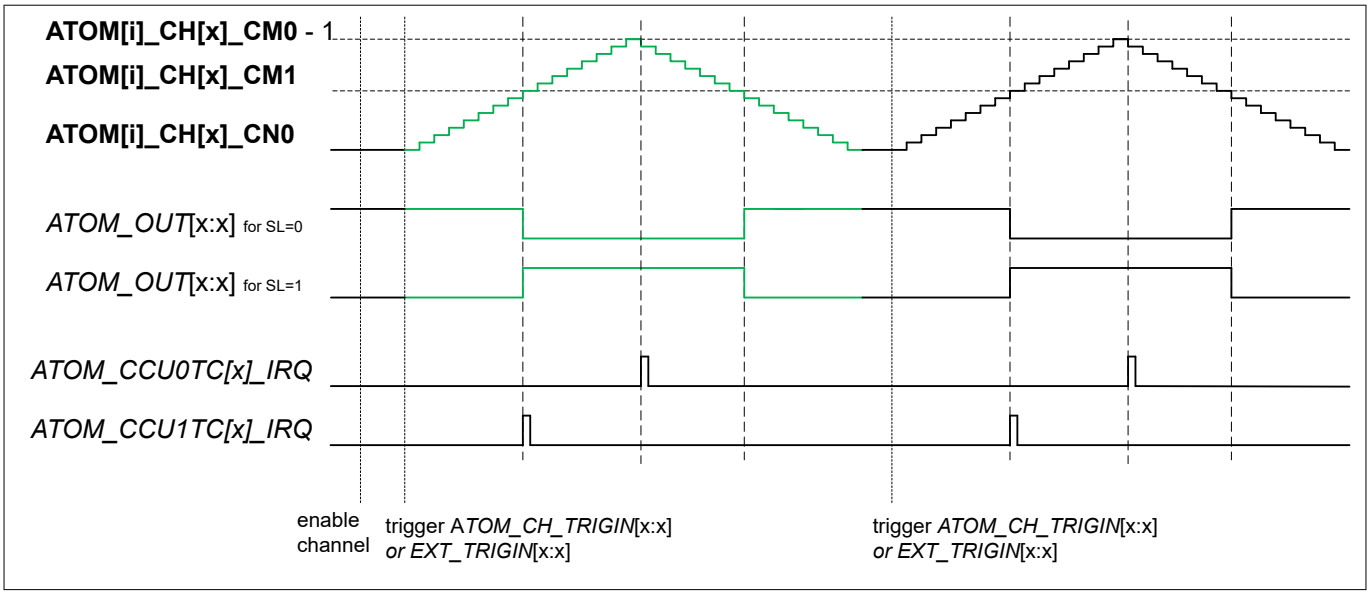


Figure 1088 PWM **output with respect to configuration bit** CLSi_ATOM_CHx_CTRL_SOMP**.**SL **in one-shot counting up-down mode if** CLSi_ATOM_CHx_CTRL_SOMP**.**OSM_TRIG**= 1**

Pulse count modulation mode

At the output ATOM_OUT a pulse count modulated signal can be generated instead of the simple PWM output signal in SOMP mode. This specific behavior is called pulse count modulation (PCM) mode. PCM mode is only available in SOMP mode when CLSi_ATOM_CHx_CTRL_SOMP**.**UDMODE=0.

PCM mode is enabled by setting bit CLSi_ATOM_CHx_CTRL_SOMP**.**BITREV[6:6] to 1.

PCM mode is only available for odd numbered **ATOM** channels 1,3,5,7.

With the configuration bit CLSi_ATOM_CHx_CTRL_SOMP**.**BITREV=1 a bit-reversing of the counter output CLSi_ATOM_CHx_CN0**.**CN0 is configured. In this case the bits LSB and MSB are swapped, the bits LSB+1 and MSB-1 are swapped, the bits LSB+2 and MSB-2 are swapped and so on.

The effect of bit-reversing of the CLSi_ATOM_CHx_CN0**.**CN0 register value is shown in the following Figure 1089.

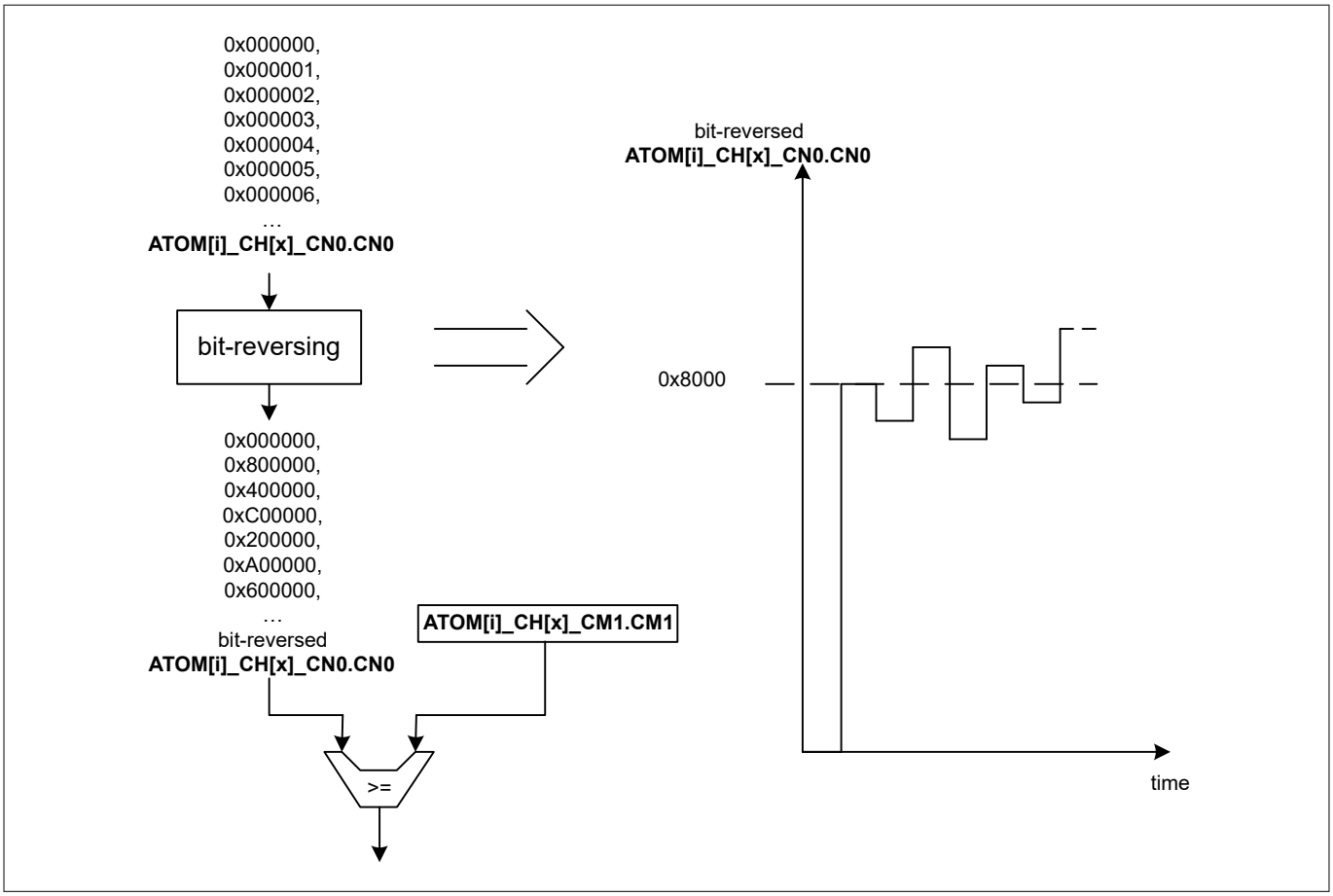


Figure 1089 Bit reversing of counter CLSi_ATOM_CHx_CN0.CN0 output

In the PCM mode the counter register CLSi_ATOM_CHx_CN0.CN0 is incremented by every clock tick depending on configured **CMU** clock resolution (CCM[i]_CLK_RES).

The output of counter register CLSi_ATOM_CHx_CN0.CN0 is first bit-reversed and then compared with the configured register value CLSi_ATOM_CHx_CM1.CM1.

When the bit-reversed value of register CLSi_ATOM_CHx_CN0.CN0 is greater than or equal to CLSi_ATOM_CHx_CM1.CM1, the output ATOM_OUT[x:x] is set to !CLSi_ATOM_CHx_CTRL.SL and otherwise set to CLSi_ATOM_CHx_CTRL.SL.

In PCM mode the CLSi_ATOM_CHx_CM0.CM0 register - in which the period is defined - normally has to be set to its maximum value $FFFFFF_H$.

To reduce time period of updating pulse width value in CLSi_ATOM_CHx_CM1.CM1 register, it is additionally possible to setup period value in CLSi_ATOM_CHx_CM0.CM0 register to smaller values than maximum value as described before.

Possible values for CLSi_ATOM_CHx_CM0.CM0 register are each even numbered values to the power of 2 e.g. 800000_H, 400000_H, 200000_H ... .

In this case the pulse width has to be configured in the following manner.

Depending on how much the period in CLSi_ATOM_CHx_CM0.CM0 register is decreased - means shifted right starting from 1000000_H - the pulse width in CLSi_ATOM_CHx_CM1.CM1 register has to be shifted to the left (= rotated: shift MSB back into LSB) with same value, e.g.:

period CLSi_ATOM_CHx_CM0.CM0 = 0010000_H -> shifted 8 bits right from 1000000_H

--> so pulse width has to be shifted 8 bits left:

e.g. 50% duty cycle: pulse width = 0008000_H -> shift 8 bits left -> CLSi_ATOM_CHx_CM1.CM1 = 800000_H

Table 2232 PCM examples

CLSi_ATOM_CHx_CM0.CM0 **(period)**	CLSi_ATOM_CHx_CM1.CM1 **(pulse width)**	**shift**	CLSi_ATOM_CHx_CM1.CM1 **(configured value)**
$FFFFFF_H$	800000_H	no shift	800000_H
800000_H	400000_H	shift 1 bit left	800000_H
400000_H	100000_H	shift 2 bits left	400000_H
200000_H	$0FFFFF_H$	shift 3 bits left	$7FFFF8_H$
100000_H	033333_H	shift 4 bits left	333330_H
080000_H	005555_H	shift 5 bits left	$0AAAA0_H$
000020_H	000008_H	shift 19 bits left	400000_H
000010_H	000005_H	shift 20 bits left	500000_H

In this mode the interrupt CLSi_ATOM_CHx_IRQ_NOTIFY.CCU1TC is set every time, when bit reverse value of CLSi_ATOM_CHx_CN0.CN0 is greater than or equal to CLSi_ATOM_CHx_CM1.CM1 which may be multiple times during one period. Therefore, from application point of view it is not useful to enable this interrupt.

Trigger generation

For applications with constant PWM period defined by CLSi_ATOM_CHx_CM0.CM0, it is not necessary to update the CLSi_ATOM_CHx_CM0.CM0 register regularly with CLSi_ATOM_CHx_SR0.SR0 register. For these applications, the CLSi_ATOM_CHx_SR0.SR0 register can be used to define an additional output signal and interrupt trigger.

If bit CLSi_ATOM_CHx_CTRL_SOMP.SR0_TRIG is set to 1, the register CLSi_ATOM_CHx_SR0.SR0 is no longer used as a shadow register for register CLSi_ATOM_CHx_CM0.CM0. Instead, CLSi_ATOM_CHx_SR0.SR0 is compared against CLSi_ATOM_CHx_CN0.CN0. If both are equal, a pulse of signal level '1' is generated at the output ATOM_OUT_T[x:x] of instance i, channel x. In this case, CLSi_ATOM_CHx_CM1.CM1 will still be synchronously updated from its shadow register CLSi_ATOM_CHx_SR1.SR1 if the update is enabled by setting CLSi_ATOM_AGC_GLB_CTRL.UPEN_CTRLk (k=0-7)=10_B.

The bit CLSi_ATOM_CHx_CTRL_SOMP.SR0_TRIG can only be set if bit CLSi_ATOM_CHx_CTRL_SOMP.RST_CCU0 of this channel is 0.

If bit CLSi_ATOM_CHx_CTRL_SOMP.SR0_TRIG=1 is set, the interrupt on ATOM_CCU1TC[x]_IRQ is set at the same time with pulse generation on ATOM_OUT_T in case of a compare equal match of CLSi_ATOM_CHx_SR0.SR0 = CLSi_ATOM_CHx_CN0.CN0. That means when the interrupt notify flag CLSi_ATOM_CHx_IRQ_NOTIFY.CCU1TC is no longer set on a compare match of CLSi_ATOM_CHx_CM1.CM1 and CLSi_ATOM_CHx_CN0.CN0. Instead, the CLSi_ATOM_CHx_IRQ_NOTIFY.CCU1TC interrupt notify flag is set in case of a compare equal match of CLSi_ATOM_CHx_SR0.SR0 and CLSi_ATOM_CHx_CN0.CN0. However, the interrupt generation of ATOM_CCU0TC[x]_IRQ is the same as the case of CLSi_ATOM_CHx_CTRL_SOMP.SR0_TRIG=0. The output generation with interrupt signals are shown in Figure 1090.

With configuration bit CLSi_ATOM_CHx_CTRL_SOMP.TRIG_PULSE, it is configurable if the output ATOM_OUT_T of instance i and channel x is high as long as CLSi_ATOM_CHx_CN0.CN0=CLSi_ATOM_CHx_SR0.SR0 (CLSi_ATOM_CHx_CTRL_SOMP.TRIG_PULSE=0) or if there will be only one pulse of length one cluster clock period when CLSi_ATOM_CHx_CN0.CN0 becomes CLSi_ATOM_CHx_SR0.SR0 (CLSi_ATOM_CHx_CTRL_SOMP.TRIG_PULSE=1). Figure 1090 shows the case of CLSi_ATOM_CHx_CTRL_SOMP.TRIG_PULSE=1.

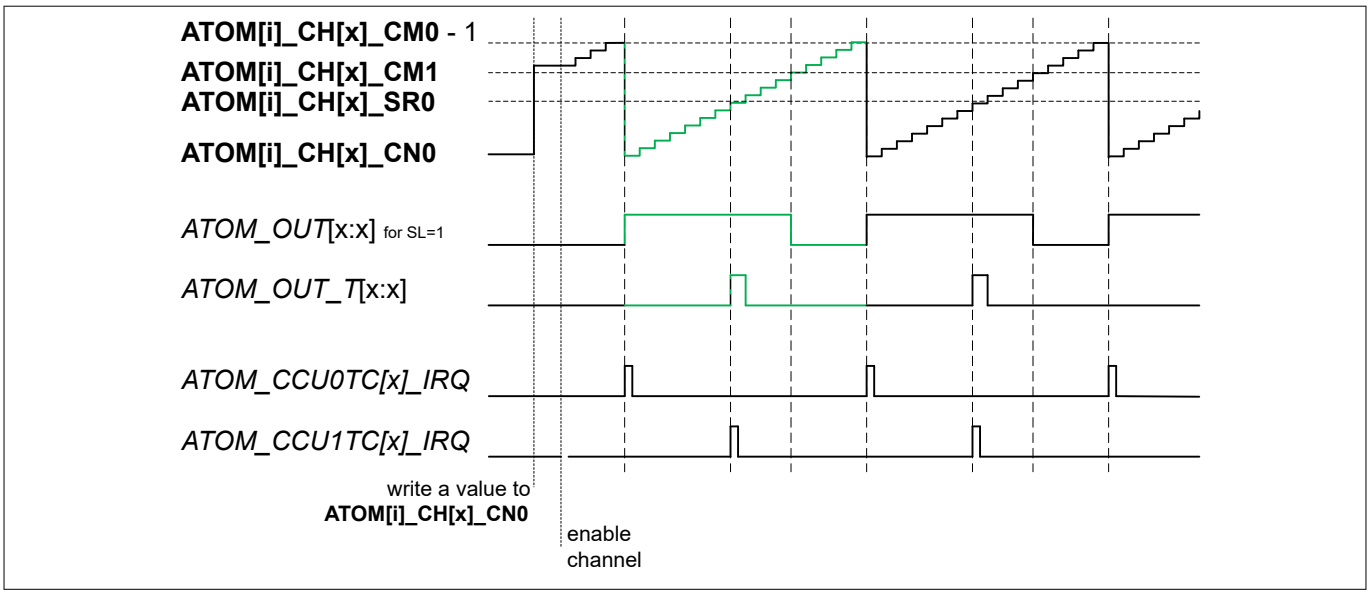


Figure 1090 Output behavior with respect to the CLSi_ATOM_CHx_CTRL_SOMP.TRIG_PULSE=1 in continuous counting up mode if CLSi_ATOM_CHx_CTRL_SOMP.SR0_TRIG=1

High-resolution PWM support (HRES mode)

ATOM also provides high-resolution PWM support mode (HRES mode). In HRES mode, **ATOM** offers the output signal ATOM_OUT_HRES[x] to control a circuit, e.g. a delay chain, which is needed to realize the high-resolution PWM and which has to be implemented outside the **eGTM**. Please refer to High-resolution PWM support for additional information of the high-resolution PWM support of the **eGTM**.

High-resolution mode in **ATOM** sub-module is only available in SOMP continuous counting up mode (CLSi_ATOM_CHx_CTRL_SOMP.UDMODE=0, CLSi_ATOM_CHx_CTRL_SOMP.SR0_TRIG=0). HRES mode is not supported in SOMP continuous counting up-down mode or when CLSi_ATOM_CHx_CTRL_SOMP.SR0_TRIG=1.

The high-resolution PWM support is not available in **ATOM** SOMP one-shot mode. If one-shot mode is enabled by setting of CLSi_ATOM_CHx_CTRL_SOMP.OSM to 1, the high-resolution PWM support has to be disabled by setting of CLSi_ATOM_CHx_CTRL2.HRES to 0, because it is not supported.

The high-resolution PWM support is also not available in PCM mode (CLSi_ATOM_CHx_CTRL_SOMP.BITREV=1).

This output signal ATOM_OUT_HRES[x] has a width of five bits while the decimal value of these five bits defines the number of micro-steps, by which the output signal ATOM_OUT[x:x] will be delayed in a suitable circuit (e.g. delay chain) outside the **eGTM**.

The high-resolution support will be enabled by setting bit CLSi_ATOM_CHx_CTRL2.HRES = 1.

If HRES mode is switched off by setting the configuration bit CLSi_ATOM_CHx_CTRL2.HRES to zero, then the output signals ATOM_OUT_HRES[x] and ATOM_OUT_T_HRES[x] are clamped to zero.

The value for the PWM period, stored in the registers CLSi_ATOM_CHx_CM0.CM0 and CLSi_ATOM_CHx_SR0.SR0, and pulse width, stored in the registers CLSi_ATOM_CHx_CM1.CM1 and CLSi_ATOM_CHx_SR1.SR1, represent respectively the length of period and pulse width in number of cluster clock steps and micro-steps.

Hereby the number of cluster clock steps for the PWM output signal ATOM_OUT[x:x] are defined by the upper bits [23:5], whereas the lower 5 bits [4:0] determine the additional micro-steps, by which the output signal has to be delayed outside the **eGTM**. This is shown in Figure 1091.

Table 2233 Overview of the HRES supporting modes

Mode	HRES supported
ATOM SOMI Mode	No
ATOM SOMC Mode	No
ATOM SOMB Mode	No
ATOM SOMS Mode	No
ATOM SOMP Continuous Counting Up Mode	Yes
ATOM SOMP Continuous Counting Up−Down Mode	No
ATOM SOMP One-Shot Counting Up Mode	No
ATOM SOMP One-Shot Counting Up-Down Mode	No
ATOM SOMP Pulse Count Modulation Mode	No

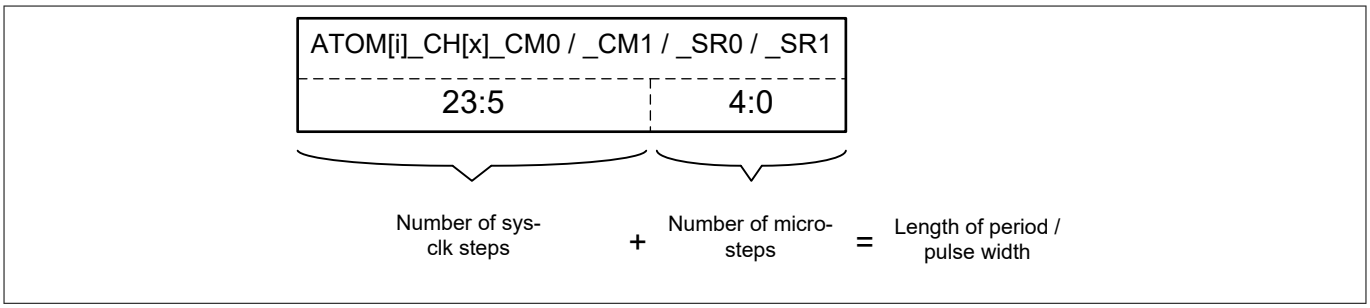
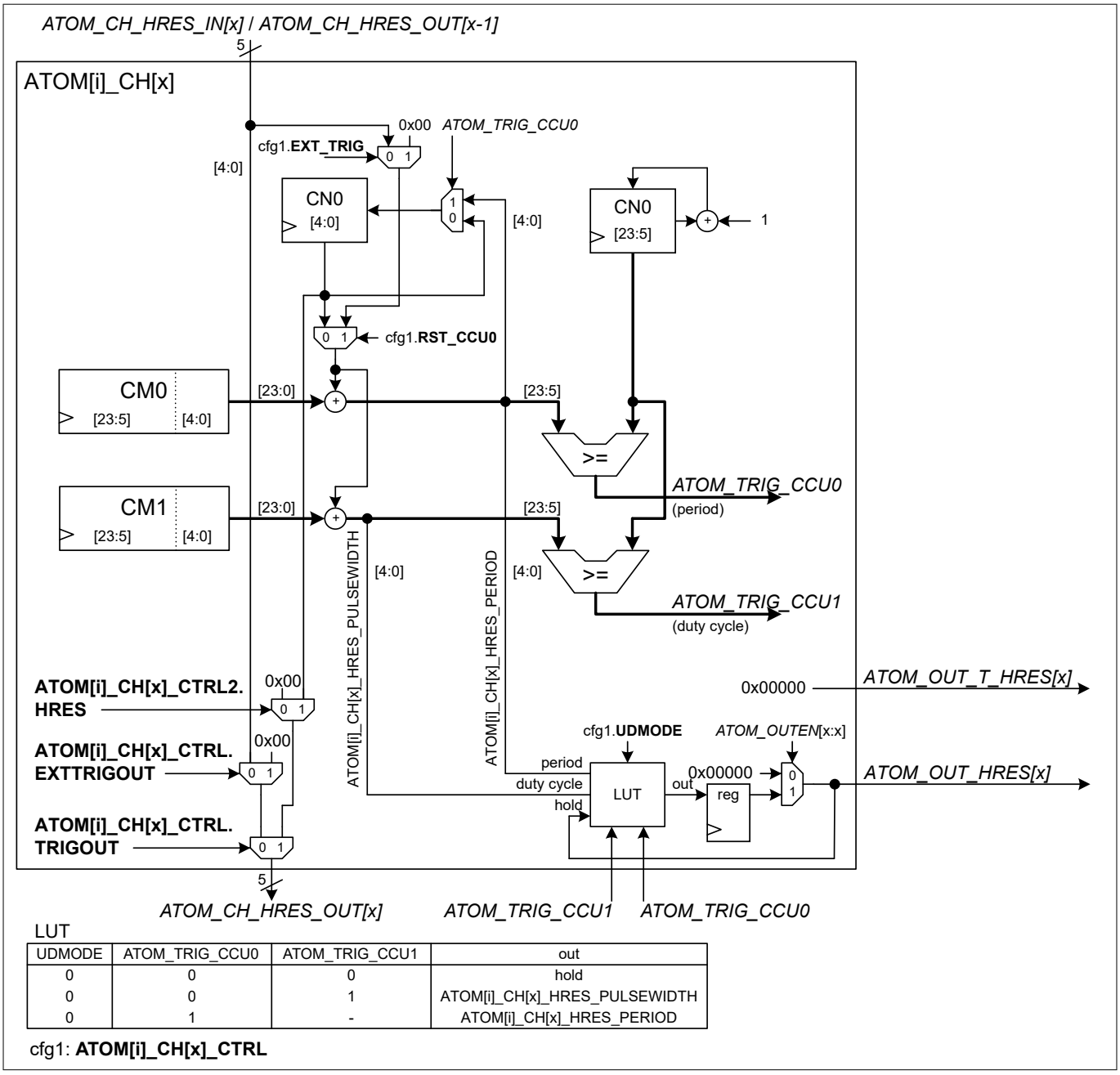


Figure 1091 Period and duty-cycle register in HRES mode

Figure 1092 shows the high-level schematic of the generation of the output signal ATOM_OUT_HRES[x], which defines the number of micro-steps mentioned above. If the **ATOM** channel x has HRES support with the configuration of x<ATOM_HIGH_RES *[i]*, a HRES trigger chain of ATOM_CH_HRES_IN[x] / ATOM_CH_HRES_OUT[x] is available together with the trigger chain of ATOM_CH_TRIGIN[x:x]/ ATOM_CH_TRIGOUT[x:x]. As shown in Figure 1074, the HRES trigger chain signals are routed through all **ATOM** channels and instances that have HRES support. ATOM0 channel 0 is driven by the output of ATOM0 channel 7 with a register in the trigger chain, which means a delay of one cluster clock period. The pipeline registers in the trigger chain of ATOM_CH_TRIGIN[x:x] /ATOM_CH_TRIGOUT[x:x] as mentioned in Internal trigger interface are also inserted in the trigger chain of ATOM_CH_HRES_IN[x]/ATOM_CH_HRES_OUT[x].

The channels x without HRES support (x>=ATOM_HRES_RES[i]) still have the HRES trigger chain of ATOM_CH_HRES_IN[x] / ATOM_CH_HRES_OUT[x] across these channels. The ATOM_CH_HRES_IN[x] coming from *ATOM_CH_HRES_OUT*[x-1] (previous channel) is directly connected to ATOM_CH_HRES_OUT[x] that is connected to *ATOM_CH_HRES_IN*[x+1] (next channel). There is no specific HRES functionalities in these channels without HRES support. Both HRES output signals ATOM_OUT_HRES[x] and ATOM_OUT_T_HRES[x] of these channels (x>=ATOM_HIGH_RES *[i]*) are clamped to zero.

When CLSi_ATOM_CHx_CN0.CN0 >= CLSi_ATOM_CHx_CM0.CM0-1, ATOM_TRIG_CCU0 =1 will be set and then ATOM_OUT[x:x] will be set to CLSi_ATOM_CHx_CTRL.SL in the next **CMU** period. Similarly, when CLSi_ATOM_CHx_CN0.CN0 >= CLSi_ATOM_CHx_CM1.CM1-1, ATOM_TRIG_CCU1 =1 will be set and then ATOM_OUT[x:x] will be set to !CLSi_ATOM_CHx_CTRL.SL. When CLSi_ATOM_CHx_CN0.CN0 < CLSi_ATOM_CHx_CM0.CM0-1, ATOM_TRIG_CCU0 will be reset as 0. Similarly, when CLSi_ATOM_CHx_CN0.CN0 < CLSi_ATOM_CHx_CM1.CM1-1, ATOM_TRIG_CCU1 will be reset to 0. According to the unit LUT in Figure 1092, the corresponding output signal ATOM_OUT_HRES[x] is generated together with each edge of PWM output signal ATOM_OUT[x:x].



LUT

UDMODE	ATOM_TRIG_CCU0	ATOM_TRIG_CCU1	out
0	0	0	hold
0	0	1	ATOM[i]_CH[x]_HRES_PULSEWIDTH
0	1	-	ATOM[i]_CH[x]_HRES_PERIOD

cfg1: **ATOM[i]_CH[x]_CTRL**

Figure 1092 High-level schematic view of ATOM_OUT_HRES[x] calculation

As shown in Figure 1092, when it is configured as continuous counting up mode (CLSi_ATOM_CHx_CTRL_SOMP.UDMODE=0) and CLSi_ATOM_CHx_CTRL_SOMP.RST_CCU0=0, the output signal ATOM_OUT_HRES[x] has to be calculated based on the internal signal ATOM[i]_CH[x]_HRES_PERIOD and ATOM[i]_CH[x]_HRES_PULSEWIDTH in the following manner (here, t means the index of a PWM period):

- period: ATOM[i]_CH[x]_HRES_PERIOD = CLSi_ATOM_CHx_CN0.CN0[4:0] + CLSi_ATOM_CHx_CM0.CM0[4:0]
- pulse width: ATOM[i]_CH[x]_HRES_PULSEWIDTH = CLSi_ATOM_CHx_CN0.CN0[4:0] + CLSi_ATOM_CHx_CM1.CM1[4:0]
- with: $(\text{CLSi_ATOM_CHx_CN0.CN0}[4:0])_{t+1} = (\text{CLSi_ATOM_CHx_CN0.CN0}[4:0])_t + (\text{CLSi_ATOM_CHx_CM0.CM0}[4:0])_t$

The signal diagram in Figure 1093 shows an example for the output generation in HRES mode when CLSi_ATOM_CHx_CTRL_SOMP.RST_CCU0=0 and CLSi_ATOM_CHx_CTRL_SOMP.SL=1. The PWM output signal ATOM_OUT[x:x] with cluster clock steps is generated based on the upper bits [23:5] of

CLSi_ATOM_CHx_CM0.CM0 and CLSi_ATOM_CHx_CM1.CM1 according to the same way of continuous counting up mode. Please refer to Figure 1081 for more details. The output signal ATOM_OUT_HRES[x] can be calculated based on the lower bits [4:0] of CLSi_ATOM_CHx_CM0.CM0 and CLSi_ATOM_CHx_CM1.CM1 according to Figure 1092 and the formulas above. As shown in Figure 1093, the value of ATOM_OUT_HRES[x] represents the number of micro-steps that should be taken by the external circuit to delay each edge of ATOM_OUT[x:x]. In this way, the delayed PWM signal ATOM_EXT_HRES[x:x] achieves higher resolution.

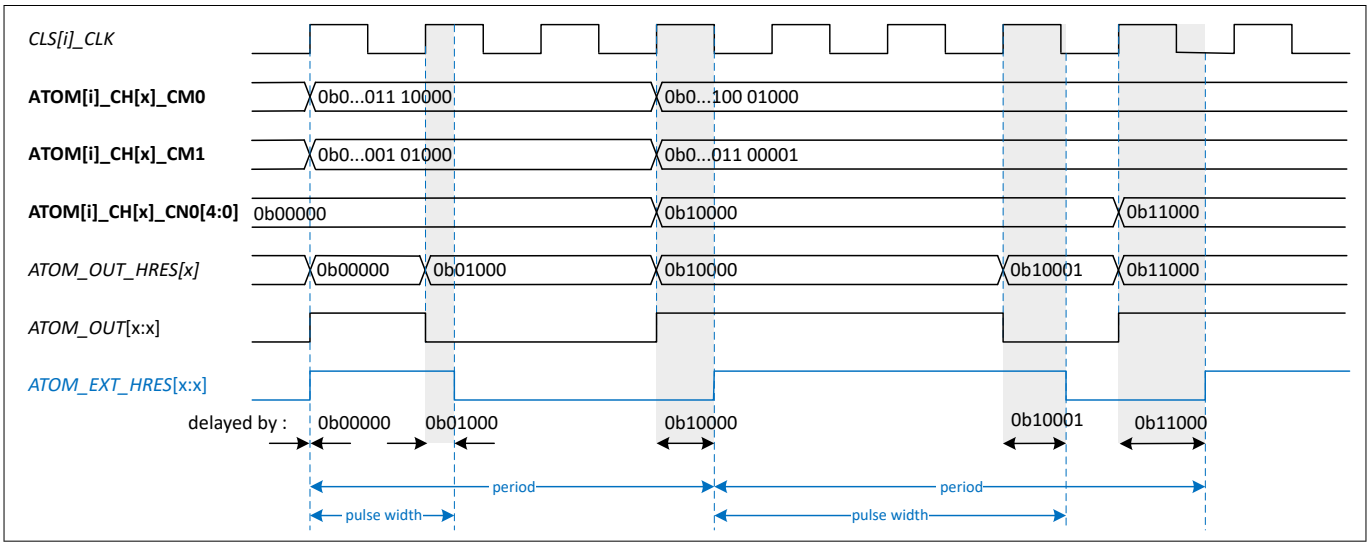


Figure 1093 Signals in HRES mode when CLSi_ATOM_CHx_CTRL_SOMP.RST_CCU0=0 and CLSi_ATOM_CHx_CTRL_SOMP.SL=1

The HRES mode is also available in continuous counting up mode when CLSi_ATOM_CHx_CTRL_SOMP.RST_CCU0=1.

In this case (CLSi_ATOM_CHx_CTRL_SOMP.RST_CCU0 = 1), if CLSi_ATOM_CHx_CTRL_SOMP.EXT_TRIG =0, the internal trigger signal ATOM_CH_HRES_IN[x] from the preceding channel is used for calculation instead of CLSi_ATOM_CHx_CN0.CN0[4:0] :

- period: ATOM[i]_CH[x]_HRES_PERIOD = ATOM_CH_HRES_IN[x] + CLSi_ATOM_CHx_CM0.CM0[4:0]
- pulse width: ATOM[i]_CH[x]_HRES_PULSEWIDTH = ATOM_CH_HRES_IN[x] + CLSi_ATOM_CHx_CM1.CM1[4:0]

The signal diagram in Figure 1094 shows an example for the output generation in HRES mode when CLSi_ATOM_CHx_CTRL_SOMP.RST_CCU0=1, CLSi_ATOM_CHx_CTRL_SOMP.EXT_TRIG=0 and CLSi_ATOM_CHx_CTRL_SOMP.SL=1. The PWM output signal ATOM_OUT[x:x] with cluster clock steps is generated based on the upper bits [23:5] of CLSi_ATOM_CHx_CM0.CM0 and CLSi_ATOM_CHx_CM1.CM1 according to the same way of continuous counting up mode. Please refer to Figure 1082 for more details. The output signal ATOM_OUT_HRES[x] representing the number of micro-steps can be calculated based on the lower bits [4:0] of CLSi_ATOM_CHx_CM0.CM0, CLSi_ATOM_CHx_CM1.CM1 and ATOM_CH_HRES_IN[x] according to Figure 1092 and the formulas above. Then the external circuit can take the value of ATOM_OUT_HRES[x] to generate the delayed signal ATOM_EXT_HRES[x:x].

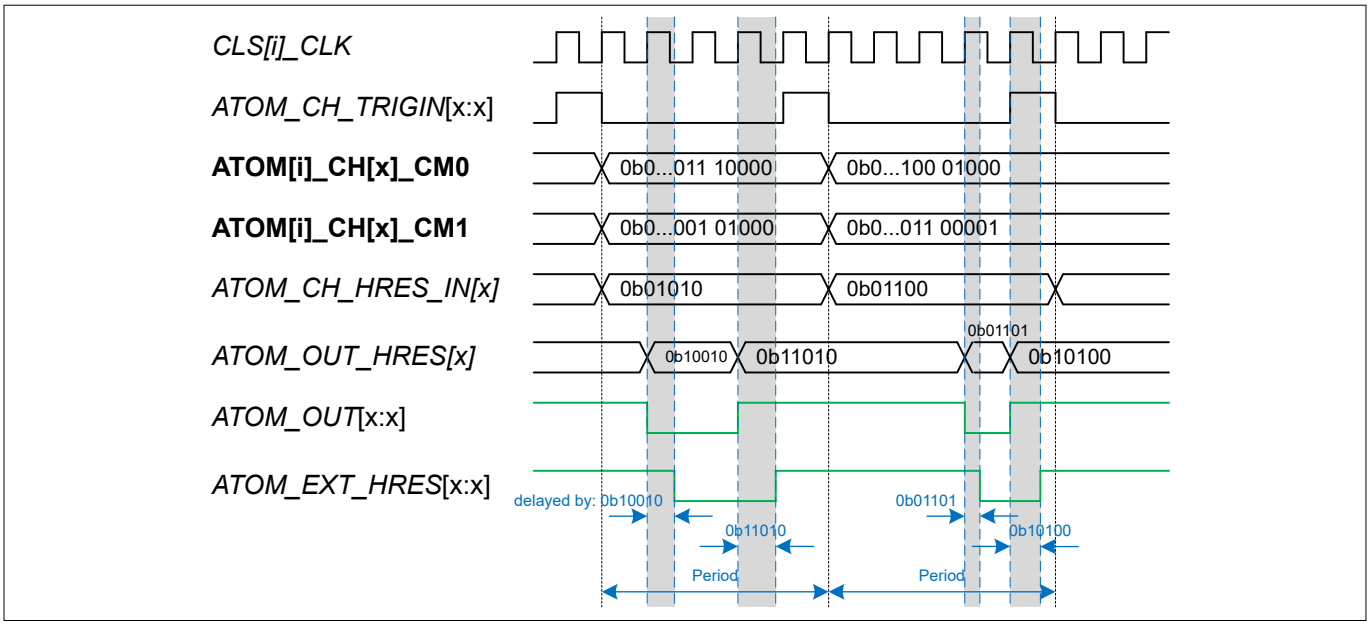


Figure 1094 Signals in HRES mode when CLSi_ATOM_CHx_CTRL_SOMP.RST_CCU0=1, CLSi_ATOM_CHx_CTRL_SOMP.EXT_TRIG=0 and CLSi_ATOM_CHx_CTRL_SOMP.SL=1

If CLSi_ATOM_CHx_CTRL_SOMP.EXT_TRIG =1, the HRES output signal is only depending on CLSi_ATOM_CHx_CM0.CM0[4:0] and CLSi_ATOM_CHx_CM1.CM1[4:0]:

- period: ATOM[i]_CH[x]_HRES_PERIOD = 00_B + CLSi_ATOM_CHx_CM0.CM0[4:0]
- pulse width: ATOM[i]_CH[x]_HRES_PULSEWIDTH = 00_B + CLSi_ATOM_CHx_CM1.CM1[4:0]

The signal diagram in Figure 1095 shows an example for the output generation in HRES mode when CLSi_ATOM_CHx_CTRL_SOMP.RST_CCU0=1, CLSi_ATOM_CHx_CTRL_SOMP.EXT_TRIG=1 and CLSi_ATOM_CHx_CTRL_SOMP.SL=1. Similarly, ATOM_OUT[x:x] is normal continuous counting up mode PWM signal with cluster clock steps, while ATOM_OUT_HRES[x] represents the number of micro-steps for generating ATOM_EXT_HRES[x:x].

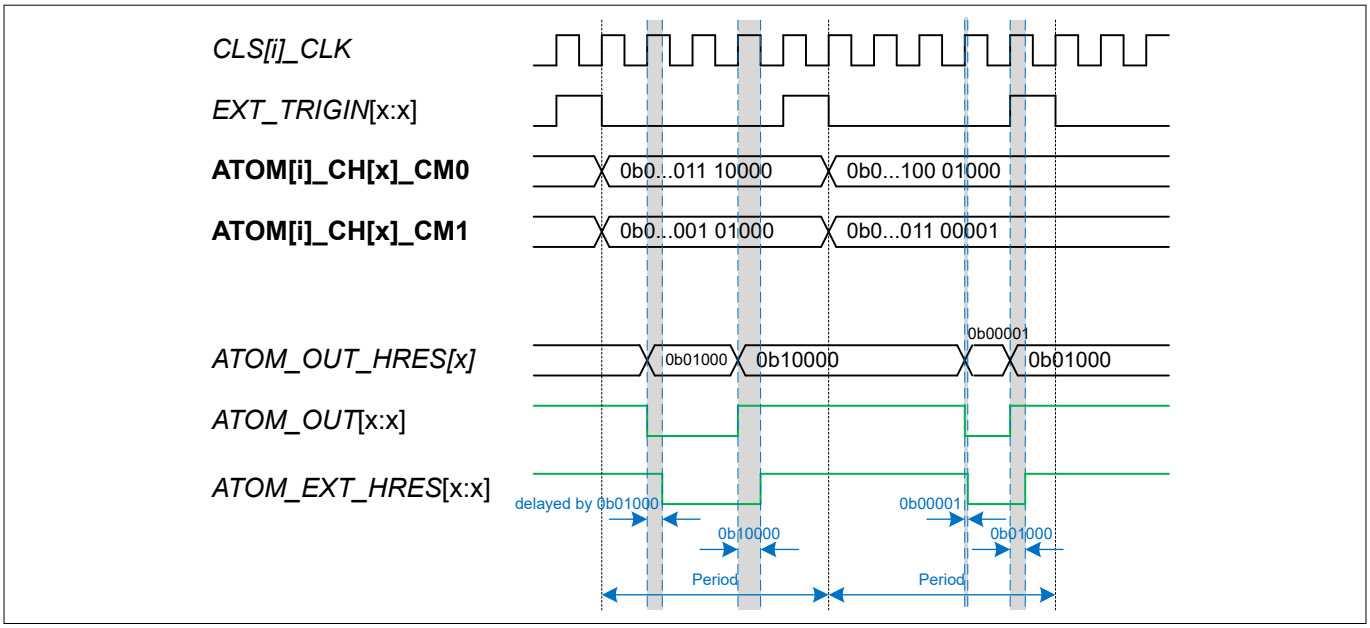


Figure 1095 Signals in HRES mode when CLSi_ATOM_CHx_CTRL_SOMP.RST_CCU0=1, CLSi_ATOM_CHx_CTRL_SOMP.EXT_TRIG=1 and CLSi_ATOM_CHx_CTRL_SOMP.SL=1

The following restrictions and characteristics must be considered, if the HRES mode is enabled:

- The selected clock resolution source CLSi_ATOM_CHx_CTRL_SOMP.CLK_SRC of CCM[i]_CLK_RES[x:x] has to be configured to the cluster clock CLS[j]_CLK.
- The maximum possible period is reduced to 2^{19} cluster clock-cycles.
- Only one edge on ATOM_OUT[x:x] with one value of ATOM_OUT_HRES[x] per cluster clock-cycle is possible.

In order to fulfill the constraint above, there are more constraints when CLSi_ATOM_CHx_CTRL_SOMP.RST_CCU0=0:

- For normal PWM signal, the constraint CLSi_ATOM_CHx_CM0.CM0 >= 64 and 32<=CLSi_ATOM_CHx_CM1.CM1 <= CLSi_ATOM_CHx_CM0.CM0 - 32 must be fulfilled.
- For 0% duty cycle PWM signal (keep !CLSi_ATOM_CHx_CTRL_SOMP.SL), the constraint CLSi_ATOM_CHx_CM0.CM0 >= 64 and CLSi_ATOM_CHx_CM1.CM1 = 0 must be fulfilled.
- For 100% duty cycle PWM signal (keep CLSi_ATOM_CHx_CTRL_SOMP.SL), the constraint CLSi_ATOM_CHx_CM0.CM0 = CLSi_ATOM_CHx_CM1.CM1 >= 64 must be fulfilled.

However, when CLSi_ATOM_CHx_CTRL_SOMP.RST_CCU0=1, there are other constraints and some of them are different from the case when HRES option is disabled:

- When configuring CLSi_ATOM_CHx_CM0.CM0[23:5]=0 or CLSi_ATOM_CHx_CM1.CM1[23:5]=0, the lower HRES bits CLSi_ATOM_CHx_CM0.CM0[4:0]=0 or CLSi_ATOM_CHx_CM1.CM1[4:0] must also be 0.
- It is allowed to configure the registers as CLSi_ATOM_CHx_CM1.CM1 <= CLSi_ATOM_CHx_CM0.CM0 - 32 or CLSi_ATOM_CHx_CM1.CM1 = CLSi_ATOM_CHx_CM0.CM0.
- The configuration CLSi_ATOM_CHx_CM1.CM1 > CLSi_ATOM_CHx_CM0.CM0 is not allowed.

Furthermore, if CLSi_ATOM_CHx_CTRL_SOMP.RST_CCU0=1 and CLSi_ATOM_CHx_CTRL_SOMP.EXT_TRIG=0, the channel is triggered by the preceding channel with HRES output trigger chain. Besides the constraints above, there are more constraints between the triggered channel and the triggering channel. Here, the registers of the triggered channel are noted with a subscript "triggered" while the triggering of the preceding channel is marked with "master".

- The configuration must fulfill (CLSi_ATOM_CHx_CM0.CM0)$_{triggered}$ <= (CLSi_ATOM_CHx_CM0.CM0)$_{master}$ or (CLSi_ATOM_CHx_CM0.CM0[23:5])$_{triggered}$ > (CLSi_ATOM_CHx_CM0.CM0[23:5])$_{master}$.
- If (CLSi_ATOM_CHx_CM0.CM0[23:5])$_{triggered}$ = (CLSi_ATOM_CHx_CM0.CM0[23:5])$_{master}$, (CLSi_ATOM_CHx_CM1.CM1)$_{triggered}$ must be greater or equal to 32.

When HRES mode is enabled, ATOM_CH_HRES_IN[x]/ATOM_CH_HRES_OUT[x] are always generated synchronously together with ATOM_CH_TRIGIN[x:x]/ATOM_CH_TRIGOUT[x:x] and may be delayed by same pipeline registers according to device configuration parameter configuration as defined in Internal trigger interface. If one channel triggers several following several channels, the 4 trigger signals ATOM_CH_TRIGIN[x:x]/ ATOM_CH_TRIGOUT[x:x] with ATOM_CH_HRES_IN[x]/ATOM_CH_HRES_OUT[x] build up a trigger chain from the triggering channel to the triggered channels. In this case, if there is no pipeline register in the trigger chain, these triggered channels can be configured with same operation registers and synchronously triggered to generate consistent PWM signals. However, if there are pipeline registers that may introduce delay of number of cluster clock periods, it is hard to simultaneously trigger several channels to generate consistent PWM signals. The reason for that is in HRES mode the operation clock frequency is the cluster clock and each pipeline register may cause a delay or shift of one cluster clock on the trigger signal pulse and this delay must be taken into account.

ATOM signal output mode serial (SOMS)

In **ATOM** Signal Output Mode Serial (SOMS) the **ATOM** channel acts as a serial output shift register where the content of the CLSi_ATOM_CHx_CM1.CM1 register in the CCU1 unit is shifted out whenever the unit is triggered by the clock source, which is selected by CLSi_ATOM_CHx_CTRL_SOMS.CLK_SRC. The shift direction is configurable with the bit CLSi_ATOM_CHx_CTRL_SOMS.ACB0.

The data inside the CLSi_ATOM_CHx_CM1.CM1 register has to be aligned according to the selected shift direction in the CLSi_ATOM_CHx_CTRL_SOMS.ACB0/CLSi_ATOM_CHx_STAT.ACBI[0:0] bit. This means that

when a right shift is selected, the data word has to be aligned to bit 0 of the CLSi_ATOM_CHx_CM1.CM1 register and when a left shift is selected, the data has to be aligned to bit 23 of the CLSi_ATOM_CHx_CM1.CM1 register.

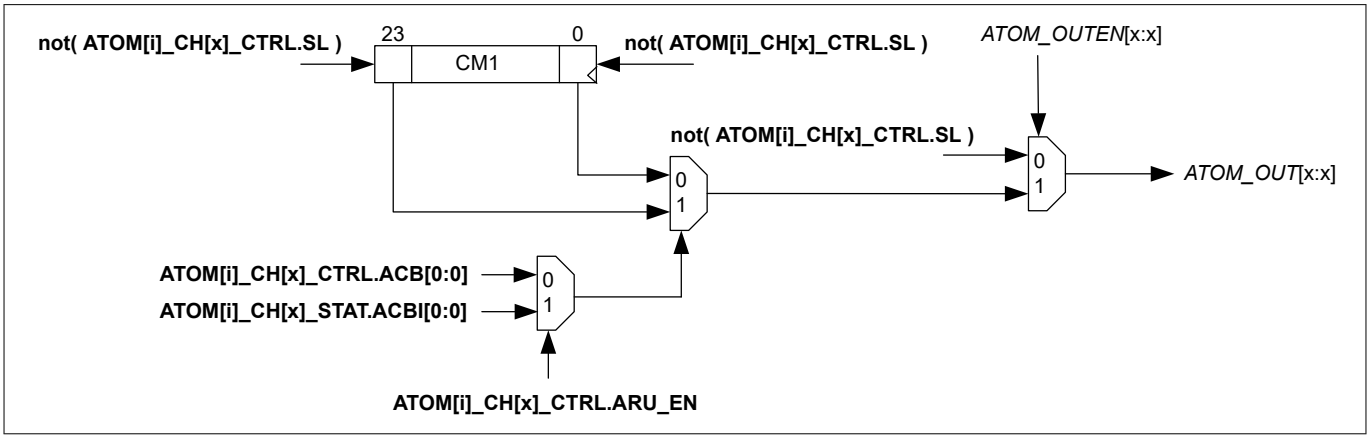


Figure 1096 SOMS mode output generation

Figure 1096 shows the output generation in case of SOMS mode is selected.

In SOMS mode CCU0 runs in counter/compare mode and counts the number of bits shifted out so far. The total number of bits that should be shifted is defined as CLSi_ATOM_CHx_CM0.CM0. The total number of bits that are visible at ATOM_OUT is CLSi_ATOM_CHx_CM0.CM0+1.

When the output is disabled the ATOM_OUT is set to the inverse CLSi_ATOM_CHx_CTRL_SOMS.SL bit definition.

When the content of the CLSi_ATOM_CHx_CM1.CM1 register is shifted out, the inverse signal level is shifted into the CLSi_ATOM_CHx_CM1.CM1 register.

When the output is enabled while CLSi_ATOM_AGC_GLB_CTRL.UPEN_CTRLk (k=0-7) **(k=x)** is disabled, the ATOM_OUT signal level is defined by bit CLSi_ATOM_CHx_CM1.CM1[23:0], dependent on the shift direction defined by CLSi_ATOM_CHx_CTRL_SOMS.ACB0 or CLSi_ATOM_CHx_STAT.ACBI[0:0] register setting. Figure 1097 should clarify the **ATOM** channel startup behavior in this case for right shift. For left shift the CLSi_ATOM_CHx_CM1.CM1[0:0] in Figure 1097 has to be replaced by CLSi_ATOM_CHx_CM1.CM1[23:23].

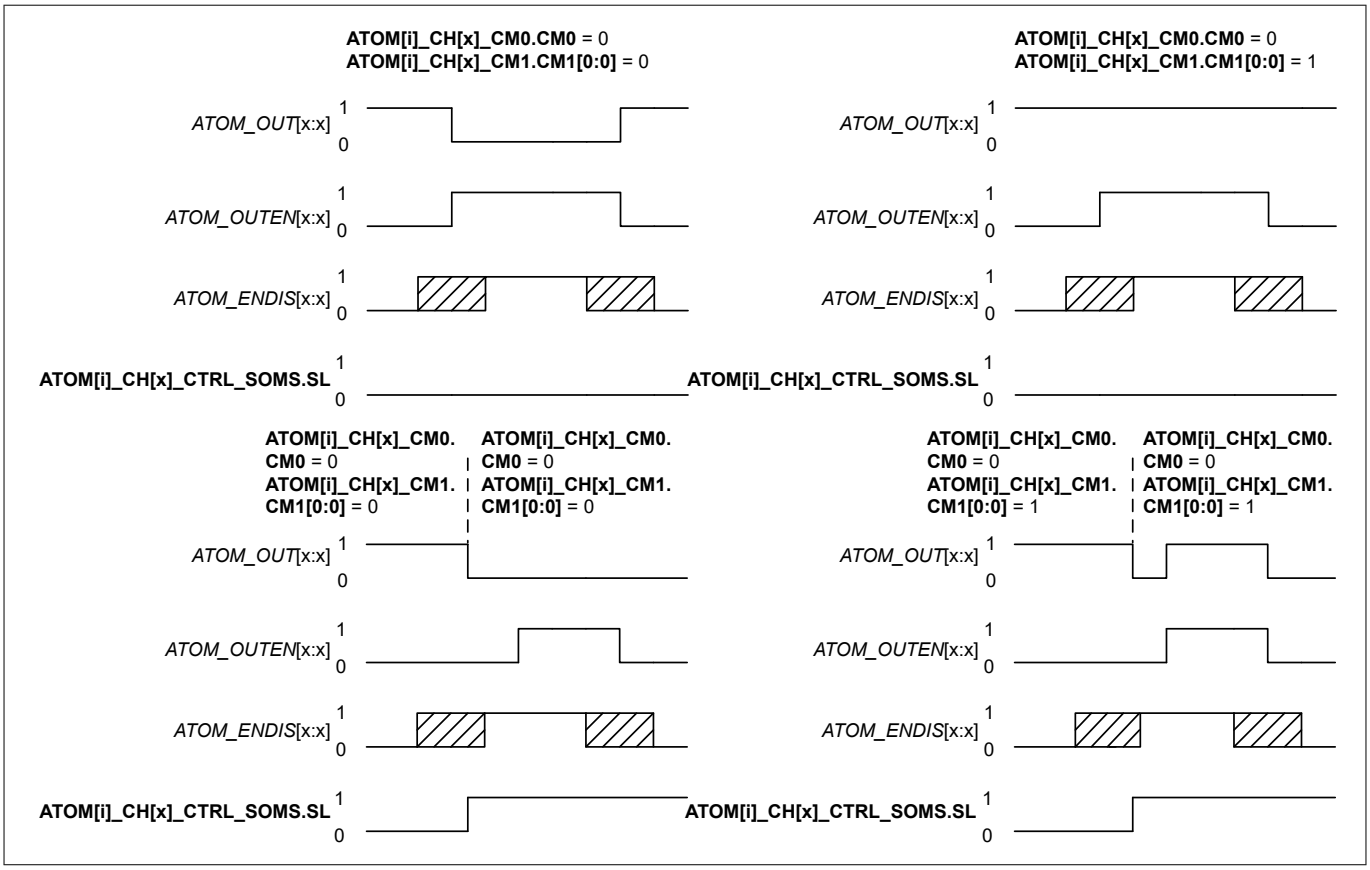


Figure 1097 SOMS output signal level at startup, CLSi_ATOM_AGC_GLB_CTRL.UPEN_CTRLk (k=0-7) (k=x) disabled

If CLSi_ATOM_AGC_GLB_CTRL.UPEN_CTRLk (k=0-7) **(k=x)** is set and the channel is enabled, the output level is defined by bit CLSi_ATOM_CHx_CM1.CM1[0:0] or CLSi_ATOM_CHx_CM1.CM1[23:23] depending on the shift direction. Figure 1098 shows the output behavior in that case.

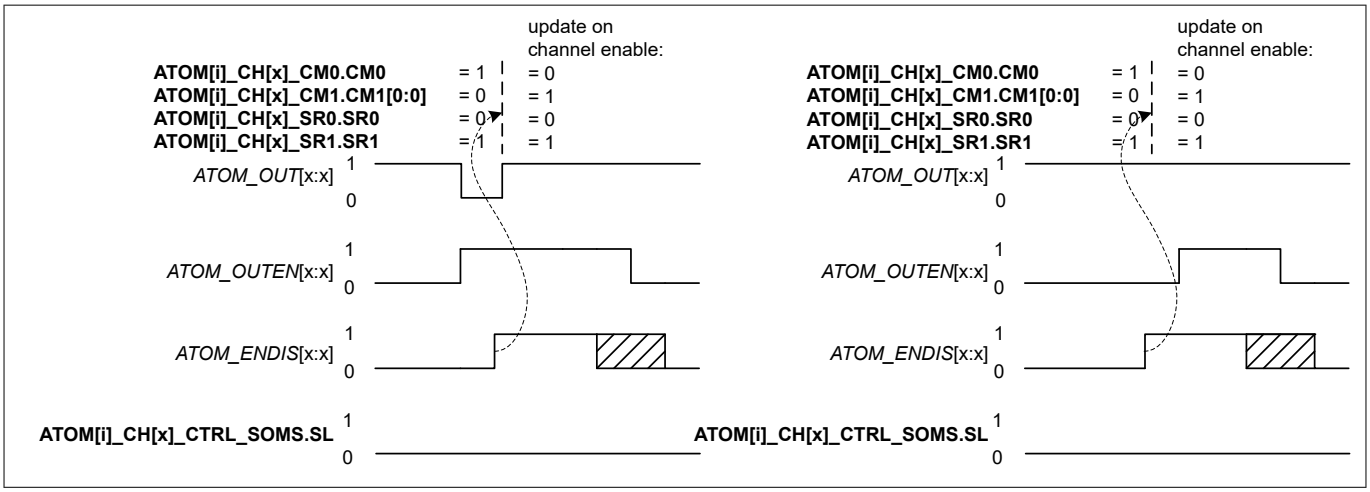


Figure 1098 SOMS output signal level at startup, CLSi_ATOM_AGC_GLB_CTRL.UPEN_CTRLk (k=0-7) (k=x) enabled

If bit CLSi_ATOM_AGC_GLB_CTRL.UPEN_CTRLk (k=0-7) **(k=x)** of a channel x is not set, the **ATOM** may stop shifting after CLSi_ATOM_CHx_CN0.CN0 has reached CLSi_ATOM_CHx_CM0.CM0.

In this case also no automatic restart of shifting is possible.

SOMS mode with CLSi_ATOM_CHx_CTRL_SOMS.OSM = 0, CLSi_ATOM_AGC_GLB_CTRL.UPEN_CTRLk (k=0-7) (k=x) = 1:

In case bit CLSi_ATOM_CHx_CTRL_SOMS.OSM is not set, the **ATOM** channel updates its CLSi_ATOM_CHx_CM0.CM0/CLSi_ATOM_CHx_CM1.CM1 register with the content of the CLSi_ATOM_CHx_SR0.SR0/CLSi_ATOM_CHx_SR1.SR1 register and restarts shifting immediately. The first bit of new CLSi_ATOM_CHx_CM1.CM1 register value will be applied at the output without any gap to the last bit of the previous CLSi_ATOM_CHx_CM1.CM1 register value.

SOMS mode with CLSi_ATOM_CHx_CTRL_SOMS.OSM = 1, CLSi_ATOM_AGC_GLB_CTRL.UPEN_CTRLk (k=0-7) (k=x) = 1:

In case bit CLSi_ATOM_CHx_CTRL_SOMS.OSM is set, the **ATOM** channel stops shifting when CLSi_ATOM_CHx_CN0.CN0 reaches CLSi_ATOM_CHx_CM0.CM0 and no update of CLSi_ATOM_CHx_CM0.CM0 and CLSi_ATOM_CHx_CM1.CM1 is performed.

Then, the shifting of the channel can be restarted again by writing a zero to the CLSi_ATOM_CHx_CN0.CN0 register again.

The writing of a zero to CLSi_ATOM_CHx_CN0.CN0 also causes an immediate update of CLSi_ATOM_CHx_CM0.CM0/CLSi_ATOM_CHx_CM1.CM1 register with the content of CLSi_ATOM_CHx_SR0.SR0/ CLSi_ATOM_CHx_SR1.SR1 register.

Restarting the shifting by writing a zero to CLSi_ATOM_CHx_CN0.CN0 can also be done while a shift cycle is already active. Then an immediate update of CLSi_ATOM_CHx_CM0.CM0/CLSi_ATOM_CHx_CM1.CM1 register with the content of CLSi_ATOM_CHx_SR0.SR0/CLSi_ATOM_CHx_SR1.SR1 register is done followed by a new shift operation.

SOMS mode with CLSi_ATOM_CHx_CTRL_SOMS.OSM = 0, CLSi_ATOM_AGC_GLB_CTRL.UPEN_CTRLk (k=0-7) (k=x) = 0:

In case of bit CLSi_ATOM_CHx_CTRL_SOMS.OSM is not set and the update enable bit CLSi_ATOM_AGC_GLB_CTRL.UPEN_CTRLk (k=0-7) **(k=x)** is not set, no update of CLSi_ATOM_CHx_CM0.CM0/ CLSi_ATOM_CHx_CM1.CM1 register is executed. The shifting is done continuously without reloading from shadow register CLSi_ATOM_CHx_SR0.SR0/CLSi_ATOM_CHx_SR1.SR1 and without any gap between the shift cycles.

SOMS mode with double output

If the mode bit DSO is additionally set in SOMS mode (in register CLSi_ATOM_CHx_CTRL_SOMS), two 12-bit data streams can be shifted out on the outputs ATOM_OUT and ATOM_OUT_T of instance i and channel x in parallel.

This is achieved by splitting the register CLSi_ATOM_CHx_CM1.CM1 into two parts. The lower 12 bits are used as a shift register of output ATOM_OUT (i.e. bit 0 is assigned to output ATOM_OUT), the upper 12 bits are used as a shift register of output ATOM_OUT_T (i.e. bit 12 is assigned to output ATOM_OUT_T).

On bit CLSi_ATOM_CHx_CM1.CM1[23:11] the value !CLSi_ATOM_CHx_CTRL_SOMS.SL is shifted in.

In this mode only right shift is possible. Bit CLSi_ATOM_CHx_CTRL_SOMS.ACB0 is ignored.

This behavior is depicted in the following figure:

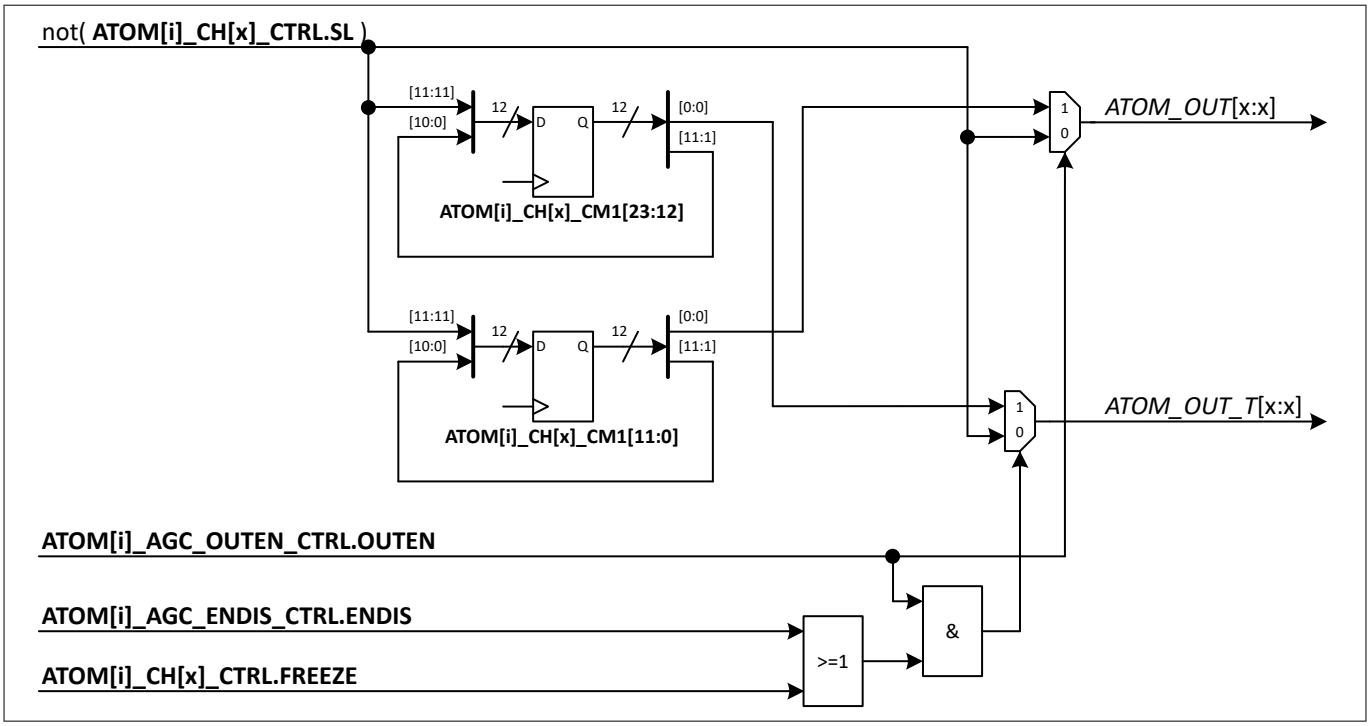


Figure 1099 Double output shift mode

Interrupts in SOMS mode

In **ATOM** Signal Output Mode Serial only the interrupt CLSi_ATOM_CHx_IRQ_NOTIFY.CCU0TC in case of CLSi_ATOM_CHx_CN0.CN0 >= CLSi_ATOM_CHx_CM0.CM0 is generated. The interrupt CLSi_ATOM_CHx_IRQ_NOTIFY.CCU1TC has no meaning and is not generated.

ATOM signal output mode buffered compare (SOMB)

Overview

In **ATOM** Signal Output Mode Buffered Compare (SOMB) the output action is performed according to the comparison result of the input values located in CLSi_ATOM_CHx_CM0.CM0 and/or CLSi_ATOM_CHx_CM1.CM1 registers and the two (three) time base values CCM[i]_TBU_TS0 or CCM[i]_TBU_TS1 (or CCM[i]_TBU_TS2) provided by the **TBU**. For a description of the time base generation please refer to the **TBU** specification in Time base unit (**TBU**). It is configurable, which of the two (three) time bases is to be compared with one or both values in CLSi_ATOM_CHx_CM0.CM0 and CLSi_ATOM_CHx_CM1.CM1.

The compare strategy of the two compare units CCU0 and CCU1 is controlled by the value of bit field of CLSi_ATOM_CHx_STAT.ACBI. This bit field is only readable over configuration interface.

The table below lists all valid control configurations for bit field CLSi_ATOM_CHx_STAT.ACBI.

Table 2234 ATOM SOMB compare strategies

CLSi_ATOM_CHx_STAT.ACBI[4:4]	CLSi_ATOM_CHx_STAT.ACBI[3:3]	CLSi_ATOM_CHx_STAT.ACBI[2:2]	CCU0/CCU1 comparison control
0	0	0	Reserved. Has no effect.
0	0	1	Reserved. Has no effect.

(table continues...)

Table 2234 (continued) ATOM SOMB compare strategies

CLSi_ATOM_CHx_STAT.ACBI[4:4]	CLSi_ATOM_CHx_STAT.ACBI[3:3]	CLSi_ATOM_CHx_STAT.ACBI[2:2]	CCU0/CCU1 comparison control
0	1	0	Compare in CCU0 only, use time base CCM[i]_TBU_TS0. Output signal level at the compare match event is defined by combination of CLSi_ATOM_CHx_CTRL_SOMB.SL and CLSi_ATOM_CHx_CTRL_SOMB.ACB10/ CLSi_ATOM_CHx_STAT.ACBI[1:0] bits (details see Table 2235).
0	1	1	Compare in CCU1 only, use time base CCM[i]_TBU_TS1 or CCM[i]_TBU_TS2. Output signal level at the compare match event is defined by combination of CLSi_ATOM_CHx_CTRL_SOMB.SL and CLSi_ATOM_CHx_CTRL_SOMB.ACB10/ CLSi_ATOM_CHx_STAT.ACBI[1:0] bits (details see Table 2235).
1	0	0	Serve Last: Compare in CCU0 and then in CCU1 using CCM[i]_TBU_TS0. Output signal level at the CCU0 compare match event is defined by combination of CLSi_ATOM_CHx_CTRL_SOMB.SL and CLSi_ATOM_CHx_STAT.ACBI[1:0] (details see Table 2235). At the CCU1 compare match event the output level is toggled.

(table continues...)

Table 2234 (continued) ATOM SOMB compare strategies

CLSi_ATOM_CHx_STAT.ACBI[4:4]	CLSi_ATOM_CHx_STAT.ACBI[3:3]	CLSi_ATOM_CHx_STAT.ACBI[2:2]	CCU0/CCU1 comparison control
1	0	1	Serve Last: Compare in CCU0 and then in CCU1 using CCM[i]_TBU_TS1 or CCM[i]_TBU_TS2. Output signal level at the CCU0 compare match event is defined by combination of CLSi_ATOM_CHx_CTRL_SOMB.SL and CLSi_ATOM_CHx_STAT.ACBI[1:0] (details see Table 2235). At the CCU1 compare match event the output level is toggled.
1	1	0	Serve Last: Compare in CCU0 using CCM[i]_TBU_TS0 and then in CCU1 using CCM[i]_TBU_TS1 or CCM[i]_TBU_TS2. Output signal level at the CCU1 compare match event is defined by combination of CLSi_ATOM_CHx_CTRL_SOMB.SL and CLSi_ATOM_CHx_STAT.ACBI[1:0] (details see Table 2235).
1	1	1	Cancels pending comparison

Based on the configured compare strategy CLSi_ATOM_CHx_STAT.ACBI[4:2], CCU0 unit always performs cyclic event compare between CLSi_ATOM_CHx_CM0.CM0 and the selected **TBU** time base CCM[i]_TBU_TS0/ CCM[i]_TBU_TS1/CCM[i]_TBU_TS2, while CCU1 unit always performs cyclic event compare between CLSi_ATOM_CHx_CM1.CM1 and the selected **TBU** time base CCM[i]_TBU_TS0/CCM[i]_TBU_TS1/ CCM[i]_TBU_TS2. Please refer to Cyclic event compare for more information about cyclic event compare strategy. When the selected time base is "cyclically greater than or equal to" the compare value of CLSi_ATOM_CHx_CM0.CM0, CCU0 compare match event is considered to have occurred. When the selected time base is "cyclically greater than or equal to" the compare value of CLSi_ATOM_CHx_CM1.CM1, CCU1 compare match event is considered to have occurred. Whenever CCU0 or CCU1 compare match event has occurred, an edge is generated on output ATOM_OUT[x:x], depending on the predefined signal level in CLSi_ATOM_CHx_CTRL_SOMB.SL bit in combination with two control bits CLSi_ATOM_CHx_STAT.ACBI[1:0].

If the scheduled compare match event in CCU0 and/or CCU1 have occurred and the CLSi_ATOM_CHx_SR0.SR0 / CLSi_ATOM_CHx_SR1.SR1 register contain new valid values, the registers CLSi_ATOM_CHx_CM0.CM0 and CLSi_ATOM_CHx_CM1.CM1 will be updated automatically with the content of the corresponding registers CLSi_ATOM_CHx_SR0.SR0 and CLSi_ATOM_CHx_SR1.SR1, the CLSi_ATOM_CHx_STAT.ACBI bit field will be updated with the content of internal *ACB_SR* register and the CLSi_ATOM_CHx_STAT.DV bit will be set. If the

registers CLSi_ATOM_CHx_SR0.SR0 / CLSi_ATOM_CHx_SR1.SR1 and CLSi_ATOM_CHx_CM0.CM0 / CLSi_ATOM_CHx_CM1.CM1 contain no valid value, the compare units will wait in the idle state.

On a compare match event in one of the compare units CCU0/CCU1 units the output ATOM_OUT[x:x] is set according to combination of the bits CLSi_ATOM_CHx_STAT.ACBI[1:0] and CLSi_ATOM_CHx_CTRL_SOMB.SL according to the following table.

Table 2235 ATOM SOMB output control by CLSi_ATOM_CHx_STAT.ACBI[1:0] and CLSi_ATOM_CHx_CTRL_SOMB.SL

SL	CLSi_ATOM_CHx_STAT.ACBI[1:1]	CLSi_ATOM_CHx_STAT.ACBI[0:0]	Output Behavior
0	0	0	No signal level change at output.
0	0	1	Set output signal level to 1.
0	1	0	Set output signal level to 0.
0	1	1	Toggle output signal level.
1	0	0	No signal level change at output.
1	0	1	Set output signal level to 0.
1	1	0	Set output signal level to 1.
1	1	1	Toggle output signal level.

In contrast to the SOMC mode, no timestamp value of **TBU** is captured in CLSi_ATOM_CHx_SR0.SR0 / CLSi_ATOM_CHx_SR1.SR1 register.

The flag CLSi_ATOM_CHx_STAT.DV of register CLSi_ATOM_CHx_STAT indicates that at least one of the CLSi_ATOM_CHx_CM0.CM0/CLSi_ATOM_CHx_CM1.CM1 register contains valid data and a compare event may be pending (if the channel is enabled).

The CLSi_ATOM_CHx_STAT.DV flag is reset if none of the CLSi_ATOM_CHx_CM0.CM0/CLSi_ATOM_CHx_CM1.CM1 registers contain valid data.

SOMB controlled over configuration interface

Writing to one of the CLSi_ATOM_CHx_CM0.CM0/CLSi_ATOM_CHx_CM1.CM1 registers automatically sets CLSi_ATOM_CHx_STAT.DV = 1 to validate the new compare value. A comparison depending on value CLSi_ATOM_CHx_STAT.ACBI is started immediately.

***Note**: It is strongly recommended to update the bit fields CLSi_ATOM_CHx_CTRL_SOMB.ACB10 and CLSi_ATOM_CHx_CTRL_SOMB.ACB42 before writing to the registers CLSi_ATOM_CHx_CM0.CM0/ CLSi_ATOM_CHx_CM1.CM1 and CLSi_ATOM_CHx_SR0.SR0/CLSi_ATOM_CHx_SR1.SR1, as only the bits CLSi_ATOM_CHx_CTRL_SOMB.ACB10 and CLSi_ATOM_CHx_CTRL_SOMB.ACB42 can be written and these bit fields serve as shadow registers for the working register CLSi_ATOM_CHx_STAT.ACBI.*

***Note**: It is mandatory to always write both bit fields CLSi_ATOM_CHx_SR0.SR0 and CLSi_ATOM_CHx_SR1.SR1 regardless of the compare strategy.*

The compare strategy is controlled by the value stored in bit field CLSi_ATOM_CHx_STAT.ACBI. This bit field can only be updated with the value of bit fields CLSi_ATOM_CHx_CTRL_SOMB.ACB10 and CLSi_ATOM_CHx_CTRL_SOMB.ACB42.

The update of bit field CLSi_ATOM_CHx_STAT.ACBI can be triggered by a forced update or the normal update mechanism with CLSi_ATOM_AGC_GLB_CTRL.UPEN_CTRLk (k=0-7) **(k=x)**=10_B.

A force update updates the CLSi_ATOM_CHx_CM0.CM0 / CLSi_ATOM_CHx_CM1.CM1 register with the content of CLSi_ATOM_CHx_SR0.SR0 / CLSi_ATOM_CHx_SR1.SR1 register and the CLSi_ATOM_CHx_STAT.ACBI bit field with the content of bit field CLSi_ATOM_CHx_CTRL_SOMB.ACB10 and CLSi_ATOM_CHx_CTRL_SOMB.ACB42.

Writing to one of the CLSi_ATOM_CHx_SR0.SR0 / CLSi_ATOM_CHx_SR1.SR1 registers and triggering a forced update event can update CLSi_ATOM_CHx_CM0.CM0 / CLSi_ATOM_CHx_CM1.CM1 register with the value of CLSi_ATOM_CHx_SR0.SR0 / CLSi_ATOM_CHx_SR1.SR1 register and update CLSi_ATOM_CHx_STAT.ACBI bit field with the content of bits CLSi_ATOM_CHx_CTRL_SOMB.ACB10 and CLSi_ATOM_CHx_CTRL_SOMB.ACB42. Then a new comparison will be started.

When a previous comparison is finished (CLSi_ATOM_CHx_STAT.DV bit is reset), writing to one of the CLSi_ATOM_CHx_SR0.SR0/CLSi_ATOM_CHx_SR1.SR1 registers will update CLSi_ATOM_CHx_CM0.CM0, CLSi_ATOM_CHx_CM1.CM1 and CLSi_ATOM_CHx_STAT.ACBI while CLSi_ATOM_AGC_GLB_CTRL.UPEN_CTRLk (k=0-7) **(k=x)**= 10_B and starts comparison with these updated new values.

44.3.1.8.4 ATOM software reset of channels

Each channel x is assigned a unique bit x in the configuration register CLSi_ATOM_AGC_GLB_CTRL for software reset of the **ATOM[i]** channel x.

Writing the value 1_B to the bit field CLSi_ATOM_AGC_GLB_CTRL.RST_CHk (k=0-7) **(k=x)** via the configuration interface, will immediately reset the content of the following channel x registers to the initial hardware reset state.

- CLSi_ATOM_CHx_CTRL
- CLSi_ATOM_CHx_CTRL_SR
- CLSi_ATOM_CHx_CTRL2
- CLSi_ATOM_CHx_CN0.CN0
- CLSi_ATOM_CHx_CM0.CM0
- CLSi_ATOM_CHx_CM1.CM1
- CLSi_ATOM_CHx_SR0.SR0
- CLSi_ATOM_CHx_SR1.SR1
- CLSi_ATOM_CHx_STAT
- CLSi_ATOM_CHx_IRQ_NOTIFY
- CLSi_ATOM_CHx_IRQ_EN
- CLSi_ATOM_CHx_IRQ_MODE
- internal signals inside the **ATOM[i]** channel [x]:

Atomic reset of multiple channels is possible by writing a 1 on each associated bit. For example, writing $0F_H$ to register CLSi_ATOM_AGC_GLB_CTRL **(i=0)** will reset the ATOM0 channel x (x∈{0, 1, 2, 3}).

44.3.1.8.5 eGTM ATOM Configuration Registers Description

Sub-chapter ATOM Configuration Registers Description

Table 2236

Register Shortname	Description
CLSi_ATOM_AGC_GLB_CTRL	ATOM[i] AGC global control register (i=0-2)
CLSi_ATOM_AGC_ENDIS_CTRL	ATOM[i] AGC enable/disable control register (i=0-2)
CLSi_ATOM_AGC_ENDIS_STAT	ATOM[i] AGC enable/disable status register (i=0-2)
CLSi_ATOM_AGC_ACT_TB	ATOM[i] AGC action time base register (i=0-2)
CLSi_ATOM_AGC_OUTEN_CTRL	ATOM[i] AGC output enable control register (i=0-2)
CLSi_ATOM_AGC_OUTEN_STAT	ATOM[i] AGC output enable status register (i=0-2)
CLSi_ATOM_AGC_FUPD_CTRL	ATOM[i] AGC force update control register (i=0-2)
CLSi_ATOM_AGC_INT_TRIG	ATOM[i] AGC internal trigger control register (i=0-2)
CLSi_ATOM_CHx_CTRL	ATOM[i] channel [x] control register (x=0-7;i=0-2) (i=0-2;x=0-7)
CLSi_ATOM_CHx_CTRL_SOMI	ATOM[i] channel [x] control register (x=0-7;i=0-2) (i=0-2;x=0-7)
CLSi_ATOM_CHx_CTRL_SOMC	ATOM[i] channel [x] control register (x=0-7;i=0-2) (i=0-2;x=0-7)
CLSi_ATOM_CHx_CTRL_SOMP	ATOM[i] channel [x] control register (x=0-7;i=0-2) (i=0-2;x=0-7)
CLSi_ATOM_CHx_CTRL_SOMS	ATOM[i] channel [x] control register (x=0-7;i=0-2) (i=0-2;x=0-7)
CLSi_ATOM_CHx_CTRL_SOMB	ATOM[i] channel [x] control register (x=0-7;i=0-2) (i=0-2;x=0-7)
CLSi_ATOM_CHx_CTRL_SR	ATOM[i] channel [x] control shadow register (x=0-7;i=0-2)
CLSi_ATOM_CHx_CTRL2	ATOM[i] channel [x] control2 register (x=0-7;i=0-2)
CLSi_ATOM_CHx_STAT	ATOM[i] channel [x] status register (x=0-7;i=0-2)
CLSi_ATOM_CHx_CN0	ATOM[i] channel [x] CCU0 counter register (x=0-7;i=0-2)
CLSi_ATOM_CHx_CM0	ATOM[i] channel [x] CCU0 compare register (x=0-7;i=0-2)
CLSi_ATOM_CHx_SR0	ATOM[i] channel [x] CCU0 compare shadow register (x=0-7;i=0-2)
CLSi_ATOM_CHx_CM1	ATOM[i] channel [x] CCU1 compare register (x=0-7;i=0-2)
CLSi_ATOM_CHx_SR1	ATOM[i] channel [x] CCU1 compare shadow register (x=0-7;i=0-2)

(table continues...)

Table 2236 (continued)

Register Shortname	Description
CLSi_ATOM_CHx_IRQ_NOTIFY	ATOM[i] channel [x] interrupt notification register (x=0-7;i=0-2)
CLSi_ATOM_CHx_IRQ_EN	ATOM[i] channel [x] interrupt enable register (x=0-7;i=0-2)
CLSi_ATOM_CHx_IRQ_FORCINT	ATOM[i] channel [x] software interrupt generation (x=0-7;i=0-2)
CLSi_ATOM_CHx_IRQ_MODE	ATOM[i] channel [x] interrupt mode configuration register (x=0-7;i=0-2)

44.3.1.8.6 ATOM signal description

Table 2237 ATOM reset

Name	Description
ATOM[j]_RESET_CH[x]	**ATOM** instance [j] channel [x] reset active signal

Table 2238 ATOM Interrupt Signals

Name	Description
ATOM_CCU0TC[x]_IRQ	CCU0 Trigger condition interrupt for channel x. Internal interrupt signal when compare event occurs in CCU0 unit.
ATOM_CCU1TC[x]_IRQ	CCU1 Trigger condition interrupt for channel x. Internal interrupt signal when compare event occurs in CCU0 unit.

Table 2239 ATOM Signal Group

Name	Description
CAP	signaling a compare match event for capture
ATOM_ENDIS	A bundle of channel enable signals controlled by AGC unit
ATOM_CTRL_TRIG	trigger signal to update the AGC registers ATOM[i]_agc_OUTEN_STAT, CLSi_ATOM_AGC_ENDIS_STAT and to activate ATOM_FUPD signals
EXT_TRIGIN_S	synchronized EXT_TRIGIN signal to selected **CMU** clock resolution
ATOM_FUPD	A bundle of channel force update event signals controlled by AGC unit
ATOM_RSTCN0_CH	A bundle of channel force update event signals controlled AGC unit
ATOM_OUTEN	output enable signal from AGC by unit
ATOM_TRIG_CCU0	trigger signal for CCU0 compare match
ATOM_TRIG_CCU1	trigger signal for CCU1 compare match

(table continues...)

Table 2239 (continued) ATOM Signal Group

Name	Description
ATOM_UPEN	A bundle of channel update enable signals controlled by AGC unit
SEL_CMU_CLKEN	The selected **CMU** clock enable signal
ATOM_EXT_HRES	Not internal signal from **eGTM** or **ATOM**. A signal to explain how the external circuit can generate a signal for each channel x that consists of edges from ATOM_OUT *[x:x]* with delay of ATOM_OUT_HRES[x].
ATOM_CH_TRIGIN	Trigger input signal for each channel x. It comes from the preceding channel x-1 or channel 7 of the preceding instance.
ATOM_CH_TRIGOUT	Trigger output signal for each channel x. It is connected to the trigger input signal of the next channel x+1 or channel 0 of the succeeding instance.
ATOM_CH_HRES_IN[x]	HRES trigger input signal for each channel x. It is 5 bits value. Together with ATOM_CH_TRIGIN *[x:x]*, it comes from the preceding channel x-1 or channel 7 of the preceding instance.
ATOM_CH_HRES_OUT[x]	HRES trigout signal for each channel x. It is 5 bits value. Together with ATOM_CH_TRIGOUT *[x:x]*, it is connected with the HRES trigger input signal of the next channel x+1 or channel 0 of the next instance.
ATOM[i]_CH[x]_HRES_PERIOD	HRES value of **ATOM** i channel x that is calculated with regards to CLSi_ATOM_CHx_CM0**.**CM0.
ATOM[i]_CH[x]_HRES_PULSEWIDTH	HRES value of **ATOM** i channel x that is calculated with regards to CLSi_ATOM_CHx_CM1**.**CM1.

44.3.1.8.7 ATOM port description

Table 2240 ATOM signal interface

Name	Description
EXT_TRIGIN	EXT_TRIGIN[7:0] of all channels of **ATOM** instance i are coming from TIM_EXT_CAPTURE *[7:0]* signal from **TIM** instance i
ATOM_TRIGIN	**ATOM** trigger chain input of **ATOM** instance
ATOM_TRIGOUT	**ATOM** trigger chain output of **ATOM** instance
ATOM_TRIGOUT_DEL	**ATOM** trigger chain output delayed by 1 cluster clock; will be used only in **ATOM** [0] instance as ATOM_TRIGIN
ATOM_OUT	Output signal ATOM_OUT
ATOM_OUT_T	Output signal ATOM_OUT_T

Table 2241 ATOM interrupt interface

Name	Description
ATOM_IRQ	**ATOM IRQ** line signals
ATOM_IRQ_CLR	**ATOM IRQ** clear signal, hardware clear of CLSi_ATOM_CHx_IRQ_NOTIFY register
ATOM_IRQ_OCC	**ATOM IRQ** occurred signal, corresponds to CLSi_ATOM_CHx_IRQ_NOTIFY register

Table 2242 ATOM signal HRES interface

Name	Description
ATOM_OUT_HRES[x]	ATOM_OUT high-resolution output signal for channel x
ATOM_OUT_T_HRES[x]	ATOM_OUT_T high-resolution output signal channel x
ATOM_HRES_IN	**ATOM** high-resolution trigger chain input; will be driven by previous **ATOM** instance i-1
ATOM_HRES_OUT	**ATOM** high-resolution trigger chain output; will be used by following **ATOM** instance i+1
ATOM_HRES_OUT_DEL	**ATOM** high-resolution trigger chain output delayed by 1 cluster clock; will be used only in **ATOM** instance 0 as ATOM_HRES_IN

44.3.1.9 Dead-time module (DTM)

44.3.1.9.1 Overview

The following figure gives an overview of the structure of the Dead-time Module (**DTM**).

Register name indices used within this module:

- i:={0, 1,..., NCCM-1} instance index of cluster
- j:={0, 1,..., NCCM-1} instance index in case this index refers to a cluster/module with j!=i
- d:={0, 1,..., 5} index for a dedicated **DTM** ((d∈{0, 1, 2, 3}): **DTM** after **TOM**; (d∈{4, 5}): **DTM** after **ATOM**
- x:={0, 1,..., 3} channel index in a **DTM** instance
- y:={0, 1,..., N-1} with N:={NTOM, NATOM} channel index of another module e.g. **TOM**, **ATOM**, **TIM**
- k:={0, 1} channel index with a special meaning e.g. limited value subset

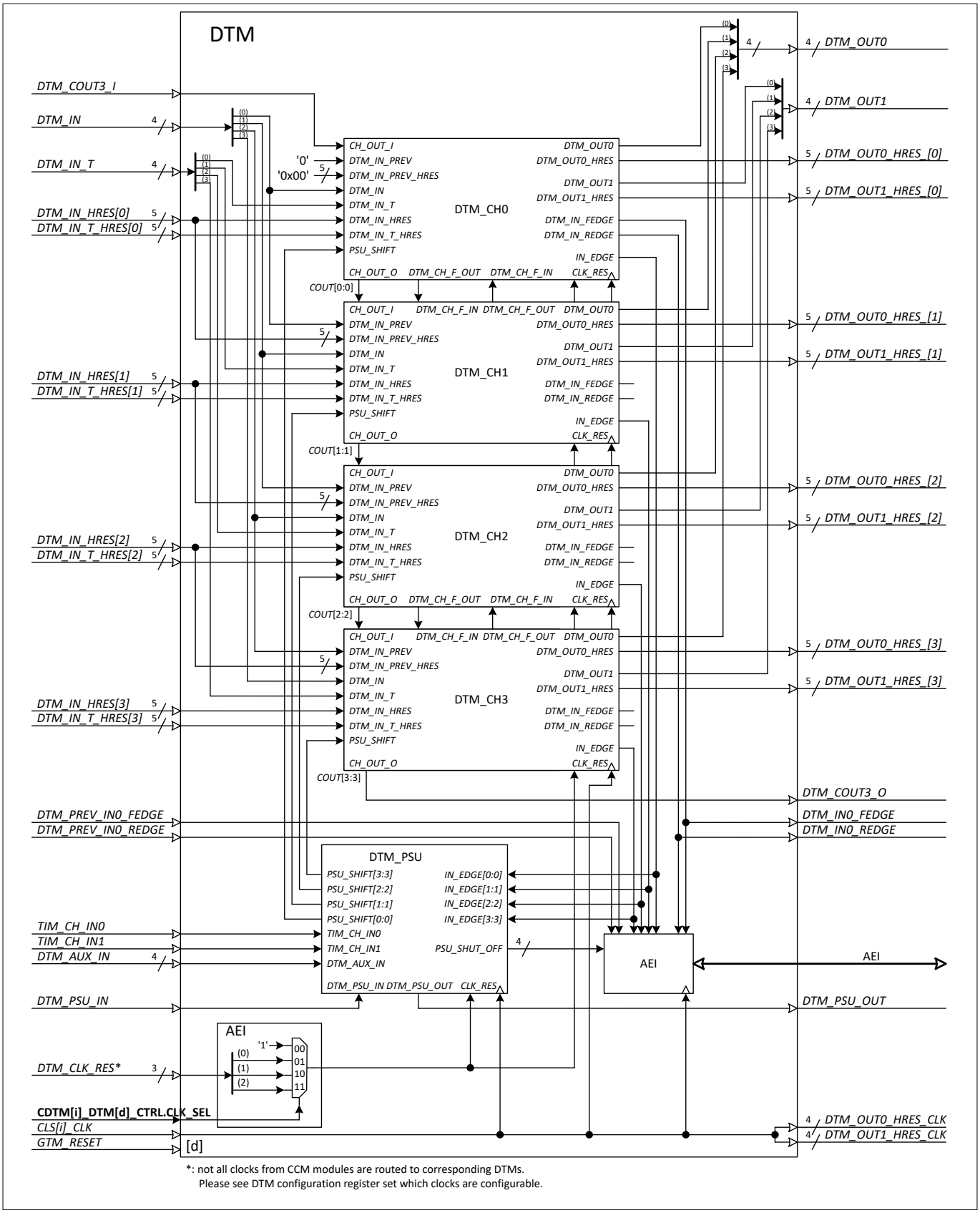


Figure 1100 DTM overview

***Note**: The figure is a general picture from the IP vendor. The exact configuration can be found in the device specific configuration.*

The **DTM** module can enhance the functions of modules **TOM**, **ATOM** by modifying their output signals.

In general, the **DTM** instances are placed behind **TOM**, **ATOM**, i.e., the outputs TOM_OUT[y:y] and TOM_OUT_T[y:y], ATOM_OUT[y:y] and ATOM_OUT_T[y:y] are each routed to the **DTM** instance inputs DTM_IN and DTM_IN_T.

Four **DTM** instances behind a **TOM** instance i, two **DTM** instances behind an **ATOM** instance i are grouped together in a Cluster **DTM** hierarchy called CDTM[i].

The connections between **DTM** and the modules **TOM**, **ATOM** are depicted in Figure 1101.

Attention**: **The initial configuration of a DTM channel or any asynchronous channel configuration to the running application is only allowed, if the corresponding TOM, ATOM channel is disabled. A configuration of DTM channel realized from the shadow registers synchronous to the running application can be executed without disabling the interfaced TOM, ATOM channels.

The main function of the **DTM** is to derive the individual inverse signal (DTM_OUT1[3:0]) for each input DTM_IN[3:0] and to apply an edge-specific delay between the edge of the original signal and the derived inverted signal (i.e., the dead-time). This function is mainly used for controlling of half bridges.

A second function provided by **DTM** is to set the outputs of one channel to the value of the preceding channel if requested by a trigger on input TIM_CH_IN0, TIM_CH_IN1 or DTM_AUX_IN[0:0]. This feature allows a phase shift on one PWM signal to the phase of the preceding PWM signal up to the next edge on this channel.

The third function provided by **DTM** is to (N)**AND**/(N)**OR**/X(N)**OR** combine the input DTM_IN[x:x] signal of one **DTM** channel with the signal on inputs TIM_CH_IN0, TIM_CH_IN1 or DTM_AUX_IN[0:0] (selected inside PSU and assigned to one of the signals PSU_SHIFT[x:x]) or with the combinational output (signal COUT[x-1:x-1]) of the preceding channel.

As a result, COUT[1:1] will be the combined signal of DTM_IN[0:0] and TIM_CH_IN0, TIM_CH_IN1 or DTM_AUX_IN[0:0] and the signal DTM_IN[1:1]. For COUT[2:2], this chain can be combined again with signal DTM_IN[2:2].

For COUT[3:3], this chain can be combined again with signal DTM_IN[3:3].

- The outputs of each channel may be swapped individually to provide the function of combining signals on each output of a channel.
- For channel 0 the function is controlled by CLSi_CDTM_DTMd_CH_CTRL1.SWAP_0.
- For channel 1 the function is controlled by CLSi_CDTM_DTMd_CH_CTRL1.SWAP_1.
- For channel 2 the function is controlled by CLSi_CDTM_DTMd_CH_CTRL1.SWAP_2.
- For channel 3 the function is controlled by CLSi_CDTM_DTMd_CH_CTRL1.SWAP_3.

The 4th function is the high-resolution PWM support. For each of the two input signals of a **DTM** channel, a high-resolution value is given by a corresponding input signal. Here, a configurable value is added to the high-resolution input value.

The high-resolution mode in **DTM** is only supported for standard dead-time mode and for phase shift mode, where standard dead-time is used.

Depending on the device configuration, not every **DTM** instance is available. E.g. a device may only have one **DTM** connected to the first four channels of **ATOM**. In this case, the other four channels (channel 4 to channel 7) are connected over a buffer stage to **eGTM** outputs.

For detailed information, which **DTM** instance is available, refer to the corresponding device-specific documentations.

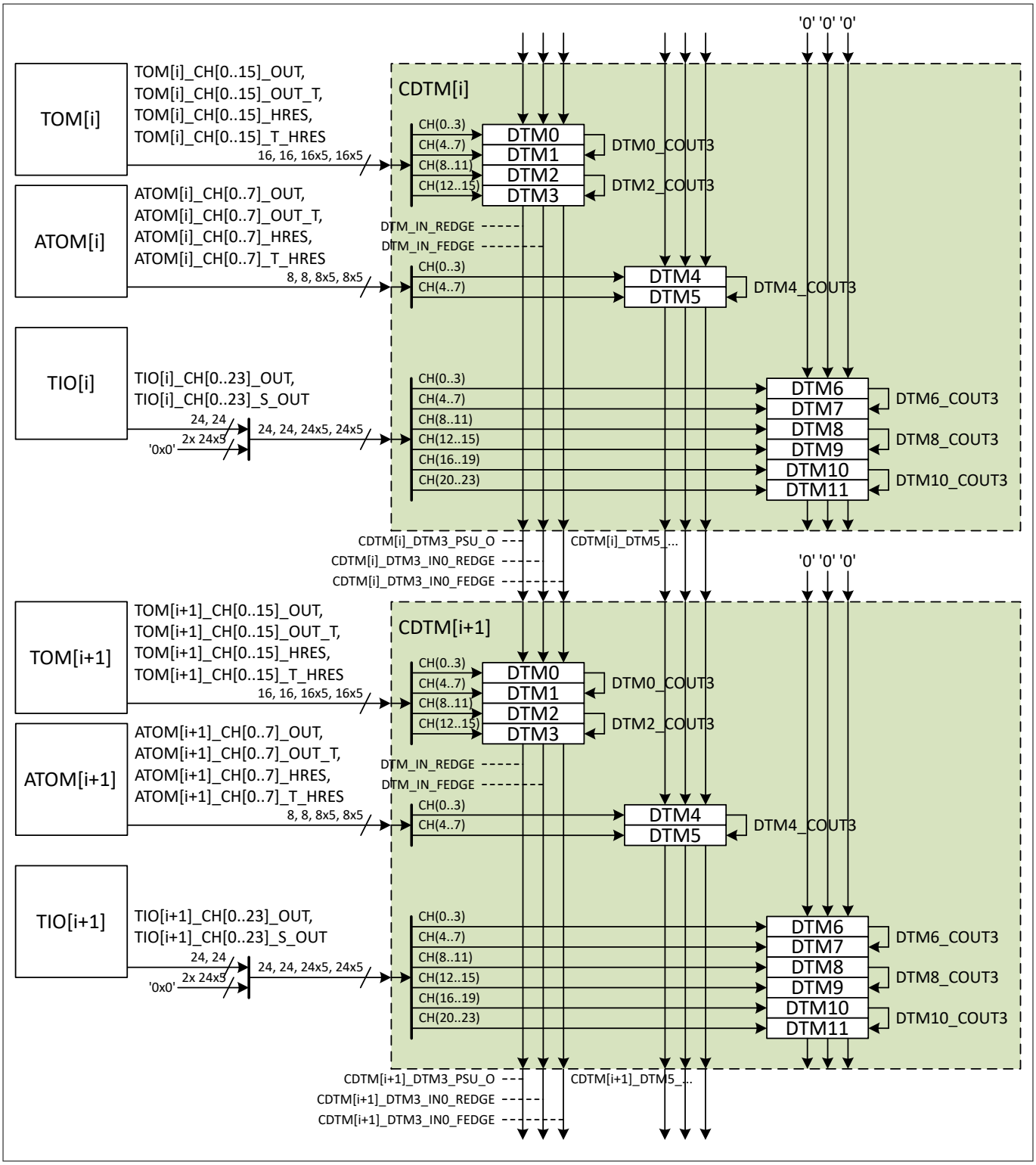


Figure 1101 Connections of TOM, ATOM to DTM Inputs DTM_IN *[x:x]*/DTM_IN_T *[x:x]*

***Note**: The figure is a general picture from the IP vendor. The exact configuration can be found in the device specific configuration.*

Additionally, the **DTM** instances have inputs TIM_CH_IN0, TIM_CH_IN1 which are driven by **TIM** output signals F_OUT of instances i and channel y.

Two configurations of **TIM** to **DTM** connections are possible depending on the **DTM** channel-specific configuration bit CLSi_CDTM_DTMd_PS_CTRL.TIM_SEL.

In case of CLSi_CDTM_DTMd_PS_CTRL.TIM_SEL=0, the connected **TIM** input may not be of the same cluster as the **DTM**.

In case of CLSi_CDTM_DTMd_PS_CTRL.TIM_SEL=1, the **TIM** input is of the same cluster as the **DTM**.

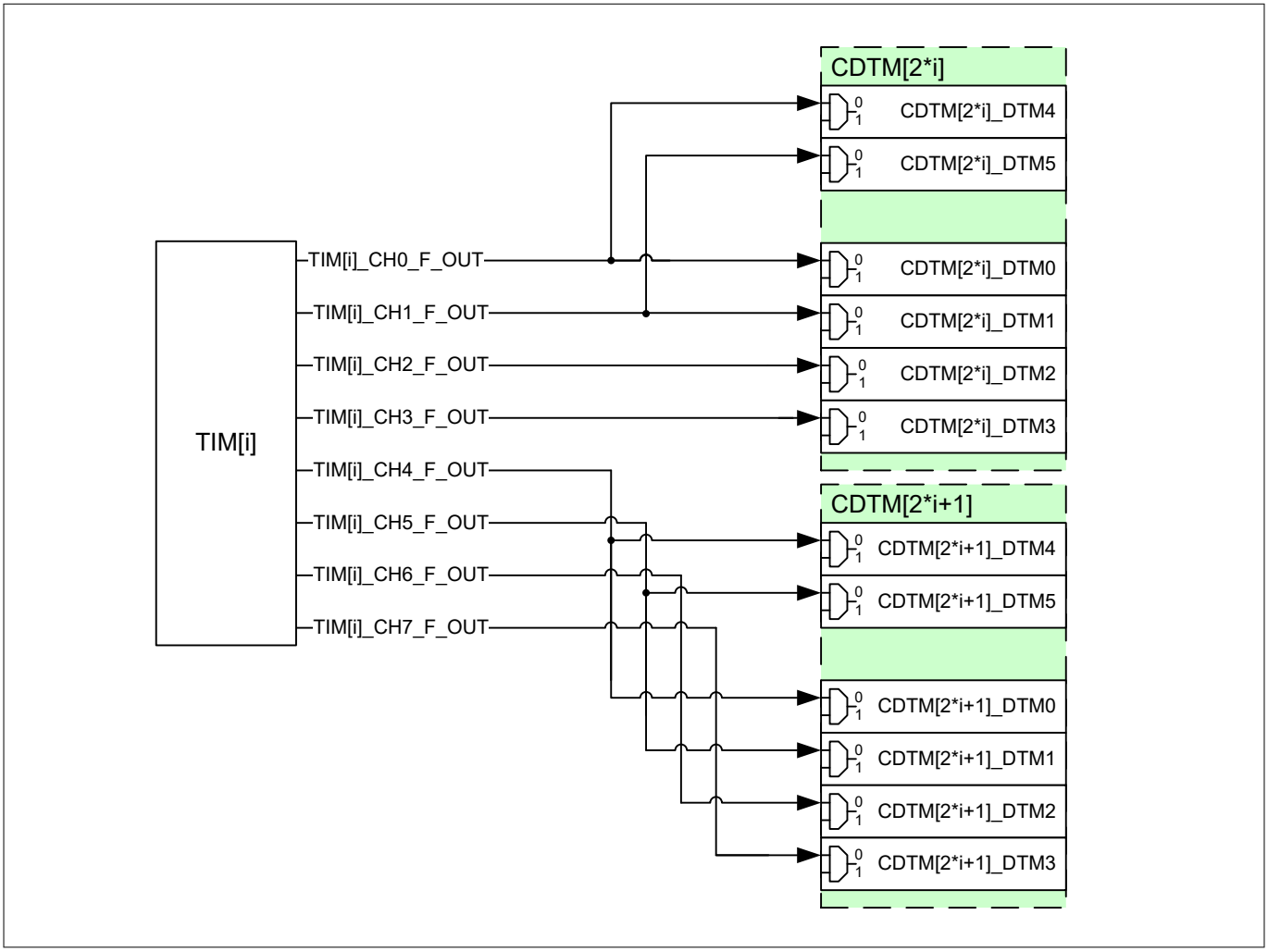


Figure 1102 Connections of TIM to DTM inputs TIM_CH_IN0/TIM_CH_IN1 for CLSi_CDTM_DTMd_PS_CTRL.TIM_SEL=0

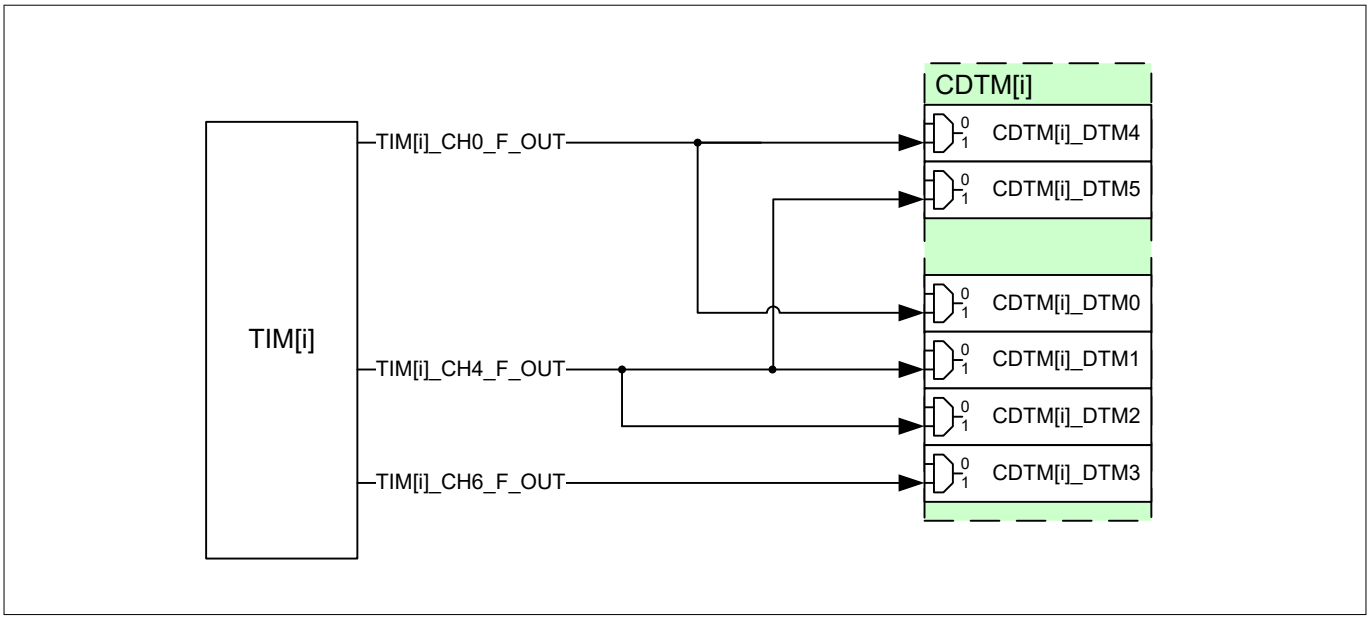


Figure 1103 Connections of TIM to DTM inputs TIM_CH_IN0/TIM_CH_IN1 for CLSi_CDTM_DTMd_PS_CTRL.TIM_SEL=1

***Note**: The multiplexer shown in Figure 1102 and Figure 1103 and driven by CLSi_CDTM_DTMd_PS_CTRL.TIM_SEL is located in the block DTM_PSU.*

There are also connections between **DTM** instances inside one cluster.

For each pair of **DTM** instance 2d and instance 2d+1 the combinatorial output COUT[3:3] (COUT[3:3] routed to DTM_COUT3_O) of **DTM** instance 2d channel 3 is connected to **DTM** instance 2d+1 channel 0 at signal DTM_COUT3_I .

With this, a combinatorial chain over two neighbored **DTM** instances can be configured.

If one of the two neighbored **DTM** instances is not available (i.e. it is an empty instance) the inputs used for connections between two neighbored **DTM** instances are hard wired to 0.

These connections are shown in Figure 1100 and Figure 1101.

- For channel 0 of **DTM** instance 2d input signal COUT[x-1:x-1] is unused and CLSi_CDTM_DTMd_CH_CTRL1.I1SEL_0 is defined as 0.
- For channel 0 of **DTM** instance 2d input signal COUT[x-1:x-1] is unused and CLSi_CDTM_DTMd_CH_CTRL1.I1SEL_1 is defined as 0.
- For channel 0 of **DTM** instance 2d input signal COUT[x-1:x-1] is unused and CLSi_CDTM_DTMd_CH_CTRL1.I1SEL_2 is defined as 0.
- For channel 0 of **DTM** instance 2d input signal COUT[x-1:x-1] is unused and CLSi_CDTM_DTMd_CH_CTRL1.I1SEL_3 is defined as 0.

An additional link between **DTM** instances behind an **ATOM** is a forwarding of output DTM_PSU_OUT of **DTM** instance d to the input DTM_PSU_IN of next available instance d+1 behind an **ATOM**.

The same link is available between all available **DTM** instances behind a **TOM**.

For unavailable **DTM[d]** instances (i.e. the instance **DTM[d]** is called empty) the signal DTM_PSU_OUT of **DTM** instance d-1 is passed through empty instance **DTM[d]** to DTM_PSU_IN of **DTM** instance d+1, DTM_IN0_FEDGE and DTM_IN0_REDGE of **DTM** instance d-1 are passed through **DTM[d]** to DTM_PREV_IN0_FEDGE and DTM_PREV_IN0_REDGE of **DTM** instance d+1.

44.3.1.9.2 DTM channel

The following figure depicts the circuit of a **DTM** channel.

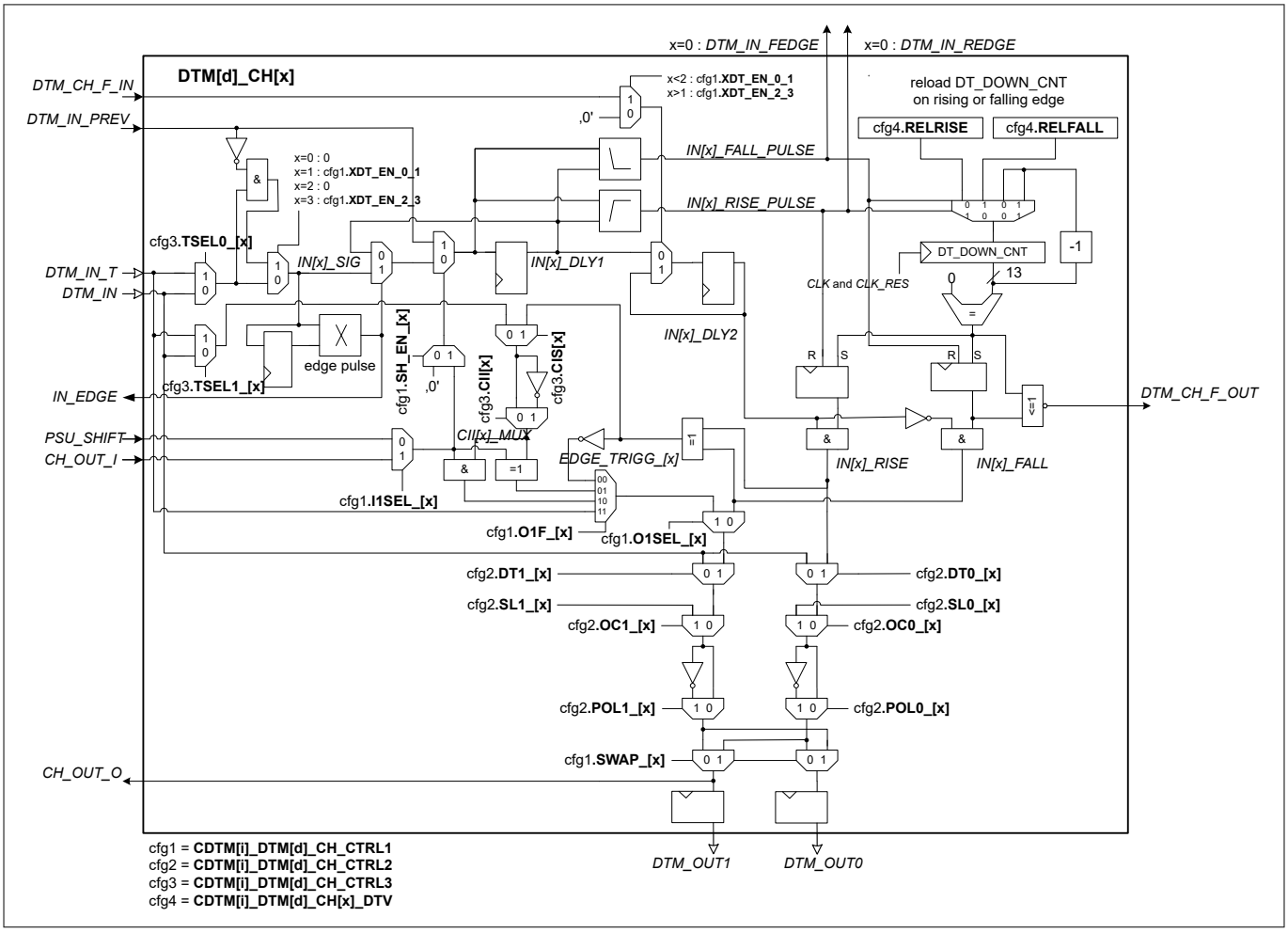


Figure 1104 DTM channel overview

The main feature of each channel is to derive the inverse signal out of the input signal DTM_IN[x:x], apply an edge dependent delay to the two resulting signal paths and provide these signals at the outputs DTM_OUT0[x:x] and DTM_OUT1[x:x].

There are two possibilities to apply dead-time on **eGTM** output signals.

One is to use one **DTM** channel per **TOM**/**ATOM** channel and generate inside the **DTM** the second inverse signal. This is called the standard dead-time generation.

The second way is to generate two signals out of two **TOM**/**ATOM** channel and to apply inside the **DTM** only the dead-time by using two cross-linked **DTM** channels. This is called the cross dead-time generation.

Standard dead-time generation

The dead-time can be configured for each edge individually. The bit field CLSi_CDTM_DTMd_CH_DTVx.RELRISE contains the reload value for the counter and defines the delay for rising edges in multiples of selected clock ticks.

The bit field CLSi_CDTM_DTMd_CH_DTVx.RELFALL contains the reload value for the counter and defines the delay for falling edges in multiples of selected clock ticks.

***Note**: If the selected clock ticks are modified by writing CLSi_CDTM_DTMd_CTRL.CLK_SEL during a running application, the configured times (e.g. DT_DOWN_CNT) may differ, because the counters can run once partly with the old and the new clock ticks. It is recommended not to reprogram CLSi_CDTM_DTMd_CTRL.CLK_SEL during operation.*

The counter is reloaded with the value of CLSi_CDTM_DTMd_CH_DTVx.RELRISE on a rising edge and reloaded with the value of CLSi_CDTM_DTMd_CH_DTVx.RELFALL on a falling edge on input DTM_IN[x:x] (or DTM_IN_PREV in case of shift enable CLSi_CDTM_DTMd_CH_CTRL1.SH_EN_1, CLSi_CDTM_DTMd_CH_CTRL1.SH_EN_2, CLSi_CDTM_DTMd_CH_CTRL1.SH_EN_3).(x=1)

The signal flow for function of standard dead-time signal generation is depicted in Figure 1105.

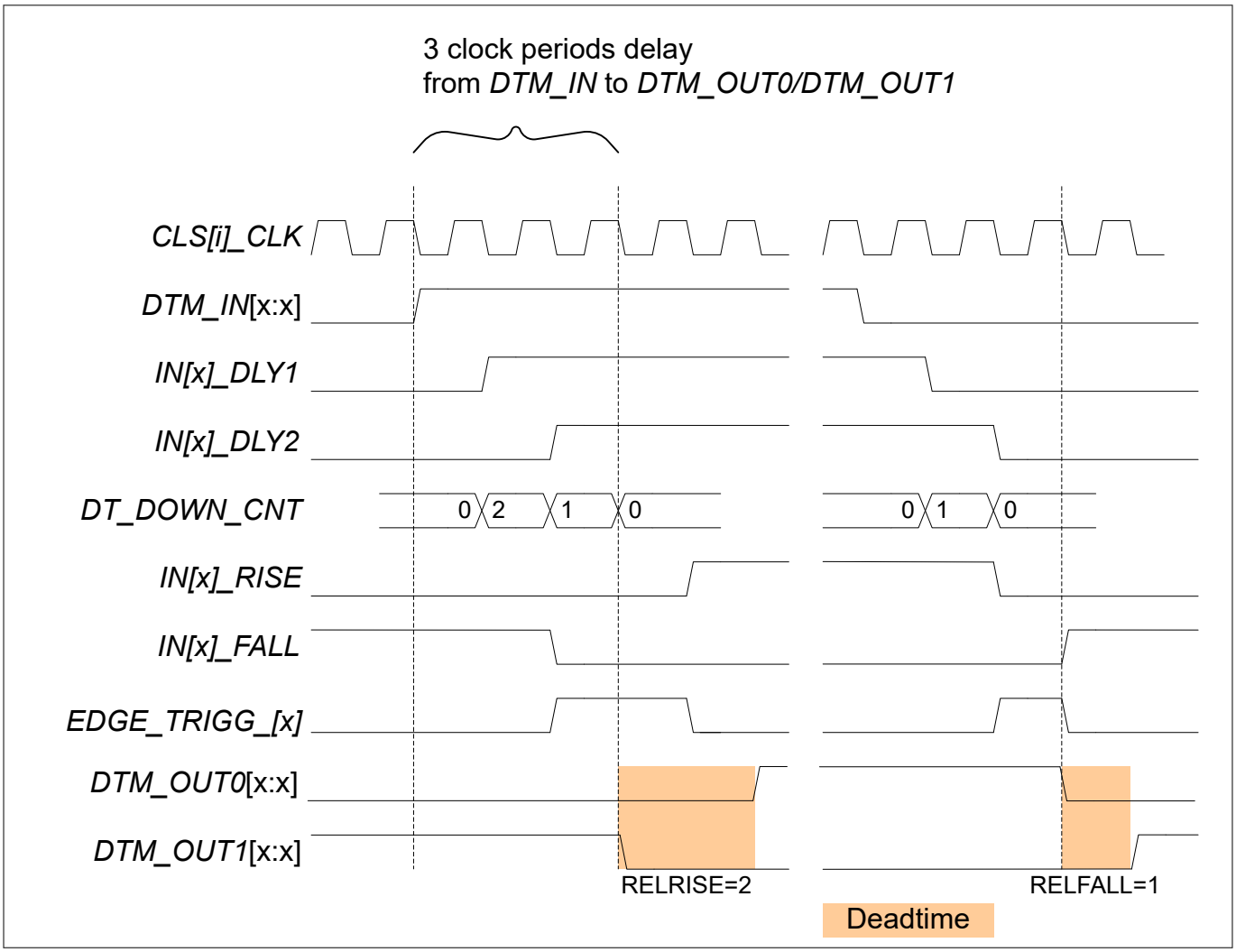


Figure 1105 Wave signals for function of dead-time generation

***Note**: The figure above shows the timing for CLSi_CDTM_DTMd_CTRL.CLK_SEL=00_B. If another clock resolution is configured, be aware that only the counter DT_DOWN_CNT runs with the reduced clock resolution.*

- The delay from the input signal DTM_IN[0:0] to the output signals DTM_OUT0[0:0] and DTM_OUT1[0:0] is three system clock periods, if CLSi_CDTM_DTMd_CH_CTRL2.DT0_0=1 and CLSi_CDTM_DTMd_CH_CTRL2.DT1_0=1.
- The delay from the input signal DTM_IN[1:1] to the output signals DTM_OUT0[1:1] and DTM_OUT1[1:1] is three system clock periods, if CLSi_CDTM_DTMd_CH_CTRL2.DT0_1=1 and CLSi_CDTM_DTMd_CH_CTRL2.DT1_1=1.

- The delay from the input signal DTM_IN[2:2] to the output signals DTM_OUT0[2:2] and DTM_OUT1[2:2] is three system clock periods, if CLSi_CDTM_DTMd_CH_CTRL2.DT0_2=1 and CLSi_CDTM_DTMd_CH_CTRL2.DT1_2=1.
- The delay from the input signal DTM_IN[3:3] to the output signals DTM_OUT0[3:3] and DTM_OUT1[3:3] is three system clock periods, if CLSi_CDTM_DTMd_CH_CTRL2.DT0_3=1 and CLSi_CDTM_DTMd_CH_CTRL2.DT1_3=1.
- The delay from the input signal DTM_IN[0:0] to the output signals DTM_OUT0[0:0] and DTM_OUT1[0:0] is one system clock period if CLSi_CDTM_DTMd_CH_CTRL2.DT0_0=0 and CLSi_CDTM_DTMd_CH_CTRL2.DT1_0=0.
- The delay from the input signal DTM_IN[1:1] to the output signals DTM_OUT0[1:1] and DTM_OUT1[1:1] is one system clock period if CLSi_CDTM_DTMd_CH_CTRL2.DT0_1=0 and CLSi_CDTM_DTMd_CH_CTRL2.DT1_1=0.
- The delay from the input signal DTM_IN[2:2] to the output signals DTM_OUT0[2:2] and DTM_OUT1[2:2] is one system clock period if CLSi_CDTM_DTMd_CH_CTRL2.DT0_2=0 and CLSi_CDTM_DTMd_CH_CTRL2.DT1_2=0.
- The delay from the input signal DTM_IN[3:3] to the output signals DTM_OUT0[3:3] and DTM_OUT1[3:3] is one system clock period if CLSi_CDTM_DTMd_CH_CTRL2.DT0_3=0 and CLSi_CDTM_DTMd_CH_CTRL2.DT1_3=0.
- The delay from the input signal DTM_IN_T[0:0] to the output signals DTM_OUT0[0:0] and DTM_OUT1[0:0] is three system clock periods, in case CLSi_CDTM_DTMd_CH_CTRL1.O1SEL_0=0.
- The delay from the input signal DTM_IN_T[1:1] to the output signals DTM_OUT0[1:1] and DTM_OUT1[1:1] is three system clock periods, in case CLSi_CDTM_DTMd_CH_CTRL1.O1SEL_1=0.
- The delay from the input signal DTM_IN_T[2:2] to the output signals DTM_OUT0[2:2] and DTM_OUT1[2:2] is three system clock periods, in case CLSi_CDTM_DTMd_CH_CTRL1.O1SEL_2=0.
- The delay from the input signal DTM_IN_T[3:3] to the output signals DTM_OUT0[3:3] and DTM_OUT1[3:3] is three system clock periods, in case CLSi_CDTM_DTMd_CH_CTRL1.O1SEL_3=0.
- The delay from the input signal DTM_IN_T[0:0] to the output signals DTM_OUT0[0:0] and DTM_OUT1[0:0] is one system clock period in case of CLSi_CDTM_DTMd_CH_CTRL1.O1F_0=11_B and CLSi_CDTM_DTMd_CH_CTRL1.O1SEL_0=1.
- The delay from the input signal DTM_IN_T[1:1] to the output signals DTM_OUT0[1:1] and DTM_OUT1[1:1] is one system clock period in case of CLSi_CDTM_DTMd_CH_CTRL1.O1F_1=11_B and CLSi_CDTM_DTMd_CH_CTRL1.O1SEL_1=1.
- The delay from the input signal DTM_IN_T[2:2] to the output signals DTM_OUT0[2:2] and DTM_OUT1[2:2] is one system clock period in case of CLSi_CDTM_DTMd_CH_CTRL1.O1F_2=11_B and CLSi_CDTM_DTMd_CH_CTRL1.O1SEL_2=1.
- The delay from the input signal DTM_IN_T[3:3] to the output signals DTM_OUT0[3:3] and DTM_OUT1[3:3] is one system clock period in case of CLSi_CDTM_DTMd_CH_CTRL1.O1F_3=11_B and CLSi_CDTM_DTMd_CH_CTRL1.O1SEL_3=1.

For CLSi_CDTM_DTMd_CH_CTRL1.O1SEL_0=1 and CLSi_CDTM_DTMd_CH_CTRL1.O1F_0!=11_B there are further configurations possible which result in different paths from the selected input signal to the output signals causing different delays of 1 or 3 clock periods. For details see Figure 1104.

The reset level of the output signals DTM_OUT0[x:x] connected from **ATOM** module depends on the hardware configuration value ATOM_OUT_RST chosen by the silicon vendor.

The reset level of the output signals DTM_OUT1[x:x] connected from **ATOM** module is defined by the inverse hardware configuration value ATOM_OUT_RST chosen by the silicon vendor.

The reset level of the output signals DTM_OUT0[x:x] connected from **TOM** module is defined by the hardware configuration value TOM_OUT_RST chosen by the silicon vendor.

The reset level of the output signals DTM_OUT1[x:x] connected from **TOM** module is defined by the inverse hardware configuration value TOM_OUT_RST chosen by the silicon vendor.

Cross channel dead-time

A second way to apply a dead-time value on two output signals is the cross channel dead-time.

In opposite to the dead-time described in Standard dead-time generation the cross channel dead-time mode generates a dead-time on the input signals of two neighbored **DTM** channels.

To do this, two neighbored **DTM** input signals (on **DTM** channel (2k) and (2k+1) for (k∈{0, 1})) are cross-linked together in the way that a falling edge on one channel leads to a hold phase of current signal value on the cross-linked channel.

This behavior is achieved by the following:

A falling edge on input signal of **DTM** channel 2k reloads the counter DT_DOWN_CNT with the value of CLSi_CDTM_DTMd_CH_DTVx**.**RELFALL. While this counter is counting down, the output signal of the cross-linked channel (2k+1) keeps its value. If the counter DT_DOWN_CNT has reached 0 again (DT_DOWN_CNT=0), the channel (2k+1) output is released and can follow the value on its input.

The timing of the cross channel dead-time is depicted in Figure 1106.

The following Figure 1106 shows the behavior in case of input edges at DTM_IN[2k:2k] and DTM_IN[2k+1:2k+1] occur at the same point in time.

Then the falling edge is forwarded immediately (with only three clock-cycles delay) and the rising edge is delayed additionally by the number of clock ticks specified by the CLSi_CDTM_DTMd_CH_DTVx**.**RELFALL parameter of the cross-linked channel.

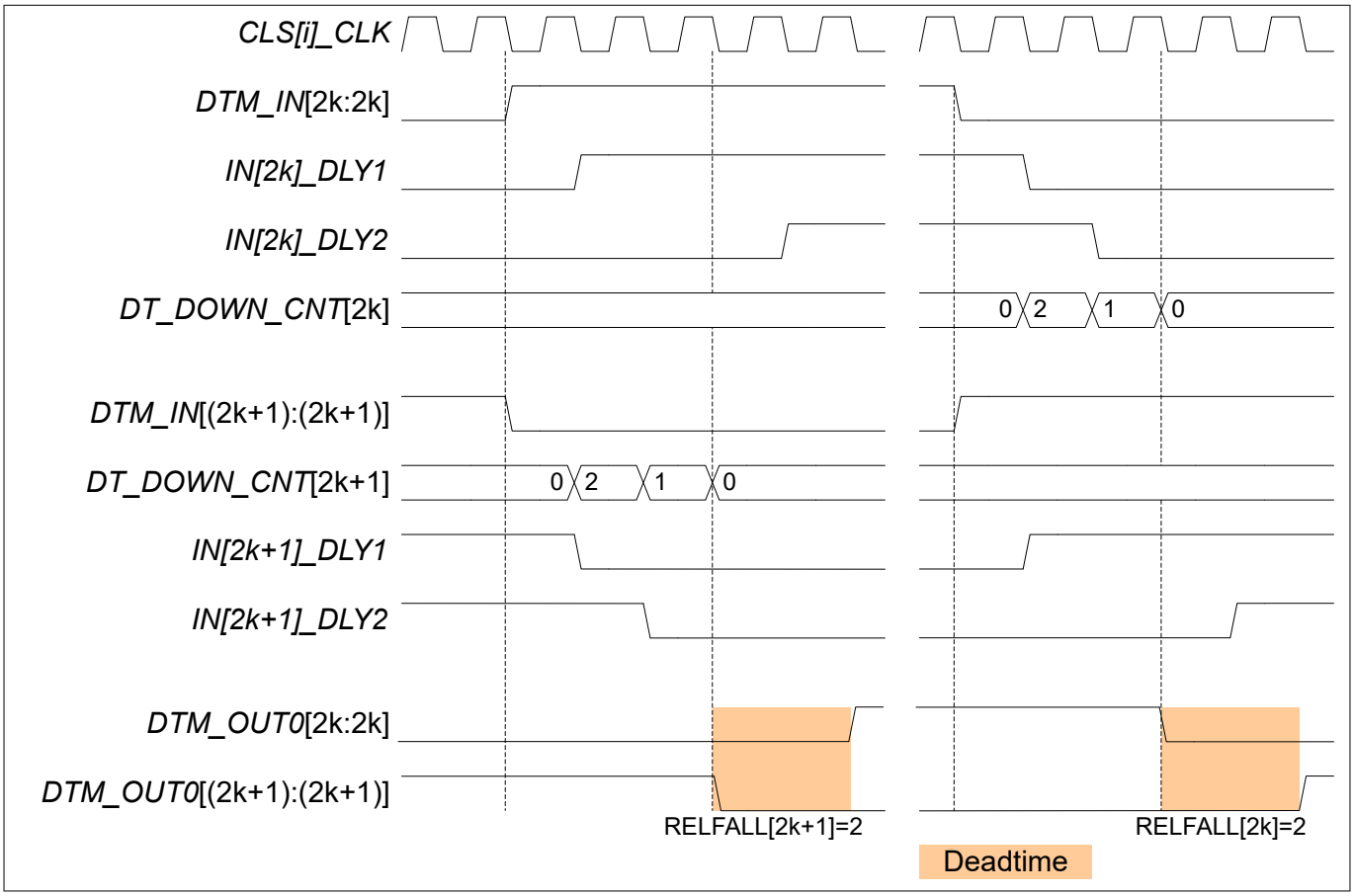


Figure 1106 Cross channel dead-time timing diagram

***Note**: The figure above shows the timing for CLSi_CDTM_DTMd_CTRL.CLK_SEL=00_B. If another clock resolution is configured, be aware that only the counters DT_DOWN_CNT run with the reduced clock resolution.*

In case of level high (i.e. 1) at the **DTM** inputs DTM_IN[2k:2k] and DTM_IN[(2k+1):(2k+1)] at the same point in time, the channel of (2k) has higher priority than the corresponding channel (2k+1). This means that in this case the input DTM_IN[(2k+1):(2k+1)] is forced to level low (i.e. 0) immediately at the channel input.

As a result the **DTM** output of channel DTM_OUT0[(2k+1):(2k+1)] can never be high if the cross-linked channel DTM_OUT0[2k:2k] is high.

Dead-time shadow register

The values CLSi_CDTM_DTMd_CH_DTVx.RELRISE and CLSi_CDTM_DTMd_CH_DTVx.RELFALL define the dead-time as described in the prior chapters. To be able to change these values synchronized to a generated waveform an additional shadow register CLSi_CDTM_DTMd_CH_DTV_SRx is implemented. The register is accessible over the configuration interface.

The register CLSi_CDTM_DTMd_CH_DTV_SRx includes the bit fields CLSi_CDTM_DTMd_CH_DTV_SRx.RELRISE_SR, CLSi_CDTM_DTMd_CH_DTV_SRx.RELFALL_SR which are the shadow bit fields of CLSi_CDTM_DTMd_CH_DTVx.RELRISE and CLSi_CDTM_DTMd_CH_DTVx.RELFALL and further four bits to control the update mechanism.

CLSi_CDTM_DTMd_CH_DTV_SRx.RELRISE_SR is the shadow value for CLSi_CDTM_DTMd_CH_DTVx.RELRISE. CLSi_CDTM_DTMd_CH_DTV_SRx.RELFALL_SR is the shadow value for CLSi_CDTM_DTMd_CH_DTVx.RELFALL.

The update mechanism of CLSi_CDTM_DTMd_CH_DTVx.RELRISE is enabled if CLSi_CDTM_DTMd_CH_DTV_SRx.RELRISE_UPD_EN=1 is set.

The update mechanism of CLSi_CDTM_DTMd_CH_DTVx.RELFALL is enabled if CLSi_CDTM_DTMd_CH_DTV_SRx.RELFALL_UPD_EN=1 is set.

Bit CLSi_CDTM_DTMd_CH_DTV_SRx.RELRISE_UPD_FE0RE1 controls if the falling edge (=0) or the rising edge (=1) of DTM_IN[x:x] triggers the update of CLSi_CDTM_DTMd_CH_DTVx.RELRISE.

Bit CLSi_CDTM_DTMd_CH_DTV_SRx.RELFALL_UPD_FE0RE1 controls if the falling edge (=0) or the rising edge (=1) of DTM_IN[x:x] triggers the update of CLSi_CDTM_DTMd_CH_DTVx.RELFALL.

Figure 1107 shows the extension to Figure 1104.

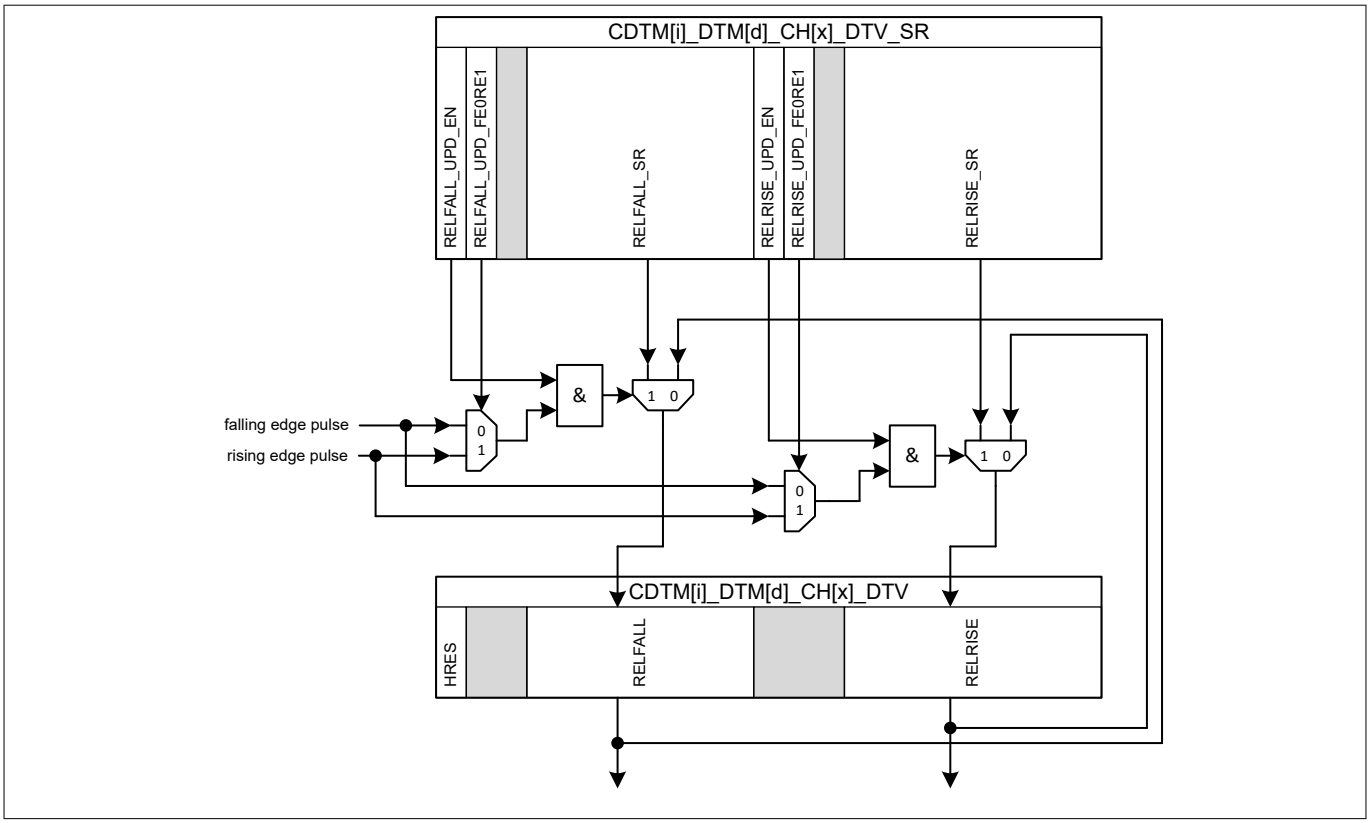


Figure 1107 Dead-time shadow register

44.3.1.9.3 Phase shift unit - overcurrent shutoff feature for full bridge push-pull converters

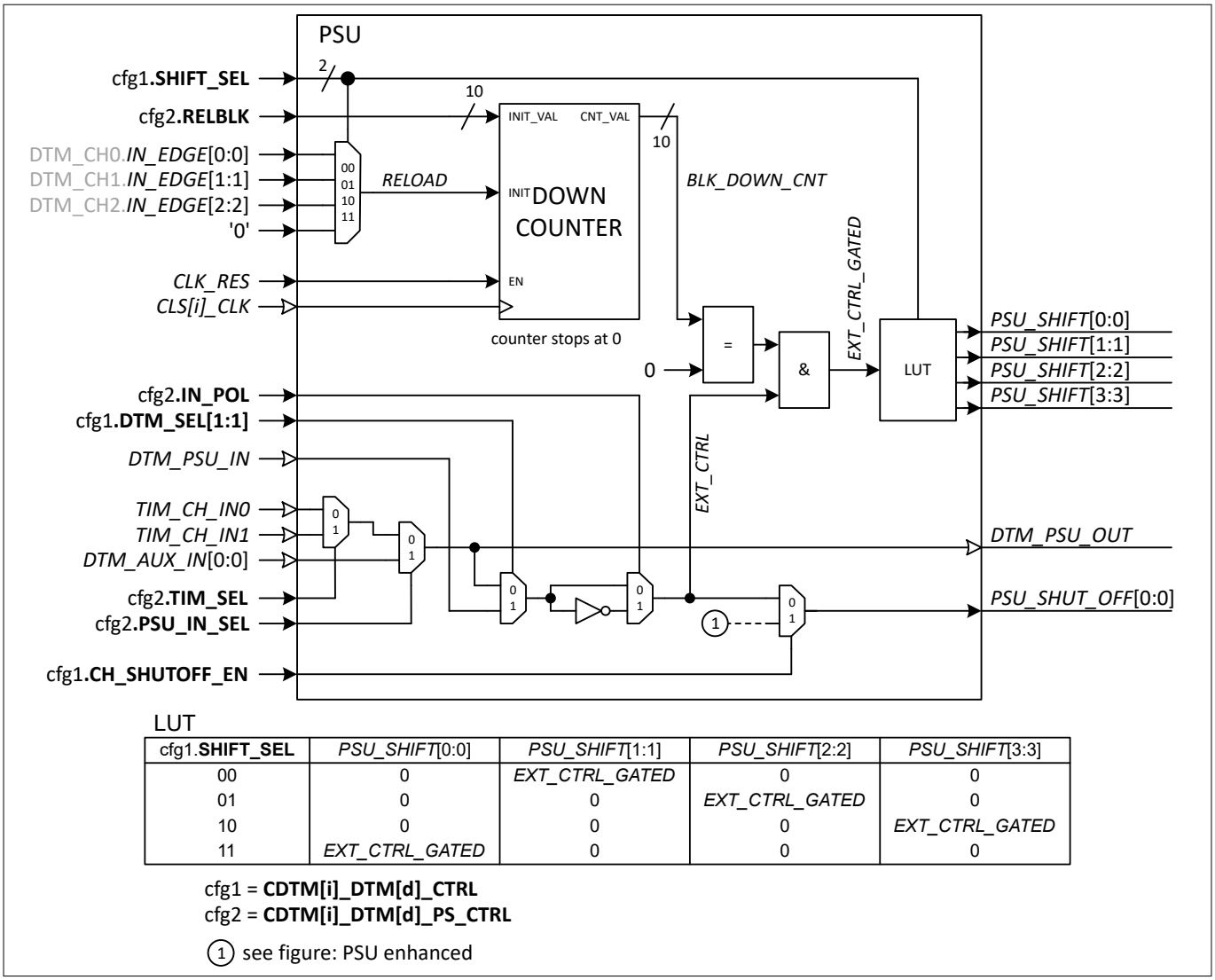
The Phase Shift Unit (PSU) is depicted in the following figure. It supports the second major function of the **DTM** module to copy a PWM signal from a preceding channel to the current channel beginning with the next edge on the preceding channel.

The feature can be used for applications like the phase-shift control of a full bridge push-pull converter. For such an application two consecutive (A)**TOM** channels generate a PWM signal with 50 percent duty cycle. The phase shift between the two PWM signals defines how long two of the four full bridge semiconductors are active and a current flows. An over current detection can trigger the phase shift unit to copy the PWM signal of one **DTM** channel to the other and the semiconductors are switched off directly.

During the development of the function, the following assumptions were made:

- Two consecutive **DTM** channels [x] and [x-1] get e.g. from two (A)**TOM** channels a PWM signal.
- The PWM signals have a phase shift of 180 degrees to each other.
- The configured blanking window value is greater than or equal to the configured dead-time value.
- The configured dead-time is much smaller than the high phase of the PWM signal.
- It was never planned to use this feature in combination with the cross channel dead-time. This is forbidden, because the behavior is undefined.

Figure 1108 shows the corresponding hardware block. The block realizes the blanking window and generates the signal for the **DTM** channel which controls the switch to copy a PWM signal from another **DTM** channel.



LUT

cfg1.**SHIFT_SEL**	*PSU_SHIFT*[0:0]	*PSU_SHIFT*[1:1]	*PSU_SHIFT*[2:2]	*PSU_SHIFT*[3:3]
00	0	*EXT_CTRL_GATED*	0	0
01	0	0	*EXT_CTRL_GATED*	0
10	0	0	0	*EXT_CTRL_GATED*
11	*EXT_CTRL_GATED*	0	0	0

cfg1 = **CDTM[i]_DTM[d]_CTRL**
cfg2 = **CDTM[i]_DTM[d]_PS_CTRL**

(1) see figure: PSU enhanced

Figure 1108 Phase shift unit

The input signals TIM_CH_IN0, TIM_CH_IN1, DTM_AUX_IN and DTM_PSU_IN can be used to signalize e.g. an overcurrent situation and activate the phase shift feature in one of the four **DTM** channels. Over the bit fields CLSi_CDTM_DTMd_PS_CTRL**.**PSU_IN_SEL, CLSi_CDTM_DTMd_PS_CTRL**.**TIM_SEL and CLSi_CDTM_DTMd_CTRL**.**DTM_SEL[1:1] one of the input signals can be selected. Further the polarity of the selected input signal can be inverted by bit field CLSi_CDTM_DTMd_PS_CTRL**.**IN_POL. The resulting signal *EXT_CTRL* is routed over signal *EXT_CTRL_GATED* to the phase shift unit output signal PSU_SHIFT[3:0] which controls the phase shift feature in the four **DTM** channel blocks, see Figure 1104.

Bit field CLSi_CDTM_DTMd_PS_CTRL**.**SHIFT_SEL controls which **DTM** channel is used for the phase shift feature by selecting the corresponding signal IN_EDGE of one of the four **DTM** channels. The bit field controls in the same manner to which **DTM** channel the signal *EXT_CTRL_GATED* is routed. For all other, not selected **DTM** channels signal PSU_SHIFT is set to 0 (see LUT table in Figure 1108.

The counter (DOWN COUNTER) counts down while its value is greater than 0 and if the counter reaches 0 it stops. While the counter is running (BLK_DOWN_CNT>0) signal *EXT_CTRL_GATED* is 0. Thus, the phase shift output signal PSU_SHIFT is 0_H and the phase shift feature can't be activated. If the counter is 0 *EXT_CTRL_GATED==EXT_CTRL* and the phase shift feature can be activated depending on the signal level of the selected input signal TIM_CH_IN0 or TIM_CH_IN1 or DTM_AUX_IN[0:0] or DTM_PSU_IN.

The blanking window usage allows that the phase shift condition (exception condition e.g. overcurrent detected) can be masked for a certain period after an input signal change (e.g. at DTM_IN) has occurred.

The activation of the phase shift feature by signal *EXT_CTRL_GATED*=1 can result in an edge at signal IN[x]_DLY1.

The deactivation of the phase shift feature by signal *EXT_CTRL_GATED*=0 results in an edge on signal IN[x]_DLY1 earliest with a signal change at IN[x]_SIG (see Figure 1104.

If phase shift feature for **DTM** channel 1 is used (CLSi_CDTM_DTMd_PS_CTRL.SHIFT_SEL==00_B signal PSU_SHIFT[1:1] activates (if CLSi_CDTM_DTMd_CH_CTRL1.SH_EN_1=1) the update of the first storage element on channel 1 (i.e. representing IN[x]_DLY1) to the input value DTM_IN_PREV (DTM_IN of channel x-1). If this update leads to an edge, the succeeding part of **DTM** channel derives the inverse signal and applies the corresponding dead-time (i.e. the edge delay) to the output signals of the channel (x=1).

If phase shift feature for **DTM** channel 2 is used (CLSi_CDTM_DTMd_PS_CTRL.SHIFT_SEL==01_B signal PSU_SHIFT[2:2] activates (if CLSi_CDTM_DTMd_CH_CTRL1.SH_EN_2=1) the update of the first storage element on channel 2 (i.e. representing IN[x]_DLY1) to the input value DTM_IN_PREV (DTM_IN of channel x-1). If this update leads to an edge, the succeeding part of **DTM** channel derives the inverse signal and applies the corresponding dead-time (i.e. the edge delay) to the output signals of the channel (x=2).

If phase shift feature for **DTM** channel 3 is used (CLSi_CDTM_DTMd_PS_CTRL.SHIFT_SEL==10_B signal PSU_SHIFT[3:3] activates (if CLSi_CDTM_DTMd_CH_CTRL1.SH_EN_3=1) the update of the first storage element on channel 3 (i.e. representing IN[x]_DLY1) to the input value DTM_IN_PREV (DTM_IN of channel x-1). If this update leads to an edge, the succeeding part of **DTM** channel derives the inverse signal and applies the corresponding dead-time (i.e. the edge delay) to the output signals of the channel (x=3).

For channel x=0 no blanking window feature and no phase shift feature is available.

Further, the exception signal PSU_SHIFT[x:x] can be used as an input to the Boolean function, see Multiple output signal combination (next section: Multiple output signal combination). This feature is available for all four **DTM** channels.

The following figure shows an example of phase shifting on channel 1.

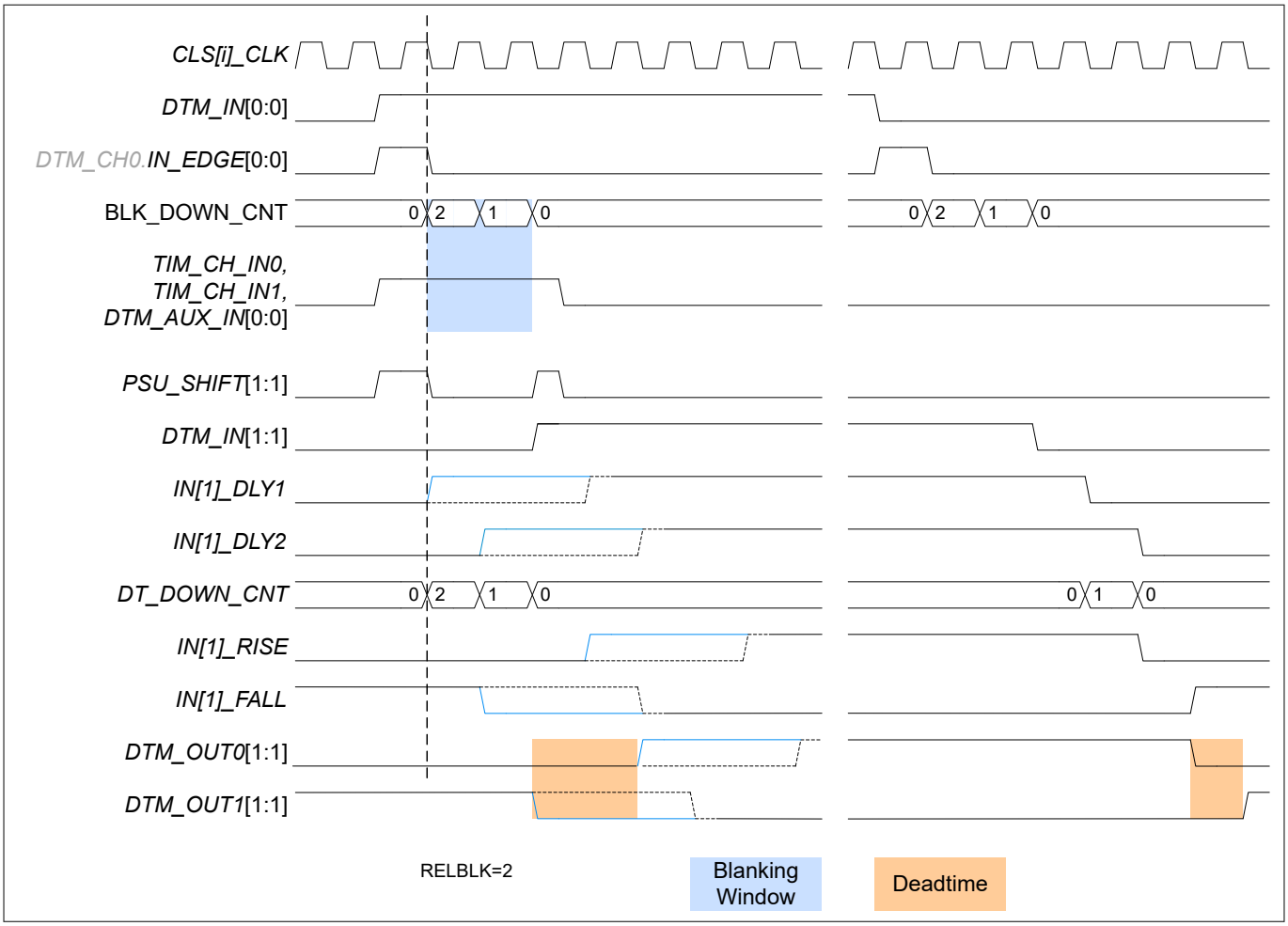


Figure 1109 Example wave of phase shift on channel 1

***Note**: The figure above shows the timing for CLSi_CDTM_DTMd_CTRL.CLK_SEL=00_B. If another clock resolution is configured, be aware that only the counter DT_DOWN_CNT runs with the reduced clock resolution.*

The bit field CLSi_CDTM_DTMd_CTRL.CH_SHUTOFF_EN shown in Figure 1108 is used for the **DTM** output shutoff feature which is described in **DTM** shutoff.

44.3.1.9.4 Multiple output signal combination

Additionally, each **DTM** channel (see Figure 1104) provides the possibility to combine the channel input signals DTM_IN or DTM_IN_T with signals PSU_SHIFT or CH_OUT_I by an **AND** or an **XOR** gate.

The relevant input signals of **DTM** channel [0] for this feature are:

- DTM_IN[0:0]
- DTM_IN_T[0:0]
- PSU_SHIFT[0:0]
- CH_OUT_I

***Note**: Signal CH_OUT_I of channel [0] is equal to signal DTM_COUT3_I, see Figure 1100.*

The relevant input signals of **DTM** channel [1] for this feature are:

- DTM_IN[1:1]

- DTM_IN_T[1:1]
- PSU_SHIFT[1:1]
- CH_OUT_I

***Note**: Signal CH_OUT_I of channel [1] is equal to signal CH_OUT_O of channel [0], see Figure 1100.*

The relevant input signals of **DTM** channel [2] for this feature are:

- DTM_IN[2:2]
- DTM_IN_T[2:2]
- PSU_SHIFT[2:2]
- CH_OUT_I

***Note**: Signal CH_OUT_I of channel [2] is equal to signal CH_OUT_O of channel [1], see Figure 1100.*

The relevant input signals of **DTM** channel [3] for this feature are:

- DTM_IN[3:3]
- DTM_IN_T[3:3]
- PSU_SHIFT[3:3]
- CH_OUT_I

***Note**: Signal CH_OUT_I of channel [3] is equal to signal CH_OUT_O of channel [2], see Figure 1100.*

The registers CLSi_CDTM_DTMd_CH_CTRL3.TSEL1_0, CLSi_CDTM_DTMd_CH_CTRL3.TSEL1_1, CLSi_CDTM_DTMd_CH_CTRL3.TSEL1_2 and CLSi_CDTM_DTMd_CH_CTRL3.TSEL1_3 select if the signal DTM_IN or DTM_IN_T is used.

The registers CLSi_CDTM_DTMd_CH_CTRL3.CII0, CLSi_CDTM_DTMd_CH_CTRL3.CII1, CLSi_CDTM_DTMd_CH_CTRL3.CII2 and CLSi_CDTM_DTMd_CH_CTRL3.CII3 configure if the selected input signal DTM_IN or DTM_IN_T is inverted.

The registers CLSi_CDTM_DTMd_CH_CTRL1.I1SEL_0, CLSi_CDTM_DTMd_CH_CTRL1.I1SEL_1, CLSi_CDTM_DTMd_CH_CTRL1.I1SEL_2 and CLSi_CDTM_DTMd_CH_CTRL1.I1SEL_3 select if the signal PSU_SHIFT or signal CH_OUT_I is used.

It is recommended to use the combination of signals only if bit field CLSi_CDTM_DTMd_PS_CTRL.RELBLK is 0. Otherwise, the signal TIM_CH_IN0, TIM_CH_IN1 or DTM_AUX_IN[0:0] may be disturbed by the blanking window counter.

Together with the inverter inside sub-module PSU (see Figure 1108, the inverter on each output of a channel and the possibility to invert the selected input signal (DTM_IN or DTM_IN_T) inside connected **TOM** / **ATOM** channel, a (N)**AND**, (N)**OR** or X(N)**OR** combination of the signals is possible (see e.g. Table 2243).

The inverter inside the sub-module PSU can be configured by CLSi_CDTM_DTMd_PS_CTRL.IN_POL.

The inverter on each output of a **DTM** channel can be configured by

CLSi_CDTM_DTMd_CH_CTRL2.POL0_0, CLSi_CDTM_DTMd_CH_CTRL2.POL1_0,

CLSi_CDTM_DTMd_CH_CTRL2.POL0_1, CLSi_CDTM_DTMd_CH_CTRL2.POL1_1,

CLSi_CDTM_DTMd_CH_CTRL2.POL0_2, CLSi_CDTM_DTMd_CH_CTRL2.POL1_2,

CLSi_CDTM_DTMd_CH_CTRL2.POL0_3 and CLSi_CDTM_DTMd_CH_CTRL2.POL1_3.

Combination of input signal TIM_CH_INy/DTM_AUX_IN with TOM/ATOM signal

***Note**: The description below is given for channel [0], but the same applies for the other **DTM** channels [1..3] using the corresponding configuration register bit fields.*

If the input selection CLSi_CDTM_DTMd_CH_CTRL1.I1SEL_0 is set to 0, the output selection CLSi_CDTM_DTMd_CH_CTRL1.O1SEL_0 is set to 1, CLSi_CDTM_DTMd_CH_CTRL2.DT1_0 is set to 1, CLSi_CDTM_DTMd_CH_CTRL2.OC1_0 is set to 0 and CLSi_CDTM_DTMd_CH_CTRL1.SWAP_0 is set to 0, depending on CLSi_CDTM_DTMd_PS_CTRL.PSU_IN_SEL either TIM_CH_IN0, TIM_CH_IN1 or DTM_AUX_IN[0:0] can be combined with signal DTM_IN[0:0] or DTM_IN_T[0:0].

The function of combination on **DTM** output DTM_OUT1[0:0] (and also CH_OUT_O of channel [0]) is defined by CLSi_CDTM_DTMd_CH_CTRL1.O1F_0 and CLSi_CDTM_DTMd_CH_CTRL2.POL1_0 in the following way:

Table 2243 Function of combination on DTM channel x=0 output

	CLSi_CDTM_DTMd_CH_CTRL1.O1F_0	CLSi_CDTM_DTMd_CH_CTRL2.POL1_0	CLSi_CDTM_DTMd_PS_CTRL.IN_POL	TOM/ATOM output inverted
XOR	01	0	0	no
AND	10	0	0	no
XNOR	01	1	0	no
NAND	10	1	0	no
XNOR	01	1	1	yes
OR	10	1	1	yes
XOR	01	0	1	yes
NOR	10	0	1	yes

***Note**: It is pointed out that the rightmost column of the table above is to be understood as a prerequisite. This column shows the state of the input of the **DTM** which is the output of the respective source on the **TOM/ATOM**. "Output inverted" in this column means that the output is inverted at its source module (**TOM/ATOM/TIO**).*

Table 2244 Function of combination on DTM channel x=1 output

	CLSi_CDTM_DTMd_CH_CTRL1.O1F_1	CLSi_CDTM_DTMd_CH_CTRL2.POL1_1	CLSi_CDTM_DTMd_PS_CTRL.IN_POL	TOM/ATOM output inverted
XOR	01	0	0	no
AND	10	0	0	no
XNOR	01	1	0	no
NAND	10	1	0	no
XNOR	01	1	1	yes
OR	10	1	1	yes
XOR	01	0	1	yes
NOR	10	0	1	yes

***Note**: It is pointed out that the rightmost column of the table above is to be understood as a prerequisite. This column shows the state of the input of the **DTM** which is the output of the respective source on the **TOM/ATOM**. "Output inverted" in this column means that the output is inverted at its source module (**TOM/ATOM**).*

Table 2245 Function of combination on DTM channel x=2 output

	CLSi_CDTM_DTMd_CH_CTRL1.O1F_2	CLSi_CDTM_DTMd_CH_CTRL2.POL1_2	CLSi_CDTM_DTMd_PS_CTRL.IN_POL	TOM/ATOM output inverted
XOR	01	0	0	no
AND	10	0	0	no
XNOR	01	1	0	no
NAND	10	1	0	no
XNOR	01	1	1	yes
OR	10	1	1	yes
XOR	01	0	1	yes
NOR	10	0	1	yes

***Note**: It is pointed out that the rightmost column of the table above is to be understood as a prerequisite. This column shows the state of the input of the **DTM** which is the output of the respective source on the **TOM**/**ATOM**. "Output inverted" in this column means that the output is inverted at its source module (**TOM**/**ATOM**).*

Table 2246 Function of combination on DTM channel x=3 output

	CLSi_CDTM_DTMd_CH_CTRL1.O1F_3	CLSi_CDTM_DTMd_CH_CTRL2.POL1_3	CLSi_CDTM_DTMd_PS_CTRL.IN_POL	TOM/ATOM output inverted
XOR	01	0	0	no
AND	10	0	0	no
XNOR	01	1	0	no
NAND	10	1	0	no
XNOR	01	1	1	yes
OR	10	1	1	yes
XOR	01	0	1	yes
NOR	10	0	1	yes

***Note**: It is pointed out that the rightmost column of the table above is to be understood as a prerequisite. This column shows the state of the input of the **DTM** which is the output of the respective source on the **TOM**/**ATOM**. "Output inverted" in this column means that the output is inverted at its source module (**TOM**/**ATOM**).*

***Note**: The inversion of the (A)**TOM** output can be achieved by switching the CLSi_ATOM_CHx_CTRL.SL bit (for **TOM** and **ATOM** SOMP/SOMC mode).*

Combination of multiple TOM/ATOM output signals

***Note**: The description below is given for channel [1], but the same applies for the **DTM** channels [2..3] using the corresponding configuration registers bit fields.*

If the input selection CLSi_CDTM_DTMd_CH_CTRL1.I1SEL_1 is set to 1, the output selection CLSi_CDTM_DTMd_CH_CTRL1.O1SEL_1 is set to 1, CLSi_CDTM_DTMd_CH_CTRL2.DT1_1 is set to 1, CLSi_CDTM_DTMd_CH_CTRL2.OC1_1 is set to 0 and CLSi_CDTM_DTMd_CH_CTRL1.SWAP_1 is set to 0, the output of the preceding **DTM** channel CH_OUT_I can be combined with signal DTM_IN[1:1] or with signal DTM_IN_T[1:1] depending on CLSi_CDTM_DTMd_CH_CTRL3.TSEL1_1.

The function of combination on **DTM** output DTM_OUT1[1:1] is defined by CLSi_CDTM_DTMd_CH_CTRL1.O1F_1, CLSi_CDTM_DTMd_CH_CTRL2.POL1_1 and CLSi_CDTM_DTMd_CH_CTRL2.POL1_0 in the following way:

Table 2247 Function of combination on DTM on channel [1] output DTM_OUT1 *[1:1]*

	CLSi_CDTM_DTMd_CH_CTRL1.O1F_1	CLSi_CDTM_DTMd_CH_CTRL2.POL1_1	CLSi_CDTM_DTMd_CH_CTRL2.POL1_0	TOM/ATOM output inverted
XOR	01	0	0	no
AND	10	0	0	no
XNOR	01	1	0	no
NAND	10	1	0	no
XNOR	01	1	1	yes
OR	10	1	1	yes
XOR	01	0	1	yes
NOR	10	0	1	yes

Table 2248 Function of combination on DTM on channel [2] output DTM_OUT1 *[2:2]*

	CLSi_CDTM_DTMd_CH_CTRL1.O1F_2	CLSi_CDTM_DTMd_CH_CTRL2.POL1_2	CLSi_CDTM_DTMd_CH_CTRL2.POL1_1	TOM/ATOM output inverted
XOR	01	0	0	no
AND	10	0	0	no
XNOR	01	1	0	no
NAND	10	1	0	no
XNOR	01	1	1	yes
OR	10	1	1	yes
XOR	01	0	1	yes
NOR	10	0	1	yes

Table 2249 Function of combination on DTM on channel [3] output DTM_OUT1 *[3:3]*

	CLSi_CDTM_DTMd_CH_CTRL1.O1F_3	CLSi_CDTM_DTMd_CH_CTRL2.POL1_3	CLSi_CDTM_DTMd_CH_CTRL2.POL1_2	TOM/ATOM output inverted
XOR	01	0	0	no
AND	10	0	0	no
XNOR	01	1	0	no
NAND	10	1	0	no
XNOR	01	1	1	yes

(table continues...)

Table 2249 (continued) Function of combination on DTM on channel [3] output DTM_OUT1 *[3:3]*

	CLSi_CDTM_DTMd_CH_CTRL1.O1F_3	CLSi_CDTM_DTMd_CH_CTRL2.POL1_3	CLSi_CDTM_DTMd_CH_CTRL2.POL1_2	**TOM/ATOM output inverted**
OR	10	1	1	yes
XOR	01	0	1	yes
NOR	10	0	1	yes

By setting CLSi_CDTM_DTMd_CH_CTRL1.I1SEL_0 to 1, CLSi_CDTM_DTMd_CH_CTRL1.I1SEL_1 to 1, CLSi_CDTM_DTMd_CH_CTRL1.I1SEL_2 to 1 and CLSi_CDTM_DTMd_CH_CTRL1.I1SEL_3 to 1, a combination of all four signals DTM_IN[0:0], DTM_IN[1:1], DTM_IN[2:2] and DTM_IN[3:3] can be achieved (combinatorial chain).

If the result of the combination of the signals should be visible at the **DTM** output DTM_OUT0[x:x], the internal signals just in front of the output register stage can be swapped by setting CLSi_CDTM_DTMd_CH_CTRL1.SWAP_0 to 1 for channel 0, CLSi_CDTM_DTMd_CH_CTRL1.SWAP_1 to 1 for channel 1, CLSi_CDTM_DTMd_CH_CTRL1.SWAP_2 to 1 for channel 2 and CLSi_CDTM_DTMd_CH_CTRL1.SWAP_3 to 1 for channel 3.

Pulse generation on edge

Another feature of the **DTM** is to generate on the second output DTM_OUT1[x:x] a pulse on every edge of corresponding input signal DTM_IN[x:x].

This can be achieved for channel [0] by configuring CLSi_CDTM_DTMd_CH_CTRL1.O1SEL_0 to 1, i.e. selecting signal EDGE_TRIGG_[x] as the output signal (CLSi_CDTM_DTMd_CH_CTRL1.O1F_0 has to be 00_B).

This can be achieved for channel [1] by configuring CLSi_CDTM_DTMd_CH_CTRL1.O1SEL_1 to 1, i.e. selecting signal EDGE_TRIGG_[x] as the output signal (CLSi_CDTM_DTMd_CH_CTRL1.O1F_1 has to be 00_B).

This can be achieved for channel [2] by configuring CLSi_CDTM_DTMd_CH_CTRL1.O1SEL_2 to 1, i.e. selecting signal EDGE_TRIGG_[x] as the output signal (CLSi_CDTM_DTMd_CH_CTRL1.O1F_2 has to be 00_B).

This can be achieved for channel [3] by configuring CLSi_CDTM_DTMd_CH_CTRL1.O1SEL_3 to 1, i.e. selecting signal EDGE_TRIGG_[x] as the output signal (CLSi_CDTM_DTMd_CH_CTRL1.O1F_3 has to be 00_B).

The signal EDGE_TRIGG_[x] is depicted in Figure 1105.

The pulse length can be adjusted individually for each edge type by the configuration value CLSi_CDTM_DTMd_CH_DTVx.RELRISE and CLSi_CDTM_DTMd_CH_DTVx.RELFALL.

The parameter CLSi_CDTM_DTMd_CH_DTVx.RELRISE defines the pulse length in case of a rising edge on input DTM_IN[x:x], the parameter CLSi_CDTM_DTMd_CH_DTVx.RELFALL defines the pulse length in case of a falling edge on input DTM_IN[x:x].

The generated edge signal EDGE_TRIGG_[x] can be combined with the output signal of the preceding **DTM** channel [x-1] at channel input CH_OUT_I.

To route the output of the **XOR** gate to DTM_OUT1, here for example for channel 0, the CLSi_CDTM_DTMd_CH_CTRL1.O1F_0 is set to 01_B, CLSi_CDTM_DTMd_CH_CTRL1.O1SEL_0 is set to 1, CLSi_CDTM_DTMd_CH_CTRL2.DT1_0 is set to 1 and CLSi_CDTM_DTMd_CH_CTRL2.SL1_0 is set to 0. For the other channels the corresponding configuration bits have to be set.

With the configuration CLSi_CDTM_DTMd_CH_CTRL3.CIS0 is set to 1 and CLSi_CDTM_DTMd_CH_CTRL1.I1SEL_0 is set to 1, CLSi_CDTM_DTMd_CH_CTRL3.CII0 is set to 0 and CLSi_CDTM_DTMd_CH_CTRL2.POL1_3 is set to 1, the signal EDGE_TRIGG_[x] is **XOR**ed with the inverse signal DTM_COUT3_I (CH_OUT_O of channel [3]).

With the configuration CLSi_CDTM_DTMd_CH_CTRL3.CIS1 is set to 1 and CLSi_CDTM_DTMd_CH_CTRL1.I1SEL_1 is set to 1, CLSi_CDTM_DTMd_CH_CTRL3.CII1 is set to 0 and CLSi_CDTM_DTMd_CH_CTRL2.POL1_0 is set to 1, the signal EDGE_TRIGG_[x] is **XOR**ed with the inverse signal COUT[0:0] (CH_OUT_O of channel [0]).

With the configuration CLSi_CDTM_DTMd_CH_CTRL3.CIS2 is set to 1 and CLSi_CDTM_DTMd_CH_CTRL1.I1SEL_2 is set to 1, CLSi_CDTM_DTMd_CH_CTRL3.CII2 is set to 0 and CLSi_CDTM_DTMd_CH_CTRL2.POL1_1 is set to 1, the signal EDGE_TRIGG_[x] is **XOR**ed with the inverse signal COUT[1:1] (CH_OUT_O of channel [1]).

With the configuration CLSi_CDTM_DTMd_CH_CTRL3.CIS3 is set to 1 and CLSi_CDTM_DTMd_CH_CTRL1.I1SEL_3 is set to 1, CLSi_CDTM_DTMd_CH_CTRL3.CII3 is set to 0 and CLSi_CDTM_DTMd_CH_CTRL2.POL1_2 is set to 1, the signal EDGE_TRIGG_[x] is **XOR**ed with the inverse signal COUT[2:2] (CH_OUT_O of channel [2]).

As a result of this configuration, one can generate at each edge on **DTM** input DTM_IN[x:x] a pulse signal and **OR**-combine these generated pulse signals with the generated signal of preceding **DTM** channel. If the combinatorial chain is configured over all four **DTM** channels the final signal is available at last **DTM** output, which is, according to Figure 1100 the DTM_COUT3_O and with one clock delay the DTM_OUT1[3:3].

44.3.1.9.5 Synchronous update of channel control register 2

It is possible to use the shadow register CLSi_CDTM_DTMd_CH_CTRL2_SR and a selected edge of one of the channel 0 to channel 3 to update the operation register CLSi_CDTM_DTMd_CH_CTRL2.

The update mechanism and its configuration are depicted in the following figure.

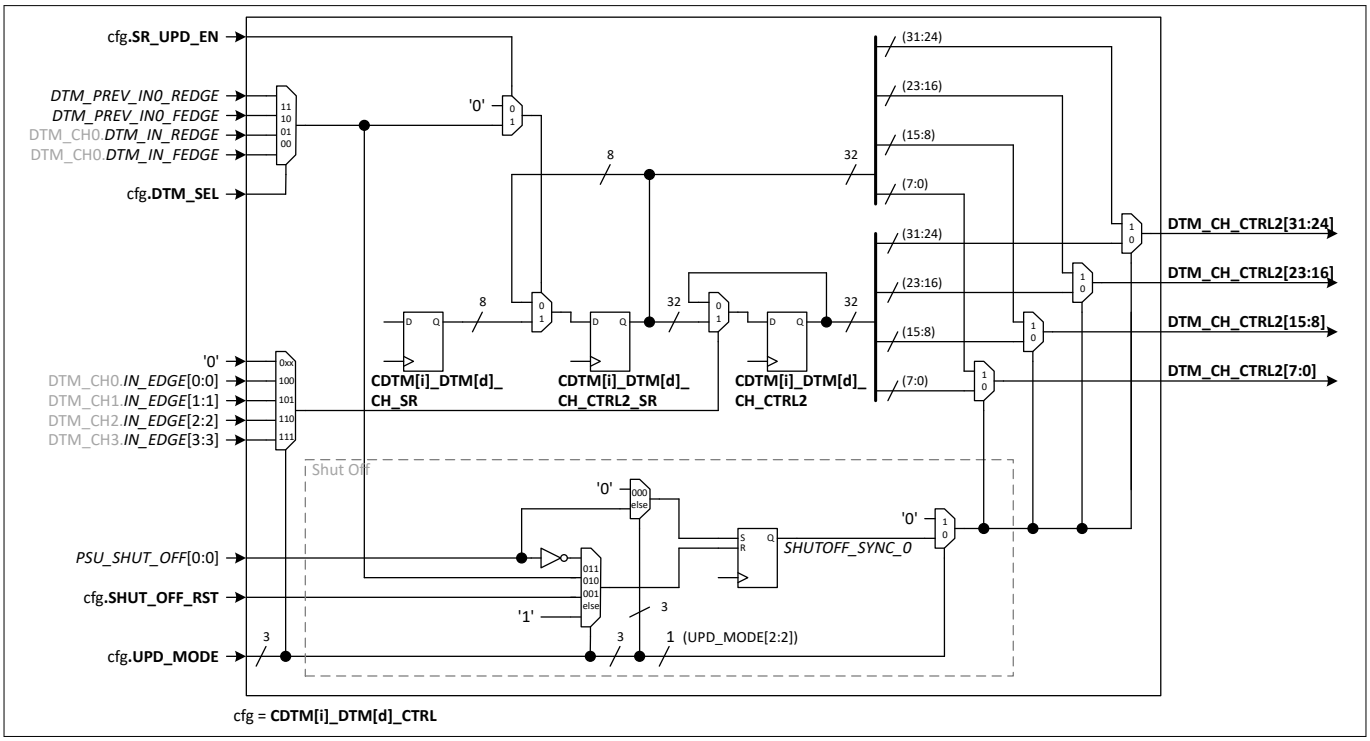


Figure 1110 Synchronous update mechanism of register CLSi_CDTM_DTMd_CH_CTRL2

If enabled by the bit field CLSi_CDTM_DTMd_CTRL.UPD_MODE (i.e. CLSi_CDTM_DTMd_CTRL.UPD_MODE=1xx), the register CLSi_CDTM_DTMd_CH_CTRL2_SR serves as a shadow register of register CLSi_CDTM_DTMd_CH_CTRL2. The update is then triggered by an edge of signal IN_EDGE[3:0].

The synchronous update allows the user to change output polarity. The selection of the constant signal level and the routing from **DTM** inputs (DTM_IN, DTM_IN_T) to outputs (DTM_OUT0, DTM_OUT1) is synchronized for all four channels in parallel to one of the input edges on IN_EDGE[3:0].

Any asynchronous configuration change (asynchronous means a configuration change will be done immediately without any edge detection) can cause a change of DTM_OUT0_HRES_[x] and DTM_OUT1_HRES_[x] outputs without any input edge detection, even if the corresponding output DTM_OUT0[x:x] or DTM_OUT1[x:x] also changes.

44.3.1.9.6 DTM shutoff

A fast shutoff for the eight outputs of **DTM** instance d can be triggered by one of the two assigned inputs TIM_CH_IN0, TIM_CH_IN1 or DTM_AUX_IN[0:0] or the corresponding three inputs of the previous **DTM** instance d-1 by selecting signal DTM_PSU_IN. The selection of the trigger signal source is done by the bits CLSi_CDTM_DTMd_PS_CTRL.TIM_SEL, CLSi_CDTM_DTMd_PS_CTRL.PSU_IN_SEL and CLSi_CDTM_DTMd_CTRL.DTM_SEL[1:1] (see Figure 1108). The selected trigger signal (output signal in Figure 1108 and input signal Figure 1110) is named PSU_SHUT_OFF.

Enabling of the shutoff feature is done by setting CLSi_CDTM_DTMd_CTRL.UPD_MODE[2:0] to one of the values 001_B, 010_B or 011_B.

The shutoff behavior of the **DTM** outputs is defined by the value of register CLSi_CDTM_DTMd_CH_CTRL2_SR.

If the shutoff feature is enabled by CLSi_CDTM_DTMd_CTRL.UPD_MODE, the register CLSi_CDTM_DTMd_CH_CTRL2 defines the output signal behavior as long as signal SHUTOFF_SYNC_0 is 0. If signal SHUTOFF_SYNC_0 is 1 the output signal behavior is defined by register CLSi_CDTM_DTMd_CH_CTRL2_SR.

Signal SHUTOFF_SYNC_0 is set to 1 if signal PSU_SHUT_OFF switches to 1. The reset depends on value of CLSi_CDTM_DTMd_CTRL.UPD_MODE.

There are four different ways to reset the signal SHUTOFF_SYNC_0 to 0:

- A write-access of value 1 via the configuration interface to bit CLSi_CDTM_DTMd_CTRL.SHUT_OFF_RST (with CLSi_CDTM_DTMd_CTRL.UPD_MODE = 001_B).
- Synchronous to an edge on **DTM** channel 0 input of this **DTM** instance d or on an edge on **DTM** channel 0 input of preceding **DTM** instance d-1 (with CLSi_CDTM_DTMd_CTRL.UPD_MODE = 010_B).
- Asynchronous to any **DTM** channel input edge, if signal PSU_SHUT_OFF switches back to 0 (with CLSi_CDTM_DTMd_CTRL.UPD_MODE = 011_B).
- Additionally, setting CLSi_CDTM_DTMd_CTRL.UPD_MODE to a value 000_B or $1xx_B$ resets also the signal SHUTOFF_SYNC_0, except PSU_SHUT_OFF is set and CLSi_CDTM_DTMd_CTRL.UPD_MODE is unequal to 000_B.

Figure 1110 depicts the shutoff feature and the different shutoff release possibilities.

The reset of SHUTOFF_SYNC_0 has lower priority than the set of this signal.

A second shadow register CLSi_CDTM_DTMd_CH_SR exists for the eight bits of the shadow register CLSi_CDTM_DTMd_CH_CTRL2_SR.

If enabled by configuration bit CLSi_CDTM_DTMd_CTRL.SR_UPD_EN, the update of CLSi_CDTM_DTMd_CH_CTRL2_SR.SL0_0_SR, CLSi_CDTM_DTMd_CH_CTRL2_SR.SL1_0_SR, CLSi_CDTM_DTMd_CH_CTRL2_SR.SL0_1_SR, CLSi_CDTM_DTMd_CH_CTRL2_SR.SL1_1_SR, CLSi_CDTM_DTMd_CH_CTRL2_SR.SL0_2_SR, CLSi_CDTM_DTMd_CH_CTRL2_SR.SL1_2_SR, CLSi_CDTM_DTMd_CH_CTRL2_SR.SL0_3_SR and CLSi_CDTM_DTMd_CH_CTRL2_SR.SL1_3_SR bits can be triggered by one of the signals selected by bit field CLSi_CDTM_DTMd_CTRL.DTM_SEL. This trigger signal is either the rising or the falling edge detected on **DTM** channel [0] (DTM_IN_FEDGE or DTM_IN_REDGE) or the corresponding edges of the **DTM** channel [0] of a preceding instance d-1 (DTM_PREV_IN0_FEDGE or DTM_PREV_IN0_REDGE).

If an update trigger occurs in the same clock-cycle, a write-access to CLSi_CDTM_DTMd_CH_CTRL2_SR is done and the register is written except the eight bits: CLSi_CDTM_DTMd_CH_CTRL2_SR.SL0_0_SR, CLSi_CDTM_DTMd_CH_CTRL2_SR.SL1_0_SR, CLSi_CDTM_DTMd_CH_CTRL2_SR.SL0_1_SR, CLSi_CDTM_DTMd_CH_CTRL2_SR.SL1_1_SR, CLSi_CDTM_DTMd_CH_CTRL2_SR.SL0_2_SR, CLSi_CDTM_DTMd_CH_CTRL2_SR.SL1_2_SR, CLSi_CDTM_DTMd_CH_CTRL2_SR.SL0_3_SR and CLSi_CDTM_DTMd_CH_CTRL2_SR.SL1_3_SR. The eight bits are copied from CLSi_CDTM_DTMd_CH_SR. In this case CLSi_CDTM_DTMd_CH_SR has priority over **AEI** write data.

As depicted in Figure 1102, Figure 1103 and Figure 1108 the **DTM** input signal TIM_CH_IN0, TIM_CH_IN1 or DTM_AUX_IN[0:0] can be forwarded to the succeeding instance d+1. Thus, it can be used to trigger shutoff in two consecutive **DTM** instances.

44.3.1.9.7 DTM individual channel shutoff

The shutoff feature can be enhanced by setting bit CLSi_CDTM_DTMd_CTRL.CH_SHUTOFF_EN = 1. Then all four **DTM** channels can be shut off individually. If an individual shutoff is done (SHUTOFF_SYNC_0, SHUTOFF_SYNC_1, SHUTOFF_SYNC_2 and SHUTOFF_SYNC_3 is set) the configuration bits of the corresponding **DTM** channel (e.g. for channel 0 the bits [7:0]) are copied from the shadow register CLSi_CDTM_DTMd_CH_CTRL2_SR to output CLSi_CDTM_DTMd_CH_CTRL2.

As described above, there are four different ways to reset the signal SHUTOFF_SYNC_0 to 0. The configuration bits CLSi_CDTM_DTMd_CTRL2.UPD_MODE_0 are described below:

- Asynchronous to any **DTM** channel input edge, if signal PSU_SHUT_OFF[0:0] switches back to 0 (with CLSi_CDTM_DTMd_CTRL2.UPD_MODE_0 = 11_B)
- Synchronous to an edge on **DTM** channel 0 input of this **DTM** instance d or on an edge on **DTM** channel 0 input of preceding **DTM** instance d-1 (with CLSi_CDTM_DTMd_CTRL2.UPD_MODE_0 = 10_B).
- The CPU writes a 1 to bit CLSi_CDTM_DTMd_CTRL2.SHUT_OFF_RST_0 (with CLSi_CDTM_DTMd_CTRL2.UPD_MODE_0= 01_B).
- Additionally, setting CLSi_CDTM_DTMd_CTRL2.UPD_MODE_0 to a value 00_B.

As described above, there are four different ways to reset the signal SHUTOFF_SYNC_1 to 0. The configuration bits CLSi_CDTM_DTMd_CTRL2.UPD_MODE_1 are described below:

- Asynchronous to any **DTM** channel input edge, if signal PSU_SHUT_OFF[1:1] switches back to 0 (with CLSi_CDTM_DTMd_CTRL2.UPD_MODE_1 = 11_B)
- Synchronous to an edge on **DTM** channel 0 input of this **DTM** instance d or on an edge on **DTM** channel 0 input of preceding **DTM** instance d-1 (with CLSi_CDTM_DTMd_CTRL2.UPD_MODE_1 = 10_B).
- The CPU writes a 1 to bit CLSi_CDTM_DTMd_CTRL2.SHUT_OFF_RST_1 (with CLSi_CDTM_DTMd_CTRL2.UPD_MODE_1= 01_B).
- Additionally, setting CLSi_CDTM_DTMd_CTRL2.UPD_MODE_1 to a value 00_B.

As described above, there are four different ways to reset the signal SHUTOFF_SYNC_2 to 0. The configuration bits CLSi_CDTM_DTMd_CTRL2.UPD_MODE_2 are described here:

- Asynchronous to any **DTM** channel input edge, if signal PSU_SHUT_OFF[2:2] switches back to 0 (with CLSi_CDTM_DTMd_CTRL2.UPD_MODE_2 = 11_B)
- Synchronous to an edge on **DTM** channel 0 input of this **DTM** instance d or on an edge on **DTM** channel 0 input of preceding **DTM** instance d-1 (with CLSi_CDTM_DTMd_CTRL2.UPD_MODE_2 = 10_B).
- The CPU writes a 1 to bit CLSi_CDTM_DTMd_CTRL2.SHUT_OFF_RST_2 (with CLSi_CDTM_DTMd_CTRL2.UPD_MODE_2= 01_B).
- Additionally, setting CLSi_CDTM_DTMd_CTRL2.UPD_MODE_2 to a value 00_B.

As described above, there are four different ways to reset the signal SHUTOFF_SYNC_3 to 0. The configuration bits CLSi_CDTM_DTMd_CTRL2.UPD_MODE_3 are described here:

- Asynchronous to any **DTM** channel input edge, if signal PSU_SHUT_OFF[3:3] switches back to 0 (with CLSi_CDTM_DTMd_CTRL2.UPD_MODE_3 = 11_B)
- Synchronous to an edge on **DTM** channel 0 input of this **DTM** instance d or on an edge on **DTM** channel 0 input of preceding **DTM** instance d-1 (with CLSi_CDTM_DTMd_CTRL2.UPD_MODE_3 = 10_B).
- The CPU writes a 1 to bit CLSi_CDTM_DTMd_CTRL2.SHUT_OFF_RST_3 (with CLSi_CDTM_DTMd_CTRL2.UPD_MODE_3= 01_B).
- Additionally, setting CLSi_CDTM_DTMd_CTRL2.UPD_MODE_3 to a value 00_B.

Figure 1111 depicts the channel shutoff feature and the different shutoff release possibilities.

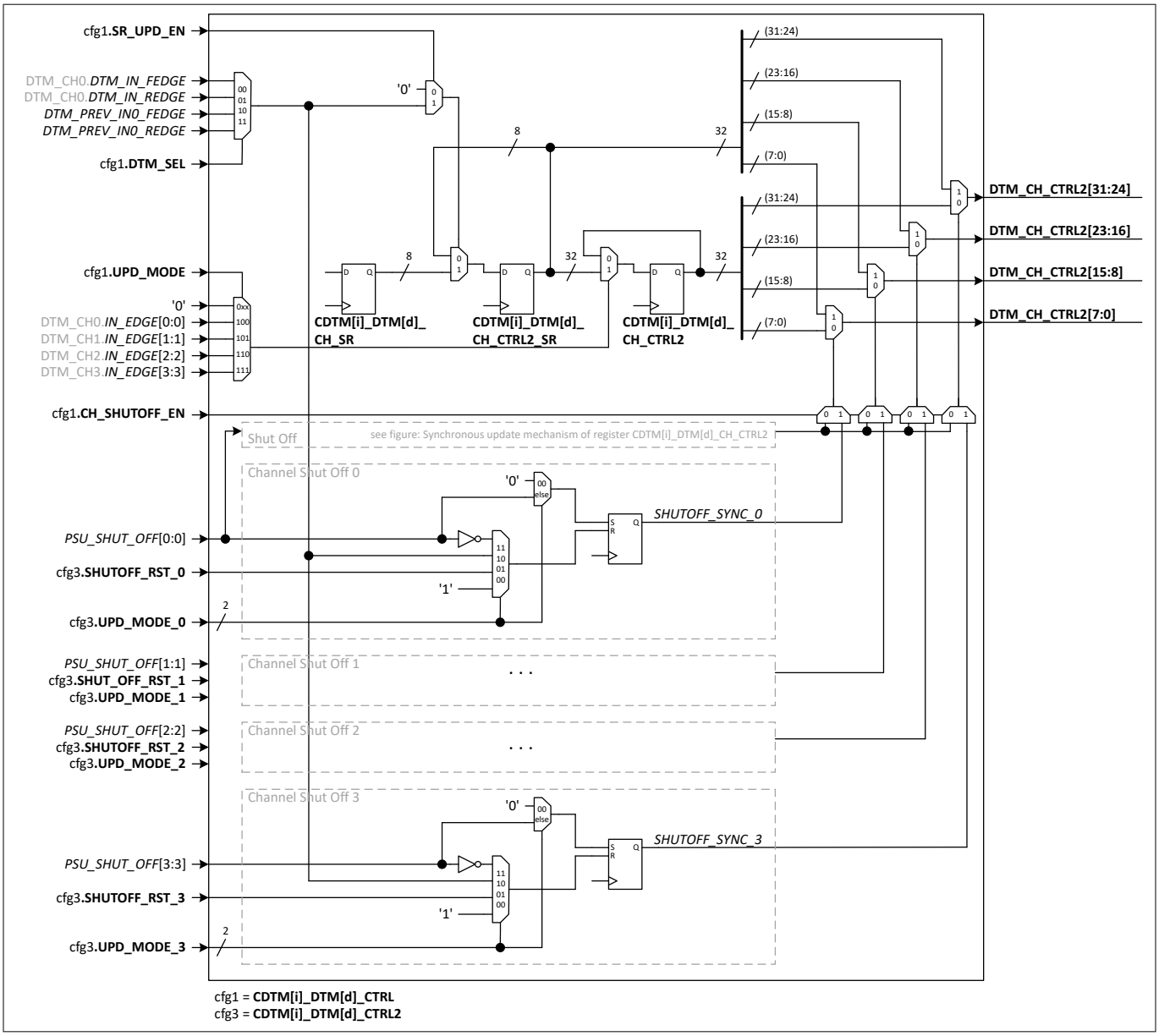


Figure 1111 Individual channel shutoff

The reset of a SHUTOFF_SYNC_0, SHUTOFF_SYNC_1, SHUTOFF_SYNC_2 and SHUTOFF_SYNC_3 has lower priority than the set of such a signal.

If the individual channel shutoff feature is activated (CLSi_CDTM_DTMd_CTRL.CH_SHUTOFF_EN = 1) the selection of the trigger source located in **DTM** block Phase Shift Unit (PSU) is also enhanced.

By using the bit fields CLSi_CDTM_DTMd_CTRL2.SHUTOFF_SEL_0, CLSi_CDTM_DTMd_CTRL2.SHUTOFF_SEL_1, CLSi_CDTM_DTMd_CTRL2.SHUTOFF_SEL_2 and CLSi_CDTM_DTMd_CTRL2.SHUTOFF_SEL_3 for each **DTM** channel one of the following sources can be selected:

- TIM_CH_IN0
- TIM_CH_IN1
- DTM_AUX_IN[0:0]
- DTM_AUX_IN[1:1]
- DTM_AUX_IN[2:2]
- DTM_AUX_IN[3:3]

- constant zero (0_B)
- constant one (1_B)

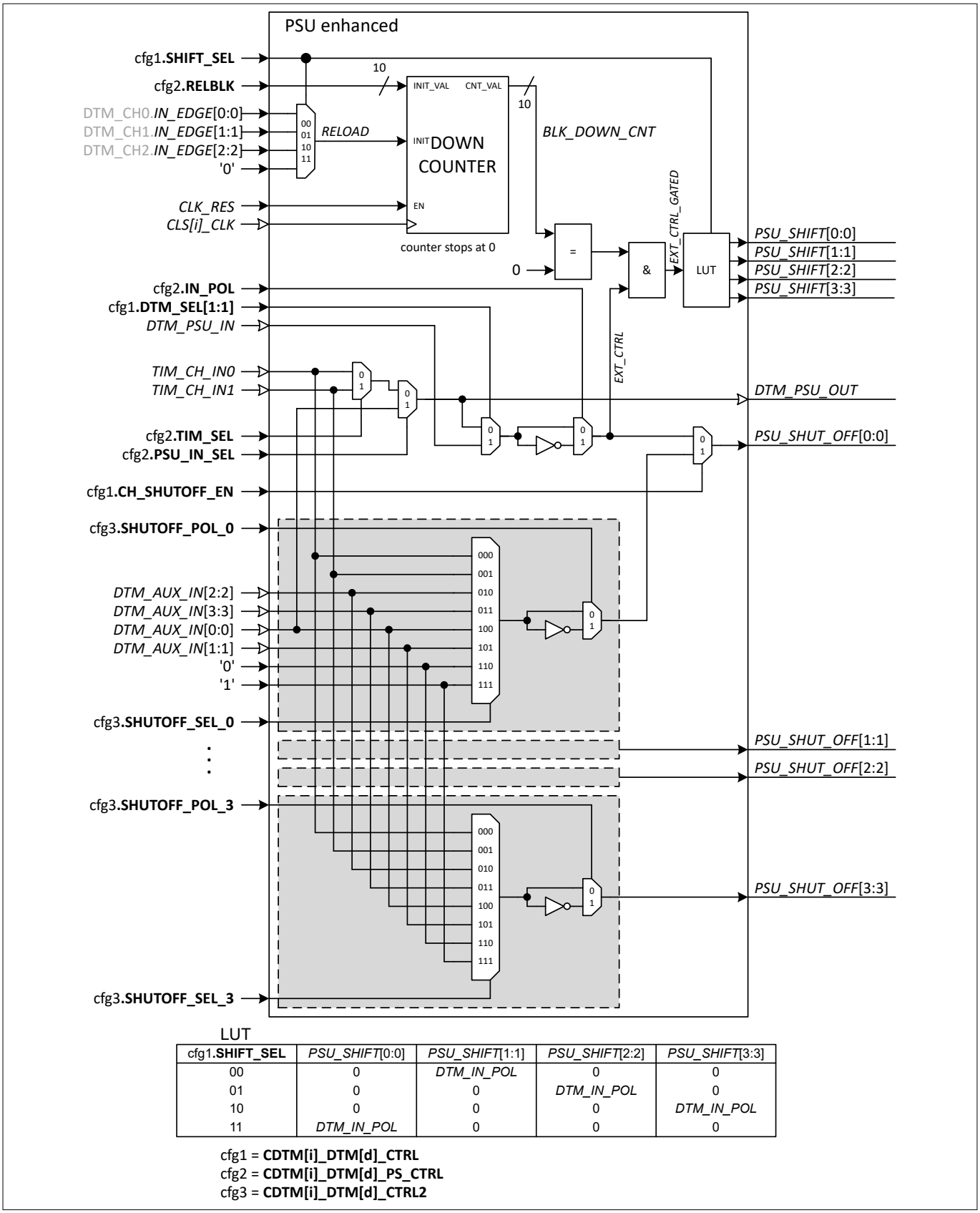
The routing from the external ports (e)GTM_CDTM[i]_DTM[d]_AUX_IN (with d being the index of the **DTM** within the **CDTM** and i being the index of the hosting cluster) to DTM_AUX_IN is summarized below (see also Figure 1021):

The following list shows the possible input sources.

- DTM_AUX_IN[0:0] is wired to the respective (e)GTM_CDTM[i]_DTM[d]_AUX_IN[0:0].
- DTM_AUX_IN[1:1] is wired to the respective (e)GTM_CDTM[i]_DTM[d]_AUX_IN[1:1].
- DTM_AUX_IN[2:2] is wired to (e)GTM_CDTM[i]_DTM[d]_AUX_IN[0:0]. If the previous **DTM** is not implemented or is placed after a module of a different type from the current one (i.e. d ∈ {4,6}), DTM_AUX_IN[2:2] is tied to '0'.
- DTM_AUX_IN[3:3] is wired to (e)GTM_CDTM[i]_DTM[d]_AUX_IN[1:1]. If the previous **DTM** is not implemented or is placed after a module of a different type from the current one (i.e. d ∈ {4,6}), DTM_AUX_IN[3:3] is tied to '0'.

***Note**: For sake of compatibility to previous **eGTM** versions, DTM_AUX_IN[0:0] has to be driven for each CDTM[i]_DTM[d] instance by the DTM_AUX_IN[0:0] primary input.*

After each source multiplexer there is an inverter. Over bit CLSi_CDTM_DTMd_CTRL2.SHUTOFF_POL_0, CLSi_CDTM_DTMd_CTRL2.SHUTOFF_POL_1, CLSi_CDTM_DTMd_CTRL2.SHUTOFF_POL_2 and CLSi_CDTM_DTMd_CTRL2.SHUTOFF_POL_3 signal PSU_SHUT_OFF[0:0], PSU_SHUT_OFF[1:1], PSU_SHUT_OFF[2:2] and PSU_SHUT_OFF[3:3] can be inverted.



LUT

cfg1.**SHIFT_SEL**	*PSU_SHIFT*[0:0]	*PSU_SHIFT*[1:1]	*PSU_SHIFT*[2:2]	*PSU_SHIFT*[3:3]
00	0	*DTM_IN_POL*	0	0
01	0	0	*DTM_IN_POL*	0
10	0	0	0	*DTM_IN_POL*
11	*DTM_IN_POL*	0	0	0

cfg1 = **CDTM[i]_DTM[d]_CTRL**
cfg2 = **CDTM[i]_DTM[d]_PS_CTRL**
cfg3 = **CDTM[i]_DTM[d]_CTRL2**

Figure 1112 PSU enhanced

The register layout of CLSi_CDTM_DTMd_CTRL2 ensures that atomic operations on multiple **DTM** channels are possible. Therefore, the register includes the bits CLSi_CDTM_DTMd_CTRL2.WR_EN_0,

CLSi_CDTM_DTMd_CTRL2.WR_EN_1, CLSi_CDTM_DTMd_CTRL2.WR_EN_2 and CLSi_CDTM_DTMd_CTRL2.WR_EN_3. The bits of CLSi_CDTM_DTMd_CTRL2 of a **DTM** channel (x∈{0...3}) can be written only if the corresponding CLSi_CDTM_DTMd_CTRL2.WR_EN_0, CLSi_CDTM_DTMd_CTRL2.WR_EN_1, CLSi_CDTM_DTMd_CTRL2.WR_EN_2 and CLSi_CDTM_DTMd_CTRL2.WR_EN_3 bits are also written to 1.

44.3.1.9.8 High-resolution PWM support

Additionally to each incoming PWM signal DTM_IN[x:x], DTM_IN_T[x:x] x:={0, 1,..., 3} another input signal is delivered from **TOM**/**ATOM** sub-module which is called DTM_IN_HRES[x], DTM_IN_T_HRES[x]. Further each channel x has an additional input signal pair DTM_IN_PREV[x:x] and DTM_IN_PREV_HRES[x], which is the input signal pair of the previous **DTM** channel. All these signals are directly connected to the related **TOM**/**ATOM** channel. Please refer to High-resolution PWM support for additional information of the high-resolution PWM support of the **eGTM**. Figure 1113 shows the high-resolution PWM related **DTM** channel.

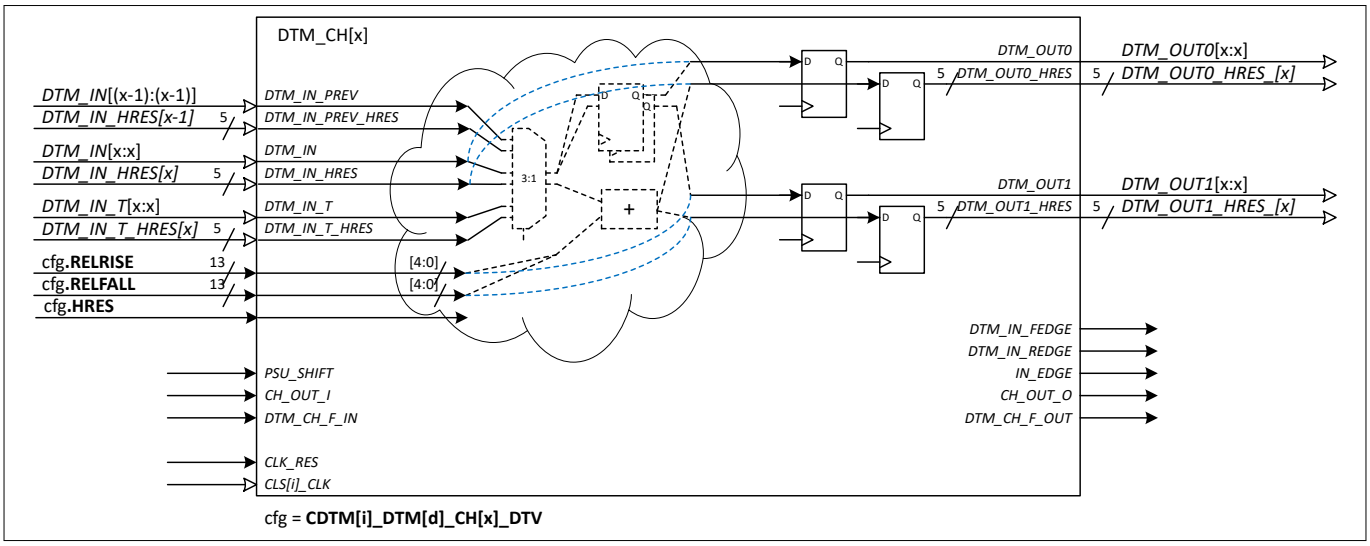


Figure 1113 DTM channel enhancement for high-resolution PWM support

- As shown in the figure above, one of the three input signals DTM_IN[3:0], DTM_IN_T[3:0] and DTM_IN_PREV[3:0] (DTM_IN_PREV is the DTM_IN signal of the previous channel [x-1]) that can be routed to the **DTM** outputs. The **DTM** module delivers two output signals DTM_OUT0[3:0] and DTM_OUT1[3:0] for the selected input signal.
- The corresponding high-resolution signals DTM_IN_HRES[x][4:0], DTM_IN_T_HRES[x][4:0] and DTM_IN_PREV_HRES[x][4:0] are routed to the outputs DTM_OUT0_HRES_[x][4:0] and DTM_OUT1_HRES_[x] [4:0] in the same manner.

Any additional delay on the path from the selected **DTM** input to the **DTM** output inside the logic cloud are also added to the high-resolution signals. Thus for each channel the DTM_OUT0_HRES_[x] and DTM_OUT1_HRES_[x] signals are switching synchronously to the DTM_OUT0[x:x], DTM_OUT1[x:x] signals respectively. Details about the routing through the logic cloud are described in the previous chapters of the **DTM**, Figure 1104 and Figure 1114.

The blue lines in the figure above show the direct path from the inputs to the outputs through the logic cloud. If this path is configured the high-resolution input values from **TOM**/**ATOM** are not modified by the **DTM** module.

The next figure shows the principal routing of the high-resolution signals only through the **DTM** channel.

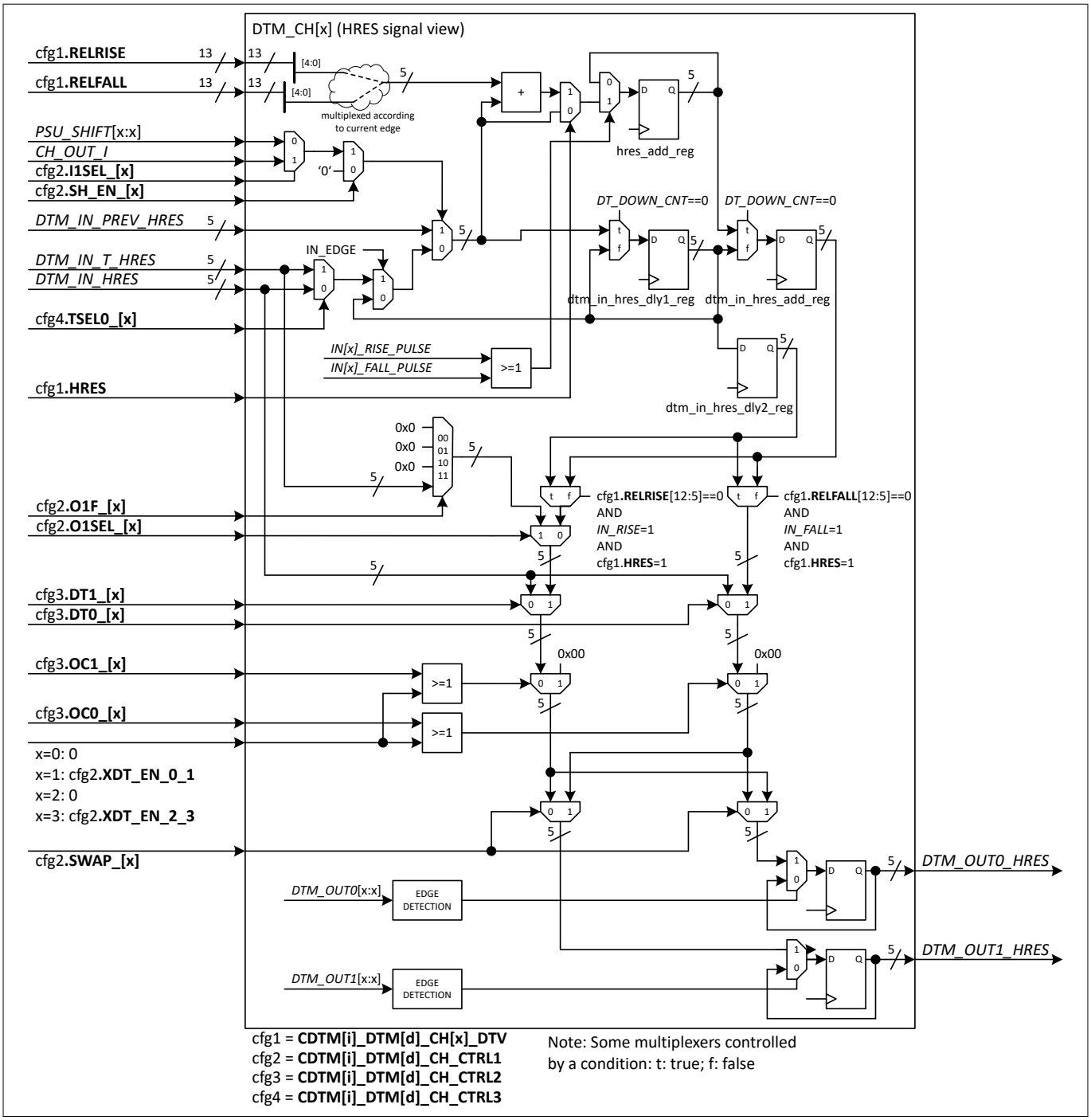


Figure 1114 Routing of high-resolution signals inside DTM channel

The output signals DTM_OUT0_HRES_[x] and DTM_OUT1_HRES_[x] are switching synchronous to the DTM_OUT0, DTM_OUT1 signals and always follows exactly the selected input signal DTM_IN_HRES[x], DTM_IN_T_HRES[x], and DTM_IN_PREV_HRES[x] with the same number of clock-cycles delay as the PWM output signals DTM_OUT0[3:0], DTM_OUT1[3:0] relating to the PWM input signal DTM_IN[3:0].

If the **DTM** output signals are swapped (CLSi_CDTM_DTMd_CH_CTRL1.SWAP_0=1, CLSi_CDTM_DTMd_CH_CTRL1.SWAP_1=1, CLSi_CDTM_DTMd_CH_CTRL1.SWAP_2=1 or CLSi_CDTM_DTMd_CH_CTRL1.SWAP_3=1), then the high-resolution output signals are also swapped.

***Note**:*

- DTM_OUT0_HRES_[x] = 0, when CLSi_CDTM_DTMd_CH_CTRL2.OC0_0 = 1/CLSi_CDTM_DTMd_CH_CTRL2.OC0_1 = 1/CLSi_CDTM_DTMd_CH_CTRL2.OC0_2 = 1/ CLSi_CDTM_DTMd_CH_CTRL2.OC0_3 = 1 and CLSi_CDTM_DTMd_CH_CTRL1.SWAP_0 = 0/CLSi_CDTM_DTMd_CH_CTRL1.SWAP_1 = 0/CLSi_CDTM_DTMd_CH_CTRL1.SWAP_2 = 0/ CLSi_CDTM_DTMd_CH_CTRL1.SWAP_3 = 0
- DTM_OUT1_HRES_[x] = 0, when CLSi_CDTM_DTMd_CH_CTRL2.OC1_0 = 1/CLSi_CDTM_DTMd_CH_CTRL2.OC1_1 = 1/CLSi_CDTM_DTMd_CH_CTRL2.OC1_2 = 1/ CLSi_CDTM_DTMd_CH_CTRL2.OC1_3 = 1 = 1 and CLSi_CDTM_DTMd_CH_CTRL1.SWAP_0 = 0/CLSi_CDTM_DTMd_CH_CTRL1.SWAP_1 = 0/CLSi_CDTM_DTMd_CH_CTRL1.SWAP_2 = 0/ CLSi_CDTM_DTMd_CH_CTRL1.SWAP_3 = 0
- DTM_OUT1_HRES_[x] = 0, when CLSi_CDTM_DTMd_CH_CTRL1.O1SEL_0 = 1/ CLSi_CDTM_DTMd_CH_CTRL1.O1SEL_1 = 1/CLSi_CDTM_DTMd_CH_CTRL1.O1SEL_2 = 1/ CLSi_CDTM_DTMd_CH_CTRL1.O1SEL_3 = 1 and CLSi_CDTM_DTMd_CH_CTRL1.O1F_0 != 11_B/CLSi_CDTM_DTMd_CH_CTRL1.O1F_1 != 11_B/CLSi_CDTM_DTMd_CH_CTRL1.O1F_2 != 11_B/ CLSi_CDTM_DTMd_CH_CTRL1.O1F_3 != 11_B and CLSi_CDTM_DTMd_CH_CTRL1.SWAP_0 = 0/CLSi_CDTM_DTMd_CH_CTRL1.SWAP_1 = 0/CLSi_CDTM_DTMd_CH_CTRL1.SWAP_2 = 0/ CLSi_CDTM_DTMd_CH_CTRL1.SWAP_3 = 0
- For the above three statements, CLSi_CDTM_DTMd_CH_CTRL1.SWAP_0 = 0/CLSi_CDTM_DTMd_CH_CTRL1.SWAP_1 = 0/CLSi_CDTM_DTMd_CH_CTRL1.SWAP_2 = 0/ CLSi_CDTM_DTMd_CH_CTRL1.SWAP_3 = 0 is assumed. Please see description above for CLSi_CDTM_DTMd_CH_CTRL1.SWAP_0 = 1/CLSi_CDTM_DTMd_CH_CTRL1.SWAP_1 = 1/ CLSi_CDTM_DTMd_CH_CTRL1.SWAP_2 = 1/CLSi_CDTM_DTMd_CH_CTRL1.SWAP_3 = 1.

***Note**:* *The high-resolution input signal changes its value exactly (!! and only !!) with its dedicated input signal edge. If there are no edges on the PWM input signal (DTM_IN/DTM_IN_T), then there are also no changes on the dedicated high-resolution input signal. This depends only on the signal generation in the (A)**TOM** modules.*

***Note**:* *The high-resolution output signals DTM_OUT0_HRES_[x] and DTM_OUT1_HRES_[x] will change only, if there is an edge on the associated output signal DTM_OUT0[x:x], DTM_OUT1[x:x].*

But this is only true, if the output edge occurs due to an input edge or due to a change of the configuration bits DT0/DT1 for standard dead-time. This configuration change will also lead to an edge on one of the output signals.

- If CLSi_CDTM_DTMd_CH_CTRL2.OC0_0=1 and CLSi_CDTM_DTMd_CH_CTRL2.OC1_0=1 (and CLSi_CDTM_DTMd_CH_CTRL1.SWAP_0=0) then the signal level bits CLSi_CDTM_DTMd_CH_CTRL2.SL0_0 and CLSi_CDTM_DTMd_CH_CTRL2.SL1_0 define the values of DTM_OUT0[0:0] and DTM_OUT1[0:0] and the high-resolution signals DTM_OUT0_HRES_[x] and DTM_OUT1_HRES_[x] are set to 0_H. Thus the activation of the SL path can lead to a change of DTM_OUT0_HRES_[x] and DTM_OUT1_HRES_[x] from its current value to 0_H. After activation of the SL path a configuration change of the signal level bits CLSi_CDTM_DTMd_CH_CTRL2.SL0_0, CLSi_CDTM_DTMd_CH_CTRL2.SL1_0 will lead to an edge on the output signals DTM_OUT0[0:0] and DTM_OUT1[0:0], but not to an update of the high-resolution output signals DTM_OUT0_HRES_[x] and DTM_OUT1_HRES_[x].
- If CLSi_CDTM_DTMd_CH_CTRL2.OC0_1=1 and CLSi_CDTM_DTMd_CH_CTRL2.OC1_1=1 (and CLSi_CDTM_DTMd_CH_CTRL1.SWAP_1=0) then the signal level bits CLSi_CDTM_DTMd_CH_CTRL2.SL0_1 and CLSi_CDTM_DTMd_CH_CTRL2.SL1_1 define the values of DTM_OUT0[1:1] and DTM_OUT1[1:1] and the high-resolution signals DTM_OUT0_HRES_[x] and DTM_OUT1_HRES_[x] are set to 0_H. Thus the activation of the SL path can lead to a change of DTM_OUT0_HRES_[x] and DTM_OUT1_HRES_[x] from its current value to 0_H. After activation of the SL path a configuration change of the signal level bits CLSi_CDTM_DTMd_CH_CTRL2.SL0_1, CLSi_CDTM_DTMd_CH_CTRL2.SL1_1 will lead to an edge on the

output signals DTM_OUT0[1:1] and DTM_OUT1[1:1], but not to an update of the high-resolution output signals DTM_OUT0_HRES_[x] and DTM_OUT1_HRES_[x].

- If CLSi_CDTM_DTMd_CH_CTRL2.OC0_2=1 and CLSi_CDTM_DTMd_CH_CTRL2.OC1_2=1 (and CLSi_CDTM_DTMd_CH_CTRL1.SWAP_2=0) then the signal level bits CLSi_CDTM_DTMd_CH_CTRL2.SL0_2 and CLSi_CDTM_DTMd_CH_CTRL2.SL1_2 define the values of DTM_OUT0[2:2] and DTM_OUT1[2:2] and the high-resolution signals DTM_OUT0_HRES_[x] and DTM_OUT1_HRES_[x] are set to 0_H. Thus the activation of the SL path can lead to a change of DTM_OUT0_HRES_[x] and DTM_OUT1_HRES_[x] from its current value to 0_H. After activation of the SL path a configuration change of the signal level bits CLSi_CDTM_DTMd_CH_CTRL2.SL0_2, CLSi_CDTM_DTMd_CH_CTRL2.SL1_2 will lead to an edge on the output signals DTM_OUT0[2:2] and DTM_OUT1[2:2], but not to an update of the high-resolution output signals DTM_OUT0_HRES_[x] and DTM_OUT1_HRES_[x].
- If CLSi_CDTM_DTMd_CH_CTRL2.OC0_3=1 and CLSi_CDTM_DTMd_CH_CTRL2.OC1_3=1 (and CLSi_CDTM_DTMd_CH_CTRL1.SWAP_3=0) then the signal level bits CLSi_CDTM_DTMd_CH_CTRL2.SL0_3 and CLSi_CDTM_DTMd_CH_CTRL2.SL1_3 define the values of DTM_OUT0[3:3] and DTM_OUT1[3:3] and the high-resolution signals DTM_OUT0_HRES_[x] and DTM_OUT1_HRES_[x] are set to 0_H. Thus the activation of the SL path can lead to a change of DTM_OUT0_HRES_[x] and DTM_OUT1_HRES_[x] from its current value to 0_H. After activation of the SL path a configuration change of the signal level bits CLSi_CDTM_DTMd_CH_CTRL2.SL0_3, CLSi_CDTM_DTMd_CH_CTRL2.SL1_3 will lead to an edge on the output signals DTM_OUT0[3:3] and DTM_OUT1[3:3], but not to an update of the high-resolution output signals DTM_OUT0_HRES_[x] and DTM_OUT1_HRES_[x].

If the configuration bit CLSi_CDTM_DTMd_CH_DTVx.HRES = 1, it is possible to insert dead-time between the two inverted output signals DTM_OUT0[3:0] and DTM_OUT1[3:0] on a micro-step level by calculating the output signals DTM_OUT0_HRES_[x] and DTM_OUT1_HRES_[x] in the following manner.

Figure 1115 shows the principle schematic view of the calculation of the output values of DTM_OUT0_HRES_[x] / DTM_OUT1_HRES_[x].

As shown in figure, the input value of DTM_IN_HRES[x][4:0] is added to CLSi_CDTM_DTMd_CH_DTVx.RELRISE[12:0] / CLSi_CDTM_DTMd_CH_DTVx.RELFALL[12:0] value. The output signals DTM_OUT0_HRES_[x] and DTM_OUT1_HRES_[x] are the bits [4:0] of the result. The counter DT_DOWN_CNT gets the bits [12:5] of the result and the DT_DOWN_CNT result (counter output signal is DT_DOWN_CNT) includes the value of the carry bit or adding the lower bits [4:0].

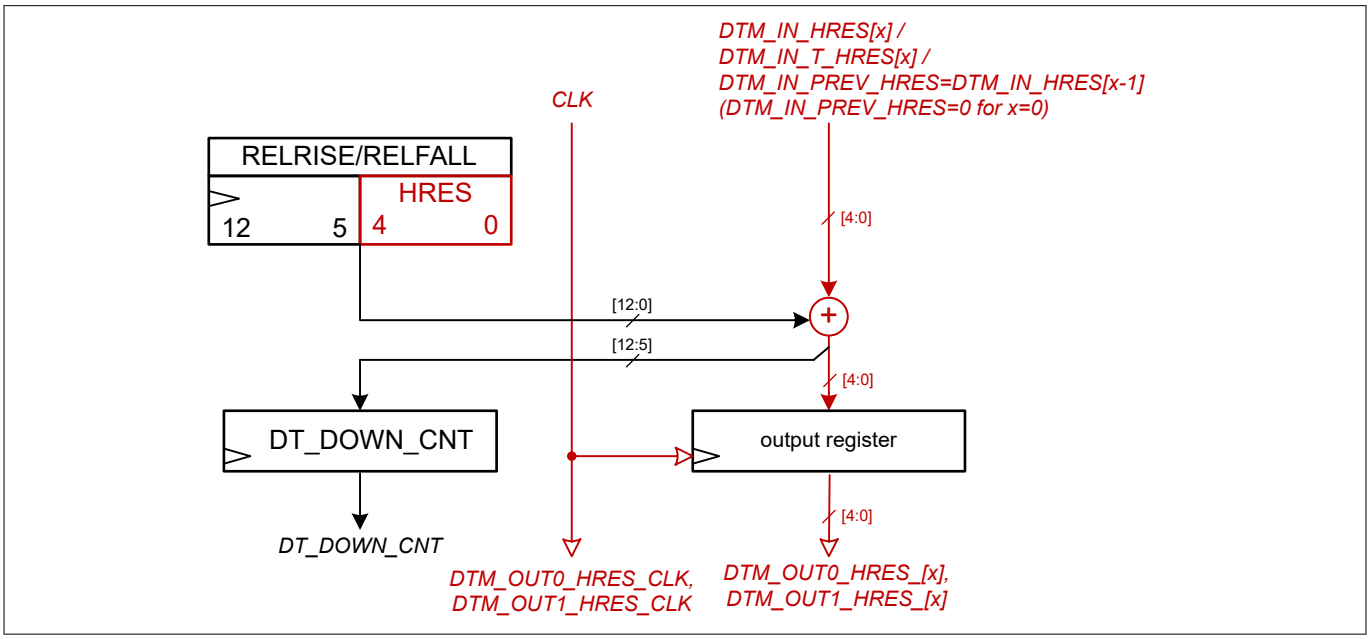


Figure 1115 Calculation of HRES output signal

***Note**: In the figure above all three possible input signals are shown, but only one of them is selected corresponding to the selected **DTM** input signal (DTM_IN_PREV, DTM_IN, DTM_IN_T).*

It has to be considered that the maximum range of dead-time insertion is reduced to 2^8 clock-cycles when high-resolution support is switched on in **DTM**. Without high-resolution support the maximum range is 2^{13} clock-cycles based on **DTM** internal clock resolution, see Figure 1100.

***Note**: The clock-cycles or clock frequency depends on the **DTM** internal clock resolution CLK_RES; see Figure 1100.*

The signal diagram in Figure 1116 shows an example of how the dead-time adjustment on micro-step level works. The red and blue delays are realized by customer logic outside of the **DTM**/**eGTM**. The red dotted lines in Figure 1116 are the PWM signals after the external delay chain.

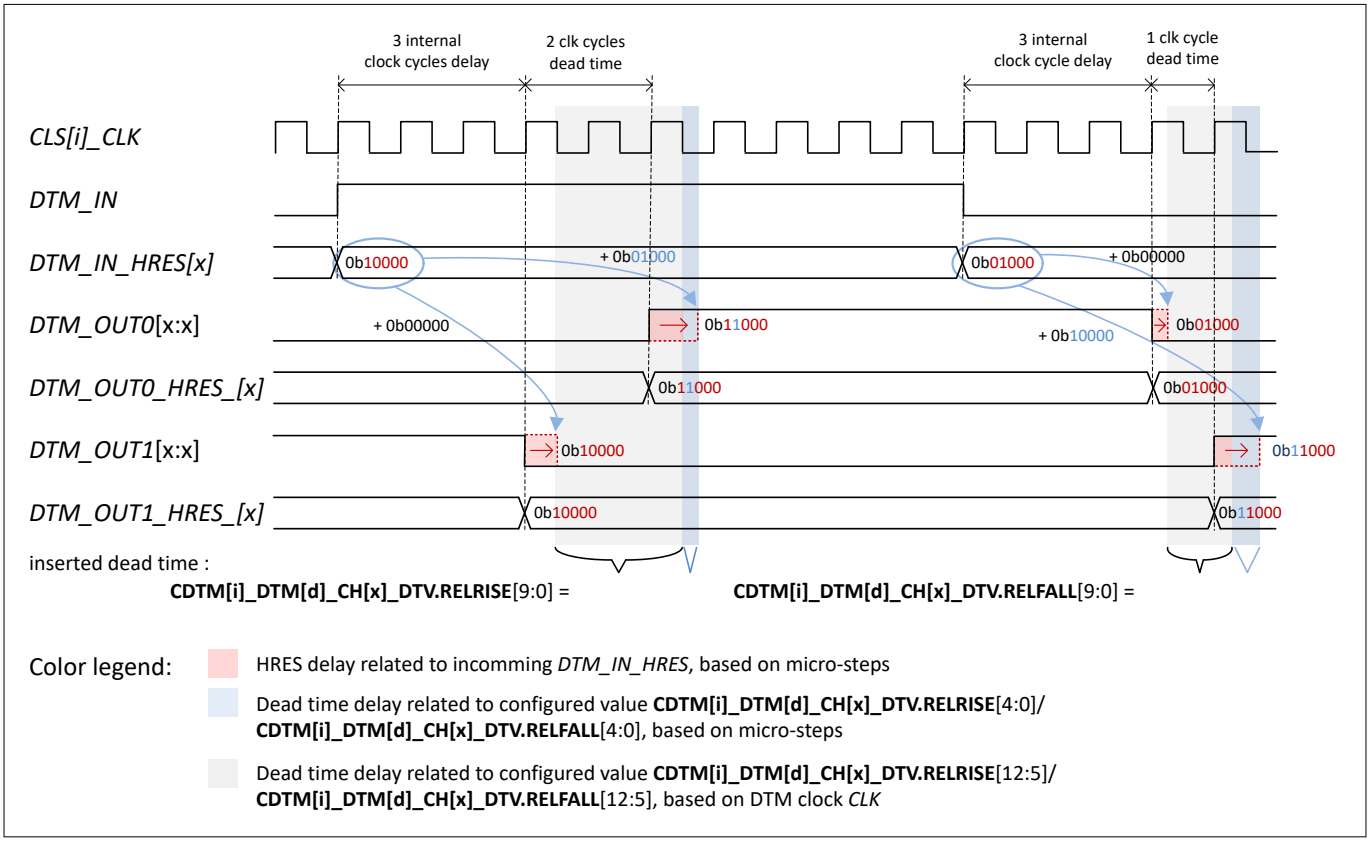


Figure 1116 Standard dead-time adjustment on micro-step level

In Figure 1116 it can be seen that for the rising edge of input signal DTM_IN[0:0] the value of the output signal DTM_OUT1_HRES_[x] is the same as the value of input signal DTM_IN_HRES[x] whereas the value of DTM_OUT0_HRES_[x] is calculated by adding the value of bit field CLSi_CDTM_DTMd_CH_DTVx.RELRISE[4:0] to DTM_IN_HRES[x].

For the falling edge of input signal DTM_IN[0:0] the same is done in opposite way whereas the value of the output signal DTM_OUT0_HRES_[x] has the same value as of the input signal DTM_IN_HRES[x] and the value of DTM_OUT1_HRES_[x] is calculated by adding the value of bit field CLSi_CDTM_DTMd_CH_DTVx.RELFALL[4:0] to DTM_IN_HRES[x].

The Figure 1116 shows the input signals DTM_IN[x:x] and DTM_IN_HRES[x][4:0], but the same applies if the input signals DTM_IN_T[x:x], DTM_IN_T_HRES[x][4:0] or DTM_IN_PREV[x:x], DTM_IN_PREV_HRES[x][4:0] are selected.

The selection of the **DTM** input signals DTM_IN, DTM_IN_T and DTM_IN_PREV are shown in Figure 1104. The selection of DTM_IN or DTM_IN_T is controlled by CLSi_CDTM_DTMd_CH_CTRL3.TSEL0_0,

CLSi_CDTM_DTMd_CH_CTRL3.TSEL0_1, CLSi_CDTM_DTMd_CH_CTRL3.TSEL0_2, CLSi_CDTM_DTMd_CH_CTRL3.TSEL0_3. To select DTM_IN_PREV the signals PSU_SHIFT or CH_OUT_I and the bit fields CLSi_CDTM_DTMd_CH_CTRL1.I1SEL_0, CLSi_CDTM_DTMd_CH_CTRL1.I1SEL_1, CLSi_CDTM_DTMd_CH_CTRL1.I1SEL_2, CLSi_CDTM_DTMd_CH_CTRL1.I1SEL_3 and CLSi_CDTM_DTMd_CH_CTRL1.SH_EN_1, CLSi_CDTM_DTMd_CH_CTRL1.SH_EN_2, CLSi_CDTM_DTMd_CH_CTRL1.SH_EN_3 have to be considered. Accordingly, the selection of the corresponding high-resolution input is done automatically.

The high-resolution mode in **DTM** is only supported for standard dead-time (as described before) and for the phase shift mode, where standard dead-time is used.

In all other **DTM** modes (cross channel dead-time, multiple output signal combination, pulse generation on edge) the high-resolution PWM is not supported. In this case, the output signals DTM_OUT0_HRES_[x], DTM_OUT1_HRES_[x] are overwritten to zero. Setting the configuration bit CLSi_CDTM_DTMd_CH_DTVx.HRES has no effect in this **DTM** mode.

Table 2250 High-resolution PWM support depending on DTM modes

DTM mode	high-resolution PWM support	comment
standard dead-time	yes	DTM_OUT0_HRES_[x] and DTM_OUT1_HRES_[x] output calculated
phase shift mode	yes	DTM_OUT0_HRES_[x] and DTM_OUT1_HRES_[x] output calculated
cross channel dead-time	no	DTM_OUT0_HRES_[x] and DTM_OUT1_HRES_[x] outputs overwritten to zero
multiple output signal combination	no	high-resolution outputs connected to corresponding high-resolution input (see Special configurations, paths or behavior, too)
pulse generation on edge	no	high-resolution outputs connected to corresponding high-resolution input

Special configurations, paths or behavior

If special function on output 1 (DTM_OUT1) is selected (CLSi_CDTM_DTMd_CH_CTRL1.O1SEL_0=1 and CLSi_CDTM_DTMd_CH_CTRL2.DT1_0=1), the following high-resolution input is connected to high-resolution output depending on CLSi_CDTM_DTMd_CH_CTRL1.O1F_0:

- CLSi_CDTM_DTMd_CH_CTRL1.O1F_0=00_B: 00_H
- CLSi_CDTM_DTMd_CH_CTRL1.O1F_0=01_B: 00_H
- CLSi_CDTM_DTMd_CH_CTRL1.O1F_0=10_B: 00_H
- CLSi_CDTM_DTMd_CH_CTRL1.O1F_0=11_B: DTM_IN_T_HRES[x]

If the **DTM** output signals are swapped (CLSi_CDTM_DTMd_CH_CTRL1.SWAP_0=1), then the high-resolution output signals are also swapped.

Further, there is an exception if CLSi_CDTM_DTMd_CH_CTRL2.OC0_0=1 or CLSi_CDTM_DTMd_CH_CTRL2.OC1_0=1 then the corresponding high-resolution output value is set to 0_H.

***Note**: This ensures that the result of an active shutoff is available at an output signal as soon as possible.*

If special function on output 1 (DTM_OUT1) is selected (CLSi_CDTM_DTMd_CH_CTRL1.O1SEL_1=1 and CLSi_CDTM_DTMd_CH_CTRL2.DT1_1=1), the following high-resolution input is connected to high-resolution output depending on CLSi_CDTM_DTMd_CH_CTRL1.O1F_1:

- CLSi_CDTM_DTMd_CH_CTRL1.O1F_1=00_B: 00_H
- CLSi_CDTM_DTMd_CH_CTRL1.O1F_1=01_B: 00_H
- CLSi_CDTM_DTMd_CH_CTRL1.O1F_1=10_B: 00_H
- CLSi_CDTM_DTMd_CH_CTRL1.O1F_1=11_B: DTM_IN_T_HRES[x]

If the **DTM** output signals are swapped (CLSi_CDTM_DTMd_CH_CTRL1.SWAP_1=0), then the high-resolution output signals are also swapped.

Further, there is an exception if CLSi_CDTM_DTMd_CH_CTRL2.OC0_1=1 or CLSi_CDTM_DTMd_CH_CTRL2.OC1_1=1 then the corresponding high-resolution output value is set to 0_H.

***Note**: This ensures that the result of an active shutoff is available at an output signal as soon as possible.*

If special function on output 1 (DTM_OUT1) is selected (CLSi_CDTM_DTMd_CH_CTRL1.O1SEL_2=1 and CLSi_CDTM_DTMd_CH_CTRL2.DT1_2=1), the following high-resolution input is connected to high-resolution output depending on CLSi_CDTM_DTMd_CH_CTRL1.O1F_2:

- CLSi_CDTM_DTMd_CH_CTRL1.O1F_2=00_B: 00_H
- CLSi_CDTM_DTMd_CH_CTRL1.O1F_2=01_B: 00_H
- CLSi_CDTM_DTMd_CH_CTRL1.O1F_2=10_B: 00_H
- CLSi_CDTM_DTMd_CH_CTRL1.O1F_2=11_B: DTM_IN_T_HRES[x]

If the **DTM** output signals are swapped (CLSi_CDTM_DTMd_CH_CTRL1.SWAP_2=0, then the high-resolution output signals are also swapped.

Further, there is an exception if CLSi_CDTM_DTMd_CH_CTRL2.OC0_2=1 or CLSi_CDTM_DTMd_CH_CTRL2.OC1_2=1 then the corresponding high-resolution output value is set to 0_H.

***Note**: This ensures that the result of an active shutoff is available at an output signal as soon as possible.*

If special function on output 1 (DTM_OUT1) is selected (CLSi_CDTM_DTMd_CH_CTRL1.O1SEL_3=1 and CLSi_CDTM_DTMd_CH_CTRL2.DT1_3=1), the following high-resolution input is connected to high-resolution output depending on CLSi_CDTM_DTMd_CH_CTRL1.O1F_3:

- CLSi_CDTM_DTMd_CH_CTRL1.O1F_3=00_B: 00_H
- CLSi_CDTM_DTMd_CH_CTRL1.O1F_3=01_B: 00_H
- CLSi_CDTM_DTMd_CH_CTRL1.O1F_3=10_B: 00_H
- CLSi_CDTM_DTMd_CH_CTRL1.O1F_3=11_B: DTM_IN_T_HRES[x]

If the **DTM** output signals are swapped (CLSi_CDTM_DTMd_CH_CTRL1.SWAP_3=0), then the high-resolution output signals are also swapped.

Further, there is an exception if CLSi_CDTM_DTMd_CH_CTRL2.OC0_3=1 or CLSi_CDTM_DTMd_CH_CTRL2.OC1_3=1 then the corresponding high-resolution output value is set to 0_H.

***Note**: This ensures that the result of an active shutoff is available at an output signal as soon as possible.*

44.3.1.9.9 DTM connections on eGTM toplevel

If **DTM** instance is present, it is placed behind the outputs of a **TOM**, **ATOM** depending on the **eGTM** device configuration.

In case **DTM** exists, the outputs TOM_OUT[y:y], ATOM_OUT[y:y] and TOM_OUT_T[y:y], ATOM_OUT_T[y:y] are connected to **DTM** inputs DTM_IN and DTM_IN_T. The outputs of the **DTM** are routed directly to the toplevel of **eGTM**.

The behavior of **DTM** after reset is shown in Figure 1117.

***Note**: If a device configuration is used without high-resolution support (device configuration parameter DTM_HIGH_RES_G=0) then all high-resolution outputs of **DTM** (DTM_OUT0_HRES_[x] and DTM_OUT1_HRES_[x]) are zero.*

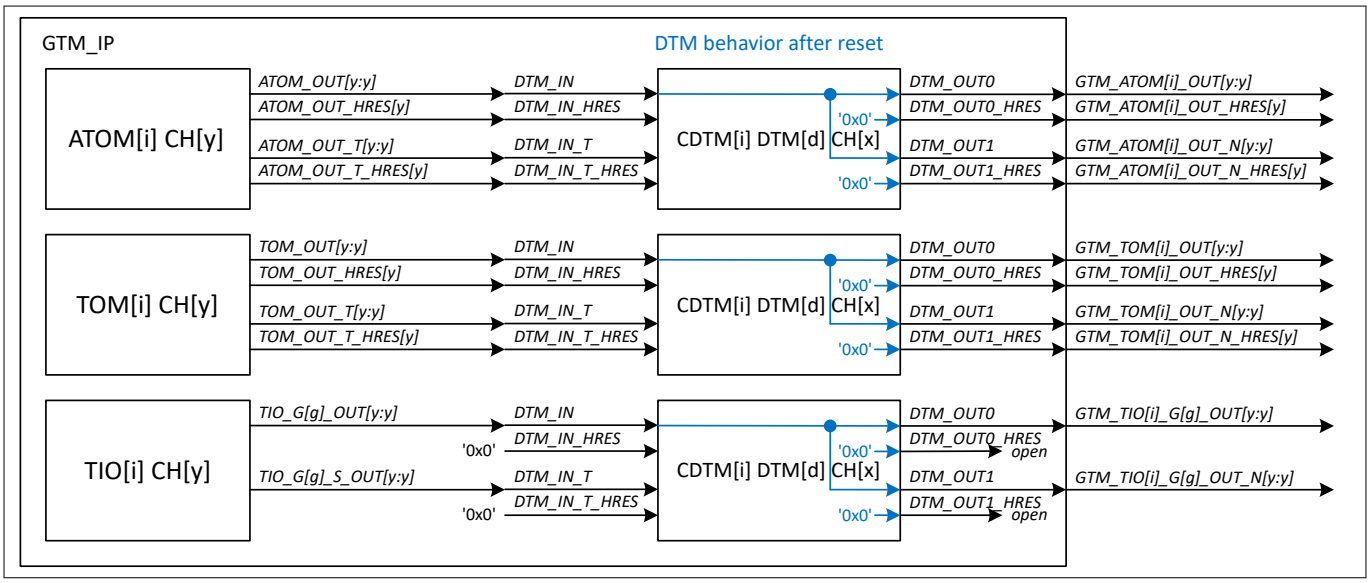


Figure 1117 DTM behavior after reset

***Note**: The figure is a general picture from the IP vendor. The exact configuration can be found in the device specific configuration.*

- To enable the second pass through route of signal DTM_IN_T[x:x] to the **DTM** output DTM_OUT1[x:x], the following **DTM** channel configuration has to be chosen:
- CLSi_CDTM_DTMd_CH_CTRL1.O1F_0 = 11_B, CLSi_CDTM_DTMd_CH_CTRL1.O1SEL_0=1 and CLSi_CDTM_DTMd_CH_CTRL2.DT1_0=1.
- CLSi_CDTM_DTMd_CH_CTRL1.O1F_1 = 11_B, CLSi_CDTM_DTMd_CH_CTRL1.O1SEL_1=1 and CLSi_CDTM_DTMd_CH_CTRL2.DT1_1=1.
- CLSi_CDTM_DTMd_CH_CTRL1.O1F_2 = 11_B, CLSi_CDTM_DTMd_CH_CTRL1.O1SEL_2=1 and CLSi_CDTM_DTMd_CH_CTRL2.DT1_2=1.
- CLSi_CDTM_DTMd_CH_CTRL1.O1F_3 = 11_B, CLSi_CDTM_DTMd_CH_CTRL1.O1SEL_3=1 and CLSi_CDTM_DTMd_CH_CTRL2.DT1_3=1.

The signal names and the signal routing in the case of no **DTM** instance placed behind a **TOM**, **ATOM**, is shown in Figure 1118.

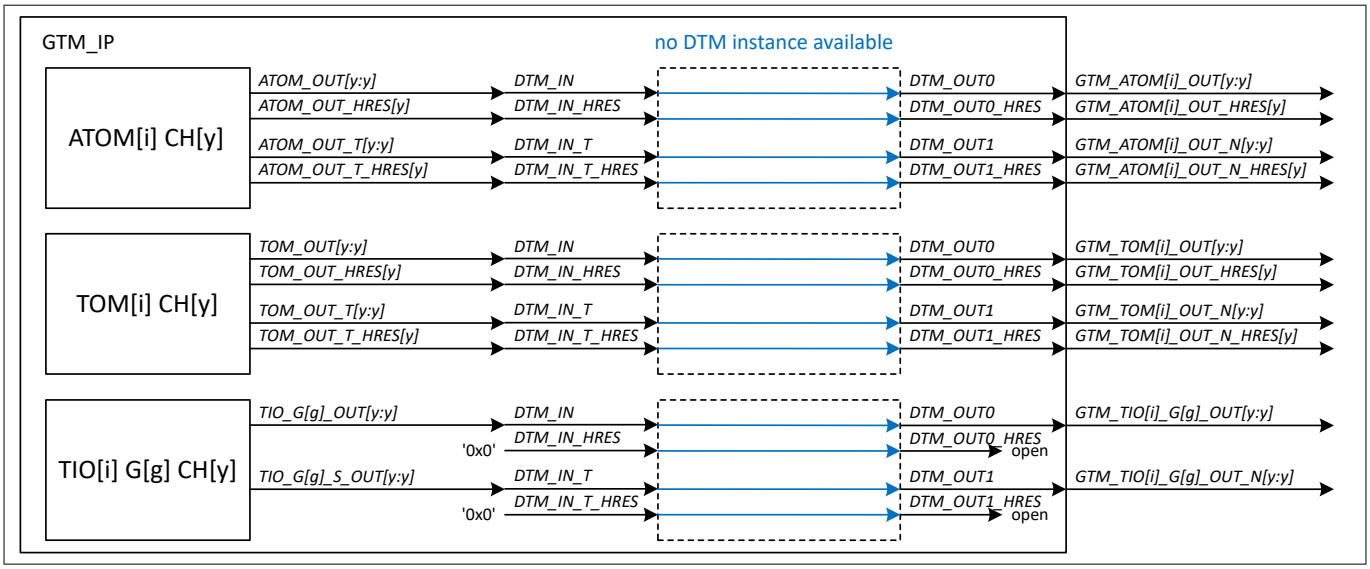


Figure 1118 (A)TOM output signal routing in case of no DTM instance available

***Note**: The figures above Figure 1117 and Figure 1118 show only the routing without implemented register stages.*

***Note**: The figure is a general picture from the IP vendor. The exact configuration can be found in the device specific configuration.*

44.3.1.9.10 eGTM DTM Configuration Registers Description

Sub-chapter DTM Configuration Registers Description

Table 2251

Register Shortname	Description
CLSi_CDTM_DTMd_CTRL	CDTM[i]_DTM[d] global configuration and control register (d=0-3;i=0-2)
CLSi_CDTM_DTMd_CH_CTRL1	CDTM[i]_DTM[d] channel control register 1 (d=0,2,4;i=0-2)
CLSi_CDTM_DTMd_CH_CTRL1	CDTM[i]_DTM[d] channel control register 1 (d=1,3,5;i=0-2)
CLSi_CDTM_DTMd_CH_CTRL2	CDTM[i]_DTM[d] channel control register 2 (d=0-5;i=0-2)
CLSi_CDTM_DTMd_CH_CTRL2_SR	CDTM[i] DTM[d] channel control register 2 shadow (d=0-5;i=0-2)
CLSi_CDTM_DTMd_CH_CTRL3	CDTM[i]_DTM[d] channel control register 3 (d=0-5;i=0-2)
CLSi_CDTM_DTMd_PS_CTRL	CDTM[i]_DTM[d] phase shift unit configuration and control register (d=0-5;i=0-2)
CLSi_CDTM_DTMd_CH_DTVx	CDTM[i]_DTM[d] channel [x] dead-time reload values (x=0-3;i=0-2;d=0-3)

(table continues...)

Table 2251 (continued)

Register Shortname	Description
CLSi_CDTM_DTMd_CH_DTVx	CDTM[i]_DTM[d] channel [x] dead-time reload values (x=0-3;i=0-2;d=4-5)
CLSi_CDTM_DTMd_CH_SR	CDTM[i]_DTM[d] channel shadow register (d=0-5;i=0-2)
CLSi_CDTM_DTMd_CTRL2	CDTM[i]_DTM[d] global configuration and control register 2 (d=0-5;i=0-2)
CLSi_CDTM_DTMd_CH_DTV_SRx	CDTM[i]_DTM[d] channel [x] dead-time shadow values (x=0-3;i=0-2;d=0-5)

44.3.1.9.11 DTM signal description

Table 2252 DTM Signals

Name	Description
CLK_RES	clock resolution signal in use in **DTM** module
COUT	signal used for multiple output signal combination feature; bit x of signal vector is output of channel x to be able to combine it with further signals of channel x+1
CH_OUT_I	signal used for multiple output signal combination feature; input signal off a **DTM** channel
CH_OUT_O	signal used for multiple output signal combination feature; output signal off a **DTM** channel
PSU_SHIFT	signal of phase shift unit to be used inside **DTM** channels; signal PSU_SHIFT[x:x] used by corresponding channel [x]
IN_EDGE	signal of each channel [x] to trigger a reload of counter BLK_DOWN_CNT or trigger an update of a register from a shadow register
DTM_IN_PREV	**DTM** channel input signal from a previous channel
DTM_IN_PREV_HRES[x]	**DTM** channel dtm_in high-resolution signal from a previous channel
DTM_IN_FEDGE	signal of each channel [x]; only for channel [0] the signal is routed to DTM_IN0_FEDGE
DTM_IN_REDGE	signal of each channel [x]; only for channel [0] the signal is routed to DTM_IN0_REDGE
DTM_CH_F_IN	input signal of channel [x] used with cross channel dead-time feature
DTM_CH_F_OUT	output signal of channel [x] used with cross channel dead-time feature
DTM_IN_HRES	**DTM** channel dtm_in high-resolution signal from (A)**TOM**

(table continues...)

Table 2252 (continued) DTM Signals

Name	Description
IN[x]_SIG	signal of channel [x] after the first two multiplexers on which the edge detection for signal IN_EDGE is done
EDGE_TRIGG_[x]	signal of channel [x] which shows if the selected input source has an edge
CII[x]_MUX	signal of channel [x] which is the multiplexer output which is in front of the **XOR** and **AND** gates for the combinational feature of the channel
IN[x]_DLY1	selected input source signal of channel [x] one clock-cycle delayed; used for edge detection
IN[x]_DLY2	selected input source signal of channel [x] two clock-cycle delayed; used for edge detection
IN[x]_RISE_PULSE	signal of **DTM** channel [x] that a rising edge at the input occurs
IN[x]_FALL_PULSE	signal of **DTM** channel [x] that a falling edge at the input occurs
IN[x]_RISE	signal of **DTM** channel after a dead-time insertion for a rising edge
IN[x]_FALL	signal of **DTM** channel after a dead-time insertion for a falling edge
DTM_AUX_IN0	signal used for feature phase shift or to be (N)**AND**/(N)**OR**/ X(N)**OR** combined with further channel signals
DTM_AUX_IN1	signal used for feature phase shift or to be (N)**AND**/(N)**OR**/ X(N)**OR** combined with further channel signals
PSU_SHUT_OFF	output signal of phase shift unit used for feature shutoff of channel x
SHUTOFF_SYNC_0	signal used by individual channel shutoff feature of channel 0
SHUTOFF_SYNC_1	signal used by individual channel shutoff feature of channel 1
SHUTOFF_SYNC_2	signal used by individual channel shutoff feature of channel 2
SHUTOFF_SYNC_3	signal used by individual channel shutoff feature of channel 3
BLK_DOWN_CNT	output signal of blanking window counter DOWN COUNTER
DT_DOWN_CNT	output signal of dead-time counter DT_DOWN_CNT

44.3.1.9.12 DTM port description

Table 2253 DTM AEI Infrastructure Interface

Name	Description
AEI_DTM_SEL	**AEI** module select
DTM_AEI_W1R0	**AEI** address
DTM_AEI_ADDR	**AEI** access: write = 1, read = 0

(table continues...)

Table 2253 (continued) DTM AEI Infrastructure Interface

Name	Description
DTM_AEI_WDATA	**AEI** write data
DTM_AEI_RDATA	**AEI** read data
DTM_AEI_READY	**AEI** ready: access completed
DTM_AEI_STATUS	**AEI** status

Table 2254 DTM Signal Interface

Name	Description
DTM_IN	**DTM** input signal from (A)**TOM**
DTM_IN_T	**DTM** input signal from (A)**TOM**
TIM_CH_IN0	**DTM** input signal from **TIM**
TIM_CH_IN1	**DTM** input signal from **TIM**
DTM_AUX_IN	**DTM** input signal from external
DTM_OUT0	**DTM** output 0 signal
DTM_OUT1	**DTM** output 1 signal

Table 2255 DTM Signal HRES Interface

Name	Description
DTM_IN_HRES[x]	**DTM** channel x DTM_IN high-resolution signal from (A)**TOM**
DTM_IN_T_HRES[x]	**DTM** channel x DTM_IN_T high-resolution signal from (A)**TOM**
DTM_OUT0_HRES_CLK	**DTM** DTM_OUT0 high-resolution output high-resolution clock signals
DTM_OUT0_HRES_[x]	**DTM** channel [x] DTM_OUT0 high-resolution output signal
DTM_OUT1_HRES_CLK	**DTM** DTM_OUT1 high-resolution output high-resolution clock signals
DTM_OUT1_HRES_[x]	**DTM** channel x DTM_OUT1 high-resolution output signal

Table 2256 DTM / DTM Signal Interface

Name	Description
DTM_COUT3_I	Signal between **DTM** instances; driven by previous instance
DTM_COUT3_O	Signal between **DTM** instances; used by following instance
DTM_PSU_IN	Signal between **DTM** instances; driven by previous instance
DTM_PSU_OUT	Signal between **DTM** instances; used by following instance
DTM_PREV_IN0_REDGE	Signal chain through all **DTM** instances; driven by previous instance
DTM_PREV_IN0_FEDGE	Signal chain through all **DTM** instances; driven by previous instance

(table continues...)

Table 2256 (continued) DTM / DTM Signal Interface

Name	Description
DTM_IN0_REDGE	Signal chain through all **DTM** instances; used by following instance
DTM_IN0_FEDGE	Signal chain through all **DTM** instances; used by following instance

44.3.1.10 Sensor pattern evaluation (SPE)

44.3.1.10.1 Overview

The Sensor Pattern Evaluation (**SPE**) sub-module can be used to evaluate three hall sensor inputs and together with the **TOM** module to support the drive of BLDC engines. Thus, the input signals are filtered already in the connected **TIM** channels. The integration of the **SPE** sub-module into the overall **eGTM** architecture concept is shown in Figure 1119.

Indices and their range as used inside this chapter are:

- i:={0, 1,..., NSPE-1} instance index of cluster / **SPE** module
- x, y, z channel index (x !=y!=z)
- t:={0, 1,..., 7} index of input pattern
- p:={0, 1,..., 7} index for pattern
- o:={0, 1,..., 7} index for output pattern

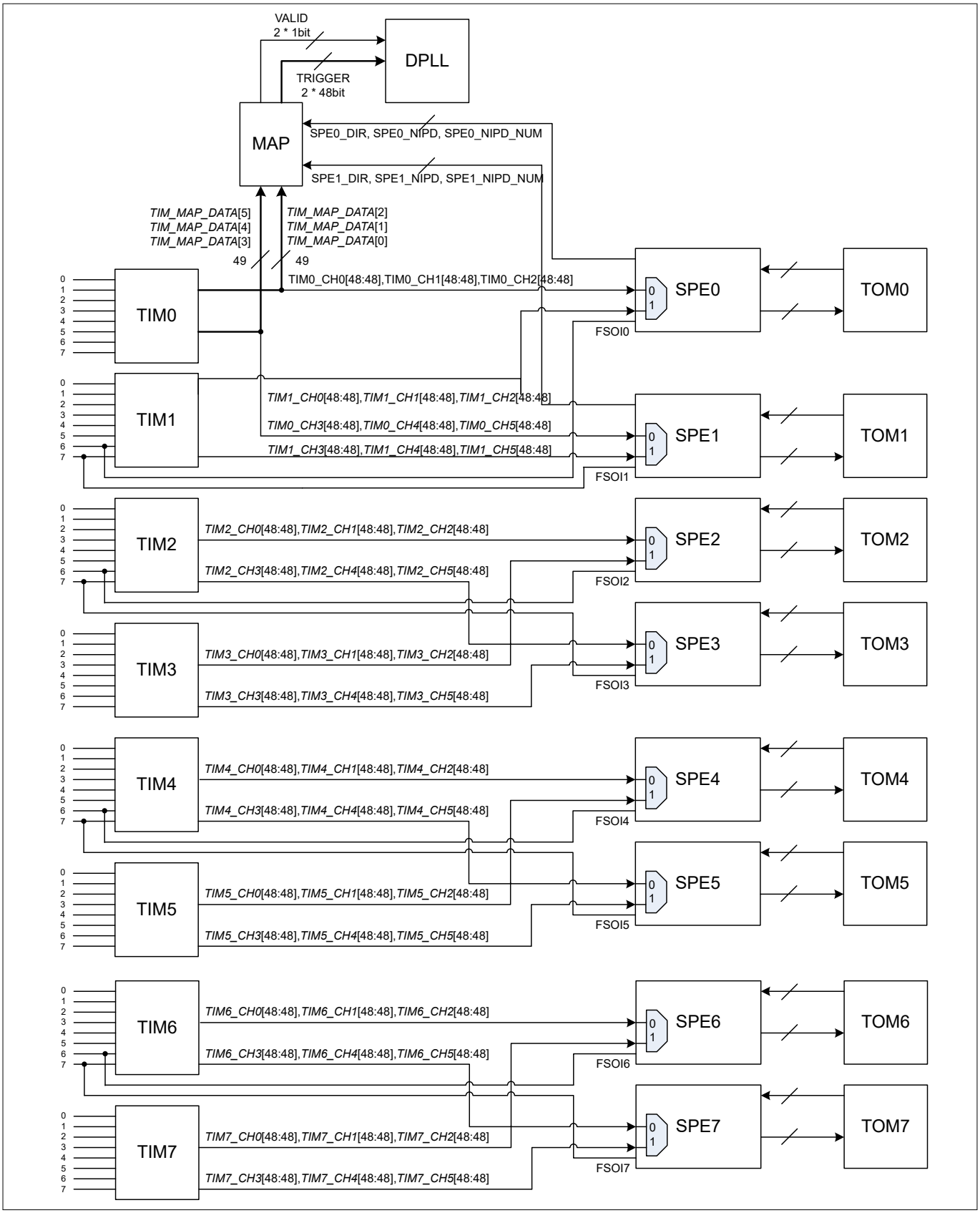


Figure 1119 SPE sub-module integration concept into eGTM

***Note**: The figure is a general picture from the IP vendor. The exact configuration can be found in the device specific configuration.*

The **SPE** sub-module can determine a rotation direction out of the combined TIM[i]_CH[x:x], TIM[i]_CH[y:y] and TIM[i]_CH[z:z] signals. On this input signals a pattern match algorithm is applied on behalf of the temporal relation between these input patterns. A possible sample pattern of the three input signals is shown in Figure 1120. In general, the input pattern is programmable within the **SPE** sub-module.

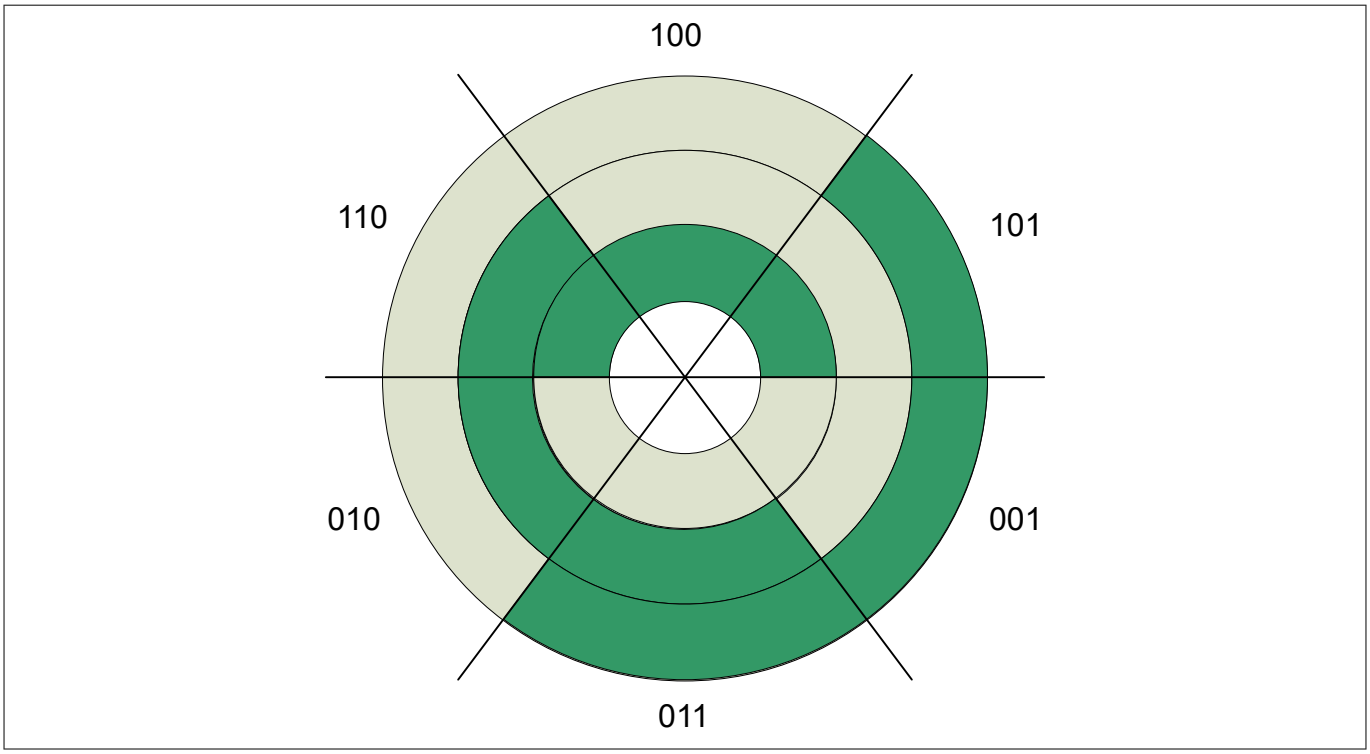


Figure 1120 SPE sample input pattern

In Figure 1120 the input signals define the pattern from the input sensors which have a 50% high and 50% low phase. The pattern according to Figure 1120 is as follows:

100 - 110 - 010 - 011 - 001 - 101 - 100

Where the first bit (smallest circle) represents TIM[i]_CH[x:x], the second bit represents TIM[i]_CH[y:y], and the third bit (greatest circle) represents TIM[i]_CH[z:z].

***Note**: The **SPE** module expects that with every new pattern only one of the three input signals changes its value.*

44.3.1.10.2 SPE sub-module description

The **SPE** sub-module can handle sensor pattern inputs. Every time if one of the input signals TIM[i]_CH[x:x], TIM[i]_CH[y:y] or TIM[i]_CH[z:z] changes its value, a sample of all three input signals is made. Derived from the sample of the three inputs the encoded rotation direction and the validity of the input pattern sequence is determined and signaled. When a valid input pattern is detected, the **SPE** sub-module can control the outputs of a dedicated connected **TOM** sub-module. This connection is shown in Figure 1121.

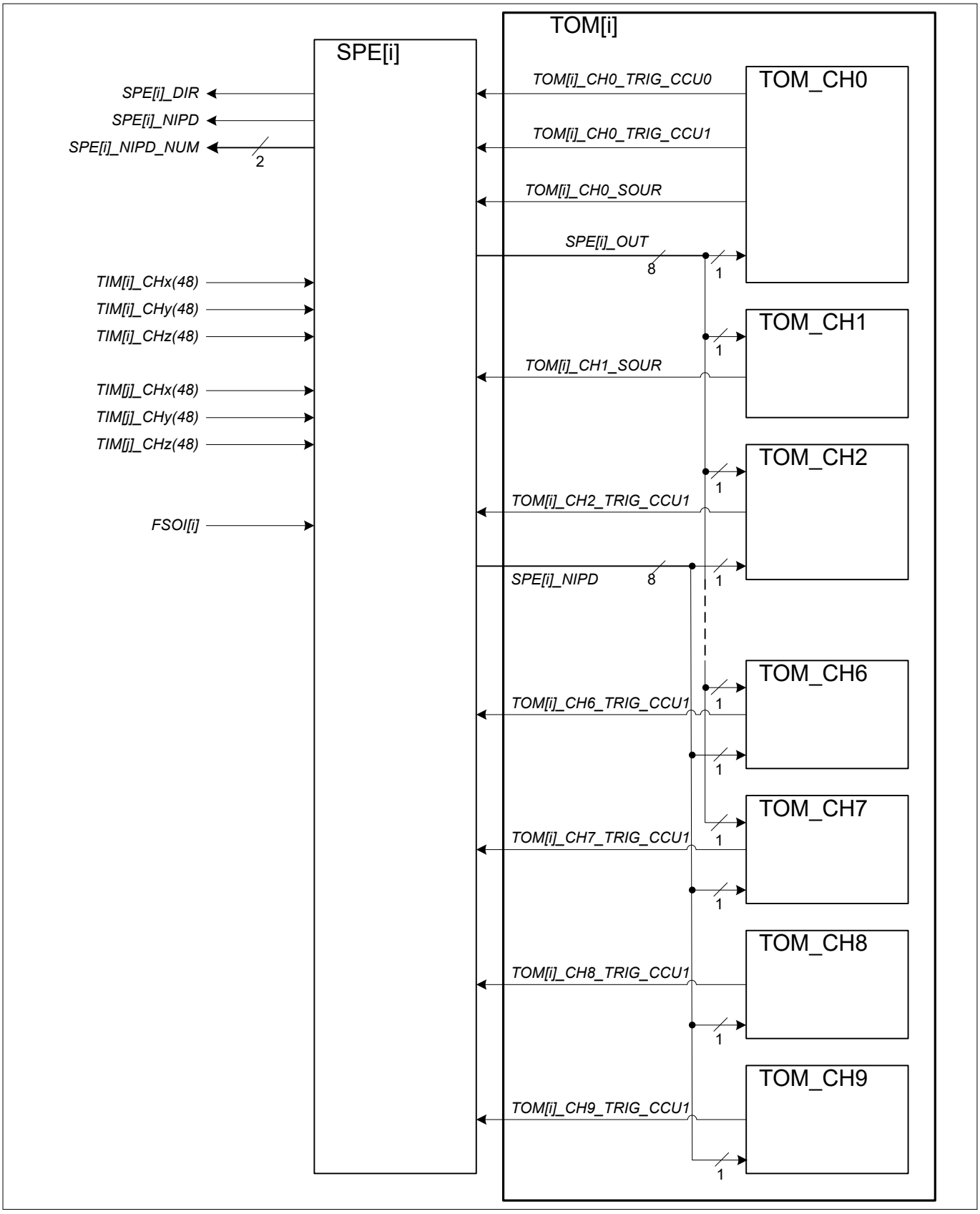


Figure 1121 SPE to TOM connections

The TOM[i]_CH[x]_TRIG_CCU0, TOM[i]_CH[x]_TRIG_CCU1 and TOM[i]_CH[x]_SOUR signal lines are used to evaluate the current state of the **TOM** outputs, whereas the SPE[i]_OUT output vector is used to control the

TOM output depending on the new input pattern. The SPE[i]_OUT output vector is defined inside the **SPE** sub-module in a pattern definition table CLSi_SPE_OUT_PATp. The internal **SPE** sub-module architecture is shown in Figure 1122.

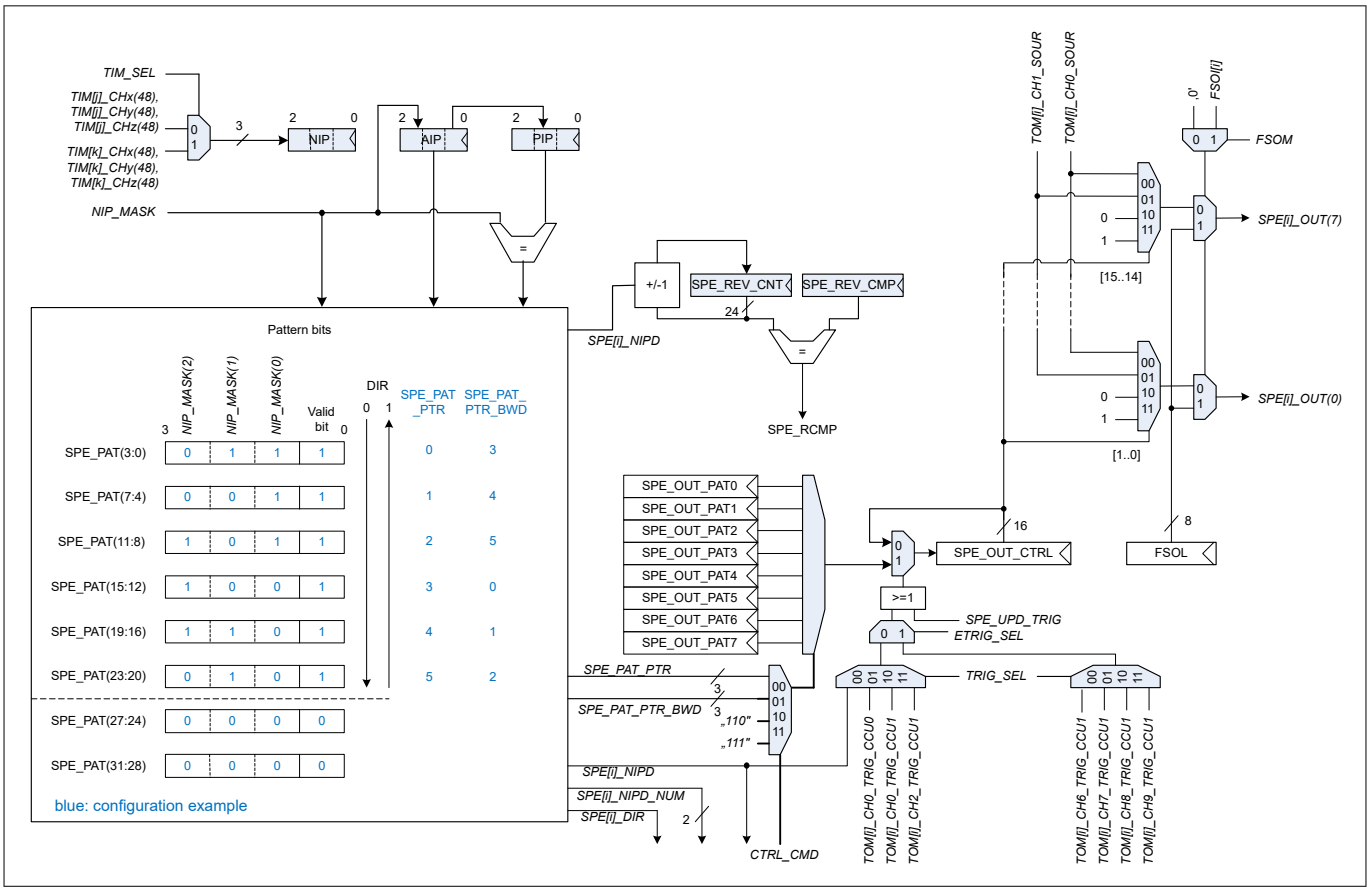


Figure 1122 SPE sub-module architecture

The CLSi_SPE_PAT register holds the valid input pattern for the three input patterns TIM[i]_CH[x:x], TIM[i]_CH[y:y] and TIM[i]_CH[z:z]. The input pattern is programmable. The valid bit shows if the programmed pattern is a valid one. Figure 1122 shows the programming of the CLSi_SPE_PAT register for the input pattern defined in Figure 1120.

The rotation direction is determined by the order of the valid input pattern. This rotation direction defines if CLSi_SPE_CTRL_STAT.SPE_PAT_PTR is incremented or decremented . Whenever a valid input pattern is detected, the SPE[i]_NIPD signal is raised, the CLSi_SPE_CTRL_STAT.SPE_PAT_PTR is incremented/ decremented and a new output control signal SPE[i]_OUT[x:x] is sent to the corresponding **TOM** sub-module.

To command directly the forward or backward rotation the **SPE** provides with CLSi_SPE_CTRL_STAT.SPE_PAT_PTR and CLSi_SPE_CTRL_STAT2.SPE_PAT_PTR_BWD two pointers to array CLSi_SPE_OUT_PATp. Both can point to different values of CLSi_SPE_OUT_PATp at the same point in time. CLSi_SPE_CTRL_STAT.SPE_PAT_PTR is intended to point to the pattern for forward commanding and CLSi_SPE_CTRL_STAT2.SPE_PAT_PTR_BWD is intended to point to the pattern for backward commanding.

On startup, both pointers have to be configured to an initial value that corresponds to different direction depending on the start pattern of CLSi_SPE_OUT_PATp.

With each valid new input pattern indicated by SPE[i]_NIPD , both pointers will be incremented or decremented according to the detected direction.

Switching from forward command to backward command can then be done by changing the selected pointer to CLSi_SPE_OUT_PATp array, i.e. changing CLSi_SPE_CMD.SPE_CTRL_CMD from selecting CLSi_SPE_CTRL_STAT.SPE_PAT_PTR to selecting CLSi_SPE_CTRL_STAT2.SPE_PAT_PTR_BWD or vice versa.

The intended behavior is depicted in the following figure:

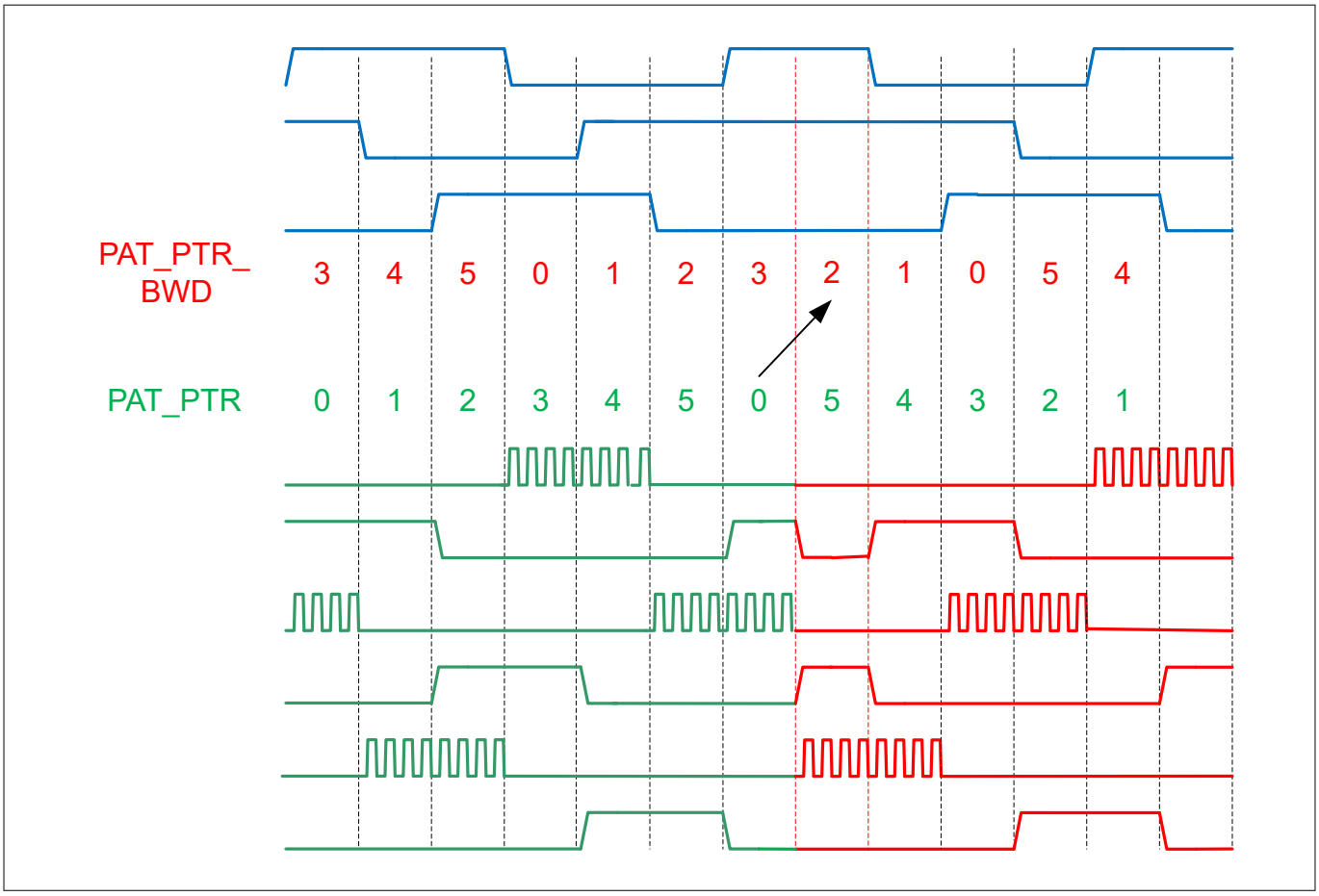


Figure 1123 SPE forward-backward commanding

With command CLSi_SPE_CMD.SPE_CTRL_CMD = 10_B or 11_B a dedicated configurable output pattern configured to the pattern CLSi_SPE_OUT_PATp.SPE_OUT_PATo (o=0-7) **(o=6)** or CLSi_SPE_OUT_PATp.SPE_OUT_PATo (o=0-7) **(o=7)** can be commanded to the outputs.

An example is the introduction of a SW dead-time if switching from pointer CLSi_SPE_CTRL_STAT.SPE_PAT_PTR to CLSi_SPE_CTRL_STAT2.SPE_PAT_PTR_BWD or vice versa. E.g. if in CLSi_SPE_OUT_PATp **(p=6)** the value 10_B is programmed for each output (i.e. set SPE[i]_OUT[n:n] to 0), this can be used as an intermediate step to introduce this 'all off' when switching between CLSi_SPE_CTRL_STAT.SPE_PAT_PTR to CLSi_SPE_CTRL_STAT2.SPE_PAT_PTR_BWD or vice versa.

Selectable by CLSi_SPE_CTRL_STAT.TRIG_SEL and CLSi_SPE_CTRL_STAT.ETRIG_SEL the CCU1 trigger of either the **TOM** channel z (z∈{2, 6, 7, 8, 9}) can be used together with the **SPE** module to trigger a delayed update of the CLSi_SPE_OUT_CTRL.SPE_OUT_CTRLo (o=0-7) register after new input pattern detected by **SPE** (signaled by SPE[i]_NIPD).

To do this, the **TOM** channel z (z∈{2, 6, 7, 8, 9}) has to be configured to work in one-shot mode (set bit CLSi_TOM_CHx_CTRL.OSM **(x=z)**). The **SPE** trigger of this channel has to be enabled too (set description of bit CLSi_TOM_CHx_CTRL.SPEM **(x=z)** and bit CLSi_TOM_CHx_CTRL.SPE_TRIG **(x=z)**). The **SPE** module has to be configured to update CLSi_SPE_OUT_CTRL.SPE_OUT_CTRLo (o=0-7) on TOM[i]_CH[x]_TRIG_CCU1 (set in CLSi_SPE_CTRL_STAT bits CLSi_SPE_CTRL_STAT.TRIG_SEL to 11_B). Then, on new input detected by **SPE**, the signal SPE[i]_NIPD triggers the start of the **TOM** channel z to generate one PWM period by resetting CLSi_TOM_CHx_CN0.CN0 to 0. On second PWM edge triggered by CCU1 of **TOM** channel z, the signal TOM[i]_CH[x]_TRIG_CCU1 triggers the update of CLSi_SPE_OUT_CTRL.SPE_OUT_CTRLo (o=0-7). Be aware, that setting **TOM** CCU1 compare register CLSi_TOM_CHx_CM1.CM1 to a value of '0' will not trigger an update of the CLSi_SPE_OUT_CTRL.SPE_OUT_CTRLo (o=0-7) register.

The update of CLSi_SPE_OUT_CTRL with the content of one of the CLSi_SPE_OUT_PATp.SPE_OUT_PATo (o=0-7) register field can be triggered at any time by writing a 1 to bit CLSi_SPE_CMD.SPE_UPD_TRIG.

The regular trigger for update of CLSi_SPE_OUT_CTRL (commutation trigger) is selected by CLSi_SPE_CTRL_STAT.TRIG_SEL and CLSi_SPE_CTRL_STAT.ETRIG_SEL.

According to Figure 1122, the two input patterns 000_B and 111_B are not allowed combinations and will end in a SPE[i]_PERR interrupt. These two patterns can be used to determine a sensor input error. A SPE[i]_PERR interrupt will also be raised, if the input patterns occur in a wrong order, e.g. if the pattern 010_B does not follow the pattern 110_B or 011_B.

The CLSi_SPE_CTRL_STAT bit field is implemented inside the CLSi_SPE_CTRL_STAT register, where the input pattern history is stored by the **SPE** sub-module. The CPU can determine a broken sensor when the SPE[i]_PERR interrupt occurs by analyzing the bit pattern readable via bit field CLSi_SPE_CTRL_STAT.NIP inside the CLSi_SPE_CTRL_STAT register. The input pattern in the CLSi_SPE_CTRL_STAT register is updated whenever a valid edge is detected on one of the input lines TIM[i]_CH[x:x], TIM[i]_CH[y:y] or TIM[i]_CH[z:z]. The pattern bit fields are then shifted. The input pattern history generation inside the CLSi_SPE_CTRL_STAT register is shown in Figure 1124.

In addition to the sensor pattern evaluation, the **SPE** module also provides the feature of fast shutoff for all **TOM** channels controlled by the **SPE** module. The feature is enabled by setting bit CLSi_SPE_CTRL_STAT.FSOM. The fast shutoff level itself is defined in the bit field CLSi_SPE_CTRL_STAT.FSOL of register CLSi_SPE_CTRL_STAT. The **TIM** input used to trigger the fast shutoff is either **TIM** channel 6 or **TIM** channel 7 depending on the **TIM** instance connected to the **SPE** module. For details of connections please refer to Figure 1119.

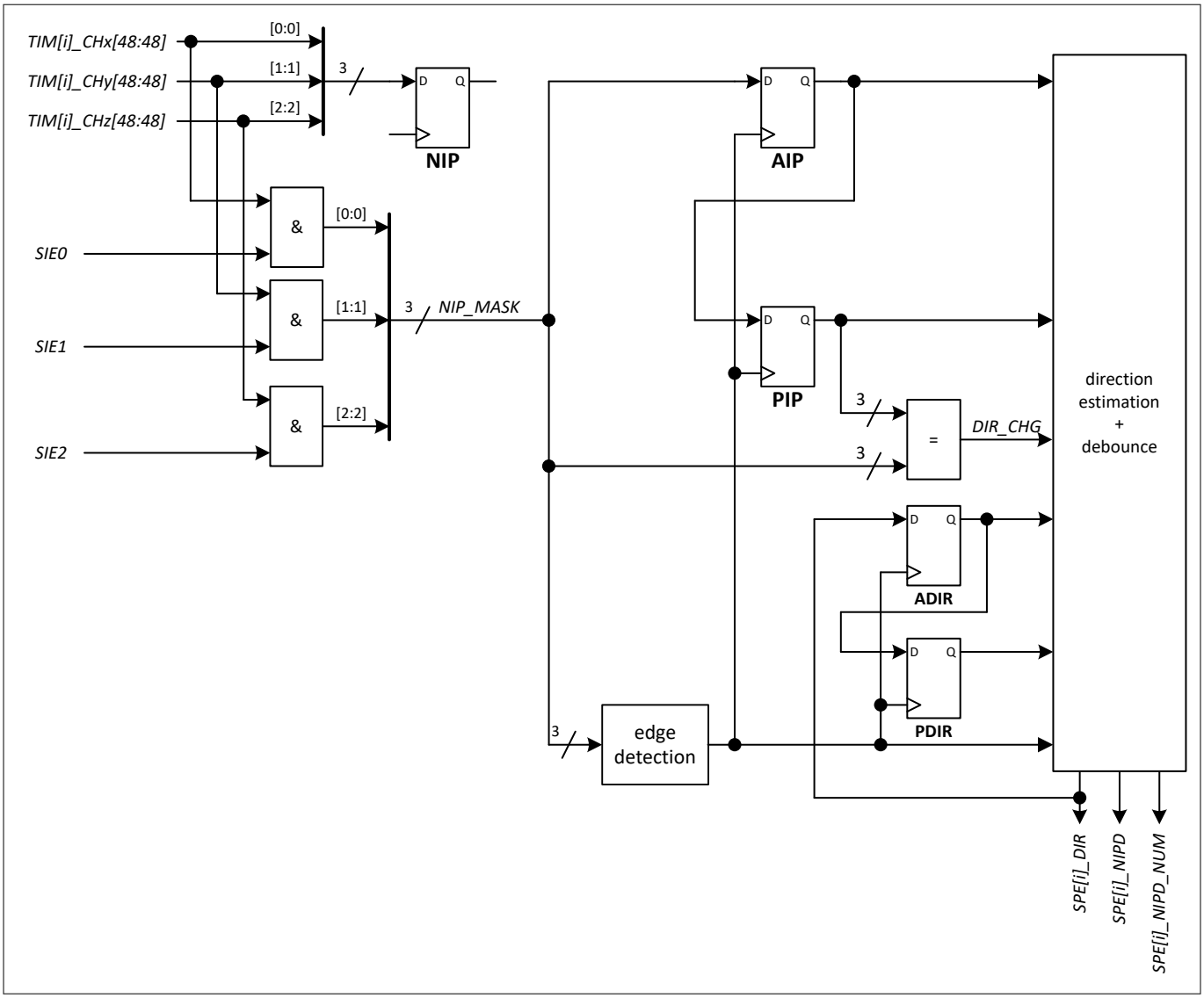


Figure 1124 CLSi_SPE_CTRL_STAT **register representation**

The CPU can disable one of the three input signals, e.g. when a broken input sensor was detected, by disabling the input with the three input enable bits CLSi_SPE_CTRL_STAT.SIEk (k=0-2).

Whenever at least one of the input signals TIM[i]_CH[x:x], TIM[i]_CH[y:y] or TIM[i]_CH[z:z] changes, the **SPE** sub-module stores the new bit pattern in an internal register bit field CLSi_SPE_CTRL_STAT.NIP (New Input Pattern). If the current input pattern in CLSi_SPE_CTRL_STAT.NIP is the same as in the previous input pattern (CLSi_SPE_CTRL_STAT.PIP) the direction of the engine is changed, the SPE[i]_DCHG interrupt is raised, the direction change is stored internally and the pattern in the CLSi_SPE_CTRL_STAT.PIP bit field is filled with the CLSi_SPE_CTRL_STAT.AIP bit field and the CLSi_SPE_CTRL_STAT.AIP bit field is filled with the CLSi_SPE_CTRL_STAT.NIP bit field. The CLSi_SPE_CTRL_STAT.ADIR bit inside the CLSi_SPE_CTRL_STAT register is toggled.

If the **SPE** encounters that with the next input pattern, new input pattern CLSi_SPE_CTRL_STAT.NIP again detects the change in direction, the input signal is categorized as bouncing and the bouncing input signal interrupt SPE[i]_BIS is raised.

Immediately after update of register bit field CLSi_SPE_CTRL_STAT.NIP, when the new detected input pattern doesn't match the CLSi_SPE_CTRL_STAT.PIP pattern (i.e. no direction change was detected), the **SPE** shifts the value of register CLSi_SPE_CTRL_STAT.AIP to register CLSi_SPE_CTRL_STAT.PIP and the value of register CLSi_SPE_CTRL_STAT.NIP to register CLSi_SPE_CTRL_STAT.AIP. The SPE[i]_NIPD interrupt is raised.

If a sensor error was detected, the CPU has to define the pattern in the CLSi_SPE_CTRL_STAT register, which input line comes from the broken sensor. The faulty signal line has to be masked by the CPU and the **SPE** sub-module determines the rotation direction on behalf of the two remaining TIM[i]_CH[x:x] input lines.

The pattern history can be determined by the CPU by reading the two bit fields CLSi_SPE_CTRL_STAT.AIP and CLSi_SPE_CTRL_STAT.PIP of the CLSi_SPE_CTRL_STAT register. The CLSi_SPE_CTRL_STAT.AIP register field holds the current input pattern at TIM[i]_CH[x:x], TIM[i]_CH[y:y] and TIM[i]_CH[z:z] and the CLSi_SPE_CTRL_STAT.PIP holds the previous pattern.

After reset the register fields CLSi_SPE_CTRL_STAT.NIP, CLSi_SPE_CTRL_STAT.AIP and CLSi_SPE_CTRL_STAT.PIP as well as the registers CLSi_SPE_CTRL_STAT.SPE_PAT_PTR and CLSi_SPE_OUT_CTRL will not contain valid startup values which would allow correct behavior after enabling **SPE** and detecting the first input patterns.

Thus, it is necessary to initialize these registers to correct values.

To do this, before enabling the **SPE**, the bit field CLSi_SPE_CTRL_STAT.NIP can be read and depending on this value the initialization values for the register fields CLSi_SPE_CTRL_STAT.AIP, CLSi_SPE_CTRL_STAT.PIP, CLSi_SPE_CTRL_STAT.SPE_PAT_PTR and CLSi_SPE_OUT_CTRL can be determined.

SPE revolution detection

The **SPE** sub-module is able to detect and count the number of valid input patterns detected at the specified input ports. This is done with a 24-bit revolution counter CLSi_SPE_REV_CNT.REV_CNT. The counter is incremented by a value of one (1) when a new valid input pattern indicating forward direction is detected. The counter is decremented by a value of one (1) when a new valid input pattern indicating backward direction is detected.

In addition, there exists a 24-bit CLSi_SPE_REV_CMP register. The user can initialize this register with a compare value, where an interrupt SPE[i]_RCMP is raised, when the revolution counter equals the compare value either in forward or backward direction.

Both registers may be written by software at any time.

44.3.1.10.3 eGTM SPE Configuration Registers Description

Sub-chapter SPE Configuration Registers Description

Table 2257

Register Shortname	Description
CLSi_SPE_CTRL_STAT	SPE[i] control status register (i=0)
CLSi_SPE_PAT	SPE[i] input pattern definition register (i=0-2)
CLSi_SPE_OUT_PATp	SPE[i] output definition register [p] (p=0-7;i=0-2)
CLSi_SPE_OUT_CTRL	SPE[i] output control register (i=0-2)
CLSi_SPE_REV_CNT	SPE[i] Input Revolution Counter (i=0-2)
CLSi_SPE_REV_CMP	SPE[i] Revolution Counter Compare Value (i=0-2)
CLSi_SPE_IRQ_NOTIFY	SPE[i] interrupt notification register (i=0-2)
CLSi_SPE_IRQ_EN	SPE[i] interrupt enable register (i=0-2)
CLSi_SPE_IRQ_FORCINT	SPE[i] Interrupt Generation By Software (i=0-2)
CLSi_SPE_IRQ_MODE	SPE[i] interrupt mode configuration register (i=0-2)
CLSi_SPE_EIRQ_EN	SPE[i] error interrupt enable register (i=0-2)

(table continues...)

Table 2257 (continued)

Register Shortname	Description
CLSi_SPE_CTRL_STAT2	SPE[i] control status register 2 (i=0-2)
CLSi_SPE_CMD	SPE[i] command register (i=0-2)

44.3.1.10.4 SPE signal description

Table 2258 SPE Signals

Name	Description
SPE[i]_BIS_EIRQ	**SPE** Bouncing input signal error interrupt.
SPE[i]_RCMP_EIRQ	**SPE** Revolution counter compare error interrupt.
TIM[i]_CH	**TIM** instance i filtered output F_OUT .
TRIG_SEL	Trigger select signal.

Table 2259 SPE Interrupt Signals

Name	Description
SPE[i]_DCHG	**SPE** Rotation direction change detected on behalf of input pattern.
SPE[i]_NIPD	**SPE** New valid input pattern detected.
SPE[i]_RCMP	**SPE** Revolution counter compare value reached.
SPE[i]_PERR	**SPE** Invalid input pattern detected.
SPE[i]_BIS	**SPE** Bouncing input signal detected at input.

44.3.1.10.5 SPE port description

Table 2260 SPE Ports

Name	Description
SPE[i]_OUT	**SPE** output signal to **TOM**.

44.3.1.11 Interrupt concentrator module (ICM)

44.3.1.11.1 Overview

The Interrupt Concentrator Module (**ICM**) is used to group (bundle) the **eGTM** interrupt lines of the individual sub-modules into interrupt groups. By bundling, a smaller number of interrupt lines is visible at the outside of the **eGTM**.

The individual interrupts of the **eGTM** sub-modules and channels have to be enabled or disabled inside the sub-modules and channels.

The feed-through architecture of bundled interrupt lines is used for the sub-modules **AEI**, **CMP**, **SPE**, **TIM**, **TOM**, **ATOM**.

To determine the detailed interrupt source the microcontroller has to read the sub-module/channel interrupt notification register NOTIFY and serve the channel individual interrupt.

***Note**: The interrupts are only visible inside the **ICM** and in consequence outside of the **eGTM**, when the interrupt is enabled inside the sub-modules themselves.*

Indices and their range as used inside this chapter are:

- j:={0, 1,..., NModule-1} instance index of module NModule:={NTOM, NATOM, NSPE}
- x = channel index in a module (**TOM** / **ATOM** / **SPE**)
- g = index for dedicated register groups

44.3.1.11.2 Bundling

The **eGTM** sub-module individual interrupt sources are connected to the **ICM**. There, the individual interrupt lines are either fed through and signaled to the outside world or bundled a second time and signaled to the outside world.

The **ICM** interrupt bundling is described in the following sections:

eGTM infrastructure interrupt bundling

The first interrupt group contains interrupts of the infrastructure and safety components of the **eGTM**. This interrupt group therefore includes interrupt lines coming from the **AEI**, **SPE** and **CMP** sub-modules. In this interrupt group each individual channel of the sub-modules has its own interrupt line to the outside world.

Thus, the active interrupt line can be used by the CPU to determine the **eGTM** sub-module channel that raised the interrupt. The interrupts are also represented in the CLSi_ICM_IRQG_0 register. This register is typically not read by the CPU, but it is readable.

The interrupt line status for each **SPE** is shown in the CLSi_ICM_IRQG_SPE_CI register. Typically, the interrupt source determined by the corresponding interrupt line and the CLSi_ICM_IRQG_SPE_CI register is typically not read out by the CPU, but is readable.

TIM interrupt bundling

Inside this group, sub-modules which handle **eGTM** input signals are treated. This is the case for the **TIM[j]** sub-modules. Each **TIM** sub-module channel is able to generate 6 individual interrupts if enabled inside the **TIM** channel. These six interrupts are bundled into one interrupt per **TIM** channel connected to the **ICM**.

The **ICM** performs no further bundling. Thus for the **eGTM**, 8 interrupt lines per **TIM** module are provided. The channel responsible for the interrupt can be determined by the raised interrupt line.

In addition, the registers CLSi_ICM_IRQG_2 are mirrors for the **TIM** sub-module channel interrupts and typically not read out by the CPU, but they are readable.

TOM and ATOM interrupt bundling

For the **TOM** and **ATOM** sub-modules, the interrupts are bundled within the **ICM** sub-module a second time to reduce external interrupt lines. The interrupts are **OR**ed in a manner that one **eGTM** external interrupt line represents two adjacent **TOM** or **ATOM** channel interrupts. For **TOM[j]** and **ATOM[j]** the bundling is as shown in Table 2261.

Table 2261 ATOM interrupt bundling within ICM

ATOM[j] input IRQs	ATOM-output IRQs (OR-ed)
ATOM[j]_CH0_IRQ or ATOM[j]_CH1_IRQ	(e)GTM_ATOM[i]_IRQ[0:0]
ATOM[j]_CH2_IRQ or ATOM[j]_CH3_IRQ	(e)GTM_ATOM[i]_IRQ[1:1]
ATOM[j]_CH4_IRQ or ATOM[j]_CH5_IRQ	(e)GTM_ATOM[i]_IRQ[2:2]
ATOM[j]_CH6_IRQ or ATOM[j]_CH7_IRQ	(e)GTM_ATOM[i]_IRQ[3:3]

Table 2262 TOM interrupt bundling within ICM

TOM[j]-input IRQs	TOM-output IRQs (OR-ed)
TOM[j]_CH0_IRQ or TOM[j]_CH1_IRQ	(e)GTM_TOM[i]_IRQ[0:0]
TOM[j]_CH2_IRQ or TOM[j]_CH3_IRQ	(e)GTM_TOM[i]_IRQ[1:1]
TOM[j]_CH4_IRQ or TOM[j]_CH5_IRQ	(e)GTM_TOM[i]_IRQ[2:2]
TOM[j]_CH6_IRQ or TOM[j]_CH7_IRQ	(e)GTM_TOM[i]_IRQ[3:3]
TOM[j]_CH8_IRQ or TOM[j]_CH9_IRQ	(e)GTM_TOM[i]_IRQ[4:4]
TOM[j]_CH10_IRQ or TOM[j]_CH11_IRQ	(e)GTM_TOM[i]_IRQ[5:5]
TOM[j]_CH12_IRQ or TOM[j]_CH13_IRQ	(e)GTM_TOM[i]_IRQ[6:6]
TOM[j]_CH14_IRQ or TOM[j]_CH15_IRQ	(e)GTM_TOM[i]_IRQ[7:7]

The interrupts coming from the **TOM[j]** sub-modules are registered in the CLSi_ICM_IRQG_TOMg_CI register. Two **TOM**s are always bundled in one **ICM** register for e.g., TOM0 and TOM1 are bundled in CLSi_ICM_IRQG_TOMg_CI. To identify the **TOM** sub-module channel in which the interrupt occurred, the CPU has to read out the CLSi_ICM_IRQG_TOMg_CI register first before it goes to the **TOM** sub-module channel itself.

The CLSi_ICM_IRQG_TOMg_CI register bits are cleared automatically when their corresponding interrupt in the sub-module channels is cleared.

The interrupts coming from the **ATOM[j]** sub-modules are registered in the CLSi_ICM_IRQG_ATOMg_CI register. Four **ATOM**s are bundled in one **ICM** register for e.g., ATOM0, ATOM1, ATOM2 and ATOM3 are bundled in CLSi_ICM_IRQG_ATOMg_CI. To identify the **ATOM** sub-module channel in which the interrupt occurred, the CPU has to read out the CLSi_ICM_IRQG_ATOMg_CI register first before it goes to the **ATOM** sub-module channel itself.

The CLSi_ICM_IRQG_ATOMg_CI register bits are cleared automatically when their corresponding interrupt in the sub-module channels is cleared.

The interrupt line status of two 16 channel **TOM**s is shown in each CLSi_ICM_IRQG_TOMg_CI register. TOM0 and TOM1 are bundled in CLSi_ICM_IRQG_TOMg_CI **(g=0)**. The interrupt source is determined by the corresponding interrupt line and the CLSi_ICM_IRQG_TOMg_CI registers are typically not read out by the CPU, but they are readable.

The interrupt line status of four 8 channel **ATOM**s is shown in each CLSi_ICM_IRQG_ATOMg_CI register, ATOM0, ATOM1, ATOM2 and ATOM3 are bundled in CLSi_ICM_IRQG_ATOMg_CI **(g=0)**. Typically, the interrupt source is determined by the corresponding interrupt line and the CLSi_ICM_IRQG_ATOMg_CI registers are typically not read out by the CPU, but they are readable.

Module error interrupt bundling

The Module Error Interrupt group handles the error interrupts coming from the **TIM**, **SPE**, **CMP** sub-modules of the **eGTM**. The Module Error interrupts are additionally identified in the CLSi_ICM_IRQG_MEI error interrupt group register. This register is typically not read out by the CPU, but it is readable.

In addition, the error interrupt line status for each **SPE** is shown in the CLSi_ICM_IRQG_SPE_CEI register. Typically, the error interrupt source is determined by the corresponding interrupt line and the CLSi_ICM_IRQG_SPE_CEI register is not read out by the CPU, but it is readable.

TIM channel error interrupt bundling

The **TIM** Channel Error Interrupt group handles the error interrupts coming from the **TIM** channel of the **eGTM**. The **TIM** Channel Error interrupts are additionally identified for the sub-modules TIM0, TIM1, TIM2 and TIM3 in the CLSi_ICM_IRQG_CEI1 error interrupt group register. These registers are typically not read out by the CPU, but they are readable.

The CLSi_ICM_IRQG_CEI1 register bits are cleared automatically, when their corresponding error interrupt in the sub-module channel is cleared.

Error interrupt cluster bundling

The Error Interrupt lines of up to 4 clusters are bundled in each CLSi_ICM_IRQG_CLSg_MEI.

Actually each cluster collects one EIRQ of one **TIM**, **SPE**.

These registers are typically not read out by the CPU, but they are readable.

44.3.1.11.3 ICM toplevel interrupt signals

In the **ICM** module each toplevel interrupt signal is pipelined with a single stage register, which operates on the **eGTM** clock CLK.

The above table shows the **eGTM** interrupt lines that are visible outside of the **eGTM**.

44.3.1.11.4 eGTM ICM Configuration Registers Description

Sub-chapter ICM Configuration Registers Description

Table 2263

Register Shortname	Description
CLSi_ICM_IRQG_0	ICM_IRQG_0 (i=0)
CLSi_ICM_IRQG_2	ICM Interrupt group register covering TIM0, TIM1, TIM2, TIM3 (i=0)

(table continues...)

Table 2263 (continued)

Register Shortname	Description
CLSi_ICM_IRQG_MEI	ICM Interrupt group register for module error interrupt information (i=0)
CLSi_ICM_IRQG_CEI1	ICM Interrupt group register 1 for channel error interrupt information (i=0)
CLSi_ICM_IRQG_SPE_CI	ICM Interrupt group SPE for module Interrupt information (i=0)
CLSi_ICM_IRQG_SPE_CEI	ICM Interrupt group SPE for module Error Interrupt information (i=0)
CLSi_ICM_IRQG_TOMg_CI	ICM_IRQG_TOM_[g]_CI (g=0;i=0)
CLSi_ICM_IRQG_TOMg_CI	ICM_IRQG_TOM_[g]_CI (g=1;i=0)
CLSi_ICM_IRQG_ATOMg_CI	ICM_IRQG_ATOM_[g]_CI (g=0;i=0)
CLSi_ICM_IRQG_CLSg_MEI	ICM_IRQG_CLS_[g]_MEI (g=0;i=0)

44.3.1.11.5 ICM signal description

Table 2264 ICM TOM Interrupt Signals

Name	Description
TOM[j]_CH0_IRQ	**TOM[j]** interrupt signal channel 0
TOM[j]_CH1_IRQ	**TOM[j]** interrupt signal channel 1
TOM[j]_CH2_IRQ	**TOM[j]** interrupt signal channel 2
TOM[j]_CH3_IRQ	**TOM[j]** interrupt signal channel 3
TOM[j]_CH4_IRQ	**TOM[j]** interrupt signal channel 4
TOM[j]_CH5_IRQ	**TOM[j]** interrupt signal channel 5
TOM[j]_CH6_IRQ	**TOM[j]** interrupt signal channel 6
TOM[j]_CH7_IRQ	**TOM[j]** interrupt signal channel 7
TOM[j]_CH8_IRQ	**TOM[j]** interrupt signal channel 8
TOM[j]_CH9_IRQ	**TOM[j]** interrupt signal channel 9
TOM[j]_CH10_IRQ	**TOM[j]** interrupt signal channel 10
TOM[j]_CH11_IRQ	**TOM[j]** interrupt signal channel 11
TOM[j]_CH12_IRQ	**TOM[j]** interrupt signal channel 12
TOM[j]_CH13_IRQ	**TOM[j]** interrupt signal channel 13
TOM[j]_CH14_IRQ	**TOM[j]** interrupt signal channel 14
TOM[j]_CH15_IRQ	**TOM[j]** interrupt signal channel 15

Table 2265 ICM ATOM Interrupt Signals

Name	Description
ATOM[j]_CH0_IRQ	**ATOM[j]** interrupt signal channel 0
ATOM[j]_CH1_IRQ	**ATOM[j]** interrupt signal channel 1
ATOM[j]_CH2_IRQ	**ATOM[j]** interrupt signal channel 2
ATOM[j]_CH3_IRQ	**ATOM[j]** interrupt signal channel 3
ATOM[j]_CH4_IRQ	**ATOM[j]** interrupt signal channel 4
ATOM[j]_CH5_IRQ	**ATOM[j]** interrupt signal channel 5
ATOM[j]_CH6_IRQ	**ATOM[j]** interrupt signal channel 6
ATOM[j]_CH7_IRQ	**ATOM[j]** interrupt signal channel 7

44.3.1.12 Output compare unit (CMP)

44.3.1.12.1 Overview

The Output Compare Unit (**CMP**) is designed for the use in safety relevant applications. The main idea is to have the possibility to duplicate outputs in order to be compared in this unit. Because of the simple EXOR function used, it is necessary to ensure the total cycle accurate output behavior of the output modules to be compared. This is given when two neighboring **DTM** channels (e.g. CDTM0_DTM4_CH0_OUT and CDTM0_DTM4_CH1_OUT, for neighboring **DTM** channel pairs see Table 2272) generate identical signals with phase shift zero at their outputs. This can be reached if they start their output generation at the same time. This start of synchronization is possible by means of the trigger mechanisms provided by the **TOM** or **ATOM** as shown in the **TOM** Timer output module (**TOM**) or **ATOM** **ARU**-connected timer output module (**ATOM**).

Indices and their range as used inside this chapter are:

- i:={0, 1, 2} (index of cluster or module instance)
- c:={0, 1,..., 11} (index of bitwise comparators)
- d:={0, 1,..., 5} (index of **DTM** unit after **TOM**/ **ATOM**)
- x:={0, 1, 2} (index of output channel in **DTM**)

The **CMP** enables the comparison of 2x24 channels of the CDTM[i]_DTM[d] instances and is to neighboring channels. The first 24 **CMP** channels are the 24 **DTM** channels placed behind **TOM[0]** and **TOM[1]** instances and the second 24 **CMP** channels are the 24 **DTM** channels placed behind the **ATOM[0]**, **ATOM[1]** and **ATOM[2]** instances.

***Note**: In case the cluster 1 is operating on half **eGTM** system clock CLK and signals in cluster 0 or 2 are generated on a **eGTM** system clock CLK resolution, a deviation of the compare signals by one CLK clock cycle might not be detected by the bitwise compare unit.*

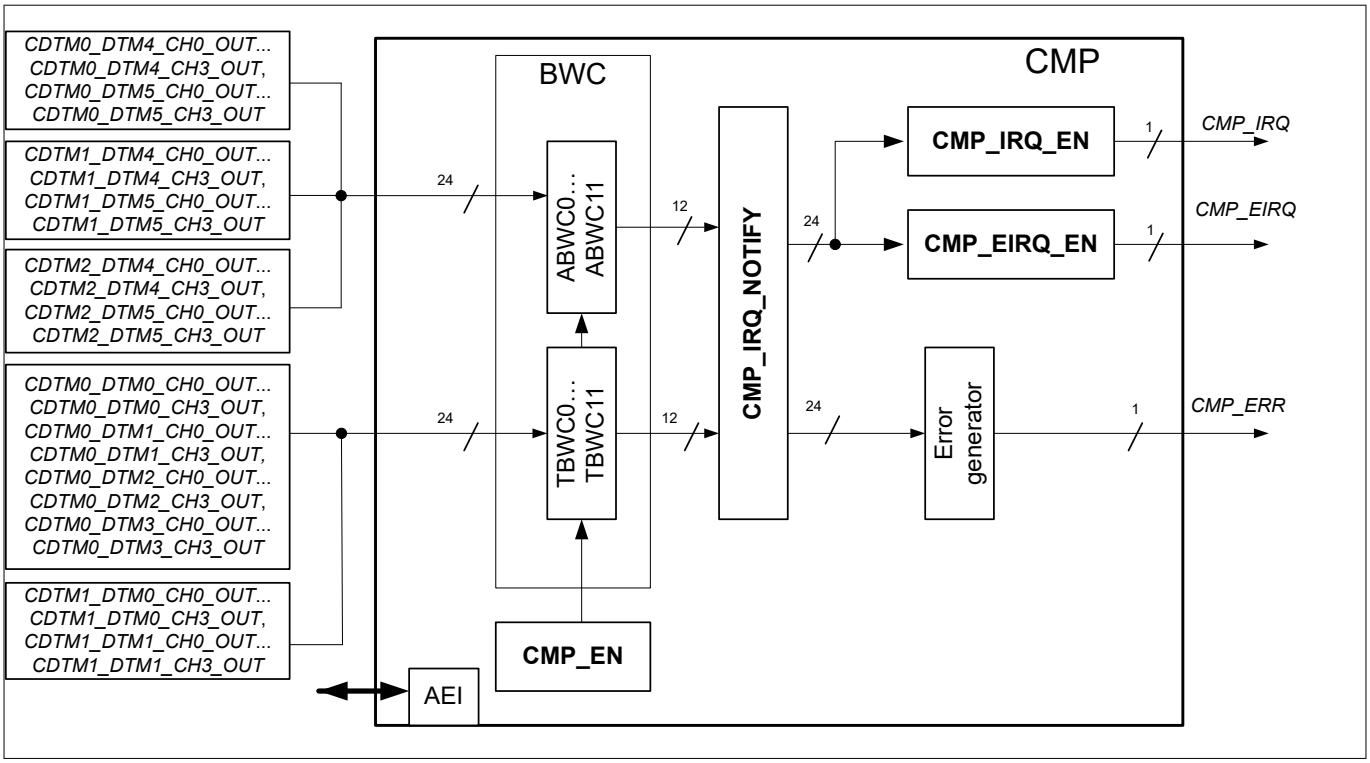


Figure 1125 Architecture of the compare unit

44.3.1.12.2 Bitwise compare unit (BWC)

The Bitwise Compare Unit ABWC compares the combinations shown in following table in pairs.

Table 2266 ABWC compare unit

ABWC Comparator	Comparator Input 1	Comparator Input 2
ABWC0	CDTM0_DTM4_CH0_OUT	CDTM0_DTM4_CH1_OUT
ABWC1	CDTM0_DTM4_CH2_OUT	CDTM0_DTM4_CH3_OUT
ABWC2	CDTM0_DTM5_CH0_OUT	CDTM0_DTM5_CH1_OUT
ABWC3	CDTM0_DTM5_CH2_OUT	CDTM0_DTM5_CH3_OUT
ABWC4	CDTM1_DTM4_CH0_OUT	CDTM1_DTM4_CH1_OUT
ABWC5	CDTM1_DTM4_CH2_OUT	CDTM1_DTM4_CH3_OUT
ABWC6	CDTM1_DTM5_CH0_OUT	CDTM1_DTM5_CH1_OUT
ABWC7	CDTM1_DTM5_CH2_OUT	CDTM1_DTM5_CH3_OUT
ABWC8	CDTM2_DTM4_CH0_OUT	CDTM2_DTM4_CH1_OUT
ABWC9	CDTM2_DTM4_CH2_OUT	CDTM2_DTM4_CH3_OUT
ABWC10	CDTM2_DTM5_CH0_OUT	CDTM2_DTM5_CH1_OUT
ABWC11	CDTM2_DTM5_CH2_OUT	CDTM2_DTM5_CH3_OUT

The Bitwise Compare Unit TBWC compares the combinations shown in following table in pairs.

Table 2267 TBWC compare unit

TBWC Comparator	Comparator Input 1	Comparator Input 2
TBWC0	CDTM0_DTM0_CH0_OUT	CDTM0_DTM0_CH1_OUT
TBWC1	CDTM0_DTM0_CH2_OUT	CDTM0_DTM0_CH3_OUT
TBWC2	CDTM0_DTM1_CH0_OUT	CDTM0_DTM1_CH1_OUT
TBWC3	CDTM0_DTM1_CH2_OUT	CDTM0_DTM1_CH3_OUT
TBWC4	CDTM0_DTM2_CH0_OUT	CDTM0_DTM2_CH1_OUT
TBWC5	CDTM0_DTM2_CH2_OUT	CDTM0_DTM2_CH3_OUT
TBWC6	CDTM0_DTM3_CH0_OUT	CDTM0_DTM3_CH1_OUT
TBWC7	CDTM0_DTM3_CH2_OUT	CDTM0_DTM3_CH3_OUT
TBWC8	CDTM1_DTM0_CH0_OUT	CDTM1_DTM0_CH1_OUT
TBWC9	CDTM1_DTM0_CH2_OUT	CDTM1_DTM0_CH3_OUT
TBWC10	CDTM1_DTM1_CH0_OUT	CDTM1_DTM1_CH1_OUT
TBWC11	CDTM1_DTM1_CH2_OUT	CDTM1_DTM1_CH3_OUT

44.3.1.12.3 Configuration of the compare unit

Because of the restrictions described in the section above, the Compare Unit consists of 24 antivalence (EXOR) elements, a select register CLSi_CMP_EN which selects the corresponding comparisons and a status register CLSi_CMP_IRQ_NOTIFY which shows and stores each mismatching result, when selected.

For each mismatching error enabled with CLSi_CMP_IRQ_EN an interrupt signal on CMP_IRQ is generated.

For each mismatching error enabled with CLSi_CMP_EIRQ_EN an interrupt signal on CMP_EIRQ is generated.

44.3.1.12.4 Error generator

The error generator generates an error signal to be transmitted to the **MON** unit directly and independent of the CMP_IRQ and CMP_EIRQ. The error is set when in the CLSi_CMP_IRQ_NOTIFY register at least one bit is set. The CLSi_CMP_IRQ_NOTIFY bits are not maskable for this purpose.

Additionally, CMP_ERR is a primary output port for interrupt actions by CPU itself.

44.3.1.12.5 CMP interrupt signals

The **CMP** sub-module has two interrupt signals, one normal interrupt and one error interrupt. The source of both interrupts can be determined by reading the CLSi_CMP_IRQ_NOTIFY register under consideration of the registers CLSi_CMP_IRQ_EN and CLSi_CMP_EIRQ_EN. Each source can be forced separately for debug purposes using the interrupt force CLSi_CMP_IRQ_FORCINT register. CLSi_CMP_IRQ_MODE configures interrupt output characteristic. All interrupt modes are described in detail in **eGTM** interrupt concept.

44.3.1.12.6 eGTM CMP Configuration Registers Description

Sub-chapter CMP Configuration Registers Description

Table 2268

Register Shortname	Description
CLSi_CMP_EN	CMP comparator enable register (i=1)
CLSi_CMP_IRQ_NOTIFY	CMP event notification register (i=1)
CLSi_CMP_IRQ_EN	CMP interrupt enable register (i=1)
CLSi_CMP_IRQ_FORCINT	CMP interrupt force register (i=1)
CLSi_CMP_IRQ_MODE	CMP interrupt mode configuration register (i=1)
CLSi_CMP_EIRQ_EN	CMP error interrupt enable register (i=1)

44.3.1.12.7 CMP port description

Table 2269 CMP signal interface

Name	Description
TOM0	Output signals from TOM0
TOM1	Output signals from TOM1
ATOM0	Output signals from ATOM0
ATOM1	Output signals from ATOM1
ATOM2	Output signals from ATOM2

Table 2270 CMP interrupt interface

Name	Description
CMP_IRQ	**CMP IRQ** line signal
CMP_EIRQ	**CMP** error **IRQ** line signal

Table 2271 CMP ERR interface

Name	Description
CMP_ERR	**CMP** error line to **MON**

Table 2272 CMP ABWC Comparator Input 1

Name	Description
CDTM0_DTM4_CH0_OUT	ABWC0 Comparator Input 1
CDTM0_DTM4_CH2_OUT	ABWC1 Comparator Input 1
CDTM0_DTM5_CH0_OUT	ABWC2 Comparator Input 1
CDTM0_DTM5_CH2_OUT	ABWC3 Comparator Input 1
CDTM1_DTM4_CH0_OUT	ABWC4 Comparator Input 1
CDTM1_DTM4_CH2_OUT	ABWC5 Comparator Input 1

(table continues...)

Table 2272 (continued) CMP ABWC Comparator Input 1

Name	Description
CDTM1_DTM5_CH0_OUT	ABWC6 Comparator Input 1
CDTM1_DTM5_CH2_OUT	ABWC7 Comparator Input 1
CDTM2_DTM4_CH0_OUT	ABWC8 Comparator Input 1
CDTM2_DTM4_CH2_OUT	ABWC9 Comparator Input 1
CDTM2_DTM5_CH0_OUT	ABWC10 Comparator Input 1
CDTM2_DTM5_CH2_OUT	ABWC11 Comparator Input 1

Table 2273 CMP ABWC Comparator Input 2

Name	Description
CDTM0_DTM4_CH1_OUT	ABWC0 Comparator Input 2
CDTM0_DTM4_CH3_OUT	ABWC1 Comparator Input 2
CDTM0_DTM5_CH1_OUT	ABWC2 Comparator Input 2
CDTM0_DTM5_CH3_OUT	ABWC3 Comparator Input 2
CDTM1_DTM4_CH1_OUT	ABWC4 Comparator Input 2
CDTM1_DTM4_CH3_OUT	ABWC5 Comparator Input 2
CDTM1_DTM5_CH1_OUT	ABWC6 Comparator Input 2
CDTM1_DTM5_CH3_OUT	ABWC7 Comparator Input 2
CDTM2_DTM4_CH1_OUT	ABWC8 Comparator Input 2
CDTM2_DTM4_CH3_OUT	ABWC9 Comparator Input 2
CDTM2_DTM5_CH1_OUT	ABWC10 Comparator Input 2
CDTM2_DTM5_CH3_OUT	ABWC11 Comparator Input 2

Table 2274 CMP TBWC Comparator Input 1

Name	Description
CDTM0_DTM0_CH0_OUT	TBWC0 Comparator Input 1
CDTM0_DTM0_CH2_OUT	TBWC1 Comparator Input 1
CDTM0_DTM1_CH0_OUT	TBWC2 Comparator Input 1
CDTM0_DTM1_CH2_OUT	TBWC3 Comparator Input 1
CDTM0_DTM2_CH0_OUT	TBWC4 Comparator Input 1
CDTM0_DTM2_CH2_OUT	TBWC5 Comparator Input 1
CDTM0_DTM3_CH0_OUT	TBWC6 Comparator Input 1
CDTM0_DTM3_CH2_OUT	TBWC7 Comparator Input 1
CDTM1_DTM0_CH0_OUT	TBWC8 Comparator Input 1
CDTM1_DTM0_CH2_OUT	TBWC9 Comparator Input 1

(table continues...)

Table 2274 (continued) CMP TBWC Comparator Input 1

Name	Description
CDTM1_DTM1_CH0_OUT	TBWC10 Comparator Input 1
CDTM1_DTM1_CH2_OUT	TBWC11 Comparator Input 1

Table 2275 CMP TBWC Comparator Input 2

Name	Description
CDTM0_DTM0_CH1_OUT	TBWC0 Comparator Input 2
CDTM0_DTM0_CH3_OUT	TBWC1 Comparator Input 2
CDTM0_DTM1_CH1_OUT	TBWC2 Comparator Input 2
CDTM0_DTM1_CH3_OUT	TBWC3 Comparator Input 2
CDTM0_DTM2_CH1_OUT	TBWC4 Comparator Input 2
CDTM0_DTM2_CH3_OUT	TBWC5 Comparator Input 2
CDTM0_DTM3_CH1_OUT	TBWC6 Comparator Input 2
CDTM0_DTM3_CH3_OUT	TBWC7 Comparator Input 2
CDTM1_DTM0_CH1_OUT	TBWC8 Comparator Input 2
CDTM1_DTM0_CH3_OUT	TBWC9 Comparator Input 2
CDTM1_DTM1_CH1_OUT	TBWC10 Comparator Input 2
CDTM1_DTM1_CH3_OUT	TBWC11 Comparator Input 2

44.3.1.13 Monitor unit (MON)

44.3.1.13.1 Overview

The Monitor Unit (**MON**) is designed for the use in safety relevant applications. The main idea is to have a possibility to supervise commonly used circuitry and resources. In this way the activity of the clocks is supervised. By generation of an activity signal per channel for each such performed comparison, the activity of **TIM** and the used clocks is checked implicitly.

Indices and their range as used inside this chapter are:

- x:={0, 1,..., 7} channel index of **TOM**, **ATOM** channels
- y:={0, 1,..., 4} index for fixed clock for **TOM** FXCLK(y)
- z:={0, 1} index for external clocks ECLK(z)

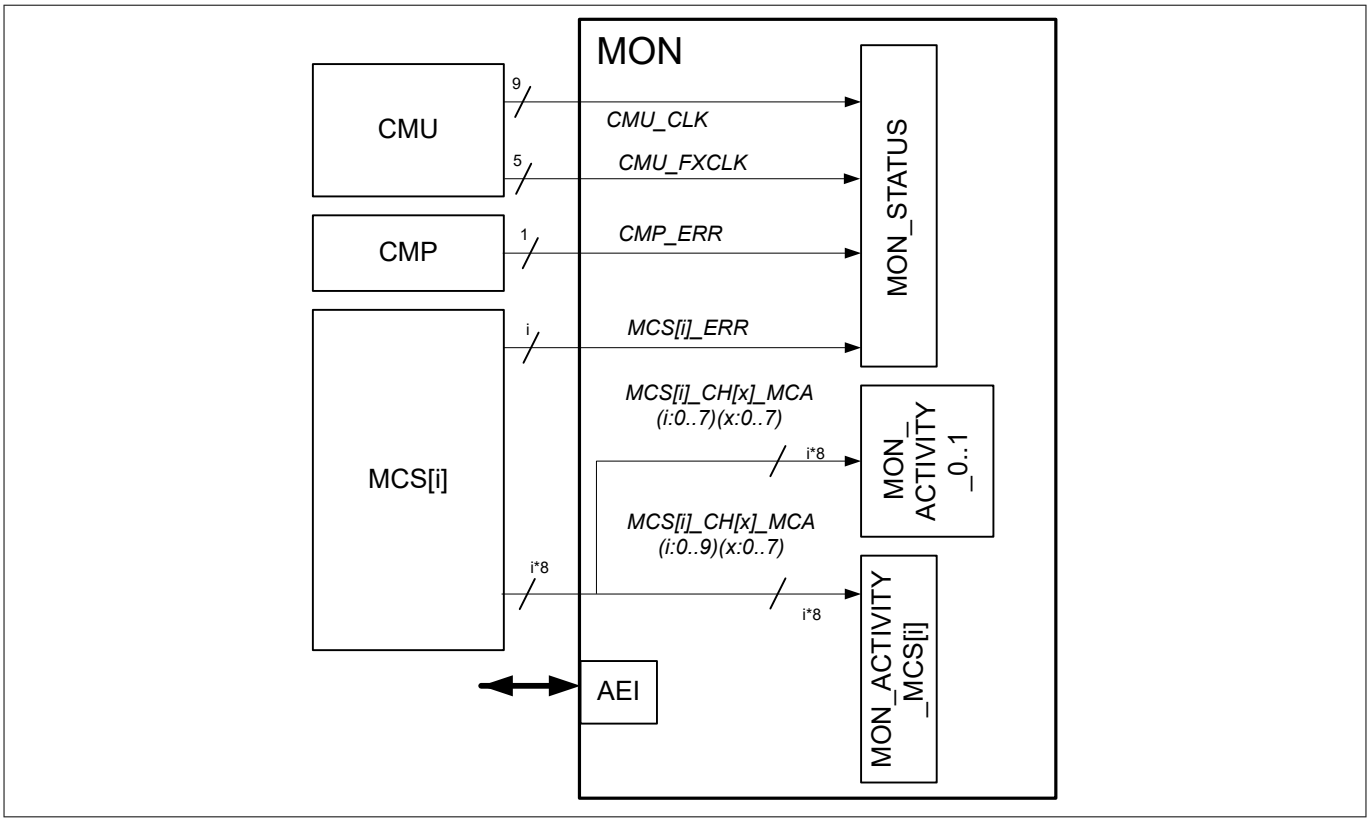


Figure 1126 MON block diagram

***Note**: The figure is a general picture from the IP vendor. The exact configuration can be found in the device specific configuration.*

Realization without activity checker of the clock signals

An activity checker of the clock signals used is not needed because these signals are only enables to be used in combination with the system clock. Therefore, the clock enables are to be checked to have a high value.

44.3.1.13.2 Clock monitoring

The monitor unit has a connection to each of the 9 clocks CCM[i]_CLK_RES, provided by the **CMU**. Some of these clocks can be used for special tasks (see Clock management unit (**CMU**)).

In addition, the 5 clock inputs of the **TOM**s CCM[i]_FXCLK_RES are also connected to the **MON** unit.

The supervising of the clocks is done by scanning for activity of each clock.

A high value is defined as the state to be monitored.

When a high value of the clock enable is detected, the corresponding bit in the status register CLSi_MON_STATUS is set.

The status register bits are reset by writing a one.

When the register is polled by the CPU and the time between two read-accesses is higher than the period of the slowest clock, all bits of the corresponding clocks must have been set.

When polling in shorter time distances, not for all clocks an activity can be shown, although they are still working.

Because of the realization without a select register for the clock signals only the bits of the status register are to be considered for which the clock signal is enabled in the **CMU**.

44.3.1.13.3 CMP error monitoring

The signal CLSi_MON_STATUS.CMP_ERR is to be received directly from module **CMP** and is set if an error occurred.

44.3.1.13.4 MON interrupt signals

The **MON** sub-module has no interrupt signals.

44.3.1.13.5 eGTM MON Configuration Registers Description

Sub-chapter MON Configuration Registers Description

Table 2276

Register Shortname	Description
CLSi_MON_STATUS	MON status register (i=1)

44.3.1.13.6 MON port description

Table 2277 MON Ports

Name	Description
MON_TBU_TS0	Timestamp 0 from **TBU**, not in **MON**

44.3.1.14 Appendix A

44.3.1.14.1 Acronym definitions

Table 2278 Acronym definition

Abbreviation	Definition/Meaning
ADC	Analog to Digital Converter
ADTM	**ATOM** with **DTM**
AEI	Automotive Electronics Interface (Bosch proprietary bus protocol)
ASIC	Application-Specific Integrated Circuit
ATPG	Automatic Test Pattern Generation
ATOM	**ARU**-connected Timer Output Module
CDC	Clock Domain Crossing
CMP	Output Compare Unit
CMU	Clock Management Unit
DfT	Design for Testability
DMA	Direct Memory Access
DTM	Dead-time Module

(table continues...)

Table 2278 (continued) Acronym definition

Abbreviation	Definition/Meaning
EC	Equivalence Check
FLT	Filter Unit in **TIM** Channel
eGTM	enhanced Generic Timer Module
(E)GTM_AEI	Bus interface bridge (module which converts multiple **AEI** bus protocol variants to **eGTM** internal accesses)
GTM_RM	enhanced Generic Timer Module Reference Model
ICM	Interrupt Concentrator Module
ICPA	Inter cluster pulse adapter
IP	Intellectual Property
IRC	Interrupt Controller
IRQ	Interrupt Request
LBIST	Logic Built-In Self-Test
MON	Monitor Unit
PCM	Pulse Count Modulation
PWM	Pulse Width Modulation
SPE	Sensor Pattern Evaluation
TBU	Time Base Unit
TDTM	**TOM** with **DTM**
TDU	Timeout detection unit in **TIM** channel
TIM	Timer Input Module
TIM_CH	Channel unit in **TIM** module
TOM	Timer Output Module

44.3.1.14.2 eGTM device configuration variables

Table 2279 eGTM device configuration variables, define available module resources

Device configuration variable	Type	Array index definition	Description
ATOM_HIGH_RES	ARRAY[NATOM] of INT	[i]	Number of available high-resolution channels in **ATOM** instance of cluster i.
CDTM	ARRAY[NCDTM] of LIST	[i]	The i-th array member defines the list of **DTM** instances available within the cluster i.

(table continues...)

Table 2279 (continued) eGTM device configuration variables, define available module resources

Device configuration variable	Type	Array index definition	Description
CFG_CLOCK_RATE	INT		Configurable clock rate available.
FAST_CLK_CLS	ARRAY[NCCM] of INT	[i]	0: Cluster i only supports clock frequency divided by 2. 1: Cluster i supports with clock frequency divider 1/2.
INT_CLK_EN_GEN	INT		**eGTM** internal clock enable signals in use.
NATOM	INT		Number of **ATOM** modules.
NCCM	INT		Number of **CCM** modules.
NCDTM	INT		Number of elements in array **CDTM**.
NCMP	INT		Number of **CMP** modules.
NMON	INT		Number of **MON** modules.
NSPE	INT		Number of **SPE** modules.
NTIM	INT		Number of **TIM** modules.
NTOM	INT		Number of **TOM** modules.

Table 2280 eGTM device configuration variables

Device configuration variable	Type	Description
AEI_ADDR_PIPELINE_STAGE	INT	**AEI** address pipeline stage implemented
AEI_RDATA_PIPELINE_STAGE	INT	**AEI** read data pipeline stage implemented
ATOM_OUT_RST	INT	ATOM_OUT reset level.
ATOM_TRIG_CHAIN	INT	Define **ATOM** external trigger chain configuration.
ATOM_TRIG_INTCHAIN	INT	Define **ATOM** internal trigger chain configuration.
AXIS_DATA_WIDTH	INT	AXI slave data bus width.
BRIDGE_BUFF_DPT	INT	Buffer depth of the **eGTM AEI** bridge implementation.
GRSTEN	INT	Global **eGTM** reset register enabled.
IRQ_MODE_RST_VAL	INT	Initial value of **IRQ** mode.

(table continues...)

Table 2280 (continued) eGTM device configuration variables

Device configuration variable	Type	Description
RESET_ACTIVE	INT	Active level of asynchronous reset
SYNC_INPUT_REG	INT	Additional pipelined stage implemented. All accesses in synchronous.
TOM_OUT_RST	INT	TOM_OUT reset level.
TOM_TRIG_CHAIN	INT	Define **TOM** external trigger chain configuration.
TOM_TRIG_INTCHAIN	INT	Define **TOM** internal trigger chain configuration.

Table 2281 eGTM device configuration variables, fixed or derived values

Device configuration variable	Type	Array index definition	Description
ATOM_OUT_N_WITH_DTM	ARRAY[NCDTM][8] of INT	[i][x]	0: Channel x of signal ATOM_OUT_N in cluster i has no **DTM** implemented. 1: Channel x of signal ATOM_OUT_N in cluster i has a **DTM** implemented.
GTM_REV_DEVICE_CODE	INT		Device encoding digit.
GTM_REV_REL_BASE	INT		**eGTM** release base.
GTM_REV_REL_ITER	INT		**eGTM** release iteration.
GTM_REV_VENDOR_CODE	INT		Vendor encoding digit.
GTM_REV_VER_MAJOR	INT		Define major version number of **eGTM** specification.
GTM_REV_VER_MINOR	INT		Define minor version number of **eGTM** specification.
NARP	INT		Number of Address Range Protectors per **CCM**.
NHBP	INT		Number of Hardware Break Points.
TOM_OUT_N_WITH_DTM	ARRAY[NCDTM][16] of INT	[i][x]	0: Channel x of signal TOM_OUT_N in cluster i has no **DTM** implemented. 1: Channel x of signal TOM_OUT_N in cluster i has a **DTM** implemented.

44.3.1.14.3 Specific definitions

Table 2282 Specific definition

Definition	Description
Pulse width	It is a definition for pulse-width modulation (PWM) signal. In a PWM period, pulse width is the time when a signal or system is active (in **eGTM** specification on or off is possible). The signal level in the pulse width can be specified as 1 or 0, e.g. configured by CLSi_ATOM_CHx_CTRL.SL in **ATOM** module.
Duty cycle	It is a definition for pulse-width modulation (PWM) signal. The term duty cycle describes the proportion of pulse time to the total PWM period. In **eGTM** specification, duty cycle can be specified as the ratio of "on" time or "off" time of a PWM signal.

44.3.1.14.4 eGTM application constraints

Table 2283 Application constraints for eGTM 4100

Nr	Module	Description	Required value	Effect, when not considered
6	**TOM**	If channel is triggered by preceding channel via TOM_CH_TRIGOUT[x-1:x-1] signal, the selected SEL_FXCMU_CLKEN/ CCM[i]_FXCLK_RES of both channels has to be the same.	Identical SEL_FXCMU_CLKEN/ CCM[i]_FXCLK_RES	The trigger of preceding channel may be lost
7	**ATOM**	If channel is triggered by preceding channel via ATOM_CH_TRIGOUT [x-1:x-1] signal, the selected SEL_CMU_CLKEN/ CCM[i]_CLK_RES of both channels has to be the same.	Identical SEL_CMU_CLKEN/ CCM[i]_CLK_RES	The trigger of preceding channel may be lost

Attention: The user must be careful when writing directly to operation registers via the configuration interface. Write accesses might take a variable number of clock cycles until they are executed. For example, the load on the internal AEI bus or the external bus in the MCU could influence the duration of accesses. If the user attempts to directly perform a write access on an operation register at the point in time the update from the shadow register takes place, there is a race condition - the source that writes last will have its value stored in the register.

44.3.1.14.5 eGTM internal functional dependencies

Table 2284 Dependency of ATOM functions

ATOM[i] depends on	Valid index	Description	Type
eGTM inputs	i < NATOM	Error **IRQ** clear, **IRQ** clear signals	Cluster external uses ICPA
Module **ATOM[0]**	i == 0 && NATOM > 1	ATOM_TRIGOUT_DEL , ATOM_HRES_OUT_DEL signals	Cluster internal
Module **ATOM[i-1]**	i > 1 && i < min(NATOM+1)	ATOM_TRIGOUT signals	Cluster external uses ICPA
Module **ATOM[i-1]**	i > 1 && i < min(NATOM+1)	ATOM_HRES_OUT signals	Cluster external, no ICPA in use
Module **CCM[i]**	i < NATOM	CCM[i]_CLK_RES, CCM[i]_TBU_TS0, CCM[i]_TBU_TS1, CCM[i]_TBU_TS2 signals	Cluster internal
Module **TIM[i]**	i < min(NTIM, NATOM)	TIM_EXT_CAPTURE signals	Cluster internal

Table 2285 Dependency of ARCH functions

ARCH depends on	Valid index	Description	Type
eGTM inputs	i == 0	Error **IRQ** clear, **IRQ** clear signals	Cluster external uses ICPA

Table 2286 Dependency of CCM functions

CCM[i] depends on	Valid index	Description	Type
Module **CMU**	i < NCCM	CMU_CLK_RES, CMU_FXCLK_RES signals	Cluster external uses ICPA
Module **TBU**	i < NCCM	TBU_TS0_BASE, TBU_TS1_BASE, TBU_TS2_BASE signals	Cluster internal
Module **TBU**	i < NCCM	TBU_UP0, TBU_UP1, TBU_UP2 signals	Cluster internal
Module CDTM[i]_DTM[d]	i < NCCM && d ∈ {0, .. 5}	DTM_OUT0, DTM_OUT1 signals	Cluster internal
Module **TIM[i]**	i < NTIM	TIM_EXT_CAPTURE signals	Cluster internal

Table 2287 Dependency of CMP functions

CMP depends on	Valid index	Description	Type
eGTM inputs	NCMP == 1	Error **IRQ** clear, **IRQ** clear signals	Cluster external uses ICPA
Module CDTM[i]_DTM[d]	NCMP == 1 && i < min(NATOM, 3) && d ∈ {4, 5}	DTM_OUT0 signals	Cluster external, no ICPA in use; signals typically are not single cluster clock pulses
Module CDTM[i]_DTM[d]	NCMP == 1 && ((NTOM>1 && d ∈ {0, .. 3}) || (NTOM>2 && d ∈ {0, .. 1}))	DTM_OUT0 signals	Cluster external, no ICPA in use; signals typically are not single cluster clock pulses

Table 2288 Dependency of DTM functions

CDTM[i]_DTM[d] depends on	Valid index	Description	Type
Module **ATOM[i]**	i < NATOM && d > 3 && d < 6 && d ∈ CDTM[i]	ATOM_OUT, ATOM_OUT_T , ATOM_OUT_HRES[x], ATOM_OUT_T_HRES[x] signals; x ∈ { 0, .. 7}	Cluster internal
Module **CCM[i]**	i < NCCM && d > 3 && d ∈ CDTM[i]	CCM[i]_CLK_RES signals	Cluster internal
Module **CCM[i]**	i < NCCM && d < 4 && d ∈ CDTM[i]	CCM[i]_FXCLK_RES, CCM[i]_CLK_RES[8:8] signals	Cluster internal
Module CDTM[i]_DTM[d-1]	d ∈ {1, 2, 3, 5} && i < NCCM && d ∈ CDTM[i]	DTM_PSU_OUT, DTM_IN0_FEDGE, DTM_IN0_REDGE signals	Cluster internal
Module CDTM[i]_DTM[d-1]	d ∈ {1, 3, 5} && i < NCCM && d ∈ CDTM[i]	DTM_COUT3_O signals	Cluster internal
Module CDTM[i-1]_DTM[e]	i > 1 && e ∈ {3, 5} && d ∈ {0, 4} && i < NCCM && d ∈ CDTM[i]	DTM_PSU_OUT, DTM_IN0_FEDGE, DTM_IN0_REDGE signals	Cluster external uses ICPA
Module **TIM[i]**	i < NTIM && d < 6 && d ∈ CDTM[i]	F_OUT[x:x] signals; x ∈ { 0, .. 1}	Cluster internal
Module **TOM[i]**	i < NTOM && d < 4 && d ∈ CDTM[i]	TOM_OUT, TOM_OUT_T signals; x ∈ { 0, .. 15}	Cluster internal

Table 2289 Dependency of ICM functions

ICM depends on	Valid index	Description	Type
Module **ARCH**	i == 0	EIRQ, **IRQ** signals	Cluster internal
Module **ATOM[i]**	i < NATOM	**IRQ** signals	Cluster internal for i==0; else cluster external

(table continues...)

Table 2289 (continued) Dependency of ICM functions

ICM depends on	Valid index	Description	Type
Module **CMP**	i == 0 && NCMP > 0	EIRQ, **IRQ** signals	Cluster external
Module **SPE[i]**	i < NSPE	EIRQ, **IRQ** signals	Cluster internal for i==0; else cluster external
Module **TIM[i]**	i < NTIM	EIRQ, **IRQ** signals	Cluster internal for i==0; else cluster external
Module **TOM[i]**	i < NTOM	**IRQ** signals	Cluster internal for i==0; else cluster external

Table 2290 Dependency of MON functions

MON depends on	Valid index	Description	Type
Module **CCM[1]**	NCMP == 1 && NCCM > 1	CCM[i]_CLK_RES, CCM[i]_FXCLK_RES signals	Cluster internal
Module **CMP**	NCMP == 1 && NMON == 1	CMP_ERR signal	Cluster internal

Table 2291 Dependency of SPE functions

SPE[i] depends on	Valid index	Description	Type
eGTM inputs	i < NSPE	Error **IRQ** clear, **IRQ** clear signals	Cluster external uses ICPA

Table 2292 Dependency of TBU functions

TBU depends on	Valid index	Description	Type
Module **CCM[0]**	i == 0	CCM[i]_CLK_RES signals	Cluster internal

Table 2293 Dependency of TIM functions

TIM[i] depends on	Valid index	Description	Type
eGTM inputs	i < NTIM	Error **IRQ** clear, **IRQ** clear signals	Cluster external uses ICPA
Module **CCM[i]**	i < NTIM	CCM[i]_CLK_RES, CCM[i]_TBU_TS0, CCM[i]_TBU_TS1, CCM[i]_TBU_TS2 signals	Cluster internal
Module CDTM[i]_DTM[d]	i < NTIM && d ∈ {0, .. 5}	DTM_OUT0, DTM_OUT1 signals	Cluster internal
Module **TIM[i+1]**	i < NTIM-1	TIMCH0_EXT_CAP_SRC_OUT signals	Cluster external uses ICPA

Table 2294 Dependency of TOM functions

TOM[i] depends on	Valid index	Description	Type
eGTM inputs	i < NTOM	**IRQ** clear signals	Cluster external uses ICPA

(table continues...)

Table 2294 (continued) Dependency of TOM functions

TOM[i] depends on	Valid index	Description	Type
Module **CCM[i]**	i < NTOM	CCM[i]_FXCLK_RES, CCM[i]_TBU_TS0, CCM[i]_TBU_TS1, CCM[i]_TBU_TS2 signals	Cluster internal
Module **SPE[i]**	i < min(NTOM, NSPE)	SPE[i]_OUT, SPE[i]_NIPD, signals	Cluster internal
Module **TIM[i]**	i < min(NTIM, NTOM)	TIM_EXT_CAPTURE signals	Cluster internal
Module **TOM[0]**	i == 0 && NTOM > 1	TOM_TRIGOUT_DEL signals	Cluster internal
Module **TOM[i-1]**	i > 1 && i < min(NTOM+1)	TOM_TRIGOUT signals	Cluster external uses ICPA

44.3.1.14.6 Compatibility notes / functional changes versus eGTM v.3.1.5 release

Find next a listing of all functional changes which have been introduced with **eGTM** v4.1.x release where backward compatibility to **eGTM** v3.1.5 is not existing. Software developers have to introduce changes to their application code to ensure correct operation of the same application on **eGTM** 4.1.x devices.

eGTM architecture

- Address map changed, base address of all modules changed. Each module inside a cluster has now a defined offset to the cluster start address. As long as applications use C-HAL or symbolic names no corrective actions are needed.
- Legacy register from **eGTM** generation 1 and **eGTM** generation 2 removed.
- GTM_TIM[i]_AUX_IN_SRC replaced by CLSi_CCM_TIM_AUX_IN_SRC
- GTM_TOM0[i]_OUT replaced by CLSi_CCM_TOM_OUT
- (e)GTM_ATOM[i]_OUT replaced by CLSi_CCM_ATOM_OUT
- CLSi_ARCH_AEI_STA_XPT.W1R0 changed to bit 24.
- Redefinition of AEI_STATUS behavior in case of **AEI** configuration access to undefined, protected, disabled register. **eGTM** generic bus interface (**AEI**)

ICM

Removed the following legacy and redundant **ICM** registers: **ICM_IRQG_6**, **ICM_IRQG_7**, **ICM_IRQG_8**, **ICM_IRQG_9**, **ICM_IRQG_10**.

ATOM

- Bit field ECLK_SRC of CLSi_ATOM_CHx_CTRL removed and CLSi_ATOM_CHx_CTRL.CLK_SRC bit field expanded by one bit and coding of bit field CLSi_ATOM_CHx_CTRL.CLK_SRC changed: CLSi_ATOM_CHx_CTRL_SOMP, CLSi_ATOM_CHx_CTRL_SOMS, CLSi_ATOM_CHx_CTRL_SOMI
- Behavior of retriggering a one-shot cycle in SOMP mode changed: One-shot counting up mode

TOM

- CLSi_TOM_CHx_CTRL.CLK_SRC, CLSi_TOM_CHx_CTRL.SL shadow register functionality (see CLSi_TOM_CHx_CTRL_SR): To ensure backward compatibility, new shadow register must be written with same values as operational register CLSi_TOM_CHx_CTRL_SR = CLSi_TOM_CHx_CTRL.
- Bit field ECLK_SRC of CLSi_TOM_CHx_CTRL removed and CLSi_TOM_CHx_CTRL.CLK_SRC bit field expanded by one bit and coding of bit field CLSi_TOM_CHx_CTRL.CLK_SRC changed.
- Behavior of retriggering a one-shot cycle changed: One-shot counting up mode

CCM

- Remove bit field ARU_CONNECT_CONFIG in register CLSi_CCM_HW_CONF due to discontinued device configuration parameter.

44.3.1.14.7 New features / functional enhancements

Find next a listing of all functional enhancements which have been introduced with **eGTM** v4.1.x release where backward compatibility to **eGTM** v3.1.5 is not affected.

eGTM architecture

- GTM_CTL: extended functionality of CLSi_ARCH_CTRL.TO_MODE; internal **AEI** accesses can be retrieved or aborted. See chapter CLSi_ARCH_CTRL

ATOM

- High-resolution support: High-resolution PWM support (HRES mode)
- Retrigger one-shot mode: One-shot counting up mode , CLSi_ATOM_CHx_STAT.OSM_RTF
- Shadow register for CLSi_ATOM_CHx_CTRL.SL and CLSi_ATOM_CHx_CTRL.CLK_SRC: CLSi_ATOM_CHx_CTRL_SR
- **ATOM** OUT_T controlled by ENDIS (enhancement): Value of ATOM_OUT_T is inverse signal level (CLSi_ATOM_CHx_CTRL.SL) when channel is disabled (ENDIS=0): CLSi_ATOM_AGC_OUTEN_CTRL.OUTEN_CTRLk (k=0-7)

TOM

- Retrigger one-shot mode: One-shot counting up mode
- Shadow register for CLSi_TOM_CHx_CTRL.SL and CLK_SRC: CLSi_TOM_CHx_CTRL_SR
- **TOM** OUT_T controlled by ENDIS (enhancement): Value of TOM_OUT_T is inverse signal level (CLSi_TOM_CHx_CTRL.SL) when channel is disabled (ENDIS=0): CLSi_TOM_TGCg_OUTEN_CTRL.OUTEN_CTRLc (c=0-7)

DTM

- Dead-time shadow register including CLSi_CDTM_DTMd_CH_DTVx.RELRISE and CLSi_CDTM_DTMd_CH_DTVx.RELFALL: Dead-time shadow register, CLSi_CDTM_DTMd_CH_DTV_SRx
- Expand bitwidth of CLSi_CDTM_DTMd_CH_DTVx.RELRISE/CLSi_CDTM_DTMd_CH_DTVx.RELFALL to 13 bits.
- AUXIN enhancements for **eGTM** Gen4 (individual shutoff): **DTM** individual channel shutoff
- High-resolution support: High-resolution PWM support

TIM

- Added new coding for CLSi_TIM_CHx_ECTRL.TDU_STOP=110_B. See chapter CLSi_TIM_CHx_ECTRL

CCM

- Added register CLSi_CCM_HW_CONF2 to monitor additional device configuration parameter.
- Added bit fields CLSi_CCM_HW_CONF.RESET_ACTIVE, CLSi_CCM_HW_CONF.AEI_ADDR_PIPELINE_STAGE, CLSi_CCM_HW_CONF.AEI_RDATA_PIPELINE_STAGE to monitor additional device configuration parameter.

44.3.2 eGTM implementation

The eGTM is optimized for electrical drives applications, but it is also a simple timer, capable of all necessary PWM generation timer based functions.

***Note**: The second GTM (GTM classical approach - not available on all devices) is optimized for engine control and for all applications which need a particularly large amount of timers.*

Related information

Functional overview on page 7307
eGTM control registers on page 7575
Document structure on page 7309

44.3.2.1 TC4x family eGTM configuration

The following table shows the configuration of the eGTM module on the TC4x devices with 3 clusters. The interface to the HRPWM module is optional:

Table 2295 eGTM configuration (no GTM listed)

eGTM modules	eGTM with 3 clusters
TIM	3x8 channels (TIM0-2)
TOM	3x16 channels (TOM0-2)
ATOM	3x8 channels (ATOM0-2) with high resolution extension, if existing
DTM/CDTM	12 @ TOMs (TOM0-2) 6 @ ATOM (ATOM 0-2)
HRES	Existing, but no connection to the HRPWM supporting blocks in TC4Dx and TC4Zx
MCS	not existing
SPE	3 (SPE0-2)
PSM	not existing
DPLL	not existing
TBU	3 (TBU0-2)
BRC	not existing
MON	1
CMP	1
eGTM Clusters	3 (CCM0-2)

44.3.2.2 Naming of resources with or without DTM

The following table includes the configuration of the eGTM, including the resource names for the output signals. For eGTM all outputs are including a DTM.

Table 2296 CDTMs to resources

CDTMn	CDTMn_DTMj	Resource TOM/ATOM
CDTM0	DTM0	TOM0 channel 0-3
CDTM0	DTM1	TOM0 channel 4-7
CDTM0	DTM2	TOM0 channel 8-11
CDTM0	DTM3	TOM0 channel 12-15
CDTM1	DTM0	TOM1 channel 0-3
CDTM1	DTM1	TOM1 channel 4-7
CDTM1	DTM2	TOM1 channel 8-11
CDTM1	DTM3	TOM1 channel 12-15
CDTM2	DTM0	TOM2 channel 0-3
CDTM2	DTM1	TOM2 channel 4-7
CDTM2	DTM2	TOM2 channel 8-11
CDTM2	DTM3	TOM2 channel 12-15
CDTM0	DTM4	ATOM0 channel 0-3
CDTM0	DTM5	ATOM0 channel 4-7
CDTM1	DTM4	ATOM1 channel 0-3
CDTM1	DTM5	ATOM1 channel 4-7
CDTM2	DTM4	ATOM2 channel 0-3
CDTM2	DTM5	ATOM2 channel 4-7

The following figure shows the names of the eGTM outputs.

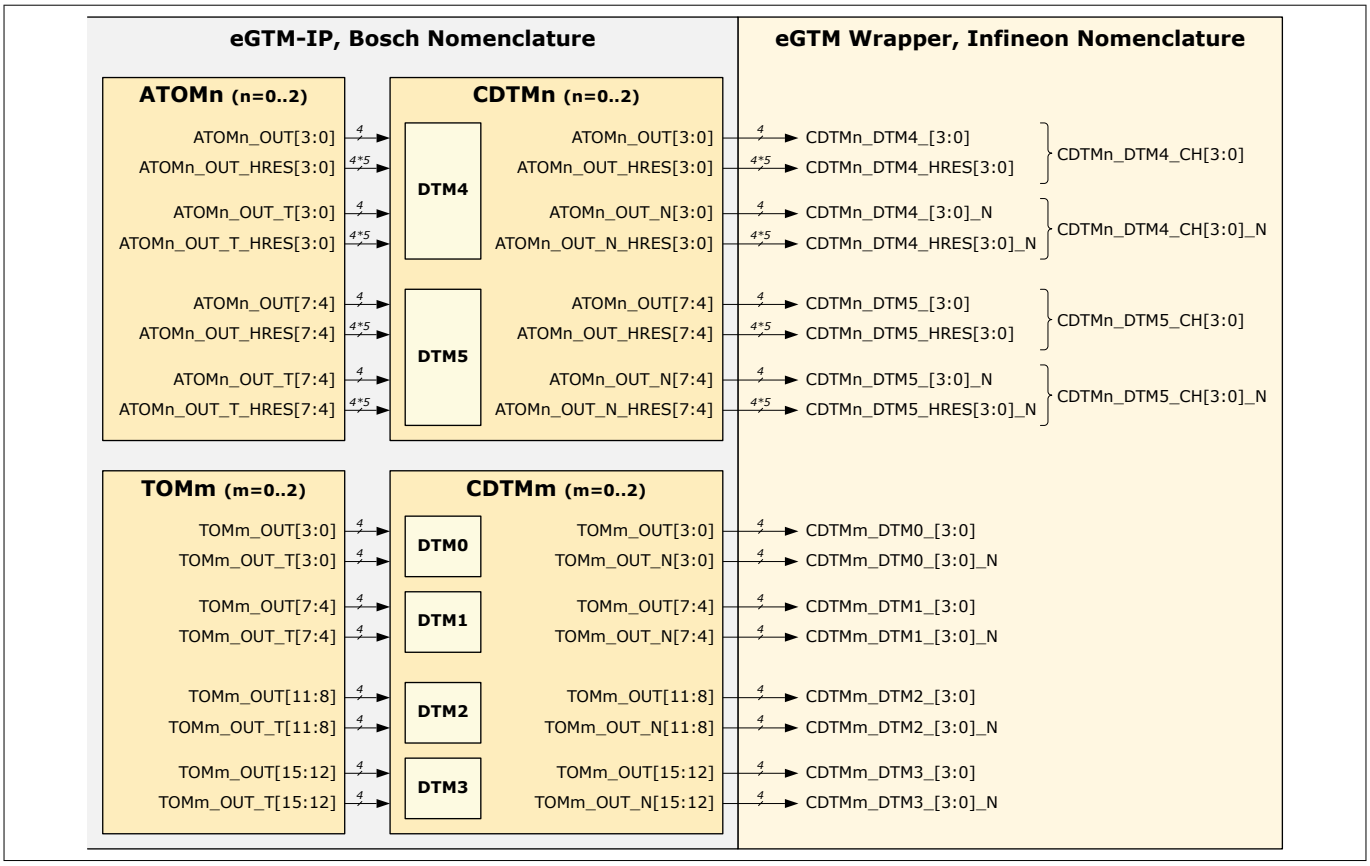


Figure 1127 Naming between eGTM-IP and eGTM wrapper

44.3.2.3 eGTM-IP internal bus bridge

The eGTM is connected to the system bus (w or w/o a bridge). The eGTM AEI bridge should use synchronous mode if $f_{eGTM} = f_{CPU}$, otherwise asynchronous mode should be used. If no other configuration takes place, f_{eGTM} is identical to f_{CPU}. Use the asynchronous mode for all clock divider settings except for divide by 1.

In case of equal frequencies:

- EGTM_CLS0_AEI_BRIDGE_MODE.B.BRG_MODE = 0; for synchronous mode

In case of different frequencies:

- EGTM_CLS0_AEI_BRIDGE_MODE.B.BRG_MODE = 1; for asynchronous mode
- EGTM_CLS0_AEI_BRIDGE_MODE.B.BYPASS_SYNC = 1; bypass synchronization

Always the following setting shall be taken:

- EGTM_CLS0_AEI_BRIDGE_MODE.B.MSK_WR_RSP = 1; Activate write buffer

***Note**: For devices with frequencies higher than 400 MHz, cluster clocks have to be set to 1:2. In case a device is supporting a frequency higher than 400 MHz, it is not allowed to program cluster clocks to higher frequencies, than 250 MHz*

44.3.2.4 eGTM is not an interrupt service provider

The eGTM is a standard module connected to the interrupt request unit. It is not a service provider.

44.3.2.5 Using the protection mechanism

This section describes the protection concept for the eGTM module. It shows the relation between the Access Mode Glossary and the Register Overview.

The eGTM protection has six different groups:

1. ACCEN and PROTSE are unprotected
2. All OCDS dependent registers, which are not part of the standard registers, and CLC, OCS, RST_CTRL(A/B) and RST_STAT are protected by APU-P
3. All wrapper registers are protected by PWRAP
4. All eGTM internal standard cluster registers have the protection according to their cluster number, for example cluster 0 has PCL0 up to cluster 2 with PCL2
5. CMP and MON are protected by PSAFE
6. ICM, TBU, CMU, ARCH and AEI related registers are protected by PGTM

The protection rights are defined in the Register Overview table. Violating these protections will trigger an alarm to SMU.

Related information

Register overview - access mode glossary on page 7621

Register overview - EGTM domain CPU (ascending offset address) on page 7621

44.3.2.6 eGTM control registers

This section lists all Infineon specific registers implemented in the TC4x family.

44.3.2.6.1 eGTM note input line 0 safe state

General remark for outputs: The multiplexer input lines 0 (selected by .SEL=0) are tied to 0, in order to enable a safe state.

44.3.2.6.2 ADC connections

This chapter introduces the connections of the eGTM to and from the ADC sub-block.

Boundary Flag naming between ADC and GTM/eGTM

Boundary flags have the following names:

Name in ADC chapter

Name in ADC chapter	Name in GTM/eGTM chapter
ETBFxSEL.OUTySEL (name of the controllable bit) eGTM_CLx_BLFy (name of the signal)	GTBFxOUTy

With x being the actual cluster number and y being the line.

Currently 4 lines exist.

Therefore the signals GTBFxOUT2_4 connect to signal eGTM_CLx_BLF2 and GTBFxOUT3_5 connect to eGTM_CLx_BLF3.

Connection between eGTM and ADC

This sub-chapter describes all interfaces from and to the ADC function block.

Feature list

This subsection is divided into the following parts:

- Input from ADC
- Trigger signals to ADC

***Note**: SENT connections are treated the same way as ADC connections, with separate register set.*

Functional description

Inputs from ADC

Input signals coming from the ADC routed to the TIM modules are controlled by the TIMnINSEL register group. The ADC output signals can be found on bit position 12 of the multiplexer for eGTM TIM input.

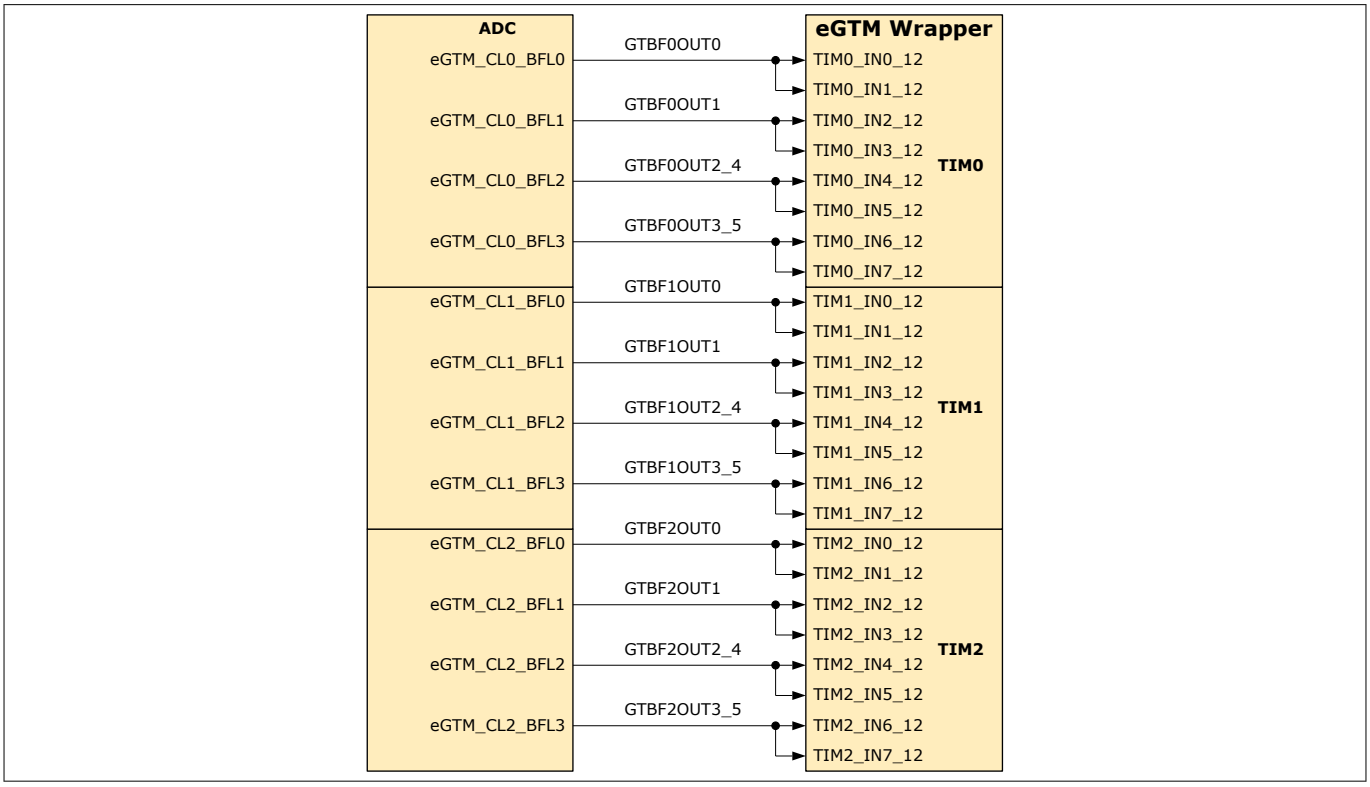


Figure 1128 ADC to TIMnINSEL connection

Further connections exist in TIMnINSELB and TIMn_ADCSELCHi (not available for TC4DxA).

Output to ADC

This chapter shows all connections from eGTM to ADCs.

ADC output

This sub-chapter describes ADC_OUT registers.

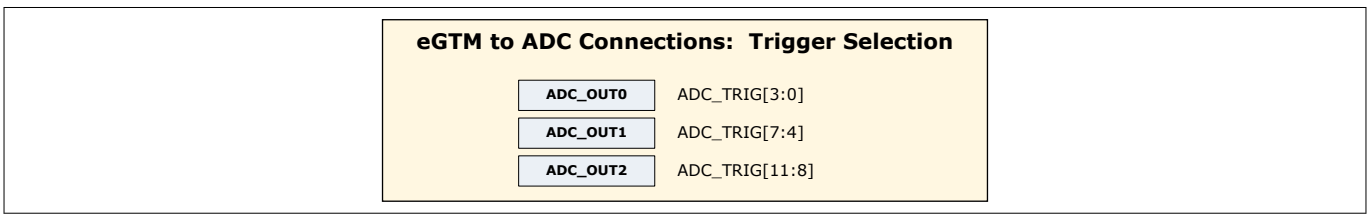


Figure 1129 Overview on ADC_OUT registers

In this register, each cluster has four multiplexers as to generate 4 trigger outputs. The bit-fields select between all ATOM and TOM resources within this cluster.

These trigger outputs enable the eGTM to trigger an action on an ADC module.

Please be aware that the output ADC_TRIG[11:0] from eGTM are mapped to input TRIG[47:36] of ADC block.

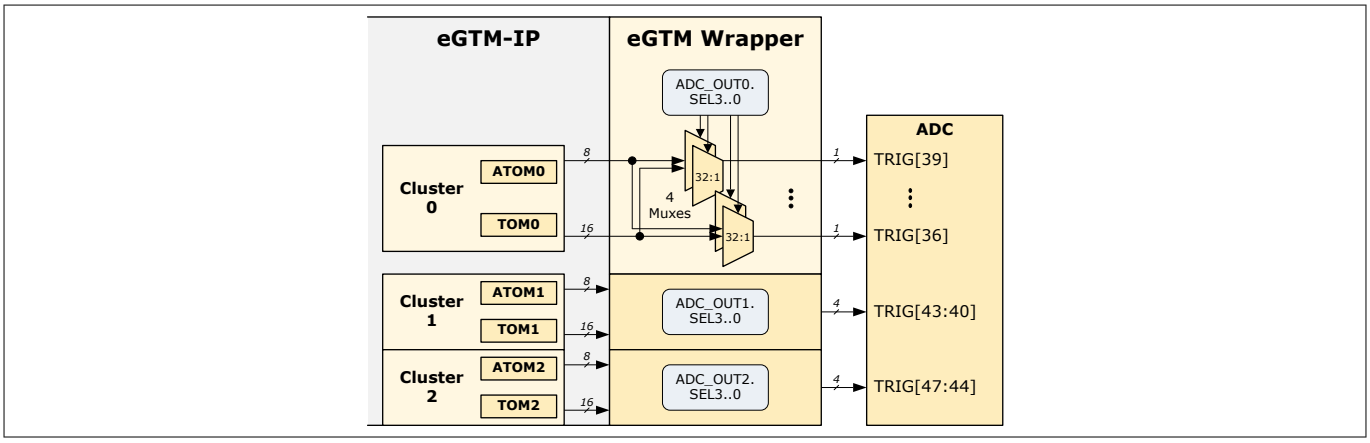


Figure 1130 eGTM outputs to ADC

eGTM to ADC output selection register n (n=0-2)

The bit fields in this register define which TOM/ATOM channel output is used as ADC trigger x.

ADC_OUTn (n=0-2) Offset address: 061700_H+n*4

eGTM to ADC output selection register n Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0			SEL3					0			SEL2				
r			rw					r			rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			SEL1					0			SEL0				
r			rw					r			rw				

Field	Bits	Type	Description
SELx (x=0-1)	8*x+4:8*x	rw	**eGTM to ADC trigger x output selection**
SELx (x=2-3)	8*x+4:8*x	rw	**eGTM to ADC trigger x output selection**
0	31:29, 23:21, 15:13, 7:5	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.3.2.6.3 Port connections

This section summarizes the port connections, which define the eGTM interface signals.

Table 2297 eGTM port connections overview

Description	Table
eGTM to GPIO connections	See Data Sheet and TOUTSEL registers

(table continues...)

Table 2297 (continued) eGTM port connections overview

GPIO to eGTM connections	See Data Sheet and TIMnINSEL registers
DTM_AUX Input connections	See Data Sheet and DTMAUX registers
eGTM ECLK (external clock) to GPIO connections	See Data Sheet and table below
GPIO availability by package	See Data Sheet - hints provided in TOUTSEL, in case of doubt Data Sheet is correct

eGTM output to port connections

The table below lists all connections between the eGTM output modules (TOMs/ATOMs/DTMs) and the ports.

Select the eGTM outputs by the TOUTSEL registers.

The eGTM outputs CMU_ECLK[2:0] are connected to dedicated ports.

Table 2298 eGTM External Clock to port mapping

eGTM External Clock	Output to PIN
CMU_ECLK0	P23.1
CMU_ECLK1	P32.4
CMU_ECLK2	P11.12/P16.14 this is dependent on the package variant

eGTM output to port control registers (TOUTSEL)

The outputs of the modules of eGTM (TOM, ATOM and DTM) are connected to several alternate functions of the port, selected by TOUTSEL.

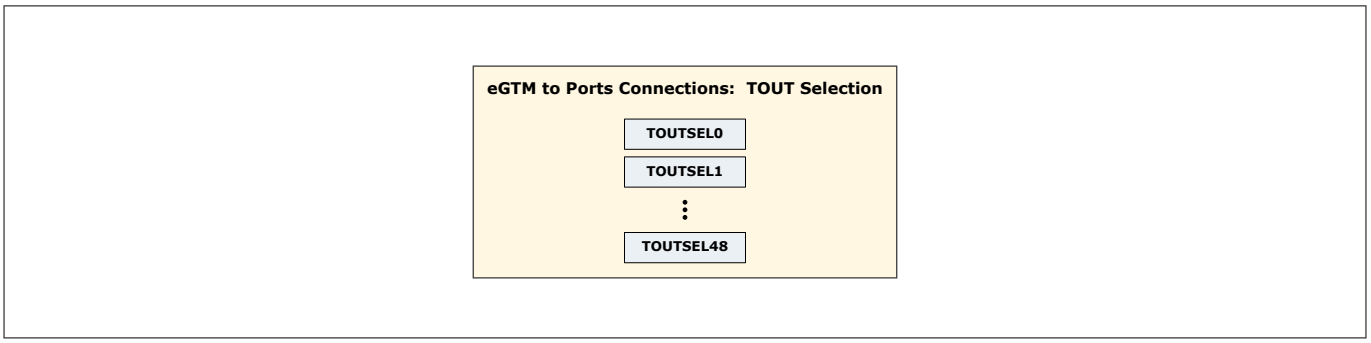


Figure 1131 TOUTSELn register overview

The setting inside this register, maps a eGTM resource to a certain port pin.

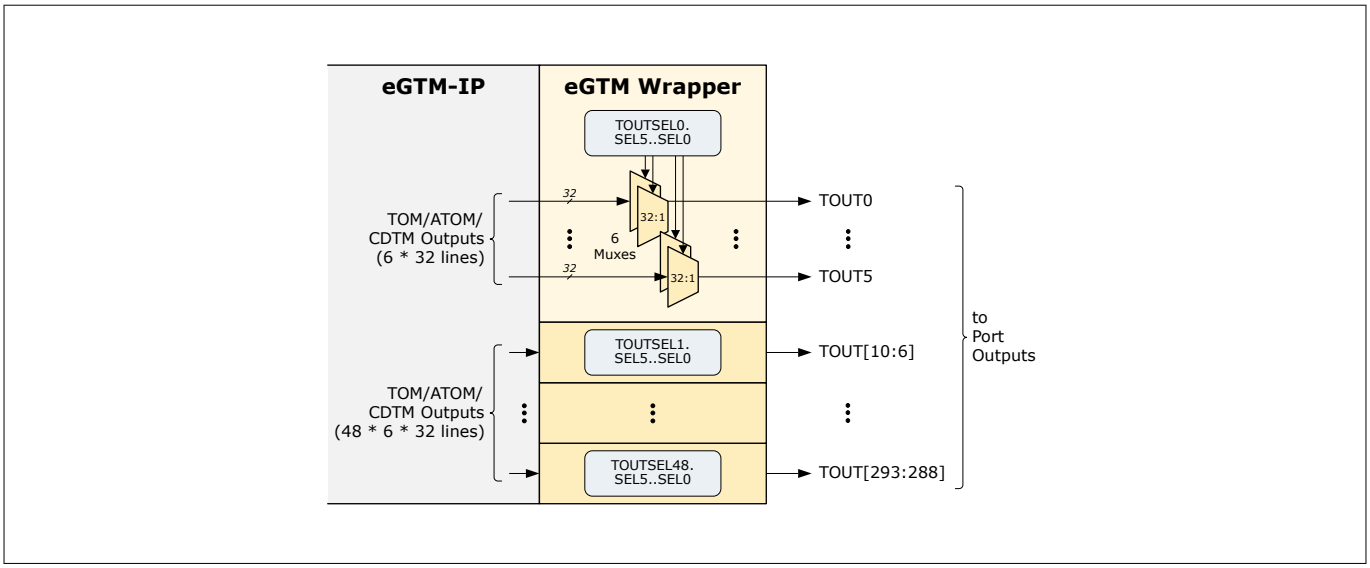


Figure 1132 eGTM outputs to ports

The register shown below, is showing the structure.

TOUT145 is not connected to an output. It is connected to an XTAL connection on-chip.

eGTM to Ports signal TOUT output selection register n (n=0-48)

The bit fields in this register define which TOM/ATOM channel output is used as port signal TOUT(n*6+x).

TOUTSELn (n=0-48) Offset address: 061100_H+n*4

eGTM to Ports signal TOUT output selection register n Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	SEL5					SEL4					SEL3				
r	rw					rw					rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	SEL2					SEL1					SEL0				
r	rw					rw					rw				

Field	Bits	Type	Description
SELx (x=0-2)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection**
SELx (x=3-5)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection**
0	15, 31	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

Port to eGTM connections - TIM inputs

Each of the TIM modules of the eGTM has an external signal input for each of its channels. These inputs are called TIMn_INy, with n being the number of the TIM module, and y the number of the channel within the TIM.

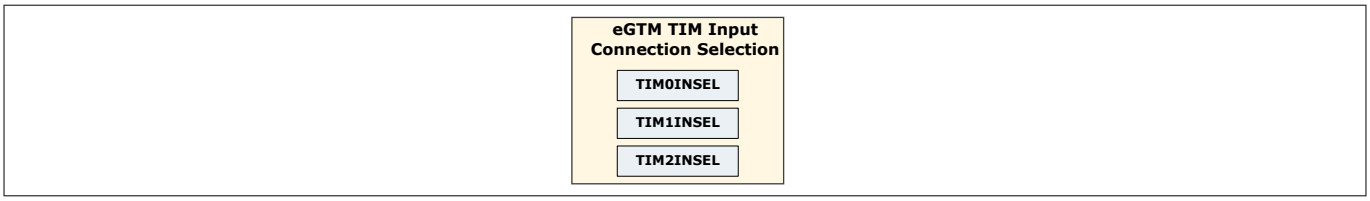


Figure 1133 TIMnINSEL register overview

To provide a high flexibility, the eGTM wrapper implements a 16:1 multiplexer for each of the TIMn_INy inputs. These multiplexers are controlled by the CHySEL bit-fields in the registers TIMnINSEL. See figure below for details.

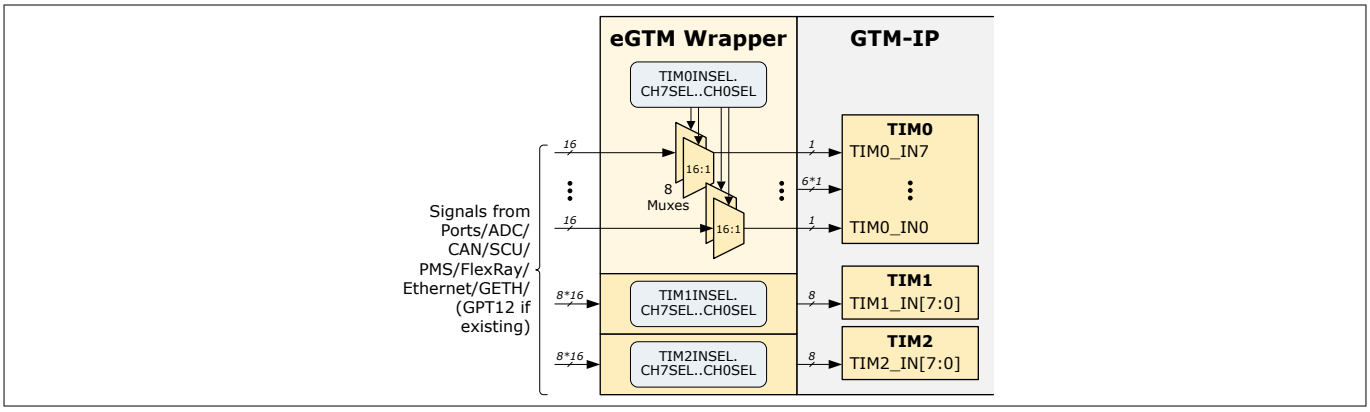


Figure 1134 Details on TIMnINSEL register structure

The external signals connected to the inputs of the multiplexers come from the port as well as from other on-chip modules such as the ADC, CAN(XL), SCU, PMS, FlexRay, GPT12, and (G/L)ETH. These signals provide status, interrupt, and synchronization information, which can then be processed by the eGTM.

The signals from the on-chip modules to the TIM module input multiplexers are shown in the following simplified overviews. Note that each input is named TIMn_y_z, with n = 0..2 (TIM module), y = 0..7 (TIMn channel), and z = 0..15 (multiplexer input number).

Note also that most of the signals are connected to more than one TIM input multiplexer, to increase the flexibility.

To provide more flexibility and to keep software compatibility to previous devices, a second group of TIMnINSEL registers has been introduced TIMnINSELB.

TIMnINSEL registers

The TIMnINSEL registers select inputs of the eGTM.

GPT12 to eGTM:

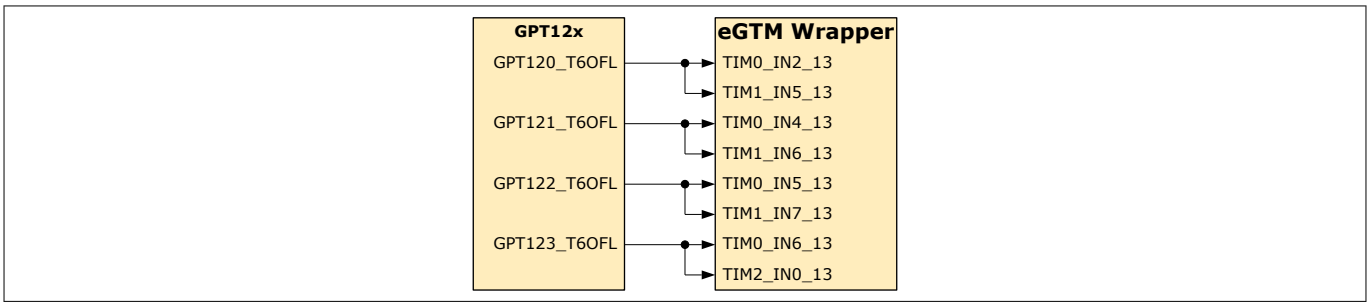


Figure 1135 GPT12 to TIMnINSEL

SCU to eGTM:

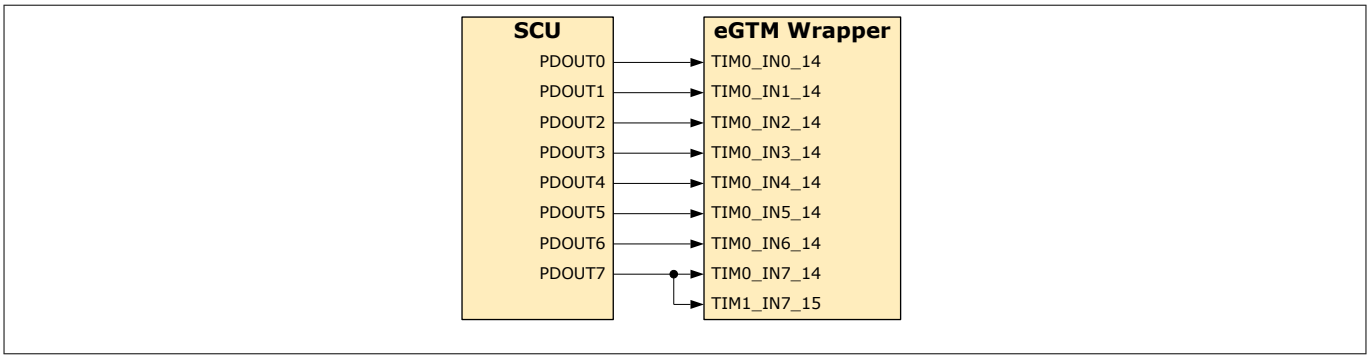


Figure 1136 SCU - PDOUT to TIMnINSEL

PMS to eGTM

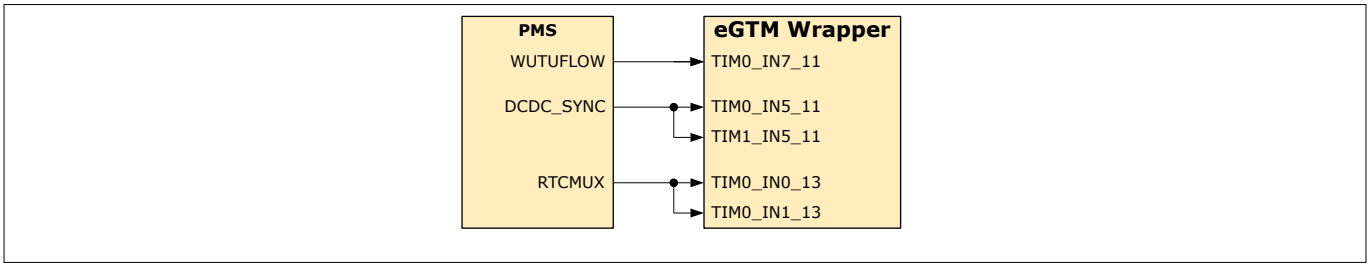


Figure 1137 PMS signals to TIMnINSEL

CAN to eGTM:

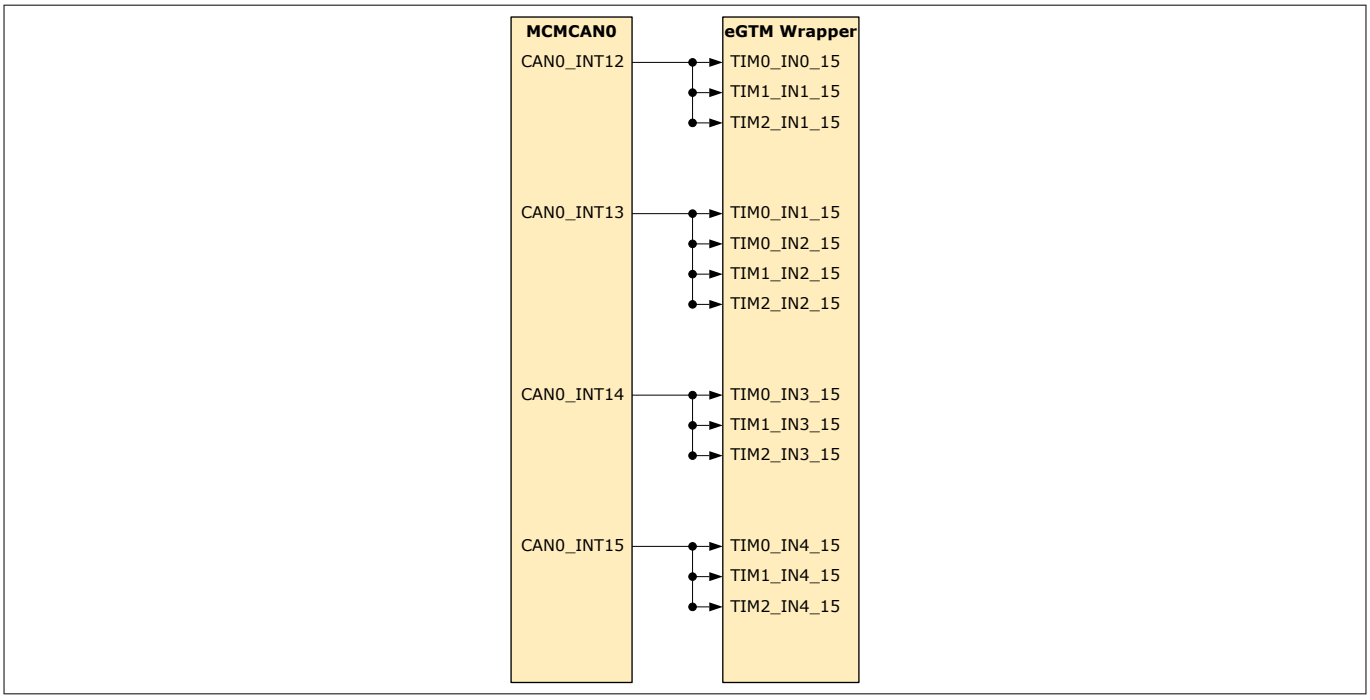


Figure 1138 CAN to TIMnINSEL

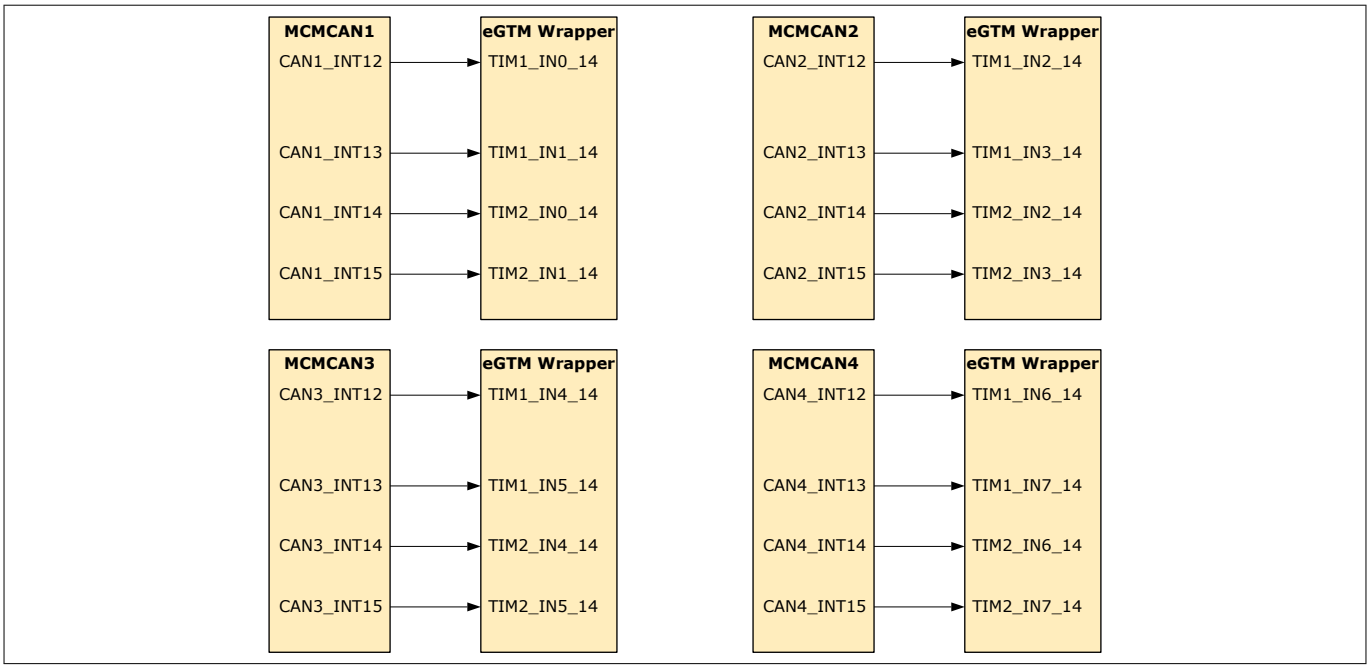


Figure 1139 CAN 1-4 to TIMnINSEL

CAN XL to eGTM:

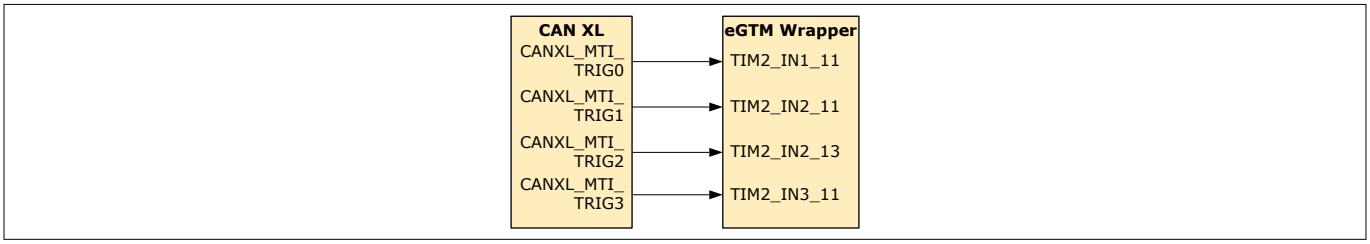


Figure 1140 CAN XL to eGTM

FlexRay to eGTM:

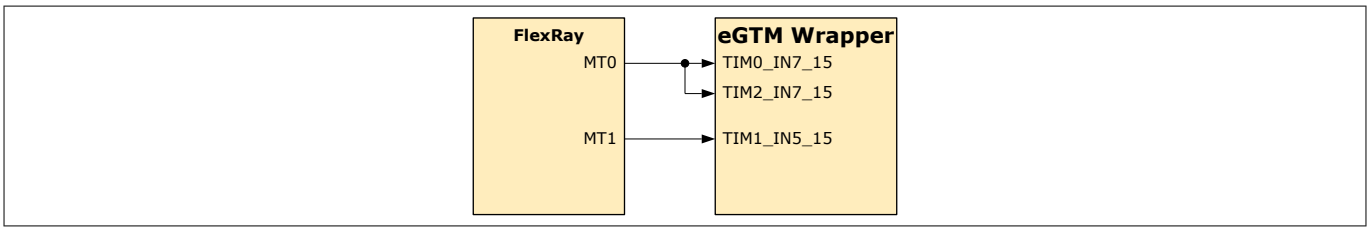


Figure 1141 FlexRay to TIMnINSEL

ADC to eGTM:

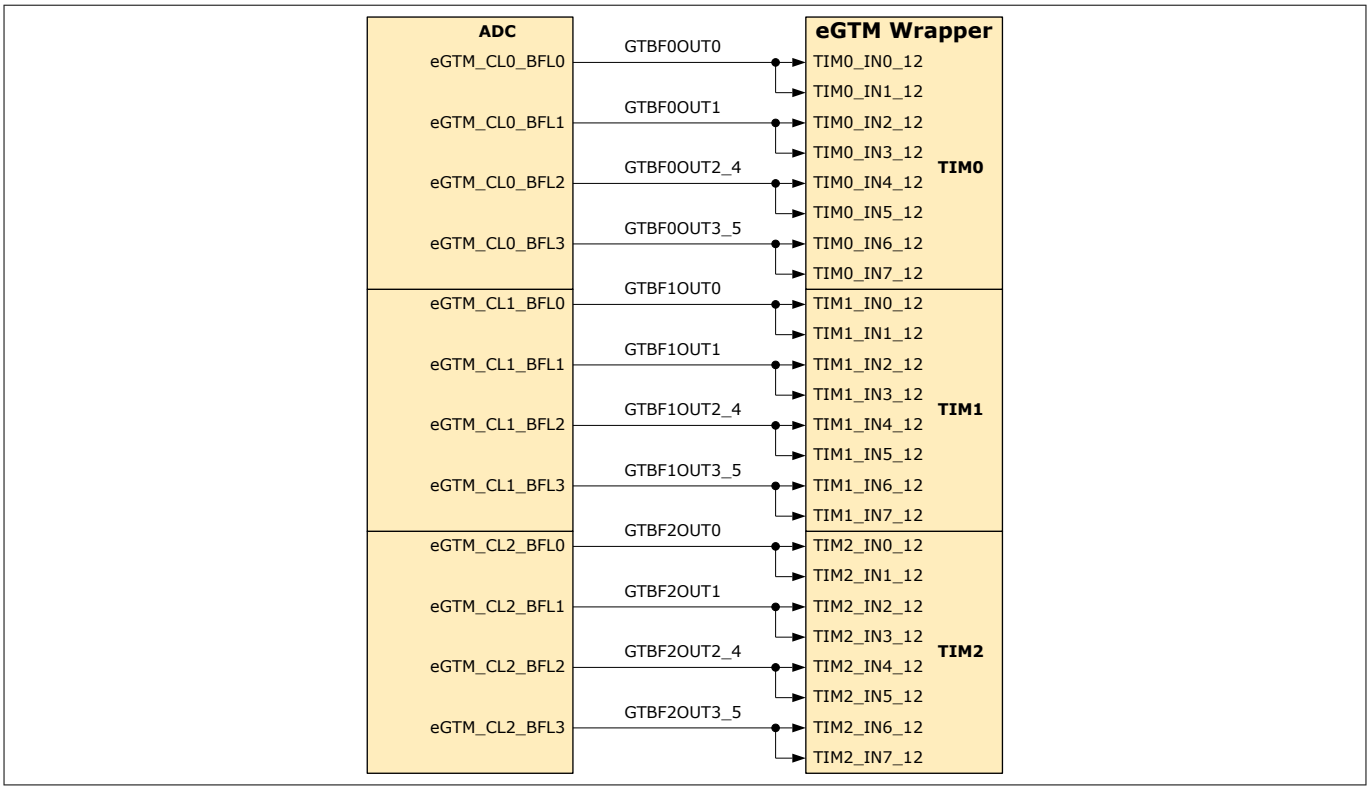


Figure 1142 ADC to TIMnINSEL

Gigabit Ethernet to eGTM:

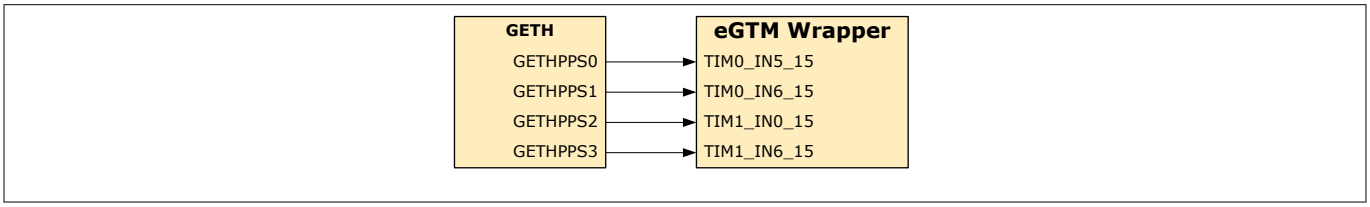


Figure 1143 Gigabit Ethernet to TIMnINSEL

Lite Ethernet to eGTM:

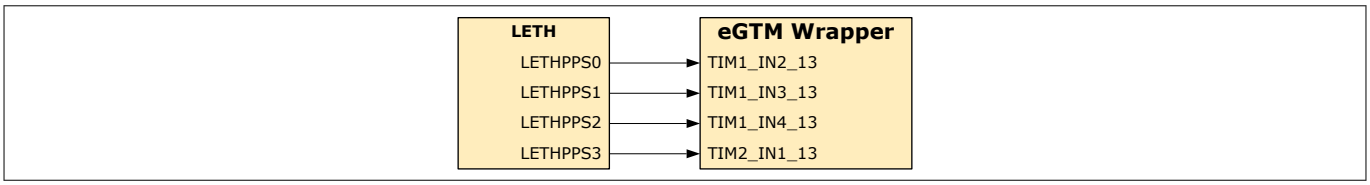


Figure 1144 Lite Ethernet to TIMnINSEL

TIMn input selection register (n=0-2)

TIMnINSEL (n=0-2) Offset address: 061000_H+n*4

TIMn input selection register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CH7SEL				CH6SEL				CH5SEL				CH4SEL			
rw				rw				rw				rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH3SEL				CH2SEL				CH1SEL				CH0SEL			
rw				rw				rw				rw			

Field	Bits	Type	Description
CHxSEL (x=0-7)	4*x+3:4*x	rw	**eGTM TIMn channel x input selection** This bit field selects which input is connected for TIMn channel x. The input is either derived from a port pad or from an on-chip module.

TIMnINSELB registers

The TIMn input selection register group B (n=0-2) registers select inputs of the eGTM.

The main purpose is the synchronization to ADC interrupts. These ADC interrupt service requests can be chosen in TIMn input selection register group B (n=0-2). The specific instance is programmed in the corresponding register Instance Selection for TIM[n] channel[i] (i=0-7;n=0-2). Please be aware, that the register TIMn input selection register group B (n=0-2) is dependent on its content.

For PDOUT only TIMn input selection register group B (n=0-2) needs to be configured. In this case Instance Selection for TIM[n] channel[i] (i=0-7;n=0-2) is not needed.

***Note**: This function is not available in TC4Dx, related registers are not existing.*

This function employs a streamlined two-stage multiplexer (MUX). In the first stage, a shared selection signal TIMn input selection register group B (n=0-2) simultaneously controls two parallel operations:

- 16:1 MUX that selects one of the predefined 16 ADC interrupt groups, producing a 16-bit intermediate output
- 16:1 MUX that selects one of the bit select signals configured in Instance Selection for TIM[n] channel[i] (i=0-7;n=0-2), generating a 4-bit control signal

In the second stage, the 4-bit signal acts as a selector to pinpoint a single bit from the 16-bit intermediate output.

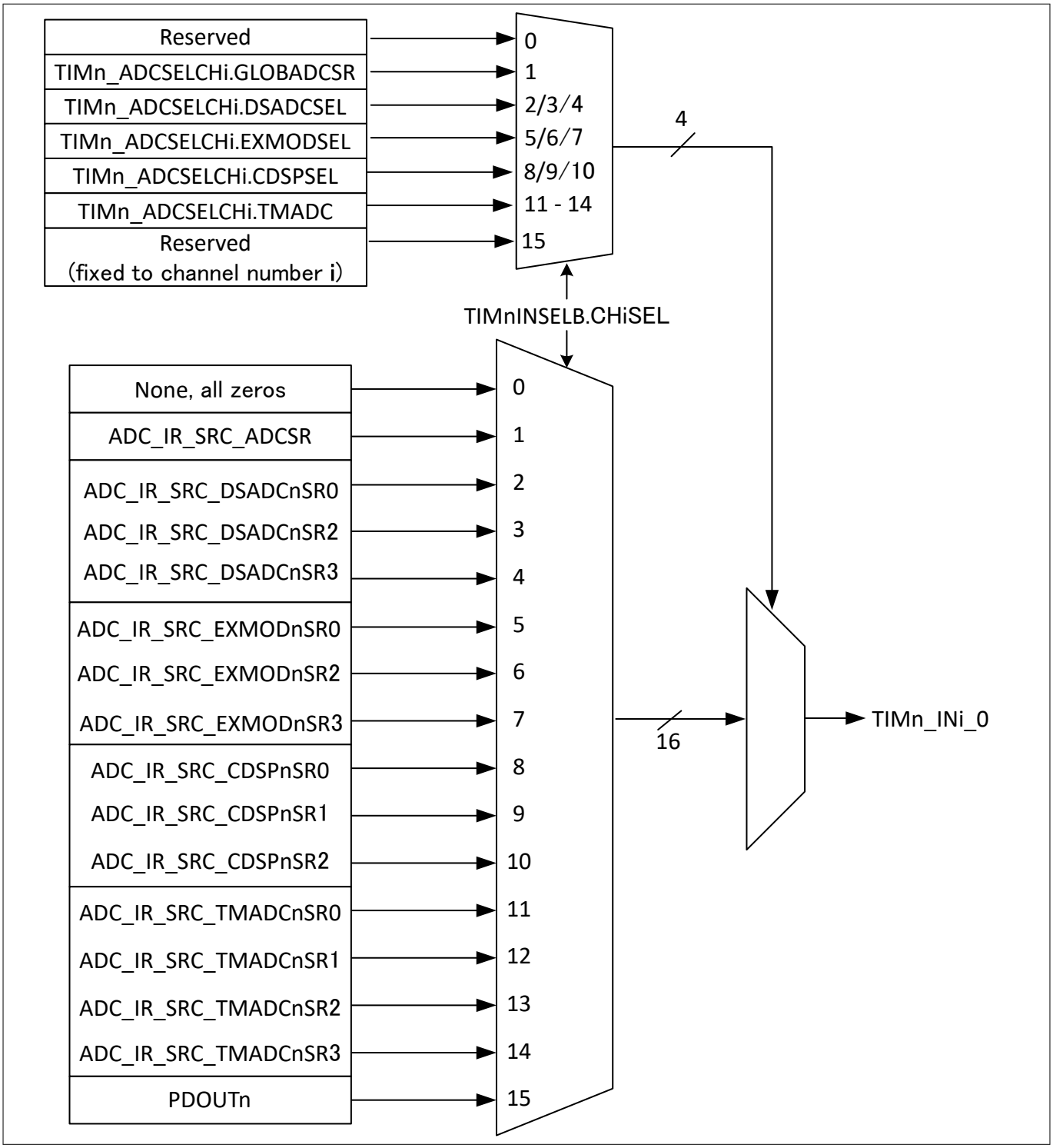


Figure 1145 Register structure of TIMnINSELB

TIMn input selection register group B (n=0-2)

TIMnINSELB (n=0-2) Offset address: 061050_H+n*4

TIMn input selection register group B Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CH7SEL				CH6SEL				CH5SEL				CH4SEL			
rw				rw				rw				rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH3SEL				CH2SEL				CH1SEL				CH0SEL			
rw				rw				rw				rw			

Field	Bits	Type	Description
CHxSEL (x=0-7)	4*x+3:4*x	rw	**eGTM TIMn channel x input selection group B** This bit field selects which input is connected for TIMn channel x (group B). The input is either derived from a port pad or from an on-chip module.

Instance Selection for TIM[n] channel[i] (i=0-7;n=0-2)

TIMn_ADCSELCHi (i=0-7;n=0-2) Offset address: $061B00_H+n*20_H+i*4$

Instance Selection for TIM[n] channel[i] Kernel Reset value: $XXXX\ XXXX_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															
r															

Field	Bits	Type	Description
0	31:0	r	**Reserved, must be written with 0, shall read 0.**

44.3.2.6.4 GPT12 connections

This feature only exists, if and only if a GPT12 is present on the device. To provide filtered inputs to GPT12, a connection from eGTM to GPT12 is implemented. The signal FOUT is available after the filter on each TIM channel (see eGTM TIM description FOUT signal).

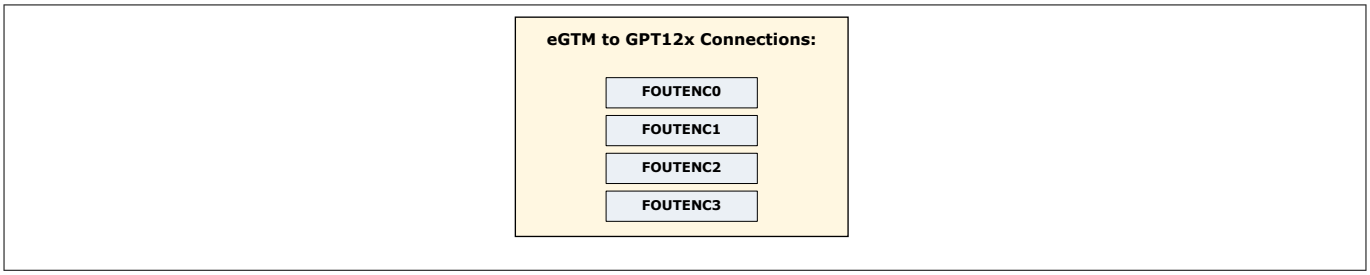


Figure 1146 Register structure FOUTENC

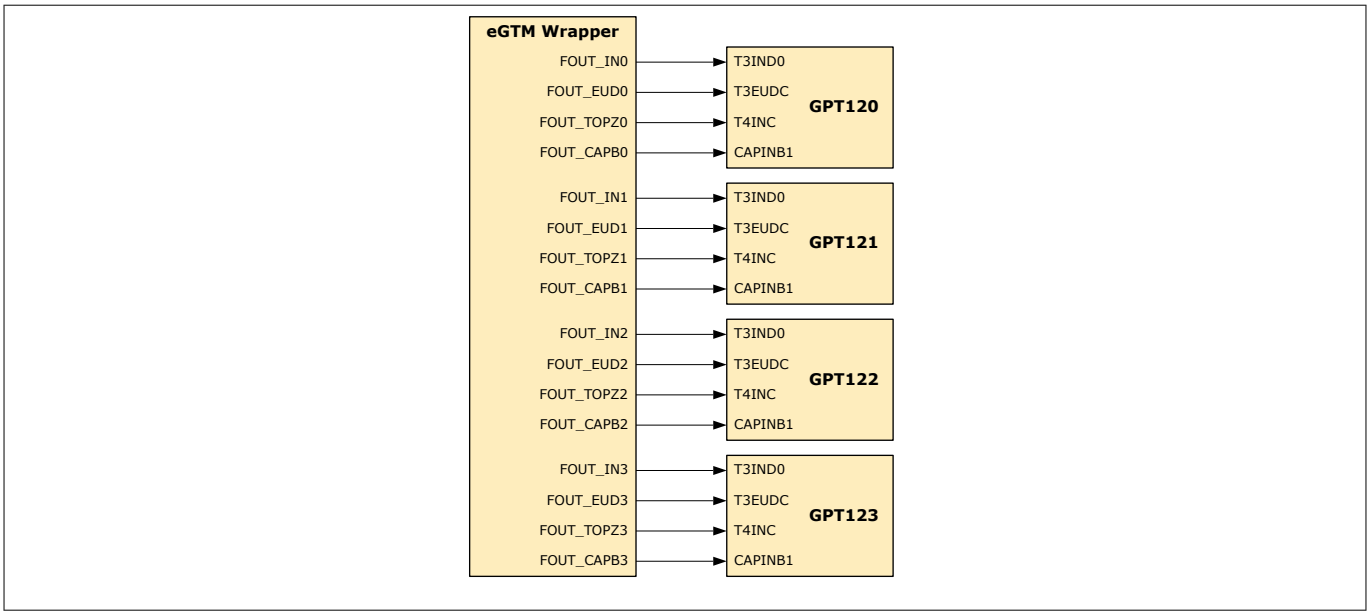


Figure 1147 Overview on eGTM FOUT signals connected to GPT12

The signals are provided to each GPT12 on the device for the two inputs associated with core timer T3 (T3IN, T3EUD) on their positions C, one input associated with auxiliary timer T4 (T4IN) on its position C, and additionally with the input pin CAPIN B. The CAPIN B (CAPINB1) on GPT12 is pre-multiplexed to switch to eGTM CAPIN B.

For each input signal from the eGTM the application can choose between all TIM clusters and TIM channels.

For safety applications it is possible to feed two GPT12 instances with the same data.

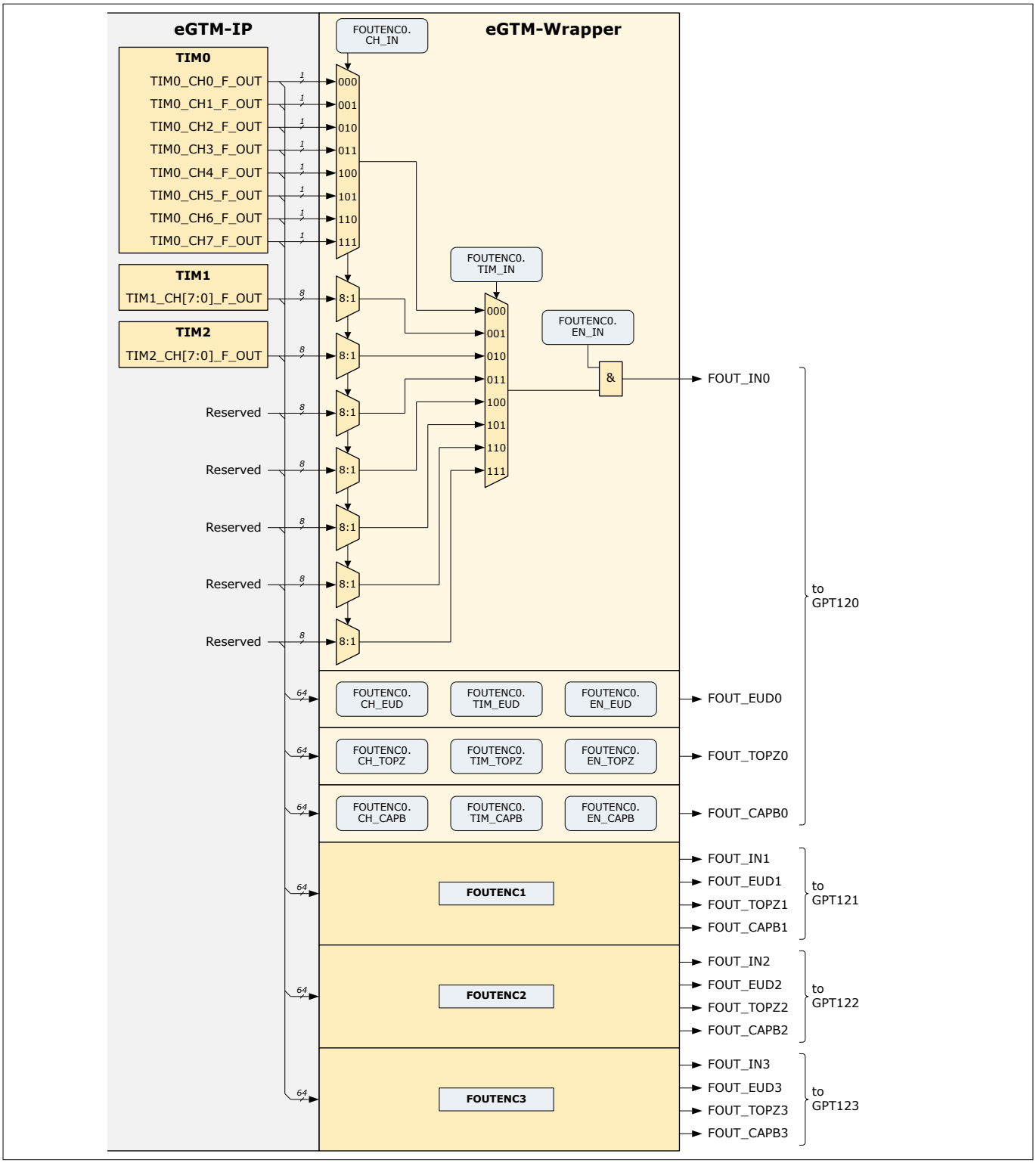


Figure 1148 Filtered inputs to GPT12

- FOUTENCx.CHSEL
  - Bit-field to choose the channel number
- FOUTENCx.TIMSEL
  - Bit-field to choose the TIM module cluster

***Note**: This feature can be controlled by the GPT12.*

The main aim of this interface is to provide a clean input to an encoder interface.

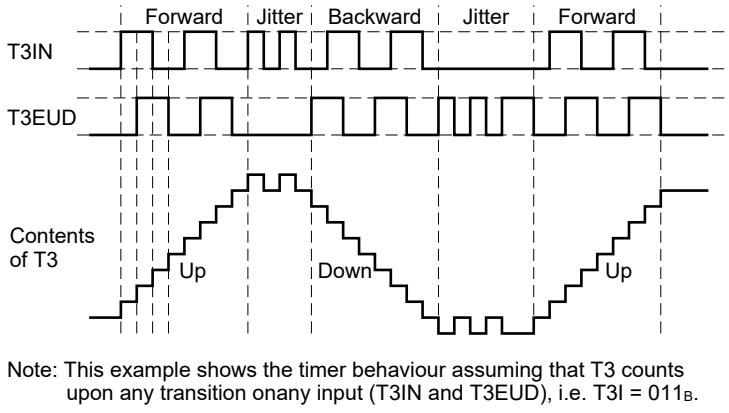


Figure 1149 Evaluation of incremental encoder signals, 2 count inputs

The connection is for T3IN, T3EUD, and for the corresponding TOPZ input.

In addition, the CAPIN B input receives the same type of filtered input. On the GPT12 side this is an option which can be selected through the input multiplexer. This option on CAPIN B is added on an additional multiplexer.

***Note**: TOPZ is connected to T4IN.*

FOUT Selection for Encoder Interface x, if GPT resources exist. (x=0-3)

FOUTENCx (x=0-3) Offset address: 061500_H+x*4

FOUT Selection for Encoder Interface x, if GPT resources exist. Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN_CAPB	CH_CAPB			0	TIM_CAPB			EN_TOPZ	CH_TOPZ			0	TIM_TOPZ		
rw	rw			r	rw			rw	rw			r	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN_EUD	CH_EUD			0	TIM_EUD			EN_IN	CH_IN			0	TIM_IN		
rw	rw			r	rw			rw	rw			r	rw		

Field	Bits	Type	Description
TIM_IN	2:0	rw	**TIM module selection for encoder interface input IN of GPT12x** TIM out of corresponding cluster. No function if TIM number does not exist. 000_B **TIM0**: TIM of cluster 0 selected for IN input of GPT12x 001_B **TIM1**: TIM of cluster 1 selected for IN input of GPT12x 010_B **TIM2**: TIM of cluster 2 selected for IN input of GPT12x 011_B **Reserved**: Reserved, do not use 100_B **Reserved**: Reserved, do not use 101_B **Reserved**: Reserved, do not use 110_B **Reserved**: Reserved, do not use 111_B **Reserved**: Reserved, do not use
CH_IN	6:4	rw	**TIM channel selection for encoder interface input IN of GPT12x** 000_B **CH0**: Channel 0 within TIM selected for IN input of GPT12x 001_B **CH1**: Channel 1 within TIM selected for IN input of GPT12x 010_B **CH2**: Channel 2 within TIM selected for IN input of GPT12x 011_B **CH3**: Channel 3 within TIM selected for IN input of GPT12x 100_B **CH4**: Channel 4 within TIM selected for IN input of GPT12x 101_B **CH5**: Channel 5 within TIM selected for IN input of GPT12x 110_B **CH6**: Channel 6 within TIM selected for IN input of GPT12x 111_B **CH7**: Channel 7 within TIM selected for IN input of GPT12x
EN_IN	7	rw	**Enable connection to encoder interface input IN of GPT12x** 0_B **0**: Disable the connection of eGTM to GPT12x_IN 1_B **1**: Enable the connection of eGTM to GPT12x_IN
TIM_EUD	10:8	rw	**TIM module selection for encoder interface input EUD of GPT12x** 000_B **TIM0**: TIM of cluster 0 selected for EUD input of GPT12x 001_B **TIM1**: TIM of cluster 1 selected for EUD input of GPT12x 010_B **TIM2**: TIM of cluster 2 selected for EUD input of GPT12x 011_B **Reserved**: Reserved, must not be used. 100_B **Reserved**: Reserved, must not be used. 101_B **Reserved**: Reserved, must not be used. 110_B **Reserved**: Reserved, must not be used. 111_B **Reserved**: Reserved, must not be used.
CH_EUD	14:12	rw	**TIM channel selection for encoder interface input EUD of GPT12x** 000_B **CH0**: Channel 0 within TIM selected for EUD input of GPT12x 001_B **CH1**: Channel 1 within TIM selected for EUD input of GPT12x 010_B **CH2**: Channel 2 within TIM selected for EUD input of GPT12x 011_B **CH3**: Channel 3 within TIM selected for EUD input of GPT12x 100_B **CH4**: Channel 4 within TIM selected for EUD input of GPT12x 101_B **CH5**: Channel 5 within TIM selected for EUD input of GPT12x 110_B **CH6**: Channel 6 within TIM selected for EUD input of GPT12x 111_B **CH7**: Channel 7 within TIM selected for EUD input of GPT12x

(table continues...)

(continued)

Field	Bits	Type	Description
EN_EUD	15	rw	**Enable connection to encoder interface input EUD of GPT12x** 0_B **0**: Disable the connection of eGTM to GPT12x_EUD 1_B **1**: Enable the connection of eGTM to GPT12x_EUD
TIM_TOPZ	18:16	rw	**TIM module selection for encoder interface input TOP Zero (T4IN) of GPT12x** 000_B **TIM0**: TIM of cluster 0 selected for TOPZ input of GPT12x 001_B **TIM1**: TIM of cluster 1 selected for TOPZ input of GPT12x 010_B **TIM2**: TIM of cluster 2 selected for TOPZ input of GPT12x 011_B **Reserved**: Reserved, do not use 100_B **Reserved**: Reserved, do not use 101_B **Reserved**: Reserved, do not use 110_B **Reserved**: Reserved, do not use 111_B **Reserved**: Reserved, do not use
CH_TOPZ	22:20	rw	**TIM channel selection for encoder interface input TOP Zero (T4IN) of GPT12x** 000_B **CH0**: Channel 0 within TIM selected for TOPZ input of GPT12x 001_B **CH1**: Channel 1 within TIM selected for TOPZ input of GPT12x 010_B **CH2**: Channel 2 within TIM selected for TOPZ input of GPT12x 011_B **CH3**: Channel 3 within TIM selected for TOPZ input of GPT12x 100_B **CH4**: Channel 4 within TIM selected for TOPZ input of GPT12x 101_B **CH5**: Channel 5 within TIM selected for TOPZ input of GPT12x 110_B **CH6**: Channel 6 within TIM selected for TOPZ input of GPT12x 111_B **CH7**: Channel 7 within TIM selected for TOPZ input of GPT12x
EN_TOPZ	23	rw	**Enable connection to encoder interface input TOP Zero (T4IN) of GPT12x** 0_B **0**: Disable the connection of eGTM to GPT12x_TOPZ (T4IN) 1_B **1**: Enable the connection of eGTM to GPT12x_TOPZ (T4IN)
TIM_CAPB	26:24	rw	**TIM module selection for CAPINB input of GPT12x** 000_B **TIM0**: TIM of cluster 0 selected for CAPINB input of GPT12x 001_B **TIM1**: TIM of cluster 1 selected for CAPINB input of GPT12x 010_B **TIM2**: TIM of cluster 2 selected for CAPINB input of GPT12x 011_B **Reserved**: Reserved, do not use 100_B **Reserved**: Reserved, do not use 101_B **Reserved**: Reserved, do not use 110_B **Reserved**: Reserved, do not use 111_B **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
CH_CAPB	30:28	rw	**TIM channel selection for CAPINB input of GPT12x** 000_B **CH0**: Channel 0 within TIM selected for CAPINB input of GPT12x 001_B **CH1**: Channel 1 within TIM selected for CAPINB input of GPT12x 010_B **CH2**: Channel 2 within TIM selected for CAPINB input of GPT12x 011_B **CH3**: Channel 3 within TIM selected for CAPINB input of GPT12x 100_B **CH4**: Channel 4 within TIM selected for CAPINB input of GPT12x 101_B **CH5**: Channel 5 within TIM selected for CAPINB input of GPT12x 110_B **CH6**: Channel 6 within TIM selected for CAPINB input of GPT12x 111_B **CH7**: Channel 7 within TIM selected for CAPINB input of GPT12x
EN_CAPB	31	rw	**Enable connection to CAPINB input of GPT12x** 0_B **0**: Disable the connection of eGTM to GPT12x_CAPINB 1_B **1**: Enable the connection of eGTM to GPT12x_CAPINB
0	3, 11, 19, 27	r	**Reserved, do not use** Read as 0, shall be written with 0.

44.3.2.6.5 SCU interrupt trigger

Outputs of ATOM or TOM can generate SCU interrupts.

The SCU trigger have the same structure as the ADC trigger. Every TOM or ATOM can be mapped to one of the four SCU interrupts for the eGTM.

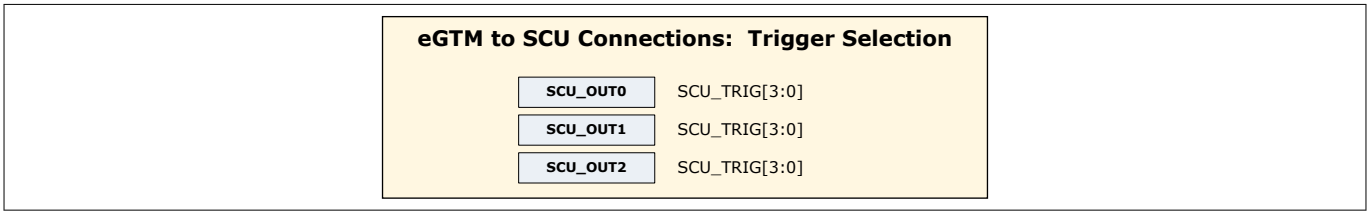


Figure 1150 SCU trigger

Together with the SCU connections mapped to TIMnINSEL, it is possible to have synchronization mechanism, either between different clusters of the eGTM or among eGTM and GTM (if available in this device).

The multiplexing scheme is described by the following figure

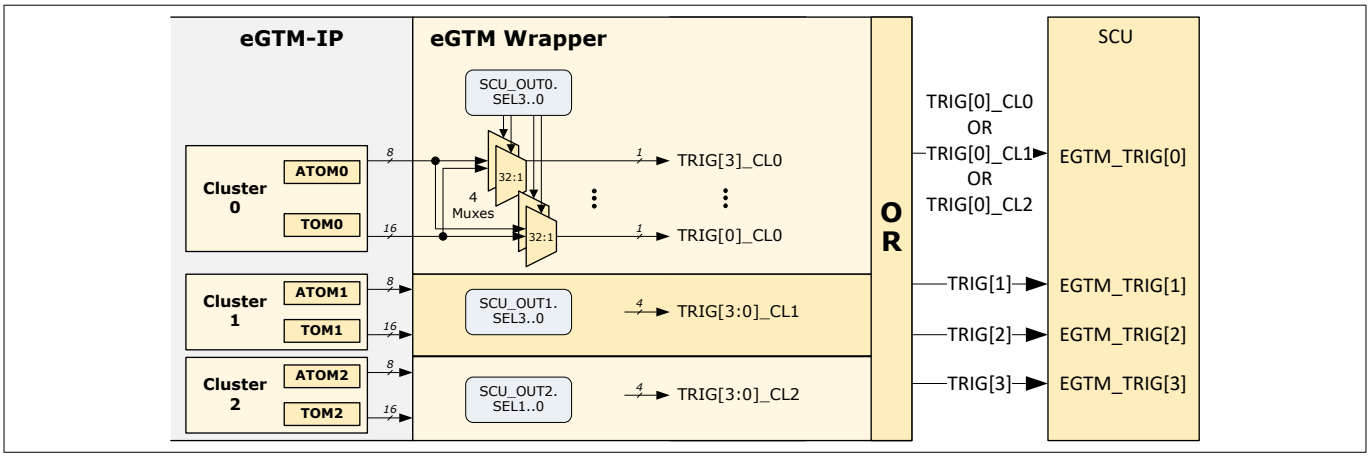


Figure 1151 Mapping of trigger outputs to the SCU

***Note**: TRIG0 of every cluster is ORed, the same applies to TRIG1, TRIG2, ..*

The register layout is shown in the next paragraph.

eGTM to SCU output selection register n eGTM_SCU_TRIG(n) (n=0-2)

The bit fields in this register define which TOM/ATOM channel output is used as SCU trigger x.

SCU_OUTn (n=0-2) Offset address: 061800_H+n*4

eGTM to SCU output selection register n eGTM_SCU_TRIG(n) Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0			SEL3					0			SEL2				
r			rw					r			rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			SEL1					0			SEL0				
r			rw					r			rw				

Field	Bits	Type	Description
SELx (x=0-1)	8*x+4:8*x	rw	**eGTM to SCU trigger x output selection eGTM_SCU_TRIG(x)**
SELx (x=2-3)	8*x+4:8*x	rw	**eGTM to SCU trigger x output selection eGTM_SCU_TRIG(x)**
0	15:13, 7:5, 31:29, 23:21	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.3.2.6.6 DTMAUX - AUX_IN connections

The registers for DTMAUX configure the eGTM inputs of (e)GTM_CDTM[i]_DTM[d]_AUX_IN[0:0] and (e)GTM_CDTM[i]_DTM[d]_AUX_IN[1:1]. These inputs provide for example the possibility to stop an electric drive in case of an emergency stop (also called TRAP or HALT). An inactive level can be defined inside the DTMAUX configuration.

The Cluster n DTMAUX input y selection register (y=0-1;n=0) registers provide the input configuration for (e)GTM_CDTM[i]_DTM[d]_AUX_IN[0:0] and (e)GTM_CDTM[i]_DTM[d]_AUX_IN[1:1].

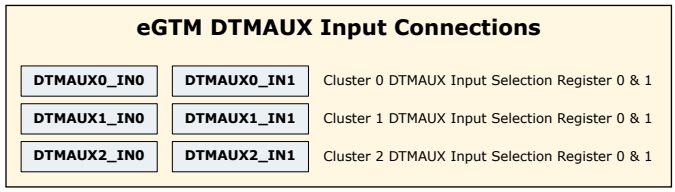


Figure 1152 DTMAUX register overview

Each of the CLSi_CDTM_DTMd modules of the eGTM has two external auxiliary signal inputs. These inputs are called (e)GTM_CDTM[i]_DTM[d]_AUX_IN with i being the number of the CDTM module, d the number of the DTM block within the CDTM, and y the number of the auxiliary input.

To provide a high flexibility in selecting the desired input signal, the eGTM wrapper implements 8:1 multiplexers for each of the auxiliary inputs. These multiplexers are controlled by the SELx bit-fields in the registers DTMAUXn_INy. See the figures below for details.

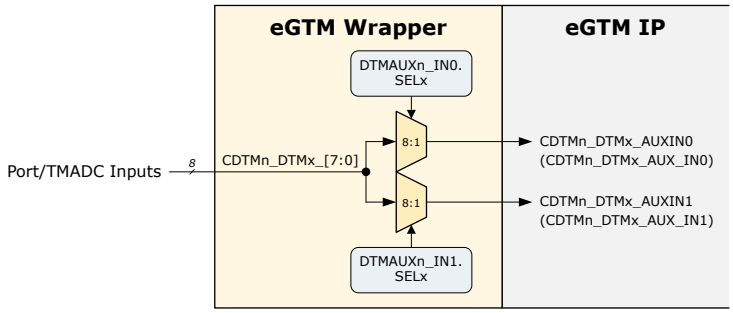


Figure 1153 DTMAUX input multiplexing of IN

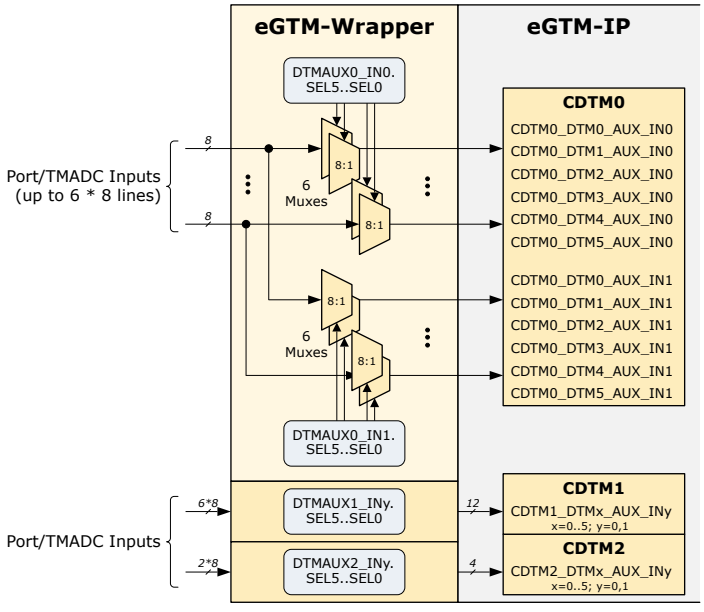


Figure 1154 Register structure of DTMAUX

***Note**: The two multiplexers for the two auxiliary inputs receive the same set of selectable signal inputs, however, the selection is independent for each auxiliary input, as each multiplexer has its own selection control.*

The layout of the multiplexer control registers DTMAUXn_INy is shown below.

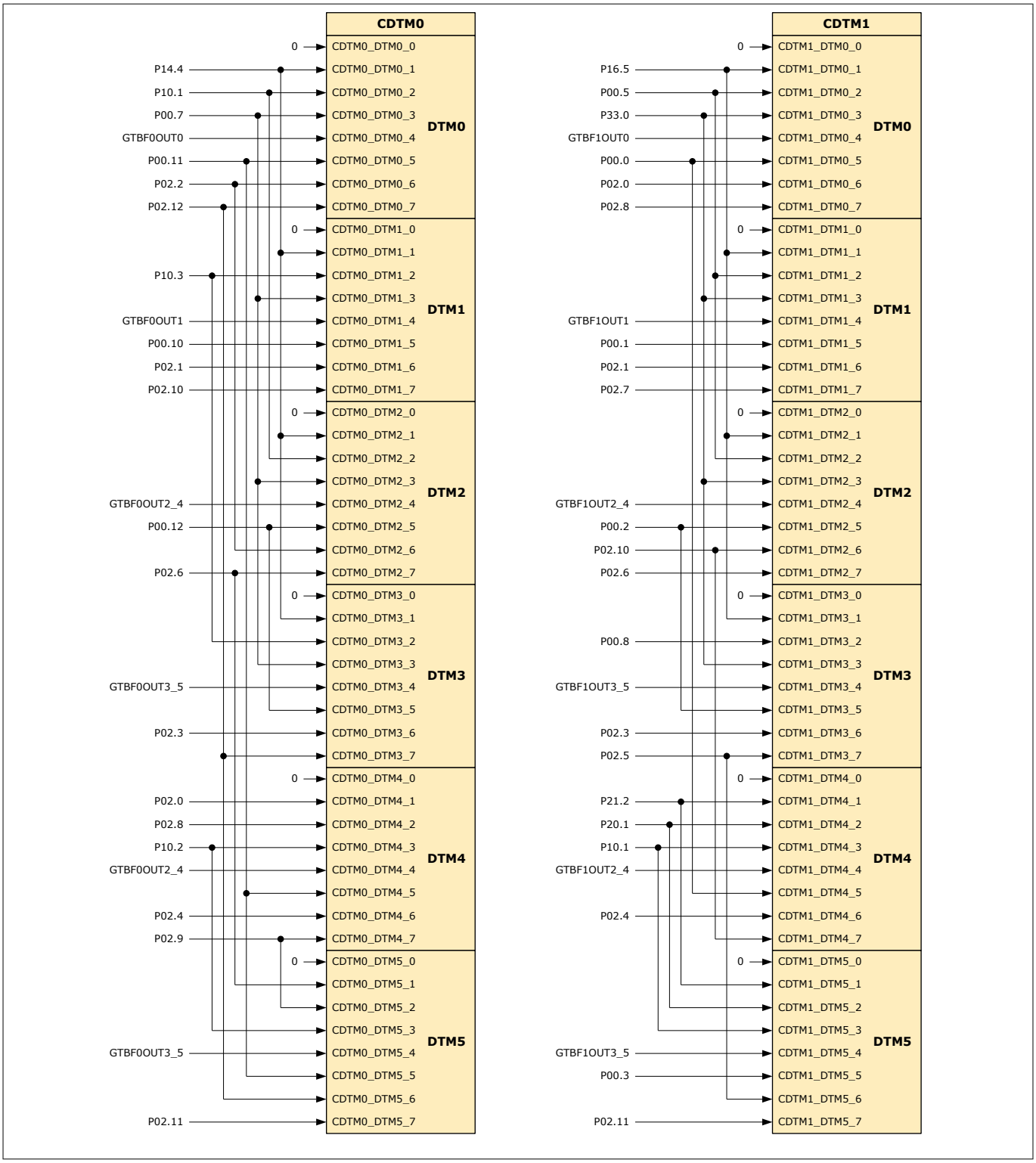


Figure 1155 Detailed overview on DTMAUX register settings 0 and 1

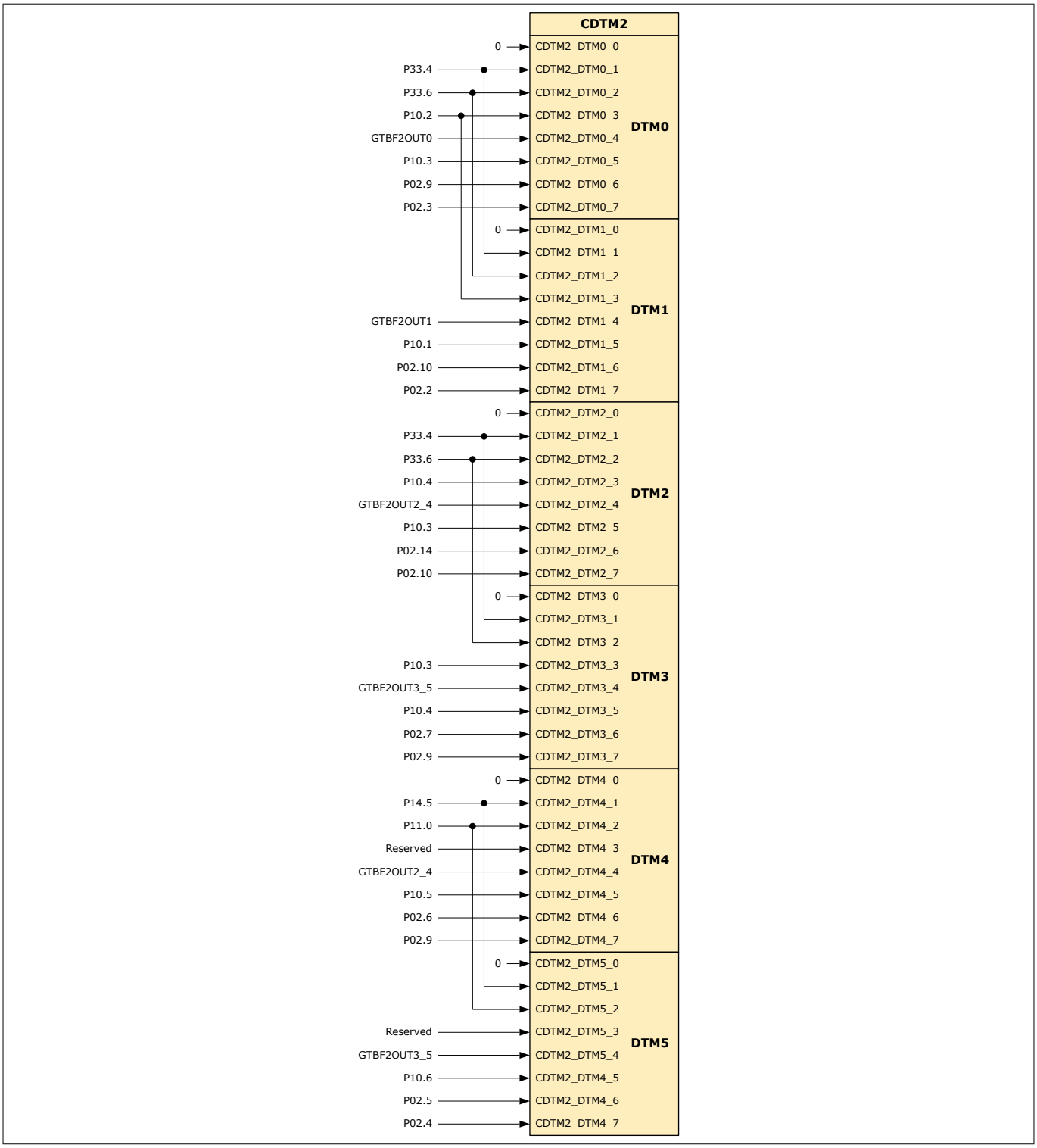


Figure 1156 Detailed overview on DTMAUX register settings 2

Cluster n DTMAUX input y selection register (y=0-1;n=0-2)

DTMAUXn_INy (n=0-2;y=0-1) Offset address: 061780_H+n*8+y*4

Cluster n DTMAUX input y selection register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								0	SEL5			0	SEL4		
r								r	rw			r	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	SEL3			0	SEL2			0	SEL1			0	SEL0		
r	rw			r	rw			r	rw			r	rw		

Field	Bits	Type	Description
SELx (x=0-5)	4*x+2:4*x	rw	**Input selection for CDTMn_DTMx_AUXINy**
0	23, 19, 15, 11, 7, 3, 31:24	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.3.2.6.7 LCDCDC (PMS) interface connection

The LCDCDC interface, is used to control an LC DC/DC based regulator. An input and an output to a timer is required to control the regulator.

Input

The input is selected by the TIMnINSEL registers.

The signal PMS_DCDC_SYNC is used as input and is available on the TIMnINSEL register.

- TIM0INSEL.CH5SEL
- TIM1INSEL.CH5SEL

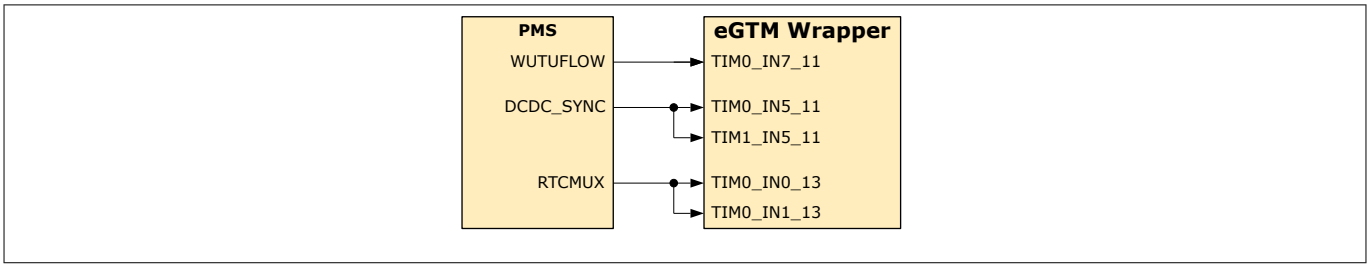


Figure 1157 LCDCDC connection with focus on (PMS_)DCDC_SYNC

There is a direct connection between PMS and eGTM configurable by the TIMnINSEL register, which is called PMS_DCDC_SYNC.

Output

The register LCDCDC_OUT selects an output. Several outputs can control the power supply.

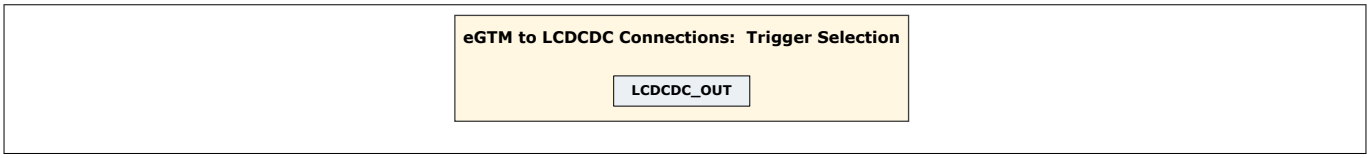


Figure 1158 Register overview

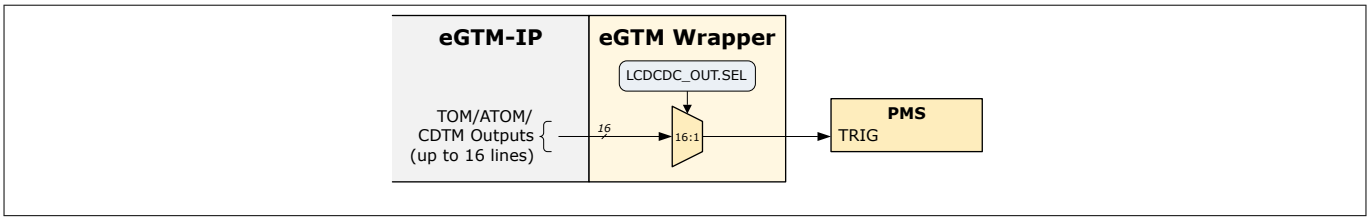


Figure 1159 Detailed structure of LCDCDC_OUT

LCDCDC related registers are: TIMnINSEL for cluster 0 & 1 and LCDCDC_OUT.

eGTM to LCDCDC output trigger selection register

The bit field in this register defines which TOM/ATOM channel output is used as LCDCDC trigger.

LCDCDC_OUT Offset address: 061420_H

eGTM to LCDCDC output trigger selection register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												SEL			
r												rw			

Field	Bits	Type	Description
SEL	3:0	rw	**eGTM to LCDCDC trigger output selection**
0	31:4	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.3.2.6.8 eGTM connections with communication modules

This paragraph describes the connections between eGTM and communication modules. Connections from communication modules to the eGTM are all handled by TIMnINSEL.

Connections to communication modules

The eGTM has connections to communication modules to trigger actions, or to measure a communication.

eGTM to CAN connection

Each CAN module receives a trigger input from the eGTM.

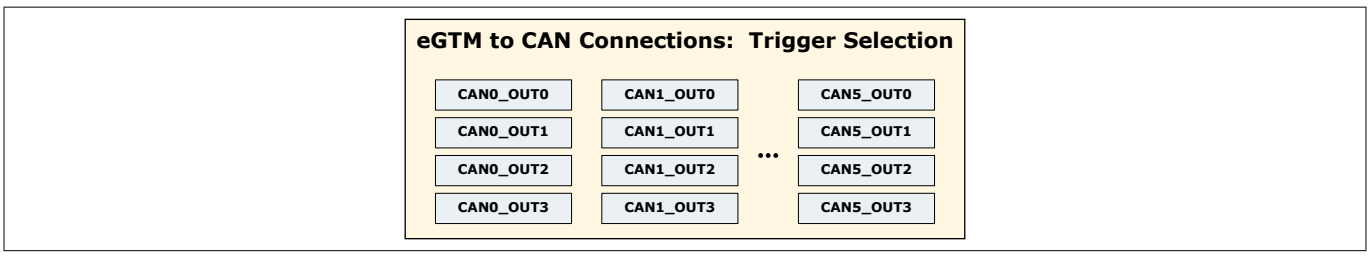


Figure 1160 CANn_OUT overview

- For each CAN node, one separate register called CANn_OUTx, is implemented
  - Index n is the CAN module number
  - Index x is the CAN node number
- The 4-bit-field SEL selects 1 out of 16 possible eGTM output lines as CANn_TRIGx trigger output signal

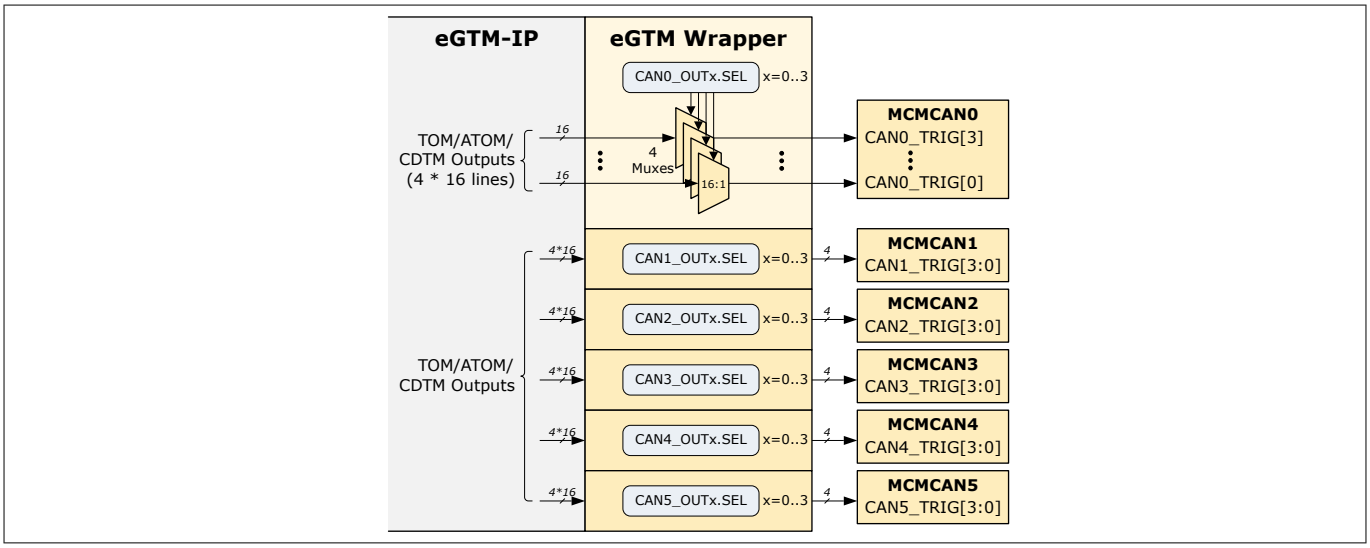


Figure 1161 eGTM to CAN trigger selection and connection

eGTM to CANn output selection register x. n*4+ x<24 , if interface exists. (x=0-3;n=0-5)

The bit field in this register defines which TOM/ATOM channel output is used as CANn node trigger x.

CANn_OUTx (n=0-5;x=0-3)

Offset address: 061480_H+n*10_H+x*4

eGTM to CANn output selection register x. n*4+ x<24 , if interface exists.

Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												SEL			
r												rw			

Field	Bits	Type	Description
SEL	3:0	rw	**eGTM to CANn trigger x output selection**
0	31:4	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

CAN to eGTM connection

The CAN interrupts 12 to 15 are connected to the TIM inputs of the eGTM.

- CAN0 interrupt lines 12, 14 and 15 are each connected to 3 TIM inputs and interrupt line 13 is connected to 4 TIM inputs (mapped to bit 15 of TIM)
- CAN1 - CAN4 interrupt lines are each connected to 1 TIM input (mapped to bit 14 of TIM)
- TIM clusters 0 to 2 are used for these connections

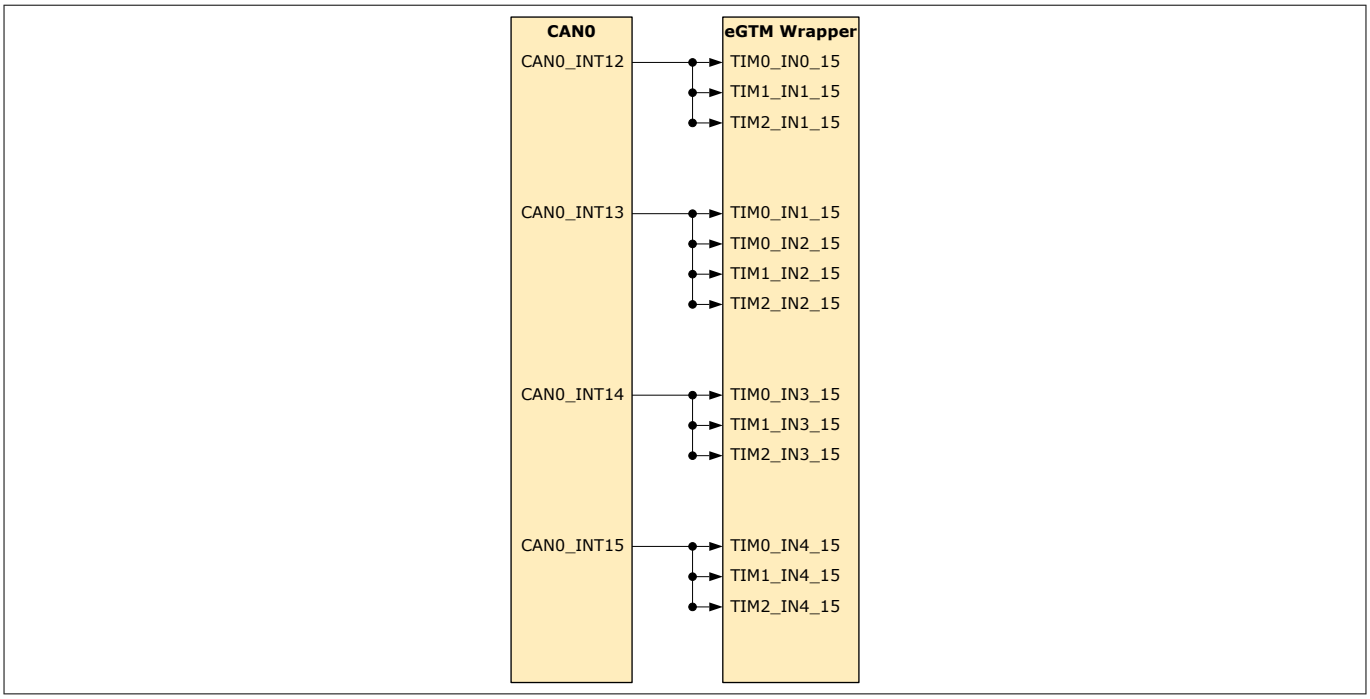


Figure 1162 CAN0 to eGTM connections

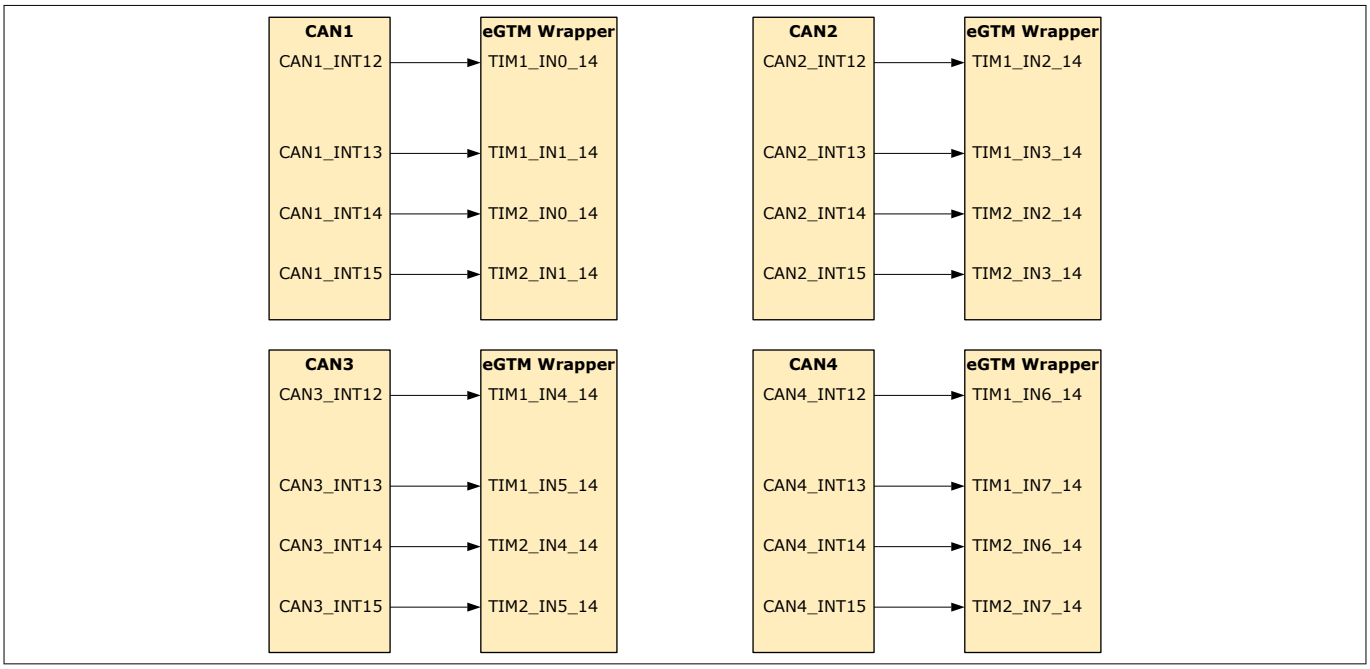


Figure 1163 CAN1-4 to eGTM connections

In addition connections from CAN XL to TIMnINSEL exist.

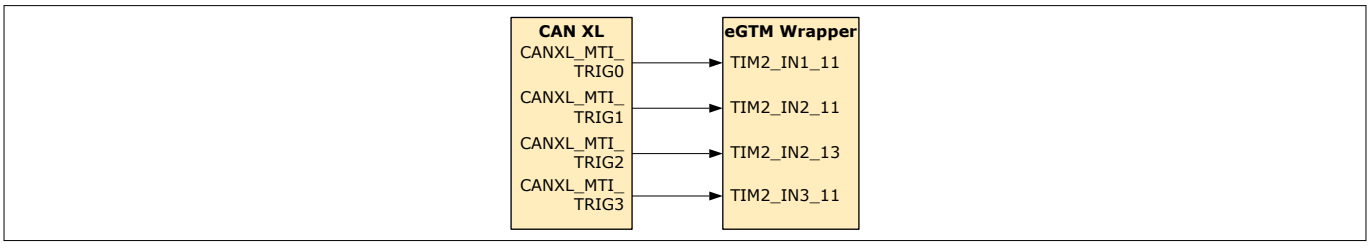


Figure 1164 CAN XL to eGTM connections

eGTM to ASCLIN connection

This connection enables the application to trigger an action on the ASCLIN module through a timer.

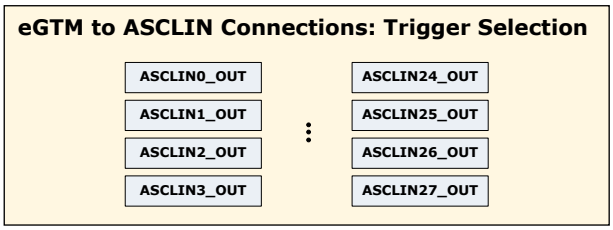


Figure 1165 Registers ASCLINn_OUT overview

- For each ASCLIN module one separate register called ASCLINn_OUT is implemented
  - Index n corresponds to the ASCLIN module number
  - The 5-bit-field SEL selects one out of 32 possible eGTM output lines as ASCLINn_TRIG trigger output signal

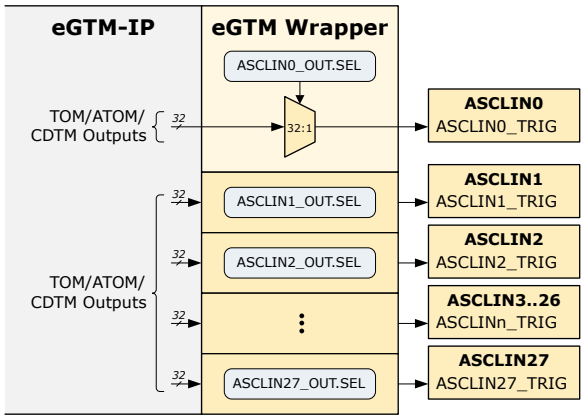


Figure 1166 Details on ASCLINn_OUT registers

eGTM to ASCLINn output selection register. n<28 , if interface exists. (n=0-27)

The bit field in this register defines which TOM/ATOM channel output is used as ASCLINn trigger.

ASCLINn_OUT (n=0-27)

Offset address: 061600_H+n*4

eGTM to ASCLINn output selection register. n<28 , if interface exists.

Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0											SEL				
r											rw				

Field	Bits	Type	Description
SEL	4:0	rw	**eGTM to ASCLINn trigger output selection**

(table continues...)

(continued)

Field	Bits	Type	Description
0	31:5	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

eGTM to QSPI connection

This connection enables the application to trigger an action on the QSPI module through a timer.

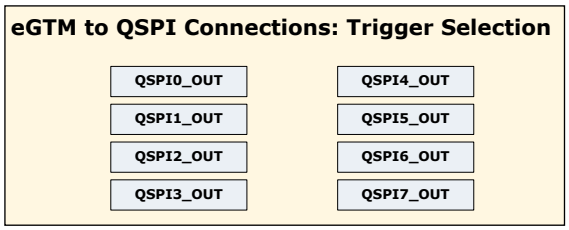


Figure 1167 Register QSPIn_OUT overview

- For each QSPI module, one separate register called QSPIn_OUT is implemented. (index n is the QSPI module number)
- The 5-bit-field SEL selects one out of 32 eGTM output lines as QSPIn_TRIG trigger output signal

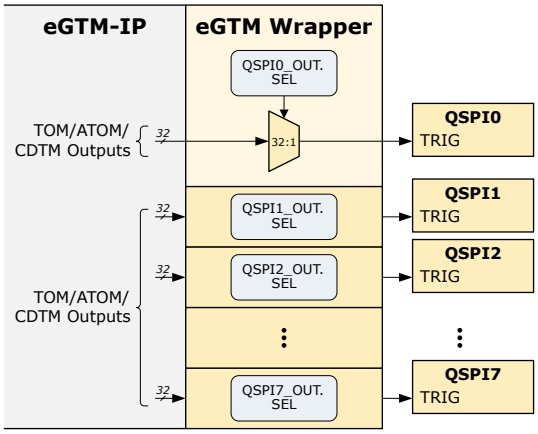


Figure 1168 QSPIn_OUT register overview

eGTM to QSPIn output selection register. n<8 , if interface exists. (n=0-7)

The bit field in this register defines which TOM/ATOM channel output is used as QSPIn trigger.

QSPIn_OUT (n=0-7)

Offset address: 061580_H+n*4

eGTM to QSPIn output selection register. n<8 , if interface exists.

Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0											SEL				
r											rw				

Field	Bits	Type	Description
SEL	4:0	rw	**eGTM to QSPIn trigger output selection**
0	31:5	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

Ethernet to eGTM connection

Signals from the Ethernet module are connected to the TIM inputs of the eGTM for measuring purposes.

GETH PPSn interrupts

The following only exists, if a Gigabit Ethernet module is present.

- The GETH PPSn interrupts (signals GETHPPSn) are connected to the TIM modules of cluster 0 and 1.
  - For GETHPPSn, n ranges from 0 to 3
- The position on the TIM input multiplexers is line 15

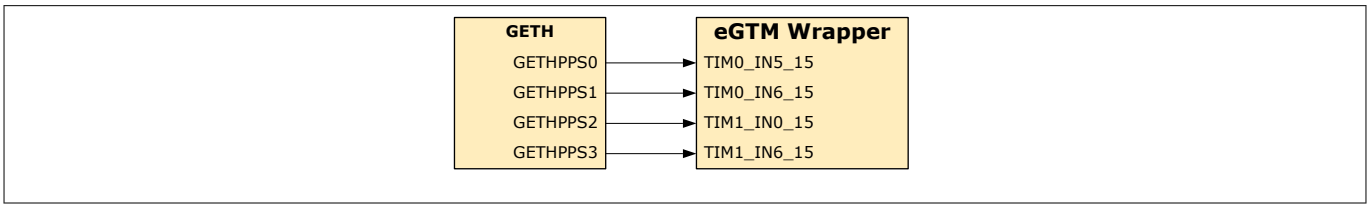


Figure 1169 Gigabit Ethernet to eGTM connections

LETH PPSn interrupts

- The LETH PPSn interrupts (signals LETHPPSn) are connected to the TIM modules of cluster 1 and 2.
  - For LETHPPSn, n is from 0 to 3
- The position on the TIM input multiplexers is line 13

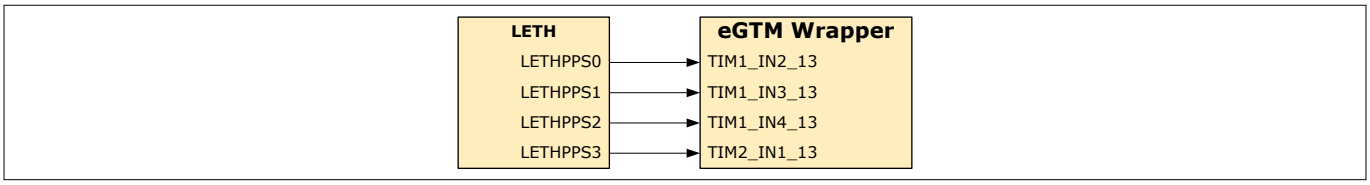


Figure 1170 Lite Ethernet to eGTM connections

FlexRay to eGTM connection

The macrotick signal MT0, is connected from the FlexRay modules 0 and 1 to the TIM inputs of the eGTM. These signals allows to measure the master clock.

- TIM cluster 0 and 2, are used for the MT0 input

- TIM cluster 1, are used for the MT1 input
- The position on the TIM input multiplexers is line 15

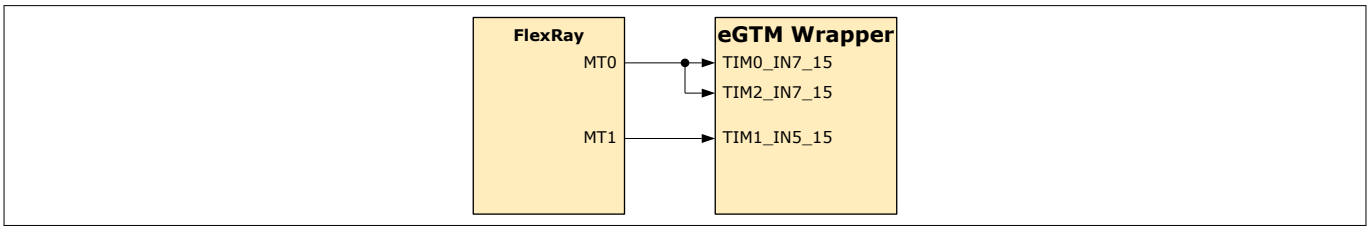


Figure 1171 FlexRay to eGTM connection

eGTM to MSC connection

The eGTM to MSC connections scales with eGTM output resources as well as MSC modules. Lower eGTM cluster numbers relate to lower MSC module numbers.

To get a first impression on the MSC connections, an overview is provided:

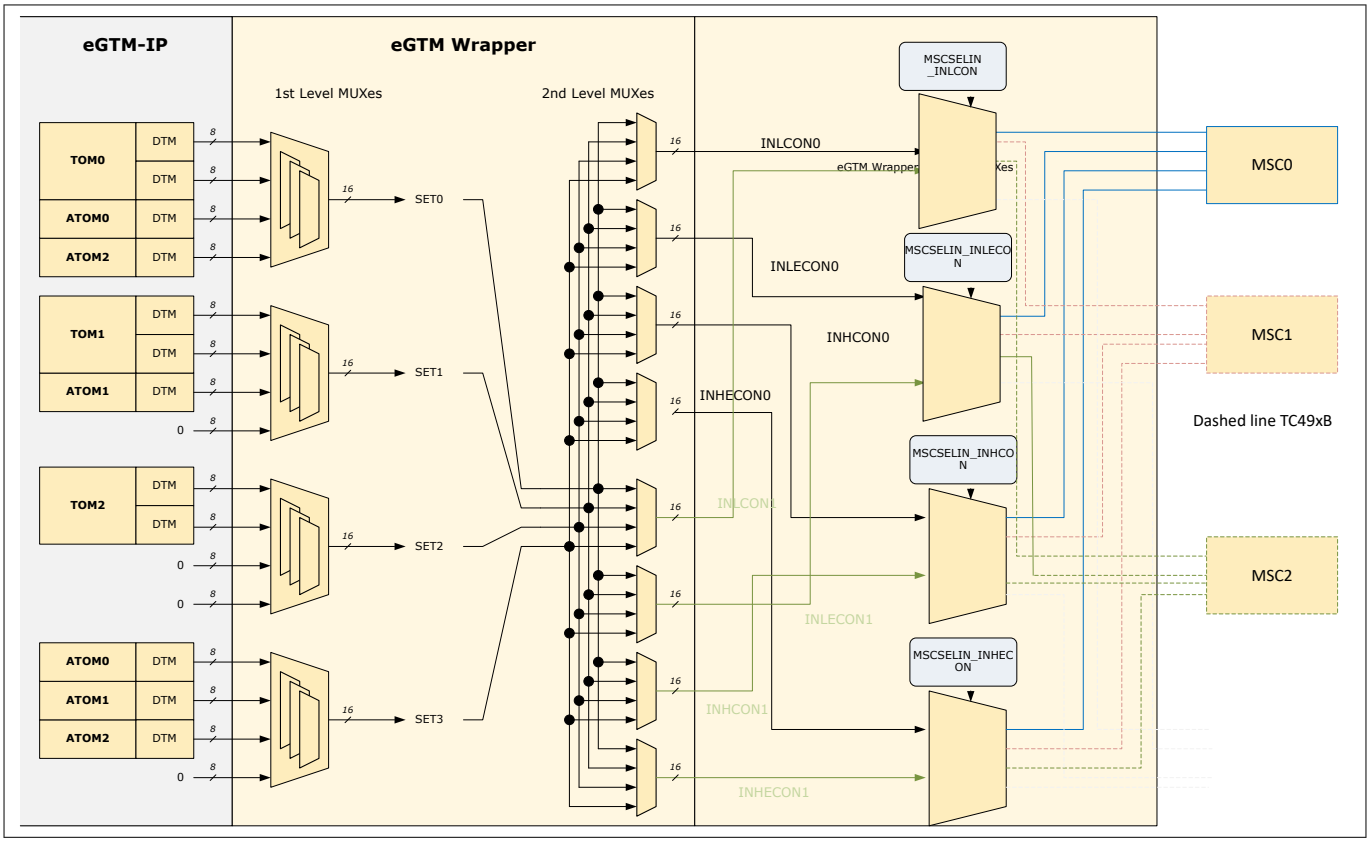


Figure 1172 MSC overview

The control registers are listed below.

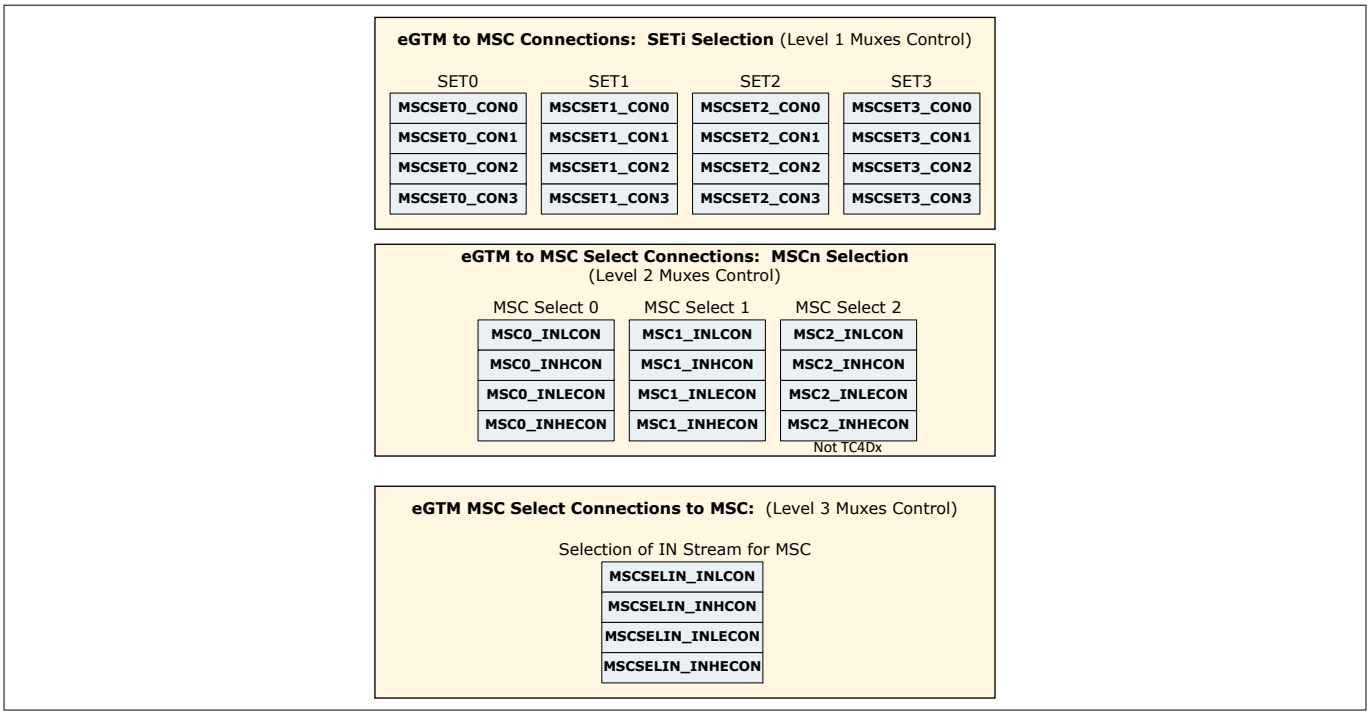


Figure 1173 Registers for MSC control

To allow a flexible distribution of the eGTM output signals to the MSC ALTIN(L/LE/H/HE)CON inputs, a three-stage approach is implemented.

First-level multiplexers

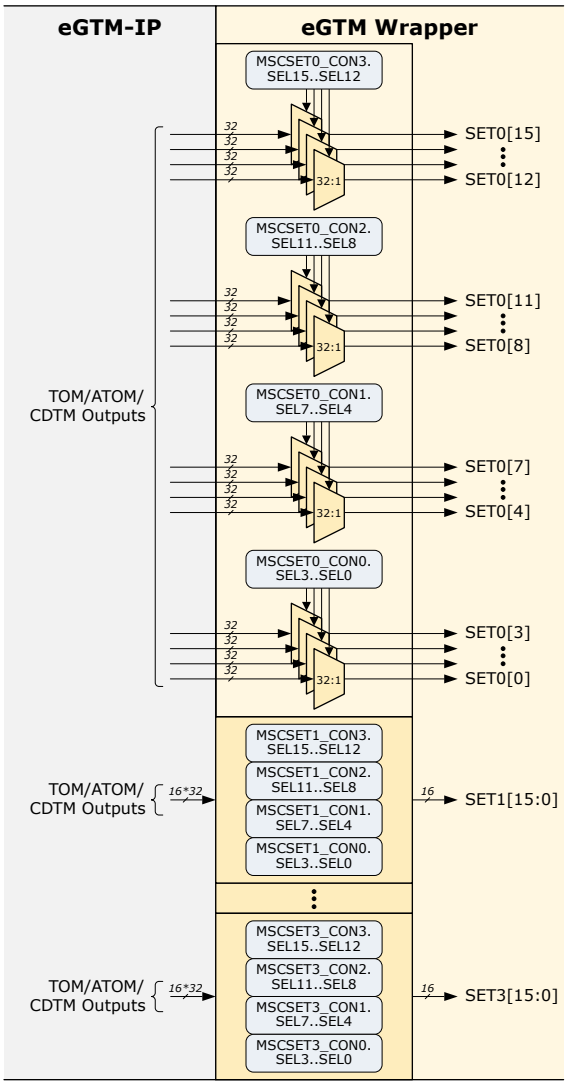


Figure 1174 First level multiplexer overview - control of the sets

The details of the first-level multiplexers for the selection of the individual lines of a SETn is shown in the figure above (eGTM configuration from TOMs/ATOMs to SETs).

- In the first stage, the first-level MUXes, several sets of 16 signals can be generated. These are called SETn
  - For eGTM, 4 sets are available in first level
- For each of the 16 lines of a SETn, one out of 32 eGTM output signals can be selected
  - Therefore, sixteen 32-to-1 multiplexers are used, each controlled by SELx
  - Each multiplexer is connected to the same 32 eGTM output signals
- Four registers, called MSCSETi_CON0 to MSCSETi_CON3, are used to hold the required number of 4 SELx bit-fields
- For each SETn, a different combination of TOM and ATOM output signals is available

Second-level multiplexers

The details of the second-level multiplexers for the selection of a SETn for an MSC alternate input bus is shown in the following figure:

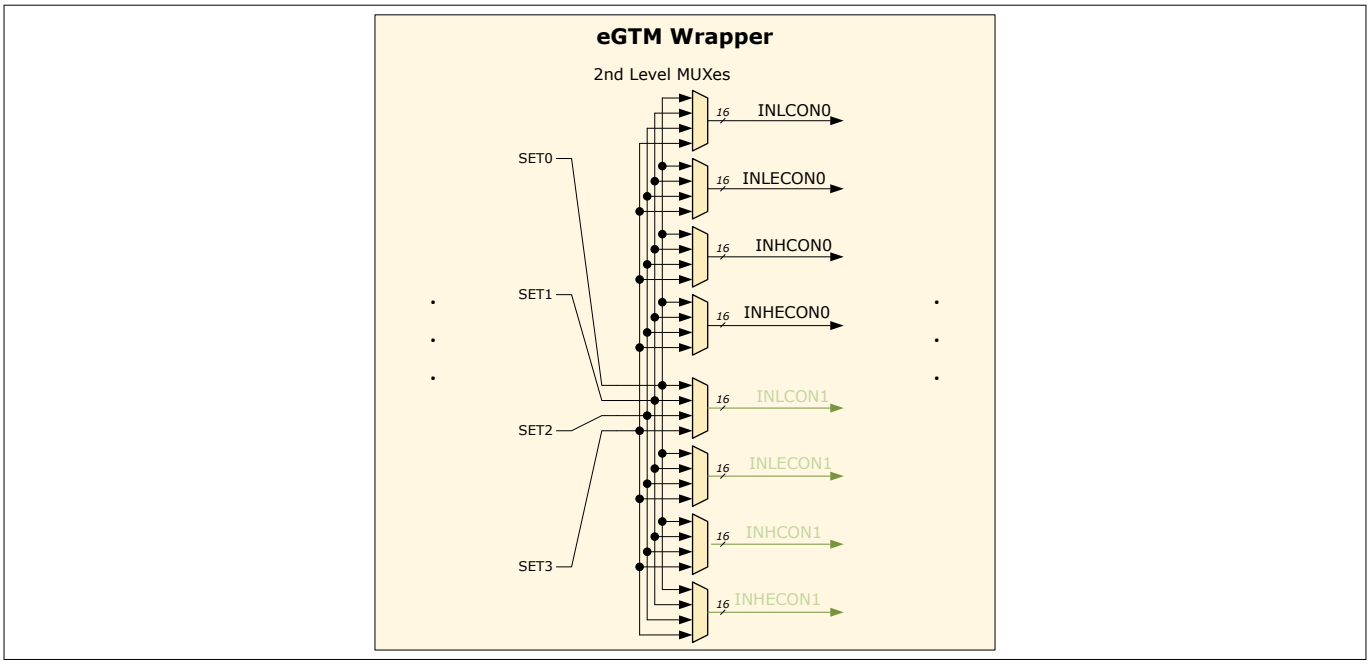


Figure 1175 Second-level multiplexer

- In the second stage, by the second-level MUXes, one of these SETn signal sets can be individually selected
- For each input bus, an associated configuration register is implemented: MSCn_INLCON, MSCn_INLECON, MSCn_INHCON, MSCn_INHECON
- Individually for each line of an input bus, the respective line out of four possible SETn signal sets can be selected. Each line has a 2-bit-field, SEL0 to SEL15, in its associated register

Additional MSC mapping

This additional multiplexer stage provides the possibility to switch a signal set to a different MSC. As this is the third stage, it is called third-level multiplexer.

The MSC has a third-level multiplexer which maps the eGTM to input bus MSCn_INLCON signal selection register (n=0-1), eGTM to input bus MSCn_INLECON signal selection register (n=0-1), eGTM to input bus MSCn_INHCON signal selection register (n=0-1) and eGTM to input bus MSCn_INHECON signal selection register (n=0-1) stream, to a Selection of INLCON stream for MSC, Selection of INLECON stream for MSC, Selection of INHCON stream for MSC and Selection of INHECON stream for MSC which is including the configuration on mapping a specific stream with an MSC. This mapping enables the possibility of having a flexible approach of mapping GTM resources to a corresponding MSC.

The mapping is shown in the figure below.

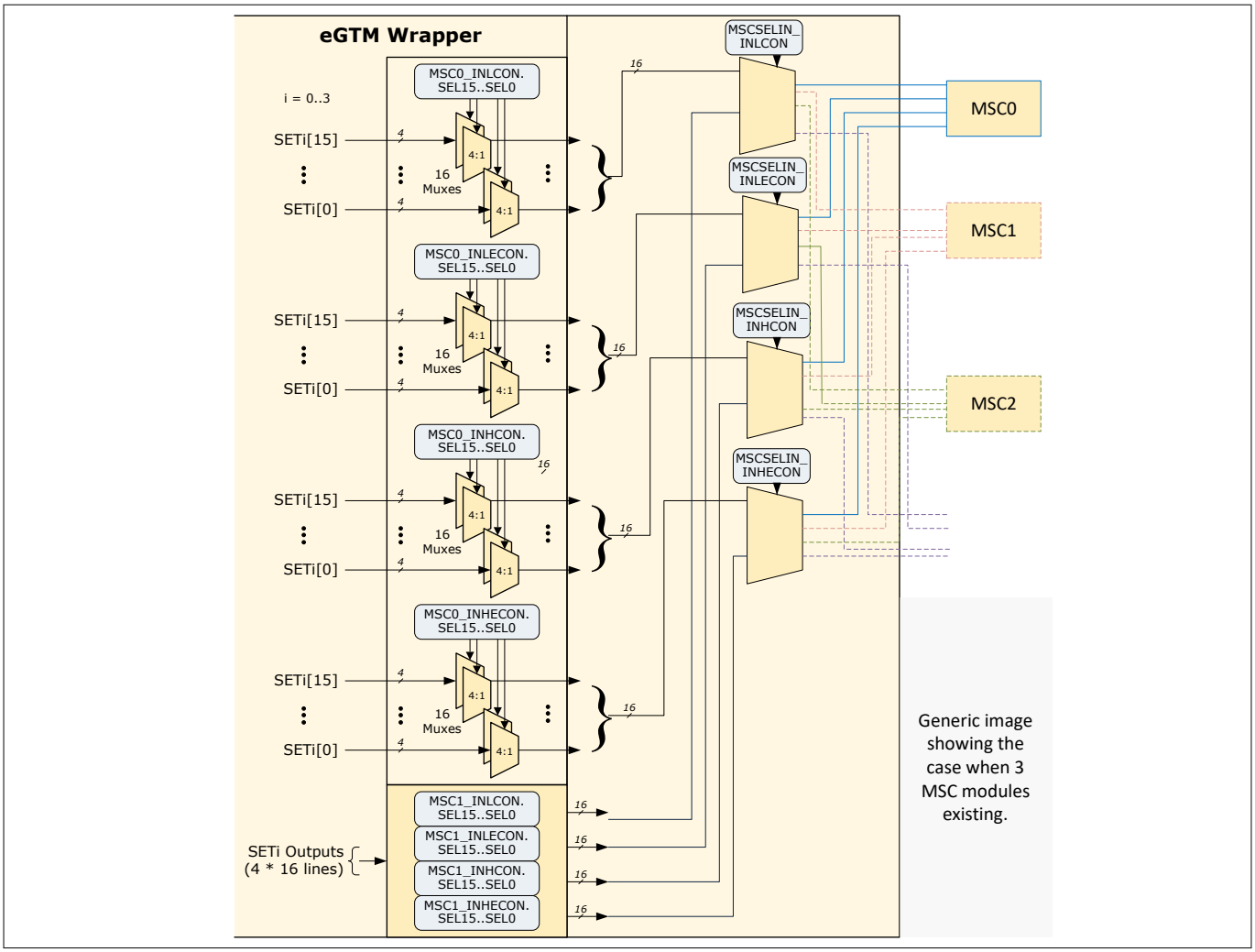


Figure 1176 MSC mapping

As shown in the figure, the registers control multiplexers mapping the signals from GTM to a specific MSC.

eGTM to MSC SETi output selection register j (j=0;i=0-3)

The bit fields in this register define which TOM/ATOM channel output is used as SETi signal x.

MSCSETi_CONj (i=0-3;j=0) Offset address: 061200_H+i*10_H+j*4

eGTM to MSC SETi output selection register j Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0			SEL3					0			SEL2				
r			rw					r			rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			SEL1					0			SEL0				
r			rw					r			rw				

Field	Bits	Type	Description
SELx (x=0-3)	8*x+4:8*x	rw	**SETi.x signal output selection**

(table continues...)

(continued)

Field	Bits	Type	Description
0	31:29, 23:21, 15:13, 7:5	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

eGTM to MSC SETi output selection register j (j=1;i=0-3)

For a description of this register, see eGTM to MSC SETi output selection register j (j=0;i=0-3).

MSCSETi_CONj (i=0-3;j=1) Offset address: 061200_H+i*10_H+j*4

eGTM to MSC SETi output selection register j Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0			SEL7					0			SEL6				
r			rw					r			rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			SEL5					0			SEL4				
r			rw					r			rw				

Field	Bits	Type	Description
SELx (x=4-7)	8*x-28:8*x-32	rw	**SETi.x signal output selection**
0	31:29, 23:21, 15:13, 7:5	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

eGTM to MSC SETi output selection register j (j=2;i=0-3)

For a description of this register, see eGTM to MSC SETi output selection register j (j=0;i=0-3).

MSCSETi_CONj (i=0-3;j=2) Offset address: 061200_H+i*10_H+j*4

eGTM to MSC SETi output selection register j Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0			SEL11					0			SEL10				
r			rw					r			rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			SEL9					0			SEL8				
r			rw					r			rw				

Field	Bits	Type	Description
SELx (x=8-11)	8*x-60:8*x-64	rw	**SETi.x signal output selection**
0	31:29, 23:21, 15:13, 7:5	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

eGTM to MSC SETi output selection register j (j=3;i=0-3)

For a description of this register, see eGTM to MSC SETi output selection register j (j=0;i=0-3).

MSCSETi_CONj (i=0-3;j=3) Offset address: 061200_H+i*10_H+j*4

eGTM to MSC SETi output selection register j Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0			SEL15					0			SEL14				
r			rw					r			rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			SEL13					0			SEL12				
r			rw					r			rw				

Field	Bits	Type	Description
SELx (x=12-15)	8*x-92:8*x-96	rw	**SETi.x signal output selection**
0	31:29, 23:21, 15:13, 7:5	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

eGTM to input bus MSCn_INLCON signal selection register (n=0-1)

The bit fields in this register define which SETi.x signal is selected for the MSCn_INLCON.x bus signal.

MSCn_INLCON (n=0-1) Offset address: 061290_H+n*10_H

eGTM to input bus MSCn_INLCON signal selection register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SEL15		SEL14		SEL13		SEL12		SEL11		SEL10		SEL9		SEL8	
rw		rw		rw		rw		rw		rw		rw		rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SEL7		SEL6		SEL5		SEL4		SEL3		SEL2		SEL1		SEL0	
rw		rw		rw		rw		rw		rw		rw		rw	

Field	Bits	Type	Description
SELx (x=0-15)	2*x+1:2*x	rw	**nth bus line INLCONx SETi signal selection**

eGTM to input bus MSCn_INLECON signal selection register (n=0-1)

The bit fields in this register define which SETi.x signals are selected for the MSCn_INLECON.x bus signal.

MSCn_INLECON (n=0-1) Offset address: 061298_H+n*10_H

eGTM to input bus MSCn_INLECON signal selection register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SEL15		SEL14		SEL13		SEL12		SEL11		SEL10		SEL9		SEL8	
rw		rw		rw		rw		rw		rw		rw		rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SEL7		SEL6		SEL5		SEL4		SEL3		SEL2		SEL1		SEL0	
rw		rw		rw		rw		rw		rw		rw		rw	

Field	Bits	Type	Description
SELx (x=0-15)	2*x+1:2*x	rw	**nth bus line INLECON.x SETi signal selection**

eGTM to input bus MSCn_INHCON signal selection register (n=0-1)

The bit fields in this register define which SETi.x signal is selected for the MSCn_INHCON.x bus signal.

MSCn_INHCON (n=0-1)

Offset address: 061294_H+n*10_H

eGTM to input bus MSCn_INHCON signal selection register

Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SEL15		SEL14		SEL13		SEL12		SEL11		SEL10		SEL9		SEL8	
rw		rw		rw		rw		rw		rw		rw		rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SEL7		SEL6		SEL5		SEL4		SEL3		SEL2		SEL1		SEL0	
rw		rw		rw		rw		rw		rw		rw		rw	

Field	Bits	Type	Description
SELx (x=0-15)	2*x+1:2*x	rw	**nth bus line INHCON.x SETi signal selection**

eGTM to input bus MSCn_INHECON signal selection register (n=0-1)

The bit fields in this register define which SETi.x signal is selected for the MSCn_INHECON.x bus signal.

MSCn_INHECON (n=0-1)

Offset address: $06129C_H+n*10_H$

eGTM to input bus MSCn_INHECON signal selection register

Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SEL15		SEL14		SEL13		SEL12		SEL11		SEL10		SEL9		SEL8	
rw		rw		rw		rw		rw		rw		rw		rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SEL7		SEL6		SEL5		SEL4		SEL3		SEL2		SEL1		SEL0	
rw		rw		rw		rw		rw		rw		rw		rw	

Field	Bits	Type	Description
SELx (x=0-15)	2*x+1:2*x	rw	**nth bus line INHECON.x SETi signal selection**

Selection of INLCON stream for MSC

MSCSELIN_INLCON Offset address: $0612D0_H$

Selection of INLCON stream for MSC Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								0					MMSC2		
r								r					rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0					MMSC1			0					MMSC0		
r					rw			r					rw		

Field	Bits	Type	Description
MMSCk (k=0-2)	8*k+2:8*k	rw	**Select the INLCON source of the existing MSCk** Select the INLCON source of the existing MSCk, with k being the number of available MSC modules minus one of the device. 000_B Map INLCONk to MSCk 100_B Map INLCON0 to MSCk 101_B Map INLCON1 to MSCk 110_B Map INLCON2 to MSCk **others,** Reserved, must not be used.
0	31:24, 23:19, 15:11, 7:3	r	**Reserved, must not be used.**

Selection of INLECON stream for MSC

MSCSELIN_INLECON Offset address: $0612D8_H$

Selection of INLECON stream for MSC Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								0					MMSC2		
r								r					rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0					MMSC1			0					MMSC0		
r					rw			r					rw		

Field	Bits	Type	Description
MMSCk (k=0-2)	8*k+2:8*k	rw	**Select the INLECON source of the existing MSCk** Select the INLECON source of the existing MSCk, with k being the number of available MSC modules minus one of the device. 000_B Map INLECONk to MSCk 100_B Map INLECON0 to MSCk 101_B Map INLECON1 to MSCk 110_B Map INLECON2 to MSCk **others,** Reserved, must not be used.
0	31:24, 23:19, 15:11, 7:3	r	**Reserved, must not be used.**

Selection of INHCON stream for MSC

MSCSELIN_INHCON Offset address: $0612D4_H$

Selection of INHCON stream for MSC Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								0					MMSC2		
r								r					rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0					MMSC1			0					MMSC0		
r					rw			r					rw		

Field	Bits	Type	Description
MMSCk (k=0-2)	8*k+2:8*k	rw	**Select the INHCON source of the existing MSCk** Select the INHCON source of the existing MSCk, with k being the number of available MSC modules minus one of the device. 000_B Map INHCONk to MSCk 100_B Map INHCON0 to MSCk 101_B Map INHCON1 to MSCk 110_B Map INHCON2 to MSCk **others,** Reserved, must not be used.
0	31:24, 23:19, 15:11, 7:3	r	**Reserved, must not be used.**

Selection of INHECON stream for MSC

MSCSELIN_INHECON Offset address: $0612DC_H$

Selection of INHECON stream for MSC Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								0					MMSC2		
r								r					rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0					MMSC1			0					MMSC0		
r					rw			r					rw		

Field	Bits	Type	Description
MMSCk (k=0-2)	8*k+2:8*k	rw	**Select the INHECON source of the existing MSCk** Select the INHECON source of the existing MSCk, with k being the number of available MSC modules minus one of the device. 000_B Map INHECONk to MSCk 100_B Map INHECON0 to MSCk 101_B Map INHECON1 to MSCk 110_B Map INHECON2 to MSCk **others,** Reserved, must not be used.
0	31:24, 23:19, 15:11, 7:3	r	**Reserved, must not be used.**

Sensor interface connections

The following sub-chapter shows the connection from the eGTM to PSI5/PSI5S sensors and SENT SPC interface.

eGTM to PSI5 connection

The eGTM to PSI5 connection enables the application to activate trigger signals to a PSI5 device (Manchester Decoded Interface).

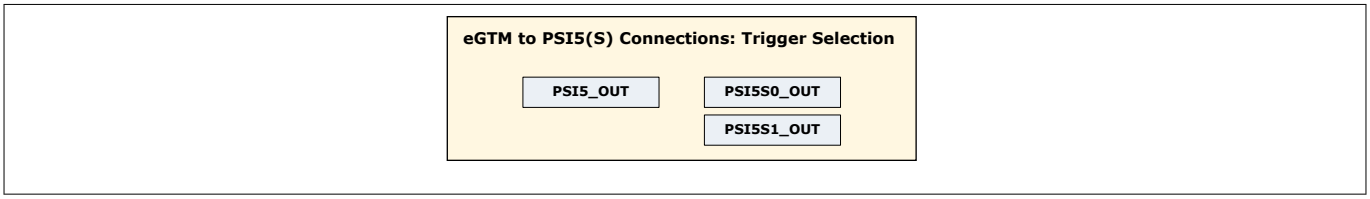


Figure 1177 PSI5 register overview

- One register called PSI5_OUT is implemented, containing 6 bit-fields
- SEL0 through to SEL5 select 1 out of 16 eGTM output lines as PSI5_TRIG0 to PSI5_TRIG5 trigger output signals
- For each trigger signal, the same set of selectable eGTM output signals are available.

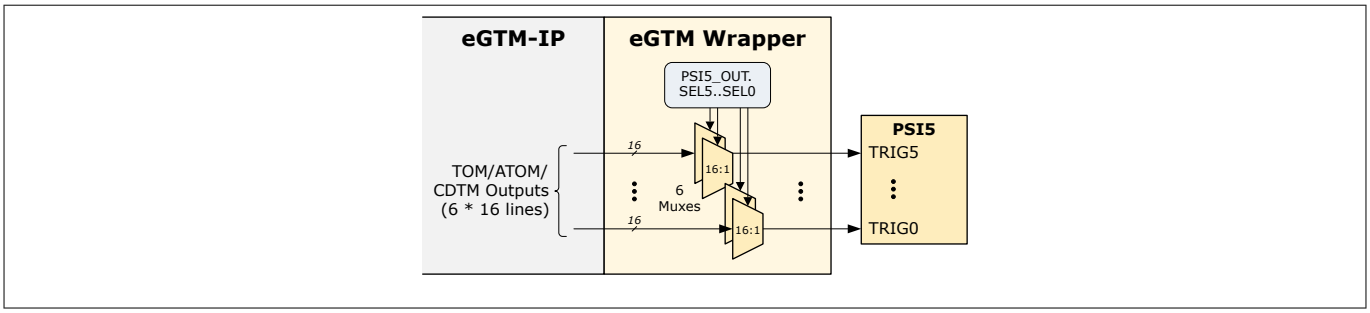


Figure 1178 PSI5_OUT details

eGTM to PSI5 output trigger selection register

The bit fields in this register define which TOM/ATOM channel output is used as PSI5 trigger x.

PSI5_OUT Offset address: 061400_H

eGTM to PSI5 output trigger selection register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								SEL5				SEL4			
r								rw				rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SEL3				SEL2				SEL1				SEL0			
rw				rw				rw				rw			

Field	Bits	Type	Description
SELx (x=0-5)	4*x+3:4*x	rw	**eGTM to PSI5 trigger x output selection**
0	31:24	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

eGTM to PSI5-S connection

The eGTM to PSI5-S connection enables the application to activate trigger signals to a PSI5-S device, which is a UART type Interface.

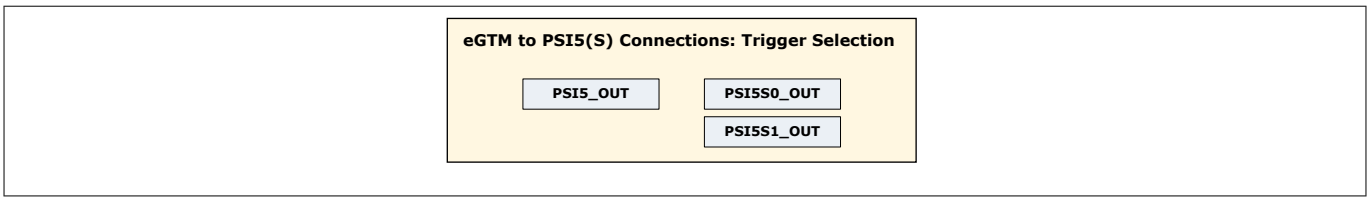


Figure 1179 PSI5S register overview

- For each PSI5_S instance, one register, called PSI5Sn_OUT, is implemented, containing 8 bit-fields
  - SEL0 to SEL7 select 1 out of 16 eGTM output lines as PSI5Sn_TRIG0 to PSI5Sn_TRIG7 trigger output signals
  - For each trigger signal, an individual set of selectable eGTM output signals is available

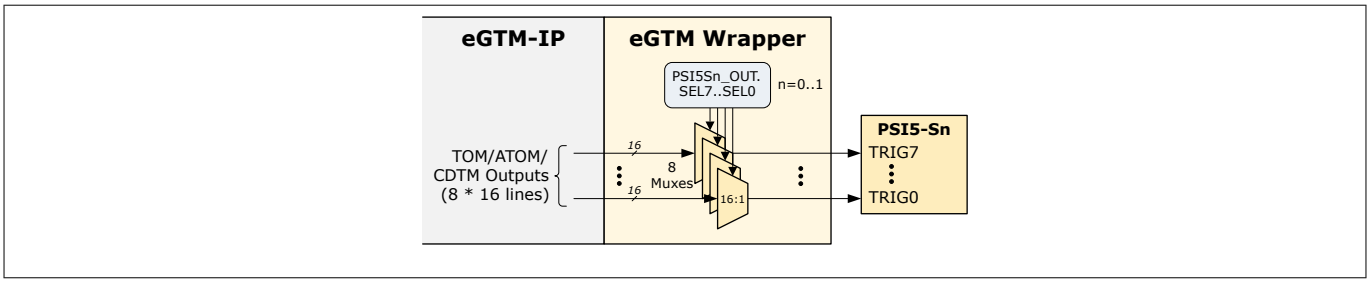


Figure 1180 PSI5Sn_OUT register details

eGTM to PSI5-Sn output trigger selection register (n=0-1)

The bit fields in this register define which TOM/ATOM channel output is used as PSI5-Sn trigger x.

PSI5Sn_OUT (n=0-1) Offset address: 061410_H+n*4

eGTM to PSI5-Sn output trigger selection register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SEL7				SEL6				SEL5				SEL4			
rw				rw				rw				rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SEL3				SEL2				SEL1				SEL0			
rw				rw				rw				rw			

Field	Bits	Type	Description
SELx (x=0-7)	4*x+3:4*x	rw	**eGTM to PSI5-Sn trigger x output selection**

eGTM to SENT connection

The connection from eGTM to SENT is for SPC mode, selecting signals to the SPC controlled SENT sensor.

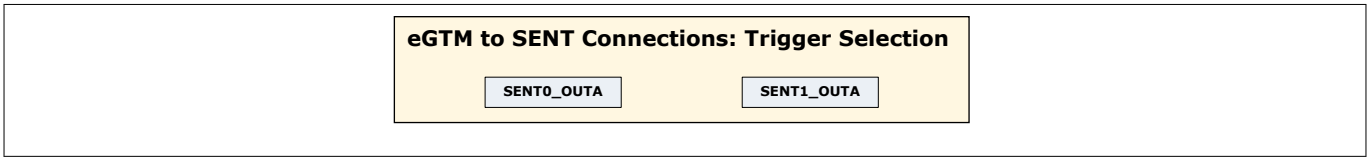


Figure 1181 SENT register overview

- 15 trigger signals can be generated for the SENT module
  - Signals SENTn_TRIG[2:0] are from the eGTM
  - Signals SENTn_TRIG[14:3] are from the classic GTM (if available)
- Select the trigger signals by: SENTn_OUTA
  - Each register holds a number of 5-bit-fields to select one out of up to 24 eGTM output signals for the respective trigger signal. Each bit-field is assigned to one eGTM cluster
- The upper 7 lines of the 32:1 multiplexers are reserved

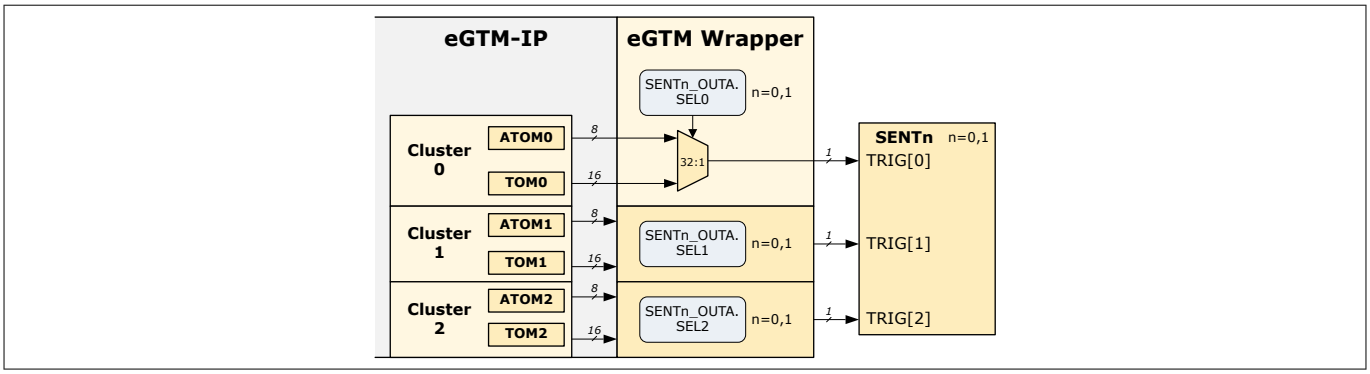


Figure 1182 SENTn_TRIG register details

eGTM to SENTn trigger x output selection register A (n=0-1)

The bit fields in this register define which TOM/ATOM channel output is used as SENTn trigger x.

SENTn_OUTA (n=0-1) Offset address: 061680_H+n*8

eGTM to SENTn trigger x output selection register A Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	SEL2					SEL1					SEL0				
r	rw					rw					rw				

Field	Bits	Type	Description
SELx (x=0-2)	5*x+4:5*x	rw	**eGTM to SENTn trigger x output selection**
0	31:15	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.3.2.7 eGTM memories address map

The eGTM has no memory based GTM modules.

44.3.2.8 eGTM OCDS interface

The eGTM OCDS module supports software development and debugging with the following features:

- Read and write access to all eGTM registers and memories while running or suspended
- Selected eGTM signals can be routed to device pins
- eGTM can be suspended by OCDS. This is controlled through the OCS register

44.3.2.8.1 eGTM suspend

If eGTM is suspended, the (e)GTM_HALT_REQ of the GTM IP is used. This leads to the fact, that in case of a soft-suspend, all outputs remain on their current state. Suspend is globally controlled in OTGS and eGTM locally with the OCS register.

This sequence can be executed by the tool transparent to the user.

***Note**: For electrical drive applications, it is not recommended to use any kind of suspend. Outputs, which remain on the very same level can lead to physical damage of switches and electrical drive. In case of usage of suspend, Infineon is not liable for any harm this might cause. In case of a need of suspend, disconnect all lines connected to outputs of the eGTM and debug the pure pins only.*

44.3.2.8.2 OCDS trigger bus (OTGB) interface

The OTGB0 and OTGB1 trigger buses are connected through the regular OTGB infrastructure to device pins and to MCDS.

OTGB0 and OTGB1 are 16-bit trigger buses.

Further information about OTGB0 and OTGB1 is available in the OTGM (OCDS Trigger MUX) section of the OCDS chapter.

eGTM trigger and trace features

- IO signals
  - TIM input signals (after filter)
  - TOM output signals
  - ATOM output signals
  - SPE NIPD and DIR signals
  - Groups of eight signals, such as all inputs of a specific TIM for example
  - Two or four arbitrary groups using OTGB0 or OTGB1
  - Signals can be used as triggers or processed with OTGS (OCDS trigger switch)
  - Signals can be routed to TGO device pins through the OTGS

The eGTM module has 16-bit and 32-bit Trigger Sets, which are selected with the OTSS register.

The OTGB0 and OTGB1 signals relate to the eGTM internal clock, which can be slower or equal to the OTGB clock.

***Note**: The trigger sets mentioned above only exist if the corresponding resources exist on the device.*

44.3.2.8.3 IO and other signal trigger sets

The IO trigger sets consist of the most important TIM, TOM, and ATOM signals in groups of 8. There are two groups of SPE signals and one group for miscellaneous signals.

The multiplexer allows for mapping arbitrary and different signal groups to the high and the low byte of a 16-bit trigger set. It is also possible to use one or two 16-bit trigger sets with this flexibility. This is controlled with the OTSC0 register.

For normal clock configurations, this difference will not be more than a factor of two, which means a maximum delay of one eGTM clock.

Table 2299 eGTM trigger sets

Trigger set
TS16_IOS Trigger Set IO and Other Signals

The following trigger set mapping options table shows all trigger set mapping options.

Table 2300 Trigger set mapping options

OTGB0	OTGB1
TS16_IOS	TS16_IOS

The OTGB0 and OTGB1 signals relate to the eGTM internal clock, which can be slower or equal to the OTGB clock.

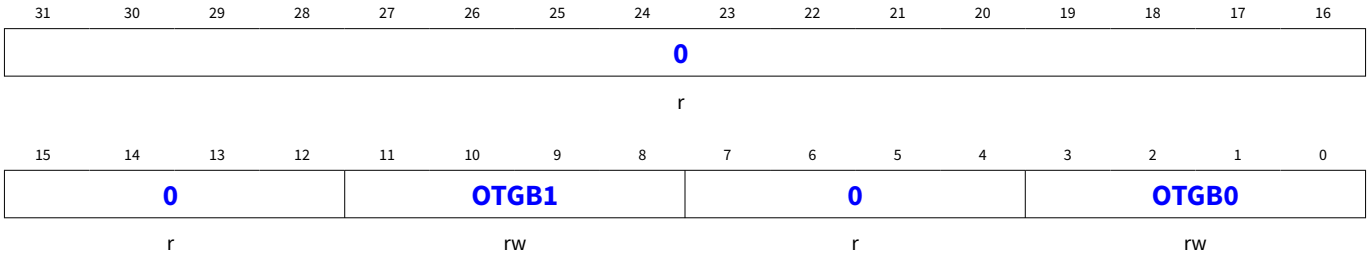
Table 2301 TS16_IOS trigger set IO and other signals

Bits	Name	Description	
[7:0]	SG0	A group of 8 signals from a selected TIM, TOM, ATOM or SPE.	
		TIM	All 8 input signals after filter F_OUT.
		TOML	Lower 8 output signals (channel 0-7) of TOM.
		TOMH	Higher 8 output signals (channel 8-15) of TOM.
		ATOM	All 8 output signals of ATOM channels.
		SPE	For OTSC0.BxxMI Module Instance 0: Bits 0,1: SPE0_NIPD, SPE_DIR0 Bits 2,3: SPE1_NIPD, SPE_DIR1 Bits 4,5: SPE2_NIPD, SPE_DIR2 others, reserved (value is 0)
		MISC	Bit 0: reserved Bit 1: reserved Bit 4: TBU0 trigger (OTBU0T) Bit 5: TBU1 trigger (OTBU1T) Bit 6: TBU2 trigger (OTBU2T) Bit 7: reserved others, reserved
[15:8]	SG1		Independent selection with same options as for bits [7:0]

44.3.2.8.4 OCDS trigger set select register

OCDS_OTSS Offset address: 060510_H

OCDS trigger set select register Debug Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
OTGB0	3:0	rw	**Trigger set for OTGB0** 0_H No trigger set selected 1_H Trigger set TS16_IOS in table "TS16_IOS trigger set IO and other signals" **others,** Reserved, must not be used.
OTGB1	11:8	rw	**Trigger set for OTGB1** 0_H No trigger set selected 1_H Trigger set TS16_IOS in table "TS16_IOS trigger set IO and other signals" **others,** Reserved, must not be used.
0	7:4, 15:12, 31:16	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.3.2.9 eGTM application constraints and limitations

If setting up an application on eGTM, the following constraints have to be taken into account. Otherwise, the eGTM may not be able to fulfill the specified behavior.

For devices being capable of frequencies above 400 MHz, setting CCM[i] configuration register (i=0,2).CLS_CLK_DIV==01_B is not allowed for frequencies above 250 MHz, as the device might be malfunctioning.

44.3.2.10 General remarks

The eGTM kernel is a pure 32-bit module, therefore accessing the eGTM, shall be not byte-wise or half word. This shall not be done and might have strange effects on the module like overwriting registers with scrapped information. For TriCore™ architecture conform accesses, no error will be shown to the outside.

The eGTM shall be used with Pulse Notify as Interrupt Mode, as otherwise the correct functionality of eGTM interrupts and TriCore™ cannot be guaranteed.

44.4 High Resolution Pulse Width Modulation (HRPWM)

This device does not provide HRPWM. The function and the registers are not implemented in this device.

44.5 Registers

44.5.1 Register overview - access mode glossary

Table 2302 Register overview - access mode glossary

Keyword	Description
E	Access protection using PROT register CTRL_PROTE.
SE	Access protection using PROT register PROTSE.
APU-P	Protection group consisting of registers CTRL_ACCEN_WRA, CTRL_ACCEN_WRB, CTRL_ACCEN_RDA, CTRL_ACCEN_RDB, CTRL_ACCEN_VM, CTRL_ACCEN_PRS.
P	Access protection using *APU-P* registers.
APU-PCLi (i=0-2)	Protection group consisting of registers CLi_ACCEN_WRA, CLi_ACCEN_WRB, CLi_ACCEN_RDA, CLi_ACCEN_RDB, CLi_ACCEN_VM, CLi_ACCEN_PRS.
PCLi	Access protection using *APU-PCLi* registers.
APU-PGTM	Protection group consisting of registers CTRLGTM_ACCEN_WRA, CTRLGTM_ACCEN_WRB, CTRLGTM_ACCEN_RDA, CTRLGTM_ACCEN_RDB, CTRLGTM_ACCEN_VM, CTRLGTM_ACCEN_PRS.
PGTM	Access protection using *APU-PGTM* registers.
APU-PSAFE	Protection group consisting of registers SAFEGTM_ACCEN_WRA, SAFEGTM_ACCEN_WRB, SAFEGTM_ACCEN_RDA, SAFEGTM_ACCEN_RDB, SAFEGTM_ACCEN_VM, SAFEGTM_ACCEN_PRS.
PSAFE	Access protection using *APU-PSAFE* registers.
APU-PWRAP	Protection group consisting of registers WRAP_ACCEN_WRA, WRAP_ACCEN_WRB, WRAP_ACCEN_RDA, WRAP_ACCEN_RDB, WRAP_ACCEN_VM, WRAP_ACCEN_PRS.
PWRAP	Access protection using *APU-PWRAP* registers.
32	Access only when using 32-bit width.
BE	Always returns a Bus Error.
U	No access restrictions.
PROT	Access restrictions as defined in the PROT register access rules.
nBE	Indicates that no Bus Error is generated when accessing this address range, even though it is either an access to an undefined address or the access does not follow the given rules.

44.5.2 Register overview - EGTM domain CPU (ascending offset address)

Table 2303 Register overview - EGTM domain CPU (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CLSi_ARCH_REV	eGTM version control register	000000_H +i*20000 $_H$	PGTM, 32	BE	Kernel Reset	7635

(table continues...)

Table 2303 (continued) Register overview - EGTM domain CPU (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CLSi_ARCH_RST	eGTM global reset register	000004_H +i*20000 H	PGTM, 32	PGTM, 32	Kernel Reset	7635
CLSi_ARCH_CTRL	eGTM global control register	000008_H +i*20000 H	PGTM, 32	PGTM, 32	Kernel Reset	7636
CLSi_ARCH_CFG	eGTM configuration register	$00000C_H$ +i*20000 H	PGTM, 32	PGTM, 32	Kernel Reset	7637
CLSi_ARCH_AEI_ADDR_XPT	eGTM AEI timeout exception address register	000010_H +i*20000 H	PGTM, 32	BE	See 7638	7638
CLSi_ARCH_AEI_STA_XPT	eGTM AEI non-zero status register	000014_H +i*20000 H	PGTM, 32	BE	See 7639	7639
CLSi_ARCH_IRQ_NOTIFY	eGTM Interrupt notification register	000018_H +i*20000 H	PGTM, 32	PGTM, 32	Kernel Reset	7640
CLSi_ARCH_IRQ_EN	eGTM interrupt enable register	$00001C_H$ +i*20000 H	PGTM, 32	PGTM, 32	Kernel Reset	7643
CLSi_ARCH_EIRQ_EN	eGTM error interrupt enable register	000020_H +i*20000 H	PGTM, 32	PGTM, 32	Kernel Reset	7644
CLSi_ARCH_IRQ_FORCINT	eGTM Software interrupt generation register	000024_H +i*20000 H	PGTM, 32	PGTM, 32	Kernel Reset	7645
CLSi_ARCH_IRQ_MODE	eGTM toplevel interrupts mode selection	000028_H +i*20000 H	PGTM, 32	PGTM, 32	Kernel Reset	7647
CLSi_ARCH_CLK_CFG	eGTM Cluster Clock Configuration	$00002C_H$ +i*20000 H	PGTM, 32	PGTM, 32	Kernel Reset	7648
CLSi_AEI_BRIDGE_MODE	eGTM AEI bridge mode register	000040_H +i*20000 H	PGTM, 32	PGTM, 32	See 7649	7649
CLSi_AEI_BRIDGE_PTR1	eGTM AEI bridge pointer 1 register	000044_H +i*20000 H	PGTM, 32	BE	See 7652	7652
CLSi_AEI_BRIDGE_PTR2	eGTM AEI bridge pointer 2 register	000048_H +i*20000 H	PGTM, 32	BE	See 7653	7653
CLSi_CMU_CLK_EN	CMU clock enable	000080_H +i*20000 H	PGTM, 32	PGTM, 32	Kernel Reset	7653

(table continues...)

Table 2303 (continued) Register overview - EGTM domain CPU (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CLSi_CMU_GCLK_NUM	The numerator for CMU global clock resolution generator	000084_H +i*20000 $_H$	PGTM, 32	PGTM, 32	Kernel Reset	7654
CLSi_CMU_GCLK_DEN	The denominator for CMU global clock resolution generator	000088_H +i*20000 $_H$	PGTM, 32	PGTM, 32	Kernel Reset	7655
CLSi_CMU_CLKx_CTRL	CMU control for clock resolution generator [x]	$00008C_H$ +i*20000 $_H$+x*4	PGTM, 32	PGTM, 32	Kernel Reset	7656
CLSi_CMU_CLK_6_CTRL	CMU control for clock resolution generator 6	$0000A4_H$ +i*20000 $_H$	PGTM, 32	PGTM, 32	Kernel Reset	7657
CLSi_CMU_CLK_7_CTRL	CMU control for clock resolution generator 7	$0000A8_H$ +i*20000 $_H$	PGTM, 32	PGTM, 32	Kernel Reset	7657
CLSi_CMU_ECLKz_NUM	The numerator for the external clock resolution generator [z]	$0000AC_H$ +i*20000 $_H$+z*8	PGTM, 32	PGTM, 32	Kernel Reset	7658
CLSi_CMU_ECLKz_DEN	The denominator for the external clock resolution generator [z]	$0000B0_H$ +i*20000 $_H$+z*8	PGTM, 32	PGTM, 32	Kernel Reset	7659
CLSi_CMU_FXCLK_CTRL	CMU control for selection of FCR subblock input	$0000C4_H$ +i*20000 $_H$	PGTM, 32	PGTM, 32	Kernel Reset	7660
CLSi_CMU_CLK_CTRL	CMU control for clock resolution generator	$0000CC_H$ +i*20000 $_H$	PGTM, 32	PGTM, 32	Kernel Reset	7661
CLSi_TBU_CHEN	TBU global channel enable	000100_H +i*20000 $_H$	PGTM, 32	PGTM, 32	Kernel Reset	7662
CLSi_TBU_CH0_CTRL	TBU channel 0 control	000104_H +i*20000 $_H$	PGTM, 32	PGTM, 32	Kernel Reset	7663
CLSi_TBU_CH0_BASE	TBU channel 0 base	000108_H +i*20000 $_H$	PGTM, 32	PGTM, 32	Kernel Reset	7664
CLSi_TBU_CH1_CTRL	TBU channel 1 control	$00010C_H$ +i*20000 $_H$	PGTM, 32	PGTM, 32	Kernel Reset	7664
CLSi_TBU_CHx_BASE	TBU channel [x] base	000110_H +i*20000 $_H$+(x-1)*8	PGTM, 32	PGTM, 32	Kernel Reset	7665
CLSi_TBU_CH2_CTRL	TBU channel 2 control	000114_H +i*20000 $_H$	PGTM, 32	PGTM, 32	Kernel Reset	7666

(table continues...)

Table 2303 (continued) Register overview - EGTM domain CPU (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CLSi_ICM_IRQG_0	ICM_IRQG_0	000400_H +i*20000 H	PGTM, 32	BE	Kernel Reset	7667
CLSi_ICM_IRQG_2	ICM Interrupt group register covering TIM0, TIM1, TIM2, TIM3	000408_H +i*20000 H	PGTM, 32	BE	Kernel Reset	7668
CLSi_ICM_IRQG_MEI	ICM Interrupt group register for module error interrupt information	000430_H +i*20000 H	PGTM, 32	BE	Kernel Reset	7669
CLSi_ICM_IRQG_CEI1	ICM Interrupt group register 1 for channel error interrupt information	000438_H +i*20000 H	PGTM, 32	BE	Kernel Reset	7671
CLSi_ICM_IRQG_SPE_CEI	ICM Interrupt group SPE for module Error Interrupt information	$0004B4_H$ +i*20000 H	PGTM, 32	BE	Kernel Reset	7672
CLSi_ICM_IRQG_CLSg_MEI	ICM_IRQG_CLS_[g]_MEI	000510_H +i*20000 H	PGTM, 32	BE	Kernel Reset	7672
CLSi_ICM_IRQG_SPE_CI	ICM Interrupt group SPE for module Interrupt information	000570_H +i*20000 H	PGTM, 32	BE	Kernel Reset	7674
CLSi_ICM_IRQG_ATOMg_CI	ICM_IRQG_ATOM_[g]_CI	000590_H +i*20000 H	PGTM, 32	BE	Kernel Reset	7675
CLSi_ICM_IRQG_TOMg_CI	ICM_IRQG_TOM_[g]_CI	$0005A0_H$ +i*20000 H+g*4	PGTM, 32	BE	Kernel Reset	7676
CLSi_MON_STATUS	MON status register	000680_H +i*20000 H	PSAFE, 32	PSAFE, 32	Kernel Reset	7678
CLSi_CMP_EN	CMP comparator enable register	$0006C0_H$ +i*20000 H	PSAFE, 32	PSAFE, 32	Kernel Reset	7679
CLSi_CMP_IRQ_NOTIFY	CMP event notification register	$0006C4_H$ +i*20000 H	PSAFE, 32	PSAFE, 32	Kernel Reset	7680
CLSi_CMP_IRQ_EN	CMP interrupt enable register	$0006C8_H$ +i*20000 H	PSAFE, 32	PSAFE, 32	Kernel Reset	7681
CLSi_CMP_IRQ_FORCINT	CMP interrupt force register	$0006CC_H$ +i*20000 H	PSAFE, 32	PSAFE, 32	Kernel Reset	7682
CLSi_CMP_IRQ_MODE	CMP interrupt mode configuration register	$0006D0_H$ +i*20000 H	PSAFE, 32	PSAFE, 32	Kernel Reset	7683

(table continues...)

Table 2303 (continued) Register overview - EGTM domain CPU (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CLSi_CMP_EIRQ_EN	CMP error interrupt enable register	$0006D4_H$ +i*20000_H	PSAFE, 32	PSAFE, 32	Kernel Reset	7683
CLSi_TIM_CHx_GPR0	TIM[i] channel [x] general-purpose 0 register	000800_H +i*20000_H+x*80_H	PCLi, 32	PCLi, 32	See 7684	7684
CLSi_TIM_CHx_GPR1	TIM[i] channel [x] general-purpose 1 register	000804_H +i*20000_H+x*80_H	PCLi, 32	PCLi, 32	See 7685	7685
CLSi_TIM_CHx_CNT	TIM[i] channel [x] SMU counter register	000808_H +i*20000_H+x*80_H	PCLi, 32	BE	See 7686	7686
CLSi_TIM_CHx_ECNT	TIM[i] channel [x] SMU edge counter register	$00080C_H$ +i*20000_H+x*80_H	PCLi, 32	BE	See 7687	7687
CLSi_TIM_CHx_CNTS	TIM[i] channel [x] SMU shadow counter register	000810_H +i*20000_H+x*80_H	PCLi, 32	PCLi, 32	See 7688	7688
CLSi_TIM_CHx_TDUC	TIM[i]_CH[x]_TDUC	000814_H +i*20000_H+x*80_H	PCLi, 32	PCLi, 32	See 7689	7689
CLSi_TIM_CHx_TDUV	TIM[i]_CH[x]_TDUV	000818_H +i*20000_H+x*80_H	PCLi, 32	PCLi, 32	See 7690	7690
CLSi_TIM_CHx_FLT_RE	TIM[i] channel [x] filter parameter 0 register	$00081C_H$ +i*20000_H+x*80_H	PCLi, 32	PCLi, 32	See 7691	7691
CLSi_TIM_CHx_FLT_FE	TIM[i] channel [x] filter parameter 1 register	000820_H +i*20000_H+x*80_H	PCLi, 32	PCLi, 32	See 7692	7692
CLSi_TIM_CHx_CTRL	TIM[i] channel [x] control register	000824_H +i*20000_H+x*80_H	PCLi, 32	PCLi, 32	See 7692	7692
CLSi_TIM_CHx_ECTRL	TIM[i] channel [x] extended control register	000828_H +i*20000_H+x*80_H	PCLi, 32	PCLi, 32	See 7707	7707
CLSi_TIM_CHx_IRQ_NOTIFY	TIM[i] channel [x] interrupt notification register	$00082C_H$ +i*20000_H+x*80_H	PCLi, 32	PCLi, 32	See 7717	7717
CLSi_TIM_CHx_IRQ_EN	TIM[i] channel [x] interrupt enable register	000830_H +i*20000_H+x*80_H	PCLi, 32	PCLi, 32	See 7719	7719
CLSi_TIM_CHx_IRQ_FORCINT	TIM[i] channel [x] force interrupt register	000834_H +i*20000_H+x*80_H	PCLi, 32	PCLi, 32	See 7720	7720

(table continues...)

Table 2303 (continued) Register overview - EGTM domain CPU (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CLSi_TIM_CHx_IRQ_MODE	TIM[i] channel [x] interrupt mode configuration register	000838_H +i*20000_H+x*80_H	PCLi, 32	PCLi, 32	See 7722	7722
CLSi_TIM_CHx_EIRQ_EN	TIM[i] channel [x] error interrupt enable register	$00083C_H$ +i*20000_H+x*80_H	PCLi, 32	PCLi, 32	See 7723	7723
CLSi_TIM_INP_VAL	TIM[i] input value observation register	$000C00_H$ +i*20000_H	PCLi, 32	BE	Kernel Reset	7724
CLSi_TIM_IN_SRC	TIM[i] AUX IN source selection register	$000C04_H$ +i*20000_H	PCLi, 32	PCLi, 32	Kernel Reset	7724
CLSi_TIM_RST	TIM[i] global software reset register	$000C08_H$ +i*20000_H	PCLi, 32	PCLi, 32	Kernel Reset	7726
CLSi_TOM_CHx_CTRL	TOM[i] channel [x] control register	001000_H +i*20000_H+x*40_H	PCLi, 32	PCLi, 32	See 7727	7727
CLSi_TOM_CHx_SR0	TOM[i] channel [x] CCU0 compare shadow register	001004_H +i*20000_H+x*40_H	PCLi, 32	PCLi, 32	See 7759	7759
CLSi_TOM_CHx_SR1	TOM[i] channel [x] CCU1 compare shadow register	001008_H +i*20000_H+x*40_H	PCLi, 32	PCLi, 32	See 7759	7759
CLSi_TOM_CHx_CM0	TOM[i] channel [x] CCU0 compare register	$00100C_H$ +i*20000_H+x*40_H	PCLi, 32	PCLi, 32	See 7760	7760
CLSi_TOM_CHx_CM1	TOM[i] channel [x] CCU1 compare register	001010_H +i*20000_H+x*40_H	PCLi, 32	PCLi, 32	See 7760	7760
CLSi_TOM_CHx_CN0	TOM[i] channel [x] CCU0 counter	001014_H +i*20000_H+x*40_H	PCLi, 32	PCLi, 32	See 7761	7761
CLSi_TOM_CHx_STAT	TOM[i] channel [x] status register	001018_H +i*20000_H+x*40_H	PCLi, 32	PCLi, 32	See 7762	7762
CLSi_TOM_CHx_IRQ_NOTIFY	TOM[i] channel [x] interrupt notification register	$00101C_H$ +i*20000_H+x*40_H	PCLi, 32	PCLi, 32	See 7762	7762
CLSi_TOM_CHx_IRQ_EN	TOM[i] channel [x] interrupt enable register	001020_H +i*20000_H+x*40_H	PCLi, 32	PCLi, 32	See 7764	7764
CLSi_TOM_CHx_IRQ_FORCINT	TOM[i] channel [x] force interrupt register	001024_H +i*20000_H+x*40_H	PCLi, 32	PCLi, 32	See 7765	7765

(table continues...)

Table 2303 (continued) Register overview - EGTM domain CPU (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CLSi_TOM_CHx_IRQ_MODE	TOM[i] channel [x] interrupt mode register	001028_H +i*20000_H+x*40_H	PCLi, 32	PCLi, 32	See 7766	7766
CLSi_TOM_CHx_CTRL_SR	TOM[i] channel [x] control shadow register	001030_H +i*20000_H+x*40_H	PCLi, 32	PCLi, 32	See 7767	7767
CLSi_TOM_TGCg_GLB_CTRL	TOM[i] TGC [g] global control register	001430_H +i*20000_H+g*80_H	PCLi, 32	PCLi, 32	Kernel Reset	7768
CLSi_TOM_TGCg_ACT_TB	TOM[i] TGC [g] action time base register	001434_H +i*20000_H+g*80_H	PCLi, 32	PCLi, 32	Kernel Reset	7769
CLSi_TOM_TGCg_FUPD_CTRL	TOM[i] TGC [g] force update control register	001438_H +i*20000_H+g*80_H	PCLi, 32	PCLi, 32	Kernel Reset	7770
CLSi_TOM_TGCg_INT_TRIG	TOM[i] TGC [g] internal trigger control register	$00143C_H$ +i*20000_H+g*80_H	PCLi, 32	PCLi, 32	Kernel Reset	7771
CLSi_TOM_TGCg_ENDIS_CTRL	TOM[i] TGC [g] enable/ disable control register	001470_H +i*20000_H+g*80_H	PCLi, 32	PCLi, 32	Kernel Reset	7772
CLSi_TOM_TGCg_ENDIS_STAT	TOM[i] TGC [g] enable/ disable status register	001474_H +i*20000_H+g*80_H	PCLi, 32	PCLi, 32	Kernel Reset	7773
CLSi_TOM_TGCg_OUTEN_CTRL	TOM[i] TGC [g] output enable control register	001478_H +i*20000_H+g*80_H	PCLi, 32	PCLi, 32	Kernel Reset	7774
CLSi_TOM_TGCg_OUTEN_STAT	TOM[i] TGC [g] output enable status register	$00147C_H$ +i*20000_H+g*80_H	PCLi, 32	PCLi, 32	Kernel Reset	7774
CLSi_ATOM_CHx_CTRL	ATOM[i] channel [x] control register	001804_H +i*20000_H+x*80_H	PCLi, 32	PCLi, 32	See 7775	7775
CLSi_ATOM_CHx_CTRL_SOMI	ATOM[i] channel [x] control register	001804_H +i*20000_H+x*80_H	PCLi, 32	PCLi, 32	See 7777	7777
CLSi_ATOM_CHx_CTRL_SOMC	ATOM[i] channel [x] control register	001804_H +i*20000_H+x*80_H	PCLi, 32	PCLi, 32	See 7781	7781
CLSi_ATOM_CHx_CTRL_SOMP	ATOM[i] channel [x] control register	001804_H +i*20000_H+x*80_H	PCLi, 32	PCLi, 32	See 7788	7788
CLSi_ATOM_CHx_CTRL_SOMS	ATOM[i] channel [x] control register	001804_H +i*20000_H+x*80_H	PCLi, 32	PCLi, 32	See 7797	7797

(table continues...)

Table 2303 (continued) Register overview - EGTM domain CPU (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CLSi_ATOM_CHx_CTRL_SOMB	ATOM[i] channel [x] control register	001804_H +i*20000_H+x*80_H	PCLi, 32	PCLi, 32	See 7802	7802
CLSi_ATOM_CHx_SR0	ATOM[i] channel [x] CCU0 compare shadow register	001808_H +i*20000_H+x*80_H	PCLi, 32	PCLi, 32	See 7808	7808
CLSi_ATOM_CHx_SR1	ATOM[i] channel [x] CCU1 compare shadow register	$00180C_H$ +i*20000_H+x*80_H	PCLi, 32	PCLi, 32	See 7809	7809
CLSi_ATOM_CHx_CM0	ATOM[i] channel [x] CCU0 compare register	001810_H +i*20000_H+x*80_H	PCLi, 32	PCLi, 32	See 7810	7810
CLSi_ATOM_CHx_CM1	ATOM[i] channel [x] CCU1 compare register	001814_H +i*20000_H+x*80_H	PCLi, 32	PCLi, 32	See 7810	7810
CLSi_ATOM_CHx_CN0	ATOM[i] channel [x] CCU0 counter register	001818_H +i*20000_H+x*80_H	PCLi, 32	PCLi, 32	See 7811	7811
CLSi_ATOM_CHx_STAT	ATOM[i] channel [x] status register	$00181C_H$ +i*20000_H+x*80_H	PCLi, 32	PCLi, 32	See 7811	7811
CLSi_ATOM_CHx_IRQ_NOTIFY	ATOM[i] channel [x] interrupt notification register	001820_H +i*20000_H+x*80_H	PCLi, 32	PCLi, 32	See 7815	7815
CLSi_ATOM_CHx_IRQ_EN	ATOM[i] channel [x] interrupt enable register	001824_H +i*20000_H+x*80_H	PCLi, 32	PCLi, 32	See 7816	7816
CLSi_ATOM_CHx_IRQ_FORCINT	ATOM[i] channel [x] software interrupt generation	001828_H +i*20000_H+x*80_H	PCLi, 32	PCLi, 32	See 7817	7817
CLSi_ATOM_CHx_IRQ_MODE	ATOM[i] channel [x] interrupt mode configuration register	$00182C_H$ +i*20000_H+x*80_H	PCLi, 32	PCLi, 32	See 7818	7818
CLSi_ATOM_CHx_CTRL2	ATOM[i] channel [x] control2 register	001830_H +i*20000_H+x*80_H	PCLi, 32	PCLi, 32	See 7818	7818
CLSi_ATOM_CHx_CTRL_SR	ATOM[i] channel [x] control shadow register	001834_H +i*20000_H+x*80_H	PCLi, 32	PCLi, 32	See 7819	7819
CLSi_ATOM_AGC_GLB_CTRL	ATOM[i] AGC global control register	$001C40_H$ +i*20000_H	PCLi, 32	PCLi, 32	Kernel Reset	7820
CLSi_ATOM_AGC_ENDIS_CTRL	ATOM[i] AGC enable/disable control register	$001C44_H$ +i*20000_H	PCLi, 32	PCLi, 32	Kernel Reset	7822

(table continues...)

Table 2303 (continued) Register overview - EGTM domain CPU (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CLSi_ATOM_AGC_ENDIS_STAT	ATOM[i] AGC enable/disable status register	001C48$_H$ +i*20000 $_H$	PCLi, 32	PCLi, 32	Kernel Reset	7823
CLSi_ATOM_AGC_ACT_TB	ATOM[i] AGC action time base register	001C4C$_H$ +i*20000 $_H$	PCLi, 32	PCLi, 32	Kernel Reset	7824
CLSi_ATOM_AGC_OUTEN_CTRL	ATOM[i] AGC output enable control register	001C50$_H$ +i*20000 $_H$	PCLi, 32	PCLi, 32	Kernel Reset	7825
CLSi_ATOM_AGC_OUTEN_STAT	ATOM[i] AGC output enable status register	001C54$_H$ +i*20000 $_H$	PCLi, 32	PCLi, 32	Kernel Reset	7826
CLSi_ATOM_AGC_FUPD_CTRL	ATOM[i] AGC force update control register	001C58$_H$ +i*20000 $_H$	PCLi, 32	PCLi, 32	Kernel Reset	7827
CLSi_ATOM_AGC_INT_TRIG	ATOM[i] AGC internal trigger control register	001C5C$_H$ +i*20000 $_H$	PCLi, 32	PCLi, 32	Kernel Reset	7828
CLSi_CCM_HW_CONF2	CCM[i] 2. hardware configuration register	0041D4$_H$ +i*20000 $_H$	PCLi, 32	BE	Kernel Reset	7829
CLSi_CCM_HW_CONF	CCM[i] hardware configuration register	0041DC$_H$ +i*20000 $_H$	PCLi, 32	BE	Kernel Reset	7830
CLSi_CCM_TIM_AUX_IN_SRC	CCM[i] TIM AUX input source register	0041E0$_H$ +i*20000 $_H$	PCLi, 32	PCLi, 32	Kernel Reset	7832
CLSi_CCM_TOM_OUT	CCM[i] TOM output register	0041E8$_H$ +i*20000 $_H$	PCLi, 32	BE	Kernel Reset	7835
CLSi_CCM_ATOM_OUT	CCM[i] ATOM output register	0041EC$_H$ +i*20000 $_H$	PCLi, 32	BE	Kernel Reset	7836
CLSi_CCM_CMU_CLK_CFG	CCM[i] CMU clock configuration register	0041F0$_H$ +i*20000 $_H$	PCLi, 32	PCLi, 32	Kernel Reset	7838
CLSi_CCM_CMU_FXCLK_CFG	CCM[i] CMU fixed clock configuration register	0041F4$_H$ +i*20000 $_H$	PCLi, 32	PCLi, 32	Kernel Reset	7839
CLSi_CCM_CFG	CCM[i] configuration register	0041F8$_H$ +i*20000 $_H$	PCLi, 32	PCLi, 32	Kernel Reset	7839
CLSi_CCM_PROT	CCM[i] protection register	0041FC$_H$ +i*20000 $_H$	PCLi, 32	PCLi, 32	Kernel Reset	7842

(table continues...)

Table 2303 (continued) Register overview - EGTM domain CPU (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CLSi_CDTM_DTMd_CTRL	CDTM[i]_DTM[d] global configuration and control register	004400_H +i*20000 $_H$+d*40$_H$	PCLi, 32	PCLi, 32	Kernel Reset	7843
CLSi_CDTM_DTMd_CH_CTRL1	CDTM[i]_DTM[d] channel control register 1	004404_H +i*20000 $_H$+d*40$_H$	PCLi, 32	PCLi, 32	Kernel Reset	7847
CLSi_CDTM_DTMd_CH_CTRL2	CDTM[i]_DTM[d] channel control register 2	004408_H +i*20000 $_H$+d*40$_H$	PCLi, 32	PCLi, 32	Kernel Reset	7855
CLSi_CDTM_DTMd_CH_CTRL2_SR	CDTM[i] DTM[d] channel control register 2 shadow	$00440C_H$ +i*20000 $_H$+d*40$_H$	PCLi, 32	PCLi, 32	Kernel Reset	7858
CLSi_CDTM_DTMd_PS_CTRL	CDTM[i]_DTM[d] phase shift unit configuration and control register	004410_H +i*20000 $_H$+d*40$_H$	PCLi, 32	PCLi, 32	Kernel Reset	7861
CLSi_CDTM_DTMd_CH_DTVx	CDTM[i]_DTM[d] channel [x] dead-time reload values	004414_H +i*20000 $_H$+d*40$_H$ +x*4	PCLi, 32	PCLi, 32	Kernel Reset	7862
CLSi_CDTM_DTMd_CH_SR	CDTM[i]_DTM[d] channel shadow register	004424_H +i*20000 $_H$+d*40$_H$	PCLi, 32	PCLi, 32	Kernel Reset	7863
CLSi_CDTM_DTMd_CH_CTRL3	CDTM[i]_DTM[d] channel control register 3	004428_H +i*20000 $_H$+d*40$_H$	PCLi, 32	PCLi, 32	Kernel Reset	7864
CLSi_CDTM_DTMd_CTRL2	CDTM[i]_DTM[d] global configuration and control register 2	$00442C_H$ +i*20000 $_H$+d*40$_H$	PCLi, 32	PCLi, 32	Kernel Reset	7866
CLSi_CDTM_DTMd_CH_DTV_SRx	CDTM[i]_DTM[d] channel [x] dead-time shadow values	004430_H +i*20000 $_H$+d*40$_H$ +x*4	PCLi, 32	PCLi, 32	Kernel Reset	7874
CLSi_SPE_CTRL_STAT	SPE[i] control status register	$004C00_H$ +i*20000 $_H$	PCLi, 32	PCLi, 32	Kernel Reset	7875
CLSi_SPE_PAT	SPE[i] input pattern definition register	$004C04_H$ +i*20000 $_H$	PCLi, 32	PCLi, 32	Kernel Reset	7882
CLSi_SPE_OUT_PATp	SPE[i] output definition register [p]	$004C08_H$ +i*20000 $_H$+p*4	PCLi, 32	PCLi, 32	Kernel Reset	7882
CLSi_SPE_OUT_CTRL	SPE[i] output control register	$004C28_H$ +i*20000 $_H$	PCLi, 32	PCLi, 32	Kernel Reset	7883

(table continues...)

Table 2303 (continued) Register overview - EGTM domain CPU (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CLSi_SPE_IRQ_NOTIFY	SPE[i] interrupt notification register	$004C2C_H$ +i*20000 H	PCLi, 32	PCLi, 32	Kernel Reset	7883
CLSi_SPE_IRQ_EN	SPE[i] interrupt enable register	$004C30_H$ +i*20000 H	PCLi, 32	PCLi, 32	Kernel Reset	7885
CLSi_SPE_IRQ_FORCINT	SPE[i] Interrupt Generation By Software	$004C34_H$ +i*20000 H	PCLi, 32	PCLi, 32	Kernel Reset	7886
CLSi_SPE_IRQ_MODE	SPE[i] interrupt mode configuration register	$004C38_H$ +i*20000 H	PCLi, 32	PCLi, 32	Kernel Reset	7887
CLSi_SPE_EIRQ_EN	SPE[i] error interrupt enable register	$004C3C_H$ +i*20000 H	PCLi, 32	PCLi, 32	Kernel Reset	7888
CLSi_SPE_REV_CNT	SPE[i] Input Revolution Counter	$004C40_H$ +i*20000 H	PCLi, 32	PCLi, 32	Kernel Reset	7889
CLSi_SPE_REV_CMP	SPE[i] Revolution Counter Compare Value	$004C44_H$ +i*20000 H	PCLi, 32	PCLi, 32	Kernel Reset	7889
CLSi_SPE_CTRL_STAT2	SPE[i] control status register 2	$004C48_H$ +i*20000 H	PCLi, 32	PCLi, 32	Kernel Reset	7890
CLSi_SPE_CMD	SPE[i] command register	$004C4C_H$ +i*20000 H	PCLi, 32	PCLi, 32	Kernel Reset	7891
CLC	Clock control register	060000_H	P	P, SV, E	Application Reset	7892
OCS	OCDS control and status register	060004_H	P	SV, P	Debug Reset	7893
RST_CTRLA	Reset control register A	060010_H	P	P, SV, E	Application Reset	7894
RST_CTRLB	Reset control register B	060014_H	P	P, SV, E	Application Reset	7894
RST_STAT	Reset status register	060018_H	P	BE	Application Reset	7895
PROTSE	PROT register Safe Endinit	$06001C_H$	U	SV, PROT	Application Reset	7896
CLi_ACCEN_WRA	Write access enable register A for cluster i	060024_H +i*24_H	U	SE, SV	Application Reset	7897
CLi_ACCEN_WRB	Write access enable register B for cluster i	060028_H +i*24_H	U	SE, SV	Application Reset	7898

(table continues...)

Table 2303 (continued) Register overview - EGTM domain CPU (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CLi_ACCEN_RDA	Read access enable register A for cluster i	$06002C_H$ +i*24_H	U	SE, SV	Application Reset	7898
CLi_ACCEN_RDB	Read access enable register B for cluster i	060030_H +i*24_H	U	SE, SV	Application Reset	7899
CLi_ACCEN_VM	VM access enable register for cluster i	060034_H +i*24_H	U	SE, SV	Application Reset	7899
CLi_ACCEN_PRS	PRS access enable register for cluster i	060038_H +i*24_H	U	SE, SV	Application Reset	7900
CTRL_PROTE	PROT register Endinit for control logic	$0601D0_H$	U	SV, PROT	Application Reset	7901
CTRL_ACCEN_WRA	Write access enable register A for control logic	$0601D4_H$	U	SE, SV	Application Reset	7902
CTRL_ACCEN_WRB	Write access enable register B for control logic	$0601D8_H$	U	SE, SV	Application Reset	7903
CTRL_ACCEN_RDA	Read access enable register A for control logic	$0601DC_H$	U	SE, SV	Application Reset	7903
CTRL_ACCEN_RDB	Read access enable register B for control logic	$0601E0_H$	U	SE, SV	Application Reset	7904
CTRL_ACCEN_VM	VM access enable register for control logic	$0601E4_H$	U	SE, SV	Application Reset	7904
CTRL_ACCEN_PRS	PRS access enable register for control logic	$0601E8_H$	U	SE, SV	Application Reset	7905
WRAP_ACCEN_WRA	Write access enable register A for wrapper logic	$0601F8_H$	U	SE, SV	Application Reset	7906
WRAP_ACCEN_WRB	Write access enable register B for wrapper logic	$0601FC_H$	U	SE, SV	Application Reset	7906
WRAP_ACCEN_RDA	Read access enable register A for wrapper logic	060200_H	U	SE, SV	Application Reset	7907
WRAP_ACCEN_RDB	Read access enable register B for wrapper logic	060204_H	U	SE, SV	Application Reset	7907
WRAP_ACCEN_VM	VM access enable register for wrapper logic	060208_H	U	SE, SV	Application Reset	7908
WRAP_ACCEN_PRS	PRS access enable register for wrapper logic	$06020C_H$	U	SE, SV	Application Reset	7908
CTRLGTM_ACCEN_WRA	Write access enable register A for wrapper logic	$06021C_H$	U	SE, SV	Application Reset	7909
CTRLGTM_ACCEN_WRB	Write access enable register B for wrapper logic	060220_H	U	SE, SV	Application Reset	7909
CTRLGTM_ACCEN_RDA	Read access enable register A for wrapper logic	060224_H	U	SE, SV	Application Reset	7910
CTRLGTM_ACCEN_RDB	Read access enable register B for wrapper logic	060228_H	U	SE, SV	Application Reset	7910

(table continues...)

Table 2303 (continued) Register overview - EGTM domain CPU (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CTRLGTM_ACCEN_VM	VM access enable register for wrapper logic	$06022C_H$	U	SE, SV	Application Reset	7911
CTRLGTM_ACCEN_PRS	PRS access enable register for wrapper logic	060230_H	U	SE, SV	Application Reset	7912
SAFEGTM_ACCEN_WRA	Write access enable register A for wrapper logic	060240_H	U	SE, SV	Application Reset	7912
SAFEGTM_ACCEN_WRB	Write access enable register B for wrapper logic	060244_H	U	SE, SV	Application Reset	7913
SAFEGTM_ACCEN_RDA	Read access enable register A for wrapper logic	060248_H	U	SE, SV	Application Reset	7913
SAFEGTM_ACCEN_RDB	Read access enable register B for wrapper logic	$06024C_H$	U	SE, SV	Application Reset	7914
SAFEGTM_ACCEN_VM	VM access enable register for wrapper logic	060250_H	U	SE, SV	Application Reset	7914
SAFEGTM_ACCEN_PRS	PRS access enable register for wrapper logic	060254_H	U	SE, SV	Application Reset	7915
SPARE	Spare Bits	060480_H	U	U	Kernel Reset	7916
OCDS_OTBU0T	OCDS TBU0 trigger register	060500_H	P	P, SV	Debug Reset	7917
OCDS_OTBU1T	OCDS TBU1 trigger register	060504_H	P	P, SV	Debug Reset	7917
OCDS_OTBU2T	OCDS TBU2 trigger register	060508_H	P	P, SV	Debug Reset	7918
OCDS_OTSS	OCDS trigger set select register	060510_H	P	P, SV	Debug Reset	7620
OCDS_OTSC0	OCDS trigger set control 0 register	060514_H	P	P, SV	Debug Reset	7919
TIMnINSEL	TIMn input selection register	061000_H +n*4	PWRAP	PWRAP, SV	Kernel Reset	7583
TIMnINSELB	TIMn input selection register group B	061050_H +n*4	PWRAP	PWRAP, SV	Kernel Reset	7585
TOUTSELn	eGTM to Ports signal TOUT output selection register n	061100_H +n*4	PWRAP	PWRAP, SV	Kernel Reset	7579
MSCSETi_CONj	eGTM to MSC SETi output selection register j	061200_H +i*10_H+j *4	PWRAP	PWRAP, SV	Kernel Reset	7608
MSCn_INLCON	eGTM to input bus MSCn_INLCON signal selection register	061290_H +n*10_H	PWRAP	PWRAP, SV	Kernel Reset	7610
MSCn_INHCON	eGTM to input bus MSCn_INHCON signal selection register	061294_H +n*10_H	PWRAP	PWRAP, SV	Kernel Reset	7611
MSCn_INLECON	eGTM to input bus MSCn_INLECON signal selection register	061298_H +n*10_H	PWRAP	PWRAP, SV	Kernel Reset	7610

(table continues...)

Table 2303 (continued) Register overview - EGTM domain CPU (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
MSCn_INHECON	eGTM to input bus MSCn_INHECON signal selection register	$06129C_H$ +n*10_H	PWRAP	PWRAP, SV	Kernel Reset	7611
MSCSELIN_INLCON	Selection of INLCON stream for MSC	$0612D0_H$	PWRAP	PWRAP, SV	Kernel Reset	7612
MSCSELIN_INHCON	Selection of INHCON stream for MSC	$0612D4_H$	PWRAP	PWRAP, SV	Kernel Reset	7613
MSCSELIN_INLECON	Selection of INLECON stream for MSC	$0612D8_H$	PWRAP	PWRAP, SV	Kernel Reset	7612
MSCSELIN_INHECON	Selection of INHECON stream for MSC	$0612DC_H$	PWRAP	PWRAP, SV	Kernel Reset	7613
PSI5_OUT	eGTM to PSI5 output trigger selection register	061400_H	PWRAP	PWRAP, SV	Kernel Reset	7615
PSI5Sn_OUT	eGTM to PSI5-Sn output trigger selection register	061410_H +n*4	PWRAP	PWRAP, SV	Kernel Reset	7616
LCDCDC_OUT	eGTM to LCDCDC output trigger selection register	061420_H	PWRAP	PWRAP, SV	Kernel Reset	7598
CANn_OUTx	eGTM to CANn output selection register x. n*4+ x<24 , if interface exists.	061480_H +n*10_H+ x*4	PWRAP	PWRAP, SV	Kernel Reset	7599
FOUTENCx	FOUT Selection for Encoder Interface x, if GPT resources exist.	061500_H +x*4	PWRAP	PWRAP, SV	Kernel Reset	7589
QSPIn_OUT	eGTM to QSPIn output selection register. n<8 , if interface exists.	061580_H +n*4	PWRAP	PWRAP, SV	Kernel Reset	7602
ASCLINn_OUT	eGTM to ASCLINn output selection register. n<28 , if interface exists.	061600_H +n*4	PWRAP	PWRAP, SV	Kernel Reset	7601
SENTn_OUTA	eGTM to SENTn trigger x output selection register A	061680_H +n*8	PWRAP	PWRAP, SV	Kernel Reset	7617
ADC_OUTn	eGTM to ADC output selection register n	061700_H +n*4	PWRAP	PWRAP, SV	Kernel Reset	
DTMAUXn_INy	Cluster n DTMAUX input y selection register	061780_H +n*8+y* 4	PWRAP	PWRAP, SV	Kernel Reset	7596
SCU_OUTn	eGTM to SCU output selection register n eGTM_SCU_TRIG(n)	061800_H +n*4	PWRAP	PWRAP, SV	Kernel Reset	7593
TIMn_ADCSELCHi	Instance Selection for TIM[n] channel[i]	$061B00_H$ +n*20_H+i *4	PWRAP	PWRAP, SV	Kernel Reset	7586

(table continues...)

Table 2303 (continued) Register overview - EGTM domain CPU (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
HRPWMMUXz_CHSEL	HRPWMz 16:1 Channel MUX select	062000_H +z*4	PCLz	PCLz	Kernel Reset	7936
HRPWMz_EN	HRPWMz module and channel enable register	062400_H +z*400_H	PCLz	PCLz	Kernel Reset	7937

44.5.3 eGTM version control register (i=0)

***Note**: The numbers are encoded in BCD. Values "A" - "F" are characters.*

CLSi_ARCH_REV (i=0) Offset address: 000000_H + i*20000_H

eGTM version control register Kernel Reset value: 4121 $00C0_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
VER_MAJOR				VER_MINOR				DEVICE_CODE				VENDOR_CODE			
r				r				r				r			

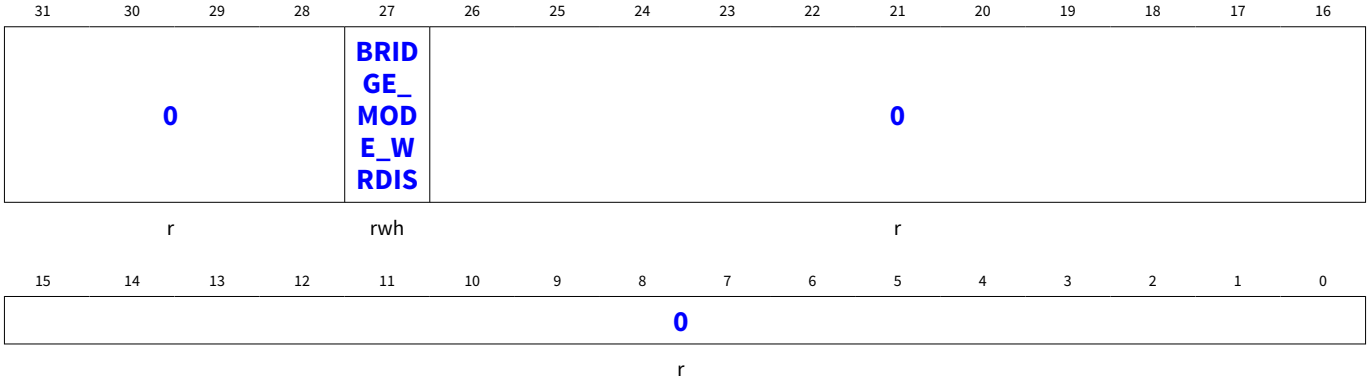
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				REL_BASE								REL_ITER			
r				r								r			

Field	Bits	Type	Description
REL_ITER	3:0	r	**Delivery number** Defines **eGTM** release iteration.
REL_BASE	11:4	r	**Release step** Defines **eGTM** release base.
VENDOR_CODE	19:16	r	**Device encoding digit 1** Defines vendor code.
DEVICE_CODE	23:20	r	**Device encoding digit 0** Defines device code.
VER_MINOR	27:24	r	**Minor version number** Defines minor version number of **eGTM** specification.
VER_MAJOR	31:28	r	**Major version number** Defines major version number of **eGTM** specification.
0	15:12	r	**Reserved** Read as 0.

44.5.4 eGTM global reset register (i=0)

CLSi_ARCH_RST (i=0) Offset address: 000004_H+i*20000_H

eGTM global reset register Kernel Reset value: 0000 0000_H

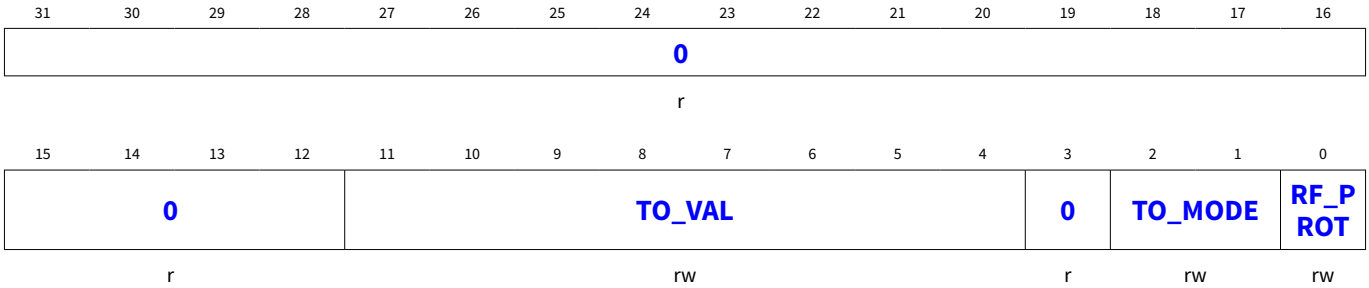


Field	Bits	Type	Description
BRIDGE_MODE_WRDIS	27	rwh	CLSi_AEI_BRIDGE_MODE **write disable** ***Note**: This bit is write-protected by (CLSi_ARCH_CTRL.RF_PROT==1 && AEI_WDATA[27:27] != CLSi_ARCH_RST.BRIDGE_MODE_WRDIS && AEI_W1R0==1).* 0_B Writing of CLSi_AEI_BRIDGE_MODE register is enabled 1_B Writing of CLSi_AEI_BRIDGE_MODE register is disabled
0	26:0, 31:28	r	**Reserved** Read as 0; must be written with 0.

44.5.5 eGTM global control register (i=0)

CLSi_ARCH_CTRL (i=0) Offset address: 000008_H+i*20000_H

eGTM global control register Kernel Reset value: $0000\ 0001_H$



Field	Bits	Type	Description
RF_PROT	0	rw	**RST and FORCINT protection** 0_B SW RST (global), SW interrupt FORCINT, and SW RAM reset functionality is enabled 1_B SW RST (global), SW interrupt FORCINT, and SW RAM reset functionality is disabled

(table continues...)

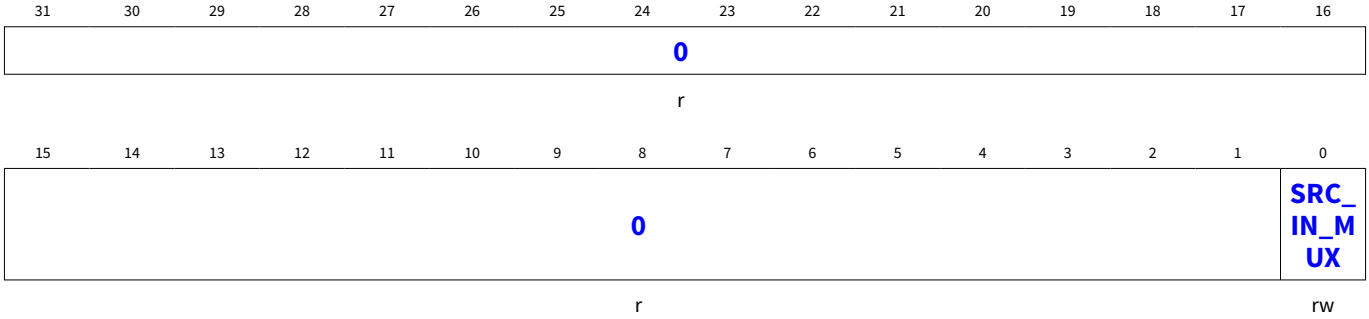
(continued)

Field	Bits	Type	Description
TO_MODE	2:1	rw	**AEI timeout mode** 00_B Observe once: If condition timeout_counter=CLSi_ARCH_CTRL.TO_VAL occurs the address AEI_ADDR and AEI_W1R0 signal will be stored to the CLSi_ARCH_AEI_ADDR_XPT register, the timeout flag in CLSi_ARCH_IRQ_NOTIFY.AEI_TO_XPT will be set. Following timeout_counter = CLSi_ARCH_CTRL.TO_VAL accesses will not overwrite the first entry in the CLSi_ARCH_AEI_ADDR_XPT register. Clearing the timeout flag CLSi_ARCH_IRQ_NOTIFY.AEI_TO_XPT will re-enable the store the next access for which the condition timeout_counter=CLSi_ARCH_CTRL.TO_VAL is fulfilled. 01_B Abort once: Same behavior as defined for observe once. Additionally, if condition timeout_counter=CLSi_ARCH_CTRL.TO_VAL occurs the pending access will be aborted by signaling an illegal module access on aei_status and ready to be sent. In case of a read-access AEI_RDATA=0 is responded. Following accesses will not be aborted, this could lead to a dead lock if the access will never terminate with AEI_READY=1. 10_B Retry: Same behavior as defined for observe once. Additionally, if condition timeout_counter=CLSi_ARCH_CTRL.TO_VAL occurs each pending access is paused by signaling AEI_SEL=0 for 1 clock-cycle, afterwards the access is active again. This leads to a retry of the actual access until the target address responds with AEI_READY=1. 11_B Abort: Same behavior as defined for observe once. Additionally, if condition timeout_counter=CLSi_ARCH_CTRL.TO_VAL occurs each pending access will be aborted by signaling an illegal module access on aei_status and ready to be sent. In case of a read-access AEI_RDATA=0 is responded.
TO_VAL	11:4	rw	**AEI timeout value** ***Note**: These bits define the number of cycles after which a timeout event occurs. When CLSi_ARCH_CTRL.TO_VAL equals zero (0) the **AEI** timeout functionality is disabled.*
0	3, 15:12, 31:16	r	**Reserved** Read as 0; must be written with 0.

44.5.6 eGTM configuration register (i=0)

CLSi_ARCH_CFG (i=0) Offset address: $00000C_H+i*20000_H$

eGTM configuration register Kernel Reset value: $0000\ 0000_H$

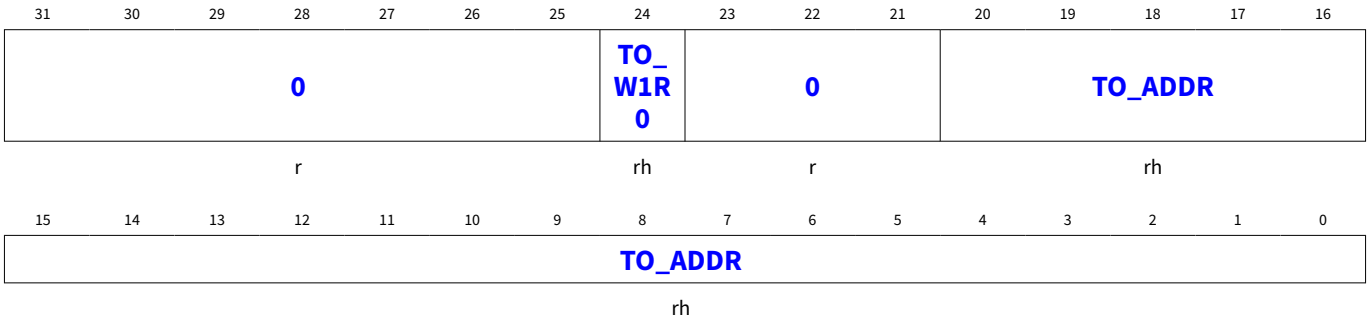


Field	Bits	Type	Description
SRC_IN_MUX	0	rw	**Input source selection for signal** TIM[i]_AUX_IN **(input port** AUX_IN **at module TIM)** See Figure 1018 for details. 0_B Use for TIM[i] channel j the output of TOM[k] channel j+h; k = floor(i / 2); h = mod(i , 2) *8 (j∈{0, 1,..., 7}) 1_B Use for TIM[i] channel j the output of TOM[i] channel j (TIM and TOM are in the same cluster) (j∈{0, 1,. .., 7})
0	31:1	r	**Reserved** Read as 0; must be written with 0.

44.5.7 eGTM AEI timeout exception address register (i=0)

CLSi_ARCH_AEI_ADDR_XPT (i=0) Offset address: $000010_H+i*20000_H$

eGTM AEI timeout exception address register Reset values see: Table 2304



Field	Bits	Type	Description
TO_ADDR	20:0	rh	**AEI timeout address** ***Note:*** *This bit field defines the **AEI** address for which the **AEI** timeout event occurred.*

(table continues...)

(continued)

Field	Bits	Type	Description
TO_W1R0	24	rh	**AEI timeout Read/Write flag** ***Note**: This bit defines the **AEI** Read/Write flag for which the **AEI** timeout event occurred.* 0_B timeout occurred for read transaction 1_B timeout occurred for write transaction
0	23:21, 31:25	r	**Reserved** Read as 0.

Table 2304 Reset values of CLSi_ARCH_AEI_ADDR_XPT (i=0)

Reset	Reset value	Note
Kernel Reset	0000 0000$_H$	
Bridge reset	0000 0000$_H$	

44.5.8 eGTM AEI non-zero status register (i=0)

CLSi_ARCH_AEI_STA_XPT (i=0) Offset address: 000014$_H$+i*20000$_H$

eGTM AEI non-zero status register Reset values see: Table 2305

Field	Bits	Type	Description
ADDR	20:0	rh	**AEI exception address** ***Note**: This bit field captures the address of the first **AEI** access resulting with a non-zero **AEI** status signal. The bit field can be cleared by clearing the interrupt flags* CLSi_ARCH_IRQ_NOTIFY**.**AEI_USP_ADDR*,* CLSi_ARCH_IRQ_NOTIFY**.**AEI_USP_BE*, and* CLSi_ARCH_IRQ_NOTIFY**.**AEI_IM_ADDR ***Note**: Exception: The address will not be captured in case a protected write (*CLSi_ARCH_RST**.**BRIDGE_MODE_WRDIS *= 1) to register* CLSi_AEI_BRIDGE_MODE *is performed.*

(table continues...)

(continued)

Field	Bits	Type	Description
W1R0	24	rh	**AEI exception Read/Write flag** ***Note**: This bit field captures the address of the first **AEI** access resulting with a non-zero **AEI** status signal. The bit field can be cleared by clearing the interrupt flags CLSi_ARCH_IRQ_NOTIFY.AEI_USP_ADDR, CLSi_ARCH_IRQ_NOTIFY.AEI_USP_BE, and CLSi_ARCH_IRQ_NOTIFY.AEI_IM_ADDR.* ***Note**: Exception: This bit will not be set in case a protected write (CLSi_ARCH_RST.BRIDGE_MODE_WRDIS = 1) to register CLSi_AEI_BRIDGE_MODE is performed.* 0_B Non-zero status occurred for read transaction 1_B Non-zero status occurred for write transaction
0	23:21, 31:25	r	**Reserved** Read as 0.

Table 2305 Reset values of CLSi_ARCH_AEI_STA_XPT (i=0)

Reset	Reset value	Note
Kernel Reset	$0000\ 0000_H$	
Bridge reset	$0000\ 0000_H$	

44.5.9 eGTM Interrupt notification register (i=0)

CLSi_ARCH_IRQ_NOTIFY (i=0) Offset address: $000018_H+i*20000_H$

eGTM Interrupt notification register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		CLK_EN_EXP_STATE		0		CLK_EN_ERR_STATE		0							
r		rh		r		rh		r							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0							CLK_PER_ERR	CLK_EN_ERR	0	0	0	AEI_USP_BE	AEI_IM_ADDR	AEI_USP_ADDR	AEI_TO_XPT
r							rw1ch	rw1ch	r	r	r	rw1ch	rw1ch	rw1ch	rw1ch

Field	Bits	Type	Description
AEI_TO_XPT	0	rw1ch	**AEI timeout exception occurred** ***Note**: This bit will be cleared on a CPU write-access of value '1'. A read-access leaves the bit unchanged.* **If reading:** 0_B No interrupt raised 1_B Interrupt was raised **If writing:** 0_B No action 1_B Clear interrupt
AEI_USP_ADDR	1	rw1ch	**AEI unsupported address interrupt** ***Note**: This bit will be cleared on a CPU write-access of value '1'. A read-access leaves the bit unchanged.* **If reading:** 0_B No interrupt raised 1_B Interrupt was raised **If writing:** 0_B No action 1_B Clear interrupt
AEI_IM_ADDR	2	rw1ch	**AEI illegal Module address interrupt** ***Note**: This bit will be cleared on a CPU write-access of value '1'. A read-access leaves the bit unchanged.* ***Note**: Exception: This bit will not be set in case a protected write (CLSi_ARCH_RST.BRIDGE_MODE_WRDIS = 1) to register CLSi_AEI_BRIDGE_MODE is performed.* **If reading:** 0_B No interrupt raised 1_B Interrupt was raised **If writing:** 0_B No action 1_B Clear Interrupt
AEI_USP_BE	3	rw1ch	**AEI unsupported byte enable interrupt** ***Note**: This bit will be cleared on a CPU write-access of value '1'. A read-access leaves the bit unchanged.* **If reading:** 0_B No interrupt raised 1_B Interrupt was raised **If writing:** 0_B No action 1_B Clear Interrupt

(table continues...)

(continued)

Field	Bits	Type	Description
CLK_EN_ERR	7	rw1ch	**Clock enable error interrupt** ***Note**: This bit will be cleared on a CPU write-access of value '1'. A read-access leaves the bit unchanged.* **If reading:** 0_B No interrupt raised 1_B Interrupt was raised **If writing:** 0_B No action 1_B Clear interrupt
CLK_PER_ERR	8	rw1ch	**Clock period error interrupt** ***Note**: This bit will be cleared on a CPU write-access of value '1'. A read-access leaves the bit unchanged.* **If reading:** 0_B No interrupt raised 1_B Interrupt was raised **If writing:** 0_B No action 1_B Clear interrupt
CLK_EN_ERR_STATE	25:24	rh	**Erroneous clock enable state** This bit field defines the **eGTM** external clk enable state at occurrence of the CLSi_ARCH_IRQ_NOTIFY.CLK_EN_ERR event. 00_B disable state for external clock CLK / 2; disable state for external clock CLK 01_B disable state for external clock CLK/ 2; enable state for external clock CLK 10_B enable state for external clock CLK/ 2; disable state for external clock CLK 11_B enable state for external clock CLK/ 2; enable state for external clock CLK
CLK_EN_EXP_STATE	29:28	rh	**Expected clock enable state** This bit field defines the **eGTM** expected clk enable state at occurrence of the CLSi_ARCH_IRQ_NOTIFY.CLK_EN_ERR event. 00_B Disable state expected for external clock CLK/ 2; disable state expected for external clock CLK 01_B Disable state expected for external clock CLK/ 2; enable state expected for external clock CLK 10_B Enable state expected for external clock CLK/ 2; disable state expected for external clock CLK 11_B Enable state expected for external clock CLK/ 2; enable state expected for external clock CLK

(table continues...)

(continued)

Field	Bits	Type	Description
0	4, 5, 6, 23:9, 27:26, 31:30	r	**Reserved** Read as 0; must be written with 0.

44.5.10 eGTM interrupt enable register (i=0)

CLSi_ARCH_IRQ_EN (i=0) Offset address: $00001C_H+i*20000_H$

eGTM interrupt enable register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0							CLK_PER_ERR_IRQ_EN	CLK_EN_ERR_IRQ_EN	0	0	0	AEI_USP_BE_IRQ_EN	AEI_IM_ADDR_IRQ_EN	AEI_USP_ADDR_IRQ_EN	AEI_TO_XPT_IRQ_EN
r							rw	rw	r	r	r	rw	rw	rw	rw

Field	Bits	Type	Description
AEI_TO_XPT_IRQ_EN	0	rw	AEI_TO_XPT_IRQ **interrupt enable** 0_B Disable interrupt, interrupt is not visible outside eGTM 1_B Enable interrupt, interrupt is visible outside eGTM
AEI_USP_ADDR_IRQ_EN	1	rw	AEI_USP_ADDR_IRQ **interrupt enable** 0_B Disable interrupt, interrupt is not visible outside eGTM 1_B Enable interrupt, interrupt is visible outside eGTM
AEI_IM_ADDR_IRQ_EN	2	rw	AEI_IM_ADDR_IRQ **interrupt enable** 0_B Disable interrupt, interrupt is not visible outside eGTM 1_B Enable interrupt, interrupt is visible outside eGTM
AEI_USP_BE_IRQ_EN	3	rw	AEI_USP_BE_IRQ **interrupt enable** 0_B Disable interrupt, interrupt is not visible outside eGTM 1_B Enable interrupt, interrupt is visible outside eGTM
CLK_EN_ERR_IRQ_EN	7	rw	CLK_EN_ERR_IRQ **interrupt enable** 0_B Disable interrupt, interrupt is not visible outside eGTM 1_B Enable interrupt, interrupt is visible outside eGTM

(table continues...)

(continued)

Field	Bits	Type	Description
CLK_PER_ERR_IRQ_EN	8	rw	CLK_PER_ERR_IRQ **interrupt enable** 0_B Disable interrupt, interrupt is not visible outside eGTM 1_B Enable interrupt, interrupt is visible outside eGTM
0	4, 5, 6, 31:9	r	**Reserved** Read as 0; must be written with 0.

44.5.11 eGTM error interrupt enable register (i=0)

CLSi_ARCH_EIRQ_EN (i=0) Offset address: 000020_H+i*20000_H

eGTM error interrupt enable register Kernel Reset value: $0000\ 0180_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0							CLK_PER_ERR_EIRQ_EN	CLK_EN_ERR_EIRQ_EN	0	0	0	AEI_USP_BE_EIRQ_EN	AEI_IM_ADDR_EIRQ_EN	AEI_USP_ADDR_EIRQ_EN	AEI_TO_XPT_EIRQ_EN
r							rw	rw	r	r	r	rw	rw	rw	rw

Field	Bits	Type	Description
AEI_TO_XPT_EIRQ_EN	0	rw	AEI_TO_XPT_EIRQ **error interrupt enable** 0_B Disable error interrupt, interrupt is not visible outside eGTM 1_B Enable error interrupt, interrupt is visible outside eGTM
AEI_USP_ADDR_EIRQ_EN	1	rw	AEI_USP_ADDR_EIRQ **error interrupt enable** 0_B Disable error interrupt, interrupt is not visible outside eGTM 1_B Enable error interrupt, interrupt is visible outside eGTM
AEI_IM_ADDR_EIRQ_EN	2	rw	AEI_IM_ADDR_EIRQ **error interrupt enable** 0_B Disable error interrupt, interrupt is not visible outside eGTM 1_B Enable error interrupt, interrupt is visible outside eGTM
AEI_USP_BE_EIRQ_EN	3	rw	AEI_USP_BE_EIRQ **error interrupt enable** 0_B Disable error interrupt, interrupt is not visible outside eGTM 1_B Enable error interrupt, interrupt is visible outside eGTM
CLK_EN_ERR_EIRQ_EN	7	rw	CLK_EN_ERR_EIRQ **interrupt enable** 0_B Disable interrupt, interrupt is not visible outside eGTM 1_B Enable interrupt, interrupt is visible outside eGTM

(table continues...)

(continued)

Field	Bits	Type	Description
CLK_PER_ERR_EIRQ_EN	8	rw	CLK_PER_ERR_EIRQ **interrupt enable** 0_B Disable interrupt, interrupt is not visible outside eGTM 1_B Enable interrupt, interrupt is visible outside eGTM
0	4, 5, 6, 31:9	r	**Reserved** Read as 0; must be written with 0.

44.5.12 eGTM Software interrupt generation register (i=0)

CLSi_ARCH_IRQ_FORCINT (i=0) Offset address: $000024_H+i*20000_H$

eGTM Software interrupt generation register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0							TRG_CLK_PER_ERR	TRG_CLK_EN_ERR	0	0	0	TRG_AEI_USP_BE	TRG_AEI_IM_ADDR	TRG_AEI_USP_ADDR	TRG_AEI_TO_XPT
r							rw	rw	r	r	r	rw	rw	rw	rw

Field	Bits	Type	Description
TRG_AEI_TO_XPT	0	rw	**Trigger the bit** CLSi_ARCH_IRQ_NOTIFY**.**AEI_TO_XPT **by software.** ***Note**:* *This bit is cleared automatically after write.* ***Note**:* *This bit is write-protected by (*CLSi_ARCH_CTRL**.**RF_PROT*==1 &&* AEI_WDATA*[8:0] != 0 &&* AEI_W1R0*==1).* **If reading:** 0_B No status **If writing:** 0_B No action 1_B Force setting of CLSi_ARCH_IRQ_NOTIFY.AEI_TO_XPT

(table continues...)

(continued)

Field	Bits	Type	Description
TRG_AEI_USP_ADDR	1	rw	**Trigger the bit** CLSi_ARCH_IRQ_NOTIFY.AEI_USP_ADDR **by software.** ***Note**: This bit is cleared automatically after write.* ***Note**: This bit is write-protected by (CLSi_ARCH_CTRL.RF_PROT==1 && AEI_WDATA[8:0] != 0 && AEI_W1R0==1).* **If reading:** 0_B No status **If writing:** 0_B No action 1_B Force setting of CLSi_ARCH_IRQ_NOTIFY.AEI_USP_ADDR
TRG_AEI_IM_ADDR	2	rw	**Trigger the bit** CLSi_ARCH_IRQ_NOTIFY.AEI_IM_ADDR **by software.** ***Note**: This bit is cleared automatically after write.* ***Note**: This bit is write-protected by (CLSi_ARCH_CTRL.RF_PROT==1 && AEI_WDATA[8:0] != 0 && AEI_W1R0==1).* **If reading:** 0_B No status **If writing:** 0_B No action 1_B Force setting of CLSi_ARCH_IRQ_NOTIFY.AEI_IM_ADDR
TRG_AEI_USP_BE	3	rw	**Trigger the bit** CLSi_ARCH_IRQ_NOTIFY.AEI_USP_BE **by software.** ***Note**: This bit is cleared automatically after write.* ***Note**: This bit is write-protected by (CLSi_ARCH_CTRL.RF_PROT==1 && AEI_WDATA[8:0] != 0 && AEI_W1R0==1).* **If reading:** 0_B No status **If writing:** 0_B No action 1_B Force setting of CLSi_ARCH_IRQ_NOTIFY.AEI_USP_BE

(table continues...)

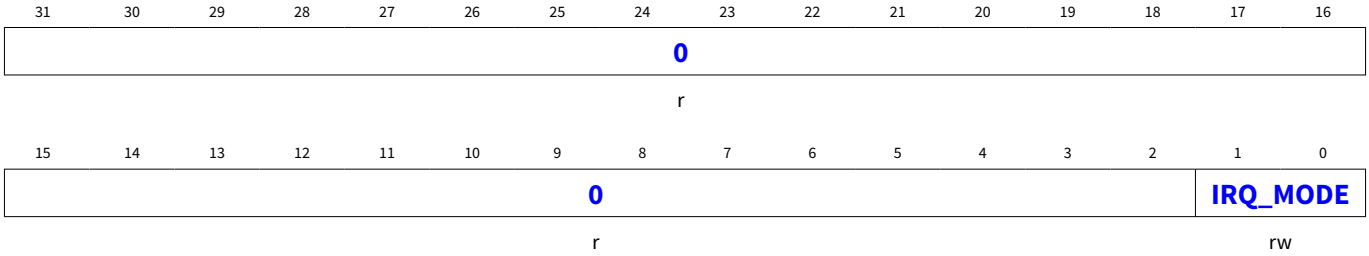
(continued)

Field	Bits	Type	Description
TRG_CLK_EN_ERR	7	rw	**Trigger the bit** CLSi_ARCH_IRQ_NOTIFY.CLK_EN_ERR **by software.** ***Note**: This bit is cleared automatically after write.* ***Note**: This bit is write-protected by (CLSi_ARCH_CTRL.RF_PROT==1 && AEI_WDATA[8:0] != 0 && AEI_W1R0==1).* **If reading:** 0_B No status **If writing:** 0_B No action 1_B Force setting of CLSi_ARCH_IRQ_NOTIFY.CLK_EN_ERR
TRG_CLK_PER_ERR	8	rw	**Trigger the bit** CLSi_ARCH_IRQ_NOTIFY.CLK_PER_ERR **by software.** ***Note**: This bit is cleared automatically after write.* ***Note**: This bit is write-protected by (CLSi_ARCH_CTRL.RF_PROT==1 && AEI_WDATA[8:0] != 0 && AEI_W1R0==1).* **If reading:** 0_B No status **If writing:** 0_B No action 1_B Force setting of CLSi_ARCH_IRQ_NOTIFY.CLK_PER_ERR
0	4, 5, 6, 31:9	r	**Reserved** Read as 0; must be written with 0.

44.5.13 eGTM toplevel interrupts mode selection (i=0)

CLSi_ARCH_IRQ_MODE (i=0) Offset address: 000028_H+i*20000_H

eGTM toplevel interrupts mode selection Kernel Reset value: $0000\ 0002_H$



Field	Bits	Type	Description
IRQ_MODE	1:0	rw	**Interrupt strategy mode selection for the AEI timeout and address monitoring interrupts.** ***Note**: The interrupt modes are described in **eGTM** interrupt concept.* ***Note**: This mode selection is only valid for the interrupts described in register CLSi_ARCH_IRQ_NOTIFY.* 00_B Level mode 01_B Pulse mode 10_B Pulse-Notify mode 11_B Single-Pulse mode
0	31:2	r	**Reserved** Read as 0; must be written with 0.

44.5.14 eGTM Cluster Clock Configuration (i=0)

CLSi_ARCH_CLK_CFG (i=0) Offset address: $00002C_H+i*20000_H$

eGTM Cluster Clock Configuration Kernel Reset value: $0000\ 002A_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								0		0		0		0	
r								r		r		r		r	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		0		0		0		0		CLS2_CLK_DIV		CLS1_CLK_DIV		CLS0_CLK_DIV	
r		r		r		r		r		rwh		rwh		rwh	

Field	Bits	Type	Description
CLSj_CLK_DIV (j=0-2)	2*j+1:2*j	rwh	**Cluster [j] Clock Divider** ***Note**: These bits are only writable if bit field CLSi_ARCH_CTRL.RF_PROT is cleared.* ***Note**: Writing a reserved value will not change the bit field value.* ***Note**: The configuration value CLSi_ARCH_CLK_CFG.CLSj_CLK_DIV **(j=0)** defines the primary input clock period for **CMU**.* ***Note**: If CLSi_ARCH_CLK_CFG.CLSj_CLK_DIV **(j=0)** is configured to a value 10_B (i.e. clock divider 2), the maximum **CMU** resolution signal frequency for all other cluster k=1..n is also limited to configured **CMU** resolution frequency of cluster 0.* 00_B Cluster is disabled 01_B Cluster is enabled without clock divider 10_B Cluster is enabled with clock divider 2
0	23:22, 21:20, 19:18, 17:16, 15:14, 13:12, 11:10, 9:8, 7:6, 31:24	r	**Reserved** Read as 0; must be written with 0.

44.5.15 eGTM AEI bridge mode register (i=0)

***Note**: All writable bits are write-protected by bit CLSi_ARCH_RST.BRIDGE_MODE_WRDIS.*

CLSi_AEI_BRIDGE_MODE (i=0) Offset address: 000040_H+i*20000_H

eGTM AEI bridge mode register Reset values see: Table 2306

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BUFF_DPT								0							BRG_RST
r								r							rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			SYNC_INPUT_REG	0		BUFF_OVL	MODE_UP_PGR	0					BYPASS_SYNC	MSK_WR_RSP	BRG_MODE
r			r	r		rw1ch	rh	r					rw	rw	rw

Field	Bits	Type	Description
BRG_MODE	0	rw	**Defines the operation mode for the AEI bridge** 0_B AEI bridge operates in sync_bridge mode 1_B AEI bridge operates in async_bridge mode
MSK_WR_RSP	1	rw	**Mask write response** ***Note**:* *The status of the executed write-accesses can be observed by using the notify bits CLSi_ARCH_IRQ_NOTIFY.AEI_USP_ADDR, CLSi_ARCH_IRQ_NOTIFY.AEI_IM_ADDR, CLSi_ARCH_IRQ_NOTIFY.AEI_USP_BE.* ***Note**:* *With active write buffer CLSi_AEI_BRIDGE_MODE.MSK_WR_RSP=1, execution of actions can be delayed due to previous inserted write actions in the transaction buffer which wait to be serviced. This can lead to the fact that an access on the bus to a different peripheral than the **eGTM** might be executed earlier in time than the write-access buffered in the **eGTM**. Applications must be setup up with this in mind otherwise unexpected operation can happen.* 0_B Do not mask the write response. Depending on the selected address the latency for execution can vary due to eGTM internal arbitration. After this time the status of the access will be signaled by the signal AEI_STATUS to the bus interface. 1_B Mask write response. The write buffer of the bridge is activated, the actual access will be stored to the write buffer and without latency on the bus interface the acceptance of the access is signaled. AEI_STATUS=00_B will be signaled. In case of a full write buffer the actual access will be postponed until the next write buffer entry becomes free.
BYPASS_SYNC	2	rw	**Bypass synchronizer storage elements** **If CLSi_AEI_BRIDGE_MODE .BRG_MODE == 1:** 0_B Synchronizer storage elements in use, latency increase due to synchronization (aei_clk to CLKand back CLKto aei_clk). This setting must be used if aei_clk and CLK operate fully asynchronous by independent clock sources. 1_B Synchronizer storage elements are bypassed. No additional latency due to synchronization. This setting can be used if aei_clk and CLK are generated by clock gating or clock division out of a common clock source. Clock edges on aei_clk and CLK generated out of the same clock edge of the common clock source must have zero skew. **If CLSi_AEI_BRIDGE_MODE .BRG_MODE == 0:** 0_B In synchronous mode (CLSi_AEI_BRIDGE_MODE.BRG_MODE = 0), this bit field is not relevant and has no impact 1_B In synchronous mode (CLSi_AEI_BRIDGE_MODE.BRG_MODE = 0), this bit field is not relevant and has no impact

(table continues...)

(continued)

Field	Bits	Type	Description
MODE_UP_PGR	8	rh	**Mode update in progress.** 0_B No update in progress 1_B Update in progress
BUFF_OVL	9	rw1ch	**Buffer overflow register** ***Note**:* *A buffer overflow can occur while multiple aborts are issued by the external bus or a pipelined instruction is started while FBC = 0 (see CLSi_AEI_BRIDGE_PTR1 register).* **If reading:** 0_B No buffer overflow occurred 1_B Buffer overflow occurred **If writing:** 0_B No action 1_B Clear buffer overflow status
SYNC_INPUT_REG	12	r	**Additional pipelined stage in synchronous bridge mode** ***Note**:* *Reset value depends on the hardware configuration chosen by silicon vendor.* 0_B No additional pipelined stage implemented 1_B Additional pipelined stage implemented. All accesses in synchronous mode will be increased by one clock-cycle.
BRG_RST	16	rw	**Bridge software reset** ***Note**:* *This bit is cleared while bridge reset is executed.* **If writing:** 0_B No bridge reset request 1_B Bridge reset request **If reading:** 0_B No status
BUFF_DPT	31:24	r	**Buffer depth of AEI bridge** Signals the buffer depth of the **eGTM AEI** bridge implementation. ***Note**:* *Reset value depends on the hardware configuration chosen by silicon vendor.*
0	7:3, 11:10, 15:13, 23:17	r	**Reserved** Read as 0; must be written with 0.

Table 2306 Reset values of CLSi_AEI_BRIDGE_MODE (i=0)

Reset	Reset value	Note
Kernel Reset	$0400\ 0001_H$	
Bridge reset	$0000\ 0100\ 0000\ 0000\ 0000\ 000–\ 0000\ 0---_B$	

44.5.16 eGTM AEI bridge pointer 1 register (i=0)

***Note**: This register operates on the AEI_CLK domain.*

***Note**: This register holds diagnosis information about the **AEI** bus bridge. Each access to the **eGTM** will update the defined pointer bit fields. Depending on which values of CLSi_AEI_BRIDGE_MODE.BRG_MODE, CLSi_AEI_BRIDGE_MODE.MSK_WR_RSP, the **AEI** protocol, and operating frequency are used, the 4 pointer bit fields change at different clock-cycles relative to the start of the transaction. As a result of this, reading the register can show values not equal to the defined initial value, even directly after a write to CLSi_AEI_BRIDGE_MODE.BRG_RST=1.*

CLSi_AEI_BRIDGE_PTR1 (i=0) Offset address: $000044_H+i*20000_H$

eGTM AEI bridge pointer 1 register Reset values see: Table 2307

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSP_TRAN_RDY						FBC						ABT_TRAN_PGR			
rh						rh						rh			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ABT_TRAN_PGR	TRAN_IN_PGR					FIRST_RSP_PTR					NEW_TRAN_PTR				
rh	rh					rh					rh				

Field	Bits	Type	Description
NEW_TRAN_PTR	4:0	rh	**New transaction pointer** Signals the actual value of the new transaction pointer.
FIRST_RSP_PTR	9:5	rh	**First response pointer** Signals the actual value of first response pointer.
TRAN_IN_PGR	14:10	rh	**Transaction in progress pointer (acquire)** Transaction in progress pointer.
ABT_TRAN_PGR	19:15	rh	**Aborted transaction in progress pointer** Aborted transaction in progress pointer.
FBC	25:20	rh	**Free buffer count** Number of free buffer entries. ***Note**: Initial value depends on the hardware configuration chosen by silicon vendor. (see bit field CLSi_AEI_BRIDGE_MODE.BUFF_DPT).*
RSP_TRAN_RDY	31:26	rh	**Response transactions ready** Number of ready response transactions.

Table 2307 Reset values of CLSi_AEI_BRIDGE_PTR1 (i=0)

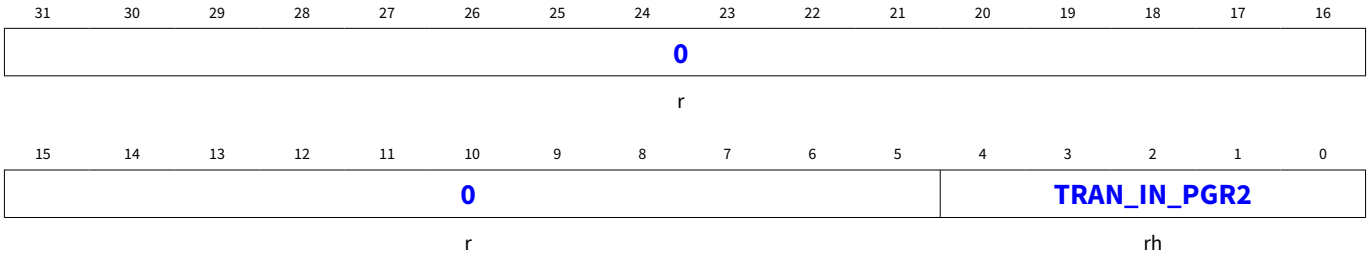
Reset	Reset value	Note
Kernel Reset	$0040\ 0000_H$	
Bridge reset	$0040\ 0000_H$	

44.5.17 eGTM AEI bridge pointer 2 register (i=0)

***Note**: This register operates on the GTM_CLK domain.*

CLSi_AEI_BRIDGE_PTR2 (i=0) Offset address: $000048_H+i*20000_H$

eGTM AEI bridge pointer 2 register Reset values see: Table 2308



Field	Bits	Type	Description
TRAN_IN_PGR2	4:0	rh	**Transaction in progress pointer (aquire2)** Transaction in progress pointer 2.
0	31:5	r	**Reserved** Read as 0.

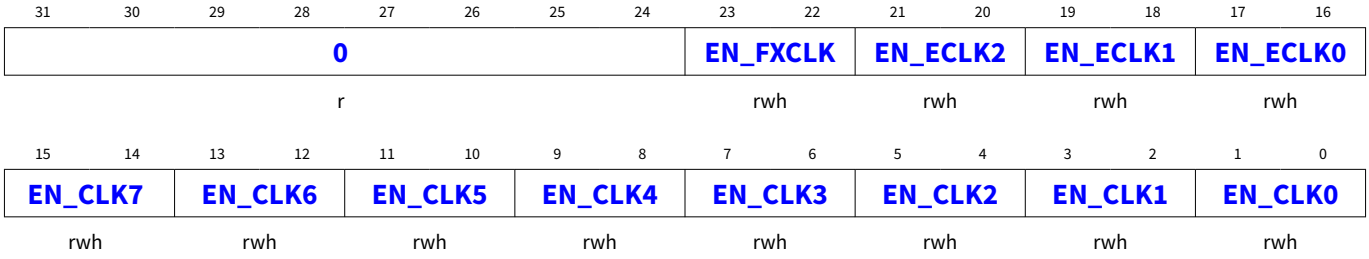
Table 2308 Reset values of CLSi_AEI_BRIDGE_PTR2 (i=0)

Reset	Reset value	Note
Kernel Reset	$0000\ 0000_H$	
Bridge reset	$0000\ 0000_H$	

44.5.18 CMU clock enable (i=0)

CLSi_CMU_CLK_EN (i=0) Offset address: $000080_H+i*20000_H$

CMU clock enable Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
EN_CLKx (x=0-7)	2*x+1:2*x	rwh	**Enable clock resolution** CMU_CLK_RES***[x:x]*** **If writing:** 00_B Don't care, bits will not be changed 01_B Disable clock signal and reinitialize internal states 10_B Enable clock signal 11_B Don't care, bits will not be changed **If reading:** 00_B Clock is disabled 01_B Unused 10_B Unused 11_B Clock is enabled
EN_ECLKz (z=0-2)	2*z+17:2*z +16	rwh	**Enable clock** CMU_ECLK***[z:z]*** **If writing:** 00_B Don't care, bits will not be changed 01_B Disable clock signal and reinitialize internal states 10_B Enable clock signal 11_B Don't care, bits will not be changed **If reading:** 00_B Clock is disabled 01_B Unused 10_B Unused 11_B Clock is enabled
EN_FXCLK	23:22	rwh	**Enable clock resolution** CMU_FXCLK_RES **If writing:** 00_B Don't care, bits will not be changed 01_B Disable clock signal and reinitialize internal states 10_B Enable clock signal 11_B Don't care, bits will not be changed **If reading:** 00_B Clock is disabled 01_B Unused 10_B Unused 11_B Clock is enabled
0	31:24	r	**Reserved** Read as 0; must be written with 0.

44.5.19 The numerator for CMU global clock resolution generator (i=0)

CLSi_CMU_GCLK_NUM (i=0)

The numerator for CMU global clock resolution generator

Offset address: 000084_H+i*20000_H

Kernel Reset value: $0000\ 0001_H$

Field	Bits	Type	Description
GCLK_NUM	23:0	rwh	**Numerator for global clock resolution generator. Defines numerator of the fractional divider.** ***Note**: Value can only be modified when all clock enables CLSi_CMU_CLK_EN.EN_CLKx and the CLSi_CMU_CLK_EN.EN_FXCLK are disabled. Protected by (CLSi_CMU_CLK_EN.EN_CLKx != 0 or CLSi_CMU_CLK_EN.EN_FXCLK != 0)* ***Note**: The **CMU** hardware alters the content of CLSi_CMU_GCLK_NUM.GCLK_NUM and CLSi_CMU_GCLK_DEN.GCLK_DEN automatically to 1_H, if CLSi_CMU_GCLK_NUM.GCLK_NUM is specified less than CLSi_CMU_GCLK_DEN.GCLK_DEN or one of the values is specified with a value zero. Thus, a secure way for altering the values is writing twice to the register CLSi_CMU_GCLK_NUM followed by a single write to register CLSi_CMU_GCLK_DEN.*
0	31:24	r	**Reserved** Read as 0; must be written with 0.

44.5.20 The denominator for CMU global clock resolution generator (i=0)

CLSi_CMU_GCLK_DEN (i=0)

Offset address: $000088_H+i*20000_H$

The denominator for CMU global clock resolution generator

Kernel Reset value: $0000\ 0001_H$

Field	Bits	Type	Description
GCLK_DEN	23:0	rwh	**Denominator for global resolution generator. Defines denominator of the fractional divider.** ***Note**: Value can only be modified when all clock enables CLSi_CMU_CLK_EN.EN_CLKx and the CLSi_CMU_CLK_EN.EN_FXCLK are disabled. Protected by (CLSi_CMU_CLK_EN.EN_CLKx != 0 or CLSi_CMU_CLK_EN.EN_FXCLK != 0)* ***Note**: The **CMU** hardware alters the content of CLSi_CMU_GCLK_NUM.GCLK_NUM and CLSi_CMU_GCLK_DEN.GCLK_DEN automatically to 1_H, if CLSi_CMU_GCLK_NUM.GCLK_NUM is specified less than CLSi_CMU_GCLK_DEN.GCLK_DEN or one of the values is specified with a value zero. Thus, a secure way for altering the values is writing twice to the register CLSi_CMU_GCLK_NUM followed by a single write to register CLSi_CMU_GCLK_DEN.*
0	31:24	r	**Reserved** Read as 0; must be written with 0.

44.5.21 CMU control for clock resolution generator [x] (x=0-5;i=0)

CLSi_CMU_CLKx_CTRL (i=0;x=0-5) Offset address: $00008C_H+i*20000_H+x*4$

CMU control for clock resolution generator [x] Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								CLK_CNT							
r								rwh							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CLK_CNT															
rwh															

Field	Bits	Type	Description
CLK_CNT	23:0	rwh	**Clock count. Defines count value for the clock divider.** ***Note**: Value can only be modified when clock enable CLSi_CMU_CLK_EN.EN_CLKx (x∈{0, 1,..., 5}) and CLSi_CMU_CLK_EN.EN_ECLKz **(z=1)** are disabled. Protected by (CLSi_CMU_CLK_EN.EN_CLKx != 0 or CLSi_CMU_CLK_EN.EN_ECLKz **(z=1)** != 0)*
0	31:24	r	**Reserved** Read as 0; must be written with 0.

44.5.22 CMU control for clock resolution generator 6 (i=0)

CLSi_CMU_CLK_6_CTRL (i=0) Offset address: $0000A4_H$+i*20000_H

CMU control for clock resolution generator 6 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0						CLK_SEL		CLK_CNT							
r						rwh		rwh							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CLK_CNT															
rwh															

Field	Bits	Type	Description
CLK_CNT	23:0	rwh	**Clock count. Define count value for the clock resolution generator responsible for** CMU_CLK_RES***[6:6]*** **generation.** ***Note**: Value can only be modified when clock enable* CLSi_CMU_CLK_EN.EN_CLKx ***(x=6)*** *and* CLSi_CMU_CLK_EN.EN_ECLKz ***(z=1)*** *are disabled. Protected by (*CLSi_CMU_CLK_EN.EN_CLKx ***(x=6)*** *!= 0 or* CLSi_CMU_CLK_EN.EN_ECLKz ***(z=1)*** *!= 0)*
CLK_SEL	25:24	rwh	**Source selection** ***Note**: Value can only be modified when clock enable* CLSi_CMU_CLK_EN.EN_CLKx ***(x=6)*** *and* CLSi_CMU_CLK_EN.EN_ECLKz ***(z=1)*** *are disabled. Protected by (*CLSi_CMU_CLK_EN.EN_CLKx ***(x=6)*** *!= 0 or* CLSi_CMU_CLK_EN.EN_ECLKz ***(z=1)*** *!= 0)* 00_B Use output of clock resolution generator 6 01_B input tied to low, no DPLL present in device 10_B input tied to low, no DPLL present in device 11_B Use signal CCM[i]_CLK_RES[7:7] of sub-module CCM0
0	31:26	r	**Reserved** Read as 0; must be written with 0.

44.5.23 CMU control for clock resolution generator 7 (i=0)

CLSi_CMU_CLK_7_CTRL (i=0) Offset address: $0000A8_H$+i*20000_H

CMU control for clock resolution generator 7 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0						CLK_SEL		CLK_CNT							
r						rwh		rwh							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CLK_CNT															
rwh															

Field	Bits	Type	Description
CLK_CNT	23:0	rwh	**Clock count. Define count value for the clock resolution generator responsible for** CMU_CLK_RES***[7:7]*** **generation.** ***Note**: Value can only be modified when clock enable* CLSi_CMU_CLK_EN.EN_CLKx ***(x=7)*** *and* CLSi_CMU_CLK_EN.EN_ECLKz ***(z=1)*** *are disabled. Protected by (*CLSi_CMU_CLK_EN.EN_CLKx ***(x=7)*** *!= 0 or* CLSi_CMU_CLK_EN.EN_ECLKz ***(z=1)*** *!= 0)*
CLK_SEL	25:24	rwh	**Source selection** ***Note**: Value can only be modified when clock enable* CLSi_CMU_CLK_EN.EN_CLKx ***(x=7)*** *and* CLSi_CMU_CLK_EN.EN_ECLKz ***(z=1)*** *are disabled. Protected by (*CLSi_CMU_CLK_EN.EN_CLKx ***(x=7)*** *!= 0 or* CLSi_CMU_CLK_EN.EN_ECLKz ***(z=1)*** *!= 0)* 00_B Use output of clock resolution generator 7 01_B input tied to low, no DPLL present in device 10_B input tied to low, no DPLL present in device 11_B Reserved, input tied to low
0	31:26	r	**Reserved** Read as 0; must be written with 0.

44.5.24 The numerator for the external clock resolution generator [z] (z=0-2;i=0)

CLSi_CMU_ECLKz_NUM (i=0;z=0-2)

Offset address: $0000AC_H$+i*20000_H+z*8

The numerator for the external clock resolution generator [z]

Kernel Reset value: $0000\ 0001_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								ECLK_NUM							
r								rwh							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECLK_NUM															
rwh															

Field	Bits	Type	Description
ECLK_NUM	23:0	rwh	**The numerator for external clock resolution generator. Defines numerator of the fractional divider.** ***Note**: Value can only be modified when clock enable CLSi_CMU_CLK_EN.EN_ECLKz is disabled. Protected by (CLSi_CMU_CLK_EN.EN_ECLKz != 0)* ***Note**: The **CMU** hardware alters the content of CLSi_CMU_ECLKz_NUM and CLSi_CMU_ECLKz_DEN automatically to 1_H, if CLSi_CMU_ECLKz_NUM is specified less than CLSi_CMU_ECLKz_DEN or one of the values is specified with a value zero. Thus, a secure way for altering the values is writing twice to the register CLSi_CMU_ECLKz_NUM followed by a single write to register CLSi_CMU_ECLKz_DEN.*
0	31:24	r	**Reserved** Read as 0; must be written with 0.

44.5.25 The denominator for the external clock resolution generator [z] (z=0-2;i=0)

CLSi_CMU_ECLKz_DEN (i=0;z=0-2) Offset address: $0000B0_H$+i*20000_H+z*8

The denominator for the external clock resolution generator [z] Kernel Reset value: $0000\ 0001_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								ECLK_DEN							
r								rwh							

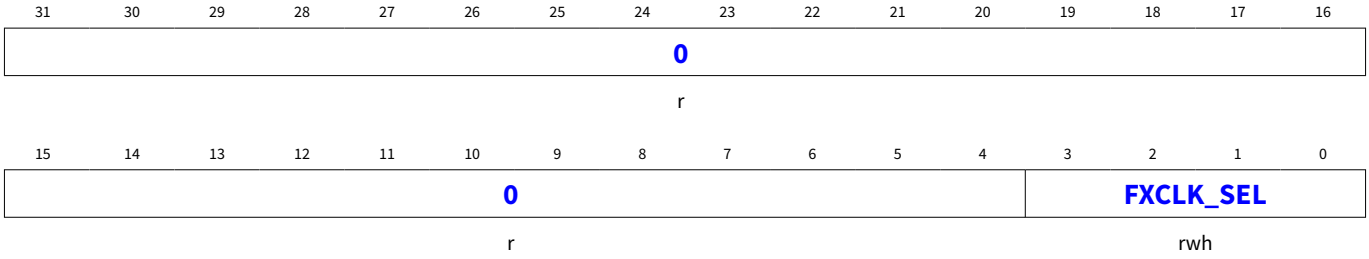
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECLK_DEN															
rwh															

Field	Bits	Type	Description
ECLK_DEN	23:0	rwh	**The denominator for external clock resolution generator. Defines denominator of the fractional divider.** ***Note**: Value can only be modified when clock enable CLSi_CMU_CLK_EN.EN_ECLKz is disabled. Protected by (CLSi_CMU_CLK_EN.EN_ECLKz != 0)* ***Note**: The **CMU** hardware alters the content of CLSi_CMU_ECLKz_NUM and CLSi_CMU_ECLKz_DEN automatically to 1_H, if CLSi_CMU_ECLKz_NUM is specified less than CLSi_CMU_ECLKz_DEN or one of the values is specified with a value zero. Thus, a secure way for altering the values is writing twice to the register CLSi_CMU_ECLKz_NUM followed by a single write to register CLSi_CMU_ECLKz_DEN.*
0	31:24	r	**Reserved** Read as 0; must be written with 0.

44.5.26 CMU control for selection of FCR subblock input (i=0)

CLSi_CMU_FXCLK_CTRL (i=0) Offset address: $0000C4_H+i*20000_H$

CMU control for selection of FCR subblock input Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
FXCLK_SEL	3:0	rwh	**Input selection for** EN_FXCLK **line** ***Note**: This value can only be written, when the CMU_FXCLK_RES generation is disabled. See bits [23:22] in register CLSi_CMU_CLK_EN. Protected by (CLSi_CMU_CLK_EN.EN_FXCLK != 0)* ***Note**: Other values for CLSi_CMU_FXCLK_CTRL.FXCLK_SEL are reserved and should not be used, but they behave like CLSi_CMU_FXCLK_CTRL.FXCLK_SEL = 0.* 0_H CMU_GCLK_EN selected 1_H CMU_CLK_RES[0:0] selected 2_H CMU_CLK_RES[1:1] selected 3_H CMU_CLK_RES[2:2] selected 4_H CMU_CLK_RES[3:3] selected 5_H CMU_CLK_RES[4:4] selected 6_H CMU_CLK_RES[5:5] selected 7_H CMU_CLK_RES[6:6] selected 8_H CMU_CLK_RES[7:7] selected **others,** Reserved
0	31:4	r	**Reserved** Read as 0; must be written with 0.

44.5.27 CMU control for clock resolution generator (i=0)

CLSi_CMU_CLK_CTRL (i=0) Offset address: $0000CC_H+i*20000_H$

CMU control for clock resolution generator Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0							CLK8_EXT_DIVIDER	CLK7_EXT_DIVIDER	CLK6_EXT_DIVIDER	CLK5_EXT_DIVIDER	CLK4_EXT_DIVIDER	CLK3_EXT_DIVIDER	CLK2_EXT_DIVIDER	CLK1_EXT_DIVIDER	CLK0_EXT_DIVIDER
r							rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
CLKx_EXT_DIVIDER (x=0-7)	x	rwh	**Input selection for Clock Resolution Generator [x]** ***Note**: Value can only be modified when clock enable CLSi_CMU_CLK_EN.EN_CLKx and CLSi_CMU_CLK_EN.EN_ECLKz **(z=1)** are disabled. Protected by (CLSi_CMU_CLK_EN.EN_CLKx != 0 or CLSi_CMU_CLK_EN.EN_ECLKz **(z=1)** != 0)* 0_B Use Clock resolution CMU_GCLK_EN 1_B Use Clock resolution CMU_ECLK1_EN, which corresponds to the clock CMU_ECLK[1:1]
CLK8_EXT_DIVIDER	8	rwh	**Source selection for CMU_CLK_RES*[8:8]*** ***Note**: Value can only be modified when CLSi_CMU_CLK_EN.EN_ECLKz **(z=0)** is disabled. Protected by (CLSi_CMU_CLK_EN.EN_ECLKz **(z=0)** != 0)* 0_B Use Clock resolution '1', which corresponds to the cluster clock CLS[j]_CLK operating frequency CLS[j]_CLK 1_B Use Clock resolution CMU_ECLK0_EN, which corresponds to the clock CMU_ECLK[0:0]
0	31:9	r	**Reserved** Read as 0; must be written with 0.

44.5.28 TBU global channel enable (i=0)

***Note**: These bits are only applicable if channel is implemented for this device, otherwise read and write as zero*

CLSi_TBU_CHEN (i=0) Offset address: 000100_H+i*20000_H

TBU global channel enable Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								0		ENDIS_CH2		ENDIS_CH1		ENDIS_CH0	
r								r		rwh		rwh		rwh	

Field	Bits	Type	Description
ENDIS_CHy (y=0-2)	2*y+1:2*y	rwh	**TBU channel [y] enable/disable control** **If reading:** 00_B Channel is disabled 01_B Unused 10_B Unused 11_B Channel is enabled **If writing:** 00_B Don't care, bits will not be changed 01_B Disable channel 10_B Enable channel 11_B Don't care, bits will not be changed
0	7:6, 31:8	r	**Reserved** Read as 0; must be written with 0.

44.5.29 TBU channel 0 control (i=0)

CLSi_TBU_CH0_CTRL (i=0) Offset address: $000104_H+i*20000_H$

TBU channel 0 control Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												CH_CLK_SRC			LOW_RES
r												rwh			rwh

Field	Bits	Type	Description
LOW_RES	0	rwh	CLSi_TBU_CH0_BASE **register resolution** ***Note**: This value can only be modified if channel 0 is disabled.* 0_B TBU channel uses lower counter bits [23:0] 1_B TBU channel uses upper counter bits [26:3]

(table continues...)

(continued)

Field	Bits	Type	Description
CH_CLK_SRC	3:1	rwh	**Clock source for channel 0, channel 1 and channel 2 time base counter** ***Note**: This value can only be modified if channel 0 is disabled.* 000_B CCM[i]_CLK_RES[0:0] selected 001_B CCM[i]_CLK_RES[1:1] selected 010_B CCM[i]_CLK_RES[2:2] selected 011_B CCM[i]_CLK_RES[3:3] selected 100_B CCM[i]_CLK_RES[4:4] selected 101_B CCM[i]_CLK_RES[5:5] selected 110_B CCM[i]_CLK_RES[6:6] selected 111_B CCM[i]_CLK_RES[7:7] selected
0	31:4	r	**Reserved** Read as 0; must be written with 0.

44.5.30 TBU channel 0 base (i=0)

CLSi_TBU_CH0_BASE (i=0) Offset address: 000108_H+i*20000_H

TBU channel 0 base Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0					BASE										
r					rwh										

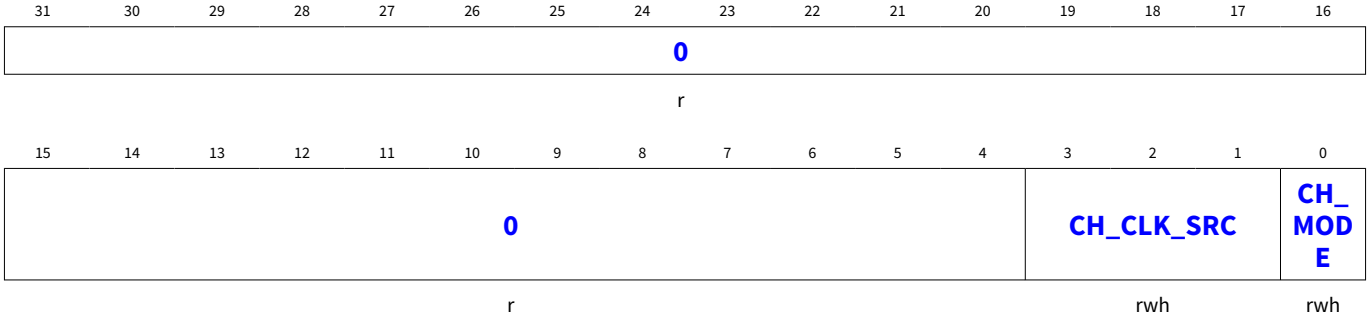
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BASE															
rwh															

Field	Bits	Type	Description
BASE	26:0	rwh	**Time base value for channel 0** ***Note**: The value of CLSi_TBU_CH0_BASE.BASE can only be written if the **TBU** channel 0 is disabled* ***Note**: If channel 0 is enabled, a read-access to this register provides the current value of the underlying 27-bit counter.*
0	31:27	r	**Reserved** Read as 0; must be written with 0.

44.5.31 TBU channel 1 control (i=0)

CLSi_TBU_CH1_CTRL (i=0) Offset address: $00010C_H$+i*20000_H

TBU channel 1 control Kernel Reset value: $0000\ 0000_H$

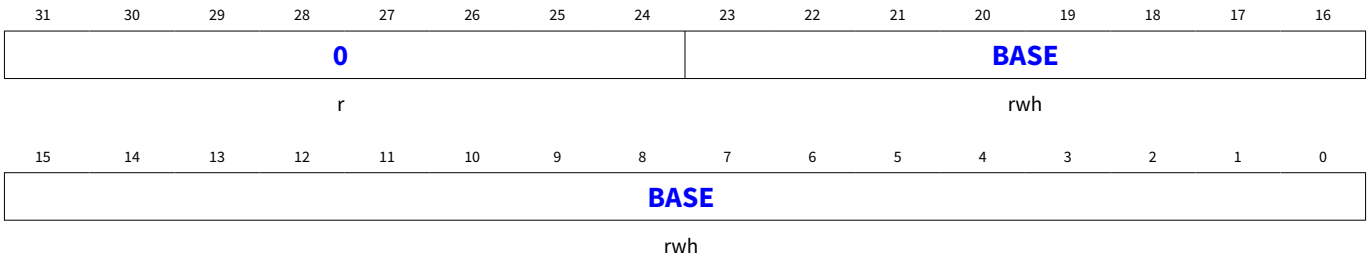


Field	Bits	Type	Description
CH_MODE	0	rwh	**Channel mode** ***Note**: This value can only be modified if channel 1 is disabled. In free running counter mode the **CMU** clock source specified by CLSi_TBU_CH1_CTRL.CH_CLK_SRC is used for the counter.* 0_B Free running counter mode 1_B Forward/backward counter mode
CH_CLK_SRC	3:1	rwh	**Clock source for channel 1 time base counter** ***Note**: This value can only be modified if channel 1 is disabled* 000_B CCM[i]_CLK_RES[0:0] selected 001_B CCM[i]_CLK_RES[1:1] selected 010_B CCM[i]_CLK_RES[2:2] selected 011_B CCM[i]_CLK_RES[3:3] selected 100_B CCM[i]_CLK_RES[4:4] selected 101_B CCM[i]_CLK_RES[5:5] selected 110_B CCM[i]_CLK_RES[6:6] selected 111_B CCM[i]_CLK_RES[7:7] selected
0	31:4	r	**Reserved** Read as 0; must be written with 0.

44.5.32 TBU channel [x] base (x=1-2;i=0)

CLSi_TBU_CHx_BASE (i=0;x=1-2) Offset address: 000110_H+i*20000_H+(x-1)*8

TBU channel [x] base Kernel Reset value: $0000\ 0000_H$

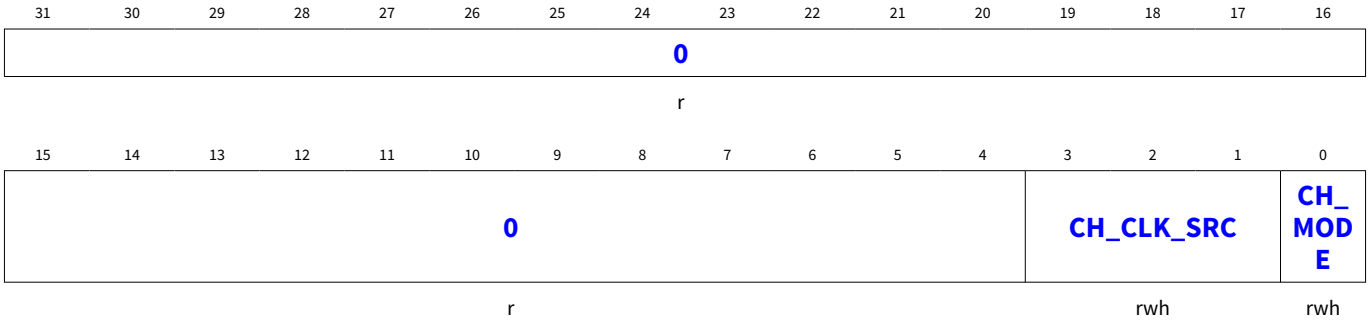


Field	Bits	Type	Description
BASE	23:0	rwh	**Time base value for channel [x]** ***Note**: The value of CLSi_TBU_CHx_BASE.BASE can be written only if the corresponding **TBU** channel y (y∈{1, 2}) is disabled* ***Note**: If the corresponding channel y (y∈{1, 2}) is enabled, a read-access to this register provides the current value of the underlying counter.*
0	31:24	r	**Reserved** Read as 0; must be written with 0.

44.5.33 TBU channel 2 control (i=0)

CLSi_TBU_CH2_CTRL (i=0) Offset address: $000114_H+i*20000_H$

TBU channel 2 control Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
CH_MODE	0	rwh	**Channel mode** ***Note**: This value can be modified only if channel 2 is disabled. In free running counter mode the **CMU** clock source specified by CLSi_TBU_CH2_CTRL.CH_CLK_SRC is used for the counter.* 0_B Free running counter mode 1_B Forward/backward counter mode
CH_CLK_SRC	3:1	rwh	**Clock source for channel 2 time base counter** ***Note**: This value can only be modified if channel 2 is disabled* 000_B CCM[i]_CLK_RES[0:0] selected 001_B CCM[i]_CLK_RES[1:1] selected 010_B CCM[i]_CLK_RES[2:2] selected 011_B CCM[i]_CLK_RES[3:3] selected 100_B CCM[i]_CLK_RES[4:4] selected 101_B CCM[i]_CLK_RES[5:5] selected 110_B CCM[i]_CLK_RES[6:6] selected 111_B CCM[i]_CLK_RES[7:7] selected

(table continues...)

(continued)

Field	Bits	Type	Description
0	31:4	r	**Reserved** Read as 0; must be written with 0.

44.5.34 ICM_IRQG_0 (i=0)

ICM Interrupt group register covering infrastructural and safety components (**AEI**, **CMP**, **SPE**)

CLSi_ICM_IRQG_0 (i=0) Offset address: $000400_H+i*20000_H$

ICM_IRQG_0 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		0	0	0	0	0	SPE2_IRQ	SPE1_IRQ	SPE0_IRQ	CMP_IRQ	AEI_IRQ	0	0	0	0
r		r	r	r	r	r	rh	rh	rh	rh	rh	r	r	r	r

Field	Bits	Type	Description
AEI_IRQ	4	rh	CLSi_ICM_IRQG_0.AEI_IRQ**: AEI_IRQ interrupt** ***Note**: This bit is only set, when the interrupt is enabled in the interrupt enable register of the corresponding sub-module.* ***Note**: Set this bit represents an **OR** function of the interrupt sources AEI_TO_XPT_IRQ, AEI_USP_ADDR_IRQ, AEI_IM_ADDR_IRQ, AEI_USP_BE_IRQ, CLK_EN_ERR_IRQ or CLK_PER_ERR_IRQ.* 0_B no interrupt occurred 1_B interrupt was raised by the corresponding sub-module
CMP_IRQ	5	rh	**CMP shared sub-module interrupt** ***Note**: This bit is only set, when the interrupt is enabled in the interrupt enable register of the corresponding sub-module.* 0_B no interrupt occurred 1_B interrupt was raised by the corresponding sub-module
SPEj_IRQ (j=0-2)	j+6	rh	**SPE[j] shared sub-module interrupt** ***Note**: This bit is only set, when the interrupt is enabled in the interrupt enable register of the corresponding sub-module.* ***Note**: Set this bit represents an **OR** function of the five interrupt sources SPE[i]_NIPD, SPE[i]_DCHG, SPE[i]_PERR, SPE[i]_BIS or SPE[i]_RCMP .* 0_B no interrupt occurred 1_B interrupt was raised by the corresponding sub-module

(table continues...)

(continued)

Field	Bits	Type	Description
0	0, 1, 2, 3, 13, 12, 11, 10, 9, 15:14, 23, 22, 21, 20, 19, 18, 17, 16, 31, 30, 29, 28, 27, 26, 25, 24	r	**Reserved** Read as 0.

44.5.35 ICM Interrupt group register covering TIM0, TIM1, TIM2, TIM3 (i=0)

CLSi_ICM_IRQG_2 (i=0) Offset address: 000408_H+i*20000_H

ICM Interrupt group register covering TIM0, TIM1, TIM2, TIM3 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	TIM2_CH7_IRQ	TIM2_CH6_IRQ	TIM2_CH5_IRQ	TIM2_CH4_IRQ	TIM2_CH3_IRQ	TIM2_CH2_IRQ	TIM2_CH1_IRQ	TIM2_CH0_IRQ
r	r	r	r	r	r	r	r	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TIM1_CH7_IRQ	TIM1_CH6_IRQ	TIM1_CH5_IRQ	TIM1_CH4_IRQ	TIM1_CH3_IRQ	TIM1_CH2_IRQ	TIM1_CH1_IRQ	TIM1_CH0_IRQ	TIM0_CH7_IRQ	TIM0_CH6_IRQ	TIM0_CH5_IRQ	TIM0_CH4_IRQ	TIM0_CH3_IRQ	TIM0_CH2_IRQ	TIM0_CH1_IRQ	TIM0_CH0_IRQ
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
TIM0_CHx_IRQ (x=0-7)	x	rh	**TIM0 shared interrupt channel [x]** ***Note**:* *This bit is only set, when the interrupt is enabled in the interrupt enable register of the corresponding sub-module.* ***Note**:* *Set this bit represents an **OR** function of the interrupt sources TIM_NEWVAL_IRQ[x:x], TIM_ECNTOFL_IRQ[x:x], TIM_CNTOFL_IRQ[x:x], TIM_GPROFL_IRQ[x:x], TIM_GLITCHDET_IRQ[x:x] of **TIM** instance 0, channel [x].* 0_B no interrupt occurred 1_B interrupt was raised by the corresponding sub-module

(table continues...)

(continued)

Field	Bits	Type	Description
TIM1_CHx_IRQ (x=0-7)	x+8	rh	**TIM1 shared interrupt channel [x]** ***Note**: This bit is only set, when the interrupt is enabled in the interrupt enable register of the corresponding sub-module.* ***Note**: Set this bit represents an **OR** function of the interrupt sources TIM_NEWVAL_IRQ[x:x], TIM_ECNTOFL_IRQ[x:x], TIM_CNTOFL_IRQ[x:x], TIM_GPROFL_IRQ[x:x], TIM_GLITCHDET_IRQ[x:x] of **TIM** instance 1, channel [x].* 0_B no interrupt occurred 1_B interrupt was raised by the corresponding sub-module
TIM2_CHx_IRQ (x=0-7)	x+16	rh	**TIM2 shared interrupt channel [x]** ***Note**: This bit is only set, when the interrupt is enabled in the interrupt enable register of the corresponding sub-module.* ***Note**: Set this bit represents an **OR** function of the interrupt sources TIM_NEWVAL_IRQ[x:x], TIM_ECNTOFL_IRQ[x:x], TIM_CNTOFL_IRQ[x:x], TIM_GPROFL_IRQ[x:x], TIM_GLITCHDET_IRQ[x:x] of **TIM** instance 2, channel [x].* 0_B no interrupt occurred 1_B interrupt was raised by the corresponding sub-module
0	31, 30, 29, 28, 27, 26, 25, 24	r	**Reserved** Read as 0.

44.5.36 ICM Interrupt group register for module error interrupt information (i=0)

CLSi_ICM_IRQG_MEI (i=0)

Offset address: $000430_H+i*20000_H$

ICM Interrupt group register for module error interrupt information

Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0						0	CMP_EIRQ	0	SPE2_EIRQ	SPE1_EIRQ	SPE0_EIRQ	0	0	0	0
r						r	rh	r	rh	rh	rh	r	r	r	r

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	TIM2_EIRQ	TIM1_EIRQ	TIM0_EIRQ	0	0	0	GTM_EIRQ
r	r	r	r	r	r	r	r	r	rh	rh	rh	r	r	r	rh

Field	Bits	Type	Description
GTM_EIRQ	0	rh	**AEI error interrupt request** ***Note**:* *This bit is only set, when the error interrupt is enabled in the error interrupt enable register of the corresponding sub-module.* ***Note**:* *Set this bit represents an **OR** function of the interrupt sources AEI_TO_XPT_EIRQ, AEI_USP_ADDR_EIRQ, AEI_IM_ADDR_EIRQ, AEI_USP_BE_EIRQ, CLK_EN_ERR_IRQ or CLK_PER_ERR_IRQ.* 0_B no interrupt occurred 1_B interrupt was raised by the corresponding sub-module
TIMj_EIRQ (j=0-2)	j+4	rh	**TIM[j] error interrupt** ***Note**:* *This bit is only set, when the error interrupt is enabled in the error interrupt enable register of the corresponding sub-module.* 0_B no interrupt occurred 1_B interrupt was raised by the corresponding sub-module
SPEj_EIRQ (j=0-2)	j+20	rh	**SPE[j] error interrupt** ***Note**:* *This bit is only set, when the error interrupt is enabled in the error interrupt enable register of the corresponding sub-module.* 0_B no interrupt occurred 1_B interrupt was raised by the corresponding sub-module
CMP_EIRQ	24	rh	**CMP error interrupt** ***Note**:* *This bit is only set, when the error interrupt is enabled in the error interrupt enable register of the corresponding sub-module.* 0_B no interrupt occurred 1_B interrupt was raised by the corresponding sub-module
0	1, 3, 2, 11, 10, 9, 8, 7, 19, 18, 17, 16, 15, 14, 13, 12, 23, 25, 31:26	r	**Reserved** Read as 0.

44.5.37 ICM Interrupt group register 1 for channel error interrupt information (i=0)

CLSi_ICM_IRQG_CEI1 (i=0)

Offset address: $000438_H+i*20000_H$

ICM Interrupt group register 1 for channel error interrupt information

Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	TIM2_CH7_EIRQ	TIM2_CH6_EIRQ	TIM2_CH5_EIRQ	TIM2_CH4_EIRQ	TIM2_CH3_EIRQ	TIM2_CH2_EIRQ	TIM2_CH1_EIRQ	TIM2_CH0_EIRQ
r	r	r	r	r	r	r	r	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TIM1_CH7_EIRQ	TIM1_CH6_EIRQ	TIM1_CH5_EIRQ	TIM1_CH4_EIRQ	TIM1_CH3_EIRQ	TIM1_CH2_EIRQ	TIM1_CH1_EIRQ	TIM1_CH0_EIRQ	TIM0_CH7_EIRQ	TIM0_CH6_EIRQ	TIM0_CH5_EIRQ	TIM0_CH4_EIRQ	TIM0_CH3_EIRQ	TIM0_CH2_EIRQ	TIM0_CH1_EIRQ	TIM0_CH0_EIRQ
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
TIM0_CHx_EIRQ (x=0-7)	x	rh	**TIM0 channel [x] error interrupt** ***Note:*** *This bit is only set, when the error interrupt is enabled in the error interrupt enable register of the corresponding sub-module.* 0_B no error interrupt occurred 1_B error interrupt was raised by the corresponding sub-module
TIM1_CHx_EIRQ (x=0-7)	x+8	rh	**TIM1 channel [x] error interrupt** ***Note:*** *This bit is only set, when the error interrupt is enabled in the error interrupt enable register of the corresponding sub-module.* 0_B no error interrupt occurred 1_B error interrupt was raised by the corresponding sub-module
TIM2_CHx_EIRQ (x=0-7)	x+16	rh	**TIM2 channel [x] error interrupt** ***Note:*** *This bit is only set, when the error interrupt is enabled in the error interrupt enable register of the corresponding sub-module.* 0_B no error interrupt occurred 1_B error interrupt was raised by the corresponding sub-module
0	31, 30, 29, 28, 27, 26, 25, 24	r	**Reserved** Read as 0.

44.5.38 ICM Interrupt group SPE for module Error Interrupt information (i=0)

CLSi_ICM_IRQG_SPE_CEI (i=0) Offset address: $0004B4_H$+i*20000_H

ICM Interrupt group SPE for module Error Interrupt information Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								0	0	0	0	0	SPE2_EIRQ	SPE1_EIRQ	SPE0_EIRQ
r								r	r	r	r	r	rh	rh	rh

Field	Bits	Type	Description
SPEj_EIRQ (j=0-2)	j	rh	**SPE[j] error interrupt** ***Note**: This bit is only set, when the interrupt is enabled in the interrupt enable register of the corresponding sub-module.* 0_B no interrupt occurred 1_B interrupt was raised by the corresponding sub-module
0	7, 6, 5, 4, 3, 31:8	r	**Reserved** Read as 0.

44.5.39 ICM_IRQG_CLS_[g]_MEI (g=0;i=0)

ICM Interrupt group for module Error Interrupt information for each **TIM[j]**, **SPE[j]** (j=4*[g]+k, k∈{0, ... ,3})

CLSi_ICM_IRQG_CLSg_MEI (g=0;i=0) Offset address: 000510_H+i*20000_H

ICM_IRQG_CLS_[g]_MEI Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0				0	0	0	0	0				0	SPE_M2_EIRQ	0	TIM_M2_EIRQ
r				r	r	r	r	r				r	rh	r	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				0	SPE_M1_EIRQ	0	TIM_M1_EIRQ	0				0	SPE_M0_EIRQ	0	TIM_M0_EIRQ
r				r	rh	r	rh	r				r	rh	r	rh

Field	Bits	Type	Description
TIM_M0_EIRQ	0	rh	**Error interrupt TIM[j]_EIRQ (j=4*[g])** ***Note**:* *This bit is only set, when the error interrupt is enabled in the error interrupt enable register of the corresponding sub-module.* 0_B no interrupt occurred 1_B error interrupt was raised by the corresponding sub-module
SPE_M0_EIRQ	2	rh	**Error interrupt SPE[j]_EIRQ ([j]=4*[g]+0)** ***Note**:* *This bit is only set, when the error interrupt is enabled in the error interrupt enable register of the corresponding sub-module.* 0_B no interrupt occurred 1_B error interrupt was raised by the corresponding sub-module
TIM_M1_EIRQ	8	rh	**Error interrupt TIM[j]_EIRQ ([j]=4*[g]+1)** ***Note**:* *This bit is only set, when the error interrupt is enabled in the error interrupt enable register of the corresponding sub-module.* 0_B no interrupt occurred 1_B error interrupt was raised by the corresponding sub-module
SPE_M1_EIRQ	10	rh	**Error interrupt SPE[j]_EIRQ ([j]=4*[g]+1)** ***Note**:* *This bit is only set, when the error interrupt is enabled in the error interrupt enable register of the corresponding sub-module.* 0_B no interrupt occurred 1_B error interrupt was raised by the corresponding sub-module
TIM_M2_EIRQ	16	rh	**Error interrupt TIM[j]_EIRQ ([j]=4*[g]+2)** ***Note**:* *This bit is only set, when the error interrupt is enabled in the error interrupt enable register of the corresponding sub-module.* 0_B no interrupt occurred 1_B error interrupt was raised by the corresponding sub-module
SPE_M2_EIRQ	18	rh	**Error interrupt SPE[j]_EIRQ ([j]=4*[g]+2)** ***Note**:* *This bit is only set, when the error interrupt is enabled in the error interrupt enable register of the corresponding sub-module.* 0_B no interrupt occurred 1_B error interrupt was raised by the corresponding sub-module

(table continues...)

(continued)

Field	Bits	Type	Description
0	1, 3, 7:4, 9, 11, 15:12, 17, 19, 23:20, 24, 25, 26, 27, 31:28	r	**Reserved** Read as 0.

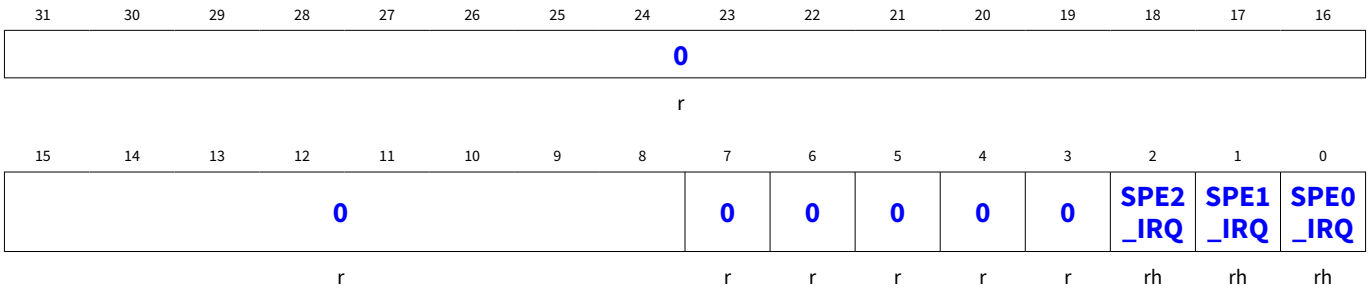
44.5.40 ICM Interrupt group SPE for module Interrupt information (i=0)

CLSi_ICM_IRQG_SPE_CI (i=0)

Offset address: $000570_H+i*20000_H$

ICM Interrupt group SPE for module Interrupt information

Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
SPEj_IRQ (j=0-2)	j	rh	**SPE[j] interrupt** ***Note**: This bit is only set, when the interrupt is enabled in the interrupt enable register of the corresponding sub-module.* 0_B no interrupt occurred 1_B interrupt was raised by the corresponding sub-module
0	7, 6, 5, 4, 3, 31:8	r	**Reserved** Read as 0.

44.5.41 ICM_IRQG_ATOM_[g]_CI (g=0;i=0)

ICM Interrupt group **ATOM** [g] for Channel Interrupt information of **ATOM[j]** (m=4*[g]+(0..3))

CLSi_ICM_IRQG_ATOMg_CI (g=0;i=0) Offset address: 000590_H+i*20000_H

ICM_IRQG_ATOM_[g]_CI Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	ATOM_M2_CH7_IRQ	ATOM_M2_CH6_IRQ	ATOM_M2_CH5_IRQ	ATOM_M2_CH4_IRQ	ATOM_M2_CH3_IRQ	ATOM_M2_CH2_IRQ	ATOM_M2_CH1_IRQ	ATOM_M2_CH0_IRQ
r	r	r	r	r	r	r	r	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ATOM_M1_CH7_IRQ	ATOM_M1_CH6_IRQ	ATOM_M1_CH5_IRQ	ATOM_M1_CH4_IRQ	ATOM_M1_CH3_IRQ	ATOM_M1_CH2_IRQ	ATOM_M1_CH1_IRQ	ATOM_M1_CH0_IRQ	ATOM_M0_CH7_IRQ	ATOM_M0_CH6_IRQ	ATOM_M0_CH5_IRQ	ATOM_M0_CH4_IRQ	ATOM_M0_CH3_IRQ	ATOM_M0_CH2_IRQ	ATOM_M0_CH1_IRQ	ATOM_M0_CH0_IRQ
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
ATOM_M0_CHx_IRQ (x=0-7)	x	rh	**ATOM[j] channel [x] interrupt ([j]=4*[g]+0)** ***Note**: This bit is only set, when the interrupt is enabled in the interrupt enable register of the corresponding sub-module.* ***Note**: Set this bit represents an **OR** function of the two interrupt sources ATOM_CCU0TC[x]_IRQ or ATOM_CCU1TC[x]_IRQ of **ATOM** instance 0 channel [x].* 0_B no interrupt occurred 1_B interrupt was raised by the corresponding sub-module
ATOM_M1_CHx_IRQ (x=0-7)	x+8	rh	**ATOM[j] channel [x] interrupt ([j]=4*[g]+1)** ***Note**: This bit is only set, when the interrupt is enabled in the interrupt enable register of the corresponding sub-module.* ***Note**: Set this bit represents an **OR** function of the two interrupt sources ATOM_CCU0TC[x]_IRQ or ATOM_CCU1TC[x]_IRQ of **ATOM** instance 1 channel [x].* 0_B no interrupt occurred 1_B interrupt was raised by the corresponding sub-module

(table continues...)

(continued)

Field	Bits	Type	Description
ATOM_M2_CHx_IRQ (x=0-7)	x+16	rh	**ATOMm channel [x] interrupt ([j]=4*[g]+2)** ***Note**: This bit is only set, when the interrupt is enabled in the interrupt enable register of the corresponding sub-module.* ***Note**: Set this bit represents an **OR** function of the two interrupt sources ATOM_CCU0TC[x]_IRQ or ATOM_CCU1TC[x]_IRQ of **ATOM** instance 2 channel [x].* 0_B no interrupt occurred 1_B interrupt was raised by the corresponding sub-module
0	31, 30, 29, 28, 27, 26, 25, 24	r	**Reserved** Read as 0.

44.5.42 ICM_IRQG_TOM_[g]_CI (g=0;i=0)

ICM Interrupt group **TOM** [g] for Channel Interrupt information of **TOM**m ([j]=2*[g]+(0..1))

CLSi_ICM_IRQG_TOMg_CI (g=0;i=0) Offset address: $0005A0_H$+i*20000_H+g*4

ICM_IRQG_TOM_[g]_CI Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TOM_M1_CH15_IRQ	TOM_M1_CH14_IRQ	TOM_M1_CH13_IRQ	TOM_M1_CH12_IRQ	TOM_M1_CH11_IRQ	TOM_M1_CH10_IRQ	TOM_M1_CH9_IRQ	TOM_M1_CH8_IRQ	TOM_M1_CH7_IRQ	TOM_M1_CH6_IRQ	TOM_M1_CH5_IRQ	TOM_M1_CH4_IRQ	TOM_M1_CH3_IRQ	TOM_M1_CH2_IRQ	TOM_M1_CH1_IRQ	TOM_M1_CH0_IRQ
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TOM_M0_CH15_IRQ	TOM_M0_CH14_IRQ	TOM_M0_CH13_IRQ	TOM_M0_CH12_IRQ	TOM_M0_CH11_IRQ	TOM_M0_CH10_IRQ	TOM_M0_CH9_IRQ	TOM_M0_CH8_IRQ	TOM_M0_CH7_IRQ	TOM_M0_CH6_IRQ	TOM_M0_CH5_IRQ	TOM_M0_CH4_IRQ	TOM_M0_CH3_IRQ	TOM_M0_CH2_IRQ	TOM_M0_CH1_IRQ	TOM_M0_CH0_IRQ
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
TOM_M0_CHx_IRQ (x=0-15)	x	rh	**TOM[j] channel [x] interrupt ([j]=2*[g]+0)** ***Note**: This bit is only set, when the interrupt is enabled in the interrupt enable register of the corresponding sub-module.* ***Note**: Set this bit represents an **OR** function of the two interrupt sources TOM_CCU0TC[x]_IRQ or TOM_CCU1TC[x]_IRQ of **TOM** instance 0 channel [x].* 0_B no interrupt occurred 1_B interrupt was raised by the corresponding sub-module

(table continues...)

(continued)

Field	Bits	Type	Description
TOM_M1_CHx_IRQ (x=0-15)	x+16	rh	**TOM[j] channel [x] interrupt ([j]=2*[g]+1)** ***Note**:* *This bit is only set, when the interrupt is enabled in the interrupt enable register of the corresponding sub-module.* ***Note**:* *Set this bit represents an **OR** function of the two interrupt sources TOM_CCU0TC[x]_IRQ or TOM_CCU1TC[x]_IRQ of **TOM** instance 1 channel [x].* 0_B no interrupt occurred 1_B interrupt was raised by the corresponding sub-module

44.5.43 ICM_IRQG_TOM_[g]_CI (g=1;i=0)

For a description of this register, see ICM_IRQG_TOM_[g]_CI (g=0;i=0).

CLSi_ICM_IRQG_TOMg_CI (g=1;i=0) Offset address: $0005A0_H$+i*20000_H+g*4

ICM_IRQG_TOM_[g]_CI Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TOM_M0_CH15_IRQ	TOM_M0_CH14_IRQ	TOM_M0_CH13_IRQ	TOM_M0_CH12_IRQ	TOM_M0_CH11_IRQ	TOM_M0_CH10_IRQ	TOM_M0_CH9_IRQ	TOM_M0_CH8_IRQ	TOM_M0_CH7_IRQ	TOM_M0_CH6_IRQ	TOM_M0_CH5_IRQ	TOM_M0_CH4_IRQ	TOM_M0_CH3_IRQ	TOM_M0_CH2_IRQ	TOM_M0_CH1_IRQ	TOM_M0_CH0_IRQ
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
TOM_M0_CHx_IRQ (x=0-15)	x	rh	**TOM[j] channel [x] interrupt ([j]=2*[g]+0)** ***Note**:* *This bit is only set, when the interrupt is enabled in the interrupt enable register of the corresponding sub-module.* ***Note**:* *Set this bit represents an **OR** function of the two interrupt sources TOM_CCU0TC[x]_IRQ or TOM_CCU1TC[x]_IRQ of **TOM** instance 0 channel [x].* 0_B no interrupt occurred 1_B interrupt was raised by the corresponding sub-module
0	31, 30, 29, 28, 27, 26, 25, 24, 23, 22, 21, 20, 19, 18, 17, 16	r	**Reserved** Read as 0.

44.5.44 MON status register (i=1)

***Note**: Bits [12:0] are set, when a rising edge is detected at the considered clock.*

***Note**: Bits [16:16] and [29:20] are set, when the corresponding unit reports an error.*

CLSi_MON_STATUS (i=1) Offset address: 000680_H+i*20000_H

MON status register Kernel Reset value: $0000\ 4000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		0	0	0	0	0	0	0	0	0	0	0			CMP_ERR
r		r	r	r	r	r	r	r	r	r	r	r			rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	ACT_CMU8	0	ACT_CMUFX4	ACT_CMUFX3	ACT_CMUFX2	ACT_CMUFX1	ACT_CMUFX0	ACT_CMU7	ACT_CMU6	ACT_CMU5	ACT_CMU4	ACT_CMU3	ACT_CMU2	ACT_CMU1	ACT_CMU0
r	rw1ch	r	rw1ch	rw1ch	rw1ch	rw1ch	rw1ch	rw1ch	rw1ch	rw1ch	rw1ch	rw1ch	rw1ch	rw1ch	rw1ch

Field	Bits	Type	Description
ACT_CMUx (x=0-7)	x	rw1ch	CCM[i]_CLK_RES **activity [x]** ***Note**: This bit will be cleared on a CPU write-access of value 1. A read-access leaves the bit unchanged.* **If reading:** 0_B No activity detected for CCM[i]_CLK_RES[x:x] 1_B Activity detected for CCM[i]_CLK_RES[x:x] **If writing:** 0_B No action 1_B Clear activity for CCM[i]_CLK_RES[x:x]
ACT_CMUFXy (y=0-4)	y+8	rw1ch	CCM[i]_FXCLK_RES **activity [y]** ***Note**: This bit will be cleared on a CPU write-access of value 1. A read-access leaves the bit unchanged.* **If reading:** 0_B No activity detected for CCM[i]_FXCLK_RES[y:y] 1_B Activity detected for CCM[i]_FXCLK_RES[y:y] **If writing:** 0_B No action 1_B Clear activity for CCM[i]_FXCLK_RES[y:y]

(table continues...)

(continued)

Field	Bits	Type	Description
ACT_CMU8	14	rw1ch	CCM[i]_CLK_RES*[8:8]* **activity** ***Note**: This bit will be cleared on a CPU write-access of value 1. A read-access leaves the bit unchanged.* ***Note**: Bit is set, when a rising edge is detected at the considered clock.* **If reading:** 0_B No activity detected for CCM[i]_CLK_RES[8:8] 1_B Activity detected for CCM[i]_CLK_RES[8:8] **If writing:** 0_B No action 1_B Clear activity for CCM[i]_CLK_RES[8:8]
CMP_ERR	16	rh	**Error detected at CMP** ***Note**: This bit will be readable only.* 0_B No error detected 1_B Error detected
0	13, 15, 19:17, 29, 28, 27, 26, 25, 24, 23, 22, 21, 20, 31:30	r	**Reserved** Read as 0; must be written with 0.

44.5.45 CMP comparator enable register (i=1)

CLSi_CMP_EN (i=1) Offset address: $0006C0_H+i*20000_H$

CMP comparator enable register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								TBWC11_EN	TBWC10_EN	TBWC9_EN	TBWC8_EN	TBWC7_EN	TBWC6_EN	TBWC5_EN	TBWC4_EN
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TBWC3_EN	TBWC2_EN	TBWC1_EN	TBWC0_EN	ABWC11_EN	ABWC10_EN	ABWC9_EN	ABWC8_EN	ABWC7_EN	ABWC6_EN	ABWC5_EN	ABWC4_EN	ABWC3_EN	ABWC2_EN	ABWC1_EN	ABWC0_EN
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ABWCc_EN (c=0-11)	c	rw	**Enable comparator channel [c] in ABWC** (see Bitwise compare unit (BWC)) 0_B ABWC Comparator [c] is disabled 1_B ABWC Comparator [c] is enabled
TBWCc_EN (c=0-11)	c+12	rw	**Enable comparator channel [c] in TBWC** (see Bitwise compare unit (BWC)) 0_B TBWC comparator [c] is disabled 1_B TBWC comparator [c] is enabled
0	31:24	r	**Reserved** Read as 0; must be written with 0.

44.5.46 CMP event notification register (i=1)

CLSi_CMP_IRQ_NOTIFY (i=1) Offset address: $0006C4_H+i*20000_H$

CMP event notification register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								TBWC11	TBWC10	TBWC9	TBWC8	TBWC7	TBWC6	TBWC5	TBWC4
r								rw1ch	rw1ch	rw1ch	rw1ch	rw1ch	rw1ch	rw1ch	rw1ch

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TBWC3	TBWC2	TBWC1	TBWC0	ABWC11	ABWC10	ABWC9	ABWC8	ABWC7	ABWC6	ABWC5	ABWC4	ABWC3	ABWC2	ABWC1	ABWC0
rw1ch	rw1ch	rw1ch	rw1ch	rw1ch	rw1ch	rw1ch	rw1ch	rw1ch	rw1ch	rw1ch	rw1ch	rw1ch	rw1ch	rw1ch	rw1ch

Field	Bits	Type	Description
ABWCc (c=0-11)	c	rw1ch	**ATOM sub-modules output bitwise comparator [c] error indication** ***Note**: This bit will be cleared on a CPU write-access of value '1'. A read-access leaves the bit unchanged.* **If reading:** 0_B No interrupt raised 1_B Interrupt was raised **If writing:** 0_B No action 1_B Clear interrupt

(table continues...)

(continued)

Field	Bits	Type	Description
TBWCc (c=0-11)	c+12	rw1ch	**TOM sub-modules output bitwise comparator [c] error indication** ***Note**: This bit will be cleared on a CPU write-access of value '1'. A read-access leaves the bit unchanged.* **If reading:** 0_B No interrupt raised 1_B Interrupt was raised **If writing:** 0_B No action 1_B Clear interrupt
0	31:24	r	**Reserved** Read as 0; must be written with 0.

44.5.47 CMP interrupt enable register (i=1)

CLSi_CMP_IRQ_EN (i=1) Offset address: $0006C8_H+i*20000_H$

CMP interrupt enable register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								TBWC11_EN_IRQ	TBWC10_EN_IRQ	TBWC9_EN_IRQ	TBWC8_EN_IRQ	TBWC7_EN_IRQ	TBWC6_EN_IRQ	TBWC5_EN_IRQ	TBWC4_EN_IRQ
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TBWC3_EN_IRQ	TBWC2_EN_IRQ	TBWC1_EN_IRQ	TBWC0_EN_IRQ	ABWC11_EN_IRQ	ABWC10_EN_IRQ	ABWC9_EN_IRQ	ABWC8_EN_IRQ	ABWC7_EN_IRQ	ABWC6_EN_IRQ	ABWC5_EN_IRQ	ABWC4_EN_IRQ	ABWC3_EN_IRQ	ABWC2_EN_IRQ	ABWC1_EN_IRQ	ABWC0_EN_IRQ
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ABWCc_EN_IRQ (c=0-11)	c	rw	**Enable** CLSi_CMP_IRQ_NOTIFY (i=1).ABWCc **(ATOM [c]) interrupt source for** CMP_IRQ **line** 0_B interrupt source of ABWC comparator [c] is disabled 1_B interrupt source of ABWC comparator [c] is enabled
TBWCc_EN_IRQ (c=0-11)	c+12	rw	**Enable** CLSi_CMP_IRQ_NOTIFY (i=1).TBWCc **(TOM [c]) interrupt source for** CMP_IRQ **line** 0_B interrupt source of TBWC comparator [c] is disabled 1_B interrupt source of TBWC comparator [c] is enabled
0	31:24	r	**Reserved** Read as 0; must be written with 0.

44.5.48 CMP interrupt force register (i=1)

CLSi_CMP_IRQ_FORCINT (i=1) Offset address: $0006CC_H+i*20000_H$

CMP interrupt force register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								TRG_TBWC11	TRG_TBWC10	TRG_TBWC9	TRG_TBWC8	TRG_TBWC7	TRG_TBWC6	TRG_TBWC5	TRG_TBWC4
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TRG_TBWC3	TRG_TBWC2	TRG_TBWC1	TRG_TBWC0	TRG_ABWC11	TRG_ABWC10	TRG_ABWC9	TRG_ABWC8	TRG_ABWC7	TRG_ABWC6	TRG_ABWC5	TRG_ABWC4	TRG_ABWC3	TRG_ABWC2	TRG_ABWC1	TRG_ABWC0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
TRG_ABWCc (c=0-11)	c	rw	**Trigger the** CLSi_CMP_IRQ_NOTIFY (i=1)**.**ABWCc **bit (ATOM [c]) by software** ***Note**: This bit is cleared automatically after write.* ***Note**: This bit is write-protected by bit* CLSi_ARCH_CTRL**.**RF_PROT*.* **If reading:** 0_B No status **If writing:** 0_B No action 1_B Force setting of CLSi_CMP_IRQ_NOTIFY (i=1).ABWCc
TRG_TBWCc (c=0-11)	c+12	rw	**Trigger** CLSi_CMP_IRQ_NOTIFY (i=1)**.**TBWCc **bit (TOM [c]) by software** ***Note**: This bit is cleared automatically after write.* ***Note**: This bit is write-protected by bit* CLSi_ARCH_CTRL**.**RF_PROT*.* **If reading:** 0_B No status **If writing:** 0_B No action 1_B Force setting of CLSi_CMP_IRQ_NOTIFY (i=1).TBWCc
0	31:24	r	**Reserved** Read as 0; must be written with 0.

44.5.49 CMP interrupt mode configuration register (i=1)

CLSi_CMP_IRQ_MODE (i=1) Offset address: $0006D0_H+i*20000_H$

CMP interrupt mode configuration register Kernel Reset value: $0000\ 0002_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0														IRQ_MODE	
r														rw	

Field	Bits	Type	Description
IRQ_MODE	1:0	rw	**IRQ mode selection** ***Note**: The interrupt modes are described in **eGTM** interrupt concept.* 00_B Level mode 01_B Pulse mode 10_B Pulse-Notify mode 11_B Single-Pulse mode
0	31:2	r	**Reserved** Read as 0; must be written with 0.

44.5.50 CMP error interrupt enable register (i=1)

CLSi_CMP_EIRQ_EN (i=1) Offset address: $0006D4_H+i*20000_H$

CMP error interrupt enable register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								TBWC11_EN_EIRQ	TBWC10_EN_EIRQ	TBWC9_EN_EIRQ	TBWC8_EN_EIRQ	TBWC7_EN_EIRQ	TBWC6_EN_EIRQ	TBWC5_EN_EIRQ	TBWC4_EN_EIRQ
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TBWC3_EN_EIRQ	TBWC2_EN_EIRQ	TBWC1_EN_EIRQ	TBWC0_EN_EIRQ	ABWC11_EN_EIRQ	ABWC10_EN_EIRQ	ABWC9_EN_EIRQ	ABWC8_EN_EIRQ	ABWC7_EN_EIRQ	ABWC6_EN_EIRQ	ABWC5_EN_EIRQ	ABWC4_EN_EIRQ	ABWC3_EN_EIRQ	ABWC2_EN_EIRQ	ABWC1_EN_EIRQ	ABWC0_EN_EIRQ
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ABWCc_EN_EIRQ (c=0-11)	c	rw	**Enable ABWC comparator [c] interrupt source for CMP_EIRQ line** 0_B interrupt source of ABWC comparator [c] is disabled 1_B interrupt source of ABWC comparator [c] is enabled
TBWCc_EN_EIRQ (c=0-11)	c+12	rw	**Enable TBWC comparator [c] interrupt source for CMP_EIRQ line** 0_B interrupt source of TBWC comparator [c] is disabled 1_B interrupt source of TBWC comparator [c] is enabled
0	31:24	r	**Reserved** Read as 0; must be written with 0.

44.5.51 TIM[i] channel [x] general-purpose 0 register (x=0-7;i=0-2)

CLSi_TIM_CHx_GPR0 (i=0-2;x=0-7) Offset address: 000800_H+i*20000_H+x*80_H

TIM[i] channel [x] general-purpose 0 register Reset values see: Table 2309

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECNT								GPR0							
rh								rwh							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPR0															
rwh															

Field	Bits	Type	Description
GPR0	23:0	rwh	**Input signal characteristic parameter 0** ***Note**: The content of this register has different meaning for the **TIM** channels modes. The content directly depends on the bit fields CLSi_TIM_CHx_CTRL.EGPR0_SEL, CLSi_TIM_CHx_CTRL.GPR0_SEL.* ***Note**: The content of this register can only be written in **TIM** channel mode TSSM.*

(table continues...)

(continued)

Field	Bits	Type	Description
ECNT	31:24	rh	**Edge counter, value refers to** CLSi_TIM_CHx_ECNT**.**ECNT **[7:0].** ***Note**:* *The* CLSi_TIM_CHx_GPR0**.**ECNT *counts every incoming filtered edge (rising and falling). The counter value is uneven in case of detected rising, and even in case of detected falling edge. Thus, the input signal level is part of the counter and can be obtained by bit 0 of* CLSi_TIM_CHx_GPR0**.**ECNT*.* ***Note**:* *The* CLSi_TIM_CHx_GPR0**.**ECNT *register is reset to its initial value when the channel is enabled.* ***Note**:* *Bit 0 depends on the input level coming from the filter unit and defines this value.*

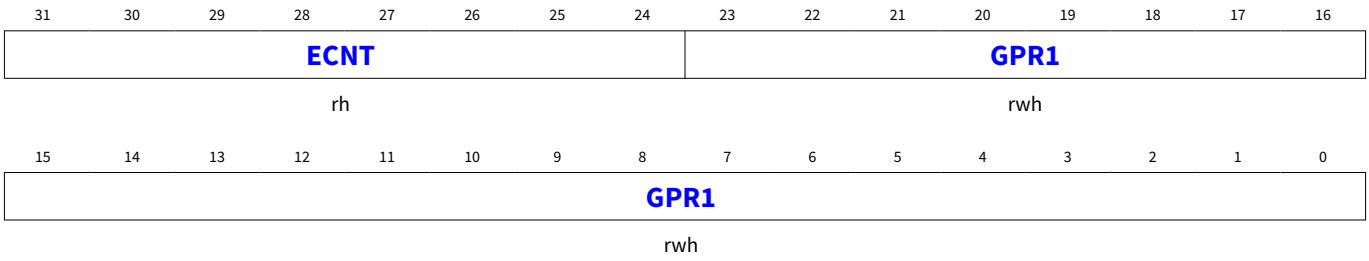
Table 2309 Reset values of CLSi_TIM_CHx_GPR0 (i=0-2;x=0-7)

Reset	Reset value	Note
Kernel Reset	0000 0000$_H$	
TIM channel reset	0000 0000$_H$	

44.5.52 TIM[i] channel [x] general-purpose 1 register (x=0-7;i=0-2)

CLSi_TIM_CHx_GPR1 (i=0-2;x=0-7) Offset address: 000804$_H$+i*20000$_H$+x*80$_H$

TIM[i] channel [x] general-purpose 1 register Reset values see: Table 2310



Field	Bits	Type	Description
GPR1	23:0	rwh	**Input signal characteristic parameter 1** ***Note**:* *The content of this register has different meaning for the **TIM** channels modes. The content directly depends on the bit fields CLSi_TIM_CHx_CTRL.EGPR1_SEL, CLSi_TIM_CHx_CTRL.GPR1_SEL.* ***Note**:* *In TBCM mode if CLSi_TIM_CHx_CTRL.EGPR1_SEL=1, CLSi_TIM_CHx_CTRL.GPR1_SEL= 01_B then CLSi_TIM_INP_VAL is used as input in all other cases **TIM** Filter F_OUT is used as input and Bits CLSi_TIM_CHx_GPR1[23:8] = 0* ***Note**:* *The content of this register can only be written in **TIM** channel mode TGPS and TSSM.*
ECNT	31:24	rh	**Edge counter, value refers to CLSi_TIM_CHx_ECNT.ECNT [7:0].** ***Note**:* *The CLSi_TIM_CHx_GPR1.ECNT counts every incoming filtered edge (rising and falling). The counter value is uneven in case of detected rising, and even in case of detected falling edge. Thus, the input signal level is part of the counter and can be obtained by bit 0 of CLSi_TIM_CHx_GPR1.ECNT.* ***Note**:* *The CLSi_TIM_CHx_GPR1.ECNT register is reset to its initial value when the channel is enabled.* ***Note**:* *Bit 0 depends on the input level coming from the filter unit and defines this value.*

Table 2310 Reset values of CLSi_TIM_CHx_GPR1 (i=0-2;x=0-7)

Reset	Reset value	Note
Kernel Reset	0000 0000_H	
TIM channel reset	0000 0000_H	

44.5.53 TIM[i] channel [x] SMU counter register (x=0-7;i=0-2)

CLSi_TIM_CHx_CNT (i=0-2;x=0-7) Offset address: 000808_H+i*20000_H+x*80_H

TIM[i] channel [x] SMU counter register Reset values see: Table 2311

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								CNT							
r								rh							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CNT															
rh															

Field	Bits	Type	Description
CNT	23:0	rh	**Actual SMU counter value** ***Note**:* *The meaning of this value depends on the configured mode:* *TPWM = actual duration of PWM signal.* *TPIM = actual duration of all pulses (sum of pulses).* *TIEM = actual number of received edges.* *TIPM = actual number of received edges.* *TGPS = elapsed time for periodic sampling.* *TSSM = shift data.*
0	31:24	r	**Reserved** Read as 0.

Table 2311 Reset values of CLSi_TIM_CHx_CNT (i=0-2;x=0-7)

Reset	Reset value	Note
Kernel Reset	$0000\ 0000_H$	
TIM channel reset	$0000\ 0000_H$	

44.5.54 TIM[i] channel [x] SMU edge counter register (x=0-7;i=0-2)

CLSi_TIM_CHx_ECNT (i=0-2;x=0-7) Offset address: $00080C_H+i*20000_H+x*80_H$

TIM[i] channel [x] SMU edge counter register Reset values see: Table 2312

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECNT															
rh															

Field	Bits	Type	Description
ECNT	15:0	rh	**Edge counter** ***Note**:* *The CLSi_TIM_CHx_ECNT register is reset to its initial value when the channel is enabled. Bit 0 depends on the input level coming from the filter unit and defines this value.* ***Note**:* *If **TIM** channel is disabled the content of CLSi_TIM_CHx_ECNT.ECNT gets frozen and a read will auto clear the bits [15:1]. Further read-accesses to CLSi_TIM_CHx_ECNT.ECNT will show on Bit 0 the actual input signal value of the channel.*
0	31:16	r	**Reserved** Read as 0.

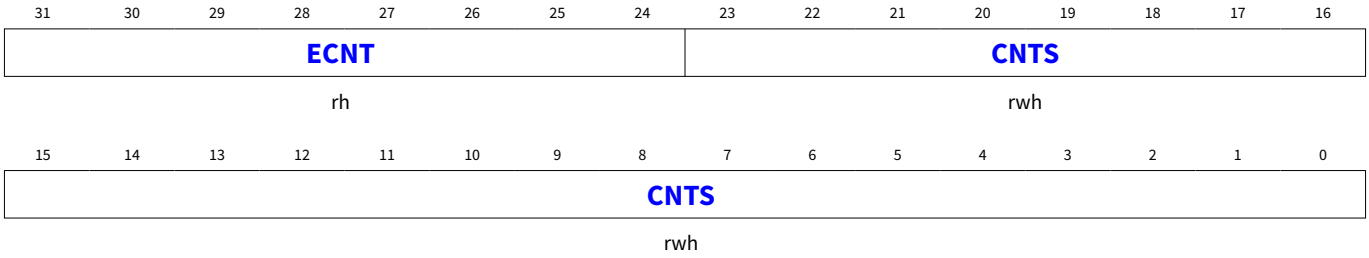
Table 2312 Reset values of CLSi_TIM_CHx_ECNT (i=0-2;x=0-7)

Reset	Reset value	Note
Kernel Reset	0000 0000$_H$	
TIM channel reset	0000 0000$_H$	

44.5.55 TIM[i] channel [x] SMU shadow counter register (x=0-7;i=0-2)

CLSi_TIM_CHx_CNTS (i=0-2;x=0-7) Offset address: 000810$_H$+i*20000$_H$+x*80$_H$

TIM[i] channel [x] SMU shadow counter register Reset values see: Table 2313



Field	Bits	Type	Description
CNTS	23:0	rwh	**Counter shadow register** ***Note**: The content of this register has different meaning for the **TIM** channels modes. The content depends directly on the bit field CLSi_TIM_CHx_CTRL.CNTS_SEL.* ***Note**: The register CLSi_TIM_CHx_CNTS is only writable in TIPM,TBCM ,TGPS and TSSM mode.*
ECNT	31:24	rh	**Edge counter** ***Note**: The CLSi_TIM_CHx_CNTS.ECNT counts every incoming filtered edge (rising and falling). The counter value is uneven in case of detected rising, and even in case of detected falling edge. Thus, the input signal level is part of the counter and can be obtained by bit 0 of CLSi_TIM_CHx_CNTS.ECNT.* ***Note**: The CLSi_TIM_CHx_CNTS.ECNT register is reset to its initial value when the channel is enabled.* ***Note**: Bit 0 depends on the input level coming from the filter unit and defines the reset value immediately.*

Table 2313 Reset values of CLSi_TIM_CHx_CNTS (i=0-2;x=0-7)

Reset	Reset value	Note
Kernel Reset	0000 0000$_H$	
TIM channel reset	0000 0000$_H$	

44.5.56 TIM[i]_CH[x]_TDUC (x=0-7;i=0-2)

TIM[i] channel [x] TDU counter register

***Note**: The register CLSi_TIM_CHx_TDUC is writable if Timeout unit is disabled (CLSi_TIM_CHx_CTRL.TOCTRL=00_B).*

***Note**: If CLSi_TIM_CHx_ECTRL.USE_LUT != 00_B (input signal generation by lookup-table) the bit field CLSi_TIM_CHx_TDUC.TO_CNT2 is writable at any time, CLSi_TIM_CHx_TDUC.TO_CNT, CLSi_TIM_CHx_TDUC.TO_CNT1 will not be changed.*

CLSi_TIM_CHx_TDUC (i=0-2;x=0-7) Offset address: $000814_H+i*20000_H+x*80_H$

TIM[i]_CH[x]_TDUC Reset values see: Table 2314

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								**TO_CNT2**							
r								rwh							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TO_CNT1								**TO_CNT**							
rwh								rwh							

Field	Bits	Type	Description
TO_CNT	7:0	rwh	**Current Timeout value slice0 for channel [y]** If bit field CLSi_TIM_CHx_TDUV.SLICING ~= 3: counter will be reset to 00_H on CLSi_TIM_CHx_ECTRL.TDU_RESYNC condition If bit field CLSi_TIM_CHx_TDUV.SLICING == 3: counter will be loaded with CLSi_TIM_CHx_TDUV.TOV2 on CLSi_TIM_CHx_ECTRL.TDU_RESYNC condition
TO_CNT1	15:8	rwh	**Current Timeout value slice1 for channel [y]** Counter will be reset to 00_H on CLSi_TIM_CHx_ECTRL.TDU_RESYNC condition
TO_CNT2	23:16	rwh	**Current Timeout value slice2 for channel [y]** Counter will be reset to 00_H on CLSi_TIM_CHx_ECTRL.TDU_RESYNC condition
0	31:24	r	**Reserved** Read as 0; must be written with 0.

Table 2314 Reset values of CLSi_TIM_CHx_TDUC (i=0-2;x=0-7)

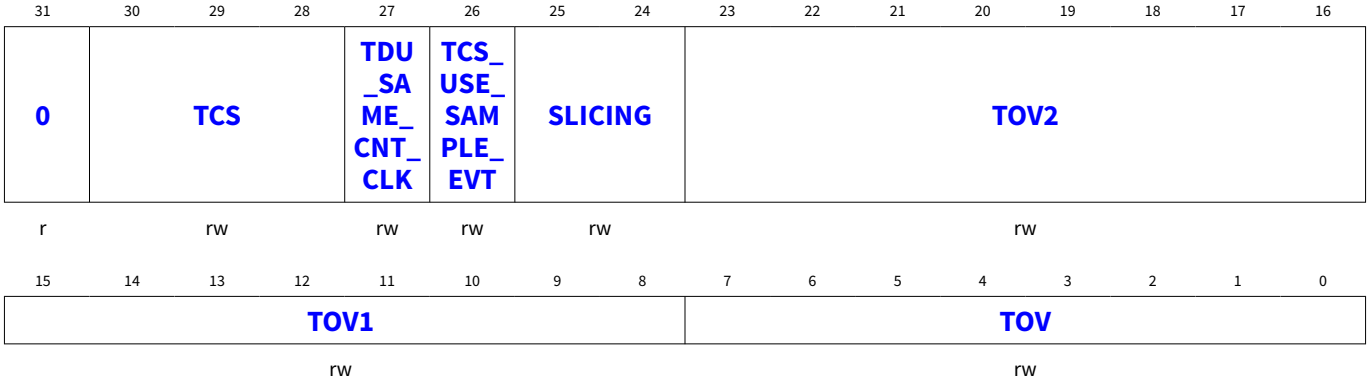
Reset	Reset value	Note
Kernel Reset	$0000\ 0000_H$	
TIM channel reset	$0000\ 0000_H$	

44.5.57 TIM[i]_CH[x]_TDUV (x=0-7;i=0-2)

TIM[i] channel [x] TDU control register

CLSi_TIM_CHx_TDUV (i=0-2;x=0-7) Offset address: 000818_H+i*20000_H+x*80_H

TIM[i]_CH[x]_TDUV Reset values see: Table 2315



Field	Bits	Type	Description
TOV	7:0	rw	**Time out compare value slice0 for channel [y]** Compare value for CLSi_TIM_CHx_TDUC.TO_CNT
TOV1	15:8	rw	**Time out compare value slice1 for channel [y]** Compare value for CLSi_TIM_CHx_TDUC.TO_CNT1
TOV2	23:16	rw	**Time out compare value slice2 for channel [y]** If bit field CLSi_TIM_CHx_TDUV.SLICING ~= 3: Compare value for CLSi_TIM_CHx_TDUC.TO_CNT2 If bit field CLSi_TIM_CHx_TDUV.SLICING == 3: CLSi_TIM_CHx_TDUV.TOV2 operates as a shadow register for CLSi_TIM_CHx_TDUC.TO_CNT
SLICING	25:24	rw	**Cascading of counter slices** **If** CLSi_TIM_CHx_ECTRL **.USE_LUT == 0:** 00_B Combine slice2, slice1, slice0 to one 24-bit counter 01_B Combine slice1, slice0 to one 16-bit counter; use slice2 as one 8-bit counter 10_B Use slice2, slice1, slice0 as three 8-bit counters 11_B Use slice1, slice0 as two 8-bit counters **If** CLSi_TIM_CHx_ECTRL **.USE_LUT !=0:** 01_B Combine slice1, slice0 to one 16-bit counter; slice2 not usable 10_B Use slice1, slice0 as two 8-bit counters; slice2 not usable 11_B Use slice1, slice0 as two 8-bit counters
TCS_USE_SAMPLE_EVT	26	rw	**Use** TDU_SAMPLE_EVT **as Timeout Clock** 0_B CCM[i]_CLK_RES selected by CLSi_TIM_CHx_TDUV.TCS is in use by CLSi_TIM_CHx_TDUC.TO_CNT, CLSi_TIM_CHx_TDUC.TO_CNT2 1_B CCM[i]_CLK_RES selected by CLSi_TIM_CHx_TDUV.TCS is in use by CLSi_TIM_CHx_TDUC.TO_CNT2; TDU_SAMPLE_EVT is in use by CLSi_TIM_CHx_TDUC.TO_CNT

(table continues...)

(continued)

Field	Bits	Type	Description
TDU_SAME_CNT_CLK	27	rw	**Define clocking of TO_CNT, TO_CNT1** 0_B CLSi_TIM_CHx_TDUC.TO_CNT clock selected by (TCS, CLSi_TIM_CHx_TDUV.TCS_USE_SAMPLE_EVT); CLSi_TIM_CHx_TDUC.TO_CNT1 clocked on tdu_word_event 1_B CLSi_TIM_CHx_TDUC.TO_CNT1 uses same clock as CLSi_TIM_CHx_TDUC.TO_CNT
TCS	30:28	rw	**Timeout clock selection** 000_B CCM[i]_CLK_RES[0:0] resolution in use 001_B CCM[i]_CLK_RES[1:1] resolution in use 010_B CCM[i]_CLK_RES[2:2] resolution in use 011_B CCM[i]_CLK_RES[3:3] resolution in use 100_B CCM[i]_CLK_RES[4:4] resolution in use 101_B CCM[i]_CLK_RES[5:5] resolution in use 110_B CCM[i]_CLK_RES[6:6] resolution in use 111_B CCM[i]_CLK_RES[7:7] resolution in use
0	31	r	**Reserved** Read as 0; must be written with 0.

Table 2315 Reset values of CLSi_TIM_CHx_TDUV (i=0-2;x=0-7)

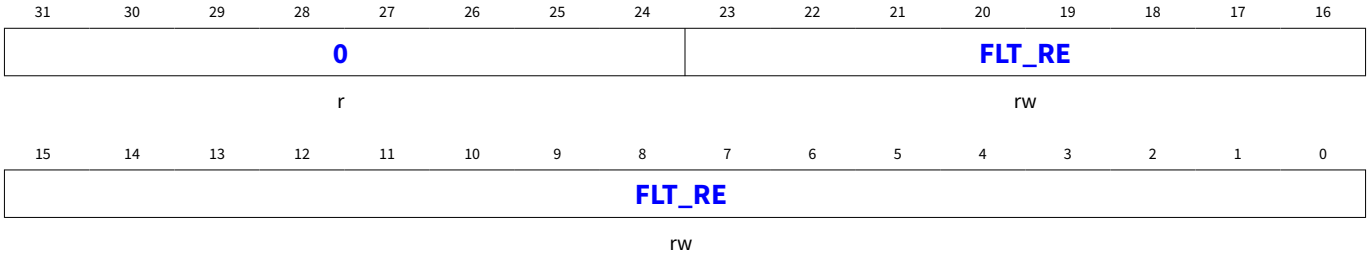
Reset	Reset value	Note
Kernel Reset	$0000\ 0000_H$	
TIM channel reset	$0000\ 0000_H$	

44.5.58 TIM[i] channel [x] filter parameter 0 register (x=0-7;i=0-2)

***Note**:* CLSi_TIM_CHx_FLT_RE *has different meanings in the various filter modes:*
Immediate edge propagation mode = acceptance time for rising edge
Individual deglitch time mode = deglitch time for rising edge

CLSi_TIM_CHx_FLT_RE (i=0-2;x=0-7) Offset address: $00081C_H+i*20000_H+x*80_H$

TIM[i] channel [x] filter parameter 0 register Reset values see: Table 2316



Field	Bits	Type	Description
FLT_RE	23:0	rw	**Filter parameter for rising edge**

(table continues...)

(continued)

Field	Bits	Type	Description
0	31:24	r	**Reserved** Read as 0; must be written with 0.

Table 2316 Reset values of CLSi_TIM_CHx_FLT_RE (i=0-2;x=0-7)

Reset	Reset value	Note
Kernel Reset	0000 0000$_H$	
TIM channel reset	0000 0000$_H$	

44.5.59 TIM[i] channel [x] filter parameter 1 register (x=0-7;i=0-2)

***Note**: CLSi_TIM_CHx_FLT_FE has different meanings in the various filter modes:*
Immediate edge propagation mode = acceptance time for falling edge
Individual deglitch time mode = deglitch time for falling edge

CLSi_TIM_CHx_FLT_FE (i=0-2;x=0-7) Offset address: 000820$_H$+i*20000$_H$+x*80$_H$

TIM[i] channel [x] filter parameter 1 register Reset values see: Table 2317

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								FLT_FE							
r								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FLT_FE															
rw															

Field	Bits	Type	Description
FLT_FE	23:0	rw	**Filter parameter for falling edge**
0	31:24	r	**Reserved** Read as 0; must be written with 0.

Table 2317 Reset values of CLSi_TIM_CHx_FLT_FE (i=0-2;x=0-7)

Reset	Reset value	Note
Kernel Reset	0000 0000$_H$	
TIM channel reset	0000 0000$_H$	

44.5.60 TIM[i] channel [x] control register (x=0-7;i=0)

CLSi_TIM_CHx_CTRL (i=0;x=0-7) Offset address: 000824$_H$+i*20000$_H$+x*80$_H$

TIM[i] channel [x] control register Reset values see: Table 2318

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TOCTRL		EGPR1_SEL	EGPR0_SEL	FR_ECNT_OFL	CLK_SEL			FLT_CTR_FE	FLT_MODE_FE	FLT_CTR_RE	FLT_MODE_RE	EXT_CAP_EN	FLT_CNT_FRQ		FLT_EN
rw		rw	rw	rw	rw			rw	rw	rw	rw	rw	rw		rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECNT_RESET	ISL	DSL	CNTS_SEL	GPR1_SEL		GPR0_SEL		TBU0_SEL	CICTRL	0	OSM	TIM_MODE			TIM_EN
rw	rw	rw	rw	rw		rw		rw	rw	r	rw	rw			rwh

Field	Bits	Type	Description
TIM_EN	0	rwh	**TIM channel [x] enable** ***Note**:* *Enabling of the channel resets the registers* CLSi_TIM_CHx_ECNT, CLSi_TIM_CHx_CNT, CLSi_TIM_CHx_GPR0, *and* CLSi_TIM_CHx_GPR1 *to their reset values.* ***Note**:* *After finishing the action in one-shot mode the* CLSi_TIM_CHx_CTRL**.**TIM_EN *bit is cleared automatically. Otherwise, the bit must be cleared manually.* 0_B Channel disabled 1_B Channel enabled

(table continues...)

(continued)

Field	Bits	Type	Description
TIM_MODE	3:1	rw	**TIM channel [x] mode** ***Note**: The CLSi_TIM_CHx_CTRL.TIM_MODE register should not be changed while the **TIM** channel is enabled.* ***Note**: If the **TIM** channel is enabled and operating in TPWM or TPIM mode after the first valid edge defined by CLSi_TIM_CHx_CTRL.DSL has occurred, a reconfiguration of CLSi_TIM_CHx_CTRL.DSL, CLSi_TIM_CHx_CTRL.ISL, CLSi_TIM_CHx_CTRL.TIM_MODE will not change the channel behavior. Reading these bit fields after reconfiguration will show the newly configured settings but the initial channel behavior will not change. Only a disabling of the **TIM** channel by setting CLSi_TIM_CHx_CTRL.TIM_EN= 0 and reenabling with CLSi_TIM_CHx_CTRL.TIM_EN= 1 will change the channel operation mode.* **If reading:** 000_B PWM Measurement Mode (TPWM) 001_B Pulse Integration Mode (TPIM) 010_B Input Event Mode (TIEM) 011_B Input Prescaler Mode (TIPM) 100_B Bit Compression Mode (TBCM) 101_B Gated Periodic Sampling Mode (TGPS) 110_B Serial Shift Mode (TSSM) 111_B unused **If writing:** 000_B PWM Measurement Mode (TPWM) 001_B Pulse Integration Mode (TPIM) 010_B Input Event Mode (TIEM) 011_B Input Prescaler Mode (TIPM) 100_B Bit Compression Mode (TBCM) 101_B Gated Periodic Sampling Mode (TGPS) 110_B Serial Shift Mode (TSSM) 111_B PWM Measurement Mode (TPWM) 000_B should be used, 111_B coding might be used for new defined functionality in future devices
OSM	4	rw	**One-shot mode** ***Note**: After finishing the action in one-shot mode the CLSi_TIM_CHx_CTRL.TIM_EN bit is cleared automatically.* 0_B Continuous operation mode 1_B One-shot mode
CICTRL	6	rw	**Channel Input Control** 0_B Use signal TIM_IN[x:x] as input for channel x 1_B Use signal TIM_IN[x-1:x-1] as input for channel x (or TIM_IN[7:7] of instance [i])

(table continues...)

(continued)

Field	Bits	Type	Description
TBU0_SEL	7	rw	TIM_TBU_TS0 **bits input select for** CLSi_TIM_CHx_GPR0**(i=0) and** CLSi_TIM_CHx_GPR1**(i=0)** ***Note**: This bit is only applicable for TIM0* 0_B Use TIM_TBU_TS0[23:0] to store in CLSi_TIM_CHx_GPR0 (i=0)/ CLSi_TIM_CHx_GPR1 (i=0) 1_B Use TIM_TBU_TS0[26:3] to store in CLSi_TIM_CHx_GPR0 (i=0)/ CLSi_TIM_CHx_GPR1 (i=0)
GPR0_SEL	9:8	rw	**Selection for** CLSi_TIM_CHx_GPR0 **register** **If** CLSi_TIM_CHx_CTRL **.EGPR0_SEL == 0:** 00_B Use TIM_TBU_TS0 as input 01_B Use TIM_TBU_TS1 as input 10_B Use TIM_TBU_TS2 as input 11_B Use CLSi_TIM_CHx_CNTS as input; if TGPS mode in channel 0 is selected use Filter F_OUT as input **If** CLSi_TIM_CHx_CTRL **.EGPR0_SEL == 1:** 00_B Use CLSi_TIM_CHx_ECNT as input 01_B Use CLSi_TIM_INP_VAL as input 10_B Prohibited 11_B Prohibited
GPR1_SEL	11:10	rw	**Selection for** CLSi_TIM_CHx_GPR1 **register** **If** CLSi_TIM_CHx_CTRL **.TIM_MODE !=4 &&** CLSi_TIM_CHx_CTRL **.EGPR1_SEL == 0:** 00_B Use TIM_TBU_TS0 as input 01_B Use TIM_TBU_TS1 as input 10_B Use TIM_TBU_TS2 as input 11_B Use CLSi_TIM_CHx_CNT as input **If** CLSi_TIM_CHx_CTRL **.TIM_MODE !=4 &&** CLSi_TIM_CHx_CTRL **.EGPR1_SEL == 1:** 00_B Use CLSi_TIM_CHx_ECNT as input 01_B Use CLSi_TIM_INP_VAL as input 10_B Prohibited 11_B Prohibited **If** CLSi_TIM_CHx_CTRL **.TIM_MODE == 4 &&** CLSi_TIM_CHx_CTRL **.EGPR1_SEL == 0:** 00_B Use Filter F_OUT as input 01_B Use Filter F_OUT as input 10_B Use Filter F_OUT as input 11_B Use Filter F_OUT as input **If** CLSi_TIM_CHx_CTRL **.TIM_MODE == 4 &&** CLSi_TIM_CHx_CTRL **.EGPR1_SEL == 1:** 00_B Use Filter F_OUT as input 01_B Use CLSi_TIM_INP_VAL as input 10_B Prohibited 11_B Prohibited

(table continues...)

(continued)

Field	Bits	Type	Description
CNTS_SEL	12	rw	**Selection for** CLSi_TIM_CHx_CNTS **register** ***Note:*** CLSi_TIM_CHx_CTRL**.**CNTS_SEL *in TSSM mode selects the source signal for registered or latched shift out operation.* *0 = Use* F_OUT*[x:x]* *1 = Use* TIM_IN*[x:x]* 0_B Use CLSi_TIM_CHx_CNT register as input 1_B Use TIM_TBU_TS0 as input
DSL	13	rw	**Signal level control** ***Note:*** *In* CLSi_TIM_CHx_CTRL**.**TIM_MODE*=110_B (TSSM) the bit field* CLSi_TIM_CHx_CTRL**.**DSL *defines the shift direction.* *0 = Shift left* *1 = Shift right* 0_B Measurement starts with falling edge (level low measurement) 1_B Measurement starts with rising edge (level high measurement)
ISL	14	rw	**Ignore signal level** This bit is mode dependent and will have different meanings (see details in the **TIM** Channel mode description). 0_B Use CLSi_TIM_CHx_CTRL.DSL bit for selecting active signal level (TIEM) 1_B Ignore CLSi_TIM_CHx_CTRL.DSL and treat both edges as active edge (TIEM)

(table continues...)

(continued)

Field	Bits	Type	Description
ECNT_RESET	15	rw	**Enables resetting of counter in certain modes** **If** CLSi_TIM_CHx_CTRL **.TIM_MODE == 0:** 0_B CLSi_TIM_CHx_ECNT.ECNT counter operating in wrap around mode, CLSi_TIM_CHx_CNT.CNT is reset on active input edge defined by CLSi_TIM_CHx_CTRL.DSL 1_B CLSi_TIM_CHx_ECNT.ECNT counter operating in wrap around mode, CLSi_TIM_CHx_CNT.CNT is reset on active and inactive input edge **If** CLSi_TIM_CHx_CTRL **.TIM_MODE == 5:** 0_B CLSi_TIM_CHx_ECNT.ECNT counter operating in wrap around mode 1_B CLSi_TIM_CHx_ECNT.ECNT counter is reset with periodic sampling **If** CLSi_TIM_CHx_CTRL **.TIM_MODE == 6:** 0_B Input shift value TSSM_IN[x:x]=0 is in use with CLSi_TIM_CHx_CTRL.ISL=1 1_B Input shift value TSSM_IN[x:x]=1 is in use with CLSi_TIM_CHx_CTRL.ISL=1 **If (**CLSi_TIM_CHx_CTRL **.TIM_MODE !=0 &&** CLSi_TIM_CHx_CTRL **.TIM_MODE !=5) &&** CLSi_TIM_CHx_CTRL **.TIM_MODE !=6:** 0_B CLSi_TIM_CHx_ECNT.ECNT counter operating in wrap around mode 1_B CLSi_TIM_CHx_ECNT.ECNT counter operating in wrap around mode
FLT_EN	16	rw	**Filter enable for channel [x]** ***Note**: If the filter is disabled all filter related units (including CSU) are bypassed, which means that the signal F_IN is directly routed to signal F_OUT.* 0_B Filter disabled and internal states are reset 1_B Filter enabled
FLT_CNT_FRQ	18:17	rw	**Filter counter frequency select** 00_B CCM[i]_CLK_RES[0:0] resolution in use 01_B CCM[i]_CLK_RES[1:1] resolution in use 10_B CCM[i]_CLK_RES[6:6] resolution in use 11_B CCM[i]_CLK_RES[7:7] resolution in use
EXT_CAP_EN	19	rw	**Enables external capture mode. The selected TIM mode is only sensitive to external capture pulses the input event changes are ignored.** 0_B External capture disabled 1_B External capture enabled
FLT_MODE_RE	20	rw	**Filter mode for rising edge** 0_B Immediate edge propagation mode 1_B Individual deglitch mode

(table continues...)

(continued)

Field	Bits	Type	Description
FLT_CTR_RE	21	rw	**Filter counter mode for rising edge** **If** CLSi_TIM_CHx_CTRL **.FLT_MODE_RE == 1 &&** CLSi_TIM_CHx_ECTRL **.EFLT_CTR_RE == 0:** 0_B Up-Down Counter individual deglitch mode 1_B Hold Counter individual deglitch mode **If** CLSi_TIM_CHx_CTRL **.FLT_MODE_RE == 0 &&** CLSi_TIM_CHx_ECTRL **.EFLT_CTR_RE == 0:** 0_B Immediate edge propagation mode 1_B Immediate edge propagation mode **If** CLSi_TIM_CHx_CTRL **.FLT_MODE_RE == 1 &&** CLSi_TIM_CHx_ECTRL **.EFLT_CTR_RE == 1:** 0_B Reset Counter individual deglitch mode 1_B Prohibited **If** CLSi_TIM_CHx_CTRL **.FLT_MODE_RE == 0 &&** CLSi_TIM_CHx_ECTRL **.EFLT_CTR_RE == 1:** 0_B Initial value, no function in this mode 1_B Prohibited
FLT_MODE_FE	22	rw	**Filter mode for falling edge** 0_B Immediate edge propagation mode 1_B Individual deglitch mode
FLT_CTR_FE	23	rw	**Filter counter mode for falling edge** **If** CLSi_TIM_CHx_CTRL **.FLT_MODE_FE == 1 &&** CLSi_TIM_CHx_ECTRL **.EFLT_CTR_FE == 0:** 0_B Up-Down Counter individual deglitch mode 1_B Hold Counter individual deglitch mode **If** CLSi_TIM_CHx_CTRL **.FLT_MODE_FE == 0 &&** CLSi_TIM_CHx_ECTRL **.EFLT_CTR_FE == 0:** 0_B Immediate edge propagation mode 1_B Immediate edge propagation mode **If** CLSi_TIM_CHx_CTRL **.FLT_MODE_FE == 1 &&** CLSi_TIM_CHx_ECTRL **.EFLT_CTR_FE == 1:** 0_B Reset Counter individual deglitch mode 1_B Prohibited **If** CLSi_TIM_CHx_CTRL **.FLT_MODE_FE == 0 &&** CLSi_TIM_CHx_ECTRL **.EFLT_CTR_FE == 1:** 0_B Initial value, no function in this mode 1_B Prohibited

(table continues...)

(continued)

Field	Bits	Type	Description
CLK_SEL	26:24	rw	**CMU clock source select for channel** **If** CLSi_TIM_CHx_ECTRL **.ECLK_SEL == 0:** 000_B CCM[i]_CLK_RES[0:0] resolution in use 001_B CCM[i]_CLK_RES[1:1] resolution in use 010_B CCM[i]_CLK_RES[2:2] resolution in use 011_B CCM[i]_CLK_RES[3:3] resolution in use 100_B CCM[i]_CLK_RES[4:4] resolution in use 101_B CCM[i]_CLK_RES[5:5] resolution in use 110_B CCM[i]_CLK_RES[6:6] resolution in use 111_B CCM[i]_CLK_RES[7:7] resolution in use **If** CLSi_TIM_CHx_ECTRL **.ECLK_SEL == 1:** 000_B TDU_SAMPLE_EVT of TDU selected 001_B Prohibited 010_B Prohibited 011_B Prohibited 100_B Prohibited 101_B Prohibited 110_B Prohibited 111_B Prohibited
FR_ECNT_OFL	27	rw	**Extended Edge counter overflow behavior** 0_B Overflow will be signaled on CLSi_TIM_CHx_ECNT bit width = 8 1_B Overflow will be signaled on CLSi_TIM_CHx_ECNT bit width (full range)
EGPR0_SEL	28	rw	**Extension of** CLSi_TIM_CHx_CTRL**.**GPR0_SEL **bit field** Details described in CLSi_TIM_CHx_CTRL.GPR0_SEL bit field.
EGPR1_SEL	29	rw	**Extension of** CLSi_TIM_CHx_CTRL**.**GPR1_SEL **bit field** Details described in CLSi_TIM_CHx_CTRL.GPR1_SEL bit field.
TOCTRL	31:30	rw	**Timeout control** ***Note**: It has to mention that writing of CLSi_TIM_CHx_CTRL.TOCTRL= 0 will every time stop the TDU, independent of the previous state of CLSi_TIM_CHx_CTRL.TOCTRL.* 00_B Timeout feature disabled 01_B Timeout feature enabled for rising edge only 10_B Timeout feature enabled for falling edge only 11_B Timeout feature enabled for both edges
0	5	r	**Reserved** Read as 0; must be written with 0.

Table 2318 Reset values of CLSi_TIM_CHx_CTRL (i=0;x=0-7)

Reset	Reset value	Note
Kernel Reset	$0000\ 0000_H$	
TIM channel reset	$0000\ 0000_H$	

44.5.61 TIM[i] channel [x] control register (x=0-7;i=1-2)

CLSi_TIM_CHx_CTRL (i=1-2;x=0-7) Offset address: $000824_H+i*20000_H+x*80_H$

TIM[i] channel [x] control register Reset values see: Table 2319

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TOCTRL		EGPR1_SEL	EGPR0_SEL	FR_ECNT_OFL	CLK_SEL			FLT_CTR_FE	FLT_MODE_FE	FLT_CTR_RE	FLT_MODE_RE	EXT_CAP_EN	FLT_CNT_FRQ		FLT_EN
rw		rw	rw	rw	rw			rw	rw	rw	rw	rw	rw		rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECNT_RESET	ISL	DSL	CNTS_SEL	GPR1_SEL		GPR0_SEL		0	CICTRL	0	OSM	TIM_MODE			TIM_EN
rw	rw	rw	rw	rw		rw		r	rw	r	rw	rw			rwh

Field	Bits	Type	Description
TIM_EN	0	rwh	**TIM channel [x] enable** ***Note**: Enabling of the channel resets the registers CLSi_TIM_CHx_ECNT, CLSi_TIM_CHx_CNT, CLSi_TIM_CHx_GPR0, and CLSi_TIM_CHx_GPR1 to their reset values.* ***Note**: After finishing the action in one-shot mode the CLSi_TIM_CHx_CTRL.TIM_EN bit is cleared automatically. Otherwise, the bit must be cleared manually.* 0_B Channel disabled 1_B Channel enabled

(table continues...)

(continued)

Field	Bits	Type	Description
TIM_MODE	3:1	rw	**TIM channel [x] mode** ***Note**: The CLSi_TIM_CHx_CTRL.TIM_MODE register should not be changed while the **TIM** channel is enabled.* ***Note**: If the **TIM** channel is enabled and operating in TPWM or TPIM mode after the first valid edge defined by CLSi_TIM_CHx_CTRL.DSL has occurred, a reconfiguration of CLSi_TIM_CHx_CTRL.DSL, CLSi_TIM_CHx_CTRL.ISL, CLSi_TIM_CHx_CTRL.TIM_MODE will not change the channel behavior. Reading these bit fields after reconfiguration will show the newly configured settings but the initial channel behavior will not change. Only a disabling of the **TIM** channel by setting CLSi_TIM_CHx_CTRL.TIM_EN= 0 and reenabling with CLSi_TIM_CHx_CTRL.TIM_EN= 1 will change the channel operation mode.* **If reading:** 000_B PWM Measurement Mode (TPWM) 001_B Pulse Integration Mode (TPIM) 010_B Input Event Mode (TIEM) 011_B Input Prescaler Mode (TIPM) 100_B Bit Compression Mode (TBCM) 101_B Gated Periodic Sampling Mode (TGPS) 110_B Serial Shift Mode (TSSM) 111_B unused **If writing:** 000_B PWM Measurement Mode (TPWM) 001_B Pulse Integration Mode (TPIM) 010_B Input Event Mode (TIEM) 011_B Input Prescaler Mode (TIPM) 100_B Bit Compression Mode (TBCM) 101_B Gated Periodic Sampling Mode (TGPS) 110_B Serial Shift Mode (TSSM) 111_B PWM Measurement Mode (TPWM) 000_B should be used, 111_B coding might be used for new defined functionality in future devices
OSM	4	rw	**One-shot mode** ***Note**: After finishing the action in one-shot mode the CLSi_TIM_CHx_CTRL.TIM_EN bit is cleared automatically.* 0_B Continuous operation mode 1_B One-shot mode
CICTRL	6	rw	**Channel Input Control** 0_B Use signal TIM_IN[x:x] as input for channel x 1_B Use signal TIM_IN[x-1:x-1] as input for channel x (or TIM_IN[7:7] of instance [i])

(table continues...)

(continued)

Field	Bits	Type	Description
GPR0_SEL	9:8	rw	**Selection for** CLSi_TIM_CHx_GPR0 **register** **If** CLSi_TIM_CHx_CTRL **.EGPR0_SEL == 0:** 00_B Use TIM_TBU_TS0 as input 01_B Use TIM_TBU_TS1 as input 10_B Use TIM_TBU_TS2 as input 11_B Use CLSi_TIM_CHx_CNTS as input; if TGPS mode in channel 0 is selected use Filter F_OUT as input **If** CLSi_TIM_CHx_CTRL **.EGPR0_SEL == 1:** 00_B Use CLSi_TIM_CHx_ECNT as input 01_B Use CLSi_TIM_INP_VAL as input 10_B Prohibited 11_B Prohibited
GPR1_SEL	11:10	rw	**Selection for** CLSi_TIM_CHx_GPR1 **register** **If** CLSi_TIM_CHx_CTRL **.TIM_MODE !=4 &&** CLSi_TIM_CHx_CTRL **.EGPR1_SEL == 0:** 00_B Use TIM_TBU_TS0 as input 01_B Use TIM_TBU_TS1 as input 10_B Use TIM_TBU_TS2 as input 11_B Use CLSi_TIM_CHx_CNT as input **If** CLSi_TIM_CHx_CTRL **.TIM_MODE !=4 &&** CLSi_TIM_CHx_CTRL **.EGPR1_SEL == 1:** 00_B Use CLSi_TIM_CHx_ECNT as input 01_B Use CLSi_TIM_INP_VAL as input 10_B Prohibited 11_B Prohibited **If** CLSi_TIM_CHx_CTRL **.TIM_MODE == 4 &&** CLSi_TIM_CHx_CTRL **.EGPR1_SEL == 0:** 00_B Use Filter F_OUT as input 01_B Use Filter F_OUT as input 10_B Use Filter F_OUT as input 11_B Use Filter F_OUT as input **If** CLSi_TIM_CHx_CTRL **.TIM_MODE == 4 &&** CLSi_TIM_CHx_CTRL **.EGPR1_SEL == 1:** 00_B Use Filter F_OUT as input 01_B Use CLSi_TIM_INP_VAL as input 10_B Prohibited 11_B Prohibited

(table continues...)

(continued)

Field	Bits	Type	Description
CNTS_SEL	12	rw	**Selection for** CLSi_TIM_CHx_CNTS **register** ***Note:*** CLSi_TIM_CHx_CTRL**.**CNTS_SEL *in TSSM mode selects the source signal for registered or latched shift out operation.* *0 = Use* F_OUT*[x:x]* *1 = Use* TIM_IN*[x:x]* 0_B Use CLSi_TIM_CHx_CNT register as input 1_B Use TIM_TBU_TS0 as input
DSL	13	rw	**Signal level control** ***Note:*** *In* CLSi_TIM_CHx_CTRL**.**TIM_MODE*=110_B (TSSM) the bit field* CLSi_TIM_CHx_CTRL**.**DSL *defines the shift direction.* *0 = Shift left* *1 = Shift right* 0_B Measurement starts with falling edge (level low measurement) 1_B Measurement starts with rising edge (level high measurement)
ISL	14	rw	**Ignore signal level** This bit is mode dependent and will have different meanings (see details in the **TIM** Channel mode description). 0_B Use CLSi_TIM_CHx_CTRL.DSL bit for selecting active signal level (TIEM) 1_B Ignore CLSi_TIM_CHx_CTRL.DSL and treat both edges as active edge (TIEM)

(table continues...)

(continued)

Field	Bits	Type	Description
ECNT_RESET	15	rw	**Enables resetting of counter in certain modes** **If** CLSi_TIM_CHx_CTRL **.TIM_MODE == 0:** 0_B CLSi_TIM_CHx_ECNT.ECNT counter operating in wrap around mode, CLSi_TIM_CHx_CNT.CNT is reset on active input edge defined by CLSi_TIM_CHx_CTRL.DSL 1_B CLSi_TIM_CHx_ECNT.ECNT counter operating in wrap around mode, CLSi_TIM_CHx_CNT.CNT is reset on active and inactive input edge **If** CLSi_TIM_CHx_CTRL **.TIM_MODE == 5:** 0_B CLSi_TIM_CHx_ECNT.ECNT counter operating in wrap around mode 1_B CLSi_TIM_CHx_ECNT.ECNT counter is reset with periodic sampling **If** CLSi_TIM_CHx_CTRL **.TIM_MODE == 6:** 0_B Input shift value TSSM_IN[x:x]=0 is in use with CLSi_TIM_CHx_CTRL.ISL=1 1_B Input shift value TSSM_IN[x:x]=1 is in use with CLSi_TIM_CHx_CTRL.ISL=1 **If (**CLSi_TIM_CHx_CTRL **.TIM_MODE !=0 &&** CLSi_TIM_CHx_CTRL **.TIM_MODE !=5) &&** CLSi_TIM_CHx_CTRL **.TIM_MODE !=6:** 0_B CLSi_TIM_CHx_ECNT.ECNT counter operating in wrap around mode 1_B CLSi_TIM_CHx_ECNT.ECNT counter operating in wrap around mode
FLT_EN	16	rw	**Filter enable for channel [x]** ***Note**: If the filter is disabled all filter related units (including CSU) are bypassed, which means that the signal F_IN is directly routed to signal F_OUT.* 0_B Filter disabled and internal states are reset 1_B Filter enabled
FLT_CNT_FRQ	18:17	rw	**Filter counter frequency select** 00_B CCM[i]_CLK_RES[0:0] resolution in use 01_B CCM[i]_CLK_RES[1:1] resolution in use 10_B CCM[i]_CLK_RES[6:6] resolution in use 11_B CCM[i]_CLK_RES[7:7] resolution in use
EXT_CAP_EN	19	rw	**Enables external capture mode. The selected TIM mode is only sensitive to external capture pulses the input event changes are ignored.** 0_B External capture disabled 1_B External capture enabled
FLT_MODE_RE	20	rw	**Filter mode for rising edge** 0_B Immediate edge propagation mode 1_B Individual deglitch mode

(table continues...)

(continued)

Field	Bits	Type	Description
FLT_CTR_RE	21	rw	**Filter counter mode for rising edge** **If** CLSi_TIM_CHx_CTRL **.FLT_MODE_RE == 1 &&** CLSi_TIM_CHx_ECTRL **.EFLT_CTR_RE == 0:** 0_B Up-Down Counter individual deglitch mode 1_B Hold Counter individual deglitch mode **If** CLSi_TIM_CHx_CTRL **.FLT_MODE_RE == 0 &&** CLSi_TIM_CHx_ECTRL **.EFLT_CTR_RE == 0:** 0_B Immediate edge propagation mode 1_B Immediate edge propagation mode **If** CLSi_TIM_CHx_CTRL **.FLT_MODE_RE == 1 &&** CLSi_TIM_CHx_ECTRL **.EFLT_CTR_RE == 1:** 0_B Reset Counter individual deglitch mode 1_B Prohibited **If** CLSi_TIM_CHx_CTRL **.FLT_MODE_RE == 0 &&** CLSi_TIM_CHx_ECTRL **.EFLT_CTR_RE == 1:** 0_B Initial value, no function in this mode 1_B Prohibited
FLT_MODE_FE	22	rw	**Filter mode for falling edge** 0_B Immediate edge propagation mode 1_B Individual deglitch mode
FLT_CTR_FE	23	rw	**Filter counter mode for falling edge** **If** CLSi_TIM_CHx_CTRL **.FLT_MODE_FE == 1 &&** CLSi_TIM_CHx_ECTRL **.EFLT_CTR_FE == 0:** 0_B Up-Down Counter individual deglitch mode 1_B Hold Counter individual deglitch mode **If** CLSi_TIM_CHx_CTRL **.FLT_MODE_FE == 0 &&** CLSi_TIM_CHx_ECTRL **.EFLT_CTR_FE == 0:** 0_B Immediate edge propagation mode 1_B Immediate edge propagation mode **If** CLSi_TIM_CHx_CTRL **.FLT_MODE_FE == 1 &&** CLSi_TIM_CHx_ECTRL **.EFLT_CTR_FE == 1:** 0_B Reset Counter individual deglitch mode 1_B Prohibited **If** CLSi_TIM_CHx_CTRL **.FLT_MODE_FE == 0 &&** CLSi_TIM_CHx_ECTRL **.EFLT_CTR_FE == 1:** 0_B Initial value, no function in this mode 1_B Prohibited

(table continues...)

(continued)

Field	Bits	Type	Description
CLK_SEL	26:24	rw	**CMU clock source select for channel** **If** CLSi_TIM_CHx_ECTRL **.ECLK_SEL == 0:** 000_B CCM[i]_CLK_RES[0:0] resolution in use 001_B CCM[i]_CLK_RES[1:1] resolution in use 010_B CCM[i]_CLK_RES[2:2] resolution in use 011_B CCM[i]_CLK_RES[3:3] resolution in use 100_B CCM[i]_CLK_RES[4:4] resolution in use 101_B CCM[i]_CLK_RES[5:5] resolution in use 110_B CCM[i]_CLK_RES[6:6] resolution in use 111_B CCM[i]_CLK_RES[7:7] resolution in use **If** CLSi_TIM_CHx_ECTRL **.ECLK_SEL == 1:** 000_B TDU_SAMPLE_EVT of TDU selected 001_B Prohibited 010_B Prohibited 011_B Prohibited 100_B Prohibited 101_B Prohibited 110_B Prohibited 111_B Prohibited
FR_ECNT_OFL	27	rw	**Extended Edge counter overflow behavior** 0_B Overflow will be signaled on CLSi_TIM_CHx_ECNT bit width = 8 1_B Overflow will be signaled on CLSi_TIM_CHx_ECNT bit width (full range)
EGPR0_SEL	28	rw	**Extension of** CLSi_TIM_CHx_CTRL**.**GPR0_SEL **bit field** Details described in CLSi_TIM_CHx_CTRL**.**GPR0_SEL bit field.
EGPR1_SEL	29	rw	**Extension of** CLSi_TIM_CHx_CTRL**.**GPR1_SEL **bit field** Details described in CLSi_TIM_CHx_CTRL**.**GPR1_SEL bit field.
TOCTRL	31:30	rw	**Timeout control** ***Note**: It has to mention that writing of CLSi_TIM_CHx_CTRL**.**TOCTRL*= 0 will every time stop the TDU, independent of the previous state of CLSi_TIM_CHx_CTRL**.**TOCTRL.* 00_B Timeout feature disabled 01_B Timeout feature enabled for rising edge only 10_B Timeout feature enabled for falling edge only 11_B Timeout feature enabled for both edges
0	5, 7	r	**Reserved** Read as 0; must be written with 0.

Table 2319 Reset values of CLSi_TIM_CHx_CTRL (i=1-2;x=0-7)

Reset	Reset value	Note
Kernel Reset	$0000\ 0000_H$	
TIM channel reset	$0000\ 0000_H$	

44.5.62 TIM[i] channel [x] extended control register (x=0-7;i=0-2)

CLSi_TIM_CHx_ECTRL (i=0-2;x=0-7) Offset address: 000828_H+i*20000_H+x*80_H

TIM[i] channel [x] extended control register Reset values see: Table 2320

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
USE_PREV_CH_IN	ECLK_SEL	IMM_START	SWAP_CAPTURE	0		EFLT_CTR_FE	EFLT_CTR_RE	USE_LUT		0		TDU_RESYNC			
rw	rw	rw	rw	r		rw	rw	rw		r		rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	TDU_STOP			0	TDU_START			TODET_IRQ_SRC		USE_PREV_TDU_IN	0	EXT_CAP_SRC			
r	rw			r	rw			rw		rw	r	rw			

Field	Bits	Type	Description
EXT_CAP_SRC	3:0	rw	**Defines selected source for triggering the** TIM_EXT_CAPTURE **functionality.** 0_H NEWVAL_EVT of following channel selected 1_H AUX_IN selected 2_H CNTOFL_EVT of following channel selected 3_H If CLSi_TIM_CHx_CTRL.CICTRL = 1 : use signal TIM_IN[x:x] as input for channel x; if CLSi_TIM_CHx_CTRL.CICTRL = 0 : use signal TIM_IN[x-1:x-1] as input for channel x (or TIM_IN[7:7] of instance [i-1] if x is 0) 4_H ECNTOFL_EVT of following channel selected 5_H TODET_EVT of following channel selected 6_H GLITCHDET_EVT of following channel selected 7_H GPROFL_EVT of following channel selected 8_H CCM[i]_CLK_RES selected by CLSi_TIM_CHx_CTRL.CLK_SEL of following channel 9_H REDGE_DET of following channel selected A_H FEDGE_DET of following channel selected B_H Logical or of (FEDGE_DET, REDGE_DET) of following channel selected C_H TDU_SAMPLE_EVT of local TDU selected D_H TDU_WORD_EVT of local TDU selected E_H TDU_FRAME_EVT of local TDU selected **others,** Reserved
USE_PREV_TDU_IN	5	rw	**Select input data source for** TDU 0_B Use input data of local filter for TDU 1_B Use input data of previous channel (after filter unit) for TDU

(table continues...)

(continued)

Field	Bits	Type	Description
TODET_IRQ_SRC	7:6	rw	**Selection of source for** TIM_TODET_IRQ 00_B Use TDU_TIMEOUT_EVT 01_B Use TDU_WORD_EVT 10_B Use TDU_FRAME_EVT 11_B Use TDU_SAMPLE_EVT
TDU_START	10:8	rw	**Defines condition which will start the** TDU **unit.** ***Note**:* TDU_START_000_EVENT *is defined as: Each writing of* CLSi_TIM_CHx_CTRL.TOCTRL *!= 0 (independent of current* CLSi_TIM_CHx_CTRL.TOCTRL*) while* CLSi_TIM_CHx_ECTRL.TDU_START=000_B *and* TDU *is stopped (initially or stopped by* CLSi_TIM_CHx_ECTRL.TDU_STOP *event). This event will last 1 system clock-cycle.* ***Note**:* *In mode* CLSi_TIM_CHx_TDUV.SLICING=11_B *every start/restart will load the* CLSi_TIM_CHx_TDUC.TO_CNT *with value* CLSi_TIM_CHx_TDUV.TOV2*.* 000_B Start once immediate on TDU_START_000_EVENT (see Note) 001_B Start once with occurrence of first cmu_clk selected by CLSi_TIM_CHx_CTRL.CLK_SEL when measure unit is enabled by CLSi_TIM_CHx_CTRL.TIM_EN=1 010_B Start once with occurrence of first active edge selected by CLSi_TIM_CHx_CTRL.TOCTRL; restart on TDU_FRAME_EVT if TDU is stopped 011_B Start once with occurrence of first active edge selected by CLSi_TIM_CHx_CTRL.TOCTRL 100_B Start/restart with occurrence of external capture event (if TDU is stopped, restart again) 101_B Start/restart with occurrence of first cmu_clk selected by CLSi_TIM_CHx_CTRL.CLK_SEL when measure unit is enabled by CLSi_TIM_CHx_CTRL.TIM_EN=1 (if TDU is stopped, restart again) 110_B Start once with occurrence of external capture event; restart on TDU_FRAME_EVT if TDU is stopped 111_B Start/restart with occurrence of first active edge selected by CLSi_TIM_CHx_CTRL.TOCTRL (if TDU is stopped, restart again)

(table continues...)

(continued)

Field	Bits	Type	Description
TDU_STOP	14:12	rw	**Defines condition which will stop the TDU unit.** ***Note**:* TDU_TOCTRL_0_EVENT *is defined as: Each writing of* CLSi_TIM_CHx_CTRL**.**TOCTRL *= 0 (independent of current* CLSi_TIM_CHx_CTRL**.**TOCTRL*) while* TDU *is started. This event will last 1 system clock-cycle.* **If** CLSi_TIM_CHx_TDUV **.SLICING == 2 ||** CLSi_TIM_CHx_TDUV **.SLICING == 3:** 000_B Immediate stop counting of TDU on TDU_TOCTRL_0_EVENT (see Note) 001_B Stop counting of TDU on TDU_WORD_EVT or on TDU_TOCTRL_0_EVENT (see Note) 010_B Stop counting of TDU on TDU_FRAME_EVT or on TDU_TOCTRL_0_EVENT (see Note) 011_B Stop counting of TDU on TDU_TIMEOUT_EVT or on TDU_TOCTRL_0_EVENT (see Note) 100_B Stop counting of TDU on external capture event or on TDU_TOCTRL_0_EVENT (see Note) 101_B Stop counting of CLSi_TIM_CHx_TDUC.TO_CNT on TDU_WORD_EVT or on TDU_TOCTRL_0_EVENT (see Note); stop counting of CLSi_TIM_CHx_TDUC.TO_CNT1 on TDU_FRAME_EVT or on TDU_TOCTRL_0_EVENT (see Note); stop counting of CLSi_TIM_CHx_TDUC.TO_CNT2 on TDU_TOCTRL_0_EVENT (see Note); 110_B Increase CLSi_TIM_CHx_TDUC.TO_CNT only if F_OUT[x:x] = 0; stop counting of CLSi_TIM_CHx_TDUC.TO_CNT on TDU_WORD_EVT or on TDU_TOCTRL_0_EVENT (see Note); increase CLSi_TIM_CHx_TDUC.TO_CNT1 only if F_OUT[x:x] = 1; stop counting of CLSi_TIM_CHx_TDUC.TO_CNT1 on TDU_FRAME_EVT or on TDU_TOCTRL_0_EVENT (see Note); stop counting of CLSi_TIM_CHx_TDUC.TO_CNT2 on TDU_TOCTRL_0_EVENT (see Note); **If** CLSi_TIM_CHx_TDUV **.SLICING == 0 ||** CLSi_TIM_CHx_TDUV **.SLICING == 1:** 000_B Immediate stop counting of TDU on TDU_TOCTRL_0_EVENT (see Note) 001_B Stop counting of TDU on TDU_WORD_EVT or on TDU_TOCTRL_0_EVENT (see Note) 010_B Stop counting of TDU on TDU_FRAME_EVT or on TDU_TOCTRL_0_EVENT (see Note) 011_B Stop counting of TDU on TDU_TIMEOUT_EVT or on TDU_TOCTRL_0_EVENT (see Note) 100_B Stop counting of TDU on external capture event or on TDU_TOCTRL_0_EVENT (see Note)

(table continues...)

(continued)

Field	Bits	Type	Description
TDU_RESYNC	19:16	rw	**Defines condition which will resynchronize the TDU unit.** ***Note**:* TDU_START_000_EVENT *is defined as: Each writing of* CLSi_TIM_CHx_CTRL.TOCTRL *!= 0 (independent of current* CLSi_TIM_CHx_CTRL.TOCTRL*) while* CLSi_TIM_CHx_ECTRL.TDU_START=000_B *and* TDU *is stopped (initially or stopped by* CLSi_TIM_CHx_ECTRL.TDU_STOP *event). This event will last 1 system clock-cycle.* **If** CLSi_TIM_CHx_TDUV **.SLICING == 0:** 0_H Reset counter CLSi_TIM_CHx_TDUC.TO_CNT2 on each active edge selected by CLSi_TIM_CHx_CTRL.TOCTRL or TDU_TIMEOUT_EVT or on TDU_START_000_EVENT (see Note); reset counters CLSi_TIM_CHx_TDUC.TO_CNT, CLSi_TIM_CHx_TDUC.TO_CNT1 on TDU_TIMEOUT_EVT or on TDU_START_000_EVENT (see Note); reset counters CLSi_TIM_CHx_TDUC.TO_CNT, CLSi_TIM_CHx_TDUC.TO_CNT1 on each active edge selected by CLSi_TIM_CHx_CTRL.TOCTRL; 1_H Reset counters CLSi_TIM_CHx_TDUC.TO_CNT, CLSi_TIM_CHx_TDUC.TO_CNT1 on each active edge selected by CLSi_TIM_CHx_CTRL.TOCTRL; reset counter CLSi_TIM_CHx_TDUC.TO_CNT2 on each active edge selected by CLSi_TIM_CHx_CTRL.TOCTRL ; 2_H Reset counter CLSi_TIM_CHx_TDUC.TO_CNT on TDU_WORD_EVT 3_H Reset counters CLSi_TIM_CHx_TDUC.TO_CNT, CLSi_TIM_CHx_TDUC.TO_CNT1 on each active edge selected by CLSi_TIM_CHx_CTRL.TOCTRL; reset counter CLSi_TIM_CHx_TDUC.TO_CNT2 on each active edge selected by CLSi_TIM_CHx_CTRL.TOCTRL ; reset counter CLSi_TIM_CHx_TDUC.TO_CNT on TDU_WORD_EVT ; 4_H Reset counter CLSi_TIM_CHx_TDUC.TO_CNT1 on TDU_FRAME_EVT 5_H Reset counters CLSi_TIM_CHx_TDUC.TO_CNT, CLSi_TIM_CHx_TDUC.TO_CNT1 on each active edge selected by CLSi_TIM_CHx_CTRL.TOCTRL; reset counter CLSi_TIM_CHx_TDUC.TO_CNT2 on each active edge selected by CLSi_TIM_CHx_CTRL.TOCTRL ; reset counter CLSi_TIM_CHx_TDUC.TO_CNT1 on TDU_FRAME_EVT ; 6_H Reset counter CLSi_TIM_CHx_TDUC.TO_CNT on TDU_WORD_EVT ; reset counter CLSi_TIM_CHx_TDUC.TO_CNT1 on TDU_FRAME_EVT ; 7_H Reset counters CLSi_TIM_CHx_TDUC.TO_CNT, CLSi_TIM_CHx_TDUC.TO_CNT1 on each active edge selected by CLSi_TIM_CHx_CTRL.TOCTRL; reset counter CLSi_TIM_CHx_TDUC.TO_CNT2 on each active edge selected by CLSi_TIM_CHx_CTRL.TOCTRL ; reset counter CLSi_TIM_CHx_TDUC.TO_CNT on TDU_WORD_EVT ; reset counter CLSi_TIM_CHx_TDUC.TO_CNT1 on TDU_FRAME_EVT ; 8_H Reset counters CLSi_TIM_CHx_TDUC.TO_CNT, CLSi_TIM_CHx_TDUC.TO_CNT1, CLSi_TIM_CHx_TDUC.TO_CNT2 on event selected by CLSi_TIM_CHx_ECTRL.EXT_CAP_SRC

(table continues...)

(continued)

Field	Bits	Type	Description
			9_H Reset counter CLSi_TIM_CHx_TDUC.TO_CNT2 on each active edge selected by CLSi_TIM_CHx_CTRL.TOCTRL; reset counters CLSi_TIM_CHx_TDUC.TO_CNT, CLSi_TIM_CHx_TDUC.TO_CNT1 on each active edge selected by CLSi_TIM_CHx_CTRL.TOCTRL; A_H Reset counter CLSi_TIM_CHx_TDUC.TO_CNT on TDU_WORD_EVT B_H Reset counter CLSi_TIM_CHx_TDUC.TO_CNT2 on each active edge selected by CLSi_TIM_CHx_CTRL.TOCTRL; reset counters CLSi_TIM_CHx_TDUC.TO_CNT, CLSi_TIM_CHx_TDUC.TO_CNT1 on each active edge selected by CLSi_TIM_CHx_CTRL.TOCTRL; reset counter CLSi_TIM_CHx_TDUC.TO_CNT on TDU_WORD_EVT; C_H Reset counter CLSi_TIM_CHx_TDUC.TO_CNT1 on TDU_FRAME_EVT D_H Reset counter CLSi_TIM_CHx_TDUC.TO_CNT2 on each active edge selected by CLSi_TIM_CHx_CTRL.TOCTRL; reset counters CLSi_TIM_CHx_TDUC.TO_CNT, CLSi_TIM_CHx_TDUC.TO_CNT1 on each active edge selected by CLSi_TIM_CHx_CTRL.TOCTRL; reset counter CLSi_TIM_CHx_TDUC.TO_CNT1 on TDU_FRAME_EVT ; E_H Reset counter CLSi_TIM_CHx_TDUC.TO_CNT on TDU_WORD_EVT; reset counter CLSi_TIM_CHx_TDUC.TO_CNT1 on TDU_FRAME_EVT ; F_H Reset counter CLSi_TIM_CHx_TDUC.TO_CNT2 on each active edge selected by CLSi_TIM_CHx_CTRL.TOCTRL; reset counters CLSi_TIM_CHx_TDUC.TO_CNT, CLSi_TIM_CHx_TDUC.TO_CNT1 on each active edge selected by CLSi_TIM_CHx_CTRL.TOCTRL; reset counter CLSi_TIM_CHx_TDUC.TO_CNT on TDU_WORD_EVT; reset counter CLSi_TIM_CHx_TDUC.TO_CNT1 on TDU_FRAME_EVT ; **If** CLSi_TIM_CHx_TDUV **.SLICING == 1:** 0_H Reset counter CLSi_TIM_CHx_TDUC.TO_CNT2 on each active edge selected by CLSi_TIM_CHx_CTRL.TOCTRL or TDU_TIMEOUT_EVT or on TDU_START_000_EVENT (see Note); reset counters CLSi_TIM_CHx_TDUC.TO_CNT, CLSi_TIM_CHx_TDUC.TO_CNT1 on TDU_TIMEOUT_EVT or on TDU_START_000_EVENT (see Note); reset counters CLSi_TIM_CHx_TDUC.TO_CNT, CLSi_TIM_CHx_TDUC.TO_CNT1 on each active edge selected by CLSi_TIM_CHx_CTRL.TOCTRL; 1_H Reset counters CLSi_TIM_CHx_TDUC.TO_CNT, CLSi_TIM_CHx_TDUC.TO_CNT1 on each active edge selected by CLSi_TIM_CHx_CTRL.TOCTRL 2_H Reset counter CLSi_TIM_CHx_TDUC.TO_CNT on TDU_WORD_EVT 3_H Reset counters CLSi_TIM_CHx_TDUC.TO_CNT, CLSi_TIM_CHx_TDUC.TO_CNT1 on each active edge selected by CLSi_TIM_CHx_CTRL.TOCTRL; reset counter CLSi_TIM_CHx_TDUC.TO_CNT on TDU_WORD_EVT ; 4_H Reset counter CLSi_TIM_CHx_TDUC.TO_CNT1 on TDU_FRAME_EVT ; reset counter CLSi_TIM_CHx_TDUC.TO_CNT on TDU_FRAME_EVT;

(table continues...)

(continued)

Field	Bits	Type	Description
			5_H Reset counters CLSi_TIM_CHx_TDUC.TO_CNT, CLSi_TIM_CHx_TDUC.TO_CNT1 on each active edge selected by CLSi_TIM_CHx_CTRL.TOCTRL; reset counter CLSi_TIM_CHx_TDUC.TO_CNT1 on TDU_FRAME_EVT ; reset counter CLSi_TIM_CHx_TDUC.TO_CNT on TDU_FRAME_EVT; 6_H Reset counter CLSi_TIM_CHx_TDUC.TO_CNT on TDU_WORD_EVT ; reset counter CLSi_TIM_CHx_TDUC.TO_CNT1 on TDU_FRAME_EVT ; reset counter CLSi_TIM_CHx_TDUC.TO_CNT on TDU_FRAME_EVT; 7_H Reset counters CLSi_TIM_CHx_TDUC.TO_CNT, CLSi_TIM_CHx_TDUC.TO_CNT1 on each active edge selected by CLSi_TIM_CHx_CTRL.TOCTRL; reset counter CLSi_TIM_CHx_TDUC.TO_CNT on TDU_WORD_EVT ; reset counter CLSi_TIM_CHx_TDUC.TO_CNT1 on TDU_FRAME_EVT ; reset counter CLSi_TIM_CHx_TDUC.TO_CNT on TDU_FRAME_EVT; 8_H Reset counters CLSi_TIM_CHx_TDUC.TO_CNT, CLSi_TIM_CHx_TDUC.TO_CNT1, CLSi_TIM_CHx_TDUC.TO_CNT2 on event selected by CLSi_TIM_CHx_ECTRL.EXT_CAP_SRC; reset counter CLSi_TIM_CHx_TDUC.TO_CNT2 on TDU_SAMPLE_EVT ; 9_H Reset counter CLSi_TIM_CHx_TDUC.TO_CNT2 on TDU_SAMPLE_EVT ; reset counter CLSi_TIM_CHx_TDUC.TO_CNT2 on each active edge selected by CLSi_TIM_CHx_CTRL.TOCTRL; reset counters CLSi_TIM_CHx_TDUC.TO_CNT, CLSi_TIM_CHx_TDUC.TO_CNT1 on each active edge selected by CLSi_TIM_CHx_CTRL.TOCTRL; A_H Reset counter CLSi_TIM_CHx_TDUC.TO_CNT2 on TDU_SAMPLE_EVT ; reset counter CLSi_TIM_CHx_TDUC.TO_CNT on TDU_WORD_EVT; B_H Reset counter CLSi_TIM_CHx_TDUC.TO_CNT2 on TDU_SAMPLE_EVT ; reset counter CLSi_TIM_CHx_TDUC.TO_CNT2 on each active edge selected by CLSi_TIM_CHx_CTRL.TOCTRL; reset counters CLSi_TIM_CHx_TDUC.TO_CNT, CLSi_TIM_CHx_TDUC.TO_CNT1 on each active edge selected by CLSi_TIM_CHx_CTRL.TOCTRL; reset counter CLSi_TIM_CHx_TDUC.TO_CNT on TDU_WORD_EVT; C_H Reset counter CLSi_TIM_CHx_TDUC.TO_CNT2 on TDU_SAMPLE_EVT ; reset counter CLSi_TIM_CHx_TDUC.TO_CNT1 on TDU_FRAME_EVT ; reset counter CLSi_TIM_CHx_TDUC.TO_CNT on TDU_FRAME_EVT; D_H Reset counter CLSi_TIM_CHx_TDUC.TO_CNT2 on TDU_SAMPLE_EVT ; reset counter CLSi_TIM_CHx_TDUC.TO_CNT2 on each active edge selected by CLSi_TIM_CHx_CTRL.TOCTRL; reset counters CLSi_TIM_CHx_TDUC.TO_CNT, CLSi_TIM_CHx_TDUC.TO_CNT1 on each active edge selected by CLSi_TIM_CHx_CTRL.TOCTRL; reset counter CLSi_TIM_CHx_TDUC.TO_CNT1 on TDU_FRAME_EVT ; reset counter CLSi_TIM_CHx_TDUC.TO_CNT on TDU_FRAME_EVT; E_H Reset counter CLSi_TIM_CHx_TDUC.TO_CNT2 on TDU_SAMPLE_EVT ; reset counter CLSi_TIM_CHx_TDUC.TO_CNT on TDU_WORD_EVT; reset counter CLSi_TIM_CHx_TDUC.TO_CNT1 on TDU_FRAME_EVT ; reset counter CLSi_TIM_CHx_TDUC.TO_CNT on TDU_FRAME_EVT;

(table continues...)

(continued)

Field	Bits	Type	Description
			F_H Reset counter CLSi_TIM_CHx_TDUC.TO_CNT2 on TDU_SAMPLE_EVT ; reset counter CLSi_TIM_CHx_TDUC.TO_CNT2 on each active edge selected by CLSi_TIM_CHx_CTRL.TOCTRL; reset counters CLSi_TIM_CHx_TDUC.TO_CNT, CLSi_TIM_CHx_TDUC.TO_CNT1 on each active edge selected by CLSi_TIM_CHx_CTRL.TOCTRL; reset counter CLSi_TIM_CHx_TDUC.TO_CNT on TDU_WORD_EVT; reset counter CLSi_TIM_CHx_TDUC.TO_CNT1 on TDU_FRAME_EVT ; reset counter CLSi_TIM_CHx_TDUC.TO_CNT on TDU_FRAME_EVT; **If** CLSi_TIM_CHx_TDUV **.SLICING == 2:** 0_H Reset counter CLSi_TIM_CHx_TDUC.TO_CNT2 on each active edge selected by CLSi_TIM_CHx_CTRL.TOCTRL or TDU_TIMEOUT_EVT or on TDU_START_000_EVENT (see Note); reset counters CLSi_TIM_CHx_TDUC.TO_CNT, CLSi_TIM_CHx_TDUC.TO_CNT1 on TDU_TIMEOUT_EVT or on TDU_START_000_EVENT (see Note); if CLSi_TIM_CHx_CTRL.TOCTRL[0:0] = 1 then reset counter CLSi_TIM_CHx_TDUC.TO_CNT on rising input edge; if CLSi_TIM_CHx_CTRL.TOCTRL[1:1] = 1 then reset counter CLSi_TIM_CHx_TDUC.TO_CNT1 on falling input edge; 1_H If CLSi_TIM_CHx_CTRL.TOCTRL[0:0] = 1 then reset counter CLSi_TIM_CHx_TDUC.TO_CNT on rising input edge; if CLSi_TIM_CHx_CTRL.TOCTRL[1:1] = 1 then reset counter CLSi_TIM_CHx_TDUC.TO_CNT1 on falling input edge; 2_H Reset counter CLSi_TIM_CHx_TDUC.TO_CNT on TDU_WORD_EVT 3_H If CLSi_TIM_CHx_CTRL.TOCTRL[0:0] = 1 then reset counter CLSi_TIM_CHx_TDUC.TO_CNT on rising input edge; if CLSi_TIM_CHx_CTRL.TOCTRL[1:1] = 1 then reset counter CLSi_TIM_CHx_TDUC.TO_CNT1 on falling input edge; reset counter CLSi_TIM_CHx_TDUC.TO_CNT on TDU_WORD_EVT ; 4_H Reset counter CLSi_TIM_CHx_TDUC.TO_CNT1 on TDU_FRAME_EVT 5_H If CLSi_TIM_CHx_CTRL.TOCTRL[0:0] = 1 then reset counter CLSi_TIM_CHx_TDUC.TO_CNT on rising input edge; if CLSi_TIM_CHx_CTRL.TOCTRL[1:1] = 1 then reset counter CLSi_TIM_CHx_TDUC.TO_CNT1 on falling input edge; reset counter CLSi_TIM_CHx_TDUC.TO_CNT1 on TDU_FRAME_EVT ; 6_H Reset counter CLSi_TIM_CHx_TDUC.TO_CNT on TDU_WORD_EVT ; reset counter CLSi_TIM_CHx_TDUC.TO_CNT1 on TDU_FRAME_EVT ; 7_H If CLSi_TIM_CHx_CTRL.TOCTRL[0:0] = 1 then reset counter CLSi_TIM_CHx_TDUC.TO_CNT on rising input edge; if CLSi_TIM_CHx_CTRL.TOCTRL[1:1] = 1 then reset counter CLSi_TIM_CHx_TDUC.TO_CNT1 on falling input edge; reset counter CLSi_TIM_CHx_TDUC.TO_CNT on TDU_WORD_EVT ; reset counter CLSi_TIM_CHx_TDUC.TO_CNT1 on TDU_FRAME_EVT ; 8_H Reset counters CLSi_TIM_CHx_TDUC.TO_CNT, CLSi_TIM_CHx_TDUC.TO_CNT1, CLSi_TIM_CHx_TDUC.TO_CNT2 on event selected by CLSi_TIM_CHx_ECTRL.EXT_CAP_SRC; reset counter CLSi_TIM_CHx_TDUC.TO_CNT2 on TDU_SAMPLE_EVT ;

(table continues...)

(continued)

Field	Bits	Type	Description
			9_H Reset counter CLSi_TIM_CHx_TDUC.TO_CNT2 on TDU_SAMPLE_EVT ; reset counter CLSi_TIM_CHx_TDUC.TO_CNT2 on each active edge selected by CLSi_TIM_CHx_CTRL.TOCTRL; if CLSi_TIM_CHx_CTRL.TOCTRL[0:0] = 1 then reset counter CLSi_TIM_CHx_TDUC.TO_CNT on rising input edge; if CLSi_TIM_CHx_CTRL.TOCTRL[1:1] = 1 then reset counter CLSi_TIM_CHx_TDUC.TO_CNT1 on falling input edge; A_H Reset counter CLSi_TIM_CHx_TDUC.TO_CNT2 on TDU_SAMPLE_EVT ; reset counter CLSi_TIM_CHx_TDUC.TO_CNT on TDU_WORD_EVT; B_H Reset counter CLSi_TIM_CHx_TDUC.TO_CNT2 on TDU_SAMPLE_EVT ; reset counter CLSi_TIM_CHx_TDUC.TO_CNT2 on each active edge selected by CLSi_TIM_CHx_CTRL.TOCTRL; if CLSi_TIM_CHx_CTRL.TOCTRL[0:0] = 1 then reset counter CLSi_TIM_CHx_TDUC.TO_CNT on rising input edge; if CLSi_TIM_CHx_CTRL.TOCTRL[1:1] = 1 then reset counter CLSi_TIM_CHx_TDUC.TO_CNT1 on falling input edge; reset counter CLSi_TIM_CHx_TDUC.TO_CNT on TDU_WORD_EVT; C_H Reset counter CLSi_TIM_CHx_TDUC.TO_CNT2 on TDU_SAMPLE_EVT ; reset counter CLSi_TIM_CHx_TDUC.TO_CNT1 on TDU_FRAME_EVT ; D_H Reset counter CLSi_TIM_CHx_TDUC.TO_CNT2 on TDU_SAMPLE_EVT ; reset counter CLSi_TIM_CHx_TDUC.TO_CNT2 on each active edge selected by CLSi_TIM_CHx_CTRL.TOCTRL; if CLSi_TIM_CHx_CTRL.TOCTRL[0:0] = 1 then reset counter CLSi_TIM_CHx_TDUC.TO_CNT on rising input edge; if CLSi_TIM_CHx_CTRL.TOCTRL[1:1] = 1 then reset counter CLSi_TIM_CHx_TDUC.TO_CNT1 on falling input edge; reset counter CLSi_TIM_CHx_TDUC.TO_CNT1 on TDU_FRAME_EVT ; E_H Reset counter CLSi_TIM_CHx_TDUC.TO_CNT2 on TDU_SAMPLE_EVT ; reset counter CLSi_TIM_CHx_TDUC.TO_CNT on TDU_WORD_EVT; reset counter CLSi_TIM_CHx_TDUC.TO_CNT1 on TDU_FRAME_EVT ; F_H Reset counter CLSi_TIM_CHx_TDUC.TO_CNT2 on TDU_SAMPLE_EVT ; reset counter CLSi_TIM_CHx_TDUC.TO_CNT2 on each active edge selected by CLSi_TIM_CHx_CTRL.TOCTRL; if CLSi_TIM_CHx_CTRL.TOCTRL[0:0] = 1 then reset counter CLSi_TIM_CHx_TDUC.TO_CNT on rising input edge; if CLSi_TIM_CHx_CTRL.TOCTRL[1:1] = 1 then reset counter CLSi_TIM_CHx_TDUC.TO_CNT1 on falling input edge; reset counter CLSi_TIM_CHx_TDUC.TO_CNT on TDU_WORD_EVT; reset counter CLSi_TIM_CHx_TDUC.TO_CNT1 on TDU_FRAME_EVT ; **If** CLSi_TIM_CHx_TDUV **.SLICING == 3:** 0_H Load counter CLSi_TIM_CHx_TDUC.TO_CNT with CLSi_TIM_CHx_TDUV.TOV2 on each active edge selected by CLSi_TIM_CHx_CTRL.TOCTRL or TDU_WORD_EVT or on TDU_START_000_EVENT (see Note); reset counter CLSi_TIM_CHx_TDUC.TO_CNT1 on each active edge selected by CLSi_TIM_CHx_CTRL.TOCTRL or TDU_WORD_EVT or on TDU_START_000_EVENT (see Note) 1_H Load counter CLSi_TIM_CHx_TDUC.TO_CNT with CLSi_TIM_CHx_TDUV.TOV2 on each active edge selected by

(table continues...)

(continued)

Field	Bits	Type	Description
			CLSi_TIM_CHx_CTRL.TOCTRL; reset counter CLSi_TIM_CHx_TDUC.TO_CNT1 on each active edge selected by CLSi_TIM_CHx_CTRL.TOCTRL 2_H Load counter CLSi_TIM_CHx_TDUC.TO_CNT with CLSi_TIM_CHx_TDUV.TOV2 on TDU_WORD_EVT 3_H Load counter CLSi_TIM_CHx_TDUC.TO_CNT with CLSi_TIM_CHx_TDUV.TOV2 on TDU_WORD_EVT or on each active edge selected by CLSi_TIM_CHx_CTRL.TOCTRL; reset counter CLSi_TIM_CHx_TDUC.TO_CNT1 on each active edge selected by CLSi_TIM_CHx_CTRL.TOCTRL 4_H Reset counter CLSi_TIM_CHx_TDUC.TO_CNT1 on TDU_FRAME_EVT 5_H Load counter CLSi_TIM_CHx_TDUC.TO_CNT with CLSi_TIM_CHx_TDUV.TOV2 on each active edge selected by CLSi_TIM_CHx_CTRL.TOCTRL; reset counter CLSi_TIM_CHx_TDUC.TO_CNT1 on TDU_FRAME_EVT or each active edge selected by CLSi_TIM_CHx_CTRL.TOCTRL; 6_H Load counter CLSi_TIM_CHx_TDUC.TO_CNT with CLSi_TIM_CHx_TDUV.TOV2 on TDU_WORD_EVT ; reset counter CLSi_TIM_CHx_TDUC.TO_CNT1 on TDU_FRAME_EVT 7_H Load counter CLSi_TIM_CHx_TDUC.TO_CNT with CLSi_TIM_CHx_TDUV.TOV2 on TDU_WORD_EVT or on each active edge selected by CLSi_TIM_CHx_CTRL.TOCTRL; reset counter CLSi_TIM_CHx_TDUC.TO_CNT1 on TDU_FRAME_EVT or each active edge selected by CLSi_TIM_CHx_CTRL.TOCTRL; 8_H Load counter CLSi_TIM_CHx_TDUC.TO_CNT with CLSi_TIM_CHx_TDUV.TOV2 on event selected by CLSi_TIM_CHx_ECTRL.EXT_CAP_SRC; reset counter CLSi_TIM_CHx_TDUC.TO_CNT1 on event selected by CLSi_TIM_CHx_ECTRL.EXT_CAP_SRC 9_H Load counter CLSi_TIM_CHx_TDUC.TO_CNT with CLSi_TIM_CHx_TDUV.TOV2 on each active edge selected by CLSi_TIM_CHx_CTRL.TOCTRL; reset counter CLSi_TIM_CHx_TDUC.TO_CNT1 on each active edge selected by CLSi_TIM_CHx_CTRL.TOCTRL A_H Reset counter CLSi_TIM_CHx_TDUC.TO_CNT on TDU_WORD_EVT B_H Load counter CLSi_TIM_CHx_TDUC.TO_CNT with CLSi_TIM_CHx_TDUV.TOV2 on each active edge selected by CLSi_TIM_CHx_CTRL.TOCTRL else reset counter CLSi_TIM_CHx_TDUC.TO_CNT on TDU_WORD_EVT ; reset counter CLSi_TIM_CHx_TDUC.TO_CNT1 on each active edge selected by CLSi_TIM_CHx_CTRL.TOCTRL E_H Reset counter CLSi_TIM_CHx_TDUC.TO_CNT on TDU_WORD_EVT; reset counter CLSi_TIM_CHx_TDUC.TO_CNT1 on TDU_FRAME_EVT F_H Load counter CLSi_TIM_CHx_TDUC.TO_CNT with CLSi_TIM_CHx_TDUV.TOV2 on each active edge selected by CLSi_TIM_CHx_CTRL.TOCTRL else reset counter CLSi_TIM_CHx_TDUC.TO_CNT on TDU_WORD_EVT ; reset counter CLSi_TIM_CHx_TDUC.TO_CNT1 on TDU_FRAME_EVT or on each active edge selected by CLSi_TIM_CHx_CTRL.TOCTRL;

(table continues...)

(continued)

Field	Bits	Type	Description
USE_LUT	23:22	rw	**Generate filter input by lookup-table** 00_B Lookup table not in use, LUT_IN0[x:x] used as filter input 01_B Use 3-bit lookup-table with index = TIM_EXT_CAPTURE[x:x] & LUT_IN1[x:x] & LUT_IN0[x:x]. Filter input is defined by CLSi_TIM_CHx_TDUC.TO_CNT2. 10_B Use 3-bit lookup-table with index = FOUT_PREV[x:x] & LUT_IN1[x:x] & LUT_IN0[x:x]. Filter input is defined by CLSi_TIM_CHx_TDUC.TO_CNT2. 11_B Use 3-bit lookup-table with index = TSSM_OUT[x:x] & LUT_IN1[x:x] & LUT_IN0[x:x]. Filter input is defined by CLSi_TIM_CHx_TDUC.TO_CNT2.
EFLT_CTR_RE	24	rw	**Extension of bit field** CLSi_TIM_CHx_CTRL.FLT_CTR_RE Details described in CLSi_TIM_CHx_CTRL.FLT_CTR_RE bit field.
EFLT_CTR_FE	25	rw	**Extension of bit field** CLSi_TIM_CHx_CTRL.FLT_CTR_FE Details described in CLSi_TIM_CHx_CTRL.FLT_CTR_FE bit field.
SWAP_CAPTURE	28	rw	**Swap point in time of capturing** CLSi_TIM_CHx_CNTS **and** CLSi_TIM_CHx_GPR1 This bit is only applicable in TPWM and TPIM mode. Set to 0 in all other modes. 0_B Inactive edge will capture data in CLSi_TIM_CHx_CNTS; TIM_NEWVAL_IRQ event will capture data in CLSi_TIM_CHx_GPR1 1_B Swap time of capture: inactive edge will capture data in CLSi_TIM_CHx_GPR1; TIM_NEWVAL_IRQ event will capture data in CLSi_TIM_CHx_CNTS
IMM_START	29	rw	**Start the measurement immediately** This bit is only applicable in TPWM and TPIM mode. Set to 0 in all other modes. 0_B Start with first active edge the measurement 1_B Start immediately after enable (CLSi_TIM_CHx_CTRL.TIM_EN=1) the measurement
ECLK_SEL	30	rw	**Extension of bit field** CLSi_TIM_CHx_CTRL.CLK_SEL Details described in CLSi_TIM_CHx_CTRL.CLK_SEL bit field.
USE_PREV_CH_IN	31	rw	**Select input data source for TIM channel** 0_B Use input data of local filter unit for channel measurements 1_B Use input data of previous channel (after filter unit) for channel measurements
0	4, 11, 15, 21:20, 27:26	r	**Reserved** Read as 0; must be written with 0.

Table 2320 Reset values of CLSi_TIM_CHx_ECTRL (i=0-2;x=0-7)

Reset	Reset value	Note
Kernel Reset	0000 0000$_H$	
TIM channel reset	0000 0000$_H$	

44.5.63 TIM[i] channel [x] interrupt notification register (x=0-7;i=0-2)

CLSi_TIM_CHx_IRQ_NOTIFY (i=0-2;x=0-7) Offset address: 00082C$_H$+i*20000$_H$+x*80$_H$

TIM[i] channel [x] interrupt notification register Reset values see: Table 2321

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0										GLITCHDET	TODET	GPROFL	CNTOFL	ECNTOFL	NEWVAL
r										rw1ch	rw1ch	rw1ch	rw1ch	rw1ch	rw1ch

Field	Bits	Type	Description
NEWVAL	0	rw1ch	**New measurement value detected by the channel [x] ([x]:0...m-1)** ***Note**: This bit will be cleared on a CPU write-access of value '1'. A read-access leaves the bit unchanged.* **If reading:** 0_B No interrupt raised 1_B Interrupt was raised **If writing:** 0_B No action 1_B Clear interrupt
ECNTOFL	1	rw1ch	**ECNT counter overflow of channel [x] ([x]:0...m-1)** ***Note**: This bit will be cleared on a CPU write-access of value '1'. A read-access leaves the bit unchanged.* **If reading:** 0_B No interrupt raised 1_B Interrupt was raised **If writing:** 0_B No action 1_B Clear interrupt

(table continues...)

(continued)

Field	Bits	Type	Description
CNTOFL	2	rw1ch	**SMU CLSi_TIM_CHx_CNT counter overflow of channel [x], ([x]:0...m-1)** ***Note**: This bit will be cleared on a CPU write-access of value '1'. A read-access leaves the bit unchanged.* **If reading:** 0_B No interrupt raised 1_B Interrupt was raised **If writing:** 0_B No action 1_B Clear interrupt
GPROFL	3	rw1ch	**CLSi_TIM_CHx_GPR0 and CLSi_TIM_CHx_GPR1 data overflow, old data not read out before new data has arrived at input pin, ([x]:0...m-1)** ***Note**: This bit will be cleared on a CPU write-access of value '1'. A read-access leaves the bit unchanged.* **If reading:** 0_B No interrupt raised 1_B Interrupt was raised **If writing:** 0_B No action 1_B Clear interrupt
TODET	4	rw1ch	**Timeout reached for input signal of channel [x], ([x]:0...m-1)** ***Note**: This bit will be cleared on a CPU write-access of value '1'. A read-access leaves the bit unchanged.* **If reading:** 0_B No interrupt raised 1_B Interrupt was raised **If writing:** 0_B No action 1_B Clear interrupt
GLITCHDET	5	rw1ch	**Glitch detected on channel [x], ([x]:0...m-1)** ***Note**: This bit will be cleared on a CPU write-access of value '1'. A read-access leaves the bit unchanged.* **If reading:** 0_B No interrupt raised 1_B Interrupt was raised **If writing:** 0_B No action 1_B Clear interrupt

(table continues...)

(continued)

Field	Bits	Type	Description
0	31:6	r	**Reserved** Read as 0; must be written with 0.

Table 2321 Reset values of CLSi_TIM_CHx_IRQ_NOTIFY (i=0-2;x=0-7)

Reset	Reset value	Note
Kernel Reset	0000 0000$_H$	
TIM channel reset	0000 0000$_H$	

44.5.64 TIM[i] channel [x] interrupt enable register (x=0-7;i=0-2)

CLSi_TIM_CHx_IRQ_EN (i=0-2;x=0-7) Offset address: 000830$_H$+i*20000$_H$+x*80$_H$

TIM[i] channel [x] interrupt enable register Reset values see: Table 2322

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0										GLITCHDET_IRQ_EN	TODET_IRQ_EN	GPROFL_IRQ_EN	CNTOFL_IRQ_EN	ECNTOFL_IRQ_EN	NEWVAL_IRQ_EN
r										rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
NEWVAL_IRQ_EN	0	rw	TIM_NEWVAL_IRQ***[x:x]*** **interrupt enable** 0_B Disable interrupt, interrupt is not visible outside eGTM 1_B Enable interrupt, interrupt is visible outside eGTM
ECNTOFL_IRQ_EN	1	rw	TIM_ECNTOFL_IRQ***[x:x]*** **interrupt enable** 0_B Disable interrupt, interrupt is not visible outside eGTM 1_B Enable interrupt, interrupt is visible outside eGTM
CNTOFL_IRQ_EN	2	rw	TIM_CNTOFL_IRQ***[x:x]*** **interrupt enable** 0_B Disable interrupt, interrupt is not visible outside eGTM 1_B Enable interrupt, interrupt is visible outside eGTM
GPROFL_IRQ_EN	3	rw	TIM_GPROFL_IRQ***[x:x]*** **interrupt enable** 0_B Disable interrupt, interrupt is not visible outside eGTM 1_B Enable interrupt, interrupt is visible outside eGTM
TODET_IRQ_EN	4	rw	TIM_TODET_IRQ***[x:x]*** **interrupt enable** 0_B Disable interrupt, interrupt is not visible outside eGTM 1_B Enable interrupt, interrupt is visible outside eGTM

(table continues...)

(continued)

Field	Bits	Type	Description
GLITCHDET_IRQ_EN	5	rw	TIM_GLITCHDET_IRQ*[x:x]* **interrupt enable** 0_B Disable interrupt, interrupt is not visible outside eGTM 1_B Enable interrupt, interrupt is visible outside eGTM
0	31:6	r	**Reserved** Read as 0; must be written with 0.

Table 2322 Reset values of CLSi_TIM_CHx_IRQ_EN (i=0-2;x=0-7)

Reset	Reset value	Note
Kernel Reset	$0000\ 0000_H$	
TIM channel reset	$0000\ 0000_H$	

44.5.65 TIM[i] channel [x] force interrupt register (x=0-7;i=0-2)

CLSi_TIM_CHx_IRQ_FORCINT (i=0-2;x=0-7) Offset address: 000834_H+i*20000_H+x*80_H

TIM[i] channel [x] force interrupt register Reset values see: Table 2323

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0										TRG_GLITCHDET	TRG_TODET	TRG_GPROFL	TRG_CNTOFL	TRG_ECNTOFL	TRG_NEWVAL
r										rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
TRG_NEWVAL	0	rw	**Trigger the bit** CLSi_TIM_CHx_IRQ_NOTIFY**.**NEWVAL **by software** ***Note**: This bit is cleared automatically after write.* ***Note**: This bit is write-protected by bit* CLSi_ARCH_CTRL**.**RF_PROT **If reading:** 0_B No status **If writing:** 0_B No interrupt triggering 1_B Force setting of CLSi_TIM_CHx_IRQ_NOTIFY.NEWVAL

(table continues...)

(continued)

Field	Bits	Type	Description
TRG_ECNTOFL	1	rw	**Trigger the bit TIM[i]_CHx_IRQ_NOTIFY.ECNTOFL by software** ***Note**: This bit is cleared automatically after write.* ***Note**: This bit is write-protected by bit CLSi_ARCH_CTRL.RF_PROT* **If reading:** 0_B No status **If writing:** 0_B No interrupt triggering 1_B Force setting of CLSi_TIM_CHx_IRQ_NOTIFY.ECNTOFL
TRG_CNTOFL	2	rw	**Trigger the bit CLSi_TIM_CHx_IRQ_NOTIFY.CNTOFL by software** ***Note**: This bit is cleared automatically after write.* ***Note**: This bit is write-protected by bit CLSi_ARCH_CTRL.RF_PROT* **If reading:** 0_B No status **If writing:** 0_B No interrupt triggering 1_B Force setting of CLSi_TIM_CHx_IRQ_NOTIFY.CNTOFL
TRG_GPROFL	3	rw	**Trigger the bit CLSi_TIM_CHx_IRQ_NOTIFY.GPROFL by software** ***Note**: This bit is cleared automatically after write.* ***Note**: This bit is write-protected by bit CLSi_ARCH_CTRL.RF_PROT* **If reading:** 0_B No status **If writing:** 0_B No interrupt triggering 1_B Force setting of CLSi_TIM_CHx_IRQ_NOTIFY.GPROFL
TRG_TODET	4	rw	**Trigger the bit CLSi_TIM_CHx_IRQ_NOTIFY.TODET by software** ***Note**: This bit is cleared automatically after write.* ***Note**: This bit is write-protected by bit CLSi_ARCH_CTRL.RF_PROT* **If reading:** 0_B No status **If writing:** 0_B No interrupt triggering 1_B Force setting of CLSi_TIM_CHx_IRQ_NOTIFY.TODET

(table continues...)

(continued)

Field	Bits	Type	Description
TRG_GLITCHDET	5	rw	**Trigger the bit** CLSi_TIM_CHx_IRQ_NOTIFY.GLITCHDET **by software** ***Note:*** *This bit is cleared automatically after write.* ***Note:*** *This bit is write-protected by bit* CLSi_ARCH_CTRL.RF_PROT **If reading:** 0_B No status **If writing:** 0_B No interrupt triggering 1_B Force setting of CLSi_TIM_CHx_IRQ_NOTIFY.GLITCHDET
0	31:6	r	**Reserved** Read as 0; must be written with 0.

Table 2323 Reset values of CLSi_TIM_CHx_IRQ_FORCINT (i=0-2;x=0-7)

Reset	Reset value	Note
Kernel Reset	$0000\ 0000_H$	
TIM channel reset	$0000\ 0000_H$	

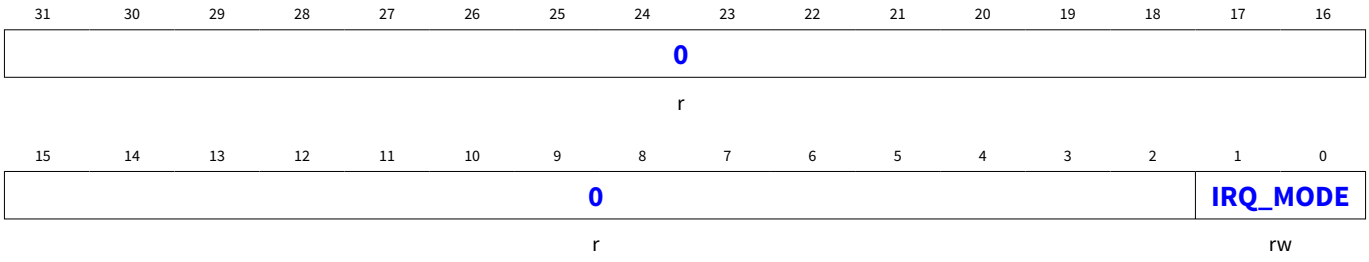
44.5.66 TIM[i] channel [x] interrupt mode configuration register (x=0-7;i=0-2)

CLSi_TIM_CHx_IRQ_MODE (i=0-2;x=0-7)

Offset address: $000838_H+i*20000_H+x*80_H$

TIM[i] channel [x] interrupt mode configuration register

Reset values see: Table 2324



Field	Bits	Type	Description
IRQ_MODE	1:0	rw	**IRQ mode selection of channel [x]** ***Note:*** *The interrupt modes are described in* **eGTM** interrupt concept. 00_B Level mode 01_B Pulse mode 10_B Pulse-Notify mode 11_B Single-Pulse mode

(table continues...)

(continued)

Field	Bits	Type	Description
0	31:2	r	**Reserved** Read as 0; must be written with 0.

Table 2324 Reset values of CLSi_TIM_CHx_IRQ_MODE (i=0-2;x=0-7)

Reset	Reset value	Note
Kernel Reset	$0000\ 0002_H$	
TIM channel reset	$0000\ 0002_H$	

44.5.67 TIM[i] channel [x] error interrupt enable register (x=0-7;i=0-2)

CLSi_TIM_CHx_EIRQ_EN (i=0-2;x=0-7)

Offset address: $00083C_H+i*20000_H+x*80_H$

TIM[i] channel [x] error interrupt enable register

Reset values see: Table 2325

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0										GLITCHDET_EIRQ_EN	TODET_EIRQ_EN	GPROFL_EIRQ_EN	CNTOFL_EIRQ_EN	ECNTOFL_EIRQ_EN	NEWVAL_EIRQ_EN
r										rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
NEWVAL_EIRQ_EN	0	rw	**TIM_NEWVAL[x]_EIRQ error interrupt enable** 0_B Disable error interrupt, error interrupt is not visible outside eGTM 1_B Enable error interrupt, error interrupt is visible outside eGTM
ECNTOFL_EIRQ_EN	1	rw	TIM_ECNTOFL_IRQ***[x:x]*** **interrupt enable** 0_B Disable error interrupt, error interrupt is not visible outside eGTM 1_B Enable error interrupt, error interrupt is visible outside eGTM
CNTOFL_EIRQ_EN	2	rw	TIM_CNTOFL_IRQ***[x:x]*** **interrupt enable** 0_B Disable error interrupt, error interrupt is not visible outside eGTM 1_B Enable error interrupt, error interrupt is visible outside eGTM
GPROFL_EIRQ_EN	3	rw	TIM_GPROFL_IRQ***[x:x]*** **interrupt enable** 0_B Disable error interrupt, error interrupt is not visible outside eGTM 1_B Enable error interrupt, error interrupt is visible outside eGTM
TODET_EIRQ_EN	4	rw	TIM_TODET_IRQ***[x:x]*** **interrupt enable** 0_B Disable error interrupt, error interrupt is not visible outside eGTM 1_B Enable error interrupt, error interrupt is visible outside eGTM

(table continues...)

(continued)

Field	Bits	Type	Description
GLITCHDET_EIRQ_EN	5	rw	TIM_GLITCHDET_IRQ*[x:x]* **interrupt enable** 0_B Disable error interrupt, error interrupt is not visible outside eGTM 1_B Enable error interrupt, error interrupt is visible outside eGTM
0	31:6	r	**Reserved** Read as 0; must be written with 0.

Table 2325 Reset values of CLSi_TIM_CHx_EIRQ_EN (i=0-2;x=0-7)

Reset	Reset value	Note
Kernel Reset	0000 0000$_H$	
TIM channel reset	0000 0000$_H$	

44.5.68 TIM[i] input value observation register (i=0-2)

CLSi_TIM_INP_VAL (i=0-2) Offset address: 000C00$_H$+i*20000$_H$

TIM[i] input value observation register Kernel Reset value: 0000 0000$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								TIM_IN7	TIM_IN6	TIM_IN5	TIM_IN4	TIM_IN3	TIM_IN2	TIM_IN1	TIM_IN0
r								rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
F_IN7	F_IN6	F_IN5	F_IN4	F_IN3	F_IN2	F_IN1	F_IN0	F_OUT7	F_OUT6	F_OUT5	F_OUT4	F_OUT3	F_OUT2	F_OUT1	F_OUT0
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
F_OUTx (x=0-7)	x	rh	**Signal channel [x] after TIM** FLT **unit** 0_B Input data is 0 1_B Input data is 1
F_INx (x=0-7)	x+8	rh	**Signal channel [x] after INPSRC selection, before TIM** FLT **unit** 0_B Input data is 0 1_B Input data is 1
TIM_INx (x=0-7)	x+16	rh	**Signal channel [x] after TIM input signal synchronization** 0_B Input data is 0 1_B Input data is 1
0	31:24	r	**Reserved** Read as 0.

44.5.69 TIM[i] AUX IN source selection register (i=0-2)

CLSi_TIM_IN_SRC (i=0-2) Offset address: 000C04$_H$+i*20000$_H$

TIM[i] AUX IN source selection register Kernel Reset value: 0000 0000$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MODE7		VAL7		MODE6		VAL6		MODE5		VAL5		MODE4		VAL4	
rwh		rwh		rwh		rwh		rwh		rwh		rwh		rwh	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MODE3		VAL3		MODE2		VAL2		MODE1		VAL1		MODE0		VAL0	
rwh		rwh		rwh		rwh		rwh		rwh		rwh		rwh	

Field	Bits	Type	Description
VALx (x=0-7)	4*x+1:4*x	rwh	**Value to be fed to channel [x]** Multithread encoding in use (CLSi_TIM_IN_SRC.VALx[1:1] defines the state of the signal) Function depends on the combination of CLSi_TIM_IN_SRC.VALx[1:1] and CLSi_TIM_IN_SRC.MODEx[1:1] see CLSi_TIM_IN_SRC.MODEx description. ***Note**: Any read-access to a CLSi_TIM_IN_SRC.VALx bit field will always result in a value 00 or 11 indicating current state. A modification of the state is only performed with the values 01 and 10. Writing the values 00 and 11 is always ignored.* **If writing:** 00_B don't care, bits will not be changed 01_B Change state to 0 10_B Change state to 1 11_B don't care, bits will not be changed **If reading:** 00_B State is 0 01_B unused 10_B unused 11_B State is 1

(table continues...)

(continued)

Field	Bits	Type	Description
MODEx (x=0-7)	4*x+3:4*x+2	rwh	**Input source to channel [x]** Multithread encoding in use (CLSi_TIM_IN_SRC.MODEx[1:1] defines the state of the signal) Function table: CLSi_TIM_IN_SRC.MODEx[1:1] = 0, CLSi_TIM_IN_SRC.VALx[1:1] = 0: The input signal defined by bit field CLSi_TIM_CHx_CTRL.CICTRL of the **TIM** channel is used as input source. CLSi_TIM_IN_SRC.MODEx[1:1] = 0, CLSi_TIM_IN_SRC.VALx[1:1] = 1: The signal AUX_IN[x:x] of the **TIM** channel is used as input source. CLSi_TIM_IN_SRC.MODEx[1:1] = 1: The state CLSi_TIM_IN_SRC.VALx[1:1] defines the input level for the **TIM** channel. ***Note**: Any read-access to a CLSi_TIM_IN_SRC.MODEx bit field will always result in a value 00 or 11 indicating current state. A modification of the state is only performed with the values 01 and 10. Writing the values 00 and 11 is always ignored.* **If writing:** 00_B don't care, bits will not be changed 01_B Change state to 0 10_B Change state to 1 11_B don't care, bits will not be changed **If reading:** 00_B State is 0 01_B unused 10_B unused 11_B State is 1

44.5.70 TIM[i] global software reset register (i=0-2)

CLSi_TIM_RST (i=0-2) Offset address: $000C08_H+i*20000_H$

TIM[i] global software reset register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RST_CH7	RST_CH6	RST_CH5	RST_CH4	RST_CH3	RST_CH2	RST_CH1	RST_CH0
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RST_CHx (x=0-7)	x	rw	**Software reset of channel [x]** ***Note**: This bit is cleared automatically after write by CPU. The channel [x] registers are set to their reset values and channel [x] operation is stopped immediately.* **If writing:** 0_B No action 1_B Reset channel [x] **If reading:** 0_B No Status 1_B Reserved
0	31:8	r	**Reserved** Read as 0; must be written with 0.

44.5.71 TOM[i] channel [x] control register (x=0-1,3-5;i=0-2)

CLSi_TOM_CHx_CTRL (i=0-2;x=0-1,3-5) Offset address: 001000_H+i*20000_H+x*40_H

TOM[i] channel [x] control register Reset values see: Table 2326

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FREEZE	0	GCM	SPEM	0	OSM	SPE_TRIG	TRIGOUT	EXTTRIGOUT	EXT_TRIG	OSM_TRIG	RST_CCU0	UDMODE		TRIG_PULSE	0
rw	r	rw	rw	r	rw	rw	rw	rw	rw	rw	rw	rw		rw	r

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CLK_SRC				SL	0			SR0_TRIG	0						
rwh				rwh	r			rw	r						

Field	Bits	Type	Description
SR0_TRIG	7	rw	CLSi_TOM_CHx_SR0 **is used to generate a trigger on output** TOM_OUT_T***[x:x]*** **if equal to** CLSi_TOM_CHx_CN0**.** ***Attention:*** ***This bit is only supported if*** CLSi_TOM_CHx_CTRL**.**RST_CCU0 ***of this channel is 0.*** **If** CLSi_TOM_CHx_CTRL **.RST_CCU0 == 0:** 0_B CLSi_TOM_CHx_SR0 is used as a shadow register for register CLSi_TOM_CHx_CM0.CM0 1_B CLSi_TOM_CHx_SR0 is not used as a shadow register for register CLSi_TOM_CHx_CM0.CM0. CLSi_TOM_CHx_SR0 is compared with CLSi_TOM_CHx_CN0 and if both are equal, a trigger pulse is generated at output TOM_OUT_T[x:x]. **If** CLSi_TOM_CHx_CTRL **.RST_CCU0 == 1:** 0_B Initial value, no function in this mode 1_B Prohibited

(table continues...)

(continued)

Field	Bits	Type	Description
SL	11	rwh	**Signal level for pulse width** ***Note**: Reset value depends on the hardware configuration chosen by silicon vendor.* ***Note**: If the output is disabled, the output TOM_OUT[x:x] is set to inverse value of CLSi_TOM_CHx_CTRL.SL.* ***Note**: If CLSi_TOM_CHx_CTRL.FREEZE=0, when the channel is disabled (TOM_ENDIS[x:x]=0) and the output is enabled (TOM_OUTEN[x:x]=1), the output TOM_OUT[x:x] will be set to !CLSi_TOM_CHx_CTRL.SL and in continuous counting up-down mode with CLSi_TOM_CHx_CTRL.RST_CCU0=1 the other output TOM_OUT_T[x:x] will be also set to ! CLSi_TOM_CHx_CTRL.SL. Thus, when the channel is re-enabled (TOM_ENDIS[x:x]=1 && TOM_OUTEN[x:x]=1) afterwards, the output TOM_OUT[x:x] is initially ! CLSi_TOM_CHx_CTRL.SL and in continuous counting up-down mode with CLSi_TOM_CHx_CTRL.RST_CCU0=1 the other output TOM_OUT_T[x:x] is also initially ! CLSi_TOM_CHx_CTRL.SL. In other PWM modes, TOM_OUT_T[x:x] is not supported.* ***Note**: If CLSi_TOM_CHx_CTRL.FREEZE=1, when the channel is disabled (TOM_ENDIS[x:x]=0) and the output is enabled(TOM_OUTEN[x:x]=1), the output TOM_OUT[x:x] will be not changed and in continuous counting up-down mode with CLSi_TOM_CHx_CTRL.RST_CCU0=1 the other output TOM_OUT_T[x:x] will also be not changed. Thus, when the channel is re-enabled (TOM_ENDIS[x:x]=1 && TOM_OUTEN[x:x]=1) afterwards, the output TOM_OUT[x:x] is initially the same state as it is before the channel is disabled and in continuous counting up-down mode with CLSi_TOM_CHx_CTRL.RST_CCU0=1 the other output TOM_OUT_T[x:x] also initially remains the same. In other PWM modes, TOM_OUT_T[x:x] is not supported.* 0_B Low signal level 1_B High signal level

(table continues...)

(continued)

Field	Bits	Type	Description
CLK_SRC	15:12	rwh	**Clock source select for channel** The register CLSi_TOM_CHx_CTRL.CLK_SRC is updated with the value of CLSi_TOM_CHx_CTRL_SR.CLK_SRC_SR together with the update of register CLSi_TOM_CHx_CM0.CM0 and CLSi_TOM_CHx_CM1.CM1. ***Note**: The clock of the channel is stopped, if "1100_B" value is configured. Restarting a channel by writing a value for a valid clock selection again is possible.* 0_H CCM[i]_FXCLK_RES[0:0] selected 1_H CCM[i]_FXCLK_RES[1:1] selected 2_H CCM[i]_FXCLK_RES[2:2] selected 3_H CCM[i]_FXCLK_RES[3:3] selected 4_H CCM[i]_FXCLK_RES[4:4] selected 5_H Prohibited 6_H Prohibited 7_H Prohibited 8_H Prohibited 9_H Prohibited A_H Prohibited B_H Prohibited C_H Functional operation stopped, clock resolution tied to zero D_H TOM_CH_TRIGOUT[x-1:x-1] selected E_H TOM_EXT_TRIGIN[x:x] selected F_H Prohibited
TRIG_PULSE	17	rw	**Trigger output pulse length of one cluster clock period** ***Attention**: **This bit may only be changed if bit CLSi_TOM_CHx_CTRL.SR0_TRIG of this channel is 1.*** **If CLSi_TOM_CHx_CTRL .SR0_TRIG == 1:** 0_B Output on TOM_OUT_T of instance i, channel x is 1 as long as CLSi_TOM_CHx_CN0.CN0=CLSi_TOM_CHx_SR0.SR0 1_B Output on TOM_OUT_T of instance i, channel x is 1 for only one cluster clock period for CLSi_TOM_CHx_CN0.CN0=CLSi_TOM_CHx_SR0.SR0 **If CLSi_TOM_CHx_CTRL .SR0_TRIG == 0:** 0_B Initial value, no function in this mode 1_B Prohibited

(table continues...)

(continued)

Field	Bits	Type	Description
UDMODE	19:18	rw	**Up-down counter mode** 00_B Up/down counter mode disabled: CLSi_TOM_CHx_CN0.CN0 counts always up, TOM[i]_CH[x].CM0.CM0, CLSi_TOM_CHx_CM1.CM1, CLSi_TOM_CHx_CTRL.SL and CLSi_TOM_CHx_CTRL.CLK_SRC are updated if CLSi_TOM_CHx_CN0.CN0 reaches the value of CLSi_TOM_CHx_CM0.CM0 -1. 01_B Up/down counter mode enabled: CLSi_TOM_CHx_CN0.CN0 counts up and down, CLSi_TOM_CHx_CM0.CM0, CLSi_TOM_CHx_CM1.CM1, CLSi_TOM_CHx_CTRL.SL and CLSi_TOM_CHx_CTRL.CLK_SRC are updated if CLSi_TOM_CHx_CN0.CN0 is 0 (i.e. the counting direction changes from down to up). 10_B Up/down counter mode enabled: CLSi_TOM_CHx_CN0.CN0 counts up and down, CLSi_TOM_CHx_CM0.CM0, CLSi_TOM_CHx_CM1.CM1, CLSi_TOM_CHx_CTRL.SL and CLSi_TOM_CHx_CTRL.CLK_SRC are updated if CLSi_TOM_CHx_CN0.CN0 reaches the value of CLSi_TOM_CHx_CM0.CM0-1 or the channel receives the trigger signal and the counting direction changes from up to down. 11_B Up/down counter mode enabled: CLSi_TOM_CHx_CN0.CN0 counts up and down, CLSi_TOM_CHx_CM0.CM0, CLSi_TOM_CHx_CM1.CM1, CLSi_TOM_CHx_CTRL.SL and CLSi_TOM_CHx_CTRL.CLK_SRC are updated if CLSi_TOM_CHx_CN0.CN0 is 0 or reaches the value of CLSi_TOM_CHx_CM0.CM0-1 or the channel receives the trigger signal and the counting direction changes from up to down.
RST_CCU0	20	rw	**Reset source of CCU0** ***Attention:*** ***This bit may only be set if bit*** CLSi_TOM_CHx_CTRL.OSM***=0 (i.e. in continuous mode)*** 0_B Reset counter CLSi_TOM_CHx_CN0.CN0 to 0 on matching comparison CLSi_TOM_CHx_CM0.CM0 1_B Reset counter CLSi_TOM_CHx_CN0.CN0 to 0 on trigger TRIG_IN[x:x] or TOM_EXT_TRIGIN[x:x]
OSM_TRIG	21	rw	**One-shot pulse generation enabled by the selected trigger signal** ***Attention:*** ***This bit may only be set if bit*** CLSi_TOM_CHx_CTRL.OSM***=1 and bit*** CLSi_TOM_CHx_CTRL.RST_CCU0***=0.*** 0_B Signal TOM_CH_TRIGIN[x:x] or TOM_EXT_TRIGIN[x:x] cannot trigger start of single pulse generation 1_B Signal TOM_CH_TRIGIN[x:x] or TOM_EXT_TRIGIN[x:x] can trigger start of single pulse generation (only if bit CLSi_TOM_CHx_CTRL.OSM = 1)

(table continues...)

(continued)

Field	Bits	Type	Description
EXT_TRIG	22	rw	**Select** TOM_EXT_TRIGIN***[x:x]*** **as trigger signal** 0_B If CLSi_TOM_CHx_CTRL.RST_CCU0=1, signal TOM_CH_TRIGIN[x:x] is selected as trigger to reset CLSi_TOM_CHx_CN0.CN0 or to start single pulse generation 1_B If CLSi_TOM_CHx_CTRL.RST_CCU0=1, signal TOM_EXT_TRIGIN[x:x] is selected as trigger to reset CLSi_TOM_CHx_CN0.CN0 or to start single pulse generation
EXTTRIGOUT	23	rw	**Select** TOM_EXT_TRIGIN***[x:x]*** **as potential output signal** TOM_CH_TRIGOUT***[x:x]*** 0_B Signal TOM_CH_TRIGIN[x:x] is selected as output on TOM_CH_TRIGOUT[x:x] (if CLSi_TOM_CHx_CTRL.TRIGOUT =0) 1_B Signal TOM_EXT_TRIGIN[x:x] after synchronization to the selected CMU clock resolution is selected as output on TOM_CH_TRIGOUT[x:x] (if CLSi_TOM_CHx_CTRL.TRIGOUT = 0)
TRIGOUT	24	rw	**Trigger output selection (output signal** TOM_CH_TRIGOUT***[x:x]***) **of module TOM channel [x]** 0_B TOM_CH_TRIGOUT[x:x] is TOM_CH_TRIGIN[x:x] or TOM_EXT_TRIGIN[x:x] 1_B TOM_CH_TRIGOUT[x:x] is TRIG_CCU0

(table continues...)

(continued)

Field	Bits	Type	Description
SPE_TRIG	25	rw	**SPE trigger to reset** CLSi_TOM_CHx_CN0 For **TOM[i]** channel x (x∈{0, 2, 6, 7, 8, 9}) this bit defines in combination with bit CLSi_TOM_CHx_CTRL.SPEM the source of output pin TOM_OUT[x:x] and if CLSi_TOM_CHx_CN0.CN0 can be reset by **TOM[i]** input signal TOM[i]_SPE_NIPD. ***Note**: If a configuration of CLSi_TOM_CHx_CTRL.SPEM = 0 and CLSi_TOM_CHx_CTRL.SPE_TRIG = 1 or CLSi_TOM_CHx_CTRL.SPEM = 1 and CLSi_TOM_CHx_CTRL.SPE_TRIG = 0 is chosen and CLSi_TOM_CHx_CTRL.RST_CCU0 = 1, the CLSi_TOM_CHx_CN0.CN0 is reset by a TOM[i]_SPE_NIPD event or a TOM_CH_TRIGIN[x:x] event if CLSi_TOM_CHx_CTRL.EXT_TRIG = 0 or a TOM_EXT_TRIGIN[x:x] event if CLSi_TOM_CHx_CTRL.EXT_TRIG = 1.* ***Note**: For **TOM[i]** channel 8 and 9 this bit defines only if CLSi_TOM_CHx_CN0.CN0 reset is defined by input signal TOM[i]_SPE_NIPD or by configuration of CLSi_TOM_CHx_CTRL.RST_CCU0. The output TOM_OUT[x:x] is not affected.* *The configuration bit CLSi_TOM_CHx_CTRL.SPEM is not supported for these channels and thus assumed to be 0.* ***Note**: If a configuration of CLSi_TOM_CHx_CTRL.SPEM = 0 and CLSi_TOM_CHx_CTRL.SPE_TRIG = 1 or CLSi_TOM_CHx_CTRL.SPEM = 1 and CLSi_TOM_CHx_CTRL.SPE_TRIG = 0 is chosen (i.e. CLSi_TOM_CHx_CN0.CN0 is reset by signal TOM[i]_SPE_NIPD), the one-shot mode in corresponding **TOM[i]** channel should also be enabled by setting bit CLSi_TOM_CHx_CTRL.OSM = 1 to generate one PWM pulse in case of trigger TOM[i]_SPE_NIPD.* ***Note**: In **SPE** module one of the trigger signals TOM[i]_CH[x]_TRIG_CCU1, TOM[i]_CH[x]_TRIG_CCU1, TOM[i]_CH[x]_TRIG_CCU1, TOM[i]_CH[x]_TRIG_CCU1, TOM[i]_CH[x]_TRIG_CCU1, or TOM[i]_CH[x]_TRIG_CCU1 of **TOM** instance i can be used to trigger the update of register CLSi_SPE_OUT_CTRL.* 0_B Initial value, no function in this mode 1_B Prohibited

(table continues...)

(continued)

Field	Bits	Type	Description
OSM	26	rw	**One-shot mode.** In this mode the counter CLSi_TOM_CHx_CN0 counts for only one period. The length of period is defined by CLSi_TOM_CHx_CM0. A write-access to the register CLSi_TOM_CHx_CN0 triggers the start of counting. 0_B One-shot mode disabled 1_B One-shot mode enabled
SPEM	28	rw	**SPE output mode enable for channel.** ***Note**:* *The bit is only implemented for **TOM[i]** instances connected to a **SPE** module and only for **TOM[i]** channels (x∈{0, 1,..., 7}). In other channels, no matter which value is written to this bit, the reading value of the bit is always 0.* ***Note**:* *For **TOM[i]** channel x (x∈{2, 6, 7, 8, 9}) this bit defines in combination with bit CLSi_TOM_CHx_CTRL.SPE_TRIG the source of output pin TOM_OUT[x:x] and if CLSi_TOM_CHx_CN0.CN0 can be reset by **TOM[i]** input signal TOM[i]_SPE_NIPD.* 0_B SPE output mode disabled: TOM_OUT[x:x] defined by TOM[i] channel [x] SOUR register 1_B SPE output mode enabled: TOM_OUT[x:x] is defined by TOM[i]_SPE_OUT[x:x]
GCM	29	rw	**Gated Counter Mode enable** ***Note**:* *The Gated Counter Mode is only available for **TOM[i]** instances connected to a **SPE** module and only for channels x (x∈{0, 1,..., 7}).* 0_B Gated Counter Mode disabled 1_B Gated Counter Mode enabled
FREEZE	31	rw	**TOM[i] Freeze Mode enable** 0_B A channel disable/enable may change internal register and output register 1_B A channel enable/disable does not change an internal or output register but stops counter CLSi_TOM_CHx_CN0.CN0
0	6:0, 10:8, 16, 27, 30	r	**Reserved** Read as 0; must be written with 0.

Table 2326 Reset values of CLSi_TOM_CHx_CTRL (i=0-2;x=0-1,3-5)

Reset	Reset value	Note
Kernel Reset	$0000\ 0800_H$	
TOM channel reset	$0000\ 0800_H$	

44.5.72 TOM[i] channel [x] control register (x=2,6-7;i=0-2)

CLSi_TOM_CHx_CTRL (i=0-2;x=2,6-7) Offset address: 001000_H+i*20000_H+x*40_H

TOM[i] channel [x] control register Reset values see: Table 2327

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FREEZE	0	GCM	SPEM	0	OSM	SPE_TRIG	TRIGOUT	EXTTRIGOUT	EXT_TRIG	OSM_TRIG	RST_CCU0	UDMODE		TRIG_PULSE	0
rw	r	rw	rw	r	rw	rw	rw	rw	rw	rw	rw	rw		rw	r

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CLK_SRC				SL	0			SR0_TRIG	0						
rwh				rwh	r			rw	r						

Field	Bits	Type	Description
SR0_TRIG	7	rw	CLSi_TOM_CHx_SR0 **is used to generate a trigger on output** TOM_OUT_T***[x:x]*** **if equal to** CLSi_TOM_CHx_CN0**.** ***Attention:*** ***This bit is only supported if*** CLSi_TOM_CHx_CTRL**.**RST_CCU0 ***of this channel is 0.*** **If** CLSi_TOM_CHx_CTRL **.RST_CCU0 == 0:** 0_B CLSi_TOM_CHx_SR0 is used as a shadow register for register CLSi_TOM_CHx_CM0.CM0 1_B CLSi_TOM_CHx_SR0 is not used as a shadow register for register CLSi_TOM_CHx_CM0.CM0. CLSi_TOM_CHx_SR0 is compared with CLSi_TOM_CHx_CN0 and if both are equal, a trigger pulse is generated at output TOM_OUT_T[x:x]. **If** CLSi_TOM_CHx_CTRL **.RST_CCU0 == 1:** 0_B Initial value, no function in this mode 1_B Prohibited

(table continues...)

(continued)

Field	Bits	Type	Description
SL	11	rwh	**Signal level for pulse width** ***Note**: Reset value depends on the hardware configuration chosen by silicon vendor.* ***Note**: If the output is disabled, the output TOM_OUT[x:x] is set to inverse value of CLSi_TOM_CHx_CTRL.SL.* ***Note**: If CLSi_TOM_CHx_CTRL.FREEZE=0, when the channel is disabled (TOM_ENDIS[x:x]=0) and the output is enabled (TOM_OUTEN[x:x]=1), the output TOM_OUT[x:x] will be set to !CLSi_TOM_CHx_CTRL.SL and in continuous counting up-down mode with CLSi_TOM_CHx_CTRL.RST_CCU0=1 the other output TOM_OUT_T[x:x] will be also set to ! CLSi_TOM_CHx_CTRL.SL. Thus, when the channel is re-enabled (TOM_ENDIS[x:x]=1 && TOM_OUTEN[x:x]=1) afterwards, the output TOM_OUT[x:x] is initially ! CLSi_TOM_CHx_CTRL.SL and in continuous counting up-down mode with CLSi_TOM_CHx_CTRL.RST_CCU0=1 the other output TOM_OUT_T[x:x] is also initially ! CLSi_TOM_CHx_CTRL.SL. In other PWM modes, TOM_OUT_T[x:x] is not supported.* ***Note**: If CLSi_TOM_CHx_CTRL.FREEZE=1, when the channel is disabled (TOM_ENDIS[x:x]=0) and the output is enabled(TOM_OUTEN[x:x]=1), the output TOM_OUT[x:x] will be not changed and in continuous counting up-down mode with CLSi_TOM_CHx_CTRL.RST_CCU0=1 the other output TOM_OUT_T[x:x] will also be not changed. Thus, when the channel is re-enabled (TOM_ENDIS[x:x]=1 && TOM_OUTEN[x:x]=1) afterwards, the output TOM_OUT[x:x] is initially the same state as it is before the channel is disabled and in continuous counting up-down mode with CLSi_TOM_CHx_CTRL.RST_CCU0=1 the other output TOM_OUT_T[x:x] also initially remains the same. In other PWM modes, TOM_OUT_T[x:x] is not supported.* 0_B Low signal level 1_B High signal level

(table continues...)

(continued)

Field	Bits	Type	Description
CLK_SRC	15:12	rwh	**Clock source select for channel** The register CLSi_TOM_CHx_CTRL.CLK_SRC is updated with the value of CLSi_TOM_CHx_CTRL_SR.CLK_SRC_SR together with the update of register CLSi_TOM_CHx_CM0.CM0 and CLSi_TOM_CHx_CM1.CM1. ***Note**: The clock of the channel is stopped, if "1100$_B$" value is configured. Restarting a channel by writing a value for a valid clock selection again is possible.* 0_H CCM[i]_FXCLK_RES[0:0] selected 1_H CCM[i]_FXCLK_RES[1:1] selected 2_H CCM[i]_FXCLK_RES[2:2] selected 3_H CCM[i]_FXCLK_RES[3:3] selected 4_H CCM[i]_FXCLK_RES[4:4] selected 5_H Prohibited 6_H Prohibited 7_H Prohibited 8_H Prohibited 9_H Prohibited A_H Prohibited B_H Prohibited C_H Functional operation stopped, clock resolution tied to zero D_H TOM_CH_TRIGOUT[x-1:x-1] selected E_H TOM_EXT_TRIGIN[x:x] selected F_H Prohibited
TRIG_PULSE	17	rw	**Trigger output pulse length of one cluster clock period** ***Attention**: **This bit may only be changed if bit CLSi_TOM_CHx_CTRL.SR0_TRIG of this channel is 1.*** **If CLSi_TOM_CHx_CTRL .SR0_TRIG == 1:** 0_B Output on TOM_OUT_T of instance i, channel x is 1 as long as CLSi_TOM_CHx_CN0.CN0=CLSi_TOM_CHx_SR0.SR0 1_B Output on TOM_OUT_T of instance i, channel x is 1 for only one cluster clock period for CLSi_TOM_CHx_CN0.CN0=CLSi_TOM_CHx_SR0.SR0 **If CLSi_TOM_CHx_CTRL .SR0_TRIG == 0:** 0_B Initial value, no function in this mode 1_B Prohibited

(table continues...)

(continued)

Field	Bits	Type	Description
UDMODE	19:18	rw	**Up-down counter mode** 00_B Up/down counter mode disabled: CLSi_TOM_CHx_CN0.CN0 counts always up, TOM[i]_CH[x].CM0.CM0, CLSi_TOM_CHx_CM1.CM1, CLSi_TOM_CHx_CTRL.SL and CLSi_TOM_CHx_CTRL.CLK_SRC are updated if CLSi_TOM_CHx_CN0.CN0 reaches the value of CLSi_TOM_CHx_CM0.CM0 -1. 01_B Up/down counter mode enabled: CLSi_TOM_CHx_CN0.CN0 counts up and down, CLSi_TOM_CHx_CM0.CM0, CLSi_TOM_CHx_CM1.CM1, CLSi_TOM_CHx_CTRL.SL and CLSi_TOM_CHx_CTRL.CLK_SRC are updated if CLSi_TOM_CHx_CN0.CN0 is 0 (i.e. the counting direction changes from down to up). 10_B Up/down counter mode enabled: CLSi_TOM_CHx_CN0.CN0 counts up and down, CLSi_TOM_CHx_CM0.CM0, CLSi_TOM_CHx_CM1.CM1, CLSi_TOM_CHx_CTRL.SL and CLSi_TOM_CHx_CTRL.CLK_SRC are updated if CLSi_TOM_CHx_CN0.CN0 reaches the value of CLSi_TOM_CHx_CM0.CM0-1 or the channel receives the trigger signal and the counting direction changes from up to down. 11_B Up/down counter mode enabled: CLSi_TOM_CHx_CN0.CN0 counts up and down, CLSi_TOM_CHx_CM0.CM0, CLSi_TOM_CHx_CM1.CM1, CLSi_TOM_CHx_CTRL.SL and CLSi_TOM_CHx_CTRL.CLK_SRC are updated if CLSi_TOM_CHx_CN0.CN0 is 0 or reaches the value of CLSi_TOM_CHx_CM0.CM0-1 or the channel receives the trigger signal and the counting direction changes from up to down.
RST_CCU0	20	rw	**Reset source of CCU0** ***Attention:*** ***This bit may only be set if bit*** CLSi_TOM_CHx_CTRL.OSM***=0 (i.e. in continuous mode)*** 0_B Reset counter CLSi_TOM_CHx_CN0.CN0 to 0 on matching comparison CLSi_TOM_CHx_CM0.CM0 1_B Reset counter CLSi_TOM_CHx_CN0.CN0 to 0 on trigger TRIG_IN[x:x] or TOM_EXT_TRIGIN[x:x]
OSM_TRIG	21	rw	**One-shot pulse generation enabled by the selected trigger signal** ***Attention:*** ***This bit may only be set if bit*** CLSi_TOM_CHx_CTRL.OSM***=1 and bit*** CLSi_TOM_CHx_CTRL.RST_CCU0***=0.*** 0_B Signal TOM_CH_TRIGIN[x:x] or TOM_EXT_TRIGIN[x:x] cannot trigger start of single pulse generation 1_B Signal TOM_CH_TRIGIN[x:x] or TOM_EXT_TRIGIN[x:x] can trigger start of single pulse generation (only if bit CLSi_TOM_CHx_CTRL.OSM = 1)

(table continues...)

(continued)

Field	Bits	Type	Description
EXT_TRIG	22	rw	**Select** TOM_EXT_TRIGIN***[x:x]*** **as trigger signal** 0_B If CLSi_TOM_CHx_CTRL.RST_CCU0=1, signal TOM_CH_TRIGIN[x:x] is selected as trigger to reset CLSi_TOM_CHx_CN0.CN0 or to start single pulse generation 1_B If CLSi_TOM_CHx_CTRL.RST_CCU0=1, signal TOM_EXT_TRIGIN[x:x] is selected as trigger to reset CLSi_TOM_CHx_CN0.CN0 or to start single pulse generation
EXTTRIGOUT	23	rw	**Select** TOM_EXT_TRIGIN***[x:x]*** **as potential output signal** TOM_CH_TRIGOUT***[x:x]*** 0_B Signal TOM_CH_TRIGIN[x:x] is selected as output on TOM_CH_TRIGOUT[x:x] (if CLSi_TOM_CHx_CTRL.TRIGOUT =0) 1_B Signal TOM_EXT_TRIGIN[x:x] after synchronization to the selected CMU clock resolution is selected as output on TOM_CH_TRIGOUT[x:x] (if CLSi_TOM_CHx_CTRL.TRIGOUT = 0)
TRIGOUT	24	rw	**Trigger output selection (output signal** TOM_CH_TRIGOUT***[x:x]*****) of module TOM channel [x]** 0_B TOM_CH_TRIGOUT[x:x] is TOM_CH_TRIGIN[x:x] or TOM_EXT_TRIGIN[x:x] 1_B TOM_CH_TRIGOUT[x:x] is TRIG_CCU0

(table continues...)

(continued)

Field	Bits	Type	Description
SPE_TRIG	25	rw	**SPE trigger to reset** CLSi_TOM_CHx_CN0 For **TOM[i]** channel x (x∈{0, 2, 6, 7, 8, 9}) this bit defines in combination with bit CLSi_TOM_CHx_CTRL.SPEM the source of output pin TOM_OUT[x:x] and if CLSi_TOM_CHx_CN0.CN0 can be reset by **TOM[i]** input signal TOM[i]_SPE_NIPD. ***Note**: If a configuration of CLSi_TOM_CHx_CTRL.SPEM = 0 and CLSi_TOM_CHx_CTRL.SPE_TRIG = 1 or CLSi_TOM_CHx_CTRL.SPEM = 1 and CLSi_TOM_CHx_CTRL.SPE_TRIG = 0 is chosen and CLSi_TOM_CHx_CTRL.RST_CCU0 = 1, the CLSi_TOM_CHx_CN0.CN0 is reset by a TOM[i]_SPE_NIPD event or a TOM_CH_TRIGIN[x:x] event if CLSi_TOM_CHx_CTRL.EXT_TRIG = 0 or a TOM_EXT_TRIGIN[x:x] event if CLSi_TOM_CHx_CTRL.EXT_TRIG = 1.* ***Note**: For **TOM[i]** channel 8 and 9 this bit defines only if CLSi_TOM_CHx_CN0.CN0 reset is defined by input signal TOM[i]_SPE_NIPD or by configuration of CLSi_TOM_CHx_CTRL.RST_CCU0. The output TOM_OUT[x:x] is not affected.* *The configuration bit CLSi_TOM_CHx_CTRL.SPEM is not supported for these channels and thus assumed to be 0.* ***Note**: If a configuration of CLSi_TOM_CHx_CTRL.SPEM = 0 and CLSi_TOM_CHx_CTRL.SPE_TRIG = 1 or CLSi_TOM_CHx_CTRL.SPEM = 1 and CLSi_TOM_CHx_CTRL.SPE_TRIG = 0 is chosen (i.e. CLSi_TOM_CHx_CN0.CN0 is reset by signal TOM[i]_SPE_NIPD), the one-shot mode in corresponding **TOM[i]** channel should also be enabled by setting bit CLSi_TOM_CHx_CTRL.OSM = 1 to generate one PWM pulse in case of trigger TOM[i]_SPE_NIPD.* ***Note**: In **SPE** module one of the trigger signals TOM[i]_CH[x]_TRIG_CCU1, TOM[i]_CH[x]_TRIG_CCU1, TOM[i]_CH[x]_TRIG_CCU1, TOM[i]_CH[x]_TRIG_CCU1, TOM[i]_CH[x]_TRIG_CCU1, or TOM[i]_CH[x]_TRIG_CCU1 of **TOM** instance i can be used to trigger the update of register CLSi_SPE_OUT_CTRL.* **If** CLSi_TOM_CHx_CTRL **.SPEM == 0:** 0_B CLSi_TOM_CHx_CN0.CN0 reset is defined by configuration of bit CLSi_TOM_CHx_CTRL.RST_CCU0 1_B CLSi_TOM_CHx_CN0.CN0 is reset by signal at the port TOM[i]_SPE_NIPD and depends on configuration bit CLSi_TOM_CHx_CTRL.RST_CCU0 **If** CLSi_TOM_CHx_CTRL **.SPEM == 1:**

(table continues...)

(continued)

Field	Bits	Type	Description
			0_B CLSi_TOM_CHx_CN0.CN0 is reset by signal at the port TOM[i]_SPE_NIPD and depends on configuration bit CLSi_TOM_CHx_CTRL.RST_CCU0 1_B CLSi_TOM_CHx_CN0.CN0 reset is defined by configuration of bit CLSi_TOM_CHx_CTRL.RST_CCU0
OSM	26	rw	**One-shot mode.** In this mode the counter CLSi_TOM_CHx_CN0 counts for only one period. The length of period is defined by CLSi_TOM_CHx_CM0. A write-access to the register CLSi_TOM_CHx_CN0 triggers the start of counting. 0_B One-shot mode disabled 1_B One-shot mode enabled
SPEM	28	rw	**SPE output mode enable for channel.** ***Note**:* *The bit is only implemented for **TOM[i]** instances connected to a **SPE** module and only for **TOM[i]** channels (x∈{0, 1,..., 7}). In other channels, no matter which value is written to this bit, the reading value of the bit is always 0.* ***Note**:* *For **TOM[i]** channel x (x∈{2, 6, 7, 8, 9}) this bit defines in combination with bit CLSi_TOM_CHx_CTRL.SPE_TRIG the source of output pin TOM_OUT[x:x] and if CLSi_TOM_CHx_CN0.CN0 can be reset by **TOM[i]** input signal TOM[i]_SPE_NIPD.* 0_B SPE output mode disabled: TOM_OUT[x:x] defined by TOM[i] channel [x] SOUR register 1_B SPE output mode enabled: TOM_OUT[x:x] is defined by TOM[i]_SPE_OUT[x:x]
GCM	29	rw	**Gated Counter Mode enable** ***Note**:* *The Gated Counter Mode is only available for **TOM[i]** instances connected to a **SPE** module and only for channels x (x∈{0, 1,..., 7}).* 0_B Gated Counter Mode disabled 1_B Gated Counter Mode enabled
FREEZE	31	rw	**TOM[i] Freeze Mode enable** 0_B A channel disable/enable may change internal register and output register 1_B A channel enable/disable does not change an internal or output register but stops counter CLSi_TOM_CHx_CN0.CN0
0	6:0, 10:8, 16, 27, 30	r	**Reserved** Read as 0; must be written with 0.

Table 2327 Reset values of CLSi_TOM_CHx_CTRL (i=0-2;x=2,6-7)

Reset	Reset value	Note
Kernel Reset	0000 0800$_H$	
TOM channel reset	0000 0800$_H$	

44.5.73 TOM[i] channel [x] control register (x=8-9;i=0-2)

CLSi_TOM_CHx_CTRL (i=0-2;x=8-9) Offset address: 001000$_H$+i*20000$_H$+x*40$_H$

TOM[i] channel [x] control register Reset values see: Table 2328

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FREEZE	**0**	**0**	**0**	**0**	**OSM**	**SPE_TRIG**	**TRIGOUT**	**EXTTRIGOUT**	**EXT_TRIG**	**OSM_TRIG**	**RST_CCU0**	**UDMODE**		**TRIG_PULSE**	**0**
rw	r	r	r	r	rw	rw	rw	rw	rw	rw	rw	rw		rw	r

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CLK_SRC				**SL**	**0**			**SR0_TRIG**	**0**						
rwh				rwh	r			rw	r						

Field	Bits	Type	Description
SR0_TRIG	7	rw	CLSi_TOM_CHx_SR0 **is used to generate a trigger on output** TOM_OUT_T***[x:x]*** **if equal to** CLSi_TOM_CHx_CN0**.** ***Attention:*** ***This bit is only supported if*** CLSi_TOM_CHx_CTRL**.**RST_CCU0 ***of this channel is 0.*** **If** CLSi_TOM_CHx_CTRL **.RST_CCU0 == 0:** 0_B CLSi_TOM_CHx_SR0 is used as a shadow register for register CLSi_TOM_CHx_CM0.CM0 1_B CLSi_TOM_CHx_SR0 is not used as a shadow register for register CLSi_TOM_CHx_CM0.CM0. CLSi_TOM_CHx_SR0 is compared with CLSi_TOM_CHx_CN0 and if both are equal, a trigger pulse is generated at output TOM_OUT_T[x:x]. **If** CLSi_TOM_CHx_CTRL **.RST_CCU0 == 1:** 0_B Initial value, no function in this mode 1_B Prohibited

(table continues...)

(continued)

Field	Bits	Type	Description
SL	11	rwh	**Signal level for pulse width** ***Note**: Reset value depends on the hardware configuration chosen by silicon vendor.* ***Note**: If the output is disabled, the output TOM_OUT[x:x] is set to inverse value of CLSi_TOM_CHx_CTRL.SL.* ***Note**: If CLSi_TOM_CHx_CTRL.FREEZE=0, when the channel is disabled (TOM_ENDIS[x:x]=0) and the output is enabled (TOM_OUTEN[x:x]=1), the output TOM_OUT[x:x] will be set to !CLSi_TOM_CHx_CTRL.SL and in continuous counting up-down mode with CLSi_TOM_CHx_CTRL.RST_CCU0=1 the other output TOM_OUT_T[x:x] will be also set to ! CLSi_TOM_CHx_CTRL.SL. Thus, when the channel is re-enabled (TOM_ENDIS[x:x]=1 && TOM_OUTEN[x:x]=1) afterwards, the output TOM_OUT[x:x] is initially ! CLSi_TOM_CHx_CTRL.SL and in continuous counting up-down mode with CLSi_TOM_CHx_CTRL.RST_CCU0=1 the other output TOM_OUT_T[x:x] is also initially ! CLSi_TOM_CHx_CTRL.SL. In other PWM modes, TOM_OUT_T[x:x] is not supported.* ***Note**: If CLSi_TOM_CHx_CTRL.FREEZE=1, when the channel is disabled (TOM_ENDIS[x:x]=0) and the output is enabled(TOM_OUTEN[x:x]=1), the output TOM_OUT[x:x] will be not changed and in continuous counting up-down mode with CLSi_TOM_CHx_CTRL.RST_CCU0=1 the other output TOM_OUT_T[x:x] will also be not changed. Thus, when the channel is re-enabled (TOM_ENDIS[x:x]=1 && TOM_OUTEN[x:x]=1) afterwards, the output TOM_OUT[x:x] is initially the same state as it is before the channel is disabled and in continuous counting up-down mode with CLSi_TOM_CHx_CTRL.RST_CCU0=1 the other output TOM_OUT_T[x:x] also initially remains the same. In other PWM modes, TOM_OUT_T[x:x] is not supported.* 0_B Low signal level 1_B High signal level

(table continues...)

(continued)

Field	Bits	Type	Description
CLK_SRC	15:12	rwh	**Clock source select for channel** The register CLSi_TOM_CHx_CTRL.CLK_SRC is updated with the value of CLSi_TOM_CHx_CTRL_SR.CLK_SRC_SR together with the update of register CLSi_TOM_CHx_CM0.CM0 and CLSi_TOM_CHx_CM1.CM1. ***Note**: The clock of the channel is stopped, if "1100$_B$" value is configured. Restarting a channel by writing a value for a valid clock selection again is possible.* 0_H CCM[i]_FXCLK_RES[0:0] selected 1_H CCM[i]_FXCLK_RES[1:1] selected 2_H CCM[i]_FXCLK_RES[2:2] selected 3_H CCM[i]_FXCLK_RES[3:3] selected 4_H CCM[i]_FXCLK_RES[4:4] selected 5_H Prohibited 6_H Prohibited 7_H Prohibited 8_H Prohibited 9_H Prohibited A_H Prohibited B_H Prohibited C_H Functional operation stopped, clock resolution tied to zero D_H TOM_CH_TRIGOUT[x-1:x-1] selected E_H TOM_EXT_TRIGIN[x:x] selected F_H Prohibited
TRIG_PULSE	17	rw	**Trigger output pulse length of one cluster clock period** ***Attention**: **This bit may only be changed if bit CLSi_TOM_CHx_CTRL.SR0_TRIG of this channel is 1.*** **If CLSi_TOM_CHx_CTRL .SR0_TRIG == 1:** 0_B Output on TOM_OUT_T of instance i, channel x is 1 as long as CLSi_TOM_CHx_CN0.CN0=CLSi_TOM_CHx_SR0.SR0 1_B Output on TOM_OUT_T of instance i, channel x is 1 for only one cluster clock period for CLSi_TOM_CHx_CN0.CN0=CLSi_TOM_CHx_SR0.SR0 **If CLSi_TOM_CHx_CTRL .SR0_TRIG == 0:** 0_B Initial value, no function in this mode 1_B Prohibited

(table continues...)

(continued)

Field	Bits	Type	Description
UDMODE	19:18	rw	**Up-down counter mode** 00_B Up/down counter mode disabled: CLSi_TOM_CHx_CN0.CN0 counts always up, TOM[i]_CH[x].CM0.CM0, CLSi_TOM_CHx_CM1.CM1, CLSi_TOM_CHx_CTRL.SL and CLSi_TOM_CHx_CTRL.CLK_SRC are updated if CLSi_TOM_CHx_CN0.CN0 reaches the value of CLSi_TOM_CHx_CM0.CM0 -1. 01_B Up/down counter mode enabled: CLSi_TOM_CHx_CN0.CN0 counts up and down, CLSi_TOM_CHx_CM0.CM0, CLSi_TOM_CHx_CM1.CM1, CLSi_TOM_CHx_CTRL.SL and CLSi_TOM_CHx_CTRL.CLK_SRC are updated if CLSi_TOM_CHx_CN0.CN0 is 0 (i.e. the counting direction changes from down to up). 10_B Up/down counter mode enabled: CLSi_TOM_CHx_CN0.CN0 counts up and down, CLSi_TOM_CHx_CM0.CM0, CLSi_TOM_CHx_CM1.CM1, CLSi_TOM_CHx_CTRL.SL and CLSi_TOM_CHx_CTRL.CLK_SRC are updated if CLSi_TOM_CHx_CN0.CN0 reaches the value of CLSi_TOM_CHx_CM0.CM0-1 or the channel receives the trigger signal and the counting direction changes from up to down. 11_B Up/down counter mode enabled: CLSi_TOM_CHx_CN0.CN0 counts up and down, CLSi_TOM_CHx_CM0.CM0, CLSi_TOM_CHx_CM1.CM1, CLSi_TOM_CHx_CTRL.SL and CLSi_TOM_CHx_CTRL.CLK_SRC are updated if CLSi_TOM_CHx_CN0.CN0 is 0 or reaches the value of CLSi_TOM_CHx_CM0.CM0-1 or the channel receives the trigger signal and the counting direction changes from up to down.
RST_CCU0	20	rw	**Reset source of CCU0** ***Attention:*** ***This bit may only be set if bit*** CLSi_TOM_CHx_CTRL.OSM***=0 (i.e. in continuous mode)*** 0_B Reset counter CLSi_TOM_CHx_CN0.CN0 to 0 on matching comparison CLSi_TOM_CHx_CM0.CM0 1_B Reset counter CLSi_TOM_CHx_CN0.CN0 to 0 on trigger TRIG_IN[x:x] or TOM_EXT_TRIGIN[x:x]
OSM_TRIG	21	rw	**One-shot pulse generation enabled by the selected trigger signal** ***Attention:*** ***This bit may only be set if bit*** CLSi_TOM_CHx_CTRL.OSM***=1 and bit*** CLSi_TOM_CHx_CTRL.RST_CCU0***=0.*** 0_B Signal TOM_CH_TRIGIN[x:x] or TOM_EXT_TRIGIN[x:x] cannot trigger start of single pulse generation 1_B Signal TOM_CH_TRIGIN[x:x] or TOM_EXT_TRIGIN[x:x] can trigger start of single pulse generation (only if bit CLSi_TOM_CHx_CTRL.OSM = 1)

(table continues...)

(continued)

Field	Bits	Type	Description
EXT_TRIG	22	rw	**Select** TOM_EXT_TRIGIN***[x:x]*** **as trigger signal** 0_B If CLSi_TOM_CHx_CTRL.RST_CCU0=1, signal TOM_CH_TRIGIN[x:x] is selected as trigger to reset CLSi_TOM_CHx_CN0.CN0 or to start single pulse generation 1_B If CLSi_TOM_CHx_CTRL.RST_CCU0=1, signal TOM_EXT_TRIGIN[x:x] is selected as trigger to reset CLSi_TOM_CHx_CN0.CN0 or to start single pulse generation
EXTTRIGOUT	23	rw	**Select** TOM_EXT_TRIGIN***[x:x]*** **as potential output signal** TOM_CH_TRIGOUT***[x:x]*** 0_B Signal TOM_CH_TRIGIN[x:x] is selected as output on TOM_CH_TRIGOUT[x:x] (if CLSi_TOM_CHx_CTRL.TRIGOUT =0) 1_B Signal TOM_EXT_TRIGIN[x:x] after synchronization to the selected CMU clock resolution is selected as output on TOM_CH_TRIGOUT[x:x] (if CLSi_TOM_CHx_CTRL.TRIGOUT = 0)
TRIGOUT	24	rw	**Trigger output selection (output signal** TOM_CH_TRIGOUT***[x:x]***) **of module TOM channel [x]** 0_B TOM_CH_TRIGOUT[x:x] is TOM_CH_TRIGIN[x:x] or TOM_EXT_TRIGIN[x:x] 1_B TOM_CH_TRIGOUT[x:x] is TRIG_CCU0

(table continues...)

(continued)

Field	Bits	Type	Description
SPE_TRIG	25	rw	**SPE trigger to reset** CLSi_TOM_CHx_CN0 For **TOM[i]** channel x (x∈{0, 2, 6, 7, 8, 9}) this bit defines in combination with bit CLSi_TOM_CHx_CTRL.SPEM the source of output pin TOM_OUT[x:x] and if CLSi_TOM_CHx_CN0.CN0 can be reset by **TOM[i]** input signal TOM[i]_SPE_NIPD. ***Note**: If a configuration of CLSi_TOM_CHx_CTRL.SPEM = 0 and CLSi_TOM_CHx_CTRL.SPE_TRIG = 1 or CLSi_TOM_CHx_CTRL.SPEM = 1 and CLSi_TOM_CHx_CTRL.SPE_TRIG = 0 is chosen and CLSi_TOM_CHx_CTRL.RST_CCU0 = 1, the CLSi_TOM_CHx_CN0.CN0 is reset by a TOM[i]_SPE_NIPD event or a TOM_CH_TRIGIN[x:x] event if CLSi_TOM_CHx_CTRL.EXT_TRIG = 0 or a TOM_EXT_TRIGIN[x:x] event if CLSi_TOM_CHx_CTRL.EXT_TRIG = 1.* ***Note**: For **TOM[i]** channel 8 and 9 this bit defines only if CLSi_TOM_CHx_CN0.CN0 reset is defined by input signal TOM[i]_SPE_NIPD or by configuration of CLSi_TOM_CHx_CTRL.RST_CCU0. The output TOM_OUT[x:x] is not affected.* *The configuration bit CLSi_TOM_CHx_CTRL.SPEM is not supported for these channels and thus assumed to be 0.* ***Note**: If a configuration of CLSi_TOM_CHx_CTRL.SPEM = 0 and CLSi_TOM_CHx_CTRL.SPE_TRIG = 1 or CLSi_TOM_CHx_CTRL.SPEM = 1 and CLSi_TOM_CHx_CTRL.SPE_TRIG = 0 is chosen (i.e. CLSi_TOM_CHx_CN0.CN0 is reset by signal TOM[i]_SPE_NIPD), the one-shot mode in corresponding **TOM[i]** channel should also be enabled by setting bit CLSi_TOM_CHx_CTRL.OSM = 1 to generate one PWM pulse in case of trigger TOM[i]_SPE_NIPD.* ***Note**: In **SPE** module one of the trigger signals TOM[i]_CH[x]_TRIG_CCU1, TOM[i]_CH[x]_TRIG_CCU1, TOM[i]_CH[x]_TRIG_CCU1, TOM[i]_CH[x]_TRIG_CCU1, TOM[i]_CH[x]_TRIG_CCU1, or TOM[i]_CH[x]_TRIG_CCU1 of **TOM** instance i can be used to trigger the update of register CLSi_SPE_OUT_CTRL.* **If** CLSi_TOM_CHx_CTRL **.SPEM == 0:** 0_B CLSi_TOM_CHx_CN0.CN0 reset is defined by configuration of bit CLSi_TOM_CHx_CTRL.RST_CCU0 1_B CLSi_TOM_CHx_CN0.CN0 is reset by signal at the port TOM[i]_SPE_NIPD and depends on configuration bit CLSi_TOM_CHx_CTRL.RST_CCU0 **If** CLSi_TOM_CHx_CTRL **.SPEM == 1:**

(table continues...)

(continued)

Field	Bits	Type	Description
			0_B CLSi_TOM_CHx_CN0.CN0 is reset by signal at the port TOM[i]_SPE_NIPD and depends on configuration bit CLSi_TOM_CHx_CTRL.RST_CCU0 1_B CLSi_TOM_CHx_CN0.CN0 reset is defined by configuration of bit CLSi_TOM_CHx_CTRL.RST_CCU0
OSM	26	rw	**One-shot mode.** In this mode the counter CLSi_TOM_CHx_CN0 counts for only one period. The length of period is defined by CLSi_TOM_CHx_CM0. A write-access to the register CLSi_TOM_CHx_CN0 triggers the start of counting. 0_B One-shot mode disabled 1_B One-shot mode enabled
FREEZE	31	rw	**TOM[i] Freeze Mode enable** 0_B A channel disable/enable may change internal register and output register 1_B A channel enable/disable does not change an internal or output register but stops counter CLSi_TOM_CHx_CN0.CN0
0	6:0, 10:8, 16, 27, 28, 29, 30	r	**Reserved** Read as 0; must be written with 0.

Table 2328 Reset values of CLSi_TOM_CHx_CTRL (i=0-2;x=8-9)

Reset	Reset value	Note
Kernel Reset	$0000\ 0800_H$	
TOM channel reset	$0000\ 0800_H$	

44.5.74 TOM[i] channel [x] control register (x=10-14;i=0-2)

CLSi_TOM_CHx_CTRL (i=0-2;x=10-14) Offset address: 001000_H+i*20000_H+x*40_H

TOM[i] channel [x] control register Reset values see: Table 2329

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FREEZE	0	0	0	0	OSM	0	TRIGOUT	EXTTRIGOUT	EXT_TRIG	OSM_TRIG	RST_CCU0	UDMODE		TRIG_PULSE	0
rw	r	r	r	r	rw	r	rw	rw	rw	rw	rw	rw		rw	r

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CLK_SRC				SL	0			SR0_TRIG	0						
rwh				rwh	r			rw	r						

Field	Bits	Type	Description
SR0_TRIG	7	rw	CLSi_TOM_CHx_SR0 **is used to generate a trigger on output** TOM_OUT_T***[x:x]*** **if equal to** CLSi_TOM_CHx_CN0**.** ***Attention:*** ***This bit is only supported if*** CLSi_TOM_CHx_CTRL**.**RST_CCU0 ***of this channel is 0.*** **If** CLSi_TOM_CHx_CTRL **.RST_CCU0 == 0:** 0_B CLSi_TOM_CHx_SR0 is used as a shadow register for register CLSi_TOM_CHx_CM0.CM0 1_B CLSi_TOM_CHx_SR0 is not used as a shadow register for register CLSi_TOM_CHx_CM0.CM0. CLSi_TOM_CHx_SR0 is compared with CLSi_TOM_CHx_CN0 and if both are equal, a trigger pulse is generated at output TOM_OUT_T[x:x]. **If** CLSi_TOM_CHx_CTRL **.RST_CCU0 == 1:** 0_B Initial value, no function in this mode 1_B Prohibited

(table continues...)

(continued)

Field	Bits	Type	Description
SL	11	rwh	**Signal level for pulse width** ***Note**: Reset value depends on the hardware configuration chosen by silicon vendor.* ***Note**: If the output is disabled, the output TOM_OUT[x:x] is set to inverse value of CLSi_TOM_CHx_CTRL.SL.* ***Note**: If CLSi_TOM_CHx_CTRL.FREEZE=0, when the channel is disabled (TOM_ENDIS[x:x]=0) and the output is enabled (TOM_OUTEN[x:x]=1), the output TOM_OUT[x:x] will be set to !CLSi_TOM_CHx_CTRL.SL and in continuous counting up-down mode with CLSi_TOM_CHx_CTRL.RST_CCU0=1 the other output TOM_OUT_T[x:x] will be also set to ! CLSi_TOM_CHx_CTRL.SL. Thus, when the channel is re-enabled (TOM_ENDIS[x:x]=1 && TOM_OUTEN[x:x]=1) afterwards, the output TOM_OUT[x:x] is initially ! CLSi_TOM_CHx_CTRL.SL and in continuous counting up-down mode with CLSi_TOM_CHx_CTRL.RST_CCU0=1 the other output TOM_OUT_T[x:x] is also initially ! CLSi_TOM_CHx_CTRL.SL. In other PWM modes, TOM_OUT_T[x:x] is not supported.* ***Note**: If CLSi_TOM_CHx_CTRL.FREEZE=1, when the channel is disabled (TOM_ENDIS[x:x]=0) and the output is enabled(TOM_OUTEN[x:x]=1), the output TOM_OUT[x:x] will be not changed and in continuous counting up-down mode with CLSi_TOM_CHx_CTRL.RST_CCU0=1 the other output TOM_OUT_T[x:x] will also be not changed. Thus, when the channel is re-enabled (TOM_ENDIS[x:x]=1 && TOM_OUTEN[x:x]=1) afterwards, the output TOM_OUT[x:x] is initially the same state as it is before the channel is disabled and in continuous counting up-down mode with CLSi_TOM_CHx_CTRL.RST_CCU0=1 the other output TOM_OUT_T[x:x] also initially remains the same. In other PWM modes, TOM_OUT_T[x:x] is not supported.* 0_B Low signal level 1_B High signal level

(table continues...)

(continued)

Field	Bits	Type	Description
CLK_SRC	15:12	rwh	**Clock source select for channel** The register CLSi_TOM_CHx_CTRL.CLK_SRC is updated with the value of CLSi_TOM_CHx_CTRL_SR.CLK_SRC_SR together with the update of register CLSi_TOM_CHx_CM0.CM0 and CLSi_TOM_CHx_CM1.CM1. ***Note**: The clock of the channel is stopped, if "1100_B" value is configured. Restarting a channel by writing a value for a valid clock selection again is possible.* 0_H CCM[i]_FXCLK_RES[0:0] selected 1_H CCM[i]_FXCLK_RES[1:1] selected 2_H CCM[i]_FXCLK_RES[2:2] selected 3_H CCM[i]_FXCLK_RES[3:3] selected 4_H CCM[i]_FXCLK_RES[4:4] selected 5_H Prohibited 6_H Prohibited 7_H Prohibited 8_H Prohibited 9_H Prohibited A_H Prohibited B_H Prohibited C_H Functional operation stopped, clock resolution tied to zero D_H TOM_CH_TRIGOUT[x-1:x-1] selected E_H TOM_EXT_TRIGIN[x:x] selected F_H Prohibited
TRIG_PULSE	17	rw	**Trigger output pulse length of one cluster clock period** ***Attention**: **This bit may only be changed if bit CLSi_TOM_CHx_CTRL.SR0_TRIG of this channel is 1.*** **If CLSi_TOM_CHx_CTRL .SR0_TRIG == 1:** 0_B Output on TOM_OUT_T of instance i, channel x is 1 as long as CLSi_TOM_CHx_CN0.CN0=CLSi_TOM_CHx_SR0.SR0 1_B Output on TOM_OUT_T of instance i, channel x is 1 for only one cluster clock period for CLSi_TOM_CHx_CN0.CN0=CLSi_TOM_CHx_SR0.SR0 **If CLSi_TOM_CHx_CTRL .SR0_TRIG == 0:** 0_B Initial value, no function in this mode 1_B Prohibited

(table continues...)

(continued)

Field	Bits	Type	Description
UDMODE	19:18	rw	**Up-down counter mode** 00_B Up/down counter mode disabled: CLSi_TOM_CHx_CN0.CN0 counts always up, TOM[i]_CH[x].CM0.CM0, CLSi_TOM_CHx_CM1.CM1, CLSi_TOM_CHx_CTRL.SL and CLSi_TOM_CHx_CTRL.CLK_SRC are updated if CLSi_TOM_CHx_CN0.CN0 reaches the value of CLSi_TOM_CHx_CM0.CM0 -1. 01_B Up/down counter mode enabled: CLSi_TOM_CHx_CN0.CN0 counts up and down, CLSi_TOM_CHx_CM0.CM0, CLSi_TOM_CHx_CM1.CM1, CLSi_TOM_CHx_CTRL.SL and CLSi_TOM_CHx_CTRL.CLK_SRC are updated if CLSi_TOM_CHx_CN0.CN0 is 0 (i.e. the counting direction changes from down to up). 10_B Up/down counter mode enabled: CLSi_TOM_CHx_CN0.CN0 counts up and down, CLSi_TOM_CHx_CM0.CM0, CLSi_TOM_CHx_CM1.CM1, CLSi_TOM_CHx_CTRL.SL and CLSi_TOM_CHx_CTRL.CLK_SRC are updated if CLSi_TOM_CHx_CN0.CN0 reaches the value of CLSi_TOM_CHx_CM0.CM0-1 or the channel receives the trigger signal and the counting direction changes from up to down. 11_B Up/down counter mode enabled: CLSi_TOM_CHx_CN0.CN0 counts up and down, CLSi_TOM_CHx_CM0.CM0, CLSi_TOM_CHx_CM1.CM1, CLSi_TOM_CHx_CTRL.SL and CLSi_TOM_CHx_CTRL.CLK_SRC are updated if CLSi_TOM_CHx_CN0.CN0 is 0 or reaches the value of CLSi_TOM_CHx_CM0.CM0-1 or the channel receives the trigger signal and the counting direction changes from up to down.
RST_CCU0	20	rw	**Reset source of CCU0** ***Attention:*** ***This bit may only be set if bit*** CLSi_TOM_CHx_CTRL.OSM***=0 (i.e. in continuous mode)*** 0_B Reset counter CLSi_TOM_CHx_CN0.CN0 to 0 on matching comparison CLSi_TOM_CHx_CM0.CM0 1_B Reset counter CLSi_TOM_CHx_CN0.CN0 to 0 on trigger TRIG_IN[x:x] or TOM_EXT_TRIGIN[x:x]
OSM_TRIG	21	rw	**One-shot pulse generation enabled by the selected trigger signal** ***Attention:*** ***This bit may only be set if bit*** CLSi_TOM_CHx_CTRL.OSM***=1 and bit*** CLSi_TOM_CHx_CTRL.RST_CCU0***=0.*** 0_B Signal TOM_CH_TRIGIN[x:x] or TOM_EXT_TRIGIN[x:x] cannot trigger start of single pulse generation 1_B Signal TOM_CH_TRIGIN[x:x] or TOM_EXT_TRIGIN[x:x] can trigger start of single pulse generation (only if bit CLSi_TOM_CHx_CTRL.OSM = 1)

(table continues...)

(continued)

Field	Bits	Type	Description
EXT_TRIG	22	rw	**Select** TOM_EXT_TRIGIN***[x:x]*** **as trigger signal** 0_B If CLSi_TOM_CHx_CTRL.RST_CCU0=1, signal TOM_CH_TRIGIN[x:x] is selected as trigger to reset CLSi_TOM_CHx_CN0.CN0 or to start single pulse generation 1_B If CLSi_TOM_CHx_CTRL.RST_CCU0=1, signal TOM_EXT_TRIGIN[x:x] is selected as trigger to reset CLSi_TOM_CHx_CN0.CN0 or to start single pulse generation
EXTTRIGOUT	23	rw	**Select** TOM_EXT_TRIGIN***[x:x]*** **as potential output signal** TOM_CH_TRIGOUT***[x:x]*** 0_B Signal TOM_CH_TRIGIN[x:x] is selected as output on TOM_CH_TRIGOUT[x:x] (if CLSi_TOM_CHx_CTRL.TRIGOUT =0) 1_B Signal TOM_EXT_TRIGIN[x:x] after synchronization to the selected CMU clock resolution is selected as output on TOM_CH_TRIGOUT[x:x] (if CLSi_TOM_CHx_CTRL.TRIGOUT = 0)
TRIGOUT	24	rw	**Trigger output selection (output signal** TOM_CH_TRIGOUT***[x:x]***) **of module TOM channel [x]** 0_B TOM_CH_TRIGOUT[x:x] is TOM_CH_TRIGIN[x:x] or TOM_EXT_TRIGIN[x:x] 1_B TOM_CH_TRIGOUT[x:x] is TRIG_CCU0
OSM	26	rw	**One-shot mode.** In this mode the counter CLSi_TOM_CHx_CN0 counts for only one period. The length of period is defined by CLSi_TOM_CHx_CM0. A write-access to the register CLSi_TOM_CHx_CN0 triggers the start of counting. 0_B One-shot mode disabled 1_B One-shot mode enabled
FREEZE	31	rw	**TOM[i] Freeze Mode enable** 0_B A channel disable/enable may change internal register and output register 1_B A channel enable/disable does not change an internal or output register but stops counter CLSi_TOM_CHx_CN0.CN0
0	6:0, 10:8, 16, 25, 27, 28, 29, 30	r	**Reserved** Read as 0; must be written with 0.

Table 2329 Reset values of CLSi_TOM_CHx_CTRL (i=0-2;x=10-14)

Reset	Reset value	Note
Kernel Reset	$0000\ 0800_H$	
TOM channel reset	$0000\ 0800_H$	

44.5.75 TOM[i] channel [x] control register (x=15;i=0-2)

CLSi_TOM_CHx_CTRL (i=0-2;x=15) Offset address: 001000_H+i*20000_H+x*40_H

TOM[i] channel [x] control register Reset values see: Table 2330

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FREEZE	0	0	0	BITREV	OSM	0	TRIGOUT	EXTTRIGOUT	EXT_TRIG	OSM_TRIG	RST_CCU0	UDMODE		TRIG_PULSE	0
rw	r	r	r	rw	rw	r	rw	rw	rw	rw	rw	rw		rw	r

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CLK_SRC				SL	0			SR0_TRIG	0						
rwh				rwh	r			rw	r						

Field	Bits	Type	Description
SR0_TRIG	7	rw	CLSi_TOM_CHx_SR0 **is used to generate a trigger on output** TOM_OUT_T***[x:x]*** **if equal to** CLSi_TOM_CHx_CN0**.** ***Attention:*** ***This bit is only supported if*** CLSi_TOM_CHx_CTRL**.**RST_CCU0 ***of this channel is 0.*** **If** CLSi_TOM_CHx_CTRL **.RST_CCU0 == 0:** 0_B CLSi_TOM_CHx_SR0 is used as a shadow register for register CLSi_TOM_CHx_CM0.CM0 1_B CLSi_TOM_CHx_SR0 is not used as a shadow register for register CLSi_TOM_CHx_CM0.CM0. CLSi_TOM_CHx_SR0 is compared with CLSi_TOM_CHx_CN0 and if both are equal, a trigger pulse is generated at output TOM_OUT_T[x:x]. **If** CLSi_TOM_CHx_CTRL **.RST_CCU0 == 1:** 0_B Initial value, no function in this mode 1_B Prohibited

(table continues...)

(continued)

Field	Bits	Type	Description
SL	11	rwh	**Signal level for pulse width** ***Note**: Reset value depends on the hardware configuration chosen by silicon vendor.* ***Note**: If the output is disabled, the output TOM_OUT[x:x] is set to inverse value of CLSi_TOM_CHx_CTRL.SL.* ***Note**: If CLSi_TOM_CHx_CTRL.FREEZE=0, when the channel is disabled (TOM_ENDIS[x:x]=0) and the output is enabled (TOM_OUTEN[x:x]=1), the output TOM_OUT[x:x] will be set to !CLSi_TOM_CHx_CTRL.SL and in continuous counting up-down mode with CLSi_TOM_CHx_CTRL.RST_CCU0=1 the other output TOM_OUT_T[x:x] will be also set to ! CLSi_TOM_CHx_CTRL.SL. Thus, when the channel is re-enabled (TOM_ENDIS[x:x]=1 && TOM_OUTEN[x:x]=1) afterwards, the output TOM_OUT[x:x] is initially ! CLSi_TOM_CHx_CTRL.SL and in continuous counting up-down mode with CLSi_TOM_CHx_CTRL.RST_CCU0=1 the other output TOM_OUT_T[x:x] is also initially ! CLSi_TOM_CHx_CTRL.SL. In other PWM modes, TOM_OUT_T[x:x] is not supported.* ***Note**: If CLSi_TOM_CHx_CTRL.FREEZE=1, when the channel is disabled (TOM_ENDIS[x:x]=0) and the output is enabled(TOM_OUTEN[x:x]=1), the output TOM_OUT[x:x] will be not changed and in continuous counting up-down mode with CLSi_TOM_CHx_CTRL.RST_CCU0=1 the other output TOM_OUT_T[x:x] will also be not changed. Thus, when the channel is re-enabled (TOM_ENDIS[x:x]=1 && TOM_OUTEN[x:x]=1) afterwards, the output TOM_OUT[x:x] is initially the same state as it is before the channel is disabled and in continuous counting up-down mode with CLSi_TOM_CHx_CTRL.RST_CCU0=1 the other output TOM_OUT_T[x:x] also initially remains the same. In other PWM modes, TOM_OUT_T[x:x] is not supported.* 0_B Low signal level 1_B High signal level

(table continues...)

(continued)

Field	Bits	Type	Description
CLK_SRC	15:12	rwh	**Clock source select for channel** The register CLSi_TOM_CHx_CTRL.CLK_SRC is updated with the value of CLSi_TOM_CHx_CTRL_SR.CLK_SRC_SR together with the update of register CLSi_TOM_CHx_CM0.CM0 and CLSi_TOM_CHx_CM1.CM1. ***Note**: The clock of the channel is stopped, if "1100_B" value is configured. Restarting a channel by writing a value for a valid clock selection again is possible.* 0_H CCM[i]_FXCLK_RES[0:0] selected 1_H CCM[i]_FXCLK_RES[1:1] selected 2_H CCM[i]_FXCLK_RES[2:2] selected 3_H CCM[i]_FXCLK_RES[3:3] selected 4_H CCM[i]_FXCLK_RES[4:4] selected 5_H Prohibited 6_H Prohibited 7_H Prohibited 8_H Prohibited 9_H Prohibited A_H Prohibited B_H Prohibited C_H Functional operation stopped, clock resolution tied to zero D_H TOM_CH_TRIGOUT[x-1:x-1] selected E_H TOM_EXT_TRIGIN[x:x] selected F_H Prohibited
TRIG_PULSE	17	rw	**Trigger output pulse length of one cluster clock period** ***Attention**: **This bit may only be changed if bit CLSi_TOM_CHx_CTRL.SR0_TRIG of this channel is 1.*** **If CLSi_TOM_CHx_CTRL .SR0_TRIG == 1:** 0_B Output on TOM_OUT_T of instance i, channel x is 1 as long as CLSi_TOM_CHx_CN0.CN0=CLSi_TOM_CHx_SR0.SR0 1_B Output on TOM_OUT_T of instance i, channel x is 1 for only one cluster clock period for CLSi_TOM_CHx_CN0.CN0=CLSi_TOM_CHx_SR0.SR0 **If CLSi_TOM_CHx_CTRL .SR0_TRIG == 0:** 0_B Initial value, no function in this mode 1_B Prohibited

(table continues...)

(continued)

Field	Bits	Type	Description
UDMODE	19:18	rw	**Up-down counter mode** 00_B Up/down counter mode disabled: CLSi_TOM_CHx_CN0.CN0 counts always up, TOM[i]_CH[x].CM0.CM0, CLSi_TOM_CHx_CM1.CM1, CLSi_TOM_CHx_CTRL.SL and CLSi_TOM_CHx_CTRL.CLK_SRC are updated if CLSi_TOM_CHx_CN0.CN0 reaches the value of CLSi_TOM_CHx_CM0.CM0 -1. 01_B Up/down counter mode enabled: CLSi_TOM_CHx_CN0.CN0 counts up and down, CLSi_TOM_CHx_CM0.CM0, CLSi_TOM_CHx_CM1.CM1, CLSi_TOM_CHx_CTRL.SL and CLSi_TOM_CHx_CTRL.CLK_SRC are updated if CLSi_TOM_CHx_CN0.CN0 is 0 (i.e. the counting direction changes from down to up). 10_B Up/down counter mode enabled: CLSi_TOM_CHx_CN0.CN0 counts up and down, CLSi_TOM_CHx_CM0.CM0, CLSi_TOM_CHx_CM1.CM1, CLSi_TOM_CHx_CTRL.SL and CLSi_TOM_CHx_CTRL.CLK_SRC are updated if CLSi_TOM_CHx_CN0.CN0 reaches the value of CLSi_TOM_CHx_CM0.CM0-1 or the channel receives the trigger signal and the counting direction changes from up to down. 11_B Up/down counter mode enabled: CLSi_TOM_CHx_CN0.CN0 counts up and down, CLSi_TOM_CHx_CM0.CM0, CLSi_TOM_CHx_CM1.CM1, CLSi_TOM_CHx_CTRL.SL and CLSi_TOM_CHx_CTRL.CLK_SRC are updated if CLSi_TOM_CHx_CN0.CN0 is 0 or reaches the value of CLSi_TOM_CHx_CM0.CM0-1 or the channel receives the trigger signal and the counting direction changes from up to down.
RST_CCU0	20	rw	**Reset source of CCU0** ***Attention:*** ***This bit may only be set if bit*** CLSi_TOM_CHx_CTRL.OSM***=0 (i.e. in continuous mode)*** 0_B Reset counter CLSi_TOM_CHx_CN0.CN0 to 0 on matching comparison CLSi_TOM_CHx_CM0.CM0 1_B Reset counter CLSi_TOM_CHx_CN0.CN0 to 0 on trigger TRIG_IN[x:x] or TOM_EXT_TRIGIN[x:x]
OSM_TRIG	21	rw	**One-shot pulse generation enabled by the selected trigger signal** ***Attention:*** ***This bit may only be set if bit*** CLSi_TOM_CHx_CTRL.OSM***=1 and bit*** CLSi_TOM_CHx_CTRL.RST_CCU0***=0.*** 0_B Signal TOM_CH_TRIGIN[x:x] or TOM_EXT_TRIGIN[x:x] cannot trigger start of single pulse generation 1_B Signal TOM_CH_TRIGIN[x:x] or TOM_EXT_TRIGIN[x:x] can trigger start of single pulse generation (only if bit CLSi_TOM_CHx_CTRL.OSM = 1)

(table continues...)

(continued)

Field	Bits	Type	Description
EXT_TRIG	22	rw	**Select** TOM_EXT_TRIGIN***[x:x]*** **as trigger signal** 0_B If CLSi_TOM_CHx_CTRL.RST_CCU0=1, signal TOM_CH_TRIGIN[x:x] is selected as trigger to reset CLSi_TOM_CHx_CN0.CN0 or to start single pulse generation 1_B If CLSi_TOM_CHx_CTRL.RST_CCU0=1, signal TOM_EXT_TRIGIN[x:x] is selected as trigger to reset CLSi_TOM_CHx_CN0.CN0 or to start single pulse generation
EXTTRIGOUT	23	rw	**Select** TOM_EXT_TRIGIN***[x:x]*** **as potential output signal** TOM_CH_TRIGOUT***[x:x]*** 0_B Signal TOM_CH_TRIGIN[x:x] is selected as output on TOM_CH_TRIGOUT[x:x] (if CLSi_TOM_CHx_CTRL.TRIGOUT =0) 1_B Signal TOM_EXT_TRIGIN[x:x] after synchronization to the selected CMU clock resolution is selected as output on TOM_CH_TRIGOUT[x:x] (if CLSi_TOM_CHx_CTRL.TRIGOUT = 0)
TRIGOUT	24	rw	**Trigger output selection (output signal** TOM_CH_TRIGOUT***[x:x]***) **of module TOM channel [x]** 0_B TOM_CH_TRIGOUT[x:x] is TOM_CH_TRIGIN[x:x] or TOM_EXT_TRIGIN[x:x] 1_B TOM_CH_TRIGOUT[x:x] is TRIG_CCU0
OSM	26	rw	**One-shot mode.** In this mode the counter CLSi_TOM_CHx_CN0 counts for only one period. The length of period is defined by CLSi_TOM_CHx_CM0. A write-access to the register CLSi_TOM_CHx_CN0 triggers the start of counting. 0_B One-shot mode disabled 1_B One-shot mode enabled
BITREV	27	rw	**Bit-reversing of output of counter** CLSi_TOM_CHx_CN0**.** ***Note**: This bit enables the PCM mode and is only available for **TOM[i]** channel 15.*
FREEZE	31	rw	**TOM[i] Freeze Mode enable** 0_B A channel disable/enable may change internal register and output register 1_B A channel enable/disable does not change an internal or output register but stops counter CLSi_TOM_CHx_CN0.CN0
0	6:0, 10:8, 16, 25, 28, 29, 30	r	**Reserved** Read as 0; must be written with 0.

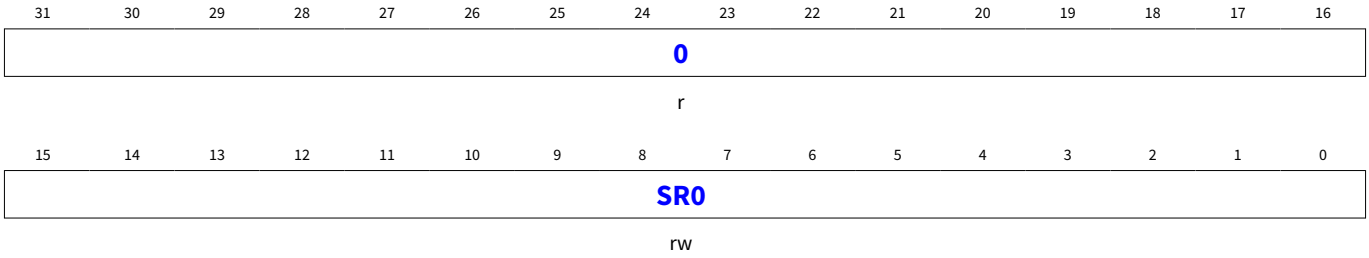
Table 2330 Reset values of CLSi_TOM_CHx_CTRL (i=0-2;x=15)

Reset	Reset value	Note
Kernel Reset	0000 0800$_H$	
TOM channel reset	0000 0800$_H$	

44.5.76 TOM[i] channel [x] CCU0 compare shadow register (x=0-15;i=0-2)

CLSi_TOM_CHx_SR0 (i=0-2;x=0-15) Offset address: 001004$_H$+i*20000$_H$+x*40$_H$

TOM[i] channel [x] CCU0 compare shadow register Reset values see: Table 2331



Field	Bits	Type	Description
SR0	15:0	rw	**TOM[i] channel [x] shadow register** CLSi_TOM_CHx_SR0 **for update of compare register** CLSi_TOM_CHx_CM0
0	31:16	r	**Reserved** Read as 0; must be written with 0.

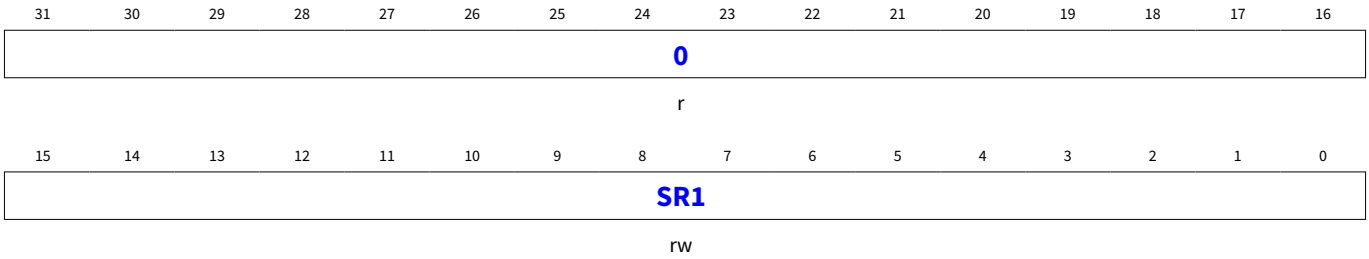
Table 2331 Reset values of CLSi_TOM_CHx_SR0 (i=0-2;x=0-15)

Reset	Reset value	Note
Kernel Reset	0000 0000$_H$	
TOM channel reset	0000 0000$_H$	

44.5.77 TOM[i] channel [x] CCU1 compare shadow register (x=0-15;i=0-2)

CLSi_TOM_CHx_SR1 (i=0-2;x=0-15) Offset address: 001008$_H$+i*20000$_H$+x*40$_H$

TOM[i] channel [x] CCU1 compare shadow register Reset values see: Table 2332



Field	Bits	Type	Description
SR1	15:0	rw	**TOM[i] channel [x] shadow register** CLSi_TOM_CHx_SR1 **for update of compare register** CLSi_TOM_CHx_CM1
0	31:16	r	**Reserved** Read as 0; must be written with 0.

Table 2332 Reset values of CLSi_TOM_CHx_SR1 (i=0-2;x=0-15)

Reset	Reset value	Note
Kernel Reset	0000 0000$_H$	
TOM channel reset	0000 0000$_H$	

44.5.78 TOM[i] channel [x] CCU0 compare register (x=0-15;i=0-2)

CLSi_TOM_CHx_CM0 (i=0-2;x=0-15) Offset address: 00100C$_H$+i*20000$_H$+x*40$_H$

TOM[i] channel [x] CCU0 compare register Reset values see: Table 2333

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CM0															
rwh															

Field	Bits	Type	Description
CM0	15:0	rwh	**TOM[i] channel [x] CCU0 compare register**
0	31:16	r	**Reserved** Read as 0; must be written with 0.

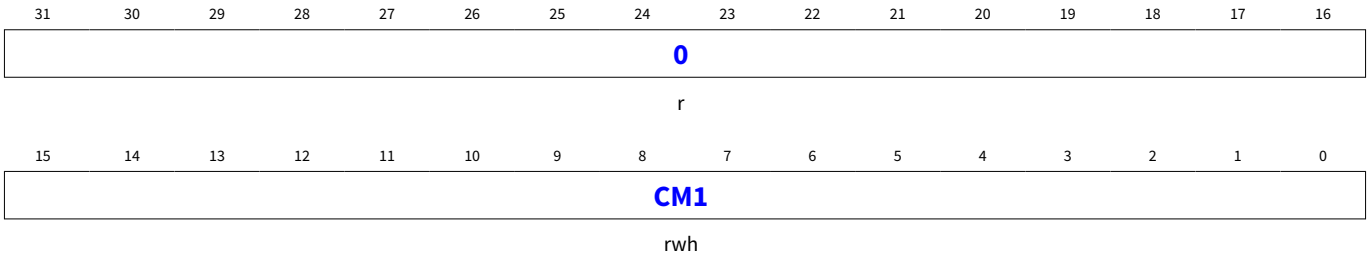
Table 2333 Reset values of CLSi_TOM_CHx_CM0 (i=0-2;x=0-15)

Reset	Reset value	Note
Kernel Reset	0000 0000$_H$	
TOM channel reset	0000 0000$_H$	

44.5.79 TOM[i] channel [x] CCU1 compare register (x=0-15;i=0-2)

CLSi_TOM_CHx_CM1 (i=0-2;x=0-15) Offset address: 001010$_H$+i*20000$_H$+x*40$_H$

TOM[i] channel [x] CCU1 compare register Reset values see: Table 2334



Field	Bits	Type	Description
CM1	15:0	rwh	**TOM[i] channel [x] CCU1 compare register**
0	31:16	r	**Reserved** Read as 0; must be written with 0.

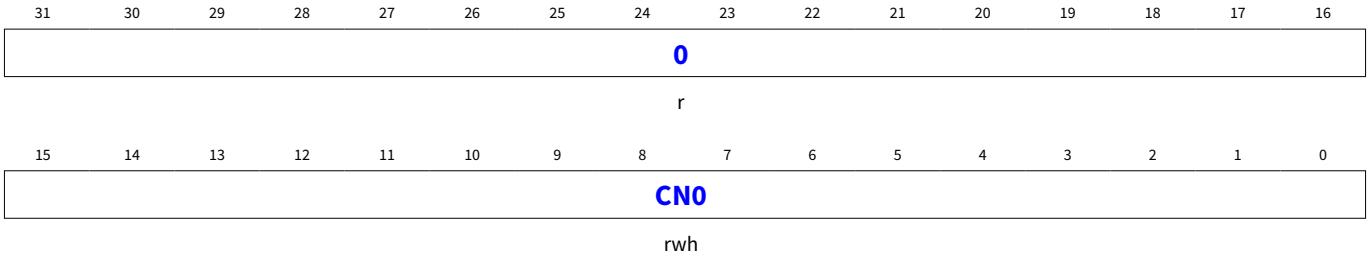
Table 2334 Reset values of CLSi_TOM_CHx_CM1 (i=0-2;x=0-15)

Reset	Reset value	Note
Kernel Reset	0000 0000$_H$	
TOM channel reset	0000 0000$_H$	

44.5.80 TOM[i] channel [x] CCU0 counter (x=0-15;i=0-2)

CLSi_TOM_CHx_CN0 (i=0-2;x=0-15) Offset address: 001014$_H$+i*20000$_H$+x*40$_H$

TOM[i] channel [x] CCU0 counter Reset values see: Table 2335



Field	Bits	Type	Description
CN0	15:0	rwh	**TOM[i] CCU0 counter** This counter is stopped if the **TOM[i]** channel is disabled and not reset on an enable event of **TOM[i]** channel.
0	31:16	r	**Reserved** Read as 0; must be written with 0.

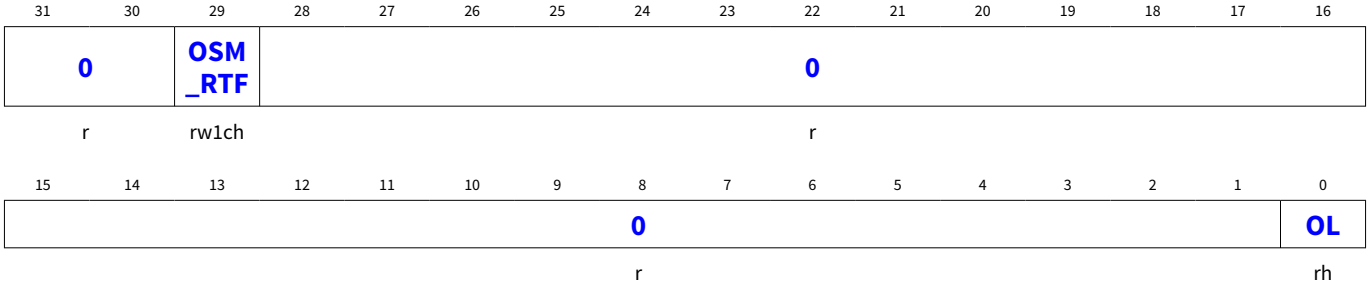
Table 2335 Reset values of CLSi_TOM_CHx_CN0 (i=0-2;x=0-15)

Reset	Reset value	Note
Kernel Reset	0000 0000$_H$	
TOM channel reset	0000 0000$_H$	

44.5.81 TOM[i] channel [x] status register (x=0-15;i=0-2)

CLSi_TOM_CHx_STAT (i=0-2;x=0-15) Offset address: 001018_H+i*20000_H+x*40_H

TOM[i] channel [x] status register Reset values see: Table 2336



Field	Bits	Type	Description
OL	0	rh	**Output level of output** TOM_OUT***[x:x]*** ***Note**: Reset value is the inverted value of* CLSi_TOM_CHx_CTRL**.**SL *bit which depends on the hardware configuration chosen by silicon vendor.*
OSM_RTF	29	rw1ch	**One-shot mode retrigger failed flag** ***Note**: This bit will be cleared on a configuration interface write-access of value 1. A read-access leaves the bit unchanged.* **If reading:** 0_B One-shot retrigger operation successful 1_B One-shot retrigger operation failed **If writing:** 0_B No action 1_B Clear status bit to zero
0	28:1, 31:30	r	**Reserved** Read as 0; must be written with 0.

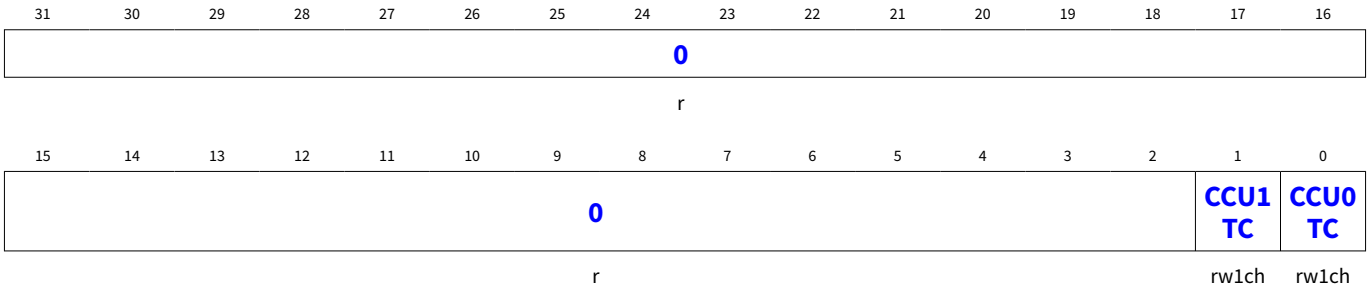
Table 2336 Reset values of CLSi_TOM_CHx_STAT (i=0-2;x=0-15)

Reset	Reset value	Note
Kernel Reset	0000 0000_H	
TOM channel reset	0000 0000_H	

44.5.82 TOM[i] channel [x] interrupt notification register (x=0-15;i=0-2)

CLSi_TOM_CHx_IRQ_NOTIFY (i=0-2;x=0-15) Offset address: $00101C_H$+i*20000_H+x*40_H

TOM[i] channel [x] interrupt notification register Reset values see: Table 2337



Field	Bits	Type	Description
CCU0TC	0	rw1ch	**CCU0 Trigger condition interrupt for channel [x]** ***Note**: The notification of the interrupt is only triggered once after reaching the condition CLSi_TOM_CHx_CN0.CN0 >= CLSi_TOM_CHx_CM0.CM0-1.* ***Note**: This bit will be cleared on a configuration interface write-access of value 1. A read-access leaves the bit unchanged.* **If reading:** 0_B No interrupt triggered by software or by the compare event of CLSi_TOM_CHx_CN0.CN0 >= CLSi_TOM_CHx_CM0.CM0-1 in CCU0 unit was raised 1_B Interrupt triggered by software or by the compare event of CLSi_TOM_CHx_CN0.CN0 >= CLSi_TOM_CHx_CM0.CM0-1 in CCU0 unit was raised **If writing:** 0_B No action 1_B Clear interrupt

(table continues...)

(continued)

Field	Bits	Type	Description
CCU1TC	1	rw1ch	**CCU1 Trigger condition interrupt for channel [x]** ***Note**: If CLSi_TOM_CHx_CTRL.SR0_TRIG = 0, the notification of the interrupt is only triggered once after reaching the condition CLSi_TOM_CHx_CN0.CN0 >= CLSi_TOM_CHx_CM1.CM1-1.* ***Note**: This bit will be cleared on a configuration interface write-access of value 1. A read-access leaves the bit unchanged.* **If reading and CLSi_TOM_CHx_CTRL .SR0_TRIG == 0:** 0_B No interrupt triggered by software or the compare event of CLSi_TOM_CHx_CN0.CN0 >= CLSi_TOM_CHx_CM1.CM1-1 in CCU1 unit was raised 1_B Interrupt triggered by software or by the compare event of CLSi_TOM_CHx_CN0.CN0 >= CLSi_TOM_CHx_CM1.CM1-1 in CCU1 unit was raised **If writing and CLSi_TOM_CHx_CTRL .SR0_TRIG == 0:** 0_B No action 1_B Clear interrupt **If reading and CLSi_TOM_CHx_CTRL .SR0_TRIG == 1:** 0_B No interrupt triggered by software or the compare event of CLSi_TOM_CHx_CN0.CN0 >= CLSi_TOM_CHx_SR0.SR0-1 in CCU0 unit was raised 1_B Interrupt triggered by software or the compare event of CLSi_TOM_CHx_CN0.CN0 >= CLSi_TOM_CHx_SR0.SR0-1 in CCU0 unit was raised **If writing and CLSi_TOM_CHx_CTRL .SR0_TRIG == 1:** 0_B No action 1_B Clear interrupt
0	31:2	r	**Reserved** Read as 0; must be written with 0.

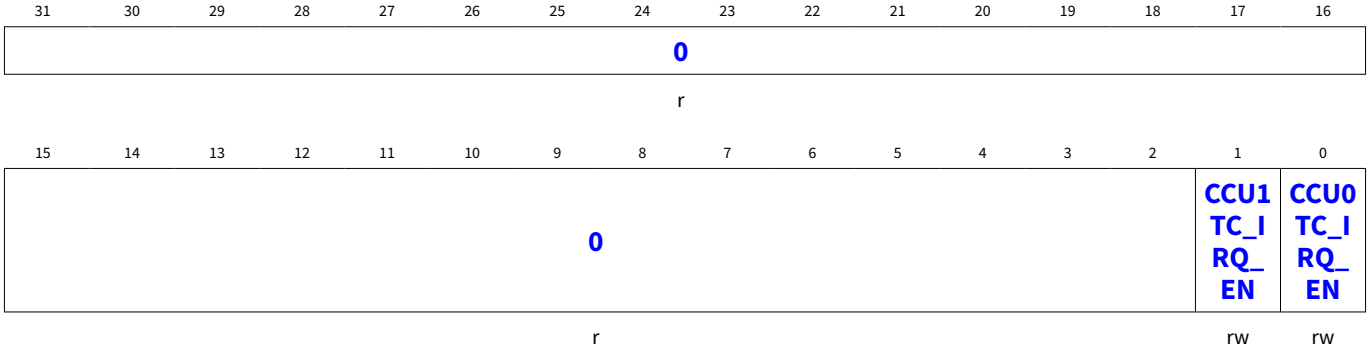
Table 2337 Reset values of CLSi_TOM_CHx_IRQ_NOTIFY (i=0-2;x=0-15)

Reset	Reset value	Note
Kernel Reset	0000 0000$_H$	
TOM channel reset	0000 0000$_H$	

44.5.83 TOM[i] channel [x] interrupt enable register (x=0-15;i=0-2)

CLSi_TOM_CHx_IRQ_EN (i=0-2;x=0-15) Offset address: 001020$_H$+i*20000$_H$+x*40$_H$

TOM[i] channel [x] interrupt enable register Reset values see: Table 2338



Field	Bits	Type	Description
CCU0TC_IRQ_EN	0	rw	TOM_CCU0TC[x]_IRQ **interrupt enable** 0_B The interrupt TOM_CCU0TC[x]_IRQ is not visible outside eGTM 1_B The interrupt TOM_CCU0TC[x]_IRQ is visible outside eGTM
CCU1TC_IRQ_EN	1	rw	TOM_CCU1TC[x]_IRQ **interrupt enable** 0_B The interrupt TOM_CCU1TC[x]_IRQ is not visible outside eGTM 1_B The interrupt TOM_CCU1TC[x]_IRQ is visible outside eGTM
0	31:2	r	**Reserved** Read as 0; must be written with 0.

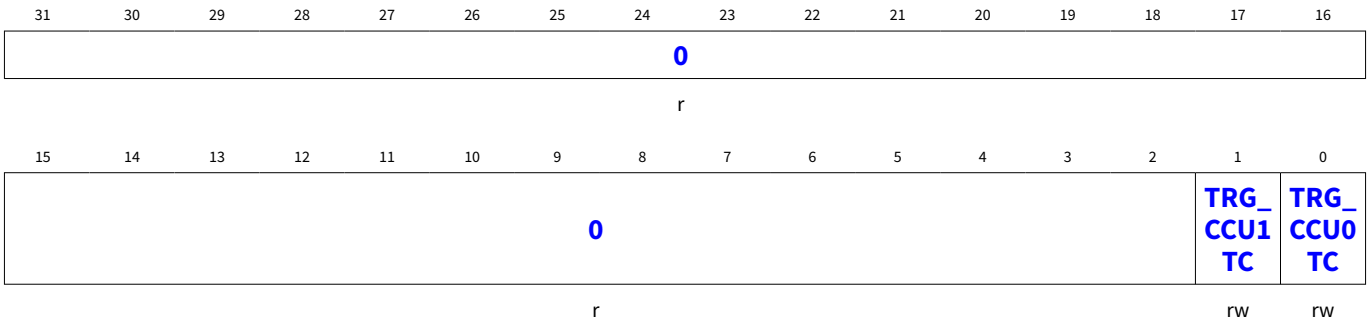
Table 2338 Reset values of CLSi_TOM_CHx_IRQ_EN (i=0-2;x=0-15)

Reset	Reset value	Note
Kernel Reset	$0000\ 0000_H$	
TOM channel reset	$0000\ 0000_H$	

44.5.84 TOM[i] channel [x] force interrupt register (x=0-15;i=0-2)

CLSi_TOM_CHx_IRQ_FORCINT (i=0-2;x=0-15) Offset address: $001024_H+i*20000_H+x*40_H$

TOM[i] channel [x] force interrupt register Reset values see: Table 2339



Field	Bits	Type	Description
TRG_CCU0TC	0	rw	**Trigger the bit** CLSi_TOM_CHx_IRQ_NOTIFY.CCU0TC **by software** ***Note**: This bit is cleared automatically after write.* ***Note**: This bit is write-protected by register bit CLSi_ARCH_CTRL.RF_PROT.* **If reading:** 0_B No status **If writing:** 0_B No action 1_B Force setting of CLSi_TOM_CHx_IRQ_NOTIFY.CCU0TC = 1
TRG_CCU1TC	1	rw	**Trigger the bit** CLSi_TOM_CHx_IRQ_NOTIFY.CCU1TC **by software** ***Note**: This bit is cleared automatically after write.* ***Note**: This bit is write-protected by register bit CLSi_ARCH_CTRL.RF_PROT.* **If reading:** 0_B No status **If writing:** 0_B No action 1_B Force setting of CLSi_TOM_CHx_IRQ_NOTIFY.CCU1TC = 1
0	31:2	r	**Reserved** Read as 0; must be written with 0.

Table 2339 Reset values of CLSi_TOM_CHx_IRQ_FORCINT (i=0-2;x=0-15)

Reset	Reset value	Note
Kernel Reset	$0000\ 0000_H$	
TOM channel reset	$0000\ 0000_H$	

44.5.85 TOM[i] channel [x] interrupt mode register (x=0-15;i=0-2)

CLSi_TOM_CHx_IRQ_MODE (i=0-2;x=0-15) Offset address: 001028_H+i*20000_H+x*40_H

TOM[i] channel [x] interrupt mode register Reset values see: Table 2340

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0														IRQ_MODE	
r														rw	

Field	Bits	Type	Description
IRQ_MODE	1:0	rw	**IRQ mode selection** ***Note**: The interrupt modes are described in **eGTM** interrupt concept.* 00_B Level mode 01_B Pulse mode 10_B Pulse-Notify mode 11_B Single-Pulse mode
0	31:2	r	**Reserved** Read as 0; must be written with 0.

Table 2340 Reset values of CLSi_TOM_CHx_IRQ_MODE (i=0-2;x=0-15)

Reset	Reset value	Note
Kernel Reset	$0000\ 0002_H$	
TOM channel reset	$0000\ 0002_H$	

44.5.86 TOM[i] channel [x] control shadow register (x=0-15;i=0-2)

CLSi_TOM_CHx_CTRL_SR (i=0-2;x=0-15) Offset address: 001030_H+i*20000_H+x*40_H

TOM[i] channel [x] control shadow register Reset values see: Table 2341

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CLK_SRC_SR				SL_SR	0										
rw				rw	r										

Field	Bits	Type	Description
SL_SR	11	rw	**Shadow register for** CLSi_TOM_CHx_CTRL**.**SL ***Note**: Reset value depends on the hardware configuration chosen by silicon vendor.* 0_B Low signal level 1_B High signal level

(table continues...)

(continued)

Field	Bits	Type	Description
CLK_SRC_SR	15:12	rw	**Shadow register for** CLSi_TOM_CHx_CTRL.CLK_SRC ***Note**: This register is a shadow register for CLSi_TOM_CHx_CTRL.CLK_SRC. Thus, if the selected clock resolution source for PWM generation should be changed during operation, the old selected clock resolution has to operate until the update of the **TOM[i]** channels register CLSi_TOM_CHx_CTRL.CLK_SRC by the CLSi_TOM_CHx_CTRL_SR.CLK_SRC_SR content is done either by an end of a period or a forced update.*
0	10:0, 31:16	r	**Reserved** Read as 0; must be written with 0.

Table 2341 Reset values of CLSi_TOM_CHx_CTRL_SR (i=0-2;x=0-15)

Reset	Reset value	Note
Kernel Reset	$0000\ 0800_H$	
TOM channel reset	$0000\ 0800_H$	

44.5.87 TOM[i] TGC [g] global control register (g=0-1;i=0-2)

CLSi_TOM_TGCg_GLB_CTRL (g=0-1;i=0-2) Offset address: 001430_H+i*20000_H+g*80_H

TOM[i] TGC [g] global control register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UPEN_CTRL7		UPEN_CTRL6		UPEN_CTRL5		UPEN_CTRL4		UPEN_CTRL3		UPEN_CTRL2		UPEN_CTRL1		UPEN_CTRL0	
rw		rw		rw		rw		rw		rw		rw		rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RST_CH7	RST_CH6	RST_CH5	RST_CH4	RST_CH3	RST_CH2	RST_CH1	RST_CH0	0							HOST_TRIG
rw	rw	rw	rw	rw	rw	rw	rw	r							w1c

Field	Bits	Type	Description
HOST_TRIG	0	w1c	**Trigger request signal to update the register** CLSi_TOM_TGCg_ENDIS_STAT, CLSi_TOM_TGCg_OUTEN_STAT ***Note**: This flag is reset automatically after triggering the update.* 0_B No trigger request 1_B Set trigger request

(table continues...)

(continued)

Field	Bits	Type	Description
RST_CHc (c=0-7)	c+8	rw	**Software reset of channel [x]; x = c + g*8** ***Note**: This bit is cleared automatically after a write over the configuration interface. The channel registers are set to their reset values and channel operation is stopped immediately. The internal SR register SOUR is set to '1'.* ***Note**: The write-access to CLSi_TOM_TGCg_GLB_CTRL.RST_CHc bit will take two internal bus clock-cycles.* 0_B No action 1_B Reset channel
UPEN_CTRLc (c=0-7)	2*c+17:2*c+16	rw	**TOM[i] channel [x] (x=c + g*8) enable update of register CLSi_TOM_CHx_CM0, CLSi_TOM_CHx_CM1, CLSi_TOM_CHx_CTRL.SL and CLSi_TOM_CHx_CTRL.CLK_SRC from CLSi_TOM_CHx_SR0, CLSi_TOM_CHx_SR1, CLSi_TOM_CHx_CTRL_SR.SL_SR and CLSi_TOM_CHx_CTRL_SR.CLK_SRC_SR.** **If writing:** 00_B Don't care, bits CLSi_TOM_TGCg_GLB_CTRL.UPEN_CTRLc will not be changed 01_B Disable update 10_B Enable update 11_B Don't care, bits CLSi_TOM_TGCg_GLB_CTRL.UPEN_CTRLc will not be changed **If reading:** 00_B Update disabled 01_B Unused 10_B Unused 11_B Update enabled
0	7:1	r	**Reserved** Read as 0; must be written with 0.

44.5.88 TOM[i] TGC [g] action time base register (g=0-1;i=0-2)

CLSi_TOM_TGCg_ACT_TB (g=0-1;i=0-2) Offset address: 001434_H+i*20000_H+g*80_H

TOM[i] TGC [g] action time base register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0					TBU_SEL		TB_TRIG	ACT_TB							
r					rw		rwh	rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ACT_TB															
rw															

Field	Bits	Type	Description
ACT_TB	23:0	rw	**Action time base. When the selected TBU time base CCM[i]_TBU_TS0/CCM[i]_TBU_TS1/CCM[i]_TBU_TS2 is "cyclically greater than or equal to" the action time base CLSi_TOM_TGCg_ACT_TB.ACT_TB, the compare event specified in CLSi_TOM_TGCg_ACT_TB.ACT_TB is considered to have occurred in the past and so the trigger CTRL_TRIG is immediately generated in TGC[g] unit if CLSi_TOM_TGCg_ACT_TB.TB_TRIG is 1. Please refer to chapter eGTM Architecture for more information about cyclic event compare strategy.**
TB_TRIG	24	rwh	**Set trigger request** ***Note**: This flag is reset automatically if the selected time base unit (CCM[i]_TBU_TS0 or CCM[i]_TBU_TS1 or CCM[i]_TBU_TS2 if present) has reached the value CLSi_TOM_TGCg_ACT_TB.ACT_TB and the update of the register was triggered.* 0_B No trigger request 1_B Set trigger request
TBU_SEL	26:25	rw	**Selection of time base used for comparison** 00_B CCM[i]_TBU_TS0 selected 01_B CCM[i]_TBU_TS1 selected 10_B CCM[i]_TBU_TS2 selected 11_B Same as 00_B
0	31:27	r	**Reserved** Read as 0; must be written with 0.

44.5.89 TOM[i] TGC [g] force update control register (g=0-1;i=0-2)

CLSi_TOM_TGCg_FUPD_CTRL (g=0-1;i=0-2) Offset address: $001438_H+i*20000_H+g*80_H$

TOM[i] TGC [g] force update control register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSTCN0_CH7		RSTCN0_CH6		RSTCN0_CH5		RSTCN0_CH4		RSTCN0_CH3		RSTCN0_CH2		RSTCN0_CH1		RSTCN0_CH0	
rw		rw		rw		rw		rw		rw		rw		rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FUPD_CTRL7		FUPD_CTRL6		FUPD_CTRL5		FUPD_CTRL4		FUPD_CTRL3		FUPD_CTRL2		FUPD_CTRL1		FUPD_CTRL0	
rw		rw		rw		rw		rw		rw		rw		rw	

Field	Bits	Type	Description
FUPD_CTRLc (c=0-7)	2*c+1:2*c	rw	**Force update control of operation registers of TOM[i] channel [x] (x = c + g*8)** If enabled, force update of register CLSi_TOM_CHx_CM0.CM0, CLSi_TOM_CHx_CM1.CM1, CLSi_TOM_CHx_CTRL.SL and CLSi_TOM_CHx_CTRL.CLK_SRC triggered by CLSi_TOM_TGCg_GLB_CTRL.HOST_TRIG, CLSi_TOM_TGCg_ACT_TB.ACT_TB compare match or internal trigger. ***Note**: The force update request is stored and executed synchronized to the selected clock source resolution defined in CLSi_TOM_CHx_CTRL.CLK_SRC.* **If writing:** 00_B Don't care, bits CLSi_TOM_TGCg_FUPD_CTRL.FUPD_CTRLc will not be changed 01_B Disable force update 10_B Enable force update 11_B Don't care, bits CLSi_TOM_TGCg_FUPD_CTRL.FUPD_CTRLc will not be changed **If reading:** 00_B Force update disabled 01_B Unused 10_B Unused 11_B Force update enabled
RSTCN0_CHc (c=0-7)	2*c+17:2*c +16	rw	**Reset CLSi_TOM_CHx_CN0 of channel [x] (x=c + g*8) with the force update event** If enabled, CLSi_TOM_CHx_CN0.CN0 is reset together with a force update trigger event. **If writing:** 00_B Don't care, bits CLSi_TOM_TGCg_FUPD_CTRL.RSTCN0_CHc will not be changed 01_B Do not reset CLSi_TOM_CHx_CN0.CN0 on forced update 10_B Reset CLSi_TOM_CHx_CN0.CN0 on forced update 11_B Don't care, bits CLSi_TOM_TGCg_FUPD_CTRL.RSTCN0_CHc will not be changed **If reading:** 00_B CLSi_TOM_CHx_CN0.CN0 is not reset on forced update 01_B Unused 10_B Unused 11_B CLSi_TOM_CHx_CN0.CN0 is reset on forced update

44.5.90 TOM[i] TGC [g] internal trigger control register (g=0-1;i=0-2)

CLSi_TOM_TGCg_INT_TRIG (g=0-1;i=0-2) Offset address: $00143C_H$+i*20000_H+g*80_H

TOM[i] TGC [g] internal trigger control register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
INT_TRIG7		INT_TRIG6		INT_TRIG5		INT_TRIG4		INT_TRIG3		INT_TRIG2		INT_TRIG1		INT_TRIG0	
rw		rw		rw		rw		rw		rw		rw		rw	

Field	Bits	Type	Description
INT_TRIGc (c=0-7)	2*c+1:2*c	rw	**Select input signal** TOM_CH_TRIGOUT***[x:x]*** **as a trigger source for TGC[g] (x=c + g*8)** **If writing:** 00_B Don't care, bits CLSi_TOM_TGCg_INT_TRIG.INT_TRIGc will not be changed 01_B Do not use internal trigger from channel x (TOM_CH_TRIGOUT[x:x]) 10_B Use internal trigger from channel x (TOM_CH_TRIGOUT[x:x]) 11_B Don't care, bits CLSi_TOM_TGCg_INT_TRIG.INT_TRIGc will not be changed **If reading:** 00_B Internal trigger from channel x (TOM_CH_TRIGOUT[x:x]) not used 01_B Unused 10_B Unused 11_B Internal trigger from channel x (TOM_CH_TRIGOUT[x:x]) used
0	31:16	r	**Reserved** Read as 0; must be written with 0.

44.5.91 TOM[i] TGC [g] enable/disable control register (g=0-1;i=0-2)

CLSi_TOM_TGCg_ENDIS_CTRL (g=0-1;i=0-2) Offset address: $001470_H+i*20000_H+g*80_H$

TOM[i] TGC [g] enable/disable control register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ENDIS_CTRL7		ENDIS_CTRL6		ENDIS_CTRL5		ENDIS_CTRL4		ENDIS_CTRL3		ENDIS_CTRL2		ENDIS_CTRL1		ENDIS_CTRL0	
rw		rw		rw		rw		rw		rw		rw		rw	

Field	Bits	Type	Description
ENDIS_CTRLc (c=0-7)	2*c+1:2*c	rw	**TOM[i] channel [x] (x=c + g*8) enable/disable control register.** **If writing:** 00_B Bit field CLSi_TOM_TGCg_ENDIS_STAT.ENDIS_STATc will not be changed on an update trigger 01_B Disable channel on an update trigger 10_B Enable channel on an update trigger 11_B Don't change bits of this register **If reading:** 00_B Bit field CLSi_TOM_TGCg_ENDIS_STAT.ENDIS_STATc will not be changed on an update trigger 01_B Disable channel on an update trigger 10_B Enable channel on an update trigger
0	31:16	r	**Reserved** Read as 0; must be written with 0.

44.5.92 TOM[i] TGC [g] enable/disable status register (g=0-1;i=0-2)

CLSi_TOM_TGCg_ENDIS_STAT (g=0-1;i=0-2) Offset address: $001474_H+i*20000_H+g*80_H$

TOM[i] TGC [g] enable/disable status register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ENDIS_STAT7		ENDIS_STAT6		ENDIS_STAT5		ENDIS_STAT4		ENDIS_STAT3		ENDIS_STAT2		ENDIS_STAT1		ENDIS_STAT0	
rwh		rwh		rwh		rwh		rwh		rwh		rwh		rwh	

Field	Bits	Type	Description
ENDIS_STATc (c=0-7)	2*c+1:2*c	rwh	**TOM[i] channel [x] (x=c + g*8) enable/disable status register** **If writing:** 00_B Don't care, bits CLSi_TOM_TGCg_ENDIS_STAT.ENDIS_STATc will not be changed 01_B Disable channel 10_B Enable channel 11_B Don't care, bits CLSi_TOM_TGCg_ENDIS_STAT.ENDIS_STATc will not be changed **If reading:** 00_B Channel disabled 01_B Unused 10_B Unused 11_B Channel enabled

(table continues...)

(continued)

Field	Bits	Type	Description
0	31:16	r	**Reserved** Read as 0; must be written with 0.

44.5.93 TOM[i] TGC [g] output enable control register (g=0-1;i=0-2)

CLSi_TOM_TGCg_OUTEN_CTRL (g=0-1;i=0-2) Offset address: 001478_H+i*20000_H+g*80_H

TOM[i] TGC [g] output enable control register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

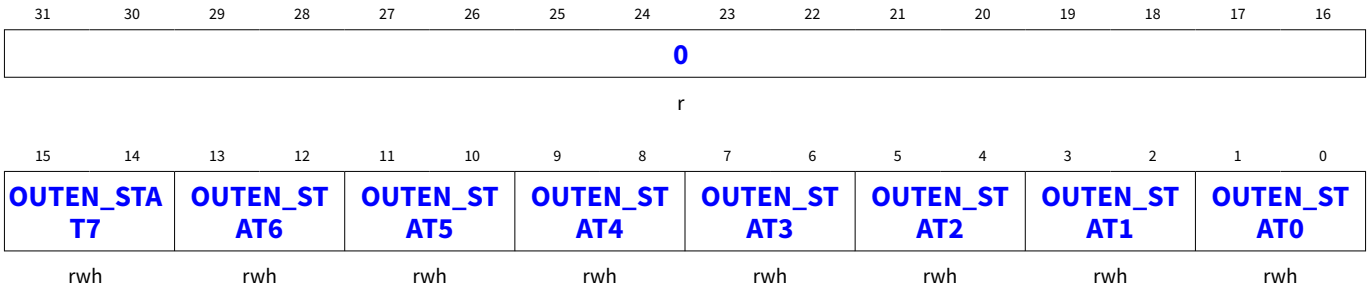
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OUTEN_CTRL7		OUTEN_CTRL6		OUTEN_CTRL5		OUTEN_CTRL4		OUTEN_CTRL3		OUTEN_CTRL2		OUTEN_CTRL1		OUTEN_CTRL0	
rw		rw		rw		rw		rw		rw		rw		rw	

Field	Bits	Type	Description
OUTEN_CTRLc (c=0-7)	2*c+1:2*c	rw	**Output enable control of TOM [i] channel [x] output** TOM_OUT***[x:x]*****, x=c+g*8** **If writing:** 00_B Bit field CLSi_TOM_TGCg_OUTEN_STAT.OUTEN_STATc will not be changed on an update trigger 01_B Disable channel output on an update trigger 10_B Enable channel output on an update trigger 11_B Don't change bits of this register **If reading:** 00_B Bit field CLSi_TOM_TGCg_OUTEN_STAT.OUTEN_STATc will not be changed on an update trigger 01_B Disable channel output on an update trigger 10_B Enable channel output on an update trigger
0	31:16	r	**Reserved** Read as 0; must be written with 0.

44.5.94 TOM[i] TGC [g] output enable status register (g=0-1;i=0-2)

CLSi_TOM_TGCg_OUTEN_STAT (g=0-1;i=0-2) Offset address: $00147C_H$+i*20000_H+g*80_H

TOM[i] TGC [g] output enable status register Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
OUTEN_STATc (c=0-7)	2*c+1:2*c	rwh	**Output enable status of TOM [i] channel [x] output** TOM_OUT***[x:x]*****, x=c+g*8** ***Note**:* *If the output is disabled (*TOM_OUTEN*[x:x]=0), the **TOM[i]** channel x output* TOM_OUT*[x:x] and* TOM_OUT_T*[x:x] is the inverse value of bit* CLSi_TOM_CHx_CTRL**.**SL*.* **If writing:** 00_B Don't care, bits CLSi_TOM_TGCg_OUTEN_STAT.OUTEN_STATc will not be changed 01_B Disable output 10_B Enable output 11_B Don't care, bits CLSi_TOM_TGCg_OUTEN_STAT.OUTEN_STATc will not be changed **If reading:** 00_B Output disabled 01_B Unused 10_B Unused 11_B Output enabled
0	31:16	r	**Reserved** Read as 0; must be written with 0.

44.5.95 ATOM[i] channel [x] control register (x=0-7;i=0-2) (i=0-2;x=0-7)

***Note**:* *Please see following multiview register description for more details.*

CLSi_ATOM_CHx_CTRL (i=0-2;x=0-7)

Offset address: 001804_H+i*20000_H+x*80_H

ATOM[i] channel [x] control register

Reset values see: Table 2342

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FREEZE	SOMB	EXT_FUPD	0	0	OSM	0	TRIGOUT	EXTTRIGOUT	EXT_TRIG	OSM_TRIG	RST_CCU0	UDMODE		TRIG_PULSE	WR_REQ
rw	rw	rw	r	r	rw	r	rw	rw	rw	rw	rw	rw		rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CLK_SRC				SL	EUPM	CMP_CTRL	ACB					0	TB12_SEL	MODE	
rwh				rwh	rw	rw	rw					r	rw	rw	

Field	Bits	Type	Description
MODE	1:0	rw	**ATOM channel mode select.** 00_B SOMI mode 01_B SOMC mode 10_B SOMP mode 11_B SOMS mode
TB12_SEL	2	rw	**Select time base value** CCM[i]_TBU_TS1 **or** CCM[i]_TBU_TS2**.**
ACB	8:4	rw	**ATOM Mode control bits.**
CMP_CTRL	9	rw	**CCU[x] compare strategy select.**
EUPM	10	rw	**Extended update mode**
SL	11	rwh	**Initial signal level.**
CLK_SRC	15:12	rwh	**CMU clock source**
WR_REQ	16	rw	**CPU Write request bit for late compare register update.**
TRIG_PULSE	17	rw	**Trigger output pulse length of one cluster clock period**
UDMODE	19:18	rw	**Up/down counter mode**
RST_CCU0	20	rw	**Reset source of CCU0**
OSM_TRIG	21	rw	**Enable trigger of one-shot pulse by the selected trigger signal**
EXT_TRIG	22	rw	**Select** EXT_TRIGIN***[x:x]*** **as trigger signal**
EXTTRIGOUT	23	rw	**Select** EXT_TRIGIN***[x:x]*** **as potential output signal** ATOM_CH_TRIGOUT***[x:x]***
TRIGOUT	24	rw	**Trigger output selection (output signal** ATOM_CH_TRIGOUT***[x:x]***) **of module ATOM [i] channel [x].**
OSM	26	rw	**One-shot mode**
EXT_FUPD	29	rw	**Enable force update by external trigger signal**
SOMB	30	rw	**SOMB mode** 0_B Disable SOMB mode 1_B Enable SOMB mode
FREEZE	31	rw	**ATOM Freeze Mode enable**

(table continues...)

(continued)

Field	Bits	Type	Description
0	3, 25, 27, 28	r	**Reserved** Read as 0; must be written with 0.

Table 2342 Reset values of CLSi_ATOM_CHx_CTRL (i=0-2;x=0-7)

Reset	Reset value	Note
Kernel Reset	0000 0800$_H$	
ATOM channel reset	0000 0800$_H$	

44.5.96 ATOM[i] channel [x] control register (x=0-7;i=0-2) (i=0-2;x=0-7)

***Note**: Please see following multiview register description for more details.*

CLSi_ATOM_CHx_CTRL_SOMI (i=0-2;x=0-7) Offset address: 001804$_H$+i*20000$_H$+x*80$_H$

ATOM[i] channel [x] control register Reset values see: Table 2343

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FREEZE	SOMB	EXT_FUPD	0	ABM	OSM	SLA	TRIGOUT	EXTTRIGOUT	EXT_TRIG	OSM_TRIG	RST_CCU0	UDMODE		TRIG_PULSE	WR_REQ
rw	rw	rw	r	rw	rw	rw	rw	rw	rw	rw	rw	rw		rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CLK_SRC				SL	EUPM	CMP_CTRL	ACB41				ACB0	0	TB12_SEL	MODE	
rw				rw	rw	rw	rw				rw	r	rw	rw	

Field	Bits	Type	Description
MODE	1:0	rw	**ATOM channel mode select.** ***Note**: Set* CLSi_ATOM_CHx_CTRL_SOMI.MODE *= 00$_B$ for **ATOM** Signal Output Mode Immediate (SOMI)* 00$_B$ SOMI mode 01$_B$ SOMC mode 10$_B$ SOMP mode 11$_B$ SOMS mode
TB12_SEL	2	rw	**Select time base value** CCM[i]_TBU_TS1 **or** CCM[i]_TBU_TS2**, but not applicable in SOMI mode.** 0$_B$ Initial value, no function in this mode 1$_B$ Prohibited

(table continues...)

(continued)

Field	Bits	Type	Description
ACB0	4	rw	**ATOM output control** 0_B Set output to the inverse value of CLSi_ATOM_CHx_CTRL_SOMI.SL bit 1_B Set output to CLSi_ATOM_CHx_CTRL_SOMI.SL
ACB41	8:5	rw	**ATOM Mode control bits but not applicable in SOMI mode.** 0_H Initial value, no function in this mode 1_H Prohibited 2_H Prohibited 3_H Prohibited 4_H Prohibited 5_H Prohibited 6_H Prohibited 7_H Prohibited 8_H Prohibited 9_H Prohibited A_H Prohibited B_H Prohibited C_H Prohibited D_H Prohibited E_H Prohibited F_H Prohibited
CMP_CTRL	9	rw	**CCU[x] compare strategy selection, but not applicable in SOMI mode.** 0_B Initial value, no function in this mode 1_B Prohibited
EUPM	10	rw	**Extended update mode, but not applicable in SOMI mode.** 0_B Initial value, no function in this mode 1_B Prohibited
SL	11	rw	**Initial signal level.** ***Note**:* *Reset value depends on the hardware configuration chosen by silicon vendor.* ***Note**:* *If the output of **ATOM** i channel x is disabled (ATOM_OUTEN[x:x]=0), the output ATOM_OUT[x:x] of instance i, channel x will be set to the inverse value of CLSi_ATOM_CHx_CTRL_SOMI.SL. If the output is enabled (ATOM_OUTEN[x:x]=1), the output ATOM_OUT[x:x] of instance i, channel x will be set according to the configuration of CLSi_ATOM_CHx_CTRL_SOMI. The output is not relevant to ATOM_ENDIS[x:x] or CLSi_ATOM_CHx_CTRL_SOMI.FREEZE. The output ATOM_OUT_T[x:x] is not supported in SOMI mode.* 0_B Low signal level 1_B High signal level

(table continues...)

(continued)

Field	Bits	Type	Description
CLK_SRC	15:12	rw	**CMU clock source but not applicable in SOMI mode** 0_H CCM[i]_CLK_RES[0:0] resolution in use 1_H Prohibited 2_H Prohibited 3_H Prohibited 4_H Prohibited 5_H Prohibited 6_H Prohibited 7_H Prohibited 8_H Prohibited 9_H Prohibited A_H Prohibited B_H Prohibited C_H Prohibited D_H Prohibited E_H Prohibited F_H Prohibited
WR_REQ	16	rw	**CPU Write request bit for late compare register update, but not applicable in SOMI mode.** 0_B Initial value, no function in this mode 1_B Prohibited
TRIG_PULSE	17	rw	**Trigger output pulse length of one cluster clock period, , but not applicable in SOMI mode.** 0_B Initial value, no function in this mode 1_B Prohibited
UDMODE	19:18	rw	**Up/down counter mode, but not applicable in SOMI mode.** 00_B Initial value, no function in this mode 01_B Prohibited 10_B Prohibited 11_B Prohibited
RST_CCU0	20	rw	**Reset source of CCU0, but not applicable in SOMI mode.** 0_B Initial value, no function in this mode 1_B Prohibited
OSM_TRIG	21	rw	**Enable trigger of one-shot pulse by trigger signal ATOM_CH_TRIGOUT*[x-1:x-1]* or EXT_TRIGIN*[x:x]*, but not applicable in SOMI mode.** ***Note**: This bit may only be set if bit CLSi_ATOM_CHx_CTRL_SOMP.OSM=1 and bit CLSi_ATOM_CHx_CTRL_SOMP.RST_CCU0=0.* 0_B Initial value, no function in this mode 1_B Prohibited

(table continues...)

(continued)

Field	Bits	Type	Description
EXT_TRIG	22	rw	**Select** EXT_TRIGIN***[x:x]*** **as trigger signal, but not applicable in SOMI mode.** 0_B Initial value, no function in this mode 1_B Prohibited
EXTTRIGOUT	23	rw	**Select** EXT_TRIGIN***[x:x]*** **as potential output signal** ATOM_CH_TRIGOUT***[x:x]*** 0_B signal ATOM_CH_TRIGIN[x:x] is selected as output on ATOM_CH_TRIGOUT[x:x] (if CLSi_ATOM_CHx_CTRL_SOMI.TRIGOUT = 0) 1_B signal EXT_TRIGIN[x:x] after synchronization to the selected clock resolution is selected as output on ATOM_CH_TRIGOUT[x:x] (if CLSi_ATOM_CHx_CTRL_SOMI.TRIGOUT = 0)
TRIGOUT	24	rw	**Trigger output selection (output signal** ATOM_CH_TRIGOUT***[x:x]***) **of module ATOM channel x.** 0_B ATOM_CH_TRIGOUT[x:x] is ATOM_CH_TRIGIN[x:x] or EXT_TRIGIN[x:x] 1_B ATOM_CH_TRIGOUT[x:x] is ATOM_TRIG_CCU0
SLA	25	rw	**'Serve last' ARU communication strategy, but not applicable in SOMI mode.** 0_B Initial value, no function in this mode 1_B Prohibited
OSM	26	rw	**One-shot mode, but not applicable in SOMI mode.** 0_B Initial value, no function in this mode 1_B Prohibited
ABM	27	rw	**ARU blocking mode, but not applicable in SOMI mode.** 0_B Initial value, no function in this mode 1_B Prohibited
EXT_FUPD	29	rw	**External forced update** 0_B use ATOM_FUPD[x:x] signal from AGC to force update 1_B use EXT_TRIGIN[x:x] signal to force update
SOMB	30	rw	**SOMB mode** ***Note:*** *Set* CLSi_ATOM_CHx_CTRL_SOMI**.**SOMB *= 0 for SOMI mode.* 0_B Disable SOMB mode, initial value 1_B Enable SOMB mode, prohibited
FREEZE	31	rw	**ATOM freeze mode enable configuration but it is not valid and supported in SOMI mode.**
0	3, 28	r	**Reserved** Read as 0; must be written with 0.

Table 2343 Reset values of CLSi_ATOM_CHx_CTRL_SOMI (i=0-2;x=0-7)

Reset	Reset value	Note
Kernel Reset	$0000\ 0800_H$	

(table continues...)

Table 2343 (continued) Reset values of CLSi_ATOM_CHx_CTRL_SOMI (i=0-2;x=0-7)

Reset	Reset value	Note
ATOM channel reset	0000 0800_H	

44.5.97 ATOM[i] channel [x] control register (x=0-7;i=0-2) (i=0-2;x=0-7)

***Note**: Please see following multiview register description for more details.*

CLSi_ATOM_CHx_CTRL_SOMC (i=0-2;x=0-7) Offset address: 001804_H+i*20000_H+x*80_H

ATOM[i] channel [x] control register Reset values see: Table 2344

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FREEZE	SOMB	EXT_FUPD	0	0	OSM	0	TRIGOUT	EXTTRIGOUT	EXT_TRIG	OSM_TRIG	RST_CCU0	UDMODE		TRIG_PULSE	WR_REQ
rw	rw	rw	r	r	rw	r	rw	rw	rw	rw	rw	rw		rw	rwh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CLK_SRC				SL	EUPM	CMP_CTRL	ACB42			ACB10		0	TB12_SEL	MODE	
rw				rw	rw	rw	rw			rw		r	rw	rw	

Field	Bits	Type	Description
MODE	1:0	rw	**ATOM channel mode select** ***Note**: Set* CLSi_ATOM_CHx_CTRL_SOMC**.**MODE *= 01_B for **ATOM** Signal Output Mode Compare (SOMC)* 00_B SOMI mode 01_B SOMC mode 10_B SOMP mode 11_B SOMS mode
TB12_SEL	2	rw	**Select time base value** CCM[i]_TBU_TS1 **or** CCM[i]_TBU_TS2**.** ***Note**: This bit is only applicable if three time bases are present in the **eGTM**. Otherwise, this bit is reserved.* 0_B CCM[i]_TBU_TS1 selected for comparison 1_B CCM[i]_TBU_TS2 selected for comparison

(table continues...)

(continued)

Field	Bits	Type	Description
ACB10	5:4	rw	**Signal level control** **If** CLSi_ATOM_CHx_CTRL_SOMC **.ARU_EN == 0:** 00_B No signal level change at output (exception in Table 2230 mode CLSi_ATOM_CHx_CTRL_SOMC.ACB42 =001_B) 01_B Set output signal level to 1 when CLSi_ATOM_CHx_CTRL_SOMC.SL = 0 else output signal level to 0 (exception in Table 2230 mode CLSi_ATOM_CHx_CTRL_SOMC.ACB42 =001_B) 10_B Set output signal level to 0 when CLSi_ATOM_CHx_CTRL_SOMC.SL = 0 else output signal level to 1 (exception in Table 2230 mode CLSi_ATOM_CHx_CTRL_SOMC.ACB42 =001_B) 11_B Toggle output signal level (exception in Table 2230 mode CLSi_ATOM_CHx_CTRL_SOMC.ACB42 =001_B) **If** CLSi_ATOM_CHx_CTRL_SOMC **.ARU_EN !=0:** 00_B Initial value, no function in this mode 01_B Prohibited 10_B Prohibited 11_B Prohibited

(table continues...)

(continued)

Field	Bits	Type	Description
ACB42	8:6	rw	**Compare strategy** ***Note**: A writing of 111_B cancels any pending CCU0 or CCU1 compare.* **If** CLSi_ATOM_CHx_CTRL_SOMC **.ARU_EN == 0:** 000_B Compare in CCU0 and CCU1 in parallel, disable the CCUx on a compare match on either of compare units. Use CCM[i]_TBU_TS0 in CCU0 and CCM[i]_TBU_TS1 or CCM[i]_TBU_TS2 in CCU1. 001_B Compare in CCU0 and CCU1 in parallel, disable the CCU0/CCU1 on a compare match on either compare units. Use CCM[i]_TBU_TS0 in CCU0 and CCM[i]_TBU_TS1 or CCM[i]_TBU_TS2 in CCU1. 010_B Compare in CCU0 only against CCM[i]_TBU_TS0 011_B Compare in CCU1 only against CCM[i]_TBU_TS1 or CCM[i]_TBU_TS2 100_B Compare first in CCU0 and then in CCU1. Use CCM[i]_TBU_TS0. 101_B Compare first in CCU0 and then in CCU1. Use CCM[i]_TBU_TS1 or CCM[i]_TBU_TS2. 110_B Compare first in CCU0 and then in CCU1. Use CCM[i]_TBU_TS0 in CCU0 and CCM[i]_TBU_TS1 or CCM[i]_TBU_TS2 in CCU1. 111_B Cancel pending compare events **If** CLSi_ATOM_CHx_CTRL_SOMC **.ARU_EN !=0:** 000_B Initial value, no function in this mode 001_B Prohibited 010_B Prohibited 011_B Prohibited 100_B Prohibited 101_B Prohibited 110_B Prohibited 111_B Cancel pending compare events

(table continues...)

(continued)

Field	Bits	Type	Description
CMP_CTRL	9	rw	**CCU[x] compare strategy select.** ***Note**: If the selected **TBU** time base is CCM[i]_TBU_TS0 that can only be up-counting, cyclic event compare of the selected time base CCM[i]_TBU_TS0 against CLSi_ATOM_CHx_CM0.CM0/ CLSi_ATOM_CHx_CM1.CM1 is performed independent of CLSi_ATOM_CHx_CTRL_SOMC.CMP_CTRL.* 0_B Cyclic event compare of the selected time base (CCM[i]_TBU_TS1/ CCM[i]_TBU_TS2) against CLSi_ATOM_CHx_CM0.CM0/ CLSi_ATOM_CHx_CM1.CM1 is performed when the selected TBU time base (CCM[i]_TBU_TS1/CCM[i]_TBU_TS2) is up-counting. Please refer to chapter Cyclic event compare for more information about cyclic event compare strategy. 1_B Cyclic event compare of the selected time base (CCM[i]_TBU_TS1/ CCM[i]_TBU_TS2) against CLSi_ATOM_CHx_CM0.CM0/ CLSi_ATOM_CHx_CM1.CM1 is performed when the selected TBU time base (CCM[i]_TBU_TS1/CCM[i]_TBU_TS2) is down-counting. Please refer to chapter Cyclic event compare for more information about cyclic event compare strategy.
EUPM	10	rw	**Extended update mode** ***Note**: If CLSi_ATOM_CHx_CTRL_SOMC.EUPM=1 a write-access to CLSi_ATOM_CHx_CM0.CM0 or CLSi_ATOM_CHx_CM1.CM1 never causes an **AEI** write status 10_B.* 0_B No extended update of CLSi_ATOM_CHx_CM0.CM0 and CLSi_ATOM_CHx_CM1.CM1 1_B Extended update mode in case of compare strategy 'serve last': update of CLSi_ATOM_CHx_CM1.CM1 after CCU0 compare match possible.

(table continues...)

(continued)

Field	Bits	Type	Description
SL	11	rw	**Initial signal level.** ***Note**: Reset value depends on the hardware configuration chosen by silicon vendor.* ***Note**: If the output is disabled (ATOM_OUTEN=0), the outputs ATOM_OUT[x:x] and ATOM_OUT_T[x:x] of instance i and channel x are set to !CLSi_ATOM_CHx_CTRL_SOMC.SL. It is independent of CLSi_ATOM_CHx_CTRL_SOMC.FREEZE.* ***Note**: If CLSi_ATOM_CHx_CTRL_SOMC.FREEZE=0, when the channel is disabled (ATOM_ENDIS[x:x]=0) but the output is still enabled (ATOM_OUTEN[x:x]=1), the output ATOM_OUT[x:x] will be set to the value of CLSi_ATOM_CHx_CTRL_SOMC.SL. Thus, if the channel is re-enabled afterwards (ATOM_OUTEN[x:x]=1 && ATOM_ENDIS[x:x]=1), the output ATOM_OUT[x:x] is initially CLSi_ATOM_CHx_CTRL_SOMC.SL. ATOM_OUT_T[x:x] is not supported in SOMC mode.* ***Note**: If CLSi_ATOM_CHx_CTRL_SOMC.FREEZE=1, when the channel is disabled (ATOM_ENDIS[x:x]=0) but the output is still enabled (ATOM_OUTEN[x:x]=1), the output ATOM_OUT[x:x] will be not changed. Thus, if the channel is re-enabled afterwards (ATOM_OUTEN[x:x]=1 && ATOM_ENDIS[x:x]=1), the output ATOM_OUT[x:x] is initially the same state as it is before the channel is disabled. ATOM_OUT_T[x:x] is not supported in SOMC mode.* 0_B Low signal level 1_B High signal level
CLK_SRC	15:12	rw	**CMU clock source but not applicable in SOMC mode** 0_H Initial value, no function in this mode 1_H Prohibited 2_H Prohibited 3_H Prohibited 4_H Prohibited 5_H Prohibited 6_H Prohibited 7_H Prohibited 8_H Prohibited 9_H Prohibited A_H Prohibited B_H Prohibited C_H Prohibited D_H Prohibited E_H Prohibited F_H Prohibited

(table continues...)

(continued)

Field	Bits	Type	Description
WR_REQ	16	rwh	**CPU Write request bit for late compare register update.** ***Note**: With a compare match event or 2 compare match events at "serve last" strategy, the CLSi_ATOM_CHx_CTRL_SOMC.WR_REQ bit will be reset by hardware.* ***Note**: At the point of the force update only the shadow registers CLSi_ATOM_CHx_SR0.SR0 and CLSi_ATOM_CHx_SR1.SR1 are transferred into the CLSi_ATOM_CHx_CM0.CM0, CLSi_ATOM_CHx_CM1.CM1 registers. The output action is still defined by the CLSi_ATOM_CHx_STAT.ACBI bit field described by the **ARU** together with the old compare values for CLSi_ATOM_CHx_CM0.CM0 / CLSi_ATOM_CHx_CM1.CM1.* 0_B No late update requested by writing via configuration interface 1_B Late update requested by writing via configuration interface
TRIG_PULSE	17	rw	**Trigger output pulse length of one cluster clock period, , but not applicable in SOMC mode.** 0_B Initial value, no function in this mode 1_B Prohibited
UDMODE	19:18	rw	**Up/down counter mode, but not applicable in SOMC mode.** 00_B Initial value, no function in this mode 01_B Prohibited 10_B Prohibited 11_B Prohibited
RST_CCU0	20	rw	**Reset source of CCU0, but not applicable in SOMC mode.** 0_B Initial value, no function in this mode 1_B Prohibited
OSM_TRIG	21	rw	**Enable trigger of one-shot pulse by trigger signal ATOM_CH_TRIGOUT*[x-1:x-1]* or EXT_TRIGIN*[x:x]*, but not applicable in SOMC mode.** 0_B Initial value, no function in this mode 1_B Prohibited
EXT_TRIG	22	rw	**Select EXT_TRIGIN*[x:x]* as trigger signal, but not applicable in SOMC mode.** 0_B Initial value, no function in this mode 1_B Prohibited

(table continues...)

(continued)

Field	Bits	Type	Description
EXTTRIGOUT	23	rw	**Select** EXT_TRIGIN***[x:x]*** **as potential output signal** ATOM_CH_TRIGOUT***[x:x]*** 0_B signal ATOM_CH_TRIGIN[x:x] is selected as output on ATOM_CH_TRIGOUT[x:x] (if CLSi_ATOM_CHx_CTRL_SOMC.TRIGOUT = 0) 1_B signal EXT_TRIGIN[x:x] after synchronization to the selected clock resolution is selected as output on ATOM_CH_TRIGOUT[x:x] (if CLSi_ATOM_CHx_CTRL_SOMC.TRIGOUT = 0)
TRIGOUT	24	rw	**Trigger output selection (output signal** ATOM_CH_TRIGOUT***[x:x]***) **of module ATOM channel x.** 0_B ATOM_CH_TRIGOUT[x:x] is ATOM_CH_TRIGIN[x:x] or EXT_TRIGIN[x:x] 1_B ATOM_CH_TRIGOUT[x:x] is ATOM_TRIG_CCU0
OSM	26	rw	**One-shot mode, but not applicable in SOMC mode.** 0_B Initial value, no function in this mode 1_B Prohibited
EXT_FUPD	29	rw	**External forced update, but not applicable in SOMC mode.** 0_B Initial value, no function in this mode 1_B Prohibited
SOMB	30	rw	**SOMB mode** ***Note:*** *Set* CLSi_ATOM_CHx_CTRL_SOMC**.**SOMB *= 0 in SOMC mode* 0_B Disable SOMB mode, initial value 1_B Enable SOMB mode, prohibited
FREEZE	31	rw	**ATOM freeze mode enable** 0_B a channel disable/enable may change internal register and output register 1_B a channel enable/disable does not change an internal or output register but stops comparison
0	3, 25, 27, 28	r	**Reserved** Read as 0; must be written with 0.

Table 2344 Reset values of CLSi_ATOM_CHx_CTRL_SOMC (i=0-2;x=0-7)

Reset	Reset value	Note
Kernel Reset	$0000\ 0800_H$	
ATOM channel reset	$0000\ 0800_H$	

44.5.98 ATOM[i] channel [x] control register (x=0-7;i=0-2) (i=0-2;x=0-7)

***Note**: Please see following multiview register description for more details.*

CLSi_ATOM_CHx_CTRL_SOMP (i=0-2;x=0-7)

Offset address: 001804_H+i*20000_H+x*80_H

ATOM[i] channel [x] control register

Reset values see: Table 2345

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FREEZE	SOMB	EXT_FUPD	0	ABM	OSM	SLA	TRIGOUT	EXTTRIGOUT	EXT_TRIG	OSM_TRIG	RST_CCU0	UDMODE		TRIG_PULSE	WR_REQ
rw	rw	rw	r	rw	rw	rw	rw	rw	rw	rw	rw	rw		rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CLK_SRC				SL	EUPM	CMP_CTRL	ACB4	SR0_TRIG	BITREV	0		0	TB12_SEL	MODE	
rwh				rwh	rw	rw	rw	rw	rw	r		r	rw	rw	

Field	Bits	Type	Description
MODE	1:0	rw	**ATOM channel mode select** ***Note**: Set CLSi_ATOM_CHx_CTRL_SOMP.MODE = 10_B for **ATOM** Signal Output Mode PWM (SOMP)* 00_B SOMI mode 01_B SOMC mode 10_B SOMP mode 11_B SOMS mode
TB12_SEL	2	rw	**Select time base value CCM[i]_TBU_TS1 or CCM[i]_TBU_TS2, but not applicable in SOMP mode.** 0_B Initial value, no function in this mode 1_B Prohibited

(table continues...)

(continued)

Field	Bits	Type	Description
BITREV	6	rw	PCM **mode enable** ***Note**:* PCM *mode is only available for odd numbered **ATOM** channels (channel 1,3,5,7).* ***Note**:* PCM *mode is only available in SOMP mode when* CLSi_ATOM_CHx_CTRL_SOMP**.**UDMODE*=0.* **If (((x == 1 || x == 3) || x == 5) || x == 7) &&** CLSi_ATOM_CHx_CTRL_SOMP **.UDMODE == 0:** 0_B PCM mode disabled (no bit-reversing of output of counter register CLSi_ATOM_CHx_CN0.CN0) 1_B PCM mode enabled (bit-reversing of output of counter register CLSi_ATOM_CHx_CN0.CN0) **If (((x !=1 && x !=3) && x !=5) && x !=7) ||** CLSi_ATOM_CHx_CTRL_SOMP **.UDMODE !=0:** 0_B Initial value, no function in this mode 1_B Prohibited
SR0_TRIG	7	rw	**SR0 used for** ATOM_OUT_T **of instance i and channel x** ***Note**:* *This bit may only be set if* CLSi_ATOM_CHx_CTRL_SOMP**.**RST_CCU0 *of this channel is 0.* **If** CLSi_ATOM_CHx_CTRL_SOMP **.RST_CCU0 == 0:** 0_B CLSi_ATOM_CHx_SR0.SR0 is used as a shadow register for register CLSi_ATOM_CHx_CM0.CM0 1_B CLSi_ATOM_CHx_SR0.SR0 is not used as a shadow register for register CLSi_ATOM_CHx_CM0.CM0. CLSi_ATOM_CHx_SR0.SR0 is compared with CLSi_ATOM_CHx_CN0.CN0 and if both are equal, a trigger pulse is generated at output ATOM_OUT_T of instance i and channel x. **If** CLSi_ATOM_CHx_CTRL_SOMP **.RST_CCU0 !=0:** 0_B Initial value, no function in this mode 1_B Prohibited
ACB4	8	rw	**ATOM output control, but not applicable in SOMP mode.** 0_B Initial value, no function in this mode 1_B Prohibited
CMP_CTRL	9	rw	**CCU[x] compare strategy selection, but not applicable in SOMP mode.** 0_B Initial value, no function in this mode 1_B Prohibited
EUPM	10	rw	**Extended update mode, but not applicable in SOMP mode.** 0_B Initial value, no function in this mode 1_B Prohibited

(table continues...)

(continued)

Field	Bits	Type	Description
SL	11	rwh	**Initial signal level.** ***Note**: Reset value depends on the hardware configuration chosen by silicon vendor.* ***Note**: If the output is disabled (ATOM_OUTEN[x:x]=0), the outputs ATOM_OUT[x:x] and ATOM_OUT_T[x:x] of instance i and channel x are set to inverse CLSi_ATOM_CHx_CTRL_SOMP.SL.* ***Note**: If CLSi_ATOM_CHx_CTRL_SOMP.FREEZE=0, when the channel is disabled (ATOM_ENDIS[x:x]=0) and the output is enabled(ATOM_OUTEN[x:x]=1), the output ATOM_OUT[x:x] will be set to !CLSi_ATOM_CHx_CTRL_SOMP.SL and in continuous counting up-down mode with CLSi_ATOM_CHx_CTRL_SOMP.RST_CCU0=1 the other output ATOM_OUT_T[x:x] will be also set to ! CLSi_ATOM_CHx_CTRL_SOMP.SL. Thus, when the channel is re-enabled (ATOM_ENDIS[x:x]=1 && ATOM_OUTEN[x:x]=1) afterwards, the output ATOM_OUT[x:x] is initially !CLSi_ATOM_CHx_CTRL_SOMP.SL and in continuous counting up-down mode with CLSi_ATOM_CHx_CTRL_SOMP.RST_CCU0=1 the other output ATOM_OUT_T[x:x] is also initially ! CLSi_ATOM_CHx_CTRL_SOMP.SL. In other SOMP modes, ATOM_OUT_T[x:x] is not supported.* ***Note**: If CLSi_ATOM_CHx_CTRL_SOMP.FREEZE=1, when the channel is disabled (ATOM_ENDIS[x:x]=0) and the output is enabled(ATOM_OUTEN[x:x]=1), the output ATOM_OUT[x:x] will be not changed and in continuous counting up-down mode with CLSi_ATOM_CHx_CTRL_SOMP.RST_CCU0=1 the other output ATOM_OUT_T[x:x] will also be not changed. Thus, when the channel is re-enabled (ATOM_ENDIS[x:x]=1 && ATOM_OUTEN[x:x]=1) afterwards, the output ATOM_OUT[x:x] is initially the same state as it is before the channel is disabled and in continuous counting up-down mode with CLSi_ATOM_CHx_CTRL_SOMP.RST_CCU0=1 the other output ATOM_OUT_T[x:x] also initially remains the same. In other SOMP modes, ATOM_OUT_T[x:x] is not supported.* 0_B Low signal level 1_B High signal level

(table continues...)

(continued)

Field	Bits	Type	Description
CLK_SRC	15:12	rwh	**CMU clock source for SOMP mode** ***Note**: The register CLSi_ATOM_CHx_CTRL_SOMP.CLK_SRC can be synchronously updated with the value of CLSi_ATOM_CHx_CTRL_SR.CLK_SRC_SR together with the update of register CLSi_ATOM_CHx_CM0.CM0 and CLSi_ATOM_CHx_CM1.CM1.* ***Note**: After (channel) reset the channel operation will run with CLS[j]_CLK. This fact is not reflected by reset value of CLSi_ATOM_CHx_CTRL.CLK_SRC bit field (which is set to 0000_B after channel reset). Only directly writing CLSi_ATOM_CHx_CTRL.CLK_SRC does not change the **CMU** clock source selection. The value read from CLSi_ATOM_CHx_CTRL.CLK_SRC bit field after channel reset does not reflect the current clock source selection status.* ***Note**: After (channel) reset the channel operation will run with CLS[j]_CLK. In order to change the clock source resolution, it is necessary to perform a synchronous update (via CLSi_ATOM_CHx_CTRL_SR.CLK_SRC_SR) before enabling the channel. This can be done by performing the following configuration sequence before enabling the channel:* CLSi_ATOM_CHx_CTRL_SR.CLK_SRC_SR=<selected **CCM** clock source> CLSi_ATOM_CHx_SR0=0_H CLSi_ATOM_CHx_CM0=0_H CLSi_ATOM_AGC_GLB_CTRL.UPEN_CTRLk ***(k=x)*** = 10_H *With this configuration sequence, CLSi_ATOM_CHx_CTRL.CLK_SRC will be immediately updated with the clock source value stored in the shadow register. Afterwards once the channel is enabled, the channel will operate on the selected clock source. Since then, the value read from the register CLSi_ATOM_CHx_CTRL.CLK_SRC reflects the actual selected clock source of operation.* ***Note**: After the register has been updated once from its shadow register after reset state, it is possible to change the clock source also by directly writing CLSi_ATOM_CHx_CTRL.CLK_SRC. That means if the running channel is disabled but not reset afterwards, it is possible to directly configure CLSi_ATOM_CHx_CTRL.CLK_SRC with a new clock source selection.* ***Note**: The clock of the channel is stopped, if "1100_B" value is configured. Restarting a channel by writing a value for a valid clock selection again is possible.*

(table continues...)

(continued)

Field	Bits	Type	Description
			0_H after its first update from the shadow register CLSi_ATOM_CHx_CTRL_SR.CLK_SRC_SR, CCM[i]_CLK_RES[0:0] resolution is selected to be used 1_H after its first update from the shadow register CLSi_ATOM_CHx_CTRL_SR.CLK_SRC_SR, CCM[i]_CLK_RES[1:1] resolution is selected to be used 2_H after its first update from the shadow register CLSi_ATOM_CHx_CTRL_SR.CLK_SRC_SR, CCM[i]_CLK_RES[2:2] resolution is selected to be used 3_H after its first update from the shadow register CLSi_ATOM_CHx_CTRL_SR.CLK_SRC_SR, CCM[i]_CLK_RES[3:3] resolution is selected to be used 4_H after its first update from the shadow register CLSi_ATOM_CHx_CTRL_SR.CLK_SRC_SR, CCM[i]_CLK_RES[4:4] resolution is selected to be used 5_H after its first update from the shadow register CLSi_ATOM_CHx_CTRL_SR.CLK_SRC_SR, CCM[i]_CLK_RES[5:5] resolution is selected to be used 6_H after its first update from the shadow register CLSi_ATOM_CHx_CTRL_SR.CLK_SRC_SR, CCM[i]_CLK_RES[6:6] resolution is selected to be used 7_H after its first update from the shadow register CLSi_ATOM_CHx_CTRL_SR.CLK_SRC_SR, CCM[i]_CLK_RES[7:7] resolution is selected to be used 8_H Prohibited 9_H Prohibited A_H Prohibited B_H Prohibited C_H Functional operation stopped, clock resolution tied to zero D_H after its first update from the shadow register CLSi_ATOM_CHx_CTRL_SR.CLK_SRC_SR, ATOM_CH_TRIGOUT[x-1:x-1] is selected to be used E_H after its first update from the shadow register CLSi_ATOM_CHx_CTRL_SR.CLK_SRC_SR, EXT_TRIGIN is selected to be used F_H Prohibited
WR_REQ	16	rw	**CPU Write request bit for late compare register update, but not applicable in SOMP mode.** 0_B Initial value, no function in this mode 1_B Prohibited

(table continues...)

(continued)

Field	Bits	Type	Description
TRIG_PULSE	17	rw	**Trigger output pulse length of one cluster clock period** ***Attention:*** ***This bit may only be changed if bit*** CLSi_ATOM_CHx_CTRL_SOMP**.**SR0_TRIG ***of this channel is zero.*** **If** CLSi_ATOM_CHx_CTRL_SOMP **.SR0_TRIG == 1:** 0_B output on ATOM_OUT_T[x:x] is '1' as long as CLSi_ATOM_CHx_CN0.CN0=CLSi_ATOM_CHx_SR0.SR0 1_B output on ATOM_OUT_T[x:x] is '1' for only one cluster clock period if CLSi_ATOM_CHx_CN0.CN0=CLSi_ATOM_CHx_SR0.SR0 **If** CLSi_ATOM_CHx_CTRL_SOMP **.SR0_TRIG == 0:** 0_B Initial value, no function in this mode 1_B Prohibited

(table continues...)

(continued)

Field	Bits	Type	Description
UDMODE	19:18	rw	**Up/down counter mode** ***Note**: The update mechanism is enabled by CLSi_ATOM_AGC_GLB_CTRL.UPEN_CTRLk **(k=x)** = 1. Especially, if CLSi_ATOM_CHx_CTRL_SOMP.RST_CCU0 = 0 and CLSi_ATOM_CHx_CTRL_SOMP.UDMODE = 10_B or 11_B, the update may occur when CLSi_ATOM_CHx_CN0.CN0 matches CLSi_ATOM_CHx_CM1.CM1-1. In this case, it is not recommended to update CLSi_ATOM_CHx_CM0.CM0, CLSi_ATOM_CHx_CTRL_SOMP.CLK_SRC and CLSi_ATOM_CHx_CTRL_SOMP.SL with a different value. The behavior after such an update may be unexpected.* 00_B up/down counter mode disabled: CLSi_ATOM_CHx_CN0.CN0 counts always up, CLSi_ATOM_CHx_CM0.CM0, CLSi_ATOM_CHx_CM1.CM1, CLSi_ATOM_CHx_CTRL_SOMP.SL and CLSi_ATOM_CHx_CTRL_SOMP.CLK_SRC are updated if CLSi_ATOM_CHx_CN0.CN0 reaches the value of CLSi_ATOM_CHx_CM0.CM0 -1. 01_B up/down counter mode enabled: CLSi_ATOM_CHx_CN0.CN0 counts up and down, CLSi_ATOM_CHx_CM0.CM0, CLSi_ATOM_CHx_CM1.CM1, CLSi_ATOM_CHx_CTRL_SOMP.SL and CLSi_ATOM_CHx_CTRL_SOMP.CLK_SRC are updated if CLSi_ATOM_CHx_CN0.CN0 is 0 (i.e. the counting direction changes from down to up). 10_B up/down counter mode enabled: CLSi_ATOM_CHx_CN0.CN0 counts up and down, CLSi_ATOM_CHx_CM0.CM0, CLSi_ATOM_CHx_CM1.CM1, CLSi_ATOM_CHx_CTRL_SOMP.SL and CLSi_ATOM_CHx_CTRL_SOMP.CLK_SRC are updated if CLSi_ATOM_CHx_CN0.CN0 reaches the value of CLSi_ATOM_CHx_CM0.CM0 -1 or the channel receives the trigger signal and the counting direction changes from up to down. 11_B up/down counter mode enabled: CLSi_ATOM_CHx_CN0.CN0 counts up and down, CLSi_ATOM_CHx_CM0.CM0, CLSi_ATOM_CHx_CM1.CM1, CLSi_ATOM_CHx_CTRL_SOMP.SL and CLSi_ATOM_CHx_CTRL_SOMP.CLK_SRC are updated if CLSi_ATOM_CHx_CN0.CN0 is 0 or reaches the value of CLSi_ATOM_CHx_CM0.CM0 -1 or the channel receives the trigger signal and the counting direction changes from up to down.

(table continues...)

(continued)

Field	Bits	Type	Description
RST_CCU0	20	rw	**Reset source of CCU0** ***Note**:* *If* CLSi_ATOM_CHx_CTRL_SOMP**.**RST_CCU0*=1 and* CLSi_ATOM_AGC_GLB_CTRL**.**UPEN_CTRLk ***(k=x)****=1 are set,* ATOM_CH_TRIGOUT*[x-1:x-1] or* EXT_TRIGIN*[x:x] triggers also the update of work register (*CLSi_ATOM_CHx_CM0**.**CM0*, *CLSi_ATOM_CHx_CM1**.**CM1*, CLSi_ATOM_CHx_CTRL_SOMP**.**SL *and* CLSi_ATOM_CHx_CTRL_SOMP**.**CLK_SRC*).* ***Note**:* *This bit may only be set if bit* CLSi_ATOM_CHx_CTRL_SOMP**.**OSM*=0 (i.e. in continuous mode)* **If** CLSi_ATOM_CHx_CTRL_SOMP **.OSM == 0:** 0_B Reset counter register CLSi_ATOM_CHx_CN0.CN0 to 0 on matching comparison with CLSi_ATOM_CHx_CM0.CM0 1_B Reset counter register CLSi_ATOM_CHx_CN0.CN0 to 0 on trigger ATOM_CH_TRIGOUT[x-1:x-1] or EXT_TRIGIN **If** CLSi_ATOM_CHx_CTRL_SOMP **.OSM !=0:** 0_B Initial value, no function in this mode 1_B Prohibited
OSM_TRIG	21	rw	**Enable trigger of one-shot pulse by trigger signal** ATOM_CH_TRIGOUT***[x-1:x-1]*** **or** EXT_TRIGIN***[x:x]*** ***Note**:* *This bit may only be set if bit* CLSi_ATOM_CHx_CTRL_SOMP**.**OSM*=1 and bit* CLSi_ATOM_CHx_CTRL_SOMP**.**RST_CCU0*=0.* **If** CLSi_ATOM_CHx_CTRL_SOMP **.OSM == 1 &&** CLSi_ATOM_CHx_CTRL_SOMP **.RST_CCU0 == 0:** 0_B The trigger signal ATOM_CH_TRIGOUT[x-1:x-1] or EXT_TRIGIN cannot trigger start of single pulse generation 1_B The trigger signal ATOM_CH_TRIGOUT[x-1:x-1] or EXT_TRIGIN generation can trigger start of single pulse (only if bit CLSi_ATOM_CHx_CTRL_SOMP.OSM = 1) **If** CLSi_ATOM_CHx_CTRL_SOMP **.OSM !=1 ||** CLSi_ATOM_CHx_CTRL_SOMP **.RST_CCU0 !=0:** 0_B Initial value, no function in this mode 1_B Prohibited
EXT_TRIG	22	rw	**Select** EXT_TRIGIN***[x:x]*** **as trigger signal** 0_B if CLSi_ATOM_CHx_CTRL_SOMP.RST_CCU0=1 or CLSi_ATOM_CHx_CTRL_SOMP.OSM_TRIG=1, signal ATOM_CH_TRIGIN[x:x] is selected as trigger to reset CLSi_ATOM_CHx_CN0.CN0 or to start single pulse generation 1_B if CLSi_ATOM_CHx_CTRL_SOMP.RST_CCU0=1 or CLSi_ATOM_CHx_CTRL_SOMP.OSM_TRIG=1, signal EXT_TRIGIN[x:x] is selected as trigger to reset CLSi_ATOM_CHx_CN0.CN0 or to start single pulse generation

(table continues...)

(continued)

Field	Bits	Type	Description
EXTTRIGOUT	23	rw	**Select** EXT_TRIGIN***[x:x]*** **as potential output signal** ATOM_CH_TRIGOUT***[x:x]*** 0_B signal ATOM_CH_TRIGIN[x:x] is selected as output on ATOM_CH_TRIGOUT[x:x] (if CLSi_ATOM_CHx_CTRL_SOMP.TRIGOUT = 0) 1_B signal EXT_TRIGIN[x:x] after synchronization to the selected clock resolution is selected as output on ATOM_CH_TRIGOUT[x:x] (if CLSi_ATOM_CHx_CTRL_SOMC.TRIGOUT = 0)
TRIGOUT	24	rw	**Trigger output selection (output signal** ATOM_CH_TRIGOUT***[x:x]***) **of module ATOM [i] channel [x].** 0_B ATOM_CH_TRIGOUT[x:x] is ATOM_CH_TRIGIN[x:x] or EXT_TRIGIN[x:x] 1_B ATOM_CH_TRIGOUT[x:x] is ATOM_TRIG_CCU0
SLA	25	rw	**'Serve last' ARU communication strategy, but not applicable in SOMP mode.** 0_B Initial value, no function in this mode 1_B Prohibited
OSM	26	rw	**One-shot mode** 0_B Continuous PWM generation after channel enable 1_B A single pulse is generated
ABM	27	rw	**ARU blocking mode, but not applicable in SOMP mode.** 0_B Initial value, no function in this mode 1_B Prohibited
EXT_FUPD	29	rw	**External forced update** 0_B use ATOM_FUPD[x:x] signal from AGC to force update 1_B use EXT_TRIGIN[x:x] signal to force update
SOMB	30	rw	**SOMB mode** ***Note:*** *Set CLSi_ATOM_CHx_CTRL_SOMP.SOMB = 0 for SOMP mode* 0_B Disable SOMB mode, initial value 1_B Enable SOMB mode, prohibited
FREEZE	31	rw	**ATOM Freeze Mode enable** 0_B a channel disable/enable may change internal register and output register 1_B a channel enable/disable does not change an internal or output register but stops counter CLSi_ATOM_CHx_CN0.CN0
0	3, 5:4, 28	r	**Reserved** Read as 0; must be written with 0.

Table 2345 Reset values of CLSi_ATOM_CHx_CTRL_SOMP (i=0-2;x=0-7)

Reset	Reset value	Note
Kernel Reset	$0000\ 0800_H$	

(table continues...)

Table 2345 (continued) Reset values of CLSi_ATOM_CHx_CTRL_SOMP (i=0-2;x=0-7)

Reset	Reset value	Note
ATOM channel reset	0000 0800_H	

44.5.99 ATOM[i] channel [x] control register (x=0-7;i=0-2) (i=0-2;x=0-7)

***Note**: Please see following multiview register description for more details.*

CLSi_ATOM_CHx_CTRL_SOMS (i=0-2;x=0-7) Offset address: 001804_H+i*20000_H+x*80_H

ATOM[i] channel [x] control register Reset values see: Table 2346

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FREEZE	SOMB	EXT_FUPD	0	ABM	OSM	SLA	TRIGOUT	EXTTRIGOUT	EXT_TRIG	OSM_TRIG	RST_CCU0	UDMODE		TRIG_PULSE	WR_REQ
rw	rw	rw	r	rw	rw	rw	rw	rw	rw	rw	rw	rw		rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CLK_SRC				SL	EUPM	CMP_CTRL	ACB4	DSO	ACB21		ACB0	0	TB12_SEL	MODE	
rwh				rw	rw	rw	rw	rw	rw		rw	r	rw	rw	

Field	Bits	Type	Description
MODE	1:0	rw	**ATOM channel mode select.** ***Note**: Set* CLSi_ATOM_CHx_CTRL_SOMS**.**MODE *= 11_B for **ATOM Signal Output Mode Serial (SOMS)*** 00_B SOMI mode 01_B SOMC mode 10_B SOMP mode 11_B SOMS mode
TB12_SEL	2	rw	**Select time base value** CCM[i]_TBU_TS1 **or** CCM[i]_TBU_TS2**, but not applicable in SOMS mode.** 0_B Initial value, no function in this mode 1_B Prohibited

(table continues...)

(continued)

Field	Bits	Type	Description
ACB0	4	rw	**Shift direction** ***Note**:* *The data that has to be shifted out to be aligned inside the CLSi_ATOM_CHx_CM1.CM1 register according to the defined shift direction.* ***Note**:* *If the direction (CLSi_ATOM_CHx_CTRL_SOMS.ACB0) is changed, the output ATOM_OUT switches immediately to the other 'first' bit of CLSi_ATOM_CHx_CM1.CM1 (bit 0 if CLSi_ATOM_CHx_CTRL_SOMS.ACB0 =0, bit 23 if CLSi_ATOM_CHx_CTRL_SOMS.ACB0 = 1).* **If CLSi_ATOM_CHx_CTRL_SOMS .ARU_EN == 0:** 0_B Right shift of data is started from bit CLSi_ATOM_CHx_CM1.CM1[0:0] 1_B Left shift of data is started from bit CLSi_ATOM_CHx_CM1.CM1[23:23] **If CLSi_ATOM_CHx_CTRL_SOMS .ARU_EN == 1:** 0_B Initial value, no function in this mode 1_B Prohibited
ACB21	6:5	rw	**ATOM output control, but not applicable in SOMS mode.** 00_B Initial value, no function in this mode 01_B Prohibited 10_B Prohibited 11_B Prohibited
DSO	7	rw	**Double shift output** ***Note**:* *If CLSi_ATOM_CHx_CTRL_SOMS.DSO=1 , only shift right is possible.* 0_B CLSi_ATOM_CHx_CM1.CM1 is used as a 24-bit shift register 1_B CLSi_ATOM_CHx_CM1.CM1 is split into two 12-bit shift register
ACB4	8	rw	**ATOM output control, but not applicable in SOMS mode.** 0_B Initial value, no function in this mode 1_B Prohibited
CMP_CTRL	9	rw	**CCU[x] compare strategy selection, but not applicable in SOMS mode.** 0_B Initial value, no function in this mode 1_B Prohibited
EUPM	10	rw	**Extended update mode, but not applicable in SOMS mode.** 0_B Initial value, no function in this mode 1_B Prohibited

(table continues...)

(continued)

Field	Bits	Type	Description
SL	11	rw	**Initial signal level.** ***Note**: Reset value depends on the hardware configuration chosen by silicon vendor.* ***Note**: The inverse value of CLSi_ATOM_CHx_CTRL_SOMS.SL is shifted into the CLSi_ATOM_CHx_CM1.CM1 register.* ***Note**: If the output of **ATOM** i channel x is disabled (ATOM_OUTEN[x:x]=0), both **ATOM** channel outputs ATOM_OUT and ATOM_OUT_T of instance i, channel x are the inverse value of bit CLSi_ATOM_CHx_CTRL.SL.* ***Note**: If the output of **ATOM** i channel x is enabled (ATOM_OUTEN[x:x]=1) but the channel is disabled (ATOM_ENDIS[x:x]=0), the output ATOM_OUT[x:x] will be set as CLSi_ATOM_CHx_CM1.CM1[0:0] when configured as right shift or CLSi_ATOM_CHx_CM1.CM1[23:23] when configured as left shift. Afterwards when the channel is re-enabled (ATOM_OUTEN[x:x]=1 && ATOM_ENDIS[x:x]=1), the output ATOM_OUT[x:x] is initially CLSi_ATOM_CHx_CM1.CM1[0:0] or CLSi_ATOM_CHx_CM1.CM1[23:23].* ***Note**: The output ATOM_OUT_T[x:x] is supported in double shift mode but not supported in single shift mode. In double shift mode, if CLSi_ATOM_CHx_CTRL_SOMS.FREEZE=0, when the output of **ATOM** i channel x is enabled (ATOM_OUTEN[x:x]=1) but the channel is disabled (ATOM_ENDIS[x:x]=0), the output ATOM_OUT_T[x:x] is set to !CLSi_ATOM_CHx_CTRL.SL. Afterwards when the channel is re-enabled (ATOM_OUTEN[x:x]=1 && ATOM_ENDIS[x:x]=1), the output ATOM_OUT_T[x:x] that is initially !CLSi_ATOM_CHx_CTRL.SL.* The output ATOM_OUT_T[x:x] is supported in double shift mode but not supported in single shift mode. In double shift mode, if CLSi_ATOM_CHx_CTRL_SOMS.FREEZE=1, when the output of **ATOM** i channel x is enabled (ATOM_OUTEN[x:x]=1) but the channel is disabled (ATOM_ENDIS[x:x]=0), the output ATOM_OUT_T[x:x] will not be changed. Afterwards when the channel is re-enabled (ATOM_OUTEN[x:x]=1 && ATOM_ENDIS[x:x]=1), the output ATOM_OUT_T[x:x] that is initially the same state as it is when deactivated(ATOM_ENDIS[x:x]=0) last time. 0_B Low signal level 1_B High signal level

(table continues...)

(continued)

Field	Bits	Type	Description
CLK_SRC	15:12	rwh	**CMU clock source for SOMS mode** ***Note**: The register CLSi_ATOM_CHx_CTRL_SOMS.CLK_SRC is updated with the value of CLSi_ATOM_CHx_CTRL_SR.CLK_SRC_SR together with the update of register CLSi_ATOM_CHx_CM0.CM0 and CLSi_ATOM_CHx_CM1.CM1.* ***Note**: After (channel) reset the selected clock source value is the cluster clock (input of Global Clock Divider).* ***Note**: The clock of the channel is stopped, if "1100_B" value is configured. Restarting a channel by writing a value for a valid clock selection again is possible.* 0_H CCM[i]_CLK_RES[0:0] resolution in use 1_H CCM[i]_CLK_RES[1:1] resolution in use 2_H CCM[i]_CLK_RES[2:2] resolution in use 3_H CCM[i]_CLK_RES[3:3] resolution in use 4_H CCM[i]_CLK_RES[4:4] resolution in use 5_H CCM[i]_CLK_RES[5:5] resolution in use 6_H CCM[i]_CLK_RES[6:6] resolution in use 7_H CCM[i]_CLK_RES[7:7] resolution in use 8_H Prohibited 9_H Prohibited A_H Prohibited B_H Prohibited C_H Functional operation stopped, clock resolution tied to zero D_H ATOM_CH_TRIGOUT[x-1:x-1] selected E_H EXT_TRIGIN selected F_H Prohibited
WR_REQ	16	rw	**CPU Write request bit for late compare register update, but not applicable in SOMS mode.** 0_B Initial value, no function in this mode 1_B Prohibited
TRIG_PULSE	17	rw	**Trigger output pulse length of one cluster clock period, , but not applicable in SOMS mode.** 0_B Initial value, no function in this mode 1_B Prohibited
UDMODE	19:18	rw	**Up/down counter mode, but not applicable in SOMS mode.** 00_B Initial value, no function in this mode 01_B Prohibited 10_B Prohibited 11_B Prohibited

(table continues...)

(continued)

Field	Bits	Type	Description
RST_CCU0	20	rw	**Reset source of CCU0, but not applicable in SOMS mode.** 0_B Initial value, no function in this mode 1_B Prohibited
OSM_TRIG	21	rw	**Enable trigger of one-shot pulse by trigger signal ATOM_CH_TRIGOUT*[x-1:x-1]* or EXT_TRIGIN*[x:x]*, but not applicable in SOMS mode.** 0_B Initial value, no function in this mode 1_B Prohibited
EXT_TRIG	22	rw	**Select EXT_TRIGIN*[x:x]* as trigger signal, but not applicable in SOMS mode.** 0_B Initial value, no function in this mode 1_B Prohibited
EXTTRIGOUT	23	rw	**Select EXT_TRIGIN*[x:x]* as potential output signal ATOM_CH_TRIGOUT*[x:x]*** 0_B signal ATOM_CH_TRIGIN[x:x] is selected as output on ATOM_CH_TRIGOUT[x:x] (if CLSi_ATOM_CHx_CTRL_SOMS.TRIGOUT = 0) 1_B signal EXT_TRIGIN after synchronization to the selected clock resolution is selected as output on ATOM_CH_TRIGOUT[x:x] (if CLSi_ATOM_CHx_CTRL_SOMS.TRIGOUT = 0)
TRIGOUT	24	rw	**Trigger output selection (output signal ATOM_CH_TRIGOUT*[x:x]*) of module ATOM channel x.** 0_B ATOM_CH_TRIGOUT[x:x] is ATOM_CH_TRIGIN[x:x] or EXT_TRIGIN[x:x] 1_B ATOM_CH_TRIGOUT[x:x] is ATOM_TRIG_CCU0
SLA	25	rw	**'Serve last' ARU communication strategy, but not applicable in SOMS mode.** 0_B Initial value, no function in this mode 1_B Prohibited
OSM	26	rw	**One-shot mode** 0_B Continuous PWM generation after channel enable 1_B A single pulse is generated
ABM	27	rw	**ARU blocking mode, but not applicable in SOMS mode.** 0_B Initial value, no function in this mode 1_B Prohibited
EXT_FUPD	29	rw	**External forced update** 0_B use ATOM_FUPD[x:x] signal from AGC to force update 1_B use EXT_TRIGIN signal to force update

(table continues...)

(continued)

Field	Bits	Type	Description
SOMB	30	rw	**SOMB mode** ***Note:*** *Set CLSi_ATOM_CHx_CTRL_SOMS.SOMB = 0 for SOMS mode.* 0_B Disable SOMB mode, initial value 1_B Enable SOMB mode, prohibited
FREEZE	31	rw	**ATOM Freeze Mode enable** ***Note:*** *If channel is disabled and output is enabled, the output depends directly on CLSi_ATOM_CHx_CTRL_SOMS.SL bit, irrespective of CLSi_ATOM_CHx_CTRL_SOMS.FREEZE mode.*
0	3, 28	r	**Reserved** Read as 0; must be written with 0.

Table 2346 Reset values of CLSi_ATOM_CHx_CTRL_SOMS (i=0-2;x=0-7)

Reset	Reset value	Note
Kernel Reset	0000 0800_H	
ATOM channel reset	0000 0800_H	

44.5.100 ATOM[i] channel [x] control register (x=0-7;i=0-2) (i=0-2;x=0-7)

Note: *Please see following multiview register description for more details.*

CLSi_ATOM_CHx_CTRL_SOMB (i=0-2;x=0-7) Offset address: 001804_H+i*20000_H+x*80_H

ATOM[i] channel [x] control register Reset values see: Table 2347

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FREEZE	SOMB	EXT_FUPD	0	0	OSM	SLA	TRIGOUT	EXTTRIGOUT	EXT_TRIG	OSM_TRIG	RST_CCU0	UDMODE		TRIG_PULSE	WR_REQ
rw	rw	rw	r	r	rw	rw	rw	rw	rw	rw	rw	rw		rw	rwh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CLK_SRC				SL	EUPM	CMP_CTRL	ACB42			ACB10		0	TB12_SEL	MODE	
rw				rw	rw	rw	rw			rw		r	rw	rw	

Field	Bits	Type	Description
MODE	1:0	rw	**ATOM channel mode select, but not applicable in SOMB mode.** ***Note**:* *Set CLSi_ATOM_CHx_CTRL_SOMB.MODE = 00_B for **ATOM** Signal Output Mode Compare (SOMB)* 00_B Initial value, no function in this mode 01_B Prohibited 10_B Prohibited 11_B Prohibited
TB12_SEL	2	rw	**Select time base value CCM[i]_TBU_TS1 or CCM[i]_TBU_TS2.** ***Note**:* *This bit is only applicable if three time bases are present in the **eGTM**. Otherwise, this bit is reserved.* 0_B CCM[i]_TBU_TS1 selected for comparison 1_B CCM[i]_TBU_TS2 selected for comparison
ACB10	5:4	rw	**Signal level control** ***Note**:* *For details see Table 2235* **If CLSi_ATOM_CHx_CTRL_SOMB .ARU_EN == 0:** 00_B No signal level change at output 01_B Set output signal level to 1 when CLSi_ATOM_CHx_CTRL_SOMB.SL bit = 0 else output signal level to 0 10_B Set output signal level to 0 when CLSi_ATOM_CHx_CTRL_SOMB.SL bit = 0 else output signal level to 1 11_B Toggle output signal level **If CLSi_ATOM_CHx_CTRL_SOMB .ARU_EN !=0:** 00_B Initial value, no function in this mode 01_B Prohibited 10_B Prohibited 11_B Prohibited

(table continues...)

(continued)

Field	Bits	Type	Description
ACB42	8:6	rw	**Compare strategy** ***Note**: For details see Table 2234* **If** CLSi_ATOM_CHx_CTRL_SOMB **.ARU_EN == 0:** 000_B Reserved. Has no effect. 001_B Reserved. Has no effect. 010_B Compare in CCU0 only against CCM[i]_TBU_TS0 011_B Compare in CCU1 only against CCM[i]_TBU_TS1 or CCM[i]_TBU_TS2 100_B Compare first in CCU0 and then in CCU1. Use CCM[i]_TBU_TS0. 101_B Compare first in CCU0 and then in CCU1. Use CCM[i]_TBU_TS1 or CCM[i]_TBU_TS2. 110_B Compare first in CCU0 and then in CCU1. Use CCM[i]_TBU_TS0 in CCU0 and CCM[i]_TBU_TS1 or CCM[i]_TBU_TS2 in CCU1. 111_B Cancel pending comparisons **If** CLSi_ATOM_CHx_CTRL_SOMB **.ARU_EN !=0:** 000_B Initial value, no function in this mode 001_B Prohibited 010_B Prohibited 011_B Prohibited 100_B Prohibited 101_B Prohibited 110_B Prohibited 111_B Cancel pending compare events
CMP_CTRL	9	rw	**CCU[x] compare strategy select.** ***Note**: If the selected **TBU** time base is CCM[i]_TBU_TS0 that can only be up-counting, cyclic event compare of the selected time base CCM[i]_TBU_TS0 against CLSi_ATOM_CHx_CM0**.**CM0/ CLSi_ATOM_CHx_CM1**.**CM1 is performed independent of CLSi_ATOM_CHx_CTRL_SOMB**.**CMP_CTRL.* 0_B Up-counting: Cyclic event compare of the selected time base (CCM[i]_TBU_TS1/CCM[i]_TBU_TS2) against CLSi_ATOM_CHx_CM0.CM0/CLSi_ATOM_CHx_CM1.CM1 is performed when the selected TBU time base (CCM[i]_TBU_TS1/ CCM[i]_TBU_TS2) is up-counting. Please refer to Cyclic event compare for more information about cyclic event compare strategy. 1_B Down-counting: Cyclic event compare of the selected time base (CCM[i]_TBU_TS1/CCM[i]_TBU_TS2) against CLSi_ATOM_CHx_CM0.CM0/CLSi_ATOM_CHx_CM1.CM1 is performed when the selected TBU time base (CCM[i]_TBU_TS1/ CCM[i]_TBU_TS2) is down-counting. Please refer to Cyclic event compare for more information about cyclic event compare strategy.

(table continues...)

(continued)

Field	Bits	Type	Description
EUPM	10	rw	**Extended update mode** ***Note**:* *If CLSi_ATOM_CHx_CTRL_SOMB.EUPM=1 a write-access to CLSi_ATOM_CHx_CM0.CM0 or CLSi_ATOM_CHx_CM1.CM1 never causes an **AEI** write status 10_B.* 0_B No extended update of CLSi_ATOM_CHx_CM0.CM0 and CLSi_ATOM_CHx_CM1.CM1 1_B Extended update mode in case of compare strategy 'serve last': update of CLSi_ATOM_CHx_CM1.CM1 after CCU0 compare match possible.
SL	11	rw	**Initial signal level.** ***Note**:* *Reset value depends on the hardware configuration chosen by silicon vendor.* ***Note**:* *If the output is disabled (ATOM_OUTEN[x:x]=0), the outputs ATOM_OUT[x:x] and ATOM_OUT_T[x:x] of instance i and channel x are set to !CLSi_ATOM_CHx_CTRL_SOMB.SL. It is independent of CLSi_ATOM_CHx_CTRL_SOMB.FREEZE.* ***Note**:* *If CLSi_ATOM_CHx_CTRL_SOMB.FREEZE=0, when the channel is disabled (ATOM_ENDIS[x:x]=0) but the output is still enabled (ATOM_OUTEN[x:x]=1), the output ATOM_OUT[x:x] is set to the value of CLSi_ATOM_CHx_CTRL_SOMB.SL. Thus, if the channel is enabled afterwards (ATOM_ENDIS[x:x]=1 && ATOM_OUTEN[x:x]=1), the output ATOM_OUT[x:x] is initially CLSi_ATOM_CHx_CTRL_SOMB.SL. ATOM_OUT_T[x:x] is not supported in SOMB mode.* ***Note**:* *If CLSi_ATOM_CHx_CTRL_SOMB.FREEZE=1, when the channel is disabled (ATOM_ENDIS[x:x]=0) but the output is still enabled (ATOM_OUTEN[x:x]=1), the output ATOM_OUT[x:x] is not changed. Thus, if the channel is enabled afterwards (ATOM_ENDIS[x:x]=1 && ATOM_OUTEN[x:x]=1), the output ATOM_OUT[x:x] is initially the same state as it is before the channel is disabled. ATOM_OUT_T[x:x] is not supported in SOMB mode.* 0_B Low signal level 1_B High signal level

(table continues...)

(continued)

Field	Bits	Type	Description
CLK_SRC	15:12	rw	**CMU clock source but not applicable in SOMB mode** 0_H Initial value, no function in this mode 1_H Prohibited 2_H Prohibited 3_H Prohibited 4_H Prohibited 5_H Prohibited 6_H Prohibited 7_H Prohibited 8_H Prohibited 9_H Prohibited A_H Prohibited B_H Prohibited C_H Prohibited D_H Prohibited E_H Prohibited F_H Prohibited
WR_REQ	16	rwh	**CPU Write request bit for late compare register update.** ***Note**:* *With a compare match event or 2 compare match events at "serve last" strategy, the CLSi_ATOM_CHx_CTRL_SOMC.WR_REQ bit will be reset by hardware.* ***Note**:* *At the point of the force update only the shadow registers CLSi_ATOM_CHx_SR0.SR0 and CLSi_ATOM_CHx_SR1.SR1 are transferred into the CLSi_ATOM_CHx_CM0.CM0, CLSi_ATOM_CHx_CM1.CM1 registers. The output action is still defined by the CLSi_ATOM_CHx_STAT.ACBI bit field described by the **ARU** together with the old compare values for CLSi_ATOM_CHx_CM0.CM0/CLSi_ATOM_CHx_CM1.CM1.* 0_B No late update requested by writing via configuration interface 1_B Late update requested by writing via configuration interface
TRIG_PULSE	17	rw	**Trigger output pulse length of one cluster clock period, , but not applicable in SOMB mode.** 0_B Initial value, no function in this mode 1_B Prohibited
UDMODE	19:18	rw	**Up/down counter mode, but not applicable in SOMB mode.** 00_B Initial value, no function in this mode 01_B Prohibited 10_B Prohibited 11_B Prohibited
RST_CCU0	20	rw	**Reset source of CCU0, but not applicable in SOMB mode.** 0_B Initial value, no function in this mode 1_B Prohibited

(table continues...)

(continued)

Field	Bits	Type	Description
OSM_TRIG	21	rw	**Enable trigger of one-shot pulse by trigger signal ATOM_CH_TRIGOUT*[x-1:x-1]* or EXT_TRIGIN*[x:x]*, but not applicable in SOMB mode.** ***Note**: This bit may only be set if bit CLSi_ATOM_CHx_CTRL_SOMP.OSM=1 and bit CLSi_ATOM_CHx_CTRL_SOMP.RST_CCU0=0.* 0_B Initial value, no function in this mode 1_B Prohibited
EXT_TRIG	22	rw	**Select EXT_TRIGIN*[x:x]* as trigger signal, but not applicable in SOMB mode.** 0_B Initial value, no function in this mode 1_B Prohibited
EXTTRIGOUT	23	rw	**Select EXT_TRIGIN*[x:x]* as potential output signal ATOM_CH_TRIGOUT*[x:x]*** 0_B signal ATOM_CH_TRIGIN[x:x] is selected as output on ATOM_CH_TRIGOUT[x:x] (if CLSi_ATOM_CHx_CTRL_SOMB.TRIGOUT = 0) 1_B signal EXT_TRIGIN after synchronization to the selected clock resolution is selected as output on ATOM_CH_TRIGOUT[x:x] (if CLSi_ATOM_CHx_CTRL_SOMB.TRIGOUT = 0)
TRIGOUT	24	rw	**Trigger output selection (output signal ATOM_CH_TRIGOUT*[x:x]*) of module ATOM [i] channel [x].** 0_B ATOM_CH_TRIGOUT[x:x] is ATOM_CH_TRIGIN[x:x] or EXT_TRIGIN[x:x] 1_B ATOM_CH_TRIGOUT[x:x] is ATOM_TRIG_CCU0
SLA	25	rw	**'Serve last' ARU communication strategy, but not applicable in SOMP mode.** 0_B Initial value, no function in this mode 1_B Prohibited
OSM	26	rw	**One-shot mode, but not applicable in SOMB mode.** 0_B Initial value, no function in this mode 1_B Prohibited
EXT_FUPD	29	rw	**External forced update** 0_B use ATOM_FUPD[x:x] signal from AGC to force update 1_B use EXT_TRIGIN[x:x] signal to force update
SOMB	30	rw	**SOMB mode** ***Note**: Set CLSi_ATOM_CHx_CTRL_SOMB.SOMB = 1 for **ATOM** SOMB mode.* 0_B Disable SOMB mode 1_B Enable SOMB mode

(table continues...)

(continued)

Field	Bits	Type	Description
FREEZE	31	rw	**ATOM Freeze Mode enable** 0_B a channel disable/enable may change internal register and output register 1_B a channel enable/disable does not change an internal or output register but stops comparison
0	3, 27, 28	r	**Reserved** Read as 0; must be written with 0.

Table 2347 Reset values of CLSi_ATOM_CHx_CTRL_SOMB (i=0-2;x=0-7)

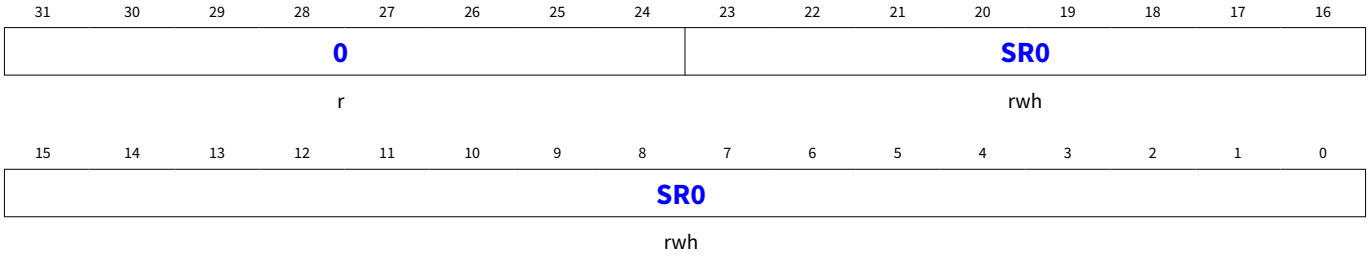
Reset	Reset value	Note
Kernel Reset	0000 0800_H	
ATOM channel reset	0000 0800_H	

44.5.101 ATOM[i] channel [x] CCU0 compare shadow register (x=0-7;i=0-2)

***Note**: When in SOMC mode a compare event occurs, this register will be write-protected and return* AEI_STATUS*=10_B with write-access until the register has been read out via **ARU** or the configuration interface.*

CLSi_ATOM_CHx_SR0 (i=0-2;x=0-7) Offset address: 001808_H+i*20000_H+x*80_H

ATOM[i] channel [x] CCU0 compare shadow register Reset values see: Table 2348



Field	Bits	Type	Description
SR0	23:0	rwh	**ATOM channel [x] shadow register SR0.** ***Note**: The* CLSi_ATOM_CHx_SR0**.**SR0 *register is used as shadow register for* CLSi_ATOM_CHx_CM0**.**CM0 *in SOMP and SOMS modes and is used as capture register for time base* CCM[i]_TBU_TS0 *in SOMC mode.*
0	31:24	r	**Reserved** Read as 0; must be written with 0.

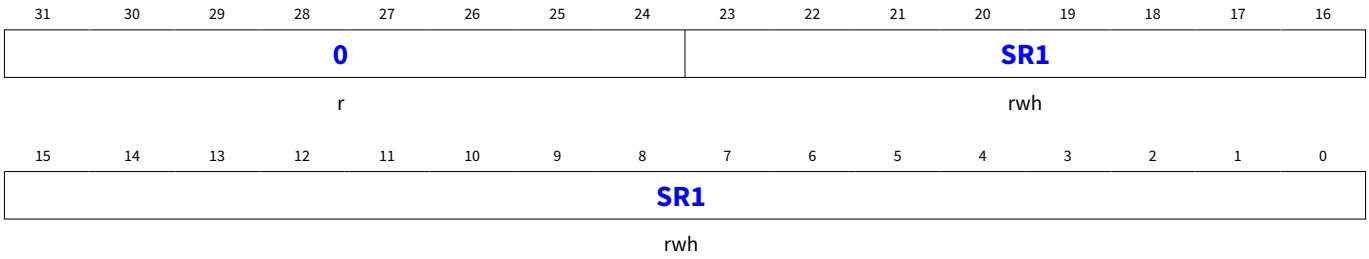
Table 2348 Reset values of CLSi_ATOM_CHx_SR0 (i=0-2;x=0-7)

Reset	Reset value	Note
Kernel Reset	0000 0000$_H$	
ATOM channel reset	0000 0000$_H$	

44.5.102 ATOM[i] channel [x] CCU1 compare shadow register (x=0-7;i=0-2)

CLSi_ATOM_CHx_SR1 (i=0-2;x=0-7) Offset address: 00180C$_H$+i*20000$_H$+x*80$_H$

ATOM[i] channel [x] CCU1 compare shadow register Reset values see: Table 2349



Field	Bits	Type	Description
SR1	23:0	rwh	**ATOM channel [x] shadow register SR1.** ***Note**:* *The CLSi_ATOM_CHx_SR1.SR1 register is used as shadow register for CLSi_ATOM_CHx_CM1.CM1 in SOMP and SOMS modes and is used as capture register for time base CCM[i]_TBU_TS1 or CCM[i]_TBU_TS2 (as selected in CLSi_ATOM_CHx_CTRL register) in SOMC mode.* ***Note**:* *When in SOMC mode a compare event occurs, this register will be write-protected and return AEI_STATUS=10$_B$ with write-access until the register has been read out via **ARU** or the configuration interface.*
0	31:24	r	**Reserved** Read as 0; must be written with 0.

Table 2349 Reset values of CLSi_ATOM_CHx_SR1 (i=0-2;x=0-7)

Reset	Reset value	Note
Kernel Reset	0000 0000$_H$	
ATOM channel reset	0000 0000$_H$	

44.5.103 ATOM[i] channel [x] CCU0 compare register (x=0-7;i=0-2)

***Note**: If the channel is configured in SOMC mode 'serve last' compare strategy with CLSi_ATOM_CHx_CTRL.EUPM=0, when the first compare event in CCU0 occurs, the register will be write-protected and return AEI_STATUS=10_B with write-access until the second compare event in CCU1.*

CLSi_ATOM_CHx_CM0 (i=0-2;x=0-7) Offset address: 001810_H+i*20000_H+x*80_H

ATOM[i] channel [x] CCU0 compare register Reset values see: Table 2350

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								CM0							
r								rwh							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CM0															
rwh															

Field	Bits	Type	Description
CM0	23:0	rwh	**ATOM CCU0 compare register.**
0	31:24	r	**Reserved** Read as 0; must be written with 0.

Table 2350 Reset values of CLSi_ATOM_CHx_CM0 (i=0-2;x=0-7)

Reset	Reset value	Note
Kernel Reset	$0000\ 0000_H$	
ATOM channel reset	$0000\ 0000_H$	

44.5.104 ATOM[i] channel [x] CCU1 compare register (x=0-7;i=0-2)

***Note**: If the channel is configured in SOMC mode 'serve last' compare strategy with CLSi_ATOM_CHx_CTRL.EUPM=0, when the first compare event in CCU0 occurs, the register will be write-protected and return AEI_STATUS=10_B with write-access until the second compare event in CCU1.*

CLSi_ATOM_CHx_CM1 (i=0-2;x=0-7) Offset address: 001814_H+i*20000_H+x*80_H

ATOM[i] channel [x] CCU1 compare register Reset values see: Table 2351

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								CM1							
r								rwh							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CM1															
rwh															

Field	Bits	Type	Description
CM1	23:0	rwh	**ATOM CCU1 compare register.**
0	31:24	r	**Reserved** Read as 0; must be written with 0.

Table 2351 Reset values of CLSi_ATOM_CHx_CM1 (i=0-2;x=0-7)

Reset	Reset value	Note
Kernel Reset	0000 0000$_H$	
ATOM channel reset	0000 0000$_H$	

44.5.105 ATOM[i] channel [x] CCU0 counter register (x=0-7;i=0-2)

CLSi_ATOM_CHx_CN0 (i=0-2;x=0-7) Offset address: 001818$_H$+i*20000$_H$+x*80$_H$

ATOM[i] channel [x] CCU0 counter register Reset values see: Table 2352

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								**CN0**							
r								rwh							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CN0															
rwh															

Field	Bits	Type	Description
CN0	23:0	rwh	**ATOM CCU0 counter register.**
0	31:24	r	**Reserved** Read as 0; must be written with 0.

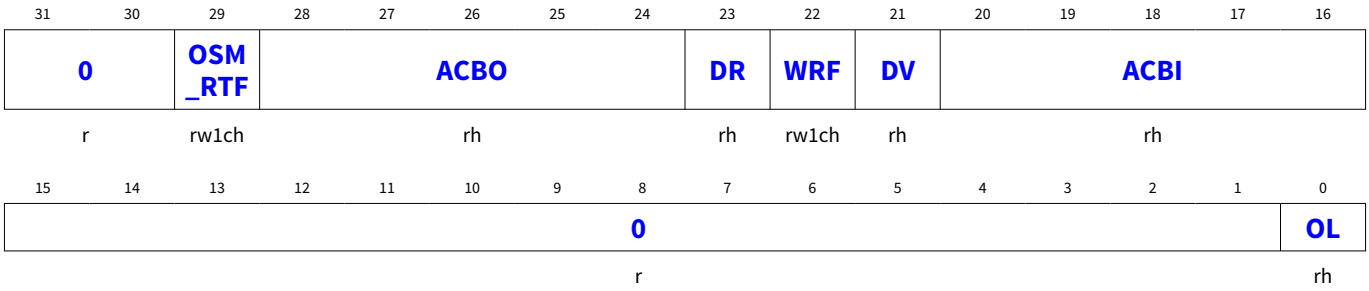
Table 2352 Reset values of CLSi_ATOM_CHx_CN0 (i=0-2;x=0-7)

Reset	Reset value	Note
Kernel Reset	0000 0000$_H$	
ATOM channel reset	0000 0000$_H$	

44.5.106 ATOM[i] channel [x] status register (x=0-7;i=0-2)

CLSi_ATOM_CHx_STAT (i=0-2;x=0-7) Offset address: 00181C$_H$+i*20000$_H$+x*80$_H$

ATOM[i] channel [x] status register Reset values see: Table 2353



Field	Bits	Type	Description
OL	0	rh	**Output signal level of** ATOM_OUT***[x:x]*****.** ***Note**: Reset value is the inverted value of bit CLSi_ATOM_CHx_CTRL.SL which depends on the hardware configuration chosen by silicon vendor.* 0_B Output signal level is low 1_B Output signal level is high
ACBI	20:16	rh	**ATOM Mode control bits.** ***Note**: For SOMB mode this bit field serves as the work register of the compare strategy. It can be updated with the value of bit field CLSi_ATOM_CHx_CTRL.ACB or the value of internal shadow register ACB_SR.*

(table continues...)

(continued)

Field	Bits	Type	Description
DV	21	rh	**Valid data stored in compare registers.** ***Note**: In SOMI mode, the bit is not supported (CLSi_ATOM_CHx_STAT.DV = 0).* **If reading and CLSi_ATOM_CHx_CTRL .MODE !=0 || CLSi_ATOM_CHx_CTRL .SOMB !=0:** 0_B In SOMC/SOMB mode, no valid data is stored in registers CLSi_ATOM_CHx_CM0.CM0 and/or CLSi_ATOM_CHx_CM1.CM1 and CLSi_ATOM_CHx_CTRL.ACB/CLSi_ATOM_CHx_STAT.ACBI. Therefore when the channel is enabled, no comparison is activated. In SOMP/SOMS mode, no valid ARU data stored in register CLSi_ATOM_CHx_SR0.SR0 and/or CLSi_ATOM_CHx_SR1.SR1 and CLSi_ATOM_CHx_STAT.ACBI. 1_B In SOMC/SOMB mode, new valid data is stored in registers CLSi_ATOM_CHx_CM0.CM0 and/or CLSi_ATOM_CHx_CM1.CM1 and CLSi_ATOM_CHx_CTRL.ACB/CLSi_ATOM_CHx_STAT.ACBI. Therefore when the channel is enabled, comparison can be activated depending on CLSi_ATOM_CHx_CTRL.ACB[4:2]/ CLSi_ATOM_CHx_STAT.ACBI[4:2]. In SOMP/SOMS mode, new valid ARU data stored in register CLSi_ATOM_CHx_SR0.SR0 and/or CLSi_ATOM_CHx_SR1.SR1 and CLSi_ATOM_CHx_STAT.ACBI. **If reading and CLSi_ATOM_CHx_CTRL .MODE == 0 && CLSi_ATOM_CHx_CTRL .SOMB == 0:** 0_B Initial value, no function in this mode 1_B Prohibited
WRF	22	rw1ch	**Write request of CPU failed for late update.** ***Note**: The bit CLSi_ATOM_CHx_STAT.WRF can be cleared by writing a 1 to it.* ***Note**: This bit is only applicable in SOMC and SOMB mode.* **If reading:** 0_B Late update was successful, CCU0/CCU1 units wait for comparison 1_B Late update failed **If writing:** 0_B No action 1_B Clear status bit to zero
DR	23	rh	**Data rejected flag** ***Note**: The flag is cleared if valid data is received and stored.* 0_B Received data stored 1_B Received data rejected

(table continues...)

(continued)

Field	Bits	Type	Description
ACBO	28:24	rh	**ATOM Internal status bits.** ***Note**: These bits are only set in SOMC mode.* ***Note**: CLSi_ATOM_CHx_STAT.ACBO is reset to 00000_B on an update of register CLSi_ATOM_CHx_CM0.CM0 or CLSi_ATOM_CHx_CM1.CM1* **If reading and CLSi_ATOM_CHx_CTRL .MODE == 1 && CLSi_ATOM_CHx_CTRL .SOMB == 0:** 00_H no compare match occurred 08_H CCU0 compare match occurred 10_H CCU1 compare match occurred **If reading and CLSi_ATOM_CHx_CTRL .MODE !=1 || CLSi_ATOM_CHx_CTRL .SOMB == 1:** 00_H Initial value, no function in this mode 08_H Prohibited 10_H Prohibited **others,** Reserved
OSM_RTF	29	rw1ch	**One-shot mode retrigger failed flag.** ***Note**: This bit will be cleared by writing 1 on the configuration interface. A read-access leaves the bit unchanged.* ***Note**: This bit is only set in SOMP mode.* **If CLSi_ATOM_CHx_CTRL .MODE == 2 && CLSi_ATOM_CHx_CTRL .SOMB == 0:** 0_B One-shot retrigger operation successful 1_B One-shot retrigger operation failed **If CLSi_ATOM_CHx_CTRL .MODE !=2 || CLSi_ATOM_CHx_CTRL .SOMB != 0:** 0_B Initial value, no function in this mode 1_B Prohibited
0	15:1, 31:30	r	**Reserved** Read as 0; must be written with 0.

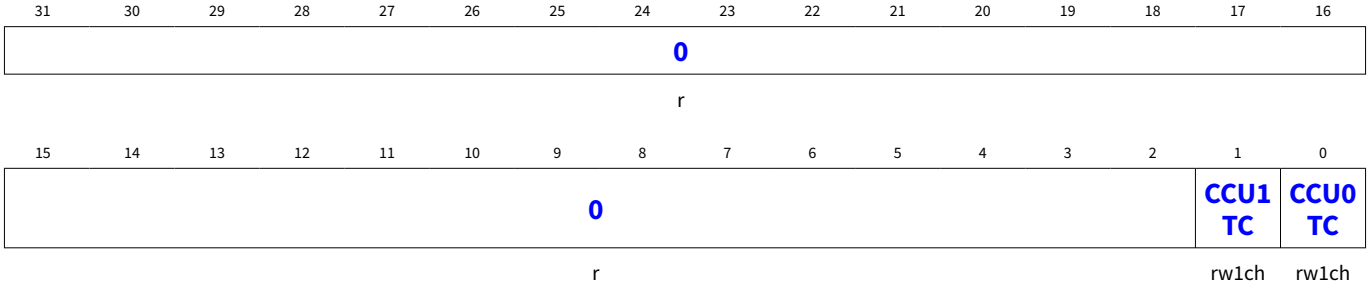
Table 2353 Reset values of CLSi_ATOM_CHx_STAT (i=0-2;x=0-7)

Reset	Reset value	Note
Kernel Reset	$0000\ 0000_H$	
ATOM channel reset	$0000\ 0000_H$	

44.5.107 ATOM[i] channel [x] interrupt notification register (x=0-7;i=0-2)

CLSi_ATOM_CHx_IRQ_NOTIFY (i=0-2;x=0-7) Offset address: 001820_H+i*20000_H+x*80_H

ATOM[i] channel [x] interrupt notification register Reset values see: Table 2354



Field	Bits	Type	Description
CCU0TC	0	rw1ch	**CCU0 Trigger condition interrupt for channel [x].** ***Note**: This bit will be cleared on a configuration interface write-access of value 1. A read-access leaves the bit unchanged.* **If reading:** 0_B No interrupt triggered by software or by the compare event in CCU0 unit was raised 1_B Interrupt triggered by software or by the compare event in CCU0 unit was raised **If writing:** 0_B No action 1_B Clear interrupt
CCU1TC	1	rw1ch	**CCU Trigger condition interrupt for channel [x].** ***Note**: This bit will be cleared on a configuration interface write-access of value 1. A read-access leaves the bit unchanged.* ***Note**: If bit CLSi_ATOM_CHx_CTRL_SOMP.SR0_TRIG is set to 1 (only valid in SOMP mode), this interrupt notify flag CLSi_ATOM_CHx_IRQ_NOTIFY.CCU1TC is set in case CLSi_ATOM_CHx_SR0.SR0 is equal to CLSi_ATOM_CHx_CN0.CN0 and not set in case CLSi_ATOM_CHx_CM1.CM1 is equal to CLSi_ATOM_CHx_CN0.CN0.* **If reading:** 0_B No interrupt triggered by software or by the compare event in CCU1 unit was raised 1_B Interrupt triggered by software or by the compare event in CCU1 unit was raised **If writing:** 0_B No action 1_B Clear interrupt

(table continues...)

(continued)

Field	Bits	Type	Description
0	31:2	r	**Reserved** Read as 0; must be written with 0.

Table 2354 Reset values of CLSi_ATOM_CHx_IRQ_NOTIFY (i=0-2;x=0-7)

Reset	Reset value	Note
Kernel Reset	0000 0000$_H$	
ATOM channel reset	0000 0000$_H$	

44.5.108 ATOM[i] channel [x] interrupt enable register (x=0-7;i=0-2)

CLSi_ATOM_CHx_IRQ_EN (i=0-2;x=0-7) Offset address: 001824$_H$+i*20000$_H$+x*80$_H$

ATOM[i] channel [x] interrupt enable register Reset values see: Table 2355

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0														CCU1TC_IRQ_EN	CCU0TC_IRQ_EN
r														rw	rw

Field	Bits	Type	Description
CCU0TC_IRQ_EN	0	rw	ATOM_CCU0TC[x]_IRQ **interrupt enable.** 0_B the interrupt signal ATOM_CCU0TC[x]_IRQ is not visible outside eGTM-IP 1_B the interrupt signal ATOM_CCU0TC[x]_IRQ is visible outside eGTM-IP
CCU1TC_IRQ_EN	1	rw	ATOM_CCU1TC[x]_IRQ **interrupt enable.** 0_B the interrupt signal ATOM_CCU1TC[x]_IRQ is not visible outside eGTM-IP 1_B the interrupt signal ATOM_CCU1TC[x]_IRQ is visible outside eGTM-IP
0	31:2	r	**Reserved** Read as 0; must be written with 0.

Table 2355 Reset values of CLSi_ATOM_CHx_IRQ_EN (i=0-2;x=0-7)

Reset	Reset value	Note
Kernel Reset	0000 0000$_H$	

(table continues...)

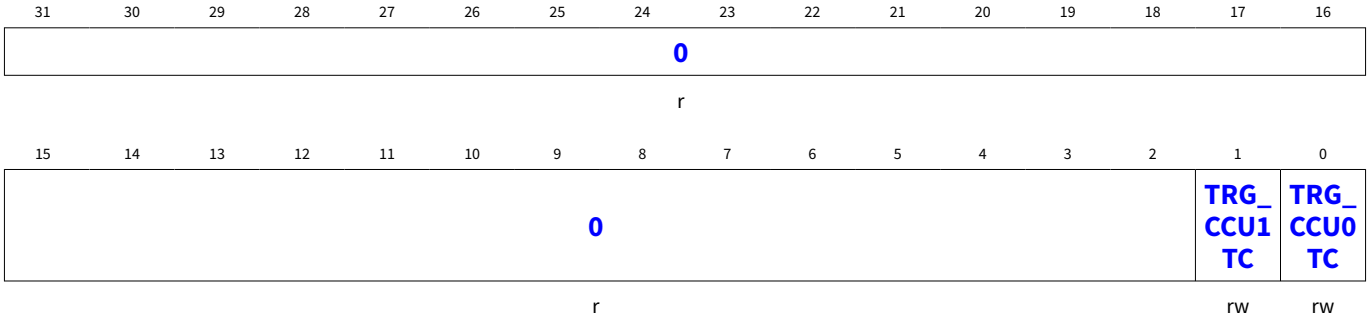
Table 2355 (continued) Reset values of CLSi_ATOM_CHx_IRQ_EN (i=0-2;x=0-7)

Reset	Reset value	Note
ATOM channel reset	0000 0000_H	

44.5.109 ATOM[i] channel [x] software interrupt generation (x=0-7;i=0-2)

CLSi_ATOM_CHx_IRQ_FORCINT (i=0-2;x=0-7) Offset address: 001828_H+i*20000_H+x*80_H

ATOM[i] channel [x] software interrupt generation Reset values see: Table 2356



Field	Bits	Type	Description
TRG_CCU0TC	0	rw	**Trigger the bit** CLSi_ATOM_CHx_IRQ_NOTIFY.CCU0TC **by software.** ***Note**: This bit is cleared automatically after write.* ***Note**: This bit is write-protected by register bit* CLSi_ARCH_CTRL.RF_PROT*.* **If reading:** 0_B No status **If writing:** 0_B No action 1_B Force setting of CLSi_ATOM_CHx_IRQ_NOTIFY.CCU0TC = 1
TRG_CCU1TC	1	rw	**Trigger the bit** CLSi_ATOM_CHx_IRQ_NOTIFY.CCU1TC **by software.** ***Note**: This bit is cleared automatically after write.* ***Note**: This bit is write-protected by register bit* CLSi_ARCH_CTRL.RF_PROT*.* **If reading:** 0_B No status **If writing:** 0_B No action 1_B Force setting of CLSi_ATOM_CHx_IRQ_NOTIFY.CCU1TC = 1
0	31:2	r	**Reserved** Read as 0; must be written with 0.

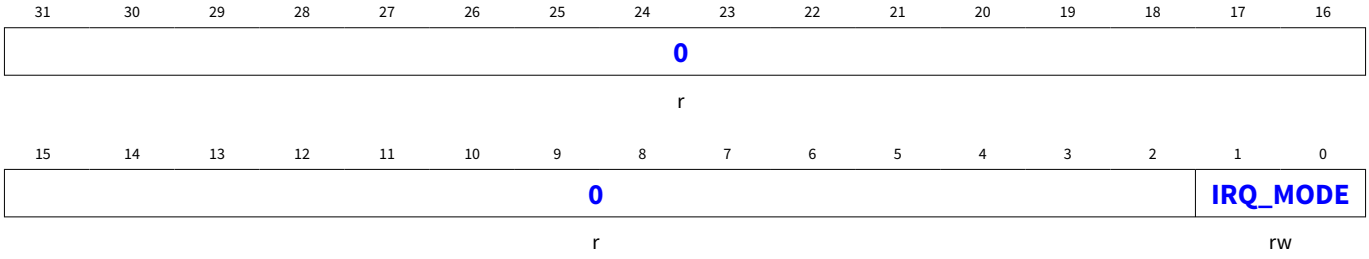
Table 2356 Reset values of CLSi_ATOM_CHx_IRQ_FORCINT (i=0-2;x=0-7)

Reset	Reset value	Note
Kernel Reset	$0000\ 0000_H$	
ATOM channel reset	$0000\ 0000_H$	

44.5.110 ATOM[i] channel [x] interrupt mode configuration register (x=0-7;i=0-2)

CLSi_ATOM_CHx_IRQ_MODE (i=0-2;x=0-7) Offset address: $00182C_H+i*20000_H+x*80_H$

ATOM[i] channel [x] interrupt mode configuration register Reset values see: Table 2357



Field	Bits	Type	Description
IRQ_MODE	1:0	rw	**IRQ mode selection** ***Note**: The interrupt modes are described in **eGTM** interrupt concept.* 00_B Level mode 01_B Pulse mode 10_B Pulse-Notify mode 11_B Single-Pulse mode
0	31:2	r	**Reserved** Read as 0; must be written with 0.

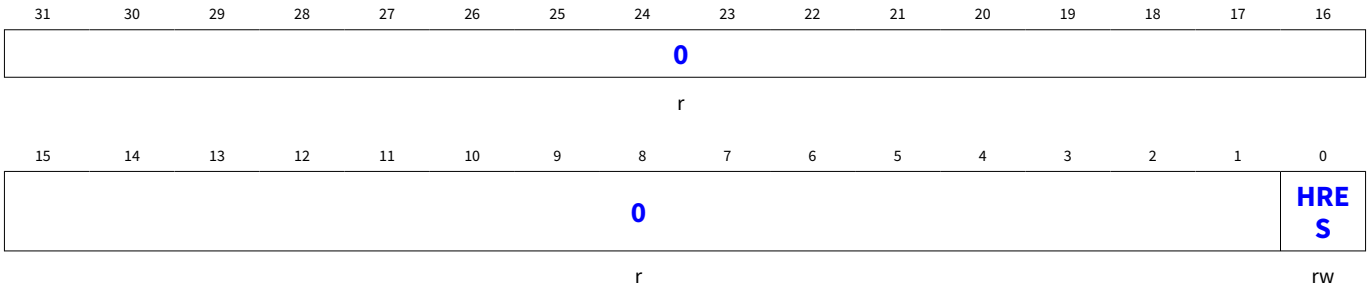
Table 2357 Reset values of CLSi_ATOM_CHx_IRQ_MODE (i=0-2;x=0-7)

Reset	Reset value	Note
Kernel Reset	$0000\ 0002_H$	
ATOM channel reset	$0000\ 0002_H$	

44.5.111 ATOM[i] channel [x] control2 register (x=0-7;i=0-2)

CLSi_ATOM_CHx_CTRL2 (i=0-2;x=0-7) Offset address: $001830_H+i*20000_H+x*80_H$

ATOM[i] channel [x] control2 register Reset values see: Table 2358



Field	Bits	Type	Description
HRES	0	rw	CLSi_ATOM_CHx_CTRL2.HRES: **ATOM high-resolution support** ***Note:*** *CLSi_ATOM_CHx_CTRL2.HRES mode is available in connection with **ATOM** SOMP mode only.* **If x < 8,8,8 [i] && CLSi_ATOM_CHx_CTRL .MODE == 2:** 0_B High-resolution support disabled 1_B High-resolution support enabled **If x >= 8,8,8 [i] || CLSi_ATOM_CHx_CTRL .MODE !=2:** 0_B Initial value, no function in this mode 1_B Prohibited
0	31:1	r	**Reserved** Read as 0; must be written with 0.

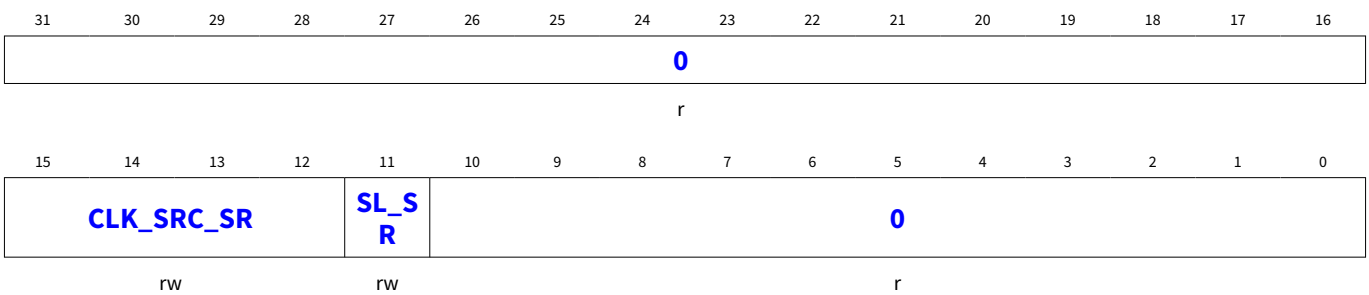
Table 2358 Reset values of CLSi_ATOM_CHx_CTRL2 (i=0-2;x=0-7)

Reset	Reset value	Note
Kernel Reset	0000 0000_H	
ATOM channel reset	0000 0000_H	

44.5.112 ATOM[i] channel [x] control shadow register (x=0-7;i=0-2)

CLSi_ATOM_CHx_CTRL_SR (i=0-2;x=0-7) Offset address: 001834_H+i*20000_H+x*80_H

ATOM[i] channel [x] control shadow register Reset values see: Table 2359



Field	Bits	Type	Description
SL_SR	11	rw	**Shadow register for** CLSi_ATOM_CHx_CTRL**.**SL ***Note**: Reset value depends on the hardware configuration chosen by silicon vendor.* 0_B Low signal level 1_B High signal level
CLK_SRC_SR	15:12	rw	**Shadow register for** CLSi_ATOM_CHx_CTRL**.**CLK_SRC ***Note**: This register is a shadow register for* CLSi_ATOM_CHx_CTRL**.**CLK_SRC*. Thus, if the* CCM[i]_CLK_RES *source for PWM generation should be changed during operation, the old* CCM[i]_CLK_RES *has to operate until the **ATOM** channel's* CLSi_ATOM_CHx_CTRL**.**CLK_SRC *register is updated with the content of* CLSi_ATOM_CHx_CTRL_SR**.**CLK_SRC_SR *either by the end of a period or a forced update.* 0_H CCM[i]_CLK_RES[0:0] resolution in use 1_H CCM[i]_CLK_RES[1:1] resolution in use 2_H CCM[i]_CLK_RES[2:2] resolution in use 3_H CCM[i]_CLK_RES[3:3] resolution in use 4_H CCM[i]_CLK_RES[4:4] resolution in use 5_H CCM[i]_CLK_RES[5:5] resolution in use 6_H CCM[i]_CLK_RES[6:6] resolution in use 7_H CCM[i]_CLK_RES[7:7] resolution in use 8_H Reserved 9_H Reserved A_H Reserved B_H Reserved C_H Reserved D_H ATOM_CH_TRIGOUT[x-1:x-1] selected E_H EXT_TRIGIN selected F_H Reserved
0	10:0, 31:16	r	**Reserved** Read as 0; must be written with 0.

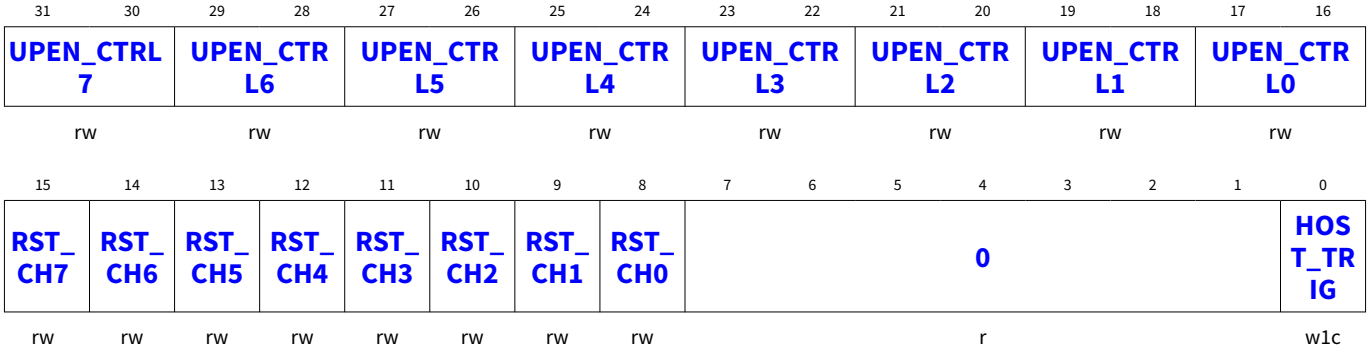
Table 2359 Reset values of CLSi_ATOM_CHx_CTRL_SR (i=0-2;x=0-7)

Reset	Reset value	Note
Kernel Reset	$0000\ 0000_H$	
ATOM channel reset	$0000\ 0000_H$	

44.5.113 ATOM[i] AGC global control register (i=0-2)

CLSi_ATOM_AGC_GLB_CTRL (i=0-2) Offset address: $001C40_H+i*20000_H$

ATOM[i] AGC global control register Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
HOST_TRIG	0	w1c	**Trigger request signal (see AGC) to update the register** CLSi_ATOM_AGC_ENDIS_STAT **and** CLSi_ATOM_AGC_OUTEN_STAT ***Note**: This flag is reset automatically after triggering the update* 0_B no trigger request 1_B set trigger request
RST_CHk (k=0-7)	k+8	rw	**Software reset of channel [k]** ***Note**: This bit is cleared automatically after write-access via the configuration interface. The channel registers are set to their reset values and channel operation is stopped immediately. The output register of SOU unit is reset to inverse reset value of* CLSi_ATOM_CHx_CTRL**.**SL ***(x=k)** bit.* ***Note**: The write-access to* CLSi_ATOM_AGC_GLB_CTRL**.**RST_CHk ***(k=c)** bit will take two internal bus clock-cycles. The internal bus clock is the cluster clock.* **If writing:** 0_B No action 1_B Reset channel **If reading:** 0_B No status 1_B Reserved

(table continues...)

(continued)

Field	Bits	Type	Description
UPEN_CTRLk (k=0-7)	2*k+17:2*k+16	rw	**ATOM channel [k] enable update of register** CLSi_ATOM_CHx_CM0**(x=k),** CLSi_ATOM_CHx_CM1**(x=k),** CLSi_ATOM_CHx_CTRL**.**SL **(x=k) and** CLSi_ATOM_CHx_CTRL**.**CLK_SRC **(x=k) from** CLSi_ATOM_CHx_SR0**(x=k),** CLSi_ATOM_CHx_SR1**(x=k),** CLSi_ATOM_CHx_CTRL_SR**.**SL_SR **(x=k) and** CLSi_ATOM_CHx_CTRL_SR**.**CLK_SRC_SR **(x=k).** **If writing:** 00_B don't care, bits will not be changed 01_B disable update 10_B enable update 11_B don't care, bits will not be changed **If reading:** 00_B update disabled 01_B unused 10_B unused 11_B update enabled
0	7:1	r	**Reserved** Read as 0; must be written with 0.

44.5.114 ATOM[i] AGC enable/disable control register (i=0-2)

CLSi_ATOM_AGC_ENDIS_CTRL (i=0-2) Offset address: $001C44_H+i*20000_H$

ATOM[i] AGC enable/disable control register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ENDIS_CTRL7		ENDIS_CTRL6		ENDIS_CTRL5		ENDIS_CTRL4		ENDIS_CTRL3		ENDIS_CTRL2		ENDIS_CTRL1		ENDIS_CTRL0	
rw		rw		rw		rw		rw		rw		rw		rw	

Field	Bits	Type	Description
ENDIS_CTRLk (k=0-7)	2*k+1:2*k	rw	**ATOM [i] channel [k] enable/disable control register.** **If writing:** 00_B bit field CLSi_ATOM_AGC_ENDIS_STAT.ENDIS_STATk of register ATOM[i] will not be changed on an update trigger 01_B disable channel on an update trigger 10_B enable channel on an update trigger 11_B don't change bits of this register **If reading:** 00_B bit field CLSi_ATOM_AGC_ENDIS_STAT.ENDIS_STATk will not be changed on an update trigger 01_B disable channel on an update trigger 10_B enable channel on an update trigger
0	31:16	r	**Reserved** Read as 0; must be written with 0.

44.5.115 ATOM[i] AGC enable/disable status register (i=0-2)

CLSi_ATOM_AGC_ENDIS_STAT (i=0-2) Offset address: $001C48_H+i*20000_H$

ATOM[i] AGC enable/disable status register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ENDIS_STAT7		ENDIS_STAT6		ENDIS_STAT5		ENDIS_STAT4		ENDIS_STAT3		ENDIS_STAT2		ENDIS_STAT1		ENDIS_STAT0	
rwh		rwh		rwh		rwh		rwh		rwh		rwh		rwh	

Field	Bits	Type	Description
ENDIS_STATk (k=0-7)	2*k+1:2*k	rwh	**ATOM [i] channel [k] enable/disable status register** ***Note**: If a channel of **ATOM** instance i is disabled (ATOM_ENDIS[x:x]=0), the counter CLSi_ATOM_CHx_CN0.CN0 is stopped (SOMP, SOMS mode) and the comparisons in CCU0 and CCU1 are stopped (SOMP, SOMC, SOMB mode). When the channel is re-enabled afterwards, the counter CLSi_ATOM_CHx_CN0.CN0 starts counting from its current value or/and the comparison is restarted. In SOMI mode, the operation and output generation is independent of CLSi_ATOM_AGC_ENDIS_STAT/ CLSi_ATOM_AGC_ENDIS_CTRL and ATOM_ENDIS[x:x].* **If writing:** 00_B don't care, bits CLSi_ATOM_AGC_ENDIS_STAT.ENDIS_STATk will not be changed 01_B disable channel 10_B enable channel 11_B don't care, bits CLSi_ATOM_AGC_ENDIS_STAT.ENDIS_STATk will not be changed **If reading:** 00_B channel disabled 01_B unused 10_B unused 11_B channel enabled
0	31:16	r	**Reserved** Read as 0; must be written with 0.

44.5.116 ATOM[i] AGC action time base register (i=0-2)

CLSi_ATOM_AGC_ACT_TB (i=0-2) Offset address: $001C4C_H+i*20000_H$

ATOM[i] AGC action time base register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0					TBU_SEL		TB_TRIG	ACT_TB							
r					rw		rwh	rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ACT_TB															
rw															

Field	Bits	Type	Description
ACT_TB	23:0	rw	**Action time base. When the selected TBU time base CCM[i]_TBU_TS0/CCM[i]_TBU_TS1/CCM[i]_TBU_TS2 is "cyclically greater than or equal to" the action time base CLSi_ATOM_AGC_ACT_TB.ACT_TB, the compare event specified in CLSi_ATOM_AGC_ACT_TB.ACT_TB is considered to have occurred in the past and so the trigger ATOM_CTRL_TRIG is immediately generated in AGC unit if CLSi_ATOM_AGC_ACT_TB.TB_TRIG is 1. Please refer to chapter eGTM Architecture for more information about cyclic event compare strategy.**
TB_TRIG	24	rwh	**Set trigger request** ***Note**: This flag is reset automatically if the selected time base unit (CCM[i]_TBU_TS0 or CCM[i]_TBU_TS1 or CCM[i]_TBU_TS2 if present) has reached the value CLSi_ATOM_AGC_ACT_TB.ACT_TB and the update of the register were triggered.* 0_B no trigger request 1_B set trigger request
TBU_SEL	26:25	rw	**Selection of time base used for comparison** 00_B CCM[i]_TBU_TS0 selected 01_B CCM[i]_TBU_TS1 selected 10_B CCM[i]_TBU_TS2 selected 11_B same as 00_B
0	31:27	r	**Reserved** Read as 0; must be written with 0.

44.5.117 ATOM[i] AGC output enable control register (i=0-2)

CLSi_ATOM_AGC_OUTEN_CTRL (i=0-2) Offset address: $001C50_H+i*20000_H$

ATOM[i] AGC output enable control register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OUTEN_CTRL7		OUTEN_CTRL6		OUTEN_CTRL5		OUTEN_CTRL4		OUTEN_CTRL3		OUTEN_CTRL2		OUTEN_CTRL1		OUTEN_CTRL0	
rw		rw		rw		rw		rw		rw		rw		rw	

Field	Bits	Type	Description
OUTEN_CTRLk (k=0-7)	2*k+1:2*k	rw	**Output enable control of ATOM [i] channel [k] output** ATOM_OUT***[x:x]*** **If writing:** 00_B bit field CLSi_ATOM_AGC_OUTEN_STAT.OUTEN_STATk will not be changed on an update trigger 01_B disable channel output on an update trigger 10_B enable channel output on an update trigger 11_B don't change bits of this register **If reading:** 00_B bit field CLSi_ATOM_AGC_OUTEN_STAT.OUTEN_STATk will not be changed on an update trigger 01_B disable channel output on an update trigger 10_B enable channel output on an update trigger
0	31:16	r	**Reserved** Read as 0; must be written with 0.

44.5.118 ATOM[i] AGC output enable status register (i=0-2)

CLSi_ATOM_AGC_OUTEN_STAT (i=0-2) Offset address: $001C54_H+i*20000_H$

ATOM[i] AGC output enable status register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OUTEN_STAT7		OUTEN_STAT6		OUTEN_STAT5		OUTEN_STAT4		OUTEN_STAT3		OUTEN_STAT2		OUTEN_STAT1		OUTEN_STAT0	
rwh		rwh		rwh		rwh		rwh		rwh		rwh		rwh	

Field	Bits	Type	Description
OUTEN_STATk (k=0-7)	2*k+1:2*k	rwh	**Output enable status of ATOM [i] channel [x] output** ATOM_OUT***[k:k]*** ***Note**: If the output of **ATOM** i channel x is disabled (ATOM_OUTEN[x:x]=0), both **ATOM** channel outputs ATOM_OUT[x:x] and ATOM_OUT_T[x:x] of instance i, channel x are the inverse value of bit CLSi_ATOM_CHx_CTRL.SL.* **If writing:** 00_B don't care, bits CLSi_ATOM_AGC_OUTEN_STAT.OUTEN[k] will not be changed 01_B disable output 10_B enable output 11_B don't care, bits CLSi_ATOM_AGC_OUTEN_STAT.OUTEN[k] will not be changed **If reading:** 00_B output disabled 01_B unused 10_B unused 11_B output enabled
0	31:16	r	**Reserved** Read as 0; must be written with 0.

44.5.119 ATOM[i] AGC force update control register (i=0-2)

CLSi_ATOM_AGC_FUPD_CTRL (i=0-2) Offset address: $001C58_H+i*20000_H$

ATOM[i] AGC force update control register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSTCN0_CH7		RSTCN0_CH6		RSTCN0_CH5		RSTCN0_CH4		RSTCN0_CH3		RSTCN0_CH2		RSTCN0_CH1		RSTCN0_CH0	
rw		rw		rw		rw		rw		rw		rw		rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FUPD_CTRL7		FUPD_CTRL6		FUPD_CTRL5		FUPD_CTRL4		FUPD_CTRL3		FUPD_CTRL2		FUPD_CTRL1		FUPD_CTRL0	
rw		rw		rw		rw		rw		rw		rw		rw	

Field	Bits	Type	Description
FUPD_CTRLk (k=0-7)	2*k+1:2*k	rw	**Force update of ATOM channel [k] operation registers** If enabled, force update of register CLSi_ATOM_CHx_CM0.CM0, CLSi_ATOM_CHx_CM1.CM1, CLSi_ATOM_CHx_CTRL.SL and CLSi_ATOM_CHx_CTRL.CLK_SRC triggered by CLSi_ATOM_AGC_GLB_CTRL.HOST_TRIG, CLSi_ATOM_AGC_ACT_TB.ACT_TB compare match or internal trigger. **If writing:** 00_B don't care, bits CLSi_ATOM_AGC_FUPD_CTRL.FUPD_CTRLk will not be changed 01_B disable force update 10_B enable force update 11_B don't care, bits CLSi_ATOM_AGC_FUPD_CTRL.FUPD_CTRLk will not be changed **If reading:** 00_B force update disabled 01_B unused 10_B unused 11_B force update enabled
RSTCN0_CHk (k=0-7)	2*k+17:2*k +16	rw	**Reset CLSi_ATOM_CHx_CN0(x=k) of channel [k] on force update event** If enabled, CLSi_ATOM_CHx_CN0.CN0 will be reset with a force update event. **If writing:** 00_B don't care, bits will not be changed 01_B do not reset CLSi_ATOM_CHx_CN0.CN0 on forced update 10_B reset CLSi_ATOM_CHx_CN0.CN0 on forced update 11_B don't care, bits will not be changed **If reading:** 00_B CLSi_ATOM_CHx_CN0.CN0 is not reset on forced update 01_B unused 10_B unused 11_B CLSi_ATOM_CHx_CN0.CN0 is reset on forced update

44.5.120 ATOM[i] AGC internal trigger control register (i=0-2)

CLSi_ATOM_AGC_INT_TRIG (i=0-2) Offset address: $001C5C_H+i*20000_H$

ATOM[i] AGC internal trigger control register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
INT_TRIG7		INT_TRIG6		INT_TRIG5		INT_TRIG4		INT_TRIG3		INT_TRIG2		INT_TRIG1		INT_TRIG0	
rw		rw		rw		rw		rw		rw		rw		rw	

Field	Bits	Type	Description
INT_TRIGk (k=0-7)	2*k+1:2*k	rw	**Select input signal** ATOM_CH_TRIGOUT***[k:k]*** **as a trigger source** **If writing:** 00_B don't care, bits will not be changed 01_B do not use internal trigger from channel k (ATOM_CH_TRIGOUT[k:k]) 10_B use internal trigger from channel k (ATOM_CH_TRIGOUT[k:k]) 11_B don't care, bits will not be changed **If reading:** 00_B internal trigger from channel k (ATOM_CH_TRIGOUT[k:k]) not used 01_B unused 10_B unused 11_B internal trigger from channel k (ATOM_CH_TRIGOUT[k:k]) used
0	31:16	r	**Reserved** Read as 0; must be written with 0.

44.5.121 CCM[i] 2. hardware configuration register (i=0-2)

CLSi_CCM_HW_CONF2 (i=0-2) Offset address: $0041D4_H+i*20000_H$

CCM[i] 2. hardware configuration register Kernel Reset value: $0005\ 02A6_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0													1	0	1
r													r1	r	r1

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0						1	0	1	0	1	RES_0				
r						r1	r	r1	r	r1	r*				

Field	Bits	Type	Description
RES_0	4:0	r*	**Reserved** Read as 6
1	5, 7, 9, 16, 18	r1	**Reserved** Read as 1
0	6, 8, 15:10, 17, 31:19	r	**Reserved** Read as 0.

44.5.122 CCM[i] hardware configuration register (i=0-2)

***Note**: Reset value depends on the hardware configuration chosen by silicon vendor.*

CLSi_CCM_HW_CONF (i=0-2) Offset address: $0041DC_H+i*20000_H$

CCM[i] hardware configuration register Kernel Reset value: $488F\ 022A_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AEI_RDATA_PIPELINE_STAGE	AEI_ADDR_PIPELINE_STAGE	INT_CLK_EN_GEN	TOM_TRIG_INTCHAIN					ATOM_TRIG_INTCHAIN				IRQ_MODE_SINGLE_PULSE	IRQ_MODE_PULSE_NOTIFY	IRQ_MODE_PULSE	IRQ_MODE_LEVEL
r	r	r	r					r				r	r	r	r

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESET_ACTIVE	0	0	RAM_INIT_RST	TOM_TRIG_CHAIN			TOM_OUT_RST	ATOM_TRIG_CHAIN			ATOM_OUT_RST	1	SYNC_INPUT_REG	BRIDGE_MODE_RST	0
r	r	r	r	r			r	r			r	r1	r	r	r

Field	Bits	Type	Description
BRIDGE_MODE_RST	1	r	**Bridge mode after reset** 0_B Bridge starts in synchronous mode after reset 1_B Bridge starts in asynchronous mode after reset
SYNC_INPUT_REG	2	r	**Additional pipelined stage in synchronous bridge mode** ***Note**: This register is only relevant (if existing) for synchronous bridge mode* 0_B No additional pipelined stage implemented 1_B additional pipelined stage implemented. All accesses in synchronous mode will be increased by one clock-cycle.
ATOM_OUT_RST	4	r	CCM_ATOM_OUT **reset level** ***Note**: This value represents the **ATOM** output level after reset. The inverse value of this bit is the reset value of bit CLSi_ATOM_CHx_CTRL.SL in all **ATOM** channels.* 0_B CCM_ATOM_OUT reset level is '0' 1_B CCM_ATOM_OUT reset level is '1'
ATOM_TRIG_CHAIN	7:5	r	**ATOM trigger chain length without synchronization register** It defines after which **ATOM** instance count a synchronization register is introduced into trigger chain (after ATOM_TRIGOUT output of instance [i] and ATOM_TRIGIN input of instance [i]+1). Valid values are 1 to 7. One (1) means that after each instance a synchronization register is placed.

(table continues...)

(continued)

Field	Bits	Type	Description
TOM_OUT_RST	8	r	CCM_TOM_OUT **reset level** ***Note**: This value represents the **TOM** output level after reset. The inverse value of this bit is the reset value of bit CLSi_TOM_CHx_CTRL.SL in all **TOM** channels.* 0_B CCM_TOM_OUT reset level is '0' 1_B CCM_TOM_OUT reset level is '1'
TOM_TRIG_CHAIN	11:9	r	**TOM trigger chain length without synchronization register** It defines, after which **TOM** instance count a synchronization register is introduced into trigger chain (after TOM_TRIGOUT output of instance [i] and TOM_TRIGIN input of instance [i]+1). Valid values are 1 to 7. 1 means that after each instance a synchronization register is placed.
RAM_INIT_RST	12	r	**RAM initialization from reset** 0_B RAM is not initialized after reset 1_B RAM is initialized after reset
RESET_ACTIVE	15	r	**Active level of asynchronous reset** 0_B asynchronous reset active low 1_B asynchronous reset active high
IRQ_MODE_LEVEL	16	r	**Signalize availability of Level IRQ mode** 0_B level mode not available 1_B level mode available
IRQ_MODE_PULSE	17	r	**Signalize availability of Pulse IRQ mode** 0_B pulse mode not available 1_B pulse mode available
IRQ_MODE_PULSE_NOTIFY	18	r	**Signalize availability of Pulse Notify IRQ mode** 0_B pulse notify mode not available 1_B pulse notify mode available
IRQ_MODE_SINGLE_PULSE	19	r	**Signalize availability of Single Pulse IRQ mode** 0_B single pulse mode not available 1_B single pulse mode available
ATOM_TRIG_INTCHAIN	23:20	r	**ATOM internal trigger chain length for pipeline register** 4_H Pipeline register present in trigger chain signals of channel 4. This will not have any functional impact if cluster i of device has FAST_CLK_CLS[i]=0 or if the cluster is enabled with clock divider 2 (CLSi_ARCH_CLK_CFG.CLSj_CLK_DIV (j=i) = 2). 8_H No pipeline register in trigger chain signals channel 4 exists **others,** Reserved

(table continues...)

(continued)

Field	Bits	Type	Description
TOM_TRIG_INTCHAIN	28:24	r	**TOM internal trigger chain length for pipeline register** 08_H Pipeline register in trigger chain signals of channel 8 present. This will not have any functional impact if cluster i of device has FAST_CLK_CLS[i]=0 defined. 10_H No pipeline register in trigger chain signals channel 8 exists **others,** Reserved
INT_CLK_EN_GEN	29	r	**Internal clock enable generation** 0_B eGTM external clock enable signals in use 1_B eGTM internal clock enable signals in use
AEI_ADDR_PIPELINE_STAGE	30	r	**Address pipeline stage implemented** 0_B no address pipeline stage implemented 1_B address pipeline stage implemented
AEI_RDATA_PIPELINE_STAGE	31	r	**Read data pipeline stage implemented** 0_B no read data pipeline stage implemented 1_B read data pipeline stage implemented
0	0, 13, 14	r	**Reserved** Read as 0.
1	3	r1	**Reserved** Read as 1

44.5.123 CCM[i] TIM AUX input source register (i=0-2)

CLSi_CCM_TIM_AUX_IN_SRC (i=0-2) Offset address: $0041E0_H+i*20000_H$

CCM[i] TIM AUX input source register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								SEL_OUT_N_CH7	SEL_OUT_N_CH6	SEL_OUT_N_CH5	SEL_OUT_N_CH4	SEL_OUT_N_CH3	SEL_OUT_N_CH2	SEL_OUT_N_CH1	SEL_OUT_N_CH0
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								SRC_CH7	SRC_CH6	SRC_CH5	SRC_CH4	SRC_CH3	SRC_CH2	SRC_CH1	SRC_CH0
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
SRC_CH0	0	rw	**Defines** CCM_AUX_IN **source of TIM[i] channel 0** **If** CLSi_CCM_TIM_AUX_IN_SRC **.SEL_OUT_N_CH0 == 0 &&** CLSi_ARCH_CFG **.SRC_IN_MUX == 0:** 0_B DTM[h/4] of instance [k] Output DTM_OUT0[0:0] selected where h = mod(i, 2)*8 and k = i/2 1_B DTM4 of instance [i] Output DTM_OUT0[0:0] selected **If** CLSi_CCM_TIM_AUX_IN_SRC **.SEL_OUT_N_CH0 == 0 &&** CLSi_ARCH_CFG **.SRC_IN_MUX == 1:** 0_B DTM0 of instance [i] Output DTM_OUT0[0:0] selected 1_B DTM4 of instance [i] Output DTM_OUT0[0:0] selected **If** CLSi_CCM_TIM_AUX_IN_SRC **.SEL_OUT_N_CH0 == 1:** 0_B DTM0 of instance [i] Output DTM_OUT1[1:1] selected 1_B DTM4 of instance [i] Output DTM_OUT1[1:1] selected
SRC_CH1	1	rw	**Defines** CCM_AUX_IN **source of TIM[i] channel 1** **If** CLSi_CCM_TIM_AUX_IN_SRC **.SEL_OUT_N_CH1 == 0 &&** CLSi_ARCH_CFG **.SRC_IN_MUX == 0:** 0_B DTM[h/4] of instance [k] Output DTM_OUT0[1:1] selected where h = mod(i, 2)*8 and k = i/2 1_B DTM4 of instance [i] Output DTM_OUT0[1:1] selected **If** CLSi_CCM_TIM_AUX_IN_SRC **.SEL_OUT_N_CH1 == 0 &&** CLSi_ARCH_CFG **.SRC_IN_MUX == 1:** 0_B DTM0 of instance [i] Output DTM_OUT0[1:1] selected 1_B DTM4 of instance [i] Output DTM_OUT0[1:1] selected **If** CLSi_CCM_TIM_AUX_IN_SRC **.SEL_OUT_N_CH1 == 1:** 0_B DTM0 of instance [i] Output DTM_OUT1[2:2] selected 1_B DTM4 of instance [i] Output DTM_OUT1[2:2] selected
SRC_CH2	2	rw	**Defines** CCM_AUX_IN **source of TIM[i] channel 2** **If** CLSi_CCM_TIM_AUX_IN_SRC **.SEL_OUT_N_CH2 == 0 &&** CLSi_ARCH_CFG **.SRC_IN_MUX == 0:** 0_B DTM[h/4] of instance [k] Output DTM_OUT0[2:2] selected where h = mod(i, 2)*8 and k = i/2 1_B DTM4 of instance [i] Output DTM_OUT0[2:2] selected **If** CLSi_CCM_TIM_AUX_IN_SRC **.SEL_OUT_N_CH2 == 0 &&** CLSi_ARCH_CFG **.SRC_IN_MUX == 1:** 0_B DTM0 of instance [i] Output DTM_OUT0[2:2] selected 1_B DTM4 of instance [i] Output DTM_OUT0[2:2] selected **If** CLSi_CCM_TIM_AUX_IN_SRC **.SEL_OUT_N_CH2 == 1:** 0_B DTM0 of instance [i] Output DTM_OUT1[3:3] selected 1_B DTM4 of instance [i] Output DTM_OUT1[3:3] selected

(table continues...)

(continued)

Field	Bits	Type	Description
SRC_CH3	3	rw	**Defines** CCM_AUX_IN **source of TIM[i] channel 3** **If** CLSi_CCM_TIM_AUX_IN_SRC **.SEL_OUT_N_CH3 == 0 &&** CLSi_ARCH_CFG **.SRC_IN_MUX == 0:** 0_B DTM[h/4] of instance [k] Output DTM_OUT0[3:3] selected where h = mod(i, 2)*8 and k = i/2 1_B DTM4 of instance [i] Output DTM_OUT0[3:3] selected **If** CLSi_CCM_TIM_AUX_IN_SRC **.SEL_OUT_N_CH3 == 0 &&** CLSi_ARCH_CFG **.SRC_IN_MUX == 1:** 0_B 0 = DTM0 of instance [i] Output DTM_OUT0[3:3] selected 1_B DTM4 of instance [i] Output DTM_OUT0[3:3] selected **If** CLSi_CCM_TIM_AUX_IN_SRC **.SEL_OUT_N_CH3 == 1:** 0_B DTM1 of instance [i] Output DTM_OUT1[0:0] selected 1_B DTM5 of instance [i] Output DTM_OUT1[0:0] selected
SRC_CH4	4	rw	**Defines** CCM_AUX_IN **source of TIM[i] channel 4** **If** CLSi_CCM_TIM_AUX_IN_SRC **.SEL_OUT_N_CH4 == 0 &&** CLSi_ARCH_CFG **.SRC_IN_MUX == 0:** 0_B DTM[h/4+1] of instance [k] Output DTM_OUT0[0:0] selected where h = mod(i, 2)*8 and k = i/2 1_B DTM5 of instance [i] Output DTM_OUT0[0:0] selected **If** CLSi_CCM_TIM_AUX_IN_SRC **.SEL_OUT_N_CH4 == 0 &&** CLSi_ARCH_CFG **.SRC_IN_MUX == 1:** 0_B DTM1 of instance [i] Output DTM_OUT0[0:0] selected 1_B DTM5 of instance [i] Output DTM_OUT0[0:0] selected **If** CLSi_CCM_TIM_AUX_IN_SRC **.SEL_OUT_N_CH4 == 1:** 0_B DTM1 of instance [i] Output DTM_OUT1[1:1] selected 1_B DTM5 of instance [i] Output DTM_OUT1[1:1] selected
SRC_CH5	5	rw	**Defines** CCM_AUX_IN **source of TIM[i] channel 5** **If** CLSi_CCM_TIM_AUX_IN_SRC **.SEL_OUT_N_CH5 == 0 &&** CLSi_ARCH_CFG **.SRC_IN_MUX == 0:** 0_B DTM[h/4+1] of instance [k] Output DTM_OUT0[1:1] selected where h = mod(i, 2)*8 and k = i/2 1_B DTM5 of instance [i] Output DTM_OUT0[1:1] selected **If** CLSi_CCM_TIM_AUX_IN_SRC **.SEL_OUT_N_CH5 == 0 &&** CLSi_ARCH_CFG **.SRC_IN_MUX == 1:** 0_B DTM1 of instance [i] Output DTM_OUT0[1:1] selected 1_B DTM5 of instance [i] Output DTM_OUT0[1:1] selected **If** CLSi_CCM_TIM_AUX_IN_SRC **.SEL_OUT_N_CH5 == 1:** 0_B DTM1 of instance [i] Output DTM_OUT1[2:2] selected 1_B DTM5 of instance [i] Output DTM_OUT1[2:2] selected

(table continues...)

(continued)

Field	Bits	Type	Description
SRC_CH6	6	rw	**Defines** CCM_AUX_IN **source of TIM[i] channel 6** **If** CLSi_CCM_TIM_AUX_IN_SRC **.SEL_OUT_N_CH6 == 0 &&** CLSi_ARCH_CFG **.SRC_IN_MUX == 0:** 0_B DTM[h/4+1] of instance [k] Output DTM_OUT0[2:2] selected where h = mod(i, 2)*8 and k = i/2 1_B DTM5 of instance [i] Output DTM_OUT0[2:2] selected **If** CLSi_CCM_TIM_AUX_IN_SRC **.SEL_OUT_N_CH6 == 0 &&** CLSi_ARCH_CFG **.SRC_IN_MUX == 1:** 0_B DTM1 of instance [i] Output DTM_OUT0[2:2] selected 1_B DTM5 of instance [i] Output DTM_OUT0[2:2] selected **If** CLSi_CCM_TIM_AUX_IN_SRC **.SEL_OUT_N_CH6 == 1:** 0_B DTM1 of instance [i] Output DTM_OUT1[3:3] selected 1_B DTM5 of instance [i] Output DTM_OUT1[3:3] selected
SRC_CH7	7	rw	**Defines** CCM_AUX_IN **source of TIM[i] channel 7** **If** CLSi_CCM_TIM_AUX_IN_SRC **.SEL_OUT_N_CH7 == 0 &&** CLSi_ARCH_CFG **.SRC_IN_MUX == 0:** 0_B DTM[h/4+1] of instance [k] Output DTM_OUT0[3:3] selected where h = mod(i, 2)*8 and k = i/2 1_B DTM5 of instance [i] Output DTM_OUT0[3:3] selected **If** CLSi_CCM_TIM_AUX_IN_SRC **.SEL_OUT_N_CH7 == 0 &&** CLSi_ARCH_CFG **.SRC_IN_MUX == 1:** 0_B DTM1 of instance [i] Output DTM_OUT0[3:3] selected 1_B DTM5 of instance [i] Output DTM_OUT0[3:3] selected **If** CLSi_CCM_TIM_AUX_IN_SRC **.SEL_OUT_N_CH7 == 1:** 0_B DTM0 of instance [i] Output DTM_OUT1[0:0] selected 1_B DTM4 of instance [i] Output DTM_OUT1[0:0] selected
SEL_OUT_N_CHx (x=0-7)	x+16	rw	**Use timer output signals (coming from** DTM_OUT0 **ports) or the inverted output signals (coming from** DTM_OUT1 **ports) as** CCM_AUX_IN **source of TIM[i] channel [x]** 0_B Use DTM_OUT0[q:q] signal as CCM_AUX_IN source of TIM[i] channel x; q = mod(x, 4) 1_B Use DTM_OUT1[q:q] signal as CCM_AUX_IN source of TIM[i] channel x; q = mod(x+1, 4)
0	15:8, 31:24	r	**Reserved** Read as 0; must be written with 0.

44.5.124 CCM[i] TOM output register (i=0-2)

CLSi_CCM_TOM_OUT (i=0-2)

CCM[i] TOM output register

Offset address: $0041E8_H$+i*20000_H

Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TOM_OUT_N15	TOM_OUT_N14	TOM_OUT_N13	TOM_OUT_N12	TOM_OUT_N11	TOM_OUT_N10	TOM_OUT_N9	TOM_OUT_N8	TOM_OUT_N7	TOM_OUT_N6	TOM_OUT_N5	TOM_OUT_N4	TOM_OUT_N3	TOM_OUT_N2	TOM_OUT_N1	TOM_OUT_N0
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TOM_OUT15	TOM_OUT14	TOM_OUT13	TOM_OUT12	TOM_OUT11	TOM_OUT10	TOM_OUT9	TOM_OUT8	TOM_OUT7	TOM_OUT6	TOM_OUT5	TOM_OUT4	TOM_OUT3	TOM_OUT2	TOM_OUT1	TOM_OUT0
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
TOM_OUTx (x=0-15)	x	rh	**Output level snapshot of** CCM_TOM_OUT **channel [x]** Actual level of primary output ports CCM_TOM_OUT[x:x] of channel x (after **DTM**) ***Note**: Reset value depends on the hardware configuration chosen by silicon vendor. See CLSi_CCM_HW_CONF for chosen value.*
TOM_OUT_Nx (x=0-15)	x+16	rh	**Output level snapshot of** CCM_TOM_OUT_N **channel [x]** Actual level of primary output ports CCM_TOM_OUT_N[x:x] of channel x (after **DTM**) ***Note**: Reset value depends on the hardware configuration chosen by silicon vendor. See CLSi_CCM_HW_CONF for chosen value.*

44.5.125 CCM[i] ATOM output register (i=0-1)

CLSi_CCM_ATOM_OUT (i=0-1) Offset address: $0041EC_H+i*20000_H$

CCM[i] ATOM output register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ATOM_IP1_OUT_N7	ATOM_IP1_OUT_N6	ATOM_IP1_OUT_N5	ATOM_IP1_OUT_N4	ATOM_IP1_OUT_N3	ATOM_IP1_OUT_N2	ATOM_IP1_OUT_N1	ATOM_IP1_OUT_N0	ATOM_IP1_OUT7	ATOM_IP1_OUT6	ATOM_IP1_OUT5	ATOM_IP1_OUT4	ATOM_IP1_OUT3	ATOM_IP1_OUT2	ATOM_IP1_OUT1	ATOM_IP1_OUT0
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ATOM_I_OUT_N7	ATOM_I_OUT_N6	ATOM_I_OUT_N5	ATOM_I_OUT_N4	ATOM_I_OUT_N3	ATOM_I_OUT_N2	ATOM_I_OUT_N1	ATOM_I_OUT_N0	ATOM_I_OUT7	ATOM_I_OUT6	ATOM_I_OUT5	ATOM_I_OUT4	ATOM_I_OUT3	ATOM_I_OUT2	ATOM_I_OUT1	ATOM_I_OUT0
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
ATOM_I_OUTx (x=0-7)	x	rh	**Output level snapshot of** CCM_ATOM_OUT **channel [x]** Actual level of primary output ports CCM_ATOM_OUT[x:x] of channel x (after **DTM**) ***Note**: Reset value depends on the hardware configuration chosen by silicon vendor. See CLSi_CCM_HW_CONF for chosen value.*
ATOM_I_OUT_Nx (x=0-7)	x+8	rh	**Output level snapshot of** CCM_ATOM_OUT_N **channel [x]** Actual level of primary output ports CCM_ATOM_OUT_N[x:x] of channel x (after **DTM**) ***Note**: Reset value depends on the hardware configuration chosen by silicon vendor. See CLSi_CCM_HW_CONF for chosen value.*
ATOM_IP1_OUTx (x=0-7)	x+16	rh	**Output level snapshot of ATOM[i+1]_OUT channel [x]** Actual level of primary output ports CCM_ATOM_OUT[x:x] of instance i+1 channel x (after **DTM**) ***Note**: Reset value depends on the hardware configuration chosen by silicon vendor. See CLSi_CCM_HW_CONF for chosen value.*
ATOM_IP1_OUT_Nx (x=0-7)	x+24	rh	**Output level snapshot of** CCM_ATOM_OUT***[x:x]*** **of instance i+1 channel x** Actual level of primary output ports CCM_ATOM_OUT_N[x:x] of instance i+1 channel x (after **DTM**) ***Note**: Reset value depends on the hardware configuration chosen by silicon vendor. See CLSi_CCM_HW_CONF for chosen value.*

44.5.126 CCM[i] ATOM output register (i=2)

CLSi_CCM_ATOM_OUT (i=2) Offset address: $0041EC_H+i*20000_H$

CCM[i] ATOM output register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ATOM_I_OUT_N7	ATOM_I_OUT_N6	ATOM_I_OUT_N5	ATOM_I_OUT_N4	ATOM_I_OUT_N3	ATOM_I_OUT_N2	ATOM_I_OUT_N1	ATOM_I_OUT_N0	ATOM_I_OUT7	ATOM_I_OUT6	ATOM_I_OUT5	ATOM_I_OUT4	ATOM_I_OUT3	ATOM_I_OUT2	ATOM_I_OUT1	ATOM_I_OUT0
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
ATOM_I_OUTx (x=0-7)	x	rh	**Output level snapshot of** CCM_ATOM_OUT **channel [x]** Actual level of primary output ports CCM_ATOM_OUT[x:x] of channel x (after **DTM**) ***Note**: Reset value depends on the hardware configuration chosen by silicon vendor. See CLSi_CCM_HW_CONF for chosen value.*
ATOM_I_OUT_Nx (x=0-7)	x+8	rh	**Output level snapshot of** CCM_ATOM_OUT_N **channel [x]** Actual level of primary output ports CCM_ATOM_OUT_N[x:x] of channel x (after **DTM**) ***Note**: Reset value depends on the hardware configuration chosen by silicon vendor. See CLSi_CCM_HW_CONF for chosen value.*
0	23, 22, 21, 20, 19, 18, 17, 16, 31, 30, 29, 28, 27, 26, 25, 24	r	**Reserved** Read as 0.

44.5.127 CCM[i] CMU clock configuration register (i=0-2)

***Note**: The bit fields of this register are only writable if bit field CLSi_CCM_PROT.CLS_PROT is cleared.*

CLSi_CCM_CMU_CLK_CFG (i=0-2) Offset address: $0041F0_H+i*20000_H$

CCM[i] CMU clock configuration register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		CLK7_SRC		0		CLK6_SRC		0		CLK5_SRC		0		CLK4_SRC	
r		rwh		r		rwh		r		rwh		r		rwh	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		CLK3_SRC		0		CLK2_SRC		0		CLK1_SRC		0		CLK0_SRC	
r		rwh		r		rwh		r		rwh		r		rwh	

Field	Bits	Type	Description
CLKy_SRC (y=0-7)	4*y+1:4*y	rwh	**Clock [y] source signal selector** 00_B Use CMU_CLK_RES[y:y] signal of CMU as CCM[i]_CLK_RES[y:y] signal within Cluster 01_B Use CMU_CLK_RES[8:8] signal of CMU as CCM[i]_CLK_RES[y:y] signal within Cluster 10_B Use TIM_EXT_CAPTURE[y:y] of TIM[i] signal as CCM[i]_CLK_RES[y:y] signal within cluster **others,** Reserved

(table continues...)

(continued)

Field	Bits	Type	Description
0	3:2, 7:6, 11:10, 15:14, 19:18, 23:22, 27:26, 31:30	r	**Reserved** Read as 0; must be written with 0.

44.5.128 CCM[i] CMU fixed clock configuration register (i=0-2)

CLSi_CCM_CMU_FXCLK_CFG (i=0-2) Offset address: $0041F4_H+i*20000_H$

CCM[i] CMU fixed clock configuration register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												FXCLK0_SRC			
r												rwh			

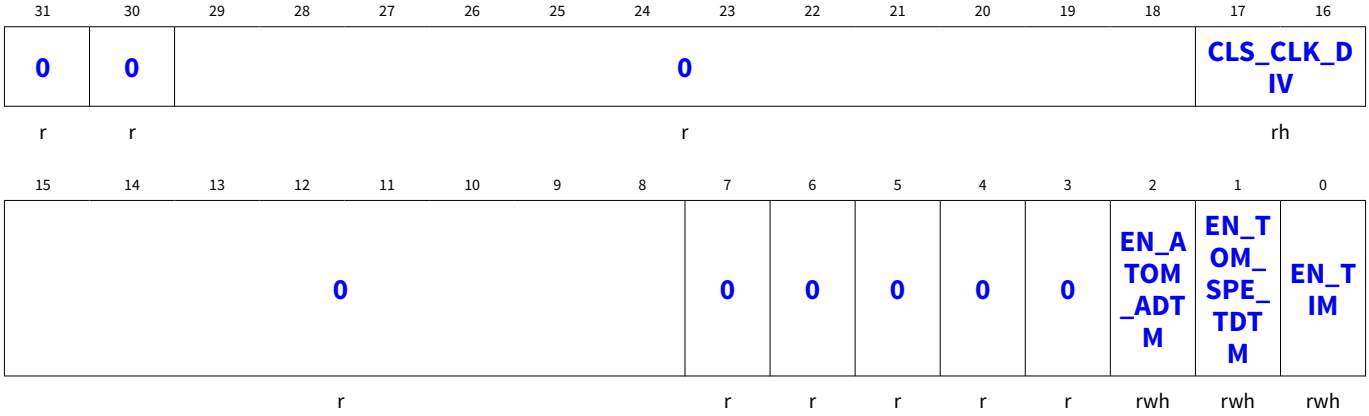
Field	Bits	Type	Description
FXCLK0_SRC	3:0	rwh	**Fixed clock 0 source signal selector** ***Note**: These bits are only writable if bit field CLSi_CCM_PROT.CLS_PROT is cleared.* 0_H Use CMU_FXCLK_RES[0:0] signal of CMU as CCM[i]_FXCLK_RES[0:0] signal within Cluster 1_H Use CMU_CLK_RES[8:8] signal of CMU as CCM[i]_FXCLK_RES[0:0] signal within Cluster **others,** Reserved
0	31:4	r	**Reserved** Read as 0; must be written with 0.

44.5.129 CCM[i] configuration register (i=0,2)

***Note**: The module-specific clock enable registers (bit field EN_*) are only implemented if the corresponding module is available in the i-th cluster.*

CLSi_CCM_CFG (i=0,2) Offset address: $0041F8_H+i*20000_H$

CCM[i] configuration register Kernel Reset value: $0002\ 0007_H$



Field	Bits	Type	Description
EN_TIM	0	rwh	**Enable TIM** ***Note:*** *This bit is only writable if bit field CLSi_CCM_PROT.CLS_PROT is cleared.* 0_B Disable clock signal for sub-module TIM 1_B Enable clock signal for sub-module TIM
EN_TOM_SPE_TDTM	1	rwh	**Enable TOM, SPE and TDTM** ***Note:*** *This bit is only writable if bit field CLSi_CCM_PROT.CLS_PROT is cleared.* 0_B Disable clock signal for modules TOM, SPE, and its related DTM modules 1_B Enable clock signal for modules TOM, SPE, and its related DTM modules
EN_ATOM_ADTM	2	rwh	**Enable ATOM and ADTM** ***Note:*** *This bit is only writable if bit field CLSi_CCM_PROT.CLS_PROT is cleared.* 0_B Disable clock signal for modules ATOM and its related DTM modules 1_B Enable clock signal for modules ATOM and its related DTM modules
CLS_CLK_DIV	17:16	rh	**Cluster Clock Divider** ***Note:*** *The value of this bit field mirrors the bit field CLSi_ARCH_CLK_CFG.CLSj_CLK_DIV **(j=i)** , whereas i equals the cluster index.* 00_B Cluster is disabled 01_B Cluster is enabled without clock divider 10_B Cluster is enabled with clock divider 2 **others,** Reserved

(table continues...)

(continued)

Field	Bits	Type	Description
0	3, 4, 5, 6, 7, 15:8, 29:18, 31, 30	r	**Reserved** Read as 0; must be written with 0.

44.5.130 CCM[i] configuration register (i=1)

For a description of this register, see CCM[i] configuration register (i=0,2).

CLSi_CCM_CFG (i=1) Offset address: $0041F8_H+i*20000_H$

CCM[i] configuration register Kernel Reset value: $0002\ 0087_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0												CLS_CLK_DIV	
r	r	r												rh	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EN_CMP_MON	0	0	0	0	EN_ATOM_ADTM	EN_TOM_SPE_TDTM	EN_TIM
r								rwh	r	r	r	r	rwh	rwh	rwh

Field	Bits	Type	Description
EN_TIM	0	rwh	**Enable TIM** ***Note:*** *This bit is only writable if bit field CLSi_CCM_PROT.CLS_PROT is cleared.* 0_B Disable clock signal for sub-module TIM 1_B Enable clock signal for sub-module TIM
EN_TOM_SPE_TDTM	1	rwh	**Enable TOM, SPE and TDTM** ***Note:*** *This bit is only writable if bit field CLSi_CCM_PROT.CLS_PROT is cleared.* 0_B Disable clock signal for modules TOM, SPE, and its related DTM modules 1_B Enable clock signal for modules TOM, SPE, and its related DTM modules

(table continues...)

(continued)

Field	Bits	Type	Description
EN_ATOM_ADTM	2	rwh	**Enable ATOM and** ADTM ***Note:*** *This bit is only writable if bit field* CLSi_CCM_PROT.CLS_PROT *is cleared.* 0_B Disable clock signal for modules ATOM and its related DTM modules 1_B Enable clock signal for modules ATOM and its related DTM modules
EN_CMP_MON	7	rwh	**Enable CMP and MON** ***Note:*** *This bit is only writable if bit field* CLSi_CCM_PROT.CLS_PROT *is cleared.* 0_B Disable clock signal for modules CMP and MON 1_B Enable clock signal for modules CMP and MON
CLS_CLK_DIV	17:16	rh	**Cluster Clock Divider** ***Note:*** *The value of this bit field mirrors the bit field* CLSi_ARCH_CLK_CFG.CLSj_CLK_DIV ***(j=i)*** *, whereas i equals the cluster index.* 00_B Cluster is disabled 01_B Cluster is enabled without clock divider 10_B Cluster is enabled with clock divider 2 **others,** Reserved
0	3, 4, 5, 6, 15:8, 29:18, 31, 30	r	**Reserved** Read as 0; must be written with 0.

44.5.131 CCM[i] protection register (i=0-2)

CLSi_CCM_PROT (i=0-2) Offset address: $0041FC_H+i*20000_H$

CCM[i] protection register Kernel Reset value: $0000\ 0001_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															CLS_PROT
r															rw

Field	Bits	Type	Description
CLS_PROT	0	rw	**Cluster Protection** 0_B Write-protection of cluster configuration registers disabled 1_B Write-protection of cluster configuration registers enabled
0	31:1	r	**Reserved** Read as 0; must be written with 0.

44.5.132 CDTM[i]_DTM[d] global configuration and control register (d=0-3;i=0-2)

CLSi_CDTM_DTMd_CTRL (d=0-3;i=0-2)

Offset address: 004400_H+i*20000_H+d*40_H

CDTM[i]_DTM[d] global configuration and control register

Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															SHUT_OFF_RST
r															rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0							SR_UPD_EN	CH_SHUTOFF_EN	UPD_MODE			DTM_SEL		CLK_SEL	
r							rw	rw	rw			rw		rw	

Field	Bits	Type	Description
CLK_SEL	1:0	rw	**Clock resolution selection** 00_B CLS[j]_CLK resolution selected (cluster clock) 01_B CCM[i]_CLK_RES[0:0] resolution in use, selected input DTM_CLK_RES[0:0] 10_B CCM[i]_FXCLK_RES[0:0] resolution in use, selected input DTM_CLK_RES[1:1](DTM is connected to TOM) 11_B CCM[i]_FXCLK_RES[1:1] resolution in use, selected input DTM_CLK_RES[2:2] (DTM is connected to TOM)
DTM_SEL	3:2	rw	**Select DTM update and PSU_SHUT_OFF reset signal** On value 00_B and 01_B shutoff is executed by signal TIM_CH_IN0, TIM_CH_IN1 or DTM_AUX_IN. On value 10_B and 11_B shutoff is executed by signal DTM_PSU_IN. 00_B select falling edge on DTM[d] channel 0 input 01_B select rising edge on DTM[d] channel 0 input 10_B select falling edge on DTM[d-1] channel 0 input 11_B select rising edge on DTM[d-1] channel 0 input

(table continues...)

(continued)

Field	Bits	Type	Description
UPD_MODE	6:4	rw	**Update mode** ***Note**: If IN_EDGE[x:x] is not implemented the value is unused.* 000_B asynchronous update - CLSi_CDTM_DTMd_CH_CTRL2_SR not used for update of CLSi_CDTM_DTMd_CH_CTRL2 001_B shutoff release by writing 1 to bit CLSi_CDTM_DTMd_CTRL.SHUT_OFF_RST 010_B shutoff release by signal DTM_PREV_IN0_REDGE or DTM_PREV_IN0_FEDGE or DTM_IN_REDGE or DTM_IN_FEDGE defined by bit field CLSi_CDTM_DTMd_CTRL.DTM_SEL 011_B shutoff release by shutoff signal PSU_SHUT_OFF (defined by bits CLSi_CDTM_DTMd_PS_CTRL.PSU_IN_SEL and CLSi_CDTM_DTMd_PS_CTRL.IN_POL and CLSi_CDTM_DTMd_CTRL.DTM_SEL[2:2]) 100_B Signal IN_EDGE[0:0] (from channel 0) used to trigger update of CLSi_CDTM_DTMd_CH_CTRL2 with content of CLSi_CDTM_DTMd_CH_CTRL2_SR 101_B Signal IN_EDGE[1:1] (from channel 1) used to trigger update of CLSi_CDTM_DTMd_CH_CTRL2 with content of CLSi_CDTM_DTMd_CH_CTRL2_SR 110_B Signal IN_EDGE[2:2] (from channel 2) used to trigger update of CLSi_CDTM_DTMd_CH_CTRL2 with content of CLSi_CDTM_DTMd_CH_CTRL2_SR 111_B Signal IN_EDGE[3:3] (from channel 3) used to trigger update of CLSi_CDTM_DTMd_CH_CTRL2 with content of CLSi_CDTM_DTMd_CH_CTRL2_SR
CH_SHUTOFF_EN	7	rw	**Individual shutoff feature enable** 0_B individual shutoff feature disabled 1_B individual shutoff feature enabled
SR_UPD_EN	8	rw	**Shadow register update enable** 0_B no update of CLSi_CDTM_DTMd_CH_CTRL2_SR.SL0_0_SR, CLSi_CDTM_DTMd_CH_CTRL2_SR.SL0_1_SR, CLSi_CDTM_DTMd_CH_CTRL2_SR.SL0_2_SR, CLSi_CDTM_DTMd_CH_CTRL2_SR.SL0_3_SR bits, CLSi_CDTM_DTMd_CH_CTRL2_SR.SL1_0_SR bits, CLSi_CDTM_DTMd_CH_CTRL2_SR.SL1_1_SR bits, CLSi_CDTM_DTMd_CH_CTRL2_SR.SL1_2_SR bits and CLSi_CDTM_DTMd_CH_CTRL2_SR.SL1_3_SR bits 1_B update of CLSi_CDTM_DTMd_CH_CTRL2_SR.SL0_0_SR, CLSi_CDTM_DTMd_CH_CTRL2_SR.SL0_1_SR, CLSi_CDTM_DTMd_CH_CTRL2_SR.SL0_2_SR, CLSi_CDTM_DTMd_CH_CTRL2_SR.SL0_3_SR bits, CLSi_CDTM_DTMd_CH_CTRL2_SR.SL1_0_SR bits, CLSi_CDTM_DTMd_CH_CTRL2_SR.SL1_1_SR bits, CLSi_CDTM_DTMd_CH_CTRL2_SR.SL1_2_SR bits and CLSi_CDTM_DTMd_CH_CTRL2_SR.SL1_3_SR bits on trigger

(table continues...)

(continued)

Field	Bits	Type	Description
SHUT_OFF_RST	16	rw	**Shutoff reset** ***Note:*** *Writing a 1_B releases shutoff (resets signal SHUTOFF_SYNC_0 if selected by CLSi_CDTM_DTMd_CTRL.UPD_MODE[2:0]=001_B)* **If reading:** 0_B No action **If writing:** 0_B No action 1_B shutoff reset
0	15:9, 31:17	r	**Reserved** Read as 0; must be written with 0.

44.5.133 CDTM[i]_DTM[d] global configuration and control register (d=4-5;i=0-2)

CLSi_CDTM_DTMd_CTRL (d=4-5;i=0-2)

Offset address: 004400_H+i*20000_H+d*40_H

CDTM[i]_DTM[d] global configuration and control register

Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															SHUT_OFF_RST
r															rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0							SR_UPD_EN	CH_SHUTOFF_EN	UPD_MODE			DTM_SEL		CLK_SEL	
r							rw	rw	rw			rw		rw	

Field	Bits	Type	Description
CLK_SEL	1:0	rw	**Clock resolution selection** 00_B CLS[j]_CLK resolution selected (cluster clock) 01_B CCM[i]_CLK_RES[0:0] resolution in use, selected input DTM_CLK_RES[0:0] 10_B CCM[i]_CLK_RES[1:1] resolution in use, selected input DTM_CLK_RES[1:1] (DTM is connected to ATOM) 11_B CCM[i]_CLK_RES[2:2] resolution in use, selected input DTM_CLK_RES[2:2] (DTM is connected to ATOM)

(table continues...)

(continued)

Field	Bits	Type	Description
DTM_SEL	3:2	rw	**Select DTM update and PSU_SHUT_OFF reset signal** On value 00_B and 01_B shutoff is executed by signal TIM_CH_IN0, TIM_CH_IN1 or DTM_AUX_IN. On value 10_B and 11_B shutoff is executed by signal DTM_PSU_IN. 00_B select falling edge on DTM[d] channel 0 input 01_B select rising edge on DTM[d] channel 0 input 10_B select falling edge on DTM[d-1] channel 0 input 11_B select rising edge on DTM[d-1] channel 0 input
UPD_MODE	6:4	rw	**Update mode** ***Note**: If IN_EDGE[x:x] is not implemented the value is unused.* 000_B asynchronous update - CLSi_CDTM_DTMd_CH_CTRL2_SR not used for update of CLSi_CDTM_DTMd_CH_CTRL2 001_B shutoff release by writing 1 to bit CLSi_CDTM_DTMd_CTRL.SHUT_OFF_RST 010_B shutoff release by signal DTM_PREV_IN0_REDGE or DTM_PREV_IN0_FEDGE or DTM_IN_REDGE or DTM_IN_FEDGE defined by bit field CLSi_CDTM_DTMd_CTRL.DTM_SEL 011_B shutoff release by shutoff signal PSU_SHUT_OFF (defined by bits CLSi_CDTM_DTMd_PS_CTRL.PSU_IN_SEL and CLSi_CDTM_DTMd_PS_CTRL.IN_POL and CLSi_CDTM_DTMd_CTRL.DTM_SEL[2:2]) 100_B Signal IN_EDGE[0:0] (from channel 0) used to trigger update of CLSi_CDTM_DTMd_CH_CTRL2 with content of CLSi_CDTM_DTMd_CH_CTRL2_SR 101_B Signal IN_EDGE[1:1] (from channel 1) used to trigger update of CLSi_CDTM_DTMd_CH_CTRL2 with content of CLSi_CDTM_DTMd_CH_CTRL2_SR 110_B Signal IN_EDGE[2:2] (from channel 2) used to trigger update of CLSi_CDTM_DTMd_CH_CTRL2 with content of CLSi_CDTM_DTMd_CH_CTRL2_SR 111_B Signal IN_EDGE[3:3] (from channel 3) used to trigger update of CLSi_CDTM_DTMd_CH_CTRL2 with content of CLSi_CDTM_DTMd_CH_CTRL2_SR
CH_SHUTOFF_EN	7	rw	**Individual shutoff feature enable** 0_B individual shutoff feature disabled 1_B individual shutoff feature enabled

(table continues...)

(continued)

Field	Bits	Type	Description
SR_UPD_EN	8	rw	**Shadow register update enable** 0_B no update of CLSi_CDTM_DTMd_CH_CTRL2_SR.SL0_0_SR, CLSi_CDTM_DTMd_CH_CTRL2_SR.SL0_1_SR, CLSi_CDTM_DTMd_CH_CTRL2_SR.SL0_2_SR, CLSi_CDTM_DTMd_CH_CTRL2_SR.SL0_3_SR bits, CLSi_CDTM_DTMd_CH_CTRL2_SR.SL1_0_SR bits, CLSi_CDTM_DTMd_CH_CTRL2_SR.SL1_1_SR bits, CLSi_CDTM_DTMd_CH_CTRL2_SR.SL1_2_SR bits and CLSi_CDTM_DTMd_CH_CTRL2_SR.SL1_3_SR bits 1_B update of CLSi_CDTM_DTMd_CH_CTRL2_SR.SL0_0_SR, CLSi_CDTM_DTMd_CH_CTRL2_SR.SL0_1_SR, CLSi_CDTM_DTMd_CH_CTRL2_SR.SL0_2_SR, CLSi_CDTM_DTMd_CH_CTRL2_SR.SL0_3_SR bits, CLSi_CDTM_DTMd_CH_CTRL2_SR.SL1_0_SR bits, CLSi_CDTM_DTMd_CH_CTRL2_SR.SL1_1_SR bits, CLSi_CDTM_DTMd_CH_CTRL2_SR.SL1_2_SR bits and CLSi_CDTM_DTMd_CH_CTRL2_SR.SL1_3_SR bits on trigger
SHUT_OFF_RST	16	rw	**Shutoff reset** ***Note:*** *Writing a 1_B releases shutoff (resets signal SHUTOFF_SYNC_0 if selected by CLSi_CDTM_DTMd_CTRL.UPD_MODE[2:0]=001_B)* **If reading:** 0_B No action **If writing:** 0_B No action 1_B shutoff reset
0	15:9, 31:17	r	**Reserved** Read as 0; must be written with 0.

44.5.134 CDTM[i]_DTM[d] channel control register 1 (d=0,2,4;i=0-2)

CLSi_CDTM_DTMd_CH_CTRL1 (d=0,2,4;i=0-2) Offset address: 004404_H+i*20000_H+d*40_H

CDTM[i]_DTM[d] channel control register 1 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		O1F_3		SWAP_3	SH_EN_3	I1SEL_3	O1SEL_3	0	XDT_EN_2_3	O1F_2		SWAP_2	SH_EN_2	I1SEL_2	O1SEL_2
r		rw		rw	rw	rw	rw	r	rw	rw		rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		O1F_1		SWAP_1	SH_EN_1	I1SEL_1	O1SEL_1	0	XDT_EN_0_1	O1F_0		SWAP_0	0	0	O1SEL_0
r		rw		rw	rw	rw	rw	r	rw	rw		rw	r	r	rw

Field	Bits	Type	Description
O1SEL_0	0	rw	**Output 1 select channel 0** 0_B inverse dead-time signal selected 1_B special function on output 1 selected (defined by CLSi_CDTM_DTMd_CH_CTRL1.O1F_0)
SWAP_0	3	rw	**Swap outputs DTM_OUT0*[0:0]* of instance d and DTM_OUT1*[0:0]* of instance d (before final output register)** 0_B outputs not swapped 1_B swap outputs DTM_OUT0[0:0] of instance d and DTM_OUT1[0:0] of instance d
O1F_0	5:4	rw	**Output 1 function channel 0** 00_B Signal EDGE_TRIGG_[x] of channel [x=0] is selected 01_B XOR of signal CII[x]_MUX of channel [x=0] of instance [d] and signal PSU_SHIFT[0:0] / DTM_COUT3_I 10_B AND of signal CII[x]_MUX of channel [x=0] of instance [d] and signal PSU_SHIFT[0:0] / DTM_COUT3_I 11_B DTM_IN_T[0:0] of instance d selected
XDT_EN_0_1	6	rw	**Cross dead-time enable on channel 0 and 1** ***Note**:* *When a '1' is written to bit CLSi_CDTM_DTMd_CH_CTRL1.XDT_EN_0_1, the internal register IN[x]_DLY1, IN[x]_DLY2 and the counter DT_DOWN_CNT are reset to '0' (x∈{0, 1}) Figure 1104* ***Note**:* *CLSi_CDTM_DTMd_CH_CTRL3.TSEL0_0 and CLSi_CDTM_DTMd_CH_CTRL1.SH_EN_1 must be '0' for using cross dead-time to avoid wrong input signals.* *CLSi_CDTM_DTMd_CH_CTRL3.TSEL0_1 and CLSi_CDTM_DTMd_CH_CTRL1.SH_EN_1 must be '0' for using cross dead-time to avoid wrong input signals.* 0_B cross dead-time disabled on channel 0 and 1 1_B cross dead-time enabled on channel 0 and 1
O1SEL_1	8	rw	**Output 1 select channel 1** 0_B inverse dead-time signal selected 1_B special function on output 1 selected (defined by CLSi_CDTM_DTMd_CH_CTRL1.O1F_1)
I1SEL_1	9	rw	**Input 1 select channel 1** 0_B signal PSU_SHIFT[1:1] selected 1_B signal COUT[0:0] selected
SH_EN_1	10	rw	**Shift enable channel 1** 0_B switch between DTM inputs of channel [x] and DTM_IN_PREV by PSU_SHIFT or CH_OUT_I (depends to CLSi_CDTM_DTMd_CH_CTRL1.I1SEL_1) is disabled 1_B switch between DTM inputs of channel [x] and DTM_IN_PREV by PSU_SHIFT or CH_OUT_I (depends to CLSi_CDTM_DTMd_CH_CTRL1.I1SEL_1) is enabled

(table continues...)

(continued)

Field	Bits	Type	Description
SWAP_1	11	rw	**Swap outputs DTM_OUT0*[1:1]* of instance d and DTM_OUT1*[1:1]* of instance d (before final output register)** 0_B outputs not swapped 1_B swap outputs DTM_OUT0[1:1] of instance d and DTM_OUT1[1:1] of instance d
O1F_1	13:12	rw	**Output 1 function channel 1** 00_B Signal EDGE_TRIGG_[x] of channel [x=1] is selected 01_B XOR of signal CII[x]_MUX of channel [x=1] of instance [d] and signal PSU_SHIFT[1:1] / COUT[0:0] 10_B AND of signal CII[x]_MUX of channel [x=1] of instance [d] and signal PSU_SHIFT[1:1] / COUT[0:0] 11_B DTM_IN_T[1:1] of instance d selected
O1SEL_2	16	rw	**Output 1 select channel 2** 0_B inverse dead-time signal selected 1_B special function on output 1 selected (defined by CLSi_CDTM_DTMd_CH_CTRL1.O1F_2)
I1SEL_2	17	rw	**Input 1 select channel 2** 0_B signal PSU_SHIFT[2:2] selected 1_B signal COUT[1:1] selected
SH_EN_2	18	rw	**Shift enable channel 2** 0_B switch between DTM inputs of channel [x] and DTM_IN_PREV by PSU_SHIFT or CH_OUT_I (depends to CLSi_CDTM_DTMd_CH_CTRL1.I1SEL_2) is disabled 1_B switch between DTM inputs of channel [x] and DTM_IN_PREV by PSU_SHIFT or CH_OUT_I (depends to CLSi_CDTM_DTMd_CH_CTRL1.I1SEL_2) is enabled
SWAP_2	19	rw	**Swap outputs DTM_OUT0*[2:2]* of instance d and DTM_OUT1*[2:2]* of instance d (before final output register)** 0_B outputs not swapped 1_B swap outputs DTM_OUT0[2:2] of instance d and DTM_OUT1[2:2] of instance d
O1F_2	21:20	rw	**Output 1 function channel 2** 00_B Signal EDGE_TRIGG_[x] of channel [x=2] is selected 01_B XOR of signal CII[x]_MUX of channel [x=2] of instance [d] and signal PSU_SHIFT[2:2] / COUT[1:1] 10_B AND of signal CII[x]_MUX of channel [x=2] of instance [d] and signal PSU_SHIFT[2:2] / COUT[1:1] 11_B DTM_IN_T[2:2] of instance d selected

(table continues...)

(continued)

Field	Bits	Type	Description
XDT_EN_2_3	22	rw	**Cross dead-time enable on channel 2 and 3** ***Note**: When a '1' is written to bit CLSi_CDTM_DTMd_CH_CTRL1.XDT_EN_2_3, the internal register IN[x]_DLY1, IN[x]_DLY2 and DT_DOWN_CNT are reset to '0' (x∈{2, 3})* ***Note**: CLSi_CDTM_DTMd_CH_CTRL3.TSEL0_2, CLSi_CDTM_DTMd_CH_CTRL3.TSEL0_3, CLSi_CDTM_DTMd_CH_CTRL1.SH_EN_2 and CLSi_CDTM_DTMd_CH_CTRL1.SH_EN_3 must be '0' for using cross dead-time to avoid wrong input signals.* 0_B cross dead-time disabled on channel 2 and 3 1_B cross dead-time enabled on channel 2 and 3
O1SEL_3	24	rw	**Output 1 select channel 3** 0_B inverse dead-time signal selected 1_B special function on output 1 selected (defined by CLSi_CDTM_DTMd_CH_CTRL1.O1F_3)
I1SEL_3	25	rw	**Input 1 select channel 3** 0_B signal PSU_SHIFT[3:3] selected 1_B signal COUT[2:2] selected
SH_EN_3	26	rw	**Shift enable channel 3** 0_B switch between DTM inputs of channel [x] and DTM_IN_PREV by PSU_SHIFT or CH_OUT_I (depends to CLSi_CDTM_DTMd_CH_CTRL1.I1SEL_3) is disabled 1_B switch between DTM inputs of channel [x] and DTM_IN_PREV by PSU_SHIFT or CH_OUT_I (depends to CLSi_CDTM_DTMd_CH_CTRL1.I1SEL_3) is enabled
SWAP_3	27	rw	**Swap outputs DTM_OUT0*[3:3]* of instance d and DTM_OUT1*[3:3]* of instance d (before final output register)** 0_B outputs not swapped 1_B swap outputs DTM_OUT0[3:3] of instance d and DTM_OUT1[3:3] of instance d
O1F_3	29:28	rw	**Output 1 function channel 3** 00_B Signal EDGE_TRIGG_[x] of channel [x=3] is selected 01_B XOR of signal CII[x]_MUX of channel [x=3] of instance [d] and signal PSU_SHIFT[3:3] / COUT[2:2] 10_B AND of signal CII[x]_MUX of channel [x=3] of instance [d] and signal PSU_SHIFT[3:3] / COUT[2:2] 11_B DTM_IN_T[3:3] of instance d selected

(table continues...)

(continued)

Field	Bits	Type	Description
0	1, 2, 7, 15:14, 23, 31:30	r	**Reserved** Read as 0; must be written with 0.

44.5.135 CDTM[i]_DTM[d] channel control register 1 (d=1,3,5;i=0-2)

CLSi_CDTM_DTMd_CH_CTRL1 (d=1,3,5;i=0-2) Offset address: 004404_H+i*20000_H+d*40_H

CDTM[i]_DTM[d] channel control register 1 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		O1F_3		SWAP_3	SH_EN_3	I1SEL_3	O1SEL_3	0	XDT_EN_2_3	O1F_2		SWAP_2	SH_EN_2	I1SEL_2	O1SEL_2
r		rw		rw	rw	rw	rw	r	rw	rw		rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		O1F_1		SWAP_1	SH_EN_1	I1SEL_1	O1SEL_1	0	XDT_EN_0_1	O1F_0		SWAP_0	0	I1SEL_0	O1SEL_0
r		rw		rw	rw	rw	rw	r	rw	rw		rw	r	rw	rw

Field	Bits	Type	Description
O1SEL_0	0	rw	**Output 1 select channel 0** 0_B inverse dead-time signal selected 1_B special function on output 1 selected (defined by CLSi_CDTM_DTMd_CH_CTRL1.O1F_0)
I1SEL_0	1	rw	**Input 1 select channel 0** ***Note**: If d is even, CLSi_CDTM_DTMd_CH_CTRL1.I1SEL_0 is not implemented.* 0_B signal PSU_SHIFT[0:0] selected 1_B signal DTM_COUT3_I from DTM[d-1] selected
SWAP_0	3	rw	**Swap outputs DTM_OUT0*[0:0]* of instance d and DTM_OUT1*[0:0]* of instance d (before final output register)** 0_B outputs not swapped 1_B swap outputs DTM_OUT0[0:0] of instance d and DTM_OUT1[0:0] of instance d

(table continues...)

(continued)

Field	Bits	Type	Description
O1F_0	5:4	rw	**Output 1 function channel 0** 00_B Signal EDGE_TRIGG_[x] of channel [x=0] is selected 01_B XOR of signal CII[x]_MUX of channel [x=0] of instance [d] and signal PSU_SHIFT[0:0] / DTM_COUT3_I 10_B AND of signal CII[x]_MUX of channel [x=0] of instance [d] and signal PSU_SHIFT[0:0] / DTM_COUT3_I 11_B DTM_IN_T[0:0] of instance d selected
XDT_EN_0_1	6	rw	**Cross dead-time enable on channel 0 and 1** ***Note**:* *When a '1' is written to bit CLSi_CDTM_DTMd_CH_CTRL1.XDT_EN_0_1, the internal register IN[x]_DLY1, IN[x]_DLY2 and the counter DT_DOWN_CNT are reset to '0' (x∈{0, 1})* Figure 1104 ***Note**:* CLSi_CDTM_DTMd_CH_CTRL3.TSEL0_0 *and* CLSi_CDTM_DTMd_CH_CTRL1.SH_EN_1 *must be '0' for using cross dead-time to avoid wrong input signals.* CLSi_CDTM_DTMd_CH_CTRL3.TSEL0_1 *and* CLSi_CDTM_DTMd_CH_CTRL1.SH_EN_1 *must be '0' for using cross dead-time to avoid wrong input signals.* 0_B cross dead-time disabled on channel 0 and 1 1_B cross dead-time enabled on channel 0 and 1
O1SEL_1	8	rw	**Output 1 select channel 1** 0_B inverse dead-time signal selected 1_B special function on output 1 selected (defined by CLSi_CDTM_DTMd_CH_CTRL1.O1F_1)
I1SEL_1	9	rw	**Input 1 select channel 1** 0_B signal PSU_SHIFT[1:1] selected 1_B signal COUT[0:0] selected
SH_EN_1	10	rw	**Shift enable channel 1** 0_B switch between DTM inputs of channel [x] and DTM_IN_PREV by PSU_SHIFT or CH_OUT_I (depends to CLSi_CDTM_DTMd_CH_CTRL1.I1SEL_1) is disabled 1_B switch between DTM inputs of channel [x] and DTM_IN_PREV by PSU_SHIFT or CH_OUT_I (depends to CLSi_CDTM_DTMd_CH_CTRL1.I1SEL_1) is enabled
SWAP_1	11	rw	**Swap outputs DTM_OUT0*[1:1]* of instance d and DTM_OUT1*[1:1]* of instance d (before final output register)** 0_B outputs not swapped 1_B swap outputs DTM_OUT0[1:1] of instance d and DTM_OUT1[1:1] of instance d

(table continues...)

(continued)

Field	Bits	Type	Description
O1F_1	13:12	rw	**Output 1 function channel 1** 00_B Signal EDGE_TRIGG_[x] of channel [x=1] is selected 01_B XOR of signal CII[x]_MUX of channel [x=1] of instance [d] and signal PSU_SHIFT[1:1] / COUT[0:0] 10_B AND of signal CII[x]_MUX of channel [x=1] of instance [d] and signal PSU_SHIFT[1:1] / COUT[0:0] 11_B DTM_IN_T[1:1] of instance d selected
O1SEL_2	16	rw	**Output 1 select channel 2** 0_B inverse dead-time signal selected 1_B special function on output 1 selected (defined by CLSi_CDTM_DTMd_CH_CTRL1.O1F_2)
I1SEL_2	17	rw	**Input 1 select channel 2** 0_B signal PSU_SHIFT[2:2] selected 1_B signal COUT[1:1] selected
SH_EN_2	18	rw	**Shift enable channel 2** 0_B switch between DTM inputs of channel [x] and DTM_IN_PREV by PSU_SHIFT or CH_OUT_I (depends to CLSi_CDTM_DTMd_CH_CTRL1.I1SEL_2) is disabled 1_B switch between DTM inputs of channel [x] and DTM_IN_PREV by PSU_SHIFT or CH_OUT_I (depends to CLSi_CDTM_DTMd_CH_CTRL1.I1SEL_2) is enabled
SWAP_2	19	rw	**Swap outputs** DTM_OUT0***[2:2]*** **of instance d and** DTM_OUT1***[2:2]*** **of instance d (before final output register)** 0_B outputs not swapped 1_B swap outputs DTM_OUT0[2:2] of instance d and DTM_OUT1[2:2] of instance d
O1F_2	21:20	rw	**Output 1 function channel 2** 00_B Signal EDGE_TRIGG_[x] of channel [x=2] is selected 01_B XOR of signal CII[x]_MUX of channel [x=2] of instance [d] and signal PSU_SHIFT[2:2] / COUT[1:1] 10_B AND of signal CII[x]_MUX of channel [x=2] of instance [d] and signal PSU_SHIFT[2:2] / COUT[1:1] 11_B DTM_IN_T[2:2] of instance d selected

(table continues...)

(continued)

Field	Bits	Type	Description
XDT_EN_2_3	22	rw	**Cross dead-time enable on channel 2 and 3** ***Note**: When a '1' is written to bit CLSi_CDTM_DTMd_CH_CTRL1.XDT_EN_2_3, the internal register IN[x]_DLY1, IN[x]_DLY2 and DT_DOWN_CNT are reset to '0' (x∈{2, 3})* ***Note**: CLSi_CDTM_DTMd_CH_CTRL3.TSEL0_2, CLSi_CDTM_DTMd_CH_CTRL3.TSEL0_3, CLSi_CDTM_DTMd_CH_CTRL1.SH_EN_2 and CLSi_CDTM_DTMd_CH_CTRL1.SH_EN_3 must be '0' for using cross dead-time to avoid wrong input signals.* 0_B cross dead-time disabled on channel 2 and 3 1_B cross dead-time enabled on channel 2 and 3
O1SEL_3	24	rw	**Output 1 select channel 3** 0_B inverse dead-time signal selected 1_B special function on output 1 selected (defined by CLSi_CDTM_DTMd_CH_CTRL1.O1F_3)
I1SEL_3	25	rw	**Input 1 select channel 3** 0_B signal PSU_SHIFT[3:3] selected 1_B signal COUT[2:2] selected
SH_EN_3	26	rw	**Shift enable channel 3** 0_B switch between DTM inputs of channel [x] and DTM_IN_PREV by PSU_SHIFT or CH_OUT_I (depends to CLSi_CDTM_DTMd_CH_CTRL1.I1SEL_3) is disabled 1_B switch between DTM inputs of channel [x] and DTM_IN_PREV by PSU_SHIFT or CH_OUT_I (depends to CLSi_CDTM_DTMd_CH_CTRL1.I1SEL_3) is enabled
SWAP_3	27	rw	**Swap outputs DTM_OUT0*[3:3]* of instance d and DTM_OUT1*[3:3]* of instance d (before final output register)** 0_B outputs not swapped 1_B swap outputs DTM_OUT0[3:3] of instance d and DTM_OUT1[3:3] of instance d
O1F_3	29:28	rw	**Output 1 function channel 3** 00_B Signal EDGE_TRIGG_[x] of channel [x=3] is selected 01_B XOR of signal CII[x]_MUX of channel [x=3] of instance [d] and signal PSU_SHIFT[3:3] / COUT[2:2] 10_B AND of signal CII[x]_MUX of channel [x=3] of instance [d] and signal PSU_SHIFT[3:3] / COUT[2:2] 11_B DTM_IN_T[3:3] of instance d selected
0	2, 7, 15:14, 23, 31:30	r	**Reserved** Read as 0; must be written with 0.

44.5.136 CDTM[i]_DTM[d] channel control register 2 (d=0-5;i=0-2)

CLSi_CDTM_DTMd_CH_CTRL2 (d=0-5;i=0-2) Offset address: 004408_H+i*20000_H+d*40_H

CDTM[i]_DTM[d] channel control register 2 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DT1_3	SL1_3	OC1_3	POL1_3	DT0_3	SL0_3	OC0_3	POL0_3	DT1_2	SL1_2	OC1_2	POL1_2	DT0_2	SL0_2	OC0_2	POL0_2
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DT1_1	SL1_1	OC1_1	POL1_1	DT0_1	SL0_1	OC0_1	POL0_1	DT1_0	SL1_0	OC1_0	POL1_0	DT0_0	SL0_0	OC0_0	POL0_0
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
POL0_0	0	rwh	**Polarity on output 0 channel 0** 0_B Output signal not inverted 1_B Output signal inverted
OC0_0	1	rwh	**Output 0 control channel 0** 0_B Functional output 1_B Constant output defined by CLSi_CDTM_DTMd_CH_CTRL2.SL0_0
SL0_0	2	rwh	**Signal level on output 0 channel 0** 0_B set Signal Level to 0 1_B set Signal Level to 1
DT0_0	3	rwh	**Dead-time path enable on output 0 channel 0** 0_B direct path from DTM_IN[0:0] to DTM_OUT0[0.0] of instance d enabled 1_B dead-time path enabled
POL1_0	4	rwh	**Polarity on output 1 channel 0** 0_B Output signal not inverted 1_B Output signal inverted
OC1_0	5	rwh	**Output 1 control channel 0** 0_B Functional output 1_B Constant output defined by CLSi_CDTM_DTMd_CH_CTRL2.SL1_0
SL1_0	6	rwh	**Signal level on output 1 channel 0** 0_B set Signal Level to 0 1_B set Signal Level to 1
DT1_0	7	rwh	**Dead-time path enable on output 1 channel 0** 0_B direct path from DTM_IN[0:0] to DTM_OUT1[0:0] of instance d enabled 1_B path for DTM functions enabled like dead-time or multiple signal combinations

(table continues...)

(continued)

Field	Bits	Type	Description
POL0_1	8	rwh	**Polarity on output 0 channel 1** 0_B Output signal not inverted 1_B Output signal inverted
OC0_1	9	rwh	**Output 0 control channel 1** 0_B Functional output 1_B Constant output defined by CLSi_CDTM_DTMd_CH_CTRL2.SL0_1
SL0_1	10	rwh	**Signal level on output 0 channel 1** 0_B set Signal Level to 0 1_B set Signal Level to 1
DT0_1	11	rwh	**Dead-time path enable on output 0 channel 1** 0_B direct path from DTM_IN[1:1] to DTM_OUT0[1:1] of instance d enabled 1_B dead-time path enabled
POL1_1	12	rwh	**Polarity on output 1 channel 1** 0_B Output signal not inverted 1_B Output signal inverted
OC1_1	13	rwh	**Output 1 control channel 1** 0_B Functional output 1_B Constant output defined by CLSi_CDTM_DTMd_CH_CTRL2.SL1_1
SL1_1	14	rwh	**Signal level on output 1 channel 1** 0_B set Signal Level to 0 1_B set Signal Level to 1
DT1_1	15	rwh	**Dead-time path enable on output 1 channel 1** 0_B direct path from DTM_IN[1:1] to DTM_OUT1[1:1] of instance d enabled 1_B path for DTM functions enabled like dead-time or multiple signal combinations
POL0_2	16	rwh	**Polarity on output 0 channel 2** 0_B Output signal not inverted 1_B Output signal inverted
OC0_2	17	rwh	**Output 0 control channel 2** 0_B Functional output 1_B Constant output defined by CLSi_CDTM_DTMd_CH_CTRL2.SL0_2
SL0_2	18	rwh	**Signal level on output 0 channel 2** 0_B set Signal Level to 0 1_B set Signal Level to 1
DT0_2	19	rwh	**Dead-time path enable on output 0 channel 2** 0_B direct path from DTM_IN[2:2] to DTM_OUT0[2:2] of instance d enabled 1_B dead-time path enabled

(table continues...)

(continued)

Field	Bits	Type	Description
POL1_2	20	rwh	**Polarity on output 1 channel 2** 0_B Output signal not inverted 1_B Output signal inverted
OC1_2	21	rwh	**Output 1 control channel 2** 0_B Functional output 1_B Constant output defined by CLSi_CDTM_DTMd_CH_CTRL2.SL1_2
SL1_2	22	rwh	**Signal level on output 1 channel 2** 0_B set Signal Level to 0 1_B set Signal Level to 1
DT1_2	23	rwh	**Dead-time path enable on output 1 channel 2** 0_B direct path from DTM_IN[2:2] to DTM_OUT1[2:2] of instance d enabled 1_B path for DTM functions enabled like dead-time or multiple signal combinations
POL0_3	24	rwh	**Polarity on output 0 channel 3** 0_B Output signal not inverted 1_B Output signal inverted
OC0_3	25	rwh	**Output 0 control channel 3** 0_B Functional output 1_B Constant output defined by CLSi_CDTM_DTMd_CH_CTRL2.SL0_3
SL0_3	26	rwh	**Signal level on output 0 channel 3** 0_B set Signal Level to 0 1_B set Signal Level to 1
DT0_3	27	rwh	**Dead-time path enable on output 0 channel 3** 0_B direct path from DTM_IN[3:3] of instance d to DTM_OUT0[3:3] of instance d enabled 1_B dead-time path enabled
POL1_3	28	rwh	**Polarity on output 1 channel 3** 0_B Output signal not inverted 1_B Output signal inverted
OC1_3	29	rwh	**Output 1 control channel 3** 0_B Functional output 1_B Constant output defined by CLSi_CDTM_DTMd_CH_CTRL2.SL1_3
SL1_3	30	rwh	**Signal level on output 1 channel 3** 0_B set Signal Level to 0 1_B set Signal Level to 1
DT1_3	31	rwh	**Dead-time path enable on output 1 channel 3** 0_B direct path from DTM_IN[3:3] to DTM_OUT1[3:3] of instance d enabled 1_B path for DTM functions enabled like dead-time or multiple signal combinations

44.5.137 CDTM[i] DTM[d] channel control register 2 shadow (d=0-5;i=0-2)

CLSi_CDTM_DTMd_CH_CTRL2_SR (d=0-5;i=0-2) Offset address: $00440C_H+i*20000_H+d*40_H$

CDTM[i] DTM[d] channel control register 2 shadow Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DT1_3_SR	SL1_3_SR	OC1_3_SR	POL1_3_SR	DT0_3_SR	SL0_3_SR	OC0_3_SR	POL0_3_SR	DT1_2_SR	SL1_2_SR	OC1_2_SR	POL1_2_SR	DT0_2_SR	SL0_2_SR	OC0_2_SR	POL0_2_SR
rw	rwh	rw	rw	rw	rwh	rw	rw	rw	rwh	rw	rw	rw	rwh	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DT1_1_SR	SL1_1_SR	OC1_1_SR	POL1_1_SR	DT0_1_SR	SL0_1_SR	OC0_1_SR	POL0_1_SR	DT1_0_SR	SL1_0_SR	OC1_0_SR	POL1_0_SR	DT0_0_SR	SL0_0_SR	OC0_0_SR	POL0_0_SR
rw	rwh	rw	rw	rw	rwh	rw	rw	rw	rwh	rw	rw	rw	rwh	rw	rw

Field	Bits	Type	Description
POL0_0_SR	0	rw	**Polarity on output 0 channel 0 shadow register** 0_B Output signal not inverted 1_B Output signal inverted
OC0_0_SR	1	rw	**Output 0 control channel 0 shadow register** 0_B Functional output 1_B Constant output defined by CLSi_CDTM_DTMd_CH_CTRL2_SR.SL0_0_SR
SL0_0_SR	2	rwh	**Signal level on output 0 channel 0 shadow register** 0_B set Signal Level to 0 1_B set Signal Level to 1
DT0_0_SR	3	rw	**Dead-time path enable on output 0 channel 0 shadow register** 0_B direct path from DTM_IN[0:0] to DTM_OUT0[0:0] of instance d enabled 1_B dead-time path enabled
POL1_0_SR	4	rw	**Polarity on output 1 channel 0 shadow register** 0_B Output signal not inverted 1_B Output signal inverted
OC1_0_SR	5	rw	**Output 1 control channel 0 shadow register** 0_B Functional output 1_B Constant output defined by CLSi_CDTM_DTMd_CH_CTRL2_SR.SL1_0_SR
SL1_0_SR	6	rwh	**Signal level on output 1 channel 0 shadow register** 0_B set Signal Level to 0 1_B set Signal Level to 1

(table continues...)

(continued)

Field	Bits	Type	Description
DT1_0_SR	7	rw	**Dead-time path enable on output 1 channel 0 shadow register** 0_B direct path from DTM_IN[0:0] to DTM_OUT1[0:0] of instance d enabled 1_B path for DTM functions enabled like dead-time or multiple signal combinations
POL0_1_SR	8	rw	**Polarity on output 0 channel 1 shadow register** 0_B Output signal not inverted 1_B Output signal inverted
OC0_1_SR	9	rw	**Output 0 control channel 1 shadow register** 0_B Functional output 1_B Constant output defined by CLSi_CDTM_DTMd_CH_CTRL2_SR.SL0_1_SR
SL0_1_SR	10	rwh	**Signal level on output 0 channel 1 shadow register** 0_B set Signal Level to 0 1_B set Signal Level to 1
DT0_1_SR	11	rw	**Dead-time path enable on output 0 channel 1 shadow register** 0_B direct path from DTM_IN[1:1] to DTM_OUT0[1:1] of instance d enabled 1_B dead-time path enabled
POL1_1_SR	12	rw	**Polarity on output 1 channel 1 shadow register** 0_B Output signal not inverted 1_B Output signal inverted
OC1_1_SR	13	rw	**Output 1 control channel 1 shadow register** 0_B Functional output 1_B Constant output defined by CLSi_CDTM_DTMd_CH_CTRL2_SR.SL1_1_SR
SL1_1_SR	14	rwh	**Signal level on output 1 channel 1 shadow register** 0_B set Signal Level to 0 1_B set Signal Level to 1
DT1_1_SR	15	rw	**Dead-time path enable on output 1 channel 1 shadow register** 0_B direct path from DTM_IN[1:1] to DTM_OUT1[1:1] of instance d 1_B path for DTM functions enabled like dead-time or multiple signal combinations
POL0_2_SR	16	rw	**Polarity on output 0 channel 2 shadow register** 0_B Output signal not inverted 1_B Output signal inverted
OC0_2_SR	17	rw	**Output 0 control channel 2 shadow register** 0_B Functional output 1_B Constant output defined by CLSi_CDTM_DTMd_CH_CTRL2_SR.SL0_2_SR

(table continues...)

(continued)

Field	Bits	Type	Description
SL0_2_SR	18	rwh	**Signal level on output 0 channel 2 shadow register** 0_B set Signal Level to 0 1_B set Signal Level to 1
DT0_2_SR	19	rw	**Dead-time path enable on output 0 channel 2 shadow register** 0_B direct path from DTM_IN[2:2] to DTM_OUT0[2:2] of instance d 1_B dead-time path enabled
POL1_2_SR	20	rw	**Polarity on output 1 channel 2 shadow register** 0_B Output signal not inverted 1_B Output signal inverted
OC1_2_SR	21	rw	**Output 1 control channel 2 shadow register** 0_B Functional output 1_B Constant output defined by CLSi_CDTM_DTMd_CH_CTRL2_SR.SL1_2_SR
SL1_2_SR	22	rwh	**Signal level on output 1 channel 2 shadow register** 0_B set Signal Level to 0 1_B set Signal Level to 1
DT1_2_SR	23	rw	**Dead-time path enable on output 1 channel 2 shadow register** 0_B direct path from DTM_IN[2:2] to DTM_OUT1[2:2] of instance d 1_B path for DTM functions enabled like dead-time or multiple signal combinations
POL0_3_SR	24	rw	**Polarity on output 0 channel 3 shadow register** 0_B Output signal not inverted 1_B Output signal inverted
OC0_3_SR	25	rw	**Output 0 control channel 3 shadow register** 0_B Functional output 1_B Constant output defined by CLSi_CDTM_DTMd_CH_CTRL2_SR.SL0_3_SR
SL0_3_SR	26	rwh	**Signal level on output 0 channel 3 shadow register** 0_B set Signal Level to 0 1_B set Signal Level to 1
DT0_3_SR	27	rw	**Dead-time path enable on output 0 channel 3 shadow register** 0_B direct path from DTM_IN[3:3] to DTM_OUT0[3:3] of instance d 1_B dead-time path enabled
POL1_3_SR	28	rw	**Polarity on output 1 channel 3 shadow register** 0_B Output signal not inverted 1_B Output signal inverted
OC1_3_SR	29	rw	**Output 1 control channel 3 shadow register** 0_B Functional output 1_B Constant output defined by CLSi_CDTM_DTMd_CH_CTRL2_SR.SL1_3_SR

(table continues...)

(continued)

Field	Bits	Type	Description
SL1_3_SR	30	rwh	**Signal level on output 1 channel 3 shadow register** 0_B set Signal Level to 0 1_B set Signal Level to 1
DT1_3_SR	31	rw	**Dead-time path enable on output 1 channel 3 shadow register** 0_B direct path from DTM_IN[3:3] to DTM_OUT1[3:3] of instance d 1_B path for DTM functions enabled like dead-time or multiple signal combinations

44.5.138 CDTM[i]_DTM[d] phase shift unit configuration and control register (d=0-5;i=0-2)

CLSi_CDTM_DTMd_PS_CTRL (d=0-5;i=0-2) Offset address: 004410_H+i*20000_H+d*40_H

CDTM[i]_DTM[d] phase shift unit configuration and control register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0										SHIFT_SEL		0	TIM_SEL	IN_POL	PSU_IN_SEL
r										rw		r	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0						RELBLK									
r						rw									

Field	Bits	Type	Description
RELBLK	9:0	rw	**Reload value blanking window** ***Note:*** *A value of 000_H resets counter* BLK_DOWN_CNT
PSU_IN_SEL	16	rw	**PSU input select** 0_B TIM_CH_IN0 or TIM_CH_IN1 selected 1_B DTM_AUX_IN selected
IN_POL	17	rw	**Input polarity** 0_B input signal is not inverted 1_B input signal is inverted
TIM_SEL	18	rw	**TIM input select** 0_B select TIM_CH_IN0 1_B select TIM_CH_IN1

(table continues...)

(continued)

Field	Bits	Type	Description
SHIFT_SEL	21:20	rw	**Shift select** ***Note**: If a channel is not implemented the value is unused. A write with this unused value returns "10" on status.* 00_B trigger signal for reloading DOWN COUNTER is IN_EDGE[0:0] of DTM channel [0]; signal TIM_CH_IN0 or TIM_CH_IN1 or DTM_AUX_IN[0:0] or DTM_PSU_IN via signal EXT_CTRL_GATED is connected via signal PSU_SHIFT[1:1] to DTM channel 1 01_B trigger signal for reloading DOWN COUNTER is IN_EDGE[1:1] of DTM channel [1]; signal TIM_CH_IN0 or TIM_CH_IN1 or DTM_AUX_IN[0:0] or DTM_PSU_IN via signal EXT_CTRL_GATED is connected via signal PSU_SHIFT[2:2] to DTM channel 2 10_B trigger signal for reloading DOWN COUNTER is IN_EDGE[2:2] of DTM channel [2]; signal TIM_CH_IN0 or TIM_CH_IN1 or DTM_AUX_IN[0:0] or DTM_PSU_IN via signal EXT_CTRL_GATED is connected via signal PSU_SHIFT[3:3] to DTM channel 3 11_B no loading of DOWN COUNTER (blanking window feature deactivated); signal TIM_CH_IN0 or TIM_CH_IN1 or DTM_AUX_IN[0:0] or DTM_PSU_IN via signal EXT_CTRL_GATED is connected via signal PSU_SHIFT[0:0] to DTM channel 0
0	15:10, 19, 31:22	r	**Reserved** Read as 0; must be written with 0.

44.5.139 CDTM[i]_DTM[d] channel [x] dead-time reload values (x=0-3;i=0-2;d=0-3)

CLSi_CDTM_DTMd_CH_DTVx (d=0-3;i=0-2;x=0-3) Offset address: 004414_H+i*20000_H+d*40_H+x*4

CDTM[i]_DTM[d] channel [x] dead-time reload values Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		0							RELFALL						
r		r							rwh						

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			RELRISE												
r			rwh												

Field	Bits	Type	Description
RELRISE	12:0	rwh	**Reload value for rising edge dead-time**
RELFALL	28:16	rwh	**Reload value for falling edge dead-time**
0	15:13, 30:29, 31	r	**Reserved** Read as 0; must be written with 0.

44.5.140 CDTM[i]_DTM[d] channel [x] dead-time reload values (x=0-3;i=0-2;d=4-5)

CLSi_CDTM_DTMd_CH_DTVx (d=4-5;i=0-2;x=0-3) Offset address: 004414_H+i*20000_H+d*40_H+x*4

CDTM[i]_DTM[d] channel [x] dead-time reload values Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
HRES	0		RELFALL												
rw	r		rwh												

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			RELRISE												
r			rwh												

Field	Bits	Type	Description
RELRISE	12:0	rwh	**Reload value for rising edge dead-time**
RELFALL	28:16	rwh	**Reload value for falling edge dead-time**
HRES	31	rw	**high-resolution PWM support** 0_B high-resolution PWM support off 1_B high-resolution PWM support on
0	15:13, 30:29	r	**Reserved** Read as 0; must be written with 0.

44.5.141 CDTM[i]_DTM[d] channel shadow register (d=0-5;i=0-2)

CLSi_CDTM_DTMd_CH_SR (d=0-5;i=0-2) Offset address: 004424_H+i*20000_H+d*40_H

CDTM[i]_DTM[d] channel shadow register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								SL1_3_SR_SR	SL0_3_SR_SR	SL1_2_SR_SR	SL0_2_SR_SR	SL1_1_SR_SR	SL0_1_SR_SR	SL1_0_SR_SR	SL0_0_SR_SR
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
SL0_0_SR_SR	0	rw	**Shadow register for bit** CLSi_CDTM_DTMd_CH_CTRL2_SR.SL0_0_SR
SL1_0_SR_SR	1	rw	**Shadow register for bit** CLSi_CDTM_DTMd_CH_CTRL2_SR.SL1_0_SR
SL0_1_SR_SR	2	rw	**Shadow register for bit** CLSi_CDTM_DTMd_CH_CTRL2_SR.SL0_1_SR

(table continues...)

(continued)

Field	Bits	Type	Description
SL1_1_SR_SR	3	rw	**Shadow register for bit** CLSi_CDTM_DTMd_CH_CTRL2_SR.SL1_1_SR
SL0_2_SR_SR	4	rw	**Shadow register for bit** CLSi_CDTM_DTMd_CH_CTRL2_SR.SL0_2_SR
SL1_2_SR_SR	5	rw	**Shadow register for bit** CLSi_CDTM_DTMd_CH_CTRL2_SR.SL1_2_SR
SL0_3_SR_SR	6	rw	**Shadow register for bit** CLSi_CDTM_DTMd_CH_CTRL2_SR.SL0_3_SR
SL1_3_SR_SR	7	rw	**Shadow register for bit** CLSi_CDTM_DTMd_CH_CTRL2_SR.SL1_3_SR
0	31:8	r	**Reserved** Read as 0; must be written with 0.

44.5.142 CDTM[i]_DTM[d] channel control register 3 (d=0-5;i=0-2)

CLSi_CDTM_DTMd_CH_CTRL3 (d=0-5;i=0-2) Offset address: 004428_H+i*20000_H+d*40_H

CDTM[i]_DTM[d] channel control register 3 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0				TSEL1_3	TSEL0_3	CIS3	CII3	0				TSEL1_2	TSEL0_2	CIS2	CII2
r				rw	rw	rw	rw	r				rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				TSEL1_1	TSEL0_1	CIS1	CII1	0				TSEL1_0	TSEL0_0	CIS0	CII0
r				rw	rw	rw	rw	r				rw	rw	rw	rw

Field	Bits	Type	Description
CII0	0	rw	**Combinational input invert channel 0** 0_B do not invert input 1_B invert input
CIS0	1	rw	**Combinational input select channel 0** 0_B select the input DTM_IN[0:0] or DTM_IN_T[0:0] of instance d (CLSi_CDTM_DTMd_CH_CTRL3.TSEL0_[0] selects one of the two input signals) 1_B select internal signal EDGE_TRIGG_[x]
TSEL0_0	2	rw	**Input selection for dead-time / edge trigger generation** 0_B use DTM_IN[0:0] of instance d as input for dead-time / edge trigger generation 1_B use DTM_IN_T[0:0] of instance d as input for dead-time / edge trigger generation
TSEL1_0	3	rw	**Input selection combinational logic path** 0_B use DTM_IN[0:0] of instance d as input for combinational logic path 1_B use DTM_IN_T[0:0] of instance d as input for combinational logic path

(table continues...)

(continued)

Field	Bits	Type	Description
CII1	8	rw	**Combinational input invert channel 1** 0_B do not invert input 1_B invert input
CIS1	9	rw	**Combinational input select channel 1** 0_B select the input DTM_IN[1:1] or DTM_IN_T[1:1] of instance d (CLSi_CDTM_DTMd_CH_CTRL3.TSEL0_[1] selects one of the two input signals) 1_B select internal signal EDGE_TRIGG_[x]
TSEL0_1	10	rw	**Input selection for dead-time / edge trigger generation** 0_B use DTM_IN[1:1] of instance d as input for dead-time / edge trigger generation 1_B use DTM_IN_T[1:1] of instance d as input for dead-time / edge trigger generation
TSEL1_1	11	rw	**Input selection combinational logic path** 0_B use DTM_IN[1:1] of instance d as input for combinational logic path 1_B use DTM_IN_T[1:1] of instance d as input for combinational logic path
CII2	16	rw	**Combinational input invert channel 2** 0_B do not invert input 1_B invert input
CIS2	17	rw	**Combinational input select channel 2** 0_B select the input DTM_IN[2:2] or DTM_IN_T[2:2] of instance d (CLSi_CDTM_DTMd_CH_CTRL3.TSEL0_[2] selects one of the two input signals) 1_B select internal signal EDGE_TRIGG_[x]
TSEL0_2	18	rw	**Input selection for dead-time / edge trigger generation** 0_B use DTM_IN[2:2] of instance d as input for dead-time / edge trigger generation 1_B use DTM_IN_T[2:2] of instance d as input for dead-time / edge trigger generation
TSEL1_2	19	rw	**Input selection combinational logic path** 0_B use DTM_IN[2:2] of instance d as input for combinational logic path 1_B use DTM_IN_T[2:2] of instance d as input for combinational logic path
CII3	24	rw	**Combinational input invert channel 3** 0_B do not invert input 1_B invert input

(table continues...)

(continued)

Field	Bits	Type	Description
CIS3	25	rw	**Combinational input select channel 3** 0_B select the input DTM_IN[3:3] or DTM_IN_T[3:3] of instance d (CLSi_CDTM_DTMd_CH_CTRL3.TSEL0_[3] selects one of the two input signals) 1_B select internal signal EDGE_TRIGG_[x]
TSEL0_3	26	rw	**Input selection for dead-time / edge trigger generation** 0_B use DTM_IN[3:3] of instance d as input for dead-time / edge trigger generation 1_B use DTM_IN_T[3:3] of instance d as input for dead-time / edge trigger generation
TSEL1_3	27	rw	**Input selection combinational logic path** 0_B use DTM_IN[3:3] of instance d as input for combinational logic path 1_B use DTM_IN_T[3:3] of instance d as input for combinational logic path
0	7:4, 15:12, 23:20, 31:28	r	**Reserved** Read as 0; must be written with 0.

44.5.143 CDTM[i]_DTM[d] global configuration and control register 2 (d=0-5;i=0-2)

CLSi_CDTM_DTMd_CTRL2 (d=0-5;i=0-2) Offset address: $00442C_H+i*20000_H+d*40_H$

CDTM[i]_DTM[d] global configuration and control register 2 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
WR_EN_3	SHUT_OFF_RST_3	UPD_MODE_3		SHUTOFF_POL_3	SHUTOFF_SEL_3			WR_EN_2	SHUT_OFF_RST_2	UPD_MODE_2		SHUTOFF_POL_2	SHUTOFF_SEL_2		
rw	rw	rw		rw	rw			rw	rw	rw		rw	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WR_EN_1	SHUT_OFF_RST_1	UPD_MODE_1		SHUTOFF_POL_1	SHUTOFF_SEL_1			WR_EN_0	SHUT_OFF_RST_0	UPD_MODE_0		SHUTOFF_POL_0	SHUTOFF_SEL_0		
rw	rw	rw		rw	rw			rw	rw	rw		rw	rw		

Field	Bits	Type	Description
SHUTOFF_SEL_0	2:0	rw	**Channel 0: Select input signal to be used as shutoff signal.** ***Note**: This bit field is write-protected by bit CLSi_CDTM_DTMd_CTRL2.WR_EN_0 = 0.* 000_B TIM_CH_IN0 selected 001_B TIM_CH_IN1 selected 010_B DTM_AUX_IN[2:2] of instance d (DTM_AUX_IN[0:0] of instance d-1) selected 011_B DTM_AUX_IN[3:3] of instance d (DTM_AUX_IN[1:1] of instance d-1) selected 100_B DTM_AUX_IN[0:0] of instance d selected 101_B DTM_AUX_IN[1:1] of instance d selected 110_B '0' selected 111_B '1' selected
SHUTOFF_POL_0	3	rw	**Channel 0: Configure if the selected shutoff input signal used as shutoff output signal is inverted or not.** ***Note**: This bit field is write-protected by bit CLSi_CDTM_DTMd_CTRL2.WR_EN_0 = 0.* 0_B inverter disabled 1_B inverter enabled
UPD_MODE_0	5:4	rw	**Channel 0: Control the update mode of the internal SHUTOFF_SYNC_0 signal.** ***Note**: The reset of SHUTOFF_SYNC_0 has lower priority than the set.* ***Note**: This bit field is write-protected by bit CLSi_CDTM_DTMd_CTRL2.WR_EN_0 = 0.* 00_B internal signal SHUTOFF_SYNC_0 is cleared; a set is not possible 01_B signal PSU_SHUT_OFF[0:0] sets internal signal SHUTOFF_SYNC_0; internal signal SHUTOFF_SYNC_0 is cleared depending on bit field CLSi_CDTM_DTMd_CTRL2.SHUT_OFF_RST_0 10_B signal PSU_SHUT_OFF[0:0] sets internal signal SHUTOFF_SYNC_0; internal signal SHUTOFF_SYNC_0 is cleared by signal DTM_PREV_IN0_REDGE or DTM_PREV_IN0_FEDGE or DTM_IN_REDGE of channel 0 or DTM_IN_FEDGE of channel 0 defined by bit field CLSi_CDTM_DTMd_CTRL.DTM_SEL 11_B signal PSU_SHUT_OFF[0:0] =1 sets internal signal SHUTOFF_SYNC_0; signal PSU_SHUT_OFF[0:0] =0 clears internal signal SHUTOFF_SYNC_0

(table continues...)

(continued)

Field	Bits	Type	Description
SHUT_OFF_RST_0	6	rw	**Channel 0: Clear of internal signal SHUTOFF_SYNC_0 if selected as control source.** ***Note**: This bit field is write-protected by bit CLSi_CDTM_DTMd_CTRL2.WR_EN_0 = 0.* **If reading:** 0_B No action **If writing:** 0_B No action 1_B shutoff reset (clear of internal signal SHUTOFF_SYNC_0 enabled)
WR_EN_0	7	rw	**Channel 0: Write enable of bit fields** ***Note**: Write enable bit field always read as zero, because no storage element implemented.* 0_B writing of the bit fields CLSi_CDTM_DTMd_CTRL2.SHUTOFF_SEL_0, CLSi_CDTM_DTMd_CTRL2.SHUTOFF_POL_0, CLSi_CDTM_DTMd_CTRL2.UPD_MODE_0, CLSi_CDTM_DTMd_CTRL2.SHUT_OFF_RST_0 disabled 1_B writing of the bit fields CLSi_CDTM_DTMd_CTRL2.SHUTOFF_SEL_0, CLSi_CDTM_DTMd_CTRL2.SHUTOFF_POL_0, CLSi_CDTM_DTMd_CTRL2.UPD_MODE_0, CLSi_CDTM_DTMd_CTRL2.SHUT_OFF_RST_0 enabled
SHUTOFF_SEL_1	10:8	rw	**Channel 1: Select input signal to be used as shutoff signal.** ***Note**: This bit field is write-protected by bit CLSi_CDTM_DTMd_CTRL2.WR_EN_1 = 0.* 000_B TIM_CH_IN0 selected 001_B TIM_CH_IN1 selected 010_B DTM_AUX_IN[2:2] of DTM instance d (DTM_AUX_IN[0:0] of DTM instance d-1) selected 011_B DTM_AUX_IN[3:3] of DTM instance d (DTM_AUX_IN[1:1] of DTM instance d-1) selected 100_B DTM_AUX_IN[0:0] of DTM instance d selected 101_B DTM_AUX_IN[1:1] of DTM instance d selected 110_B '0' selected 111_B '1' selected
SHUTOFF_POL_1	11	rw	**Channel 1: Configure if the selected shutoff input signal used as shutoff output signal is inverted or not.** ***Note**: This bit field is write-protected by bit CLSi_CDTM_DTMd_CTRL2.WR_EN_1 = 0.* 0_B inverter disabled 1_B inverter enabled

(table continues...)

(continued)

Field	Bits	Type	Description
UPD_MODE_1	13:12	rw	**Channel 1: Control the update mode of the internal SHUTOFF_SYNC_1 signal.** ***Note**: The reset of SHUTOFF_SYNC_1 has lower priority than the set.* ***Note**: This bit field is write-protected by bit CLSi_CDTM_DTMd_CTRL2.WR_EN_1 = 0.* 00_B internal signal SHUTOFF_SYNC_1 is cleared; a set is not possible 01_B signal PSU_SHUT_OFF[1:1] sets internal signal SHUTOFF_SYNC_1; internal signal SHUTOFF_SYNC_1 is cleared depending to bit field CLSi_CDTM_DTMd_CTRL2.SHUT_OFF_RST_1 10_B signal PSU_SHUT_OFF[1:1] sets internal signal SHUTOFF_SYNC_1; internal signal SHUTOFF_SYNC_1 is cleared by signal DTM_PREV_IN0_REDGE or DTM_PREV_IN0_FEDGE or DTM_IN_REDGE of channel 0 or DTM_IN_FEDGE of channel 0 defined by bit field CLSi_CDTM_DTMd_CTRL.DTM_SEL 11_B signal PSU_SHUT_OFF[1:1]=1 sets internal signal SHUTOFF_SYNC_1; signal PSU_SHUT_OFF[1:1]=0 clears internal signal SHUTOFF_SYNC_1
SHUT_OFF_RST_1	14	rw	**Channel 1: Clear of internal signal SHUTOFF_SYNC_1 if selected as control source.** ***Note**: This bit field is write-protected by bit CLSi_CDTM_DTMd_CTRL2.WR_EN_1 = 0.* **If reading:** 0_B No action **If writing:** 0_B No action 1_B shutoff reset (clear of internal signal SHUTOFF_SYNC_1 enabled)

(table continues...)

(continued)

Field	Bits	Type	Description
WR_EN_1	15	rw	**Channel 1: Write enable of bit fields** Writing a 1_B enables the writing of the bit fields CLSi_CDTM_DTMd_CTRL2.SHUTOFF_SEL_1, CLSi_CDTM_DTMd_CTRL2.SHUTOFF_POL_1, CLSi_CDTM_DTMd_CTRL2.UPD_MODE_1, CLSi_CDTM_DTMd_CTRL2.SHUT_OFF_RST_1. ***Note**: Write enable bit field always read as zero, because no storage element implemented.* 0_B writing of the bit fields CLSi_CDTM_DTMd_CTRL2.SHUTOFF_SEL_1, CLSi_CDTM_DTMd_CTRL2.SHUTOFF_POL_1, CLSi_CDTM_DTMd_CTRL2.UPD_MODE_1, CLSi_CDTM_DTMd_CTRL2.SHUT_OFF_RST_1 disabled 1_B writing of the bit fields CLSi_CDTM_DTMd_CTRL2.SHUTOFF_SEL_1, CLSi_CDTM_DTMd_CTRL2.SHUTOFF_POL_1, CLSi_CDTM_DTMd_CTRL2.UPD_MODE_1, CLSi_CDTM_DTMd_CTRL2.SHUT_OFF_RST_1 enabled
SHUTOFF_SEL_2	18:16	rw	**Channel 2: Select input signal to be used as shutoff signal.** ***Note**: This bit field is write-protected by bit CLSi_CDTM_DTMd_CTRL2.WR_EN_2 = 0.* 000_B TIM_CH_IN0 selected 001_B TIM_CH_IN1 selected 010_B DTM_AUX_IN[2:2] of DTM instance d (DTM_AUX_IN[0:0] of DTM instance d-1) selected 011_B DTM_AUX_IN[3:3] of DTM instance d (DTM_AUX_IN[1:1] of DTM instance d-1) selected 100_B DTM_AUX_IN[0:0] of DTM instance d selected 101_B DTM_AUX_IN[1:1] of DTM instance d selected 110_B '0' selected 111_B '1' selected
SHUTOFF_POL_2	19	rw	**Channel 2: Configure if the selected shutoff input signal used as shutoff output signal is inverted or not.** ***Note**: This bit field is write-protected by bit CLSi_CDTM_DTMd_CTRL2.WR_EN_2 = 0.* 0_B inverter disabled 1_B inverter enabled

(table continues...)

(continued)

Field	Bits	Type	Description
UPD_MODE_2	21:20	rw	**Channel 2: Control the update mode of the internal SHUTOFF_SYNC_2 signal.** ***Note**: The reset of SHUTOFF_SYNC_2 has lower priority than the set.* ***Note**: This bit field is write-protected by bit CLSi_CDTM_DTMd_CTRL2.WR_EN_2 = 0.* 00_B internal signal SHUTOFF_SYNC_2 is cleared; a set is not possible 01_B signal PSU_SHUT_OFF[2:2] sets internal signal SHUTOFF_SYNC_2; internal signal SHUTOFF_SYNC_2 is cleared depending to bit field CLSi_CDTM_DTMd_CTRL2.SHUT_OFF_RST_2 10_B signal PSU_SHUT_OFF[2:2] sets internal signal SHUTOFF_SYNC_2; internal signal SHUTOFF_SYNC_2 is cleared by signal DTM_PREV_IN0_REDGE or DTM_PREV_IN0_FEDGE or DTM_IN_REDGE of channel 0 or DTM_IN_FEDGE of channel 0 defined by bit field CLSi_CDTM_DTMd_CTRL.DTM_SEL 11_B signal PSU_SHUT_OFF[2:2]=1 sets internal signal SHUTOFF_SYNC_2; signal PSU_SHUT_OFF[2:2]=0 clears internal signal SHUTOFF_SYNC_2
SHUT_OFF_RST_2	22	rw	**Channel 2: Clear of internal signal SHUTOFF_SYNC_2 if selected as control source.** ***Note**: This bit field is write-protected by bit CLSi_CDTM_DTMd_CTRL2.WR_EN_2 = 0.* **If reading:** 0_B No action **If writing:** 0_B No action 1_B shutoff reset (clear of internal signal SHUTOFF_SYNC_2 enabled)

(table continues...)

(continued)

Field	Bits	Type	Description
WR_EN_2	23	rw	**Channel 2: Write enable of bit fields** Writing a 1_B enables the writing of the bit fields CLSi_CDTM_DTMd_CTRL2.SHUTOFF_SEL_2, CLSi_CDTM_DTMd_CTRL2.SHUTOFF_POL_2, CLSi_CDTM_DTMd_CTRL2.UPD_MODE_2, CLSi_CDTM_DTMd_CTRL2.SHUT_OFF_RST_2. ***Note**: Write enable bit field always read as zero, because no storage element implemented.* 0_B writing of the bit fields CLSi_CDTM_DTMd_CTRL2.SHUTOFF_SEL_2, CLSi_CDTM_DTMd_CTRL2.SHUTOFF_POL_2, CLSi_CDTM_DTMd_CTRL2.UPD_MODE_2, CLSi_CDTM_DTMd_CTRL2.SHUT_OFF_RST_2 disabled 1_B writing of the bit fields CLSi_CDTM_DTMd_CTRL2.SHUTOFF_SEL_2, CLSi_CDTM_DTMd_CTRL2.SHUTOFF_POL_2, CLSi_CDTM_DTMd_CTRL2.UPD_MODE_2, CLSi_CDTM_DTMd_CTRL2.SHUT_OFF_RST_2 enabled
SHUTOFF_SEL_3	26:24	rw	**Channel 3: Select input signal to be used as shutoff signal.** ***Note**: This bit field is write-protected by bit CLSi_CDTM_DTMd_CTRL2.WR_EN_3 = 0.* 000_B TIM_CH_IN0 selected 001_B TIM_CH_IN1 selected 010_B DTM_AUX_IN[2:2] of DTM instance d (DTM_AUX_IN[0:0] of DTM instance d-1) selected 011_B DTM_AUX_IN[3:3] of DTM instance d (DTM_AUX_IN[1:1] of DTM instance d-1) selected 100_B DTM_AUX_IN[0:0] of DTM instance d selected 101_B DTM_AUX_IN[1:1] of DTM instance d selected 110_B '0' selected 111_B '1' selected
SHUTOFF_POL_3	27	rw	**Channel 3: Configure if the selected shutoff input signal used as shutoff output signal is inverted or not.** ***Note**: This bit field is write-protected by bit CLSi_CDTM_DTMd_CTRL2.WR_EN_3 = 0.* 0_B inverter disabled 1_B inverter enabled

(table continues...)

(continued)

Field	Bits	Type	Description
UPD_MODE_3	29:28	rw	**Channel 3: Control the update mode of the internal SHUTOFF_SYNC_3 signal.** ***Note**: The reset of SHUTOFF_SYNC_3 has lower priority than the set.* ***Note**: This bit field is write-protected by bit CLSi_CDTM_DTMd_CTRL2.WR_EN_3 = 0.* 00_B internal signal SHUTOFF_SYNC_3 is cleared; a set is not possible 01_B signal PSU_SHUT_OFF[3:3] sets internal signal SHUTOFF_SYNC_3; internal signal SHUTOFF_SYNC_3 is cleared depending to bit field CLSi_CDTM_DTMd_CTRL2.SHUT_OFF_RST_3 10_B signal PSU_SHUT_OFF[3:3] sets internal signal SHUTOFF_SYNC_3; internal signal SHUTOFF_SYNC_3 is cleared by signal DTM_PREV_IN0_REDGE or DTM_PREV_IN0_FEDGE or DTM_IN_REDGE of channel 0 or DTM_IN_FEDGE of channel 0 defined by bit field CLSi_CDTM_DTMd_CTRL.DTM_SEL 11_B signal PSU_SHUT_OFF[3:3] =1 sets internal signal SHUTOFF_SYNC_3; signal PSU_SHUT_OFF[3:3] =0 clears internal signal SHUTOFF_SYNC_3
SHUT_OFF_RST_3	30	rw	**Channel 3: Clear of internal signal SHUTOFF_SYNC_3 if selected as control source.** ***Note**: This bit field is write-protected by bit CLSi_CDTM_DTMd_CTRL2.WR_EN_3 = 0.* **If reading:** 0_B No action **If writing:** 0_B No action 1_B shutoff reset (clear of internal signal SHUTOFF_SYNC_3 enabled)

(table continues...)

(continued)

Field	Bits	Type	Description
WR_EN_3	31	rw	**Channel 3: Write enable of bit fields** Writing a 1_B enables the writing of the bit fields CLSi_CDTM_DTMd_CTRL2.SHUTOFF_SEL_3, CLSi_CDTM_DTMd_CTRL2.SHUTOFF_POL_3, CLSi_CDTM_DTMd_CTRL2.UPD_MODE_3, CLSi_CDTM_DTMd_CTRL2.SHUT_OFF_RST_3. ***Note**: Write enable bit field always read as zero, because no storage element implemented.* 0_B writing of the bit fields CLSi_CDTM_DTMd_CTRL2.SHUTOFF_SEL_3, CLSi_CDTM_DTMd_CTRL2.SHUTOFF_POL_3, CLSi_CDTM_DTMd_CTRL2.UPD_MODE_3, CLSi_CDTM_DTMd_CTRL2.SHUT_OFF_RST_3 disabled 1_B writing of the bit fields CLSi_CDTM_DTMd_CTRL2.SHUTOFF_SEL_3, CLSi_CDTM_DTMd_CTRL2.SHUTOFF_POL_3, CLSi_CDTM_DTMd_CTRL2.UPD_MODE_3, CLSi_CDTM_DTMd_CTRL2.SHUT_OFF_RST_3 enabled

44.5.144 CDTM[i]_DTM[d] channel [x] dead-time shadow values (x=0-3;i=0-2;d=0-5)

CLSi_CDTM_DTMd_CH_DTV_SRx (d=0-5;i=0-2;x=0-3) Offset address: 004430_H+i*20000_H+d*40_H+x*4

CDTM[i]_DTM[d] channel [x] dead-time shadow values Kernel Reset value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RELFALL_UPD_EN	RELFALL_UPD_FE0RE1	0	RELFALL_SR												
rw	rw	r	rw												

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RELRISE_UPD_EN	RELRISE_UPD_FE0RE1	0	RELRISE_SR												
rw	rw	r	rw												

Field	Bits	Type	Description
RELRISE_SR	12:0	rw	**Shadow value for rising edge dead-time**

(table continues...)

(continued)

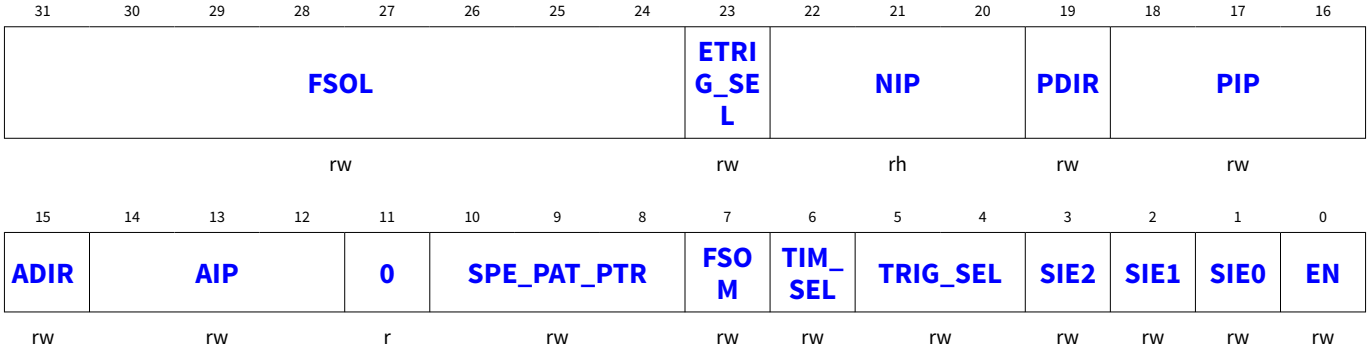
Field	Bits	Type	Description
RELRISE_UPD_FE0RE1	14	rw	**Control if falling edge or rising edge triggers update of** CLSi_CDTM_DTMd_CH_DTVx**.**RELRISE 0_B update of CLSi_CDTM_DTMd_CH_DTVx.RELRISE triggered by falling edge 1_B update of CLSi_CDTM_DTMd_CH_DTVx.RELRISE triggered by rising edge
RELRISE_UPD_EN	15	rw	**Control bit to enable update of** CLSi_CDTM_DTMd_CH_DTVx**.**RELRISE 0_B update mechanism of CLSi_CDTM_DTMd_CH_DTVx.RELRISE disabled 1_B update mechanism of CLSi_CDTM_DTMd_CH_DTVx.RELRISE enabled
RELFALL_SR	28:16	rw	**Shadow value for falling edge dead-time**
RELFALL_UPD_FE0RE1	30	rw	**Control if falling edge or rising edge triggers update of** CLSi_CDTM_DTMd_CH_DTVx**.**RELFALL 0_B update of CLSi_CDTM_DTMd_CH_DTVx.RELFALL triggered by falling edge 1_B update of CLSi_CDTM_DTMd_CH_DTVx.RELFALL triggered by rising edge
RELFALL_UPD_EN	31	rw	**Control bit to enable update of** CLSi_CDTM_DTMd_CH_DTVx**.**RELFALL 0_B update mechanism of CLSi_CDTM_DTMd_CH_DTVx.RELFALL disabled 1_B update mechanism of CLSi_CDTM_DTMd_CH_DTVx.RELFALL enabled
0	13, 29	r	**Reserved** Read as 0; must be written with 0.

44.5.145 SPE[i] control status register (i=0)

***Note**:* CLSi_SPE_CTRL_STAT *holds the current and the previous states of the hall pattern and can be written by the CPU for initialization purposes*
*after reset before enabling **SPE**. Any CPU write to* CLSi_SPE_CTRL_STAT
*has precedence over internal updates of this register and will also disable the generation of all **SPE** interrupts during this access.*

CLSi_SPE_CTRL_STAT (i=0) Offset address: $004C00_H$+i*20000_H

SPE[i] control status register Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
EN	0	rw	**SPE sub-module enable** 0_B SPE disabled 1_B SPE enabled
SIEk (k=0-2)	k+1	rw	**SPE Input [k] enable for** TIM[i]_CH***[x:x]*****,** TIM[i]_CH***[y:y]*****,** TIM[i]_CH***[z:z]*****.** ***Note**:* *When the input is disabled, a 0 signal is sampled for this input.* *However, the bit field* CLSi_SPE_CTRL_STAT**.**NIP *shows the true value of the input signal.* 0_B SPE Input is disabled 1_B SPE Input is enabled
TRIG_SEL	5:4	rw	**Select trigger input signal** ***Note**:* *In case* CLSi_SPE_CTRL_STAT**.**ETRIG_SEL *= 1, according to the selected TOM_CH[x]_TRIG_CCU1 signal, the configuration bits* CLSi_TOM_CHx_CTRL**.**SPE_TRIG *and* CLSi_TOM_CHx_CTRL**.**OSM *have to be set in the same* ***TOM*** *channel x to enable the trigger signal generation in one-shot mode.* **If** CLSi_SPE_CTRL_STAT **.ETRIG_SEL == 0:** 00_B SPE[i]_NIPD selected 01_B TOM[i]_CH[x]_TRIG_CCU0 selected 10_B TOM[i]_CH[x]_TRIG_CCU1 selected 11_B TOM[i]_CH[x]_TRIG_CCU1 selected **If** CLSi_SPE_CTRL_STAT **.ETRIG_SEL == 1:** 00_B TOM[i]_CH[x]_TRIG_CCU1 selected 01_B TOM[i]_CH[x]_TRIG_CCU1 selected 10_B TOM[i]_CH[x]_TRIG_CCU1 selected 11_B TOM[i]_CH[x]_TRIG_CCU1 selected
TIM_SEL	6	rw	**Select TIM input signal** 0_B TIM[i]_CH[2:0] (Cluster 0) 1_B TIM[i]_CH[2:0] (Cluster 1)

(table continues...)

(continued)

Field	Bits	Type	Description
FSOM	7	rw	**Fast Shutoff Mode** 0_B Fast Shutoff mode disabled 1_B Fast Shutoff mode enabled
SPE_PAT_PTR	10:8	rw	**Pattern selector for TOM output signals.** Index into the CLSi_SPE_OUT_PATp register table. Each register CLSi_SPE_OUT_PATp is fixed assigned to one bit field CLSi_SPE_PAT.IPt_PAT. Thus, the pointer CLSi_SPE_CTRL_STAT.SPE_PAT_PTR represents an index to the selected CLSi_SPE_OUT_PATp register as well as the actual detected input pattern CLSi_SPE_PAT.IPt_PAT. 000_B CLSi_SPE_OUT_PATp (p=0) selected 001_B CLSi_SPE_OUT_PATp (p=1) selected 010_B CLSi_SPE_OUT_PATp (p=2) selected 011_B CLSi_SPE_OUT_PATp (p=3) selected 100_B CLSi_SPE_OUT_PATp (p=4) selected 101_B CLSi_SPE_OUT_PATp (p=5) selected 110_B CLSi_SPE_OUT_PATp (p=6) selected 111_B CLSi_SPE_OUT_PATp (p=7) selected
AIP	14:12	rw	**Input pattern that was detected by a regular input pattern change.**
ADIR	15	rw	**Rotation direction.** 0_B Rotation direction is 0 according to CLSi_SPE_PAT register 1_B Rotation direction is 1 according to CLSi_SPE_PAT register
PIP	18:16	rw	**Previous input pattern that was detected by a regular input pattern change.**
PDIR	19	rw	**Previous rotation direction** 0_B Rotation direction is 0 according to CLSi_SPE_PAT register 1_B Rotation direction is 1 according to CLSi_SPE_PAT register
NIP	22:20	rh	**New input pattern that was detected.** ***Note**:* *This bit field mirrors the new input pattern. **SPE** internal functionality is triggered on each change of this bit field.*
ETRIG_SEL	23	rw	**Extended trigger selection of signal** TRIG_SEL 0_B Selection of TRIG_SEL signal (For explanation see note at CLSi_SPE_CTRL_STAT.TRIG_SEL) 1_B Selection of TRIG_SEL signal (For explanation see note at CLSi_SPE_CTRL_STAT.TRIG_SEL)
FSOL	31:24	rw	**Fast Shutoff Level for TOM[i] channel 0 to 7**
0	11	r	**Reserved** Read as 0; must be written with 0.

44.5.146 SPE[i] control status register (i=1)

For a description of this register, see SPE[i] control status register (i=0).

CLSi_SPE_CTRL_STAT (i=1) Offset address: $004C00_H$+i*20000_H

SPE[i] control status register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FSOL								ETRIG_SEL	NIP			PDIR	PIP		
rw								rw	rh			rw	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADIR	AIP			0	SPE_PAT_PTR			FSOM	TIM_SEL	TRIG_SEL		SIE2	SIE1	SIE0	EN
rw	rw			r	rw			rw	rw	rw		rw	rw	rw	rw

Field	Bits	Type	Description
EN	0	rw	**SPE sub-module enable** 0_B SPE disabled 1_B SPE enabled
SIEk (k=0-2)	k+1	rw	**SPE Input [k] enable for** TIM[i]_CH***[x:x]*****,** TIM[i]_CH***[y:y]*****,** TIM[i]_CH***[z:z]*****.** ***Note**:* *When the input is disabled, a 0 signal is sampled for this input.* *However, the bit field* CLSi_SPE_CTRL_STAT**.**NIP *shows the true value of the input signal.* 0_B SPE Input is disabled 1_B SPE Input is enabled
TRIG_SEL	5:4	rw	**Select trigger input signal** ***Note**:* *In case* CLSi_SPE_CTRL_STAT**.**ETRIG_SEL *= 1, according to the selected TOM_CH[x]_TRIG_CCU1 signal, the configuration bits* CLSi_TOM_CHx_CTRL**.**SPE_TRIG *and* CLSi_TOM_CHx_CTRL**.**OSM *have to be set in the same* ***TOM*** *channel x to enable the trigger signal generation in one-shot mode.* **If** CLSi_SPE_CTRL_STAT **.ETRIG_SEL == 0:** 00_B SPE[i]_NIPD selected 01_B TOM[i]_CH[x]_TRIG_CCU0 selected 10_B TOM[i]_CH[x]_TRIG_CCU1 selected 11_B TOM[i]_CH[x]_TRIG_CCU1 selected **If** CLSi_SPE_CTRL_STAT **.ETRIG_SEL == 1:** 00_B TOM[i]_CH[x]_TRIG_CCU1 selected 01_B TOM[i]_CH[x]_TRIG_CCU1 selected 10_B TOM[i]_CH[x]_TRIG_CCU1 selected 11_B TOM[i]_CH[x]_TRIG_CCU1 selected

(table continues...)

(continued)

Field	Bits	Type	Description
TIM_SEL	6	rw	**Select TIM input signal** 0_B TIM[i]_CH[5:3] (Cluster 0) 1_B TIM[i]_CH[5:3] (Cluster 1)
FSOM	7	rw	**Fast Shutoff Mode** 0_B Fast Shutoff mode disabled 1_B Fast Shutoff mode enabled
SPE_PAT_PTR	10:8	rw	**Pattern selector for TOM output signals.** Index into the CLSi_SPE_OUT_PATp register table. Each register CLSi_SPE_OUT_PATp is fixed assigned to one bit field CLSi_SPE_PAT.IPt_PAT. Thus, the pointer CLSi_SPE_CTRL_STAT.SPE_PAT_PTR represents an index to the selected CLSi_SPE_OUT_PATp register as well as the actual detected input pattern CLSi_SPE_PAT.IPt_PAT. 000_B CLSi_SPE_OUT_PATp (p=0) selected 001_B CLSi_SPE_OUT_PATp (p=1) selected 010_B CLSi_SPE_OUT_PATp (p=2) selected 011_B CLSi_SPE_OUT_PATp (p=3) selected 100_B CLSi_SPE_OUT_PATp (p=4) selected 101_B CLSi_SPE_OUT_PATp (p=5) selected 110_B CLSi_SPE_OUT_PATp (p=6) selected 111_B CLSi_SPE_OUT_PATp (p=7) selected
AIP	14:12	rw	**Input pattern that was detected by a regular input pattern change.**
ADIR	15	rw	**Rotation direction.** 0_B Rotation direction is 0 according to CLSi_SPE_PAT register 1_B Rotation direction is 1 according to CLSi_SPE_PAT register
PIP	18:16	rw	**Previous input pattern that was detected by a regular input pattern change.**
PDIR	19	rw	**Previous rotation direction** 0_B Rotation direction is 0 according to CLSi_SPE_PAT register 1_B Rotation direction is 1 according to CLSi_SPE_PAT register
NIP	22:20	rh	**New input pattern that was detected.** ***Note**: This bit field mirrors the new input pattern. **SPE** internal functionality is triggered on each change of this bit field.*
ETRIG_SEL	23	rw	**Extended trigger selection of signal** TRIG_SEL 0_B Selection of TRIG_SEL signal (For explanation see note at CLSi_SPE_CTRL_STAT.TRIG_SEL) 1_B Selection of TRIG_SEL signal (For explanation see note at CLSi_SPE_CTRL_STAT.TRIG_SEL)
FSOL	31:24	rw	**Fast Shutoff Level for TOM[i] channel 0 to 7**
0	11	r	**Reserved** Read as 0; must be written with 0.

44.5.147 SPE[i] control status register (i=2)

For a description of this register, see SPE[i] control status register (i=0).

CLSi_SPE_CTRL_STAT (i=2) Offset address: $004C00_H$+i*20000_H

SPE[i] control status register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FSOL								ETRIG_SEL	NIP			PDIR	PIP		
rw								rw	rh			rw	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADIR	AIP			0	SPE_PAT_PTR			FSOM	TIM_SEL	TRIG_SEL		SIE2	SIE1	SIE0	EN
rw	rw			r	rw			rw	rw	rw		rw	rw	rw	rw

Field	Bits	Type	Description
EN	0	rw	**SPE sub-module enable** 0_B SPE disabled 1_B SPE enabled
SIEk (k=0-2)	k+1	rw	**SPE Input [k] enable for** TIM[i]_CH***[x:x]***, TIM[i]_CH***[y:y]***, TIM[i]_CH***[z:z]*****.** ***Note**: When the input is disabled, a 0 signal is sampled for this input.* *However, the bit field CLSi_SPE_CTRL_STAT.NIP shows the true value of the input signal.* 0_B SPE Input is disabled 1_B SPE Input is enabled
TRIG_SEL	5:4	rw	**Select trigger input signal** ***Note**: In case CLSi_SPE_CTRL_STAT.ETRIG_SEL = 1, according to the selected TOM_CH[x]_TRIG_CCU1 signal, the configuration bits CLSi_TOM_CHx_CTRL.SPE_TRIG and CLSi_TOM_CHx_CTRL.OSM have to be set in the same **TOM** channel x to enable the trigger signal generation in one-shot mode.* **If CLSi_SPE_CTRL_STAT .ETRIG_SEL == 0:** 00_B SPE[i]_NIPD selected 01_B TOM[i]_CH[x]_TRIG_CCU0 selected 10_B TOM[i]_CH[x]_TRIG_CCU1 selected 11_B TOM[i]_CH[x]_TRIG_CCU1 selected **If CLSi_SPE_CTRL_STAT .ETRIG_SEL == 1:** 00_B TOM[i]_CH[x]_TRIG_CCU1 selected 01_B TOM[i]_CH[x]_TRIG_CCU1 selected 10_B TOM[i]_CH[x]_TRIG_CCU1 selected 11_B TOM[i]_CH[x]_TRIG_CCU1 selected

(table continues...)

(continued)

Field	Bits	Type	Description
TIM_SEL	6	rw	**Select TIM input signal** 0_B TIM[i]_CH[2:0] (Cluster 2) 1_B TIM[i]_CH[2:0] (Cluster 3)
FSOM	7	rw	**Fast Shutoff Mode** 0_B Fast Shutoff mode disabled 1_B Fast Shutoff mode enabled
SPE_PAT_PTR	10:8	rw	**Pattern selector for TOM output signals.** Index into the CLSi_SPE_OUT_PATp register table. Each register CLSi_SPE_OUT_PATp is fixed assigned to one bit field CLSi_SPE_PAT.IPt_PAT. Thus, the pointer CLSi_SPE_CTRL_STAT.SPE_PAT_PTR represents an index to the selected CLSi_SPE_OUT_PATp register as well as the actual detected input pattern CLSi_SPE_PAT.IPt_PAT. 000_B CLSi_SPE_OUT_PATp (p=0) selected 001_B CLSi_SPE_OUT_PATp (p=1) selected 010_B CLSi_SPE_OUT_PATp (p=2) selected 011_B CLSi_SPE_OUT_PATp (p=3) selected 100_B CLSi_SPE_OUT_PATp (p=4) selected 101_B CLSi_SPE_OUT_PATp (p=5) selected 110_B CLSi_SPE_OUT_PATp (p=6) selected 111_B CLSi_SPE_OUT_PATp (p=7) selected
AIP	14:12	rw	**Input pattern that was detected by a regular input pattern change.**
ADIR	15	rw	**Rotation direction.** 0_B Rotation direction is 0 according to CLSi_SPE_PAT register 1_B Rotation direction is 1 according to CLSi_SPE_PAT register
PIP	18:16	rw	**Previous input pattern that was detected by a regular input pattern change.**
PDIR	19	rw	**Previous rotation direction** 0_B Rotation direction is 0 according to CLSi_SPE_PAT register 1_B Rotation direction is 1 according to CLSi_SPE_PAT register
NIP	22:20	rh	**New input pattern that was detected.** ***Note**: This bit field mirrors the new input pattern. **SPE** internal functionality is triggered on each change of this bit field.*
ETRIG_SEL	23	rw	**Extended trigger selection of signal** TRIG_SEL 0_B Selection of TRIG_SEL signal (For explanation see note at CLSi_SPE_CTRL_STAT.TRIG_SEL) 1_B Selection of TRIG_SEL signal (For explanation see note at CLSi_SPE_CTRL_STAT.TRIG_SEL)
FSOL	31:24	rw	**Fast Shutoff Level for TOM[i] channel 0 to 7**
0	11	r	**Reserved** Read as 0; must be written with 0.

44.5.148 SPE[i] input pattern definition register (i=0-2)

***Note**: Only the first block of valid input patterns defines the commutation. All input pattern following the first marked invalid input pattern are ignored.*

CLSi_SPE_PAT (i=0-2) Offset address: $004C04_H+i*20000_H$

SPE[i] input pattern definition register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
IP7_PAT			IP7_VAL	IP6_PAT			IP6_VAL	IP5_PAT			IP5_VAL	IP4_PAT			IP4_VAL
rw			rw	rw			rw	rw			rw	rw			rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IP3_PAT			IP3_VAL	IP2_PAT			IP2_VAL	IP1_PAT			IP1_VAL	IP0_PAT			IP0_VAL
rw			rw	rw			rw	rw			rw	rw			rw

Field	Bits	Type	Description
IPt_VAL (t=0-7)	4*t	rw	**Input pattern [t] is a valid pattern.** 0_B Pattern invalid 1_B Pattern valid
IPt_PAT (t=0-7)	4*t+3:4*t+1	rw	**Input pattern [t]** Bit field defines the first input pattern of the **SPE** input signals. Bit 1 defines the TIM[i]_CH[x:x] input signal. Bit 2 defines the TIM[i]_CH[y:y] input signal. Bit 3 defines the TIM[i]_CH[z:z] input signal.

44.5.149 SPE[i] output definition register [p] (p=0-7;i=0-2)

***Note**: Register bit field CLSi_SPE_OUT_PATp.SPE_OUT_PATo defines the output selection for **TOM** instance i, channel 0 to **TOM** instance i, channel 7 depending on the input pattern CLSi_SPE_PAT.IPt_PAT.*

CLSi_SPE_OUT_PATp (i=0-2;p=0-7) Offset address: $004C08_H+i*20000_H+p*4$

SPE[i] output definition register [p] Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SPE_OUT_PAT7		SPE_OUT_PAT6		SPE_OUT_PAT5		SPE_OUT_PAT4		SPE_OUT_PAT3		SPE_OUT_PAT2		SPE_OUT_PAT1		SPE_OUT_PAT0	
rw		rw		rw		rw		rw		rw		rw		rw	

Field	Bits	Type	Description
SPE_OUT_PATo (o=0-7)	2*o+1:2*o	rw	**SPE output control value for TOM channel [o]** CLSi_SPE_OUT_PATp.SPE_OUT_PATo defines the output select signal of **TOM** instance i, channel n. 00_B Set SPE[i]_OUT[o:o] to TOM[i]_CH[x]_SOUR 01_B Set SPE[i]_OUT[o:o] to TOM[i]_CH[x]_SOUR 10_B Set SPE[i]_OUT[o:o] to 0 11_B Set SPE[i]_OUT[o:o] to 1
0	31:16	r	**Reserved** Read as 0; must be written with 0.

44.5.150 SPE[i] output control register (i=0-2)

CLSi_SPE_OUT_CTRL (i=0-2) Offset address: $004C28_H+i*20000_H$

SPE[i] output control register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

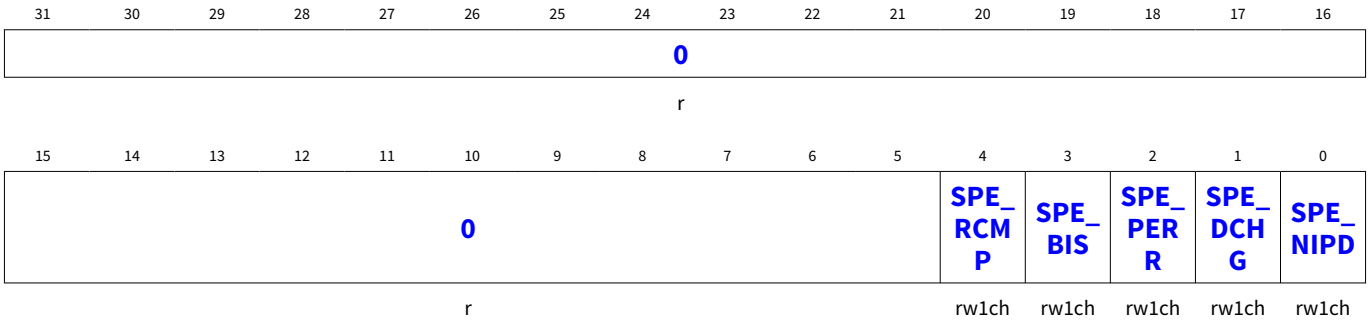
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SPE_OUT_CTRL7		SPE_OUT_CTRL6		SPE_OUT_CTRL5		SPE_OUT_CTRL4		SPE_OUT_CTRL3		SPE_OUT_CTRL2		SPE_OUT_CTRL1		SPE_OUT_CTRL0	
rw		rw		rw		rw		rw		rw		rw		rw	

Field	Bits	Type	Description
SPE_OUT_CTRLo (o=0-7)	2*o+1:2*o	rw	**SPE output control value for TOM channel [o]** CLSi_SPE_OUT_CTRL.SPE_OUT_CTRLo defines the output select signal of **TOM** instance i, channel n. ***Note:*** *Current output control selection for SPE[i]_OUT[7:0].* 00_B Set SPE[i]_OUT[o:o] to TOM[i]_CH[x]_SOUR 01_B Set SPE[i]_OUT[o:o] to TOM[i]_CH[x]_SOUR 10_B Set SPE[i]_OUT[o:o] to 0 11_B Set SPE[i]_OUT[o:o] to 1
0	31:16	r	**Reserved** Read as 0; must be written with 0.

44.5.151 SPE[i] interrupt notification register (i=0-2)

CLSi_SPE_IRQ_NOTIFY (i=0-2) Offset address: $004C2C_H+i*20000_H$

SPE[i] interrupt notification register Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
SPE_NIPD	0	rw1ch	**New input pattern interrupt occurred** ***Note**: This bit will be cleared on a CPU write-access of value 1. A read-access leaves the bit unchanged.* **If reading:** 0_B No interrupt raised 1_B Interrupt was raised **If writing:** 0_B No action 1_B Clear interrupt
SPE_DCHG	1	rw1ch	**SPE_DIR bit changed on behalf of new input pattern.** **If reading:** 0_B No interrupt raised 1_B Interrupt was raised **If writing:** 0_B No action 1_B Clear interrupt
SPE_PERR	2	rw1ch	**Wrong or invalid pattern detected at input.** **If reading:** 0_B No interrupt raised 1_B Interrupt was raised **If writing:** 0_B No action 1_B Clear interrupt
SPE_BIS	3	rw1ch	**Bouncing input signal detected** **If reading:** 0_B No interrupt raised 1_B Interrupt was raised **If writing:** 0_B No action 1_B Clear interrupt

(table continues...)

(continued)

Field	Bits	Type	Description
SPE_RCMP	4	rw1ch	**SPE revolution counter match event** **If reading:** 0_B No interrupt raised 1_B Interrupt was raised **If writing:** 0_B No action 1_B Clear interrupt
0	31:5	r	**Reserved** Read as 0; must be written with 0.

44.5.152 SPE[i] interrupt enable register (i=0-2)

CLSi_SPE_IRQ_EN (i=0-2) Offset address: $004C30_H+i*20000_H$

SPE[i] interrupt enable register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0											SPE_RCMP_IRQ_EN	SPE_BIS_IRQ_EN	SPE_PERR_IRQ_EN	SPE_DCHG_IRQ_EN	SPE_NIPD_IRQ_EN
r											rw	rw	rw	rw	rw

Field	Bits	Type	Description
SPE_NIPD_IRQ_EN	0	rw	**SPE_NIPD_IRQ interrupt enable** 0_B Disable interrupt, interrupt is not visible outside eGTM-IP 1_B Enable interrupt, interrupt is visible outside eGTM-IP
SPE_DCHG_IRQ_EN	1	rw	**SPE_DCHG_IRQ interrupt enable** 0_B Disable interrupt, interrupt is not visible outside eGTM-IP 1_B Enable interrupt, interrupt is visible outside eGTM-IP
SPE_PERR_IRQ_EN	2	rw	**SPE_PERR_IRQ interrupt enable** 0_B Disable interrupt, interrupt is not visible outside eGTM-IP 1_B Enable interrupt, interrupt is visible outside eGTM-IP
SPE_BIS_IRQ_EN	3	rw	**SPE_BIS_IRQ interrupt enable** 0_B Disable interrupt, interrupt is not visible outside eGTM-IP 1_B Enable interrupt, interrupt is visible outside eGTM-IP

(table continues...)

(continued)

Field	Bits	Type	Description
SPE_RCMP_IRQ_EN	4	rw	**SPE_RCMP_IRQ interrupt enable** 0_B Disable interrupt, interrupt is not visible outside eGTM-IP 1_B Enable interrupt, interrupt is visible outside eGTM-IP
0	31:5	r	**Reserved** Read as 0; must be written with 0.

44.5.153 SPE[i] Interrupt Generation By Software (i=0-2)

CLSi_SPE_IRQ_FORCINT (i=0-2) Offset address: $004C34_H+i*20000_H$

SPE[i] Interrupt Generation By Software Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0											TRG_SPE_RCMP	TRG_SPE_BIS	TRG_SPE_PERR	TRG_SPE_DCHG	TRG_SPE_NIPD
r											rw	rw	rw	rw	rw

Field	Bits	Type	Description
TRG_SPE_NIPD	0	rw	**Trigger** CLSi_SPE_IRQ_NOTIFY.SPE_NIPD **by software** ***Note:*** *This bit is cleared automatically after interrupt is released* ***Note:*** *This bit is write-protected by bit* CLSi_ARCH_CTRL.RF_PROT **If reading:** 0_B No status **If writing:** 0_B No action 1_B Force setting of CLSi_SPE_IRQ_NOTIFY.SPE_NIPD
TRG_SPE_DCHG	1	rw	**Trigger** CLSi_SPE_IRQ_NOTIFY.SPE_DCHG **by software.** ***Note:*** *This bit is cleared automatically after interrupt is released* ***Note:*** *This bit is write-protected by bit* CLSi_ARCH_CTRL.RF_PROT **If reading:** 0_B No status **If writing:** 0_B No action 1_B Force setting of CLSi_SPE_IRQ_NOTIFY.SPE_DCHG

(table continues...)

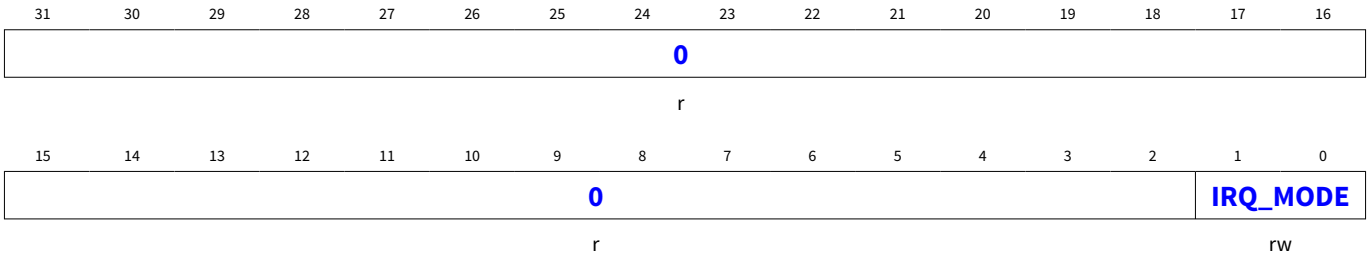
(continued)

Field	Bits	Type	Description
TRG_SPE_PERR	2	rw	**Trigger** CLSi_SPE_IRQ_NOTIFY.SPE_PERR **by software.** ***Note**: This bit is cleared automatically after interrupt is released* ***Note**: This bit is write-protected by bit* CLSi_ARCH_CTRL.RF_PROT **If reading:** 0_B No status **If writing:** 0_B No action 1_B Force setting of CLSi_SPE_IRQ_NOTIFY.SPE_PERR
TRG_SPE_BIS	3	rw	**Trigger** CLSi_SPE_IRQ_NOTIFY.SPE_BIS **by software.** ***Note**: This bit is cleared automatically after interrupt is released* ***Note**: This bit is write-protected by bit* CLSi_ARCH_CTRL.RF_PROT **If reading:** 0_B No status **If writing:** 0_B No action 1_B Force setting of CLSi_SPE_IRQ_NOTIFY.SPE_BIS
TRG_SPE_RCMP	4	rw	**Trigger** CLSi_SPE_IRQ_NOTIFY.SPE_RCMP **by software.** ***Note**: This bit is cleared automatically after interrupt is released* ***Note**: This bit is write-protected by bit* CLSi_ARCH_CTRL.RF_PROT **If reading:** 0_B No status **If writing:** 0_B No action 1_B Force setting of CLSi_SPE_IRQ_NOTIFY.SPE_RCMP
0	31:5	r	**Reserved** Read as 0; must be written with 0.

44.5.154 SPE[i] interrupt mode configuration register (i=0-2)

CLSi_SPE_IRQ_MODE (i=0-2) Offset address: $004C38_H+i*20000_H$

SPE[i] interrupt mode configuration register Kernel Reset value: $0000\ 0002_H$

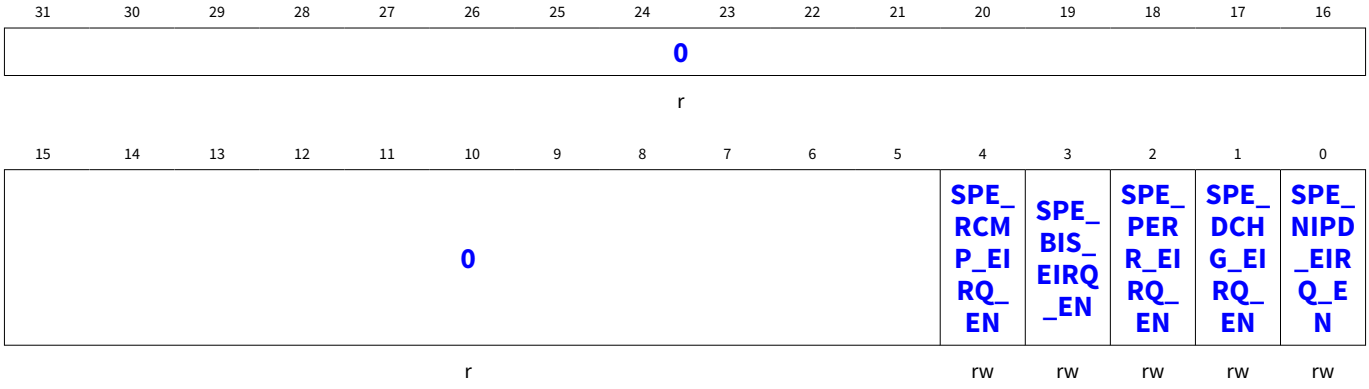


Field	Bits	Type	Description
IRQ_MODE	1:0	rw	**IRQ mode selection** ***Note**:* *The interrupt modes are described in* **eGTM** interrupt concept. 00_B Level mode 01_B Pulse mode 10_B Pulse-Notify mode 11_B Single-Pulse mode
0	31:2	r	**Reserved** Read as 0; must be written with 0.

44.5.155 SPE[i] error interrupt enable register (i=0-2)

CLSi_SPE_EIRQ_EN (i=0-2) Offset address: $004C3C_H+i*20000_H$

SPE[i] error interrupt enable register Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
SPE_NIPD_EIRQ_EN	0	rw	**SPE_NIPD_EIRQ interrupt enable** 0_B Disable error interrupt, error interrupt is not visible outside eGTM-IP 1_B Enable error interrupt, error interrupt is visible outside eGTM-IP
SPE_DCHG_EIRQ_EN	1	rw	**SPE_DCHG_EIRQ error interrupt enable** 0_B Disable error interrupt, error interrupt is not visible outside eGTM-IP 1_B Enable error interrupt, error interrupt is visible outside eGTM-IP

(table continues...)

(continued)

Field	Bits	Type	Description
SPE_PERR_EIRQ_EN	2	rw	**SPE_PERR_EIRQ error interrupt enable** 0_B Disable error interrupt, error interrupt is not visible outside eGTM-IP 1_B Enable error interrupt, error interrupt is visible outside eGTM-IP
SPE_BIS_EIRQ_EN	3	rw	SPE[i]_BIS_EIRQ **error interrupt enable** 0_B Disable error interrupt, error interrupt is not visible outside eGTM-IP 1_B Enable error interrupt, error interrupt is visible outside eGTM-IP
SPE_RCMP_EIRQ_EN	4	rw	SPE[i]_RCMP_EIRQ **error interrupt enable** 0_B Disable error interrupt, error interrupt is not visible outside eGTM-IP 1_B Enable error interrupt, error interrupt is visible outside eGTM-IP
0	31:5	r	**Reserved** Read as 0; must be written with 0.

44.5.156 SPE[i] Input Revolution Counter (i=0-2)

CLSi_SPE_REV_CNT (i=0-2) Offset address: $004C40_H+i*20000_H$

SPE[i] Input Revolution Counter Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								REV_CNT							
r								rw							

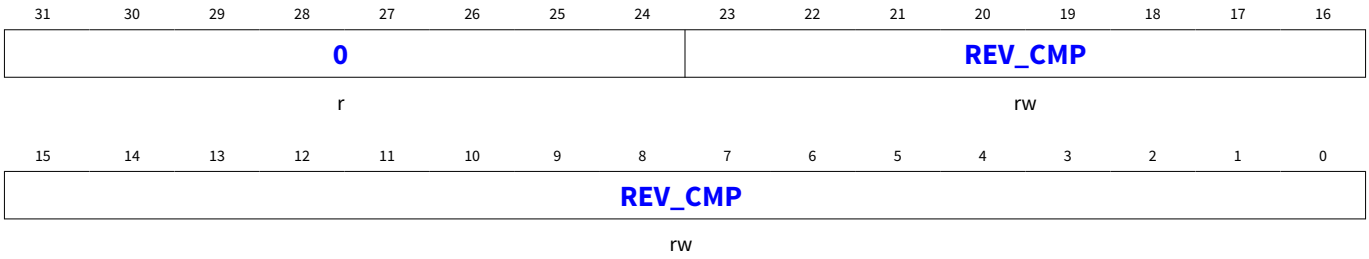
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
REV_CNT															
rw															

Field	Bits	Type	Description
REV_CNT	23:0	rw	**Input signal revolution counter** The counter is running if **SPE** module is enabled (bit CLSi_SPE_CTRL_STAT.EN). CLSi_SPE_REV_CNT.REV_CNT is incrementing if CLSi_SPE_CTRL_STAT.SPE_PAT_PTR is incrementing CLSi_SPE_REV_CNT.REV_CNT is decrementing if CLSi_SPE_CTRL_STAT.SPE_PAT_PTR is decrementing
0	31:24	r	**Reserved** Read as 0; must be written with 0.

44.5.157 SPE[i] Revolution Counter Compare Value (i=0-2)

CLSi_SPE_REV_CMP (i=0-2) Offset address: $004C44_H+i*20000_H$

SPE[i] Revolution Counter Compare Value Kernel Reset value: $0000\ 0000_H$

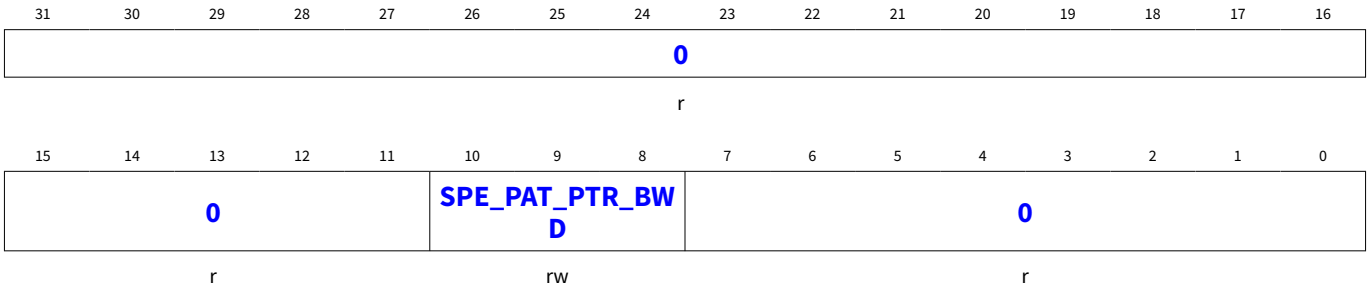


Field	Bits	Type	Description
REV_CMP	23:0	rw	**Input signal revolution counter compare value** The interrupt SPE[i]_RCMP is raised when the CLSi_SPE_REV_CNT.REV_CNT value equals the CLSi_SPE_REV_CMP.REV_CMP register field. ***Note**: SPE[i]_RCMP is only raised if CLSi_SPE_REV_CNT is incremented or decremented due to an input signal change. Any update of CLSi_SPE_REV_CNT or CLSi_SPE_REV_CMP via **AEI** does not raise a SPE[i]_RCMP interrupt.*
0	31:24	r	**Reserved** Read as 0; must be written with 0.

44.5.158 SPE[i] control status register 2 (i=0-2)

CLSi_SPE_CTRL_STAT2 (i=0-2) Offset address: $004C48_H+i*20000_H$

SPE[i] control status register 2 Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
SPE_PAT_PTR_BWD	10:8	rw	**Pattern selector for TOM output signals in case of CLSi_SPE_CMD.SPE_CTRL_CMD = 01_B (e.g. backward direction).** • Index into the CLSi_SPE_OUT_PATp register table in case of CLSi_SPE_CMD.SPE_CTRL_CMD = 01_B which may be used for backward direction. • Each register CLSi_SPE_OUT_PATp is fixed assigned to one bit field CLSi_SPE_PAT.IPt_PAT of register CLSi_SPE_PAT. Thus, the pointer CLSi_SPE_CTRL_STAT2.SPE_PAT_PTR_BWD represents an index to the selected CLSi_SPE_OUT_PATp register as well as the detected input pattern CLSi_SPE_PAT.IPt_PAT. • The index pointer CLSi_SPE_CTRL_STAT2.SPE_PAT_PTR_BWD is used if CLSi_SPE_CMD.SPE_CTRL_CMD = 01_B. • The index pointer CLSi_SPE_CTRL_STAT.SPE_PAT_PTR is used if CLSi_SPE_CMD.SPE_CTRL_CMD = 00_B (by default). • 000_B = CLSi_SPE_OUT_PATp**(p=0)** selected
0	7:0, 31:11	r	**Reserved** Read as 0; must be written with 0.

44.5.159 SPE[i] command register (i=0-2)

CLSi_SPE_CMD (i=0-2) Offset address: $004C4C_H+i*20000_H$

SPE[i] command register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															SPE_UPD_TRIG
r															rwh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0														SPE_CTRL_CMD	
r														rw	

Field	Bits	Type	Description
SPE_CTRL_CMD	1:0	rw	**SPE control command** ***Note**: On switch between 00_B and 01_B the direction flag CLSi_SPE_CTRL_STAT.ADIR will be independently set according to the input pattern CLSi_SPE_CTRL_STAT.NIP and CLSi_SPE_CTRL_STAT.AIP.* 00_B Use pi: CLSi_SPE_CTRL_STAT.SPE_PAT_PTR as an index pointer to select CLSi_SPE_OUT_PATp 01_B Use pi: CLSi_SPE_CTRL_STAT2.SPE_PAT_PTR_BWD as an index pointer to select CLSi_SPE_OUT_PATp 10_B Select CLSi_SPE_OUT_PATp (p=6) 11_B Select CLSi_SPE_OUT_PATp (p=7)
SPE_UPD_TRIG	16	rwh	**SPE updater trigger** ***Note**: This bit is automatically reset to 0.* 0_B Do nothing 1_B Trigger update of CLSi_SPE_OUT_CTRL.SPE_OUT_CTRLo with register selected by CLSi_SPE_CMD.SPE_CTRL_CMD multiplexer
0	15:2, 31:17	r	**Reserved** Read as 0; must be written with 0.

44.5.160 Clock control register

The Clock Control Register CLC allows the programmer to adapt the functionality and power consumption of the module to the requirements of the application.

Register CLC controls the module clock signal and the reactivity to the sleep signal.

CLC Offset address: 060000_H

Clock control register Application Reset value: $0000\ 0003_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												EDIS	0	DISS	DISR
r												rw	r	rh	rw

Field	Bits	Type	Description
DISR	0	rw	**Module Disable Request Bit** Used for enable/disable control of the module. 0_B On request: enable the module clock 1_B Off request: stop the module clock

(table continues...)

(continued)

Field	Bits	Type	Description
DISS	1	rh	**Module Disable Status Bit** 0_B Module clock is enabled 1_B Off: module is not clocked
EDIS	3	rw	**Sleep Mode Enable Control** Used to control the module's reaction to sleep mode. 0_B Sleep mode request is enabled and functional 1_B Module disregards the sleep mode control signal
0	2, 31:4	r	**Reserved, must not be used.** Read as 0; must be written with 0.

44.5.161 OCDS control and status register

OCS

OCDS control and status register

Offset address: 060004_H

Debug Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		SUSSTA	SUS_P	SUS				0							
r		rh	w	rw				r							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															
r															

Field	Bits	Type	Description
SUS	27:24	rw	**OCDS suspend control** Controls the sensitivity to the suspend signal coming from the OCDS Trigger Switch (OTGS). Not listed combinations are reserved. 0_H Will not suspend 1_H Reserved, must not be used. 2_H Reserved, must not be used. 3_H Soft suspend mode For soft-suspend, the GTM_HALT input of the GTM-IP is used. Thus, all outputs remain at their current state when soft-suspend is activated. Reads as well as writes on GTM-IP are possible.
SUS_P	28	w	**SUS write protection** SUS is only written when SUS_P is 1, otherwise unchanged. Read as 0.
SUSSTA	29	rh	**Suspend state** 0_B Module is not (yet) suspended 1_B Module is suspended
0	23:0, 31:30	r	**Reserved, must not be used.** Read as 0; shall be written with 0.

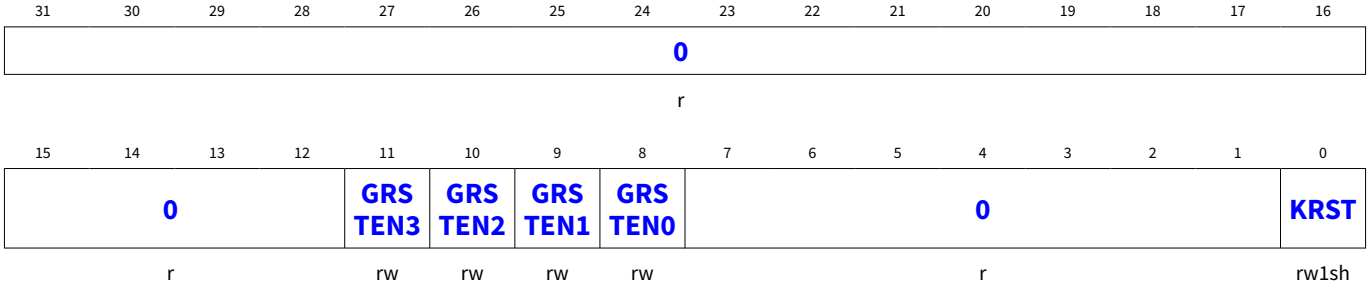
Table 2360 Access types of OCS in order of evaluation

Condition	Type		Description
write 1 to .**SUS_P**	rw	SUS	Set SUS_P during write access
otherwise	r	SUS	

44.5.162 Reset control register A

RST_CTRLA Offset address: 060010_H

Reset control register A Application Reset value: $0000\ 0000_H$

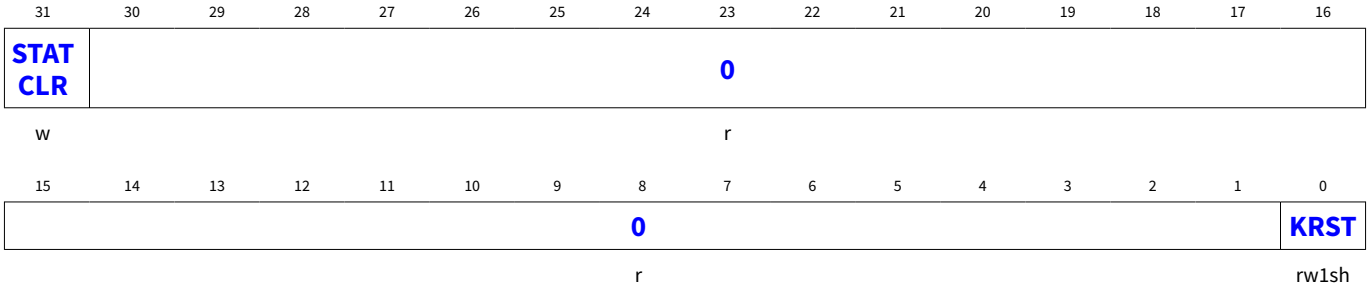


Field	Bits	Type	Description
KRST	0	rw1sh	**Kernel Reset** Request a kernel reset. The requested reset is executed if CTRLA.KRST=1 and CTRLB.KRST=1. KRST is cleared after the kernel reset was executed. 0_B No action 1_B A kernel reset was requested
GRSTENx (x=0-3)	x+8	rw	**Enable for Global Module Reset Group x** 0_B Global module reset group x does not have any effect 1_B Global module reset group x results in a kernel reset
0	7:1, 31:12	r	**Reserved, must not be used.** Read as 0; must be written with 0.

44.5.163 Reset control register B

RST_CTRLB Offset address: 060014_H

Reset control register B Application Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
KRST	0	rw1sh	**Kernel Reset** Request a kernel reset. The requested reset is executed if CTRLA.KRST=1 and CTRLB.KRST=1. KRST is cleared after the kernel reset was executed. 0_B No action 1_B A kernel reset was requested
STATCLR	31	w	**Kernel Reset Status Clear** Clears all status bits in RST_STAT when 1 is written. Read will return 0. 0_B No action 1_B Write with ´1´ clears bits STAT.GRSTx and bit STAT.KRST .
0	30:1	r	**Reserved, must not be used.** Read as 0; must be written with 0.

44.5.164 Reset status register

The reset status register contains the status bits for kernel reset (KRST) and the global module reset groups (GRSTx).

RST_STAT Offset address: 060018_H

Reset status register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				GRST3	GRST2	GRST1	GRST0	0							KRST
r				rh	rh	rh	rh	r							rh

Field	Bits	Type	Description
KRST	0	rh	**Kernel Reset Status** Indicates an executed kernel reset. RST_STAT.KRST is set after the execution of a kernel reset in the same clock cycle in which the reset bits are cleared. Clear KRST by setting bit STATCLR in register RST_CTRLB.
GRSTx (x=0-3)	x+8	rh	**Status for Global Module Reset Group x** 0_B Reset was not triggered by Global Reset Group x 1_B Reset was triggered by Global Reset Group x
0	7:1, 31:12	r	**Reserved, must not be used.** Read as 0.

44.5.165 PROT register Safe Endinit

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner or CS master to update the PROT state as described in the PROT mechanism.

The PROTSE register controls lock / unlock of the local Safe Endinit (SE) protected control registers.

PROTSE Offset address: $06001C_H$

PROT register Safe Endinit Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OWEN	ODEF	TAGID						PRSEN	PRS			VMEN	VM		
w	rw	rw						rw	rw			rw	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												SWEN	STATE		
r												w	rwh		

Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
VM	18:16	rw	**Virtual Machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual Machine definition Enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection Set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.

(table continues...)

(continued)

Field	Bits	Type	Description
PRSEN	23	rw	**Protection Set definition Enable for PROT owner** 0_B **Disable**: PRS is not part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved, must not be used.** Read as 0; must be written with 0.

Table 2361 Access types of PROTSE in order of evaluation

Condition	Type		Description
write 1 to .**SWEN**	rwh	STATE	
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
otherwise	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

44.5.166 Write access enable register A for cluster i (i=0-2)

This register defines which master TAG ID encodings are enabled or disabled for write accesses to the access protected region.

CLi_ACCEN_WRA (i=0-2) Offset address: 060024_H+i*24_H

Write access enable register A for cluster i Application Reset value: $1000\ 0003_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Write access enable for Master TAG ID q** This bit enables write access to the access protected region for transactions with the Master TAG ID q. 0_B Disabled for write access 1_B Enabled for write access

44.5.167 Write access enable register B for cluster i (i=0-2)

This register defines which master TAG ID encodings are enabled or disabled for write accesses to the access protected region.

CLi_ACCEN_WRB (i=0-2) Offset address: 060028_H+i*24_H

Write access enable register B for cluster i Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Write access enable for Master TAG ID q** This bit enables write access to the access protected region for transactions with the Master TAG ID q. 0_B Disabled for write access 1_B Enabled for write access
0	31:8	r	**Reserved, must not be used.** Read as 0; must be written with 0.

44.5.168 Read access enable register A for cluster i (i=0-2)

This register defines which master TAG ID encodings are enabled or disabled for read accesses to the access protected region.

CLi_ACCEN_RDA (i=0-2) Offset address: $06002C_H+i*24_H$

Read access enable register A for cluster i Application Reset value: $FFFF\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Read access enable for Master TAG ID q** This bit enables read access to the access protected region for transactions with the Master TAG ID q. 0_B Disabled for read access 1_B Enabled for read access

44.5.169 Read access enable register B for cluster i (i=0-2)

This register defines which master TAG ID encodings are enabled or disabled for read accesses to the access protected region.

CLi_ACCEN_RDB (i=0-2) Offset address: 060030_H+i*24_H

Read access enable register B for cluster i Application Reset value: $0000\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Read access enable for Master TAG ID q** This bit enables read access to the access protected region for transactions with the Master TAG ID q. 0_B Disabled for read access 1_B Enabled for read access
0	31:8	r	**Reserved, must not be used.** Read as 0; must be written with 0.

44.5.170 VM access enable register for cluster i (i=0-2)

This register defines which virtual machine encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each VMx.

CLi_ACCEN_VM (i=0-2) Offset address: 060034_H+i*24_H

VM access enable register for cluster i Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for VM ID q** This bit enables read access to the access protected region for transactions with the VM ID q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for VM ID q** This bit enables write access to the access protected region for transactions with the VM ID q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved, must not be used.** Read as 0; must be written with 0.

44.5.171 PRS access enable register for cluster i (i=0-2)

This register defines which protection register set encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each PRSx.

CLi_ACCEN_PRS (i=0-2) Offset address: 060038_H+i*24_H

PRS access enable register for cluster i Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for PRS q** This bit enables read access to the access protected region for transactions with the PRS q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for PRS q** This bit enables write access to the access protected region for transactions with the PRS q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved, must not be used.** Read as 0; must be written with 0.

44.5.172 PROT register Endinit for control logic

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner or CS master to update the PROT state as described in the PROT mechanism.

The PROTE register controls lock / unlock of the local Endinit (E) protected control registers.

CTRL_PROTE Offset address: $0601D0_H$

PROT register Endinit for control logic Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OWEN	ODEF	TAGID						PRSEN	PRS			VMEN	VM		
w	rw	rw						rw	rw			rw	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												SWEN	STATE		
r												w	rwh		

Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
VM	18:16	rw	**Virtual Machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual Machine definition Enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection Set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.

(table continues...)

(continued)

Field	Bits	Type	Description
PRSEN	23	rw	**Protection Set definition Enable for PROT owner** 0_B **Disable**: PRS is not part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved, must not be used.** Read as 0; must be written with 0.

Table 2362 Access types of CTRL_PROTE in order of evaluation

Condition	Type		Description
write 1 to .**SWEN**	rwh	STATE	
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
otherwise	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

44.5.173 Write access enable register A for control logic

This register defines which master TAG ID encodings are enabled or disabled for write accesses to the access protected region.

CTRL_ACCEN_WRA Offset address: $0601D4_H$

Write access enable register A for control logic Application Reset value: $1000\ 0003_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Write access enable for Master TAG ID q** This bit enables write access to the access protected region for transactions with the Master TAG ID q. 0_B Disabled for write access 1_B Enabled for write access

44.5.174 Write access enable register B for control logic

This register defines which master TAG ID encodings are enabled or disabled for write accesses to the access protected region.

CTRL_ACCEN_WRB Offset address: $0601D8_H$

Write access enable register B for control logic Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Write access enable for Master TAG ID q** This bit enables write access to the access protected region for transactions with the Master TAG ID q. 0_B Disabled for write access 1_B Enabled for write access
0	31:8	r	**Reserved, must not be used.** Read as 0; must be written with 0.

44.5.175 Read access enable register A for control logic

This register defines which master TAG ID encodings are enabled or disabled for read accesses to the access protected region.

CTRL_ACCEN_RDA Offset address: $0601DC_H$

Read access enable register A for control logic Application Reset value: $FFFF\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Read access enable for Master TAG ID q** This bit enables read access to the access protected region for transactions with the Master TAG ID q. 0_B Disabled for read access 1_B Enabled for read access

44.5.176 Read access enable register B for control logic

This register defines which master TAG ID encodings are enabled or disabled for read accesses to the access protected region.

CTRL_ACCEN_RDB Offset address: $0601E0_H$

Read access enable register B for control logic Application Reset value: $0000\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Read access enable for Master TAG ID q** This bit enables read access to the access protected region for transactions with the Master TAG ID q. 0_B Disabled for read access 1_B Enabled for read access
0	31:8	r	**Reserved, must not be used.** Read as 0; must be written with 0.

44.5.177 VM access enable register for control logic

This register defines which virtual machine encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each VMx.

CTRL_ACCEN_VM Offset address: $0601E4_H$

VM access enable register for control logic Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for VM ID q** This bit enables read access to the access protected region for transactions with the VM ID q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for VM ID q** This bit enables write access to the access protected region for transactions with the VM ID q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved, must not be used.** Read as 0; must be written with 0.

44.5.178 PRS access enable register for control logic

This register defines which protection register set encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each PRSx.

CTRL_ACCEN_PRS Offset address: $0601E8_H$

PRS access enable register for control logic Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for PRS q** This bit enables read access to the access protected region for transactions with the PRS q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for PRS q** This bit enables write access to the access protected region for transactions with the PRS q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved, must not be used.** Read as 0; must be written with 0.

44.5.179 Write access enable register A for wrapper logic

This register defines which master TAG ID encodings are enabled or disabled for write accesses to the access protected region.

WRAP_ACCEN_WRA Offset address: $0601F8_H$

Write access enable register A for wrapper logic Application Reset value: $1000\ 0003_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Write access enable for Master TAG ID q** This bit enables write access to the access protected region for transactions with the Master TAG ID q. 0_B Disabled for write access 1_B Enabled for write access

44.5.180 Write access enable register B for wrapper logic

This register defines which master TAG ID encodings are enabled or disabled for write accesses to the access protected region.

WRAP_ACCEN_WRB Offset address: $0601FC_H$

Write access enable register B for wrapper logic Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Write access enable for Master TAG ID q** This bit enables write access to the access protected region for transactions with the Master TAG ID q. 0_B Disabled for write access 1_B Enabled for write access
0	31:8	r	**Reserved, must not be used.** Read as 0; must be written with 0.

44.5.181 Read access enable register A for wrapper logic

This register defines which master TAG ID encodings are enabled or disabled for read accesses to the access protected region.

WRAP_ACCEN_RDA Offset address: 060200_H

Read access enable register A for wrapper logic Application Reset value: $FFFF\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Read access enable for Master TAG ID q** This bit enables read access to the access protected region for transactions with the Master TAG ID q. 0_B Disabled for read access 1_B Enabled for read access

44.5.182 Read access enable register B for wrapper logic

This register defines which master TAG ID encodings are enabled or disabled for read accesses to the access protected region.

WRAP_ACCEN_RDB Offset address: 060204_H

Read access enable register B for wrapper logic Application Reset value: $0000\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Read access enable for Master TAG ID q** This bit enables read access to the access protected region for transactions with the Master TAG ID q. 0_B Disabled for read access 1_B Enabled for read access
0	31:8	r	**Reserved, must not be used.** Read as 0; must be written with 0.

44.5.183 VM access enable register for wrapper logic

This register defines which virtual machine encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each VMx.

WRAP_ACCEN_VM Offset address: 060208_H

VM access enable register for wrapper logic Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for VM ID q** This bit enables read access to the access protected region for transactions with the VM ID q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for VM ID q** This bit enables write access to the access protected region for transactions with the VM ID q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved, must not be used.** Read as 0; must be written with 0.

44.5.184 PRS access enable register for wrapper logic

This register defines which protection register set encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each PRSx.

WRAP_ACCEN_PRS Offset address: $06020C_H$

PRS access enable register for wrapper logic Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for PRS q** This bit enables read access to the access protected region for transactions with the PRS q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for PRS q** This bit enables write access to the access protected region for transactions with the PRS q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved, must not be used.** Read as 0; must be written with 0.

44.5.185 Write access enable register A for wrapper logic

This register defines which master TAG ID encodings are enabled or disabled for write accesses to the access protected region.

CTRLGTM_ACCEN_WRA Offset address: $06021C_H$

Write access enable register A for wrapper logic Application Reset value: $1000\ 0003_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Write access enable for Master TAG ID q** This bit enables write access to the access protected region for transactions with the Master TAG ID q. 0_B Disabled for write access 1_B Enabled for write access

44.5.186 Write access enable register B for wrapper logic

This register defines which master TAG ID encodings are enabled or disabled for write accesses to the access protected region.

CTRLGTM_ACCEN_WRB Offset address: 060220_H

Write access enable register B for wrapper logic Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Write access enable for Master TAG ID q** This bit enables write access to the access protected region for transactions with the Master TAG ID q. 0_B Disabled for write access 1_B Enabled for write access
0	31:8	r	**Reserved, must not be used.** Read as 0; must be written with 0.

44.5.187 Read access enable register A for wrapper logic

This register defines which master TAG ID encodings are enabled or disabled for read accesses to the access protected region.

CTRLGTM_ACCEN_RDA Offset address: 060224_H

Read access enable register A for wrapper logic Application Reset value: $FFFF\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Read access enable for Master TAG ID q** This bit enables read access to the access protected region for transactions with the Master TAG ID q. 0_B Disabled for read access 1_B Enabled for read access

44.5.188 Read access enable register B for wrapper logic

This register defines which master TAG ID encodings are enabled or disabled for read accesses to the access protected region.

CTRLGTM_ACCEN_RDB Offset address: 060228_H

Read access enable register B for wrapper logic Application Reset value: $0000\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Read access enable for Master TAG ID q** This bit enables read access to the access protected region for transactions with the Master TAG ID q. 0_B Disabled for read access 1_B Enabled for read access
0	31:8	r	**Reserved, must not be used.** Read as 0; must be written with 0.

44.5.189 VM access enable register for wrapper logic

This register defines which virtual machine encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each VMx.

CTRLGTM_ACCEN_VM Offset address: $06022C_H$

VM access enable register for wrapper logic Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for VM ID q** This bit enables read access to the access protected region for transactions with the VM ID q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for VM ID q** This bit enables write access to the access protected region for transactions with the VM ID q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved, must not be used.** Read as 0; must be written with 0.

44.5.190 PRS access enable register for wrapper logic

This register defines which protection register set encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each PRSx.

CTRLGTM_ACCEN_PRS Offset address: 060230_H

PRS access enable register for wrapper logic Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for PRS q** This bit enables read access to the access protected region for transactions with the PRS q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for PRS q** This bit enables write access to the access protected region for transactions with the PRS q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved, must not be used.** Read as 0; must be written with 0.

44.5.191 Write access enable register A for wrapper logic

This register defines which master TAG ID encodings are enabled or disabled for write accesses to the access protected region.

SAFEGTM_ACCEN_WRA Offset address: 060240_H

Write access enable register A for wrapper logic Application Reset value: $1000\ 0003_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Write access enable for Master TAG ID q** This bit enables write access to the access protected region for transactions with the Master TAG ID q. 0_B Disabled for write access 1_B Enabled for write access

44.5.192 Write access enable register B for wrapper logic

This register defines which master TAG ID encodings are enabled or disabled for write accesses to the access protected region.

SAFEGTM_ACCEN_WRB Offset address: 060244_H

Write access enable register B for wrapper logic Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Write access enable for Master TAG ID q** This bit enables write access to the access protected region for transactions with the Master TAG ID q. 0_B Disabled for write access 1_B Enabled for write access
0	31:8	r	**Reserved, must not be used.** Read as 0; must be written with 0.

44.5.193 Read access enable register A for wrapper logic

This register defines which master TAG ID encodings are enabled or disabled for read accesses to the access protected region.

SAFEGTM_ACCEN_RDA Offset address: 060248_H

Read access enable register A for wrapper logic Application Reset value: $FFFF\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Read access enable for Master TAG ID q** This bit enables read access to the access protected region for transactions with the Master TAG ID q. 0_B Disabled for read access 1_B Enabled for read access

44.5.194 Read access enable register B for wrapper logic

This register defines which master TAG ID encodings are enabled or disabled for read accesses to the access protected region.

SAFEGTM_ACCEN_RDB Offset address: $06024C_H$

Read access enable register B for wrapper logic Application Reset value: $0000\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Read access enable for Master TAG ID q** This bit enables read access to the access protected region for transactions with the Master TAG ID q. 0_B Disabled for read access 1_B Enabled for read access
0	31:8	r	**Reserved, must not be used.** Read as 0; must be written with 0.

44.5.195 VM access enable register for wrapper logic

This register defines which virtual machine encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each VMx.

SAFEGTM_ACCEN_VM Offset address: 060250_H

VM access enable register for wrapper logic Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for VM ID q** This bit enables read access to the access protected region for transactions with the VM ID q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for VM ID q** This bit enables write access to the access protected region for transactions with the VM ID q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved, must not be used.** Read as 0; must be written with 0.

44.5.196 PRS access enable register for wrapper logic

This register defines which protection register set encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each PRSx.

SAFEGTM_ACCEN_PRS Offset address: 060254_H

PRS access enable register for wrapper logic Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for PRS q** This bit enables read access to the access protected region for transactions with the PRS q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for PRS q** This bit enables write access to the access protected region for transactions with the PRS q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved, must not be used.** Read as 0; must be written with 0.

44.5.197 Spare Bits

SPARE Offset address: 060480_H

Spare Bits Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SPABH				SPABG				SPABF				SPABE			
rwh				rwh				rwh				rwh			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SPABD				SPABC				SPABB				SPABA			
rwh				rwh				rwh				rwh			

Field	Bits	Type	Description
SPABA	3:0	rwh	**Spare Bits A - For future extensions** 0_H Disabled 1_H Setting 1 2_H Setting 2 3_H Setting 3
SPABB	7:4	rwh	**Spare Bits B - for future extensions** 0_H Disabled 1_H Setting 1 2_H Setting 2 3_H Setting 3
SPABC	11:8	rwh	**Spare Bits C - for future extensions** 0_H Disabled 1_H Setting 1 2_H Setting 2 3_H Setting 3
SPABD	15:12	rwh	**Spare Bits D - for future extensions** 0_H Disabled 1_H Setting 1 2_H Setting 2 3_H Setting 3
SPABE	19:16	rwh	**Spare Bits E - for future extensions** 0_H Disabled 1_H Setting 1 2_H Setting 2 3_H Setting 3
SPABF	23:20	rwh	**Spare Bits F - for future extensions** 0_H Disabled 1_H Setting 1 2_H Setting 2 3_H Setting 3

(table continues...)

(continued)

Field	Bits	Type	Description
SPABG	27:24	rwh	**Spare Bits G - for future extensions** 0_H Disabled 1_H Setting 1 2_H Setting 2 3_H Setting 3
SPABH	31:28	rwh	**Spare Bits H - for future extensions** 0_H Disabled 1_H Setting 1 2_H Setting 2 3_H Setting 3

44.5.198 OCDS TBU0 trigger register

OCDS_OTBU0T Offset address: 060500_H

OCDS TBU0 trigger register Debug Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		CM		0	CV										
r		rw		r	rw										

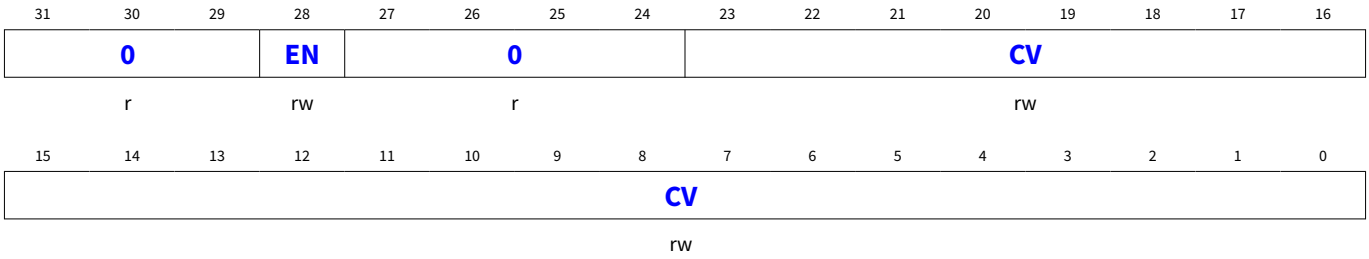
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CV															
rw															

Field	Bits	Type	Description
CV	26:0	rw	**Compare value** This value is compared to the TBU_CH0_BASE register. As long as both match, the associated TS16_IOS.MISC bit in table "TS16_IOS trigger set IO and other signals" is active.
CM	29:28	rw	**Compare mode** 00_B Disabled 01_B Compare lower 24 bits 10_B Compare upper 24 bits 11_B Compare all 27 bits
0	27, 31:30	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.5.199 OCDS TBU1 trigger register

OCDS_OTBU1T Offset address: 060504_H

OCDS TBU1 trigger register Debug Reset value: $0000\ 0000_H$

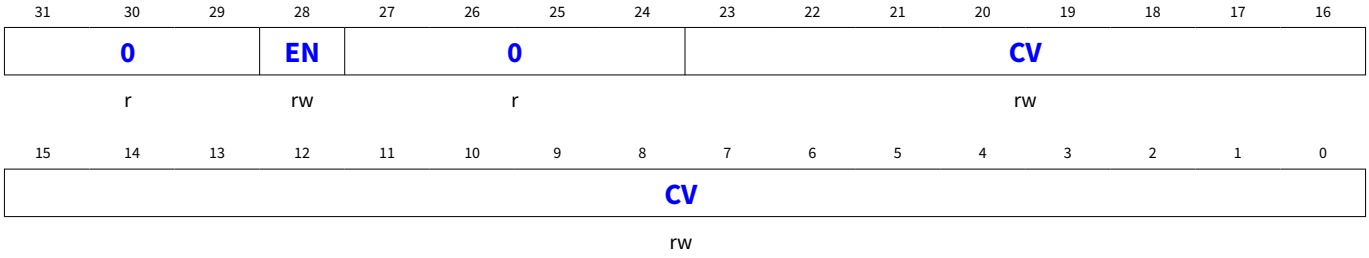


Field	Bits	Type	Description
CV	23:0	rw	**Compare value** This value is compared to the TBU_CH1_BASE register. As long as both match, the associated TS16_IOS.MISC bit in table "TS16_IOS trigger set IO and other signals" is active.
EN	28	rw	**Enable** 0_B Disabled 1_B Enabled
0	27:24, 31:29	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.5.200 OCDS TBU2 trigger register

OCDS_OTBU2T Offset address: 060508_H

OCDS TBU2 trigger register Debug Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
CV	23:0	rw	**Compare value** This value is compared to the TBU_CH2_BASE register. As long as both match, the associated TS16_IOS.MISC bit bit in table "TS16_IOS trigger set IO and other signals" is active.
EN	28	rw	**Enable** 0_B Disabled 1_B Enabled
0	27:24, 31:29	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.5.201 OCDS trigger set select register

OCDS_OTSS Offset address: 060510_H

OCDS trigger set select register Debug Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				OTGB1				0				OTGB0			
r				rw				r				rw			

Field	Bits	Type	Description
OTGB0	3:0	rw	**Trigger set for OTGB0** 0_H No trigger set selected 1_H Trigger set TS16_IOS in table "TS16_IOS trigger set IO and other signals" **others,** Reserved, must not be used.
OTGB1	11:8	rw	**Trigger set for OTGB1** 0_H No trigger set selected 1_H Trigger set TS16_IOS in table "TS16_IOS trigger set IO and other signals" **others,** Reserved, must not be used.
0	7:4, 15:12, 31:16	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.5.202 OCDS trigger set control 0 register

OCDS_OTSC0 Offset address: 060514_H

OCDS trigger set control 0 register Debug Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
B1HMI				0	B1HMT			B1LMI				0	B1LMT		
rw				r	rw			rw				r	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
B0HMI				0	B0HMT			B0LMI				0	B0LMT		
rw				r	rw			rw				r	rw		

Field	Bits	Type	Description
B0LMT	2:0	rw	**OTGB0 TS16_IOS low-byte module type** 000_B No module selected 001_B TIM (F_OUT[7:0]) 010_B TOML (TOM_OUT[7:0]) 011_B TOMH (TOM_OUT[15:8]) 100_B ATOM (ATOM_OUT[7:0]) 101_B SPE signals in table "TS16_IOS trigger set IO and other signals" 110_B MISC signals in table "TS16_IOS trigger set IO and other signals" **others,** Reserved, must not be used.
B0LMI	7:4	rw	**OTGB0 TS16_IOS low-byte module instance** Index of the module instance. Index starts with 0, the max. value depends on eGTM configuration and module type. For SPE signals, there is a specific mapping in table "TS16_IOS trigger set IO and other signals". For MISC signals, this index is ignored.
B0HMT	10:8	rw	**OTGB0 TS16_IOS high-byte module type** 000_B No module selected 001_B TIM (F_OUT[7:0]) 010_B TOML (TOM_OUT[7:0]) 011_B TOMH (TOM_OUT[15:8]) 100_B ATOM (ATOM_OUT[7:0]) 101_B SPE signals in table "TS16_IOS trigger set IO and other signals" 110_B MISC signals in table "TS16_IOS trigger set IO and other signals" **others,** Reserved, must not be used.
B0HMI	15:12	rw	**OTGB0 TS16_IOS high-byte module instance** Index of the module instance. Index starts with 0, the max. value depends on eGTM configuration and module type. For SPE signals, there is a specific mapping in table "TS16_IOS trigger set IO and other signals". For MISC signals, this index is ignored.
B1LMT	18:16	rw	**OTGB1 TS16_IOS low-byte module type** 000_B No module selected 001_B TIM (F_OUT[7:0]) 010_B TOML (TOM_OUT[7:0]) 011_B TOMH (TOM_OUT[15:8]) 100_B ATOM (ATOM_OUT[7:0]) 101_B SPE signals in table "TS16_IOS trigger set IO and other signals" 110_B MISC signals in table "TS16_IOS trigger set IO and other signals" **others,** Reserved, must not be used.
B1LMI	23:20	rw	**OTGB1 TS16_IOS low-byte module instance** Index of the module instance. Index starts with 0, the max. value depends on eGTM configuration and module type. For SPE signals, there is a specific mapping in table "TS16_IOS trigger set IO and other signals". For MISC signals, this index is ignored.

(table continues...)

(continued)

Field	Bits	Type	Description
B1HMT	26:24	rw	**OTGB1 TS16_IOS high-byte module type** 000_B No module selected 001_B TIM (F_OUT[7:0]) 010_B TOML (TOM_OUT[7:0]) 011_B TOMH (TOM_OUT[15:8]) 100_B ATOM (ATOM_OUT[7:0]) 101_B SPE signals in table "TS16_IOS trigger set IO and other signals" 110_B MISC signals in table "TS16_IOS trigger set IO and other signals" **others,** Reserved, must not be used.
B1HMI	31:28	rw	**OTGB1 TS16_IOS high-byte module instance** Index of the module instance. Index starts with 0, the max. value depends on eGTM configuration and module type. For SPE signals, there is a specific mapping in table "TS16_IOS trigger set IO and other signals". For MISC signals, this index is ignored.
0	3, 11, 19, 27	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.5.203 TIMn input selection register (n=0-2)

TIMnINSEL (n=0-2) Offset address: 061000_H+n*4

TIMn input selection register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CH7SEL				CH6SEL				CH5SEL				CH4SEL			
rw				rw				rw				rw			

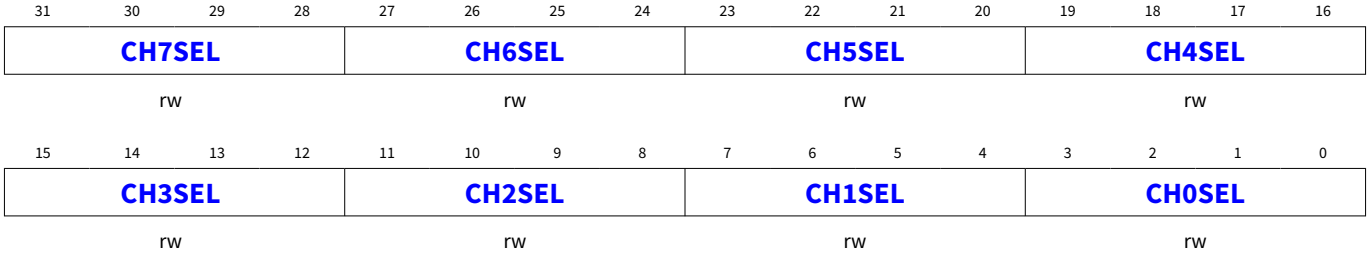
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH3SEL				CH2SEL				CH1SEL				CH0SEL			
rw				rw				rw				rw			

Field	Bits	Type	Description
CHxSEL (x=0-7)	4*x+3:4*x	rw	**eGTM TIMn channel x input selection** This bit field selects which input is connected for TIMn channel x. The input is either derived from a port pad or from an on-chip module.

44.5.204 TIMn input selection register group B (n=0-2)

TIMnINSELB (n=0-2) Offset address: 061050_H+n*4

TIMn input selection register group B Kernel Reset value: $0000\ 0000_H$



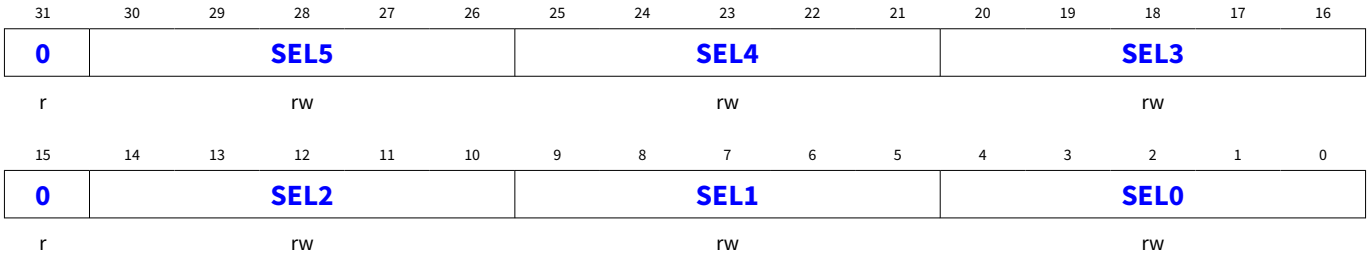
Field	Bits	Type	Description
CHxSEL (x=0-7)	4*x+3:4*x	rw	**eGTM TIMn channel x input selection group B** This bit field selects which input is connected for TIMn channel x (group B). The input is either derived from a port pad or from an on-chip module.

44.5.205 eGTM to Ports signal TOUT output selection register n (n=0-48)

The bit fields in this register define which TOM/ATOM channel output is used as port signal TOUT(n*6+x).

TOUTSELn (n=0-48) Offset address: 061100_H+n*4

eGTM to Ports signal TOUT output selection register n Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
SELx (x=0-2)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection**
SELx (x=3-5)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection**
0	15, 31	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.5.206 eGTM to MSC SETi output selection register j (j=0;i=0-3)

The bit fields in this register define which TOM/ATOM channel output is used as SETi signal x.

MSCSETi_CONj (i=0-3;j=0) Offset address: 061200_H+i*10_H+j*4

eGTM to MSC SETi output selection register j Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0			SEL3					0			SEL2				
r			rw					r			rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			SEL1					0			SEL0				
r			rw					r			rw				

Field	Bits	Type	Description
SELx (x=0-3)	8*x+4:8*x	rw	**SETi.x signal output selection**
0	31:29, 23:21, 15:13, 7:5	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.5.207 eGTM to MSC SETi output selection register j (j=1;i=0-3)

For a description of this register, see eGTM to MSC SETi output selection register j (j=0;i=0-3).

MSCSETi_CONj (i=0-3;j=1) Offset address: 061200_H+i*10_H+j*4

eGTM to MSC SETi output selection register j Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0			SEL7					0			SEL6				
r			rw					r			rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			SEL5					0			SEL4				
r			rw					r			rw				

Field	Bits	Type	Description
SELx (x=4-7)	8*x-28:8*x-32	rw	**SETi.x signal output selection**
0	31:29, 23:21, 15:13, 7:5	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.5.208 eGTM to MSC SETi output selection register j (j=2;i=0-3)

For a description of this register, see eGTM to MSC SETi output selection register j (j=0;i=0-3).

MSCSETi_CONj (i=0-3;j=2) Offset address: 061200_H+i*10_H+j*4

eGTM to MSC SETi output selection register j Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0			SEL11					0			SEL10				
r			rw					r			rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			SEL9					0			SEL8				
r			rw					r			rw				

Field	Bits	Type	Description
SELx (x=8-11)	8*x-60:8*x-64	rw	**SETi.x signal output selection**
0	31:29, 23:21, 15:13, 7:5	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.5.209 eGTM to MSC SETi output selection register j (j=3;i=0-3)

For a description of this register, see eGTM to MSC SETi output selection register j (j=0;i=0-3).

MSCSETi_CONj (i=0-3;j=3) Offset address: 061200_H+i*10_H+j*4

eGTM to MSC SETi output selection register j Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0			SEL15					0			SEL14				
r			rw					r			rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			SEL13					0			SEL12				
r			rw					r			rw				

Field	Bits	Type	Description
SELx (x=12-15)	8*x-92:8*x-96	rw	**SETi.x signal output selection**
0	31:29, 23:21, 15:13, 7:5	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.5.210 eGTM to input bus MSCn_INLCON signal selection register (n=0-1)

The bit fields in this register define which SETi.x signal is selected for the MSCn_INLCON.x bus signal.

MSCn_INLCON (n=0-1) Offset address: 061290_H+n*10_H

eGTM to input bus MSCn_INLCON signal selection register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SEL15		SEL14		SEL13		SEL12		SEL11		SEL10		SEL9		SEL8	
rw		rw		rw		rw		rw		rw		rw		rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SEL7		SEL6		SEL5		SEL4		SEL3		SEL2		SEL1		SEL0	
rw		rw		rw		rw		rw		rw		rw		rw	

Field	Bits	Type	Description
SELx (x=0-15)	2*x+1:2*x	rw	**nth bus line INLCONx SETi signal selection**

44.5.211 eGTM to input bus MSCn_INHCON signal selection register (n=0-1)

The bit fields in this register define which SETi.x signal is selected for the MSCn_INHCON.x bus signal.

MSCn_INHCON (n=0-1)

Offset address: 061294_H+n*10_H

eGTM to input bus MSCn_INHCON signal selection register

Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SEL15		SEL14		SEL13		SEL12		SEL11		SEL10		SEL9		SEL8	
rw		rw		rw		rw		rw		rw		rw		rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SEL7		SEL6		SEL5		SEL4		SEL3		SEL2		SEL1		SEL0	
rw		rw		rw		rw		rw		rw		rw		rw	

Field	Bits	Type	Description
SELx (x=0-15)	2*x+1:2*x	rw	**nth bus line INHCON.x SETi signal selection**

44.5.212 eGTM to input bus MSCn_INLECON signal selection register (n=0-1)

The bit fields in this register define which SETi.x signals are selected for the MSCn_INLECON.x bus signal.

MSCn_INLECON (n=0-1)

Offset address: 061298_H+n*10_H

eGTM to input bus MSCn_INLECON signal selection register

Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SEL15		SEL14		SEL13		SEL12		SEL11		SEL10		SEL9		SEL8	
rw		rw		rw		rw		rw		rw		rw		rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SEL7		SEL6		SEL5		SEL4		SEL3		SEL2		SEL1		SEL0	
rw		rw		rw		rw		rw		rw		rw		rw	

Field	Bits	Type	Description
SELx (x=0-15)	2*x+1:2*x	rw	**nth bus line INLECON.x SETi signal selection**

44.5.213 eGTM to input bus MSCn_INHECON signal selection register (n=0-1)

The bit fields in this register define which SETi.x signal is selected for the MSCn_INHECON.x bus signal.

MSCn_INHECON (n=0-1) Offset address: $06129C_H+n*10_H$

eGTM to input bus MSCn_INHECON signal selection register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SEL15		SEL14		SEL13		SEL12		SEL11		SEL10		SEL9		SEL8	
rw		rw		rw		rw		rw		rw		rw		rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SEL7		SEL6		SEL5		SEL4		SEL3		SEL2		SEL1		SEL0	
rw		rw		rw		rw		rw		rw		rw		rw	

Field	Bits	Type	Description
SELx (x=0-15)	2*x+1:2*x	rw	**nth bus line INHECON.x SETi signal selection**

44.5.214 Selection of INLCON stream for MSC

MSCSELIN_INLCON Offset address: $0612D0_H$

Selection of INLCON stream for MSC Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								0					MMSC2		
r								r					rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0					MMSC1			0					MMSC0		
r					rw			r					rw		

Field	Bits	Type	Description
MMSCk (k=0-2)	8*k+2:8*k	rw	**Select the INLCON source of the existing MSCk** Select the INLCON source of the existing MSCk, with k being the number of available MSC modules minus one of the device. 000_B Map INLCONk to MSCk 100_B Map INLCON0 to MSCk 101_B Map INLCON1 to MSCk 110_B Map INLCON2 to MSCk **others,** Reserved, must not be used.
0	31:24, 23:19, 15:11, 7:3	r	**Reserved, must not be used.**

44.5.215 Selection of INHCON stream for MSC

MSCSELIN_INHCON Offset address: $0612D4_H$

Selection of INHCON stream for MSC Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								0					MMSC2		
r								r					rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0					MMSC1			0					MMSC0		
r					rw			r					rw		

Field	Bits	Type	Description
MMSCk (k=0-2)	8*k+2:8*k	rw	**Select the INHCON source of the existing MSCk** Select the INHCON source of the existing MSCk, with k being the number of available MSC modules minus one of the device. 000_B Map INHCONk to MSCk 100_B Map INHCON0 to MSCk 101_B Map INHCON1 to MSCk 110_B Map INHCON2 to MSCk **others,** Reserved, must not be used.
0	31:24, 23:19, 15:11, 7:3	r	**Reserved, must not be used.**

44.5.216 Selection of INLECON stream for MSC

MSCSELIN_INLECON Offset address: $0612D8_H$

Selection of INLECON stream for MSC Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								0					MMSC2		
r								r					rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0					MMSC1			0					MMSC0		
r					rw			r					rw		

Field	Bits	Type	Description
MMSCk (k=0-2)	8*k+2:8*k	rw	**Select the INLECON source of the existing MSCk** Select the INLECON source of the existing MSCk, with k being the number of available MSC modules minus one of the device. 000_B Map INLECONk to MSCk 100_B Map INLECON0 to MSCk 101_B Map INLECON1 to MSCk 110_B Map INLECON2 to MSCk **others,** Reserved, must not be used.
0	31:24, 23:19, 15:11, 7:3	r	**Reserved, must not be used.**

44.5.217 Selection of INHECON stream for MSC

MSCSELIN_INHECON Offset address: $0612DC_H$

Selection of INHECON stream for MSC Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								0					MMSC2		
r								r					rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0					MMSC1			0					MMSC0		
r					rw			r					rw		

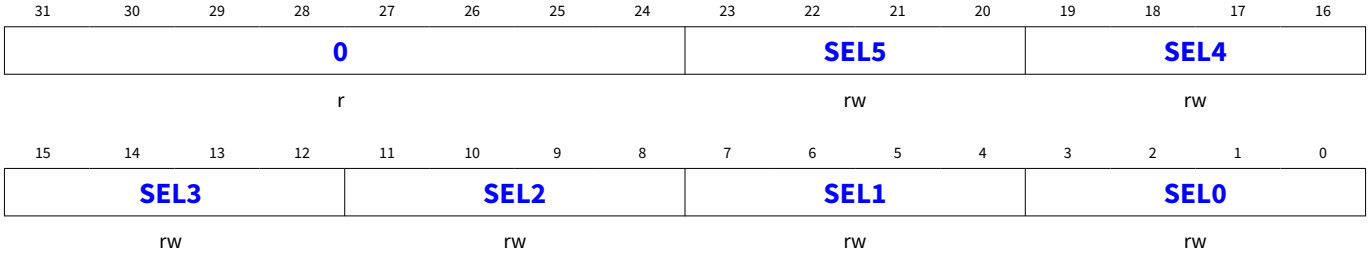
Field	Bits	Type	Description
MMSCk (k=0-2)	8*k+2:8*k	rw	**Select the INHECON source of the existing MSCk** Select the INHECON source of the existing MSCk, with k being the number of available MSC modules minus one of the device. 000_B Map INHECONk to MSCk 100_B Map INHECON0 to MSCk 101_B Map INHECON1 to MSCk 110_B Map INHECON2 to MSCk **others,** Reserved, must not be used.
0	31:24, 23:19, 15:11, 7:3	r	**Reserved, must not be used.**

44.5.218 eGTM to PSI5 output trigger selection register

The bit fields in this register define which TOM/ATOM channel output is used as PSI5 trigger x.

PSI5_OUT Offset address: 061400_H

eGTM to PSI5 output trigger selection register Kernel Reset value: $0000\ 0000_H$



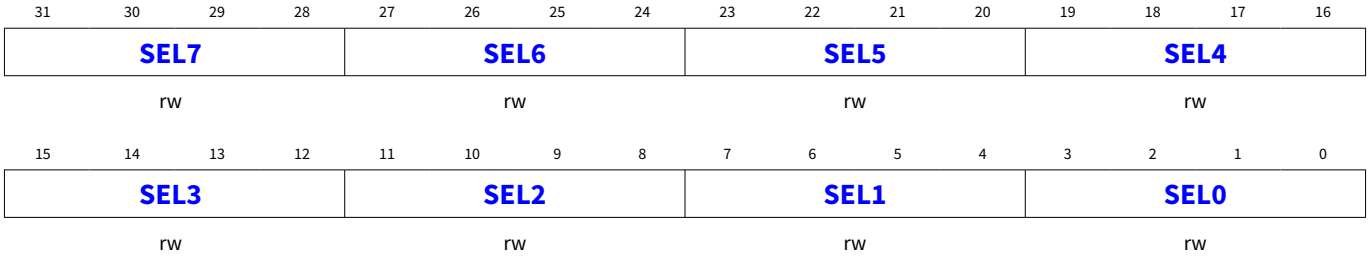
Field	Bits	Type	Description
SELx (x=0-5)	4*x+3:4*x	rw	**eGTM to PSI5 trigger x output selection**
0	31:24	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.5.219 eGTM to PSI5-Sn output trigger selection register (n=0-1)

The bit fields in this register define which TOM/ATOM channel output is used as PSI5-Sn trigger x.

PSI5Sn_OUT (n=0-1) Offset address: 061410_H+n*4

eGTM to PSI5-Sn output trigger selection register Kernel Reset value: $0000\ 0000_H$



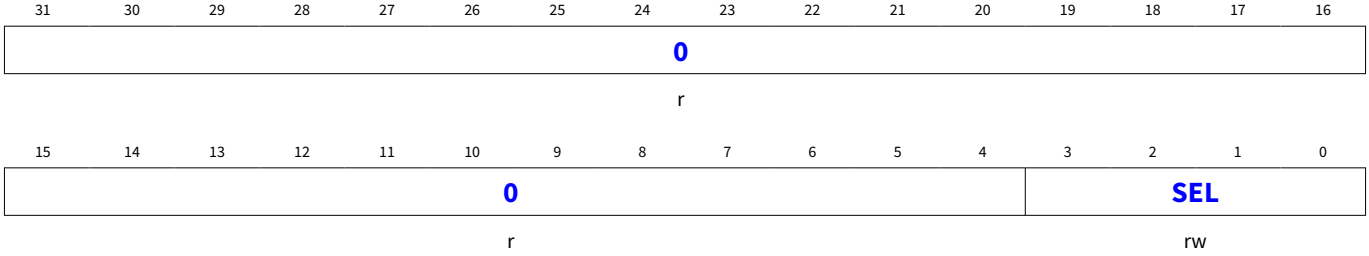
Field	Bits	Type	Description
SELx (x=0-7)	4*x+3:4*x	rw	**eGTM to PSI5-Sn trigger x output selection**

44.5.220 eGTM to LCDCDC output trigger selection register

The bit field in this register defines which TOM/ATOM channel output is used as LCDCDC trigger.

LCDCDC_OUT Offset address: 061420_H

eGTM to LCDCDC output trigger selection register Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
SEL	3:0	rw	**eGTM to LCDCDC trigger output selection**
0	31:4	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.5.221 eGTM to CANn output selection register x. n*4+ x<24 , if interface exists. (x=0-3;n=0-5)

The bit field in this register defines which TOM/ATOM channel output is used as CANn node trigger x.

CANn_OUTx (n=0-5;x=0-3)

Offset address: 061480_H+n*10_H+x*4

eGTM to CANn output selection register x. n*4+ x<24 , if interface exists.

Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												SEL			
r												rw			

Field	Bits	Type	Description
SEL	3:0	rw	**eGTM to CANn trigger x output selection**
0	31:4	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.5.222 FOUT Selection for Encoder Interface x, if GPT resources exist. (x=0-3)

FOUTENCx (x=0-3)

Offset address: 061500_H+x*4

FOUT Selection for Encoder Interface x, if GPT resources exist.

Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN_CAPB	CH_CAPB			0	TIM_CAPB			EN_TOPZ	CH_TOPZ			0	TIM_TOPZ		
rw	rw			r	rw			rw	rw			r	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN_EUD	CH_EUD			0	TIM_EUD			EN_IN	CH_IN			0	TIM_IN		
rw	rw			r	rw			rw	rw			r	rw		

Field	Bits	Type	Description
TIM_IN	2:0	rw	**TIM module selection for encoder interface input IN of GPT12x** TIM out of corresponding cluster. No function if TIM number does not exist. 000_B **TIM0**: TIM of cluster 0 selected for IN input of GPT12x 001_B **TIM1**: TIM of cluster 1 selected for IN input of GPT12x 010_B **TIM2**: TIM of cluster 2 selected for IN input of GPT12x 011_B **Reserved**: Reserved, do not use 100_B **Reserved**: Reserved, do not use 101_B **Reserved**: Reserved, do not use 110_B **Reserved**: Reserved, do not use 111_B **Reserved**: Reserved, do not use
CH_IN	6:4	rw	**TIM channel selection for encoder interface input IN of GPT12x** 000_B **CH0**: Channel 0 within TIM selected for IN input of GPT12x 001_B **CH1**: Channel 1 within TIM selected for IN input of GPT12x 010_B **CH2**: Channel 2 within TIM selected for IN input of GPT12x 011_B **CH3**: Channel 3 within TIM selected for IN input of GPT12x 100_B **CH4**: Channel 4 within TIM selected for IN input of GPT12x 101_B **CH5**: Channel 5 within TIM selected for IN input of GPT12x 110_B **CH6**: Channel 6 within TIM selected for IN input of GPT12x 111_B **CH7**: Channel 7 within TIM selected for IN input of GPT12x
EN_IN	7	rw	**Enable connection to encoder interface input IN of GPT12x** 0_B **0**: Disable the connection of eGTM to GPT12x_IN 1_B **1**: Enable the connection of eGTM to GPT12x_IN
TIM_EUD	10:8	rw	**TIM module selection for encoder interface input EUD of GPT12x** 000_B **TIM0**: TIM of cluster 0 selected for EUD input of GPT12x 001_B **TIM1**: TIM of cluster 1 selected for EUD input of GPT12x 010_B **TIM2**: TIM of cluster 2 selected for EUD input of GPT12x 011_B **Reserved**: Reserved, must not be used. 100_B **Reserved**: Reserved, must not be used. 101_B **Reserved**: Reserved, must not be used. 110_B **Reserved**: Reserved, must not be used. 111_B **Reserved**: Reserved, must not be used.
CH_EUD	14:12	rw	**TIM channel selection for encoder interface input EUD of GPT12x** 000_B **CH0**: Channel 0 within TIM selected for EUD input of GPT12x 001_B **CH1**: Channel 1 within TIM selected for EUD input of GPT12x 010_B **CH2**: Channel 2 within TIM selected for EUD input of GPT12x 011_B **CH3**: Channel 3 within TIM selected for EUD input of GPT12x 100_B **CH4**: Channel 4 within TIM selected for EUD input of GPT12x 101_B **CH5**: Channel 5 within TIM selected for EUD input of GPT12x 110_B **CH6**: Channel 6 within TIM selected for EUD input of GPT12x 111_B **CH7**: Channel 7 within TIM selected for EUD input of GPT12x

(table continues...)

(continued)

Field	Bits	Type	Description
EN_EUD	15	rw	**Enable connection to encoder interface input EUD of GPT12x** 0_B **0**: Disable the connection of eGTM to GPT12x_EUD 1_B **1**: Enable the connection of eGTM to GPT12x_EUD
TIM_TOPZ	18:16	rw	**TIM module selection for encoder interface input TOP Zero (T4IN) of GPT12x** 000_B **TIM0**: TIM of cluster 0 selected for TOPZ input of GPT12x 001_B **TIM1**: TIM of cluster 1 selected for TOPZ input of GPT12x 010_B **TIM2**: TIM of cluster 2 selected for TOPZ input of GPT12x 011_B **Reserved**: Reserved, do not use 100_B **Reserved**: Reserved, do not use 101_B **Reserved**: Reserved, do not use 110_B **Reserved**: Reserved, do not use 111_B **Reserved**: Reserved, do not use
CH_TOPZ	22:20	rw	**TIM channel selection for encoder interface input TOP Zero (T4IN) of GPT12x** 000_B **CH0**: Channel 0 within TIM selected for TOPZ input of GPT12x 001_B **CH1**: Channel 1 within TIM selected for TOPZ input of GPT12x 010_B **CH2**: Channel 2 within TIM selected for TOPZ input of GPT12x 011_B **CH3**: Channel 3 within TIM selected for TOPZ input of GPT12x 100_B **CH4**: Channel 4 within TIM selected for TOPZ input of GPT12x 101_B **CH5**: Channel 5 within TIM selected for TOPZ input of GPT12x 110_B **CH6**: Channel 6 within TIM selected for TOPZ input of GPT12x 111_B **CH7**: Channel 7 within TIM selected for TOPZ input of GPT12x
EN_TOPZ	23	rw	**Enable connection to encoder interface input TOP Zero (T4IN) of GPT12x** 0_B **0**: Disable the connection of eGTM to GPT12x_TOPZ (T4IN) 1_B **1**: Enable the connection of eGTM to GPT12x_TOPZ (T4IN)
TIM_CAPB	26:24	rw	**TIM module selection for CAPINB input of GPT12x** 000_B **TIM0**: TIM of cluster 0 selected for CAPINB input of GPT12x 001_B **TIM1**: TIM of cluster 1 selected for CAPINB input of GPT12x 010_B **TIM2**: TIM of cluster 2 selected for CAPINB input of GPT12x 011_B **Reserved**: Reserved, do not use 100_B **Reserved**: Reserved, do not use 101_B **Reserved**: Reserved, do not use 110_B **Reserved**: Reserved, do not use 111_B **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
CH_CAPB	30:28	rw	**TIM channel selection for CAPINB input of GPT12x** 000_B **CH0**: Channel 0 within TIM selected for CAPINB input of GPT12x 001_B **CH1**: Channel 1 within TIM selected for CAPINB input of GPT12x 010_B **CH2**: Channel 2 within TIM selected for CAPINB input of GPT12x 011_B **CH3**: Channel 3 within TIM selected for CAPINB input of GPT12x 100_B **CH4**: Channel 4 within TIM selected for CAPINB input of GPT12x 101_B **CH5**: Channel 5 within TIM selected for CAPINB input of GPT12x 110_B **CH6**: Channel 6 within TIM selected for CAPINB input of GPT12x 111_B **CH7**: Channel 7 within TIM selected for CAPINB input of GPT12x
EN_CAPB	31	rw	**Enable connection to CAPINB input of GPT12x** 0_B **0**: Disable the connection of eGTM to GPT12x_CAPINB 1_B **1**: Enable the connection of eGTM to GPT12x_CAPINB
0	3, 11, 19, 27	r	**Reserved, do not use** Read as 0, shall be written with 0.

44.5.223 eGTM to QSPIn output selection register. n<8 , if interface exists. (n=0-7)

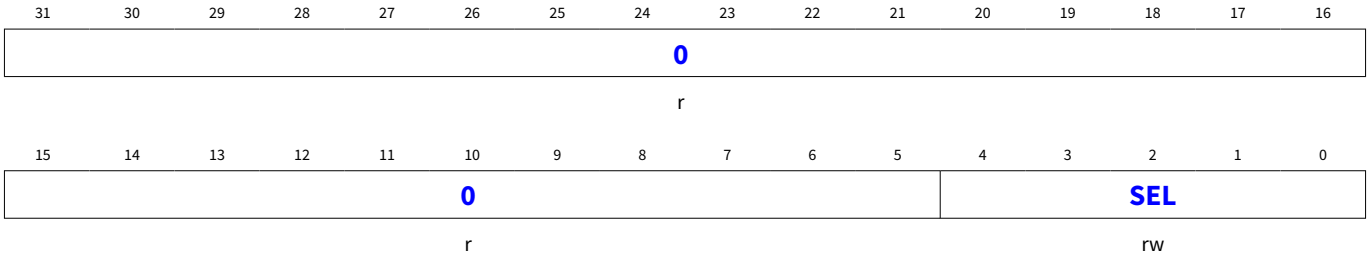
The bit field in this register defines which TOM/ATOM channel output is used as QSPIn trigger.

QSPIn_OUT (n=0-7)

Offset address: 061580_H+n*4

eGTM to QSPIn output selection register. n<8 , if interface exists.

Kernel Reset value: $0000\ 0000_H$



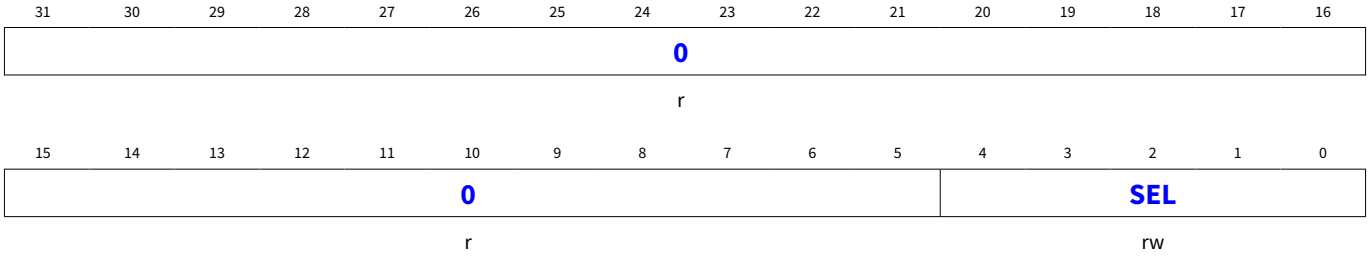
Field	Bits	Type	Description
SEL	4:0	rw	**eGTM to QSPIn trigger output selection**
0	31:5	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.5.224 eGTM to ASCLINn output selection register. n<28 , if interface exists. (n=0-27)

The bit field in this register defines which TOM/ATOM channel output is used as ASCLINn trigger.

ASCLINn_OUT (n=0-27) Offset address: 061600_H+n*4

eGTM to ASCLINn output selection register. n<28 , if interface exists. Kernel Reset value: $0000\ 0000_H$



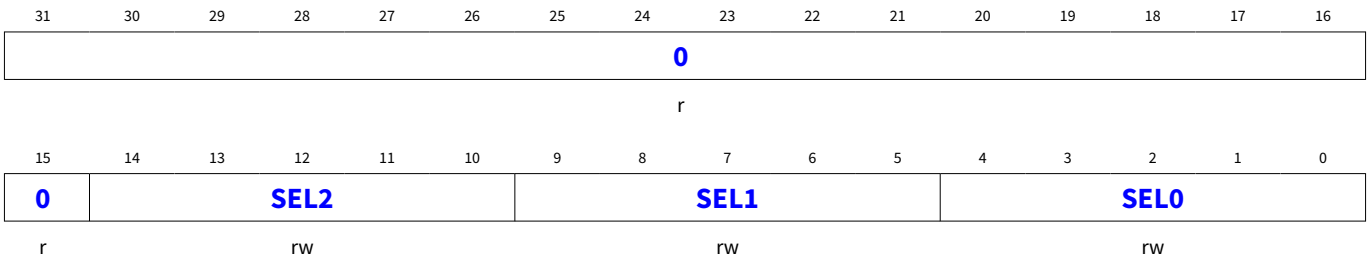
Field	Bits	Type	Description
SEL	4:0	rw	**eGTM to ASCLINn trigger output selection**
0	31:5	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.5.225 eGTM to SENTn trigger x output selection register A (n=0-1)

The bit fields in this register define which TOM/ATOM channel output is used as SENTn trigger x.

SENTn_OUTA (n=0-1) Offset address: 061680_H+n*8

eGTM to SENTn trigger x output selection register A Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
SELx (x=0-2)	5*x+4:5*x	rw	**eGTM to SENTn trigger x output selection**
0	31:15	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.5.226 eGTM to ADC output selection register n (n=0-2)

The bit fields in this register define which TOM/ATOM channel output is used as ADC trigger x.

ADC_OUTn (n=0-2) Offset address: 061700_H+n*4

eGTM to ADC output selection register n Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0			SEL3					0			SEL2				
r			rw					r			rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			SEL1					0			SEL0				
r			rw					r			rw				

Field	Bits	Type	Description
SELx (x=0-1)	8*x+4:8*x	rw	**eGTM to ADC trigger x output selection**
SELx (x=2-3)	8*x+4:8*x	rw	**eGTM to ADC trigger x output selection**
0	31:29, 23:21, 15:13, 7:5	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.5.227 Cluster n DTMAUX input y selection register (y=0-1;n=0-2)

DTMAUXn_INy (n=0-2;y=0-1) Offset address: 061780_H+n*8+y*4

Cluster n DTMAUX input y selection register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								0	SEL5			0	SEL4		
r								r	rw			r	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	SEL3			0	SEL2			0	SEL1			0	SEL0		
r	rw			r	rw			r	rw			r	rw		

Field	Bits	Type	Description
SELx (x=0-5)	4*x+2:4*x	rw	**Input selection for CDTMn_DTMx_AUXINy**
0	23, 19, 15, 11, 7, 3, 31:24	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.5.228 eGTM to SCU output selection register n eGTM_SCU_TRIG(n) (n=0-2)

The bit fields in this register define which TOM/ATOM channel output is used as SCU trigger x.

SCU_OUTn (n=0-2) Offset address: 061800_H+n*4

eGTM to SCU output selection register n eGTM_SCU_TRIG(n) Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0			SEL3					0			SEL2				
r			rw					r			rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			SEL1					0			SEL0				
r			rw					r			rw				

Field	Bits	Type	Description
SELx (x=0-1)	8*x+4:8*x	rw	**eGTM to SCU trigger x output selection eGTM_SCU_TRIG(x)**
SELx (x=2-3)	8*x+4:8*x	rw	**eGTM to SCU trigger x output selection eGTM_SCU_TRIG(x)**
0	15:13, 7:5, 31:29, 23:21	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.5.229 Instance Selection for TIM[n] channel[i] (i=0-7;n=0-2)

TIMn_ADCSELCHi (i=0-7;n=0-2) Offset address: $061B00_H+n*20_H+i*4$

Instance Selection for TIM[n] channel[i] Kernel Reset value: $XXXX\ XXXX_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															
r															

Field	Bits	Type	Description
0	31:0	r	**Reserved, must be written with 0, shall read 0.**

44.5.230 HRPWMz 16:1 Channel MUX select (z=0-2)

HRPWMMUXz_CHSEL (z=0-2) Offset address: 062000_H+z*4

HRPWMz 16:1 Channel MUX select Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CH7				CH6				CH5				CH4			
rw				rw				rw				rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH3				CH2				CH1				CH0			
rw				rw				rw				rw			

Field	Bits	Type	Description
CHq (q=0-7)	4*q+3:4*q	rw	**Channel q select** 0_H CDTM[z]_DTM4_0 is selected 1_H CDTM[z]_DTM4_0_N is selected 2_H CDTM[z]_DTM4_1 is selected 3_H CDTM[z]_DTM4_1_N is selected 4_H CDTM[z]_DTM4_2 is selected 5_H CDTM[z]_DTM4_2_N is selected 6_H CDTM[z]_DTM4_3 is selected 7_H CDTM[z]_DTM4_3_N is selected 8_H CDTM[z]_DTM5_0 is selected 9_H CDTM[z]_DTM5_0_N is selected A_H CDTM[z]_DTM5_1 is selected B_H CDTM[z]_DTM5_1_N is selected C_H CDTM[z]_DTM5_2 is selected D_H CDTM[z]_DTM5_2_N is selected E_H CDTM[z]_DTM5_3 is selected F_H CDTM[z]_DTM5_3_N is selected

44.5.231 HRPWMz module and channel enable register (z=0-2)

HRPWMz_EN (z=0-2) Offset address: $062400_H+z*400_H$

HRPWMz module and channel enable register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CH7	CH6	CH5	CH4	CH3	CH2	CH1	CH0	0							
rw	rw	rw	rw	rw	rw	rw	rw	r							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															MOD
r															rw

Field	Bits	Type	Description
MOD	0	rw	**HRPWMz module enable** 0_B module disabled, all channels bypassed 1_B module enabled
CHy (y=0-7)	y+24	rw	**HRPWMz channel y enable** 0_B Channel bypassed 1_B Channel enabled
0	23:1	r	**Reserved, must be written with 0, shall read 0.**

44.6 Debug information

This section is not applicable for the module being described.

44.7 References

The module is based on GTM4.1, a delivery from Bosch AE.

44.8 eGTM revision history

Reference	Description of change(s)
Date range: 2024-08-17 to 2025-03-27	
Enhanced Generic Timer Module (eGTM)	• Import new Bosch AE specification • Fixed typos, rephrased sentences, updated register descriptions.

44.9 TC4Dx eGTM information

44.9.1 TC4Dx eGTM configuration

eGTM represents the 3 cluster module.

Table 2363 eGTM specific parameters

Parameter	Value
Number of eGTM instances	1 eGTM
TIM sub-block	3 x 8 channels (TIM0-TIM2)
TOM sub-block	3 x 16 channels (TOM0-TOM2)
High resolution on TOM	not existing
ATOM sub-block	3 x 8 channels (ATOM0-ATOM2)
High resolution block connected to analogue block	not existing
DTM/CDTM sub-blocks	12 @TOMs (TOM0-TOM2) 6 @ATOMs (ATOM0-ATOM2)
MCS sub-block	not existing
ARU sub-block	not existing
MAP sub-block	not existing
SPE sub-block	3 (SPE0-SPE2)
PSM sub-block	not existing
DPLL sub-block	not existing. SUB_INCn signals are DPLL signals
TBU sub-block	3 channels (TS0-TS2) TBU channel 3 is only usable with DPLL, therefore the 4th TBU channel exists only in devices with DPLL.
BRC sub-block	not existing
MON sub-block	1
CMP sub-block	1
eGTM Clusters	3 (CCM0-CCM2)
TIO sub-block	No TIO is present in any Infineon device of the TC4x family

Table 2364 eGTM connections in SoC

Connecting to module	Superset Connections with registers for control	eGTM connections with modules connected
ASCLIN	28	28
CAN	24	24
ADC-TMADC	7	4
ADC-FCC	10	0

(table continues...)

Table 2364 (continued) eGTM connections in SoC

Connecting to module	Superset Connections with registers for control	eGTM connections with modules connected
ADC-DSADC	10	0
ADC-CDSP	12	2
ADC-EXMOD	8	0
ERAY (FlexRay)	2	2
GPT12	4	0
SENT	2	2
PSI5	4	2
PSI5S	2	1
LETH/GETH	4/2	4/2
TOUT	294	294
MSC	3	1
QSPI	8	8

44.9.2 TC4Dx eGTM feature list

The device specific registers can be found in the subsection eGTM registers. The family subsection includes all register structures for the device.

The configuration table of the eGTM itself can be found in eGTM configuration.

The cluster frequency is limited to 250 MHz.

44.9.3 TC4Dx eGTM functional description

The functional description is listed in the family part. Please be aware, that no HRPWM analogue block exists, therefore the corresponding feature does not work, even though the eGTM-IP itself is capable of the feature.

44.9.3.1 TOUTSEL and TIMnINSEL(B) Reference Tables

The following table includes the registers available to support mapping between the module and the ports. Dependent on the mapping inside the datasheet, those may see a connection to a port pin or not. In any case the connection mentioned in the datasheet is correct.

Table 2365 TOUTSEL registers in relation to their select bit fields

Registername	SEL5	SEL4	SEL3	SEL2	SEL1	SEL0
TOUTSEL0	TOUT5	TOUT4	TOUT3	TOUT2	TOUT1	TOUT0
TOUTSEL1	TOUT11	TOUT10	TOUT9	TOUT8	TOUT7	TOUT6
TOUTSEL2	TOUT17	TOUT16	TOUT15	TOUT14	TOUT13	TOUT12
TOUTSEL3	TOUT23	TOUT22	TOUT21	TOUT20	TOUT19	TOUT18
TOUTSEL4	TOUT29	TOUT28	TOUT27	TOUT26	TOUT25	TOUT24
TOUTSEL5	TOUT35	TOUT34	TOUT33	TOUT32	TOUT31	TOUT30

(table continues...)

Table 2365 (continued) TOUTSEL registers in relation to their select bit fields

TOUTSEL6	TOUT41	TOUT40	TOUT39	TOUT38	TOUT37	TOUT36
TOUTSEL7	TOUT47	TOUT46	TOUT45	TOUT44	TOUT43	TOUT42
TOUTSEL8	TOUT53	TOUT52	TOUT51	TOUT50	TOUT49	TOUT48
TOUTSEL9	TOUT59	TOUT58	TOUT57	TOUT56	TOUT55	TOUT54
TOUTSEL10	TOUT65	TOUT64	TOUT63	TOUT62	TOUT61	TOUT60
TOUTSEL11	TOUT71	TOUT70	TOUT69	TOUT68	TOUT67	TOUT66
TOUTSEL12	TOUT77	TOUT76	TOUT75	TOUT74	TOUT73	TOUT72
TOUTSEL13	TOUT83	TOUT82	TOUT81	TOUT80	TOUT79	TOUT78
TOUTSEL14	TOUT89	TOUT88	TOUT87	TOUT86	TOUT85	TOUT84
TOUTSEL15	TOUT95	TOUT94	TOUT93	TOUT92	TOUT91	TOUT90
TOUTSEL16	TOUT101	TOUT100	TOUT99	TOUT98	TOUT97	TOUT96
TOUTSEL17	TOUT107	TOUT106	TOUT105	TOUT104	TOUT103	TOUT102
TOUTSEL18	TOUT113	TOUT112	TOUT111	TOUT110	TOUT109	TOUT108
TOUTSEL19	TOUT119	TOUT118	TOUT117	TOUT116	TOUT115	TOUT114
TOUTSEL20	TOUT125	TOUT124	TOUT123	TOUT122	TOUT121	TOUT120
TOUTSEL21	TOUT131	TOUT130	TOUT129	TOUT128	TOUT127	TOUT126
TOUTSEL22	TOUT137	TOUT136	TOUT135	TOUT134	TOUT133	TOUT132
TOUTSEL23	TOUT143	TOUT142	TOUT141	TOUT140	TOUT139	TOUT138
TOUTSEL24	TOUT149	TOUT148	TOUT147	TOUT146	TOUT145	TOUT144
TOUTSEL25	TOUT155	TOUT154	TOUT153	TOUT152	TOUT151	TOUT150
TOUTSEL26	TOUT161	TOUT160	TOUT159	TOUT158	TOUT157	TOUT156
TOUTSEL27	TOUT167	TOUT166	TOUT165	TOUT164	TOUT163	TOUT162
TOUTSEL28	TOUT173	TOUT172	TOUT171	TOUT170	TOUT169	TOUT168
TOUTSEL29	TOUT179	TOUT178	TOUT177	TOUT176	TOUT175	TOUT174
TOUTSEL30	TOUT185	TOUT184	TOUT183	TOUT182	TOUT181	TOUT180
TOUTSEL31	TOUT191	TOUT190	TOUT189	TOUT188	TOUT187	TOUT186
TOUTSEL32	TOUT197	TOUT196	TOUT195	TOUT194	TOUT193	TOUT192
TOUTSEL33	TOUT203	TOUT202	TOUT201	TOUT200	TOUT199	TOUT198
TOUTSEL34	TOUT209	TOUT208	TOUT207	TOUT206	TOUT205	TOUT204
TOUTSEL35	TOUT215	TOUT214	TOUT213	TOUT212	TOUT211	TOUT210
TOUTSEL36	TOUT221	TOUT220	TOUT219	TOUT218	TOUT217	TOUT216
TOUTSEL37	TOUT227	TOUT226	TOUT225	TOUT224	TOUT223	TOUT222
TOUTSEL38	TOUT233	TOUT232	TOUT231	TOUT230	TOUT229	TOUT228
TOUTSEL39	TOUT239	TOUT238	TOUT237	TOUT236	TOUT235	TOUT234

(table continues...)

Table 2365 (continued) TOUTSEL registers in relation to their select bit fields

TOUTSEL40	TOUT245	TOUT244	TOUT243	TOUT242	TOUT241	TOUT240
TOUTSEL41	TOUT251	TOUT250	TOUT249	TOUT248	TOUT247	TOUT246
TOUTSEL42	TOUT257	TOUT256	TOUT255	TOUT254	TOUT253	TOUT252
TOUTSEL43	TOUT263	TOUT262	TOUT261	TOUT260	TOUT259	TOUT258
TOUTSEL44	TOUT269	TOUT268	TOUT267	TOUT266	TOUT265	TOUT264
TOUTSEL45	TOUT275	TOUT274	TOUT273	TOUT272	TOUT271	TOUT270
TOUTSEL46	TOUT281	TOUT280	TOUT279	TOUT278	TOUT277	TOUT276
TOUTSEL47	TOUT287	TOUT286	TOUT285	TOUT284	TOUT283	TOUT282
TOUTSEL48	TOUT293	TOUT292	TOUT291	TOUT290	TOUT289	TOUT288

Table 2366 TIMnINSEL and TIMnINSELB

Registername	Bitfieldnames							
TIM0INSEL / TIM0INSELB	CH7SEL	CH6SEL	CH5SEL	CH4SEL	CH3SEL	CH2SEL	CH1SEL	CH0SEL
TIM1INSEL / TIM1INSELB	CH7SEL	CH6SEL	CH5SEL	CH4SEL	CH3SEL	CH2SEL	CH1SEL	CH0SEL
TIM2INSEL / TIM2INSELB	CH7SEL	CH6SEL	CH5SEL	CH4SEL	CH3SEL	CH2SEL	CH1SEL	CH0SEL

44.9.4 TC4Dx eGTM registers

44.9.4.1 Register address space - EGTM

Table 2367 Registers address space - EGTM

Module	Base address	End address	Note
EGTM	$F9080000_H$	$F90E2FFF_H$	LLI slave interface

44.9.4.2 Register overview - access mode glossary

Table 2368 Register overview - access mode glossary

Keyword	Description
E	Access protection using PROT register EGTM_CTRL_PROTE .
SE	Access protection using PROT register EGTM_PROTSE .

(table continues...)

Table 2368 (continued) Register overview - access mode glossary

Keyword	Description
APU-P	Protection group consisting of registers EGTM_CTRL_ACCEN_WRA , EGTM_CTRL_ACCEN_WRB , EGTM_CTRL_ACCEN_RDA , EGTM_CTRL_ACCEN_RDB , EGTM_CTRL_ACCEN_VM , EGTM_CTRL_ACCEN_PRS .
P	Access protection using *APU-P* registers.
APU-PCLi (i=0-2)	Protection group consisting of registers EGTM_CLi_ACCEN_WRA , EGTM_CLi_ACCEN_WRB , EGTM_CLi_ACCEN_RDA , EGTM_CLi_ACCEN_RDB , EGTM_CLi_ACCEN_VM , EGTM_CLi_ACCEN_PRS .
PCLi	Access protection using *APU-PCLi* registers.
APU-PGTM	Protection group consisting of registers EGTM_CTRLGTM_ACCEN_WRA , EGTM_CTRLGTM_ACCEN_WRB , EGTM_CTRLGTM_ACCEN_RDA , EGTM_CTRLGTM_ACCEN_RDB , EGTM_CTRLGTM_ACCEN_VM , EGTM_CTRLGTM_ACCEN_PRS .
PGTM	Access protection using *APU-PGTM* registers.
APU-PSAFE	Protection group consisting of registers EGTM_SAFEGTM_ACCEN_WRA , EGTM_SAFEGTM_ACCEN_WRB , EGTM_SAFEGTM_ACCEN_RDA , EGTM_SAFEGTM_ACCEN_RDB , EGTM_SAFEGTM_ACCEN_VM , EGTM_SAFEGTM_ACCEN_PRS .
PSAFE	Access protection using *APU-PSAFE* registers.
APU-PWRAP	Protection group consisting of registers EGTM_WRAP_ACCEN_WRA , EGTM_WRAP_ACCEN_WRB , EGTM_WRAP_ACCEN_RDA , EGTM_WRAP_ACCEN_RDB , EGTM_WRAP_ACCEN_VM , EGTM_WRAP_ACCEN_PRS .
PWRAP	Access protection using *APU-PWRAP* registers.
32	Access only when using 32-bit width.
BE	Always returns a Bus Error.
U	No access restrictions.
PROT	Access restrictions as defined in the PROT register access rules.
nBE	Indicates that no Bus Error is generated when accessing this address range, even though it is either an access to an undefined address or the access does not follow the given rules.

44.9.4.3 Register overview - EGTM domain CPU (ascending offset address)

Table 2369 Register overview - EGTM domain CPU (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
EGTM_CLSi_ARCH_REV (i=0)	eGTM version control register	000000_H +i*20000 H	PGTM, 32	BE	Kernel Reset	7635
EGTM_CLSi_ARCH_RST (i=0)	eGTM global reset register	000004_H +i*20000 H	PGTM, 32	PGTM, 32	Kernel Reset	7635

(table continues...)

Table 2369 (continued) Register overview - EGTM domain CPU (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
EGTM_CLSi_ARCH_CTRL (i=0)	eGTM global control register	000008_H +i*20000 H	PGTM, 32	PGTM, 32	Kernel Reset	7636
EGTM_CLSi_ARCH_CFG (i=0)	eGTM configuration register	$00000C_H$ +i*20000 H	PGTM, 32	PGTM, 32	Kernel Reset	7637
EGTM_CLSi_ARCH_AEI_ADDR_XPT (i=0)	eGTM AEI timeout exception address register	000010_H +i*20000 H	PGTM, 32	BE	7638	7638
EGTM_CLSi_ARCH_AEI_STA_XPT (i=0)	eGTM AEI non-zero status register	000014_H +i*20000 H	PGTM, 32	BE	7639	7639
EGTM_CLSi_ARCH_IRQ_NOTIFY (i=0)	eGTM Interrupt notification register	000018_H +i*20000 H	PGTM, 32	PGTM, 32	7640	7640
EGTM_CLSi_ARCH_IRQ_EN (i=0)	eGTM interrupt enable register	$00001C_H$ +i*20000 H	PGTM, 32	PGTM, 32	Kernel Reset	7643
EGTM_CLSi_ARCH_EIRQ_EN (i=0)	eGTM error interrupt enable register	000020_H +i*20000 H	PGTM, 32	PGTM, 32	Kernel Reset	7644
EGTM_CLSi_ARCH_IRQ_FORCINT (i=0)	eGTM Software interrupt generation register	000024_H +i*20000 H	PGTM, 32	PGTM, 32	Kernel Reset	7645
EGTM_CLSi_ARCH_IRQ_MODE (i=0)	eGTM toplevel interrupts mode selection	000028_H +i*20000 H	PGTM, 32	PGTM, 32	Kernel Reset	7647
EGTM_CLSi_ARCH_CLK_CFG (i=0)	eGTM Cluster Clock Configuration	$00002C_H$ +i*20000 H	PGTM, 32	PGTM, 32	7648	7648
EGTM_CLSi_AEI_BRIDGE_MODE (i=0)	eGTM AEI bridge mode register	000040_H +i*20000 H	PGTM, 32	PGTM, 32	7649	7649
EGTM_CLSi_AEI_BRIDGE_PTR1 (i=0)	eGTM AEI bridge pointer 1 register	000044_H +i*20000 H	PGTM, 32	BE	7652	7652
EGTM_CLSi_AEI_BRIDGE_PTR2 (i=0)	eGTM AEI bridge pointer 2 register	000048_H +i*20000 H	PGTM, 32	BE	7653	7653

(table continues...)

Table 2369 (continued) Register overview - EGTM domain CPU (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
EGTM_CLSi_CMU_CLK_EN (i=0)	CMU clock enable	000080_H +i*20000 H	PGTM, 32	PGTM, 32	7653	7653
EGTM_CLSi_CMU_GCLK_NUM (i=0)	The numerator for CMU global clock resolution generator	000084_H +i*20000 H	PGTM, 32	PGTM, 32	Kernel Reset	7654
EGTM_CLSi_CMU_GCLK_DEN (i=0)	The denominator for CMU global clock resolution generator	000088_H +i*20000 H	PGTM, 32	PGTM, 32	Kernel Reset	7655
EGTM_CLSi_CMU_CLKx_CTRL (i=0;x=0-5)	CMU control for clock resolution generator [x]	$00008C_H$ +i*20000 H+x*4	PGTM, 32	PGTM, 32	Kernel Reset	7656
EGTM_CLSi_CMU_CLK_6_CTRL (i=0)	CMU control for clock resolution generator 6	$0000A4_H$ +i*20000 H	PGTM, 32	PGTM, 32	Kernel Reset	7657
EGTM_CLSi_CMU_CLK_7_CTRL (i=0)	CMU control for clock resolution generator 7	$0000A8_H$ +i*20000 H	PGTM, 32	PGTM, 32	Kernel Reset	7657
EGTM_CLSi_CMU_ECLKz_NUM (i=0;z=0-2)	The numerator for the external clock resolution generator [z]	$0000AC_H$ +i*20000 H+z*8	PGTM, 32	PGTM, 32	Kernel Reset	7658
EGTM_CLSi_CMU_ECLKz_DEN (i=0;z=0-2)	The denominator for the external clock resolution generator [z]	$0000B0_H$ +i*20000 H+z*8	PGTM, 32	PGTM, 32	Kernel Reset	7659
EGTM_CLSi_CMU_FXCLK_CTRL (i=0)	CMU control for selection of FCR subblock input	$0000C4_H$ +i*20000 H	PGTM, 32	PGTM, 32	Kernel Reset	7660
EGTM_CLSi_CMU_CLK_CTRL (i=0)	CMU control for clock resolution generator	$0000CC_H$ +i*20000 H	PGTM, 32	PGTM, 32	Kernel Reset	7661
EGTM_CLSi_TBU_CHEN (i=0)	TBU global channel enable	000100_H +i*20000 H	PGTM, 32	PGTM, 32	7662	7662
EGTM_CLSi_TBU_CH0_CTRL (i=0)	TBU channel 0 control	000104_H +i*20000 H	PGTM, 32	PGTM, 32	Kernel Reset	7663
EGTM_CLSi_TBU_CH0_BASE (i=0)	TBU channel 0 base	000108_H +i*20000 H	PGTM, 32	PGTM, 32	Kernel Reset	7664
EGTM_CLSi_TBU_CH1_CTRL (i=0)	TBU channel 1 control	$00010C_H$ +i*20000 H	PGTM, 32	PGTM, 32	Kernel Reset	7664

(table continues...)

Table 2369 (continued) Register overview - EGTM domain CPU (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
EGTM_CLSi_TBU_CHx_BASE (i=0;x=1-2)	TBU channel [x] base	000110_H +i*20000 $_H$+(x-1)*8	PGTM, 32	PGTM, 32	Kernel Reset	7665
EGTM_CLSi_TBU_CH2_CTRL (i=0)	TBU channel 2 control	000114_H +i*20000 $_H$	PGTM, 32	PGTM, 32	Kernel Reset	7666
EGTM_CLSi_ICM_IRQG_0 (i=0)	ICM_IRQG_0	000400_H +i*20000 $_H$	PGTM, 32	BE	7667	7667
EGTM_CLSi_ICM_IRQG_2 (i=0)	ICM Interrupt group register covering TIM0, TIM1, TIM2, TIM3	000408_H +i*20000 $_H$	PGTM, 32	BE	7668	7668
EGTM_CLSi_ICM_IRQG_MEI (i=0)	ICM Interrupt group register for module error interrupt information	000430_H +i*20000 $_H$	PGTM, 32	BE	7669	7669
EGTM_CLSi_ICM_IRQG_CEI1 (i=0)	ICM Interrupt group register 1 for channel error interrupt information	000438_H +i*20000 $_H$	PGTM, 32	BE	7671	7671
EGTM_CLSi_ICM_IRQG_SPE_CEI (i=0)	ICM Interrupt group SPE for module Error Interrupt information	$0004B4_H$ +i*20000 $_H$	PGTM, 32	BE	Kernel Reset	7672
EGTM_CLSi_ICM_IRQG_CLSg_MEI (g=0;i=0)	ICM_IRQG_CLS_[g]_MEI	000510_H +i*20000 $_H$	PGTM, 32	BE	7672	7672
EGTM_CLSi_ICM_IRQG_SPE_CI (i=0)	ICM Interrupt group SPE for module Interrupt information	000570_H +i*20000 $_H$	PGTM, 32	BE	Kernel Reset	7674
EGTM_CLSi_ICM_IRQG_ATOMg_CI (g=0;i=0)	ICM_IRQG_ATOM_[g]_CI	000590_H +i*20000 $_H$	PGTM, 32	BE	7675	7675
EGTM_CLSi_ICM_IRQG_TOMg_CI (g=0-1;i=0)	ICM_IRQG_TOM_[g]_CI	$0005A0_H$ +i*20000 $_H$+g*4	PGTM, 32	BE	7676	7676
EGTM_CLSi_MON_STATUS (i=1)	MON status register	000680_H +i*20000 $_H$	PSAFE, 32	PSAFE, 32	7678	7678
EGTM_CLSi_CMP_EN (i=1)	CMP comparator enable register	$0006C0_H$ +i*20000 $_H$	PSAFE, 32	PSAFE, 32	7679	7679
EGTM_CLSi_CMP_IRQ_NOTIFY (i=1)	CMP event notification register	$0006C4_H$ +i*20000 $_H$	PSAFE, 32	PSAFE, 32	7680	7680

(table continues...)

Table 2369 (continued) Register overview - EGTM domain CPU (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
EGTM_CLSi_CMP_IRQ_EN (i=1)	CMP interrupt enable register	$0006C8_H$ +i*20000 $_H$	PSAFE, 32	PSAFE, 32	7681	7681
EGTM_CLSi_CMP_IRQ_FORCINT (i=1)	CMP interrupt force register	$0006CC_H$ +i*20000 $_H$	PSAFE, 32	PSAFE, 32	7682	7682
EGTM_CLSi_CMP_IRQ_MODE (i=1)	CMP interrupt mode configuration register	$0006D0_H$ +i*20000 $_H$	PSAFE, 32	PSAFE, 32	Kernel Reset	7683
EGTM_CLSi_CMP_EIRQ_EN (i=1)	CMP error interrupt enable register	$0006D4_H$ +i*20000 $_H$	PSAFE, 32	PSAFE, 32	7683	7683
EGTM_CLSi_TIM_CHx_GPR0 (i=0-2;x=0-7)	TIM[i] channel [x] general-purpose 0 register	000800_H +i*20000 $_H$+x*80_H	PCLi, 32	PCLi, 32	7684	7684
EGTM_CLSi_TIM_CHx_GPR1 (i=0-2;x=0-7)	TIM[i] channel [x] general-purpose 1 register	000804_H +i*20000 $_H$+x*80_H	PCLi, 32	PCLi, 32	7685	7685
EGTM_CLSi_TIM_CHx_CNT (i=0-2;x=0-7)	TIM[i] channel [x] SMU counter register	000808_H +i*20000 $_H$+x*80_H	PCLi, 32	BE	7686	7686
EGTM_CLSi_TIM_CHx_ECNT (i=0-2;x=0-7)	TIM[i] channel [x] SMU edge counter register	$00080C_H$ +i*20000 $_H$+x*80_H	PCLi, 32	BE	7687	7687
EGTM_CLSi_TIM_CHx_CNTS (i=0-2;x=0-7)	TIM[i] channel [x] SMU shadow counter register	000810_H +i*20000 $_H$+x*80_H	PCLi, 32	PCLi, 32	7688	7688
EGTM_CLSi_TIM_CHx_TDUC (i=0-2;x=0-7)	TIM[i]_CH[x]_TDUC	000814_H +i*20000 $_H$+x*80_H	PCLi, 32	PCLi, 32	7689	7689
EGTM_CLSi_TIM_CHx_TDUV (i=0-2;x=0-7)	TIM[i]_CH[x]_TDUV	000818_H +i*20000 $_H$+x*80_H	PCLi, 32	PCLi, 32	7690	7690
EGTM_CLSi_TIM_CHx_FLT_RE (i=0-2;x=0-7)	TIM[i] channel [x] filter parameter 0 register	$00081C_H$ +i*20000 $_H$+x*80_H	PCLi, 32	PCLi, 32	7691	7691
EGTM_CLSi_TIM_CHx_FLT_FE (i=0-2;x=0-7)	TIM[i] channel [x] filter parameter 1 register	000820_H +i*20000 $_H$+x*80_H	PCLi, 32	PCLi, 32	7692	7692

(table continues...)

Table 2369 (continued) Register overview - EGTM domain CPU (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
EGTM_CLSi_TIM_CHx_CTRL (i=0;x=0-7) (i=1-2;x=0-7)	TIM[i] channel [x] control register	000824_H +i*20000_H+x*80_H	PCLi, 32	PCLi, 32	7692	7692
EGTM_CLSi_TIM_CHx_ECTRL (i=0-2;x=0-7)	TIM[i] channel [x] extended control register	000828_H +i*20000_H+x*80_H	PCLi, 32	PCLi, 32	7707	7707
EGTM_CLSi_TIM_CHx_IRQ_NOTIFY (i=0-2;x=0-7)	TIM[i] channel [x] interrupt notification register	$00082C_H$ +i*20000_H+x*80_H	PCLi, 32	PCLi, 32	7717	7717
EGTM_CLSi_TIM_CHx_IRQ_EN (i=0-2;x=0-7)	TIM[i] channel [x] interrupt enable register	000830_H +i*20000_H+x*80_H	PCLi, 32	PCLi, 32	7719	7719
EGTM_CLSi_TIM_CHx_IRQ_FORCINT (i=0-2;x=0-7)	TIM[i] channel [x] force interrupt register	000834_H +i*20000_H+x*80_H	PCLi, 32	PCLi, 32	7720	7720
EGTM_CLSi_TIM_CHx_IRQ_MODE (i=0-2;x=0-7)	TIM[i] channel [x] interrupt mode configuration register	000838_H +i*20000_H+x*80_H	PCLi, 32	PCLi, 32	7722	7722
EGTM_CLSi_TIM_CHx_EIRQ_EN (i=0-2;x=0-7)	TIM[i] channel [x] error interrupt enable register	$00083C_H$ +i*20000_H+x*80_H	PCLi, 32	PCLi, 32	7723	7723
EGTM_CLSi_TIM_INP_VAL (i=0-2)	TIM[i] input value observation register	$000C00_H$ +i*20000_H	PCLi, 32	BE	Kernel Reset	7724
EGTM_CLSi_TIM_IN_SRC (i=0-2)	TIM[i] AUX IN source selection register	$000C04_H$ +i*20000_H	PCLi, 32	PCLi, 32	Kernel Reset	7724
EGTM_CLSi_TIM_RST (i=0-2)	TIM[i] global software reset register	$000C08_H$ +i*20000_H	PCLi, 32	PCLi, 32	Kernel Reset	7726
EGTM_CLSi_TOM_CHx_CTRL (i=0-2;x=0-15)	TOM[i] channel [x] control register	001000_H +i*20000_H+x*40_H	PCLi, 32	PCLi, 32	7727	7727
EGTM_CLSi_TOM_CHx_SR0 (i=0-2;x=0-15)	TOM[i] channel [x] CCU0 compare shadow register	001004_H +i*20000_H+x*40_H	PCLi, 32	PCLi, 32	7759	7759
EGTM_CLSi_TOM_CHx_SR1 (i=0-2;x=0-15)	TOM[i] channel [x] CCU1 compare shadow register	001008_H +i*20000_H+x*40_H	PCLi, 32	PCLi, 32	7759	7759

(table continues...)

Table 2369 (continued) Register overview - EGTM domain CPU (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
EGTM_CLSi_TOM_CHx_CM0 (i=0-2;x=0-15)	TOM[i] channel [x] CCU0 compare register	$00100C_H$ +i*20000_H+x*40_H	PCLi, 32	PCLi, 32	7760	7760
EGTM_CLSi_TOM_CHx_CM1 (i=0-2;x=0-15)	TOM[i] channel [x] CCU1 compare register	001010_H +i*20000_H+x*40_H	PCLi, 32	PCLi, 32	7760	7760
EGTM_CLSi_TOM_CHx_CN0 (i=0-2;x=0-15)	TOM[i] channel [x] CCU0 counter	001014_H +i*20000_H+x*40_H	PCLi, 32	PCLi, 32	7761	7761
EGTM_CLSi_TOM_CHx_STAT (i=0-2;x=0-15)	TOM[i] channel [x] status register	001018_H +i*20000_H+x*40_H	PCLi, 32	PCLi, 32	7762	7762
EGTM_CLSi_TOM_CHx_IRQ_NOTIFY (i=0-2;x=0-15)	TOM[i] channel [x] interrupt notification register	$00101C_H$ +i*20000_H+x*40_H	PCLi, 32	PCLi, 32	7762	7762
EGTM_CLSi_TOM_CHx_IRQ_EN (i=0-2;x=0-15)	TOM[i] channel [x] interrupt enable register	001020_H +i*20000_H+x*40_H	PCLi, 32	PCLi, 32	7764	7764
EGTM_CLSi_TOM_CHx_IRQ_FORCINT (i=0-2;x=0-15)	TOM[i] channel [x] force interrupt register	001024_H +i*20000_H+x*40_H	PCLi, 32	PCLi, 32	7765	7765
EGTM_CLSi_TOM_CHx_IRQ_MODE (i=0-2;x=0-15)	TOM[i] channel [x] interrupt mode register	001028_H +i*20000_H+x*40_H	PCLi, 32	PCLi, 32	7766	7766
EGTM_CLSi_TOM_CHx_CTRL_SR (i=0-2;x=0-15)	TOM[i] channel [x] control shadow register	001030_H +i*20000_H+x*40_H	PCLi, 32	PCLi, 32	7767	7767
EGTM_CLSi_TOM_TGCg_GLB_CTRL (g=0-1;i=0-2)	TOM[i] TGC [g] global control register	001430_H +i*20000_H+g*80_H	PCLi, 32	PCLi, 32	Kernel Reset	7768
EGTM_CLSi_TOM_TGCg_ACT_TB (g=0-1;i=0-2)	TOM[i] TGC [g] action time base register	001434_H +i*20000_H+g*80_H	PCLi, 32	PCLi, 32	Kernel Reset	7769
EGTM_CLSi_TOM_TGCg_FUPD_CTRL (g=0-1;i=0-2)	TOM[i] TGC [g] force update control register	001438_H +i*20000_H+g*80_H	PCLi, 32	PCLi, 32	Kernel Reset	7770

(table continues...)

Table 2369 (continued) Register overview - EGTM domain CPU (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
EGTM_CLSi_TOM_TGCg_INT_TRIG (g=0-1;i=0-2)	TOM[i] TGC [g] internal trigger control register	$00143C_H$ +i*20000_H+g*80_H	PCLi, 32	PCLi, 32	Kernel Reset	7771
EGTM_CLSi_TOM_TGCg_ENDIS_CTRL (g=0-1;i=0-2)	TOM[i] TGC [g] enable/ disable control register	001470_H +i*20000_H+g*80_H	PCLi, 32	PCLi, 32	Kernel Reset	7772
EGTM_CLSi_TOM_TGCg_ENDIS_STAT (g=0-1;i=0-2)	TOM[i] TGC [g] enable/ disable status register	001474_H +i*20000_H+g*80_H	PCLi, 32	PCLi, 32	Kernel Reset	7773
EGTM_CLSi_TOM_TGCg_OUTEN_CTRL (g=0-1;i=0-2)	TOM[i] TGC [g] output enable control register	001478_H +i*20000_H+g*80_H	PCLi, 32	PCLi, 32	Kernel Reset	7774
EGTM_CLSi_TOM_TGCg_OUTEN_STAT (g=0-1;i=0-2)	TOM[i] TGC [g] output enable status register	$00147C_H$ +i*20000_H+g*80_H	PCLi, 32	PCLi, 32	Kernel Reset	7774
EGTM_CLSi_ATOM_CHx_CTRL (i=0-2;x=0-7)	ATOM[i] channel [x] control register	001804_H +i*20000_H+x*80_H	PCLi, 32	PCLi, 32	7775	7775
EGTM_CLSi_ATOM_CHx_CTRL_SOMI (i=0-2;x=0-7)	ATOM[i] channel [x] control register	001804_H +i*20000_H+x*80_H	PCLi, 32	PCLi, 32	7777	7777
EGTM_CLSi_ATOM_CHx_CTRL_SOMC (i=0-2;x=0-7)	ATOM[i] channel [x] control register	001804_H +i*20000_H+x*80_H	PCLi, 32	PCLi, 32	7781	7781
EGTM_CLSi_ATOM_CHx_CTRL_SOMP (i=0-2;x=0-7)	ATOM[i] channel [x] control register	001804_H +i*20000_H+x*80_H	PCLi, 32	PCLi, 32	7788	7788
EGTM_CLSi_ATOM_CHx_CTRL_SOMS (i=0-2;x=0-7)	ATOM[i] channel [x] control register	001804_H +i*20000_H+x*80_H	PCLi, 32	PCLi, 32	7797	7797
EGTM_CLSi_ATOM_CHx_CTRL_SOMB (i=0-2;x=0-7)	ATOM[i] channel [x] control register	001804_H +i*20000_H+x*80_H	PCLi, 32	PCLi, 32	7802	7802

(table continues...)

Table 2369 (continued) Register overview - EGTM domain CPU (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
EGTM_CLSi_ATOM_CHx_SR0 (i=0-2;x=0-7)	ATOM[i] channel [x] CCU0 compare shadow register	001808_H +i*20000_H+x*80_H	PCLi, 32	PCLi, 32	7808	7808
EGTM_CLSi_ATOM_CHx_SR1 (i=0-2;x=0-7)	ATOM[i] channel [x] CCU1 compare shadow register	$00180C_H$ +i*20000_H+x*80_H	PCLi, 32	PCLi, 32	7809	7809
EGTM_CLSi_ATOM_CHx_CM0 (i=0-2;x=0-7)	ATOM[i] channel [x] CCU0 compare register	001810_H +i*20000_H+x*80_H	PCLi, 32	PCLi, 32	7810	7810
EGTM_CLSi_ATOM_CHx_CM1 (i=0-2;x=0-7)	ATOM[i] channel [x] CCU1 compare register	001814_H +i*20000_H+x*80_H	PCLi, 32	PCLi, 32	7810	7810
EGTM_CLSi_ATOM_CHx_CN0 (i=0-2;x=0-7)	ATOM[i] channel [x] CCU0 counter register	001818_H +i*20000_H+x*80_H	PCLi, 32	PCLi, 32	7811	7811
EGTM_CLSi_ATOM_CHx_STAT (i=0-2;x=0-7)	ATOM[i] channel [x] status register	$00181C_H$ +i*20000_H+x*80_H	PCLi, 32	PCLi, 32	7811	7811
EGTM_CLSi_ATOM_CHx_IRQ_NOTIFY (i=0-2;x=0-7)	ATOM[i] channel [x] interrupt notification register	001820_H +i*20000_H+x*80_H	PCLi, 32	PCLi, 32	7815	7815
EGTM_CLSi_ATOM_CHx_IRQ_EN (i=0-2;x=0-7)	ATOM[i] channel [x] interrupt enable register	001824_H +i*20000_H+x*80_H	PCLi, 32	PCLi, 32	7816	7816
EGTM_CLSi_ATOM_CHx_IRQ_FORCINT (i=0-2;x=0-7)	ATOM[i] channel [x] software interrupt generation	001828_H +i*20000_H+x*80_H	PCLi, 32	PCLi, 32	7817	7817
EGTM_CLSi_ATOM_CHx_IRQ_MODE (i=0-2;x=0-7)	ATOM[i] channel [x] interrupt mode configuration register	$00182C_H$ +i*20000_H+x*80_H	PCLi, 32	PCLi, 32	7818	7818
EGTM_CLSi_ATOM_CHx_CTRL2 (i=0-2;x=0-7)	ATOM[i] channel [x] control2 register	001830_H +i*20000_H+x*80_H	PCLi, 32	PCLi, 32	7818	7818
EGTM_CLSi_ATOM_CHx_CTRL_SR (i=0-2;x=0-7)	ATOM[i] channel [x] control shadow register	001834_H +i*20000_H+x*80_H	PCLi, 32	PCLi, 32	7819	7819

(table continues...)

Table 2369 (continued) Register overview - EGTM domain CPU (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
EGTM_CLSi_ATOM_AGC_GLB_CTRL (i=0-2)	ATOM[i] AGC global control register	$001C40_H$ +i*20000 H	PCLi, 32	PCLi, 32	Kernel Reset	7820
EGTM_CLSi_ATOM_AGC_ENDIS_CTRL (i=0-2)	ATOM[i] AGC enable/disable control register	$001C44_H$ +i*20000 H	PCLi, 32	PCLi, 32	Kernel Reset	7822
EGTM_CLSi_ATOM_AGC_ENDIS_STAT (i=0-2)	ATOM[i] AGC enable/disable status register	$001C48_H$ +i*20000 H	PCLi, 32	PCLi, 32	Kernel Reset	7823
EGTM_CLSi_ATOM_AGC_ACT_TB (i=0-2)	ATOM[i] AGC action time base register	$001C4C_H$ +i*20000 H	PCLi, 32	PCLi, 32	Kernel Reset	7824
EGTM_CLSi_ATOM_AGC_OUTEN_CTRL (i=0-2)	ATOM[i] AGC output enable control register	$001C50_H$ +i*20000 H	PCLi, 32	PCLi, 32	Kernel Reset	7825
EGTM_CLSi_ATOM_AGC_OUTEN_STAT (i=0-2)	ATOM[i] AGC output enable status register	$001C54_H$ +i*20000 H	PCLi, 32	PCLi, 32	Kernel Reset	7826
EGTM_CLSi_ATOM_AGC_FUPD_CTRL (i=0-2)	ATOM[i] AGC force update control register	$001C58_H$ +i*20000 H	PCLi, 32	PCLi, 32	Kernel Reset	7827
EGTM_CLSi_ATOM_AGC_INT_TRIG (i=0-2)	ATOM[i] AGC internal trigger control register	$001C5C_H$ +i*20000 H	PCLi, 32	PCLi, 32	Kernel Reset	7828
EGTM_CLSi_CCM_HW_CONF2 (i=0-2)	CCM[i] 2. hardware configuration register	$0041D4_H$ +i*20000 H	PCLi, 32	BE	Kernel Reset	7829
EGTM_CLSi_CCM_HW_CONF (i=0-2)	CCM[i] hardware configuration register	$0041DC_H$ +i*20000 H	PCLi, 32	BE	Kernel Reset	7830
EGTM_CLSi_CCM_TIM_AUX_IN_SRC (i=0-2)	CCM[i] TIM AUX input source register	$0041E0_H$ +i*20000 H	PCLi, 32	PCLi, 32	Kernel Reset	7832

(table continues...)

Table 2369 (continued) Register overview - EGTM domain CPU (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
EGTM_CLSi_CCM_TOM_OUT (i=0-2)	CCM[i] TOM output register	$0041E8_H$ +i*20000 H	PCLi, 32	BE	Kernel Reset	7835
EGTM_CLSi_CCM_ATOM_OUT (i=0-2)	CCM[i] ATOM output register	$0041EC_H$ +i*20000 H	PCLi, 32	BE	7836	7836
EGTM_CLSi_CCM_CMU_CLK_CFG (i=0-2)	CCM[i] CMU clock configuration register	$0041F0_H$ +i*20000 H	PCLi, 32	PCLi, 32	Kernel Reset	7838
EGTM_CLSi_CCM_CMU_FXCLK_CFG (i=0-2)	CCM[i] CMU fixed clock configuration register	$0041F4_H$ +i*20000 H	PCLi, 32	PCLi, 32	Kernel Reset	7839
EGTM_CLSi_CCM_CFG (i=0-2)	CCM[i] configuration register	$0041F8_H$ +i*20000 H	PCLi, 32	PCLi, 32	7839	7839
EGTM_CLSi_CCM_PROT (i=0-2)	CCM[i] protection register	$0041FC_H$ +i*20000 H	PCLi, 32	PCLi, 32	Kernel Reset	7842
EGTM_CLSi_CDTM_DTMd_CTRL (d=0-5;i=0-2)	CDTM[i]_DTM[d] global configuration and control register	004400_H +i*20000 $_H$+d*40_H	PCLi, 32	PCLi, 32	Kernel Reset	7843
EGTM_CLSi_CDTM_DTMd_CH_CTRL1 (d=0-5;i=0-2)	CDTM[i]_DTM[d] channel control register 1	004404_H +i*20000 $_H$+d*40_H	PCLi, 32	PCLi, 32	Kernel Reset	7847
EGTM_CLSi_CDTM_DTMd_CH_CTRL2 (d=0-5;i=0-2)	CDTM[i]_DTM[d] channel control register 2	004408_H +i*20000 $_H$+d*40_H	PCLi, 32	PCLi, 32	Kernel Reset	7855
EGTM_CLSi_CDTM_DTMd_CH_CTRL2_SR (d=0-5;i=0-2)	CDTM[i] DTM[d] channel control register 2 shadow	$00440C_H$ +i*20000 $_H$+d*40_H	PCLi, 32	PCLi, 32	Kernel Reset	7858
EGTM_CLSi_CDTM_DTMd_PS_CTRL (d=0-5;i=0-2)	CDTM[i]_DTM[d] phase shift unit configuration and control register	004410_H +i*20000 $_H$+d*40_H	PCLi, 32	PCLi, 32	Kernel Reset	7861

(table continues...)

Table 2369 (continued) Register overview - EGTM domain CPU (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
EGTM_CLSi_CDTM_DTMd_CH_DTVx (d=0-3;i=0-2;x=0-3) (d=4-5;i=0-2;x=0-3)	CDTM[i]_DTM[d] channel [x] dead-time reload values	004414_H +i*20000 $_H$+d*40_H +x*4	PCLi, 32	PCLi, 32	Kernel Reset	7862
EGTM_CLSi_CDTM_DTMd_CH_SR (d=0-5;i=0-2)	CDTM[i]_DTM[d] channel shadow register	004424_H +i*20000 $_H$+d*40_H	PCLi, 32	PCLi, 32	Kernel Reset	7863
EGTM_CLSi_CDTM_DTMd_CH_CTRL3 (d=0-5;i=0-2)	CDTM[i]_DTM[d] channel control register 3	004428_H +i*20000 $_H$+d*40_H	PCLi, 32	PCLi, 32	Kernel Reset	7864
EGTM_CLSi_CDTM_DTMd_CTRL2 (d=0-5;i=0-2)	CDTM[i]_DTM[d] global configuration and control register 2	$00442C_H$ +i*20000 $_H$+d*40_H	PCLi, 32	PCLi, 32	Kernel Reset	7866
EGTM_CLSi_CDTM_DTMd_CH_DTV_SRx (d=0-5;i=0-2;x=0-3)	CDTM[i]_DTM[d] channel [x] dead-time shadow values	004430_H +i*20000 $_H$+d*40_H +x*4	PCLi, 32	PCLi, 32	Kernel Reset	7874
EGTM_CLSi_SPE_CTRL_STAT (i=0-2)	SPE[i] control status register	$004C00_H$ +i*20000 $_H$	PCLi, 32	PCLi, 32	Kernel Reset	7875
EGTM_CLSi_SPE_PAT (i=0-2)	SPE[i] input pattern definition register	$004C04_H$ +i*20000 $_H$	PCLi, 32	PCLi, 32	Kernel Reset	7882
EGTM_CLSi_SPE_OUT_PATp (i=0-2;p=0-7)	SPE[i] output definition register [p]	$004C08_H$ +i*20000 $_H$+p*4	PCLi, 32	PCLi, 32	Kernel Reset	7882
EGTM_CLSi_SPE_OUT_CTRL (i=0-2)	SPE[i] output control register	$004C28_H$ +i*20000 $_H$	PCLi, 32	PCLi, 32	Kernel Reset	7883
EGTM_CLSi_SPE_IRQ_NOTIFY (i=0-2)	SPE[i] interrupt notification register	$004C2C_H$ +i*20000 $_H$	PCLi, 32	PCLi, 32	Kernel Reset	7883
EGTM_CLSi_SPE_IRQ_EN (i=0-2)	SPE[i] interrupt enable register	$004C30_H$ +i*20000 $_H$	PCLi, 32	PCLi, 32	Kernel Reset	7885
EGTM_CLSi_SPE_IRQ_FORCINT (i=0-2)	SPE[i] Interrupt Generation By Software	$004C34_H$ +i*20000 $_H$	PCLi, 32	PCLi, 32	Kernel Reset	7886

(table continues...)

Table 2369 (continued) Register overview - EGTM domain CPU (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
EGTM_CLSi_SPE_IRQ_MODE (i=0-2)	SPE[i] interrupt mode configuration register	$004C38_H$ +i*20000 H	PCLi, 32	PCLi, 32	Kernel Reset	7887
EGTM_CLSi_SPE_EIRQ_EN (i=0-2)	SPE[i] error interrupt enable register	$004C3C_H$ +i*20000 H	PCLi, 32	PCLi, 32	Kernel Reset	7888
EGTM_CLSi_SPE_REV_CNT (i=0-2)	SPE[i] Input Revolution Counter	$004C40_H$ +i*20000 H	PCLi, 32	PCLi, 32	Kernel Reset	7889
EGTM_CLSi_SPE_REV_CMP (i=0-2)	SPE[i] Revolution Counter Compare Value	$004C44_H$ +i*20000 H	PCLi, 32	PCLi, 32	Kernel Reset	7889
EGTM_CLSi_SPE_CTRL_STAT2 (i=0-2)	SPE[i] control status register 2	$004C48_H$ +i*20000 H	PCLi, 32	PCLi, 32	Kernel Reset	7890
EGTM_CLSi_SPE_CMD (i=0-2)	SPE[i] command register	$004C4C_H$ +i*20000 H	PCLi, 32	PCLi, 32	Kernel Reset	7891
EGTM_CLC	Clock control register	060000_H	P	P, SV, E	Application Reset	7892
EGTM_OCS	OCDS control and status register	060004_H	P	SV, P	Debug Reset	7893
EGTM_RST_CTRLA	Reset control register A	060010_H	P	P, SV, E	Application Reset	7894
EGTM_RST_CTRLB	Reset control register B	060014_H	P	P, SV, E	Application Reset	7894
EGTM_RST_STAT	Reset status register	060018_H	P	BE	Application Reset	7895
EGTM_PROTSE	PROT register Safe Endinit	$06001C_H$	U	SV, PROT	Application Reset	7896
EGTM_CLi_ACCEN_WRA (i=0-2)	Write access enable register A for cluster i	060024_H +i*24_H	U	SE, SV	Application Reset	7897
EGTM_CLi_ACCEN_WRB (i=0-2)	Write access enable register B for cluster i	060028_H +i*24_H	U	SE, SV	Application Reset	7898
EGTM_CLi_ACCEN_RDA (i=0-2)	Read access enable register A for cluster i	$06002C_H$ +i*24_H	U	SE, SV	Application Reset	7898
EGTM_CLi_ACCEN_RDB (i=0-2)	Read access enable register B for cluster i	060030_H +i*24_H	U	SE, SV	Application Reset	7899

(table continues...)

Table 2369 (continued) Register overview - EGTM domain CPU (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
EGTM_CLi_ACCEN_VM (i=0-2)	VM access enable register for cluster i	060034_H +i*24_H	U	SE, SV	Application Reset	7899
EGTM_CLi_ACCEN_PRS (i=0-2)	PRS access enable register for cluster i	060038_H +i*24_H	U	SE, SV	Application Reset	7900
EGTM_CTRL_PROTE	PROT register Endinit for control logic	$0601D0_H$	U	SV, PROT	Application Reset	7901
EGTM_CTRL_ACCEN_WRA	Write access enable register A for control logic	$0601D4_H$	U	SE, SV	Application Reset	7902
EGTM_CTRL_ACCEN_WRB	Write access enable register B for control logic	$0601D8_H$	U	SE, SV	Application Reset	7903
EGTM_CTRL_ACCEN_RDA	Read access enable register A for control logic	$0601DC_H$	U	SE, SV	Application Reset	7903
EGTM_CTRL_ACCEN_RDB	Read access enable register B for control logic	$0601E0_H$	U	SE, SV	Application Reset	7904
EGTM_CTRL_ACCEN_VM	VM access enable register for control logic	$0601E4_H$	U	SE, SV	Application Reset	7904
EGTM_CTRL_ACCEN_PRS	PRS access enable register for control logic	$0601E8_H$	U	SE, SV	Application Reset	7905
EGTM_WRAP_ACCEN_WRA	Write access enable register A for wrapper logic	$0601F8_H$	U	SE, SV	Application Reset	7906
EGTM_WRAP_ACCEN_WRB	Write access enable register B for wrapper logic	$0601FC_H$	U	SE, SV	Application Reset	7906
EGTM_WRAP_ACCEN_RDA	Read access enable register A for wrapper logic	060200_H	U	SE, SV	Application Reset	7907
EGTM_WRAP_ACCEN_RDB	Read access enable register B for wrapper logic	060204_H	U	SE, SV	Application Reset	7907
EGTM_WRAP_ACCEN_VM	VM access enable register for wrapper logic	060208_H	U	SE, SV	Application Reset	7908
EGTM_WRAP_ACCEN_PRS	PRS access enable register for wrapper logic	$06020C_H$	U	SE, SV	Application Reset	7908
EGTM_CTRLGTM_ACCEN_WRA	Write access enable register A for wrapper logic	$06021C_H$	U	SE, SV	Application Reset	7909
EGTM_CTRLGTM_ACCEN_WRB	Write access enable register B for wrapper logic	060220_H	U	SE, SV	Application Reset	7909
EGTM_CTRLGTM_ACCEN_RDA	Read access enable register A for wrapper logic	060224_H	U	SE, SV	Application Reset	7910
EGTM_CTRLGTM_ACCEN_RDB	Read access enable register B for wrapper logic	060228_H	U	SE, SV	Application Reset	7910
EGTM_CTRLGTM_ACCEN_VM	VM access enable register for wrapper logic	$06022C_H$	U	SE, SV	Application Reset	7911

(table continues...)

Table 2369 (continued) Register overview - EGTM domain CPU (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
EGTM_CTRLGTM_ACCEN_PRS	PRS access enable register for wrapper logic	060230_H	U	SE, SV	Application Reset	7912
EGTM_SAFEGTM_ACCEN_WRA	Write access enable register A for wrapper logic	060240_H	U	SE, SV	Application Reset	7912
EGTM_SAFEGTM_ACCEN_WRB	Write access enable register B for wrapper logic	060244_H	U	SE, SV	Application Reset	7913
EGTM_SAFEGTM_ACCEN_RDA	Read access enable register A for wrapper logic	060248_H	U	SE, SV	Application Reset	7913
EGTM_SAFEGTM_ACCEN_RDB	Read access enable register B for wrapper logic	$06024C_H$	U	SE, SV	Application Reset	7914
EGTM_SAFEGTM_ACCEN_VM	VM access enable register for wrapper logic	060250_H	U	SE, SV	Application Reset	7914
EGTM_SAFEGTM_ACCEN_PRS	PRS access enable register for wrapper logic	060254_H	U	SE, SV	Application Reset	7915
EGTM_SPARE	Spare Bits	060480_H	U	U	Kernel Reset	7916
EGTM_OCDS_OTBU0T	OCDS TBU0 trigger register	060500_H	P	P, SV	Debug Reset	7917
EGTM_OCDS_OTBU1T	OCDS TBU1 trigger register	060504_H	P	P, SV	Debug Reset	7917
EGTM_OCDS_OTBU2T	OCDS TBU2 trigger register	060508_H	P	P, SV	Debug Reset	7918
EGTM_OCDS_OTSS	OCDS trigger set select register	060510_H	P	P, SV	Debug Reset	7620
EGTM_OCDS_OTSC0	OCDS trigger set control 0 register	060514_H	P	P, SV	Debug Reset	7919
EGTM_TIMnINSEL (n=0-2)	TIMn input selection register	061000_H +n*4	PWRAP	PWRAP, SV	Kernel Reset	7959
EGTM_TIMnINSELB (n=0-2)	TIMn input selection register group B	061050_H +n*4	PWRAP	PWRAP, SV	Kernel Reset	7983
EGTM_TOUTSELn (n=0-48)	eGTM to Ports signal TOUT output selection register n	061100_H +n*4	PWRAP	PWRAP, SV	Kernel Reset	7989
EGTM_MSCSETi_CONj (i=0;j=0-3) (i=1;j=0-3) (i=2;j=0-3) (i=3;j=0-3)	eGTM to MSC SETi output selection register j	061200_H +i*10_H+j*4	PWRAP	PWRAP, SV	Kernel Reset	8205

(table continues...)

Table 2369 (continued) Register overview - EGTM domain CPU (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
EGTM_MSCn_INLCON (n=0-1)	eGTM to input bus MSCn_INLCON signal selection register	061290_H +n*10_H	PWRAP	PWRAP, SV	Kernel Reset	8222
EGTM_MSCn_INHCON (n=0-1)	eGTM to input bus MSCn_INHCON signal selection register	061294_H +n*10_H	PWRAP	PWRAP, SV	Kernel Reset	8223
EGTM_MSCn_INLECON (n=0-1)	eGTM to input bus MSCn_INLECON signal selection register	061298_H +n*10_H	PWRAP	PWRAP, SV	Kernel Reset	8224
EGTM_MSCn_INHECON (n=0-1)	eGTM to input bus MSCn_INHECON signal selection register	$06129C_H$ +n*10_H	PWRAP	PWRAP, SV	Kernel Reset	8225
EGTM_MSCSELIN_INLCON	Selection of INLCON stream for MSC	$0612D0_H$	PWRAP	PWRAP, SV	Kernel Reset	7612
EGTM_MSCSELIN_INHCON	Selection of INHCON stream for MSC	$0612D4_H$	PWRAP	PWRAP, SV	Kernel Reset	7613
EGTM_MSCSELIN_INLECON	Selection of INLECON stream for MSC	$0612D8_H$	PWRAP	PWRAP, SV	Kernel Reset	7612
EGTM_MSCSELIN_INHECON	Selection of INHECON stream for MSC	$0612DC_H$	PWRAP	PWRAP, SV	Kernel Reset	7613
EGTM_PSI5_OUT	eGTM to PSI5 output trigger selection register	061400_H	PWRAP	PWRAP, SV	Kernel Reset	8226
EGTM_PSI5Sn_OUT (n=0-1)	eGTM to PSI5-Sn output trigger selection register	061410_H +n*4	PWRAP	PWRAP, SV	Kernel Reset	8227
EGTM_LCDCDC_OUT	eGTM to LCDCDC output trigger selection register	061420_H	PWRAP	PWRAP, SV	Kernel Reset	8230
EGTM_CANn_OUTx (n=0;x=0-3) (n=1;x=0-3) (n=2;x=0-3) (n=3;x=0-3) (n=4;x=0-3) (n=5;x=0-3)	eGTM to CANn output selection register x. n*4+x<24 , if interface exists.	061480_H +n*10_H+x*4	PWRAP	PWRAP, SV	Kernel Reset	8231
EGTM_FOUTENCx (x=0-3)	FOUT Selection for Encoder Interface x, if GPT resources exist.	061500_H +x*4	PWRAP	PWRAP, SV	Kernel Reset	7589
EGTM_QSPIn_OUT (n=0-7)	eGTM to QSPIn output selection register. n<8 , if interface exists.	061580_H +n*4	PWRAP	PWRAP, SV	Kernel Reset	8247

(table continues...)

Table 2369 (continued) Register overview - EGTM domain CPU (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
EGTM_ASCLINn_OUT (n=0-27)	eGTM to ASCLINn output selection register. n<28 , if interface exists.	061600_H +n*4	PWRAP	PWRAP, SV	Kernel Reset	8255
EGTM_SENTn_OUTA (n=0-1)	eGTM to SENTn trigger x output selection register A	061680_H +n*8	PWRAP	PWRAP, SV	Kernel Reset	8283
EGTM_ADC_OUTn (n=0-2)	eGTM to ADC output selection register n	061700_H +n*4	PWRAP	PWRAP, SV	Kernel Reset	8286
EGTM_DTMAUXn_INy (n=0;y=0-1) (n=1;y=0-1) (n=2;y=0-1)	Cluster n DTMAUX input y selection register	061780_H +n*8+y*4	PWRAP	PWRAP, SV	Kernel Reset	8292
EGTM_SCU_OUTn (n=0-2)	eGTM to SCU output selection register n eGTM_SCU_TRIG(n)	061800_H +n*4	PWRAP	PWRAP, SV	Kernel Reset	8298
EGTM_TIMn_ADCSELCHi (i=0-7;n=0-2)	Instance Selection for TIM[n] channel[i]	$061B00_H$ +n*20_H+i*4	PWRAP	PWRAP, SV	Kernel Reset	7586
EGTM_HRPWMMUXz_CHSEL (z=0-2)	HRPWMz 16:1 Channel MUX select	062000_H +z*4	PCLz	PCLz	Kernel Reset	7936
EGTM_HRPWMz_EN (z=0-2)	HRPWMz module and channel enable register	062400_H +z*400_H	PCLz	PCLz	Kernel Reset	7937

44.9.4.4 TIMn input selection register (n=0)

EGTM_TIMnINSEL (n=0) Offset address: 061000_H+n*4

TIMn input selection register Kernel Reset value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CH7SEL				CH6SEL				CH5SEL				CH4SEL			
rw				rw				rw				rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH3SEL				CH2SEL				CH1SEL				CH0SEL			
rw				rw				rw				rw			

Field	Bits	Type	Description
CHxSEL (x=0)	4*x+3:4*x	rw	**eGTM TIMn channel x input selection** This bit field selects which input is connected for TIMn channel x. The input is either derived from a port pad or from an on-chip module. 0_H **None**: Safe state. No input connected to TIM0_IN0 1_H **P00.9**: Port pad input. Can be used, if existent on device. See datasheet. 2_H **P02.0**: Port pad input. Can be used, if existent on device. See datasheet. 3_H **P10.7**: Port pad input. Can be used, if existent on device. See datasheet. 4_H **P14.5**: Port pad input. Can be used, if existent on device. See datasheet. 5_H **P14.7**: Port pad input. Can be used, if existent on device. See datasheet. 6_H **P15.6**: Port pad input. Can be used, if existent on device. See datasheet. 7_H **P21.2**: Port pad input. Can be used, if existent on device. See datasheet. 8_H **P22.1**: Port pad input. Can be used, if existent on device. See datasheet. 9_H **P33.10**: Port pad input. Can be used, if existent on device. See datasheet. A_H **P33.4**: Port pad input. Can be used, if existent on device. See datasheet. B_H **P13.12**: Port pad input. Can be used, if existent on device. See datasheet. C_H **GTBF0OUT0**: Output from ADC Multiplexer D_H **RTCMUX**: PMS RTC Multiplexer Input E_H **PDOUT0**: SCU/ERU pattern detection output 0 F_H **CAN0_INT12**: CAN0 interrupt output INT12

(table continues...)

(continued)

Field	Bits	Type	Description
CHxSEL (x=1)	4*x+3:4*x	rw	**eGTM TIMn channel x input selection** This bit field selects which input is connected for TIMn channel x. The input is either derived from a port pad or from an on-chip module. 0_H **None**: Safe state. No input connected to TIM0_IN1 1_H **P00.10**: Port pad input. Can be used, if existent on device. See datasheet. 2_H **P02.1**: Port pad input. Can be used, if existent on device. See datasheet. 3_H **P10.1**: Port pad input. Can be used, if existent on device. See datasheet. 4_H **P14.6**: Port pad input. Can be used, if existent on device. See datasheet. 5_H **P15.7**: Port pad input. Can be used, if existent on device. See datasheet. 6_H **P21.3**: Port pad input. Can be used, if existent on device. See datasheet. 7_H **P22.0**: Port pad input. Can be used, if existent on device. See datasheet. 8_H **P33.5**: Port pad input. Can be used, if existent on device. See datasheet. 9_H **P33.9**: Port pad input. Can be used, if existent on device. See datasheet. A_H **P10.9**: Port pad input. Can be used, if existent on device. See datasheet. B_H **P16.0**: Port pad input. Can be used, if existent on device. See datasheet. C_H **GTBF0OUT0**: Output from ADC Multiplexer D_H **RTCMUX**: PMS RTC Multiplexer Input E_H **PDOUT1**: SCU/ERU pattern detection output 1 F_H **CAN0_INT13**: CAN0 interrupt output INT13

(table continues...)

(continued)

Field	Bits	Type	Description
CHxSEL (x=2)	4*x+3:4*x	rw	**eGTM TIMn channel x input selection** This bit field selects which input is connected for TIMn channel x. The input is either derived from a port pad or from an on-chip module. 0_H **None**: Safe state. No input connected to TIM0_IN2 1_H **P00.11**: Port pad input. Can be used, if existent on device. See datasheet. 2_H **P02.2**: Port pad input. Can be used, if existent on device. See datasheet. 3_H **P10.2**: Port pad input. Can be used, if existent on device. See datasheet. 4_H **P10.5**: Port pad input. Can be used, if existent on device. See datasheet. 5_H **P15.8**: Port pad input. Can be used, if existent on device. See datasheet. 6_H **P21.4**: Port pad input. Can be used, if existent on device. See datasheet. 7_H **P23.5**: Port pad input. Can be used, if existent on device. See datasheet. 8_H **P33.11**: Port pad input. Can be used, if existent on device. See datasheet. 9_H **P33.6**: Port pad input. Can be used, if existent on device. See datasheet. A_H **P02.9**: Port pad input. Can be used, if existent on device. See datasheet. B_H **P10.10**: Port pad input. Can be used, if existent on device. See datasheet. C_H **GTBF0OUT1**: Output from ADC Multiplexer D_H **GPT120_T6OFL**: T6OFL from GPT120 module can be used, if existent on device. See datasheet. E_H **PDOUT2**: SCU/ERU pattern detection output 2 F_H **CAN0_INT13**: CAN0 interrupt output INT13

(table continues...)

(continued)

Field	Bits	Type	Description
CHxSEL (x=3)	4*x+3:4*x	rw	**eGTM TIMn channel x input selection** This bit field selects which input is connected for TIMn channel x. The input is either derived from a port pad or from an on-chip module. 0_H **None**: Safe state. No input connected to TIM0_IN3 1_H **P00.12**: Port pad input. Can be used, if existent on device. See datasheet. 2_H **P02.3**: Port pad input. Can be used, if existent on device. See datasheet. 3_H **P10.3**: Port pad input. Can be used, if existent on device. See datasheet. 4_H **P10.6**: Port pad input. Can be used, if existent on device. See datasheet. 5_H **P14.0**: Port pad input. Can be used, if existent on device. See datasheet. 6_H **P21.5**: Port pad input. Can be used, if existent on device. See datasheet. 7_H **P22.2**: Port pad input. Can be used, if existent on device. See datasheet. 8_H **P32.2**: Port pad input. Can be used, if existent on device. See datasheet. 9_H **P33.7**: Port pad input. Can be used, if existent on device. See datasheet. A_H **P02.10**: Port pad input. Can be used, if existent on device. See datasheet. B_H **P10.14**: Port pad input. Can be used, if existent on device. See datasheet. C_H **GTBF0OUT1**: Output from ADC Multiplexer D_H **P02.7**: Port pad input. Can be used, if existent on device. See datasheet. E_H **PDOUT3**: SCU/ERU pattern detection output 3 F_H **CAN0_INT14**: CAN0 interrupt output INT14

(table continues...)

(continued)

Field	Bits	Type	Description
CHxSEL (x=4)	4*x+3:4*x	rw	**eGTM TIMn channel x input selection** This bit field selects which input is connected for TIMn channel x. The input is either derived from a port pad or from an on-chip module. 0_H **None**: Safe state. No input connected to TIM0_IN4 1_H **P02.4**: Port pad input. Can be used, if existent on device. See datasheet. 2_H **P10.0**: Port pad input. Can be used, if existent on device. See datasheet. 3_H **P14.1**: Port pad input. Can be used, if existent on device. See datasheet. 4_H **P22.3**: Port pad input. Can be used, if existent on device. See datasheet. 5_H **P13.8**: Port pad input. Can be used, if existent on device. See datasheet. 6_H **P33.0**: Port pad input. Can be used, if existent on device. See datasheet. 7_H **P33.8**: Port pad input. Can be used, if existent on device. See datasheet. 8_H **P21.6**: Port pad input. Can be used, if existent on device. See datasheet. 9_H **P10.13**: Port pad input. Can be used, if existent on device. See datasheet. A_H **P00.0**: Port pad input. Can be used, if existent on device. See datasheet. B_H **P00.1**: Port pad input. Can be used, if existent on device. See datasheet. C_H **GTBF0OUT2_4**: Output from ADC Multiplexer D_H **GPT121_T6OFL**: T6OFL from GPT121 module can be used, if existent on device. See datasheet. E_H **PDOUT4**: SCU/ERU pattern detection output 4 F_H **CAN0_INT15**: CAN0 interrupt output INT15

(table continues...)

(continued)

Field	Bits	Type	Description
CHxSEL (x=5)	4*x+3:4*x	rw	**eGTM TIMn channel x input selection** This bit field selects which input is connected for TIMn channel x. The input is either derived from a port pad or from an on-chip module. 0_H **None**: Safe state. No input connected to TIM0_IN5 1_H **P02.5**: Port pad input. Can be used, if existent on device. See datasheet. 2_H **P10.8**: Port pad input. Can be used, if existent on device. See datasheet. 3_H **P14.2**: Port pad input. Can be used, if existent on device. See datasheet. 4_H **P23.0**: Port pad input. Can be used, if existent on device. See datasheet. 5_H **P32.4**: Port pad input. Can be used, if existent on device. See datasheet. 6_H **P33.1**: Port pad input. Can be used, if existent on device. See datasheet. 7_H **P21.7**: Port pad input. Can be used, if existent on device. See datasheet. 8_H **P01.3**: Port pad input. Can be used, if existent on device. See datasheet. 9_H **P10.11**: Port pad input. Can be used, if existent on device. See datasheet. A_H **P16.0**: Port pad input. Can be used, if existent on device. See datasheet. B_H **DCDC_SYNC**: Sync Signal from PMS for DCDC control C_H **GTBF0OUT2_4**: Output from ADC Multiplexer D_H **GPT122_T6OFL**: T6OFL from GPT122 module can be used, if existent on device. See datasheet. E_H **PDOUT5**: SCU/ERU pattern detection output 5 F_H **GETHPPS0**: Gigabit Ethernet PPS0 Interrupt

(table continues...)

(continued)

Field	Bits	Type	Description
CHxSEL (x=6)	4*x+3:4*x	rw	**eGTM TIMn channel x input selection** This bit field selects which input is connected for TIMn channel x. The input is either derived from a port pad or from an on-chip module. 0_H **None**: Safe state. No input connected to TIM0_IN6 1_H **P02.6**: Port pad input. Can be used, if existent on device. See datasheet. 2_H **P10.4**: Port pad input. Can be used, if existent on device. See datasheet. 3_H **P14.3**: Port pad input. Can be used, if existent on device. See datasheet. 4_H **P23.1**: Port pad input. Can be used, if existent on device. See datasheet. 5_H **P23.2**: Port pad input. Can be used, if existent on device. See datasheet. 6_H **P33.2**: Port pad input. Can be used, if existent on device. See datasheet. 7_H **P20.0**: Port pad input. Can be used, if existent on device. See datasheet. 8_H **P01.4**: Port pad input. Can be used, if existent on device. See datasheet. 9_H **P10.15**: Port pad input. Can be used, if existent on device. See datasheet. A_H **P16.4**: Port pad input. Can be used, if existent on device. See datasheet. B_H **P00.5**: Port pad input. Can be used, if existent on device. See datasheet. C_H **GTBF0OUT3_5**: Output from ADC Multiplexer D_H **GPT123_T6OFL**: T6OFL from GPT123 module can be used, if existent on device. See datasheet. E_H **PDOUT6**: SCU/ERU pattern detection output 6 F_H **GETHPPS1**: Gigabit Ethernet PPS1 Interrupt

(table continues...)

(continued)

Field	Bits	Type	Description
CHxSEL (x=7)	4*x+3:4*x	rw	**eGTM TIMn channel x input selection** This bit field selects which input is connected for TIMn channel x. The input is either derived from a port pad or from an on-chip module. 0_H **None**: Safe state. No input connected to TIM0_IN7 1_H **P02.7**: Port pad input. Can be used, if existent on device. See datasheet. 2_H **P14.4**: Port pad input. Can be used, if existent on device. See datasheet. 3_H **P20.8**: Port pad input. Can be used, if existent on device. See datasheet. 4_H **P23.3**: Port pad input. Can be used, if existent on device. See datasheet. 5_H **P23.4**: Port pad input. Can be used, if existent on device. See datasheet. 6_H **P33.3**: Port pad input. Can be used, if existent on device. See datasheet. 7_H **P02.11**: Port pad input. Can be used, if existent on device. See datasheet. 8_H **P4.9**: Port pad input. Can be used, if existent on device. See datasheet. 9_H **P16.3**: Port pad input. Can be used, if existent on device. See datasheet. A_H **P00.8**: Port pad input. Can be used, if existent on device. See datasheet. B_H **WUTUFLOW**: PMS: Underflow output to support WUT calibration C_H **GTBF0OUT3_5**: Output from ADC Multiplexer D_H **P02.8**: Port pad input. Can be used, if existent on device. See datasheet. E_H **PDOUT7**: SCU/ERU pattern detection output 7 F_H **MT0**: FlexRay macrotick MT0 clock from CC

44.9.4.5 TIMn input selection register (n=1)

EGTM_TIMnINSEL (n=1) Offset address: 061000_H+n*4

TIMn input selection register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CH7SEL				CH6SEL				CH5SEL				CH4SEL			
rw				rw				rw				rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH3SEL				CH2SEL				CH1SEL				CH0SEL			
rw				rw				rw				rw			

Field	Bits	Type	Description
CHxSEL (x=0)	4*x+3:4*x	rw	**eGTM TIMn channel x input selection** This bit field selects which input is connected for TIMn channel x. The input is either derived from a port pad or from an on-chip module. 0_H **None**: Safe state. No input connected to TIM1_IN0 1_H **P00.9**: Port pad input. Can be used, if existent on device. See datasheet. 2_H **P02.0**: Port pad input. Can be used, if existent on device. See datasheet. 3_H **P10.7**: Port pad input. Can be used, if existent on device. See datasheet. 4_H **P14.5**: Port pad input. Can be used, if existent on device. See datasheet. 5_H **P14.7**: Port pad input. Can be used, if existent on device. See datasheet. 6_H **P15.6**: Port pad input. Can be used, if existent on device. See datasheet. 7_H **P21.2**: Port pad input. Can be used, if existent on device. See datasheet. 8_H **P22.1**: Port pad input. Can be used, if existent on device. See datasheet. 9_H **P33.10**: Port pad input. Can be used, if existent on device. See datasheet. A_H **P33.4**: Port pad input. Can be used, if existent on device. See datasheet. B_H **P16.5**: Port pad input. Can be used, if existent on device. See datasheet. C_H **GTBF1OUT0**: Output from ADC Multiplexer D_H **P02.8**: Port pad input. Can be used, if existent on device. See datasheet. E_H **CAN1_INT12**: CAN1 interrupt output INT12 F_H **GETHPPS2**: Gigabit Ethernet PPS2 Interrupt

(table continues...)

(continued)

Field	Bits	Type	Description
CHxSEL (x=1)	4*x+3:4*x	rw	**eGTM TIMn channel x input selection** This bit field selects which input is connected for TIMn channel x. The input is either derived from a port pad or from an on-chip module. 0_H **None**: Safe state. No input connected to TIM1_IN1 1_H **P00.10**: Port pad input. Can be used, if existent on device. See datasheet. 2_H **P02.1**: Port pad input. Can be used, if existent on device. See datasheet. 3_H **P10.1**: Port pad input. Can be used, if existent on device. See datasheet. 4_H **P14.6**: Port pad input. Can be used, if existent on device. See datasheet. 5_H **P15.7**: Port pad input. Can be used, if existent on device. See datasheet. 6_H **P21.3**: Port pad input. Can be used, if existent on device. See datasheet. 7_H **P22.0**: Port pad input. Can be used, if existent on device. See datasheet. 8_H **P33.5**: Port pad input. Can be used, if existent on device. See datasheet. 9_H **P33.9**: Port pad input. Can be used, if existent on device. See datasheet. A_H **P16.4**: Port pad input. Can be used, if existent on device. See datasheet. B_H **P16.13**: Port pad input. Can be used, if existent on device. See datasheet. C_H **GTBF1OUT0**: Output from ADC Multiplexer D_H **P02.6**: Port pad input. Can be used, if existent on device. See datasheet. E_H **CAN1_INT13**: CAN1 interrupt output INT13 F_H **CAN0_INT12**: CAN0 interrupt output INT12

(table continues...)

(continued)

Field	Bits	Type	Description
CHxSEL (x=2)	4*x+3:4*x	rw	**eGTM TIMn channel x input selection** This bit field selects which input is connected for TIMn channel x. The input is either derived from a port pad or from an on-chip module. 0_H **None**: Safe state. No input connected to TIM1_IN2 1_H **P00.11**: Port pad input. Can be used, if existent on device. See datasheet. 2_H **P02.2**: Port pad input. Can be used, if existent on device. See datasheet. 3_H **P10.2**: Port pad input. Can be used, if existent on device. See datasheet. 4_H **P10.5**: Port pad input. Can be used, if existent on device. See datasheet. 5_H **P15.8**: Port pad input. Can be used, if existent on device. See datasheet. 6_H **P21.4**: Port pad input. Can be used, if existent on device. See datasheet. 7_H **P23.5**: Port pad input. Can be used, if existent on device. See datasheet. 8_H **P33.11**: Port pad input. Can be used, if existent on device. See datasheet. 9_H **P33.6**: Port pad input. Can be used, if existent on device. See datasheet. A_H **P23.6**: Port pad input. Can be used, if existent on device. See datasheet. B_H **P00.1**: Port pad input. Can be used, if existent on device. See datasheet. C_H **GTBF1OUT1**: Output from ADC Multiplexer D_H **LETHPPS0**: Lite Ethernet PPS0 Interrupt E_H **CAN2_INT12**: CAN2 interrupt output INT12 F_H **CAN0_INT13**: CAN0 interrupt output INT13

(table continues...)

(continued)

Field	Bits	Type	Description
CHxSEL (x=3)	4*x+3:4*x	rw	**eGTM TIMn channel x input selection** This bit field selects which input is connected for TIMn channel x. The input is either derived from a port pad or from an on-chip module. 0_H **None**: Safe state. No input connected to TIM1_IN3 1_H **P00.12**: Port pad input. Can be used, if existent on device. See datasheet. 2_H **P02.3**: Port pad input. Can be used, if existent on device. See datasheet. 3_H **P10.3**: Port pad input. Can be used, if existent on device. See datasheet. 4_H **P10.6**: Port pad input. Can be used, if existent on device. See datasheet. 5_H **P14.0**: Port pad input. Can be used, if existent on device. See datasheet. 6_H **P21.5**: Port pad input. Can be used, if existent on device. See datasheet. 7_H **P22.2**: Port pad input. Can be used, if existent on device. See datasheet. 8_H **P32.2**: Port pad input. Can be used, if existent on device. See datasheet. 9_H **P33.7**: Port pad input. Can be used, if existent on device. See datasheet. A_H **P23.7**: Port pad input. Can be used, if existent on device. See datasheet. B_H **P00.3**: Port pad input. Can be used, if existent on device. See datasheet. C_H **GTBF1OUT1**: Output from ADC Multiplexer D_H **LETHPPS1**: Lite Ethernet PPS1 Interrupt E_H **CAN2_INT13**: CAN2 interrupt output INT13 F_H **CAN0_INT14**: CAN0 interrupt output INT14

(table continues...)

(continued)

Field	Bits	Type	Description
CHxSEL (x=4)	4*x+3:4*x	rw	**eGTM TIMn channel x input selection** This bit field selects which input is connected for TIMn channel x. The input is either derived from a port pad or from an on-chip module. 0_H **None**: Safe state. No input connected to TIM1_IN4 1_H **P02.4**: Port pad input. Can be used, if existent on device. See datasheet. 2_H **P10.0**: Port pad input. Can be used, if existent on device. See datasheet. 3_H **P14.1**: Port pad input. Can be used, if existent on device. See datasheet. 4_H **P22.3**: Port pad input. Can be used, if existent on device. See datasheet. 5_H **P13.8**: Port pad input. Can be used, if existent on device. See datasheet. 6_H **P33.0**: Port pad input. Can be used, if existent on device. See datasheet. 7_H **P33.8**: Port pad input. Can be used, if existent on device. See datasheet. 8_H **P21.6**: Port pad input. Can be used, if existent on device. See datasheet. 9_H **P20.0**: Port pad input. Can be used, if existent on device. See datasheet. A_H **P16.5**: Port pad input. Can be used, if existent on device. See datasheet. B_H **P16.13**: Port pad input. Can be used, if existent on device. See datasheet. C_H **GTBF1OUT2_4**: Output from ADC Multiplexer D_H **LETHPPS2**: Lite Ethernet PPS2 Interrupt E_H **CAN3_INT12**: CAN3 interrupt output INT12 F_H **CAN0_INT15**: CAN0 interrupt output INT15

(table continues...)

(continued)

Field	Bits	Type	Description
CHxSEL (x=5)	4*x+3:4*x	rw	**eGTM TIMn channel x input selection** This bit field selects which input is connected for TIMn channel x. The input is either derived from a port pad or from an on-chip module. 0_H **None**: Safe state. No input connected to TIM1_IN5 1_H **P02.5**: Port pad input. Can be used, if existent on device. See datasheet. 2_H **P10.8**: Port pad input. Can be used, if existent on device. See datasheet. 3_H **P14.2**: Port pad input. Can be used, if existent on device. See datasheet. 4_H **P23.0**: Port pad input. Can be used, if existent on device. See datasheet. 5_H **P32.4**: Port pad input. Can be used, if existent on device. See datasheet. 6_H **P33.1**: Port pad input. Can be used, if existent on device. See datasheet. 7_H **P21.7**: Port pad input. Can be used, if existent on device. See datasheet. 8_H **P20.7**: Port pad input. Can be used, if existent on device. See datasheet. 9_H **P16.3**: Port pad input. Can be used, if existent on device. See datasheet. A_H **P16.12**: Port pad input. Can be used, if existent on device. See datasheet. B_H **DCDC_SYNC**: Sync Signal from PMS for DCDC control C_H **GTBF1OUT2_4**: Output from ADC Multiplexer D_H **GPT120_T6OFL**: T6OFL from GPT120 module can be used, if existent on device. See datasheet. E_H **CAN3_INT13**: CAN3 interrupt output INT13 F_H **MT1**: FlexRay macrotick clock MT1 from CC

(table continues...)

(continued)

Field	Bits	Type	Description
CHxSEL (x=6)	4*x+3:4*x	rw	**eGTM TIMn channel x input selection** This bit field selects which input is connected for TIMn channel x. The input is either derived from a port pad or from an on-chip module. 0_H **None**: Safe state. No input connected to TIM1_IN6 1_H **P02.6**: Port pad input. Can be used, if existent on device. See datasheet. 2_H **P10.4**: Port pad input. Can be used, if existent on device. See datasheet. 3_H **P14.3**: Port pad input. Can be used, if existent on device. See datasheet. 4_H **P23.1**: Port pad input. Can be used, if existent on device. See datasheet. 5_H **P23.2**: Port pad input. Can be used, if existent on device. See datasheet. 6_H **P33.2**: Port pad input. Can be used, if existent on device. See datasheet. 7_H **P20.0**: Port pad input. Can be used, if existent on device. See datasheet. 8_H **P16.2**: Port pad input. Can be used, if existent on device. See datasheet. 9_H **P16.8**: Port pad input. Can be used, if existent on device. See datasheet. A_H **P00.5**: Port pad input. Can be used, if existent on device. See datasheet. B_H **P00.7**: Port pad input. Can be used, if existent on device. See datasheet. C_H **GTBF1OUT3_5**: Output from ADC Multiplexer D_H **GPT121_T6OFL**: T6OFL from GPT121 module can be used, if existent on device. See datasheet. E_H **CAN4_INT12**: CAN4 interrupt output INT12 F_H **GETHPPS3**: Gigabit Ethernet PPS3 Interrupt

(table continues...)

(continued)

Field	Bits	Type	Description
CHxSEL (x=7)	4*x+3:4*x	rw	**eGTM TIMn channel x input selection** This bit field selects which input is connected for TIMn channel x. The input is either derived from a port pad or from an on-chip module. 0_H **None**: Safe state. No input connected to TIM1_IN7 1_H **P02.7**: Port pad input. Can be used, if existent on device. See datasheet. 2_H **P14.4**: Port pad input. Can be used, if existent on device. See datasheet. 3_H **P20.8**: Port pad input. Can be used, if existent on device. See datasheet. 4_H **P23.3**: Port pad input. Can be used, if existent on device. See datasheet. 5_H **P23.4**: Port pad input. Can be used, if existent on device. See datasheet. 6_H **P33.3**: Port pad input. Can be used, if existent on device. See datasheet. 7_H **P16.2**: Port pad input. Can be used, if existent on device. See datasheet. 8_H **P16.9**: Port pad input. Can be used, if existent on device. See datasheet. 9_H **P16.8**: Port pad input. Can be used, if existent on device. See datasheet. A_H **P00.8**: Port pad input. Can be used, if existent on device. See datasheet. B_H **P00.9**: Port pad input. Can be used, if existent on device. See datasheet. C_H **GTBF1OUT3_5**: Output from ADC Multiplexer D_H **GPT122_T6OFL**: T6OFL from GPT122 module can be used, if existent on device. See datasheet. E_H **CAN4_INT13**: CAN4 interrupt output INT13 F_H **PDOUT7**: SCU/ERU pattern detection output 7

44.9.4.6 TIMn input selection register (n=2)

EGTM_TIMnINSEL (n=2) Offset address: 061000_H+n*4

TIMn input selection register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CH7SEL				CH6SEL				CH5SEL				CH4SEL			
rw				rw				rw				rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH3SEL				CH2SEL				CH1SEL				CH0SEL			
rw				rw				rw				rw			

Field	Bits	Type	Description
CHxSEL (x=0)	4*x+3:4*x	rw	**eGTM TIMn channel x input selection** This bit field selects which input is connected for TIMn channel x. The input is either derived from a port pad or from an on-chip module. 0_H **None**: Safe state. No input connected to TIM2_IN0 1_H **P00.0**: Port pad input. Can be used, if existent on device. See datasheet. 2_H **P02.8**: Port pad input. Can be used, if existent on device. See datasheet. 3_H **P13.3**: Port pad input. Can be used, if existent on device. See datasheet. 4_H **P15.5**: Port pad input. Can be used, if existent on device. See datasheet. 5_H **P20.12**: Port pad input. Can be used, if existent on device. See datasheet. 6_H **P33.12**: Port pad input. Can be used, if existent on device. See datasheet. 7_H **P11.0**: Port pad input. Can be used, if existent on device. See datasheet. 8_H **P33.14**: Port pad input. Can be used, if existent on device. See datasheet. 9_H **P11.10**: Port pad input. Can be used, if existent on device. See datasheet. A_H **P31.0**: Port pad input. Can be used, if existent on device. See datasheet. B_H **P01.3**: Port pad input. Can be used, if existent on device. See datasheet. C_H **GTBF2OUT0**: Output from ADC Multiplexer D_H **GPT123_T6OFL**: T6OFL from GPT123 module can be used, if existent on device. See datasheet. E_H **CAN1_INT14**: CAN1 interrupt output INT14 F_H **GETHPPS0**: Gigabit Ethernet PPS0 Interrupt

(table continues...)

(continued)

Field	Bits	Type	Description
CHxSEL (x=1)	4*x+3:4*x	rw	**eGTM TIMn channel x input selection** This bit field selects which input is connected for TIMn channel x. The input is either derived from a port pad or from an on-chip module. 0_H **None**: Safe state. No input connected to TIM2_IN1 1_H **P00.1**: Port pad input. Can be used, if existent on device. See datasheet. 2_H **P00.2**: Port pad input. Can be used, if existent on device. See datasheet. 3_H **P11.2**: Port pad input. Can be used, if existent on device. See datasheet. 4_H **P20.13**: Port pad input. Can be used, if existent on device. See datasheet. 5_H **P33.13**: Port pad input. Can be used, if existent on device. See datasheet. 6_H **P11.1**: Port pad input. Can be used, if existent on device. See datasheet. 7_H **P33.15**: Port pad input. Can be used, if existent on device. See datasheet. 8_H **P15.10**: Port pad input. Can be used, if existent on device. See datasheet. 9_H **P31.1**: Port pad input. Can be used, if existent on device. See datasheet. A_H **P01.4**: Port pad input. Can be used, if existent on device. See datasheet. B_H **CANXL_MTI_TRIG0**: CANXL MTI TRIG0 C_H **GTBF2OUT0**: Output from ADC Multiplexer D_H **LETHPPS3**: Lite Ethernet PPS3 Interrupt E_H **CAN1_INT15**: CAN1 interrupt output INT15 F_H **CAN0_INT12**: CAN0 interrupt output INT12

(table continues...)

(continued)

Field	Bits	Type	Description
CHxSEL (x=2)	4*x+3:4*x	rw	**eGTM TIMn channel x input selection** This bit field selects which input is connected for TIMn channel x. The input is either derived from a port pad or from an on-chip module. 0_H **None**: Safe state. No input connected to TIM2_IN2 1_H **P00.3**: Port pad input. Can be used, if existent on device. See datasheet. 2_H **P11.3**: Port pad input. Can be used, if existent on device. See datasheet. 3_H **P14.8**: Port pad input. Can be used, if existent on device. See datasheet. 4_H **P20.14**: Port pad input. Can be used, if existent on device. See datasheet. 5_H **P32.0**: Port pad input. Can be used, if existent on device. See datasheet. 6_H **P11.4**: Port pad input. Can be used, if existent on device. See datasheet. 7_H **P01.5**: Port pad input. Can be used, if existent on device. See datasheet. 8_H **P15.11**: Port pad input. Can be used, if existent on device. See datasheet. 9_H **P31.2**: Port pad input. Can be used, if existent on device. See datasheet. A_H **P15.6**: Port pad input. Can be used, if existent on device. See datasheet. B_H **CANXL_MTI_TRIG1**: CANXL MTI TRIG1 C_H **GTBF2OUT1**: Output from ADC Multiplexer D_H **CANXL_MTI_TRIG2**: CANXL MTI TRIG2 E_H **CAN2_INT14**: CAN2 interrupt output INT14 F_H **CAN0_INT13**: CAN0 interrupt output INT13

(table continues...)

(continued)

Field	Bits	Type	Description
CHxSEL (x=3)	4*x+3:4*x	rw	**eGTM TIMn channel x input selection** This bit field selects which input is connected for TIMn channel x. The input is either derived from a port pad or from an on-chip module. 0_H **None**: Safe state. No input connected to TIM2_IN3 1_H **P00.4**: Port pad input. Can be used, if existent on device. See datasheet. 2_H **P11.6**: Port pad input. Can be used, if existent on device. See datasheet. 3_H **P14.9**: Port pad input. Can be used, if existent on device. See datasheet. 4_H **P15.0**: Port pad input. Can be used, if existent on device. See datasheet. 5_H **P20.1**: Port pad input. Can be used, if existent on device. See datasheet. 6_H **P15.12**: Port pad input. Can be used, if existent on device. See datasheet. 7_H **P01.5**: Port pad input. Can be used, if existent on device. See datasheet. 8_H **P11.5**: Port pad input. Can be used, if existent on device. See datasheet. 9_H **P34.1**: Port pad input. Can be used, if existent on device. See datasheet. A_H **P31.3**: Port pad input. Can be used, if existent on device. See datasheet. B_H **CANXL_MTI_TRIG3**: CANXL MTI TRIG3 C_H **GTBF2OUT1**: Output from ADC Multiplexer D_H **TMADC0SR0**: TMADC0 SR0 E_H **CAN2_INT15**: CAN2 interrupt output INT15 F_H **CAN0_INT14**: CAN0 interrupt output INT14

(table continues...)

(continued)

Field	Bits	Type	Description
CHxSEL (x=4)	4*x+3:4*x	rw	**eGTM TIMn channel x input selection** This bit field selects which input is connected for TIMn channel x. The input is either derived from a port pad or from an on-chip module. 0_H **None**: Safe state. No input connected to TIM2_IN4 1_H **P00.5**: Port pad input. Can be used, if existent on device. See datasheet. 2_H **P11.9**: Port pad input. Can be used, if existent on device. See datasheet. 3_H **P14.10**: Port pad input. Can be used, if existent on device. See datasheet. 4_H **P15.1**: Port pad input. Can be used, if existent on device. See datasheet. 5_H **P20.3**: Port pad input. Can be used, if existent on device. See datasheet. 6_H **P21.0**: Port pad input. Can be used, if existent on device. See datasheet. 7_H **P11.7**: Port pad input. Can be used, if existent on device. See datasheet. 8_H **P34.2**: Port pad input. Can be used, if existent on device. See datasheet. 9_H **P15.13**: Port pad input. Can be used, if existent on device. See datasheet. A_H **P31.4**: Port pad input. Can be used, if existent on device. See datasheet. B_H **P02.14**: Port pad input. Can be used, if existent on device. See datasheet. C_H **GTBF2OUT2_4**: Output from ADC Multiplexer D_H **TMADC1SR0**: TMADC1 SR0 E_H **CAN3_INT14**: CAN3 interrupt output INT14 F_H **CAN0_INT15**: CAN0 interrupt output INT15

(table continues...)

(continued)

Field	Bits	Type	Description
CHxSEL (x=5)	4*x+3:4*x	rw	**eGTM TIMn channel x input selection** This bit field selects which input is connected for TIMn channel x. The input is either derived from a port pad or from an on-chip module. 0_H **None**: Safe state. No input connected to TIM2_IN5 1_H **P00.6**: Port pad input. Can be used, if existent on device. See datasheet. 2_H **P11.10**: Port pad input. Can be used, if existent on device. See datasheet. 3_H **P13.0**: Port pad input. Can be used, if existent on device. See datasheet. 4_H **P15.2**: Port pad input. Can be used, if existent on device. See datasheet. 5_H **P20.9**: Port pad input. Can be used, if existent on device. See datasheet. 6_H **P21.1**: Port pad input. Can be used, if existent on device. See datasheet. 7_H **P01.6**: Port pad input. Can be used, if existent on device. See datasheet. 8_H **P11.8**: Port pad input. Can be used, if existent on device. See datasheet. 9_H **P34.3**: Port pad input. Can be used, if existent on device. See datasheet. A_H **P15.14**: Port pad input. Can be used, if existent on device. See datasheet. B_H **P31.5**: Port pad input. Can be used, if existent on device. See datasheet. C_H **GTBF2OUT2_4**: Output from ADC Multiplexer D_H **P02.15**: Port pad input. Can be used, if existent on device. See datasheet. E_H **CAN3_INT15**: CAN3 interrupt output INT15 F_H **TMADC2SR0**: TMADC2 SR0

(table continues...)

(continued)

Field	Bits	Type	Description
CHxSEL (x=6)	4*x+3:4*x	rw	**eGTM TIMn channel x input selection** This bit field selects which input is connected for TIMn channel x. The input is either derived from a port pad or from an on-chip module. 0_H **None**: Safe state. No input connected to TIM2_IN6 1_H **P00.7**: Port pad input. Can be used, if existent on device. See datasheet. 2_H **P11.11**: Port pad input. Can be used, if existent on device. See datasheet. 3_H **P13.1**: Port pad input. Can be used, if existent on device. See datasheet. 4_H **P15.3**: Port pad input. Can be used, if existent on device. See datasheet. 5_H **P20.6**: Port pad input. Can be used, if existent on device. See datasheet. 6_H **P20.10**: Port pad input. Can be used, if existent on device. See datasheet. 7_H **P11.13**: Port pad input. Can be used, if existent on device. See datasheet. 8_H **P34.4**: Port pad input. Can be used, if existent on device. See datasheet. 9_H **P15.15**: Port pad input. Can be used, if existent on device. See datasheet. A_H **P31.6**: Port pad input. Can be used, if existent on device. See datasheet. B_H **P22.6**: Port pad input. Can be used, if existent on device. See datasheet. C_H **GTBF2OUT3_5**: Output from ADC Multiplexer D_H **P01.0**: Port pad input. Can be used, if existent on device. See datasheet. E_H **CAN4_INT14**: CAN4 interrupt output INT14 F_H **TMADC3SR0**: TMADC3 SR0

(table continues...)

(continued)

Field	Bits	Type	Description
CHxSEL (x=7)	4*x+3:4*x	rw	**eGTM TIMn channel x input selection** This bit field selects which input is connected for TIMn channel x. The input is either derived from a port pad or from an on-chip module. 0_H **None**: Safe state. No input connected to TIM2_IN7 1_H **P00.8**: Port pad input. Can be used, if existent on device. See datasheet. 2_H **P11.12**: Port pad input. Can be used, if existent on device. See datasheet. 3_H **P13.2**: Port pad input. Can be used, if existent on device. See datasheet. 4_H **P15.4**: Port pad input. Can be used, if existent on device. See datasheet. 5_H **P20.7**: Port pad input. Can be used, if existent on device. See datasheet. 6_H **P20.11**: Port pad input. Can be used, if existent on device. See datasheet. 7_H **P01.7**: Port pad input. Can be used, if existent on device. See datasheet. 8_H **P11.14**: Port pad input. Can be used, if existent on device. See datasheet. 9_H **P34.5**: Port pad input. Can be used, if existent on device. See datasheet. A_H **P13.9**: Port pad input. Can be used, if existent on device. See datasheet. B_H **P31.7**: Port pad input. Can be used, if existent on device. See datasheet. C_H **GTBF2OUT3_5**: Output from ADC Multiplexer D_H **PDOUT7**: SCU Pattern Detect Output 7 E_H **CAN4_INT15**: CAN4 interrupt output INT15 F_H **MT0**: FlexRay macrotick MT0 clock from CC

44.9.4.7 TIMn input selection register group B (n=0)

EGTM_TIMnINSELB (n=0) Offset address: 061050_H+n*4

TIMn input selection register group B Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CH7SEL				CH6SEL				CH5SEL				CH4SEL			
rw				rw				rw				rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH3SEL				CH2SEL				CH1SEL				CH0SEL			
rw				rw				rw				rw			

Field	Bits	Type	Description
CHxSEL (x=0)	4*x+3:4*x	rw	**eGTM TIMn channel x input selection group B** This bit field selects which input is connected for TIMn channel x (group B). The input is either derived from a port pad or from an on-chip module. 0_H **None**: Safe state. No input connected to TIM0_INB0 1_H **Reserved**: Reserved … F_H **Reserved**: Reserved
CHxSEL (x=1)	4*x+3:4*x	rw	**eGTM TIMn channel x input selection group B** This bit field selects which input is connected for TIMn channel x (group B). The input is either derived from a port pad or from an on-chip module. 0_H **None**: Safe state. No input connected to TIM0_INB1 1_H **Reserved**: Reserved … F_H **Reserved**: Reserved
CHxSEL (x=2)	4*x+3:4*x	rw	**eGTM TIMn channel x input selection group B** This bit field selects which input is connected for TIMn channel x (group B). The input is either derived from a port pad or from an on-chip module. 0_H **None**: Safe state. No input connected to TIM0_INB2 1_H **Reserved**: Reserved … F_H **Reserved**: Reserved
CHxSEL (x=3)	4*x+3:4*x	rw	**eGTM TIMn channel x input selection group B** This bit field selects which input is connected for TIMn channel x (group B). The input is either derived from a port pad or from an on-chip module. 0_H **None**: Safe state. No input connected to TIM0_INB3 1_H **Reserved**: Reserved … F_H **Reserved**: Reserved
CHxSEL (x=4)	4*x+3:4*x	rw	**eGTM TIMn channel x input selection group B** This bit field selects which input is connected for TIMn channel x (group B). The input is either derived from a port pad or from an on-chip module. 0_H **None**: Safe state. No input connected to TIM0_INB4 1_H **Reserved**: Reserved … F_H **Reserved**: Reserved

(table continues...)

(continued)

Field	Bits	Type	Description
CHxSEL (x=5)	4*x+3:4*x	rw	**eGTM TIMn channel x input selection group B** This bit field selects which input is connected for TIMn channel x (group B). The input is either derived from a port pad or from an on-chip module. 0_H **None**: Safe state. No input connected to TIM0_INB5 1_H **Reserved**: Reserved … F_H **Reserved**: Reserved
CHxSEL (x=6)	4*x+3:4*x	rw	**eGTM TIMn channel x input selection group B** This bit field selects which input is connected for TIMn channel x (group B). The input is either derived from a port pad or from an on-chip module. 0_H **None**: Safe state. No input connected to TIM0_INB6 1_H **Reserved**: Reserved … F_H **Reserved**: Reserved
CHxSEL (x=7)	4*x+3:4*x	rw	**eGTM TIMn channel x input selection group B** This bit field selects which input is connected for TIMn channel x (group B). The input is either derived from a port pad or from an on-chip module. 0_H **None**: Safe state. No input connected to TIM0_INB7 1_H **Reserved**: Reserved … F_H **Reserved**: Reserved

44.9.4.8 TIMn input selection register group B (n=1)

EGTM_TIMnINSELB (n=1) Offset address: 061050_H+n*4

TIMn input selection register group B Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CH7SEL				CH6SEL				CH5SEL				CH4SEL			
rw				rw				rw				rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH3SEL				CH2SEL				CH1SEL				CH0SEL			
rw				rw				rw				rw			

Field	Bits	Type	Description
CHxSEL (x=0)	4*x+3:4*x	rw	**eGTM TIMn channel x input selection group B** This bit field selects which input is connected for TIMn channel x (group B). The input is either derived from a port pad or from an on-chip module. 0_H **None**: Safe state. No input connected to TIM1_INB0 1_H **Reserved**: Reserved … F_H **Reserved**: Reserved
CHxSEL (x=1)	4*x+3:4*x	rw	**eGTM TIMn channel x input selection group B** This bit field selects which input is connected for TIMn channel x (group B). The input is either derived from a port pad or from an on-chip module. 0_H **None**: Safe state. No input connected to TIM1_INB1 1_H **Reserved**: Reserved … F_H **Reserved**: Reserved
CHxSEL (x=2)	4*x+3:4*x	rw	**eGTM TIMn channel x input selection group B** This bit field selects which input is connected for TIMn channel x (group B). The input is either derived from a port pad or from an on-chip module. 0_H **None**: Safe state. No input connected to TIM1_INB2 1_H **Reserved**: Reserved … F_H **Reserved**: Reserved
CHxSEL (x=3)	4*x+3:4*x	rw	**eGTM TIMn channel x input selection group B** This bit field selects which input is connected for TIMn channel x (group B). The input is either derived from a port pad or from an on-chip module. 0_H **None**: Safe state. No input connected to TIM1_INB3 1_H **Reserved**: Reserved … F_H **Reserved**: Reserved
CHxSEL (x=4)	4*x+3:4*x	rw	**eGTM TIMn channel x input selection group B** This bit field selects which input is connected for TIMn channel x (group B). The input is either derived from a port pad or from an on-chip module. 0_H **None**: Safe state. No input connected to TIM1_INB4 1_H **Reserved**: Reserved … F_H **Reserved**: Reserved

(table continues...)

(continued)

Field	Bits	Type	Description
CHxSEL (x=5)	4*x+3:4*x	rw	**eGTM TIMn channel x input selection group B** This bit field selects which input is connected for TIMn channel x (group B). The input is either derived from a port pad or from an on-chip module. 0_H **None**: Safe state. No input connected to TIM1_INB5 1_H **Reserved**: Reserved … F_H **Reserved**: Reserved
CHxSEL (x=6)	4*x+3:4*x	rw	**eGTM TIMn channel x input selection group B** This bit field selects which input is connected for TIMn channel x (group B). The input is either derived from a port pad or from an on-chip module. 0_H **None**: Safe state. No input connected to TIM1_INB6 1_H **Reserved**: Reserved … F_H **Reserved**: Reserved
CHxSEL (x=7)	4*x+3:4*x	rw	**eGTM TIMn channel x input selection group B** This bit field selects which input is connected for TIMn channel x (group B). The input is either derived from a port pad or from an on-chip module. 0_H **None**: Safe state. No input connected to TIM1_INB7 1_H **Reserved**: Reserved … F_H **Reserved**: Reserved

44.9.4.9 TIMn input selection register group B (n=2)

EGTM_TIMnINSELB (n=2) Offset address: 061050_H+n*4

TIMn input selection register group B Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CH7SEL				CH6SEL				CH5SEL				CH4SEL			
rw				rw				rw				rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH3SEL				CH2SEL				CH1SEL				CH0SEL			
rw				rw				rw				rw			

Field	Bits	Type	Description
CHxSEL (x=0)	4*x+3:4*x	rw	**eGTM TIMn channel x input selection group B** This bit field selects which input is connected for TIMn channel x (group B). The input is either derived from a port pad or from an on-chip module. 0_H **None**: Safe state. No input connected to TIM2_INB0 1_H **Reserved**: Reserved … F_H **Reserved**: Reserved
CHxSEL (x=1)	4*x+3:4*x	rw	**eGTM TIMn channel x input selection group B** This bit field selects which input is connected for TIMn channel x (group B). The input is either derived from a port pad or from an on-chip module. 0_H **None**: Safe state. No input connected to TIM2_INB1 1_H **Reserved**: Reserved … F_H **Reserved**: Reserved
CHxSEL (x=2)	4*x+3:4*x	rw	**eGTM TIMn channel x input selection group B** This bit field selects which input is connected for TIMn channel x (group B). The input is either derived from a port pad or from an on-chip module. 0_H **None**: Safe state. No input connected to TIM2_INB2 1_H **Reserved**: Reserved … F_H **Reserved**: Reserved
CHxSEL (x=3)	4*x+3:4*x	rw	**eGTM TIMn channel x input selection group B** This bit field selects which input is connected for TIMn channel x (group B). The input is either derived from a port pad or from an on-chip module. 0_H **None**: Safe state. No input connected to TIM2_INB3 1_H **Reserved**: Reserved … F_H **Reserved**: Reserved
CHxSEL (x=4)	4*x+3:4*x	rw	**eGTM TIMn channel x input selection group B** This bit field selects which input is connected for TIMn channel x (group B). The input is either derived from a port pad or from an on-chip module. 0_H **None**: Safe state. No input connected to TIM2_INB4 1_H **Reserved**: Reserved … F_H **Reserved**: Reserved

(table continues...)

(continued)

Field	Bits	Type	Description
CHxSEL (x=5)	4*x+3:4*x	rw	**eGTM TIMn channel x input selection group B** This bit field selects which input is connected for TIMn channel x (group B). The input is either derived from a port pad or from an on-chip module. 0_H **None**: Safe state. No input connected to TIM2_INB5 1_H **Reserved**: Reserved … F_H **Reserved**: Reserved
CHxSEL (x=6)	4*x+3:4*x	rw	**eGTM TIMn channel x input selection group B** This bit field selects which input is connected for TIMn channel x (group B). The input is either derived from a port pad or from an on-chip module. 0_H **None**: Safe state. No input connected to TIM2_INB6 1_H **Reserved**: Reserved … F_H **Reserved**: Reserved
CHxSEL (x=7)	4*x+3:4*x	rw	**eGTM TIMn channel x input selection group B** This bit field selects which input is connected for TIMn channel x (group B). The input is either derived from a port pad or from an on-chip module. 0_H **None**: Safe state. No input connected to TIM2_INB7 1_H **Reserved**: Reserved … F_H **Reserved**: Reserved

44.9.4.10 eGTM to Ports signal TOUT output selection register n (n=0)

For a description of this register, see 7579.

EGTM_TOUTSELn (n=0) Offset address: 061100_H+n*4

eGTM to Ports signal TOUT output selection register n Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	SEL5					SEL4					SEL3				
r	rw					rw					rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	SEL2					SEL1					SEL0				
r	rw					rw					rw				

Field	Bits	Type	Description
SELx (x=0)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT0 (P02.0) 01_H **CDTM0_DTM4_0**: Dead-time output of ATOM0_0 02_H **CDTM1_DTM4_0_N**: Inverted dead-time output of ATOM1_0 03_H **CDTM2_DTM4_0**: Dead-time output of ATOM2_0 04_H **CDTM0_DTM0_0**: Dead-time output of TOM0_0 05_H **CDTM1_DTM0_0_N**: Inverted dead-time output of TOM1_0 06_H **CDTM2_DTM0_0**: Dead-time output of TOM2_0 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM2_0**: Dead-time output of TOM0_8 11_H **CDTM1_DTM2_0**: Dead-time output of TOM1_8 12_H **CDTM0_DTM4_0**: Dead-time output of ATOM0_0 13_H **CDTM1_DTM4_0**: Dead-time output of ATOM1_0 14_H **CDTM0_DTM0_0**: Dead-time output of TOM0_0 15_H **CDTM1_DTM0_0**: Dead-time output of TOM1_0 16_H **CDTM0_DTM1_0**: Dead-time output of TOM0_4 17_H **CDTM1_DTM1_0**: Dead-time output of TOM1_4 18_H **CDTM0_DTM5_1_N**: Inverted dead-time output of ATOM0_5 19_H **CDTM1_DTM5_1_N**: Inverted dead-time output of ATOM1_5 $1A_H$ **CDTM2_DTM5_1_N**: Inverted dead-time output of ATOM2_5 $1B_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=1)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT1 (P02.1) 01_H **CDTM0_DTM4_0_N**: Inverted dead-time output of ATOM0_0 02_H **CDTM1_DTM4_0**: Dead-time output of ATOM1_0 03_H **CDTM2_DTM4_0_N**: Inverted dead-time output of ATOM2_0 04_H **CDTM0_DTM0_0_N**: Inverted dead-time output of TOM0_0 05_H **CDTM1_DTM0_0**: Dead-time output of TOM1_0 06_H **CDTM2_DTM0_0_N**: Inverted dead-time output of TOM2_0 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM2_1**: Dead-time output of TOM0_9 11_H **CDTM1_DTM2_1**: Dead-time output of TOM1_9 12_H **CDTM0_DTM4_1**: Dead-time output of ATOM0_1 13_H **CDTM1_DTM4_1**: Dead-time output of ATOM1_1 14_H **CDTM0_DTM0_1**: Dead-time output of TOM0_1 15_H **CDTM1_DTM0_1**: Dead-time output of TOM1_1 16_H **CDTM0_DTM1_0_N**: Inverted dead-time output of TOM0_4 17_H **CDTM1_DTM1_0_N**: Inverted dead-time output of TOM1_4 18_H **CDTM0_DTM5_2_N**: Inverted dead-time output of ATOM0_6 19_H **CDTM1_DTM5_2_N**: Inverted dead-time output of ATOM1_6 $1A_H$ **CDTM2_DTM5_2**: Dead-time output of ATOM2_6 $1B_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=2)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT2 (P02.2) 01_H **CDTM0_DTM4_1**: Dead-time output of ATOM0_1 02_H **CDTM1_DTM4_1_N**: Inverted dead-time output of ATOM1_1 03_H **CDTM2_DTM4_1**: Dead-time output of ATOM2_1 04_H **CDTM0_DTM0_1**: Dead-time output of TOM0_1 05_H **CDTM1_DTM0_1_N**: Inverted dead-time output of TOM1_1 06_H **CDTM2_DTM0_1**: Dead-time output of TOM2_1 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM2_2**: Dead-time output of TOM0_10 11_H **CDTM1_DTM2_2**: Dead-time output of TOM1_10 12_H **CDTM0_DTM4_2**: Dead-time output of ATOM0_2 13_H **CDTM1_DTM4_2**: Dead-time output of ATOM1_2 14_H **CDTM0_DTM0_2**: Dead-time output of TOM0_2 15_H **CDTM1_DTM0_2**: Dead-time output of TOM1_2 16_H **CDTM0_DTM1_1**: Dead-time output of TOM0_5 17_H **CDTM1_DTM1_1**: Dead-time output of TOM1_5 18_H **CDTM0_DTM5_3_N**: Inverted dead-time output of ATOM0_7 19_H **CDTM1_DTM5_3_N**: Inverted dead-time output of ATOM1_7 $1A_H$ **CDTM2_DTM5_3_N**: Inverted dead-time output of ATOM2_7 $1B_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=3)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT3 (P02.3) 01_H **CDTM0_DTM4_1_N**: Inverted dead-time output of ATOM0_1 02_H **CDTM1_DTM4_1**: Dead-time output of ATOM1_1 03_H **CDTM2_DTM4_1_N**: Inverted dead-time output of ATOM2_1 04_H **CDTM0_DTM0_1_N**: Inverted dead-time output of TOM0_1 05_H **CDTM1_DTM0_1**: Dead-time output of TOM1_1 06_H **CDTM2_DTM0_1_N**: Inverted dead-time output of TOM2_1 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM2_3**: Dead-time output of TOM0_11 11_H **CDTM1_DTM2_3**: Dead-time output of TOM1_11 12_H **CDTM0_DTM4_3**: Dead-time output of ATOM0_3 13_H **CDTM1_DTM4_3**: Dead-time output of ATOM1_3 14_H **CDTM0_DTM0_3**: Dead-time output of TOM0_3 15_H **CDTM1_DTM0_3**: Dead-time output of TOM1_3 16_H **CDTM0_DTM1_1_N**: Inverted dead-time output of TOM0_5 17_H **CDTM1_DTM1_1_N**: Inverted dead-time output of TOM1_5 18_H **CDTM0_DTM5_0_N**: Inverted dead-time output of ATOM0_4 19_H **CDTM1_DTM5_0_N**: Inverted dead-time output of ATOM1_4 $1A_H$ **CDTM2_DTM5_0_N**: Inverted dead-time output of ATOM2_4 $1B_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=4)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT4 (P02.4) 01_H **CDTM0_DTM4_2**: Dead-time output of ATOM0_2 02_H **CDTM1_DTM4_2_N**: Inverted dead-time output of ATOM1_2 03_H **CDTM2_DTM4_2**: Dead-time output of ATOM2_2 04_H **CDTM0_DTM0_2**: Dead-time output of TOM0_2 05_H **CDTM1_DTM0_2_N**: Inverted dead-time output of TOM1_2 06_H **CDTM2_DTM0_2**: Dead-time output of TOM2_2 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM3_0**: Dead-time output of TOM0_12 11_H **CDTM1_DTM3_0**: Dead-time output of TOM1_12 12_H **CDTM0_DTM5_0**: Dead-time output of ATOM0_4 13_H **CDTM1_DTM5_0**: Dead-time output of ATOM1_4 14_H **CDTM0_DTM0_1_N**: Inverted dead-time output of TOM0_1 15_H **CDTM1_DTM0_1_N**: Inverted dead-time output of TOM1_1 16_H **CDTM0_DTM1_2**: Dead-time output of TOM0_6 17_H **CDTM1_DTM1_2**: Dead-time output of TOM1_6 18_H **CDTM0_DTM4_1_N**: Inverted dead-time output of ATOM0_1 19_H **CDTM1_DTM4_1_N**: Inverted dead-time output of ATOM1_1 $1A_H$ **CDTM2_DTM4_1_N**: Inverted dead-time output of ATOM2_1 $1B_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=5)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT5 (P02.5) 01_H **CDTM0_DTM4_2_N**: Inverted dead-time output of ATOM0_2 02_H **CDTM1_DTM4_2**: Dead-time output of ATOM1_2 03_H **CDTM2_DTM4_2_N**: Inverted dead-time output of ATOM2_2 04_H **CDTM0_DTM0_2_N**: Inverted dead-time output of TOM0_2 05_H **CDTM1_DTM0_2**: Dead-time output of TOM1_2 06_H **CDTM2_DTM0_2_N**: Inverted dead-time output of TOM2_2 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM3_1**: Dead-time output of TOM0_13 11_H **CDTM1_DTM3_1**: Dead-time output of TOM1_13 12_H **CDTM0_DTM5_1**: Dead-time output of ATOM0_5 13_H **CDTM1_DTM5_1**: Dead-time output of ATOM1_5 14_H **CDTM0_DTM0_2_N**: Inverted dead-time output of TOM0_2 15_H **CDTM1_DTM0_2_N**: Inverted dead-time output of TOM1_2 16_H **CDTM0_DTM1_2_N**: Inverted dead-time output of TOM0_6 17_H **CDTM1_DTM1_2_N**: Inverted dead-time output of TOM1_6 18_H **CDTM0_DTM4_2_N**: Inverted dead-time output of ATOM0_2 19_H **CDTM1_DTM4_2_N**: Inverted dead-time output of ATOM1_2 $1A_H$ **CDTM2_DTM4_2**: Dead-time output of ATOM2_2 $1B_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	15, 31	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.11 eGTM to Ports signal TOUT output selection register n (n=1)

For a description of this register, see 7579.

EGTM_TOUTSELn (n=1) Offset address: 061100_H+n*4

eGTM to Ports signal TOUT output selection register n Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	SEL5					SEL4					SEL3				
r	rw					rw					rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	SEL2					SEL1					SEL0				
r	rw					rw					rw				

Field	Bits	Type	Description
SELx (x=0)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT6 (P02.6) 01_H **CDTM0_DTM4_3**: Dead-time output of ATOM0_3 02_H **CDTM1_DTM4_3_N**: Inverted dead-time output of ATOM1_3 03_H **CDTM2_DTM4_3**: Dead-time output of ATOM2_3 04_H **CDTM0_DTM0_3**: Dead-time output of TOM0_3 05_H **CDTM1_DTM0_3_N**: Inverted dead-time output of TOM1_3 06_H **CDTM2_DTM0_3**: Dead-time output of TOM2_3 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM3_2**: Dead-time output of TOM0_14 11_H **CDTM1_DTM3_2**: Dead-time output of TOM1_14 12_H **CDTM0_DTM5_2**: Dead-time output of ATOM0_6 13_H **CDTM1_DTM5_2**: Dead-time output of ATOM1_6 14_H **CDTM0_DTM0_3_N**: Inverted dead-time output of TOM0_3 15_H **CDTM1_DTM0_3_N**: Inverted dead-time output of TOM1_3 16_H **CDTM0_DTM1_3**: Dead-time output of TOM0_7 17_H **CDTM1_DTM1_3**: Dead-time output of TOM1_7 18_H **CDTM0_DTM4_3_N**: Inverted dead-time output of ATOM0_3 19_H **CDTM1_DTM4_3_N**: Inverted dead-time output of ATOM1_3 $1A_H$ **CDTM2_DTM4_3_N**: Inverted dead-time output of ATOM2_3 $1B_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=1)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT7 (P02.7) 01_H **CDTM0_DTM4_3_N**: Inverted dead-time output of ATOM0_3 02_H **CDTM1_DTM4_3**: Dead-time output of ATOM1_3 03_H **CDTM2_DTM4_3_N**: Inverted dead-time output of ATOM2_3 04_H **CDTM0_DTM0_3_N**: Inverted dead-time output of TOM0_3 05_H **CDTM1_DTM0_3**: Dead-time output of TOM1_3 06_H **CDTM2_DTM0_3_N**: Inverted dead-time output of TOM2_3 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM3_3**: Dead-time output of TOM0_15 11_H **CDTM1_DTM3_3**: Dead-time output of TOM1_15 12_H **CDTM0_DTM5_3**: Dead-time output of ATOM0_7 13_H **CDTM1_DTM5_3**: Dead-time output of ATOM1_7 14_H **CDTM0_DTM0_0_N**: Inverted dead-time output of TOM0_0 15_H **CDTM1_DTM0_0_N**: Inverted dead-time output of TOM1_0 16_H **CDTM0_DTM1_3_N**: Inverted dead-time output of TOM0_7 17_H **CDTM1_DTM1_3_N**: Inverted dead-time output of TOM1_7 18_H **CDTM0_DTM4_0_N**: Inverted dead-time output of ATOM0_0 19_H **CDTM1_DTM4_0_N**: Inverted dead-time output of ATOM1_0 $1A_H$ **CDTM2_DTM4_0_N**: Inverted dead-time output of ATOM2_0 $1B_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=2)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT8 (P02.8) 01_H **CDTM0_DTM5_0**: Dead-time output of ATOM0_4 02_H **CDTM1_DTM5_0_N**: Inverted dead-time output of ATOM1_4 03_H **CDTM2_DTM5_0**: Dead-time output of ATOM2_4 04_H **CDTM0_DTM1_0**: Dead-time output of TOM0_4 05_H **CDTM1_DTM1_0_N**: Inverted dead-time output of TOM1_4 06_H **CDTM2_DTM1_0**: Dead-time output of TOM2_4 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM2_0**: Dead-time output of TOM0_8 11_H **CDTM1_DTM0_0**: Dead-time output of TOM1_0 12_H **CDTM0_DTM4_0**: Dead-time output of ATOM0_0 13_H **CDTM1_DTM4_0**: Dead-time output of ATOM1_0 14_H **Reserved**: Reserved, do not use 15_H **Reserved**: Reserved, do not use 16_H **CDTM0_DTM1_0_N**: Inverted dead-time output of TOM0_4 17_H **CDTM1_DTM1_0_N**: Inverted dead-time output of TOM1_4 18_H **CDTM0_DTM5_1_N**: Inverted dead-time output of ATOM0_5 19_H **CDTM1_DTM5_1_N**: Inverted dead-time output of ATOM1_5 $1A_H$ **CDTM2_DTM5_1_N**: Inverted dead-time output of ATOM2_5 $1B_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=3)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT9 (P00.0) 01_H **CDTM0_DTM5_0_N**: Inverted dead-time output of ATOM0_4 02_H **CDTM1_DTM5_0**: Dead-time output of ATOM1_4 03_H **CDTM2_DTM5_0_N**: Inverted dead-time output of ATOM2_4 04_H **CDTM0_DTM1_0_N**: Inverted dead-time output of TOM0_4 05_H **CDTM1_DTM1_0**: Dead-time output of TOM1_4 06_H **CDTM2_DTM1_0_N**: Inverted dead-time output of TOM2_4 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM2_0**: Dead-time output of TOM0_8 11_H **CDTM1_DTM0_0**: Dead-time output of TOM1_0 12_H **CDTM0_DTM4_0**: Dead-time output of ATOM0_0 13_H **CDTM1_DTM4_0**: Dead-time output of ATOM1_0 14_H **CDTM0_DTM0_0_N**: Inverted dead-time output of TOM0_0 15_H **CDTM1_DTM1_1_N**: Inverted dead-time output of TOM1_5 16_H **CDTM0_DTM1_0**: Dead-time output of TOM0_4 17_H **CDTM1_DTM1_0**: Dead-time output of TOM1_4 18_H **CDTM0_DTM5_1_N**: Inverted dead-time output of ATOM0_5 19_H **CDTM1_DTM5_1_N**: Inverted dead-time output of ATOM1_5 $1A_H$ **CDTM2_DTM4_1_N**: Inverted dead-time output of ATOM2_1 $1B_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=4)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT10 (P00.1) 01_H **CDTM0_DTM5_1**: Dead-time output of ATOM0_5 02_H **CDTM1_DTM5_1_N**: Inverted dead-time output of ATOM1_5 03_H **CDTM2_DTM5_1**: Dead-time output of ATOM2_5 04_H **CDTM0_DTM1_1**: Dead-time output of TOM0_5 05_H **CDTM1_DTM1_1_N**: Inverted dead-time output of TOM1_5 06_H **CDTM2_DTM1_1**: Dead-time output of TOM2_5 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM2_1**: Dead-time output of TOM0_9 11_H **CDTM1_DTM0_1**: Dead-time output of TOM1_1 12_H **CDTM0_DTM4_1**: Dead-time output of ATOM0_1 13_H **CDTM1_DTM4_1**: Dead-time output of ATOM1_1 14_H **CDTM0_DTM0_1_N**: Inverted dead-time output of TOM0_1 15_H **CDTM1_DTM1_2_N**: Inverted dead-time output of TOM1_6 16_H **CDTM0_DTM1_0_N**: Inverted dead-time output of TOM0_4 17_H **CDTM1_DTM1_0_N**: Inverted dead-time output of TOM1_4 18_H **CDTM0_DTM5_2_N**: Inverted dead-time output of ATOM0_6 19_H **CDTM1_DTM5_2_N**: Inverted dead-time output of ATOM1_6 $1A_H$ **CDTM2_DTM4_2_N**: Inverted dead-time output of ATOM2_2 $1B_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=5)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT11 (P00.2) 01_H **CDTM0_DTM5_1_N**: Inverted dead-time output of ATOM0_5 02_H **CDTM1_DTM5_1**: Dead-time output of ATOM1_5 03_H **CDTM2_DTM5_1_N**: Inverted dead-time output of ATOM2_5 04_H **CDTM0_DTM1_1_N**: Inverted dead-time output of TOM0_5 05_H **CDTM1_DTM1_1**: Dead-time output of TOM1_5 06_H **CDTM2_DTM1_1_N**: Inverted dead-time output of TOM2_5 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM2_1**: Dead-time output of TOM0_9 11_H **CDTM1_DTM0_1**: Dead-time output of TOM1_1 12_H **CDTM0_DTM4_1**: Dead-time output of ATOM0_1 13_H **CDTM1_DTM4_1**: Dead-time output of ATOM1_1 14_H **CDTM0_DTM0_2_N**: Inverted dead-time output of TOM0_2 15_H **CDTM1_DTM1_2_N**: Inverted dead-time output of TOM1_6 16_H **CDTM0_DTM1_1**: Dead-time output of TOM0_5 17_H **CDTM1_DTM1_1**: Dead-time output of TOM1_5 18_H **CDTM0_DTM5_2_N**: Inverted dead-time output of ATOM0_6 19_H **CDTM1_DTM5_2_N**: Inverted dead-time output of ATOM1_6 $1A_H$ **CDTM2_DTM4_3_N**: Inverted dead-time output of ATOM2_3 $1B_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	15, 31	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.12 eGTM to Ports signal TOUT output selection register n (n=2)

For a description of this register, see 7579.

EGTM_TOUTSELn (n=2) Offset address: 061100_H+n*4

eGTM to Ports signal TOUT output selection register n Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	SEL5					SEL4					SEL3				
r	rw					rw					rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	SEL2					SEL1					SEL0				
r	rw					rw					rw				

Field	Bits	Type	Description
SELx (x=0)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT12 (P00.3) 01_H **CDTM0_DTM5_2**: Dead-time output of ATOM0_6 02_H **CDTM1_DTM5_2_N**: Inverted dead-time output of ATOM1_6 03_H **CDTM2_DTM5_2**: Dead-time output of ATOM2_6 04_H **CDTM0_DTM1_2**: Dead-time output of TOM0_6 05_H **CDTM1_DTM1_2_N**: Inverted dead-time output of TOM1_6 06_H **CDTM2_DTM1_2**: Dead-time output of TOM2_6 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM2_2**: Dead-time output of TOM0_10 11_H **CDTM1_DTM0_2**: Dead-time output of TOM1_2 12_H **CDTM0_DTM4_2**: Dead-time output of ATOM0_2 13_H **CDTM1_DTM4_2**: Dead-time output of ATOM1_2 14_H **CDTM0_DTM0_3_N**: Inverted dead-time output of TOM0_3 15_H **CDTM1_DTM1_3_N**: Inverted dead-time output of TOM1_7 16_H **CDTM0_DTM1_1_N**: Inverted dead-time output of TOM0_5 17_H **CDTM1_DTM1_1_N**: Inverted dead-time output of TOM1_5 18_H **CDTM0_DTM5_3_N**: Inverted dead-time output of ATOM0_7 19_H **CDTM1_DTM5_3_N**: Inverted dead-time output of ATOM1_7 $1A_H$ **CDTM2_DTM4_0_N**: Inverted dead-time output of ATOM2_0 $1B_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=1)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT13 (P00.4) 01_H **CDTM0_DTM5_2_N**: Inverted dead-time output of ATOM0_6 02_H **CDTM1_DTM5_2**: Dead-time output of ATOM1_6 03_H **CDTM2_DTM5_2_N**: Inverted dead-time output of ATOM2_6 04_H **CDTM0_DTM1_2_N**: Inverted dead-time output of TOM0_6 05_H **CDTM1_DTM1_2**: Dead-time output of TOM1_6 06_H **CDTM2_DTM1_2_N**: Inverted dead-time output of TOM2_6 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM2_3**: Dead-time output of TOM0_11 11_H **CDTM1_DTM0_3**: Dead-time output of TOM1_3 12_H **CDTM0_DTM4_3**: Dead-time output of ATOM0_3 13_H **CDTM1_DTM4_3**: Dead-time output of ATOM1_3 14_H **Reserved**: Reserved, do not use 15_H **CDTM1_DTM1_0_N**: Inverted dead-time output of TOM1_4 16_H **CDTM0_DTM1_2**: Dead-time output of TOM0_6 17_H **CDTM1_DTM1_2**: Dead-time output of TOM1_6 18_H **CDTM0_DTM5_0_N**: Inverted dead-time output of ATOM0_4 19_H **CDTM1_DTM5_0_N**: Inverted dead-time output of ATOM1_4 $1A_H$ **CDTM2_DTM5_0_N**: Inverted dead-time output of ATOM2_4 $1B_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=2)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT14 (P00.5) 01_H **CDTM0_DTM5_3**: Dead-time output of ATOM0_7 02_H **CDTM1_DTM5_3_N**: Inverted dead-time output of ATOM1_7 03_H **CDTM2_DTM5_3**: Dead-time output of ATOM2_7 04_H **CDTM0_DTM1_3**: Dead-time output of TOM0_7 05_H **CDTM1_DTM1_3_N**: Inverted dead-time output of TOM1_7 06_H **CDTM2_DTM1_3**: Dead-time output of TOM2_7 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM3_0**: Dead-time output of TOM0_12 11_H **CDTM1_DTM1_0**: Dead-time output of TOM1_4 12_H **CDTM0_DTM5_0**: Dead-time output of ATOM0_4 13_H **CDTM1_DTM5_0**: Dead-time output of ATOM1_4 14_H **CDTM0_DTM0_1_N**: Inverted dead-time output of TOM0_1 15_H **CDTM1_DTM0_1_N**: Inverted dead-time output of TOM1_1 16_H **CDTM0_DTM1_2_N**: Inverted dead-time output of TOM0_6 17_H **CDTM1_DTM1_2_N**: Inverted dead-time output of TOM1_6 18_H **CDTM0_DTM4_1_N**: Inverted dead-time output of ATOM0_1 19_H **CDTM1_DTM4_1_N**: Inverted dead-time output of ATOM1_1 $1A_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=3)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT15 (P00.6) 01_H **CDTM0_DTM5_3_N**: Inverted dead-time output of ATOM0_7 02_H **CDTM1_DTM5_3**: Dead-time output of ATOM1_7 03_H **CDTM2_DTM5_3_N**: Inverted dead-time output of ATOM2_7 04_H **CDTM0_DTM1_3_N**: Inverted dead-time output of TOM0_7 05_H **CDTM1_DTM1_3**: Dead-time output of TOM1_7 06_H **CDTM2_DTM1_3_N**: Inverted dead-time output of TOM2_7 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM3_1**: Dead-time output of TOM0_13 11_H **CDTM1_DTM1_1**: Dead-time output of TOM1_5 12_H **CDTM0_DTM5_1**: Dead-time output of ATOM0_5 13_H **CDTM1_DTM5_1**: Dead-time output of ATOM1_5 14_H **CDTM0_DTM0_2_N**: Inverted dead-time output of TOM0_2 15_H **CDTM1_DTM0_2_N**: Inverted dead-time output of TOM1_2 16_H **CDTM0_DTM1_3**: Dead-time output of TOM0_7 17_H **CDTM1_DTM1_3**: Dead-time output of TOM1_7 18_H **CDTM0_DTM4_2_N**: Inverted dead-time output of ATOM0_2 19_H **CDTM1_DTM4_2_N**: Inverted dead-time output of ATOM1_2 $1A_H$ **CDTM2_DTM4_2_N**: Inverted dead-time output of ATOM2_2 $1B_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=4)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT16 (P00.7) 01_H **CDTM0_DTM4_0**: Dead-time output of ATOM0_0 02_H **CDTM1_DTM4_0_N**: Inverted dead-time output of ATOM1_0 03_H **CDTM2_DTM4_0**: Dead-time output of ATOM2_0 04_H **CDTM0_DTM2_0**: Dead-time output of TOM0_8 05_H **CDTM1_DTM2_0_N**: Inverted dead-time output of TOM1_8 06_H **CDTM2_DTM2_0**: Dead-time output of TOM2_8 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM3_2**: Dead-time output of TOM0_14 11_H **CDTM1_DTM1_2**: Dead-time output of TOM1_6 12_H **CDTM0_DTM5_2**: Dead-time output of ATOM0_6 13_H **CDTM1_DTM5_2**: Dead-time output of ATOM1_6 14_H **CDTM0_DTM0_3_N**: Inverted dead-time output of TOM0_3 15_H **CDTM1_DTM0_3_N**: Inverted dead-time output of TOM1_3 16_H **CDTM0_DTM1_3_N**: Inverted dead-time output of TOM0_7 17_H **CDTM1_DTM1_3_N**: Inverted dead-time output of TOM1_7 18_H **CDTM0_DTM4_3_N**: Inverted dead-time output of ATOM0_3 19_H **CDTM1_DTM4_3_N**: Inverted dead-time output of ATOM1_3 $1A_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=5)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT17 (P00.8) 01_H **CDTM0_DTM4_0_N**: Inverted dead-time output of ATOM0_0 02_H **CDTM1_DTM4_0**: Dead-time output of ATOM1_0 03_H **CDTM2_DTM4_0_N**: Inverted dead-time output of ATOM2_0 04_H **CDTM0_DTM2_0_N**: Inverted dead-time output of TOM0_8 05_H **CDTM1_DTM2_0**: Dead-time output of TOM1_8 06_H **CDTM2_DTM2_0_N**: Inverted dead-time output of TOM2_8 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM3_3**: Dead-time output of TOM0_15 11_H **CDTM1_DTM1_3**: Dead-time output of TOM1_7 12_H **CDTM0_DTM5_3**: Dead-time output of ATOM0_7 13_H **CDTM1_DTM5_3**: Dead-time output of ATOM1_7 14_H **CDTM0_DTM0_0_N**: Inverted dead-time output of TOM0_0 15_H **CDTM1_DTM0_0_N**: Inverted dead-time output of TOM1_0 16_H **CDTM0_DTM0_3**: Dead-time output of TOM0_3 17_H **Reserved**: Reserved, do not use 18_H **CDTM0_DTM4_0_N**: Inverted dead-time output of ATOM0_0 19_H **CDTM1_DTM4_0_N**: Inverted dead-time output of ATOM1_0 $1A_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	15, 31	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.13 eGTM to Ports signal TOUT output selection register n (n=3)

For a description of this register, see 7579.

EGTM_TOUTSELn (n=3) Offset address: 061100_H+n*4

eGTM to Ports signal TOUT output selection register n Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	SEL5					SEL4					SEL3				
r	rw					rw					rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	SEL2					SEL1					SEL0				
r	rw					rw					rw				

Field	Bits	Type	Description
SELx (x=0)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT18 (P00.9) 01_H **CDTM0_DTM4_1**: Dead-time output of ATOM0_1 02_H **CDTM1_DTM4_1_N**: Inverted dead-time output of ATOM1_1 03_H **CDTM2_DTM4_1**: Dead-time output of ATOM2_1 04_H **CDTM0_DTM2_1**: Dead-time output of TOM0_9 05_H **CDTM1_DTM2_1_N**: Inverted dead-time output of TOM1_9 06_H **CDTM2_DTM2_1**: Dead-time output of TOM2_9 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM0_0**: Dead-time output of TOM0_0 11_H **CDTM1_DTM0_0**: Dead-time output of TOM1_0 12_H **CDTM2_DTM4_0**: Dead-time output of ATOM2_0 13_H **Reserved**: Reserved, do not use 14_H **CDTM0_DTM1_1_N**: Inverted dead-time output of TOM0_5 15_H **CDTM1_DTM1_1_N**: Inverted dead-time output of TOM1_5 16_H **CDTM1_DTM5_1_N**: Inverted dead-time output of ATOM1_5 17_H **CDTM2_DTM5_1_N**: Inverted dead-time output of ATOM2_5 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
SELx (x=1)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT19 (P00.10) 01_H **CDTM0_DTM4_1_N**: Inverted dead-time output of ATOM0_1 02_H **CDTM1_DTM4_1**: Dead-time output of ATOM1_1 03_H **CDTM2_DTM4_1_N**: Inverted dead-time output of ATOM2_1 04_H **CDTM0_DTM2_1_N**: Inverted dead-time output of TOM0_9 05_H **CDTM1_DTM2_1**: Dead-time output of TOM1_9 06_H **CDTM2_DTM2_1_N**: Inverted dead-time output of TOM2_9 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM0_1**: Dead-time output of TOM0_1 11_H **CDTM1_DTM0_1**: Dead-time output of TOM1_1 12_H **CDTM2_DTM4_1**: Dead-time output of ATOM2_1 13_H **Reserved**: Reserved, do not use 14_H **CDTM0_DTM1_2_N**: Inverted dead-time output of TOM0_6 15_H **CDTM1_DTM1_2_N**: Inverted dead-time output of TOM1_6 16_H **CDTM1_DTM5_2_N**: Inverted dead-time output of ATOM1_6 17_H **CDTM2_DTM5_2_N**: Inverted dead-time output of ATOM2_6 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=2)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT20 (P00.11) 01_H **CDTM0_DTM4_2**: Dead-time output of ATOM0_2 02_H **CDTM1_DTM4_2_N**: Inverted dead-time output of ATOM1_2 03_H **CDTM2_DTM4_2**: Dead-time output of ATOM2_2 04_H **CDTM0_DTM2_2**: Dead-time output of TOM0_10 05_H **CDTM1_DTM2_2_N**: Inverted dead-time output of TOM1_10 06_H **CDTM2_DTM2_2**: Dead-time output of TOM2_10 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM0_2**: Dead-time output of TOM0_2 11_H **CDTM1_DTM0_2**: Dead-time output of TOM1_2 12_H **CDTM2_DTM4_2**: Dead-time output of ATOM2_2 13_H **Reserved**: Reserved, do not use 14_H **CDTM0_DTM1_3_N**: Inverted dead-time output of TOM0_7 15_H **CDTM1_DTM1_3_N**: Inverted dead-time output of TOM1_7 16_H **CDTM1_DTM5_3_N**: Inverted dead-time output of ATOM1_7 17_H **CDTM2_DTM5_3_N**: Inverted dead-time output of ATOM2_7 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
SELx (x=3)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT21 (P00.12) 01_H **CDTM0_DTM4_2_N**: Inverted dead-time output of ATOM0_2 02_H **CDTM1_DTM4_2**: Dead-time output of ATOM1_2 03_H **CDTM2_DTM4_2_N**: Inverted dead-time output of ATOM2_2 04_H **CDTM0_DTM2_2_N**: Inverted dead-time output of TOM0_10 05_H **CDTM1_DTM2_2**: Dead-time output of TOM1_10 06_H **CDTM2_DTM2_2_N**: Inverted dead-time output of TOM2_10 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM0_3**: Dead-time output of TOM0_3 11_H **CDTM1_DTM0_3**: Dead-time output of TOM1_3 12_H **CDTM2_DTM4_3**: Dead-time output of ATOM2_3 13_H **Reserved**: Reserved, do not use 14_H **CDTM0_DTM1_0_N**: Inverted dead-time output of TOM0_4 15_H **CDTM1_DTM1_0_N**: Inverted dead-time output of TOM1_4 16_H **CDTM1_DTM5_0_N**: Inverted dead-time output of ATOM1_4 17_H **CDTM2_DTM5_0_N**: Inverted dead-time output of ATOM2_4 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=4)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT22 (P33.0) 01_H **CDTM0_DTM4_3**: Dead-time output of ATOM0_3 02_H **CDTM1_DTM4_3_N**: Inverted dead-time output of ATOM1_3 03_H **CDTM2_DTM4_3**: Dead-time output of ATOM2_3 04_H **CDTM0_DTM2_3**: Dead-time output of TOM0_11 05_H **CDTM1_DTM2_3_N**: Inverted dead-time output of TOM1_11 06_H **CDTM2_DTM2_3**: Dead-time output of TOM2_11 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM1_0**: Dead-time output of TOM0_4 11_H **CDTM1_DTM1_0**: Dead-time output of TOM1_4 12_H **CDTM2_DTM5_0**: Dead-time output of ATOM2_4 13_H **Reserved**: Reserved, do not use 14_H **CDTM0_DTM0_1_N**: Inverted dead-time output of TOM0_1 15_H **CDTM1_DTM0_1_N**: Inverted dead-time output of TOM1_1 16_H **CDTM1_DTM4_1_N**: Inverted dead-time output of ATOM1_1 17_H **Reserved**: Reserved, do not use 18_H **CDTM2_DTM4_1_N**: Inverted dead-time output of ATOM2_1 19_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=5)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT23 (P33.1) 01_H **CDTM0_DTM4_3_N**: Inverted dead-time output of ATOM0_3 02_H **CDTM1_DTM4_3**: Dead-time output of ATOM1_3 03_H **CDTM2_DTM4_3_N**: Inverted dead-time output of ATOM2_3 04_H **CDTM0_DTM2_3_N**: Inverted dead-time output of TOM0_11 05_H **CDTM1_DTM2_3**: Dead-time output of TOM1_11 06_H **CDTM2_DTM2_3_N**: Inverted dead-time output of TOM2_11 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM1_1**: Dead-time output of TOM0_5 11_H **CDTM1_DTM1_1**: Dead-time output of TOM1_5 12_H **CDTM0_DTM5_1**: Dead-time output of ATOM0_5 13_H **CDTM1_DTM5_1**: Dead-time output of ATOM1_5 14_H **CDTM0_DTM0_2_N**: Inverted dead-time output of TOM0_2 15_H **CDTM1_DTM0_2_N**: Inverted dead-time output of TOM1_2 16_H **CDTM1_DTM4_2**: Dead-time output of ATOM1_2 17_H **Reserved**: Reserved, do not use 18_H **CDTM2_DTM4_2_N**: Inverted dead-time output of ATOM2_2 19_H **Reserved**: Reserved, do not use $1A_H$ **CDTM0_DTM4_2_N**: Inverted dead-time output of ATOM0_2 $1B_H$ **CDTM1_DTM4_2_N**: Inverted dead-time output of ATOM1_2 $1C_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	15, 31	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.14 eGTM to Ports signal TOUT output selection register n (n=4)

For a description of this register, see 7579.

EGTM_TOUTSELn (n=4) Offset address: 061100_H+n*4

eGTM to Ports signal TOUT output selection register n Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	SEL5					SEL4					SEL3				
r	rw					rw					rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	SEL2					SEL1					SEL0				
r	rw					rw					rw				

Field	Bits	Type	Description
SELx (x=0)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT24 (P33.2) 01_H **CDTM0_DTM5_0**: Dead-time output of ATOM0_4 02_H **CDTM1_DTM5_0_N**: Inverted dead-time output of ATOM1_4 03_H **CDTM2_DTM5_0**: Dead-time output of ATOM2_4 04_H **CDTM0_DTM3_0**: Dead-time output of TOM0_12 05_H **CDTM1_DTM3_0_N**: Inverted dead-time output of TOM1_12 06_H **CDTM2_DTM3_0**: Dead-time output of TOM2_12 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM1_2**: Dead-time output of TOM0_6 11_H **CDTM1_DTM1_2**: Dead-time output of TOM1_6 12_H **CDTM0_DTM5_2**: Dead-time output of ATOM0_6 13_H **CDTM1_DTM5_2**: Dead-time output of ATOM1_6 14_H **CDTM0_DTM0_3_N**: Inverted dead-time output of TOM0_3 15_H **CDTM1_DTM0_3_N**: Inverted dead-time output of TOM1_3 16_H **CDTM1_DTM4_3**: Dead-time output of ATOM1_3 17_H **Reserved**: Reserved, do not use 18_H **CDTM2_DTM4_3_N**: Inverted dead-time output of ATOM2_3 19_H **Reserved**: Reserved, do not use $1A_H$ **CDTM0_DTM4_3_N**: Inverted dead-time output of ATOM0_3 $1B_H$ **CDTM1_DTM4_3_N**: Inverted dead-time output of ATOM1_3 $1C_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=1)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT25 (P33.3) 01_H **CDTM0_DTM5_0_N**: Inverted dead-time output of ATOM0_4 02_H **CDTM1_DTM5_0**: Dead-time output of ATOM1_4 03_H **CDTM2_DTM5_0_N**: Inverted dead-time output of ATOM2_4 04_H **CDTM0_DTM3_0_N**: Inverted dead-time output of TOM0_12 05_H **CDTM1_DTM3_0**: Dead-time output of TOM1_12 06_H **CDTM2_DTM3_0_N**: Inverted dead-time output of TOM2_12 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM1_3**: Dead-time output of TOM0_7 11_H **CDTM1_DTM1_3**: Dead-time output of TOM1_7 12_H **CDTM0_DTM5_3**: Dead-time output of ATOM0_7 13_H **CDTM1_DTM5_3**: Dead-time output of ATOM1_7 14_H **CDTM0_DTM0_0_N**: Inverted dead-time output of TOM0_0 15_H **CDTM1_DTM0_0_N**: Inverted dead-time output of TOM1_0 16_H **CDTM1_DTM4_0_N**: Inverted dead-time output of ATOM1_0 17_H **CDTM2_DTM4_0_N**: Inverted dead-time output of ATOM2_0 18_H **CDTM2_DTM4_0_N**: Inverted dead-time output of ATOM2_0 19_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=2)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT26 (P33.4) 01_H **CDTM0_DTM5_1**: Dead-time output of ATOM0_5 02_H **CDTM1_DTM5_1_N**: Inverted dead-time output of ATOM1_5 03_H **CDTM2_DTM5_1**: Dead-time output of ATOM2_5 04_H **CDTM0_DTM3_1**: Dead-time output of TOM0_13 05_H **CDTM1_DTM3_1_N**: Inverted dead-time output of TOM1_13 06_H **CDTM2_DTM3_1**: Dead-time output of TOM2_13 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM0_0**: Dead-time output of TOM0_0 11_H **CDTM1_DTM0_0**: Dead-time output of TOM1_0 12_H **CDTM2_DTM4_0**: Dead-time output of ATOM2_0 13_H **Reserved**: Reserved, do not use 14_H **CDTM0_DTM1_1_N**: Inverted dead-time output of TOM0_5 15_H **CDTM1_DTM1_1_N**: Inverted dead-time output of TOM1_5 16_H **CDTM2_DTM5_1**: Dead-time output of ATOM2_5 17_H **Reserved**: Reserved, do not use 18_H **CDTM2_DTM5_1_N**: Inverted dead-time output of ATOM2_5 19_H **Reserved**: Reserved, do not use $1A_H$ **CDTM0_DTM5_1_N**: Inverted dead-time output of ATOM0_5 $1B_H$ **CDTM1_DTM5_1_N**: Inverted dead-time output of ATOM1_5 $1C_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=3)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT27 (P33.5) 01_H **CDTM0_DTM5_1_N**: Inverted dead-time output of ATOM0_5 02_H **CDTM1_DTM5_1**: Dead-time output of ATOM1_5 03_H **CDTM2_DTM5_1_N**: Inverted dead-time output of ATOM2_5 04_H **CDTM0_DTM3_1_N**: Inverted dead-time output of TOM0_13 05_H **CDTM1_DTM3_1**: Dead-time output of TOM1_13 06_H **CDTM2_DTM3_1_N**: Inverted dead-time output of TOM2_13 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM0_1**: Dead-time output of TOM0_1 11_H **CDTM1_DTM0_1**: Dead-time output of TOM1_1 12_H **CDTM2_DTM4_1**: Dead-time output of ATOM2_1 13_H **Reserved**: Reserved, do not use 14_H **CDTM0_DTM1_2_N**: Inverted dead-time output of TOM0_6 15_H **CDTM1_DTM1_2_N**: Inverted dead-time output of TOM1_6 16_H **CDTM0_DTM1_1**: Dead-time output of TOM0_5 17_H **CDTM1_DTM1_1**: Dead-time output of TOM1_5 18_H **CDTM2_DTM5_2_N**: Inverted dead-time output of ATOM2_6 19_H **Reserved**: Reserved, do not use $1A_H$ **CDTM0_DTM5_2_N**: Inverted dead-time output of ATOM0_6 $1B_H$ **CDTM1_DTM5_2_N**: Inverted dead-time output of ATOM1_6 $1C_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=4)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT28 (P33.6) 01_H **CDTM0_DTM5_2**: Dead-time output of ATOM0_6 02_H **CDTM1_DTM5_2_N**: Inverted dead-time output of ATOM1_6 03_H **CDTM2_DTM5_2**: Dead-time output of ATOM2_6 04_H **CDTM0_DTM3_2**: Dead-time output of TOM0_14 05_H **CDTM1_DTM3_2_N**: Inverted dead-time output of TOM1_14 06_H **CDTM2_DTM3_2**: Dead-time output of TOM2_14 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM0_2**: Dead-time output of TOM0_2 11_H **CDTM1_DTM0_2**: Dead-time output of TOM1_2 12_H **CDTM2_DTM4_2**: Dead-time output of ATOM2_2 13_H **Reserved**: Reserved, do not use 14_H **CDTM0_DTM1_3_N**: Inverted dead-time output of TOM0_7 15_H **CDTM1_DTM1_3_N**: Inverted dead-time output of TOM1_7 16_H **CDTM0_DTM1_1_N**: Inverted dead-time output of TOM0_5 17_H **CDTM1_DTM1_1_N**: Inverted dead-time output of TOM1_5 18_H **CDTM2_DTM5_3_N**: Inverted dead-time output of ATOM2_7 19_H **Reserved**: Reserved, do not use $1A_H$ **CDTM0_DTM5_3_N**: Inverted dead-time output of ATOM0_7 $1B_H$ **CDTM1_DTM5_3_N**: Inverted dead-time output of ATOM1_7 $1C_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=5)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT29 (P33.7) 01_H **CDTM0_DTM5_2_N**: Inverted dead-time output of ATOM0_6 02_H **CDTM1_DTM5_2**: Dead-time output of ATOM1_6 03_H **CDTM2_DTM5_2_N**: Inverted dead-time output of ATOM2_6 04_H **CDTM0_DTM3_2_N**: Inverted dead-time output of TOM0_14 05_H **CDTM1_DTM3_2**: Dead-time output of TOM1_14 06_H **CDTM2_DTM3_2_N**: Inverted dead-time output of TOM2_14 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM0_3**: Dead-time output of TOM0_3 11_H **CDTM1_DTM0_3**: Dead-time output of TOM1_3 12_H **CDTM2_DTM4_3**: Dead-time output of ATOM2_3 13_H **Reserved**: Reserved, do not use 14_H **CDTM0_DTM1_0_N**: Inverted dead-time output of TOM0_4 15_H **CDTM1_DTM1_0_N**: Inverted dead-time output of TOM1_4 16_H **CDTM0_DTM1_2**: Dead-time output of TOM0_6 17_H **CDTM1_DTM1_2**: Dead-time output of TOM1_6 18_H **CDTM2_DTM5_0_N**: Inverted dead-time output of ATOM2_4 19_H **Reserved**: Reserved, do not use $1A_H$ **CDTM0_DTM5_0_N**: Inverted dead-time output of ATOM0_4 $1B_H$ **CDTM1_DTM5_0_N**: Inverted dead-time output of ATOM1_4 $1C_H$ **CDTM0_DTM5_0**: Dead-time output of ATOM0_4 $1D_H$ **CDTM1_DTM5_0**: Dead-time output of ATOM1_4 $1E_H$ **Reserved**: Reserved, do not use $1F_H$ **Reserved**: Reserved, do not use
0	15, 31	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.15 eGTM to Ports signal TOUT output selection register n (n=5)

For a description of this register, see 7579.

EGTM_TOUTSELn (n=5) Offset address: 061100_H+n*4

eGTM to Ports signal TOUT output selection register n Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	SEL5					SEL4					SEL3				
r	rw					rw					rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	SEL2					SEL1					SEL0				
r	rw					rw					rw				

Field	Bits	Type	Description
SELx (x=0)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT30 (P33.8) 01_H **CDTM0_DTM5_3**: Dead-time output of ATOM0_7 02_H **CDTM1_DTM5_3_N**: Inverted dead-time output of ATOM1_7 03_H **CDTM2_DTM5_3**: Dead-time output of ATOM2_7 04_H **CDTM0_DTM3_3**: Dead-time output of TOM0_15 05_H **CDTM1_DTM3_3_N**: Inverted dead-time output of TOM1_15 06_H **CDTM2_DTM3_3**: Dead-time output of TOM2_15 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM1_0**: Dead-time output of TOM0_4 11_H **CDTM1_DTM1_0**: Dead-time output of TOM1_4 12_H **CDTM2_DTM5_0**: Dead-time output of ATOM2_4 13_H **Reserved**: Reserved, do not use 14_H **CDTM0_DTM0_1_N**: Inverted dead-time output of TOM0_1 15_H **CDTM1_DTM0_1_N**: Inverted dead-time output of TOM1_1 16_H **CDTM0_DTM1_2_N**: Inverted dead-time output of TOM0_6 17_H **CDTM1_DTM1_2_N**: Inverted dead-time output of TOM1_6 18_H **CDTM2_DTM4_1_N**: Inverted dead-time output of ATOM2_1 19_H **Reserved**: Reserved, do not use $1A_H$ **CDTM0_DTM4_1_N**: Inverted dead-time output of ATOM0_1 $1B_H$ **CDTM1_DTM4_1_N**: Inverted dead-time output of ATOM1_1 $1C_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=1)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT31 (P33.9) 01_H **CDTM0_DTM5_3_N**: Inverted dead-time output of ATOM0_7 02_H **CDTM1_DTM5_3**: Dead-time output of ATOM1_7 03_H **CDTM2_DTM5_3_N**: Inverted dead-time output of ATOM2_7 04_H **CDTM0_DTM3_3_N**: Inverted dead-time output of TOM0_15 05_H **CDTM1_DTM3_3**: Dead-time output of TOM1_15 06_H **CDTM2_DTM3_3_N**: Inverted dead-time output of TOM2_15 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM0_1**: Dead-time output of TOM0_1 11_H **CDTM1_DTM0_1**: Dead-time output of TOM1_1 12_H **CDTM0_DTM4_1**: Dead-time output of ATOM0_1 13_H **CDTM1_DTM4_1**: Dead-time output of ATOM1_1 14_H **CDTM0_DTM1_2_N**: Inverted dead-time output of TOM0_6 15_H **CDTM1_DTM1_2_N**: Inverted dead-time output of TOM1_6 16_H **CDTM0_DTM1_3**: Dead-time output of TOM0_7 17_H **CDTM1_DTM1_3**: Dead-time output of TOM1_7 18_H **Reserved**: Reserved, do not use 19_H **CDTM2_DTM4_2_N**: Inverted dead-time output of ATOM2_2 $1A_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=2)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT32 (P33.10) 01_H **CDTM0_DTM4_0**: Dead-time output of ATOM0_0 02_H **CDTM1_DTM4_0_N**: Inverted dead-time output of ATOM1_0 03_H **CDTM2_DTM4_0**: Dead-time output of ATOM2_0 04_H **CDTM0_DTM0_0**: Dead-time output of TOM0_0 05_H **CDTM1_DTM0_0_N**: Inverted dead-time output of TOM1_0 06_H **CDTM2_DTM0_0**: Dead-time output of TOM2_0 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM0_0**: Dead-time output of TOM0_0 11_H **CDTM1_DTM0_0**: Dead-time output of TOM1_0 12_H **CDTM2_DTM4_0**: Dead-time output of ATOM2_0 13_H **Reserved**: Reserved, do not use 14_H **CDTM0_DTM1_1_N**: Inverted dead-time output of TOM0_5 15_H **CDTM1_DTM1_1_N**: Inverted dead-time output of TOM1_5 16_H **CDTM0_DTM1_3_N**: Inverted dead-time output of TOM0_7 17_H **CDTM1_DTM1_3_N**: Inverted dead-time output of TOM1_7 18_H **CDTM2_DTM4_0_N**: Inverted dead-time output of ATOM2_0 19_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=3)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT33 (P33.11) 01_H **CDTM0_DTM4_0_N**: Inverted dead-time output of ATOM0_0 02_H **CDTM1_DTM4_0**: Dead-time output of ATOM1_0 03_H **CDTM2_DTM4_0_N**: Inverted dead-time output of ATOM2_0 04_H **CDTM0_DTM0_0_N**: Inverted dead-time output of TOM0_0 05_H **CDTM1_DTM0_0**: Dead-time output of TOM1_0 06_H **CDTM2_DTM0_0_N**: Inverted dead-time output of TOM2_0 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM0_2**: Dead-time output of TOM0_2 11_H **CDTM1_DTM0_2**: Dead-time output of TOM1_2 12_H **CDTM0_DTM4_2**: Dead-time output of ATOM0_2 13_H **CDTM1_DTM4_2**: Dead-time output of ATOM1_2 14_H **CDTM0_DTM1_3_N**: Inverted dead-time output of TOM0_7 15_H **CDTM1_DTM1_3_N**: Inverted dead-time output of TOM1_7 16_H **CDTM2_DTM1_3_N**: Inverted dead-time output of TOM2_7 17_H **Reserved**: Reserved, do not use 18_H **CDTM2_DTM4_3_N**: Inverted dead-time output of ATOM2_3 19_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=4)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT34 (P33.12) 01_H **CDTM0_DTM4_1**: Dead-time output of ATOM0_1 02_H **CDTM1_DTM4_1_N**: Inverted dead-time output of ATOM1_1 03_H **CDTM2_DTM4_1**: Dead-time output of ATOM2_1 04_H **CDTM0_DTM0_1**: Dead-time output of TOM0_1 05_H **CDTM1_DTM0_1_N**: Inverted dead-time output of TOM1_1 06_H **CDTM2_DTM0_1**: Dead-time output of TOM2_1 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM1_DTM3_0**: Dead-time output of TOM1_12 11_H **CDTM2_DTM3_0**: Dead-time output of TOM2_12 12_H **CDTM2_DTM5_0**: Dead-time output of ATOM2_4 13_H **Reserved**: Reserved, do not use 14_H **CDTM0_DTM0_0_N**: Inverted dead-time output of TOM0_0 15_H **CDTM1_DTM0_0_N**: Inverted dead-time output of TOM1_0 16_H **CDTM2_DTM0_0_N**: Inverted dead-time output of TOM2_0 17_H **Reserved**: Reserved, do not use 18_H **CDTM2_DTM4_1_N**: Inverted dead-time output of ATOM2_1 19_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=5)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT35 (P33.13) 01_H **CDTM0_DTM4_1_N**: Inverted dead-time output of ATOM0_1 02_H **CDTM1_DTM4_1**: Dead-time output of ATOM1_1 03_H **CDTM2_DTM4_1_N**: Inverted dead-time output of ATOM2_1 04_H **CDTM0_DTM0_1_N**: Inverted dead-time output of TOM0_1 05_H **CDTM1_DTM0_1**: Dead-time output of TOM1_1 06_H **CDTM2_DTM0_1_N**: Inverted dead-time output of TOM2_1 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM1_DTM3_1**: Dead-time output of TOM1_13 11_H **CDTM2_DTM3_1**: Dead-time output of TOM2_13 12_H **CDTM2_DTM5_1**: Dead-time output of ATOM2_5 13_H **Reserved**: Reserved, do not use 14_H **CDTM0_DTM0_1_N**: Inverted dead-time output of TOM0_1 15_H **CDTM1_DTM0_1_N**: Inverted dead-time output of TOM1_1 16_H **CDTM2_DTM0_1_N**: Inverted dead-time output of TOM2_1 17_H **Reserved**: Reserved, do not use 18_H **CDTM2_DTM4_2_N**: Inverted dead-time output of ATOM2_2 19_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	15, 31	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.16 eGTM to Ports signal TOUT output selection register n (n=6)

For a description of this register, see 7579.

EGTM_TOUTSELn (n=6) Offset address: 061100_H+n*4

eGTM to Ports signal TOUT output selection register n Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	SEL5					SEL4					SEL3				
r	rw					rw					rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	SEL2					SEL1					SEL0				
r	rw					rw					rw				

Field	Bits	Type	Description
SELx (x=0)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT36 (P04.8) 01_H **CDTM0_DTM4_2**: Dead-time output of ATOM0_2 02_H **CDTM1_DTM4_2_N**: Inverted dead-time output of ATOM1_2 03_H **CDTM2_DTM4_2**: Dead-time output of ATOM2_2 04_H **CDTM0_DTM0_2**: Dead-time output of TOM0_2 05_H **CDTM1_DTM0_2_N**: Inverted dead-time output of TOM1_2 06_H **CDTM2_DTM0_2**: Dead-time output of TOM2_2 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM1_DTM3_2**: Dead-time output of TOM1_14 11_H **CDTM2_DTM3_2**: Dead-time output of TOM2_14 12_H **CDTM2_DTM5_2**: Dead-time output of ATOM2_6 13_H **Reserved**: Reserved, do not use 14_H **CDTM0_DTM1_0_N**: Inverted dead-time output of TOM0_4 15_H **CDTM1_DTM1_0_N**: Inverted dead-time output of TOM1_4 16_H **CDTM2_DTM1_0_N**: Inverted dead-time output of TOM2_4 17_H **Reserved**: Reserved, do not use 18_H **CDTM2_DTM4_3_N**: Inverted dead-time output of ATOM2_3 19_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=1)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT37 (P04.9) 01_H **CDTM0_DTM4_2_N**: Inverted dead-time output of ATOM0_2 02_H **CDTM1_DTM4_2**: Dead-time output of ATOM1_2 03_H **CDTM2_DTM4_2_N**: Inverted dead-time output of ATOM2_2 04_H **CDTM0_DTM0_2_N**: Inverted dead-time output of TOM0_2 05_H **CDTM1_DTM0_2**: Dead-time output of TOM1_2 06_H **CDTM2_DTM0_2_N**: Inverted dead-time output of TOM2_2 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM1_DTM3_3**: Dead-time output of TOM1_15 11_H **CDTM2_DTM3_3**: Dead-time output of TOM2_15 12_H **CDTM2_DTM5_3**: Dead-time output of ATOM2_7 13_H **Reserved**: Reserved, do not use 14_H **CDTM0_DTM1_0**: Dead-time output of TOM0_4 15_H **CDTM1_DTM1_0**: Dead-time output of TOM1_4 16_H **CDTM0_DTM1_1_N**: Inverted dead-time output of TOM0_5 17_H **CDTM1_DTM1_1_N**: Inverted dead-time output of TOM1_5 18_H **CDTM2_DTM5_3**: Dead-time output of ATOM2_7 19_H **Reserved**: Reserved, do not use $1A_H$ **CDTM2_DTM4_0_N**: Inverted dead-time output of ATOM2_0 $1B_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=2)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT38 (P32.2) 01_H **CDTM0_DTM4_3**: Dead-time output of ATOM0_3 02_H **CDTM1_DTM4_3_N**: Inverted dead-time output of ATOM1_3 03_H **CDTM2_DTM4_3**: Dead-time output of ATOM2_3 04_H **CDTM0_DTM0_3**: Dead-time output of TOM0_3 05_H **CDTM1_DTM0_3_N**: Inverted dead-time output of TOM1_3 06_H **CDTM2_DTM0_3**: Dead-time output of TOM2_3 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM0_3**: Dead-time output of TOM0_3 11_H **CDTM1_DTM0_3**: Dead-time output of TOM1_3 12_H **CDTM0_DTM4_3**: Dead-time output of ATOM0_3 13_H **CDTM1_DTM4_3**: Dead-time output of ATOM1_3 14_H **CDTM0_DTM1_2_N**: Inverted dead-time output of TOM0_6 15_H **CDTM1_DTM1_2_N**: Inverted dead-time output of TOM1_6 16_H **CDTM0_DTM1_2_N**: Inverted dead-time output of TOM0_6 17_H **CDTM1_DTM1_2_N**: Inverted dead-time output of TOM1_6 18_H **CDTM2_DTM5_2_N**: Inverted dead-time output of ATOM2_6 19_H **Reserved**: Reserved, do not use $1A_H$ **CDTM2_DTM5_2_N**: Inverted dead-time output of ATOM2_6 $1B_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=3)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT39 (P13.8) 01_H **CDTM0_DTM4_3_N**: Inverted dead-time output of ATOM0_3 02_H **CDTM1_DTM4_3**: Dead-time output of ATOM1_3 03_H **CDTM2_DTM4_3_N**: Inverted dead-time output of ATOM2_3 04_H **CDTM0_DTM0_3_N**: Inverted dead-time output of TOM0_3 05_H **CDTM1_DTM0_3**: Dead-time output of TOM1_3 06_H **CDTM2_DTM0_3_N**: Inverted dead-time output of TOM2_3 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM1_0**: Dead-time output of TOM0_4 11_H **CDTM1_DTM1_0**: Dead-time output of TOM1_4 12_H **CDTM0_DTM5_0**: Dead-time output of ATOM0_4 13_H **CDTM1_DTM5_0**: Dead-time output of ATOM1_4 14_H **CDTM0_DTM1_3_N**: Inverted dead-time output of TOM0_7 15_H **CDTM1_DTM1_3_N**: Inverted dead-time output of TOM1_7 16_H **CDTM0_DTM0_1_N**: Inverted dead-time output of TOM0_1 17_H **CDTM1_DTM0_1_N**: Inverted dead-time output of TOM1_1 18_H **Reserved**: Reserved, do not use 19_H **CDTM0_DTM4_1_N**: Inverted dead-time output of ATOM0_1 $1A_H$ **CDTM1_DTM4_1_N**: Inverted dead-time output of ATOM1_1 $1B_H$ **CDTM2_DTM4_1_N**: Inverted dead-time output of ATOM2_1 $1C_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=4)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT40 (P32.4) 01_H **CDTM0_DTM5_0**: Dead-time output of ATOM0_4 02_H **CDTM1_DTM5_0_N**: Inverted dead-time output of ATOM1_4 03_H **CDTM2_DTM5_0**: Dead-time output of ATOM2_4 04_H **CDTM0_DTM1_0**: Dead-time output of TOM0_4 05_H **CDTM1_DTM1_0_N**: Inverted dead-time output of TOM1_4 06_H **CDTM2_DTM1_0**: Dead-time output of TOM2_4 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM1_1**: Dead-time output of TOM0_5 11_H **CDTM1_DTM1_1**: Dead-time output of TOM1_5 12_H **CDTM0_DTM5_1**: Dead-time output of ATOM0_5 13_H **CDTM1_DTM5_1**: Dead-time output of ATOM1_5 14_H **CDTM0_DTM1_2_N**: Inverted dead-time output of TOM0_6 15_H **CDTM1_DTM1_2_N**: Inverted dead-time output of TOM1_6 16_H **CDTM0_DTM0_2_N**: Inverted dead-time output of TOM0_2 17_H **CDTM1_DTM0_2_N**: Inverted dead-time output of TOM1_2 18_H **Reserved**: Reserved, do not use 19_H **CDTM0_DTM4_2_N**: Inverted dead-time output of ATOM0_2 $1A_H$ **CDTM1_DTM4_2_N**: Inverted dead-time output of ATOM1_2 $1B_H$ **CDTM2_DTM4_2_N**: Inverted dead-time output of ATOM2_2 $1C_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=5)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT41 (P23.0) 01_H **CDTM0_DTM5_0_N**: Inverted dead-time output of ATOM0_4 02_H **CDTM1_DTM5_0**: Dead-time output of ATOM1_4 03_H **CDTM2_DTM5_0_N**: Inverted dead-time output of ATOM2_4 04_H **CDTM0_DTM1_0_N**: Inverted dead-time output of TOM0_4 05_H **CDTM1_DTM1_0**: Dead-time output of TOM1_4 06_H **CDTM2_DTM1_0_N**: Inverted dead-time output of TOM2_4 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM2_2**: Dead-time output of TOM0_10 11_H **CDTM1_DTM1_1**: Dead-time output of TOM1_5 12_H **CDTM0_DTM5_1**: Dead-time output of ATOM0_5 13_H **CDTM1_DTM5_1**: Dead-time output of ATOM1_5 14_H **CDTM0_DTM0_2_N**: Inverted dead-time output of TOM0_2 15_H **CDTM1_DTM0_2_N**: Inverted dead-time output of TOM1_2 16_H **CDTM0_DTM0_2**: Dead-time output of TOM0_2 17_H **CDTM1_DTM0_2**: Dead-time output of TOM1_2 18_H **CDTM0_DTM4_1_N**: Inverted dead-time output of ATOM0_1 19_H **Reserved**: Reserved, do not use $1A_H$ **CDTM1_DTM4_1_N**: Inverted dead-time output of ATOM1_1 $1B_H$ **CDTM2_DTM4_1_N**: Inverted dead-time output of ATOM2_1 $1C_H$ **CDTM0_DTM4_2_N**: Inverted dead-time output of ATOM0_2 $1D_H$ **CDTM1_DTM4_2_N**: Inverted dead-time output of ATOM1_2 $1E_H$ **Reserved**: Reserved, do not use $1F_H$ **Reserved**: Reserved, do not use
0	15, 31	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.17 eGTM to Ports signal TOUT output selection register n (n=7)

For a description of this register, see 7579.

EGTM_TOUTSELn (n=7) Offset address: 061100_H+n*4

eGTM to Ports signal TOUT output selection register n Kernel Reset value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	SEL5					SEL4					SEL3				
r	rw					rw					rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	SEL2					SEL1					SEL0				
r	rw					rw					rw				

Field	Bits	Type	Description
SELx (x=0)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT42 (P23.1) 01_H **CDTM0_DTM5_1**: Dead-time output of ATOM0_5 02_H **CDTM1_DTM5_1_N**: Inverted dead-time output of ATOM1_5 03_H **CDTM2_DTM5_1**: Dead-time output of ATOM2_5 04_H **CDTM0_DTM1_1**: Dead-time output of TOM0_5 05_H **CDTM1_DTM1_1_N**: Inverted dead-time output of TOM1_5 06_H **CDTM2_DTM1_1**: Dead-time output of TOM2_5 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM1_2**: Dead-time output of TOM0_6 11_H **CDTM0_DTM3_3**: Dead-time output of TOM0_15 12_H **CDTM0_DTM5_2**: Dead-time output of ATOM0_6 13_H **CDTM1_DTM5_2**: Dead-time output of ATOM1_6 14_H **CDTM0_DTM1_3_N**: Inverted dead-time output of TOM0_7 15_H **CDTM1_DTM1_3_N**: Inverted dead-time output of TOM1_7 16_H **CDTM0_DTM0_3_N**: Inverted dead-time output of TOM0_3 17_H **CDTM1_DTM0_3_N**: Inverted dead-time output of TOM1_3 18_H **CDTM0_DTM4_2**: Dead-time output of ATOM0_2 19_H **Reserved**: Reserved, do not use $1A_H$ **CDTM0_DTM4_3_N**: Inverted dead-time output of ATOM0_3 $1B_H$ **CDTM1_DTM4_3_N**: Inverted dead-time output of ATOM1_3 $1C_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=1)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT43 (P23.2) 01_H **CDTM0_DTM5_1_N**: Inverted dead-time output of ATOM0_5 02_H **CDTM1_DTM5_1**: Dead-time output of ATOM1_5 03_H **CDTM2_DTM5_1_N**: Inverted dead-time output of ATOM2_5 04_H **CDTM0_DTM1_1_N**: Inverted dead-time output of TOM0_5 05_H **CDTM1_DTM1_1**: Dead-time output of TOM1_5 06_H **CDTM2_DTM1_1_N**: Inverted dead-time output of TOM2_5 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM2_3**: Dead-time output of TOM0_11 11_H **CDTM1_DTM1_2**: Dead-time output of TOM1_6 12_H **CDTM2_DTM4_1**: Dead-time output of ATOM2_1 13_H **Reserved**: Reserved, do not use 14_H **CDTM2_DTM0_0_N**: Inverted dead-time output of TOM2_0 15_H **CDTM0_DTM0_0_N**: Inverted dead-time output of TOM0_0 16_H **CDTM1_DTM0_3_N**: Inverted dead-time output of TOM1_3 17_H **CDTM2_DTM0_3_N**: Inverted dead-time output of TOM2_3 18_H **CDTM0_DTM4_0_N**: Inverted dead-time output of ATOM0_0 19_H **Reserved**: Reserved, do not use $1A_H$ **CDTM0_DTM4_3_N**: Inverted dead-time output of ATOM0_3 $1B_H$ **CDTM1_DTM4_3_N**: Inverted dead-time output of ATOM1_3 $1C_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=2)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT44 (P23.3) 01_H **CDTM0_DTM5_2**: Dead-time output of ATOM0_6 02_H **CDTM1_DTM5_2_N**: Inverted dead-time output of ATOM1_6 03_H **CDTM2_DTM5_2**: Dead-time output of ATOM2_6 04_H **CDTM0_DTM1_2**: Dead-time output of TOM0_6 05_H **CDTM1_DTM1_2_N**: Inverted dead-time output of TOM1_6 06_H **CDTM2_DTM1_2**: Dead-time output of TOM2_6 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM3_0**: Dead-time output of TOM0_12 11_H **CDTM1_DTM1_3**: Dead-time output of TOM1_7 12_H **CDTM2_DTM4_2**: Dead-time output of ATOM2_2 13_H **Reserved**: Reserved, do not use 14_H **CDTM2_DTM0_1_N**: Inverted dead-time output of TOM2_1 15_H **CDTM0_DTM0_1_N**: Inverted dead-time output of TOM0_1 16_H **CDTM1_DTM0_2_N**: Inverted dead-time output of TOM1_2 17_H **CDTM2_DTM0_2_N**: Inverted dead-time output of TOM2_2 18_H **CDTM0_DTM4_2_N**: Inverted dead-time output of ATOM0_2 19_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=3)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT45 (P23.4) 01_H **CDTM0_DTM5_2_N**: Inverted dead-time output of ATOM0_6 02_H **CDTM1_DTM5_2**: Dead-time output of ATOM1_6 03_H **CDTM2_DTM5_2_N**: Inverted dead-time output of ATOM2_6 04_H **CDTM0_DTM1_2_N**: Inverted dead-time output of TOM0_6 05_H **CDTM1_DTM1_2**: Dead-time output of TOM1_6 06_H **CDTM2_DTM1_2_N**: Inverted dead-time output of TOM2_6 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM1_3**: Dead-time output of TOM0_7 11_H **CDTM1_DTM1_3**: Dead-time output of TOM1_7 12_H **CDTM0_DTM5_3**: Dead-time output of ATOM0_7 13_H **CDTM1_DTM5_3**: Dead-time output of ATOM1_7 14_H **CDTM2_DTM0_3_N**: Inverted dead-time output of TOM2_3 15_H **CDTM0_DTM0_3_N**: Inverted dead-time output of TOM0_3 16_H **CDTM1_DTM0_3_N**: Inverted dead-time output of TOM1_3 17_H **CDTM2_DTM0_3_N**: Inverted dead-time output of TOM2_3 18_H **CDTM0_DTM4_3_N**: Inverted dead-time output of ATOM0_3 19_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=4)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT46 (P23.5) 01_H **CDTM0_DTM5_3**: Dead-time output of ATOM0_7 02_H **CDTM1_DTM5_3_N**: Inverted dead-time output of ATOM1_7 03_H **CDTM2_DTM5_3**: Dead-time output of ATOM2_7 04_H **CDTM0_DTM1_3**: Dead-time output of TOM0_7 05_H **CDTM1_DTM1_3_N**: Inverted dead-time output of TOM1_7 06_H **CDTM2_DTM1_3**: Dead-time output of TOM2_7 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM2_2**: Dead-time output of TOM0_10 11_H **CDTM2_DTM0_2**: Dead-time output of TOM2_2 12_H **CDTM0_DTM4_2**: Dead-time output of ATOM0_2 13_H **Reserved**: Reserved, do not use 14_H **CDTM2_DTM0_2_N**: Inverted dead-time output of TOM2_2 15_H **CDTM0_DTM0_2_N**: Inverted dead-time output of TOM0_2 16_H **CDTM1_DTM0_3_N**: Inverted dead-time output of TOM1_3 17_H **CDTM2_DTM0_3_N**: Inverted dead-time output of TOM2_3 18_H **CDTM0_DTM4_3_N**: Inverted dead-time output of ATOM0_3 19_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=5)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT47 (P22.0) 01_H **CDTM0_DTM5_3_N**: Inverted dead-time output of ATOM0_7 02_H **CDTM1_DTM5_3**: Dead-time output of ATOM1_7 03_H **CDTM2_DTM5_3_N**: Inverted dead-time output of ATOM2_7 04_H **CDTM0_DTM1_3_N**: Inverted dead-time output of TOM0_7 05_H **CDTM1_DTM1_3**: Dead-time output of TOM1_7 06_H **CDTM2_DTM1_3_N**: Inverted dead-time output of TOM2_7 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM2_1**: Dead-time output of TOM0_9 11_H **CDTM2_DTM0_1**: Dead-time output of TOM2_1 12_H **CDTM0_DTM4_1**: Dead-time output of ATOM0_1 13_H **Reserved**: Reserved, do not use 14_H **CDTM2_DTM0_3_N**: Inverted dead-time output of TOM2_3 15_H **CDTM0_DTM0_3_N**: Inverted dead-time output of TOM0_3 16_H **CDTM1_DTM0_3_N**: Inverted dead-time output of TOM1_3 17_H **CDTM2_DTM0_3_N**: Inverted dead-time output of TOM2_3 18_H **CDTM0_DTM4_3_N**: Inverted dead-time output of ATOM0_3 19_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	15, 31	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.18 eGTM to Ports signal TOUT output selection register n (n=8)

For a description of this register, see 7579.

EGTM_TOUTSELn (n=8) Offset address: 061100_H+n*4

eGTM to Ports signal TOUT output selection register n Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	SEL5					SEL4					SEL3				
r	rw					rw					rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	SEL2					SEL1					SEL0				
r	rw					rw					rw				

Field	Bits	Type	Description
SELx (x=0)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT48 (P22.1) 01_H **CDTM0_DTM4_0**: Dead-time output of ATOM0_0 02_H **CDTM1_DTM4_0_N**: Inverted dead-time output of ATOM1_0 03_H **CDTM2_DTM4_0**: Dead-time output of ATOM2_0 04_H **CDTM0_DTM2_0**: Dead-time output of TOM0_8 05_H **CDTM1_DTM2_0_N**: Inverted dead-time output of TOM1_8 06_H **CDTM2_DTM2_0**: Dead-time output of TOM2_8 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM2_0**: Dead-time output of TOM0_8 11_H **CDTM2_DTM0_0**: Dead-time output of TOM2_0 12_H **CDTM0_DTM4_0**: Dead-time output of ATOM0_0 13_H **Reserved**: Reserved, do not use 14_H **CDTM2_DTM1_2**: Dead-time output of TOM2_6 15_H **CDTM0_DTM1_2**: Dead-time output of TOM0_6 16_H **CDTM0_DTM1_2**: Dead-time output of TOM0_6 17_H **CDTM1_DTM1_2**: Dead-time output of TOM1_6 18_H **CDTM0_DTM5_2**: Dead-time output of ATOM0_6 19_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=1)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT49 (P22.2) 01_H **CDTM0_DTM4_0_N**: Inverted dead-time output of ATOM0_0 02_H **CDTM1_DTM4_0**: Dead-time output of ATOM1_0 03_H **CDTM2_DTM4_0_N**: Inverted dead-time output of ATOM2_0 04_H **CDTM0_DTM2_0_N**: Inverted dead-time output of TOM0_8 05_H **CDTM1_DTM2_0**: Dead-time output of TOM1_8 06_H **CDTM2_DTM2_0_N**: Inverted dead-time output of TOM2_8 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM2_3**: Dead-time output of TOM0_11 11_H **CDTM2_DTM0_3**: Dead-time output of TOM2_3 12_H **CDTM0_DTM4_3**: Dead-time output of ATOM0_3 13_H **Reserved**: Reserved, do not use 14_H **CDTM2_DTM1_2**: Dead-time output of TOM2_6 15_H **CDTM0_DTM1_2**: Dead-time output of TOM0_6 16_H **CDTM0_DTM1_2_N**: Inverted dead-time output of TOM0_6 17_H **CDTM1_DTM1_2_N**: Inverted dead-time output of TOM1_6 18_H **CDTM0_DTM5_2**: Dead-time output of ATOM0_6 19_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=2)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT50 (P22.3) 01_H **CDTM0_DTM4_1**: Dead-time output of ATOM0_1 02_H **CDTM1_DTM4_1_N**: Inverted dead-time output of ATOM1_1 03_H **CDTM2_DTM4_1**: Dead-time output of ATOM2_1 04_H **CDTM0_DTM2_1**: Dead-time output of TOM0_9 05_H **CDTM1_DTM2_1_N**: Inverted dead-time output of TOM1_9 06_H **CDTM2_DTM2_1**: Dead-time output of TOM2_9 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM3_0**: Dead-time output of TOM0_12 11_H **CDTM2_DTM1_0**: Dead-time output of TOM2_4 12_H **CDTM0_DTM5_0**: Dead-time output of ATOM0_4 13_H **Reserved**: Reserved, do not use 14_H **CDTM2_DTM0_2**: Dead-time output of TOM2_2 15_H **CDTM2_DTM0_1_N**: Inverted dead-time output of TOM2_1 16_H **CDTM0_DTM1_3**: Dead-time output of TOM0_7 17_H **CDTM1_DTM1_3**: Dead-time output of TOM1_7 18_H **CDTM0_DTM4_1_N**: Inverted dead-time output of ATOM0_1 19_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=3)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT51 (P21.0) 01_H **CDTM0_DTM4_1_N**: Inverted dead-time output of ATOM0_1 02_H **CDTM1_DTM4_1**: Dead-time output of ATOM1_1 03_H **CDTM2_DTM4_1_N**: Inverted dead-time output of ATOM2_1 04_H **CDTM0_DTM2_1_N**: Inverted dead-time output of TOM0_9 05_H **CDTM1_DTM2_1**: Dead-time output of TOM1_9 06_H **CDTM2_DTM2_1_N**: Inverted dead-time output of TOM2_9 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM2_0**: Dead-time output of TOM0_8 11_H **CDTM2_DTM2_0**: Dead-time output of TOM2_8 12_H **CDTM2_DTM5_0**: Dead-time output of ATOM2_4 13_H **Reserved**: Reserved, do not use 14_H **CDTM2_DTM0_0_N**: Inverted dead-time output of TOM2_0 15_H **CDTM0_DTM0_0_N**: Inverted dead-time output of TOM0_0 16_H **CDTM1_DTM0_0_N**: Inverted dead-time output of TOM1_0 17_H **CDTM2_DTM0_0_N**: Inverted dead-time output of TOM2_0 18_H **CDTM2_DTM4_1_N**: Inverted dead-time output of ATOM2_1 19_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=4)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT52 (P21.1) 01_H **CDTM0_DTM4_2**: Dead-time output of ATOM0_2 02_H **CDTM1_DTM4_2_N**: Inverted dead-time output of ATOM1_2 03_H **CDTM2_DTM4_2**: Dead-time output of ATOM2_2 04_H **CDTM0_DTM2_2**: Dead-time output of TOM0_10 05_H **CDTM1_DTM2_2_N**: Inverted dead-time output of TOM1_10 06_H **CDTM2_DTM2_2**: Dead-time output of TOM2_10 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM2_1**: Dead-time output of TOM0_9 11_H **CDTM2_DTM2_1**: Dead-time output of TOM2_9 12_H **CDTM2_DTM5_1**: Dead-time output of ATOM2_5 13_H **Reserved**: Reserved, do not use 14_H **CDTM2_DTM0_1_N**: Inverted dead-time output of TOM2_1 15_H **CDTM0_DTM0_1_N**: Inverted dead-time output of TOM0_1 16_H **CDTM1_DTM0_1_N**: Inverted dead-time output of TOM1_1 17_H **CDTM2_DTM0_1_N**: Inverted dead-time output of TOM2_1 18_H **CDTM2_DTM4_2_N**: Inverted dead-time output of ATOM2_2 19_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=5)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT53 (P21.2) 01_H **CDTM0_DTM4_2_N**: Inverted dead-time output of ATOM0_2 02_H **CDTM1_DTM4_2**: Dead-time output of ATOM1_2 03_H **CDTM2_DTM4_2_N**: Inverted dead-time output of ATOM2_2 04_H **CDTM0_DTM2_2_N**: Inverted dead-time output of TOM0_10 05_H **CDTM1_DTM2_2**: Dead-time output of TOM1_10 06_H **CDTM2_DTM2_2_N**: Inverted dead-time output of TOM2_10 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM0_0**: Dead-time output of TOM0_0 11_H **CDTM2_DTM0_0**: Dead-time output of TOM2_0 12_H **CDTM0_DTM4_0**: Dead-time output of ATOM0_0 13_H **Reserved**: Reserved, do not use 14_H **CDTM0_DTM1_1_N**: Inverted dead-time output of TOM0_5 15_H **CDTM2_DTM1_1_N**: Inverted dead-time output of TOM2_5 16_H **CDTM0_DTM1_0**: Dead-time output of TOM0_4 17_H **CDTM1_DTM1_0**: Dead-time output of TOM1_4 18_H **CDTM0_DTM4_1_N**: Inverted dead-time output of ATOM0_1 19_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	15, 31	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.19 eGTM to Ports signal TOUT output selection register n (n=9)

For a description of this register, see 7579.

EGTM_TOUTSELn (n=9) Offset address: 061100_H+n*4

eGTM to Ports signal TOUT output selection register n Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	SEL5					SEL4					SEL3				
r	rw					rw					rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	SEL2					SEL1					SEL0				
r	rw					rw					rw				

Field	Bits	Type	Description
SELx (x=0)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT54 (P21.3) 01_H **CDTM0_DTM4_3**: Dead-time output of ATOM0_3 02_H **CDTM1_DTM4_3_N**: Inverted dead-time output of ATOM1_3 03_H **CDTM2_DTM4_3**: Dead-time output of ATOM2_3 04_H **CDTM0_DTM2_3**: Dead-time output of TOM0_11 05_H **CDTM1_DTM2_3_N**: Inverted dead-time output of TOM1_11 06_H **CDTM2_DTM2_3**: Dead-time output of TOM2_11 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM0_1**: Dead-time output of TOM0_1 11_H **CDTM2_DTM0_1**: Dead-time output of TOM2_1 12_H **CDTM0_DTM4_1**: Dead-time output of ATOM0_1 13_H **Reserved**: Reserved, do not use 14_H **CDTM0_DTM1_2_N**: Inverted dead-time output of TOM0_6 15_H **CDTM2_DTM1_2_N**: Inverted dead-time output of TOM2_6 16_H **CDTM0_DTM1_0_N**: Inverted dead-time output of TOM0_4 17_H **CDTM1_DTM1_0_N**: Inverted dead-time output of TOM1_4 18_H **CDTM0_DTM4_0_N**: Inverted dead-time output of ATOM0_0 19_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=1)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT55 (P21.4) 01_H **CDTM0_DTM4_3_N**: Inverted dead-time output of ATOM0_3 02_H **CDTM1_DTM4_3**: Dead-time output of ATOM1_3 03_H **CDTM2_DTM4_3_N**: Inverted dead-time output of ATOM2_3 04_H **CDTM0_DTM2_3_N**: Inverted dead-time output of TOM0_11 05_H **CDTM1_DTM2_3**: Dead-time output of TOM1_11 06_H **CDTM2_DTM2_3_N**: Inverted dead-time output of TOM2_11 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM0_2**: Dead-time output of TOM0_2 11_H **CDTM2_DTM0_2**: Dead-time output of TOM2_2 12_H **CDTM0_DTM4_2**: Dead-time output of ATOM0_2 13_H **Reserved**: Reserved, do not use 14_H **CDTM0_DTM1_3_N**: Inverted dead-time output of TOM0_7 15_H **CDTM2_DTM1_3_N**: Inverted dead-time output of TOM2_7 16_H **CDTM0_DTM1_1**: Dead-time output of TOM0_5 17_H **CDTM1_DTM1_1**: Dead-time output of TOM1_5 18_H **CDTM0_DTM4_1_N**: Inverted dead-time output of ATOM0_1 19_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=2)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT56 (P21.5) 01_H **CDTM0_DTM5_0**: Dead-time output of ATOM0_4 02_H **CDTM1_DTM5_0_N**: Inverted dead-time output of ATOM1_4 03_H **CDTM2_DTM5_0**: Dead-time output of ATOM2_4 04_H **CDTM0_DTM3_0**: Dead-time output of TOM0_12 05_H **CDTM1_DTM3_0_N**: Inverted dead-time output of TOM1_12 06_H **CDTM2_DTM3_0**: Dead-time output of TOM2_12 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM0_3**: Dead-time output of TOM0_3 11_H **CDTM2_DTM0_3**: Dead-time output of TOM2_3 12_H **CDTM0_DTM4_3**: Dead-time output of ATOM0_3 13_H **Reserved**: Reserved, do not use 14_H **CDTM0_DTM1_0_N**: Inverted dead-time output of TOM0_4 15_H **CDTM2_DTM1_0_N**: Inverted dead-time output of TOM2_4 16_H **CDTM0_DTM1_1_N**: Inverted dead-time output of TOM0_5 17_H **CDTM1_DTM1_1_N**: Inverted dead-time output of TOM1_5 18_H **CDTM0_DTM4_2_N**: Inverted dead-time output of ATOM0_2 19_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=3)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT57 (P21.6) 01_H **CDTM0_DTM5_0_N**: Inverted dead-time output of ATOM0_4 02_H **CDTM1_DTM5_0**: Dead-time output of ATOM1_4 03_H **CDTM2_DTM5_0_N**: Inverted dead-time output of ATOM2_4 04_H **CDTM0_DTM3_0_N**: Inverted dead-time output of TOM0_12 05_H **CDTM1_DTM3_0**: Dead-time output of TOM1_12 06_H **CDTM2_DTM3_0_N**: Inverted dead-time output of TOM2_12 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM1_0**: Dead-time output of TOM0_4 11_H **CDTM2_DTM1_0**: Dead-time output of TOM2_4 12_H **CDTM0_DTM5_0**: Dead-time output of ATOM0_4 13_H **Reserved**: Reserved, do not use 14_H **CDTM0_DTM0_1_N**: Inverted dead-time output of TOM0_1 15_H **CDTM2_DTM0_1_N**: Inverted dead-time output of TOM2_1 16_H **CDTM0_DTM0_1**: Dead-time output of TOM0_1 17_H **CDTM1_DTM0_1**: Dead-time output of TOM1_1 18_H **CDTM2_DTM5_0_N**: Inverted dead-time output of ATOM2_4 19_H **Reserved**: Reserved, do not use $1A_H$ **CDTM0_DTM4_1_N**: Inverted dead-time output of ATOM0_1 $1B_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=4)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT58 (P21.7) 01_H **CDTM0_DTM5_1**: Dead-time output of ATOM0_5 02_H **CDTM1_DTM5_1_N**: Inverted dead-time output of ATOM1_5 03_H **CDTM2_DTM5_1**: Dead-time output of ATOM2_5 04_H **CDTM0_DTM3_1**: Dead-time output of TOM0_13 05_H **CDTM1_DTM3_1_N**: Inverted dead-time output of TOM1_13 06_H **CDTM2_DTM3_1**: Dead-time output of TOM2_13 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM1_1**: Dead-time output of TOM0_5 11_H **CDTM2_DTM1_1**: Dead-time output of TOM2_5 12_H **CDTM0_DTM5_1**: Dead-time output of ATOM0_5 13_H **Reserved**: Reserved, do not use 14_H **CDTM0_DTM0_2_N**: Inverted dead-time output of TOM0_2 15_H **CDTM2_DTM0_2_N**: Inverted dead-time output of TOM2_2 16_H **Reserved**: Reserved, do not use 17_H **Reserved**: Reserved, do not use 18_H **CDTM0_DTM4_2_N**: Inverted dead-time output of ATOM0_2 19_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=5)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT59 (P20.0) 01_H **CDTM0_DTM5_1_N**: Inverted dead-time output of ATOM0_5 02_H **CDTM1_DTM5_1**: Dead-time output of ATOM1_5 03_H **CDTM2_DTM5_1_N**: Inverted dead-time output of ATOM2_5 04_H **CDTM0_DTM3_1_N**: Inverted dead-time output of TOM0_13 05_H **CDTM1_DTM3_1**: Dead-time output of TOM1_13 06_H **CDTM2_DTM3_1_N**: Inverted dead-time output of TOM2_13 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM1_2**: Dead-time output of TOM0_6 11_H **CDTM2_DTM1_2**: Dead-time output of TOM2_6 12_H **CDTM0_DTM5_2**: Dead-time output of ATOM0_6 13_H **Reserved**: Reserved, do not use 14_H **CDTM0_DTM0_3_N**: Inverted dead-time output of TOM0_3 15_H **CDTM2_DTM0_3_N**: Inverted dead-time output of TOM2_3 16_H **Reserved**: Reserved, do not use 17_H **Reserved**: Reserved, do not use 18_H **CDTM2_DTM5_1_N**: Inverted dead-time output of ATOM2_5 19_H **Reserved**: Reserved, do not use $1A_H$ **CDTM0_DTM4_3_N**: Inverted dead-time output of ATOM0_3 $1B_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	15, 31	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.20 eGTM to Ports signal TOUT output selection register n (n=10)

For a description of this register, see 7579.

EGTM_TOUTSELn (n=10) Offset address: 061100_H+n*4

eGTM to Ports signal TOUT output selection register n Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	SEL5					SEL4					SEL3				
r	rw					rw					rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	SEL2					SEL1					SEL0				
r	rw					rw					rw				

Field	Bits	Type	Description
SELx (x=0)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT60 (P20.1) 01_H **CDTM0_DTM5_2**: Dead-time output of ATOM0_6 02_H **CDTM1_DTM5_2_N**: Inverted dead-time output of ATOM1_6 03_H **CDTM2_DTM5_2**: Dead-time output of ATOM2_6 04_H **CDTM0_DTM3_2**: Dead-time output of TOM0_14 05_H **CDTM1_DTM3_2_N**: Inverted dead-time output of TOM1_14 06_H **CDTM2_DTM3_2**: Dead-time output of TOM2_14 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM1_DTM2_3**: Dead-time output of TOM1_11 11_H **CDTM2_DTM0_3**: Dead-time output of TOM2_3 12_H **CDTM0_DTM4_3**: Dead-time output of ATOM0_3 13_H **CDTM1_DTM4_3**: Dead-time output of ATOM1_3 14_H **CDTM0_DTM0_3_N**: Inverted dead-time output of TOM0_3 15_H **CDTM2_DTM1_0_N**: Inverted dead-time output of TOM2_4 16_H **Reserved**: Reserved, do not use 17_H **Reserved**: Reserved, do not use 18_H **CDTM2_DTM5_2_N**: Inverted dead-time output of ATOM2_6 19_H **Reserved**: Reserved, do not use $1A_H$ **CDTM0_DTM5_0_N**: Inverted dead-time output of ATOM0_4 $1B_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=1)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT61 (P20.3) 01_H **CDTM0_DTM5_2_N**: Inverted dead-time output of ATOM0_6 02_H **CDTM1_DTM5_2**: Dead-time output of ATOM1_6 03_H **CDTM2_DTM5_2_N**: Inverted dead-time output of ATOM2_6 04_H **CDTM0_DTM3_2_N**: Inverted dead-time output of TOM0_14 05_H **CDTM1_DTM3_2**: Dead-time output of TOM1_14 06_H **CDTM2_DTM3_2_N**: Inverted dead-time output of TOM2_14 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM1_DTM3_0**: Dead-time output of TOM1_12 11_H **CDTM2_DTM1_0**: Dead-time output of TOM2_4 12_H **CDTM0_DTM5_0**: Dead-time output of ATOM0_4 13_H **CDTM1_DTM5_0**: Dead-time output of ATOM1_4 14_H **CDTM2_DTM0_0_N**: Inverted dead-time output of TOM2_0 15_H **CDTM0_DTM0_0_N**: Inverted dead-time output of TOM0_0 16_H **CDTM2_DTM0_1_N**: Inverted dead-time output of TOM2_1 17_H **Reserved**: Reserved, do not use 18_H **CDTM2_DTM5_3_N**: Inverted dead-time output of ATOM2_7 19_H **Reserved**: Reserved, do not use $1A_H$ **CDTM0_DTM4_1_N**: Inverted dead-time output of ATOM0_1 $1B_H$ **CDTM1_DTM4_1_N**: Inverted dead-time output of ATOM1_1 $1C_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=2)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT62 (P20.6) 01_H **CDTM0_DTM5_3**: Dead-time output of ATOM0_7 02_H **CDTM1_DTM5_3_N**: Inverted dead-time output of ATOM1_7 03_H **CDTM2_DTM5_3**: Dead-time output of ATOM2_7 04_H **CDTM0_DTM3_3**: Dead-time output of TOM0_15 05_H **CDTM1_DTM3_3_N**: Inverted dead-time output of TOM1_15 06_H **CDTM2_DTM3_3**: Dead-time output of TOM2_15 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM1_DTM2_2**: Dead-time output of TOM1_10 11_H **CDTM2_DTM2_2**: Dead-time output of TOM2_10 12_H **CDTM2_DTM5_2**: Dead-time output of ATOM2_6 13_H **Reserved**: Reserved, do not use 14_H **CDTM2_DTM0_3**: Dead-time output of TOM2_3 15_H **CDTM0_DTM0_3**: Dead-time output of TOM0_3 16_H **CDTM2_DTM1_0**: Dead-time output of TOM2_4 17_H **Reserved**: Reserved, do not use … 19_H **Reserved**: Reserved, do not use $1A_H$ **CDTM2_DTM4_3_N**: Inverted dead-time output of ATOM2_3 $1B_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
SELx (x=3)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT63 (P20.7) 01_H **CDTM0_DTM5_3_N**: Inverted dead-time output of ATOM0_7 02_H **CDTM1_DTM5_3**: Dead-time output of ATOM1_7 03_H **CDTM2_DTM5_3_N**: Inverted dead-time output of ATOM2_7 04_H **CDTM0_DTM3_3_N**: Inverted dead-time output of TOM0_15 05_H **CDTM1_DTM3_3**: Dead-time output of TOM1_15 06_H **CDTM2_DTM3_3_N**: Inverted dead-time output of TOM2_15 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM1_DTM2_3**: Dead-time output of TOM1_11 11_H **CDTM2_DTM2_3**: Dead-time output of TOM2_11 12_H **CDTM2_DTM5_3**: Dead-time output of ATOM2_7 13_H **Reserved**: Reserved, do not use 14_H **CDTM2_DTM0_2**: Dead-time output of TOM2_2 15_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=4)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT64 (P20.8) 01_H **CDTM0_DTM4_0**: Dead-time output of ATOM0_0 02_H **CDTM1_DTM4_0_N**: Inverted dead-time output of ATOM1_0 03_H **CDTM2_DTM4_0**: Dead-time output of ATOM2_0 04_H **CDTM0_DTM0_0**: Dead-time output of TOM0_0 05_H **CDTM1_DTM0_0_N**: Inverted dead-time output of TOM1_0 06_H **CDTM2_DTM0_0**: Dead-time output of TOM2_0 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM1_DTM1_3**: Dead-time output of TOM1_7 11_H **CDTM2_DTM1_3**: Dead-time output of TOM2_7 12_H **CDTM0_DTM5_3**: Dead-time output of ATOM0_7 13_H **Reserved**: Reserved, do not use 14_H **CDTM2_DTM0_1**: Dead-time output of TOM2_1 15_H **CDTM0_DTM1_0**: Dead-time output of TOM0_4 16_H **CDTM1_DTM1_0**: Dead-time output of TOM1_4 17_H **CDTM2_DTM1_0**: Dead-time output of TOM2_4 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=5)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT65 (P20.9) 01_H **CDTM0_DTM4_0_N**: Inverted dead-time output of ATOM0_0 02_H **CDTM1_DTM4_0**: Dead-time output of ATOM1_0 03_H **CDTM2_DTM4_0_N**: Inverted dead-time output of ATOM2_0 04_H **CDTM0_DTM0_0_N**: Inverted dead-time output of TOM0_0 05_H **CDTM1_DTM0_0**: Dead-time output of TOM1_0 06_H **CDTM2_DTM0_0_N**: Inverted dead-time output of TOM2_0 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM1_DTM3_1**: Dead-time output of TOM1_13 11_H **CDTM2_DTM3_1**: Dead-time output of TOM2_13 12_H **CDTM1_DTM5_1**: Dead-time output of ATOM1_5 13_H **Reserved**: Reserved, do not use 14_H **CDTM2_DTM0_0**: Dead-time output of TOM2_0 15_H **CDTM0_DTM1_0_N**: Inverted dead-time output of TOM0_4 16_H **CDTM1_DTM1_0_N**: Inverted dead-time output of TOM1_4 17_H **CDTM2_DTM1_0**: Dead-time output of TOM2_4 18_H **Reserved**: Reserved, do not use 19_H **CDTM2_DTM5_1**: Dead-time output of ATOM2_5 $1A_H$ **CDTM0_DTM4_2_N**: Inverted dead-time output of ATOM0_2 $1B_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	15, 31	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.21 eGTM to Ports signal TOUT output selection register n (n=11)

For a description of this register, see 7579.

EGTM_TOUTSELn (n=11) Offset address: 061100_H+n*4

eGTM to Ports signal TOUT output selection register n Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	SEL5					SEL4					SEL3				
r	rw					rw					rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	SEL2					SEL1					SEL0				
r	rw					rw					rw				

Field	Bits	Type	Description
SELx (x=0)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT66 (P20.10) 01_H **CDTM0_DTM4_1**: Dead-time output of ATOM0_1 02_H **CDTM1_DTM4_1_N**: Inverted dead-time output of ATOM1_1 03_H **CDTM2_DTM4_1**: Dead-time output of ATOM2_1 04_H **CDTM0_DTM0_1**: Dead-time output of TOM0_1 05_H **CDTM1_DTM0_1_N**: Inverted dead-time output of TOM1_1 06_H **CDTM2_DTM0_1**: Dead-time output of TOM2_1 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM1_DTM3_2**: Dead-time output of TOM1_14 11_H **CDTM2_DTM3_2**: Dead-time output of TOM2_14 12_H **CDTM1_DTM5_2**: Dead-time output of ATOM1_6 13_H **Reserved**: Reserved, do not use 14_H **CDTM2_DTM0_1**: Dead-time output of TOM2_1 15_H **CDTM0_DTM1_1**: Dead-time output of TOM0_5 16_H **CDTM1_DTM1_1**: Dead-time output of TOM1_5 17_H **CDTM2_DTM1_1_N**: Inverted dead-time output of TOM2_5 18_H **Reserved**: Reserved, do not use 19_H **CDTM2_DTM5_2**: Dead-time output of ATOM2_6 $1A_H$ **CDTM0_DTM4_3_N**: Inverted dead-time output of ATOM0_3 $1B_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=1)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT67 (P20.11) 01_H **CDTM0_DTM4_1_N**: Inverted dead-time output of ATOM0_1 02_H **CDTM1_DTM4_1**: Dead-time output of ATOM1_1 03_H **CDTM2_DTM4_1_N**: Inverted dead-time output of ATOM2_1 04_H **CDTM0_DTM0_1_N**: Inverted dead-time output of TOM0_1 05_H **CDTM1_DTM0_1**: Dead-time output of TOM1_1 06_H **CDTM2_DTM0_1_N**: Inverted dead-time output of TOM2_1 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM1_DTM3_3**: Dead-time output of TOM1_15 11_H **CDTM2_DTM3_3**: Dead-time output of TOM2_15 12_H **CDTM1_DTM5_3**: Dead-time output of ATOM1_7 13_H **Reserved**: Reserved, do not use 14_H **CDTM2_DTM0_2_N**: Inverted dead-time output of TOM2_2 15_H **CDTM0_DTM1_1_N**: Inverted dead-time output of TOM0_5 16_H **CDTM1_DTM1_1_N**: Inverted dead-time output of TOM1_5 17_H **CDTM2_DTM1_2**: Dead-time output of TOM2_6 18_H **Reserved**: Reserved, do not use 19_H **CDTM2_DTM5_3_N**: Inverted dead-time output of ATOM2_7 $1A_H$ **CDTM0_DTM5_0**: Dead-time output of ATOM0_4 $1B_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=2)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT68 (P20.12) 01_H **CDTM0_DTM4_2**: Dead-time output of ATOM0_2 02_H **CDTM1_DTM4_2_N**: Inverted dead-time output of ATOM1_2 03_H **CDTM2_DTM4_2**: Dead-time output of ATOM2_2 04_H **CDTM0_DTM0_2**: Dead-time output of TOM0_2 05_H **CDTM1_DTM0_2_N**: Inverted dead-time output of TOM1_2 06_H **CDTM2_DTM0_2**: Dead-time output of TOM2_2 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM1_DTM0_0**: Dead-time output of TOM1_0 11_H **CDTM2_DTM2_0**: Dead-time output of TOM2_8 12_H **CDTM1_DTM4_0**: Dead-time output of ATOM1_0 13_H **Reserved**: Reserved, do not use 14_H **CDTM0_DTM1_1_N**: Inverted dead-time output of TOM0_5 15_H **CDTM0_DTM1_2**: Dead-time output of TOM0_6 16_H **CDTM1_DTM1_2**: Dead-time output of TOM1_6 17_H **CDTM2_DTM0_3**: Dead-time output of TOM2_3 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
SELx (x=3)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT69 (P20.13) 01_H **CDTM0_DTM4_2_N**: Inverted dead-time output of ATOM0_2 02_H **CDTM1_DTM4_2**: Dead-time output of ATOM1_2 03_H **CDTM2_DTM4_2_N**: Inverted dead-time output of ATOM2_2 04_H **CDTM0_DTM0_2_N**: Inverted dead-time output of TOM0_2 05_H **CDTM1_DTM0_2**: Dead-time output of TOM1_2 06_H **CDTM2_DTM0_2_N**: Inverted dead-time output of TOM2_2 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM1_DTM0_1**: Dead-time output of TOM1_1 11_H **CDTM2_DTM2_1**: Dead-time output of TOM2_9 12_H **CDTM1_DTM4_1**: Dead-time output of ATOM1_1 13_H **Reserved**: Reserved, do not use 14_H **Reserved**: Reserved, do not use 15_H **CDTM0_DTM1_2_N**: Inverted dead-time output of TOM0_6 16_H **CDTM1_DTM1_2_N**: Inverted dead-time output of TOM1_6 17_H **CDTM2_DTM1_0**: Dead-time output of TOM2_4 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=4)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT70 (P20.14) 01_H **CDTM0_DTM4_3**: Dead-time output of ATOM0_3 02_H **CDTM1_DTM4_3_N**: Inverted dead-time output of ATOM1_3 03_H **CDTM2_DTM4_3**: Dead-time output of ATOM2_3 04_H **CDTM0_DTM0_3**: Dead-time output of TOM0_3 05_H **CDTM1_DTM0_3_N**: Inverted dead-time output of TOM1_3 06_H **CDTM2_DTM0_3**: Dead-time output of TOM2_3 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM1_DTM0_2**: Dead-time output of TOM1_2 11_H **CDTM2_DTM2_2**: Dead-time output of TOM2_10 12_H **CDTM1_DTM4_2**: Dead-time output of ATOM1_2 13_H **Reserved**: Reserved, do not use 14_H **Reserved**: Reserved, do not use 15_H **CDTM0_DTM1_3**: Dead-time output of TOM0_7 16_H **CDTM1_DTM1_3**: Dead-time output of TOM1_7 17_H **CDTM2_DTM1_3**: Dead-time output of TOM2_7 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=5)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT71 (P15.0) 01_H **CDTM0_DTM4_3_N**: Inverted dead-time output of ATOM0_3 02_H **CDTM1_DTM4_3**: Dead-time output of ATOM1_3 03_H **CDTM2_DTM4_3_N**: Inverted dead-time output of ATOM2_3 04_H **CDTM0_DTM0_3_N**: Inverted dead-time output of TOM0_3 05_H **CDTM1_DTM0_3**: Dead-time output of TOM1_3 06_H **CDTM2_DTM0_3_N**: Inverted dead-time output of TOM2_3 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM1_DTM0_3**: Dead-time output of TOM1_3 11_H **CDTM2_DTM2_3**: Dead-time output of TOM2_11 12_H **CDTM1_DTM4_3**: Dead-time output of ATOM1_3 13_H **Reserved**: Reserved, do not use 14_H **CDTM0_DTM0_3**: Dead-time output of TOM0_3 15_H **CDTM1_DTM1_0_N**: Inverted dead-time output of TOM1_4 16_H **CDTM0_DTM1_3_N**: Inverted dead-time output of TOM0_7 17_H **CDTM1_DTM1_3_N**: Inverted dead-time output of TOM1_7 18_H **Reserved**: Reserved, do not use 19_H **CDTM1_DTM5_0_N**: Inverted dead-time output of ATOM1_4 $1A_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	15, 31	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.22 eGTM to Ports signal TOUT output selection register n (n=12)

For a description of this register, see 7579.

EGTM_TOUTSELn (n=12) Offset address: 061100_H+n*4

eGTM to Ports signal TOUT output selection register n Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	SEL5					SEL4					SEL3				
r	rw					rw					rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	SEL2					SEL1					SEL0				
r	rw					rw					rw				

Field	Bits	Type	Description
SELx (x=0)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT72 (P15.1) 01_H **CDTM0_DTM5_0**: Dead-time output of ATOM0_4 02_H **CDTM1_DTM5_0_N**: Inverted dead-time output of ATOM1_4 03_H **CDTM2_DTM5_0**: Dead-time output of ATOM2_4 04_H **CDTM0_DTM1_0**: Dead-time output of TOM0_4 05_H **CDTM1_DTM1_0_N**: Inverted dead-time output of TOM1_4 06_H **CDTM2_DTM1_0**: Dead-time output of TOM2_4 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM1_DTM1_0**: Dead-time output of TOM1_4 11_H **CDTM2_DTM3_0**: Dead-time output of TOM2_12 12_H **CDTM1_DTM5_0**: Dead-time output of ATOM1_4 13_H **Reserved**: Reserved, do not use 14_H **CDTM0_DTM0_0_N**: Inverted dead-time output of TOM0_0 15_H **CDTM1_DTM0_1_N**: Inverted dead-time output of TOM1_1 16_H **CDTM0_DTM1_0**: Dead-time output of TOM0_4 17_H **CDTM1_DTM1_0**: Dead-time output of TOM1_4 18_H **Reserved**: Reserved, do not use 19_H **CDTM1_DTM4_1_N**: Inverted dead-time output of ATOM1_1 $1A_H$ **CDTM2_DTM4_1**: Dead-time output of ATOM2_1 $1B_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=1)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT73 (P15.2) 01_H **CDTM0_DTM5_0_N**: Inverted dead-time output of ATOM0_4 02_H **CDTM1_DTM5_0**: Dead-time output of ATOM1_4 03_H **CDTM2_DTM5_0_N**: Inverted dead-time output of ATOM2_4 04_H **CDTM0_DTM1_0_N**: Inverted dead-time output of TOM0_4 05_H **CDTM1_DTM1_0**: Dead-time output of TOM1_4 06_H **CDTM2_DTM1_0_N**: Inverted dead-time output of TOM2_4 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM1_DTM1_1**: Dead-time output of TOM1_5 11_H **CDTM2_DTM3_1**: Dead-time output of TOM2_13 12_H **CDTM1_DTM5_1**: Dead-time output of ATOM1_5 13_H **Reserved**: Reserved, do not use 14_H **CDTM0_DTM0_1_N**: Inverted dead-time output of TOM0_1 15_H **CDTM1_DTM0_2_N**: Inverted dead-time output of TOM1_2 16_H **CDTM0_DTM1_0_N**: Inverted dead-time output of TOM0_4 17_H **CDTM1_DTM1_0_N**: Inverted dead-time output of TOM1_4 18_H **Reserved**: Reserved, do not use 19_H **CDTM1_DTM4_2_N**: Inverted dead-time output of ATOM1_2 $1A_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=2)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT74 (P15.3) 01_H **CDTM0_DTM5_1**: Dead-time output of ATOM0_5 02_H **CDTM1_DTM5_1_N**: Inverted dead-time output of ATOM1_5 03_H **CDTM2_DTM5_1**: Dead-time output of ATOM2_5 04_H **CDTM0_DTM1_1**: Dead-time output of TOM0_5 05_H **CDTM1_DTM1_1_N**: Inverted dead-time output of TOM1_5 06_H **CDTM2_DTM1_1**: Dead-time output of TOM2_5 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM1_DTM1_2**: Dead-time output of TOM1_6 11_H **CDTM2_DTM3_2**: Dead-time output of TOM2_14 12_H **CDTM1_DTM5_2**: Dead-time output of ATOM1_6 13_H **Reserved**: Reserved, do not use 14_H **CDTM0_DTM0_2_N**: Inverted dead-time output of TOM0_2 15_H **CDTM1_DTM0_3_N**: Inverted dead-time output of TOM1_3 16_H **CDTM0_DTM1_1**: Dead-time output of TOM0_5 17_H **CDTM1_DTM1_1**: Dead-time output of TOM1_5 18_H **Reserved**: Reserved, do not use 19_H **CDTM1_DTM4_3_N**: Inverted dead-time output of ATOM1_3 $1A_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=3)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT75 (P15.4) 01_H **CDTM0_DTM5_1_N**: Inverted dead-time output of ATOM0_5 02_H **CDTM1_DTM5_1**: Dead-time output of ATOM1_5 03_H **CDTM2_DTM5_1_N**: Inverted dead-time output of ATOM2_5 04_H **CDTM0_DTM1_1_N**: Inverted dead-time output of TOM0_5 05_H **CDTM1_DTM1_1**: Dead-time output of TOM1_5 06_H **CDTM2_DTM1_1_N**: Inverted dead-time output of TOM2_5 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM1_DTM1_3**: Dead-time output of TOM1_7 11_H **CDTM2_DTM3_3**: Dead-time output of TOM2_15 12_H **CDTM1_DTM5_3**: Dead-time output of ATOM1_7 13_H **Reserved**: Reserved, do not use 14_H **CDTM0_DTM0_3_N**: Inverted dead-time output of TOM0_3 15_H **CDTM1_DTM0_0_N**: Inverted dead-time output of TOM1_0 16_H **CDTM0_DTM1_2**: Dead-time output of TOM0_6 17_H **CDTM1_DTM1_2**: Dead-time output of TOM1_6 18_H **Reserved**: Reserved, do not use 19_H **CDTM1_DTM5_0_N**: Inverted dead-time output of ATOM1_4 $1A_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=4)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT76 (P15.5) 01_H **CDTM0_DTM5_2**: Dead-time output of ATOM0_6 02_H **CDTM1_DTM5_2_N**: Inverted dead-time output of ATOM1_6 03_H **CDTM2_DTM5_2**: Dead-time output of ATOM2_6 04_H **CDTM0_DTM1_2**: Dead-time output of TOM0_6 05_H **CDTM1_DTM1_2_N**: Inverted dead-time output of TOM1_6 06_H **CDTM2_DTM1_2**: Dead-time output of TOM2_6 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM0_0**: Dead-time output of TOM0_0 11_H **CDTM1_DTM0_0**: Dead-time output of TOM1_0 12_H **CDTM1_DTM4_0**: Dead-time output of ATOM1_0 13_H **Reserved**: Reserved, do not use 14_H **CDTM2_DTM0_0_N**: Inverted dead-time output of TOM2_0 15_H **CDTM1_DTM0_0_N**: Inverted dead-time output of TOM1_0 16_H **CDTM0_DTM1_1_N**: Inverted dead-time output of TOM0_5 17_H **CDTM1_DTM1_1_N**: Inverted dead-time output of TOM1_5 18_H **Reserved**: Reserved, do not use 19_H **CDTM1_DTM5_1_N**: Inverted dead-time output of ATOM1_5 $1A_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=5)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT77 (P15.6) 01_H **CDTM0_DTM5_2_N**: Inverted dead-time output of ATOM0_6 02_H **CDTM1_DTM5_2**: Dead-time output of ATOM1_6 03_H **CDTM2_DTM5_2_N**: Inverted dead-time output of ATOM2_6 04_H **CDTM0_DTM1_2_N**: Inverted dead-time output of TOM0_6 05_H **CDTM1_DTM1_2**: Dead-time output of TOM1_6 06_H **CDTM2_DTM1_2_N**: Inverted dead-time output of TOM2_6 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM0_0**: Dead-time output of TOM0_0 11_H **CDTM1_DTM0_0**: Dead-time output of TOM1_0 12_H **CDTM1_DTM4_0**: Dead-time output of ATOM1_0 13_H **Reserved**: Reserved, do not use 14_H **CDTM2_DTM0_3_N**: Inverted dead-time output of TOM2_3 15_H **CDTM0_DTM1_1_N**: Inverted dead-time output of TOM0_5 16_H **CDTM1_DTM1_1_N**: Inverted dead-time output of TOM1_5 17_H **CDTM2_DTM1_1**: Dead-time output of TOM2_5 18_H **Reserved**: Reserved, do not use 19_H **CDTM1_DTM5_1_N**: Inverted dead-time output of ATOM1_5 $1A_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	15, 31	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.23 eGTM to Ports signal TOUT output selection register n (n=13)

For a description of this register, see 7579.

EGTM_TOUTSELn (n=13) Offset address: 061100_H+n*4

eGTM to Ports signal TOUT output selection register n Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	SEL5					SEL4					SEL3				
r	rw					rw					rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	SEL2					SEL1					SEL0				
r	rw					rw					rw				

Field	Bits	Type	Description
SELx (x=0)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT78 (P15.7) 01_H **CDTM0_DTM5_3**: Dead-time output of ATOM0_7 02_H **CDTM1_DTM5_3_N**: Inverted dead-time output of ATOM1_7 03_H **CDTM2_DTM5_3**: Dead-time output of ATOM2_7 04_H **CDTM0_DTM1_3**: Dead-time output of TOM0_7 05_H **CDTM1_DTM1_3_N**: Inverted dead-time output of TOM1_7 06_H **CDTM2_DTM1_3**: Dead-time output of TOM2_7 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM0_1**: Dead-time output of TOM0_1 11_H **CDTM1_DTM0_1**: Dead-time output of TOM1_1 12_H **CDTM1_DTM4_1**: Dead-time output of ATOM1_1 13_H **Reserved**: Reserved, do not use 14_H **CDTM2_DTM0_2_N**: Inverted dead-time output of TOM2_2 15_H **CDTM0_DTM1_2_N**: Inverted dead-time output of TOM0_6 16_H **CDTM1_DTM1_2_N**: Inverted dead-time output of TOM1_6 17_H **CDTM2_DTM1_2**: Dead-time output of TOM2_6 18_H **Reserved**: Reserved, do not use 19_H **CDTM1_DTM5_2_N**: Inverted dead-time output of ATOM1_6 $1A_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=1)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT79 (P15.8) 01_H **CDTM0_DTM5_3_N**: Inverted dead-time output of ATOM0_7 02_H **CDTM1_DTM5_3**: Dead-time output of ATOM1_7 03_H **CDTM2_DTM5_3_N**: Inverted dead-time output of ATOM2_7 04_H **CDTM0_DTM1_3_N**: Inverted dead-time output of TOM0_7 05_H **CDTM1_DTM1_3**: Dead-time output of TOM1_7 06_H **CDTM2_DTM1_3_N**: Inverted dead-time output of TOM2_7 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM0_2**: Dead-time output of TOM0_2 11_H **CDTM1_DTM0_2**: Dead-time output of TOM1_2 12_H **CDTM1_DTM4_1**: Dead-time output of ATOM1_1 13_H **Reserved**: Reserved, do not use 14_H **CDTM2_DTM0_1_N**: Inverted dead-time output of TOM2_1 15_H **CDTM0_DTM1_3_N**: Inverted dead-time output of TOM0_7 16_H **CDTM1_DTM1_3_N**: Inverted dead-time output of TOM1_7 17_H **CDTM2_DTM1_3_N**: Inverted dead-time output of TOM2_7 18_H **Reserved**: Reserved, do not use 19_H **CDTM1_DTM5_3_N**: Inverted dead-time output of ATOM1_7 $1A_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=2)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT80 (P14.0) 01_H **CDTM0_DTM4_0**: Dead-time output of ATOM0_0 02_H **CDTM1_DTM4_0_N**: Inverted dead-time output of ATOM1_0 03_H **CDTM2_DTM4_0**: Dead-time output of ATOM2_0 04_H **CDTM0_DTM2_0**: Dead-time output of TOM0_8 05_H **CDTM1_DTM2_0_N**: Inverted dead-time output of TOM1_8 06_H **CDTM2_DTM2_0**: Dead-time output of TOM2_8 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM0_3**: Dead-time output of TOM0_3 11_H **CDTM1_DTM0_3**: Dead-time output of TOM1_3 12_H **CDTM1_DTM4_2**: Dead-time output of ATOM1_2 13_H **Reserved**: Reserved, do not use 14_H **CDTM0_DTM1_2**: Dead-time output of TOM0_6 15_H **CDTM1_DTM1_2**: Dead-time output of TOM1_6 16_H **CDTM2_DTM1_2**: Dead-time output of TOM2_6 17_H **Reserved**: Reserved, do not use 18_H **Reserved**: Reserved, do not use 19_H **CDTM1_DTM4_3_N**: Inverted dead-time output of ATOM1_3 $1A_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=3)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT81 (P14.1) 01_H **CDTM0_DTM4_0_N**: Inverted dead-time output of ATOM0_0 02_H **CDTM1_DTM4_0**: Dead-time output of ATOM1_0 03_H **CDTM2_DTM4_0_N**: Inverted dead-time output of ATOM2_0 04_H **CDTM0_DTM2_0_N**: Inverted dead-time output of TOM0_8 05_H **CDTM1_DTM2_0**: Dead-time output of TOM1_8 06_H **CDTM2_DTM2_0_N**: Inverted dead-time output of TOM2_8 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM1_0**: Dead-time output of TOM0_4 11_H **CDTM1_DTM1_0**: Dead-time output of TOM1_4 12_H **CDTM0_DTM5_0**: Dead-time output of ATOM0_4 13_H **Reserved**: Reserved, do not use 14_H **CDTM0_DTM1_3**: Dead-time output of TOM0_7 15_H **CDTM1_DTM1_3**: Dead-time output of TOM1_7 16_H **CDTM2_DTM1_3**: Dead-time output of TOM2_7 17_H **Reserved**: Reserved, do not use 18_H **Reserved**: Reserved, do not use 19_H **CDTM1_DTM5_0_N**: Inverted dead-time output of ATOM1_4 $1A_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=4)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT82 (P14.2) 01_H **CDTM0_DTM4_1**: Dead-time output of ATOM0_1 02_H **CDTM1_DTM4_1_N**: Inverted dead-time output of ATOM1_1 03_H **CDTM2_DTM4_1**: Dead-time output of ATOM2_1 04_H **CDTM0_DTM2_1**: Dead-time output of TOM0_9 05_H **CDTM1_DTM2_1_N**: Inverted dead-time output of TOM1_9 06_H **CDTM2_DTM2_1**: Dead-time output of TOM2_9 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM1_1**: Dead-time output of TOM0_5 11_H **CDTM1_DTM1_1**: Dead-time output of TOM1_5 12_H **CDTM0_DTM4_3**: Dead-time output of ATOM0_3 13_H **Reserved**: Reserved, do not use 14_H **CDTM0_DTM1_2_N**: Inverted dead-time output of TOM0_6 15_H **CDTM1_DTM1_2_N**: Inverted dead-time output of TOM1_6 16_H **CDTM2_DTM1_2**: Dead-time output of TOM2_6 17_H **Reserved**: Reserved, do not use 18_H **Reserved**: Reserved, do not use 19_H **CDTM1_DTM5_1_N**: Inverted dead-time output of ATOM1_5 $1A_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=5)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT83 (P14.3) 01_H **CDTM0_DTM4_1_N**: Inverted dead-time output of ATOM0_1 02_H **CDTM1_DTM4_1**: Dead-time output of ATOM1_1 03_H **CDTM2_DTM4_1_N**: Inverted dead-time output of ATOM2_1 04_H **CDTM0_DTM2_1_N**: Inverted dead-time output of TOM0_9 05_H **CDTM1_DTM2_1**: Dead-time output of TOM1_9 06_H **CDTM2_DTM2_1_N**: Inverted dead-time output of TOM2_9 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM1_2**: Dead-time output of TOM0_6 11_H **CDTM1_DTM1_2**: Dead-time output of TOM1_6 12_H **CDTM0_DTM4_2**: Dead-time output of ATOM0_2 13_H **Reserved**: Reserved, do not use 14_H **CDTM0_DTM1_1**: Dead-time output of TOM0_5 15_H **CDTM1_DTM1_1**: Dead-time output of TOM1_5 16_H **CDTM2_DTM1_1**: Dead-time output of TOM2_5 17_H **Reserved**: Reserved, do not use 18_H **Reserved**: Reserved, do not use 19_H **CDTM1_DTM5_2_N**: Inverted dead-time output of ATOM1_6 $1A_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	15, 31	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.24 eGTM to Ports signal TOUT output selection register n (n=14)

For a description of this register, see 7579.

EGTM_TOUTSELn (n=14) Offset address: 061100_H+n*4

eGTM to Ports signal TOUT output selection register n Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	SEL5					SEL4					SEL3				
r	rw					rw					rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	SEL2					SEL1					SEL0				
r	rw					rw					rw				

Field	Bits	Type	Description
SELx (x=0)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT84 (P14.4) 01_H **CDTM0_DTM4_2**: Dead-time output of ATOM0_2 02_H **CDTM1_DTM4_2_N**: Inverted dead-time output of ATOM1_2 03_H **CDTM2_DTM4_2**: Dead-time output of ATOM2_2 04_H **CDTM0_DTM2_2**: Dead-time output of TOM0_10 05_H **CDTM1_DTM2_2_N**: Inverted dead-time output of TOM1_10 06_H **CDTM2_DTM2_2**: Dead-time output of TOM2_10 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM1_3**: Dead-time output of TOM0_7 11_H **CDTM1_DTM1_3**: Dead-time output of TOM1_7 12_H **CDTM0_DTM4_1**: Dead-time output of ATOM0_1 13_H **Reserved**: Reserved, do not use 14_H **CDTM0_DTM1_3_N**: Inverted dead-time output of TOM0_7 15_H **CDTM1_DTM1_3_N**: Inverted dead-time output of TOM1_7 16_H **CDTM2_DTM1_3**: Dead-time output of TOM2_7 17_H **Reserved**: Reserved, do not use 18_H **Reserved**: Reserved, do not use 19_H **CDTM1_DTM5_3_N**: Inverted dead-time output of ATOM1_7 $1A_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=1)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT85 (P14.5) 01_H **CDTM0_DTM4_2_N**: Inverted dead-time output of ATOM0_2 02_H **CDTM1_DTM4_2**: Dead-time output of ATOM1_2 03_H **CDTM2_DTM4_2_N**: Inverted dead-time output of ATOM2_2 04_H **CDTM0_DTM2_2_N**: Inverted dead-time output of TOM0_10 05_H **CDTM1_DTM2_2**: Dead-time output of TOM1_10 06_H **CDTM2_DTM2_2_N**: Inverted dead-time output of TOM2_10 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM0_0**: Dead-time output of TOM0_0 11_H **CDTM1_DTM0_0**: Dead-time output of TOM1_0 12_H **CDTM0_DTM4_0**: Dead-time output of ATOM0_0 13_H **Reserved**: Reserved, do not use 14_H **CDTM0_DTM1_3_N**: Inverted dead-time output of TOM0_7 15_H **CDTM1_DTM1_3_N**: Inverted dead-time output of TOM1_7 16_H **CDTM2_DTM1_3**: Dead-time output of TOM2_7 17_H **Reserved**: Reserved, do not use 18_H **Reserved**: Reserved, do not use 19_H **CDTM1_DTM4_0_N**: Inverted dead-time output of ATOM1_0 $1A_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=2)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT86 (P14.6) 01_H **CDTM0_DTM4_3**: Dead-time output of ATOM0_3 02_H **CDTM1_DTM4_3_N**: Inverted dead-time output of ATOM1_3 03_H **CDTM2_DTM4_3**: Dead-time output of ATOM2_3 04_H **CDTM0_DTM2_3**: Dead-time output of TOM0_11 05_H **CDTM1_DTM2_3_N**: Inverted dead-time output of TOM1_11 06_H **CDTM2_DTM2_3**: Dead-time output of TOM2_11 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM0_1**: Dead-time output of TOM0_1 11_H **CDTM1_DTM0_1**: Dead-time output of TOM1_1 12_H **CDTM1_DTM4_1**: Dead-time output of ATOM1_1 13_H **Reserved**: Reserved, do not use 14_H **CDTM0_DTM1_2_N**: Inverted dead-time output of TOM0_6 15_H **CDTM2_DTM1_2_N**: Inverted dead-time output of TOM2_6 16_H **Reserved**: Reserved, do not use 17_H **Reserved**: Reserved, do not use 18_H **CDTM2_DTM5_0_N**: Inverted dead-time output of ATOM2_4 19_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=3)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT87 (P14.7) 01_H **CDTM0_DTM4_3_N**: Inverted dead-time output of ATOM0_3 02_H **CDTM1_DTM4_3**: Dead-time output of ATOM1_3 03_H **CDTM2_DTM4_3_N**: Inverted dead-time output of ATOM2_3 04_H **CDTM0_DTM2_3_N**: Inverted dead-time output of TOM0_11 05_H **CDTM1_DTM2_3**: Dead-time output of TOM1_11 06_H **CDTM2_DTM2_3_N**: Inverted dead-time output of TOM2_11 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM0_0**: Dead-time output of TOM0_0 11_H **CDTM2_DTM0_0**: Dead-time output of TOM2_0 12_H **CDTM1_DTM4_0**: Dead-time output of ATOM1_0 13_H **Reserved**: Reserved, do not use 14_H **CDTM0_DTM1_1_N**: Inverted dead-time output of TOM0_5 15_H **CDTM2_DTM1_1_N**: Inverted dead-time output of TOM2_5 16_H **Reserved**: Reserved, do not use 17_H **Reserved**: Reserved, do not use 18_H **CDTM2_DTM5_1_N**: Inverted dead-time output of ATOM2_5 19_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=4)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT88 (P14.8) 01_H **CDTM0_DTM5_0**: Dead-time output of ATOM0_4 02_H **CDTM1_DTM5_0_N**: Inverted dead-time output of ATOM1_4 03_H **CDTM2_DTM5_0**: Dead-time output of ATOM2_4 04_H **CDTM0_DTM3_0**: Dead-time output of TOM0_12 05_H **CDTM1_DTM3_0_N**: Inverted dead-time output of TOM1_12 06_H **CDTM2_DTM3_0**: Dead-time output of TOM2_12 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM0_2**: Dead-time output of TOM0_2 11_H **CDTM2_DTM0_2**: Dead-time output of TOM2_2 12_H **CDTM2_DTM4_2**: Dead-time output of ATOM2_2 13_H **Reserved**: Reserved, do not use 14_H **CDTM0_DTM1_3_N**: Inverted dead-time output of TOM0_7 15_H **CDTM2_DTM1_3_N**: Inverted dead-time output of TOM2_7 16_H **Reserved**: Reserved, do not use 17_H **Reserved**: Reserved, do not use 18_H **CDTM2_DTM5_2_N**: Inverted dead-time output of ATOM2_6 19_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=5)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT89 (P14.9) 01_H **CDTM0_DTM5_0_N**: Inverted dead-time output of ATOM0_4 02_H **CDTM1_DTM5_0**: Dead-time output of ATOM1_4 03_H **CDTM2_DTM5_0_N**: Inverted dead-time output of ATOM2_4 04_H **CDTM0_DTM3_0_N**: Inverted dead-time output of TOM0_12 05_H **CDTM1_DTM3_0**: Dead-time output of TOM1_12 06_H **CDTM2_DTM3_0_N**: Inverted dead-time output of TOM2_12 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM0_3**: Dead-time output of TOM0_3 11_H **CDTM2_DTM0_3**: Dead-time output of TOM2_3 12_H **CDTM2_DTM4_3**: Dead-time output of ATOM2_3 13_H **Reserved**: Reserved, do not use 14_H **CDTM0_DTM1_0_N**: Inverted dead-time output of TOM0_4 15_H **CDTM2_DTM1_0_N**: Inverted dead-time output of TOM2_4 16_H **Reserved**: Reserved, do not use 17_H **Reserved**: Reserved, do not use 18_H **CDTM2_DTM5_3_N**: Inverted dead-time output of ATOM2_7 19_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	15, 31	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.25 eGTM to Ports signal TOUT output selection register n (n=15)

For a description of this register, see 7579.

EGTM_TOUTSELn (n=15) Offset address: 061100_H+n*4

eGTM to Ports signal TOUT output selection register n Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	SEL5					SEL4					SEL3				
r	rw					rw					rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	SEL2					SEL1					SEL0				
r	rw					rw					rw				

Field	Bits	Type	Description
SELx (x=0)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT90 (P14.10) 01_H **CDTM0_DTM5_1**: Dead-time output of ATOM0_5 02_H **CDTM1_DTM5_1_N**: Inverted dead-time output of ATOM1_5 03_H **CDTM2_DTM5_1**: Dead-time output of ATOM2_5 04_H **CDTM0_DTM3_1**: Dead-time output of TOM0_13 05_H **CDTM1_DTM3_1_N**: Inverted dead-time output of TOM1_13 06_H **CDTM2_DTM3_1**: Dead-time output of TOM2_13 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM1_0**: Dead-time output of TOM0_4 11_H **CDTM2_DTM1_0**: Dead-time output of TOM2_4 12_H **CDTM2_DTM5_0**: Dead-time output of ATOM2_4 13_H **Reserved**: Reserved, do not use 14_H **CDTM0_DTM0_1_N**: Inverted dead-time output of TOM0_1 15_H **CDTM2_DTM0_1_N**: Inverted dead-time output of TOM2_1 16_H **Reserved**: Reserved, do not use 17_H **Reserved**: Reserved, do not use 18_H **CDTM2_DTM4_1**: Dead-time output of ATOM2_1 19_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=1)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT91 (P13.0) 01_H **CDTM0_DTM5_1_N**: Inverted dead-time output of ATOM0_5 02_H **CDTM1_DTM5_1**: Dead-time output of ATOM1_5 03_H **CDTM2_DTM5_1_N**: Inverted dead-time output of ATOM2_5 04_H **CDTM0_DTM3_1_N**: Inverted dead-time output of TOM0_13 05_H **CDTM1_DTM3_1**: Dead-time output of TOM1_13 06_H **CDTM2_DTM3_1_N**: Inverted dead-time output of TOM2_13 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM1_1**: Dead-time output of TOM0_5 11_H **CDTM2_DTM1_1**: Dead-time output of TOM2_5 12_H **CDTM2_DTM5_1**: Dead-time output of ATOM2_5 13_H **Reserved**: Reserved, do not use 14_H **CDTM0_DTM0_0_N**: Inverted dead-time output of TOM0_0 15_H **CDTM2_DTM0_0_N**: Inverted dead-time output of TOM2_0 16_H **CDTM0_DTM1_2_N**: Inverted dead-time output of TOM0_6 17_H **CDTM1_DTM1_2_N**: Inverted dead-time output of TOM1_6 18_H **CDTM2_DTM4_1_N**: Inverted dead-time output of ATOM2_1 19_H **Reserved**: Reserved, do not use $1A_H$ **CDTM2_DTM4_1_N**: Inverted dead-time output of ATOM2_1 $1B_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=2)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT92 (P13.1) 01_H **CDTM0_DTM5_2**: Dead-time output of ATOM0_6 02_H **CDTM1_DTM5_2_N**: Inverted dead-time output of ATOM1_6 03_H **CDTM2_DTM5_2**: Dead-time output of ATOM2_6 04_H **CDTM0_DTM3_2**: Dead-time output of TOM0_14 05_H **CDTM1_DTM3_2_N**: Inverted dead-time output of TOM1_14 06_H **CDTM2_DTM3_2**: Dead-time output of TOM2_14 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM1_2**: Dead-time output of TOM0_6 11_H **CDTM2_DTM1_2**: Dead-time output of TOM2_6 12_H **CDTM2_DTM5_2**: Dead-time output of ATOM2_6 13_H **Reserved**: Reserved, do not use 14_H **CDTM0_DTM0_1_N**: Inverted dead-time output of TOM0_1 15_H **CDTM2_DTM0_1_N**: Inverted dead-time output of TOM2_1 16_H **CDTM0_DTM1_3**: Dead-time output of TOM0_7 17_H **CDTM1_DTM1_3**: Dead-time output of TOM1_7 18_H **CDTM2_DTM4_2_N**: Inverted dead-time output of ATOM2_2 19_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=3)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT93 (P13.2) 01_H **CDTM0_DTM5_2_N**: Inverted dead-time output of ATOM0_6 02_H **CDTM1_DTM5_2**: Dead-time output of ATOM1_6 03_H **CDTM2_DTM5_2_N**: Inverted dead-time output of ATOM2_6 04_H **CDTM0_DTM3_2_N**: Inverted dead-time output of TOM0_14 05_H **CDTM1_DTM3_2**: Dead-time output of TOM1_14 06_H **CDTM2_DTM3_2_N**: Inverted dead-time output of TOM2_14 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM1_3**: Dead-time output of TOM0_7 11_H **CDTM2_DTM1_3**: Dead-time output of TOM2_7 12_H **CDTM2_DTM5_3**: Dead-time output of ATOM2_7 13_H **Reserved**: Reserved, do not use 14_H **CDTM0_DTM0_2_N**: Inverted dead-time output of TOM0_2 15_H **CDTM2_DTM0_2_N**: Inverted dead-time output of TOM2_2 16_H **CDTM0_DTM1_3_N**: Inverted dead-time output of TOM0_7 17_H **CDTM1_DTM1_3_N**: Inverted dead-time output of TOM1_7 18_H **CDTM2_DTM4_3_N**: Inverted dead-time output of ATOM2_3 19_H **Reserved**: Reserved, do not use $1A_H$ **CDTM2_DTM5_0_N**: Inverted dead-time output of ATOM2_4 $1B_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=4)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT94 (P13.3) 01_H **CDTM0_DTM5_3**: Dead-time output of ATOM0_7 02_H **CDTM1_DTM5_3_N**: Inverted dead-time output of ATOM1_7 03_H **CDTM2_DTM5_3**: Dead-time output of ATOM2_7 04_H **CDTM0_DTM3_3**: Dead-time output of TOM0_15 05_H **CDTM1_DTM3_3_N**: Inverted dead-time output of TOM1_15 06_H **CDTM2_DTM3_3**: Dead-time output of TOM2_15 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM2_0**: Dead-time output of TOM0_8 11_H **CDTM2_DTM0_0**: Dead-time output of TOM2_0 12_H **CDTM2_DTM4_0**: Dead-time output of ATOM2_0 13_H **Reserved**: Reserved, do not use 14_H **CDTM0_DTM0_3_N**: Inverted dead-time output of TOM0_3 15_H **CDTM2_DTM1_1_N**: Inverted dead-time output of TOM2_5 16_H **CDTM0_DTM1_0**: Dead-time output of TOM0_4 17_H **CDTM1_DTM1_0**: Dead-time output of TOM1_4 18_H **Reserved**: Reserved, do not use 19_H **CDTM2_DTM5_1_N**: Inverted dead-time output of ATOM2_5 $1A_H$ **CDTM2_DTM5_2_N**: Inverted dead-time output of ATOM2_6 $1B_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=5)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT95 (P11.2, P16.3) 01_H **CDTM0_DTM5_3_N**: Inverted dead-time output of ATOM0_7 02_H **CDTM1_DTM5_3**: Dead-time output of ATOM1_7 03_H **CDTM2_DTM5_3_N**: Inverted dead-time output of ATOM2_7 04_H **CDTM0_DTM3_3_N**: Inverted dead-time output of TOM0_15 05_H **CDTM1_DTM3_3**: Dead-time output of TOM1_15 06_H **CDTM2_DTM3_3_N**: Inverted dead-time output of TOM2_15 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM2_0**: Dead-time output of TOM0_8 11_H **CDTM2_DTM0_1**: Dead-time output of TOM2_1 12_H **CDTM2_DTM4_1**: Dead-time output of ATOM2_1 13_H **Reserved**: Reserved, do not use 14_H **CDTM2_DTM1_2_N**: Inverted dead-time output of TOM2_6 15_H **CDTM0_DTM1_0_N**: Inverted dead-time output of TOM0_4 16_H **CDTM2_DTM0_0**: Dead-time output of TOM2_0 17_H **CDTM1_DTM1_0_N**: Inverted dead-time output of TOM1_4 18_H **Reserved**: Reserved, do not use 19_H **CDTM2_DTM5_2_N**: Inverted dead-time output of ATOM2_6 $1A_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	15, 31	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.26 eGTM to Ports signal TOUT output selection register n (n=16)

For a description of this register, see 7579.

EGTM_TOUTSELn (n=16) Offset address: 061100_H+n*4

eGTM to Ports signal TOUT output selection register n Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	SEL5					SEL4					SEL3				
r	rw					rw					rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	SEL2					SEL1					SEL0				
r	rw					rw					rw				

Field	Bits	Type	Description
SELx (x=0)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT96 (P11.3, P16.2) 01_H **CDTM0_DTM4_0**: Dead-time output of ATOM0_0 02_H **CDTM1_DTM4_0_N**: Inverted dead-time output of ATOM1_0 03_H **CDTM2_DTM4_0**: Dead-time output of ATOM2_0 04_H **CDTM0_DTM0_0**: Dead-time output of TOM0_0 05_H **CDTM1_DTM0_0_N**: Inverted dead-time output of TOM1_0 06_H **CDTM2_DTM0_0**: Dead-time output of TOM2_0 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM2_2**: Dead-time output of TOM0_10 11_H **CDTM2_DTM0_2**: Dead-time output of TOM2_2 12_H **CDTM2_DTM4_2**: Dead-time output of ATOM2_2 13_H **Reserved**: Reserved, do not use 14_H **CDTM2_DTM1_3_N**: Inverted dead-time output of TOM2_7 15_H **CDTM0_DTM1_1**: Dead-time output of TOM0_5 16_H **CDTM0_DTM1_0_N**: Inverted dead-time output of TOM0_4 17_H **CDTM1_DTM1_1**: Dead-time output of TOM1_5 18_H **Reserved**: Reserved, do not use 19_H **CDTM2_DTM5_3_N**: Inverted dead-time output of ATOM2_7 $1A_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=1)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT97 (P11.6, P16.1) 01_H **CDTM0_DTM4_0_N**: Inverted dead-time output of ATOM0_0 02_H **CDTM1_DTM4_0**: Dead-time output of ATOM1_0 03_H **CDTM2_DTM4_0_N**: Inverted dead-time output of ATOM2_0 04_H **CDTM0_DTM0_0_N**: Inverted dead-time output of TOM0_0 05_H **CDTM1_DTM0_0**: Dead-time output of TOM1_0 06_H **CDTM2_DTM0_0_N**: Inverted dead-time output of TOM2_0 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM2_3**: Dead-time output of TOM0_11 11_H **CDTM2_DTM0_3**: Dead-time output of TOM2_3 12_H **CDTM2_DTM4_3**: Dead-time output of ATOM2_3 13_H **Reserved**: Reserved, do not use 14_H **CDTM2_DTM1_0_N**: Inverted dead-time output of TOM2_4 15_H **CDTM0_DTM1_1_N**: Inverted dead-time output of TOM0_5 16_H **CDTM0_DTM1_2_N**: Inverted dead-time output of TOM0_6 17_H **CDTM1_DTM1_1_N**: Inverted dead-time output of TOM1_5 18_H **Reserved**: Reserved, do not use 19_H **CDTM2_DTM5_0_N**: Inverted dead-time output of ATOM2_4 $1A_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=2)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT98 (P11.9, P16.7) 01_H **CDTM0_DTM4_1**: Dead-time output of ATOM0_1 02_H **CDTM1_DTM4_1_N**: Inverted dead-time output of ATOM1_1 03_H **CDTM2_DTM4_1**: Dead-time output of ATOM2_1 04_H **CDTM0_DTM0_1**: Dead-time output of TOM0_1 05_H **CDTM1_DTM0_1_N**: Inverted dead-time output of TOM1_1 06_H **CDTM2_DTM0_1**: Dead-time output of TOM2_1 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM3_0**: Dead-time output of TOM0_12 11_H **CDTM2_DTM1_0**: Dead-time output of TOM2_4 12_H **CDTM2_DTM5_0**: Dead-time output of ATOM2_4 13_H **Reserved**: Reserved, do not use 14_H **CDTM2_DTM0_1_N**: Inverted dead-time output of TOM2_1 15_H **CDTM0_DTM1_2**: Dead-time output of TOM0_6 16_H **CDTM2_DTM1_3_N**: Inverted dead-time output of TOM2_7 17_H **CDTM1_DTM1_2**: Dead-time output of TOM1_6 18_H **Reserved**: Reserved, do not use 19_H **CDTM2_DTM4_1_N**: Inverted dead-time output of ATOM2_1 $1A_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=3)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT99 (P11.10, P16.6) 01_H **CDTM0_DTM4_1_N**: Inverted dead-time output of ATOM0_1 02_H **CDTM1_DTM4_1**: Dead-time output of ATOM1_1 03_H **CDTM2_DTM4_1_N**: Inverted dead-time output of ATOM2_1 04_H **CDTM0_DTM0_1_N**: Inverted dead-time output of TOM0_1 05_H **CDTM1_DTM0_1**: Dead-time output of TOM1_1 06_H **CDTM2_DTM0_1_N**: Inverted dead-time output of TOM2_1 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM3_1**: Dead-time output of TOM0_13 11_H **CDTM2_DTM1_1**: Dead-time output of TOM2_5 12_H **CDTM2_DTM5_1**: Dead-time output of ATOM2_5 13_H **Reserved**: Reserved, do not use 14_H **CDTM2_DTM0_2_N**: Inverted dead-time output of TOM2_2 15_H **CDTM0_DTM1_2_N**: Inverted dead-time output of TOM0_6 16_H **CDTM2_DTM1_3_N**: Inverted dead-time output of TOM2_7 17_H **CDTM1_DTM1_2_N**: Inverted dead-time output of TOM1_6 18_H **Reserved**: Reserved, do not use 19_H **CDTM2_DTM4_2_N**: Inverted dead-time output of ATOM2_2 $1A_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=4)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT100 (P11.11, P16.13) 01_H **CDTM0_DTM4_2**: Dead-time output of ATOM0_2 02_H **CDTM1_DTM4_2_N**: Inverted dead-time output of ATOM1_2 03_H **CDTM2_DTM4_2**: Dead-time output of ATOM2_2 04_H **CDTM0_DTM0_2**: Dead-time output of TOM0_2 05_H **CDTM1_DTM0_2_N**: Inverted dead-time output of TOM1_2 06_H **CDTM2_DTM0_2**: Dead-time output of TOM2_2 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM3_2**: Dead-time output of TOM0_14 11_H **CDTM2_DTM1_2**: Dead-time output of TOM2_6 12_H **CDTM2_DTM5_2**: Dead-time output of ATOM2_6 13_H **Reserved**: Reserved, do not use 14_H **CDTM2_DTM0_3_N**: Inverted dead-time output of TOM2_3 15_H **CDTM0_DTM1_3_N**: Inverted dead-time output of TOM0_7 16_H **CDTM2_DTM1_3_N**: Inverted dead-time output of TOM2_7 17_H **CDTM1_DTM1_3_N**: Inverted dead-time output of TOM1_7 18_H **Reserved**: Reserved, do not use 19_H **CDTM2_DTM4_3_N**: Inverted dead-time output of ATOM2_3 $1A_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=5)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT101 (P11.12, P16.14) 01_H **CDTM0_DTM4_2_N**: Inverted dead-time output of ATOM0_2 02_H **CDTM1_DTM4_2**: Dead-time output of ATOM1_2 03_H **CDTM2_DTM4_2_N**: Inverted dead-time output of ATOM2_2 04_H **CDTM0_DTM0_2_N**: Inverted dead-time output of TOM0_2 05_H **CDTM1_DTM0_2**: Dead-time output of TOM1_2 06_H **CDTM2_DTM0_2_N**: Inverted dead-time output of TOM2_2 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM3_3**: Dead-time output of TOM0_15 11_H **CDTM2_DTM1_3**: Dead-time output of TOM2_7 12_H **CDTM2_DTM5_3**: Dead-time output of ATOM2_7 13_H **Reserved**: Reserved, do not use 14_H **CDTM2_DTM0_0_N**: Inverted dead-time output of TOM2_0 15_H **CDTM0_DTM1_3**: Dead-time output of TOM0_7 16_H **CDTM2_DTM1_3_N**: Inverted dead-time output of TOM2_7 17_H **CDTM1_DTM1_3**: Dead-time output of TOM1_7 18_H **Reserved**: Reserved, do not use 19_H **CDTM2_DTM4_0_N**: Inverted dead-time output of ATOM2_0 $1A_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	15, 31	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.27 eGTM to Ports signal TOUT output selection register n (n=17)

For a description of this register, see 7579.

EGTM_TOUTSELn (n=17) Offset address: 061100_H+n*4

eGTM to Ports signal TOUT output selection register n Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	SEL5					SEL4					SEL3				
r	rw					rw					rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	SEL2					SEL1					SEL0				
r	rw					rw					rw				

Field	Bits	Type	Description
SELx (x=0)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT102 (P10.0) 01_H **CDTM0_DTM4_3**: Dead-time output of ATOM0_3 02_H **CDTM1_DTM4_3_N**: Inverted dead-time output of ATOM1_3 03_H **CDTM2_DTM4_3**: Dead-time output of ATOM2_3 04_H **CDTM0_DTM0_3**: Dead-time output of TOM0_3 05_H **CDTM1_DTM0_3_N**: Inverted dead-time output of TOM1_3 06_H **CDTM2_DTM0_3**: Dead-time output of TOM2_3 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM1_0**: Dead-time output of TOM0_4 11_H **CDTM2_DTM3_0**: Dead-time output of TOM2_12 12_H **CDTM1_DTM5_0**: Dead-time output of ATOM1_4 13_H **Reserved**: Reserved, do not use 14_H **CDTM2_DTM0_0_N**: Inverted dead-time output of TOM2_0 15_H **CDTM2_DTM1_0**: Dead-time output of TOM2_4 16_H **CDTM0_DTM1_3**: Dead-time output of TOM0_7 17_H **CDTM1_DTM1_3**: Dead-time output of TOM1_7 18_H **CDTM1_DTM4_1_N**: Inverted dead-time output of ATOM1_1 19_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=1)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT103 (P10.1) 01_H **CDTM0_DTM4_3_N**: Inverted dead-time output of ATOM0_3 02_H **CDTM1_DTM4_3**: Dead-time output of ATOM1_3 03_H **CDTM2_DTM4_3_N**: Inverted dead-time output of ATOM2_3 04_H **CDTM0_DTM0_3_N**: Inverted dead-time output of TOM0_3 05_H **CDTM1_DTM0_3**: Dead-time output of TOM1_3 06_H **CDTM2_DTM0_3_N**: Inverted dead-time output of TOM2_3 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM0_1**: Dead-time output of TOM0_1 11_H **CDTM2_DTM2_1**: Dead-time output of TOM2_9 12_H **CDTM1_DTM4_1**: Dead-time output of ATOM1_1 13_H **Reserved**: Reserved, do not use 14_H **CDTM2_DTM0_0_N**: Inverted dead-time output of TOM2_0 15_H **CDTM2_DTM1_1**: Dead-time output of TOM2_5 16_H **Reserved**: Reserved, do not use 17_H **Reserved**: Reserved, do not use 18_H **CDTM1_DTM5_2_N**: Inverted dead-time output of ATOM1_6 19_H **Reserved**: Reserved, do not use $1A_H$ **CDTM2_DTM5_2**: Dead-time output of ATOM2_6 $1B_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=2)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT104 (P10.2) 01_H **CDTM0_DTM5_0**: Dead-time output of ATOM0_4 02_H **CDTM1_DTM5_0_N**: Inverted dead-time output of ATOM1_4 03_H **CDTM2_DTM5_0**: Dead-time output of ATOM2_4 04_H **CDTM0_DTM1_0**: Dead-time output of TOM0_4 05_H **CDTM1_DTM1_0_N**: Inverted dead-time output of TOM1_4 06_H **CDTM2_DTM1_0**: Dead-time output of TOM2_4 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM0_2**: Dead-time output of TOM0_2 11_H **CDTM2_DTM2_2**: Dead-time output of TOM2_10 12_H **CDTM1_DTM4_2**: Dead-time output of ATOM1_2 13_H **Reserved**: Reserved, do not use 14_H **CDTM2_DTM1_1**: Dead-time output of TOM2_5 15_H **CDTM2_DTM1_2**: Dead-time output of TOM2_6 16_H **Reserved**: Reserved, do not use 17_H **Reserved**: Reserved, do not use 18_H **CDTM1_DTM5_3_N**: Inverted dead-time output of ATOM1_7 19_H **Reserved**: Reserved, do not use $1A_H$ **CDTM2_DTM4_2**: Dead-time output of ATOM2_2 $1B_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=3)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT105 (P10.3) 01_H **CDTM0_DTM5_0_N**: Inverted dead-time output of ATOM0_4 02_H **CDTM1_DTM5_0**: Dead-time output of ATOM1_4 03_H **CDTM2_DTM5_0_N**: Inverted dead-time output of ATOM2_4 04_H **CDTM0_DTM1_0_N**: Inverted dead-time output of TOM0_4 05_H **CDTM1_DTM1_0**: Dead-time output of TOM1_4 06_H **CDTM2_DTM1_0_N**: Inverted dead-time output of TOM2_4 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM0_3**: Dead-time output of TOM0_3 11_H **CDTM2_DTM2_3**: Dead-time output of TOM2_11 12_H **CDTM1_DTM4_3**: Dead-time output of ATOM1_3 13_H **Reserved**: Reserved, do not use 14_H **CDTM2_DTM1_2**: Dead-time output of TOM2_6 15_H **CDTM2_DTM1_3**: Dead-time output of TOM2_7 16_H **Reserved**: Reserved, do not use 17_H **Reserved**: Reserved, do not use 18_H **CDTM1_DTM5_0_N**: Inverted dead-time output of ATOM1_4 19_H **Reserved**: Reserved, do not use $1A_H$ **CDTM2_DTM4_3**: Dead-time output of ATOM2_3 $1B_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=4)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT106 (P10.4) 01_H **CDTM0_DTM5_1**: Dead-time output of ATOM0_5 02_H **CDTM1_DTM5_1_N**: Inverted dead-time output of ATOM1_5 03_H **CDTM2_DTM5_1**: Dead-time output of ATOM2_5 04_H **CDTM0_DTM1_1**: Dead-time output of TOM0_5 05_H **CDTM1_DTM1_1_N**: Inverted dead-time output of TOM1_5 06_H **CDTM2_DTM1_1**: Dead-time output of TOM2_5 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM1_2**: Dead-time output of TOM0_6 11_H **CDTM2_DTM1_2**: Dead-time output of TOM2_6 12_H **CDTM0_DTM5_2**: Dead-time output of ATOM0_6 13_H **Reserved**: Reserved, do not use 14_H **CDTM0_DTM0_0_N**: Inverted dead-time output of TOM0_0 15_H **CDTM2_DTM1_0_N**: Inverted dead-time output of TOM2_4 16_H **CDTM2_DTM0_3_N**: Inverted dead-time output of TOM2_3 17_H **Reserved**: Reserved, do not use 18_H **CDTM1_DTM4_3_N**: Inverted dead-time output of ATOM1_3 19_H **Reserved**: Reserved, do not use $1A_H$ **CDTM2_DTM4_1**: Dead-time output of ATOM2_1 $1B_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=5)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT107 (P10.5) 01_H **CDTM0_DTM5_1_N**: Inverted dead-time output of ATOM0_5 02_H **CDTM1_DTM5_1**: Dead-time output of ATOM1_5 03_H **CDTM2_DTM5_1_N**: Inverted dead-time output of ATOM2_5 04_H **CDTM0_DTM1_1_N**: Inverted dead-time output of TOM0_5 05_H **CDTM1_DTM1_1**: Dead-time output of TOM1_5 06_H **CDTM2_DTM1_1_N**: Inverted dead-time output of TOM2_5 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM0_2**: Dead-time output of TOM0_2 11_H **CDTM2_DTM2_2**: Dead-time output of TOM2_10 12_H **CDTM1_DTM4_2**: Dead-time output of ATOM1_2 13_H **Reserved**: Reserved, do not use 14_H **CDTM0_DTM0_1_N**: Inverted dead-time output of TOM0_1 15_H **CDTM2_DTM1_1_N**: Inverted dead-time output of TOM2_5 16_H **Reserved**: Reserved, do not use 17_H **Reserved**: Reserved, do not use 18_H **CDTM1_DTM5_3_N**: Inverted dead-time output of ATOM1_7 19_H **Reserved**: Reserved, do not use $1A_H$ **CDTM2_DTM4_2_N**: Inverted dead-time output of ATOM2_2 $1B_H$ **CDTM1_DTM5_2**: Dead-time output of ATOM1_6 $1C_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	15, 31	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.28 eGTM to Ports signal TOUT output selection register n (n=18)

For a description of this register, see 7579.

EGTM_TOUTSELn (n=18) Offset address: 061100_H+n*4

eGTM to Ports signal TOUT output selection register n Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	SEL5					SEL4					SEL3				
r	rw					rw					rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	SEL2					SEL1					SEL0				
r	rw					rw					rw				

Field	Bits	Type	Description
SELx (x=0)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT108 (P10.6) 01_H **CDTM0_DTM5_2**: Dead-time output of ATOM0_6 02_H **CDTM1_DTM5_2_N**: Inverted dead-time output of ATOM1_6 03_H **CDTM2_DTM5_2**: Dead-time output of ATOM2_6 04_H **CDTM0_DTM1_2**: Dead-time output of TOM0_6 05_H **CDTM1_DTM1_2_N**: Inverted dead-time output of TOM1_6 06_H **CDTM2_DTM1_2**: Dead-time output of TOM2_6 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM0_3**: Dead-time output of TOM0_3 11_H **CDTM2_DTM2_3**: Dead-time output of TOM2_11 12_H **CDTM1_DTM4_3**: Dead-time output of ATOM1_3 13_H **Reserved**: Reserved, do not use 14_H **CDTM0_DTM0_2_N**: Inverted dead-time output of TOM0_2 15_H **CDTM2_DTM1_2_N**: Inverted dead-time output of TOM2_6 16_H **Reserved**: Reserved, do not use 17_H **Reserved**: Reserved, do not use 18_H **CDTM1_DTM5_0_N**: Inverted dead-time output of ATOM1_4 19_H **Reserved**: Reserved, do not use $1A_H$ **CDTM2_DTM4_3_N**: Inverted dead-time output of ATOM2_3 $1B_H$ **CDTM1_DTM4_3**: Dead-time output of ATOM1_3 $1C_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=1)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT109 (P10.7) 01_H **CDTM0_DTM5_2_N**: Inverted dead-time output of ATOM0_6 02_H **CDTM1_DTM5_2**: Dead-time output of ATOM1_6 03_H **CDTM2_DTM5_2_N**: Inverted dead-time output of ATOM2_6 04_H **CDTM0_DTM1_2_N**: Inverted dead-time output of TOM0_6 05_H **CDTM1_DTM1_2**: Dead-time output of TOM1_6 06_H **CDTM2_DTM1_2_N**: Inverted dead-time output of TOM2_6 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM0_0**: Dead-time output of TOM0_0 11_H **CDTM2_DTM2_0**: Dead-time output of TOM2_8 12_H **CDTM1_DTM4_0**: Dead-time output of ATOM1_0 13_H **Reserved**: Reserved, do not use 14_H **CDTM0_DTM0_3_N**: Inverted dead-time output of TOM0_3 15_H **CDTM2_DTM1_3_N**: Inverted dead-time output of TOM2_7 16_H **Reserved**: Reserved, do not use 17_H **Reserved**: Reserved, do not use 18_H **CDTM1_DTM5_1_N**: Inverted dead-time output of ATOM1_5 19_H **Reserved**: Reserved, do not use $1A_H$ **CDTM2_DTM4_0_N**: Inverted dead-time output of ATOM2_0 $1B_H$ **CDTM1_DTM4_0**: Dead-time output of ATOM1_0 $1C_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=2)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT110 (P10.8) 01_H **CDTM0_DTM5_3**: Dead-time output of ATOM0_7 02_H **CDTM1_DTM5_3_N**: Inverted dead-time output of ATOM1_7 03_H **CDTM2_DTM5_3**: Dead-time output of ATOM2_7 04_H **CDTM0_DTM1_3**: Dead-time output of TOM0_7 05_H **CDTM1_DTM1_3_N**: Inverted dead-time output of TOM1_7 06_H **CDTM2_DTM1_3**: Dead-time output of TOM2_7 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM1_1**: Dead-time output of TOM0_5 11_H **CDTM2_DTM3_1**: Dead-time output of TOM2_13 12_H **CDTM1_DTM5_1**: Dead-time output of ATOM1_5 13_H **Reserved**: Reserved, do not use … 17_H **Reserved**: Reserved, do not use 18_H **CDTM1_DTM4_2_N**: Inverted dead-time output of ATOM1_2 19_H **Reserved**: Reserved, do not use $1A_H$ **CDTM2_DTM4_0**: Dead-time output of ATOM2_0 $1B_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
SELx (x=3)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT111 (P01.3) 01_H **CDTM0_DTM5_3_N**: Inverted dead-time output of ATOM0_7 02_H **CDTM1_DTM5_3**: Dead-time output of ATOM1_7 03_H **CDTM2_DTM5_3_N**: Inverted dead-time output of ATOM2_7 04_H **CDTM0_DTM1_3_N**: Inverted dead-time output of TOM0_7 05_H **CDTM1_DTM1_3**: Dead-time output of TOM1_7 06_H **CDTM2_DTM1_3_N**: Inverted dead-time output of TOM2_7 07_H **Reserved**: Reserved, do not use … 10_H **Reserved**: Reserved, do not use 11_H **CDTM2_DTM3_1**: Dead-time output of TOM2_13 12_H **Reserved**: Reserved, do not use … 19_H **Reserved**: Reserved, do not use $1A_H$ **CDTM2_DTM4_0**: Dead-time output of ATOM2_0 $1B_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=4)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT112 (P01.4) 01_H **CDTM0_DTM4_0**: Dead-time output of ATOM0_0 02_H **CDTM1_DTM4_0_N**: Inverted dead-time output of ATOM1_0 03_H **CDTM2_DTM4_0**: Dead-time output of ATOM2_0 04_H **CDTM0_DTM2_0**: Dead-time output of TOM0_8 05_H **CDTM1_DTM2_0_N**: Inverted dead-time output of TOM1_8 06_H **CDTM2_DTM2_0**: Dead-time output of TOM2_8 07_H **Reserved**: Reserved, do not use … 17_H **Reserved**: Reserved, do not use 18_H **CDTM0_DTM4_0_N**: Inverted dead-time output of ATOM0_0 19_H **CDTM1_DTM4_0_N**: Inverted dead-time output of ATOM1_0 $1A_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
SELx (x=5)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT113 (P01.5) 01_H **CDTM0_DTM4_0_N**: Inverted dead-time output of ATOM0_0 02_H **CDTM1_DTM4_0**: Dead-time output of ATOM1_0 03_H **CDTM2_DTM4_0_N**: Inverted dead-time output of ATOM2_0 04_H **CDTM0_DTM2_0_N**: Inverted dead-time output of TOM0_8 05_H **CDTM1_DTM2_0**: Dead-time output of TOM1_8 06_H **CDTM2_DTM2_0_N**: Inverted dead-time output of TOM2_8 07_H **Reserved**: Reserved, do not use … 17_H **Reserved**: Reserved, do not use 18_H **CDTM0_DTM4_1_N**: Inverted dead-time output of ATOM0_1 19_H **CDTM1_DTM4_1_N**: Inverted dead-time output of ATOM1_1 $1A_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	15, 31	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.29 eGTM to Ports signal TOUT output selection register n (n=19)

For a description of this register, see 7579.

EGTM_TOUTSELn (n=19) Offset address: 061100_H+n*4

eGTM to Ports signal TOUT output selection register n Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	SEL5					SEL4					SEL3				
r	rw					rw					rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	SEL2					SEL1					SEL0				
r	rw					rw					rw				

Field	Bits	Type	Description
SELx (x=0)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT114 (P01.6) 01_H **CDTM0_DTM4_1**: Dead-time output of ATOM0_1 02_H **CDTM1_DTM4_1_N**: Inverted dead-time output of ATOM1_1 03_H **CDTM2_DTM4_1**: Dead-time output of ATOM2_1 04_H **CDTM0_DTM2_1**: Dead-time output of TOM0_9 05_H **CDTM1_DTM2_1_N**: Inverted dead-time output of TOM1_9 06_H **CDTM2_DTM2_1**: Dead-time output of TOM2_9 07_H **Reserved**: Reserved, do not use … 17_H **Reserved**: Reserved, do not use 18_H **CDTM0_DTM4_2_N**: Inverted dead-time output of ATOM0_2 19_H **CDTM1_DTM4_2_N**: Inverted dead-time output of ATOM1_2 $1A_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
SELx (x=1)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT115 (P01.7) 01_H **CDTM0_DTM4_1_N**: Inverted dead-time output of ATOM0_1 02_H **CDTM1_DTM4_1**: Dead-time output of ATOM1_1 03_H **CDTM2_DTM4_1_N**: Inverted dead-time output of ATOM2_1 04_H **CDTM0_DTM2_1_N**: Inverted dead-time output of TOM0_9 05_H **CDTM1_DTM2_1**: Dead-time output of TOM1_9 06_H **CDTM2_DTM2_1_N**: Inverted dead-time output of TOM2_9 07_H **Reserved**: Reserved, do not use … 17_H **Reserved**: Reserved, do not use 18_H **CDTM0_DTM4_3_N**: Inverted dead-time output of ATOM0_3 19_H **CDTM1_DTM4_3_N**: Inverted dead-time output of ATOM1_3 $1A_H$ **CDTM2_DTM4_3_N**: Inverted dead-time output of ATOM2_3 $1B_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=2)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT116 (P02.9) 01_H **CDTM0_DTM4_2**: Dead-time output of ATOM0_2 02_H **CDTM1_DTM4_2_N**: Inverted dead-time output of ATOM1_2 03_H **CDTM2_DTM4_2**: Dead-time output of ATOM2_2 04_H **CDTM0_DTM2_2**: Dead-time output of TOM0_10 05_H **CDTM1_DTM2_2_N**: Inverted dead-time output of TOM1_10 06_H **CDTM2_DTM2_2**: Dead-time output of TOM2_10 07_H **Reserved**: Reserved, do not use … 13_H **Reserved**: Reserved, do not use 14_H **CDTM1_DTM0_1**: Dead-time output of TOM1_1 15_H **Reserved**: Reserved, do not use 16_H **CDTM0_DTM0_1**: Dead-time output of TOM0_1 17_H **CDTM1_DTM0_1**: Dead-time output of TOM1_1 18_H **CDTM0_DTM5_2_N**: Inverted dead-time output of ATOM0_6 19_H **CDTM1_DTM5_2_N**: Inverted dead-time output of ATOM1_6 $1A_H$ **CDTM2_DTM5_2_N**: Inverted dead-time output of ATOM2_6 $1B_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
SELx (x=3)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT117 (P02.10) 01_H **CDTM0_DTM4_2_N**: Inverted dead-time output of ATOM0_2 02_H **CDTM1_DTM4_2**: Dead-time output of ATOM1_2 03_H **CDTM2_DTM4_2_N**: Inverted dead-time output of ATOM2_2 04_H **CDTM0_DTM2_2_N**: Inverted dead-time output of TOM0_10 05_H **CDTM1_DTM2_2**: Dead-time output of TOM1_10 06_H **CDTM2_DTM2_2_N**: Inverted dead-time output of TOM2_10 07_H **Reserved**: Reserved, do not use … 13_H **Reserved**: Reserved, do not use 14_H **CDTM1_DTM0_2**: Dead-time output of TOM1_2 15_H **Reserved**: Reserved, do not use 16_H **CDTM0_DTM0_2**: Dead-time output of TOM0_2 17_H **CDTM1_DTM0_2**: Dead-time output of TOM1_2 18_H **CDTM0_DTM5_3_N**: Inverted dead-time output of ATOM0_7 19_H **CDTM1_DTM5_3_N**: Inverted dead-time output of ATOM1_7 $1A_H$ **CDTM0_DTM4_1**: Dead-time output of ATOM0_1 $1B_H$ **CDTM1_DTM4_1**: Dead-time output of ATOM1_1 $1C_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=4)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT118 (P02.11) 01_H **CDTM0_DTM4_3**: Dead-time output of ATOM0_3 02_H **CDTM1_DTM4_3_N**: Inverted dead-time output of ATOM1_3 03_H **CDTM2_DTM4_3**: Dead-time output of ATOM2_3 04_H **CDTM0_DTM2_3**: Dead-time output of TOM0_11 05_H **CDTM1_DTM2_3_N**: Inverted dead-time output of TOM1_11 06_H **CDTM2_DTM2_3**: Dead-time output of TOM2_11 07_H **Reserved**: Reserved, do not use … 13_H **Reserved**: Reserved, do not use 14_H **CDTM1_DTM0_3**: Dead-time output of TOM1_3 15_H **Reserved**: Reserved, do not use 16_H **CDTM0_DTM0_3**: Dead-time output of TOM0_3 17_H **CDTM1_DTM0_3**: Dead-time output of TOM1_3 18_H **CDTM0_DTM5_0_N**: Inverted dead-time output of ATOM0_4 19_H **CDTM1_DTM5_0_N**: Inverted dead-time output of ATOM1_4 $1A_H$ **CDTM0_DTM4_2**: Dead-time output of ATOM0_2 $1B_H$ **CDTM1_DTM4_2**: Dead-time output of ATOM1_2 $1C_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=5)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT119 (P11.0, P16.5) 01_H **CDTM0_DTM4_3_N**: Inverted dead-time output of ATOM0_3 02_H **CDTM1_DTM4_3**: Dead-time output of ATOM1_3 03_H **CDTM2_DTM4_3_N**: Inverted dead-time output of ATOM2_3 04_H **CDTM0_DTM2_3_N**: Inverted dead-time output of TOM0_11 05_H **CDTM1_DTM2_3**: Dead-time output of TOM1_11 06_H **CDTM2_DTM2_3_N**: Inverted dead-time output of TOM2_11 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM2_DTM0_0**: Dead-time output of TOM2_0 11_H **CDTM2_DTM0_0**: Dead-time output of TOM2_0 12_H **Reserved**: Reserved, do not use 13_H **Reserved**: Reserved, do not use 14_H **CDTM2_DTM1_1_N**: Inverted dead-time output of TOM2_5 15_H **CDTM2_DTM1_1_N**: Inverted dead-time output of TOM2_5 16_H **CDTM0_DTM1_3**: Dead-time output of TOM0_7 17_H **CDTM1_DTM1_3**: Dead-time output of TOM1_7 18_H **Reserved**: Reserved, do not use 19_H **Reserved**: Reserved, do not use $1A_H$ **CDTM0_DTM4_3**: Dead-time output of ATOM0_3 $1B_H$ **CDTM1_DTM4_3**: Dead-time output of ATOM1_3 $1C_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	15, 31	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.30 eGTM to Ports signal TOUT output selection register n (n=20)

For a description of this register, see 7579.

EGTM_TOUTSELn (n=20) Offset address: 061100_H+n*4

eGTM to Ports signal TOUT output selection register n Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	SEL5					SEL4					SEL3				
r	rw					rw					rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	SEL2					SEL1					SEL0				
r	rw					rw					rw				

Field	Bits	Type	Description
SELx (x=0)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT120 (P11.1, P16.4) 01_H **CDTM0_DTM5_0**: Dead-time output of ATOM0_4 02_H **CDTM1_DTM5_0_N**: Inverted dead-time output of ATOM1_4 03_H **CDTM2_DTM5_0**: Dead-time output of ATOM2_4 04_H **CDTM0_DTM3_0**: Dead-time output of TOM0_12 05_H **CDTM1_DTM3_0_N**: Inverted dead-time output of TOM1_12 06_H **CDTM2_DTM3_0**: Dead-time output of TOM2_12 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM2_DTM0_1**: Dead-time output of TOM2_1 11_H **CDTM2_DTM0_1_N**: Inverted dead-time output of TOM2_1 12_H **Reserved**: Reserved, do not use 13_H **Reserved**: Reserved, do not use 14_H **CDTM2_DTM1_2_N**: Inverted dead-time output of TOM2_6 15_H **CDTM2_DTM1_0_N**: Inverted dead-time output of TOM2_4 16_H **CDTM2_DTM0_0**: Dead-time output of TOM2_0 17_H **Reserved**: Reserved, do not use 18_H **Reserved**: Reserved, do not use 19_H **CDTM2_DTM5_1_N**: Inverted dead-time output of ATOM2_5 $1A_H$ **CDTM0_DTM5_1**: Dead-time output of ATOM0_5 $1B_H$ **CDTM1_DTM5_1**: Dead-time output of ATOM1_5 $1C_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=1)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT121 (P11.4, P16.0) 01_H **CDTM0_DTM5_0_N**: Inverted dead-time output of ATOM0_4 02_H **CDTM1_DTM5_0**: Dead-time output of ATOM1_4 03_H **CDTM2_DTM5_0_N**: Inverted dead-time output of ATOM2_4 04_H **CDTM0_DTM3_0_N**: Inverted dead-time output of TOM0_12 05_H **CDTM1_DTM3_0**: Dead-time output of TOM1_12 06_H **CDTM2_DTM3_0_N**: Inverted dead-time output of TOM2_12 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM2_DTM0_2**: Dead-time output of TOM2_2 11_H **CDTM2_DTM0_2_N**: Inverted dead-time output of TOM2_2 12_H **Reserved**: Reserved, do not use 13_H **Reserved**: Reserved, do not use 14_H **CDTM2_DTM1_3_N**: Inverted dead-time output of TOM2_7 15_H **CDTM2_DTM1_0_N**: Inverted dead-time output of TOM2_4 16_H **CDTM0_DTM1_1_N**: Inverted dead-time output of TOM0_5 17_H **CDTM1_DTM1_1**: Dead-time output of TOM1_5 18_H **Reserved**: Reserved, do not use 19_H **CDTM2_DTM5_1**: Dead-time output of ATOM2_5 $1A_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=2)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT122 (P11.5, P16.12) 01_H **CDTM0_DTM5_1**: Dead-time output of ATOM0_5 02_H **CDTM1_DTM5_1_N**: Inverted dead-time output of ATOM1_5 03_H **CDTM2_DTM5_1**: Dead-time output of ATOM2_5 04_H **CDTM0_DTM3_1**: Dead-time output of TOM0_13 05_H **CDTM1_DTM3_1_N**: Inverted dead-time output of TOM1_13 06_H **CDTM2_DTM3_1**: Dead-time output of TOM2_13 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM2_DTM0_3**: Dead-time output of TOM2_3 11_H **CDTM2_DTM0_3_N**: Inverted dead-time output of TOM2_3 12_H **Reserved**: Reserved, do not use 13_H **Reserved**: Reserved, do not use 14_H **CDTM2_DTM1_0_N**: Inverted dead-time output of TOM2_4 15_H **CDTM2_DTM0_3_N**: Inverted dead-time output of TOM2_3 16_H **CDTM2_DTM1_2_N**: Inverted dead-time output of TOM2_6 17_H **Reserved**: Reserved, do not use 18_H **Reserved**: Reserved, do not use 19_H **CDTM2_DTM5_1_N**: Inverted dead-time output of ATOM2_5 $1A_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=3)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT123 (P11.7, P16.8) 01_H **CDTM0_DTM5_1_N**: Inverted dead-time output of ATOM0_5 02_H **CDTM1_DTM5_1**: Dead-time output of ATOM1_5 03_H **CDTM2_DTM5_1_N**: Inverted dead-time output of ATOM2_5 04_H **CDTM0_DTM3_1_N**: Inverted dead-time output of TOM0_13 05_H **CDTM1_DTM3_1**: Dead-time output of TOM1_13 06_H **CDTM2_DTM3_1_N**: Inverted dead-time output of TOM2_13 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM2_DTM1_0**: Dead-time output of TOM2_4 11_H **CDTM2_DTM1_0_N**: Inverted dead-time output of TOM2_4 12_H **Reserved**: Reserved, do not use 13_H **Reserved**: Reserved, do not use 14_H **CDTM2_DTM0_1_N**: Inverted dead-time output of TOM2_1 15_H **CDTM2_DTM0_0_N**: Inverted dead-time output of TOM2_0 16_H **CDTM0_DTM1_3_N**: Inverted dead-time output of TOM0_7 17_H **CDTM1_DTM1_3_N**: Inverted dead-time output of TOM1_7 18_H **Reserved**: Reserved, do not use 19_H **CDTM2_DTM5_1**: Dead-time output of ATOM2_5 $1A_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=4)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT124 (P11.8, P16.9) 01_H **CDTM0_DTM5_2**: Dead-time output of ATOM0_6 02_H **CDTM1_DTM5_2_N**: Inverted dead-time output of ATOM1_6 03_H **CDTM2_DTM5_2**: Dead-time output of ATOM2_6 04_H **CDTM0_DTM3_2**: Dead-time output of TOM0_14 05_H **CDTM1_DTM3_2_N**: Inverted dead-time output of TOM1_14 06_H **CDTM2_DTM3_2**: Dead-time output of TOM2_14 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM2_DTM1_1**: Dead-time output of TOM2_5 11_H **CDTM2_DTM1_1_N**: Inverted dead-time output of TOM2_5 12_H **Reserved**: Reserved, do not use 13_H **Reserved**: Reserved, do not use 14_H **CDTM2_DTM0_2_N**: Inverted dead-time output of TOM2_2 15_H **CDTM0_DTM1_0**: Dead-time output of TOM0_4 16_H **CDTM2_DTM0_1_N**: Inverted dead-time output of TOM2_1 17_H **CDTM1_DTM1_0**: Dead-time output of TOM1_4 18_H **Reserved**: Reserved, do not use 19_H **CDTM2_DTM5_1**: Dead-time output of ATOM2_5 $1A_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=5)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT125 (P11.13) 01_H **CDTM0_DTM5_2_N**: Inverted dead-time output of ATOM0_6 02_H **CDTM1_DTM5_2**: Dead-time output of ATOM1_6 03_H **CDTM2_DTM5_2_N**: Inverted dead-time output of ATOM2_6 04_H **CDTM0_DTM3_2_N**: Inverted dead-time output of TOM0_14 05_H **CDTM1_DTM3_2**: Dead-time output of TOM1_14 06_H **CDTM2_DTM3_2_N**: Inverted dead-time output of TOM2_14 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM2_DTM1_2**: Dead-time output of TOM2_6 11_H **CDTM2_DTM1_2_N**: Inverted dead-time output of TOM2_6 12_H **Reserved**: Reserved, do not use 13_H **Reserved**: Reserved, do not use 14_H **CDTM2_DTM0_2**: Dead-time output of TOM2_2 15_H **CDTM2_DTM0_1_N**: Inverted dead-time output of TOM2_1 16_H **CDTM0_DTM1_3_N**: Inverted dead-time output of TOM0_7 17_H **CDTM1_DTM1_3_N**: Inverted dead-time output of TOM1_7 18_H **Reserved**: Reserved, do not use 19_H **CDTM2_DTM4_3_N**: Inverted dead-time output of ATOM2_3 $1A_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	15, 31	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.31 eGTM to Ports signal TOUT output selection register n (n=21)

For a description of this register, see 7579.

EGTM_TOUTSELn (n=21) Offset address: 061100_H+n*4

eGTM to Ports signal TOUT output selection register n Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	SEL5					SEL4					SEL3				
r	rw					rw					rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	SEL2					SEL1					SEL0				
r	rw					rw					rw				

Field	Bits	Type	Description
SELx (x=0)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT126 (P11.14) 01_H **CDTM0_DTM5_3**: Dead-time output of ATOM0_7 02_H **CDTM1_DTM5_3_N**: Inverted dead-time output of ATOM1_7 03_H **CDTM2_DTM5_3**: Dead-time output of ATOM2_7 04_H **CDTM0_DTM3_3**: Dead-time output of TOM0_15 05_H **CDTM1_DTM3_3_N**: Inverted dead-time output of TOM1_15 06_H **CDTM2_DTM3_3**: Dead-time output of TOM2_15 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM2_DTM1_3**: Dead-time output of TOM2_7 11_H **CDTM2_DTM1_3_N**: Inverted dead-time output of TOM2_7 12_H **Reserved**: Reserved, do not use 13_H **Reserved**: Reserved, do not use 14_H **CDTM2_DTM0_3**: Dead-time output of TOM2_3 15_H **CDTM2_DTM1_0_N**: Inverted dead-time output of TOM2_4 16_H **CDTM2_DTM0_0_N**: Inverted dead-time output of TOM2_0 17_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
SELx (x=1)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT127 (P11.15) 01_H **CDTM0_DTM5_3_N**: Inverted dead-time output of ATOM0_7 02_H **CDTM1_DTM5_3**: Dead-time output of ATOM1_7 03_H **CDTM2_DTM5_3_N**: Inverted dead-time output of ATOM2_7 04_H **CDTM0_DTM3_3_N**: Inverted dead-time output of TOM0_15 05_H **CDTM1_DTM3_3**: Dead-time output of TOM1_15 06_H **CDTM2_DTM3_3_N**: Inverted dead-time output of TOM2_15 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM2_DTM2_0**: Dead-time output of TOM2_8 11_H **CDTM2_DTM2_0**: Dead-time output of TOM2_8 12_H **Reserved**: Reserved, do not use 13_H **Reserved**: Reserved, do not use 14_H **CDTM2_DTM0_1**: Dead-time output of TOM2_1 15_H **CDTM2_DTM0_2_N**: Inverted dead-time output of TOM2_2 16_H **CDTM2_DTM0_0_N**: Inverted dead-time output of TOM2_0 17_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=2)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT128 (P12.0, P16.11) 01_H **CDTM0_DTM4_0**: Dead-time output of ATOM0_0 02_H **CDTM1_DTM4_0_N**: Inverted dead-time output of ATOM1_0 03_H **CDTM2_DTM4_0**: Dead-time output of ATOM2_0 04_H **CDTM0_DTM0_0**: Dead-time output of TOM0_0 05_H **CDTM1_DTM0_0_N**: Inverted dead-time output of TOM1_0 06_H **CDTM2_DTM0_0**: Dead-time output of TOM2_0 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM1_DTM2_0**: Dead-time output of TOM1_8 11_H **CDTM1_DTM0_0_N**: Inverted dead-time output of TOM1_0 12_H **Reserved**: Reserved, do not use 13_H **Reserved**: Reserved, do not use 14_H **CDTM2_DTM0_1**: Dead-time output of TOM2_1 15_H **CDTM2_DTM0_2_N**: Inverted dead-time output of TOM2_2 16_H **CDTM0_DTM1_1**: Dead-time output of TOM0_5 17_H **CDTM1_DTM1_1**: Dead-time output of TOM1_5 18_H **Reserved**: Reserved, do not use … $1A_H$ **Reserved**: Reserved, do not use $1B_H$ **CDTM2_DTM5_1**: Dead-time output of ATOM2_5 $1C_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=3)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT129 (P12.1, P16.10) 01_H **CDTM0_DTM4_0_N**: Inverted dead-time output of ATOM0_0 02_H **CDTM1_DTM4_0**: Dead-time output of ATOM1_0 03_H **CDTM2_DTM4_0_N**: Inverted dead-time output of ATOM2_0 04_H **CDTM0_DTM0_0_N**: Inverted dead-time output of TOM0_0 05_H **CDTM1_DTM0_0**: Dead-time output of TOM1_0 06_H **CDTM2_DTM0_0_N**: Inverted dead-time output of TOM2_0 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM1_DTM2_1**: Dead-time output of TOM1_9 11_H **CDTM1_DTM0_1_N**: Inverted dead-time output of TOM1_1 12_H **Reserved**: Reserved, do not use 13_H **Reserved**: Reserved, do not use 14_H **CDTM2_DTM0_1**: Dead-time output of TOM2_1 15_H **CDTM2_DTM0_2_N**: Inverted dead-time output of TOM2_2 16_H **CDTM0_DTM1_2**: Dead-time output of TOM0_6 17_H **CDTM1_DTM1_2**: Dead-time output of TOM1_6 18_H **Reserved**: Reserved, do not use … $1A_H$ **Reserved**: Reserved, do not use $1B_H$ **CDTM2_DTM4_1**: Dead-time output of ATOM2_1 $1C_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=4)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT130 (P22.4) 01_H **CDTM0_DTM4_1**: Dead-time output of ATOM0_1 02_H **CDTM1_DTM4_1_N**: Inverted dead-time output of ATOM1_1 03_H **CDTM2_DTM4_1**: Dead-time output of ATOM2_1 04_H **CDTM0_DTM0_1**: Dead-time output of TOM0_1 05_H **CDTM1_DTM0_1_N**: Inverted dead-time output of TOM1_1 06_H **CDTM2_DTM0_1**: Dead-time output of TOM2_1 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM2_DTM1_3**: Dead-time output of TOM2_7 11_H **CDTM2_DTM0_2_N**: Inverted dead-time output of TOM2_2 12_H **Reserved**: Reserved, do not use 13_H **Reserved**: Reserved, do not use 14_H **CDTM2_DTM0_3**: Dead-time output of TOM2_3 15_H **Reserved**: Reserved, do not use 16_H **CDTM0_DTM1_3_N**: Inverted dead-time output of TOM0_7 17_H **CDTM1_DTM1_3_N**: Inverted dead-time output of TOM1_7 18_H **CDTM1_DTM5_0**: Dead-time output of ATOM1_4 19_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=5)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT131 (P22.5) 01_H **CDTM0_DTM4_1_N**: Inverted dead-time output of ATOM0_1 02_H **CDTM1_DTM4_1**: Dead-time output of ATOM1_1 03_H **CDTM2_DTM4_1_N**: Inverted dead-time output of ATOM2_1 04_H **CDTM0_DTM0_1_N**: Inverted dead-time output of TOM0_1 05_H **CDTM1_DTM0_1**: Dead-time output of TOM1_1 06_H **CDTM2_DTM0_1_N**: Inverted dead-time output of TOM2_1 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM2_DTM2_0**: Dead-time output of TOM2_8 11_H **CDTM2_DTM0_3_N**: Inverted dead-time output of TOM2_3 12_H **Reserved**: Reserved, do not use 13_H **Reserved**: Reserved, do not use 14_H **CDTM2_DTM1_2**: Dead-time output of TOM2_6 15_H **Reserved**: Reserved, do not use … 17_H **Reserved**: Reserved, do not use 18_H **CDTM1_DTM5_0_N**: Inverted dead-time output of ATOM1_4 19_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	15, 31	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.32 eGTM to Ports signal TOUT output selection register n (n=22)

For a description of this register, see 7579.

EGTM_TOUTSELn (n=22) Offset address: 061100_H+n*4

eGTM to Ports signal TOUT output selection register n Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	SEL5					SEL4					SEL3				
r	rw					rw					rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	SEL2					SEL1					SEL0				
r	rw					rw					rw				

Field	Bits	Type	Description
SELx (x=0)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT132 (P22.6) 01_H **CDTM0_DTM4_2**: Dead-time output of ATOM0_2 02_H **CDTM1_DTM4_2_N**: Inverted dead-time output of ATOM1_2 03_H **CDTM2_DTM4_2**: Dead-time output of ATOM2_2 04_H **CDTM0_DTM0_2**: Dead-time output of TOM0_2 05_H **CDTM1_DTM0_2_N**: Inverted dead-time output of TOM1_2 06_H **CDTM2_DTM0_2**: Dead-time output of TOM2_2 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM2_DTM2_1**: Dead-time output of TOM2_9 11_H **CDTM2_DTM1_0_N**: Inverted dead-time output of TOM2_4 12_H **Reserved**: Reserved, do not use 13_H **Reserved**: Reserved, do not use 14_H **CDTM2_DTM1_1**: Dead-time output of TOM2_5 15_H **Reserved**: Reserved, do not use … 17_H **Reserved**: Reserved, do not use 18_H **CDTM1_DTM5_1_N**: Inverted dead-time output of ATOM1_5 19_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=1)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT133 (P22.7) 01_H **CDTM0_DTM4_2_N**: Inverted dead-time output of ATOM0_2 02_H **CDTM1_DTM4_2**: Dead-time output of ATOM1_2 03_H **CDTM2_DTM4_2_N**: Inverted dead-time output of ATOM2_2 04_H **CDTM0_DTM0_2_N**: Inverted dead-time output of TOM0_2 05_H **CDTM1_DTM0_2**: Dead-time output of TOM1_2 06_H **CDTM2_DTM0_2_N**: Inverted dead-time output of TOM2_2 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM2_DTM2_2**: Dead-time output of TOM2_10 11_H **CDTM2_DTM1_1_N**: Inverted dead-time output of TOM2_5 12_H **Reserved**: Reserved, do not use 13_H **Reserved**: Reserved, do not use 14_H **CDTM2_DTM1_0**: Dead-time output of TOM2_4 15_H **Reserved**: Reserved, do not use … 17_H **Reserved**: Reserved, do not use 18_H **CDTM1_DTM5_2_N**: Inverted dead-time output of ATOM1_6 19_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=2)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT134 (P22.8) 01_H **CDTM0_DTM4_3**: Dead-time output of ATOM0_3 02_H **CDTM1_DTM4_3_N**: Inverted dead-time output of ATOM1_3 03_H **CDTM2_DTM4_3**: Dead-time output of ATOM2_3 04_H **CDTM0_DTM0_3**: Dead-time output of TOM0_3 05_H **CDTM1_DTM0_3_N**: Inverted dead-time output of TOM1_3 06_H **CDTM2_DTM0_3**: Dead-time output of TOM2_3 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM2_DTM2_3**: Dead-time output of TOM2_11 11_H **CDTM2_DTM1_2_N**: Inverted dead-time output of TOM2_6 12_H **Reserved**: Reserved, do not use 13_H **Reserved**: Reserved, do not use 14_H **CDTM2_DTM1_3_N**: Inverted dead-time output of TOM2_7 15_H **Reserved**: Reserved, do not use … 17_H **Reserved**: Reserved, do not use 18_H **CDTM1_DTM5_3_N**: Inverted dead-time output of ATOM1_7 19_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=3)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT135 (P22.9) 01_H **CDTM0_DTM4_3_N**: Inverted dead-time output of ATOM0_3 02_H **CDTM1_DTM4_3**: Dead-time output of ATOM1_3 03_H **CDTM2_DTM4_3_N**: Inverted dead-time output of ATOM2_3 04_H **CDTM0_DTM0_3_N**: Inverted dead-time output of TOM0_3 05_H **CDTM1_DTM0_3**: Dead-time output of TOM1_3 06_H **CDTM2_DTM0_3_N**: Inverted dead-time output of TOM2_3 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM2_DTM3_0**: Dead-time output of TOM2_12 11_H **CDTM2_DTM1_3_N**: Inverted dead-time output of TOM2_7 12_H **Reserved**: Reserved, do not use 13_H **Reserved**: Reserved, do not use 14_H **CDTM2_DTM1_2_N**: Inverted dead-time output of TOM2_6 15_H **Reserved**: Reserved, do not use … 17_H **Reserved**: Reserved, do not use 18_H **CDTM0_DTM5_0_N**: Inverted dead-time output of ATOM0_4 19_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=4)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT136 (P22.10) 01_H **CDTM0_DTM5_0**: Dead-time output of ATOM0_4 02_H **CDTM1_DTM5_0_N**: Inverted dead-time output of ATOM1_4 03_H **CDTM2_DTM5_0**: Dead-time output of ATOM2_4 04_H **CDTM0_DTM1_0**: Dead-time output of TOM0_4 05_H **CDTM1_DTM1_0_N**: Inverted dead-time output of TOM1_4 06_H **CDTM2_DTM1_0**: Dead-time output of TOM2_4 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM2_DTM3_1**: Dead-time output of TOM2_13 11_H **Reserved**: Reserved, do not use … 13_H **Reserved**: Reserved, do not use 14_H **CDTM2_DTM1_1_N**: Inverted dead-time output of TOM2_5 15_H **Reserved**: Reserved, do not use … 17_H **Reserved**: Reserved, do not use 18_H **CDTM0_DTM5_1_N**: Inverted dead-time output of ATOM0_5 19_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=5)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT137 (P22.11) 01_H **CDTM0_DTM5_0_N**: Inverted dead-time output of ATOM0_4 02_H **CDTM1_DTM5_0**: Dead-time output of ATOM1_4 03_H **CDTM2_DTM5_0_N**: Inverted dead-time output of ATOM2_4 04_H **CDTM0_DTM1_0_N**: Inverted dead-time output of TOM0_4 05_H **CDTM1_DTM1_0**: Dead-time output of TOM1_4 06_H **CDTM2_DTM1_0_N**: Inverted dead-time output of TOM2_4 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM2_DTM3_2**: Dead-time output of TOM2_14 11_H **Reserved**: Reserved, do not use … 13_H **Reserved**: Reserved, do not use 14_H **CDTM2_DTM1_0_N**: Inverted dead-time output of TOM2_4 15_H **Reserved**: Reserved, do not use … 17_H **Reserved**: Reserved, do not use 18_H **CDTM0_DTM5_2_N**: Inverted dead-time output of ATOM0_6 19_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	15, 31	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.33 eGTM to Ports signal TOUT output selection register n (n=23)

For a description of this register, see 7579.

EGTM_TOUTSELn (n=23) Offset address: 061100_H+n*4

eGTM to Ports signal TOUT output selection register n Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	SEL5					SEL4					SEL3				
r	rw					rw					rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	SEL2					SEL1					SEL0				
r	rw					rw					rw				

Field	Bits	Type	Description
SELx (x=0)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT138 (P23.6) 01_H **CDTM0_DTM5_1**: Dead-time output of ATOM0_5 02_H **CDTM1_DTM5_1_N**: Inverted dead-time output of ATOM1_5 03_H **CDTM2_DTM5_1**: Dead-time output of ATOM2_5 04_H **CDTM0_DTM1_1**: Dead-time output of TOM0_5 05_H **CDTM1_DTM1_1_N**: Inverted dead-time output of TOM1_5 06_H **CDTM2_DTM1_1**: Dead-time output of TOM2_5 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM2_DTM5_1**: Dead-time output of ATOM2_5 11_H **Reserved**: Reserved, do not use … 17_H **Reserved**: Reserved, do not use 18_H **CDTM0_DTM5_3_N**: Inverted dead-time output of ATOM0_7 19_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
SELx (x=1)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT139 (P23.7) 01_H **CDTM0_DTM5_1_N**: Inverted dead-time output of ATOM0_5 02_H **CDTM1_DTM5_1**: Dead-time output of ATOM1_5 03_H **CDTM2_DTM5_1_N**: Inverted dead-time output of ATOM2_5 04_H **CDTM0_DTM1_1_N**: Inverted dead-time output of TOM0_5 05_H **CDTM1_DTM1_1**: Dead-time output of TOM1_5 06_H **CDTM2_DTM1_1_N**: Inverted dead-time output of TOM2_5 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM2_DTM5_2**: Dead-time output of ATOM2_6 11_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=2)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT140 (P32.5) 01_H **CDTM0_DTM5_2**: Dead-time output of ATOM0_6 02_H **CDTM1_DTM5_2_N**: Inverted dead-time output of ATOM1_6 03_H **CDTM2_DTM5_2**: Dead-time output of ATOM2_6 04_H **CDTM0_DTM1_2**: Dead-time output of TOM0_6 05_H **CDTM1_DTM1_2_N**: Inverted dead-time output of TOM1_6 06_H **CDTM2_DTM1_2**: Dead-time output of TOM2_6 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM2_DTM5_3**: Dead-time output of ATOM2_7 11_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
SELx (x=3)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT141 (P32.6) 01_H **CDTM0_DTM5_2_N**: Inverted dead-time output of ATOM0_6 02_H **CDTM1_DTM5_2**: Dead-time output of ATOM1_6 03_H **CDTM2_DTM5_2_N**: Inverted dead-time output of ATOM2_6 04_H **CDTM0_DTM1_2_N**: Inverted dead-time output of TOM0_6 05_H **CDTM1_DTM1_2**: Dead-time output of TOM1_6 06_H **CDTM2_DTM1_2_N**: Inverted dead-time output of TOM2_6 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM1_DTM2_0**: Dead-time output of TOM1_8 11_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
SELx (x=4)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT142 (P32.7) 01_H **CDTM0_DTM5_3**: Dead-time output of ATOM0_7 02_H **CDTM1_DTM5_3_N**: Inverted dead-time output of ATOM1_7 03_H **CDTM2_DTM5_3**: Dead-time output of ATOM2_7 04_H **CDTM0_DTM1_3**: Dead-time output of TOM0_7 05_H **CDTM1_DTM1_3_N**: Inverted dead-time output of TOM1_7 06_H **CDTM2_DTM1_3**: Dead-time output of TOM2_7 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM1_DTM2_1**: Dead-time output of TOM1_9 11_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=5)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT143 (P33.14) 01_H **CDTM0_DTM5_3_N**: Inverted dead-time output of ATOM0_7 02_H **CDTM1_DTM5_3**: Dead-time output of ATOM1_7 03_H **CDTM2_DTM5_3_N**: Inverted dead-time output of ATOM2_7 04_H **CDTM0_DTM1_3_N**: Inverted dead-time output of TOM0_7 05_H **CDTM1_DTM1_3**: Dead-time output of TOM1_7 06_H **CDTM2_DTM1_3_N**: Inverted dead-time output of TOM2_7 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM1_DTM2_2**: Dead-time output of TOM1_10 11_H **Reserved**: Reserved, do not use … 13_H **Reserved**: Reserved, do not use 14_H **CDTM0_DTM1_0_N**: Inverted dead-time output of TOM0_4 15_H **CDTM1_DTM1_0_N**: Inverted dead-time output of TOM1_4 16_H **CDTM0_DTM0_2_N**: Inverted dead-time output of TOM0_2 17_H **CDTM1_DTM0_2_N**: Inverted dead-time output of TOM1_2 18_H **CDTM2_DTM4_3_N**: Inverted dead-time output of ATOM2_3 19_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	15, 31	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.34 eGTM to Ports signal TOUT output selection register n (n=24)

For a description of this register, see 7579.

EGTM_TOUTSELn (n=24) Offset address: 061100_H+n*4

eGTM to Ports signal TOUT output selection register n Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	SEL5					SEL4					SEL3				
r	rw					rw					rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	SEL2					SEL1					SEL0				
r	rw					rw					rw				

Field	Bits	Type	Description
SELx (x=0)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT144 (P33.15) 01_H **CDTM0_DTM4_0**: Dead-time output of ATOM0_0 02_H **CDTM1_DTM4_0_N**: Inverted dead-time output of ATOM1_0 03_H **CDTM2_DTM4_0**: Dead-time output of ATOM2_0 04_H **CDTM0_DTM2_0**: Dead-time output of TOM0_8 05_H **CDTM1_DTM2_0_N**: Inverted dead-time output of TOM1_8 06_H **CDTM2_DTM2_0**: Dead-time output of TOM2_8 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM1_DTM2_3**: Dead-time output of TOM1_11 11_H **Reserved**: Reserved, do not use … 13_H **Reserved**: Reserved, do not use 14_H **CDTM0_DTM1_1_N**: Inverted dead-time output of TOM0_5 15_H **CDTM1_DTM1_1_N**: Inverted dead-time output of TOM1_5 16_H **CDTM0_DTM0_3_N**: Inverted dead-time output of TOM0_3 17_H **CDTM1_DTM0_3_N**: Inverted dead-time output of TOM1_3 18_H **CDTM2_DTM4_0_N**: Inverted dead-time output of ATOM2_0 19_H **Reserved**: Reserved, do not use $1A_H$ **CDTM2_DTM5_2**: Dead-time output of ATOM2_6 $1B_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=1)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT145 (XTAL) 01_H **CDTM0_DTM4_0_N**: Inverted dead-time output of ATOM0_0 02_H **CDTM1_DTM4_0**: Dead-time output of ATOM1_0 03_H **CDTM2_DTM4_0_N**: Inverted dead-time output of ATOM2_0 04_H **CDTM0_DTM2_0_N**: Inverted dead-time output of TOM0_8 05_H **CDTM1_DTM2_0**: Dead-time output of TOM1_8 06_H **CDTM2_DTM2_0_N**: Inverted dead-time output of TOM2_8 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM1_DTM2_3**: Dead-time output of TOM1_11 11_H **Reserved**: Reserved, do not use … 13_H **Reserved**: Reserved, do not use 14_H **CDTM0_DTM1_1**: Dead-time output of TOM0_5 15_H **CDTM1_DTM1_1**: Dead-time output of TOM1_5 16_H **CDTM0_DTM0_3**: Dead-time output of TOM0_3 17_H **CDTM1_DTM0_3**: Dead-time output of TOM1_3 18_H **CDTM2_DTM4_0**: Dead-time output of ATOM2_0 19_H **Reserved**: Reserved, do not use $1A_H$ **CDTM2_DTM5_3_N**: Inverted dead-time output of ATOM2_7 $1B_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=2)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT146 (P34.1) 01_H **CDTM0_DTM4_1**: Dead-time output of ATOM0_1 02_H **CDTM1_DTM4_1_N**: Inverted dead-time output of ATOM1_1 03_H **CDTM2_DTM4_1**: Dead-time output of ATOM2_1 04_H **CDTM0_DTM2_1**: Dead-time output of TOM0_9 05_H **CDTM1_DTM2_1_N**: Inverted dead-time output of TOM1_9 06_H **CDTM2_DTM2_1**: Dead-time output of TOM2_9 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM1_DTM3_1**: Dead-time output of TOM1_13 11_H **Reserved**: Reserved, do not use … 13_H **Reserved**: Reserved, do not use 14_H **CDTM0_DTM1_2_N**: Inverted dead-time output of TOM0_6 15_H **CDTM1_DTM1_2_N**: Inverted dead-time output of TOM1_6 16_H **CDTM0_DTM1_0_N**: Inverted dead-time output of TOM0_4 17_H **CDTM1_DTM1_0_N**: Inverted dead-time output of TOM1_4 18_H **CDTM0_DTM5_0**: Dead-time output of ATOM0_4 19_H **CDTM1_DTM5_0**: Dead-time output of ATOM1_4 $1A_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=3)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT147 (P34.2) 01_H **CDTM0_DTM4_1_N**: Inverted dead-time output of ATOM0_1 02_H **CDTM1_DTM4_1**: Dead-time output of ATOM1_1 03_H **CDTM2_DTM4_1_N**: Inverted dead-time output of ATOM2_1 04_H **CDTM0_DTM2_1_N**: Inverted dead-time output of TOM0_9 05_H **CDTM1_DTM2_1**: Dead-time output of TOM1_9 06_H **CDTM2_DTM2_1_N**: Inverted dead-time output of TOM2_9 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM1_DTM3_2**: Dead-time output of TOM1_14 11_H **Reserved**: Reserved, do not use … 13_H **Reserved**: Reserved, do not use 14_H **CDTM0_DTM1_0_N**: Inverted dead-time output of TOM0_4 15_H **CDTM1_DTM1_0_N**: Inverted dead-time output of TOM1_4 16_H **CDTM0_DTM1_1_N**: Inverted dead-time output of TOM0_5 17_H **CDTM1_DTM1_1_N**: Inverted dead-time output of TOM1_5 18_H **CDTM0_DTM5_0_N**: Inverted dead-time output of ATOM0_4 19_H **CDTM1_DTM5_0_N**: Inverted dead-time output of ATOM1_4 $1A_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=4)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT148 (P34.3) 01_H **CDTM0_DTM4_2**: Dead-time output of ATOM0_2 02_H **CDTM1_DTM4_2_N**: Inverted dead-time output of ATOM1_2 03_H **CDTM2_DTM4_2**: Dead-time output of ATOM2_2 04_H **CDTM0_DTM2_2**: Dead-time output of TOM0_10 05_H **CDTM1_DTM2_2_N**: Inverted dead-time output of TOM1_10 06_H **CDTM2_DTM2_2**: Dead-time output of TOM2_10 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM1_DTM3_3**: Dead-time output of TOM1_15 11_H **Reserved**: Reserved, do not use … 13_H **Reserved**: Reserved, do not use 14_H **CDTM0_DTM1_1_N**: Inverted dead-time output of TOM0_5 15_H **CDTM1_DTM1_1_N**: Inverted dead-time output of TOM1_5 16_H **CDTM0_DTM1_2_N**: Inverted dead-time output of TOM0_6 17_H **CDTM1_DTM1_2_N**: Inverted dead-time output of TOM1_6 18_H **CDTM0_DTM5_1_N**: Inverted dead-time output of ATOM0_5 19_H **CDTM1_DTM5_1_N**: Inverted dead-time output of ATOM1_5 $1A_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=5)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT149 (P34.4) 01_H **CDTM0_DTM4_2_N**: Inverted dead-time output of ATOM0_2 02_H **CDTM1_DTM4_2**: Dead-time output of ATOM1_2 03_H **CDTM2_DTM4_2_N**: Inverted dead-time output of ATOM2_2 04_H **CDTM0_DTM2_2_N**: Inverted dead-time output of TOM0_10 05_H **CDTM1_DTM2_2**: Dead-time output of TOM1_10 06_H **CDTM2_DTM2_2_N**: Inverted dead-time output of TOM2_10 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM2_DTM3_3**: Dead-time output of TOM2_15 11_H **Reserved**: Reserved, do not use … 13_H **Reserved**: Reserved, do not use 14_H **CDTM0_DTM1_2_N**: Inverted dead-time output of TOM0_6 15_H **CDTM1_DTM1_2_N**: Inverted dead-time output of TOM1_6 16_H **CDTM0_DTM1_3_N**: Inverted dead-time output of TOM0_7 17_H **CDTM1_DTM1_3_N**: Inverted dead-time output of TOM1_7 18_H **CDTM0_DTM5_2_N**: Inverted dead-time output of ATOM0_6 19_H **CDTM1_DTM5_2_N**: Inverted dead-time output of ATOM1_6 $1A_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	15, 31	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.35 eGTM to Ports signal TOUT output selection register n (n=25)

For a description of this register, see 7579.

EGTM_TOUTSELn (n=25) Offset address: 061100_H+n*4

eGTM to Ports signal TOUT output selection register n Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	SEL5					SEL4					SEL3				
r	rw					rw					rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	SEL2					SEL1					SEL0				
r	rw					rw					rw				

Field	Bits	Type	Description
SELx (x=0)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT150 (P34.5) 01_H **CDTM0_DTM4_3**: Dead-time output of ATOM0_3 02_H **CDTM1_DTM4_3_N**: Inverted dead-time output of ATOM1_3 03_H **CDTM2_DTM4_3**: Dead-time output of ATOM2_3 04_H **CDTM0_DTM2_3**: Dead-time output of TOM0_11 05_H **CDTM1_DTM2_3_N**: Inverted dead-time output of TOM1_11 06_H **CDTM2_DTM2_3**: Dead-time output of TOM2_11 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM1_DTM3_3**: Dead-time output of TOM1_15 11_H **Reserved**: Reserved, do not use … 13_H **Reserved**: Reserved, do not use 14_H **CDTM0_DTM1_3_N**: Inverted dead-time output of TOM0_7 15_H **CDTM1_DTM1_3_N**: Inverted dead-time output of TOM1_7 16_H **CDTM1_DTM0_0_N**: Inverted dead-time output of TOM1_0 17_H **CDTM2_DTM0_0_N**: Inverted dead-time output of TOM2_0 18_H **CDTM0_DTM5_3_N**: Inverted dead-time output of ATOM0_7 19_H **CDTM1_DTM5_3_N**: Inverted dead-time output of ATOM1_7 $1A_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
SELx (x=1)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT151 (P02.12) 01_H **CDTM0_DTM4_3_N**: Inverted dead-time output of ATOM0_3 02_H **CDTM1_DTM4_3**: Dead-time output of ATOM1_3 03_H **CDTM2_DTM4_3_N**: Inverted dead-time output of ATOM2_3 04_H **CDTM0_DTM2_3_N**: Inverted dead-time output of TOM0_11 05_H **CDTM1_DTM2_3**: Dead-time output of TOM1_11 06_H **CDTM2_DTM2_3_N**: Inverted dead-time output of TOM2_11 07_H **Reserved**: Reserved, do not use … 15_H **Reserved**: Reserved, do not use 16_H **CDTM0_DTM0_0**: Dead-time output of TOM0_0 17_H **CDTM1_DTM0_0_N**: Inverted dead-time output of TOM1_0 18_H **CDTM1_DTM4_0_N**: Inverted dead-time output of ATOM1_0 19_H **CDTM2_DTM4_0_N**: Inverted dead-time output of ATOM2_0 $1A_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=2)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT152 (P02.15) 01_H **CDTM0_DTM5_0**: Dead-time output of ATOM0_4 02_H **CDTM1_DTM5_0_N**: Inverted dead-time output of ATOM1_4 03_H **CDTM2_DTM5_0**: Dead-time output of ATOM2_4 04_H **CDTM0_DTM3_0**: Dead-time output of TOM0_12 05_H **CDTM1_DTM3_0_N**: Inverted dead-time output of TOM1_12 06_H **CDTM2_DTM3_0**: Dead-time output of TOM2_12 07_H **Reserved**: Reserved, do not use … 15_H **Reserved**: Reserved, do not use 16_H **CDTM0_DTM0_1**: Dead-time output of TOM0_1 17_H **CDTM1_DTM0_1_N**: Inverted dead-time output of TOM1_1 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
SELx (x=3)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT153 (P02.13) 01_H **CDTM0_DTM5_0_N**: Inverted dead-time output of ATOM0_4 02_H **CDTM1_DTM5_0**: Dead-time output of ATOM1_4 03_H **CDTM2_DTM5_0_N**: Inverted dead-time output of ATOM2_4 04_H **CDTM0_DTM3_0_N**: Inverted dead-time output of TOM0_12 05_H **CDTM1_DTM3_0**: Dead-time output of TOM1_12 06_H **CDTM2_DTM3_0_N**: Inverted dead-time output of TOM2_12 07_H **Reserved**: Reserved, do not use … 15_H **Reserved**: Reserved, do not use 16_H **CDTM0_DTM0_2**: Dead-time output of TOM0_2 17_H **CDTM1_DTM0_2_N**: Inverted dead-time output of TOM1_2 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=4)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT154 (P02.14) 01_H **CDTM0_DTM5_1**: Dead-time output of ATOM0_5 02_H **CDTM1_DTM5_1_N**: Inverted dead-time output of ATOM1_5 03_H **CDTM2_DTM5_1**: Dead-time output of ATOM2_5 04_H **CDTM0_DTM3_1**: Dead-time output of TOM0_13 05_H **CDTM1_DTM3_1_N**: Inverted dead-time output of TOM1_13 06_H **CDTM2_DTM3_1**: Dead-time output of TOM2_13 07_H **Reserved**: Reserved, do not use … 15_H **Reserved**: Reserved, do not use 16_H **CDTM0_DTM0_3**: Dead-time output of TOM0_3 17_H **CDTM1_DTM0_3_N**: Inverted dead-time output of TOM1_3 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
SELx (x=5)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT155 (P01.0) 01_H **CDTM0_DTM5_1_N**: Inverted dead-time output of ATOM0_5 02_H **CDTM1_DTM5_1**: Dead-time output of ATOM1_5 03_H **CDTM2_DTM5_1_N**: Inverted dead-time output of ATOM2_5 04_H **CDTM0_DTM3_1_N**: Inverted dead-time output of TOM0_13 05_H **CDTM1_DTM3_1**: Dead-time output of TOM1_13 06_H **CDTM2_DTM3_1_N**: Inverted dead-time output of TOM2_13 07_H **Reserved**: Reserved, do not use … 15_H **Reserved**: Reserved, do not use 16_H **CDTM0_DTM1_0**: Dead-time output of TOM0_4 17_H **CDTM1_DTM1_0_N**: Inverted dead-time output of TOM1_4 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	15, 31	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.36 eGTM to Ports signal TOUT output selection register n (n=26)

For a description of this register, see 7579.

EGTM_TOUTSELn (n=26) Offset address: 061100_H+n*4

eGTM to Ports signal TOUT output selection register n Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	SEL5					SEL4					SEL3				
r	rw					rw					rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	SEL2					SEL1					SEL0				
r	rw					rw					rw				

Field	Bits	Type	Description
SELx (x=0)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT156 (P01.2) 01_H **CDTM0_DTM5_2**: Dead-time output of ATOM0_6 02_H **CDTM1_DTM5_2_N**: Inverted dead-time output of ATOM1_6 03_H **CDTM2_DTM5_2**: Dead-time output of ATOM2_6 04_H **CDTM0_DTM3_2**: Dead-time output of TOM0_14 05_H **CDTM1_DTM3_2_N**: Inverted dead-time output of TOM1_14 06_H **CDTM2_DTM3_2**: Dead-time output of TOM2_14 07_H **Reserved**: Reserved, do not use … 15_H **Reserved**: Reserved, do not use 16_H **CDTM0_DTM1_1**: Dead-time output of TOM0_5 17_H **CDTM1_DTM1_1_N**: Inverted dead-time output of TOM1_5 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
SELx (x=1)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT157 (P01.15) 01_H **CDTM0_DTM5_2_N**: Inverted dead-time output of ATOM0_6 02_H **CDTM1_DTM5_2**: Dead-time output of ATOM1_6 03_H **CDTM2_DTM5_2_N**: Inverted dead-time output of ATOM2_6 04_H **CDTM0_DTM3_2_N**: Inverted dead-time output of TOM0_14 05_H **CDTM1_DTM3_2**: Dead-time output of TOM1_14 06_H **CDTM2_DTM3_2_N**: Inverted dead-time output of TOM2_14 07_H **Reserved**: Reserved, do not use … 15_H **Reserved**: Reserved, do not use 16_H **CDTM0_DTM1_2**: Dead-time output of TOM0_6 17_H **CDTM1_DTM1_2_N**: Inverted dead-time output of TOM1_6 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=2)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT158 (P01.12) 01_H **CDTM0_DTM5_3**: Dead-time output of ATOM0_7 02_H **CDTM1_DTM5_3_N**: Inverted dead-time output of ATOM1_7 03_H **CDTM2_DTM5_3**: Dead-time output of ATOM2_7 04_H **CDTM0_DTM3_3**: Dead-time output of TOM0_15 05_H **CDTM1_DTM3_3_N**: Inverted dead-time output of TOM1_15 06_H **CDTM2_DTM3_3**: Dead-time output of TOM2_15 07_H **Reserved**: Reserved, do not use … 15_H **Reserved**: Reserved, do not use 16_H **CDTM0_DTM1_3**: Dead-time output of TOM0_7 17_H **CDTM1_DTM1_3_N**: Inverted dead-time output of TOM1_7 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
SELx (x=3)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT159 (P01.1) 01_H **CDTM0_DTM5_3_N**: Inverted dead-time output of ATOM0_7 02_H **CDTM1_DTM5_3**: Dead-time output of ATOM1_7 03_H **CDTM2_DTM5_3_N**: Inverted dead-time output of ATOM2_7 04_H **CDTM0_DTM3_3_N**: Inverted dead-time output of TOM0_15 05_H **CDTM1_DTM3_3**: Dead-time output of TOM1_15 06_H **CDTM2_DTM3_3_N**: Inverted dead-time output of TOM2_15 07_H **Reserved**: Reserved, do not use … 15_H **Reserved**: Reserved, do not use 16_H **CDTM1_DTM0_0**: Dead-time output of TOM1_0 17_H **CDTM2_DTM0_0_N**: Inverted dead-time output of TOM2_0 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=4)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT160 (P01.9) 01_H **CDTM0_DTM4_0**: Dead-time output of ATOM0_0 02_H **CDTM1_DTM4_0_N**: Inverted dead-time output of ATOM1_0 03_H **CDTM2_DTM4_0**: Dead-time output of ATOM2_0 04_H **CDTM0_DTM0_0**: Dead-time output of TOM0_0 05_H **CDTM1_DTM0_0_N**: Inverted dead-time output of TOM1_0 06_H **CDTM2_DTM0_0**: Dead-time output of TOM2_0 07_H **Reserved**: Reserved, do not use … 15_H **Reserved**: Reserved, do not use 16_H **CDTM1_DTM0_1**: Dead-time output of TOM1_1 17_H **CDTM2_DTM0_1_N**: Inverted dead-time output of TOM2_1 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
SELx (x=5)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT161 (P01.13) 01_H **CDTM0_DTM4_0_N**: Inverted dead-time output of ATOM0_0 02_H **CDTM1_DTM4_0**: Dead-time output of ATOM1_0 03_H **CDTM2_DTM4_0_N**: Inverted dead-time output of ATOM2_0 04_H **CDTM0_DTM0_0_N**: Inverted dead-time output of TOM0_0 05_H **CDTM1_DTM0_0**: Dead-time output of TOM1_0 06_H **CDTM2_DTM0_0_N**: Inverted dead-time output of TOM2_0 07_H **Reserved**: Reserved, do not use … 15_H **Reserved**: Reserved, do not use 16_H **CDTM1_DTM0_2**: Dead-time output of TOM1_2 17_H **CDTM2_DTM0_2_N**: Inverted dead-time output of TOM2_2 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	15, 31	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.37 eGTM to Ports signal TOUT output selection register n (n=27)

For a description of this register, see 7579.

EGTM_TOUTSELn (n=27) Offset address: 061100_H+n*4

eGTM to Ports signal TOUT output selection register n Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	SEL5					SEL4					SEL3				
r	rw					rw					rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	SEL2					SEL1					SEL0				
r	rw					rw					rw				

Field	Bits	Type	Description
SELx (x=0)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT162 (P01.8) 01_H **CDTM0_DTM4_1**: Dead-time output of ATOM0_1 02_H **CDTM1_DTM4_1_N**: Inverted dead-time output of ATOM1_1 03_H **CDTM2_DTM4_1**: Dead-time output of ATOM2_1 04_H **CDTM0_DTM0_1**: Dead-time output of TOM0_1 05_H **CDTM1_DTM0_1_N**: Inverted dead-time output of TOM1_1 06_H **CDTM2_DTM0_1**: Dead-time output of TOM2_1 07_H **Reserved**: Reserved, do not use … 15_H **Reserved**: Reserved, do not use 16_H **CDTM1_DTM0_3**: Dead-time output of TOM1_3 17_H **CDTM2_DTM0_3_N**: Inverted dead-time output of TOM2_3 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
SELx (x=1)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT163 (P01.10) 01_H **CDTM0_DTM4_1_N**: Inverted dead-time output of ATOM0_1 02_H **CDTM1_DTM4_1**: Dead-time output of ATOM1_1 03_H **CDTM2_DTM4_1_N**: Inverted dead-time output of ATOM2_1 04_H **CDTM0_DTM0_1_N**: Inverted dead-time output of TOM0_1 05_H **CDTM1_DTM0_1**: Dead-time output of TOM1_1 06_H **CDTM2_DTM0_1_N**: Inverted dead-time output of TOM2_1 07_H **Reserved**: Reserved, do not use … 15_H **Reserved**: Reserved, do not use 16_H **CDTM1_DTM1_0**: Dead-time output of TOM1_4 17_H **CDTM2_DTM1_0_N**: Inverted dead-time output of TOM2_4 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=2)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT164 (P01.14) 01_H **CDTM0_DTM4_2**: Dead-time output of ATOM0_2 02_H **CDTM1_DTM4_2_N**: Inverted dead-time output of ATOM1_2 03_H **CDTM2_DTM4_2**: Dead-time output of ATOM2_2 04_H **CDTM0_DTM0_2**: Dead-time output of TOM0_2 05_H **CDTM1_DTM0_2_N**: Inverted dead-time output of TOM1_2 06_H **CDTM2_DTM0_2**: Dead-time output of TOM2_2 07_H **Reserved**: Reserved, do not use … 15_H **Reserved**: Reserved, do not use 16_H **CDTM1_DTM1_1**: Dead-time output of TOM1_5 17_H **CDTM2_DTM1_1_N**: Inverted dead-time output of TOM2_5 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
SELx (x=3)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT165 (P01.11) 01_H **CDTM0_DTM4_2_N**: Inverted dead-time output of ATOM0_2 02_H **CDTM1_DTM4_2**: Dead-time output of ATOM1_2 03_H **CDTM2_DTM4_2_N**: Inverted dead-time output of ATOM2_2 04_H **CDTM0_DTM0_2_N**: Inverted dead-time output of TOM0_2 05_H **CDTM1_DTM0_2**: Dead-time output of TOM1_2 06_H **CDTM2_DTM0_2_N**: Inverted dead-time output of TOM2_2 07_H **Reserved**: Reserved, do not use … 15_H **Reserved**: Reserved, do not use 16_H **CDTM1_DTM1_2**: Dead-time output of TOM1_6 17_H **CDTM2_DTM1_2_N**: Inverted dead-time output of TOM2_6 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=4)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT166 (P00.14) 01_H **CDTM0_DTM4_3**: Dead-time output of ATOM0_3 02_H **CDTM1_DTM4_3_N**: Inverted dead-time output of ATOM1_3 03_H **CDTM2_DTM4_3**: Dead-time output of ATOM2_3 04_H **CDTM0_DTM0_3**: Dead-time output of TOM0_3 05_H **CDTM1_DTM0_3_N**: Inverted dead-time output of TOM1_3 06_H **CDTM2_DTM0_3**: Dead-time output of TOM2_3 07_H **Reserved**: Reserved, do not use … 15_H **Reserved**: Reserved, do not use 16_H **CDTM1_DTM1_3**: Dead-time output of TOM1_7 17_H **CDTM2_DTM1_3_N**: Inverted dead-time output of TOM2_7 18_H **CDTM2_DTM5_1**: Dead-time output of ATOM2_5 19_H **Reserved**: Reserved, do not use $1A_H$ **CDTM2_DTM4_2**: Dead-time output of ATOM2_2 $1B_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
SELx (x=5)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT167 (P00.13) 01_H **CDTM0_DTM4_3_N**: Inverted dead-time output of ATOM0_3 02_H **CDTM1_DTM4_3**: Dead-time output of ATOM1_3 03_H **CDTM2_DTM4_3_N**: Inverted dead-time output of ATOM2_3 04_H **CDTM0_DTM0_3_N**: Inverted dead-time output of TOM0_3 05_H **CDTM1_DTM0_3**: Dead-time output of TOM1_3 06_H **CDTM2_DTM0_3_N**: Inverted dead-time output of TOM2_3 07_H **Reserved**: Reserved, do not use … 15_H **Reserved**: Reserved, do not use 16_H **CDTM1_DTM2_0**: Dead-time output of TOM1_8 17_H **CDTM2_DTM2_0**: Dead-time output of TOM2_8 18_H **CDTM2_DTM5_0**: Dead-time output of ATOM2_4 19_H **Reserved**: Reserved, do not use $1A_H$ **CDTM2_DTM4_1**: Dead-time output of ATOM2_1 $1B_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	15, 31	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.38 eGTM to Ports signal TOUT output selection register n (n=28)

For a description of this register, see 7579.

EGTM_TOUTSELn (n=28) Offset address: 061100_H+n*4

eGTM to Ports signal TOUT output selection register n Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	SEL5					SEL4					SEL3				
r	rw					rw					rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	SEL2					SEL1					SEL0				
r	rw					rw					rw				

Field	Bits	Type	Description
SELx (x=0)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT168 (P00.15) 01_H **CDTM0_DTM5_0**: Dead-time output of ATOM0_4 02_H **CDTM1_DTM5_0_N**: Inverted dead-time output of ATOM1_4 03_H **CDTM2_DTM5_0**: Dead-time output of ATOM2_4 04_H **CDTM0_DTM1_0**: Dead-time output of TOM0_4 05_H **CDTM1_DTM1_0_N**: Inverted dead-time output of TOM1_4 06_H **CDTM2_DTM1_0**: Dead-time output of TOM2_4 07_H **Reserved**: Reserved, do not use … 15_H **Reserved**: Reserved, do not use 16_H **CDTM1_DTM2_1**: Dead-time output of TOM1_9 17_H **CDTM2_DTM2_1**: Dead-time output of TOM2_9 18_H **CDTM2_DTM5_2**: Dead-time output of ATOM2_6 19_H **Reserved**: Reserved, do not use $1A_H$ **CDTM2_DTM4_3**: Dead-time output of ATOM2_3 $1B_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
SELx (x=1)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT169 (no pin) 01_H **CDTM0_DTM5_0_N**: Inverted dead-time output of ATOM0_4 02_H **CDTM1_DTM5_0**: Dead-time output of ATOM1_4 03_H **CDTM2_DTM5_0_N**: Inverted dead-time output of ATOM2_4 04_H **CDTM0_DTM1_0_N**: Inverted dead-time output of TOM0_4 05_H **CDTM1_DTM1_0**: Dead-time output of TOM1_4 06_H **CDTM2_DTM1_0_N**: Inverted dead-time output of TOM2_4 07_H **Reserved**: Reserved, do not use … 15_H **Reserved**: Reserved, do not use 16_H **CDTM1_DTM2_2**: Dead-time output of TOM1_10 17_H **CDTM2_DTM2_2**: Dead-time output of TOM2_10 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=2)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT170 (no pin) 01_H **CDTM0_DTM5_1**: Dead-time output of ATOM0_5 02_H **CDTM1_DTM5_1_N**: Inverted dead-time output of ATOM1_5 03_H **CDTM2_DTM5_1**: Dead-time output of ATOM2_5 04_H **CDTM0_DTM1_1**: Dead-time output of TOM0_5 05_H **CDTM1_DTM1_1_N**: Inverted dead-time output of TOM1_5 06_H **CDTM2_DTM1_1**: Dead-time output of TOM2_5 07_H **Reserved**: Reserved, do not use … 15_H **Reserved**: Reserved, do not use 16_H **CDTM1_DTM2_3**: Dead-time output of TOM1_11 17_H **CDTM2_DTM2_3**: Dead-time output of TOM2_11 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
SELx (x=3)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT171 (no pin) 01_H **CDTM0_DTM5_1_N**: Inverted dead-time output of ATOM0_5 02_H **CDTM1_DTM5_1**: Dead-time output of ATOM1_5 03_H **CDTM2_DTM5_1_N**: Inverted dead-time output of ATOM2_5 04_H **CDTM0_DTM1_1_N**: Inverted dead-time output of TOM0_5 05_H **CDTM1_DTM1_1**: Dead-time output of TOM1_5 06_H **CDTM2_DTM1_1_N**: Inverted dead-time output of TOM2_5 07_H **Reserved**: Reserved, do not use … 15_H **Reserved**: Reserved, do not use 16_H **CDTM1_DTM3_0**: Dead-time output of TOM1_12 17_H **CDTM2_DTM3_0**: Dead-time output of TOM2_12 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=4)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT172 (no pin) 01_H **CDTM0_DTM5_2**: Dead-time output of ATOM0_6 02_H **CDTM1_DTM5_2_N**: Inverted dead-time output of ATOM1_6 03_H **CDTM2_DTM5_2**: Dead-time output of ATOM2_6 04_H **CDTM0_DTM1_2**: Dead-time output of TOM0_6 05_H **CDTM1_DTM1_2_N**: Inverted dead-time output of TOM1_6 06_H **CDTM2_DTM1_2**: Dead-time output of TOM2_6 07_H **Reserved**: Reserved, do not use … 15_H **Reserved**: Reserved, do not use 16_H **CDTM1_DTM3_1**: Dead-time output of TOM1_13 17_H **CDTM2_DTM3_1**: Dead-time output of TOM2_13 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
SELx (x=5)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT173 (no pin) 01_H **CDTM0_DTM5_2_N**: Inverted dead-time output of ATOM0_6 02_H **CDTM1_DTM5_2**: Dead-time output of ATOM1_6 03_H **CDTM2_DTM5_2_N**: Inverted dead-time output of ATOM2_6 04_H **CDTM0_DTM1_2_N**: Inverted dead-time output of TOM0_6 05_H **CDTM1_DTM1_2**: Dead-time output of TOM1_6 06_H **CDTM2_DTM1_2_N**: Inverted dead-time output of TOM2_6 07_H **Reserved**: Reserved, do not use … 15_H **Reserved**: Reserved, do not use 16_H **CDTM1_DTM3_2**: Dead-time output of TOM1_14 17_H **CDTM2_DTM3_2**: Dead-time output of TOM2_14 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	15, 31	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.39 eGTM to Ports signal TOUT output selection register n (n=29)

For a description of this register, see 7579.

EGTM_TOUTSELn (n=29) Offset address: 061100_H+n*4

eGTM to Ports signal TOUT output selection register n Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	SEL5					SEL4					SEL3				
r	rw					rw					rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	SEL2					SEL1					SEL0				
r	rw					rw					rw				

Field	Bits	Type	Description
SELx (x=0)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT174 (P31.0) 01_H **CDTM0_DTM5_3**: Dead-time output of ATOM0_7 02_H **CDTM1_DTM5_3_N**: Inverted dead-time output of ATOM1_7 03_H **CDTM2_DTM5_3**: Dead-time output of ATOM2_7 04_H **CDTM0_DTM1_3**: Dead-time output of TOM0_7 05_H **CDTM1_DTM1_3_N**: Inverted dead-time output of TOM1_7 06_H **CDTM2_DTM1_3**: Dead-time output of TOM2_7 07_H **Reserved**: Reserved, do not use … 15_H **Reserved**: Reserved, do not use 16_H **CDTM1_DTM3_3**: Dead-time output of TOM1_15 17_H **CDTM2_DTM3_3**: Dead-time output of TOM2_15 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
SELx (x=1)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT175 (P31.1) 01_H **CDTM0_DTM5_3_N**: Inverted dead-time output of ATOM0_7 02_H **CDTM1_DTM5_3**: Dead-time output of ATOM1_7 03_H **CDTM2_DTM5_3_N**: Inverted dead-time output of ATOM2_7 04_H **CDTM0_DTM1_3_N**: Inverted dead-time output of TOM0_7 05_H **CDTM1_DTM1_3**: Dead-time output of TOM1_7 06_H **CDTM2_DTM1_3_N**: Inverted dead-time output of TOM2_7 07_H **Reserved**: Reserved, do not use … 16_H **Reserved**: Reserved, do not use 17_H **CDTM0_DTM0_0**: Dead-time output of TOM0_0 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=2)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT176 (P31.2) 01_H **CDTM0_DTM4_0**: Dead-time output of ATOM0_0 02_H **CDTM1_DTM4_0_N**: Inverted dead-time output of ATOM1_0 03_H **CDTM2_DTM4_0**: Dead-time output of ATOM2_0 04_H **CDTM0_DTM2_0**: Dead-time output of TOM0_8 05_H **CDTM1_DTM2_0_N**: Inverted dead-time output of TOM1_8 06_H **CDTM2_DTM2_0**: Dead-time output of TOM2_8 07_H **Reserved**: Reserved, do not use … 16_H **Reserved**: Reserved, do not use 17_H **CDTM0_DTM0_1**: Dead-time output of TOM0_1 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
SELx (x=3)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT177 (P31.3) 01_H **CDTM0_DTM4_0_N**: Inverted dead-time output of ATOM0_0 02_H **CDTM1_DTM4_0**: Dead-time output of ATOM1_0 03_H **CDTM2_DTM4_0_N**: Inverted dead-time output of ATOM2_0 04_H **CDTM0_DTM2_0_N**: Inverted dead-time output of TOM0_8 05_H **CDTM1_DTM2_0**: Dead-time output of TOM1_8 06_H **CDTM2_DTM2_0_N**: Inverted dead-time output of TOM2_8 07_H **Reserved**: Reserved, do not use … 16_H **Reserved**: Reserved, do not use 17_H **CDTM0_DTM0_2**: Dead-time output of TOM0_2 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
SELx (x=4)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT178 (P31.4) 01_H **CDTM0_DTM4_1**: Dead-time output of ATOM0_1 02_H **CDTM1_DTM4_1_N**: Inverted dead-time output of ATOM1_1 03_H **CDTM2_DTM4_1**: Dead-time output of ATOM2_1 04_H **CDTM0_DTM2_1**: Dead-time output of TOM0_9 05_H **CDTM1_DTM2_1_N**: Inverted dead-time output of TOM1_9 06_H **CDTM2_DTM2_1**: Dead-time output of TOM2_9 07_H **Reserved**: Reserved, do not use … 16_H **Reserved**: Reserved, do not use 17_H **CDTM0_DTM0_3**: Dead-time output of TOM0_3 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=5)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT179 (P31.5) 01_H **CDTM0_DTM4_1_N**: Inverted dead-time output of ATOM0_1 02_H **CDTM1_DTM4_1**: Dead-time output of ATOM1_1 03_H **CDTM2_DTM4_1_N**: Inverted dead-time output of ATOM2_1 04_H **CDTM0_DTM2_1_N**: Inverted dead-time output of TOM0_9 05_H **CDTM1_DTM2_1**: Dead-time output of TOM1_9 06_H **CDTM2_DTM2_1_N**: Inverted dead-time output of TOM2_9 07_H **Reserved**: Reserved, do not use … 16_H **Reserved**: Reserved, do not use 17_H **CDTM0_DTM1_0**: Dead-time output of TOM0_4 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	15, 31	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.40 eGTM to Ports signal TOUT output selection register n (n=30)

For a description of this register, see 7579.

EGTM_TOUTSELn (n=30) Offset address: 061100_H+n*4

eGTM to Ports signal TOUT output selection register n Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	SEL5					SEL4					SEL3				
r	rw					rw					rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	SEL2					SEL1					SEL0				
r	rw					rw					rw				

Field	Bits	Type	Description
SELx (x=0)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT180 (P31.6) 01_H **CDTM0_DTM4_2**: Dead-time output of ATOM0_2 02_H **CDTM1_DTM4_2_N**: Inverted dead-time output of ATOM1_2 03_H **CDTM2_DTM4_2**: Dead-time output of ATOM2_2 04_H **CDTM0_DTM2_2**: Dead-time output of TOM0_10 05_H **CDTM1_DTM2_2_N**: Inverted dead-time output of TOM1_10 06_H **CDTM2_DTM2_2**: Dead-time output of TOM2_10 07_H **Reserved**: Reserved, do not use … 16_H **Reserved**: Reserved, do not use 17_H **CDTM0_DTM1_1**: Dead-time output of TOM0_5 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
SELx (x=1)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT181 (P31.7) 01_H **CDTM0_DTM4_2_N**: Inverted dead-time output of ATOM0_2 02_H **CDTM1_DTM4_2**: Dead-time output of ATOM1_2 03_H **CDTM2_DTM4_2_N**: Inverted dead-time output of ATOM2_2 04_H **CDTM0_DTM2_2_N**: Inverted dead-time output of TOM0_10 05_H **CDTM1_DTM2_2**: Dead-time output of TOM1_10 06_H **CDTM2_DTM2_2_N**: Inverted dead-time output of TOM2_10 07_H **Reserved**: Reserved, do not use … 15_H **Reserved**: Reserved, do not use 16_H **CDTM1_DTM1_2**: Dead-time output of TOM1_6 17_H **CDTM0_DTM1_2**: Dead-time output of TOM0_6 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=2)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT182 (P35.0) 01_H **CDTM0_DTM4_3**: Dead-time output of ATOM0_3 02_H **CDTM1_DTM4_3_N**: Inverted dead-time output of ATOM1_3 03_H **CDTM2_DTM4_3**: Dead-time output of ATOM2_3 04_H **CDTM0_DTM2_3**: Dead-time output of TOM0_11 05_H **CDTM1_DTM2_3_N**: Inverted dead-time output of TOM1_11 06_H **CDTM2_DTM2_3**: Dead-time output of TOM2_11 07_H **Reserved**: Reserved, do not use … 15_H **Reserved**: Reserved, do not use 16_H **CDTM1_DTM1_3**: Dead-time output of TOM1_7 17_H **CDTM0_DTM1_3**: Dead-time output of TOM0_7 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
SELx (x=3)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT183 (P35.1) 01_H **CDTM0_DTM4_3_N**: Inverted dead-time output of ATOM0_3 02_H **CDTM1_DTM4_3**: Dead-time output of ATOM1_3 03_H **CDTM2_DTM4_3_N**: Inverted dead-time output of ATOM2_3 04_H **CDTM0_DTM2_3_N**: Inverted dead-time output of TOM0_11 05_H **CDTM1_DTM2_3**: Dead-time output of TOM1_11 06_H **CDTM2_DTM2_3_N**: Inverted dead-time output of TOM2_11 07_H **Reserved**: Reserved, do not use … 15_H **Reserved**: Reserved, do not use 16_H **CDTM1_DTM2_0**: Dead-time output of TOM1_8 17_H **CDTM0_DTM2_0**: Dead-time output of TOM0_8 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=4)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT184 (P35.2) 01_H **CDTM0_DTM5_0**: Dead-time output of ATOM0_4 02_H **CDTM1_DTM5_0_N**: Inverted dead-time output of ATOM1_4 03_H **CDTM2_DTM5_0**: Dead-time output of ATOM2_4 04_H **CDTM0_DTM3_0**: Dead-time output of TOM0_12 05_H **CDTM1_DTM3_0_N**: Inverted dead-time output of TOM1_12 06_H **CDTM2_DTM3_0**: Dead-time output of TOM2_12 07_H **Reserved**: Reserved, do not use … 15_H **Reserved**: Reserved, do not use 16_H **CDTM1_DTM2_1**: Dead-time output of TOM1_9 17_H **CDTM0_DTM2_1**: Dead-time output of TOM0_9 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
SELx (x=5)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT185 (P35.3) 01_H **CDTM0_DTM5_0_N**: Inverted dead-time output of ATOM0_4 02_H **CDTM1_DTM5_0**: Dead-time output of ATOM1_4 03_H **CDTM2_DTM5_0_N**: Inverted dead-time output of ATOM2_4 04_H **CDTM0_DTM3_0_N**: Inverted dead-time output of TOM0_12 05_H **CDTM1_DTM3_0**: Dead-time output of TOM1_12 06_H **CDTM2_DTM3_0_N**: Inverted dead-time output of TOM2_12 07_H **Reserved**: Reserved, do not use … 15_H **Reserved**: Reserved, do not use 16_H **CDTM1_DTM2_2**: Dead-time output of TOM1_10 17_H **CDTM0_DTM2_2**: Dead-time output of TOM0_10 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	15, 31	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.41 eGTM to Ports signal TOUT output selection register n (n=31)

For a description of this register, see 7579.

EGTM_TOUTSELn (n=31) Offset address: 061100_H+n*4

eGTM to Ports signal TOUT output selection register n Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	SEL5					SEL4					SEL3				
r	rw					rw					rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	SEL2					SEL1					SEL0				
r	rw					rw					rw				

Field	Bits	Type	Description
SELx (x=0)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT186 (P35.4) 01_H **CDTM0_DTM5_1**: Dead-time output of ATOM0_5 02_H **CDTM1_DTM5_1_N**: Inverted dead-time output of ATOM1_5 03_H **CDTM2_DTM5_1**: Dead-time output of ATOM2_5 04_H **CDTM0_DTM3_1**: Dead-time output of TOM0_13 05_H **CDTM1_DTM3_1_N**: Inverted dead-time output of TOM1_13 06_H **CDTM2_DTM3_1**: Dead-time output of TOM2_13 07_H **Reserved**: Reserved, do not use … 15_H **Reserved**: Reserved, do not use 16_H **CDTM1_DTM2_3**: Dead-time output of TOM1_11 17_H **CDTM0_DTM2_3**: Dead-time output of TOM0_11 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
SELx (x=1)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT187 (P35.5) 01_H **CDTM0_DTM5_1_N**: Inverted dead-time output of ATOM0_5 02_H **CDTM1_DTM5_1**: Dead-time output of ATOM1_5 03_H **CDTM2_DTM5_1_N**: Inverted dead-time output of ATOM2_5 04_H **CDTM0_DTM3_1_N**: Inverted dead-time output of TOM0_13 05_H **CDTM1_DTM3_1**: Dead-time output of TOM1_13 06_H **CDTM2_DTM3_1_N**: Inverted dead-time output of TOM2_13 07_H **Reserved**: Reserved, do not use … 15_H **Reserved**: Reserved, do not use 16_H **CDTM1_DTM3_0**: Dead-time output of TOM1_12 17_H **CDTM0_DTM3_0**: Dead-time output of TOM0_12 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=2)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT188 (P04.6) 01_H **CDTM0_DTM5_2**: Dead-time output of ATOM0_6 02_H **CDTM1_DTM5_2_N**: Inverted dead-time output of ATOM1_6 03_H **CDTM2_DTM5_2**: Dead-time output of ATOM2_6 04_H **CDTM0_DTM3_2**: Dead-time output of TOM0_14 05_H **CDTM1_DTM3_2_N**: Inverted dead-time output of TOM1_14 06_H **CDTM2_DTM3_2**: Dead-time output of TOM2_14 07_H **Reserved**: Reserved, do not use … 15_H **Reserved**: Reserved, do not use 16_H **CDTM1_DTM3_1**: Dead-time output of TOM1_13 17_H **CDTM0_DTM3_1**: Dead-time output of TOM0_13 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
SELx (x=3)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT189 (P04.7) 01_H **CDTM0_DTM5_2_N**: Inverted dead-time output of ATOM0_6 02_H **CDTM1_DTM5_2**: Dead-time output of ATOM1_6 03_H **CDTM2_DTM5_2_N**: Inverted dead-time output of ATOM2_6 04_H **CDTM0_DTM3_2_N**: Inverted dead-time output of TOM0_14 05_H **CDTM1_DTM3_2**: Dead-time output of TOM1_14 06_H **CDTM2_DTM3_2_N**: Inverted dead-time output of TOM2_14 07_H **Reserved**: Reserved, do not use … 15_H **Reserved**: Reserved, do not use 16_H **CDTM1_DTM3_2**: Dead-time output of TOM1_14 17_H **CDTM0_DTM3_2**: Dead-time output of TOM0_14 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=4)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT190 (P30.0) 01_H **CDTM0_DTM5_3**: Dead-time output of ATOM0_7 02_H **CDTM1_DTM5_3_N**: Inverted dead-time output of ATOM1_7 03_H **CDTM2_DTM5_3**: Dead-time output of ATOM2_7 04_H **CDTM0_DTM3_3**: Dead-time output of TOM0_15 05_H **CDTM1_DTM3_3_N**: Inverted dead-time output of TOM1_15 06_H **CDTM2_DTM3_3**: Dead-time output of TOM2_15 07_H **Reserved**: Reserved, do not use … 15_H **Reserved**: Reserved, do not use 16_H **CDTM1_DTM3_3**: Dead-time output of TOM1_15 17_H **CDTM0_DTM3_3**: Dead-time output of TOM0_15 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
SELx (x=5)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT191 (P30.1) 01_H **CDTM0_DTM5_3_N**: Inverted dead-time output of ATOM0_7 02_H **CDTM1_DTM5_3**: Dead-time output of ATOM1_7 03_H **CDTM2_DTM5_3_N**: Inverted dead-time output of ATOM2_7 04_H **CDTM0_DTM3_3_N**: Inverted dead-time output of TOM0_15 05_H **CDTM1_DTM3_3**: Dead-time output of TOM1_15 06_H **CDTM2_DTM3_3_N**: Inverted dead-time output of TOM2_15 07_H **Reserved**: Reserved, do not use … 15_H **Reserved**: Reserved, do not use 16_H **CDTM2_DTM0_0**: Dead-time output of TOM2_0 17_H **CDTM1_DTM0_0**: Dead-time output of TOM1_0 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	15, 31	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.42 eGTM to Ports signal TOUT output selection register n (n=32)

For a description of this register, see 7579.

EGTM_TOUTSELn (n=32) Offset address: 061100_H+n*4

eGTM to Ports signal TOUT output selection register n Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	SEL5					SEL4					SEL3				
r	rw					rw					rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	SEL2					SEL1					SEL0				
r	rw					rw					rw				

Field	Bits	Type	Description
SELx (x=0)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT192 (P30.2) 01_H **CDTM0_DTM4_0**: Dead-time output of ATOM0_0 02_H **CDTM1_DTM4_0_N**: Inverted dead-time output of ATOM1_0 03_H **CDTM2_DTM4_0**: Dead-time output of ATOM2_0 04_H **CDTM0_DTM0_0**: Dead-time output of TOM0_0 05_H **CDTM1_DTM0_0_N**: Inverted dead-time output of TOM1_0 06_H **CDTM2_DTM0_0**: Dead-time output of TOM2_0 07_H **Reserved**: Reserved, do not use … 15_H **Reserved**: Reserved, do not use 16_H **CDTM2_DTM0_0**: Dead-time output of TOM2_0 17_H **CDTM1_DTM0_0**: Dead-time output of TOM1_0 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
SELx (x=1)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT193 (P30.3) 01_H **CDTM0_DTM4_0_N**: Inverted dead-time output of ATOM0_0 02_H **CDTM1_DTM4_0**: Dead-time output of ATOM1_0 03_H **CDTM2_DTM4_0_N**: Inverted dead-time output of ATOM2_0 04_H **CDTM0_DTM0_0_N**: Inverted dead-time output of TOM0_0 05_H **CDTM1_DTM0_0**: Dead-time output of TOM1_0 06_H **CDTM2_DTM0_0_N**: Inverted dead-time output of TOM2_0 07_H **Reserved**: Reserved, do not use … 15_H **Reserved**: Reserved, do not use 16_H **CDTM2_DTM0_1**: Dead-time output of TOM2_1 17_H **CDTM1_DTM0_1**: Dead-time output of TOM1_1 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=2)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT194 (P30.4) 01_H **CDTM0_DTM4_1**: Dead-time output of ATOM0_1 02_H **CDTM1_DTM4_1_N**: Inverted dead-time output of ATOM1_1 03_H **CDTM2_DTM4_1**: Dead-time output of ATOM2_1 04_H **CDTM0_DTM0_1**: Dead-time output of TOM0_1 05_H **CDTM1_DTM0_1_N**: Inverted dead-time output of TOM1_1 06_H **CDTM2_DTM0_1**: Dead-time output of TOM2_1 07_H **Reserved**: Reserved, do not use … 15_H **Reserved**: Reserved, do not use 16_H **CDTM2_DTM0_2**: Dead-time output of TOM2_2 17_H **CDTM1_DTM0_2**: Dead-time output of TOM1_2 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
SELx (x=3)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT195 (P30.5) 01_H **CDTM0_DTM4_1_N**: Inverted dead-time output of ATOM0_1 02_H **CDTM1_DTM4_1**: Dead-time output of ATOM1_1 03_H **CDTM2_DTM4_1_N**: Inverted dead-time output of ATOM2_1 04_H **CDTM0_DTM0_1_N**: Inverted dead-time output of TOM0_1 05_H **CDTM1_DTM0_1**: Dead-time output of TOM1_1 06_H **CDTM2_DTM0_1_N**: Inverted dead-time output of TOM2_1 07_H **Reserved**: Reserved, do not use … 15_H **Reserved**: Reserved, do not use 16_H **CDTM2_DTM0_3**: Dead-time output of TOM2_3 17_H **CDTM1_DTM0_3**: Dead-time output of TOM1_3 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=4)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT196 (P30.6) 01_H **CDTM0_DTM4_2**: Dead-time output of ATOM0_2 02_H **CDTM1_DTM4_2_N**: Inverted dead-time output of ATOM1_2 03_H **CDTM2_DTM4_2**: Dead-time output of ATOM2_2 04_H **CDTM0_DTM0_2**: Dead-time output of TOM0_2 05_H **CDTM1_DTM0_2_N**: Inverted dead-time output of TOM1_2 06_H **CDTM2_DTM0_2**: Dead-time output of TOM2_2 07_H **Reserved**: Reserved, do not use … 15_H **Reserved**: Reserved, do not use 16_H **CDTM2_DTM1_0**: Dead-time output of TOM2_4 17_H **CDTM1_DTM1_0**: Dead-time output of TOM1_4 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
SELx (x=5)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT197 (P30.7) 01_H **CDTM0_DTM4_2_N**: Inverted dead-time output of ATOM0_2 02_H **CDTM1_DTM4_2**: Dead-time output of ATOM1_2 03_H **CDTM2_DTM4_2_N**: Inverted dead-time output of ATOM2_2 04_H **CDTM0_DTM0_2_N**: Inverted dead-time output of TOM0_2 05_H **CDTM1_DTM0_2**: Dead-time output of TOM1_2 06_H **CDTM2_DTM0_2_N**: Inverted dead-time output of TOM2_2 07_H **Reserved**: Reserved, do not use … 15_H **Reserved**: Reserved, do not use 16_H **CDTM2_DTM1_1**: Dead-time output of TOM2_5 17_H **CDTM1_DTM1_1**: Dead-time output of TOM1_5 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	15, 31	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.43 eGTM to Ports signal TOUT output selection register n (n=33)

For a description of this register, see 7579.

EGTM_TOUTSELn (n=33) Offset address: 061100_H+n*4

eGTM to Ports signal TOUT output selection register n Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	SEL5					SEL4					SEL3				
r	rw					rw					rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	SEL2					SEL1					SEL0				
r	rw					rw					rw				

Field	Bits	Type	Description
SELx (x=0)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT198 (P30.8) 01_H **CDTM0_DTM4_3**: Dead-time output of ATOM0_3 02_H **CDTM1_DTM4_3_N**: Inverted dead-time output of ATOM1_3 03_H **CDTM2_DTM4_3**: Dead-time output of ATOM2_3 04_H **CDTM0_DTM0_3**: Dead-time output of TOM0_3 05_H **CDTM1_DTM0_3_N**: Inverted dead-time output of TOM1_3 06_H **CDTM2_DTM0_3**: Dead-time output of TOM2_3 07_H **Reserved**: Reserved, do not use … 15_H **Reserved**: Reserved, do not use 16_H **CDTM2_DTM1_2**: Dead-time output of TOM2_6 17_H **CDTM1_DTM1_2**: Dead-time output of TOM1_6 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
SELx (x=1)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT199 (P30.9) 01_H **CDTM0_DTM4_3_N**: Inverted dead-time output of ATOM0_3 02_H **CDTM1_DTM4_3**: Dead-time output of ATOM1_3 03_H **CDTM2_DTM4_3_N**: Inverted dead-time output of ATOM2_3 04_H **CDTM0_DTM0_3_N**: Inverted dead-time output of TOM0_3 05_H **CDTM1_DTM0_3**: Dead-time output of TOM1_3 06_H **CDTM2_DTM0_3_N**: Inverted dead-time output of TOM2_3 07_H **Reserved**: Reserved, do not use … 15_H **Reserved**: Reserved, do not use 16_H **CDTM2_DTM1_3**: Dead-time output of TOM2_7 17_H **CDTM1_DTM1_3**: Dead-time output of TOM1_7 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=2)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT200 (P30.10) 01_H **CDTM0_DTM5_0**: Dead-time output of ATOM0_4 02_H **CDTM1_DTM5_0_N**: Inverted dead-time output of ATOM1_4 03_H **CDTM2_DTM5_0**: Dead-time output of ATOM2_4 04_H **CDTM0_DTM1_0**: Dead-time output of TOM0_4 05_H **CDTM1_DTM1_0_N**: Inverted dead-time output of TOM1_4 06_H **CDTM2_DTM1_0**: Dead-time output of TOM2_4 07_H **Reserved**: Reserved, do not use … 15_H **Reserved**: Reserved, do not use 16_H **CDTM2_DTM2_0**: Dead-time output of TOM2_8 17_H **CDTM1_DTM2_0**: Dead-time output of TOM1_8 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
SELx (x=3)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT201 (P30.11) 01_H **CDTM0_DTM5_0_N**: Inverted dead-time output of ATOM0_4 02_H **CDTM1_DTM5_0**: Dead-time output of ATOM1_4 03_H **CDTM2_DTM5_0_N**: Inverted dead-time output of ATOM2_4 04_H **CDTM0_DTM1_0_N**: Inverted dead-time output of TOM0_4 05_H **CDTM1_DTM1_0**: Dead-time output of TOM1_4 06_H **CDTM2_DTM1_0_N**: Inverted dead-time output of TOM2_4 07_H **Reserved**: Reserved, do not use … 15_H **Reserved**: Reserved, do not use 16_H **CDTM2_DTM2_1**: Dead-time output of TOM2_9 17_H **CDTM1_DTM2_1**: Dead-time output of TOM1_9 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=4)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT202 (P30.12) 01_H **CDTM0_DTM5_1**: Dead-time output of ATOM0_5 02_H **CDTM1_DTM5_1_N**: Inverted dead-time output of ATOM1_5 03_H **CDTM2_DTM5_1**: Dead-time output of ATOM2_5 04_H **CDTM0_DTM1_1**: Dead-time output of TOM0_5 05_H **CDTM1_DTM1_1_N**: Inverted dead-time output of TOM1_5 06_H **CDTM2_DTM1_1**: Dead-time output of TOM2_5 07_H **Reserved**: Reserved, do not use … 15_H **Reserved**: Reserved, do not use 16_H **CDTM2_DTM2_2**: Dead-time output of TOM2_10 17_H **CDTM1_DTM2_2**: Dead-time output of TOM1_10 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
SELx (x=5)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT203 (P30.13) 01_H **CDTM0_DTM5_1_N**: Inverted dead-time output of ATOM0_5 02_H **CDTM1_DTM5_1**: Dead-time output of ATOM1_5 03_H **CDTM2_DTM5_1_N**: Inverted dead-time output of ATOM2_5 04_H **CDTM0_DTM1_1_N**: Inverted dead-time output of TOM0_5 05_H **CDTM1_DTM1_1**: Dead-time output of TOM1_5 06_H **CDTM2_DTM1_1_N**: Inverted dead-time output of TOM2_5 07_H **Reserved**: Reserved, do not use … 15_H **Reserved**: Reserved, do not use 16_H **CDTM2_DTM2_3**: Dead-time output of TOM2_11 17_H **CDTM1_DTM2_3**: Dead-time output of TOM1_11 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	15, 31	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.44 eGTM to Ports signal TOUT output selection register n (n=34)

For a description of this register, see 7579.

EGTM_TOUTSELn (n=34) Offset address: 061100_H+n*4

eGTM to Ports signal TOUT output selection register n Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	SEL5					SEL4					SEL3				
r	rw					rw					rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	SEL2					SEL1					SEL0				
r	rw					rw					rw				

Field	Bits	Type	Description
SELx (x=0)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT204 (P30.14) 01_H **CDTM0_DTM5_2**: Dead-time output of ATOM0_6 02_H **CDTM1_DTM5_2_N**: Inverted dead-time output of ATOM1_6 03_H **CDTM2_DTM5_2**: Dead-time output of ATOM2_6 04_H **CDTM0_DTM1_2**: Dead-time output of TOM0_6 05_H **CDTM1_DTM1_2_N**: Inverted dead-time output of TOM1_6 06_H **CDTM2_DTM1_2**: Dead-time output of TOM2_6 07_H **Reserved**: Reserved, do not use … 15_H **Reserved**: Reserved, do not use 16_H **CDTM2_DTM3_0**: Dead-time output of TOM2_12 17_H **CDTM1_DTM3_0**: Dead-time output of TOM1_12 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
SELx (x=1)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT205 (P30.15) 01_H **CDTM0_DTM5_2_N**: Inverted dead-time output of ATOM0_6 02_H **CDTM1_DTM5_2**: Dead-time output of ATOM1_6 03_H **CDTM2_DTM5_2_N**: Inverted dead-time output of ATOM2_6 04_H **CDTM0_DTM1_2_N**: Inverted dead-time output of TOM0_6 05_H **CDTM1_DTM1_2**: Dead-time output of TOM1_6 06_H **CDTM2_DTM1_2_N**: Inverted dead-time output of TOM2_6 07_H **Reserved**: Reserved, do not use … 15_H **Reserved**: Reserved, do not use 16_H **CDTM2_DTM3_1**: Dead-time output of TOM2_13 17_H **CDTM1_DTM3_1**: Dead-time output of TOM1_13 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=2)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT206 (P25.0) 01_H **CDTM0_DTM5_3**: Dead-time output of ATOM0_7 02_H **CDTM1_DTM5_3_N**: Inverted dead-time output of ATOM1_7 03_H **CDTM2_DTM5_3**: Dead-time output of ATOM2_7 04_H **CDTM0_DTM1_3**: Dead-time output of TOM0_7 05_H **CDTM1_DTM1_3_N**: Inverted dead-time output of TOM1_7 06_H **CDTM2_DTM1_3**: Dead-time output of TOM2_7 07_H **Reserved**: Reserved, do not use … 15_H **Reserved**: Reserved, do not use 16_H **CDTM2_DTM3_2**: Dead-time output of TOM2_14 17_H **CDTM1_DTM3_2**: Dead-time output of TOM1_14 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
SELx (x=3)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT207 (P25.1) 01_H **CDTM0_DTM5_3_N**: Inverted dead-time output of ATOM0_7 02_H **CDTM1_DTM5_3**: Dead-time output of ATOM1_7 03_H **CDTM2_DTM5_3_N**: Inverted dead-time output of ATOM2_7 04_H **CDTM0_DTM1_3_N**: Inverted dead-time output of TOM0_7 05_H **CDTM1_DTM1_3**: Dead-time output of TOM1_7 06_H **CDTM2_DTM1_3_N**: Inverted dead-time output of TOM2_7 07_H **Reserved**: Reserved, do not use … 16_H **Reserved**: Reserved, do not use 17_H **CDTM1_DTM2_2**: Dead-time output of TOM1_10 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=4)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT208 (P25.2) 01_H **CDTM0_DTM4_0**: Dead-time output of ATOM0_0 02_H **CDTM1_DTM4_0_N**: Inverted dead-time output of ATOM1_0 03_H **CDTM2_DTM4_0**: Dead-time output of ATOM2_0 04_H **CDTM0_DTM2_0**: Dead-time output of TOM0_8 05_H **CDTM1_DTM2_0_N**: Inverted dead-time output of TOM1_8 06_H **CDTM2_DTM2_0**: Dead-time output of TOM2_8 07_H **Reserved**: Reserved, do not use … 16_H **Reserved**: Reserved, do not use 17_H **CDTM1_DTM2_1**: Dead-time output of TOM1_9 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
SELx (x=5)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT209 (P25.3) 01_H **CDTM0_DTM4_0_N**: Inverted dead-time output of ATOM0_0 02_H **CDTM1_DTM4_0**: Dead-time output of ATOM1_0 03_H **CDTM2_DTM4_0_N**: Inverted dead-time output of ATOM2_0 04_H **CDTM0_DTM2_0_N**: Inverted dead-time output of TOM0_8 05_H **CDTM1_DTM2_0**: Dead-time output of TOM1_8 06_H **CDTM2_DTM2_0_N**: Inverted dead-time output of TOM2_8 07_H **Reserved**: Reserved, do not use … 16_H **Reserved**: Reserved, do not use 17_H **CDTM1_DTM2_0**: Dead-time output of TOM1_8 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	15, 31	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.45 eGTM to Ports signal TOUT output selection register n (n=35)

For a description of this register, see 7579.

EGTM_TOUTSELn (n=35) Offset address: 061100_H+n*4

eGTM to Ports signal TOUT output selection register n Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	SEL5					SEL4					SEL3				
r	rw					rw					rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	SEL2					SEL1					SEL0				
r	rw					rw					rw				

Field	Bits	Type	Description
SELx (x=0)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT210 (P25.4) 01_H **CDTM0_DTM4_1**: Dead-time output of ATOM0_1 02_H **CDTM1_DTM4_1_N**: Inverted dead-time output of ATOM1_1 03_H **CDTM2_DTM4_1**: Dead-time output of ATOM2_1 04_H **CDTM0_DTM2_1**: Dead-time output of TOM0_9 05_H **CDTM1_DTM2_1_N**: Inverted dead-time output of TOM1_9 06_H **CDTM2_DTM2_1**: Dead-time output of TOM2_9 07_H **Reserved**: Reserved, do not use … 16_H **Reserved**: Reserved, do not use 17_H **CDTM1_DTM1_3**: Dead-time output of TOM1_7 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
SELx (x=1)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT211 (P25.5) 01_H **CDTM0_DTM4_1_N**: Inverted dead-time output of ATOM0_1 02_H **CDTM1_DTM4_1**: Dead-time output of ATOM1_1 03_H **CDTM2_DTM4_1_N**: Inverted dead-time output of ATOM2_1 04_H **CDTM0_DTM2_1_N**: Inverted dead-time output of TOM0_9 05_H **CDTM1_DTM2_1**: Dead-time output of TOM1_9 06_H **CDTM2_DTM2_1_N**: Inverted dead-time output of TOM2_9 07_H **Reserved**: Reserved, do not use … 16_H **Reserved**: Reserved, do not use 17_H **CDTM1_DTM0_0**: Dead-time output of TOM1_0 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=2)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT212 (P25.6, P26.0) 01_H **CDTM0_DTM4_2**: Dead-time output of ATOM0_2 02_H **CDTM1_DTM4_2_N**: Inverted dead-time output of ATOM1_2 03_H **CDTM2_DTM4_2**: Dead-time output of ATOM2_2 04_H **CDTM0_DTM2_2**: Dead-time output of TOM0_10 05_H **CDTM1_DTM2_2_N**: Inverted dead-time output of TOM1_10 06_H **CDTM2_DTM2_2**: Dead-time output of TOM2_10 07_H **Reserved**: Reserved, do not use … 16_H **Reserved**: Reserved, do not use 17_H **CDTM1_DTM0_1**: Dead-time output of TOM1_1 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
SELx (x=3)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT213 (P25.7) 01_H **CDTM0_DTM4_2_N**: Inverted dead-time output of ATOM0_2 02_H **CDTM1_DTM4_2**: Dead-time output of ATOM1_2 03_H **CDTM2_DTM4_2_N**: Inverted dead-time output of ATOM2_2 04_H **CDTM0_DTM2_2_N**: Inverted dead-time output of TOM0_10 05_H **CDTM1_DTM2_2**: Dead-time output of TOM1_10 06_H **CDTM2_DTM2_2_N**: Inverted dead-time output of TOM2_10 07_H **Reserved**: Reserved, do not use … 16_H **Reserved**: Reserved, do not use 17_H **CDTM1_DTM1_0**: Dead-time output of TOM1_4 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=4)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT214 (P25.8) 01_H **CDTM0_DTM4_3**: Dead-time output of ATOM0_3 02_H **CDTM1_DTM4_3_N**: Inverted dead-time output of ATOM1_3 03_H **CDTM2_DTM4_3**: Dead-time output of ATOM2_3 04_H **CDTM0_DTM2_3**: Dead-time output of TOM0_11 05_H **CDTM1_DTM2_3_N**: Inverted dead-time output of TOM1_11 06_H **CDTM2_DTM2_3**: Dead-time output of TOM2_11 07_H **Reserved**: Reserved, do not use … 16_H **Reserved**: Reserved, do not use 17_H **CDTM1_DTM0_3**: Dead-time output of TOM1_3 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
SELx (x=5)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT215 (P25.9) 01_H **CDTM0_DTM4_3_N**: Inverted dead-time output of ATOM0_3 02_H **CDTM1_DTM4_3**: Dead-time output of ATOM1_3 03_H **CDTM2_DTM4_3_N**: Inverted dead-time output of ATOM2_3 04_H **CDTM0_DTM2_3_N**: Inverted dead-time output of TOM0_11 05_H **CDTM1_DTM2_3**: Dead-time output of TOM1_11 06_H **CDTM2_DTM2_3_N**: Inverted dead-time output of TOM2_11 07_H **Reserved**: Reserved, do not use … 16_H **Reserved**: Reserved, do not use 17_H **CDTM1_DTM3_3**: Dead-time output of TOM1_15 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	15, 31	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.46 eGTM to Ports signal TOUT output selection register n (n=36)

For a description of this register, see 7579.

EGTM_TOUTSELn (n=36) Offset address: 061100_H+n*4

eGTM to Ports signal TOUT output selection register n Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	SEL5					SEL4					SEL3				
r	rw					rw					rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	SEL2					SEL1					SEL0				
r	rw					rw					rw				

Field	Bits	Type	Description
SELx (x=0)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT216 (P25.10) 01_H **CDTM0_DTM5_0**: Dead-time output of ATOM0_4 02_H **CDTM1_DTM5_0_N**: Inverted dead-time output of ATOM1_4 03_H **CDTM2_DTM5_0**: Dead-time output of ATOM2_4 04_H **CDTM0_DTM3_0**: Dead-time output of TOM0_12 05_H **CDTM1_DTM3_0_N**: Inverted dead-time output of TOM1_12 06_H **CDTM2_DTM3_0**: Dead-time output of TOM2_12 07_H **Reserved**: Reserved, do not use … 16_H **Reserved**: Reserved, do not use 17_H **CDTM1_DTM3_2**: Dead-time output of TOM1_14 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
SELx (x=1)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT217 (P25.11) 01_H **CDTM0_DTM5_0_N**: Inverted dead-time output of ATOM0_4 02_H **CDTM1_DTM5_0**: Dead-time output of ATOM1_4 03_H **CDTM2_DTM5_0_N**: Inverted dead-time output of ATOM2_4 04_H **CDTM0_DTM3_0_N**: Inverted dead-time output of TOM0_12 05_H **CDTM1_DTM3_0**: Dead-time output of TOM1_12 06_H **CDTM2_DTM3_0_N**: Inverted dead-time output of TOM2_12 07_H **Reserved**: Reserved, do not use … 15_H **Reserved**: Reserved, do not use 16_H **CDTM2_DTM0_3**: Dead-time output of TOM2_3 17_H **CDTM1_DTM0_3**: Dead-time output of TOM1_3 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=2)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT218 (P25.12) 01_H **CDTM0_DTM5_1**: Dead-time output of ATOM0_5 02_H **CDTM1_DTM5_1_N**: Inverted dead-time output of ATOM1_5 03_H **CDTM2_DTM5_1**: Dead-time output of ATOM2_5 04_H **CDTM0_DTM3_1**: Dead-time output of TOM0_13 05_H **CDTM1_DTM3_1_N**: Inverted dead-time output of TOM1_13 06_H **CDTM2_DTM3_1**: Dead-time output of TOM2_13 07_H **Reserved**: Reserved, do not use … 15_H **Reserved**: Reserved, do not use 16_H **CDTM2_DTM1_0**: Dead-time output of TOM2_4 17_H **CDTM1_DTM1_0**: Dead-time output of TOM1_4 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
SELx (x=3)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT219 (P25.13) 01_H **CDTM0_DTM5_1_N**: Inverted dead-time output of ATOM0_5 02_H **CDTM1_DTM5_1**: Dead-time output of ATOM1_5 03_H **CDTM2_DTM5_1_N**: Inverted dead-time output of ATOM2_5 04_H **CDTM0_DTM3_1_N**: Inverted dead-time output of TOM0_13 05_H **CDTM1_DTM3_1**: Dead-time output of TOM1_13 06_H **CDTM2_DTM3_1_N**: Inverted dead-time output of TOM2_13 07_H **Reserved**: Reserved, do not use … 15_H **Reserved**: Reserved, do not use 16_H **CDTM2_DTM1_1**: Dead-time output of TOM2_5 17_H **CDTM1_DTM1_1**: Dead-time output of TOM1_5 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=4)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT220 (P25.14) 01_H **CDTM0_DTM5_2**: Dead-time output of ATOM0_6 02_H **CDTM1_DTM5_2_N**: Inverted dead-time output of ATOM1_6 03_H **CDTM2_DTM5_2**: Dead-time output of ATOM2_6 04_H **CDTM0_DTM3_2**: Dead-time output of TOM0_14 05_H **CDTM1_DTM3_2_N**: Inverted dead-time output of TOM1_14 06_H **CDTM2_DTM3_2**: Dead-time output of TOM2_14 07_H **Reserved**: Reserved, do not use … 15_H **Reserved**: Reserved, do not use 16_H **CDTM2_DTM1_2**: Dead-time output of TOM2_6 17_H **CDTM1_DTM1_2**: Dead-time output of TOM1_6 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
SELx (x=5)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT221 (P25.15) 01_H **CDTM0_DTM5_2_N**: Inverted dead-time output of ATOM0_6 02_H **CDTM1_DTM5_2**: Dead-time output of ATOM1_6 03_H **CDTM2_DTM5_2_N**: Inverted dead-time output of ATOM2_6 04_H **CDTM0_DTM3_2_N**: Inverted dead-time output of TOM0_14 05_H **CDTM1_DTM3_2**: Dead-time output of TOM1_14 06_H **CDTM2_DTM3_2_N**: Inverted dead-time output of TOM2_14 07_H **Reserved**: Reserved, do not use … 15_H **Reserved**: Reserved, do not use 16_H **CDTM2_DTM1_3**: Dead-time output of TOM2_7 17_H **CDTM1_DTM1_3**: Dead-time output of TOM1_7 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	15, 31	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.47 eGTM to Ports signal TOUT output selection register n (n=37)

For a description of this register, see 7579.

EGTM_TOUTSELn (n=37) Offset address: 061100_H+n*4

eGTM to Ports signal TOUT output selection register n Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	SEL5					SEL4					SEL3				
r	rw					rw					rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	SEL2					SEL1					SEL0				
r	rw					rw					rw				

Field	Bits	Type	Description
SELx (x=0)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT222 (P24.0) 01_H **CDTM0_DTM5_3**: Dead-time output of ATOM0_7 02_H **CDTM1_DTM5_3_N**: Inverted dead-time output of ATOM1_7 03_H **CDTM2_DTM5_3**: Dead-time output of ATOM2_7 04_H **CDTM0_DTM3_3**: Dead-time output of TOM0_15 05_H **CDTM1_DTM3_3_N**: Inverted dead-time output of TOM1_15 06_H **CDTM2_DTM3_3**: Dead-time output of TOM2_15 07_H **Reserved**: Reserved, do not use … 15_H **Reserved**: Reserved, do not use 16_H **CDTM2_DTM2_0**: Dead-time output of TOM2_8 17_H **CDTM1_DTM2_0**: Dead-time output of TOM1_8 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
SELx (x=1)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT223 (P24.1) 01_H **CDTM0_DTM5_3_N**: Inverted dead-time output of ATOM0_7 02_H **CDTM1_DTM5_3**: Dead-time output of ATOM1_7 03_H **CDTM2_DTM5_3_N**: Inverted dead-time output of ATOM2_7 04_H **CDTM0_DTM3_3_N**: Inverted dead-time output of TOM0_15 05_H **CDTM1_DTM3_3**: Dead-time output of TOM1_15 06_H **CDTM2_DTM3_3_N**: Inverted dead-time output of TOM2_15 07_H **Reserved**: Reserved, do not use … 15_H **Reserved**: Reserved, do not use 16_H **CDTM2_DTM2_1**: Dead-time output of TOM2_9 17_H **CDTM1_DTM2_1**: Dead-time output of TOM1_9 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=2)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT224 (P24.2) 01_H **CDTM0_DTM4_0**: Dead-time output of ATOM0_0 02_H **CDTM1_DTM4_0_N**: Inverted dead-time output of ATOM1_0 03_H **CDTM2_DTM4_0**: Dead-time output of ATOM2_0 04_H **CDTM0_DTM0_0**: Dead-time output of TOM0_0 05_H **CDTM1_DTM0_0_N**: Inverted dead-time output of TOM1_0 06_H **CDTM2_DTM0_0**: Dead-time output of TOM2_0 07_H **Reserved**: Reserved, do not use … 15_H **Reserved**: Reserved, do not use 16_H **CDTM2_DTM2_2**: Dead-time output of TOM2_10 17_H **CDTM1_DTM2_2**: Dead-time output of TOM1_10 18_H **Reserved**: Reserved, do not use 19_H **Reserved**: Reserved, do not use $1A_H$ **CDTM0_DTM4_0_N**: Inverted dead-time output of ATOM0_0 $1B_H$ **CDTM1_DTM4_0_N**: Inverted dead-time output of ATOM1_0 $1C_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
SELx (x=3)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT225 (P24.3) 01_H **CDTM0_DTM4_0_N**: Inverted dead-time output of ATOM0_0 02_H **CDTM1_DTM4_0**: Dead-time output of ATOM1_0 03_H **CDTM2_DTM4_0_N**: Inverted dead-time output of ATOM2_0 04_H **CDTM0_DTM0_0_N**: Inverted dead-time output of TOM0_0 05_H **CDTM1_DTM0_0**: Dead-time output of TOM1_0 06_H **CDTM2_DTM0_0_N**: Inverted dead-time output of TOM2_0 07_H **Reserved**: Reserved, do not use … 15_H **Reserved**: Reserved, do not use 16_H **CDTM2_DTM2_3**: Dead-time output of TOM2_11 17_H **CDTM1_DTM2_3**: Dead-time output of TOM1_11 18_H **Reserved**: Reserved, do not use 19_H **Reserved**: Reserved, do not use $1A_H$ **CDTM0_DTM4_1_N**: Inverted dead-time output of ATOM0_1 $1B_H$ **CDTM1_DTM4_1_N**: Inverted dead-time output of ATOM1_1 $1C_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=4)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT226 (P24.4) 01_H **CDTM0_DTM4_1**: Dead-time output of ATOM0_1 02_H **CDTM1_DTM4_1_N**: Inverted dead-time output of ATOM1_1 03_H **CDTM2_DTM4_1**: Dead-time output of ATOM2_1 04_H **CDTM0_DTM0_1**: Dead-time output of TOM0_1 05_H **CDTM1_DTM0_1_N**: Inverted dead-time output of TOM1_1 06_H **CDTM2_DTM0_1**: Dead-time output of TOM2_1 07_H **Reserved**: Reserved, do not use … 15_H **Reserved**: Reserved, do not use 16_H **CDTM2_DTM3_0**: Dead-time output of TOM2_12 17_H **CDTM1_DTM3_0**: Dead-time output of TOM1_12 18_H **Reserved**: Reserved, do not use 19_H **Reserved**: Reserved, do not use $1A_H$ **CDTM0_DTM4_2_N**: Inverted dead-time output of ATOM0_2 $1B_H$ **CDTM1_DTM4_2_N**: Inverted dead-time output of ATOM1_2 $1C_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
SELx (x=5)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT227 (P24.5) 01_H **CDTM0_DTM4_1_N**: Inverted dead-time output of ATOM0_1 02_H **CDTM1_DTM4_1**: Dead-time output of ATOM1_1 03_H **CDTM2_DTM4_1_N**: Inverted dead-time output of ATOM2_1 04_H **CDTM0_DTM0_1_N**: Inverted dead-time output of TOM0_1 05_H **CDTM1_DTM0_1**: Dead-time output of TOM1_1 06_H **CDTM2_DTM0_1_N**: Inverted dead-time output of TOM2_1 07_H **Reserved**: Reserved, do not use … 15_H **Reserved**: Reserved, do not use 16_H **CDTM2_DTM3_1**: Dead-time output of TOM2_13 17_H **CDTM1_DTM3_1**: Dead-time output of TOM1_13 18_H **Reserved**: Reserved, do not use 19_H **Reserved**: Reserved, do not use $1A_H$ **CDTM0_DTM4_3_N**: Inverted dead-time output of ATOM0_3 $1B_H$ **CDTM1_DTM4_3_N**: Inverted dead-time output of ATOM1_3 $1C_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	15, 31	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.48 eGTM to Ports signal TOUT output selection register n (n=38)

For a description of this register, see 7579.

EGTM_TOUTSELn (n=38) Offset address: 061100_H+n*4

eGTM to Ports signal TOUT output selection register n Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	SEL5					SEL4					SEL3				
r	rw					rw					rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	SEL2					SEL1					SEL0				
r	rw					rw					rw				

Field	Bits	Type	Description
SELx (x=0)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT228 (P24.6) 01_H **CDTM0_DTM4_2**: Dead-time output of ATOM0_2 02_H **CDTM1_DTM4_2_N**: Inverted dead-time output of ATOM1_2 03_H **CDTM2_DTM4_2**: Dead-time output of ATOM2_2 04_H **CDTM0_DTM0_2**: Dead-time output of TOM0_2 05_H **CDTM1_DTM0_2_N**: Inverted dead-time output of TOM1_2 06_H **CDTM2_DTM0_2**: Dead-time output of TOM2_2 07_H **Reserved**: Reserved, do not use … 15_H **Reserved**: Reserved, do not use 16_H **CDTM2_DTM3_2**: Dead-time output of TOM2_14 17_H **CDTM1_DTM3_2**: Dead-time output of TOM1_14 18_H **Reserved**: Reserved, do not use 19_H **Reserved**: Reserved, do not use $1A_H$ **CDTM0_DTM5_0_N**: Inverted dead-time output of ATOM0_4 $1B_H$ **CDTM1_DTM5_0_N**: Inverted dead-time output of ATOM1_4 $1C_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=1)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT229 (P24.7) 01_H **CDTM0_DTM4_2_N**: Inverted dead-time output of ATOM0_2 02_H **CDTM1_DTM4_2**: Dead-time output of ATOM1_2 03_H **CDTM2_DTM4_2_N**: Inverted dead-time output of ATOM2_2 04_H **CDTM0_DTM0_2_N**: Inverted dead-time output of TOM0_2 05_H **CDTM1_DTM0_2**: Dead-time output of TOM1_2 06_H **CDTM2_DTM0_2_N**: Inverted dead-time output of TOM2_2 07_H **Reserved**: Reserved, do not use … 15_H **Reserved**: Reserved, do not use 16_H **CDTM2_DTM3_3**: Dead-time output of TOM2_15 17_H **CDTM1_DTM3_3**: Dead-time output of TOM1_15 18_H **Reserved**: Reserved, do not use 19_H **Reserved**: Reserved, do not use $1A_H$ **CDTM0_DTM5_1_N**: Inverted dead-time output of ATOM0_5 $1B_H$ **CDTM1_DTM5_1_N**: Inverted dead-time output of ATOM1_5 $1C_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
SELx (x=2)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT230 (P24.8) 01_H **CDTM0_DTM4_3**: Dead-time output of ATOM0_3 02_H **CDTM1_DTM4_3_N**: Inverted dead-time output of ATOM1_3 03_H **CDTM2_DTM4_3**: Dead-time output of ATOM2_3 04_H **CDTM0_DTM0_3**: Dead-time output of TOM0_3 05_H **CDTM1_DTM0_3_N**: Inverted dead-time output of TOM1_3 06_H **CDTM2_DTM0_3**: Dead-time output of TOM2_3 07_H **Reserved**: Reserved, do not use … 16_H **Reserved**: Reserved, do not use 17_H **CDTM1_DTM0_0**: Dead-time output of TOM1_0 18_H **Reserved**: Reserved, do not use 19_H **Reserved**: Reserved, do not use $1A_H$ **CDTM0_DTM5_2_N**: Inverted dead-time output of ATOM0_6 $1B_H$ **CDTM1_DTM5_2_N**: Inverted dead-time output of ATOM1_6 $1C_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=3)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT231 (P24.9) 01_H **CDTM0_DTM4_3_N**: Inverted dead-time output of ATOM0_3 02_H **CDTM1_DTM4_3**: Dead-time output of ATOM1_3 03_H **CDTM2_DTM4_3_N**: Inverted dead-time output of ATOM2_3 04_H **CDTM0_DTM0_3_N**: Inverted dead-time output of TOM0_3 05_H **CDTM1_DTM0_3**: Dead-time output of TOM1_3 06_H **CDTM2_DTM0_3_N**: Inverted dead-time output of TOM2_3 07_H **Reserved**: Reserved, do not use … 16_H **Reserved**: Reserved, do not use 17_H **CDTM1_DTM0_0**: Dead-time output of TOM1_0 18_H **Reserved**: Reserved, do not use 19_H **Reserved**: Reserved, do not use $1A_H$ **CDTM0_DTM5_3_N**: Inverted dead-time output of ATOM0_7 $1B_H$ **CDTM1_DTM5_3_N**: Inverted dead-time output of ATOM1_7 $1C_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
SELx (x=4)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT232 (P24.10) 01_H **CDTM0_DTM5_0**: Dead-time output of ATOM0_4 02_H **CDTM1_DTM5_0_N**: Inverted dead-time output of ATOM1_4 03_H **CDTM2_DTM5_0**: Dead-time output of ATOM2_4 04_H **CDTM0_DTM1_0**: Dead-time output of TOM0_4 05_H **CDTM1_DTM1_0_N**: Inverted dead-time output of TOM1_4 06_H **CDTM2_DTM1_0**: Dead-time output of TOM2_4 07_H **Reserved**: Reserved, do not use … 16_H **Reserved**: Reserved, do not use 17_H **CDTM1_DTM0_0**: Dead-time output of TOM1_0 18_H **Reserved**: Reserved, do not use 19_H **Reserved**: Reserved, do not use $1A_H$ **CDTM2_DTM4_0_N**: Inverted dead-time output of ATOM2_0 $1B_H$ **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=5)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT233 (P24.11) 01_H **CDTM0_DTM5_0_N**: Inverted dead-time output of ATOM0_4 02_H **CDTM1_DTM5_0**: Dead-time output of ATOM1_4 03_H **CDTM2_DTM5_0_N**: Inverted dead-time output of ATOM2_4 04_H **CDTM0_DTM1_0_N**: Inverted dead-time output of TOM0_4 05_H **CDTM1_DTM1_0**: Dead-time output of TOM1_4 06_H **CDTM2_DTM1_0_N**: Inverted dead-time output of TOM2_4 07_H **Reserved**: Reserved, do not use … 15_H **Reserved**: Reserved, do not use 16_H **CDTM1_DTM3_3**: Dead-time output of TOM1_15 17_H **CDTM1_DTM3_3**: Dead-time output of TOM1_15 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	15, 31	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.49 eGTM to Ports signal TOUT output selection register n (n=39)

For a description of this register, see 7579.

EGTM_TOUTSELn (n=39) Offset address: 061100_H+n*4

eGTM to Ports signal TOUT output selection register n Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	SEL5					SEL4					SEL3				
r	rw					rw					rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	SEL2					SEL1					SEL0				
r	rw					rw					rw				

Field	Bits	Type	Description
SELx (x=0)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT234 (P24.12) 01_H **CDTM0_DTM5_1**: Dead-time output of ATOM0_5 02_H **CDTM1_DTM5_1_N**: Inverted dead-time output of ATOM1_5 03_H **CDTM2_DTM5_1**: Dead-time output of ATOM2_5 04_H **CDTM0_DTM1_1**: Dead-time output of TOM0_5 05_H **CDTM1_DTM1_1_N**: Inverted dead-time output of TOM1_5 06_H **CDTM2_DTM1_1**: Dead-time output of TOM2_5 07_H **Reserved**: Reserved, do not use … 15_H **Reserved**: Reserved, do not use 16_H **CDTM2_DTM0_0**: Dead-time output of TOM2_0 17_H **CDTM0_DTM0_0**: Dead-time output of TOM0_0 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
SELx (x=1)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT235 (P24.13) 01_H **CDTM0_DTM5_1_N**: Inverted dead-time output of ATOM0_5 02_H **CDTM1_DTM5_1**: Dead-time output of ATOM1_5 03_H **CDTM2_DTM5_1_N**: Inverted dead-time output of ATOM2_5 04_H **CDTM0_DTM1_1_N**: Inverted dead-time output of TOM0_5 05_H **CDTM1_DTM1_1**: Dead-time output of TOM1_5 06_H **CDTM2_DTM1_1_N**: Inverted dead-time output of TOM2_5 07_H **Reserved**: Reserved, do not use … 15_H **Reserved**: Reserved, do not use 16_H **CDTM1_DTM1_0_N**: Inverted dead-time output of TOM1_4 17_H **CDTM2_DTM1_0**: Dead-time output of TOM2_4 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=2)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT236 (P24.14) 01_H **CDTM0_DTM5_2**: Dead-time output of ATOM0_6 02_H **CDTM1_DTM5_2_N**: Inverted dead-time output of ATOM1_6 03_H **CDTM2_DTM5_2**: Dead-time output of ATOM2_6 04_H **CDTM0_DTM1_2**: Dead-time output of TOM0_6 05_H **CDTM1_DTM1_2_N**: Inverted dead-time output of TOM1_6 06_H **CDTM2_DTM1_2**: Dead-time output of TOM2_6 07_H **Reserved**: Reserved, do not use … 15_H **Reserved**: Reserved, do not use 16_H **CDTM1_DTM0_1_N**: Inverted dead-time output of TOM1_1 17_H **CDTM2_DTM0_1**: Dead-time output of TOM2_1 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
SELx (x=3)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT237 (P24.15) 01_H **CDTM0_DTM5_2_N**: Inverted dead-time output of ATOM0_6 02_H **CDTM1_DTM5_2**: Dead-time output of ATOM1_6 03_H **CDTM2_DTM5_2_N**: Inverted dead-time output of ATOM2_6 04_H **CDTM0_DTM1_2_N**: Inverted dead-time output of TOM0_6 05_H **CDTM1_DTM1_2**: Dead-time output of TOM1_6 06_H **CDTM2_DTM1_2_N**: Inverted dead-time output of TOM2_6 07_H **Reserved**: Reserved, do not use … 15_H **Reserved**: Reserved, do not use 16_H **CDTM1_DTM0_2_N**: Inverted dead-time output of TOM1_2 17_H **CDTM2_DTM0_2**: Dead-time output of TOM2_2 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=4)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT238 (no pin) 01_H **CDTM0_DTM5_3**: Dead-time output of ATOM0_7 02_H **CDTM1_DTM5_3_N**: Inverted dead-time output of ATOM1_7 03_H **CDTM2_DTM5_3**: Dead-time output of ATOM2_7 04_H **CDTM0_DTM1_3**: Dead-time output of TOM0_7 05_H **CDTM1_DTM1_3_N**: Inverted dead-time output of TOM1_7 06_H **CDTM2_DTM1_3**: Dead-time output of TOM2_7 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM1_DTM0_0**: Dead-time output of TOM1_0 11_H **Reserved**: Reserved, do not use … 15_H **Reserved**: Reserved, do not use 16_H **CDTM1_DTM0_3_N**: Inverted dead-time output of TOM1_3 17_H **CDTM2_DTM0_3**: Dead-time output of TOM2_3 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
SELx (x=5)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT239 (no pin) 01_H **CDTM0_DTM5_3_N**: Inverted dead-time output of ATOM0_7 02_H **CDTM1_DTM5_3**: Dead-time output of ATOM1_7 03_H **CDTM2_DTM5_3_N**: Inverted dead-time output of ATOM2_7 04_H **CDTM0_DTM1_3_N**: Inverted dead-time output of TOM0_7 05_H **CDTM1_DTM1_3**: Dead-time output of TOM1_7 06_H **CDTM2_DTM1_3_N**: Inverted dead-time output of TOM2_7 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM1_DTM0_0**: Dead-time output of TOM1_0 11_H **Reserved**: Reserved, do not use … 15_H **Reserved**: Reserved, do not use 16_H **CDTM1_DTM1_0_N**: Inverted dead-time output of TOM1_4 17_H **CDTM2_DTM1_0**: Dead-time output of TOM2_4 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	15, 31	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.50 eGTM to Ports signal TOUT output selection register n (n=40)

For a description of this register, see 7579.

EGTM_TOUTSELn (n=40) Offset address: 061100_H+n*4

eGTM to Ports signal TOUT output selection register n Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	SEL5					SEL4					SEL3				
r	rw					rw					rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	SEL2					SEL1					SEL0				
r	rw					rw					rw				

Field	Bits	Type	Description
SELx (x=0)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT240 (no pin) 01_H **CDTM0_DTM4_0**: Dead-time output of ATOM0_0 02_H **CDTM1_DTM4_0_N**: Inverted dead-time output of ATOM1_0 03_H **CDTM2_DTM4_0**: Dead-time output of ATOM2_0 04_H **CDTM0_DTM2_0**: Dead-time output of TOM0_8 05_H **CDTM1_DTM2_0_N**: Inverted dead-time output of TOM1_8 06_H **CDTM2_DTM2_0**: Dead-time output of TOM2_8 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM1_DTM0_0**: Dead-time output of TOM1_0 11_H **Reserved**: Reserved, do not use … 15_H **Reserved**: Reserved, do not use 16_H **CDTM1_DTM1_1_N**: Inverted dead-time output of TOM1_5 17_H **CDTM2_DTM1_1**: Dead-time output of TOM2_5 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=1)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT241 (no pin) 01_H **CDTM0_DTM4_0_N**: Inverted dead-time output of ATOM0_0 02_H **CDTM1_DTM4_0**: Dead-time output of ATOM1_0 03_H **CDTM2_DTM4_0_N**: Inverted dead-time output of ATOM2_0 04_H **CDTM0_DTM2_0_N**: Inverted dead-time output of TOM0_8 05_H **CDTM1_DTM2_0**: Dead-time output of TOM1_8 06_H **CDTM2_DTM2_0_N**: Inverted dead-time output of TOM2_8 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM2_DTM0_0_N**: Inverted dead-time output of TOM2_0 11_H **Reserved**: Reserved, do not use … 15_H **Reserved**: Reserved, do not use 16_H **CDTM1_DTM1_2_N**: Inverted dead-time output of TOM1_6 17_H **CDTM2_DTM1_2**: Dead-time output of TOM2_6 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
SELx (x=2)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT242 (P15.10) 01_H **CDTM0_DTM4_1**: Dead-time output of ATOM0_1 02_H **CDTM1_DTM4_1_N**: Inverted dead-time output of ATOM1_1 03_H **CDTM2_DTM4_1**: Dead-time output of ATOM2_1 04_H **CDTM0_DTM2_1**: Dead-time output of TOM0_9 05_H **CDTM1_DTM2_1_N**: Inverted dead-time output of TOM1_9 06_H **CDTM2_DTM2_1**: Dead-time output of TOM2_9 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM1_DTM0_1**: Dead-time output of TOM1_1 11_H **Reserved**: Reserved, do not use … 15_H **Reserved**: Reserved, do not use 16_H **CDTM1_DTM1_2_N**: Inverted dead-time output of TOM1_6 17_H **CDTM2_DTM1_2**: Dead-time output of TOM2_6 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=3)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT243 (P15.11) 01_H **CDTM0_DTM4_1_N**: Inverted dead-time output of ATOM0_1 02_H **CDTM1_DTM4_1**: Dead-time output of ATOM1_1 03_H **CDTM2_DTM4_1_N**: Inverted dead-time output of ATOM2_1 04_H **CDTM0_DTM2_1_N**: Inverted dead-time output of TOM0_9 05_H **CDTM1_DTM2_1**: Dead-time output of TOM1_9 06_H **CDTM2_DTM2_1_N**: Inverted dead-time output of TOM2_9 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM1_DTM0_2**: Dead-time output of TOM1_2 11_H **Reserved**: Reserved, do not use … 15_H **Reserved**: Reserved, do not use 16_H **CDTM1_DTM1_3_N**: Inverted dead-time output of TOM1_7 17_H **CDTM2_DTM1_3**: Dead-time output of TOM2_7 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
SELx (x=4)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT244 (P15.12) 01_H **CDTM0_DTM4_2**: Dead-time output of ATOM0_2 02_H **CDTM1_DTM4_2_N**: Inverted dead-time output of ATOM1_2 03_H **CDTM2_DTM4_2**: Dead-time output of ATOM2_2 04_H **CDTM0_DTM2_2**: Dead-time output of TOM0_10 05_H **CDTM1_DTM2_2_N**: Inverted dead-time output of TOM1_10 06_H **CDTM2_DTM2_2**: Dead-time output of TOM2_10 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM1_DTM0_3**: Dead-time output of TOM1_3 11_H **Reserved**: Reserved, do not use … 15_H **Reserved**: Reserved, do not use 16_H **CDTM1_DTM1_0_N**: Inverted dead-time output of TOM1_4 17_H **CDTM2_DTM1_0**: Dead-time output of TOM2_4 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=5)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT245 (P15.13) 01_H **CDTM0_DTM4_2_N**: Inverted dead-time output of ATOM0_2 02_H **CDTM1_DTM4_2**: Dead-time output of ATOM1_2 03_H **CDTM2_DTM4_2_N**: Inverted dead-time output of ATOM2_2 04_H **CDTM0_DTM2_2_N**: Inverted dead-time output of TOM0_10 05_H **CDTM1_DTM2_2**: Dead-time output of TOM1_10 06_H **CDTM2_DTM2_2_N**: Inverted dead-time output of TOM2_10 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM1_DTM1_0**: Dead-time output of TOM1_4 11_H **Reserved**: Reserved, do not use … 13_H **Reserved**: Reserved, do not use 14_H **CDTM2_DTM0_1_N**: Inverted dead-time output of TOM2_1 15_H **Reserved**: Reserved, do not use 16_H **CDTM1_DTM0_1_N**: Inverted dead-time output of TOM1_1 17_H **CDTM2_DTM0_1**: Dead-time output of TOM2_1 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	15, 31	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.51 eGTM to Ports signal TOUT output selection register n (n=41)

For a description of this register, see 7579.

EGTM_TOUTSELn (n=41) Offset address: 061100_H+n*4

eGTM to Ports signal TOUT output selection register n Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	SEL5					SEL4					SEL3				
r	rw					rw					rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	SEL2					SEL1					SEL0				
r	rw					rw					rw				

Field	Bits	Type	Description
SELx (x=0)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT246 (P15.14) 01_H **CDTM0_DTM4_3**: Dead-time output of ATOM0_3 02_H **CDTM1_DTM4_3_N**: Inverted dead-time output of ATOM1_3 03_H **CDTM2_DTM4_3**: Dead-time output of ATOM2_3 04_H **CDTM0_DTM2_3**: Dead-time output of TOM0_11 05_H **CDTM1_DTM2_3_N**: Inverted dead-time output of TOM1_11 06_H **CDTM2_DTM2_3**: Dead-time output of TOM2_11 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM1_DTM1_1**: Dead-time output of TOM1_5 11_H **Reserved**: Reserved, do not use … 13_H **Reserved**: Reserved, do not use 14_H **CDTM2_DTM0_2_N**: Inverted dead-time output of TOM2_2 15_H **Reserved**: Reserved, do not use 16_H **CDTM1_DTM0_2_N**: Inverted dead-time output of TOM1_2 17_H **CDTM2_DTM0_2**: Dead-time output of TOM2_2 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
SELx (x=1)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT247 (P15.15) 01_H **CDTM0_DTM4_3_N**: Inverted dead-time output of ATOM0_3 02_H **CDTM1_DTM4_3**: Dead-time output of ATOM1_3 03_H **CDTM2_DTM4_3_N**: Inverted dead-time output of ATOM2_3 04_H **CDTM0_DTM2_3_N**: Inverted dead-time output of TOM0_11 05_H **CDTM1_DTM2_3**: Dead-time output of TOM1_11 06_H **CDTM2_DTM2_3_N**: Inverted dead-time output of TOM2_11 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM1_DTM1_2**: Dead-time output of TOM1_6 11_H **Reserved**: Reserved, do not use … 13_H **Reserved**: Reserved, do not use 14_H **CDTM2_DTM0_3_N**: Inverted dead-time output of TOM2_3 15_H **Reserved**: Reserved, do not use 16_H **CDTM1_DTM0_3_N**: Inverted dead-time output of TOM1_3 17_H **CDTM2_DTM0_3**: Dead-time output of TOM2_3 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=2)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT248 (P13.9) 01_H **CDTM0_DTM5_0**: Dead-time output of ATOM0_4 02_H **CDTM1_DTM5_0_N**: Inverted dead-time output of ATOM1_4 03_H **CDTM2_DTM5_0**: Dead-time output of ATOM2_4 04_H **CDTM0_DTM3_0**: Dead-time output of TOM0_12 05_H **CDTM1_DTM3_0_N**: Inverted dead-time output of TOM1_12 06_H **CDTM2_DTM3_0**: Dead-time output of TOM2_12 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM1_DTM1_3**: Dead-time output of TOM1_7 11_H **Reserved**: Reserved, do not use … 15_H **Reserved**: Reserved, do not use 16_H **CDTM1_DTM0_3_N**: Inverted dead-time output of TOM1_3 17_H **CDTM2_DTM0_3**: Dead-time output of TOM2_3 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
SELx (x=3)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT249 (P13.12) 01_H **CDTM0_DTM5_0_N**: Inverted dead-time output of ATOM0_4 02_H **CDTM1_DTM5_0**: Dead-time output of ATOM1_4 03_H **CDTM2_DTM5_0_N**: Inverted dead-time output of ATOM2_4 04_H **CDTM0_DTM3_0_N**: Inverted dead-time output of TOM0_12 05_H **CDTM1_DTM3_0**: Dead-time output of TOM1_12 06_H **CDTM2_DTM3_0_N**: Inverted dead-time output of TOM2_12 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM1_DTM2_0**: Dead-time output of TOM1_8 11_H **Reserved**: Reserved, do not use … 15_H **Reserved**: Reserved, do not use 16_H **CDTM1_DTM0_3_N**: Inverted dead-time output of TOM1_3 17_H **CDTM2_DTM0_3**: Dead-time output of TOM2_3 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=4)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT250 (P13.11) 01_H **CDTM0_DTM5_1**: Dead-time output of ATOM0_5 02_H **CDTM1_DTM5_1_N**: Inverted dead-time output of ATOM1_5 03_H **CDTM2_DTM5_1**: Dead-time output of ATOM2_5 04_H **CDTM0_DTM3_1**: Dead-time output of TOM0_13 05_H **CDTM1_DTM3_1_N**: Inverted dead-time output of TOM1_13 06_H **CDTM2_DTM3_1**: Dead-time output of TOM2_13 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM2_DTM0_0**: Dead-time output of TOM2_0 11_H **Reserved**: Reserved, do not use … 13_H **Reserved**: Reserved, do not use 14_H **CDTM2_DTM1_1_N**: Inverted dead-time output of TOM2_5 15_H **Reserved**: Reserved, do not use 16_H **CDTM1_DTM1_1_N**: Inverted dead-time output of TOM1_5 17_H **CDTM2_DTM1_1**: Dead-time output of TOM2_5 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
SELx (x=5)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT251 (P13.10) 01_H **CDTM0_DTM5_1_N**: Inverted dead-time output of ATOM0_5 02_H **CDTM1_DTM5_1**: Dead-time output of ATOM1_5 03_H **CDTM2_DTM5_1_N**: Inverted dead-time output of ATOM2_5 04_H **CDTM0_DTM3_1_N**: Inverted dead-time output of TOM0_13 05_H **CDTM1_DTM3_1**: Dead-time output of TOM1_13 06_H **CDTM2_DTM3_1_N**: Inverted dead-time output of TOM2_13 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM2_DTM0_1**: Dead-time output of TOM2_1 11_H **Reserved**: Reserved, do not use … 13_H **Reserved**: Reserved, do not use 14_H **CDTM2_DTM1_2_N**: Inverted dead-time output of TOM2_6 15_H **Reserved**: Reserved, do not use 16_H **CDTM1_DTM1_2_N**: Inverted dead-time output of TOM1_6 17_H **CDTM2_DTM1_2**: Dead-time output of TOM2_6 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
0	15, 31	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.52 eGTM to Ports signal TOUT output selection register n (n=42)

For a description of this register, see 7579.

EGTM_TOUTSELn (n=42) Offset address: 061100_H+n*4

eGTM to Ports signal TOUT output selection register n Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	SEL5					SEL4					SEL3				
r	rw					rw					rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	SEL2					SEL1					SEL0				
r	rw					rw					rw				

Field	Bits	Type	Description
SELx (x=0)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT252 (P13.14) 01_H **CDTM0_DTM5_2**: Dead-time output of ATOM0_6 02_H **CDTM1_DTM5_2_N**: Inverted dead-time output of ATOM1_6 03_H **CDTM2_DTM5_2**: Dead-time output of ATOM2_6 04_H **CDTM0_DTM3_2**: Dead-time output of TOM0_14 05_H **CDTM1_DTM3_2_N**: Inverted dead-time output of TOM1_14 06_H **CDTM2_DTM3_2**: Dead-time output of TOM2_14 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM2_DTM0_2**: Dead-time output of TOM2_2 11_H **Reserved**: Reserved, do not use … 13_H **Reserved**: Reserved, do not use 14_H **CDTM2_DTM1_3_N**: Inverted dead-time output of TOM2_7 15_H **Reserved**: Reserved, do not use 16_H **CDTM1_DTM1_3_N**: Inverted dead-time output of TOM1_7 17_H **CDTM2_DTM1_3**: Dead-time output of TOM2_7 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=1)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT253 (P13.4) 01_H **CDTM0_DTM5_2_N**: Inverted dead-time output of ATOM0_6 02_H **CDTM1_DTM5_2**: Dead-time output of ATOM1_6 03_H **CDTM2_DTM5_2_N**: Inverted dead-time output of ATOM2_6 04_H **CDTM0_DTM3_2_N**: Inverted dead-time output of TOM0_14 05_H **CDTM1_DTM3_2**: Dead-time output of TOM1_14 06_H **CDTM2_DTM3_2_N**: Inverted dead-time output of TOM2_14 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM2_DTM0_3**: Dead-time output of TOM2_3 11_H **Reserved**: Reserved, do not use … 13_H **Reserved**: Reserved, do not use 14_H **CDTM2_DTM1_0_N**: Inverted dead-time output of TOM2_4 15_H **Reserved**: Reserved, do not use 16_H **CDTM1_DTM1_0_N**: Inverted dead-time output of TOM1_4 17_H **CDTM2_DTM1_0**: Dead-time output of TOM2_4 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
SELx (x=2)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT254 (P13.5) 01_H **CDTM0_DTM5_3**: Dead-time output of ATOM0_7 02_H **CDTM1_DTM5_3_N**: Inverted dead-time output of ATOM1_7 03_H **CDTM2_DTM5_3**: Dead-time output of ATOM2_7 04_H **CDTM0_DTM3_3**: Dead-time output of TOM0_15 05_H **CDTM1_DTM3_3_N**: Inverted dead-time output of TOM1_15 06_H **CDTM2_DTM3_3**: Dead-time output of TOM2_15 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM2_DTM1_0**: Dead-time output of TOM2_4 11_H **Reserved**: Reserved, do not use … 13_H **Reserved**: Reserved, do not use 14_H **CDTM2_DTM0_1_N**: Inverted dead-time output of TOM2_1 15_H **Reserved**: Reserved, do not use 16_H **CDTM1_DTM0_1_N**: Inverted dead-time output of TOM1_1 17_H **CDTM2_DTM0_1**: Dead-time output of TOM2_1 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=3)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT255 (P13.6) 01_H **CDTM0_DTM5_3_N**: Inverted dead-time output of ATOM0_7 02_H **CDTM1_DTM5_3**: Dead-time output of ATOM1_7 03_H **CDTM2_DTM5_3_N**: Inverted dead-time output of ATOM2_7 04_H **CDTM0_DTM3_3_N**: Inverted dead-time output of TOM0_15 05_H **CDTM1_DTM3_3**: Dead-time output of TOM1_15 06_H **CDTM2_DTM3_3_N**: Inverted dead-time output of TOM2_15 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM2_DTM1_1**: Dead-time output of TOM2_5 11_H **Reserved**: Reserved, do not use … 13_H **Reserved**: Reserved, do not use 14_H **CDTM2_DTM0_2_N**: Inverted dead-time output of TOM2_2 15_H **Reserved**: Reserved, do not use 16_H **CDTM2_DTM1_3**: Dead-time output of TOM2_7 17_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
SELx (x=4)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT256 (P13.7) 01_H **CDTM0_DTM4_0**: Dead-time output of ATOM0_0 02_H **CDTM1_DTM4_0_N**: Inverted dead-time output of ATOM1_0 03_H **CDTM2_DTM4_0**: Dead-time output of ATOM2_0 04_H **CDTM0_DTM0_0**: Dead-time output of TOM0_0 05_H **CDTM1_DTM0_0_N**: Inverted dead-time output of TOM1_0 06_H **CDTM2_DTM0_0**: Dead-time output of TOM2_0 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM2_DTM1_2**: Dead-time output of TOM2_6 11_H **Reserved**: Reserved, do not use … 13_H **Reserved**: Reserved, do not use 14_H **CDTM2_DTM0_3_N**: Inverted dead-time output of TOM2_3 15_H **Reserved**: Reserved, do not use 16_H **CDTM2_DTM2_0**: Dead-time output of TOM2_8 17_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=5)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT257 (no pin) 01_H **CDTM0_DTM4_0_N**: Inverted dead-time output of ATOM0_0 02_H **CDTM1_DTM4_0**: Dead-time output of ATOM1_0 03_H **CDTM2_DTM4_0_N**: Inverted dead-time output of ATOM2_0 04_H **CDTM0_DTM0_0_N**: Inverted dead-time output of TOM0_0 05_H **CDTM1_DTM0_0**: Dead-time output of TOM1_0 06_H **CDTM2_DTM0_0_N**: Inverted dead-time output of TOM2_0 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM1_DTM1_3**: Dead-time output of TOM1_7 11_H **Reserved**: Reserved, do not use … 15_H **Reserved**: Reserved, do not use 16_H **CDTM2_DTM2_1**: Dead-time output of TOM2_9 17_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	15, 31	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.53 eGTM to Ports signal TOUT output selection register n (n=43)

For a description of this register, see 7579.

EGTM_TOUTSELn (n=43) Offset address: 061100_H+n*4

eGTM to Ports signal TOUT output selection register n Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	SEL5					SEL4					SEL3				
r	rw					rw					rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	SEL2					SEL1					SEL0				
r	rw					rw					rw				

Field	Bits	Type	Description
SELx (x=0)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT258 (P14.11) 01_H **CDTM0_DTM4_1**: Dead-time output of ATOM0_1 02_H **CDTM1_DTM4_1_N**: Inverted dead-time output of ATOM1_1 03_H **CDTM2_DTM4_1**: Dead-time output of ATOM2_1 04_H **CDTM0_DTM0_1**: Dead-time output of TOM0_1 05_H **CDTM1_DTM0_1_N**: Inverted dead-time output of TOM1_1 06_H **CDTM2_DTM0_1**: Dead-time output of TOM2_1 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM2_DTM2_0**: Dead-time output of TOM2_8 11_H **Reserved**: Reserved, do not use … 15_H **Reserved**: Reserved, do not use 16_H **CDTM2_DTM2_2**: Dead-time output of TOM2_10 17_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
SELx (x=1)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT259 (P14.14) 01_H **CDTM0_DTM4_1_N**: Inverted dead-time output of ATOM0_1 02_H **CDTM1_DTM4_1**: Dead-time output of ATOM1_1 03_H **CDTM2_DTM4_1_N**: Inverted dead-time output of ATOM2_1 04_H **CDTM0_DTM0_1_N**: Inverted dead-time output of TOM0_1 05_H **CDTM1_DTM0_1**: Dead-time output of TOM1_1 06_H **CDTM2_DTM0_1_N**: Inverted dead-time output of TOM2_1 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM2_DTM2_1**: Dead-time output of TOM2_9 11_H **Reserved**: Reserved, do not use … 15_H **Reserved**: Reserved, do not use 16_H **CDTM2_DTM2_3**: Dead-time output of TOM2_11 17_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=2)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT260 (P14.13) 01_H **CDTM0_DTM4_2**: Dead-time output of ATOM0_2 02_H **CDTM1_DTM4_2_N**: Inverted dead-time output of ATOM1_2 03_H **CDTM2_DTM4_2**: Dead-time output of ATOM2_2 04_H **CDTM0_DTM0_2**: Dead-time output of TOM0_2 05_H **CDTM1_DTM0_2_N**: Inverted dead-time output of TOM1_2 06_H **CDTM2_DTM0_2**: Dead-time output of TOM2_2 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM2_DTM2_2**: Dead-time output of TOM2_10 11_H **Reserved**: Reserved, do not use … 15_H **Reserved**: Reserved, do not use 16_H **CDTM2_DTM3_0**: Dead-time output of TOM2_12 17_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
SELx (x=3)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT261 (P14.12) 01_H **CDTM0_DTM4_2_N**: Inverted dead-time output of ATOM0_2 02_H **CDTM1_DTM4_2**: Dead-time output of ATOM1_2 03_H **CDTM2_DTM4_2_N**: Inverted dead-time output of ATOM2_2 04_H **CDTM0_DTM0_2_N**: Inverted dead-time output of TOM0_2 05_H **CDTM1_DTM0_2**: Dead-time output of TOM1_2 06_H **CDTM2_DTM0_2_N**: Inverted dead-time output of TOM2_2 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM2_DTM2_3**: Dead-time output of TOM2_11 11_H **Reserved**: Reserved, do not use … 15_H **Reserved**: Reserved, do not use 16_H **CDTM2_DTM3_1**: Dead-time output of TOM2_13 17_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=4)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT262 (P13.13) 01_H **CDTM0_DTM4_3**: Dead-time output of ATOM0_3 02_H **CDTM1_DTM4_3_N**: Inverted dead-time output of ATOM1_3 03_H **CDTM2_DTM4_3**: Dead-time output of ATOM2_3 04_H **CDTM0_DTM0_3**: Dead-time output of TOM0_3 05_H **CDTM1_DTM0_3_N**: Inverted dead-time output of TOM1_3 06_H **CDTM2_DTM0_3**: Dead-time output of TOM2_3 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM2_DTM3_0**: Dead-time output of TOM2_12 11_H **Reserved**: Reserved, do not use … 15_H **Reserved**: Reserved, do not use 16_H **CDTM2_DTM3_2**: Dead-time output of TOM2_14 17_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
SELx (x=5)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT263 (P14.15) 01_H **CDTM0_DTM4_3_N**: Inverted dead-time output of ATOM0_3 02_H **CDTM1_DTM4_3**: Dead-time output of ATOM1_3 03_H **CDTM2_DTM4_3_N**: Inverted dead-time output of ATOM2_3 04_H **CDTM0_DTM0_3_N**: Inverted dead-time output of TOM0_3 05_H **CDTM1_DTM0_3**: Dead-time output of TOM1_3 06_H **CDTM2_DTM0_3_N**: Inverted dead-time output of TOM2_3 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM2_DTM3_1**: Dead-time output of TOM2_13 11_H **Reserved**: Reserved, do not use … 15_H **Reserved**: Reserved, do not use 16_H **CDTM2_DTM3_3**: Dead-time output of TOM2_15 17_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	15, 31	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.54 eGTM to Ports signal TOUT output selection register n (n=44)

For a description of this register, see 7579.

EGTM_TOUTSELn (n=44) Offset address: 061100_H+n*4

eGTM to Ports signal TOUT output selection register n Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	SEL5					SEL4					SEL3				
r	rw					rw					rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	SEL2					SEL1					SEL0				
r	rw					rw					rw				

Field	Bits	Type	Description
SELx (x=0)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT264 (P13.15) 01_H **CDTM0_DTM5_0**: Dead-time output of ATOM0_4 02_H **CDTM1_DTM5_0_N**: Inverted dead-time output of ATOM1_4 03_H **CDTM2_DTM5_0**: Dead-time output of ATOM2_4 04_H **CDTM0_DTM1_0**: Dead-time output of TOM0_4 05_H **CDTM1_DTM1_0_N**: Inverted dead-time output of TOM1_4 06_H **CDTM2_DTM1_0**: Dead-time output of TOM2_4 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM2_DTM3_2**: Dead-time output of TOM2_14 11_H **Reserved**: Reserved, do not use … 13_H **Reserved**: Reserved, do not use 14_H **CDTM2_DTM0_0_N**: Inverted dead-time output of TOM2_0 15_H **Reserved**: Reserved, do not use 16_H **CDTM2_DTM1_3**: Dead-time output of TOM2_7 17_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=1)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT265 (P10.9) 01_H **CDTM0_DTM5_0_N**: Inverted dead-time output of ATOM0_4 02_H **CDTM1_DTM5_0**: Dead-time output of ATOM1_4 03_H **CDTM2_DTM5_0_N**: Inverted dead-time output of ATOM2_4 04_H **CDTM0_DTM1_0_N**: Inverted dead-time output of TOM0_4 05_H **CDTM1_DTM1_0**: Dead-time output of TOM1_4 06_H **CDTM2_DTM1_0_N**: Inverted dead-time output of TOM2_4 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM1_DTM2_1**: Dead-time output of TOM1_9 11_H **Reserved**: Reserved, do not use … 15_H **Reserved**: Reserved, do not use 16_H **CDTM2_DTM2_0**: Dead-time output of TOM2_8 17_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
SELx (x=2)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT266 (P10.10) 01_H **CDTM0_DTM5_1**: Dead-time output of ATOM0_5 02_H **CDTM1_DTM5_1_N**: Inverted dead-time output of ATOM1_5 03_H **CDTM2_DTM5_1**: Dead-time output of ATOM2_5 04_H **CDTM0_DTM1_1**: Dead-time output of TOM0_5 05_H **CDTM1_DTM1_1_N**: Inverted dead-time output of TOM1_5 06_H **CDTM2_DTM1_1**: Dead-time output of TOM2_5 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM1_DTM2_2**: Dead-time output of TOM1_10 11_H **Reserved**: Reserved, do not use … 15_H **Reserved**: Reserved, do not use 16_H **CDTM2_DTM2_1**: Dead-time output of TOM2_9 17_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=3)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT267 (P10.14) 01_H **CDTM0_DTM5_1_N**: Inverted dead-time output of ATOM0_5 02_H **CDTM1_DTM5_1**: Dead-time output of ATOM1_5 03_H **CDTM2_DTM5_1_N**: Inverted dead-time output of ATOM2_5 04_H **CDTM0_DTM1_1_N**: Inverted dead-time output of TOM0_5 05_H **CDTM1_DTM1_1**: Dead-time output of TOM1_5 06_H **CDTM2_DTM1_1_N**: Inverted dead-time output of TOM2_5 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM1_DTM2_3**: Dead-time output of TOM1_11 11_H **Reserved**: Reserved, do not use … 15_H **Reserved**: Reserved, do not use 16_H **CDTM2_DTM2_2**: Dead-time output of TOM2_10 17_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
SELx (x=4)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT268 (P10.13) 01_H **CDTM0_DTM5_2**: Dead-time output of ATOM0_6 02_H **CDTM1_DTM5_2_N**: Inverted dead-time output of ATOM1_6 03_H **CDTM2_DTM5_2**: Dead-time output of ATOM2_6 04_H **CDTM0_DTM1_2**: Dead-time output of TOM0_6 05_H **CDTM1_DTM1_2_N**: Inverted dead-time output of TOM1_6 06_H **CDTM2_DTM1_2**: Dead-time output of TOM2_6 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM1_DTM3_0**: Dead-time output of TOM1_12 11_H **Reserved**: Reserved, do not use … 15_H **Reserved**: Reserved, do not use 16_H **CDTM2_DTM2_3**: Dead-time output of TOM2_11 17_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=5)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT269 (P10.11) 01_H **CDTM0_DTM5_2_N**: Inverted dead-time output of ATOM0_6 02_H **CDTM1_DTM5_2**: Dead-time output of ATOM1_6 03_H **CDTM2_DTM5_2_N**: Inverted dead-time output of ATOM2_6 04_H **CDTM0_DTM1_2_N**: Inverted dead-time output of TOM0_6 05_H **CDTM1_DTM1_2**: Dead-time output of TOM1_6 06_H **CDTM2_DTM1_2_N**: Inverted dead-time output of TOM2_6 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM1_DTM3_1**: Dead-time output of TOM1_13 11_H **Reserved**: Reserved, do not use … 15_H **Reserved**: Reserved, do not use 16_H **CDTM2_DTM3_0**: Dead-time output of TOM2_12 17_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	15, 31	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.55 eGTM to Ports signal TOUT output selection register n (n=45)

For a description of this register, see 7579.

EGTM_TOUTSELn (n=45) Offset address: 061100_H+n*4

eGTM to Ports signal TOUT output selection register n Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	SEL5					SEL4					SEL3				
r	rw					rw					rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	SEL2					SEL1					SEL0				
r	rw					rw					rw				

Field	Bits	Type	Description
SELx (x=0)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT270 (P10.15) 01_H **CDTM0_DTM5_3**: Dead-time output of ATOM0_7 02_H **CDTM1_DTM5_3_N**: Inverted dead-time output of ATOM1_7 03_H **CDTM2_DTM5_3**: Dead-time output of ATOM2_7 04_H **CDTM0_DTM1_3**: Dead-time output of TOM0_7 05_H **CDTM1_DTM1_3_N**: Inverted dead-time output of TOM1_7 06_H **CDTM2_DTM1_3**: Dead-time output of TOM2_7 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM1_DTM3_2**: Dead-time output of TOM1_14 11_H **Reserved**: Reserved, do not use … 15_H **Reserved**: Reserved, do not use 16_H **CDTM2_DTM3_1**: Dead-time output of TOM2_13 17_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
SELx (x=1)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT271 (P03.0) 01_H **CDTM0_DTM5_3_N**: Inverted dead-time output of ATOM0_7 02_H **CDTM1_DTM5_3**: Dead-time output of ATOM1_7 03_H **CDTM2_DTM5_3_N**: Inverted dead-time output of ATOM2_7 04_H **CDTM0_DTM1_3_N**: Inverted dead-time output of TOM0_7 05_H **CDTM1_DTM1_3**: Dead-time output of TOM1_7 06_H **CDTM2_DTM1_3_N**: Inverted dead-time output of TOM2_7 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM4_0**: Dead-time output of ATOM0_0 11_H **CDTM1_DTM4_0**: Dead-time output of ATOM1_0 12_H **CDTM2_DTM5_2_N**: Inverted dead-time output of ATOM2_6 13_H **CDTM0_DTM1_2**: Dead-time output of TOM0_6 14_H **CDTM1_DTM1_0**: Dead-time output of TOM1_4 15_H **CDTM2_DTM2_0**: Dead-time output of TOM2_8 16_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=2)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT272 (P03.1) 01_H **CDTM0_DTM4_0**: Dead-time output of ATOM0_0 02_H **CDTM1_DTM4_0_N**: Inverted dead-time output of ATOM1_0 03_H **CDTM2_DTM4_0**: Dead-time output of ATOM2_0 04_H **CDTM0_DTM2_0**: Dead-time output of TOM0_8 05_H **CDTM1_DTM2_0_N**: Inverted dead-time output of TOM1_8 06_H **CDTM2_DTM2_0**: Dead-time output of TOM2_8 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM4_0_N**: Inverted dead-time output of ATOM0_0 11_H **CDTM1_DTM4_0_N**: Inverted dead-time output of ATOM1_0 12_H **CDTM2_DTM5_3**: Dead-time output of ATOM2_7 13_H **CDTM0_DTM1_3_N**: Inverted dead-time output of TOM0_7 14_H **CDTM1_DTM1_0_N**: Inverted dead-time output of TOM1_4 15_H **CDTM2_DTM2_0_N**: Inverted dead-time output of TOM2_8 16_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
SELx (x=3)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT273 (P03.2) 01_H **CDTM0_DTM4_0_N**: Inverted dead-time output of ATOM0_0 02_H **CDTM1_DTM4_0**: Dead-time output of ATOM1_0 03_H **CDTM2_DTM4_0_N**: Inverted dead-time output of ATOM2_0 04_H **CDTM0_DTM2_0_N**: Inverted dead-time output of TOM0_8 05_H **CDTM1_DTM2_0**: Dead-time output of TOM1_8 06_H **CDTM2_DTM2_0_N**: Inverted dead-time output of TOM2_8 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM4_1**: Dead-time output of ATOM0_1 11_H **CDTM1_DTM4_1**: Dead-time output of ATOM1_1 12_H **CDTM2_DTM5_3_N**: Inverted dead-time output of ATOM2_7 13_H **CDTM0_DTM1_3**: Dead-time output of TOM0_7 14_H **CDTM1_DTM1_1**: Dead-time output of TOM1_5 15_H **CDTM2_DTM2_1**: Dead-time output of TOM2_9 16_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=4)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT274 (P03.3) 01_H **CDTM0_DTM4_1**: Dead-time output of ATOM0_1 02_H **CDTM1_DTM4_1_N**: Inverted dead-time output of ATOM1_1 03_H **CDTM2_DTM4_1**: Dead-time output of ATOM2_1 04_H **CDTM0_DTM2_1**: Dead-time output of TOM0_9 05_H **CDTM1_DTM2_1_N**: Inverted dead-time output of TOM1_9 06_H **CDTM2_DTM2_1**: Dead-time output of TOM2_9 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM4_1_N**: Inverted dead-time output of ATOM0_1 11_H **CDTM1_DTM4_1_N**: Inverted dead-time output of ATOM1_1 12_H **CDTM2_DTM4_0**: Dead-time output of ATOM2_0 13_H **CDTM0_DTM2_0_N**: Inverted dead-time output of TOM0_8 14_H **CDTM1_DTM1_1_N**: Inverted dead-time output of TOM1_5 15_H **CDTM2_DTM2_1_N**: Inverted dead-time output of TOM2_9 16_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
SELx (x=5)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT275 (P03.4) 01_H **CDTM0_DTM4_1_N**: Inverted dead-time output of ATOM0_1 02_H **CDTM1_DTM4_1**: Dead-time output of ATOM1_1 03_H **CDTM2_DTM4_1_N**: Inverted dead-time output of ATOM2_1 04_H **CDTM0_DTM2_1_N**: Inverted dead-time output of TOM0_9 05_H **CDTM1_DTM2_1**: Dead-time output of TOM1_9 06_H **CDTM2_DTM2_1_N**: Inverted dead-time output of TOM2_9 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM4_2**: Dead-time output of ATOM0_2 11_H **CDTM1_DTM4_2**: Dead-time output of ATOM1_2 12_H **CDTM2_DTM4_0_N**: Inverted dead-time output of ATOM2_0 13_H **CDTM0_DTM2_0**: Dead-time output of TOM0_8 14_H **CDTM1_DTM1_2**: Dead-time output of TOM1_6 15_H **CDTM2_DTM2_2**: Dead-time output of TOM2_10 16_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	15, 31	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.56 eGTM to Ports signal TOUT output selection register n (n=46)

For a description of this register, see 7579.

EGTM_TOUTSELn (n=46) Offset address: 061100_H+n*4

eGTM to Ports signal TOUT output selection register n Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	SEL5					SEL4					SEL3				
r	rw					rw					rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	SEL2					SEL1					SEL0				
r	rw					rw					rw				

Field	Bits	Type	Description
SELx (x=0)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT276 (P03.5) 01_H **CDTM0_DTM4_2**: Dead-time output of ATOM0_2 02_H **CDTM1_DTM4_2_N**: Inverted dead-time output of ATOM1_2 03_H **CDTM2_DTM4_2**: Dead-time output of ATOM2_2 04_H **CDTM0_DTM2_2**: Dead-time output of TOM0_10 05_H **CDTM1_DTM2_2_N**: Inverted dead-time output of TOM1_10 06_H **CDTM2_DTM2_2**: Dead-time output of TOM2_10 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM4_2_N**: Inverted dead-time output of ATOM0_2 11_H **CDTM1_DTM4_2_N**: Inverted dead-time output of ATOM1_2 12_H **CDTM2_DTM4_1**: Dead-time output of ATOM2_1 13_H **CDTM0_DTM2_1_N**: Inverted dead-time output of TOM0_9 14_H **CDTM1_DTM1_2_N**: Inverted dead-time output of TOM1_6 15_H **CDTM2_DTM2_2_N**: Inverted dead-time output of TOM2_10 16_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=1)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT277 (P03.6) 01_H **CDTM0_DTM4_2_N**: Inverted dead-time output of ATOM0_2 02_H **CDTM1_DTM4_2**: Dead-time output of ATOM1_2 03_H **CDTM2_DTM4_2_N**: Inverted dead-time output of ATOM2_2 04_H **CDTM0_DTM2_2_N**: Inverted dead-time output of TOM0_10 05_H **CDTM1_DTM2_2**: Dead-time output of TOM1_10 06_H **CDTM2_DTM2_2_N**: Inverted dead-time output of TOM2_10 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM4_3**: Dead-time output of ATOM0_3 11_H **CDTM1_DTM4_3**: Dead-time output of ATOM1_3 12_H **CDTM2_DTM4_1_N**: Inverted dead-time output of ATOM2_1 13_H **CDTM0_DTM2_1**: Dead-time output of TOM0_9 14_H **CDTM1_DTM1_3**: Dead-time output of TOM1_7 15_H **CDTM2_DTM2_3**: Dead-time output of TOM2_11 16_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
SELx (x=2)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT278 (P03.7) 01_H **CDTM0_DTM4_3**: Dead-time output of ATOM0_3 02_H **CDTM1_DTM4_3_N**: Inverted dead-time output of ATOM1_3 03_H **CDTM2_DTM4_3**: Dead-time output of ATOM2_3 04_H **CDTM0_DTM2_3**: Dead-time output of TOM0_11 05_H **CDTM1_DTM2_3_N**: Inverted dead-time output of TOM1_11 06_H **CDTM2_DTM2_3**: Dead-time output of TOM2_11 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM4_3_N**: Inverted dead-time output of ATOM0_3 11_H **CDTM1_DTM4_3_N**: Inverted dead-time output of ATOM1_3 12_H **CDTM2_DTM4_2**: Dead-time output of ATOM2_2 13_H **CDTM0_DTM2_2_N**: Inverted dead-time output of TOM0_10 14_H **CDTM1_DTM1_3_N**: Inverted dead-time output of TOM1_7 15_H **CDTM2_DTM2_3_N**: Inverted dead-time output of TOM2_11 16_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=3)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT279 (P03.8) 01_H **CDTM0_DTM4_3_N**: Inverted dead-time output of ATOM0_3 02_H **CDTM1_DTM4_3**: Dead-time output of ATOM1_3 03_H **CDTM2_DTM4_3_N**: Inverted dead-time output of ATOM2_3 04_H **CDTM0_DTM2_3_N**: Inverted dead-time output of TOM0_11 05_H **CDTM1_DTM2_3**: Dead-time output of TOM1_11 06_H **CDTM2_DTM2_3_N**: Inverted dead-time output of TOM2_11 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM5_0**: Dead-time output of ATOM0_4 11_H **CDTM1_DTM5_0**: Dead-time output of ATOM1_4 12_H **CDTM2_DTM4_2_N**: Inverted dead-time output of ATOM2_2 13_H **CDTM0_DTM2_2**: Dead-time output of TOM0_10 14_H **CDTM1_DTM2_0**: Dead-time output of TOM1_8 15_H **CDTM2_DTM3_0**: Dead-time output of TOM2_12 16_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
SELx (x=4)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT280 (P03.9) 01_H **CDTM0_DTM5_0**: Dead-time output of ATOM0_4 02_H **CDTM1_DTM5_0_N**: Inverted dead-time output of ATOM1_4 03_H **CDTM2_DTM5_0**: Dead-time output of ATOM2_4 04_H **CDTM0_DTM3_0**: Dead-time output of TOM0_12 05_H **CDTM1_DTM3_0_N**: Inverted dead-time output of TOM1_12 06_H **CDTM2_DTM3_0**: Dead-time output of TOM2_12 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM5_0_N**: Inverted dead-time output of ATOM0_4 11_H **CDTM1_DTM5_0_N**: Inverted dead-time output of ATOM1_4 12_H **CDTM2_DTM4_3**: Dead-time output of ATOM2_3 13_H **CDTM0_DTM2_3_N**: Inverted dead-time output of TOM0_11 14_H **CDTM1_DTM2_0_N**: Inverted dead-time output of TOM1_8 15_H **CDTM2_DTM3_0_N**: Inverted dead-time output of TOM2_12 16_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=5)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT281 (P03.10) 01_H **CDTM0_DTM5_0_N**: Inverted dead-time output of ATOM0_4 02_H **CDTM1_DTM5_0**: Dead-time output of ATOM1_4 03_H **CDTM2_DTM5_0_N**: Inverted dead-time output of ATOM2_4 04_H **CDTM0_DTM3_0_N**: Inverted dead-time output of TOM0_12 05_H **CDTM1_DTM3_0**: Dead-time output of TOM1_12 06_H **CDTM2_DTM3_0_N**: Inverted dead-time output of TOM2_12 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM5_1**: Dead-time output of ATOM0_5 11_H **CDTM1_DTM5_1**: Dead-time output of ATOM1_5 12_H **CDTM2_DTM4_3_N**: Inverted dead-time output of ATOM2_3 13_H **CDTM0_DTM2_3**: Dead-time output of TOM0_11 14_H **CDTM1_DTM2_1**: Dead-time output of TOM1_9 15_H **CDTM2_DTM3_1**: Dead-time output of TOM2_13 16_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	15, 31	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.57 eGTM to Ports signal TOUT output selection register n (n=47)

For a description of this register, see 7579.

EGTM_TOUTSELn (n=47) Offset address: 061100_H+n*4

eGTM to Ports signal TOUT output selection register n Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	SEL5					SEL4					SEL3				
r	rw					rw					rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	SEL2					SEL1					SEL0				
r	rw					rw					rw				

Field	Bits	Type	Description
SELx (x=0)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT282 (P03.11) 01_H **CDTM0_DTM5_1**: Dead-time output of ATOM0_5 02_H **CDTM1_DTM5_1_N**: Inverted dead-time output of ATOM1_5 03_H **CDTM2_DTM5_1**: Dead-time output of ATOM2_5 04_H **CDTM0_DTM3_1**: Dead-time output of TOM0_13 05_H **CDTM1_DTM3_1_N**: Inverted dead-time output of TOM1_13 06_H **CDTM2_DTM3_1**: Dead-time output of TOM2_13 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM5_1_N**: Inverted dead-time output of ATOM0_5 11_H **CDTM1_DTM5_1_N**: Inverted dead-time output of ATOM1_5 12_H **CDTM2_DTM5_0**: Dead-time output of ATOM2_4 13_H **CDTM0_DTM3_0_N**: Inverted dead-time output of TOM0_12 14_H **CDTM1_DTM2_1_N**: Inverted dead-time output of TOM1_9 15_H **CDTM2_DTM3_1_N**: Inverted dead-time output of TOM2_13 16_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
SELx (x=1)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT283 (P03.12) 01_H **CDTM0_DTM5_1_N**: Inverted dead-time output of ATOM0_5 02_H **CDTM1_DTM5_1**: Dead-time output of ATOM1_5 03_H **CDTM2_DTM5_1_N**: Inverted dead-time output of ATOM2_5 04_H **CDTM0_DTM3_1_N**: Inverted dead-time output of TOM0_13 05_H **CDTM1_DTM3_1**: Dead-time output of TOM1_13 06_H **CDTM2_DTM3_1_N**: Inverted dead-time output of TOM2_13 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM5_2**: Dead-time output of ATOM0_6 11_H **CDTM1_DTM5_2**: Dead-time output of ATOM1_6 12_H **CDTM2_DTM5_0_N**: Inverted dead-time output of ATOM2_4 13_H **CDTM0_DTM3_0**: Dead-time output of TOM0_12 14_H **CDTM1_DTM2_2**: Dead-time output of TOM1_10 15_H **CDTM2_DTM3_2**: Dead-time output of TOM2_14 16_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=2)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT284 (P03.13) 01_H **CDTM0_DTM5_2**: Dead-time output of ATOM0_6 02_H **CDTM1_DTM5_2_N**: Inverted dead-time output of ATOM1_6 03_H **CDTM2_DTM5_2**: Dead-time output of ATOM2_6 04_H **CDTM0_DTM3_2**: Dead-time output of TOM0_14 05_H **CDTM1_DTM3_2_N**: Inverted dead-time output of TOM1_14 06_H **CDTM2_DTM3_2**: Dead-time output of TOM2_14 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM5_2_N**: Inverted dead-time output of ATOM0_6 11_H **CDTM1_DTM5_2_N**: Inverted dead-time output of ATOM1_6 12_H **CDTM2_DTM5_1**: Dead-time output of ATOM2_5 13_H **CDTM0_DTM3_1_N**: Inverted dead-time output of TOM0_13 14_H **CDTM1_DTM2_2_N**: Inverted dead-time output of TOM1_10 15_H **CDTM2_DTM3_2_N**: Inverted dead-time output of TOM2_14 16_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
SELx (x=3)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT285 (P03.14) 01_H **CDTM0_DTM5_2_N**: Inverted dead-time output of ATOM0_6 02_H **CDTM1_DTM5_2**: Dead-time output of ATOM1_6 03_H **CDTM2_DTM5_2_N**: Inverted dead-time output of ATOM2_6 04_H **CDTM0_DTM3_2_N**: Inverted dead-time output of TOM0_14 05_H **CDTM1_DTM3_2**: Dead-time output of TOM1_14 06_H **CDTM2_DTM3_2_N**: Inverted dead-time output of TOM2_14 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM5_3**: Dead-time output of ATOM0_7 11_H **CDTM1_DTM5_3**: Dead-time output of ATOM1_7 12_H **CDTM2_DTM5_1_N**: Inverted dead-time output of ATOM2_5 13_H **CDTM0_DTM3_1**: Dead-time output of TOM0_13 14_H **CDTM1_DTM2_3**: Dead-time output of TOM1_11 15_H **CDTM2_DTM3_3**: Dead-time output of TOM2_15 16_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=4)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT286 (P03.15) 01_H **CDTM0_DTM5_3**: Dead-time output of ATOM0_7 02_H **CDTM1_DTM5_3_N**: Inverted dead-time output of ATOM1_7 03_H **CDTM2_DTM5_3**: Dead-time output of ATOM2_7 04_H **CDTM0_DTM3_3**: Dead-time output of TOM0_15 05_H **CDTM1_DTM3_3_N**: Inverted dead-time output of TOM1_15 06_H **CDTM2_DTM3_3**: Dead-time output of TOM2_15 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM5_3_N**: Inverted dead-time output of ATOM0_7 11_H **CDTM1_DTM5_3_N**: Inverted dead-time output of ATOM1_7 12_H **CDTM2_DTM5_2**: Dead-time output of ATOM2_6 13_H **CDTM0_DTM3_2_N**: Inverted dead-time output of TOM0_14 14_H **CDTM1_DTM2_3_N**: Inverted dead-time output of TOM1_11 15_H **CDTM2_DTM3_3_N**: Inverted dead-time output of TOM2_15 16_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
SELx (x=5)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT287 (P04.0) 01_H **CDTM0_DTM5_3_N**: Inverted dead-time output of ATOM0_7 02_H **CDTM1_DTM5_3**: Dead-time output of ATOM1_7 03_H **CDTM2_DTM5_3_N**: Inverted dead-time output of ATOM2_7 04_H **CDTM0_DTM3_3_N**: Inverted dead-time output of TOM0_15 05_H **CDTM1_DTM3_3**: Dead-time output of TOM1_15 06_H **CDTM2_DTM3_3_N**: Inverted dead-time output of TOM2_15 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM4_0**: Dead-time output of ATOM0_0 11_H **CDTM1_DTM4_0**: Dead-time output of ATOM1_0 12_H **CDTM2_DTM5_2_N**: Inverted dead-time output of ATOM2_6 13_H **CDTM0_DTM3_2**: Dead-time output of TOM0_14 14_H **CDTM1_DTM3_0**: Dead-time output of TOM1_12 15_H **CDTM2_DTM0_0**: Dead-time output of TOM2_0 16_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	15, 31	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.58 eGTM to Ports signal TOUT output selection register n (n=48)

For a description of this register, see 7579.

EGTM_TOUTSELn (n=48) Offset address: 061100_H+n*4

eGTM to Ports signal TOUT output selection register n Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	SEL5					SEL4					SEL3				
r	rw					rw					rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	SEL2					SEL1					SEL0				
r	rw					rw					rw				

Field	Bits	Type	Description
SELx (x=0)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT288 (P04.1) 01_H **CDTM0_DTM4_0**: Dead-time output of ATOM0_0 02_H **CDTM1_DTM4_0_N**: Inverted dead-time output of ATOM1_0 03_H **CDTM2_DTM4_0**: Dead-time output of ATOM2_0 04_H **CDTM0_DTM0_0**: Dead-time output of TOM0_0 05_H **CDTM1_DTM0_0_N**: Inverted dead-time output of TOM1_0 06_H **CDTM2_DTM0_0**: Dead-time output of TOM2_0 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM4_0_N**: Inverted dead-time output of ATOM0_0 11_H **CDTM1_DTM4_0_N**: Inverted dead-time output of ATOM1_0 12_H **CDTM2_DTM5_3**: Dead-time output of ATOM2_7 13_H **CDTM0_DTM3_3_N**: Inverted dead-time output of TOM0_15 14_H **CDTM1_DTM3_0_N**: Inverted dead-time output of TOM1_12 15_H **CDTM2_DTM0_0_N**: Inverted dead-time output of TOM2_0 16_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=1)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT289 (P04.2) 01_H **CDTM0_DTM4_0_N**: Inverted dead-time output of ATOM0_0 02_H **CDTM1_DTM4_0**: Dead-time output of ATOM1_0 03_H **CDTM2_DTM4_0_N**: Inverted dead-time output of ATOM2_0 04_H **CDTM0_DTM0_0_N**: Inverted dead-time output of TOM0_0 05_H **CDTM1_DTM0_0**: Dead-time output of TOM1_0 06_H **CDTM2_DTM0_0_N**: Inverted dead-time output of TOM2_0 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM4_1**: Dead-time output of ATOM0_1 11_H **CDTM1_DTM4_1**: Dead-time output of ATOM1_1 12_H **CDTM2_DTM5_3_N**: Inverted dead-time output of ATOM2_7 13_H **CDTM0_DTM3_3**: Dead-time output of TOM0_15 14_H **CDTM1_DTM3_1**: Dead-time output of TOM1_13 15_H **CDTM2_DTM0_1**: Dead-time output of TOM2_1 16_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
SELx (x=2)	5*x+4:5*x	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT290 (P04.3) 01_H **CDTM0_DTM4_1**: Dead-time output of ATOM0_1 02_H **CDTM1_DTM4_1_N**: Inverted dead-time output of ATOM1_1 03_H **CDTM2_DTM4_1**: Dead-time output of ATOM2_1 04_H **CDTM0_DTM0_1**: Dead-time output of TOM0_1 05_H **CDTM1_DTM0_1_N**: Inverted dead-time output of TOM1_1 06_H **CDTM2_DTM0_1**: Dead-time output of TOM2_1 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM4_1_N**: Inverted dead-time output of ATOM0_1 11_H **CDTM1_DTM4_1_N**: Inverted dead-time output of ATOM1_1 12_H **CDTM2_DTM4_0**: Dead-time output of ATOM2_0 13_H **CDTM0_DTM0_0**: Dead-time output of TOM0_0 14_H **CDTM1_DTM3_1_N**: Inverted dead-time output of TOM1_13 15_H **CDTM2_DTM0_1_N**: Inverted dead-time output of TOM2_1 16_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=3)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT291 (P04.4) 01_H **CDTM0_DTM4_1_N**: Inverted dead-time output of ATOM0_1 02_H **CDTM1_DTM4_1**: Dead-time output of ATOM1_1 03_H **CDTM2_DTM4_1_N**: Inverted dead-time output of ATOM2_1 04_H **CDTM0_DTM0_1_N**: Inverted dead-time output of TOM0_1 05_H **CDTM1_DTM0_1**: Dead-time output of TOM1_1 06_H **CDTM2_DTM0_1_N**: Inverted dead-time output of TOM2_1 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM4_2**: Dead-time output of ATOM0_2 11_H **CDTM1_DTM4_2**: Dead-time output of ATOM1_2 12_H **CDTM2_DTM4_0_N**: Inverted dead-time output of ATOM2_0 13_H **CDTM0_DTM0_0_N**: Inverted dead-time output of TOM0_0 14_H **CDTM1_DTM3_2**: Dead-time output of TOM1_14 15_H **CDTM2_DTM0_2**: Dead-time output of TOM2_2 16_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
SELx (x=4)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT292 (P04.5) 01_H **CDTM0_DTM4_2**: Dead-time output of ATOM0_2 02_H **CDTM1_DTM4_2_N**: Inverted dead-time output of ATOM1_2 03_H **CDTM2_DTM4_2**: Dead-time output of ATOM2_2 04_H **CDTM0_DTM0_2**: Dead-time output of TOM0_2 05_H **CDTM1_DTM0_2_N**: Inverted dead-time output of TOM1_2 06_H **CDTM2_DTM0_2**: Dead-time output of TOM2_2 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM4_2_N**: Inverted dead-time output of ATOM0_2 11_H **CDTM1_DTM4_2_N**: Inverted dead-time output of ATOM1_2 12_H **CDTM2_DTM4_1**: Dead-time output of ATOM2_1 13_H **CDTM0_DTM0_1**: Dead-time output of TOM0_1 14_H **CDTM1_DTM3_2_N**: Inverted dead-time output of TOM1_14 15_H **CDTM2_DTM0_2_N**: Inverted dead-time output of TOM2_2 16_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=5)	5*x+5:5*x+1	rw	**eGTM to Ports signal TOUT(n*6+x) output selection** 00_H **None**: Safe state. No eGTM signal connected to TOUT293 (no pin) 01_H **CDTM0_DTM4_2_N**: Inverted dead-time output of ATOM0_2 02_H **CDTM1_DTM4_2**: Dead-time output of ATOM1_2 03_H **CDTM2_DTM4_2_N**: Inverted dead-time output of ATOM2_2 04_H **CDTM0_DTM0_2_N**: Inverted dead-time output of TOM0_2 05_H **CDTM1_DTM0_2**: Dead-time output of TOM1_2 06_H **CDTM2_DTM0_2_N**: Inverted dead-time output of TOM2_2 07_H **Reserved**: Reserved, do not use … $0F_H$ **Reserved**: Reserved, do not use 10_H **CDTM0_DTM4_3**: Dead-time output of ATOM0_3 11_H **CDTM1_DTM4_3**: Dead-time output of ATOM1_3 12_H **CDTM2_DTM4_1_N**: Inverted dead-time output of ATOM2_1 13_H **CDTM0_DTM0_1_N**: Inverted dead-time output of TOM0_1 14_H **CDTM1_DTM3_3**: Dead-time output of TOM1_15 15_H **CDTM2_DTM0_3**: Dead-time output of TOM2_3 16_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	15, 31	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.59 eGTM to MSC SETi output selection register j (j=0;i=0)

For a description of this register, see 7608.

EGTM_MSCSETi_CONj (i=0;j=0) Offset address: 061200_H+i*10_H+j*4

eGTM to MSC SETi output selection register j Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0			SEL3					0			SEL2				
r			rw					r			rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			SEL1					0			SEL0				
r			rw					r			rw				

Field	Bits	Type	Description
SELx (x=0-3)	8*x+4:8*x	rw	**SETi.x signal output selection** 00_H **CDTM0_DTM0_0**: Dead-time output of TOM0_0 01_H **CDTM0_DTM0_1**: Dead-time output of TOM0_1 02_H **CDTM0_DTM0_2**: Dead-time output of TOM0_2 03_H **CDTM0_DTM0_3**: Dead-time output of TOM0_3 04_H **CDTM0_DTM1_0**: Dead-time output of TOM0_4 05_H **CDTM0_DTM1_1**: Dead-time output of TOM0_5 06_H **CDTM0_DTM1_2**: Dead-time output of TOM0_6 07_H **CDTM0_DTM1_3**: Dead-time output of TOM0_7 08_H **CDTM0_DTM2_0**: Dead-time output of TOM0_8 09_H **CDTM0_DTM2_1**: Dead-time output of TOM0_9 $0A_H$ **CDTM0_DTM2_2**: Dead-time output of TOM0_10 $0B_H$ **CDTM0_DTM2_3**: Dead-time output of TOM0_11 $0C_H$ **CDTM0_DTM3_0**: Dead-time output of TOM0_12 $0D_H$ **CDTM0_DTM3_1**: Dead-time output of TOM0_13 $0E_H$ **CDTM0_DTM3_2**: Dead-time output of TOM0_14 $0F_H$ **CDTM0_DTM3_3**: Dead-time output of TOM0_15 10_H **CDTM0_DTM4_0**: Dead-time output of ATOM0_0 11_H **CDTM0_DTM4_1**: Dead-time output of ATOM0_1 12_H **CDTM0_DTM4_2**: Dead-time output of ATOM0_2 13_H **CDTM0_DTM4_3**: Dead-time output of ATOM0_3 14_H **CDTM0_DTM5_0**: Dead-time output of ATOM0_4 15_H **CDTM0_DTM5_1**: Dead-time output of ATOM0_5 16_H **CDTM0_DTM5_2**: Dead-time output of ATOM0_6 17_H **CDTM0_DTM5_3**: Dead-time output of ATOM0_7 18_H **CDTM2_DTM4_0**: Dead-time output of ATOM2_0 19_H **CDTM2_DTM4_1**: Dead-time output of ATOM2_1 $1A_H$ **CDTM2_DTM4_2**: Dead-time output of ATOM2_2 $1B_H$ **CDTM2_DTM4_3**: Dead-time output of ATOM2_3 $1C_H$ **CDTM2_DTM5_0**: Dead-time output of ATOM2_4 $1D_H$ **CDTM2_DTM5_1**: Dead-time output of ATOM2_5 $1E_H$ **CDTM2_DTM5_2**: Dead-time output of ATOM2_6 $1F_H$ **CDTM2_DTM5_3**: Dead-time output of ATOM2_7
0	31:29, 23:21, 15:13, 7:5	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.60 eGTM to MSC SETi output selection register j (j=1;i=0)

For a description of this register, see 7608.

EGTM_MSCSETi_CONj (i=0;j=1) Offset address: 061200_H+i*10_H+j*4

eGTM to MSC SETi output selection register j Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0			SEL7					0			SEL6				
r			rw					r			rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			SEL5					0			SEL4				
r			rw					r			rw				

Field	Bits	Type	Description
SELx (x=4-7)	8*x-28:8*x-32	rw	**SETi.x signal output selection** 00_H **CDTM0_DTM0_0**: Dead-time output of TOM0_0 01_H **CDTM0_DTM0_1**: Dead-time output of TOM0_1 02_H **CDTM0_DTM0_2**: Dead-time output of TOM0_2 03_H **CDTM0_DTM0_3**: Dead-time output of TOM0_3 04_H **CDTM0_DTM1_0**: Dead-time output of TOM0_4 05_H **CDTM0_DTM1_1**: Dead-time output of TOM0_5 06_H **CDTM0_DTM1_2**: Dead-time output of TOM0_6 07_H **CDTM0_DTM1_3**: Dead-time output of TOM0_7 08_H **CDTM0_DTM2_0**: Dead-time output of TOM0_8 09_H **CDTM0_DTM2_1**: Dead-time output of TOM0_9 $0A_H$ **CDTM0_DTM2_2**: Dead-time output of TOM0_10 $0B_H$ **CDTM0_DTM2_3**: Dead-time output of TOM0_11 $0C_H$ **CDTM0_DTM3_0**: Dead-time output of TOM0_12 $0D_H$ **CDTM0_DTM3_1**: Dead-time output of TOM0_13 $0E_H$ **CDTM0_DTM3_2**: Dead-time output of TOM0_14 $0F_H$ **CDTM0_DTM3_3**: Dead-time output of TOM0_15 10_H **CDTM0_DTM4_0**: Dead-time output of ATOM0_0 11_H **CDTM0_DTM4_1**: Dead-time output of ATOM0_1 12_H **CDTM0_DTM4_2**: Dead-time output of ATOM0_2 13_H **CDTM0_DTM4_3**: Dead-time output of ATOM0_3 14_H **CDTM0_DTM5_0**: Dead-time output of ATOM0_4 15_H **CDTM0_DTM5_1**: Dead-time output of ATOM0_5 16_H **CDTM0_DTM5_2**: Dead-time output of ATOM0_6 17_H **CDTM0_DTM5_3**: Dead-time output of ATOM0_7 18_H **CDTM2_DTM4_0**: Dead-time output of ATOM2_0 19_H **CDTM2_DTM4_1**: Dead-time output of ATOM2_1 $1A_H$ **CDTM2_DTM4_2**: Dead-time output of ATOM2_2 $1B_H$ **CDTM2_DTM4_3**: Dead-time output of ATOM2_3 $1C_H$ **CDTM2_DTM5_0**: Dead-time output of ATOM2_4 $1D_H$ **CDTM2_DTM5_1**: Dead-time output of ATOM2_5 $1E_H$ **CDTM2_DTM5_2**: Dead-time output of ATOM2_6 $1F_H$ **CDTM2_DTM5_3**: Dead-time output of ATOM2_7
0	31:29, 23:21, 15:13, 7:5	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.61 eGTM to MSC SETi output selection register j (j=2;i=0)

For a description of this register, see 7608.

EGTM_MSCSETi_CONj (i=0;j=2) Offset address: 061200_H+i*10_H+j*4

eGTM to MSC SETi output selection register j Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0			SEL11					0			SEL10				
r			rw					r			rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			SEL9					0			SEL8				
r			rw					r			rw				

Field	Bits	Type	Description
SELx (x=8-11)	8*x-60:8*x-64	rw	**SETi.x signal output selection** 00_H **CDTM0_DTM0_0**: Dead-time output of TOM0_0 01_H **CDTM0_DTM0_1**: Dead-time output of TOM0_1 02_H **CDTM0_DTM0_2**: Dead-time output of TOM0_2 03_H **CDTM0_DTM0_3**: Dead-time output of TOM0_3 04_H **CDTM0_DTM1_0**: Dead-time output of TOM0_4 05_H **CDTM0_DTM1_1**: Dead-time output of TOM0_5 06_H **CDTM0_DTM1_2**: Dead-time output of TOM0_6 07_H **CDTM0_DTM1_3**: Dead-time output of TOM0_7 08_H **CDTM0_DTM2_0**: Dead-time output of TOM0_8 09_H **CDTM0_DTM2_1**: Dead-time output of TOM0_9 $0A_H$ **CDTM0_DTM2_2**: Dead-time output of TOM0_10 $0B_H$ **CDTM0_DTM2_3**: Dead-time output of TOM0_11 $0C_H$ **CDTM0_DTM3_0**: Dead-time output of TOM0_12 $0D_H$ **CDTM0_DTM3_1**: Dead-time output of TOM0_13 $0E_H$ **CDTM0_DTM3_2**: Dead-time output of TOM0_14 $0F_H$ **CDTM0_DTM3_3**: Dead-time output of TOM0_15 10_H **CDTM0_DTM4_0**: Dead-time output of ATOM0_0 11_H **CDTM0_DTM4_1**: Dead-time output of ATOM0_1 12_H **CDTM0_DTM4_2**: Dead-time output of ATOM0_2 13_H **CDTM0_DTM4_3**: Dead-time output of ATOM0_3 14_H **CDTM0_DTM5_0**: Dead-time output of ATOM0_4 15_H **CDTM0_DTM5_1**: Dead-time output of ATOM0_5 16_H **CDTM0_DTM5_2**: Dead-time output of ATOM0_6 17_H **CDTM0_DTM5_3**: Dead-time output of ATOM0_7 18_H **CDTM2_DTM4_0**: Dead-time output of ATOM2_0 19_H **CDTM2_DTM4_1**: Dead-time output of ATOM2_1 $1A_H$ **CDTM2_DTM4_2**: Dead-time output of ATOM2_2 $1B_H$ **CDTM2_DTM4_3**: Dead-time output of ATOM2_3 $1C_H$ **CDTM2_DTM5_0**: Dead-time output of ATOM2_4 $1D_H$ **CDTM2_DTM5_1**: Dead-time output of ATOM2_5 $1E_H$ **CDTM2_DTM5_2**: Dead-time output of ATOM2_6 $1F_H$ **CDTM2_DTM5_3**: Dead-time output of ATOM2_7
0	31:29, 23:21, 15:13, 7:5	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.62 eGTM to MSC SETi output selection register j (j=3;i=0)

For a description of this register, see 7608.

EGTM_MSCSETi_CONj (i=0;j=3)

eGTM to MSC SETi output selection register j

Offset address: 061200_H+i*10_H+j*4

Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0			SEL15					0			SEL14				
r			rw					r			rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			SEL13					0			SEL12				
r			rw					r			rw				

Field	Bits	Type	Description
SELx (x=12-15)	8*x-92:8*x-96	rw	**SETi.x signal output selection** 00_H **CDTM0_DTM0_0**: Dead-time output of TOM0_0 01_H **CDTM0_DTM0_1**: Dead-time output of TOM0_1 02_H **CDTM0_DTM0_2**: Dead-time output of TOM0_2 03_H **CDTM0_DTM0_3**: Dead-time output of TOM0_3 04_H **CDTM0_DTM1_0**: Dead-time output of TOM0_4 05_H **CDTM0_DTM1_1**: Dead-time output of TOM0_5 06_H **CDTM0_DTM1_2**: Dead-time output of TOM0_6 07_H **CDTM0_DTM1_3**: Dead-time output of TOM0_7 08_H **CDTM0_DTM2_0**: Dead-time output of TOM0_8 09_H **CDTM0_DTM2_1**: Dead-time output of TOM0_9 $0A_H$ **CDTM0_DTM2_2**: Dead-time output of TOM0_10 $0B_H$ **CDTM0_DTM2_3**: Dead-time output of TOM0_11 $0C_H$ **CDTM0_DTM3_0**: Dead-time output of TOM0_12 $0D_H$ **CDTM0_DTM3_1**: Dead-time output of TOM0_13 $0E_H$ **CDTM0_DTM3_2**: Dead-time output of TOM0_14 $0F_H$ **CDTM0_DTM3_3**: Dead-time output of TOM0_15 10_H **CDTM0_DTM4_0**: Dead-time output of ATOM0_0 11_H **CDTM0_DTM4_1**: Dead-time output of ATOM0_1 12_H **CDTM0_DTM4_2**: Dead-time output of ATOM0_2 13_H **CDTM0_DTM4_3**: Dead-time output of ATOM0_3 14_H **CDTM0_DTM5_0**: Dead-time output of ATOM0_4 15_H **CDTM0_DTM5_1**: Dead-time output of ATOM0_5 16_H **CDTM0_DTM5_2**: Dead-time output of ATOM0_6 17_H **CDTM0_DTM5_3**: Dead-time output of ATOM0_7 18_H **CDTM2_DTM4_0**: Dead-time output of ATOM2_0 19_H **CDTM2_DTM4_1**: Dead-time output of ATOM2_1 $1A_H$ **CDTM2_DTM4_2**: Dead-time output of ATOM2_2 $1B_H$ **CDTM2_DTM4_3**: Dead-time output of ATOM2_3 $1C_H$ **CDTM2_DTM5_0**: Dead-time output of ATOM2_4 $1D_H$ **CDTM2_DTM5_1**: Dead-time output of ATOM2_5 $1E_H$ **CDTM2_DTM5_2**: Dead-time output of ATOM2_6 $1F_H$ **CDTM2_DTM5_3**: Dead-time output of ATOM2_7
0	31:29, 23:21, 15:13, 7:5	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.63 eGTM to MSC SETi output selection register j (j=0;i=1)

For a description of this register, see 7608.

EGTM_MSCSETi_CONj (i=1;j=0) Offset address: 061200_H+i*10_H+j*4

eGTM to MSC SETi output selection register j Kernel Reset value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0			SEL3					0			SEL2				
r			rw					r			rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			SEL1					0			SEL0				
r			rw					r			rw				

Field	Bits	Type	Description
SELx (x=0-3)	8*x+4:8*x	rw	**SETi.x signal output selection** 00_H **CDTM1_DTM0_0**: Dead-time output of TOM1_0 01_H **CDTM1_DTM0_1**: Dead-time output of TOM1_1 02_H **CDTM1_DTM0_2**: Dead-time output of TOM1_2 03_H **CDTM1_DTM0_3**: Dead-time output of TOM1_3 04_H **CDTM1_DTM1_0**: Dead-time output of TOM1_4 05_H **CDTM1_DTM1_1**: Dead-time output of TOM1_5 06_H **CDTM1_DTM1_2**: Dead-time output of TOM1_6 07_H **CDTM1_DTM1_3**: Dead-time output of TOM1_7 08_H **CDTM1_DTM2_0**: Dead-time output of TOM1_8 09_H **CDTM1_DTM2_1**: Dead-time output of TOM1_9 $0A_H$ **CDTM1_DTM2_2**: Dead-time output of TOM1_10 $0B_H$ **CDTM1_DTM2_3**: Dead-time output of TOM1_11 $0C_H$ **CDTM1_DTM3_0**: Dead-time output of TOM1_12 $0D_H$ **CDTM1_DTM3_1**: Dead-time output of TOM1_13 $0E_H$ **CDTM1_DTM3_2**: Dead-time output of TOM1_14 $0F_H$ **CDTM1_DTM3_3**: Dead-time output of TOM1_15 10_H **CDTM1_DTM4_0**: Dead-time output of ATOM1_0 11_H **CDTM1_DTM4_1**: Dead-time output of ATOM1_1 12_H **CDTM1_DTM4_2**: Dead-time output of ATOM1_2 13_H **CDTM1_DTM4_3**: Dead-time output of ATOM1_3 14_H **CDTM1_DTM5_0**: Dead-time output of ATOM1_4 15_H **CDTM1_DTM5_1**: Dead-time output of ATOM1_5 16_H **CDTM1_DTM5_2**: Dead-time output of ATOM1_6 17_H **CDTM1_DTM5_3**: Dead-time output of ATOM1_7 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	31:29, 23:21, 15:13, 7:5	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.64 eGTM to MSC SETi output selection register j (j=1;i=1)

For a description of this register, see 7608.

EGTM_MSCSETi_CONj (i=1;j=1) Offset address: 061200_H+i*10_H+j*4

eGTM to MSC SETi output selection register j Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0			SEL7					0			SEL6				
r			rw					r			rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			SEL5					0			SEL4				
r			rw					r			rw				

Field	Bits	Type	Description
SELx (x=4-7)	8*x-28:8*x-32	rw	**SETi.x signal output selection** 00_H **CDTM1_DTM0_0**: Dead-time output of TOM1_0 01_H **CDTM1_DTM0_1**: Dead-time output of TOM1_1 02_H **CDTM1_DTM0_2**: Dead-time output of TOM1_2 03_H **CDTM1_DTM0_3**: Dead-time output of TOM1_3 04_H **CDTM1_DTM1_0**: Dead-time output of TOM1_4 05_H **CDTM1_DTM1_1**: Dead-time output of TOM1_5 06_H **CDTM1_DTM1_2**: Dead-time output of TOM1_6 07_H **CDTM1_DTM1_3**: Dead-time output of TOM1_7 08_H **CDTM1_DTM2_0**: Dead-time output of TOM1_8 09_H **CDTM1_DTM2_1**: Dead-time output of TOM1_9 $0A_H$ **CDTM1_DTM2_2**: Dead-time output of TOM1_10 $0B_H$ **CDTM1_DTM2_3**: Dead-time output of TOM1_11 $0C_H$ **CDTM1_DTM3_0**: Dead-time output of TOM1_12 $0D_H$ **CDTM1_DTM3_1**: Dead-time output of TOM1_13 $0E_H$ **CDTM1_DTM3_2**: Dead-time output of TOM1_14 $0F_H$ **CDTM1_DTM3_3**: Dead-time output of TOM1_15 10_H **CDTM1_DTM4_0**: Dead-time output of ATOM1_0 11_H **CDTM1_DTM4_1**: Dead-time output of ATOM1_1 12_H **CDTM1_DTM4_2**: Dead-time output of ATOM1_2 13_H **CDTM1_DTM4_3**: Dead-time output of ATOM1_3 14_H **CDTM1_DTM5_0**: Dead-time output of ATOM1_4 15_H **CDTM1_DTM5_1**: Dead-time output of ATOM1_5 16_H **CDTM1_DTM5_2**: Dead-time output of ATOM1_6 17_H **CDTM1_DTM5_3**: Dead-time output of ATOM1_7 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	31:29, 23:21, 15:13, 7:5	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.65 eGTM to MSC SETi output selection register j (j=2;i=1)

For a description of this register, see 7608.

EGTM_MSCSETi_CONj (i=1;j=2) Offset address: 061200_H+i*10_H+j*4

eGTM to MSC SETi output selection register j Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0			SEL11					0			SEL10				
r			rw					r			rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			SEL9					0			SEL8				
r			rw					r			rw				

Field	Bits	Type	Description
SELx (x=8-11)	8*x-60:8*x-64	rw	**SETi.x signal output selection** 00_H **CDTM1_DTM0_0**: Dead-time output of TOM1_0 01_H **CDTM1_DTM0_1**: Dead-time output of TOM1_1 02_H **CDTM1_DTM0_2**: Dead-time output of TOM1_2 03_H **CDTM1_DTM0_3**: Dead-time output of TOM1_3 04_H **CDTM1_DTM1_0**: Dead-time output of TOM1_4 05_H **CDTM1_DTM1_1**: Dead-time output of TOM1_5 06_H **CDTM1_DTM1_2**: Dead-time output of TOM1_6 07_H **CDTM1_DTM1_3**: Dead-time output of TOM1_7 08_H **CDTM1_DTM2_0**: Dead-time output of TOM1_8 09_H **CDTM1_DTM2_1**: Dead-time output of TOM1_9 $0A_H$ **CDTM1_DTM2_2**: Dead-time output of TOM1_10 $0B_H$ **CDTM1_DTM2_3**: Dead-time output of TOM1_11 $0C_H$ **CDTM1_DTM3_0**: Dead-time output of TOM1_12 $0D_H$ **CDTM1_DTM3_1**: Dead-time output of TOM1_13 $0E_H$ **CDTM1_DTM3_2**: Dead-time output of TOM1_14 $0F_H$ **CDTM1_DTM3_3**: Dead-time output of TOM1_15 10_H **CDTM1_DTM4_0**: Dead-time output of ATOM1_0 11_H **CDTM1_DTM4_1**: Dead-time output of ATOM1_1 12_H **CDTM1_DTM4_2**: Dead-time output of ATOM1_2 13_H **CDTM1_DTM4_3**: Dead-time output of ATOM1_3 14_H **CDTM1_DTM5_0**: Dead-time output of ATOM1_4 15_H **CDTM1_DTM5_1**: Dead-time output of ATOM1_5 16_H **CDTM1_DTM5_2**: Dead-time output of ATOM1_6 17_H **CDTM1_DTM5_3**: Dead-time output of ATOM1_7 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	31:29, 23:21, 15:13, 7:5	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.66 eGTM to MSC SETi output selection register j (j=3;i=1)

For a description of this register, see 7608.

EGTM_MSCSETi_CONj (i=1;j=3) Offset address: 061200_H+i*10_H+j*4

eGTM to MSC SETi output selection register j Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0			SEL15					0			SEL14				
r			rw					r			rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			SEL13					0			SEL12				
r			rw					r			rw				

Field	Bits	Type	Description
SELx (x=12-15)	8*x-92:8*x-96	rw	**SETi.x signal output selection** 00_H **CDTM1_DTM0_0**: Dead-time output of TOM1_0 01_H **CDTM1_DTM0_1**: Dead-time output of TOM1_1 02_H **CDTM1_DTM0_2**: Dead-time output of TOM1_2 03_H **CDTM1_DTM0_3**: Dead-time output of TOM1_3 04_H **CDTM1_DTM1_0**: Dead-time output of TOM1_4 05_H **CDTM1_DTM1_1**: Dead-time output of TOM1_5 06_H **CDTM1_DTM1_2**: Dead-time output of TOM1_6 07_H **CDTM1_DTM1_3**: Dead-time output of TOM1_7 08_H **CDTM1_DTM2_0**: Dead-time output of TOM1_8 09_H **CDTM1_DTM2_1**: Dead-time output of TOM1_9 $0A_H$ **CDTM1_DTM2_2**: Dead-time output of TOM1_10 $0B_H$ **CDTM1_DTM2_3**: Dead-time output of TOM1_11 $0C_H$ **CDTM1_DTM3_0**: Dead-time output of TOM1_12 $0D_H$ **CDTM1_DTM3_1**: Dead-time output of TOM1_13 $0E_H$ **CDTM1_DTM3_2**: Dead-time output of TOM1_14 $0F_H$ **CDTM1_DTM3_3**: Dead-time output of TOM1_15 10_H **CDTM1_DTM4_0**: Dead-time output of ATOM1_0 11_H **CDTM1_DTM4_1**: Dead-time output of ATOM1_1 12_H **CDTM1_DTM4_2**: Dead-time output of ATOM1_2 13_H **CDTM1_DTM4_3**: Dead-time output of ATOM1_3 14_H **CDTM1_DTM5_0**: Dead-time output of ATOM1_4 15_H **CDTM1_DTM5_1**: Dead-time output of ATOM1_5 16_H **CDTM1_DTM5_2**: Dead-time output of ATOM1_6 17_H **CDTM1_DTM5_3**: Dead-time output of ATOM1_7 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	31:29, 23:21, 15:13, 7:5	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.67 eGTM to MSC SETi output selection register j (j=0;i=2)

For a description of this register, see 7608.

EGTM_MSCSETi_CONj (i=2;j=0) Offset address: 061200_H+i*10_H+j*4

eGTM to MSC SETi output selection register j Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0			SEL3					0			SEL2				
r			rw					r			rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			SEL1					0			SEL0				
r			rw					r			rw				

Field	Bits	Type	Description
SELx (x=0-3)	8*x+4:8*x	rw	**SETi.x signal output selection** 00_H **CDTM2_DTM0_0**: Dead-time output of TOM2_0 01_H **CDTM2_DTM0_1**: Dead-time output of TOM2_1 02_H **CDTM2_DTM0_2**: Dead-time output of TOM2_2 03_H **CDTM2_DTM0_3**: Dead-time output of TOM2_3 04_H **CDTM2_DTM1_0**: Dead-time output of TOM2_4 05_H **CDTM2_DTM1_1**: Dead-time output of TOM2_5 06_H **CDTM2_DTM1_2**: Dead-time output of TOM2_6 07_H **CDTM2_DTM1_3**: Dead-time output of TOM2_7 08_H **CDTM2_DTM2_0**: Dead-time output of TOM2_8 09_H **CDTM2_DTM2_1**: Dead-time output of TOM2_9 $0A_H$ **CDTM2_DTM2_2**: Dead-time output of TOM2_10 $0B_H$ **CDTM2_DTM2_3**: Dead-time output of TOM2_11 $0C_H$ **CDTM2_DTM3_0**: Dead-time output of TOM2_12 $0D_H$ **CDTM2_DTM3_1**: Dead-time output of TOM2_13 $0E_H$ **CDTM2_DTM3_2**: Dead-time output of TOM2_14 $0F_H$ **CDTM2_DTM3_3**: Dead-time output of TOM2_15 10_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	31:29, 23:21, 15:13, 7:5	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.68 eGTM to MSC SETi output selection register j (j=1;i=2)

For a description of this register, see 7608.

EGTM_MSCSETi_CONj (i=2;j=1) Offset address: 061200_H+i*10_H+j*4

eGTM to MSC SETi output selection register j Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0			SEL7					0			SEL6				
r			rw					r			rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			SEL5					0			SEL4				
r			rw					r			rw				

Field	Bits	Type	Description
SELx (x=4-7)	8*x-28:8*x-32	rw	**SETi.x signal output selection** 00_H **CDTM2_DTM0_0**: Dead-time output of TOM2_0 01_H **CDTM2_DTM0_1**: Dead-time output of TOM2_1 02_H **CDTM2_DTM0_2**: Dead-time output of TOM2_2 03_H **CDTM2_DTM0_3**: Dead-time output of TOM2_3 04_H **CDTM2_DTM1_0**: Dead-time output of TOM2_4 05_H **CDTM2_DTM1_1**: Dead-time output of TOM2_5 06_H **CDTM2_DTM1_2**: Dead-time output of TOM2_6 07_H **CDTM2_DTM1_3**: Dead-time output of TOM2_7 08_H **CDTM2_DTM2_0**: Dead-time output of TOM2_8 09_H **CDTM2_DTM2_1**: Dead-time output of TOM2_9 $0A_H$ **CDTM2_DTM2_2**: Dead-time output of TOM2_10 $0B_H$ **CDTM2_DTM2_3**: Dead-time output of TOM2_11 $0C_H$ **CDTM2_DTM3_0**: Dead-time output of TOM2_12 $0D_H$ **CDTM2_DTM3_1**: Dead-time output of TOM2_13 $0E_H$ **CDTM2_DTM3_2**: Dead-time output of TOM2_14 $0F_H$ **CDTM2_DTM3_3**: Dead-time output of TOM2_15 10_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	31:29, 23:21, 15:13, 7:5	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.69 eGTM to MSC SETi output selection register j (j=2;i=2)

For a description of this register, see 7608.

EGTM_MSCSETi_CONj (i=2;j=2) Offset address: 061200_H+i*10_H+j*4

eGTM to MSC SETi output selection register j Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0			SEL11					0			SEL10				
r			rw					r			rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			SEL9					0			SEL8				
r			rw					r			rw				

Field	Bits	Type	Description
SELx (x=8-11)	8*x-60:8*x-64	rw	**SETi.x signal output selection** 00_H **CDTM2_DTM0_0**: Dead-time output of TOM2_0 01_H **CDTM2_DTM0_1**: Dead-time output of TOM2_1 02_H **CDTM2_DTM0_2**: Dead-time output of TOM2_2 03_H **CDTM2_DTM0_3**: Dead-time output of TOM2_3 04_H **CDTM2_DTM1_0**: Dead-time output of TOM2_4 05_H **CDTM2_DTM1_1**: Dead-time output of TOM2_5 06_H **CDTM2_DTM1_2**: Dead-time output of TOM2_6 07_H **CDTM2_DTM1_3**: Dead-time output of TOM2_7 08_H **CDTM2_DTM2_0**: Dead-time output of TOM2_8 09_H **CDTM2_DTM2_1**: Dead-time output of TOM2_9 $0A_H$ **CDTM2_DTM2_2**: Dead-time output of TOM2_10 $0B_H$ **CDTM2_DTM2_3**: Dead-time output of TOM2_11 $0C_H$ **CDTM2_DTM3_0**: Dead-time output of TOM2_12 $0D_H$ **CDTM2_DTM3_1**: Dead-time output of TOM2_13 $0E_H$ **CDTM2_DTM3_2**: Dead-time output of TOM2_14 $0F_H$ **CDTM2_DTM3_3**: Dead-time output of TOM2_15 10_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	31:29, 23:21, 15:13, 7:5	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.70 eGTM to MSC SETi output selection register j (j=3;i=2)

For a description of this register, see 7608.

EGTM_MSCSETi_CONj (i=2;j=3) Offset address: 061200_H+i*10_H+j*4

eGTM to MSC SETi output selection register j Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0			SEL15					0			SEL14				
r			rw					r			rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			SEL13					0			SEL12				
r			rw					r			rw				

Field	Bits	Type	Description
SELx (x=12-15)	8*x-92:8*x-96	rw	**SETi.x signal output selection** 00_H **CDTM2_DTM0_0**: Dead-time output of TOM2_0 01_H **CDTM2_DTM0_1**: Dead-time output of TOM2_1 02_H **CDTM2_DTM0_2**: Dead-time output of TOM2_2 03_H **CDTM2_DTM0_3**: Dead-time output of TOM2_3 04_H **CDTM2_DTM1_0**: Dead-time output of TOM2_4 05_H **CDTM2_DTM1_1**: Dead-time output of TOM2_5 06_H **CDTM2_DTM1_2**: Dead-time output of TOM2_6 07_H **CDTM2_DTM1_3**: Dead-time output of TOM2_7 08_H **CDTM2_DTM2_0**: Dead-time output of TOM2_8 09_H **CDTM2_DTM2_1**: Dead-time output of TOM2_9 $0A_H$ **CDTM2_DTM2_2**: Dead-time output of TOM2_10 $0B_H$ **CDTM2_DTM2_3**: Dead-time output of TOM2_11 $0C_H$ **CDTM2_DTM3_0**: Dead-time output of TOM2_12 $0D_H$ **CDTM2_DTM3_1**: Dead-time output of TOM2_13 $0E_H$ **CDTM2_DTM3_2**: Dead-time output of TOM2_14 $0F_H$ **CDTM2_DTM3_3**: Dead-time output of TOM2_15 10_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	31:29, 23:21, 15:13, 7:5	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.71 eGTM to MSC SETi output selection register j (j=0;i=3)

For a description of this register, see 7608.

EGTM_MSCSETi_CONj (i=3;j=0) Offset address: 061200_H+i*10_H+j*4

eGTM to MSC SETi output selection register j Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0			SEL3					0			SEL2				
r			rw					r			rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			SEL1					0			SEL0				
r			rw					r			rw				

Field	Bits	Type	Description
SELx (x=0-3)	8*x+4:8*x	rw	**SETi.x signal output selection** 00_H **CDTM0_DTM4_0**: Dead-time output of ATOM0_0 01_H **CDTM0_DTM4_1**: Dead-time output of ATOM0_1 02_H **CDTM0_DTM4_2**: Dead-time output of ATOM0_2 03_H **CDTM0_DTM4_3**: Dead-time output of ATOM0_3 04_H **CDTM0_DTM5_0**: Dead-time output of ATOM0_4 05_H **CDTM0_DTM5_1**: Dead-time output of ATOM0_5 06_H **CDTM0_DTM5_2**: Dead-time output of ATOM0_6 07_H **CDTM0_DTM5_3**: Dead-time output of ATOM0_7 08_H **CDTM1_DTM4_0**: Dead-time output of ATOM1_0 09_H **CDTM1_DTM4_1**: Dead-time output of ATOM1_1 $0A_H$ **CDTM1_DTM4_2**: Dead-time output of ATOM1_2 $0B_H$ **CDTM1_DTM4_3**: Dead-time output of ATOM1_3 $0C_H$ **CDTM1_DTM5_0**: Dead-time output of ATOM1_4 $0D_H$ **CDTM1_DTM5_1**: Dead-time output of ATOM1_5 $0E_H$ **CDTM1_DTM5_2**: Dead-time output of ATOM1_6 $0F_H$ **CDTM1_DTM5_3**: Dead-time output of ATOM1_7 10_H **CDTM2_DTM4_0**: Dead-time output of ATOM2_0 11_H **CDTM2_DTM4_1**: Dead-time output of ATOM2_1 12_H **CDTM2_DTM4_2**: Dead-time output of ATOM2_2 13_H **CDTM2_DTM4_3**: Dead-time output of ATOM2_3 14_H **CDTM2_DTM5_0**: Dead-time output of ATOM2_4 15_H **CDTM2_DTM5_1**: Dead-time output of ATOM2_5 16_H **CDTM2_DTM5_2**: Dead-time output of ATOM2_6 17_H **CDTM2_DTM5_3**: Dead-time output of ATOM2_7 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	31:29, 23:21, 15:13, 7:5	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.72 eGTM to MSC SETi output selection register j (j=1;i=3)

For a description of this register, see 7608.

EGTM_MSCSETi_CONj (i=3;j=1) Offset address: 061200_H+i*10_H+j*4

eGTM to MSC SETi output selection register j Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0			SEL7					0			SEL6				
r			rw					r			rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			SEL5					0			SEL4				
r			rw					r			rw				

Field	Bits	Type	Description
SELx (x=4-7)	8*x-28:8*x-32	rw	**SETi.x signal output selection** 00_H **CDTM0_DTM4_0**: Dead-time output of ATOM0_0 01_H **CDTM0_DTM4_1**: Dead-time output of ATOM0_1 02_H **CDTM0_DTM4_2**: Dead-time output of ATOM0_2 03_H **CDTM0_DTM4_3**: Dead-time output of ATOM0_3 04_H **CDTM0_DTM5_0**: Dead-time output of ATOM0_4 05_H **CDTM0_DTM5_1**: Dead-time output of ATOM0_5 06_H **CDTM0_DTM5_2**: Dead-time output of ATOM0_6 07_H **CDTM0_DTM5_3**: Dead-time output of ATOM0_7 08_H **CDTM1_DTM4_0**: Dead-time output of ATOM1_0 09_H **CDTM1_DTM4_1**: Dead-time output of ATOM1_1 $0A_H$ **CDTM1_DTM4_2**: Dead-time output of ATOM1_2 $0B_H$ **CDTM1_DTM4_3**: Dead-time output of ATOM1_3 $0C_H$ **CDTM1_DTM5_0**: Dead-time output of ATOM1_4 $0D_H$ **CDTM1_DTM5_1**: Dead-time output of ATOM1_5 $0E_H$ **CDTM1_DTM5_2**: Dead-time output of ATOM1_6 $0F_H$ **CDTM1_DTM5_3**: Dead-time output of ATOM1_7 10_H **CDTM2_DTM4_0**: Dead-time output of ATOM2_0 11_H **CDTM2_DTM4_1**: Dead-time output of ATOM2_1 12_H **CDTM2_DTM4_2**: Dead-time output of ATOM2_2 13_H **CDTM2_DTM4_3**: Dead-time output of ATOM2_3 14_H **CDTM2_DTM5_0**: Dead-time output of ATOM2_4 15_H **CDTM2_DTM5_1**: Dead-time output of ATOM2_5 16_H **CDTM2_DTM5_2**: Dead-time output of ATOM2_6 17_H **CDTM2_DTM5_3**: Dead-time output of ATOM2_7 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	31:29, 23:21, 15:13, 7:5	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.73 eGTM to MSC SETi output selection register j (j=2;i=3)

For a description of this register, see 7608.

EGTM_MSCSETi_CONj (i=3;j=2) Offset address: 061200_H+i*10_H+j*4

eGTM to MSC SETi output selection register j Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0			SEL11					0			SEL10				
r			rw					r			rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			SEL9					0			SEL8				
r			rw					r			rw				

Field	Bits	Type	Description
SELx (x=8-11)	8*x-60:8*x-64	rw	**SETi.x signal output selection** 00_H **CDTM0_DTM4_0**: Dead-time output of ATOM0_0 01_H **CDTM0_DTM4_1**: Dead-time output of ATOM0_1 02_H **CDTM0_DTM4_2**: Dead-time output of ATOM0_2 03_H **CDTM0_DTM4_3**: Dead-time output of ATOM0_3 04_H **CDTM0_DTM5_0**: Dead-time output of ATOM0_4 05_H **CDTM0_DTM5_1**: Dead-time output of ATOM0_5 06_H **CDTM0_DTM5_2**: Dead-time output of ATOM0_6 07_H **CDTM0_DTM5_3**: Dead-time output of ATOM0_7 08_H **CDTM1_DTM4_0**: Dead-time output of ATOM1_0 09_H **CDTM1_DTM4_1**: Dead-time output of ATOM1_1 $0A_H$ **CDTM1_DTM4_2**: Dead-time output of ATOM1_2 $0B_H$ **CDTM1_DTM4_3**: Dead-time output of ATOM1_3 $0C_H$ **CDTM1_DTM5_0**: Dead-time output of ATOM1_4 $0D_H$ **CDTM1_DTM5_1**: Dead-time output of ATOM1_5 $0E_H$ **CDTM1_DTM5_2**: Dead-time output of ATOM1_6 $0F_H$ **CDTM1_DTM5_3**: Dead-time output of ATOM1_7 10_H **CDTM2_DTM4_0**: Dead-time output of ATOM2_0 11_H **CDTM2_DTM4_1**: Dead-time output of ATOM2_1 12_H **CDTM2_DTM4_2**: Dead-time output of ATOM2_2 13_H **CDTM2_DTM4_3**: Dead-time output of ATOM2_3 14_H **CDTM2_DTM5_0**: Dead-time output of ATOM2_4 15_H **CDTM2_DTM5_1**: Dead-time output of ATOM2_5 16_H **CDTM2_DTM5_2**: Dead-time output of ATOM2_6 17_H **CDTM2_DTM5_3**: Dead-time output of ATOM2_7 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	31:29, 23:21, 15:13, 7:5	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.74 eGTM to MSC SETi output selection register j (j=3;i=3)

For a description of this register, see 7608.

EGTM_MSCSETi_CONj (i=3;j=3) Offset address: 061200_H+i*10_H+j*4

eGTM to MSC SETi output selection register j Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0			SEL15					0			SEL14				
r			rw					r			rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			SEL13					0			SEL12				
r			rw					r			rw				

Field	Bits	Type	Description
SELx (x=12-15)	8*x-92:8*x-96	rw	**SETi.x signal output selection** 00_H **CDTM0_DTM4_0**: Dead-time output of ATOM0_0 01_H **CDTM0_DTM4_1**: Dead-time output of ATOM0_1 02_H **CDTM0_DTM4_2**: Dead-time output of ATOM0_2 03_H **CDTM0_DTM4_3**: Dead-time output of ATOM0_3 04_H **CDTM0_DTM5_0**: Dead-time output of ATOM0_4 05_H **CDTM0_DTM5_1**: Dead-time output of ATOM0_5 06_H **CDTM0_DTM5_2**: Dead-time output of ATOM0_6 07_H **CDTM0_DTM5_3**: Dead-time output of ATOM0_7 08_H **CDTM1_DTM4_0**: Dead-time output of ATOM1_0 09_H **CDTM1_DTM4_1**: Dead-time output of ATOM1_1 $0A_H$ **CDTM1_DTM4_2**: Dead-time output of ATOM1_2 $0B_H$ **CDTM1_DTM4_3**: Dead-time output of ATOM1_3 $0C_H$ **CDTM1_DTM5_0**: Dead-time output of ATOM1_4 $0D_H$ **CDTM1_DTM5_1**: Dead-time output of ATOM1_5 $0E_H$ **CDTM1_DTM5_2**: Dead-time output of ATOM1_6 $0F_H$ **CDTM1_DTM5_3**: Dead-time output of ATOM1_7 10_H **CDTM2_DTM4_0**: Dead-time output of ATOM2_0 11_H **CDTM2_DTM4_1**: Dead-time output of ATOM2_1 12_H **CDTM2_DTM4_2**: Dead-time output of ATOM2_2 13_H **CDTM2_DTM4_3**: Dead-time output of ATOM2_3 14_H **CDTM2_DTM5_0**: Dead-time output of ATOM2_4 15_H **CDTM2_DTM5_1**: Dead-time output of ATOM2_5 16_H **CDTM2_DTM5_2**: Dead-time output of ATOM2_6 17_H **CDTM2_DTM5_3**: Dead-time output of ATOM2_7 18_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	31:29, 23:21, 15:13, 7:5	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.75 eGTM to input bus MSCn_INLCON signal selection register (n=0)

For a description of this register, see 7610.

EGTM_MSCn_INLCON (n=0) Offset address: 061290_H+n*10_H

eGTM to input bus MSCn_INLCON signal selection register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SEL15		SEL14		SEL13		SEL12		SEL11		SEL10		SEL9		SEL8	
rw		rw		rw		rw		rw		rw		rw		rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SEL7		SEL6		SEL5		SEL4		SEL3		SEL2		SEL1		SEL0	
rw		rw		rw		rw		rw		rw		rw		rw	

Field	Bits	Type	Description
SELx (x=0-15)	2*x+1:2*x	rw	**nth bus line INLCONx SETi signal selection** 00_B **SET0**: Outputs SET0[15:0] selected 01_B **SET1**: Outputs SET1[15:0] selected 10_B **SET2**: Outputs SET2[15:0] selected 11_B **SET3**: Outputs SET3[15:0] selected

44.9.4.76 eGTM to input bus MSCn_INLCON signal selection register (n=1)

For a description of this register, see 7610.

EGTM_MSCn_INLCON (n=1)

Offset address: 061290_H+n*10_H

eGTM to input bus MSCn_INLCON signal selection register

Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SEL15		SEL14		SEL13		SEL12		SEL11		SEL10		SEL9		SEL8	
rw		rw		rw		rw		rw		rw		rw		rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SEL7		SEL6		SEL5		SEL4		SEL3		SEL2		SEL1		SEL0	
rw		rw		rw		rw		rw		rw		rw		rw	

Field	Bits	Type	Description
SELx (x=0-15)	2*x+1:2*x	rw	**nth bus line INLCONx SETi signal selection** 00_B **SET0**: Outputs SET0[15:0] selected 01_B **SET1**: Outputs SET1[15:0] selected 10_B **SET2**: Outputs SET2[15:0] selected 11_B **SET3**: Outputs SET3[15:0] selected

44.9.4.77 eGTM to input bus MSCn_INHCON signal selection register (n=0)

For a description of this register, see 7611.

EGTM_MSCn_INHCON (n=0)

Offset address: 061294_H+n*10_H

eGTM to input bus MSCn_INHCON signal selection register

Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SEL15		SEL14		SEL13		SEL12		SEL11		SEL10		SEL9		SEL8	
rw		rw		rw		rw		rw		rw		rw		rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SEL7		SEL6		SEL5		SEL4		SEL3		SEL2		SEL1		SEL0	
rw		rw		rw		rw		rw		rw		rw		rw	

Field	Bits	Type	Description
SELx (x=0-15)	2*x+1:2*x	rw	**nth bus line INHCON.x SETi signal selection** 00_B **SET0**: Outputs SET0[15:0] selected 01_B **SET1**: Outputs SET1[15:0] selected 10_B **SET2**: Outputs SET2[15:0] selected 11_B **SET3**: Outputs SET3[15:0] selected

44.9.4.78 eGTM to input bus MSCn_INHCON signal selection register (n=1)

For a description of this register, see 7611.

EGTM_MSCn_INHCON (n=1)

Offset address: 061294_H+n*10_H

eGTM to input bus MSCn_INHCON signal selection register

Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SEL15		SEL14		SEL13		SEL12		SEL11		SEL10		SEL9		SEL8	
rw		rw		rw		rw		rw		rw		rw		rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SEL7		SEL6		SEL5		SEL4		SEL3		SEL2		SEL1		SEL0	
rw		rw		rw		rw		rw		rw		rw		rw	

Field	Bits	Type	Description
SELx (x=0-15)	2*x+1:2*x	rw	**nth bus line INHCON.x SETi signal selection** 00_B **SET0**: Outputs SET0[15:0] selected 01_B **SET1**: Outputs SET1[15:0] selected 10_B **SET2**: Outputs SET2[15:0] selected 11_B **SET3**: Outputs SET3[15:0] selected

44.9.4.79 eGTM to input bus MSCn_INLECON signal selection register (n=0)

For a description of this register, see 7610.

EGTM_MSCn_INLECON (n=0)

Offset address: 061298_H+n*10_H

eGTM to input bus MSCn_INLECON signal selection register

Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SEL15		SEL14		SEL13		SEL12		SEL11		SEL10		SEL9		SEL8	
rw		rw		rw		rw		rw		rw		rw		rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SEL7		SEL6		SEL5		SEL4		SEL3		SEL2		SEL1		SEL0	
rw		rw		rw		rw		rw		rw		rw		rw	

Field	Bits	Type	Description
SELx (x=0-15)	2*x+1:2*x	rw	**nth bus line INLECON.x SETi signal selection** 00_B **SET0**: Outputs SET0[15:0] selected 01_B **SET1**: Outputs SET1[15:0] selected 10_B **SET2**: Outputs SET2[15:0] selected 11_B **SET3**: Outputs SET3[15:0] selected

44.9.4.80 eGTM to input bus MSCn_INLECON signal selection register (n=1)

For a description of this register, see 7610.

EGTM_MSCn_INLECON (n=1)

Offset address: 061298_H+n*10_H

eGTM to input bus MSCn_INLECON signal selection register

Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SEL15		SEL14		SEL13		SEL12		SEL11		SEL10		SEL9		SEL8	
rw		rw		rw		rw		rw		rw		rw		rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SEL7		SEL6		SEL5		SEL4		SEL3		SEL2		SEL1		SEL0	
rw		rw		rw		rw		rw		rw		rw		rw	

Field	Bits	Type	Description
SELx (x=0-15)	2*x+1:2*x	rw	**nth bus line INLECON.x SETi signal selection** 00_B **SET0**: Outputs SET0[15:0] selected 01_B **SET1**: Outputs SET1[15:0] selected 10_B **SET2**: Outputs SET2[15:0] selected 11_B **SET3**: Outputs SET3[15:0] selected

44.9.4.81 eGTM to input bus MSCn_INHECON signal selection register (n=0)

For a description of this register, see 7611.

EGTM_MSCn_INHECON (n=0)

Offset address: $06129C_H+n*10_H$

eGTM to input bus MSCn_INHECON signal selection register

Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SEL15		SEL14		SEL13		SEL12		SEL11		SEL10		SEL9		SEL8	
rw		rw		rw		rw		rw		rw		rw		rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SEL7		SEL6		SEL5		SEL4		SEL3		SEL2		SEL1		SEL0	
rw		rw		rw		rw		rw		rw		rw		rw	

Field	Bits	Type	Description
SELx (x=0-15)	2*x+1:2*x	rw	**nth bus line INHECON.x SETi signal selection** 00_B **SET0**: Outputs SET0[15:0] selected 01_B **SET1**: Outputs SET1[15:0] selected 10_B **SET2**: Outputs SET2[15:0] selected 11_B **SET3**: Outputs SET3[15:0] selected

44.9.4.82 eGTM to input bus MSCn_INHECON signal selection register (n=1)

For a description of this register, see 7611.

EGTM_MSCn_INHECON (n=1) Offset address: $06129C_H+n*10_H$

eGTM to input bus MSCn_INHECON signal selection register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SEL15		SEL14		SEL13		SEL12		SEL11		SEL10		SEL9		SEL8	
rw		rw		rw		rw		rw		rw		rw		rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SEL7		SEL6		SEL5		SEL4		SEL3		SEL2		SEL1		SEL0	
rw		rw		rw		rw		rw		rw		rw		rw	

Field	Bits	Type	Description
SELx (x=0-15)	2*x+1:2*x	rw	**nth bus line INHECON.x SETi signal selection** 00_B **SET0**: Outputs SET0[15:0] selected 01_B **SET1**: Outputs SET1[15:0] selected 10_B **SET2**: Outputs SET2[15:0] selected 11_B **SET3**: Outputs SET3[15:0] selected

44.9.4.83 eGTM to PSI5 output trigger selection register

For a description of this register, see 7615.

EGTM_PSI5_OUT Offset address: 061400_H

eGTM to PSI5 output trigger selection register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								SEL5				SEL4			
r								rw				rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SEL3				SEL2				SEL1				SEL0			
rw				rw				rw				rw			

Field	Bits	Type	Description
SELx (x=0-5)	4*x+3:4*x	rw	**eGTM to PSI5 trigger x output selection** 0_H **None**: Not connected (safe state) 1_H **CDTM2_DTM1_2**: Dead-time output of TOM2_6 2_H **CDTM2_DTM1_3**: Dead-time output of TOM2_7 3_H **CDTM2_DTM3_1**: Dead-time output of TOM2_13 4_H **CDTM2_DTM3_2**: Dead-time output of TOM2_14 5_H **CDTM2_DTM5_0**: Dead-time output of ATOM2_4 6_H **CDTM2_DTM5_1**: Dead-time output of ATOM2_5 7_H **CDTM2_DTM5_2**: Dead-time output of ATOM2_6 8_H **CDTM2_DTM5_3**: Dead-time output of ATOM2_7 9_H **CDTM0_DTM1_2**: Dead-time output of TOM0_6 A_H **CDTM0_DTM1_3**: Dead-time output of TOM0_7 B_H **CDTM0_DTM3_1**: Dead-time output of TOM0_13 C_H **CDTM0_DTM3_2**: Dead-time output of TOM0_14 D_H **Reserved**: Reserved, do not use … F_H **Reserved**: Reserved, do not use
0	31:24	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.84 eGTM to PSI5-Sn output trigger selection register (n=0-1)

For a description of this register, see 7616.

EGTM_PSI5Sn_OUT (n=0-1) Offset address: 061410_H+n*4

eGTM to PSI5-Sn output trigger selection register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SEL7				SEL6				SEL5				SEL4			
rw				rw				rw				rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SEL3				SEL2				SEL1				SEL0			
rw				rw				rw				rw			

Field	Bits	Type	Description
SELx (x=0)	4*x+3:4*x	rw	**eGTM to PSI5-Sn trigger x output selection** 0_H **None**: Not connected (safe state) 1_H **CDTM0_DTM5_0**: Dead-time output of ATOM0_4 2_H **CDTM1_DTM5_0**: Dead-time output of ATOM1_4 3_H **CDTM2_DTM5_0**: Dead-time output of ATOM2_4 4_H **CDTM0_DTM2_0**: Dead-time output of TOM0_8 5_H **CDTM1_DTM2_0**: Dead-time output of TOM1_8 6_H **Reserved**: Reserved, do not use 7_H **Reserved**: Reserved, do not use 8_H **CDTM2_DTM2_0**: Dead-time output of TOM2_8 9_H **Reserved**: Reserved, do not use … F_H **Reserved**: Reserved, do not use
SELx (x=1)	4*x+3:4*x	rw	**eGTM to PSI5-Sn trigger x output selection** 0_H **None**: Not connected (safe state) 1_H **CDTM0_DTM5_1**: Dead-time output of ATOM0_5 2_H **CDTM1_DTM5_1**: Dead-time output of ATOM1_5 3_H **CDTM2_DTM5_1**: Dead-time output of ATOM2_5 4_H **CDTM0_DTM2_1**: Dead-time output of TOM0_9 5_H **CDTM1_DTM2_1**: Dead-time output of TOM1_9 6_H **Reserved**: Reserved, do not use 7_H **Reserved**: Reserved, do not use 8_H **CDTM2_DTM2_1**: Dead-time output of TOM2_9 9_H **Reserved**: Reserved, do not use … F_H **Reserved**: Reserved, do not use
SELx (x=2)	4*x+3:4*x	rw	**eGTM to PSI5-Sn trigger x output selection** 0_H **None**: Not connected (safe state) 1_H **CDTM0_DTM5_2**: Dead-time output of ATOM0_6 2_H **CDTM1_DTM5_2**: Dead-time output of ATOM1_6 3_H **CDTM2_DTM5_2**: Dead-time output of ATOM2_6 4_H **CDTM0_DTM2_2**: Dead-time output of TOM0_10 5_H **CDTM1_DTM2_2**: Dead-time output of TOM1_10 6_H **Reserved**: Reserved, do not use 7_H **Reserved**: Reserved, do not use 8_H **CDTM2_DTM2_2**: Dead-time output of TOM2_10 9_H **Reserved**: Reserved, do not use … F_H **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=3)	4*x+3:4*x	rw	**eGTM to PSI5-Sn trigger x output selection** 0_H **None**: Not connected (safe state) 1_H **CDTM0_DTM5_3**: Dead-time output of ATOM0_7 2_H **CDTM1_DTM5_3**: Dead-time output of ATOM1_7 3_H **CDTM2_DTM5_3**: Dead-time output of ATOM2_7 4_H **CDTM0_DTM2_3**: Dead-time output of TOM0_11 5_H **CDTM1_DTM2_3**: Dead-time output of TOM1_11 6_H **Reserved**: Reserved, do not use 7_H **Reserved**: Reserved, do not use 8_H **CDTM2_DTM2_3**: Dead-time output of TOM2_11 9_H **Reserved**: Reserved, do not use … F_H **Reserved**: Reserved, do not use
SELx (x=4)	4*x+3:4*x	rw	**eGTM to PSI5-Sn trigger x output selection** 0_H **None**: Not connected (safe state) 1_H **CDTM0_DTM5_0**: Dead-time output of ATOM0_4 2_H **CDTM1_DTM5_0**: Dead-time output of ATOM1_4 3_H **CDTM2_DTM5_0**: Dead-time output of ATOM2_4 4_H **CDTM0_DTM3_0**: Dead-time output of TOM0_12 5_H **CDTM1_DTM3_0**: Dead-time output of TOM1_12 6_H **Reserved**: Reserved, do not use 7_H **Reserved**: Reserved, do not use 8_H **CDTM2_DTM3_0**: Dead-time output of TOM2_12 9_H **Reserved**: Reserved, do not use … F_H **Reserved**: Reserved, do not use
SELx (x=5)	4*x+3:4*x	rw	**eGTM to PSI5-Sn trigger x output selection** 0_H **None**: Not connected (safe state) 1_H **CDTM0_DTM5_1**: Dead-time output of ATOM0_5 2_H **CDTM1_DTM5_1**: Dead-time output of ATOM1_5 3_H **CDTM2_DTM5_1**: Dead-time output of ATOM2_5 4_H **CDTM0_DTM3_1**: Dead-time output of TOM0_13 5_H **CDTM1_DTM3_1**: Dead-time output of TOM1_13 6_H **Reserved**: Reserved, do not use 7_H **Reserved**: Reserved, do not use 8_H **CDTM2_DTM3_1**: Dead-time output of TOM2_13 9_H **Reserved**: Reserved, do not use … F_H **Reserved**: Reserved, do not use

(table continues...)

(continued)

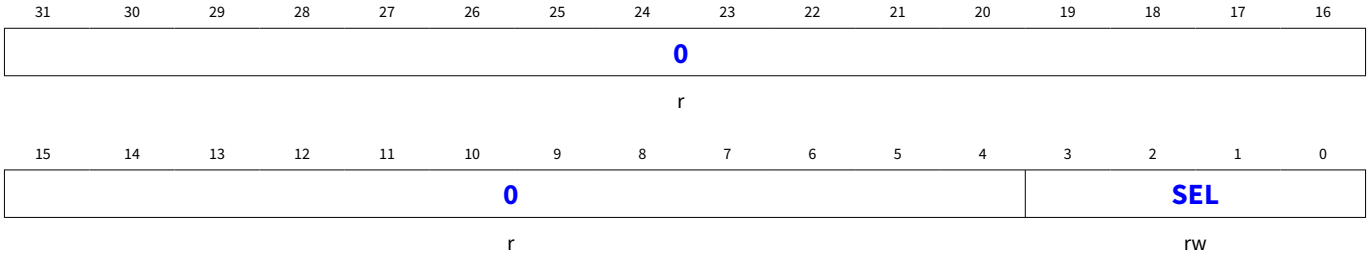
Field	Bits	Type	Description
SELx (x=6)	4*x+3:4*x	rw	**eGTM to PSI5-Sn trigger x output selection** 0_H **None**: Not connected (safe state) 1_H **CDTM0_DTM5_2**: Dead-time output of ATOM0_6 2_H **CDTM1_DTM5_2**: Dead-time output of ATOM1_6 3_H **CDTM2_DTM5_2**: Dead-time output of ATOM2_6 4_H **CDTM0_DTM3_2**: Dead-time output of TOM0_14 5_H **CDTM1_DTM3_2**: Dead-time output of TOM1_14 6_H **Reserved**: Reserved, do not use 7_H **Reserved**: Reserved, do not use 8_H **CDTM2_DTM3_2**: Dead-time output of TOM2_14 9_H **Reserved**: Reserved, do not use … F_H **Reserved**: Reserved, do not use
SELx (x=7)	4*x+3:4*x	rw	**eGTM to PSI5-Sn trigger x output selection** 0_H **None**: Not connected (safe state) 1_H **CDTM0_DTM5_3**: Dead-time output of ATOM0_7 2_H **CDTM1_DTM5_3**: Dead-time output of ATOM1_7 3_H **CDTM2_DTM5_3**: Dead-time output of ATOM2_7 4_H **CDTM0_DTM3_3**: Dead-time output of TOM0_15 5_H **CDTM1_DTM3_3**: Dead-time output of TOM1_15 6_H **Reserved**: Reserved, do not use 7_H **Reserved**: Reserved, do not use 8_H **CDTM2_DTM3_3**: Dead-time output of TOM2_15 9_H **Reserved**: Reserved, do not use … F_H **Reserved**: Reserved, do not use

44.9.4.85 eGTM to LCDCDC output trigger selection register

For a description of this register, see 7598.

EGTM_LCDCDC_OUT Offset address: 061420_H

eGTM to LCDCDC output trigger selection register Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
SEL	3:0	rw	**eGTM to LCDCDC trigger output selection** 0_H **None**: Not connected (safe state) 1_H **CDTM0_DTM4_1**: Dead-time output of ATOM0_1 2_H **CDTM1_DTM4_1**: Dead-time output of ATOM1_1 3_H **CDTM2_DTM4_1**: Dead-time output of ATOM2_1 4_H **CDTM0_DTM0_1**: Dead-time output of TOM0_1 5_H **CDTM1_DTM0_1**: Dead-time output of TOM1_1 6_H **Reserved**: Reserved, do not use … B_H **Reserved**: Reserved, do not use C_H **CDTM2_DTM0_1**: Dead-time output of TOM2_1 D_H **Reserved**: Reserved, do not use … F_H **Reserved**: Reserved, do not use
0	31:4	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.86 eGTM to CANn output selection register x. n*4+ x<24 , if interface exists. (x=0;n=0)

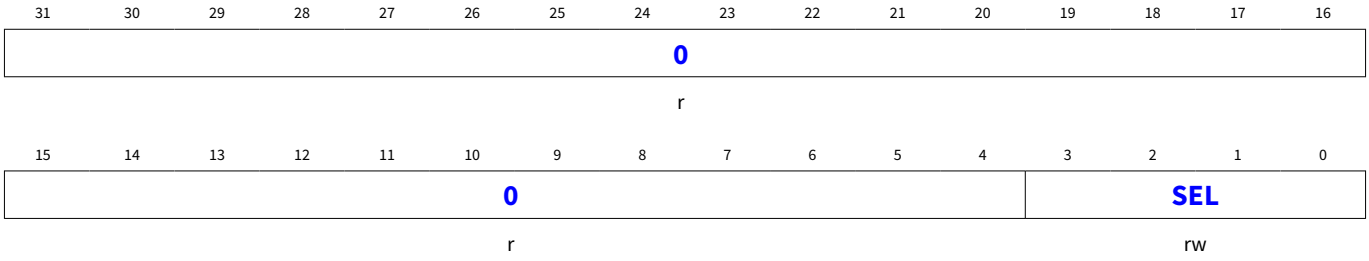
For a description of this register, see 7599.

EGTM_CANn_OUTx (n=0;x=0)

Offset address: 061480_H+n*10_H+x*4

eGTM to CANn output selection register x. n*4+ x<24 , if interface exists.

Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
SEL	3:0	rw	**eGTM to CANn trigger x output selection** 0_H **None**: Not connected (safe state) 1_H **CDTM0_DTM5_3**: Dead-time output of ATOM0_7 2_H **CDTM1_DTM5_3**: Dead-time output of ATOM1_7 3_H **ATOM2_7**: Output of ATOM2_7 4_H **CDTM0_DTM3_3**: Dead-time output of TOM0_15 5_H **CDTM1_DTM3_3**: Dead-time output of TOM1_15 6_H **TOM2_15**: Output of TOM2_15 7_H **Reserved**: Reserved, do not use … F_H **Reserved**: Reserved, do not use
0	31:4	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.87 eGTM to CANn output selection register x. n*4+ x<24 , if interface exists. (x=1;n=0)

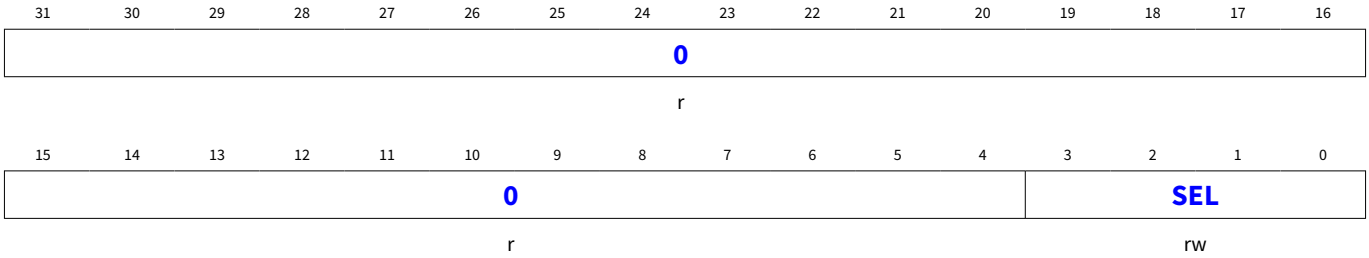
For a description of this register, see 7599.

EGTM_CANn_OUTx (n=0;x=1)

Offset address: 061480_H+n*10_H+x*4

eGTM to CANn output selection register x. n*4+ x<24 , if interface exists.

Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
SEL	3:0	rw	**eGTM to CANn trigger x output selection** 0_H **None**: Not connected (safe state) 1_H **CDTM0_DTM5_2**: Dead-time output of ATOM0_6 2_H **CDTM1_DTM5_2**: Dead-time output of ATOM1_6 3_H **ATOM2_6**: Output of ATOM2_6 4_H **CDTM0_DTM3_2**: Dead-time output of TOM0_14 5_H **CDTM1_DTM3_2**: Dead-time output of TOM1_14 6_H **TOM2_14**: Output of TOM2_14 7_H **Reserved**: Reserved, do not use … F_H **Reserved**: Reserved, do not use
0	31:4	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.88 eGTM to CANn output selection register x. n*4+ x<24 , if interface exists. (x=2;n=0)

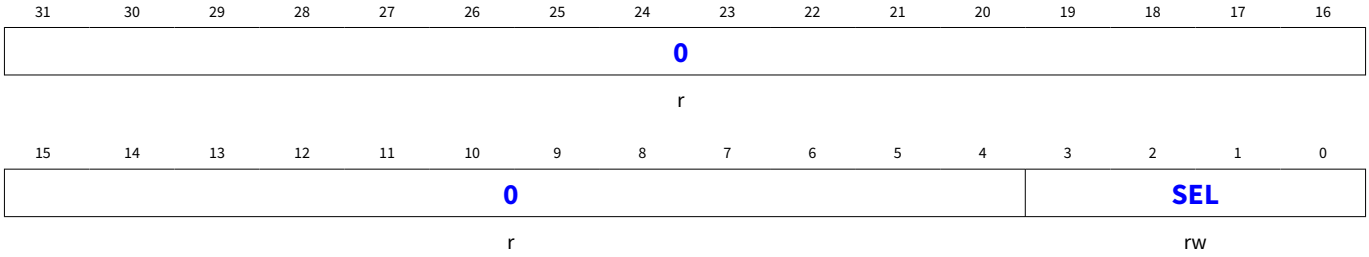
For a description of this register, see 7599.

EGTM_CANn_OUTx (n=0;x=2)

Offset address: 061480_H+n*10_H+x*4

eGTM to CANn output selection register x. n*4+ x<24 , if interface exists.

Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
SEL	3:0	rw	**eGTM to CANn trigger x output selection** 0_H **None**: Not connected (safe state) 1_H **CDTM0_DTM5_1**: Dead-time output of ATOM0_5 2_H **CDTM1_DTM5_1**: Dead-time output of ATOM1_5 3_H **ATOM2_5**: Output of ATOM2_5 4_H **CDTM0_DTM3_1**: Dead-time output of TOM0_13 5_H **CDTM1_DTM3_1**: Dead-time output of TOM1_13 6_H **TOM2_13**: Output of TOM2_13 7_H **Reserved**: Reserved, do not use … F_H **Reserved**: Reserved, do not use
0	31:4	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.89 eGTM to CANn output selection register x. n*4+ x<24 , if interface exists. (x=3;n=0)

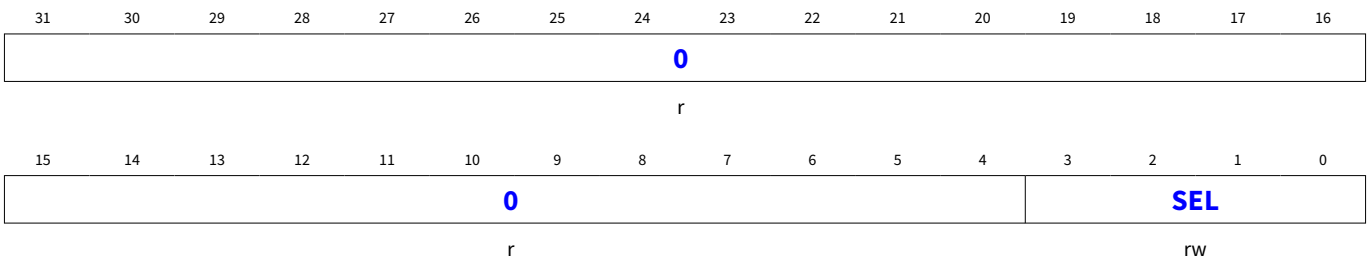
For a description of this register, see 7599.

EGTM_CANn_OUTx (n=0;x=3)

Offset address: 061480_H+n*10_H+x*4

eGTM to CANn output selection register x. n*4+ x<24 , if interface exists.

Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
SEL	3:0	rw	**eGTM to CANn trigger x output selection** 0_H **None**: Not connected (safe state) 1_H **CDTM0_DTM5_0**: Dead-time output of ATOM0_4 2_H **CDTM1_DTM5_0**: Dead-time output of ATOM1_4 3_H **ATOM2_4**: Output of ATOM2_4 4_H **CDTM0_DTM3_0**: Dead-time output of TOM0_12 5_H **CDTM1_DTM3_0**: Dead-time output of TOM1_12 6_H **TOM2_12**: Output of TOM2_12 7_H **Reserved**: Reserved, do not use … F_H **Reserved**: Reserved, do not use
0	31:4	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.90 eGTM to CANn output selection register x. n*4+ x<24 , if interface exists. (x=0;n=1)

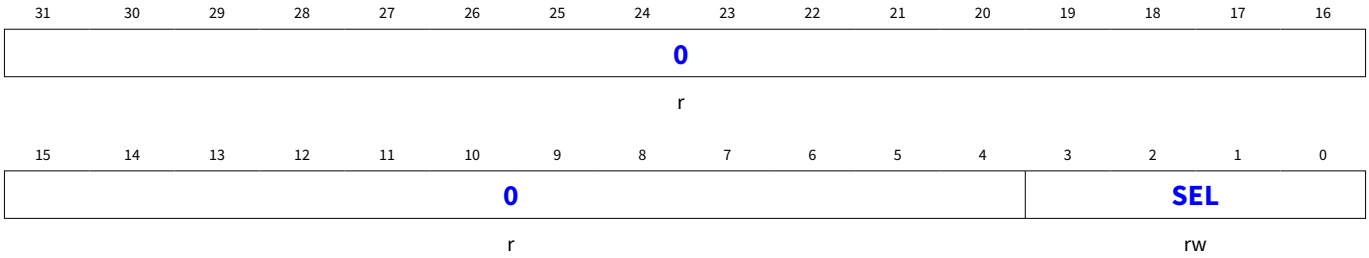
For a description of this register, see 7599.

EGTM_CANn_OUTx (n=1;x=0)

Offset address: 061480_H+n*10_H+x*4

eGTM to CANn output selection register x. n*4+ x<24 , if interface exists.

Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
SEL	3:0	rw	**eGTM to CANn trigger x output selection** 0_H **None**: Not connected (safe state) 1_H **CDTM0_DTM4_3**: Dead-time output of ATOM0_3 2_H **CDTM1_DTM4_3**: Dead-time output of ATOM1_3 3_H **ATOM2_3**: Output of ATOM2_3 4_H **CDTM0_DTM2_3**: Dead-time output of TOM0_11 5_H **CDTM1_DTM2_3**: Dead-time output of TOM1_11 6_H **TOM2_11**: Output of TOM2_11 7_H **Reserved**: Reserved, do not use … F_H **Reserved**: Reserved, do not use
0	31:4	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.91 eGTM to CANn output selection register x. n*4+ x<24 , if interface exists. (x=1;n=1)

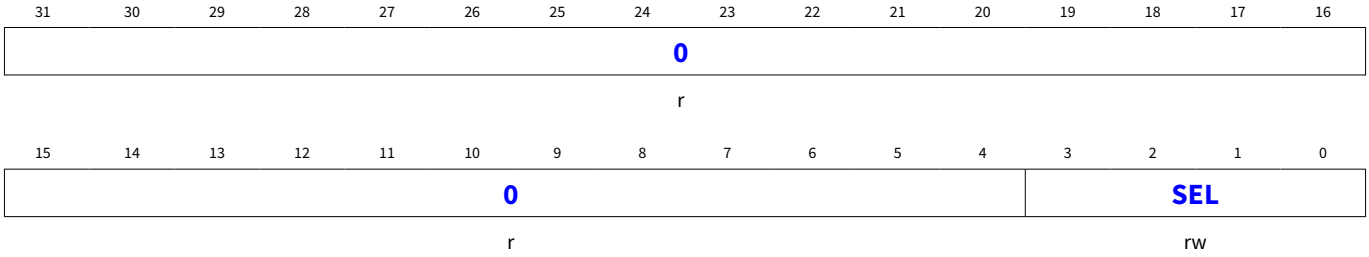
For a description of this register, see 7599.

EGTM_CANn_OUTx (n=1;x=1)

Offset address: 061480_H+n*10_H+x*4

eGTM to CANn output selection register x. n*4+ x<24 , if interface exists.

Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
SEL	3:0	rw	**eGTM to CANn trigger x output selection** 0_H **None**: Not connected (safe state) 1_H **CDTM0_DTM4_2**: Dead-time output of ATOM0_2 2_H **CDTM1_DTM4_2**: Dead-time output of ATOM1_2 3_H **ATOM2_2**: Output of ATOM2_2 4_H **CDTM0_DTM2_2**: Dead-time output of TOM0_10 5_H **CDTM1_DTM2_2**: Dead-time output of TOM1_10 6_H **TOM2_10**: Output of TOM2_10 7_H **Reserved**: Reserved, do not use … F_H **Reserved**: Reserved, do not use
0	31:4	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.92 eGTM to CANn output selection register x. n*4+ x<24 , if interface exists. (x=2;n=1)

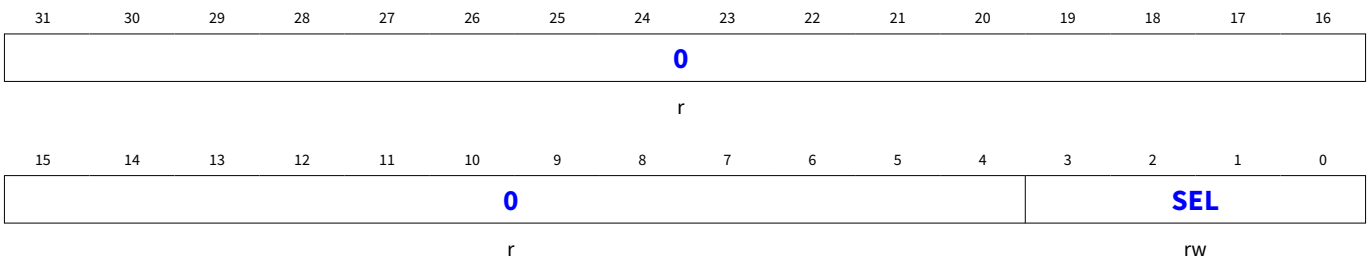
For a description of this register, see 7599.

EGTM_CANn_OUTx (n=1;x=2)

Offset address: 061480_H+n*10_H+x*4

eGTM to CANn output selection register x. n*4+ x<24 , if interface exists.

Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
SEL	3:0	rw	**eGTM to CANn trigger x output selection** 0_H **None**: Not connected (safe state) 1_H **CDTM0_DTM4_1**: Dead-time output of ATOM0_1 2_H **CDTM1_DTM4_1**: Dead-time output of ATOM1_1 3_H **ATOM2_1**: Output of ATOM2_1 4_H **CDTM0_DTM2_1**: Dead-time output of TOM0_9 5_H **CDTM1_DTM2_1**: Dead-time output of TOM1_9 6_H **TOM2_9**: Output of TOM2_9 7_H **Reserved**: Reserved, do not use … F_H **Reserved**: Reserved, do not use
0	31:4	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.93 eGTM to CANn output selection register x. n*4+ x<24 , if interface exists. (x=3;n=1)

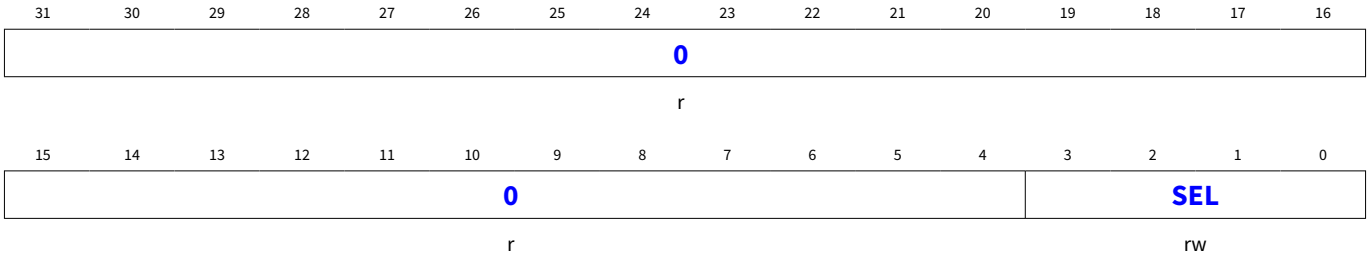
For a description of this register, see 7599.

EGTM_CANn_OUTx (n=1;x=3)

Offset address: 061480_H+n*10_H+x*4

eGTM to CANn output selection register x. n*4+ x<24 , if interface exists.

Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
SEL	3:0	rw	**eGTM to CANn trigger x output selection** 0_H **None**: Not connected (safe state) 1_H **CDTM0_DTM4_0**: Dead-time output of ATOM0_0 2_H **CDTM1_DTM4_0**: Dead-time output of ATOM1_0 3_H **ATOM2_0**: Output of ATOM2_0 4_H **CDTM0_DTM2_0**: Dead-time output of TOM0_8 5_H **CDTM1_DTM2_0**: Dead-time output of TOM1_8 6_H **TOM2_8**: Output of TOM2_8 7_H **Reserved**: Reserved, do not use … F_H **Reserved**: Reserved, do not use
0	31:4	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.94 eGTM to CANn output selection register x. n*4+ x<24 , if interface exists. (x=0;n=2)

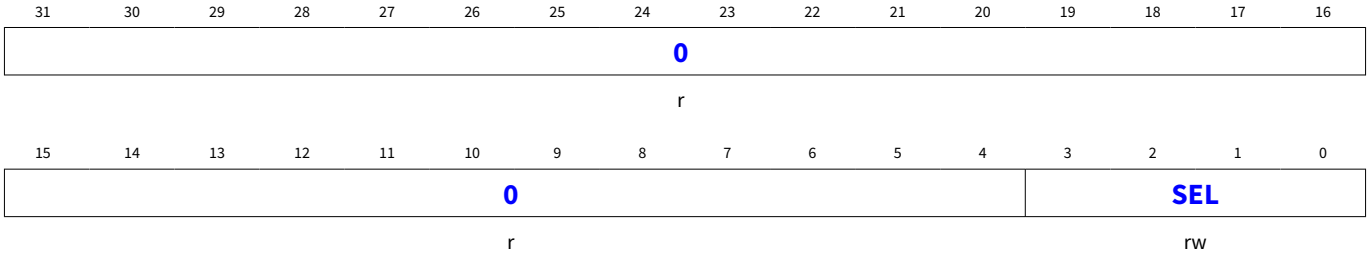
For a description of this register, see 7599.

EGTM_CANn_OUTx (n=2;x=0)

Offset address: 061480_H+n*10_H+x*4

eGTM to CANn output selection register x. n*4+ x<24 , if interface exists.

Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
SEL	3:0	rw	**eGTM to CANn trigger x output selection** 0_H **None**: Not connected (safe state) 1_H **CDTM0_DTM5_3**: Dead-time output of ATOM0_7 2_H **CDTM1_DTM5_3**: Dead-time output of ATOM1_7 3_H **ATOM2_7**: Output of ATOM2_7 4_H **CDTM0_DTM3_3**: Dead-time output of TOM0_15 5_H **CDTM1_DTM3_3**: Dead-time output of TOM1_15 6_H **TOM2_15**: Output of TOM2_15 7_H **Reserved**: Reserved, do not use … F_H **Reserved**: Reserved, do not use
0	31:4	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.95 eGTM to CANn output selection register x. n*4+ x<24 , if interface exists. (x=1;n=2)

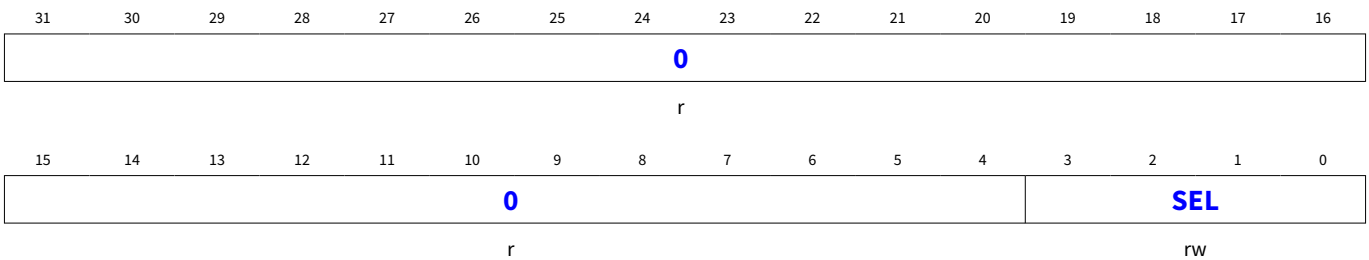
For a description of this register, see 7599.

EGTM_CANn_OUTx (n=2;x=1)

Offset address: 061480_H+n*10_H+x*4

eGTM to CANn output selection register x. n*4+ x<24 , if interface exists.

Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
SEL	3:0	rw	**eGTM to CANn trigger x output selection** 0_H **None**: Not connected (safe state) 1_H **CDTM0_DTM5_2**: Dead-time output of ATOM0_6 2_H **CDTM1_DTM5_2**: Dead-time output of ATOM1_6 3_H **ATOM2_6**: Output of ATOM2_6 4_H **CDTM0_DTM3_2**: Dead-time output of TOM0_14 5_H **CDTM1_DTM3_2**: Dead-time output of TOM1_14 6_H **TOM2_14**: Output of TOM2_14 7_H **Reserved**: Reserved, do not use … F_H **Reserved**: Reserved, do not use
0	31:4	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.96 eGTM to CANn output selection register x. n*4+ x<24 , if interface exists. (x=2;n=2)

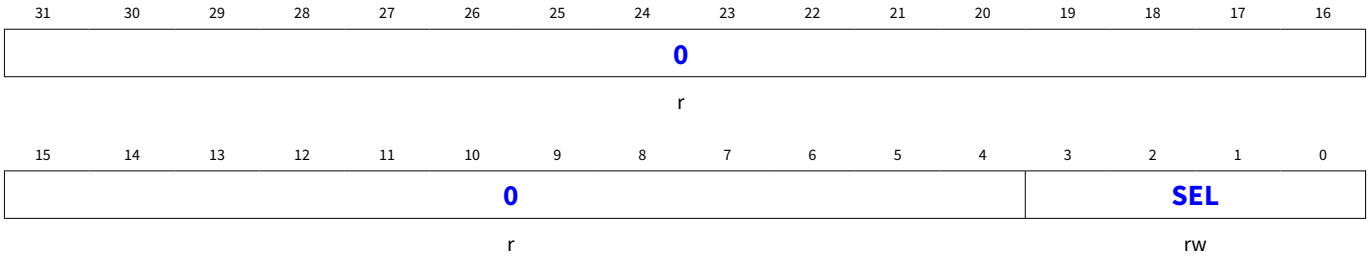
For a description of this register, see 7599.

EGTM_CANn_OUTx (n=2;x=2)

Offset address: 061480_H+n*10_H+x*4

eGTM to CANn output selection register x. n*4+ x<24 , if interface exists.

Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
SEL	3:0	rw	**eGTM to CANn trigger x output selection** 0_H **None**: Not connected (safe state) 1_H **CDTM0_DTM5_1**: Dead-time output of ATOM0_5 2_H **CDTM1_DTM5_1**: Dead-time output of ATOM1_5 3_H **ATOM2_5**: Output of ATOM2_5 4_H **CDTM0_DTM3_1**: Dead-time output of TOM0_13 5_H **CDTM1_DTM3_1**: Dead-time output of TOM1_13 6_H **TOM2_13**: Output of TOM2_13 7_H **Reserved**: Reserved, do not use … F_H **Reserved**: Reserved, do not use
0	31:4	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.97 eGTM to CANn output selection register x. n*4+ x<24 , if interface exists. (x=3;n=2)

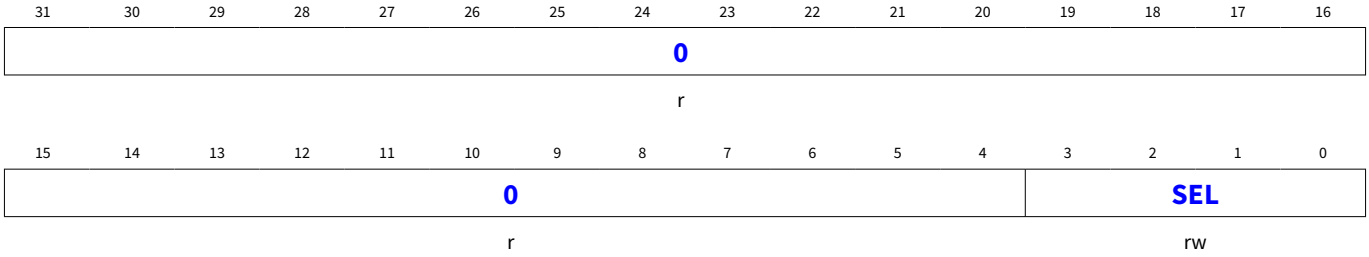
For a description of this register, see 7599.

EGTM_CANn_OUTx (n=2;x=3)

Offset address: 061480_H+n*10_H+x*4

eGTM to CANn output selection register x. n*4+ x<24 , if interface exists.

Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
SEL	3:0	rw	**eGTM to CANn trigger x output selection** 0_H **None**: Not connected (safe state) 1_H **CDTM0_DTM5_0**: Dead-time output of ATOM0_4 2_H **CDTM1_DTM5_0**: Dead-time output of ATOM1_4 3_H **ATOM2_4**: Output of ATOM2_4 4_H **CDTM0_DTM3_0**: Dead-time output of TOM0_12 5_H **CDTM1_DTM3_0**: Dead-time output of TOM1_12 6_H **TOM2_12**: Output of TOM2_12 7_H **Reserved**: Reserved, do not use … F_H **Reserved**: Reserved, do not use
0	31:4	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.98 eGTM to CANn output selection register x. n*4+ x<24 , if interface exists. (x=0;n=3)

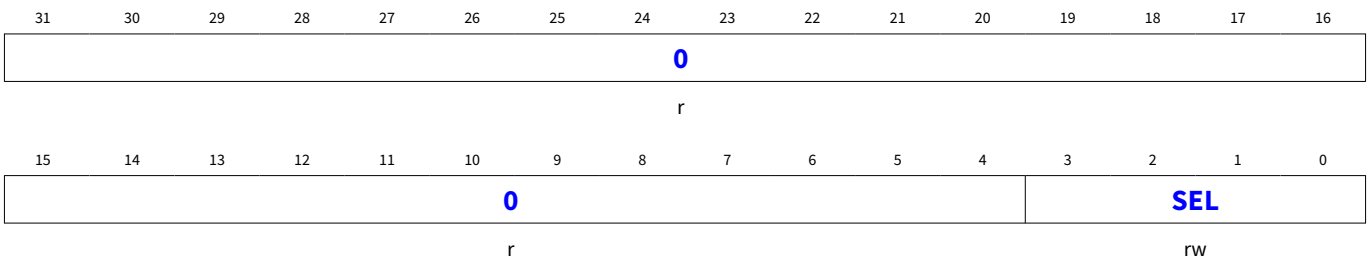
For a description of this register, see 7599.

EGTM_CANn_OUTx (n=3;x=0)

Offset address: 061480_H+n*10_H+x*4

eGTM to CANn output selection register x. n*4+ x<24 , if interface exists.

Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
SEL	3:0	rw	**eGTM to CANn trigger x output selection** 0_H **None**: Not connected (safe state) 1_H **CDTM0_DTM4_3**: Dead-time output of ATOM0_3 2_H **CDTM1_DTM4_3**: Dead-time output of ATOM1_3 3_H **ATOM2_3**: Output of ATOM2_3 4_H **CDTM0_DTM2_3**: Dead-time output of TOM0_11 5_H **CDTM1_DTM2_3**: Dead-time output of TOM1_11 6_H **TOM2_11**: Output of TOM2_11 7_H **Reserved**: Reserved, do not use … F_H **Reserved**: Reserved, do not use
0	31:4	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.99 eGTM to CANn output selection register x. n*4+ x<24 , if interface exists. (x=1;n=3)

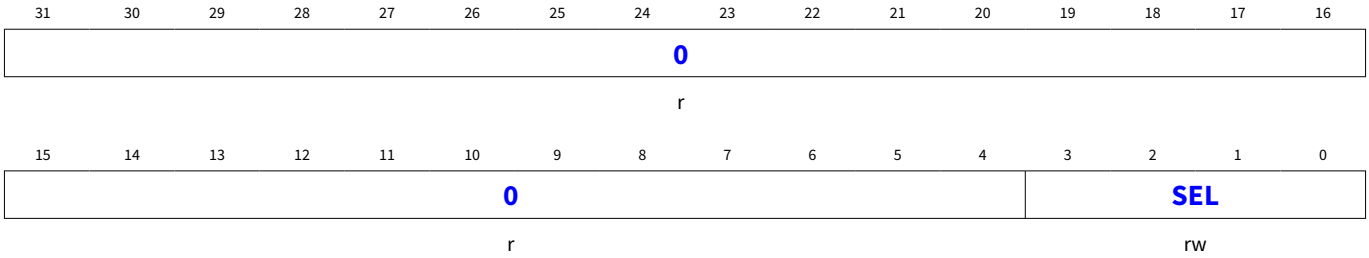
For a description of this register, see 7599.

EGTM_CANn_OUTx (n=3;x=1)

eGTM to CANn output selection register x. n*4+ x<24 , if interface exists.

Offset address: 061480_H+n*10_H+x*4

Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
SEL	3:0	rw	**eGTM to CANn trigger x output selection** 0_H **None**: Not connected (safe state) 1_H **CDTM0_DTM4_2**: Dead-time output of ATOM0_2 2_H **CDTM1_DTM4_2**: Dead-time output of ATOM1_2 3_H **ATOM2_2**: Output of ATOM2_2 4_H **CDTM0_DTM2_2**: Dead-time output of TOM0_10 5_H **CDTM1_DTM2_2**: Dead-time output of TOM1_10 6_H **TOM2_10**: Output of TOM2_10 7_H **Reserved**: Reserved, do not use … F_H **Reserved**: Reserved, do not use
0	31:4	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.100 eGTM to CANn output selection register x. n*4+ x<24 , if interface exists. (x=2;n=3)

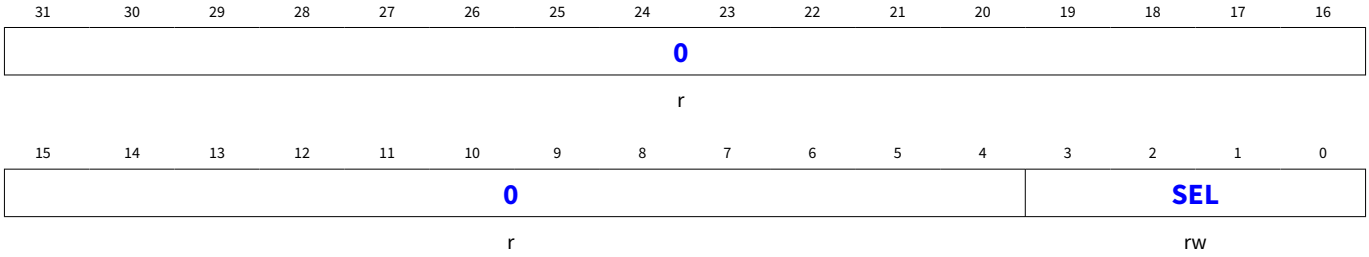
For a description of this register, see 7599.

EGTM_CANn_OUTx (n=3;x=2)

Offset address: 061480_H+n*10_H+x*4

eGTM to CANn output selection register x. n*4+ x<24 , if interface exists.

Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
SEL	3:0	rw	**eGTM to CANn trigger x output selection** 0_H **None**: Not connected (safe state) 1_H **CDTM0_DTM4_1**: Dead-time output of ATOM0_1 2_H **CDTM1_DTM4_1**: Dead-time output of ATOM1_1 3_H **ATOM2_1**: Output of ATOM2_1 4_H **CDTM0_DTM2_1**: Dead-time output of TOM0_9 5_H **CDTM1_DTM2_1**: Dead-time output of TOM1_9 6_H **TOM2_9**: Output of TOM2_9 7_H **Reserved**: Reserved, do not use … F_H **Reserved**: Reserved, do not use
0	31:4	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.101 eGTM to CANn output selection register x. n*4+ x<24 , if interface exists. (x=3;n=3)

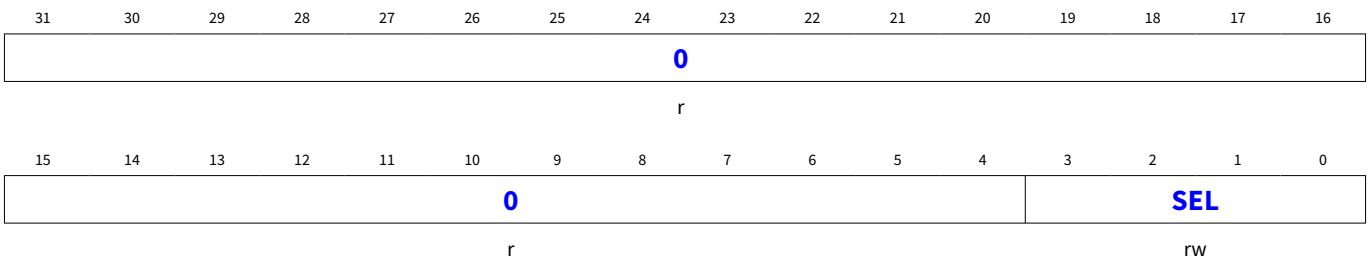
For a description of this register, see 7599.

EGTM_CANn_OUTx (n=3;x=3)

Offset address: 061480_H+n*10_H+x*4

eGTM to CANn output selection register x. n*4+ x<24 , if interface exists.

Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
SEL	3:0	rw	**eGTM to CANn trigger x output selection** 0_H **None**: Not connected (safe state) 1_H **CDTM0_DTM4_0**: Dead-time output of ATOM0_0 2_H **CDTM1_DTM4_0**: Dead-time output of ATOM1_0 3_H **ATOM2_0**: Output of ATOM2_0 4_H **CDTM0_DTM2_0**: Dead-time output of TOM0_8 5_H **CDTM1_DTM2_0**: Dead-time output of TOM1_8 6_H **TOM2_8**: Output of TOM2_8 7_H **Reserved**: Reserved, do not use … F_H **Reserved**: Reserved, do not use
0	31:4	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.102 eGTM to CANn output selection register x. n*4+ x<24 , if interface exists. (x=0;n=4)

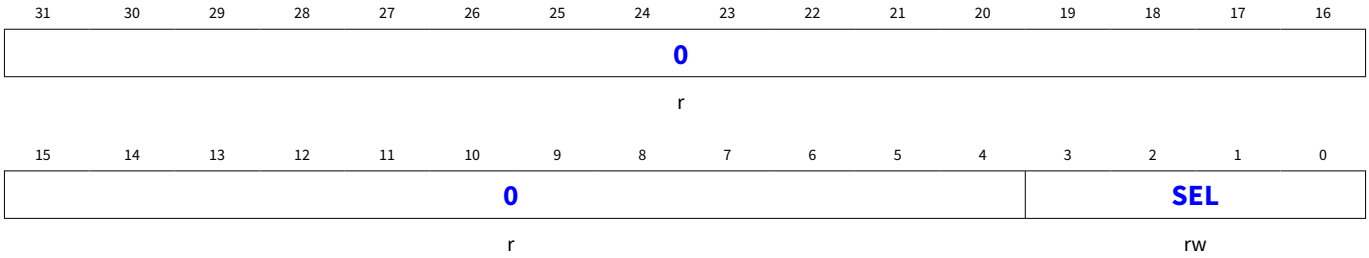
For a description of this register, see 7599.

EGTM_CANn_OUTx (n=4;x=0)

Offset address: 061480_H+n*10_H+x*4

eGTM to CANn output selection register x. n*4+ x<24 , if interface exists.

Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
SEL	3:0	rw	**eGTM to CANn trigger x output selection** 0_H **None**: Not connected (safe state) 1_H **CDTM0_DTM5_3**: Dead-time output of ATOM0_7 2_H **CDTM1_DTM5_3**: Dead-time output of ATOM1_7 3_H **ATOM2_7**: Output of ATOM2_7 4_H **CDTM0_DTM3_3**: Dead-time output of TOM0_15 5_H **CDTM1_DTM3_3**: Dead-time output of TOM1_15 6_H **TOM2_15**: Output of TOM2_15 7_H **Reserved**: Reserved, do not use … F_H **Reserved**: Reserved, do not use
0	31:4	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.103 eGTM to CANn output selection register x. n*4+ x<24 , if interface exists. (x=1;n=4)

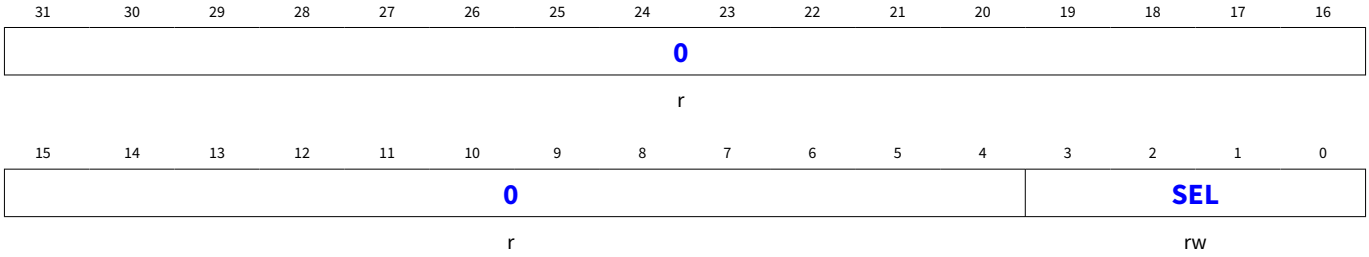
For a description of this register, see 7599.

EGTM_CANn_OUTx (n=4;x=1)

Offset address: 061480_H+n*10_H+x*4

eGTM to CANn output selection register x. n*4+ x<24 , if interface exists.

Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
SEL	3:0	rw	**eGTM to CANn trigger x output selection** 0_H **None**: Not connected (safe state) 1_H **CDTM0_DTM5_2**: Dead-time output of ATOM0_6 2_H **CDTM1_DTM5_2**: Dead-time output of ATOM1_6 3_H **ATOM2_6**: Output of ATOM2_6 4_H **CDTM0_DTM3_2**: Dead-time output of TOM0_14 5_H **CDTM1_DTM3_2**: Dead-time output of TOM1_14 6_H **TOM2_14**: Output of TOM2_14 7_H **Reserved**: Reserved, do not use … F_H **Reserved**: Reserved, do not use
0	31:4	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.104 eGTM to CANn output selection register x. n*4+ x<24 , if interface exists. (x=2;n=4)

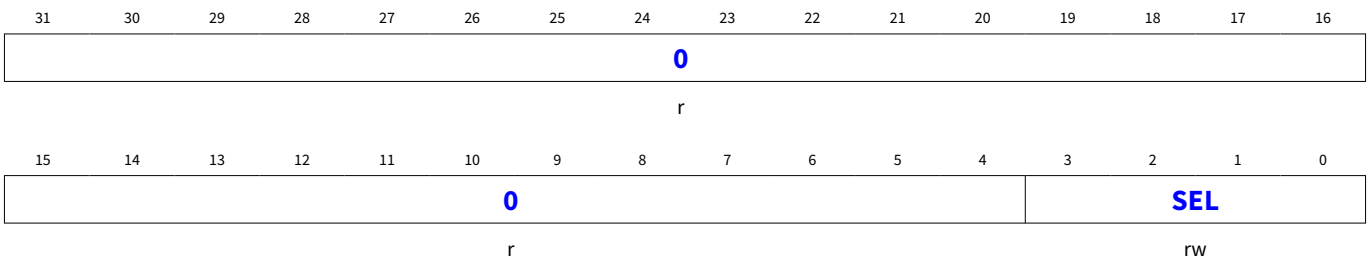
For a description of this register, see 7599.

EGTM_CANn_OUTx (n=4;x=2)

Offset address: 061480_H+n*10_H+x*4

eGTM to CANn output selection register x. n*4+ x<24 , if interface exists.

Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
SEL	3:0	rw	**eGTM to CANn trigger x output selection** 0_H **None**: Not connected (safe state) 1_H **CDTM0_DTM5_1**: Dead-time output of ATOM0_5 2_H **CDTM1_DTM5_1**: Dead-time output of ATOM1_5 3_H **ATOM2_5**: Output of ATOM2_5 4_H **CDTM0_DTM3_1**: Dead-time output of TOM0_13 5_H **CDTM1_DTM3_1**: Dead-time output of TOM1_13 6_H **TOM2_13**: Output of TOM2_13 7_H **Reserved**: Reserved, do not use … F_H **Reserved**: Reserved, do not use
0	31:4	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.105 eGTM to CANn output selection register x. n*4+ x<24 , if interface exists. (x=3;n=4)

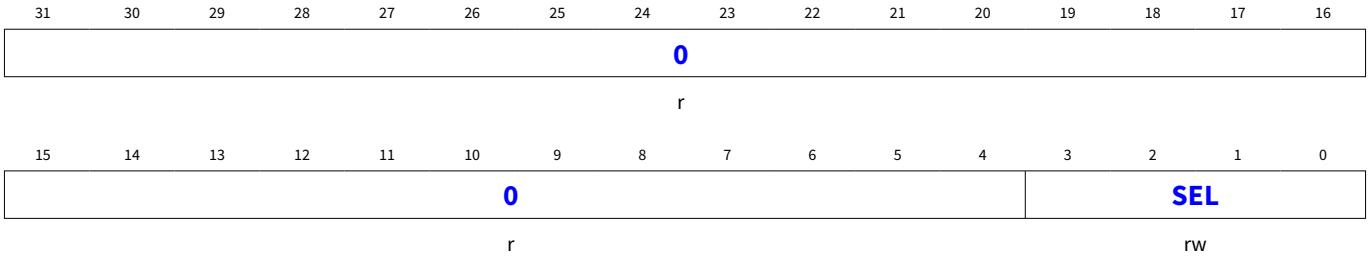
For a description of this register, see 7599.

EGTM_CANn_OUTx (n=4;x=3)

Offset address: 061480_H+n*10_H+x*4

eGTM to CANn output selection register x. n*4+ x<24 , if interface exists.

Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
SEL	3:0	rw	**eGTM to CANn trigger x output selection** 0_H **None**: Not connected (safe state) 1_H **CDTM0_DTM5_0**: Dead-time output of ATOM0_4 2_H **CDTM1_DTM5_0**: Dead-time output of ATOM1_4 3_H **ATOM2_4**: Output of ATOM2_4 4_H **CDTM0_DTM3_0**: Dead-time output of TOM0_12 5_H **CDTM1_DTM3_0**: Dead-time output of TOM1_12 6_H **TOM2_12**: Output of TOM2_12 7_H **Reserved**: Reserved, do not use … F_H **Reserved**: Reserved, do not use
0	31:4	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.106 eGTM to CANn output selection register x. n*4+ x<24 , if interface exists. (x=0;n=5)

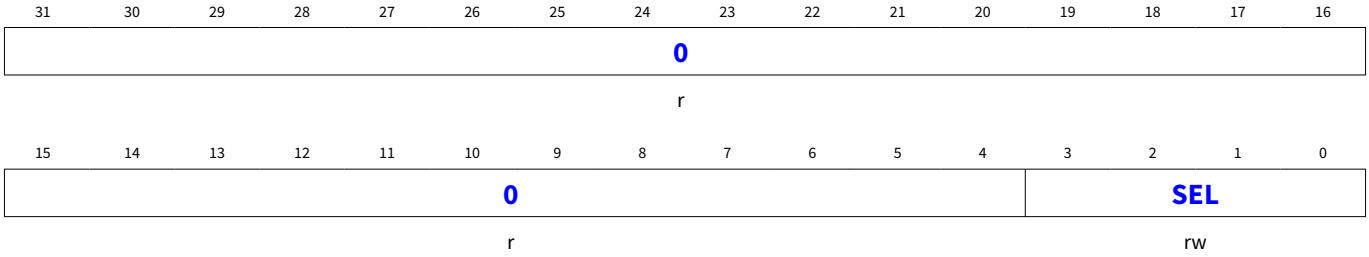
For a description of this register, see 7599.

EGTM_CANn_OUTx (n=5;x=0)

Offset address: 061480_H+n*10_H+x*4

eGTM to CANn output selection register x. n*4+ x<24 , if interface exists.

Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
SEL	3:0	rw	**eGTM to CANn trigger x output selection** 0_H **None**: Not connected (safe state) 1_H **CDTM0_DTM4_3**: Dead-time output of ATOM0_3 2_H **CDTM1_DTM4_3**: Dead-time output of ATOM1_3 3_H **ATOM2_3**: Output of ATOM2_3 4_H **CDTM0_DTM2_3**: Dead-time output of TOM0_11 5_H **CDTM1_DTM2_3**: Dead-time output of TOM1_11 6_H **TOM2_11**: Output of TOM2_11 7_H **Reserved**: Reserved, do not use … F_H **Reserved**: Reserved, do not use
0	31:4	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.107 eGTM to CANn output selection register x. n*4+ x<24 , if interface exists. (x=1;n=5)

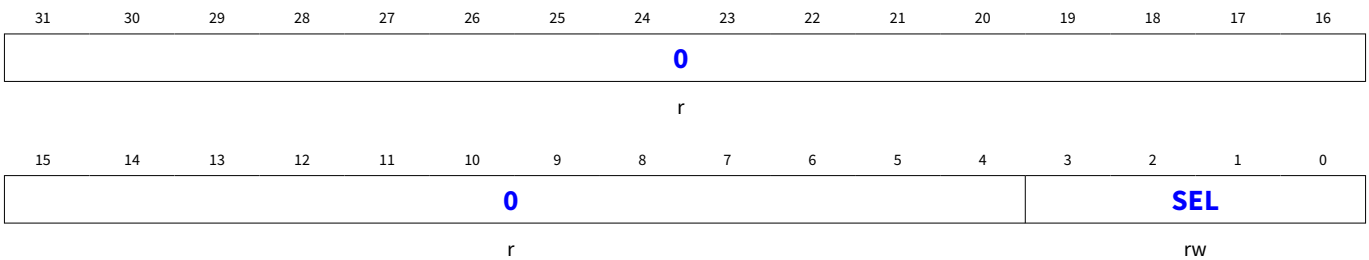
For a description of this register, see 7599.

EGTM_CANn_OUTx (n=5;x=1)

Offset address: 061480_H+n*10_H+x*4

eGTM to CANn output selection register x. n*4+ x<24 , if interface exists.

Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
SEL	3:0	rw	**eGTM to CANn trigger x output selection** 0_H **None**: Not connected (safe state) 1_H **CDTM0_DTM4_2**: Dead-time output of ATOM0_2 2_H **CDTM1_DTM4_2**: Dead-time output of ATOM1_2 3_H **ATOM2_2**: Output of ATOM2_2 4_H **CDTM0_DTM2_2**: Dead-time output of TOM0_10 5_H **CDTM1_DTM2_2**: Dead-time output of TOM1_10 6_H **TOM2_10**: Output of TOM2_10 7_H **Reserved**: Reserved, do not use … F_H **Reserved**: Reserved, do not use
0	31:4	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.108 eGTM to CANn output selection register x. n*4+ x<24 , if interface exists. (x=2;n=5)

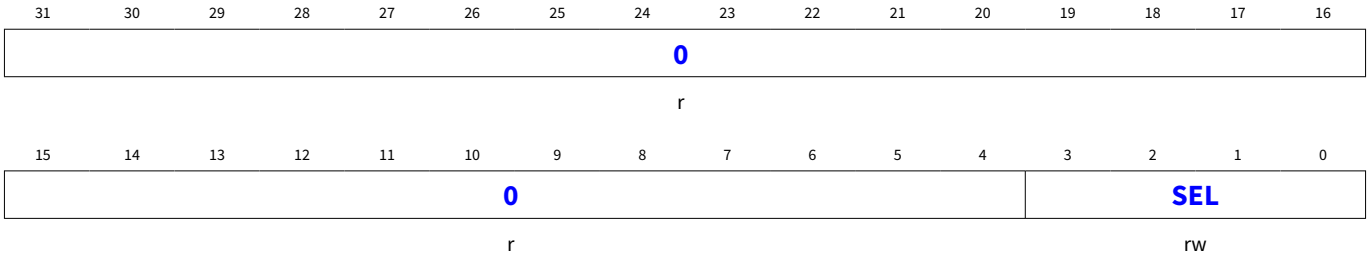
For a description of this register, see 7599.

EGTM_CANn_OUTx (n=5;x=2)

eGTM to CANn output selection register x. n*4+ x<24 , if interface exists.

Offset address: 061480_H+n*10_H+x*4

Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
SEL	3:0	rw	**eGTM to CANn trigger x output selection** 0_H **None**: Not connected (safe state) 1_H **CDTM0_DTM4_1**: Dead-time output of ATOM0_1 2_H **CDTM1_DTM4_1**: Dead-time output of ATOM1_1 3_H **ATOM2_1**: Output of ATOM2_1 4_H **CDTM0_DTM2_1**: Dead-time output of TOM0_9 5_H **CDTM1_DTM2_1**: Dead-time output of TOM1_9 6_H **TOM2_9**: Output of TOM2_9 7_H **Reserved**: Reserved, do not use … F_H **Reserved**: Reserved, do not use
0	31:4	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.109 eGTM to CANn output selection register x. n*4+ x<24 , if interface exists. (x=3;n=5)

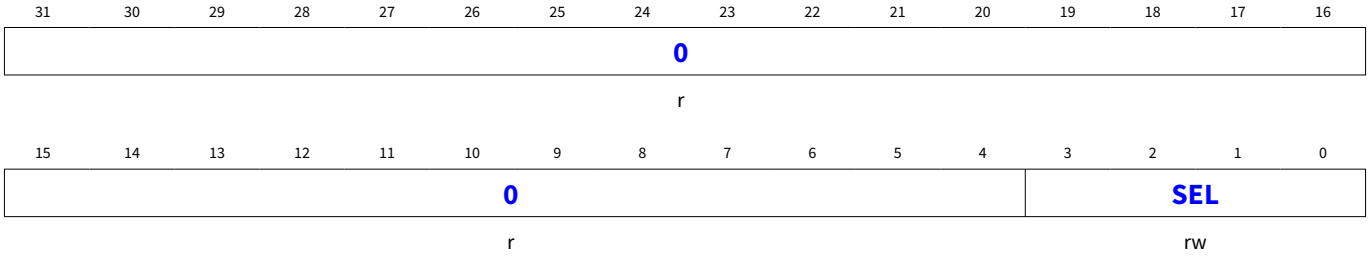
For a description of this register, see 7599.

EGTM_CANn_OUTx (n=5;x=3)

Offset address: 061480_H+n*10_H+x*4

eGTM to CANn output selection register x. n*4+ x<24 , if interface exists.

Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
SEL	3:0	rw	**eGTM to CANn trigger x output selection** 0_H **None**: Not connected (safe state) 1_H **CDTM0_DTM4_0**: Dead-time output of ATOM0_0 2_H **CDTM1_DTM4_0**: Dead-time output of ATOM1_0 3_H **ATOM2_0**: Output of ATOM2_0 4_H **CDTM0_DTM2_0**: Dead-time output of TOM0_8 5_H **CDTM1_DTM2_0**: Dead-time output of TOM1_8 6_H **TOM2_8**: Output of TOM2_8 7_H **Reserved**: Reserved, do not use … F_H **Reserved**: Reserved, do not use
0	31:4	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.110 eGTM to QSPIn output selection register. n<8 , if interface exists. (n=0)

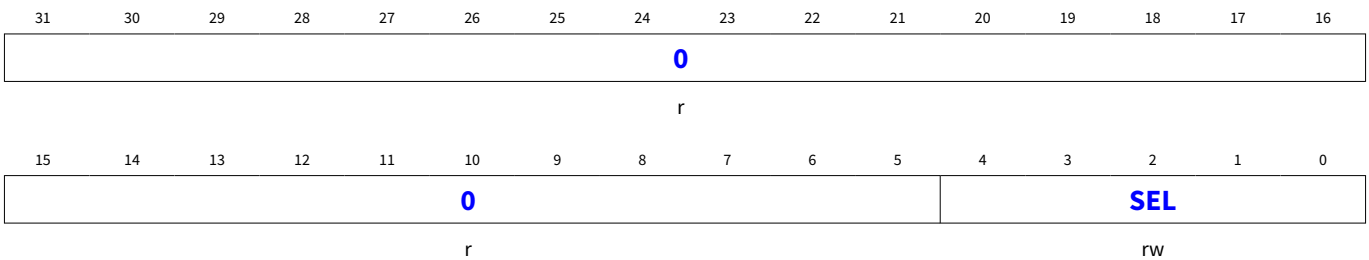
For a description of this register, see 7602.

EGTM_QSPIn_OUT (n=0)

Offset address: 061580_H+n*4

eGTM to QSPIn output selection register. n<8 , if interface exists.

Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
SEL	4:0	rw	**eGTM to QSPIn trigger output selection** 00_H **None**: Not connected (safe state) 01_H **CDTM0_DTM5_3**: Dead-time output of ATOM0_7 02_H **CDTM1_DTM5_3**: Dead-time output of ATOM1_7 03_H **CDTM2_DTM5_3**: Dead-time output of ATOM2_7 04_H **CDTM0_DTM3_3**: Dead-time output of TOM0_15 05_H **CDTM1_DTM3_3**: Dead-time output of TOM1_15 06_H **CDTM0_DTM4_0**: Dead-time output of ATOM0_0 07_H **CDTM1_DTM4_0**: Dead-time output of ATOM1_0 08_H **CDTM2_DTM4_0**: Dead-time output of ATOM2_0 09_H **CDTM0_DTM2_0**: Dead-time output of TOM0_8 $0A_H$ **CDTM1_DTM2_0**: Dead-time output of TOM1_8 $0B_H$ **CDTM2_DTM3_3**: Dead-time output of TOM2_15 $0C_H$ **Reserved**: Reserved, do not use … 17_H **Reserved**: Reserved, do not use 18_H **CDTM2_DTM2_0**: Dead-time output of TOM2_8 19_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	31:5	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.111 eGTM to QSPIn output selection register. n<8 , if interface exists. (n=1)

For a description of this register, see 7602.

EGTM_QSPIn_OUT (n=1)

eGTM to QSPIn output selection register. n<8 , if interface exists.

Offset address: 061580_H+n*4

Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0											SEL				
r											rw				

Field	Bits	Type	Description
SEL	4:0	rw	**eGTM to QSPIn trigger output selection** 00_H **None**: Not connected (safe state) 01_H **CDTM0_DTM5_2**: Dead-time output of ATOM0_6 02_H **CDTM1_DTM5_2**: Dead-time output of ATOM1_6 03_H **CDTM2_DTM5_2**: Dead-time output of ATOM2_6 04_H **CDTM0_DTM3_2**: Dead-time output of TOM0_14 05_H **CDTM1_DTM3_2**: Dead-time output of TOM1_14 06_H **CDTM0_DTM4_1**: Dead-time output of ATOM0_1 07_H **CDTM1_DTM4_1**: Dead-time output of ATOM1_1 08_H **CDTM2_DTM4_1**: Dead-time output of ATOM2_1 09_H **CDTM0_DTM2_1**: Dead-time output of TOM0_9 $0A_H$ **CDTM1_DTM2_1**: Dead-time output of TOM1_9 $0B_H$ **CDTM2_DTM3_2**: Dead-time output of TOM2_14 $0C_H$ **Reserved**: Reserved, do not use … 17_H **Reserved**: Reserved, do not use 18_H **CDTM2_DTM2_1**: Dead-time output of TOM2_9 19_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	31:5	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.112 eGTM to QSPIn output selection register. n<8 , if interface exists. (n=2)

For a description of this register, see 7602.

EGTM_QSPIn_OUT (n=2)

eGTM to QSPIn output selection register. n<8 , if interface exists.

Offset address: 061580_H+n*4

Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0											SEL				
r											rw				

Field	Bits	Type	Description
SEL	4:0	rw	**eGTM to QSPIn trigger output selection** 00_H **None**: Not connected (safe state) 01_H **CDTM0_DTM5_1**: Dead-time output of ATOM0_5 02_H **CDTM1_DTM5_1**: Dead-time output of ATOM1_5 03_H **CDTM2_DTM5_1**: Dead-time output of ATOM2_5 04_H **CDTM0_DTM3_1**: Dead-time output of TOM0_13 05_H **CDTM1_DTM3_1**: Dead-time output of TOM1_13 06_H **CDTM0_DTM4_2**: Dead-time output of ATOM0_2 07_H **CDTM1_DTM4_2**: Dead-time output of ATOM1_2 08_H **CDTM2_DTM4_2**: Dead-time output of ATOM2_2 09_H **CDTM0_DTM2_2**: Dead-time output of TOM0_10 $0A_H$ **CDTM1_DTM2_2**: Dead-time output of TOM1_10 $0B_H$ **CDTM2_DTM3_1**: Dead-time output of TOM2_13 $0C_H$ **Reserved**: Reserved, do not use … 17_H **Reserved**: Reserved, do not use 18_H **CDTM2_DTM2_2**: Dead-time output of TOM2_10 19_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	31:5	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.113 eGTM to QSPIn output selection register. n<8 , if interface exists. (n=3)

For a description of this register, see 7602.

EGTM_QSPIn_OUT (n=3)

eGTM to QSPIn output selection register. n<8 , if interface exists.

Offset address: 061580_H+n*4

Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0											SEL				
r											rw				

Field	Bits	Type	Description
SEL	4:0	rw	**eGTM to QSPIn trigger output selection** 00_H **None**: Not connected (safe state) 01_H **CDTM0_DTM5_0**: Dead-time output of ATOM0_4 02_H **CDTM1_DTM5_0**: Dead-time output of ATOM1_4 03_H **CDTM2_DTM5_0**: Dead-time output of ATOM2_4 04_H **CDTM0_DTM3_0**: Dead-time output of TOM0_12 05_H **CDTM1_DTM3_0**: Dead-time output of TOM1_12 06_H **CDTM0_DTM4_3**: Dead-time output of ATOM0_3 07_H **CDTM1_DTM4_3**: Dead-time output of ATOM1_3 08_H **CDTM2_DTM4_3**: Dead-time output of ATOM2_3 09_H **CDTM0_DTM2_3**: Dead-time output of TOM0_11 $0A_H$ **CDTM1_DTM2_3**: Dead-time output of TOM1_11 $0B_H$ **CDTM2_DTM3_0**: Dead-time output of TOM2_12 $0C_H$ **Reserved**: Reserved, do not use … 17_H **Reserved**: Reserved, do not use 18_H **CDTM2_DTM2_3**: Dead-time output of TOM2_11 19_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	31:5	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.114 eGTM to QSPIn output selection register. n<8 , if interface exists. (n=4)

For a description of this register, see 7602.

EGTM_QSPIn_OUT (n=4)

eGTM to QSPIn output selection register. n<8 , if interface exists.

Offset address: 061580_H+n*4

Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0											SEL				
r											rw				

Field	Bits	Type	Description
SEL	4:0	rw	**eGTM to QSPIn trigger output selection** 00_H **None**: Not connected (safe state) 01_H **CDTM0_DTM4_3**: Dead-time output of ATOM0_3 02_H **CDTM1_DTM4_3**: Dead-time output of ATOM1_3 03_H **CDTM2_DTM4_3**: Dead-time output of ATOM2_3 04_H **CDTM0_DTM2_3**: Dead-time output of TOM0_11 05_H **CDTM1_DTM2_3**: Dead-time output of TOM1_11 06_H **CDTM0_DTM5_0**: Dead-time output of ATOM0_4 07_H **CDTM1_DTM5_0**: Dead-time output of ATOM1_4 08_H **CDTM2_DTM5_0**: Dead-time output of ATOM2_4 09_H **CDTM0_DTM3_0**: Dead-time output of TOM0_12 $0A_H$ **CDTM1_DTM3_0**: Dead-time output of TOM1_12 $0B_H$ **CDTM2_DTM2_3**: Dead-time output of TOM2_11 $0C_H$ **Reserved**: Reserved, do not use … 17_H **Reserved**: Reserved, do not use 18_H **CDTM2_DTM3_0**: Dead-time output of TOM2_12 19_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	31:5	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.115 eGTM to QSPIn output selection register. n<8 , if interface exists. (n=5)

For a description of this register, see 7602.

EGTM_QSPIn_OUT (n=5)

eGTM to QSPIn output selection register. n<8 , if interface exists.

Offset address: 061580_H+n*4

Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0											SEL				
r											rw				

Field	Bits	Type	Description
SEL	4:0	rw	**eGTM to QSPIn trigger output selection** 00_H **None**: Not connected (safe state) 01_H **CDTM0_DTM4_2**: Dead-time output of ATOM0_2 02_H **CDTM1_DTM4_2**: Dead-time output of ATOM1_2 03_H **CDTM2_DTM4_2**: Dead-time output of ATOM2_2 04_H **CDTM0_DTM2_2**: Dead-time output of TOM0_10 05_H **CDTM1_DTM2_2**: Dead-time output of TOM1_10 06_H **CDTM0_DTM5_1**: Dead-time output of ATOM0_5 07_H **CDTM1_DTM5_1**: Dead-time output of ATOM1_5 08_H **CDTM2_DTM5_1**: Dead-time output of ATOM2_5 09_H **CDTM0_DTM3_1**: Dead-time output of TOM0_13 $0A_H$ **CDTM1_DTM3_1**: Dead-time output of TOM1_13 $0B_H$ **CDTM2_DTM2_2**: Dead-time output of TOM2_10 $0C_H$ **Reserved**: Reserved, do not use … 17_H **Reserved**: Reserved, do not use 18_H **CDTM2_DTM3_1**: Dead-time output of TOM2_13 19_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	31:5	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.116 eGTM to QSPIn output selection register. n<8 , if interface exists. (n=6)

For a description of this register, see 7602.

EGTM_QSPIn_OUT (n=6)

eGTM to QSPIn output selection register. n<8 , if interface exists.

Offset address: 061580_H+n*4

Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0											SEL				
r											rw				

Field	Bits	Type	Description
SEL	4:0	rw	**eGTM to QSPIn trigger output selection** 00_H **None**: Not connected (safe state) 01_H **CDTM0_DTM4_1**: Dead-time output of ATOM0_1 02_H **CDTM1_DTM4_1**: Dead-time output of ATOM1_1 03_H **CDTM2_DTM4_1**: Dead-time output of ATOM2_1 04_H **CDTM0_DTM2_1**: Dead-time output of TOM0_9 05_H **CDTM1_DTM2_1**: Dead-time output of TOM1_9 06_H **CDTM0_DTM5_2**: Dead-time output of ATOM0_6 07_H **CDTM1_DTM5_2**: Dead-time output of ATOM1_6 08_H **CDTM2_DTM5_2**: Dead-time output of ATOM2_6 09_H **CDTM0_DTM3_2**: Dead-time output of TOM0_14 $0A_H$ **CDTM1_DTM3_2**: Dead-time output of TOM1_14 $0B_H$ **CDTM2_DTM2_1**: Dead-time output of TOM2_9 $0C_H$ **Reserved**: Reserved, do not use … 17_H **Reserved**: Reserved, do not use 18_H **CDTM2_DTM3_2**: Dead-time output of TOM2_14 19_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	31:5	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.117 eGTM to QSPIn output selection register. n<8 , if interface exists. (n=7)

For a description of this register, see 7602.

EGTM_QSPIn_OUT (n=7)

eGTM to QSPIn output selection register. n<8 , if interface exists.

Offset address: 061580_H+n*4

Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0											SEL				
r											rw				

Field	Bits	Type	Description
SEL	4:0	rw	**eGTM to QSPIn trigger output selection** 00_H **None**: Not connected (safe state) 01_H **CDTM0_DTM4_0**: Dead-time output of ATOM0_0 02_H **CDTM1_DTM4_0**: Dead-time output of ATOM1_0 03_H **CDTM2_DTM4_0**: Dead-time output of ATOM2_0 04_H **CDTM0_DTM2_0**: Dead-time output of TOM0_8 05_H **CDTM1_DTM2_0**: Dead-time output of TOM1_8 06_H **CDTM0_DTM5_3**: Dead-time output of ATOM0_7 07_H **CDTM1_DTM5_3**: Dead-time output of ATOM1_7 08_H **CDTM2_DTM5_3**: Dead-time output of ATOM2_7 09_H **CDTM0_DTM3_3**: Dead-time output of TOM0_15 $0A_H$ **CDTM1_DTM3_3**: Dead-time output of TOM1_15 $0B_H$ **CDTM2_DTM2_0**: Dead-time output of TOM2_8 $0C_H$ **Reserved**: Reserved, do not use … 17_H **Reserved**: Reserved, do not use 18_H **CDTM2_DTM3_3**: Dead-time output of TOM2_15 19_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	31:5	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.118 eGTM to ASCLINn output selection register. n<28 , if interface exists. (n=0)

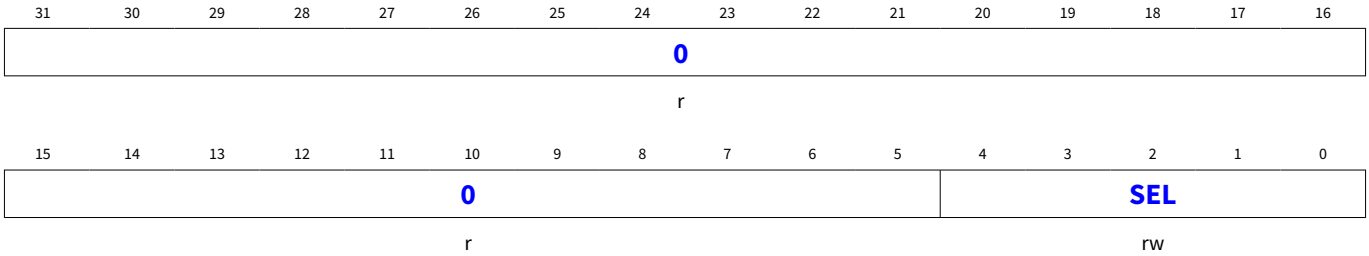
For a description of this register, see 7601.

EGTM_ASCLINn_OUT (n=0)

eGTM to ASCLINn output selection register. n<28 , if interface exists.

Offset address: 061600_H+n*4

Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
SEL	4:0	rw	**eGTM to ASCLINn trigger output selection** 00_H **None**: Not connected (safe state) 01_H **CDTM0_DTM5_3**: Dead-time output of ATOM0_7 02_H **CDTM1_DTM5_3**: Dead-time output of ATOM1_7 03_H **CDTM2_DTM5_3**: Dead-time output of ATOM2_7 04_H **CDTM0_DTM3_3**: Dead-time output of TOM0_15 05_H **CDTM1_DTM3_3**: Dead-time output of TOM1_15 06_H **CDTM0_DTM4_0**: Dead-time output of ATOM0_0 07_H **CDTM1_DTM4_0**: Dead-time output of ATOM1_0 08_H **CDTM2_DTM4_0**: Dead-time output of ATOM2_0 09_H **CDTM0_DTM2_0**: Dead-time output of TOM0_8 $0A_H$ **CDTM1_DTM2_0**: Dead-time output of TOM1_8 $0B_H$ **CDTM2_DTM3_3**: Dead-time output of TOM2_15 $0C_H$ **Reserved**: Reserved, do not use … 17_H **Reserved**: Reserved, do not use 18_H **CDTM2_DTM2_0**: Dead-time output of TOM2_8 19_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	31:5	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.119 eGTM to ASCLINn output selection register. n<28 , if interface exists. (n=1)

For a description of this register, see 7601.

EGTM_ASCLINn_OUT (n=1)

eGTM to ASCLINn output selection register. n<28 , if interface exists.

Offset address: 061600_H+n*4

Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0											SEL				
r											rw				

Field	Bits	Type	Description
SEL	4:0	rw	**eGTM to ASCLINn trigger output selection** 00_H **None**: Not connected (safe state) 01_H **CDTM0_DTM5_2**: Dead-time output of ATOM0_6 02_H **CDTM1_DTM5_2**: Dead-time output of ATOM1_6 03_H **CDTM2_DTM5_2**: Dead-time output of ATOM2_6 04_H **CDTM0_DTM3_2**: Dead-time output of TOM0_14 05_H **CDTM1_DTM3_2**: Dead-time output of TOM1_14 06_H **CDTM0_DTM4_1**: Dead-time output of ATOM0_1 07_H **CDTM1_DTM4_1**: Dead-time output of ATOM1_1 08_H **CDTM2_DTM4_1**: Dead-time output of ATOM2_1 09_H **CDTM0_DTM2_1**: Dead-time output of TOM0_9 $0A_H$ **CDTM1_DTM2_1**: Dead-time output of TOM1_9 $0B_H$ **CDTM2_DTM3_2**: Dead-time output of TOM2_14 $0C_H$ **Reserved**: Reserved, do not use … 17_H **Reserved**: Reserved, do not use 18_H **CDTM2_DTM2_1**: Dead-time output of TOM2_9 19_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	31:5	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.120 eGTM to ASCLINn output selection register. n<28 , if interface exists. (n=2)

For a description of this register, see 7601.

EGTM_ASCLINn_OUT (n=2)

eGTM to ASCLINn output selection register. n<28 , if interface exists.

Offset address: 061600_H+n*4

Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0											SEL				
r											rw				

Field	Bits	Type	Description
SEL	4:0	rw	**eGTM to ASCLINn trigger output selection** 00_H **None**: Not connected (safe state) 01_H **CDTM0_DTM5_1**: Dead-time output of ATOM0_5 02_H **CDTM1_DTM5_1**: Dead-time output of ATOM1_5 03_H **CDTM2_DTM5_1**: Dead-time output of ATOM2_5 04_H **CDTM0_DTM3_1**: Dead-time output of TOM0_13 05_H **CDTM1_DTM3_1**: Dead-time output of TOM1_13 06_H **CDTM0_DTM4_2**: Dead-time output of ATOM0_2 07_H **CDTM1_DTM4_2**: Dead-time output of ATOM1_2 08_H **CDTM2_DTM4_2**: Dead-time output of ATOM2_2 09_H **CDTM0_DTM2_2**: Dead-time output of TOM0_10 $0A_H$ **CDTM1_DTM2_2**: Dead-time output of TOM1_10 $0B_H$ **CDTM2_DTM3_1**: Dead-time output of TOM2_13 $0C_H$ **Reserved**: Reserved, do not use … 17_H **Reserved**: Reserved, do not use 18_H **CDTM2_DTM2_2**: Dead-time output of TOM2_10 19_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	31:5	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.121 eGTM to ASCLINn output selection register. n<28 , if interface exists. (n=3)

For a description of this register, see 7601.

EGTM_ASCLINn_OUT (n=3)

eGTM to ASCLINn output selection register. n<28 , if interface exists.

Offset address: 061600_H+n*4

Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0											SEL				
r											rw				

Field	Bits	Type	Description
SEL	4:0	rw	**eGTM to ASCLINn trigger output selection** 00_H **None**: Not connected (safe state) 01_H **CDTM0_DTM5_0**: Dead-time output of ATOM0_4 02_H **CDTM1_DTM5_0**: Dead-time output of ATOM1_4 03_H **CDTM2_DTM5_0**: Dead-time output of ATOM2_4 04_H **CDTM0_DTM3_0**: Dead-time output of TOM0_12 05_H **CDTM1_DTM3_0**: Dead-time output of TOM1_12 06_H **CDTM0_DTM4_3**: Dead-time output of ATOM0_3 07_H **CDTM1_DTM4_3**: Dead-time output of ATOM1_3 08_H **CDTM2_DTM4_3**: Dead-time output of ATOM2_3 09_H **CDTM0_DTM2_3**: Dead-time output of TOM0_11 $0A_H$ **CDTM1_DTM2_3**: Dead-time output of TOM1_11 $0B_H$ **CDTM2_DTM3_0**: Dead-time output of TOM2_12 $0C_H$ **Reserved**: Reserved, do not use … 17_H **Reserved**: Reserved, do not use 18_H **CDTM2_DTM2_3**: Dead-time output of TOM2_11 19_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	31:5	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.122 eGTM to ASCLINn output selection register. n<28 , if interface exists. (n=4)

For a description of this register, see 7601.

EGTM_ASCLINn_OUT (n=4)

eGTM to ASCLINn output selection register. n<28 , if interface exists.

Offset address: 061600_H+n*4

Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0											SEL				
r											rw				

Field	Bits	Type	Description
SEL	4:0	rw	**eGTM to ASCLINn trigger output selection** 00_H **None**: Not connected (safe state) 01_H **CDTM0_DTM4_3**: Dead-time output of ATOM0_3 02_H **CDTM1_DTM4_3**: Dead-time output of ATOM1_3 03_H **CDTM2_DTM4_3**: Dead-time output of ATOM2_3 04_H **CDTM0_DTM2_3**: Dead-time output of TOM0_11 05_H **CDTM1_DTM2_3**: Dead-time output of TOM1_11 06_H **CDTM0_DTM5_0**: Dead-time output of ATOM0_4 07_H **CDTM1_DTM5_0**: Dead-time output of ATOM1_4 08_H **CDTM2_DTM5_0**: Dead-time output of ATOM2_4 09_H **CDTM0_DTM3_0**: Dead-time output of TOM0_12 $0A_H$ **CDTM1_DTM3_0**: Dead-time output of TOM1_12 $0B_H$ **CDTM2_DTM2_3**: Dead-time output of TOM2_11 $0C_H$ **Reserved**: Reserved, do not use … 17_H **Reserved**: Reserved, do not use 18_H **CDTM2_DTM3_0**: Dead-time output of TOM2_12 19_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	31:5	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.123 eGTM to ASCLINn output selection register. n<28 , if interface exists. (n=5)

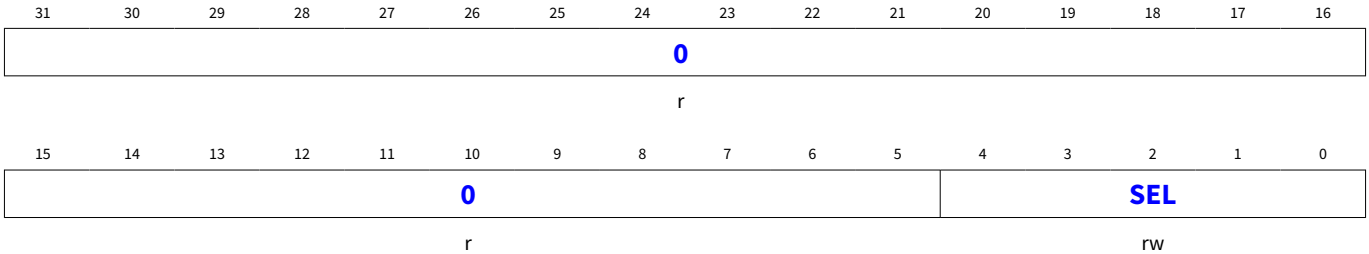
For a description of this register, see 7601.

EGTM_ASCLINn_OUT (n=5)

eGTM to ASCLINn output selection register. n<28 , if interface exists.

Offset address: 061600_H+n*4

Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
SEL	4:0	rw	**eGTM to ASCLINn trigger output selection** 00_H **None**: Not connected (safe state) 01_H **CDTM0_DTM4_2**: Dead-time output of ATOM0_2 02_H **CDTM1_DTM4_2**: Dead-time output of ATOM1_2 03_H **CDTM2_DTM4_2**: Dead-time output of ATOM2_2 04_H **CDTM0_DTM2_2**: Dead-time output of TOM0_10 05_H **CDTM1_DTM2_2**: Dead-time output of TOM1_10 06_H **CDTM0_DTM5_1**: Dead-time output of ATOM0_5 07_H **CDTM1_DTM5_1**: Dead-time output of ATOM1_5 08_H **CDTM2_DTM5_1**: Dead-time output of ATOM2_5 09_H **CDTM0_DTM3_1**: Dead-time output of TOM0_13 $0A_H$ **CDTM1_DTM3_1**: Dead-time output of TOM1_13 $0B_H$ **CDTM2_DTM2_2**: Dead-time output of TOM2_10 $0C_H$ **Reserved**: Reserved, do not use … 17_H **Reserved**: Reserved, do not use 18_H **CDTM2_DTM3_1**: Dead-time output of TOM2_13 19_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	31:5	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.124 eGTM to ASCLINn output selection register. n<28 , if interface exists. (n=6)

For a description of this register, see 7601.

EGTM_ASCLINn_OUT (n=6)

eGTM to ASCLINn output selection register. n<28 , if interface exists.

Offset address: 061600_H+n*4

Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0											SEL				
r											rw				

Field	Bits	Type	Description
SEL	4:0	rw	**eGTM to ASCLINn trigger output selection** 00_H **None**: Not connected (safe state) 01_H **CDTM0_DTM4_1**: Dead-time output of ATOM0_1 02_H **CDTM1_DTM4_1**: Dead-time output of ATOM1_1 03_H **CDTM2_DTM4_1**: Dead-time output of ATOM2_1 04_H **CDTM0_DTM2_1**: Dead-time output of TOM0_9 05_H **CDTM1_DTM2_1**: Dead-time output of TOM1_9 06_H **CDTM0_DTM5_2**: Dead-time output of ATOM0_6 07_H **CDTM1_DTM5_2**: Dead-time output of ATOM1_6 08_H **CDTM2_DTM5_2**: Dead-time output of ATOM2_6 09_H **CDTM0_DTM3_2**: Dead-time output of TOM0_14 $0A_H$ **CDTM1_DTM3_2**: Dead-time output of TOM1_14 $0B_H$ **CDTM2_DTM2_1**: Dead-time output of TOM2_9 $0C_H$ **Reserved**: Reserved, do not use … 17_H **Reserved**: Reserved, do not use 18_H **CDTM2_DTM3_2**: Dead-time output of TOM2_14 19_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	31:5	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.125 eGTM to ASCLINn output selection register. n<28 , if interface exists. (n=7)

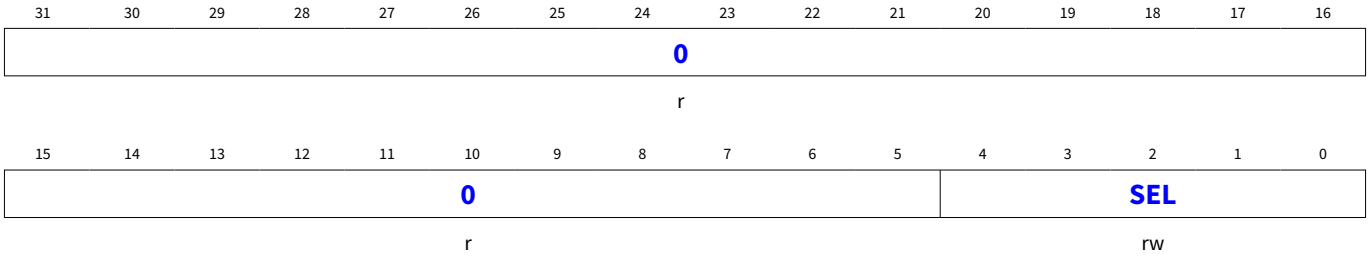
For a description of this register, see 7601.

EGTM_ASCLINn_OUT (n=7)

eGTM to ASCLINn output selection register. n<28 , if interface exists.

Offset address: 061600_H+n*4

Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
SEL	4:0	rw	**eGTM to ASCLINn trigger output selection** 00_H **None**: Not connected (safe state) 01_H **CDTM0_DTM4_0**: Dead-time output of ATOM0_0 02_H **CDTM1_DTM4_0**: Dead-time output of ATOM1_0 03_H **CDTM2_DTM4_0**: Dead-time output of ATOM2_0 04_H **CDTM0_DTM2_0**: Dead-time output of TOM0_8 05_H **CDTM1_DTM2_0**: Dead-time output of TOM1_8 06_H **CDTM0_DTM5_3**: Dead-time output of ATOM0_7 07_H **CDTM1_DTM5_3**: Dead-time output of ATOM1_7 08_H **CDTM2_DTM5_3**: Dead-time output of ATOM2_7 09_H **CDTM0_DTM3_3**: Dead-time output of TOM0_15 $0A_H$ **CDTM1_DTM3_3**: Dead-time output of TOM1_15 $0B_H$ **CDTM2_DTM2_0**: Dead-time output of TOM2_8 $0C_H$ **Reserved**: Reserved, do not use … 17_H **Reserved**: Reserved, do not use 18_H **CDTM2_DTM3_3**: Dead-time output of TOM2_15 19_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	31:5	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.126 eGTM to ASCLINn output selection register. n<28 , if interface exists. (n=8)

For a description of this register, see 7601.

EGTM_ASCLINn_OUT (n=8)

eGTM to ASCLINn output selection register. n<28 , if interface exists.

Offset address: 061600_H+n*4

Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0											SEL				
r											rw				

Field	Bits	Type	Description
SEL	4:0	rw	**eGTM to ASCLINn trigger output selection** 00_H **None**: Not connected (safe state) 01_H **CDTM0_DTM5_3**: Dead-time output of ATOM0_7 02_H **CDTM1_DTM5_3**: Dead-time output of ATOM1_7 03_H **CDTM2_DTM5_3**: Dead-time output of ATOM2_7 04_H **CDTM0_DTM3_3**: Dead-time output of TOM0_15 05_H **CDTM1_DTM3_3**: Dead-time output of TOM1_15 06_H **CDTM0_DTM4_0**: Dead-time output of ATOM0_0 07_H **CDTM1_DTM4_0**: Dead-time output of ATOM1_0 08_H **CDTM2_DTM4_0**: Dead-time output of ATOM2_0 09_H **CDTM0_DTM2_0**: Dead-time output of TOM0_8 $0A_H$ **CDTM1_DTM2_0**: Dead-time output of TOM1_8 $0B_H$ **CDTM2_DTM3_3**: Dead-time output of TOM2_15 $0C_H$ **Reserved**: Reserved, do not use … 17_H **Reserved**: Reserved, do not use 18_H **CDTM2_DTM2_0**: Dead-time output of TOM2_8 19_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	31:5	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.127 eGTM to ASCLINn output selection register. n<28 , if interface exists. (n=9)

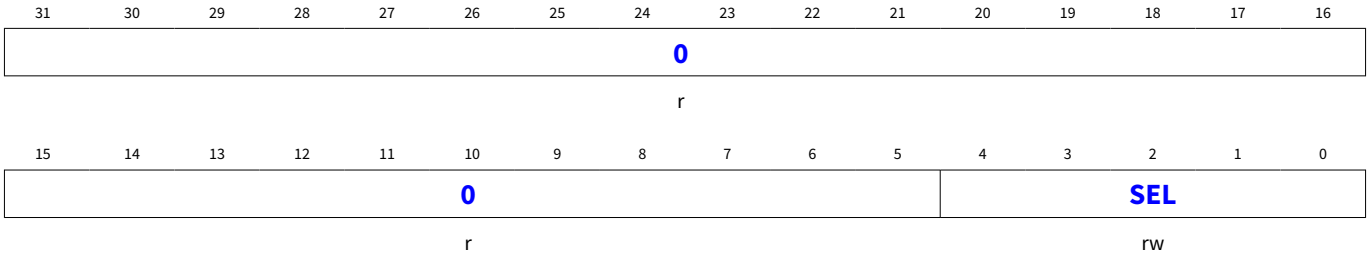
For a description of this register, see 7601.

EGTM_ASCLINn_OUT (n=9)

eGTM to ASCLINn output selection register. n<28 , if interface exists.

Offset address: 061600_H+n*4

Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
SEL	4:0	rw	**eGTM to ASCLINn trigger output selection** 00_H **None**: Not connected (safe state) 01_H **CDTM0_DTM5_2**: Dead-time output of ATOM0_6 02_H **CDTM1_DTM5_2**: Dead-time output of ATOM1_6 03_H **CDTM2_DTM5_2**: Dead-time output of ATOM2_6 04_H **CDTM0_DTM3_2**: Dead-time output of TOM0_14 05_H **CDTM1_DTM3_2**: Dead-time output of TOM1_14 06_H **CDTM0_DTM4_1**: Dead-time output of ATOM0_1 07_H **CDTM1_DTM4_1**: Dead-time output of ATOM1_1 08_H **CDTM2_DTM4_1**: Dead-time output of ATOM2_1 09_H **CDTM0_DTM2_1**: Dead-time output of TOM0_9 $0A_H$ **CDTM1_DTM2_1**: Dead-time output of TOM1_9 $0B_H$ **CDTM2_DTM3_2**: Dead-time output of TOM2_14 $0C_H$ **Reserved**: Reserved, do not use … 17_H **Reserved**: Reserved, do not use 18_H **CDTM2_DTM2_1**: Dead-time output of TOM2_9 19_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	31:5	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.128 eGTM to ASCLINn output selection register. n<28 , if interface exists. (n=10)

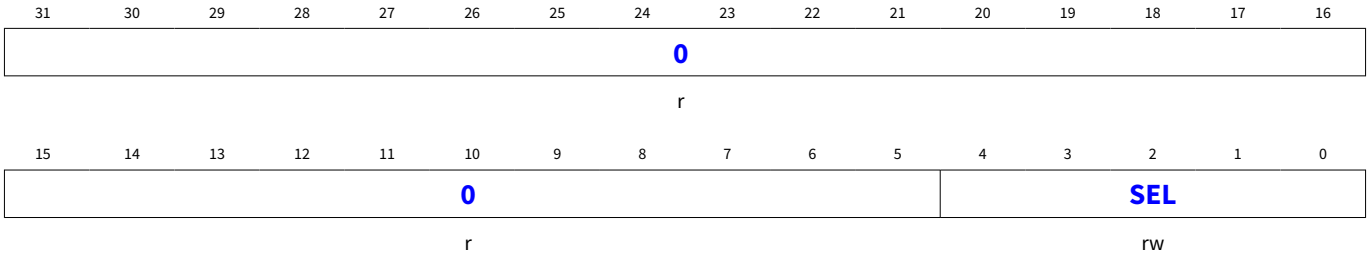
For a description of this register, see 7601.

EGTM_ASCLINn_OUT (n=10)

eGTM to ASCLINn output selection register. n<28 , if interface exists.

Offset address: 061600_H+n*4

Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
SEL	4:0	rw	**eGTM to ASCLINn trigger output selection** 00_H **None**: Not connected (safe state) 01_H **CDTM0_DTM5_1**: Dead-time output of ATOM0_5 02_H **CDTM1_DTM5_1**: Dead-time output of ATOM1_5 03_H **CDTM2_DTM5_1**: Dead-time output of ATOM2_5 04_H **CDTM0_DTM3_1**: Dead-time output of TOM0_13 05_H **CDTM1_DTM3_1**: Dead-time output of TOM1_13 06_H **CDTM0_DTM4_2**: Dead-time output of ATOM0_2 07_H **CDTM1_DTM4_2**: Dead-time output of ATOM1_2 08_H **CDTM2_DTM4_2**: Dead-time output of ATOM2_2 09_H **CDTM0_DTM2_2**: Dead-time output of TOM0_10 $0A_H$ **CDTM1_DTM2_2**: Dead-time output of TOM1_10 $0B_H$ **CDTM2_DTM3_1**: Dead-time output of TOM2_13 $0C_H$ **Reserved**: Reserved, do not use … 17_H **Reserved**: Reserved, do not use 18_H **CDTM2_DTM2_2**: Dead-time output of TOM2_10 19_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	31:5	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.129 eGTM to ASCLINn output selection register. n<28 , if interface exists. (n=11)

For a description of this register, see 7601.

EGTM_ASCLINn_OUT (n=11)

eGTM to ASCLINn output selection register. n<28 , if interface exists.

Offset address: 061600_H+n*4

Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0											SEL				
r											rw				

Field	Bits	Type	Description
SEL	4:0	rw	**eGTM to ASCLINn trigger output selection** 00_H **None**: Not connected (safe state) 01_H **CDTM0_DTM5_0**: Dead-time output of ATOM0_4 02_H **CDTM1_DTM5_0**: Dead-time output of ATOM1_4 03_H **CDTM2_DTM5_0**: Dead-time output of ATOM2_4 04_H **CDTM0_DTM3_0**: Dead-time output of TOM0_12 05_H **CDTM1_DTM3_0**: Dead-time output of TOM1_12 06_H **CDTM0_DTM4_3**: Dead-time output of ATOM0_3 07_H **CDTM1_DTM4_3**: Dead-time output of ATOM1_3 08_H **CDTM2_DTM4_3**: Dead-time output of ATOM2_3 09_H **CDTM0_DTM2_3**: Dead-time output of TOM0_11 $0A_H$ **CDTM1_DTM2_3**: Dead-time output of TOM1_11 $0B_H$ **CDTM2_DTM3_0**: Dead-time output of TOM2_12 $0C_H$ **Reserved**: Reserved, do not use … 17_H **Reserved**: Reserved, do not use 18_H **CDTM2_DTM2_3**: Dead-time output of TOM2_11 19_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	31:5	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.130 eGTM to ASCLINn output selection register. n<28 , if interface exists. (n=12)

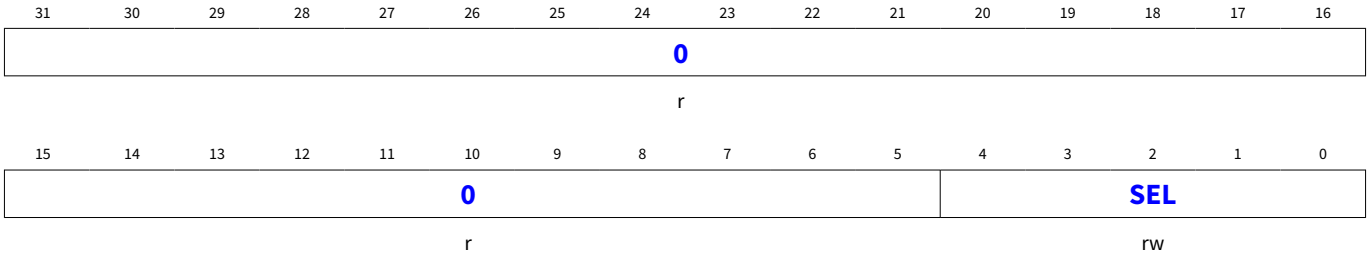
For a description of this register, see 7601.

EGTM_ASCLINn_OUT (n=12)

eGTM to ASCLINn output selection register. n<28 , if interface exists.

Offset address: 061600_H+n*4

Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
SEL	4:0	rw	**eGTM to ASCLINn trigger output selection** 00_H **None**: Not connected (safe state) 01_H **CDTM0_DTM4_3**: Dead-time output of ATOM0_3 02_H **CDTM1_DTM4_3**: Dead-time output of ATOM1_3 03_H **CDTM2_DTM4_3**: Dead-time output of ATOM2_3 04_H **CDTM0_DTM2_3**: Dead-time output of TOM0_11 05_H **CDTM1_DTM2_3**: Dead-time output of TOM1_11 06_H **CDTM0_DTM5_0**: Dead-time output of ATOM0_4 07_H **CDTM1_DTM5_0**: Dead-time output of ATOM1_4 08_H **CDTM2_DTM5_0**: Dead-time output of ATOM2_4 09_H **CDTM0_DTM3_0**: Dead-time output of TOM0_12 $0A_H$ **CDTM1_DTM3_0**: Dead-time output of TOM1_12 $0B_H$ **CDTM2_DTM2_3**: Dead-time output of TOM2_11 $0C_H$ **Reserved**: Reserved, do not use … 17_H **Reserved**: Reserved, do not use 18_H **CDTM2_DTM3_0**: Dead-time output of TOM2_12 19_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	31:5	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.131 eGTM to ASCLINn output selection register. n<28 , if interface exists. (n=13)

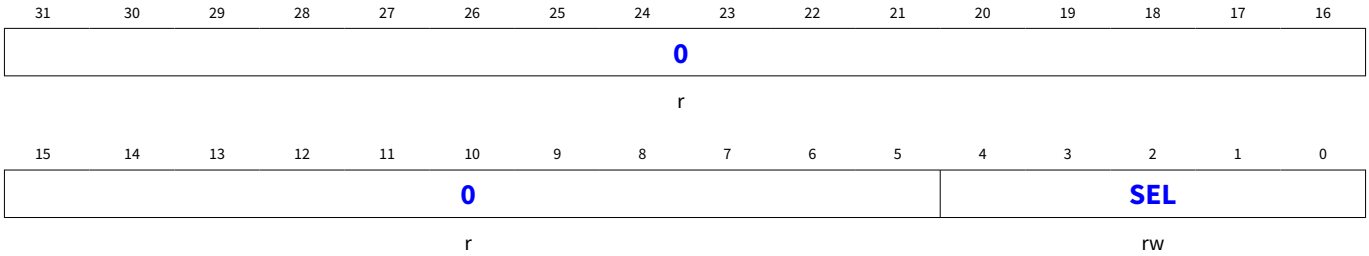
For a description of this register, see 7601.

EGTM_ASCLINn_OUT (n=13)

eGTM to ASCLINn output selection register. n<28 , if interface exists.

Offset address: 061600_H+n*4

Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
SEL	4:0	rw	**eGTM to ASCLINn trigger output selection** 00_H **None**: Not connected (safe state) 01_H **CDTM0_DTM4_2**: Dead-time output of ATOM0_2 02_H **CDTM1_DTM4_2**: Dead-time output of ATOM1_2 03_H **CDTM2_DTM4_2**: Dead-time output of ATOM2_2 04_H **CDTM0_DTM2_2**: Dead-time output of TOM0_10 05_H **CDTM1_DTM2_2**: Dead-time output of TOM1_10 06_H **CDTM0_DTM5_1**: Dead-time output of ATOM0_5 07_H **CDTM1_DTM5_1**: Dead-time output of ATOM1_5 08_H **CDTM2_DTM5_1**: Dead-time output of ATOM2_5 09_H **CDTM0_DTM3_1**: Dead-time output of TOM0_13 $0A_H$ **CDTM1_DTM3_1**: Dead-time output of TOM1_13 $0B_H$ **CDTM2_DTM2_2**: Dead-time output of TOM2_10 $0C_H$ **Reserved**: Reserved, do not use … 17_H **Reserved**: Reserved, do not use 18_H **CDTM2_DTM3_1**: Dead-time output of TOM2_13 19_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	31:5	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.132 eGTM to ASCLINn output selection register. n<28 , if interface exists. (n=14)

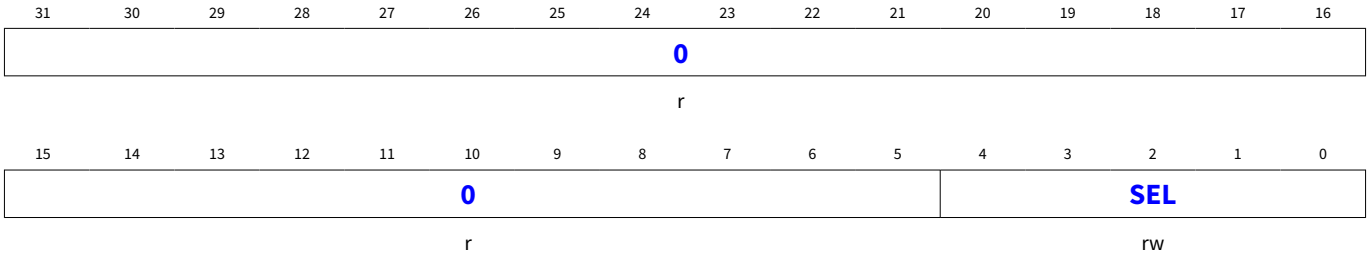
For a description of this register, see 7601.

EGTM_ASCLINn_OUT (n=14)

eGTM to ASCLINn output selection register. n<28 , if interface exists.

Offset address: 061600_H+n*4

Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
SEL	4:0	rw	**eGTM to ASCLINn trigger output selection** 00_H **None**: Not connected (safe state) 01_H **CDTM0_DTM4_1**: Dead-time output of ATOM0_1 02_H **CDTM1_DTM4_1**: Dead-time output of ATOM1_1 03_H **CDTM2_DTM4_1**: Dead-time output of ATOM2_1 04_H **CDTM0_DTM2_1**: Dead-time output of TOM0_9 05_H **CDTM1_DTM2_1**: Dead-time output of TOM1_9 06_H **CDTM0_DTM5_2**: Dead-time output of ATOM0_6 07_H **CDTM1_DTM5_2**: Dead-time output of ATOM1_6 08_H **CDTM2_DTM5_2**: Dead-time output of ATOM2_6 09_H **CDTM0_DTM3_2**: Dead-time output of TOM0_14 $0A_H$ **CDTM1_DTM3_2**: Dead-time output of TOM1_14 $0B_H$ **CDTM2_DTM2_1**: Dead-time output of TOM2_9 $0C_H$ **Reserved**: Reserved, do not use … 17_H **Reserved**: Reserved, do not use 18_H **CDTM2_DTM3_2**: Dead-time output of TOM2_14 19_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	31:5	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.133 eGTM to ASCLINn output selection register. n<28 , if interface exists. (n=15)

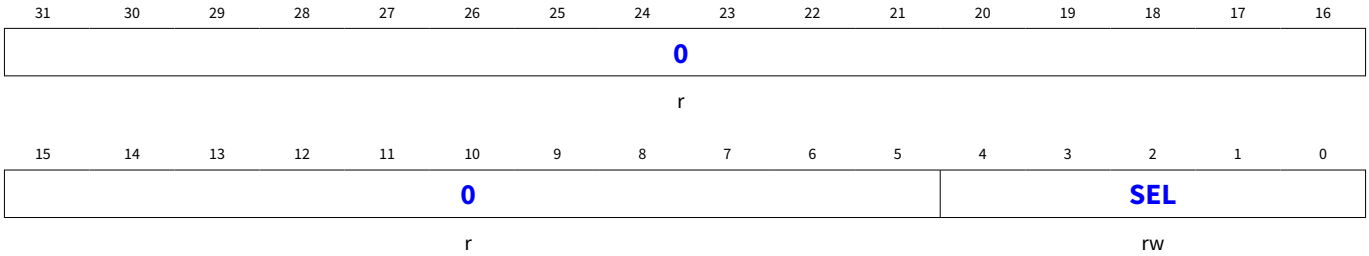
For a description of this register, see 7601.

EGTM_ASCLINn_OUT (n=15)

eGTM to ASCLINn output selection register. n<28 , if interface exists.

Offset address: 061600_H+n*4

Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
SEL	4:0	rw	**eGTM to ASCLINn trigger output selection** 00_H **None**: Not connected (safe state) 01_H **CDTM0_DTM4_0**: Dead-time output of ATOM0_0 02_H **CDTM1_DTM4_0**: Dead-time output of ATOM1_0 03_H **CDTM2_DTM4_0**: Dead-time output of ATOM2_0 04_H **CDTM0_DTM2_0**: Dead-time output of TOM0_8 05_H **CDTM1_DTM2_0**: Dead-time output of TOM1_8 06_H **CDTM0_DTM5_3**: Dead-time output of ATOM0_7 07_H **CDTM1_DTM5_3**: Dead-time output of ATOM1_7 08_H **CDTM2_DTM5_3**: Dead-time output of ATOM2_7 09_H **CDTM0_DTM3_3**: Dead-time output of TOM0_15 $0A_H$ **CDTM1_DTM3_3**: Dead-time output of TOM1_15 $0B_H$ **CDTM2_DTM2_0**: Dead-time output of TOM2_8 $0C_H$ **Reserved**: Reserved, do not use … 17_H **Reserved**: Reserved, do not use 18_H **CDTM2_DTM3_3**: Dead-time output of TOM2_15 19_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	31:5	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.134 eGTM to ASCLINn output selection register. n<28 , if interface exists. (n=16)

For a description of this register, see 7601.

EGTM_ASCLINn_OUT (n=16)

eGTM to ASCLINn output selection register. n<28 , if interface exists.

Offset address: 061600_H+n*4

Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0											SEL				
r											rw				

Field	Bits	Type	Description
SEL	4:0	rw	**eGTM to ASCLINn trigger output selection** 00_H **None**: Not connected (safe state) 01_H **CDTM0_DTM5_3**: Dead-time output of ATOM0_7 02_H **CDTM1_DTM5_3**: Dead-time output of ATOM1_7 03_H **CDTM2_DTM5_3**: Dead-time output of ATOM2_7 04_H **CDTM0_DTM3_3**: Dead-time output of TOM0_15 05_H **CDTM1_DTM3_3**: Dead-time output of TOM1_15 06_H **CDTM0_DTM4_0**: Dead-time output of ATOM0_0 07_H **CDTM1_DTM4_0**: Dead-time output of ATOM1_0 08_H **CDTM2_DTM4_0**: Dead-time output of ATOM2_0 09_H **CDTM0_DTM2_0**: Dead-time output of TOM0_8 $0A_H$ **CDTM1_DTM2_0**: Dead-time output of TOM1_8 $0B_H$ **CDTM2_DTM3_3**: Dead-time output of TOM2_15 $0C_H$ **Reserved**: Reserved, do not use … 17_H **Reserved**: Reserved, do not use 18_H **CDTM2_DTM2_0**: Dead-time output of TOM2_8 19_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	31:5	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.135 eGTM to ASCLINn output selection register. n<28 , if interface exists. (n=17)

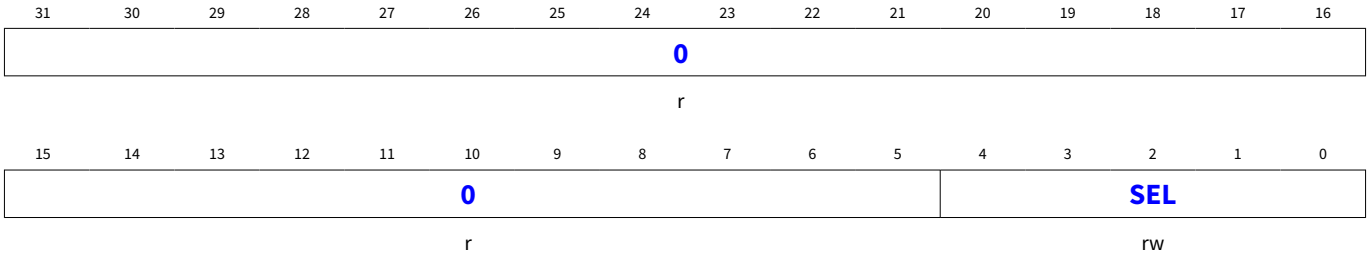
For a description of this register, see 7601.

EGTM_ASCLINn_OUT (n=17)

eGTM to ASCLINn output selection register. n<28 , if interface exists.

Offset address: 061600_H+n*4

Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
SEL	4:0	rw	**eGTM to ASCLINn trigger output selection** 00_H **None**: Not connected (safe state) 01_H **CDTM0_DTM5_2**: Dead-time output of ATOM0_6 02_H **CDTM1_DTM5_2**: Dead-time output of ATOM1_6 03_H **CDTM2_DTM5_2**: Dead-time output of ATOM2_6 04_H **CDTM0_DTM3_2**: Dead-time output of TOM0_14 05_H **CDTM1_DTM3_2**: Dead-time output of TOM1_14 06_H **CDTM0_DTM4_1**: Dead-time output of ATOM0_1 07_H **CDTM1_DTM4_1**: Dead-time output of ATOM1_1 08_H **CDTM2_DTM4_1**: Dead-time output of ATOM2_1 09_H **CDTM0_DTM2_1**: Dead-time output of TOM0_9 $0A_H$ **CDTM1_DTM2_1**: Dead-time output of TOM1_9 $0B_H$ **CDTM2_DTM3_2**: Dead-time output of TOM2_14 $0C_H$ **Reserved**: Reserved, do not use … 17_H **Reserved**: Reserved, do not use 18_H **CDTM2_DTM2_1**: Dead-time output of TOM2_9 19_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	31:5	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.136 eGTM to ASCLINn output selection register. n<28 , if interface exists. (n=18)

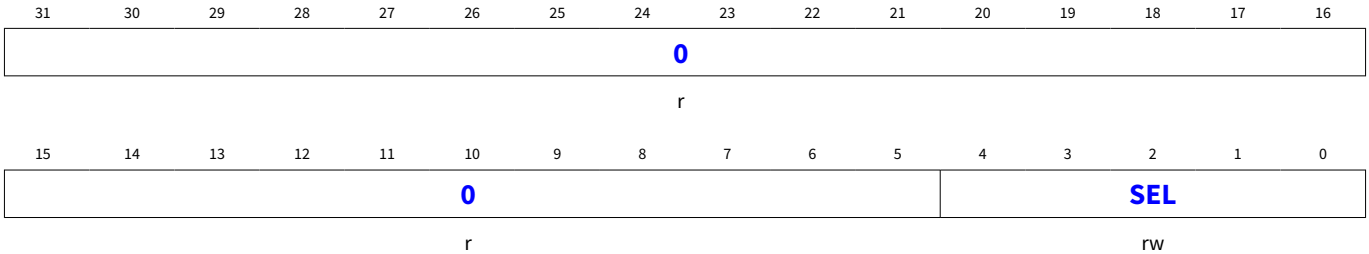
For a description of this register, see 7601.

EGTM_ASCLINn_OUT (n=18)

eGTM to ASCLINn output selection register. n<28 , if interface exists.

Offset address: 061600_H+n*4

Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
SEL	4:0	rw	**eGTM to ASCLINn trigger output selection** 00_H **None**: Not connected (safe state) 01_H **CDTM0_DTM5_1**: Dead-time output of ATOM0_5 02_H **CDTM1_DTM5_1**: Dead-time output of ATOM1_5 03_H **CDTM2_DTM5_1**: Dead-time output of ATOM2_5 04_H **CDTM0_DTM3_1**: Dead-time output of TOM0_13 05_H **CDTM1_DTM3_1**: Dead-time output of TOM1_13 06_H **CDTM0_DTM4_2**: Dead-time output of ATOM0_2 07_H **CDTM1_DTM4_2**: Dead-time output of ATOM1_2 08_H **CDTM2_DTM4_2**: Dead-time output of ATOM2_2 09_H **CDTM0_DTM2_2**: Dead-time output of TOM0_10 $0A_H$ **CDTM1_DTM2_2**: Dead-time output of TOM1_10 $0B_H$ **CDTM2_DTM3_1**: Dead-time output of TOM2_13 $0C_H$ **Reserved**: Reserved, do not use … 17_H **Reserved**: Reserved, do not use 18_H **CDTM2_DTM2_2**: Dead-time output of TOM2_10 19_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	31:5	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.137 eGTM to ASCLINn output selection register. n<28 , if interface exists. (n=19)

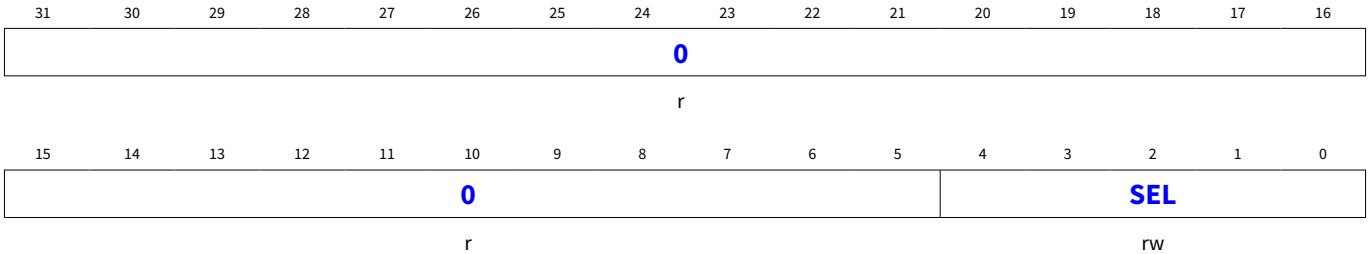
For a description of this register, see 7601.

EGTM_ASCLINn_OUT (n=19)

eGTM to ASCLINn output selection register. n<28 , if interface exists.

Offset address: 061600_H+n*4

Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
SEL	4:0	rw	**eGTM to ASCLINn trigger output selection** 00_H **None**: Not connected (safe state) 01_H **CDTM0_DTM5_0**: Dead-time output of ATOM0_4 02_H **CDTM1_DTM5_0**: Dead-time output of ATOM1_4 03_H **CDTM2_DTM5_0**: Dead-time output of ATOM2_4 04_H **CDTM0_DTM3_0**: Dead-time output of TOM0_12 05_H **CDTM1_DTM3_0**: Dead-time output of TOM1_12 06_H **CDTM0_DTM4_3**: Dead-time output of ATOM0_3 07_H **CDTM1_DTM4_3**: Dead-time output of ATOM1_3 08_H **CDTM2_DTM4_3**: Dead-time output of ATOM2_3 09_H **CDTM0_DTM2_3**: Dead-time output of TOM0_11 $0A_H$ **CDTM1_DTM2_3**: Dead-time output of TOM1_11 $0B_H$ **CDTM2_DTM3_0**: Dead-time output of TOM2_12 $0C_H$ **Reserved**: Reserved, do not use … 17_H **Reserved**: Reserved, do not use 18_H **CDTM2_DTM2_3**: Dead-time output of TOM2_11 19_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	31:5	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.138 eGTM to ASCLINn output selection register. n<28 , if interface exists. (n=20)

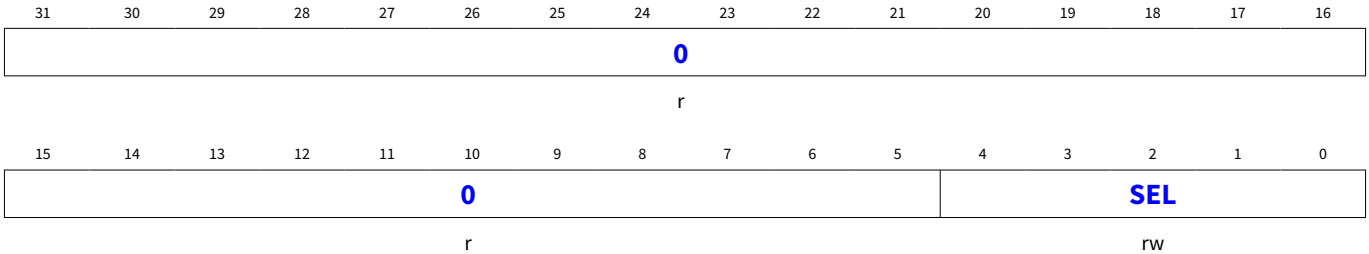
For a description of this register, see 7601.

EGTM_ASCLINn_OUT (n=20)

eGTM to ASCLINn output selection register. n<28 , if interface exists.

Offset address: 061600_H+n*4

Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
SEL	4:0	rw	**eGTM to ASCLINn trigger output selection** 00_H **None**: Not connected (safe state) 01_H **CDTM0_DTM4_3**: Dead-time output of ATOM0_3 02_H **CDTM1_DTM4_3**: Dead-time output of ATOM1_3 03_H **CDTM2_DTM4_3**: Dead-time output of ATOM2_3 04_H **CDTM0_DTM2_3**: Dead-time output of TOM0_11 05_H **CDTM1_DTM2_3**: Dead-time output of TOM1_11 06_H **CDTM0_DTM5_0**: Dead-time output of ATOM0_4 07_H **CDTM1_DTM5_0**: Dead-time output of ATOM1_4 08_H **CDTM2_DTM5_0**: Dead-time output of ATOM2_4 09_H **CDTM0_DTM3_0**: Dead-time output of TOM0_12 $0A_H$ **CDTM1_DTM3_0**: Dead-time output of TOM1_12 $0B_H$ **CDTM2_DTM2_3**: Dead-time output of TOM2_11 $0C_H$ **Reserved**: Reserved, do not use … 17_H **Reserved**: Reserved, do not use 18_H **CDTM2_DTM3_0**: Dead-time output of TOM2_12 19_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	31:5	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.139 eGTM to ASCLINn output selection register. n<28 , if interface exists. (n=21)

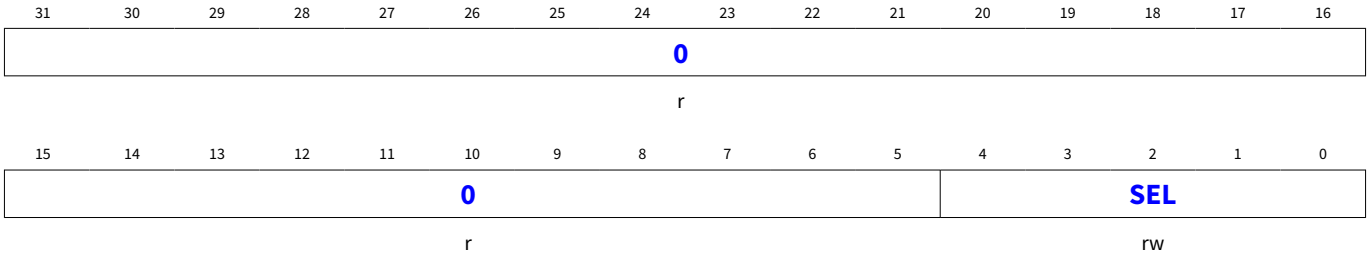
For a description of this register, see 7601.

EGTM_ASCLINn_OUT (n=21)

eGTM to ASCLINn output selection register. n<28 , if interface exists.

Offset address: 061600_H+n*4

Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
SEL	4:0	rw	**eGTM to ASCLINn trigger output selection** 00_H **None**: Not connected (safe state) 01_H **CDTM0_DTM4_2**: Dead-time output of ATOM0_2 02_H **CDTM1_DTM4_2**: Dead-time output of ATOM1_2 03_H **CDTM2_DTM4_2**: Dead-time output of ATOM2_2 04_H **CDTM0_DTM2_2**: Dead-time output of TOM0_10 05_H **CDTM1_DTM2_2**: Dead-time output of TOM1_10 06_H **CDTM0_DTM5_1**: Dead-time output of ATOM0_5 07_H **CDTM1_DTM5_1**: Dead-time output of ATOM1_5 08_H **CDTM2_DTM5_1**: Dead-time output of ATOM2_5 09_H **CDTM0_DTM3_1**: Dead-time output of TOM0_13 $0A_H$ **CDTM1_DTM3_1**: Dead-time output of TOM1_13 $0B_H$ **CDTM2_DTM2_2**: Dead-time output of TOM2_10 $0C_H$ **Reserved**: Reserved, do not use … 17_H **Reserved**: Reserved, do not use 18_H **CDTM2_DTM3_1**: Dead-time output of TOM2_13 19_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	31:5	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.140 eGTM to ASCLINn output selection register. n<28 , if interface exists. (n=22)

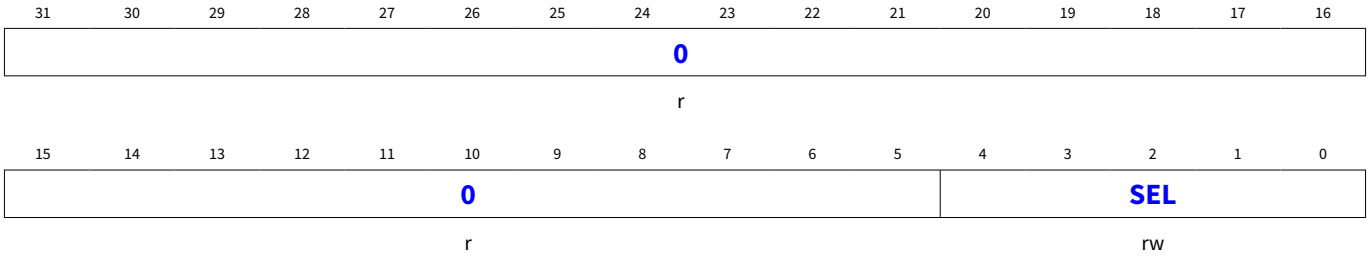
For a description of this register, see 7601.

EGTM_ASCLINn_OUT (n=22)

eGTM to ASCLINn output selection register. n<28 , if interface exists.

Offset address: 061600_H+n*4

Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
SEL	4:0	rw	**eGTM to ASCLINn trigger output selection** 00_H **None**: Not connected (safe state) 01_H **CDTM0_DTM4_1**: Dead-time output of ATOM0_1 02_H **CDTM1_DTM4_1**: Dead-time output of ATOM1_1 03_H **CDTM2_DTM4_1**: Dead-time output of ATOM2_1 04_H **CDTM0_DTM2_1**: Dead-time output of TOM0_9 05_H **CDTM1_DTM2_1**: Dead-time output of TOM1_9 06_H **CDTM0_DTM5_2**: Dead-time output of ATOM0_6 07_H **CDTM1_DTM5_2**: Dead-time output of ATOM1_6 08_H **CDTM2_DTM5_2**: Dead-time output of ATOM2_6 09_H **CDTM0_DTM3_2**: Dead-time output of TOM0_14 $0A_H$ **CDTM1_DTM3_2**: Dead-time output of TOM1_14 $0B_H$ **CDTM2_DTM2_1**: Dead-time output of TOM2_9 $0C_H$ **Reserved**: Reserved, do not use … 17_H **Reserved**: Reserved, do not use 18_H **CDTM2_DTM3_2**: Dead-time output of TOM2_14 19_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	31:5	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.141 eGTM to ASCLINn output selection register. n<28 , if interface exists. (n=23)

For a description of this register, see 7601.

EGTM_ASCLINn_OUT (n=23)

eGTM to ASCLINn output selection register. n<28 , if interface exists.

Offset address: 061600_H+n*4

Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0											SEL				
r											rw				

Field	Bits	Type	Description
SEL	4:0	rw	**eGTM to ASCLINn trigger output selection** 00_H **None**: Not connected (safe state) 01_H **CDTM0_DTM4_0**: Dead-time output of ATOM0_0 02_H **CDTM1_DTM4_0**: Dead-time output of ATOM1_0 03_H **CDTM2_DTM4_0**: Dead-time output of ATOM2_0 04_H **CDTM0_DTM2_0**: Dead-time output of TOM0_8 05_H **CDTM1_DTM2_0**: Dead-time output of TOM1_8 06_H **CDTM0_DTM5_3**: Dead-time output of ATOM0_7 07_H **CDTM1_DTM5_3**: Dead-time output of ATOM1_7 08_H **CDTM2_DTM5_3**: Dead-time output of ATOM2_7 09_H **CDTM0_DTM3_3**: Dead-time output of TOM0_15 $0A_H$ **CDTM1_DTM3_3**: Dead-time output of TOM1_15 $0B_H$ **CDTM2_DTM2_0**: Dead-time output of TOM2_8 $0C_H$ **Reserved**: Reserved, do not use … 17_H **Reserved**: Reserved, do not use 18_H **CDTM2_DTM3_3**: Dead-time output of TOM2_15 19_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	31:5	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.142 eGTM to ASCLINn output selection register. n<28 , if interface exists. (n=24)

For a description of this register, see 7601.

EGTM_ASCLINn_OUT (n=24)

eGTM to ASCLINn output selection register. n<28 , if interface exists.

Offset address: 061600_H+n*4

Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0											SEL				
r											rw				

Field	Bits	Type	Description
SEL	4:0	rw	**eGTM to ASCLINn trigger output selection** 00_H **None**: Not connected (safe state) 01_H **CDTM0_DTM5_3**: Dead-time output of ATOM0_7 02_H **CDTM1_DTM5_3**: Dead-time output of ATOM1_7 03_H **CDTM2_DTM5_3**: Dead-time output of ATOM2_7 04_H **CDTM0_DTM3_3**: Dead-time output of TOM0_15 05_H **CDTM1_DTM3_3**: Dead-time output of TOM1_15 06_H **CDTM0_DTM4_0**: Dead-time output of ATOM0_0 07_H **CDTM1_DTM4_0**: Dead-time output of ATOM1_0 08_H **CDTM2_DTM4_0**: Dead-time output of ATOM2_0 09_H **CDTM0_DTM2_0**: Dead-time output of TOM0_8 $0A_H$ **CDTM1_DTM2_0**: Dead-time output of TOM1_8 $0B_H$ **CDTM2_DTM3_3**: Dead-time output of TOM2_15 $0C_H$ **Reserved**: Reserved, do not use … 17_H **Reserved**: Reserved, do not use 18_H **CDTM2_DTM2_0**: Dead-time output of TOM2_8 19_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	31:5	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.143 eGTM to ASCLINn output selection register. n<28 , if interface exists. (n=25)

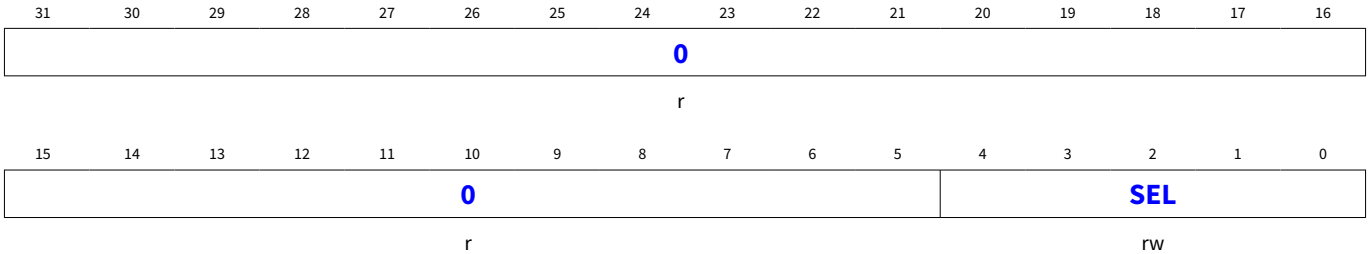
For a description of this register, see 7601.

EGTM_ASCLINn_OUT (n=25)

eGTM to ASCLINn output selection register. n<28 , if interface exists.

Offset address: 061600_H+n*4

Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
SEL	4:0	rw	**eGTM to ASCLINn trigger output selection** 00_H **None**: Not connected (safe state) 01_H **CDTM0_DTM5_2**: Dead-time output of ATOM0_6 02_H **CDTM1_DTM5_2**: Dead-time output of ATOM1_6 03_H **CDTM2_DTM5_2**: Dead-time output of ATOM2_6 04_H **CDTM0_DTM3_2**: Dead-time output of TOM0_14 05_H **CDTM1_DTM3_2**: Dead-time output of TOM1_14 06_H **CDTM0_DTM4_1**: Dead-time output of ATOM0_1 07_H **CDTM1_DTM4_1**: Dead-time output of ATOM1_1 08_H **CDTM2_DTM4_1**: Dead-time output of ATOM2_1 09_H **CDTM0_DTM2_1**: Dead-time output of TOM0_9 $0A_H$ **CDTM1_DTM2_1**: Dead-time output of TOM1_9 $0B_H$ **CDTM2_DTM3_2**: Dead-time output of TOM2_14 $0C_H$ **Reserved**: Reserved, do not use … 17_H **Reserved**: Reserved, do not use 18_H **CDTM2_DTM2_1**: Dead-time output of TOM2_9 19_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	31:5	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.144 eGTM to ASCLINn output selection register. n<28 , if interface exists. (n=26)

For a description of this register, see 7601.

EGTM_ASCLINn_OUT (n=26)

eGTM to ASCLINn output selection register. n<28 , if interface exists.

Offset address: 061600_H+n*4

Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0											SEL				
r											rw				

Field	Bits	Type	Description
SEL	4:0	rw	**eGTM to ASCLINn trigger output selection** 00_H **None**: Not connected (safe state) 01_H **CDTM0_DTM5_1**: Dead-time output of ATOM0_5 02_H **CDTM1_DTM5_1**: Dead-time output of ATOM1_5 03_H **CDTM2_DTM5_1**: Dead-time output of ATOM2_5 04_H **CDTM0_DTM3_1**: Dead-time output of TOM0_13 05_H **CDTM1_DTM3_1**: Dead-time output of TOM1_13 06_H **CDTM0_DTM4_2**: Dead-time output of ATOM0_2 07_H **CDTM1_DTM4_2**: Dead-time output of ATOM1_2 08_H **CDTM2_DTM4_2**: Dead-time output of ATOM2_2 09_H **CDTM0_DTM2_2**: Dead-time output of TOM0_10 $0A_H$ **CDTM1_DTM2_2**: Dead-time output of TOM1_10 $0B_H$ **CDTM2_DTM3_1**: Dead-time output of TOM2_13 $0C_H$ **Reserved**: Reserved, do not use … 17_H **Reserved**: Reserved, do not use 18_H **CDTM2_DTM2_2**: Dead-time output of TOM2_10 19_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	31:5	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.145 eGTM to ASCLINn output selection register. n<28 , if interface exists. (n=27)

For a description of this register, see 7601.

EGTM_ASCLINn_OUT (n=27)

eGTM to ASCLINn output selection register. n<28 , if interface exists.

Offset address: 061600_H+n*4

Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0											SEL				
r											rw				

Field	Bits	Type	Description
SEL	4:0	rw	**eGTM to ASCLINn trigger output selection** 00_H **None**: Not connected (safe state) 01_H **CDTM0_DTM5_0**: Dead-time output of ATOM0_4 02_H **CDTM1_DTM5_0**: Dead-time output of ATOM1_4 03_H **CDTM2_DTM5_0**: Dead-time output of ATOM2_4 04_H **CDTM0_DTM3_0**: Dead-time output of TOM0_12 05_H **CDTM1_DTM3_0**: Dead-time output of TOM1_12 06_H **CDTM0_DTM4_3**: Dead-time output of ATOM0_3 07_H **CDTM1_DTM4_3**: Dead-time output of ATOM1_3 08_H **CDTM2_DTM4_3**: Dead-time output of ATOM2_3 09_H **CDTM0_DTM2_3**: Dead-time output of TOM0_11 $0A_H$ **CDTM1_DTM2_3**: Dead-time output of TOM1_11 $0B_H$ **CDTM2_DTM3_0**: Dead-time output of TOM2_12 $0C_H$ **Reserved**: Reserved, do not use … 17_H **Reserved**: Reserved, do not use 18_H **CDTM2_DTM2_3**: Dead-time output of TOM2_11 19_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	31:5	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.146 eGTM to SENTn trigger x output selection register A (n=0-1)

For a description of this register, see 7617.

EGTM_SENTn_OUTA (n=0-1) Offset address: 061680_H+n*8

eGTM to SENTn trigger x output selection register A Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	SEL2					SEL1					SEL0				
r	rw					rw					rw				

Field	Bits	Type	Description
SELx (x=0)	5*x+4:5*x	rw	**eGTM to SENTn trigger x output selection** 00_H **None**: Not connected (safe state) 01_H **CDTM0_DTM4_0**: Dead-time output of ATOM0_0 02_H **CDTM0_DTM4_1**: Dead-time output of ATOM0_1 03_H **CDTM0_DTM4_2**: Dead-time output of ATOM0_2 04_H **CDTM0_DTM4_3**: Dead-time output of ATOM0_3 05_H **CDTM0_DTM5_0**: Dead-time output of ATOM0_4 06_H **CDTM0_DTM5_1**: Dead-time output of ATOM0_5 07_H **CDTM0_DTM5_2**: Dead-time output of ATOM0_6 08_H **CDTM0_DTM5_3**: Dead-time output of ATOM0_7 09_H **CDTM0_DTM0_0**: Dead-time output of TOM0_0 $0A_H$ **CDTM0_DTM0_1**: Dead-time output of TOM0_1 $0B_H$ **CDTM0_DTM0_2**: Dead-time output of TOM0_2 $0C_H$ **CDTM0_DTM0_3**: Dead-time output of TOM0_3 $0D_H$ **CDTM0_DTM1_0**: Dead-time output of TOM0_4 $0E_H$ **CDTM0_DTM1_1**: Dead-time output of TOM0_5 $0F_H$ **CDTM0_DTM1_2**: Dead-time output of TOM0_6 10_H **CDTM0_DTM1_3**: Dead-time output of TOM0_7 11_H **CDTM0_DTM2_0**: Dead-time output of TOM0_8 12_H **CDTM0_DTM2_1**: Dead-time output of TOM0_9 13_H **CDTM0_DTM2_2**: Dead-time output of TOM0_10 14_H **CDTM0_DTM2_3**: Dead-time output of TOM0_11 15_H **CDTM0_DTM3_0**: Dead-time output of TOM0_12 16_H **CDTM0_DTM3_1**: Dead-time output of TOM0_13 17_H **CDTM0_DTM3_2**: Dead-time output of TOM0_14 18_H **CDTM0_DTM3_3**: Dead-time output of TOM0_15 19_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=1)	5*x+4:5*x	rw	**eGTM to SENTn trigger x output selection** 00_H **None**: Not connected (safe state) 01_H **CDTM1_DTM4_0**: Dead-time output of ATOM1_0 02_H **CDTM1_DTM4_1**: Dead-time output of ATOM1_1 03_H **CDTM1_DTM4_2**: Dead-time output of ATOM1_2 04_H **CDTM1_DTM4_3**: Dead-time output of ATOM1_3 05_H **CDTM1_DTM5_0**: Dead-time output of ATOM1_4 06_H **CDTM1_DTM5_1**: Dead-time output of ATOM1_5 07_H **CDTM1_DTM5_2**: Dead-time output of ATOM1_6 08_H **CDTM1_DTM5_3**: Dead-time output of ATOM1_7 09_H **CDTM1_DTM0_0**: Dead-time output of TOM1_0 $0A_H$ **CDTM1_DTM0_1**: Dead-time output of TOM1_1 $0B_H$ **CDTM1_DTM0_2**: Dead-time output of TOM1_2 $0C_H$ **CDTM1_DTM0_3**: Dead-time output of TOM1_3 $0D_H$ **CDTM1_DTM1_0**: Dead-time output of TOM1_4 $0E_H$ **CDTM1_DTM1_1**: Dead-time output of TOM1_5 $0F_H$ **CDTM1_DTM1_2**: Dead-time output of TOM1_6 10_H **CDTM1_DTM1_3**: Dead-time output of TOM1_7 11_H **CDTM1_DTM2_0**: Dead-time output of TOM1_8 12_H **CDTM1_DTM2_1**: Dead-time output of TOM1_9 13_H **CDTM1_DTM2_2**: Dead-time output of TOM1_10 14_H **CDTM1_DTM2_3**: Dead-time output of TOM1_11 15_H **CDTM1_DTM3_0**: Dead-time output of TOM1_12 16_H **CDTM1_DTM3_1**: Dead-time output of TOM1_13 17_H **CDTM1_DTM3_2**: Dead-time output of TOM1_14 18_H **CDTM1_DTM3_3**: Dead-time output of TOM1_15 19_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=2)	5*x+4:5*x	rw	**eGTM to SENTn trigger x output selection** 00_H **None**: Not connected (safe state) 01_H **CDTM2_DTM4_0**: Dead-time output of ATOM2_0 02_H **CDTM2_DTM4_1**: Dead-time output of ATOM2_1 03_H **CDTM2_DTM4_2**: Dead-time output of ATOM2_2 04_H **CDTM2_DTM4_3**: Dead-time output of ATOM2_3 05_H **CDTM2_DTM5_0**: Dead-time output of ATOM2_4 06_H **CDTM2_DTM5_1**: Dead-time output of ATOM2_5 07_H **CDTM2_DTM5_2**: Dead-time output of ATOM2_6 08_H **CDTM2_DTM5_3**: Dead-time output of ATOM2_7 09_H **CDTM2_DTM0_0**: Dead-time output of TOM2_0 $0A_H$ **CDTM2_DTM0_1**: Dead-time output of TOM2_1 $0B_H$ **CDTM2_DTM0_2**: Dead-time output of TOM2_2 $0C_H$ **CDTM2_DTM0_3**: Dead-time output of TOM2_3 $0D_H$ **CDTM2_DTM1_0**: Dead-time output of TOM2_4 $0E_H$ **CDTM2_DTM1_1**: Dead-time output of TOM2_5 $0F_H$ **CDTM2_DTM1_2**: Dead-time output of TOM2_6 10_H **CDTM2_DTM1_3**: Dead-time output of TOM2_7 11_H **CDTM2_DTM2_0**: Dead-time output of TOM2_8 12_H **CDTM2_DTM2_1**: Dead-time output of TOM2_9 13_H **CDTM2_DTM2_2**: Dead-time output of TOM2_10 14_H **CDTM2_DTM2_3**: Dead-time output of TOM2_11 15_H **CDTM2_DTM3_0**: Dead-time output of TOM2_12 16_H **CDTM2_DTM3_1**: Dead-time output of TOM2_13 17_H **CDTM2_DTM3_2**: Dead-time output of TOM2_14 18_H **CDTM2_DTM3_3**: Dead-time output of TOM2_15 19_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	31:15	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.147 eGTM to ADC output selection register n (n=0)

For a description of this register, see 7934.

EGTM_ADC_OUTn (n=0) Offset address: 061700_H+n*4

eGTM to ADC output selection register n Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0			SEL3					0			SEL2				
r			rw					r			rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			SEL1					0			SEL0				
r			rw					r			rw				

Field	Bits	Type	Description
SELx (x=0-1)	8*x+4:8*x	rw	**eGTM to ADC trigger x output selection** 00_H **None**: Not connected (safe state) 01_H **CDTM0_DTM4_0**: Dead-time output of ATOM0_0 02_H **CDTM0_DTM4_1**: Dead-time output of ATOM0_1 03_H **CDTM0_DTM4_2**: Dead-time output of ATOM0_2 04_H **CDTM0_DTM4_3**: Dead-time output of ATOM0_3 05_H **CDTM0_DTM5_0**: Dead-time output of ATOM0_4 06_H **CDTM0_DTM5_1**: Dead-time output of ATOM0_5 07_H **CDTM0_DTM5_2**: Dead-time output of ATOM0_6 08_H **CDTM0_DTM5_3**: Dead-time output of ATOM0_7 09_H **CDTM0_DTM0_0**: Dead-time output of TOM0_0 $0A_H$ **CDTM0_DTM0_1**: Dead-time output of TOM0_1 $0B_H$ **CDTM0_DTM0_2**: Dead-time output of TOM0_2 $0C_H$ **CDTM0_DTM0_3**: Dead-time output of TOM0_3 $0D_H$ **CDTM0_DTM1_0**: Dead-time output of TOM0_4 $0E_H$ **CDTM0_DTM1_1**: Dead-time output of TOM0_5 $0F_H$ **CDTM0_DTM1_2**: Dead-time output of TOM0_6 10_H **CDTM0_DTM1_3**: Dead-time output of TOM0_7 11_H **CDTM0_DTM2_0**: Dead-time output of TOM0_8 12_H **CDTM0_DTM2_1**: Dead-time output of TOM0_9 13_H **CDTM0_DTM2_2**: Dead-time output of TOM0_10 14_H **CDTM0_DTM2_3**: Dead-time output of TOM0_11 15_H **CDTM0_DTM3_0**: Dead-time output of TOM0_12 16_H **CDTM0_DTM3_1**: Dead-time output of TOM0_13 17_H **CDTM0_DTM3_2**: Dead-time output of TOM0_14 18_H **CDTM0_DTM3_3**: Dead-time output of TOM0_15 19_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=2-3)	8*x+4:8*x	rw	**eGTM to ADC trigger x output selection** 00_H **None**: Not connected (safe state) 01_H **CDTM0_DTM4_0**: Dead-time output of ATOM0_0 02_H **CDTM0_DTM4_1**: Dead-time output of ATOM0_1 03_H **CDTM0_DTM4_2**: Dead-time output of ATOM0_2 04_H **CDTM0_DTM4_3**: Dead-time output of ATOM0_3 05_H **CDTM0_DTM5_0**: Dead-time output of ATOM0_4 06_H **CDTM0_DTM5_1**: Dead-time output of ATOM0_5 07_H **CDTM0_DTM5_2**: Dead-time output of ATOM0_6 08_H **CDTM0_DTM5_3**: Dead-time output of ATOM0_7 09_H **CDTM0_DTM0_0**: Dead-time output of TOM0_0 $0A_H$ **CDTM0_DTM0_1**: Dead-time output of TOM0_1 $0B_H$ **CDTM0_DTM0_2**: Dead-time output of TOM0_2 $0C_H$ **CDTM0_DTM0_3**: Dead-time output of TOM0_3 $0D_H$ **CDTM0_DTM1_0**: Dead-time output of TOM0_4 $0E_H$ **CDTM0_DTM1_1**: Dead-time output of TOM0_5 $0F_H$ **CDTM0_DTM1_2**: Dead-time output of TOM0_6 10_H **CDTM0_DTM1_3**: Dead-time output of TOM0_7 11_H **CDTM0_DTM2_0**: Dead-time output of TOM0_8 12_H **CDTM0_DTM2_1**: Dead-time output of TOM0_9 13_H **CDTM0_DTM2_2**: Dead-time output of TOM0_10 14_H **CDTM0_DTM2_3**: Dead-time output of TOM0_11 15_H **CDTM0_DTM3_0**: Dead-time output of TOM0_12 16_H **CDTM0_DTM3_1**: Dead-time output of TOM0_13 17_H **CDTM0_DTM3_2**: Dead-time output of TOM0_14 18_H **CDTM0_DTM3_3**: Dead-time output of TOM0_15 19_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	31:29, 23:21, 15:13, 7:5	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.148 eGTM to ADC output selection register n (n=1)

For a description of this register, see 7934.

EGTM_ADC_OUTn (n=1) Offset address: 061700_H+n*4

eGTM to ADC output selection register n Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0			SEL3					0			SEL2				
r			rw					r			rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			SEL1					0			SEL0				
r			rw					r			rw				

Field	Bits	Type	Description
SELx (x=0-1)	8*x+4:8*x	rw	**eGTM to ADC trigger x output selection** 00_H **None**: Not connected (safe state) 01_H **CDTM1_DTM4_0**: Dead-time output of ATOM1_0 02_H **CDTM1_DTM4_1**: Dead-time output of ATOM1_1 03_H **CDTM1_DTM4_2**: Dead-time output of ATOM1_2 04_H **CDTM1_DTM4_3**: Dead-time output of ATOM1_3 05_H **CDTM1_DTM5_0**: Dead-time output of ATOM1_4 06_H **CDTM1_DTM5_1**: Dead-time output of ATOM1_5 07_H **CDTM1_DTM5_2**: Dead-time output of ATOM1_6 08_H **CDTM1_DTM5_3**: Dead-time output of ATOM1_7 09_H **CDTM1_DTM0_0**: Dead-time output of TOM1_0 $0A_H$ **CDTM1_DTM0_1**: Dead-time output of TOM1_1 $0B_H$ **CDTM1_DTM0_2**: Dead-time output of TOM1_2 $0C_H$ **CDTM1_DTM0_3**: Dead-time output of TOM1_3 $0D_H$ **CDTM1_DTM1_0**: Dead-time output of TOM1_4 $0E_H$ **CDTM1_DTM1_1**: Dead-time output of TOM1_5 $0F_H$ **CDTM1_DTM1_2**: Dead-time output of TOM1_6 10_H **CDTM1_DTM1_3**: Dead-time output of TOM1_7 11_H **CDTM1_DTM2_0**: Dead-time output of TOM1_8 12_H **CDTM1_DTM2_1**: Dead-time output of TOM1_9 13_H **CDTM1_DTM2_2**: Dead-time output of TOM1_10 14_H **CDTM1_DTM2_3**: Dead-time output of TOM1_11 15_H **CDTM1_DTM3_0**: Dead-time output of TOM1_12 16_H **CDTM1_DTM3_1**: Dead-time output of TOM1_13 17_H **CDTM1_DTM3_2**: Dead-time output of TOM1_14 18_H **CDTM1_DTM3_3**: Dead-time output of TOM1_15 19_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=2-3)	8*x+4:8*x	rw	**eGTM to ADC trigger x output selection** 00_H **None**: Not connected (safe state) 01_H **CDTM1_DTM4_0**: Dead-time output of ATOM1_0 02_H **CDTM1_DTM4_1**: Dead-time output of ATOM1_1 03_H **CDTM1_DTM4_2**: Dead-time output of ATOM1_2 04_H **CDTM1_DTM4_3**: Dead-time output of ATOM1_3 05_H **CDTM1_DTM5_0**: Dead-time output of ATOM1_4 06_H **CDTM1_DTM5_1**: Dead-time output of ATOM1_5 07_H **CDTM1_DTM5_2**: Dead-time output of ATOM1_6 08_H **CDTM1_DTM5_3**: Dead-time output of ATOM1_7 09_H **CDTM1_DTM0_0**: Dead-time output of TOM1_0 $0A_H$ **CDTM1_DTM0_1**: Dead-time output of TOM1_1 $0B_H$ **CDTM1_DTM0_2**: Dead-time output of TOM1_2 $0C_H$ **CDTM1_DTM0_3**: Dead-time output of TOM1_3 $0D_H$ **CDTM1_DTM1_0**: Dead-time output of TOM1_4 $0E_H$ **CDTM1_DTM1_1**: Dead-time output of TOM1_5 $0F_H$ **CDTM1_DTM1_2**: Dead-time output of TOM1_6 10_H **CDTM1_DTM1_3**: Dead-time output of TOM1_7 11_H **CDTM1_DTM2_0**: Dead-time output of TOM1_8 12_H **CDTM1_DTM2_1**: Dead-time output of TOM1_9 13_H **CDTM1_DTM2_2**: Dead-time output of TOM1_10 14_H **CDTM1_DTM2_3**: Dead-time output of TOM1_11 15_H **CDTM1_DTM3_0**: Dead-time output of TOM1_12 16_H **CDTM1_DTM3_1**: Dead-time output of TOM1_13 17_H **CDTM1_DTM3_2**: Dead-time output of TOM1_14 18_H **CDTM1_DTM3_3**: Dead-time output of TOM1_15 19_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	31:29, 23:21, 15:13, 7:5	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.149 eGTM to ADC output selection register n (n=2)

For a description of this register, see 7934.

EGTM_ADC_OUTn (n=2) Offset address: 061700_H+n*4

eGTM to ADC output selection register n Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0			SEL3					0			SEL2				
r			rw					r			rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			SEL1					0			SEL0				
r			rw					r			rw				

Field	Bits	Type	Description
SELx (x=0-1)	8*x+4:8*x	rw	**eGTM to ADC trigger x output selection** 00_H **None**: Not connected (safe state) 01_H **CDTM2_DTM4_0**: Dead-time output of ATOM2_0 02_H **CDTM2_DTM4_1**: Dead-time output of ATOM2_1 03_H **CDTM2_DTM4_2**: Dead-time output of ATOM2_2 04_H **CDTM2_DTM4_3**: Dead-time output of ATOM2_3 05_H **CDTM2_DTM5_0**: Dead-time output of ATOM2_4 06_H **CDTM2_DTM5_1**: Dead-time output of ATOM2_5 07_H **CDTM2_DTM5_2**: Dead-time output of ATOM2_6 08_H **CDTM2_DTM5_3**: Dead-time output of ATOM2_7 09_H **CDTM2_DTM0_0**: Dead-time output of TOM2_0 $0A_H$ **CDTM2_DTM0_1**: Dead-time output of TOM2_1 $0B_H$ **CDTM2_DTM0_2**: Dead-time output of TOM2_2 $0C_H$ **CDTM2_DTM0_3**: Dead-time output of TOM2_3 $0D_H$ **CDTM2_DTM1_0**: Dead-time output of TOM2_4 $0E_H$ **CDTM2_DTM1_1**: Dead-time output of TOM2_5 $0F_H$ **CDTM2_DTM1_2**: Dead-time output of TOM2_6 10_H **CDTM2_DTM1_3**: Dead-time output of TOM2_7 11_H **CDTM2_DTM2_0**: Dead-time output of TOM2_8 12_H **CDTM2_DTM2_1**: Dead-time output of TOM2_9 13_H **CDTM2_DTM2_2**: Dead-time output of TOM2_10 14_H **CDTM2_DTM2_3**: Dead-time output of TOM2_11 15_H **CDTM2_DTM3_0**: Dead-time output of TOM2_12 16_H **CDTM2_DTM3_1**: Dead-time output of TOM2_13 17_H **CDTM2_DTM3_2**: Dead-time output of TOM2_14 18_H **CDTM2_DTM3_3**: Dead-time output of TOM2_15 19_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=2-3)	8*x+4:8*x	rw	**eGTM to ADC trigger x output selection** 00_H **None**: Not connected (safe state) 01_H **CDTM2_DTM4_0**: Dead-time output of ATOM2_0 02_H **CDTM2_DTM4_1**: Dead-time output of ATOM2_1 03_H **CDTM2_DTM4_2**: Dead-time output of ATOM2_2 04_H **CDTM2_DTM4_3**: Dead-time output of ATOM2_3 05_H **CDTM2_DTM5_0**: Dead-time output of ATOM2_4 06_H **CDTM2_DTM5_1**: Dead-time output of ATOM2_5 07_H **CDTM2_DTM5_2**: Dead-time output of ATOM2_6 08_H **CDTM2_DTM5_3**: Dead-time output of ATOM2_7 09_H **CDTM2_DTM0_0**: Dead-time output of TOM2_0 $0A_H$ **CDTM2_DTM0_1**: Dead-time output of TOM2_1 $0B_H$ **CDTM2_DTM0_2**: Dead-time output of TOM2_2 $0C_H$ **CDTM2_DTM0_3**: Dead-time output of TOM2_3 $0D_H$ **CDTM2_DTM1_0**: Dead-time output of TOM2_4 $0E_H$ **CDTM2_DTM1_1**: Dead-time output of TOM2_5 $0F_H$ **CDTM2_DTM1_2**: Dead-time output of TOM2_6 10_H **CDTM2_DTM1_3**: Dead-time output of TOM2_7 11_H **CDTM2_DTM2_0**: Dead-time output of TOM2_8 12_H **CDTM2_DTM2_1**: Dead-time output of TOM2_9 13_H **CDTM2_DTM2_2**: Dead-time output of TOM2_10 14_H **CDTM2_DTM2_3**: Dead-time output of TOM2_11 15_H **CDTM2_DTM3_0**: Dead-time output of TOM2_12 16_H **CDTM2_DTM3_1**: Dead-time output of TOM2_13 17_H **CDTM2_DTM3_2**: Dead-time output of TOM2_14 18_H **CDTM2_DTM3_3**: Dead-time output of TOM2_15 19_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	31:29, 23:21, 15:13, 7:5	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.150 Cluster n DTMAUX input y selection register (y=0-1;n=0)

EGTM_DTMAUXn_INy (n=0;y=0-1) Offset address: 061780_H+n*8+y*4

Cluster n DTMAUX input y selection register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								0	SEL5			0	SEL4		
r								r	rw			r	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	SEL3			0	SEL2			0	SEL1			0	SEL0		
r	rw			r	rw			r	rw			r	rw		

Field	Bits	Type	Description
SELx (x=0)	4*x+2:4*x	rw	**Input selection for CDTMn_DTMx_AUXINy** 000_B **None**: Not connected (safe state) 001_B **P14.4**: Port Pad Input 010_B **P10.1**: Port Pad Input 011_B **P00.7**: Port Pad Input 100_B **GTBF0OUT0**: Boundary Flag, see ADC 101_B **P00.11**: Port Pad Input 110_B **P02.2**: Port Pad Input 111_B **P02.12**: Port Pad Input
SELx (x=1)	4*x+2:4*x	rw	**Input selection for CDTMn_DTMx_AUXINy** 000_B **None**: Not connected (safe state) 001_B **P14.4**: Port Pad Input 010_B **P10.3**: Port Pad Input 011_B **P00.7**: Port Pad Input 100_B **GTBF0OUT1**: Boundary Flag, see ADC 101_B **P00.10**: Port Pad Input 110_B **P02.1**: Port Pad Input 111_B **P02.10**: Port Pad Input
SELx (x=2)	4*x+2:4*x	rw	**Input selection for CDTMn_DTMx_AUXINy** 000_B **None**: Not connected (safe state) 001_B **P14.4**: Port Pad Input 010_B **P10.1**: Port Pad Input 011_B **P00.7**: Port Pad Input 100_B **GTBF0OUT2_4**: Boundary Flag, see ADC 101_B **P00.12**: Port Pad Input 110_B **P02.2**: Port Pad Input 111_B **P02.6**: Port Pad Input
SELx (x=3)	4*x+2:4*x	rw	**Input selection for CDTMn_DTMx_AUXINy** 000_B **None**: Not connected (safe state) 001_B **P14.4**: Port Pad Input 010_B **P10.3**: Port Pad Input 011_B **P00.7**: Port Pad Input 100_B **GTBF0OUT3_5**: Boundary Flag, see ADC 101_B **P00.12**: Port Pad Input 110_B **P02.3**: Port Pad Input 111_B **P02.12**: Port Pad Input

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=4)	4*x+2:4*x	rw	**Input selection for CDTMn_DTMx_AUXINy** 000_B **None**: Not connected (safe state) 001_B **P02.0**: Port Pad Input 010_B **P02.8**: Port Pad Input 011_B **P10.2**: Port Pad Input 100_B **GTBF0OUT2_4**: Boundary Flag, see ADC 101_B **P00.11**: Port Pad Input 110_B **P02.4**: Port Pad Input 111_B **P02.9**: Port Pad Input
SELx (x=5)	4*x+2:4*x	rw	**Input selection for CDTMn_DTMx_AUXINy** 000_B **None**: Not connected (safe state) 001_B **P02.6**: Port Pad Input 010_B **P02.9**: Port Pad Input 011_B **P10.2**: Port Pad Input 100_B **GTBF0OUT3_5**: Boundary Flag, see ADC 101_B **P00.11**: Port Pad Input 110_B **P02.12**: Port Pad Input 111_B **P02.11**: Port Pad Input
0	23, 19, 15, 11, 7, 3, 31:24	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.151 Cluster n DTMAUX input y selection register (y=0-1;n=1)

EGTM_DTMAUXn_INy (n=1;y=0-1) Offset address: 061780_H+n*8+y*4

Cluster n DTMAUX input y selection register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								0	SEL5			0	SEL4		
r								r	rw			r	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	SEL3			0	SEL2			0	SEL1			0	SEL0		
r	rw			r	rw			r	rw			r	rw		

Field	Bits	Type	Description
SELx (x=0)	4*x+2:4*x	rw	**Input selection for CDTMn_DTMx_AUXINy** 000_B **None**: Not connected (safe state) 001_B **P16.5**: Port Pad Input 010_B **P00.5**: Port Pad Input 011_B **P33.0**: Port Pad Input 100_B **GTBF1OUT0**: Boundary Flag, see ADC 101_B **P00.0**: Port Pad Input 110_B **P02.0**: Port Pad Input 111_B **P02.8**: Port Pad Input
SELx (x=1)	4*x+2:4*x	rw	**Input selection for CDTMn_DTMx_AUXINy** 000_B **None**: Not connected (safe state) 001_B **P16.5**: Port Pad Input 010_B **P00.5**: Port Pad Input 011_B **P33.0**: Port Pad Input 100_B **GTBF1OUT1**: Boundary Flag, see ADC 101_B **P00.1**: Port Pad Input 110_B **P02.1**: Port Pad Input 111_B **P02.7**: Port Pad Input
SELx (x=2)	4*x+2:4*x	rw	**Input selection for CDTMn_DTMx_AUXINy** 000_B **None**: Not connected (safe state) 001_B **P16.5**: Port Pad Input 010_B **P00.5**: Port Pad Input 011_B **P33.0**: Port Pad Input 100_B **GTBF1OUT2_4**: Boundary Flag, see ADC 101_B **P00.2**: Port Pad Input 110_B **P02.10**: Port Pad Input 111_B **P02.6**: Port Pad Input
SELx (x=3)	4*x+2:4*x	rw	**Input selection for CDTMn_DTMx_AUXINy** 000_B **None**: Not connected (safe state) 001_B **P16.5**: Port Pad Input 010_B **P00.8**: Port Pad Input 011_B **P33.0**: Port Pad Input 100_B **GTBF1OUT3_5**: Boundary Flag, see ADC 101_B **P00.2**: Port Pad Input 110_B **P02.3**: Port Pad Input 111_B **P02.5**: Port Pad Input

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=4)	4*x+2:4*x	rw	**Input selection for CDTMn_DTMx_AUXINy** 000_B **None**: Not connected (safe state) 001_B **P21.2**: Port Pad Input 010_B **P20.1**: Port Pad Input 011_B **P10.1**: Port Pad Input 100_B **GTBF1OUT2_4**: Boundary Flag, see ADC 101_B **P00.0**: Port Pad Input 110_B **P02.4**: Port Pad Input 111_B **P02.10**: Port Pad Input
SELx (x=5)	4*x+2:4*x	rw	**Input selection for CDTMn_DTMx_AUXINy** 000_B **None**: Not connected (safe state) 001_B **P21.2**: Port Pad Input 010_B **P20.1**: Port Pad Input 011_B **P10.1**: Port Pad Input 100_B **GTBF1OUT3_5**: Boundary Flag, see ADC 101_B **P00.3**: Port Pad Input 110_B **P02.5**: Port Pad Input 111_B **P02.11**: Port Pad Input
0	23, 19, 15, 11, 7, 3, 31:24	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.152 Cluster n DTMAUX input y selection register (y=0-1;n=2)

EGTM_DTMAUXn_INy (n=2;y=0-1) Offset address: 061780_H+n*8+y*4

Cluster n DTMAUX input y selection register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								0	SEL5			0	SEL4		
r								r	rw			r	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	SEL3			0	SEL2			0	SEL1			0	SEL0		
r	rw			r	rw			r	rw			r	rw		

Field	Bits	Type	Description
SELx (x=0)	4*x+2:4*x	rw	**Input selection for CDTMn_DTMx_AUXINy** 000_B **None**: Not connected (safe state) 001_B **P33.4**: Port Pad Input 010_B **P33.6**: Port Pad Input 011_B **P10.2**: Port Pad Input 100_B **GTBF2OUT0**: Boundary Flag, see ADC 101_B **P10.3**: Port Pad Input 110_B **P02.9**: Port Pad Input 111_B **P02.3**: Port Pad Input
SELx (x=1)	4*x+2:4*x	rw	**Input selection for CDTMn_DTMx_AUXINy** 000_B **None**: Not connected (safe state) 001_B **P33.4**: Port Pad Input 010_B **P33.6**: Port Pad Input 011_B **P10.2**: Port Pad Input 100_B **GTBF2OUT1**: Boundary Flag, see ADC 101_B **P10.1**: Port Pad Input 110_B **P02.10**: Port Pad Input 111_B **P02.2**: Port Pad Input
SELx (x=2)	4*x+2:4*x	rw	**Input selection for CDTMn_DTMx_AUXINy** 000_B **None**: Not connected (safe state) 001_B **P33.4**: Port Pad Input 010_B **P33.6**: Port Pad Input 011_B **P10.4**: Port Pad Input 100_B **GTBF2OUT2_4**: Boundary Flag, see ADC 101_B **P10.3**: Port Pad Input 110_B **P02.14**: Port Pad Input 111_B **P02.10**: Port Pad Input
SELx (x=3)	4*x+2:4*x	rw	**Input selection for CDTMn_DTMx_AUXINy** 000_B **None**: Not connected (safe state) 001_B **P33.4**: Port Pad Input 010_B **P33.6**: Port Pad Input 011_B **P10.3**: Port Pad Input 100_B **GTBF2OUT3_5**: Boundary Flag, see ADC 101_B **P10.4**: Port Pad Input 110_B **P02.7**: Port Pad Input 111_B **P02.9**: Port Pad Input

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=4)	4*x+2:4*x	rw	**Input selection for CDTMn_DTMx_AUXINy** 000_B **None**: Not connected (safe state) 001_B **P14.5**: Port Pad Input 010_B **P11.0**: Port Pad Input 011_B **Reserved**: Reserved, do not use 100_B **GTBF2OUT2_4**: Boundary Flag, see ADC 101_B **P10.5**: Port Pad Input 110_B **P02.6**: Port Pad Input 111_B **P02.9**: Port Pad Input
SELx (x=5)	4*x+2:4*x	rw	**Input selection for CDTMn_DTMx_AUXINy** 000_B **None**: Not connected (safe state) 001_B **P14.5**: Port Pad Input 010_B **P11.0**: Port Pad Input 011_B **Reserved**: Reserved, do not use 100_B **GTBF2OUT3_5**: Boundary Flag, see ADC 101_B **P10.6**: Port Pad Input 110_B **P02.5**: Port Pad Input 111_B **P02.4**: Port Pad Input
0	23, 19, 15, 11, 7, 3, 31:24	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.153 eGTM to SCU output selection register n eGTM_SCU_TRIG(n) (n=0)

For a description of this register, see 7593.

EGTM_SCU_OUTn (n=0)

eGTM to SCU output selection register n eGTM_SCU_TRIG(n)

Offset address: 061800_H+n*4

Kernel Reset value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0			SEL3					0			SEL2				
r			rw					r			rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			SEL1					0			SEL0				
r			rw					r			rw				

Field	Bits	Type	Description
SELx (x=0-1)	8*x+4:8*x	rw	**eGTM to SCU trigger x output selection eGTM_SCU_TRIG(x)** 00_H **None**: Not connected (safe state) 01_H **CDTM0_DTM4_0**: Dead-time output of ATOM0_0 02_H **CDTM0_DTM4_1**: Dead-time output of ATOM0_1 03_H **CDTM0_DTM4_2**: Dead-time output of ATOM0_2 04_H **CDTM0_DTM4_3**: Dead-time output of ATOM0_3 05_H **CDTM0_DTM5_0**: Dead-time output of ATOM0_4 06_H **CDTM0_DTM5_1**: Dead-time output of ATOM0_5 07_H **CDTM0_DTM5_2**: Dead-time output of ATOM0_6 08_H **CDTM0_DTM5_3**: Dead-time output of ATOM0_7 09_H **CDTM0_DTM0_0**: Dead-time output of TOM0_0 $0A_H$ **CDTM0_DTM0_1**: Dead-time output of TOM0_1 $0B_H$ **CDTM0_DTM0_2**: Dead-time output of TOM0_2 $0C_H$ **CDTM0_DTM0_3**: Dead-time output of TOM0_3 $0D_H$ **CDTM0_DTM1_0**: Dead-time output of TOM0_4 $0E_H$ **CDTM0_DTM1_1**: Dead-time output of TOM0_5 $0F_H$ **CDTM0_DTM1_2**: Dead-time output of TOM0_6 10_H **CDTM0_DTM1_3**: Dead-time output of TOM0_7 11_H **CDTM0_DTM2_0**: Dead-time output of TOM0_8 12_H **CDTM0_DTM2_1**: Dead-time output of TOM0_9 13_H **CDTM0_DTM2_2**: Dead-time output of TOM0_10 14_H **CDTM0_DTM2_3**: Dead-time output of TOM0_11 15_H **CDTM0_DTM3_0**: Dead-time output of TOM0_12 16_H **CDTM0_DTM3_1**: Dead-time output of TOM0_13 17_H **CDTM0_DTM3_2**: Dead-time output of TOM0_14 18_H **CDTM0_DTM3_3**: Dead-time output of TOM0_15 19_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=2-3)	8*x+4:8*x	rw	**eGTM to SCU trigger x output selection eGTM_SCU_TRIG(x)** 00_H **None**: Not connected (safe state) 01_H **CDTM0_DTM4_0**: Dead-time output of ATOM0_0 02_H **CDTM0_DTM4_1**: Dead-time output of ATOM0_1 03_H **CDTM0_DTM4_2**: Dead-time output of ATOM0_2 04_H **CDTM0_DTM4_3**: Dead-time output of ATOM0_3 05_H **CDTM0_DTM5_0**: Dead-time output of ATOM0_4 06_H **CDTM0_DTM5_1**: Dead-time output of ATOM0_5 07_H **CDTM0_DTM5_2**: Dead-time output of ATOM0_6 08_H **CDTM0_DTM5_3**: Dead-time output of ATOM0_7 09_H **CDTM0_DTM0_0**: Dead-time output of TOM0_0 $0A_H$ **CDTM0_DTM0_1**: Dead-time output of TOM0_1 $0B_H$ **CDTM0_DTM0_2**: Dead-time output of TOM0_2 $0C_H$ **CDTM0_DTM0_3**: Dead-time output of TOM0_3 $0D_H$ **CDTM0_DTM1_0**: Dead-time output of TOM0_4 $0E_H$ **CDTM0_DTM1_1**: Dead-time output of TOM0_5 $0F_H$ **CDTM0_DTM1_2**: Dead-time output of TOM0_6 10_H **CDTM0_DTM1_3**: Dead-time output of TOM0_7 11_H **CDTM0_DTM2_0**: Dead-time output of TOM0_8 12_H **CDTM0_DTM2_1**: Dead-time output of TOM0_9 13_H **CDTM0_DTM2_2**: Dead-time output of TOM0_10 14_H **CDTM0_DTM2_3**: Dead-time output of TOM0_11 15_H **CDTM0_DTM3_0**: Dead-time output of TOM0_12 16_H **CDTM0_DTM3_1**: Dead-time output of TOM0_13 17_H **CDTM0_DTM3_2**: Dead-time output of TOM0_14 18_H **CDTM0_DTM3_3**: Dead-time output of TOM0_15 19_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	15:13, 7:5, 31:29, 23:21	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.154 eGTM to SCU output selection register n eGTM_SCU_TRIG(n) (n=1)

For a description of this register, see 7593.

EGTM_SCU_OUTn (n=1)

eGTM to SCU output selection register n eGTM_SCU_TRIG(n)

Offset address: 061800_H+n*4

Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0			SEL3					0			SEL2				
r			rw					r			rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			SEL1					0			SEL0				
r			rw					r			rw				

Field	Bits	Type	Description
SELx (x=0-1)	8*x+4:8*x	rw	**eGTM to SCU trigger x output selection eGTM_SCU_TRIG(x)** 00_H **None**: Not connected (safe state) 01_H **CDTM1_DTM4_0**: Dead-time output of ATOM1_0 02_H **CDTM1_DTM4_1**: Dead-time output of ATOM1_1 03_H **CDTM1_DTM4_2**: Dead-time output of ATOM1_2 04_H **CDTM1_DTM4_3**: Dead-time output of ATOM1_3 05_H **CDTM1_DTM5_0**: Dead-time output of ATOM1_4 06_H **CDTM1_DTM5_1**: Dead-time output of ATOM1_5 07_H **CDTM1_DTM5_2**: Dead-time output of ATOM1_6 08_H **CDTM1_DTM5_3**: Dead-time output of ATOM1_7 09_H **CDTM1_DTM0_0**: Dead-time output of TOM1_0 $0A_H$ **CDTM1_DTM0_1**: Dead-time output of TOM1_1 $0B_H$ **CDTM1_DTM0_2**: Dead-time output of TOM1_2 $0C_H$ **CDTM1_DTM0_3**: Dead-time output of TOM1_3 $0D_H$ **CDTM1_DTM1_0**: Dead-time output of TOM1_4 $0E_H$ **CDTM1_DTM1_1**: Dead-time output of TOM1_5 $0F_H$ **CDTM1_DTM1_2**: Dead-time output of TOM1_6 10_H **CDTM1_DTM1_3**: Dead-time output of TOM1_7 11_H **CDTM1_DTM2_0**: Dead-time output of TOM1_8 12_H **CDTM1_DTM2_1**: Dead-time output of TOM1_9 13_H **CDTM1_DTM2_2**: Dead-time output of TOM1_10 14_H **CDTM1_DTM2_3**: Dead-time output of TOM1_11 15_H **CDTM1_DTM3_0**: Dead-time output of TOM1_12 16_H **CDTM1_DTM3_1**: Dead-time output of TOM1_13 17_H **CDTM1_DTM3_2**: Dead-time output of TOM1_14 18_H **CDTM1_DTM3_3**: Dead-time output of TOM1_15 19_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=2-3)	8*x+4:8*x	rw	**eGTM to SCU trigger x output selection eGTM_SCU_TRIG(x)** 00_H **None**: Not connected (safe state) 01_H **CDTM1_DTM4_0**: Dead-time output of ATOM1_0 02_H **CDTM1_DTM4_1**: Dead-time output of ATOM1_1 03_H **CDTM1_DTM4_2**: Dead-time output of ATOM1_2 04_H **CDTM1_DTM4_3**: Dead-time output of ATOM1_3 05_H **CDTM1_DTM5_0**: Dead-time output of ATOM1_4 06_H **CDTM1_DTM5_1**: Dead-time output of ATOM1_5 07_H **CDTM1_DTM5_2**: Dead-time output of ATOM1_6 08_H **CDTM1_DTM5_3**: Dead-time output of ATOM1_7 09_H **CDTM1_DTM0_0**: Dead-time output of TOM1_0 $0A_H$ **CDTM1_DTM0_1**: Dead-time output of TOM1_1 $0B_H$ **CDTM1_DTM0_2**: Dead-time output of TOM1_2 $0C_H$ **CDTM1_DTM0_3**: Dead-time output of TOM1_3 $0D_H$ **CDTM1_DTM1_0**: Dead-time output of TOM1_4 $0E_H$ **CDTM1_DTM1_1**: Dead-time output of TOM1_5 $0F_H$ **CDTM1_DTM1_2**: Dead-time output of TOM1_6 10_H **CDTM1_DTM1_3**: Dead-time output of TOM1_7 11_H **CDTM1_DTM2_0**: Dead-time output of TOM1_8 12_H **CDTM1_DTM2_1**: Dead-time output of TOM1_9 13_H **CDTM1_DTM2_2**: Dead-time output of TOM1_10 14_H **CDTM1_DTM2_3**: Dead-time output of TOM1_11 15_H **CDTM1_DTM3_0**: Dead-time output of TOM1_12 16_H **CDTM1_DTM3_1**: Dead-time output of TOM1_13 17_H **CDTM1_DTM3_2**: Dead-time output of TOM1_14 18_H **CDTM1_DTM3_3**: Dead-time output of TOM1_15 19_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	15:13, 7:5, 31:29, 23:21	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.4.155 eGTM to SCU output selection register n eGTM_SCU_TRIG(n) (n=2)

For a description of this register, see 7593.

EGTM_SCU_OUTn (n=2)

eGTM to SCU output selection register n eGTM_SCU_TRIG(n)

Offset address: 061800_H+n*4

Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0			SEL3					0			SEL2				
r			rw					r			rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			SEL1					0			SEL0				
r			rw					r			rw				

Field	Bits	Type	Description
SELx (x=0-1)	8*x+4:8*x	rw	**eGTM to SCU trigger x output selection eGTM_SCU_TRIG(x)** 00_H **None**: Not connected (safe state) 01_H **CDTM2_DTM4_0**: Dead-time output of ATOM2_0 02_H **CDTM2_DTM4_1**: Dead-time output of ATOM2_1 03_H **CDTM2_DTM4_2**: Dead-time output of ATOM2_2 04_H **CDTM2_DTM4_3**: Dead-time output of ATOM2_3 05_H **CDTM2_DTM5_0**: Dead-time output of ATOM2_4 06_H **CDTM2_DTM5_1**: Dead-time output of ATOM2_5 07_H **CDTM2_DTM5_2**: Dead-time output of ATOM2_6 08_H **CDTM2_DTM5_3**: Dead-time output of ATOM2_7 09_H **CDTM2_DTM0_0**: Dead-time output of TOM2_0 $0A_H$ **CDTM2_DTM0_1**: Dead-time output of TOM2_1 $0B_H$ **CDTM2_DTM0_2**: Dead-time output of TOM2_2 $0C_H$ **CDTM2_DTM0_3**: Dead-time output of TOM2_3 $0D_H$ **CDTM2_DTM1_0**: Dead-time output of TOM2_4 $0E_H$ **CDTM2_DTM1_1**: Dead-time output of TOM2_5 $0F_H$ **CDTM2_DTM1_2**: Dead-time output of TOM2_6 10_H **CDTM2_DTM1_3**: Dead-time output of TOM2_7 11_H **CDTM2_DTM2_0**: Dead-time output of TOM2_8 12_H **CDTM2_DTM2_1**: Dead-time output of TOM2_9 13_H **CDTM2_DTM2_2**: Dead-time output of TOM2_10 14_H **CDTM2_DTM2_3**: Dead-time output of TOM2_11 15_H **CDTM2_DTM3_0**: Dead-time output of TOM2_12 16_H **CDTM2_DTM3_1**: Dead-time output of TOM2_13 17_H **CDTM2_DTM3_2**: Dead-time output of TOM2_14 18_H **CDTM2_DTM3_3**: Dead-time output of TOM2_15 19_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use

(table continues...)

(continued)

Field	Bits	Type	Description
SELx (x=2-3)	8*x+4:8*x	rw	**eGTM to SCU trigger x output selection eGTM_SCU_TRIG(x)** 00_H **None**: Not connected (safe state) 01_H **CDTM2_DTM4_0**: Dead-time output of ATOM2_0 02_H **CDTM2_DTM4_1**: Dead-time output of ATOM2_1 03_H **CDTM2_DTM4_2**: Dead-time output of ATOM2_2 04_H **CDTM2_DTM4_3**: Dead-time output of ATOM2_3 05_H **CDTM2_DTM5_0**: Dead-time output of ATOM2_4 06_H **CDTM2_DTM5_1**: Dead-time output of ATOM2_5 07_H **CDTM2_DTM5_2**: Dead-time output of ATOM2_6 08_H **CDTM2_DTM5_3**: Dead-time output of ATOM2_7 09_H **CDTM2_DTM0_0**: Dead-time output of TOM2_0 $0A_H$ **CDTM2_DTM0_1**: Dead-time output of TOM2_1 $0B_H$ **CDTM2_DTM0_2**: Dead-time output of TOM2_2 $0C_H$ **CDTM2_DTM0_3**: Dead-time output of TOM2_3 $0D_H$ **CDTM2_DTM1_0**: Dead-time output of TOM2_4 $0E_H$ **CDTM2_DTM1_1**: Dead-time output of TOM2_5 $0F_H$ **CDTM2_DTM1_2**: Dead-time output of TOM2_6 10_H **CDTM2_DTM1_3**: Dead-time output of TOM2_7 11_H **CDTM2_DTM2_0**: Dead-time output of TOM2_8 12_H **CDTM2_DTM2_1**: Dead-time output of TOM2_9 13_H **CDTM2_DTM2_2**: Dead-time output of TOM2_10 14_H **CDTM2_DTM2_3**: Dead-time output of TOM2_11 15_H **CDTM2_DTM3_0**: Dead-time output of TOM2_12 16_H **CDTM2_DTM3_1**: Dead-time output of TOM2_13 17_H **CDTM2_DTM3_2**: Dead-time output of TOM2_14 18_H **CDTM2_DTM3_3**: Dead-time output of TOM2_15 19_H **Reserved**: Reserved, do not use … $1F_H$ **Reserved**: Reserved, do not use
0	15:13, 7:5, 31:29, 23:21	r	**Reserved, must not be used.** Read as 0, shall be written with 0.

44.9.5 TC4Dx eGTM connectivity

The table below lists interfaces to and from this functional block to other blocks in the device.

Table 2370 List of eGTM interface signals

Interface signals	I/O	Description
CLOCK_EGTM_feGTM	In	eGTM clock input
LLI_EGTM_AXI32S	In	Bridge to eGTM
CAN0_EGTM_SRC_CANINT[15:12]	In	TIMx_INy_z: CANn_INT_O[15:12]; n=0..5; CANn interrupt outputs

(table continues...)

Table 2370 (continued) List of eGTM interface signals

Interface signals	I/O	Description
CAN1_EGTM_SRC_CANINT[15:12]	In	TIMx_INy_z: CANn_INT_O[15:12]; n=0..5; CANn interrupt outputs
CAN2_EGTM_SRC_CANINT[15:12]	In	TIMx_INy_z: CANn_INT_O[15:12]; n=0..5; CANn interrupt outputs
CAN3_EGTM_SRC_CANINT[15:12]	In	TIMx_INy_z: CANn_INT_O[15:12]; n=0..5; CANn interrupt outputs
CAN4_EGTM_SRC_CANINT[15:12]	In	TIMx_INy_z: CANn_INT_O[15:12]; n=0..5; CANn interrupt outputs
SCU_EGTM_ERU_TIMnINSEL	In	PDOUT[7:0]; SCU pattern detection outputs: TIMx_INy_z
PMS_EGTM_RTCOUT	In	PMS RTC Multiplexer Input. RTCOUT, output of the RTC
ERAY0_EGTM_MACROTICK	In	MT0; Macrotick-clock from CC (synchronous to FPI clock)
ERAY1_EGTM_MACROTICK	In	MT0; Macrotick-clock from CC (synchronous to FPI clock)
GETH0_EGTM_SRC_GETHPPS[3:0]	In	GETHPP[3:0]; Gigabit,Ethernet PPS interrupt
LETH0_EGTM_Pi_PPS[i=0:3]	In	LETHPP[3:0]; Lite Ethernet PPS interrupt
PMS_EGTM_WUTOUT	In	TIMx_INy_z: WUTUFLOW; Underflow output to support WUT calibration
PMS_EGTM_DCDCSYNCO	In	TIMx_INy_z: DCDC_SYNC; Sync signal from PMS for DCDC control
ADC_EGTM_SRC_TMADCxSR[5:0]x=[7:0]	In	TMADC SR information
PORTS_EGTM_RX_2_TIM	In	Port input signals
ADC_EGTM_CLUSTERx_BFLY[3:0]x=[2:0]	In	ETBFnOUTx; n=0..NTIM-1; x=0..5; Output from ADC multiplexer
PORTS_EGTM_RX_2_DTM	In	Port input signals
EGTM_PSI5_TRIG[5:0]	Out	eGTM timer output trigger for PSI5
EGTM_PSI5S0_TRIG[7:0]	Out	eGTM timer output trigger for PSI5_Sn
EGTM_ADC_TRIG[11:0]	Out	eGTM timer output trigger for ADC[59:48] called [11:0] on EGTM side.
EGTM_ASCLIN0_TRIG	Out	eGTM timer output trigger for ASCLINn
EGTM_ASCLIN1_TRIG	Out	eGTM timer output trigger for ASCLINn
EGTM_ASCLIN2_TRIG	Out	eGTM timer output trigger for ASCLINn
EGTM_ASCLIN3_TRIG	Out	eGTM timer output trigger for ASCLINn
EGTM_ASCLIN4_TRIG	Out	eGTM timer output trigger for ASCLINn
EGTM_ASCLIN5_TRIG	Out	eGTM timer output trigger for ASCLINn
EGTM_ASCLIN6_TRIG	Out	eGTM timer output trigger for ASCLINn
EGTM_ASCLIN7_TRIG	Out	eGTM timer output trigger for ASCLINn
EGTM_ASCLIN8_TRIG	Out	eGTM timer output trigger for ASCLINn

(table continues...)

Table 2370 (continued) List of eGTM interface signals

Interface signals	I/O	Description
EGTM_ASCLIN9_TRIG	Out	eGTM timer output trigger for ASCLINn
EGTM_ASCLIN10_TRIG	Out	eGTM timer output trigger for ASCLINn
EGTM_ASCLIN11_TRIG	Out	eGTM timer output trigger for ASCLINn
EGTM_ASCLIN12_TRIG	Out	eGTM timer output trigger for ASCLINn
EGTM_ASCLIN13_TRIG	Out	eGTM timer output trigger for ASCLINn
EGTM_ASCLIN14_TRIG	Out	eGTM timer output trigger for ASCLINn
EGTM_ASCLIN15_TRIG	Out	eGTM timer output trigger for ASCLINn
EGTM_ASCLIN16_TRIG	Out	eGTM timer output trigger for ASCLINn
EGTM_ASCLIN17_TRIG	Out	eGTM timer output trigger for ASCLINn
EGTM_ASCLIN18_TRIG	Out	eGTM timer output trigger for ASCLINn
EGTM_ASCLIN19_TRIG	Out	eGTM timer output trigger for ASCLINn
EGTM_ASCLIN20_TRIG	Out	eGTM timer output trigger for ASCLINn
EGTM_ASCLIN21_TRIG	Out	eGTM timer output trigger for ASCLINn
EGTM_ASCLIN22_TRIG	Out	eGTM timer output trigger for ASCLINn
EGTM_ASCLIN23_TRIG	Out	eGTM timer output trigger for ASCLINn
EGTM_ASCLIN24_TRIG	Out	eGTM timer output trigger for ASCLINn
EGTM_ASCLIN25_TRIG	Out	eGTM timer output trigger for ASCLINn
EGTM_ASCLIN26_TRIG	Out	eGTM timer output trigger for ASCLINn
EGTM_ASCLIN27_TRIG	Out	eGTM timer output trigger for ASCLINn
EGTM_CAN0_Nx_TRIG[3:0]	Out	eGTM timer output trigger for CANn
EGTM_CAN1_Nx_TRIG[3:0]	Out	eGTM timer output trigger for CANn
EGTM_CAN2_Nx_TRIG[3:0]	Out	eGTM timer output trigger for CANn
EGTM_CAN3_Nx_TRIG[3:0]	Out	eGTM timer output trigger for CANn
EGTM_CAN4_Nx_TRIG[3:0]	Out	eGTM timer output trigger for CANn
EGTM_QSPI0_TRIG	Out	eGTM timer output trigger for QSPIn
EGTM_QSPI1_TRIG	Out	eGTM timer output trigger for QSPIn
EGTM_QSPI2_TRIG	Out	eGTM timer output trigger for QSPIn
EGTM_QSPI3_TRIG	Out	eGTM timer output trigger for QSPIn
EGTM_QSPI4_TRIG	Out	eGTM timer output trigger for QSPIn
EGTM_QSPI5_TRIG	Out	eGTM timer output trigger for QSPIn
EGTM_QSPI6_TRIG	Out	eGTM timer output trigger for QSPIn
EGTM_QSPI7_TRIG	Out	eGTM timer output trigger for QSPIn
EGTM_SENT0_TRIG[2:0]	Out	eGTM timer output trigger for SENTn

(table continues...)

Table 2370 (continued) List of eGTM interface signals

Interface signals	I/O	Description
EGTM_SENT1_TRIG[2:0]	Out	eGTM timer output trigger for SENTn
EGTM_PORTS_TOUT[293:0]	Out	eGTM timer output signal for PORT
EGTM_SCU_TRIG[3:0]	Out	eGTM timer output trigger for SCUn: ERU: EGTM TIM output n=[2:0]
EGTM_MSC0_MSCSELIN_INLCON	Out	eGTM timer output vector for MSCn - low part
EGTM_MSC0_MSCSELIN_INHCON	Out	eGTM timer output vector for MSCn - low extension part
EGTM_MSC0_MSCSELIN_INLECON	Out	eGTM timer output vector for MSCn - high part
EGTM_MSC0_MSCSELIN_INHECON	Out	eGTM timer output vector for MSCn - high extension part
EGTM_PORTS_ECLK[2:0]	Out	eGTM CMU Clocks
SRI_EGTM_SIF	In	SRI Slave Interface for eGTM
CANXL_EGTM_MTI_TRIG[3:0]	In	TIMx_INy_z: CANXL interrupt outputs, N_NODES is set to 4
EGTM_PMS_DCDCSYNCI	Out	LCDC_OUT Sync signal from eGTM to PMS for DCDC control; eGTM timer output trigger for LCDCDC
EGTM_CANXL_TRIG_IN[3:0]	Out	eGTM timer output trigger for CANn

44.9.6 TC4Dx eGTM revision history

Reference	Description of change(s)
Date range: 2024-08-17 to 2025-04-16	
TC4Dx eGTM information	• Fixed typos, update configuration table, updated register descriptions.

45 Analog-to-Digital Converter (ADC)

The Analog-to-Digital Converter (ADC) is used to convert the analog input digital data and post- process the conversion result to extract different signal characteristics.

Depending on the device, ADC functions can include:

- Time-Multiplexed Analog-to-Digital Converter
- Fast Compare Comparator (FCC)
- Delta-Sigma Analog-to-Digital Converter (DSADC), including on-chip modulator and external modulator support
- Converter Digital Signal Processing (CDSP)

45.1 Feature list

- Analog supply support for 3.3 V and 5 V
- High number of analog inputs depending on variant and package. This can be extended with support for external analog multiplexers (EMUX)
- ASIL-D analog acquisition supported with redundant ADC channels
- Global alarms directly connected to Safety Management Unit (SMU) and Interrupt Router (IR)
- Support for individual and combined service requests
- TMADC with 12 bit resolution and direct result data interface to GTM
- Boundary flag connected from ADC to GTM and to eGTM
- Digital comparator with hysteresis support
- Synchronous sampling on all channels connected to TMADC
- DSADC with SNR up to 86 dB
- External modulator demodulation support with optional Manchester Decoding and clock recovery
- Programmable CDSP cores to implement and configure various functions

45.2 Functional overview

The ADC is built of several functions which are used for analog acquisition of external analog signals, the demodulation of an external PDM bit-stream, and post processing of the results. The super-set of functionality is documented here. Please refer to the device specific section for specific implementation details. At least one TMADC instance exists in all function subsets. Optional features are:

- Delta-Sigma ADC (DSADC)
- External modulator support (EXMOD)
- Fast Compare Comparator (FCC)
- Converter Digital Signal Processor (CDSP)
- ADC interface to GTM and eGTM (depending on the presence of GTM or eGTM)

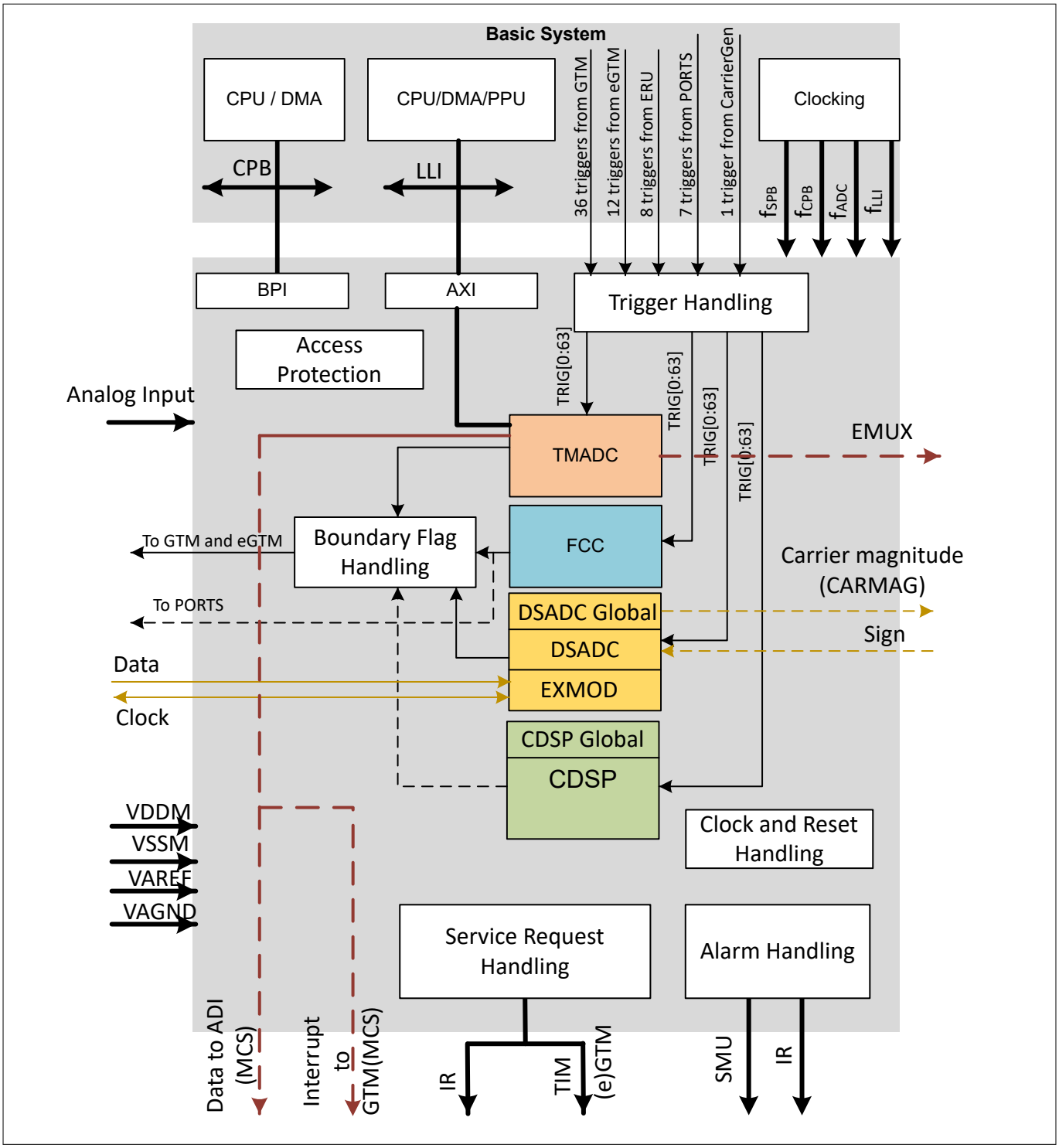


Figure 1183 ADC system interconnect

For the alarm description please refer to the alarm mapping tables in the SMU functional block and the Safety Manual.

The block diagram gives an overview of how the ADC and its sub-functions are connected to the rest of the system. Please note that:

- ADC has two bus slaves:
  - BPI slave connected to either CPB, COMPB or SPB.
    - All the ADC registers and memories are accessible through this interface

***Note**: Please check the device specific feature section which bus is connected.*

- AXI slave connected to LLI
  - Only the TMADC result, monitoring result, and timestamp registers can be read over this interface

  ***Note**: The Wait-for-read functionality is not supported over the AXI slave interface for TMADC*

  ***Note**: The LLI connection is not available in all device variants, please check the device specific feature section.*

- Boundary flags from different ADC sub-functions are handled centrally and are connected to GTM and eGTM. In addition, FCC boundary flags are also connected to ports
- TMADC receives on its monitoring sampler, low-precision band gap output voltage for conversion
- TMADC results RES14 and RES15 are accessible by every MCS(GTM) over the ADI interface. The availability of a new result on this interface could be signaled by sending an interrupt directly from ADC to MCS(GTM)
- TMADC supports the control of an external analog MUX with up to 8 analog inputs
- To support external modulator demodulation (EXMOD), it can receive the data as well as clock from an external modulator. As an option, EXMOD can also provide its clock to the external modulator
- The PWM signal for carrier generation is supported by DSADC and can receive the sign information from ports for carrier cancellation
- There are 8 alarms which are connected to SMU. The same alarms can be sent as service requests to IR

  ***Note**: The number of alarms vary from device to device*

- Each sub-function can directly send service requests to IR:
  - TMADC has 6 service requests per instance and 4 global service requests
  - FCC has 2 service requests per instance
  - DSADC has 4 service requests per instance
  - EXMOD has 4 service requests per instance
  - CDSP has 3 service requests per instance
- ADC has up to 64 input trigger lines, which are distributed to all the sub-functions. The sources of input triggers are:
  - Up to 36 triggers from GTM
  - Up to 12 triggers from eGTM
  - Up to 8 triggers from SCU(ERU)
  - Up to 7 triggers from ports
  - One trigger from carrier generator (DSADC)

  ***Note**: The number of triggers depends on the device configuration and the package variant used. The maximum number of all triggers is 64*

Related information

Trigger request control on page 8332
Sampling on page 8337
Conversion on page 8338
Conversion result handling on page 8344
Service request generation on page 8352
TC4Dx SMU alarm mapping tables on page 7233

45.3 Functional description

This section focuses on the central function within ADC, also referred as ADC top. Sub-functions inside ADC top, such as TMADC, are explained in subsequent sections.

The following functions are covered in this section:

- Register access protection, resource and reset allocation
- Global service request and alarm generation
- ADC interface to timer (GTM and eGTM)
- Initializing the ADC

45.3.1 Access protection, resource and reset allocation

This section explains how the access to different registers and memories of ADC is granted. A similar approach is followed for the reset of registers. The whole ADC is divided into resources to allow individual and independent configuration.

A resource within ADC is defined as a single instance of a sub-function. Each resource is assigned a unique resource ID.

The following table lists the resource ID assignment for specific ADC resources.

Table 2371 Resource ID for different resources within ADC

Resource ID	Resource	Resource allocation register	Registers included
00	ADC global registers (central function in ADC)	RESALLOC_GLOB	GL*, GT*, ETBF*, PDDCFG, SUPLLEV, TMSMOD, OCSTM
01 - 08	TMADC [0:7]	RESALLOC_TMADC	TMADC[0:7]*
09 - 16	FCC [0:7]	RESALLOC_FCCA	FCC[0:7]*
17 - 18	FCC [8:9]	RESALLOC_FCCB	FCC[8:9]*
19	DSADC and EXMOD global registers	RESALLOC_GLOB	DSADC_GLOB*
20 - 27	DSADC [0:7]	RESALLOC_DSADCA	DSADC[0:7]*
28 - 29	DSADC [8:9]	RESALLOC_DSADCB	DSADC[8:9]*
30 - 37	EXMOD [0:7]	RESALLOC_EXMOD	DSADC_EX[0:7]*
38	CDSP global registers	RESALLOC_GLOB	CDSP_GLOB*
39 - 56	CDSP [0:7]	RESALLOC_CDSPA	CDSP_DSP[0:7]*
	CDSP [8:15]	RESALLOC_CDSPB	CDSP_DSP[8:15]*
	CDSP [16:17]	RESALLOC_CDSPC	CDSP_DSP[16:17]*

The number of resources is product specific, but the resource ID is fixed for all products of the family. Example: EXMOD0 will have resource ID = 34 in all variants of the product family.

Access protection and reset

Read and write accesses to the ADC top registers and memories are controlled through the following registers:

- ACCENy_WRA
- ACCENy_WRB
- ACCENy_RDA

- ACCENy_RDB
- ACCENy_VM
- ACCENy_PRS

Access protection details can be found in the System interconnect chapter in the User Manual. To configure these registers, the protection has to be in an unlocked state, which is configurable through PROTSE and PROTEx.

There are 16 sets of ACCEN registers. Access is granted when PROTSE is configured in the unlocked state.

There are 16 reset control register sets available. Reset of an individual set is configured by the registers:

- RSTi_CTRLA
- RSTi_CTRLB

The status can be read through RSTi_STAT.

If RSTi_CTRL_A and RSTi_CTRL_B are set, all resources assigned to APUi are reset.

When the resource ID =0 (ADC global registers) is reset, it affects the functionality of other resources through global run control registers. Reset has to be asserted in a bottom-up manner. Before a global resource is reset, all dependent resources have to be reset to avoid unpredictable behavior.

***Note**: In the case that any enabled TMADC instance is assigned to the same APU as resource ADC GLOBAL, all enabled TMADC instances have to be assigned to the same APU so that the bottom-up manner kernel reset concept is not violated.*

Resource allocation to Access Protection Unit (APU)

You have the possibility to configure the assignment of different resources to different ACCEN and reset sets by configuring the resource allocation registers.

Each register is divided into up to 8 segments of 4 bits each. Each 4 bit segment is allocated to a fixed resource (resource name is written in the bit-field). The figure below shows the layout of resource allocation register 0.

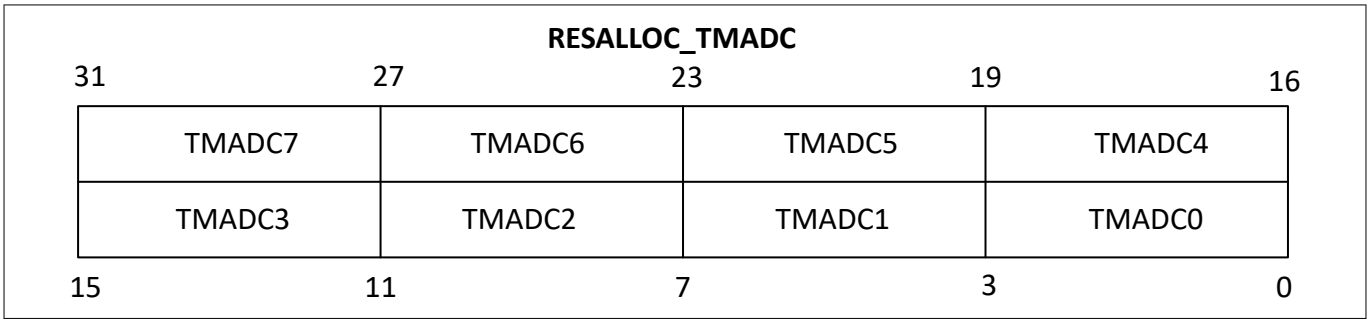


Figure 1184 Resource allocation register layout example

Use the 4-bit segment to assign a resource to an APU. For example if bits [3:0] (TMADC0) are written with 0010_B, then TMADC0 is access protected by configuration in ACCEN2, and reset by configuration in RST2. Multiple resources could be assigned to the same APU or a different one.

***Note**: There is no independent configuration for reset assignment to a resource. The same configuration as ACCEN assignment to a resource is also taken for reset sets.*

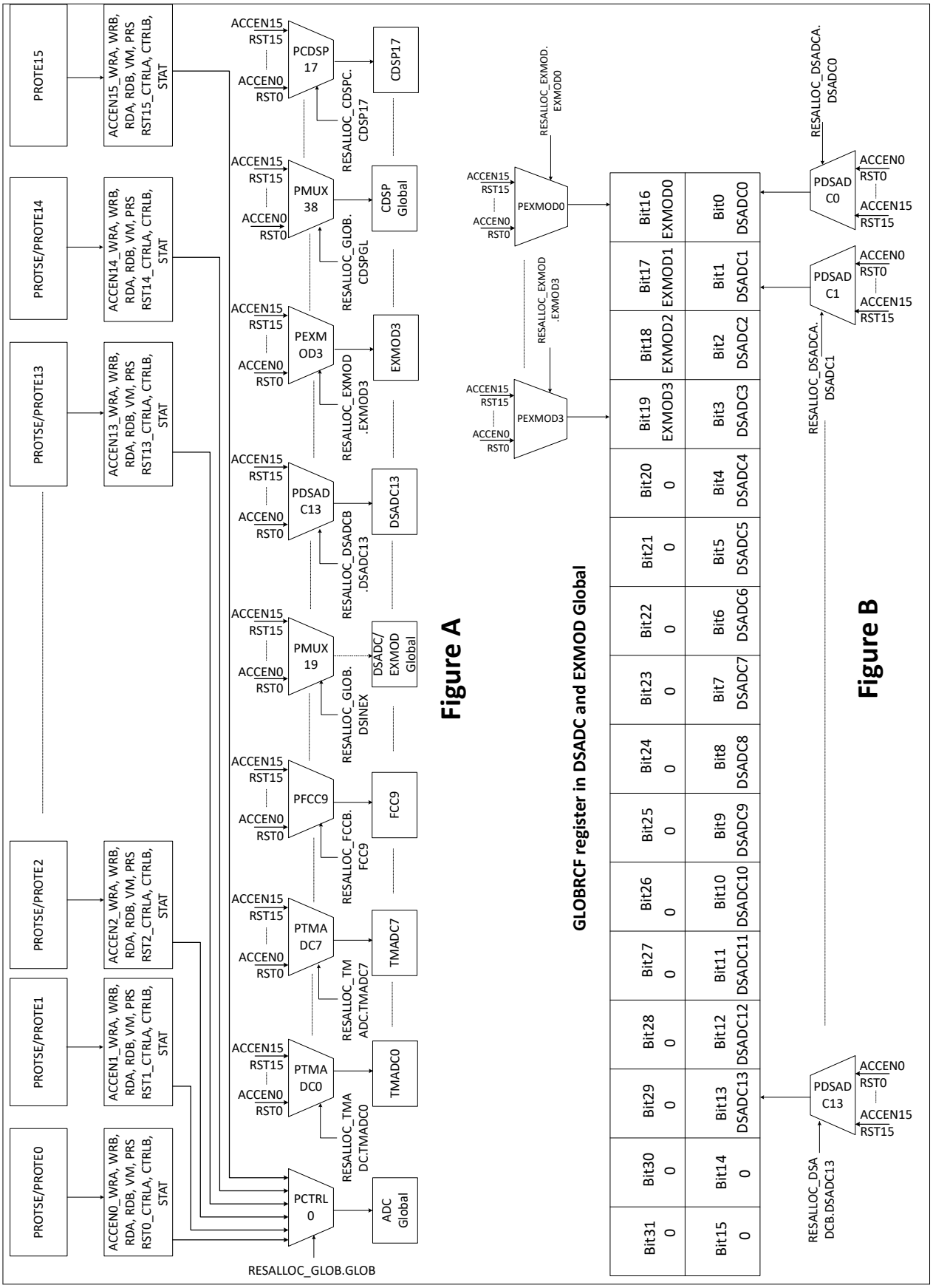


Figure 1185 Resource allocation concept

The Resource allocation concept figure shows the concept and structure of access protection and reset for different resources inside ADC. When an unauthorized master tries to access the registers and memories of a resource, it will result in a bus error.

***Note**: In the registers overview of the of the ADC registers chapter, it shows which ACCEN is used to control access to specific registers.*

In ADC there are three global resources:

- ADC global registers, where central registers of the ADC are defined
- DSADC and EXMOD global registers, where global registers of DSADC and EXMOD are defined
- CDSP global registers, where global registers of CDSP are defined

The global resources have a fixed resourced ID of 0, 19, and 38. ACCEN and reset allocation is controlled through programming registers RESALLOC_GLOB.

These global resources contain bit fields which impact the functionality of related resources. To provide more control of access protection and reset of the global resources, bit-field specific ACCEN and reset is introduced for some registers (Figure B).

Although the registers are physically in one of these global resources, ACCEN and reset of the register bits are controlled by the resource ID's named in the table below.

Table 2372 Bit-fields with individual access protection

Bit-field	Resource ID
CLKEN_TMADC.TMADCEN	8:1
CLKEN_FCC.FCCEN[9:0]	18:9
CLKEN_DSINEX.DSINEN	33:20
CLKEN_DSINEX.DSENEX	37:34
CLKEN_DSINEX.DSADCEN	19
CLKEN_CDSP.CDSPGL	38
CLKEN_CDSP.CDSPEN	56:39
CDSP_BNDCF.BNDF	56:39
CDSP_BNDCFCL.BNBDFCLR	56:39
CDSP_RESEVy.RESEV	56:39
CDSP_RESEVCLRy.RESEVCLR	56:39
CDSP_WUERREV.WUERREV	56:39
CDSP_WUERRCL.WUERRCLR	56:39
CDSP_GLOBRD.DSPRST	56:39
CDSP_GLOBRCD.CHRUND	56:39
CDSP_GLOBRSD.DSPRST	56:39
CDSP_GLOBHCD.CHHLTD	56:39
CDSP_GLOBHSTD.DSPHST	56:39
CDSP_SLPST.DSPSLP	56:39
CDSP_OCSCDSPA.SUS	47:39
CDSP_OCSCDSPA.SUSSTA	47:39
CDSP_OCSCDSPB.SUS	56:48
CDSP_OCSCDSPB.SUSSTA	56:48

(table continues...)

Table 2372 (continued) Bit-fields with individual access protection

Bit-field	Resource ID
DSADC_GLOBRCF.CHRUN	33:20
DSADC_GLOBRCF.CHRUNE	37:34
DSADC_GLOBRCM.CHRUNM	33:20
DSADC_BOUNDF.BNDEVEX	37:34
DSADC_BOUNDF.BNDEXIN	33:20
DSADC_BOUNDFCL.BNDEXCL	37:34
DSADC_BOUNDFCL.BNDINCL	33:20
DSADC_RESMF.REMEVEX	37:34
DSADC_RESMF.REMEVIN	33:20
DSADC_RESAF.REAEVEX	37:34
DSADC_RESAF.REAEVIN	33:20
DSADC_RESMFCL.REMEXCL	37:34
DSADC_RESMFCL.REMINCL	33:20
DSADC_RESAFCL.REAEXCL	37:34
DSADC_RESAFCL.REAINCL	33:20

***Note**: When an unauthorized master writes, reads, or resets the register bit-fields, only bit-fields where access is granted can be read or written or reset. Other bit-fields will read as 0 and will be returned with no-error. Write and reset will not affect the bit-fields. No bus error will be issued.*

45.3.2 Initializing the ADC

This section explains the steps to be followed before enabling any of the ADC sub-functions.

- Enable the ADC clock by writing into register bit-filed CLC.DISR =0. Once enabled, all the ADC global registers are accessible
- Allocate the ADC resources to their ACCEN and reset sets
- Enable the clocks of resources by writing to registers:
  - CLKEN_TMADC
  - CLKEN_FCC
  - CLKEN_DSINEX
  - CLKEN_CDSP

  ***Note**: Enable the clock of DSADC global, EXMOD global, or CDSP global, if any respective DSADC, EXMOD, or CDSP instance is used in any application*

- Wait 10 SPB/CPB/COMPB clock cycles. Enabling of module clocks after setting the corresponding bit in an ADC_CLKEN_* register can take up to 10 peripheral bus clock cycles
- Select the supply level for ADC depending on the VDDM used in the application by configuring the bit-fields:
  - SUPLLEV.TMADCSUP
  - SUPLLEV.FCCSUP
  - DSADC_GLOBCFG.DSADCSUP

***Note**: Since all ADCs are connected to the same VDDM, the supply setting for all ADC sub-functions is the same*

The configuration registers within different resources can now be configured.

45.3.3 Global service request and alarm generation

This section explains the configuration and generation of global service requests and alarms.

Global service request

Use the global service request generator to reduce the number of service requests of the TMADC instances. SR6 of each TMADCx is connected to the global service request generator.

- Use GLSRCFGi.EVEN to enable the use of the global service request of a TMADCx
- Set GLSRCFGi.EVOP to generate a global service request when all TMADCx.SR6 are set
- Clear GLSRCFGi.EVOP to generate a global service request for each TMADCx.SR6

For example, by writing 00001101_B in the bit field EVEN, SR6 from TMADC0, TMADC2, and TMADC3 are enabled for combining. If EVOP =1, a global service request is generated when all the SR6's from the respective TMADCs have arrived.

***Note**: SR6 coming from TMADCx is latched, before doing a logical operation. A global service request is active for 1 f_{SPB} clock period. The latches are reset after 1 f_{SPB} clock period or when GLSRCFG.EVEN or GLSRCFG.EVOP are reconfigured.*

Write to GLSRSET to trigger the global service requests for debug purposes.

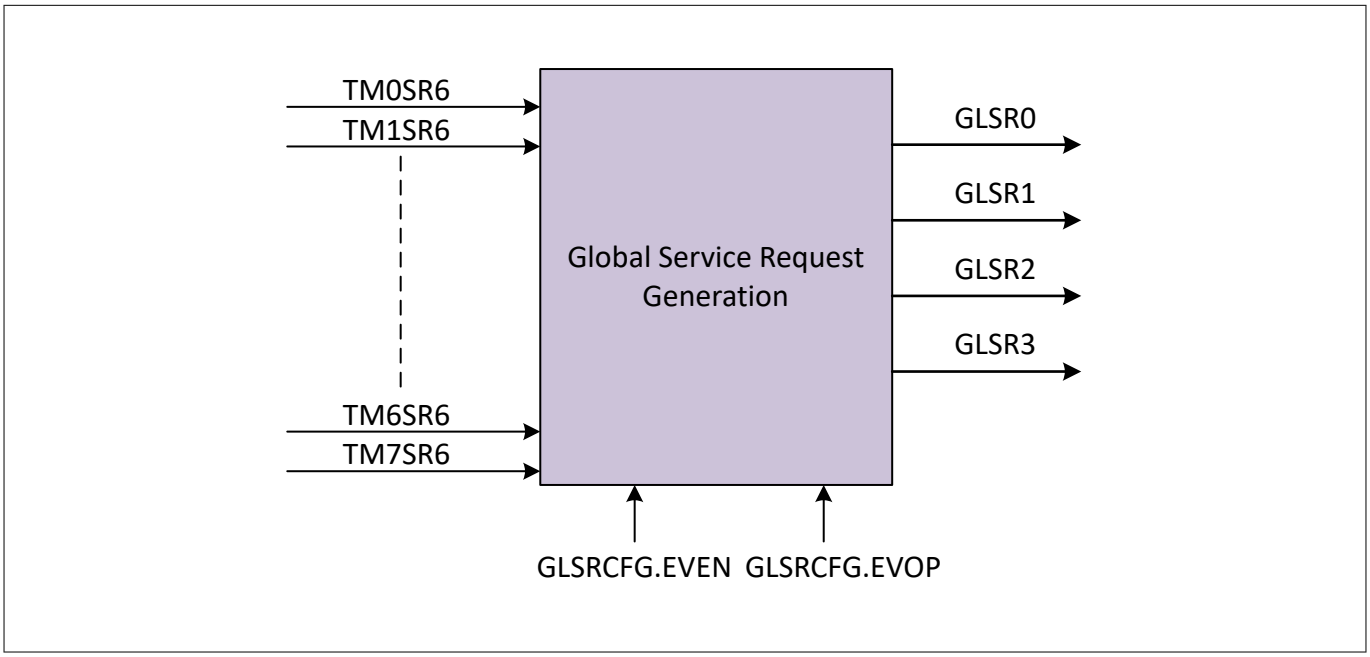


Figure 1186 Global service request generation

Global alarms

Unlike global service requests, global alarms can be used to combine 2 boundary flags of any ADC sub-module.

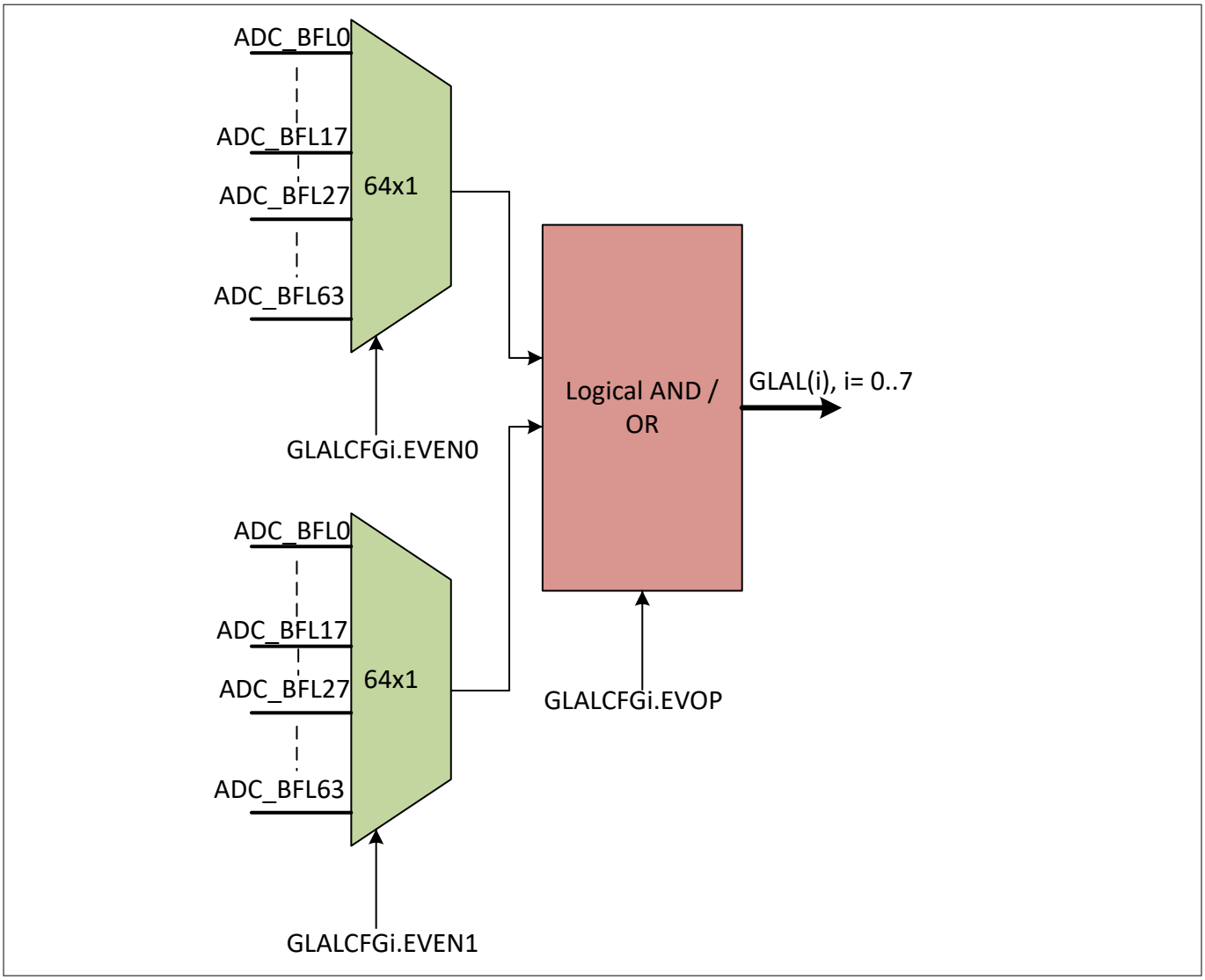


Figure 1187 Global alarm generation

The sources of the boundary flags are from TMADC, DSADC, and EXMOD channels, FCC and CDSP. Please refer to table below for the boundary flag mapping from different instances to ADC top:

Table 2373 Boundary flag mapping

Boundary flag from module	Boundary flag at ADC
TMADCx_BFLy ; x =[7:0], y =[1:0]	ADC_BFLz ; z=[15:0]
FCCx_BFL ; x =[9:0]	ADC_BFLz ; z=[25:16]
DSADCx_BFL ; x =[9:0]	ADC_BFLz ; z=[35:26]
EXMODx_BFL ; x=[7:0]	ADC_BFLz ; z=[43:36]
CDSPx_BFL ; x=[17:0]	ADC_BFLz ; z=[61:44]
Reserved	ADC_BFL62 and ADC_BFL63

Specific device variants offer 2 additional boundary flag comparators per CDSP. Selection for global alarms is shown in the picture below.

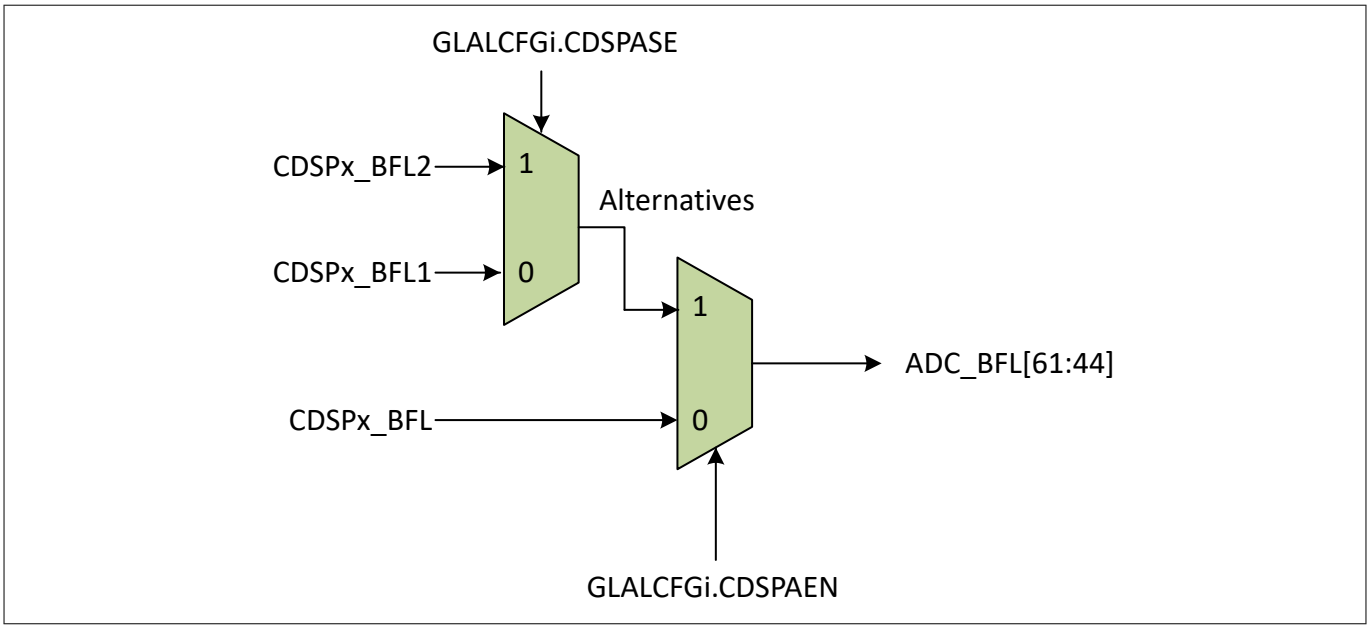


Figure 1188 Optional selection of additional CDSP boundary flags

GLALCFGi.CDSPAEN= 1_B maps in the additional CDSP boundary flags. GLALCFGi.CDSPASE does select between the 2 additional boundary flag sources. The device specific appendix will name, if the additional boundary flag comparators are available.

We configure alarms using GLALCFGi. Two boundary flags can be selected by configuring bit-fields EVEN0 and EVEN1. You can decide to generate an alarm under conditions configured through bit-field EVOP.

***Note**: There are up to 8 alarms which can be configured by the user. Alarms go as pulses into the system. Pulse width is at least 1 fSPB clock period. The same alarms are connected to IR as well as to SMU.*

You can initiate a test of the alarm connection by setting the corresponding bit of bit-field ALSET in the GLALSET register.

Related information

Service request generation on page 8352

45.3.4 Hardware triggers to ADC

There are 5 sources for hardware triggers inside the microcontroller to trigger different functions inside ADC top. These sources are GTM, eGTM, PORTS, SCU(ERU), and the Carrier Generator (DSADC)

Table 2374 Hardware triggers to ADC

Trigger number	Source	Source trigger connection	In case no GTM/eGTM available
TRIG[3:0]	GTM Cluster 0	GTM_ADC_TRIG[3:0]	GPT12_ADC_TRIG[3:0] (GPT12[2:5], T6OUT signal connected)
TRIG[7:4]	GTM Cluster 1	GTM_ADC_TRIG[7:4]	
TRIG[11:8]	GTM Cluster 2	GTM_ADC_TRIG[11:8]	
TRIG[15:12]	GTM Cluster 3	GTM_ADC_TRIG[15:12]	
TRIG[19:16]	GTM Cluster 4	GTM_ADC_TRIG[19:16]	

(table continues...)

Table 2374 (continued) Hardware triggers to ADC

Trigger number	Source	Source trigger connection	In case no GTM/eGTM available
TRIG[23:20]	GTM Cluster 5	GTM_ADC_TRIG[23:20]	
TRIG[25:24]	GTM Cluster 6	GTM_ADC_TRIG[25:24]	
TRIG[27:26]	GTM Cluster 7	GTM_ADC_TRIG[29:28]	
TRIG[29:28]	GTM Cluster 8	GTM_ADC_TRIG[33:32]	
TRIG[31:30]	GTM Cluster 9	GTM_ADC_TRIG[37:36]	
TRIG[33:32]	GTM Cluster 10	GTM_ADC_TRIG[41:40]	
TRIG[35:34]	GTM Cluster 11	GTM_ADC_TRIG[45:44]	
TRIG[39:36]	eGTM Cluster 0	eGTM_ADC_TRIG[51:48]	
TRIG[43:40]	eGTM Cluster 1	eGTM_ADC_TRIG[55:52]	
TRIG[47:44]	eGTM Cluster 2	eGTM_ADC_TRIG[59:56]	
TRIG[55:48]	SCU (ERU)	SCU_ADC_TRIG[7:0]	
TRIG[62:56]	PORTS	PORTS_ADC_TRIG[6:0]	
TRIG[63]	Carrier Generator (DSADC), see section "Trigger Generation" in chapter "Carrier Signal Generation"	Internal Signal from DSADC	

The defined hardware triggers are routed to TMADC, FCC, DSADC, EXMOD, and CDSP channels. The same trigger can be mapped to control multiple functions of the ADCs. For example the end of sampling on all or a part of the analog inputs on TMADCs synchronously.

***Note**: Please refer to the GTM and eGTM documentation to select TOM and ATOM channel(s) of a cluster which will generate the corresponding trigger(s).*

***Note**: Please refer to the SCU(ERU) documentation for the configuration of ERU triggers to ADC.*

***Note**: Depending on the device configuration, some triggers might not be available. Please refer to device specific documentation for the available registers, connectivity, and the mapped PORTS pins for triggering.*

45.3.5 ADC interface to GTM and eGTM

This section explains the interface between ADC top and Timers (GTM and eGTM). There is no interface from ADC to GPT12.

There are 5 interfaces existing between ADC and the timers GTM and eGTM:

- Triggers coming from GTM and eGTM to ADC
- Boundary Flags from different ADC modules to GTM and eGTM
- Data from TMADC to the GTM (ADI Interface)
- Service Requests from ADC to GTM (MCS)
- Service Requests from ADC to GTM and eGTM (TIM)

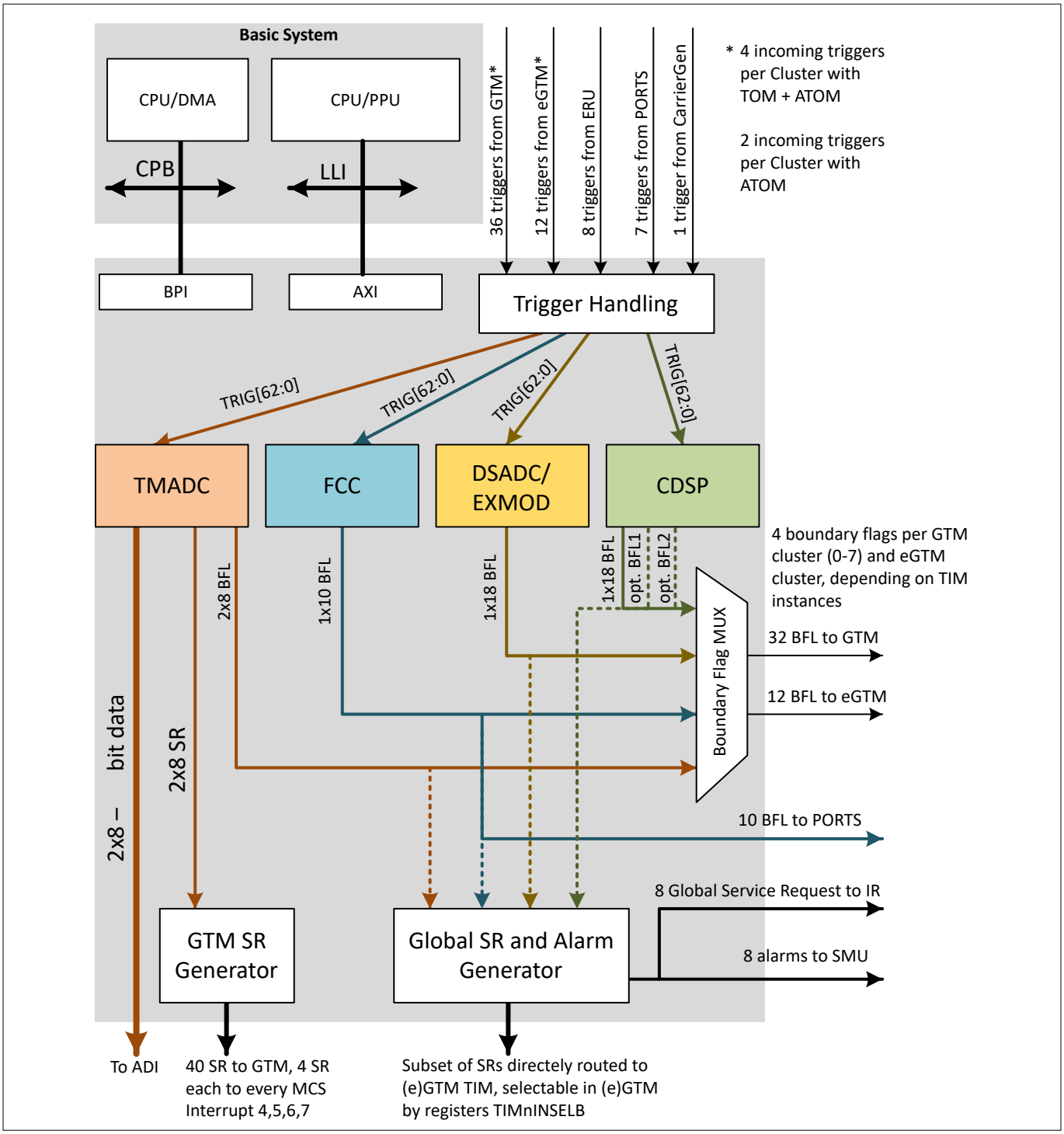


Figure 1189 ADC interface to timer (GTM and eGTM)

The figure shows the interface between ADC, GTM, and eGTM.
Every GTM and eGTM cluster with ATOM and TOM can output 4 dedicated triggers to ADC.
Every GTM and eGTM cluster with ATOM but not TOM, can output 2 triggers to ADC.
7 triggers come from PORTS directly to ADC.

***Note**: Please refer to the variant specific data-sheet for the appropriate PORT information*

Eight triggers come from SCU (ERU) to ADC, and you can use a glitch filter inside ERU to filter glitches on the trigger coming from PORTS before ERU sends a trigger to ADC. There is also one trigger signal from the carrier generator within DSADC to all of the ADC module.

45.3.5.1 Boundary flag interface to timer

This section explains the Boundary Flag interface between ADC top and Timer (GTM and eGTM).

Because of the real-time performance and deterministic functionality that several applications require, boundary flags generated inside ADC top must be directly accessible by the timer resources, GTM and eGTM respectively.

Up to 64 flags exist within ADC top. ADC top provides a configurable interface, which allows individual selection of boundary flags for each cluster of timers. The number of available boundary flags depends on the specific device variant.

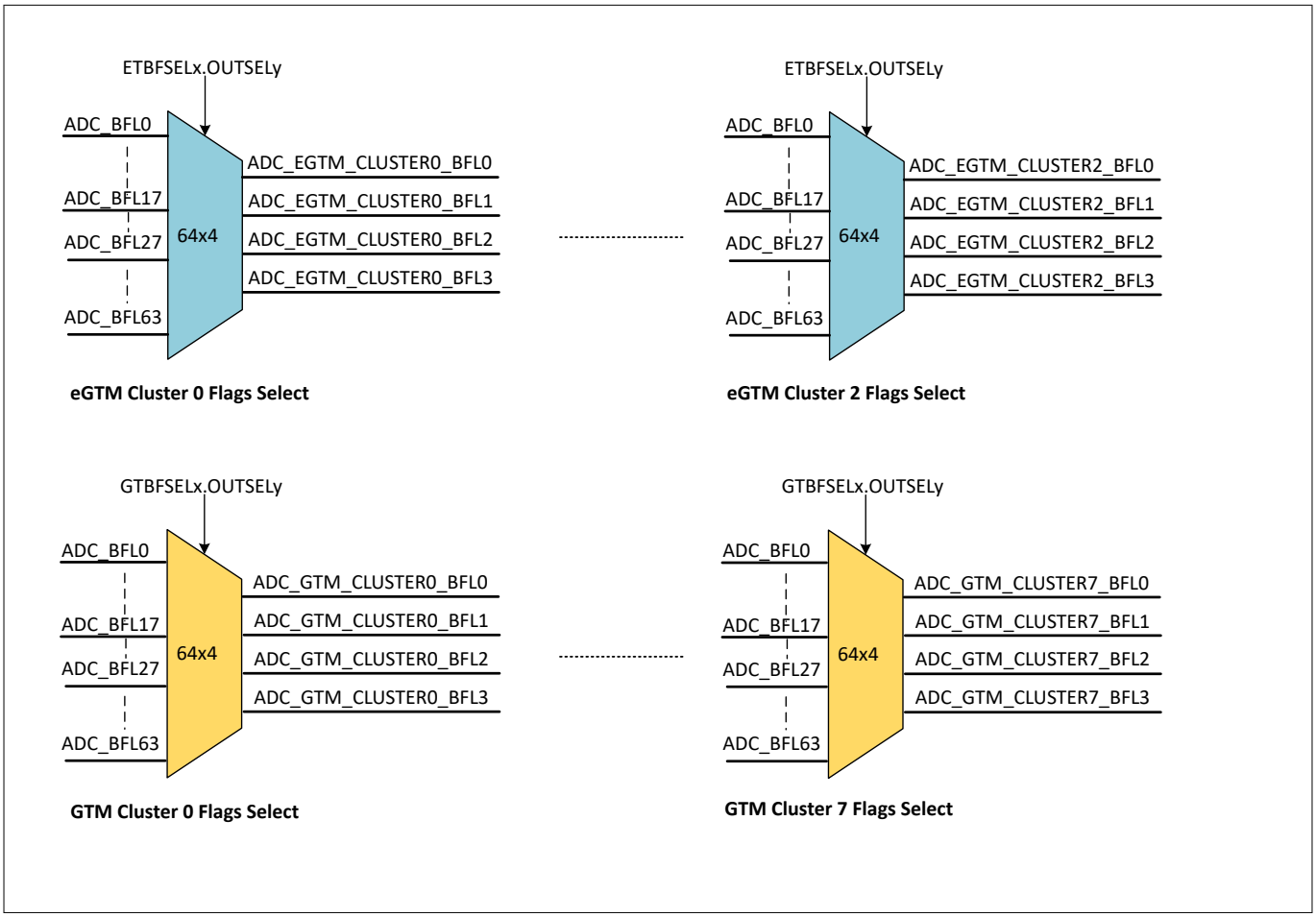


Figure 1190 Boundary flag selection for GTM and eGTM clusters

4 boundary flags can be connected to each cluster of eGTM and up to 8 clusters of GTM. Only the clusters with TIM are supported. The selection of the boundary flags for eGTM clusters is configured through the register ETBFxSEL.OUTnSEL and for GTM clusters through the register GTBFxSEL.OUTnSEL.

OUT0SEL, OUT1SEL, OUT2SEL, and OUT3SEL can be used to assign up to 4 BFL outputs to a single eGTM or GTM cluster.

***Note**: Within GTM and eGTM clusters, boundary flags are connected to TIM and DTM channels. Please refer to the GTM and eGTM chapters for more details.*

The mapping of the boundary flags of the specific ADC instances to ADC_BFL[63:0] available at ADC top can be found in chapter 'Global service request and alarm generation'.

Like described in Global service request and alarm generation, some device variants offer 2 additional boundary flag comparators per CDSP. The overrides for the boundary flags connected to GTM and eGTM are individual per eGTM and GTM cluster.

If bits ETBFxSEL.CDSPAEN[3:0] or GTBFxSEL.CDSPAEN[3:0] are set, the corresponding alternative CDSP boundary flags are mapped to eGTM or GTM.

ETBFxSEL.CDSPASE[3:0] and GTBFxSEL.CDSPASE[3:0] do select, what alternative comparator is used. If bits are set to 1_B the flags from CDSP_DSPa_BNDCFG2 are used. If set to 0_B the flags from CDSP_DSPa_BNDCFG1 are used.

The device specific appendix will name, if the additional boundary flag comparators are available.

45.3.5.2 Service request from ADC to GTM

This section explains the Service Request (SR) connections from ADC to GTM and eGTM. The Micro Channel Sequencer (MCS) inside GTM can service requests from TMADC stand-alone. Timer Input Modules (TIMs) inside GTM and eGTM can receive ADC interrupts as event signals.

Service requests from ADC to MCS

Result register 14 and result register 15 of each TMADC are mapped to the ADI interface of the MCS.

***Note**: Each channel result can be configured to go to RES14 and RES15.*

A pulse to the ADI is generated to signal new results in RES14 and RES15. Please refer to chapter "ADC Data Interface (ADI)" for more details.

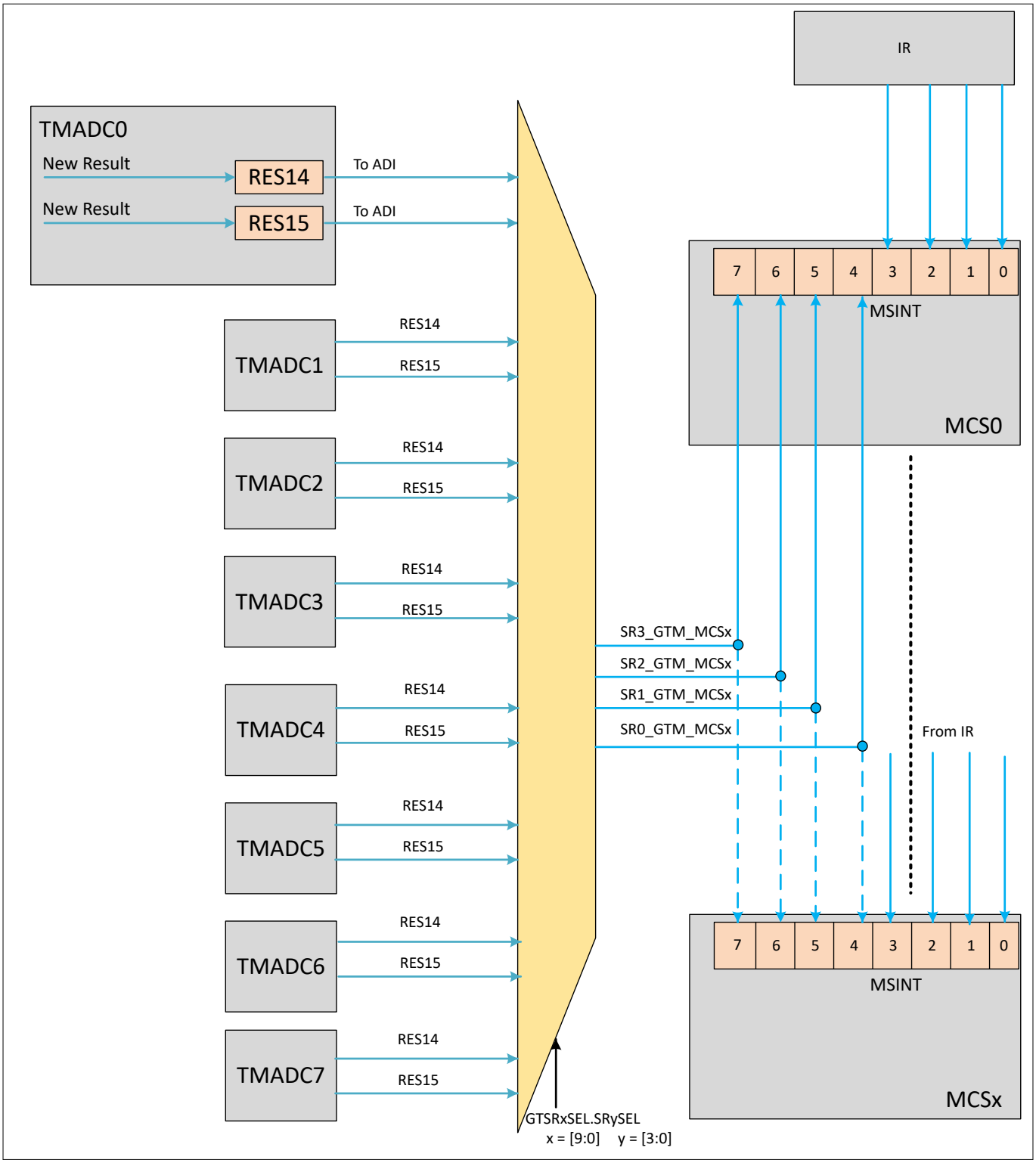


Figure 1191 Service request interface from ADC to GTM (MCS)

This figure shows the basic concept of service requests going from ADC to GTM. Service requests indicating new results in RES14 and RES15 can get routed to GTM (MCS).

Each MCS inside GTM has 8 shared interrupts, connected to MSINT register inside MCS.

- Interrupt lines 0-3 are connected to IR
- Interrupt lines 4-7 of each MCS are connected to ADC, so ADC can directly send an interrupt

For each MCS you can select 4 interrupts from up to 16 TMADC interrupts. Selection is configured through the register GTSRxSEL.SRySEL. Service request to MCS are first enabled through bit-field SRyEN.

It is possible to send the same service request to all MCS. By sending the same service request to all MCS, they all get the information about the data available on ADI and can fetch the data.

***Note**: Data is only available on the ADI interface. Additional information related to valid input has been removed from ADI. Data is qualified by SR. Please refer to the GTM chapter for more details.*

Service requests from ADC to TIM

The Timer input modules of GTM and eGTM can select a subset of ADC interrupts as trigger event. Selection is done in register TIMnINSELB. Details can be found in the timer description.

Related information

ADC data interface (ADI) on page 8350

45.4 Time-multiplexed analog-to-digital converter (TMADC)

TMADC is based on a Successive Approximation Register (SAR) concept and provides 12-bit analog to digital conversion of up-to 16 external analog input channels using 2 SAR cores supporting a maximum output sample-rate of 4 MSPS.

45.4.1 Feature list

- Analog supply voltage 5.0 V or 3.3 V
- Input voltage range from analog reference ground (VAGND) up to analog reference supply voltage (VAREF)
  - In case V_{IN} exceeds VAREF, the conversion result will be 0xFFF
  - In case V_{IN} is lower than VAGND, the conversion result will be 0x000
- Each TMADC module supports up to 16 external analog input channels mapped to 2 SAR cores
- Programmable sampling time on each input channel with 12.5 ns resolution
- Maximum output data rate up to 4 MSPS
- Flexible trigger source selection
  - Conversions triggered by software, timer events, or external events
  - Up to two hardware trigger request sources for each channel out of 64 available sources
  - Synchronous sampling support on multiple input channels with a single trigger source
- One shot or continuous conversion modes supported on single or multiple input channels
- Powerful result handling
  - Independent channel-specific result registers
  - Burst read access support for time-stamp and result registers
- Configurable checking against programmable limits
  - Support boundary value update on hardware trigger
  - Limit check on conversion results for up to two independent input channels
- Service request generation based on result events and error events
- External analog multiplexer control support
- Reference voltage monitoring support
- Built-in diagnostic features
  - Broken wire diagnostics
  - Converter self-diagnostics to verify the correct operation of the SAR cores
  - Monitoring of selected internal voltages (supplies, references)

45.4.2 Functional overview

Each external analog input channel of a TMADC has its dedicated sample and hold (SH) circuitry, therefore allowing independent SH operation on each channel. Each channel can be mapped to one of the two available SAR cores for conversion through the channel core mapping connection matrix.

The channels are triggered for conversion either by a software write, hardware timer signals, automatic triggering, or by external events. There are 2 internal monitor channels provided for the purpose of supervising other on-chip signals like internal core supply voltages, or supply voltages from other TMADC or DSADC.

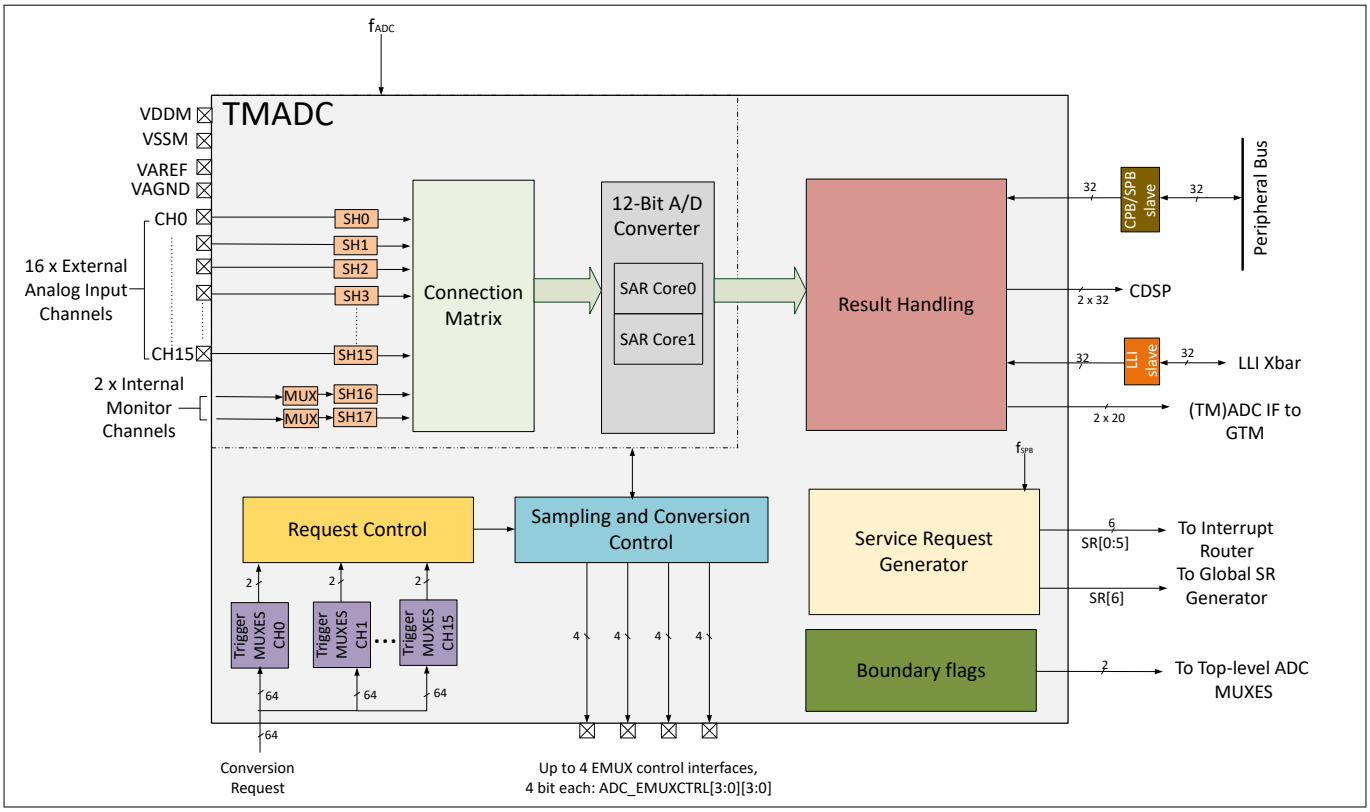


Figure 1192 TMADC block diagram

Request control

An external input channel is configurable to receive a conversion request from different request sources; 2 hardware trigger sources and a software trigger source. The selection of these two hardware trigger sources is made through the trigger multiplexers from an available option of 64 sources (see chapter "Trigger request control"). The request control logic block handles the conversion requests for the channel. A trigger event, if processed, switches the channel to a hold state for conversion. Different trigger mode configurations offer flexibility to the user to initiate a conversion request on a rising edge, falling edge or both rising and falling edges of the hardware trigger. Handling of concurrent trigger requests in the same cycle is also controlled here.

Sampling and conversion control

Channel and core state transitions are handled here (see chapters "Sampling" and "Conversion"). Intrinsic core's round-robin channel arbitration implementation handles simultaneous conversion requests on multiple channels connected to it. For off-chip analog multiplexer control, selection lines are provided for the external multiplexer's channel selection through MCU GPIO pins.

Result handling

This unit handles the configuration of the result registers where the conversion results are stored (see chapter "Conversion result handling"). The conversion result of the external analog channels is stored into one of 16 result registers. Internal monitor channel results are stored in specific monitor channel result registers. The wait-for-read mode configuration for the result registers stalls further subsequent conversion on the respective channel until their corresponding result is read, therefore, avoiding any data loss due to data overwrite.

Boundary conditions checks on the conversion results (if enabled) are configurable. TMADC result data (RES0 and RES1) can be read and further processed by CDSP module for advanced filter operations. A low latency path is available between PPU and TMADC through LLI providing faster result data read access capability. Additionally, each TMADC module provides a 20-bit ADC Data Interface (ADI) for the results (RES14 and RES15) to GTM.

Service request generation

TMADC has 6 service request lines going to the interrupt router and 1 service request line connected to the global service request generator (see chapter "Service request generation"). The generation of these service requests is configurable based on error, result or boundary event(s). These three event types are described below:

- Error event indicates an error condition on a channel that arises on receiving a conversion request trigger during its sampling or hold phases, while conversion is ongoing or while trigger delay counter is running
- Result events indicate the availability of new result data in the result registers
- Boundary events indicate successful configured boundary checks on the channel results

45.4.3 Functional description

The functional description of TMADC is divided into the following sections:

- Module operation states
- Changing module configuration
- Module start-up calibration
- Trigger request control
- Analog input channel configuration
- Conversion result handling
- Service request generation
- Diagnostics
- On-chip signal monitoring and supervision
- External Multiplexer Control

45.4.3.1 TMADC operation states

This section describes the behavior of the module response to enable, disable, run and suspend request.

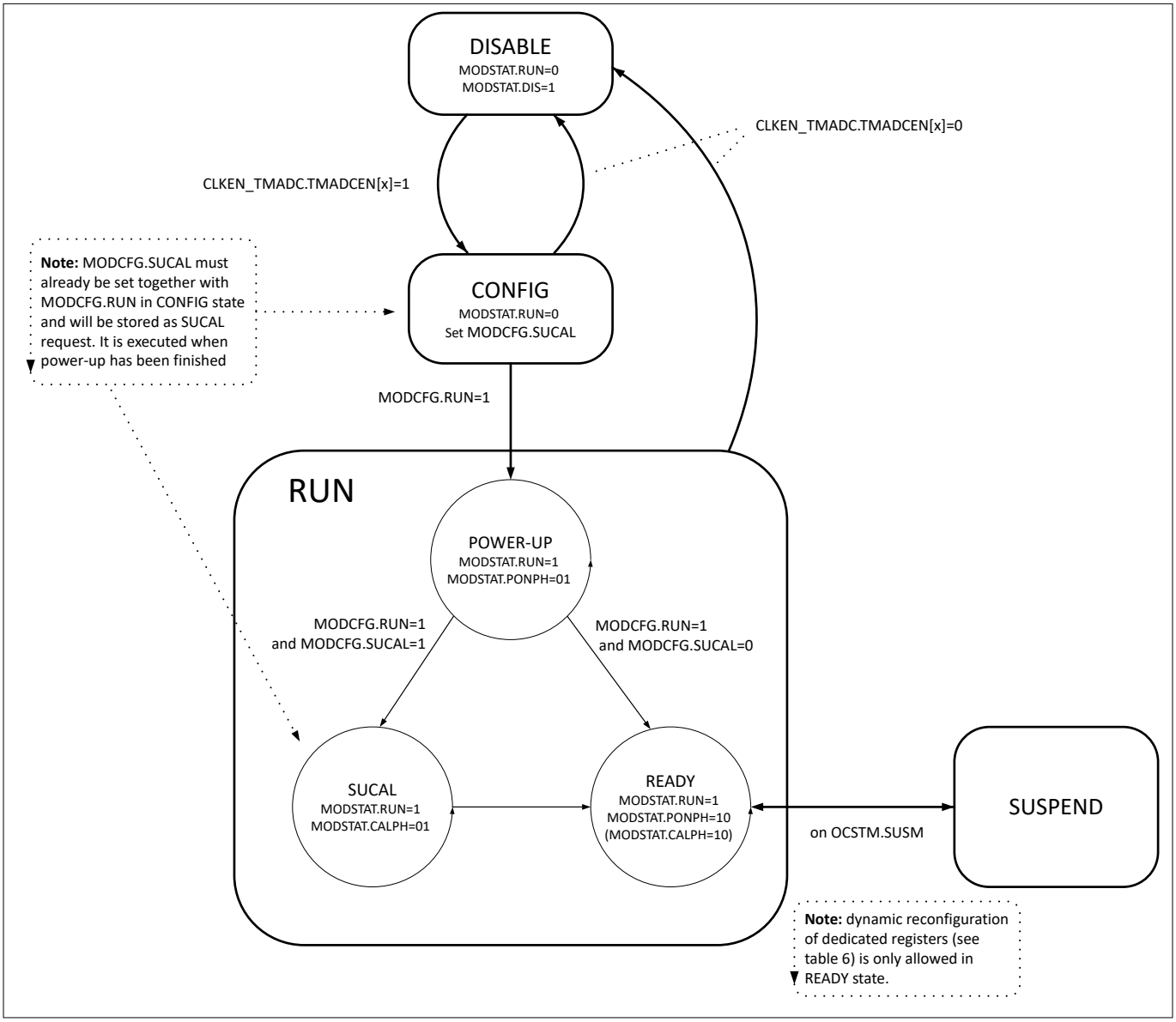


Figure 1193 TMADC module operation states

Disable

TMADC module registers are to be initialized in the CONFIG state upon setting CLKEN_TMADC.TMADCEN[x]=1_B.The TMADC module in CONFIG state is indicated with TMADCx_MODSTAT.RUN=0_B. Before setting the module to RUN state, all required settings for example trigger sources, channel enable, channel configuration registers, result registers, connection matrix setting and operation modes, should be configured in this state.

The TMADC module can be disabled while in the RUN or CONFIG state by setting CLKEN_TMADC.TMADCEN[x]=0_B. On disabling, the module ignores any further trigger requests, finishes the conversion of any currently active channel(s) in hold or conversion phase on the core(s), stores the result for the converted channel(s) and sets corresponding status flag for the result(s). Subsequently the module is powered-down, all internal module states are cleared, and all module clocks are switched-off.

The module disabled state is indicated with TMADCx_MODSTAT.DIS=1_B.

Run

Once the module registers are configured, the converter is started by setting TMADCx_MODCFG.RUN=1_B. When the module is set to RUN state, the module iss powered-on in time t_{WUDIS} (please refer to the respective data sheet for appropriate values). An active module power-on sequence is indicated with MODSTAT.PONPH=1_B and a running module state is indicated with TMADCx_MODSTAT.RUN=1_B.

When transitioning to DISABLE state from RUN state by setting CLKEN_TMADC.TMADCEN[x]= 0_B, the module ignores any further trigger requests, finishes the conversion of any currently active channel(s) in hold or conversion phase on the core(s), stores the result for the converted channel(s) and the sets corresponding status flag for the result(s). Subsequently, the TMADC analog block is powered-down, all internal module states are cleared, and transition is made to the DISABLE state, which is indicated with the clear of TMADCx_MODSTAT.RUN bit. Note that in the CONFIG state, anytime the channel configuration matrix is modified, you must enable the start-up calibration process on setting the module to RUN state.

Suspend

The module can be suspended in a hard-suspend or soft-suspend manner. On receiving a hard-suspend request OCSTM.SUSM=01_B, the module clock is disabled immediately and the module moves to suspend state. On setting OCSTM.SUSM=11_B, all channels currently in their hold state are converted as per the round-robin arbitration and any incoming conversion trigger request on the channels is ignored. The module is suspended after the result of the last converted channel is stored and the status flag set.

***Note**: The channel(s) in hold state start loosing the captured charge on the sampling capacitor(s) as the channel hold time extends above 16 periods of t_{CONV}. This can happen when the module gets suspended with OCSTM.SUSM=01_B request causing channel(s) being suspended in hold or conversion or hold state, respectively. Therefore, upon suspend removal, the conversion results for those channel(s) shall be considered invalid until the channel(s) have been re-sampled for at-least an accumulated duration of t_{mstc} + 1.6 µs.*

The following table shows the state transitions. Transitions marked as 'User scenario' result from intended use cases.

Table 2375 State transitions

Current State	Next: RUN	Next: CONFIG	Next: DISABLE	Next: SUCAL	NEXT:SUSPEND
RUN	N/A	N/A	User scenario	No user scenario	User scenario
CONFIG	User scenario	N/A	User scenario	User scenario	No user scenario
DISABLE	Unreachable	User scenario	N/A	No user scenario	Unreachable
SUCAL	User scenario	No user scenario	No user scenario	N/A	No user scenario
SUSPEND	User scenario	No user scenario	No user scenario	No user scenario	N/A

45.4.3.2 Changing module configuration

The configuration bit-fields control the operation of the functional blocks of the TMADC. Configure the TMADC while in CONFIG state.

While in READY state the following register bits can be modified:

Table 2376 TMADC dynamic configuration and implication

Dynamic configuration	Implication
Module enable (TMADCx_MODCFG.RUN)	Start or stop module. For start and stop procedure (see chapter Module operation states)

(table continues...)

Table 2376 (continued) TMADC dynamic configuration and implication

Dynamic configuration	Implication
Output supervision signal select (TMADCx_MODCFG.OSSEL)	Multiple TMADC output supervision signals (see device specific chapter) are connected on the same line. The user must ensure at any given time only one signal is active on the line
Software trigger (TMADCx_SWTRCFG.SCHSEL)	This is a channel conversion request from software and it can get triggered at any time
Monitor channel multiplexer setting (TMADCx_MCHy_CFG.SEL)	The monitor channel multiplexer setting selects the internal signal to be sampled and converted. The setting can be reconfigured in READY state
Channel broken-wire diagnostics current source enable/disable (TMADCx_BWDCFG)	BWD can be enabled or disabled on any input channel in READY state
Channel-core self diagnostics settings (TMADCx_BWDCFG)	CSD settings can be changed in READY state
TMADCx_SRi_CFG.EVEN	As described in scenarios below
TMADCx_SRi_CFG.EVSEL	As described in scenarios below
TMADCx_SRi_CFG.EVOP	As described in scenarios below
TMADCx_CHy_CFG.HWTRxM	As described in scenarios below
TMADCx_CHy_CFG.HWTRxMS	As described in scenarios below
TMADCx_CHy_CFG.ATREN	As described in scenarios below
TMADCx_RSLTu_CFG.WFR	As described in scenarios below
TMADCx_CHy_STC.TRD	As described in scenarios below

Following use cases are supported with dynamic reconfiguration:

TMADCx_CHy_CFG.HWTRxM (trigger mode) and TMADCx_CHy_CFG.HWTRxMS (tigger source) channel configuration register bitfields are changed on the fly

- Switch off the HWTRx and delay counter enable (HWTRDLYEN) in configuration register TMADCx_CHy_CFG
- Reconfigure the HWTRx source, change trigger delay (TRD) if necessary
- Re-enable the HWTRx and delay counter enable (i.e. change the HWTRx mode for the channel and set HWTRDLYEN)

  In general the trigger delay counter (TRD) can get changed any time. In case the counter is enabled and running, the new value is taken after the actual delay count expires. If counter is idle, new value is taken immediately.

Change channel configuration between ATREN (TMADCx_CHy_CFG. ATREN) and HWTR mode

- Disable the ATREN (HwTRx are already disabled)
- Configure the HWTRx source
- Configure the HWTRx mode (i.e. enable it)
- Run in HWTRx mode
- Disable HWTRx
- Re-enable the ATREN
- Run in automatic trigger mode

Scenarios for service request configuration register using TMADCx_SRi_CFG.EVEN, EVSEL, EVOP. Reconfiguration of TMADCx_SRi_CFG.EVEN is only supported in AND mode (EVOP=10b)

Change event and its operation mode.

- Disable the EVSEL = 00b (Disable)
- Reconfigure the EVEN, EVOP field
- Re-enable EVSEL = 11b/10b(Boundary/ Result)

For continuous triggered groups, the application SW may want to keep TMADCx_RSLTu_CFG.WFR= true to reduce interrupt load

- Disable trigger sources for the channel
- Update WFR for the result register associated to disabled channel
- Re-enable trigger source for the channel

Related information

TMADC operation states on page 8329

45.4.3.3 Module start-up calibration

Calibration reduces deviation errors caused by manufacturing process variations. To ensure high accuracy of generated conversion results throughout the operation time, it is required to run the start-up calibration sequence before transition to RUN state. Start-up calibration can only be triggered when t_{WUDIS} has passed.

Exception: if the channel-core mapping settings have not been altered since the previous RUN state, the calibration procedure is not required to be executed. The start-up calibration is initiated by writing 1 to bit SUCAL in register TMADCx_MODCFG.SUCAL.

It is required to run the calibration before entering the RUN state, so that after a duration of t_{WUDIS}+ t_{CAL}, the converter starts its operation with the set configurations.

Initiating calibration from RUN state requires transition to DISABLE and CONFIG state. RUN request with SUCAL bit set in CONFIG state will activate the calibration (SUCAL state)

The respective TMADC modules must be enabled when running a start-up calibration. Conversions are started, only after the calibration has completed (completed calibration sequence is indicated by bit TMADCx_MODSTAT.CALPH = 10_B). The duration of the start-up calibration phase is up to t_{CAL} (see datasheet). A currently active calibration sequence is indicated with TMADCx_MODSTAT.CALPH= 01_B.

45.4.3.4 Trigger request control

This chapter describes the trigger definition with respect to TMADC sampling behavior and handling of hardware and software trigger events.

45.4.3.4.1 Trigger definition

In TMADC a channel's sampling time (t_{st}) has 2 phases, the minimum sampling time (t_{mstc}) and the extended sampling time (t_{ext}), as shown in the next figure.

The extended sampling time is the time duration after the end of minimum sampling time and the end-of-sampling caused by a trigger event. The trigger event switches the channel to hold state for conversion. The minimum value of the hardware trigger PWM period should be configured greater than the sum of the total channel conversion time and the total sampling time. In TMADC, the total channel conversion time is a function of the channel hold time t_{hold} and channel conversion time t_{conv}, where t_{hold} depends on the numbers of channels connected to a SAR core (see chapter "Conversion" for details). A hardware trigger is processed only if it occurs after the minimum defined sampling time window is over. A software trigger can occur during minimum sampling time.

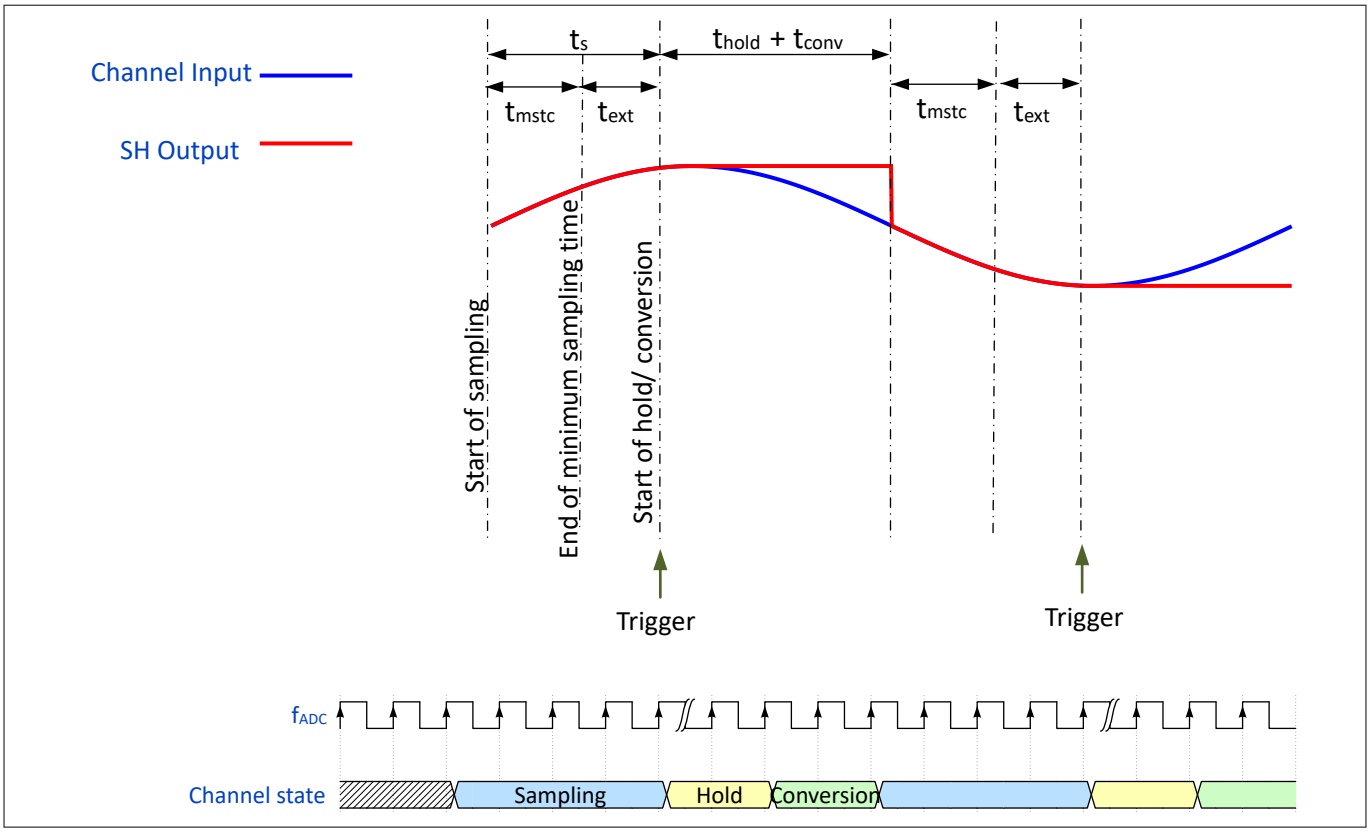


Figure 1194 TMADC trigger concept

Trigger events

The input trigger interface of TMADC is shown in the figure below. The 64 request sources are shared among all TMADC modules .

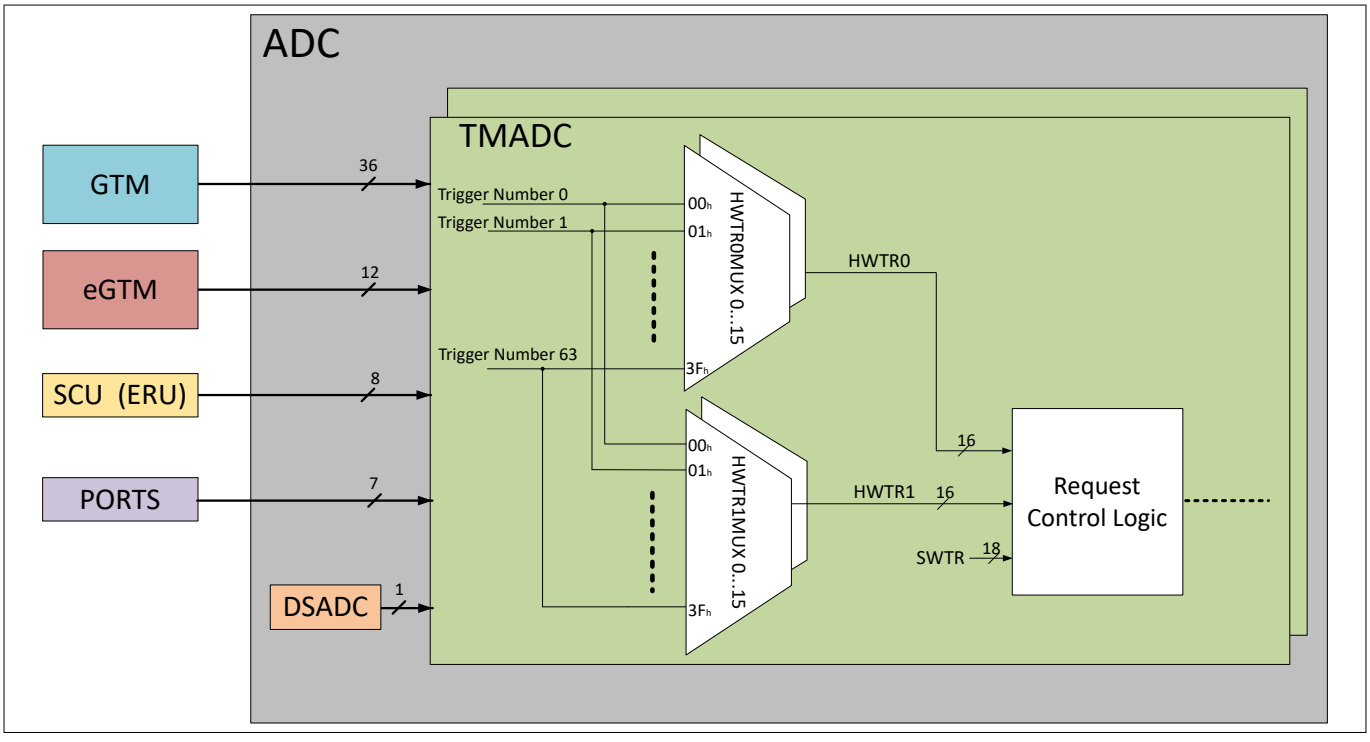


Figure 1195 TMADC trigger interface

There are 2 type of request sources for a channel as described below:

- Hardware triggers request source: For each channel 2 hardware trigger request sources can be selected from 64 available possibilities. These are configured through bit-fields HWTR0MS and HWTR1MS in the register TMADCx_CHy_CFG
- Software trigger request source: Application software can directly trigger a channel or multiple channels of a TMADC for conversion through the setting of bit-field SCHSEL in the register TMADCx_SWTRCFG. A software trigger generates a single conversion request for the channel every time the respective channel bit is set in the register TMADCx_SWTRCFG. For example, by setting TMADCx_SWTRCFG.SCHSEL=3_H software triggered conversion request is generated for channels CH0 and CH1

Related information

Conversion on page 8338

45.4.3.4.2 Trigger handling and arbitration logic

The request control logic for each channel comprises of a trigger edge detector logic, programmable trigger delay counter and a trigger source arbiter for concurrent trigger requests management from multiple request sources.

The figure below shows the hardware and software trigger propagation through request control logic block in a simplified block diagram.

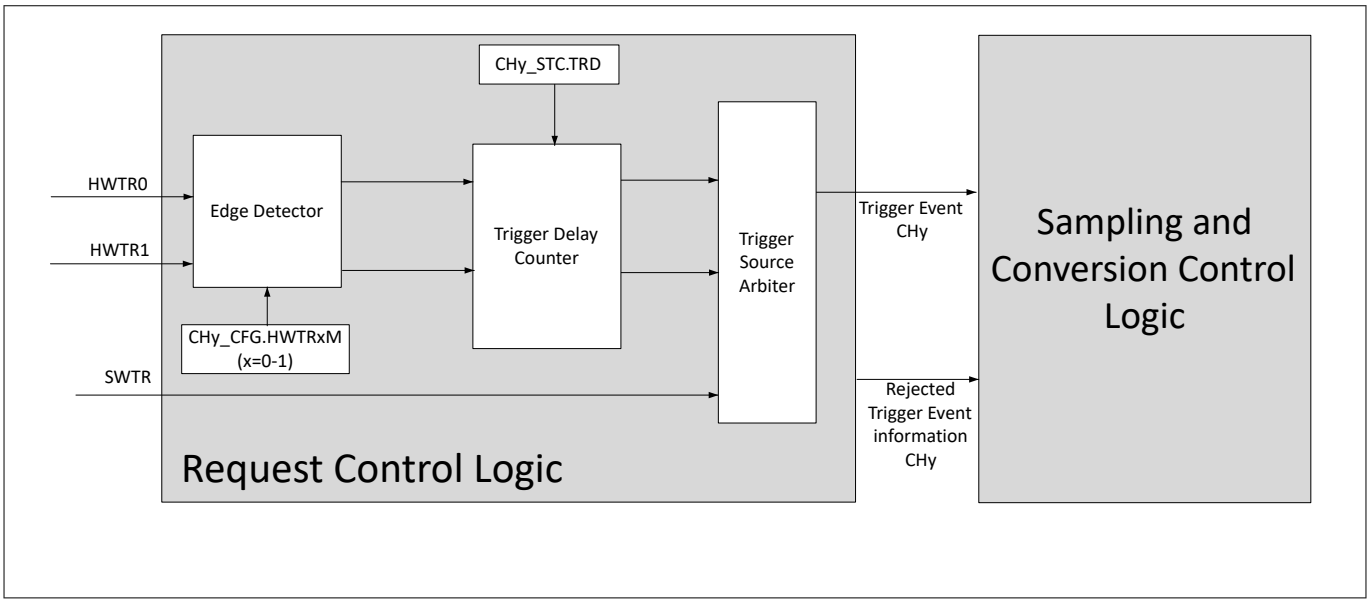


Figure 1196 Request control logic

Edge detection

A hardware trigger can be configured to initiate a channel conversion request on either its rising or falling edge or on both the edges. The edge selection for the hardware trigger used by a channel is done through bit-fields HWTR0M and HWTR1M in the register TMADCx_CHy_CFG for sources HWTR0 and HWTR1 respectively. A hardware trigger can be disabled for the channel by setting TMADCx_CHy_CFG.HWTRiM = 00_B (i=0-1). An example is shown in figure below with different trigger mode configurations of HWTR0 for channel CH0.

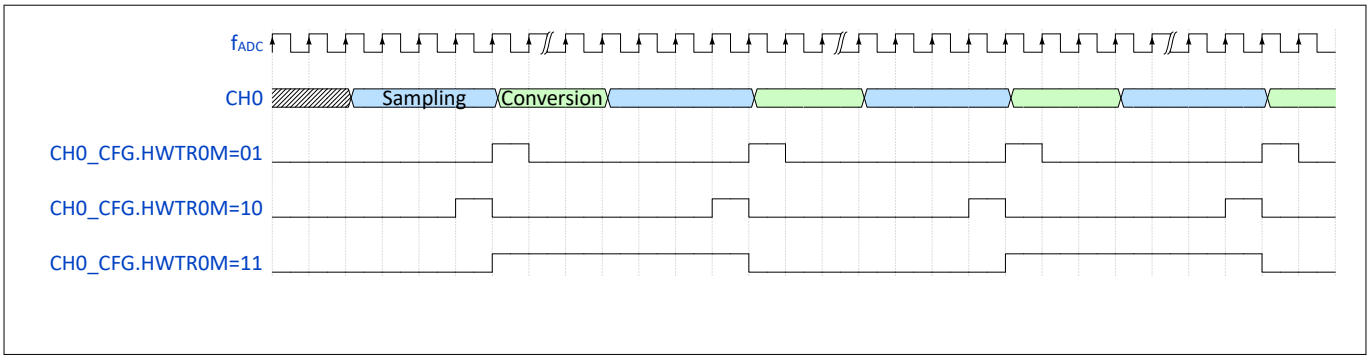


Figure 1197 TMADC trigger configuration

Trigger delay

The trigger delay counter adds a programmable delay to the input hardware trigger. It is configured through the bit-field TRD in the TMADCx_CHy_STC register. It is enabled and disabled with the bit-field HWTRDLYEN in the register TMADCx_CHy_CFG. The delay counter operates with a fixed clock frequency of 80 MHz. The trigger delay counter operates when there is a non-zero value of TRD bit-field set in the TMADCx_CHy_STC register. The counter is common to both of the hardware trigger sources so the trigger which arrives earlier sets the counter in operation and all further trigger events arriving at the counter input are ignored and indicated with error flag bit HWTRE along with the hardware trigger number information in bit-field TRENR in the TMADCx_CHy_STAT register until the counter resets back to the value specified by TMADCx_CHy_STC.TRD. When the hardware triggers HWTR0 and HWTR1 arrive in the same clock cycle at the counter input, then HWTR1 is ignored and error is flagged in the TMADCx_CHy_STAT register. Setting TMADCx_CHy_STC.TRD=00_H disables the counter and the hardware triggers HWTR0 and HWTR1 are propagated to the trigger source arbiter. For cases when the channel conversion is requested on a rising and falling edge of a trigger signal, the user must configure the trigger delay smaller than duty cycle period of the trigger. The table below displays the hardware trigger delay settings and corresponding values.

Table 2377 Hardware trigger delay configuration

Trigger delay configuration CHy_STC.TRD	Absolute trigger delay
00_H	0 ns
01_H	12.5 ns
02_H	25 ns
03_H	37.5 ns
04_H	50 ns
05_H	62.5 ns
06_H	75 ns
07_H	87.5 ns
08_H	100 ns
…	…
$FFFF_H$	819.1875 µs

***Note**: Receiving any new trigger event on either of the selected hardware trigger while the trigger delay counter is running will result in rejection of the new requests. And if the new trigger events arrive in the same cycle then only HWTR0 information will be visible to the user in the TMADCx_CHy_STAT register.*

Trigger source arbiter

The trigger source arbiter handles the incoming trigger requests. The arbiter can receive input trigger requests from 3 sources; 2 hardware trigger (HWTRx) events and a software trigger (SWTR) event as shown in Figure 1196. A software trigger event is asynchronous to the hardware trigger events. The following points describe the trigger request handling:

- Hardware triggers are processed if they occur in the extended sampling time. The extended sampling time starts once the minimum sampling time is over. Triggers are ignored during a running minimum sampling time phase or if a channel is in a hold or conversion state. Ignored hardware triggers set the hardware trigger error (HWTRE) along with the trigger error number (TRENR) in the TMADCx_CHy_STAT register
- A software trigger event is processed if the request occurs either during a running minimum sampling time phase or during extended sampling time phase. Occurrence of a software trigger during minimum sampling time phase results in channel transition to the hold state after the minimum sampling-time counter is expired. A software trigger will be ignored if it occurs in the channel's hold state. An ignored software trigger information is indicated to the user through the Software Trigger Error bit (SWTRE) in the TMADCx_CHy_STAT register

During extended sampling state of a channel, the next trigger event will get processed and others are ignored. Ignored hardware or software triggers are discarded and will not be repeated any more. The table below shows/describes the trigger source arbitration.

Table 2378 Trigger source arbitration

Channel state	Concurrent triggers	Processed trigger	Ignored trigger(s)
Minimum sampling-time phase	SWTR, HWTRx	SWTR	HWTRx
	HWTR0, HWTR1	None	HWTR0, HWTR1
Extended sampling-time phase	SWTR, HWTRx	HWTRx [1)]	SWTR
	HWTR0, HWTR1	HWTR0	HWTR1
Hold or conversion phase	SWTR, HWTRx	None	SWTR, HWTRx
	HWTR0, HWTR1	None	HWTR0, HWTR1

1) Here if both HWTR0 and HWTR1 occur simultaneously, then HWTR0 is processed and HWTR1 is ignored.

***Note**: In cases where both hardware triggers HWTRx are ignored, only the most recently ignored hardware trigger information will be available in the CHy_STAT register and if both triggers are ignored at the same time then only HWTR0 information will be visible to the user.*

Following table describes the detailed trigger and error behavior after a running trigger delay counter gets disabled by Software.

Table 2379 Trigger arbitration after Software disable of trigger delay counter

Scenario	Trigger behavior
Channel in minimum sampling/hold/conversion time window (in example: minimum sampling time not reached)	If delay counter triggered by HWTR1 expires at the same clock cycle as trigger pulse coming from HWTR0 then both hardware triggers are ignored and the ch_stat of the respective channel will be updated with the error information of HWTR1.
Channel in extended sampling time window	If delay counter triggered by HWTR1 expires at the same clock cycle as trigger pulse coming from HWTR0 then HWTR1 will be used for triggering a conversion. HWTR0 pulse will not be processed and no error flag will be set in the ch_stat register.

(table continues...)

Table 2379 (continued) Trigger arbitration after Software disable of trigger delay counter

Scenario	Trigger behavior
	If delay counter triggered by HWTR0 expires at the same clock cycle as trigger pulse coming from HWTR1 then HWTR0 will be used for triggering a conversion. HWTR1 pulse will not be processed and no error flag will be set in the ch_stat register.

The figure below shows the timing windows during channel sampling where the triggers are accepted or ignored. No triggers are accepted in channel's hold or conversion phase.

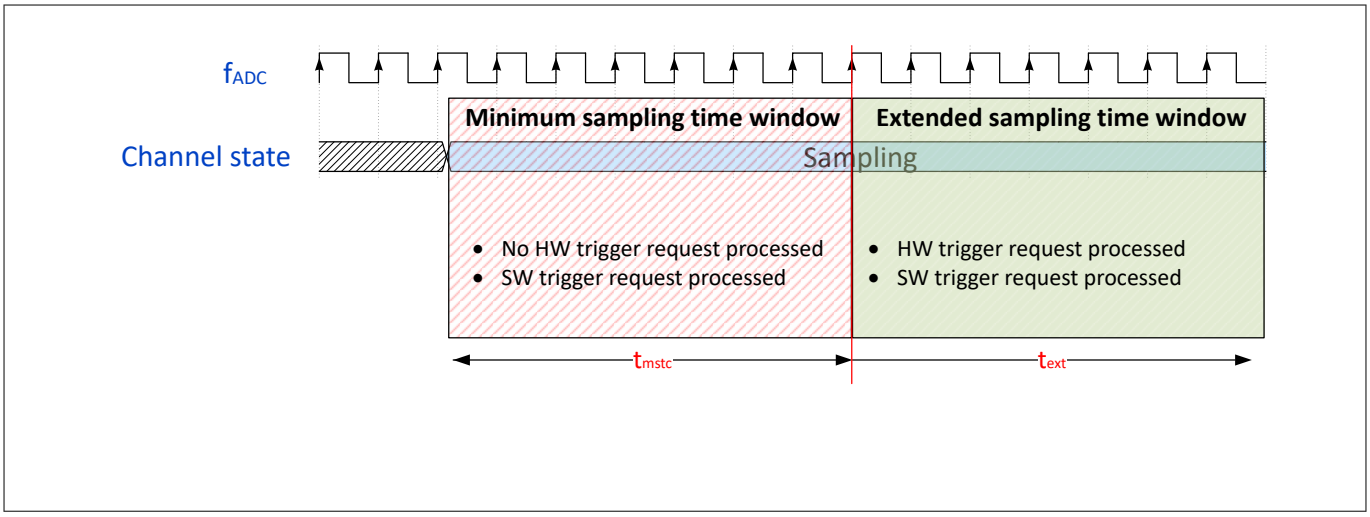


Figure 1198 TMADC trigger handling

45.4.3.5 Analog input channel configuration

This chapter describes the sampling and conversion configurations for the input channel.

45.4.3.5.1 Sampling

Channel sampling configuration

The sample and hold (SH) unit tracks the analog input signal on the channel during the sampling time. A channel is enabled by setting TMADCx_CHy_CFG.EN=1_B. The minimum sampling time (ST) is configurable by the user through bit-field ST in the TMADCx_CHy_STC register. Please check product-specific datasheet for minimum sampling time values with respect to different types of conversion.

The SH has to get enabled in CONFIG state. Transition to RUN state initiates the input sampling by loading the sampling counter with ST value. The counter operates with a clock frequency of $f_{ADC}/2$. The configurable minimum sampling time values for a channel are given in the table below. There are 2 SAR cores (Core-0 and Core-1) available for performing channel conversions. The mapping of the channel to the core is done through bit-field CHCSEL in the TMADCx_CMATCFG register.

***Note**: For low or medium range supply voltage, the sampling time shall be configured with resolution of 25 ns with minimum allowable setting of TMADCx_CHy_STC.ST = 07_H. Please check product-specific datasheet for minimum sampling time values with respect to supply voltage.*

Table 2380 Sample time coding

Sampling time configuration TMADCx_CHy_STC.ST	Minimum sampling time t_{mstc}
00_H (Reserved)	12.5 ns
01_H (Reserved)	25 ns
02_H (Reserved)	37.5 ns
03_H	50 ns
04_H	62.5 ns
05_H	75 ns
06_H	87.5 ns
07_H	100 ns
...	...
…	...
$FFFF_H$	819.2 µs

45.4.3.5.2 Conversion

TMADC supports conversion of a single channel or a group of channels in one-shot (trigger based) or continuous (auto-scan) manner. The type of TMADC channel conversions supported with some example cases are further described in this chapter.

- Single channel one-shot conversion
  - Each conversion of a channel is initiated either on a hardware trigger (HWTRx) or a software trigger. An interrupt can be configured for indicating the availability of every new channel conversion result
- Channel group one-shot conversion
  - A channel group consists of multiple TMADC channels. Each conversion of a channel group can be initiated either on the common hardware trigger or through a software trigger register. An interrupt can be configured for indicating availability of new channel group results, that is when all the results of the group are available
- Single channel continuous conversion
  - A channel can be configured in continuous conversion operation. In this mode, no external request sources are needed for initiating a channel conversion. User can stall any further channel conversion by enabling wait-for-read functionality to avoid overwriting previous data
- Channel group continuous conversion
  - Continuous conversion can be configured to a group of channels. An interrupt can be configured for indicating availability of new channel group results that is when all the results are available in the result registers pertaining to each channel in the group

A channel is added to the connected core's arbitration queue for conversion on a trigger event for one-shot conversion, or after the minimum sampling time is completed, for continuous conversion. Each core handles the conversion requests on multiple channels in parallel by following an intrinsic round-robin arbitration over the connected channels on that particular core. The round-robin algorithm implies the selection of the next higher indexed channel number in queue from the previous converted channel number. The arbitration is done at the end of the ongoing conversion cycle. There is no priority assignment to a channel in the queue. With N channels connected to a core, in the worst case a channel's conversion time can take is up to $N \times t_{conv}$.

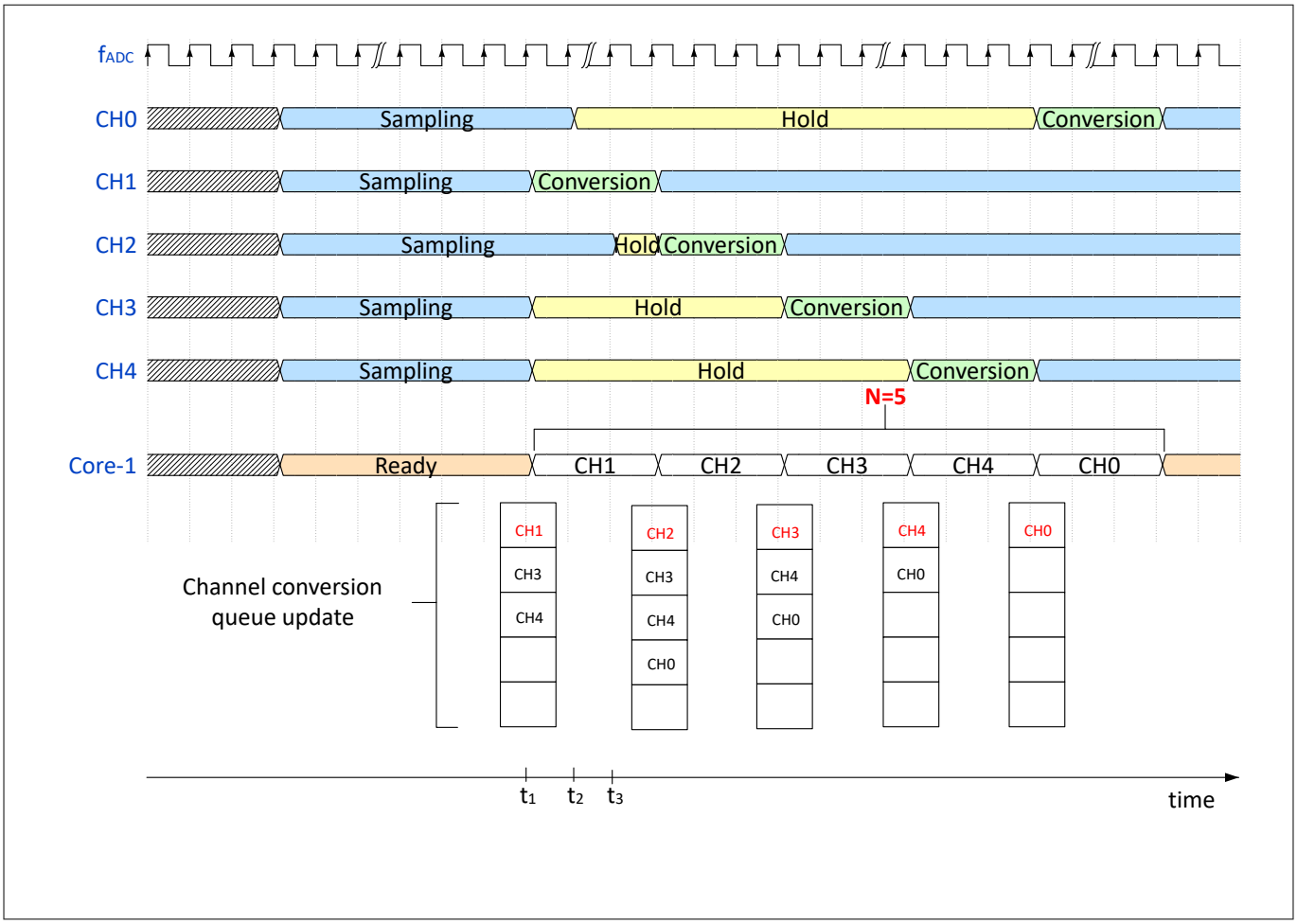


Figure 1199 Channel conversion arbitration example 1

In the figure above 5 channels, CH0 through to CH4, are connected to Core-1. Core-1 is in a ready state waiting to start a channel conversion when available. At time t = t_1, channels CH1, CH3 and CH4 are available for conversion. The queue is updated and Core-1 proceeds with the conversion of CH1 (lowest channel index holds priority) followed by the next higher index channel CH3 and then CH4. Before the end of CH1 conversion, CH0 and CH2 are available at t = t_2 and t = t_3, respectively. The next higher index after CH1 is CH2. CH2 becomes the top of the queue whereas CH0 holds the least priority and is placed at the bottom. From there, channels in the queue get converted as per the sequence.

In the example of the figure above the conversion of the 4 channels delays the conversion of CH0 (by approximately $4 \times t_{conv}$) from the time it finished sampling.

***Note**: With assignment of multiple inputs to a single SAR core, the hold time for pending samples TMADC t_{HOLD} can get violated. Channels need to be distributed across Core-0 and Core-1 in a way the TMADC t_{HOLD} constraint is satisfied.*

Another example is shown in the figure below considering the same channels and core. Here after CH2 is converted, the core goes to ready state since no further channel is ready for conversion since CH1, CH3 and CH4 are still sampling. When CH1, CH3 and CH4 are available for conversion, CH3 gets selected on the basis of being the next higher indexed channel from the last converted channel (CH2), followed by CH4 and then CH1.

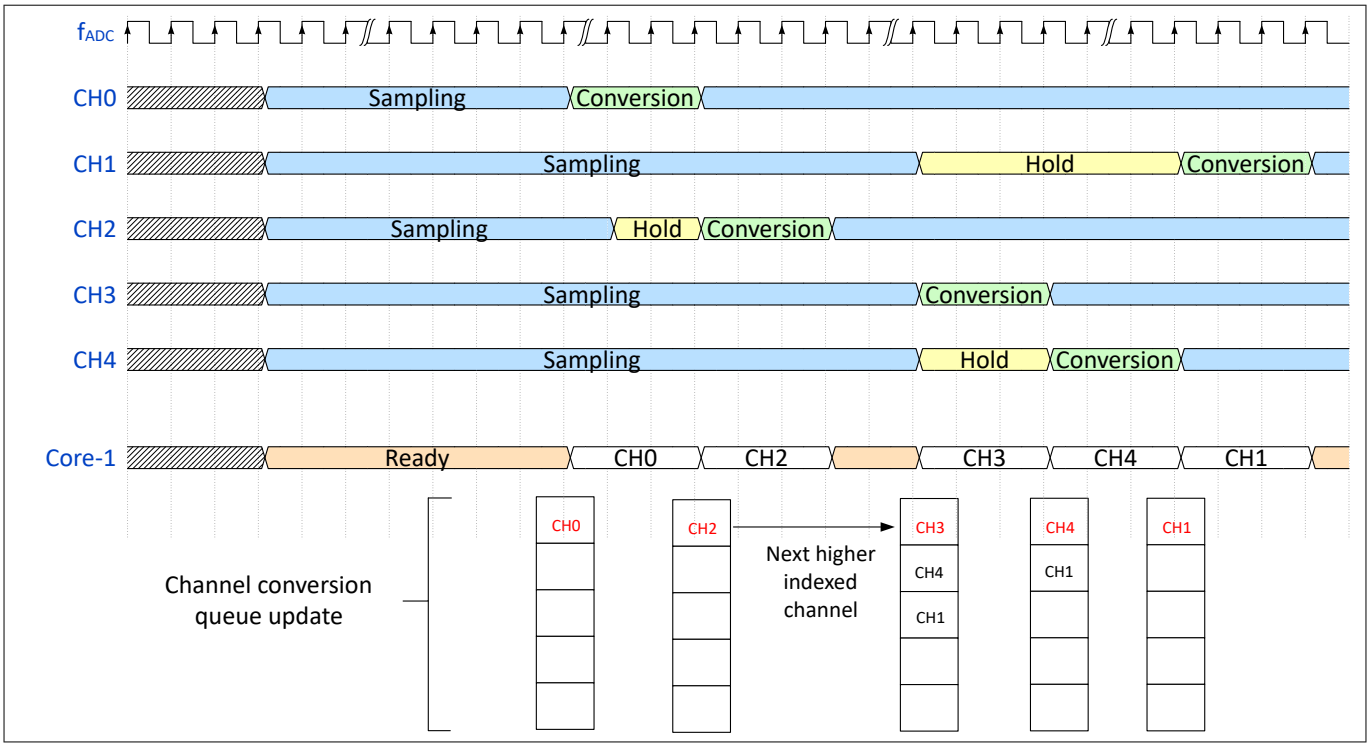


Figure 1200 Channel conversion arbitration example 2

Single channel one-shot conversion

One-shot conversion means that a channel conversion takes place on a hardware trigger (HWTR0 or HWTR1) or software trigger.

Each time the conversion takes place the result is stored in the channel-specific result registers TMADCx_AW0_RESv.

The figure below illustrates the one-shot channel conversion example for channel CH1 connected to Core-0 and channel is configured to initiate conversion on request from either of the request sources HWTR0 and HWTR1. The user should configure the hardware trigger offset time and trigger period with respect to the channel's minimum sampling time and hold time (if multiple channels connected on the core) so that the triggers are processed each time on arrival. The trigger offset duration should take into account the module power-up time, start-up calibration time and the channel's minimum sampling time as shown in the figure below. Here because HWTR1 is occurring after HWTR0, the $t_{offset1}$ needs to take into account the additional delay caused due to the conversion triggered by HWTR0. Therefore, for the illustrated case:

$$t_{\text{offset0}} \geq t_{\text{mstc}}(\text{CHi}) + t_{\text{WUDIS}} + t_{\text{CAL}}$$

$$t_{\text{offset1}} \geq 2 \times t_{\text{mstc}}(\text{CHi}) + M \times t_{\text{conv}} + t_{\text{WUDIS}} + t_{\text{CAL}}$$

In this instance the value of i is 1, and the value of M (total number of channels connected to a core) is 1 because it is considered that no other channels are connected to the same core. The table below shows the minimum hardware trigger period requirement with respect to single channel conversions on a core using single or both hardware request sources.

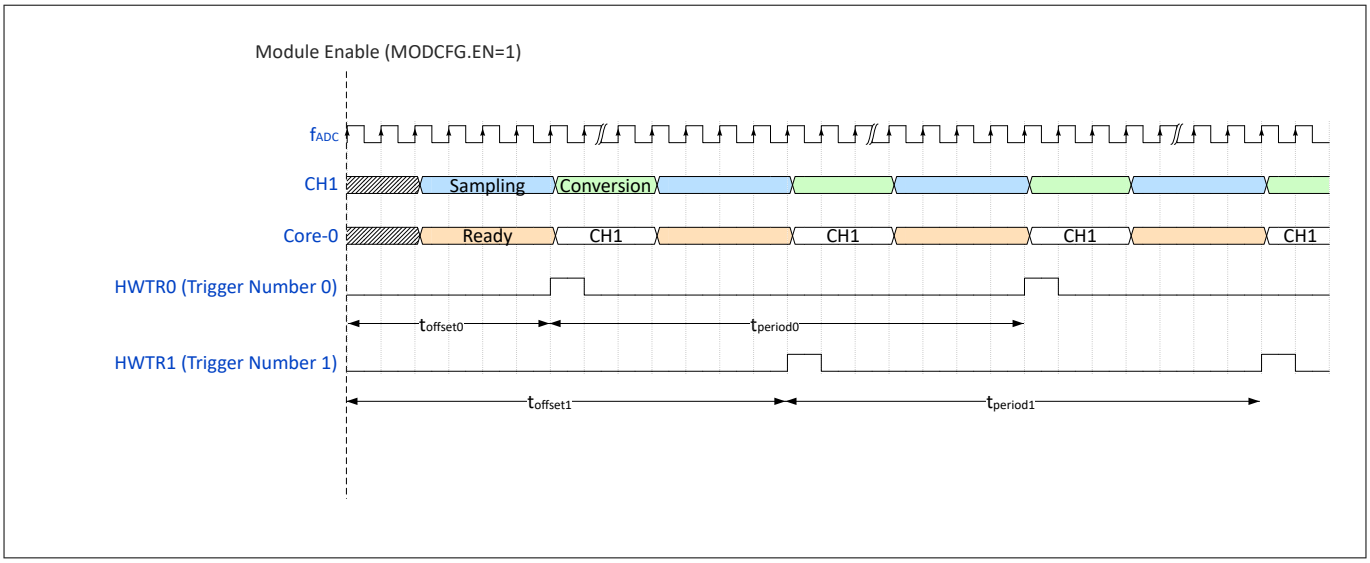


Figure 1201 Single channel one-shot conversion

Table 2382 Minimum hardware trigger period (t_{period}) requirement for a single TMADC channel (CHi) conversion

Channels connected to the core	When only one hardware request source is used	When both hardware request sources are used
1 (only CHi connected to the core)	$t_{period} \geq t_{mstc}(CHi) + t_{conv}$	$t_{period0}\ (or\ \ t_{period1}) \geq 2 \times (t_{mstc}(CHi) + t_{conv})$
M channels (where m>1 number of channels connected to the core)	$t_{period} \geq t_{mstc}(CHi) + M \times t_{conv}$	$t_{period0}\ (or\ \ t_{period1}) \geq 2 \times (t_{mstc}(CHi) + M \times t_{conv})$

Channel group one-shot conversion

A synchronous end of sampling on a channel group can be obtained by configuring the channel group on the same trigger source.

The trigger period must be configured sufficiently long enough to account for:

- The maximum sampling time value of the channel among the channel group
- The conversion time for the channel group connected to the group
- The conversion time of other channels connected to the same core

The figure below shows an example of multiple channel group one-shot conversions where the channel group consisting of 5 channels, CH1 through to CH5, is connected to Core-0 and share the same trigger that is trigger number 4 (TMADCx_CHy_CFG.HWTRxMS=04_H). The trigger event ends the sampling phase of all channels in the group and core starts the conversion. The user can enable and configure the service request on event when all the conversion results of the group are available in the result registers TMADCx_AW0_RESv.

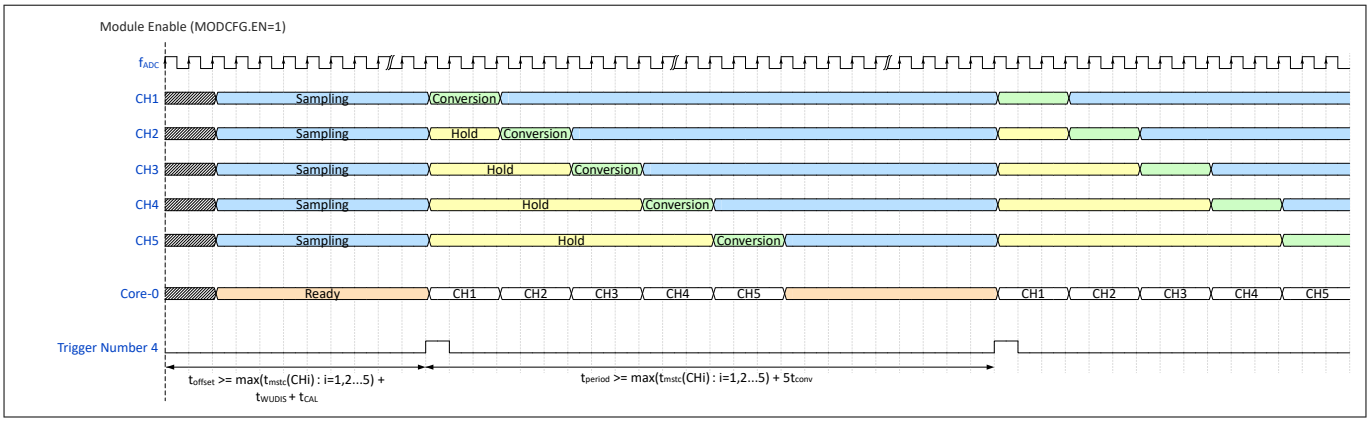


Figure 1202 Multiple channel group one-shot conversion

Table 2383 Minimum hardware trigger period (t_{period}) requirement for TMADC channel group conversion

Channel(s) connected on the core	When only one hardware request source is used	When both hardware request sources are used
One channel group of N channels	$t_{period} \geq \max(t_{mstc}(CHi)) + N \times t_{conv}$ Here CHi represents a channel of the channel group	$t_{period0}$ (or $t_{period1}$) $\geq 2 \times (\max(t_{mstc}(CHi)) + N \times t_{conv})$ Here CHi represents a channel of the channel group
One channel group of N channels + M additional channels	$t_{period} \geq \max(t_{mstc}(CHi)) + (N+M) \times t_{conv}$ Here CHi represents a channel of the channel group	$t_{period0}$ (or $t_{period1}$) $\geq 2 \times (\max(t_{mstc}(CHi)) + (N+M) \times t_{conv})$ Here CHi represents a channel of the channel group

Single channel continuous conversion

A channel is set for continuous conversion mode by setting TMADCx_CHy_CFG.ATREN=1_B. In this mode, the channel switches directly to the hold or conversion state once the configured minimum sampling time has elapsed.

Result data can be overwritten by newer conversion results if not read frequently enough. Use the sampling time to adjust the frequency of conversion results. Use wait-for-read (WFR) to avoid overwriting conversion results before reading it. Set TMADCx_RSLTu_CFG.WFR to 1 to start a conversion once the result register has been read. The sampling time of the channel is extended until the result register is read.

The figure below shows an example of continuous conversion on channel CH0 connected to Core-0, without and with using wait-for-read mode. Here the result register RES0 is configured to store the conversion result of CH0.

***Note**: Using wait-for-read mode for hardware or software trigger based channel conversion causes masking of all triggers on a channel until its result data is read by the software.*

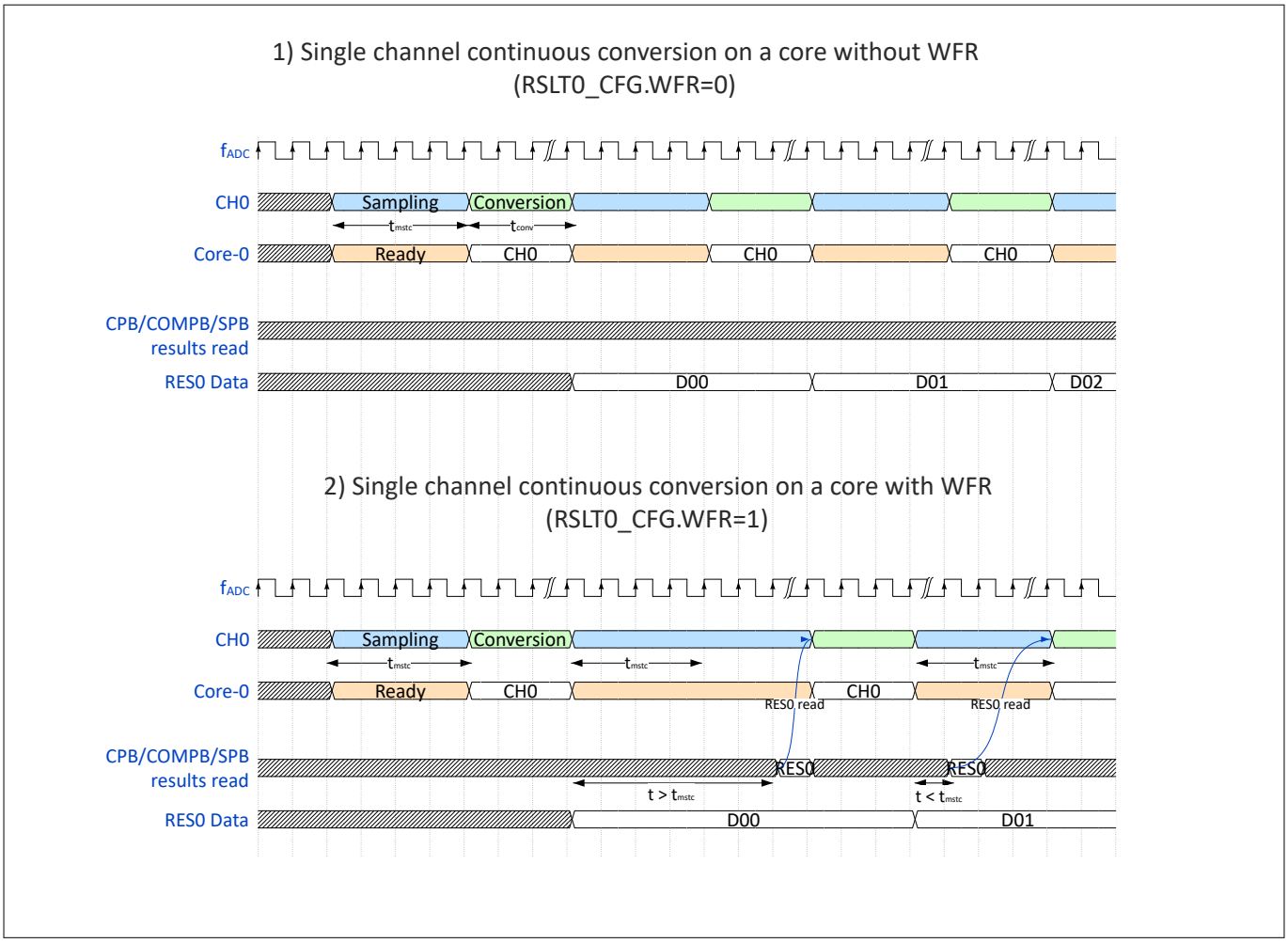


Figure 1203 Single channel continuous conversion

Channel group continuous conversion

A group of channels can be operated in continuous conversion by setting TMADCx_CHy_CFG.ATREN=1_B in the respective channel configuration register.

A service request can be configured for any new result data within a channel group or it can be configured for a single event when new result data is available for all channels in the group (see the Service request generation chapter).

The figure below shows an example of a channel group in continuous conversion mode. Here, the channel group consists of channels CH1, CH2, CH3 and CH4 connected to Core-0 and the channel conversion results are stored in result registers RES1, RES2, RES3 and RES4, respectively. The result registers are configured with wait-for-read mode enabled. A result read ends the corresponding channel's sampling phase (in this case it is shown with a burst read request of 4 result registers).

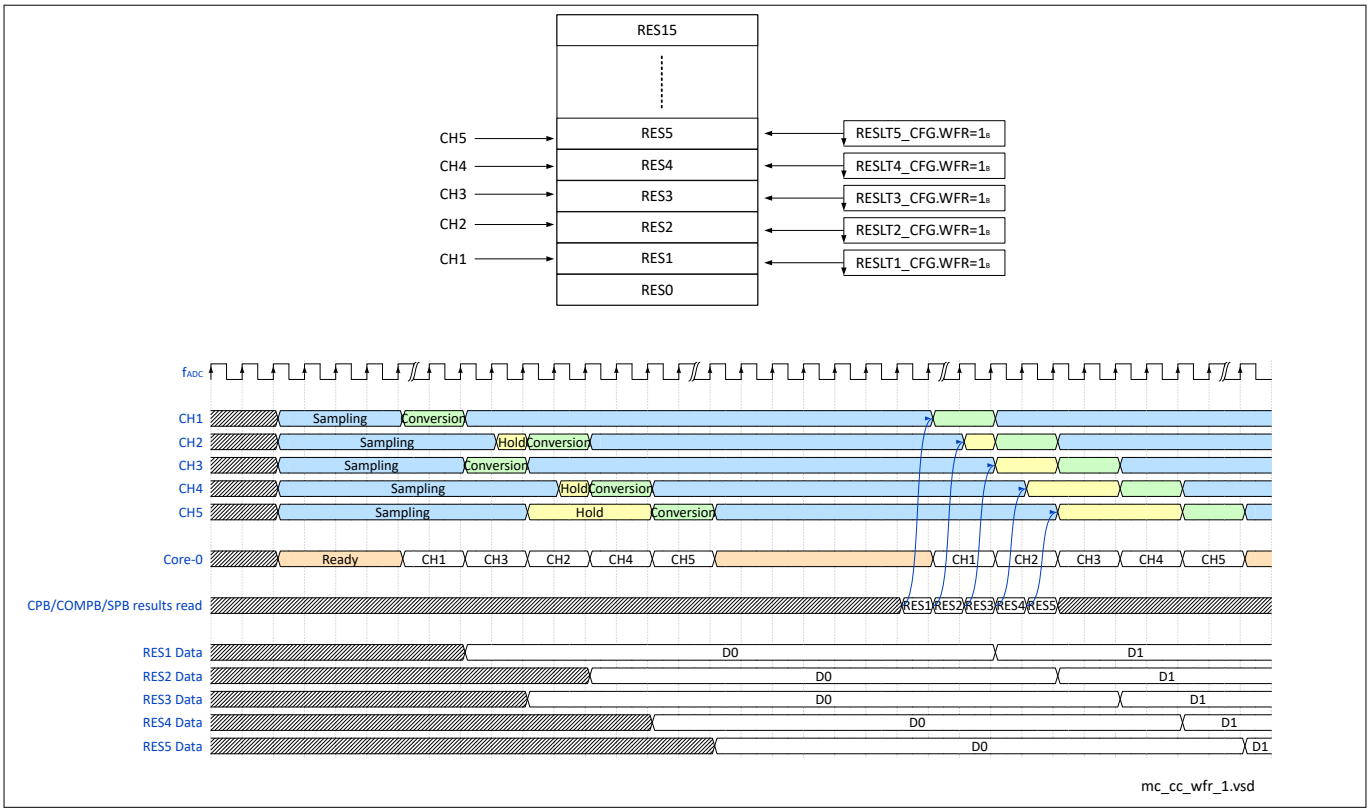


Figure 1204 Multiple channel group continuous conversion

Related information

Service request generation on page 8352

45.4.3.6 Conversion result handling

The converted result is stored in the channel-specific result register.

Related information

Storage of conversion results on page 8344
Boundary conditions on page 8348
ADC data interface (ADI) on page 8350
TMADC result read from CDSP on page 8351
TMADC result read through LLI on page 8352

45.4.3.6.1 Storage of conversion results

The channel conversion result is stored in the result register selected by TMADCx_CHy_CFG.RSEL. The result is stored in the bit-field RESULT in result register TMADCx_AW0_RESv

Additional status information such as channel conversion request by a hardware or a software trigger, hardware trigger number that initiated the channel conversion request, hardware trigger edge used for the conversion request initiation, TMADC channel number and broken wire or converter diagnostics data indication is available in the upper 2 bytes of the result register TMADCx_AW0_RESv.

Registers MRES0 and MRES1 are used to store the monitor channel results.

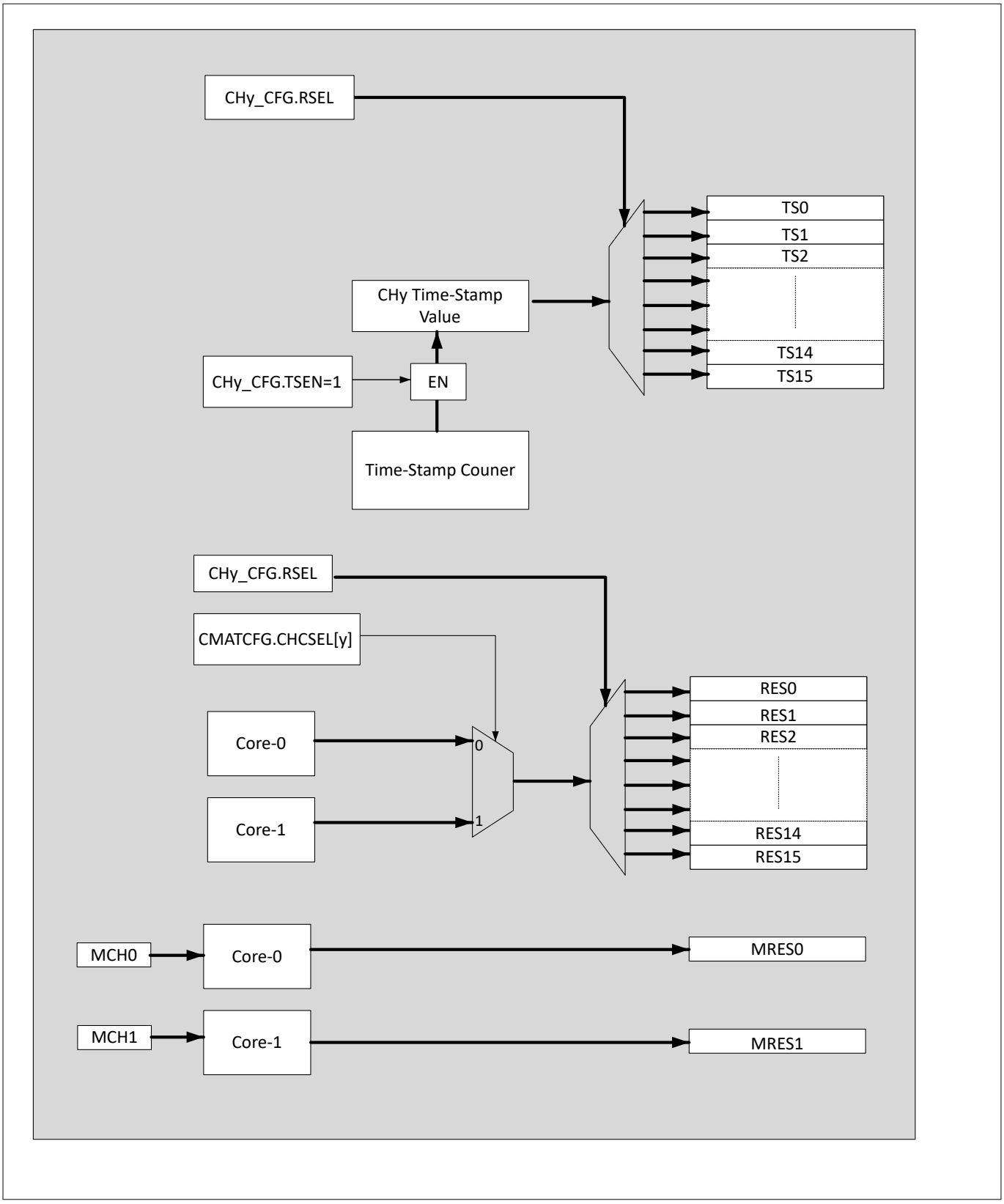


Figure 1205 TMADC time-stamp value and conversion result storage

***Note**: You must not select the same result register for more than one channel. It must be ensured by the user that the conversion result is stored in its channel-specific result register to avoid data collision. The only exception is the predictive maintenance application described in TMADC result read from CDSP chapter.*

Time-stamp support

Each TMADC kernel has a 16-bit time-stamp counter for generating time-stamp values for TMADC channels. This counter operates with a fixed 200 ns resolution and resets after every 13.1 ms. The counter is started by setting TMADCx_MODCFG.TSCRUN=1. The time-stamp value is the time-instant when a channel ends its sampling phase. This feature can be enabled for a channel by setting TMADCx_CHy_CFG.TSEN=1. The value is stored in the corresponding time-stamp register TMADCx_AW0_TSv selected via bit field TMADCx_CHy_CFG.RSEL for the channel.

The time-stamp value is stored at the same time in the time-stamp register as the corresponding result register data for the channel. If time-stamp feature is enabled for multiple channels, then user must ensure that independent time-stamp registers are selected for storing the time-stamp value for the channel. The time-stamp feature allows user to validate the channel's result for older or new data. In case a new timestamp is not equal to the old timestamp a new result has been generated.

CPB, COMPB orSPB address windows for reading result and time-stamp registers

Time-stamp and result registers read through peripheral bus (CPB, COMPB or SPB depending on variant) can take place from four available address windows, AW0, AW1, AW2 and AW3, as also shown in the following figure. The alignment of these registers in the address windows allows different type of burst read capability depending on the application use case.

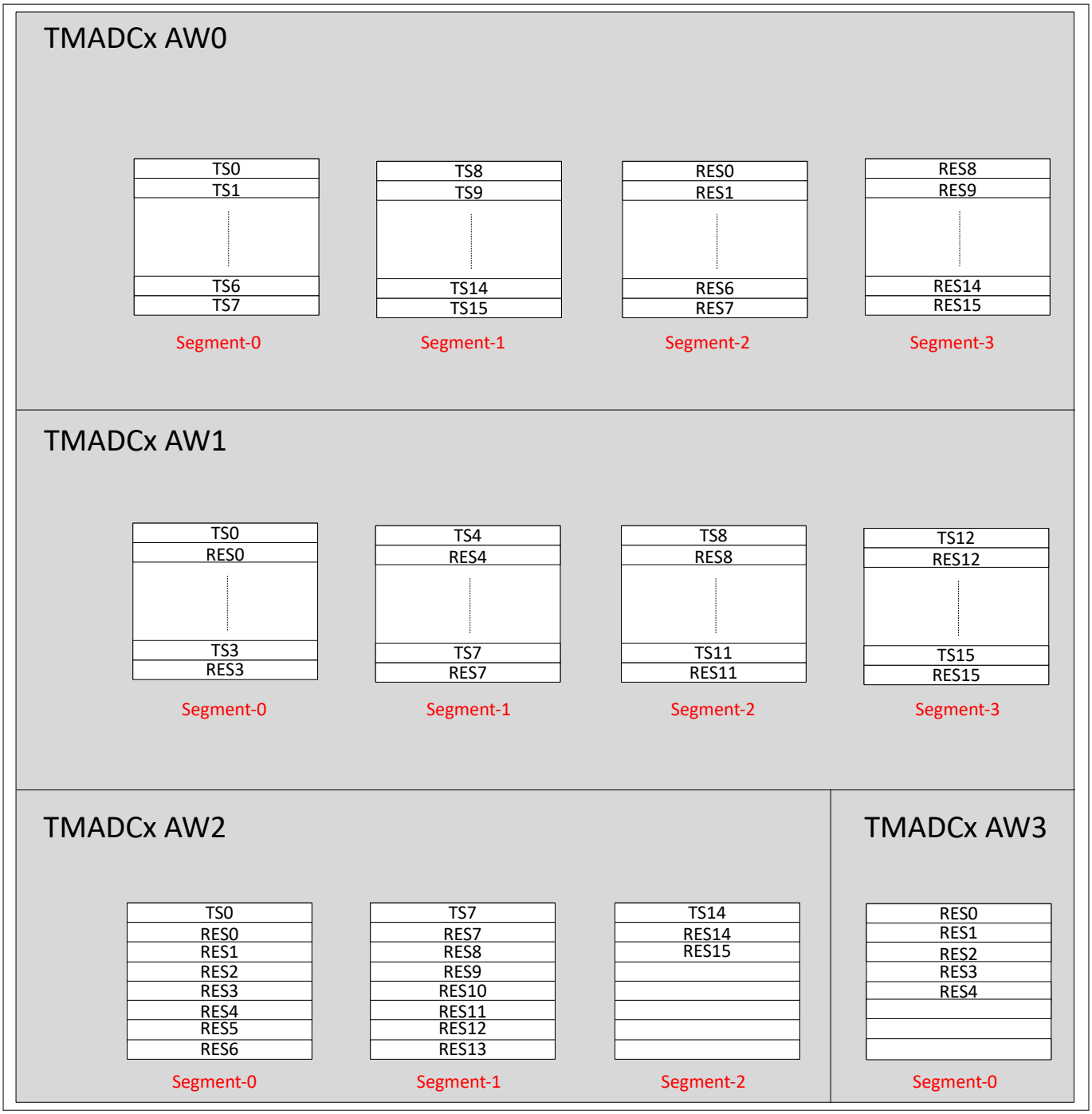


Figure 1206 Address window concept for result and time-stamp register read through peripheral bus (CPB,COMPB or SPB bus depending on variant)

A burst read from address window-0 (AW0) provides a read on the segment of time-stamp (TS0-TS7 or TS8-TS15) and result registers (RES0-RES7 or RES8-RES15) depending on size of the burst.

In address window-1 (AW1), the time-stamp registers and result registers are aligned in space-interleaved manner allowing a burst read of form TSx, RESx, TSy, RESy... so on, on each of the 4 address segments. Reading registers from address windows 2 and 3 (AW2, AW3) is useful in applications where multiple channels are grouped together on a single trigger event so that a single burst read on a segment fetches both the time-stamp value and channel group results (for example TS0, RES0, RES1, and so on). It is noted however only 3 time-stamp registers are available for read (TS0, TS7 and TS14) in AW2. AW3 allows to combine RES0-RES4 with TS7 and TS14. The register address space information on the address windows for each TMADC can be found in the register description chapter.

Related information

Registers on page 8391

45.4.3.6.2 Boundary conditions

Limit checking on the channel conversion result can be performed in TMADC itself to offload CPU tasks by issuing service requests only when results are within or outside the configured limit. The type of boundary check for a result is configured with TMADCx_RSLTu_CFG.BNDSEL value.

The boundary values can be defined through bit-fields UB for upper boundary and LB for lower boundary in the boundary register TMADCx_BNDn_SEL.

The selection of the boundary register to be used for the conversion result's limit check is done with TMADCx_RSLTu_CFG.BNDSELR.

The selection of the boundary region that is if the result is within or outside the configured boundaries is done through TMADCx_RSLTu_CFG.BNDMODE bit.

A boundary event flag is raised for the result when it fulfills the boundary check conditions.

A result is considered within the boundary under following conditions:

- Only the upper limit is used (TMADCx_RSLTu_CFG.BNDSEL=01_B) and the result is less than or equal to the boundary, otherwise result is outside the band
- Only the lower limit is used (TMADCx_RSLTu_CFG.BNDSEL=10_B) and the result is greater than or equal to the boundary, otherwise result is outside the band
- Both upper and lower limit are used (TMADCx_RSLTu_CFG.BNDSEL=11_B) and the result is greater than or equal to the lower boundary and less than or equal to the upper boundary, otherwise result is outside the band

The service request configuration on boundary events is described in detail (see chapter "Service request generation") and boundary status of results can also be read through boundary event flag register TMADCx_BNDF.

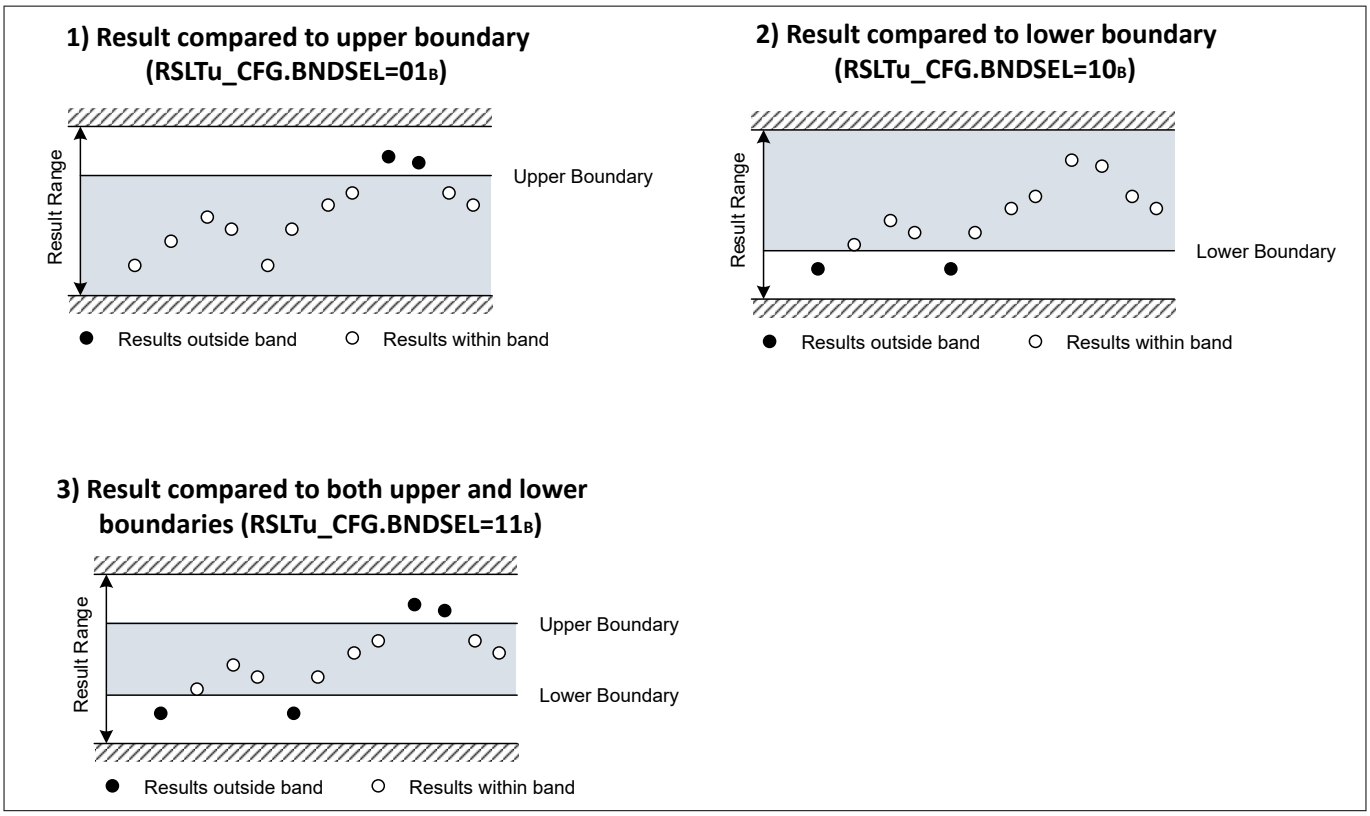


Figure 1207 Conversion result boundary conditions

Boundary values update condition

TMADC supports modification of boundary values in the TMADCx_BNDn_SEL registers during running application. There are 2 operation modes, active mode and shadow mode which define when updated boundary values become effective. The boundary update state of boundaries defined in registers BND0_SEL and BND1_SEL are controlled by the respective bit-fields BNDUPD0 and BNDUPD1 in the register TMADCx_MODCFG. Setting BNDUPDn = 0 enables the immediate update of the boundary value Setting BNDUPDn = 1 enables the update of the boundary value at the next hardware trigger event. The current boundary values can be read in TMADCx_BNDn_STAT register. The trigger configuration is done in the TMADCx_BNDn_TRGCFG register.

***Note**: After module configuration and enabling TMADC module (MODCFG.RUN=1), the configured boundary values will get initialized immediately. Afterwards update of boundary values is done immediately or with hardware trigger (shadow update) depending on TMADCx_MODCFG.BNDnUPD.*

Boundary flags

In the boundary mode, two boundary flags (BFL0 and BFL1) are provided.TMADCx_BFLn_CFG.RESEL defines the setected conversion result register corresponding with the boundary flag BFLn. The boundary flag value can be read through bit field BFL in the TMADCx_BFLn_STAT register.

Hysteresis mode

Hysteresis mode allows capturing different events (higher than upper boundary and lower than lower boundary) by timer modules using the same boundary flag signal. Configuring TMADCx_RSLTu_CFG.BNDSEL=11_B and setting TMADCx_RSLTu_CFG.HYSTM=1_B, enables the hysteresis mode where the upper and lower boundary values specified in the corresponding TMADCx_BNDn_SEL register form a hysteresis band for the selected channel result. In the hysteresis mode, the boundary value depends on the previous boundary flag output and the BNDMODE (BFL polarity) selected as shown below.

When BNDMODE=1:

- If previous BFL= 0, channel result is compared with SEL.UB
  - Next BFL = 1 if result > SEL.UB
  - Otherwise BFL = 0
- If previous BFL= 1, channel result is compared with SEL.LB
  - Next BFL = 0 if result < SEL.LB
  - Otherwise BFL = 1

When BNDMODE=0:

- If previous BFL = 0, channel result is compared with SEL.LB
  - Next BFL= 1 if result < SEL.LB
  - Otherwise BFL= 0
- If previous BFL = 1, channel result is compared with SEL.UB
  - Next BFL= 0 if result > SEL.UB
  - Otherwise BFL= 1

After module power-on the BFL value remains 0 until a comparison decision is available.

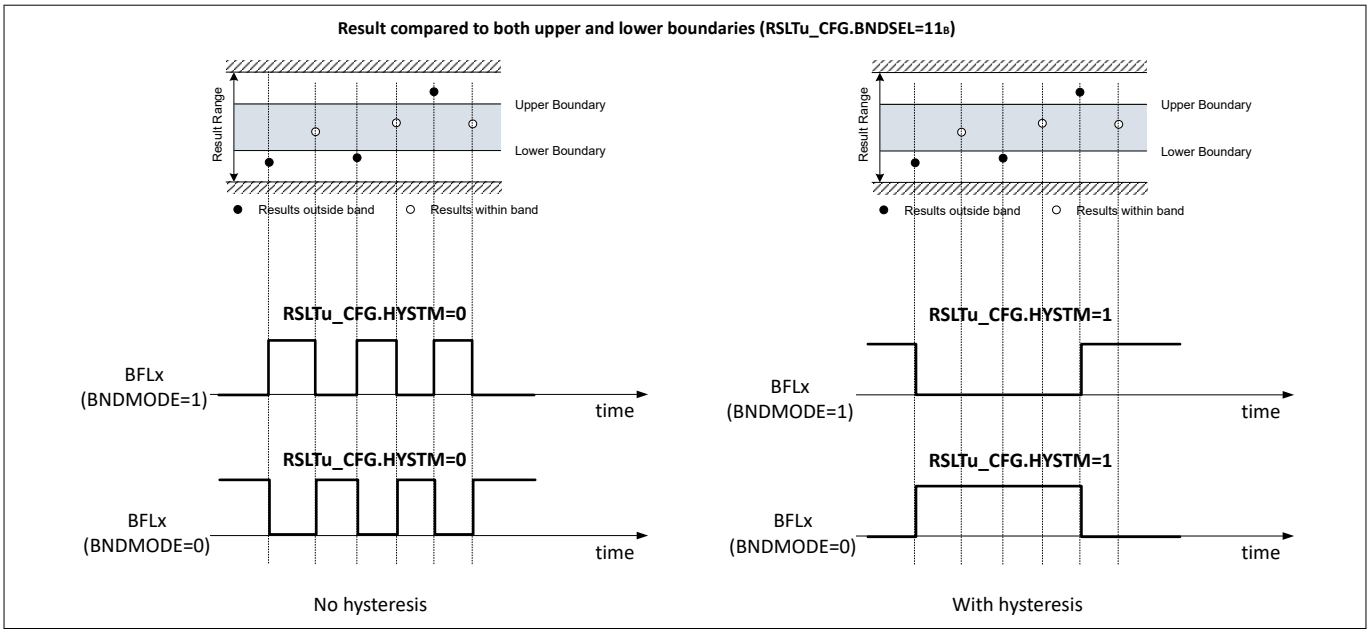


Figure 1208 Boundary conditions with and without hysteresis

Related information

Boundary flag interface to timer on page 8322
Service request generation on page 8352

45.4.3.6.3 ADC data interface (ADI)

The digital conversion results in result registers RES14 and RES15 from each TMADC instance are directly available at the TMADC/ADC top module boundary through the ADC data interface (ADI). Each time RES14 and RES15 are updated, the ADI updates the data vector.

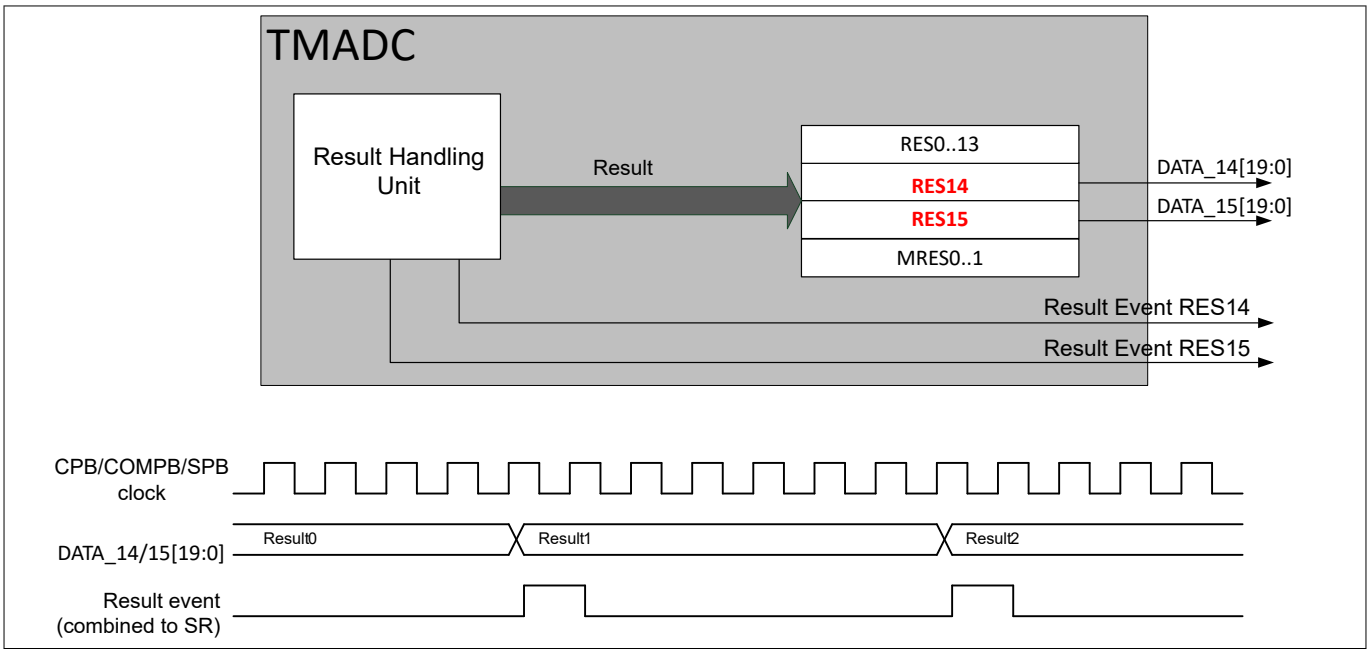


Figure 1209 ADI interface between TMADC and GTM

Qualification of the data can be done with service requests based on the respective result events. Service requests need to get routed to MCS (GTM component). MCS can process the data. The following data elements are available through the ADI:

Table 2384 ADI data assignment

Bit position	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Data content	CHNR				RESULT															

45.4.3.6.4 TMADC result read from CDSP

For post-processing of TMADC conversion results each TMADC module has 6 result registers allocated, RES0 to RES5, which can be read from any CDSP module. CDSP result data read from these registers does not consider the wait-for-read feature.

Wait-for-read if enabled will keep masking all the trigger requests until result is read from the result registers through CPB/COMPB/SPB slave, depending on variant.

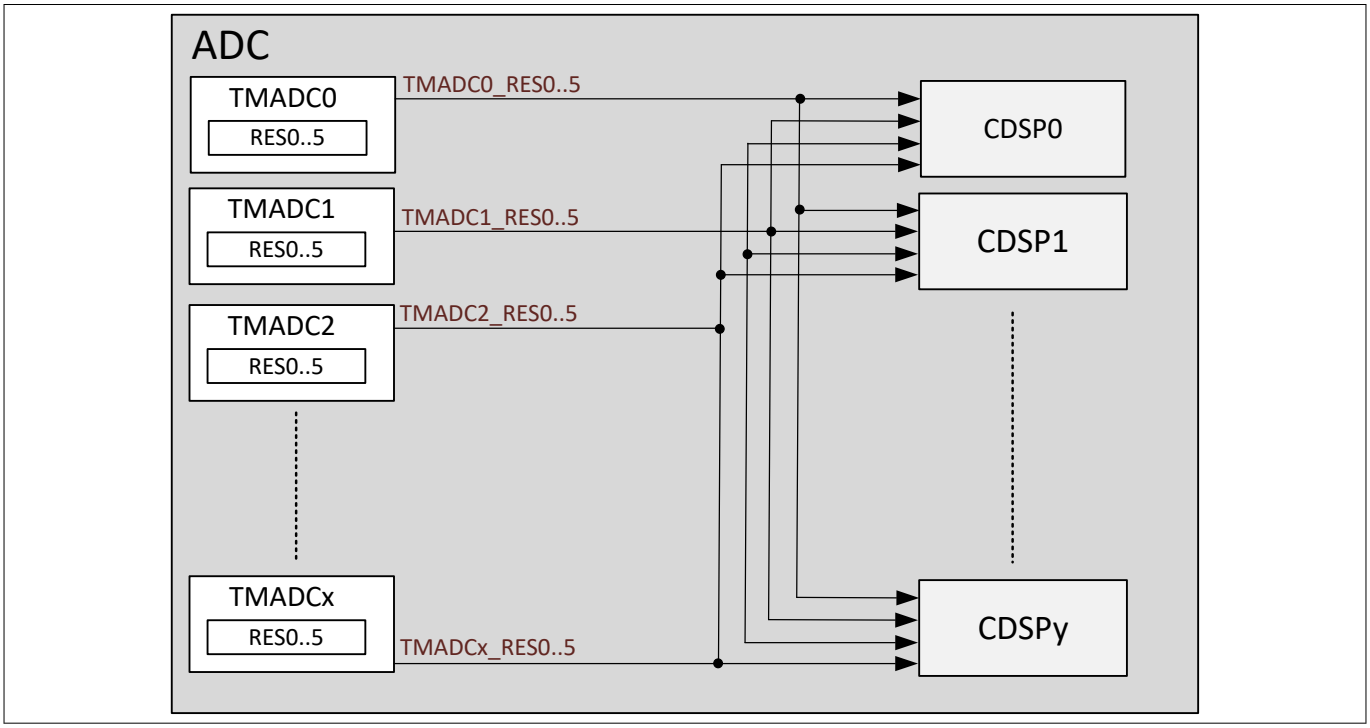


Figure 1210 TMADC result read from CDSP through AHB interface

Only the lower 16 bits of the result registers can be used as data for post-processing by the CDSP softare.

TMADC multiple channel results read from single CDSP

For predictive maintenance application, one CDSP core is used to post-process up to 16 channel results, but only RES0..5 are connected to CDSP inputs. In order to maintain the correct order of channel results which are read from CDSP, the following constraints shall be considered during the "CONFIG" state. No dynamic reconfiguration, no AUTOSAR™ support:

- Map all the used channels within the TMADC module for this application to the same CORE and same results register TMADCx_CHy_CFG.RESL=[0:5]. The CDSP shall select the same result register as input
- When other channels within the same module are used, they shall be mapped to another CORE and each of them shall be mapped to different unused result registers
- Use the same hardware trigger defined by either HWTR0MS or HWTR1MS for all channels. Only one hardware trigger is allowed
- No wait for read (TMADCx_RSLTu_CFG.WFR=0)
- Auto-trigger disabled (TMADCx_CHy_CFG.ARTEN=0), EMUX disabled (TMADCx_CHy_CFG.EMUXEN=0)

- The conversion delay between 2 successive channels shall be at least 50_H with 5 V supply mode or at least $A0_H$ with 3.3 V supply mode in trigger delay register in TMADCx_CHy_STC.TRD. The conversion delay between 2 successive channels equals TMADCx_CHy_STC.TRD*12.5 ns
- The minimum trigger period shall be 17*(TMADCx_CHy_STC.TRD*12.5 ns)
- Set minimal sampling time TMADCx_CHy_STC.ST= 28_H (500 ns) for all channels

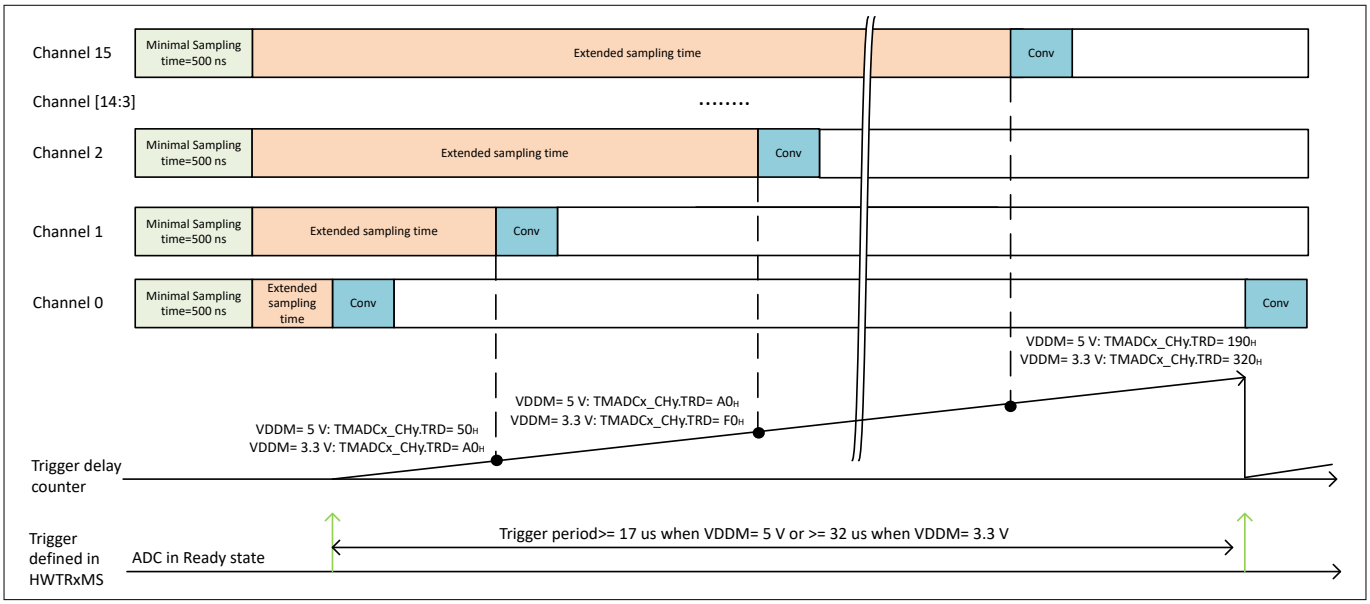


Figure 1211 TMADC multi-channel results read from single CDSP, round-robin controlled by trigger delay

***Note**: For other application, you must not select the same result register for more than one channel to avoid data collision.*

***Note**: The trigger shall satisfy the minimal sampling time definition. If the first trigger for the first channel comes during its minimum sampling time, then the trigger is ignored and no conversion will be started on this channel.*

45.4.3.6.5 TMADC result read through LLI

The time-stamp TMADCx_TSAXIv, TMADCx_RESAXIv input channel result registers and TMADCx_MRESAXIv monitor channel result registers can also be read over the low latency interconnect (LLI) interface. The wait-for-read feature is not valid for a result read through LLI.

45.4.3.7 Service request generation

Each TMADC instance can generate a total of 7 service requests (SR0...6) for issuing interrupts. Of those 7 service request, 6 (SR0...5) have direct connection to the Interrupt Router. Service request SR6 goes to the ADC global service request generator.

These service requests can be generated by three event types:

- Result event
  - Indicates a new result in the result register to trigger a read action. Result events are indicated with the result event flag setting in the TMADCx_RESF register. Software can clear the result event flag in the RESF register by writing to the corresponding result event flag clear bit in the TMADCx_RESFCLR register. For debug purposes application software can set the result event flag in the RESF register by writing to the corresponding flag set bit in the TMADCx_RESFSET register

- Boundary event
  - Indicates the boundary check comparison result for a conversion result. Boundary events are indicated with the boundary event flag setting in the TMADCx_BNDF register. Software can clear the boundary events by writing to boundary event flag clear bit in the TMADCx_BNDFCLR register
- Error event
  - Indicates channel error events that is an event of any missed channel trigger for each channel. Channel error events are indicated by the error event flag of the channel in the TMADCx_ERRF register. Software can clear the error event by writing to the corresponding error event flag clear bit in the TMADCx_ERRFCLR register. Clearing the error event flag also clears the TMADCx_CHy_STAT register of this channel

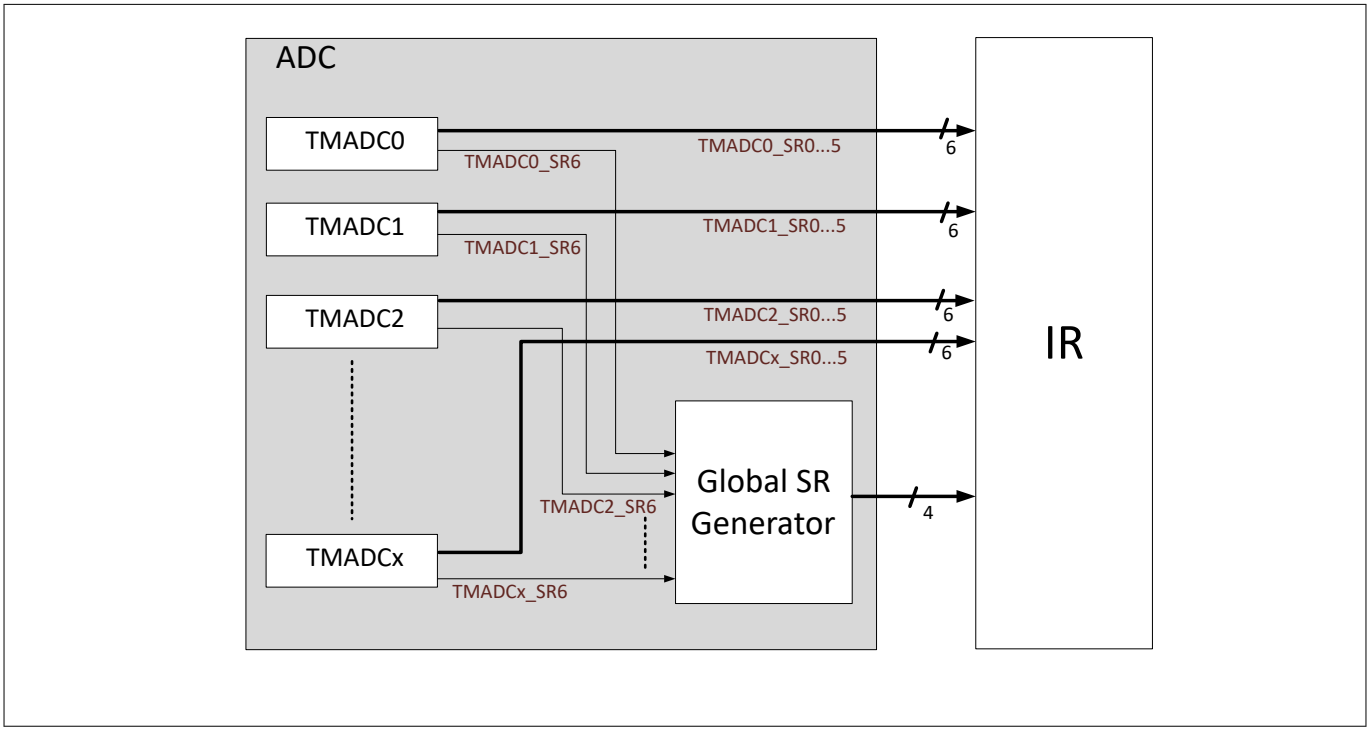


Figure 1212 TMADC service request generation overview

Each service request can be configured in the corresponding service request configuration register TMADCx_SRi_CFG. The meaning of the event enable bit-field TMADCx_SRi_CFG.EVEN depends on the setting of the event selection bit-field TMADCx_SRi_CFG.EVSEL as illustrated in the next table.

Table 2385 Service request event enable bit-field definition

SRi_CFG.EVSEL	SRi_CFG.EVEN
00_B (SRi disabled)	-
01_B (Channel error event)	Each bit corresponds to a channel
10_B (Result event)	Each bit corresponds to a result register
11_B (Boundary event)	Each bit corresponds to a result register

As shown in the next figure, you can group multiple events together by setting TMADCx_SRi_CFG.EVOP=10_B so that the service request is generated when all the user-configured events are available (AND logical operation on the enabled events).

Setting TMADCx_SRi_CFG.EVOP=00_B results in a service request generation for each user-configured event (OR logical operation on the enabled events).

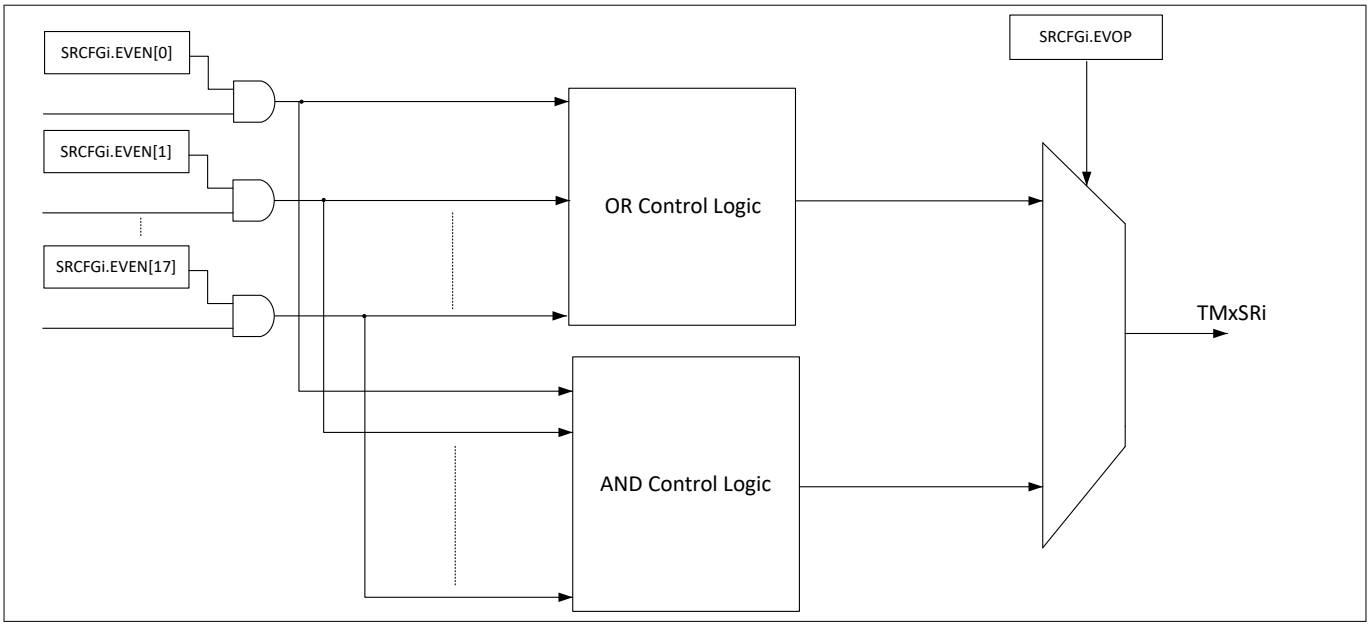


Figure 1213 Service request configuration

Result events

The following figures show examples of service request generation with different settings of the EVOP bit-field for the same configured result events (same EVEN bit-field settings). Conversion results for CH0,CH2, CH4, and CH6 are stored in result registers RES0, RES1, RES2, and RES3, respectively.

In the following figure using setting EVOP=00_B, the service request is generated each time a new result is available in any of the 4 configured result registers. Reading TMADCx_RESF returns the service request flags.

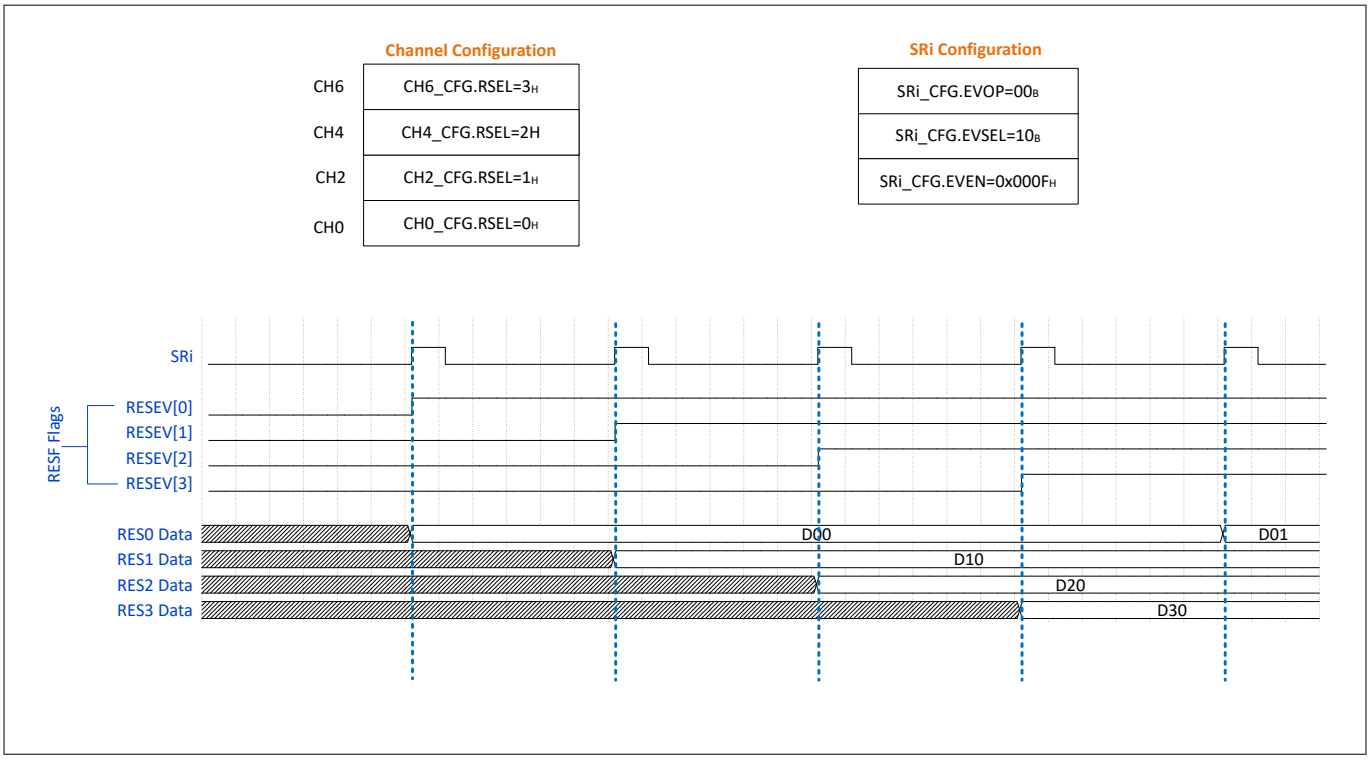


Figure 1214 TMADC service request generation on each valid result event EVOP=00_B

The following figure shows the behaviour if EVOP=10_B. The service request is generated once all 4 result registers hold new data. All RESEV flags in RESF register can be cleared using the RESFCLR register. SR gets generated for an event even when corresponding flag value is already high.

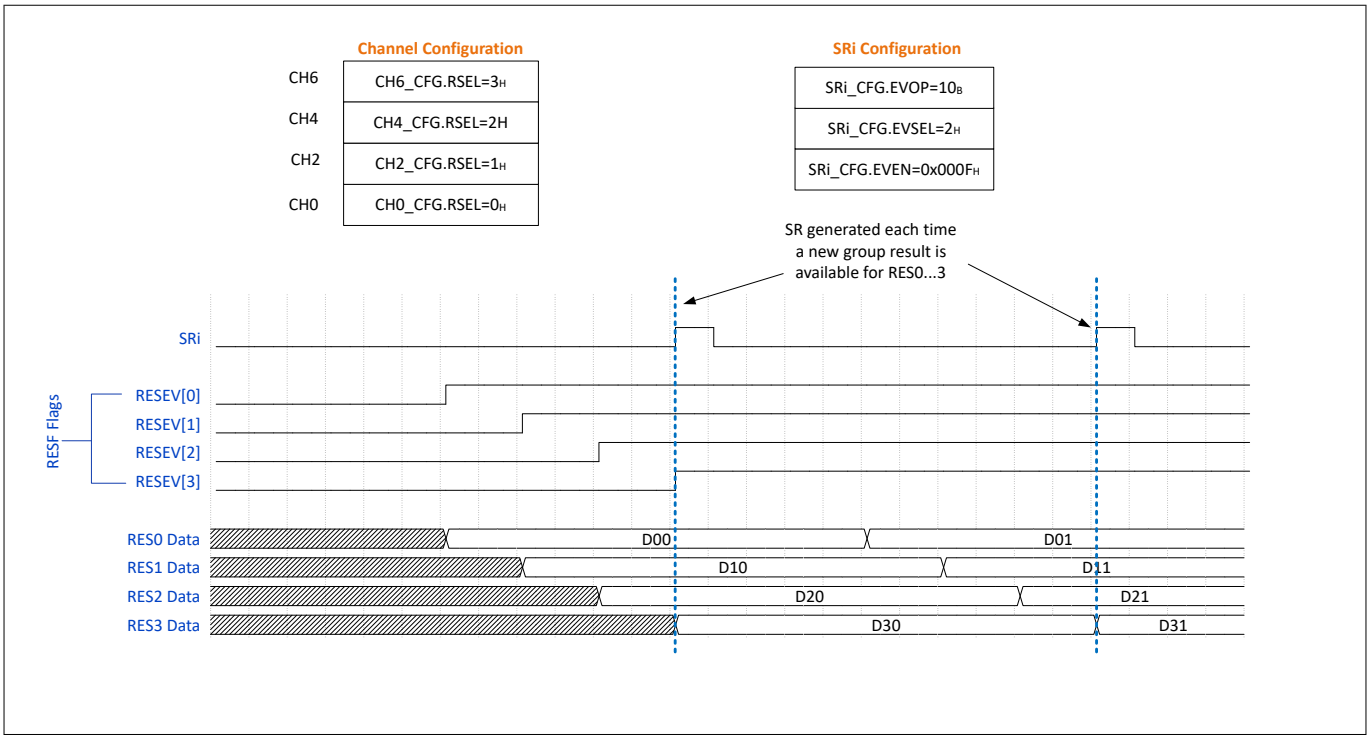


Figure 1215 TMADC service request generation on valid result group event EVOP=10_B

Boundary events

A service request is configured for boundary event(s) by setting TMADCx_SRi_CFG.EVSEL=11_B. You need to enable the individual boundary events for the results required to trigger the service request in the TMADCx_SRi_CFG.EVEN bit-field. Boundary event flags are generated only when the boundary mode for a result is enabled: That is TMADCx_RSLTu_CFG.BNDSEL≠ 00_B.

The following figure shows an example where SR gets generated for RES0, RES2, and RES3. There is no SR set for RES1.

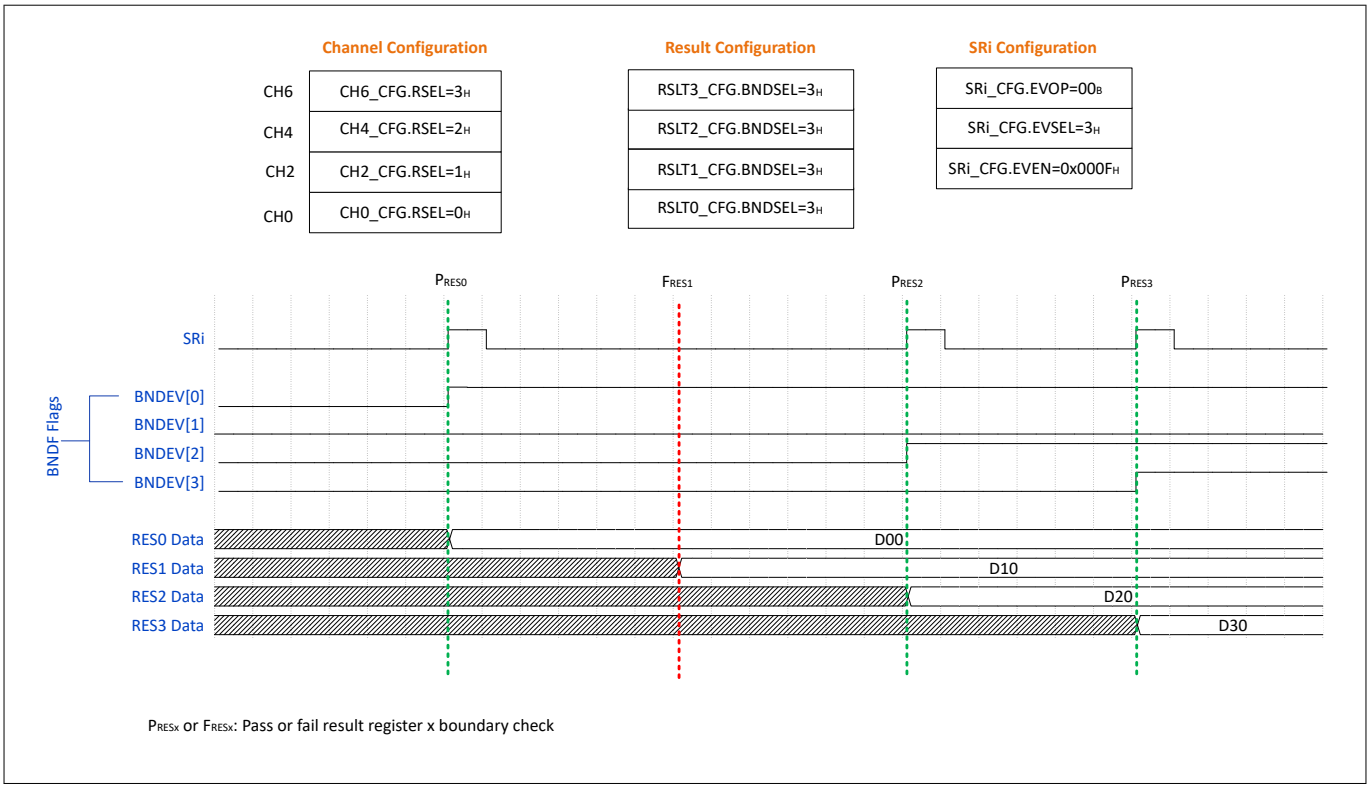


Figure 1216 TMADC service request generation: SR generated for RES0, RES2, and RES3

Another example is shown in the following figure where multiple results with boundary conditions are grouped a SR is raised, if all boundary event flags are set.

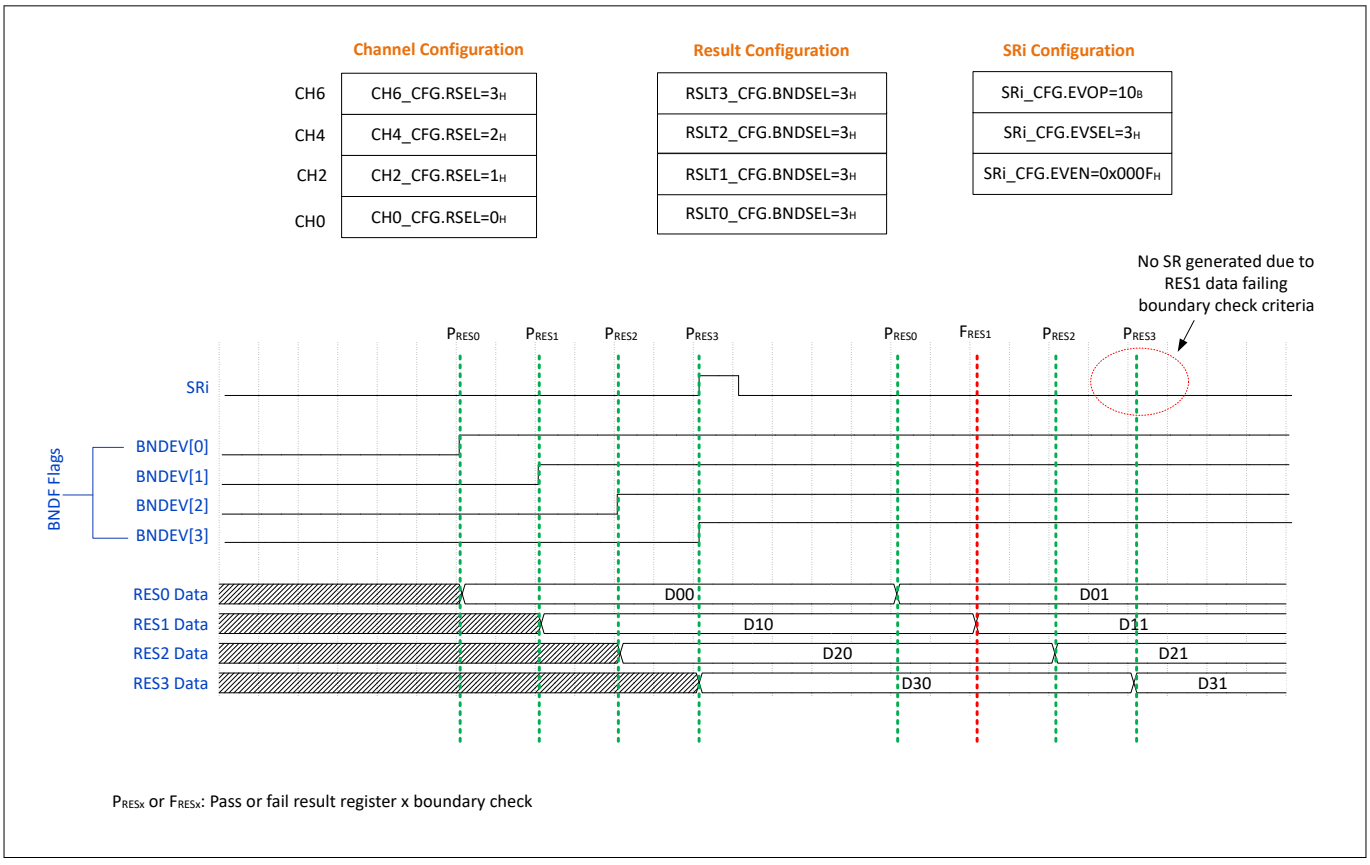


Figure 1217 TMADC service request generation: SR is raised, if all boundary event flags are set

Channel error events

Channel error events are ignored hardware or software conversion requests due to the arrival of triggers when a channel is busy. The ignored trigger event information is stored in the TMADCx_CHy_STAT register. The ignored trigger is discarded permanently and will not be repeated later.

A service request is configured for channel error event(s) by setting TMADCx_SRi_CFG.EVSEL= 01_B.

The individual error events for the channel required to trigger the service request are configured in the TMADCx_SRi_CFG.EVEN bit-field.

The following figure shows an example for the generation of a service request (SR) on an error event on CH0 and CH6. It is noted that the ERREV flag(s) in ERRF register once set after an event remains set and software shall clear the flag(s) by writing to the flag clear bit(s) in the TMADCx_ERRFCLR register. SR gets generated for an event even when the corresponding flag value is already high.

***Note**: Channel error events are configured with TMADCx_SRi_CFG.EVOP=00_B to generate an interrupt when any error event occurs.*

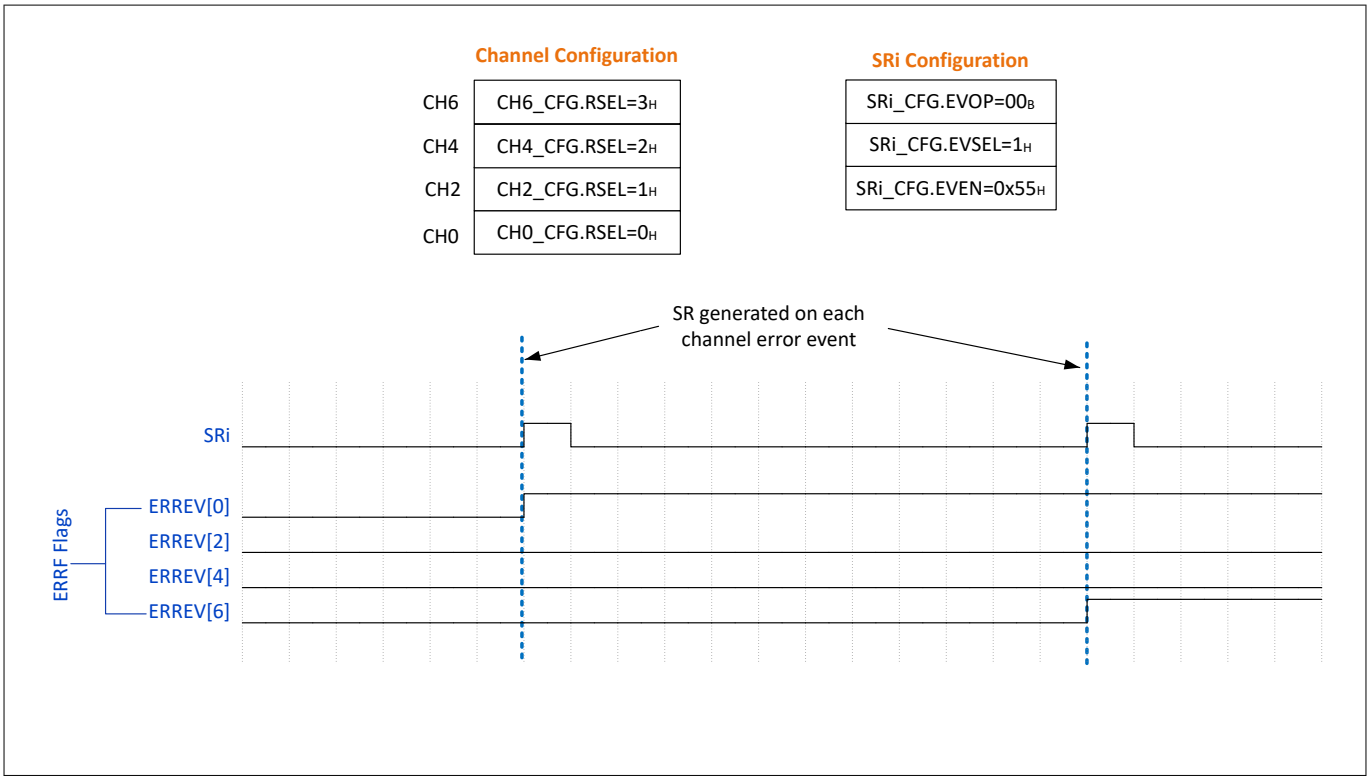


Figure 1218 TMADC service request generation on each channel error event

Wait-for-clear

The wait-for-clear functionality for the service request allows you to stall subsequent service request generation until the software clears the corresponding event flag register bit(s). This feature is enabled by setting TMADCx_SRi_CFG.EVOP=01_B.

The following figure shows an example of the wait-for-clear feature. The service request is generated on the first result event, in this case after RES0 data is available, and only after the software clears the RESEV[0:3] bit-field in the RESF register. The service request is generated again on the next available result event, in this case after RES2 data is available.

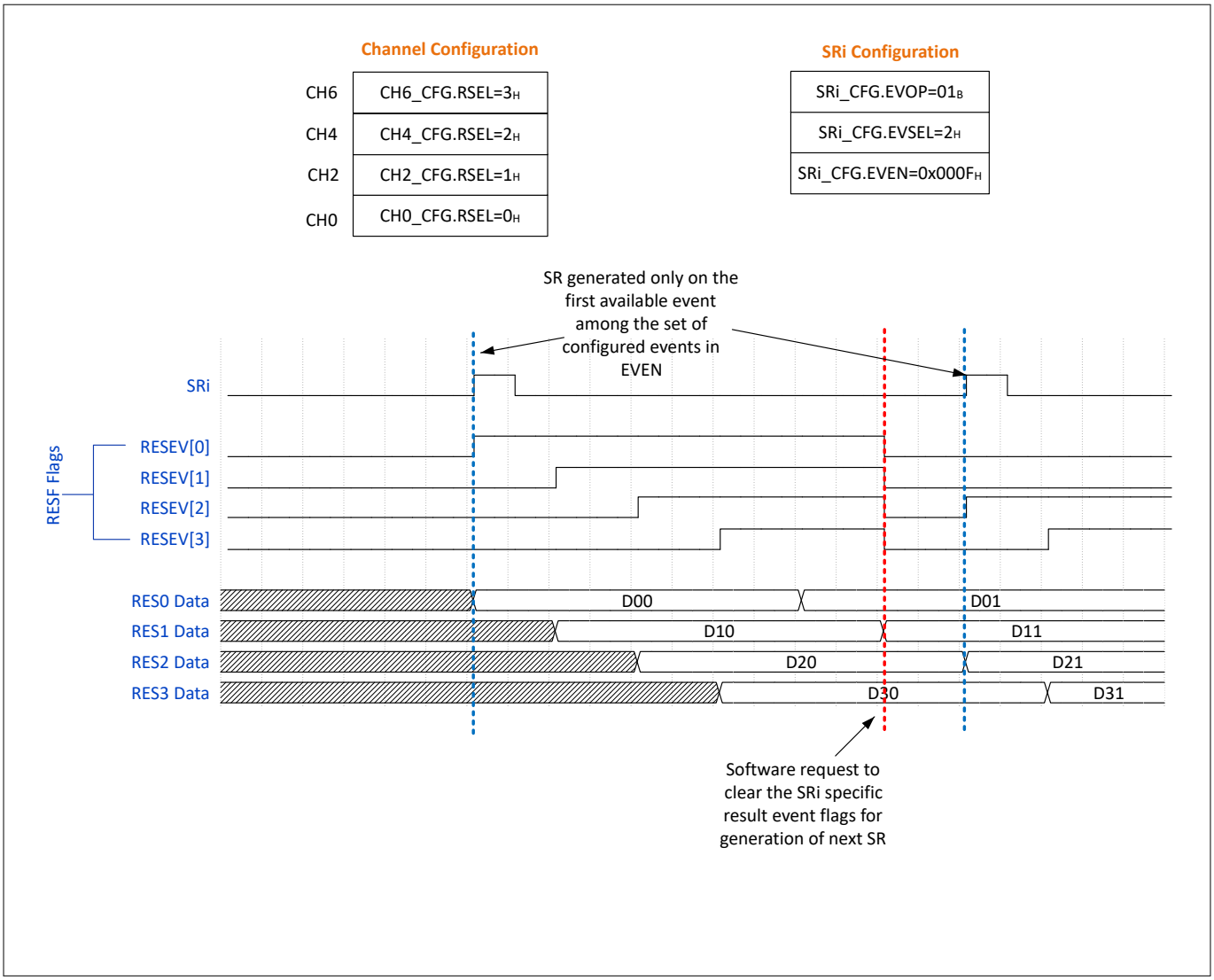


Figure 1219 TMADC service request generation with wait-for-clear feature

Related information

Global service request and alarm generation on page 8317

45.4.3.8 Diagnostics

TMADC diagnostic features are used to check the correct operation of the converter and to verify connection to the external sensor during a running application.

45.4.3.8.1 Channel-core self diagnostics (CSD)

A CSD check can be used to verify the correct functional operation of the converter during a running application. It provides the ability to sample known good values based on of VAREF and (if supported in device variant) VDDM supply.

During CSD a channel samples a known voltage (in example reference voltage). Evaluating the conversion result indicates if the converter is operating correctly. Observations of deviations outside tolerance (refer to the respective datasheet for CSD result variation) are an indication of a potential fault on a channel or core.

The CSD mechanism is configured in the core specific TMADCx_CSDn register. The CSD can be enabled for one TMADC channel (of that specific core) at a time. This is selected through the bit-field CHSEL in the CSDn register. The channel selected here should already be mapped to the core in the CMATCFG register.

The voltage to be sampled on the channel during CSD is selected through bit-field TMADCx_CSDn.SEL. 2 implementations do exist. Please refer to the description of the TMADCx_CSDn register in the device specific appendix to identify it. Variants are shown in the table below.

Table 2386 CSD configuration options

TMADCx_CSDn	Implementation 1	Implementation 2
00_B	(1972/4096)*VAREF	Shall not be used
01_B	(220/4096)*VAREF	Shall not be used
10_B	(1972/4096)*VDDM	Sampler CSD, VAREF, transfer curve shifted by 16 LSBs to match 0x07B4
11_B	(220/4096)*VDDM	Sampler CSD, VAREF, transfer curve shifted by 16 LSBs to match 0x00DC

Implementation 1 allows measurement of VDDM and VAREF fractions. Implementation 2 is based on measurement of VAREF, but the transfer curve of the converting core is modified. Therefore a positive error component can still be detected without saturating to the full-scale output code.

CSD is enabled for the channel with setting TMADCx_CSDn.EN=1 for the corresponding core.

A CSD request on a channel in sampling, hold or conversion phase is stalled until the end of its current conversion. With next sampling start the CSD setting is considered and CSD results are generated with this subsequent conversion onwards. When CSD is disabled the next result will still be with CSD settings and subsequently normal conversion results without CSD settings will be generated.

The CSD enabled channel conversion result is indicated with TMADCx_AW0_RESv.DR=1_B in the corresponding result register. Monitoring channels have dedicated result registers, where the CSD conversion can get read.

An example is shown in the figure below for CSD with respect to a channel CH5..

***Note**: To avoid VAREF loading effects, during CSD only one channel per TMADC should be configured to sample a fraction of VAREF at any given time.*

By default internal monitoring channels MCH0 and MCH1 are connected to Core-0 and Core-1 respectively. Performing CSD on these channels is done by setting CSD0.CHSEL=10_H and MCH0_CFG.SEL=11_B for MCH0 and CSD1.CHSEL=11_H and MCH1_CFG.SEL=11_B for MCH1, respectively.

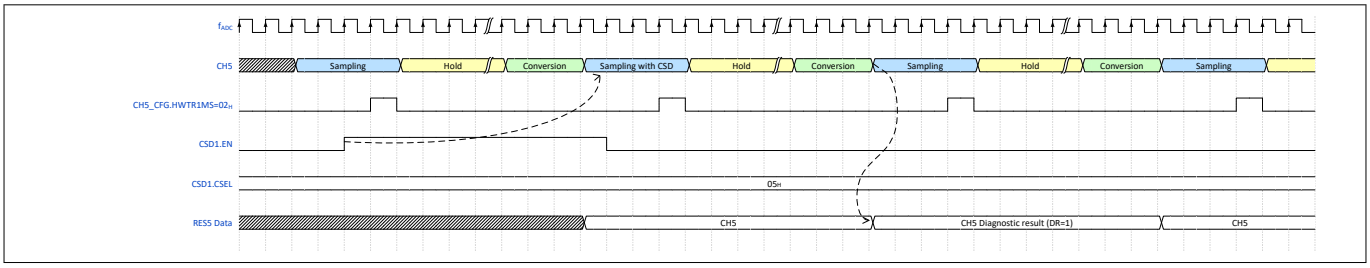


Figure 1220 Example of converter self-diagnostic sequence

45.4.3.8.2 Broken wire detection

Testing the proper connection of the external sensor to the TMADC's analog input pin can be done by enabling the broken-wire detection (BWD) mode on a channel.

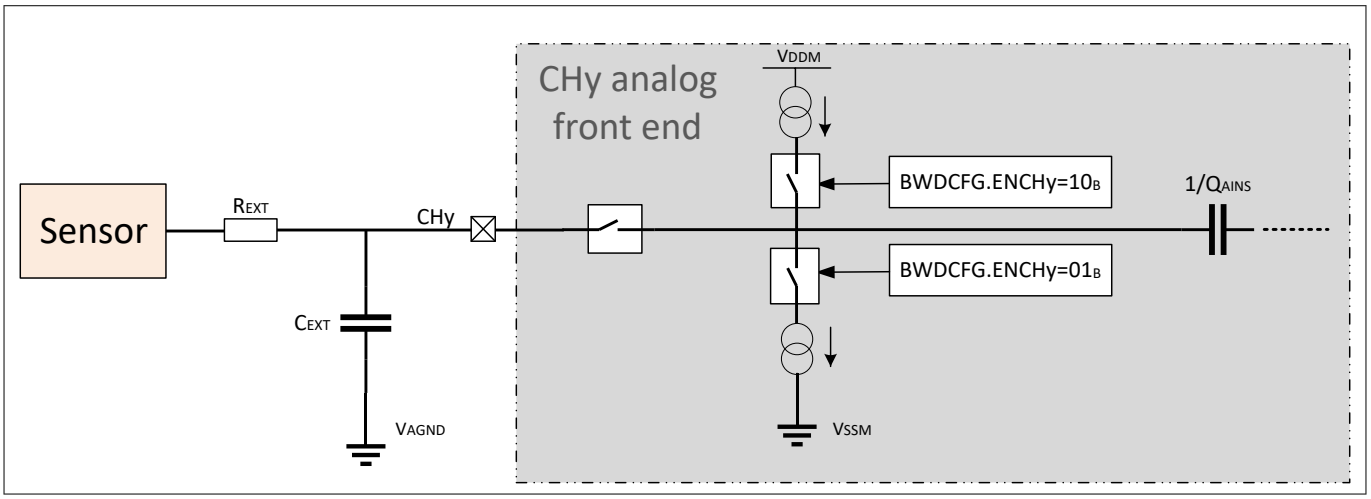


Figure 1221 Broken-wire detection circuit

A BWD mode is enabled with the setting of bit ENCHy for the respective channel's input pull-up or pull-down in the TMADCx_BWDCFG register.

The input pull-down current source can be enabled with the setting ENCHy=01_B.

The input pull-up current source can be enabled with the setting ENCHy=10_B.

The channel input is pulled up or down depending on the configuration, with the enabled current source during the sampling phase. It is disabled during conversion phase. The application software shall verify the expected behavior on the channel input with data reads on the channel conversion result.

Activation of the current sources will happen for the next full sampling phase following the activation. Matching the timing relation shown in the Channel-core self diagnostic (CSD) section.

The expected result is a function of the sensors current, the channels total sampling time and the channel capacitance.The circuit to be considered is dominated by elements outside the MCU.

Please check the device specific datasheet for the specified parameter values and consider:

- $1/Q_{AINS}$ is the sampling capacitor equivalent, datasheet value, likely to be neglected in comparison to C_{EXT}.
- C_{IO} + 2.5pF (representing the package) have to be added to C_{EXT}, but are likely neglectable in comparison to C_{EXT}.
- C_{EXT} is the off-chip external capacitor (external element, selected by user)
- I_{BWCHG} and $I_{BWDISCHG}$ are the broken wire pull-up and pull-down currents, respectively
- I_{OZ} is the pin leakage current (datasheet value, reduces I_{BW} currents)

Related information

Channel-core self diagnostics (CSD) on page 8358

45.4.3.8.3 Pull-down diagnostics (PDD)

Some TMADC modules provide a strong pull-down capability on certain analog input pins to verify the external sensor connection. The strong pull-down quickly discharges a large external buffer capacitor connected on the channel and therefore allows for the faster evaluation of diagnostic results.

PDD is enabled on the channel by setting PDDCFG.ENx=1_B.

Disable ENx and read the subsequent channel conversion result data to identify a broken sensor connection.

***Note**: Analog input pins with pull-down diagnostics device are marked in the device specific chapter.*

45.4.3.9 On-chip signal monitoring and supervision

For the purpose of monitoring on-chip signals during running application, there are dedicated internal monitor channels MCH0 and MCH1 which are mapped to TMADC channels 16 and 17.

The signal selection for conversion by the monitor channels is made by bit SEL in the TMADCx_MCHy_CFG register. The selection signals are given in following tables for the 2 monitor channels. The input connections to TMADCxMCHyIN1 are given in the device specific datasheet.

Channels MCH0 and MCH1 are by default connected to core-0 and core-1. The monitor channels are enabled by setting TMADCx_MCHy_CFG.EN=1_B. The minimum sampling time configuration for the monitor channel signals is 1 µs (TMADCx_MCHy_CFG.ST $04F_H$).

Conversion request on monitor channel(s) can be initiated on software trigger(s) configured in TMADCx_SWTRCFG.SCHSEL (bit 16 or bit 17) and TMADCx_MCHy_CFG.ATREN=1_B for the monitor channel. Software trigger is processed only during the monitor channel's sampling phase. When configured with TMADCx_MCHy_CFG.ATREN=1_B, the monitor channels do not require any external triggers for end of sampling and automatically transitions to hold phase when their configured minimum sampling time is over and get added to the respective core's round-robin arbitration queue.

The conversion results for channels MCH0 and MCH1 are stored in result register (TMADCx_MRESv) MRES0 and MRES1. Enabling wait-for-read (TMADCx_MRSLTu_CFG.WFR=1) on the monitor channel result stalls the subsequent conversion of the monitor channel signal until the corresponding result is read by the software. A change in the monitor channel's signal selection during the channel sampling, hold or conversion phase is considered only after the current conversion is completed. The figure below does show an example update sequence switching the monitoring input.

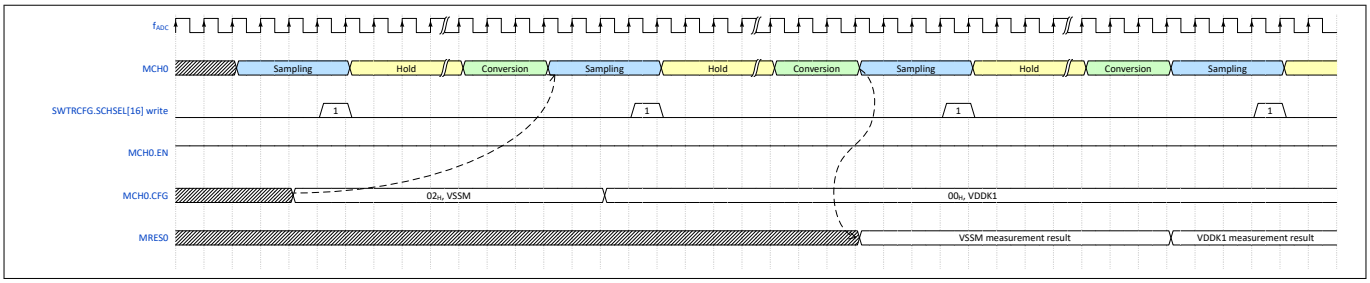


Figure 1222 Monitoring update sequence

Boundary checks can be applied on the monitor channel results by configuring the boundary mode settings in the MRSLTu register.

Each TMADC module has an output supervisor multiplexer to select signals for monitoring by another TMADC. The selection is made through bit-field OSSEL in the TMADCx_MODCFG register.

Please see the device specific chapter to check the TMADC output supervision signal connection mapping to the other TMADC module.

Table 2387 Monitor channel 0 signals

MCH0_CFG.SEL	Signal name	Signal description
0	MCH0IN0	Core-1 supply voltage (VDDK1), TMADC x
1	MCH0IN1	Supply voltages from other modules
2	MCH0IN2	VSSM
3	MCH0IN3	CSD monitor channel 0

Table 2388 Monitor channel 1 signals

MCH1_CFG.SEL	Signal name	Signal description
0	MCH1IN0	Core-0 supply voltage (VDDK0), TMADC x
1	MCH1IN1	Supply voltages from other modules
2	MCH1IN2	VSSM
3	MCH1IN3	CSD monitor channel 1

The accuracy of VAREF can be tested by measuring an internal converter core-specific bandgap signal VDDKx for example VDDK0 or VDDK1. The variation of VDDKx level depends on manufacturing deviation, supply voltage VDDM, and the die temperature. To compensate for this variation, converter core-specific correction values are measured during production at hot and cold temperature points and are stored in the on-chip Flash.

A pair of correction values for each core contains an offset (in [mV]) from a typical TMADC IVR output voltage level (V_{DDK} Controller Characteristics (CC)) measured at -40°C and 127°C. The values are recorded in the 16-bit 2's complement format. To compute the expected VDDK value, the user has to interpolate the correction value for the specific temperature (Die Temperature Sensor (DTS) reading) and reference the interpolated value to the typical IVR output voltage (V_{DDK} CC) value.

The following formula uses the current die temperature (T_j in [°C]) to compute the expected converter core-specific VDDK0 voltage level:

$VDDK0(T) = IVR_typ\ (V_{DDK}\ [mV]) + (dVDDK0(-40°C) + ((dVDDK0(127°C) - dVDDK0(-40°C))/167°C) * (T_j + 40°C))$

***Note**: On top of the computed VDDK range, the user has to also consider the tolerance of DTS and the accuracy of TMADC (TUE and noise). Ripple and noise on the VAREF rail can additionally affect VDDK measurement accuracy. For more reliable VDDK measurements it is recommended to average several consecutive samples.*

The converter core-specific correction values are stored in the User Configuration Block (UCB0) within the on-chip Flash memory. The area assigned to the TMADC begins at the offset 13004_H from UCB0 and provides records of two 32-bit words per TMADC instance. The lower half-word of a record stores the VDDK0 correction value (dVDDK0) and the next half-word stores the VDDK1 correction value (dVDDK1) as shown in Table 2389.

***Note**: Converter core-specific correction records are stored in the 16-bit 2's complement format and can represent both positive and negative values.*

Table 2389 Structure of TMADC correction values (dVDDK)

	Upper 2 bytes	Lower 2 bytes	Upper 2 bytes	Lower 2 bytes
Location TMADC0	$1300A_H$	13008_H	13006_H	13004_H
Location TMADC1	13012_H	13010_H	$1300E_H$	$1300C_H$
:	:	:	:	:
Stored value	dVDDK1(-40°C)	dVDDK0(-40°C)	dVDDK1(127°C)	dVDDK0(127°C)

Related information

TC4Dx ADC configuration on page 8548

45.4.3.10 External multiplexer control

For applications requiring more than 16 channels to be converted per TMADC kernel, external analog multiplexer (EMUX) inputs can be connected to TMADC channels. TMADCs can get configured to control one out of four 4-bit EMUX control interfaces (EMUXCTRLx[0:3], x= 0-3).

The input channel driven by an external multiplexer has to be exclusively mapped to one of the TMADCs SAR cores. All other channels need to get mapped to the alternative SAR core.

This allows to use the wait for read feature on the EMUX channel result without interference to other channels. Prerequisite for AUTOSAR support of the EMUX feature.

The TMADC channel on which the EMUX is connected should be set to the configuration TMADCx_CHy_CFG.EMUXEN=1. The TMADC is expecting an external MUX for the selected channels as shown in the example below for channel number 3.

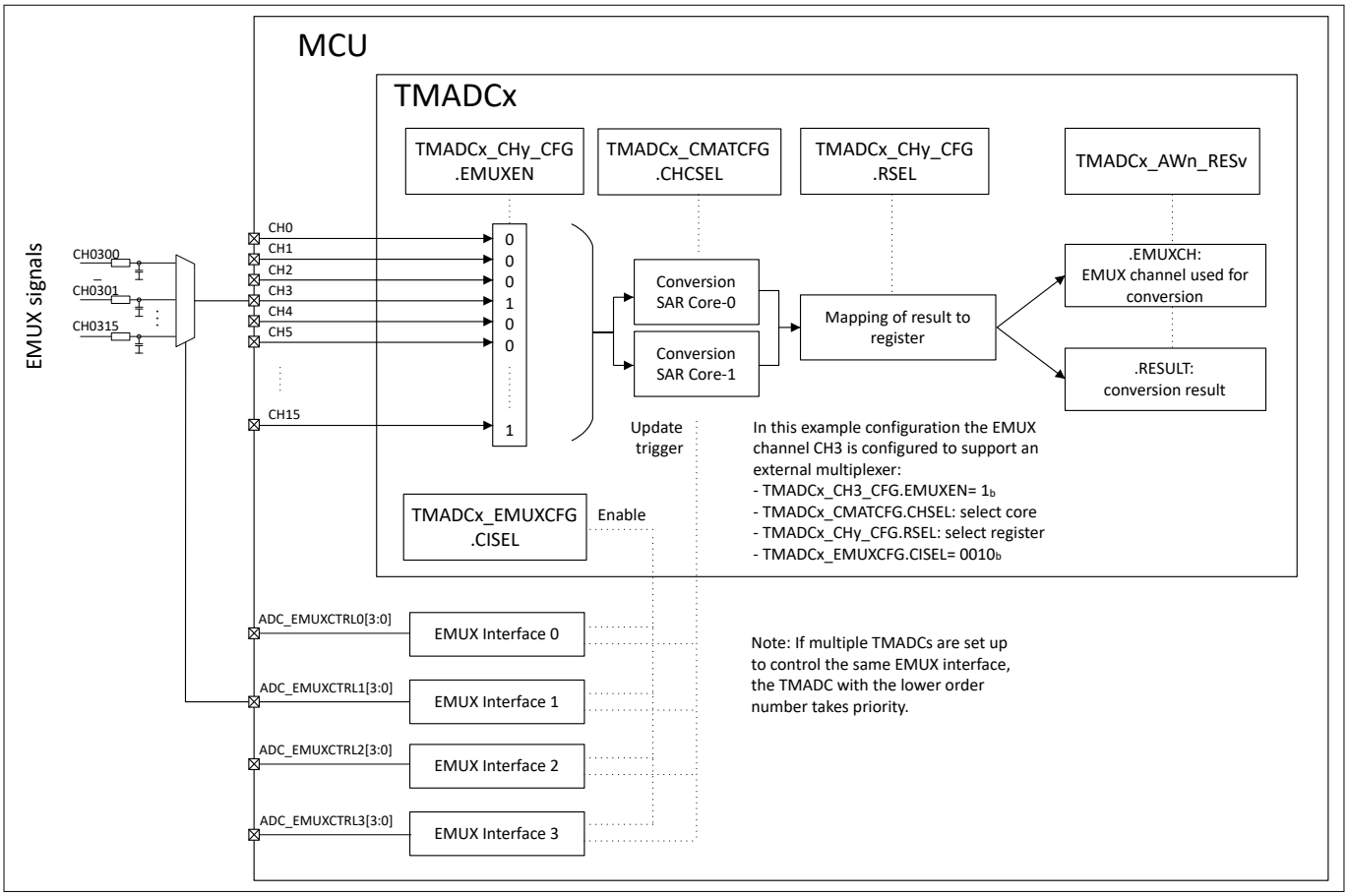


Figure 1223 TMADC EMUX control

It is possible to use up to external 16:1 multiplexers for up to 4 TMADC channels. If multiple TMADCs are set up to control the same EMUX interface, the TMADC with the lower order number takes priority.

Constraints that apply for EMUX channels

- The EMUX input channel needs to get exclusively mapped to a single SAR core (either core 0 or core 1). No other channel is allowed for that SAR core
- Switching the external multiplexers can introduce worst case transitions in the analog input signal. To suppress non-linear effects in the input channel, the minimum sampling time for EMUX channels is 1.6 µs
- Hardware trigger must be used for the EMUX channel, no automatic triggering (TMADCx_CHy_CFG.ARTEN= 0_B)
- No use of monitoring channel on the SAR core selected for EMUX use
- External multiplexer impedance adds to R_{EXT}

All the EMUX input conversion results for a single TMADC channel are stored in the same channel-specific result register configured through bit-field TMADCx_CHy_CFG.RSEL.

EMUX channels must be enabled setting TMADCx_CHy_CFG.EN=1 in CONFIG state.

EMUX operation

The EMUX channels starting from EMUX channel number 0 to the channel number specified in the TMADCx_EMUXCFG.CHCNT bit-field are selected for conversion one-by-one. After the last channel in the sequence is converted, the sequence starts from channel 0 again.

The figure below shows an example of timing waveforms for EMUX connection on CH3 with TMADCx_EMUXCFG.CHCNT = F_H for an arbitrary TMADC.

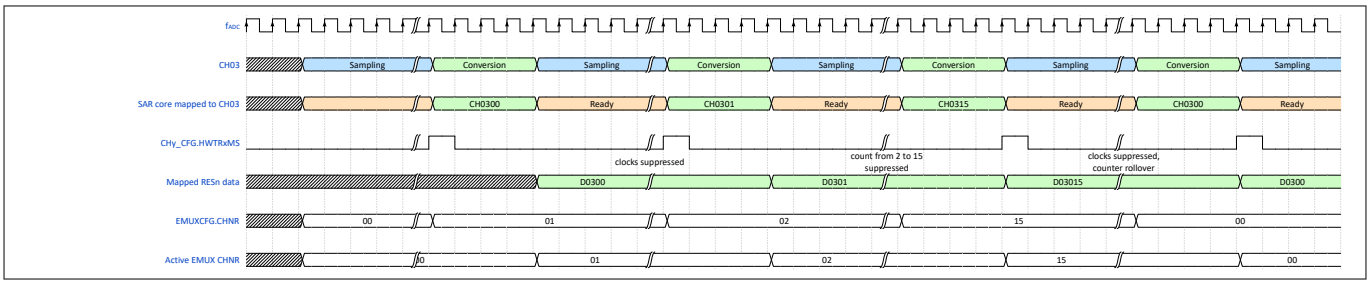


Figure 1224 EMUX automatic channel selection mode

The conversion results are stored with the corresponding EMUX channel select in the 4 most significant bits of the timestamp register (TMADCx_AW0_TSv.EMUXCH).

EMUX control signals

The EMUX control interface signals can be output in standard 4-bit binary or gray-coded format. Gray code avoids intermediate multiplexer switching when selecting a channel. The channel selection coding scheme is configured in the CS bit-field in the TMADCx_EMUXCFG register.

Table 2390 EMUX control signal coding

Channel	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
Binary	0000_B	0001_B	0010_B	0011_B	0100_B	0101_B	0110_B	0111_B
Gray	0000	0001	0011	0010	0110	0111	0101	0100
Channel	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**
Binary	1000_B	1001_B	1010_B	1011_B	1100_B	1101_B	1110_B	1111_B
Gray	1100	1101	1111	1110	1010	1011	1001	1000

45.5 Converter digital signal processor (CDSP)

The CDSP is a hardware platform for customized data processing. Depending on the variant, it consists of up to 18 independent processor cores with corresponding control and computation logic. For variant specific details please refer to the device specific User Manual appendix.

The cores utilize a 32-bit architecture with highly efficient three-stage pipeline.

Each core can be programmed and configured individually to process data from Delta Sigma ADC (DSADC), External Modulator support (EXMOD), Time Multiplexed ADC (TMADC), Carrier Pattern Magnitude (CARMAG) or General Purpose (GP) registers (see chapter "Input and Output Data Interface of CDSP").

Filter chains and algorithms provided by Infineon to serve different applications are for example:

- Configurable digital filter chains
- Fourier transformation algorithm for up to 512 points
- Angle tracking observer

Parameters of the Infineon CDSP software remain configurable in order to support existing and enable new applications. Details are described in the AURIX™ TC4xx CDSP Software User Manual.

Related information

Input and output data interface on page 8368

45.5.1 Feature list

- Individually programmable and independent 32 bit RISC processors
  - 3kB program RAM per processor
  - 3kB data RAM per processor
  - Clock frequency of 160 MHz
- Independent processor input interfaces
  - Internal result registers of DSADC
  - Internal auxiliary result registers of DSADC
  - Internal result registers of EXMOD
  - Carrier magnitude value (CARMAG) of DSADC carrier generator
  - Internal result register 0-5 of TMADC instances
  - 4 general purpose registers
  - Write access for program and data RAM through FPI bus system
- Output interfaces per processor
  - Main result path (RES0) with a FIFO of depth 4
  - 2 auxiliary result registers
  - Read access for data RAM through FPI bus system
- Debug support
  - APB interface
  - Input monitor and status registers
  - Suspend mode
- Time-stamp generation
  - Adjustable resolution
  - Time-stamp information directly accessible from main result register (RES0)
- Automatic limit checking
  - Result comparison with configurable upper and lower boundary value
  - Out of or within range indicator
  - Hysteresis mode

- Hardware triggers with configurable delay
  - Service request gate
  - Boundary update
  - Time-stamp generation
  - Software control
- Event signaling (service request generation)
  - Result generation (in FIFO mode configurable according to FIFO fill level)
  - Time-stamp generation
  - Limit checking conditions (signal within or out of range)
  - Processor status (processor wake-up error, system halted)

45.5.2 Functional overview

CDSP is connected to DSADC, EXMOD, TMADC, CARMAG and General Purpose (GP) registers to process the results coming from various sources.

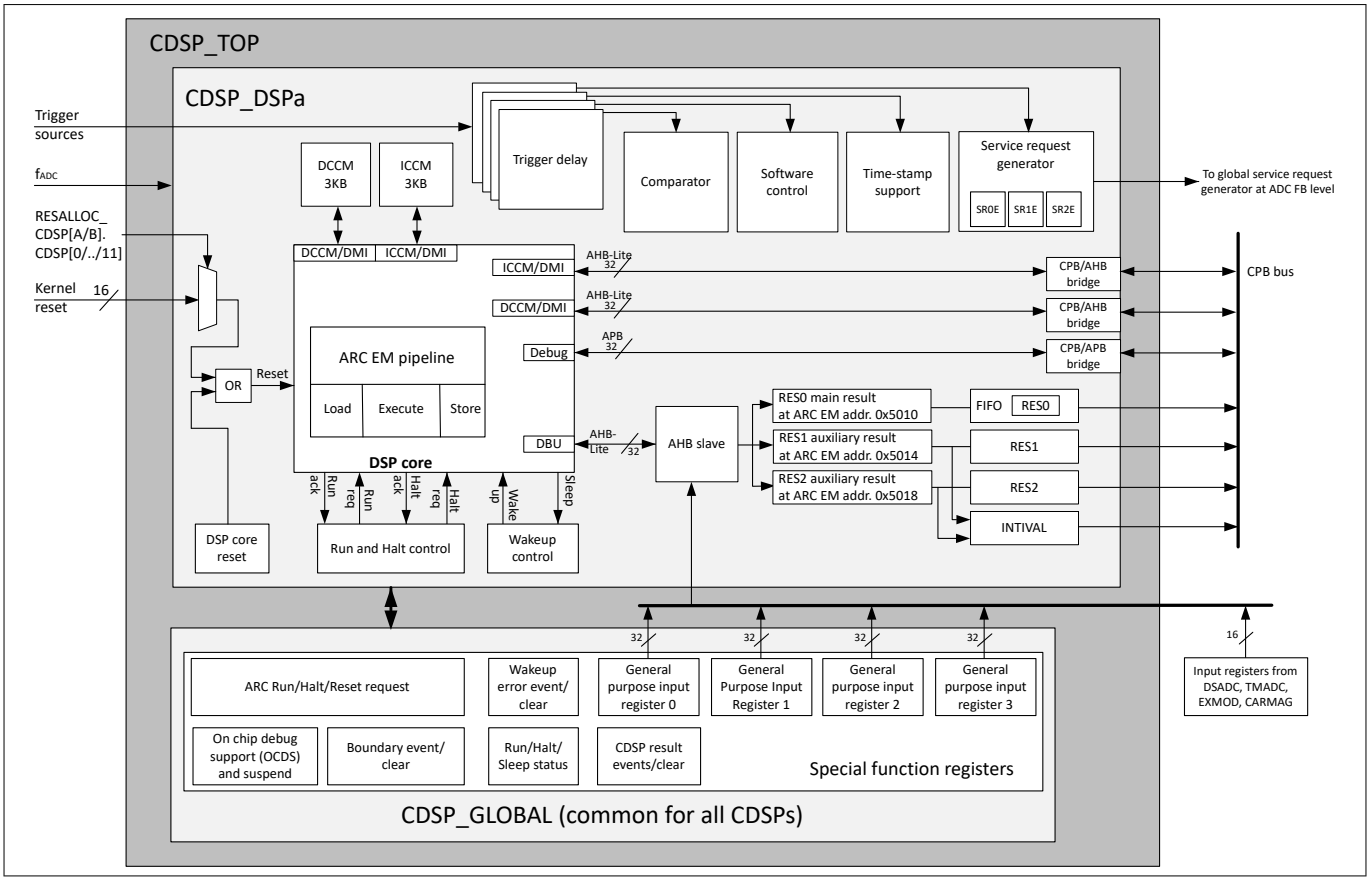


Figure 1225 CDSP architecture

CDSP operates with clock frequency f_{ADC}. The DSP core on CDSP is a three stage pipeline processor core. CDSP has two memories:

- Instruction memory, called ICCM, of 3 KB
- Data memory, called DCCM, of 3 KB

External masters can access these two memories using DSP slave ports:

- ICCM-DMI
- DCCM-DMI

DSP core reads the data from input registers using the Data Bus Unit (DBU) master port and the connected AHB slave. After processing the data, DSP core can write the output data through the DBU master interface into the

result registers of CDSP. Only DBU single word accesses are supported by CDSP AHB slave. Service request generation is combined only with the main result path RES0.

CDSP has different capabilities of handling the output result. The CDSP provides the option to generate time-stamps. You can access the time-stamp information through result registers, if configured. Results on the main result path can optionally be queued in a FIFO. CDSP can generate service requests on events such as the time-stamp information, the boundary flag event, or when the result is available in the result register. A comparator supports limit checking, for example for over-current detection. Two limit values can be defined to restrict the generation of service requests to result values within a configurable range.

45.5.3 Functional description

The CDSP hardware consists of circuitry for data processing and control logic.

The functionality of the building blocks is summarized in the following sections. An example which describes the procedure for starting up the CDSP is described in section CDSP start-up.

45.5.3.1 Central functions

CDSP_GLOBAL implements a set of central control and status registers of which the majority implements instance-specific bits with equal functionality. With this register layout the following basic functions can be triggered in different CDSP instances (CDSP_DSPa) simultaneously:

- Run, halt and reset request generation
- Wake-up, boundary or result event and service request generation
- Clearing of event flags

Corresponding status information is organized either in the same (CDSP_OCSCDSPA/CDSP_OCSCDSPB) or separate registers. Read and write accesses to the bits of these registers are controlled by the APUs which are assigned to CDSP_DSPa.

Exceptions exist for the suspend control bit fields CDSP_OCSCDSPA.SUSM, CDSP_OCSCDSPA.SUS_P, CDSP_OCSCDSPB.SUSM, CDSP_OCSCDSPB.SUS_P and the general purpose registers CDSP_GPx. These bit fields and registers are shared by multiple CDSP instances and are controlled only by the APU which is assigned to CDSP_GLOBAL.

CDSP_OCSCDSPA.SUSM and CDSP_OCSCDSPA.SUSP are shared by the CDSP instances CDSP_DSPa (a= 0-8), CDSP_OCSCDSPB.SUSM and CDSP_OCSCDSPB.SUSP are shared by the CDSP instances CDSP_DSPa (a= 9-17) and the CDSP_GPx registers are shared by all CDSP instances. An overview of the central control and status registers including implemented access protection is given in the table below.

Table 2391 Global control and status registers

Register	Function	Access protection	Details in section
CDSP_GLOBRD	DSP core reset requests	Individual CDSP (Resource ID 39-56)	Clock and reset
CDSP_GLOBRCD	Run requests	Individual CDSP (Resource ID 39-56)	Run and halt
CDSP_GLOBRSD	Run status	Individual CDSP (Resource ID 39-56)	Run and Halt
CDSP_GLOBHCD	Halt requests	Individual CDSP (Resource ID 39-56)	Run and halt
CDSP_GLOBHSTD	Halt status	Individual CDSP (Resource ID 39-56)	Run and halt

(table continues...)

Table 2391 (continued) Global control and status registers

Register	Function	Access protection	Details in section
CDSP_SLPST	Sleep status	Individual CDSP (Resource ID 39-56)	Sleep and wake-up
CDSP_RESEVy	Result event flag	Individual CDSP (Resource ID 39-56)	Result handling
CDSP_RESEVCLRy	Result event flag clearing	Individual CDSP (Resource ID 39-56)	Result handling
CDSP_GPx	General purpose input	CDSP global (Resource ID 38)	Input and output data interface
CDSP_WUERREV	Wake-up error event flags	Individual CDSP (Resource ID 39-56)	Sleep and wake-up
CDSP_WUERRCL	Wake-up error event clearing	Individual CDSP (Resource ID 39-56)	Sleep and wake-up
CDSP_BNDCF, CDSP_BNDCF1, CDSP_BNDCF2	Boundary event flags	Individual CDSP (Resource ID 39-56)	Limit checking
CDSP_BNDCFCL, CDSP_BNDCFCL1, CDSP_BNDCFCL2	Boundary event flag clearing	Individual CDSP (Resource ID 39-56)	Limit checking
CDSP_OCSCDSPA	OCDS control and status	Individual CDSP (Resource ID 39-56)	Debug
CDSP_OCSCDSPB	OCDS control and status	Individual CDSP (Resource ID 39-56)	Debug

***Note**: Central functions which are triggered in two or more CDSP instances simultaneously, will be executed simultaneously. However, corresponding status information can arrive in the status register with a synchronization jitter of 1 SPB clock cycle.*

***Note**: The event status flags of a disabled CDSP(i) are cleared, when CDSP_GLOBRCD.CHRUND(i) is set.*

Writing to registers CDSP_BNDCF, CDSP_RESEV and CDSP_WUERREV is enabled as debug option. It allows to check if the related service requests are generated for CDSP channels.

For further diagnostic and debugging, please use the global OCDS control and status registers CDSP_OCSCDSPA and CDSP_OCSCDSPB per CDSP. More details are available in the ADC Debug chapter.

45.5.3.2 Input and output data interface

Each CDSP can access input data from DSADC, EXMOD, TMADC, general purpose registers (CDSP_GPx) and the carrier magnitude register of DSADC using DSP software. Results are stored in result registers CDSP_DSPa_RESy. In addition DCCM and ICCM are address mapped to the MCU-level. They can be accessed directly through the peripheral bus (CPB, COMPB or SPB depending on variant). DSADC/EXMOD/CARMAG input registers are not available for all variants.

Input data path

45 Analog-to-Digital Converter (ADC)

The figure below shows the structure of the input data path to the DSP core. The data assembler combines the results coming from different sources with the control signals (WU, SWEN, SWINIT). This information is mapped to the internal DSP core address space.

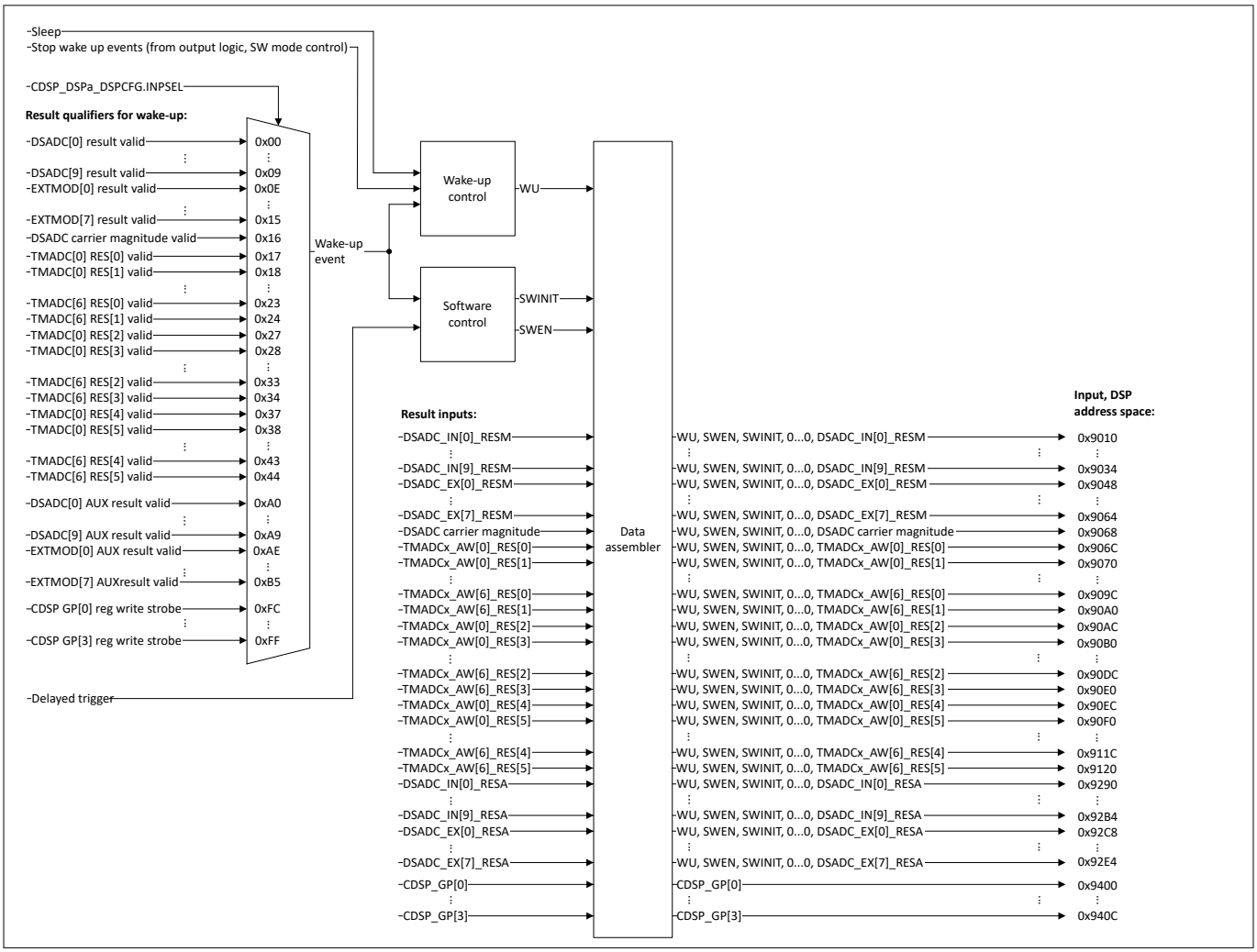


Figure 1226 Input data path

***Note**: The diagram shows the maximum configuration with respect to the data sources. If a data source is unavailable in a certain derivative, the corresponding connections will be tied to 0.*

***Note**: DSP core software access to input registers is to 32 bit.*

The control signals added to the result information are:

- WU, wake-up request generated out of wake-up event from selected input register, automatically cleared after software enable
- SWEN, enable signal for DSP software
- SWINIT, initialization signal for DSP software

The general purpose input data registers CDSP_GPx can be used to process data originating from generic sources.

When the DSP software performs a read operation from an input address, multiplexer operation between CDSP and desired input registers is done by hardware (AHB slave address decoder). In the system address map every CDSP_DSPa instance is mapped to a separate region. The DSP core address maps are equal but separated. DSP software uses the module addresses according to the table below.

Table 2392 Address mapping of input and output registers from DSP perspective

Name	System address (CPB, COMPB, SPB)	DSP core address	Description
ICCM	0xF50C0000 + a*2000$_H$	0x0000-0x0C00	DSP Instruction close coupled memory
DCCM	0xF50C1000 + a*2000$_H$	0x8000- 0x8C00	DSP data close coupled memory (input/output)
DSADCRESx, (x= 0-9)	In Register chapter	(0x9010 + 4*x)$_H$	Main result register of DSADC (input for DSP)
EXMODRESx, (x= 0-7)	In Register chapter	(0x9048 + 4*x)$_H$	Main result register of EXMOD (input for DSP)
DSADC carrier magnitude	In Register chapter	(0x9068)$_H$	Carrier signal magnitude from DSADC (input for DSP)
TMADCx_AW0_RESv, (v= 0-1), (x= 0-6)	In Register chapter	(0x906C + (4*(2*x+v)))$_H$	RES0 and RES1 of TMADC (input for DSP)
TMADCx_AW0_RESv, (v= 2-3), (x= 0-6)	In Register chapter	(0x90AC + (4*(2*x+v)))$_H$	RES2 and RES3 of TMADC (input for DSP)
TMADCx_AW0_RESv, (v= 4-5), (x= 0-6)	In Register chapter	(0x90EC + (4*(2*x+v)))$_H$	RES4 and RES5 of TMADC (input for DSP)
DSADCAUXx, (x= 0-9)	In Register chapter	(0x9290 + 4*x)$_H$	Auxiliary result register of DSADC (input for DSP)
EXMODAUXx, (x= 0-7)	In Register chapter	(0x92C8 + 4*x)$_H$	Auxiliary result register of EXMOD (input for DSP)
CDSP_GPx, x=[0:3]	In Register chapter	(0x9400 + 4*x)$_H$	General purpose register (input for DSP)
CDSP_DSPa_RESy(a=0-17;y=0-2)	In Register chapter	(0x5010 + 4*y)$_H$	Result register of CDSP (output of CDSP)

Following points shall be considered for the input data interface:

- Only the lower 16 bits of all input registers are used as data for post-processing by DSP software
- Upper 16 bits contain control information which is required by DSP software to support specific modes
- When CDSP_GPx registers are used as input for Infineon DSP software, the register layout shall follow the DSADC/TMADC input layout
- If a CDSP_GPx register is selected as wake-up source, it shall just get updated when the CDSP is in sleep mode. Otherwise a wake-up error is issued
- CDSP results are available in result registers RES0, RES1 and RES2. RES0 is a standard 32 bit main result register used for all software functions.

Input monitoring

The CDSP_DSPa_INPUTMON status register monitors the input data and allows to check the status of control signals WU, SWEN and SWINIT.

The input source to monitor in CDSP_DSPa_INPUTMON.MONDATA can be configured by CDSP_DSPa_DSPCFG.HSIMON settings according to following table.

Data selection for input monitoring

Input register	Index	HSIMON
DSADC_INx_RESM	x= 0-9	00_h - 09_h
Reserved		$0A_h$ - $0D_h$
DSADC_EXz_RESM	z= 0-7	$0E_h$ - 15_h
DSADC carrier magnitude		16_h
TMADCx_AW0_RESv	x= 0-6, v= 0-1	17_h - 24_h
Reserved		25_h - 26_h
TMADCx_AW0_RESv	x= 0-6, v= 2-3	27_h - 34_h
Reserved		35_h - 36_h
TMADCx_AW0_RESv	x= 0-6, v= 4-5	37_h - 44_h
Reserved		45_h - $9F_h$
DSADC_INx_RESA	x= 0-9	$A0_h$ - $A9_h$
Reserved		AA_h - AD_h
DSADC_EXz_RESA	z= 0-7	AE_h - $B5_h$
Reserved		$B6_h$ - FB_h
CDSP_GPx	x= 0-3	FC_h - FF_h

***Note**: In case a source is selected, that does not exist in a specific variant 0_h is read all the time.*

The next table shows the content monitored in CDSP_DSPa_INPUTMON.MONDATA depending on the selected type of source.

Table 2393 CDSP_DSPa_INPUTMON.MONDATA interpretation dependent on input source

CDSP_DSPa_INPUTMON	31	30	29	28:16	15:0
CDSP_GPx	General purpose register data [31:16]				General purpose register data [15:0]
DSADC/EXMOD	WU	SWEN	SWINIT	0	16 Bit DSADC/EXMOD result, 2's complement
TMADC	WU	SWEN	SWINIT	0	16 Bit TMADC result, 2's complement
CARMAG	WU	SWEN	SWINIT	0	5 Bit carrier magnitude, sign extended, 2's complement

Output data path

Once the processing of the input data is done on DSP core, the results are stored in result registers. The usage of the result registers depends on the CDSP software. The following scenarios are supported by the CDSP hardware and can be configured in CDSP_DSPa_SWCFG.SWMODE:

- 00_B: Software results are transmitted to the result registers as generated. INTIVAL updates are blocked. The running DSP core is sensitive to wake-up events all the time, if not in halt state. Used for example for filter chains without integrator

- 01_B: Software results are transmitted to the result registers as generated upon write access to RES1. INTIVAL register updates together with RES1 and RES2. The running DSP is sensitive to wake-up events all the time. Used for example for filter chains with integrator
- 10_B: Software results are transmitted to result registers as generated. INTIVAL updates are blocked. The running DSP core is sensitive to wake-up events until write access to RES2. Used for example for FFT

The figure below shows the structure of the output data path.

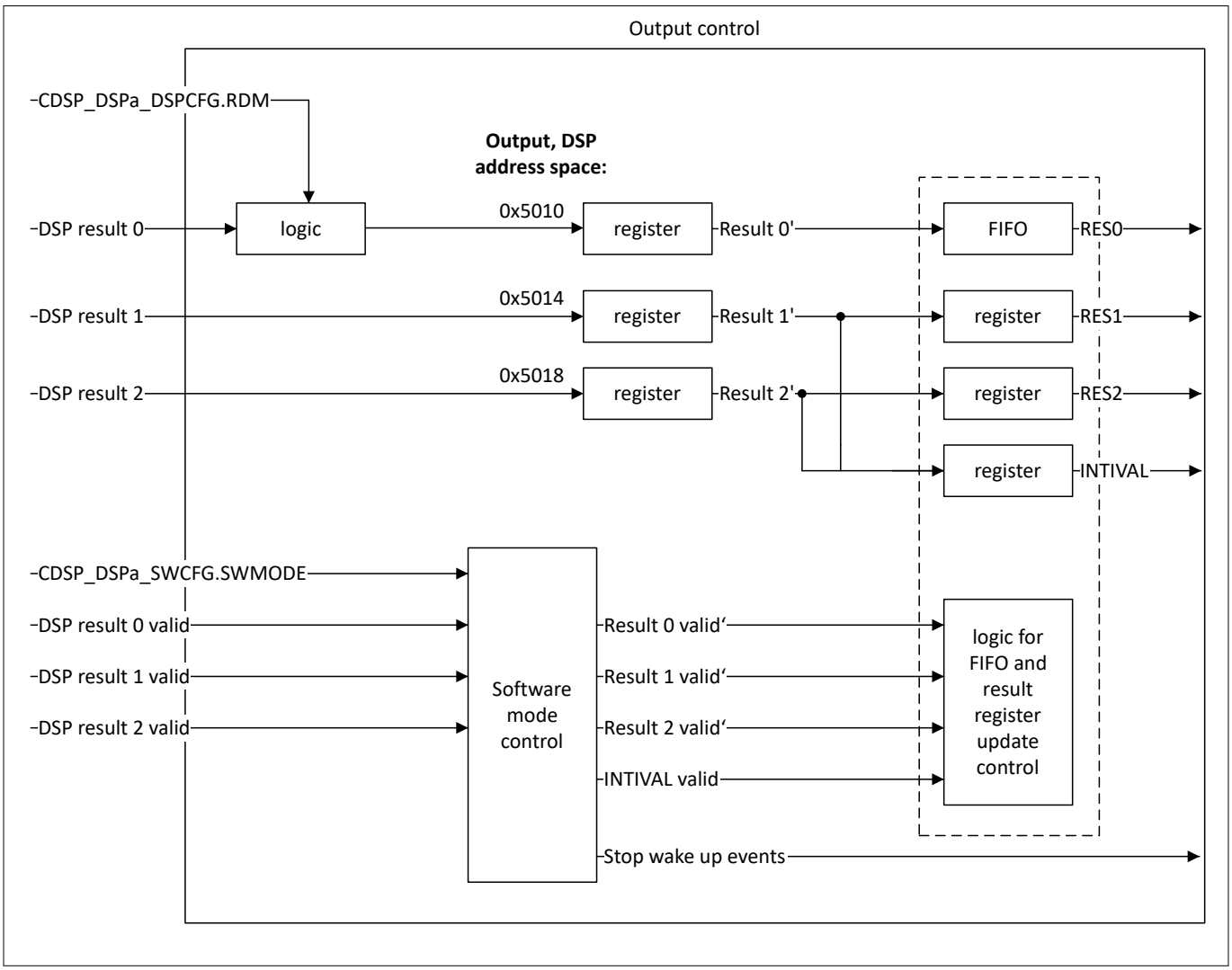


Figure 1227 Output data path

***Note**: DSP core software access to output registers is to 32 bit.*

FIFO mode is only supported through register RES0. Check section "Result Handling" for more details.

Related information

Result handling on page 8380
ARC DSP core debug features on page 8544

45.5.3.3 Clock and reset

The CDSP clock gets enabled by the clock control register CLKEN_CDSP.CDSPEN(a). In case a particular DSP core is used, the global clock (CLKEN_CDSP.CDSPGL) has to be enabled as well. Additional prerequisite is that CLC.DISR was configured to 1_b

The reset for the DSP core reset input Rst_a is dependent on CDSP_GLOBRD.DSPRST and APU controlled Kernel reset. When any of these resets is triggered, the DSP core is reset which sets the DSP core into the halt mode. Below is the block diagram explaining the clock and reset functionality.

The Debug reset resets the registers CDSP_OCSCDSPA and CDSP_OCSCDSPB.

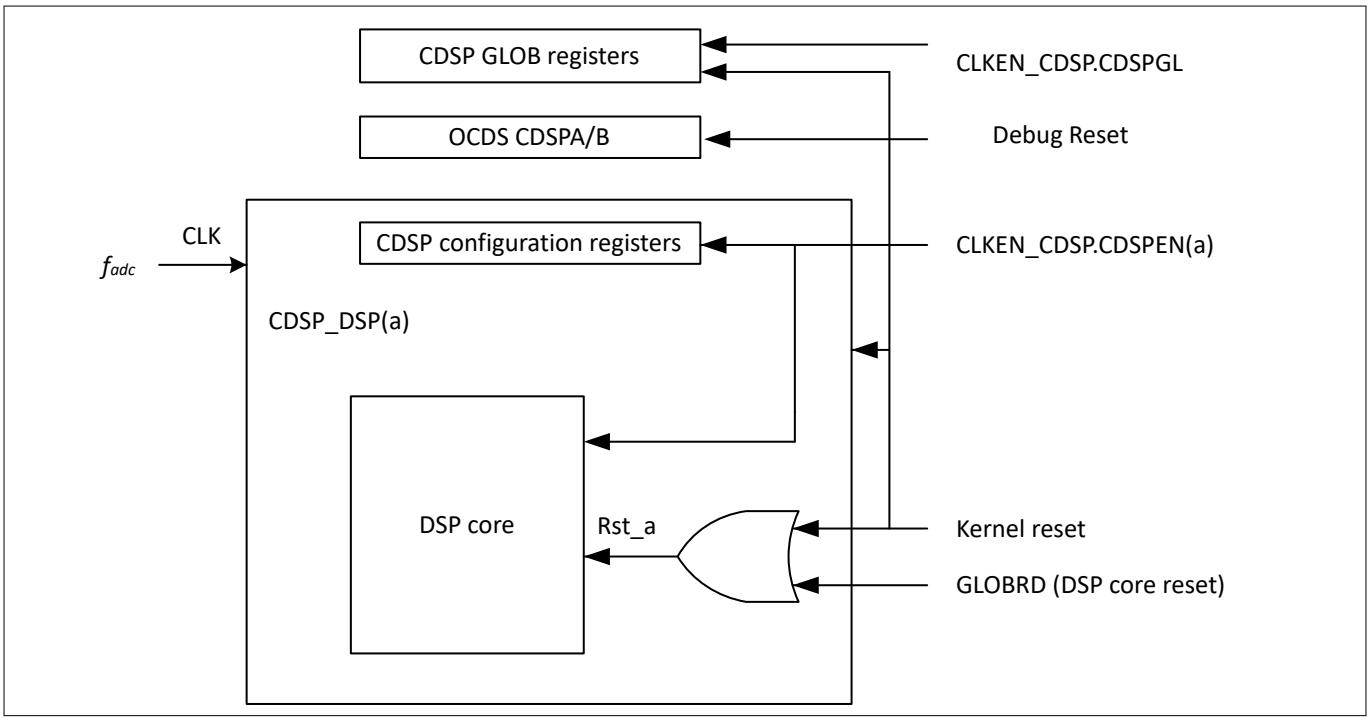


Figure 1228 CDSP clock and reset

45.5.3.4 Run and halt

By default a CDSP instance (a) is disabled after reset. It can be enabled by writing 1_b to CDSP_GLOBRCD.CHRUND(a). This action triggers a run request for the DSP core of CDSP instance (a). When this DSP core is running, the corresponding run status bit CDSP_GLOBRSD(a) will be set to 1_b, the halt status bit CDSP_GLOBHSTD(a) will be set to 0_b and all other control units of the CDSP instance (a) will be re-initialized. In this state the DSP core is ready to process data based on wake-up events as described in section 'Sleep and wake-up'.

Apart from any higher class reset (e.g. module reset or application reset), a running CDSP instance can be disabled by a halt request, a DSP reset, on demand of the CDSP software or by an uncorrectable error on the data path. A halt request for CDSP instance (a) can be issued by writing 1_b to CDSP_GLOBHCD.CHHLTD(a) and a DSP reset by writing 1_b to CDSP_GLOBRD.DSPRST(a). A DSP software call of a halt routine or an uncorrectable error on the data path will be handled automatically by the DSP core.

When the DSP core is halted, the corresponding run status bit CDSP_GLOBRSD.DSPRST(a) will be set to 0_b, the halt status bit CDSP_GLOBHSTD.DSPHST(a) will be set to 1_b. In this state wake-up events will be ignored.

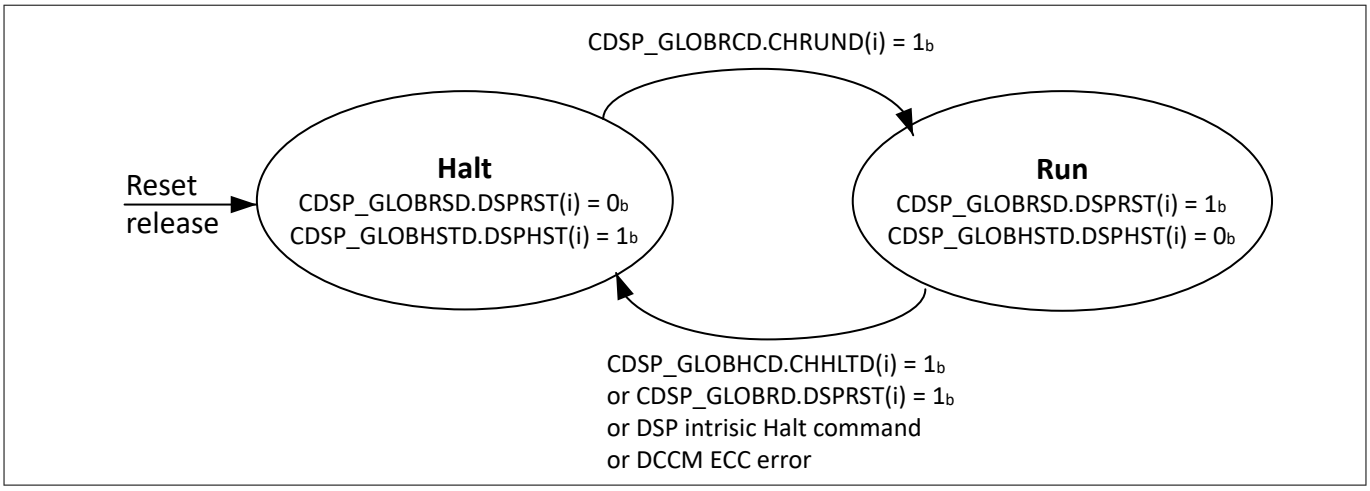


Figure 1229 Run and Halt state diagram

Due to synchronization and implementation delays the processing of run and halt requests can take up to 12 f_{ADC} + 4 peripheral bus (SPB/COMPB/CPB depending on device variant) clock cycles. The new run/halt status will be available at the latest 3 peripheral bus (SPB) clock cycles after that. This is the worst case scenario where either the enabling or the disabling of at least 2 CDSP instances happens successively within two peripheral bus (SPB/COMPB/CPB depending on device variant) clock cycles.

45.5.3.5 Sleep and wake-up

The CDSP supports wake-up and sleep based data processing. Sleep and wake status of DSP core is used to control program execution dependent on input data availability.

To this end a CDSP instance (a) must be enabled first as described in chapter Run and Halt. Afterwards the running DSP core (a) alternates between an active (awake) and an inactive (sleep) phase. During the active phase it executes the DSP software and during the inactive phase it is idle. The active phase is initiated by a hardware wake-up event and the inactive phase is entered when the DSP software executes a sleep instruction. The actual phase of DSP core(a) can be read from CDSP_SLPST.DSPSLP(a).

Wake-up events are generated by each data source whenever it stores new data. A CDSP instance (a) can be configured in CDSP_DSPa_DSPCFG.INPSEL, to listen to the wake-up events of exactly one data source. An overview of the configuration options is given in the table below.

Input register	Index	INPSEL bits
DSADC_INx_RESM	x= 0-9	00_h - 09_h
Reserved		$0A_h$ - $0D_h$
DSADC_EXz_RESM	z= 0-7	$0E_h$ - 15_h
DSADC carrier magnitude		16_h
TMADCx_AW0_RESv	x= 0-6, v= 0-1	17_h - 24_h
Reserved		25_h - 26_h
TMADCx_AW0_RESv	x= 0-6, v= 2-3	27_h - 34_h
Reserved		35_h - 36_h
TMADCx_AW0_RESv	x= 0-6, v= 4-5	37_h - 44_h
Reserved		45_h - $9F_h$
DSADC_INx_RESA	x= 0-9	$A0_h$ - $A9_h$
Reserved		AA_h - AD_h
DSADC_EXz_RESA	z= 0-7	AE_h - $B5_h$

Input register	Index	INPSEL bits
Reserved		$B6_h$ - FB_h
CDSP_GPx	x= 0-3	FC_h - FF_h

***Note**: DSP core can take up to 6 f_{ADC} clock cycles after the execution of sleep instruction to go into sleep state. DSP core can take up to 6 f_{ADC} clock cycles after receiving wake signal to completely wake up.*

Figure below explains sleep and wake up control for CDSP.

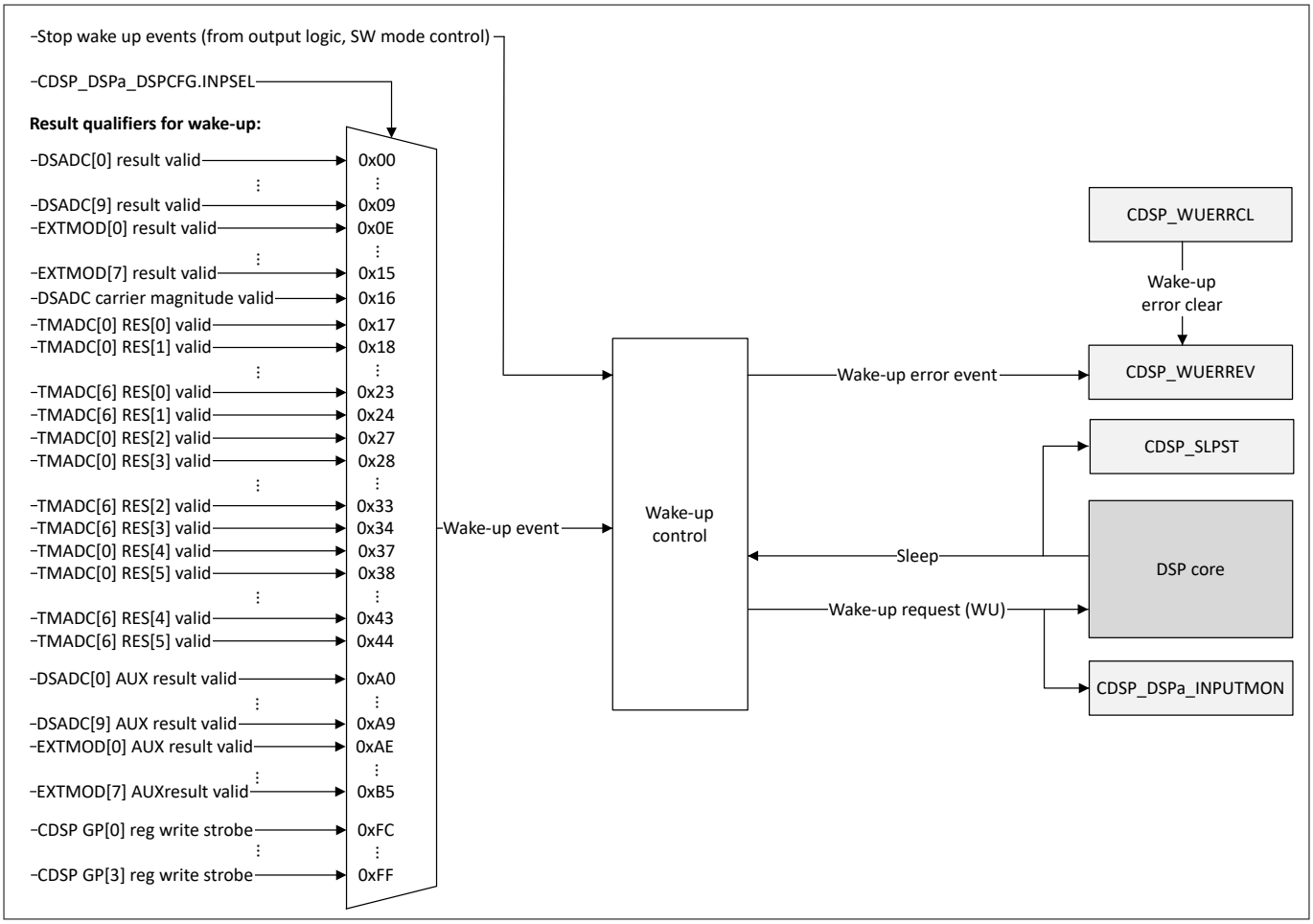


Figure 1230 Sleep and wake-up of CDSP

For proper data processing the overall system must be configured in a way that wake-up events do not arrive at DSP core (a) while it is busy (CDSP_GLOBRSD.DSPRST(a)= 1_b, CDSP_GLOBHSTD.DSPHST(a)= 0_b, CDSP_SLPST.DSPSLP(a)= 0_b). Else CDSP instance (a) will generate wake-up errors which are stored in the event flag register CDSP_WUERREV.WUERREV(a).

A wake-up error indicates that the data processing became unpredictable. The stored wake-up error flag can be cleared by writing 1_b to CDSP_WUERRCL.WUERRCL(a). Enabling a disabled DSP core with writing 1_b to instance specific bit in CDSP_GLOBRCD.CHRUND will also clear a wake-up error. Wake-up errors can cause assertion of service request line SR1 if configured by the settings CDSP_DSPa_SRCFG.SR1E (setting 10_b or 11_b).

A halt event for CDSP instance (a) freezes the current state of DSP core (a) and CDSP instance (a) will block further wake-up events. It is mandatory to reset a DSP core before reactivation.

With a reset a DSP core will be set in a state where it is halted but awake. This setting mainly serves the purpose of polling based data processing which is not supported by the CDSP hardware. But it can be usedfor the execution of a wake-up event independent startup routine when a CDSP instance (a) is started through CDSP_GLOBRCD.CHRUND for the first time after reset. In this way initialization and data processing phase can

be separated better. The duration of the startup routine depends on the CDSP software. At the end of this routine a sleep instruction must be executed.

The relations between run, halt, wake-up and sleep state of an DSP core (a) as well as the possible wake-up error conditions are visualized in figure below. Dashed lines at run requests and halt events highlight exceptional cases for these events. The normal process flow is described by the solid run request and halt event lines.

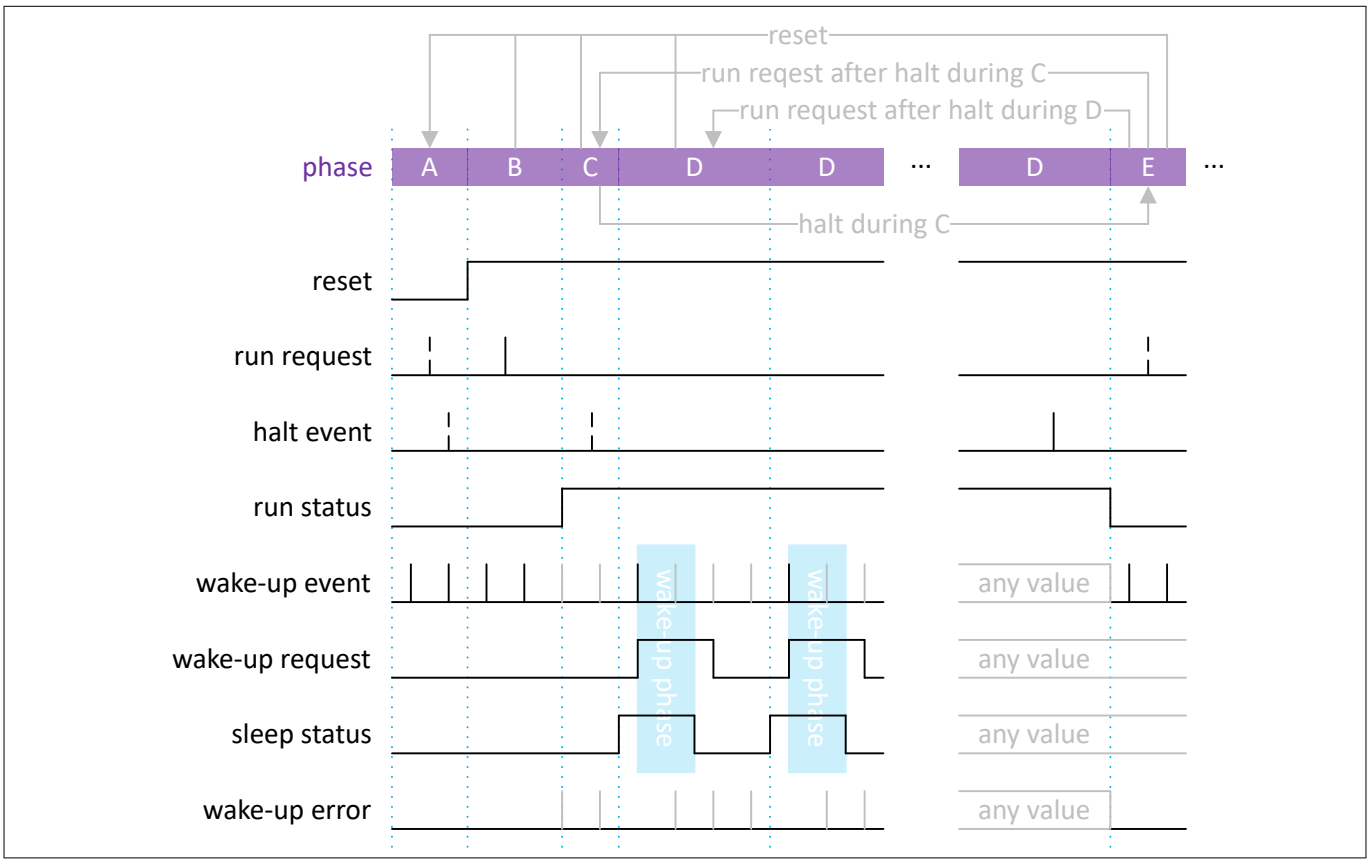


Figure 1231 Sleep and wake-up timing

Following the description of the different phases.

A) Reset

- During an active reset the DSP core will be set in a state where it is halted and awake
- A possible new run request (dashed line at run request) will be queued by the CDSP hardware and executed directly after the reset. An additional halt event (dashed line at halt event) cancels the run request again
- Since the DSP core is halted while the reset is active, wake-up events will be ignored by the CDSP hardware

B) Halt after reset

- After reset the DSP core preserves its state until it receives a run request. The run request will be processed as described in section Run and halt

C) Run with wake-up independent startup

- When the DSP core acknowledges the run request from any of the previous phases, it switches its run status to 1_b and executes a wake-up event independent startup routine (Infineon CDSP software uses this for initialization). At the end of this routine the CDSP software sets the DSP core into sleep state
- Since the DSP core is running during this phase, it is sensitive to wake-up events. For the reason that it is also awake and busy with the execution of the startup routine, all wake-up events of the selected wake-up source will cause wake-up errors (highlighted by grey wake-up events and grey wake-up errors). The wake-up errors can be avoided by selecting a reserved CDSP_DSPa_DSPCFG.INPSEL configuration after reset as long as CDSP_SLPST.DSPSLP(a) is 0_b
- If a halt event occurs during this phase (dashed line at halt event), the DSP core will stop its current activity and switch to phase E

D) Run with data processing

- At the beginning of this phase the DSP core is always running and sleeping and ready for wake-up event based data processing
- The data processing is initiated by the first wake-up event which the DSP core receives during sleep state and it is finished when the CDSP software has executed a sleep instruction
- Any wake-up event while the DSP core is about to wake-up (highlighted as wake-up phase) or busy with the execution of the CDSP software (sleep status is 0_b) will cause a wake-up error because of potential data loss (highlighted by grey wake-up events and grey wake-up errors)
- If a halt event occurs during this phase, the DSP core will stop its current activity and switch to phase E

E) Halt after run

- Whenever a halt event occurs while the DSP core is running, the DSP core freezes its current state and switches to this phase
- The wake-up request state and sleep status depends on the state in which the DSP core was at the halt event
- Wake-up events from the selected wake-up source will be ignored by the CDSP hardware because the DSP core is halted

***Note**: If CDSP_DSPa_SWCFG.SWMODE= 10_b, wake-up events of the selected wake-up source will be blocked during phase C or D as soon as the CDSP software writes to address 0x5018 (CDSP_DSPa_RES2). Infineon CDSP software uses phase D. Details about this functionality and how to release the wake-up event blocking again are described section Software control, Software mode 2*

45.5.3.6 Trigger handling

The CDSP can select between 64 trigger inputs from different hardware sources. They are used for software control, gating of service requests, time-stamp generation and update of compare values for limit checking.

The triggers received are synchronized to the f_{ADC} clock domain before reaching the CDSP. Triggers can be delayed individually in steps of 2 f_{ADC} clock cycles. The complete list can be found in the overarching ADC chapter in the topic 'Hardware triggers to ADC'.

Active edges of the incoming triggers are configured in the trigger mode registers CDSP_DSPa_TSTRCFG.TSTRM, CDSP_DSPa_SWTRCFG.SWTRM and CDSP_DSPa_BNDTRCFG.BNDTRM. The trigger based result event gating is controlled in CDSP_DSPa_SRCFG.SR0E.

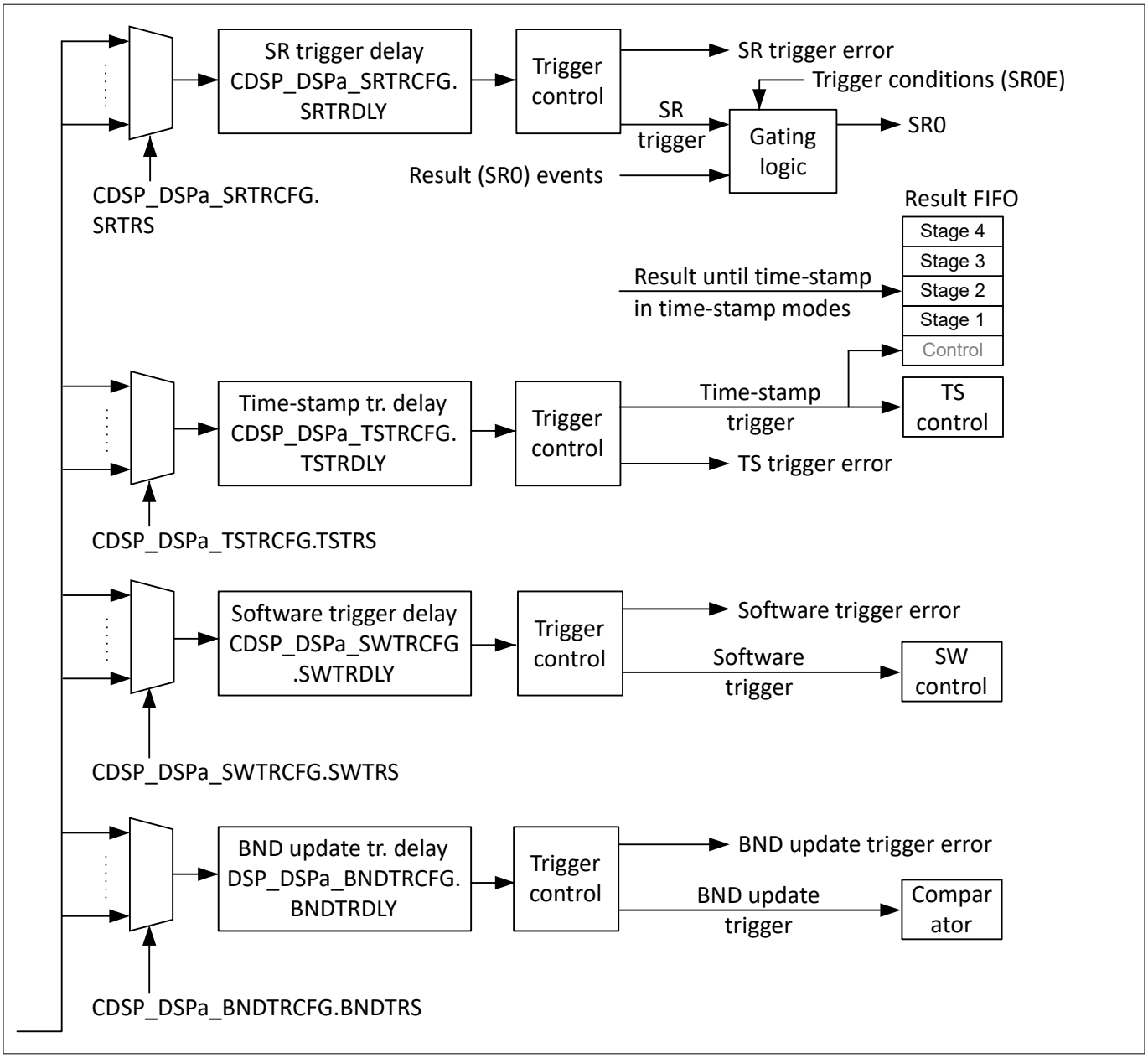


Figure 1232 Overview of trigger functionality

Trigger sources for the service request trigger are selected by CDSP_DSPa_SRTRCFG.SRTRS. The service request trigger is gating the SR0 result service request of the CDSP_DSPa. Trigger delay can get programmed in CDSP_DSPa_SRTRCFG.STRDLY. For the service request trigger there are individual trigger delay counters for positive and negative edge. The delay value is common for both edges. In case of a trigger error both trigger delay counters are reset.

The gating of the result service request can be enabled or disabled permanently. It can also be controlled by a high-active or low-active trigger (CDSP_DSPa_SRCFG.SR0E). If the Result FIFO is not in time-stamp mode, it will collect data only if service requests are enabled.

The time-stamp trigger source is selected with CDSP_DSPa_TSTRCFG.TSTRS. Trigger delay can get programmed in CDSP_DSPa_TSTRCFG.TSTRDLY. For the time-stamp trigger there are individual trigger delay counters for positive and negative edge. The delay value is common for both edges. The time-stamp trigger controls the time-stamp generation. If the FIFO is in time-stamp mode it will also control the FIFO operation.

The boundary update trigger source is selected with CDSP_DSPa_BNDTRCFG.BNDTRS. Trigger delay can get programmed in CDSP_DSPa_BNDTRCFG.TSTRDLY. For the boundary update trigger there are individual trigger

delay counters for positive and negative edge. The delay value is common for both edges. The boundary update trigger controls the update of the compare values for limit checking.

The software trigger source is set with CDSP_DSPa_SWTRCFG.SWTRS. The trigger sources are hardware based and have to be assumed asynchronous to DSP Core software execution. The software trigger is translated into software control signals SWEN and SWINIT evaluated by DSP software. The hardware timed trigger has to be translated into the software context. A fixed relation between software triggers and wake-up and sleep state of the DSPs is not expected. The CDSP hardware does this alignment automatically. Trigger delay can get programmed in CDSP_DSPa_SWTRCFG.SWTRDLY.

Trigger errors occur when

- The configured delay value is greater than the time between two equal edges of the selected trigger. In order to avoid this error condition the setup must always satisfy the relation: delay value ≤ time between two equal edges of the selected trigger.
- A new delay configuration becomes effective and changes the order of delayed edges for the actual trigger event.

45.5.3.7 Time-stamp support

Some applications need to determine the result value at certain points of time in between two regular output values. The interpolation algorithm determines the position of the required point of time. This interpolation is supported by providing a time-stamp that marks the delay since the last scheduled output value. Depending on the input/output decimation this known fraction of an output cycle enables the interpolation.

The time-stamp information is generated by the time-stamp counter. This 16-bit counter is clocked with the prescaled f_{ADC} and configurable by CDSP_DSPa_DSPCFG.TSCLK. The current value can be read from register CDSP_DSPa_TSCNT. The clock divider ratio CDSP_DSPa_DSPCFG.TSCLK shall not be changed at run-time.

Time-stamp generation is gated by Time-stamp counter run control (CDSP_DSPa_DSPCFG.TSCRUN)

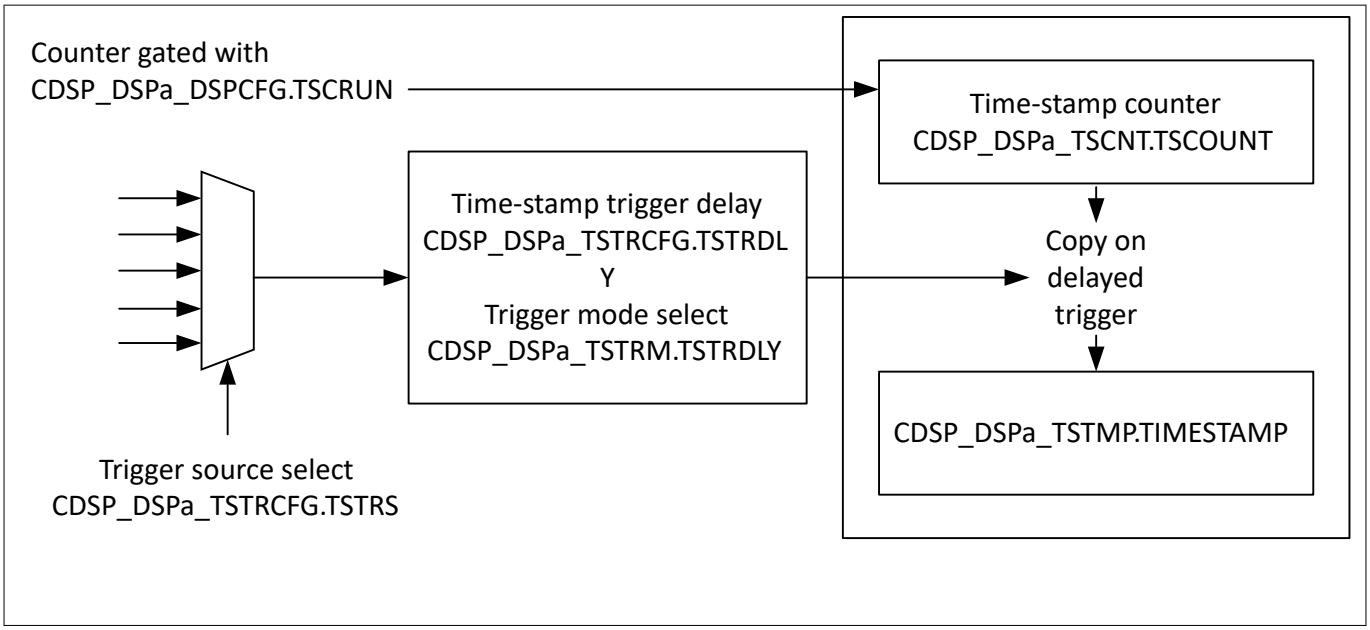


Figure 1233 Time-stamp generation

The time-stamp information is captured into the time-stamp register CDSP_DSPa_TSTMP upon a trigger event. For this purpose, the trigger signal is used, which is selected by bit-field TSTRS in register CDSP_DSPa_TSTRCFG . Bit-field CDSP_DSPa_TSTRCFG.TSTRM selects the edge(s) that capture(s) time-stamp information. With the bit field CDSP_DSPa_TSTRCFG.TSTRDELY the effective trigger edge can be optionally delayed.

In case CDSP_DSPa_SRCFG.SR1E is set to 1_h or 3_h a time-stamp trigger selected by bit-field CDSP_DSPa_SRTRCFG.SRTRS generates a service request.

The valid flag TSVAL is set when time-stamp information is stored and is cleared when the time-stamp is read from register CDSP_DSPa_TSTMP.TIMESTAMP. In addition the time-stamp counter is reset:

- If a disabled DSP core gets enabled by setting CDSP_GLOBRCD.CHRUND
- If CDSP_DSPa_DSPCFG.TSCRUN changes from 0_b to 1_b
- If the DSP core writes a new result to RES0

The time-stamp data is also available through the standard result register. More details can be found in chapter 'Result handling'.

Related information

Storage of processed data from DSP core on page 8380

45.5.3.8 Automatic reinitialization triggered by DSADC calibration

A DSP core can get reinitialized automatically when a DSADC channel starts calibration. This ensures that the start conditions for the next measurement are defined and parameters like the filter settling time remain predictable.

The filter chain of a DSADC is reset automatically by the DSADC calibration algorithm. A DSP core can select the calibration start of a specific DSADC to trigger a DSP reset. This is controlled in the bit field CDSP_DSPa_DSPCFG.INITSRC. The feature is switched off by default.

The DSP core reset triggered by DSADC calibration start does not affect the circuitry for storing run or halt requests. The last run or halt request will be executed. CDSP run status before the DSADC calibration start event is re-established. Result registers keep old measurement results until the next measurement starts.

The time-stamp control unit and the software management is reset in order to provide correct data and control signals when a new measurement starts following the DSADC calibration.

45.5.3.9 Result handling

The results from CDSP can be stored in result registers which can be retrieved by the CPU or by the DMA. One of these result registers (RES0) is combined with service request generation and can be connected additionally to a FIFO. It is also possible to leave results in the RAM so that the application software reads them directly from there.

Result handling comprises of the following functions:

- Storing CDSP output data
- Result service request generation and read sequencing

45.5.3.9.1 Storage of processed data from DSP core

The CDSP results can be stored in a FIFO-structure which is accessible through the result register CDSP_DSPa_RESy (y= 0). To this end CDSP_DSPa_DSPCFG.DRM must be configured with FIFO mode (DRM= 00_b or 10_b) and the CDSP software must write the result to address $0x5010_h$.

This enables the following features:

- Buffering reduces the result interrupt frequency depending on CDSP_DSPa_DSPCFG.SRLVL
- Two result values can be read by one single read access
- Two subsequent read accesses enable an efficient transfer of up to 4 result values
- The previous result value is still available to calculate interpolation values for time-stamp operation

- Time-stamp information can be accessed through the result register independent of FIFO mode
- One DMA channel can transfer all relevant data

***Note**:* *FIFO is supported by Result register CDSP_DSPa_RESy (y= 0), the other two result registers CDSP_DSPa_RESy (y= 1,2) can be used to retrieve results in case of special software functions, where multiple results are generated to reflect additional information. For example: filter chains with integrator. If CDSP generates only one output, then the content of CDSP_DSPa_RESy (y= 1,2) will not be changed.*

Result handling without FIFO

Depending on the single word read mode CDSP_DSPa_DSPCFG.DRM= 01_B the result values are stored in CDSP_DSPa_RESy (y= 0) directly. The result register returns the last result value if CDSP_DSPa_DSPCFG.TSM = 0_b. The high word can optionally be overwritten with the time-stamp, if TSM= 1_b. Result service requests are generated according to the CDSP_DSPa_SRCFG.SR0E configuration.

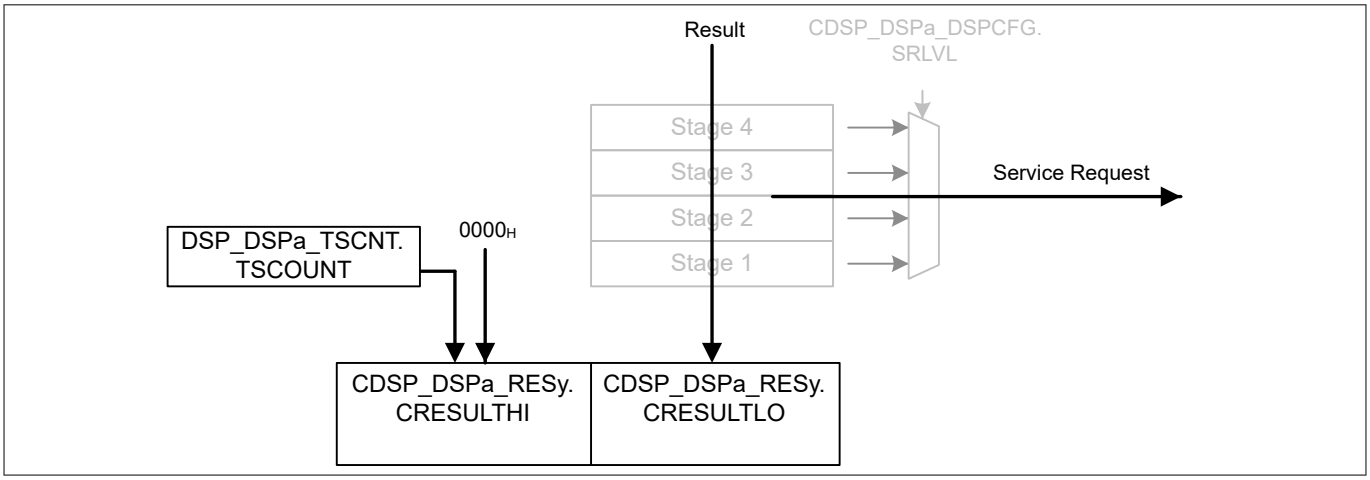


Figure 1234 Direct result storage

FIFO modes

In FIFO mode (CDSP_DSPa_DSPCFG.DRM= 00_B or CDSP_DSPa_DSPCFG.DRM= 10_B) the main filter chain results and optionally time-stamp information (CDSP_DSPa_DSPCFG.TSM= 1_b) will be stored in a FIFO. The FIFO content can be accessed via the result register CDSP_DSPa_RESy (y= 0). If the FIFO is full, additional results will be discarded and DSPST.WRERR status bit is asserted. The user should read results only based on service requests and before the next result is stored, especially after initialization.

The FIFO will be initialized (emptied) when the user:

- Enables- the DSP core by setting CDSP_GLOBRCD.CHRUND
- Flushes the FIFO by setting CDSP_DSPa_DSPCFG.FIFL
- Activates the service request generation (in modes without time-stamp)
- At the disabling edge of the time-stamp trigger (in time-stamp modes)

The FIFO can be configured in four different ways:

1. Single data read mode without time-stamp

In this mode the FIFO is queuing results in incoming order. Results will be stored in this mode only if service requests are enabled. The user has read access to the FIFO stage 1. Stage 1 is assigned to CDSP_DSPa_RESy.CRESULTLO (y= 0) and the higher 16 bits are sign extended.

Necessary setup:

- CDSP_DSPa_DSPCFG.DRM= 00_b
- CDSP_DSPa_DSPCFG.TSM= 0_b

- CDSP_DSPa_DSPCFG.SRLVL=any
- CDSP_DSPa_SRCFG.SR0E={01_b} + trigger or {10_b} + trigger or {11_b}

Figure below illustrates the data flow.

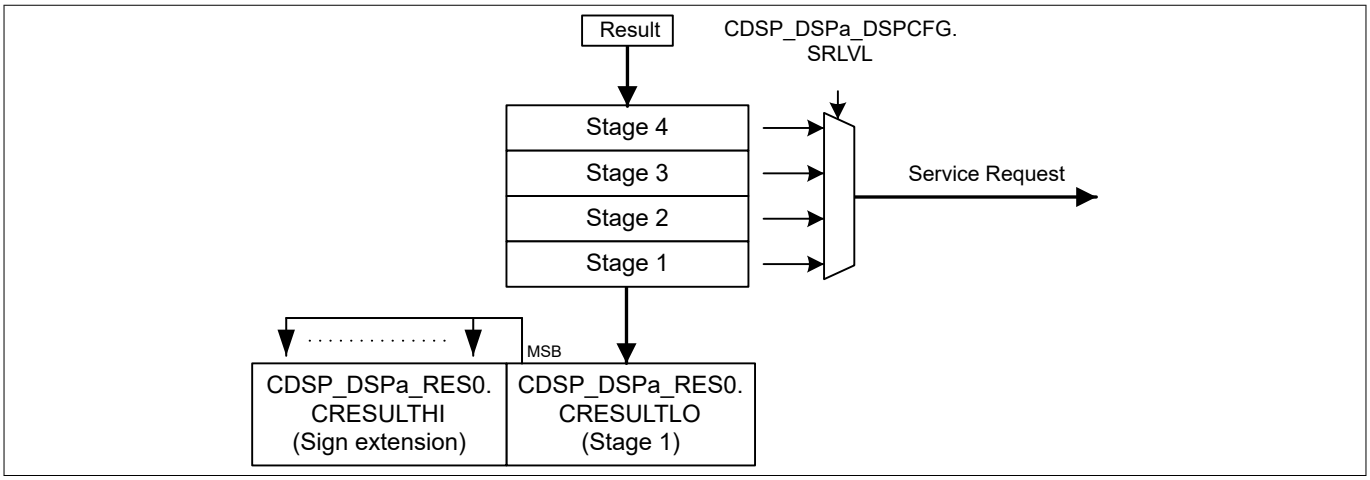


Figure 1235 Single data read mode without time-stamp

2. Single data read mode with time-stamp

In this mode the user will read the following sequence of values from the FIFO when the time-stamp was generated by the selected trigger edge (make sure that CDSP_DSPa_DSPCFG.TSCRUN=1_b in order to enable the time-stamp counter and CDSP_DSPa_TSTRCFG.TSTRM= {01_B} + trigger or {10_B} + trigger for time-stamp trigger control):

1. Time-stamp (this time-stamp will also be stored in CDSP_DSPa_TSTMP.TIMESTAMP)
2. Result before time-stamp or result at time-stamp when the time-stamp is generated at the same point in time when a new result is generated
3. 1^{st} result after time-stamp
4. 2^{nd} result after time-stamp
5. ….. n^{th} result after time-stamp

The purpose of this fixed order is that the user can differentiate between results and time-stamp because the time-stamp does not have a unique identifier. The FIFO switches between two sub-modes according to (CDSP_DSPa_TSTRCFG.TSTRM):

1. Waiting for time-stamp
   - Entered after initialization
   - Service requests will not be generated
   - User has no read access to the FIFO (read accesses will result in a read error indicated by CDSP_DSPa_DSPST.RDERR)
2. Storing results (similar to single data read mode without time-stamp)
   - Entered after time-stamp trigger
   - Result service requests on SR0 will be generated if configured by CDSP_DSPa_SRCFG.SR0E, time-stamp service requests will be generated on SR1, if configured by CDSP_DSPa_SRCFG.SR1E.
   - User has read access

After the initialization the user should wait for a service request (generated after configured time-stamp trigger edge) before reading CDSP_DSPa_RESy (y= 0).

Furthermore, it should be considered that the FIFO reacts only on the edges of the trigger signal. If for example the time-stamp trigger is high before and after setting CDSP_GLOBRCD.CHRUND= 1_b and a time-stamp shall be generated upon the rising edge of the time-stamp trigger (CDSP_DSPa_TSTRCFG.TSTRM= 10_B), then the FIFO will stay in sub-mode 'waiting for time-stamp' until the rising edge of the time-stamp trigger is generated. The FIFO will be initialized before, because the time-stamp trigger must be inactive before activating it again and therefore, a disabling time-stamp trigger edge will be generated.

Necessary setup:

- CDSP_DSPa_DSPCFG.DRM= 00_b
- CDSP_DSPa_DSPCFG.TSM= 1_b
- CDSP_DSPa_DSPCFG.TSCRUN= 1_b
- CDSP_DSPa_TSTRCFG.TSTRM= $\{01_b\}$ + trigger or $\{10_b\}$ + trigger
- CDSP_DSPa_DSPCFG.SRLVL=$\{01_b\}$ or $\{10_b\}$ or $\{11_b\}$
- CDSP_DSPa_SRCFG.SR0E=$\{01_b\}$ + trigger or $\{10_b\}$ + trigger or $\{11_b\}$

Figure below illustrates the data flow.

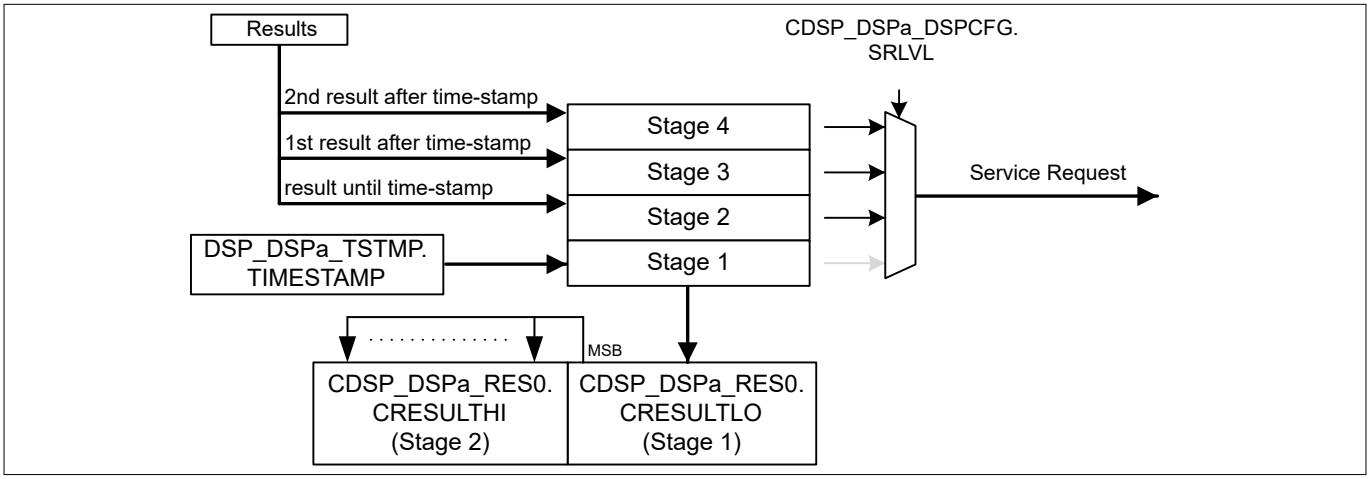


Figure 1236 Single data read mode with time-stamp

3. Double data read mode without time-stamp

In this mode the FIFO is queuing results in the order of arrival. Results will be stored in this mode only if service requests are enabled. The user has parallel read access to the FIFO stages 1 and 2. Stage 1 is assigned to CDSP_DSPa_RESy.CRESULTLO (y= 0) and stage 2 is assigned to CDSP_DSPa_RESy.CRESULTHI (y= 0)

Necessary setup:

- CDSP_DSPa_DSPCFG.DRM= 10_b
- CDSP_DSPa_DSPCFG.TSM= 0_b
- CDSP_DSPa_DSPCFG.SRLVL=$\{01_b\}$ or $\{11_b\}$
- CDSP_DSPa_SRCFG.SR0E=$\{01_b\}$ + trigger or $\{10_b\}$ + trigger or $\{11_b\}$

Figure below illustrates the data flow.

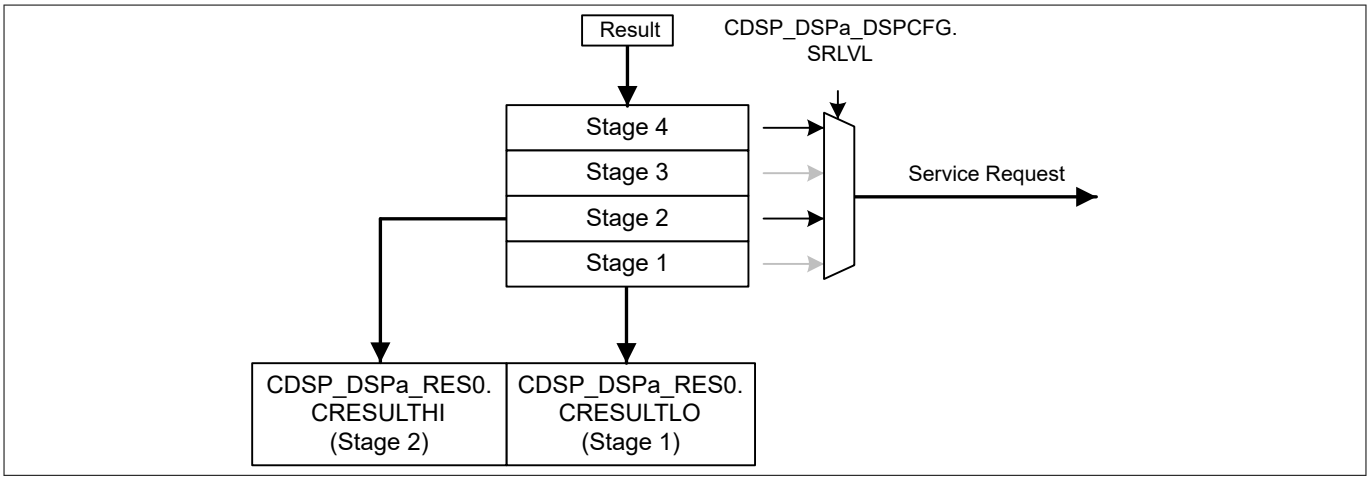


Figure 1237 Double data read mode without time-stamp

4. Double data read mode with time-stamp

In this mode the user reads the following sequence of values from the FIFO when the time-stamp was generated by the selected trigger edge:

1. Time-stamp (this time-stamp will also be stored in CDSP_DSPa_TSTMP.TIMESTAMP)
2. Result before time-stamp or result at time-stamp when the time-stamp is generated at the same point in time when a new result is generated
3. 1st result after time-stamp
4. 2nd result after time-stamp
5. …..nthresult after time-stamp

The purpose of this fixed order is that the user can differentiate between results and time-stamp because the time-stamp does not have a unique identifier. The FIFO switches between two sub-modes according to CDSP_DSPa_TSTRCFG.TSTRM:

1. Waiting for time-stamp
   - Entered after initialization
   - Service requests will not be generated
   - User has no read access to the FIFO (read accesses will result in a read error indicated by CDSP_DSPa_DSPST.RDERR)
2. Storing results (similar to double data read mode without time-stamp)
   - Entered after time-stamp trigger
   - Service requests will be generated if configured
   - User has read access

After the initialization the user should wait for a service request (generated after configured time-stamp trigger edge) before reading CDSP_DSPa_RESy (y=0).

Furthermore, it should be considered that the FIFO reacts only on the edges of the trigger signal. If for example the time-stamp trigger is high before and after setting CDSP_GLOBRCD.CHRUND= 1_b and a time-stamp shall be generated upon the rising edge of the time-stamp trigger (CDSP_DSPa_TSTRCFG.TSTRM= 10_B), then the FIFO will stay in sub-mode 'waiting for time-stamp' until the rising edge of the time-stamp trigger is generated. The FIFO will be initialized before, because the time-stamp trigger must be inactive before activating it again and therefore, a disabling time-stamp trigger edge will be generated.

Necessary setup:

- CDSP_DSPa_DSPCFG.DRM= 10_b
- CDSP_DSPa_DSPCFG.TSM= 1_b
- CDSP_DSPa_DSPCFG.TSCRUN= 1_b
- CDSP_DSPa_TSTRCFG.TSTRM= {01_b} + trigger or {10_b} + trigger
- CDSP_DSPa_DSPCFG.SRLVL={01_b} or {11_b}
- CDSP_DSPa_SRCFG.SR0E={01_b} + trigger or {10_b} + trigger or {11_b}

Figure below illustrates the data flow.

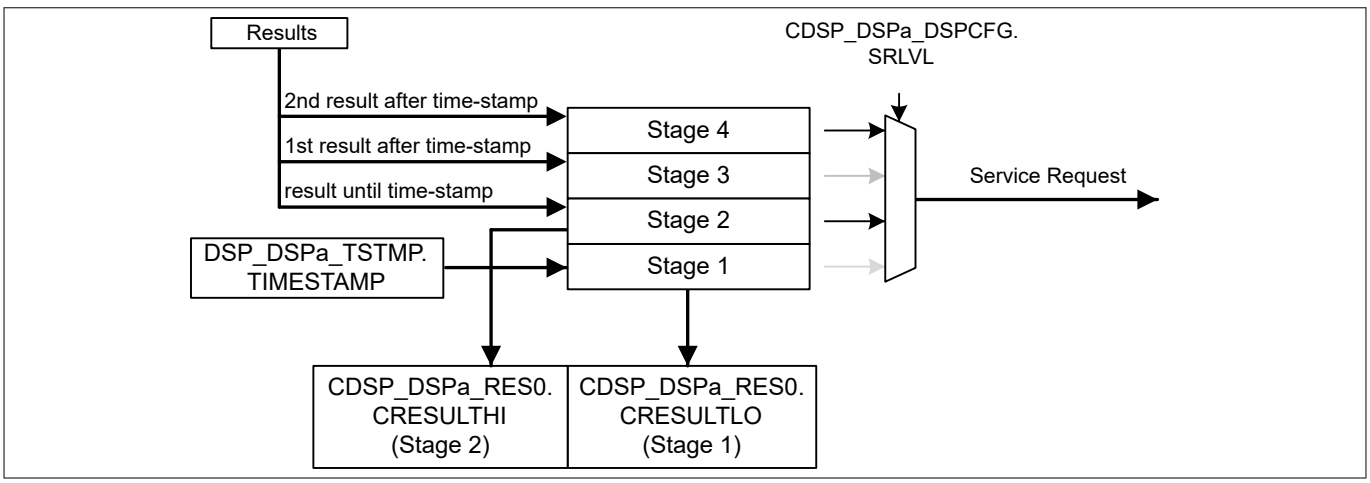


Figure 1238 Double data read mode with time-stamp

45.5.3.9.2 RES0 service request generation and read sequencing

The result values in CDSP_DSPa_RES0 (including time-stamp values, if enabled) are retrieved in a defined sequence. Result events are generated for each of these results. Result service requests are generated on SR0 depending on the configured read mode and the FIFO fill level. Each data transfer from CDSP_DSPa_RES0 (result data values or time-stamp information) is initiated by a service request. This service request is issued when a defined number of values configured in CDSP_DSPa_DSPCFG.SRLVL is available.

45.5.3.10 Limit checking

A comparator provides automatic limit checking by comparing each result with two configurable threshold values. This feature can be used to supervise the input signal and detect abnormal input values. The comparator can generate a dedicated service request.

For specific variants 2 additional equivalent comparator resources are available. Presence is noted in the device specific appendix. Register description placed at the end of this chapter.

The two threshold values are defined by the boundary select register CDSP_DSPa_BOUNDSEL and determine for example the expected result value band. The comparator could get configured to signal unexpected deviation.

The digital comparator can be used in following modes, configurable through CDSP_DSPa_BNDCFG.

- Boundary Mode: This mode is enabled by setting bit CDSP_DSPa_BNDCFG.BNDMODE= 1_b. The user can configure the generation of the boundary flags. Please refer to the figure below.

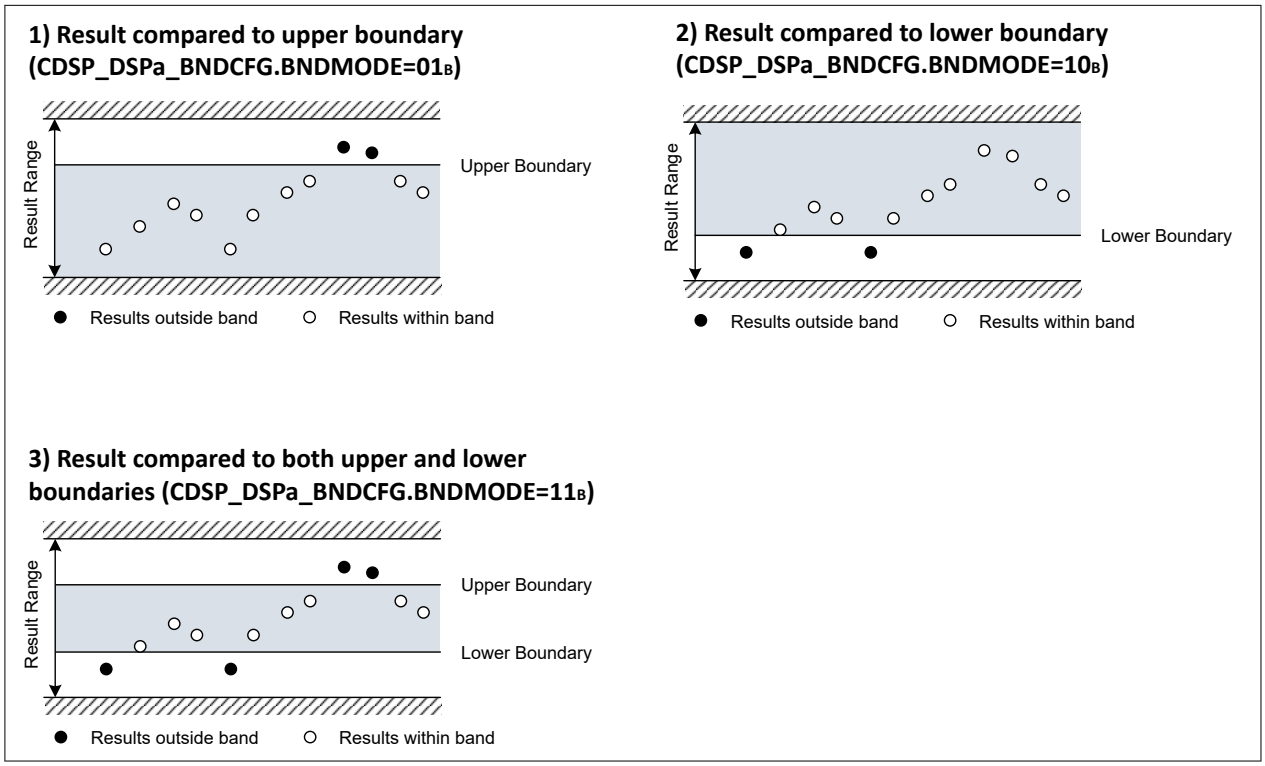


Figure 1239 Boundary mode in limit checking

- When CDSP_DSPa_BNDCFG.BNDMODE = 01_B, boundary flag is set when result exceeds upper limit

- When CDSP_DSPa_BNDCFG.BNDMODE = 10_B, boundary flag is set when result is lower than lower limit
- When CDSP_DSPa_BNDCFG,BNDMODE = 11_B, boundary flag is set if the result is either greater than the upper threshold or if the result falls short of the lower threshold.

- Hysteresis Mode: This mode is enabled by setting bit CDSP_DSPa_BNDCFG.HYSTEN =1. In this mode, the result value is compared to the upper and to the lower boundary. Boundary flag indicates when then result exceeds the upper boundary. It is cleared if the result falls short of the lower boundary

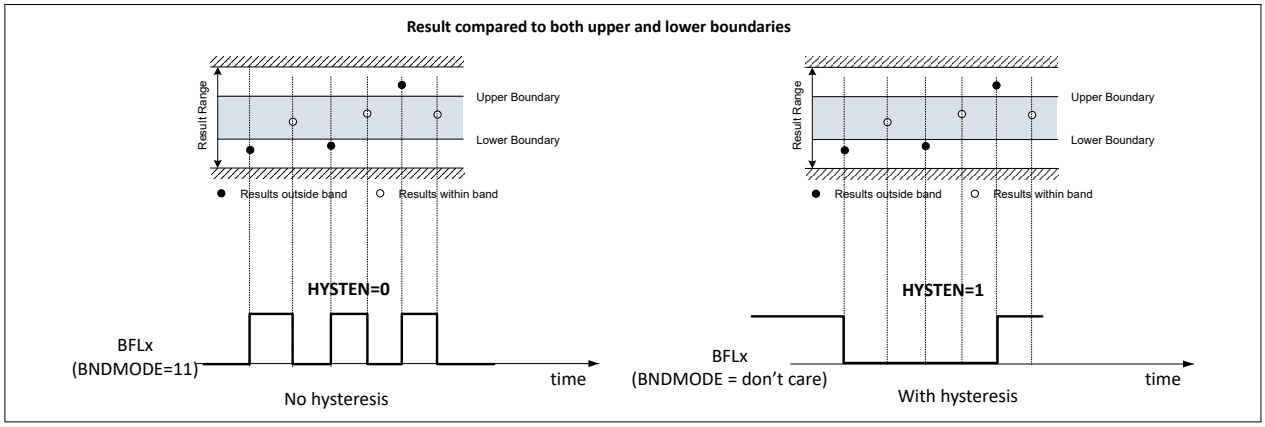


Figure 1240 Hysteresis mode in limit checking

A result value is considered between the defined boundaries when both of the following conditions are true:

- The value is less than or equal to the selected upper boundary
- The value is greater than or equal to the selected lower boundary

The limit checker can generate two types of output:

- Service requests configurable through CDSP_DSPa_SRCFG.SR2E and CDSP_DSPa_BNDCFG.SR2CFG
- Boundary flags configured by bits HYSTEN and BNDMODE of CDSP_DSPa_BNDTRCFG

The boundary values in the CDSP_DSPa_BOUNDSEL registers can be updated during runtime. The boundary update can be controlled by a selected hardware trigger. Trigger source is selectable through CDSP_DSPa_BNDTRCFG.BNDTRS and trigger mode by CDSP_DSPa_BNDTRCFG.BNDTRM. To apply a delay on the selected trigger, a delay value can be configured in CDSP_DSPa_BNDTRCFG.BNDTRDLY. Corresponding delay counters run on a fixed $f_{ADC}/2$ clock. Reading CDSP_DSPa_BOUNDST returns the active threshold for the in-band check.

Boundary events can be to result values that are inside or outside a user-defined band. This featuresupports automatic range monitoring and minimizes the CPU load by issuing service requests only under certain conditions.

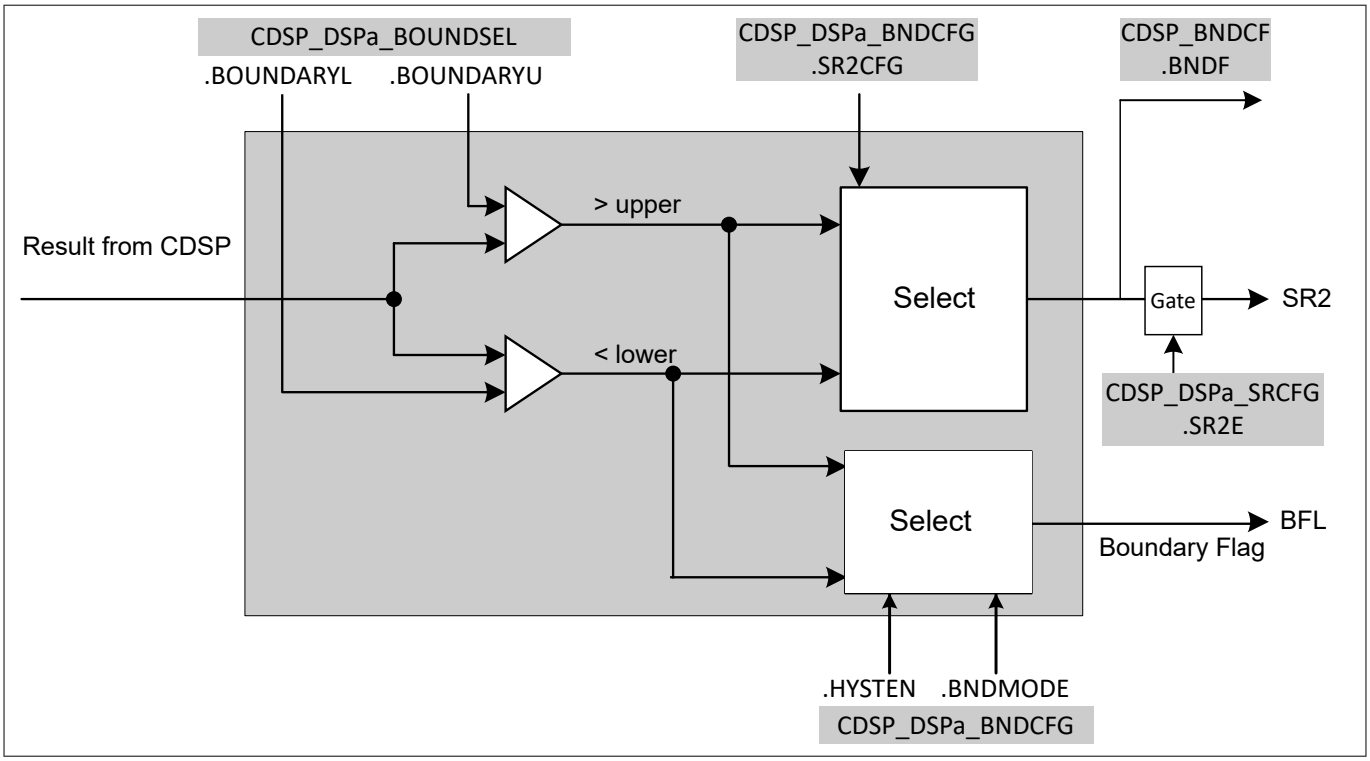


Figure 1241 Comparator structure

Only CDSP_DSPa_RESy (y= 0) lower 16 bits are used for comparison. The internal comparison is signed. In case of CDSP_DSPa_DSPCFG.RDM= 1_b (unsigned), the limits have to reflect this. 32768_d has to be subtracted from desired positive integer limit values.

In case the device variant holds 2 additional comparator resources, they can get configured and used like described above. The table below shows the equivalent register mapping for the additional comparators.

Table 2394

Family default	Additional comparator #1	Additional comparator #2
CDSP_BNDCF	CDSP_BNDCF1	CDSP_BNDCF2
CDSP_BNDCFCL	CDSP_BNDCFCL1	CDSP boundary flag event clear 2
CDSP_DSPa_BNDCFG	CDSP_DSPa_BNDCFG1	CDSP_DSPa_BNDCFG2
CDSP_DSPa_BNDTRCFG	CDSP_DSPa_BNDTRCFG1	CDSP_DSPa_BNDTRCFG2
CDSP_DSPa_BOUNDST	CDSP_DSPa_BOUNDST1	CDSP_DSPa_BOUNDST2

Service request 2 enable bits for the additional channels are CDSP_DSPa_SRCFG.SR2E1/SR2E2.

45.5.3.11 Service request generation

The CDSP can activate service request output signals to issue interrupts, to trigger DMA channels or to trigger other on-chip modules.

CDSP internal events can be assigned to service request outputs. Service requests can be generated by several types of events:

- Result events: Indicate the availability of new results. Result events are generated at the software output data rate (indicated by CDSP_RESEVy.RESEV, y= 0-2). Register RESEV0 captures the events of RES0 in all CDSPs, RESEV1 captures the events of RES1 in all CDSPs and RESEV2 captures the events of RES2 in

all CDSPs. Result service requests can be generated only for RES0 events. The corresponding control is provided by CDSP_DSPa_SRCFG.SR0E. The output rate of result service requests depends additionally on the selected FIFO fill level (see CDSP_DSPa_DSPCFG.SRLVL)

***Note**: Reading a result register (CDSP_DSPa_RESy) automatically clears the respective result event in CDSP_RESEVy.*

- Special events: Time-stamp trigger event, wake up error event or DSP core system halt event. Service requests for these events can be configured through CDSP_DSPa_SRCFG.SR1E.
- Boundary events: Indicate that a conversion result value is within or outside a programmable value range (event flags are saved in CDSP_BNDCF.BNDF). A service request is only activated if the specified conversion result range is met or exceeded. Can be configured through CDSP_DSPa_SRCFG.SR2E and CDSP_DSPa_BNDCFG.SR2CFG.

Each event is signaled by a dedicated flag. If a service request is enabled for an event, the service request is generated for each event, independent of the status of the corresponding event flag. This ensures efficient DMA handling of CDSP events (the event can generate a service request without the need to clear the indication flag).

Event flags are cleared by setting the corresponding event clear bit. The result events can be cleared manually with CDSP_RESEVCLRy.RESEVCLR. Boundary and wake-up error events can get cleared by setting the corresponding bits in DSADC_BOUNDFCL.BNDFCLR and CDSP_WUERRCL.WUERRCL. In addition event registers are cleared with assertion of a run request in CDSP_GLOBRCD.CHRUND(a).

45.5.3.12 CDSP start-up (Initialization)

Initialization steps:

1. Copy instruction code from Flash memory into the instruction memory of CDSP (ICCM)
2. Copy software configuration from the Flash memory to the dedicated area in the data memory of CDSP (DCCM)
3. Configure the CDSP_DSPa setup registers
4. Program CDSP_GLOBRCD to start program execution

45.6 Delta-Sigma Analog-to-Digital Converter (DSADC)

DSADC converts differential or single-ended analog inputs into 16-bit digital words. It is used for the continuous monitoring of signals. An external modulator can also be connected to EXMOD input channels and then the incoming bit-stream is demodulated by the EXMOD hardware.

Attention**: **DSADC is not present in this device, and so is not further described in this documentation.

45.7 Fast compare comparator (FCC)

***Attention**: FCC is not present in this device, and so is not further described in this documentation.*

45.8 Registers

45.8.1 Register overview - access mode glossary

Table 2395 Register overview - access mode glossary

Keyword	Description
Ex (x=0-15)	Access protection using PROT register PROTEx.
SE	Access protection using PROT register PROTSE.
APU-PCTRLy (y=0-15)	Protection group consisting of registers ACCENy_WRA, ACCENy_WRB, ACCENy_RDA, ACCENy_RDB, ACCENy_VM, ACCENy_PRS.
PCTRLy	Access protection using *APU-PCTRLy* registers.
PTOP ETOP PADC EADC	Access protection using one APU from *APU-PCTRL[y] (y=0-15)*. PROT_E protection using one of *E[x] (x=0-15)*. The protection source to be used is selected by bit-field: RESALLOC_GLOB.GLOB
PMUXz (z=19,38) EMUXz (z=19,38)	Access protection using one APU from *APU-PCTRL[y] (y=0-15)*. PROT_E protection using one of *E[x] (x=0-15)*. The protection source to be used is selected by bit-field: z=19: RESALLOC_GLOB.DSINEX z=38: RESALLOC_GLOB.CDSPGL
PTMADCz (z=0-6) ETMADCz (z=0-6)	Access protection using one APU from *APU-PCTRL[y] (y=0-15)*. PROT_E protection using one of *E[x] (x=0-15)*. The protection source to be used is selected by bit-field: z=0-6: RESALLOC_TMADC.TMADCz
PFCCz (z=0-9) EFCCz (z=0-9)	Access protection using one APU from *APU-PCTRL[y] (y=0-15)*. PROT_E protection using one of *E[x] (x=0-15)*. The protection source to be used is selected by bit-field: z=0-7: RESALLOC_FCCA.FCCz z=8-9: RESALLOC_FCCB.FCCz
PDSADCz (z=0-9) EDSADCz (z=0-9)	Access protection using one APU from *APU-PCTRL[y] (y=0-15)*. PROT_E protection using one of *E[x] (x=0-15)*. The protection source to be used is selected by bit-field: z=0-7: RESALLOC_DSADCA.DSADCz z=8-9: RESALLOC_DSADCB.DSADCz
PEXMODz (z=0-7) EEXMODz (z=0-7)	Access protection using one APU from *APU-PCTRL[y] (y=0-15)*. PROT_E protection using one of *E[x] (x=0-15)*. The protection source to be used is selected by bit-field: z=0-7: RESALLOC_EXMOD.EXMODz

(table continues...)

Table 2395 (continued) Register overview - access mode glossary

Keyword	Description
PCDSPz (z=0-11) ECDSPz (z=0-11)	Access protection using one APU from *APU-PCTRL[y] (y=0-15)*. PROT_E protection using one of *E[x] (x=0-15)*. The protection source to be used is selected by bit-field: z=0-7: RESALLOC_CDSPA.CDSPz z=8-11: RESALLOC_CDSPB.CDSPz
BE	Always returns a Bus Error.
U	No access restrictions.
PROT	Access restrictions as defined in the PROT register access rules.
32	Access only when using 32-bit width.

45.8.2 Register overview - ADC (ascending offset address)

Table 2396 Register overview - ADC (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CLC	Clock Control Register	00000_H	PCTRL0	PCTRL0, SV, E0	Application Reset	8403
ID	Module Identification Register	00008_H	PCTRL0	BE	PowerOn Reset	8403
PROTSE	PROT Register Safe Endinit	$0000C_H$	U	SV, PROT	Application Reset	8404
RESALLOC_GLOB	Resource allocation register for Global Rsources	00010_H	U	SE, SV	Application Reset	8406
RESALLOC_TMADC	Resource allocation register for TMADC Resources	00014_H	U	SE, SV	Application Reset	8406
RESALLOC_FCCA	Resource allocation register for FCC	00018_H	U	SE, SV	Application Reset	8407
RESALLOC_FCCB	Resource allocation register for FCC	$0001C_H$	U	SE, SV	Application Reset	8407
RESALLOC_DSADCA	Resource allocation register for DSADC	00020_H	U	SE, SV	Application Reset	8408
RESALLOC_DSADCB	Resource allocation register for DSADC	00024_H	U	SE, SV	Application Reset	8408
RESALLOC_EXMOD	Resource allocation register for EXMOD	00028_H	U	SE, SV	Application Reset	8409
RESALLOC_CDSPA	Resource allocation register for CDSP	$0002C_H$	U	SE, SV	Application Reset	8409
RESALLOC_CDSPB	Resource allocation register for CDSP	00030_H	U	SE, SV	Application Reset	8410
PROTEx	APUx PROT Register Endinit	00080_H+ x*4	U	SV, PROT	Application Reset	8410

(table continues...)

Table 2396 (continued) Register overview - ADC (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ACCENy_WRA	Write access enable register A, APU y	$000C0_H$+ y*20_H	U	SE, SV	Application Reset	8412
ACCENy_WRB	Write access enable register B, APU y	$000C4_H$+ y*20_H	U	SE, SV	Application Reset	8413
ACCENy_RDA	Read access enable register A, APU y	$000C8_H$+ y*20_H	U	SE, SV	Application Reset	8413
ACCENy_RDB	Read access enable register B, APU y	$000CC_H$+ y*20_H	U	SE, SV	Application Reset	8414
ACCENy_VM	VM access enable register, APU y	$000D0_H$+ y*20_H	U	SE, SV	Application Reset	8414
ACCENy_PRS	PRS access enable register, APU y	$000D4_H$+ y*20_H	U	SE, SV	Application Reset	8415
RSTi_CTRLA	Reset Control Register A, APU i	$002C0_H$+ i*12	PCTRLi	PCTRLi, SV, Ei	Application Reset	8416
RSTi_CTRLB	Reset Control Register B, APU i	$002C4_H$+ i*12	PCTRLi	PCTRLi, SV, Ei	Application Reset	8416
RSTi_STAT	Reset Status Register, APU i	$002C8_H$+ i*12	PCTRLi	BE	Application Reset	8417
GLSRCFGi	ADC global service request i configuration register	00800_H+i *4	PTOP	PTOP	Kernel Reset	8418
GLALCFGi	ADC global alarm i configuration register	00810_H+i *4	PTOP	PTOP	Kernel Reset	8418
GLALSET	ADC global alarm set register	00830_H	PTOP	PTOP	Kernel Reset	8420
GLSRSET	ADC global service request set register	00834_H	PTOP	PTOP	Kernel Reset	8420
ETBFxSEL	ADC to eGTM boundary flag select register, cluster x	00838_H+ x*4	PTOP	PTOP	Kernel Reset	8421
GTBFxSEL	ADC to GTM boundary flag select register, cluster x	00844_H+ x*4	PTOP	PTOP	Kernel Reset	8422
GTSRxSEL	ADC to GTM service request select register, cluster x	00864_H+ x*4	PTOP	PTOP	Kernel Reset	8424
PDDCFG	ADC pull down diagnostics configuration register	$0088C_H$	PTOP	PTOP	Kernel Reset	8425
CLKEN_TMADC	Clock enable for TMADC	00890_H	clk	clk	Kernel Reset	8426
CLKEN_FCC	Clock enable for FCC	00894_H	32, clk	32, clk	Kernel Reset	8426
CLKEN_DSINEX	Clock enable for DSADC and EXMOD	00898_H	32, clk	32, clk	Kernel Reset	8427
CLKEN_CDSP	Clock enable for CDSP	$0089C_H$	32, clk	32, clk	Kernel Reset	8427
SUPLLEV	Supply level selection for ADC	$008A0_H$	PTOP	PTOP, clk	Kernel Reset	8428

(table continues...)

Table 2396 (continued) Register overview - ADC (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
TMSMOD	TMADC supervision module register	$008A4_H$	PTOP	PTOP, clk	Kernel Reset	8429
OCSTM	OCDS control and status register for TMADC	$008A8_H$	PTOP	SV, PTOP, clk	Debug Reset	8429
TMADCx_MODCFG	TMADCx module configuration register	20000_H+ x*800_H	PTMADCx	PTMADCx, clk	Kernel Reset	8431
TMADCx_EMUXCFG	TMADCx EMUX configuration register	20004_H+ x*800_H	PTMADCx	PTMADCx, clk	Kernel Reset	8432
TMADCx_SWTRCFG	TMADCx software trigger configuration register	$2000C_H$+ x*800_H	PTMADCx, 32	PTMADCx, 32, clk	Kernel Reset	8433
TMADCx_CMATCFG	TMADCx connection matrix configuration register	20010_H+ x*800_H	PTMADCx	PTMADCx, clk	Kernel Reset	8433
TMADCx_CHy_STC	TMADCx sampling time control register, channel y	20018_H+ x*800_H+ y*12	PTMADCx	PTMADCx, clk	Kernel Reset	8434
TMADCx_CHy_CFG	TMADCx configuration register, channel y	$2001C_H$+ x*800_H+ y*12	PTMADCx	PTMADCx, clk	Kernel Reset	8435
TMADCx_CHy_STAT	TMADCx status register, channel y	20020_H+ x*800_H+ y*12	PTMADCx	PTMADCx, clk	Kernel Reset	8436
TMADCx_MCHy_CFG	TMADCx configuration register, monitor channel y	$200D8_H$+ x*800_H+ y*4	PTMADCx	PTMADCx, clk	Kernel Reset	8437
TMADCx_BNDn_SEL	TMADCx boundary select register n	$200E0_H$+ x*800_H+ n*12	PTMADCx	PTMADCx, clk	Kernel Reset	8438
TMADCx_BNDn_STAT	TMADCx boundary status regsiter n	$200E4_H$+ x*800_H+ n*12	PTMADCx	PTMADCx, clk	Kernel Reset	8439
TMADCx_BNDn_TRGCFG	TMADCx boundary trigger configuration register, boundary register n	$200E8_H$+ x*800_H+ n*12	PTMADCx	PTMADCx, clk	Kernel Reset	8439
TMADCx_BFLn_CFG	TMADCx boundary flag configuration register n	$200F8_H$+ x*800_H+ n*8	PTMADCx	PTMADCx, clk	Kernel Reset	8440
TMADCx_BFLn_STAT	TMADCx boundary flag status register n	$200FC_H$+ x*800_H+ n*8	PTMADCx	PTMADCx, clk	Kernel Reset	8441
TMADCx_RSLTu_CFG	TMADCx configuration register, result register u	$2010C_H$+ x*800_H+ u*8	PTMADCx	PTMADCx, clk	Kernel Reset	8441

(table continues...)

Table 2396 (continued) Register overview - ADC (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
TMADCx_MRSLTu_CFG	TMADCx configuration register, monitor channel u result register	20188_H+ x*800_H+ u*4	PTMADCx	PTMADCx, clk	Kernel Reset	8442
TMADCx_ERRF	TMADCx error event flag register	20190_H+ x*800_H	PTMADCx	PTMADCx, clk	Kernel Reset	8443
TMADCx_RESF	TMADCx result event flag register	20194_H+ x*800_H	PTMADCx	PTMADCx, clk	Kernel Reset	8444
TMADCx_BNDF	TMADCx boundary event flag register	20198_H+ x*800_H	PTMADCx	PTMADCx, clk	Kernel Reset	8444
TMADCx_RESFSET	TMADCx result event flag set register	$201A0_H$+ x*800_H	PTMADCx, 32	PTMADCx, 32, clk	Kernel Reset	8445
TMADCx_ERRFCLR	TMADCx error event flag clear register	$201A4_H$+ x*800_H	PTMADCx, 32	PTMADCx, 32, clk	Kernel Reset	8445
TMADCx_RESFCLR	TMADCx result event flag clear register	$201A8_H$+ x*800_H	PTMADCx, 32	PTMADCx, 32, clk	Kernel Reset	8446
TMADCx_BNDFCLR	TMADCx boundary event flag clear register	$201AC_H$+ x*800_H	PTMADCx, 32	PTMADCx, 32, clk	Kernel Reset	8446
TMADCx_SRi_CFG	TMADCx service request configuration register i	$201B0_H$+ x*800_H+i *4	PTMADCx	PTMADCx, clk	Kernel Reset	8447
TMADCx_MRESv	TMADCx monitor channel v result register	$201CC_H$+ x*800_H+ v*4	PTMADCx	PTMADCx, clk	Kernel Reset	8448
TMADCx_MODSTAT	TMADCx module status register	$201D4_H$+ x*800_H	PTMADCx	PTMADCx, clk	Kernel Reset	8449
TMADCx_BWDCFG	TMADCx broken wire detection control register	$201D8_H$+ x*800_H	PTMADCx	PTMADCx, clk	Kernel Reset	8449
TMADCx_CSDn	TMADCx converter self-diagnostics register core-n	$201DC_H$+ x*800_H+ n*4	PTMADCx	PTMADCx, clk	Kernel Reset	8450
TMADCx_AW0_TSv	TMADCx time-stamp register v	20200_H+ x*800_H+ v*4	PTMADCx	PTMADCx, clk	Kernel Reset	8451
TMADCx_AW0_RESv	TMADCx result register v	20240_H+ x*800_H+ v*4	PTMADCx	PTMADCx, clk	Kernel Reset	8451
TMADCx_AW1_TSv	TMADCx time-stamp register v	20280_H+ x*800_H+ v*8	PTMADCx	PTMADCx, clk	Kernel Reset	8451
TMADCx_AW1_RESv	TMADCx result register v	20284_H+ x*800_H+ v*8	PTMADCx	PTMADCx, clk	Kernel Reset	8451

(table continues...)

Table 2396 (continued) Register overview - ADC (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
TMADCx_AW2_TSv	TMADCx time-stamp register v	20300_H+x*800_H	PTMADCx	PTMADCx, clk	Kernel Reset	8451
TMADCx_AW2_RESv	TMADCx result register v	20304_H+x*800_H+v*4	PTMADCx	PTMADCx, clk	Kernel Reset	8451
TMADCx_AW2_TSv	TMADCx time-stamp register v	20340_H+x*800_H	PTMADCx	PTMADCx, clk	Kernel Reset	8451
TMADCx_AW2_RESv	TMADCx result register v	20344_H+x*800_H+(v-7)*4	PTMADCx	PTMADCx, clk	Kernel Reset	8451
TMADCx_AW2_TSv	TMADCx time-stamp register v	20360_H+x*800_H	PTMADCx	PTMADCx, clk	Kernel Reset	8451
TMADCx_AW2_RESv	TMADCx result register v	20364_H+x*800_H+(v-14)*4	PTMADCx	PTMADCx, clk	Kernel Reset	8451
TMADCx_AW3_RESv	TMADCx result register v	$2036C_H$+x*800_H+v*4	PTMADCx	PTMADCx, clk	Kernel Reset	8451
FCCx_THTRCFG	FCCx threshold trigger configuration register	40000_H+x*400_H	PFCCx	PFCCx, clk	Kernel Reset	8452
FCCx_GATRCFG	FCCx gate trigger configuration register	40004_H+x*400_H	PFCCx	PFCCx, clk	Kernel Reset	8453
FCCx_RMPTRCFG	FCCx ramp trigger configuration register	40008_H+x*400_H	PFCCx	PFCCx, clk	Kernel Reset	8454
FCCx_RMPCFG	FCCx ramp configuration register	$4000C_H$+x*400_H	PFCCx	PFCCx, clk	Kernel Reset	8455
FCCx_RMPT	FCCx ramp threhsold register	40010_H+x*400_H	PFCCx	PFCCx, clk	Kernel Reset	8456
FCCx_MODEN	FCCx module enable register	40014_H+x*400_H	PFCCx	PFCCx, clk	Kernel Reset	8456
FCCx_MODCFG	FCCx module configuration register	40018_H+x*400_H	PFCCx	PFCCx, clk	Kernel Reset	8456
FCCx_HYSTCFG	FCCx hyteresis configuration register	$4001C_H$+x*400_H	PFCCx	PFCCx, clk	Kernel Reset	8458
FCCx_BFLCFG	FCCx boundary flag configuration register	40020_H+x*400_H	PFCCx	PFCCx, clk	Kernel Reset	8458
FCCx_SRCFG	FCCx service request configuration register	40024_H+x*400_H	PFCCx	PFCCx, clk	Kernel Reset	8459
FCCx_MODSTAT	FCCx module status register	40028_H+x*400_H	PFCCx	PFCCx, clk	Kernel Reset	8459
DSADC_GLOBCFG	DSADC global configuration register	$5F800_H$	PMUX19	PMUX19, clk	Kernel Reset	8461

(table continues...)

Table 2396 (continued) Register overview - ADC (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
DSADC_GLOBRCF	DSADC global run control register for filter chain	$5F804_H$	U	32, clk	Kernel Reset	8461
DSADC_GLOBRCM	DSADC global run control register for analog modulator	$5F808_H$	U	32, clk	Kernel Reset	8462
DSADC_CGCFG	DSADC carrier generator configuration register	$5F80C_H$	PMUX19	PMUX19, clk	Kernel Reset	8462
DSADC_CGST	DSADC carrier generator status register	$5F810_H$	PMUX19	PMUX19, clk	Kernel Reset	8463
DSADC_RESMF	DSADC event flag register on main result register	$5F814_H$	U	32, clk	Kernel Reset	8464
DSADC_RESAF	DSADC event flag register on auxiliary result register	$5F818_H$	U	32, clk	Kernel Reset	8465
DSADC_BOUNDF	DSADC boundary flag event register	$5F81C_H$	U	32, clk	Kernel Reset	8465
DSADC_RESMFCL	DSADC result event (main) flag clear register	$5F820_H$	U	32, clk	Kernel Reset	8466
DSADC_RESAFCL	DSADC result event (auxiliary) flag clear register	$5F824_H$	U	32, clk	Kernel Reset	8466
DSADC_BOUNDFCL	DSADC boundary event flag clear register	$5F828_H$	U	32, clk	Kernel Reset	8467
DSADC_OCSDSADC	OCDS control and status register for DSADC	$5F82C_H$	U	SV, 32, clk	Debug Reset	8468
DSADC_OCSEXMOD	OCDS control and status register for EXMOD	$5F830_H$	U	SV, 32, clk	Debug Reset	8469
DSADC_INx_CHCFG	DSADCx channel configuration register	60000_H+ x*400_H	PDSADCx	PDSADCx, clk	Kernel Reset	8470
DSADC_INx_CHST	DSADCx channel status register	60004_H+ x*400_H	PDSADCx	PDSADCx, clk	Kernel Reset	8473
DSADC_INx_VCM	DSADCx common mode voltage register	60008_H+ x*400_H	PDSADCx	PDSADCx, clk	Kernel Reset	8474
DSADC_INx_CICCFG	DSADCx CIC configuration register	$6000C_H$+ x*400_H	PDSADCx	PDSADCx, clk	Kernel Reset	8475
DSADC_INx_INTCFG	DSADCx integrator configuration register	60010_H+ x*400_H	PDSADCx	PDSADCx, clk	Kernel Reset	8476
DSADC_INx_INTIVAL	DSADCx intermediate integration value register	60014_H+ x*400_H	PDSADCx	PDSADCx, clk	Kernel Reset	8477
DSADC_INx_INTST	DSADCx integrator status register	60018_H+ x*400_H	PDSADCx	PDSADCx, clk	Kernel Reset	8478
DSADC_INx_GAINCAL	DSADCx gain calibration register	$6001C_H$+ x*400_H	PDSADCx, 32	PDSADCx, 32, clk	Kernel Reset	8479

(table continues...)

Table 2396 (continued) Register overview - ADC (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
DSADC_INx_GAINCTR	DSADCx gain control register	60020_H+ x*400_H	PDSADCx	PDSADCx, clk	Kernel Reset	8479
DSADC_INx_GAINCORR	DSADCx gain correction register	60024_H+ x*400_H	PDSADCx	PDSADCx, clk	Kernel Reset	8480
DSADC_INx_OFFCOMP	DSADCx offset compensation register	60028_H+ x*400_H	PDSADCx, 32	PDSADCx, 32, clk	Kernel Reset	8481
DSADC_INx_TSTMP	DSADCx timestamp register	$6002C_H$+ x*400_H	PDSADCx	PDSADCx, clk	Kernel Reset	8481
DSADC_INx_TSCNT	DSADCx timestamp counter	60030_H+ x*400_H	PDSADCx	PDSADCx, clk	Kernel Reset	8482
DSADC_INx_RESM	DSADCx result register main	60034_H+ x*400_H	PDSADCx	PDSADCx, clk	Kernel Reset	8482
DSADC_INx_RESA	DSADCx result register auxiliary	60038_H+ x*400_H	PDSADCx	PDSADCx, clk	Kernel Reset	8483
DSADC_INx_CGSYNC	DSADCx carrier generator synchronization register	$6003C_H$+ x*400_H	PDSADCx	PDSADCx, clk	Kernel Reset	8483
DSADC_INx_RECTCFG	DSADCx rectification configuration register	60040_H+ x*400_H	PDSADCx	PDSADCx, clk	Kernel Reset	8484
DSADC_INx_BOUNDSEL	DSADCx boundary select register	60044_H+ x*400_H	PDSADCx, 32	PDSADCx, 32, clk	Kernel Reset	8485
DSADC_INx_INTTRCFG	DSADCx integrator trigger configuration register	60048_H+ x*400_H	PDSADCx, 32	PDSADCx, 32, clk	Kernel Reset	8485
DSADC_INx_TSTRCFG	DSADCx timestamp trigger configuration register	$6004C_H$+ x*400_H	PDSADCx	PDSADCx, clk	Kernel Reset	8486
DSADC_INx_IMXTRCFG	DSADCx input mux trigger configuration register	60050_H+ x*400_H	PDSADCx	PDSADCx, clk	Kernel Reset	8487
DSADC_INx_SRTRCFG	DSADCx service request trigger configuration register	60054_H+ x*400_H	PDSADCx	PDSADCx, clk	Kernel Reset	8487
DSADC_INx_CHERRCL	DSADCx channel error clear register	60058_H+ x*400_H	PDSADCx	PDSADCx, clk	Kernel Reset	8488
DSADC_INx_SRCFG	DSADCx service request configuration register	$6005C_H$+ x*400_H	PDSADCx	PDSADCx, clk	Kernel Reset	8489
DSADC_INx_OTA	DSADCx integrator operational amplifier configuration	60060_H+ x*400_H	PDSADCx	PDSADCx, clk	Kernel Reset	8490
DSADC_INx_CGRECST	DSADCx carrier generation and rectification status register	60064_H+ x*400_H	PDSADCx	PDSADCx, clk	Kernel Reset	8491
DSADC_INx_BNDCFG	DSADCx boundary mode configuration register	60068_H+ x*400_H	PDSADCx	PDSADCx, clk	Kernel Reset	8491

(table continues...)

Table 2396 (continued) Register overview - ADC (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
DSADC_INx_BNDTRCFG	DSADCx boundary request trigger configuration register	$6006C_H$+x*400_H	PDSADCx	PDSADCx, clk	Kernel Reset	8492
DSADC_INx_BOUNDST	DSADCx boundary status register	60070_H+x*400_H	PDSADCx	PDSADCx, clk	Kernel Reset	8493
DSADC_EXz_CHCFG	EXMODz channel configuration register	63800_H+z*400_H	PEXMODz	PEXMODz, clk	Kernel Reset	8493
DSADC_EXz_CHST	EXMODz channel status register	63804_H+z*400_H	PEXMODz	PEXMODz, clk	Kernel Reset	8496
DSADC_EXz_CICCFG	EXMODz CIC configuration register	63808_H+z*400_H	PEXMODz	PEXMODz, clk	Kernel Reset	8497
DSADC_EXz_INTCFG	EXMODz integrator configuration register	$6380C_H$+z*400_H	PEXMODz	PEXMODz, clk	Kernel Reset	8498
DSADC_EXz_TSTMP	EXMODz timestamp register	63810_H+z*400_H	PEXMODz	PEXMODz, clk	Kernel Reset	8498
DSADC_EXz_TSCNT	EXMODz timestamp counter	63814_H+z*400_H	PEXMODz	PEXMODz, clk	Kernel Reset	8499
DSADC_EXz_RESM	EXMODz result register main	63818_H+z*400_H	PEXMODz	PEXMODz, clk	Kernel Reset	8499
DSADC_EXz_RESA	EXMODz result register auxiliary	$6381C_H$+z*400_H	PEXMODz	PEXMODz, clk	Kernel Reset	8500
DSADC_EXz_BOUNDSEL	EXMODz boundary select register	63820_H+z*400_H	PEXMODz, 32	PEXMODz, 32, clk	Kernel Reset	8500
DSADC_EXz_INTTRCFG	EXMODz integrator trigger configuration register	63824_H+z*400_H	PEXMODz, 32	PEXMODz, 32, clk	Kernel Reset	8501
DSADC_EXz_TSTRCFG	EXMODz timestamp trigger configuration register	63828_H+z*400_H	PEXMODz	PEXMODz, clk	Kernel Reset	8501
DSADC_EXz_SRTRCFG	EXMODz service request trigger configuration register	$6382C_H$+z*400_H	PEXMODz	PEXMODz, clk	Kernel Reset	8502
DSADC_EXz_CHERRCL	EXMODz channel error clear register	63830_H+z*400_H	PEXMODz	PEXMODz, clk	Kernel Reset	8503
DSADC_EXz_GAINCORR	EXMODz gain correction register	63834_H+z*400_H	PEXMODz	PEXMODz, clk	Kernel Reset	8503
DSADC_EXz_SRCFG	EXMODz service request configuration register	63838_H+z*400_H	PEXMODz	PEXMODz, clk	Kernel Reset	8504
DSADC_EXz_BNDCFG	EXMODz boundary mode configuration register	$6383C_H$+z*400_H	PEXMODz	PEXMODz, clk	Kernel Reset	8505
DSADC_EXz_BNDTRCFG	EXMODz boundary request trigger configuration register	63840_H+z*400_H	PEXMODz	PEXMODz, clk	Kernel Reset	8506

(table continues...)

Table 2396 (continued) Register overview - ADC (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
DSADC_EXz_BOUNDST	EXMODz boundary status register	63844_H+z*400_H	PEXMODz	PEXMODz, clk	Kernel Reset	8506
CDSP_GLOBRD	CDSP global reset control	$BF804_H$	U	32, clk	Kernel Reset	8507
CDSP_BNDCF	CDSP boudary flag event register	$BF808_H$	U	32, clk	Kernel Reset	8507
CDSP_BNDCFCL	CDSP boundary flag event clear	$BF80C_H$	U	32, clk	Kernel Reset	8508
CDSP_SLPST	CDSP sleep status register	$BF810_H$	U	clk	Kernel Reset	8508
CDSP_RESEVy	CDSP event flag register y	$BF814_H$+y*4	U	32, clk	Kernel Reset	8509
CDSP_RESEVCLRy	CDSP result event flag clear y	$BF820_H$+y*4	U	32, clk	Kernel Reset	8509
CDSP_BNDCF1	CDSP boudary flag event register 1	$BF82C_H$	U	32, clk	Kernel Reset	8510
CDSP_BNDCF2	CDSP boudary flag event register 2	$BF830_H$	U	32, clk	Kernel Reset	8510
CDSP_BNDCFCL1	CDSP boundary flag event clear 1	$BF834_H$	U	32, clk	Kernel Reset	8510
CDSP_BNDCFCL2	CDSP boundary flag event clear 2	$BF838_H$	U	32, clk	Kernel Reset	8511
CDSP_GPx	CDSP input from general purpose register x	$BF8C0_H$+x*4	PMUX38	PMUX38, clk	Kernel Reset	8511
CDSP_WUERREV	CDSP wake up error flag register	$BF8D0_H$	U	32, clk	Kernel Reset	8512
CDSP_WUERRCL	CDSP wake up error event clear register	$BF8D4_H$	U	32, clk	Kernel Reset	8512
CDSP_GLOBRCD	CDSP global run control register	$BF8D8_H$	U	32, clk	Kernel Reset	8513
CDSP_GLOBHCD	CDSP global halt control register	$BF8DC_H$	U	32, clk	Kernel Reset	8513
CDSP_GLOBRSD	CDSP global run status register	$BF8E0_H$	U	clk	Kernel Reset	8514
CDSP_GLOBHSTD	CDSP global halt status register	$BF8E4_H$	U	clk	Kernel Reset	8514
CDSP_OCSCDSPA	OCDS control and status register 0 for CDSP0-8	$BF8E8_H$	U	SV, 32, clk	Debug Reset	8515
CDSP_OCSCDSPB	OCDS control and status register 1 for CDSP9-17	$BF8EC_H$	U	SV, 32, clk	Debug Reset	8516
CDSP_DBd_STATUS	CDSPd debug status register	$BFC00_H$+d*20_H	SV, PCDSPd, clk	SV, PCDSPd, 32, clk	Kernel Reset	8517

(table continues...)

Table 2396 (continued) Register overview - ADC (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CDSP_DBd_CMD	CDSPd debug command register	$BFC04_H$+ d*20_H	SV, PCDSPd, clk	SV, PCDSPd, 32, clk	Kernel Reset	8518
CDSP_DBd_ADDR	CDSPd debug address register	$BFC08_H$+ d*20_H	SV, PCDSPd, clk	SV, PCDSPd, 32, clk	Kernel Reset	8519
CDSP_DBd_DATA	CDSPd debug data register	$BFC0C_H$+ d*20_H	SV, PCDSPd, clk	SV, PCDSPd, 32, clk	Kernel Reset	8519
CDSP_DBd_RESET	CDSPd debug reset register	$BFC10_H$+ d*20_H	SV, PCDSPd, clk	SV, PCDSPd, 32, clk	Kernel Reset	8520
CDSP_DSPa_DSPCFG	CDSPa configuration register	$C1C00_H$+ a*2000_H	PCDSPa	PCDSPa, clk	Kernel Reset	8520
CDSP_DSPa_BNDTRCFG	CDSPa boundary trigger configuration register	$C1C04_H$+ a*2000_H	PCDSPa	PCDSPa, clk	Kernel Reset	8522
CDSP_DSPa_BNDCFG	CDSPa boundary mode configuration register	$C1C08_H$+ a*2000_H	PCDSPa	PCDSPa, clk	Kernel Reset	8523
CDSP_DSPa_BOUNDST	CDSPa boundary status register	$C1C0C_H$+ a*2000_H	PCDSPa	PCDSPa, clk	Kernel Reset	8523
CDSP_DSPa_BOUNDSEL	CDSPa boundary select register	$C1C10_H$+ a*2000_H	PCDSPa, 32	PCDSPa, 32, clk	Kernel Reset	8524
CDSP_DSPa_TSTRCFG	CDSPa time stamp trigger configuration register	$C1C14_H$+ a*2000_H	PCDSPa	PCDSPa, clk	Kernel Reset	8524
CDSP_DSPa_TSTMP	CDSPa time-stamp register	$C1C18_H$+ a*2000_H	PCDSPa	PCDSPa, clk	Kernel Reset	8525
CDSP_DSPa_TSCNT	CDSPa time-stamp counter	$C1C1C_H$+ a*2000_H	PCDSPa	PCDSPa, clk	Kernel Reset	8525
CDSP_DSPa_SRTRCFG	CDSPa service request trigger configuration register	$C1C20_H$+ a*2000_H	PCDSPa	PCDSPa, clk	Kernel Reset	8526
CDSP_DSPa_SRCFG	CDSPa service request configuration register	$C1C24_H$+ a*2000_H	PCDSPa	PCDSPa, clk	Kernel Reset	8527
CDSP_DSPa_SWCFG	CDSPa software configuration register	$C1C28_H$+ a*2000_H	PCDSPa	PCDSPa, clk	Kernel Reset	8528
CDSP_DSPa_INTIVAL	CDSPa intermediate integration value	$C1C2C_H$+ a*2000_H	PCDSPa	PCDSPa, clk	Kernel Reset	8528
CDSP_DSPa_SWTRCFG	CDSPa software trigger configuration register	$C1C30_H$+ a*2000_H	PCDSPa	PCDSPa, clk	Kernel Reset	8529
CDSP_DSPa_CHERRCL	CDSPa channel error clear register	$C1C34_H$+ a*2000_H	PCDSPa	PCDSPa, clk	Kernel Reset	8530
CDSP_DSPa_DSPST	CDSPa status register	$C1C38_H$+ a*2000_H	PCDSPa	PCDSPa, clk	Kernel Reset	8530

(table continues...)

Table 2396 (continued) Register overview - ADC (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
CDSP_DSPa_INPUTMON	CDSPa input from DSADC, TMADC, EXMOD or CARMAG	$C1C3C_H$+ a*2000_H	PCDSPa	PCDSPa, clk	Kernel Reset	8532
CDSP_DSPa_RESy	CDSPa result register y	$C1C40_H$+ a*2000_H +y*4	PCDSPa	PCDSPa, clk	Kernel Reset	8532
CDSP_DSPa_BNDTRCFG1	CDSPa boundary trigger configuration register 1	$C1C4C_H$+ a*2000_H	PCDSPa	PCDSPa, clk	Kernel Reset	8533
CDSP_DSPa_BNDCFG1	CDSPa boundary mode configuration register 1	$C1C50_H$+ a*2000_H	PCDSPa	PCDSPa, clk	Kernel Reset	8533
CDSP_DSPa_BOUNDST1	CDSPa boundary status register 1	$C1C54_H$+ a*2000_H	PCDSPa	PCDSPa, clk	Kernel Reset	8534
CDSP_DSPa_BOUNDSEL1	CDSPa boundary select register 1	$C1C58_H$+ a*2000_H	PCDSPa, 32	PCDSPa, 32, clk	Kernel Reset	8534
CDSP_DSPa_BNDTRCFG2	CDSPa boundary trigger configuration register 2	$C1C5C_H$+ a*2000_H	PCDSPa	PCDSPa, clk	Kernel Reset	8535
CDSP_DSPa_BNDCFG2	CDSPa boundary mode configuration register 2	$C1C60_H$+ a*2000_H	PCDSPa	PCDSPa, clk	Kernel Reset	8536
CDSP_DSPa_BOUNDST2	CDSPa boundary status register 2	$C1C64_H$+ a*2000_H	PCDSPa	PCDSPa, clk	Kernel Reset	8536
CDSP_DSPa_BOUNDSEL2	CDSPa boundary select register 2	$C1C68_H$+ a*2000_H	PCDSPa, 32	PCDSPa, 32, clk	Kernel Reset	8537

45.8.3 Register overview - ADC_AXI (ascending offset address)

Table 2397 Register overview - ADC_AXI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
TMADCRESx_TSAXIv	TMADCx timestamp register v on LLI (AXI)	20000_H+ x*800_H+ v*4	PTMADCx	BE	Kernel Reset	8537
TMADCRESx_RESAXIv	TMADCx result register v on LLI (AXI)	20040_H+ x*800_H+ v*4	PTMADCx	BE	Kernel Reset	8538
TMADCRESx_MRESAXIv	TMADCx monitor channel v result register on LLI (AXI)	20080_H+ x*800_H+ v*4	PTMADCx	BE	Kernel Reset	8539
DSADCRES_INx_RESM	DSADCx result register main	24000_H+ x*400_H	PDSADCx	BE	Kernel Reset	8539
DSADCRES_INx_RESA	DSADCx result register auxiliary	24004_H+ x*400_H	PDSADCx	BE	Kernel Reset	8540
DSADCRES_EXz_RESM	EXMODz result register main	28000_H+ z*400_H	PEXMODz	BE	Kernel Reset	8540

(table continues...)

Table 2397 (continued) Register overview - ADC_AXI (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
DSADCRES_EXz_RESA	EXMODz result register auxiliary	28004_H+z*400_H	PEXMODz	BE	Kernel Reset	8540
CDSPRESa_RESy	CDSPa result register y	$2B000_H$+a*2000_H+y*4	PCDSPa	BE	Kernel Reset	8541

45.8.4 Clock Control Register

The Clock Control Register CLC allows the programmer to adapt the functionality and power consumption of the module to the requirements of the application.

Register CLC controls the module clock signal and the reactivity to the sleep signal.

CLC Offset address: 00000_H

Clock Control Register Application Reset value: $0000\ 0003_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

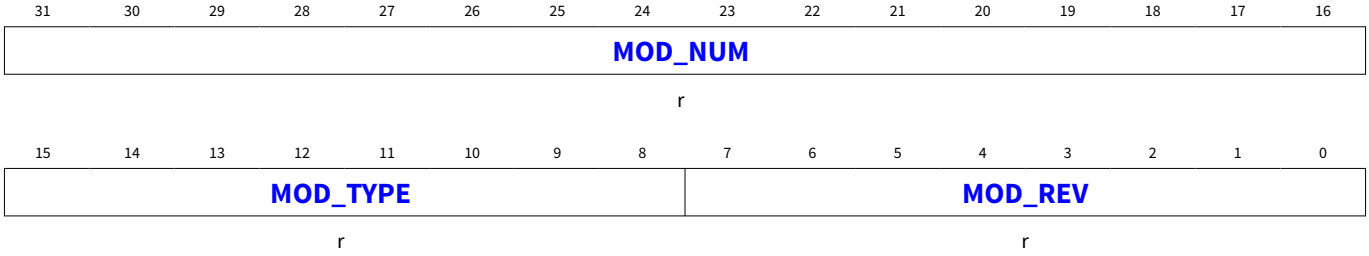
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												EDIS	0	DISS	DISR
r												rw	r	rh	rw

Field	Bits	Type	Description
DISR	0	rw	**Module Disable Request Bit** Used for enable and disable control of the module. 0_B On request: enable the module clock 1_B Off request: stop the module clock
DISS	1	rh	**Module Disable Status Bit** 0_B Module clock is enabled 1_B Off: module is not clocked
EDIS	3	rw	**Sleep Mode Enable Control** Used to control the module's reaction to sleep mode. 0_B Sleep mode request is enabled and functional 1_B Module disregards the sleep mode control signal
0	2, 31:4	r	**Reserved** Read as 0; should be written with 0.

45.8.5 Module Identification Register

ID Offset address: 00008_H

Module Identification Register PowerOn Reset value: $0108\ C002_H$



Field	Bits	Type	Description
MOD_REV	7:0	r	**Module Revision** Indicates the revision number of the implementation.
MOD_TYPE	15:8	r	**Module Type** This internal marker is fixed to $C0_H$.
MOD_NUM	31:16	r	**Module Number** Indicates the module identification number

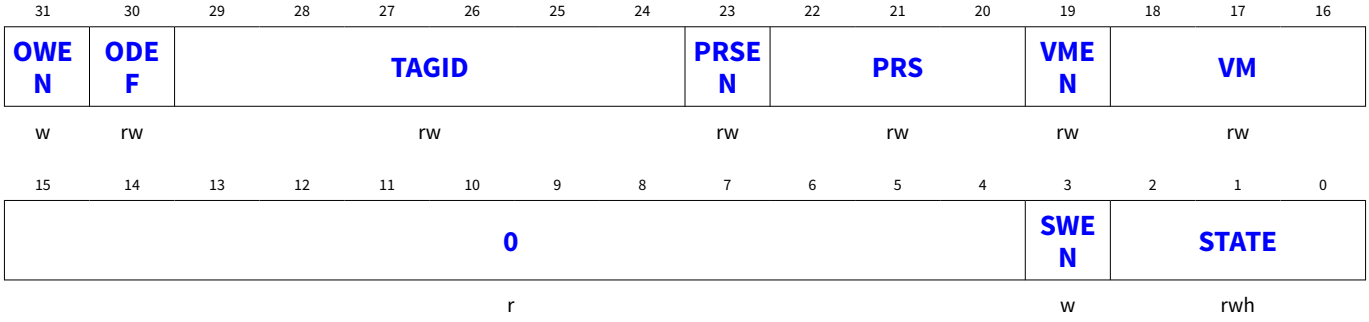
45.8.6 PROT Register Safe Endinit

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner and in some PROT states, the secure master to update the PROT state as described in the PROT mechanism.

The PROTSE register controls lock and unlock of the local Safe Endinit (SE) protected control registers.

PROTSE Offset address: $0000C_H$

PROT Register Safe Endinit Application Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
VM	18:16	rw	**Virtual Machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual Machine definition Enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection Set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.
PRSEN	23	rw	**Protection Set definition Enable for PROT owner** 0_B **Disable**: PRS is not part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; should be written with 0.

Table 2398 Access mode restrictions of PROTSE sorted by descending priority

Mode name	Access mode		Description
write 1 to .**SWEN**	rwh	STATE	
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

45.8.7 Resource allocation register for Global Rsources

RESALLOC_GLOB Offset address: 00010_H

Resource allocation register for Global Rsources Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				CDSPGL				DSINEX				GLOB			
r				rw				rw				rw			

Field	Bits	Type	Description
GLOB	3:0	rw	**APU Asignment for Global parts of ADC**
DSINEX	7:4	rw	**APU Asignment for Global parts of DSADC and EXMOD**
CDSPGL	11:8	rw	**APU Asignment for Global parts of CDSP**
0	31:12	r	**Reserved** Read as 0; should be written with 0.

45.8.8 Resource allocation register for TMADC Resources

RESALLOC_TMADC Offset address: 00014_H

Resource allocation register for TMADC Resources Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0				TMADC6				TMADC5				TMADC4			
r				rw				rw				rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TMADC3				TMADC2				TMADC1				TMADC0			
rw				rw				rw				rw			

Field	Bits	Type	Description
TMADCz (z=0-6)	4*z+3:4*z	rw	**APU Asignment for Corresponsing TMADC**
0	31:28	r	**Reserved** Read as 0; should be written with 0.

45.8.9 Resource allocation register for FCC

RESALLOC_FCCA Offset address: 00018_H

Resource allocation register for FCC Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FCC7				FCC6				FCC5				FCC4			
rw				rw				rw				rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FCC3				FCC2				FCC1				FCC0			
rw				rw				rw				rw			

Field	Bits	Type	Description
FCC0	3:0	rw	**APU Asignment for Corresponsing FCC0**
FCC1	7:4	rw	**APU Asignment for Corresponsing FCC1**
FCC2	11:8	rw	**APU Asignment for Corresponsing FCC2**
FCC3	15:12	rw	**APU Asignment for Corresponsing FCC3**
FCC4	19:16	rw	**APU Asignment for Corresponsing FCC4**
FCC5	23:20	rw	**APU Asignment for Corresponsing FCC5**
FCC6	27:24	rw	**APU Asignment for Corresponsing FCC6**
FCC7	31:28	rw	**APU Asignment for Corresponsing FCC7**

45.8.10 Resource allocation register for FCC

RESALLOC_FCCB Offset address: $0001C_H$

Resource allocation register for FCC Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								FCC9				FCC8			
r								rw				rw			

Field	Bits	Type	Description
FCC8	3:0	rw	**APU Asignment for Corresponsing FCC8**
FCC9	7:4	rw	**APU Asignment for Corresponsing FCC9**
0	31:8	r	**Reserved** Read as 0; should be written with 0.

45.8.11 Resource allocation register for DSADC

RESALLOC_DSADCA Offset address: 00020_H

Resource allocation register for DSADC Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DSADC7				DSADC6				DSADC5				DSADC4			
rw				rw				rw				rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DSADC3				DSADC2				DSADC1				DSADC0			
rw				rw				rw				rw			

Field	Bits	Type	Description
DSADC0	3:0	rw	**APU Asignment for Corresponsing DSADC0**
DSADC1	7:4	rw	**APU Asignment for Corresponsing DSADC1**
DSADC2	11:8	rw	**APU Asignment for Corresponsing DSADC2**
DSADC3	15:12	rw	**APU Asignment for Corresponsing DSADC3**
DSADC4	19:16	rw	**APU Asignment for Corresponsing DSADC4**
DSADC5	23:20	rw	**APU Asignment for Corresponsing DSADC5**
DSADC6	27:24	rw	**APU Asignment for Corresponsing DSADC6**
DSADC7	31:28	rw	**APU Asignment for corresponsing DSADC7**

45.8.12 Resource allocation register for DSADC

RESALLOC_DSADCB Offset address: 00024_H

Resource allocation register for DSADC Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								DSADC9				DSADC8			
r								rw				rw			

Field	Bits	Type	Description
DSADC8	3:0	rw	**APU Asignment for Corresponsing DSADC8**
DSADC9	7:4	rw	**APU Asignment for Corresponsing DSADC9**
0	31:8	r	**Reserved** Read as 0; should be written with 0.

45.8.13 Resource allocation register for EXMOD

RESALLOC_EXMOD Offset address: 00028_H

Resource allocation register for EXMOD Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EXMOD7				EXMOD6				EXMOD5				EXMOD4			
rw				rw				rw				rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EXMOD3				EXMOD2				EXMOD1				EXMOD0			
rw				rw				rw				rw			

Field	Bits	Type	Description
EXMOD0	3:0	rw	**APU Asignment for Corresponsing EXMOD0**
EXMOD1	7:4	rw	**APU Asignment for Corresponsing EXMOD1**
EXMOD2	11:8	rw	**APU Asignment for Corresponsing EXMOD2**
EXMOD3	15:12	rw	**APU Asignment for Corresponsing EXMOD3**
EXMOD4	19:16	rw	**APU Asignment for Corresponsing EXMOD3**
EXMOD5	23:20	rw	**APU Asignment for Corresponsing EXMOD3**
EXMOD6	27:24	rw	**APU Asignment for Corresponsing EXMOD3**
EXMOD7	31:28	rw	**APU Asignment for Corresponsing EXMOD3**

45.8.14 Resource allocation register for CDSP

RESALLOC_CDSPA Offset address: $0002C_H$

Resource allocation register for CDSP Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CDSP7				CDSP6				CDSP5				CDSP4			
rw				rw				rw				rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CDSP3				CDSP2				CDSP1				CDSP0			
rw				rw				rw				rw			

Field	Bits	Type	Description
CDSP0	3:0	rw	**APU Asignment for Corresponsing CDSP 0**
CDSP1	7:4	rw	**APU Asignment for Corresponsing CDSP1**
CDSP2	11:8	rw	**APU Asignment for Corresponsing CDSP2**
CDSP3	15:12	rw	**APU Asignment for Corresponsing CDSP3**
CDSP4	19:16	rw	**APU Asignment for Corresponsing CDSP4**
CDSP5	23:20	rw	**APU Asignment for Corresponsing CDSP5**
CDSP6	27:24	rw	**APU Asignment for Corresponsing CDSP6**
CDSP7	31:28	rw	**APU Asignment for corresponsing CDSP7**

45.8.15 Resource allocation register for CDSP

RESALLOC_CDSPB Offset address: 00030_H

Resource allocation register for CDSP Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0				0				0				0			
r				r				r				r			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CDSP11				CDSP10				CDSP9				CDSP8			
rw				rw				rw				rw			

Field	Bits	Type	Description
CDSP8	3:0	rw	**APU Asignment for Corresponsing CDSP 8**
CDSP9	7:4	rw	**APU Asignment for Corresponsing CDSP9**
CDSP10	11:8	rw	**APU Asignment for Corresponsing CDSP10**
CDSP11	15:12	rw	**APU Asignment for Corresponsing CDSP11**
0	19:16, 23:20, 27:24, 31:28	r	**Reserved** Read as 0; should be written with 0.

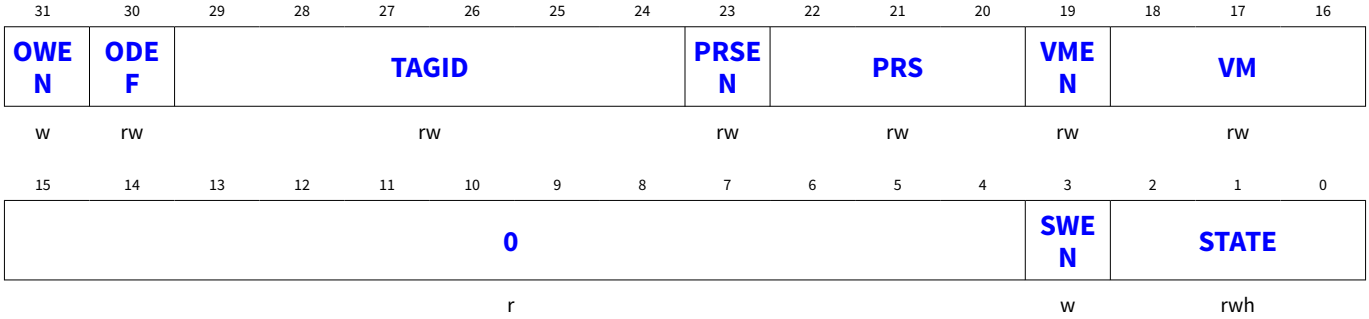
45.8.16 APUx PROT Register Endinit

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner and in some PROT states, the secure master to update the PROT state as described in the PROT mechanism.

The PROTE register controls lock and unlock of the local Endinit (E) protected control registers.

PROTEx (x=0-15) Offset address: 00080_H+x*4

APUx PROT Register Endinit Application Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
VM	18:16	rw	**Virtual Machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual Machine definition Enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection Set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.
PRSEN	23	rw	**Protection Set definition Enable for PROT owner** 0_B **Disable**: PRS is not part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner

(table continues...)

(continued)

Field	Bits	Type	Description
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; should be written with 0.

Table 2399 Access mode restrictions of PROTEx (x=0-15) sorted by descending priority

Mode name	Access mode		Description
write 1 to .**SWEN**	rwh	STATE	
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

45.8.17 Write access enable register A, APU y

This register provides controls to enable or disable write accesses for transactions with TAG IDs corresponding to the numbered bits.

ACCENy_WRA (y=0-15) Offset address: $000C0_H+y*20_H$

Write access enable register A, APU y Application Reset value: $1000\ 0003_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	**EN30**	**EN29**	**EN28**	**EN27**	**EN26**	**EN25**	**EN24**	**EN23**	**EN22**	**EN21**	**EN20**	**EN19**	**EN18**	**EN17**	**EN16**
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	**EN14**	**EN13**	**EN12**	**EN11**	**EN10**	**EN09**	**EN08**	**EN07**	**EN06**	**EN05**	**EN04**	**EN03**	**EN02**	**EN01**	**EN00**
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q 0_B Disabled for write access 1_B Enabled for write access

45.8.18 Write access enable register B, APU y

This register provides controls to enable or disable write accesses for transactions with TAG IDs corresponding to the numbered bits.

ACCENy_WRB (y=0-15) Offset address: $000C4_H+y*20_H$

Write access enable register B, APU y Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Write access enable for TAG-ID q** This bit enables write access to the access protected region for transactions with the TAG-ID q 0_B Disabled for write access 1_B Enabled for write access
0	31:8	r	**Reserved** Read as 0; should be written with 0.

45.8.19 Read access enable register A, APU y

This register provides controls to enable or disable read accesses for transactions with TAG IDs corresponding to the numbered bits.

ACCENy_RDA (y=0-15) Offset address: $000C8_H+y*20_H$

Read access enable register A, APU y Application Reset value: $FFFF\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q 0_B Disabled for read access 1_B Enabled for read access

45.8.20 Read access enable register B, APU y

This register provides controls to enable or disable read accesses for transactions with TAG IDs corresponding to the numbered bits.

ACCENy_RDB (y=0-15)

Read access enable register B, APU y

Offset address: $000CC_H+y*20_H$

Application Reset value: $0000\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Read access enable for TAG-ID q** This bit enables read access to the access protected region for transactions with the TAG-ID q 0_B Disabled for read access 1_B Enabled for read access
0	31:8	r	**Reserved** Read as 0; should be written with 0.

45.8.21 VM access enable register, APU y

This register provides separate controls to enable or disable read and write accesses to registers for transactions with VM corresponding to the numbered bits. Bits WR0 to WR7 control write access and bits RD0 to RD7 control read access.

ACCENy_VM (y=0-15)

VM access enable register, APU y

Offset address: $000D0_H+y*20_H$

Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for VM ID q** This bit enables read access to the access protected region for transactions with the VM ID q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for VM ID q** This bit enables write access to the access protected region for transactions with the VM ID q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

45.8.22 PRS access enable register, APU y

This register provides separate controls to enable or disable read and write accesses to registers for transactions with PRS corresponding to the numbered bits. Bits WR0 to WR7 control write access and bits RD0 to RD7 control read access.

ACCENy_PRS (y=0-15) Offset address: $000D4_H+y*20_H$

PRS access enable register, APU y Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for PRS q** This bit enables read access to the access protected region for transactions with the PRS q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for PRS q** This bit enables write access to the access protected region for transactions with the PRS q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

45.8.23 Reset Control Register A, APU i

RSTi_CTRLA (i=0-15) Offset address: $002C0_H$+i*12

Reset Control Register A, APU i Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				GRSTEN3	GRSTEN2	GRSTEN1	GRSTEN0	0							KRST
r				rw	rw	rw	rw	r							rwh

Field	Bits	Type	Description
KRST	0	rwh	**Kernel Reset** Request a kernel reset. The requested reset is executed if CTRLA.KRST=1 and CTRLB.KRST=1. KRST is cleared after the kernel reset was executed. 0_B No action 1_B A kernel reset was requested
GRSTENx (x=0-3)	x+8	rw	**Enable for Global Module Reset Group x** 0_B Global module reset group x does not have any effect 1_B Global module reset group x results in a kernel reset
0	7:1, 31:12	r	**Reserved** Read as 0; should be written with 0.

45.8.24 Reset Control Register B, APU i

RSTi_CTRLB (i=0-15) Offset address: $002C4_H$+i*12

Reset Control Register B, APU i Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
STATCLR								0							
w								r							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															KRST
r															rwh

Field	Bits	Type	Description
KRST	0	rwh	**Kernel Reset** Request a kernel reset. The requested reset is executed if CTRLA.KRST=1 and CTRLB.KRST=1. KRST is cleared after the kernel reset was executed. 0_B No action 1_B A kernel reset was requested
STATCLR	31	w	**Kernel Reset Status Clear** Clears all status bits in RST_STAT when 1 is written. Read will return 0. 0_B No action 1_B Write with ´1´ clears bits STAT.GRSTx and bit STAT.KRST .
0	30:1	r	**Reserved** Read as 0; should be written with 0.

45.8.25 Reset Status Register, APU i

The reset status register contains the status bits for kernel reset (KRST) and the global module reset groups (GRSTx).

RSTi_STAT (i=0-15) Offset address: $002C8_H+i*12$

Reset Status Register, APU i Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				GRST3	GRST2	GRST1	GRST0	0							KRST
r				rh	rh	rh	rh	r							rh

Field	Bits	Type	Description
KRST	0	rh	**Kernel Reset Status** Indicates an executed kernel reset. RST_STAT.KRST is set after the execution of a kernel reset in the same clock cycle in which the reset bits are cleared. Clear KRST by setting bit STATCLR in register RST_CTRLB.
GRSTx (x=0-3)	x+8	rh	**Status for Global Module Reset Group x** 0_B Reset was not triggered by Global Reset Group x 1_B Reset was triggered by Global Reset Group x
0	7:1, 31:12	r	**Reserved** Read as 0.

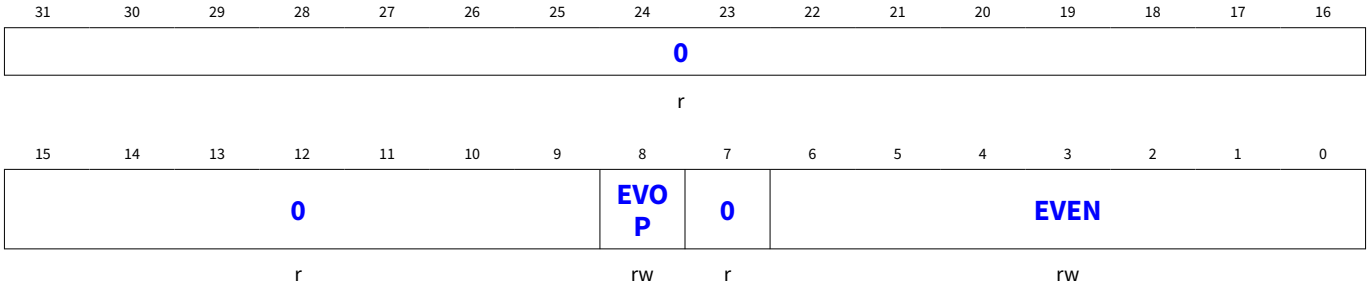
45.8.26 ADC global service request i configuration register

Global service requests are used for generating result events to IR when a set of analog channles have finished converting and results of these channels are available. The set of analog channels in this case are distributed across different TMADC instances. Global service requests are limited to TMADC channels.

Only the service request configured with SRCFG6 register in TMADCx is a valid input to the global service request.

GLSRCFGi (i=0-3) Offset address: 00800_H+i*4

ADC global service request i configuration register Kernel Reset value: $0000\ 0000_H$



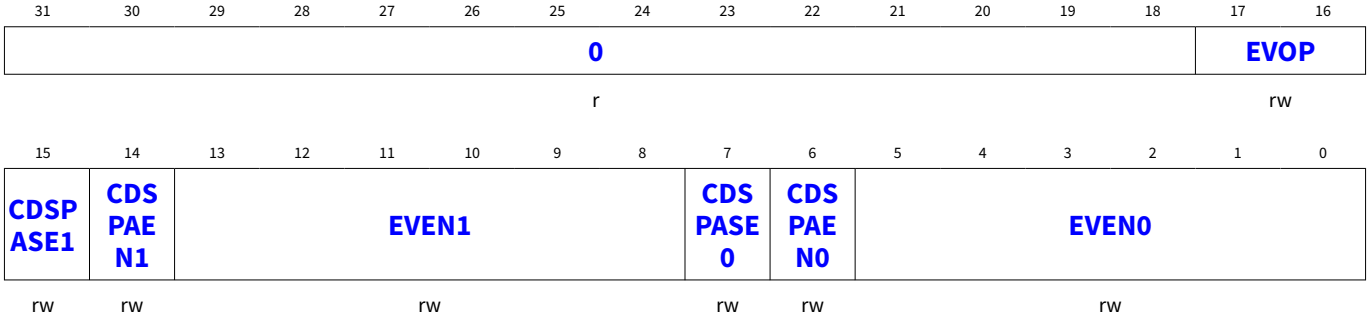
Field	Bits	Type	Description
EVEN	6:0	rw	**Event Enable** Each bit in this bitfield, if set, enables the event on SR6 for the corressponding TMADCx to be an input to the corressponding global service request 0_B Event disabled 1_B Event enabled
EVOP	8	rw	**Event Operation** This bit defines what logical operation shall be executed and under which situation global service request is generated 0_B OR logic operation 1_B AND logic operation
0	7, 31:9	r	**Reserved** Read as 0; should be written with 0

45.8.27 ADC global alarm i configuration register

Global alarms are used for generating alarms to SMU and also same alarms go as interrupts to IR. They can be generated combining 2 bounday flags from any converter.

GLALCFGi (i=0-7) Offset address: 00810_H+i*4

ADC global alarm i configuration register Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
EVEN0	5:0	rw	**Event Enable** 1st input to alarm generation. Select one of the 64 boundary flags. BFL0 - BFL15 are from TMADC. 2 BFLs for each TMADC BFL16-BFL25 are from FC. 1 BFL from each FC BFL26 - BFL35 are from DSINT. 1 BFL from each DSINT BFL36-BFL43 are from DSEXT. 1 BFL from each DSEXT BFL44-BFL61 are from CDSPs. 1 BFL from each CDSP BFL62 and BFL63 are reserved and not connected
CDSPAEN0	6	rw	**Alternative CDSP boundary flags mapped to EVEN0** Setting this bit maps the alternative BFL source configured by CDSP_DSPa_BNDCFG1 or CDSP_DSPa_BNDCFG2 to BFL44-BFL61 (selected by EVEN0).
CDSPASE0	7	rw	**Select of the alternative CDSP BFL source for EVEN0** 0_B Alternative BFL source configured by CDSP_DSPa_BNDCFG1 selected 1_B Alternative BFL source configured by CDSP_DSPa_BNDCFG2 selected
EVEN1	13:8	rw	**Event Enable** 2nd input to alarm generation. Select one of the 64 boundary flags. BFL0 - BFL15 are from TMADC. 2 BFLs for each TMADC BFL16-BFL25 are from FC. 1 BFL from each FC BFL26 - BFL35 are from DSINT. 1 BFL from each DSINT BFL36-BFL43 are from DSEXT. 1 BFL from each DSEXT BFL44-BFL61 are from CDSPs. 1 BFL from each CDSP BFL62 and BFL63 are reserved and not connected
CDSPAEN1	14	rw	**Alternative CDSP boundary flags mapped to EVEN1** Setting this bit maps the alternative BFL source configured by CDSP_DSPa_BNDCFG1 or CDSP_DSPa_BNDCFG2 to BFL44-BFL61 (selected by EVEN1).
CDSPASE1	15	rw	**Select of the alternative CDSP BFL source for EVEN1** 0_B Alternative BFL source configured by CDSP_DSPa_BNDCFG1 selected 1_B Alternative BFL source configured by CDSP_DSPa_BNDCFG2 selected

(table continues...)

(continued)

Field	Bits	Type	Description
EVOP	17:16	rw	**Event Operation** This bit defines what logical operation shall be executed and under which situation global service request is generated 00_B Alarm disabled 01_B OR logical operation The input to the OR gate are the input selected by EVENT0 and EVENT1. As soon as first high input comes then the alarm is generated. Every time there is a high input on EVENT0 or EVENT1 an alarm is generated 10_B AND logical operation The input to the AND gate are the input selected by EVENT0 and EVENT1. If EVENT0 and EVENT1 are coming at different time, the events remain latched until the alarm is generated 11_B Reserved
0	31:18	r	**Reserved** Read as 0; should be written with 0

45.8.28 ADC global alarm set register

By writing into this register, SW can generate corresponding alarms

GLALSET Offset address: 00830_H

ADC global alarm set register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								ALSET							
r								w							

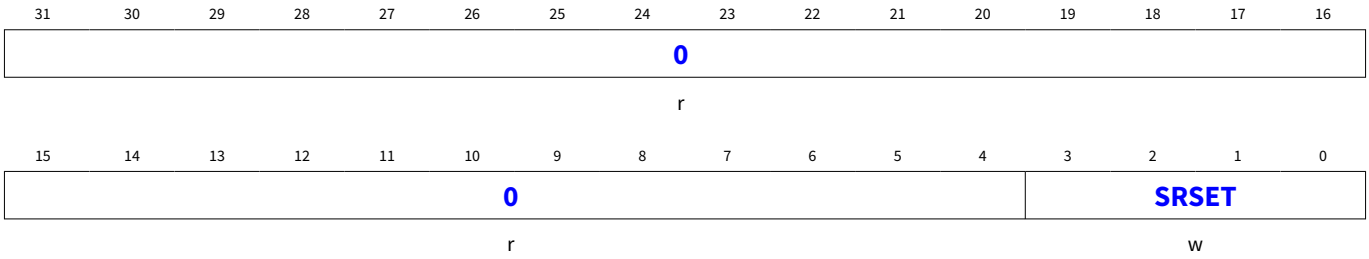
Field	Bits	Type	Description
ALSET	7:0	w	**Alarm set by SW write** By writing the corresponding bit, an alarm can be generated any time. After generating the alarm, the bit is automatically cleared by hardware.
0	31:8	r	**Reserved** Read as 0; should be written with 0

45.8.29 ADC global service request set register

By writing into this register, SW can generate corresponding service request

GLSRSET Offset address: 00834_H

ADC global service request set register Kernel Reset value: $0000\ 0000_H$

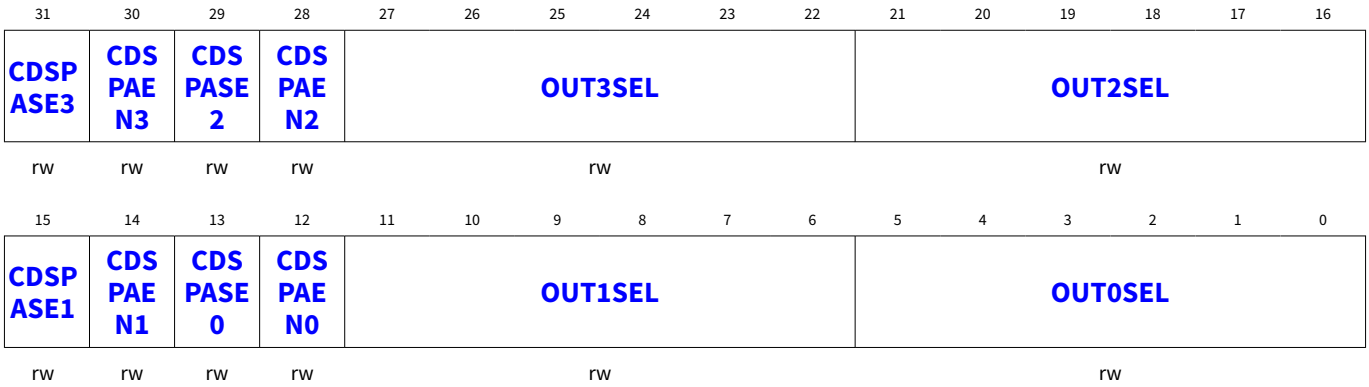


Field	Bits	Type	Description
SRSET	3:0	w	**Service request set by SW write - ALSET** By setting the individual bits (3:0) the corresponding global service request (3:0) can be generated any time. The bit is automatically cleared by hardware.
0	31:4	r	**Reserved** Read as 0; should be written with 0

45.8.30 ADC to eGTM boundary flag select register, cluster x

ETBFxSEL (x=0-2) Offset address: 00838_H+x*4

ADC to eGTM boundary flag select register, cluster x Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
OUTySEL (y=0-1)	6*y+5:6*y	rw	**Boundary flag y output select** 00_H Boundary flag 0 selected 01_H Boundary flag 1 selected 02_H Boundary flag 2 selected … $3F_H$ Boundary flag 63 selected
CDSPAEN0	12	rw	**Alternative CDSP boundary flags mapped to OUTSEL0** Setting this bit maps the alternative BFL source configured by CDSP_DSPa_BNDCFG1 or CDSP_DSPa_BNDCFG2 to BFL44-BFL61 (selected by OUTSEL0).

(table continues...)

(continued)

Field	Bits	Type	Description
CDSPASE0	13	rw	**Select of the alternative CDSP BFL source for OUTSEL0** 0_B Alternative BFL source configured by CDSP_DSPa_BNDCFG1 selected 1_B Alternative BFL source configured by CDSP_DSPa_BNDCFG2 selected
CDSPAEN1	14	rw	**Alternative CDSP boundary flags mapped to OUTSEL0** Setting this bit maps the alternative BFL source configured by CDSP_DSPa_BNDCFG1 or CDSP_DSPa_BNDCFG2 to BFL44-BFL61 (selected by OUTSEL0).
CDSPASE1	15	rw	**Select of the alternative CDSP BFL source for OUTSEL0** 0_B Alternative BFL source configured by CDSP_DSPa_BNDCFG1 selected 1_B Alternative BFL source configured by CDSP_DSPa_BNDCFG2 selected
OUTySEL (y=2-3)	6*y+9:6*y+4	rw	**Boundary flag y output select** 00_H Boundary flag 0 selected 01_H Boundary flag 1 selected 02_H Boundary flag 2 selected … $3F_H$ Boundary flag 63 selected
CDSPAEN2	28	rw	**Alternative CDSP boundary flags mapped to OUTSEL0** Setting this bit maps the alternative BFL source configured by CDSP_DSPa_BNDCFG1 or CDSP_DSPa_BNDCFG2 to BFL44-BFL61 (selected by OUTSEL2).
CDSPASE2	29	rw	**Select of the alternative CDSP BFL source for OUTSEL2** 0_B Alternative BFL source configured by CDSP_DSPa_BNDCFG1 selected 1_B Alternative BFL source configured by CDSP_DSPa_BNDCFG2 selected
CDSPAEN3	30	rw	**Alternative CDSP boundary flags mapped to OUTSEL0** Setting this bit maps the alternative BFL source configured by CDSP_DSPa_BNDCFG1 or CDSP_DSPa_BNDCFG2 to BFL44-BFL61 (selected by OUTSEL3).
CDSPASE3	31	rw	**Select of the alternative CDSP BFL source for OUTSEL3** 0_B Alternative BFL source configured by CDSP_DSPa_BNDCFG1 selected 1_B Alternative BFL source configured by CDSP_DSPa_BNDCFG2 selected

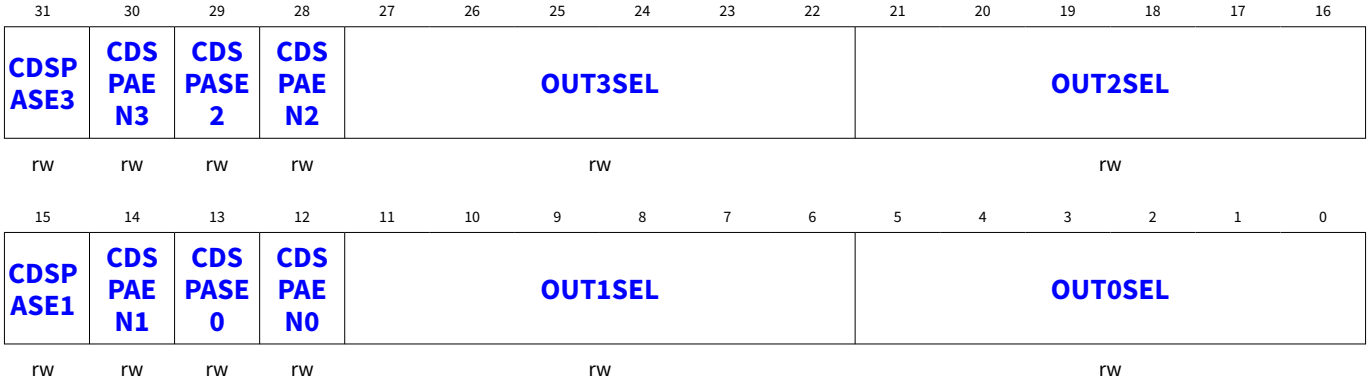
45.8.31 ADC to GTM boundary flag select register, cluster x

GTBFxSEL (x=0-7)

ADC to GTM boundary flag select register, cluster x

Offset address: 00844_H+x*4

Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
OUTySEL (y=0-1)	6*y+5:6*y	rw	**Boundary flag y output select** 00_H Boundary flag 0 selected 01_H Boundary flag 1 selected 02_H Boundary flag 2 selected … $3F_H$ Boundary flag 63 selected
CDSPAEN0	12	rw	**Alternative CDSP boundary flags mapped to OUTSEL0** Setting this bit maps the alternative BFL source configured by CDSP_DSPa_BNDCFG1 or CDSP_DSPa_BNDCFG2 to BFL44-BFL61 (selected by OUTSEL0).
CDSPASE0	13	rw	**Select of the alternative CDSP BFL source for OUTSEL0** 0_B Alternative BFL source configured by CDSP_DSPa_BNDCFG1 selected 1_B Alternative BFL source configured by CDSP_DSPa_BNDCFG2 selected
CDSPAEN1	14	rw	**Alternative CDSP boundary flags mapped to OUTSEL0** Setting this bit maps the alternative BFL source configured by CDSP_DSPa_BNDCFG1 or CDSP_DSPa_BNDCFG2 to BFL44-BFL61 (selected by OUTSEL0).
CDSPASE1	15	rw	**Select of the alternative CDSP BFL source for OUTSEL0** 0_B Alternative BFL source configured by CDSP_DSPa_BNDCFG1 selected 1_B Alternative BFL source configured by CDSP_DSPa_BNDCFG2 selected
OUTySEL (y=2-3)	6*y+9:6*y+4	rw	**Boundary flag y output select** 00_H Boundary flag 0 selected 01_H Boundary flag 1 selected 02_H Boundary flag 2 selected … $3F_H$ Boundary flag 63 selected

(table continues...)

(continued)

Field	Bits	Type	Description
CDSPAEN2	28	rw	**Alternative CDSP boundary flags mapped to OUTSEL0** Setting this bit maps the alternative BFL source configured by CDSP_DSPa_BNDCFG1 or CDSP_DSPa_BNDCFG2 to BFL44-BFL61 (selected by OUTSEL2).
CDSPASE2	29	rw	**Select of the alternative CDSP BFL source for OUTSEL2** 0_B Alternative BFL source configured by CDSP_DSPa_BNDCFG1 selected 1_B Alternative BFL source configured by CDSP_DSPa_BNDCFG2 selected
CDSPAEN3	30	rw	**Alternative CDSP boundary flags mapped to OUTSEL0** Setting this bit maps the alternative BFL source configured by CDSP_DSPa_BNDCFG1 or CDSP_DSPa_BNDCFG2 to BFL44-BFL61 (selected by OUTSEL3).
CDSPASE3	31	rw	**Select of the alternative CDSP BFL source for OUTSEL3** 0_B Alternative BFL source configured by CDSP_DSPa_BNDCFG1 selected 1_B Alternative BFL source configured by CDSP_DSPa_BNDCFG2 selected

45.8.32 ADC to GTM service request select register, cluster x

GTSRxSEL (x=0-9) Offset address: 00864_H+x*4

ADC to GTM service request select register, cluster x Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0												SR3EN	SR2EN	SR1EN	SR0EN
r												rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SR3SEL				SR2SEL				SR1SEL				SR0SEL			
rw				rw				rw				rw			

Field	Bits	Type	Description
SRySEL (y=0-3)	4*y+3:4*y	rw	**Service request y select for GTM cluster x** Selects the availability of new result on ADI which triggers an interrupt to MCS (not through IR). There are up to 16 result from TMADC mapped to GTM. Each GTM cluster with MCS has 4 interrupts connected to ADC. SR0 is connected to interrupt 4, SR1 to interrupt 5, SR2 to interrupt 6 and SR3 to interrupt 7 for the MCS. There are 16 sources of interrupts, 2 from each TMADC instance. 0_H TMADC 0 result register 14 triggers the SR 1_H TMADC 0 result register 15 triggers the SR 2_H TMADC 1 result register 14 triggers the SR 3_H TMADC 1 result register 15 triggers the SR 4_H TMADC 2 result register 14 triggers the SR 5_H TMADC 2 result register 15 triggers the SR 6_H TMADC 3 result register 14 triggers the SR 7_H TMADC 3 result register 15 triggers the SR 8_H TMADC 4 result register 14 triggers the SR 9_H TMADC 4 result register 15 triggers the SR A_H TMADC 5 result register 14 triggers the SR B_H TMADC 5 result register 15 triggers the SR C_H TMADC 6 result register 14 triggers the SR D_H TMADC 6 result register 15 triggers the SR E_H TMADC 7 result register 14 triggers the SR F_H TMADC 7 result register 15 triggers the SR
SRyEN (y=0-3)	y+16	rw	**Service Request Enable - SREN** Enables the corresponding Service Request
0	31:20	r	**Reserved** Read as 0; should be written with 0

45.8.33 ADC pull down diagnostics configuration register

PDDCFG Offset address: $0088C_H$

ADC pull down diagnostics configuration register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

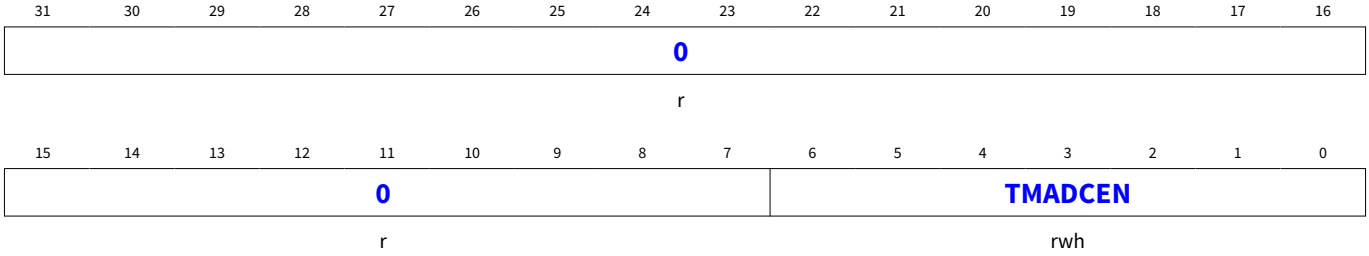
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EN7	EN6	EN5	EN4	EN3	EN2	EN1	EN0
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENp (p=0-7)	p	rw	**Enable** 0_B PDD disabled 1_B PDD enabled
0	31:8	r	**Reserved** Read as 0; should be written with 0

45.8.34 Clock enable for TMADC

CLKEN_TMADC Offset address: 00890_H

Clock enable for TMADC Kernel Reset value: $0000\ 0000_H$

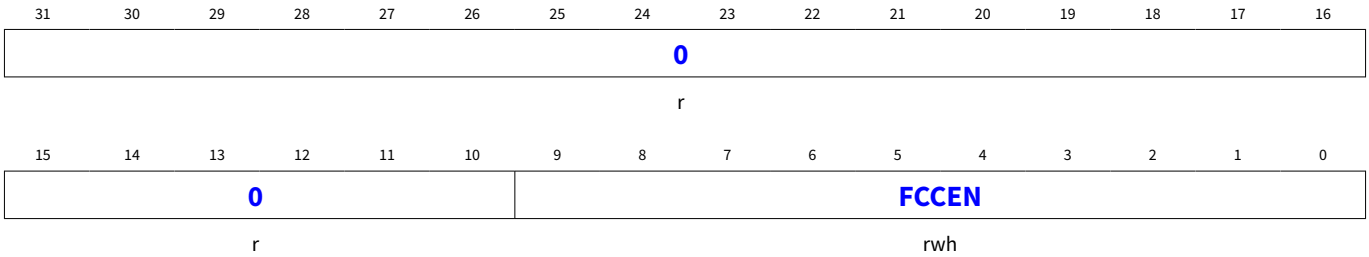


Field	Bits	Type	Description
TMADCEN	6:0	rwh	**Clock enable for TMADC** Each bit controls the clock for the corresponding TMADC channel. E.g: bit0 for TMADC0. 0 - No action 1 - Clock enable for TMADC instance
0	31:7	r	**Reserved** Read as 0; should be written with 0

45.8.35 Clock enable for FCC

CLKEN_FCC Offset address: 00894_H

Clock enable for FCC Kernel Reset value: $0000\ 0000_H$



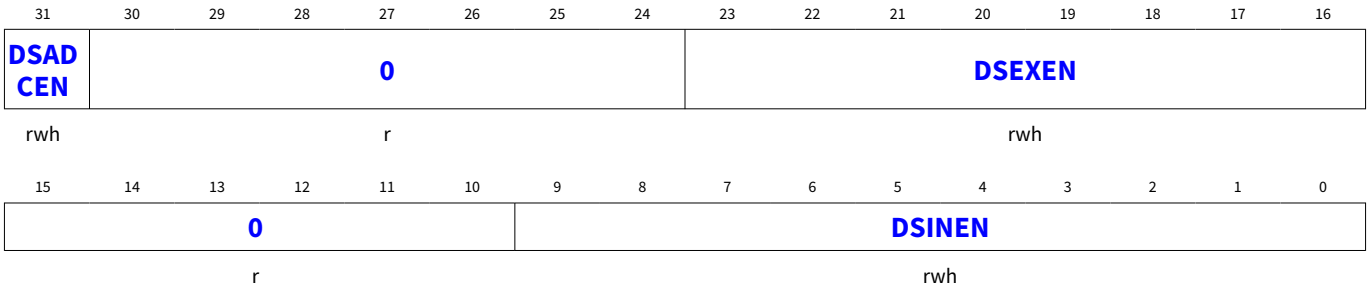
Field	Bits	Type	Description
FCCEN	9:0	rwh	**Clock enable for FCC** Each bit controls the clock for the corresponding FCC channel. E.g: bit0 for FCC0. 0 - No action 1 - Clock enable for FCC instance
0	31:10	r	**Reserved** Read as 0; should be written with 0

45.8.36 Clock enable for DSADC and EXMOD

- No action
- Clock enable for DSADC and EXMOD instance

CLKEN_DSINEX Offset address: 00898_H

Clock enable for DSADC and EXMOD Kernel Reset value: $0000\ 0000_H$



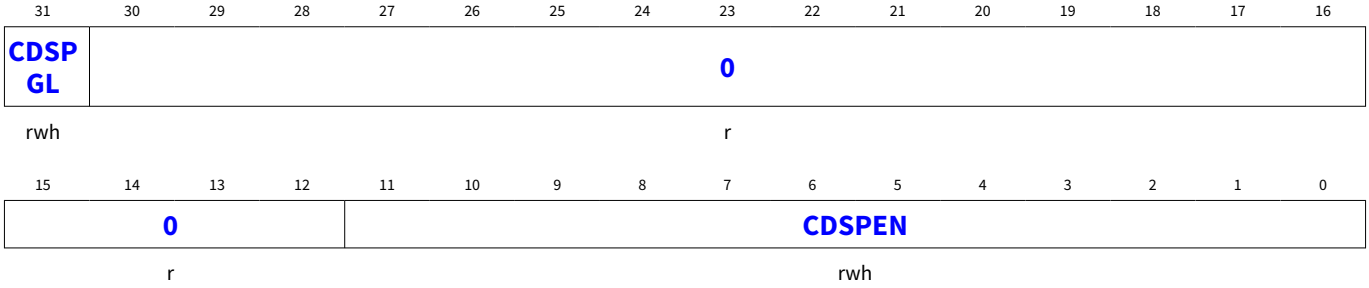
Field	Bits	Type	Description
DSINEN	9:0	rwh	**DSADC Clock Enable** Each bit controls the clock for the corresponding DSADC channel. E.g: bit0 for DSADC0.
DSEXEN	23:16	rwh	**EXMOD Clock Enable** Each bit controls the clock for the corresponding EXMOD channel. E.g: LSB for EXMOD0.
DSADCEN	31	rwh	**Enables the global part of DSADC and EXMOD** If the user intends to use DSADC or EXMOD channel, then this bit must be set to 1.
0	15:10, 30:24	r	**Reserved** Read as 0; should be written with 0

45.8.37 Clock enable for CDSP

- No action
- Clock enable for CDSP instance

CLKEN_CDSP Offset address: $0089C_H$

Clock enable for CDSP Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
CDSPEN	11:0	rwh	**Clock enable for corressponding CDSP** When set to '1' clock is enabled for the corresponding CDSP
CDSPGL	31	rwh	**Enables the global part of CDSP** If the user intends to use any CDSP, then this bit shall be set to 1
0	30:12	r	**Reserved** Read as 0; should be written with 0

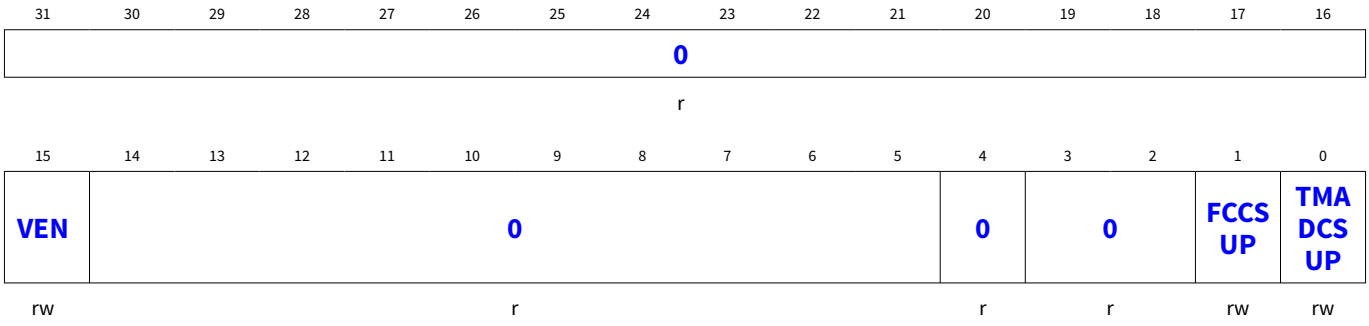
45.8.38 Supply level selection for ADC

SUPLLEV

Supply level selection for ADC

Offset address: $008A0_H$

Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
TMADCSUP	0	rw	**TMADC supply level** With this bit-field operating frequency of TMADC has to be set in relation to VDDM range 0_B 80 MHz (4.5 V <= VDDM < = 5.5 V) 1_B 40 MHz (VDDM < 4.5 V)
FCCSUP	1	rw	**FCC supply level** With this bit-field operating frequency of FCC has to be set in relation to VDDM range 0_B 80 MHz (3.7 V <= VDDM < = 5.5 V) 1_B 40 MHz (VDDM < 3.7 V)

(table continues...)

(continued)

Field	Bits	Type	Description
VEN	15	rw	**High ohmic reference voltage enable** With this bit a reference voltage (VDDMDIV) of 3/4VDDM is enabled to the input analog MUX of monitoring sampler 1 of TMADC 0. This voltage can be used to calculate on-chip clock jitter.
0	3:2, 4, 14:5, 31:16	r	**Reserved** Read as 0; should be written with 0

45.8.39 TMADC supervision module register

TMSMOD Offset address: $008A4_H$

TMADC supervision module register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0													SEL		
r													rw		

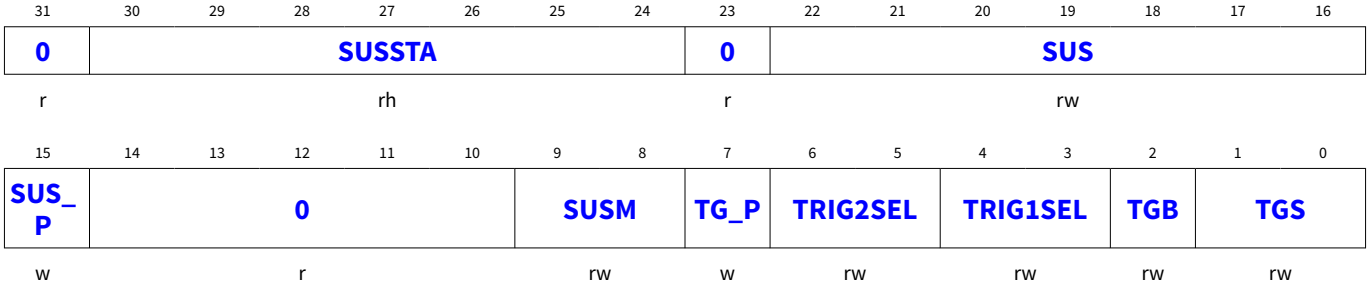
Field	Bits	Type	Description
SEL	2:0	rw	**Select** This bitfield selects the TMADC module that drives the output supervision signal that is configured in bitfield OSSEL. 0_H TMADC0 1_H TMADC1 ... 7_H TMADC7
0	31:3	r	**Reserved** Read as 0; should be written with 0

45.8.40 OCDS control and status register for TMADC

This OCDS control and status register controls the debug and trace behavior by selecting suspend modes and OTGB trigger sets. When OCDS is disabled the suspend control is ineffective. This register only controls all instances of TMADC

OCSTM Offset address: $008A8_H$

OCDS control and status register for TMADC Debug Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
TGS	1:0	rw	**Trigger set for OTGB0 and OTGB1** 00_B No trigger set output 01_B Trigger set 1 Start of conversion trigger signal of TMADC input channels. For selecting the right TMADC, refer to TRIG1SEL 10_B Trigger set 2 Start of conversion trigger signal of TMADC input channels. For selecting the right TMADC, refer to TRIG2SEL 11_B Reserved
TGB	2	rw	**OTGB0 and OTGB1 bus select** 0_B Trigger set is output on OTGB0 1_B Trigger set is output on OTGB1
TRIG1SEL	4:3	rw	**Trigger set 1 select** These bits select which TMADC instance trigger signals are put on trigger set 1 00_B TMADC1 01_B TMADC3 10_B TMADC5 11_B TMADC7
TRIG2SEL	6:5	rw	**Trigger set 2 select** These bits select which TMADC instance trigger signals are put on trigger set 2 00_B TMADC0 01_B TMADC2 10_B TMADC4 11_B TMADC6
TG_P	7	w	**TGS, TGB write protection** TGS,TGB TRIG1SEL and TRIG2SEL are only written when TG_P is 1, otherwise unchanged. Read as 0.

(table continues...)

(continued)

Field	Bits	Type	Description
SUSM	9:8	rw	**OCDS suspend control mode - SUS** These bits select the suspend mode for the module 00_B Will not suspend 01_B Hard suspend Clocks will be disabled immediately 10_B Reserved, not to be used Reserved, will not suspend 11_B Soft suspend mode All the channels in hold state for the TMADC will be converted and then module will go into suspend
SUS_P	15	w	**SUS write protection** SUS and SUSM are only written when SUS_P is 1, otherwise unchanged. Read as 0.
SUS	22:16	rw	**OCDS suspend control** Suspends the corresponding TMADC. 0 - Module not suspended, 1 - Module Suspended. Each bit controls a TMADC, with LSB for TMADC0 and MSB for the maximum number of TMADC in device.
SUSSTA	30:24	rh	**Suspend state** Suspend State of TMADC. 0 - Module not suspended. 1 - Module Suspended. Each bit gives status of a TMADC, with LSB for TMADC0 and MSB for the maximum number of TMADC in device.
0	14:10, 23, 31	r	**Reserved** Read as 0; should be written with 0.

Table 2400 Access mode restrictions of OCSTM sorted by descending priority

Mode name	Access mode		Description
write 1 to .**TG_P**	rw	TGB, TGS, TRIG1SEL, TRIG2SEL	Set TG_P during write access
write 1 to .**SUS_P**	rw	SUS, SUSM	Set SUS_P during write access
(default)	r	SUS, SUSM, TGB, TGS, TRIG1SEL, TRIG2SEL	

45.8.41 TMADCx module configuration register

TMADCx_MODCFG (x=0-6) Offset address: $20000_H+x*800_H$

TMADCx module configuration register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				OSSEL		TSCRUN	0					BND1UPD	BND0UPD	RUN	SUCAL
r				rwh		rw	r					rw	rw	rwh	w

Field	Bits	Type	Description
SUCAL	0	w	**Start-Up Calibration** 0_B No action 1_B Request Start-up Calibration
RUN	1	rwh	**Run** 0_B No action 1_B Initiate RUN state transition
BNDnUPD (n=0-1)	n+2	rw	**Boundary values update, boundary register n** 0_B Active update of boundary values 1_B Shadow update of boundary values
TSCRUN	9	rw	**Time-stamp counter run control** 0_B Time-stamp counter off 1_B Time-stamp counter starts counting
OSSEL	11:10	rwh	**Output supervision signal select** 0_H No connection 1_H Core-0 comparator supply voltage (VDDK0) 2_H Core-1 comparator supply voltage (VDDK1) 3_H VSSM
0	8:4, 31:12	r	**Reserved** Read as 0; should be written with 0

45.8.42 TMADCx EMUX configuration register

TMADCx_EMUXCFG (x=0-6) Offset address: $20004_H+x*800_H$

TMADCx EMUX configuration register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			CISEL3	CISEL2	CISEL1	CISEL0	CS	0				CHCNT			
r			rw	rw	rw	rw	rw	r				rw			

Field	Bits	Type	Description
CHCNT	3:0	rw	**External multiplexer channel count**
CS	8	rw	**External multiplexer coding scheme** 0_B EMUX channel selection lines are binary coded 1_B EMUX channel selection lines are gray coded
CISELn (n=0-3)	n+9	rw	**External multiplxer control interface n select** 0_B Control interface disabled 1_B Control interface enabled
0	7:4, 31:13	r	**Reserved** Read as 0; should be written with 0

45.8.43 TMADCx software trigger configuration register

This register configures which TMADC channel shall be triggered for the end of sampling.

TMADCx_SWTRCFG (x=0-6) Offset address: $2000C_H+x*800_H$

TMADCx software trigger configuration register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0														SCHSEL	
r														w	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SCHSEL															
w															

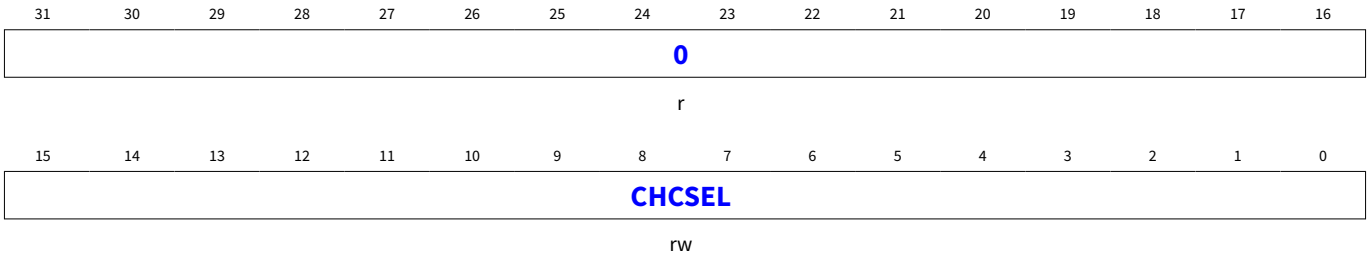
Field	Bits	Type	Description
SCHSEL	17:0	w	**Software trigger channel select** Each bit of this bitfield corresponds to the respective channel. 0_B No action 1_B Generate a trigger event by software
0	31:18	r	**Reserved** Read as 0; should be written with 0

45.8.44 TMADCx connection matrix configuration register

This register configures the connection between the channel and the SAR core for conversion.

TMADCx_CMATCFG (x=0-6) Offset address: $20010_H+x*800_H$

TMADCx connection matrix configuration register Kernel Reset value: $0000\ 0000_H$

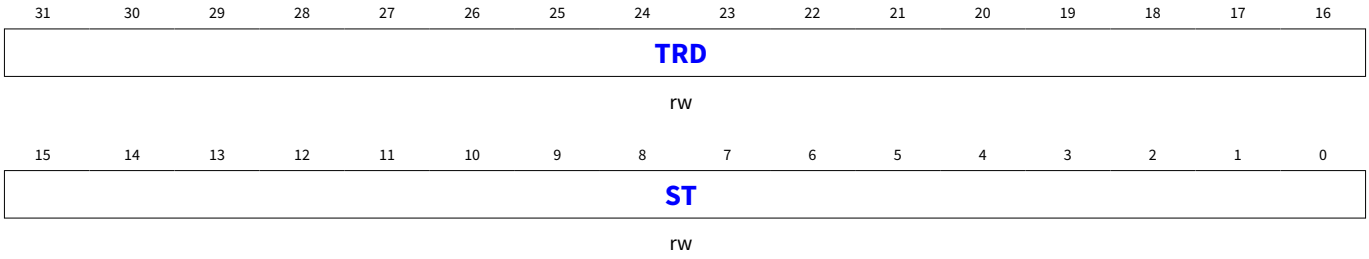


Field	Bits	Type	Description
CHCSEL	15:0	rw	**Channel core select** Each bit in this bitfield corresponds to the respective channel. 0_B Channel is connected to SAR Core-0 1_B Channel is connected to SAR Core-1
0	31:16	r	**Reserved** Read as 0; should be written with 0

45.8.45 TMADCx sampling time control register, channel y

TMADCx_CHy_STC (x=0-6;y=0-15) Offset address: $20018_H+x*800_H+y*12$

TMADCx sampling time control register, channel y Kernel Reset value: $0000\ 0003_H$

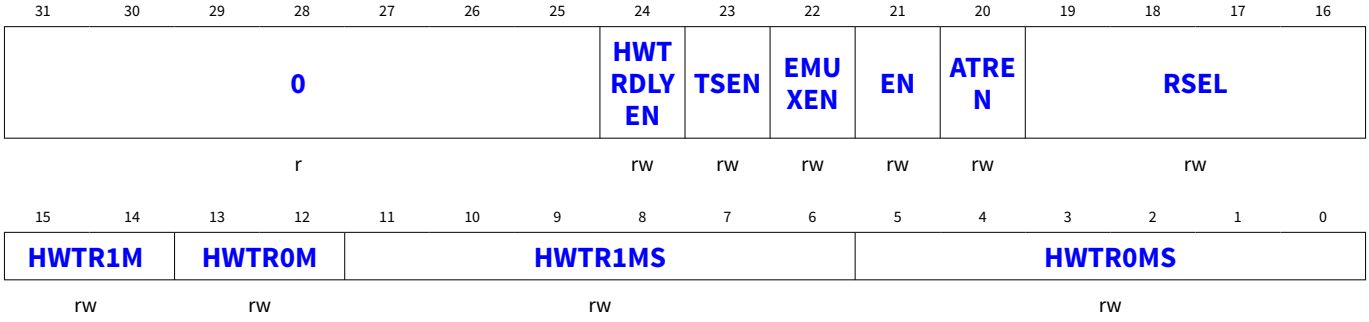


Field	Bits	Type	Description
ST	15:0	rw	**Sampling time** This bitfield represents the sampling time value for a channel. The sampling time counter runs with a clock of $f_{ADC}/2$. 0_H 12.5 ns (Reserved) 1_H 25 ns (Reserved) 2_H 37.5 ns (Reserved) 3_H 50 ns ... $FFFF_H$ 819.2 µs
TRD	31:16	rw	**Trigger delay** This bitfield represents the trigger delay value for the hardware triggers. The trigger delay counter runs with a clock of $f_{ADC}/2$

45.8.46 TMADCx configuration register, channel y

TMADCx_CHy_CFG (x=0-6;y=0-15) Offset address: $2001C_H+x*800_H+y*12$

TMADCx configuration register, channel y Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
HWTR0MS	5:0	rw	**Hardware trigger 0 multiplexer select** Refer "Hardware triggers to ADC" about the trigger source definition. 00_H Trigger number 0 selected 01_H Trigger number 1 selected 02_H Trigger number 2 selected … $3F_H$ Trigger number 63 selected
HWTR1MS	11:6	rw	**Hardware trigger 1 multiplexer select** Refer "Hardware triggers to ADC" about the trigger source definition. 00_H Trigger number 0 selected 01_H Trigger number 1 selected 02_H Trigger number 2 selected … $3F_H$ Trigger number 63 selected
HWTR0M	13:12	rw	**Hardware trigger 0 mode** 00_B No trigger edge selected (HWTR0 is disabled) 01_B Rising trigger edge selected 10_B Falling trigger edge selected 11_B Both trigger edges selected
HWTR1M	15:14	rw	**Hardware trigger 1 mode** 00_B No trigger edge selected (HWTR1 is disabled) 01_B Rising trigger edge selected 10_B Falling trigger edge selected 11_B Both trigger edges selected

(table continues...)

(continued)

Field	Bits	Type	Description
RSEL	19:16	rw	**Result and timestamp register select** This bitfield selects the result register where the conversion result value for the channel shall be stored. 0_H Result register 0 selected 1_H Result register 1 selected 2_H Result register 2 selected … F_H Result register 15 selected
ATREN	20	rw	**Auto-trigger enable** This bit if set, puts the channel into an auto-trigger mode.
EN	21	rw	**Enable** This bit if set, enables the input channel connection. 0_B Channel connection disabled 1_B Channel connection enabled
EMUXEN	22	rw	**EMUX enable** 0_B Channel not connected to EMUX 1_B Channel connected to EMUX
TSEN	23	rw	**Time-stamp enable** 0_B Time-stamp disabled 1_B Time-stamp enabled
HWTRDLYEN	24	rw	**Trigger delay enable** 0_B Trigger delay counter disabled 1_B Trigger delay counter enabled
0	31:25	r	**Reserved** Read as 0; should be written with 0

45.8.47 TMADCx status register, channel y

TMADCx_CHy_STAT (x=0-6;y=0-15) Offset address: $20020_H+x*800_H+y*12$

TMADCx status register, channel y Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								SWTRE	HWTRE	TRENR					
r								rh	rh	rh					

Field	Bits	Type	Description
TRENR	5:0	rh	**Trigger error number** This bitfield indicates the hardware trigger number which is ignored when a hardware trigger error is detected by bit HWTRE. 00_H Trigger number 0 ignored 01_H Trigger number 1 ignored 02_H Trigger number 2 ignored … $3F_H$ Trigger number 63 ignored
HWTRE	6	rh	**Hardware trigger error** This bit indicates if a hardware trigger is ignored. 0_B No error 1_B Hardware trigger is ignored
SWTRE	7	rh	**Software trigger error** This bit indicates if a software trigger is ignored. 0_B No error 1_B Software trigger is ignored
0	31:8	r	**Reserved** Read as 0; should be written with 0

45.8.48 TMADCx configuration register, monitor channel y

TMADCx_MCHy_CFG (x=0-6;y=0-1) Offset address: $200D8_H+x*800_H+y*4$

TMADCx configuration register, monitor channel y Kernel Reset value: $004F\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ST															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												EN	ATREN	SEL	
r												rw	rw	rw	

Field	Bits	Type	Description
SEL	1:0	rw	**Monitor multiplexer input selection** See chapter (On-chip signal supervision) for signal selection settings. Configuration setting 4-7 are reserved for use.
ATREN	2	rw	**Auto-trigger enable** This bit if set, puts the channel into an auto-trigger mode.

(table continues...)

(continued)

Field	Bits	Type	Description
EN	3	rw	**Enable** This bit if set, enables the monitor channel input connection. 0_B Disabled 1_B Enabled
ST	31:16	rw	**Monitor channel sampling time** This bitfield represents the sampling time value for the monitor channel. The monitor channel sampling time counter runs with a clock of $f_{ADC}/2$. 0_H 12.5 ns (Reserved) 1_H 25 ns (Reserved) 2_H 37.5 ns (Reserved) $4F_H$ 1 µs ... $FFFF_H$ 819.2 µs
0	15:4	r	**Reserved** Read as 0; should be written with 0

45.8.49 TMADCx boundary select register n

The boundary register defines the boundary values used for the limit checking of the conversion results.

TMADCx_BNDn_SEL (n=0-1;x=0-6) Offset address: $200E0_H+x*800_H+n*12$

TMADCx boundary select register n Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0				UB											
r				rw											

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				LB											
r				rw											

Field	Bits	Type	Description
LB	11:0	rw	**Lower boundary value for limit checking** This is the lower boundary threshold value.
UB	27:16	rw	**Upper boundary value for limit checking** This is the upper boundary threshold value.
0	15:12, 31:28	r	**Reserved** Read as 0; should be written with 0

45.8.50 TMADCx boundary status regsiter n

TMADCx_BNDn_STAT (n=0-1;x=0-6) Offset address: $200E4_H+x*800_H+n*12$

TMADCx boundary status regsiter n Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0				CUB											
r				rh											

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				CLB											
r				rh											

Field	Bits	Type	Description
CLB	11:0	rh	**Current lower boundary value for limit checking** This is the current lower boundary threshold value.
CUB	27:16	rh	**Current upper boundary value for limit checking** This is the current upper boundary threshold value.
0	15:12, 31:28	r	**Reserved** Read as 0; should be written with 0

45.8.51 TMADCx boundary trigger configuration register, boundary register n

TMADCx_BNDn_TRGCFG (n=0-1;x=0-6) Offset address: $200E8_H+x*800_H+n*12$

TMADCx boundary trigger configuration register, boundary register n Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TRD															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								TRMS		TRS					
r								rw		rw					

Field	Bits	Type	Description
TRS	5:0	rw	**Trigger source select for Boundary value update in shadow update mode** Refer "Hardware triggers to ADC" about the trigger source definition. 00_H Trigger number 0 selected 01_H Trigger number 1 selected 02_H Trigger number 2 selected … $3F_H$ Trigger number 63 selected
TRMS	7:6	rw	**Trigger mode select** 00_B No trigger edge selected (HWTR0 is disabled) 01_B Rising trigger edge selected 10_B Falling trigger edge selected 11_B Both trigger edges selected
TRD	31:16	rw	**Trigger delay** This bitfield represents the delay value for the boundary value update triggers. The trigger delay counter runs with a clock of fADC/2
0	15:8	r	**Reserved** Read as 0; should be written with 0

45.8.52 TMADCx boundary flag configuration register n

TMADCx_BFLn_CFG (n=0-1;x=0-6) Offset address: $200F8_H+x*800_H+n*8$

TMADCx boundary flag configuration register n Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0											BFM	RESEL			
r											rw	rw			

Field	Bits	Type	Description
RESEL	3:0	rw	**Result register select** 0_H: BFLn is generated based on conversion result in TMADCx result register 0 1_H: BFLn is generated based on conversion result in TMADCx result register 1 .. .. .. F_H: BFLn is generated based on conversion result in TMADCx result register 15

(table continues...)

(continued)

Field	Bits	Type	Description
BFM	4	rw	**Boundary flag mode** 0_B Disable boundary flag 1_B Enable boundary flag
0	31:5	r	**Reserved** Read as 0; should be written with 0

45.8.53 TMADCx boundary flag status register n

TMADCx_BFLn_STAT (n=0-1;x=0-6) Offset address: $200FC_H+x*800_H+n*8$

TMADCx boundary flag status register n Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															BFL
r															rh

Field	Bits	Type	Description
BFL	0	rh	**Boundary flag value** 0_B Boundary flag indicates logic low output 1_B Boundary flag indicates logic high output
0	31:1	r	**Reserved** Read as 0; should be written with 0

45.8.54 TMADCx configuration register, result register u

TMADCx_RSLTu_CFG (u=0-15;x=0-6) Offset address: $2010C_H+x*800_H+u*8$

TMADCx configuration register, result register u Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0										HYSTM	WFR	BNDSELR	BNDMODE	BNDSEL	
r										rw	rw	rw	rw	rw	

Field	Bits	Type	Description
BNDSEL	1:0	rw	**Boundary select** 00_B Boundary check disabled 01_B Conversion result compared to upper boundary 10_B Conversion result compared to lower boundary 11_B Conversion result compared to both upper and lower boundaries
BNDMODE	2	rw	**Boundary mode** 0_B Conversion result within boundary Boundary event will be generated if result is within boundary, otherwise no event. Boundary flag will be set to 1 if result is within boundary, otherwise clear boundary flag. 1_B Converison result outside boundary Boundary event will be generated if result is outside configured boundary, otherwise no event. Boundary flag will be set to 1 if result is outside boundary, otherwise clear boundary flag.
BNDSELR	3	rw	**Boundary register select** This bit selects the boudary select register from which the thresholds values must be used for comparisons with the conversion result data. Note: If boundary check for a result register is enabled, then a boundary event is generated only when the boundary check is met for that register. The result will however still be stored in the result register irrespective of the boundary check. 0_B Boundary register 0 selected 1_B Boundary register 1 selected
WFR	4	rw	**Wait for read** This bit if set, masks all the trigger signals to a channel until the result data has been read from the corresponding result register. 0_B Wait-for-read disabled 1_B Wait-for-read enabled
HYSTM	5	rw	**Hysteresis mode** This mode must be used only with configuration TMx_RESCFGu.BNDSEL=11, if this bit is set, the specified boundaries form a hysteresis band for the channel result. The hysteresis behavior is captured by the configured boundary flag output. 0_B Disabled 1_B Enabled
0	31:6	r	**Reserved** Read as 0; should be written with 0

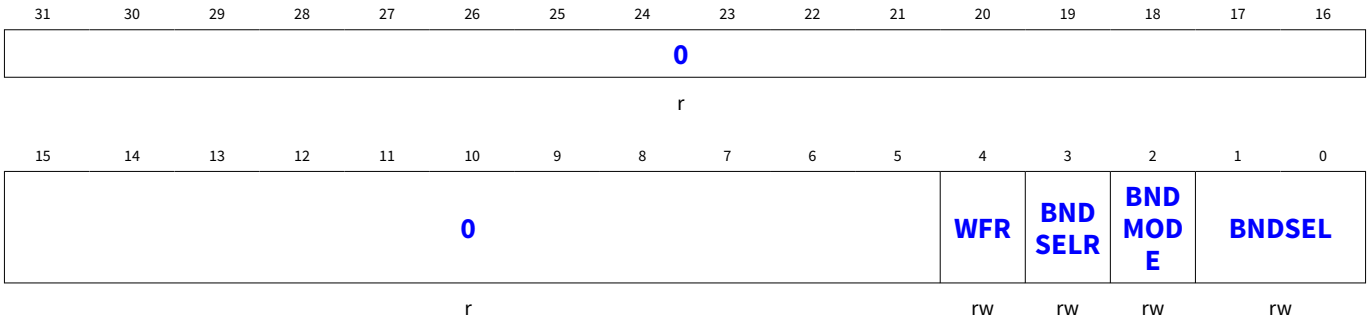
45.8.55 TMADCx configuration register, monitor channel u result register

TMADCx_MRSLTu_CFG (u=0-1;x=0-6)

Offset address: $20188_H+x*800_H+u*4$

TMADCx configuration register, monitor channel u result register

Kernel Reset value: $0000\ 0000_H$

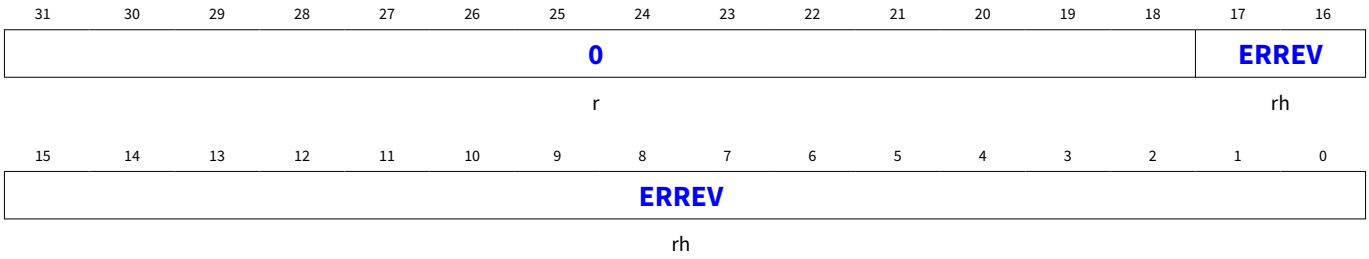


Field	Bits	Type	Description
BNDSEL	1:0	rw	**Boundary select** 00_B Boundary check disabled 01_B Conversion result compared to upper boundary 10_B Conversion result compared to lower boundary 11_B Conversion result compared to both upper and lower boundaries
BNDMODE	2	rw	**Boundary mode** 0_B Conversion result within boundary Boundary event will be generated if result within boundary, otherwise no event.Boundary flag will be set to 1 if result within boundary, otherwise clear boundary flag. 1_B Converison result outside boundary Boundary event will be generated if result outside configured boundary, otherwise no event.Boundary flag will be set to 1 if result outside boundary, otherwise clear boundary flag.
BNDSELR	3	rw	**Boundary register select** This bit selects the boudary select register from which the thresholds values must be used for comparisons with the conversion result data. Note: If boundary check for a result register is enabled, then a boundary event is generated only when the boundary check is met for that register. The result will however still be stored in the result register irrespective of the boundary check. 0_B Boundary register 0 selected 1_B Boundary register 1 selected
WFR	4	rw	**Wait for read** This bit if set, masks all the trigger signals to a channel until the result data has been read from the corresponding result register. 0_B Wait-for-read disabled 1_B Wait-for-read enabled
0	31:5	r	**Reserved** Read as 0; should be written with 0

45.8.56 TMADCx error event flag register

TMADCx_ERRF (x=0-6) Offset address: $20190_H+x*800_H$

TMADCx error event flag register Kernel Reset value: $0000\ 0000_H$

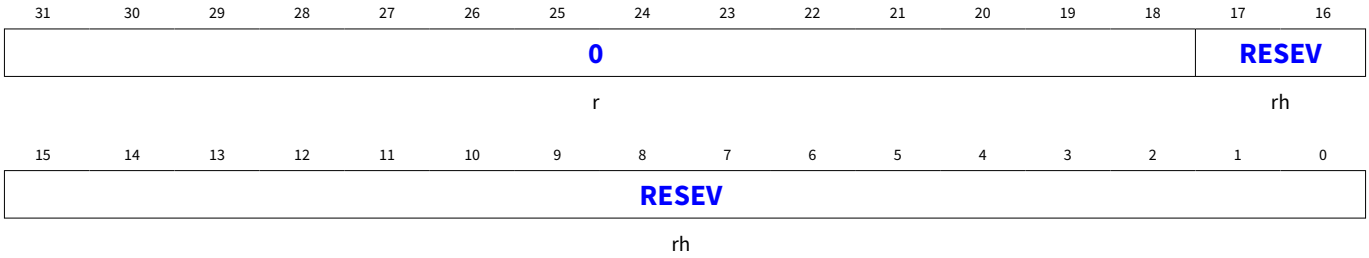


Field	Bits	Type	Description
ERREV	17:0	rh	**Error event** Each bit in this bitfield corresponds to an error event on the respective channel. 0_B No error event on the channel 1_B Error event occured on the channel
0	31:18	r	**Reserved** Read as 0; should be written with 0

45.8.57 TMADCx result event flag register

TMADCx_RESF (x=0-6) Offset address: $20194_H+x*800_H$

TMADCx result event flag register Kernel Reset value: $0000\ 0000_H$

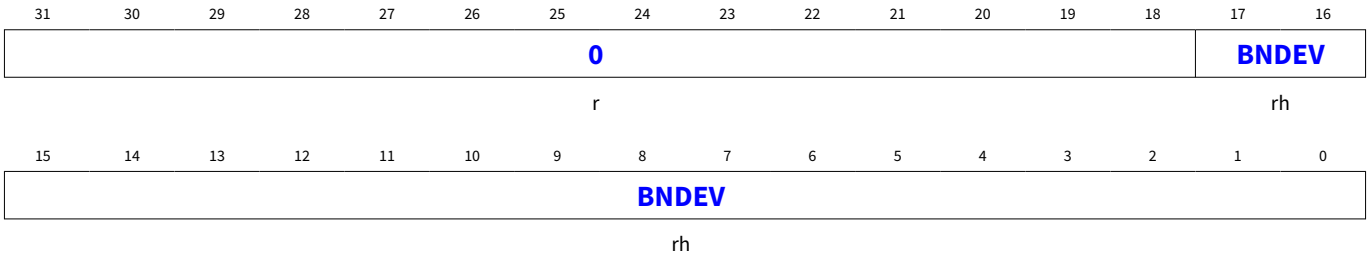


Field	Bits	Type	Description
RESEV	17:0	rh	**Result event** Each bit in this bitfield corresponds to a result event for the result register. 0_B No result event 1_B New data available in the result register
0	31:18	r	**Reserved** Read as 0; should be written with 0

45.8.58 TMADCx boundary event flag register

TMADCx_BNDF (x=0-6) Offset address: $20198_H+x*800_H$

TMADCx boundary event flag register Kernel Reset value: $0000\ 0000_H$

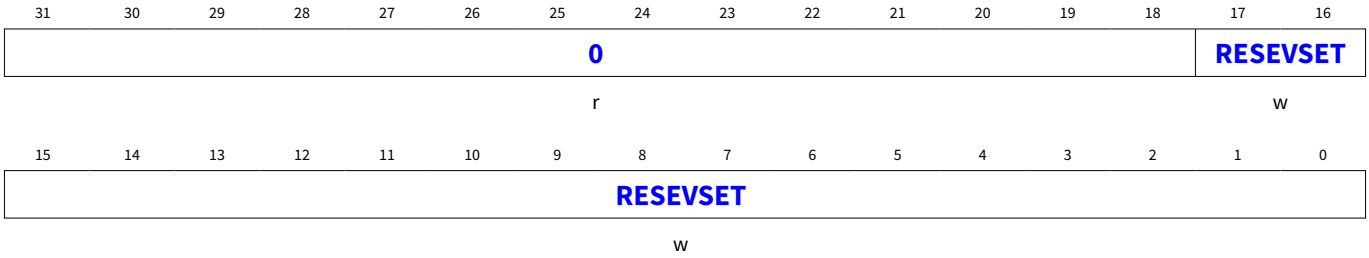


Field	Bits	Type	Description
BNDEV	17:0	rh	**Boundary event** Each bit in this bitfield corresponds to a boundary event for the respective result register. 0_B Boundary check failure 1_B Boundary check successful
0	31:18	r	**Reserved** Read as 0; should be written with 0

45.8.59 TMADCx result event flag set register

TMADCx_RESFSET (x=0-6) Offset address: $201A0_H+x*800_H$

TMADCx result event flag set register Kernel Reset value: $0000\ 0000_H$

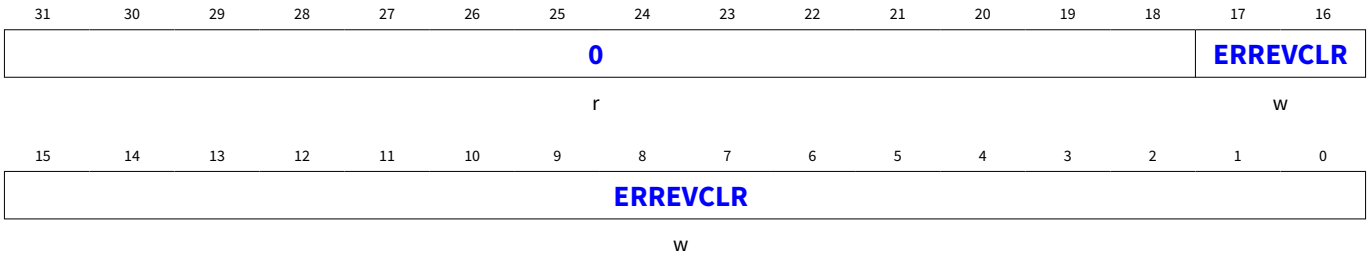


Field	Bits	Type	Description
RESEVSET	17:0	w	**Result event set** By writing to the respective result event flag set bit, the corresonding event flag in the RESF register is set. 0_B No action 1_B Set the corresponding RESEVx flag in RESF register
0	31:18	r	**Reserved** Read as 0; should be written with 0

45.8.60 TMADCx error event flag clear register

TMADCx_ERRFCLR (x=0-6) Offset address: $201A4_H+x*800_H$

TMADCx error event flag clear register Kernel Reset value: $0000\ 0000_H$

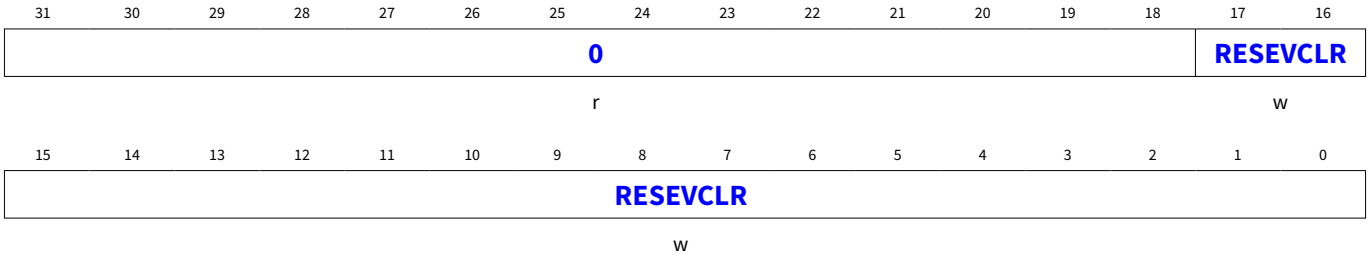


Field	Bits	Type	Description
ERREVCLR	17:0	w	**Error event clear** Each bit in this bitfield, if set, clears the error event and the error status bits of the respective channel. 0_B No action 1_B Clear the corresponding error event flag in TMx_ERRF
0	31:18	r	**Reserved** Read as 0; should be written with 0

45.8.61 TMADCx result event flag clear register

TMADCx_RESFCLR (x=0-6) Offset address: $201A8_H+x*800_H$

TMADCx result event flag clear register Kernel Reset value: $0000\ 0000_H$

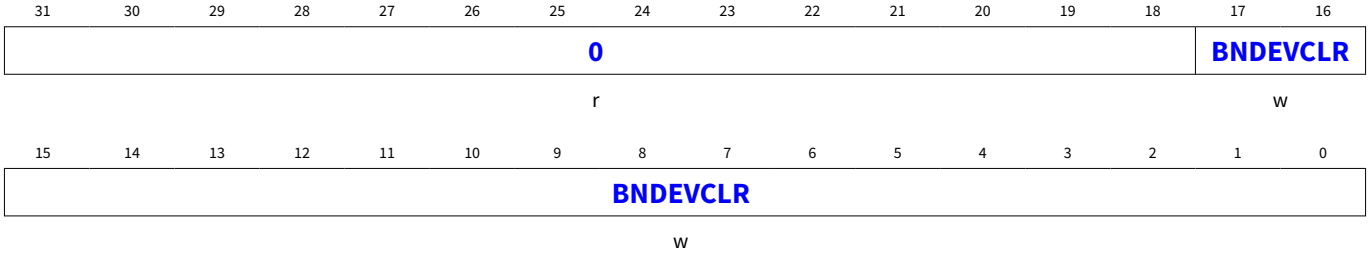


Field	Bits	Type	Description
RESEVCLR	17:0	w	**Result event clear** Each bit in this bitfield, if set, clears the result event for the respective result register. 0_B No action 1_B Clear the corresponding RESEVx flag in the RESF register
0	31:18	r	**Reserved** Read as 0; should be written with 0

45.8.62 TMADCx boundary event flag clear register

TMADCx_BNDFCLR (x=0-6) Offset address: $201AC_H+x*800_H$

TMADCx boundary event flag clear register Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
BNDEVCLR	17:0	w	**Boundary event clear** Each bit in this bitfield, if set, clears the boundary event for the respective result register. 0_B No action 1_B Clear the corresponding boundary event flag in TMx_BNDF
0	31:18	r	**Reserved** Read as 0; should be written with 0

45.8.63 TMADCx service request configuration register i

TMADCx_SRi_CFG (i=0-6;x=0-6) Offset address: $201B0_H+x*800_H+i*4$

TMADCx service request configuration register i Kernel Reset value: $0000\ 0000_H$

Field	Bits	Type	Description
EVEN	17:0	rw	**Event enable** Each bit in this bitfield, if set enables the event on the channel or result register depending on the value of the bit EVSEL for service request generation. If EVSEL=01_B,each bit here corresponds to the respective channel. If EVSEL=10_B or EVSEL=11_B, then each bit in the bitfield corresponds to the result register. 0_B Event disabled 1_B Event enabled
EVSEL	19:18	rw	**Event selection** This bit defines the selection of the events set in the bitfield EVEN. 00_B Disable service request 01_B Error event 10_B Result event 11_B Boundary event

(table continues...)

(continued)

Field	Bits	Type	Description
EVOP	21:20	rw	**Event operation** This bit if set, defines that condition of generation for service request. 00_B OR logic operation without wait-for-clear This setting corresponds to the generation of service request on the occurence of each event configured in the bitfield EVEN. 01_B OR logic operation with wait-for-clear A service request is generated only on the first available event and on the first available event after the corresponding flag register bits for the event(s) specified in EVEN bitfield are cleared by SW. 10_B AND logic operation This setting corresponds to the generation of service request only on the occurence of all events configured in the bitfield EVEN. 11_B Reserved for future use (treated same as AND logical operation if configured)
0	31:22	r	**Reserved** Read as 0; should be written with 0

45.8.64 TMADCx monitor channel v result register

TMADCx_MRESv (v=0-1;x=0-6) Offset address: $201CC_H+x*800_H+v*4$

TMADCx monitor channel v result register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	DR	0													
r	rh	r													

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESULT															
rh															

Field	Bits	Type	Description
RESULT	15:0	rh	**Separated register to store monitor chnnel results** Monitor channel result
DR	30	rh	**Diagnostic result** This bit indicates if the conversion result is a diagnostics (CSD) data or not. 0_B Not a diagnostic result 1_B Diagnostic result
0	29:16, 31	r	**Reserved** Read as 0; should be written with 0

45.8.65 TMADCx module status register

TMADCx_MODSTAT (x=0-6) Offset address: $201D4_H+x*800_H$

TMADCx module status register Kernel Reset value: $0000\ 0001_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0										PONPH		CALPH		RUN	DIS
r										rh		rh		rh	rh

Field	Bits	Type	Description
DIS	0	rh	**Disable** 0_B Module enabled 1_B Module disabled
RUN	1	rh	**Run** 0_B Indicates module in configuration state 1_B Indicates module in run state
CALPH	3:2	rh	**Calibration phase Indication** This bit indicates the status of the calibration phase. 00_B Not calibrated or calibration not started 01_B Calibration phase currently active 10_B Calibration successfully completed 11_B Reserved
PONPH	5:4	rh	**Power-on phase Indication** This bit indicates the status of the power on phase. 00_B Module is not powered on 01_B Module power-on sequence is curently active 10_B Module is powered-on 11_B Reserved
0	31:6	r	**Reserved** Read as 0; should be written with 0

45.8.66 TMADCx broken wire detection control register

TMADCx_BWDCFG (x=0-6) Offset address: $201D8_H+x*800_H$

TMADCx broken wire detection control register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ENCH15		ENCH14		ENCH13		ENCH12		ENCH11		ENCH10		ENCH9		ENCH8	
rw		rw		rw		rw		rw		rw		rw		rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ENCH7		ENCH6		ENCH5		ENCH4		ENCH3		ENCH2		ENCH1		ENCH0	
rw		rw		rw		rw		rw		rw		rw		rw	

Field	Bits	Type	Description
ENCHy (y=0-15)	2*y+1:2*y	rw	**Enable BWD channel y** 00_B No action 01_B Pull-down current source activated 10_B Pull-up current source activated 11_B No action

45.8.67 TMADCx converter self-diagnostics register core-n

TMADCx_CSDn (n=0-1;x=0-6) Offset address: $201DC_H+x*800_H+n*4$

TMADCx converter self-diagnostics register core-n Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EN	SEL		CHSEL				
r								rw	rw		rw				

Field	Bits	Type	Description
CHSEL	4:0	rw	**CSD channel select (valid only for channels connected to Core-n)** This bitfield selects the channel number connected on which CSD is implemented when enabled.
SEL	6:5	rw	**Select** This bitfield selects the signal to be converted for the selected channel in CSD. 00_B (1972/4096)*VAREF 01_B (220/4096)*VAREF 10_B (1972/4096)*VDDM 11_B (220/4096)*VDDM
EN	7	rw	**Enable** This bitfield if set enables the CSD for the selected channel.
0	31:8	r	**Reserved** Read as 0; should be written with 0

45.8.68 TMADCx time-stamp register v

TMADCx_AW0_TSv (v=0-15;x=0-6) Offset address: $20200_H+x*800_H+v*4$

TMADCx time-stamp register v Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EMUXCH				0											
rh				r											

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TIMESTAMP															
rh															

Field	Bits	Type	Description
TIMESTAMP	15:0	rh	**The most recent captured timestamp value**
EMUXCH	31:28	rh	**EMUX channel number** This bit indicates the external multiplexer channel number of the converted channel result.
0	27:16	r	**Reserved** Read as 0; should be written with 0

Table 2401 Mirrors of TMADCx_AW0_TSv (v=0-15;x=0-6)

Short name	Long name	Offset address
TMADCx_AW1_TSv (v=0-15;x=0-6)	TMADCx time-stamp register v is an alias with matching index variables.	$20280_H+x*800_H+v*8$
TMADCx_AW2_TSv (v=0;x=0-6)	TMADCx time-stamp register v is an alias with matching index variables.	$20300_H+x*800_H$
TMADCx_AW2_TSv (v=7;x=0-6)	TMADCx time-stamp register v is an alias with matching index variables.	$20340_H+x*800_H$
TMADCx_AW2_TSv (v=14;x=0-6)	TMADCx time-stamp register v is an alias with matching index variables.	$20360_H+x*800_H$

45.8.69 TMADCx result register v

TMADCx_AW0_RESv (v=0-15;x=0-6) Offset address: $20240_H+x*800_H+v*4$

TMADCx result register v Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TRE	DR	0			SWTR	TRNM						CHNR			
rh	rh	r			rh	rh						rh			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESULT															
rh															

Field	Bits	Type	Description
RESULT	15:0	rh	**Conversion result for the configured channel**
CHNR	19:16	rh	**TMADC channel number** This bit indicates the channel number of the converted channel result.
TRNM	25:20	rh	**Trigger number** This bit indicates the trigger number of the hardware trigger source which initiated the channel conversion.
SWTR	26	rh	**Software trigger status** 0_B Channel conversion not initiated by software trigger 1_B Channel conversion initiated by software trigger
DR	30	rh	**Diagnostic result** This bit indicates if the conversion result is a diagnostics (BWD or CSD) data or not. 0_B Not a diagnostic result 1_B Diagnostic result
TRE	31	rh	**Trigger edge** This bit indicates the information on the edge of the hardware trigger that initiated the channel conversion 0_B Trigger falling edge 1_B Trigger rising edge
0	29:27	r	**Reserved** Read as 0; should be written with 0

Table 2402 Mirrors of TMADCx_AW0_RESv (v=0-15;x=0-6)

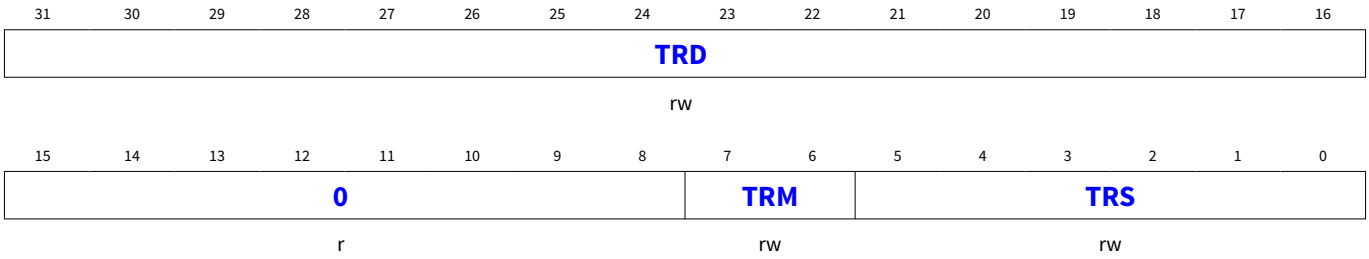
Short name	Long name	Offset address
TMADCx_AW1_RESv (v=0-15;x=0-6)	TMADCx result register v is an alias with matching index variables.	20284_H+x*800_H+v*8
TMADCx_AW2_RESv (v=0-6;x=0-6)	TMADCx result register v is an alias with matching index variables.	20304_H+x*800_H+v*4
TMADCx_AW2_RESv (v=7-13;x=0-6)	TMADCx result register v is an alias with matching index variables.	20344_H+x*800_H+(v-7)*4
TMADCx_AW2_RESv (v=14-15;x=0-6)	TMADCx result register v is an alias with matching index variables.	20364_H+x*800_H+(v-14)*4
TMADCx_AW3_RESv (v=0-4;x=0-6)	TMADCx result register v is an alias with matching index variables.	$2036C_H$+x*800_H+v*4

45.8.70 FCCx threshold trigger configuration register

Refer "Hardware triggers to ADC" about the trigger source definition

FCCx_THTRCFG (x=0-9) Offset address: 40000_H+x*400_H

FCCx threshold trigger configuration register Kernel Reset value: 0000 0000_H

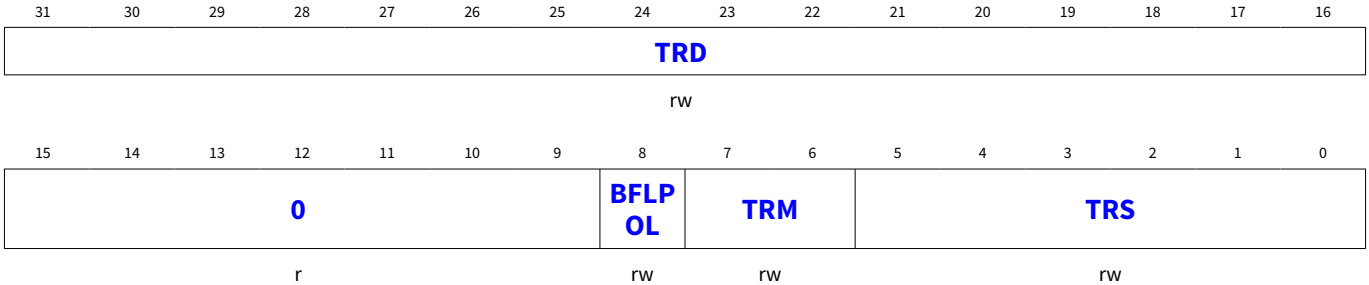


Field	Bits	Type	Description
TRS	5:0	rw	**Threshold trigger select** 00_H Trigger number 0 selected 01_H Trigger number 1 selected 02_H Trigger number 2 selected … $3F_H$ Trigger number 63 selected
TRM	7:6	rw	**Threshold trigger mode** 00_B Trigger is disabled 01_B Rising trigger edge selected 10_B Falling trigger edge selected 11_B Both trigger edges selected
TRD	31:16	rw	**Threshold trigger delay** This bitfield represents the trigger delay value for the threshold triggers. The trigger delay counter operates with a clock of 80 MHz.
0	15:8	r	**Reserved** Read as 0; should be written with 0

45.8.71 FCCx gate trigger configuration register

FCCx_GATRCFG (x=0-9) Offset address: $40004_H+x*400_H$

FCCx gate trigger configuration register Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
TRS	5:0	rw	**Trigger source select for gating boundary flag signal** Refer "Hardware triggers to ADC" about the trigger source definition. 00_H Trigger number 0 selected 01_H Trigger number 1 selected 02_H Trigger number 2 selected … $3F_H$ Trigger number 63 selected
TRM	7:6	rw	**Gate trigger mode** 00_B Trigger is disabled 01_B Rising trigger edge selected 10_B Falling trigger edge selected 11_B Both trigger edges selected
BFLPOL	8	rw	**Gated boundary flag polarity** 0_B Gated boundary flag value is 0 1_B Gated boundary flag value is 1
TRD	31:16	rw	**Gate trigger delay** This bitfield represents the trigger delay value for the threshold triggers. The trigger delay counter operates with a clock of 80 MHz.
0	15:9	r	**Reserved** Read as 0; should be written with 0

45.8.72 FCCx ramp trigger configuration register

FCCx_RMPTRCFG (x=0-9) Offset address: $40008_H+x*400_H$

FCCx ramp trigger configuration register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TRD															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								TRM		TRS					
r								rw		rw					

Field	Bits	Type	Description
TRS	5:0	rw	**Ramp trigger select** Refer "Hardware triggers to ADC" about the trigger source definition. 00_H Trigger number 0 selected 01_H Trigger number 1 selected 02_H Trigger number 2 selected … $3F_H$ Trigger number 63 selected

(table continues...)

(continued)

Field	Bits	Type	Description
TRM	7:6	rw	**Ramp trigger mode** 00_B Trigger is disabled 01_B Rising trigger edge selected 10_B Falling trigger edge selected 11_B Both trigger edges selected
TRD	31:16	rw	**Ramp trigger delay** This bitfield represents the trigger delay value for the threshold triggers. The trigger delay counter operates with a clock of 80 MHz.
0	15:8	r	**Reserved** Read as 0; should be written with 0

45.8.73 FCCx ramp configuration register

FCCx_RMPCFG (x=0-9) Offset address: $4000C_H+x*400_H$

FCCx ramp configuration register Kernel Reset value: $0000\ 000F_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0			DIR	RSWTR	0	STEPSZ									
r			rw	w	r	rw									

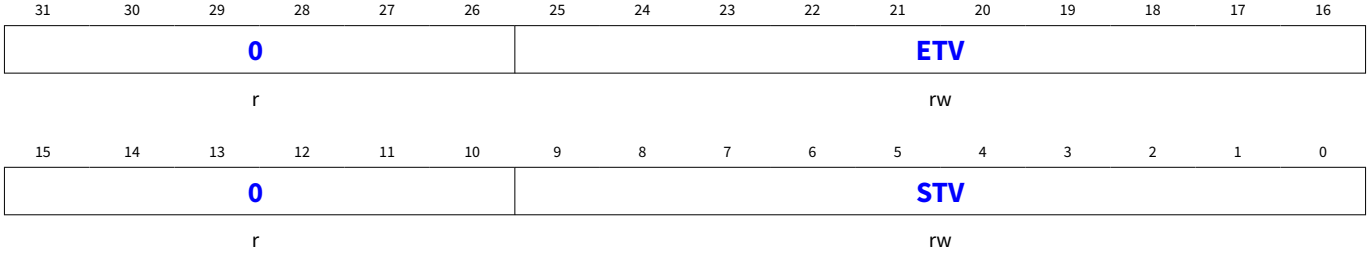
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STEPWD															
rw															

Field	Bits	Type	Description
STEPWD	15:0	rw	**Step width** This bitfield defines the step width for each reference value when the ramp mode is enabled.
STEPSZ	25:16	rw	**Step size** This bitfield defines the step size increment or decrement from each reference value when the ramp mode is enabled.
RSWTR	27	w	**Ramp start software trigger** 0_B No action 1_B Writing 1 to this bit starts the ramp (valid only for ramp modes with start by software)
DIR	28	rw	**Direction** This bit defines the ramp direction when the ramp mode is enabled. 0_B Ramp increment 1_B Ramp decrement
0	26, 31:29	r	**Reserved** Read as 0; should be written with 0

45.8.74 FCCx ramp threhsold register

FCCx_RMPT (x=0-9) Offset address: $40010_H+x*400_H$

FCCx ramp threhsold register Kernel Reset value: $0000\ 0000_H$

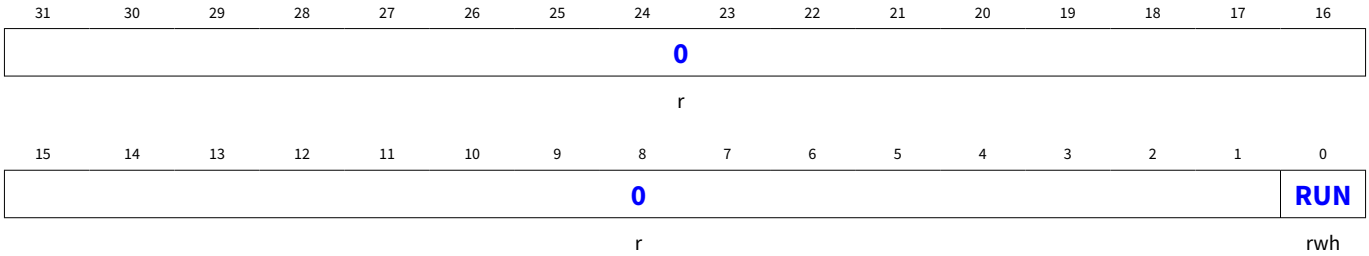


Field	Bits	Type	Description
STV	9:0	rw	**Start threshold value**
ETV	25:16	rw	**End threshold value**
0	15:10, 31:26	r	**Reserved** Read as 0; should be written with 0

45.8.75 FCCx module enable register

FCCx_MODEN (x=0-9) Offset address: $40014_H+x*400_H$

FCCx module enable register Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
RUN	0	rwh	**Run** 0_B Set module in configuration state 1_B Set module in run state
0	31:1	r	**Reserved** Read as 0; should be written with 0

45.8.76 FCCx module configuration register

FCCx_MODCFG (x=0-9) Offset address: $40018_H+x*400_H$

FCCx module configuration register Kernel Reset value: $0000\ 0000_H$

Field	Bits	Type	Description
TV	9:0	rw	**Threshold value**
SWTR	18	w	**Software trigger** 0_B No action 1_B Writing 1 to this bit sets the new threshold value for the DAC
GFD	27:20	rw	**Glitch filter depth** 00_H Glitch filter disabled 01_H Depth 1 02_H Depth 2 … FF_H Depth 255
MODE	30:28	rw	**Mode** 000_B Normal operation mode 001_B Reserved for future use (treated as normal operation if configured) 010_B Hysteresis mode 011_B Start ramp on ramp start software trigger (RSWTR=1) and end ramp when register bit field RMPT.ETV value is reached 100_B Start ramp on ramp start software trigger (RSWTR=1) and end ramp on hardware trigger or on ETV 101_B Start ramp on ramp start software trigger (RSWTR=1) and end ramp on the first transition of the boundary flag or ETV 110_B Start ramp on hardware ramp trigger and end ramp when register bit field RMPT.ETV value is reached 111_B Start/Restart ramp on hardware trigger and end on reaching the ETV value
DDCRT	31	rw	**Disable DAC charge refresh timer** Setting this bit to 0 enables the DAC charge refresh timer after module power-up 0_B Timer is enabled 1_B Timer is disabled
0	15:10, 17:16, 19	r	**Reserved** Read as 0; should be written with 0

45.8.77 FCCx hyteresis configuration register

FCCx_HYSTCFG (x=0-9) Offset address: $4001C_H+x*400_H$

FCCx hyteresis configuration register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0						DELP									
r						rw									

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0						DELM									
r						rw									

Field	Bits	Type	Description
DELM	9:0	rw	**Delta minus**
DELP	25:16	rw	**Delta plus**
0	15:10, 31:26	r	**Reserved** Read as 0; should be written with 0

45.8.78 FCCx boundary flag configuration register

FCCx_BFLCFG (x=0-9) Offset address: $40020_H+x*400_H$

FCCx boundary flag configuration register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												SBFV	BFS	BFM	BFA
r												rw	rw	rw	rw

Field	Bits	Type	Description
BFA	0	rw	**Boundary flag activation** 0_B Normal mode: Set BFL when the channel input voltage is higher than threshold, otherwise clear BFL.Hysteresis mode: Set BFL when the channel input voltage is higher than upper threshold, clear BFL when result is lower than lower threshold. 1_B Normal mode: Set BFL when the channel input voltage is lower than threshold, otherwise clear BFL. Hysteresis mode: Set BFL when the channel input voltage is lower than lower threshold, clear BFL when result is higher than upper threshold.
BFM	1	rw	**Boundar flag mode control** 0_B Disable boundary flag 1_B Enable boundary flag

(table continues...)

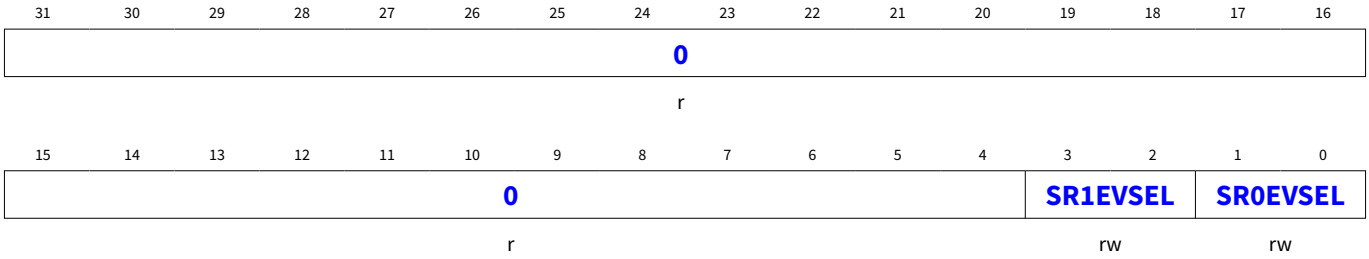
(continued)

Field	Bits	Type	Description
BFS	2	rw	**Boundary flag select** 0_B Select hardware BFL 1_B Select software BFL
SBFV	3	rw	**Software boundary flag value** 0_B Clear BFL 1_B Set BFL
0	31:4	r	**Reserved** Read as 0; should be written with 0

45.8.79 FCCx service request configuration register

FCCx_SRCFG (x=0-9) Offset address: $40024_H+x*400_H$

FCCx service request configuration register Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
SR0EVSEL	1:0	rw	**Service request 0 event select** 00_B Service request line disabled 01_B Service request on the rising edge of BFL 10_B Service request on the falling edge of BFL 11_B Service request on both the rising and fallign edges of BFL
SR1EVSEL	3:2	rw	**Service request 1 event select** 00_B Service request line disabled 01_B Service request on new threshold update (when MODE<=010_B) or Service request on end of ramp (when MODE >010_B) 10_B Reserved 11_B Reserved
0	31:4	r	**Reserved** Read as 0; should be written with 0

45.8.80 FCCx module status register

FCCx_MODSTAT (x=0-9) Offset address: $40028_H+x*400_H$

FCCx module status register Kernel Reset value: $0010\ 0000_H$

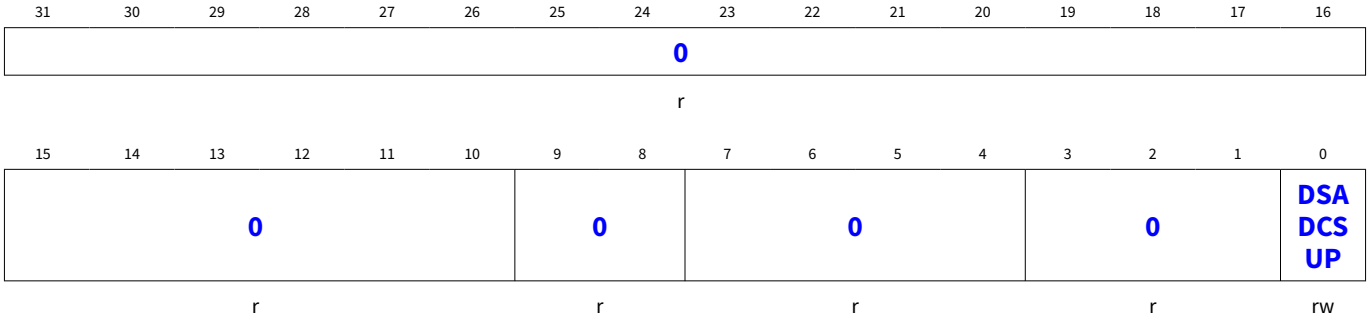
Field	Bits	Type	Description
CTV	9:0	rh	**Current threshold value**
BFL	16	rh	**Boundary flag value** 0_B Normal mode: Input voltage is less than threshold (BFLCFG.BFA=0), or higher than threshold (BFLCFG.BFA=1)Hysteresis mode: Input voltage is less than lower threshold (BFLCFG.BFA=0), or higher than upper threshold (BFLCFG.BFA=1) 1_B Channel comparison result passes boundary activation check Normal mode: Input voltage is higher than threshold (BFLCFG.BFA=0), or less than threshold (BFLCFG.BFA=1)Hysteresis mode: Input voltage is higher than upper threshold (BFLCFG.BFA=0), or less than lower threshold (BFLCFG.BFA=1)
DCR	17	rh	**Direct comparison result** 0_B Input voltage less than the threshold 1_B Input voltage higher than the threshold
VALTV	18	rh	**Valid threshold value** 0_B Current Threshold Value (CTV) is invalid 1_B Current Threshold Value (CTV) is valid
RUN	19	rh	**Run** 0_B Indicates module in configuration state 1_B Indicates module in run state
DIS	20	rh	**Disable** 0_B Module enabled 1_B Module disabled
RC	21	rh	**Ramp counter** 0_B Ramp counter is not active 1_B Ramp counter is running
0	15:10, 31:22	r	**Reserved** Read as 0; should be written with 0

45.8.81 DSADC global configuration register

Global configuration register control the selection of the supervision voltage to be converted in TMADC. User shall make sure that selection of the signal is done prior to triggering the conversion (sampling time shall be taken into consideration).

DSADC_GLOBCFG Offset address: $5F800_H$

DSADC global configuration register Kernel Reset value: $0000\ 0000_H$



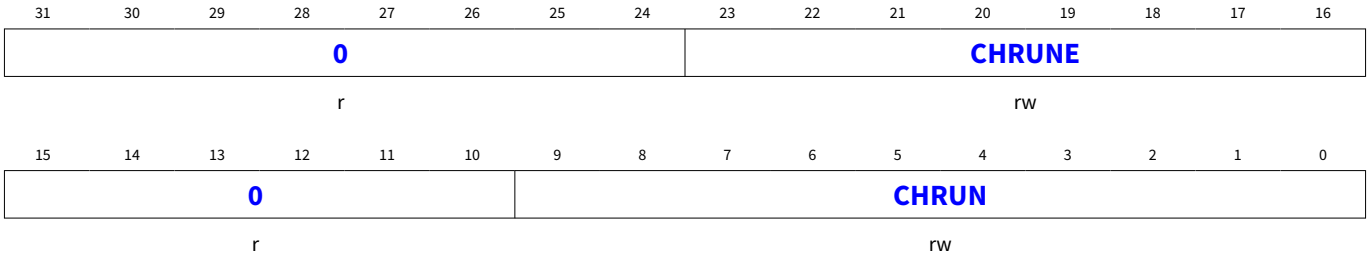
Field	Bits	Type	Description
DSADCSUP	0	rw	**DSADC supply level** With this bit-filed internal operating point of DSADC is defined by hardware based on VDDM voltage range 0_B 3.7 V <= VDDM < = 5.5 V 1_B VDDM < 3.7 V
0	3:1, 7:4, 9:8, 31:10	r	**Reserved** Read as 0; should be written with 0

45.8.82 DSADC global run control register for filter chain

Register control the run control of demodulator chain of DSADC and EXMOD channels

DSADC_GLOBRCF Offset address: $5F804_H$

DSADC global run control register for filter chain Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
CHRUN	9:0	rw	**Channel run for DSADC channels** Each bit controls a DSADC, with LSB for DSADC0 and MSB for the maximum number of DSADC in the device.
CHRUNE	23:16	rw	**Channel run for EXMOD channels** Each bit controls an EXMOD, with LSB for EXMOD0 and MSB for the maximum number of EXMOD in the device.
0	15:10, 31:24	r	**Reserved** Read as 0; should be written with 0

45.8.83 DSADC global run control register for analog modulator

Register control the run control of analog modulator for DSADC channels

DSADC_GLOBRCM Offset address: $5F808_H$

DSADC global run control register for analog modulator Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0						CHRUNM									
r						rw									

Field	Bits	Type	Description
CHRUNM	9:0	rw	**Channel run for modulator in DSADC channel** Each bit controls a DSADC, with LSB for DSADC0 and MSB for the maximum number of DSADC in the device.
0	31:10	r	**Reserved** Read as 0; should be written with 0

45.8.84 DSADC carrier generator configuration register

DSADC_CGCFG Offset address: $5F80C_H$

DSADC carrier generator configuration register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								DIVCG				SIGPOL	BREV	CGMOD	
r								rw				rw	rw	rw	

Field	Bits	Type	Description
CGMOD	1:0	rw	**Carrier generator operating mode** Stopping the carrier generator (CGMOD = 00_B) terminates the PWM output after completion of the current period (indicated by bit CGST.RUN = 0). 00_B Stopped 01_B Square wave 10_B Triangle 11_B Cosine wave
BREV	2	rw	**Bit-reverse PWM generation** 0_B Normal mode 1_B Bit-reverse mode
SIGPOL	3	rw	**Signal polarity** 0_B Normal carrier signal begins with +1 1_B Inverted carrier signal begins with -1
DIVCG	7:4	rw	**Divider factor for the PWM pattern signal generator** Defines the input frequency of the carrier signal generator, derived from the selected internal clock source: $f_{CG} = f_{ADC}$ / CGP. ***Note:*** *The frequency of the carrier signal itself is f_{CG} / 1024.* 0_H CGP = 2 … F_H CGP = 32
0	31:8	r	**Reserved** Read as 0; should be written with 0

45.8.85 DSADC carrier generator status register

Note: *Register gives the status information for the carrier generator.*

DSADC_CGST Offset address: $5F810_H$

DSADC carrier generator status register Kernel Reset value: $0000\ 01E0_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				SGNCG	STEPD	STEPS	STEPCOUNT			BITCOUNT					RUN
r				rh	rh	rh	rh			rh					rh

Field	Bits	Type	Description
RUN	0	rh	**Run indicator** 0_B Stopped (cleared at the end of a period) 1_B Running
BITCOUNT	5:1	rh	**Bit counter** Counts the 32 cycles generated for each step
STEPCOUNT	8:6	rh	**Step counter** Counts the 8 steps generated for each quadrant of the carrier signal period
STEPS	9	rh	**Step counter sign** Indicates the sign of the step counter value 0_B Step counter value is positive 1_B Step counter value is negative
STEPD	10	rh	**Step counter direction** 0_B Step counter is counting down 1_B Step counter is counting up
SGNCG	11	rh	**Sign signal from carrier generator** 0_B Positive values 1_B Negative values
0	31:12	r	**Reserved** Read as 0; should be written with 0

45.8.86 DSADC event flag register on main result register

The register capture the result events on main result register for the DSADC and EXMOD channels.

DSADC_RESMF Offset address: $5F814_H$

DSADC event flag register on main result register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								REMEVEX							
r								rwh							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0						REMEVIN									
r						rwh									

Field	Bits	Type	Description
REMEVIN	9:0	rwh	**Result event for main result register RESM of DSADC channels** Each bit corresponds to a DSADC, with LSB for DSADC0 and MSB for the maximum number of DSADC in the device.
REMEVEX	23:16	rwh	**Result event for main result register RESM of EXMOD channels** Each bit corresponds to an EXMOD, with LSB for EXMOD0 and MSB for the maximum number of EXMOD in the device.

(table continues...)

(continued)

Field	Bits	Type	Description
0	15:10, 31:24	r	**Reserved** Read as 0; should be written with 0

45.8.87 DSADC event flag register on auxiliary result register

The register capture the result events on auxiliary result register for the DSADC and EXMOD channels.

DSADC_RESAF Offset address: $5F818_H$

DSADC event flag register on auxiliary result register Kernel Reset value: $0000\ 0000_H$

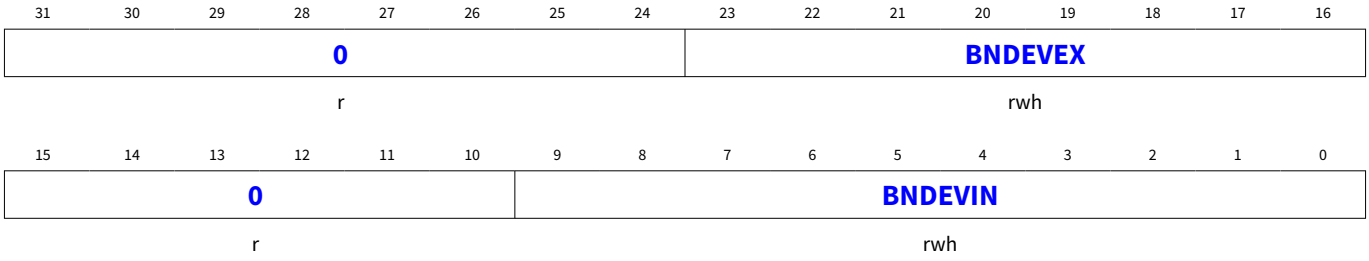
Field	Bits	Type	Description
REAEVIN	9:0	rwh	**Result event for auxiliary result register RESA of DSADC channels** Each bit corresponds to a DSADC, with LSB for DSADC0 and MSB for the maximum number of DSADC in the device.
REAEVEX	23:16	rwh	**Result event for auxiliary result register RESA of EXMOD channels** Each bit corresponds to an EXMOD, with LSB for EXMOD0 and MSB for the maximum number of EXMOD in the device.
0	15:10, 31:24	r	**Reserved** Read as 0; should be written with 0

45.8.88 DSADC boundary flag event register

Register capture the Boundary flag events for DSADC and EXMOD channels

DSADC_BOUNDF Offset address: $5F81C_H$

DSADC boundary flag event register Kernel Reset value: $0000\ 0000_H$



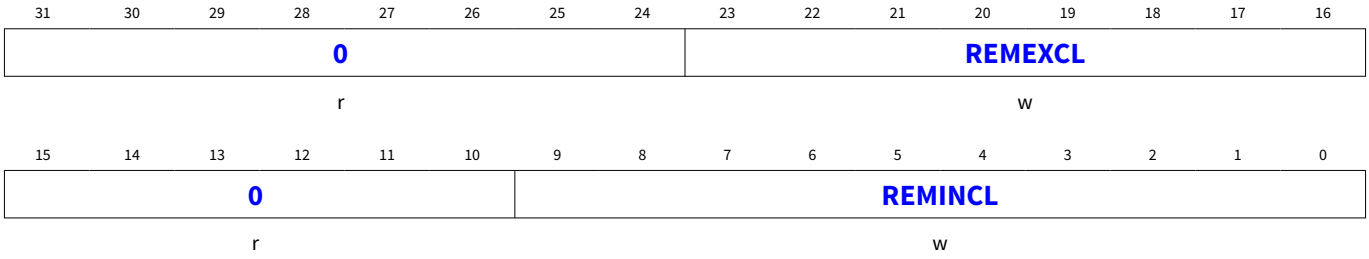
Field	Bits	Type	Description
BNDEVIN	9:0	rwh	**Boundary event for DSADC channels** Each bit corresponds to a DSADC, with LSB for DSADC0 and MSB for the maximum number of DSADC in the device.
BNDEVEX	23:16	rwh	**Boundary event for EXMOD channels** Each bit corresponds to an EXMOD, with LSB for EXMOD0 and MSB for the maximum number of EXMOD in the device.
0	15:10, 31:24	r	**Reserved** Read as 0; should be written with 0

45.8.89 DSADC result event (main) flag clear register

Clear the result event flags bits in RESMF register

DSADC_RESMFCL Offset address: $5F820_H$

DSADC result event [main) flag clear register Kernel Reset value: $0000\ 0000_H$



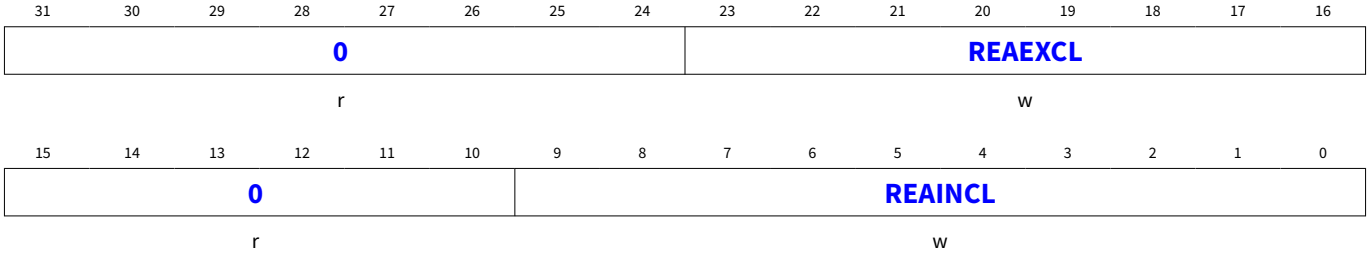
Field	Bits	Type	Description
REMINCL	9:0	w	**Result event clear for DSADC channels** Each bit corresponds to a DSADC, with LSB for DSADC0 and MSB for the maximum number of DSADC in the device.
REMEXCL	23:16	w	**Result event clear for EXMOD channels** Each bit corresponds to an EXMOD, with LSB for EXMOD0 and MSB for the maximum number of EXMOD in the device.
0	15:10, 31:24	r	**Reserved** Read as 0; should be written with 0

45.8.90 DSADC result event (auxiliary) flag clear register

Clear the result event flags bits in RESAF register

DSADC_RESAFCL Offset address: $5F824_H$

DSADC result event [auxiliary) flag clear register Kernel Reset value: $0000\ 0000_H$



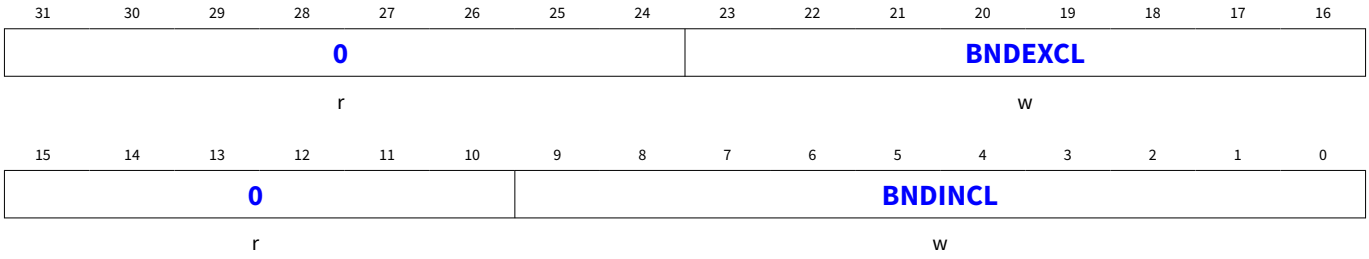
Field	Bits	Type	Description
REAINCL	9:0	w	**Auxiliary result event clear for DSADC channels** Each bit corresponds to a DSADC, with LSB for DSADC0 and MSB for the maximum number of DSADC in the device.
REAEXCL	23:16	w	**Auxiliary result event clear for EXMOD channels** Each bit corresponds to an EXMOD, with LSB for EXMOD0 and MSB for the maximum number of EXMOD in the device.
0	15:10, 31:24	r	**Reserved** Read as 0; should be written with 0

45.8.91 DSADC boundary event flag clear register

Clear the boundary event flags bits in BOUNDF register

DSADC_BOUNDFCL Offset address: $5F828_H$

DSADC boundary event flag clear register Kernel Reset value: $0000\ 0000_H$



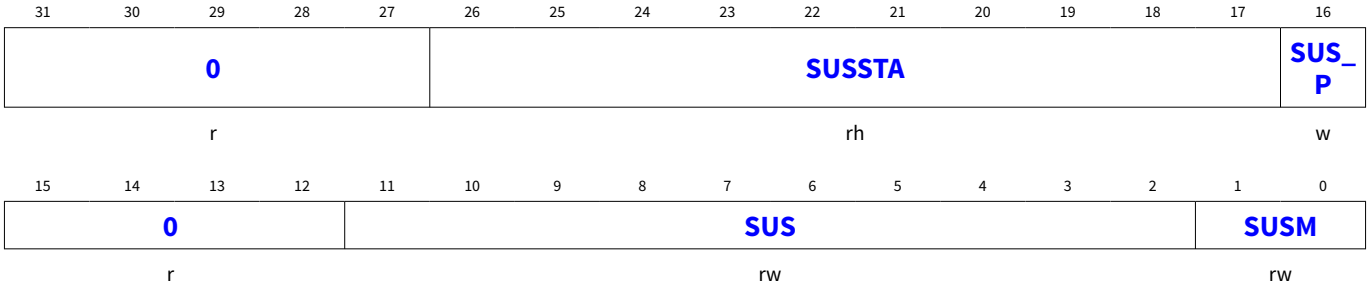
Field	Bits	Type	Description
BNDINCL	9:0	w	**Boundary event clear for DSADC channels** Each bit corresponds to a DSADC, with LSB for DSADC0 and MSB for the maximum number of DSADC in the device.
BNDEXCL	23:16	w	**Boundary event clear for EXMOD channels** Each bit corresponds to an EXMOD, with LSB for EXMOD0 and MSB for the maximum number of EXMOD in the device.
0	15:10, 31:24	r	**Reserved** Read as 0; should be written with 0

45.8.92 OCDS control and status register for DSADC

This OCDS control and status register controls the debug and trace behavior by selecting suspend modes. When OCDS is disabled the suspend control is ineffective. This register only controls all instances of DSADC. No trigger sets are supported for DSADC

DSADC_OCSDSADC Offset address: $5F82C_H$

OCDS control and status register for DSADC Debug Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
SUSM	1:0	rw	**OCDS suspend control mode - SUS** Selects the suspend mode for the module 00_B Will not suspend 01_B Hard suspend Clocks will be disabled immediately 10_B Soft suspend mode Clock will be disabled after generating the output from the auxiliary filter chain. 11_B Soft suspend mode Clock will be disabled after generating the output from the main filter chain.
SUS	11:2	rw	**OCDS suspend control** Suspends the corresponding DSADC. 0 - Module not suspended, 1 - Module suspended. Each bit controls a DSADC, with LSB for DSADC0 and MSB for the maximum number of DSADC in the device.
SUS_P	16	w	**SUS write protection** SUS and SUSM are only written when SUS_P is 1, otherwise unchanged. Read as 0.
SUSSTA	26:17	rh	**Suspend state** Suspend state of DSADC. 0 - Module not suspended. 1 - Module suspended. Each bit gives the status of a DSADC, with LSB for DSADC0 and MSB for the maximum number of DSADC in the device.
0	15:12, 31:27	r	**Reserved** Read as 0; should be written with 0.

Table 2403 Access mode restrictions of DSADC_OCSDSADC sorted by descending priority

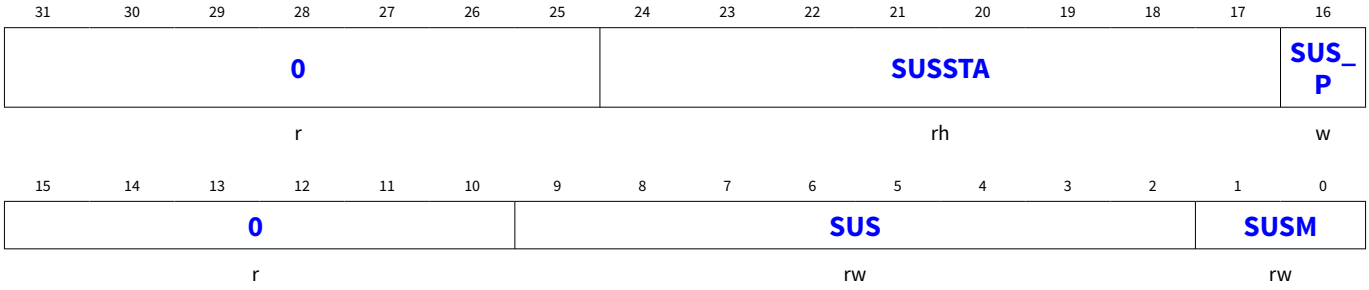
Mode name	Access mode		Description
write 1 to .**SUS_P**	rw	SUS, SUSM	Set SUS_P during write access
(default)	r	SUS, SUSM	

45.8.93 OCDS control and status register for EXMOD

This OCDS control and status register controls the debug and trace behavior by selecting suspend modes. When OCDS is disabled the suspend control is ineffective. This register only controls all instances of EXMOD. No trigger sets are supported for EXMOD

DSADC_OCSEXMOD Offset address: $5F830_H$

OCDS control and status register for EXMOD Debug Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
SUSM	1:0	rw	**OCDS suspend control mode for EXMOD - SUS** Selects the suspend mode for the Module 00_B Will not suspend 01_B Hard suspend Clocks will be disabled immediately 10_B Soft suspend mode Clock will be disabled after generating the output from the Auxiliary filter chain. 11_B Soft suspend mode Clock will be disabled after generating the output from the main filter chain.
SUS	9:2	rw	**OCDS suspend control for EXMOD** Suspends the corresponding EXMOD. 0 - Module not suspended, 1 - Module suspended. Each bit controls a EXMOD, with LSB for EXMOD0 and MSB for the maximum number of EXMOD in the device.
SUS_P	16	w	**SUS write protection** SUS and SUSM are only written when SUS_P is 1, otherwise unchanged. Read as 0.
SUSSTA	24:17	rh	**Suspend state for EXMOD** Suspend state of EXMOD. 0 - Module not suspended. 1 - Module suspended. Each bit gives the status of an EXMOD, with LSB for EXMOD0 and MSB for the maximum number of EXMOD in the device.
0	15:10, 31:25	r	**Reserved** Read as 0; should be written with 0.

Table 2404 Access mode restrictions of DSADC_OCSEXMOD sorted by descending priority

Mode name	Access mode		Description
write 1 to .**SUS_P**	rw	SUS, SUSM	Set SUS_P during write access
(default)	r	SUS, SUSM	

45.8.94 DSADCx channel configuration register

The DSADC channel configuration register selects the mode of operation of DSADC channel

- Change of modulator clock or time stamp clock division is not allowed during the run-time.
- Change in offset compensation filter during the run-time will result in invalid output and is not recommended.

DSADC_INx_CHCFG (x=0-9) Offset address: $60000_H+x*400_H$

DSADCx channel configuration register Kernel Reset value: $0000\ 0A00_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	AUTOCAL	CALIB	OFFPROT	OCEN			AMXCOPY	RDM	FIFL	SRLVL		TSCRUN	TSCLK		TSM
r	rw	w	rw	rw			rw	rw	w	rw		rw	rw		rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DRM		DIVM		DITRIM		DITHEN	0		INMAC	INSEL		INCFGN		INCFGP	
rw		rw		rw		rw	r		rw	rw		rw		rw	

Field	Bits	Type	Description
INCFGP	1:0	rw	**Configuration of positive input line** Defines the internal connection of the positive input. 00_B Input pin 01_B Reference voltage V_{AREF} 10_B VDDM/2 11_B Reference ground V_{AGND}
INCFGN	3:2	rw	**Configuration of negative input line** Defines the internal connection of the negative input. 00_B Input pin 01_B Reference voltage V_{AREF} 10_B VDDM/2 11_B Reference ground V_{AGND}
INSEL	5:4	rw	**Input pin selection** Defines the initial or permanent setting for the input multiplexer (bitfield INMUX) depending on the selected operating mode (bitfield IMXTRM). 00_B Input pin position A 01_B Input pin position B 10_B Input pin position C 11_B Input pin position D
INMAC	6	rw	**Input multiplexer action control** Defines the mechanism by which the input multiplexer is controlled. 0_B Preset mode (load INMUX upon a trigger) 1_B Single-step mode (decrement INMUX upon a trigger, wrap around to <INSEL>)

(table continues...)

(continued)

Field	Bits	Type	Description
DITHEN	9	rw	**Dithering enable** 0_B Dithering disabled 1_B Dithering enabled
DITRIM	11:10	rw	**Dithering function trim value** Controls the dithering levels for each modulator separately. For 3.3V operation, dithering levels (mV) are lower by a factor 1/3 00_B Dithering level 0 (20 mV) 01_B Dithering level 1 (40 mV) 10_B Dithering level 2 (80 mV), also default level 11_B Dithering level 3 (160 mV)
DIVM	13:12	rw	**Modulator clock frequency** Defines the frequency of the modulator clock (on-chip), derived from the peripheral clock: $f_{MOD} = f_{ADC} / 2^{(DIVM+2)}$ 00_B $f_{MOD} = f_{ADC} / 4 = 40$ MHz ($f_{ADC} = 160$ MHz) 01_B $f_{MOD} = f_{ADC} / 8 = 20$ MHz ($f_{ADC} = 160$ MHz) 10_B $f_{MOD} = f_{ADC} / 16 = 10$ MHz ($f_{ADC} = 160$ MHz) 11_B Reserved
DRM	15:14	rw	**Data read mode** Selects the data that is returned when register RESMx is read. 00_B Single: issue one 16-bit value per read access (sign on high bits) 01_B Single: issue one 16-bit value per read access (timestamp or zero on high bits) 10_B Double: issue two 16-bit values per read access 11_B Reserved
TSM	16	rw	**Time-stamp mode** 0_B No timestamp, only issue result values 1_B Insert timestamp upon the timestamp trigger
TSCLK	18:17	rw	**Timestamp counter clock selection** 00_B $f_{TSTMP} = f_{MOD}$ 01_B $f_{TSTMP} = f_{MOD} / 2$ 10_B $f_{TSTMP} = f_{MOD} / 4$ 11_B $f_{TSTMP} = f_{MOD} / 8$
TSCRUN	19	rw	**Timestamp counter run control** 0_B Timestamp counter is off 1_B Timestamp counter is counting at the rate selected by bitfield TSCLK
SRLVL	21:20	rw	**Service request FIFO Level** 00_B Generate service request when FIFO fill level reaches 1 value 01_B Generate service request when FIFO fill level reaches 2 values 10_B Generate service request when FIFO fill level reaches 3 values 11_B Generate service request when FIFO fill level reaches 4 values

(table continues...)

(continued)

Field	Bits	Type	Description
FIFL	22	w	**FIFO flush** 0_B No action 1_B Remove all entries from result FIFO
RDM	23	rw	**Result display mode** 0_B Signed mode Result values range from -2^{15} to $+2^{15}-1$ 1_B Unsigned mode Result values range from 0 to $+2^{16}-1$ (shifted by 2^{15})
AMXCOPY	24	rw	**Analog MUX setting copy enable** Allows copying of bitfield AMX into bitfield TIMESTAMP (in register TSTMPx). 0_B Do not copy, timestamp uses all 16 bits 1_B Copy AMX to bits TIMESTAMP[15:14], timestamp uses lower 14 bits
OCEN	27:25	rw	**Offset compensation filter enable** 000_B Offset compensation filter disabled, register OFFCOMP not changed 001_B Enable offset compensation filter, set cutoff frequency to rate 1 … 111_B Enable offset compensation filter, set cutoff frequency to rate 7
OFFPROT	28	rw	**Offset protection** Controls the influence of the calibration sequence on register OFFCOMP. 0_B Unprotected, calibration sequence updates bitfield OFFSET 1_B Protected, bitfield OFFSET is locked and not modified by calibration
CALIB	29	w	**Calibration trigger** 0_B No action 1_B Start the calibration algorithm now
AUTOCAL	30	rw	**Automatic calibration control** 0_B Calibration algorithm started by software (set bit CALIB) 1_B Automatically start the calibration algorithm When the selected service request gate closes
0	8:7, 31	r	**Reserved** Read as 0; should be written with 0

45.8.95 DSADCx channel status register

The DSADC channel status register gives the following information:

- Input MUX setting value
- CIC decimation count value
- CIC (auxiliary) decimation count value
- Trigger error flag status
- Calibration status

DSADC_INx_CHST (x=0-9) Offset address: $60004_H+x*400_H$

DSADCx channel status register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CAL		BNDTRER	SRTRER	IMXTRER	TSTRER	INTTRER	WRERR	RDERR	CFADCNT						
rh		rh	rh	rh	rh	rh	rh	rh	rh						

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CFMDCNT										INMUX		0	FILL		
rh										rh		r	rh		

Field	Bits	Type	Description
FILL	2:0	rh	**FIFO fill level** Not listed combinations are reserved. 000_B Result FIFO is empty 001_B Result FIFO contains 1 valid value 010_B Result FIFO contains 2 valid values 011_B Result FIFO contains 3 valid values 100_B Result FIFO contains 4 valid values
INMUX	5:4	rh	**Input multiplexer setting** Indicates the current setting of the input multiplexer connecting the input pins to the buffer inputs. The product-specific appendix details the available channels and their inputs. 00_B Input pin position A 01_B Input pin position B 10_B Input pin position C 11_B Input pin position D
CFMDCNT	15:6	rh	**CIC filter decimation counter** The decimation counter counts the filter cycles until an output is generated, i.e. the oversampling rate. CFMDCNT counts down from the respective start value.
CFADCNT	22:16	rh	**CIC filter (auxiliary) decimation counter** The decimation counter counts the filter cycles until an output is generated, i.e. the oversampling rate.

(table continues...)

(continued)

Field	Bits	Type	Description
RDERR	23	rh	**Read error flag** 0_B No problem encountered 1_B A read access occurred while the FIFO was empty A read error is also indicated when a read access occurs during the FIFO's synchronization stall phase (4 SPB clock cycles after a read access). Clear this sticky flag by writing 1 to bit RDEC.
WRERR	24	rh	**Write error flag** 0_B No problem encountered 1_B A write access occurred while the FIFO was full Clear this sticky flag by writing 1 to bit WREC.
INTTRER	25	rh	**Integrator trigger error flag** This bit tells the status of the integrator trigger error flag. If the integrator trigger could not be executed the bit will be set to 1
TSTRER	26	rh	**Timestamp trigger error flag** This bit tells the status of the timestamp trigger error flag. If the timestamp trigger could not be executed the bit will be set to 1
IMXTRER	27	rh	**Input MUX trigger error flag** This bit tells the status of the analog input mux trigger error flag. If the input mux trigger could not be executed the bit will be set to 1
SRTRER	28	rh	**Service request trigger error flag** This bit tells the status of the service request trigger error flag. If the service request trigger could not be executed the bit will be set to 1
BNDTRER	29	rh	**Boundary update trigger error flag** This bit tells the status of the boundary update trigger error flag. If the boundary update trigger could not be executed the bit will be set to 1
CAL	31:30	rh	**Calibration status flag** ***Note**:* *Bitfield CAL is set to 01_B in the next clock cycle after setting bit CALIB or after detecting the selected trigger (if auto-calibration is activated).* 00_B Uncalibrated, initial state after reset 01_B The calibration algorithm is currently running 10_B Calibrated, normal operation is possible 11_B Calibration terminated incorrectly
0	3	r	**Reserved** Read as 0; should be written with 0

45.8.96 DSADCx common mode voltage register

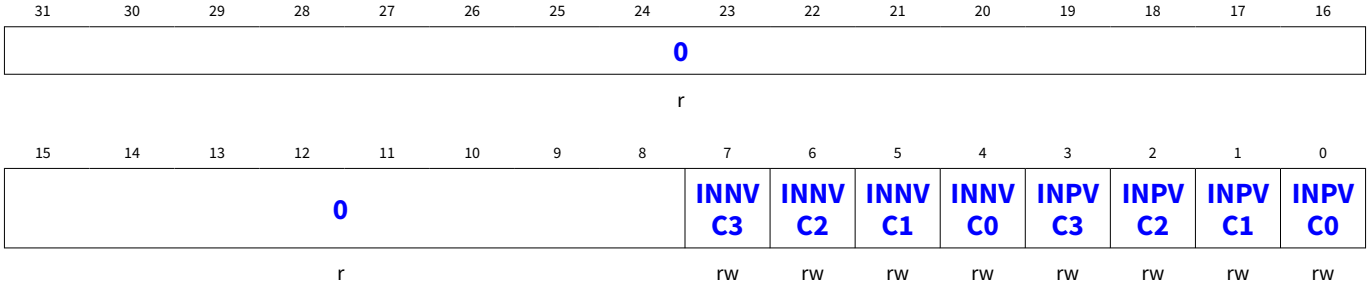
The common mode voltage registers enable the voltage generators and select which input pins are connected to the common mode hold voltage.

Only those bits are valid that correspond to an existing input pin (see datasheet for pinning).

Changes to this register during run time may result in invalid outputs. User shall avoid this.

DSADC_INx_VCM (x=0-9) Offset address: $60008_H+x*400_H$

DSADCx common mode voltage register Kernel Reset value: $0000\ 0000_H$



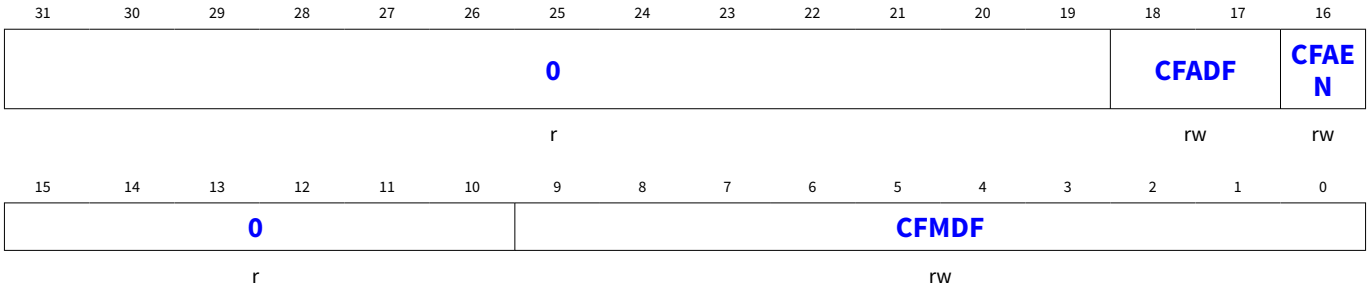
Field	Bits	Type	Description
INPVCy (y=0-3)	y	rw	**Voltage control of positive inputs y of CHx** Defines the connection of the respective positive input y to the common mode voltage. y indicates the input of the analog multiplexers (if available). 0_B No connection to common mode voltage 1_B This pin is connected to the common mode voltage
INNVCy (y=0-3)	y+4	rw	**Voltage control of negative inputs y of CHx** Defines the connection of the respective negative input y to the common mode voltage. y indicates the input of the analog multiplexers (if available). 0_B No connection to common mode voltage 1_B This pin is connected to the common mode voltage
0	31:8	r	**Reserved** Read as 0; should be written with 0

45.8.97 DSADCx CIC configuration register

Change of the decimation factor for CIC or AUX CIC during run time shall be avoided.

DSADC_INx_CICCFG (x=0-9) Offset address: $6000C_H+x*400_H$

DSADCx CIC configuration register Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
CFMDF	9:0	rw	**CIC filter decimation factor** Defines the oversampling rate of the CIC filter: OSR = CFMDF + 1. Valid values are 003_H to $3FF_H$ (OSR = 4 to 1024).

(table continues...)

(continued)

Field	Bits	Type	Description
CFAEN	16	rw	**CIC filter (auxiliary) enable** 0_B Off: auxiliary filter is not active 1_B Auxiliary filter is active and generates results and alarm events
CFADF	18:17	rw	**CIC filter (auxiliary) decimation factor** 00_B OSR = 16 01_B OSR = 32 10_B OSR = 64 11_B OSR = 128
0	15:10, 31:19	r	**Reserved** Read as 0; should be written with 0

45.8.98 DSADCx integrator configuration register

Configures the hardware integrator and the mode of operation of the integrator.
Integrator configuration change shall be avoided when INTEN = 1, if changed, may result in invalid outputs.

DSADC_INx_INTCFG (x=0-9) Offset address: $60010_H+x*400_H$

DSADCx integrator configuration register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
INTEN	NVALINT							0	NVALDIS						
rw	rw							r	rw						

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RUM	0			REPVAL				FRC		IWS		0	ISC		
rw	r			rw				rw		rw		r	rw		

Field	Bits	Type	Description
ISC	2:0	rw	**Integrator shift control** Controls the data shifter after the integrator that selects the portion of the integrator data for the result register. 111_B is reserved. ***Note**: ISC selects the respective bits in an internal register. The lowest selected bit is used for rounding and is then removed.* 000_B Select bits 4 … 20 (integration of 2 values) 001_B Select bits 5 … 21 (integration of 3 to 4 values) … 110_B Select bits 10 … 26 (integration of 65 to 128 values)

(table continues...)

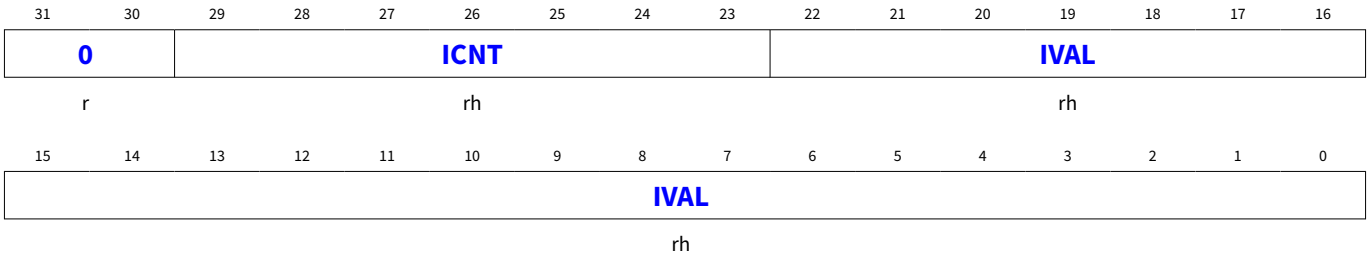
(continued)

Field	Bits	Type	Description
IWS	5:4	rw	**Integration window size** 00_B Internal control: stop integrator after REPVAL+1 integration cycles 01_B External control: stop integrator upon the inverse trigger event 10_B Software control Stop integrator when INTEN = 0 or INTTRM = 00 (when changing INTTRDLY during runtime the integration window restarts with the next selected HW trigger edge) 11_B Reserved
FRC	7:6	rw	**Filter chain restart control** 00_B Restart the main filter chain when an integration window starts, auxiliary filter chain is not restarted 01_B Restart the main and auxilary filter chain when an integration window starts 10_B No influence on main and auxiliary filter chain when an integration window starts, except for the integrator itself 11_B No influence on main filter chain when an integration window starts, except for the integrator itself and auxiliary filter chain
REPVAL	11:8	rw	**Number of integration cycles** Defines the number of integration cycles to be counted by REPCNT if activated (IWS = 0). The number of cycles is REPVAL+1.
RUM	15	rw	**Result update mode** Configures the result update behaviour 0_B Results generated independent of integration window 1_B Results generated only during integration window
NVALDIS	22:16	rw	**Number of values discarded** Start the integration cycle after NVALDIS values
NVALINT	30:24	rw	**Number of values to be accumulated** Stop the integration cycle after NVALINT+1 values ***Note**: Use intervals of 2 minimum, so no data is lost due to the data shifter.*
INTEN	31	rw	**Integrator Enable** Controls if the integrator is enabled or disabled
0	3, 14:12, 23	r	**Reserved** Read as 0; should be written with 0

45.8.99 DSADCx intermediate integration value register

DSADC_INx_INTIVAL (x=0-9) Offset address: $60014_H+x*400_H$

DSADCx intermediate integration value register Kernel Reset value: $0000\ 0000_H$

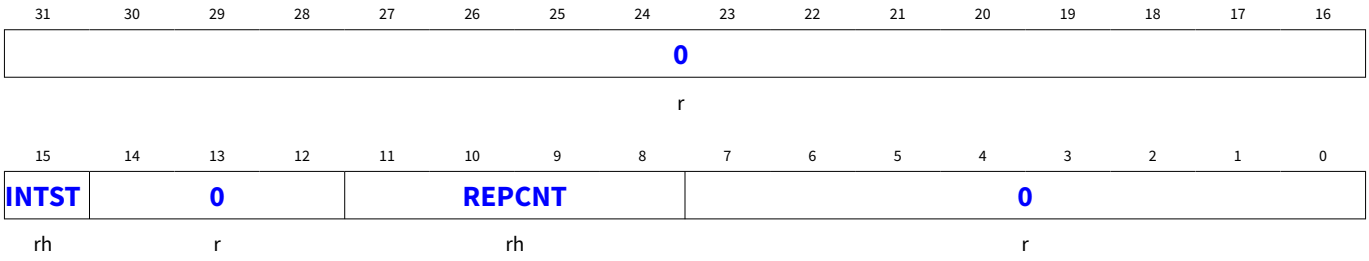


Field	Bits	Type	Description
IVAL	22:0	rh	**Integrator intermediate value** These bits output the intermediate accumulated value.
ICNT	29:23	rh	**Integrator intermediate count** The bits output the intermediate number of accumulated values or the number of discarded samples during integrator discard phase.
0	31:30	r	**Reserved** Read as 0; should be written with 0

45.8.100 DSADCx integrator status register

DSADC_INx_INTST (x=0-9) Offset address: $60018_H+x*400_H$

DSADCx integrator status register Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
REPCNT	11:8	rh	**Integration cycle counter** Counts the number of integration cycles if activated (IWS = 0). This number is selected via bitfield REPVAL.
INTST	15	rh	**Integration enable status- INTST** Indicates the activity of the integrator. 0_B Integration stopped INTST is cleared at the end of the integration window, i.e. after REPVAL cycles or upon the inverse trigger event transition of the external trigger signal. 1_B Integration enabled INTST is set when the channel is started while permanent integration is selected (ITRMODE = 11_B) or upon the defined trigger event.

(table continues...)

(continued)

Field	Bits	Type	Description
0	7:0, 14:12, 31:16	r	**Reserved** Read as 0; should be written with 0

45.8.101 DSADCx gain calibration register

DSADC_INx_GAINCAL (x=0-9) Offset address: $6001C_H+x*400_H$

DSADCx gain calibration register Kernel Reset value: $61A8\ 1000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	CALTARGET														
r	rw														

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	CALFACTOR														
r	rwh														

Field	Bits	Type	Description
CALFACTOR	12:0	rwh	**Multiplication factor for gain calibration** The resulting factor is (<CALFACTOR> / 4 096) ***Note**: The initial value of 4 096 (1000_H) corresponds to a factor of 1.000.*
CALTARGET	30:16	rw	**Target value for calibrated fullscale** Defines the target value for the calibration algorithm. ***Note**: The initial value of 25 000 ($61A8_H$) corresponds to 0.2 mV per LSB.*
0	15:13, 31	r	**Reserved** Read as 0; should be written with 0

45.8.102 DSADCx gain control register

DSADC_INx_GAINCTR (x=0-9) Offset address: $60020_H+x*400_H$

DSADCx gain control register Kernel Reset value: $0000\ 1000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0							CICDEC	0			CALCICS				
r							rw	r			rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	CALGFAC														
r	rw														

Field	Bits	Type	Description
CALGFAC	12:0	rw	**Multiplication factor for gain correction during calibration** The resulting factor is (<CALGFAC> / 4 096)
CALCICS	20:16	rw	**Position of the CIC filter output shifter during calibration** Selects the valid outputs bits from the CIC filter, depending on the chosen decimation factor (see data shifter formula), $1D_H$ … $1F_H$ are reserved. 00_H Use bits 0 ... 16 … $1C_H$ Use bits 28 ... 44
CICDEC	24	rw	**Decimation rate of the CIC filter during calibration** 0_B CIC decimation of 128 during calibration 1_B CIC decimation of 256 during calibration
0	15:13, 23:21, 31:25	r	**Reserved** Read as 0; should be written with 0

45.8.103 DSADCx gain correction register

DSADC_INx_GAINCORR (x=0-9) Offset address: $60024_H+x*400_H$

DSADCx gain correction register Kernel Reset value: $0000\ 1000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0											CICSHIFT				
r											rw				

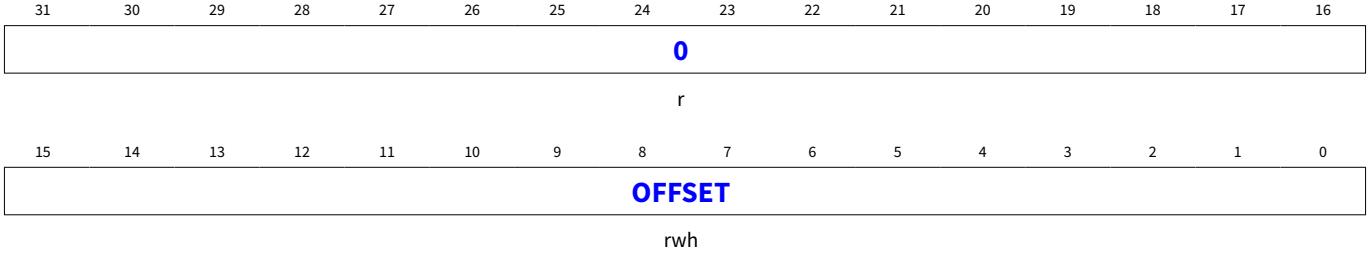
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			GAINFACTOR												
r			rw												

Field	Bits	Type	Description
GAINFACTOR	12:0	rw	**Multiplication factor for gain correction** The resulting factor is (<GAINFACTOR> / 4 096)
CICSHIFT	20:16	rw	**Position of the CIC filter output shifter** Selects the valid outputs bits from the CIC filter $1D_H$ … $1F_H$ are reserved. 00_H Use bits 0 ... 16 … $1C_H$ Use bits 28 ... 44
0	15:13, 31:21	r	**Reserved** Read as 0; should be written with 0

45.8.104 DSADCx offset compensation register

DSADC_INx_OFFCOMP (x=0-9) Offset address: $60028_H+x*400_H$

DSADCx offset compensation register Kernel Reset value: $0000\ 0000_H$

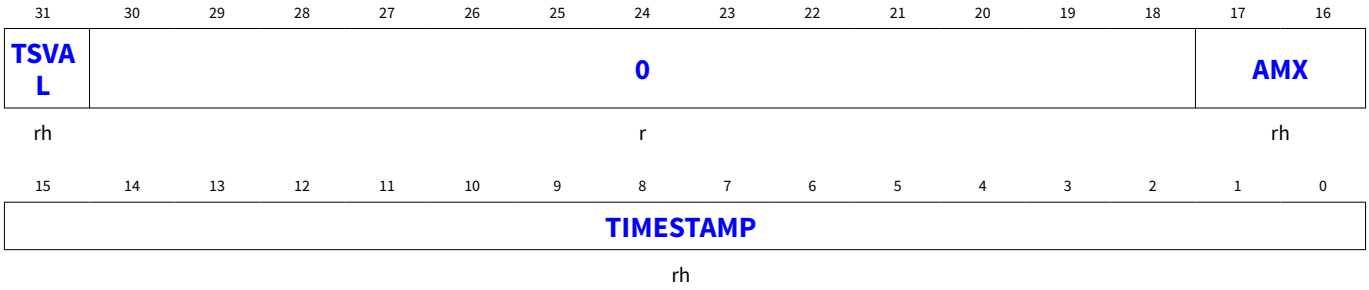


Field	Bits	Type	Description
OFFSET	15:0	rwh	**Offset value** Half of this signed value is subtracted from each result produced by the filter chain. ***Note**: Bit 0 represents 1/2 LSB. This increases the precision in case of accumulated result values, e.g. in the integrator.*
0	31:16	r	**Reserved** Read as 0; should be written with 0

45.8.105 DSADCx timestamp register

DSADC_INx_TSTMP (x=0-9) Offset address: $6002C_H+x*400_H$

DSADCx timestamp register Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
TIMESTAMP	15:0	rh	**The most recent captured timestamp value** This value is copied from the timestamp counter TSCOUNT ***Note**: If bit AMXCOPY in register TSCNT is 1, bits TIMESTAMP[15:14] are replaced with a copy of bitfield AMX.*
AMX	17:16	rh	**Analog multiplexer setting** This value is copied from bitfield INMUX in register CHCFGx

(table continues...)

(continued)

Field	Bits	Type	Description
TSVAL	31	rh	**Timestamp valid** Indicates valid timestamp information. 0_B No timestamp trigger occurred since last read access 1_B Timestamp information has been stored after a timestamp trigger
0	30:18	r	**Reserved** Read as 0; should be written with 0

45.8.106 DSADCx timestamp counter

DSADC_INx_TSCNT (x=0-9) Offset address: $60030_H+x*400_H$

DSADCx timestamp counter Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TSCOUNT															
rh															

Field	Bits	Type	Description
TSCOUNT	15:0	rh	**Timestamp counter value** TSCOUNT is clocked with the modulator clock and is cleared when a new result value has been generated.
0	31:16	r	**Reserved** Read as 0; should be written with 0

45.8.107 DSADCx result register main

DSADC_INx_RESM (x=0-9) Offset address: $60034_H+x*400_H$

DSADCx result register main Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESULTHI															
rh															

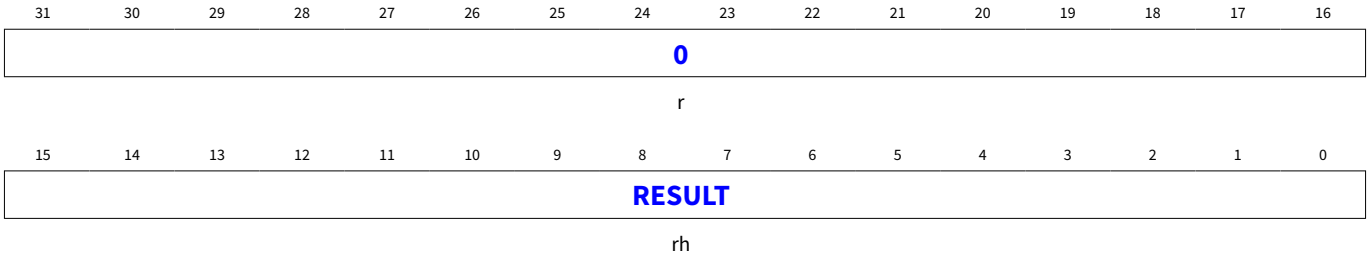
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESULTLO															
rh															

Field	Bits	Type	Description
RESULTLO	15:0	rh	**Result value lower part** Returns the next value from the result FIFO
RESULTHI	31:16	rh	**Result value higher part** Returns an additional value

45.8.108 DSADCx result register auxiliary

DSADC_INx_RESA (x=0-9) Offset address: $60038_H+x*400_H$

DSADCx result register auxiliary Kernel Reset value: $0000\ 0000_H$

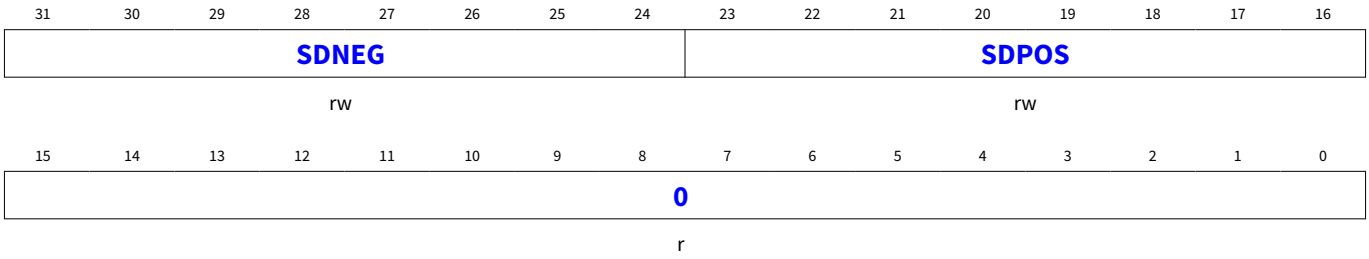


Field	Bits	Type	Description
RESULT	15:0	rh	**Most recent result of auxiliary filter**
0	31:16	r	**Reserved** Read as 0; should be written with 0

45.8.109 DSADCx carrier generator synchronization register

DSADC_INx_CGSYNC (x=0-9) Offset address: $6003C_H+x*400_H$

DSADCx carrier generator synchronization register Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
SDPOS	23:16	rw	**Sign delay value for positive halfwave** Defines the content of SDCOUNT to generate a negative delayed sign signal (SGND).
SDNEG	31:24	rw	**Sign delay value for negative halfwave** Defines the content of SDCOUNT to generate a positive delayed sign signal (SGND).

(table continues...)

(continued)

Field	Bits	Type	Description
0	15:0	r	**Reserved** Read as 0; should be written with 0

45.8.110 DSADCx rectification configuration register

DSADC_INx_RECTCFG (x=0-9) Offset address: $60040_H+x*400_H$

DSADCx rectification configuration register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				SSCH				0		SSRC		0		SDUM	RFEN
r				rw				r		rw		r		rw	rw

Field	Bits	Type	Description
RFEN	0	rw	**Rectification enable** General control of the rectifier circuit. ***Note:*** *Rectification is only active while the integrator is active.* 0_B No rectification, data not altered 1_B Data are rectified according to SGND
SDUM	1	rw	**Sign delay update mode** Controls behaviour of sign delay value updates for the carrier synchronization. 0_B Sign delay values become effective immediately 1_B Sign delay values become effective after sign update
SSRC	5:4	rw	**Sign source** Selects the sign signal that is to be delayed. 00_B On-chip carrier generator 01_B Sign of result of channel selected by bitfield SSCH 10_B External sign signal A 11_B External sign signal B
SSCH	11:8	rw	**Sign source channel** Selects the channel providing the sign signal if SSRC = 01_B. Other products of the family may have less channels and, consequently, less valid SSCH codes. Not listed combinations are reserved. 0_H Sign result from channel 0 … 9_H Sign result from channel 9

(table continues...)

(continued)

Field	Bits	Type	Description
0	3:2, 7:6, 31:12	r	**Reserved** Read as 0; should be written with 0

45.8.111 DSADCx boundary select register

DSADC_INx_BOUNDSEL (x=0-9) Offset address: $60044_H+x*400_H$

DSADCx boundary select register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BOUNDARYU															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BOUNDARYL															
rw															

Field	Bits	Type	Description
BOUNDARYL	15:0	rw	**Lower boundary value for limit checking** This (two's complement) value is compared to the upper 16 bits of the CIC filter results.
BOUNDARYU	31:16	rw	**Upper boundary value for limit checking** This (two's complement) value is compared to the upper 16 bits of the CIC filter results.

45.8.112 DSADCx integrator trigger configuration register

Configure the trigger source for doing trigger based integration.

DSADC_INx_INTTRCFG (x=0-9) Offset address: $60048_H+x*400_H$

DSADCx integrator trigger configuration register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
INTTRDLY															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0						INTTRM		0		INTTRS					
r						rw		r		rw					

Field	Bits	Type	Description
INTTRS	5:0	rw	**Trigger source select for integrator** Refer "Hardware triggers to ADC" about the trigger source definition. [1)]

(table continues...)

(continued)

Field	Bits	Type	Description
INTTRM	9:8	rw	**Trigger mode select** 00_B No trigger edge selected, integrator bypassed 01_B Rising trigger edge selected 10_B Falling trigger edge selected 11_B No trigger, integrator active all the time
INTTRDLY	31:16	rw	**Trigger delay** Controls how much the selected trigger(s) shall be delayed before being executed The period of the configured edge(s) shall be greater than the configured delay value, else an error event will occur
0	7:6, 15:10	r	**Reserved** Read as 0; should be written with 0

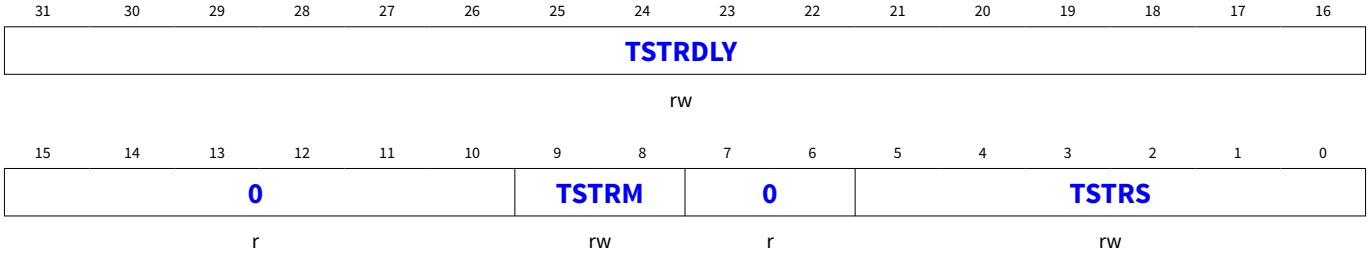
1) To avoid unintended triggers, select the trigger source first before enabling the corresponding function.

45.8.113 DSADCx timestamp trigger configuration register

Configure the trigger source for generating timestamp with hardware trigger.

DSADC_INx_TSTRCFG (x=0-9) Offset address: $6004C_H+x*400_H$

DSADCx timestamp trigger configuration register Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
TSTRS	5:0	rw	**Trigger source select** Selects an input for the trigger signal used for timestamp. *1)*
TSTRM	9:8	rw	**Trigger mode select** 00_B No trigger edge selected 01_B Rising trigger edge selected 10_B Falling trigger edge selected 11_B Both trigger edges selected
TSTRDLY	31:16	rw	**Trigger delay** Controls how much the selected trigger shall be delayed before being executed The period of the configured edge(s) shall be greater than the configured delay value, else an error event will occur
0	7:6, 15:10	r	**Reserved** Read as 0; should be written with 0

1) To avoid unintended triggers, select the trigger source first before enabling the corresponding function.

45.8.114 DSADCx input mux trigger configuration register

Configure the trigger source for changing the input of the analog input MUX.

DSADC_INx_IMXTRCFG (x=0-9) Offset address: $60050_H+x*400_H$

DSADCx input mux trigger configuration register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
IMXTRDLY															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0						IMXTRM		0		IMXTRS					
r						rw		r		rw					

Field	Bits	Type	Description
IMXTRS	5:0	rw	**Trigger source select for switching analog input MUX** Refer "Hardware triggers to ADC" about the trigger source definition 1)
IMXTRM	9:8	rw	**Trigger mode select** 00_B Software control, value configured in INSEL is taken 01_B Rising trigger edge selected 10_B Falling trigger edge selected 11_B Both trigger edges selected
IMXTRDLY	31:16	rw	**Trigger delay** Controls how much the selected trigger shall be delayed before being executed The period of the configured edge(s) shall be greater than the configured delay value, else an error event will occur
0	7:6, 15:10	r	**Reserved** Read as 0; should be written with 0

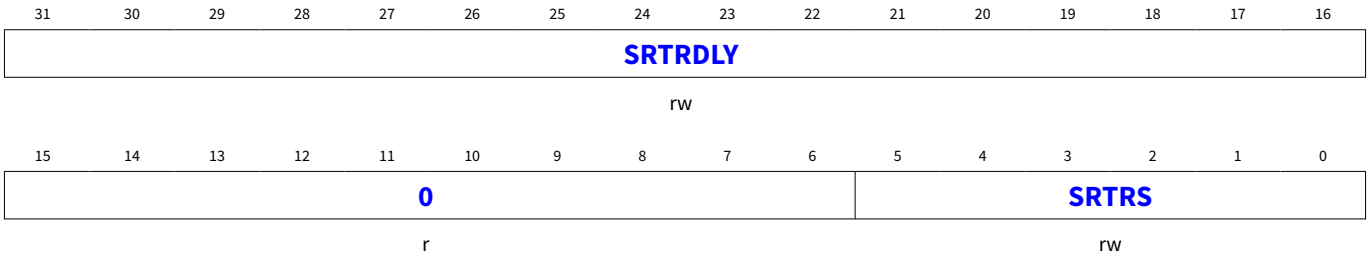
1) To avoid unintended triggers, select the trigger source first before enabling the corresponding function.

45.8.115 DSADCx service request trigger configuration register

Configures the trigger source and trigger delay for the service request trigger. Please also refer to DSADC_INx_SRCFG.SR0E register for configuring the proper mode of trigger.

DSADC_INx_SRTRCFG (x=0-9) Offset address: $60054_H+x*400_H$

DSADCx service request trigger configuration register Kernel Reset value: $0000\ 0000_H$



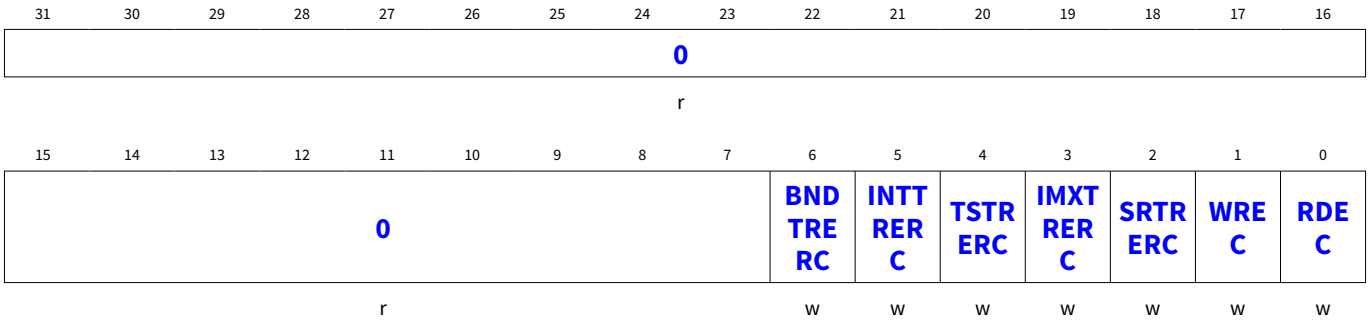
Field	Bits	Type	Description
SRTRS	5:0	rw	**Trigger source select for service request** Refer "Hardware triggers to ADC" about the trigger source definition. 1)
SRTRDLY	31:16	rw	**Trigger delay** Controls how much the selected trigger shall be delayed before being executed The period of the configured edge(s) shall be greater than the configured delay value, else an error event will occur
0	15:6	r	**Reserved** Read as 0; should be written with 0

1) To avoid unintended triggers, select the trigger source first before enabling the corresponding function.

45.8.116 DSADCx channel error clear register

DSADC_INx_CHERRCL (x=0-9) Offset address: $60058_H+x*400_H$

DSADCx channel error clear register Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
RDEC	0	w	**Read error flag clear** 0_B No action 1_B Clear flag RDERR
WREC	1	w	**Write error flag clear** 0_B No action 1_B Clear flag WRERR

(table continues...)

(continued)

Field	Bits	Type	Description
SRTRERC	2	w	**Service request trigger error clear** 0_B No action 1_B Clear flag SRTRER
IMXTRERC	3	w	**Analog input mux trigger error clear** 0_B No action 1_B Clear flag IMXTRER
TSTRERC	4	w	**Time stamp trigger error clear** 0_B No action 1_B Clear flag TSTRER
INTTRERC	5	w	**Integrator trigger error clear** 0_B No action 1_B Clear flag INTTRER
BNDTRERC	6	w	**Boundary request trigger error clear** 0_B No action 1_B Clear flag BNDTRER
0	31:7	r	**Reserved** Read as 0; should be written with 0

45.8.117 DSADCx service request configuration register

Configure and enable the service requests generated by DSADC channel towards IR.

DSADC_INx_SRCFG (x=0-9) Offset address: $6005C_H+x*400_H$

DSADCx service request configuration register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SR3E		0			SR2E	0					SR1E	0		SR0E	
rw		r			rw	r					rw	r		rw	

Field	Bits	Type	Description
SR0E	1:0	rw	**Service request 0 enable and configuration** 00_B Never 01_B Generate service request only while gate is high 10_B Generate service request only while gate is low 11_B Always
SR1E	4	rw	**Service Request 1 enable** Enable and disable for the service request 1. SR1 is generated at every auxiliary filter chain result

(table continues...)

(continued)

Field	Bits	Type	Description
SR2E	10	rw	**Service request 2 enable** Enable and disable for service request 2. SR2 is generated based on the configured boundary check through SR02CFG and SR02GT in BNDCFG register
SR3E	15:14	rw	**Service request 3 enable and configuration** 00_B Never 01_B Always for timestamp events 10_B Always for CGSYNC events 11_B Reserved
0	3:2, 9:5, 13:11, 31:16	r	**Reserved** Read as 0; should be written with 0

45.8.118 DSADCx integrator operational amplifier configuration

Adjust current and bandwidth of integrator operational amplifier inside on-chip modulator

DSADC_INx_OTA (x=0-9) Offset address: $60060_H+x*400_H$

DSADCx integrator operational amplifier configuration Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															MSCEN
r															rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			SLICEN					0			SLICEP				
r			rw					r			rw				

Field	Bits	Type	Description
SLICEP	4:0	rw	**Positive slice configuration**
SLICEN	12:8	rw	**Negative slice configuration**
MSCEN	16	rw	**DSADCx manual slice control enable** Enable manual adjustment of current and bandwidth of integrator operational amplifier inside on-chip modulator 0_B Disabled, automatic control of current and bandwidth by bitfield DIVM 1_B Enabled, control by bitfields SLICEP and SLICEN
0	7:5, 15:13, 31:17	r	**Reserved** Read as 0; should be written with 0

45.8.119 DSADCx carrier generation and rectification status register

DSADC_INx_CGRECST (x=0-9)

DSADCx carrier generation and rectification status register

Offset address: $60064_H+x*400_H$

Kernel Reset value: $8000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SGND	SGNCS	0													SDCVAL
rh	rh	r													rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SDCAP								SDCOUNT							
rh								rh							

Field	Bits	Type	Description
SDCOUNT	7:0	rh	**Sign delay counter** Counts the result values from the filter chain to delay the carrier sign signal
SDCAP	15:8	rh	**Sign delay capture value** Indicates the result values counted between the begin of the positive halfwave of the carrier signal and the first received positive value.
SDCVAL	16	rh	**Valid flag** Indicates a new value in bitfield SDCAP. 0_B No new result available 1_B Bitfield SDCAP has been updated with a new captured value and has not yet been read
SGNCS	30	rh	**Selected carrier sign signal** 0_B Positive values 1_B Negative values
SGND	31	rh	**Sign signal delayed** 0_B Positive values 1_B Negative values
0	29:17	r	**Reserved** Read as 0; should be written with 0

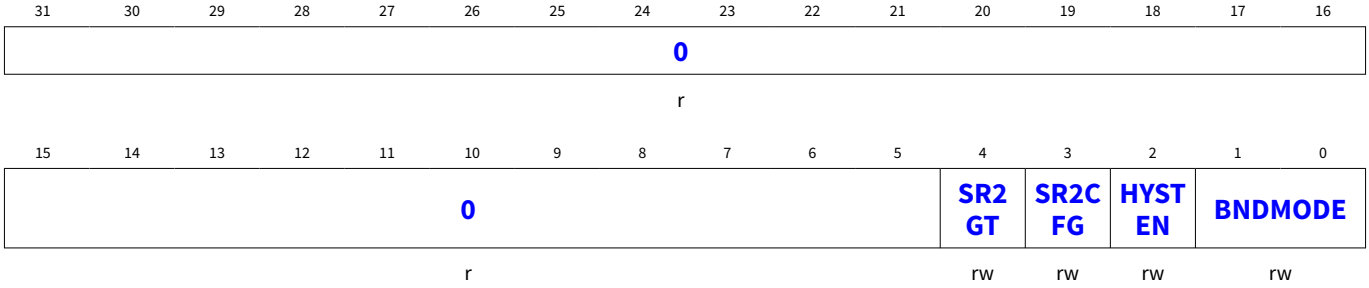
45.8.120 DSADCx boundary mode configuration register

DSADC_INx_BNDCFG (x=0-9)

DSADCx boundary mode configuration register

Offset address: $60068_H+x*400_H$

Kernel Reset value: $0000\ 0000_H$



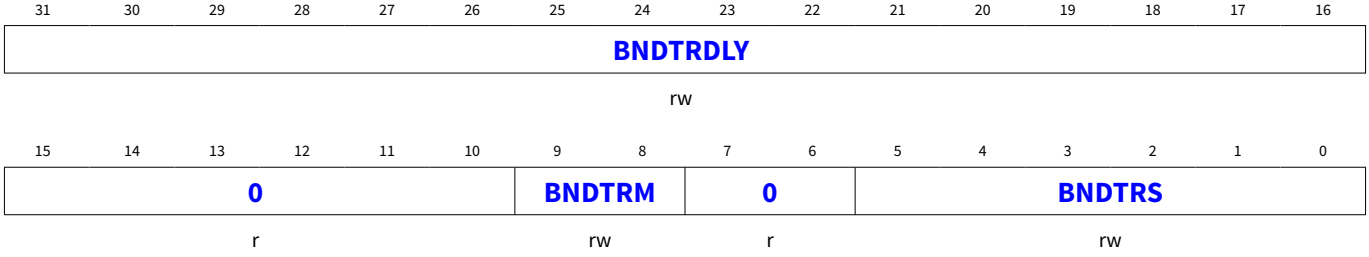
Field	Bits	Type	Description
BNDMODE	1:0	rw	**Boundary flag mode - BNDMOD** Configure the condition under which boundary flag is generated 00_B Disabled 01_B Boudary flag is high when result exceeds upper limit 10_B Boudary flag is high when result is lower than lower limit 11_B Boudary flag is high when result is out of the band
HYSTEN	2	rw	**Hysteresis enable** Enables the Hysteresis Mode. In this mode BNDMODE configuration is ignored
SR2CFG	3	rw	**Conditions under which SR2 is generated** 0_B If the result is inside boundary band 1_B If the result is outisde the boudary band
SR2GT	4	rw	**Service request gate for SR2** 0_B Service request 2 gate disabled 1_B Service request 2 is only generated when the integrator is enabled and after the discard of values defined by NVALDIS
0	31:5	r	**Reserved** Read as 0; should be written with 0

45.8.121 DSADCx boundary request trigger configuration register

Configure the trigger source for updating the boundary value for limit checking.

DSADC_INx_BNDTRCFG (x=0-9) Offset address: $6006C_H+x*400_H$

DSADCx boundary request trigger configuration register Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
BNDTRS	5:0	rw	**Trigger source select for updating the boundary value.** Refer "Hardware triggers to ADC" about the trigger source definition. 1)
BNDTRM	9:8	rw	**Trigger mode select** 00_B No trigger selected, boundary update immediately 01_B Boundary update on the rising edge of the trigger 10_B Boundary update on the falling edge of the trigger 11_B Boundary update at both rising and falling edges of the trigger
BNDTRDLY	31:16	rw	**Trigger delay** Controls how much the selected trigger shall be delayed before being executed The period of the configured edge(s) shall be greater than the configured delay value, else an error event will occur
0	7:6, 15:10	r	**Reserved** Read as 0; should be written with 0

1) To avoid unintended triggers, select the trigger source first before enabling the corresponding function.

45.8.122 DSADCx boundary status register

Boundary status register gives the current used boundary values for limit checking

DSADC_INx_BOUNDST (x=0-9) Offset address: $60070_H+x*400_H$

DSADCx boundary status register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UPPERST															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LOWERST															
rh															

Field	Bits	Type	Description
LOWERST	15:0	rh	**Lower boundary value currently used for limit checking** The current used lower boundary value
UPPERST	31:16	rh	**Upper boundary value currently used for limit checking** The current used upper boundary value

45.8.123 EXMODz channel configuration register

The channel configuration register selects the operation mode of the external modulator channel.

DSADC_EXz_CHCFG (z=0-7) Offset address: $63800_H+z*400_H$

EXMODz channel configuration register Kernel Reset value: $0000\ 3010_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MANEN	MANCLK	0						RDM	FIFL	SRLVL		TSCRUN	TSCLK		TSM
rw	rw	r						rw	w	rw		rw	rw		rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DRM		DIVM		CSR				0		DSS		0	DSR		
rw		rw		rw				r		rw		r	rw		

Field	Bits	Type	Description
DSR	2:0	rw	**Data source for external modulator** 000_B External, from input A, direct 001_B External, from input B, direct 010_B External, from input C, direct 011_B External, from input D, direct 100_B External, from input E, direct 101_B External, from input F, direct 110_B External, from input G, direct 111_B External, from input H, direct
DSS	5:4	rw	**Data stream select** 00_B Use each rising clock edge 01_B Use each falling clock edge 10_B Use every 2nd rising clock edge 11_B Use every 2nd falling clock edge
CSR	11:8	rw	**Clock source for external modulator** 0_H Clock from input A 1_H Clock from input B 2_H Clock from input C 3_H Clock from input D 4_H Clock from input E 5_H Clock from input F 6_H Clock from input G 7_H Clock from input H 8_H Clock from internal 9_H Reserved … F_H Reserved
DIVM	13:12	rw	**External modulator clock frequency** Defines the frequency of the external (off-chip) modulator clock, derived from the peripheral clock: $f_{MOD} = f_{ADC} / 2^{(DIVM+1)}$ 00_B Reserved 01_B $f_{MOD} = f_{ADC} / 4$ 10_B $f_{MOD} = f_{ADC} / 8$ 11_B $f_{MOD} = f_{ADC} / 16$

(table continues...)

(continued)

Field	Bits	Type	Description
DRM	15:14	rw	**Data read mode** Selects the data that is returned when register RESMx is read. 00_B Single: issue one 16-bit value per read access (sign on high bits) 01_B Single: issue one 16-bit value per read access (timestamp or zero on high bits) 10_B Double: issue two 16-bit values per read access 11_B Reserved
TSM	16	rw	**Time-stamp mode** Selects the time-stamp mode 0_B No timestamp, only issue result values 1_B Insert timestamp upon the trigger (when the gate opens)
TSCLK	18:17	rw	**Timestamp counter clock selection** 00_B $f_{TSTMP} = f_{MOD}$ 01_B $f_{TSTMP} = f_{MOD} / 2$ 10_B $f_{TSTMP} = f_{MOD} / 4$ 11_B $f_{TSTMP} = f_{MOD} / 8$
TSCRUN	19	rw	**Timestamp counter run control** 0_B Timestamp counter is off 1_B Timestamp counter is counting at the rate selected by bitfield TSCLK
SRLVL	21:20	rw	**Service request FIFO level** 00_B Generate service request when FIFO fill level reaches 1 value 01_B Generate service request when FIFO fill level reaches 2 values 10_B Generate service request when FIFO fill level reaches 3 values 11_B Generate service request when FIFO fill level reaches 4 values
FIFL	22	w	**FIFO flush** 0_B No action 1_B Remove all entries from result FIFO
RDM	23	rw	**Result display mode** 0_B Signed mode Result values range from -2^{15} to $+2^{15}-1$ 1_B Unsigned mode Result values range from 0 to $+2^{16}-1$ (shifted by 2^{15})
MANCLK	30	rw	**Manchester decoder clock** 0_B 10MHz 1_B 20MHz
MANEN	31	rw	**Manchester decoding enable** This bit configures, if the incoming data needs to be manchester decoded or not
0	3, 7:6, 29:24	r	**Reserved** Read as 0; should be written with 0

45.8.124 EXMODz channel status register

The channel status register gives the following information:

- Result FIFO status and error status bits
- CIC decimation count value
- CIC (auxiliary) decimation count value
- Trigger error flag status

DSADC_EXz_CHST (z=0-7) Offset address: $63804_H+z*400_H$

EXMODz channel status register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		BNDTRER	SRTRER	0	TSTRER	INTTRER	WRERR	RDERR	CFADCNT						
r		rh	rh	r	rh	rh	rh	rh	rh						

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CFMDCNT										0			FILL		
rh										r			rh		

Field	Bits	Type	Description
FILL	2:0	rh	**FIFO fill level** Not listed combinations are reserved. 000_B Result FIFO is empty 001_B Result FIFO contains 1 valid value 010_B Result FIFO contains 2 valid values 011_B Result FIFO contains 3 valid values 100_B Result FIFO contains 4 valid values
CFMDCNT	15:6	rh	**CIC filter decimation counter** The decimation counter counts the filter cycles until an output is generated, i.e. the oversampling rate. CFMDCNT counts down from the respective start value.
CFADCNT	22:16	rh	**CIC filter (auxiliary) decimation counter** The decimation counter counts the filter cycles until an output is generated, i.e. the oversampling rate.
RDERR	23	rh	**Read error flag** 0_B No problem encountered 1_B A read access occurred while the FIFO was empty A read error is also indicated when a read access occurs during the FIFO's synchronization stall phase (4 SPB clock cycles after a read access). Clear this sticky flag by writing 1 to bit RDEC.
WRERR	24	rh	**Write error flag** 0_B No problem encountered 1_B A write access occurred while the FIFO was full Clear this sticky flag by writing 1 to bit WREC.

(table continues...)

(continued)

Field	Bits	Type	Description
INTTRER	25	rh	**Integrator trigger error flag** This bit tells the status of the integrator trigger error flag. If the integrator trigger could not be executed the bit will be set to 1
TSTRER	26	rh	**Timestamp trigger error flag** Status of the timestamp trigger error flag. If the timestamp trigger could not be executed the bit will be set to 1
SRTRER	28	rh	**Service request trigger error flag** Status of the service request trigger error flag. If the service request trigger could not be executed the bit will be set to 1
BNDTRER	29	rh	**Boundary update trigger error flag** Status of the boundary update trigger error flag. If the boundary update trigger could not be executed the bit will be set to 1
0	5:3, 27, 31:30	r	**Reserved** Read as 0; should be written with 0

45.8.125 EXMODz CIC configuration register

DSADC_EXz_CICCFG (z=0-7) Offset address: $63808_H+z*400_H$

EXMODz CIC configuration register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0													CFADF		CFAEN
r													rw		rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0						CFMDF									
r						rw									

Field	Bits	Type	Description
CFMDF	9:0	rw	**CIC filter decimation factor** Defines the oversampling rate of the CIC filter: OSR = CFMDF + 1. Valid values are 003_H to $2FF_H$ (OSR = 4 to 1024).
CFAEN	16	rw	**CIC filter (auxiliary) enable** 0_B Off: auxiliary filter is not active 1_B Auxiliary filter is active and generates results and alarm events
CFADF	18:17	rw	**CIC filter (auxiliary) decimation factor** 00_B OSR = 16 01_B OSR = 32 10_B OSR = 64 11_B OSR = 128

(table continues...)

(continued)

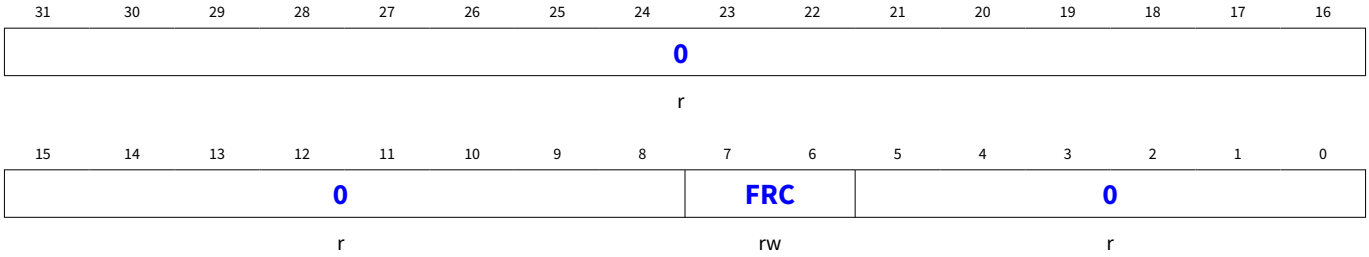
Field	Bits	Type	Description
0	15:10, 31:19	r	**Reserved** Read as 0; should be written with 0

45.8.126 EXMODz integrator configuration register

Configures the hardware integrator and the mode of operation of the integrator.
Integrator configuration change shall be avoided when INTEN = 1, if changed, may result in invalid outputs.

DSADC_EXz_INTCFG (z=0-7) Offset address: $6380C_H+z*400_H$

EXMODz integrator configuration register Kernel Reset value: $0000\ 0000_H$

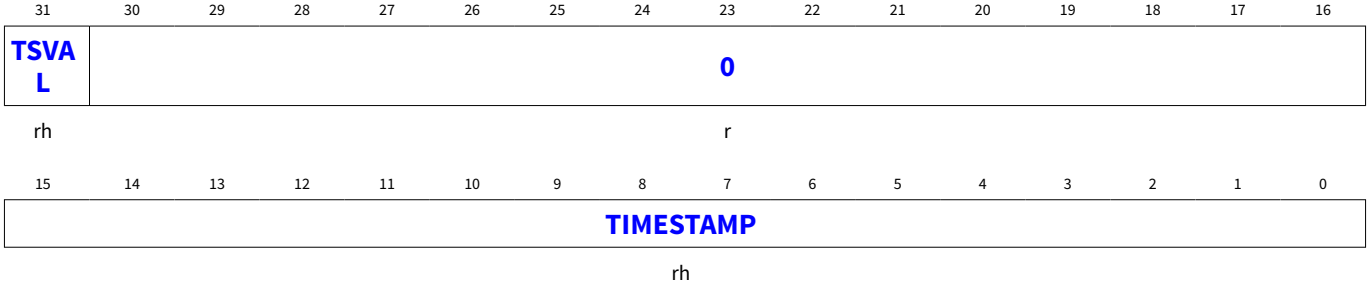


Field	Bits	Type	Description
FRC	7:6	rw	**Filter chain restart control** 00_B Restart the main filter chain when an integration window starts, auxiliary filter chain is not restarted 01_B Restart the main and auxilary filter chain when an integration window starts 10_B No influence on main and auxiliary filter chain when an integration window starts, except for the integrator itself 11_B No influence on main filter chain when an integration window starts, except for the integrator itself and auxiliary filter chain
0	5:0, 31:8	r	**Reserved** Read as 0; should be written with 0

45.8.127 EXMODz timestamp register

DSADC_EXz_TSTMP (z=0-7) Offset address: $63810_H+z*400_H$

EXMODz timestamp register Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
TIMESTAMP	15:0	rh	**The most recent captured timestamp value** This value is copied from the timestamp counter TSCOUNT
TSVAL	31	rh	**Timestamp valid** Indicates valid timestamp information. 0_B No timestamp trigger occurred since last read access 1_B Timestamp information has been stored after a timestamp trigger
0	30:16	r	**Reserved** Read as 0; should be written with 0

45.8.128 EXMODz timestamp counter

DSADC_EXz_TSCNT (z=0-7) Offset address: $63814_H+z*400_H$

EXMODz timestamp counter Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TSCOUNT															
rh															

Field	Bits	Type	Description
TSCOUNT	15:0	rh	**Timestamp counter value** TSCOUNT is clocked with the modulator clock and is cleared when a new result value has been generated.
0	31:16	r	**Reserved** Read as 0; should be written with 0

45.8.129 EXMODz result register main

DSADC_EXz_RESM (z=0-7) Offset address: $63818_H+z*400_H$

EXMODz result register main Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESULTHI															
rh															

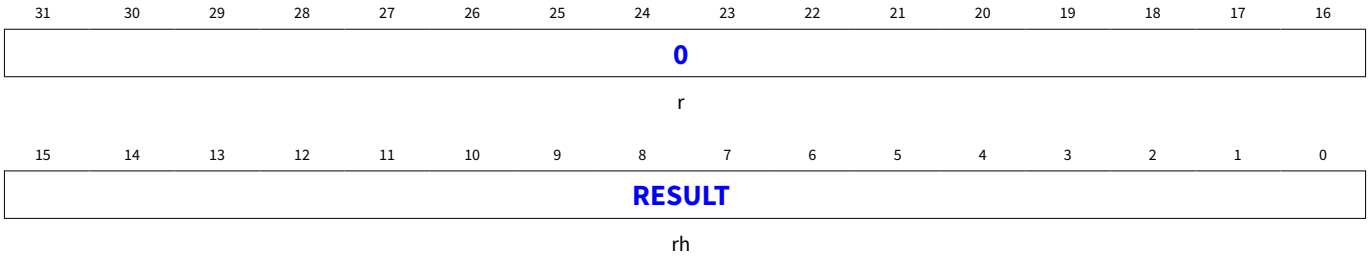
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESULTLO															
rh															

Field	Bits	Type	Description
RESULTLO	15:0	rh	**Result value lower part** Returns the next value from the result FIFO
RESULTHI	31:16	rh	**Result value higher part** Returns an additional value

45.8.130 EXMODz result register auxiliary

DSADC_EXz_RESA (z=0-7) Offset address: $6381C_H+z*400_H$

EXMODz result register auxiliary Kernel Reset value: $0000\ 0000_H$

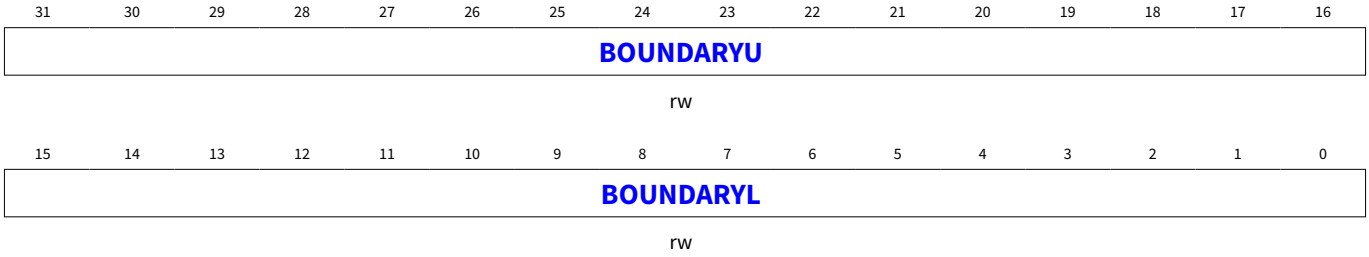


Field	Bits	Type	Description
RESULT	15:0	rh	**Most recent result of auxiliary filter**
0	31:16	r	**Reserved** Read as 0; should be written with 0

45.8.131 EXMODz boundary select register

DSADC_EXz_BOUNDSEL (z=0-7) Offset address: $63820_H+z*400_H$

EXMODz boundary select register Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
BOUNDARYL	15:0	rw	**Lower boundary value for limit checking** This (two's complement) value is compared to the upper bits of the CIC filter results.
BOUNDARYU	31:16	rw	**Upper boundary value for limit checking** This (two's complement) value is compared to the upper bits of the CIC filter results.

45.8.132 EXMODz integrator trigger configuration register

Configure the trigger source for doing trigger based integration.

DSADC_EXz_INTTRCFG (z=0-7) Offset address: $63824_H+z*400_H$

EXMODz integrator trigger configuration register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
INTTRDLY															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0						INTTRM		0		INTTRS					
r						rw		r		rw					

Field	Bits	Type	Description
INTTRS	5:0	rw	**Trigger source select for integrator** Refer "Hardware triggers to ADC" about the trigger source definition. 1)
INTTRM	9:8	rw	**Trigger mode select** 00_B No trigger edge selected, integrator bypassed 01_B Rising trigger edge selected 10_B Falling trigger edge selected 11_B No trigger, integrator active all the time
INTTRDLY	31:16	rw	**Trigger delay** Controls how much the selected trigger(s) shall be delayed before being executed The period of the configured edge(s) shall be greater than the configured delay value, else an error event will occur
0	7:6, 15:10	r	**Reserved** Read as 0; should be written with 0

1) To avoid unintended triggers, select the trigger source first before enabling the corresponding function.

45.8.133 EXMODz timestamp trigger configuration register

Configures the trigger source and trigger delay for the time stamp trigger

DSADC_EXz_TSTRCFG (z=0-7) Offset address: $63828_H+z*400_H$

EXMODz timestamp trigger configuration register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TSTRDLY															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0						TSTRM		0		TSTRS					
r						rw		r		rw					

Field	Bits	Type	Description
TSTRS	5:0	rw	**Trigger source select for timestamp** Refer "Hardware triggers to ADC" about the trigger source definition. 1)
TSTRM	9:8	rw	**Trigger mode select** 00_B No trigger edge selected 01_B Rising trigger edge selected 10_B Falling trigger edge selected 11_B Both trigger edges selected
TSTRDLY	31:16	rw	**Trigger delay** Controls how much the selected trigger shall be delayed before being executed The period of the configured edge(s) shall be greater than the configured delay value, else an error event will occur
0	7:6, 15:10	r	**Reserved** Read as 0; should be written with 0

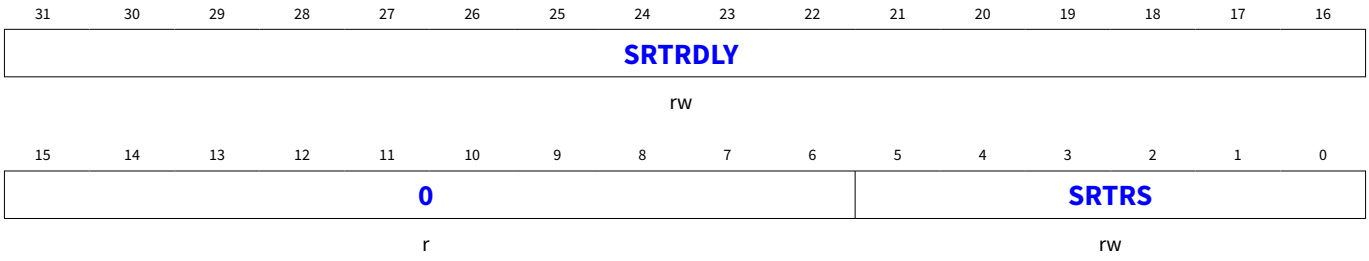
1) To avoid unintended triggers, select the trigger source first before enabling the corresponding function.

45.8.134 EXMODz service request trigger configuration register

Configures the trigger source and trigger delay for the service request trigger. Please also refer to DSADC_EXz_SRCFG.SR0E register for configuring the proper mode of trigger.

DSADC_EXz_SRTRCFG (z=0-7) Offset address: $6382C_H+z*400_H$

EXMODz service request trigger configuration register Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
SRTRS	5:0	rw	**Trigger source select for service request** Refer "Hardware triggers to ADC" about the trigger source definition. 1)
SRTRDLY	31:16	rw	**Trigger delay** Controls how much the selected trigger shall be delayed before being executed The period of the configured edge(s) shall be greater than the configured delay value, else an error event will occur
0	15:6	r	**Reserved** Read as 0; should be written with 0

1) To avoid unintended triggers, select the trigger source first before enabling the corresponding function.

45.8.135 EXMODz channel error clear register

DSADC_EXz_CHERRCL (z=0-7) Offset address: $63830_H+z*400_H$

EXMODz channel error clear register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

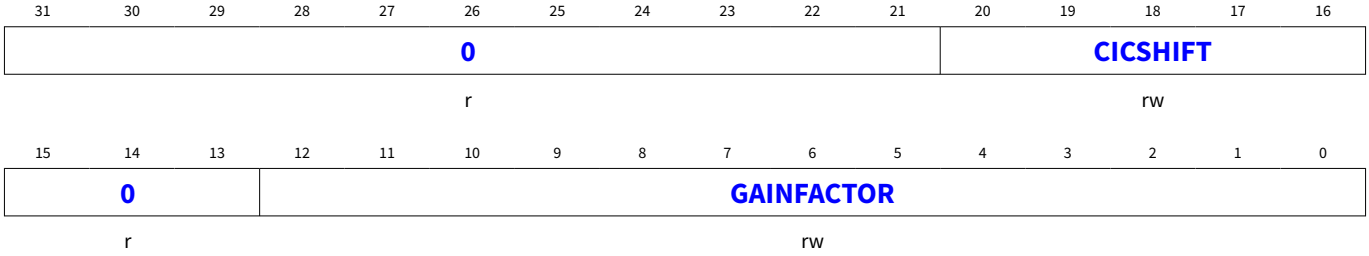
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0									BNDTRERC	INTTRERC	TSTRERC	0	SRTRERC	WREC	RDEC
r									w	w	w	r	w	w	w

Field	Bits	Type	Description
RDEC	0	w	**Read error flag clear** 0_B No action 1_B Clear flag RDERR
WREC	1	w	**Write error flag clear** 0_B No action 1_B Clear flag WRERR
SRTRERC	2	w	**Service request trigger error clear** 0_B No action 1_B Clear flag SRTRER
TSTRERC	4	w	**Time stamp trigger error clear** 0_B No action 1_B Clear flag TSTRER
INTTRERC	5	w	**Integrator trigger error clear** 0_B No action 1_B Clear flag INTTRER
BNDTRERC	6	w	**Boundary request trigger error clear** 0_B No action 1_B Clear flag BNDTRER
0	3, 31:7	r	**Reserved** Read as 0; should be written with 0

45.8.136 EXMODz gain correction register

DSADC_EXz_GAINCORR (z=0-7) Offset address: $63834_H+z*400_H$

EXMODz gain correction register Kernel Reset value: $0000\ 1000_H$

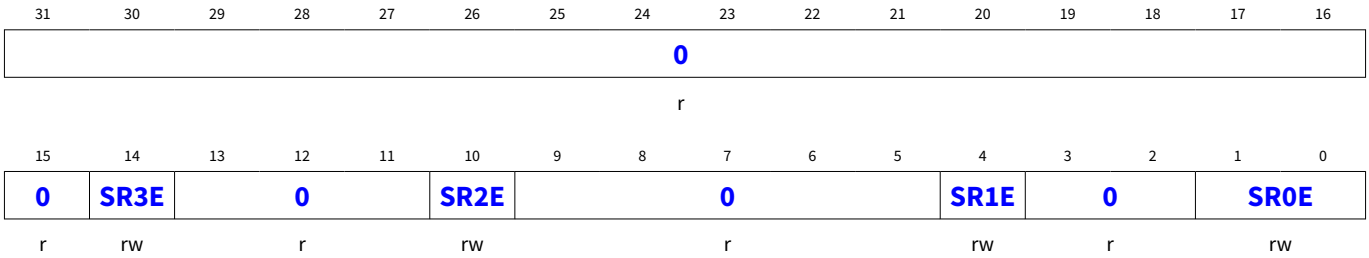


Field	Bits	Type	Description
GAINFACTOR	12:0	rw	**Multiplication factor for gain correction** The resulting factor is (<GAINFACTOR> / 4 096)
CICSHIFT	20:16	rw	**Position of the CIC filter output shifter** Selects the output bits from the CIC filter, depending on the chosen decimation factor (see formula above) $1D_H$ … $1F_H$ are reserved. 00_H Use bits 0 ... 16 … $1C_H$ Use bits 28 ... 44
0	15:13, 31:21	r	**Reserved** Read as 0; should be written with 0

45.8.137 EXMODz service request configuration register

DSADC_EXz_SRCFG (z=0-7) Offset address: $63838_H+z*400_H$

EXMODz service request configuration register Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
SR0E	1:0	rw	**Service request 0 enable and configuration** 00_B Never 01_B Generate service request only while gate is high 10_B Generate service request only while gate is low 11_B Always
SR1E	4	rw	**Service request 1 enable** Enable and disable for the service request 1. SR1 is generated at every auxiliary filter chain result

(table continues...)

(continued)

Field	Bits	Type	Description
SR2E	10	rw	**Service request 2 enable** Enable and disable for service request 2. SR2 is generated based on the configured boundary check through SR02CFG in BNDCFG register
SR3E	14	rw	**Service request 3 enable** Enable and disable for the service request 3. SR3 is generated at every time stamp generation if enabled.
0	3:2, 9:5, 13:11, 31:15	r	**Reserved** Read as 0; should be written with 0

45.8.138 EXMODz boundary mode configuration register

DSADC_EXz_BNDCFG (z=0-7) Offset address: $6383C_H+z*400_H$

EXMODz boundary mode configuration register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												SR2CFG	HYSTEN	BNDMODE	
r												rw	rw	rw	

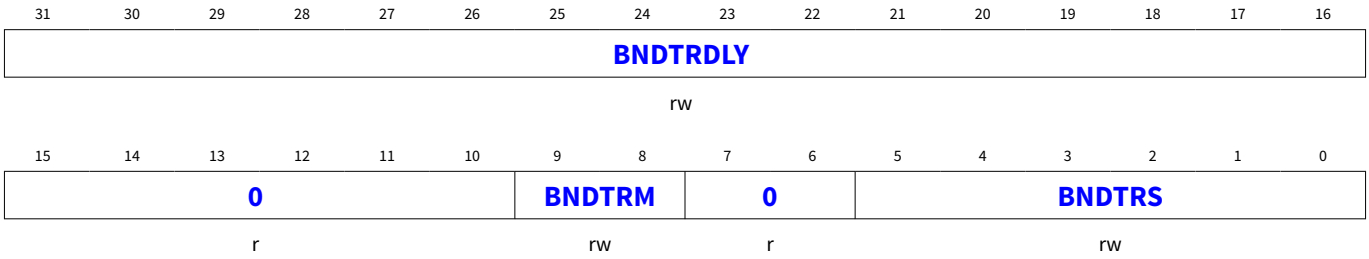
Field	Bits	Type	Description
BNDMODE	1:0	rw	**Boundary flag mode - BNDMOD** Defines the conditions in which boundary flag shall be raised 00_B Disabled 01_B Boudary flag is high when result exceeds upper limit 10_B Boudary flag is high when result is lower than lower limit 11_B Boudary flag is high when result is out of the band
HYSTEN	2	rw	**Hysteresis enable** Enables the hysteresis mode. In this mode BNDMODE configuration is ignored
SR2CFG	3	rw	**Conditions under which SR2 is generated** 0_B If the result is inside boundary band 1_B If the result is outisde the boudary band
0	31:4	r	**Reserved** Read as 0; should be written with 0

45.8.139 EXMODz boundary request trigger configuration register

Configures the trigger source and trigger delay for the update of the boundary values in a channel.

DSADC_EXz_BNDTRCFG (z=0-7) Offset address: $63840_H+z*400_H$

EXMODz boundary request trigger configuration register Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
BNDTRS	5:0	rw	**Trigger source select for updating boundary values** Refer "Hardware triggers to ADC" about the trigger source definition. [1]
BNDTRM	9:8	rw	**Trigger mode select** 00_B No trigger selected, boundary update immediately 01_B Boundary update on the rising edge of the trigger 10_B Boundary update on the falling edge of the trigger 11_B Boundary update at both rising and falling edges
BNDTRDLY	31:16	rw	**Trigger delay** Controls how much the selected trigger shall be delayed before being executed The period of the configured edge(s) shall be longer than the configured delay value, else an error event will occur
0	7:6, 15:10	r	**Reserved** Read as 0; should be written with 0

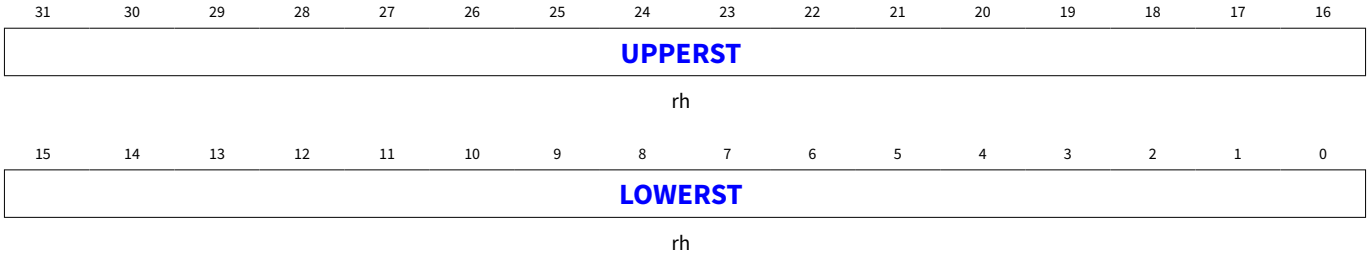
1) To avoid unintended triggers, select the trigger source first before enabling the corresponding function.

45.8.140 EXMODz boundary status register

Boundary status register gives the current used boundary values for limit checking

DSADC_EXz_BOUNDST (z=0-7) Offset address: $63844_H+z*400_H$

EXMODz boundary status register Kernel Reset value: $0000\ 0000_H$

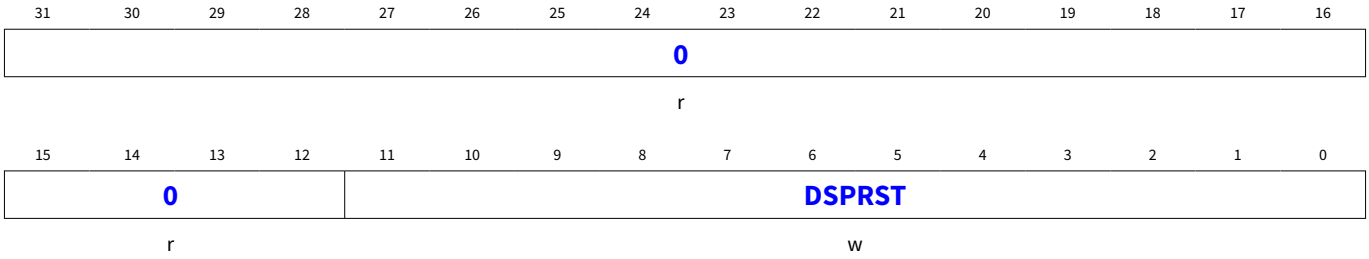


Field	Bits	Type	Description
LOWERST	15:0	rh	**Lower boundary value currently used for limit checking** The current used lower boundary value
UPPERST	31:16	rh	**Upper boundary value currently used for limit checking** The current used upper boundary value

45.8.141 CDSP global reset control

CDSP_GLOBRD Offset address: $BF804_H$

CDSP global reset control Kernel Reset value: $0000\ 0000_H$

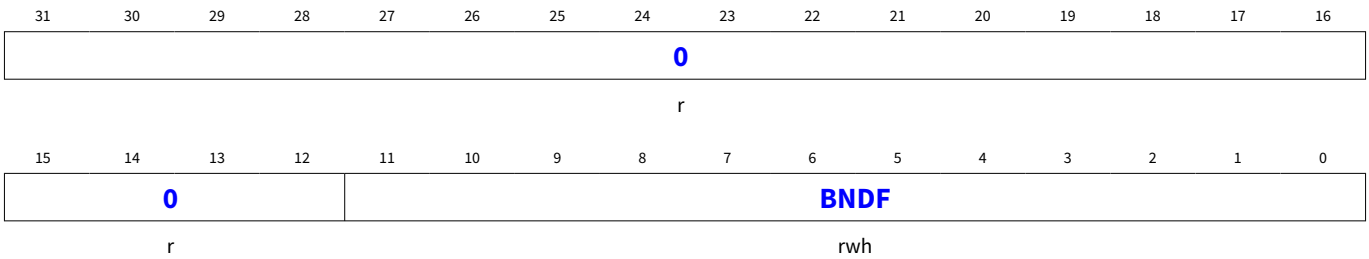


Field	Bits	Type	Description
DSPRST	11:0	w	**DSP reset** Each bit resets the corresponding CDSP instance. For example bit 0 of this register resets CDSP0.
0	31:12	r	**Reserved** Read as 0; should be written with 0

45.8.142 CDSP boudary flag event register

CDSP_BNDCF Offset address: $BF808_H$

CDSP boudary flag event register Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
BNDF	11:0	rwh	**Boundary event for CDSP channel** Status of the boundary flag event of the corresponding CDSP instance. For example bit 0 of this register indicates the status of boundary flag event of CDSP0

(table continues...)

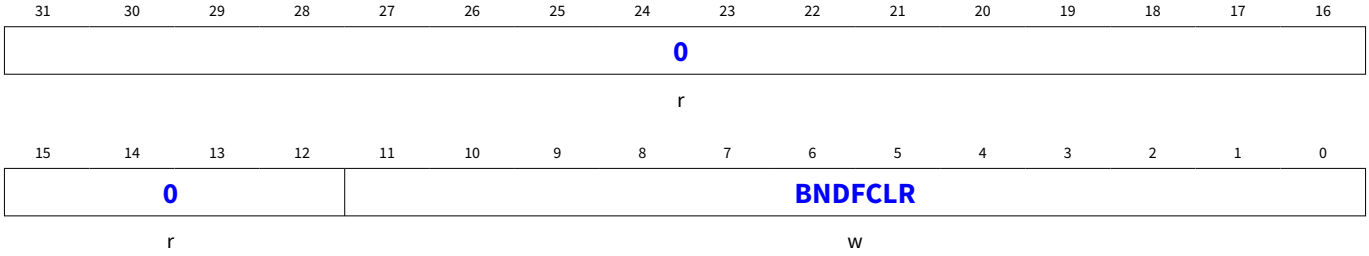
(continued)

Field	Bits	Type	Description
0	31:12	r	**Reserved** Read as 0; should be written with 0

45.8.143 CDSP boundary flag event clear

CDSP_BNDCFCL Offset address: $BF80C_H$

CDSP boundary flag event clear Kernel Reset value: $0000\ 0000_H$

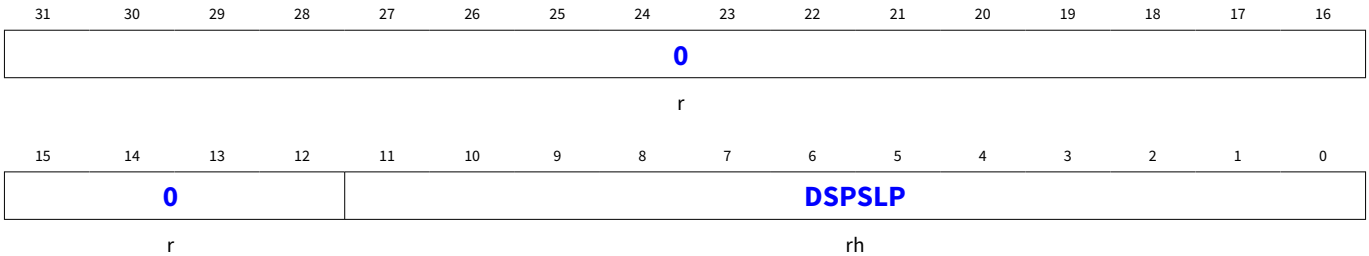


Field	Bits	Type	Description
BNDFCLR	11:0	w	**Boundary event clear for CDSP channels** Setting bits clears the bouandy event of the corresponding CDSP instance. For example bit 0 of this register clears the boundary event of the CDSP0.
0	31:12	r	**Reserved** Read as 0; should be written with 0

45.8.144 CDSP sleep status register

CDSP_SLPST Offset address: $BF810_H$

CDSP sleep status register Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
DSPSLP	11:0	rh	**CDSP sleep status** Sleep status of the DSP core of the corresponding CDSP instance. For example bit 0 of this register indicates the sleep status of the DSP core of the CDSP0.
0	31:12	r	**Reserved** Read as 0; should be written with 0

45.8.145 CDSP event flag register y

CDSP_RESEVy (y=0-2) Offset address: $BF814_H$+y*4

CDSP event flag register y Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				RESEV											
r				rwh											

Field	Bits	Type	Description
RESEV	11:0	rwh	**Result event for result register y of CDSP** Status of the result event of the corresponding CDSP instance. For example bit 0 of this register indicates an result event of CDSP0. Reading a result register (CDSP_DSPa_RESy) automatically clears the respective result event in CDSP_RESEVy.
0	31:12	r	**Reserved** Read as 0; should be written with 0

45.8.146 CDSP result event flag clear y

CDSP_RESEVCLRy (y=0-2) Offset address: $BF820_H$+y*4

CDSP result event flag clear y Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

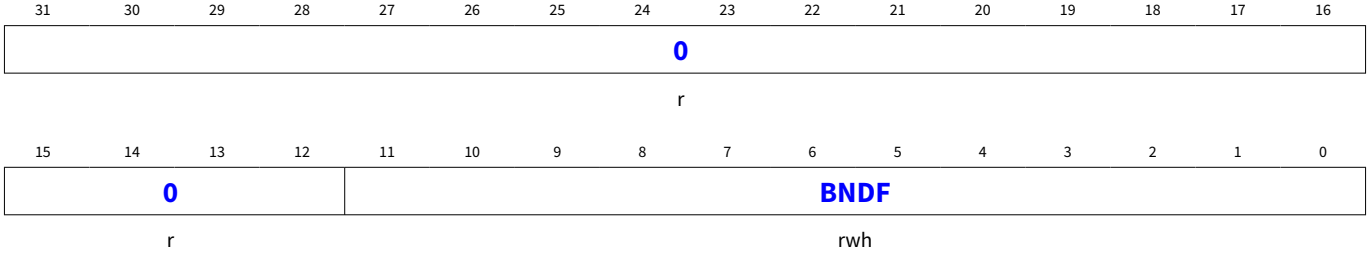
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				RESEVCLR											
r				w											

Field	Bits	Type	Description
RESEVCLR	11:0	w	**Result event clear for result register y of CDSP** Each bit clears the result event of the corresponding CDSP instance. For example setting bit 0 of this register clears the result event of the CDSP0.
0	31:12	r	**Reserved** Read as 0; should be written with 0

45.8.147 CDSP boudary flag event register 1

CDSP_BNDCF1 Offset address: BF82C$_H$

CDSP boudary flag event register 1 Kernel Reset value: 0000 0000$_H$

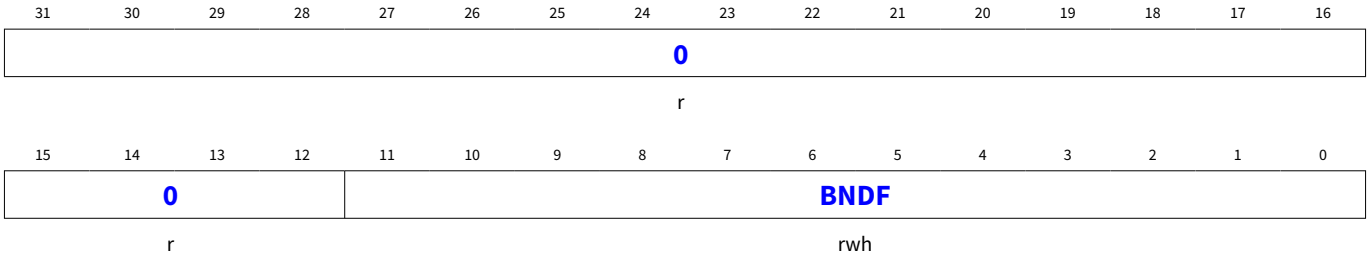


Field	Bits	Type	Description
BNDF	11:0	rwh	**Boundary event for CDSP channel** Status of the boundary flag event of the corresponding CDSP instance. For example bit 0 of this register indicates the status of boundary flag event of CDSP0
0	31:12	r	**Reserved** Read as 0; should be written with 0

45.8.148 CDSP boudary flag event register 2

CDSP_BNDCF2 Offset address: BF830$_H$

CDSP boudary flag event register 2 Kernel Reset value: 0000 0000$_H$

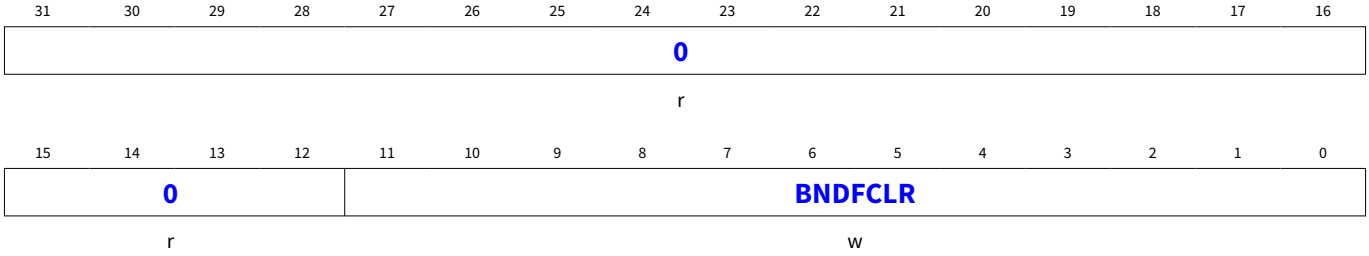


Field	Bits	Type	Description
BNDF	11:0	rwh	**Boundary event for CDSP channel** Status of the boundary flag event of the corresponding CDSP instance. For example bit 0 of this register indicates the status of boundary flag event of CDSP0
0	31:12	r	**Reserved** Read as 0; should be written with 0

45.8.149 CDSP boundary flag event clear 1

CDSP_BNDCFCL1 Offset address: BF834$_H$

CDSP boundary flag event clear 1 Kernel Reset value: 0000 0000$_H$

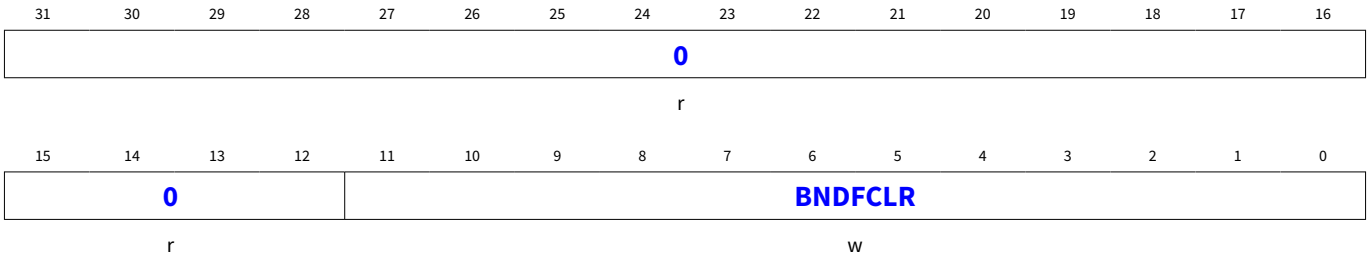


Field	Bits	Type	Description
BNDFCLR	11:0	w	**Boundary event clear for CDSP channels** Setting bits clears the bouandy event of the corresponding CDSP instance. For example bit 0 of this register clears the boundary event of the CDSP0.
0	31:12	r	**Reserved** Read as 0; should be written with 0

45.8.150 CDSP boundary flag event clear 2

CDSP_BNDCFCL2 Offset address: $BF838_H$

CDSP boundary flag event clear 2 Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
BNDFCLR	11:0	w	**Boundary event clear for CDSP channels** Setting bits clears the bouandy event of the corresponding CDSP instance. For example bit 0 of this register clears the boundary event of the CDSP0.
0	31:12	r	**Reserved** Read as 0; should be written with 0

45.8.151 CDSP input from general purpose register x

Input data for CDSP apart from ADCs

Note: Infineon software requires register layout of this register according to DSADC_INx_RESM.

CDSP_GPx (x=0-3) Offset address: $BF8C0_H+x*4$

CDSP input from general purpose register x Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GPI															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPI															
rw															

Field	Bits	Type	Description
GPI	31:0	rw	**General purpose input data mapped to DSP address space**

45.8.152 CDSP wake up error flag register

CDSP_WUERREV Offset address: $BF8D0_H$

CDSP wake up error flag register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				WUERREV											
r				rwh											

Field	Bits	Type	Description
WUERREV	11:0	rwh	**Wake up error event** Status of the wake up error event of the corresponding CDSP instance. For example bit 0 of this register indicates the status of the wake up error event of CDSP0.
0	31:12	r	**Reserved** Read as 0; should be written with 0

45.8.153 CDSP wake up error event clear register

CDSP_WUERRCL Offset address: $BF8D4_H$

CDSP wake up error event clear register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				WUERRCLR											
r				w											

Field	Bits	Type	Description
WUERRCLR	11:0	w	**Wake up error event clear** Each bit clears the wake up error event of the DSP core of the corresponding CDSP instance. For example bit 0 of this register clears the wake up error event of the DSP core of the CDSP0.
0	31:12	r	**Reserved** Read as 0; should be written with 0

45.8.154 CDSP global run control register

CDSP_GLOBRCD Offset address: $BF8D8_H$

CDSP global run control register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				CHRUND											
r				w											

Field	Bits	Type	Description
CHRUND	11:0	w	**Channel run for CDSP** Each bit of this register runs the DSP core of corresponding CDSP and logic around it. By writing '1' the CDSP switches into Run state, i.e program execution starts.
0	31:12	r	**Reserved** Read as 0; should be written with 0

45.8.155 CDSP global halt control register

CDSP_GLOBHCD Offset address: $BF8DC_H$

CDSP global halt control register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				CHHLTD											
r				w											

Field	Bits	Type	Description
CHHLTD	11:0	w	**Channel halt for DSP** Each bit of this register halts the DSP core in corresponding CDSP. Read the halt status register to check if DSP has been halted. This register is used, if DSP is halted from outside. For example bit 0 of this register controls the halt of CDSP0.
0	31:12	r	**Reserved** Read as 0; should be written with 0

45.8.156 CDSP global run status register

CDSP_GLOBRSD Offset address: $BF8E0_H$

CDSP global run status register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				DSPRST											
r				rh											

Field	Bits	Type	Description
DSPRST	11:0	rh	**DSP run status** Run status of the DSP core of corresponding CDSP and logic around it. For example bit 0 of this register indicates the Run status of CDSP0. Bit is kept high, until the core goes into Halt status.
0	31:12	r	**Reserved** Read as 0; should be written with 0

45.8.157 CDSP global halt status register

CDSP_GLOBHSTD Offset address: $BF8E4_H$

CDSP global halt status register Kernel Reset value: $0003\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				DSPHST											
r				rh											

Field	Bits	Type	Description
DSPHST	11:0	rh	**DSP halt status** Halt status of the DSP core of corresponding CDSP and logic around it. For example bit 0 of this register indicates the Halt status of CDSP0. Bit is kept high, until the core goes into Run status.
0	31:12	r	**Reserved** Read as 0; should be written with 0

45.8.158 OCDS control and status register 0 for CDSP0-8

CDSP_OCSCDSPA Offset address: $BF8E8_H$

OCDS control and status register 0 for CDSP0-8 Debug Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0						SUSSTA									SUS_P
r						rh									w

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0					SUS									SUSM	
r					rw									rw	

Field	Bits	Type	Description
SUSM	1:0	rw	**OCDS suspend control mode** Selects the suspend mode for the module. *Note: For reaching soft suspend condition the selected input data source must send enough inputs to the CDSP for a result event to be generated.* 00_B Will not suspend 01_B Hard suspend Clocks will be disabled immediately 10_B Soft suspend mode Clock will be disabled after processing the current sample. 11_B Soft suspend mode Clock will be disabled after generating the next result event, visible in the corresponding bit of CDSP_RESEV0.
SUS	10:2	rw	**OCDS suspend control** Suspends the corresponding CDSP. 0 - Module not suspended, 1 - Module Suspended. Each bit control a CDSP, with LSB for CDSP0 and MSB for the CDSP8. *Note: Please check how many CDSP are there in device*
SUS_P	16	w	**SUS write protection** SUS and SUSM are only written when SUS_P is 1, otherwise unchanged. Read as 0.

(table continues...)

(continued)

Field	Bits	Type	Description
SUSSTA	25:17	rh	**Suspend state** Suspend state of CDSP. 0 - Module not suspended. 1 - Module suspended. Each bit controls a CDSP, with LSB for CDSP0 and MSB for the CDSP8. *Note: Please check how many CDSP are there in device*
0	15:11, 31:26	r	**Reserved** Read as 0; should be written with 0.

Table 2405 Access mode restrictions of CDSP_OCSCDSPA sorted by descending priority

Mode name	Access mode		Description
write 1 to .**SUS_P**	rw	SUS, SUSM	Set SUS_P during write access
(default)	r	SUS, SUSM	

45.8.159 OCDS control and status register 1 for CDSP9-17

CDSP_OCSCDSPB Offset address: $BF8EC_H$

OCDS control and status register 1 for CDSP9-17 Debug Reset value: $0000\ 0000_H$

Field	Bits	Type	Description
SUSM	1:0	rw	**OCDS suspend control mode** Selects the suspend mode for the module. *Note: For reaching soft suspend condition the selected input data source must send enough inputs to the CDSP for a result event to be generated.* 00_B Will not suspend 01_B Hard suspend Clocks will be disabled immediately 10_B Soft suspend mode Clock will be disabled after processing the current sample. 11_B Soft suspend mode Clock will be disabled after generating the next result event, visible in the corresponding bit of CDSP_RESEV0.
SUS	4:2	rw	**OCDS suspend control** Suspends the corresponding CDSP. 0 - Module not suspended, 1 - Module suspended. Each bit control a CDSP, with LSB for CDSP9 and MSB for the CDSP17. *Note: Please check how many CDSP are there in device*

(table continues...)

(continued)

Field	Bits	Type	Description
SUS_P	16	w	**SUS write protection** SUS and SUSM are only written when SUS_P is 1, otherwise unchanged. Read as 0.
SUSSTA	19:17	rh	**Suspend state** Suspend state of CDSP. 0 - Module not suspended. 1 - Module suspended. Each bit control a CDSP, with LSB for CDSP9 and MSB for the CDSP17. *Note: Please check how many CDSP are there in device*
0	15:5, 31:20	r	**Reserved** Read as 0; should be written with 0.

Table 2406 Access mode restrictions of CDSP_OCSCDSPB sorted by descending priority

Mode name	Access mode		Description
write 1 to .**SUS_P**	rw	SUS, SUSM	Set SUS_P during write access
(default)	r	SUS, SUSM	

45.8.160 CDSPd debug status register

The DB_STATUS register provides information on all DEBUG operations. Writing to this register initiates the debug operation defined in DB_CMD. This register is volatile.

CDSP_DBd_STATUS (d=0-11) Offset address: $BFC00_H+d*20_H$

CDSPd debug status register Kernel Reset value: $0000\ 0024_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0										RA	RU	0	RD	FL	ST
r										rh	rh	r	rh	rh	rh

Field	Bits	Type	Description
ST	0	rh	**The stalled (ST) flag indicates that the debug interface is busy** 0_B Debug interface is not busy 1_B Debug interface is busy
FL	1	rh	**When true, the failure (FL) flag indicates that the debug operation has failed** 0_B Debug operation has not failed 1_B Debug operation has failed

(table continues...)

(continued)

Field	Bits	Type	Description
RD	2	rh	**Ready flag** The ready (RD) flag indicates whether the debug interface is available to accept another transaction command 0_B Debug interface is not available to accept another transaction command 1_B Debug interface is available to accept another transaction command
RU	4	rh	**The run (RU) flag is set to one when the ARC HS processor is running** 0_B Processor is not running 1_B Processor is running
RA	5	rh	**Reset applied** The Reset Applied (RA) flag is used by the debug host to determine that a target reset has occurred 0_B Target reset has not occurred 1_B Target reset has occurred
0	3, 31:6	r	**Reserved** Reserved. read-as-zero; ignored on write

45.8.161 CDSPd debug command register

The DB_CMD register defines the debug command (read and write to auxiliary register, core and memory).

CDSP_DBd_CMD (d=0-11) Offset address: $BFC04_H+d*20_H$

CDSPd debug command register Kernel Reset value: $0000\ 0003_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												COMMAND			
r												rw			

Field	Bits	Type	Description
COMMAND	3:0	rw	**Defines the debug command (read/write to aux/core/mem)** All non-enumerated values are reserved. 0_H Write to memory location 1_H Write to core register 2_H Write to auxiliary register 3_H Initiate a core and system reset 4_H Read memory location 5_H Read core register 6_H Read auxiliary register **others,** Reserved

(table continues...)

(continued)

Field	Bits	Type	Description
0	31:4	r	**Reserved** Reserved. read-as-zero; ignored on write

45.8.162 CDSPd debug address register

The DB_ADDR register specifies auxiliary register, core and memory address location. This register is volatile.

CDSP_DBd_ADDR (d=0-11) Offset address: $BFC08_H+d*20_H$

CDSPd debug address register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADDRESS															
rwh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRESS															
rwh															

Field	Bits	Type	Description
ADDRESS	31:0	rwh	**Debug address location** Auxiliary register, core or memory address location to be read from for debug read operations or written to for debug write operations

45.8.163 CDSPd debug data register

DB_DATA specifies the data to be written for debug write operations. It also holds the returned data for debug read operations. This register is volatile.

CDSP_DBd_DATA (d=0-11) Offset address: $BFC0C_H+d*20_H$

CDSPd debug data register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DATA_REGISTER															
rwh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA_REGISTER															
rwh															

Field	Bits	Type	Description
DATA_REGISTER	31:0	rwh	**Data register** Data to be written for debug write operations or returned data for debug read operations.

45.8.164 CDSPd debug reset register

Writing to the DB_RESET register triggers reset of DSP internal debug registers. Deviation from DSP specification.

CDSP_DBd_RESET (d=0-11) Offset address: $BFC10_H+d*20_H$

CDSPd debug reset register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0														RESET	
r														w	

Field	Bits	Type	Description
RESET	1:0	w	**Resets DSP internal debug registers** 00_B Resets DSP internal debug registers on writing. Independent of RESET data (here 0). 01_B Resets DSP internal debug registers on writing. Independent of RESET data (here 1). 10_B Resets DSP internal debug registers on writing. Independent of RESET data (here 2). 11_B Resets DSP internal debug registers on writing. Independent of RESET data (here 3).
0	31:2	r	**Reserved** Reserved. read-as-zero; ignored on write

45.8.165 CDSPa configuration register

Configuers the different hardware parameters of CDSP

CDSP_DSPa_DSPCFG (a=0-11) Offset address: $C1C00_H+a*2000_H$

CDSPa configuration register Kernel Reset value: $A000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
INITSRC				0		DRM		RDM	FIFL	SRLVL		TSCRUN	TSCLK		TSM
rw				r		rw		rw	w	rw		rw	rw		rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HSIMON								INPSEL							
rw								rw							

Field	Bits	Type	Description
INPSEL	7:0	rw	**Input selection** These bits configue which input register (TMADC, DSADC, EXMOD, CARMAG or general purpose register) triggers the wakeup signal of the CDSP.
HSIMON	15:8	rw	**Input register monitoring** These bits select the monitoring of the input result coming from DSADC (RESM, RESA), TMADC (RES0, RES1), EXMOD (RESM, RESA) or CARMAG
TSM	16	rw	**Time-stamp mode** 0_B No timestamp, only issue result values 1_B Insert timestamp upon the trigger (when the gate opens)
TSCLK	18:17	rw	**Timestamp counter clock selection** 00_B $f_{TS} = f_{ADC} / 4$ 01_B $f_{TS} = f_{ADC} / 8$ 10_B $f_{TS} = f_{ADC} / 16$ 11_B $f_{TS} = f_{ADC} / 32$
TSCRUN	19	rw	**Timestamp counter run control** 0_B Timestamp counter is off 1_B Timestamp counter is counting at the rate selected by bitfield TSCLK
SRLVL	21:20	rw	**Service request FIFO level** 00_B Generate service request when FIFO fill level reaches 1 value 01_B Generate service request when FIFO fill level reaches 2 values 10_B Generate service request when FIFO fill level reaches 3 values 11_B Generate service request when FIFO fill level reaches 4 values
FIFL	22	w	**FIFO flush** 0_B No action 1_B Remove all entries from result FIFO
RDM	23	rw	**Result display mode** 0_B Signed mode Result values range from -2^{15} to $+2^{15}-1$ 1_B Unsigned mode Result values range from 0 to $+2^{16}-1$ (shifted by 2^{15})
DRM	25:24	rw	**Data read mode** Selects the data that is returned when register RESMx is read 00_B Single: issue one 16-bit value per read access (sign on high bits) 01_B Single: issue one 16-bit value per read access (timestamp or zero on high bits) 10_B Double: issue two 16-bit values per read access 11_B Reserved

(table continues...)

(continued)

Field	Bits	Type	Description
INITSRC	31:28	rw	**Re-initialization event source** 0_H DSADC0 1_H DSADC1 2_H DSADC2 3_H DSADC3 4_H DSADC4 5_H DSADC5 6_H DSADC6 7_H DSADC7 8_H DSADC8 9_H DSADC9 A_H No re-initialization trigger
0	27:26	r	**Reserved** Read as 0; should be written with 0

45.8.166 CDSPa boundary trigger configuration register

Configures the trigger source, trigger mode and trigger delay for the update of the boundary values in CDSP.

CDSP_DSPa_BNDTRCFG (a=0-11) Offset address: $C1C04_H+a*2000_H$

CDSPa boundary trigger configuration register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BNDTRDLY															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0						BNDTRM		0		BNDTRS					
r						rw		r		rw					

Field	Bits	Type	Description
BNDTRS	5:0	rw	**Trigger selection select for boundary value update** Refer "Hardware triggers to ADC" about the trigger source definition. [1)]
BNDTRM	9:8	rw	**Trigger mode select** 00_B No trigger selected, Boundary update immediately 01_B Boundary update on the rising edge of the trigger 10_B Boundary update on the falling edge of the trigger 11_B Boundary update at both rising and falling edges
BNDTRDLY	31:16	rw	**Trigger delay** Controls how much the selected trigger shall be delayed before being executed The period of the configured edge(s) shall be longer than the configured delay value, else a trigger error event will occur
0	7:6, 15:10	r	**Reserved** Read as 0; should be written with 0

1) To avoid unintended triggers, select the trigger source first before enabling the corresponding function.

45.8.167 CDSPa boundary mode configuration register

CDSP_DSPa_BNDCFG (a=0-11) Offset address: $C1C08_H+a*2000_H$

CDSPa boundary mode configuration register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												SR2CFG	HYSTEN	BNDMODE	
r												rw	rw	rw	

Field	Bits	Type	Description
BNDMODE	1:0	rw	**Boundary flag mode** Defines the conditions in which boundary flag shall be raised 00_B Disabled 01_B Boudary flag is high when result exceeds upper limit 10_B Boudary flag is high when result is lower than lower limit 11_B Boudary flag is high when result is out of the band
HYSTEN	2	rw	**Hysteresis enable** Enables the hysteresis mode. In this mode BNDMODE configuration is ignored
SR2CFG	3	rw	**Conditions under which SR2 is generated** 0_B If the result is inside boundary band 1_B If the result is outisde the boudary band
0	31:4	r	**Reserved** Read as 0; should be written with 0

45.8.168 CDSPa boundary status register

Boundary status register gives the current used boundary values for limit checking.

CDSP_DSPa_BOUNDST (a=0-11) Offset address: $C1C0C_H+a*2000_H$

CDSPa boundary status register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UPPERST															
rh															

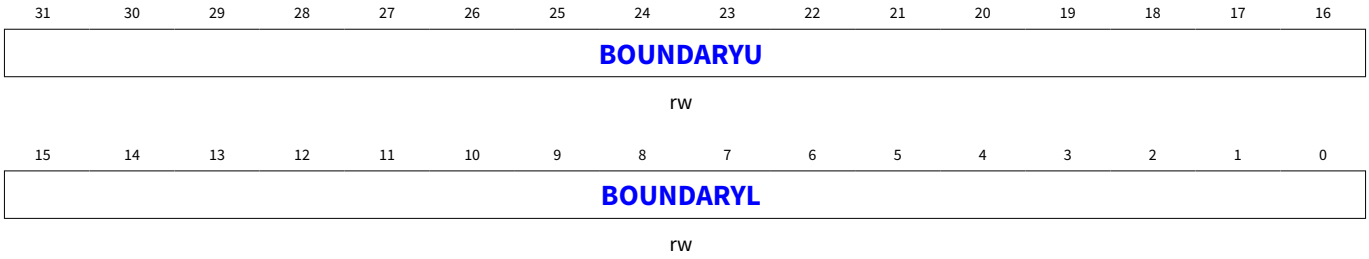
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LOWERST															
rh															

Field	Bits	Type	Description
LOWERST	15:0	rh	**Lower boundary value for limit checking** The current used lower boundary value
UPPERST	31:16	rh	**Upper boundary value for limit checking** The current used upper boundary value

45.8.169 CDSPa boundary select register

CDSP_DSPa_BOUNDSEL (a=0-11) Offset address: $C1C10_H+a*2000_H$

CDSPa boundary select register Kernel Reset value: $0000\ 0000_H$



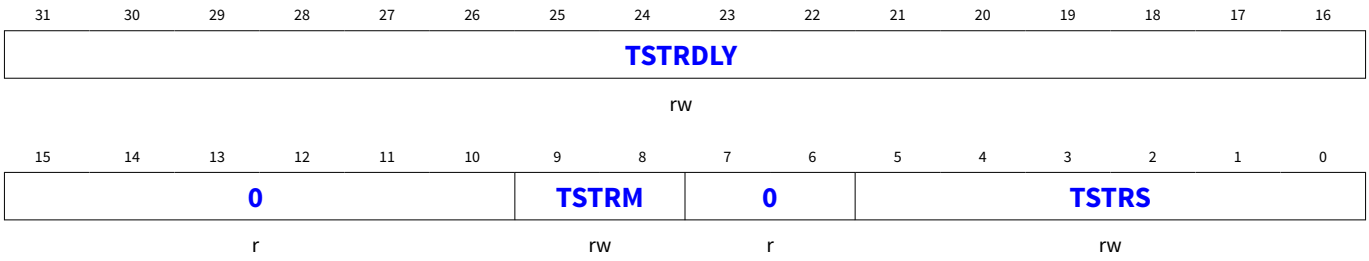
Field	Bits	Type	Description
BOUNDARYL	15:0	rw	**Lower boundary value for limit checking** Lower boundary value (two's complement) compared with lower 16 bits of CDSP result (CDSP_DSPa_RESy.CRESULTLO).
BOUNDARYU	31:16	rw	**Upper boundary value for limit checking** Upper boundary value (two's complement) compared with lower 16 bits of CDSP result (CDSP_DSPa_RESy.CRESULTLO).

45.8.170 CDSPa time stamp trigger configuration register

Configures the trigger source and trigger delay for the timestamp.

CDSP_DSPa_TSTRCFG (a=0-11) Offset address: $C1C14_H+a*2000_H$

CDSPa time stamp trigger configuration register Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
TSTRS	5:0	rw	**Trigger source select for timestamp** Refer "Hardware triggers to ADC" about the trigger source definition. [1)]

(table continues...)

(continued)

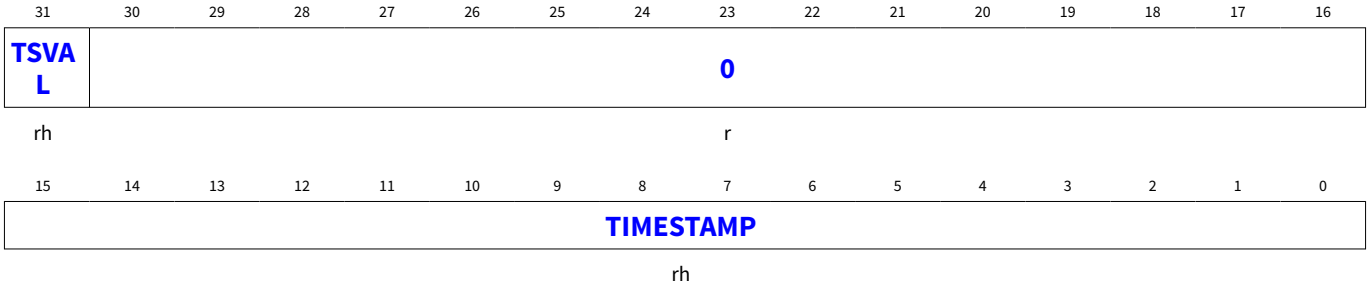
Field	Bits	Type	Description
TSTRM	9:8	rw	**Trigger mode select** 00_B No trigger edge selected 01_B Rising trigger edge selected 10_B Falling trigger edge selected 11_B Both trigger edges selected
TSTRDLY	31:16	rw	**Trigger delay** Controls how much the selected trigger shall be delayed before being executed The period of the configured edge(s) shall be longer than the configured delay value, else a trigger error event will occur
0	7:6, 15:10	r	**Reserved** Read as 0; should be written with 0

1) To avoid unintended triggers, select the trigger source first before enabling the corresponding function.

45.8.171 CDSPa time-stamp register

CDSP_DSPa_TSTMP (a=0-11) Offset address: $C1C18_H+a*2000_H$

CDSPa time-stamp register Kernel Reset value: $0000\ 0000_H$

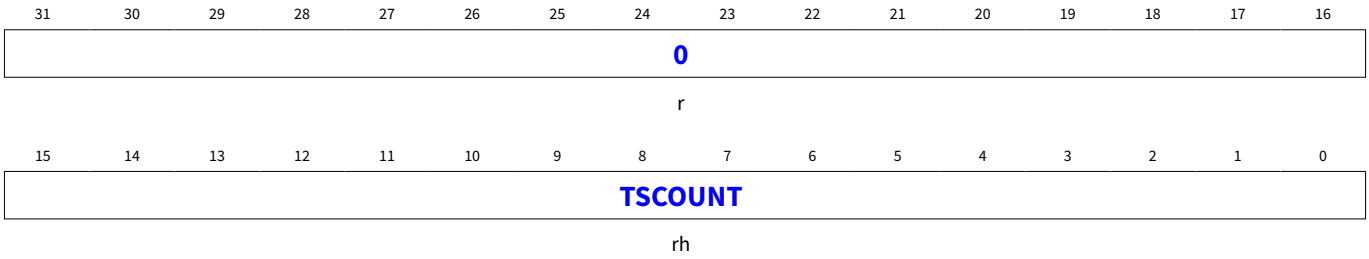


Field	Bits	Type	Description
TIMESTAMP	15:0	rh	**The most recent captured timestamp value** This value is copied from the timestamp counter TSCOUNT.
TSVAL	31	rh	**Timestamp valid** Indicates valid timestamp information. 0_B No timestamp trigger occurred since last read access 1_B Timestamp information has been stored after a timestamp trigger
0	30:16	r	**Reserved** Read as 0; should be written with 0

45.8.172 CDSPa time-stamp counter

CDSP_DSPa_TSCNT (a=0-11) Offset address: $C1C1C_H+a*2000_H$

CDSPa time-stamp counter Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
TSCOUNT	15:0	rh	**Timestamp counter value** TSCOUNT is counting on CDSP clocks and is cleared when a new result value has been generated.
0	31:16	r	**Reserved** Read as 0; should be written with 0

45.8.173 CDSPa service request trigger configuration register

Configures the trigger source and trigger delay for the service request trigger. Please also refer to CDSP_DSPa_SRCFG.SR0E register for configuring the proper mode of trigger.

CDSP_DSPa_SRTRCFG (a=0-11) Offset address: $C1C20_H+a*2000_H$

CDSPa service request trigger configuration register Kernel Reset value: $0000\ 0000_H$

Field	Bits	Type	Description
SRTRS	5:0	rw	**Trigger source select for service request** Refer "Hardware triggers to ADC" about the trigger source definition. [1)]
SRTRDLY	31:16	rw	**Trigger delay** Controls how much the selected trigger shall be delayed before being executed The period of the configured edge(s) shall be longer than the configured delay value, else an error event will occur
0	15:6	r	**Reserved** Read as 0; should be written with 0

1) To avoid unintended triggers, select the trigger source first before enabling the corresponding function.

45.8.174 CDSPa service request configuration register

CDSP_DSPa_SRCFG (a=0-11) Offset address: $C1C24_H+a*2000_H$

CDSPa service request configuration register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

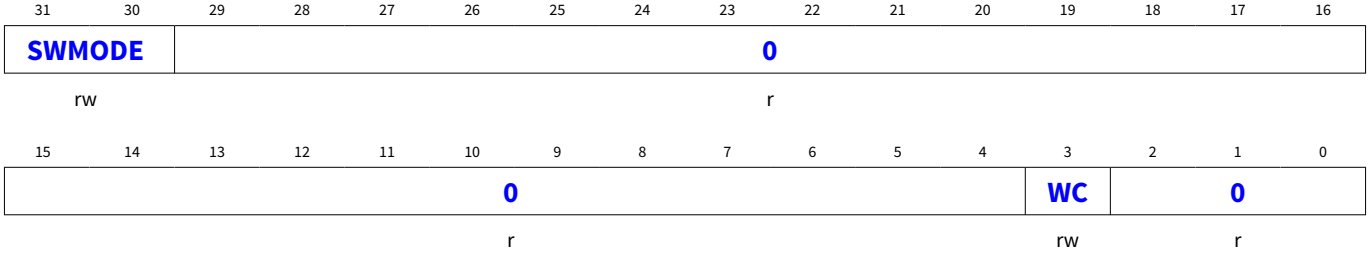
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			SR2E2	SR2E1	SR2E	0				SR1E		0		SR0E	
r			rw	rw	rw	r				rw		r		rw	

Field	Bits	Type	Description
SR0E	1:0	rw	**Conditions under which SR0 shall be generated** 00_B Never 01_B Generate Service Request only while gate is high 10_B Generate Service Request only while gate is low 11_B Always
SR1E	5:4	rw	**Service request 1 enable** Enable or disable for the service request 1, SR1 is generated at every time stamp generation if enabled 00_B SR1 disabled 01_B SR1 is generated everytime a new timstamp is generated 10_B SR1 is generated when a wakeup error occurs or ARC core is in halt state 11_B SR1 is generated on timestamp generation, wakeup error event or system halt
SR2E	10	rw	**Service request 2 enable** Enable or disable for service request 2, SR2 is generated based on the configured boundary check through SR02CFG in CDSP_DSPa_BNDCFG register
SR2E1	11	rw	**Service request 2 enable 1** Enable or disable for service request 2, SR2 is generated based on the configured boundary check through SR02CFG in CDSP_DSPa_BNDCFG register
SR2E2	12	rw	**Service request 2 enable 2** Enable or disable for service request 2, SR2 is generated based on the configured boundary check through SR02CFG in CDSP_DSPa_BNDCFG register
0	3:2, 9:6, 31:13	r	**Reserved** Read as 0; should be written with 0

45.8.175 CDSPa software configuration register

CDSP_DSPa_SWCFG (a=0-11) Offset address: $C1C28_H+a*2000_H$

CDSPa software configuration register Kernel Reset value: $0000\ 0000_H$

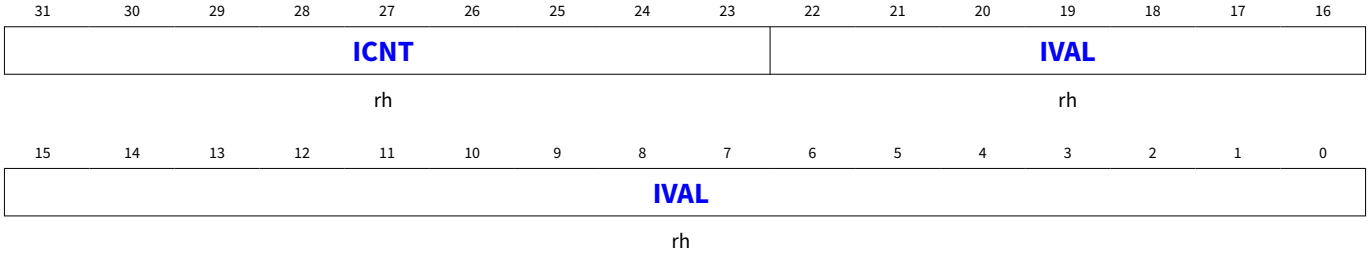


Field	Bits	Type	Description
WC	3	rw	**Window control** 0_B Internal control : trigger controlled software function will be stopped automatically according to the configuration *NOTE: the inverse activation trigger edge will be ignored* 1_B External control : trigger controlled software function will be stopped by the inverse activation trigger edge *NOTE: the FFT will stop automatically according to the configuration, if there was no inverse activation trigger edge during the FFT calculation*
SWMODE	31:30	rw	**Software mode** 00_B Software results transmitted to the result registers as generated, INTIVAL updates blocked, the running DSP is sensitive to wake-up events all the time *NOTE: to be used for filter chains without integrator* 01_B Software results transmitted to the result registers as generated upon write access to RES1, INTIVAL updates together with RES1 and RES2, the running DSP is sensitive to wake-up events all the time *NOTE: to be used for filter chains with integrator* 10_B Software results transmitted to result registers as generated, INTIVAL updates blocked, the running DSP is sensitive to wake-up events until write access to RES2 *NOTE: to be used for FFT, wake-up event sensitivity will be re-established with software function reactivation or when changing SWMODE* 11_B Reserved
0	2:0, 29:4	r	**Reserved** Read as 0; should be written with 0

45.8.176 CDSPa intermediate integration value

CDSP_DSPa_INTIVAL (a=0-11) Offset address: $C1C2C_H+a*2000_H$

CDSPa intermediate integration value Kernel Reset value: $0000\ 0000_H$

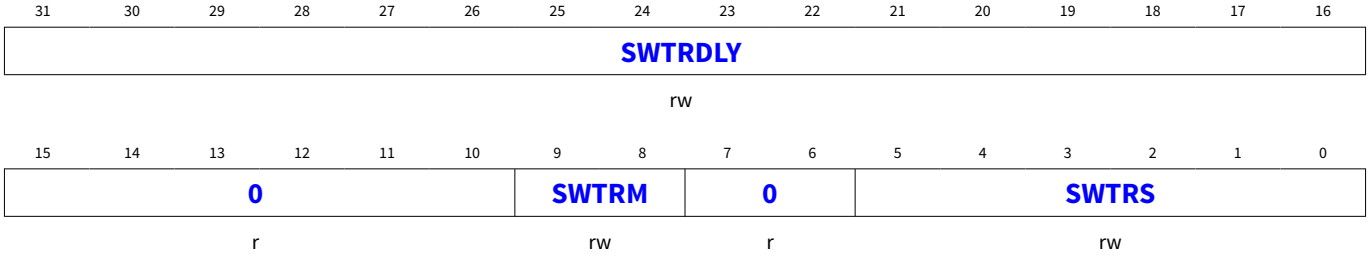


Field	Bits	Type	Description
IVAL	22:0	rh	**Integrator intermediate value**
ICNT	31:23	rh	**Integrator intermediate count**

45.8.177 CDSPa software trigger configuration register

CDSP_DSPa_SWTRCFG (a=0-11) Offset address: $C1C30_H+a*2000_H$

CDSPa software trigger configuration register Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
SWTRS	5:0	rw	**Software trigger source for the configured software function** Refer "Hardware triggers to ADC" about the trigger source definition.
SWTRM	9:8	rw	**Trigger mode select** 00_B No trigger edge selected, trigger controlled software function deactivated 01_B Rising trigger edge selected to (re)activate the trigger controlled software function 10_B Falling trigger edge selected to (re)activate the trigger controlled software function 11_B No trigger edge selected, trigger controlled software function activated *NOTE: IF SWTRM=11, the FFT will stop automatically according to the configuration (at the end of the FFT calculation) if it was not stopped before by setting SWTRM=00. The FFT can be restarted e.g. by setting SWTRM=00 and then SWTRM=11.*
SWTRDLY	31:16	rw	**Trigger delay** Controls how much the selected trigger shall be delayed before being executed. NOTE: The time between two equal trigger edges must be greater or equal the configured delay value, else an error event will be generated.

(table continues...)

(continued)

Field	Bits	Type	Description
0	7:6, 15:10	r	**Reserved** Read as 0; should be written with 0

45.8.178 CDSPa channel error clear register

CDSP_DSPa_CHERRCL (a=0-11) Offset address: $C1C34_H+a*2000_H$

CDSPa channel error clear register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

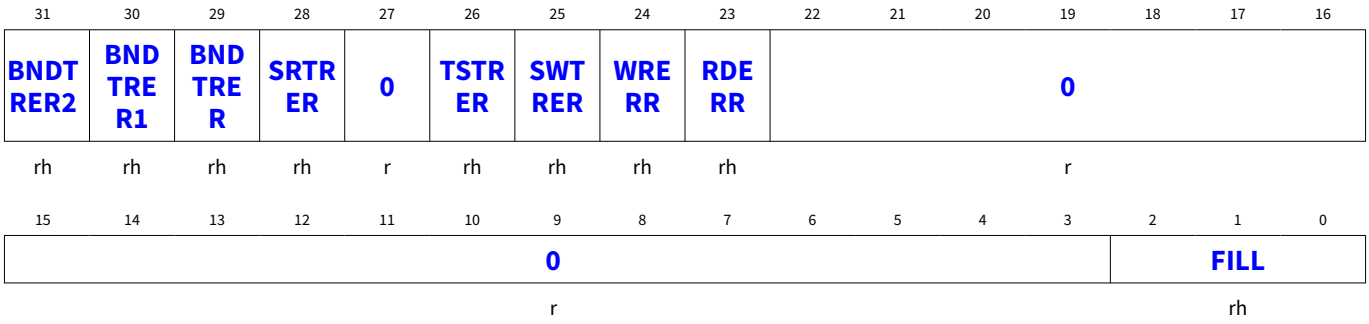
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0							BNDTRERC2	BNDTRERC1	BNDTRERC	SWTRERC	TSTRERC	0	SRTRERC	WREC	RDEC
r							w	w	w	w	w	r	w	w	w

Field	Bits	Type	Description
RDEC	0	w	**Read error flag clear** 0_B No action 1_B Clear flag RDERR
WREC	1	w	**Write error flag clear** 0_B No action 1_B Clear flag WRERR
SRTRERC	2	w	**Service request trigger error clear**
TSTRERC	4	w	**Time-stamp trigger error clear**
SWTRERC	5	w	**Software trigger error clear** 0_B No action 1_B Clear flag SWTRER
BNDTRERC	6	w	**Boundary request trigger error clear**
BNDTRERC1	7	w	**Boundary request trigger error clear 1**
BNDTRERC2	8	w	**Boundary request trigger error clear 2**
0	3, 31:9	r	**Reserved** Read as 0; should be written with 0

45.8.179 CDSPa status register

CDSP_DSPa_DSPST (a=0-11) Offset address: $C1C38_H+a*2000_H$

CDSPa status register Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
FILL	2:0	rh	**FIFO fill level** Not listed combinations are reserved. 000_B Result FIFO is empty 001_B Result FIFO contains 1 valid value 010_B Result FIFO contains 2 valid values 011_B Result FIFO contains 3 valid values 100_B Result FIFO contains 4 valid values
RDERR	23	rh	**Read error flag** 0_B No problem encountered 1_B A read access occurred while the FIFO was empty A read error is also indicated when a read access occurs during the FIFO's synchronization stall phase (4 fADC after a read access). Clear this sticky flag by writing 1 to bit RDEC.
WRERR	24	rh	**Write error flag** 0_B No problem encountered 1_B A write access occurred while the FIFO was full Clear this sticky flag by writing 1 to bit WREC.
SWTRER	25	rh	**Software trigger error flag** This bit tells the status of the software trigger error flag. If the software trigger could not be executed the bit will be set to 1
TSTRER	26	rh	**Timestamp trigger error flag** This bit tells the status of the timestamp trigger error flag. If the timestamp trigger could not be executed the bit will be set to 1
SRTRER	28	rh	**Service request trigger error flag** This bit tells the status of the service request trigger error flag. If the service request trigger could not be executed the bit will be set to 1
BNDTRER	29	rh	**Boundary update trigger error flag** This bit tells the status of the boundary update trigger error flag. If the boundary update trigger could not be executed the bit will be set to 1
BNDTRER1	30	rh	**Boundary update trigger error flag 1** This bit tells the status of the boundary update trigger error flag. If the boundary update trigger could not be executed the bit will be set to 1

(table continues...)

(continued)

Field	Bits	Type	Description
BNDTRER2	31	rh	**Boundary update trigger error flag 2** This bit tells the status of the boundary update trigger error flag. If the boundary update trigger could not be executed the bit will be set to 1
0	22:3, 27	r	**Reserved** Read as 0; should be written with 0

45.8.180 CDSPa input from DSADC, TMADC, EXMOD or CARMAG

Defines input information for CDSP from DSADC, TMADC (Core-0 and Core-1), EXMOD or CARMAG

CDSP_DSPa_INPUTMON (a=0-11) Offset address: $C1C3C_H+a*2000_H$

CDSPa input from DSADC, TMADC, EXMOD or CARMAG Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MONDATA															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MONDATA															
rh															

Field	Bits	Type	Description
MONDATA	31:0	rh	**Input data to CDSP** Input data to CDSP from DSADC, TMADC (Core-0 and Core-1), EXMOD, CARMAG or General Purpose (GP) input registers.

45.8.181 CDSPa result register y

CDSP_DSPa_RESy (a=0-11;y=0-2) Offset address: $C1C40_H+a*2000_H+y*4$

CDSPa result register y Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CRESULTHI															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CRESULTLO															
rh															

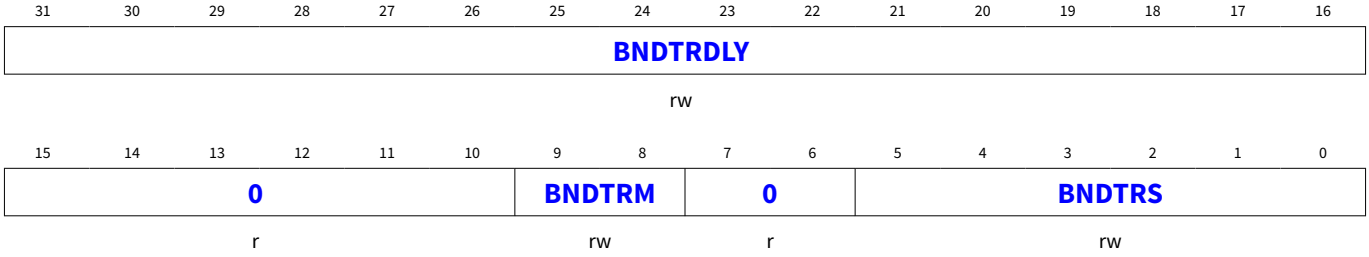
Field	Bits	Type	Description
CRESULTLO	15:0	rh	**CDSP result value lower part** Returns the next value from the result FIFO
CRESULTHI	31:16	rh	**CDSP result value higher part** Returns an additional value

45.8.182 CDSPa boundary trigger configuration register 1

Configures the trigger source, trigger mode and trigger delay for the update of the boundary values in CDSP.

CDSP_DSPa_BNDTRCFG1 (a=0-11) Offset address: $C1C4C_H+a*2000_H$

CDSPa boundary trigger configuration register 1 Kernel Reset value: $0000\ 0000_H$



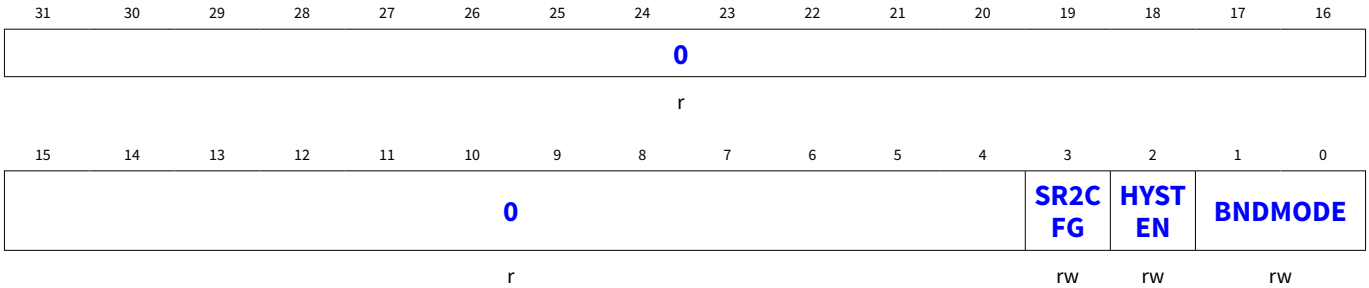
Field	Bits	Type	Description
BNDTRS	5:0	rw	**Trigger selection select for boundary value update** Refer "Hardware triggers to ADC" about the trigger source definition. 1)
BNDTRM	9:8	rw	**Trigger mode select** 00_B No trigger selected, Boundary update immediately 01_B Boundary update on the rising edge of the trigger 10_B Boundary update on the falling edge of the trigger 11_B Boundary update at both rising and falling edges
BNDTRDLY	31:16	rw	**Trigger delay** Controls how much the selected trigger shall be delayed before being executed The period of the configured edge(s) shall be longer than the configured delay value, else a trigger error event will occur
0	7:6, 15:10	r	**Reserved** Read as 0; should be written with 0

1) To avoid unintended triggers, select the trigger source first before enabling the corresponding function.

45.8.183 CDSPa boundary mode configuration register 1

CDSP_DSPa_BNDCFG1 (a=0-11) Offset address: $C1C50_H+a*2000_H$

CDSPa boundary mode configuration register 1 Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
BNDMODE	1:0	rw	**Boundary flag mode** Defines the conditions in which boundary flag shall be raised 00_B Disabled 01_B Boudary flag is high when result exceeds upper limit 10_B Boudary flag is high when result is lower than lower limit 11_B Boudary flag is high when result is out of the band
HYSTEN	2	rw	**Hysteresis enable** Enables the hysteresis mode. In this mode BNDMODE configuration is ignored
SR2CFG	3	rw	**Conditions under which SR2 is generated** 0_B If the result is inside boundary band 1_B If the result is outisde the boudary band
0	31:4	r	**Reserved** Read as 0; should be written with 0

45.8.184 CDSPa boundary status register 1

Boundary status register gives the current used boundary values for limit checking.

CDSP_DSPa_BOUNDST1 (a=0-11) Offset address: $C1C54_H+a*2000_H$

CDSPa boundary status register 1 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UPPERST															
rh															

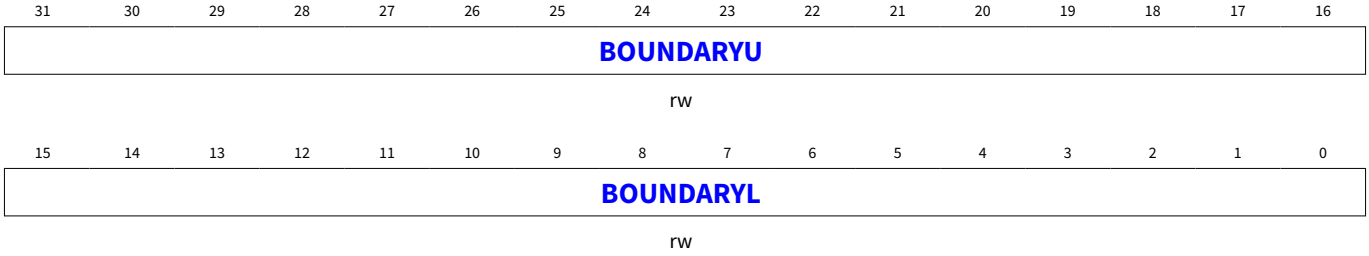
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LOWERST															
rh															

Field	Bits	Type	Description
LOWERST	15:0	rh	**Lower boundary value for limit checking** The current used lower boundary value
UPPERST	31:16	rh	**Upper boundary value for limit checking** The current used upper boundary value

45.8.185 CDSPa boundary select register 1

CDSP_DSPa_BOUNDSEL1 (a=0-11) Offset address: $C1C58_H+a*2000_H$

CDSPa boundary select register 1 Kernel Reset value: $0000\ 0000_H$



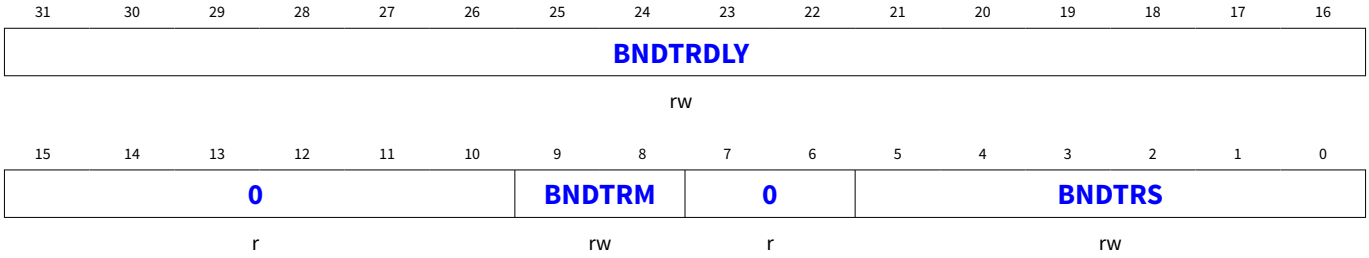
Field	Bits	Type	Description
BOUNDARYL	15:0	rw	**Lower boundary value for limit checking** Lower boundary value (two's complement) compared with lower 16 bits of CDSP result (CDSP_DSPa_RESy.CRESULTLO).
BOUNDARYU	31:16	rw	**Upper boundary value for limit checking** Upper boundary value (two's complement) compared with lower 16 bits of CDSP result (CDSP_DSPa_RESy.CRESULTLO).

45.8.186 CDSPa boundary trigger configuration register 2

Configures the trigger source, trigger mode and trigger delay for the update of the boundary values in CDSP.

CDSP_DSPa_BNDTRCFG2 (a=0-11) Offset address: $C1C5C_H+a*2000_H$

CDSPa boundary trigger configuration register 2 Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
BNDTRS	5:0	rw	**Trigger selection select for boundary value update** Refer "Hardware triggers to ADC" about the trigger source definition. [1)]
BNDTRM	9:8	rw	**Trigger mode select** 00_B No trigger selected, Boundary update immediately 01_B Boundary update on the rising edge of the trigger 10_B Boundary update on the falling edge of the trigger 11_B Boundary update at both rising and falling edges
BNDTRDLY	31:16	rw	**Trigger delay** Controls how much the selected trigger shall be delayed before being executed The period of the configured edge(s) shall be longer than the configured delay value, else a trigger error event will occur
0	7:6, 15:10	r	**Reserved** Read as 0; should be written with 0

1) To avoid unintended triggers, select the trigger source first before enabling the corresponding function.

45.8.187 CDSPa boundary mode configuration register 2

CDSP_DSPa_BNDCFG2 (a=0-11) Offset address: $C1C60_H+a*2000_H$

CDSPa boundary mode configuration register 2 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												SR2CFG	HYSTEN	BNDMODE	
r												rw	rw	rw	

Field	Bits	Type	Description
BNDMODE	1:0	rw	**Boundary flag mode** Defines the conditions in which boundary flag shall be raised 00_B Disabled 01_B Boudary flag is high when result exceeds upper limit 10_B Boudary flag is high when result is lower than lower limit 11_B Boudary flag is high when result is out of the band
HYSTEN	2	rw	**Hysteresis enable** Enables the hysteresis mode. In this mode BNDMODE configuration is ignored
SR2CFG	3	rw	**Conditions under which SR2 is generated** 0_B If the result is inside boundary band 1_B If the result is outisde the boudary band
0	31:4	r	**Reserved** Read as 0; should be written with 0

45.8.188 CDSPa boundary status register 2

Boundary status register gives the current used boundary values for limit checking.

CDSP_DSPa_BOUNDST2 (a=0-11) Offset address: $C1C64_H+a*2000_H$

CDSPa boundary status register 2 Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UPPERST															
rh															

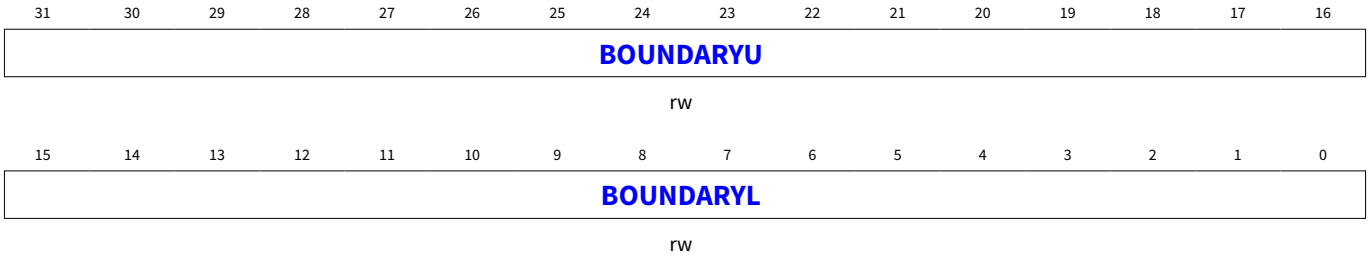
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LOWERST															
rh															

Field	Bits	Type	Description
LOWERST	15:0	rh	**Lower boundary value for limit checking** The current used lower boundary value
UPPERST	31:16	rh	**Upper boundary value for limit checking** The current used upper boundary value

45.8.189 CDSPa boundary select register 2

CDSP_DSPa_BOUNDSEL2 (a=0-11) Offset address: $C1C68_H+a*2000_H$

CDSPa boundary select register 2 Kernel Reset value: $0000\ 0000_H$

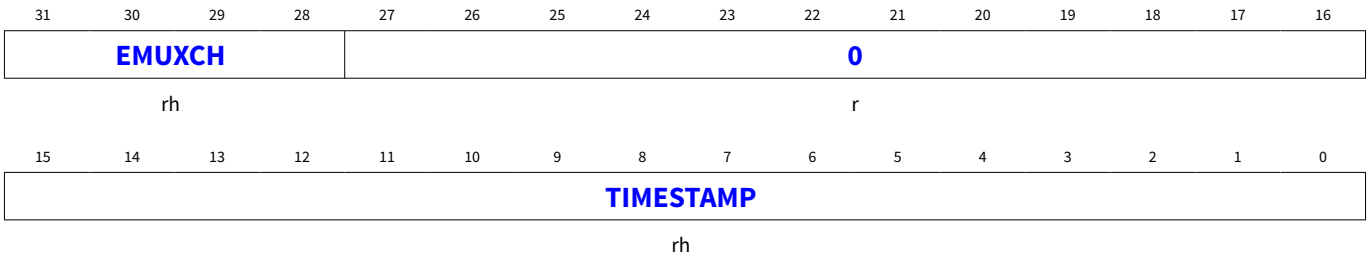


Field	Bits	Type	Description
BOUNDARYL	15:0	rw	**Lower boundary value for limit checking** Lower boundary value (two's complement) compared with lower 16 bits of CDSP result (CDSP_DSPa_RESy.CRESULTLO).
BOUNDARYU	31:16	rw	**Upper boundary value for limit checking** Upper boundary value (two's complement) compared with lower 16 bits of CDSP result (CDSP_DSPa_RESy.CRESULTLO).

45.8.190 TMADCx timestamp register v on LLI (AXI)

TMADCRESx_TSAXIv (v=0-15;x=0-6) Offset address: $20000_H+x*800_H+v*4$

TMADCx timestamp register v on LLI [AXI] Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
TIMESTAMP	15:0	rh	**The most recent captured timestamp value**

(table continues...)

(continued)

Field	Bits	Type	Description
EMUXCH	31:28	rh	**EMUX channel number** This bit indicates the external multiplexer channel number of the converted channel result.
0	27:16	r	**Reserved** Read as 0; should be written with 0

45.8.191 TMADCx result register v on LLI (AXI)

TMADCRESx_RESAXIv (v=0-15;x=0-6) Offset address: $20040_H+x*800_H+v*4$

TMADCx result register v on LLI [AXI) Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TRE	DR	0			SWTR	TRNM						CHNR			
rh	rh	r			rh	rh						rh			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESULT															
rh															

Field	Bits	Type	Description
RESULT	15:0	rh	**Conversion result for the configured channel**
CHNR	19:16	rh	**TMADC channel number** This bit indicates the channel number of the converted channel result.
TRNM	25:20	rh	**Trigger number** This bit indicates the trigger number of the hardware trigger source which initiated the channel conversion.
SWTR	26	rh	**Software trigger status** 0_B Channel conversion not initiated by software trigger 1_B Channel conversion initiated by software trigger
DR	30	rh	**Diagnostic result** This bit indicates if the conversion result is a diagnostics (BWD or CSD) data or not. 0_B Not a diagnostic result 1_B Diagnostic result
TRE	31	rh	**Trigger edge** This bit indicates the information on the edge of the hardware trigger that initiated the channel conversion 0_B Trigger falling edge 1_B Trigger rising edge
0	29:27	r	**Reserved** Read as 0; should be written with 0

45.8.192 TMADCx monitor channel v result register on LLI (AXI)

TMADCRESx_MRESAXIv (v=0-1;x=0-6) Offset address: 20080_H+x*800_H+v*4

TMADCx monitor channel v result register on LLI [AXI) Kernel Reset value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	DR	0													
r	rh	r													

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESULT															
rh															

Field	Bits	Type	Description
RESULT	15:0	rh	**Conversion result for the configured monitoring channel**
DR	30	rh	**Diagnostic result** This bit indicates if the conversion result is a diagnostics (CSD) data or not. 0_B Not a diagnostic result 1_B Diagnostic result
0	29:16, 31	r	**Reserved** Read as 0; should be written with 0

45.8.193 DSADCx result register main

DSADCRES_INx_RESM (x=0-9) Offset address: 24000_H+x*400_H

DSADCx result register main Kernel Reset value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESULTHI															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESULTLO															
rh															

Field	Bits	Type	Description
RESULTLO	15:0	rh	**Result value lower part** Returns the next value from the result FIFO
RESULTHI	31:16	rh	**Result value higher part** Returns an additional value

45.8.194 DSADCx result register auxiliary

DSADCRES_INx_RESA (x=0-9) Offset address: $24004_H+x*400_H$

DSADCx result register auxiliary Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESULT															
rh															

Field	Bits	Type	Description
RESULT	15:0	rh	**Most recent result of auxiliary filter**
0	31:16	r	**Reserved** Read as 0; should be written with 0

45.8.195 EXMODz result register main

DSADCRES_EXz_RESM (z=0-7) Offset address: $28000_H+z*400_H$

EXMODz result register main Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESULTHI															
rh															

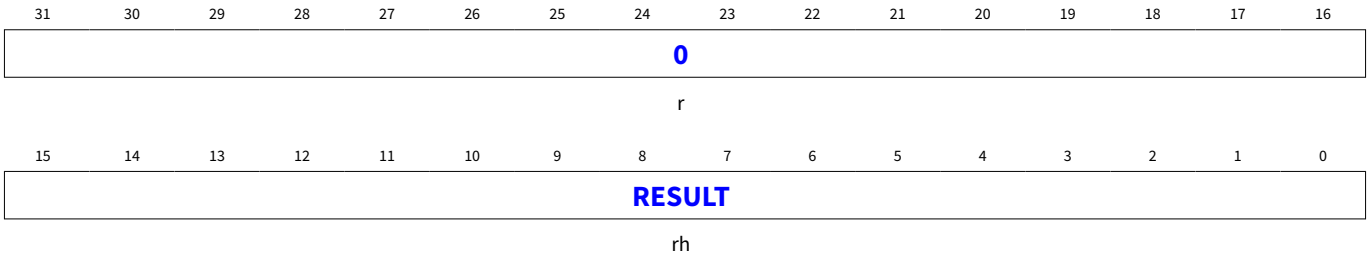
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESULTLO															
rh															

Field	Bits	Type	Description
RESULTLO	15:0	rh	**Result value lower part** Returns the next value from the result FIFO
RESULTHI	31:16	rh	**Result value higher part** Returns an additional value

45.8.196 EXMODz result register auxiliary

DSADCRES_EXz_RESA (z=0-7) Offset address: $28004_H+z*400_H$

EXMODz result register auxiliary Kernel Reset value: $0000\ 0000_H$

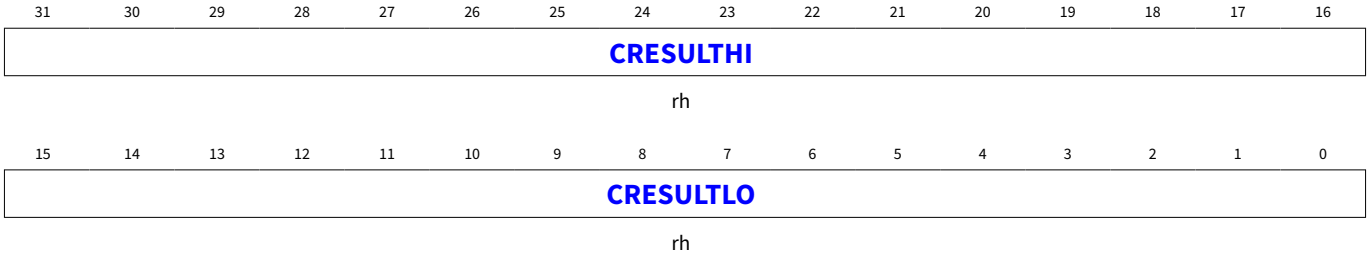


Field	Bits	Type	Description
RESULT	15:0	rh	**Most recent result of auxiliary filter**
0	31:16	r	**Reserved** Read as 0; should be written with 0

45.8.197 CDSPa result register y

CDSPRESa_RESy (a=0-11;y=0-2) Offset address: $2B000_H+a*2000_H+y*4$

CDSPa result register y Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
CRESULTLO	15:0	rh	**CDSP result value lower part** Returns the next value from the result FIFO
CRESULTHI	31:16	rh	**CDSP result value higher part** Returns an additional value

45.9 Debug information

TMADC, DSADC, EXMOD and CDSP provide individual registers for on-chip debug support. They control suspend modes and provide debug status information..

***Note**: When a module is suspended, user shall avoid kernel reset, it may result in wrong suspend status information.*

TMADC debug information

TMADC debug can be configured by the register OCSTM. Two trigger sets provide trigger information for start of conversion per TMADC instance. Select trigger sets through bit-field TGS. The selected trigger set can be output to OTGB0 or OTGB1. It is configured with bit-filed TGB. Source (TMADC instance) for the trigger set can be configured by bit-fields TRIG1SEL and TRIG2SEL (Figure below).

***Note**: Start of Conversion (SoC) is a trigger with pulse-width of 1 fSPB.*

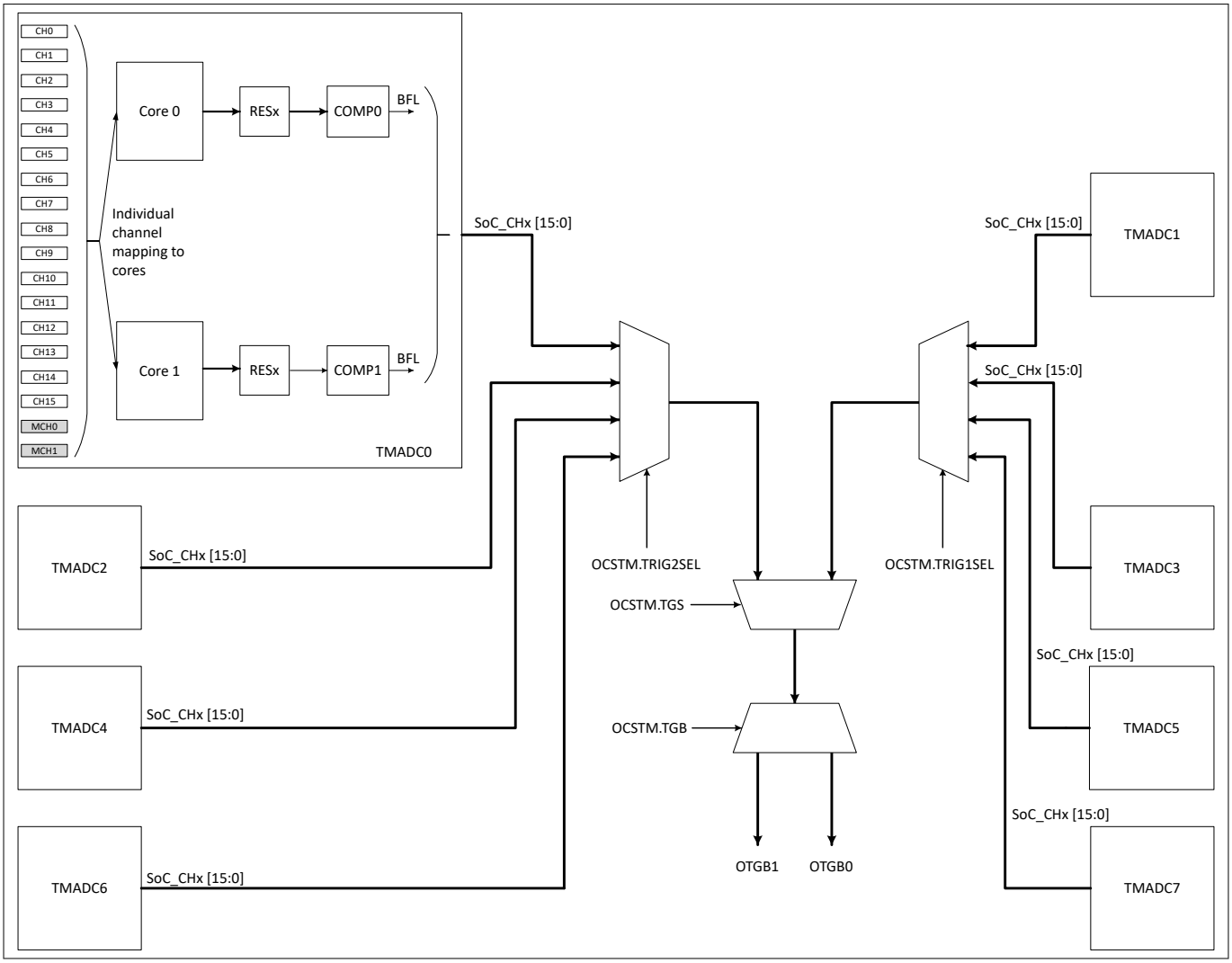


Figure 1242 Trigger source select from TMADC to OTGB

TMADC supports hard and soft suspend modes. They are selectable through bit-field SUSM. In hard suspend mode; SUSM = 01_B, module clock is immediately stopped. This may result in states not expected as static.

***Note**: Hard suspend mode shall be avoided, soft suspend modes are more useful for TMADC.*

In the soft suspend mode; SUSM = 11_B, TMADC will suspend after finishing all conversions which are in hold-state and waiting for conversion on Core 0 & Core 1. If the digital comparator is configured, the generated result is compared to threshold and a boundary flag is generated before module is suspended. User can decide to suspend multiple TMADC modules. The TMADC modules can be suspended by writing into bit-filed SUS. The status of the corresponding TMADC module is reflected in bit-field SUSTA.

Bit-fields TG_P and SUS_P protect against unintended trigger set selection and unintended module suspension.

***Note**: The first conversion result coming out of soft suspend mode will be inaccurate.*

DSADC and EXMOD debug information

DSADC and EXMOD channels can be debugged by configuring the register DSADC_OCSDSADC and DSADC_OCSEXMOD respectively.

***Note**: No trigger set on OTGB is supported by DSADC and EXMOD.*

DSADC and EXMOD support hard and soft suspend modes. In the hard suspend mode; SUSM = 01_B, module clock is immediately stopped. This may result in some unexpected static states.

***Note**: Hard suspend mode shall be avoided, soft suspend modes are more useful for DSADC and EXMOD.*

In the Soft suspend mode SUSM = 10_B DSADC and EXMOD module will suspend immediately after generating an auxiliary filter chain output. In the Soft suspend mode SUSM = 11_B the DSADC and EXMOD channels will suspend after generating a final result from the main filter chain. If the digital comparator is configured, the generated result is compared to thresholds and a boundary flag is generated before module is suspended. User can decide to suspend multiple DSADC and EXMOD channels by writing into bit-filed SUS. The status of the suspended channel i is reflected in bit-field SUSTA at bit i.

Bit-field SUS_P, protects against unintended module suspension.

CDSP debug information

CDSPs can be debugged by configuring the register CDSP_OCSCDSPA and CDSP_OCSCDSPB . CDSP_OCSCDSPB does exist for devices with more than 9 CDSP core instances.

***Note**: No trigger set on OTGB is supported by CDSP.*

CDSP supports hard and soft suspend modes. In the hard suspend mode; SUSM = 01_B module clock is immediately stopped.

***Note**: Switching to hard suspend mode may corrupt ongoing program execution after suspend mode is left and it may corrupt ongoing data memory accesses. Hard suspend mode shall be avoided, soft suspend modes are more useful for CDSP.*

In the Soft suspend mode; SUSM = 10_B CDSP will suspend operation immediately after processing the current input sample. In case a CDSP is in Halt state (indicated by CDSP_GLOBRSD.DSPRSTi= 0), it will suspend immediately.

In the Soft suspend mode; SUSM = 11_B, the CDSP will suspend after generating final result in result register RES0. In both the soft suspend modes, if the digital comparator is configured, the generated result is compared to thresholds and a boundary flag is generated before module is suspended. User can decide to suspend multiple CDSPs by writing into bit-filed SUS. The status of the corresponding suspended channel is reflected in bit-field SUSTA.

***Note**: DCCM, ICCM and DSP debug registers can not get read in suspend mode.*

Bit-field SUS_P, protects against unintended module suspension.

DSP debug access

The CDSP includes feature extensions and an embedded DSP core. Direct debug of the DSP is controlled with dedicated registers shown in the table below.

Table 2407 DSP debug registers

Register	Use
CDSP_DBd_STATUS	Provides status of all DEBUG operations. Writing to this register initiates the debug operation defined in CDSP_DBd_CMD
CDSP_DBd_CMD	Defines the debug command (read/write to aux/core/mem)
CDSP_DBd_AD	Specifies auxiliary/core/memory address location
CDSP_DBd_DATA	Specifies the data to be written for debug write operations and holds the returned data for debug read operations
CDSP_DBd_RESET	Writing triggers a reset of DSP internal debug registers. Deviation from DSP specification.

For details about debug commands please refer to the Synopsys document 'DesignWare ARC EM Databook'.

45.9.1 ARC DSP core debug features

ARM Peripheral Bus (APB) port access allows to debug the core. The debug APB port is used to interface external host (or debugger) to the resources available to the ARC EM core, including internal registers and the entire memory space.

The Debug interface is made available for systems that require access to the internals of the core. The interface is composed of memory mapped setup registers that can be written to initiate read and write operations to internal core/auxiliary registers and memory locations, as well as control operations such as instruction-stepping.

Configuration options of ARC core debug and status informations are provided by CDSP_DB_ADDR, CDSP_DB_CMD, CDSP_DB_DATA, CDSP_DB_RESET and CDSP_DB_STATUS.

Writing to the register CDSP_DB_RESET only triggers the reset of the DSP internal debug registers CDSP_DBd_ADDR and CDSP_DBd_DATA. This is a deviation from the DSP specification where CDSP system reset functionality is named and the CDSP_DBd_CMD register is reset in addition. Complete CDSP logic can get reset using Kernel reset.

CDSP_DB_STATUS.RA can be cleared by clearing the RA bit of DSP internal auxiliary register DEBUG, address 0x05.

45.9.1.1 Debug Access to ARC Core over APB

To access registers and memory within an ARC core you first need to set up the access with the DB_ADDR and DB_STATUS registers The remainder of this section lists example sequences for reads and writes.

45.9.1.1.1 Example Sequence to Read From ARC IDENTITY Register

- Read DB_STATUS (0xFFFF_0000), check bit 2 set and bit 0 clear.
- Write address to DB_ADDR (0xFFFF_0008) with 0x4.
- Write command to DB_CMD (0xFFFF_0004) with 0x6 (read from auxiliary register).
- Write DB_STATUS (0xFFFF_0000) with any value to trigger the access.
- Read DB_STATUS until bit 2 set, bit 0 clear to indicate the access has completed.
- Read data from DB_DATA (0xFFFF_000C) - should be 0x054 for ARC HS processor (single core, ARCNUM = 0) or 0x44 for ARC EM processor (single core, ARCNUM = 0).

45.9.1.1.2 Example Sequence to Write to the ARC Memory

- Read DB_STATUS (0xFFFF_0000), check bit 2 set and bit 0 clear (can be omitted if not the first access for a sequence).
- Write memory address to DB_ADDR (0xFFFF_0008).
- Write data value to DB_DATA (0xFFFF_000C).
- Write DB_CMD with 0x00 (write to memory location).
- Write DB_STATUS (0xFFFF_0000), with any value to initiate the access.
- Read DB_STATUS (0xFFFF_0000), until bit 2 set, bit 0 cleared to indicate the access completed, failure is indicated if bit 1 is set.

45.9.1.1.3 Example Sequence to Write Then Read an ARC Core Register

Write:

- Read DB_STATUS (0xFFFF_0000), check bit 2 set and bit 0 clear.
- Write address to DB_ADDR (0xFFFF_0008) with 0x0.
- Write test data to DB_DATA (0xFFFF_000C) - for example, 0xcafef00d.
- Write command to DB_CMD (0xFFFF_0004) with 0x1 (write to core register).
- Write DB_STATUS (0xFFFF_0000), with any value to trigger the access.
- Read DB_STATUS (0xFFFF_0000), until bit 2 set, bit 0 clear to indicate the access has completed.

Read Back:

- Write address to DB_ADDR (0xFFFF_0008) with 0x0.
- Write command to DB_CMD (0xFFFF_0004) with 0x5 (read from core register).
- Write DB_STATUS (0xFFFF_0000), with any value to trigger the access.
- Read DB_STATUS (0xFFFF_0000),until bit 2 set, bit 0 clear to indicate the access has completed.
- Read data from DB_DATA (0xFFFF_000C), compare with written value.

45.9.1.2 Interface Signals

***Note**: The interface supports synchronous clock ratios between the core clock and the APB debug-port clock using a clock-enable input to indicate cycles with a rising edge in both domains.*

Table 2408 APB Interface Signals

Name	Width	Direction	Comment
pclkdbg_en	1	Input	APB clock enable (tie to 1'b1 for 1:1 clocking)
presetdbgn	1	Input	Active low reset
paddrdbg	30 [31:2]	Input	32-bit addressing only
pseldbg	1	Input	Selects the external debug interface
penabledbg	1	Input	Indicates second, and subsequent, cycle of a transfer
pwritedbg	1	Input	Indicates access is a write transfer Distinguishes between a read, LOW, and a write, HIGH
pwdatadbg	32	Input	Writer data bus

(table continues...)

Table 2408 (continued) APB Interface Signals

Name	Width	Direction	Comment
preadydbg	1	Output	Extends a APB transfer by the inserting wait states.
prdatadbg	32	Output	Read data bus.
pslverrdbg	1	Output	Slave-generated error response.
dbg_prot_sel	1	Input	Assert high to select the APB port. Assert low to select the BVCI debug port.

Table 2409 CoreSight Authentication Interface Signals

Name	Width	Direction	Comments
dbgen	1	Input	Authentication interface (tie high if not in use)
niden	1	Input	Authentication interface (tie high if not in use)

For full details on the CoreSight Authentication Interface, see the ARM CoreSight Architecture Specification.

45.10 References

For more information about the CDSP programming, user can refer to:

- ARC V2 programmers reference for ARC EM 5 processor

45.11 ADC revision history

Reference	Description of change(s)
Date range: 2025-03-14 to 2025-04-11	
Global service request and alarm generation, Boundary flag interface to timer, Central functions (CDSP), Limit checking (CDSP)	• Additional limit check resource options per CDSP
Time-stamp support (CDSP)	• Typo fixed: figure title "Time-stamp generation" moved to the right position
Date range: 2024-11-22 to 2025-03-13	
Trigger handling and arbitration (TMADC)	• Extended exception to cover implementation
Channel-core self diagnostics (TMADC)	• Added CSD implementation option for RRAM devices
External multiplexer control (TMADC)	• Clarification on minimum sampling time for EMUX channels
Functional overview (DSADC)	• Corrected EXMOD channels from [8:1] to [7:0] • Replaced VREFX with VDDM/2
Gain adjustment (DSADC)	• Gain adjustment calculation corrections
Group delay and setting time (DSADC)	• Correction of group delay and settling time calculation
Ramp mode (FCC)	• Set minimum glitch filter depth in FCC ramp mode #5 to 4
Date range: 2024-08-17 to 2024-11-21	
On-chip signal monitoring and supervision	• Revised description of V_{DDK} measurement
Module start-up calibration, Single channel one-shot conversion	• Corrected references to data-sheet parameter t_{CAL}
Figure 1183, ADC interface to GTM and eGTM, Service request from ADC to GTM	• Additional interrupt routing to TIM (eGTM, GTM)

45.12 TC4Dx ADC information

45.12.1 TC4Dx ADC configuration

Table 2410 ADC configuration for TC4D

Functional block	Number of instances
TMADC	4 (8 cores)
DSADC	0
CDSP	2
FCC	0
EXMOD	0

The following table shows the pull-down diagnostic (PDD) resistor connections in TC4Dx:

Table 2411 Pull-down diagnostic

Control bit	Analog input
PDDCFG.EN0	AN7
PDDCFG.EN1	AN11
PDDCFG.EN2	AN19
PDDCFG.EN3	AN27
PDDCFG.EN4	AN35
PDDCFG.EN5	AN43

***Note**: Not all the analog inputs are available in all packages. Please refer to data-sheet for pinning information.*

Related information

On-chip signal monitoring and supervision on page 8361

45.12.2 TC4Dx ADC features

The functional description is listed in the family part. Please be aware, that no DSADC, EXMOD or FCC blocks exist in this specific device.

45.12.3 TC4Dx ADC functional description

The ADC cluster is connected to the COMPB bus. It does not connect to the Low Latency Interface (LLI).

Due to the absence of DSADC and GTM, the corresponding hardware trigger sources are not available. Similarly, the input of CDSP cannot be connected to the not available DSADC, EXMOD or CARMAG result registers. Further more, the ADI connection between TMADC and GTM does not exist.

CDSP limit checking

TC4Dx does include a single comparator for limit checking per CDSP (family default).

TMADC output supervision signal

Each TMADC module has an output supervision multiplexer to select the output supervision signal (OSS) to be monitored by another TMADC. The signal to be monitored is selected with TMADCx_MODCFG.OSSEL.

Only one TMADC output signal can be selected for output supervision signal generation at any given time and the module selection is made through bit SEL in the TMSMOD register.

The supervision signals are connected to the monitor channels MCH0 of TMADC1 and MCH1 of TMADC2 as shown in the following figure:

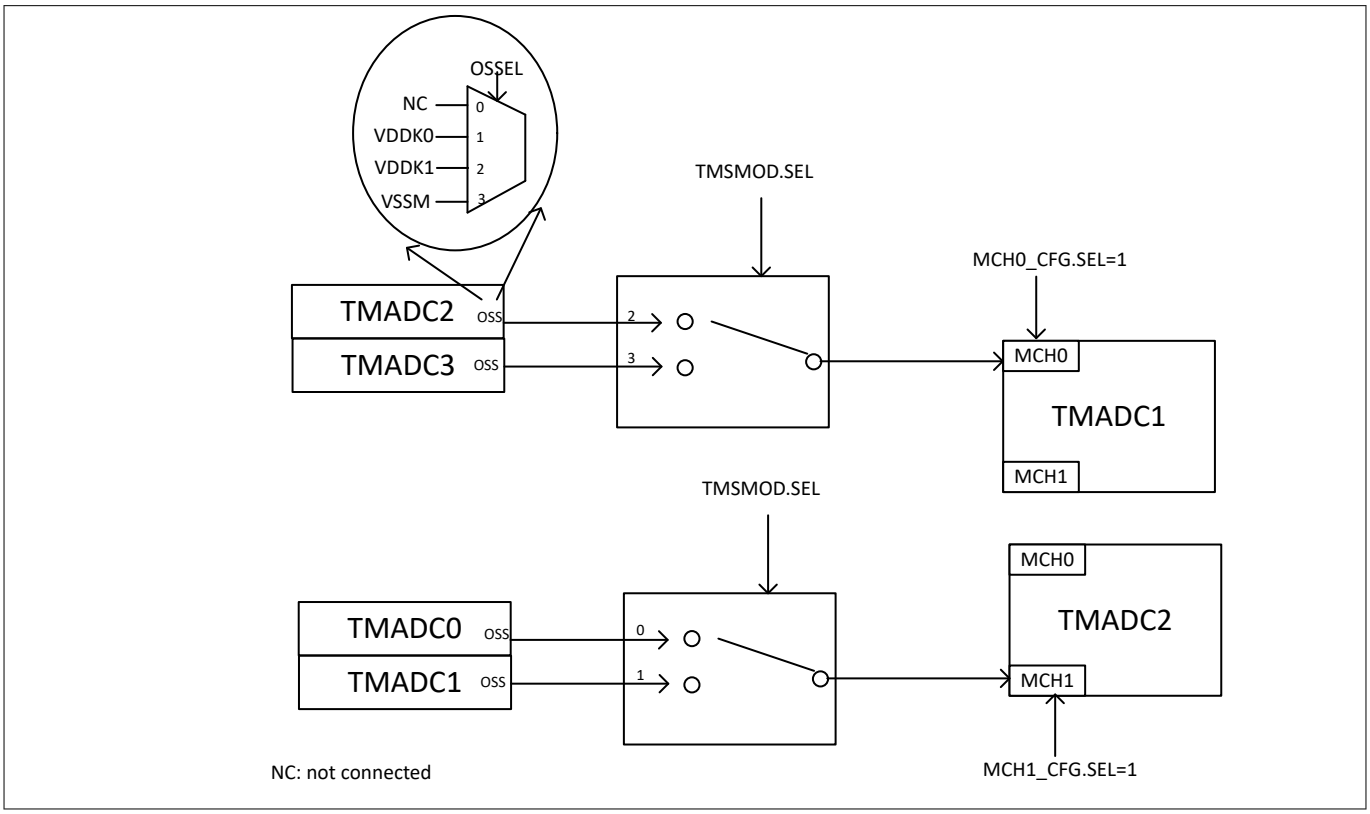


Figure 1243 TC4Dx supervision signal

Reference voltages

Table 2412 Reference voltage mapping for TC4Dx

Converter Group	Associated reference pins
TMADC0	VAREF1/VAGND1
TMADC1	VAREF1/VAGND1
TMADC2	VAREF2/VAGND2
TMADC3	VAREF2/VAGND2

45.12.4 TC4Dx ADC registers

45.12.4.1 Memory overview tables of ADC

Table 2413 Memory overview - ADC (ascending address)

Short name	Long name	Address
ADC_CDSP_DSPa_ICCM (a=0)	Instruction closely coupled memory ($00C00_H$ Byte)	$F50C0000_H$
ADC_CDSP_DSPa_DCCM (a=0)	Data closely coupled memory ($00C00_H$ Byte)	$F50C1000_H$
ADC_CDSP_DSPa_ICCM (a=1)	Instruction closely coupled memory ($00C00_H$ Byte)	$F50C2000_H$
ADC_CDSP_DSPa_DCCM (a=1)	Data closely coupled memory ($00C00_H$ Byte)	$F50C3000_H$

45.12.4.2 Register address space - ADC

Table 2414 Registers address space - ADC

Module	Base address	End address	Note
ADC	$F5000000_H$	$F50E3FFF_H$	

45.12.4.3 Register overview - access mode glossary

Table 2415 Register overview - access mode glossary

Keyword	Description
Ex (x=0-15)	Access protection using PROT register ADC_PROTEx .
SE	Access protection using PROT register ADC_PROTSE .
APU-PCTRLy (y=0-15)	Protection group consisting of registers ADC_ACCENy_WRA , ADC_ACCENy_WRB , ADC_ACCENy_RDA , ADC_ACCENy_RDB , ADC_ACCENy_VM , ADC_ACCENy_PRS .
PCTRLy	Access protection using *APU-PCTRLy* registers.
PTOP ETOP PADC EADC	Access protection using one APU from *APU-PCTRL[y] (y=0-15)*. PROT_E protection using one of *E[x] (x=0-15)*. The protection source to be used is selected by bit-field: RESALLOC_GLOB.GLOB
PMUXz EMUXz	Access protection using one APU from *APU-PCTRL[y] (y=0-15)*. PROT_E protection using one of *E[x] (x=0-15)*. The protection source to be used is selected by bit-field: z=38: RESALLOC_GLOB.CDSPGL

(table continues...)

Table 2415 (continued) Register overview - access mode glossary

Keyword	Description
PTMADCz (z=0-3) ETMADCz (z=0-3)	Access protection using one APU from *APU-PCTRL[y] (y=0-15)*. PROT_E protection using one of *E[x] (x=0-15)*. The protection source to be used is selected by bit-field: z=0-3: RESALLOC_TMADC.TMADCz
PCDSPz (z=0-1) ECDSPz (z=0-1)	Access protection using one APU from *APU-PCTRL[y] (y=0-15)*. PROT_E protection using one of *E[x] (x=0-15)*. The protection source to be used is selected by bit-field: z=0-1: RESALLOC_CDSPA.CDSPz
BE	Always returns a Bus Error.
U	No access restrictions.
PROT	Access restrictions as defined in the PROT register access rules.
32	Access only when using 32-bit width.

45.12.4.4 Registers overview - ADC (ascending offset address)

Table 2416 Registers overview - ADC (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ADC_CLC	Clock Control Register	00000_H	PCTRL0	PCTRL0, SV, E0	Application Reset	8403
ADC_ID	Module Identification Register	00008_H	PCTRL0	BE	PowerOn Reset	8403
ADC_PROTSE	PROT Register Safe Endinit	$0000C_H$	U	SV, PROT	Application Reset	8404
ADC_RESALLOC_GLOB	Resource allocation register for Global Rsources	00010_H	U	SE, SV	Application Reset	8558
ADC_RESALLOC_TMADC	Resource allocation register for TMADC Resources	00014_H	U	SE, SV	Application Reset	8558
ADC_RESALLOC_CDSPA	Resource allocation register for CDSP	$0002C_H$	U	SE, SV	Application Reset	8559
ADC_PROTEx (x=0-15)	APUx PROT Register Endinit	00080_H+ x*4	U	SV, PROT	Application Reset	8410
ADC_ACCENy_WRA (y=0-15)	Write access enable register A, APU y	$000C0_H$+ y*20_H	U	SE, SV	Application Reset	8412
ADC_ACCENy_WRB (y=0-15)	Write access enable register B, APU y	$000C4_H$+ y*20_H	U	SE, SV	Application Reset	8413

(table continues...)

Table 2416 (continued) Registers overview - ADC (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ADC_ACCENy_RDA (y=0-15)	Read access enable register A, APU y	$000C8_H$+ y*20_H	U	SE, SV	Application Reset	8413
ADC_ACCENy_RDB (y=0-15)	Read access enable register B, APU y	$000CC_H$+ y*20_H	U	SE, SV	Application Reset	8414
ADC_ACCENy_VM (y=0-15)	VM access enable register, APU y	$000D0_H$+ y*20_H	U	SE, SV	Application Reset	8414
ADC_ACCENy_PRS (y=0-15)	PRS access enable register, APU y	$000D4_H$+ y*20_H	U	SE, SV	Application Reset	8415
ADC_RSTi_CTRLA (i=0-15)	Reset Control Register A, APU i	$002C0_H$+ i*12	PCTRLi	PCTRLi, SV, Ei	Application Reset	8416
ADC_RSTi_CTRLB (i=0-15)	Reset Control Register B, APU i	$002C4_H$+ i*12	PCTRLi	PCTRLi, SV, Ei	Application Reset	8416
ADC_RSTi_STAT (i=0-15)	Reset Status Register, APU i	$002C8_H$+ i*12	PCTRLi	BE	Application Reset	8417
ADC_GLSRCFGi (i=0-3)	ADC global service request i configuration register	00800_H+i *4	PTOP	PTOP	Kernel Reset	8559
ADC_GLALCFGi (i=0-7)	ADC global alarm i configuration register	00810_H+i *4	PTOP	PTOP	Kernel Reset	8560
ADC_GLALSET	ADC global alarm set register	00830_H	PTOP	PTOP	Kernel Reset	8420
ADC_GLSRSET	ADC global service request set register	00834_H	PTOP	PTOP	Kernel Reset	8420
ADC_ETBFxSEL (x=0-2)	ADC to eGTM boundary flag select register, cluster x	00838_H+ x*4	PTOP	PTOP	Kernel Reset	8561
ADC_PDDCFG	ADC pull down diagnostics configuration register	$0088C_H$	PTOP	PTOP	Kernel Reset	8425
ADC_CLKEN_TMADC	Clock enable for TMADC	00890_H	clk	clk	Kernel Reset	8562
ADC_CLKEN_CDSP	Clock enable for CDSP	$0089C_H$	32, clk	32, clk	Kernel Reset	8562
ADC_SUPLLEV	Supply level selection for ADC	$008A0_H$	PTOP	PTOP, clk	Kernel Reset	8428
ADC_TMSMOD	TMADC supervision module register	$008A4_H$	PTOP	PTOP, clk	Kernel Reset	8429
ADC_OCSTM	OCDS control and status register for TMADC	$008A8_H$	PTOP	SV, PTOP, clk	Debug Reset	8563

(table continues...)

Table 2416 (continued) Registers overview - ADC (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ADC_TMADCx_MODCFG (x=0-3)	TMADCx module configuration register	20000_H+ x*800_H	PTMADCx	PTMADCx, clk	Kernel Reset	8431
ADC_TMADCx_EMUXCFG (x=0-3)	TMADCx EMUX configuration register	20004_H+ x*800_H	PTMADCx	PTMADCx, clk	Kernel Reset	8432
ADC_TMADCx_SWTRCFG (x=0-3)	TMADCx software trigger configuration register	$2000C_H$+ x*800_H	PTMADCx, 32	PTMADCx, 32, clk	Kernel Reset	8433
ADC_TMADCx_CMATCFG (x=0-3)	TMADCx connection matrix configuration register	20010_H+ x*800_H	PTMADCx	PTMADCx, clk	Kernel Reset	8433
ADC_TMADCx_CHy_STC (x=0-3;y=0-15)	TMADCx sampling time control register, channel y	20018_H+ x*800_H+ y*12	PTMADCx	PTMADCx, clk	Kernel Reset	8434
ADC_TMADCx_CHy_CFG (x=0-3;y=0-15)	TMADCx configuration register, channel y	$2001C_H$+ x*800_H+ y*12	PTMADCx	PTMADCx, clk	Kernel Reset	8435
ADC_TMADCx_CHy_STAT (x=0-3;y=0-15)	TMADCx status register, channel y	20020_H+ x*800_H+ y*12	PTMADCx	PTMADCx, clk	Kernel Reset	8436
ADC_TMADCx_MCHy_CFG (x=0-3;y=0-1)	TMADCx configuration register, monitor channel y	$200D8_H$+ x*800_H+ y*4	PTMADCx	PTMADCx, clk	Kernel Reset	8437
ADC_TMADCx_BNDn_SEL (n=0-1;x=0-3)	TMADCx boundary select register n	$200E0_H$+ x*800_H+ n*12	PTMADCx	PTMADCx, clk	Kernel Reset	8438
ADC_TMADCx_BNDn_STAT (n=0-1;x=0-3)	TMADCx boundary status regsiter n	$200E4_H$+ x*800_H+ n*12	PTMADCx	PTMADCx, clk	Kernel Reset	8439
ADC_TMADCx_BNDn_TRGCFG (n=0-1;x=0-3)	TMADCx boundary trigger configuration register, boundary register n	$200E8_H$+ x*800_H+ n*12	PTMADCx	PTMADCx, clk	Kernel Reset	8439
ADC_TMADCx_BFLn_CFG (n=0-1;x=0-3)	TMADCx boundary flag configuration register n	$200F8_H$+ x*800_H+ n*8	PTMADCx	PTMADCx, clk	Kernel Reset	8440
ADC_TMADCx_BFLn_STAT (n=0-1;x=0-3)	TMADCx boundary flag status register n	$200FC_H$+ x*800_H+ n*8	PTMADCx	PTMADCx, clk	Kernel Reset	8441
ADC_TMADCx_RSLTu_CFG (u=0-15;x=0-3)	TMADCx configuration register, result register u	$2010C_H$+ x*800_H+ u*8	PTMADCx	PTMADCx, clk	Kernel Reset	8441

(table continues...)

Table 2416 (continued) Registers overview - ADC (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ADC_TMADCx_MRSLTu_CFG (u=0-1;x=0-3)	TMADCx configuration register, monitor channel u result register	20188_H+x*800_H+u*4	PTMADCx	PTMADCx, clk	Kernel Reset	8442
ADC_TMADCx_ERRF (x=0-3)	TMADCx error event flag register	20190_H+x*800_H	PTMADCx	PTMADCx, clk	Kernel Reset	8443
ADC_TMADCx_RESF (x=0-3)	TMADCx result event flag register	20194_H+x*800_H	PTMADCx	PTMADCx, clk	Kernel Reset	8444
ADC_TMADCx_BNDF (x=0-3)	TMADCx boundary event flag register	20198_H+x*800_H	PTMADCx	PTMADCx, clk	Kernel Reset	8444
ADC_TMADCx_RESFSET (x=0-3)	TMADCx result event flag set register	$201A0_H$+x*800_H	PTMADCx, 32	PTMADCx, 32, clk	Kernel Reset	8445
ADC_TMADCx_ERRFCLR (x=0-3)	TMADCx error event flag clear register	$201A4_H$+x*800_H	PTMADCx, 32	PTMADCx, 32, clk	Kernel Reset	8445
ADC_TMADCx_RESFCLR (x=0-3)	TMADCx result event flag clear register	$201A8_H$+x*800_H	PTMADCx, 32	PTMADCx, 32, clk	Kernel Reset	8446
ADC_TMADCx_BNDFCLR (x=0-3)	TMADCx boundary event flag clear register	$201AC_H$+x*800_H	PTMADCx, 32	PTMADCx, 32, clk	Kernel Reset	8446
ADC_TMADCx_SRi_CFG (i=0-6;x=0-3)	TMADCx service request configuration register i	$201B0_H$+x*800_H+i*4	PTMADCx	PTMADCx, clk	Kernel Reset	8447
ADC_TMADCx_MRESv (v=0-1;x=0-3)	TMADCx monitor channel v result register	$201CC_H$+x*800_H+v*4	PTMADCx	PTMADCx, clk	Kernel Reset	8448
ADC_TMADCx_MODSTAT (x=0-3)	TMADCx module status register	$201D4_H$+x*800_H	PTMADCx	PTMADCx, clk	Kernel Reset	8449
ADC_TMADCx_BWDCFG (x=0-3)	TMADCx broken wire detection control register	$201D8_H$+x*800_H	PTMADCx	PTMADCx, clk	Kernel Reset	8449
ADC_TMADCx_CSDn (n=0-1;x=0-3)	TMADCx converter self-diagnostics register core-n	$201DC_H$+x*800_H+n*4	PTMADCx	PTMADCx, clk	Kernel Reset	8450
ADC_TMADCx_AW0_TSv (v=0-15;x=0-3)	TMADCx time-stamp register v	20200_H+x*800_H+v*4	PTMADCx	PTMADCx, clk	Kernel Reset	8451

(table continues...)

Table 2416 (continued) Registers overview - ADC (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ADC_TMADCx_AW0_RESv (v=0-15;x=0-3)	TMADCx result register v	20240_H+x*800_H+v*4	PTMADCx	PTMADCx, clk	Kernel Reset	8451
ADC_TMADCx_AW1_TSv (v=0-15;x=0-3)	TMADCx time-stamp register v	20280_H+x*800_H+v*8	PTMADCx	PTMADCx, clk	Kernel Reset	8451
ADC_TMADCx_AW1_RESv (v=0-15;x=0-3)	TMADCx result register v	20284_H+x*800_H+v*8	PTMADCx	PTMADCx, clk	Kernel Reset	8451
ADC_TMADCx_AW2_TSv (v=0;x=0-3)	TMADCx time-stamp register v	20300_H+x*800_H	PTMADCx	PTMADCx, clk	Kernel Reset	8451
ADC_TMADCx_AW2_RESv (v=0-6;x=0-3)	TMADCx result register v	20304_H+x*800_H+v*4	PTMADCx	PTMADCx, clk	Kernel Reset	8451
ADC_TMADCx_AW2_TSv (v=7;x=0-3)	TMADCx time-stamp register v	20340_H+x*800_H	PTMADCx	PTMADCx, clk	Kernel Reset	8451
ADC_TMADCx_AW2_RESv (v=7-13;x=0-3)	TMADCx result register v	20344_H+x*800_H+(v-7)*4	PTMADCx	PTMADCx, clk	Kernel Reset	8451
ADC_TMADCx_AW2_TSv (v=14;x=0-3)	TMADCx time-stamp register v	20360_H+x*800_H	PTMADCx	PTMADCx, clk	Kernel Reset	8451
ADC_TMADCx_AW2_RESv (v=14-15;x=0-3)	TMADCx result register v	20364_H+x*800_H+(v-14)*4	PTMADCx	PTMADCx, clk	Kernel Reset	8451
ADC_TMADCx_AW3_RESv (v=0-4;x=0-3)	TMADCx result register v	$2036C_H$+x*800_H+v*4	PTMADCx	PTMADCx, clk	Kernel Reset	8451
ADC_CDSP_GLOBRD	CDSP global reset control	$BF804_H$	U	32, clk	Kernel Reset	8565
ADC_CDSP_BNDCF	CDSP boudary flag event register	$BF808_H$	U	32, clk	Kernel Reset	8565
ADC_CDSP_BNDCFCL	CDSP boundary flag event clear	$BF80C_H$	U	32, clk	Kernel Reset	8566
ADC_CDSP_SLPST	CDSP sleep status register	$BF810_H$	U	clk	Kernel Reset	8566
ADC_CDSP_RESEVy (y=0-2)	CDSP event flag register y	$BF814_H$+y*4	U	32, clk	Kernel Reset	8567

(table continues...)

Table 2416 (continued) Registers overview - ADC (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ADC_CDSP_RESEVCLRy (y=0-2)	CDSP result event flag clear y	$BF820_H$+y*4	U	32, clk	Kernel Reset	8567
ADC_CDSP_GPx (x=0-3)	CDSP input from general purpose register x	$BF8C0_H$+x*4	PMUX38	PMUX38, clk	Kernel Reset	8511
ADC_CDSP_WUERREV	CDSP wake up error flag register	$BF8D0_H$	U	32, clk	Kernel Reset	8567
ADC_CDSP_WUERRCL	CDSP wake up error event clear register	$BF8D4_H$	U	32, clk	Kernel Reset	8568
ADC_CDSP_GLOBRCD	CDSP global run control register	$BF8D8_H$	U	32, clk	Kernel Reset	8568
ADC_CDSP_GLOBHCD	CDSP global halt control register	$BF8DC_H$	U	32, clk	Kernel Reset	8569
ADC_CDSP_GLOBRSD	CDSP global run status register	$BF8E0_H$	U	clk	Kernel Reset	8569
ADC_CDSP_GLOBHSTD	CDSP global halt status register	$BF8E4_H$	U	clk	Kernel Reset	8570
ADC_CDSP_OCSCDSPA	OCDS control and status register 0 for CDSP0-8	$BF8E8_H$	U	SV, 32, clk	Debug Reset	8570
ADC_CDSP_DBd_STATUS (d=0-1)	CDSPd debug status register	$BFC00_H$+d*20_H	SV, PCDSPd, clk	SV, PCDSPd, 32, clk	Kernel Reset	8517
ADC_CDSP_DBd_CMD (d=0-1)	CDSPd debug command register	$BFC04_H$+d*20_H	SV, PCDSPd, clk	SV, PCDSPd, 32, clk	Kernel Reset	8518
ADC_CDSP_DBd_ADDR (d=0-1)	CDSPd debug address register	$BFC08_H$+d*20_H	SV, PCDSPd, clk	SV, PCDSPd, 32, clk	Kernel Reset	8519
ADC_CDSP_DBd_DATA (d=0-1)	CDSPd debug data register	$BFC0C_H$+d*20_H	SV, PCDSPd, clk	SV, PCDSPd, 32, clk	Kernel Reset	8519
ADC_CDSP_DBd_RESET (d=0-1)	CDSPd debug reset register	$BFC10_H$+d*20_H	SV, PCDSPd, clk	SV, PCDSPd, 32, clk	Kernel Reset	8520
ADC_CDSP_DSPa_DSPCFG (a=0-1)	CDSPa configuration register	$C1C00_H$+a*2000_H	PCDSPa	PCDSPa, clk	Kernel Reset	8520
ADC_CDSP_DSPa_BNDTRCFG (a=0-1)	CDSPa boundary trigger configuration register	$C1C04_H$+a*2000_H	PCDSPa	PCDSPa, clk	Kernel Reset	8522

(table continues...)

Table 2416 (continued) Registers overview - ADC (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ADC_CDSP_DSPa_BNDCFG (a=0-1)	CDSPa boundary mode configuration register	$C1C08_H$+ a*2000_H	PCDSPa	PCDSPa, clk	Kernel Reset	8523
ADC_CDSP_DSPa_BOUNDST (a=0-1)	CDSPa boundary status register	$C1C0C_H$+ a*2000_H	PCDSPa	PCDSPa, clk	Kernel Reset	8523
ADC_CDSP_DSPa_BOUNDSEL (a=0-1)	CDSPa boundary select register	$C1C10_H$+ a*2000_H	PCDSPa, 32	PCDSPa, 32, clk	Kernel Reset	8524
ADC_CDSP_DSPa_TSTRCFG (a=0-1)	CDSPa time stamp trigger configuration register	$C1C14_H$+ a*2000_H	PCDSPa	PCDSPa, clk	Kernel Reset	8524
ADC_CDSP_DSPa_TSTMP (a=0-1)	CDSPa time-stamp register	$C1C18_H$+ a*2000_H	PCDSPa	PCDSPa, clk	Kernel Reset	8525
ADC_CDSP_DSPa_TSCNT (a=0-1)	CDSPa time-stamp counter	$C1C1C_H$+ a*2000_H	PCDSPa	PCDSPa, clk	Kernel Reset	8525
ADC_CDSP_DSPa_SRTRCFG (a=0-1)	CDSPa service request trigger configuration register	$C1C20_H$+ a*2000_H	PCDSPa	PCDSPa, clk	Kernel Reset	8526
ADC_CDSP_DSPa_SRCFG (a=0-1)	CDSPa service request configuration register	$C1C24_H$+ a*2000_H	PCDSPa	PCDSPa, clk	Kernel Reset	8571
ADC_CDSP_DSPa_SWCFG (a=0-1)	CDSPa software configuration register	$C1C28_H$+ a*2000_H	PCDSPa	PCDSPa, clk	Kernel Reset	8528
ADC_CDSP_DSPa_INTIVAL (a=0-1)	CDSPa intermediate integration value	$C1C2C_H$+ a*2000_H	PCDSPa	PCDSPa, clk	Kernel Reset	8528
ADC_CDSP_DSPa_SWTRCFG (a=0-1)	CDSPa software trigger configuration register	$C1C30_H$+ a*2000_H	PCDSPa	PCDSPa, clk	Kernel Reset	8529
ADC_CDSP_DSPa_CHERRCL (a=0-1)	CDSPa channel error clear register	$C1C34_H$+ a*2000_H	PCDSPa	PCDSPa, clk	Kernel Reset	8572
ADC_CDSP_DSPa_DSPST (a=0-1)	CDSPa status register	$C1C38_H$+ a*2000_H	PCDSPa	PCDSPa, clk	Kernel Reset	8573
ADC_CDSP_DSPa_INPUTMON (a=0-1)	CDSPa input from DSADC, TMADC, EXMOD or CARMAG	$C1C3C_H$+ a*2000_H	PCDSPa	PCDSPa, clk	Kernel Reset	8532

(table continues...)

Table 2416 (continued) Registers overview - ADC (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ADC_CDSP_DSPa_RESy (a=0-1;y=0-2)	CDSPa result register y	$C1C40_H$+ a*2000_H +y*4	PCDSPa	PCDSPa, clk	Kernel Reset	8532

45.12.4.5 Resource allocation register for Global Rsources

ADC_RESALLOC_GLOB Offset address: 00010_H

Resource allocation register for Global Rsources Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				CDSPGL				0				GLOB			
r				rw				r				rw			

Field	Bits	Type	Description
GLOB	3:0	rw	**APU Asignment for Global parts of ADC**
CDSPGL	11:8	rw	**APU Asignment for Global parts of CDSP**
0	7:4, 31:12	r	**Reserved** Read as 0; should be written with 0

45.12.4.6 Resource allocation register for TMADC Resources

ADC_RESALLOC_TMADC Offset address: 00014_H

Resource allocation register for TMADC Resources Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TMADC3				TMADC2				TMADC1				TMADC0			
rw				rw				rw				rw			

Field	Bits	Type	Description
TMADCz (z=0-3)	4*z+3:4*z	rw	**APU Asignment for Corresponsing TMADC**
0	31:16	r	**Reserved** Read as 0; should be written with 0.

45.12.4.7 Resource allocation register for CDSP

ADC_RESALLOC_CDSPA Offset address: $0002C_H$

Resource allocation register for CDSP Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0				0				0				0			
r				r				r				r			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				0				CDSP1				CDSP0			
r				r				rw				rw			

Field	Bits	Type	Description
CDSP0	3:0	rw	**APU Asignment for Corresponsing CDSP 0**
CDSP1	7:4	rw	**APU Asignment for Corresponsing CDSP1**
0	11:8, 15:12, 19:16, 23:20, 27:24, 31:28	r	**Reserved** Read as 0; should be written with 0.

45.12.4.8 ADC global service request i configuration register

Global service requests are used for generating result events to IR when a set of analog channles have finished converting and results of these channels are available. The set of analog channels in this case are distributed across different TMADC instances. Global service requests are limited to TMADC channels.

Only the service request configured with SRCFG6 register in TMADCx is a valid input to the global service request.

ADC_GLSRCFGi (i=0-3) Offset address: 00800_H+i*4

ADC global service request i configuration register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0							EVOP	0				EVEN			
r							rw	r				rw			

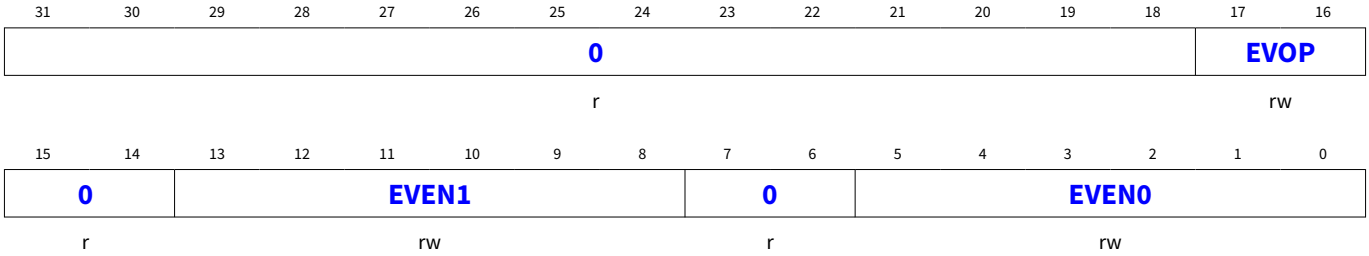
Field	Bits	Type	Description
EVEN	3:0	rw	**Event Enable** Each bit in this bitfield, if set, enables the event on SR6 for the corressponding TMADCx to be an input to the corressponding global service request 0_B Event disabled 1_B Event enabled
EVOP	8	rw	**Event Operation** This bit defines what logical operation shall be executed and under which situation global service request is generated 0_B OR logic operation 1_B AND logic operation
0	7:4, 31:9	r	**Reserved** Read as 0; should be written with 0

45.12.4.9 ADC global alarm i configuration register

Global alarms are used for generating alarms to SMU and also same alarms go as interrupts to IR. They can be generated combining 2 bounday flags from any converter.

ADC_GLALCFGi (i=0-7) Offset address: 00810_H+i*4

ADC global alarm i configuration register Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
EVEN0	5:0	rw	**Event Enable** 1st input to alarm generation. Select one of the 64 boundary flags. BFL0 - BFL15 are from TMADC. 2 BFLs for each TMADC BFL16-BFL25 are from FC. 1 BFL from each FC BFL26 - BFL35 are from DSINT. 1 BFL from each DSINT BFL36-BFL43 are from DSEXT. 1 BFL from each DSEXT BFL44-BFL61 are from CDSPs. 1 BFL from each CDSP BFL62 and BFL63 are reserved and not connected

(table continues...)

(continued)

Field	Bits	Type	Description
EVEN1	13:8	rw	**Event Enable** 2nd input to alarm generation. Select one of the 64 boundary flags. BFL0 - BFL15 are from TMADC. 2 BFLs for each TMADC BFL16-BFL25 are from FC. 1 BFL from each FC BFL26 - BFL35 are from DSINT. 1 BFL from each DSINT BFL36-BFL43 are from DSEXT. 1 BFL from each DSEXT BFL44-BFL61 are from CDSPs. 1 BFL from each CDSP BFL62 and BFL63 are reserved and not connected
EVOP	17:16	rw	**Event Operation** This bit defines what logical operation shall be executed and under which situation global service request is generated 00_B Alarm disabled 01_B OR logical operation The input to the OR gate are the input selected by EVENT0 and EVENT1. As soon as first high input comes then the alarm is generated. Every time there is a high input on EVENT0 or EVENT1 an alarm is generated 10_B AND logical operation The input to the AND gate are the input selected by EVENT0 and EVENT1. If EVENT0 and EVENT1 are coming at different time, the events remain latched until the alarm is generated 11_B Reserved
0	7:6, 15:14, 31:18	r	**Reserved** Read as 0; should be written with 0

45.12.4.10 ADC to eGTM boundary flag select register, cluster x

ADC_ETBFxSEL (x=0-2) Offset address: 00838_H+x*4

ADC to eGTM boundary flag select register, cluster x Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0				OUT3SEL						OUT2SEL					
r				rw						rw					

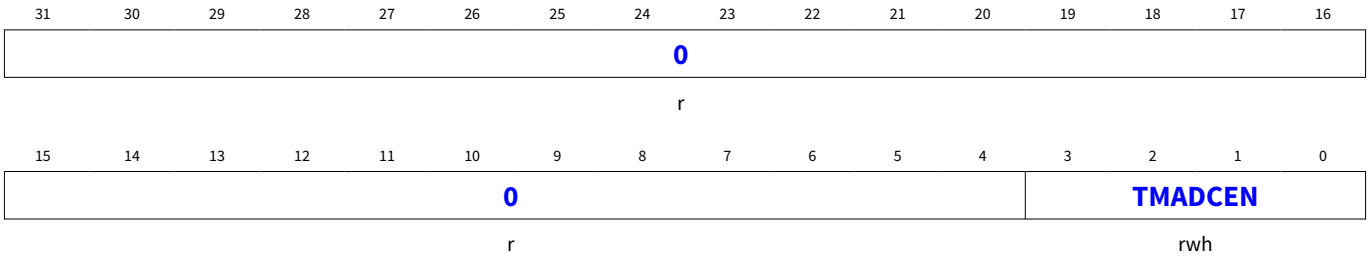
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				OUT1SEL						OUT0SEL					
r				rw						rw					

Field	Bits	Type	Description
OUTySEL (y=0-1)	6*y+5:6*y	rw	**Boundary flag y output select** 00_H Boundary flag 0 selected 01_H Boundary flag 1 selected 02_H Boundary flag 2 selected … $3F_H$ Boundary flag 63 selected
OUTySEL (y=2-3)	6*y+9:6*y+4	rw	**Boundary flag y output select** 00_H Boundary flag 0 selected 01_H Boundary flag 1 selected 02_H Boundary flag 2 selected … $3F_H$ Boundary flag 63 selected
0	15:12, 31:28	r	**Reserved** Read as 0; should be written with 0

45.12.4.11 Clock enable for TMADC

ADC_CLKEN_TMADC Offset address: 00890_H

Clock enable for TMADC Kernel Reset value: $0000\ 0000_H$



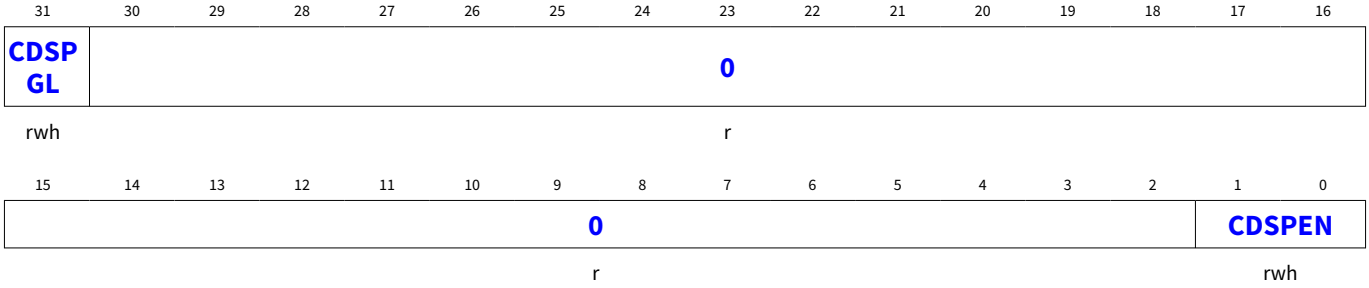
Field	Bits	Type	Description
TMADCEN	3:0	rwh	**Clock enable for TMADC** Each bit controls the clock for the corresponding TMADC channel. E.g: bit0 for TMADC0. 0 - No action 1 - Clock enable for TMADC instance
0	31:4	r	**Reserved** Read as 0; should be written with 0

45.12.4.12 Clock enable for CDSP

- No action
- Clock enable for CDSP instance

ADC_CLKEN_CDSP Offset address: $0089C_H$

Clock enable for CDSP Kernel Reset value: $0000\ 0000_H$



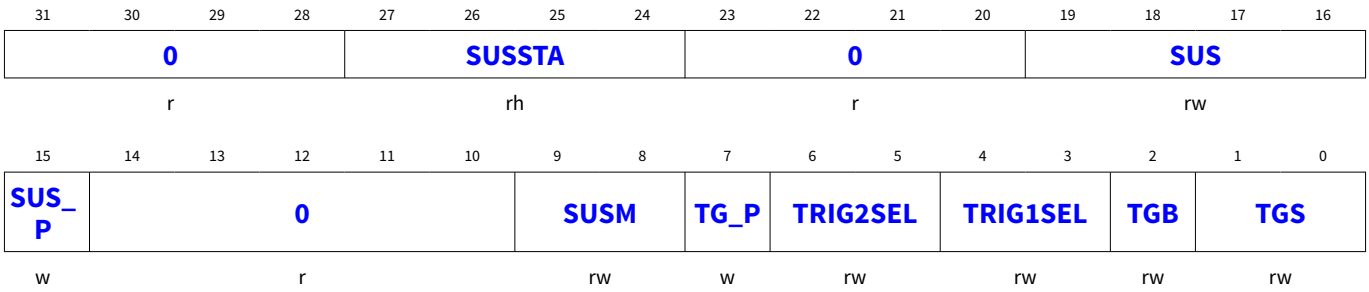
Field	Bits	Type	Description
CDSPEN	1:0	rwh	**Clock enable for corressponding CDSP** When set to '1' clock is enabled for the corresponding CDSP
CDSPGL	31	rwh	**Enables the global part of CDSP** If the user intends to use any CDSP, then this bit shall be set to 1
0	30:2	r	**Reserved** Read as 0; should be written with 0

45.12.4.13 OCDS control and status register for TMADC

This OCDS control and status register controls the debug and trace behavior by selecting suspend modes and OTGB trigger sets. When OCDS is disabled the suspend control is ineffective. This register only controls all instances of TMADC

ADC_OCSTM Offset address: $008A8_H$

OCDS control and status register for TMADC Debug Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
TGS	1:0	rw	**Trigger set for OTGB0 and OTGB1** 00_B No trigger set output 01_B Trigger set 1 Start of conversion trigger signal of TMADC input channels. For selecting the right TMADC, refer to TRIG1SEL 10_B Trigger set 2 Start of conversion trigger signal of TMADC input channels. For selecting the right TMADC, refer to TRIG2SEL 11_B Reserved

(table continues...)

(continued)

Field	Bits	Type	Description
TGB	2	rw	**OTGB0 and OTGB1 bus select** 0_B Trigger set is output on OTGB0 1_B Trigger set is output on OTGB1
TRIG1SEL	4:3	rw	**Trigger set 1 select** These bits select which TMADC instance trigger signals are put on trigger set 1 00_B TMADC1 01_B TMADC3 10_B TMADC5 11_B TMADC7
TRIG2SEL	6:5	rw	**Trigger set 2 select** These bits select which TMADC instance trigger signals are put on trigger set 2 00_B TMADC0 01_B TMADC2 10_B TMADC4 11_B TMADC6
TG_P	7	w	**TGS, TGB write protection** TGS,TGB TRIG1SEL and TRIG2SEL are only written when TG_P is 1, otherwise unchanged. Read as 0.
SUSM	9:8	rw	**OCDS suspend control mode - SUS** These bits select the suspend mode for the module 00_B Will not suspend 01_B Hard suspend Clocks will be disabled immediately 10_B Reserved, not to be used Reserved, will not suspend 11_B Soft suspend mode All the channels in hold state for the TMADC will be converted and then module will go into suspend
SUS_P	15	w	**SUS write protection** SUS and SUSM are only written when SUS_P is 1, otherwise unchanged. Read as 0.
SUS	19:16	rw	**OCDS suspend control** Suspends the corresponding TMADC. 0 - Module not suspended, 1 - Module Suspended. Each bit controls a TMADC, with LSB for TMADC0 and MSB for the maximum number of TMADC in device.
SUSSTA	27:24	rh	**Suspend state** Suspend State of TMADC. 0 - Module not suspended. 1 - Module Suspended. Each bit gives status of a TMADC, with LSB for TMADC0 and MSB for the maximum number of TMADC in device.
0	14:10, 23:20, 31:28	r	**Reserved** Read as 0; should be written with 0.

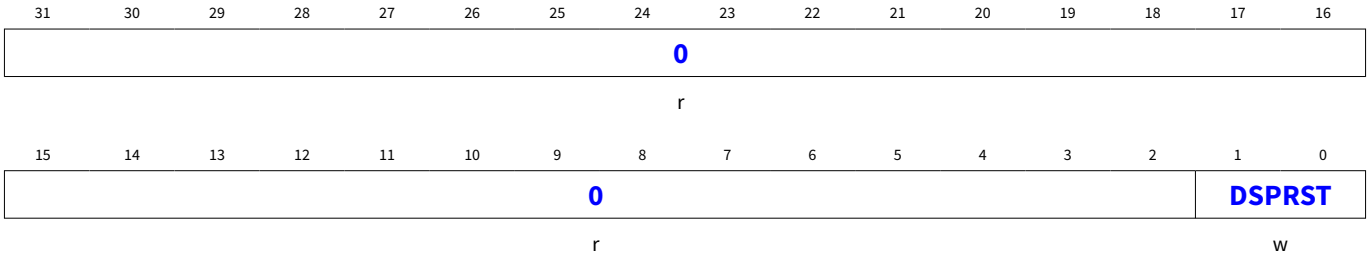
Table 2417 Access mode restrictions of ADC_OCSTM sorted by descending priority

Mode name	Access mode		Description
write 1 to .**TG_P**	rw	TGB, TGS, TRIG1SEL, TRIG2SEL	Set TG_P during write access
write 1 to .**SUS_P**	rw	SUS, SUSM	Set SUS_P during write access
(default)	r	SUS, SUSM, TGB, TGS, TRIG1SEL, TRIG2SEL	

45.12.4.14 CDSP global reset control

ADC_CDSP_GLOBRD Offset address: $BF804_H$

CDSP global reset control Kernel Reset value: $0000\ 0000_H$

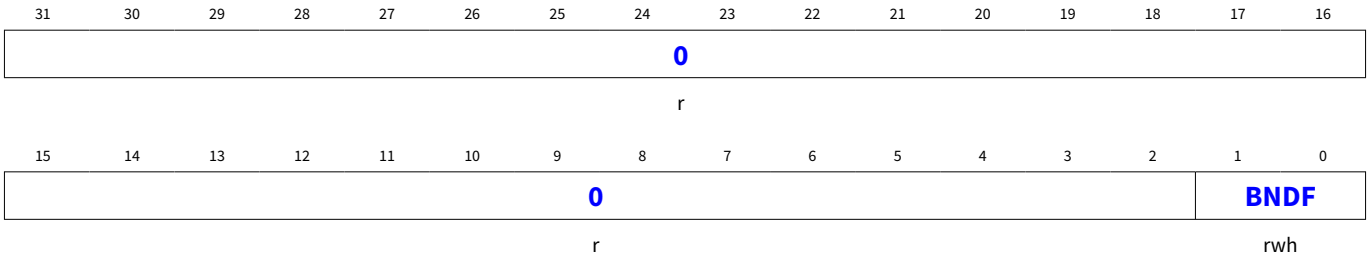


Field	Bits	Type	Description
DSPRST	1:0	w	**DSP reset** Each bit resets the corresponding CDSP instance. For example bit 0 of this register resets CDSP0.
0	31:2	r	**Reserved** Read as 0; should be written with 0

45.12.4.15 CDSP boudary flag event register

ADC_CDSP_BNDCF Offset address: $BF808_H$

CDSP boudary flag event register Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
BNDF	1:0	rwh	**Boundary event for CDSP channel** Status of the boundary flag event of the corresponding CDSP instance. For example bit 0 of this register indicates the status of boundary flag event of CDSP0

(table continues...)

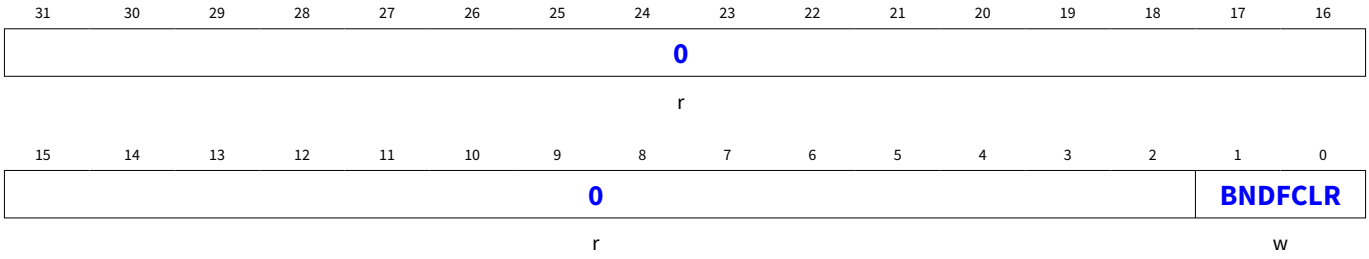
(continued)

Field	Bits	Type	Description
0	31:2	r	**Reserved** Read as 0; should be written with 0

45.12.4.16 CDSP boundary flag event clear

ADC_CDSP_BNDCFCL Offset address: $BF80C_H$

CDSP boundary flag event clear Kernel Reset value: $0000\ 0000_H$

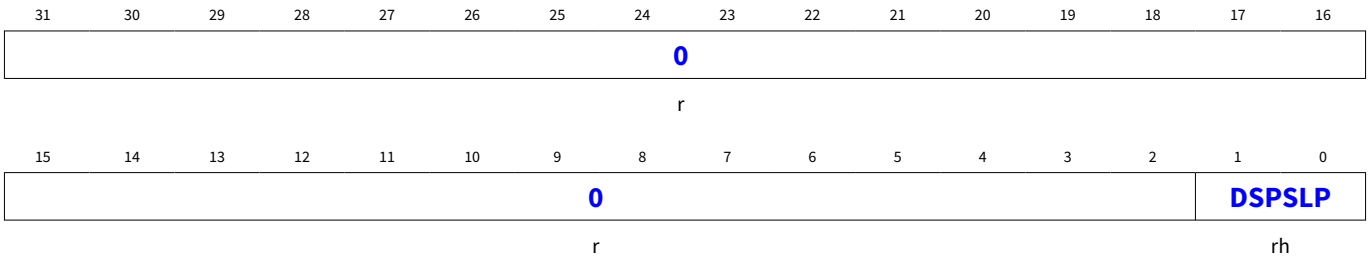


Field	Bits	Type	Description
BNDFCLR	1:0	w	**Boundary event clear for CDSP channels** Setting bits clears the bouandy event of the corresponding CDSP instance. For example bit 0 of this register clears the boundary event of the CDSP0.
0	31:2	r	**Reserved** Read as 0; should be written with 0

45.12.4.17 CDSP sleep status register

ADC_CDSP_SLPST Offset address: $BF810_H$

CDSP sleep status register Kernel Reset value: $0000\ 0000_H$

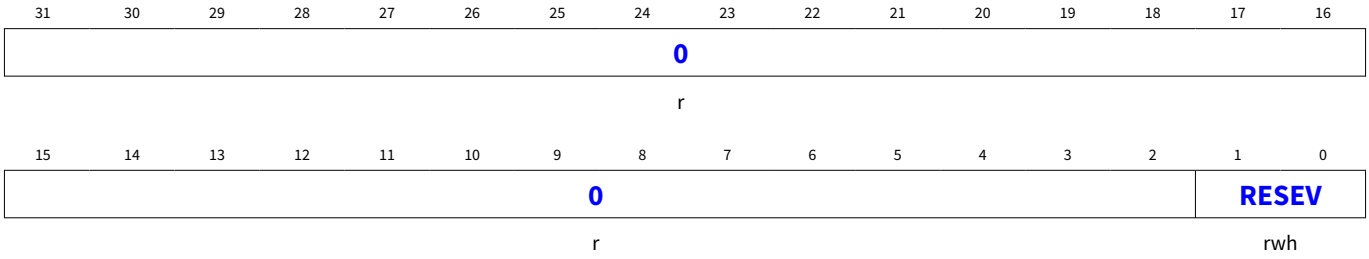


Field	Bits	Type	Description
DSPSLP	1:0	rh	**CDSP sleep status** Sleep status of the DSP core of the corresponding CDSP instance. For example bit 0 of this register indicates the sleep status of the DSP core of the CDSP0.
0	31:2	r	**Reserved** Read as 0; should be written with 0

45.12.4.18 CDSP event flag register y

ADC_CDSP_RESEVy (y=0-2) Offset address: $BF814_H+y*4$

CDSP event flag register y Kernel Reset value: $0000\ 0000_H$

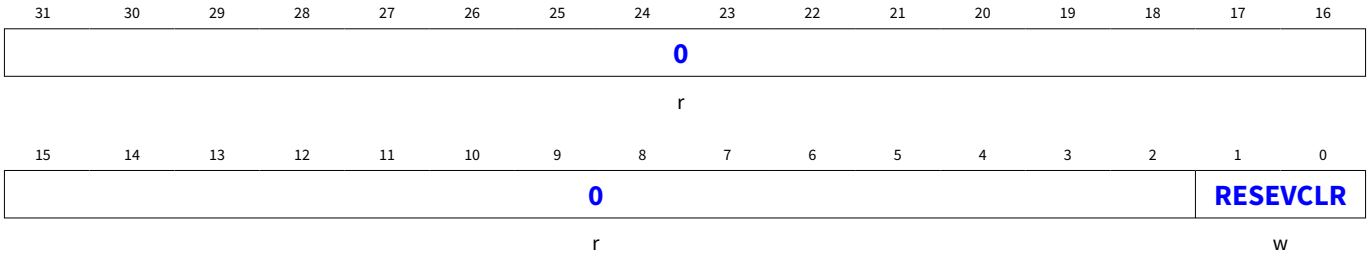


Field	Bits	Type	Description
RESEV	1:0	rwh	**Result event for result register y of CDSP** Status of the result event of the corresponding CDSP instance. For example bit 0 of this register indicates an result event of CDSP0. Reading a result register (CDSP_DSPa_RESy) automatically clears the respective result event in CDSP_RESEVy.
0	31:2	r	**Reserved** Read as 0; should be written with 0

45.12.4.19 CDSP result event flag clear y

ADC_CDSP_RESEVCLRy (y=0-2) Offset address: $BF820_H+y*4$

CDSP result event flag clear y Kernel Reset value: $0000\ 0000_H$

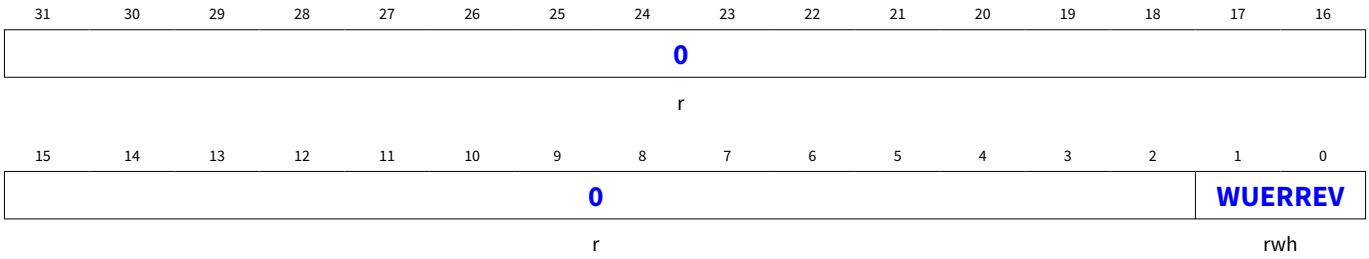


Field	Bits	Type	Description
RESEVCLR	1:0	w	**Result event clear for result register y of CDSP** Each bit clears the result event of the corresponding CDSP instance. For example setting bit 0 of this register clears the result event of the CDSP0.
0	31:2	r	**Reserved** Read as 0; should be written with 0

45.12.4.20 CDSP wake up error flag register

ADC_CDSP_WUERREV Offset address: $BF8D0_H$

CDSP wake up error flag register Kernel Reset value: $0000\ 0000_H$

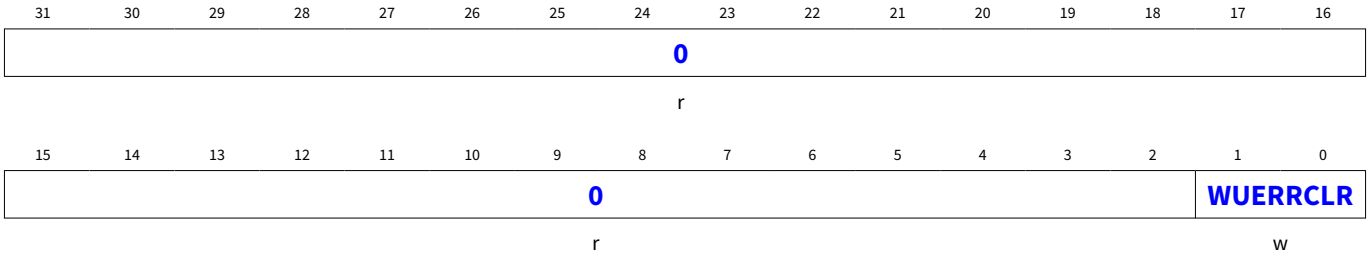


Field	Bits	Type	Description
WUERREV	1:0	rwh	**Wake up error event** Status of the wake up error event of the corresponding CDSP instance. For example bit 0 of this register indicates the status of the wake up error event of CDSP0.
0	31:2	r	**Reserved** Read as 0; should be written with 0

45.12.4.21 CDSP wake up error event clear register

ADC_CDSP_WUERRCL Offset address: $BF8D4_H$

CDSP wake up error event clear register Kernel Reset value: $0000\ 0000_H$

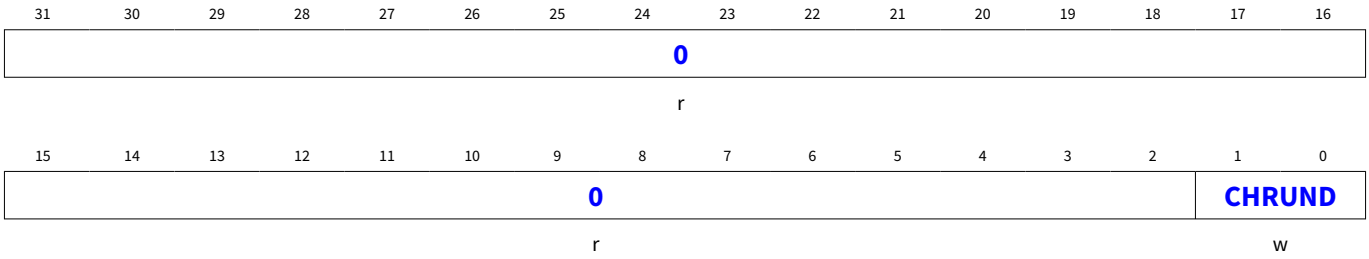


Field	Bits	Type	Description
WUERRCLR	1:0	w	**Wake up error event clear** Each bit clears the wake up error event of the DSP core of the corresponding CDSP instance. For example bit 0 of this register clears the wake up error event of the DSP core of the CDSP0.
0	31:2	r	**Reserved** Read as 0; should be written with 0

45.12.4.22 CDSP global run control register

ADC_CDSP_GLOBRCD Offset address: $BF8D8_H$

CDSP global run control register Kernel Reset value: $0000\ 0000_H$

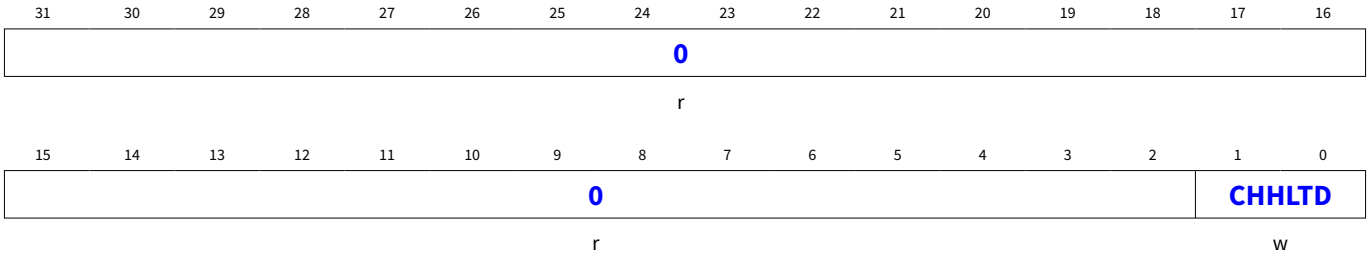


Field	Bits	Type	Description
CHRUND	1:0	w	**Channel run for CDSP** Each bit of this register runs the DSP core of corresponding CDSP and logic around it. By writing '1' the CDSP switches into Run state, i.e program execution starts.
0	31:2	r	**Reserved** Read as 0; should be written with 0

45.12.4.23 CDSP global halt control register

ADC_CDSP_GLOBHCD Offset address: $BF8DC_H$

CDSP global halt control register Kernel Reset value: $0000\ 0000_H$

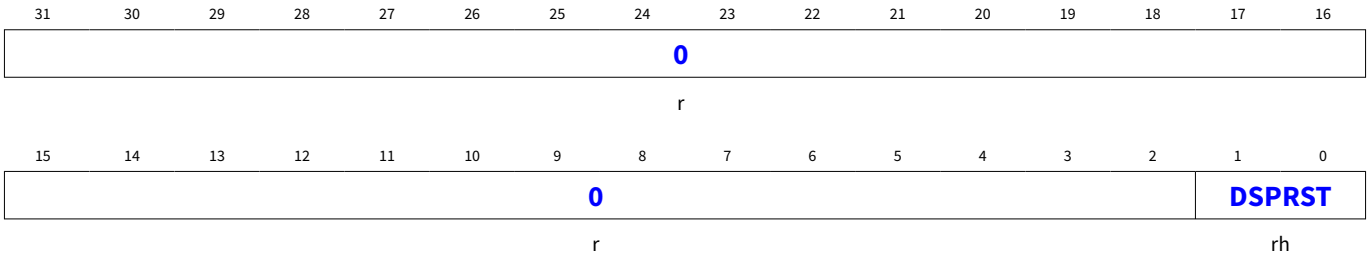


Field	Bits	Type	Description
CHHLTD	1:0	w	**Channel halt for DSP** Each bit of this register halts the DSP core in corresponding CDSP. Read the halt status register to check if DSP has been halted. This register is used, if DSP is halted from outside. For example bit 0 of this register controls the halt of CDSP0.
0	31:2	r	**Reserved** Read as 0; should be written with 0

45.12.4.24 CDSP global run status register

ADC_CDSP_GLOBRSD Offset address: $BF8E0_H$

CDSP global run status register Kernel Reset value: $0000\ 0000_H$

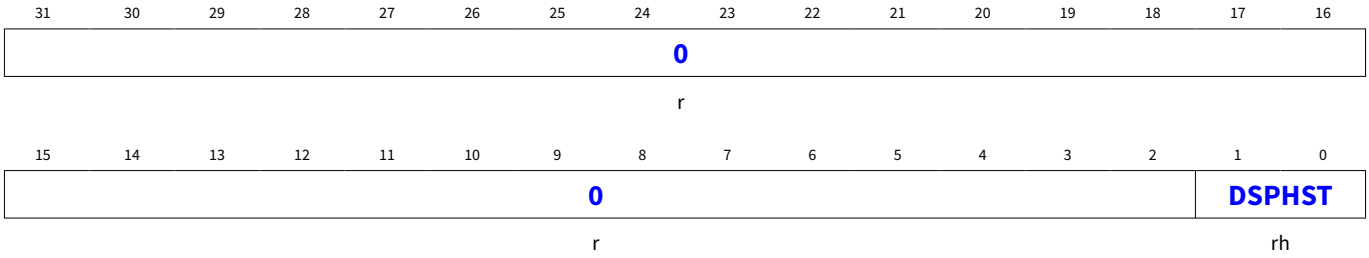


Field	Bits	Type	Description
DSPRST	1:0	rh	**DSP run status** Run status of the DSP core of corresponding CDSP and logic around it. For example bit 0 of this register indicates the Run status of CDSP0. Bit is kept high, until the core goes into Halt status.
0	31:2	r	**Reserved** Read as 0; should be written with 0

45.12.4.25 CDSP global halt status register

ADC_CDSP_GLOBHSTD Offset address: $BF8E4_H$

CDSP global halt status register Kernel Reset value: $0003\ FFFF_H$

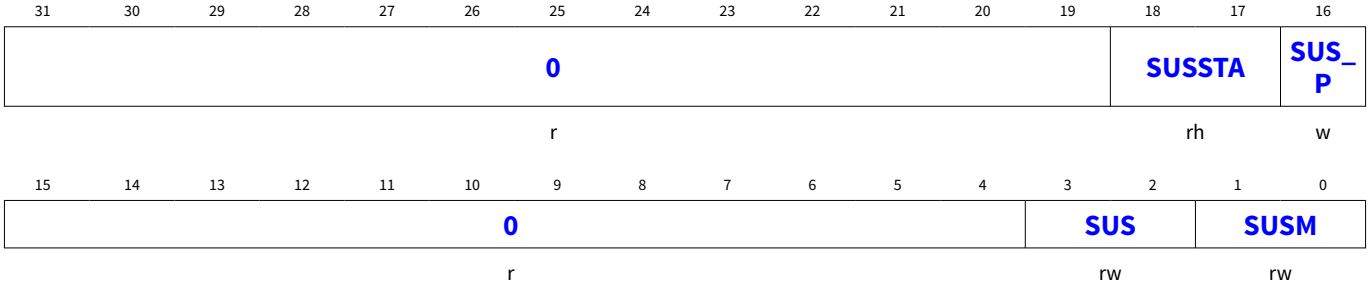


Field	Bits	Type	Description
DSPHST	1:0	rh	**DSP halt status** Halt status of the DSP core of corresponding CDSP and logic around it. For example bit 0 of this register indicates the Halt status of CDSP0. Bit is kept high, until the core goes into Run status.
0	31:2	r	**Reserved** Read as 0; should be written with 0

45.12.4.26 OCDS control and status register 0 for CDSP0-8

ADC_CDSP_OCSCDSPA Offset address: $BF8E8_H$

OCDS control and status register 0 for CDSP0-8 Debug Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
SUSM	1:0	rw	**OCDS suspend control mode** Selects the suspend mode for the module. *Note: For reaching soft suspend condition the selected input data source must send enough inputs to the CDSP for a result event to be generated.* 00_B Will not suspend 01_B Hard suspend Clocks will be disabled immediately 10_B Soft suspend mode Clock will be disabled after processing the current sample. 11_B Soft suspend mode Clock will be disabled after generating the next result event, visible in the corresponding bit of CDSP_RESEV0.
SUS	3:2	rw	**OCDS suspend control** Suspends the corresponding CDSP. 0 - Module not suspended, 1 - Module Suspended. Each bit control a CDSP, with LSB for CDSP0 and MSB for the CDSP8. *Note: Please check how many CDSP are there in device*
SUS_P	16	w	**SUS write protection** SUS and SUSM are only written when SUS_P is 1, otherwise unchanged. Read as 0.
SUSSTA	18:17	rh	**Suspend state** Suspend state of CDSP. 0 - Module not suspended. 1 - Module suspended. Each bit controls a CDSP, with LSB for CDSP0 and MSB for the CDSP8. *Note: Please check how many CDSP are there in device*
0	15:4, 31:19	r	**Reserved** Read as 0; should be written with 0.

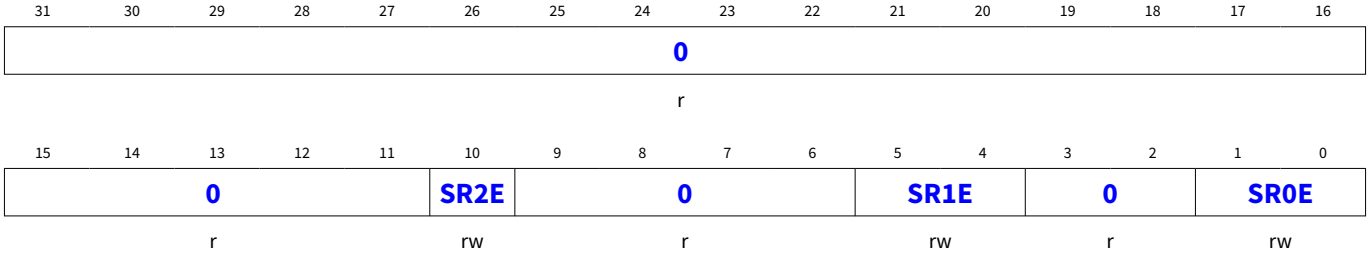
Table 2418 Access mode restrictions of ADC_CDSP_OCSCDSPA sorted by descending priority

Mode name	Access mode		Description
write 1 to .**SUS_P**	rw	SUS, SUSM	Set SUS_P during write access
(default)	r	SUS, SUSM	

45.12.4.27 CDSPa service request configuration register

ADC_CDSP_DSPa_SRCFG (a=0-1) Offset address: $C1C24_H+a*2000_H$

CDSPa service request configuration register Kernel Reset value: $0000\ 0000_H$

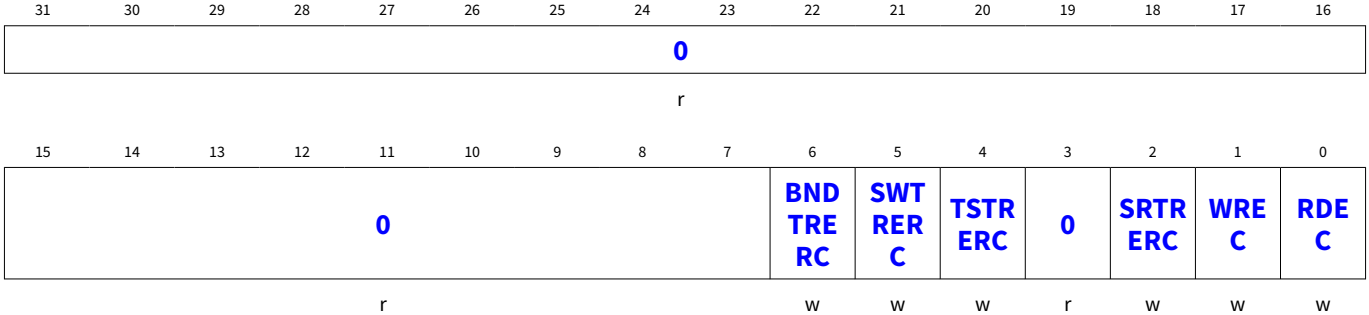


Field	Bits	Type	Description
SR0E	1:0	rw	**Conditions under which SR0 shall be generated** 00_B Never 01_B Generate Service Request only while gate is high 10_B Generate Service Request only while gate is low 11_B Always
SR1E	5:4	rw	**Service request 1 enable** Enable or disable for the service request 1, SR1 is generated at every time stamp generation if enabled 00_B SR1 disabled 01_B SR1 is generated everytime a new timstamp is generated 10_B SR1 is generated when a wakeup error occurs or ARC core is in halt state 11_B SR1 is generated on timestamp generation, wakeup error event or system halt
SR2E	10	rw	**Service request 2 enable** Enable or disable for service request 2, SR2 is generated based on the configured boundary check through SR02CFG in CDSP_DSPa_BNDCFG register
0	3:2, 9:6, 31:11	r	**Reserved** Read as 0; should be written with 0

45.12.4.28 CDSPa channel error clear register

ADC_CDSP_DSPa_CHERRCL (a=0-1) Offset address: $C1C34_H+a*2000_H$

CDSPa channel error clear register Kernel Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
RDEC	0	w	**Read error flag clear** 0_B No action 1_B Clear flag RDERR
WREC	1	w	**Write error flag clear** 0_B No action 1_B Clear flag WRERR
SRTRERC	2	w	**Service request trigger error clear**
TSTRERC	4	w	**Time-stamp trigger error clear**
SWTRERC	5	w	**Software trigger error clear** 0_B No action 1_B Clear flag SWTRER
BNDTRERC	6	w	**Boundary request trigger error clear**
0	3, 31:7	r	**Reserved** Read as 0; should be written with 0

45.12.4.29 CDSPa status register

ADC_CDSP_DSPa_DSPST (a=0-1) Offset address: $C1C38_H+a*2000_H$

CDSPa status register Kernel Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		BNDTRER	SRTRER	0	TSTRER	SWTRER	WRERR	RDERR	0						
r		rh	rh	r	rh	rh	rh	rh	r						

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0														FILL	
r														rh	

Field	Bits	Type	Description
FILL	2:0	rh	**FIFO fill level** Not listed combinations are reserved. 000_B Result FIFO is empty 001_B Result FIFO contains 1 valid value 010_B Result FIFO contains 2 valid values 011_B Result FIFO contains 3 valid values 100_B Result FIFO contains 4 valid values

(table continues...)

(continued)

Field	Bits	Type	Description
RDERR	23	rh	**Read error flag** 0_B No problem encountered 1_B A read access occurred while the FIFO was empty A read error is also indicated when a read access occurs during the FIFO's synchronization stall phase (4 fADC after a read access). Clear this sticky flag by writing 1 to bit RDEC.
WRERR	24	rh	**Write error flag** 0_B No problem encountered 1_B A write access occurred while the FIFO was full Clear this sticky flag by writing 1 to bit WREC.
SWTRER	25	rh	**Software trigger error flag** This bit tells the status of the software trigger error flag. If the software trigger could not be executed the bit will be set to 1
TSTRER	26	rh	**Timestamp trigger error flag** This bit tells the status of the timestamp trigger error flag. If the timestamp trigger could not be executed the bit will be set to 1
SRTRER	28	rh	**Service request trigger error flag** This bit tells the status of the service request trigger error flag. If the service request trigger could not be executed the bit will be set to 1
BNDTRER	29	rh	**Boundary update trigger error flag** This bit tells the status of the boundary update trigger error flag. If the boundary update trigger could not be executed the bit will be set to 1
0	22:3, 27, 31:30	r	**Reserved** Read as 0; should be written with 0

45.12.5 TC4Dx ADC connectivity

The table below lists interfaces to and from this functional block to other blocks in the device.

Table 2419 List of ADC interface signals

Interface signals	I/O	Description
LLI_ADC_AXI32S	In	LLI Slave
CLOCK_ADC_fADC	In	ADC Clock
ADC_EGTM_CLUSTERx_BFLY[3:0]x=[2:0]	Out	Each eGTM Cluster with TIM gets 4 flags to be used in TIM and DTM
ADC_PORTS_BFL[9:0]	Out	Connecting FCC BFL to PORTS
ADC_PORTS_EMUXCTRL[5:0]	Out	TMADC EMUX0 & EMUX 1 CTRL signals
EGTM_ADC_TRIG[11:0]	In	Each GTM cluster with TOM & ATOM generated 4 trigger and cluster with ATOM only 2 trigger signals
SCU_ADC_ERU[7:0]	In	8 trigger signals coming from ERU

(table continues...)

Table 2419 (continued) List of ADC interface signals

Interface signals	I/O	Description
PORTS_ADC_TRIG[62:56]	In	8 trigger signals coming from PORTS
PORTS_ADC_DATA[7:0]	In	Data input from PORTS to EXMOD
PORTS_ADC_CLK[7:0]	In	Clock input from PORTS to EXMOD
ADC_PORTS_CLK[3:0]	Out	Clock outputs from EXMOD to PORTS
ADC_PORTS_PWM[1:0]	Out	Carrier Generator Output
PORTS_ADC_SIGN[1:0]	In	Sign input Carrier Signal
ADC_IR_SRC_ADCSR[3:0]	Out	Global Service request from ADC to IR
ADC_IR_SRC_ADCSR[11:4]	Out	Alarms also routed as SR
ADC_IR_SRC_TMADCxSR[5:0]x=[6:0]	Out	TMADC Service request to IR
ADC_EGTM_SRC_TMADCxSR[5:0]x=[6:0]	Out	TMADC Service request to eGTM
ADC_IR_SRC_FCCxSR[1:0]x=[9:0]	Out	FCC Service reuqest to IR
ADC_IR_SRC_DSADCxSR[3:0]x=[9:0]	Out	DSADC Service request to IR
ADC_IR_SRC_EXMODxSR[3:0]x=[7:0]	Out	External Modulator Service request to IR
ADC_IR_SRC_CDSPxSR[2:0]x=[11:0]	Out	CDSP Service request to IR
ADC_SMU_ALARM[7:0]	Out	ADC Alarms to SMU
ADC_EGTM_SR[39:0]	Out	EGTM control does collect ADC SRs

45.12.6 TC4Dx ADC revision history

Reference	Description of change(s)
Date range: 2025-03-14 to 2025-04-11	
TC4Dx ADC functional description	• Added CDSP limit checking and note pointing to non-existence for TC4Dx
Date range: 2024-08-17 to 2025-03-13	
TC4Dx ADC connectivity	• PORTS_ADC_TRIG[62:56]: corrected the description from 8 to 7 trigger signals

46 Debug and Trace

The Debug and Trace functions support the developer in analyzing issues on both a software and system level.

The main Debug and Trace system features are:

- Tool access to debug resources and all other address mapped registers and memories
- Routing of triggers for synchronized start or stop of CPUs and peripherals, and to and from trigger pins
- Parallel multi-core tracing to on-chip buffers or off-chip to dedicated (SGBT, DAP) or functionally shared interfaces (GETH)

***Note**: The on-chip multi-tool system (OCMTS) is composed of Debug and Trace modules. If both modules are involved in a specific topic, then they are also referred to as simply as 'OCMTS'.*

46.1 Feature list

The OCMTS sub-system consists of intrusive Debug and non-intrusive Trace functionality. They offer the following capabilities:

Debug

- Tool access through two Device Access Ports (DAP, DAPE) or JTAG
- Enable tool access to debug and functional resources
- Safe and secure debug function handling
- Run control of embedded processor cores and suspend control of peripherals
- Internal and external trigger action and reaction
- Debug event routing internally and to the debug host
- Stress and fault injection
- Control the debug modes regarding power saving, clock and reset control

Trace

- Supports applications such as trace based measurement or software development (debugging and performance analysis)
- Six operating modes provided: three on-chip memory and three streaming trace options
- Three kinds of trace interfaces available: Device Access Port (DAP), Ethernet (with software support), Serial GigaBit Trace (SGBT)
- Selectable program and data trace sources: all CPU(s) in parallel, PPU, GTM, DMA(s), OLDA, SPB, OTGB (signals of complex peripherals like EVR state or interrupt requests)
- Common programming model for single Multi-Core Debug Solution (MCDS) and dual MCDS configurations
- Common Multi-Trace Stream Container (MTSC) format to simplify trace message decoder development
- Program trace (instructions, program flow, functions only) support for virtualized TriCores™
- Very efficient support for software based data trace filtering
- Pre and post-event trace buffering (“digital oscilloscope”)
- Sophisticated trace qualification or filtering, and optimized trace message compression
- Central time stamp to correlate traces from different trace sources
- Robust recovery from trace buffer memory overflow
- Trace functionality can be enabled independently from debug functionality
- SGBT interface is compliant to the Aurora protocol and can be configured to operate at baud rates of 6.25, 5 or 2.5 Gbps (including 8B/10B encoding) with an available trace bandwidth of 4.816 Gbps at 6.25 Gbps

  Example for bandwidth calculations: 4.816 Gbps = 6.25 Gbps x 8B/10B encoding x (CC efficiency := 9992/10000) x (SGBT frame efficiency := 4096/4116) x (MTSC efficiency := 31/32)

46 Debug and Trace

- Smooth migration path from AURIX™ 2nd Generation for tool vendors and users
- Highly compatible architecture within the AURIX™ TC4xx family from low-end to high-end

46.2 Functional overview

The two main OCMTS functions, Debug and Trace, are implemented as separate entities in the device. They may be used independently or in cooperative mode.

The data flow and connectivity between the two functions and the rest of the device is shown in the following figure:

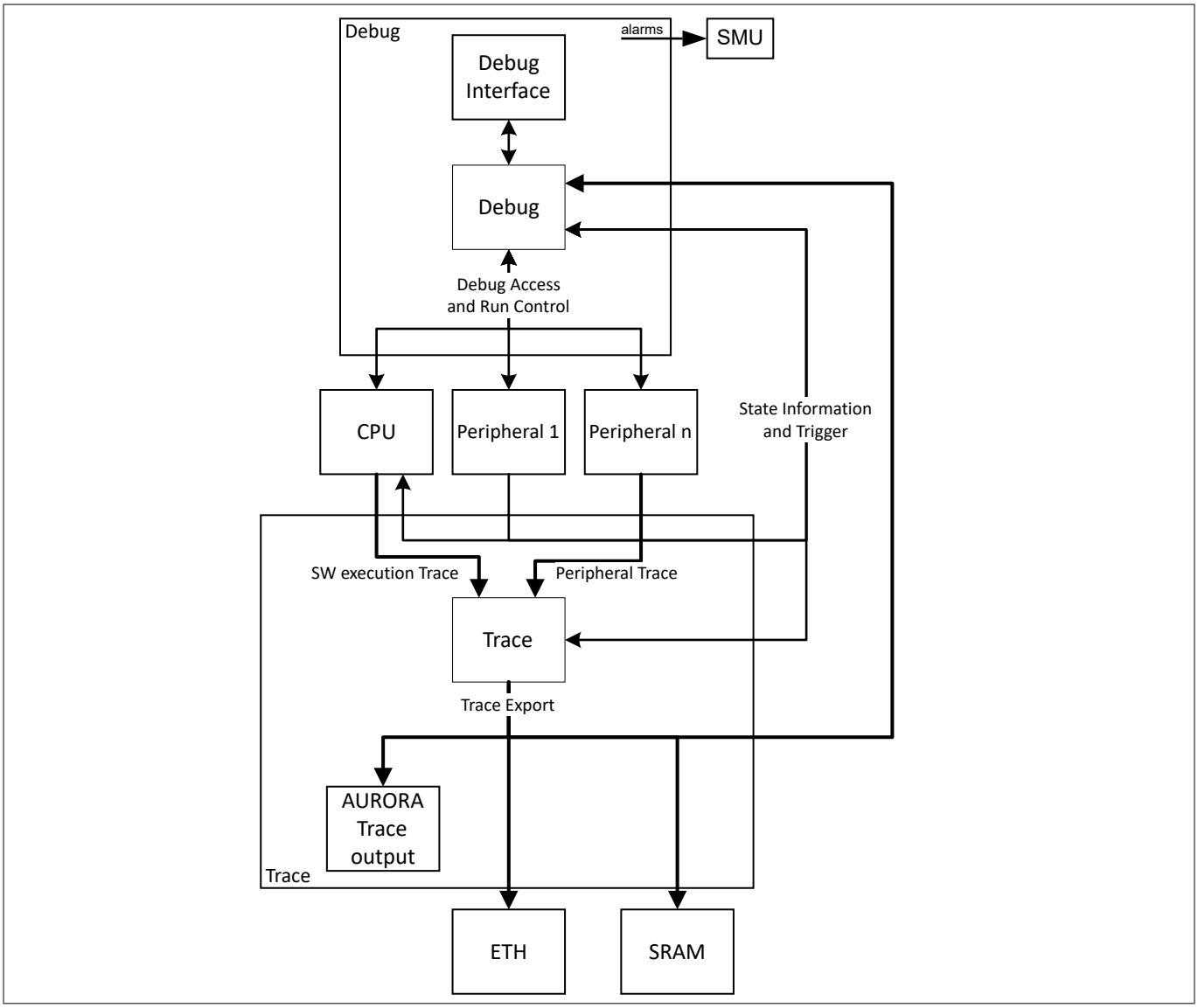


Figure 1244 Debug and Trace block diagram

For the alarm description please refer to the alarm mapping tables in the SMU functional block and the Safety Manual.

The Debug block includes interfaces which are used to interact with the devices external tooling. The tool driven communication reaches the central Debug block by using an IOClient mechanism. The Debug block includes the necessary functionality to interact with the system over in-band bus accesses. Additional dedicated connections allow intrusive control of the program execution in processing cores, influence peripherals based on debug needs, and capture signal states over the triggering system.

The Trace block captures system execution flows in processing cores, peripherals, and the system interconnect. Dedicated connections to these blocks transport the data. The captured data items are processed, converted to a common trace format, and exported to dedicated trace ports or shared capabilities such as SRAM or network connections.

Related information

TC4Dx SMU alarm mapping tables on page 7233

46.3 Functional description

The following figure shows the modular split of the Debug and Trace sub-functions.

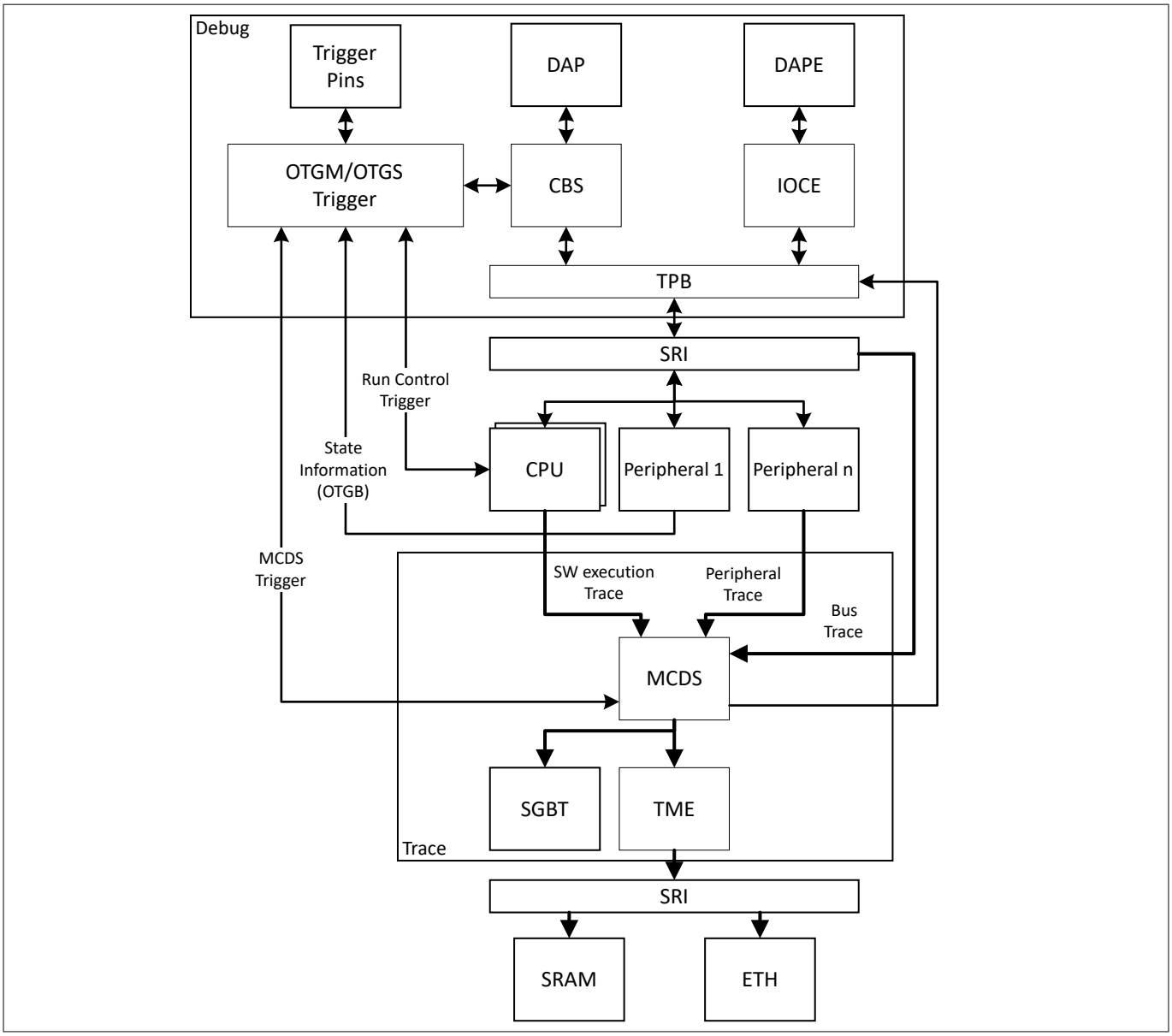


Figure 1245 Module connectivity in the Debug and Trace system

Debug function mapping

Debug functions are mapped on to the following modules:

Debug Interfaces

- Debug access port (DAP/DAPE):
  - 2 to 4 wire debug tool connection.

- Both DAP interfaces may be used in parallel by 2 mostly independent tool connections.
- ***Note**:* *IEEE1149.1 based JTAG access is also supported. However the preferred, more robust, and higher performance method, is to use DAP based connectivity.*

- Trigger Pins:
  - Up to 8 Trigger in and output pins.

Debug system access and control

- Cerberus (CBS) :
  - The main Debug enable, control, and system access block.
  - Directly accessed by the DAP interface.
  - Injector for fault and stress test (IFS).
- IOCE secondary system access:
  - Used by the DAPE interface.

Trigger functions

- OCDS trigger switch (OTGS):
  - Run control and trigger distribution.
- OCDS trigger multiplexer (OTGM):
  - Peripheral trigger collection and forwarding.

Trace function mapping

Trace functions are mapped on to the following modules:

Multi-core debug solution (MCDS):

- Trace data collection from trace sources.
- Filter based on user definable parameters.
- Convert to multi-trace stream container (MTSC) trace format.
- Storage towards user defined DAP, TME, or SGBT path.

Trace move engine (TME):

- Trace data export to system address space:
  - Store towards SRAM.
  - Export on shared device functions such as GETH.

Serial gigabit trace (SGBT)

- AURORA based differential Trace export to pins.

46.3.1 Debug access

Access to the device Debug system can be implemented over several methods, but usually DAP will be used.

Configuration and data retrieval using DAP

A debug tool may use any of the DAP modules to access the system and, assuming access is granted from a security perspective, may configure any of the OCMTS functions over the TPB bus.

***Note**:* *The number of debug access functions differs between DAP and DAPE. DAP may be the first choice for debug centric tooling while the DAPE function space may fit better for pure trace or calibration tools.*

The same path may be used to access the bridge towards the system wide Shared Resource Interconnect (SRI). This link allows a debug tool to configure the CPU, or peripheral local debug and trace features.

Software supported debug

By adding debug monitor software on a CPU, and by using the TPB register access to CBS, most of the debug functions can be made available over GETH for example.

46.3.2 Trace export

Captured and processed Trace data may be exported over several paths: Using DAP, forwarded to GETH, or using SGBT

Using DAP

Data remains stored in MCDS or is forwarded by TME to on-chip SRAM. The tool uses DAP to read the data.

This method does not require any connection other than the tool link. Limitations are the DAP transfer speed and the available storage space in MCDS or SRAM.

Forwarded to GETH

Captured trace data is stored on the device SRAM by TME, and forwarded to GETH.

This path uses the SRAM on the device as a buffer and exports the data with minimal software overhead over an ethernet connection. The CPU needs to handle the interrupt from the Trace sub-system and set up the DMA transfer towards the GETH module.

Using SGBT

Data is directly exported to pins using the SGBT module.

This path offers the highest performance and it is non-intrusive. The limitations of using this method are that a dedicated tool connection with specialized tooling is required, and HSPHY must not be used by other high speed IO.

46.3.3 Tool interface options

The AURIX™ TC4xx offers several options for connecting a tool. The one you choose depends on the target application, but it must be considered very early in the application design so that you do not accidentally block required interface pins or impact startup requirements.

46.3.3.1 Tool interface overview and enabling

This section focuses on dedicated tool interfaces which support controlled, flexible, read and write access. This does not include shared interfaces like Ethernet or the SGBT trace interface.

The options for connecting tools which need read and write access are:

- JTAG
- DAP (Device access port)
- DXCPL (DAP over CAN physical layer)
- DXCM (DAP over CAN messages)
- DAPE (second DAP)

JTAG is available for standard JTAG boundary scan; otherwise, it is always a second choice compared to DAP because of performance, pin count, error detection, and robustness.

The other interface options are optimized for different use cases and provide maximum compatibility due to their design based on DAP.

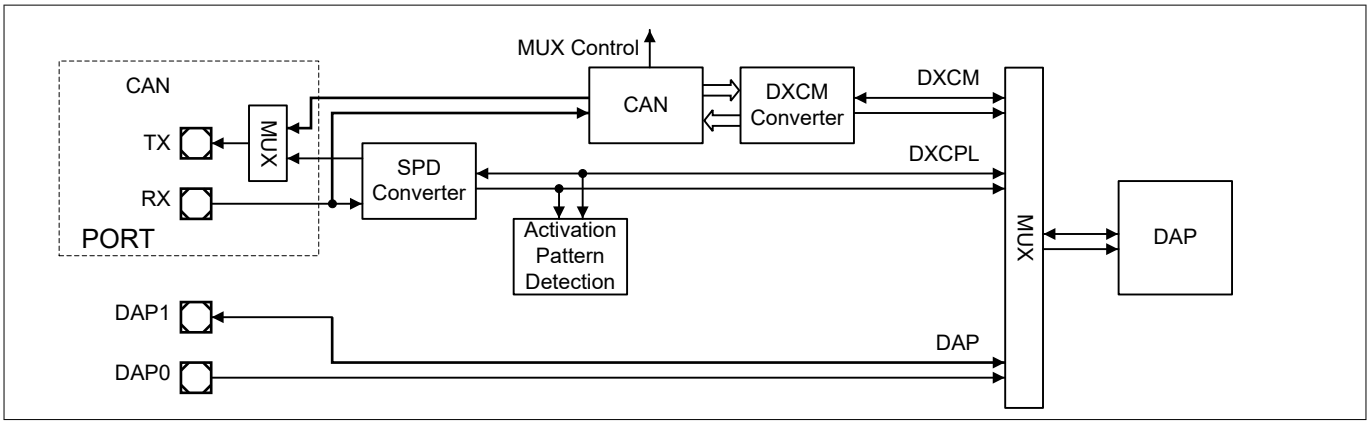


Figure 1246 DAP, DXCPL and DXCM block diagram

The CAN bus signals are usually available at the ECU connector, so this physical connection allows for the connection of all kinds of tools without opening the ECU housing.

The main use case of DXCPL is the analysis of field returns. The DXCPL standard pins are the CAN bootstrap loader pins . Another DXCPL use case is a tool interface for deeply embedded transmission applications. For this use case DXCPL is also available on other pins.

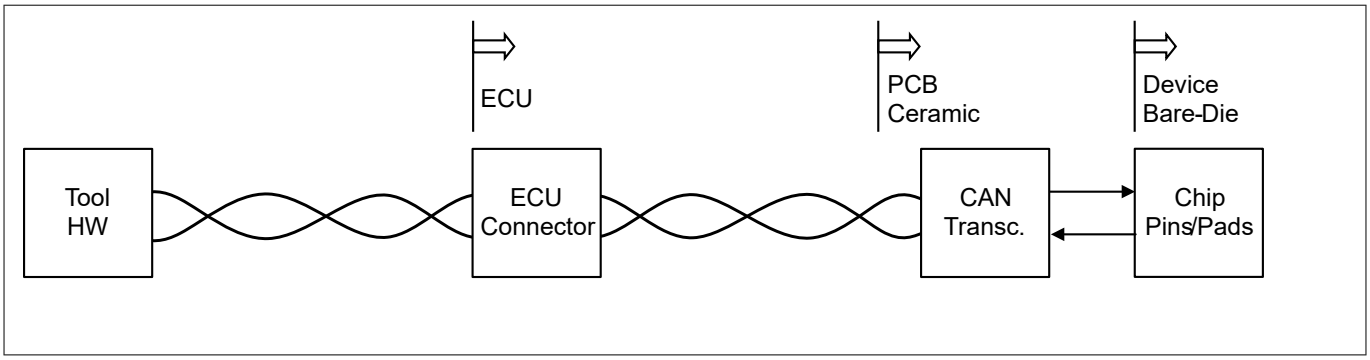


Figure 1247 DXCPL use case transmission ECU

DXCM transmits DAP telegrams and responses as the payload of regular CAN messages. This allows tool access while the system is using CAN. DXCPL is faster and more robust, but CAN cannot be used in parallel.

46.3.3.1.1 Comparison of DAP, DXCPL, and DXCM

The main difference between DXCM and DXCPL is that DXCM shares the CAN bandwidth with the regular CAN bus traffic. The following table shows a comparison between DAP, DXCPL, and DXCM.

Table 2420 Comparison of DAP, DXCPL, and DXCM

Property	DAP	DXCPL	DXCM
Performance	15(30) MB/s	200-800 KB/s	20-40 KB/s
Interface	2(3) DAP pins	CAN pins (for example of ECU connector)	
User resources	None (port function of DAP2 pin)	CAN pins	CAN bandwidth Two CAN IDs Two CAN MOs
Halt after PORST	Yes	Yes for default DXCPL pins.	No
Device test	Structural device test possible if security enabled.		No
Boundary scan	Yes		No
Security	Same security level and operation for all interfaces.		
Robustness against device failures	Reference	Only EVR oscillator as clock source needed.	Flash and clock system have to be functional.
Robustness against operation failures	Setup of physical connection and interface mode selection required (proper board, cable and tool configuration).		Additional matching CAN configuration.

46.3.3.1.2 Enabling of JTAG, DAP, DXCPL, or DXCM

The interface selection has several steps for addressing the different use cases in a robust and flexible way. The following table lists the main decision points for the tool interface enabling. More details about the conditions and transitions are available in the sections about the different interfaces.

Table 2421 Decision points for tool interface enabling

Point	Decision	Interface	Condition	Disabled Tool Interfaces
(1)[1]	JTAG	JTAG	$\overline{\text{TRST}}$ low at $\overline{\text{PORST}}$ release.	DAP, DXCPL, DXCM DAPE on JTAG pins
(2)[2]	Select one of: DAP, DAPE, DXCPL, DXCM.	DAP	Enabled by default, but needs DAPISC telegram to be activated.	DXCPL, DXCM (DAP turn-off to JTAG possible)
		DXCPL	DXCPL activation pattern detected on CAN RX.	JTAG, DAP, DXCM
		DXCM	DXCM is enabled with bit CAN0_MCR.DXCM.	JTAG, DAP
		DAPE	Enabled by default, but needs DAPISC telegram to be activated.	JTAG
(3)[3]	DAP or DAPE pin conflicts	DAP DAPE	DAP has priority when both tools claim the same pin with their private DAPISC.MODE .	JTAG DAPE when DAP claims DAP3 pin DAPE wide mode when DAP claims DAP2

1) Decision point (1) occurs within the startup firmware after release.
2) For decision point (2), the slightly simplified rule is that the first claims the interface.
3) Decision point (3) is only relevant when the DAPE interface is used. See the link to 'DAP, JTAG and trigger pins'.

Related information

For decision point (2), see 'DAP, JTAG, and trigger pins' on page 8586

46.3.3.1.3 Comparison of DAP and DAPE

While sharing the same bus connectivity, there are still differences in the functionality between tool connections over DAP and DAPE. The following table shows what functionality is shared and what is different between DAP and DAPE.

Table 2422 DAP and DAPE comparison

Function	DAP	DAPE
Full Chip Access (RW-mode)	X	X
Halt After Reset	X	X
Chip State readback using IOINFO	X	X
Reset request over OJCONF	X	-
Device unlocking	X	-
Communication mode with Interrupt generation	X	-
JTAG infrastructure access	X	-
Triggered Transfers	X	-

***Notes**:*

1. *X = Function available*

2. *In RW-mode DAPE has the option to access CBS functions through the register interface*

46.3.3.2 DAP, JTAG, and trigger pins

Tool connection through dedicated pins provides the highest robustness and performance. 'Dedicated' means that the pins must not be used for the application if there is any possibility that a tool might be attached at some point. As the attachment of tools can come very late in the design cycle, during calibration for example, this constraint usually results in a total loss of any other functionality overlaid on these pins.

Table 2423 DAP, JTAG, Trigger and GPIO pin configurations

In this table the supported interface configurations are listed with their pin usage. The configuration is controlled with the OIFM and DAPISC registers. All pins listed always use TTL voltage input levels when used for JTAG or DAP. The pins P21.6 and P21.7 have standard port input behavior when configured as GPIO.

Package pin	Two-pin	Three-pin	Four-pin	Trigger	Five-pin		GPIO
DAP0/TCK				-		TCK	-
DAP1/TMS				-		TMS	-
TDO	-	DAP2	DAP2	TGI3/TGO3	DAP2	TDO	P21.7
TDI	-	-	DAP3	TGI2/TGO2	DAP3	TDI	P21.6
$\overline{\text{TRST}}$	Needs to be high at $\overline{\text{PORST}}$ release for enabling DAP.				DAP4	$\overline{\text{TRST}}$	-

Table 2424 DAP/JTAG interface pins' on-chip pull device setting

This table lists the on-chip pull device characteristics. It is possible to operate the DAP and the DAPE interfaces in parallel using the same "JTAG" pins. This table shows the options which are controlled with the OIFM register and the two DAPISC registers of DAP and DAPE.

Package Pin	PORST			normal mode/ GPIO
DAP0/TCK	Please refer to respective device datasheet.	down	down	please refer to datasheet
DAP1/TMS		down	down	please refer to datasheet
TDO		up/down*	tristate	up/down configurable
TDI		up/down*	down	up/down configurable
$\overline{\text{TRST}}$		up	up	please refer to datasheet

***Note**: The value of the marked entries(*) depend on the selected DAP mode- default is 2- pin mode using DAP0, DAP1 only.*

***Note**: When switching from two-pin to a wider mode, a pull-up resistor on TDO might disturb the generation of the start bit in the first response of the configuration message. The debug tool may detect this disturbance when it checks the success of the first message after the mode switch. The pull-up resistor did not corrupt the communication in the given setup if this message was properly acknowledged and executed by the device. The communication may have been disturbed in case the message does not result in a reaction of the device. This is avoidable by using a pull-down resistor on TDO.*

***Note**: OIFM.DAPMODE default selection of the five pin mode must be changed to a narrower pin mode to allow usage of TDI as Trigger pin.*

Table 2425 DAP/DAPE interface configurations on "JTAG" pins

Package pin	Two-pin	Three-pin	Three-pin E	Two-pin E
DAP0/TCK			-	-
DAP1/TMS			-	-
TDO	-	DAP2	DAPE2	
TDI	-	-	DAPE1	DAPE1
$\overline{\text{TRST}}$	-	-	DAPE0	DAPE0

***Note**: DAPE tool hardware needs to sense the $\overline{\text{PORST}}$ pin for ensuring $\overline{\text{TRST}}$ high when $\overline{\text{PORST}}$ becomes inactive. DAPE tool hardware shall keep DAPE0/$\overline{\text{TRST}}$ low when not used for clock pulses.*

The $\overline{\text{PORST}}$ sensing is required for unsolicited resets, such as a chip-internal EVR under-voltage reset pulling $\overline{\text{PORST}}$ low. If JTAG were enabled due to $\overline{\text{TRST}}$ low at $\overline{\text{PORST}}$ release, this state could only be left with another Power-on Reset. Uncontrolled effects due to an unintentionally enabled JTAG interface will not occur when the DAPE tool hardware operates DAPE0/$\overline{\text{TRST}}$ as requested in the note above, effectively keeping the JTAG state machine in reset state.

The trigger in or out signals controlled by the OTGS Trigger Switch can be mapped to different pins controlled by the TIPR and TOPR registers. The possible locations shown in the next table are standardized for the whole device family. The availability of the higher port number trigger pins depends on device and package type. Please refer to the connectivity table in the device specific chapter.

Table 2426 TGI and TGO trigger pins

Package pin	Trigger input	TIPR	Trigger output	TOPR
P20.0	$\overline{\text{TGI0}}$	PIN0	$\overline{\text{TGO0}}$	PIN0
P20.1	$\overline{\text{TGI1}}$	PIN1	$\overline{\text{TGO1}}$	PIN1
P21.6	$\overline{\text{TGI2}}$	PIN2	$\overline{\text{TGO2}}$	PIN2
P21.7	$\overline{\text{TGI3}}$	PIN3	$\overline{\text{TGO3}}$	PIN3
P32.6	$\overline{\text{TGI4}}$	PIN4	$\overline{\text{TGO4}}$	PIN4
P32.7	$\overline{\text{TGI5}}$	PIN5	$\overline{\text{TGO5}}$	PIN5
P33.14	$\overline{\text{TGI6}}$	PIN6	$\overline{\text{TGO6}}$	PIN6
P33.15	$\overline{\text{TGI7}}$	PIN7	$\overline{\text{TGO7}}$	PIN7

It can be seen that there are potential conflicts, but as a general rule the interface signals defined in the pin configurations table and the interface table take precedence; the TGO trigger out connection will only be enabled if a pin is not needed.

The TGI trigger input functionality, configured in the Trigger Switch register TIPR is independent of any other usage of the pin.

46.3.3.3 DAP over CAN physical layer (DXCPL)

The CAN bus physical layer is a bi-directional connection with differential encoding on a single pair of wires. An external CAN transceiver translates between the logical level of the TXD or RXD device pins and the electrically robust signaling of the CAN bus. This direction separated signaling with TXD or RXD can be used to transmit SPD (see related information) encoded DAP messages (shown in the block diagram below).

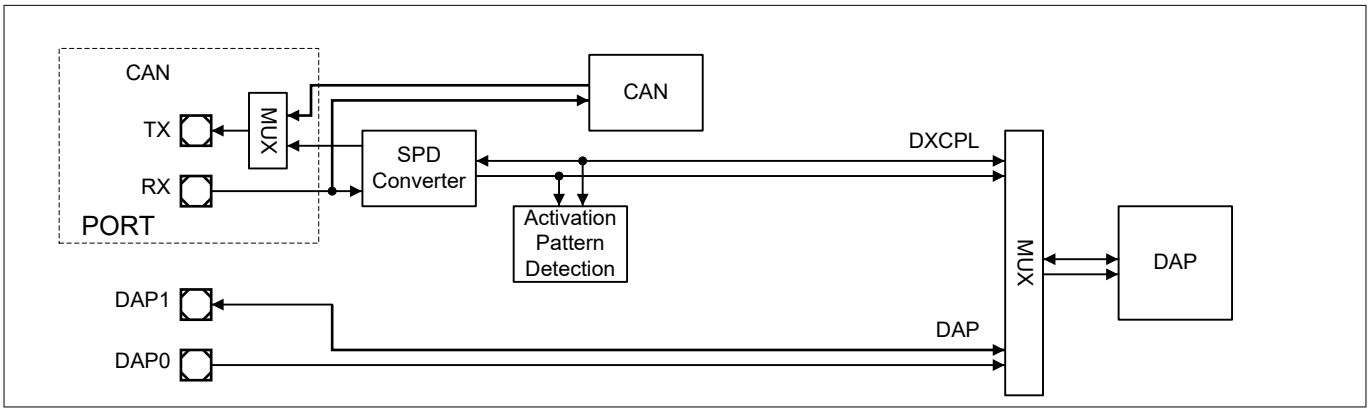


Figure 1248 DXCPL block diagram

The CAN bus signals are usually available at the ECU connector, so this physical connection allows you to connect all kinds of tools without opening the ECU housing.

DXCPL applications

- General debug when no other debug interface is available or accessible. After activation the OCDS infrastructure is accessible without restrictions. Test modes however are locked for safety and security reasons
- An electrically robust two wire measurement and calibration tool connection for deeply embedded applications such as transmission ECUs within the gear box

The other option for tool communication through CAN is DAP over CAN messages (DXCM).

Related information

Halt after reset request (HARR) on page 8592
CAN as a tool communication option; DAP over CAN messages (DXCM) on page 8594
Comparison of DAP, DXCPL, and DXCM on page 8584
Tool interface overview and enabling on page 8583
SPD protocol overview on page 8593
DXCPL SPD parameters on page 8591

46.3.3.3.1 Enabling and pins

DXCPL is by default enabled after , which means that it listens on the default RX pin for the activation sequence. Only after the activation sequence is sensed, including the DAPISC telegram, will the RX and TX pins be taken over exclusively for DXCPL.

***Notes**:*

1. *If DXCPL is just enabled but not yet activated, it has no impact on any device behavior. It just listens. Only when the DXCPL activation pattern and the DAPISC telegram (50 significant bits) are received, will the RX and TX pins be taken over. With the default settings (TRI_DXCCTR.TIC), this needs to happen in the time window between Power On Reset release and Flash start-up (~ 1 ms)*
2. *A prerequisite for DXCPL communication is an enabled CAN transceiver. This means that for the default TRI_DXCCTR.TIC) the DXCPL activation time window is already over when the user software enables the CAN transceiver. Only the other settings will allow DXCPL communication*
3. *Please refer also to the 'Exceptions for Emergency Stop' in the PORT configuration of the CAN pins used*

It is recommended to ensure that the CAN transceiver is enabled immediately after ; otherwise, it will not be possible to debug a range of situations using DXCPL. The robustness of DXCPL is then just on a similar level to DXCM including that no Halt after is possible.

Table 2427 DXCPL enabling and pins controlled by TRI_DXCCTR.TIC

TIC	Pins	Description
0_H	Default pins:	DXCPL will be disabled if not already activated by the tool after power-on reset.
1_H		DXCPL will stay enabled/activated for standard CAN pins.
2_H	Alternative pins: P33.12 (CAN00_RXDD) P33.13 (CAN00_TXD)	DXCPL is enabled for alternative CAN pins.

Related information

Comparison of DAP, DXCPL, and DXCM on page 8584
Halt after reset request (HARR) on page 8592
DXCPL activation sequence on page 8590

DXCPL activation sequence

DXCPL is only available when DAP is not used after Power-On Reset. The activation pattern is defined as 640 consecutive ‘0’-values in SPD format followed by a DAPISC telegram. The consecutive ‘0’-values mean permanent signal changes every 0.55 µs or less on the CAN bus.

***Note**: Depending on the situation (for example, an untrimmed EVR oscillator), the devices internal SPD decision point will be biased. In such a case a tool can try with a 20% shorter or longer edge distance. As soon as the device responds to the DAPISC telegram, the device internal biasing of the decision point can be determined and the tool can adapt accordingly for robust communication.*

The activation pattern will never occur inside any legal CAN traffic:

- The CAN protocol consists of specific frames (data-frames, remote-frames, and error-frames) with a maximum length of up to 128 bits, where the last seven "End-of-Frame" bits are transmitted as ‘1’. At least three more ’1’ bits must follow before the next frame can start
- Normal CAN data exchange is made with 512 kbit/s, or at fastest 1 Mbit/s. So a signal change will occur not faster than every 1-2 µs
- For CAN FD the data payload of maximum 64 bytes is too small to have an unintended activation by a data pattern

Some further safeguarding:

- SPD will assume a ‘1’-value whenever the time between two signal changes is longer than 0.55µs, which is the case at the latest at the end of every CAN FD frame
- Whenever SPD recognizes a ’1’ before 640 ‘0’-values have been seen, the whole activation sequence must be restarted
- The DXCPL activation is in two steps. The first is the SPD activation pattern, and the second a valid DAPISC telegram. Only after the second step, will the CAN module be disconnected from the RX/TX pins

For the special case that the tool is plugged on to a “live” bus, the following points must be considered:

- Because the transmission of SPD-conforming ‘0’-values represents a violation of the CAN-protocol, any active CAN-node will react by actively sending an error-frame
- This is repeated by the same or another node, until the internal error counter(s) overflow(s)
- Sooner or later all nodes will reach this state and be disabled automatically
- These error-frames may disturb the SPD activation sequence when a ’1’ results from any bus conflict. This may require that more than 640 ‘0’ values must be transmitted until SPD is activated

The DXCPL/SPD active operating condition is also indicated to any user software by SMM register bit STSTAT.SPDEN. This allows you to ignore errors reported by the disconnected CAN module.

DXCPL de-activation

Switching the activated interface pins back to regular CAN mode is only possible through Power-on Reset.

Disabling DXCPL

DXCPL can be disabled with TRI_DXCCTR.DXCPLDIS (even if already active).

Disabling Application Test Mode

After enabling DXCPL TRI_DXCCTR.ATMEN is used to determine whether Application Test Mode (ATM) can be entered or not.

46.3.3.3.2 DXCPL SPD parameters

DXCPL will normally be used with a CAN transceiver. It is however possible to connect any other kind of physical layer. The only restriction from the SPD encoding is the requirement for having similar transmission delays for positive and negative signal edges.

Due to the properties of the CAN physical layer, DXCPL uses inverted signal levels compared to standard SPD, as shown in the next figure. CAN's recessive level (high) corresponds to SPD's inactive level (low).

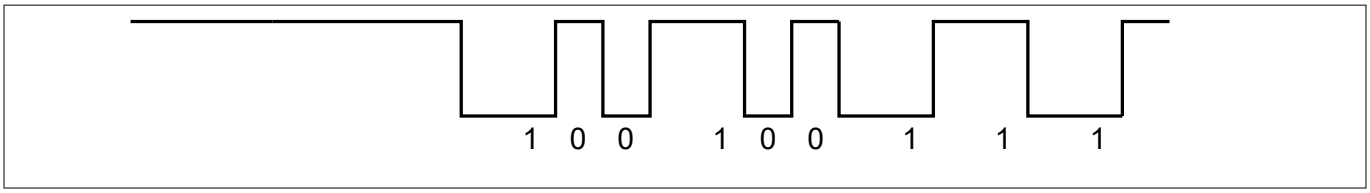


Figure 1249 DXCPL with SPD edge distance encoding

Table 2428 DXCPL SPD parameters

Parameter	Description
Bandwidth	Similar to regular DAP with 1.6 MHz clock with standard settings. It can be increased (DAPISC.CONF) by a factor of 2, 3, or even 4 depending on device type and system design, for example edge delay asymmetry of CAN transceiver.
Clock Source	EVR oscillator (100 MHz nominal)
Transmit Edge distance '0'	0.4 µs (nominal)
Transmit Edge distance '1'	1.1 µs (nominal)
Decision point on receive side	0.75 µs (nominal)
SPD_AUX bit	DAP TGIP trigger information

Attention**: **These DXCPL SPD parameters apply for the whole AURIX™ family.

Achievable bandwidth with a CAN transceiver

The bandwidth is limited by the posedge or negedge asymmetry and the jitter of the transceiver. For CAN transceivers a variation of up to 100 ns has to be taken into account.

The SPD detection fails when, due to this CAN transceiver variation, a short '0' edge distance is prolonged or a long '1' edge distance is shortened so that the receiver detects the wrong value. It also fails when a short

distance is shortened to less than one sample and a long distance to more than the timeout value of the receiver. This effect needs to be taken into account for the SPD parametrization for both tool and device.

On-chip SPD parameters

For minimized quantization loss, an over-sample rate of 5 or more is required. The following parameters are used by the on-chip SPD module:

- Over-sample rate (time step) is 5.
- Transmitter outputs a short edge distance (‘0’) with 4 time steps.
- Transmitter outputs a long edge distance (‘1’) with 11 time steps.
- Receiver recognizes an edge distance of 1 to 7 time steps as ‘0’.
- Receiver recognizes an edge distance of 8 to 14 time steps as ‘1’.

If the same SPD parameters are also used on the tool side, a bandwidth of 600 KB/s can be achieved. This requires good CAN (FD) transceivers at both ends which ensure a 90 ns or less edge distance variation for the whole signal path.

46.3.3.3.3 Halt after reset request (HARR)

For the default DXCPL pins it is possible to have a reliable halt after a . Two options are described here which depend on whether or not the PORST pin can be controlled by the tool. A third option, although not for a , is using a Halt after reset request (HARR), as for DXCM.

***Note**: The target system may have an external watchdog which will create a reset when it is not served by the device software. This effect can be mitigated when the tool takes over this role.*

HARR with controlling the PORST pin

It is possible to have a reliable halt after reset request (HARR) with DXCPL.

The tool needs to execute these steps after releasing the pin:

- Wait for approximately 100 µs for on-board and device internal release
- Apply the DXCPL activation sequence including DAPISC DAP initialization telegram (which takes approximately 300 µs)
- Execute the DAP HARR sequence (see AURIX™ TC4xx debug and trace reference manual)

HARR without controlling the PORST pin

Similar to regular DAP Halt after PORST with DAP, it is possible to have a reliable HARR even in the case that the tool does not control the PORST pin. The tool periodically sends the DXCPL activation sequence, including the DAPISC DAP initialization telegram. When the device responds to the telegram, the tool proceeds with the usual DAP HARR sequence.

For “small” devices with a short HARR time window, the DXCPL activation sequence can be optionally shortened to 170 µs by using half the regular SPD ‘0’ bit width (0.2 µs) for the 640 ‘0’ bits. The DAPISC DAP initialization telegram needs to be sent with a regular bit width.

Related information

DAP over CAN physical layer (DXCPL) on page 8588
CAN as a tool communication option; DAP over CAN messages (DXCM) on page 8594
Comparison of DAP, DXCPL, and DXCM on page 8584
Tool interface overview and enabling on page 8583
SPD protocol overview on page 8593
DXCPL SPD parameters on page 8591
DXCM operation overview on page 8594

DXCPL activation sequence on page 8590

46.3.3.4 SPD protocol overview

The SPD protocol encodes the DAP bits with the distances between the SPD signal edges. A DAP1 value of ’0’ is encoded with a short distance, while a value of ’1' has a distance that is twice as long. Standard encodings provide data bandwidth and latency performance similar to regular DAP connections clocked at 1.0 MHz to 1.6 MHz.

An SPD receiver implementation which is maximum robust against clock frequency deviations will have the decision point between ’0’ and ’1’ centered, with ’0’ being 2/3 of the decision point and ’1’ being 4/3.

This encoding is used for both directions and all data. Initially the SPD signal level will be low and the SPD module will wait for the first telegram from the tool, which will always start with a positive SPD ’1’ signal pulse with the encoded DAP start bit. The transmission of the rest of the telegram will be independent of the edge direction. If the SPD signal level is high after the last bit of the telegram, the tool has to add a ’0’ bit with a negative edge.

DAP has a clean request response protocol, where the tool is always the requester and the device executes and sends a response. So the DAP module will change the direction after it has received a DAP telegram to send a response. This response is the DAP reply telegram with two additions:

- After the start bit one bit is inserted. This SPD_AUX bit can be the value of the port output bit when SPD is overlaid to a GPIO pin, a trigger indicator like DAP TGIP, or a constant ’0’. Note that this extra bit is not included in the CRC of the DAP reply telegram.
- If the SPD output value is 1 after the last telegram bit, the SPD module will also automatically append an SPD’0’ bit to bring the SPD value back to '0'.

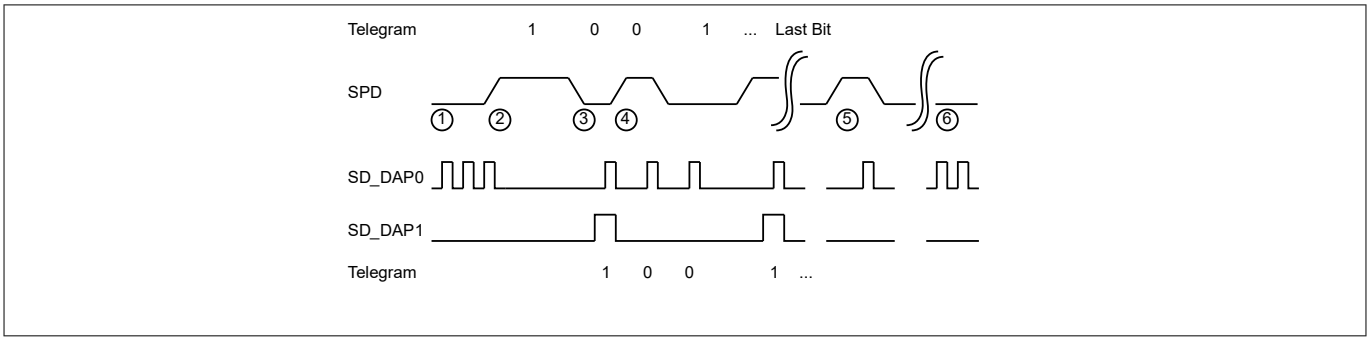


Figure 1250 SPD encoding example

The figure above shows the conversion of an SPD telegram to DAP as an example. SPD shows the input signal to be converted, while the signals SD_* are the signals towards the DAP.

Initially (1) the SPD module is in IDLE state and will clock ’0’ bits into the DAP module.

At point (2) it detects a rising edge which will bring it to RECEIVE state, waiting for the next edge at point (3). The distance between these two points is 1µs which means the first bit is recognized as’1’ (start bit) and output to the DAP module.

At point (4), 0.5 µs later the next bit, in this example with the value’0’, is detected and then output to the DAP module. If the last bit of the DAP telegram results in a rising SPD edge the tool will append a’0’ bit to bring the SPD level back to Low (5).

If the SPD level is Low for a certain amount of time, the SPD module will automatically switch back to IDLE mode (6). The SPD protocol can be used for all DAP telegrams except the poll and turn_off telegrams.

Tool interaction

All SPD/DAP communication is initiated by the tool. So the tool will send an SPD encoded telegram and then just switch the SPD signal direction to input to wait for the response from the device. The device will detect this

state due to the missing SPD edges and will start to create the response. The response will be either an SPD telegram or just an SPD start bit as defined in the DAP protocol.

46.3.3.5 DAP over CAN Messages (DXCM)

AURIX™ TC4xx supports standardized CAN tool access in parallel to regular CAN bus traffic. The main application is debugging, but it can also be used for any other kind of tool, that requires robust device access independent of the user software.

The generic concept is DAP over Serial (DXS) and in this case using CAN messages (DXCM). The effective DAP protocol is two-pin DAP. DAP telegrams and replies are transmitted as regular CAN messages.

The DXCPL/DXCM default pins are CAN01_TXD and CAN01_RXDB.

46.3.3.5.1 DXCM introduction

The basic idea is to have another transmission layer for DAP telegrams and replies. This is accomplished by packaging the DAP payload as CAN message data.

Features

- Tool access in parallel to regular CAN traffic
- Fully compliant to CAN protocol
- Nearly unrestricted support of all DAP features
- Security level and operation the same as for regular DAP
- Performance of 1 MBaud CAN similar to a 300 kHz regular DAP interface

The following figure shows the DXCM integration in AURIX™ TC4xx. The DXCM converter between the CAN module and DAP module serializes the incoming CAN message data and de-serializes the reply from the DAP module.

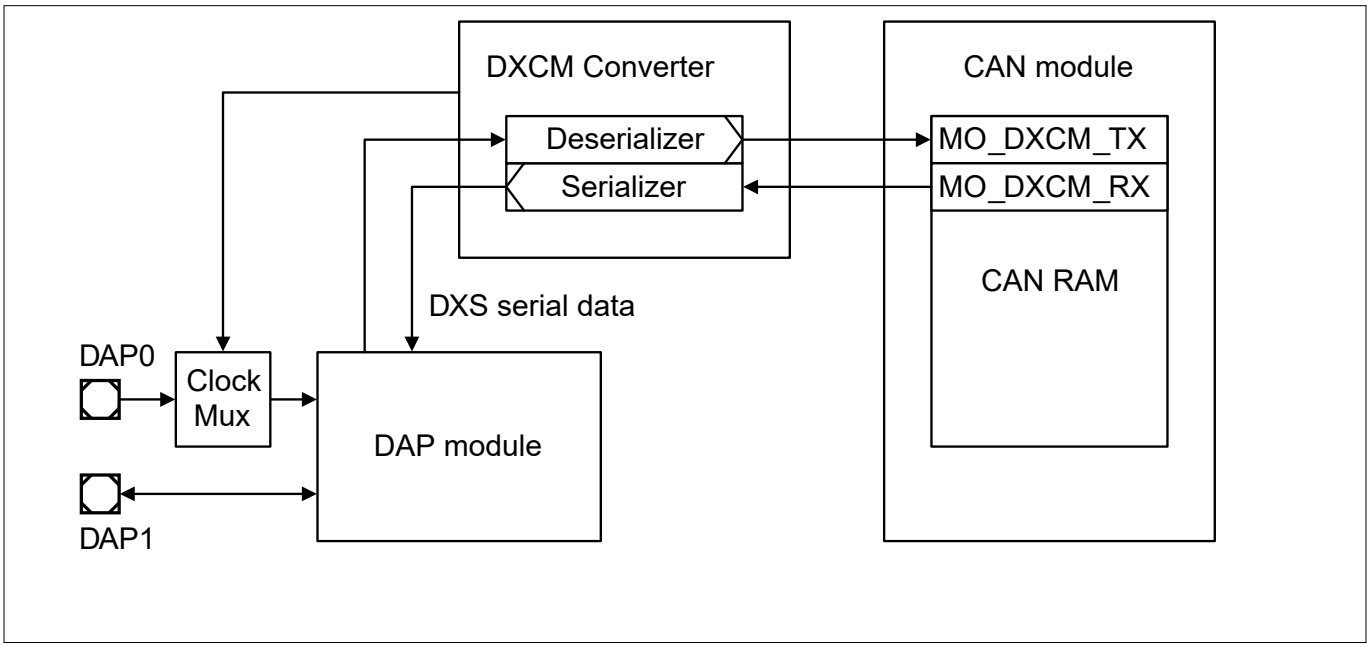


Figure 1251 DXCM block diagram

46.3.3.5.2 DXCM operation overview

There are no special restrictions for configuring the CAN module. The tool is simply another node in the CAN network. The tool must follow in principle the “request and wait for response” scheme. All messages, except for

a DAP turnoff telegram, will lead to a response message. All DAP telegrams can be used except the ones which obviously do not make sense, such as turn off JTAG for example.

There are only a few restrictions for DXCM compared to regular DAP:

- The achievable performance is very limited due to the CAN bit rate
- JTAG boundary scan is not possible since the DXCM pins are part of the scan chain

Related information

Halt after reset request (HARR) on page 8592

DXCM enabling

DXCM is only available when DAP is not used after , so no DAPISC DAP initialization telegram has been received. This means that either DAP, DXCM, or DXCPL can be used at any time. DXCM needs to be initialized in the CAN0 module by user software (MCR.DXCM).

Related information

Controller Area Network interface (CAN) on page 3918

DXCM protocol

DAP telegrams are sent by the tool without a start bit and CRC6 as the data of regular CAN messages. The LSB of the first byte contains the LSB of the DAP CMD field. The CAN message payload has just as many bytes as needed for transmitting the telegram bits. The remaining bits in the last byte must be set to zero.

The device reply is also without a start bit and CRC6. If the Trigger in Protocol (TGIP) is enabled, the DAP1 trigger bit follows immediately after the RDATA field. Reply parcels are sent as individual CAN messages. The trigger bit is appended according to the standard DAP behavior.

If the DAP reply is just a start bit a zero byte CAN message (DLC = 0) is generated.

Pipelining is not supported. The tool operation is a simple request response scheme, just the same as for regular DAP. But for maximum performance the tool can have the next DXCM CAN message fully prepared while it waits for the reply.

Halt after reset request (HARR) with a breakpoint

The initial DXCM tool connection is always a hot attach, and the tool can then request a System or Application Reset using the OJCONF register, which is the same as for regular DAP. It is however not possible to request a HARR on this level, since the CAN module will be reset and the HARR would prevent DXCM initialization by the user software.

The solution is to halt the user software with a breakpoint after the CAN module initialization, since the OCDS resources are not reset by a System or Application Reset.

This breakpoint based HARR approach is not a real restriction since:

- The user software before the CAN module initialization can be debugged through a DXCPL connection if needed. This part of the user software is independent from any CAN bus activity in the system
- The CAN module initialization can be optionally split in two parts: a very early initialization, which just sets up DXCM, and later a full initialization which adds dynamic user settings for example. This allows a very early DXCM connection; for example, before the entry to main()

46.3.3.6 Tool identification

The tool identifier and start-up handshake registers (TIHDAP,TIHDAPE and TIHGP) allow an external tool to identify itself to the device software and optionally set up a handshake procedure to reliably establish a link to the device.

***Note**: The tool identification interface is primarily intended to be used with measurement and calibration tools. Debug tools would not usually make use of this interface.*

46.3.3.7 Recommended tool connectors

Two connection options can be used to gain an easy and reliable connection.

- AP24003 DAP connector: DAP standard connector based on a 0.05 inch double row, 10 pin micro terminal with keying shroud
- AP32536 HSTCU: High speed tool connector based on USB C-type connector

46.3.3.8 Device access server (DAS)

The DAS API provides an abstraction of the physical device interface for tool access. The key paradigm of DAS is to read or write data in one or several address spaces of the target device.

DAS features:

- Standard interface for all types of tools.
- Efficient and robust methods for data transfer.
- Standardized system security support (authorization).
- Several independent tools can share the same physical interface.
- Infineon's DAS miniWiggler supports DAP, SPD, and JTAG.

46.3.4 Debug and Trace interaction

Debug and Trace may be used in a cooperative manner regarding OCMTS configuration and data export.

OCMTS configuration with Debug interface

Because Debug has full system access, it may be used to configure the whole OCMTS space including trace capabilities. This is done by accessing the TPB and SRI based access spaces with the CBS or IOCE Master interface.

***Note**: For Debug support, the device security state must allow Debug IO clients to access the system in read write mode. In addtion, the Debug TAG-ID must be allowed for write accesses in the PROT mechanism of peripherals where a debug tool should be allowed to alter the configuration.*

46.3.5 Emulation Device variant features

Table 2429 Emulation Device (ED)variant features

Features	TC...								
	4Dx	4Zx	48x	44x	49x	46x	4Ex	4xx	45x, 4Px, 42x, 4Sx
	Emulation Device							Productive Device	
Basic Overlay (DSPR, PSPR, DLMU, OLDA	X[1]	X	X	X	X	X	X	X	X
OWN (Overlay with NVM)	X	X	X	X	X	X	X	-[2]	-
LMU (EMEM) for Overlay and trace data	X	X	-	-	X	X	-	-	-
Full MCDS function for 48/32/16 kB TBUF trace	X	X	X	X	X	X	X	X	-
Trace session limitation after reset	No	No	No	No	No	No	No	Yes[3]	-
Trace to any RAM (e.g. LMU, if available)	X	X	X	X	X	X	X	-	-
DAP	X	X	X	X	X	X	X	X	X
DAPE, WM, SWM[4]	X	X	X	X	X	X	X	-	X
SGBT/Aurora trace output	X	X	X	X	X	X	-	-	-
High speed trace via Ethernet (SGMII)	X	X	X	X	X*[5]	X*	-	-	-

1) 'X' in this table, indicates that the feature is supported
2) '-' in this table indicates that the feature is not supported
3) Limited to one trace session after reset. Trace session defines the time between trace start and trace stop. For production devices between one trace session and the following one, a system reset is needed.
4) SWM feature is not available in TC4Dx, TC49x,TC48x
5) 'X*' - Supported in BGA436 package only

The Table above shows the Emulation device specific features and comparison to the Productive devices. All TC4x Productive devices (PD) have basic debug and overlay features. However,TC4x Emulation Device (ED) chips include debug and tracing features relevant for development phase only.

46.4 Revision history

Reference	Description of change(s)
Date range: 2024-04-26 to 2024-11-08	
DAP, JTAG, and trigger pins	• Five-pin DAP added in table "DAP, JTAG, Trigger and GPIO pin configurations"
Emulation Device variant features	• Chapter added

47 Ports

Each Port module 'Port slice' controls a set of assigned General Purpose Input Output (GPIO) port lines which are connected to pads connected to device pins or balls.

47.1 Feature list

Depending on its configuration a port module can have the following features:

- Controls up to 16 port lines
- Multiplexes one out of a maximum of 15 alternate functions to each port line
- Supports direct IO (Input Output) control by a peripheral on a per line granularity
- Enables software to control the output of each port line
  - Each port line has its own set of control registers
  - Output modification registers ease clearing, setting, and toggling of single or multiple port lines without affecting the state of other port lines
  - Enables software to read the input value of each port line
- Supports local access control with up to 8 access groups
  - Each port line inside a port can be mapped freely to an access group
- Supported pad characteristics of the assigned pads such as drive strength, slew rate, pull-up and pull-down, push or pull or open-drain operation, selection of TTL (Transistor-transistor logic) or CMOS (Complementary Metal Oxide Semiconductor) or automotive input levels
- The emergency stop feature allows the SMU (Safety Management Unit) or special port pins to change the driver configuration of configurable port lines
- For pad pairs with LVDS (Low voltage differential signaling) functionality, it controls LVDS characteristics and allows switching between LVDS and CMOS modes
- Also in packages with reduced pin count the port module is used to disable these particular pins which are not available in this packages

Device dependent implementation of ports

***Note**: The implementation of the ports is dependent on the device. Configuration is done partly during start-up by the firmware, for example to disable pins not available in a certain package. Please refer to the device specific documentation for the configuration.*

47.2 Functional overview

Each port line has a number of control and data bits, enabling very flexible usage of the line. Each port pad can be configured for input or output operation.

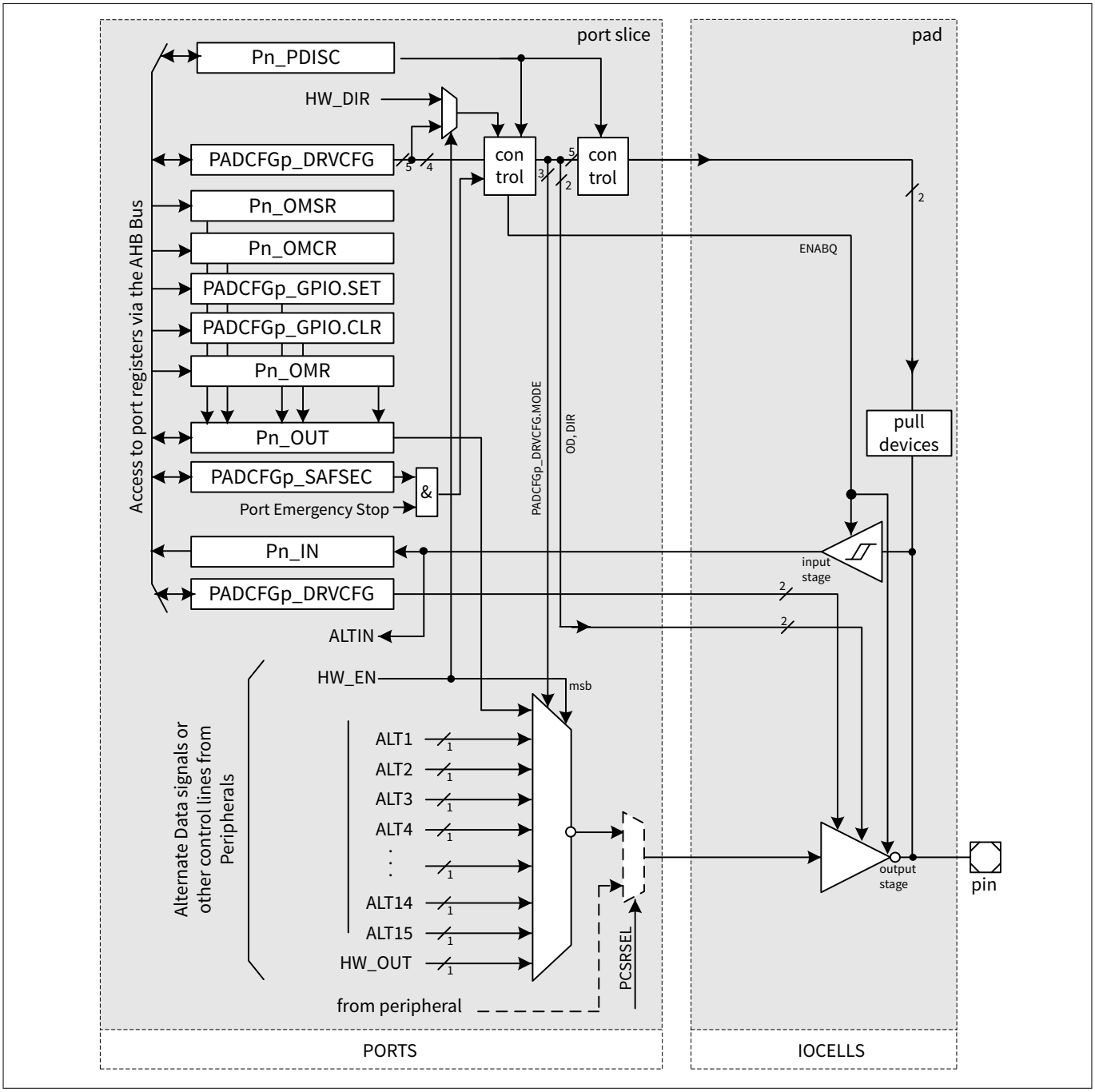


Figure 1252 PORTS block diagram

For the alarm description please refer to the alarm mapping tables in the SMU functional block and the Safety Manual.

The available features depend on the type of pad. For example, input-only pads (e.g. SENT pads) do not offer output control. In this text we describe the capabilities of a general CMOS pad. The section on LVDS Port Operation adds information for switchable LVDS or CMOS pads.

In input mode, which is the default mode after reset, the output driver is switched off (high-impedance). The actual voltage level present at the port pin is translated into a logical 0 or 1 through a Schmitt-Trigger device and can be read with the read-only register IN. Input signals are connected directly to the various inputs of the

peripheral units. The function of the input line from the pin to the input register IN and to the peripheral is independent of whether the port pin operates as input or output. This means that when the port is in output mode, the level of the pin can be read by software through IN or a peripheral can use the pin level as an input.

In output mode, the output driver is activated and drives the value supplied through the multiplexer to the port pin. Switching between input and output mode is accomplished through the PADCFGp_DRVCFG register, which enables or disables the output driver. If a peripheral unit uses a GPIO port line as a bi-directional IO line, register PADCFGp_DRVCFG has to be written in order to configure input and output settings according to system's needs. The PADCFGp_DRVCFG register further controls the driver type of the output driver, and determines whether an internal weak pull-up, pull-down, or without input pull device is alternatively connected to the pin when used as an input.

The output multiplexer in front of the output driver selects the signal source for the GPIO line when used as output. If the pin is used as general-purpose output, the multiplexer is switched by software (PADCFGp_DRVCFG register) to the Output Data Register OUT.

The status of OUT can be changed through separate OMSR or OMCR registers.

The set or clear operations for the bits in OUT can be made by setting the corresponding PADCFGp_GPIO.SET and PADCFGp_GPIO.CLR registers. Alternatively, the set, clear, or toggle function can be achieved through OMR, where all pins within the same port can be set, cleared, or toggled within one write operation. The manipulation of the control bits in these registers can directly influence the state of the port pin. If the on-chip peripheral units use the pin for output signals, the alternate output lines ALT1 to ALT15 can be switched through the multiplexer to the output driver.

The data written into the output register OUT by software can be used as input data to an on-chip peripheral. This enables for example, peripheral tests using software without external circuitry (ALTIN path).

When selected as general-purpose output line, the logic state of each port pin can be changed individually by programming the pin-related bits in the

- Port n Output Modification Set Register: OMSR.PSx (where x_0 through to 15)
- Port n GPIO Control Register for Pin p: PADCFGp_GPIO.SET
- Port n Output Modification Clear Register: OMCR.PCLx (where x=0 through to 15)
- Port n GPIO Control Register for Pin p: PADCFGp_GPIO.CLR
- Port n Output Modification Register: OMR

The bits in OMSR and OMCR make it possible to set and clear the bits in the OUT register.

The bits in OMR allows the bits in OUT to be set, cleared, toggled or remain unchanged.

When the GPIO is selected as general-purpose output line, its actual logic level at the pin can be examined through reading IN and compared against the applied output level, either applied through software through the output register OUT, or through an alternate output function of a peripheral unit. This can be used to detect some electrical failures at the pin caused through external circuitry. In addition, software-supported arbitration schemes can be implemented in this way using the open-drain configuration and an external wired-And circuitry. Collisions on the external communication lines can be detected when a high level (1) is output, but a low level (0) is seen when reading the pin value through the input register IN.

Most of the digital GPIO lines have an emergency stop logic. This logic makes it possible to individually disconnect outputs and put them onto a well defined logic state in an emergency case. In an emergency case, the pin is switched to input function with internal pull-up device connected or tri-state (depending on global configuration). The Emergency Stop Register PADCFGp_SAFSEC determines whether an output is enabled or disabled in an emergency case.

LVDS port operation

The LVDS pads offer LVDS RX or TX functionality for a pair of pins. Additionally, they contain, if not specified differently, a bi-directional CMOS pad for each of the pins. The CMOS functionality is controlled by the set of standard registers. The LVDS functionality is controlled by additional LPCRx.LRXTERM registers.

Related information

TC4Dx SMU alarm mapping tables on page 7233

47.3 Functional description

System connectivity of ports

The connectivity of the Ports is described in the pinning tables, which are available in the datasheet. The following table shows the general structure of a connectivity description of a port pin.

Table 2430 Example port table

Pin number	Symbol	Ctrl.	Buffer type	Function
10	Pxx.y	I	FAST/PU1/VEXT	General -purpose input
	TIMn_m			GTM_TIN
	Pxx.y	O0		General-purpose output
	TOMa_b	O1		GTM_TOUT
	TOMc_d			GTM_TOUT
	IOM_REFv_w			IOM reference input
	ASC_LINz_RTS	O2		ASCLIN0 output
	...	...		...
	ETH0_MDIO	O		Ethernet output

The symbols in the Ctrl. column indicate the multiplexer connectivity:

- I = Input (for GPIO port lines with DRVCFG bit field Selection MODEx = 0XXXB)
- I# = Input (for GPIO port lines which are shared with Standby Controller (SCR))
- AI = Analog input
- O = Output (the assigned function is active when a peripheral takes control of the port line)
- O0 = Output with DRVCFG bit field selection MODEx = 0000_B (Pxx.y)
- O1 = Output with DRVCFG bit field selection MODEx = 0001_B (ALT1)
- O2 = Output with DRVCFG bit field selection MODEx = 0010_B (ALT2)
- O3 = Output with DRVCFG bit field selection MODEx = 0011_B (ALT3)
- O4 = Output with DRVCFG bit field selection MODEx = 0100_B (ALT4)
- O5 = Output with DRVCFG bit field selection MODEx = 0101_B (ALT5)
- O6 = Output with DRVCFG bit field selection MODEx = 0110_B (ALT6)
- O7 = Output with DRVCFG bit field selection MODEx = 0111_B (ALT7)
- …
- O15 = Output with DRVCFG bit field selection MODEx = 1111_B (ALT15)
- O# = Output (for GPIO port lines which are shared with Standby Controller (SCR)) or HW_OUT or direct out (controlled by PCSRSEL bit on port slices not controlled by Standby Controller)

Several functions can be connected to each input. The peripherals' configuration defines if this input is used.

The port module decides which of the 16 output signals O0 to O15 drives the pad.

***Note**: Some Ox rows list more than one function. For example, several TOM outputs. The GTM module has its own sub-multiplexer structure that defines which of the GTM sub-units drives this signal.*

***Note**: In order to achieve a spike free switching of the ouput function selection it is recommended to disable the output driver before the selection, change to the desired output function (ALTx), and then to enable the output driver again.*

Pins without port module

The pinning of some devices shows Px.y pins without assigned Port module “Px”. This is the case for HSPHYx (used for SGBT, PCIe and SGMII functionaltiy) module pins for which the HSPHYx modules themselves contain the necessary control registers.

Control of port line by peripheral (HW_DIR/HW_EN/HW_OUT)

By using the HW_DIR/HW_EN/HW_OUT interface, a peripheral can take over control of the output path of a port line. When the peripheral activates HW_EN its data line connected to HW_OUT is driven to the output driver. With HW_DIR the peripheral can control the output driver activation. The data line connected to HW_OUT is shown in the pinning tables with the ctrl type as “O”.

When a peripheral activates HW_DIR and HW_EN the output driver operates in push/pull mode.

Wakeup control and wakeup status

After a wakeup on a falling edge of a port slice the corresponding wake status (which pin(s) have caused the wakeup) can be read in the corresponding port WKSTS. In order to make the status visible the corresponding bits inside the WKEN have to be set by user software directly after wakeup.

***Note**: During LBIST operation wakeup functionality is disabled but the wakeup status is preserved.*

Ports protection concept

The following figure gives an overview of the port protection concept.

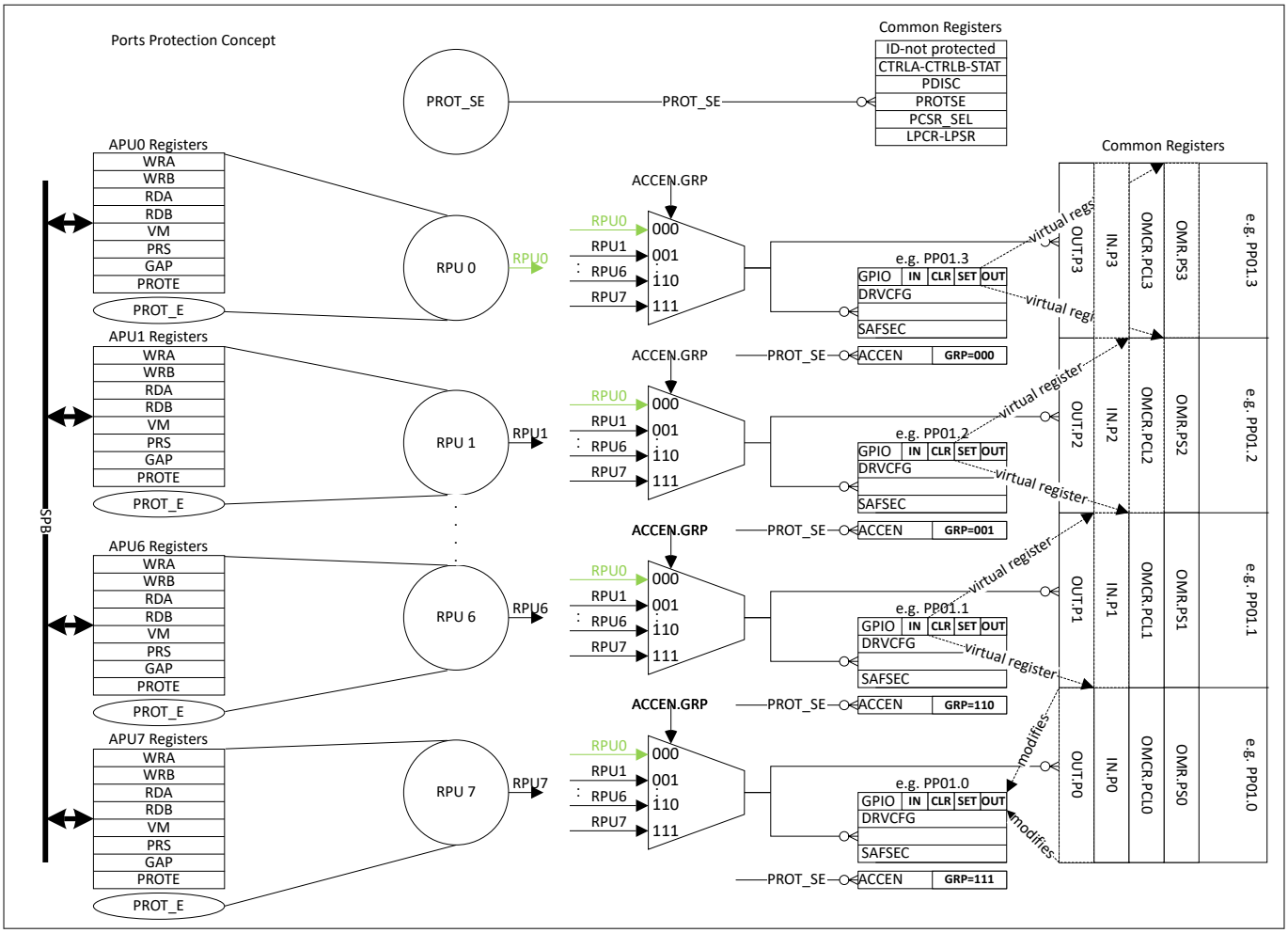


Figure 1253 Protection concept example demonstrated on 8 port slices

Introduction

Once the Power on Reset (PORST) is completed, the SYS_FW will hand over the control to the system start software or Operating system, which runs under Supervisor mode or, in case of a fully booted and virtualized application environment, as Hypervisor with the VM Tag 0. This software will set the remaining configuration, where the set values will depend on the use cases.

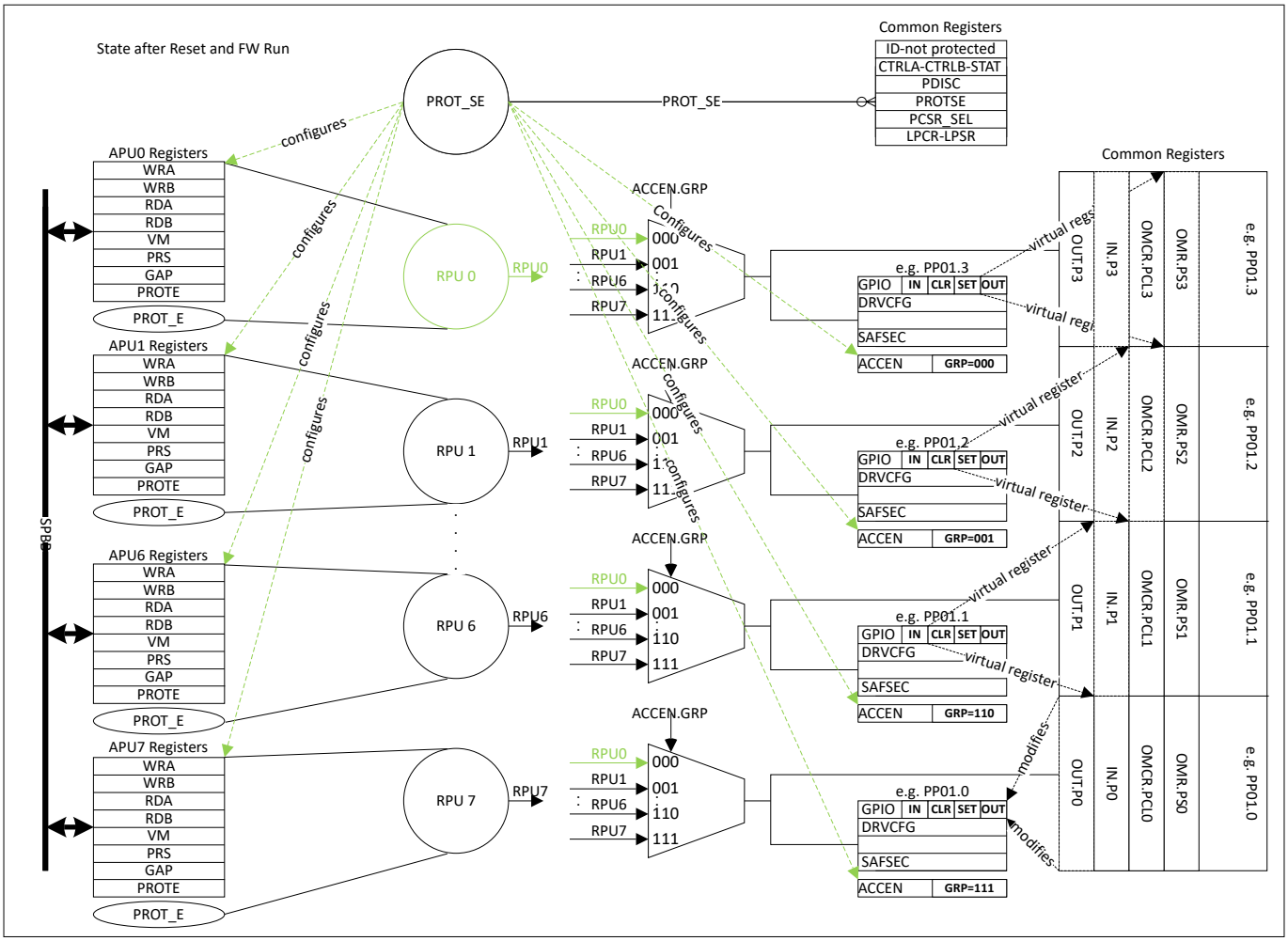


Figure 1254 Protection configuration example by PROT_SE owner

When all PAD configurations like Push Pull, GPIO or the alternative function in use are set, the system software will lock the configuration by setting ENDINIT.

In this mode, the functional driver software will change the GPIO pin levels via the dedicated registers. If such a driver erroneously or in a hostile manner accesses configuration registers, the access will be prohibited and the Safety Management Unit will be informed.

Port slice

Each port consists of maximum 16 port slices which are controlled by their dedicated register sets holding the control of the Pad configuration and GPIO control. Those port slice control registers are:

- PADCFGp_GPIO
- PADCFGp_DRVCFG
- PADCFGp_ACCEN
- PADCFGp_SAFSEC

The access to those port slice registers is offered in a grouped fashion. All slices within a port can be grouped in maximum eight access groups (configurable by the PADCFGp_ACCEN.GRP register).The number of access groups determines the number of APUs (four access groups result in 8 APUs). Each APU includes the configurability for all required ID types by offering a register per ID type.

Access group usage example

The following figure shows an example of an access flow and how the access groups do control the different functions in a port module.

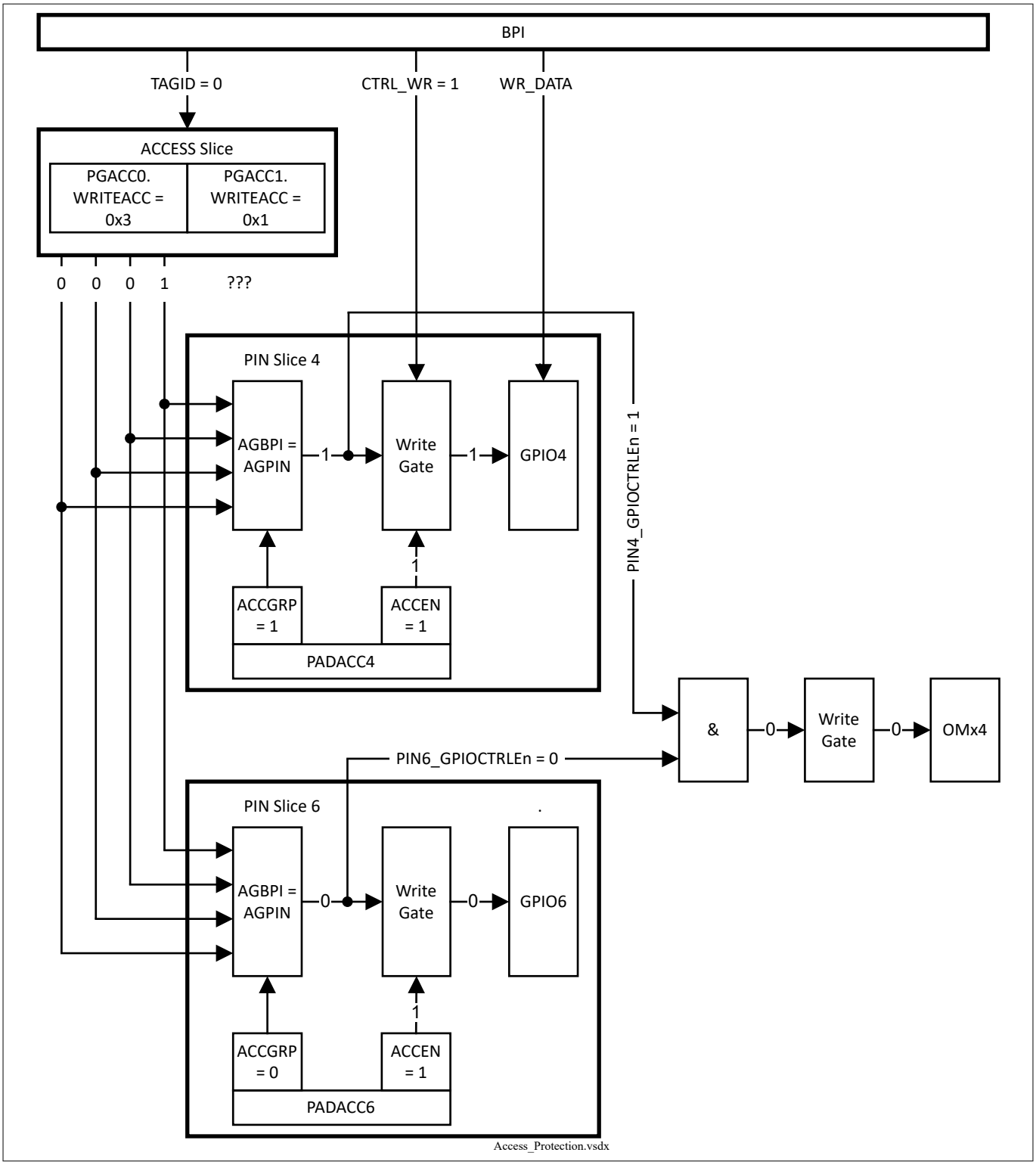


Figure 1255 Access example of two port slices

1. CPU with TAG ID = 0 writes to PORT
2. The ACCESS slice indicates a match in ACCESS group 1
   - Result: TAGID0 is enabled for write accesses on all GPIOs assigned to this group
3. Port slice4 enables write to GPIO4
4. Port slice6 and OMx4 are locked as Pin6 is assigned to non-matching access group 3

Access violation for ID register

The ID register is a read only register where the read access does not generate any access violation.

Access violation for Pn_IN register

The IN and the HWSELSTAT registers are common registers and are read only. In case of a read operation by a master not owning the access rights for a particular bit field this bit field will be read as 0. All other bit fields owned by the reading master will show the real status of the owned pin.

Access violation for Pn_OUT register

The OUT register is a special case within the group of common registers. Either writing a zero or writing a one will affect the status. In this case a modification of a bit field within this register can only be done when the writing master owns this particular bit field it wants to write. All other bit fields not owned by the writing master will not be modified. As the writing master has also tried to modify bits which it does not own, a safety alarm is generated.

In order to be able to perform a simple toggle activity on a pin the dedicated registers PADCFGp_GPIO.SET and PADCFGp_GPIO.CLR have to be set alternately.

Access violation for Pn_OMR, Pn_OMCR and Pn_OMSR register

For registers OMR, OMCR and OMSR only writing a one affects the status. In case group access protection is active, any write of a one to a bit-field not owned by the writing master will not lead to a reaction on pin level and will cause a safety alarm.

Reading those registers will return a zero.

***Note**: The OMR, OMCR and the OMSR registers are virtual registers and therefore do not contain any flip flop.*

Behavior on access failures

The port module reacts differently depending on the cause of access failure:

- An access attempted to a non-existing address triggers a bus error
- In case an access violates the currently valid access group settings, an alarm is raised towards the SMU

Related information

Padframe alarm trigger signals on page 8649

47.4 Registers

47.4.1 Register overview - access mode glossary

Table 2431 Register overview - access mode glossary

Keyword	Description
Ey (y=0-7)	Access protection using PROT register ACCGRPy_PROTE.
SE	Access protection using PROT register PROTSE.
APU-PGy (y=0-7)	Protection group consisting of registers ACCGRPy_WRA, ACCGRPy_WRB, ACCGRPy_RDA, ACCGRPy_RDB, ACCGRPy_VM, ACCGRPy_PRS.
PGy	Access protection using *APU-PGy* registers.
PPSz (z=0-15) EPSz (z=0-15)	Access protection using one APU from *APU-PG[y] (y=0-7)*. PROT_E protection using one of *E[y] (y=0-7)*. The protection source to be used is selected by bit-field: z=0-15: PADCFGp_ACCEN.GRP
U	No access restrictions.
BE	Always returns a Bus Error.
PROT	Access restrictions as defined in the PROT register access rules.
nBE	Indicates that no Bus Error is generated when accessing this address range, even though it is either an access to an undefined address or the access does not follow the given rules.

47.4.2 Registers overview - Pn (ascending offset address)

Table 2432 Registers overview - Pn (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
ID	Module Identification Register	000_H	U	BE	Application Reset	8610
RST_CTRLA	Reset control register A	004_H	SV, SE	SV, SE	Application Reset	8610
RST_CTRLB	Reset control register B	008_H	SV, SE	SV, SE	Application Reset	8611
RST_STAT	Reset status register	$00C_H$	SV, SE	BE	Application Reset	8612
WKEN	Port n Wake Enable Register	010_H	U	U	Application Reset	8612
WKSTS	Port n Wake Status Register	014_H	U	BE	Application Reset	8613
WKENSTS	Port n Wake Enable Status Register	018_H	U	BE	Application Reset	8613
WKENSTSCLR	Port n Wake Status Clear Register	$01C_H$	U	U	Application Reset	8614
OUT	Port n Output Register	020_H	U	U	See 8614	8614

(table continues...)

Table 2432 (continued) Registers overview - Pn (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
IN	Port n Input Register	024_H	U	BE	See 8615	8615
HWSELSTAT	Port n HWSEL status Register	028_H	U	BE	See 8616	8616
PDISC	Port n Pin Function Decision Control Register	$02C_H$	U	SV, SE	See 8617	8617
PROTSE	PROT Register Safe Endinit (SE)	030_H	U	SV, PROT	Application Reset	8618
PCSRSEL	Selection Register for Module x Port n Control	034_H	U	SV, SE	PowerOn Reset	8619
BGTRIM	Pad Bandgap Trim Register	038_H	U	SV, SE	See 8620	8620
OMR	Port n Output Modification Register	$03C_H$	nBE	U	Application Reset	8622
OMCR	Port n Output Modification Clear Register	040_H	U	U	Application Reset	8622
OMSR	Port n Output Modification Set Register	044_H	U	U	Application Reset	8623
LPSRx	Port n LVDS Pad Status Register x	048_H+x* 8	U	BE	Application Reset	8624
LPCRx	Port n LVDS Pad Control Register x	$04C_H$+x* 8	U	SE, SV	See 8624	8624
ACCGRPy_WRA	Write access enable register A	090_H+y* 20_H	U	SE, SV	Application Reset	8626
ACCGRPy_WRB	Write access enable register B	094_H+y* 20_H	U	SE, SV	Application Reset	8627
ACCGRPy_RDA	Read access enable register A	098_H+y* 20_H	U	SE, SV	Application Reset	8628
ACCGRPy_RDB	Read access enable register B	$09C_H$+y* 20_H	U	SE, SV	Application Reset	8628
ACCGRPy_VM	VM access enable register	$0A0_H$+y* 20_H	U	SE, SV	Application Reset	8629
ACCGRPy_PRS	PRS access enable register	$0A4_H$+y* 20_H	U	SE, SV	Application Reset	8630
ACCGRPy_PROTE	PROT Register Endinit	$0AC_H$+y* 20_H	U	SV, PROT	Application Reset	8630
ENCONF_RDDIAG	Port n LDO Configuration and control register	190_H	U	U	See 8641	8641
TRIMCONF_PROT	Port n LDO Trimming register	$19C_H$	U	U	See 8642	8642
PADCFGp_GPIO	Port n GPIO Control Register for pin p	300_H+p* 10_H	PPSp	PPSp	See 8632	8632

(table continues...)

Table 2432 (continued) Registers overview - Pn (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
PADCFGp_DRVCFG	Port n Pad configuration Control Register for padp	304_H+p* 10_H	PPSp	EPSp, SV, PPSp	See 8633	8633
PADCFGp_ACCEN	Port n Pad Access Control Register for padp	308_H+p* 10_H	U	SE, SV	Application Reset	8638
PADCFGp_SAFSEC	Port n Pad Safety and Security Control Register for padp	$30C_H$+p* 10_H	PPSp	EPSp, SV, PPSp	See 8638	8638

47.4.3 Module Identification Register

ID Offset address: 000_H

Module Identification Register Application Reset value: $00C8\ C000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MOD_NUM															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MOD_TYPE								MOD_REV							
r								r							

Field	Bits	Type	Description
MOD_REV	7:0	r	**Module Revision** Indicates the revision number of the implementation.
MOD_TYPE	15:8	r	**Module Type** This internal marker is fixed to $C0_H$.
MOD_NUM	31:16	r	**Module Number** This bit field defines the module identification number. The value for the Ports module is $00C8_H$

47.4.4 Reset control register A

RST_CTRLA Offset address: 004_H

Reset control register A Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				GRSTEN3	GRSTEN2	GRSTEN1	GRSTEN0	0							KRST
r				rw	rw	rw	rw	r							rwh

Field	Bits	Type	Description
KRST	0	rwh	**Kernel reset** Request a kernel reset. The requested reset is executed if CTRLA.KRST=1 and CTRLB.KRST=1. KRST is cleared after the kernel reset was executed. 0_B No action 1_B A kernel reset was requested
GRSTENx (x=0-3)	x+8	rw	**Enable for global module reset group x** 0_B Global module reset group x does not have any effect 1_B Global module reset group x results in a kernel reset
0	7:1, 31:12	r	**Reserved** Read as 0; should be written with 0.

47.4.5 Reset control register B

RST_CTRLB Offset address: 008_H

Reset control register B Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
STATCLR								0							
w								r							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															KRST
r															rwh

Field	Bits	Type	Description
KRST	0	rwh	**Kernel reset** Request a kernel reset. The requested reset is executed if CTRLA.KRST=1 and CTRLB.KRST=1. KRST is cleared after the kernel reset was executed. 0_B No action 1_B A kernel reset was requested
STATCLR	31	w	**Kernel reset status clear** Clears all status bits in RST_STAT when 1 is written. Read will return 0. 0_B No action 1_B Write with ´1´ clears bits STAT.GRSTx and bit STAT.KRST .
0	30:1	r	**Reserved** Read as 0; should be written with 0.

47.4.6 Reset status register

The reset status register contains the status bits for kernel reset (KRST) and the global module reset groups (GRSTx).

RST_STAT Offset address: $00C_H$

Reset status register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				GRST3	GRST2	GRST1	GRST0	0							KRST
r				rh	rh	rh	rh	r							rh

Field	Bits	Type	Description
KRST	0	rh	**Kernel reset status** Indicates an executed kernel reset. RST_STAT.KRST is set after the execution of a kernel reset in the same clock cycle in which the reset bits are cleared. Clear KRST by setting bit STATCLR in register RST_CTRLB.
GRSTx (x=0-3)	x+8	rh	**Status for global module reset group x** 0_B Reset was not triggered by global reset group x 1_B Reset was triggered by global reset group x
0	7:1, 31:12	r	**Reserved** Read as 0.

47.4.7 Port n Wake Enable Register

The wake enable register contains the enable of the wakeup circuitry of the dedicated pad. If the pad is enabled for wakeup it will act as wakeup source in STANDBY mode.

WKEN Offset address: 010_H

Port n Wake Enable Register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WKEN15	WKEN14	WKEN13	WKEN12	WKEN11	WKEN10	WKEN9	WKEN8	WKEN7	WKEN6	WKEN5	WKEN4	WKEN3	WKEN2	WKEN1	WKEN0
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
WKENx (x=0-15)	x	rwh	**Wake Enable Bit x** This bit enables or disables the wakeup at the pin Pn.x. 0_B The wakeup capability of Pn.x is disabled. 1_B The wakeup capability of Pn.x is enabled.
0	31:16	r	**Reserved** Read as 0; should be written with 0.

47.4.8 Port n Wake Status Register

The wake status register contains the wakeup information provided by the wakeup circuitry of a dedicated pad.

WKSTS Offset address: 014_H

Port n Wake Status Register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WKSTS15	WKSTS14	WKSTS13	WKSTS12	WKSTS11	WKSTS10	WKSTS9	WKSTS8	WKSTS7	WKSTS6	WKSTS5	WKSTS4	WKSTS3	WKSTS2	WKSTS1	WKSTS0
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
WKSTSx (x=0-15)	x	rh	**Wake Status Bit x** This bit indicates the wakeup status at the input pin Pn.x. 0_B No wakeup triggered by Pn.x. 1_B Wakeup triggered by Pn.x.
0	31:16	r	**Reserved** Read as 0.

47.4.9 Port n Wake Enable Status Register

The wake enable status reflects the status of the wakeup circuitry. It is set when the corresponding WKEN is set to one and indicates that wakeup cicuitry is available and STANDBY mode can be entered.

WKENSTS Offset address: 018_H

Port n Wake Enable Status Register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WKENS15	WKENS14	WKENS13	WKENS12	WKENS11	WKENS10	WKENS9	WKENS8	WKENS7	WKENS6	WKENS5	WKENS4	WKENS3	WKENS2	WKENS1	WKENS0
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
WKENSx (x=0-15)	x	rh	**Wake Enable Status Bit x** This bit indicates if the enable of a wakeup at the pin Pn.x was succesful. 0_B The enable of a wakeup at Pn.x was not successful. 1_B The enable of a wakeup at Pn.x was successful.
0	31:16	r	**Reserved** Read as 0; should be written with 0.

47.4.10 Port n Wake Status Clear Register

The wake status clear register contains the clear flags used to clear the wake enable status information.

WKENSTSCLR Offset address: $01C_H$

Port n Wake Status Clear Register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WKENSC15	WKENSC14	WKENSC13	WKENSC12	WKENSC11	WKENSC10	WKENSC9	WKENSC8	WKENSC7	WKENSC6	WKENSC5	WKENSC4	WKENSC3	WKENSC2	WKENSC1	WKENSC0
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

Field	Bits	Type	Description
WKENSCx (x=0-15)	x	w	**Wake Enable Status Clear Bit x** This bit clears the WKENSTS bit of Pin Pn.x. 0_B No clear of WKENSTS of Pin Pn.x. 1_B Clear of WKENSTS of Pin Pn.x.
0	31:16	r	**Reserved** Read as 0; should be written with 0.

47.4.11 Port n Output Register

The port output register determines the value of a GPIO pin when it is selected by Pn_DRVCFGxx as output. Writing a 0 to a Pn_OUT.Px (x = 0-15) bit position delivers a low level at the corresponding output pin. A high level is output when the corresponding bit is written with a 1. Note that the bits of Pn_OUT.Px can be individually set or cleared by writing appropriate values into the port output modification set register Pn_OMSR or port output modification clear register Pn_OMCR, respectively. The Pn_OUT.Px bits can also be set, cleared or toggled with register Pn_OMR within the same write operation.

OUT Offset address: 020_H

Port n Output Register Reset values see: Table 2433

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
P15	P14	P13	P12	P11	P10	P9	P8	P7	P6	P5	P4	P3	P2	P1	P0
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
Px (x=0-15)	x	rwh	**Input Bit x** This bit indicates the level at the output pin Pn.x. 0_B The output level of Pn.x is 0. 1_B The output level of Pn.x is 1.
0	31:16	r	**Reserved** Read as 0; should be written with 0.

Table 2433 Reset values of OUT

Reset type	Reset value	Note
Application Reset	0000 0000_H	
Kernel Reset	0000 0000_H	

47.4.12 Port n Input Register

The logic level of a GPIO pin can be read via the read-only port input register Pn_IN.Reading the Pn_IN register always returns the current logical value at the GPIO pin independently whether the pin is selected as input or output.

IN Offset address: 024_H

Port n Input Register Reset values see: Table 2434

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
P15	P14	P13	P12	P11	P10	P9	P8	P7	P6	P5	P4	P3	P2	P1	P0
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
Px (x=0-15)	x	rh	**Input Bit x** This bit indicates the level at the input pin Pn.x. 0_B The input level of Pn.x is 0. 1_B The input level of Pn.x is 1.
0	31:16	r	**Reserved** Read as 0.

Table 2434 Reset values of IN

Reset type	Reset value	Note
Application Reset	0000 0000$_H$	
Kernel Reset	0000 0000$_H$	

47.4.13 Port n HWSEL status Register

The register HWSELSTAT allows reading back the status of the HW selection lines of all the pins.

This allows the SW to determinate if the Pin state is driven by the SW selection or if a HW module like e.g. LETHx or CSRM took over the control over any signal in the port.

Note: An active Emergency stop request is not reflected in this status register

HWSELSTAT Offset address: 028$_H$

Port n HWSEL status Register Reset values see: Table 2435

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STAT 15	STAT 14	STAT 13	STAT 12	STAT 11	STAT 10	STAT 9	STAT 8	STAT 7	STAT 6	STAT 5	STAT 4	STAT 3	STAT 2	STAT 1	STAT 0
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
STATx (x=0-15)	x	rh	**Status of HWSEL line for Pin x** Status of HWSEL line x
0	31:16	r	**Reserved** Read as 0.

Table 2435 Reset values of HWSELSTAT

Reset type	Reset value	Note
Application Reset	0000 0000$_H$	
Kernel Reset	0000 0000$_H$	

47.4.14 Port n Pin Function Decision Control Register

The pad structure of the GPIO lines offers the possibility to disable/enable port pad, select digital input or analog ADC input functionalities. Note that Class S pads have different characteristics than other digital input pads. For analog inputs, PDISx shall be set to 1 to disable the Schmitt trigger input buffer and digital output driver (if exists). Otherwise the specified accuracy of analog functions on overlaid pins (see TMADC accuracy parameters and their footnotes in datasheet) cannot be applied. For the ADC diagnostic features “PDD” however the corresponding PDISCx needs to be 0 to allow activation of their pull resistors. This feature can be controlled by individual bits in the Pn_PDISC register, independently from input/output and pull-up/pull-down control functionality as programmed in the Pn_DRVCFGx register. One Pn_PDISC register is assigned to each port.

***Note**: After reset, all Px_PDISC registers have the reset value of 0000 0000$_H$. The startup software disables only the pads with digital input/output functionality which are available in that package by writing the corresponding bits of not available pads to one. P40_PDISC and P41_PDISC are configured by the SSW for analog input function (kept disabled). The documented reset value shows the value in the largest package.*

PDISC Offset address: 02C$_H$

Port n Pin Function Decision Control Register Reset values see: Table 2436

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PDIS 15	PDIS 14	PDIS 13	PDIS 12	PDIS 11	PDIS 10	PDIS 9	PDIS 8	PDIS 7	PDIS 6	PDIS 5	PDIS 4	PDIS 3	PDIS 2	PDIS 1	PDIS 0
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
PDISx (x=0-15)	x	rwh	**Pin Function Decision Control for Pin x** This bit selects the function of the port pad. 0_B Digital functionality of pad Pn.x is enabled. 1_B Digital functionality (including pull resistors) of pad Pn.x is disabled. Analog input function (where this is available) can be used.
0	31:16	r	**Reserved** Read as 0; should be written with 0.

Table 2436 Reset values of PDISC

Reset type	Reset value	Note
PowerOn Reset	0000 0000$_H$	
After Boot-FW Value	0000 0000$_H$	Initial value in largest package

47.4.15 PROT Register Safe Endinit (SE)

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner and in some PROT states, the secure master to update the PROT state as described in the PROT mechanism.

PROTSE controls lock / unlock of the local Safe Endinit (SE) protected control registers.

A detailed description of the PROT mechanism and the underlying state machine can be found in chapter Safe and Secure Lock of control registers (PROT).

Note: as long as MEN=1 TAG-ID, VM, PRS, MEN and M_P can only be changed by the master function M (as defined by TAG-ID, VM, PRS). Write accesses from other master functions do not change these bits / bit fields.

PROTSE Offset address: 030_H

PROT Register Safe Endinit [SE] Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OWEN	ODEF	TAGID						PRSEN	PRS			VMEN	VM		
w	rw	rw						rw	rw			rw	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												SWEN	STATE		
r												w	rwh		

Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
VM	18:16	rw	**Virtual Machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual Machine definition Enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition

(table continues...)

(continued)

Field	Bits	Type	Description
PRS	22:20	rw	**Protection Set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.
PRSEN	23	rw	**Protection Set definition Enable for PROT owner** 0_B **Disable**: PRS is not part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; should be written with 0.

Table 2437 Access mode restrictions of PROTSE sorted by descending priority

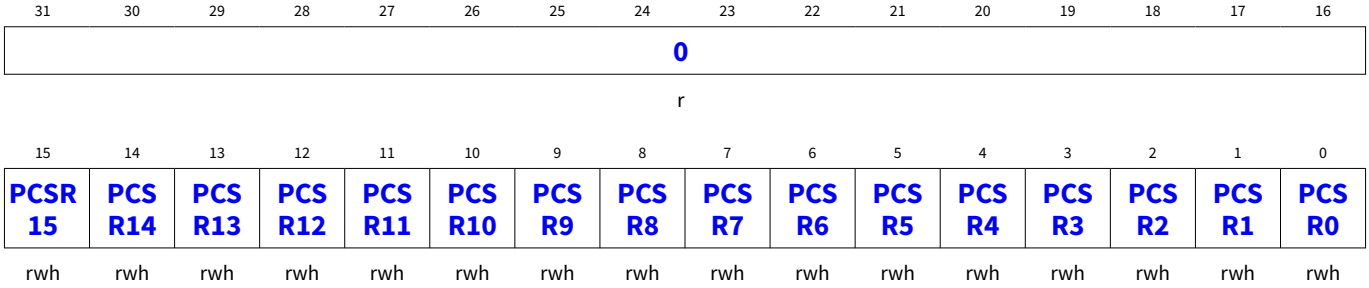
Mode name	Access mode		Description
write 1 to .**SWEN**	rwh	STATE	
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

47.4.16 Selection Register for Module x Port n Control

This register enables or disables a module x to have direct port control. Modules making use of this configuration feature are: SCR, xSPI, GETHx, LETHx and HRPWMx

PCSRSEL Offset address: 034_H

Selection Register for Module x Port n Control PowerOn Reset value: $0000\ 0000_H$



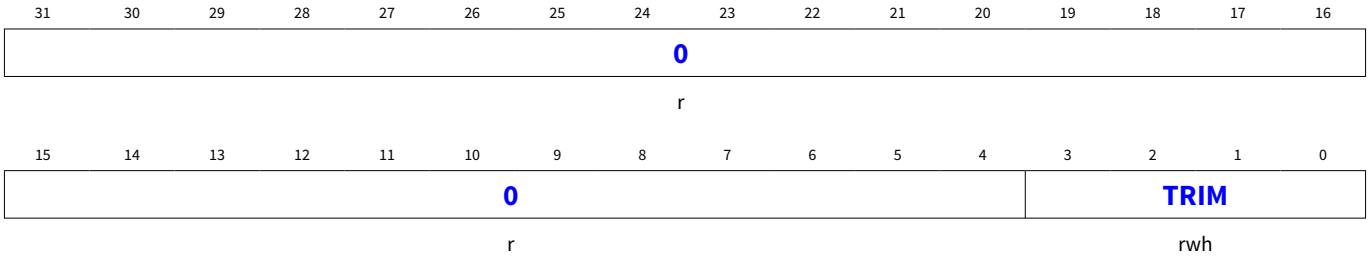
Field	Bits	Type	Description
PCSRx (x=0-15)	x	rwh	**Pad control selection x** This bit indicates whether a module x has control over pad Pn.x. Selection is valid only if the pad is shared with a particular module x and if the module is enabled. 0_B Main system has control over pad 1_B Control over pad is given to a particular module x
0	31:16	r	**Reserved** Read as 0; should be written with 0.

47.4.17 Pad Bandgap Trim Register

The BGTRIM register contains the trimming information of the bandgap located in the padframe.

BGTRIM Offset address: 038_H

Pad Bandgap Trim Register Reset values see: Table 2438



Field	Bits	Type	Description
TRIM	3:0	rwh	**Bandgap Trimming Bit** Trimming value for bandgap reference. 0_H corresponds to a value of 1.199V The trimming value is zero. 1_H corresponds to a value of 1.184V The trimming value is one. 2_H corresponds to a value of 1.168V The trimming value is two. 3_H corresponds to a value of 1.153V The trimming value is three. 4_H corresponds to a value of 1.139V The trimming value is four. 5_H corresponds to a value of 1.125V The trimming value is five. 6_H corresponds to a value of 1.112V The trimming value is six. 7_H corresponds to a value of 1.098V The trimming value is seven. 8_H corresponds to a value of 1.340V, The trimming value is eight. 9_H corresponds to a value of 1.321V The trimming value is nine. A_H corresponds to a value of 1.302V The trimming value is ten. B_H corresponds to a value of 1.283V The trimming value is eleven. C_H corresponds to a value of 1.266V The trimming value is twelve. D_H corresponds to a value of 1.248V The trimming value is thirteen. E_H corresponds to a value of 1.231V The trimming value is fourteen. F_H corresponds to a value of 1.215V The trimming value is fifteen.
0	31:4	r	**Reserved** Read as 0; should be written with 0.

Table 2438 Reset values of BGTRIM

Reset type	Reset value	Note
PowerOn Reset	$0000\ 0000_H$	
After Boot-FW Value	$0000\ 000X_H$	Exact value depends on silicon characteristics

47.4.18 Port n Output Modification Register

The port output modification register contains control bits that make it possible to individually set, clear or toggle the logic state of a single port line by manipulating the output register.

OMR Offset address: $03C_H$

Port n Output Modification Register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PCL15	PCL14	PCL13	PCL12	PCL11	PCL10	PCL9	PCL8	PCL7	PCL6	PCL5	PCL4	PCL3	PCL2	PCL1	PCL0
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PS15	PS14	PS13	PS12	PS11	PS10	PS9	PS8	PS7	PS6	PS5	PS4	PS3	PS2	PS1	PS0
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

Field	Bits	Type	Description
PSx (x=0-15)	x	w	**Set Bit x** Setting this bit will set or toggle the corresponding bit in the port output register Pn_OUT. Read as 0. 0_B No operation 1_B Sets or toggles Pn_OUT.Px.
PCLx (x=0-15)	x+16	w	**Clear Bit x** Setting this bit will clear or toggle the corresponding bit in the port output register Pn_OUT. Read as 0. 0_B No operation 1_B Clears or toggles Pn_OUT.Px.

***Note**: Register Pn_OMR is virtual and does not contain any flip-flop. A read action delivers the value of 0. One 8 or 16-bits write behaves as a 32-bit write padded with zeros.*

- PCL.x/PSx = 0/0 = Bit Pn_OUTPx is not changed.
- PCL.x/PSx = 0/1 = Bit Pn_OUTPx is set.
- PCL.x/PSx = 1/0 = Bit Pn_OUTPx is reset.
- PCL.x/PSx = 1/1 = Bit Pn_OUTPx is toggled.

47.4.19 Port n Output Modification Clear Register

The port output modification clear register contains control bits that make it possible to individually clear the logic state of a single port line by manipulating the output register.

***Note**: Register Pn_OMCR is virtual and does not contain any flip-flop. A read action delivers the value of 0. One 8 or 16-bits write behaves as a 32-bit write padded with zeros.*

OMCR Offset address: 040_H

Port n Output Modification Clear Register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PCL15	PCL14	PCL13	PCL12	PCL11	PCL10	PCL9	PCL8	PCL7	PCL6	PCL5	PCL4	PCL3	PCL2	PCL1	PCL0
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															
r															

Field	Bits	Type	Description
PCLx (x=0-15)	x+16	w	**Clear Bit x** Setting this bit will clear the corresponding bit in the port output register Pn_OUT. Read as 0. 0_B No operation 1_B Clears Pn_OUT.Px.
0	15:0	r	**Reserved** Read as 0; should be written with 0

47.4.20 Port n Output Modification Set Register

The port output modification set register contains control bits that make it possible to individually set the logic state of a single port line by manipulating the output register.

***Note**: Register Pn_OMSR is virtual and does not contain any flip-flop. A read action delivers the value of 0. One 8 or 16-bits write behaves as a 32-bit write padded with zeros.*

OMSR Offset address: 044_H

Port n Output Modification Set Register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PS15	PS14	PS13	PS12	PS11	PS10	PS9	PS8	PS7	PS6	PS5	PS4	PS3	PS2	PS1	PS0
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

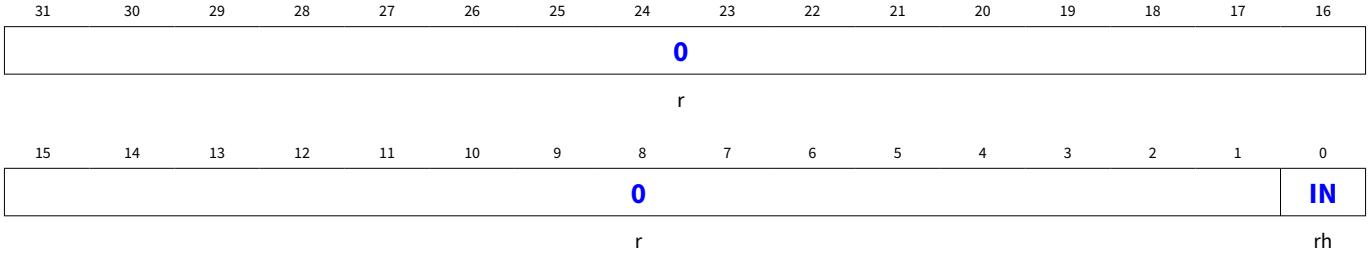
Field	Bits	Type	Description
PSx (x=0-15)	x	w	**Set Bit x** Setting this bit will set the corresponding bit in the port output register Pn_OUT. Read as 0. 0_B No operation 1_B Sets Pn_OUT.Px
0	31:16	r	**Reserved** Read as 0; should be written with 0.

47.4.21 Port n LVDS Pad Status Register x

The LVDS Pad Status Register reflects the status of the RX LVDS pads.

LPSRx (x=0-7) Offset address: 048_H+x*8

Port n LVDS Pad Status Register x Application Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
IN	0	rh	**Input value of LVDS receive pad** 0_B Status of LVDS input is 0. 1_B Status of LVDS input is 1.
0	31:1	r	**Reserved** Read as 0.

47.4.22 Port n LVDS Pad Control Register x

The LVDS Pad Control Register controls the RX or TX functions of the LVDS pads. For usage of RX pad, bit field [7:0] is applicable. If the register is used for TX pad, bit field [15:7] applies.

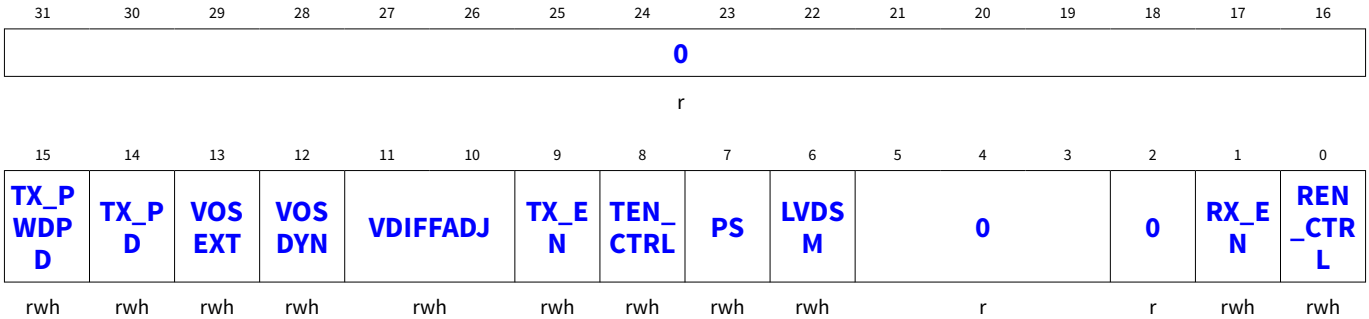
The sleep functionality of the LVDS pads is not controllable by this register. This is exclusively controlled by the HSCT module when this is connected.

The register x controls in general the pad pair 2*x and 2*x+1 of the port n.

Exceptionally when available the pad pair P14.9 and P14.10 is controlled by P14_LPCR5.

LPCRx (x=0-7) Offset address: $04C_H+x*8$

Port n LVDS Pad Control Register x Reset values see: Table 2439



Field	Bits	Type	Description
REN_CTRL	0	rwh	**LVDS RX_EN controller** The LVDS RX_EN control function can be selected from the Port (default) or HSCT module (where this is connected). 0_B Port controlled 1_B HSCT controlled (reserved where no HSCT is connected)
RX_EN	1	rwh	**Enable Receive LVDS** Enable the receive LVDS / disable CMOS path. If this bit is set to 0 – no transfer from the LVDS sender can be received and the receiver LVDS is in low power state. 0_B disable LVDS / enable CMOS mode (reserved for pads without CMOS input stage) 1_B enable LVDS / disable CMOS mode
LVDSM	6	rwh	**LVDS-M Mode** Selects increased frequency mode “LVDS-M” of the receiver. This mode increases the static current of the RX pad. The max data rate is increased to 700 Mbps (350 MHz). 0_B LVDS - Normal Mode 1_B LVDS - Fast Mode
PS	7	rwh	**Pad Supply Selection** Selects between 5V or 3.3V supply on V_{EXT} for the pad-pair. Used in RX and TX pads! 0_B 3.3V supply 1_B 5V supply
TEN_CTRL	8	rwh	**LVDS TX_EN controller** The LVDS TX_EN control function can be selected from the Port (default) or HSCT module (where this is connected). 0_B Port controlled 1_B HSCT controlled (reserved where no HSCT is connected)
TX_EN	9	rwh	**Enable Transmit LVDS** Enable the transmit LVDS / disable CMOS path. If this bit is set to 0 - no transfer on LVDS data path can be initiated and the LVDS driver is disabled (powered down). 0_B disable LVDS / enable CMOS mode 1_B enable LVDS / disable CMOS mode
VDIFFADJ	11:10	rwh	**LVDS Output Amplitude Tuning** With these two configuration bits the LVDS output current/amplitude can be adjusted. The voltage swing depending on VDIFFADJ setting is documented in the Data Sheet, see parameter V_{OD} .
VOSDYN	12	rwh	**Tune Bit of VOS Control Loop Static/Dynamic** Tune bit to change V_{OS} control loop between static and dynamic mode. Don’t change reset value.

(table continues...)

(continued)

Field	Bits	Type	Description
VOSEXT	13	rwh	**Tune Bit of VOS Control Loop Internal/External** Tune bit to change V_{OS} control loop. Don't change reset value.
TX_PD	14	rwh	**LVDS Power Down** Unused in this device. LVDS disabled by TX_EN means power down. 0_B LVDS power on 1_B LVDS power down (default)
TX_PWDPD	15	rwh	**Enable TX Power down pull down.** This function disables or enables the LVDS pull down resistor. The application code must disable TX power down pull down resistor with a power up. With a LVDS Power Down configuration the pull down function must be enabled, if required. 0_B disabled TX Power down pull down resistor. 1_B enabled TX Power down pull down resistor.
0	2, 5:3, 31:16	r	**Reserved** Read as 0; should be written with 0

Table 2439 Reset values of LPCRx (x=0-7)

Reset type	Reset value	Note
PowerOn Reset	0000 5480_H	
After Boot-FW Value	0000 0000 0000 0000 0101 XX00 10XX $X000_B$	Initial value depends on RX/TX and trimming

47.4.23 Write access enable register A

Each group of 4 Registers controls one pin access group OR Pin group y

- Pn_ACCGRPy_RDA/B [0..NrOfACCGRPs] - 1 Bit per Master Tag regarding read access
- Pn_ACCGRPy_WRA/B [0..NrOfACCGRPs] - 1 Bit per Master Tag regarding write access
- Pn_ACCGRPy_VM [0..NrOfACCGRPs] - 2 Bits per VMID to controll read and write access
- Pn_ACCGRPy_PRS [0..NrOfACCGRPs] - Access allowance based on PRS ID

This register defines which master master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

ACCGRPy_WRA (y=0-7) Offset address: 090_H+y*20_H

Write access enable register A Application Reset value: $1000\ 0003_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Write access enable for Master TAG-ID q** This bit enables write access to the access protected region for transactions with the Master TAG ID q 0_B Disabled for write access 1_B Enabled for write access

47.4.24 Write access enable register B

Each group of 4 Registers controls one pin access group OR Pin group y

- Pn_ACCGRPy_RDA/B [0..NrOfACCGRPs] - 1 Bit per Master Tag regarding read access
- Pn_ACCGRPy_WRA/B [0..NrOfACCGRPs] - 1 Bit per Master Tag regarding write access
- Pn_ACCGRPy_VM [0..NrOfACCGRPs] - 2 Bits per VMID to controll read and write access
- Pn_ACCGRPy_PRS [0..NrOfACCGRPs] - Access allowance based on PRS ID

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for write accesses to the access protected region.

ACCGRPy_WRB (y=0-7) Offset address: 094_H+y*20_H

Write access enable register B Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Write access enable for Master TAG-ID q** This bit enables write access to the access protected region for transactions with the Master TAG ID q 0_B Disabled for write access 1_B Enabled for write access
0	31:8	r	**Reserved** Read as 0; should be written with 0.

47.4.25 Read access enable register A

Each group of 4 Registers controls one pin access group OR Pin group y

- Pn_ACCGRPy_RDA/B [0..NrOfACCGRPs] - 1 Bit per Master Tag regarding read access
- Pn_ACCGRPy_WRA/B [0..NrOfACCGRPs] - 1 Bit per Master Tag regarding write access
- Pn_ACCGRPy_VM [0..NrOfACCGRPs] - 2 Bits per VMID to controll read and write access
- Pn_ACCGRPy_PRS [0..NrOfACCGRPs] - Access allowance based on PRS ID

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

ACCGRPy_RDA (y=0-7) Offset address: 098_H+y*20_H

Read access enable register A Application Reset value: FFFF FFFF$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN09	EN08	EN07	EN06	EN05	EN04	EN03	EN02	EN01	EN00
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=00-31)	q	rw	**Read access enable for Master TAG-ID q** This bit enables read access to the access protected region for transactions with the Master TAG ID q 0_B Disabled for read access 1_B Enabled for read access

47.4.26 Read access enable register B

Each group of 4 Registers controls one pin access group OR Pin group y

- Pn_ACCGRPy_RDA/B [0..NrOfACCGRPs] - 1 Bit per Master Tag regarding read access
- Pn_ACCGRPy_WRA/B [0..NrOfACCGRPs] - 1 Bit per Master Tag regarding write access
- Pn_ACCGRPy_VM [0..NrOfACCGRPs] - 2 Bits per VMID to controll read and write access
- Pn_ACCGRPy_PRS [0..NrOfACCGRPs] - Access allowance based on PRS ID

This register defines which master agent function TAG identifier (TAG-ID) encodings are enabled or disabled for read accesses to the access protected region.

ACCGRPy_RDB (y=0-7) Offset address: $09C_H+y*20_H$

Read access enable register B Application Reset value: 0000 00FF$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EN39	EN38	EN37	EN36	EN35	EN34	EN33	EN32
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENq (q=32-39)	q-32	rw	**Read access enable for Master TAG-ID q** This bit enables read access to the access protected region for transactions with the Master TAG ID q 0_B Disabled for read access 1_B Enabled for read access
0	31:8	r	**Reserved** Read as 0; should be written with 0.

47.4.27 VM access enable register

Each group of 4 Registers controls one pin access group OR Pin group y

- Pn_ACCGRPy_RDA/B [0..NrOfACCGRPs] - 1 Bit per Master Tag regarding read access
- Pn_ACCGRPy_WRA/B [0..NrOfACCGRPs] - 1 Bit per Master Tag regarding write access
- Pn_ACCGRPy_VM [0..NrOfACCGRPs] - 2 Bits per VMID to controll read and write access
- Pn_ACCGRPy_PRS [0..NrOfACCGRPs] - Access allowance based on PRS ID

This register defines which virtual machine encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each VMx.

ACCGRPy_VM (y=0-7) Offset address: $0A0_H+y*20_H$

VM access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for VM ID q** This bit enables read access to the access protected region for transactions with the VM ID q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for VM ID q** This bit enables write access to the access protected region for transactions with the VM ID q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

47.4.28 PRS access enable register

Each group of 4 Registers controls one pin access group OR Pin group y

- Pn_ACCGRPy_RDA/B [0..NrOfACCGRPs] - 1 Bit per Master Tag regarding read access
- Pn_ACCGRPy_WRA/B [0..NrOfACCGRPs] - 1 Bit per Master Tag regarding write access
- Pn_ACCGRPy_VM [0..NrOfACCGRPs] - 2 Bits per VMID to controll read and write access
- Pn_ACCGRPy_PRS [0..NrOfACCGRPs] - Access allowance based on PRS ID

This register defines which protection register set encodings are enabled or disabled for accesses to the access protected region. There is one RDx and WRx bit for each PRSx.

ACCGRPy_PRS (y=0-7) Offset address: $0A4_H+y*20_H$

PRS access enable register Application Reset value: $00FF\ 00FF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								WR07	WR06	WR05	WR04	WR03	WR02	WR01	WR00
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00
r								rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RDq (q=00-07)	q	rw	**Read access enable for PRS q** This bit enables read access to the access protected region for transactions with the PRS q. 0_B Disabled for read access 1_B Enabled for read access
WRq (q=00-07)	q+16	rw	**Write access enable for PRS q** This bit enables write access to the access protected region for transactions with the PRS q. 0_B Disabled for write access 1_B Enabled for write access
0	15:8, 31:24	r	**Reserved** Read as 0; should be written with 0.

47.4.29 PROT Register Endinit

Each group of 4 Registers controls one pin access group OR Pin group y

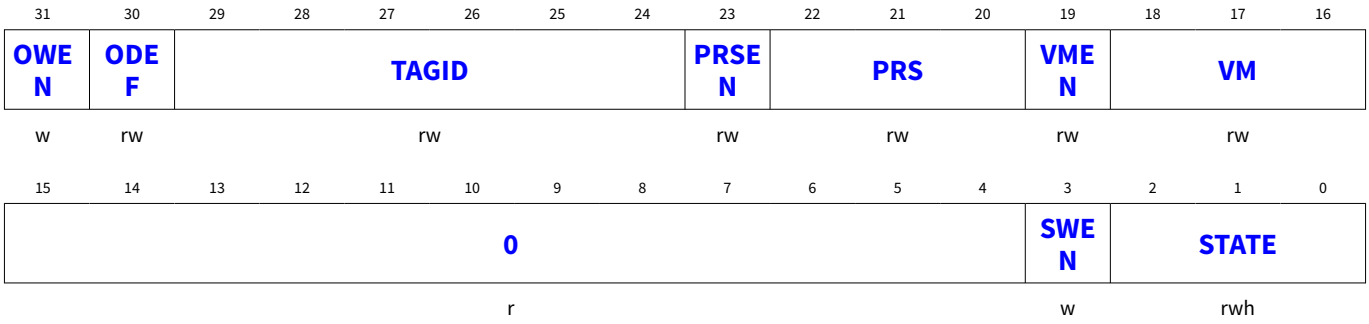
- Pn_ACCGRPy_RDA/B [0..NrOfACCGRPs] - 1 Bit per Master Tag regarding read access
- Pn_ACCGRPy_WRA/B [0..NrOfACCGRPs] - 1 Bit per Master Tag regarding write access
- Pn_ACCGRPy_VM [0..NrOfACCGRPs] - 2 Bits per VMID to controll read and write access
- Pn_ACCGRPy_PRS [0..NrOfACCGRPs] - Access allowance based on PRS ID

The resource protection register allows the definition of a specific PROT owner, and allows the PROT owner and in some PROT states, the secure master to update the PROT state as described in the PROT mechanism.

The PROTE register controls lock / unlock of the local Endinit (E) protected control registers.

ACCGRPy_PROTE (y=0-7) Offset address: $0AC_H+y*20_H$

PROT Register Endinit Application Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
STATE	2:0	rwh	**Resource protection state** Returns the current PROT state when read. Can be written by the PROT owner or CS master to modify the current PROT state if the state transition is valid in the PROT state diagram. 000_B Init (unlocked state) 001_B Config (unlocked state) 010_B ConfigSec (unlocked state) 011_B CheckSec (locked state) 100_B Run (locked state) 101_B RunSec (locked state) 110_B RunLock (locked state) 111_B RunLock (locked state)
SWEN	3	w	**State write enable** Write-enable for the STATE field. Reads always return 0. 0_B **Disable**: STATE field is not updated 1_B **Enable**: STATE field is updated by the write
VM	18:16	rw	**Virtual Machine definition for PROT owner** Defines the VM of the PROT owner. This field is ignored if VMEN=0_B or ODEF=0_B.
VMEN	19	rw	**Virtual Machine definition Enable for PROT owner** 0_B **Disable**: VM is not part of PROT owner definition 1_B **Enable**: VM is part of PROT owner definition
PRS	22:20	rw	**Protection Set definition for PROT owner** Defines the PRS of the PROT owner. This field is ignored if PRSEN=0_B or ODEF=0_B.
PRSEN	23	rw	**Protection Set definition Enable for PROT owner** 0_B **Disable**: PRS is not part of PROT owner definition 1_B **Enable**: PRS is part of PROT owner definition
TAGID	29:24	rw	**TAG ID definition for PROT owner** Defines the TAGID of the PROT owner. This field is ignored if ODEF=0_B.
ODEF	30	rw	**Enable for PROT owner definition** 0_B **Undefined**: Any master can act as PROT owner (TAGID, VMEN, VM, PRSEN, PRS are ignored) 1_B **Defined**: TAGID, VMEN, VM, PRSEN, PRS define the PROT owner

(table continues...)

(continued)

Field	Bits	Type	Description
OWEN	31	w	**Owner write enable** Write-enable for the owner fields. Reads always return 0. 0_B **Disable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are not updated 1_B **Enable**: PROT owner fields (ODEF, TAGID, VMEN, VM, PRSEN, PRS) are updated by the write
0	15:4	r	**Reserved** Read as 0; should be written with 0.

Table 2440 Access mode restrictions of ACCGRPy_PROTE (y=0-7) sorted by descending priority

Mode name	Access mode		Description
write 1 to .**SWEN**	rwh	STATE	
write 1 to .**OWEN**	rw	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
(default)	r	ODEF, PRS, PRSEN, TAGID, VM, VMEN	
	rh	STATE	

47.4.30 Port n GPIO Control Register for pin p

GPIO level control

If a Pin was configured in DRVCFG to be a general purpose Input or Output this register may be used by Application Software to request a pin level change.

When the Set and Clear Bits are set in the same write cycle to 1, the pin level request will toggle to the inverted value.

PADCFGp_GPIO (p=0-15) Offset address: 300_H+p*10_H

Port n GPIO Control Register for pin p Reset values see: Table 2441

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0							IN	0				CLR	SET	0	OUT
r							rh	r				w	w	r	rh

Field	Bits	Type	Description
OUT	0	rh	**Output value** The port output field reports the requested value of a GPIO pin when it is selected by Pn_DRVVCFGx as output. A 0 in Pn_GPIOx.OUT (x = 0-15) bit equals a low level requested at the corresponding output pin. A high level corresponds to the bit read as 1. Note that the bits Pn_DRVVCFGx.OUT can be individually set or cleared by writing appropriate values into the port output modification set register Pn_OMSR or port output modification clear register Pn_OMCR, respectively. The bits can also be used to set, cleared or toggled with register Pn_OMR within the same write operation. The OUT bit in this register is explicitly set to be read only in order to avoid clashes between this bit and the Set/Clear bits in the same register. 0_B Set the input level of Pn.x to the low state. 1_B Set the level of Pn.x to the high state.
SET	2	w	**Set** Writing a 1 to the SET field will set the Pin to a Voltage level equivalent to the High state
CLR	3	w	**Clear** Writing a 1 to the CLR field will set the Pin to a Voltage level equivalent to the Low state
IN	8	rh	**Read back Pin value** The logic level of a pin can be read via the read-only port input register field Pn_PORTCTRL.IN. Reading the IN field always returns the current logical value at the pin independently from the pin direction or if the pin was set to be driven by GPIO, ALTSEL or HW_SEL. 0_B The input level of Pn.x is 0. 1_B The input level of Pn.x is 1.
0	1, 7:4, 31:9	r	**Reserved** Read as 0; should be written with 0

Table 2441 Reset values of PADCFGp_GPIO (p=0-15)

Reset type	Reset value	Note
Application Reset	$0000\ 0000_H$	
Kernel Reset	$0000\ 0000_H$	

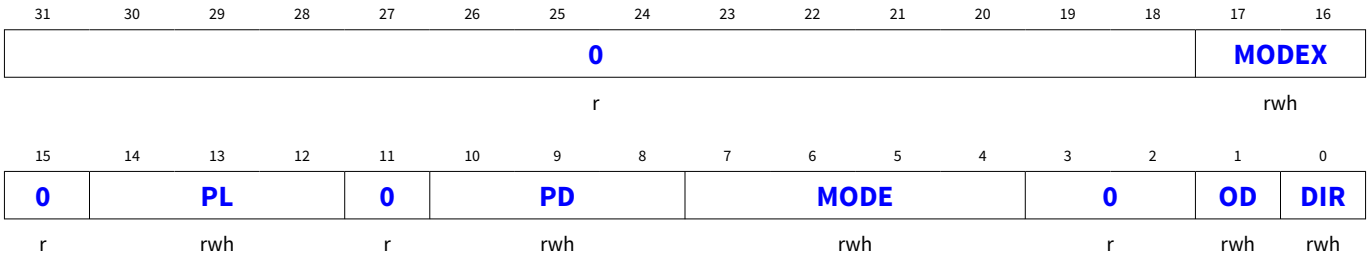
47.4.31 Port n Pad configuration Control Register for padp

Driver Configuration

The Pin Driver Configuration register contains the bit fields needed to set up the required pin configuration such as signal source and driver type selection.

PADCFGp_DRVCFG (p=0-15) Offset address: 304_H+p*10_H

Port n Pad configuration Control Register for padp Reset values see: Table 2442



Field	Bits	Type	Description
DIR	0	rwh	**Pad Signal direction** The DIR field defines the direction of the Port Pad in GPIO mode 0_B **IN**: Pad confligured as Input 1_B **OUT**: Pad confligured as Output
OD	1	rwh	**Output Buffer configuration** The bitfield OD defines the Output driver configuration when the Pin is configured in output mode. The Pad may be configured to Open Drain or Push/Pull driver types. 0_B **PP**: Push Pull 1_B **OD**: Open Drain
MODE	7:4	rwh	**Input configuration or Alternative Output signal selection** If the Pad is configured as Input, the MODE field determinates the Input mode regarding pure tristate mode versus a connected pull device. Note: If the direction is set to input only the lower 2 bits of the Mode field will be evaluated. If the Pad is configured as Output, the MODE field selects the number of the alternative output function driving the value towards the pin number n. Values higher than 0 do select the alternative output function matching to the absolute number of the value (e.g. MODE=1 selects alternative function number 1) When set to 0 the Pin acts as GPIO port. There the Pin state is SW controllable via the register Pn_GPIO 0_H **GPIO_TRI**: Pad configured in Tristate input mode or GPIO output mode 1_H **ALT01_PD**: Pad connected to alternative function 1 or input pull-down device 2_H **ALT02_PU**: Pad connected to alternative function 2 or input pull-up device 3_H **ALT03_TRI**: Pad connected to alternative function 3 or input tristate mode 4_H **4**: Alternative Output function 4 selected … F_H **15**: Alternative Output function 15 selected

(table continues...)

(continued)

Field	Bits	Type	Description
PD	10:8	rwh	**Pad drive mode selection** **Driver strength selection for Fast and Slow GPIO** **Driver strength selection bitfield (PD) decoding for Fast GPIO** [PD2, PD1, PD0] = [X, 0, 0]: Driver = strong, Edge = sharp (ss) [PD2, PD1, PD0] = [X, 0, 1]: Driver = strong, Edge = medium (sm) [PD2, PD1, PD0] = [X, 1, 0]: Driver = medium, Edge = medium (m) [PD2, PD1, PD0] = [X, 1, 1]: do not use **Driver strength selection bitfield (PD) decoding for Slow GPIO** [PD2, PD1, PD0] = [X, X, 0]: Driver = medium, Edge = sharp (sm) [PD2, PD1, PD0] = [X, X, 1]: Driver = medium, Edge = medium (m) **Driver strength selection bitfield (PD) decoding for HSFAST GPIO** In case the register PCSRSEL of the corresponding port slice is set to zero the decoding of the FAST GPIO applies In case the register PCSRSEL of the corresponding port slice is set to one the decoding of the configurable output impedances in the datasheet (section: HSFast x.x V GPIO characteristics) applies 000_B **PD_SS**: [PD2, PD1, PD0] = [0, 0, 0] Coding for the respective pad type see table above 001_B **PD_SM**: [PD2, PD1, PD0] = [0, 0, 1] Coding for the respective pad type see table above 010_B **PD_M**: [PD2, PD1, PD0] = [0, 1, 0] Coding for the respective pad type see table above 011_B **PD_DNU**: [PD2, PD1, PD0] = [0, 1, 1] do not use this combination 100_B **PD_SS**: [PD2, PD1, PD0] = [1, 0, 0] Coding for the respective pad type see table above 101_B **PD_SM**: [PD2, PD1, PD0] = [1, 0, 1] Coding for the respective pad type see table above 110_B **PD_M**: [PD2, PD1, PD0] = [1, 1, 0] Coding for the respective pad type see table above 111_B **PD_DNU**: [PD2, PD1, PD0] = [1, 1, 1] do not use this combination

(table continues...)

(continued)

Field	Bits	Type	Description
PL	14:12	rwh	**Pad level selection** **PLx Input Level control** [PL2, PL1, PL0] = [X, 0, X]: Automotive level "AL" [PL2, PL1, PL0] = [X, 1, 0]: TTL level for 5V pad supply. [PL2, PL1, PL0] = [X, 1, 1]: TTL level for 3.3V pad supply. [PL2, PL1, PL0] = [X, X, X]: do not use ***Note:*** *In case of P16.xx the PL0 bit is tight to zero. When PL1 is set to one TTL level for 3.3V pad supply is selected.* 000_B **PL_AL**: Automotive level "AL" Automotive level "AL" 001_B **PL_AL**: Automotive level "AL" Automotive level "AL" 010_B **PL_TTL5V**: TTL level for 5V pad supply TTL level for 5V pad supply 011_B **PL_TTL3_3V**: TTL level for 3.3V pad supply TTL level for 3.3V pad supply 100_B **PL_AL**: Automotive level "AL" Automotive level "AL" 101_B **PL_AL**: Automotive level "AL" Automotive level "AL" 110_B **PL_TTL5V**: TTL level for 5V pad supply TTL level for 5V pad supply 111_B **PL_TTL3_3V**: TTL level for 3.3V pad supply TTL level for 3.3V pad supply

(table continues...)

(continued)

Field	Bits	Type	Description
MODEX	17:16	rwh	**Configuration of output driver for HS_Fast pad (1.8V and 3.3V)** The MODEX bits are dedicated configuration bits for Port 16 only. The bit configuration is only valid if the PCSRSEL bit of the corresponding port slice is set to 0. In case Port16 is used for GPIO functionality the MODEX field has to be configured properly according to the applied voltage on the Port16 supply VDDHSIF. **Note:** In case of the usage of xSPI and RGMII function on Port16 the register DLL_CFG (bitfields PEN3VX and PMODE) inside the HSPHY functional block has to be configured properly. 00_B **HSF_DRV_1V8**: HSFast 1.8V xSPI / RGMII driver setting used Pad is configured for 1.8V mode (input buffer and output stage operating with 1.8V supply) . The active driver settings for this mode are the 1.8 V HSFast driver settings (controlled by bitfield PD [2:0]) which are also used for xSPI/RGMII operation.*For Port Slice P16.0 this setting is used to provide a differetial output signal on P16.0 and P16.1.* 01_B **HSF_DRV_1V8**: HS_Fast 1.8V xSPI / RGMII driver setting used Pad is configured for 1.8V mode (input buffer and output stage operating with 1.8V supply) . The active driver settings for this mode are the 1.8 V HSFast driver settings (controlled by bitfield PD [2:0]) which are also used for xSPI/RGMII operation.*For Port Slice P16.0 this setting is used to provide a single ended output signal on P16.0 and P16.1.* 10_B **GPIO_DRV_3V3**: HS_Fast 3.3V GPIO driver setting used Pad is configured for 3.3V mode (input buffer and output stage operating with 3.3V supply). The active driver settings for this mode are the HSFast - GPIO driver settings (controlled by bitfield PD [1:0]). 11_B **HSF_DRV_3V3**: HS_Fast 3.3V xSPI / RGMII driver setting used Pad is configured for 3.3V mode (input buffer and output stage operating with 3.3V supply). The active driver settings for this mode are the 3.3 V HSFast driver settings (controlled by bitfield PD [2:0]) which are also used for xSPI/RGMII operation.
0	3:2, 11, 15, 31:18	r	**Reserved** Read as 0; should be written with 0

Table 2442 Reset values of PADCFGp_DRVCFG (p=0-15)

Reset type	Reset value	Note
Application Reset	0000 0000 0000 0000 0000 0000 00X0 0000_B	Depending on HWCFG[6]: X:0 (tri-state mode) X:1 (input pull-up mode)
Kernel Reset	0000 0000 0000 0000 0000 0000 00X0 0000_B	Depending on HWCFG[6]: X:0 (tri-state mode) X:1 (input pull-up mode)

47.4.32 Port n Pad Access Control Register for padp

Access group selection and enable

The Pin Configuration register ACCEN contains the bit fields needed to set up the required access group settings.

PADCFGp_ACCEN (p=0-15) Offset address: 308_H+p*10_H

Port n Pad Access Control Register for padp Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0													GRP		
r													rw		

Field	Bits	Type	Description
GRP	2:0	rw	**Access Protection Group selection** Protection group selection The selected group in this field corresponds to the Access group block used to protect the accesses to the registers PADCFGp_GPIO and PADCFGp_DRVCFG. Note: If this bitfield is set to a non existing group the logic will block all accesses and may respond with a safety alert. 000_B **GRP0__GRP**: Port Access Group selected ... 111_B **GRP7__GRP**: Port Access Group selected
0	31:3	r	**Reserved** Read as 0; should be written with 0

47.4.33 Port n Pad Safety and Security Control Register for padp

Safety and Security Pin settings

The Pin Configuration register SAFSEC contains the safety and security relevant bits.

PADCFGp_SAFSEC (p=0-15) Offset address: $30C_H+p*10_H$

Port n Pad Safety and Security Control Register for padp Reset values see: Table 2443

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0										ESR_PD		0		ESR_PPL	ESR_EN
r										rwh		r		rwh	rwh

Field	Bits	Type	Description
ESR_EN	0	rwh	**Emergency Request Action Enable** Most GPIO lines have an emergency stop logic implemented (see Figure “PORTS block diagram” in the Family chapter). Each of these GPIO lines has its own emergency stop enable bit ENx that is located in the emergency stop field ESR of Port Pin n. If the emergency stop signal becomes active, one of two states can be selected. Exceptions for Emergency Stop Implementation The Emergency Stop function is available for all GPIO Ports with the following exceptions: • Not available for P20.2 (General Purpose Input/GPI only, overlayed with Testmode) • Not available for P40_x and P41_x (analog input ANx overlayed with GPI) • Not available for P21.2 (used as SMU_EXT_EMERGENCYSTOP_REQ pin). • Not available for dedicated I/O without General Purpose Output function (e.g ESRx, TMS, TCK) The Emergency Stop function can be overruled on the following GPIO Ports: • P00.x: Emergency Stop can be overruled by the TMADC. Overruling can be disabled via the control register P00_PCSR. • P14.0 and P14.1: Emergency Stop can be overruled in the DXCPL (DAP over CAN physical layer) mode. No Overruling in the DXCM (Debug over can message) mode • P21.6: Emergency Stop can be overruled in JTAG mode if this pin is used as TDI • P21.7: Emergency Stop can be overruled in JTAG or Three Pin DAP mode. • P33.0-7, P33.9-15 and P34.1: Emergency Stop can be overruled by the 8-Bit Standby Controller (SCR), if implemented. Overruling can be disabled via the control register P33_PCSR and P34_PCSR. On pins with LVDS TX pads the Emergency Stop affects only the CMOS driver not the LVDS driver. Thus only when LPCRx.TX_EN selects CMOS mode the output is switched off. When TX_EN selects LVDS mode the output is not switched off. 0_B **DIS**: Emergency stop function disabled. The output line remains connected (alternate function). 1_B **EN**: Emergency stop function enabled. The mapped output function is disconnected and the safe state is entered by switching to input function with internal pull-up connected or tri-state, depending on the configured value of the corresponding Port slice PADCFGp_SAFSEC.ESR_PD register.
ESR_PPL	1	rwh	**Emergency status Push-Pull level** Value driven when pin is put in push-pull setup by ESR_PD 0_B The onput level of Pn.x is 0. 1_B The onput level of Pn.x is 1.

(table continues...)

(continued)

Field	Bits	Type	Description
ESR_PD	5:4	rwh	**Emergency status Pad configuration** This Bitfield defines the PAD mode when ESR is active and ESR_EN = 1 00_B **PP**: PAD in Push-PULL configuration when ESR is active. Level defined by field ESR_PPL 01_B **TPU**: Tristate, weak pull-up 10_B **TPD**: Tristate, weak pull-down 11_B **RES**: Reserved - defaults to Tristate Pull Down
0	3:2, 31:6	r	**Reserved** Read as 0; should be written with 0

Table 2443 Reset values of PADCFGp_SAFSEC (p=0-15)

Reset type	Reset value	Note
Application Reset	$0000\ 0000_H$	(p=0)
Application Reset	$0000\ 0000_H$	(p=1)
Application Reset	$0000\ 0000_H$	(p=2)
Application Reset	$0000\ 0000_H$	(p=3)
Application Reset	$0000\ 0000_H$	(p=4)
Application Reset	$0000\ 0000_H$	(p=5)
Application Reset	$0000\ 0000_H$	(p=6)
Application Reset	$0000\ 0000_H$	(p=7)
Application Reset	$0000\ 0000_H$	(p=8)
Application Reset	$0000\ 0000_H$	(p=9)
Application Reset	$0000\ 0000_H$	(p=10)
Application Reset	$0000\ 0000_H$	(p=11)
Application Reset	$0000\ 0000_H$	(p=12)
Application Reset	$0000\ 0000_H$	(p=13)
Application Reset	$0000\ 0000_H$	(p=14)
Application Reset	$0000\ 0000_H$	(p=15)
Kernel Reset	$0000\ 0000_H$	(p=0)
Kernel Reset	$0000\ 0000_H$	(p=1)
Kernel Reset	$0000\ 0000_H$	(p=2)
Kernel Reset	$0000\ 0000_H$	(p=3)
Kernel Reset	$0000\ 0000_H$	(p=4)
Kernel Reset	$0000\ 0000_H$	(p=5)
Kernel Reset	$0000\ 0000_H$	(p=6)
Kernel Reset	$0000\ 0000_H$	(p=7)
Kernel Reset	$0000\ 0000_H$	(p=8)
Kernel Reset	$0000\ 0000_H$	(p=9)

(table continues...)

Table 2443 (continued) Reset values of PADCFGp_SAFSEC (p=0-15)

Reset type	Reset value	Note
Kernel Reset	0000 0000$_H$	(p=10)
Kernel Reset	0000 0000$_H$	(p=11)
Kernel Reset	0000 0000$_H$	(p=12)
Kernel Reset	0000 0000$_H$	(p=13)
Kernel Reset	0000 0000$_H$	(p=14)
Kernel Reset	0000 0000$_H$	(p=15)

47.4.34 Port n LDO Configuration and control register

ENCONF_RDDIAG Offset address: 190$_H$

Port n LDO Configuration and control register Reset values see: Table 2444

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			RESERVED						SH_VDD	SHOFF_PMS	SH_LDO_OFF_EN	SHHEN	SHORT_TIMER		ENABLE
r			r						rh	rh	rw	rw	rw		rwh

Field	Bits	Type	Description
ENABLE	0	rwh	**Main Enable for complete IP** 0_B **OFF**: LDO is disabled 1_B **ON**: LDO is enabled
SHORT_TIMER	2:1	rw	**Blanking Timer configuration for Short Detection** 00_B Corresponds to blanking time of 0 ms 01_B Corresponds to blanking time of 2 ms 10_B Corresponds to blanking time of 4 ms 11_B Corresponds to blanking time of 7 ms
SHHEN	3	rw	**Short Detection Enabling** 0_B **OFF**: Short detection disabled 1_B **ON**: Short detection enabled
SH_LDO_OFF_EN	4	rw	**Short Detection OFF LDO enabling (SHHEN = 1)** 0_B **OFF**: Switch off LDO on short detection disable 1_B **ON**: Switch off LDO on short detection enable

(table continues...)

(continued)

Field	Bits	Type	Description
SHOFF_PMS	5	rh	**Indication that LDO will be shut down in several clk cycles. This bit is set high only when SH_LDO_OFF_EN is 1 and persistent short to VDD detected** 0_B **NO_SD**: No shutdown procedure because of short started 1_B **SD**: Shutdown procedure because of short started
SH_VDD	6	rh	**Indication that short to VDD was detected** 0_B **NO_SH**: No short to VDD detected 1_B **SH**: Short to VDD detected
RESERVED	12:7	r	**Reserved for Future Usage**
0	31:13	r	**Reserved**

Table 2444 Reset values of ENCONF_RDDIAG

Reset type	Reset value	Note
PowerOn Reset	0000 0000_H	
After Boot-FW Value	0000 0000_H	

47.4.35 Port n LDO Trimming register

Trim configuration handled in safe way

TRIMCONF_PROT Offset address: $19C_H$

Port n LDO Trimming register Reset values see: Table 2445

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0										TRIM					
r										rw					

Field	Bits	Type	Description
TRIM	5:0	rw	**LDO Trimming** 00_H corresponds to a value of 1.692V $1D_H$ ... $1E_H$ corresponds to a value of 1.790V $1F_H$ corresponds to a value of 1.800V 20_H corresponds to a value of 1.810V 21_H ... $3F_H$ corresponds to a value of 1.908V
0	31:6	r	**Reserved**

Table 2445 Reset values of TRIMCONF_PROT

Reset type	Reset value	Note
PowerOn Reset	0000 001F$_H$	
After Boot-FW Value	0000 0000 0000 0000 0000 0000 00XX XXXX$_B$	

47.5 Debug information

This section is not applicable for the module being described.

47.6 References

This section is not applicable for the module being described.

47.7 PORTS revision history

Reference	Description of change(s)
Date range: 2024-08-17 to 2025-03-13	
Functional overview	• Changed drawing due to AE feedback. Block diagram figure was modified to show PCSRSEL functionality for particular modules like LETH0
Functional description	• Section "Padframe alarm trigger signals" moved to device specific functional description (TC49x-N and TC4Dx only)
Ports	• Typos fixed in several sub chapters
PADCFGp_DRVCFG	Changed bit field description of PD bit field

47.8 TC4Dx PORTS information

47.8.1 TC4Dx PORTS configuration

The following table shows the configuration of all TC4Dx port modules. Also the exceptions are listed:

Table 2446 Port implementation in TC4DxAA-step

Port module	Port slices	Pad supply domain	Available pins per port	Number of Access Groups	Configuration, exceptions
P00	16 slices	VDDEXT	P00.0 - P00.15	8	All port slices are available
P01	16 slices	VDDEXT	P01.0 - P01.15	8	All port slices are available
P02	16 slices	VDDEXT	P02.0 - P02.15	8	All port slices are available
P03	16 slices	VDDEXT	P03.0 - P03.15	8	All port slices are available
P04	10 slices	VDDEXT	P04.0 - P04.9	8	P04.10 - P04.15 not available
P10	15 slices	VDDEXT	P10.0 - P10.15	8	P10.12 not available
P13	16 slices	VDDEXT	P13.0 - P13.15	8	All port slices are available
P14	16 slices	VDDEXT	P14.0 - P14.15	8	All port slices are available
P15	15 slices	VDDEXT	P15.0 - P15.8 and P15.10 - P14.15	8	P15.9 not available
P16	15 slices	VDDHSIF	P16.0 - P16.14	8	P16.15 not available
P20	13 slices	VDDEXT	P20.0 - P20.3 and P20.6 - P20.14	8	P20.4, P20.5 and P20.15 are not available
P21	8 slices	VDDEXT	P21.0 - P21.7	4	P21.8 - P21.15 not available
P22	12 slices	VDDEXT	P22.0 - P22.11	8	P22.12 - P22.15 not available
P23	8 slices	VDDEXT	P23.0 - P23.7	4	P23.8 - P23.15 not available
P25	1 slices	VDDEXT	P25.1	4	P25.0, P25.2 - P25.15 not available;
P30	16 slices	VDDEXT	P30.0 - P30.15	8	All port slices are available
P31	2 slices	VDDEXT	P31.6 - P31.7	4	P31.0 - P31.5 and P31.8 - P31.15 not available

(table continues...)

Table 2446 (continued) Port implementation in TC4DxAA-step

Port module	Port slices	Pad supply domain	Available pins per port	Number of Access Groups	Configuration, exceptions
P32	5 slices	VDDEVRSB	P32.2 - P32.7	4	P32.0 - P32.1, P32.3 and P32.8 - P32.15 not available; Overlaid with SCR functions
P33	16 slices	VDDEVRSB	P33.0 - P33.15	8	Overlaid with SCR functions
P34	5 slices	VDDEVRSB	P34.1 - P34.5	4	P34.0 and P34.6 - P34.15 not available; Overlaid with SCR functions
P35	6 slices	VDDEVRSB	P35.0 - P35.5	4	P35.6 - P35.15 not available; Overlaid with SCR functions
P40	15 slices	VDDM	P40.0 - P40.14	8	P40.15 is not available. Registers dedicated to P40.15 functionality have no effect as the corresponding pad cell is not available; Analog inputs overlaid with SENT functionality

***Note**: Pin names with prefix PHYx_ do not provide any overlaid functionality*

47.8.2 TC4Dx PORTS features

Table 2447 Port implementation in TC4DxAA-step

Port module	Pins	Wake-up capable pins	Exceptions and special features
P00	16 pins	P00.0 - P00.15	Port 00 special features: • All pins of this port support the wake-up feature • GETH0_PX_MDIO (HW_OUT) on P00.0 • P00.1 - P00.12 are used as inputs for TMADC2
P01	16 pins	P01.0 - P01.15	Port 01 special features: • All pins of this port support the wake-up feature • P01.3 - P01.5 are used as inputs for TMADC2
P02	16 pins	P02.0 - P02.15	Port 02 special features: • All pins of this port support the wake-up feature • P02.11 is used as input for TMADC2
P03	16 pins	P03.0 - P03.15	Port 03 special features: • All pins of this port support the wake-up feature • For the usage of LETH0_P2_RMII output function on this port the corresponding PCSRSEL bit slice on P03.0, P03.8 and P03.9 has to be set to one and additionally the proper ALT function related to LETH0_P2_RMII output has to be selected

(table continues...)

Table 2447 (continued) Port implementation in TC4DxAA-step

Port module	Pins	Wake-up capable pins	Exceptions and special features
P04	10 pins	P04.0 - P04.9	Port 04 special features: • All pins of this port support the wake-up feature
P10	15 pins	P10.0 - P10.11 and P10.13 -P10.15	Port 10 special features: • All pins of this port support the wake-up feature • For the usage of LETH0_P2_RMII output function on this port the corresponding PCSRSEL bit slice on P10.10, P10.11 and P10.13 has to be set to one and additionally the proper ALT function related to LETH0_P2_RMII output has to be selected
P13	16 pins	P13.8 - P13.15	Port 13 exceptions and special features: • LVDS pins P13.0 - P13.7 do not support the wake-up feature • All other pins of this port support the wake-up feature • For the usage of LETH0_P3_RMII output function on this port the corresponding PCSRSEL bit slice on P13.8 - P13.10 has to be set to one and additionally the proper ALT function related to LETH0_P3_RMII output has to be selected
P14	16 pins	P14.0 - P14.8 and P14.11 - P14.15	Port 14 exceptions and special features: • LVDS pins P14.9 - P14.10 do not support the wake-up feature • All other pins of this port support the wake-up feature • For the usage of LETH0_P2_RMII output function on this port the corresponding PCSRSEL bit slice P14.7, P14.9 and P14.10 has to be set to one and additionally the proper ALT function related to LETH0_P2_RMII output has to be selected
P15	15 pins	P15.0 - P15.8 and P15.14 - P15.15	Port 15 exceptions and special features: • LVDS pins P15.10 - P15.13 do not support the wake-up feature • All other pins of this port support the wake-up feature
P16	15 pins	-	Port 16 exceptions and special features: • P16 does not support the wake-up feature • For usage of XSPI and GETH0_P0_RGMII output and input function on this port the corresponding PCSRSEL bit slice has to be set to one and ***Note**: Exception is P16.0 where PCSRSEL has to be 0 in order to get XSPI and GETH0_P0_RGMII output and input function (valid for 3 V operation only)*
P20	13 pins	P20.1, P20.12 - P20.14	Port 20 special features: • P20.0, P20.2 - P20.11 do not support the wake-up feature • All other pins of this port support the wake-up feature

(table continues...)

Table 2447 (continued) Port implementation in TC4DxAA-step

Port module	Pins	Wake-up capable pins	Exceptions and special features
P21	8 pins	P21.6 - P21.7	Port 21 exceptions and special features: • LVDS pins P21.0 - P21.5 do not support the wake-up feature • All other pins of this port support the wake-up feature
P22	12 pins	P22.4 - P22.11	Port 22 exceptions and special features: • LVDS pins P22.0 - P22.3 do not support the wake-up feature • All other pins of this port support the wake-up feature
P23	8 pins	P23.0 - P23.7	Port 23 special features: • All pins of this port support the wake-up feature
P25	1 pins	P25.1	Port 25 special features: • All pins of this port support the wake-up feature
P30	16 pins	P30.0 - P30.15	Port 30 special features: • All pins of this port support the wake-up feature • For the usage of LETH0_P1_RMII output function on this port the corresponding PCSRSEL bit slice on P30.9 - P30.11 has to be set to one and additionally the proper ALT function related to LETH0_P1_RMII output has to be selected
P31	2 pins	P31.6 - P31.7	Port 31 special features: • All pins of this port support the wake-up feature
P32	5 pins	P32.4 - P32.7	Port 32 exceptions and special features: • P32.2 and P32.5 do not support the wake-up feature as they are SMU FSP pins • All other pins of this port support the wake-up feature
P33	16 pins	P33.8, P33.11 - P33.15	Port 33 special features: • P33.0 - P33.7, P33.9 and P33.10 do not support the wake-up feature • All other pins of this port support the wake-up feature
P34	5 pins	P34.1 - P34.5	Port 34 special features: • All pins of this port support the wake-up feature
P35	6 pins	P35.0 - P35.5	Port 35 special features: • All pins of this port support the wake-up feature
P40	15 pins	-	Port 40 special features: • All pins have the SENT feature mapped • Port 40 is not affected by the HWCFG[6] configuration option during reset

47.8.3 TC4Dx PORTS functional description

47.8.3.1 Padframe alarm trigger signals

The pads of the device are combined in a padframe which is split over all four sides (Top (where A1 ball is located), Bottom, Left and Right) of the device. All four parts of the padframe have their own reference voltage generation which is required for a proper pad functionality. If one of the reference voltage generators is not working correctly (reference voltage is too low) an alarm trigger to the SMU is issued. The padframe alarm triggers are grouped in the system trigger alarms:

- ALMSF8 [11] indicates an issue in the reference voltage generation of the bottom part of the padframe
- ALMSF8 [12] indicates an issue in the reference voltage generation of the right part of the padframe
- ALMSF8 [13] indicates an issue in the reference voltage generation of the top part of the padframe
- ALMSF8 [14] indicates an issue in the reference voltage generation of the left part of the padframe

In case a padframe alarm is issued it is recommended to check the pad functionality of the device.

47.8.4 TC4Dx PORTS registers

47.8.4.1 Register address space - Pn

Table 2448 Registers address space - Pn

Module	Base address	End address	Note
P00	$F003A000_H$	$F003A3FF_H$	SPB bus slave interface
P01	$F003A400_H$	$F003A7FF_H$	SPB bus slave interface
P02	$F003A800_H$	$F003ABFF_H$	SPB bus slave interface
P03	$F003AC00_H$	$F003AFFF_H$	SPB bus slave interface
P04	$F003B000_H$	$F003B3FF_H$	SPB bus slave interface
P10	$F003C800_H$	$F003CBFF_H$	SPB bus slave interface
P13	$F003D400_H$	$F003D7FF_H$	SPB bus slave interface
P14	$F003D800_H$	$F003DBFF_H$	SPB bus slave interface
P15	$F003DC00_H$	$F003DFFF_H$	SPB bus slave interface
P16	$F003E000_H$	$F003E3FF_H$	SPB bus slave interface
P20	$F003F000_H$	$F003F3FF_H$	SPB bus slave interface
P21	$F003F400_H$	$F003F7FF_H$	SPB bus slave interface
P22	$F003F800_H$	$F003FBFF_H$	SPB bus slave interface
P23	$F003FC00_H$	$F003FFFF_H$	SPB bus slave interface
P25	$F0040400_H$	$F00407FF_H$	SPB bus slave interface
P30	$F0041800_H$	$F0041BFF_H$	SPB bus slave interface
P31	$F0041C00_H$	$F0041FFF_H$	SPB bus slave interface
P32	$F0042000_H$	$F00423FF_H$	SPB bus slave interface
P33	$F0042400_H$	$F00427FF_H$	SPB bus slave interface
P34	$F0042800_H$	$F0042BFF_H$	SPB bus slave interface
P35	$F0042C00_H$	$F0042FFF_H$	SPB bus slave interface
P40	$F0044000_H$	$F00443FF_H$	SPB bus slave interface

47.8.4.2 Register overview - access mode glossary

Table 2449 Register overview - access mode glossary

Keyword	Description
Ey (y=0-3)	Access protection using PROT register P00_ACCGRPy_PROTE or P01_ACCGRPy_PROTE or P02_ACCGRPy_PROTE or P03_ACCGRPy_PROTE or P04_ACCGRPy_PROTE or P10_ACCGRPy_PROTE or P13_ACCGRPy_PROTE or P14_ACCGRPy_PROTE or P15_ACCGRPy_PROTE or P16_ACCGRPy_PROTE or P20_ACCGRPy_PROTE or P21_ACCGRPy_PROTE or P22_ACCGRPy_PROTE or P23_ACCGRPy_PROTE or P25_ACCGRPy_PROTE or P30_ACCGRPy_PROTE or P31_ACCGRPy_PROTE or P32_ACCGRPy_PROTE or P33_ACCGRPy_PROTE or P34_ACCGRPy_PROTE or P35_ACCGRPy_PROTE or P40_ACCGRPy_PROTE .
Ey (y=4-7)	Access protection using PROT register P00_ACCGRPy_PROTE or P01_ACCGRPy_PROTE or P02_ACCGRPy_PROTE or P03_ACCGRPy_PROTE or P04_ACCGRPy_PROTE or P10_ACCGRPy_PROTE or P13_ACCGRPy_PROTE or P14_ACCGRPy_PROTE or P15_ACCGRPy_PROTE or P16_ACCGRPy_PROTE or P20_ACCGRPy_PROTE or P22_ACCGRPy_PROTE or P30_ACCGRPy_PROTE or P33_ACCGRPy_PROTE or P40_ACCGRPy_PROTE .
SE	Access protection using PROT register P00_PROTSE or P01_PROTSE or P02_PROTSE or P03_PROTSE or P04_PROTSE or P10_PROTSE or P13_PROTSE or P14_PROTSE or P15_PROTSE or P16_PROTSE or P20_PROTSE or P21_PROTSE or P22_PROTSE or P23_PROTSE or P25_PROTSE or P30_PROTSE or P31_PROTSE or P32_PROTSE or P33_PROTSE or P34_PROTSE or P35_PROTSE or P40_PROTSE .

(table continues...)

Table 2449 (continued) Register overview - access mode glossary

Keyword	Description
APU-PGy (y=0-3)	Protection group consisting of registers P00_ACCGRPy_WRA , P00_ACCGRPy_WRB , P00_ACCGRPy_RDA , P00_ACCGRPy_RDB , P00_ACCGRPy_VM , P00_ACCGRPy_PRS or P01_ACCGRPy_WRA , P01_ACCGRPy_WRB , P01_ACCGRPy_RDA , P01_ACCGRPy_RDB , P01_ACCGRPy_VM , P01_ACCGRPy_PRS or P02_ACCGRPy_WRA , P02_ACCGRPy_WRB , P02_ACCGRPy_RDA , P02_ACCGRPy_RDB , P02_ACCGRPy_VM , P02_ACCGRPy_PRS or P03_ACCGRPy_WRA , P03_ACCGRPy_WRB , P03_ACCGRPy_RDA , P03_ACCGRPy_RDB , P03_ACCGRPy_VM , P03_ACCGRPy_PRS or P04_ACCGRPy_WRA , P04_ACCGRPy_WRB , P04_ACCGRPy_RDA , P04_ACCGRPy_RDB , P04_ACCGRPy_VM , P04_ACCGRPy_PRS or P10_ACCGRPy_WRA , P10_ACCGRPy_WRB , P10_ACCGRPy_RDA , P10_ACCGRPy_RDB , P10_ACCGRPy_VM , P10_ACCGRPy_PRS or P13_ACCGRPy_WRA , P13_ACCGRPy_WRB , P13_ACCGRPy_RDA , P13_ACCGRPy_RDB , P13_ACCGRPy_VM , P13_ACCGRPy_PRS or P14_ACCGRPy_WRA , P14_ACCGRPy_WRB , P14_ACCGRPy_RDA , P14_ACCGRPy_RDB , P14_ACCGRPy_VM , P14_ACCGRPy_PRS or P15_ACCGRPy_WRA , P15_ACCGRPy_WRB , P15_ACCGRPy_RDA , P15_ACCGRPy_RDB , P15_ACCGRPy_VM , P15_ACCGRPy_PRS or P16_ACCGRPy_WRA , P16_ACCGRPy_WRB , P16_ACCGRPy_RDA , P16_ACCGRPy_RDB , P16_ACCGRPy_VM , P16_ACCGRPy_PRS or P20_ACCGRPy_WRA , P20_ACCGRPy_WRB , P20_ACCGRPy_RDA , P20_ACCGRPy_RDB , P20_ACCGRPy_VM , P20_ACCGRPy_PRS or P21_ACCGRPy_WRA , P21_ACCGRPy_WRB , P21_ACCGRPy_RDA , P21_ACCGRPy_RDB , P21_ACCGRPy_VM , P21_ACCGRPy_PRS or P22_ACCGRPy_WRA , P22_ACCGRPy_WRB , P22_ACCGRPy_RDA , P22_ACCGRPy_RDB , P22_ACCGRPy_VM , P22_ACCGRPy_PRS or P23_ACCGRPy_WRA , P23_ACCGRPy_WRB , P23_ACCGRPy_RDA , P23_ACCGRPy_RDB , P23_ACCGRPy_VM , P23_ACCGRPy_PRS or P25_ACCGRPy_WRA , P25_ACCGRPy_WRB , P25_ACCGRPy_RDA , P25_ACCGRPy_RDB , P25_ACCGRPy_VM , P25_ACCGRPy_PRS or P30_ACCGRPy_WRA , P30_ACCGRPy_WRB , P30_ACCGRPy_RDA , P30_ACCGRPy_RDB , P30_ACCGRPy_VM , P30_ACCGRPy_PRS or P31_ACCGRPy_WRA , P31_ACCGRPy_WRB , P31_ACCGRPy_RDA , P31_ACCGRPy_RDB , P31_ACCGRPy_VM , P31_ACCGRPy_PRS or P32_ACCGRPy_WRA , P32_ACCGRPy_WRB , P32_ACCGRPy_RDA , P32_ACCGRPy_RDB , P32_ACCGRPy_VM , P32_ACCGRPy_PRS or P33_ACCGRPy_WRA , P33_ACCGRPy_WRB , P33_ACCGRPy_RDA , P33_ACCGRPy_RDB , P33_ACCGRPy_VM , P33_ACCGRPy_PRS or P34_ACCGRPy_WRA , P34_ACCGRPy_WRB , P34_ACCGRPy_RDA , P34_ACCGRPy_RDB , P34_ACCGRPy_VM , P34_ACCGRPy_PRS or P35_ACCGRPy_WRA , P35_ACCGRPy_WRB , P35_ACCGRPy_RDA , P35_ACCGRPy_RDB , P35_ACCGRPy_VM , P35_ACCGRPy_PRS

(table continues...)

Table 2449 (continued) Register overview - access mode glossary

Keyword	Description
	or P40_ACCGRPy_WRA , P40_ACCGRPy_WRB , P40_ACCGRPy_RDA , P40_ACCGRPy_RDB , P40_ACCGRPy_VM , P40_ACCGRPy_PRS .
PGy	Access protection using *APU-PGy* registers.
APU-PGy (y=4-7)	Protection group consisting of registers P00_ACCGRPy_WRA , P00_ACCGRPy_WRB , P00_ACCGRPy_RDA , P00_ACCGRPy_RDB , P00_ACCGRPy_VM , P00_ACCGRPy_PRS or P01_ACCGRPy_WRA , P01_ACCGRPy_WRB , P01_ACCGRPy_RDA , P01_ACCGRPy_RDB , P01_ACCGRPy_VM , P01_ACCGRPy_PRS or P02_ACCGRPy_WRA , P02_ACCGRPy_WRB , P02_ACCGRPy_RDA , P02_ACCGRPy_RDB , P02_ACCGRPy_VM , P02_ACCGRPy_PRS or P03_ACCGRPy_WRA , P03_ACCGRPy_WRB , P03_ACCGRPy_RDA , P03_ACCGRPy_RDB , P03_ACCGRPy_VM , P03_ACCGRPy_PRS or P04_ACCGRPy_WRA , P04_ACCGRPy_WRB , P04_ACCGRPy_RDA , P04_ACCGRPy_RDB , P04_ACCGRPy_VM , P04_ACCGRPy_PRS or P10_ACCGRPy_WRA , P10_ACCGRPy_WRB , P10_ACCGRPy_RDA , P10_ACCGRPy_RDB , P10_ACCGRPy_VM , P10_ACCGRPy_PRS or P13_ACCGRPy_WRA , P13_ACCGRPy_WRB , P13_ACCGRPy_RDA , P13_ACCGRPy_RDB , P13_ACCGRPy_VM , P13_ACCGRPy_PRS or P14_ACCGRPy_WRA , P14_ACCGRPy_WRB , P14_ACCGRPy_RDA , P14_ACCGRPy_RDB , P14_ACCGRPy_VM , P14_ACCGRPy_PRS or P15_ACCGRPy_WRA , P15_ACCGRPy_WRB , P15_ACCGRPy_RDA , P15_ACCGRPy_RDB , P15_ACCGRPy_VM , P15_ACCGRPy_PRS or P16_ACCGRPy_WRA , P16_ACCGRPy_WRB , P16_ACCGRPy_RDA , P16_ACCGRPy_RDB , P16_ACCGRPy_VM , P16_ACCGRPy_PRS or P20_ACCGRPy_WRA , P20_ACCGRPy_WRB , P20_ACCGRPy_RDA , P20_ACCGRPy_RDB , P20_ACCGRPy_VM , P20_ACCGRPy_PRS or P22_ACCGRPy_WRA , P22_ACCGRPy_WRB , P22_ACCGRPy_RDA , P22_ACCGRPy_RDB , P22_ACCGRPy_VM , P22_ACCGRPy_PRS or P30_ACCGRPy_WRA , P30_ACCGRPy_WRB , P30_ACCGRPy_RDA , P30_ACCGRPy_RDB , P30_ACCGRPy_VM , P30_ACCGRPy_PRS or P33_ACCGRPy_WRA , P33_ACCGRPy_WRB , P33_ACCGRPy_RDA , P33_ACCGRPy_RDB , P33_ACCGRPy_VM , P33_ACCGRPy_PRS or P40_ACCGRPy_WRA , P40_ACCGRPy_WRB , P40_ACCGRPy_RDA , P40_ACCGRPy_RDB , P40_ACCGRPy_VM , P40_ACCGRPy_PRS .
PPSz (z=0-15) EPSz (z=0-15)	Access protection using one APU from *APU-PG[y] (y=0-7)*. PROT_E protection using one of *E[y] (y=0-7)*. The protection source to be used is selected by bit-field: z=0-15: PADCFGp_ACCEN.GRP
U	No access restrictions.
BE	Always returns a Bus Error.
PROT	Access restrictions as defined in the PROT register access rules.

(table continues...)

Table 2449 (continued) Register overview - access mode glossary

Keyword	Description
nBE	Indicates that no Bus Error is generated when accessing this address range, even though it is either an access to an undefined address or the access does not follow the given rules.

47.8.4.3 Register overview - P00 (ascending offset address)

Table 2450 Register overview - P00 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
P00_ID	Module Identification Register	000_H	U	BE	Application Reset	8610
P00_RST_CTRLA	Reset control register A	004_H	SV, SE	SV, SE	Application Reset	8610
P00_RST_CTRLB	Reset control register B	008_H	SV, SE	SV, SE	Application Reset	8611
P00_RST_STAT	Reset status register	$00C_H$	SV, SE	BE	Application Reset	8612
P00_WKEN	Port 00 Wake Enable Register	010_H	U	U	Application Reset	8612
P00_WKSTS	Port 00 Wake Status Register	014_H	U	BE	Application Reset	8613
P00_WKENSTS	Port 00 Wake Enable Status Register	018_H	U	BE	Application Reset	8613
P00_WKENSTSCLR	Port 00 Wake Status Clear Register	$01C_H$	U	U	Application Reset	8614
P00_OUT	Port 00 Output Register	020_H	U	U	8614	8614
P00_IN	Port 00 Input Register	024_H	U	BE	8615	8615
P00_HWSELSTAT	Port 00 HWSEL status Register	028_H	U	BE	8616	8616
P00_PDISC	Port 00 Pin Function Decision Control Register	$02C_H$	U	SV, SE	See 8720	8720
P00_PROTSE	PROT Register Safe Endinit (SE)	030_H	U	SV, PROT	Application Reset	8618
P00_PCSRSEL	Selection Register for Module x Port 00 Control	034_H	U	SV, SE	PowerOn Reset	8619
P00_OMR	Port 00 Output Modification Register	$03C_H$	nBE	U	Application Reset	8622
P00_OMCR	Port 00 Output Modification Clear Register	040_H	U	U	Application Reset	8622
P00_OMSR	Port 00 Output Modification Set Register	044_H	U	U	Application Reset	8623

(table continues...)

Table 2450 (continued) Register overview - P00 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
P00_ACCGRPy_WRA (y=0-7)	Write access enable register A	090_H+y*20_H	U	SE, SV	Application Reset	8626
P00_ACCGRPy_WRB (y=0-7)	Write access enable register B	094_H+y*20_H	U	SE, SV	Application Reset	8627
P00_ACCGRPy_RDA (y=0-7)	Read access enable register A	098_H+y*20_H	U	SE, SV	Application Reset	8628
P00_ACCGRPy_RDB (y=0-7)	Read access enable register B	$09C_H$+y*20_H	U	SE, SV	Application Reset	8628
P00_ACCGRPy_VM (y=0-7)	VM access enable register	$0A0_H$+y*20_H	U	SE, SV	Application Reset	8629
P00_ACCGRPy_PRS (y=0-7)	PRS access enable register	$0A4_H$+y*20_H	U	SE, SV	Application Reset	8630
P00_ACCGRPy_PROTE (y=0-7)	PROT Register Endinit	$0AC_H$+y*20_H	U	SV, PROT	Application Reset	8630
P00_PADCFGp_GPIO (p=0-15)	Port 00 GPIO Control Register for pin p	300_H+p*10_H	PPSp	PPSp	8632	8632
P00_PADCFGp_DRVCFG (p=0-15)	Port 00 Pad configuration Control Register for padp	304_H+p*10_H	PPSp	EPSp, SV, PPSp	See 8749	8749
P00_PADCFGp_ACCEN (p=0-15)	Port 00 Pad Access Control Register for padp	308_H+p*10_H	U	SE, SV	Application Reset	8638
P00_PADCFGp_SAFSEC (p=0-15)	Port 00 Pad Safety and Security Control Register for padp	$30C_H$+p*10_H	PPSp	EPSp, SV, PPSp	See 8764	8764

47.8.4.4 Register overview - P01 (ascending offset address)

Table 2451 Register overview - P01 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
P01_ID	Module Identification Register	000_H	U	BE	Application Reset	8610

(table continues...)

Table 2451 (continued) Register overview - P01 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
P01_RST_CTRLA	Reset control register A	004_H	SV, SE	SV, SE	Application Reset	8610
P01_RST_CTRLB	Reset control register B	008_H	SV, SE	SV, SE	Application Reset	8611
P01_RST_STAT	Reset status register	$00C_H$	SV, SE	BE	Application Reset	8612
P01_WKEN	Port 01 Wake Enable Register	010_H	U	U	Application Reset	8612
P01_WKSTS	Port 01 Wake Status Register	014_H	U	BE	Application Reset	8613
P01_WKENSTS	Port 01 Wake Enable Status Register	018_H	U	BE	Application Reset	8613
P01_WKENSTSCLR	Port 01 Wake Status Clear Register	$01C_H$	U	U	Application Reset	8614
P01_OUT	Port 01 Output Register	020_H	U	U	8614	8614
P01_IN	Port 01 Input Register	024_H	U	BE	8615	8615
P01_HWSELSTAT	Port 01 HWSEL status Register	028_H	U	BE	8616	8616
P01_PDISC	Port 01 Pin Function Decision Control Register	$02C_H$	U	SV, SE	See 8720	8720
P01_PROTSE	PROT Register Safe Endinit (SE)	030_H	U	SV, PROT	Application Reset	8618
P01_PCSRSEL	Selection Register for Module x Port 01 Control	034_H	U	SV, SE	PowerOn Reset	8619
P01_BGTRIM	Pad Bandgap Trim Register	038_H	U	SV, SE	8620	8620
P01_OMR	Port 01 Output Modification Register	$03C_H$	nBE	U	Application Reset	8622
P01_OMCR	Port 01 Output Modification Clear Register	040_H	U	U	Application Reset	8622
P01_OMSR	Port 01 Output Modification Set Register	044_H	U	U	Application Reset	8623
P01_ACCGRPy_WRA (y=0-7)	Write access enable register A	090_H+y*20_H	U	SE, SV	Application Reset	8626
P01_ACCGRPy_WRB (y=0-7)	Write access enable register B	094_H+y*20_H	U	SE, SV	Application Reset	8627
P01_ACCGRPy_RDA (y=0-7)	Read access enable register A	098_H+y*20_H	U	SE, SV	Application Reset	8628

(table continues...)

Table 2451 (continued) Register overview - P01 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
P01_ACCGRPy_RDB (y=0-7)	Read access enable register B	$09C_H$+y* 20_H	U	SE, SV	Application Reset	8628
P01_ACCGRPy_VM (y=0-7)	VM access enable register	$0A0_H$+y* 20_H	U	SE, SV	Application Reset	8629
P01_ACCGRPy_PRS (y=0-7)	PRS access enable register	$0A4_H$+y* 20_H	U	SE, SV	Application Reset	8630
P01_ACCGRPy_PROTE (y=0-7)	PROT Register Endinit	$0AC_H$+y* 20_H	U	SV, PROT	Application Reset	8630
P01_PADCFGp_GPIO (p=0-15)	Port 01 GPIO Control Register for pin p	300_H+p* 10_H	PPSp	PPSp	8632	8632
P01_PADCFGp_DRVCFG (p=0-15)	Port 01 Pad configuration Control Register for padp	304_H+p* 10_H	PPSp	EPSp, SV, PPSp	See 8749	8749
P01_PADCFGp_ACCEN (p=0-15)	Port 01 Pad Access Control Register for padp	308_H+p* 10_H	U	SE, SV	Application Reset	8638
P01_PADCFGp_SAFSEC (p=0-15)	Port 01 Pad Safety and Security Control Register for padp	$30C_H$+p* 10_H	PPSp	EPSp, SV, PPSp	See 8764	8764

47.8.4.5 Register overview - P02 (ascending offset address)

Table 2452 Register overview - P02 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
P02_ID	Module Identification Register	000_H	U	BE	Application Reset	8610
P02_RST_CTRLA	Reset control register A	004_H	SV, SE	SV, SE	Application Reset	8610
P02_RST_CTRLB	Reset control register B	008_H	SV, SE	SV, SE	Application Reset	8611
P02_RST_STAT	Reset status register	$00C_H$	SV, SE	BE	Application Reset	8612
P02_WKEN	Port 02 Wake Enable Register	010_H	U	U	Application Reset	8612
P02_WKSTS	Port 02 Wake Status Register	014_H	U	BE	Application Reset	8613

(table continues...)

Table 2452 (continued) Register overview - P02 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
P02_WKENSTS	Port 02 Wake Enable Status Register	018_H	U	BE	Application Reset	8613
P02_WKENSTSCLR	Port 02 Wake Status Clear Register	$01C_H$	U	U	Application Reset	8614
P02_OUT	Port 02 Output Register	020_H	U	U	8614	8614
P02_IN	Port 02 Input Register	024_H	U	BE	8615	8615
P02_HWSELSTAT	Port 02 HWSEL status Register	028_H	U	BE	8616	8616
P02_PDISC	Port 02 Pin Function Decision Control Register	$02C_H$	U	SV, SE	See 8720	8720
P02_PROTSE	PROT Register Safe Endinit (SE)	030_H	U	SV, PROT	Application Reset	8618
P02_PCSRSEL	Selection Register for Module x Port 02 Control	034_H	U	SV, SE	PowerOn Reset	8619
P02_OMR	Port 02 Output Modification Register	$03C_H$	nBE	U	Application Reset	8622
P02_OMCR	Port 02 Output Modification Clear Register	040_H	U	U	Application Reset	8622
P02_OMSR	Port 02 Output Modification Set Register	044_H	U	U	Application Reset	8623
P02_ACCGRPy_WRA (y=0-7)	Write access enable register A	090_H+y*20_H	U	SE, SV	Application Reset	8626
P02_ACCGRPy_WRB (y=0-7)	Write access enable register B	094_H+y*20_H	U	SE, SV	Application Reset	8627
P02_ACCGRPy_RDA (y=0-7)	Read access enable register A	098_H+y*20_H	U	SE, SV	Application Reset	8628
P02_ACCGRPy_RDB (y=0-7)	Read access enable register B	$09C_H$+y*20_H	U	SE, SV	Application Reset	8628
P02_ACCGRPy_VM (y=0-7)	VM access enable register	$0A0_H$+y*20_H	U	SE, SV	Application Reset	8629
P02_ACCGRPy_PRS (y=0-7)	PRS access enable register	$0A4_H$+y*20_H	U	SE, SV	Application Reset	8630
P02_ACCGRPy_PROTE (y=0-7)	PROT Register Endinit	$0AC_H$+y*20_H	U	SV, PROT	Application Reset	8630

(table continues...)

Table 2452 (continued) Register overview - P02 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
P02_PADCFGp_GPIO (p=0-15)	Port 02 GPIO Control Register for pin p	300_H+p* 10_H	PPSp	PPSp	8632	8632
P02_PADCFGp_DRVCFG (p=0-15)	Port 02 Pad configuration Control Register for padp	304_H+p* 10_H	PPSp	EPSp, SV, PPSp	See 8749	8749
P02_PADCFGp_ACCEN (p=0-15)	Port 02 Pad Access Control Register for padp	308_H+p* 10_H	U	SE, SV	Application Reset	8638
P02_PADCFGp_SAFSEC (p=0-15)	Port 02 Pad Safety and Security Control Register for padp	$30C_H$+p* 10_H	PPSp	EPSp, SV, PPSp	See 8764	8764

47.8.4.6 Register overview - P03 (ascending offset address)

Table 2453 Register overview - P03 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
P03_ID	Module Identification Register	000_H	U	BE	Application Reset	8610
P03_RST_CTRLA	Reset control register A	004_H	SV, SE	SV, SE	Application Reset	8610
P03_RST_CTRLB	Reset control register B	008_H	SV, SE	SV, SE	Application Reset	8611
P03_RST_STAT	Reset status register	$00C_H$	SV, SE	BE	Application Reset	8612
P03_WKEN	Port 03 Wake Enable Register	010_H	U	U	Application Reset	8612
P03_WKSTS	Port 03 Wake Status Register	014_H	U	BE	Application Reset	8613
P03_WKENSTS	Port 03 Wake Enable Status Register	018_H	U	BE	Application Reset	8613
P03_WKENSTSCLR	Port 03 Wake Status Clear Register	$01C_H$	U	U	Application Reset	8614
P03_OUT	Port 03 Output Register	020_H	U	U	8614	8614
P03_IN	Port 03 Input Register	024_H	U	BE	8615	8615
P03_HWSELSTAT	Port 03 HWSEL status Register	028_H	U	BE	8616	8616
P03_PDISC	Port 03 Pin Function Decision Control Register	$02C_H$	U	SV, SE	See 8720	8720

(table continues...)

Table 2453 (continued) Register overview - P03 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
P03_PROTSE	PROT Register Safe Endinit (SE)	030_H	U	SV, PROT	Application Reset	8618
P03_PCSRSEL	Selection Register for Module x Port 03 Control	034_H	U	SV, SE	PowerOn Reset	8619
P03_OMR	Port 03 Output Modification Register	$03C_H$	nBE	U	Application Reset	8622
P03_OMCR	Port 03 Output Modification Clear Register	040_H	U	U	Application Reset	8622
P03_OMSR	Port 03 Output Modification Set Register	044_H	U	U	Application Reset	8623
P03_ACCGRPy_WRA (y=0-7)	Write access enable register A	090_H+y* 20_H	U	SE, SV	Application Reset	8626
P03_ACCGRPy_WRB (y=0-7)	Write access enable register B	094_H+y* 20_H	U	SE, SV	Application Reset	8627
P03_ACCGRPy_RDA (y=0-7)	Read access enable register A	098_H+y* 20_H	U	SE, SV	Application Reset	8628
P03_ACCGRPy_RDB (y=0-7)	Read access enable register B	$09C_H$+y* 20_H	U	SE, SV	Application Reset	8628
P03_ACCGRPy_VM (y=0-7)	VM access enable register	$0A0_H$+y* 20_H	U	SE, SV	Application Reset	8629
P03_ACCGRPy_PRS (y=0-7)	PRS access enable register	$0A4_H$+y* 20_H	U	SE, SV	Application Reset	8630
P03_ACCGRPy_PROTE (y=0-7)	PROT Register Endinit	$0AC_H$+y* 20_H	U	SV, PROT	Application Reset	8630
P03_PADCFGp_GPIO (p=0-15)	Port 03 GPIO Control Register for pin p	300_H+p* 10_H	PPSp	PPSp	8632	8632
P03_PADCFGp_DRVCFG (p=0-15)	Port 03 Pad configuration Control Register for padp	304_H+p* 10_H	PPSp	EPSp, SV, PPSp	See 8749	8749
P03_PADCFGp_ACCEN (p=0-15)	Port 03 Pad Access Control Register for padp	308_H+p* 10_H	U	SE, SV	Application Reset	8638

(table continues...)

Table 2453 (continued) Register overview - P03 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
P03_PADCFGp_SAFSEC (p=0-15)	Port 03 Pad Safety and Security Control Register for padp	$30C_H$+p* 10_H	PPSp	EPSp, SV, PPSp	See 8764	8764

47.8.4.7 Register overview - P04 (ascending offset address)

Table 2454 Register overview - P04 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
P04_ID	Module Identification Register	000_H	U	BE	Application Reset	8610
P04_RST_CTRLA	Reset control register A	004_H	SV, SE	SV, SE	Application Reset	8610
P04_RST_CTRLB	Reset control register B	008_H	SV, SE	SV, SE	Application Reset	8611
P04_RST_STAT	Reset status register	$00C_H$	SV, SE	BE	Application Reset	8612
P04_WKEN	Port 04 Wake Enable Register	010_H	U	U	Application Reset	8693
P04_WKSTS	Port 04 Wake Status Register	014_H	U	BE	Application Reset	8696
P04_WKENSTS	Port 04 Wake Enable Status Register	018_H	U	BE	Application Reset	8699
P04_WKENSTSCLR	Port 04 Wake Status Clear Register	$01C_H$	U	U	Application Reset	8703
P04_OUT	Port 04 Output Register	020_H	U	U	See 8707	8707
P04_IN	Port 04 Input Register	024_H	U	BE	See 8711	8711
P04_HWSELSTAT	Port 04 HWSEL status Register	028_H	U	BE	See 8715	8715
P04_PDISC	Port 04 Pin Function Decision Control Register	$02C_H$	U	SV, SE	See 8722	8722
P04_PROTSE	PROT Register Safe Endinit (SE)	030_H	U	SV, PROT	Application Reset	8618
P04_PCSRSEL	Selection Register for Module x Port 04 Control	034_H	U	SV, SE	PowerOn Reset	8728
P04_OMR	Port 04 Output Modification Register	$03C_H$	U	U	Application Reset	8732
P04_OMCR	Port 04 Output Modification Clear Register	040_H	U	U	Application Reset	8737
P04_OMSR	Port 04 Output Modification Set Register	044_H	U	U	Application Reset	8741

(table continues...)

Table 2454 (continued) Register overview - P04 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
P04_ACCGRPy_WRA (y=0-7)	Write access enable register A	090_H+y*20_H	U	SE, SV	Application Reset	8626
P04_ACCGRPy_WRB (y=0-7)	Write access enable register B	094_H+y*20_H	U	SE, SV	Application Reset	8627
P04_ACCGRPy_RDA (y=0-7)	Read access enable register A	098_H+y*20_H	U	SE, SV	Application Reset	8628
P04_ACCGRPy_RDB (y=0-7)	Read access enable register B	$09C_H$+y*20_H	U	SE, SV	Application Reset	8628
P04_ACCGRPy_VM (y=0-7)	VM access enable register	$0A0_H$+y*20_H	U	SE, SV	Application Reset	8629
P04_ACCGRPy_PRS (y=0-7)	PRS access enable register	$0A4_H$+y*20_H	U	SE, SV	Application Reset	8630
P04_ACCGRPy_PROTE (y=0-7)	PROT Register Endinit	$0AC_H$+y*20_H	U	SV, PROT	Application Reset	8630
P04_PADCFGp_GPIO (p=0-9)	Port 04 GPIO Control Register for pin p	300_H+p*10_H	PPSp	PPSp	8632	8632
P04_PADCFGp_DRVCFG (p=0-9)	Port 04 Pad configuration Control Register for padp	304_H+p*10_H	PPSp	EPSp, SV, PPSp	See 8749	8749
P04_PADCFGp_ACCEN (p=0-9)	Port 04 Pad Access Control Register for padp	308_H+p*10_H	U	SE, SV	Application Reset	8638
P04_PADCFGp_SAFSEC (p=0-9)	Port 04 Pad Safety and Security Control Register for padp	$30C_H$+p*10_H	PPSp	EPSp, SV, PPSp	See 8764	8764

47.8.4.8 Register overview - P10 (ascending offset address)

Table 2455 Register overview - P10 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
P10_ID	Module Identification Register	000_H	U	BE	Application Reset	8610

(table continues...)

Table 2455 (continued) Register overview - P10 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
P10_RST_CTRLA	Reset control register A	004_H	SV, SE	SV, SE	Application Reset	8610
P10_RST_CTRLB	Reset control register B	008_H	SV, SE	SV, SE	Application Reset	8611
P10_RST_STAT	Reset status register	$00C_H$	SV, SE	BE	Application Reset	8612
P10_WKEN	Port 10 Wake Enable Register	010_H	U	U	Application Reset	8612
P10_WKSTS	Port 10 Wake Status Register	014_H	U	BE	Application Reset	8613
P10_WKENSTS	Port 10 Wake Enable Status Register	018_H	U	BE	Application Reset	8613
P10_WKENSTSCLR	Port 10 Wake Status Clear Register	$01C_H$	U	U	Application Reset	8614
P10_OUT	Port 10 Output Register	020_H	U	U	8614	8614
P10_IN	Port 10 Input Register	024_H	U	BE	8615	8615
P10_HWSELSTAT	Port 10 HWSEL status Register	028_H	U	BE	8616	8616
P10_PDISC	Port 10 Pin Function Decision Control Register	$02C_H$	U	SV, SE	See 8720	8720
P10_PROTSE	PROT Register Safe Endinit (SE)	030_H	U	SV, PROT	Application Reset	8618
P10_PCSRSEL	Selection Register for Module x Port 10 Control	034_H	U	SV, SE	PowerOn Reset	8619
P10_OMR	Port 10 Output Modification Register	$03C_H$	nBE	U	Application Reset	8622
P10_OMCR	Port 10 Output Modification Clear Register	040_H	U	U	Application Reset	8622
P10_OMSR	Port 10 Output Modification Set Register	044_H	U	U	Application Reset	8623
P10_ACCGRPy_WRA (y=0-7)	Write access enable register A	090_H+y*20_H	U	SE, SV	Application Reset	8626
P10_ACCGRPy_WRB (y=0-7)	Write access enable register B	094_H+y*20_H	U	SE, SV	Application Reset	8627
P10_ACCGRPy_RDA (y=0-7)	Read access enable register A	098_H+y*20_H	U	SE, SV	Application Reset	8628

(table continues...)

Table 2455 (continued) Register overview - P10 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
P10_ACCGRPy_RDB (y=0-7)	Read access enable register B	$09C_H$+y*20_H	U	SE, SV	Application Reset	8628
P10_ACCGRPy_VM (y=0-7)	VM access enable register	$0A0_H$+y*20_H	U	SE, SV	Application Reset	8629
P10_ACCGRPy_PRS (y=0-7)	PRS access enable register	$0A4_H$+y*20_H	U	SE, SV	Application Reset	8630
P10_ACCGRPy_PROTE (y=0-7)	PROT Register Endinit	$0AC_H$+y*20_H	U	SV, PROT	Application Reset	8630
P10_PADCFGp_GPIO (p=0-15)	Port 10 GPIO Control Register for pin p	300_H+p*10_H	PPSp	PPSp	8632	8632
P10_PADCFGp_DRVCFG (p=0-15)	Port 10 Pad configuration Control Register for padp	304_H+p*10_H	PPSp	EPSp, SV, PPSp	See 8749	8749
P10_PADCFGp_ACCEN (p=0-15)	Port 10 Pad Access Control Register for padp	308_H+p*10_H	U	SE, SV	Application Reset	8638
P10_PADCFGp_SAFSEC (p=0-15)	Port 10 Pad Safety and Security Control Register for padp	$30C_H$+p*10_H	PPSp	EPSp, SV, PPSp	See 8764	8764

47.8.4.9 Register overview - P13 (ascending offset address)

Table 2456 Register overview - P13 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
P13_ID	Module Identification Register	000_H	U	BE	Application Reset	8610
P13_RST_CTRLA	Reset control register A	004_H	SV, SE	SV, SE	Application Reset	8610
P13_RST_CTRLB	Reset control register B	008_H	SV, SE	SV, SE	Application Reset	8611
P13_RST_STAT	Reset status register	$00C_H$	SV, SE	BE	Application Reset	8612
P13_WKEN	Port 13 Wake Enable Register	010_H	U	U	Application Reset	8612
P13_WKSTS	Port 13 Wake Status Register	014_H	U	BE	Application Reset	8613

(table continues...)

Table 2456 (continued) Register overview - P13 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
P13_WKENSTS	Port 13 Wake Enable Status Register	018_H	U	BE	Application Reset	8613
P13_WKENSTSCLR	Port 13 Wake Status Clear Register	$01C_H$	U	U	Application Reset	8614
P13_OUT	Port 13 Output Register	020_H	U	U	8614	8614
P13_IN	Port 13 Input Register	024_H	U	BE	8615	8615
P13_HWSELSTAT	Port 13 HWSEL status Register	028_H	U	BE	8616	8616
P13_PDISC	Port 13 Pin Function Decision Control Register	$02C_H$	U	SV, SE	See 8720	8720
P13_PROTSE	PROT Register Safe Endinit (SE)	030_H	U	SV, PROT	Application Reset	8618
P13_PCSRSEL	Selection Register for Module x Port 13 Control	034_H	U	SV, SE	PowerOn Reset	8619
P13_BGTRIM	Pad Bandgap Trim Register	038_H	U	SV, SE	8620	8620
P13_OMR	Port 13 Output Modification Register	$03C_H$	nBE	U	Application Reset	8622
P13_OMCR	Port 13 Output Modification Clear Register	040_H	U	U	Application Reset	8622
P13_OMSR	Port 13 Output Modification Set Register	044_H	U	U	Application Reset	8623
P13_LPSRx (x=0-3)	Port 13 LVDS Pad Status Register x	048_H+x*8	U	BE	Application Reset	8745
P13_LPCRx (x=0-3)	Port 13 LVDS Pad Control Register x	$04C_H$+x*8	U	SE, SV	See 8745	8745
P13_ACCGRPy_WRA (y=0-7)	Write access enable register A	090_H+y*20_H	U	SE, SV	Application Reset	8626
P13_ACCGRPy_WRB (y=0-7)	Write access enable register B	094_H+y*20_H	U	SE, SV	Application Reset	8627
P13_ACCGRPy_RDA (y=0-7)	Read access enable register A	098_H+y*20_H	U	SE, SV	Application Reset	8628
P13_ACCGRPy_RDB (y=0-7)	Read access enable register B	$09C_H$+y*20_H	U	SE, SV	Application Reset	8628
P13_ACCGRPy_VM (y=0-7)	VM access enable register	$0A0_H$+y*20_H	U	SE, SV	Application Reset	8629

(table continues...)

Table 2456 (continued) Register overview - P13 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
P13_ACCGRPy_PRS (y=0-7)	PRS access enable register	$0A4_H$+y*20_H	U	SE, SV	Application Reset	8630
P13_ACCGRPy_PROTE (y=0-7)	PROT Register Endinit	$0AC_H$+y*20_H	U	SV, PROT	Application Reset	8630
P13_PADCFGp_GPIO (p=0-15)	Port 13 GPIO Control Register for pin p	300_H+p*10_H	PPSp	PPSp	8632	8632
P13_PADCFGp_DRVCFG (p=0-15)	Port 13 Pad configuration Control Register for padp	304_H+p*10_H	PPSp	EPSp, SV, PPSp	See 8749	8749
P13_PADCFGp_ACCEN (p=0-15)	Port 13 Pad Access Control Register for padp	308_H+p*10_H	U	SE, SV	Application Reset	8638
P13_PADCFGp_SAFSEC (p=0-15)	Port 13 Pad Safety and Security Control Register for padp	$30C_H$+p*10_H	PPSp	EPSp, SV, PPSp	See 8764	8764

47.8.4.10 Register overview - P14 (ascending offset address)

Table 2457 Register overview - P14 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
P14_ID	Module Identification Register	000_H	U	BE	Application Reset	8610
P14_RST_CTRLA	Reset control register A	004_H	SV, SE	SV, SE	Application Reset	8610
P14_RST_CTRLB	Reset control register B	008_H	SV, SE	SV, SE	Application Reset	8611
P14_RST_STAT	Reset status register	$00C_H$	SV, SE	BE	Application Reset	8612
P14_WKEN	Port 14 Wake Enable Register	010_H	U	U	Application Reset	8612
P14_WKSTS	Port 14 Wake Status Register	014_H	U	BE	Application Reset	8613
P14_WKENSTS	Port 14 Wake Enable Status Register	018_H	U	BE	Application Reset	8613
P14_WKENSTSCLR	Port 14 Wake Status Clear Register	$01C_H$	U	U	Application Reset	8614
P14_OUT	Port 14 Output Register	020_H	U	U	8614	8614

(table continues...)

Table 2457 (continued) Register overview - P14 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
P14_IN	Port 14 Input Register	024_H	U	BE	8615	8615
P14_HWSELSTAT	Port 14 HWSEL status Register	028_H	U	BE	8616	8616
P14_PDISC	Port 14 Pin Function Decision Control Register	$02C_H$	U	SV, SE	See 8720	8720
P14_PROTSE	PROT Register Safe Endinit (SE)	030_H	U	SV, PROT	Application Reset	8618
P14_PCSRSEL	Selection Register for Module x Port 14 Control	034_H	U	SV, SE	PowerOn Reset	8619
P14_BGTRIM	Pad Bandgap Trim Register	038_H	U	SV, SE	8620	8620
P14_OMR	Port 14 Output Modification Register	$03C_H$	nBE	U	Application Reset	8622
P14_OMCR	Port 14 Output Modification Clear Register	040_H	U	U	Application Reset	8622
P14_OMSR	Port 14 Output Modification Set Register	044_H	U	U	Application Reset	8623
P14_LPSRx (x=5)	Port 14 LVDS Pad Status Register x	048_H+x*8	U	BE	Application Reset	8624
P14_LPCRx (x=5)	Port 14 LVDS Pad Control Register x	$04C_H$+x*8	U	SE, SV	See 8747	8747
P14_ACCGRPy_WRA (y=0-7)	Write access enable register A	090_H+y*20_H	U	SE, SV	Application Reset	8626
P14_ACCGRPy_WRB (y=0-7)	Write access enable register B	094_H+y*20_H	U	SE, SV	Application Reset	8627
P14_ACCGRPy_RDA (y=0-7)	Read access enable register A	098_H+y*20_H	U	SE, SV	Application Reset	8628
P14_ACCGRPy_RDB (y=0-7)	Read access enable register B	$09C_H$+y*20_H	U	SE, SV	Application Reset	8628
P14_ACCGRPy_VM (y=0-7)	VM access enable register	$0A0_H$+y*20_H	U	SE, SV	Application Reset	8629
P14_ACCGRPy_PRS (y=0-7)	PRS access enable register	$0A4_H$+y*20_H	U	SE, SV	Application Reset	8630
P14_ACCGRPy_PROTE (y=0-7)	PROT Register Endinit	$0AC_H$+y*20_H	U	SV, PROT	Application Reset	8630

(table continues...)

Table 2457 (continued) Register overview - P14 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
P14_PADCFGp_GPIO (p=0-15)	Port 14 GPIO Control Register for pin p	300_H+p* 10_H	PPSp	PPSp	8632	8632
P14_PADCFGp_DRVCFG (p=0-15)	Port 14 Pad configuration Control Register for padp	304_H+p* 10_H	PPSp	EPSp, SV, PPSp	See 8749	8749
P14_PADCFGp_ACCEN (p=0-15)	Port 14 Pad Access Control Register for padp	308_H+p* 10_H	U	SE, SV	Application Reset	8638
P14_PADCFGp_SAFSEC (p=0-15)	Port 14 Pad Safety and Security Control Register for padp	$30C_H$+p* 10_H	PPSp	EPSp, SV, PPSp	See 8764	8764

47.8.4.11 Register overview - P15 (ascending offset address)

Table 2458 Register overview - P15 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
P15_ID	Module Identification Register	000_H	U	BE	Application Reset	8610
P15_RST_CTRLA	Reset control register A	004_H	SV, SE	SV, SE	Application Reset	8610
P15_RST_CTRLB	Reset control register B	008_H	SV, SE	SV, SE	Application Reset	8611
P15_RST_STAT	Reset status register	$00C_H$	SV, SE	BE	Application Reset	8612
P15_WKEN	Port 15 Wake Enable Register	010_H	U	U	Application Reset	8612
P15_WKSTS	Port 15 Wake Status Register	014_H	U	BE	Application Reset	8613
P15_WKENSTS	Port 15 Wake Enable Status Register	018_H	U	BE	Application Reset	8613
P15_WKENSTSCLR	Port 15 Wake Status Clear Register	$01C_H$	U	U	Application Reset	8614
P15_OUT	Port 15 Output Register	020_H	U	U	8614	8614
P15_IN	Port 15 Input Register	024_H	U	BE	8615	8615
P15_HWSELSTAT	Port 15 HWSEL status Register	028_H	U	BE	8616	8616
P15_PDISC	Port 15 Pin Function Decision Control Register	$02C_H$	U	SV, SE	See 8720	8720

(table continues...)

Table 2458 (continued) Register overview - P15 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
P15_PROTSE	PROT Register Safe Endinit (SE)	030_H	U	SV, PROT	Application Reset	8618
P15_PCSRSEL	Selection Register for Module x Port 15 Control	034_H	U	SV, SE	PowerOn Reset	8619
P15_OMR	Port 15 Output Modification Register	$03C_H$	nBE	U	Application Reset	8622
P15_OMCR	Port 15 Output Modification Clear Register	040_H	U	U	Application Reset	8622
P15_OMSR	Port 15 Output Modification Set Register	044_H	U	U	Application Reset	8623
P15_LPSRx (x=5-6)	Port 15 LVDS Pad Status Register x	048_H+x* 8	U	BE	Application Reset	8745
P15_LPCRx (x=5-6)	Port 15 LVDS Pad Control Register x	$04C_H$+x* 8	U	SE, SV	See 8745	8745
P15_ACCGRPy_WRA (y=0-7)	Write access enable register A	090_H+y* 20_H	U	SE, SV	Application Reset	8626
P15_ACCGRPy_WRB (y=0-7)	Write access enable register B	094_H+y* 20_H	U	SE, SV	Application Reset	8627
P15_ACCGRPy_RDA (y=0-7)	Read access enable register A	098_H+y* 20_H	U	SE, SV	Application Reset	8628
P15_ACCGRPy_RDB (y=0-7)	Read access enable register B	$09C_H$+y* 20_H	U	SE, SV	Application Reset	8628
P15_ACCGRPy_VM (y=0-7)	VM access enable register	$0A0_H$+y* 20_H	U	SE, SV	Application Reset	8629
P15_ACCGRPy_PRS (y=0-7)	PRS access enable register	$0A4_H$+y* 20_H	U	SE, SV	Application Reset	8630
P15_ACCGRPy_PROTE (y=0-7)	PROT Register Endinit	$0AC_H$+y* 20_H	U	SV, PROT	Application Reset	8630
P15_PADCFGp_GPIO (p=0-15)	Port 15 GPIO Control Register for pin p	300_H+p* 10_H	PPSp	PPSp	8632	8632
P15_PADCFGp_DRVCFG (p=0-15)	Port 15 Pad configuration Control Register for padp	304_H+p* 10_H	PPSp	EPSp, SV, PPSp	See 8749	8749

(table continues...)

Table 2458 (continued) Register overview - P15 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
P15_PADCFGp_ACCEN (p=0-15)	Port 15 Pad Access Control Register for padp	308_H+p* 10_H	U	SE, SV	Application Reset	8638
P15_PADCFGp_SAFSEC (p=0-15)	Port 15 Pad Safety and Security Control Register for padp	$30C_H$+p* 10_H	PPSp	EPSp, SV, PPSp	See 8764	8764

47.8.4.12 Register overview - P16 (ascending offset address)

Table 2459 Register overview - P16 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
P16_ID	Module Identification Register	000_H	U	BE	Application Reset	8610
P16_RST_CTRLA	Reset control register A	004_H	SV, SE	SV, SE	Application Reset	8610
P16_RST_CTRLB	Reset control register B	008_H	SV, SE	SV, SE	Application Reset	8611
P16_RST_STAT	Reset status register	$00C_H$	SV, SE	BE	Application Reset	8612
P16_OUT	Port 16 Output Register	020_H	U	U	See 8707	8707
P16_IN	Port 16 Input Register	024_H	U	BE	See 8712	8712
P16_HWSELSTAT	Port 16 HWSEL status Register	028_H	U	BE	See 8716	8716
P16_PDISC	Port 16 Pin Function Decision Control Register	$02C_H$	U	SV, SE	See 8723	8723
P16_PROTSE	PROT Register Safe Endinit (SE)	030_H	U	SV, PROT	Application Reset	8618
P16_PCSRSEL	Selection Register for Module x Port 16 Control	034_H	U	SV, SE	PowerOn Reset	8729
P16_OMR	Port 16 Output Modification Register	$03C_H$	U	U	Application Reset	8733
P16_OMCR	Port 16 Output Modification Clear Register	040_H	U	U	Application Reset	8738
P16_OMSR	Port 16 Output Modification Set Register	044_H	U	U	Application Reset	8741
P16_ACCGRPy_WRA (y=0-7)	Write access enable register A	090_H+y* 20_H	U	SE, SV	Application Reset	8626

(table continues...)

Table 2459 (continued) Register overview - P16 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
P16_ACCGRPy_WRB (y=0-7)	Write access enable register B	094_H+y* 20_H	U	SE, SV	Application Reset	8627
P16_ACCGRPy_RDA (y=0-7)	Read access enable register A	098_H+y* 20_H	U	SE, SV	Application Reset	8628
P16_ACCGRPy_RDB (y=0-7)	Read access enable register B	$09C_H$+y* 20_H	U	SE, SV	Application Reset	8628
P16_ACCGRPy_VM (y=0-7)	VM access enable register	$0A0_H$+y* 20_H	U	SE, SV	Application Reset	8629
P16_ACCGRPy_PRS (y=0-7)	PRS access enable register	$0A4_H$+y* 20_H	U	SE, SV	Application Reset	8630
P16_ACCGRPy_PROTE (y=0-7)	PROT Register Endinit	$0AC_H$+y* 20_H	U	SV, PROT	Application Reset	8630
P16_PADCFGp_GPIO (p=0-14)	Port 16 GPIO Control Register for pin p	300_H+p* 10_H	PPSp	PPSp	8632	8632
P16_PADCFGp_DRVCFG (p=0-14)	Port 16 Pad configuration Control Register for padp	304_H+p* 10_H	PPSp	EPSp, SV, PPSp	8633	8633
P16_PADCFGp_ACCEN (p=0-14)	Port 16 Pad Access Control Register for padp	308_H+p* 10_H	U	SE, SV	Application Reset	8638
P16_PADCFGp_SAFSEC (p=0-14)	Port 16 Pad Safety and Security Control Register for padp	$30C_H$+p* 10_H	PPSp	EPSp, SV, PPSp	See 8764	8764

47.8.4.13 Register overview - P20 (ascending offset address)

Table 2460 Register overview - P20 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
P20_ID	Module Identification Register	000_H	U	BE	Application Reset	8610
P20_RST_CTRLA	Reset control register A	004_H	SV, SE	SV, SE	Application Reset	8610
P20_RST_CTRLB	Reset control register B	008_H	SV, SE	SV, SE	Application Reset	8611

(table continues...)

Table 2460 (continued) Register overview - P20 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
P20_RST_STAT	Reset status register	$00C_H$	SV, SE	BE	Application Reset	8612
P20_WKEN	Port 20 Wake Enable Register	010_H	U	U	Application Reset	8693
P20_WKSTS	Port 20 Wake Status Register	014_H	U	BE	Application Reset	8697
P20_WKENSTS	Port 20 Wake Enable Status Register	018_H	U	BE	Application Reset	8700
P20_WKENSTSCLR	Port 20 Wake Status Clear Register	$01C_H$	U	U	Application Reset	8704
P20_OUT	Port 20 Output Register	020_H	U	U	See 8707	8707
P20_IN	Port 20 Input Register	024_H	U	BE	See 8712	8712
P20_HWSELSTAT	Port 20 HWSEL status Register	028_H	U	BE	See 8716	8716
P20_PDISC	Port 20 Pin Function Decision Control Register	$02C_H$	U	SV, SE	See 8723	8723
P20_PROTSE	PROT Register Safe Endinit (SE)	030_H	U	SV, PROT	Application Reset	8618
P20_PCSRSEL	Selection Register for Module x Port 20 Control	034_H	U	SV, SE	PowerOn Reset	8729
P20_OMR	Port 20 Output Modification Register	$03C_H$	U	U	Application Reset	8733
P20_OMCR	Port 20 Output Modification Clear Register	040_H	U	U	Application Reset	8738
P20_OMSR	Port 20 Output Modification Set Register	044_H	U	U	Application Reset	8741
P20_ACCGRPy_WRA (y=0-7)	Write access enable register A	090_H+y*20_H	U	SE, SV	Application Reset	8626
P20_ACCGRPy_WRB (y=0-7)	Write access enable register B	094_H+y*20_H	U	SE, SV	Application Reset	8627
P20_ACCGRPy_RDA (y=0-7)	Read access enable register A	098_H+y*20_H	U	SE, SV	Application Reset	8628
P20_ACCGRPy_RDB (y=0-7)	Read access enable register B	$09C_H$+y*20_H	U	SE, SV	Application Reset	8628
P20_ACCGRPy_VM (y=0-7)	VM access enable register	$0A0_H$+y*20_H	U	SE, SV	Application Reset	8629

(table continues...)

Table 2460 (continued) Register overview - P20 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
P20_ACCGRPy_PRS (y=0-7)	PRS access enable register	$0A4_H$+y*20_H	U	SE, SV	Application Reset	8630
P20_ACCGRPy_PROTE (y=0-7)	PROT Register Endinit	$0AC_H$+y*20_H	U	SV, PROT	Application Reset	8630
P20_PADCFGp_GPIO (p=0-14)	Port 20 GPIO Control Register for pin p	300_H+p*10_H	PPSp	PPSp	8632	8632
P20_PADCFGp_DRVCFG (p=0-14)	Port 20 Pad configuration Control Register for padp	304_H+p*10_H	PPSp	EPSp, SV, PPSp	See 8749	8749
P20_PADCFGp_ACCEN (p=0-14)	Port 20 Pad Access Control Register for padp	308_H+p*10_H	U	SE, SV	Application Reset	8638
P20_PADCFGp_SAFSEC (p=0-1,3,6-14) (p=2,4-5)	Port 20 Pad Safety and Security Control Register for padp	$30C_H$+p*10_H	PPSp	EPSp, SV, PPSp	See 8766	8766

47.8.4.14 Register overview - P21 (ascending offset address)

Table 2461 Register overview - P21 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
P21_ID	Module Identification Register	000_H	U	BE	Application Reset	8610
P21_RST_CTRLA	Reset control register A	004_H	SV, SE	SV, SE	Application Reset	8610
P21_RST_CTRLB	Reset control register B	008_H	SV, SE	SV, SE	Application Reset	8611
P21_RST_STAT	Reset status register	$00C_H$	SV, SE	BE	Application Reset	8612
P21_WKEN	Port 21 Wake Enable Register	010_H	U	U	Application Reset	8694
P21_WKSTS	Port 21 Wake Status Register	014_H	U	BE	Application Reset	8697
P21_WKENSTS	Port 21 Wake Enable Status Register	018_H	U	BE	Application Reset	8701
P21_WKENSTSCLR	Port 21 Wake Status Clear Register	$01C_H$	U	U	Application Reset	8704

(table continues...)

Table 2461 (continued) Register overview - P21 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
P21_OUT	Port 21 Output Register	020_H	U	U	See 8708	8708
P21_IN	Port 21 Input Register	024_H	U	BE	See 8713	8713
P21_HWSELSTAT	Port 21 HWSEL status Register	028_H	U	BE	See 8717	8717
P21_PDISC	Port 21 Pin Function Decision Control Register	$02C_H$	U	SV, SE	See 8724	8724
P21_PROTSE	PROT Register Safe Endinit (SE)	030_H	U	SV, PROT	Application Reset	8618
P21_PCSRSEL	Selection Register for Module x Port 21 Control	034_H	U	SV, SE	PowerOn Reset	8730
P21_BGTRIM	Pad Bandgap Trim Register	038_H	U	SV, SE	8620	8620
P21_OMR	Port 21 Output Modification Register	$03C_H$	U	U	Application Reset	8734
P21_OMCR	Port 21 Output Modification Clear Register	040_H	U	U	Application Reset	8738
P21_OMSR	Port 21 Output Modification Set Register	044_H	U	U	Application Reset	8742
P21_LPSRx (x=0-2)	Port 21 LVDS Pad Status Register x	048_H+x*8	U	BE	Application Reset	8745 and
P21_LPCRx (x=0-2)	Port 21 LVDS Pad Control Register x	$04C_H$+x*8	U	SE, SV	See 8745	8745
P21_ACCGRPy_WRA (y=0-3)	Write access enable register A	090_H+y*20_H	U	SE, SV	Application Reset	8626
P21_ACCGRPy_WRB (y=0-3)	Write access enable register B	094_H+y*20_H	U	SE, SV	Application Reset	8627
P21_ACCGRPy_RDA (y=0-3)	Read access enable register A	098_H+y*20_H	U	SE, SV	Application Reset	8628
P21_ACCGRPy_RDB (y=0-3)	Read access enable register B	$09C_H$+y*20_H	U	SE, SV	Application Reset	8628
P21_ACCGRPy_VM (y=0-3)	VM access enable register	$0A0_H$+y*20_H	U	SE, SV	Application Reset	8629
P21_ACCGRPy_PRS (y=0-3)	PRS access enable register	$0A4_H$+y*20_H	U	SE, SV	Application Reset	8630

(table continues...)

Table 2461 (continued) Register overview - P21 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
P21_ACCGRPy_PROTE (y=0-3)	PROT Register Endinit	$0AC_H$+y*20_H	U	SV, PROT	Application Reset	8630
P21_PADCFGp_GPIO (p=0-7)	Port 21 GPIO Control Register for pin p	300_H+p*10_H	PPSp	PPSp	8632	8632
P21_PADCFGp_DRVCFG (p=0-7)	Port 21 Pad configuration Control Register for padp	304_H+p*10_H	PPSp	EPSp, SV, PPSp	See 8753	8753
P21_PADCFGp_ACCEN (p=0-7)	Port 21 Pad Access Control Register for padp	308_H+p*10_H	U	SE, SV	Application Reset	8763
P21_PADCFGp_SAFSEC (p=0-1,3-7) (p=2)	Port 21 Pad Safety and Security Control Register for padp	$30C_H$+p*10_H	PPSp	EPSp, SV, PPSp	See 8766	8766

47.8.4.15 Register overview - P22 (ascending offset address)

Table 2462 Register overview - P22 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
P22_ID	Module Identification Register	000_H	U	BE	Application Reset	8610
P22_RST_CTRLA	Reset control register A	004_H	SV, SE	SV, SE	Application Reset	8610
P22_RST_CTRLB	Reset control register B	008_H	SV, SE	SV, SE	Application Reset	8611
P22_RST_STAT	Reset status register	$00C_H$	SV, SE	BE	Application Reset	8612
P22_WKEN	Port 22 Wake Enable Register	010_H	U	U	Application Reset	8694
P22_WKSTS	Port 22 Wake Status Register	014_H	U	BE	Application Reset	8698
P22_WKENSTS	Port 22 Wake Enable Status Register	018_H	U	BE	Application Reset	8701
P22_WKENSTSCLR	Port 22 Wake Status Clear Register	$01C_H$	U	U	Application Reset	8705
P22_OUT	Port 22 Output Register	020_H	U	U	See 8709	8709
P22_IN	Port 22 Input Register	024_H	U	BE	See 8713	8713

(table continues...)

Table 2462 (continued) Register overview - P22 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
P22_HWSELSTAT	Port 22 HWSEL status Register	028_H	U	BE	See 8718	8718
P22_PDISC	Port 22 Pin Function Decision Control Register	$02C_H$	U	SV, SE	See 8725	8725
P22_PROTSE	PROT Register Safe Endinit (SE)	030_H	U	SV, PROT	Application Reset	8618
P22_PCSRSEL	Selection Register for Module x Port 22 Control	034_H	U	SV, SE	PowerOn Reset	8730
P22_BGTRIM	Pad Bandgap Trim Register	038_H	U	SV, SE	8620	8620
P22_OMR	Port 22 Output Modification Register	$03C_H$	U	U	Application Reset	8735
P22_OMCR	Port 22 Output Modification Clear Register	040_H	U	U	Application Reset	8739
P22_OMSR	Port 22 Output Modification Set Register	044_H	U	U	Application Reset	8743
P22_LPSRx (x=0-1)	Port 22 LVDS Pad Status Register x	048_H+x*8	U	BE	Application Reset	8745
P22_LPCRx (x=0-1)	Port 22 LVDS Pad Control Register x	$04C_H$+x*8	U	SE, SV	See 8745	8745
P22_ACCGRPy_WRA (y=0-7)	Write access enable register A	090_H+y*20_H	U	SE, SV	Application Reset	8626
P22_ACCGRPy_WRB (y=0-7)	Write access enable register B	094_H+y*20_H	U	SE, SV	Application Reset	8627
P22_ACCGRPy_RDA (y=0-7)	Read access enable register A	098_H+y*20_H	U	SE, SV	Application Reset	8628
P22_ACCGRPy_RDB (y=0-7)	Read access enable register B	$09C_H$+y*20_H	U	SE, SV	Application Reset	8628
P22_ACCGRPy_VM (y=0-7)	VM access enable register	$0A0_H$+y*20_H	U	SE, SV	Application Reset	8629
P22_ACCGRPy_PRS (y=0-7)	PRS access enable register	$0A4_H$+y*20_H	U	SE, SV	Application Reset	8630
P22_ACCGRPy_PROTE (y=0-7)	PROT Register Endinit	$0AC_H$+y*20_H	U	SV, PROT	Application Reset	8630

(table continues...)

Table 2462 (continued) Register overview - P22 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
P22_PADCFGp_GPIO (p=0-11)	Port 22 GPIO Control Register for pin p	300_H+p* 10_H	PPSp	PPSp	8632	8632
P22_PADCFGp_DRVCFG (p=0-11)	Port 22 Pad configuration Control Register for padp	304_H+p* 10_H	PPSp	EPSp, SV, PPSp	See 8753	8753
P22_PADCFGp_ACCEN (p=0-11)	Port 22 Pad Access Control Register for padp	308_H+p* 10_H	U	SE, SV	Application Reset	8638
P22_PADCFGp_SAFSEC (p=0-11)	Port 22 Pad Safety and Security Control Register for padp	$30C_H$+p* 10_H	PPSp	EPSp, SV, PPSp	See 8766	8766

47.8.4.16 Register overview - P23 (ascending offset address)

Table 2463 Register overview - P23 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
P23_ID	Module Identification Register	000_H	U	BE	Application Reset	8610
P23_RST_CTRLA	Reset control register A	004_H	SV, SE	SV, SE	Application Reset	8610
P23_RST_CTRLB	Reset control register B	008_H	SV, SE	SV, SE	Application Reset	8611
P23_RST_STAT	Reset status register	$00C_H$	SV, SE	BE	Application Reset	8612
P23_WKEN	Port 23 Wake Enable Register	010_H	U	U	Application Reset	8694
P23_WKSTS	Port 23 Wake Status Register	014_H	U	BE	Application Reset	8697
P23_WKENSTS	Port 23 Wake Enable Status Register	018_H	U	BE	Application Reset	8701
P23_WKENSTSCLR	Port 23 Wake Status Clear Register	$01C_H$	U	U	Application Reset	8704
P23_OUT	Port 23 Output Register	020_H	U	U	See 8708	8708
P23_IN	Port 23 Input Register	024_H	U	BE	See 8713	8713
P23_HWSELSTAT	Port 23 HWSEL status Register	028_H	U	BE	See 8717	8717
P23_PDISC	Port 23 Pin Function Decision Control Register	$02C_H$	U	SV, SE	See 8724	8724

(table continues...)

Table 2463 (continued) Register overview - P23 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
P23_PROTSE	PROT Register Safe Endinit (SE)	030_H	U	SV, PROT	Application Reset	8618
P23_PCSRSEL	Selection Register for Module x Port 23 Control	034_H	U	SV, SE	PowerOn Reset	8730
P23_OMR	Port 23 Output Modification Register	$03C_H$	U	U	Application Reset	8734
P23_OMCR	Port 23 Output Modification Clear Register	040_H	U	U	Application Reset	8738
P23_OMSR	Port 23 Output Modification Set Register	044_H	U	U	Application Reset	8742
P23_ACCGRPy_WRA (y=0-3)	Write access enable register A	090_H+y* 20_H	U	SE, SV	Application Reset	8626
P23_ACCGRPy_WRB (y=0-3)	Write access enable register B	094_H+y* 20_H	U	SE, SV	Application Reset	8627
P23_ACCGRPy_RDA (y=0-3)	Read access enable register A	098_H+y* 20_H	U	SE, SV	Application Reset	8628
P23_ACCGRPy_RDB (y=0-3)	Read access enable register B	$09C_H$+y* 20_H	U	SE, SV	Application Reset	8628
P23_ACCGRPy_VM (y=0-3)	VM access enable register	$0A0_H$+y* 20_H	U	SE, SV	Application Reset	8629
P23_ACCGRPy_PRS (y=0-3)	PRS access enable register	$0A4_H$+y* 20_H	U	SE, SV	Application Reset	8630
P23_ACCGRPy_PROTE (y=0-3)	PROT Register Endinit	$0AC_H$+y* 20_H	U	SV, PROT	Application Reset	8630
P23_PADCFGp_GPIO (p=0-7)	Port 23 GPIO Control Register for pin p	300_H+p* 10_H	PPSp	PPSp	8632	8632
P23_PADCFGp_DRVCFG (p=0-7)	Port 23 Pad configuration Control Register for padp	304_H+p* 10_H	PPSp	EPSp, SV, PPSp	See 8753	8753
P23_PADCFGp_ACCEN (p=0-7)	Port 23 Pad Access Control Register for padp	308_H+p* 10_H	U	SE, SV	Application Reset	8763

(table continues...)

Table 2463 (continued) Register overview - P23 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
P23_PADCFGp_SAFSEC (p=0-7)	Port 23 Pad Safety and Security Control Register for padp	$30C_H$+p* 10_H	PPSp	EPSp, SV, PPSp	See 8766	8766

47.8.4.17 Register overview - P25 (ascending offset address)

Table 2464 Register overview - P25 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
P25_ID	Module Identification Register	000_H	U	BE	Application Reset	8610
P25_RST_CTRLA	Reset control register A	004_H	SV, SE	SV, SE	Application Reset	8610
P25_RST_CTRLB	Reset control register B	008_H	SV, SE	SV, SE	Application Reset	8611
P25_RST_STAT	Reset status register	$00C_H$	SV, SE	BE	Application Reset	8612
P25_WKEN	Port 25 Wake Enable Register	010_H	U	U	Application Reset	8695
P25_WKSTS	Port 25 Wake Status Register	014_H	U	BE	Application Reset	8698
P25_WKENSTS	Port 25 Wake Enable Status Register	018_H	U	BE	Application Reset	8702
P25_WKENSTSCLR	Port 25 Wake Status Clear Register	$01C_H$	U	U	Application Reset	8705
P25_OUT	Port 25 Output Register	020_H	U	U	See 8710	8710
P25_IN	Port 25 Input Register	024_H	U	BE	See 8714	8714
P25_HWSELSTAT	Port 25 HWSEL status Register	028_H	U	BE	See 8718	8718
P25_PDISC	Port 25 Pin Function Decision Control Register	$02C_H$	U	SV, SE	See 8726	8726
P25_PROTSE	PROT Register Safe Endinit (SE)	030_H	U	SV, PROT	Application Reset	8618
P25_PCSRSEL	Selection Register for Module x Port 25 Control	034_H	U	SV, SE	PowerOn Reset	8731
P25_OMR	Port 25 Output Modification Register	$03C_H$	U	U	Application Reset	8735
P25_OMCR	Port 25 Output Modification Clear Register	040_H	U	U	Application Reset	8739
P25_OMSR	Port 25 Output Modification Set Register	044_H	U	U	Application Reset	8743

(table continues...)

Table 2464 (continued) Register overview - P25 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
P25_ACCGRPy_WRA (y=0-3)	Write access enable register A	090_H+y*20_H	U	SE, SV	Application Reset	8626
P25_ACCGRPy_WRB (y=0-3)	Write access enable register B	094_H+y*20_H	U	SE, SV	Application Reset	8627
P25_ACCGRPy_RDA (y=0-3)	Read access enable register A	098_H+y*20_H	U	SE, SV	Application Reset	8628
P25_ACCGRPy_RDB (y=0-3)	Read access enable register B	$09C_H$+y*20_H	U	SE, SV	Application Reset	8628
P25_ACCGRPy_VM (y=0-3)	VM access enable register	$0A0_H$+y*20_H	U	SE, SV	Application Reset	8629
P25_ACCGRPy_PRS (y=0-3)	PRS access enable register	$0A4_H$+y*20_H	U	SE, SV	Application Reset	8630
P25_ACCGRPy_PROTE (y=0-3)	PROT Register Endinit	$0AC_H$+y*20_H	U	SV, PROT	Application Reset	8630
P25_PADCFGp_GPIO (p=0-1)	Port 25 GPIO Control Register for pin p	300_H+p*10_H	PPSp	PPSp	8632	8632
P25_PADCFGp_DRVCFG (p=0-1)	Port 25 Pad configuration Control Register for padp	304_H+p*10_H	PPSp	EPSp, SV, PPSp	See 8753	8753
P25_PADCFGp_ACCEN (p=0-1)	Port 25 Pad Access Control Register for padp	308_H+p*10_H	U	SE, SV	Application Reset	8763
P25_PADCFGp_SAFSEC (p=0-1)	Port 25 Pad Safety and Security Control Register for padp	$30C_H$+p*10_H	PPSp	EPSp, SV, PPSp	See 8766	8766

47.8.4.18 Register overview - P30 (ascending offset address)

Table 2465 Register overview - P30 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
P30_ID	Module Identification Register	000_H	U	BE	Application Reset	8610

(table continues...)

Table 2465 (continued) Register overview - P30 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
P30_RST_CTRLA	Reset control register A	004_H	SV, SE	SV, SE	Application Reset	8610
P30_RST_CTRLB	Reset control register B	008_H	SV, SE	SV, SE	Application Reset	8611
P30_RST_STAT	Reset status register	$00C_H$	SV, SE	BE	Application Reset	8612
P30_WKEN	Port 30 Wake Enable Register	010_H	U	U	Application Reset	8612
P30_WKSTS	Port 30 Wake Status Register	014_H	U	BE	Application Reset	8613
P30_WKENSTS	Port 30 Wake Enable Status Register	018_H	U	BE	Application Reset	8613
P30_WKENSTSCLR	Port 30 Wake Status Clear Register	$01C_H$	U	U	Application Reset	8614
P30_OUT	Port 30 Output Register	020_H	U	U	8614	8614
P30_IN	Port 30 Input Register	024_H	U	BE	8615	8615
P30_HWSELSTAT	Port 30 HWSEL status Register	028_H	U	BE	8616	8616
P30_PDISC	Port 30 Pin Function Decision Control Register	$02C_H$	U	SV, SE	See 8720	8720
P30_PROTSE	PROT Register Safe Endinit (SE)	030_H	U	SV, PROT	Application Reset	8618
P30_PCSRSEL	Selection Register for Module x Port 30 Control	034_H	U	SV, SE	PowerOn Reset	8619
P30_BGTRIM	Pad Bandgap Trim Register	038_H	U	SV, SE	8620	8620
P30_OMR	Port 30 Output Modification Register	$03C_H$	nBE	U	Application Reset	8622
P30_OMCR	Port 30 Output Modification Clear Register	040_H	U	U	Application Reset	8622
P30_OMSR	Port 30 Output Modification Set Register	044_H	U	U	Application Reset	8623
P30_ACCGRPy_WRA (y=0-7)	Write access enable register A	090_H+y* 20_H	U	SE, SV	Application Reset	8626
P30_ACCGRPy_WRB (y=0-7)	Write access enable register B	094_H+y* 20_H	U	SE, SV	Application Reset	8627
P30_ACCGRPy_RDA (y=0-7)	Read access enable register A	098_H+y* 20_H	U	SE, SV	Application Reset	8628

(table continues...)

Table 2465 (continued) Register overview - P30 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
P30_ACCGRPy_RDB (y=0-7)	Read access enable register B	$09C_H$+y*20_H	U	SE, SV	Application Reset	8628
P30_ACCGRPy_VM (y=0-7)	VM access enable register	$0A0_H$+y*20_H	U	SE, SV	Application Reset	8629
P30_ACCGRPy_PRS (y=0-7)	PRS access enable register	$0A4_H$+y*20_H	U	SE, SV	Application Reset	8630
P30_ACCGRPy_PROTE (y=0-7)	PROT Register Endinit	$0AC_H$+y*20_H	U	SV, PROT	Application Reset	8630
P30_PADCFGp_GPIO (p=0-15)	Port 30 GPIO Control Register for pin p	300_H+p*10_H	PPSp	PPSp	8632	8632
P30_PADCFGp_DRVCFG (p=0-15)	Port 30 Pad configuration Control Register for padp	304_H+p*10_H	PPSp	EPSp, SV, PPSp	See 8753	8753
P30_PADCFGp_ACCEN (p=0-15)	Port 30 Pad Access Control Register for padp	308_H+p*10_H	U	SE, SV	Application Reset	8638
P30_PADCFGp_SAFSEC (p=0-15)	Port 30 Pad Safety and Security Control Register for padp	$30C_H$+p*10_H	PPSp	EPSp, SV, PPSp	See 8766	8766

47.8.4.19 Register overview - P31 (ascending offset address)

Table 2466 Register overview - P31 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
P31_ID	Module Identification Register	000_H	U	BE	Application Reset	8610
P31_RST_CTRLA	Reset control register A	004_H	SV, SE	SV, SE	Application Reset	8610
P31_RST_CTRLB	Reset control register B	008_H	SV, SE	SV, SE	Application Reset	8611
P31_RST_STAT	Reset status register	$00C_H$	SV, SE	BE	Application Reset	8612
P31_WKEN	Port 31 Wake Enable Register	010_H	U	U	Application Reset	8694
P31_WKSTS	Port 31 Wake Status Register	014_H	U	BE	Application Reset	8697

(table continues...)

Table 2466 (continued) Register overview - P31 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
P31_WKENSTS	Port 31 Wake Enable Status Register	018_H	U	BE	Application Reset	8701
P31_WKENSTSCLR	Port 31 Wake Status Clear Register	$01C_H$	U	U	Application Reset	8704
P31_OUT	Port 31 Output Register	020_H	U	U	See 8708	8708
P31_IN	Port 31 Input Register	024_H	U	BE	See 8713	8713
P31_HWSELSTAT	Port 31 HWSEL status Register	028_H	U	BE	See 8717	8717
P31_PDISC	Port 31 Pin Function Decision Control Register	$02C_H$	U	SV, SE	See 8724	8724
P31_PROTSE	PROT Register Safe Endinit (SE)	030_H	U	SV, PROT	Application Reset	8618
P31_PCSRSEL	Selection Register for Module x Port 31 Control	034_H	U	SV, SE	PowerOn Reset	8730
P31_OMR	Port 31 Output Modification Register	$03C_H$	U	U	Application Reset	8734
P31_OMCR	Port 31 Output Modification Clear Register	040_H	U	U	Application Reset	8738
P31_OMSR	Port 31 Output Modification Set Register	044_H	U	U	Application Reset	8742
P31_ACCGRPy_WRA (y=0-3)	Write access enable register A	090_H+y*20_H	U	SE, SV	Application Reset	8626
P31_ACCGRPy_WRB (y=0-3)	Write access enable register B	094_H+y*20_H	U	SE, SV	Application Reset	8627
P31_ACCGRPy_RDA (y=0-3)	Read access enable register A	098_H+y*20_H	U	SE, SV	Application Reset	8628
P31_ACCGRPy_RDB (y=0-3)	Read access enable register B	$09C_H$+y*20_H	U	SE, SV	Application Reset	8628
P31_ACCGRPy_VM (y=0-3)	VM access enable register	$0A0_H$+y*20_H	U	SE, SV	Application Reset	8629
P31_ACCGRPy_PRS (y=0-3)	PRS access enable register	$0A4_H$+y*20_H	U	SE, SV	Application Reset	8630
P31_ACCGRPy_PROTE (y=0-3)	PROT Register Endinit	$0AC_H$+y*20_H	U	SV, PROT	Application Reset	8630

(table continues...)

Table 2466 (continued) Register overview - P31 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
P31_PADCFGp_GPIO (p=0-7)	Port 31 GPIO Control Register for pin p	300_H+p* 10_H	PPSp	PPSp	8632	8632
P31_PADCFGp_DRVCFG (p=0-7)	Port 31 Pad configuration Control Register for padp	304_H+p* 10_H	PPSp	EPSp, SV, PPSp	See 8753	8753
P31_PADCFGp_ACCEN (p=0-7)	Port 31 Pad Access Control Register for padp	308_H+p* 10_H	U	SE, SV	Application Reset	8763
P31_PADCFGp_SAFSEC (p=0-7)	Port 31 Pad Safety and Security Control Register for padp	$30C_H$+p* 10_H	PPSp	EPSp, SV, PPSp	See 8766	8766

47.8.4.20 Register overview - P32 (ascending offset address)

Table 2467 Register overview - P32 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
P32_ID	Module Identification Register	000_H	U	BE	Application Reset	8610
P32_RST_CTRLA	Reset control register A	004_H	SV, SE	SV, SE	Application Reset	8610
P32_RST_CTRLB	Reset control register B	008_H	SV, SE	SV, SE	Application Reset	8611
P32_RST_STAT	Reset status register	$00C_H$	SV, SE	BE	Application Reset	8612
P32_WKEN	Port 32 Wake Enable Register	010_H	U	U	Application Reset	8694
P32_WKSTS	Port 32 Wake Status Register	014_H	U	BE	Application Reset	8697
P32_WKENSTS	Port 32 Wake Enable Status Register	018_H	U	BE	Application Reset	8701
P32_WKENSTSCLR	Port 32 Wake Status Clear Register	$01C_H$	U	U	Application Reset	8704
P32_OUT	Port 32 Output Register	020_H	U	U	See 8708	8708
P32_IN	Port 32 Input Register	024_H	U	BE	See 8713	8713
P32_HWSELSTAT	Port 32 HWSEL status Register	028_H	U	BE	See 8717	8717
P32_PDISC	Port 32 Pin Function Decision Control Register	$02C_H$	U	SV, SE	See 8724	8724

(table continues...)

Table 2467 (continued) Register overview - P32 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
P32_PROTSE	PROT Register Safe Endinit (SE)	030_H	U	SV, PROT	Application Reset	8618
P32_PCSRSEL	Selection Register for Module x Port 32 Control	034_H	U	SV, SE	PowerOn Reset	8730
P32_OMR	Port 32 Output Modification Register	$03C_H$	U	U	Application Reset	8734
P32_OMCR	Port 32 Output Modification Clear Register	040_H	U	U	Application Reset	8738
P32_OMSR	Port 32 Output Modification Set Register	044_H	U	U	Application Reset	8742
P32_ACCGRPy_WRA (y=0-3)	Write access enable register A	090_H+y* 20_H	U	SE, SV	Application Reset	8626
P32_ACCGRPy_WRB (y=0-3)	Write access enable register B	094_H+y* 20_H	U	SE, SV	Application Reset	8627
P32_ACCGRPy_RDA (y=0-3)	Read access enable register A	098_H+y* 20_H	U	SE, SV	Application Reset	8628
P32_ACCGRPy_RDB (y=0-3)	Read access enable register B	$09C_H$+y* 20_H	U	SE, SV	Application Reset	8628
P32_ACCGRPy_VM (y=0-3)	VM access enable register	$0A0_H$+y* 20_H	U	SE, SV	Application Reset	8629
P32_ACCGRPy_PRS (y=0-3)	PRS access enable register	$0A4_H$+y* 20_H	U	SE, SV	Application Reset	8630
P32_ACCGRPy_PROTE (y=0-3)	PROT Register Endinit	$0AC_H$+y* 20_H	U	SV, PROT	Application Reset	8630
P32_PADCFGp_GPIO (p=0-7)	Port 32 GPIO Control Register for pin p	300_H+p* 10_H	PPSp	PPSp	8632	8632
P32_PADCFGp_DRVCFG (p=0-1,3-4,6-7) (p=2,5)	Port 32 Pad configuration Control Register for padp	304_H+p* 10_H	PPSp	EPSp, SV, PPSp	See 8753	8753
P32_PADCFGp_ACCEN (p=0-7)	Port 32 Pad Access Control Register for padp	308_H+p* 10_H	U	SE, SV	Application Reset	8763

(table continues...)

Table 2467 (continued) Register overview - P32 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
P32_PADCFGp_SAFSEC (p=2,4-7) (p=0-1,3)	Port 32 Pad Safety and Security Control Register for padp	$30C_H$+p* 10_H	PPSp	EPSp, SV, PPSp	See 8769	8769

47.8.4.21 Register overview - P33 (ascending offset address)

Table 2468 Register overview - P33 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
P33_ID	Module Identification Register	000_H	U	BE	Application Reset	8610
P33_RST_CTRLA	Reset control register A	004_H	SV, SE	SV, SE	Application Reset	8610
P33_RST_CTRLB	Reset control register B	008_H	SV, SE	SV, SE	Application Reset	8611
P33_RST_STAT	Reset status register	$00C_H$	SV, SE	BE	Application Reset	8612
P33_WKEN	Port 33 Wake Enable Register	010_H	U	U	Application Reset	8612
P33_WKSTS	Port 33 Wake Status Register	014_H	U	BE	Application Reset	8613
P33_WKENSTS	Port 33 Wake Enable Status Register	018_H	U	BE	Application Reset	8613
P33_WKENSTSCLR	Port 33 Wake Status Clear Register	$01C_H$	U	U	Application Reset	8614
P33_OUT	Port 33 Output Register	020_H	U	U	8614	8614
P33_IN	Port 33 Input Register	024_H	U	BE	8615	8615
P33_HWSELSTAT	Port 33 HWSEL status Register	028_H	U	BE	8616	8616
P33_PDISC	Port 33 Pin Function Decision Control Register	$02C_H$	U	SV, SE	See 8720	8720
P33_PROTSE	PROT Register Safe Endinit (SE)	030_H	U	SV, PROT	Application Reset	8618
P33_PCSRSEL	Selection Register for Module x Port 33 Control	034_H	U	SV, SE	PowerOn Reset	8728
P33_OMR	Port 33 Output Modification Register	$03C_H$	nBE	U	Application Reset	8622
P33_OMCR	Port 33 Output Modification Clear Register	040_H	U	U	Application Reset	8622

(table continues...)

Table 2468 (continued) Register overview - P33 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
P33_OMSR	Port 33 Output Modification Set Register	044_H	U	U	Application Reset	8623
P33_ACCGRPy_WRA (y=0-7)	Write access enable register A	090_H+y* 20_H	U	SE, SV	Application Reset	8626
P33_ACCGRPy_WRB (y=0-7)	Write access enable register B	094_H+y* 20_H	U	SE, SV	Application Reset	8627
P33_ACCGRPy_RDA (y=0-7)	Read access enable register A	098_H+y* 20_H	U	SE, SV	Application Reset	8628
P33_ACCGRPy_RDB (y=0-7)	Read access enable register B	$09C_H$+y* 20_H	U	SE, SV	Application Reset	8628
P33_ACCGRPy_VM (y=0-7)	VM access enable register	$0A0_H$+y* 20_H	U	SE, SV	Application Reset	8629
P33_ACCGRPy_PRS (y=0-7)	PRS access enable register	$0A4_H$+y* 20_H	U	SE, SV	Application Reset	8630
P33_ACCGRPy_PROTE (y=0-7)	PROT Register Endinit	$0AC_H$+y* 20_H	U	SV, PROT	Application Reset	8630
P33_PADCFGp_GPIO (p=0-15)	Port 33 GPIO Control Register for pin p	300_H+p* 10_H	PPSp	PPSp	8632	8632
P33_PADCFGp_DRVCFG (p=0-7,9-11,13-15) (p=8,12)	Port 33 Pad configuration Control Register for padp	304_H+p* 10_H	PPSp	EPSp, SV, PPSp	See 8753	8753
P33_PADCFGp_ACCEN (p=0-15)	Port 33 Pad Access Control Register for padp	308_H+p* 10_H	U	SE, SV	Application Reset	8638
P33_PADCFGp_SAFSEC (p=0-7,9-15) (p=8)	Port 33 Pad Safety and Security Control Register for padp	$30C_H$+p* 10_H	PPSp	EPSp, SV, PPSp	See 8766	8766

47.8.4.22 Register overview - P34 (ascending offset address)

Table 2469 Register overview - P34 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
P34_ID	Module Identification Register	000_H	U	BE	Application Reset	8610
P34_RST_CTRLA	Reset control register A	004_H	SV, SE	SV, SE	Application Reset	8610
P34_RST_CTRLB	Reset control register B	008_H	SV, SE	SV, SE	Application Reset	8611
P34_RST_STAT	Reset status register	$00C_H$	SV, SE	BE	Application Reset	8612
P34_WKEN	Port 34 Wake Enable Register	010_H	U	U	Application Reset	8695
P34_WKSTS	Port 34 Wake Status Register	014_H	U	BE	Application Reset	8699
P34_WKENSTS	Port 34 Wake Enable Status Register	018_H	U	BE	Application Reset	8702
P34_WKENSTSCLR	Port 34 Wake Status Clear Register	$01C_H$	U	U	Application Reset	8706
P34_OUT	Port 34 Output Register	020_H	U	U	See 8710	8710
P34_IN	Port 34 Input Register	024_H	U	BE	See 8715	8715
P34_HWSELSTAT	Port 34 HWSEL status Register	028_H	U	BE	See 8719	8719
P34_PDISC	Port 34 Pin Function Decision Control Register	$02C_H$	U	SV, SE	See 8727	8727
P34_PROTSE	PROT Register Safe Endinit (SE)	030_H	U	SV, PROT	Application Reset	8618
P34_PCSRSEL	Selection Register for Module x Port 34 Control	034_H	U	SV, SE	PowerOn Reset	8731
P34_OMR	Port 34 Output Modification Register	$03C_H$	U	U	Application Reset	8736
P34_OMCR	Port 34 Output Modification Clear Register	040_H	U	U	Application Reset	8740
P34_OMSR	Port 34 Output Modification Set Register	044_H	U	U	Application Reset	8744
P34_ACCGRPy_WRA (y=0-3)	Write access enable register A	090_H+y*20_H	U	SE, SV	Application Reset	8626
P34_ACCGRPy_WRB (y=0-3)	Write access enable register B	094_H+y*20_H	U	SE, SV	Application Reset	8627

(table continues...)

Table 2469 (continued) Register overview - P34 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
P34_ACCGRPy_RDA (y=0-3)	Read access enable register A	098_H+y*20_H	U	SE, SV	Application Reset	8628
P34_ACCGRPy_RDB (y=0-3)	Read access enable register B	$09C_H$+y*20_H	U	SE, SV	Application Reset	8628
P34_ACCGRPy_VM (y=0-3)	VM access enable register	$0A0_H$+y*20_H	U	SE, SV	Application Reset	8629
P34_ACCGRPy_PRS (y=0-3)	PRS access enable register	$0A4_H$+y*20_H	U	SE, SV	Application Reset	8630
P34_ACCGRPy_PROTE (y=0-3)	PROT Register Endinit	$0AC_H$+y*20_H	U	SV, PROT	Application Reset	8630
P34_PADCFGp_GPIO (p=0-5)	Port 34 GPIO Control Register for pin p	300_H+p*10_H	PPSp	PPSp	8632	8632
P34_PADCFGp_DRVCFG (p=0-5)	Port 34 Pad configuration Control Register for padp	304_H+p*10_H	PPSp	EPSp, SV, PPSp	See 8753	8753
P34_PADCFGp_ACCEN (p=0-5)	Port 34 Pad Access Control Register for padp	308_H+p*10_H	U	SE, SV	Application Reset	8763
P34_PADCFGp_SAFSEC (p=0-5)	Port 34 Pad Safety and Security Control Register for padp	$30C_H$+p*10_H	PPSp	EPSp, SV, PPSp	See 8766	8766

47.8.4.23 Register overview - P35 (ascending offset address)

Table 2470 Register overview - P35 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
P35_ID	Module Identification Register	000_H	U	BE	Application Reset	8610
P35_RST_CTRLA	Reset control register A	004_H	SV, SE	SV, SE	Application Reset	8610
P35_RST_CTRLB	Reset control register B	008_H	SV, SE	SV, SE	Application Reset	8611
P35_RST_STAT	Reset status register	$00C_H$	SV, SE	BE	Application Reset	8612

(table continues...)

Table 2470 (continued) Register overview - P35 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
P35_WKEN	Port 35 Wake Enable Register	010_H	U	U	Application Reset	8695
P35_WKSTS	Port 35 Wake Status Register	014_H	U	BE	Application Reset	8699
P35_WKENSTS	Port 35 Wake Enable Status Register	018_H	U	BE	Application Reset	8702
P35_WKENSTSCLR	Port 35 Wake Status Clear Register	$01C_H$	U	U	Application Reset	8706
P35_OUT	Port 35 Output Register	020_H	U	U	See 8710	8710
P35_IN	Port 35 Input Register	024_H	U	BE	See 8715	8715
P35_HWSELSTAT	Port 35 HWSEL status Register	028_H	U	BE	See 8719	8719
P35_PDISC	Port 35 Pin Function Decision Control Register	$02C_H$	U	SV, SE	See 8727	8727
P35_PROTSE	PROT Register Safe Endinit (SE)	030_H	U	SV, PROT	Application Reset	8618
P35_PCSRSEL	Selection Register for Module x Port 35 Control	034_H	U	SV, SE	PowerOn Reset	8731
P35_OMR	Port 35 Output Modification Register	$03C_H$	U	U	Application Reset	8736
P35_OMCR	Port 35 Output Modification Clear Register	040_H	U	U	Application Reset	8740
P35_OMSR	Port 35 Output Modification Set Register	044_H	U	U	Application Reset	8744
P35_ACCGRPy_WRA (y=0-3)	Write access enable register A	090_H+y*20_H	U	SE, SV	Application Reset	8626
P35_ACCGRPy_WRB (y=0-3)	Write access enable register B	094_H+y*20_H	U	SE, SV	Application Reset	8627
P35_ACCGRPy_RDA (y=0-3)	Read access enable register A	098_H+y*20_H	U	SE, SV	Application Reset	8628
P35_ACCGRPy_RDB (y=0-3)	Read access enable register B	$09C_H$+y*20_H	U	SE, SV	Application Reset	8628
P35_ACCGRPy_VM (y=0-3)	VM access enable register	$0A0_H$+y*20_H	U	SE, SV	Application Reset	8629

(table continues...)

Table 2470 (continued) Register overview - P35 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
P35_ACCGRPy_PRS (y=0-3)	PRS access enable register	$0A4_H$+y*20_H	U	SE, SV	Application Reset	8630
P35_ACCGRPy_PROTE (y=0-3)	PROT Register Endinit	$0AC_H$+y*20_H	U	SV, PROT	Application Reset	8630
P35_PADCFGp_GPIO (p=0-5)	Port 35 GPIO Control Register for pin p	300_H+p*10_H	PPSp	PPSp	8632	8632
P35_PADCFGp_DRVCFG (p=0-5)	Port 35 Pad configuration Control Register for padp	304_H+p*10_H	PPSp	EPSp, SV, PPSp	See 8753	8753
P35_PADCFGp_ACCEN (p=0-5)	Port 35 Pad Access Control Register for padp	308_H+p*10_H	U	SE, SV	Application Reset	8763
P35_PADCFGp_SAFSEC (p=0-5)	Port 35 Pad Safety and Security Control Register for padp	$30C_H$+p*10_H	PPSp	EPSp, SV, PPSp	See 8766	8766

47.8.4.24 Register overview - P40 (ascending offset address)

Table 2471 Register overview - P40 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
P40_ID	Module Identification Register	000_H	U	BE	Application Reset	8610
P40_RST_CTRLA	Reset control register A	004_H	SV, SE	SV, SE	Application Reset	8610
P40_RST_CTRLB	Reset control register B	008_H	SV, SE	SV, SE	Application Reset	8611
P40_RST_STAT	Reset status register	$00C_H$	SV, SE	BE	Application Reset	8612
P40_OUT	Port 40 Output Register	020_H	U	U	8614	8614
P40_IN	Port 40 Input Register	024_H	U	BE	8615	8615
P40_HWSELSTAT	Port 40 HWSEL status Register	028_H	U	BE	8616	8616
P40_PDISC	Port 40 Pin Function Decision Control Register	$02C_H$	U	SV, SE	See 8721	8721
P40_PROTSE	PROT Register Safe Endinit (SE)	030_H	U	SV, PROT	Application Reset	8618

(table continues...)

Table 2471 (continued) Register overview - P40 (ascending offset address)

Short name	Long name	Offset address	Access mode		Reset	See
			Read	Write		
P40_PCSRSEL	Selection Register for Module x Port 40 Control	034_H	U	SV, SE	PowerOn Reset	8619
P40_OMR	Port 40 Output Modification Register	$03C_H$	nBE	U	Application Reset	8622
P40_OMCR	Port 40 Output Modification Clear Register	040_H	U	U	Application Reset	8622
P40_OMSR	Port 40 Output Modification Set Register	044_H	U	U	Application Reset	8623
P40_ACCGRPy_WRA (y=0-7)	Write access enable register A	090_H+y* 20_H	U	SE, SV	Application Reset	8626
P40_ACCGRPy_WRB (y=0-7)	Write access enable register B	094_H+y* 20_H	U	SE, SV	Application Reset	8627
P40_ACCGRPy_RDA (y=0-7)	Read access enable register A	098_H+y* 20_H	U	SE, SV	Application Reset	8628
P40_ACCGRPy_RDB (y=0-7)	Read access enable register B	$09C_H$+y* 20_H	U	SE, SV	Application Reset	8628
P40_ACCGRPy_VM (y=0-7)	VM access enable register	$0A0_H$+y* 20_H	U	SE, SV	Application Reset	8629
P40_ACCGRPy_PRS (y=0-7)	PRS access enable register	$0A4_H$+y* 20_H	U	SE, SV	Application Reset	8630
P40_ACCGRPy_PROTE (y=0-7)	PROT Register Endinit	$0AC_H$+y* 20_H	U	SV, PROT	Application Reset	8630
P40_PADCFGp_GPIO (p=0-15)	Port 40 GPIO Control Register for pin p	300_H+p* 10_H	PPSp	PPSp	8632	8632
P40_PADCFGp_DRVCFG (p=0-15)	Port 40 Pad configuration Control Register for padp	304_H+p* 10_H	PPSp	EPSp, SV, PPSp	See 8760	8760
P40_PADCFGp_ACCEN (p=0-15)	Port 40 Pad Access Control Register for padp	308_H+p* 10_H	U	SE, SV	Application Reset	8638
P40_PADCFGp_SAFSEC (p=0-15)	Port 40 Pad Safety and Security Control Register for padp	$30C_H$+p* 10_H	PPSp	EPSp, SV, PPSp	See 8771	8771

47.8.4.25 Port 04 Wake Enable Register

The wake enable register contains the enable of the wakeup circuitry of the dedicated pad. If the pad is enabled for wakeup it will act as wakeup source in STANDBY mode.

P04_WKEN — Offset address: 010_H

Port 04 Wake Enable Register — Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	WKEN9	WKEN8	WKEN7	WKEN6	WKEN5	WKEN4	WKEN3	WKEN2	WKEN1	WKEN0
r	r	r	r	r	r	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
WKENx (x=0-9)	x	rwh	**Wake Enable Bit x** This bit enables or disables the wakeup at the pin Pn.x. 0_B The wakeup capability of Pn.x is disabled. 1_B The wakeup capability of Pn.x is enabled.
0	15, 14, 13, 12, 11, 10, 31:16	r	**Reserved** Read as 0; should be written with 0.

47.8.4.26 Port 20 Wake Enable Register

P20_WKEN — Offset address: 010_H

Port 20 Wake Enable Register — Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	WKEN14	WKEN13	WKEN12	WKEN11	WKEN10	WKEN9	WKEN8	WKEN7	WKEN6	WKEN5	WKEN4	WKEN3	WKEN2	WKEN1	WKEN0
r	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
WKENx (x=0-14)	x	rwh	**Wake Enable Bit x** This bit enables or disables the wakeup at the pin Pn.x. 0_B The wakeup capability of Pn.x is disabled. 1_B The wakeup capability of Pn.x is enabled.
0	15, 31:16	r	**Reserved** Read as 0; should be written with 0.

47.8.4.27 Port port_no Wake Enable Register (port_no=21,23,31-32)

P21_WKEN Offset address: 010_H

Port 21 Wake Enable Register Application Reset value: $0000\ 0000_H$

P23_WKEN Offset address: 010_H

Port 23 Wake Enable Register Application Reset value: $0000\ 0000_H$

P31_WKEN Offset address: 010_H

Port 31 Wake Enable Register Application Reset value: $0000\ 0000_H$

P32_WKEN Offset address: 010_H

Port 32 Wake Enable Register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	WKEN7	WKEN6	WKEN5	WKEN4	WKEN3	WKEN2	WKEN1	WKEN0
r	r	r	r	r	r	r	r	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
WKENx (x=0-7)	x	rwh	**Wake Enable Bit x** This bit enables or disables the wakeup at the pin Pn.x. 0_B The wakeup capability of Pn.x is disabled. 1_B The wakeup capability of Pn.x is enabled.
0	15, 14, 13, 12, 11, 10, 9, 8, 31:16	r	**Reserved** Read as 0; should be written with 0.

47.8.4.28 Port 22 Wake Enable Register

P22_WKEN Offset address: 010_H

Port 22 Wake Enable Register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	WKEN11	WKEN10	WKEN9	WKEN8	WKEN7	WKEN6	WKEN5	WKEN4	WKEN3	WKEN2	WKEN1	WKEN0
r	r	r	r	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
WKENx (x=0-11)	x	rwh	**Wake Enable Bit x** This bit enables or disables the wakeup at the pin Pn.x. 0_B The wakeup capability of Pn.x is disabled. 1_B The wakeup capability of Pn.x is enabled.
0	15, 14, 13, 12, 31:16	r	**Reserved** Read as 0; should be written with 0.

47.8.4.29 Port 25 Wake Enable Register

P25_WKEN Offset address: 010_H

Port 25 Wake Enable Register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	WKEN1	WKEN0
r	r	r	r	r	r	r	r	r	r	r	r	r	r	rwh	rwh

Field	Bits	Type	Description
WKENx (x=0-1)	x	rwh	**Wake Enable Bit x** This bit enables or disables the wakeup at the pin Pn.x. 0_B The wakeup capability of Pn.x is disabled. 1_B The wakeup capability of Pn.x is enabled.
0	15, 14, 13, 12, 11, 10, 9, 8, 7, 6, 5, 4, 3, 2, 31:16	r	**Reserved** Read as 0; should be written with 0.

47.8.4.30 Port port_no Wake Enable Register (port_no=34-35)

P34_WKEN Offset address: 010_H

Port 34 Wake Enable Register Application Reset value: $0000\ 0000_H$

P35_WKEN Offset address: 010_H

Port 35 Wake Enable Register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	WKEN5	WKEN4	WKEN3	WKEN2	WKEN1	WKEN0
r	r	r	r	r	r	r	r	r	r	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
WKENx (x=0-5)	x	rwh	**Wake Enable Bit x** This bit enables or disables the wakeup at the pin Pn.x. 0_B The wakeup capability of Pn.x is disabled. 1_B The wakeup capability of Pn.x is enabled.
0	15, 14, 13, 12, 11, 10, 9, 8, 7, 6, 31:16	r	**Reserved** Read as 0; should be written with 0.

47.8.4.31 Port 04 Wake Status Register

The wake status register contains the wakeup information provided by the wakeup circuitry of a dedicated pad.

P04_WKSTS Offset address: 014_H

Port 04 Wake Status Register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	WKSTS9	WKSTS8	WKSTS7	WKSTS6	WKSTS5	WKSTS4	WKSTS3	WKSTS2	WKSTS1	WKSTS0
r	r	r	r	r	r	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
WKSTSx (x=0-9)	x	rh	**Wake Status Bit x** This bit indicates the wakeup status at the input pin Pn.x. 0_B No wakeup triggered by Pn.x. 1_B Wakeup triggered by Pn.x.
0	15, 14, 13, 12, 11, 10, 31:16	r	**Reserved** Read as 0; should be written with 0.

47.8.4.32 Port 20 Wake Status Register

P20_WKSTS Offset address: 014_H

Port 20 Wake Status Register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	WKSTS14	WKSTS13	WKSTS12	WKSTS11	WKSTS10	WKSTS9	WKSTS8	WKSTS7	WKSTS6	WKSTS5	WKSTS4	WKSTS3	WKSTS2	WKSTS1	WKSTS0
r	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
WKSTSx (x=0-14)	x	rh	**Wake Status Bit x** This bit indicates the wakeup status at the input pin Pn.x. 0_B No wakeup triggered by Pn.x. 1_B Wakeup triggered by Pn.x.
0	15, 31:16	r	**Reserved** Read as 0; should be written with 0.

47.8.4.33 Port port_no Wake Status Register (port_no=21,23,31-32)

P21_WKSTS Offset address: 014_H

Port 21 Wake Status Register Application Reset value: $0000\ 0000_H$

P23_WKSTS Offset address: 014_H

Port 23 Wake Status Register Application Reset value: $0000\ 0000_H$

P31_WKSTS Offset address: 014_H

Port 31 Wake Status Register Application Reset value: $0000\ 0000_H$

P32_WKSTS Offset address: 014_H

Port 32 Wake Status Register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	WKSTS7	WKSTS6	WKSTS5	WKSTS4	WKSTS3	WKSTS2	WKSTS1	WKSTS0
r	r	r	r	r	r	r	r	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
WKSTSx (x=0-7)	x	rh	**Wake Status Bit x** This bit indicates the wakeup status at the input pin Pn.x. 0_B No wakeup triggered by Pn.x. 1_B Wakeup triggered by Pn.x.
0	15, 14, 13, 12, 11, 10, 9, 8, 31:16	r	**Reserved** Read as 0; should be written with 0.

47.8.4.34 Port 22 Wake Status Register

P22_WKSTS Offset address: 014_H

Port 22 Wake Status Register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	WKSTS11	WKSTS10	WKSTS9	WKSTS8	WKSTS7	WKSTS6	WKSTS5	WKSTS4	WKSTS3	WKSTS2	WKSTS1	WKSTS0
r	r	r	r	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
WKSTSx (x=0-11)	x	rh	**Wake Status Bit x** This bit indicates the wakeup status at the input pin Pn.x. 0_B No wakeup triggered by Pn.x. 1_B Wakeup triggered by Pn.x.
0	15, 14, 13, 12, 31:16	r	**Reserved** Read as 0; should be written with 0.

47.8.4.35 Port 25 Wake Status Register

P25_WKSTS Offset address: 014_H

Port 25 Wake Status Register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	WKSTS1	WKSTS0
r	r	r	r	r	r	r	r	r	r	r	r	r	r	rh	rh

Field	Bits	Type	Description
WKSTSx (x=0-1)	x	rh	**Wake Status Bit x** This bit indicates the wakeup status at the input pin Pn.x. 0_B No wakeup triggered by Pn.x. 1_B Wakeup triggered by Pn.x.
0	15, 14, 13, 12, 11, 10, 9, 8, 7, 6, 5, 4, 3, 2, 31:16	r	**Reserved** Read as 0; should be written with 0.

47.8.4.36 Port port_no Wake Status Register (port_no=34-35)

P34_WKSTS Offset address: 014_H

Port 34 Wake Status Register Application Reset value: $0000\ 0000_H$

P35_WKSTS Offset address: 014_H

Port 35 Wake Status Register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	WKSTS5	WKSTS4	WKSTS3	WKSTS2	WKSTS1	WKSTS0
r	r	r	r	r	r	r	r	r	r	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
WKSTSx (x=0-5)	x	rh	**Wake Status Bit x** This bit indicates the wakeup status at the input pin Pn.x. 0_B No wakeup triggered by Pn.x. 1_B Wakeup triggered by Pn.x.
0	15, 14, 13, 12, 11, 10, 9, 8, 7, 6, 31:16	r	**Reserved** Read as 0; should be written with 0.

47.8.4.37 Port 04 Wake Enable Status Register

The wake enable status reflects the status of the wakeup circuitry. It is set when the corresponding WKEN is set to one and indicates that wakeup cicuitry is available and STANDBY mode can be entered.

P04_WKENSTS Offset address: 018_H

Port 04 Wake Enable Status Register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	WKENS9	WKENS8	WKENS7	WKENS6	WKENS5	WKENS4	WKENS3	WKENS2	WKENS1	WKENS0
r	r	r	r	r	r	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
WKENSx (x=0-9)	x	rh	**Wake Enable Status Bit x** This bit indicates if the enable of a wakeup at the pin Pn.x was succesful. 0_B The enable of a wakeup at Pn.x was not successful. 1_B The enable of a wakeup at Pn.x was successful.
0	15, 14, 13, 12, 11, 10, 31:16	r	**Reserved** Read as 0; should be written with 0.

47.8.4.38 Port 20 Wake Enable Status Register

P20_WKENSTS Offset address: 018_H

Port 20 Wake Enable Status Register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	WKENS14	WKENS13	WKENS12	WKENS11	WKENS10	WKENS9	WKENS8	WKENS7	WKENS6	WKENS5	WKENS4	WKENS3	WKENS2	WKENS1	WKENS0
r	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
WKENSx (x=0-14)	x	rh	**Wake Enable Status Bit x** This bit indicates if the enable of a wakeup at the pin Pn.x was succesful. 0_B The enable of a wakeup at Pn.x was not successful. 1_B The enable of a wakeup at Pn.x was successful.
0	15, 31:16	r	**Reserved** Read as 0; should be written with 0.

47.8.4.39 Port port_no Wake Enable Status Register (port_no=21,23,31-32)

P21_WKENSTS Offset address: 018_H

Port 21 Wake Enable Status Register Application Reset value: $0000\ 0000_H$

P23_WKENSTS Offset address: 018_H

Port 23 Wake Enable Status Register Application Reset value: $0000\ 0000_H$

P31_WKENSTS Offset address: 018_H

Port 31 Wake Enable Status Register Application Reset value: $0000\ 0000_H$

P32_WKENSTS Offset address: 018_H

Port 32 Wake Enable Status Register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	WKENS7	WKENS6	WKENS5	WKENS4	WKENS3	WKENS2	WKENS1	WKENS0
r	r	r	r	r	r	r	r	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
WKENSx (x=0-7)	x	rh	**Wake Enable Status Bit x** This bit indicates if the enable of a wakeup at the pin Pn.x was succesful. 0_B The enable of a wakeup at Pn.x was not successful. 1_B The enable of a wakeup at Pn.x was successful.
0	15, 14, 13, 12, 11, 10, 9, 8, 31:16	r	**Reserved** Read as 0; should be written with 0.

47.8.4.40 Port 22 Wake Enable Status Register

P22_WKENSTS Offset address: 018_H

Port 22 Wake Enable Status Register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	WKENS11	WKENS10	WKENS9	WKENS8	WKENS7	WKENS6	WKENS5	WKENS4	WKENS3	WKENS2	WKENS1	WKENS0
r	r	r	r	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
WKENSx (x=0-11)	x	rh	**Wake Enable Status Bit x** This bit indicates if the enable of a wakeup at the pin Pn.x was succesful. 0_B The enable of a wakeup at Pn.x was not successful. 1_B The enable of a wakeup at Pn.x was successful.
0	15, 14, 13, 12, 31:16	r	**Reserved** Read as 0; should be written with 0.

47.8.4.41 Port 25 Wake Enable Status Register

P25_WKENSTS Offset address: 018_H

Port 25 Wake Enable Status Register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	WKENS1	WKENS0
r	r	r	r	r	r	r	r	r	r	r	r	r	r	rh	rh

Field	Bits	Type	Description
WKENSx (x=0-1)	x	rh	**Wake Enable Status Bit x** This bit indicates if the enable of a wakeup at the pin Pn.x was succesful. 0_B The enable of a wakeup at Pn.x was not successful. 1_B The enable of a wakeup at Pn.x was successful.
0	15, 14, 13, 12, 11, 10, 9, 8, 7, 6, 5, 4, 3, 2, 31:16	r	**Reserved** Read as 0; should be written with 0.

47.8.4.42 Port port_no Wake Enable Status Register (port_no=34-35)

P34_WKENSTS Offset address: 018_H

Port 34 Wake Enable Status Register Application Reset value: $0000\ 0000_H$

P35_WKENSTS Offset address: 018_H

Port 35 Wake Enable Status Register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	WKENS5	WKENS4	WKENS3	WKENS2	WKENS1	WKENS0
r	r	r	r	r	r	r	r	r	r	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
WKENSx (x=0-5)	x	rh	**Wake Enable Status Bit x** This bit indicates if the enable of a wakeup at the pin Pn.x was succesful. 0_B The enable of a wakeup at Pn.x was not successful. 1_B The enable of a wakeup at Pn.x was successful.
0	15, 14, 13, 12, 11, 10, 9, 8, 7, 6, 31:16	r	**Reserved** Read as 0; should be written with 0.

47.8.4.43 Port 04 Wake Status Clear Register

The wake status clear register contains the clear flags used to clear the wake enable status information.

P04_WKENSTSCLR Offset address: $01C_H$

Port 04 Wake Status Clear Register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	WKENSC9	WKENSC8	WKENSC7	WKENSC6	WKENSC5	WKENSC4	WKENSC3	WKENSC2	WKENSC1	WKENSC0
r	r	r	r	r	r	w	w	w	w	w	w	w	w	w	w

Field	Bits	Type	Description
WKENSCx (x=0-9)	x	w	**Wake Enable Status Clear Bit x** This bit clears the WKENSTS bit of Pin Pn.x. 0_B No clear of WKENSTS of Pin Pn.x. 1_B Clear of WKENSTS of Pin Pn.x.
0	15, 14, 13, 12, 11, 10, 31:16	r	**Reserved** Read as 0; should be written with 0.

47.8.4.44 Port 20 Wake Status Clear Register

P20_WKENSTSCLR Offset address: $01C_H$

Port 20 Wake Status Clear Register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	WKENSC14	WKENSC13	WKENSC12	WKENSC11	WKENSC10	WKENSC9	WKENSC8	WKENSC7	WKENSC6	WKENSC5	WKENSC4	WKENSC3	WKENSC2	WKENSC1	WKENSC0
r	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

Field	Bits	Type	Description
WKENSCx (x=0-14)	x	w	**Wake Enable Status Clear Bit x** This bit clears the WKENSTS bit of Pin Pn.x. 0_B No clear of WKENSTS of Pin Pn.x. 1_B Clear of WKENSTS of Pin Pn.x.
0	15, 31:16	r	**Reserved** Read as 0; should be written with 0.

47.8.4.45 Port port_no Wake Status Clear Register (port_no=21,23,31-32)

P21_WKENSTSCLR Offset address: $01C_H$

Port 21 Wake Status Clear Register Application Reset value: $0000\ 0000_H$

P23_WKENSTSCLR Offset address: $01C_H$

Port 23 Wake Status Clear Register Application Reset value: $0000\ 0000_H$

P31_WKENSTSCLR Offset address: $01C_H$

Port 31 Wake Status Clear Register Application Reset value: $0000\ 0000_H$

P32_WKENSTSCLR Offset address: $01C_H$

Port 32 Wake Status Clear Register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	WKENSC7	WKENSC6	WKENSC5	WKENSC4	WKENSC3	WKENSC2	WKENSC1	WKENSC0
r	r	r	r	r	r	r	r	w	w	w	w	w	w	w	w

Field	Bits	Type	Description
WKENSCx (x=0-7)	x	w	**Wake Enable Status Clear Bit x** This bit clears the WKENSTS bit of Pin Pn.x. 0_B No clear of WKENSTS of Pin Pn.x. 1_B Clear of WKENSTS of Pin Pn.x.
0	15, 14, 13, 12, 11, 10, 9, 8, 31:16	r	**Reserved** Read as 0; should be written with 0.

47.8.4.46 Port 22 Wake Status Clear Register

P22_WKENSTSCLR Offset address: $01C_H$

Port 22 Wake Status Clear Register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

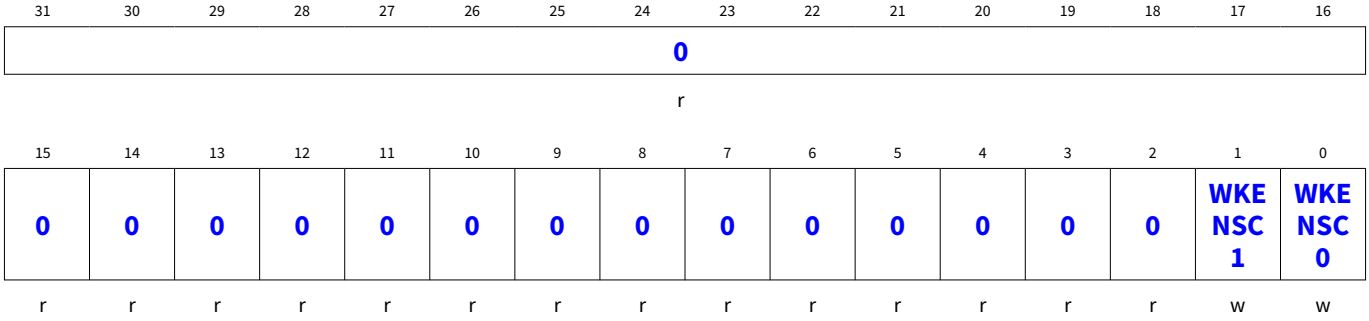
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	WKENSC11	WKENSC10	WKENSC9	WKENSC8	WKENSC7	WKENSC6	WKENSC5	WKENSC4	WKENSC3	WKENSC2	WKENSC1	WKENSC0
r	r	r	r	w	w	w	w	w	w	w	w	w	w	w	w

Field	Bits	Type	Description
WKENSCx (x=0-11)	x	w	**Wake Enable Status Clear Bit x** This bit clears the WKENSTS bit of Pin Pn.x. 0_B No clear of WKENSTS of Pin Pn.x. 1_B Clear of WKENSTS of Pin Pn.x.
0	15, 14, 13, 12, 31:16	r	**Reserved** Read as 0; should be written with 0.

47.8.4.47 Port 25 Wake Status Clear Register

P25_WKENSTSCLR Offset address: $01C_H$

Port 25 Wake Status Clear Register Application Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
WKENSCx (x=0-1)	x	w	**Wake Enable Status Clear Bit x** This bit clears the WKENSTS bit of Pin Pn.x. 0_B No clear of WKENSTS of Pin Pn.x. 1_B Clear of WKENSTS of Pin Pn.x.
0	15, 14, 13, 12, 11, 10, 9, 8, 7, 6, 5, 4, 3, 2, 31:16	r	**Reserved** Read as 0; should be written with 0.

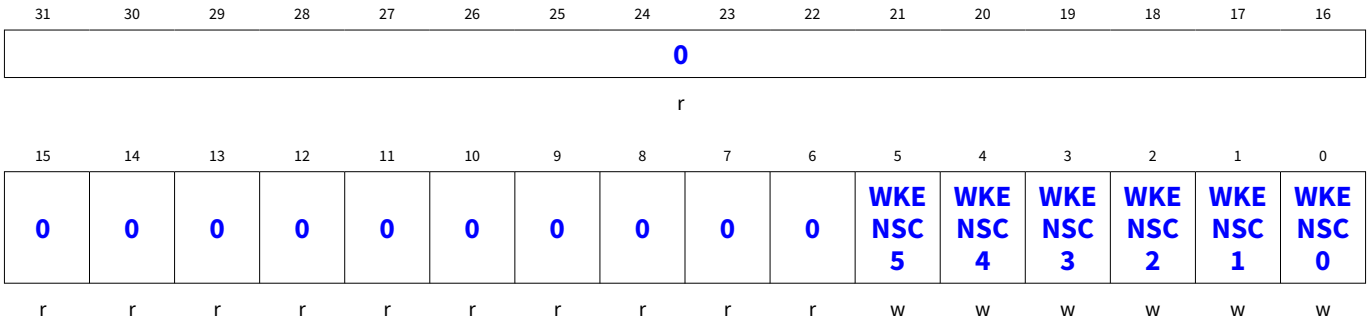
47.8.4.48 Port port_no Wake Status Clear Register (port_no=34-35)

P34_WKENSTSCLR Offset address: $01C_H$

Port 34 Wake Status Clear Register Application Reset value: $0000\ 0000_H$

P35_WKENSTSCLR Offset address: $01C_H$

Port 35 Wake Status Clear Register Application Reset value: $0000\ 0000_H$



Field	Bits	Type	Description
WKENSCx (x=0-5)	x	w	**Wake Enable Status Clear Bit x** This bit clears the WKENSTS bit of Pin Pn.x. 0_B No clear of WKENSTS of Pin Pn.x. 1_B Clear of WKENSTS of Pin Pn.x.

(table continues...)

(continued)

Field	Bits	Type	Description
0	15, 14, 13, 12, 11, 10, 9, 8, 7, 6, 31:16	r	**Reserved** Read as 0; should be written with 0.

47.8.4.49 Port 04 Output Register

The port output register determines the value of a GPIO pin when it is selected by Pn_DRVCFGxx as output. Writing a 0 to a Pn_OUT.Px (x = 0-15) bit position delivers a low level at the corresponding output pin. A high level is output when the corresponding bit is written with a 1. Note that the bits of Pn_OUT.Px can be individually set or cleared by writing appropriate values into the port output modification set register Pn_OMSR or port output modification clear register Pn_OMCR, respectively. The Pn_OUT.Px bits can also be set, cleared or toggled with register Pn_OMR within the same write operation.

P04_OUT Offset address: 020_H

Port 04 Output Register Reset values see: Table 2472

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	P9	P8	P7	P6	P5	P4	P3	P2	P1	P0
r	r	r	r	r	r	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
Px (x=0-9)	x	rwh	**Input Bit x** This bit indicates the level at the output pin Pn.x. 0_B The output level of Pn.x is 0. 1_B The output level of Pn.x is 1.
0	15, 14, 13, 12, 11, 10, 31:16	r	**Reserved** Read as 0; should be written with 0.

Table 2472 Reset values of P04_OUT

Reset type	Reset value	Note
Application Reset	$0000\ 0000_H$	
Kernel Reset	$0000\ 0000_H$	

47.8.4.50 Port port_no Output Register (port_no=16,20)

P16_OUT Offset address: 020_H

Port 16 Output Register Reset values see: Table 2473

P20_OUT Offset address: 020_H

Port 20 Output Register Reset values see: Table 2473

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	P14	P13	P12	P11	P10	P9	P8	P7	P6	P5	P4	P3	P2	P1	P0
r	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
Px (x=0-14)	x	rwh	**Input Bit x** This bit indicates the level at the output pin Pn.x. 0_B The output level of Pn.x is 0. 1_B The output level of Pn.x is 1.
0	15, 31:16	r	**Reserved** Read as 0; should be written with 0.

Table 2473 Reset values

Applies to P16_OUT

Applies to P20_OUT

Reset type	Reset value	Note
Application Reset	0000 0000$_H$	
Kernel Reset	0000 0000$_H$	

47.8.4.51 Port port_no Output Register (port_no=21,23,31-32)

P21_OUT Offset address: 020$_H$

Port 21 Output Register Reset values see: Table 2474

P23_OUT Offset address: 020$_H$

Port 23 Output Register Reset values see: Table 2474

P31_OUT Offset address: 020$_H$

Port 31 Output Register Reset values see: Table 2474

P32_OUT Offset address: 020$_H$

Port 32 Output Register Reset values see: Table 2474

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	P7	P6	P5	P4	P3	P2	P1	P0
r	r	r	r	r	r	r	r	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
Px (x=0-7)	x	rwh	**Input Bit x** This bit indicates the level at the output pin Pn.x. 0_B The output level of Pn.x is 0. 1_B The output level of Pn.x is 1.
0	15, 14, 13, 12, 11, 10, 9, 8, 31:16	r	**Reserved** Read as 0; should be written with 0.

Table 2474 Reset values

Applies to P21_OUT
Applies to P23_OUT
Applies to P31_OUT
Applies to P32_OUT

Reset type	Reset value	Note
Application Reset	0000 0000$_H$	
Kernel Reset	0000 0000$_H$	

47.8.4.52 Port 22 Output Register

P22_OUT Offset address: 020$_H$

Port 22 Output Register Reset values see: Table 2475

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	P11	P10	P9	P8	P7	P6	P5	P4	P3	P2	P1	P0
r	r	r	r	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
Px (x=0-11)	x	rwh	**Input Bit x** This bit indicates the level at the output pin Pn.x. 0_B The output level of Pn.x is 0. 1_B The output level of Pn.x is 1.
0	15, 14, 13, 12, 31:16	r	**Reserved** Read as 0; should be written with 0.

Table 2475 Reset values of P22_OUT

Reset type	Reset value	Note
Application Reset	0000 0000$_H$	
Kernel Reset	0000 0000$_H$	

47.8.4.53 Port 25 Output Register

P25_OUT Offset address: 020$_H$

Port 25 Output Register Reset values see: Table 2476

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	P1	P0
r	r	r	r	r	r	r	r	r	r	r	r	r	r	rwh	rwh

Field	Bits	Type	Description
Px (x=0-1)	x	rwh	**Input Bit x** This bit indicates the level at the output pin Pn.x. 0_B The output level of Pn.x is 0. 1_B The output level of Pn.x is 1.
0	15, 14, 13, 12, 11, 10, 9, 8, 7, 6, 5, 4, 3, 2, 31:16	r	**Reserved** Read as 0; should be written with 0.

Table 2476 Reset values of P25_OUT

Reset type	Reset value	Note
Application Reset	0000 0000$_H$	
Kernel Reset	0000 0000$_H$	

47.8.4.54 Port port_no Output Register (port_no=34-35)

P34_OUT Offset address: 020$_H$

Port 34 Output Register Reset values see: Table 2477

P35_OUT Offset address: 020$_H$

Port 35 Output Register Reset values see: Table 2477

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	P5	P4	P3	P2	P1	P0
r	r	r	r	r	r	r	r	r	r	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
Px (x=0-5)	x	rwh	**Input Bit x** This bit indicates the level at the output pin Pn.x. 0_B The output level of Pn.x is 0. 1_B The output level of Pn.x is 1.
0	15, 14, 13, 12, 11, 10, 9, 8, 7, 6, 31:16	r	**Reserved** Read as 0; should be written with 0.

Table 2477 Reset values

Applies to P34_OUT

Applies to P35_OUT

Reset type	Reset value	Note
Application Reset	0000 0000$_H$	
Kernel Reset	0000 0000$_H$	

47.8.4.55 Port 04 Input Register

The logic level of a GPIO pin can be read via the read-only port input register Pn_IN.Reading the Pn_IN register always returns the current logical value at the GPIO pin independently whether the pin is selected as input or output.

P04_IN Offset address: 024$_H$

Port 04 Input Register Reset values see: Table 2478

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	P9	P8	P7	P6	P5	P4	P3	P2	P1	P0
r	r	r	r	r	r	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
Px (x=0-9)	x	rh	**Input Bit x** This bit indicates the level at the input pin Pn.x. 0_B The input level of Pn.x is 0. 1_B The input level of Pn.x is 1.
0	15, 14, 13, 12, 11, 10, 31:16	r	**Reserved** Read as 0; should be written with 0.

Table 2478 Reset values of P04_IN

Reset type	Reset value	Note
Application Reset	0000 0000_H	
Kernel Reset	0000 0000_H	

47.8.4.56 Port port_no Input Register (port_no=16,20)

P16_IN Offset address: 024_H

Port 16 Input Register Reset values see: Table 2479

P20_IN Offset address: 024_H

Port 20 Input Register Reset values see: Table 2479

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	P14	P13	P12	P11	P10	P9	P8	P7	P6	P5	P4	P3	P2	P1	P0
r	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
Px (x=0-14)	x	rh	**Input Bit x** This bit indicates the level at the input pin Pn.x. 0_B The input level of Pn.x is 0. 1_B The input level of Pn.x is 1.
0	15, 31:16	r	**Reserved** Read as 0; should be written with 0.

Table 2479 Reset values

Applies to P16_IN

Applies to P20_IN

Reset type	Reset value	Note
Application Reset	0000 0000_H	
Kernel Reset	0000 0000_H	

47.8.4.57 Port port_no Input Register (port_no=21,23,31-32)

P21_IN Offset address: 024_H

Port 21 Input Register Reset values see: Table 2480

P23_IN Offset address: 024_H

Port 23 Input Register Reset values see: Table 2480

P31_IN Offset address: 024_H

Port 31 Input Register Reset values see: Table 2480

P32_IN Offset address: 024_H

Port 32 Input Register Reset values see: Table 2480

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	P7	P6	P5	P4	P3	P2	P1	P0
r	r	r	r	r	r	r	r	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
Px (x=0-7)	x	rh	**Input Bit x** This bit indicates the level at the input pin Pn.x. 0_B The input level of Pn.x is 0. 1_B The input level of Pn.x is 1.
0	15, 14, 13, 12, 11, 10, 9, 8, 31:16	r	**Reserved** Read as 0; should be written with 0.

Table 2480 Reset values

Applies to P21_IN
Applies to P23_IN
Applies to P31_IN
Applies to P32_IN

Reset type	Reset value	Note
Application Reset	$0000\ 0000_H$	
Kernel Reset	$0000\ 0000_H$	

47.8.4.58 Port 22 Input Register

P22_IN Offset address: 024_H

Port 22 Input Register Reset values see: Table 2481

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	P11	P10	P9	P8	P7	P6	P5	P4	P3	P2	P1	P0
r	r	r	r	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
Px (x=0-11)	x	rh	**Input Bit x** This bit indicates the level at the input pin Pn.x. 0_B The input level of Pn.x is 0. 1_B The input level of Pn.x is 1.
0	15, 14, 13, 12, 31:16	r	**Reserved** Read as 0; should be written with 0.

Table 2481 Reset values of P22_IN

Reset type	Reset value	Note
Application Reset	$0000\ 0000_H$	
Kernel Reset	$0000\ 0000_H$	

47.8.4.59 Port 25 Input Register

P25_IN Offset address: 024_H

Port 25 Input Register Reset values see: Table 2482

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	P1	P0
r	r	r	r	r	r	r	r	r	r	r	r	r	r	rh	rh

Field	Bits	Type	Description
Px (x=0-1)	x	rh	**Input Bit x** This bit indicates the level at the input pin Pn.x. 0_B The input level of Pn.x is 0. 1_B The input level of Pn.x is 1.
0	15, 14, 13, 12, 11, 10, 9, 8, 7, 6, 5, 4, 3, 2, 31:16	r	**Reserved** Read as 0; should be written with 0.

Table 2482 Reset values of P25_IN

Reset type	Reset value	Note
Application Reset	0000 0000$_H$	
Kernel Reset	0000 0000$_H$	

47.8.4.60 Port port_no Input Register (port_no=34-35)

P34_IN Offset address: 024$_H$

Port 34 Input Register Reset values see: Table 2483

P35_IN Offset address: 024$_H$

Port 35 Input Register Reset values see: Table 2483

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	P5	P4	P3	P2	P1	P0
r	r	r	r	r	r	r	r	r	r	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
Px (x=0-5)	x	rh	**Input Bit x** This bit indicates the level at the input pin Pn.x. 0_B The input level of Pn.x is 0. 1_B The input level of Pn.x is 1.
0	15, 14, 13, 12, 11, 10, 9, 8, 7, 6, 31:16	r	**Reserved** Read as 0; should be written with 0.

Table 2483 Reset values

Applies to P34_IN

Applies to P35_IN

Reset type	Reset value	Note
Application Reset	0000 0000$_H$	
Kernel Reset	0000 0000$_H$	

47.8.4.61 Port 04 HWSEL status Register

The register HWSELSTAT allows reading back the status of the HW selection lines of all the pins.

This allows the SW to determinate if the Pin state is driven by the SW selection or if a HW module like e.g. LETHx or CSRM took over the control over any signal in the port.

Note: An active Emergency stop request is not reflected in this status register

P04_HWSELSTAT Offset address: 028$_H$

Port 04 HWSEL status Register Reset values see: Table 2484

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	STAT 9	STAT 8	STAT 7	STAT 6	STAT 5	STAT 4	STAT 3	STAT 2	STAT 1	STAT 0
r	r	r	r	r	r	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
STATx (x=0-9)	x	rh	**Status of HWSEL line for Pin x** Status of HWSEL line x
0	15, 14, 13, 12, 11, 10, 31:16	r	**Reserved** Read as 0; should be written with 0.

Table 2484 Reset values of P04_HWSELSTAT

Reset type	Reset value	Note
Application Reset	$0000\ 0000_H$	
Kernel Reset	$0000\ 0000_H$	

47.8.4.62 Port port_no HWSEL status Register (port_no=16,20)

P16_HWSELSTAT Offset address: 028_H

Port 16 HWSEL status Register Reset values see: Table 2485

P20_HWSELSTAT Offset address: 028_H

Port 20 HWSEL status Register Reset values see: Table 2485

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	STAT 14	STAT 13	STAT 12	STAT 11	STAT 10	STAT 9	STAT 8	STAT 7	STAT 6	STAT 5	STAT 4	STAT 3	STAT 2	STAT 1	STAT 0
r	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
STATx (x=0-14)	x	rh	**Status of HWSEL line for Pin x** Status of HWSEL line x
0	15, 31:16	r	**Reserved** Read as 0; should be written with 0.

Table 2485 Reset values

Applies to P16_HWSELSTAT

Applies to P20_HWSELSTAT

Reset type	Reset value	Note
Application Reset	0000 0000$_H$	
Kernel Reset	0000 0000$_H$	

47.8.4.63 Port port_no HWSEL status Register (port_no=21,23,31-32)

P21_HWSELSTAT Offset address: 028$_H$

Port 21 HWSEL status Register Reset values see: Table 2486

P23_HWSELSTAT Offset address: 028$_H$

Port 23 HWSEL status Register Reset values see: Table 2486

P31_HWSELSTAT Offset address: 028$_H$

Port 31 HWSEL status Register Reset values see: Table 2486

P32_HWSELSTAT Offset address: 028$_H$

Port 32 HWSEL status Register Reset values see: Table 2486

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	STAT 7	STAT 6	STAT 5	STAT 4	STAT 3	STAT 2	STAT 1	STAT 0
r	r	r	r	r	r	r	r	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
STATx (x=0-7)	x	rh	**Status of HWSEL line for Pin x** Status of HWSEL line x
0	15, 14, 13, 12, 11, 10, 9, 8, 31:16	r	**Reserved** Read as 0; should be written with 0.

Table 2486 Reset values

Applies to P21_HWSELSTAT

Applies to P23_HWSELSTAT

Applies to P31_HWSELSTAT

Applies to P32_HWSELSTAT

Reset type	Reset value	Note
Application Reset	0000 0000$_H$	
Kernel Reset	0000 0000$_H$	

47.8.4.64 Port 22 HWSEL status Register

P22_HWSELSTAT Offset address: 028_H

Port 22 HWSEL status Register Reset values see: Table 2487

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	STAT 11	STAT 10	STAT 9	STAT 8	STAT 7	STAT 6	STAT 5	STAT 4	STAT 3	STAT 2	STAT 1	STAT 0
r	r	r	r	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
STATx (x=0-11)	x	rh	**Status of HWSEL line for Pin x** Status of HWSEL line x
0	15, 14, 13, 12, 31:16	r	**Reserved** Read as 0; should be written with 0.

Table 2487 Reset values of P22_HWSELSTAT

Reset type	Reset value	Note
Application Reset	$0000\ 0000_H$	
Kernel Reset	$0000\ 0000_H$	

47.8.4.65 Port 25 HWSEL status Register

P25_HWSELSTAT Offset address: 028_H

Port 25 HWSEL status Register Reset values see: Table 2488

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	STAT 1	STAT 0
r	r	r	r	r	r	r	r	r	r	r	r	r	r	rh	rh

Field	Bits	Type	Description
STATx (x=0-1)	x	rh	**Status of HWSEL line for Pin x** Status of HWSEL line x

(table continues...)

(continued)

Field	Bits	Type	Description
0	15, 14, 13, 12, 11, 10, 9, 8, 7, 6, 5, 4, 3, 2, 31:16	r	**Reserved** Read as 0; should be written with 0.

Table 2488 Reset values of P25_HWSELSTAT

Reset type	Reset value	Note
Application Reset	0000 0000$_H$	
Kernel Reset	0000 0000$_H$	

47.8.4.66 Port port_no HWSEL status Register (port_no=34-35)

P34_HWSELSTAT Offset address: 028$_H$

Port 34 HWSEL status Register Reset values see: Table 2489

P35_HWSELSTAT Offset address: 028$_H$

Port 35 HWSEL status Register Reset values see: Table 2489

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	STAT 5	STAT 4	STAT 3	STAT 2	STAT 1	STAT 0
r	r	r	r	r	r	r	r	r	r	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
STATx (x=0-5)	x	rh	**Status of HWSEL line for Pin x** Status of HWSEL line x
0	15, 14, 13, 12, 11, 10, 9, 8, 7, 6, 31:16	r	**Reserved** Read as 0; should be written with 0.

Table 2489 Reset values

Applies to P34_HWSELSTAT

Applies to P35_HWSELSTAT

Reset type	Reset value	Note
Application Reset	0000 0000$_H$	
Kernel Reset	0000 0000$_H$	

47.8.4.67 Port port_no Pin Function Decision Control Register (port_no=00-03,10,13-33)

The pad structure of the GPIO lines offers the possibility to disable/enable port pad, select digital input or analog ADC input functionalities. Note that Class S pads have different characteristics than other digital input pads. For analog inputs, PDISx shall be set to 1 to disable the Schmitt trigger input buffer and digital output driver (if exists). Otherwise the specified accuracy of analog functions on overlaid pins (see TMADC accuracy parameters and their footnotes in datasheet) cannot be applied. For the ADC diagnostic features “PDD” however the corresponding PDISCx needs to be 0 to allow activation of their pull resistors. This feature can be controlled by individual bits in the Pn_PDISC register, independently from input/output and pull-up/pull-down control functionality as programmed in the Pn_DRVCFGx register. One Pn_PDISC register is assigned to each port.

***Note**: After reset, all Px_PDISC registers have the reset value of 0000 0000$_H$. The startup software disables only the pads with digital input/output functionality which are available in that package by writing the corresponding bits of not available pads to one. P40_PDISC and P41_PDISC are configured by the SSW for analog input function (kept disabled). The documented reset value shows the value in the largest package.*

P00_PDISC	Offset address:	02C$_H$
Port 00 Pin Function Decision Control Register	Reset values see:	Table 2490
P01_PDISC	Offset address:	02C$_H$
Port 01 Pin Function Decision Control Register	Reset values see:	Table 2490
P02_PDISC	Offset address:	02C$_H$
Port 02 Pin Function Decision Control Register	Reset values see:	Table 2490
P03_PDISC	Offset address:	02C$_H$
Port 03 Pin Function Decision Control Register	Reset values see:	Table 2490
P10_PDISC	Offset address:	02C$_H$
Port 10 Pin Function Decision Control Register	Reset values see:	Table 2490
P13_PDISC	Offset address:	02C$_H$
Port 13 Pin Function Decision Control Register	Reset values see:	Table 2490
P14_PDISC	Offset address:	02C$_H$
Port 14 Pin Function Decision Control Register	Reset values see:	Table 2490
P30_PDISC	Offset address:	02C$_H$
Port 30 Pin Function Decision Control Register	Reset values see:	Table 2490
P33_PDISC	Offset address:	02C$_H$
Port 33 Pin Function Decision Control Register	Reset values see:	Table 2490
P15_PDISC	Offset address:	02C$_H$
Port 15 Pin Function Decision Control Register	Reset values see:	Table 2491

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PDIS 15	PDIS 14	PDIS 13	PDIS 12	PDIS 11	PDIS 10	PDIS 9	PDIS 8	PDIS 7	PDIS 6	PDIS 5	PDIS 4	PDIS 3	PDIS 2	PDIS 1	PDIS 0
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
PDISx (x=0-15)	x	rwh	**Pin Function Decision Control for Pin x** This bit selects the function of the port pad. 0_B Digital functionality of pad Pn.x is enabled. 1_B Digital functionality (including pull resistors) of pad Pn.x is disabled. Analog input function (where this is available) can be used.
0	31:16	r	**Reserved** Read as 0; should be written with 0.

Table 2490 Reset values variant 1

Applies to P00_PDISC
Applies to P01_PDISC
Applies to P02_PDISC
Applies to P03_PDISC
Applies to P10_PDISC
Applies to P13_PDISC
Applies to P14_PDISC
Applies to P30_PDISC
Applies to P33_PDISC

Reset type	Reset value	Note
PowerOn Reset	0000 0000$_H$	
After Boot-FW Value	0000 0000$_H$	Initial value in largest package

Table 2491 Reset values of P15_PDISC

Reset type	Reset value	Note
PowerOn Reset	0000 0000$_H$	
After Boot-FW Value	0000 0200$_H$	Initial value in largest package

47.8.4.68 Port 40 Pin Function Decision Control Register

P40_PDISC Offset address: 02C$_H$

Port 40 Pin Function Decision Control Register Reset values see: Table 2492

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PDIS 15	PDIS 14	PDIS 13	PDIS 12	PDIS 11	PDIS 10	PDIS 9	PDIS 8	PDIS 7	PDIS 6	PDIS 5	PDIS 4	PDIS 3	PDIS 2	PDIS 1	PDIS 0
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
PDISx (x=0-15)	x	rwh	**Pin Function Decision Control for Pin x** This bit selects the function of the port pad. 0_B Digital functionality of pad Pn.x is enabled. 1_B Digital functionality (including pull resistors) of pad Pn.x is disabled. Analog input function (where this is available) can be used.
0	31:16	r	**Reserved** Read as 0; should be written with 0.

Table 2492 Reset values of P40_PDISC

Reset type	Reset value	Note
PowerOn Reset	0000 0000$_H$	
After Boot-FW Value	0000 7FFF$_H$	Initial value in largest package

47.8.4.69 Port 04 Pin Function Decision Control Register

P04_PDISC Offset address: 02C$_H$

Port 04 Pin Function Decision Control Register Reset values see: Table 2493

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	PDIS 9	PDIS 8	PDIS 7	PDIS 6	PDIS 5	PDIS 4	PDIS 3	PDIS 2	PDIS 1	PDIS 0
r	r	r	r	r	r	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
PDISx (x=0-9)	x	rwh	**Pin Function Decision Control for Pin x** This bit selects the function of the port pad. 0_B Digital functionality of pad Pn.x is enabled. 1_B Digital functionality (including pull resistors) of pad Pn.x is disabled. Analog input function (where this is available) can be used.
0	15, 14, 13, 12, 11, 10, 31:16	r	**Reserved** Read as 0; should be written with 0.

Table 2493 Reset values of P04_PDISC

Reset type	Reset value	Note
PowerOn Reset	0000 0000$_H$	

(table continues...)

Table 2493 (continued) Reset values of P04_PDISC

Reset type	Reset value	Note
After Boot-FW Value	0000 0000$_H$	Initial value in largest package

47.8.4.70 Port 16 Pin Function Decision Control Register

P16_PDISC Offset address: 02C$_H$

Port 16 Pin Function Decision Control Register Reset values see: Table 2494

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	PDIS 14	PDIS 13	PDIS 12	PDIS 11	PDIS 10	PDIS 9	PDIS 8	PDIS 7	PDIS 6	PDIS 5	PDIS 4	PDIS 3	PDIS 2	PDIS 1	PDIS 0
r	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
PDISx (x=0-14)	x	rwh	**Pin Function Decision Control for Pin x** This bit selects the function of the port pad. 0_B Digital functionality of pad Pn.x is enabled. 1_B Digital functionality (including pull resistors) of pad Pn.x is disabled. Analog input function (where this is available) can be used.
0	15, 31:16	r	**Reserved** Read as 0; should be written with 0.

Table 2494 Reset values of P16_PDISC

Reset type	Reset value	Note
PowerOn Reset	0000 0000$_H$	
After Boot-FW Value	0000 0000$_H$	Initial value in largest package

47.8.4.71 Port 20 Pin Function Decision Control Register

P20_PDISC Offset address: 02C$_H$

Port 20 Pin Function Decision Control Register Reset values see: Table 2495

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	PDIS 14	PDIS 13	PDIS 12	PDIS 11	PDIS 10	PDIS 9	PDIS 8	PDIS 7	PDIS 6	0	0	PDIS 3	PDIS 2	PDIS 1	PDIS 0
r	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	r	r	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
PDISx (x=0-3,6-14)	x	rwh	**Pin Function Decision Control for Pin x** This bit selects the function of the port pad. 0_B Digital functionality of pad Pn.x is enabled. 1_B Digital functionality (including pull resistors) of pad Pn.x is disabled. Analog input function (where this is available) can be used.
0	15, 5, 4, 31:16	r	**Reserved** Read as 0; should be written with 0.

Table 2495 Reset values of P20_PDISC

Reset type	Reset value	Note
PowerOn Reset	$0000\ 0000_H$	
After Boot-FW Value	$0000\ 0000_H$	Initial value in largest package

47.8.4.72 Port port_no Pin Function Decision Control Register (port_no=21,23,31-32)

P21_PDISC Offset address: $02C_H$

Port 21 Pin Function Decision Control Register Reset values see: Table 2496

P23_PDISC Offset address: $02C_H$

Port 23 Pin Function Decision Control Register Reset values see: Table 2496

P32_PDISC Offset address: $02C_H$

Port 32 Pin Function Decision Control Register Reset values see: Table 2496

P31_PDISC Offset address: $02C_H$

Port 31 Pin Function Decision Control Register Reset values see: Table 2497

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	PDIS 7	PDIS 6	PDIS 5	PDIS 4	PDIS 3	PDIS 2	PDIS 1	PDIS 0
r	r	r	r	r	r	r	r	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
PDISx (x=0-7)	x	rwh	**Pin Function Decision Control for Pin x** This bit selects the function of the port pad. 0_B Digital functionality of pad Pn.x is enabled. 1_B Digital functionality (including pull resistors) of pad Pn.x is disabled. Analog input function (where this is available) can be used.
0	15, 14, 13, 12, 11, 10, 9, 8, 31:16	r	**Reserved** Read as 0; should be written with 0.

Table 2496 Reset values variant 1

Applies to P21_PDISC
Applies to P23_PDISC
Applies to P32_PDISC

Reset type	Reset value	Note
PowerOn Reset	0000 0000$_H$	
After Boot-FW Value	0000 0000$_H$	Initial value in largest package

Table 2497 Reset values of P31_PDISC

Reset type	Reset value	Note
PowerOn Reset	0000 0000$_H$	
After Boot-FW Value	0000 003F$_H$	Initial value in largest package

47.8.4.73 Port 22 Pin Function Decision Control Register

P22_PDISC Offset address: 02C$_H$

Port 22 Pin Function Decision Control Register Reset values see: Table 2498

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	PDIS 11	PDIS 10	PDIS 9	PDIS 8	PDIS 7	PDIS 6	PDIS 5	PDIS 4	PDIS 3	PDIS 2	PDIS 1	PDIS 0
r	r	r	r	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
PDISx (x=0-11)	x	rwh	**Pin Function Decision Control for Pin x** This bit selects the function of the port pad. 0_B Digital functionality of pad Pn.x is enabled. 1_B Digital functionality (including pull resistors) of pad Pn.x is disabled. Analog input function (where this is available) can be used.
0	15, 14, 13, 12, 31:16	r	**Reserved** Read as 0; should be written with 0.

Table 2498 Reset values of P22_PDISC

Reset type	Reset value	Note
PowerOn Reset	0000 0000$_H$	
After Boot-FW Value	0000 0000$_H$	Initial value in largest package

47.8.4.74 Port 25 Pin Function Decision Control Register

P25_PDISC Offset address: 02C$_H$

Port 25 Pin Function Decision Control Register Reset values see: Table 2499

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	PDIS 1	PDIS 0
r	r	r	r	r	r	r	r	r	r	r	r	r	r	rwh	rwh

Field	Bits	Type	Description
PDISx (x=0-1)	x	rwh	**Pin Function Decision Control for Pin x** This bit selects the function of the port pad. 0_B Digital functionality of pad Pn.x is enabled. 1_B Digital functionality (including pull resistors) of pad Pn.x is disabled. Analog input function (where this is available) can be used.
0	15, 14, 13, 12, 11, 10, 9, 8, 7, 6, 5, 4, 3, 2, 31:16	r	**Reserved** Read as 0; should be written with 0.

Table 2499 Reset values of P25_PDISC

Reset type	Reset value	Note
PowerOn Reset	0000 0000$_H$	
After Boot-FW Value	0000 0000$_H$	Initial value in largest package

47.8.4.75 Port port_no Pin Function Decision Control Register (port_no=34-35)

P34_PDISC Offset address: 02C$_H$

Port 34 Pin Function Decision Control Register Reset values see: Table 2500

P35_PDISC Offset address: 02C$_H$

Port 35 Pin Function Decision Control Register Reset values see: Table 2500

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	PDIS5	PDIS4	PDIS3	PDIS2	PDIS1	PDIS0
r	r	r	r	r	r	r	r	r	r	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
PDISx (x=0-5)	x	rwh	**Pin Function Decision Control for Pin x** This bit selects the function of the port pad. 0_B Digital functionality of pad Pn.x is enabled. 1_B Digital functionality (including pull resistors) of pad Pn.x is disabled. Analog input function (where this is available) can be used.
0	15, 14, 13, 12, 11, 10, 9, 8, 7, 6, 31:16	r	**Reserved** Read as 0; should be written with 0.

Table 2500 Reset values

Applies to P34_PDISC

Applies to P35_PDISC

Reset type	Reset value	Note
PowerOn Reset	0000 0000$_H$	
After Boot-FW Value	0000 0000$_H$	Initial value in largest package

47.8.4.76 Selection Register for Module x Port 33 Control

This register enables or disables a module x to have direct port control. Modules making use of this configuration feature are: SCR, xSPI, GETHx, LETHx and HRPWMx

P33_PCSRSEL Offset address: 034_H

Selection Register for Module x Port 33 Control PowerOn Reset value: $0000\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PCSR15	PCSR14	PCSR13	PCSR12	PCSR11	PCSR10	PCSR9	PCSR8	PCSR7	PCSR6	PCSR5	PCSR4	PCSR3	PCSR2	PCSR1	PCSR0
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
PCSRx (x=0-15)	x	rwh	**Pad control selection x** This bit indicates whether a module x has control over pad Pn.x. Selection is valid only if the pad is shared with a particular module x and if the module is enabled. 0_B Main system has control over pad 1_B Control over pad is given to a particular module x
0	31:16	r	**Reserved** Read as 0; should be written with 0.

47.8.4.77 Selection Register for Module x Port 04 Control

P04_PCSRSEL Offset address: 034_H

Selection Register for Module x Port 04 Control PowerOn Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	PCSR9	PCSR8	PCSR7	PCSR6	PCSR5	PCSR4	PCSR3	PCSR2	PCSR1	PCSR0
r	r	r	r	r	r	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
PCSRx (x=0-9)	x	rwh	**Pad control selection x** This bit indicates whether a module x has control over pad Pn.x. Selection is valid only if the pad is shared with a particular module x and if the module is enabled. 0_B Main system has control over pad 1_B Control over pad is given to a particular module x
0	15, 14, 13, 12, 11, 10, 31:16	r	**Reserved** Read as 0; should be written with 0.

47.8.4.78 Selection Register for Module x Port port_no Control (port_no=16,20)

P16_PCSRSEL Offset address: 034_H

Selection Register for Module x Port 16 Control PowerOn Reset value: $0000\ 0000_H$

P20_PCSRSEL Offset address: 034_H

Selection Register for Module x Port 20 Control PowerOn Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	PCSR14	PCSR13	PCSR12	PCSR11	PCSR10	PCSR9	PCSR8	PCSR7	PCSR6	PCSR5	PCSR4	PCSR3	PCSR2	PCSR1	PCSR0
r	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
PCSRx (x=0-14)	x	rwh	**Pad control selection x** This bit indicates whether a module x has control over pad Pn.x. Selection is valid only if the pad is shared with a particular module x and if the module is enabled. 0_B Main system has control over pad 1_B Control over pad is given to a particular module x
0	15, 31:16	r	**Reserved** Read as 0; should be written with 0.

47.8.4.79 Selection Register for Module x Port port_no Control (port_no=21,23,31-32)

P21_PCSRSEL Offset address: 034_H

Selection Register for Module x Port 21 Control PowerOn Reset value: $0000\ 0000_H$

P23_PCSRSEL Offset address: 034_H

Selection Register for Module x Port 23 Control PowerOn Reset value: $0000\ 0000_H$

P31_PCSRSEL Offset address: 034_H

Selection Register for Module x Port 31 Control PowerOn Reset value: $0000\ 0000_H$

P32_PCSRSEL Offset address: 034_H

Selection Register for Module x Port 32 Control PowerOn Reset value: $0000\ 00F4_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	PCSR7	PCSR6	PCSR5	PCSR4	PCSR3	PCSR2	PCSR1	PCSR0
r	r	r	r	r	r	r	r	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
PCSRx (x=0-7)	x	rwh	**Pad control selection x** This bit indicates whether a module x has control over pad Pn.x. Selection is valid only if the pad is shared with a particular module x and if the module is enabled. 0_B Main system has control over pad 1_B Control over pad is given to a particular module x
0	15, 14, 13, 12, 11, 10, 9, 8, 31:16	r	**Reserved** Read as 0; should be written with 0.

47.8.4.80 Selection Register for Module x Port 22 Control

P22_PCSRSEL Offset address: 034_H

Selection Register for Module x Port 22 Control PowerOn Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	PCSR11	PCSR10	PCSR9	PCSR8	PCSR7	PCSR6	PCSR5	PCSR4	PCSR3	PCSR2	PCSR1	PCSR0
r	r	r	r	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
PCSRx (x=0-11)	x	rwh	**Pad control selection x** This bit indicates whether a module x has control over pad Pn.x. Selection is valid only if the pad is shared with a particular module x and if the module is enabled. 0_B Main system has control over pad 1_B Control over pad is given to a particular module x
0	15, 14, 13, 12, 31:16	r	**Reserved** Read as 0; should be written with 0.

47.8.4.81 Selection Register for Module x Port 25 Control

P25_PCSRSEL Offset address: 034_H

Selection Register for Module x Port 25 Control PowerOn Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	PCSR1	PCSR0
r	r	r	r	r	r	r	r	r	r	r	r	r	r	rwh	rwh

Field	Bits	Type	Description
PCSRx (x=0-1)	x	rwh	**Pad control selection x** This bit indicates whether a module x has control over pad Pn.x. Selection is valid only if the pad is shared with a particular module x and if the module is enabled. 0_B Main system has control over pad 1_B Control over pad is given to a particular module x
0	15, 14, 13, 12, 11, 10, 9, 8, 7, 6, 5, 4, 3, 2, 31:16	r	**Reserved** Read as 0; should be written with 0.

47.8.4.82 Selection Register for Module x Port port_no Control (port_no=34-35)

P34_PCSRSEL Offset address: 034_H

Selection Register for Module x Port 34 Control PowerOn Reset value: $0000\ 003E_H$

P35_PCSRSEL Offset address: 034_H

Selection Register for Module x Port 35 Control PowerOn Reset value: $0000\ 003F_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	PCSR5	PCSR4	PCSR3	PCSR2	PCSR1	PCSR0
r	r	r	r	r	r	r	r	r	r	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
PCSRx (x=0-5)	x	rwh	**Pad control selection x** This bit indicates whether a module x has control over pad Pn.x. Selection is valid only if the pad is shared with a particular module x and if the module is enabled. 0_B Main system has control over pad 1_B Control over pad is given to a particular module x
0	15, 14, 13, 12, 11, 10, 9, 8, 7, 6, 31:16	r	**Reserved** Read as 0; should be written with 0.

47.8.4.83 Port 04 Output Modification Register

The port output modification register contains control bits that make it possible to individually set, clear or toggle the logic state of a single port line by manipulating the output register.

P04_OMR Offset address: $03C_H$

Port 04 Output Modification Register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	PCL9	PCL8	PCL7	PCL6	PCL5	PCL4	PCL3	PCL2	PCL1	PCL0
r	r	r	r	r	r	w	w	w	w	w	w	w	w	w	w

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	PS9	PS8	PS7	PS6	PS5	PS4	PS3	PS2	PS1	PS0
r	r	r	r	r	r	w	w	w	w	w	w	w	w	w	w

Field	Bits	Type	Description
PSx (x=0-9)	x	w	**Set Bit x** Setting this bit will set or toggle the corresponding bit in the port output register Pn_OUT. Read as 0. 0_B No operation 1_B Sets or toggles Pn_OUT.Px.

(table continues...)

(continued)

Field	Bits	Type	Description
PCLx (x=0-9)	x+16	w	**Clear Bit x** Setting this bit will clear or toggle the corresponding bit in the port output register Pn_OUT. Read as 0. 0_B No operation 1_B Clears or toggles Pn_OUT.Px.
0	15, 14, 13, 12, 11, 10, 31, 30, 29, 28, 27, 26	r	**Reserved** Read as 0; should be written with 0.

***Note**: Register Pn_OMR is virtual and does not contain any flip-flop. A read action delivers the value of 0. One 8 or 16-bits write behaves as a 32-bit write padded with zeros.*

- PCL.x/PSx = 0/0 = Bit Pn_OUTPx is not changed.
- PCL.x/PSx = 0/1 = Bit Pn_OUTPx is set.
- PCL.x/PSx = 1/0 = Bit Pn_OUTPx is reset.
- PCL.x/PSx = 1/1 = Bit Pn_OUTPx is toggled.

47.8.4.84 Port port_no Output Modification Register (port_no=16,20)

P16_OMR Offset address: $03C_H$

Port 16 Output Modification Register Application Reset value: $0000\ 0000_H$

P20_OMR Offset address: $03C_H$

Port 20 Output Modification Register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	**PCL14**	**PCL13**	**PCL12**	**PCL11**	**PCL10**	**PCL9**	**PCL8**	**PCL7**	**PCL6**	**PCL5**	**PCL4**	**PCL3**	**PCL2**	**PCL1**	**PCL0**
r	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	**PS14**	**PS13**	**PS12**	**PS11**	**PS10**	**PS9**	**PS8**	**PS7**	**PS6**	**PS5**	**PS4**	**PS3**	**PS2**	**PS1**	**PS0**
r	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

Field	Bits	Type	Description
PSx (x=0-14)	x	w	**Set Bit x** Setting this bit will set or toggle the corresponding bit in the port output register Pn_OUT. Read as 0. 0_B No operation 1_B Sets or toggles Pn_OUT.Px.

(table continues...)

(continued)

Field	Bits	Type	Description
PCLx (x=0-14)	x+16	w	**Clear Bit x** Setting this bit will clear or toggle the corresponding bit in the port output register Pn_OUT. Read as 0. 0_B No operation 1_B Clears or toggles Pn_OUT.Px.
0	15, 31	r	**Reserved** Read as 0; should be written with 0.

47.8.4.85 Port port_no Output Modification Register (port_no=21,23,31-32)

P21_OMR Offset address: $03C_H$

Port 21 Output Modification Register Application Reset value: $0000\ 0000_H$

P23_OMR Offset address: $03C_H$

Port 23 Output Modification Register Application Reset value: $0000\ 0000_H$

P31_OMR Offset address: $03C_H$

Port 31 Output Modification Register Application Reset value: $0000\ 0000_H$

P32_OMR Offset address: $03C_H$

Port 32 Output Modification Register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	PCL7	PCL6	PCL5	PCL4	PCL3	PCL2	PCL1	PCL0
r	r	r	r	r	r	r	r	w	w	w	w	w	w	w	w

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	PS7	PS6	PS5	PS4	PS3	PS2	PS1	PS0
r	r	r	r	r	r	r	r	w	w	w	w	w	w	w	w

Field	Bits	Type	Description
PSx (x=0-7)	x	w	**Set Bit x** Setting this bit will set or toggle the corresponding bit in the port output register Pn_OUT. Read as 0. 0_B No operation 1_B Sets or toggles Pn_OUT.Px.
PCLx (x=0-7)	x+16	w	**Clear Bit x** Setting this bit will clear or toggle the corresponding bit in the port output register Pn_OUT. Read as 0. 0_B No operation 1_B Clears or toggles Pn_OUT.Px.

(table continues...)

(continued)

Field	Bits	Type	Description
0	15, 14, 13, 12, 11, 10, 9, 8, 31, 30, 29, 28, 27, 26, 25, 24	r	**Reserved** Read as 0; should be written with 0.

47.8.4.86 Port 22 Output Modification Register

P22_OMR Offset address: $03C_H$

Port 22 Output Modification Register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	PCL11	PCL10	PCL9	PCL8	PCL7	PCL6	PCL5	PCL4	PCL3	PCL2	PCL1	PCL0
r	r	r	r	w	w	w	w	w	w	w	w	w	w	w	w

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	PS11	PS10	PS9	PS8	PS7	PS6	PS5	PS4	PS3	PS2	PS1	PS0
r	r	r	r	w	w	w	w	w	w	w	w	w	w	w	w

Field	Bits	Type	Description
PSx (x=0-11)	x	w	**Set Bit x** Setting this bit will set or toggle the corresponding bit in the port output register Pn_OUT. Read as 0. 0_B No operation 1_B Sets or toggles Pn_OUT.Px.
PCLx (x=0-11)	x+16	w	**Clear Bit x** Setting this bit will clear or toggle the corresponding bit in the port output register Pn_OUT. Read as 0. 0_B No operation 1_B Clears or toggles Pn_OUT.Px.
0	15, 14, 13, 12, 31, 30, 29, 28	r	**Reserved** Read as 0; should be written with 0.

47.8.4.87 Port 25 Output Modification Register

P25_OMR Offset address: $03C_H$

Port 25 Output Modification Register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	PCL1	PCL0
r	r	r	r	r	r	r	r	r	r	r	r	r	r	w	w

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	PS1	PS0
r	r	r	r	r	r	r	r	r	r	r	r	r	r	w	w

Field	Bits	Type	Description
PSx (x=0-1)	x	w	**Set Bit x** Setting this bit will set or toggle the corresponding bit in the port output register Pn_OUT. Read as 0. 0_B No operation 1_B Sets or toggles Pn_OUT.Px.
PCLx (x=0-1)	x+16	w	**Clear Bit x** Setting this bit will clear or toggle the corresponding bit in the port output register Pn_OUT. Read as 0. 0_B No operation 1_B Clears or toggles Pn_OUT.Px.
0	15, 14, 13, 12, 11, 10, 9, 8, 7, 6, 5, 4, 3, 2, 31, 30, 29, 28, 27, 26, 25, 24, 23, 22, 21, 20, 19, 18	r	**Reserved** Read as 0; should be written with 0.

47.8.4.88 Port port_no Output Modification Register (port_no=34-35)

P34_OMR Offset address: $03C_H$

Port 34 Output Modification Register Application Reset value: $0000\ 0000_H$

P35_OMR Offset address: $03C_H$

Port 35 Output Modification Register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	PCL5	PCL4	PCL3	PCL2	PCL1	PCL0
r	r	r	r	r	r	r	r	r	r	w	w	w	w	w	w

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	PS5	PS4	PS3	PS2	PS1	PS0
r	r	r	r	r	r	r	r	r	r	w	w	w	w	w	w

Field	Bits	Type	Description
PSx (x=0-5)	x	w	**Set Bit x** Setting this bit will set or toggle the corresponding bit in the port output register Pn_OUT. Read as 0. 0_B No operation 1_B Sets or toggles Pn_OUT.Px.
PCLx (x=0-5)	x+16	w	**Clear Bit x** Setting this bit will clear or toggle the corresponding bit in the port output register Pn_OUT. Read as 0. 0_B No operation 1_B Clears or toggles Pn_OUT.Px.
0	15, 14, 13, 12, 11, 10, 9, 8, 7, 6, 31, 30, 29, 28, 27, 26, 25, 24, 23, 22	r	**Reserved** Read as 0; should be written with 0.

47.8.4.89 Port 04 Output Modification Clear Register

The port output modification clear register contains control bits that make it possible to individually clear the logic state of a single port line by manipulating the output register.

***Note**: Register Pn_OMCR is virtual and does not contain any flip-flop. A read action delivers the value of 0. One 8 or 16-bits write behaves as a 32-bit write padded with zeros.*

P04_OMCR Offset address: 040_H

Port 04 Output Modification Clear Register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	PCL9	PCL8	PCL7	PCL6	PCL5	PCL4	PCL3	PCL2	PCL1	PCL0
r	r	r	r	r	r	w	w	w	w	w	w	w	w	w	w

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															
r															

Field	Bits	Type	Description
PCLx (x=0-9)	x+16	w	**Clear Bit x** Setting this bit will clear the corresponding bit in the port output register Pn_OUT. Read as 0. 0_B No operation 1_B Clears Pn_OUT.Px.
0	15:0, 31, 30, 29, 28, 27, 26	r	**Reserved** Read as 0; should be written with 0

47.8.4.90 Port port_no Output Modification Clear Register (port_no=16,20)

P16_OMCR Offset address: 040_H

Port 16 Output Modification Clear Register Application Reset value: $0000\ 0000_H$

P20_OMCR Offset address: 040_H

Port 20 Output Modification Clear Register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	PCL14	PCL13	PCL12	PCL11	PCL10	PCL9	PCL8	PCL7	PCL6	PCL5	PCL4	PCL3	PCL2	PCL1	PCL0
r	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															
r															

Field	Bits	Type	Description
PCLx (x=0-14)	x+16	w	**Clear Bit x** Setting this bit will clear the corresponding bit in the port output register Pn_OUT. Read as 0. 0_B No operation 1_B Clears Pn_OUT.Px.
0	15:0, 31	r	**Reserved** Read as 0; should be written with 0

47.8.4.91 Port port_no Output Modification Clear Register (port_no=21,23,31-32)

P21_OMCR Offset address: 040_H

Port 21 Output Modification Clear Register Application Reset value: $0000\ 0000_H$

P23_OMCR Offset address: 040_H

Port 23 Output Modification Clear Register Application Reset value: $0000\ 0000_H$

P31_OMCR Offset address: 040_H

Port 31 Output Modification Clear Register Application Reset value: $0000\ 0000_H$

P32_OMCR Offset address: 040_H

Port 32 Output Modification Clear Register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	PCL7	PCL6	PCL5	PCL4	PCL3	PCL2	PCL1	PCL0
r	r	r	r	r	r	r	r	w	w	w	w	w	w	w	w

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															
r															

Field	Bits	Type	Description
PCLx (x=0-7)	x+16	w	**Clear Bit x** Setting this bit will clear the corresponding bit in the port output register Pn_OUT. Read as 0. 0_B No operation 1_B Clears Pn_OUT.Px.
0	15:0, 31, 30, 29, 28, 27, 26, 25, 24	r	**Reserved** Read as 0; should be written with 0

47.8.4.92 Port 22 Output Modification Clear Register

P22_OMCR Offset address: 040_H

Port 22 Output Modification Clear Register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	PCL11	PCL10	PCL9	PCL8	PCL7	PCL6	PCL5	PCL4	PCL3	PCL2	PCL1	PCL0
r	r	r	r	w	w	w	w	w	w	w	w	w	w	w	w

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															
r															

Field	Bits	Type	Description
PCLx (x=0-11)	x+16	w	**Clear Bit x** Setting this bit will clear the corresponding bit in the port output register Pn_OUT. Read as 0. 0_B No operation 1_B Clears Pn_OUT.Px.
0	15:0, 31, 30, 29, 28	r	**Reserved** Read as 0; should be written with 0

47.8.4.93 Port 25 Output Modification Clear Register

P25_OMCR Offset address: 040_H

Port 25 Output Modification Clear Register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	PCL1	PCL0
r	r	r	r	r	r	r	r	r	r	r	r	r	r	w	w

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															
r															

Field	Bits	Type	Description
PCLx (x=0-1)	x+16	w	**Clear Bit x** Setting this bit will clear the corresponding bit in the port output register Pn_OUT. Read as 0. 0_B No operation 1_B Clears Pn_OUT.Px.
0	15:0, 31, 30, 29, 28, 27, 26, 25, 24, 23, 22, 21, 20, 19, 18	r	**Reserved** Read as 0; should be written with 0

47.8.4.94 Port port_no Output Modification Clear Register (port_no=34-35)

P34_OMCR Offset address: 040_H

Port 34 Output Modification Clear Register Application Reset value: $0000\ 0000_H$

P35_OMCR Offset address: 040_H

Port 35 Output Modification Clear Register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	PCL5	PCL4	PCL3	PCL2	PCL1	PCL0
r	r	r	r	r	r	r	r	r	r	w	w	w	w	w	w

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															
r															

Field	Bits	Type	Description
PCLx (x=0-5)	x+16	w	**Clear Bit x** Setting this bit will clear the corresponding bit in the port output register Pn_OUT. Read as 0. 0_B No operation 1_B Clears Pn_OUT.Px.

(table continues...)

(continued)

Field	Bits	Type	Description
0	15:0, 31, 30, 29, 28, 27, 26, 25, 24, 23, 22	r	**Reserved** Read as 0; should be written with 0

47.8.4.95 Port 04 Output Modification Set Register

The port output modification set register contains control bits that make it possible to individually set the logic state of a single port line by manipulating the output register.

***Note**: Register Pn_OMSR is virtual and does not contain any flip-flop. A read action delivers the value of 0. One 8 or 16-bits write behaves as a 32-bit write padded with zeros.*

P04_OMSR Offset address: 044_H

Port 04 Output Modification Set Register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	PS9	PS8	PS7	PS6	PS5	PS4	PS3	PS2	PS1	PS0
r	r	r	r	r	r	w	w	w	w	w	w	w	w	w	w

Field	Bits	Type	Description
PSx (x=0-9)	x	w	**Set Bit x** Setting this bit will set the corresponding bit in the port output register Pn_OUT. Read as 0. 0_B No operation 1_B Sets Pn_OUT.Px
0	15, 14, 13, 12, 11, 10, 31:16	r	**Reserved** Read as 0; should be written with 0.

47.8.4.96 Port port_no Output Modification Set Register (port_no=16,20)

P16_OMSR Offset address: 044_H

Port 16 Output Modification Set Register Application Reset value: $0000\ 0000_H$

P20_OMSR Offset address: 044_H

Port 20 Output Modification Set Register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	PS14	PS13	PS12	PS11	PS10	PS9	PS8	PS7	PS6	PS5	PS4	PS3	PS2	PS1	PS0
r	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

Field	Bits	Type	Description
PSx (x=0-14)	x	w	**Set Bit x** Setting this bit will set the corresponding bit in the port output register Pn_OUT. Read as 0. 0_B No operation 1_B Sets Pn_OUT.Px
0	15, 31:16	r	**Reserved** Read as 0; should be written with 0.

47.8.4.97 Port port_no Output Modification Set Register (port_no=21,23,31-32)

P21_OMSR Offset address: 044_H

Port 21 Output Modification Set Register Application Reset value: $0000\ 0000_H$

P23_OMSR Offset address: 044_H

Port 23 Output Modification Set Register Application Reset value: $0000\ 0000_H$

P31_OMSR Offset address: 044_H

Port 31 Output Modification Set Register Application Reset value: $0000\ 0000_H$

P32_OMSR Offset address: 044_H

Port 32 Output Modification Set Register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	PS7	PS6	PS5	PS4	PS3	PS2	PS1	PS0
r	r	r	r	r	r	r	r	w	w	w	w	w	w	w	w

Field	Bits	Type	Description
PSx (x=0-7)	x	w	**Set Bit x** Setting this bit will set the corresponding bit in the port output register Pn_OUT. Read as 0. 0_B No operation 1_B Sets Pn_OUT.Px

(table continues...)

(continued)

Field	Bits	Type	Description
0	15, 14, 13, 12, 11, 10, 9, 8, 31:16	r	**Reserved** Read as 0; should be written with 0.

47.8.4.98 Port 22 Output Modification Set Register

P22_OMSR Offset address: 044_H

Port 22 Output Modification Set Register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	PS11	PS10	PS9	PS8	PS7	PS6	PS5	PS4	PS3	PS2	PS1	PS0
r	r	r	r	w	w	w	w	w	w	w	w	w	w	w	w

Field	Bits	Type	Description
PSx (x=0-11)	x	w	**Set Bit x** Setting this bit will set the corresponding bit in the port output register Pn_OUT. Read as 0. 0_B No operation 1_B Sets Pn_OUT.Px
0	15, 14, 13, 12, 31:16	r	**Reserved** Read as 0; should be written with 0.

47.8.4.99 Port 25 Output Modification Set Register

P25_OMSR Offset address: 044_H

Port 25 Output Modification Set Register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	PS1	PS0
r	r	r	r	r	r	r	r	r	r	r	r	r	r	w	w

Field	Bits	Type	Description
PSx (x=0-1)	x	w	**Set Bit x** Setting this bit will set the corresponding bit in the port output register Pn_OUT. Read as 0. 0_B No operation 1_B Sets Pn_OUT.Px
0	15, 14, 13, 12, 11, 10, 9, 8, 7, 6, 5, 4, 3, 2, 31:16	r	**Reserved** Read as 0; should be written with 0.

47.8.4.100 Port port_no Output Modification Set Register (port_no=34-35)

P34_OMSR Offset address: 044_H

Port 34 Output Modification Set Register Application Reset value: $0000\ 0000_H$

P35_OMSR Offset address: 044_H

Port 35 Output Modification Set Register Application Reset value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

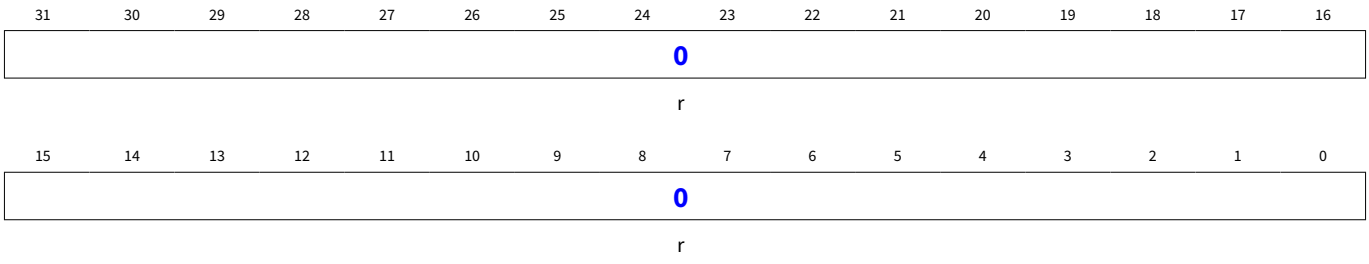
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	PS5	PS4	PS3	PS2	PS1	PS0
r	r	r	r	r	r	r	r	r	r	w	w	w	w	w	w

Field	Bits	Type	Description
PSx (x=0-5)	x	w	**Set Bit x** Setting this bit will set the corresponding bit in the port output register Pn_OUT. Read as 0. 0_B No operation 1_B Sets Pn_OUT.Px
0	15, 14, 13, 12, 11, 10, 9, 8, 7, 6, 31:16	r	**Reserved** Read as 0; should be written with 0.

47.8.4.101 Port port_no LVDS Pad Status Register x (port_no=13;x=0-3) (port_no=15;x=5-6) (port_no=21;x=2) (port_no=22;x=0-1)

The LVDS Pad Status Register reflects the status of the RX LVDS pads.

P13_LPSRx (x=0-3)	Offset address:	048_H+x*8
Port 13 LVDS Pad Status Register x	Application Reset value:	0000 0000_H
P22_LPSRx (x=0-1)	Offset address:	048_H+x*8
Port 22 LVDS Pad Status Register x	Application Reset value:	0000 0000_H
P21_LPSRx (x=2)	Offset address:	048_H+x*8
Port 21 LVDS Pad Status Register x	Application Reset value:	0000 0000_H
P15_LPSRx (x=5-6)	Offset address:	048_H+x*8
Port 15 LVDS Pad Status Register x	Application Reset value:	0000 0000_H



Field	Bits	Type	Description
0	31:0	r	**Reserved** Read as 0.

47.8.4.102 Port port_no LVDS Pad Control Register x (port_no=13;x=0-3) (port_no=15;x=5-6) (port_no=22;x=0-1) (port_no=21;x=2)

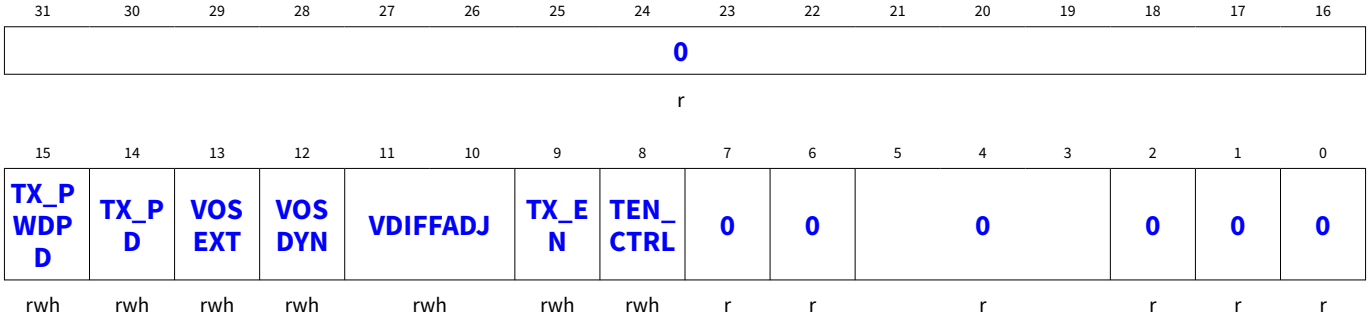
The LVDS Pad Control Register controls the RX or TX functions of the LVDS pads. For usage of RX pad, bit field [7:0] is applicable. If the register is used for TX pad, bit field [15:7] applies.

The sleep functionality of the LVDS pads is not controllable by this register. This is exclusively controlled by the HSCT module when this is connected.

The register x controls in general the pad pair 2*x and 2*x+1 of the port n.

Exceptionally when available the pad pair P14.9 and P14.10 is controlled by P14_LPCR5.

P13_LPCRx (x=0-3)	Offset address:	$04C_H$+x*8
Port 13 LVDS Pad Control Register x	Reset values see:	Table 2501
P22_LPCRx (x=0-1)	Offset address:	$04C_H$+x*8
Port 22 LVDS Pad Control Register x	Reset values see:	Table 2501
P21_LPCRx (x=2)	Offset address:	$04C_H$+x*8
Port 21 LVDS Pad Control Register x	Reset values see:	Table 2502
P15_LPCRx (x=5-6)	Offset address:	$04C_H$+x*8
Port 15 LVDS Pad Control Register x	Reset values see:	Table 2501



Field	Bits	Type	Description
TEN_CTRL	8	rwh	**LVDS TX_EN controller** The LVDS TX_EN control function can be selected from the Port (default) or HSCT module (where this is connected). 0_B Port controlled 1_B Reserved
TX_EN	9	rwh	**Enable Transmit LVDS** Enable the transmit LVDS / disable CMOS path. If this bit is set to 0 - no transfer on LVDS data path can be initiated and the LVDS driver is disabled (powered down). 0_B disable LVDS / enable CMOS mode 1_B enable LVDS / disable CMOS mode
VDIFFADJ	11:10	rwh	**LVDS Output Amplitude Tuning** With these two configuration bits the LVDS output current/amplitude can be adjusted. The voltage swing depending on VDIFFADJ setting is documented in the Data Sheet, see parameter V_{OD} .
VOSDYN	12	rwh	**Tune Bit of VOS Control Loop Static/Dynamic** Tune bit to change V_{OS} control loop between static and dynamic mode. Don’t change reset value.
VOSEXT	13	rwh	**Tune Bit of VOS Control Loop Internal/External** Tune bit to change V_{OS} control loop. Don’t change reset value.
TX_PD	14	rwh	**LVDS Power Down** Unused in this device. LVDS disabled by TX_EN means power down. 0_B LVDS power on 1_B LVDS power down (default)
TX_PWDPD	15	rwh	**Enable TX Power down pull down.** This function disables or enables the LVDS pull down resistor. The application code must disable TX power down pull down resistor with a power up. With a LVDS Power Down configuration the pull down function must be enabled, if required. 0_B disabled TX Power down pull down resistor. 1_B enabled TX Power down pull down resistor.

(table continues...)

(continued)

Field	Bits	Type	Description
0	0, 1, 2, 5:3, 6, 7, 31:16	r	**Reserved** Read as 0; should be written with 0

Table 2501 Reset values variant 1

Applies to P13_LPCRx (x=0-3)
Applies to P15_LPCRx (x=5-6)
Applies to P22_LPCRx (x=0-1)

Reset type	Reset value	Note
PowerOn Reset	0054 8000$_H$	
After Boot-FW Value	0054 8000$_H$	Initial value of RX depends on trimming

Table 2502 Reset values of P21_LPCRx (x=2)

Reset type	Reset value	Note
PowerOn Reset	0054 8000$_H$	
After Boot-FW Value	0054 8000$_H$	Initial value of RX depends on trimming

47.8.4.103 Port port_no LVDS Pad Control Register x (port_no=14;x=5) (port_no=21;x=0-1)

P21_LPCRx (x=0-1) Offset address: 04C$_H$+x*8

Port 21 LVDS Pad Control Register x Reset values see: Table 2504

P14_LPCRx (x=5) Offset address: 04C$_H$+x*8

Port 14 LVDS Pad Control Register x Reset values see: Table 2503

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0		0	0	0	0	0			0	RX_EN	REN_CTRL
r	r	r	r	r		r	r	r	r	r			r	rwh	rwh

Field	Bits	Type	Description
REN_CTRL	0	rwh	**LVDS RX_EN controller** The LVDS RX_EN control function can be selected from the Port (default) or HSCT module (where this is connected). 0_B Port controlled 1_B Reserved
RX_EN	1	rwh	**Enable Receive LVDS** Enable the receive LVDS / disable CMOS path. If this bit is set to 0 – no transfer from the LVDS sender can be received and the receiver LVDS is in low power state. 0_B disable LVDS / enable CMOS mode (reserved for pads without CMOS input stage) 1_B enable LVDS / disable CMOS mode
0	2, 5:3, 6, 7, 8, 9, 11:10, 12, 13, 14, 15, 31:16	r	**Reserved** Read as 0; should be written with 0

Table 2503 Reset values of P14_LPCRx (x=5)

Reset type	Reset value	Note
PowerOn Reset	0000 00E0$_H$	
After Boot-FW Value	0000 00E0$_H$	Initial value of RX depends on trimming

Table 2504 Reset values of P21_LPCRx (x=0-1)

Reset type	Reset value	Note
PowerOn Reset	0000 00E0$_H$	
After Boot-FW Value	0000 00E0$_H$	Initial value of RX depends on trimming

47.8.4.104 Port port_no Pad configuration Control Register for padp (port_no=00-03,10,13-15;p=0-15) (port_no=04;p=0-9) (port_no=20;p=0-14)

Driver Configuration

The Pin Driver Configuration register contains the bit fields needed to set up the required pin configuration such as signal source and driver type selection.

P00_PADCFGp_DRVCFG (p=0-15) Offset address: 304_H+p*10_H

Port 00 Pad configuration Control Register for padp Reset values see: Table 2505

P01_PADCFGp_DRVCFG (p=0-15) Offset address: 304_H+p*10_H

Port 01 Pad configuration Control Register for padp Reset values see: Table 2505

P02_PADCFGp_DRVCFG (p=0-15) Offset address: 304_H+p*10_H

Port 02 Pad configuration Control Register for padp Reset values see: Table 2505

P03_PADCFGp_DRVCFG (p=0-15) Offset address: 304_H+p*10_H

Port 03 Pad configuration Control Register for padp Reset values see: Table 2505

P04_PADCFGp_DRVCFG (p=0-9) Offset address: 304_H+p*10_H

Port 04 Pad configuration Control Register for padp Reset values see: Table 2505

P10_PADCFGp_DRVCFG (p=0-15) Offset address: 304_H+p*10_H

Port 10 Pad configuration Control Register for padp Reset values see: Table 2505

P13_PADCFGp_DRVCFG (p=0-15) Offset address: 304_H+p*10_H

Port 13 Pad configuration Control Register for padp Reset values see: Table 2505

P14_PADCFGp_DRVCFG (p=0-15) Offset address: 304_H+p*10_H

Port 14 Pad configuration Control Register for padp Reset values see: Table 2505

P15_PADCFGp_DRVCFG (p=0-15) Offset address: 304_H+p*10_H

Port 15 Pad configuration Control Register for padp Reset values see: Table 2505

P20_PADCFGp_DRVCFG (p=0-14) Offset address: 304_H+p*10_H

Port 20 Pad configuration Control Register for padp Reset values see: Table 2505

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0														0	
r														r	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	PL			0	PD			MODE				0		OD	DIR
r	rwh			r	rwh			rwh				r		rwh	rwh

Field	Bits	Type	Description
DIR	0	rwh	**Pad Signal direction** The DIR field defines the direction of the Port Pad in GPIO mode 0_B **IN**: Pad configured as Input 1_B **OUT**: Pad configured as Output

(table continues...)

(continued)

Field	Bits	Type	Description
OD	1	rwh	**Output Buffer configuration** The bitfield OD defines the Output driver configuration when the Pin is configured in output mode. The Pad may be configured to Open Drain or Push/Pull driver types. 0_B **PP**: Push Pull 1_B **OD**: Open Drain
MODE	7:4	rwh	**Input configuration or Alternative Output signal selection** If the Pad is configured as Input, the MODE field determinates the Input mode regarding pure tristate mode versus a connected pull device. Note: If the direction is set to input only the lower 2 bits of the Mode field will be evaluated. If the Pad is configured as Output, the MODE field selects the number of the alternative output function driving the value towards the pin number n. Values higher than 0 do select the alternative output function matching to the absolute number of the value (e.g. MODE=1 selects alternative function number 1) When set to 0 the Pin acts as GPIO port. There the Pin state is SW controllable via the register Pn_GPIO 0_H **GPIO_TRI**: Pad configured in Tristate input mode or GPIO output mode 1_H **ALT01_PD**: Pad connected to alternative function 1 or input pull-down device 2_H **ALT02_PU**: Pad connected to alternative function 2 or input pull-up device 3_H **ALT03_TRI**: Pad connected to alternative function 3 or input tristate mode 4_H **4**: Alternative Output function 4 selected … F_H **15**: Alternative Output function 15 selected

(table continues...)

(continued)

Field	Bits	Type	Description
PD	10:8	rwh	**Pad drive mode selection** **Driver strength selection for Fast and Slow GPIO** **Driver strength selection bitfield (PD) decoding for Fast GPIO** [PD2, PD1, PD0] = [X, 0, 0]: Driver = strong, Edge = sharp (ss) [PD2, PD1, PD0] = [X, 0, 1]: Driver = strong, Edge = medium (sm) [PD2, PD1, PD0] = [X, 1, 0]: Driver = medium, Edge = medium (m) [PD2, PD1, PD0] = [X, 1, 1]: do not use **Driver strength selection bitfield (PD) decoding for Slow GPIO** [PD2, PD1, PD0] = [X, X, 0]: Driver = medium, Edge = sharp (sm) [PD2, PD1, PD0] = [X, X, 1]: Driver = medium, Edge = medium (m) **Driver strength selection bitfield (PD) decoding for HSFAST GPIO** In case the register PCSRSEL of the corresponding port slice is set to zero the decoding of the FAST GPIO applies In case the register PCSRSEL of the corresponding port slice is set to one the decoding of the configurable output impedances in the datasheet (section: HSFast x.x V GPIO characteristics) applies 000_B **PD_SS**: [PD2, PD1, PD0] = [0, 0, 0] Coding for the respective pad type see table above 001_B **PD_SM**: [PD2, PD1, PD0] = [0, 0, 1] Coding for the respective pad type see table above 010_B **PD_M**: [PD2, PD1, PD0] = [0, 1, 0] Coding for the respective pad type see table above 011_B **PD_DNU**: [PD2, PD1, PD0] = [0, 1, 1] do not use this combination 100_B **PD_SS**: [PD2, PD1, PD0] = [1, 0, 0] Coding for the respective pad type see table above 101_B **PD_SM**: [PD2, PD1, PD0] = [1, 0, 1] Coding for the respective pad type see table above 110_B **PD_M**: [PD2, PD1, PD0] = [1, 1, 0] Coding for the respective pad type see table above 111_B **PD_DNU**: [PD2, PD1, PD0] = [1, 1, 1] do not use this combination

(table continues...)

(continued)

Field	Bits	Type	Description
PL	14:12	rwh	**Pad level selection** **PLx Input Level control** [PL2, PL1, PL0] = [X, 0, X]: Automotive level "AL" [PL2, PL1, PL0] = [X, 1, 0]: TTL level for 5V pad supply. [PL2, PL1, PL0] = [X, 1, 1]: TTL level for 3.3V pad supply. [PL2, PL1, PL0] = [X, X, X]: do not use ***Note:*** *In case of P16.xx the PL0 bit is tight to zero. When PL1 is set to one TTL level for 3.3V pad supply is selected.* 000_B **PL_AL**: Automotive level "AL" Automotive level "AL" 001_B **PL_AL**: Automotive level "AL" Automotive level "AL" 010_B **PL_TTL5V**: TTL level for 5V pad supply TTL level for 5V pad supply 011_B **PL_TTL3_3V**: TTL level for 3.3V pad supply TTL level for 3.3V pad supply 100_B **PL_AL**: Automotive level "AL" Automotive level "AL" 101_B **PL_AL**: Automotive level "AL" Automotive level "AL" 110_B **PL_TTL5V**: TTL level for 5V pad supply TTL level for 5V pad supply 111_B **PL_TTL3_3V**: TTL level for 3.3V pad supply TTL level for 3.3V pad supply
0	3:2, 11, 15, 17:16, 31:18	r	**Reserved** Read as 0; should be written with 0

Table 2505 Reset values

Applies to P00_PADCFGp_DRVCFG (p=0-15)
Applies to P01_PADCFGp_DRVCFG (p=0-15)
Applies to P02_PADCFGp_DRVCFG (p=0-15)
Applies to P03_PADCFGp_DRVCFG (p=0-15)
Applies to P04_PADCFGp_DRVCFG (p=0-9)
Applies to P10_PADCFGp_DRVCFG (p=0-15)
Applies to P13_PADCFGp_DRVCFG (p=0-15)
Applies to P14_PADCFGp_DRVCFG (p=0-15)
Applies to P15_PADCFGp_DRVCFG (p=0-15)
Applies to P20_PADCFGp_DRVCFG (p=0-14)

Reset type	Reset value	Note
Application Reset	0000 0000 0000 0000 0000 0000 00X0 0000_B	Depending on HWCFG[6]: X:0 (tri-state mode) X:1 (input pull-up mode)
Kernel Reset	0000 0000 0000 0000 0000 0000 00X0 0000_B	Depending on HWCFG[6]: X:0 (tri-state mode) X:1 (input pull-up mode)

47.8.4.105 Port port_no Pad configuration Control Register for padp (port_no=21,23,31;p=0-7) (port_no=22;p=0-11) (port_no=25;p=0-1) (port_no=30;p=0-15) (port_no=32;p=0-1,3-4,6-7) (port_no=33;p=0-7,9-11,13-15) (port_no=34-35;p=0-5)

P21_PADCFGp_DRVCFG (p=0-7) Offset address: 304_H+p*10_H
Port 21 Pad configuration Control Register for padp Reset values see: Table 2506

P22_PADCFGp_DRVCFG (p=0-11) Offset address: 304_H+p*10_H
Port 22 Pad configuration Control Register for padp Reset values see: Table 2506

P23_PADCFGp_DRVCFG (p=0-7) Offset address: 304_H+p*10_H
Port 23 Pad configuration Control Register for padp Reset values see: Table 2506

P25_PADCFGp_DRVCFG (p=0-1) Offset address: 304_H+p*10_H
Port 25 Pad configuration Control Register for padp Reset values see: Table 2506

P30_PADCFGp_DRVCFG (p=0-15) Offset address: 304_H+p*10_H
Port 30 Pad configuration Control Register for padp Reset values see: Table 2506

P31_PADCFGp_DRVCFG (p=0-7) Offset address: 304_H+p*10_H
Port 31 Pad configuration Control Register for padp Reset values see: Table 2506

P32_PADCFGp_DRVCFG (p=0-1,3-4,6-7) Offset address: 304_H+p*10_H
Port 32 Pad configuration Control Register for padp Reset values see: Table 2506

P33_PADCFGp_DRVCFG (p=0-7,9-11,13-15) Offset address: 304_H+p*10_H
Port 33 Pad configuration Control Register for padp Reset values see: Table 2506

P34_PADCFGp_DRVCFG (p=0-5) Offset address: 304_H+p*10_H
Port 34 Pad configuration Control Register for padp Reset values see: Table 2506

P35_PADCFGp_DRVCFG (p=0-5) Offset address: 304_H+p*10_H

Port 35 Pad configuration Control Register for padp Reset values see: Table 2506

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0														0	
r														r	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	PL			0	PD			MODE				0		OD	DIR
r	rwh			r	rwh			rwh				r		rwh	rwh

Field	Bits	Type	Description
DIR	0	rwh	**Pad Signal direction** The DIR field defines the direction of the Port Pad in GPIO mode 0_B **IN**: Pad configured as Input 1_B **OUT**: Pad configured as Output
OD	1	rwh	**Output Buffer configuration** The bitfield OD defines the Output driver configuration when the Pin is configured in output mode. The Pad may be configured to Open Drain or Push/Pull driver types. 0_B **PP**: Push Pull 1_B **OD**: Open Drain
MODE	7:4	rwh	**Input configuration or Alternative Output signal selection** If the Pad is configured as Input, the MODE field determinates the Input mode regarding pure tristate mode versus a connected pull device. Note: If the direction is set to input only the lower 2 bits of the Mode field will be evaluated. If the Pad is configured as Output, the MODE field selects the number of the alternative output function driving the value towards the pin number n. Values higher than 0 do select the alternative output function matching to the absolute number of the value (e.g. MODE=1 selects alternative function number 1) When set to 0 the Pin acts as GPIO port. There the Pin state is SW controllable via the register Pn_GPIO 0_H **GPIO_TRI**: Pad configured in Tristate input mode or GPIO output mode 1_H **ALT01_PD**: Pad connected to alternative function 1 or input pull-down device 2_H **ALT02_PU**: Pad connected to alternative function 2 or input pull-up device 3_H **ALT03_TRI**: Pad connected to alternative function 3 or input tristate mode 4_H **4**: Alternative Output function 4 selected … F_H **15**: Alternative Output function 15 selected

(table continues...)

(continued)

Field	Bits	Type	Description
PD	10:8	rwh	**Pad drive mode selection** **Driver strength selection for Fast and Slow GPIO** **Driver strength selection bitfield (PD) decoding for Fast GPIO** [PD2, PD1, PD0] = [X, 0, 0]: Driver = strong, Edge = sharp (ss) [PD2, PD1, PD0] = [X, 0, 1]: Driver = strong, Edge = medium (sm) [PD2, PD1, PD0] = [X, 1, 0]: Driver = medium, Edge = medium (m) [PD2, PD1, PD0] = [X, 1, 1]: do not use **Driver strength selection bitfield (PD) decoding for Slow GPIO** [PD2, PD1, PD0] = [X, X, 0]: Driver = medium, Edge = sharp (sm) [PD2, PD1, PD0] = [X, X, 1]: Driver = medium, Edge = medium (m) **Driver strength selection bitfield (PD) decoding for HSFAST GPIO** In case the register PCSRSEL of the corresponding port slice is set to zero the decoding of the FAST GPIO applies In case the register PCSRSEL of the corresponding port slice is set to one the decoding of the configurable output impedances in the datasheet (section: HSFast x.x V GPIO characteristics) applies 000_B **PD_SS**: [PD2, PD1, PD0] = [0, 0, 0] Coding for the respective pad type see table above 001_B **PD_SM**: [PD2, PD1, PD0] = [0, 0, 1] Coding for the respective pad type see table above 010_B **PD_M**: [PD2, PD1, PD0] = [0, 1, 0] Coding for the respective pad type see table above 011_B **PD_DNU**: [PD2, PD1, PD0] = [0, 1, 1] do not use this combination 100_B **PD_SS**: [PD2, PD1, PD0] = [1, 0, 0] Coding for the respective pad type see table above 101_B **PD_SM**: [PD2, PD1, PD0] = [1, 0, 1] Coding for the respective pad type see table above 110_B **PD_M**: [PD2, PD1, PD0] = [1, 1, 0] Coding for the respective pad type see table above 111_B **PD_DNU**: [PD2, PD1, PD0] = [1, 1, 1] do not use this combination

(table continues...)

(continued)

Field	Bits	Type	Description
PL	14:12	rwh	**Pad level selection** **PLx Input Level control** [PL2, PL1, PL0] = [X, 0, X]: Automotive level "AL" [PL2, PL1, PL0] = [X, 1, 0]: TTL level for 5V pad supply. [PL2, PL1, PL0] = [X, 1, 1]: TTL level for 3.3V pad supply. [PL2, PL1, PL0] = [X, X, X]: do not use ***Note:*** *In case of P16.xx the PL0 bit is tight to zero. When PL1 is set to one TTL level for 3.3V pad supply is selected.* 000_B **PL_AL**: Automotive level "AL" Automotive level "AL" 001_B **PL_AL**: Automotive level "AL" Automotive level "AL" 010_B **PL_TTL5V**: TTL level for 5V pad supply TTL level for 5V pad supply 011_B **PL_TTL3_3V**: TTL level for 3.3V pad supply TTL level for 3.3V pad supply 100_B **PL_AL**: Automotive level "AL" Automotive level "AL" 101_B **PL_AL**: Automotive level "AL" Automotive level "AL" 110_B **PL_TTL5V**: TTL level for 5V pad supply TTL level for 5V pad supply 111_B **PL_TTL3_3V**: TTL level for 3.3V pad supply TTL level for 3.3V pad supply
0	3:2, 11, 15, 17:16, 31:18	r	**Reserved** Read as 0; should be written with 0

Table 2506 Reset values

Applies to P21_PADCFGp_DRVCFG (p=0-7)
Applies to P22_PADCFGp_DRVCFG (p=0-11)
Applies to P23_PADCFGp_DRVCFG (p=0-7)
Applies to P25_PADCFGp_DRVCFG (p=0-1)
Applies to P30_PADCFGp_DRVCFG (p=0-15)
Applies to P31_PADCFGp_DRVCFG (p=0-7)
Applies to P32_PADCFGp_DRVCFG (p=0-1,3-4,6-7)
Applies to P33_PADCFGp_DRVCFG (p=0-7,9-11,13-15)
Applies to P34_PADCFGp_DRVCFG (p=0-5)
Applies to P35_PADCFGp_DRVCFG (p=0-5)

Reset type	Reset value	Note
Application Reset	0000 0000 0000 0000 0000 0000 00X0 0000_B	Depending on HWCFG[6]: X:0 (tri-state mode) X:1 (input pull-up mode)
Kernel Reset	0000 0000 0000 0000 0000 0000 00X0 0000_B	Depending on HWCFG[6]: X:0 (tri-state mode) X:1 (input pull-up mode)

47.8.4.106 Port port_no Pad configuration Control Register for padp (port_no=32;p=2,5) (port_no=33;p=8,12)

P32_PADCFGp_DRVCFG (p=2,5) Offset address: 304_H+p*10_H

Port 32 Pad configuration Control Register for padp Reset values see: Table 2507

P33_PADCFGp_DRVCFG (p=8,12) Offset address: 304_H+p*10_H

Port 33 Pad configuration Control Register for padp Reset values see: Table 2507

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0														0	
r														r	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	PL			0	PD			MODE				0		OD	DIR
r	rwh			r	rwh			rwh				r		rwh	rwh

Field	Bits	Type	Description
DIR	0	rwh	**Pad Signal direction** The DIR field defines the direction of the Port Pad in GPIO mode 0_B **IN**: Pad configured as Input 1_B **OUT**: Pad configured as Output
OD	1	rwh	**Output Buffer configuration** The bitfield OD defines the Output driver configuration when the Pin is configured in output mode. The Pad may be configured to Open Drain or Push/Pull driver types. 0_B **PP**: Push Pull 1_B **OD**: Open Drain

(table continues...)

(continued)

Field	Bits	Type	Description
MODE	7:4	rwh	**Input configuration or Alternative Output signal selection** If the Pad is configured as Input, the MODE field determinates the Input mode regarding pure tristate mode versus a connected pull device. Note: If the direction is set to input only the lower 2 bits of the Mode field will be evaluated. If the Pad is configured as Output, the MODE field selects the number of the alternative output function driving the value towards the pin number n. Values higher than 0 do select the alternative output function matching to the absolute number of the value (e.g. MODE=1 selects alternative function number 1) When set to 0 the Pin acts as GPIO port. There the Pin state is SW controllable via the register Pn_GPIO 0_H **GPIO_TRI**: Pad configured in Tristate input mode or GPIO output mode 1_H **ALT01_PD**: Pad connected to alternative function 1 or input pull-down device 2_H **ALT02_PU**: Pad connected to alternative function 2 or input pull-up device 3_H **ALT03_TRI**: Pad connected to alternative function 3 or input tristate mode 4_H **4**: Alternative Output function 4 selected … F_H **15**: Alternative Output function 15 selected

(table continues...)

(continued)

Field	Bits	Type	Description
PD	10:8	rwh	**Pad drive mode selection** **Driver strength selection for Fast and Slow GPIO** **Driver strength selection bitfield (PD) decoding for Fast GPIO** [PD2, PD1, PD0] = [X, 0, 0]: Driver = strong, Edge = sharp (ss) [PD2, PD1, PD0] = [X, 0, 1]: Driver = strong, Edge = medium (sm) [PD2, PD1, PD0] = [X, 1, 0]: Driver = medium, Edge = medium (m) [PD2, PD1, PD0] = [X, 1, 1]: do not use **Driver strength selection bitfield (PD) decoding for Slow GPIO** [PD2, PD1, PD0] = [X, X, 0]: Driver = medium, Edge = sharp (sm) [PD2, PD1, PD0] = [X, X, 1]: Driver = medium, Edge = medium (m) **Driver strength selection bitfield (PD) decoding for HSFAST GPIO** In case the register PCSRSEL of the corresponding port slice is set to zero the decoding of the FAST GPIO applies In case the register PCSRSEL of the corresponding port slice is set to one the decoding of the configurable output impedances in the datasheet (section: HSFast x.x V GPIO characteristics) applies 000_B **PD_SS**: [PD2, PD1, PD0] = [0, 0, 0] Coding for the respective pad type see table above 001_B **PD_SM**: [PD2, PD1, PD0] = [0, 0, 1] Coding for the respective pad type see table above 010_B **PD_M**: [PD2, PD1, PD0] = [0, 1, 0] Coding for the respective pad type see table above 011_B **PD_DNU**: [PD2, PD1, PD0] = [0, 1, 1] do not use this combination 100_B **PD_SS**: [PD2, PD1, PD0] = [1, 0, 0] Coding for the respective pad type see table above 101_B **PD_SM**: [PD2, PD1, PD0] = [1, 0, 1] Coding for the respective pad type see table above 110_B **PD_M**: [PD2, PD1, PD0] = [1, 1, 0] Coding for the respective pad type see table above 111_B **PD_DNU**: [PD2, PD1, PD0] = [1, 1, 1] do not use this combination

(table continues...)

(continued)

Field	Bits	Type	Description
PL	14:12	rwh	**Pad level selection** **PLx Input Level control** [PL2, PL1, PL0] = [X, 0, X]: Automotive level "AL" [PL2, PL1, PL0] = [X, 1, 0]: TTL level for 5V pad supply. [PL2, PL1, PL0] = [X, 1, 1]: TTL level for 3.3V pad supply. [PL2, PL1, PL0] = [X, X, X]: do not use ***Note:*** *In case of P16.xx the PL0 bit is tight to zero. When PL1 is set to one TTL level for 3.3V pad supply is selected.* 000_B **PL_AL**: Automotive level "AL" Automotive level "AL" 001_B **PL_AL**: Automotive level "AL" Automotive level "AL" 010_B **PL_TTL5V**: TTL level for 5V pad supply TTL level for 5V pad supply 011_B **PL_TTL3_3V**: TTL level for 3.3V pad supply TTL level for 3.3V pad supply 100_B **PL_AL**: Automotive level "AL" Automotive level "AL" 101_B **PL_AL**: Automotive level "AL" Automotive level "AL" 110_B **PL_TTL5V**: TTL level for 5V pad supply TTL level for 5V pad supply 111_B **PL_TTL3_3V**: TTL level for 3.3V pad supply TTL level for 3.3V pad supply
0	3:2, 11, 15, 17:16, 31:18	r	**Reserved** Read as 0; should be written with 0

Table 2507 Reset values

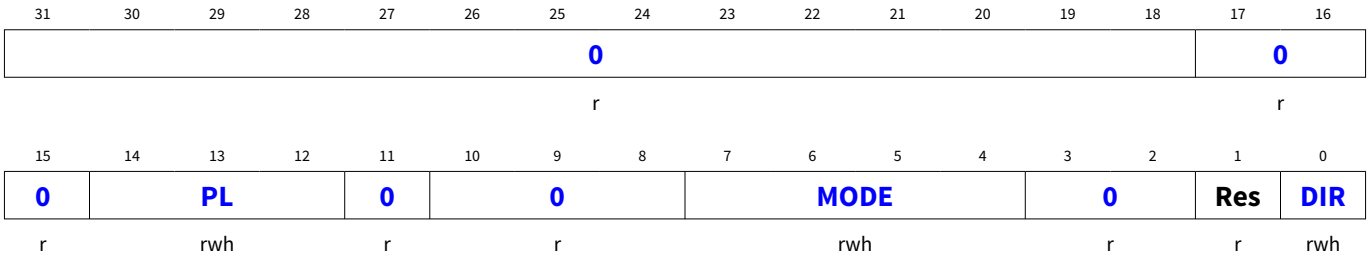
Applies to P32_PADCFGp_DRVCFG (p=2,5)
Applies to P33_PADCFGp_DRVCFG (p=8,12)

Reset type	Reset value	Note
Application Reset	$0000\ 0000_H$	SFF related reset value
Kernel Reset	$0000\ 0000_H$	SFF related reset value

47.8.4.107 Port 40 Pad configuration Control Register for padp

P40_PADCFGp_DRVCFG (p=0-15) Offset address: 304_H+p*10_H

Port 40 Pad configuration Control Register for padp Reset values see: Table 2508



Field	Bits	Type	Description
DIR	0	rwh	**Pad Signal direction** The DIR field defines the direction of the Port Pad in GPIO mode 0_B **IN**: Pad confligured as Input 1_B **OUT**: Pad confligured as Output
MODE	7:4	rwh	**Input configuration or Alternative Output signal selection** If the Pad is configured as Input, the MODE field determinates the Input mode regarding pure tristate mode versus a connected pull device. Note: If the direction is set to input only the lower 2 bits of the Mode field will be evaluated. If the Pad is configured as Output, the MODE field selects the number of the alternative output function driving the value towards the pin number n. Values higher than 0 do select the alternative output function matching to the absolute number of the value (e.g. MODE=1 selects alternative function number 1) When set to 0 the Pin acts as GPIO port. There the Pin state is SW controllable via the register Pn_GPIO 0_H **GPIO_TRI**: Pad configured in Tristate input mode or GPIO output mode 1_H **ALT01_PD**: Pad connected to alternative function 1 or input pull-down device 2_H **ALT02_PU**: Pad connected to alternative function 2 or input pull-up device 3_H **ALT03_TRI**: Pad connected to alternative function 3 or input tristate mode 4_H **4**: Alternative Output function 4 selected … F_H **15**: Alternative Output function 15 selected

(table continues...)

(continued)

Field	Bits	Type	Description
PL	14:12	rwh	**Pad level selection** **PLx Input Level control** [PL2, PL1, PL0] = [X, 0, X]: Automotive level "AL" [PL2, PL1, PL0] = [X, 1, 0]: TTL level for 5V pad supply. [PL2, PL1, PL0] = [X, 1, 1]: TTL level for 3.3V pad supply. [PL2, PL1, PL0] = [X, X, X]: do not use ***Note:*** *In case of P16.xx the PL0 bit is tight to zero. When PL1 is set to one TTL level for 3.3V pad supply is selected.* 000_B **PL_AL**: Automotive level "AL" Automotive level "AL" 001_B **PL_AL**: Automotive level "AL" Automotive level "AL" 010_B **PL_TTL5V**: TTL level for 5V pad supply TTL level for 5V pad supply 011_B **PL_TTL3_3V**: TTL level for 3.3V pad supply TTL level for 3.3V pad supply 100_B **PL_AL**: Automotive level "AL" Automotive level "AL" 101_B **PL_AL**: Automotive level "AL" Automotive level "AL" 110_B **PL_TTL5V**: TTL level for 5V pad supply TTL level for 5V pad supply 111_B **PL_TTL3_3V**: TTL level for 3.3V pad supply TTL level for 3.3V pad supply
0	3:2, 10:8, 11, 15, 17:16, 31:18	r	**Reserved** Read as 0; should be written with 0

Table 2508 Reset values of P40_PADCFGp_DRVCFG (p=0-15)

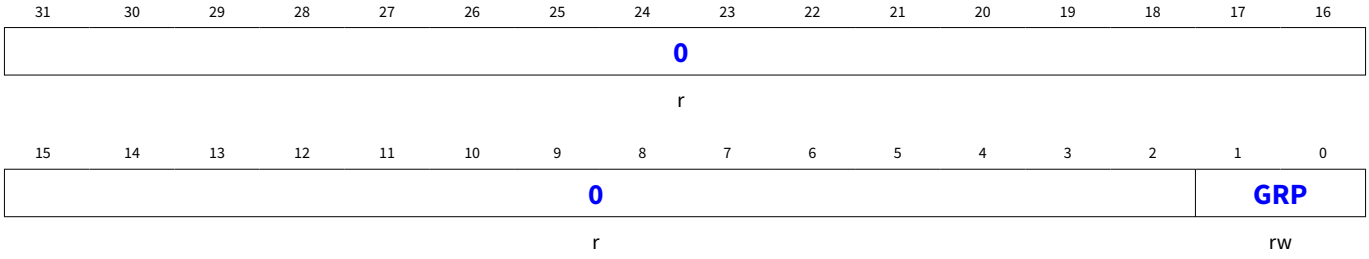
Reset type	Reset value	Note
Application Reset	0000 0000 0000 0000 0000 0000 00X0 0000_B	Depending on HWCFG[6]: X:0 (tri-state mode) X:1 (input pull-up mode)
Kernel Reset	0000 0000 0000 0000 0000 0000 00X0 0000_B	Depending on HWCFG[6]: X:0 (tri-state mode) X:1 (input pull-up mode)

47.8.4.108 Port port_no Pad Access Control Register for padp (port_no=21,23,31-32;p=0-7) (port_no=25;p=0-1) (port_no=34-35;p=0-5)

Access group selection and enable

The Pin Configuration register ACCEN contains the bit fields needed to set up the required access group settings.

P21_PADCFGp_ACCEN (p=0-7)	Offset address:	308_H+p*10_H
Port 21 Pad Access Control Register for padp	Application Reset value:	$0000\ 0000_H$
P23_PADCFGp_ACCEN (p=0-7)	Offset address:	308_H+p*10_H
Port 23 Pad Access Control Register for padp	Application Reset value:	$0000\ 0000_H$
P25_PADCFGp_ACCEN (p=0-1)	Offset address:	308_H+p*10_H
Port 25 Pad Access Control Register for padp	Application Reset value:	$0000\ 0000_H$
P31_PADCFGp_ACCEN (p=0-7)	Offset address:	308_H+p*10_H
Port 31 Pad Access Control Register for padp	Application Reset value:	$0000\ 0000_H$
P32_PADCFGp_ACCEN (p=0-7)	Offset address:	308_H+p*10_H
Port 32 Pad Access Control Register for padp	Application Reset value:	$0000\ 0000_H$
P34_PADCFGp_ACCEN (p=0-5)	Offset address:	308_H+p*10_H
Port 34 Pad Access Control Register for padp	Application Reset value:	$0000\ 0000_H$
P35_PADCFGp_ACCEN (p=0-5)	Offset address:	308_H+p*10_H
Port 35 Pad Access Control Register for padp	Application Reset value:	$0000\ 0000_H$



Field	Bits	Type	Description
GRP	1:0	rw	**Access Protection Group selection** Protection group selection The selected group in this field corresponds to the Access group block used to protect the accesses to the registers PADCFGp_GPIO and PADCFGp_DRVCFG. Note: If this bitfield is set to a non existing group the logic will rollover to an existing group, e.g. if only four groups are specified but a value of $(100)_2$.will be written it will result in a selection of group 0 with setting $(000)_2$. 00_B **GRP0__GRP**: Port Access Group selected … 11_B **GRP3__GRP**: Port Access Group selected
0	31:2	r	**Reserved** Read as 0; should be written with 0

47.8.4.109 Port port_no Pad Safety and Security Control Register for padp (port_no=00-03,10,13-15;p=0-15) (port_no=04;p=0-9) (port_no=16;p=0-14)

Safety and Security Pin settings

The Pin Configuration register SAFSEC contains the safety and security relevant bits.

P00_PADCFGp_SAFSEC (p=0-15) Offset address: $30C_H+p*10_H$

Port 00 Pad Safety and Security Control Register for padp Reset values see: Table 2509

P01_PADCFGp_SAFSEC (p=0-15) Offset address: $30C_H+p*10_H$

Port 01 Pad Safety and Security Control Register for padp Reset values see: Table 2509

P02_PADCFGp_SAFSEC (p=0-15) Offset address: $30C_H+p*10_H$

Port 02 Pad Safety and Security Control Register for padp Reset values see: Table 2509

P03_PADCFGp_SAFSEC (p=0-15) Offset address: $30C_H+p*10_H$

Port 03 Pad Safety and Security Control Register for padp Reset values see: Table 2509

P04_PADCFGp_SAFSEC (p=0-9) Offset address: $30C_H+p*10_H$

Port 04 Pad Safety and Security Control Register for padp Reset values see: Table 2509

P10_PADCFGp_SAFSEC (p=0-15) Offset address: $30C_H+p*10_H$

Port 10 Pad Safety and Security Control Register for padp Reset values see: Table 2509

P13_PADCFGp_SAFSEC (p=0-15) Offset address: $30C_H+p*10_H$

Port 13 Pad Safety and Security Control Register for padp Reset values see: Table 2509

P14_PADCFGp_SAFSEC (p=0-15) Offset address: $30C_H+p*10_H$

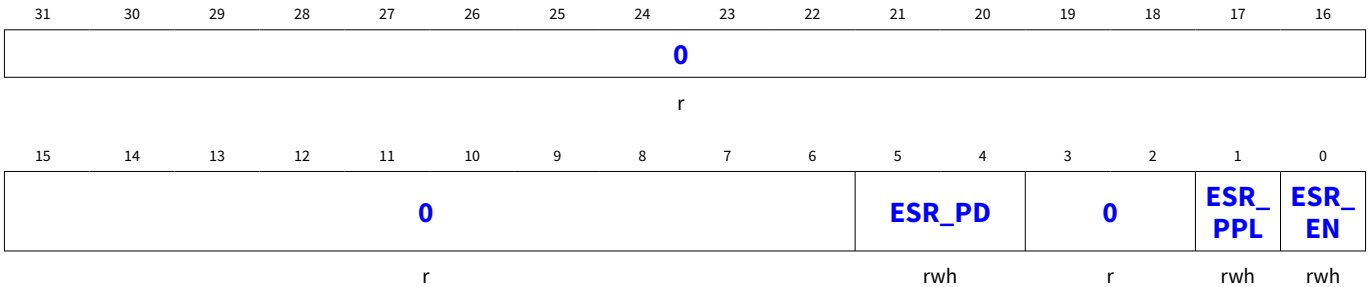
Port 14 Pad Safety and Security Control Register for padp Reset values see: Table 2509

P15_PADCFGp_SAFSEC (p=0-15) Offset address: $30C_H+p*10_H$

Port 15 Pad Safety and Security Control Register for padp Reset values see: Table 2509

P16_PADCFGp_SAFSEC (p=0-14) Offset address: $30C_H+p*10_H$

Port 16 Pad Safety and Security Control Register for padp Reset values see: Table 2509



Field	Bits	Type	Description
ESR_EN	0	rwh	**Emergency Request Action Enable** Most GPIO lines have an emergency stop logic implemented (see Figure “PORTS block diagram” in the Family chapter). Each of these GPIO lines has its own emergency stop enable bit ENx that is located in the emergency stop field ESR of Port Pin n. If the emergency stop signal becomes active, one of two states can be selected. Exceptions for Emergency Stop Implementation The Emergency Stop function is available for all GPIO Ports with the following exceptions: • Not available for P20.2 (General Purpose Input/GPI only, overlayed with Testmode) • Not available for P40_x and P41_x (analog input ANx overlayed with GPI) • Not available for P21.2 (used as SMU_EXT_EMERGENCYSTOP_REQ pin). • Not available for dedicated I/O without General Purpose Output function (e.g ESRx, TMS, TCK) The Emergency Stop function can be overruled on the following GPIO Ports: • P00.x: Emergency Stop can be overruled by the TMADC. Overruling can be disabled via the control register P00_PCSR. • P14.0 and P14.1: Emergency Stop can be overruled in the DXCPL (DAP over CAN physical layer) mode. No Overruling in the DXCM (Debug over can message) mode • P21.6: Emergency Stop can be overruled in JTAG mode if this pin is used as TDI • P21.7: Emergency Stop can be overruled in JTAG or Three Pin DAP mode. • P33.0-7, P33.9-15 and P34.1: Emergency Stop can be overruled by the 8-Bit Standby Controller (SCR), if implemented. Overruling can be disabled via the control register P33_PCSR and P34_PCSR. On pins with LVDS TX pads the Emergency Stop affects only the CMOS driver not the LVDS driver. Thus only when LPCRx.TX_EN selects CMOS mode the output is switched off. When TX_EN selects LVDS mode the output is not switched off. 0_B **DIS**: Emergency stop function disabled. The output line remains connected (alternate function). 1_B **EN**: Emergency stop function enabled. The mapped output function is disconnected and the safe state is entered by switching to input function with internal pull-up connected or tri-state, depending on the configured value of the corresponding Port slice PADCFGp_SAFSEC.ESR_PD register.
ESR_PPL	1	rwh	**Emergency status Push-Pull level** Value driven when pin is put in push-pull setup by ESR_PD 0_B The onput level of Pn.x is 0. 1_B The onput level of Pn.x is 1.

(table continues...)

(continued)

Field	Bits	Type	Description
ESR_PD	5:4	rwh	**Emergency status Pad configuration** This Bitfield defines the PAD mode when ESR is active and ESR_EN = 1 00_B **PP**: PAD in Push-PULL configuration when ESR is active. Level defined by field ESR_PPL 01_B **TPU**: Tristate, weak pull-up 10_B **TPD**: Tristate, weak pull-down 11_B **RES**: Reserved - defaults to Tristate Pull Down
0	3:2, 31:6	r	**Reserved** Read as 0; should be written with 0

Table 2509 Reset values

Applies to P00_PADCFGp_SAFSEC (p=0-15)
Applies to P01_PADCFGp_SAFSEC (p=0-15)
Applies to P02_PADCFGp_SAFSEC (p=0-15)
Applies to P03_PADCFGp_SAFSEC (p=0-15)
Applies to P04_PADCFGp_SAFSEC (p=0-9)
Applies to P10_PADCFGp_SAFSEC (p=0-15)
Applies to P13_PADCFGp_SAFSEC (p=0-15)
Applies to P14_PADCFGp_SAFSEC (p=0-15)
Applies to P15_PADCFGp_SAFSEC (p=0-15)
Applies to P16_PADCFGp_SAFSEC (p=0-14)

Reset type	Reset value	Note
Application Reset	$0000\ 0000_H$	
Kernel Reset	$0000\ 0000_H$	

47.8.4.110 Port port_no Pad Safety and Security Control Register for padp (port_no=20;p=0-1,3,6-14) (port_no=21;p=0-1,3-7) (port_no=22;p=0-11) (port_no=23,31;p=0-7) (port_no=25;p=0-1) (port_no=30;p=0-15) (port_no=33;p=0-7,9-15) (port_no=34-35;p=0-5)

P20_PADCFGp_SAFSEC (p=0-1,3,6-14) Offset address: $30C_H+p*10_H$

Port 20 Pad Safety and Security Control Register for padp — Reset values see: Table 2510

P21_PADCFGp_SAFSEC (p=0-1,3-7) Offset address: $30C_H+p*10_H$

Port 21 Pad Safety and Security Control Register for padp — Reset values see: Table 2510

P22_PADCFGp_SAFSEC (p=0-11) Offset address: $30C_H+p*10_H$

Port 22 Pad Safety and Security Control Register for padp — Reset values see: Table 2510

P23_PADCFGp_SAFSEC (p=0-7) Offset address: $30C_H+p*10_H$

Port 23 Pad Safety and Security Control Register for padp — Reset values see: Table 2510

P25_PADCFGp_SAFSEC (p=0-1) Offset address: $30C_H+p*10_H$

Port 25 Pad Safety and Security Control Register for padp Reset values see: Table 2510

P30_PADCFGp_SAFSEC (p=0-15) Offset address: $30C_H+p*10_H$

Port 30 Pad Safety and Security Control Register for padp Reset values see: Table 2510

P31_PADCFGp_SAFSEC (p=0-7) Offset address: $30C_H+p*10_H$

Port 31 Pad Safety and Security Control Register for padp Reset values see: Table 2510

P33_PADCFGp_SAFSEC (p=0-7,9-15) Offset address: $30C_H+p*10_H$

Port 33 Pad Safety and Security Control Register for padp Reset values see: Table 2510

P34_PADCFGp_SAFSEC (p=0-5) Offset address: $30C_H+p*10_H$

Port 34 Pad Safety and Security Control Register for padp Reset values see: Table 2510

P35_PADCFGp_SAFSEC (p=0-5) Offset address: $30C_H+p*10_H$

Port 35 Pad Safety and Security Control Register for padp Reset values see: Table 2510

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0										ESR_PD		0		ESR_PPL	ESR_EN
r										rwh		r		rwh	rwh

Field	Bits	Type	Description
ESR_EN	0	rwh	**Emergency Request Action Enable** Most GPIO lines have an emergency stop logic implemented (see Figure “PORTS block diagram” in the Family chapter). Each of these GPIO lines has its own emergency stop enable bit ENx that is located in the emergency stop field ESR of Port Pin n. If the emergency stop signal becomes active, one of two states can be selected. Exceptions for Emergency Stop Implementation The Emergency Stop function is available for all GPIO Ports with the following exceptions: • Not available for P20.2 (General Purpose Input/GPI only, overlayed with Testmode) • Not available for P40_x and P41_x (analog input ANx overlayed with GPI) • Not available for P21.2 (used as SMU_EXT_EMERGENCYSTOP_REQ pin). • Not available for dedicated I/O without General Purpose Output function (e.g ESRx, TMS, TCK) The Emergency Stop function can be overruled on the following GPIO Ports: • P00.x: Emergency Stop can be overruled by the TMADC. Overruling can be disabled via the control register P00_PCSR. • P14.0 and P14.1: Emergency Stop can be overruled in the DXCPL (DAP over CAN physical layer) mode. No Overruling in the DXCM (Debug over can message) mode • P21.6: Emergency Stop can be overruled in JTAG mode if this pin is used as TDI • P21.7: Emergency Stop can be overruled in JTAG or Three Pin DAP mode. • P33.0-7, P33.9-15 and P34.1: Emergency Stop can be overruled by the 8-Bit Standby Controller (SCR), if implemented. Overruling can be disabled via the control register P33_PCSR and P34_PCSR. On pins with LVDS TX pads the Emergency Stop affects only the CMOS driver not the LVDS driver. Thus only when LPCRx.TX_EN selects CMOS mode the output is switched off. When TX_EN selects LVDS mode the output is not switched off. 0_B **DIS**: Emergency stop function disabled. The output line remains connected (alternate function). 1_B **EN**: Emergency stop function enabled. The mapped output function is disconnected and the safe state is entered by switching to input function with internal pull-up connected or tri-state, depending on the configured value of the corresponding Port slice PADCFGp_SAFSEC.ESR_PD register.
ESR_PPL	1	rwh	**Emergency status Push-Pull level** Value driven when pin is put in push-pull setup by ESR_PD 0_B The onput level of Pn.x is 0. 1_B The onput level of Pn.x is 1.

(table continues...)

(continued)

Field	Bits	Type	Description
ESR_PD	5:4	rwh	**Emergency status Pad configuration** This Bitfield defines the PAD mode when ESR is active and ESR_EN = 1 00_B **PP**: PAD in Push-PULL configuration when ESR is active. Level defined by field ESR_PPL 01_B **TPU**: Tristate, weak pull-up 10_B **TPD**: Tristate, weak pull-down 11_B **RES**: Reserved - defaults to Tristate Pull Down
0	3:2, 31:6	r	**Reserved** Read as 0; should be written with 0

Table 2510 Reset values

Applies to P20_PADCFGp_SAFSEC (p=0-1,3,6-14)
Applies to P21_PADCFGp_SAFSEC (p=0-1,3-7)
Applies to P22_PADCFGp_SAFSEC (p=0-11)
Applies to P23_PADCFGp_SAFSEC (p=0-7)
Applies to P25_PADCFGp_SAFSEC (p=0-1)
Applies to P30_PADCFGp_SAFSEC (p=0-15)
Applies to P31_PADCFGp_SAFSEC (p=0-7)
Applies to P33_PADCFGp_SAFSEC (p=0-7,9-15)
Applies to P34_PADCFGp_SAFSEC (p=0-5)
Applies to P35_PADCFGp_SAFSEC (p=0-5)

Reset type	Reset value	Note
Application Reset	$0000\ 0000_H$	
Kernel Reset	$0000\ 0000_H$	

47.8.4.111 Port 32 Pad Safety and Security Control Register for padp

P32_PADCFGp_SAFSEC (p=2,4-7)

Offset address: $30C_H+p*10_H$

Port 32 Pad Safety and Security Control Register for padp

Reset values see: Table 2511

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0										ESR_PD		0		ESR_PPL	ESR_EN
r										rwh		r		rwh	rwh

Field	Bits	Type	Description
ESR_EN	0	rwh	**Emergency Request Action Enable** Most GPIO lines have an emergency stop logic implemented (see Figure “PORTS block diagram” in the Family chapter). Each of these GPIO lines has its own emergency stop enable bit ENx that is located in the emergency stop field ESR of Port Pin n. If the emergency stop signal becomes active, one of two states can be selected. Exceptions for Emergency Stop Implementation The Emergency Stop function is available for all GPIO Ports with the following exceptions: • Not available for P20.2 (General Purpose Input/GPI only, overlayed with Testmode) • Not available for P40_x and P41_x (analog input ANx overlayed with GPI) • Not available for P21.2 (used as SMU_EXT_EMERGENCYSTOP_REQ pin). • Not available for dedicated I/O without General Purpose Output function (e.g ESRx, TMS, TCK) The Emergency Stop function can be overruled on the following GPIO Ports: • P00.x: Emergency Stop can be overruled by the TMADC. Overruling can be disabled via the control register P00_PCSR. • P14.0 and P14.1: Emergency Stop can be overruled in the DXCPL (DAP over CAN physical layer) mode. No Overruling in the DXCM (Debug over can message) mode • P21.6: Emergency Stop can be overruled in JTAG mode if this pin is used as TDI • P21.7: Emergency Stop can be overruled in JTAG or Three Pin DAP mode. • P33.0-7, P33.9-15 and P34.1: Emergency Stop can be overruled by the 8-Bit Standby Controller (SCR), if implemented. Overruling can be disabled via the control register P33_PCSR and P34_PCSR. On pins with LVDS TX pads the Emergency Stop affects only the CMOS driver not the LVDS driver. Thus only when LPCRx.TX_EN selects CMOS mode the output is switched off. When TX_EN selects LVDS mode the output is not switched off. 0_B **DIS**: Emergency stop function disabled. The output line remains connected (alternate function). 1_B **EN**: Emergency stop function enabled. The mapped output function is disconnected and the safe state is entered by switching to input function with internal pull-up connected or tri-state, depending on the configured value of the corresponding Port slice PADCFGp_SAFSEC.ESR_PD register.
ESR_PPL	1	rwh	**Emergency status Push-Pull level** Value driven when pin is put in push-pull setup by ESR_PD 0_B The onput level of Pn.x is 0. 1_B The onput level of Pn.x is 1.

(table continues...)

(continued)

Field	Bits	Type	Description
ESR_PD	5:4	rwh	**Emergency status Pad configuration** This Bitfield defines the PAD mode when ESR is active and ESR_EN = 1 00_B **PP**: PAD in Push-PULL configuration when ESR is active. Level defined by field ESR_PPL 01_B **TPU**: Tristate, weak pull-up 10_B **TPD**: Tristate, weak pull-down 11_B **RES**: Reserved - defaults to Tristate Pull Down
0	3:2, 31:6	r	**Reserved** Read as 0; should be written with 0

Table 2511 Reset values of P32_PADCFGp_SAFSEC (p=2,4-7)

Reset type	Reset value	Note
Application Reset	$0000\ 0000_H$	
Kernel Reset	$0000\ 0000_H$	

47.8.4.112 Port port_no Pad Safety and Security Control Register for padp (port_no=20;p=2,4-5) (port_no=21;p=2) (port_no=32;p=0-1,3) (port_no=33;p=8) (port_no=40;p=0-15)

P32_PADCFGp_SAFSEC (p=0-1,3) Offset address: $30C_H+p*10_H$

Port 32 Pad Safety and Security Control Register for padp — Reset values see: Table 2512

P40_PADCFGp_SAFSEC (p=0-15) Offset address: $30C_H+p*10_H$

Port 40 Pad Safety and Security Control Register for padp — Reset values see: Table 2512

P20_PADCFGp_SAFSEC (p=2,4-5) Offset address: $30C_H+p*10_H$

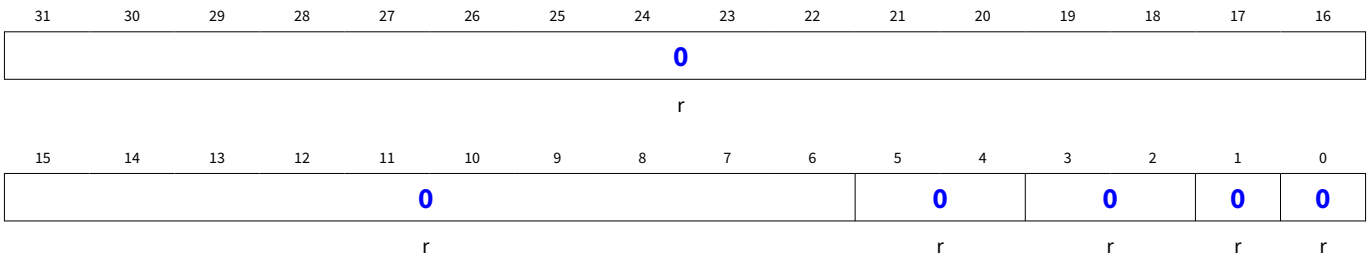
Port 20 Pad Safety and Security Control Register for padp — Reset values see: Table 2512

P21_PADCFGp_SAFSEC (p=2) Offset address: $30C_H+p*10_H$

Port 21 Pad Safety and Security Control Register for padp — Reset values see: Table 2512

P33_PADCFGp_SAFSEC (p=8) Offset address: $30C_H+p*10_H$

Port 33 Pad Safety and Security Control Register for padp — Reset values see: Table 2512



Field	Bits	Type	Description
0	0, 1, 3:2, 5:4, 31:6	r	**Reserved** Read as 0; should be written with 0

Table 2512 Reset values

Applies to P20_PADCFGp_SAFSEC (p=2,4-5)
Applies to P21_PADCFGp_SAFSEC (p=2)
Applies to P32_PADCFGp_SAFSEC (p=0-1,3)
Applies to P33_PADCFGp_SAFSEC (p=8)
Applies to P40_PADCFGp_SAFSEC (p=0-15)

Reset type	Reset value	Note
Application Reset	$0000\ 0000_H$	
Kernel Reset	$0000\ 0000_H$	

47.8.5 TC4Dx PORTS connectivity

The connectivity of PORTS can be found within the corresponding device datasheet.

47.8.6 TC4Dx PORTS revision history

Reference	Description of change(s)
Date range: 2024-08-17 to 2025-03-13	
Padframe alarm trigger signals	• Section "Padframe alarm trigger signals" moved to device specific functional description (TC49x-N and TC4Dx only)

48 Firmware

Firmware (FW) consists of software components that are stored in read-only memory (ROM). It performs start-up of the device after power-on or reset, and supports additional features such as handling debug access to the device, bootstrap loading of executable code, controlled shut-down after reset request, and power-friendly debug monitor routines for example.

48.1 Feature list

- Real-time cluster (RTC) or system firmware (SYS_FW)
- Cyber-security (CSRM, SECURE) sub-system firmware, abbreviated as CSRM_FW
- User code start address selection and triggering execution
- Standby Controller (SCR) sub-system firmware, abbreviated as SCR_FW

48.2 Functional overview

The figures below summarize the start-up behavior of the device as firmware-controlled after different reset types.

48 Firmware

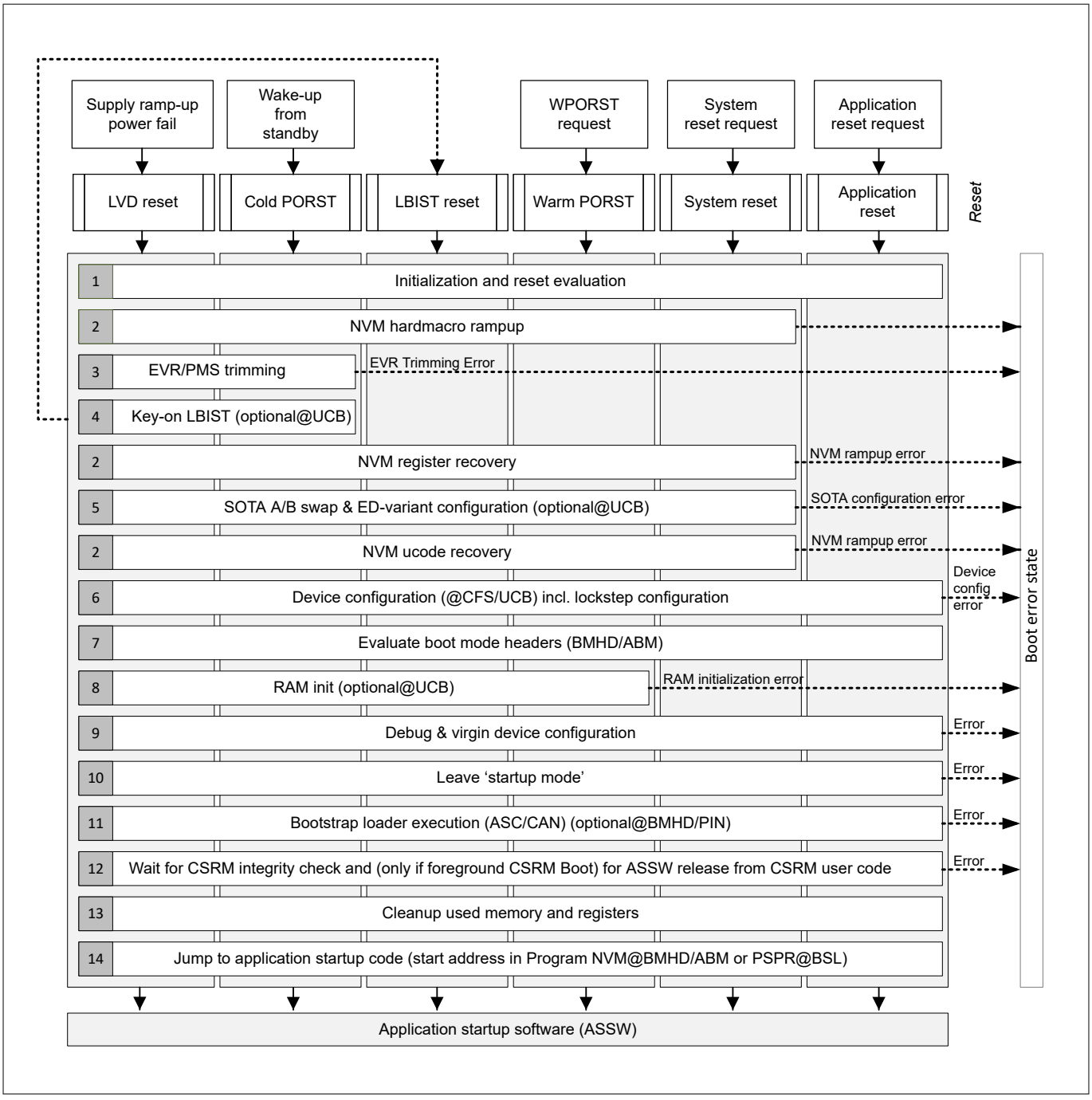


Figure 1256 System firmware flow (HOST cluster)

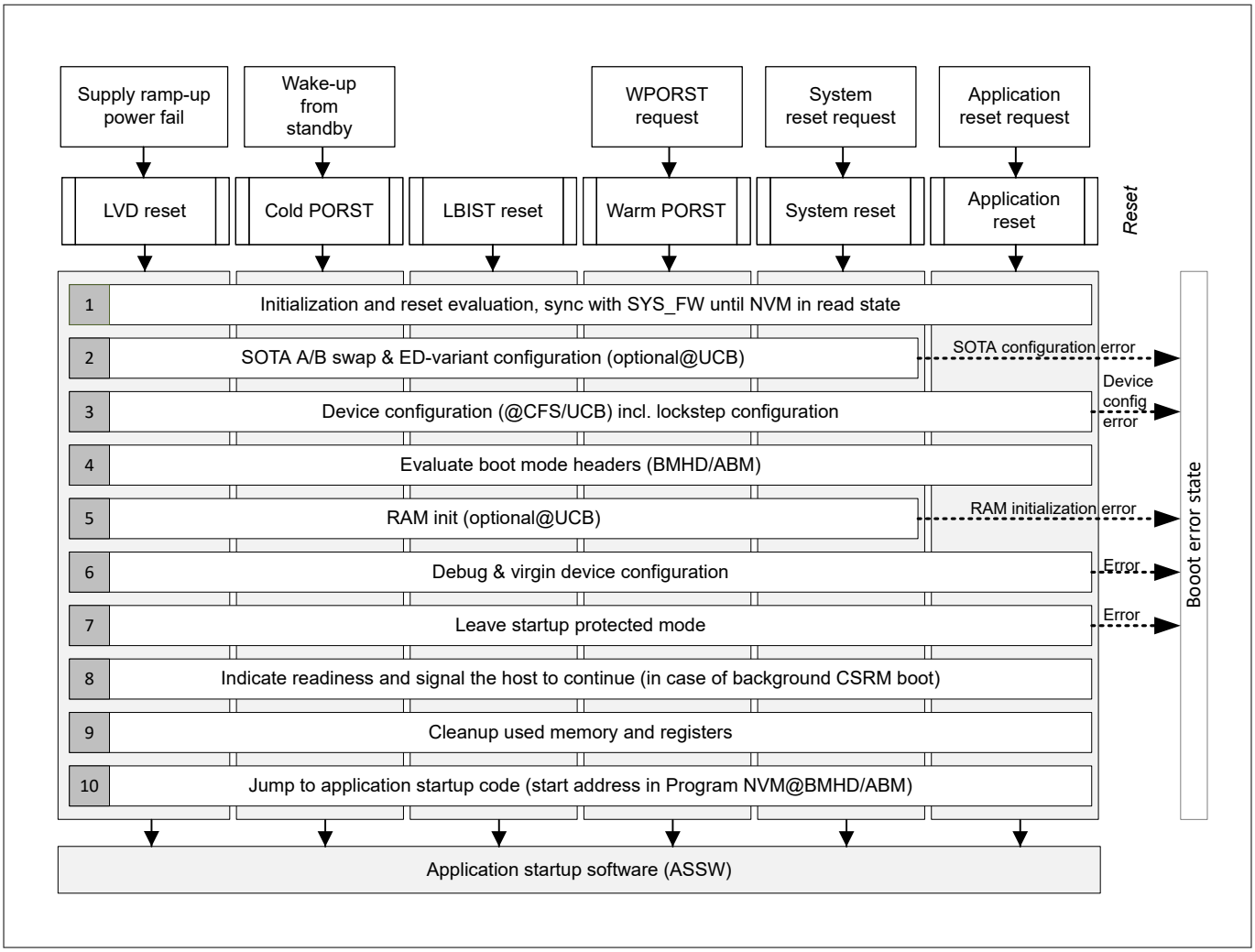


Figure 1257 CSRM firmware flow (SECURE cluster)

Attention*: *Hardware resources used by Firmware

The start-up procedure executed during device boot uses some memories, which means that they can be overwritten during start-up and the content is not preserved through resets, as follows:

- Data scratch-pad RAM (DSPR) of CPU0 and CPUcs - the first 8KBytes, upon any device reset
- Program scratch-pad RAM (PSPR) of CPU0 and CPUcs - the first 2KBytes, upon any device reset

Related information

TC4Dx SMU alarm mapping tables on page 7233

48.3 Functional description

48.3.1 Device dual-core start-up

After every device reset event, Firmware execution starts automatically in parallel on two Central Processing Units (CPU):

- CPU0
  - Also called the Boot Core of the HOST cluster, executing SYS_FW
- Cyber security core (CPUcs)
  - Also called the Boot Core of the SECURE cluster, executing CSRM_FW

It is important to understand that SYS_FW and CSRM_FW work in parallel and are synchronized to ensure for the correct and aligned boot of both the HOST and SECURE clusters.

In addition, it should be noted that:

- CPU0 and CPUcs are only affected by device resets. They are not affected by module or kernel resets
- SCR core is not automatically started by the system. It must be triggered by the user through another core, which may eventually lead to SCR_FW execution
- PPU_FW consists only of functions that support load jump limits during shutdown and software debugging

At the end of the start-up performed by SYS_FW and CSRM_FW, the device is configured and prepared for user software execution according to data stored in the embedded non-volatile memory (NVM) of the device. As the last step from their execution, both the SYS_FW and CSRM_FW perform a jump to the first addresses of respective user software.

48.3.1.1 Variations in the start-up sequence

There are different types of device resets supported in the TC4xx family, all triggering SYS_FW and CSRM_FW execution.

Because the reset types affect different parts of the device, the FW activity after a given reset type also varies. The main variations are distinguished for the following events:

- Power-on, LBIST and system resets
- Application reset

Power-on, LBIST, and system resets

These reset events affect most of the devices functional blocks (FBs), by putting them into their configuration reference state. Most importantly this impacts the clock system and the NVM.

At the end of FW execution after one of these resets:

- Back-up oscillator is used as clock source for the device. The main clock signals have a frequency of 100 MHz or are derived from it keeping the recommended settings. For the clock settings after boot FW phase, refer to “After boot-FW value” information for clock control registers in CLOCKING Chapter of the UM.

  ***Note**: If you have configured the clock system in accordance to the application needs, so that for example PLL is locked and the device is supplied with a higher frequency than 100 MHz, this configuration will be lost after power-on, LBIST, and system resets, and must be re-implemented afterwards.*

- The NVM is ready for execution of all the operations, such as read and program for example

***Note**: If you configure the clock system in accordance to the application needs, so that for example PLL is locked and the device is supplied with a higher frequency than 100 MHz, then the NVM clock dividers must be re-configured to comply with NVM speed limitations.*

After system reset the SYS_FW has a single flow. However there are different flows after power-related and LBIST reset:

- After LVD reset:
  - The SYS_FW performs full device initialization including power system trimming
- After wake-up from stand-by mode:
  - The SYS_FW activity is nearly the same as after LVD reset, except that the SRAMs (volatile memories) being powered during stand-by mode are not initialized, even if selected for initialization after cold power-on according to the user configuration
- After warm power-on and LBIST reset:
  - The SYS_FW flow considers the reset event as “warm power-on”. The power-related components are not touched

Application reset

This type of reset puts fewer functional parts into configuration reference states and the FW activity is the shortest. In particular:

- Clock system configuration is not changed by the application reset or afterward by the firmware
- No memory content is modified, except the memory used by the firmware itself

Please refer to the figures in the 'Start-up sequence' section for the boot flow after different reset types.

Related information

Crystal oscillator control register on page 5582

48.3.2 Start-up in mission mode (ASIL-D)

The primary purpose of the TC4xx Firmware is to execute safe (ASIL-D) device start-up in so-called 'mission mode'. This is the usual production use of the device as the core part of an automotive ECU (Electronic Control Unit) for example.

The main other non-mission-mode features include development and debug support, which are classified as QM.

***Note**: QM refers to ‘Quality Managed’ – meaning that the development process for such components follows standard and repeatable methodology for development.*

Only after adding some safety mechanisms – e.g. for achieving freedom from interference (FFI) on top of a QM system - given ASIL-level resp. safety compliance can be achieved.

Such features can be invoked within the start-up sequence on user-request, yet they are not meant to be an inherent part of safe mission-mode. QM-feature(s) invocation during the boot is indicated by SYS_FW to the application software (refer to “Exit information after FW execution” Section).

Once development is finalized, deactivation of QM-feature(s) is mandatory in order to support ASIL-claims for boot procedure.

***Note**: CSRM_FW interaction with SYS_FW does not compromise the overall safe start-up of the device in mission mode because of sufficient Freedom From Interference (FFI), which is ensured by the product architecture.*

In case of an error occurring during start-up, such as a lockstep error on CPU0 for example, firmware performs error-handling, resulting in up to 2 additional reboot cycles (refer also to "Firmware exit information" Section).

48 Firmware

The major steps taken by Firmware during safe device start-up are described in the following sections, and the flows are shown in figures at the end of this chapter.

48.3.2.1 Device configuration and trimming of Infineon proprietary features

Configuration and trimming data stored by Infineon during production into NVM is used by Firmware to install essential device settings that ensure correct microcontroller functionality under the specified conditions.

Some data is not installed by Firmware, but can be read by user software. An example of a reason for reading such data is to identify on which exact type of device the user software is running (see CHIPID below).

The settings and data are:

1. Power-management system (PMS) trimming

   The main components covered are the embedded voltage regulators (EVR) and the back-up clock oscillator.

   After this trimming the jitter of the internal oscillator is limited, allowing robust execution of complete start-up. This (fBACK) clock is used as device clock source during the complete start-up procedure after system and power-on resets
2. Variant and package specific configurations

   The following parameters are part of the variant specific configurations:

   - Module and/or pin availability and configuration
   - Timing parameters of SRAMs for optimized performance

     The information from the above groups is installed by Firmware during device start-up in the respective registers.
   - Device identification information on product variant level

   This information can be read by application software from UCB0, where two types of data are available:

   - compact chip-identification within one 32-bit word - CHIPID
   - full product-name known as the SAK number, available for reading by user software in the form of 16-symbol ASCII-coded text - refer to PRODUCT_NAME0 … PRODUCT_NAME3
3. Chip specific configurations:

   - Chip-individual trimming and calibration
   - Security keys
   - Device identification information on chip-unique level

   The device identification information on chip-unique level can be read by the user software from UCID0 … UCID2 and Flash Microcode Version.
4. LBIST configuration and execution

   The LBIST configuration is specific for each AURIX™ derivative, and is optimized for execution time and failure coverage. LBIST configuration is installed into respective control registers by Firmware after cold power-on. LBIST can be triggered during the boot by Firmware if selected by user-programmable settings.

   User selections must be stored in UCB0_17 - refer to Table 2533 - and the detailed description of User selection for LBIST execution.

   If LBIST is triggered by Firmware, the result is available for evaluation by user software after finishing the start-up. The expected LBIST results can be read from LBIST_STAT_RESx_KEYON locations in UCB0, where x represents the target scan domain (x= 0..7).

   The same settings as installed during start-up, if not changed by application, may be used for later user-triggered LBIST execution

   ***Note**: The LBIST configuration settings are always installed after cold power-on, independent of whether LBIST execution during start-up is activated in LBISTEXE.LBEX.*

48.3.2.2 Start-up handling based on user-programmable data

Apart from the Infineon proprietary data stored during production, Firmware also processes and reacts to data stored into NVM by the user.

Attention**:* ***Any incorrectness of the user-programmable data identified during boot will cause the fall-back mechanism to be executed. Note, that following from this for example: even if there is a correct Boot-Mode Header (BMHD) configured but another user configuration is invalid, the device will not execute the start-up according to that BMHD, but according to the fall-back mechanism.
The types of incorrectness which can be identified include (refer to the below definitions):
- wrong system address of this location (SAL) value
- wrong cyclic redundancy check (CRC) value
- undefined/reserved value for some parameter within user-programmable data

Definition of terms used in Register Section:

- System address of this location (SAL)

This 32-bit address is calculated adding the "Offset address" for the location/register shown in its description to the "Base address" of the considered Configuration area module - refer to:

- - Table 2525 for UCB0 module
  - Table 2543 for UCB1 module

The purpose of adding this information is to assure the device start-up against address errors during reading configuration data from NVM. This SAL entry corresponds to the actual address inside the NVM (refer to Configuration area overview table), where from the reading of a configuration data portion starts.

- Length of this table (LEN)

This is the number of 32-bit words, including SAL, LEN itself, all the data in the table and the CRC at its end.

- Check result for this table (CRC)

The validity of configuration data read from NVM is assured by performing a CRC of the data. The data is only used if the check passes.

For the check, IEEE 802.3 standard CRC32 Ethernet polynomial 0x04C11DB7 is used - another representation:
$x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$.

48.3.2.2.1 Device handling in delivery state

In delivery state of the device, there are different contents in the various parts of user-programmable data in configuration area/UCB:

- the most of user-programmable data is left in erased state/unprogrammed - meaning all data content is zero and also no valid CRCs are stored, therefore the respective data-blocks are in fact invalid

  If a device is started with such UCB content - no user-code will be started but the firmware will execute so-called "Fall-back mechanism" (refer to the respective Section) allowing the correct user-configuration data to be programmed without erase before that, and then the device started properly
- the following UCBs are programmed during production because if left unprogrammed the device will go into error-state - the information there is security-critical and special coding is used, these are:
  - UCB_CS_DBG_ORIG/COPY (refer to Table 2544)
  - UCB_CS_USERCFG_ORIG/COPY (refer to Table 2553)
  - UCB_RTC_USERCFG_ORIG/COPY (refer to Table 2533)

  If user wants to change the default/programmed state - the respective UCB must be first erased and only afterwards programmed with the target content.

48.3.2.2.2 User-specific device configurations

User configurable options which are handled by Firmware and affect device start-up include:

1. Application reset and/or ESR0 or ESR2 output pin release prolongation.

   This is to allow better start-up synchronization between the microcontroller and other ECU components if or when you forsee such synchronization.

   The values to be installed into ESRxCNTCTRL registers must be stored in UCB0_17 (refer to Table 2533) at SMM_EXTIF_ESR0CNTCTRL and PMS_PAD_ESR2CON locations, according to the register layout described in the SMM and PMS chapters

2. Selectable SRAM initialization and auto-initialization.

   The customer can select - installing respective configuration data in UCB0_17 (refer to Table 2533) - whether the FW execution will impact device SRAMs, in particular:

   - initialize some SRAMs during start-up - for this feature, refer to "SRAM initialization" Section
   - enable or not the auto-initialization of some SRAMs upon enabling of the respective SSH module - refer to "Security-sensitive memories and auto-initialization" in VMT User Manual

3. LBIST execution during start-up

   User selections in regard to this functionality must be stored in UCB0_17 (refer to Table 2533). In delivery state of the device, the LBIST during start-up is deactivated.

   If the user wants to use this feature and activates it in User selection for LBIST execution.LBEX , the following should be considered in regard to customer settings in User selection for LBIST execution.LTC :

   - if LTC = 0..6 – the respective number of sequential unexpected power-drops (1..7) during LBIST execution leading to power-reset and LBIST re-start will at the end put the device into Boot Error state, which is a safe state and can be identified from outside (refer to “Firmware exit information” Section)
   - if LTC = 7 – as long as sequential unexpected power-drops happen during LBIST execution leading to power-reset, LBIST will be continuously restarted; only upon a successful LBIST execution the device will continue and exit the boot phase

4. Re-configuring the Program NVM address map.

   This feature provides the ability to emulate code execution on another device. The advantage of this feature is the easy portability of code, developed and debugged on one device to another one, without the need to relocate software images stored in Program NVM.

   Selections if address re-routing for a device should be activated during start-up must be stored in UCB0_17 (refer to Table 2533) at User selection for ADDRESS ROUTING location.

   The devices from TC4xx family are divided into two groups, and address re-routing is possible only for the devices from a given group. The groups are:

   - Group 1: TC4Dx, TC49xA, TC48x, TC44x, TC42x, TC4Zx
   - Group 2: TC49xN, TC46x, TC4Ex, TC4Px

   Attention**: **The optional configuration described here only affects the access from the CPU side to PFLASH (all CPUs and PFLASH banks) and nothing else. Therefore this feature has to be considered as address re-routing only and not as “full device emulation”.

Additionally to the above, there are user-configuration values which are not handled by Firmware and have no impact on the device start-up - there are two types of such values and respective locations in NVM:

- values, stored by Infineon during production, available for customer usage - these are collected in UCB0_38, refer to Table 2542
- locations, left unprogrammed during production and available for the customer to store there whatever data can be useful for his application - UCB0_15 and UCB0_16, refer to Table 2531, Table 2532

48.3.2.2.3 Boot Mode Lock (BML)

This feature, once installed in UCB, prevents the device from being accessed and user code from being started if invalid boot-mode or other user-programmable information is stored in UCB. See the 'Boot mode evaluation' section. The user selection whether BML is activated must be stored in UCB1 - refer to BML

Attention**:* ***BML is the final switch to set a device into productive state - afterward the Fall-back mechanism is not supported anymore, respectively any configuration-data failure will put the device into errored state. If the failure is permanent - meaning not resolved by consecutive resets - the device can not be booted anymore.

48.3.2.2.4 Software Over-the-Air (SOTA)

SOTA allows 'on-the-fly' update of the software executed on a microcontroller. At any point of time, placeholders for two user-software images are available in NVM. While the microcontroller is executing code from one of these placeholders, the other one can be re-programmed with an alternative code image.

However this is not possible for CPUcs. Aside from having placeholders for two code images, PFLASHcs cannot support reading or executing and programming or verifying simultaneously. This means that while modifying one of the images, no code execution is possible from PFLASHcs.

***Note**:*

- *You must take care that the alternative code is correctly programmed and error-free before selecting it as the active code*
- *The active used or executed code-image is selected from information stored by you into a UCB. The respective UCB and its usage allow for a large number of re-programming cycles during the overall life-time of the device*

The SOTA feature is supported in TC4xx independently for both the HOST and SECURE clusters. This means that the code for execution on either CPUx and/or CPUcs can be fetched from one out of two alternate NVM images for each.

***Note**:* *Because SOTA is supported independently for both HOST and SECURE clusters, if the code images running on the HOST and SECURE clusters must be 'in sync' in regard to the current SOTA or SWAP selections, such synchronization needs to be assured by the user software.*

Your selection if the SOTA feature is in general enabled must be stored:

- For the HOST cluster, in UCB0 at User selection for SWAP activation
- For the SECURE cluster, in UCB1 at User selection for SWAP activation in CS

The selection of the active SWAP configuration is specified in UCB0 for HOST cluster (refer to Register Overview UCB0_19), and UCB1 for the SECURE cluster (refer to Register Overview UCB1_12).

Up to 32 SWAP configurations may be programmed sequentially into any of the UCBs, for HOST and SECURE clusters respectively, without erasing.

After programming and verifying a new SWAP configuration, the previous one must be invalidated by over-programming the SAL and STATUS locations to all-1, which is possible without erasing. The general rules for UCB ORIG and COPY handling (refer to "UCB COPY and ORIG update procedure" Section in NVM Chapter) must be strictly followed for UCB_SWAP and UCB_SWAPcs.

After filling up all the 32 placeholders, the respective UCB should be erased and configuration storage restarted from the beginning of the UCB.

48.3.2.2.5 SRAM initialization

According to the user-programmed information in UCB, Firmware can initialize selected SRAMs both in the HOST and/or SECURE clusters. 'Initialization' here means:

- The data content is set to all-zero
- ECC (error-correction code) values are installed to 'valid'

After initializing the SRAMs can be read, even before being written, and the read will not trigger an ECC error unless there is a real error.

***Note**: Initialization is only done for SRAMs located in power-domains which are supplied during start-up.*

Your selections which SRAMs from the HOST cluster shall be initialized during start-up must be stored in UCB0 - refer to RAMINIT.

Your selections which SRAMs from the SECURE cluster shall be initialized during start-up must be stored in UCB1 - refer to UCB1_15_CSSRAM_init.

48.3.2.3 User code start

The main task of the TC4xx Firmware, targeting both the HOST and SECURE cores, is to prepare and transfer the code execution on CPU0 and CPUcs to user software.

48.3.2.3.1 Boot options

This section summarizes the start-up configurations.

Internal start

In this basic start-up mode, the first user instruction is fetched from the Internal Program NVM of the device.

The user code start address is configurable in the Boot Mode Header data structures, which are programmable in UCB_BMHDx.

Bootstrap loader modes

Different Bootstrap Loader routines are used in these modes to download code or data into the Program Scratchpad RAM of CPU0 (CPU0_PSPR).

The supported Bootloader selections are:

- ASC Bootloader:
  - ASC communication protocol through ASC pins
- Generic Bootloader through CAN pins:
  - The communication protocol is automatically selected by the Firmware between ASC and CAN

After downloading the code, the user code execution is started from the first location in CPU0 Program Scratchpad RAM (PSPR).

Alternate boot modes (ABM)

In these alternate Boot modes, program code is started from a user-defined address, but only if all defined conditions are satisfied. If the conditions are not met, the Firmware can fall back to a Bootstrap Loader mode or Debug Interface connection, to allow download and execution of user code.

All the information needed for the Firmware to handle ABM start-up mode is collected into Alternate Boot Mode Headers (ABMHD).

If this mode is selected and the ABMHD is valid, the user code start address is set to the respective value from the Alternate Boot Mode Header.

48.3.2.3.2 Boot mode evaluation

The Firmware fetches the start address to transfer code execution after device start-up based on user-programmable information structures - called Boot Mode Headers - from the UCBs.

For both the HOST and SECURE clusters, Boot Mode Headers (BMHDx, x=0...3) are evaluated basically with the same sequence:

- The evaluation starts with BMHD0 and goes through up to BMHD3
- For any evaluated BMHDx
  - The confirmation state must be CONFIRMED or UNLOCKED, else the target BMHDx is considered as invalid
  - All the entries must be valid according to the definitions in the next Sections, else the target BMHDx is considered as invalid
  - All the CRC checks must pass, else the target BMHDx is considered as invalid
- Once a valid BMHD is found - it is taken for execution, the rest Boot Mode Headers (if any) remain unchecked
- If no valid BMHD is found - the processing continues as described in "Fall-back mechanism'" section

The Boot Mode Header data-structures allow for selection between:

- Starting the user code directly from a defined NVM address in 'Internal start' mode
- Checking using CRC for validity, a defined NVM address range and start the execution from a defined address if the check is successful, in Alternate Boot mode (ABM)
- For CPU0 (HOST cluster) only:
  - Download code into program scratchpad (volatile memory) of the boot core using a supported communication interface, then start the downloaded code in Bootstrap Loader (BSL) mode

***Note**: All the CRC calculations are made using the TriCore™ instruction CRC32 which implements the IEEE 802.3 standard: CRC-32 polynomial is used and the CRC result is bit-reversed and inverted (where applicable, for ABMHD - refer to "Alternate Boot Mode" Section).*

HOST cluster

***Attention**: For device-specific definitions - refer to "TCxx FIRMWARE information" Section.*

For the HOST cluster, the start-up configuration is stored in one out of four Boot Mode Headers, from UCB_RTC_BMHD0 through to UCB_RTC_BMHD3. Please refer to the detailed structure below, and to:

- Register overview UCB0_09
- Register overview UCB0_10
- Register overview UCB0_11
- Register overview UCB0_12

Table 2513 Boot mode header (UCB_RTC_BMHDx) structure for RTC (HOST cluster)

Field name	Sub-field	Description
SAL	---	System Address of this Location, 32 bit
BMI	---	Boot Mode Index, 16 bit
	reserved bit [0]	Reserved. Must be configured to 1 in UCB_BMHDx

(table continues...)

Table 2513 (continued) Boot mode header (UCB_RTC_BMHDx) structure for RTC (HOST cluster)

Field name	Sub-field	Description
	MODE bits [3:1]	Start-up mode selection: 111_B Internal start from NVM 110_B Alternate Boot Mode (ABM) 100_B Generic Bootstrap Loader Mode (ASC/CAN BSL) 011_B ASC Bootstrap Loader Mode (ASC BSL) else invalid
	reserved bits [15:4]	Reserved. Must be configured to 0 in UCB_BMHDx
BMHDID	---	Boot Mode Header Identifier - 16 bit: $B359_H$ BMHDID OK else BMHDID invalid
STAD	---	Start address, which must always be inside HOST-cluster Program NVM and word-aligned, 32 bit: - if ABM selected - then it is the start address of the Alternate Boot Mode Header - if Internal start selected - then it is the start address of the user code - else - it is not considered for mode selection
CRCBMHD	---	Check result for the Boot Mode Header, 32 bit

SECURE cluster

For the SECURE cluster the start-up configuration is stored in one out of four Boot Mode Headers, from UCB_CS_BMHD0 through to UCB_CS_BMHD3. Please refer to:

- Register overview UCB1_3
- Register overview UCB1_04
- Register overview UCB1_05
- Register overview UCB1_06

Table 2514 Boot mode header (UCB_CS_BMHDx) structure for CSRM (SECURE cluster)

Field name	Sub-field	Description
SAL	---	System Address of this Location, 32 bit
BMI	---	Boot Mode Index. 16 bit
	reserved bit [0]	Reserved, must be configures to 1 in UCB_CS_BMHDx
	MODE bits [3:1]	Start-up mode selection: 111_B Internal start from NVM 110_B Alternate Boot Mode (ABM) else invalid
	reserved bits [15:4]	Reserved. Must be configured to 0 in UCB_BMHDx

(table continues...)

Table 2514 (continued) Boot mode header (UCB_CS_BMHDx) structure for CSRM (SECURE cluster)

Field name	Sub-field	Description
BMHDID	---	Boot Mode Header Identifier, 16 bit: $B359_H$ BMHDID OK else BMHDID invalid
STAD	---	Start address which must always be inside SECURE-cluster NVM and word-aligned, 32 bit: - if ABM selected - then it is the start address of the Alternate Boot Mode Header - if Internal start selected - then it is the start address of the user code - else - not considered for mode selection
CRCBMHD	---	Check result for the Boot Mode Header, 32 bit

Alternate Boot Mode

Alternate Boot Mode handling is controlled by the content of Alternate Boot Mode Headers. The structure of a single Alternate Boot Mode Header is defined below.

Table 2515 Alternate Boot Mode Header (ABMHD) structure

Offset address	Size byte	Field name	Description
00_H	4	STADABM	User Code Start Address in ABM mode
04_H	4	ABMHDID	Alternate Boot Mode Header Identifier: $FA7C\ B359_H$ - ABMHDID OK else - ABMHDID invalid
08_H	4	CHKSTART	Memory Range to be checked - Start Address
$0C_H$	4	CHKEND	Memory Range to be checked - End Address
10_H	4	CRCRANGE	Check Result for the Memory Range
14_H	4	CRCRANGE_N	Inverted Check Result for the Memory Range
18_H	4	CRCABMHD	Check Result for the ABM Header
$1C_H$	4	CRCABMHD_N	Inverted Check Result for the ABM Header

The Alternate Boot Mode Header(s) can be located starting at arbitrary word-aligned locations inside Program NVM. The start address is taken from the STAD field of the Boot Mode Header selecting ABM.

Attention*:** ***Both the NVM address areas for cached and non-cached access are considered as allowed locations by this evaluation procedure. However both CHKSTART and CHKEND must be either in cached or non-cached segment and mixing is not allowed.

***Note*:** *The size of the Memory Range to be checked must not exceed 512 KB otherwise the specified device boot-time will be violated.*

48.3.2.3.3 Boot-Mode Headers handling

All the UCBs intended to contain BMHDs can be programmed without further as long as being in UNLOCKED state.

Once set to CONFIRMED state – the UCB content becomes write-protected and password is needed to deactivate the protection and allow programming. This password should be stored by the customer at the respective locations before setting the UCB to CONFIRMED state, if not – the default password is all-zero.

Besides all the BMHD UCBs do contain locations for holding passwords, it is enough to provide correct password to anyone out of the BMHD UCBs for the respective cluster (HOST/SECURE) to unlock all the BMHD UCBs within that cluster.

The general sequence to be followed (both for HOST and SECURE cluster, short notation clst=RTC, CS below) when a new BMHD must be programmed is:

- check if there is still a not yet programmed UCB_clst_BMHDx to be used for the new BMHD
  - if not – use UCB_clst_BMHD0 for the new BMHD
- check if the target UCB for the new BMHD is in CONFIRMED state
  - if yes – provide correct password to the target or any other BMHD UCB from the respective cluster to de-activate the protection
- program and verify the new BMHD content
- invalidate the previously active BMHD UCB by over-programming the SAL and BMI_BMHDID locations to all-one
- set the target UCB to CONFIRMED state - if desired to make it protected
- once the new BMHD must be taken for device re-start – trigger a system reset

48.3.2.3.4 Fall-back mechanism

This fall-back mechanism is activated if some invalid user-configuration data is found during start-up. The processing in such a case is different for HOST and SECURE clusters, depending primarily on the user-configurable Boot Mode Lock (BML) in UCB1 and which user-configurable data failed the checks.

***Note**: Once Fall-back mechanism is activated - the device start-up is no more safe, and this is indicated by setting STMEM1.QME bit (QM-part entered) - refer to SMM Chapter.*

- In HOST cluster:
  - if BMHD UCB(s) failed AND BML evaluation in UCB1 has been successful
    - if BML is active - device enters Boot Error State, "BMHD invalid" is indicated by SMM_STMEM1.BV=0
    - if BML is not active
      - in case debug access to device is granted and halt-after-reset request is sent through the debug interface - CPU0 shall enter HARR as fall-back state
      - otherwise - SYS_FW starts Generic BSL routine, waiting to start communication for downloading code or data through CAN or ASC interface
  - If other invalid user-programmable data is found in UCBs for HOST cluster
    - device boot continues
    - default configuration settings are taken, as follows:
      - LBIST execution during boot is disabled
      - SWAP is disabled, default address map after reset remains active
      - no SRAM initialization during start-up

- auto data-init enabled
- default ESR settings after reset remain active

- In SECURE cluster:
  - if BMHD UCB(s) failed AND BML evaluation has been successful
    - if BML is active - CPUcs enters Error State meaning no user code is started in SECURE cluster, HOST boot may continue
    - if BML is not active
      - in case debug access to device is granted and CSRM debugging is enabled (in Debug configurations.CSRMDBGDIS) - CPUcs shall enter HARR as fall-back state
      - otherwise - CPUcs enters Error State meaning no user code is started in SECURE cluster, HOST boot may continue
  - if UCB_CS_SWAP evaluation failed
    - device boot continues
    - SWAPcs is disabled, default address map after reset remains active
  - if other invalid user-programmable data is found in UCBs for SECURE cluster
  - - CPUcs enters Error State meaning no user code is started in SECURE cluster
    - HOST boot may continue - this will be the case only if background non-blocking SECURE boot is configured, refer to "Foreground and background CSRM boot" Section.

The CPUcs Error State represents an endless NOP-loop, and this is the fallback behavior for SECURE cluster.

The only error-scenario in SECURE cluster which can be identified from outside is if CPUcs does not enter Error State and:

- no valid BMHD found and BML is not active, AND
- debug access is granted and CSRM debugging is enabled in Debug configurations.CSRMDBGDIS, AND
- CSCU APU has been opened by code executed on CPUcs for accesses from CBS

If the above conditions are met, CSCU_STMEMx registers can be read through debug interface and the following flags checked:

- CSCU_STMEM1.BV = 0 – no valid BMHD found
- CSCU_STMEM1.CON_E = 1 – invalid configuration found in UCB1
- CSCU_STMEM2.BOOT_ADDR = 0 – no valid boot-address for CPUcs

HARR as fall-back state

The handling in this case is basically the same as any Halt-after-reset, with one difference:

- in the generic case, the program counter of the CPU being halted is installed with the address of the first user-software instruction - according to the selected mode and start address as from BMHD information
- when HAR taken as fall-back state, the program counter of the CPU being halted is installed with the first address in the user-area of Program scratch-pad RAM (PSPR) of the respective CPU - 0xC0000800 (the first 2KB of Boot CPU PSPR are used by Firmware).This is the local address, meaning any halted Boot CPU (CPU0, CPUcs) will have PC register pointing to its own PSPR.

48.3.2.3.5 Foreground and background CSRM boot

The start-up of HOST and SECURE clusters is synchronized by interaction between SYS_FW and CSRM_FW.

User Configuration data selects between:

- Foreground CSRM boot
  - With this option CSRM_FW starts the user code execution on the CSRM CPU first. Meanwhile SYS_FW waits for a release trigger handshake from the CSRM user code

    CSRM user code must later release SYS_FW to continue its execution by installing CSCU_CSRM2HT.RL according to the register description (refer to User Manual, CSRM Chapter). Only after such a release trigger from CSRM the user code execution in the Real Time Cluster is started by SYS_FW.

    With this selection, any failure during CSRM boot terminates the start-up and the device enters error-state.
- Background CSRM boot
  - With this option the user code execution in both the HOST (on CPU0) and SECURE (on CPUcs) clusters is started by SYS_FW and CSRM_FW immediately after successful device boot. If further interaction between the clusters is needed, it must be fully handled by the user software.

    There are two options for this CSRM boot type:

    - fatal/blocking - any failure during CSRM boot terminates the start-up and the device enters error-state
    - non-blocking - upon a failure during boot, CSRM_FW enters endless loop, respectively no user-code is started in SECURE cluster. SYS_FW will be completed and user-code started in HOST cluster, if possible - e.g. if Data NVM in SECURE cluster is successfully ramped to allow reading configuration data from there.

      The CSRM boot failure is indicated by setting bit CS_E=1 in STMEM1 register

The selection which CSRM boot mode to use upon any reset type must be stored in UCB1 - refer to CSRM boot - user selection.

IMPORTANT NOTE:

Only combinations between foreground/background CSRM boot mode upon different reset types shall be configured by the user as shown in the below Table.

Table 2516 Allowed CSRM boot mode combinations vs. device reset types

Combination No.	Cold power-on	Warm power-on	System reset	Application reset
1.	Foreground	Foreground	Foreground	Foreground
2.	Foreground	Foreground	Foreground	Background
3.	Foreground	Foreground	Background	Background
4.	Background	Background	Background	Background

48.3.3 Development support

Before a device can be utilized in Mission mode, that is running as part of a real application (ECU), it must first be configured and programmed as required for the given application requirements. The Firmware therefore implements several development support features for this purpose.

***Note**: Once any of the development features are used, device start-up is no longer safe or ASIL-D compliant, because these features are not intended for use in Mission mode. This is indicated by setting STMEM1.QME bit (QM-part entered) - refer to SMM Chapter.*

Related information

Debug information on page 9005

48.3.3.1 Bootstrap loaders

One of the supported ways to start user code after SYS_FW completion is the Bootstrap Loader (BSL) mode.

If selected by user-configuration in UCB, this mode:

- Allows for use of either the CAN or ASC communication interface for downloading code or data into the Program Scratchpad RAM (PSPR) of the HOST boot core (CPU0)

  ***Note**:* *The BSL feature is not supported in SECURE cluster, CPUcs.*

- Start execution of the downloaded code from the first PSPR location after SYS_FW completion

  Attention**:* ***BSL mode is intended for initial programming and/or configuration of the device being initially in delivery state, and not for usage in normal operational mode of the device as part of ECU, and particularly not in case the security functionality is utilized.

Related information

Boot mode evaluation on page 8784

48.3.3.1.1 ASC bootstrap loader

The ASC bootloading routine implements the following steps:

- Configure RxD and TxD pins
- Wait to receive a zero byte (8 data and 1 stop bit)
- Calculate baudrate based on the zero byte duration
- Send acknowledge byte $D5_H$ to the host, indicating the device is ready to accept data transfer
- Enter a loop to receive 128 bytes which are stored as 32 words in CPU0 Program Scratchpad RAM (PSPR), starting from the first address location
- Once 128 bytes are received, user code execution is started from the first CPU0_PSPR address location

  In the intended use-case for initial device programming (refer to the previous Section), the Bootstrap Loader supports the following ASC communication speeds:

  - 28000 Baud
  - 38400 Baud
  - 57600 Baud
  - 115200 Baud
  - 300000 Baud

48.3.3.1.2 CAN bootstrap loader

The CAN bootstrap loader (CAN BSL) transfers program code or data from an external host through a CAN interface into CPU0 Program Scratchpad RAM. It is a primary bootloader, and should not be confused with the secondary bootloader which can be purchased from tool partners.

The main characteristics of the AURIX™ TC4xx CAN Bootstrap Loader are:

- Classical CAN is supported
- The initial identifier is fixed to 555_H
- The number of data frames to be received by BSL is programmable

The communication between the TC4xx and an external host is based on the following three CAN standard frames:

- Initialization frame(s)
  - Sent by the external host to TC4xx
- Acknowledgment frame(s)
  - Sent by the TC4xx to the external host
- Data frame(s)
  - Sent by the external host to TC4xx

Related information

CAN BSL flow on page 8792

Back-up clock used for CANXL

This is the clock system status after device system and power-on reset during CAN BSL execution.

This is the case in the delivery state of the device, when the back-up clock is used as clock-source for the CANXL module during CAN BSL, after performing the following configurations:

• fBACK/5 is selected as clock source for fCANXLI

• fCANXLI is selected as clock source fCANXL

The CANXL module operates with a 20 MHz clock for the asynchronous part as provided by the TC4xx internal back-up clock source.

The supported features are:

- Classical CAN
- Sample point of 60%
- Baudrates:
  - 100 KBit/s
  - 125 KBit/s
  - 250 KBit/s
  - 500 KBit/s

CAN BSL flow

The CAN bootstrap loader flow, shown in the figure below, consists of 3 phases. There is a phase 1 is initialization and knowledge, followed by a phase 2 initialization and knowledge, and then a data loading phase.

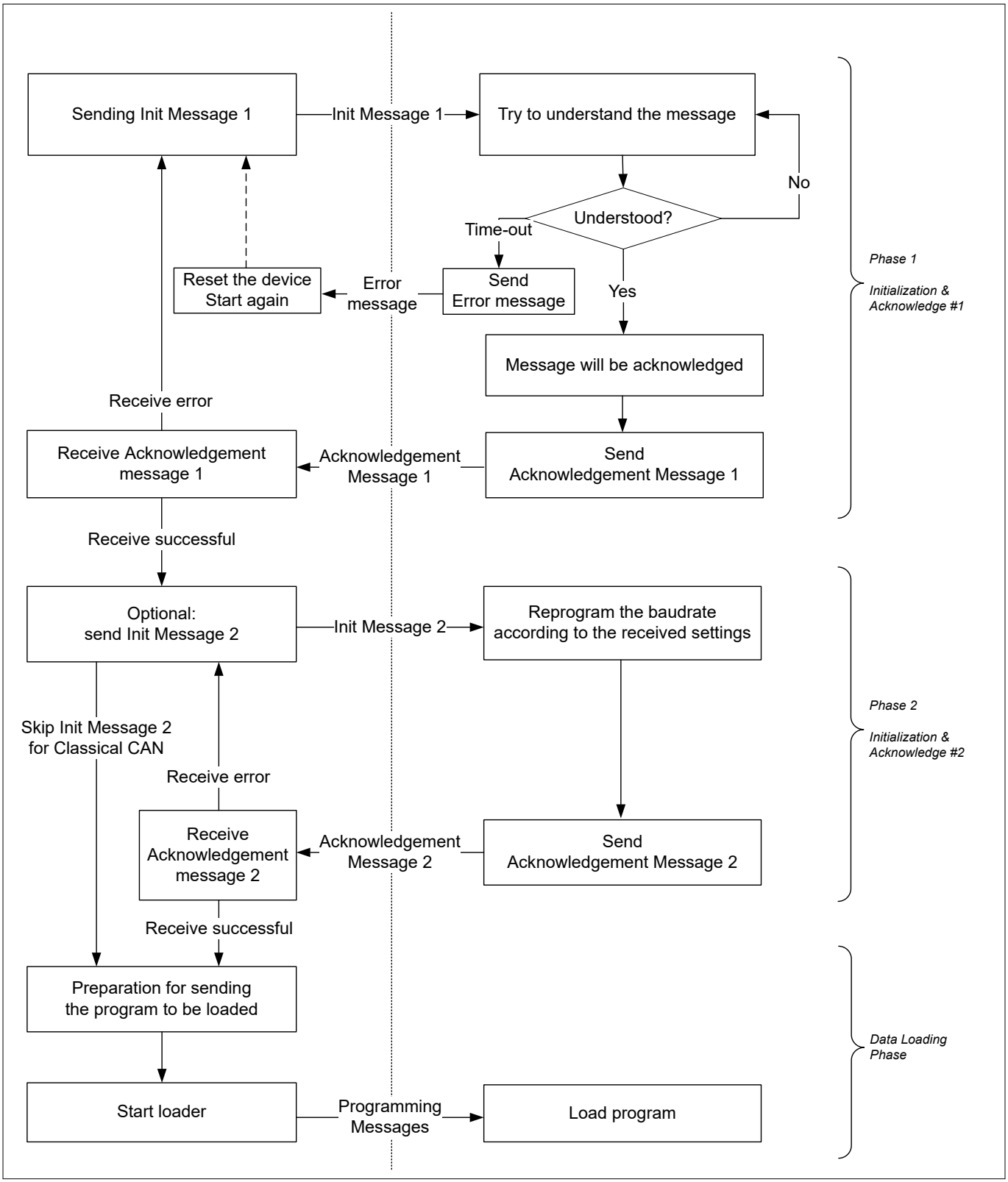


Figure 1258 CAN BSL flow

Key to abbreviations:

- ACKID
  - Acknowledgment Message shall have this 11-bit Identifier (like in register)
- DMSGC (data messages count)
  - Amount of messages which will be sent including the program
- DMSGID (data messages identifier)
  - The data frames will have this 11-bit ID (like in register)
- TSEG1 and TSEG2
  - Baudrate as used (copied from register)
- NBTP
  - Nominal bit timing and pre-scaler register settings (Classical CAN has arbitration segment only)
- DBTP
  - Data bit timing and pre-scaler register settings (Fast Segment)
- DBPM
  - Data Bytes per Message (DLC= [9, 11] is not allowed)

Phase 1 initialization and acknowledge (#1)

During this phase, CAN BSL tries to determine the CAN baud rate at which the external host is communicating and to understand (receive correctly) the initialization message (classical CAN message) sent by the host.

This requires that the external host sends initialization message 1 continuously to the AURIX™ TC4xx in classical CAN format, as shown in the following table:

Table 2517 Initialization message 1

ID	DLC	DB0	DB1	DB2	DB3	DB4	DB5	DB6	DB7
Requested frame format: Classical CAN									
555_H	8	55_H	55_H	ACKID Low	ACKID High	DMSGC Low	DMSGC High	DMSGID Low	DMSGID High

CAN BSL evaluates the incoming messages trying to identify content according to the table above, and according to the result:

- If the message is understood then the CAN BSL:
  - Configures the CANXL module for the selected baudrate settings
  - Acknowledges the next received initialization message 1 Tx ACK bit field as dominant
  - Sends acknowledgment message 1 with the format shown in the acknowledgment table below. This message also informs the host whether the reception was error-free or not
- If initialization messages can not be understood for any supported baudrate settings then:
  - CAN BSL sends error message with the format shown the table below: Error message (No baudrate detected)
  - CAN BSL terminates - the device should be re-started with a reset

Table 2518 Acknowledgment message 1

ID	DLC	DB0	DB1	DB2	DB3	DB4	DB5	DB6	DB7
Frame: Message received error free									
ACKID	8	ACKID Low	ACKID High	DMSGC Low	DMSGC High	DMSGID Low	DMSGID High	TSEG1	TSEG2
Frame: Faulty DLC									
ACKID if not received AA_H	as received	AA_H	AA_H	AA_H	AA_H	AA_H	AA_H	AA_H	AA_H

Table 2519 Error message (No baudrate detected)

Frame	ID	DLC
I do not understand	AAA_H	0

Phase 2 initialization and acknowledge (#2)

Once the acknowledgment message 1 from CAN BSL is successfully received, the host can send initialization message 2 with the format shown in the table below.

This message is optional allowing the host to provide a more precise baudrate for further classical CAN operation. In such a case, a jump in baudrate in any direction for the arbitration segment is not allowed:

Table 2520 Initialization message 2

ID	DLC	DB0	DB1	DB2	DB3	DB4	DB5	DB6	DB7
Frame format requested: Classical CAN									
555_H	4	NBTP. NTSEG2	NBTP. NTSEG1	NBTP. [23..16]	NBTP. [31..24]				

If after sending acknowledgment message 1 that initialization message 2 is received but not programming message (differentiation done on the message ID), the CAN BSL analyzes this message against the format shown in the table below, and then:

- If classical CAN has been requested (DLC=4):
  - Install the (more precise) baudrate settings (NBTP register) as received
  - Send acknowledgment message 2 indicating 'error free' (refer to the table below)
- If the message has been wrongly received (DLC other than 4):
  - Send acknowledgment message 2 indicating 'Faulty DLC' (refer to the table below)

Acknowledgment message 2 sent by CAN BSL and indicating 'Error free', tells the external host that the device is now ready to receive data frames.

Table 2521 Acknowledgment message 2

ID	DLC	DB0	DB1	DB2	DB3	DB4	DB5	DB6	DB7
Frame: Message received error free									
ACKID	4	NBTP. NTSEG2	NBTP. NTSEG1	NBTP. [23..16]	NBTP. [31..24]				

(table continues...)

Table 2521 (continued) Acknowledgment message 2

ID	DLC	DB0	DB1	DB2	DB3	DB4	DB5	DB6	DB7
Frame: Faulty DLC									
ACKID	as received	AA_H	AA_H	AA_H	AA_H	AA_H	AA_H	AA_H	AA_H

Data loading phase

After receiving acknowledgment message 2 from CAN BSL, or if Phase 2 (initialization and acknowledge) has been skipped, the host starts sending programming messages with the format shown in the table below.

CAN BSL receives the data and stores it in CPU0 Program Scratchpad RAM (PSPR) starting from address $C0000000_H$.

Both communication partners evaluate the data message count until the requested number of CAN data frames has been transmitted.

After the reception of the last CAN data frame, CAN BSL terminates and the user code will be started from address $C0000000_H$ in CPU0 PSPR.

Table 2522 Programming message

DLC	DB0	DB1	DB2	DB3	DB4	DB5	DB6	DB7
Frame format requested: Classical CAN								
8	Hex Code Byte	Hex Code Byte	Hex Code Byte	Hex Code Byte	Hex Code Byte	Hex Code Byte	Hex Code Byte	Hex Code Byte

48.3.4 Firmware exit information

After successful completion of the device boot, information is indicated to the application software in System Mode Management (SMM) registers. A summary of that information is provided here - for register layout information, refer to SMM Chapter.

- STMEM1 register indicates the boot status:
  - Which BMHD is valid and taken
  - The start-up mode selected
  - Whether and which SOTA configuration is installed
  - Whether CSRM start was successful
  - Either a QM part of SYS_FW has been executed or not. The device start-up is safe only if no QM part entered
- STMEM2 register contains the address of the first user instruction taken after Firmware
- RSSTATBFW indicates the reset type being processed by Firmware

If Boot Error State is entered during device start-up, the error code can be read-out from COMDATA register through the debug interface.

For the description of COMDATA register in regard to hardware-related attributes (address, access conditions) - refer to the DEBUG chapter in the OCDS documentation.

The bit field description in case the register is used to indicate a boot error is shown in the below table.

Table 2523 COMDATA boot error bit field description

Field	Bits	Description
TYPE	31:30	Error Type: 00b - no error occured 01b - runtime 10b - development 11b - transient
FEATURE_ID	29:24	Feature causing the error: 0x00 - Generic error (e.g. unexpected trap) 0x01 - MCU preparation 0x02 - NVM ramp-up phase A - analog block ramp 0x03 - NVM ramp-up phase B - NVM Bank ramp 0x04 - NVM ramp-up - register recovery phase 0x05 - BMHD evaluation 0x06 - SOTA support handling 0x08 - NVM ramp-up - microcode recovery 0x0F - Debug handling 0x15 - SRAM initialization 0x16 - NVM ramp-up - configure DMU else - contact Infineon representative/FAE for further information
ERROR_ID	7:0	ID of the error - contact Infineon representative/FAE for further information
0	23:8	Reserved

Related information

System Mode Management (SMM) on page 6315
Registers overview - SMM (ascending offset address) on page 6368

48.3.5 Power-related features

In addition to device configuration, user code start and development support, another Firmware aim is to 'flatten' power-consumption fluctuations during run-time slots that are under Firmware control.

Several Firmware features are available to help reduce these fluctuations, but this section focuses on:

- Limiting load jumps during start-up
- Shutdown request handlers

Related information

Power-friendly debug monitor routines on page 9006

48.3.5.1 Limiting load jumps during start-up

This limiting measure is applied for events other than application reset. Application reset is intended not to change the major device or system settings such as clock configuration, but just to re-start application software execution

After system and power-on resets, firmware takes care for power-consumption increase steps to be within the specified or allowed range. The measures foreseen include:

- Clock increase in steps, if needed, according to the consumption of the modules supposed to be active at any phase
- SRAMs initialization in groups, and not all selected at once. Grouping is adapted to the consumption and work frequency of the target memories

After application reset, the Ramp Oscillator is used to increase the system frequency in small steps.

Related information

Power-friendly debug monitor routines on page 9006

48.3.5.2 Shutdown request handlers

Routines with this purpose are integrated as parts of SYS_FW, CSRM_FW, and PPU_FW, and they are all automatically called, by hardware, after any warm reset request received or processed by the Reset Controller. The Reset Controller is part of the System Mode Management unit (SMM).

The idea behind is the same as the motivation for implementing 'Power-friendly debug monitor routines'. As the user code execution before the reset may have been causing some power-consuming activity, just stopping this activity with a reset will cause a strong drop-down in power consumption, and this may lead to big and undesired fluctuations on the power-supply lines.

These routines are different for CPUs from the HOST cluster (CPU0…n), the SECURE cluster (CPUcs), and the PPU, but the purpose is the same:

- Once entered, any of the routines starts execution of a code loop with “mediate power consumption”
- This loop is executed:
  - For CPUx (x=0…n) and CPUcs, until a dedicated to any CPUx time-out is elapsed, indicating the “shutdown state” is reached for the respective target CPU, then that CPU is put by firmware into IDLE state. This reduces power-consumption for the respective sub-system to a minimum
  - For PPU, until the reset is effectively activated (by SMM)

Additionally, after an application reset request, the Ramp Oscillator is used to decrease the system frequency in small steps.

As result from the above measures, the power-consumption of the device after reset-request activation is consequently reduced without big drops until the reset is effectively triggered. Thus the customer system power supply gets no sudden load-drops - which drops could cause supply voltage to jump outside of the specified range.

Related information

Registers overview - SMM (ascending offset address) on page 6368
Power-friendly debug monitor routines on page 9006

48.3.6 Stand-by controller firmware

The stand-by controller firmware (SCR_FW) features are described in the start-up procedure and basic debugging capability topics that follow.

Related information

Start-up procedure on page 8798
Basic debugging capability on page 8798

48.3.6.1 Start-up procedure

This procedure initializes the SCR to a known state after reset to ensure stable operation, before branching to various boot modes.

Which boot mode is taken depends on the SCR configuration (SCRCFG) value provided in the PMS_SCR_CON0 register. The SCRCFG value and corresponding boot mode are detailed in the following table:

Table 2524 SCR boot modes and SCRCFG values

SCRCFG value	SCR boot mode
000_B	User mode. Program execution started from XRAM (default value after reset)
001_B	OCDS mode, SoC debug interface (DAP or SPD, controlled from SoC)
010_B	OCDS mode, SCR DAP interface, DAP0_0 and DAP1_0 pins selected
011_B	OCDS mode, SCR DAP interface, DAP0_1 and DAP1_1 pins selected
100_B	OCDS mode, SCR SPD interface, SPD_0 pin selected
101_B	OCDS mode, SCR SPD interface, SPD_1 pin selected
else	Reserved

Acronyms used in the table:

- DAP
  - Device access port
- SPD
  - Single pin DAP

When the SCR is booted in User mode, the SCR_FW checks whether the content at SCR XRAM addresses $7FF8_H$ through $7FFF_H$ matches the pattern 55_H, AA_H, 55_H, AA_H, 55_H, AA_H, 55_H, AA_H which is used as condition that valid code is available in SCR XRAM:

- if the pattern is found
  - SCR_FW indicates OK with setting SCR_SCRINTEXCHG register to 80_H
  - SCR_FW starts the user software from SCR XRAM address 0_H
- if the pattern not found
  - SCR_FW indicates FAIL with setting SCR_SCRINTEXCHG register to 81_H
  - SCR_FW enters an endless loop until the next SCR reset

This means that before the SCR is released from reset, the SCR user software must be downloaded into SCR XRAM, where:

- The first instruction or jump table must be located at address 0_H
- At addresses $7FF8_H$ through to $7FFF_H$ the pattern 55_H, AA_H, 55_H, AA_H, 55_H, AA_H, 55_H, AA_H must be stored - it is used by SCR_FW as check-condition that valid code is available in SCR XRAM

48.3.6.2 Basic debugging capability

This FW part in combination with the hardware of On-Chip Debug Support (OCDS) system of SCR provides basic debugging capability to the user, when the device is started in OCDS mode (as of the above Table), refer to "Debug Information" Section in SCR Chapter of this Manual for further information.

Related information

Start-up procedure on page 8798

48.4 Registers

48.4.1 Configuration area overview

An overview of the complete configuration data area in DFLASH0 and DFLASH1 is shown in the following table.

Name	Content	Start address	End address	Read condition	Program and user commands condition	Erase condition
FCB0	Reserved.	$AE600000_H$	$AE60A7FF_H$			
ICB0	Reserved.	$AE60A800_H$	$AE60DFFF_H$			
FCBR	Reserved.	$AE60E000_H$	$AE613FFF_H$			
FCB1	Reserved.	$AEE00000_H$	$AEE0A7FF_H$			
ICB1	Reserved.	$AEE0A800_H$	$AEE0AFFF_H$			
UCB0_00	UCB_RTC_MOID Device configuration RTC, MO-specific, user readable. Detailed layout in Table 2526	$AE400000_H$	$AE4007FF_H$	Y	N	N
UCB0_01	Reserved	$AE400800_H$	$AE400FFF_H$			
UCB0_02	Reserved	$AE401000_H$	$AE4017FF_H$			
UCB0_03	Reserved	$AE401800_H$	$AE401FFF_H$			
UCB0_04	Reserved	$AE402000_H$	$AE4027FF_H$			
UCB0_05	Reserved	$AE402800_H$	$AE402FFF_H$			
UCB0_06	Reserved	$AE403000_H$	$AE4037FF_H$			
UCB0_07	Reserved	$AE403800_H$	$AE403FFF_H$			
UCB0_08	Reserved	$AE404000_H$	$AE4047FF_H$			
UCB0_09	UCB_RTC_BMHD0 Detailed layout in Table 2527	$AE404800_H$	$AE404FFF_H$	Y but PWD	UNL/PWD	UNL/PWD
UCB0_10	UCB_RTC_BMHD1 Detailed layout in Table 2528	$AE405000_H$	$AE4057FF_H$	Y but PWD	UNL/PWD	UNL/PWD
UCB0_11	UCB_RTC_BMHD2 Detailed layout in Table 2529	$AE405800_H$	$AE405FFF_H$	Y but PWD	UNL/PWD	UNL/PWD
UCB0_12	UCB_RTC_BMHD3 Detailed layout in Table 2530	$AE406000_H$	$AE4067FF_H$	Y but PWD	UNL/PWD	UNL/PWD
UCB0_13	Reserved	$AE406800_H$	$AE406FFF_H$			
UCB0_14	Reserved	$AE407000_H$	$AE4077FF_H$			
UCB0_15	UCB_USER_DATA0 - User data RTC storage 0 Detailed layout in Table 2531	$AE407800_H$	$AE407FFF_H$	Y but PWD	UNL/PWD	UNL/PWD

Name	Content	Start address	End address	Read condition	Program and user commands condition	Erase condition
UCB0_16	UCB_USER_DATA1 - User data RTC storage 1 Detailed layout in Table 2532	$AE408000_H$	$AE4087FF_H$	Y but PWD	UNL/PWD	UNL/PWD
UCB0_17	UCB_RTC_USERCFG_ORIG - Device configuration RTC, user selections, ORIG Detailed layout in Table 2533	$AE408800_H$	$AE408FFF_H$	Y but PWD	UNL/PWD	UNL/PWD
UCB0_18	UCB_RTC_USERCFG_COPY - Device configuration RTC, user selections, COPY Detailed layout as in Table 2533 but from the start address shown on the right	$AE409000_H$	$AE4097FF_H$	Y but PWD	UNL/PWD	UNL/PWD
UCB0_19	UCB_RTC_SWAP_ORIG - SWAP configuration RTC, user selections, ORIG Detailed layout in Table 2534	$AE409800_H$	$AE409FFF_H$	Y but PWD	UNL/PWD	UNL/PWD
UCB0_20	UCB_RTC_SWAP_COPY - SWAP configuration RTC, user selections, COPY Detailed layout as in Table 2534 but from the start address shown on the right	$AE40A000_H$	$AE40A7FF_H$	Y but PWD	UNL/PWD	UNL/PWD
UCB0_21	UCB_RTC_FLASH_ORIG Detailed layout in Table 2535	$AE40A800_H$	$AE40AFFF_H$	Y but PWD	UNL/PWD	UNL/PWD
UCB0_22	UCB_RTC_FLASH_COPY Detailed layout as in Table 2535 but from the start address shown on the right	$AE40B000_H$	$AE40B7FF_H$	Y but PWD	UNL/PWD	UNL/PWD
UCB0_23	UCB_RTC_ECPRIO_ORIG Detailed layout in Table 2536	$AE40B800_H$	$AE40BFFF_H$	Y but PWD	UNL/PWD	UNL/PWD
UCB0_24	UCB_RTC_ECPRIO_COPY Detailed layout as in Table 2536 but from the start address shown on the right	$AE40C000_H$	$AE40C7FF_H$	Y but PWD	UNL/PWD	UNL/PWD
UCB0_25	UCB_RTC_PFLASH_OTP0_ORIG Detailed layout in	$AE40C800_H$	$AE40CFFF_H$	Y	if set UNL	if all set UNL
UCB0_26	UCB_RTC_PFLASH_OTP1_ORIG Detailed layout in	$AE40D000_H$	$AE40D7FF_H$	Y	if set UNL	if all set UNL

Name	Content	Start address	End address	Read condition	Program and user commands condition	Erase condition
UCB0_27	UCB_RTC_PFLASH_OTP0_COPY Detailed layout as in but from the start address shown on the right	$AE40D800_H$	$AE40DFFF_H$	Y	if set UNL	if all set UNL
UCB0_28	UCB_RTC_PFLASH_OTP1_COPY Detailed layout as in but from the start address shown on the right	$AE40E000_H$	$AE40E7FF_H$	Y	if set UNL	if all set UNL
UCB0_29	Reserved	$AE40E800_H$	$AE40EFFF_H$			
UCB0_30	Reserved	$AE40F000_H$	$AE40F7FF_H$			
UCB0_31	Reserved	$AE40F800_H$	$AE40FFFF_H$			
UCB0_32	Reserved	$AE410000_H$	$AE4107FF_H$			
UCB0_33	Reserved	$AE410800_H$	$AE410FFF_H$			
UCB0_34	Reserved	$AE411000_H$	$AE4117FF_H$			
UCB0_35	UCB_TEST Detailed layout in Table 2539	$AE411800_H$	$AE411FFF_H$	Y	N	N
UCB0_36	UCB_INFO Detailed layout in Table 2540	$AE412000_H$	$AE4127FF_H$	Y	N	N
UCB0_37	UCB_RTCCFG - Device configuration and trimming - RTC, non-MO-specific Detailed layout in Table 2541	$AE412800_H$	$AE412FFF_H$	Y	N	N
UCB0_38	UCB_USER - User configuration values, not handled by firmware and not installed during device start-up Detailed layout in Table 2542	$AE413000_H$	$AE4137FF_H$	Y	N	N
UCB0_39	Reserved	$AE413800_H$	$AE413FFF_H$			
UCB1_00	Reserved	$AEC00000_H$	$AEC007FF_H$			
UCB1_01	UCB_CS_DBG_ORIG Detailed layout in Table 2544	$AEC00800_H$	$AEC00FFF_H$	Y but PWD	EDDM	EDDM
UCB1_02	UCB_CS_DBG_COPY Detailed layout as in Table 2544 but from the start address shown on the right	$AEC01000_H$	$AEC017FF_H$	Y but PWD	EDDM	EDDM

Name	Content	Start address	End address	Read condition	Program and user commands condition	Erase condition
UCB1_03	UCB_CS_BMHD0 Detailed layout in Table 2545	$AEC01800_H$	$AEC01FFF_H$	Y but PWD	UNL/PWD	UNL/PWD
UCB1_04	UCB_CS_BMHD1 Detailed layout in Table 2546	$AEC02000_H$	$AEC027FF_H$	Y but PWD	UNL/PWD	UNL/PWD
UCB1_05	UCB_CS_BMHD2 Detailed layout in Table 2547	$AEC02800_H$	$AEC02FFF_H$	Y but PWD	UNL/PWD	UNL/PWD
UCB1_06	UCB_CS_BMHD3 Detailed layout in Table 2548	$AEC03000_H$	$AEC037FF_H$	Y but PWD	UNL/PWD	UNL/PWD
UCB1_07	UCB_CS_PFLASH_OTP0_ORIG Detailed layout in Table 2549	$AEC03800_H$	$AEC03FFF_H$	Y		
UCB1_08	UCB_CS_ECPRIO_ORIG Detailed layout in Table 2550	$AEC04000_H$	$AEC047FF_H$	Y but PWD	UNL/PWD	UNL/PWD
UCB1_09	UCB_CS_PFLASH_OTP0_COPY Detailed layout as in Table 2549 but from the start address shown on the right	$AEC04800_H$	$AEC04FFF_H$	Y		
UCB1_10	UCB_CS_ECPRIO_COPY Detailed layout as in Table 2550 but from the start address shown on the right	$AEC05000_H$	$AEC057FF_H$	Y but PWD	UNL/PWD	UNL/PWD
UCB1_11	UCB_CS_FLASH_ORIG Detailed layout in Table 2551	$AEC05800_H$	$AEC05FFF_H$	Y but PWD	UNL/PWD	UNL/PWD
UCB1_12	UCB_CS_SWAP_ORIG - SWAP configuration CSRM, user selections, ORIG Detailed layout in Table 2552	$AEC06000_H$	$AEC067FF_H$	Y but PWD	UNL/PWD	UNL/PWD
UCB1_13	UCB_CS_SWAP_COPY - SWAP configuration CSRM, user selections, COPY Detailed layout as in Table 2552 but from the start address shown on the right	$AEC06800_H$	$AEC06FFF_H$	Y but PWD	UNL/PWD	UNL/PWD
UCB1_14	UCB_CS_FLASH_COPY Detailed layout as in Table 2551 but from the start address shown on the right	$AEC07000_H$	$AEC077FF_H$	Y but PWD	UNL/PWD	UNL/PWD
UCB1_15	UCB_CS_USERCFG_ORIG - CSRM user configuration, ORIG Detailed layout in Table 2553	$AEC07800_H$	$AEC07FFF_H$	Y but PWD	UNL/PWD	UNL/PWD

Name	Content	Start address	End address	Read condition	Program and user commands condition	Erase condition
UCB1_16	UCB_CS_USERCFG_COPY - CSRM user configuration, COPY Detailed layout as in Table 2553 but from the start address shown on the right	$AEC08000_H$	$AEC087FF_H$	Y but PWD	UNL/PWD	UNL/PWD
UCB1_17	Reserved	$AEC08800_H$	$AEC08FFF_H$			
UCB1_18	Reserved	$AEC09000_H$	$AEC097FF_H$			
UCB1_19	Reserved	$AEC09800_H$	$AEC09FFF_H$			
UCB1_20	Reserved	$AEC0A000_H$	$AEC0A7FF_H$			
UCB1_21	Reserved	$AEC0A800_H$	$AEC0AFFF_H$			
UCB1_22	Reserved	$AEC0B000_H$	$AEC0B7FF_H$			
UCB1_23	Reserved	$AEC0B800_H$	$AEC0BFFF_H$			
UCB1_24	UCB_CS_USER_DATA0 - User data CS storage 0 Detailed layout in Table 2554	$AEC0C000_H$	$AEC0C7FF_H$	Y but PWD	UNL/PWD	UNL/PWD
UCB1_25	UCB_CS_USER_DATA1 - User data CS storage 1 Detailed layout in Table 2555	$AEC0C800_H$	$AEC0CFFF_H$	Y but PWD	UNL/PWD	UNL/PWD

Access conditions description:

- Y - always accessible
- N - not accessible
- Y but PWD - all content is readable, the PWD is only readable if UCB is UNLOCKED or if the correct password (PWD) was provided to disable protection command
- UNL/PWD - accessible if confirmation state is UNLOCKED or if the correct password (PWD) was provided to disable protection command
- if set UNL - accessible if this "set" is UNLOCKED
- if all set UNL - accessible if all "sets" are UNLOCKED
- EDDM - accessible if confirmation state is UNLOCKED or Destructive Debug Mode is NOT entered

***Note**: In the further descriptions, for the standard locations as:*

- *System address of this location (SAL)*
- *Length of this table*
- *Check result for this table (CRC)*
- *Password*
- *Confirmation Code and (optional) its copy*

only the locations are shown in the Register overview Tables, no detailed description as it is implicitly clear.

Refer to the above Configuration area overview to identify the presence of password for any part.

Reserved - meaning currently not used - locations are not shown, except some ones, which may become used in future - in these cases the potential target registers are shown.

***Note**: In the further detailed register descriptions, the shown (reset) values are in principle not corresponding to the locations' content after device boot – it depends on:*

- *Values stored by Infineon during chip production and by OEM/Tier1 during ECU production (if chip mounted)*
- *Values stored by the customer into user-programmable locations*

For more information – refer to "Start-up handling based on user-programmable data" Section.

48.4.2 User-relevance of the data in Configuration area

The below Table shows how the configuration data areas in DFLASH0 and DFLASH1 are handled in case of error found during boot, and at which phase of chip life-cycle the respective content is programmed.

Name	Content	Handling upon error?	Programming where?
FCB0	Reserved.	No boot	IFX
ICB0	Reserved.	No boot	IFX
FCBR	Reserved.	No boot	IFX
FCB1	Reserved.	No boot	IFX
ICB1	Reserved.	No boot	IFX
UCB0_00	UCB_RTC_MOID Device configuration RTC, MO-specific, user readable. Detailed layout in Table 2526	No boot	IFX
UCB0_01	Reserved	No boot	IFX
UCB0_02	Reserved	N.A.	N.A.
UCB0_03	Reserved	N.A.	N.A.
UCB0_04	Reserved	N.A.	N.A.
UCB0_05	Reserved	N.A.	N.A.
UCB0_06	Reserved	N.A.	N.A.
UCB0_07	Reserved	N.A.	N.A.
UCB0_08	Reserved	N.A.	N.A.

Name	Content	Handling upon error?	Programming where?
UCB0_09	UCB_RTC_BMHD0 Detailed layout in Table 2527	BML *	Plant
UCB0_10	UCB_RTC_BMHD1 Detailed layout in Table 2528	BML *	Plant
UCB0_11	UCB_RTC_BMHD2 Detailed layout in Table 2529	BML *	Plant
UCB0_12	UCB_RTC_BMHD3 Detailed layout in Table 2530	BML *	Plant
UCB0_13	Reserved	N.A.	N.A.
UCB0_14	Reserved	N.A.	N.A.
UCB0_15	UCB_USER_DATA0 - User data RTC storage 0 Detailed layout in Table 2531	N.A.	Plant
UCB0_16	UCB_USER_DATA1 - User data RTC storage 1 Detailed layout in Table 2532	N.A.	Plant
UCB0_17	UCB_RTC_USERCFG_ORIG - Device configuration RTC, user selections, ORIG Detailed layout in Table 2533	BML	Plant
UCB0_18	UCB_RTC_USERCFG_COPY - Device configuration RTC, user selections, COPY Detailed layout as in Table 2533 but from the start address shown on the right	BML	Plant
UCB0_19	UCB_RTC_SWAP_ORIG - SWAP configuration RTC, user selections, ORIG Detailed layout in Table 2534	N.A.	Field
UCB0_20	UCB_RTC_SWAP_COPY - SWAP configuration RTC, user selections, COPY Detailed layout as in Table 2534 but from the start address shown on the right	N.A.	Field
UCB0_21	UCB_RTC_FLASH_ORIG Detailed layout in Table 2535	BML	Plant
UCB0_22	UCB_RTC_FLASH_COPY Detailed layout as in Table 2535 but from the start address shown on the right	BML	Plant
UCB0_23	UCB_RTC_ECPRIO_ORIG Detailed layout in Table 2536	BML	Plant

Name	Content	Handling upon error?	Programming where?
UCB0_24	UCB_RTC_ECPRIO_COPY Detailed layout as in Table 2536 but from the start address shown on the right	BML	Plant
UCB0_25	UCB_RTC_PFLASH_OTP0_ORIG Detailed layout in	BML	Plant
UCB0_26	UCB_RTC_PFLASH_OTP1_ORIG Detailed layout in	BML	Plant
UCB0_27	UCB_RTC_PFLASH_OTP0_COPY Detailed layout as in but from the start address shown on the right	BML	Plant
UCB0_28	UCB_RTC_PFLASH_OTP1_COPY Detailed layout as in but from the start address shown on the right	BML	Plant
UCB0_29	Reserved	N.A.	N.A.
UCB0_30	Reserved	N.A.	N.A.
UCB0_31	Reserved	N.A.	N.A.
UCB0_32	Reserved	N.A.	N.A.
UCB0_33	Reserved	N.A.	N.A.
UCB0_34	Reserved	N.A.	N.A.
UCB0_35	UCB_TEST Detailed layout in Table 2539	N.A.	IFX
UCB0_36	UCB_INFO Detailed layout in Table 2540	N.A.	IFX
UCB0_37	UCB_RTCCFG - Device configuration and trimming - RTC, non-MO-specific Detailed layout in Table 2541	No boot	IFX
UCB0_38	UCB_USER - User configuration values, not handled by firmware and not installed during device start-up Detailed layout in Table 2542	N.A.	IFX
UCB0_39	Reserved	N.A.	N.A.
UCB1_00	Reserved	No boot	IFX
UCB1_01	UCB_CS_DBG_ORIG Detailed layout in Table 2544	No boot	Plant

Name	Content	Handling upon error?	Programming where?
UCB1_02	UCB_CS_DBG_COPY Detailed layout as in Table 2544 but from the start address shown on the right	No boot	Plant
UCB1_03	UCB_CS_BMHD0 Detailed layout in Table 2545	BML *	Plant
UCB1_04	UCB_CS_BMHD1 Detailed layout in Table 2546	BML *	Plant
UCB1_05	UCB_CS_BMHD2 Detailed layout in Table 2547	BML *	Plant
UCB1_06	UCB_CS_BMHD3 Detailed layout in Table 2548	BML *	Plant
UCB1_07	UCB_CS_PFLASH_OTP0_ORIG Detailed layout in Table 2549	BML	Plant
UCB1_08	UCB_CS_ECPRIO_ORIG Detailed layout in Table 2550	BML	Plant
UCB1_09	UCB_CS_PFLASH_OTP0_COPY Detailed layout as in Table 2549 but from the start address shown on the right	BML	Plant
UCB1_10	UCB_CS_ECPRIO_COPY Detailed layout as in Table 2550 but from the start address shown on the right	BML	Plant
UCB1_11	UCB_CS_FLASH_ORIG Detailed layout in Table 2551	BML	Plant
UCB1_12	UCB_CS_SWAP_ORIG - SWAP configuration CSRM, user selections, ORIG Detailed layout in Table 2552	N.A.	Field
UCB1_13	UCB_CS_SWAP_COPY - SWAP configuration CSRM, user selections, COPY Detailed layout as in Table 2552 but from the start address shown on the right	N.A.	Field
UCB1_14	UCB_CS_FLASH_COPY Detailed layout as in Table 2551 but from the start address shown on the right	BML	Plant

Name	Content	Handling upon error?	Programming where?
UCB1_15	UCB_CS_USERCFG_ORIG - CSRM user configuration, ORIG Detailed layout in Table 2553	BML	Plant
UCB1_16	UCB_CS_USERCFG_COPY - CSRM user configuration, COPY Detailed layout as in Table 2553 but from the start address shown on the right	BML	Plant
UCB1_17	Reserved	N.A.	IFX
UCB1_18	Reserved	No boot	IFX
UCB1_19	Reserved	N.A.	N.A.
UCB1_20	Reserved	N.A.	N.A.
UCB1_21	Reserved	N.A.	N.A.
UCB1_22	Reserved	N.A.	N.A.
UCB1_23	Reserved	N.A.	N.A.
UCB1_24	UCB_CS_USER_DATA0 - User data CS storage 0 Detailed layout in Table 2554	N.A.	Plant
UCB1_25	UCB_CS_USER_DATA1 - User data CS storage 1 Detailed layout in Table 2555	N.A.	Plant

Handling upon error in content:

- No boot - The device does not boot
- BML - Fallback entered if BML not active
- BML * - Fallback entered if BML not active and all BMHDs failed evaluation
- N.A. - Does not affect the device boot

Programming:

- IFX - During production of the chip
- Plant - During production of the ECU,acc. to customer application requirements **
- Field - On the field, by the customer application **
- N.A. - Not used - left unprogrammed

** The differentiation between “plant” and “field” programming is just a proposal - both are under customer control and can be done wherever appropriate for the target application

48.4.3 Register address space - UCB0

Table 2525 Registers address space - UCB0

Module	Base address	End address	Note
UCB0	$AE400000_H$	$AE413FFF_H$	UCB0 Range

48.4.3.1 Register overview - UCB0_00 (ascending offset address)

Table 2526 Register overview - UCB0_00 (ascending offset address)

Short name	Long name	Offset address	See
UCB0_00_SAL	System Address of this Location	0000_H	
UCB0_00_LENGTH1	Length of this table	0004_H	
CHIPID	CHIPID	0008_H	8809
PRODUCT_NAME0	Product Full Name, ASCII-coded text, Word0	$000C_H$	8812
PRODUCT_NAME1	Product Full Name, ASCII-coded text, Word1	0010_H	8812
PRODUCT_NAME2	Product Full Name, ASCII-coded text, Word2	0014_H	8813
PRODUCT_NAME3	Product Full Name, ASCII-coded text, Word3	0018_H	8814
DEVCFG_MO	DEVCFG_MO version, coded in format Va.Vb.Vc.Vx.Vy	$001C_H$	8815
MO_VER	MO list Excel version, coded in format Vx.Vy	0020_H	8816
CRC_RTCMOID1	Check Result for this table	0024_H	
UCB0_00_CONFIRMATION	Confirmation CODE ORIGINAL, 32-bit	$07F0_H$	
UCB0_00_CONFIRMATION_COPY	Confirmation CODE COPY, 32-bit	$07F8_H$	

48.4.3.1.1 CHIPID

CHIPID Offset address: 0008_H

CHIPID value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TEMP		PROD				PCK				CORES			FEATURE		
r		r				r				r			r		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FEATURE		MEMSIZE			ACC		SEC		FREQ			PINOUT			
r		r			r		r		r			r			

Field	Bits	Type	Description
PINOUT	3:0	r	**Package Type (Material/Type/Pinout)** 0x0 = "CC" Cu Plane BGA COM Pinout 0x1 = "CS" Cu Plane BGA STD Pinout 0x2 = "MC" Molded BGA COM Pinout 0x3 = "MS" Molded BGA STD Pinout 0x4 = "LC" Lid BGA COM Pinout 0x5 = "LS" Lid BGA STD Pinout 0x6 = "ML" Molded LQFP 0x7 = "MT" Molded TQFP else - reserved
FREQ	6:4	r	**Max Frequency** 0b010 - 400 MHz 0b100 - 500 MHz else - reserved
SEC	8:7	r	**Security** 0b10 = "F" - Full CSRM 0b11 = "O" - Full CSRM + OSCCA else - reserved
ACC	10:9	r	**Accelerator** 0b00 - No PPU 0b01 - Entry PPU 128bits 0b10 - Mid PPU 256bits 0b11 - Full PPU 512bits
MEMSIZE	14:11	r	**PFLASH Memory Size** 0x2 - 3MB 0x3 - 4MB 0x4 - 5 MB 0x5 - 6MB 0x6 - 8 MB 0x8 - 12 MB 0x9 - 16 MB 0xA - 20MB 0xB - 24MB else - reserved

(table continues...)

(continued)

Field	Bits	Type	Description
FEATURE	18:15	r	**Feature Option** 0x0 ="M" - Customer specific 0x1 ="Z" - Zone architectures 0x2 ="P" - Standard 0x3 ="X" - Extended features 0x4 ="C" - Customer specific 0x5 ="Q" - Standard + CAN-XL 0x8 ="E" - Emulation device else - reserved
CORES	21:19	r	**Core Architecture** 0b001 - Single 0b010 - Dual 0b011 - Triple 0b100 - Quad 0b101 - Penta 0b110 - Hexa else - reserved
PCK	25:22	r	**Package Class** 0x2 - 100 pin 0x3 - 144 pin 0x4 - 176 pin 0x5 - 188 pin 0x6 - 224 pin 0x7 - 292 pin 0x8 - 324 pin 0x9 - 436 pin else - reserved
PROD	29:26	r	**Product ID** 0b0001 - TC41x 0b0010 - TC42x 0b0011 - TC43x 0b0100 - TC44x 0b0101 - TC45x 0b0110 - TC46x 0b1000 - TC48x 0b1001 - TC49x 0b1010 - TC4Px 0b1011 - TC4Zx 0b1100 - TC4Rx 0b1101 - TC4Dx 0b1110 - TC4Ex else - reserved

(table continues...)

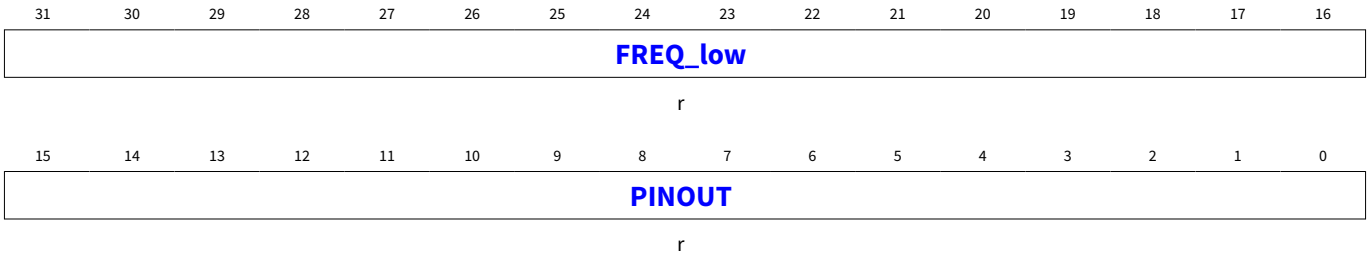
(continued)

Field	Bits	Type	Description
TEMP	31:30	r	**Temperature Range** 0b01 -40..+125degC 0b10 -40..+150degC 0b11 -40..+105degC else - reserved

48.4.3.1.2 Product Full Name, ASCII-coded text, Word0

PRODUCT_NAME0 Offset address: $000C_H$

Product Full Name, ASCII-coded text, Word0 value: $0000\ 0000_H$



Field	Bits	Type	Description
PINOUT	15:0	r	**Package Type (Material/Type/Pinout)** CC - Cu Plane BGA COM Pinout CS - Cu Plane BGA STD Pinout MC - Molded BGA COM Pinout MS - Molded BGA STD Pinout LC - Lid BGA COM Pinout LS - Lid BGA STD Pinout ML - Molded LQFP MT - Molded TQFP
FREQ_low	31:16	r	**Max Frequency** (for the highest Byte - refer to PRODUCT_NAME1) 400 - 400 MHz 500 - 500 MHz

48.4.3.1.3 Product Full Name, ASCII-coded text, Word1

PRODUCT_NAME1 Offset address: 0010_H

Product Full Name, ASCII-coded text, Word1 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MEMSIZE_low								ACC							
r								r							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SEC								FREQ_high							
r								r							

Field	Bits	Type	Description
FREQ_high	7:0	r	**Max Frequency** (for the lowest 2 Bytes - refer to PRODUCT_NAME0) 400 - 400 MHz 500 - 500 MHz
SEC	15:8	r	**Security** F - Full CSRM O - Full CSRM + OSCCA
ACC	23:16	r	**Accelerator** N - No PPU E - Entry PPU 128bits M - Mid PPU 256bits F - Full PPU 512bits
MEMSIZE_low	31:24	r	**PFLASH Memory Size** (for the higher Byte - refer to PRODUCT_NAME2) 03 - 3MB 04 - 4MB 05 - 5 MB 06 - 6MB 08 - 8 MB 12 - 12 MB 16 - 16 MB 20 - 20MB 24 - 24MB

48.4.3.1.4 Product Full Name, ASCII-coded text, Word2

PRODUCT_NAME2 Offset address: 0014_H

Product Full Name, ASCII-coded text, Word2 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CORES								FEATURE							
r								r							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								MEMSIZE_high							
r								r							

Field	Bits	Type	Description
MEMSIZE_high	7:0	r	**PFLASH Memory Size** (for the lower Byte - refer to PRODUCT_NAME1) 03 - 3MB 04 - 4MB 05 - 5 MB 06 - 6MB 08 - 8 MB 12 - 12 MB 16 - 16 MB 20 - 20MB 24 - 24MB
RESERVED	15:8	r	**Read as 0**
FEATURE	23:16	r	**Feature Option** Z - Zone architectures P - Standard X - Extended features C - Customer specific M - Customer specific Q - Standard + CAN-XL E - Emulation device
CORES	31:24	r	**Core Architecture** L - Single D - Dual T - Triple Q - Quad P - Penta X - Hexa

48.4.3.1.5 Product Full Name, ASCII-coded text, Word3

PRODUCT_NAME3 Offset address: 0018_H

Product Full Name, ASCII-coded text, Word3 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TEMP								GEN							
r								r							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SERIES								PCK							
r								r							

Field	Bits	Type	Description
PCK	7:0	r	**Package Class** 0x2 - 100 pin 0x3 - 144 pin 0x4 - 176 pin 0x5 - 188 pin 0x6 - 224 pin 0x7 - 292 pin 0x8 - 324 pin 0x9 - 436 pin else - reserved
SERIES	15:8	r	**Device Series** 1 - TC41x 2 - TC42x 3 - TC43x 4 - TC44x 6 - TC46x 8 - TC48x 9 - TC49x P - TC4Px S - TC4Sx R - TC4Rx D - TC4Dx E - TC4Ex
GEN	23:16	r	**Device Generation** 4 - A3G
TEMP	31:24	r	**Temperature Range** K -40..+125degC L -40..+150degC A -40..+105degC

48.4.3.1.6 DEVCFG_MO version, coded in format Va.Vb.Vc.Vx.Vy

DEVCFG_MO Offset address: $001C_H$

DEVCFG_MO version, coded in format Va.Vb.Vc.Vx.Vy value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Va_high				Va_low				Vb				Vc			
r				r				r				r			

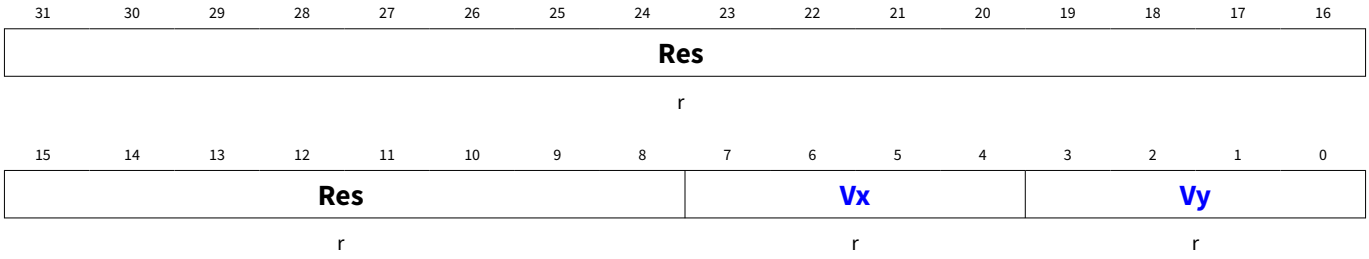
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Vx_high				Vx_low				Vy_high				Vy_low			
r				r				r				r			

Field	Bits	Type	Description
Vy_low	3:0	r	**Vy_low**
Vy_high	7:4	r	**Vy_high**
Vx_low	11:8	r	**Vx_low**
Vx_high	15:12	r	**Vx_high**
Vc	19:16	r	**Vc**
Vb	23:20	r	**Vb**
Va_low	27:24	r	**Va_low**
Va_high	31:28	r	**Va_high**

48.4.3.1.7 MO list Excel version, coded in format Vx.Vy

MO_VER Offset address: 0020_H

MO list Excel version, coded in format Vx.Vy value: $0000\ 0000_H$



Field	Bits	Type	Description
Vy	3:0	r	**Vy**
Vx	7:4	r	**Vx**

48.4.3.2 Register overview - UCB0_09 (ascending offset address)

Table 2527 Register overview - UCB0_09 (ascending offset address)

Short name	Long name	Offset address	See
UCB0_09_SAL	UCB_RTC_BMHD0 - System Address of this Location	04800_H	8817
UCB0_09_BMI_BMHDID	UCB_RTC_BMHD0 - Boot Mode Index and Boot Mode Header ID	04804_H	8817
UCB0_09_STAD	UCB_RTC_BMHD0 - Start address	04808_H	8818
UCB0_09_CRCBMHD	UCB_RTC_BMHD0 - Check Result for the BMI Header (offset 000H - 00BH)	$0480C_H$	8818
UCB0_09_PW0	UCB_BMHD0_PW - 256-bit password protection, PW0 (least significant)	$04FD0_H$	
UCB0_09_PW1	UCB_BMHD0_PW - 256-bit password protection, PW1	$04FD4_H$	
UCB0_09_PW2	UCB_BMHD0_PW - 256-bit password protection, PW2	$04FD8_H$	

(table continues...)

Table 2527 (continued) Register overview - UCB0_09 (ascending offset address)

Short name	Long name	Offset address	See
UCB0_09_PW3	UCB_BMHD0_PW - 256-bit password protection, PW3	$04FDC_H$	
UCB0_09_PW4	UCB_BMHD0_PW - 256-bit password protection, PW4	$04FE0_H$	
UCB0_09_PW5	UCB_BMHD0_PW - 256-bit password protection, PW5	$04FE4_H$	
UCB0_09_PW6	UCB_BMHD0_PW - 256-bit password protection, PW6	$04FE8_H$	
UCB0_09_PW7	UCB_BMHD0_PW - 256-bit password protection, PW7	$04FEC_H$	
UCB0_09_CONFIRMATION	UCB_BMHD0_ORIG_CODE - 32-bit CODE	$04FF0_H$	
UCB0_09_CONFIRMATION_COPY	UCB_BMHD0_COPY_CODE - 32-bit CODE	$04FF8_H$	

48.4.3.2.1 UCB_RTC_BMHD0 - System Address of this Location

UCB0_09_SAL Offset address: 04800_H

UCB_RTC_BMHD0 - System Address of this Location value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SAL															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SAL															
rh															

Field	Bits	Type	Description
SAL	31:0	rh	**System Address of this Location**

48.4.3.2.2 UCB_RTC_BMHD0 - Boot Mode Index and Boot Mode Header ID

UCB0_09_BMI_BMHDID Offset address: 04804_H

UCB_RTC_BMHD0 - Boot Mode Index and Boot Mode Header ID value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BMHDID															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BMI															
rh															

Field	Bits	Type	Description
BMI	15:0	rh	**Boot Mode Index**
BMHDID	31:16	rh	**Boot Mode Header ID** $B359_H$ **BMHDID**: Constant value for BMHD identification

48.4.3.2.3 UCB_RTC_BMHD0 - Start address

UCB0_09_STAD Offset address: 04808_H

UCB_RTC_BMHD0 - Start address value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
STAD															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STAD															
rh															

Field	Bits	Type	Description
STAD	31:0	rh	**Start Address** ABHMDx start address (in case BMI.MODE = ABM = 110_B). User Code start address (in case BMI.MODE = Flash start = 111_B).

48.4.3.2.4 UCB_RTC_BMHD0 - Check Result for the BMI Header (offset 000H - 00BH)

UCB0_09_CRCBMHD Offset address: $0480C_H$

UCB_RTC_BMHD0 - Check Result for the BMI Header [offset 000H - 00BH) value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CRCBMHD															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CRCBMHD															
rh															

Field	Bits	Type	Description
CRCBMHD	31:0	rh	**Check Result for Boot Mode Header** Check Result for the Boot Mode Header

48.4.3.3 Register overview - UCB0_10 (ascending offset address)

Table 2528 Register overview - UCB0_10 (ascending offset address)

Short name	Long name	Offset address	See
UCB0_10_SAL	UCB_RTC_BMHD1 - System Address of this Location	05000_H	
UCB0_10_BMI_BMHDID	UCB_RTC_BMHD1 - Boot Mode Index (BMI) and Boot Mode Header ID (CODE) = B359H	05004_H	
UCB0_10_STAD	UCB_RTC_BMHD1 - Start address	05008_H	
UCB0_10_CRCBMHD	UCB_RTC_BMHD1 - Check Result for the BMI Header (offset 000H - 007H)	$0500C_H$	
UCB0_10_PW0	UCB_BMHD1_PW - 256-bit password protection, PW0 (least significant)	$057D0_H$	
UCB0_10_PW1	UCB_BMHD1_PW - 256-bit password protection, PW1	$057D4_H$	
UCB0_10_PW2	UCB_BMHD1_PW - 256-bit password protection, PW2	$057D8_H$	
UCB0_10_PW3	UCB_BMHD1_PW - 256-bit password protection, PW3	$057DC_H$	
UCB0_10_PW4	UCB_BMHD1_PW - 256-bit password protection, PW4	$057E0_H$	
UCB0_10_PW5	UCB_BMHD1_PW - 256-bit password protection, PW5	$057E4_H$	
UCB0_10_PW6	UCB_BMHD1_PW - 256-bit password protection, PW6	$057E8_H$	
UCB0_10_PW7	UCB_BMHD1_PW - 256-bit password protection, PW7	$057EC_H$	
UCB0_10_CONFIRMATION	UCB_BMHD1_ORIG_CODE - 32-bit CODE	$057F0_H$	
UCB0_10_CONFIRMATION_COPY	UCB_BMHD1_COPY_CODE - 32-bit CODE	$057F8_H$	

48.4.3.4 Register overview - UCB0_11 (ascending offset address)

Table 2529 Register overview - UCB0_11 (ascending offset address)

Short name	Long name	Offset address	See
UCB0_11_SAL	UCB_RTC_BMHD2 - System Address of this Location	05800_H	
UCB0_11_BMI_BMHDID	UCB_RTC_BMHD2 - Boot Mode Index (BMI) and Boot Mode Header ID (CODE) = B359H	05804_H	
UCB0_11_STAD	UCB_RTC_BMHD2 - Start address	05808_H	
UCB0_11_CRCBMHD	UCB_RTC_BMHD2 - Check Result for the BMI Header (offset 000H - 007H)	$0580C_H$	
UCB0_11_PW0	UCB_BMHD2_PW - 256-bit password protection, PW0 (least significant)	$05FD0_H$	
UCB0_11_PW1	UCB_BMHD2_PW - 256-bit password protection, PW1	$05FD4_H$	
UCB0_11_PW2	UCB_BMHD2_PW - 256-bit password protection, PW2	$05FD8_H$	
UCB0_11_PW3	UCB_BMHD2_PW - 256-bit password protection, PW3	$05FDC_H$	
UCB0_11_PW4	UCB_BMHD2_PW - 256-bit password protection, PW4	$05FE0_H$	
UCB0_11_PW5	UCB_BMHD2_PW - 256-bit password protection, PW5	$05FE4_H$	

(table continues...)

Table 2529 (continued) Register overview - UCB0_11 (ascending offset address)

Short name	Long name	Offset address	See
UCB0_11_PW6	UCB_BMHD2_PW - 256-bit password protection, PW6	$05FE8_H$	
UCB0_11_PW7	UCB_BMHD2_PW - 256-bit password protection, PW7	$05FEC_H$	
UCB0_11_CONFIRMATION	UCB_BMHD2_ORIG_CODE - 32-bit CODE	$05FF0_H$	
UCB0_11_CONFIRMATION_COPY	UCB_BMHD2_COPY_CODE - 32-bit CODE	$05FF8_H$	

48.4.3.5 Register overview - UCB0_12 (ascending offset address)

Table 2530 Register overview - UCB0_12 (ascending offset address)

Short name	Long name	Offset address	See
UCB0_12_SAL	UCB_RTC_BMHD3 - System Address of this Location	06000_H	
UCB0_12_BMI_BMHDID	UCB_RTC_BMHD3 - Boot Mode Index (BMI) and Boot Mode Header ID (CODE) = B359H	06004_H	
UCB0_12_STAD	UCB_RTC_BMHD3 - Start address	06008_H	
UCB0_12_CRCBMHD	UCB_RTC_BMHD3 - Check Result for the BMI Header (offset 000H - 007H)	$0600C_H$	
UCB0_12_PW0	UCB_BMHD3_PW - 256-bit password protection, PW0 (least significant)	$067D0_H$	
UCB0_12_PW1	UCB_BMHD3_PW - 256-bit password protection, PW1	$067D4_H$	
UCB0_12_PW2	UCB_BMHD3_PW - 256-bit password protection, PW2	$067D8_H$	
UCB0_12_PW3	UCB_BMHD3_PW - 256-bit password protection, PW3	$067DC_H$	
UCB0_12_PW4	UCB_BMHD3_PW - 256-bit password protection, PW4	$067E0_H$	
UCB0_12_PW5	UCB_BMHD3_PW - 256-bit password protection, PW5	$067E4_H$	
UCB0_12_PW6	UCB_BMHD3_PW - 256-bit password protection, PW6	$067E8_H$	
UCB0_12_PW7	UCB_BMHD3_PW - 256-bit password protection, PW7	$067EC_H$	
UCB0_12_CONFIRMATION	UCB_BMHD3_ORIG_CODE - 32-bit CODE	$067F0_H$	
UCB0_12_CONFIRMATION_COPY	UCB_BMHD3_COPY_CODE - 32-bit CODE	$067F8_H$	

48.4.3.6 Register overview - UCB0_15 (ascending offset address)

Table 2531 Register overview - UCB0_15 (ascending offset address)

Short name	Long name	Offset address	See
CUST_RTC_DATA0_WORDx	Customer RTC data0 Word x	07800_H+x*4	8821
UCB0_15_PW0	256-bit password protection, PW0 (least significant)	$07FD0_H$	

(table continues...)

Table 2531 (continued) Register overview - UCB0_15 (ascending offset address)

Short name	Long name	Offset address	See
UCB0_15_PW1	256-bit password protection, PW1	$07FD4_H$	
UCB0_15_PW2	256-bit password protection, PW2	$07FD8_H$	
UCB0_15_PW3	256-bit password protection, PW3	$07FDC_H$	
UCB0_15_PW4	256-bit password protection, PW4	$07FE0_H$	
UCB0_15_PW5	256-bit password protection, PW5	$07FE4_H$	
UCB0_15_PW6	256-bit password protection, PW6	$07FE8_H$	
UCB0_15_PW7	256-bit password protection, PW7	$07FEC_H$	
UCB0_15_CONFIRMATION	Confirmation CODE ORIGINAL, 32-bit	$07FF0_H$	
UCB0_15_CONFIRMATION_COPY	Confirmation CODE COPY, 32-bit	$07FF8_H$	

48.4.3.6.1 Customer RTC data0 Word x

CUST_RTC_DATA0_WORDx (x=0-499) Offset address: 07800_H+x*4

Customer RTC data0 Word x value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CUST_RTC_DATA0_WORD															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CUST_RTC_DATA0_WORD															
r															

Field	Bits	Type	Description
CUST_RTC_DATA0_WORD	31:0	r	**CUST_RTC_DATA0_WORD**

48.4.3.7 Register overview - UCB0_16 (ascending offset address)

Table 2532 Register overview - UCB0_16 (ascending offset address)

Short name	Long name	Offset address	See
CUST_RTC_DATA1_WORDx	Customer RTC data1 Word x	08000_H+x*4	8822
UCB0_16_PW0	256-bit password protection, PW0 (least significant)	$087D0_H$	
UCB0_16_PW1	256-bit password protection, PW1	$087D4_H$	
UCB0_16_PW2	256-bit password protection, PW2	$087D8_H$	
UCB0_16_PW3	256-bit password protection, PW3	$087DC_H$	

(table continues...)

Table 2532 (continued) Register overview - UCB0_16 (ascending offset address)

Short name	Long name	Offset address	See
UCB0_16_PW4	256-bit password protection, PW4	$087E0_H$	
UCB0_16_PW5	256-bit password protection, PW5	$087E4_H$	
UCB0_16_PW6	256-bit password protection, PW6	$087E8_H$	
UCB0_16_PW7	256-bit password protection, PW7	$087EC_H$	
UCB0_16_CONFIRMATION	Confirmation CODE ORIGINAL, 32-bit	$087F0_H$	
UCB0_16_CONFIRMATION_COPY	Confirmation CODE COPY, 32-bit	$087F8_H$	

48.4.3.7.1 Customer RTC data1 Word x

CUST_RTC_DATA1_WORDx (x=0-499) Offset address: 08000_H+x*4

Customer RTC data1 Word x value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CUST_RTC_DATA1_WORD															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CUST_RTC_DATA1_WORD															
r															

Field	Bits	Type	Description
CUST_RTC_DATA1_WORD	31:0	r	**CUST_RTC_DATA1_WORD**

48.4.3.8 Register overview - UCB0_17 (ascending offset address)

Table 2533 Register overview - UCB0_17 (ascending offset address)

Short name	Long name	Offset address	See
UCB0_17_SAL	System Address of this Location	08800_H	8823
UCB0_17_LSENA	User selection for LOCKSTEP activation	08804_H	8823
UCB0_17_RESERVED_1	RESERVED	08808_H	
UCB0_SMM_EXTIF_ESR0CNTCTRL	User configuration value for SMM_EXTIF_ESR0CNTCTRL	$0880C_H$	8824
UCB0_PMS_PAD_ESR2CON	User configuration value for PMS_PAD_ESR2CON	08810_H	8824
UCB0_RAMINIT	User selection for RTC SRAMs initialization	08814_H	8825
UCB0_LBISTEXE	User selection for LBIST execution	08818_H	8826

(table continues...)

Table 2533 (continued) Register overview - UCB0_17 (ascending offset address)

Short name	Long name	Offset address	See
UCB0_HOSTRAMAIN	HOST SRAMs auto-init control	$0881C_H$	8826
UCB0_SWAP_ena	User selection for SWAP activation	08820_H	8827
UCB0_ADDRR_ena	User selection for ADDRESS ROUTING	08824_H	8827
CRC_RTCUSERCFGO1	Check Result for this table	08828_H	8828
UCB0_17_PW0	256-bit password protection, PW0 (least significant)	$08FD0_H$	
UCB0_17_PW1	256-bit password protection, PW1	$08FD4_H$	
UCB0_17_PW2	256-bit password protection, PW2	$08FD8_H$	
UCB0_17_PW3	256-bit password protection, PW3	$08FDC_H$	
UCB0_17_PW4	256-bit password protection, PW4	$08FE0_H$	
UCB0_17_PW5	256-bit password protection, PW5	$08FE4_H$	
UCB0_17_PW6	256-bit password protection, PW6	$08FE8_H$	
UCB0_17_PW7	256-bit password protection, PW7	$08FEC_H$	
UCB0_17_CONFIRMATION	Confirmation CODE ORIGINAL, 32-bit	$08FF0_H$	

48.4.3.8.1 System Address of this Location

This location is programmed with device-specific value during production.

UCB0_17_SAL Offset address: 08800_H

System Address of this Location value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UCB0_17_SAL															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UCB0_17_SAL															
r															

Field	Bits	Type	Description
UCB0_17_SAL	31:0	r	**SAL value for reference 1 in UCB_RTC_USERCFG**

48.4.3.8.2 User selection for LOCKSTEP activation

UCB0_17_LSENA Offset address: 08804_H

User selection for LOCKSTEP activation value: $0000\ 003F_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES0										LSENA5	LSENA4	LSENA3	LSENA2	LSENA1	RES
r										r	r	r	r	r	r

Field	Bits	Type	Description
RES	0	r	**Reserved, write with 1**
LSENAx (x=1-5)	x	r	**Lockstep handling during start-up for CPUx** 0_B Lockstep shall be disabled during start-up 1_B Lockstep shall be kept enabled
RES0	31:6	r	**Reserved, write with 0**

48.4.3.8.3 User configuration value for SMM_EXTIF_ESR0CNTCTRL

UCB0_SMM_EXTIF_ESR0CNTCTRL Offset address: $0880C_H$

User configuration value for SMM_EXTIF_ESR0CNTCTRL value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SMM_ESR0CNTCTRL															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SMM_ESR0CNTCTRL															
r															

Field	Bits	Type	Description
SMM_ESR0CNTCTRL	31:0	r	**SMM_ESR0CNTCTRL user configuration value**

48.4.3.8.4 User configuration value for PMS_PAD_ESR2CON

UCB0_PMS_PAD_ESR2CON Offset address: 08810_H

User configuration value for PMS_PAD_ESR2CON value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UCB0_PAD_ESR2CON															
r															

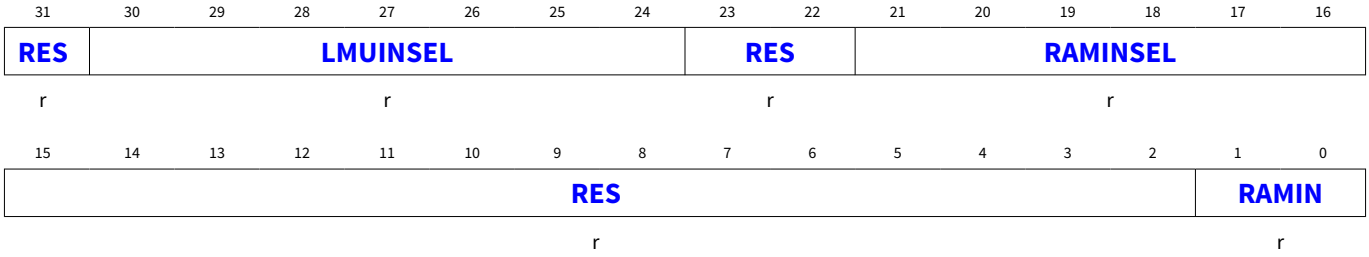
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UCB0_PAD_ESR2CON															
r															

Field	Bits	Type	Description
UCB0_PAD_ESR2CON	31:0	r	**PAD_ESR2CON user configuration value**

48.4.3.8.5 User selection for RTC SRAMs initialization

UCB0_RAMINIT Offset address: 08814_H

User selection for RTC SRAMs initialization value: $0000\ 0000_H$

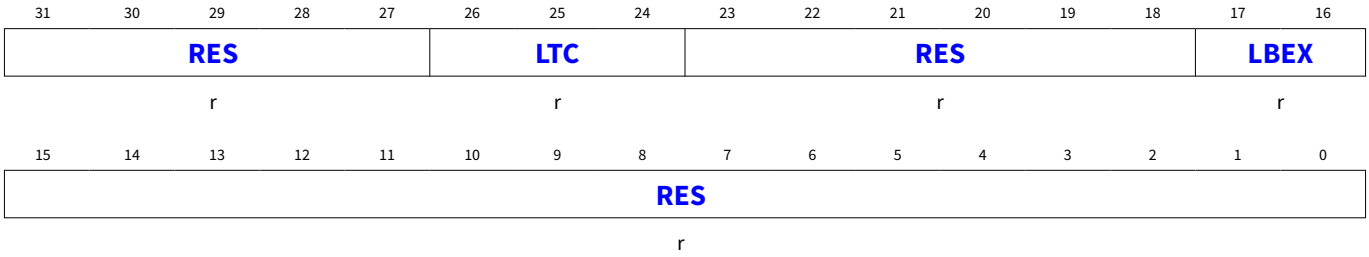


Field	Bits	Type	Description
RAMIN	1:0	r	**RTC SRAM initialization done by FW** 00_B **WCPORST_INIT**: after warm and cold power-on 01_B **WPORST_INIT**: after warm power-on only 10_B **CPORST_INIT**: after cold power-on only 11_B **NO_INIT**: never
RES	15:2, 23:22, 31	r	**Reserved, write with 0**
RAMINSEL	21:16	r	**RTC CPUx SRAM initialization done by FW** 0bxxxxx0 - for CPU0 SRAMs 0bxxxx0x - for CPU1 SRAMs 0bxxx0xx - for CPU2 SRAMs 0bxx0xxx - for CPU3 SRAMs 0bx0xxxx - for CPU4 SRAMs 0b0xxxxx - for CPU5 SRAMs
LMUINSEL	30:24	r	**RTC LMUx SRAM initialization done by FW** 0bxxxxxx0 - for CPU0 DLMU SRAM 0bxxxxx0x - for CPU1 DLMU SRAM 0bxxxx0xx - for CPU2 DLMU SRAM 0bxxx0xxx - for CPU3 DLMU SRAM 0bxx0xxxx - for CPU4 DLMU SRAM 0bx0xxxxx - for CPU5 DLMU SRAM else - reserved

48.4.3.8.6 User selection for LBIST execution

UCB0_LBISTEXE Offset address: 08818_H

User selection for LBIST execution value: $0000\ 0000_H$

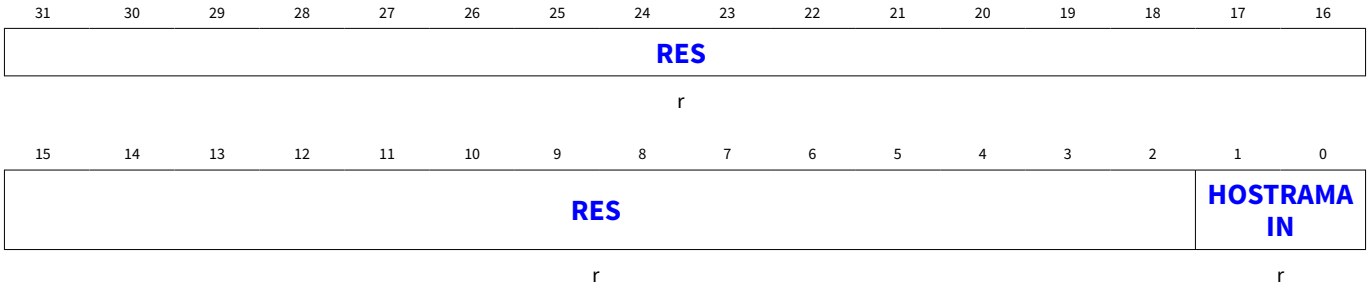


Field	Bits	Type	Description
RES	15:0, 23:18, 31:27	r	**Reserved, write with 0**
LBEX	17:16	r	**LBIST execution during start-up** 11b - triggered by FW 00b - not triggered by FW else - invalid, enter error state
LTC	26:24	r	**LBIST regtrigger count** 0 - enter error state if the first run fails (PORST triggered) 1..6 - Max number of LBIST-execution retries during boot, then enter error state if PORST still triggered 7 - legacy behavior - no retries-counting, LBIST may run endless if PORST permanently triggered

48.4.3.8.7 HOST SRAMs auto-init control

UCB0_HOSTRAMAIN Offset address: $0881C_H$

HOST SRAMs auto-init control value: $0000\ 0000_H$

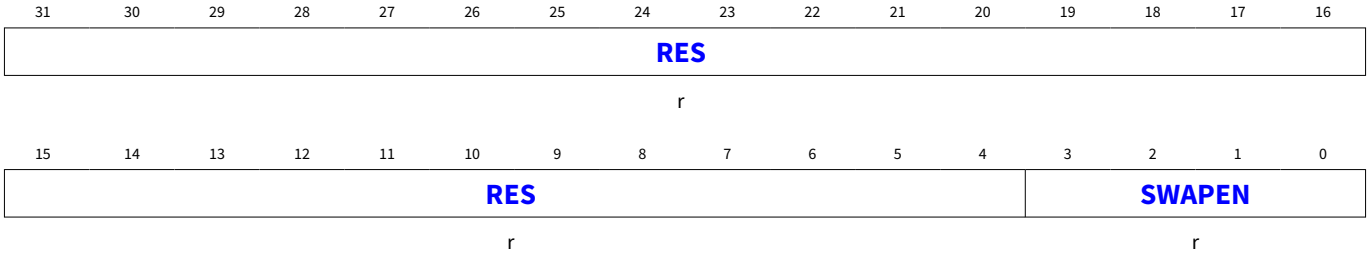


Field	Bits	Type	Description
HOSTRAMAIN	1:0	r	**HOST SRAMs auto-initialization** 01b - enabled 10b - disabled else - invalid, enter error state
RES	31:2	r	**Reserved, write with 0**

48.4.3.8.8 User selection for SWAP activation

UCB0_SWAP_ena Offset address: 08820_H

User selection for SWAP activation value: $0000\ 0000_H$

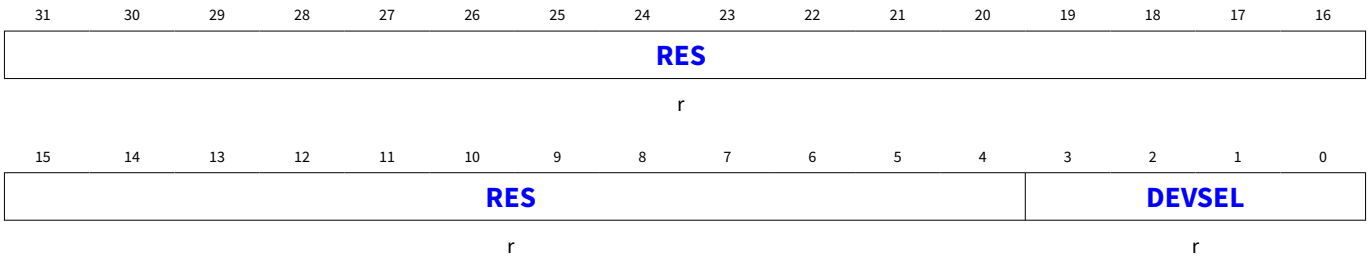


Field	Bits	Type	Description
SWAPEN	3:0	r	**SWAP activation for HOST cluster** 1010b - enable 0101b - disable else - invalid, enter fall-back/error state
RES	31:4	r	**Reserved, write with 0**

48.4.3.8.9 User selection for ADDRESS ROUTING

UCB0_ADDRR_ena Offset address: 08824_H

User selection for ADDRESS ROUTING value: $0000\ 0000_H$

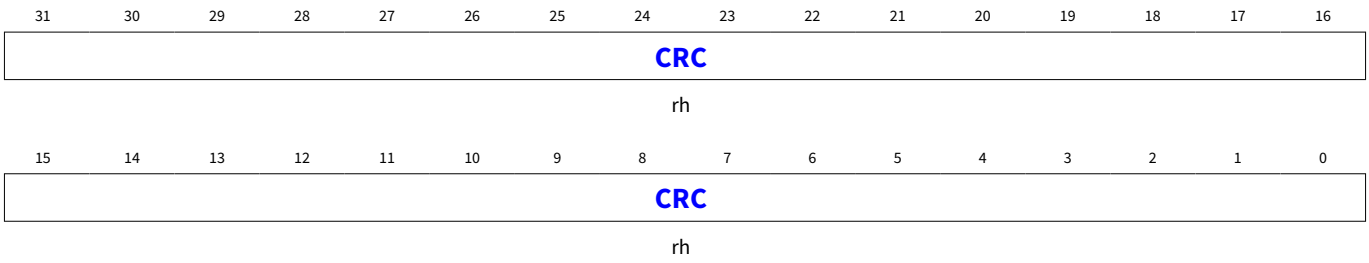


Field	Bits	Type	Description
DEVSEL	3:0	r	**Device selection for address routing** 0000b - no address routing Group 1 devices: 0010b - TC42x 0100b - TC44x 1000b - TC48x 1001b - TC49xA 1011b - TC4Zx 1101b - TC4Dx Group 2 devices: 0110b - TC46x 1001b - TC49xN 1010b - TC4Px 1110b - TC4Ex else - invalid
RES	31:4	r	**Reserved, write with 0**

48.4.3.8.10 Check Result for this table

CRC_RTCUSERCFGO1 Offset address: 08828_H

Check Result for this table value: $0000\ 0000_H$



Field	Bits	Type	Description
CRC	31:0	rh	**Check result for this table**

48.4.3.9 Register overview - UCB0_19 (ascending offset address)

Table 2534 Register overview - UCB0_19 (ascending offset address)

Short name	Long name	Offset address	See
SWAP_ORIG_SALx	SWAP_ORIG_SAL x	09800_H+x*10_H	8829
SWAP_ORIG_STATUSx	SWAP_ORIG_STATUS x	09804_H+x*10_H	8829
SWAP_ORIG_MARKERx	SWAP_ORIG_MARKER x	09808_H+x*10_H	8830

(table continues...)

Table 2534 (continued) Register overview - UCB0_19 (ascending offset address)

Short name	Long name	Offset address	See
SWAP_ORIG_CRCSEx	SWAP_ORIG_CRCSE x	$0980C_H$+x*10_H	8830
UCB0_19_PW0	256-bit password protection, PW0 (least significant)	$09FD0_H$	
UCB0_19_PW1	256-bit password protection, PW1	$09FD4_H$	
UCB0_19_PW2	256-bit password protection, PW2	$09FD8_H$	
UCB0_19_PW3	256-bit password protection, PW3	$09FDC_H$	
UCB0_19_PW4	256-bit password protection, PW4	$09FE0_H$	
UCB0_19_PW5	256-bit password protection, PW5	$09FE4_H$	
UCB0_19_PW6	256-bit password protection, PW6	$09FE8_H$	
UCB0_19_PW7	256-bit password protection, PW7	$09FEC_H$	
UCB0_19_CONFIRMATION	Confirmation CODE ORIGINAL, 32-bit	$09FF0_H$	

48.4.3.9.1 SWAP_ORIG_SAL x

SWAP_ORIG_SALx (x=0-31) Offset address: 09800_H+x*10_H

SWAP_ORIG_SAL x value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SALx															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SALx															
r															

Field	Bits	Type	Description
SALx	31:0	r	**System address of this location**

48.4.3.9.2 SWAP_ORIG_STATUS x

SWAP_ORIG_STATUSx (x=0-31) Offset address: 09804_H+x*10_H

SWAP_ORIG_STATUS x value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
STATUSx															
rh															

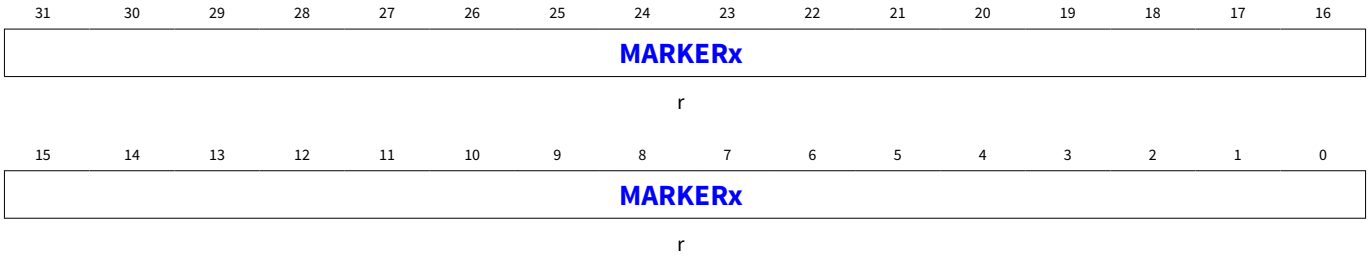
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STATUSx															
rh															

Field	Bits	Type	Description
STATUSx	31:0	rh	**Confirmation State of this entry** 43211234_H - This entry is VALID $57B5327F_H$ and any other value - This entry is INVALID

48.4.3.9.3 SWAP_ORIG_MARKER x

SWAP_ORIG_MARKERx (x=0-31) Offset address: 09808_H+x*10_H

SWAP_ORIG_MARKER x value: $0000\ 0000_H$

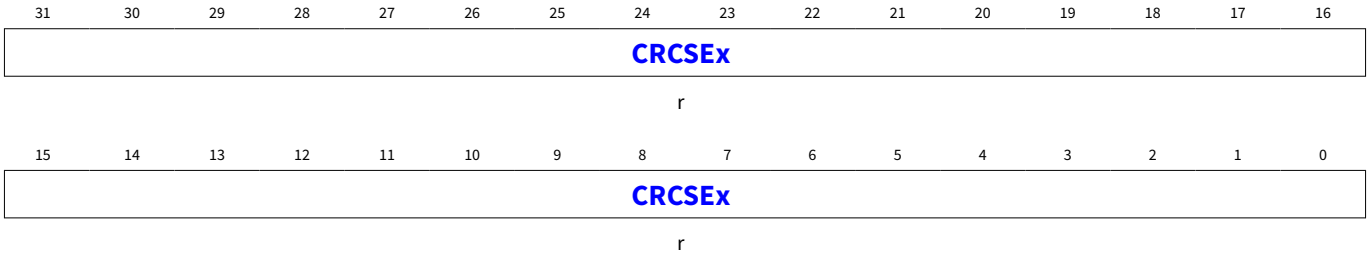


Field	Bits	Type	Description
MARKERx	31:0	r	**Marker SWAP-mode selection for HOST CPUs** 0x00000055 - linear or SOTA_A 0x000000AA - SOTA_B else - this entry is invalid

48.4.3.9.4 SWAP_ORIG_CRCSE x

SWAP_ORIG_CRCSEx (x=0-31) Offset address: $0980C_H+x*10_H$

SWAP_ORIG_CRCSE x value: $0000\ 0000_H$



Field	Bits	Type	Description
CRCSEx	31:0	r	**CRC over this SWAP-entry (SALx, STATUSx, MARKERx)**

48.4.3.10 Register overview - UCB0_21 (ascending offset address)

Table 2535 Register overview - UCB0_21 (ascending offset address)

Short name	Long name	Offset address	See
UCB0_21_DMU_GP_HOST_PFPROCONWP0An	DMU_GP_HOST_PFPROCONWP0A n	$0A800_H$+n*4	8832
UCB0_21_DMU_GP_HOST_PFPROCONWP0Bn	DMU_GP_HOST_PFPROCONWP0B n	$0A810_H$+n*4	8833
UCB0_21_DMU_GP_HOST_PFPROCONWP1An	DMU_GP_HOST_PFPROCONWP1A n	$0A820_H$+n*4	8833
UCB0_21_DMU_GP_HOST_PFPROCONWP1Bn	DMU_GP_HOST_PFPROCONWP1B n	$0A830_H$+n*4	8833
UCB0_21_DMU_GP_HOST_PFPROCONWP2An	DMU_GP_HOST_PFPROCONWP2A n	$0A840_H$+n*4	8834
UCB0_21_DMU_GP_HOST_PFPROCONWP2Bn	DMU_GP_HOST_PFPROCONWP2 n	$0A850_H$+n*4	8834
UCB0_21_DMU_GP_HOST_PFPROCONWP3An	DMU_GP_HOST_PFPROCONWP3A n	$0A860_H$+n*4	8835
UCB0_21_DMU_GP_HOST_PFPROCONWP3Bn	DMU_GP_HOST_PFPROCONWP3B n	$0A870_H$+n*4	8835
UCB0_21_DMU_GP_HOST_PFPROCONWP4An	DMU_GP_HOST_PFPROCONWP4A n	$0A880_H$+n*4	8836
UCB0_21_DMU_GP_HOST_PFPROCONWP4Bn	DMU_GP_HOST_PFPROCONWP4B n	$0A890_H$+n*4	8836
UCB0_21_DMU_GP_HOST_PFPROCONWP5An	DMU_GP_HOST_PFPROCONWP5A n	$0A8A0_H$+n*4	8836
UCB0_21_DMU_GP_HOST_PFPROCONWP5Bn	DMU_GP_HOST_PFPROCONWP5B n	$0A8B0_H$+n*4	8837
UCB0_21_DMU_GP_HOST_PFPROCONWP6An	Reserved for DMU_GP_HOST_PFPROCONWP6A n	$0A8C0_H$+n*4	
UCB0_21_DMU_GP_HOST_PFPROCONWP6Bn	Reserved for DMU_GP_HOST_PFPROCONWP6B n	$0A8D0_H$+n*4	

(table continues...)

Table 2535 (continued) Register overview - UCB0_21 (ascending offset address)

Short name	Long name	Offset address	See
UCB0_21_DMU_GP_HOST_PFPROCONWP7An	Reserved for DMU_GP_HOST_PFPROCONWP7A n	$0A8E0_H$+n*4	
UCB0_21_DMU_GP_HOST_PFPROCONWP7Bn	Reserved for DMU_GP_HOST_PFPROCONWP7B n	$0A8F0_H$+n*4	
UCB0_21_DMU_GP_HOST_PFPROCON	User configuration value for DMU_GP_HOST_PFPROCON	$0A900_H$	8837
UCB0_21_DMU_GP_HOST_DFPROCON	User configuration value for DMU_GP_HOST_DFPROCON	$0A904_H$	8838
UCB0_21_PW0	256-bit password protection, PW0 (least significant)	$0AFD0_H$	
UCB0_21_PW1	256-bit password protection, PW1	$0AFD4_H$	
UCB0_21_PW2	256-bit password protection, PW2	$0AFD8_H$	
UCB0_21_PW3	256-bit password protection, PW3	$0AFDC_H$	
UCB0_21_PW4	256-bit password protection, PW4	$0AFE0_H$	
UCB0_21_PW5	256-bit password protection, PW5	$0AFE4_H$	
UCB0_21_PW6	256-bit password protection, PW6	$0AFE8_H$	
UCB0_21_PW7	256-bit password protection, PW7	$0AFEC_H$	
UCB0_21_CONFIRMATION	Confirmation CODE ORIGINAL, 32-bit	$0AFF0_H$	

48.4.3.10.1 DMU_GP_HOST_PFPROCONWP0A n

UCB0_21_DMU_GP_HOST_PFPROCONWP0An (n=0-3) Offset address: $0A800_H$+n*4

DMU_GP_HOST_PFPROCONWP0A n value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONWP0An															
r															

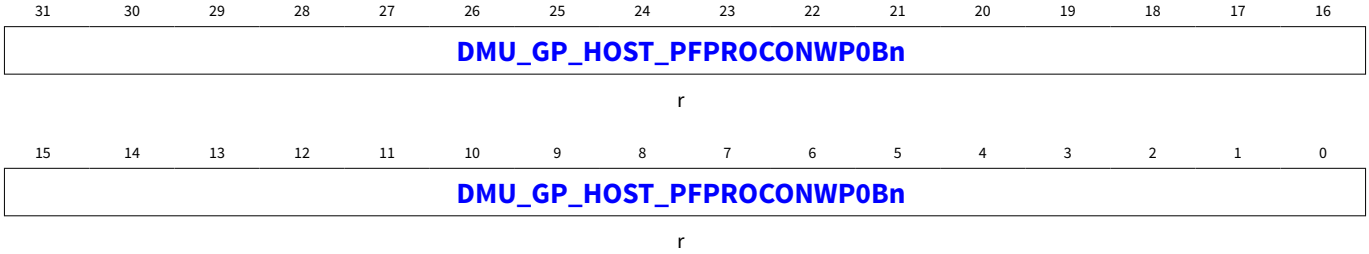
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONWP0An															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONWP0An	31:0	r	**DMU_GP_HOST_PFPROCONWP0An register value**

48.4.3.10.2 DMU_GP_HOST_PFPROCONWP0B n

UCB0_21_DMU_GP_HOST_PFPROCONWP0Bn (n=0-3) Offset address: $0A810_H$+n*4

DMU_GP_HOST_PFPROCONWP0B n value: $0000\ 0000_H$

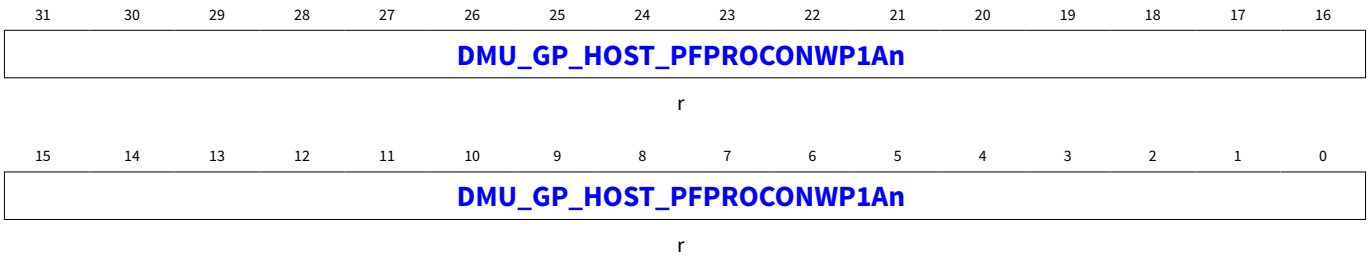


Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONWP0Bn	31:0	r	**DMU_GP_HOST_PFPROCONWP0Bn register value**

48.4.3.10.3 DMU_GP_HOST_PFPROCONWP1A n

UCB0_21_DMU_GP_HOST_PFPROCONWP1An (n=0-3) Offset address: $0A820_H$+n*4

DMU_GP_HOST_PFPROCONWP1A n value: $0000\ 0000_H$



Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONWP1An	31:0	r	**DMU_GP_HOST_PFPROCONWP1An register value**

48.4.3.10.4 DMU_GP_HOST_PFPROCONWP1B n

UCB0_21_DMU_GP_HOST_PFPROCONWP1Bn (n=0-3) Offset address: $0A830_H$+n*4

DMU_GP_HOST_PFPROCONWP1B n value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONWP1Bn															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONWP1Bn															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONWP1Bn	31:0	r	**DMU_GP_HOST_PFPROCONWP1Bn register value**

48.4.3.10.5 DMU_GP_HOST_PFPROCONWP2A n

UCB0_21_DMU_GP_HOST_PFPROCONWP2An (n=0-3) Offset address: $0A840_H$+n*4

DMU_GP_HOST_PFPROCONWP2A n value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONWP2An															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONWP2An															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONWP2An	31:0	r	**DMU_GP_HOST_PFPROCONWP2An register value**

48.4.3.10.6 DMU_GP_HOST_PFPROCONWP2 n

UCB0_21_DMU_GP_HOST_PFPROCONWP2Bn (n=0-3) Offset address: $0A850_H$+n*4

DMU_GP_HOST_PFPROCONWP2 n value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONWP2Bn															
r															

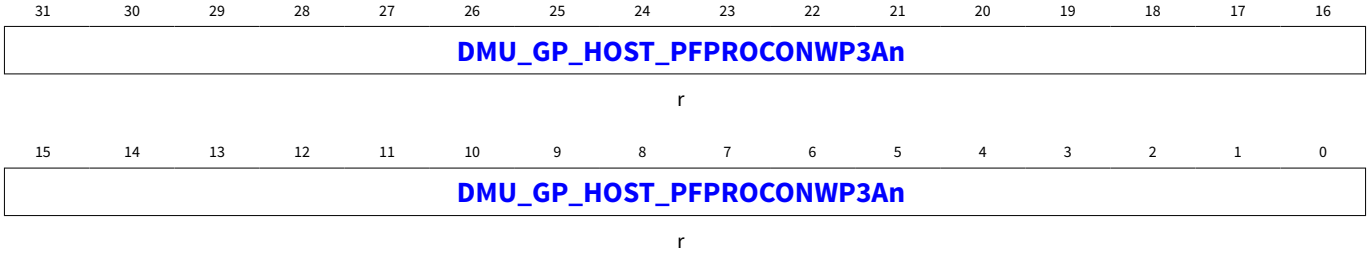
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONWP2Bn															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONWP2Bn	31:0	r	**DMU_GP_HOST_PFPROCONWP2Bn register value**

48.4.3.10.7 DMU_GP_HOST_PFPROCONWP3A n

UCB0_21_DMU_GP_HOST_PFPROCONWP3An (n=0-3) Offset address: $0A860_H+n*4$

DMU_GP_HOST_PFPROCONWP3A n value: $0000\ 0000_H$

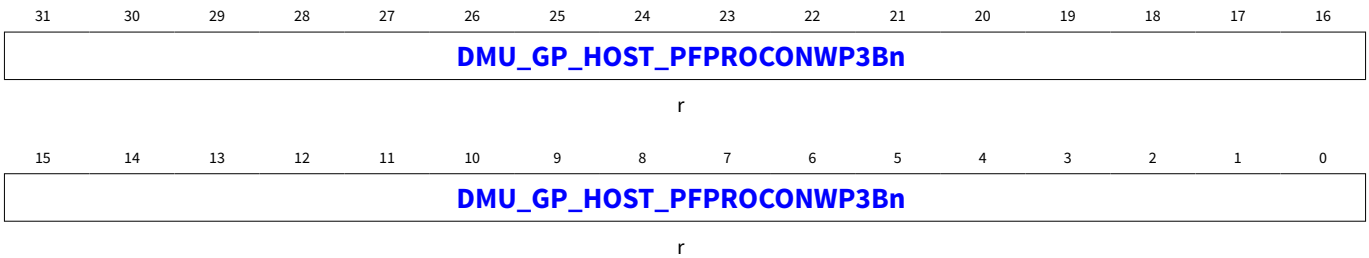


Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONWP3An	31:0	r	**DMU_GP_HOST_PFPROCONWP3An register value**

48.4.3.10.8 DMU_GP_HOST_PFPROCONWP3B n

UCB0_21_DMU_GP_HOST_PFPROCONWP3Bn (n=0-3) Offset address: $0A870_H+n*4$

DMU_GP_HOST_PFPROCONWP3B n value: $0000\ 0000_H$



Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONWP3Bn	31:0	r	**DMU_GP_HOST_PFPROCONWP3Bn register value**

48.4.3.10.9 DMU_GP_HOST_PFPROCONWP4A n

UCB0_21_DMU_GP_HOST_PFPROCONWP4An (n=0-3) Offset address: $0A880_H$+n*4

DMU_GP_HOST_PFPROCONWP4A n value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONWP4An															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONWP4An															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONWP4An	31:0	r	**DMU_GP_HOST_PFPROCONWP4An register value**

48.4.3.10.10 DMU_GP_HOST_PFPROCONWP4B n

UCB0_21_DMU_GP_HOST_PFPROCONWP4Bn (n=0-3) Offset address: $0A890_H$+n*4

DMU_GP_HOST_PFPROCONWP4B n value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONWP4Bn															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONWP4Bn															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONWP4Bn	31:0	r	**DMU_GP_HOST_PFPROCONWP4Bn register value**

48.4.3.10.11 DMU_GP_HOST_PFPROCONWP5A n

UCB0_21_DMU_GP_HOST_PFPROCONWP5An (n=0-3) Offset address: $0A8A0_H$+n*4

DMU_GP_HOST_PFPROCONWP5A n value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONWP5An															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONWP5An															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONWP5An	31:0	r	**DMU_GP_HOST_PFPROCONWP5An register value**

48.4.3.10.12 DMU_GP_HOST_PFPROCONWP5B n

UCB0_21_DMU_GP_HOST_PFPROCONWP5Bn (n=0-3) Offset address: $0A8B0_H+n*4$

DMU_GP_HOST_PFPROCONWP5B n value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONWP5Bn															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONWP5Bn															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONWP5Bn	31:0	r	**DMU_GP_HOST_PFPROCONWP5Bn register value**

48.4.3.10.13 User configuration value for DMU_GP_HOST_PFPROCON

UCB0_21_DMU_GP_HOST_PFPROCON Offset address: $0A900_H$

User configuration value for DMU_GP_HOST_PFPROCON value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCON															
r															

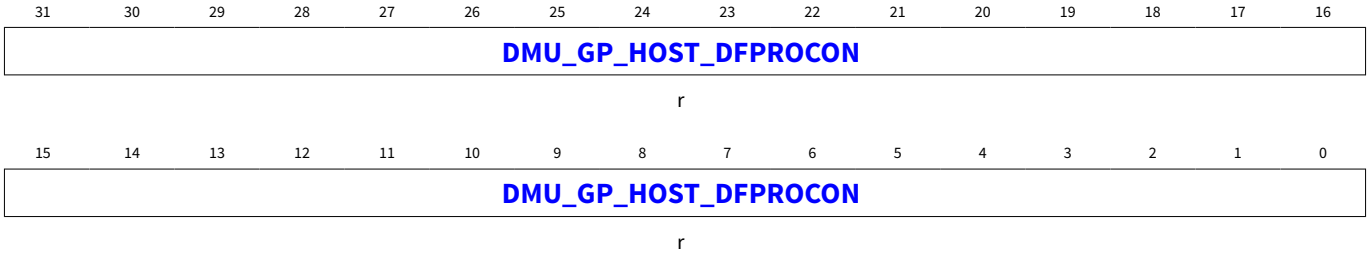
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCON															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCON	31:0	r	**DMU_GP_HOST_PFPROCON**

48.4.3.10.14 User configuration value for DMU_GP_HOST_DFPROCON

UCB0_21_DMU_GP_HOST_DFPROCON Offset address: $0A904_H$

User configuration value for DMU_GP_HOST_DFPROCON value: $0000\ 0000_H$



Field	Bits	Type	Description
DMU_GP_HOST_DFPROCON	31:0	r	**DMU_GP_HOST_DFPROCON**

48.4.3.11 Register overview - UCB0_23 (ascending offset address)

Table 2536 Register overview - UCB0_23 (ascending offset address)

Short name	Long name	Offset address	See
UCB0_23_DMU_GP_HOST_PFECPRIO0An	DMU_GP_HOST_PFECPRIO0A n	$0B800_H$+n*4	8839
UCB0_23_DMU_GP_HOST_PFECPRIO0Bn	DMU_GP_HOST_PFECPRIO0B n	$0B810_H$+n*4	8840
UCB0_23_DMU_GP_HOST_PFECPRIO1An	DMU_GP_HOST_PFECPRIO1A n	$0B820_H$+n*4	8840
UCB0_23_DMU_GP_HOST_PFECPRIO1Bn	DMU_GP_HOST_PFECPRIO1B n	$0B830_H$+n*4	8840
UCB0_23_DMU_GP_HOST_PFECPRIO2An	DMU_GP_HOST_PFECPRIO2A n	$0B840_H$+n*4	8841
UCB0_23_DMU_GP_HOST_PFECPRIO2Bn	DMU_GP_HOST_PFECPRIO2B n	$0B850_H$+n*4	8841
UCB0_23_DMU_GP_HOST_PFECPRIO3An	DMU_GP_HOST_PFECPRIO3A n	$0B860_H$+n*4	8842
UCB0_23_DMU_GP_HOST_PFECPRIO3Bn	DMU_GP_HOST_PFECPRIO3B n	$0B870_H$+n*4	8842
UCB0_23_DMU_GP_HOST_PFECPRIO4An	DMU_GP_HOST_PFECPRIO4A n	$0B880_H$+n*4	8842

(table continues...)

Table 2536 (continued) Register overview - UCB0_23 (ascending offset address)

Short name	Long name	Offset address	See
UCB0_23_DMU_GP_HOST_PFECPRIO4Bn	DMU_GP_HOST_PFECPRIO4B n	$0B890_H$+n*4	8843
UCB0_23_DMU_GP_HOST_PFECPRIO5An	DMU_GP_HOST_PFECPRIO5A n	$0B8A0_H$+n*4	8843
UCB0_23_DMU_GP_HOST_PFECPRIO5Bn	DMU_GP_HOST_PFECPRIO5B n	$0B8B0_H$+n*4	8844
UCB0_23_DMU_GP_HOST_PFECPRIO6An	Reserved for DMU_GP_HOST_PFECPRIO6A n	$0B8C0_H$+n*4	
UCB0_23_DMU_GP_HOST_PFECPRIO6Bn	Reserved for DMU_GP_HOST_PFECPRIO6B n	$0B8D0_H$+n*4	
UCB0_23_DMU_GP_HOST_PFECPRIO7An	Reserved for DMU_GP_HOST_PFECPRIO7A n	$0B8E0_H$+n*4	
UCB0_23_DMU_GP_HOST_PFECPRIO7Bn	Reserved for DMU_GP_HOST_PFECPRIO7B n	$0B8F0_H$+n*4	
UCB0_23_PW0	256-bit password protection, PW0 (least significant)	$0BFD0_H$	
UCB0_23_PW1	256-bit password protection, PW1	$0BFD4_H$	
UCB0_23_PW2	256-bit password protection, PW2	$0BFD8_H$	
UCB0_23_PW3	256-bit password protection, PW3	$0BFDC_H$	
UCB0_23_PW4	256-bit password protection, PW4	$0BFE0_H$	
UCB0_23_PW5	256-bit password protection, PW5	$0BFE4_H$	
UCB0_23_PW6	256-bit password protection, PW6	$0BFE8_H$	
UCB0_23_PW7	256-bit password protection, PW7	$0BFEC_H$	
UCB0_23_CONFIRMATION	Confirmation CODE ORIGINAL, 32-bit	$0BFF0_H$	

48.4.3.11.1 DMU_GP_HOST_PFECPRIO0A n

UCB0_23_DMU_GP_HOST_PFECPRIO0An (n=0-3) Offset address: $0B800_H$+n*4

DMU_GP_HOST_PFECPRIO0A n value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFECPRIO0An															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFECPRIO0An															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFECPRIO0An	31:0	r	**DMU_GP_HOST_PFECPRIO0An register value**

48.4.3.11.2 DMU_GP_HOST_PFECPRIO0B n

UCB0_23_DMU_GP_HOST_PFECPRIO0Bn (n=0-3) Offset address: $0B810_H$+n*4

DMU_GP_HOST_PFECPRIO0B n value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFECPRIO0Bn															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFECPRIO0Bn															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFECPRIO0Bn	31:0	r	**DMU_GP_HOST_PFECPRIO0Bn register value**

48.4.3.11.3 DMU_GP_HOST_PFECPRIO1A n

UCB0_23_DMU_GP_HOST_PFECPRIO1An (n=0-3) Offset address: $0B820_H$+n*4

DMU_GP_HOST_PFECPRIO1A n value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFECPRIO1An															
r															

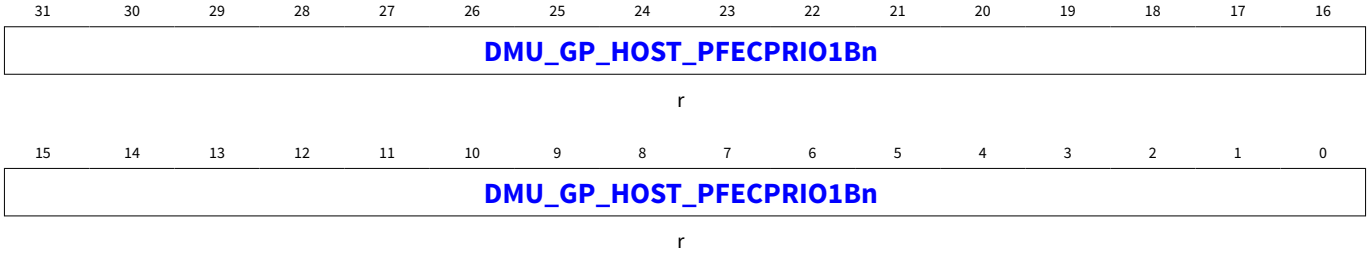
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFECPRIO1An															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFECPRIO1An	31:0	r	**DMU_GP_HOST_PFECPRIO1An register value**

48.4.3.11.4 DMU_GP_HOST_PFECPRIO1B n

UCB0_23_DMU_GP_HOST_PFECPRIO1Bn (n=0-3) Offset address: $0B830_H$+n*4

DMU_GP_HOST_PFECPRIO1B n value: $0000\ 0000_H$

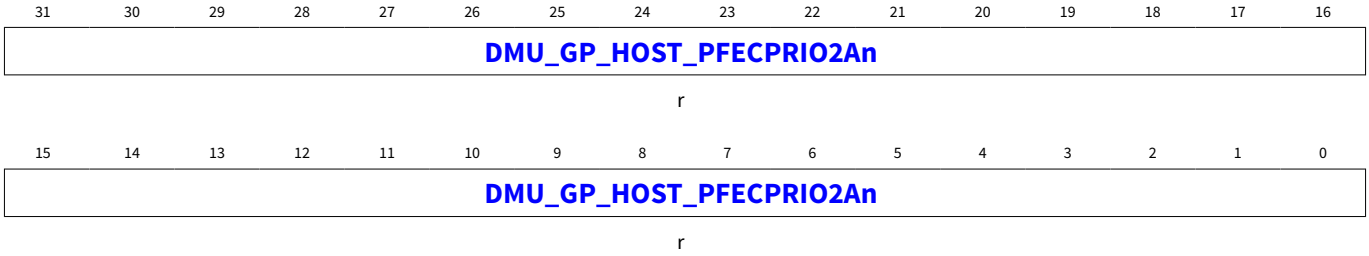


Field	Bits	Type	Description
DMU_GP_HOST_PFECPRIO1Bn	31:0	r	**DMU_GP_HOST_PFECPRIO1Bn register value**

48.4.3.11.5 DMU_GP_HOST_PFECPRIO2A n

UCB0_23_DMU_GP_HOST_PFECPRIO2An (n=0-3) Offset address: $0B840_H$+n*4

DMU_GP_HOST_PFECPRIO2A n value: $0000\ 0000_H$

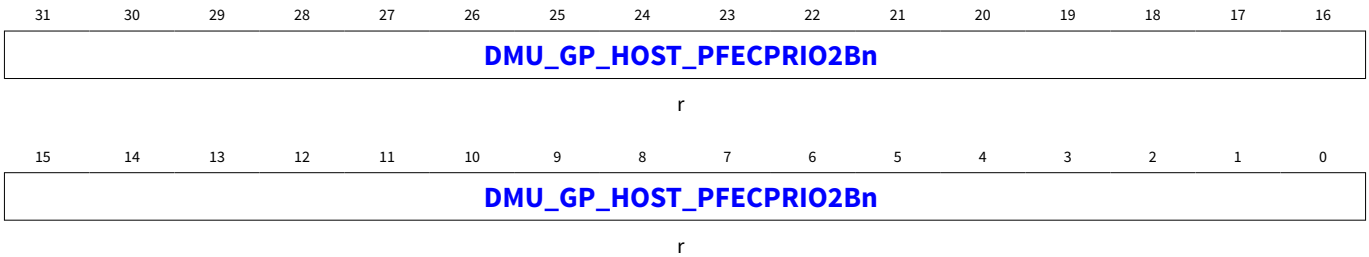


Field	Bits	Type	Description
DMU_GP_HOST_PFECPRIO2An	31:0	r	**DMU_GP_HOST_PFECPRIO2An register value**

48.4.3.11.6 DMU_GP_HOST_PFECPRIO2B n

UCB0_23_DMU_GP_HOST_PFECPRIO2Bn (n=0-3) Offset address: $0B850_H$+n*4

DMU_GP_HOST_PFECPRIO2B n value: $0000\ 0000_H$

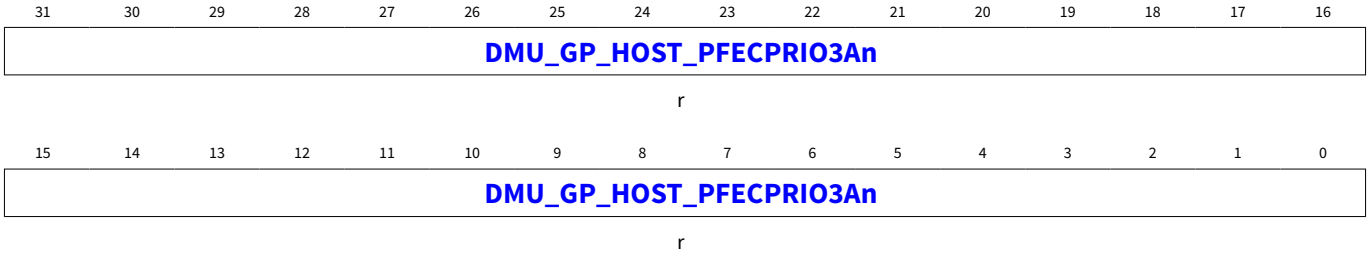


Field	Bits	Type	Description
DMU_GP_HOST_PFECPRIO2Bn	31:0	r	**DMU_GP_HOST_PFECPRIO2Bn register value**

48.4.3.11.7 DMU_GP_HOST_PFECPRIO3A n

UCB0_23_DMU_GP_HOST_PFECPRIO3An (n=0-3) Offset address: $0B860_H$+n*4

DMU_GP_HOST_PFECPRIO3A n value: $0000\ 0000_H$

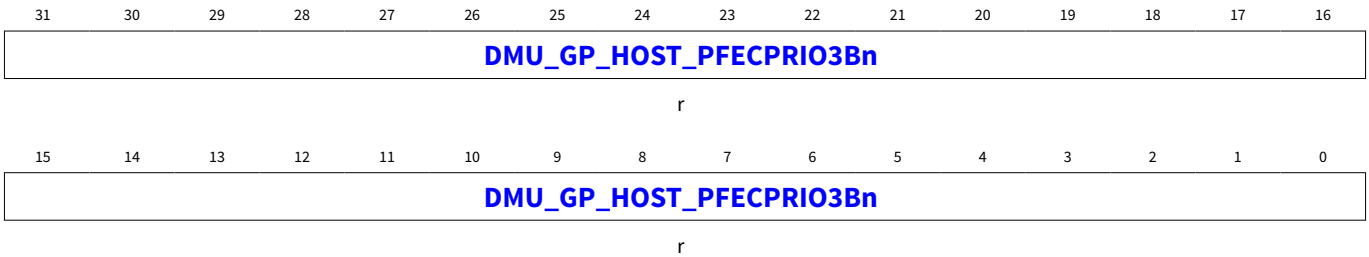


Field	Bits	Type	Description
DMU_GP_HOST_PFECPRIO3An	31:0	r	**DMU_GP_HOST_PFECPRIO3An register value**

48.4.3.11.8 DMU_GP_HOST_PFECPRIO3B n

UCB0_23_DMU_GP_HOST_PFECPRIO3Bn (n=0-3) Offset address: $0B870_H$+n*4

DMU_GP_HOST_PFECPRIO3B n value: $0000\ 0000_H$



Field	Bits	Type	Description
DMU_GP_HOST_PFECPRIO3Bn	31:0	r	**DMU_GP_HOST_PFECPRIO3Bn register value**

48.4.3.11.9 DMU_GP_HOST_PFECPRIO4A n

UCB0_23_DMU_GP_HOST_PFECPRIO4An (n=0-3) Offset address: $0B880_H$+n*4

DMU_GP_HOST_PFECPRIO4A n value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFECPRIO4An															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFECPRIO4An															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFECPRIO4An	31:0	r	**DMU_GP_HOST_PFECPRIO4An register value**

48.4.3.11.10 DMU_GP_HOST_PFECPRIO4B n

UCB0_23_DMU_GP_HOST_PFECPRIO4Bn (n=0-3) Offset address: $0B890_H$+n*4

DMU_GP_HOST_PFECPRIO4B n value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFECPRIO4Bn															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFECPRIO4Bn															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFECPRIO4Bn	31:0	r	**DMU_GP_HOST_PFECPRIO4Bn register value**

48.4.3.11.11 DMU_GP_HOST_PFECPRIO5A n

UCB0_23_DMU_GP_HOST_PFECPRIO5An (n=0-3) Offset address: $0B8A0_H$+n*4

DMU_GP_HOST_PFECPRIO5A n value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFECPRIO5An															
r															

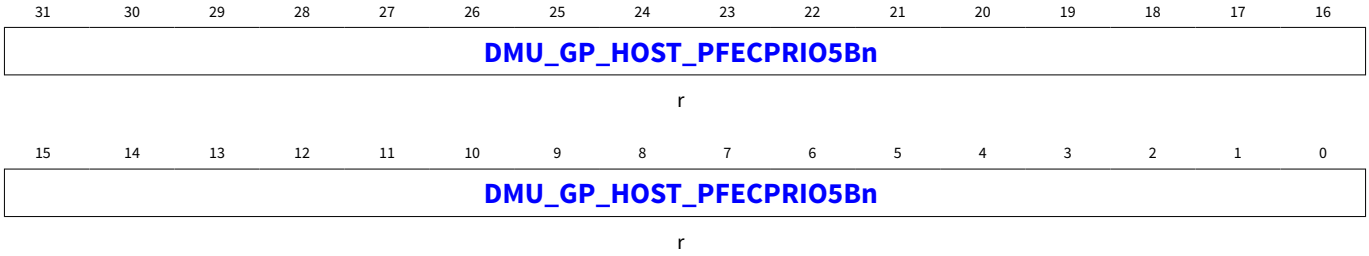
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFECPRIO5An															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFECPRIO5An	31:0	r	**DMU_GP_HOST_PFECPRIO5An register value**

48.4.3.11.12 DMU_GP_HOST_PFECPRIO5B n

UCB0_23_DMU_GP_HOST_PFECPRIO5Bn (n=0-3) Offset address: $0B8B0_H$+n*4

DMU_GP_HOST_PFECPRIO5B n value: $0000\ 0000_H$



Field	Bits	Type	Description
DMU_GP_HOST_PFECPRIO5Bn	31:0	r	**DMU_GP_HOST_PFECPRIO5Bn register value**

48.4.3.12 Register overview - UCB0_25 (ascending offset address)

Table 2537 Register overview - UCB0_25 (ascending offset address)

Short name	Long name	Offset address	See
UCB0_25_DMU_GP_HOST_PFPROCONPWP0An	DMU_GP_HOST_PFPROCONPWP0An, OTP set 1	$0C800_H$+n*4	8849
UCB0_25_DMU_GP_HOST_PFPROCONPWP0Bn	DMU_GP_HOST_PFPROCONPWP0Bn, OTP set 1	$0C810_H$+n*4	8850
UCB0_25_DMU_GP_HOST_PFPROCONPWP1An	DMU_GP_HOST_PFPROCONPWP1An, OTP set 1	$0C820_H$+n*4	8850
UCB0_25_DMU_GP_HOST_PFPROCONPWP1Bn	DMU_GP_HOST_PFPROCONPWP1Bn, OTP set 1	$0C830_H$+n*4	8850
UCB0_25_DMU_GP_HOST_PFPROCONPWP2An	DMU_GP_HOST_PFPROCONPWP2An, OTP set 1	$0C840_H$+n*4	8851
UCB0_25_DMU_GP_HOST_PFPROCONPWP2Bn	DMU_GP_HOST_PFPROCONPWP2Bn, OTP set 1	$0C850_H$+n*4	8851

(table continues...)

Table 2537 (continued) Register overview - UCB0_25 (ascending offset address)

Short name	Long name	Offset address	See
UCB0_25_DMU_GP_HOST_PFPROCONPWP3An	DMU_GP_HOST_PFPROCONPWP3An, OTP set 1	$0C860_H$+n*4	8852
UCB0_25_DMU_GP_HOST_PFPROCONPWP3Bn	DMU_GP_HOST_PFPROCONPWP3Bn, OTP set 1	$0C870_H$+n*4	8852
UCB0_25_DMU_GP_HOST_PFPROCONPWP4An	DMU_GP_HOST_PFPROCONPWP4An, OTP set 1	$0C880_H$+n*4	8853
UCB0_25_DMU_GP_HOST_PFPROCONPWP4Bn	DMU_GP_HOST_PFPROCONPWP4Bn, OTP set 1	$0C890_H$+n*4	8853
UCB0_25_DMU_GP_HOST_PFPROCONPWP5An	DMU_GP_HOST_PFPROCONPWP5An, OTP set 1	$0C8A0_H$+n*4	8853
UCB0_25_DMU_GP_HOST_PFPROCONPWP5Bn	DMU_GP_HOST_PFPROCONPWP5Bn, OTP set 1	$0C8B0_H$+n*4	8854
UCB0_25_DMU_GP_HOST_PFPROCONPWOP0An	DMU_GP_HOST_PFPROCONPWOP0An, OTP set 1	$0C900_H$+n*4	8854
UCB0_25_DMU_GP_HOST_PFPROCONPWOP0Bn	DMU_GP_HOST_PFPROCONPWOP0Bn, OTP set 1	$0C910_H$+n*4	8855
UCB0_25_DMU_GP_HOST_PFPROCONPWOP1An	DMU_GP_HOST_PFPROCONPWOP1An, OTP set 1	$0C920_H$+n*4	8855
UCB0_25_DMU_GP_HOST_PFPROCONPWOP1Bn	DMU_GP_HOST_PFPROCONPWOP1Bn, OTP set 1	$0C930_H$+n*4	8856
UCB0_25_DMU_GP_HOST_PFPROCONPWOP2An	DMU_GP_HOST_PFPROCONPWOP2An, OTP set 1	$0C940_H$+n*4	8856
UCB0_25_DMU_GP_HOST_PFPROCONPWOP2Bn	DMU_GP_HOST_PFPROCONPWOP2Bn, OTP set 1	$0C950_H$+n*4	8856
UCB0_25_DMU_GP_HOST_PFPROCONPWOP3An	DMU_GP_HOST_PFPROCONPWOP3An, OTP set 1	$0C960_H$+n*4	8857
UCB0_25_DMU_GP_HOST_PFPROCONPWOP3Bn	DMU_GP_HOST_PFPROCONPWOP3Bn, OTP set 1	$0C970_H$+n*4	8857
UCB0_25_DMU_GP_HOST_PFPROCONPWOP4An	DMU_GP_HOST_PFPROCONPWOP4An, OTP set 1	$0C980_H$+n*4	8858

(table continues...)

Table 2537 (continued) Register overview - UCB0_25 (ascending offset address)

Short name	Long name	Offset address	See
UCB0_25_DMU_GP_HOST_PFPROCONPWOP4Bn	DMU_GP_HOST_PFPROCONPWOP4Bn, OTP set 1	$0C990_H$+n*4	8858
UCB0_25_DMU_GP_HOST_PFPROCONPWOP5An	DMU_GP_HOST_PFPROCONPWOP5An, OTP set 1	$0C9A0_H$+n*4	8859
UCB0_25_DMU_GP_HOST_PFPROCONPWOP5Bn	DMU_GP_HOST_PFPROCONPWOP5Bn, OTP set 1	$0C9B0_H$+n*4	8859
UCB0_25_DMU_GP_HOST_PFPROCONPWP0An	DMU_GP_HOST_PFPROCONPWP0An, OTP set 2	$0CA00_H$+n*4	8859
UCB0_25_DMU_GP_HOST_PFPROCONPWP0Bn	DMU_GP_HOST_PFPROCONPWP0Bn, OTP set 2	$0CA10_H$+n*4	8860
UCB0_25_DMU_GP_HOST_PFPROCONPWP1An	DMU_GP_HOST_PFPROCONPWP1An, OTP set 2	$0CA20_H$+n*4	8860
UCB0_25_DMU_GP_HOST_PFPROCONPWP1Bn	DMU_GP_HOST_PFPROCONPWP1Bn, OTP set 2	$0CA30_H$+n*4	8861
UCB0_25_DMU_GP_HOST_PFPROCONPWP2An	DMU_GP_HOST_PFPROCONPWP2An, OTP set 2	$0CA40_H$+n*4	8861
UCB0_25_DMU_GP_HOST_PFPROCONPWP2Bn	DMU_GP_HOST_PFPROCONPWP2Bn, OTP set 2	$0CA50_H$+n*4	8862
UCB0_25_DMU_GP_HOST_PFPROCONPWP3An	DMU_GP_HOST_PFPROCONPWP3An, OTP set 2	$0CA60_H$+n*4	8862
UCB0_25_DMU_GP_HOST_PFPROCONPWP3Bn	DMU_GP_HOST_PFPROCONPWP3Bn, OTP set 2	$0CA70_H$+n*4	8862
UCB0_25_DMU_GP_HOST_PFPROCONPWP4An	DMU_GP_HOST_PFPROCONPWP4An, OTP set 2	$0CA80_H$+n*4	8863
UCB0_25_DMU_GP_HOST_PFPROCONPWP4Bn	DMU_GP_HOST_PFPROCONPWP4Bn, OTP set 2	$0CA90_H$+n*4	8863
UCB0_25_DMU_GP_HOST_PFPROCONPWP5An	DMU_GP_HOST_PFPROCONPWP5An, OTP set 2	$0CAA0_H$+n*4	8864
UCB0_25_DMU_GP_HOST_PFPROCONPWP5Bn	DMU_GP_HOST_PFPROCONPWP5Bn, OTP set 2	$0CAB0_H$+n*4	8864

(table continues...)

Table 2537 (continued) Register overview - UCB0_25 (ascending offset address)

Short name	Long name	Offset address	See
UCB0_25_DMU_GP_HOST_PFPROCONPWOP0An	DMU_GP_HOST_PFPROCONPWOP0An, OTP set 2	$0CB00_H$+n*4	8865
UCB0_25_DMU_GP_HOST_PFPROCONPWOP0Bn	DMU_GP_HOST_PFPROCONPWOP0Bn, OTP set 2	$0CB10_H$+n*4	8865
UCB0_25_DMU_GP_HOST_PFPROCONPWOP1An	DMU_GP_HOST_PFPROCONPWOP1An, OTP set 2	$0CB20_H$+n*4	8865
UCB0_25_DMU_GP_HOST_PFPROCONPWOP1Bn	DMU_GP_HOST_PFPROCONPWOP1Bn, OTP set 2	$0CB30_H$+n*4	8866
UCB0_25_DMU_GP_HOST_PFPROCONPWOP2An	DMU_GP_HOST_PFPROCONPWOP2An, OTP set 2	$0CB40_H$+n*4	8866
UCB0_25_DMU_GP_HOST_PFPROCONPWOP2Bn	DMU_GP_HOST_PFPROCONPWOP2Bn, OTP set 2	$0CB50_H$+n*4	8867
UCB0_25_DMU_GP_HOST_PFPROCONPWOP3An	DMU_GP_HOST_PFPROCONPWOP3An, OTP set 2	$0CB60_H$+n*4	8867
UCB0_25_DMU_GP_HOST_PFPROCONPWOP3Bn	DMU_GP_HOST_PFPROCONPWOP3Bn, OTP set 2	$0CB70_H$+n*4	8868
UCB0_25_DMU_GP_HOST_PFPROCONPWOP4An	DMU_GP_HOST_PFPROCONPWOP4An, OTP set 2	$0CB80_H$+n*4	8868
UCB0_25_DMU_GP_HOST_PFPROCONPWOP4Bn	DMU_GP_HOST_PFPROCONPWOP4Bn, OTP set 2	$0CB90_H$+n*4	8868
UCB0_25_DMU_GP_HOST_PFPROCONPWOP5An	DMU_GP_HOST_PFPROCONPWOP5An, OTP set 2	$0CBA0_H$+n*4	8869
UCB0_25_DMU_GP_HOST_PFPROCONPWOP5Bn	DMU_GP_HOST_PFPROCONPWOP5Bn, OTP set 2	$0CBB0_H$+n*4	8869
UCB0_25_DMU_GP_HOST_PFPROCONPWP0An	DMU_GP_HOST_PFPROCONPWP0An, OTP set 3	$0CC00_H$+n*4	8870
UCB0_25_DMU_GP_HOST_PFPROCONPWP0Bn	DMU_GP_HOST_PFPROCONPWP0Bn, OTP set 3	$0CC10_H$+n*4	8870
UCB0_25_DMU_GP_HOST_PFPROCONPWP1An	DMU_GP_HOST_PFPROCONPWP1An, OTP set 3	$0CC20_H$+n*4	8871

(table continues...)

Table 2537 (continued) Register overview - UCB0_25 (ascending offset address)

Short name	Long name	Offset address	See
UCB0_25_DMU_GP_HOST_PFPROCONPWP1Bn	DMU_GP_HOST_PFPROCONPWP1Bn, OTP set 3	$0CC30_H$+n*4	8871
UCB0_25_DMU_GP_HOST_PFPROCONPWP2An	DMU_GP_HOST_PFPROCONPWP2An, OTP set 3	$0CC40_H$+n*4	8871
UCB0_25_DMU_GP_HOST_PFPROCONPWP2Bn	DMU_GP_HOST_PFPROCONPWP2Bn, OTP set 3	$0CC50_H$+n*4	8872
UCB0_25_DMU_GP_HOST_PFPROCONPWP3An	DMU_GP_HOST_PFPROCONPWP3An, OTP set 3	$0CC60_H$+n*4	8872
UCB0_25_DMU_GP_HOST_PFPROCONPWP3Bn	DMU_GP_HOST_PFPROCONPWP3Bn, OTP set 3	$0CC70_H$+n*4	8873
UCB0_25_DMU_GP_HOST_PFPROCONPWP4An	DMU_GP_HOST_PFPROCONPWP4An, OTP set 3	$0CC80_H$+n*4	8873
UCB0_25_DMU_GP_HOST_PFPROCONPWP4Bn	DMU_GP_HOST_PFPROCONPWP4Bn, OTP set 3	$0CC90_H$+n*4	
UCB0_25_DMU_GP_HOST_PFPROCONPWP5An	DMU_GP_HOST_PFPROCONPWP5An, OTP set 3	$0CCA0_H$+n*4	8874
UCB0_25_DMU_GP_HOST_PFPROCONPWP5Bn	DMU_GP_HOST_PFPROCONPWP5Bn, OTP set 3	$0CCB0_H$+n*4	8874
UCB0_25_DMU_GP_HOST_PFPROCONPWOP0An	DMU_GP_HOST_PFPROCONPWOP0An, OTP set 3	$0CD00_H$+n*4	8874
UCB0_25_DMU_GP_HOST_PFPROCONPWOP0Bn	DMU_GP_HOST_PFPROCONPWOP0Bn, OTP set 3	$0CD10_H$+n*4	8875
UCB0_25_DMU_GP_HOST_PFPROCONPWOP1An	DMU_GP_HOST_PFPROCONPWOP1An, OTP set 3	$0CD20_H$+n*4	8875
UCB0_25_DMU_GP_HOST_PFPROCONPWOP1Bn	DMU_GP_HOST_PFPROCONPWOP1Bn, OTP set 3	$0CD30_H$+n*4	8876
UCB0_25_DMU_GP_HOST_PFPROCONPWOP2An	DMU_GP_HOST_PFPROCONPWOP2An, OTP set 3	$0CD40_H$+n*4	8876
UCB0_25_DMU_GP_HOST_PFPROCONPWOP2Bn	DMU_GP_HOST_PFPROCONPWOP2Bn, OTP set 3	$0CD50_H$+n*4	8877

(table continues...)

Table 2537 (continued) Register overview - UCB0_25 (ascending offset address)

Short name	Long name	Offset address	See
UCB0_25_DMU_GP_HOST_PFPROCONPWOP3An	DMU_GP_HOST_PFPROCONPWOP3An, OTP set 3	$0CD60_H$+n*4	8877
UCB0_25_DMU_GP_HOST_PFPROCONPWOP3Bn	DMU_GP_HOST_PFPROCONPWOP3Bn, OTP set 3	$0CD70_H$+n*4	8877
UCB0_25_DMU_GP_HOST_PFPROCONPWOP4An	DMU_GP_HOST_PFPROCONPWOP4An, OTP set 3	$0CD80_H$+n*4	8878
UCB0_25_DMU_GP_HOST_PFPROCONPWOP4Bn	DMU_GP_HOST_PFPROCONPWOP4Bn, OTP set 3	$0CD90_H$+n*4	8878
UCB0_25_DMU_GP_HOST_PFPROCONPWOP5An	DMU_GP_HOST_PFPROCONPWOP5An, OTP set 3	$0CDA0_H$+n*4	8879
UCB0_25_DMU_GP_HOST_PFPROCONPWOP5Bn	DMU_GP_HOST_PFPROCONPWOP5Bn, OTP set 3	$0CDB0_H$+n*4	
UCB0_25_CONFIRMATION_Set_1	Confirmation CODE ORIGINAL Set_1, 32-bit	$0CFD0_H$	8879
UCB0_25_CONFIRMATION_Set_2	Confirmation CODE ORIGINAL Set_2, 32-bit	$0CFE0_H$	8879
UCB0_25_CONFIRMATION_Set_3	Confirmation CODE ORIGINAL Set_3, 32-bit	$0CFF0_H$	8880

48.4.3.12.1 DMU_GP_HOST_PFPROCONPWP0An, OTP set 1

UCB0_25_DMU_GP_HOST_PFPROCONPWP0An (n=0-3) Offset address: $0C800_H$+n*4

DMU_GP_HOST_PFPROCONPWP0An, OTP set 1 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONPWP0An															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONPWP0An															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWP0An	31:0	r	**DMU_GP_HOST_PFPROCONPWP0An register value**

48.4.3.12.2 DMU_GP_HOST_PFPROCONPWP0Bn, OTP set 1

UCB0_25_DMU_GP_HOST_PFPROCONPWP0Bn (n=0-3) Offset address: $0C810_H$+n*4

DMU_GP_HOST_PFPROCONPWP0Bn, OTP set 1 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONPWP0Bn															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONPWP0Bn															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWP0Bn	31:0	r	**DMU_GP_HOST_PFPROCONPWP0Bn register value**

48.4.3.12.3 DMU_GP_HOST_PFPROCONPWP1An, OTP set 1

UCB0_25_DMU_GP_HOST_PFPROCONPWP1An (n=0-3) Offset address: $0C820_H$+n*4

DMU_GP_HOST_PFPROCONPWP1An, OTP set 1 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONPWP1An															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONPWP1An															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWP1An	31:0	r	**DMU_GP_HOST_PFPROCONPWP1An register value**

48.4.3.12.4 DMU_GP_HOST_PFPROCONPWP1Bn, OTP set 1

UCB0_25_DMU_GP_HOST_PFPROCONPWP1Bn (n=0-3) Offset address: $0C830_H$+n*4

DMU_GP_HOST_PFPROCONPWP1Bn, OTP set 1 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONPWP1Bn															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONPWP1Bn															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWP1Bn	31:0	r	**DMU_GP_HOST_PFPROCONPWP1Bn register value**

48.4.3.12.5 DMU_GP_HOST_PFPROCONPWP2An, OTP set 1

UCB0_25_DMU_GP_HOST_PFPROCONPWP2An (n=0-3) Offset address: $0C840_H$+n*4

DMU_GP_HOST_PFPROCONPWP2An, OTP set 1 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONPWP2An															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONPWP2An															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWP2An	31:0	r	**DMU_GP_HOST_PFPROCONPWP2An register value**

48.4.3.12.6 DMU_GP_HOST_PFPROCONPWP2Bn, OTP set 1

UCB0_25_DMU_GP_HOST_PFPROCONPWP2Bn (n=0-3) Offset address: $0C850_H$+n*4

DMU_GP_HOST_PFPROCONPWP2Bn, OTP set 1 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONPWP2Bn															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONPWP2Bn															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWP2Bn	31:0	r	**DMU_GP_HOST_PFPROCONPWP2Bn register value**

48.4.3.12.7 DMU_GP_HOST_PFPROCONPWP3An, OTP set 1

UCB0_25_DMU_GP_HOST_PFPROCONPWP3An (n=0-3) Offset address: $0C860_H$+n*4

DMU_GP_HOST_PFPROCONPWP3An, OTP set 1 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONPWP3An															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONPWP3An															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWP3An	31:0	r	**DMU_GP_HOST_PFPROCONPWP3An register value**

48.4.3.12.8 DMU_GP_HOST_PFPROCONPWP3Bn, OTP set 1

UCB0_25_DMU_GP_HOST_PFPROCONPWP3Bn (n=0-3) Offset address: $0C870_H$+n*4

DMU_GP_HOST_PFPROCONPWP3Bn, OTP set 1 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONPWP3Bn															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONPWP3Bn															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWP3Bn	31:0	r	**DMU_GP_HOST_PFPROCONPWP3Bn register value**

48.4.3.12.9 DMU_GP_HOST_PFPROCONPWP4An, OTP set 1

UCB0_25_DMU_GP_HOST_PFPROCONPWP4An (n=0-3) Offset address: $0C880_H$+n*4

DMU_GP_HOST_PFPROCONPWP4An, OTP set 1 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONPWP4An															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONPWP4An															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWP4An	31:0	r	**DMU_GP_HOST_PFPROCONPWP4An register value**

48.4.3.12.10 DMU_GP_HOST_PFPROCONPWP4Bn, OTP set 1

UCB0_25_DMU_GP_HOST_PFPROCONPWP4Bn (n=0-3) Offset address: $0C890_H$+n*4

DMU_GP_HOST_PFPROCONPWP4Bn, OTP set 1 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONPWP4Bn															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONPWP4Bn															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWP4Bn	31:0	r	**DMU_GP_HOST_PFPROCONPWP4Bn register value**

48.4.3.12.11 DMU_GP_HOST_PFPROCONPWP5An, OTP set 1

UCB0_25_DMU_GP_HOST_PFPROCONPWP5An (n=0-3) Offset address: $0C8A0_H$+n*4

DMU_GP_HOST_PFPROCONPWP5An, OTP set 1 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONPWP5An															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONPWP5An															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWP5An	31:0	r	**DMU_GP_HOST_PFPROCONPWP5An register value**

48.4.3.12.12 DMU_GP_HOST_PFPROCONPWP5Bn, OTP set 1

UCB0_25_DMU_GP_HOST_PFPROCONPWP5Bn (n=0-3) Offset address: $0C8B0_H$+n*4

DMU_GP_HOST_PFPROCONPWP5Bn, OTP set 1 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONPWP5Bn															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONPWP5Bn															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWP5Bn	31:0	r	**DMU_GP_HOST_PFPROCONPWP5Bn register value**

48.4.3.12.13 DMU_GP_HOST_PFPROCONPWOP0An, OTP set 1

UCB0_25_DMU_GP_HOST_PFPROCONPWOP0An (n=0-3) Offset address: $0C900_H$+n*4

DMU_GP_HOST_PFPROCONPWOP0An, OTP set 1 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONPWOP0An															
r															

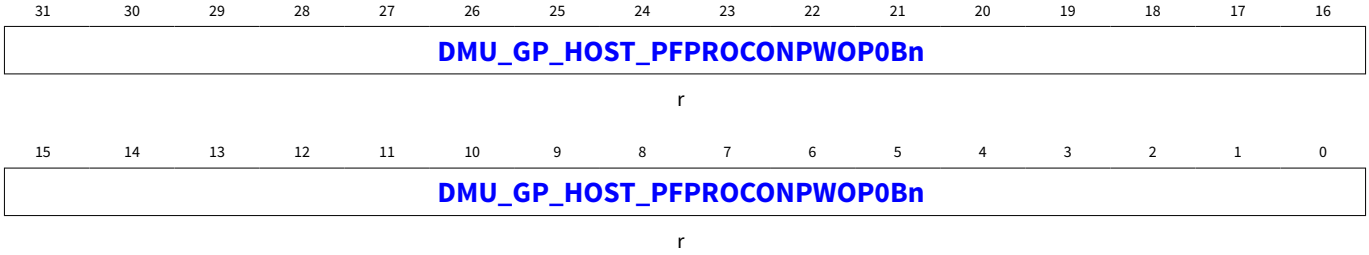
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONPWOP0An															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWOP0An	31:0	r	**DMU_GP_HOST_PFPROCONPWOP0An register value**

48.4.3.12.14 DMU_GP_HOST_PFPROCONPWOP0Bn, OTP set 1

UCB0_25_DMU_GP_HOST_PFPROCONPWOP0Bn (n=0-3) Offset address: $0C910_H$+n*4

DMU_GP_HOST_PFPROCONPWOP0Bn, OTP set 1 value: $0000\ 0000_H$

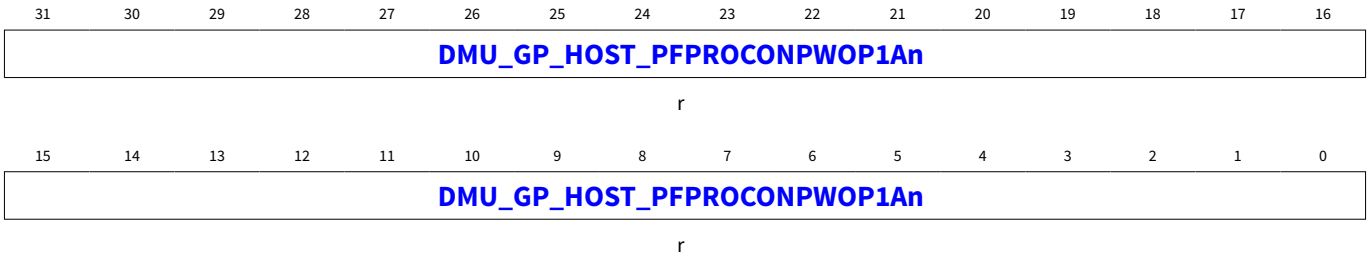


Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWOP0Bn	31:0	r	**DMU_GP_HOST_PFPROCONPWOP0Bn register value**

48.4.3.12.15 DMU_GP_HOST_PFPROCONPWOP1An, OTP set 1

UCB0_25_DMU_GP_HOST_PFPROCONPWOP1An (n=0-3) Offset address: $0C920_H$+n*4

DMU_GP_HOST_PFPROCONPWOP1An, OTP set 1 value: $0000\ 0000_H$

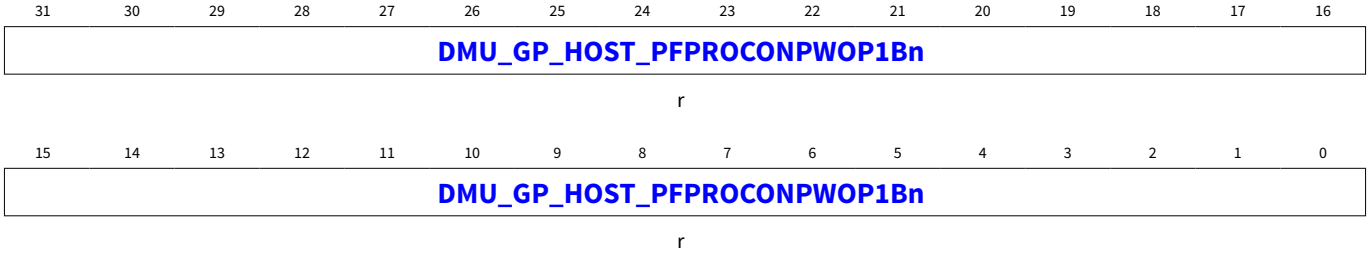


Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWOP1An	31:0	r	**DMU_GP_HOST_PFPROCONPWOP1An register value**

48.4.3.12.16 DMU_GP_HOST_PFPROCONPWOP1Bn, OTP set 1

UCB0_25_DMU_GP_HOST_PFPROCONPWOP1Bn (n=0-3) Offset address: $0C930_H$+n*4

DMU_GP_HOST_PFPROCONPWOP1Bn, OTP set 1 value: $0000\ 0000_H$

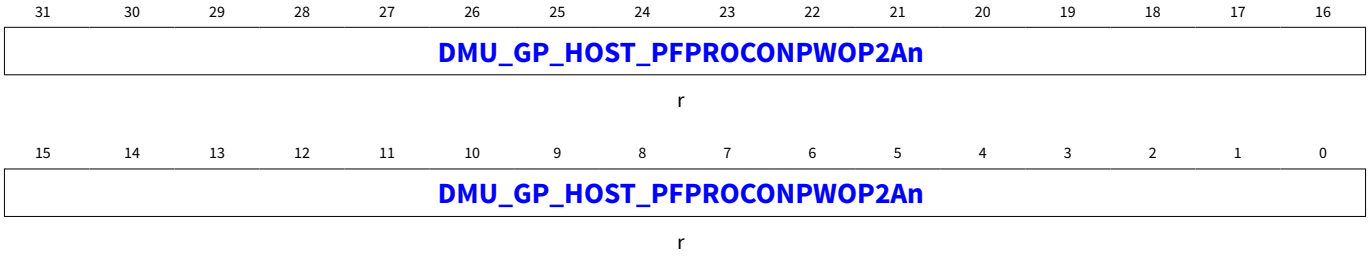


Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWOP1Bn	31:0	r	**DMU_GP_HOST_PFPROCONPWOP1Bn register value**

48.4.3.12.17 DMU_GP_HOST_PFPROCONPWOP2An, OTP set 1

UCB0_25_DMU_GP_HOST_PFPROCONPWOP2An (n=0-3) Offset address: $0C940_H$+n*4

DMU_GP_HOST_PFPROCONPWOP2An, OTP set 1 value: $0000\ 0000_H$



Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWOP2An	31:0	r	**DMU_GP_HOST_PFPROCONPWOP2An register value**

48.4.3.12.18 DMU_GP_HOST_PFPROCONPWOP2Bn, OTP set 1

UCB0_25_DMU_GP_HOST_PFPROCONPWOP2Bn (n=0-3) Offset address: $0C950_H$+n*4

DMU_GP_HOST_PFPROCONPWOP2Bn, OTP set 1 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONPWOP2Bn															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONPWOP2Bn															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWOP2Bn	31:0	r	**DMU_GP_HOST_PFPROCONPWOP2Bn register value**

48.4.3.12.19 DMU_GP_HOST_PFPROCONPWOP3An, OTP set 1

UCB0_25_DMU_GP_HOST_PFPROCONPWOP3An (n=0-3) Offset address: $0C960_H+n*4$

DMU_GP_HOST_PFPROCONPWOP3An, OTP set 1 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONPWOP3An															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONPWOP3An															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWOP3An	31:0	r	**DMU_GP_HOST_PFPROCONPWOP4An register value**

48.4.3.12.20 DMU_GP_HOST_PFPROCONPWOP3Bn, OTP set 1

UCB0_25_DMU_GP_HOST_PFPROCONPWOP3Bn (n=0-3) Offset address: $0C970_H+n*4$

DMU_GP_HOST_PFPROCONPWOP3Bn, OTP set 1 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONPWOP3Bn															
r															

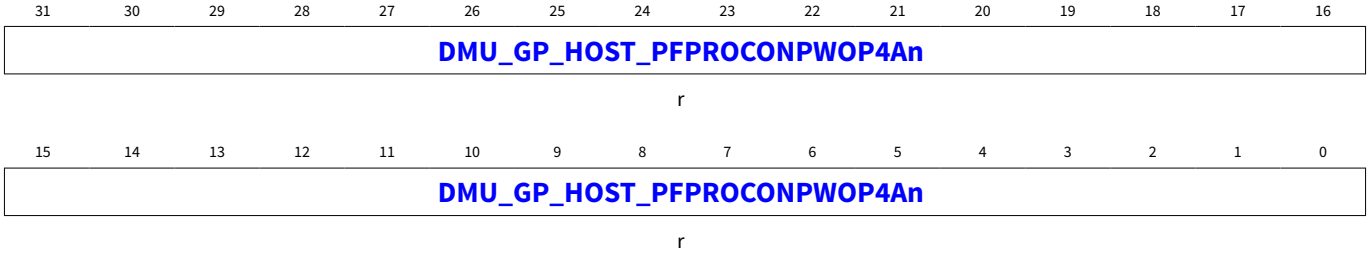
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONPWOP3Bn															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWOP3Bn	31:0	r	**DMU_GP_HOST_PFPROCONPWOP3Bn register value**

48.4.3.12.21 DMU_GP_HOST_PFPROCONPWOP4An, OTP set 1

UCB0_25_DMU_GP_HOST_PFPROCONPWOP4An (n=0-3) Offset address: $0C980_H$+n*4

DMU_GP_HOST_PFPROCONPWOP4An, OTP set 1 value: $0000\ 0000_H$

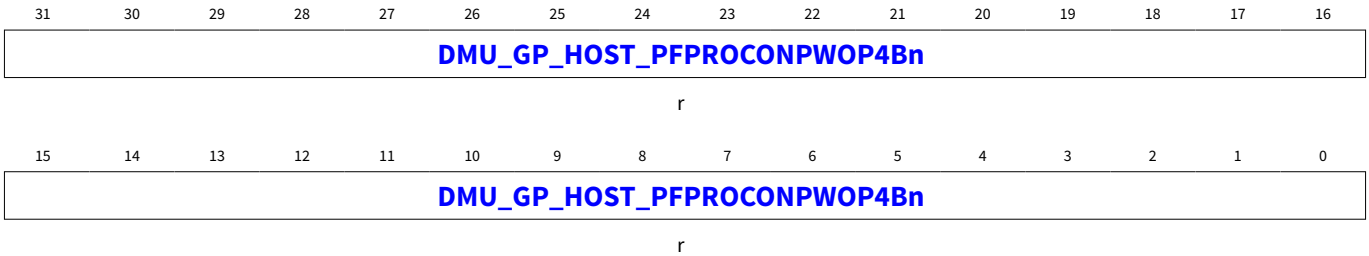


Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWOP4An	31:0	r	**DMU_GP_HOST_PFPROCONPWOP4An register value**

48.4.3.12.22 DMU_GP_HOST_PFPROCONPWOP4Bn, OTP set 1

UCB0_25_DMU_GP_HOST_PFPROCONPWOP4Bn (n=0-3) Offset address: $0C990_H$+n*4

DMU_GP_HOST_PFPROCONPWOP4Bn, OTP set 1 value: $0000\ 0000_H$



Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWOP4Bn	31:0	r	**DMU_GP_HOST_PFPROCONPWOP4Bn register value**

48.4.3.12.23 DMU_GP_HOST_PFPROCONPWOP5An, OTP set 1

UCB0_25_DMU_GP_HOST_PFPROCONPWOP5An (n=0-3) Offset address: $0C9A0_H$+n*4

DMU_GP_HOST_PFPROCONPWOP5An, OTP set 1 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONPWOP5An															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONPWOP5An															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWOP5An	31:0	r	**DMU_GP_HOST_PFPROCONPWOP5An register value**

48.4.3.12.24 DMU_GP_HOST_PFPROCONPWOP5Bn, OTP set 1

UCB0_25_DMU_GP_HOST_PFPROCONPWOP5Bn (n=0-3) Offset address: $0C9B0_H$+n*4

DMU_GP_HOST_PFPROCONPWOP5Bn, OTP set 1 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONPWOP5Bn															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONPWOP5Bn															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWOP5Bn	31:0	r	**DMU_GP_HOST_PFPROCONPWOP5Bn register value**

48.4.3.12.25 DMU_GP_HOST_PFPROCONPWP0An, OTP set 2

UCB0_25_DMU_GP_HOST_PFPROCONPWP0An (n=0-3) Offset address: $0CA00_H$+n*4

DMU_GP_HOST_PFPROCONPWP0An, OTP set 2 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONPWP0An															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONPWP0An															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWP0An	31:0	r	**DMU_GP_HOST_PFPROCONPWP0An register value**

48.4.3.12.26 DMU_GP_HOST_PFPROCONPWP0Bn, OTP set 2

UCB0_25_DMU_GP_HOST_PFPROCONPWP0Bn (n=0-3) Offset address: $0CA10_H+n*4$

DMU_GP_HOST_PFPROCONPWP0Bn, OTP set 2 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONPWP0Bn															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONPWP0Bn															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWP0Bn	31:0	r	**DMU_GP_HOST_PFPROCONPWP0Bn register value**

48.4.3.12.27 DMU_GP_HOST_PFPROCONPWP1An, OTP set 2

UCB0_25_DMU_GP_HOST_PFPROCONPWP1An (n=0-3) Offset address: $0CA20_H+n*4$

DMU_GP_HOST_PFPROCONPWP1An, OTP set 2 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONPWP1An															
r															

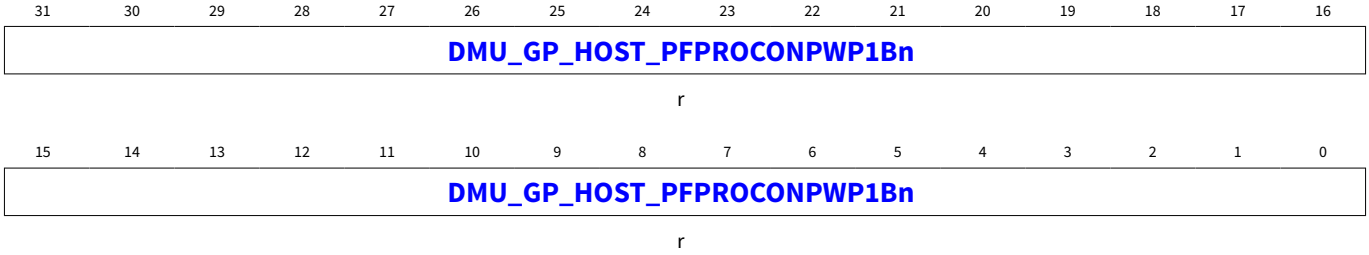
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONPWP1An															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWP1An	31:0	r	**DMU_GP_HOST_PFPROCONPWP1An register value**

48.4.3.12.28 DMU_GP_HOST_PFPROCONPWP1Bn, OTP set 2

UCB0_25_DMU_GP_HOST_PFPROCONPWP1Bn (n=0-3) Offset address: $0CA30_H+n*4$

DMU_GP_HOST_PFPROCONPWP1Bn, OTP set 2 value: $0000\ 0000_H$

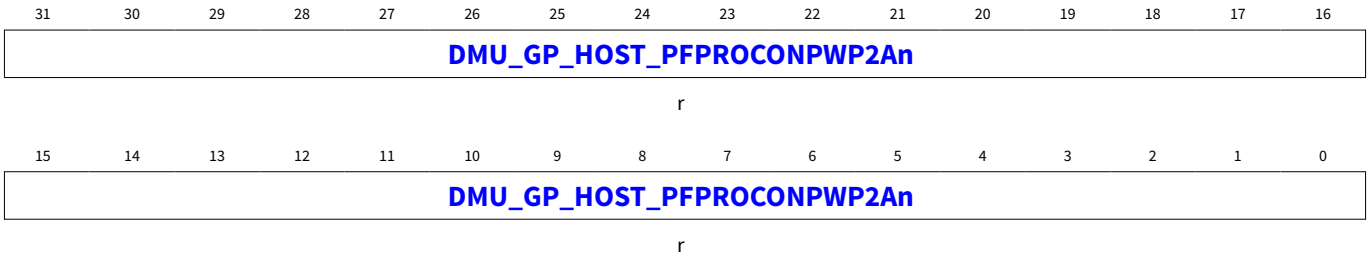


Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWP1Bn	31:0	r	**DMU_GP_HOST_PFPROCONPWP1Bn register value**

48.4.3.12.29 DMU_GP_HOST_PFPROCONPWP2An, OTP set 2

UCB0_25_DMU_GP_HOST_PFPROCONPWP2An (n=0-3) Offset address: $0CA40_H+n*4$

DMU_GP_HOST_PFPROCONPWP2An, OTP set 2 value: $0000\ 0000_H$



Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWP2An	31:0	r	**DMU_GP_HOST_PFPROCONPWP2An register value**

48.4.3.12.30 DMU_GP_HOST_PFPROCONPWP2Bn, OTP set 2

UCB0_25_DMU_GP_HOST_PFPROCONPWP2Bn (n=0-3) Offset address: $0CA50_H+n*4$

DMU_GP_HOST_PFPROCONPWP2Bn, OTP set 2 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONPWP2Bn															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONPWP2Bn															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWP2Bn	31:0	r	**DMU_GP_HOST_PFPROCONPWP2Bn register value**

48.4.3.12.31 DMU_GP_HOST_PFPROCONPWP3An, OTP set 2

UCB0_25_DMU_GP_HOST_PFPROCONPWP3An (n=0-3) Offset address: $0CA60_H+n*4$

DMU_GP_HOST_PFPROCONPWP3An, OTP set 2 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONPWP3An															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONPWP3An															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWP3An	31:0	r	**DMU_GP_HOST_PFPROCONPWP3An register value**

48.4.3.12.32 DMU_GP_HOST_PFPROCONPWP3Bn, OTP set 2

UCB0_25_DMU_GP_HOST_PFPROCONPWP3Bn (n=0-3) Offset address: $0CA70_H+n*4$

DMU_GP_HOST_PFPROCONPWP3Bn, OTP set 2 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONPWP3Bn															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONPWP3Bn															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWP3Bn	31:0	r	**DMU_GP_HOST_PFPROCONPWP3Bn register value**

48.4.3.12.33 DMU_GP_HOST_PFPROCONPWP4An, OTP set 2

UCB0_25_DMU_GP_HOST_PFPROCONPWP4An (n=0-3) Offset address: $0CA80_H+n*4$

DMU_GP_HOST_PFPROCONPWP4An, OTP set 2 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONPWP4An															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONPWP4An															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWP4An	31:0	r	**DMU_GP_HOST_PFPROCONPWP4An register value**

48.4.3.12.34 DMU_GP_HOST_PFPROCONPWP4Bn, OTP set 2

UCB0_25_DMU_GP_HOST_PFPROCONPWP4Bn (n=0-3) Offset address: $0CA90_H+n*4$

DMU_GP_HOST_PFPROCONPWP4Bn, OTP set 2 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONPWP4Bn															
r															

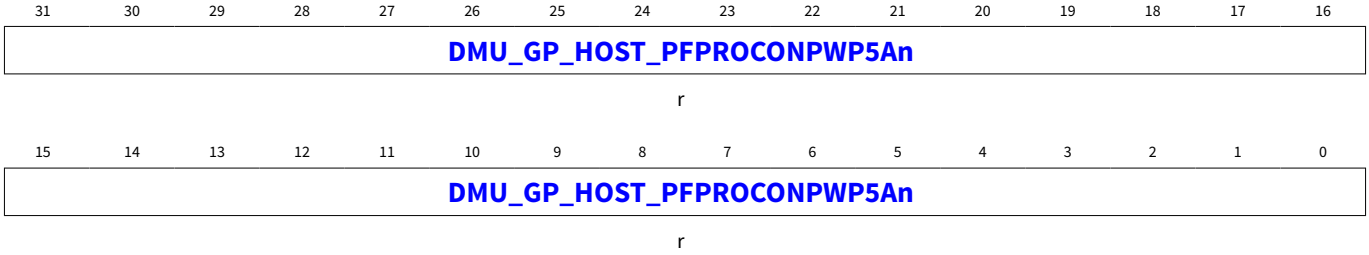
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONPWP4Bn															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWP4Bn	31:0	r	**DMU_GP_HOST_PFPROCONPWP4Bn register value**

48.4.3.12.35 DMU_GP_HOST_PFPROCONPWP5An, OTP set 2

UCB0_25_DMU_GP_HOST_PFPROCONPWP5An (n=0-3) Offset address: $0CAA0_H+n*4$

DMU_GP_HOST_PFPROCONPWP5An, OTP set 2 value: $0000\ 0000_H$

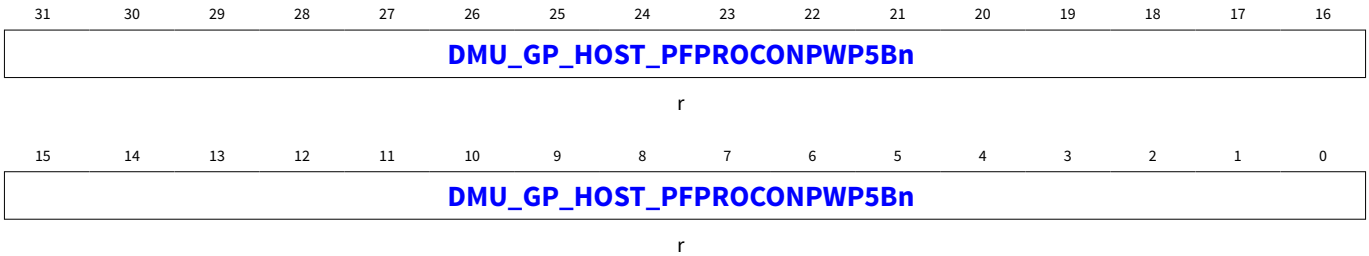


Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWP5An	31:0	r	**DMU_GP_HOST_PFPROCONPWP5An register value**

48.4.3.12.36 DMU_GP_HOST_PFPROCONPWP5Bn, OTP set 2

UCB0_25_DMU_GP_HOST_PFPROCONPWP5Bn (n=0-3) Offset address: $0CAB0_H+n*4$

DMU_GP_HOST_PFPROCONPWP5Bn, OTP set 2 value: $0000\ 0000_H$



Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWP5Bn	31:0	r	**DMU_GP_HOST_PFPROCONPWP5Bn register value**

48.4.3.12.37 DMU_GP_HOST_PFPROCONPWOP0An, OTP set 2

UCB0_25_DMU_GP_HOST_PFPROCONPWOP0An (n=0-3) Offset address: $0CB00_H$+n*4

DMU_GP_HOST_PFPROCONPWOP0An, OTP set 2 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONPWOP0An															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONPWOP0An															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWOP0An	31:0	r	**DMU_GP_HOST_PFPROCONPWOP0An register value**

48.4.3.12.38 DMU_GP_HOST_PFPROCONPWOP0Bn, OTP set 2

UCB0_25_DMU_GP_HOST_PFPROCONPWOP0Bn (n=0-3) Offset address: $0CB10_H$+n*4

DMU_GP_HOST_PFPROCONPWOP0Bn, OTP set 2 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONPWOP0Bn															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONPWOP0Bn															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWOP0Bn	31:0	r	**DMU_GP_HOST_PFPROCONPWOP0Bn register value**

48.4.3.12.39 DMU_GP_HOST_PFPROCONPWOP1An, OTP set 2

UCB0_25_DMU_GP_HOST_PFPROCONPWOP1An (n=0-3) Offset address: $0CB20_H$+n*4

DMU_GP_HOST_PFPROCONPWOP1An, OTP set 2 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONPWOP1An															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONPWOP1An															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWOP1An	31:0	r	**DMU_GP_HOST_PFPROCONPWOP1An register value**

48.4.3.12.40 DMU_GP_HOST_PFPROCONPWOP1Bn, OTP set 2

UCB0_25_DMU_GP_HOST_PFPROCONPWOP1Bn (n=0-3) Offset address: $0CB30_H+n*4$

DMU_GP_HOST_PFPROCONPWOP1Bn, OTP set 2 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONPWOP1Bn															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONPWOP1Bn															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWOP1Bn	31:0	r	**DMU_GP_HOST_PFPROCONPWOP1Bn register value**

48.4.3.12.41 DMU_GP_HOST_PFPROCONPWOP2An, OTP set 2

UCB0_25_DMU_GP_HOST_PFPROCONPWOP2An (n=0-3) Offset address: $0CB40_H+n*4$

DMU_GP_HOST_PFPROCONPWOP2An, OTP set 2 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONPWOP2An															
r															

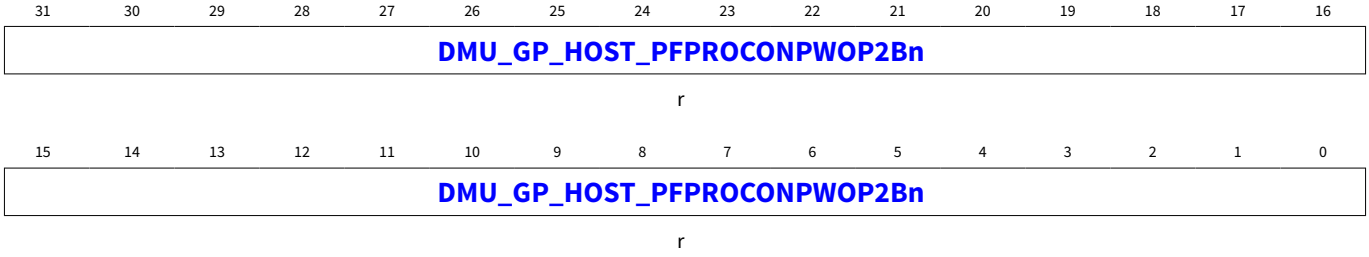
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONPWOP2An															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWOP2An	31:0	r	**DMU_GP_HOST_PFPROCONPWOP2An register value**

48.4.3.12.42 DMU_GP_HOST_PFPROCONPWOP2Bn, OTP set 2

UCB0_25_DMU_GP_HOST_PFPROCONPWOP2Bn (n=0-3) Offset address: $0CB50_H+n*4$

DMU_GP_HOST_PFPROCONPWOP2Bn, OTP set 2 value: $0000\ 0000_H$

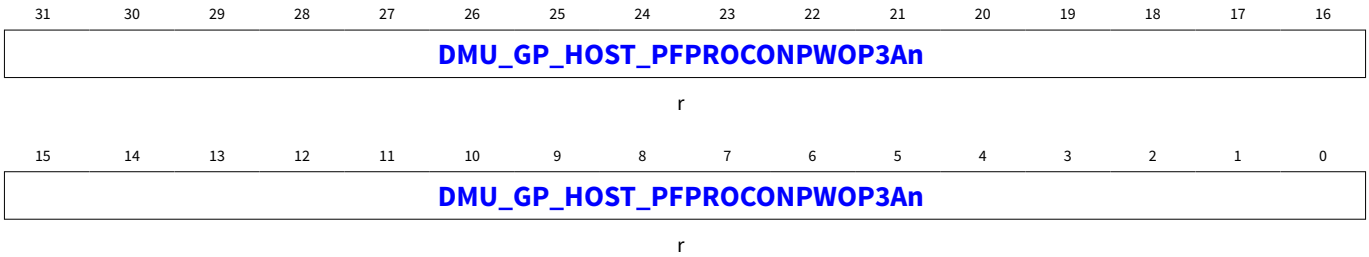


Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWOP2Bn	31:0	r	**DMU_GP_HOST_PFPROCONPWOP2Bn register value**

48.4.3.12.43 DMU_GP_HOST_PFPROCONPWOP3An, OTP set 2

UCB0_25_DMU_GP_HOST_PFPROCONPWOP3An (n=0-3) Offset address: $0CB60_H+n*4$

DMU_GP_HOST_PFPROCONPWOP3An, OTP set 2 value: $0000\ 0000_H$



Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWOP3An	31:0	r	**DMU_GP_HOST_PFPROCONPWOP3An register value**

48.4.3.12.44 DMU_GP_HOST_PFPROCONPWOP3Bn, OTP set 2

UCB0_25_DMU_GP_HOST_PFPROCONPWOP3Bn (n=0-3) Offset address: $0CB70_H$+n*4

DMU_GP_HOST_PFPROCONPWOP3Bn, OTP set 2 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONPWOP3Bn															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONPWOP3Bn															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWOP3Bn	31:0	r	**DMU_GP_HOST_PFPROCONPWOP3Bn register value**

48.4.3.12.45 DMU_GP_HOST_PFPROCONPWOP4An, OTP set 2

UCB0_25_DMU_GP_HOST_PFPROCONPWOP4An (n=0-3) Offset address: $0CB80_H$+n*4

DMU_GP_HOST_PFPROCONPWOP4An, OTP set 2 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONPWOP4An															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONPWOP4An															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWOP4An	31:0	r	**DMU_GP_HOST_PFPROCONPWOP4An register value**

48.4.3.12.46 DMU_GP_HOST_PFPROCONPWOP4Bn, OTP set 2

UCB0_25_DMU_GP_HOST_PFPROCONPWOP4Bn (n=0-3) Offset address: $0CB90_H$+n*4

DMU_GP_HOST_PFPROCONPWOP4Bn, OTP set 2 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONPWOP4Bn															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONPWOP4Bn															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWOP4Bn	31:0	r	**DMU_GP_HOST_PFPROCONPWOP4Bn register value**

48.4.3.12.47 DMU_GP_HOST_PFPROCONPWOP5An, OTP set 2

UCB0_25_DMU_GP_HOST_PFPROCONPWOP5An (n=0-3) Offset address: $0CBA0_H+n*4$

DMU_GP_HOST_PFPROCONPWOP5An, OTP set 2 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONPWOP5An															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONPWOP5An															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWOP5An	31:0	r	**DMU_GP_HOST_PFPROCONPWOP5An register value**

48.4.3.12.48 DMU_GP_HOST_PFPROCONPWOP5Bn, OTP set 2

UCB0_25_DMU_GP_HOST_PFPROCONPWOP5Bn (n=0-3) Offset address: $0CBB0_H+n*4$

DMU_GP_HOST_PFPROCONPWOP5Bn, OTP set 2 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONPWOP5Bn															
r															

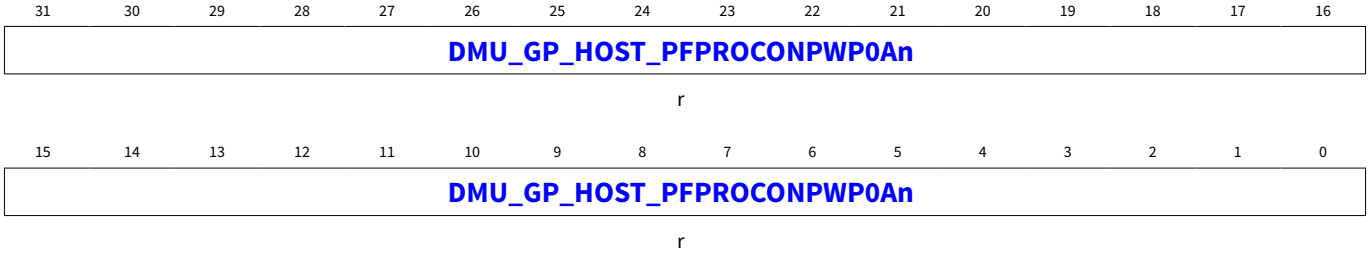
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONPWOP5Bn															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWOP5Bn	31:0	r	**DMU_GP_HOST_PFPROCONPWOP5Bn register value**

48.4.3.12.49 DMU_GP_HOST_PFPROCONPWP0An, OTP set 3

UCB0_25_DMU_GP_HOST_PFPROCONPWP0An (n=0-3) Offset address: $0CC00_H+n*4$

DMU_GP_HOST_PFPROCONPWP0An, OTP set 3 value: $0000\ 0000_H$

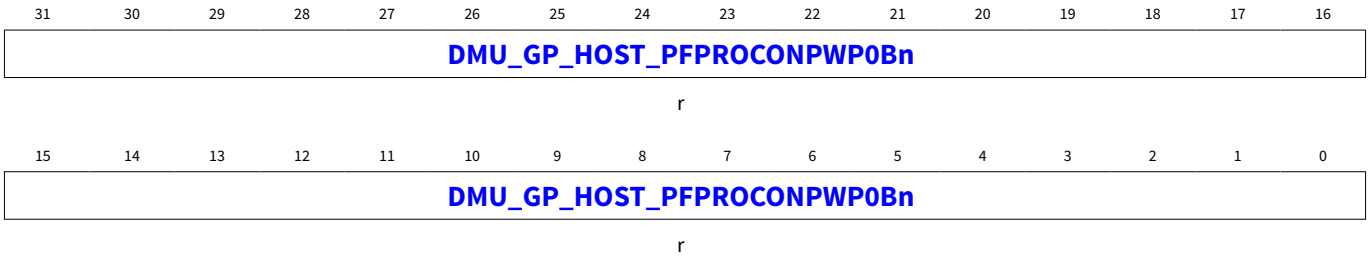


Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWP0An	31:0	r	**DMU_GP_HOST_PFPROCONPWP0An register value**

48.4.3.12.50 DMU_GP_HOST_PFPROCONPWP0Bn, OTP set 3

UCB0_25_DMU_GP_HOST_PFPROCONPWP0Bn (n=0-3) Offset address: $0CC10_H+n*4$

DMU_GP_HOST_PFPROCONPWP0Bn, OTP set 3 value: $0000\ 0000_H$



Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWP0Bn	31:0	r	**DMU_GP_HOST_PFPROCONPWP0Bn register value**

48.4.3.12.51 DMU_GP_HOST_PFPROCONPWP1An, OTP set 3

UCB0_25_DMU_GP_HOST_PFPROCONPWP1An (n=0-3) Offset address: $0CC20_H+n*4$

DMU_GP_HOST_PFPROCONPWP1An, OTP set 3 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONPWP1An															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONPWP1An															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWP1An	31:0	r	**DMU_GP_HOST_PFPROCONPWP1An register value**

48.4.3.12.52 DMU_GP_HOST_PFPROCONPWP1Bn, OTP set 3

UCB0_25_DMU_GP_HOST_PFPROCONPWP1Bn (n=0-3) Offset address: $0CC30_H+n*4$

DMU_GP_HOST_PFPROCONPWP1Bn, OTP set 3 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONPWP1Bn															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONPWP1Bn															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWP1Bn	31:0	r	**DMU_GP_HOST_PFPROCONPWP1Bn register value**

48.4.3.12.53 DMU_GP_HOST_PFPROCONPWP2An, OTP set 3

UCB0_25_DMU_GP_HOST_PFPROCONPWP2An (n=0-3) Offset address: $0CC40_H+n*4$

DMU_GP_HOST_PFPROCONPWP2An, OTP set 3 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONPWP2An															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONPWP2An															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWP2An	31:0	r	**DMU_GP_HOST_PFPROCONPWP2An register value**

48.4.3.12.54 DMU_GP_HOST_PFPROCONPWP2Bn, OTP set 3

UCB0_25_DMU_GP_HOST_PFPROCONPWP2Bn (n=0-3) Offset address: $0CC50_H+n*4$

DMU_GP_HOST_PFPROCONPWP2Bn, OTP set 3 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONPWP2Bn															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONPWP2Bn															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWP2Bn	31:0	r	**DMU_GP_HOST_PFPROCONPWP2Bn register value**

48.4.3.12.55 DMU_GP_HOST_PFPROCONPWP3An, OTP set 3

UCB0_25_DMU_GP_HOST_PFPROCONPWP3An (n=0-3) Offset address: $0CC60_H+n*4$

DMU_GP_HOST_PFPROCONPWP3An, OTP set 3 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONPWP3An															
r															

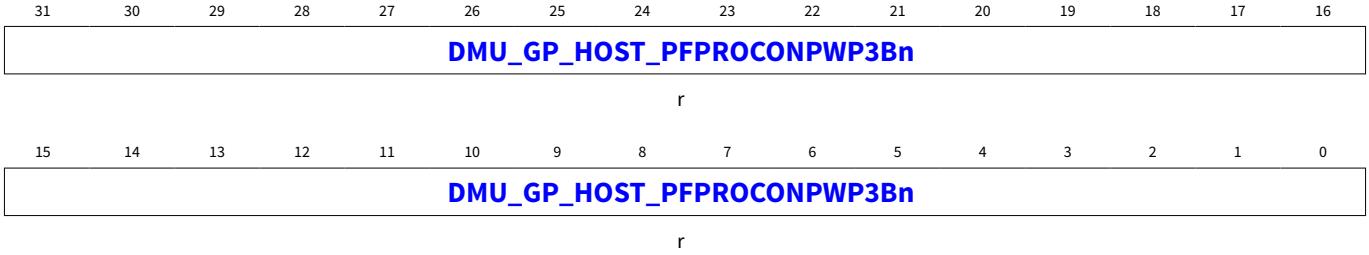
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONPWP3An															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWP3An	31:0	r	**DMU_GP_HOST_PFPROCONPWP3An register value**

48.4.3.12.56 DMU_GP_HOST_PFPROCONPWP3Bn, OTP set 3

UCB0_25_DMU_GP_HOST_PFPROCONPWP3Bn (n=0-3) Offset address: $0CC70_H+n*4$

DMU_GP_HOST_PFPROCONPWP3Bn, OTP set 3 value: $0000\ 0000_H$

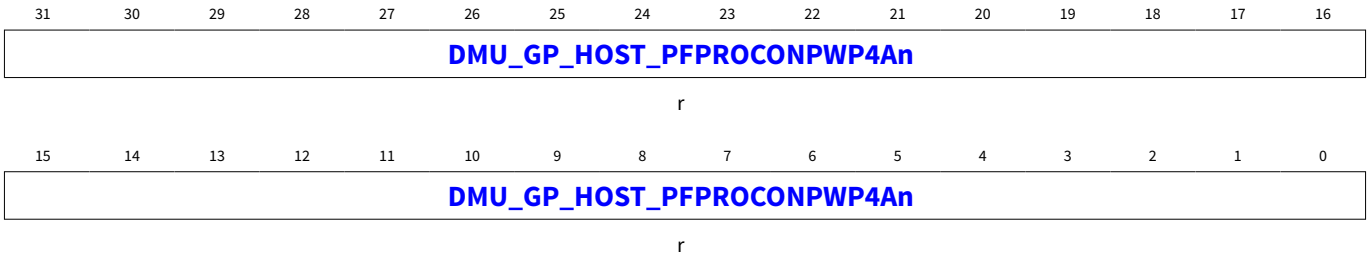


Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWP3Bn	31:0	r	**DMU_GP_HOST_PFPROCONPWP3Bn register value**

48.4.3.12.57 DMU_GP_HOST_PFPROCONPWP4An, OTP set 3

UCB0_25_DMU_GP_HOST_PFPROCONPWP4An (n=0-3) Offset address: $0CC80_H+n*4$

DMU_GP_HOST_PFPROCONPWP4An, OTP set 3 value: $0000\ 0000_H$



Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWP4An	31:0	r	**DMU_GP_HOST_PFPROCONPWP4An register value**

48.4.3.12.58 DMU_GP_HOST_PFPROCONPWP5An, OTP set 3

UCB0_25_DMU_GP_HOST_PFPROCONPWP5An (n=0-3) Offset address: $0CCA0_H+n*4$

DMU_GP_HOST_PFPROCONPWP5An, OTP set 3 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONPWP5An															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONPWP5An															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWP5An	31:0	r	**DMU_GP_HOST_PFPROCONPWP5An register value**

48.4.3.12.59 DMU_GP_HOST_PFPROCONPWP5Bn, OTP set 3

UCB0_25_DMU_GP_HOST_PFPROCONPWP5Bn (n=0-3) Offset address: $0CCB0_H+n*4$

DMU_GP_HOST_PFPROCONPWP5Bn, OTP set 3 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONPWP5Bn															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONPWP5Bn															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWP5Bn	31:0	r	**DMU_GP_HOST_PFPROCONPWP5Bn register value**

48.4.3.12.60 DMU_GP_HOST_PFPROCONPWOP0An, OTP set 3

UCB0_25_DMU_GP_HOST_PFPROCONPWOP0An (n=0-3) Offset address: $0CD00_H+n*4$

DMU_GP_HOST_PFPROCONPWOP0An, OTP set 3 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONPWOP0An															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONPWOP0An															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWOP0An	31:0	r	**DMU_GP_HOST_PFPROCONPWOP0An register value**

48.4.3.12.61 DMU_GP_HOST_PFPROCONPWOP0Bn, OTP set 3

UCB0_25_DMU_GP_HOST_PFPROCONPWOP0Bn (n=0-3) Offset address: $0CD10_H+n*4$

DMU_GP_HOST_PFPROCONPWOP0Bn, OTP set 3 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONPWOP0Bn															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONPWOP0Bn															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWOP0Bn	31:0	r	**DMU_GP_HOST_PFPROCONPWOP0Bn register value**

48.4.3.12.62 DMU_GP_HOST_PFPROCONPWOP1An, OTP set 3

UCB0_25_DMU_GP_HOST_PFPROCONPWOP1An (n=0-3) Offset address: $0CD20_H+n*4$

DMU_GP_HOST_PFPROCONPWOP1An, OTP set 3 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONPWOP1An															
r															

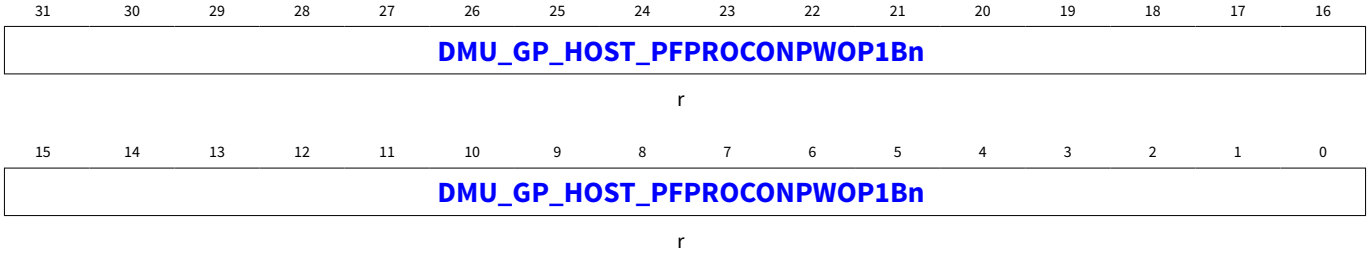
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONPWOP1An															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWOP1An	31:0	r	**DMU_GP_HOST_PFPROCONPWOP1An register value**

48.4.3.12.63 DMU_GP_HOST_PFPROCONPWOP1Bn, OTP set 3

UCB0_25_DMU_GP_HOST_PFPROCONPWOP1Bn (n=0-3) Offset address: $0CD30_H+n*4$

DMU_GP_HOST_PFPROCONPWOP1Bn, OTP set 3 value: $0000\ 0000_H$

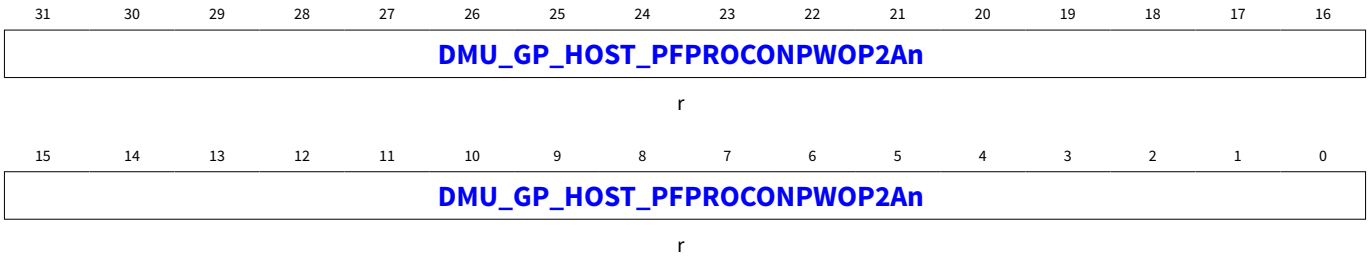


Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWOP1Bn	31:0	r	**DMU_GP_HOST_PFPROCONPWOP1Bn register value**

48.4.3.12.64 DMU_GP_HOST_PFPROCONPWOP2An, OTP set 3

UCB0_25_DMU_GP_HOST_PFPROCONPWOP2An (n=0-3) Offset address: $0CD40_H+n*4$

DMU_GP_HOST_PFPROCONPWOP2An, OTP set 3 value: $0000\ 0000_H$

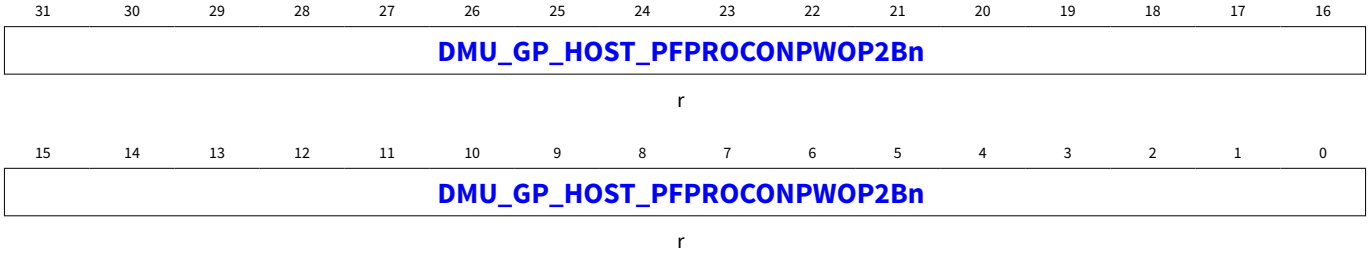


Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWOP2An	31:0	r	**DMU_GP_HOST_PFPROCONPWOP2An register value**

48.4.3.12.65 DMU_GP_HOST_PFPROCONPWOP2Bn, OTP set 3

UCB0_25_DMU_GP_HOST_PFPROCONPWOP2Bn (n=0-3) Offset address: $0CD50_H$+n*4

DMU_GP_HOST_PFPROCONPWOP2Bn, OTP set 3 value: $0000\ 0000_H$

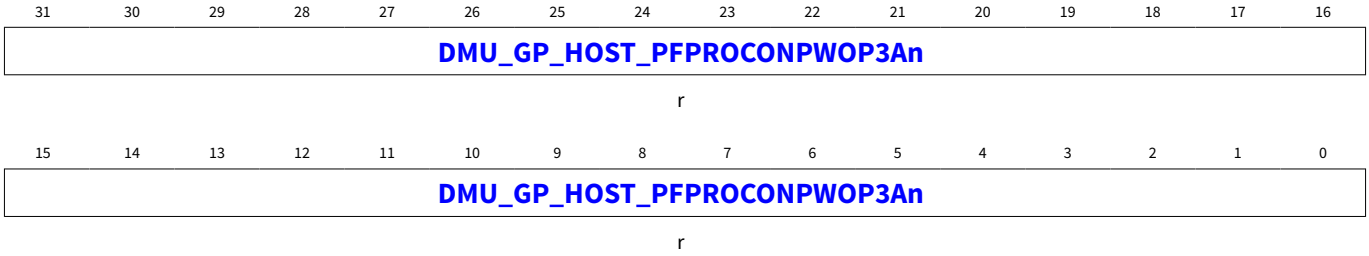


Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWOP2Bn	31:0	r	**DMU_GP_HOST_PFPROCONPWOP2Bn register value**

48.4.3.12.66 DMU_GP_HOST_PFPROCONPWOP3An, OTP set 3

UCB0_25_DMU_GP_HOST_PFPROCONPWOP3An (n=0-3) Offset address: $0CD60_H$+n*4

DMU_GP_HOST_PFPROCONPWOP3An, OTP set 3 value: $0000\ 0000_H$



Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWOP3An	31:0	r	**DMU_GP_HOST_PFPROCONPWOP3An register value**

48.4.3.12.67 DMU_GP_HOST_PFPROCONPWOP3Bn, OTP set 3

UCB0_25_DMU_GP_HOST_PFPROCONPWOP3Bn (n=0-3) Offset address: $0CD70_H$+n*4

DMU_GP_HOST_PFPROCONPWOP3Bn, OTP set 3 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONPWOP3Bn															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONPWOP3Bn															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWOP3Bn	31:0	r	**DMU_GP_HOST_PFPROCONPWOP3Bn register value**

48.4.3.12.68 DMU_GP_HOST_PFPROCONPWOP4An, OTP set 3

UCB0_25_DMU_GP_HOST_PFPROCONPWOP4An (n=0-3) Offset address: $0CD80_H+n*4$

DMU_GP_HOST_PFPROCONPWOP4An, OTP set 3 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONPWOP4An															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONPWOP4An															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWOP4An	31:0	r	**DMU_GP_HOST_PFPROCONPWOP4An register value**

48.4.3.12.69 DMU_GP_HOST_PFPROCONPWOP4Bn, OTP set 3

UCB0_25_DMU_GP_HOST_PFPROCONPWOP4Bn (n=0-3) Offset address: $0CD90_H+n*4$

DMU_GP_HOST_PFPROCONPWOP4Bn, OTP set 3 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONPWOP4Bn															
r															

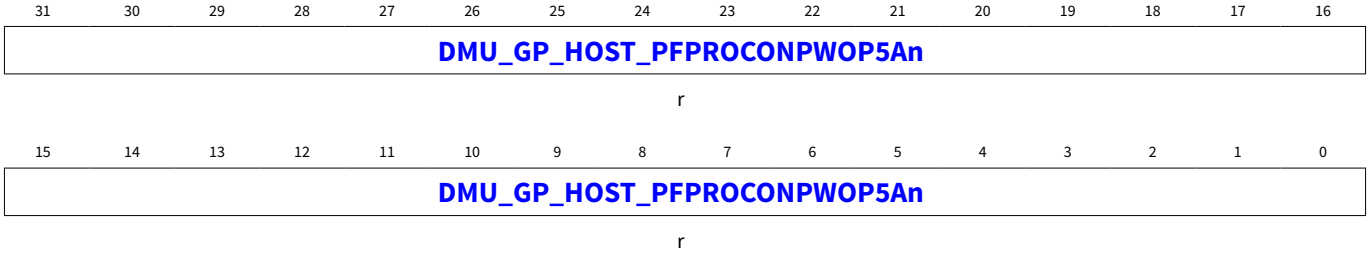
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONPWOP4Bn															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWOP4Bn	31:0	r	**DMU_GP_HOST_PFPROCONPWOP4Bn register value**

48.4.3.12.70 DMU_GP_HOST_PFPROCONPWOP5An, OTP set 3

UCB0_25_DMU_GP_HOST_PFPROCONPWOP5An (n=0-3) Offset address: $0CDA0_H+n*4$

DMU_GP_HOST_PFPROCONPWOP5An, OTP set 3 value: $0000\ 0000_H$

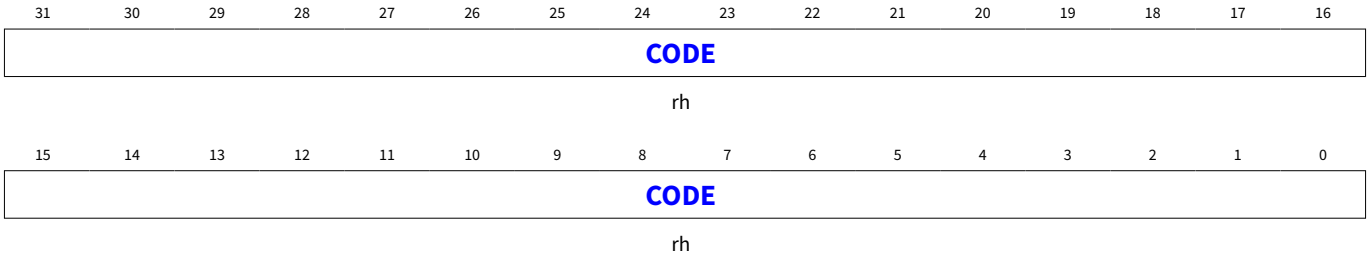


Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWOP5An	31:0	r	**DMU_GP_HOST_PFPROCONPWOP5An register value**

48.4.3.12.71 Confirmation CODE ORIGINAL Set_1, 32-bit

UCB0_25_CONFIRMATION_Set_1 Offset address: $0CFD0_H$

Confirmation CODE ORIGINAL Set_1, 32-bit value: $4321\ 1234_H$



Field	Bits	Type	Description
CODE	31:0	rh	**Confirmation Code** 43211234_H Confirmation state is UNLOCKED. $57B5327F_H$ Confirmation state is CONFIRMED.

48.4.3.12.72 Confirmation CODE ORIGINAL Set_2, 32-bit

UCB0_25_CONFIRMATION_Set_2 Offset address: $0CFE0_H$

Confirmation CODE ORIGINAL Set_2, 32-bit value: $4321\ 1234_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CODE															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CODE															
rh															

Field	Bits	Type	Description
CODE	31:0	rh	**Confirmation Code** 43211234_H Confirmation state is UNLOCKED. $57B5327F_H$ Confirmation state is CONFIRMED.

48.4.3.12.73 Confirmation CODE ORIGINAL Set_3, 32-bit

UCB0_25_CONFIRMATION_Set_3 Offset address: $0CFF0_H$

Confirmation CODE ORIGINAL Set_3, 32-bit value: $4321\ 1234_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CODE															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CODE															
rh															

Field	Bits	Type	Description
CODE	31:0	rh	**Confirmation Code** 43211234_H Confirmation state is UNLOCKED. $57B5327F_H$ Confirmation state is CONFIRMED.

48.4.3.13 Register overview - UCB0_26 (ascending offset address)

Table 2538 Register overview - UCB0_26 (ascending offset address)

Short name	Long name	Offset address	See
UCB0_26_DMU_GP_HOST_PFPROCONPWP0An	DMU_GP_HOST_PFPROCONPWP0An, OTP set 4	$0D000_H$+n*4	8885
UCB0_26_DMU_GP_HOST_PFPROCONPWP0Bn	DMU_GP_HOST_PFPROCONPWP0Bn, OTP set 4	$0D010_H$+n*4	8886
UCB0_26_DMU_GP_HOST_PFPROCONPWP1An	DMU_GP_HOST_PFPROCONPWP1An, OTP set 4	$0D020_H$+n*4	8886

(table continues...)

Table 2538 (continued) Register overview - UCB0_26 (ascending offset address)

Short name	Long name	Offset address	See
UCB0_26_DMU_GP_HOST_PFPROCONPWP1Bn	DMU_GP_HOST_PFPROCONPWP1Bn, OTP set 4	$0D030_H$+n*4	8887
UCB0_26_DMU_GP_HOST_PFPROCONPWP2An	DMU_GP_HOST_PFPROCONPWP2An, OTP set 4	$0D040_H$+n*4	8887
UCB0_26_DMU_GP_HOST_PFPROCONPWP2Bn	DMU_GP_HOST_PFPROCONPWP2Bn, OTP set 4	$0D050_H$+n*4	8888
UCB0_26_DMU_GP_HOST_PFPROCONPWP3An	DMU_GP_HOST_PFPROCONPWP3An, OTP set 4	$0D060_H$+n*4	8888
UCB0_26_DMU_GP_HOST_PFPROCONPWP3Bn	DMU_GP_HOST_PFPROCONPWP3Bn, OTP set 4	$0D070_H$+n*4	8888
UCB0_26_DMU_GP_HOST_PFPROCONPWP4An	DMU_GP_HOST_PFPROCONPWP4An, OTP set 4	$0D080_H$+n*4	8889
UCB0_26_DMU_GP_HOST_PFPROCONPWP4Bn	DMU_GP_HOST_PFPROCONPWP4Bn, OTP set 4	$0D090_H$+n*4	8889
UCB0_26_DMU_GP_HOST_PFPROCONPWP5An	DMU_GP_HOST_PFPROCONPWP5An, OTP set 4	$0D0A0_H$+n*4	8890
UCB0_26_DMU_GP_HOST_PFPROCONPWP5Bn	DMU_GP_HOST_PFPROCONPWP5Bn, OTP set 4	$0D0B0_H$+n*4	8890
UCB0_26_DMU_GP_HOST_PFPROCONPWOP0An	DMU_GP_HOST_PFPROCONPWOP0An, OTP set 4	$0D100_H$+n*4	8891
UCB0_26_DMU_GP_HOST_PFPROCONPWOP0Bn	DMU_GP_HOST_PFPROCONPWOP0Bn, OTP set 4	$0D110_H$+n*4	8891
UCB0_26_DMU_GP_HOST_PFPROCONPWOP1An	DMU_GP_HOST_PFPROCONPWOP1An, OTP set 4	$0D120_H$+n*4	8891
UCB0_26_DMU_GP_HOST_PFPROCONPWOP1Bn	DMU_GP_HOST_PFPROCONPWOP1Bn, OTP set 4	$0D130_H$+n*4	8892
UCB0_26_DMU_GP_HOST_PFPROCONPWOP2An	DMU_GP_HOST_PFPROCONPWOP2An, OTP set 4	$0D140_H$+n*4	8892
UCB0_26_DMU_GP_HOST_PFPROCONPWOP2Bn	DMU_GP_HOST_PFPROCONPWOP2Bn, OTP set 4	$0D150_H$+n*4	8893

(table continues...)

Table 2538 (continued) Register overview - UCB0_26 (ascending offset address)

Short name	Long name	Offset address	See
UCB0_26_DMU_GP_HOST_PFPROCONPWOP3An	DMU_GP_HOST_PFPROCONPWOP3An, OTP set 4	$0D160_H$+n*4	8893
UCB0_26_DMU_GP_HOST_PFPROCONPWOP3Bn	DMU_GP_HOST_PFPROCONPWOP3Bn, OTP set 4	$0D170_H$+n*4	8894
UCB0_26_DMU_GP_HOST_PFPROCONPWOP4An	DMU_GP_HOST_PFPROCONPWOP4An, OTP set 4	$0D180_H$+n*4	8894
UCB0_26_DMU_GP_HOST_PFPROCONPWOP4Bn	DMU_GP_HOST_PFPROCONPWOP4Bn, OTP set 4	$0D190_H$+n*4	8894
UCB0_26_DMU_GP_HOST_PFPROCONPWOP5An	DMU_GP_HOST_PFPROCONPWOP5An, OTP set 4	$0D1A0_H$+n*4	8895
UCB0_26_DMU_GP_HOST_PFPROCONPWOP5Bn	DMU_GP_HOST_PFPROCONPWOP5Bn, OTP set 4	$0D1B0_H$+n*4	8895
UCB0_26_DMU_GP_HOST_PFPROCONPWP0An	DMU_GP_HOST_PFPROCONPWP0An, OTP set 5	$0D200_H$+n*4	8896
UCB0_26_DMU_GP_HOST_PFPROCONPWP0Bn	DMU_GP_HOST_PFPROCONPWP0Bn, OTP set 5	$0D210_H$+n*4	8896
UCB0_26_DMU_GP_HOST_PFPROCONPWP1An	DMU_GP_HOST_PFPROCONPWP1An, OTP set 5	$0D220_H$+n*4	8897
UCB0_26_DMU_GP_HOST_PFPROCONPWP1Bn	DMU_GP_HOST_PFPROCONPWP1Bn, OTP set 5	$0D230_H$+n*4	8897
UCB0_26_DMU_GP_HOST_PFPROCONPWP2An	DMU_GP_HOST_PFPROCONPWP2An, OTP set 5	$0D240_H$+n*4	8897
UCB0_26_DMU_GP_HOST_PFPROCONPWP2Bn	DMU_GP_HOST_PFPROCONPWP2Bn, OTP set 5	$0D250_H$+n*4	8898
UCB0_26_DMU_GP_HOST_PFPROCONPWP3An	DMU_GP_HOST_PFPROCONPWP3An, OTP set 5	$0D260_H$+n*4	8898
UCB0_26_DMU_GP_HOST_PFPROCONPWP3Bn	DMU_GP_HOST_PFPROCONPWP3Bn, OTP set 5	$0D270_H$+n*4	8899
UCB0_26_DMU_GP_HOST_PFPROCONPWP4An	DMU_GP_HOST_PFPROCONPWP4An, OTP set 5	$0D280_H$+n*4	8899

(table continues...)

Table 2538 (continued) Register overview - UCB0_26 (ascending offset address)

Short name	Long name	Offset address	See
UCB0_26_DMU_GP_HOST_PFPROCONPWP4Bn	DMU_GP_HOST_PFPROCONPWP4Bn, OTP set 5	$0D290_H$+n*4	8900
UCB0_26_DMU_GP_HOST_PFPROCONPWP5An	DMU_GP_HOST_PFPROCONPWP5An, OTP set 5	$0D2A0_H$+n*4	8900
UCB0_26_DMU_GP_HOST_PFPROCONPWP5Bn	DMU_GP_HOST_PFPROCONPWP5Bn, OTP set 5	$0D2B0_H$+n*4	8900
UCB0_26_DMU_GP_HOST_PFPROCONPWOP0An	DMU_GP_HOST_PFPROCONPWOP0An, OTP set 5	$0D300_H$+n*4	8901
UCB0_26_DMU_GP_HOST_PFPROCONPWOP0Bn	DMU_GP_HOST_PFPROCONPWOP0Bn, OTP set 5	$0D310_H$+n*4	8901
UCB0_26_DMU_GP_HOST_PFPROCONPWOP1An	DMU_GP_HOST_PFPROCONPWOP1An, OTP set 5	$0D320_H$+n*4	8902
UCB0_26_DMU_GP_HOST_PFPROCONPWOP1Bn	DMU_GP_HOST_PFPROCONPWOP1Bn, OTP set 5	$0D330_H$+n*4	8902
UCB0_26_DMU_GP_HOST_PFPROCONPWOP2An	DMU_GP_HOST_PFPROCONPWOP2An, OTP set 5	$0D340_H$+n*4	8903
UCB0_26_DMU_GP_HOST_PFPROCONPWOP2Bn	DMU_GP_HOST_PFPROCONPWOP2Bn, OTP set 5	$0D350_H$+n*4	8903
UCB0_26_DMU_GP_HOST_PFPROCONPWOP3An	DMU_GP_HOST_PFPROCONPWOP3An, OTP set 5	$0D360_H$+n*4	8903
UCB0_26_DMU_GP_HOST_PFPROCONPWOP3Bn	DMU_GP_HOST_PFPROCONPWOP3Bn, OTP set 5	$0D370_H$+n*4	8904
UCB0_26_DMU_GP_HOST_PFPROCONPWOP4An	DMU_GP_HOST_PFPROCONPWOP4An, OTP set 5	$0D380_H$+n*4	8904
UCB0_26_DMU_GP_HOST_PFPROCONPWOP4Bn	DMU_GP_HOST_PFPROCONPWOP4Bn, OTP set 5	$0D390_H$+n*4	8905
UCB0_26_DMU_GP_HOST_PFPROCONPWOP5An	DMU_GP_HOST_PFPROCONPWOP5An, OTP set 5	$0D3A0_H$+n*4	8905
UCB0_26_DMU_GP_HOST_PFPROCONPWOP5Bn	DMU_GP_HOST_PFPROCONPWOP5Bn, OTP set 5	$0D3B0_H$+n*4	8906

(table continues...)

Table 2538 (continued) Register overview - UCB0_26 (ascending offset address)

Short name	Long name	Offset address	See
UCB0_26_DMU_GP_HOST_PFPROCONPWP0An	DMU_GP_HOST_PFPROCONPWP0An, OTP set 6	$0D400_H$+n*4	8906
UCB0_26_DMU_GP_HOST_PFPROCONPWP0Bn	DMU_GP_HOST_PFPROCONPWP0Bn, OTP set 6	$0D410_H$+n*4	8906
UCB0_26_DMU_GP_HOST_PFPROCONPWP1An	DMU_GP_HOST_PFPROCONPWP1An, OTP set 6	$0D420_H$+n*4	8907
UCB0_26_DMU_GP_HOST_PFPROCONPWP1Bn	DMU_GP_HOST_PFPROCONPWP1Bn, OTP set 6	$0D430_H$+n*4	8907
UCB0_26_DMU_GP_HOST_PFPROCONPWP2An	DMU_GP_HOST_PFPROCONPWP2An, OTP set 6	$0D440_H$+n*4	8908
UCB0_26_DMU_GP_HOST_PFPROCONPWP2Bn	DMU_GP_HOST_PFPROCONPWP2Bn, OTP set 6	$0D450_H$+n*4	8908
UCB0_26_DMU_GP_HOST_PFPROCONPWP3An	DMU_GP_HOST_PFPROCONPWP3An, OTP set 6	$0D460_H$+n*4	8909
UCB0_26_DMU_GP_HOST_PFPROCONPWP3Bn	DMU_GP_HOST_PFPROCONPWP3Bn, OTP set 6	$0D470_H$+n*4	8909
UCB0_26_DMU_GP_HOST_PFPROCONPWP4An	DMU_GP_HOST_PFPROCONPWP4An, OTP set 6	$0D480_H$+n*4	8909
UCB0_26_DMU_GP_HOST_PFPROCONPWP4Bn	DMU_GP_HOST_PFPROCONPWP4Bn, OTP set 6	$0D490_H$+n*4	8910
UCB0_26_DMU_GP_HOST_PFPROCONPWP5An	DMU_GP_HOST_PFPROCONPWP5An, OTP set 6	$0D4A0_H$+n*4	8910
UCB0_26_DMU_GP_HOST_PFPROCONPWP5Bn	DMU_GP_HOST_PFPROCONPWP5Bn, OTP set 6	$0D4B0_H$+n*4	
UCB0_26_DMU_GP_HOST_PFPROCONPWOP0An	DMU_GP_HOST_PFPROCONPWOP0An, OTP set 6	$0D500_H$+n*4	8911
UCB0_26_DMU_GP_HOST_PFPROCONPWOP0Bn	DMU_GP_HOST_PFPROCONPWOP0Bn, OTP set 6	$0D510_H$+n*4	8911
UCB0_26_DMU_GP_HOST_PFPROCONPWOP1An	DMU_GP_HOST_PFPROCONPWOP1An, OTP set 6	$0D520_H$+n*4	8912

(table continues...)

Table 2538 (continued) Register overview - UCB0_26 (ascending offset address)

Short name	Long name	Offset address	See
UCB0_26_DMU_GP_HOST_PFPROCONPWOP1Bn	DMU_GP_HOST_PFPROCONPWOP1Bn, OTP set 6	$0D530_H$+n*4	8912
UCB0_26_DMU_GP_HOST_PFPROCONPWOP2An	DMU_GP_HOST_PFPROCONPWOP2An, OTP set 6	$0D540_H$+n*4	8912
UCB0_26_DMU_GP_HOST_PFPROCONPWOP2Bn	DMU_GP_HOST_PFPROCONPWOP2Bn, OTP set 6	$0D550_H$+n*4	8913
UCB0_26_DMU_GP_HOST_PFPROCONPWOP3An	DMU_GP_HOST_PFPROCONPWOP3An, OTP set 6	$0D560_H$+n*4	8913
UCB0_26_DMU_GP_HOST_PFPROCONPWOP3Bn	DMU_GP_HOST_PFPROCONPWOP3Bn, OTP set 6	$0D570_H$+n*4	8914
UCB0_26_DMU_GP_HOST_PFPROCONPWOP4An	DMU_GP_HOST_PFPROCONPWOP4An, OTP set 6	$0D580_H$+n*4	8914
UCB0_26_DMU_GP_HOST_PFPROCONPWOP4Bn	DMU_GP_HOST_PFPROCONPWOP4Bn, OTP set 6	$0D590_H$+n*4	
UCB0_26_DMU_GP_HOST_PFPROCONPWOP5An	DMU_GP_HOST_PFPROCONPWOP5An, OTP set 6	$0D5A0_H$+n*4	8915
UCB0_26_DMU_GP_HOST_PFPROCONPWOP5Bn	DMU_GP_HOST_PFPROCONPWOP5Bn, OTP set 6	$0D5B0_H$+n*4	8915
UCB0_26_CONFIRMATION_Set_4	Confirmation CODE ORIGINAL Set_4, 32-bit	$0D7D0_H$	8915
UCB0_26_CONFIRMATION_Set_5	Confirmation CODE ORIGINAL Set_5, 32-bit	$0D7E0_H$	8916
UCB0_26_CONFIRMATION_Set_6	Confirmation CODE ORIGINAL Set_6, 32-bit	$0D7F0_H$	8916

48.4.3.13.1 DMU_GP_HOST_PFPROCONPWP0An, OTP set 4

UCB0_26_DMU_GP_HOST_PFPROCONPWP0An (n=0-3) Offset address: $0D000_H$+n*4

DMU_GP_HOST_PFPROCONPWP0An, OTP set 4 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONPWP0An															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONPWP0An															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWP0An	31:0	r	**DMU_GP_HOST_PFPROCONPWP0An register value**

48.4.3.13.2 DMU_GP_HOST_PFPROCONPWP0Bn, OTP set 4

UCB0_26_DMU_GP_HOST_PFPROCONPWP0Bn (n=0-3) Offset address: $0D010_H$+n*4

DMU_GP_HOST_PFPROCONPWP0Bn, OTP set 4 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONPWP0Bn															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONPWP0Bn															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWP0Bn	31:0	r	**DMU_GP_HOST_PFPROCONPWP0Bn register value**

48.4.3.13.3 DMU_GP_HOST_PFPROCONPWP1An, OTP set 4

UCB0_26_DMU_GP_HOST_PFPROCONPWP1An (n=0-3) Offset address: $0D020_H$+n*4

DMU_GP_HOST_PFPROCONPWP1An, OTP set 4 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONPWP1An															
r															

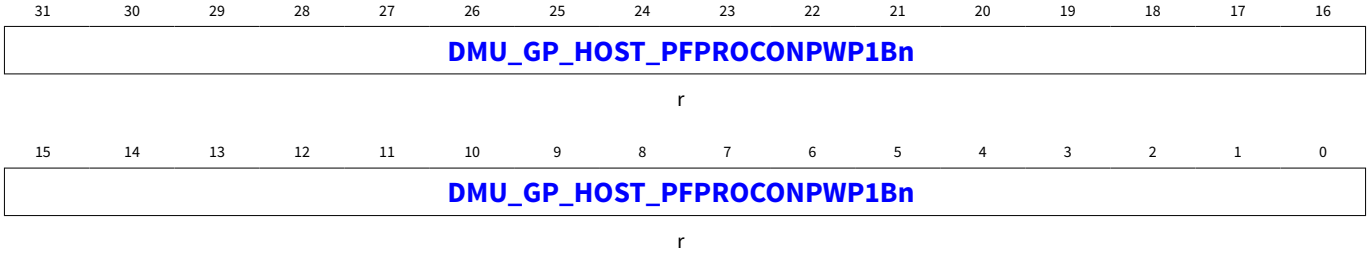
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONPWP1An															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWP1An	31:0	r	**DMU_GP_HOST_PFPROCONPWP1An register value**

48.4.3.13.4 DMU_GP_HOST_PFPROCONPWP1Bn, OTP set 4

UCB0_26_DMU_GP_HOST_PFPROCONPWP1Bn (n=0-3) Offset address: $0D030_H+n*4$

DMU_GP_HOST_PFPROCONPWP1Bn, OTP set 4 value: $0000\ 0000_H$

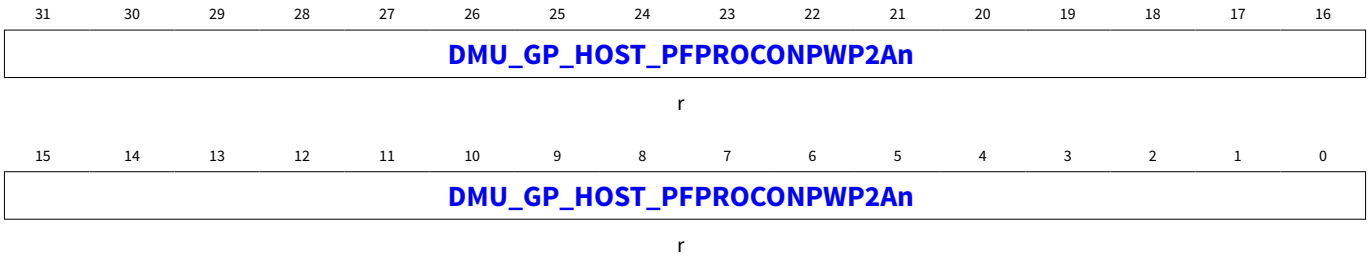


Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWP1Bn	31:0	r	**DMU_GP_HOST_PFPROCONPWP1Bn register value**

48.4.3.13.5 DMU_GP_HOST_PFPROCONPWP2An, OTP set 4

UCB0_26_DMU_GP_HOST_PFPROCONPWP2An (n=0-3) Offset address: $0D040_H+n*4$

DMU_GP_HOST_PFPROCONPWP2An, OTP set 4 value: $0000\ 0000_H$



Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWP2An	31:0	r	**DMU_GP_HOST_PFPROCONPWP2An register value**

48.4.3.13.6 DMU_GP_HOST_PFPROCONPWP2Bn, OTP set 4

UCB0_26_DMU_GP_HOST_PFPROCONPWP2Bn (n=0-3) Offset address: $0D050_H$+n*4

DMU_GP_HOST_PFPROCONPWP2Bn, OTP set 4 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONPWP2Bn															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONPWP2Bn															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWP2Bn	31:0	r	**DMU_GP_HOST_PFPROCONPWP2Bn register value**

48.4.3.13.7 DMU_GP_HOST_PFPROCONPWP3An, OTP set 4

UCB0_26_DMU_GP_HOST_PFPROCONPWP3An (n=0-3) Offset address: $0D060_H$+n*4

DMU_GP_HOST_PFPROCONPWP3An, OTP set 4 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONPWP3An															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONPWP3An															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWP3An	31:0	r	**DMU_GP_HOST_PFPROCONPWP3An register value**

48.4.3.13.8 DMU_GP_HOST_PFPROCONPWP3Bn, OTP set 4

UCB0_26_DMU_GP_HOST_PFPROCONPWP3Bn (n=0-3) Offset address: $0D070_H$+n*4

DMU_GP_HOST_PFPROCONPWP3Bn, OTP set 4 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONPWP3Bn															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONPWP3Bn															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWP3Bn	31:0	r	**DMU_GP_HOST_PFPROCONPWP3Bn register value**

48.4.3.13.9 DMU_GP_HOST_PFPROCONPWP4An, OTP set 4

UCB0_26_DMU_GP_HOST_PFPROCONPWP4An (n=0-3) Offset address: $0D080_H$+n*4

DMU_GP_HOST_PFPROCONPWP4An, OTP set 4 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONPWP4An															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONPWP4An															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWP4An	31:0	r	**DMU_GP_HOST_PFPROCONPWP4An register value**

48.4.3.13.10 DMU_GP_HOST_PFPROCONPWP4Bn, OTP set 4

UCB0_26_DMU_GP_HOST_PFPROCONPWP4Bn (n=0-3) Offset address: $0D090_H$+n*4

DMU_GP_HOST_PFPROCONPWP4Bn, OTP set 4 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONPWP4Bn															
r															

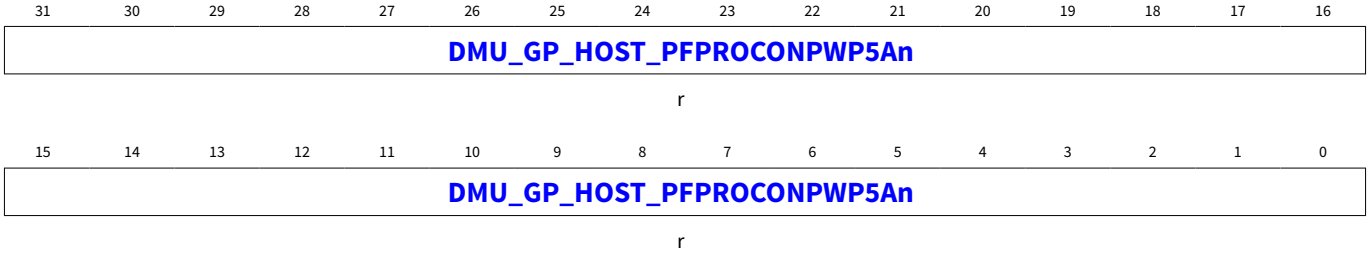
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONPWP4Bn															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWP4Bn	31:0	r	**DMU_GP_HOST_PFPROCONPWP4Bn register value**

48.4.3.13.11 DMU_GP_HOST_PFPROCONPWP5An, OTP set 4

UCB0_26_DMU_GP_HOST_PFPROCONPWP5An (n=0-3) Offset address: $0D0A0_H+n*4$

DMU_GP_HOST_PFPROCONPWP5An, OTP set 4 value: $0000\ 0000_H$

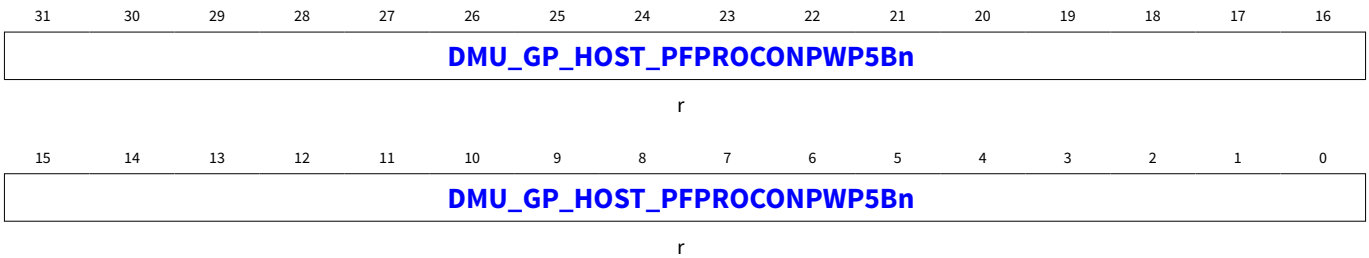


Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWP5An	31:0	r	**DMU_GP_HOST_PFPROCONPWP5An register value**

48.4.3.13.12 DMU_GP_HOST_PFPROCONPWP5Bn, OTP set 4

UCB0_26_DMU_GP_HOST_PFPROCONPWP5Bn (n=0-3) Offset address: $0D0B0_H+n*4$

DMU_GP_HOST_PFPROCONPWP5Bn, OTP set 4 value: $0000\ 0000_H$



Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWP5Bn	31:0	r	**DMU_GP_HOST_PFPROCONPWP5Bn register value**

48.4.3.13.13 DMU_GP_HOST_PFPROCONPWOP0An, OTP set 4

UCB0_26_DMU_GP_HOST_PFPROCONPWOP0An (n=0-3) Offset address: $0D100_H+n*4$

DMU_GP_HOST_PFPROCONPWOP0An, OTP set 4 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONPWOP0An															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONPWOP0An															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWOP0An	31:0	r	**DMU_GP_HOST_PFPROCONPWOP0An register value**

48.4.3.13.14 DMU_GP_HOST_PFPROCONPWOP0Bn, OTP set 4

UCB0_26_DMU_GP_HOST_PFPROCONPWOP0Bn (n=0-3) Offset address: $0D110_H+n*4$

DMU_GP_HOST_PFPROCONPWOP0Bn, OTP set 4 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONPWOP0Bn															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONPWOP0Bn															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWOP0Bn	31:0	r	**DMU_GP_HOST_PFPROCONPWOP0Bn register value**

48.4.3.13.15 DMU_GP_HOST_PFPROCONPWOP1An, OTP set 4

UCB0_26_DMU_GP_HOST_PFPROCONPWOP1An (n=0-3) Offset address: $0D120_H+n*4$

DMU_GP_HOST_PFPROCONPWOP1An, OTP set 4 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONPWOP1An															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONPWOP1An															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWOP1An	31:0	r	**DMU_GP_HOST_PFPROCONPWOP1An register value**

48.4.3.13.16 DMU_GP_HOST_PFPROCONPWOP1Bn, OTP set 4

UCB0_26_DMU_GP_HOST_PFPROCONPWOP1Bn (n=0-3) Offset address: $0D130_H+n*4$

DMU_GP_HOST_PFPROCONPWOP1Bn, OTP set 4 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONPWOP1Bn															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONPWOP1Bn															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWOP1Bn	31:0	r	**DMU_GP_HOST_PFPROCONPWOP1Bn register value**

48.4.3.13.17 DMU_GP_HOST_PFPROCONPWOP2An, OTP set 4

UCB0_26_DMU_GP_HOST_PFPROCONPWOP2An (n=0-3) Offset address: $0D140_H+n*4$

DMU_GP_HOST_PFPROCONPWOP2An, OTP set 4 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONPWOP2An															
r															

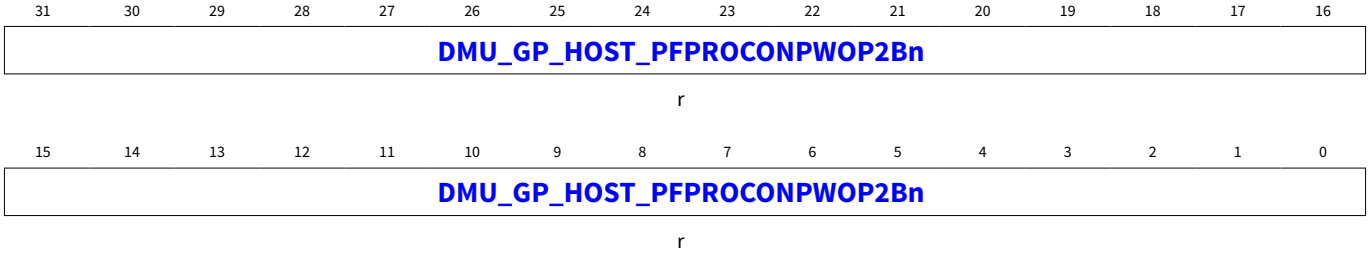
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONPWOP2An															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWOP2An	31:0	r	**DMU_GP_HOST_PFPROCONPWOP2An register value**

48.4.3.13.18 DMU_GP_HOST_PFPROCONPWOP2Bn, OTP set 4

UCB0_26_DMU_GP_HOST_PFPROCONPWOP2Bn (n=0-3) Offset address: $0D150_H+n*4$

DMU_GP_HOST_PFPROCONPWOP2Bn, OTP set 4 value: $0000\ 0000_H$

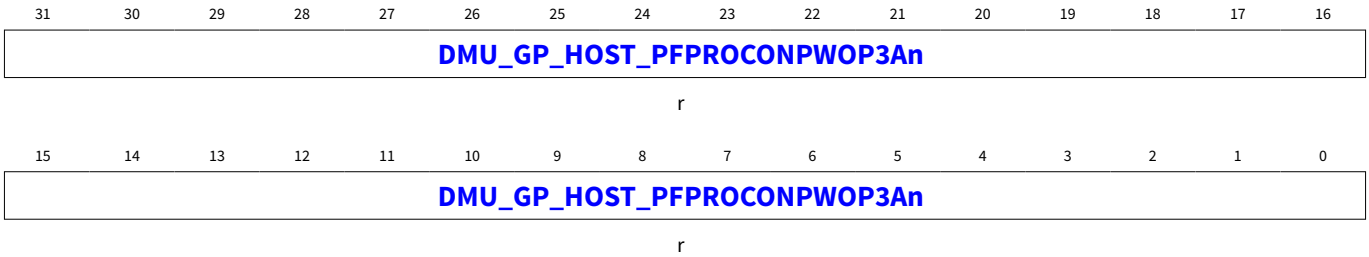


Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWOP2Bn	31:0	r	**DMU_GP_HOST_PFPROCONPWOP2Bn register value**

48.4.3.13.19 DMU_GP_HOST_PFPROCONPWOP3An, OTP set 4

UCB0_26_DMU_GP_HOST_PFPROCONPWOP3An (n=0-3) Offset address: $0D160_H+n*4$

DMU_GP_HOST_PFPROCONPWOP3An, OTP set 4 value: $0000\ 0000_H$



Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWOP3An	31:0	r	**DMU_GP_HOST_PFPROCONPWOP4An register value**

48.4.3.13.20 DMU_GP_HOST_PFPROCONPWOP3Bn, OTP set 4

UCB0_26_DMU_GP_HOST_PFPROCONPWOP3Bn (n=0-3) Offset address: $0D170_H$+n*4

DMU_GP_HOST_PFPROCONPWOP3Bn, OTP set 4 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONPWOP3An															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONPWOP3An															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWOP3An	31:0	r	**DMU_GP_HOST_PFPROCONPWOP3An register value**

48.4.3.13.21 DMU_GP_HOST_PFPROCONPWOP4An, OTP set 4

UCB0_26_DMU_GP_HOST_PFPROCONPWOP4An (n=0-3) Offset address: $0D180_H$+n*4

DMU_GP_HOST_PFPROCONPWOP4An, OTP set 4 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONPWOP4An															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONPWOP4An															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWOP4An	31:0	r	**DMU_GP_HOST_PFPROCONPWOP4An register value**

48.4.3.13.22 DMU_GP_HOST_PFPROCONPWOP4Bn, OTP set 4

UCB0_26_DMU_GP_HOST_PFPROCONPWOP4Bn (n=0-3) Offset address: $0D190_H$+n*4

DMU_GP_HOST_PFPROCONPWOP4Bn, OTP set 4 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONPWOP4Bn															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONPWOP4Bn															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWOP4Bn	31:0	r	**DMU_GP_HOST_PFPROCONPWOP4Bn register value**

48.4.3.13.23 DMU_GP_HOST_PFPROCONPWOP5An, OTP set 4

UCB0_26_DMU_GP_HOST_PFPROCONPWOP5An (n=0-3) Offset address: $0D1A0_H+n*4$

DMU_GP_HOST_PFPROCONPWOP5An, OTP set 4 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONPWOP5An															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONPWOP5An															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWOP5An	31:0	r	**DMU_GP_HOST_PFPROCONPWOP5An register value**

48.4.3.13.24 DMU_GP_HOST_PFPROCONPWOP5Bn, OTP set 4

UCB0_26_DMU_GP_HOST_PFPROCONPWOP5Bn (n=0-3) Offset address: $0D1B0_H+n*4$

DMU_GP_HOST_PFPROCONPWOP5Bn, OTP set 4 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONPWOP5Bn															
r															

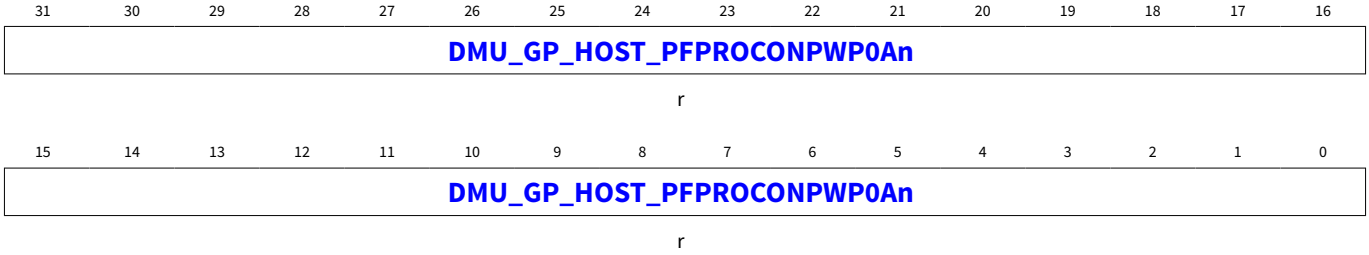
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONPWOP5Bn															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWOP5Bn	31:0	r	**DMU_GP_HOST_PFPROCONPWOP5Bn register value**

48.4.3.13.25 DMU_GP_HOST_PFPROCONPWP0An, OTP set 5

UCB0_26_DMU_GP_HOST_PFPROCONPWP0An (n=0-3) Offset address: $0D200_H+n*4$

DMU_GP_HOST_PFPROCONPWP0An, OTP set 5 value: $0000\ 0000_H$

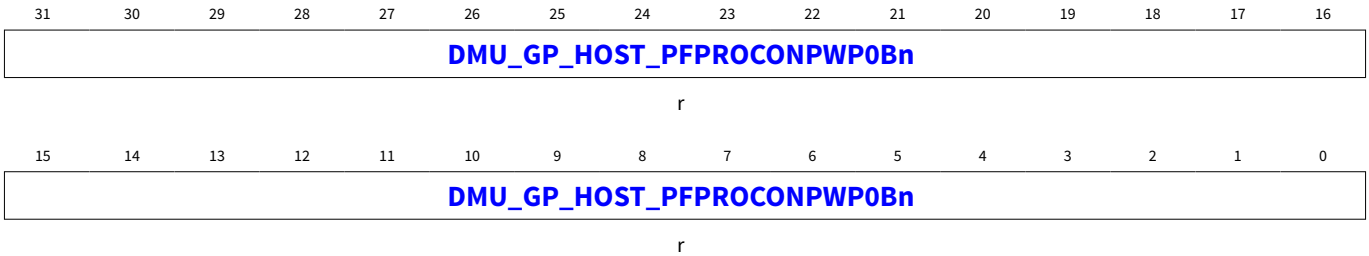


Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWP0An	31:0	r	**DMU_GP_HOST_PFPROCONPWP0An register value**

48.4.3.13.26 DMU_GP_HOST_PFPROCONPWP0Bn, OTP set 5

UCB0_26_DMU_GP_HOST_PFPROCONPWP0Bn (n=0-3) Offset address: $0D210_H+n*4$

DMU_GP_HOST_PFPROCONPWP0Bn, OTP set 5 value: $0000\ 0000_H$



Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWP0Bn	31:0	r	**DMU_GP_HOST_PFPROCONPWP0Bn register value**

48.4.3.13.27 DMU_GP_HOST_PFPROCONPWP1An, OTP set 5

UCB0_26_DMU_GP_HOST_PFPROCONPWP1An (n=0-3) Offset address: $0D220_H$+n*4

DMU_GP_HOST_PFPROCONPWP1An, OTP set 5 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONPWP1An															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONPWP1An															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWP1An	31:0	r	**DMU_GP_HOST_PFPROCONPWP1An register value**

48.4.3.13.28 DMU_GP_HOST_PFPROCONPWP1Bn, OTP set 5

UCB0_26_DMU_GP_HOST_PFPROCONPWP1Bn (n=0-3) Offset address: $0D230_H$+n*4

DMU_GP_HOST_PFPROCONPWP1Bn, OTP set 5 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONPWP1Bn															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONPWP1Bn															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWP1Bn	31:0	r	**DMU_GP_HOST_PFPROCONPWP1Bn register value**

48.4.3.13.29 DMU_GP_HOST_PFPROCONPWP2An, OTP set 5

UCB0_26_DMU_GP_HOST_PFPROCONPWP2An (n=0-3) Offset address: $0D240_H$+n*4

DMU_GP_HOST_PFPROCONPWP2An, OTP set 5 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONPWP2An															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONPWP2An															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWP2An	31:0	r	**DMU_GP_HOST_PFPROCONPWP2An register value**

48.4.3.13.30 DMU_GP_HOST_PFPROCONPWP2Bn, OTP set 5

UCB0_26_DMU_GP_HOST_PFPROCONPWP2Bn (n=0-3) Offset address: $0D250_H$+n*4

DMU_GP_HOST_PFPROCONPWP2Bn, OTP set 5 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONPWP2Bn															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONPWP2Bn															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWP2Bn	31:0	r	**DMU_GP_HOST_PFPROCONPWP2Bn register value**

48.4.3.13.31 DMU_GP_HOST_PFPROCONPWP3An, OTP set 5

UCB0_26_DMU_GP_HOST_PFPROCONPWP3An (n=0-3) Offset address: $0D260_H$+n*4

DMU_GP_HOST_PFPROCONPWP3An, OTP set 5 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONPWP3An															
r															

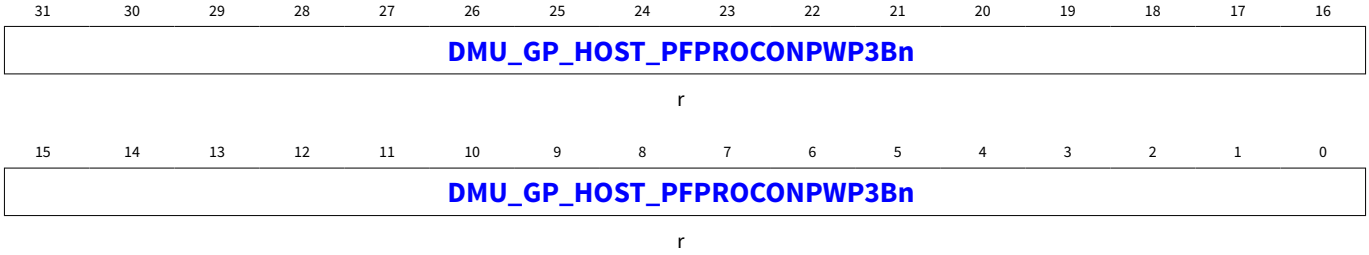
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONPWP3An															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWP3An	31:0	r	**DMU_GP_HOST_PFPROCONPWP3An register value**

48.4.3.13.32 DMU_GP_HOST_PFPROCONPWP3Bn, OTP set 5

UCB0_26_DMU_GP_HOST_PFPROCONPWP3Bn (n=0-3) Offset address: $0D270_H+n*4$

DMU_GP_HOST_PFPROCONPWP3Bn, OTP set 5 value: $0000\ 0000_H$

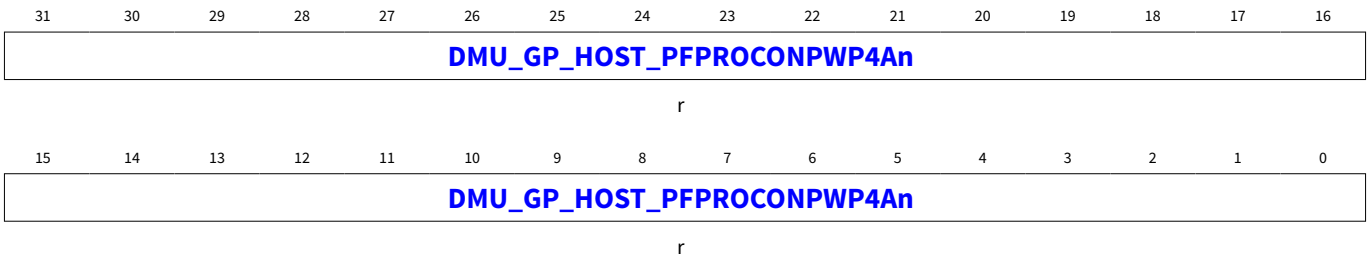


Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWP3Bn	31:0	r	**DMU_GP_HOST_PFPROCONPWP3Bn register value**

48.4.3.13.33 DMU_GP_HOST_PFPROCONPWP4An, OTP set 5

UCB0_26_DMU_GP_HOST_PFPROCONPWP4An (n=0-3) Offset address: $0D280_H+n*4$

DMU_GP_HOST_PFPROCONPWP4An, OTP set 5 value: $0000\ 0000_H$



Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWP4An	31:0	r	**DMU_GP_HOST_PFPROCONPWP4An register value**

48.4.3.13.34 DMU_GP_HOST_PFPROCONPWP4Bn, OTP set 5

UCB0_26_DMU_GP_HOST_PFPROCONPWP4Bn (n=0-3) Offset address: $0D290_H$+n*4

DMU_GP_HOST_PFPROCONPWP4Bn, OTP set 5 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONPWP4Bn															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONPWP4Bn															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWP4Bn	31:0	r	**DMU_GP_HOST_PFPROCONPWP4Bn register value**

48.4.3.13.35 DMU_GP_HOST_PFPROCONPWP5An, OTP set 5

UCB0_26_DMU_GP_HOST_PFPROCONPWP5An (n=0-3) Offset address: $0D2A0_H$+n*4

DMU_GP_HOST_PFPROCONPWP5An, OTP set 5 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONPWP5An															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONPWP5An															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWP5An	31:0	r	**DMU_GP_HOST_PFPROCONPWP5An register value**

48.4.3.13.36 DMU_GP_HOST_PFPROCONPWP5Bn, OTP set 5

UCB0_26_DMU_GP_HOST_PFPROCONPWP5Bn (n=0-3) Offset address: $0D2B0_H$+n*4

DMU_GP_HOST_PFPROCONPWP5Bn, OTP set 5 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONPWP5Bn															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONPWP5Bn															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWP5Bn	31:0	r	**DMU_GP_HOST_PFPROCONPWP5Bn register value**

48.4.3.13.37 DMU_GP_HOST_PFPROCONPWOP0An, OTP set 5

UCB0_26_DMU_GP_HOST_PFPROCONPWOP0An (n=0-3) Offset address: $0D300_H+n*4$

DMU_GP_HOST_PFPROCONPWOP0An, OTP set 5 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONPWOP0An															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONPWOP0An															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWOP0An	31:0	r	**DMU_GP_HOST_PFPROCONPWOP0An register value**

48.4.3.13.38 DMU_GP_HOST_PFPROCONPWOP0Bn, OTP set 5

UCB0_26_DMU_GP_HOST_PFPROCONPWOP0Bn (n=0-3) Offset address: $0D310_H+n*4$

DMU_GP_HOST_PFPROCONPWOP0Bn, OTP set 5 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONPWOP0Bn															
r															

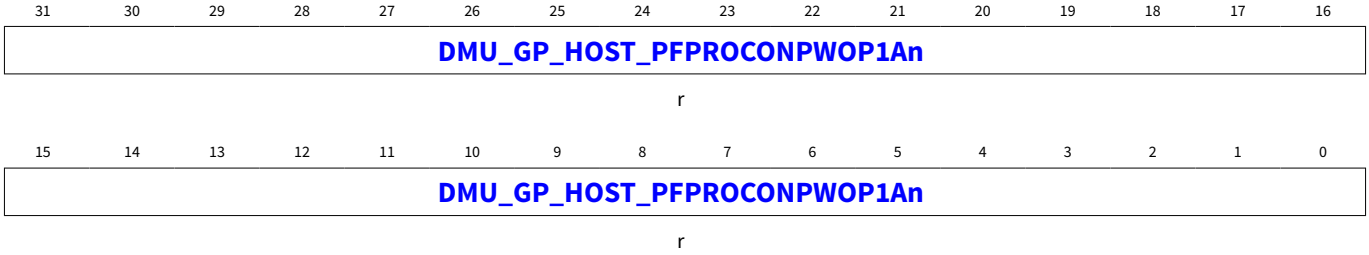
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONPWOP0Bn															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWOP0Bn	31:0	r	**DMU_GP_HOST_PFPROCONPWOP0Bn register value**

48.4.3.13.39 DMU_GP_HOST_PFPROCONPWOP1An, OTP set 5

UCB0_26_DMU_GP_HOST_PFPROCONPWOP1An (n=0-3) Offset address: $0D320_H+n*4$

DMU_GP_HOST_PFPROCONPWOP1An, OTP set 5 value: $0000\ 0000_H$

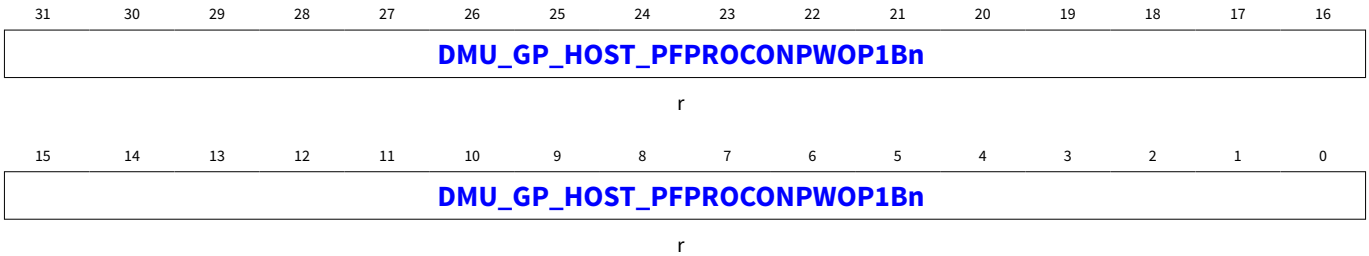


Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWOP1An	31:0	r	**DMU_GP_HOST_PFPROCONPWOP1An register value**

48.4.3.13.40 DMU_GP_HOST_PFPROCONPWOP1Bn, OTP set 5

UCB0_26_DMU_GP_HOST_PFPROCONPWOP1Bn (n=0-3) Offset address: $0D330_H+n*4$

DMU_GP_HOST_PFPROCONPWOP1Bn, OTP set 5 value: $0000\ 0000_H$



Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWOP1Bn	31:0	r	**DMU_GP_HOST_PFPROCONPWOP1Bn register value**

48.4.3.13.41 DMU_GP_HOST_PFPROCONPWOP2An, OTP set 5

UCB0_26_DMU_GP_HOST_PFPROCONPWOP2An (n=0-3) Offset address: $0D340_H$+n*4

DMU_GP_HOST_PFPROCONPWOP2An, OTP set 5 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONPWOP2An															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONPWOP2An															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWOP2An	31:0	r	**DMU_GP_HOST_PFPROCONPWOP2An register value**

48.4.3.13.42 DMU_GP_HOST_PFPROCONPWOP2Bn, OTP set 5

UCB0_26_DMU_GP_HOST_PFPROCONPWOP2Bn (n=0-3) Offset address: $0D350_H$+n*4

DMU_GP_HOST_PFPROCONPWOP2Bn, OTP set 5 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONPWOP2Bn															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONPWOP2Bn															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWOP2Bn	31:0	r	**DMU_GP_HOST_PFPROCONPWOP2Bn register value**

48.4.3.13.43 DMU_GP_HOST_PFPROCONPWOP3An, OTP set 5

UCB0_26_DMU_GP_HOST_PFPROCONPWOP3An (n=0-3) Offset address: $0D360_H$+n*4

DMU_GP_HOST_PFPROCONPWOP3An, OTP set 5 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONPWOP3An															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONPWOP3An															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWOP3An	31:0	r	**DMU_GP_HOST_PFPROCONPWOP4An register value**

48.4.3.13.44 DMU_GP_HOST_PFPROCONPWOP3Bn, OTP set 5

UCB0_26_DMU_GP_HOST_PFPROCONPWOP3Bn (n=0-3) Offset address: $0D370_H+n*4$

DMU_GP_HOST_PFPROCONPWOP3Bn, OTP set 5 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONPWOP3Bn															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONPWOP3Bn															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWOP3Bn	31:0	r	**DMU_GP_HOST_PFPROCONPWOP3Bn register value**

48.4.3.13.45 DMU_GP_HOST_PFPROCONPWOP4An, OTP set 5

UCB0_26_DMU_GP_HOST_PFPROCONPWOP4An (n=0-3) Offset address: $0D380_H+n*4$

DMU_GP_HOST_PFPROCONPWOP4An, OTP set 5 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONPWOP4An															
r															

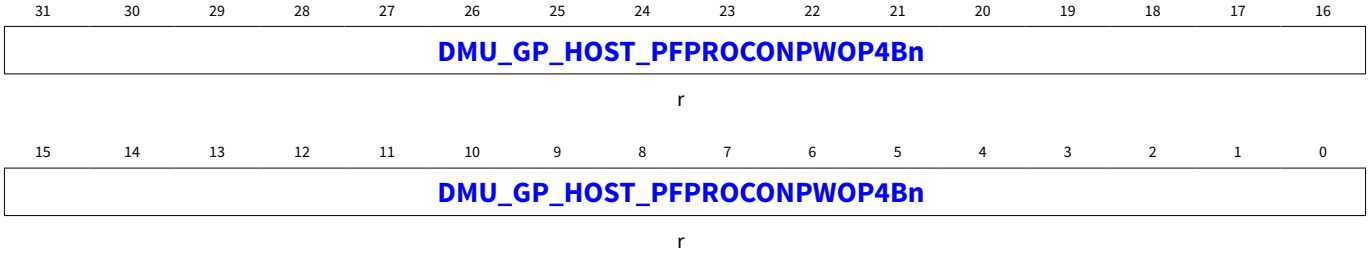
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONPWOP4An															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWOP4An	31:0	r	**DMU_GP_HOST_PFPROCONPWOP4An register value**

48.4.3.13.46 DMU_GP_HOST_PFPROCONPWOP4Bn, OTP set 5

UCB0_26_DMU_GP_HOST_PFPROCONPWOP4Bn (n=0-3) Offset address: $0D390_H+n*4$

DMU_GP_HOST_PFPROCONPWOP4Bn, OTP set 5 value: $0000\ 0000_H$

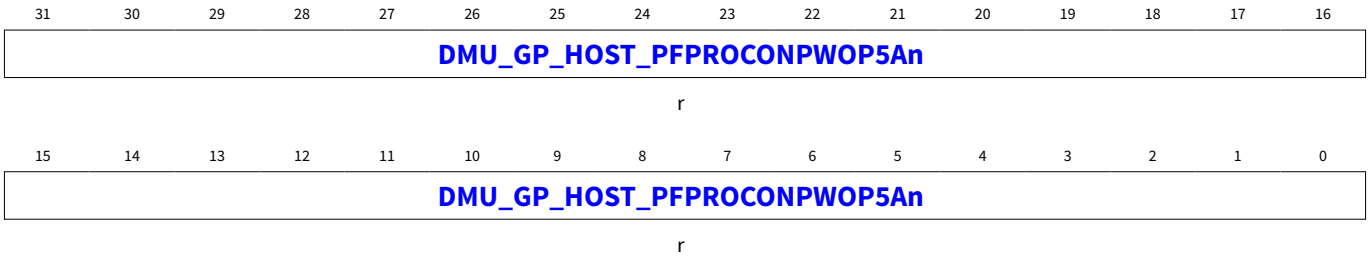


Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWOP4Bn	31:0	r	**DMU_GP_HOST_PFPROCONPWOP4Bn register value**

48.4.3.13.47 DMU_GP_HOST_PFPROCONPWOP5An, OTP set 5

UCB0_26_DMU_GP_HOST_PFPROCONPWOP5An (n=0-3) Offset address: $0D3A0_H+n*4$

DMU_GP_HOST_PFPROCONPWOP5An, OTP set 5 value: $0000\ 0000_H$



Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWOP5An	31:0	r	**DMU_GP_HOST_PFPROCONPWOP5An register value**

48.4.3.13.48 DMU_GP_HOST_PFPROCONPWOP5Bn, OTP set 5

UCB0_26_DMU_GP_HOST_PFPROCONPWOP5Bn (n=0-3) Offset address: $0D3B0_H+n*4$

DMU_GP_HOST_PFPROCONPWOP5Bn, OTP set 5 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONPWOP5Bn															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONPWOP5Bn															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWOP5Bn	31:0	r	**DMU_GP_HOST_PFPROCONPWOP5Bn register value**

48.4.3.13.49 DMU_GP_HOST_PFPROCONPWP0An, OTP set 6

UCB0_26_DMU_GP_HOST_PFPROCONPWP0An (n=0-3) Offset address: $0D400_H+n*4$

DMU_GP_HOST_PFPROCONPWP0An, OTP set 6 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONPWP0An															
r															

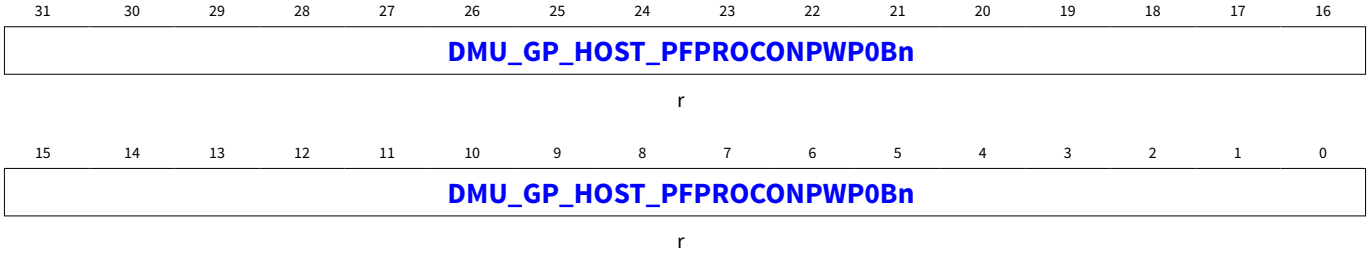
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONPWP0An															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWP0An	31:0	r	**DMU_GP_HOST_PFPROCONPWP0An register value**

48.4.3.13.50 DMU_GP_HOST_PFPROCONPWP0Bn, OTP set 6

UCB0_26_DMU_GP_HOST_PFPROCONPWP0Bn (n=0-3) Offset address: $0D410_H+n*4$

DMU_GP_HOST_PFPROCONPWP0Bn, OTP set 6 value: $0000\ 0000_H$

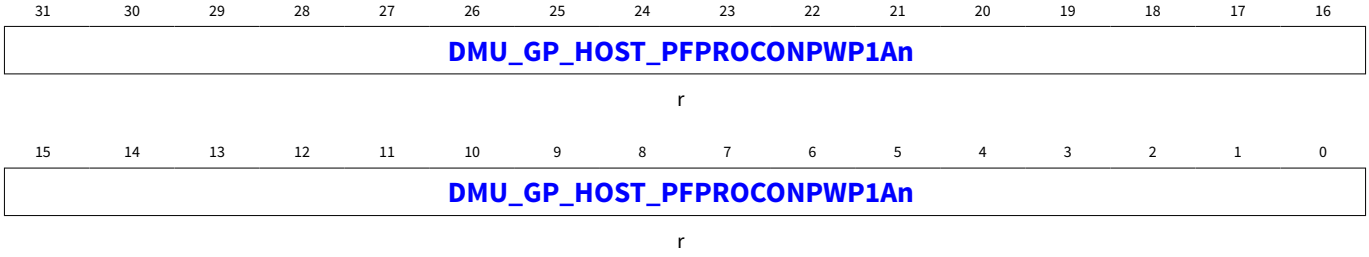


Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWP0Bn	31:0	r	**DMU_GP_HOST_PFPROCONPWP0Bn register value**

48.4.3.13.51 DMU_GP_HOST_PFPROCONPWP1An, OTP set 6

UCB0_26_DMU_GP_HOST_PFPROCONPWP1An (n=0-3) Offset address: $0D420_H+n*4$

DMU_GP_HOST_PFPROCONPWP1An, OTP set 6 value: $0000\ 0000_H$

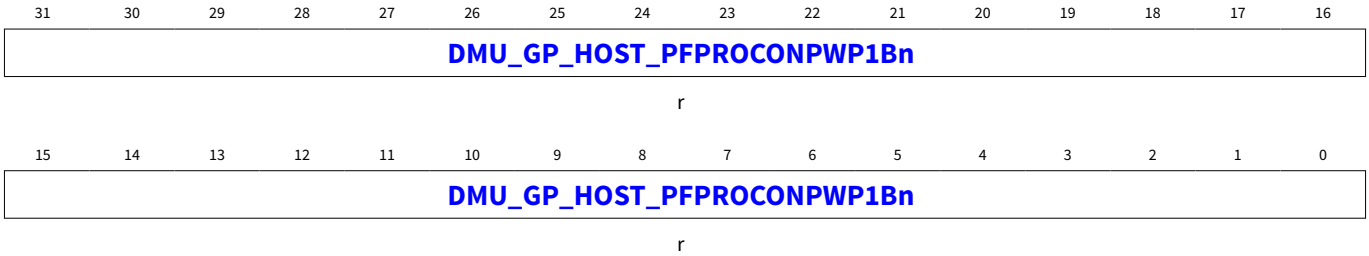


Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWP1An	31:0	r	**DMU_GP_HOST_PFPROCONPWP1An register value**

48.4.3.13.52 DMU_GP_HOST_PFPROCONPWP1Bn, OTP set 6

UCB0_26_DMU_GP_HOST_PFPROCONPWP1Bn (n=0-3) Offset address: $0D430_H+n*4$

DMU_GP_HOST_PFPROCONPWP1Bn, OTP set 6 value: $0000\ 0000_H$



Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWP1Bn	31:0	r	**DMU_GP_HOST_PFPROCONPWP1Bn register value**

48.4.3.13.53 DMU_GP_HOST_PFPROCONPWP2An, OTP set 6

UCB0_26_DMU_GP_HOST_PFPROCONPWP2An (n=0-3) Offset address: $0D440_H+n*4$

DMU_GP_HOST_PFPROCONPWP2An, OTP set 6 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONPWP2An															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONPWP2An															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWP2An	31:0	r	**DMU_GP_HOST_PFPROCONPWP2An register value**

48.4.3.13.54 DMU_GP_HOST_PFPROCONPWP2Bn, OTP set 6

UCB0_26_DMU_GP_HOST_PFPROCONPWP2Bn (n=0-3) Offset address: $0D450_H+n*4$

DMU_GP_HOST_PFPROCONPWP2Bn, OTP set 6 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONPWP2Bn															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONPWP2Bn															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWP2Bn	31:0	r	**DMU_GP_HOST_PFPROCONPWP2Bn register value**

48.4.3.13.55 DMU_GP_HOST_PFPROCONPWP3An, OTP set 6

UCB0_26_DMU_GP_HOST_PFPROCONPWP3An (n=0-3) Offset address: $0D460_H$+n*4

DMU_GP_HOST_PFPROCONPWP3An, OTP set 6 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONPWP3An															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONPWP3An															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWP3An	31:0	r	**DMU_GP_HOST_PFPROCONPWP3An register value**

48.4.3.13.56 DMU_GP_HOST_PFPROCONPWP3Bn, OTP set 6

UCB0_26_DMU_GP_HOST_PFPROCONPWP3Bn (n=0-3) Offset address: $0D470_H$+n*4

DMU_GP_HOST_PFPROCONPWP3Bn, OTP set 6 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONPWP3Bn															
r															

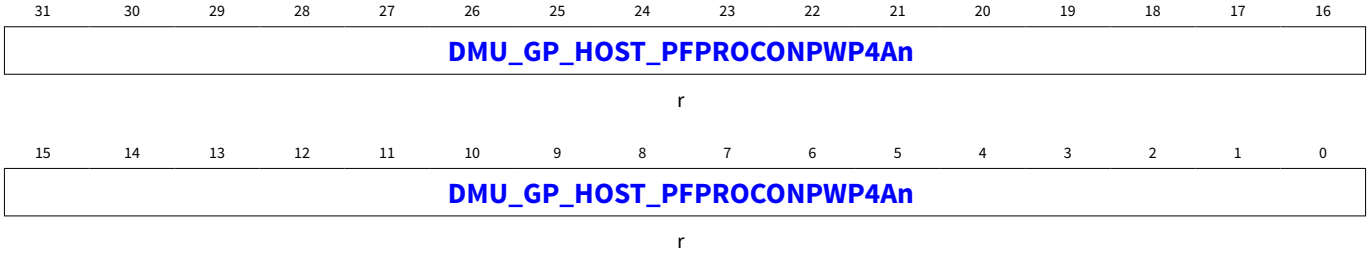
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONPWP3Bn															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWP3Bn	31:0	r	**DMU_GP_HOST_PFPROCONPWP3Bn register value**

48.4.3.13.57 DMU_GP_HOST_PFPROCONPWP4An, OTP set 6

UCB0_26_DMU_GP_HOST_PFPROCONPWP4An (n=0-3) Offset address: $0D480_H$+n*4

DMU_GP_HOST_PFPROCONPWP4An, OTP set 6 value: $0000\ 0000_H$

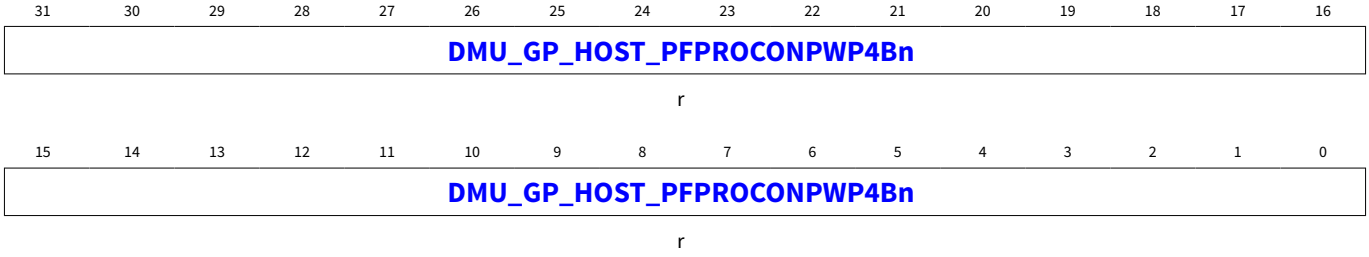


Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWP4An	31:0	r	**DMU_GP_HOST_PFPROCONPWP4An register value**

48.4.3.13.58 DMU_GP_HOST_PFPROCONPWP4Bn, OTP set 6

UCB0_26_DMU_GP_HOST_PFPROCONPWP4Bn (n=0-3) Offset address: $0D490_H$+n*4

DMU_GP_HOST_PFPROCONPWP4Bn, OTP set 6 value: $0000\ 0000_H$

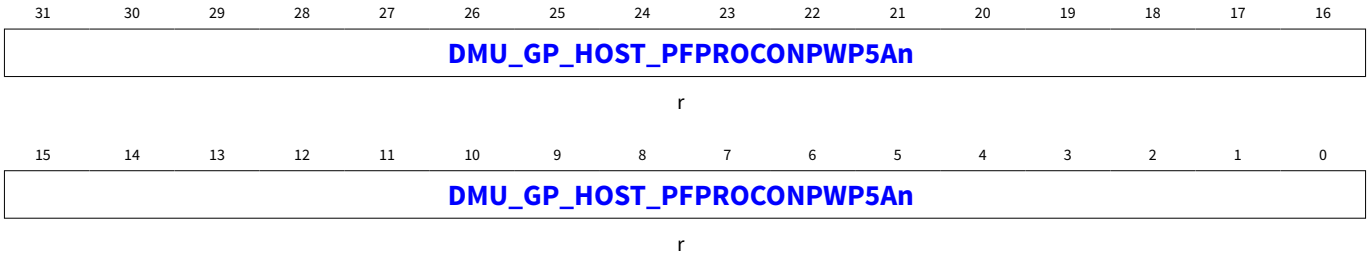


Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWP4Bn	31:0	r	**DMU_GP_HOST_PFPROCONPWP4Bn register value**

48.4.3.13.59 DMU_GP_HOST_PFPROCONPWP5An, OTP set 6

UCB0_26_DMU_GP_HOST_PFPROCONPWP5An (n=0-3) Offset address: $0D4A0_H$+n*4

DMU_GP_HOST_PFPROCONPWP5An, OTP set 6 value: $0000\ 0000_H$

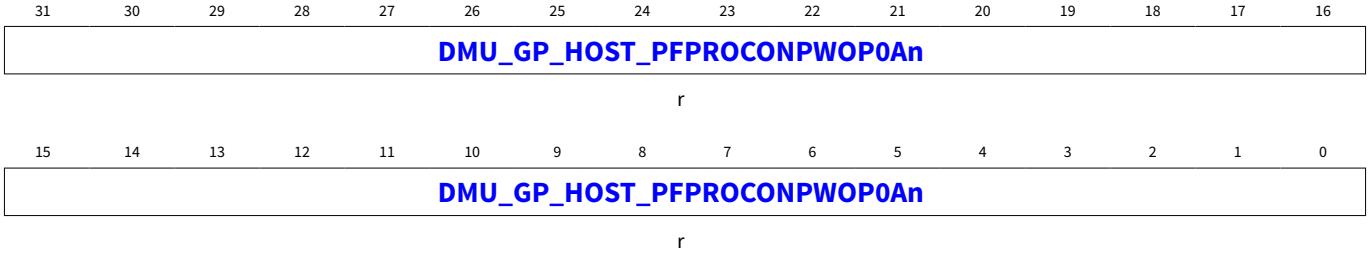


Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWP5An	31:0	r	**DMU_GP_HOST_PFPROCONPWP5An register value**

48.4.3.13.60 DMU_GP_HOST_PFPROCONPWOP0An, OTP set 6

UCB0_26_DMU_GP_HOST_PFPROCONPWOP0An (n=0-3) Offset address: $0D500_H+n*4$

DMU_GP_HOST_PFPROCONPWOP0An, OTP set 6 value: $0000\ 0000_H$

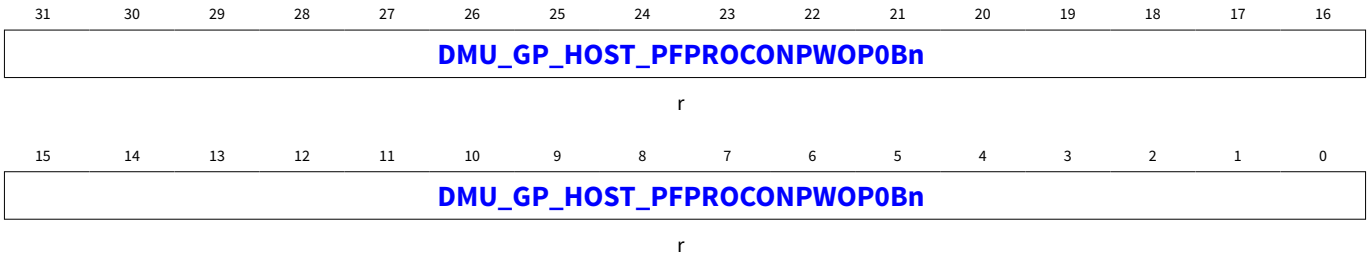


Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWOP0An	31:0	r	**DMU_GP_HOST_PFPROCONPWOP0An register value**

48.4.3.13.61 DMU_GP_HOST_PFPROCONPWOP0Bn, OTP set 6

UCB0_26_DMU_GP_HOST_PFPROCONPWOP0Bn (n=0-3) Offset address: $0D510_H+n*4$

DMU_GP_HOST_PFPROCONPWOP0Bn, OTP set 6 value: $0000\ 0000_H$



Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWOP0Bn	31:0	r	**DMU_GP_HOST_PFPROCONPWOP0Bn register value**

48.4.3.13.62 DMU_GP_HOST_PFPROCONPWOP1An, OTP set 6

UCB0_26_DMU_GP_HOST_PFPROCONPWOP1An (n=0-3) Offset address: $0D520_H$+n*4

DMU_GP_HOST_PFPROCONPWOP1An, OTP set 6 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONPWOP1An															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONPWOP1An															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWOP1An	31:0	r	**DMU_GP_HOST_PFPROCONPWOP1An register value**

48.4.3.13.63 DMU_GP_HOST_PFPROCONPWOP1Bn, OTP set 6

UCB0_26_DMU_GP_HOST_PFPROCONPWOP1Bn (n=0-3) Offset address: $0D530_H$+n*4

DMU_GP_HOST_PFPROCONPWOP1Bn, OTP set 6 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONPWOP1Bn															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONPWOP1Bn															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWOP1Bn	31:0	r	**DMU_GP_HOST_PFPROCONPWOP1Bn register value**

48.4.3.13.64 DMU_GP_HOST_PFPROCONPWOP2An, OTP set 6

UCB0_26_DMU_GP_HOST_PFPROCONPWOP2An (n=0-3) Offset address: $0D540_H$+n*4

DMU_GP_HOST_PFPROCONPWOP2An, OTP set 6 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONPWOP2An															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONPWOP2An															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWOP2An	31:0	r	**DMU_GP_HOST_PFPROCONPWOP2An register value**

48.4.3.13.65 DMU_GP_HOST_PFPROCONPWOP2Bn, OTP set 6

UCB0_26_DMU_GP_HOST_PFPROCONPWOP2Bn (n=0-3) Offset address: $0D550_H+n*4$

DMU_GP_HOST_PFPROCONPWOP2Bn, OTP set 6 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONPWOP2Bn															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONPWOP2Bn															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWOP2Bn	31:0	r	**DMU_GP_HOST_PFPROCONPWOP2Bn register value**

48.4.3.13.66 DMU_GP_HOST_PFPROCONPWOP3An, OTP set 6

UCB0_26_DMU_GP_HOST_PFPROCONPWOP3An (n=0-3) Offset address: $0D560_H+n*4$

DMU_GP_HOST_PFPROCONPWOP3An, OTP set 6 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONPWOP3An															
r															

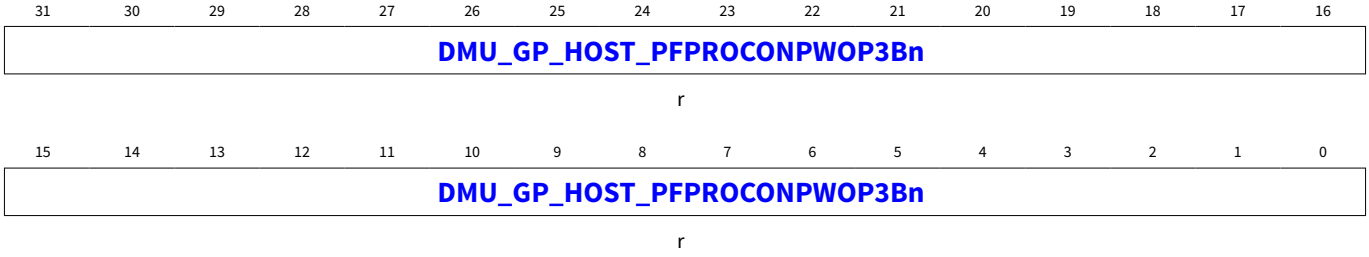
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONPWOP3An															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWOP3An	31:0	r	**DMU_GP_HOST_PFPROCONPWOP4An register value**

48.4.3.13.67 DMU_GP_HOST_PFPROCONPWOP3Bn, OTP set 6

UCB0_26_DMU_GP_HOST_PFPROCONPWOP3Bn (n=0-3) Offset address: $0D570_H+n*4$

DMU_GP_HOST_PFPROCONPWOP3Bn, OTP set 6 value: $0000\ 0000_H$

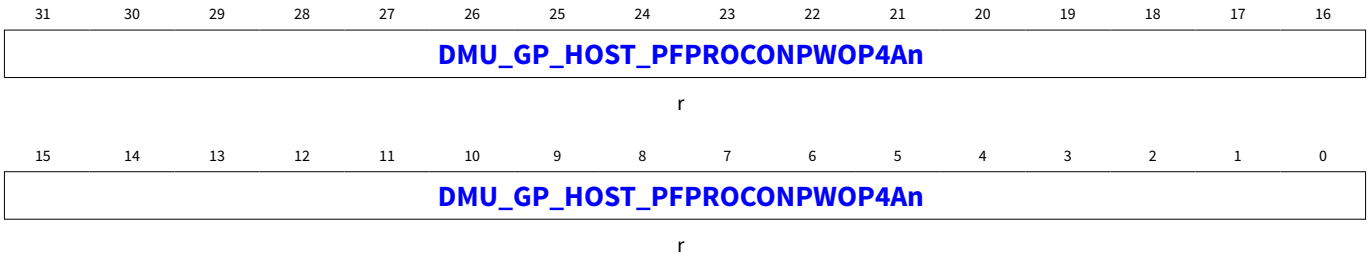


Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWOP3Bn	31:0	r	**DMU_GP_HOST_PFPROCONPWOP3Bn register value**

48.4.3.13.68 DMU_GP_HOST_PFPROCONPWOP4An, OTP set 6

UCB0_26_DMU_GP_HOST_PFPROCONPWOP4An (n=0-3) Offset address: $0D580_H+n*4$

DMU_GP_HOST_PFPROCONPWOP4An, OTP set 6 value: $0000\ 0000_H$



Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWOP4An	31:0	r	**DMU_GP_HOST_PFPROCONPWOP4An register value**

48.4.3.13.69 DMU_GP_HOST_PFPROCONPWOP5An, OTP set 6

UCB0_26_DMU_GP_HOST_PFPROCONPWOP5An (n=0-3) Offset address: $0D5A0_H+n*4$

DMU_GP_HOST_PFPROCONPWOP5An, OTP set 6 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONPWOP5An															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONPWOP5An															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWOP5An	31:0	r	**DMU_GP_HOST_PFPROCONPWOP5An register value**

48.4.3.13.70 DMU_GP_HOST_PFPROCONPWOP5Bn, OTP set 6

UCB0_26_DMU_GP_HOST_PFPROCONPWOP5Bn (n=0-3) Offset address: $0D5B0_H+n*4$

DMU_GP_HOST_PFPROCONPWOP5Bn, OTP set 6 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_HOST_PFPROCONPWOP5Bn															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_HOST_PFPROCONPWOP5Bn															
r															

Field	Bits	Type	Description
DMU_GP_HOST_PFPROCONPWOP5Bn	31:0	r	**DMU_GP_HOST_PFPROCONPWOP5Bn register value**

48.4.3.13.71 Confirmation CODE ORIGINAL Set_4, 32-bit

UCB0_26_CONFIRMATION_Set_4 Offset address: $0D7D0_H$

Confirmation CODE ORIGINAL Set_4, 32-bit value: $4321\ 1234_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CODE															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CODE															
rh															

Field	Bits	Type	Description
CODE	31:0	rh	**Confirmation Code** 43211234_H Confirmation state is UNLOCKED. $57B5327F_H$ Confirmation state is CONFIRMED.

48.4.3.13.72 Confirmation CODE ORIGINAL Set_5, 32-bit

UCB0_26_CONFIRMATION_Set_5 Offset address: $0D7E0_H$

Confirmation CODE ORIGINAL Set_5, 32-bit value: $4321\ 1234_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CODE															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CODE															
rh															

Field	Bits	Type	Description
CODE	31:0	rh	**Confirmation Code** 43211234_H Confirmation state is UNLOCKED. $57B5327F_H$ Confirmation state is CONFIRMED.

48.4.3.13.73 Confirmation CODE ORIGINAL Set_6, 32-bit

UCB0_26_CONFIRMATION_Set_6 Offset address: $0D7F0_H$

Confirmation CODE ORIGINAL Set_6, 32-bit value: $4321\ 1234_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CODE															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CODE															
rh															

Field	Bits	Type	Description
CODE	31:0	rh	**Confirmation Code** 43211234_H Confirmation state is UNLOCKED. $57B5327F_H$ Confirmation state is CONFIRMED.

48.4.3.14 Register overview - UCB0_35 (ascending offset address)

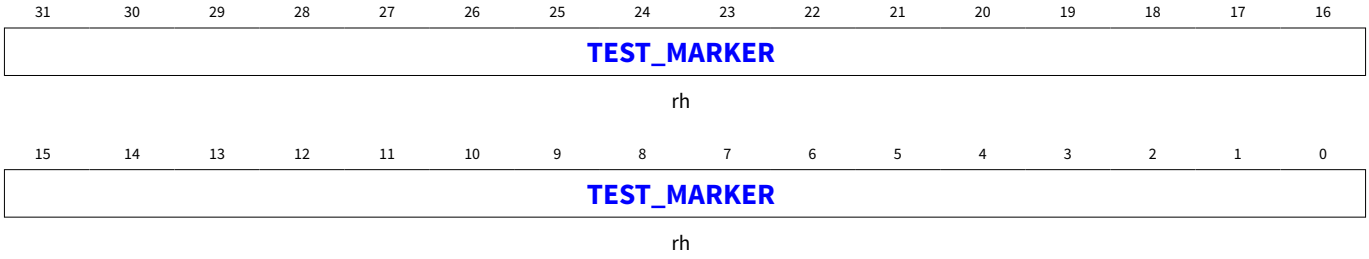
Table 2539 Register overview - UCB0_35 (ascending offset address)

Short name	Long name	Offset address	See
TEST_MARKER	TEST_PASS_MARKER	11900_H	8917

48.4.3.14.1 TEST_PASS_MARKER

TEST_MARKER Offset address: 11900_H

TEST_PASS_MARKER value: $0000\ 0000_H$



Field	Bits	Type	Description
TEST_MARKER	31:0	rh	**TEST_MARKER** 0x80658383 - Test PASSED else - Test FAILED

48.4.3.15 Register overview - UCB0_36 (ascending offset address)

Table 2540 Register overview - UCB0_36 (ascending offset address)

Short name	Long name	Offset address	See
UCID0	Unique Chip Identifier Word 0	12000_H	8918
UCID1	Unique Chip Identifier Word 1	12004_H	8918
UCID2	Unique Chip Identifier Word 2	12008_H	8919
BOM	Bill of materials	12010_H	8919
UCODE	Flash Microcode Version	12014_H	8920
DEVCFG_SBF_version	DEVCFG - SBF version	12018_H	8920
DEVCFG_PTE_version	DEVCFG - PTE version	$1201C_H$	8921

(table continues...)

Table 2540 (continued) Register overview - UCB0_36 (ascending offset address)

Short name	Long name	Offset address	See
UCB0_36_CONFIRMATION	Confirmation CODE ORIGINAL, 32-bit	$127F0_H$	
UCB0_36_CONFIRMATION_COPY	Confirmation CODE COPY, 32-bit	$127F8_H$	

Related information

Chip-unique identification on page 9005

48.4.3.15.1 Unique Chip Identifier Word 0

UCID0 Offset address: 12000_H

Unique Chip Identifier Word 0 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BLIND	X_coor								Y_coor						
r	r								r						

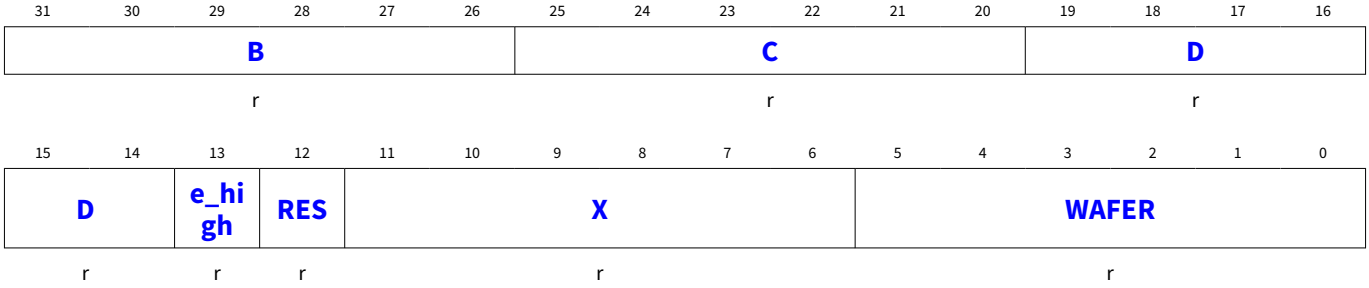
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Y_coor	e_low					RES		Man_ID							
r	r					r		r							

Field	Bits	Type	Description
Man_ID	7:0	r	**Manufacturer ID (0x40 for IFX)**
RES	9:8	r	**RESERVED**
e_low	14:10	r	**e[4:0]**
Y_coor	22:15	r	**Y coordinate wafer (invalid if BLIND=1)**
X_coor	30:23	r	**X coordinate wafer (invalid if BLIND=1)**
BLIND	31	r	**Indicator for wafer test or blind assembly device: 0=wafer tested; 1=blind assembled**

48.4.3.15.2 Unique Chip Identifier Word 1

UCID1 Offset address: 12004_H

Unique Chip Identifier Word 1 value: $0000\ 0000_H$

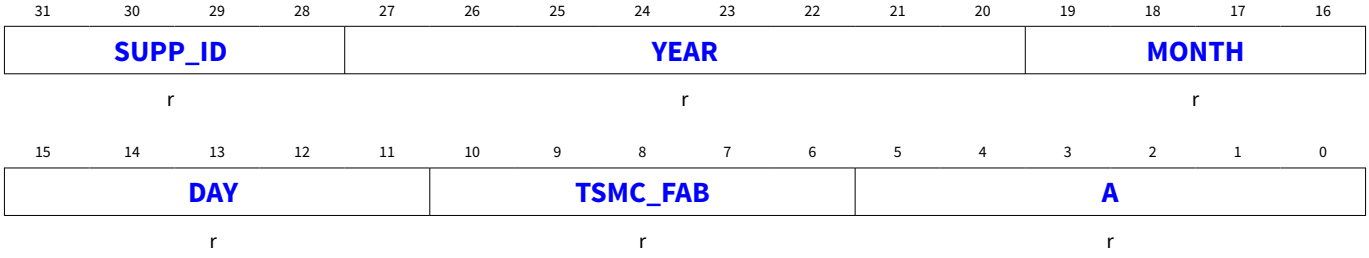


Field	Bits	Type	Description
WAFER	5:0	r	**Wafer Number (invalid if UCID0.BLIND=1)**
X	11:6	r	**Splitlot coding character x[5:0]**
RES	12	r	**RESERVED**
e_high	13	r	**e[5] (always 0)**
D	19:14	r	**Serial Number d[5:0]**
C	25:20	r	**c[5:0]**
B	31:26	r	**b[5:0]**

48.4.3.15.3 Unique Chip Identifier Word 2

UCID2 Offset address: 12008_H

Unique Chip Identifier Word 2 value: $0000\ 0000_H$



Field	Bits	Type	Description
A	5:0	r	**a[5:0]**
TSMC_FAB	10:6	r	**TSMC Fab ID (0b10100)**
DAY	15:11	r	**Day of production**
MONTH	19:16	r	**Month of production**
YEAR	27:20	r	**Year of production**
SUPP_ID	31:28	r	**Supplier ID (DD=0001, TSMC=0010 (65nm), TSMC=0011(28nm), GF=0100 (40nm))**

48.4.3.15.4 Bill of materials

BOM Offset address: 12010_H

Bill of materials value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BOM															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BOM															
r															

Field	Bits	Type	Description
BOM	31:0	r	**Bill of materials**

48.4.3.15.5 Flash Microcode Version

UCODE Offset address: 12014_H

Flash Microcode Version value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UCODE															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UCODE															
r															

Field	Bits	Type	Description
UCODE	31:0	r	**Flash Microcode Version**

48.4.3.15.6 DEVCFG - SBF version

DEVCFG_SBF_version Offset address: 12018_H

DEVCFG - SBF version value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Va_high				Va_low				Vb				Vc			
r				r				r				r			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Vy_high				Vx_low				Vy_high				Vy_low			
r				r				r				r			

Field	Bits	Type	Description
Vy_low	3:0	r	**Vy_low**
Vy_high	7:4	r	**Vy_high**
	15:12	r	**Vy_high**

(table continues...)

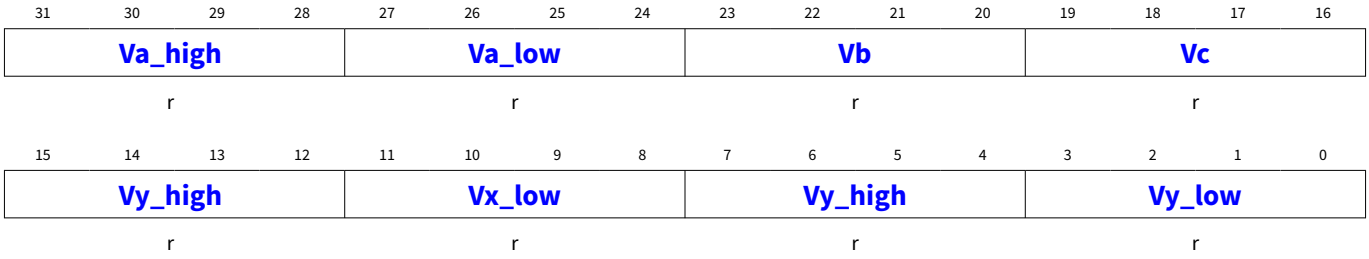
(continued)

Field	Bits	Type	Description
Vx_low	11:8	r	**Vx_low**
Vc	19:16	r	**Vc**
Vb	23:20	r	**Vb**
Va_low	27:24	r	**Va_low**
Va_high	31:28	r	**Va_high**

48.4.3.15.7 DEVCFG - PTE version

DEVCFG_PTE_version Offset address: $1201C_H$

DEVCFG - PTE version value: $0000\ 0000_H$



Field	Bits	Type	Description
Vy_low	3:0	r	**Vy_low**
Vy_high	7:4	r	**Vy_high**
	15:12	r	**Vy_high**
Vx_low	11:8	r	**Vx_low**
Vc	19:16	r	**Vc**
Vb	23:20	r	**Vb**
Va_low	27:24	r	**Va_low**
Va_high	31:28	r	**Va_high**

48.4.3.16 Register overview - UCB0_37 (ascending offset address)

Table 2541 Register overview - UCB0_37 (ascending offset address)

Short name	Long name	Offset address	See
UCB0_37_SAL	System Address of this Location	12800_H	
LENGTH1	Length of this table	12804_H	8927
UCB0_PMS_VMONP_VDDCON	VMONP_VDDCON register value	12808_H	
UCB0_PMS_VMONP_VDDRST	VMONP_VDDRST register value	$1280C_H$	

(table continues...)

Table 2541 (continued) Register overview - UCB0_37 (ascending offset address)

Short name	Long name	Offset address	See
UCB0_PMS_VMONP_VDDCLDRST	VMONP_VDDCLDRST register value	12810_H	
UCB0_PMS_VMONP_VDDEXTCON	PMS_VMONP_VDDEXTCON register value	12814_H	
UCB0_PMS_VMONP_VDDEXTRST	VMONP_VDDEXTRST register value	12818_H	
UCB0_PMS_VMONP_VSSDCHSCON	VMONP_VSSDCHSCON register value	$1281C_H$	
UCB0_PMS_VMONP_VSSDCHSRST	VMONP_VSSDCHSRST register value	12820_H	
UCB0_PMS_VMONP_VDDEXTDCCON	VMONP_VDDEXTDCCON register value	12824_H	
UCB0_PMS_VMONP_VDDEXTDCRST	VMONP_VDDEXTDCRST register value	12828_H	
UCB0_PMS_VMONP_VDDDCLSCON	VMONP_VDDDCLSCON register value	$1282C_H$	
UCB0_PMS_VMONP_VDDDCLSRST	VMONP_VDDDCLSRST register value	12830_H	
UCB0_PMS_VMONP_VDDEVRSBCON	VMONP_VDDEVRSBCON register value	12834_H	
UCB0_PMS_VMONP_VDDEVRSBRST	VMONP_VDDEVRSBRST register value	12838_H	
UCB0_PMS_VMONP_VDDMCON	VMONP_VDDMCON register value	$1283C_H$	
UCB0_PMS_VMONP_VDDMRST	VMONP_VDDMRST register value	12840_H	
UCB0_PMS_VMONP_VDDPHPHY0RST	VMONP_VDDPHPHY0RST register value	12844_H	
UCB0_PMS_OVERLAP_VDDPHPHY1PHRIF0RST	VMONP_VDDPHPHY1RST/VMONP_VDDPHRIF0RST register value	12848_H	
UCB0_PMS_OVERLAP_VDDPHPHY2P3NVMRST	VMONP_VDDPHPHY2RST/VMONP_VDDP3NVMRST register value	$1284C_H$	
UCB0_PMS_VMONP_VDDHSIFRST	VMONP_VDDHSIFRST register value	12850_H	
UCB0_PMS_VMONP_VDDPADRST	VMONP_VDDPADRST register value	12854_H	
UCB0_PMS_OVERLAP_VDDCPUALMU0RST	VMONP_VDDCPUARST/VMONP_VDDLMU0RST register value	12858_H	

(table continues...)

Table 2541 (continued) Register overview - UCB0_37 (ascending offset address)

Short name	Long name	Offset address	See
UCB0_PMS_OVERLAP_VDDCPUBLMU1RST	VMONP_VDDCPUBRST/VMONP_VDDLMU1RST register value	$1285C_H$	
UCB0_PMS_VMONP_VDDPPURST	VMONP_VDDPPURST register value	12860_H	
UCB0_PMS_VMONP_VDDPHY0RST	VMONP_VDDPHY0RST register value	12864_H	
UCB0_PMS_OVERLAP_VDDPHY1RIF0RST	VMONP_VDDPHY1RST/VMONP_VDDRIF0RST register value	12868_H	
UCB0_PMS_OVERLAP_VDDPHY2SBRST	VMONP_VDDPHY2RST/VMONP_VDDSBRST register value	$1286C_H$	
UCB0_PMS_VMONP_VDDSPARERST	VMONP_VDDSPARERST register value	12870_H	
UCB0_PMS_VMONP_VDDSPAREHVRST	VMONP_VDDSPAREHVRST register value	12874_H	
UCB0_PMS_VMONP_VDDSPAREHVCON	VMONP_VDDSPAREHVCON register value	12878_H	
UCB0_PMS_OVERLAP_VDDPHPHY2P3NVMCON	VMONP_VDDPHPHY2CON/VDDP3NVMCON register value	$1287C_H$	
UCB0_PMS_VMONP_TRKTRIM0	VMONP_TRKTRIM0 register value	12880_H	
UCB0_PMS_VMONP_TRKTRIM1	VMONP_TRKTRIM1 register value	12884_H	
UCB0_PMS_VMONP_TRKTRIM2	VMONP_TRKTRIM2 register value	12888_H	
UCB0_PMS_VMONS_VDDPMS0CON	VMONS_VDDPMS0CON register value	$1288C_H$	
UCB0_PMS_VMONS_VDDPMS1CON	VMONS_VDDPMS1CON register value	12890_H	
UCB0_PMS_VMONS_VDDPMS2CON	VMONS_VDDPMS2CON register value	12894_H	
UCB0_PMS_VMONS_LVLSEL	VMONS_LVLSEL register value	12898_H	
UCB0_PMS_VID_CON0	VID_CON0 register value	$1289C_H$	
UCB0_PMS_VID_CON1	VID_CON1 register value	$128A0_H$	
UCB0_PMS_HPOSCCON0	HPOSCCON0 register value	$128A4_H$	
UCB0_PMS_HPBGCON0	HPBGCON0 register value	$128A8_H$	

(table continues...)

Table 2541 (continued) Register overview - UCB0_37 (ascending offset address)

Short name	Long name	Offset address	See
UCB0_PMS_ADCCON	ADCCON register value	$128AC_H$	
UCB0_PMS_LPCON	LPCON register value	$128B0_H$	
UCB0_PMS_EVRC_CON0I	EVRC_CON0I register value	$128B4_H$	
UCB0_PMS_EVRC_CON0	EVRC_CON0 register value	$128B8_H$	
UCB0_PMS_EVRC_CON1	EVRC_CON1 register value	$128BC_H$	
UCB0_PMS_EVRC_CON2	EVRC_CON2 register value	$128C0_H$	
UCB0_PMS_EVRC_CON3	EVRC_CON3 register value	$128C4_H$	
UCB0_PMS_EVRC_CON4	EVRC_CON4 value	$128C8_H$	
UCB0_PMS_EVRC_COEFF0	EVRC_COEFF0 register value	$128CC_H$	
UCB0_PMS_EVRC_COEFF1	EVRC_COEFF1 register value	$128D0_H$	
UCB0_PMS_EVRC_COEFF2	EVRC_COEFF2 register value	$128D4_H$	
UCB0_PMS_EVRC_COEFF3	EVRC_COEFF3 register value	$128D8_H$	
UCB0_PMS_EVRC_COEFF4	EVRC_COEFF4 register value	$128DC_H$	
UCB0_PMS_EVRC_COEFF5	EVRC_COEFF5 register value	$128E0_H$	
UCB0_PMS_DTS_TRIM	PMS_DTS_TRIM register value	$128E4_H$	
UCB0_PMS_ALIAS_PVDDEXTDCCON	ALIAS_PVDDEXTDCCON register value	$128E8_H$	
UCB0_PMS_ALIAS_VDDEXTDCRST	ALIAS_VDDEXTDCRST register value	$128EC_H$	
UCB0_PMS_ALIAS_EVRCCON1	ALIAS_EVRCCON1 register value	$128F0_H$	
UCB0_PMS_ALIAS_EVRCCOEFF0	ALIAS_EVRCCOEFF0 register value	$128F4_H$	
UCB0_PMS_ALIAS_EVRCCOEFF1	ALIAS_EVRCCOEFF1 register value	$128F8_H$	
UCB0_PMS_ALIAS_EVRCCOEFF2	ALIAS_EVRCCOEFF2 register value	$128FC_H$	
UCB0_PMS_ALIAS_EVRCCOEFF3	ALIAS_EVRCCOEFF3 register value	12900_H	

(table continues...)

Table 2541 (continued) Register overview - UCB0_37 (ascending offset address)

Short name	Long name	Offset address	See
UCB0_PMS_ALIAS_EVRCCOEFF4	ALIAS_EVRCCOEFF4 register value	12904_H	
UCB0_PMS_HPOSCTRIM	PMS_HPOSCTRIM register value	12908_H	
UCB0_PMS_HPBGTRIM	PMS_HPBGTRIM register value	$1290C_H$	
UCB0_PMS_PMS0TRIM	PMS_PMS0TRIM register value	12910_H	
UCB0_PMS_DTS_CON	PMS_DTS_CON register value	12914_H	
CRC_RTCCFG1	CRC Checksum for values from RTCCFG1 area	12918_H	8954
UCB0_RTCCFG2	System Address of this Location	12960_H	8955
LENGTH2	Length of this table	12964_H	8955
UCB0_TRI_LBIST_CTRL_CFGA	TRI_LBIST_CTRL_CFGA register value	12968_H	
UCB0_TRI_LBIST_CTRL_CFGB	TRI_LBIST_CTRL_CFGB register value	$1296C_H$	
UCB0_TRI_LBIST_CTRL_CFGC	TRI_LBIST_CTRL_CFGC register value	12970_H	
UCB0_TRI_LBIST_CTRL_CFGD	TRI_LBIST_CTRL_CFGD register value	12974_H	
UCB0_TRI_LBIST_CTRL_CFGE	TRI_LBIST_CTRL_CFGE register value	12978_H	
UCB0_TRI_LBIST_CTRL_CFGF	TRI_LBIST_CTRL_CFGF register value	$1297C_H$	
UCB0_TRI_LBIST_CTRL_DOM	TRI_LBIST_CTRL_DOM register value	12980_H	
UCB0_HSPHY_DLL_DLLCTL0	HSPHY_DLL_DLLCTL0 register value	12984_H	
UCB0_HSPHY_DLL_DLLCTL1	HSPHY_DLL_DLLCTL1 register value	12988_H	
UCB0_HSPHY_PADTRIM	HSPHY_PADTRIM register value	$1298C_H$	
CRC_RTCCFG2	CRC Checksum for values from RTCCFG2 area	12990_H	8959
UCB0_RTCCFG3	System Address of this Location	$129F0_H$	8960
LENGTH3	Length of this table	$129F4_H$	8960
UCB0_P13_LPCR0	P13_LPCR0 register value	$129F8_H$	
UCB0_P13_LPCR1	P13_LPCR1 register value	$129FC_H$	
UCB0_P13_LPCR2	P13_LPCR2 register value	$12A00_H$	
UCB0_P13_LPCR3	P13_LPCR3 register value	$12A04_H$	

(table continues...)

Table 2541 (continued) Register overview - UCB0_37 (ascending offset address)

Short name	Long name	Offset address	See
UCB0_P14_LPCR5	P14_LPCR5 register value	$12A08_H$	
UCB0_P15_LPCR5	P15_LPCR5 register value	$12A0C_H$	
UCB0_P15_LPCR6	P15_LPCR6 register value	$12A10_H$	
UCB0_P21_LPCR0	P21_LPCR0 register value	$12A14_H$	
UCB0_P21_LPCR1	P21_LPCR1 register value	$12A18_H$	
UCB0_P21_LPCR2	P21_LPCR2 register value	$12A1C_H$	
UCB0_P22_LPCR0	P22_LPCR0 register value	$12A20_H$	
UCB0_P22_LPCR1	P22_LPCR1 register value	$12A24_H$	
UCB0_DMU_GP_PCONTROL	DMU_GP_PCONTROL	$12A28_H$	
UCB0_TCB	TCB Test constraint Register for Resistor trimming/default	$12A2C_H$	
UCB0_SCU_SPARE0	SCU_SPARE0 register value	$12A30_H$	
UCB0_SCU_SPARE1	SCU_SPARE1 register value	$12A34_H$	
CRC_RTCCFG3	CRC Checksum for values from RTCCFG3 area	$12A38_H$	8966
UCB0_RTCCFG4	System Address of this Location	$12A40_H$	8967
LENGTH4	Length of this table	$12A44_H$	8967
UCB0_P01_BGTRIM	P01_BGTRIM register value	$12A48_H$	
UCB0_P13_BGTRIM	P13_BGTRIM register value	$12A4C_H$	
UCB0_P14_BGTRIM	P14_BGTRIM register value	$12A50_H$	
UCB0_P21_BGTRIM	P21_BGTRIM register value	$12A54_H$	
UCB0_P22_BGTRIM	P22_BGTRIM register value	$12A58_H$	
UCB0_P30_BGTRIM	P30_BGTRIM register value	$12A5C_H$	
CRC_RTCCFG4	CRC Checksum for values from RTCCFG4 area	$12A60_H$	8970
UCB0_RTCCFG7	System Address of this Location	$12A90_H$	8970
LENGTH7	Length of this table	$12A94_H$	8970
UCB0_VTMON_DTS0_TRIM	VTMON_DTSx_TRIM (for x=0; x=1..4 if using broadcast)	$12A98_H$	
UCB0_VTMON_DTS1_TRIM	VTMON_DTS1_TRIM (if separate value, no broadcast)	$12A9C_H$	
UCB0_VTMON_DTS2_TRIM	VTMON_DTS2_TRIM (if separate value, no broadcast)	$12AA0_H$	
UCB0_VTMON_DTS3_TRIM	VTMON_DTS3_TRIM (if separate value, no broadcast)	$12AA4_H$	
UCB0_VTMON_DTS4_TRIM	VTMON_DTS4_TRIM (if separate value, no broadcast)	$12AA8_H$	
CRC_RTCCFG7	CRC Checksum for values from RTCCFG7 area	$12AAC_H$	8973

(table continues...)

Table 2541 (continued) Register overview - UCB0_37 (ascending offset address)

Short name	Long name	Offset address	See
UCB0_37_CONFIRMATION_ORIG	Confirmation CODE ORIGINAL, 32-bit	$12FF0_H$	
UCB0_37_CONFIRMATION_COPY	Confirmation CODE COPY, 32-bit	$12FF8_H$	

48.4.3.16.1 System Address of this Location

UCB0_RTCCFG1 Offset address: 12800_H

System Address of this Location value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RTCCFG1															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RTCCFG1															
r															

Field	Bits	Type	Description
RTCCFG1	31:0	r	**SAL value for reference 1 in UCB_RTCCFG**

48.4.3.16.2 Length of this table

LENGTH1 Offset address: 12804_H

Length of this table value: $0000\ 0047_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LENGTH1															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LENGTH1															
r															

Field	Bits	Type	Description
LENGTH1	31:0	r	**Length of this table**

48.4.3.16.3 VMONP_VDDCON register value

PMS_VMONP_VDDCON Offset address: 12808_H

VMONP_VDDCON register value value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UCB0_PMS_VMONP_VDDCON															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UCB0_PMS_VMONP_VDDCON															
rh															

Field	Bits	Type	Description
UCB0_PMS_VMONP_VDDCON	31:0	rh	**PMS_VMONP_VDDCON register value**

48.4.3.16.4 VMONP_VDDRST register value

PMS_VMONP_VDDRST Offset address: $1280C_H$

VMONP_VDDRST register value value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UCB0_PMS_VMONP_VDDRST															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UCB0_PMS_VMONP_VDDRST															
rh															

Field	Bits	Type	Description
UCB0_PMS_VMONP_VDDRST	31:0	rh	**PMS_VMONP_VDDRST register value**

48.4.3.16.5 VMONP_VDDCLDRST register value

PMS_VMONP_VDDCLDRST Offset address: 12810_H

VMONP_VDDCLDRST register value value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UCB0_PMS_VMONP_VDDCLDRST															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UCB0_PMS_VMONP_VDDCLDRST															
rh															

Field	Bits	Type	Description
UCB0_PMS_VMONP_VDDCLDRST	31:0	rh	**PMS_VMONP_VDDCLDRST register value**

48.4.3.16.6 PMS_VMONP_VDDEXTCON register value

PMS_VMONP_VDDEXTCON Offset address: 12814_H

PMS_VMONP_VDDEXTCON register value value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UCB0_PMS_VMONP_VDDEXTCON															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UCB0_PMS_VMONP_VDDEXTCON															
rh															

Field	Bits	Type	Description
UCB0_PMS_VMONP_VDDEXTCON	31:0	rh	**PMS_VMONP_VDDEXTCON register value**

48.4.3.16.7 VMONP_VDDEXTRST register value

PMS_VMONP_VDDEXTRST Offset address: 12818_H

VMONP_VDDEXTRST register value value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UCB0_PMS_VMONP_VDDEXTRST															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UCB0_PMS_VMONP_VDDEXTRST															
rh															

Field	Bits	Type	Description
UCB0_PMS_VMONP_VDDEXTRST	31:0	rh	**PMS_VMONP_VDDEXTRST register value**

48.4.3.16.8 VMONP_VSSDCHSCON register value

PMS_VMONP_VSSDCHSCON Offset address: $1281C_H$

VMONP_VSSDCHSCON register value value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UCB0_PMS_VMONP_VSSDCHSCON															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UCB0_PMS_VMONP_VSSDCHSCON															
r															

Field	Bits	Type	Description
UCB0_PMS_VMONP_VSSDCHSCON	31:0	r	**PMS_VMONP_VSSDCHSCON register value**

48.4.3.16.9 VMONP_VSSDCHSRST register value

PMS_VMONP_VSSDCHSRST Offset address: 12820_H

VMONP_VSSDCHSRST register value value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UCB0_PMS_VMONP_VSSDCHSRST															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UCB0_PMS_VMONP_VSSDCHSRST															
rh															

Field	Bits	Type	Description
UCB0_PMS_VMONP_VSSDCHSRST	31:0	rh	**PMS_VMONP_VSSDCHSRST register value**

48.4.3.16.10 VMONP_VDDEXTDCCON register value

PMS_VMONP_VDDEXTDCCON Offset address: 12824_H

VMONP_VDDEXTDCCON register value value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UCB0_PMS_VMONP_VDDEXTDCCON															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UCB0_PMS_VMONP_VDDEXTDCCON															
rh															

Field	Bits	Type	Description
UCB0_PMS_VMONP_VDDEXTDCCON	31:0	rh	**PMS_VMONP_VDDEXTDCCON register value**

48.4.3.16.11 VMONP_VDDEXTDCRST register value

PMS_VMONP_VDDEXTDCRST Offset address: 12828_H

VMONP_VDDEXTDCRST register value value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UCB0_PMS_VMONP_VDDEXTDCRST															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UCB0_PMS_VMONP_VDDEXTDCRST															
rh															

Field	Bits	Type	Description
UCB0_PMS_VMONP_VDDEXTDCRST	31:0	rh	**PMS_VMONP_VDDEXTDCRST register value**

48.4.3.16.12 VMONP_VDDDCLSCON register value

PMS_VMONP_VDDDCLSCON Offset address: $1282C_H$

VMONP_VDDDCLSCON register value value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UCB0_PMS_VMONP_VDDDCLSCON															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UCB0_PMS_VMONP_VDDDCLSCON															
rh															

Field	Bits	Type	Description
UCB0_PMS_VMONP_VDDDCLSCON	31:0	rh	**PMS_VMONP_VDDDCLSCON register value**

48.4.3.16.13 VMONP_VDDDCLSRST register value

PMS_VMONP_VDDDCLSRST Offset address: 12830_H

VMONP_VDDDCLSRST register value value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UCB0_PMS_VMONP_VDDDCLSRST															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UCB0_PMS_VMONP_VDDDCLSRST															
rh															

Field	Bits	Type	Description
UCB0_PMS_VMONP_VDDDCLSRST	31:0	rh	**PMS_VMONP_VDDDCLSRST register value**

48.4.3.16.14 VMONP_VDDEVRSBCON register value

PMS_VMONP_VDDEVRSBCON Offset address: 12834_H

VMONP_VDDEVRSBCON register value value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UCB0_PMS_VMONP_VDDEVRSBCON															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UCB0_PMS_VMONP_VDDEVRSBCON															
r															

Field	Bits	Type	Description
UCB0_PMS_VMONP_VDDEVRSBCON	31:0	r	**PMS_VMONP_VDDEVRSBCON register value**

48.4.3.16.15 VMONP_VDDEVRSBRST register value

PMS_VMONP_VDDEVRSBRST Offset address: 12838_H

VMONP_VDDEVRSBRST register value value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UCB0_PMS_VMONP_VDDEVRSBRST															
rh															

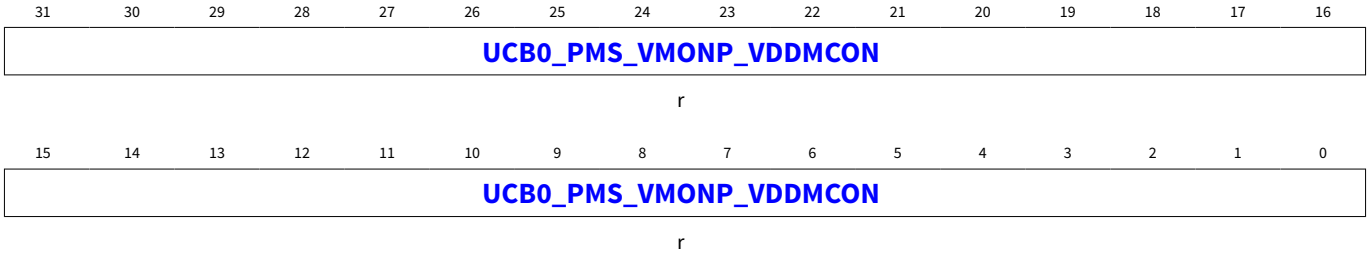
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UCB0_PMS_VMONP_VDDEVRSBRST															
rh															

Field	Bits	Type	Description
UCB0_PMS_VMONP_VDDEVRSBRST	31:0	rh	**PMS_VMONP_VDDEVRSBRST register value**

48.4.3.16.16 VMONP_VDDMCON register value

PMS_VMONP_VDDMCON Offset address: $1283C_H$

VMONP_VDDMCON register value value: $0000\ 0000_H$

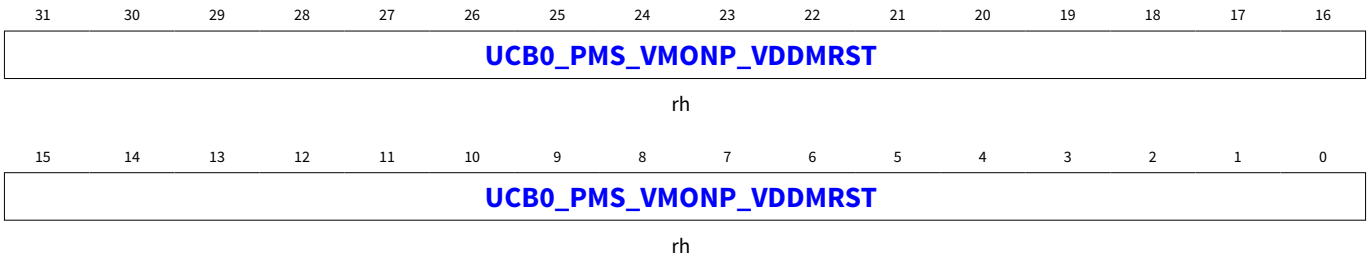


Field	Bits	Type	Description
UCB0_PMS_VMONP_VDDMCON	31:0	r	**PMS_VMONP_VDDMCON register value**

48.4.3.16.17 VMONP_VDDMRST register value

PMS_VMONP_VDDMRST Offset address: 12840_H

VMONP_VDDMRST register value value: $0000\ 0000_H$



Field	Bits	Type	Description
UCB0_PMS_VMONP_VDDMRST	31:0	rh	**PMS_VMONP_VDDMRST register value**

48.4.3.16.18 VMONP_VDDPHPHY0RST register value

PMS_VMONP_VDDPHPHY0RST Offset address: 12844_H

VMONP_VDDPHPHY0RST register value value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UCB0_PMS_VMONP_VDDPHPHY0RST															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UCB0_PMS_VMONP_VDDPHPHY0RST															
rh															

Field	Bits	Type	Description
UCB0_PMS_VMONP_VDDPHPHY0RST	31:0	rh	**PMS_VMONP_VDDPHPHY0RST register value**

48.4.3.16.19 VMONP_VDDPHPHY1RST/VMONP_VDDPHRIF0RST register value

PMS_OVERLAP_VDDPHPHY1PHRIF0RST Offset address: 12848_H

VMONP_VDDPHPHY1RST/VMONP_VDDPHRIF0RST register value value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UCB0_PMS_OVERLAP_VDDPHPHY1PHRIF0RST															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UCB0_PMS_OVERLAP_VDDPHPHY1PHRIF0RST															
rh															

Field	Bits	Type	Description
UCB0_PMS_OVERLAP_VDDPHPHY1PHRIF0RST	31:0	rh	**PMS_OVERLAP_VDDPHPHY1PHRIF0RST register value**

48.4.3.16.20 VMONP_VDDPHPHY2RST/VMONP_VDDP3NVMRST register value

PMS_OVERLAP_VDDPHPHY2P3NVMRST Offset address: $1284C_H$

VMONP_VDDPHPHY2RST/VMONP_VDDP3NVMRST register value value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UCB0_PMS_OVERLAP_VDDPHPHY2P3NVMRST															
rh															

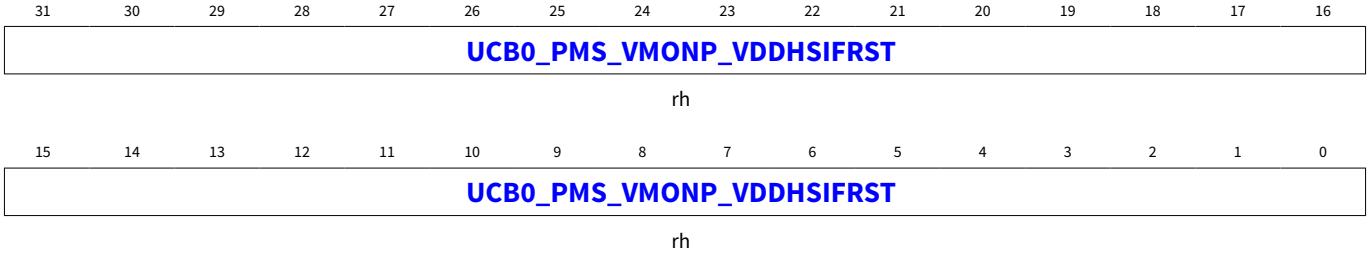
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UCB0_PMS_OVERLAP_VDDPHPHY2P3NVMRST															
rh															

Field	Bits	Type	Description
UCB0_PMS_OVERLAP_VDDPHPHY2P3NVMRST	31:0	rh	**PMS_OVERLAP_VDDPHPHY2P3NVMRST register value**

48.4.3.16.21 VMONP_VDDHSIFRST register value

PMS_VMONP_VDDHSIFRST Offset address: 12850_H

VMONP_VDDHSIFRST register value value: $0000\ 0000_H$

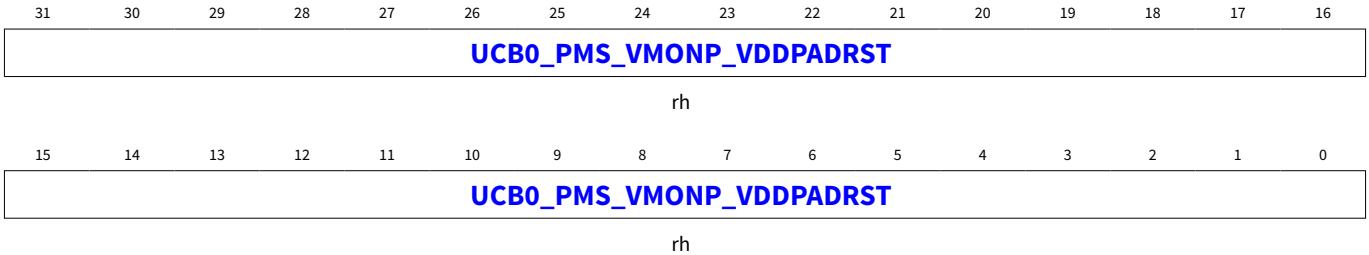


Field	Bits	Type	Description
UCB0_PMS_VMONP_VDDHSIFRST	31:0	rh	**PMS_VMONP_VDDHSIFRST register value**

48.4.3.16.22 VMONP_VDDPADRST register value

PMS_VMONP_VDDPADRST Offset address: 12854_H

VMONP_VDDPADRST register value value: $0000\ 0000_H$

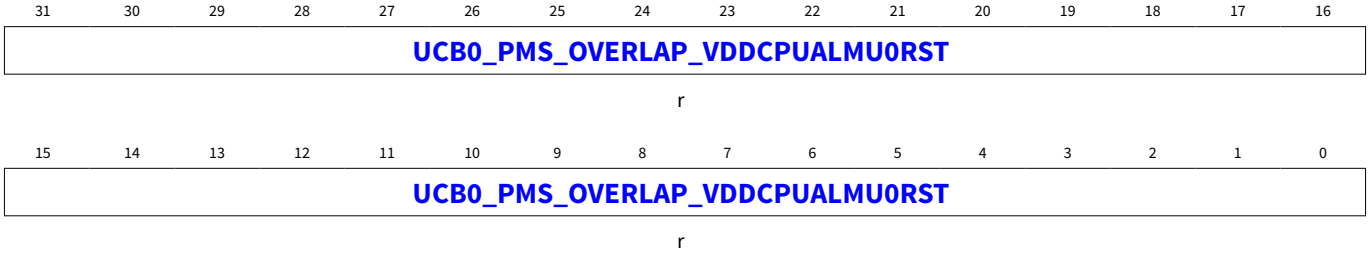


Field	Bits	Type	Description
UCB0_PMS_VMONP_VDDPADRST	31:0	rh	**PMS_VMONP_VDDPADRST register value**

48.4.3.16.23 VMONP_VDDCPUARST/VMONP_VDDLMU0RST register value

PMS_OVERLAP_VDDCPUALMU0RST Offset address: 12858_H

VMONP_VDDCPUARST/VMONP_VDDLMU0RST register value value: $0000\ 0000_H$

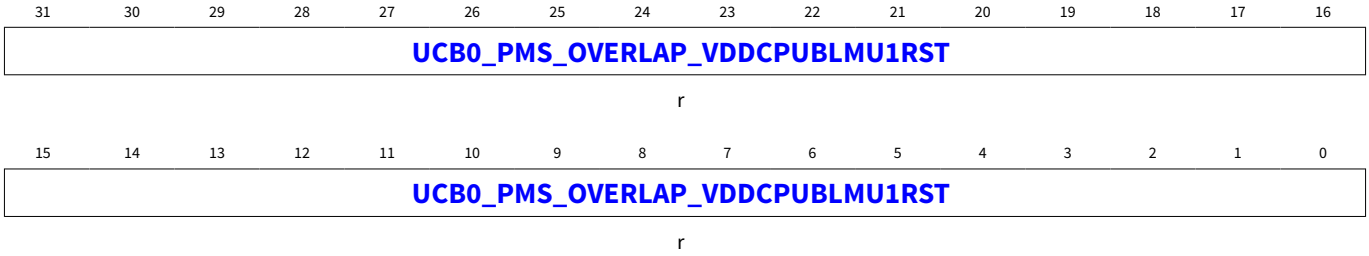


Field	Bits	Type	Description
UCB0_PMS_OVERLAP_VDDCPUALMU0RST	31:0	r	**PMS_OVERLAP_VDDCPUALMU0RST register value**

48.4.3.16.24 VMONP_VDDCPUBRST/VMONP_VDDLMU1RST register value

PMS_OVERLAP_VDDCPUBLMU1RST Offset address: $1285C_H$

VMONP_VDDCPUBRST/VMONP_VDDLMU1RST register value value: $0000\ 0000_H$



Field	Bits	Type	Description
UCB0_PMS_OVERLAP_VDDCPUBLMU1RST	31:0	r	**PMS_OVERLAP_VDDCPUBLMU1RST register value**

48.4.3.16.25 VMONP_VDDPPURST register value

PMS_VMONP_VDDPPURST Offset address: 12860_H

VMONP_VDDPPURST register value value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UCB0_PMS_VMONP_VDDPPURST															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UCB0_PMS_VMONP_VDDPPURST															
rh															

Field	Bits	Type	Description
UCB0_PMS_VMONP_VDDPPURST	31:0	rh	**PMS_VMONP_VDDPPURST register value**

48.4.3.16.26 VMONP_VDDPHY0RST register value

PMS_VMONP_VDDPHY0RST Offset address: 12864_H

VMONP_VDDPHY0RST register value value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UCB0_PMS_VMONP_VDDPHY0RST															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UCB0_PMS_VMONP_VDDPHY0RST															
rh															

Field	Bits	Type	Description
UCB0_PMS_VMONP_VDDPHY0RST	31:0	rh	**PMS_VMONP_VDDPHY0RST register value**

48.4.3.16.27 VMONP_VDDPHY1RST/VMONP_VDDRIF0RST register value

PMS_OVERLAP_VDDPHY1RIF0RST Offset address: 12868_H

VMONP_VDDPHY1RST/VMONP_VDDRIF0RST register value value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UCB0_PMS_OVERLAP_VDDPHY1RIF0RST															
rh															

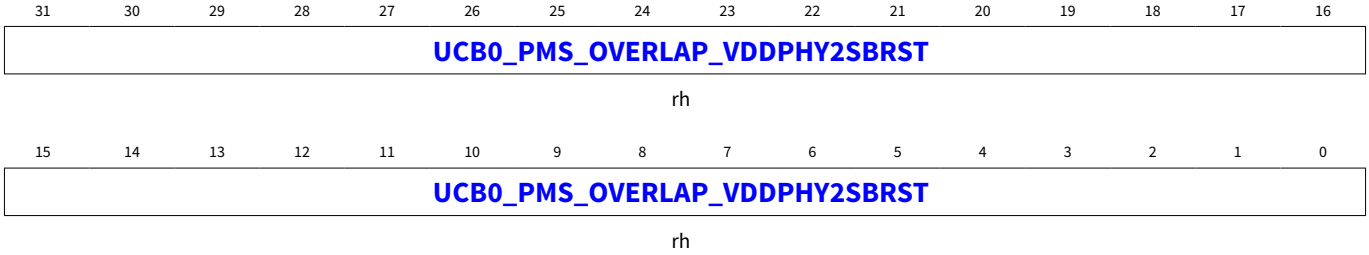
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UCB0_PMS_OVERLAP_VDDPHY1RIF0RST															
rh															

Field	Bits	Type	Description
UCB0_PMS_OVERLAP_VDDPHY1RIF0RST	31:0	rh	**PMS_OVERLAP_VDDPHY1RIF0RST register value**

48.4.3.16.28 VMONP_VDDPHY2RST/VMONP_VDDSBRST register value

PMS_OVERLAP_VDDPHY2SBRST Offset address: $1286C_H$

VMONP_VDDPHY2RST/VMONP_VDDSBRST register value value: $0000\ 0000_H$

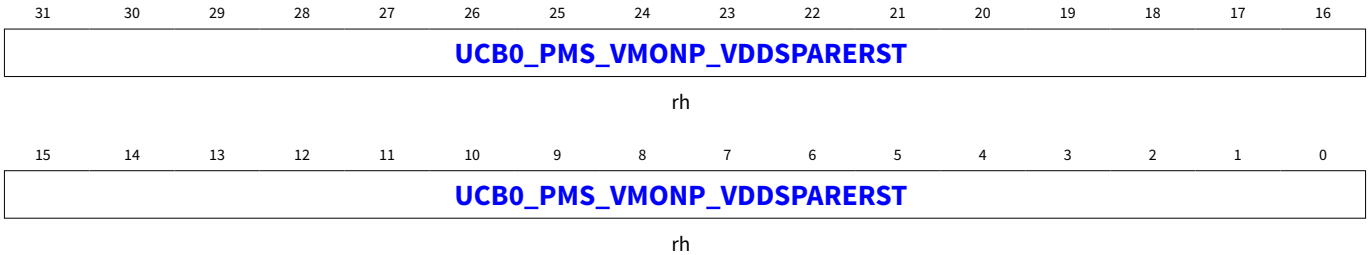


Field	Bits	Type	Description
UCB0_PMS_OVERLAP_VDDPHY2SBRST	31:0	rh	**PMS_OVERLAP_VDDPHY2SBRST register value**

48.4.3.16.29 VMONP_VDDSPARERST register value

PMS_VMONP_VDDSPARERST Offset address: 12870_H

VMONP_VDDSPARERST register value value: $0000\ 0000_H$

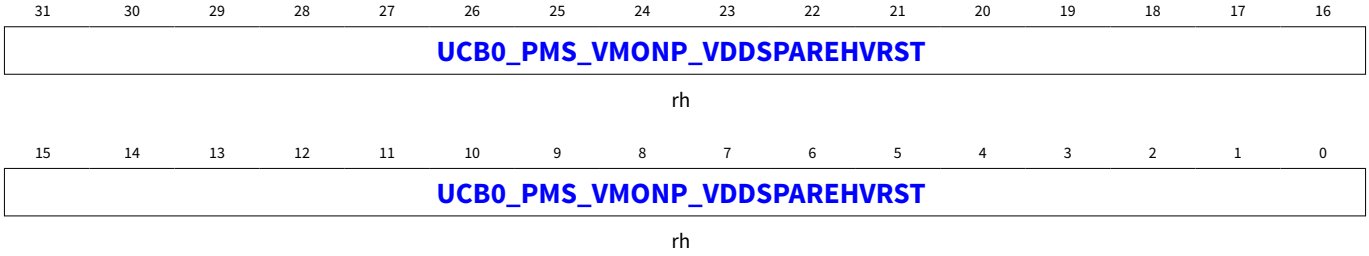


Field	Bits	Type	Description
UCB0_PMS_VMONP_VDDSPARERST	31:0	rh	**PMS_VMONP_VDDSPARERST register value**

48.4.3.16.30 VMONP_VDDSPAREHVRST register value

PMS_VMONP_VDDSPAREHVRST Offset address: 12874_H

VMONP_VDDSPAREHVRST register value value: $0000\ 0000_H$

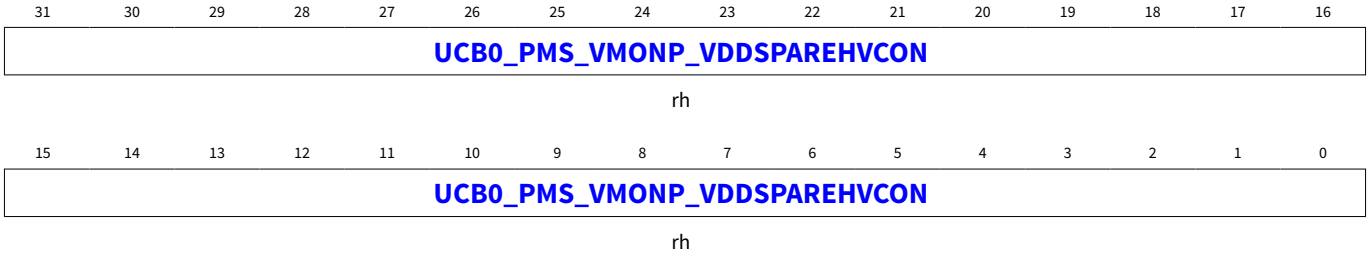


Field	Bits	Type	Description
UCB0_PMS_VMONP_VDDSPAREHVRST	31:0	rh	**PMS_VMONP_VDDSPAREHVRST register value**

48.4.3.16.31 VMONP_VDDSPAREHVCON register value

PMS_VMONP_VDDSPAREHVCON Offset address: 12878_H

VMONP_VDDSPAREHVCON register value value: $0000\ 0000_H$

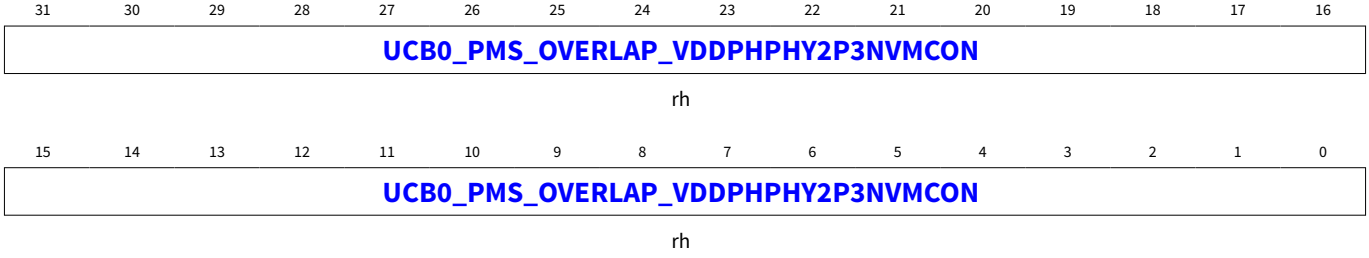


Field	Bits	Type	Description
UCB0_PMS_VMONP_VDDSPAREHVCON	31:0	rh	**PMS_VMONP_VDDSPAREHVCON register value**

48.4.3.16.32 VMONP_VDDPHPHY2CON/VDDP3NVMCON register value

PMS_OVERLAP_VDDPHPHY2P3NVMCON Offset address: $1287C_H$

VMONP_VDDPHPHY2CON/VDDP3NVMCON register value value: $0000\ 0000_H$



Field	Bits	Type	Description
UCB0_PMS_OVERLAP_VDDPHPHY2P3NVMCON	31:0	rh	**PMS_OVERLAP_VDDPHPHY2P3NVMCON register value**

48.4.3.16.33 VMONP_TRKTRIM0 register value

PMS_VMONP_TRKTRIM0 Offset address: 12880_H

VMONP_TRKTRIM0 register value value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UCB0_PMS_VMONP_TRKTRIM0															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UCB0_PMS_VMONP_TRKTRIM0															
rh															

Field	Bits	Type	Description
UCB0_PMS_VMONP_TRKTRIM0	31:0	rh	**PMS_VMONP_TRKTRIM0 register value**

48.4.3.16.34 VMONP_TRKTRIM1 register value

PMS_VMONP_TRKTRIM1 Offset address: 12884_H

VMONP_TRKTRIM1 register value value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UCB0_PMS_VMONP_TRKTRIM1															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UCB0_PMS_VMONP_TRKTRIM1															
rh															

Field	Bits	Type	Description
UCB0_PMS_VMONP_TRKTRIM1	31:0	rh	**PMS_VMONP_TRKTRIM1 register value**

48.4.3.16.35 VMONP_TRKTRIM2 register value

PMS_VMONP_TRKTRIM2 Offset address: 12888_H

VMONP_TRKTRIM2 register value value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UCB0_PMS_VMONP_TRKTRIM2															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UCB0_PMS_VMONP_TRKTRIM2															
rh															

Field	Bits	Type	Description
UCB0_PMS_VMONP_TRKTRIM2	31:0	rh	**PMS_VMONP_TRKTRIM2 register value**

48.4.3.16.36 VMONS_VDDPMS0CON register value

PMS_VMONS_VDDPMS0CON Offset address: $1288C_H$

VMONS_VDDPMS0CON register value value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UCB0_PMS_VMONS_VDDPMS0CON															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UCB0_PMS_VMONS_VDDPMS0CON															
rh															

Field	Bits	Type	Description
UCB0_PMS_VMONS_VDDPMS0CON	31:0	rh	**PMS_VMONS_VDDPMS0CON register value**

48.4.3.16.37 VMONS_VDDPMS1CON register value

PMS_VMONS_VDDPMS1CON Offset address: 12890_H

VMONS_VDDPMS1CON register value value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UCB0_PMS_VMONS_VDDPMS1CON															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UCB0_PMS_VMONS_VDDPMS1CON															
rh															

Field	Bits	Type	Description
UCB0_PMS_VMONS_VDDPMS1CON	31:0	rh	**PMS_VMONS_VDDPMS1CON register value**

48.4.3.16.38 VMONS_VDDPMS2CON register value

PMS_VMONS_VDDPMS2CON Offset address: 12894_H

VMONS_VDDPMS2CON register value value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UCB0_PMS_VMONS_VDDPMS2CON															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UCB0_PMS_VMONS_VDDPMS2CON															
rh															

Field	Bits	Type	Description
UCB0_PMS_VMONS_VDDPMS2CON	31:0	rh	**PMS_VMONS_VDDPMS2CON register value**

48.4.3.16.39 VMONS_LVLSEL register value

PMS_VMONS_LVLSEL Offset address: 12898_H

VMONS_LVLSEL register value value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UCB0_PMS_VMONS_LVLSEL															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UCB0_PMS_VMONS_LVLSEL															
rh															

Field	Bits	Type	Description
UCB0_PMS_VMONS_LVLSEL	31:0	rh	**PMS_VMONS_LVLSEL register value**

48.4.3.16.40 VID_CON0 register value

PMS_VID_CON0 Offset address: $1289C_H$

VID_CON0 register value value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UCB0_PMS_VID_CON0															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UCB0_PMS_VID_CON0															
rh															

Field	Bits	Type	Description
UCB0_PMS_VID_CON0	31:0	rh	**PMS_VID_CON0 register value**

48.4.3.16.41 VID_CON1 register value

PMS_VID_CON1 Offset address: $128A0_H$

VID_CON1 register value value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UCB0_PMS_VID_CON1															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UCB0_PMS_VID_CON1															
rh															

Field	Bits	Type	Description
UCB0_PMS_VID_CON1	31:0	rh	**PMS_VID_CON1 register value**

48.4.3.16.42 HPOSCCON0 register value

PMS_HPOSCCON0 Offset address: $128A4_H$

HPOSCCON0 register value value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UCB0_PMS_HPOSCCON0															
rh															

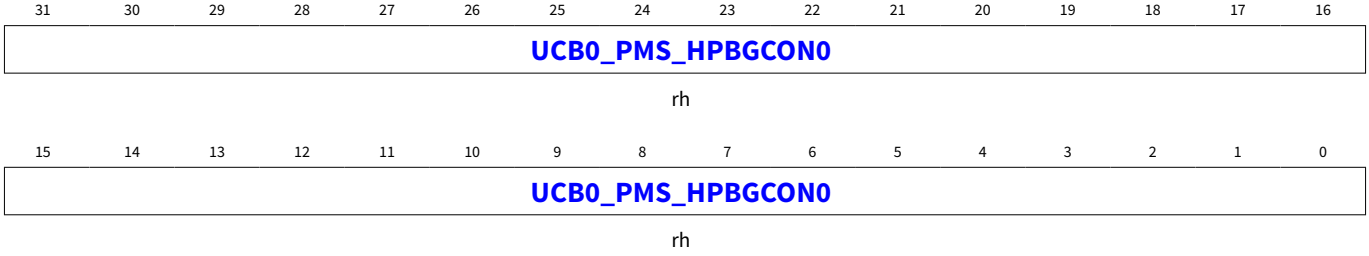
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UCB0_PMS_HPOSCCON0															
rh															

Field	Bits	Type	Description
UCB0_PMS_HPOSCCON0	31:0	rh	**PMS_HPOSCCON0 register value**

48.4.3.16.43 HPBGCON0 register value

PMS_HPBGCON0 Offset address: $128A8_H$

HPBGCON0 register value value: $0000\ 0000_H$

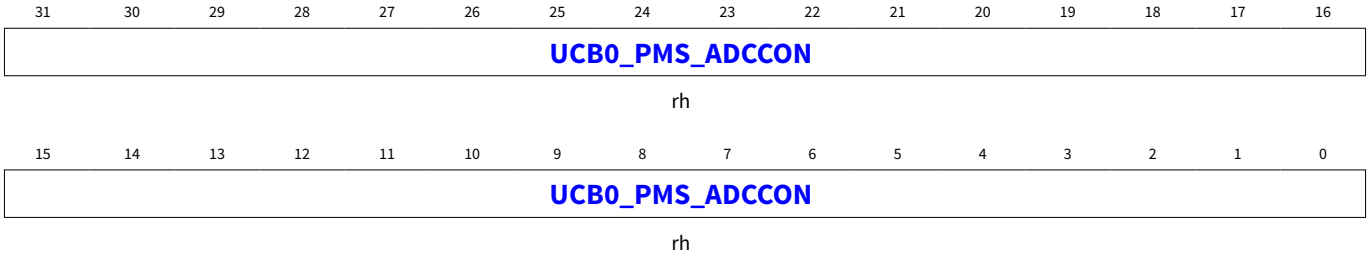


Field	Bits	Type	Description
UCB0_PMS_HPBGCON0	31:0	rh	**PMS_HPBGCON0 register value**

48.4.3.16.44 ADCCON register value

PMS_ADCCON Offset address: $128AC_H$

ADCCON register value value: $0000\ 0000_H$

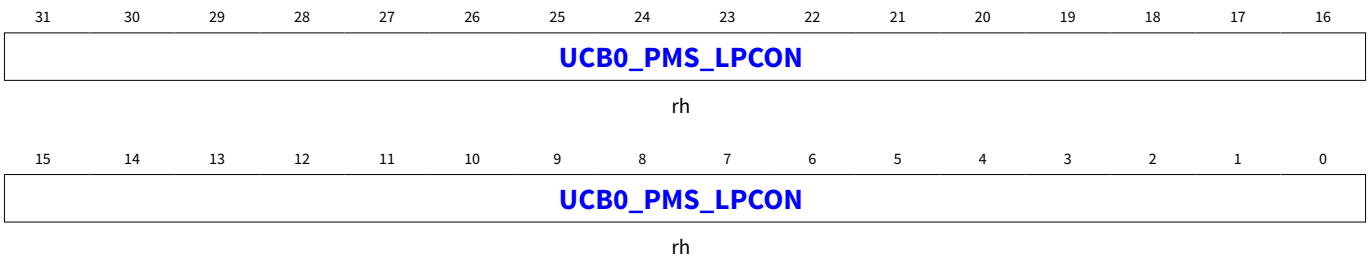


Field	Bits	Type	Description
UCB0_PMS_ADCCON	31:0	rh	**PMS_ADCCON register value**

48.4.3.16.45 LPCON register value

PMS_LPCON Offset address: $128B0_H$

LPCON register value value: $0000\ 0000_H$



Field	Bits	Type	Description
UCB0_PMS_LPCON	31:0	rh	**PMS_LPCON register value**

48.4.3.16.46 EVRC_CON0I register value

PMS_EVRC_CON0I Offset address: $128B4_H$

EVRC_CON0I register value value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UCB0_PMS_EVRC_CON0I															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UCB0_PMS_EVRC_CON0I															
rh															

Field	Bits	Type	Description
UCB0_PMS_EVRC_CON0I	31:0	rh	**PMS_EVRC_CON0I register value**

48.4.3.16.47 EVRC_CON0 register value

PMS_EVRC_CON0 Offset address: $128B8_H$

EVRC_CON0 register value value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UCB0_PMS_EVRC_CON0															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UCB0_PMS_EVRC_CON0															
rh															

Field	Bits	Type	Description
UCB0_PMS_EVRC_CON0	31:0	rh	**PMS_EVRC_CON0 register value**

48.4.3.16.48 EVRC_CON1 register value

PMS_EVRC_CON1 Offset address: $128BC_H$

EVRC_CON1 register value value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UCB0_PMS_EVRC_CON1															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UCB0_PMS_EVRC_CON1															
rh															

Field	Bits	Type	Description
UCB0_PMS_EVRC_CON1	31:0	rh	**PMS_EVRC_CON1 register value**

48.4.3.16.49 EVRC_CON2 register value

PMS_EVRC_CON2 Offset address: $128C0_H$

EVRC_CON2 register value value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UCB0_PMS_EVRC_CON2															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UCB0_PMS_EVRC_CON2															
rh															

Field	Bits	Type	Description
UCB0_PMS_EVRC_CON2	31:0	rh	**PMS_EVRC_CON2 register value**

48.4.3.16.50 EVRC_CON3 register value

PMS_EVRC_CON3 Offset address: $128C4_H$

EVRC_CON3 register value value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UCB0_PMS_EVRC_CON3															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UCB0_PMS_EVRC_CON3															
rh															

Field	Bits	Type	Description
UCB0_PMS_EVRC_CON3	31:0	rh	**PMS_EVRC_CON3 register value**

48.4.3.16.51 EVRC_CON4 value

PMS_EVRC_CON4 Offset address: $128C8_H$

EVRC_CON4 value value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UCB0_PMS_EVRC_CON4															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UCB0_PMS_EVRC_CON4															
rh															

Field	Bits	Type	Description
UCB0_PMS_EVRC_CON4	31:0	rh	**PMS_EVRC_CON4 register value**

48.4.3.16.52 EVRC_COEFF0 register value

PMS_EVRC_COEFF0 Offset address: $128CC_H$

EVRC_COEFF0 register value value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UCB0_PMS_EVRC_COEFF0															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UCB0_PMS_EVRC_COEFF0															
rh															

Field	Bits	Type	Description
UCB0_PMS_EVRC_COEFF0	31:0	rh	**PMS_EVRC_COEFF0 register value**

48.4.3.16.53 EVRC_COEFF1 register value

PMS_EVRC_COEFF1 Offset address: $128D0_H$

EVRC_COEFF1 register value value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UCB0_PMS_EVRC_COEFF1															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UCB0_PMS_EVRC_COEFF1															
rh															

Field	Bits	Type	Description
UCB0_PMS_EVRC_COEFF1	31:0	rh	**PMS_EVRC_COEFF1 register value**

48.4.3.16.54 EVRC_COEFF2 register value

PMS_EVRC_COEFF2 Offset address: $128D4_H$

EVRC_COEFF2 register value value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UCB0_PMS_EVRC_COEFF2															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UCB0_PMS_EVRC_COEFF2															
rh															

Field	Bits	Type	Description
UCB0_PMS_EVRC_COEFF2	31:0	rh	**PMS_EVRC_COEFF2 register value**

48.4.3.16.55 EVRC_COEFF3 register value

PMS_EVRC_COEFF3 Offset address: $128D8_H$

EVRC_COEFF3 register value value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UCB0_PMS_EVRC_COEFF3															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UCB0_PMS_EVRC_COEFF3															
rh															

Field	Bits	Type	Description
UCB0_PMS_EVRC_COEFF3	31:0	rh	**PMS_EVRC_COEFF3 register value**

48.4.3.16.56 EVRC_COEFF4 register value

PMS_EVRC_COEFF4 Offset address: $128DC_H$

EVRC_COEFF4 register value value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UCB0_PMS_EVRC_COEFF4															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UCB0_PMS_EVRC_COEFF4															
rh															

Field	Bits	Type	Description
UCB0_PMS_EVRC_COEFF4	31:0	rh	**PMS_EVRC_COEFF4 register value**

48.4.3.16.57 EVRC_COEFF5 register value

PMS_EVRC_COEFF5 Offset address: $128E0_H$

EVRC_COEFF5 register value value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UCB0_PMS_EVRC_COEFF5															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UCB0_PMS_EVRC_COEFF5															
rh															

Field	Bits	Type	Description
UCB0_PMS_EVRC_COEFF5	31:0	rh	**PMS_EVRC_COEFF5 register value**

48.4.3.16.58 PMS_DTS_TRIM register value

PMS_DTS_TRIM Offset address: $128E4_H$

PMS_DTS_TRIM register value value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UCB0_PMS_DTS_TRIM															
rh															

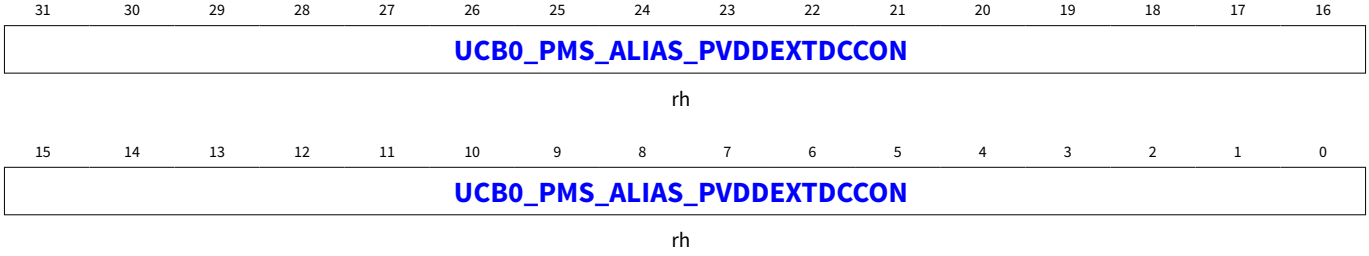
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UCB0_PMS_DTS_TRIM															
rh															

Field	Bits	Type	Description
UCB0_PMS_DTS_TRIM	31:0	rh	**PMS_DTS_TRIM register value**

48.4.3.16.59 ALIAS_PVDDEXTDCCON register value

PMS_ALIAS_PVDDEXTDCCON Offset address: $128E8_H$

ALIAS_PVDDEXTDCCON register value value: $0000\ 0000_H$

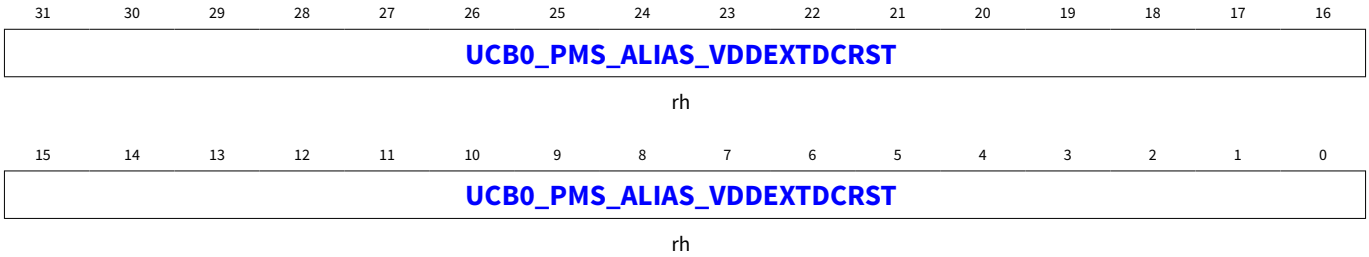


Field	Bits	Type	Description
UCB0_PMS_ALIAS_PVDDEXTDCCON	31:0	rh	**PMS_ALIAS_PVDDEXTDCCON register value**

48.4.3.16.60 ALIAS_VDDEXTDCRST register value

PMS_ALIAS_VDDEXTDCRST Offset address: $128EC_H$

ALIAS_VDDEXTDCRST register value value: $0000\ 0000_H$



Field	Bits	Type	Description
UCB0_PMS_ALIAS_VDDEXTDCRST	31:0	rh	**PMS_ALIAS_VDDEXTDCRST register value**

48.4.3.16.61 ALIAS_EVRCCON1 register value

PMS_ALIAS_EVRCCON1 Offset address: $128F0_H$

ALIAS_EVRCCON1 register value value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UCB0_PMS_ALIAS_EVRCCON1															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UCB0_PMS_ALIAS_EVRCCON1															
rh															

Field	Bits	Type	Description
UCB0_PMS_ALIAS_EVRCCON1	31:0	rh	**PMS_ALIAS_EVRCCON1 register value**

48.4.3.16.62 ALIAS_EVRCCOEFF0 register value

PMS_ALIAS_EVRCCOEFF0 Offset address: $128F4_H$

ALIAS_EVRCCOEFF0 register value value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UCB0_PMS_ALIAS_EVRCCOEFF0															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UCB0_PMS_ALIAS_EVRCCOEFF0															
rh															

Field	Bits	Type	Description
UCB0_PMS_ALIAS_EVRCCOEFF0	31:0	rh	**PMS_ALIAS_EVRCCOEFF0 register value**

48.4.3.16.63 ALIAS_EVRCCOEFF1 register value

PMS_ALIAS_EVRCCOEFF1 Offset address: $128F8_H$

ALIAS_EVRCCOEFF1 register value value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UCB0_PMS_ALIAS_EVRCCOEFF1															
rh															

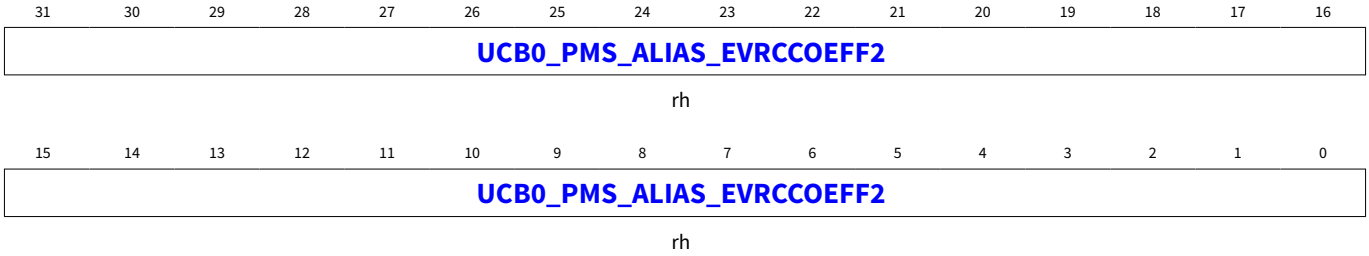
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UCB0_PMS_ALIAS_EVRCCOEFF1															
rh															

Field	Bits	Type	Description
UCB0_PMS_ALIAS_EVRCCOEFF1	31:0	rh	**PMS_ALIAS_EVRCCOEFF1 register value**

48.4.3.16.64 ALIAS_EVRCCOEFF2 register value

PMS_ALIAS_EVRCCOEFF2 Offset address: $128FC_H$

ALIAS_EVRCCOEFF2 register value value: $0000\ 0000_H$

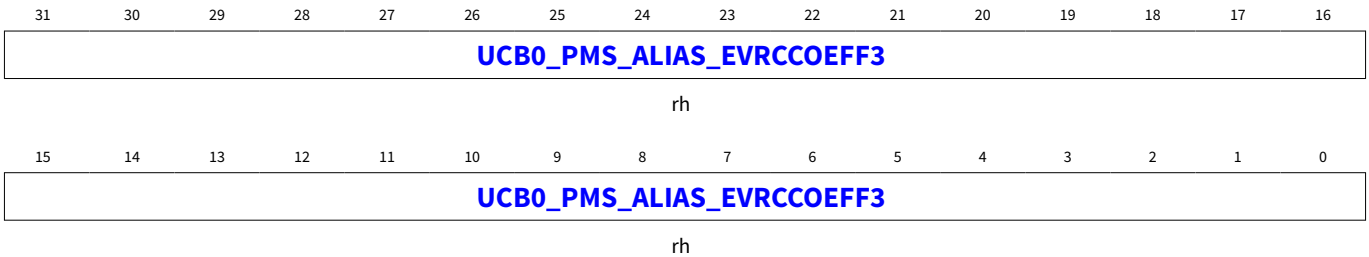


Field	Bits	Type	Description
UCB0_PMS_ALIAS_EVRCCOEFF2	31:0	rh	**PMS_ALIAS_EVRCCOEFF2 register value**

48.4.3.16.65 ALIAS_EVRCCOEFF3 register value

PMS_ALIAS_EVRCCOEFF3 Offset address: 12900_H

ALIAS_EVRCCOEFF3 register value value: $0000\ 0000_H$



Field	Bits	Type	Description
UCB0_PMS_ALIAS_EVRCCOEFF3	31:0	rh	**PMS_ALIAS_EVRCCOEFF3 register value**

48.4.3.16.66 ALIAS_EVRCCOEFF4 register value

PMS_ALIAS_EVRCCOEFF4 Offset address: 12904_H

ALIAS_EVRCCOEFF4 register value value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UCB0_PMS_ALIAS_EVRCCOEFF4															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UCB0_PMS_ALIAS_EVRCCOEFF4															
rh															

Field	Bits	Type	Description
UCB0_PMS_ALIAS_EVRCCOEFF4	31:0	rh	**PMS_ALIAS_EVRCCOEFF4 register value**

48.4.3.16.67 PMS_HPOSCTRIM register value

PMS_HPOSCTRIM Offset address: 12908_H

PMS_HPOSCTRIM register value value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UCB0_PMS_HPOSCTRIM															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UCB0_PMS_HPOSCTRIM															
rh															

Field	Bits	Type	Description
UCB0_PMS_HPOSCTRIM	31:0	rh	**PMS_HPOSCTRIM register value**

48.4.3.16.68 PMS_HPBGTRIM register value

PMS_HPBGTRIM Offset address: $1290C_H$

PMS_HPBGTRIM register value value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UCB0_PMS_HPBGTRIM															
rh															

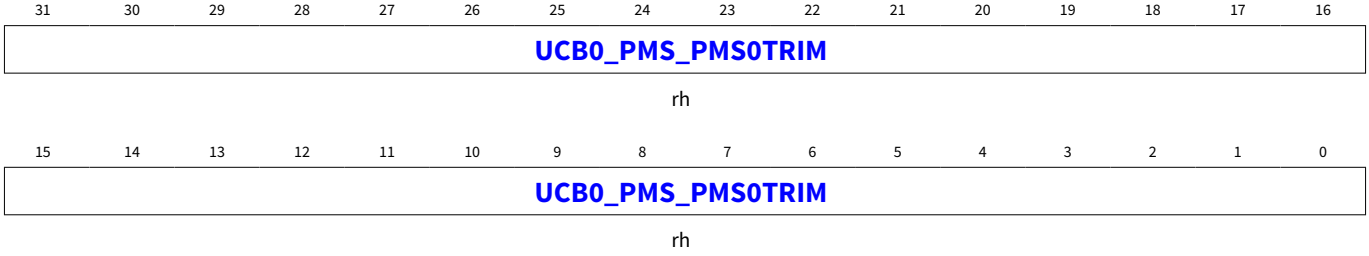
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UCB0_PMS_HPBGTRIM															
rh															

Field	Bits	Type	Description
UCB0_PMS_HPBGTRIM	31:0	rh	**PMS_HPBGTRIM register value**

48.4.3.16.69 PMS_PMS0TRIM register value

PMS_PMS0TRIM Offset address: 12910_H

PMS_PMS0TRIM register value value: $0000\ 0000_H$

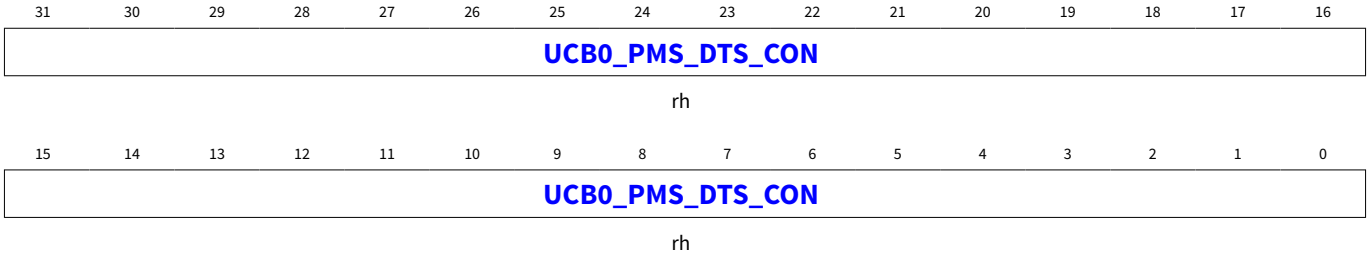


Field	Bits	Type	Description
UCB0_PMS_PMS0TRIM	31:0	rh	**PMS_PMS0TRIM register value**

48.4.3.16.70 PMS_DTS_CON register value

PMS_DTS_CON Offset address: 12914_H

PMS_DTS_CON register value value: $0000\ 0000_H$

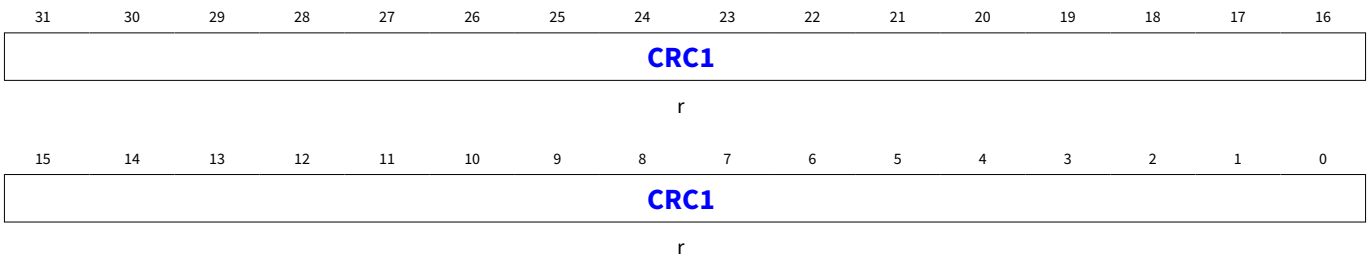


Field	Bits	Type	Description
UCB0_PMS_DTS_CON	31:0	rh	**PMS_DTS_CON register value**

48.4.3.16.71 CRC Checksum for values from RTCCFG1 area

CRC_RTCCFG1 Offset address: 12918_H

CRC Checksum for values from RTCCFG1 area value: $0000\ 0000_H$

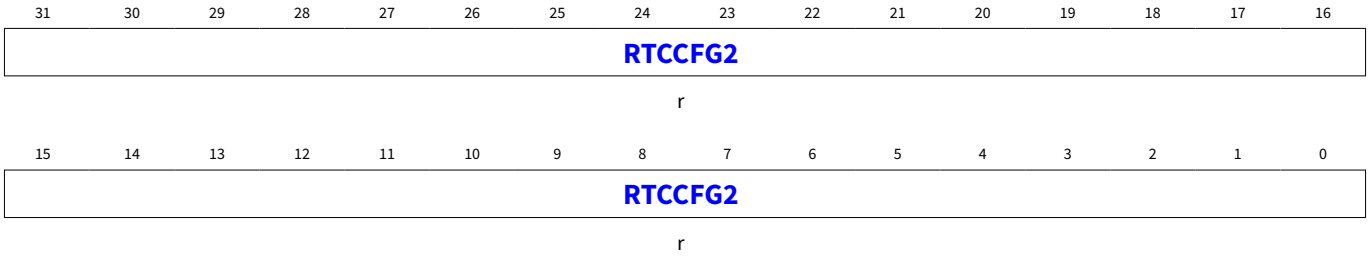


Field	Bits	Type	Description
CRC1	31:0	r	**Check Result** CRC1 check result.

48.4.3.16.72 System Address of this Location

UCB0_RTCCFG2 Offset address: 12960_H

System Address of this Location value: $0000\ 0000_H$

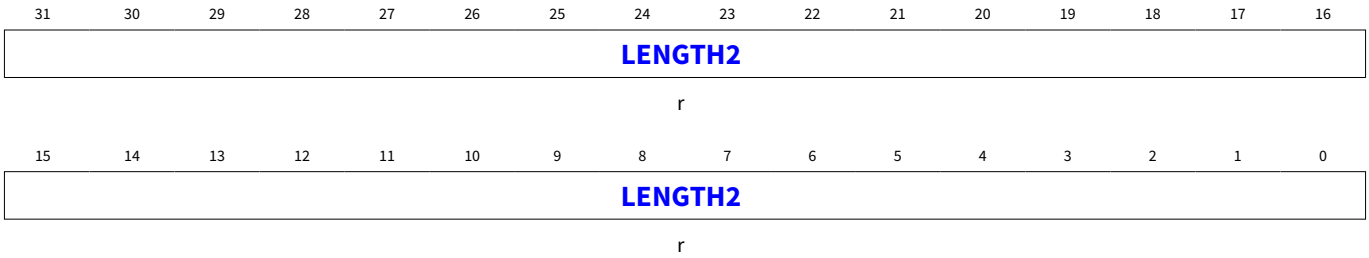


Field	Bits	Type	Description
RTCCFG2	31:0	r	**SAL value for reference 2 in UCB_RTCCFG**

48.4.3.16.73 Length of this table

LENGTH2 Offset address: 12964_H

Length of this table value: $0000\ 000D_H$



Field	Bits	Type	Description
LENGTH2	31:0	r	**Length of this table**

48.4.3.16.74 TRI_LBIST_CTRL_CFGA register value

TRI_LBIST_CTRL_CFGA Offset address: 12968_H

TRI_LBIST_CTRL_CFGA register value value: $0678\ 1388_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LBIST_CTRL_CFGA															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LBIST_CTRL_CFGA															
rh															

Field	Bits	Type	Description
LBIST_CTRL_CFGA	31:0	rh	**LBIST_CTRL_CFGA register value**

48.4.3.16.75 TRI_LBIST_CTRL_CFGB register value

TRI_LBIST_CTRL_CFGB Offset address: $1296C_H$

TRI_LBIST_CTRL_CFGB register value value: $334D\ 34A6_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LBIST_CTRL_CFGB															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LBIST_CTRL_CFGB															
rh															

Field	Bits	Type	Description
LBIST_CTRL_CFGB	31:0	rh	**LBIST_CTRL_CFGB register value**

48.4.3.16.76 TRI_LBIST_CTRL_CFGC register value

TRI_LBIST_CTRL_CFGC Offset address: 12970_H

TRI_LBIST_CTRL_CFGC register value value: $2E84\ 2835_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LBIST_CTRL_CFGC															
rh															

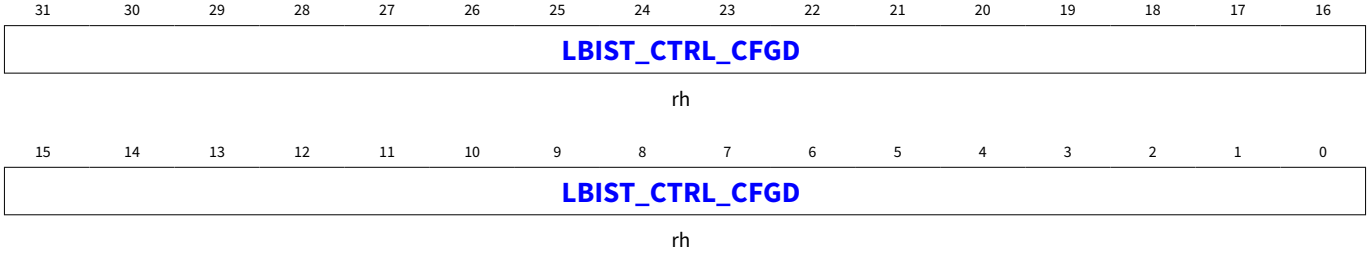
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LBIST_CTRL_CFGC															
rh															

Field	Bits	Type	Description
LBIST_CTRL_CFGC	31:0	rh	**LBIST_CTRL_CFGC register value**

48.4.3.16.77 TRI_LBIST_CTRL_CFGD register value

TRI_LBIST_CTRL_CFGD Offset address: 12974_H

TRI_LBIST_CTRL_CFGD register value value: $2835\ 2841_H$

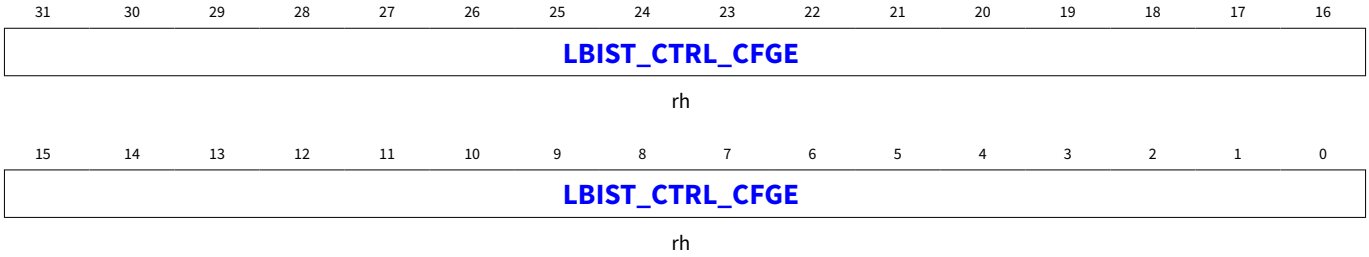


Field	Bits	Type	Description
LBIST_CTRL_CFGD	31:0	rh	**LBIST_CTRL_CFGD register value**

48.4.3.16.78 TRI_LBIST_CTRL_CFGE register value

TRI_LBIST_CTRL_CFGE Offset address: 12978_H

TRI_LBIST_CTRL_CFGE register value value: $0000\ 32A8_H$

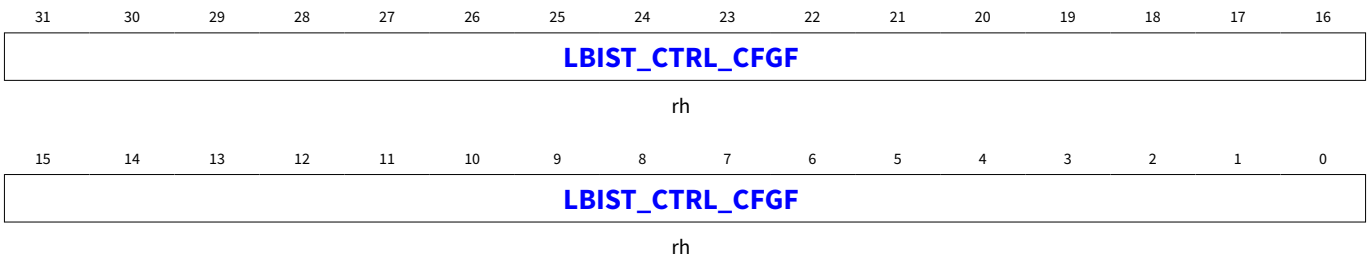


Field	Bits	Type	Description
LBIST_CTRL_CFGE	31:0	rh	**LBIST_CTRL_CFGE register value**

48.4.3.16.79 TRI_LBIST_CTRL_CFGF register value

TRI_LBIST_CTRL_CFGF Offset address: $1297C_H$

TRI_LBIST_CTRL_CFGF register value value: $0000\ 0003_H$

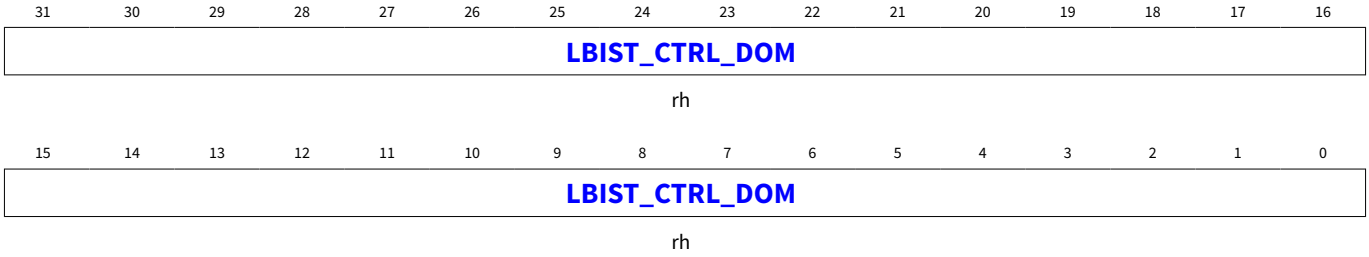


Field	Bits	Type	Description
LBIST_CTRL_CFGF	31:0	rh	**LBIST_CTRL_CFGF register value**

48.4.3.16.80 TRI_LBIST_CTRL_DOM register value

TRI_LBIST_CTRL_DOM Offset address: 12980_H

TRI_LBIST_CTRL_DOM register value value: $0000\ 007F_H$

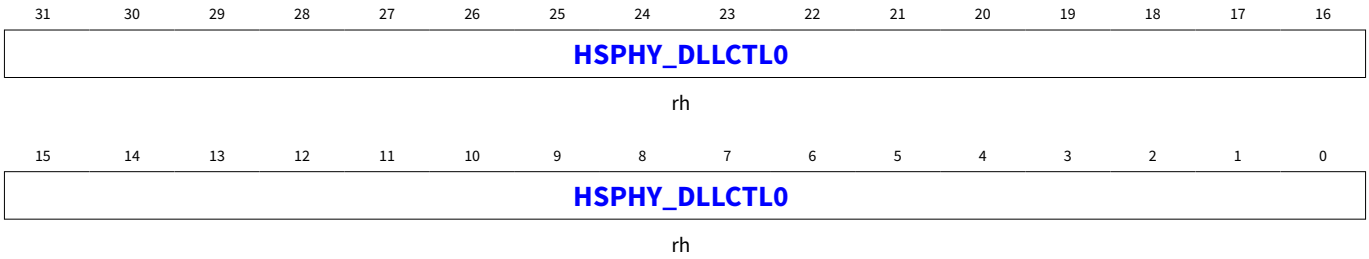


Field	Bits	Type	Description
LBIST_CTRL_DOM	31:0	rh	**LBIST_CTRL_DOM register value**

48.4.3.16.81 HSPHY_DLL_DLLCTL0 register value

HSPHY_DLL_DLLCTL0 Offset address: 12984_H

HSPHY_DLL_DLLCTL0 register value value: $0000\ 0000_H$



Field	Bits	Type	Description
HSPHY_DLLCTL0	31:0	rh	**HSPHY_DLLCTL0 register value**

48.4.3.16.82 HSPHY_DLL_DLLCTL1 register value

HSPHY_DLL_DLLCTL1 Offset address: 12988_H

HSPHY_DLL_DLLCTL1 register value value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
HSPHY_DLLCTL1															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HSPHY_DLLCTL1															
rh															

Field	Bits	Type	Description
HSPHY_DLLCTL1	31:0	rh	**HSPHY_DLLCTL1 register value**

48.4.3.16.83 HSPHY_PADTRIM register value

HSPHY_PADTRIM Offset address: $1298C_H$

HSPHY_PADTRIM register value value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
HSPHY_PADTRIM															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HSPHY_PADTRIM															
rh															

Field	Bits	Type	Description
HSPHY_PADTRIM	31:0	rh	**HSPHY_PADTRIM register value**

48.4.3.16.84 CRC Checksum for values from RTCCFG2 area

CRC_RTCCFG2 Offset address: 12990_H

CRC Checksum for values from RTCCFG2 area value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CRC2															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CRC2															
r															

Field	Bits	Type	Description
CRC2	31:0	r	**Check Result** CRC2 check result.

48.4.3.16.85 System Address of this Location

UCB0_RTCCFG3 Offset address: $129F0_H$

System Address of this Location value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RTCCFG3															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RTCCFG3															
r															

Field	Bits	Type	Description
RTCCFG3	31:0	r	**SAL value for reference 3 in UCB_RTCCFG**

48.4.3.16.86 Length of this table

LENGTH3 Offset address: $129F4_H$

Length of this table value: $0000\ 0013_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LENGTH3															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LENGTH3															
r															

Field	Bits	Type	Description
LENGTH3	31:0	r	**Length of this table**

48.4.3.16.87 P13_LPCR0 register value

P13_LPCR0 Offset address: $129F8_H$

P13_LPCR0 register value value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Pm_LPCRn															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pm_LPCRn															
r															

Field	Bits	Type	Description
Pm_LPCRn	31:0	r	**Pm_LPCRn register value**

48.4.3.16.88 P13_LPCR1 register value

P13_LPCR1 Offset address: $129FC_H$

P13_LPCR1 register value value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Pm_LPCRn															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pm_LPCRn															
r															

Field	Bits	Type	Description
Pm_LPCRn	31:0	r	**Pm_LPCRn register value**

48.4.3.16.89 P13_LPCR2 register value

P13_LPCR2 Offset address: $12A00_H$

P13_LPCR2 register value value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Pm_LPCRn															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pm_LPCRn															
r															

Field	Bits	Type	Description
Pm_LPCRn	31:0	r	**Pm_LPCRn register value**

48.4.3.16.90 P13_LPCR3 register value

P13_LPCR3 Offset address: $12A04_H$

P13_LPCR3 register value value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Pm_LPCRn															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pm_LPCRn															
r															

Field	Bits	Type	Description
Pm_LPCRn	31:0	r	**Pm_LPCRn register value**

48.4.3.16.91 P14_LPCR5 register value

P14_LPCR5 Offset address: $12A08_H$

P14_LPCR5 register value value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Pm_LPCRn															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pm_LPCRn															
r															

Field	Bits	Type	Description
Pm_LPCRn	31:0	r	**Pm_LPCRn register value**

48.4.3.16.92 P15_LPCR5 register value

P15_LPCR5 Offset address: $12A0C_H$

P15_LPCR5 register value value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Pm_LPCRn															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pm_LPCRn															
r															

Field	Bits	Type	Description
Pm_LPCRn	31:0	r	**Pm_LPCRn register value**

48.4.3.16.93 P15_LPCR6 register value

P15_LPCR6 Offset address: $12A10_H$

P15_LPCR6 register value value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Pm_LPCRn															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pm_LPCRn															
r															

Field	Bits	Type	Description
Pm_LPCRn	31:0	r	**Pm_LPCRn register value**

48.4.3.16.94 P21_LPCR0 register value

P21_LPCR0 Offset address: $12A14_H$

P21_LPCR0 register value value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Pm_LPCRn															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pm_LPCRn															
r															

Field	Bits	Type	Description
Pm_LPCRn	31:0	r	**Pm_LPCRn register value**

48.4.3.16.95 P21_LPCR1 register value

P21_LPCR1 Offset address: $12A18_H$

P21_LPCR1 register value value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Pm_LPCRn															
r															

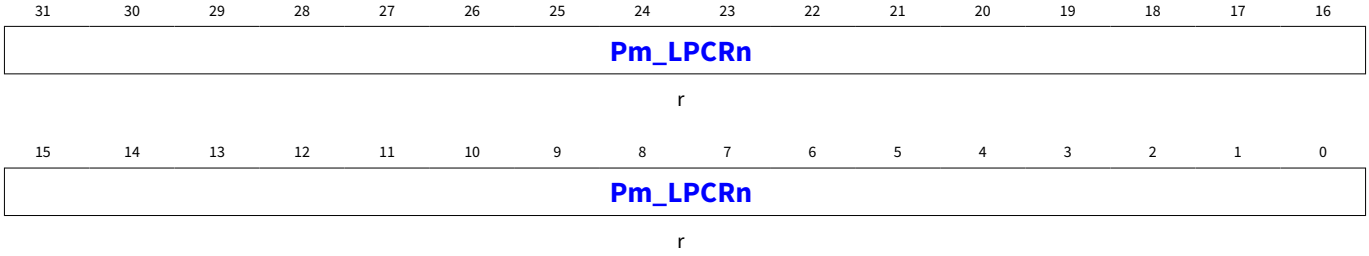
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pm_LPCRn															
r															

Field	Bits	Type	Description
Pm_LPCRn	31:0	r	**Pm_LPCRn register value**

48.4.3.16.96 P21_LPCR2 register value

P21_LPCR2 Offset address: $12A1C_H$

P21_LPCR2 register value value: $0000\ 0000_H$

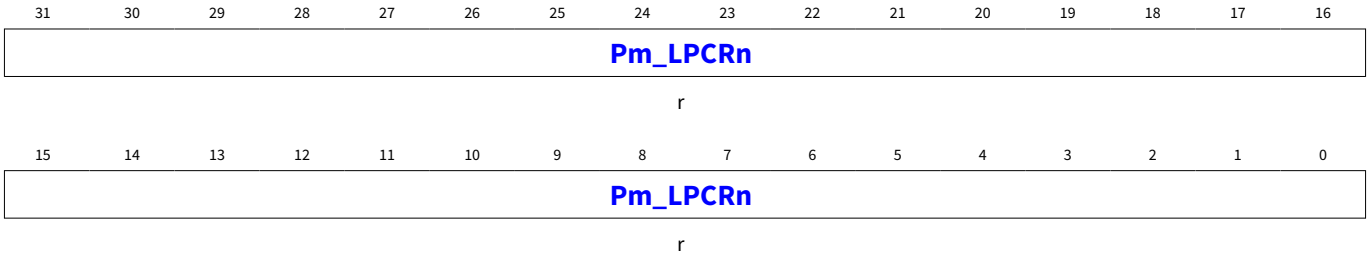


Field	Bits	Type	Description
Pm_LPCRn	31:0	r	**Pm_LPCRn register value**

48.4.3.16.97 P22_LPCR0 register value

P22_LPCR0 Offset address: $12A20_H$

P22_LPCR0 register value value: $0000\ 0000_H$

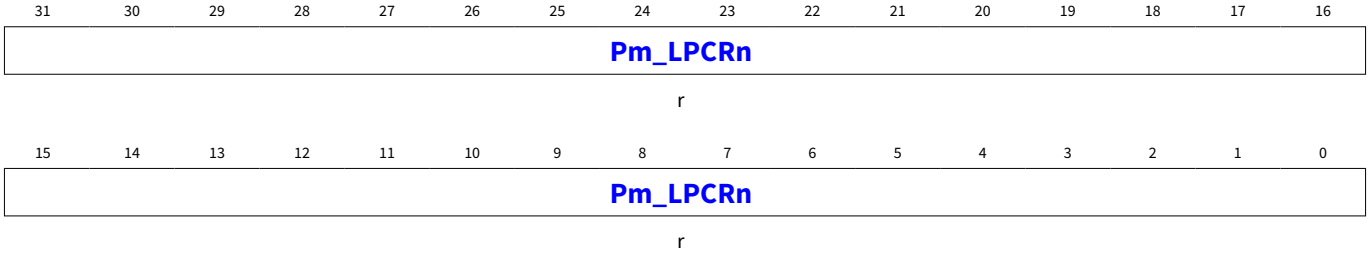


Field	Bits	Type	Description
Pm_LPCRn	31:0	r	**Pm_LPCRn register value**

48.4.3.16.98 P22_LPCR1 register value

P22_LPCR1 Offset address: $12A24_H$

P22_LPCR1 register value value: $0000\ 0000_H$

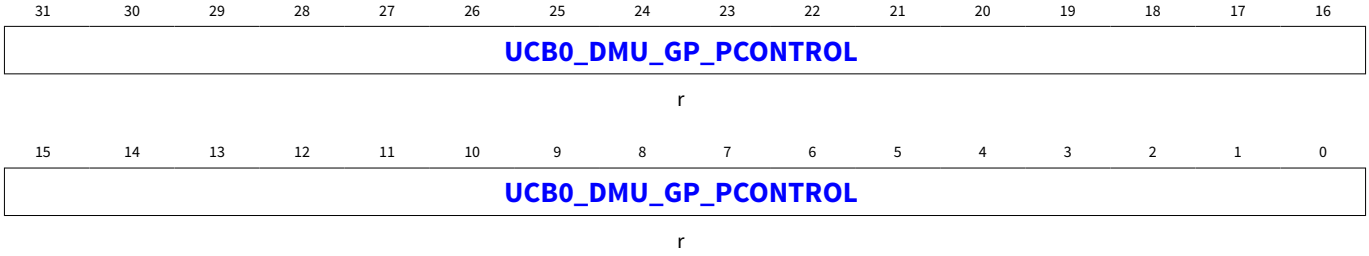


Field	Bits	Type	Description
Pm_LPCRn	31:0	r	**Pm_LPCRn register value**

48.4.3.16.99 DMU_GP_PCONTROL

DMU_GP_PCONTROL Offset address: $12A28_H$

DMU_GP_PCONTROL value: $0000\ 0000_H$

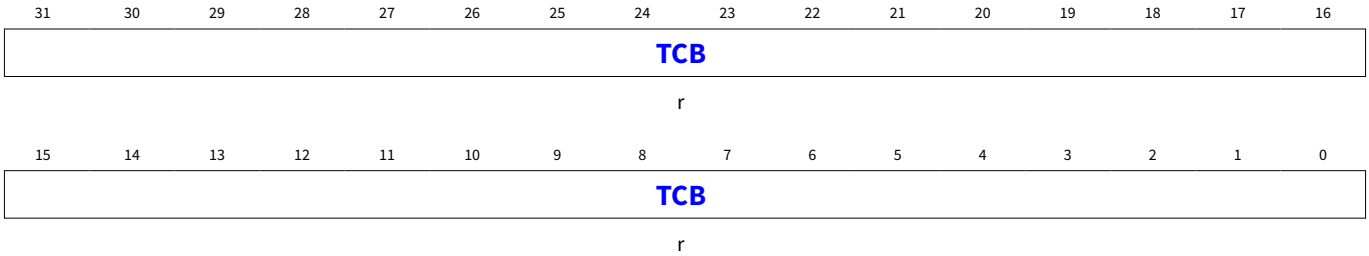


Field	Bits	Type	Description
UCB0_DMU_GP_PCONTROL	31:0	r	**DMU_GP_PCONTROL register value**

48.4.3.16.100 TCB Test constraint Register for Resistor trimming/default

TCB Offset address: $12A2C_H$

TCB Test constraint Register for Resistor trimming/ default value: $0000\ 0000_H$



Field	Bits	Type	Description
TCB	31:0	r	**TCB register value**

48.4.3.16.101 SCU_SPARE0 register value

SCU_SPARE0 Offset address: $12A30_H$

SCU_SPARE0 register value value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SCU_SPARE0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SCU_SPARE0															
r															

Field	Bits	Type	Description
SCU_SPARE0	31:0	r	**SCU_SPARE0 register value**

48.4.3.16.102 SCU_SPARE1 register value

SCU_SPARE1 Offset address: $12A34_H$

SCU_SPARE1 register value value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SCU_SPARE1															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SCU_SPARE1															
r															

Field	Bits	Type	Description
SCU_SPARE1	31:0	r	**SCU_SPARE1 register value**

48.4.3.16.103 CRC Checksum for values from RTCCFG3 area

CRC_RTCCFG3 Offset address: $12A38_H$

CRC Checksum for values from RTCCFG3 area value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CRC3															
r															

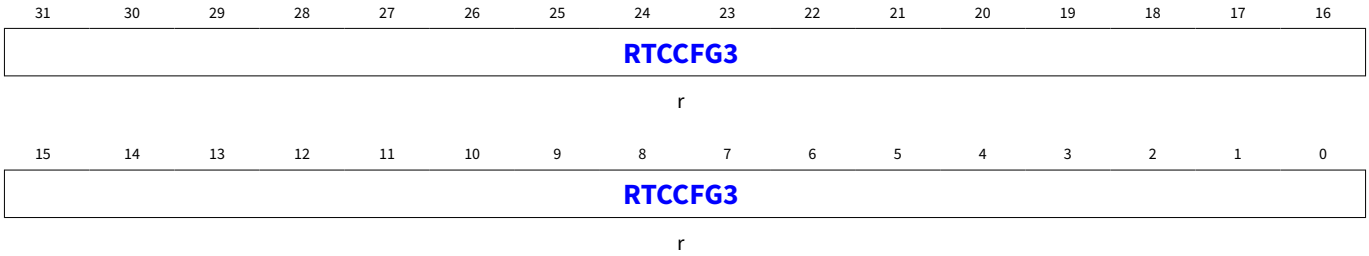
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CRC3															
r															

Field	Bits	Type	Description
CRC3	31:0	r	**Check Result** CRC3 check result.

48.4.3.16.104 System Address of this Location

UCB0_RTCCFG4 Offset address: $12A40_H$

System Address of this Location value: $0000\ 0000_H$

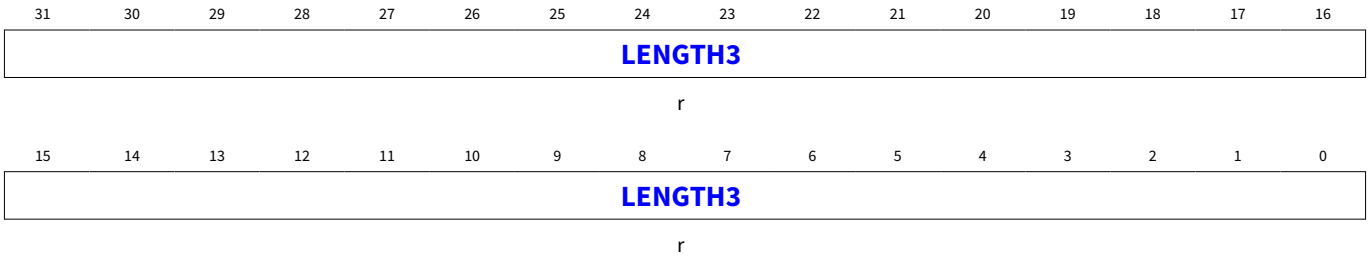


Field	Bits	Type	Description
RTCCFG3	31:0	r	**SAL value for reference 3 in UCB_RTCCFG**

48.4.3.16.105 Length of this table

LENGTH4 Offset address: $12A44_H$

Length of this table value: $0000\ 0009_H$

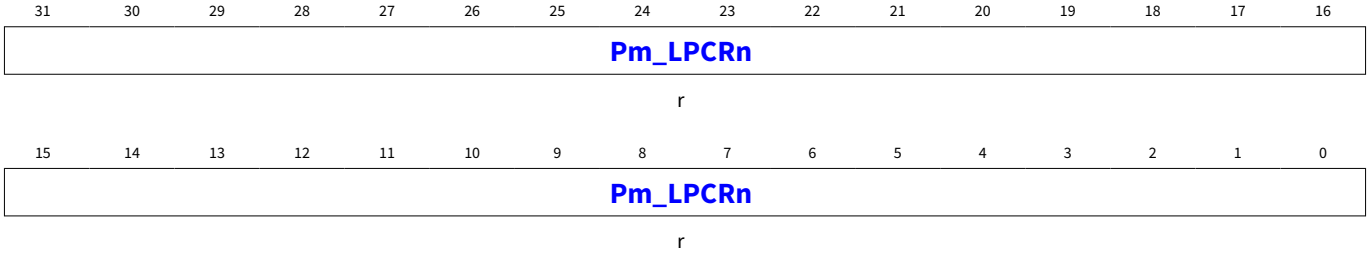


Field	Bits	Type	Description
LENGTH3	31:0	r	**Length of this table**

48.4.3.16.106 P01_BGTRIM register value

P01_BGTRIM Offset address: $12A48_H$

P01_BGTRIM register value value: $0000\ 0000_H$

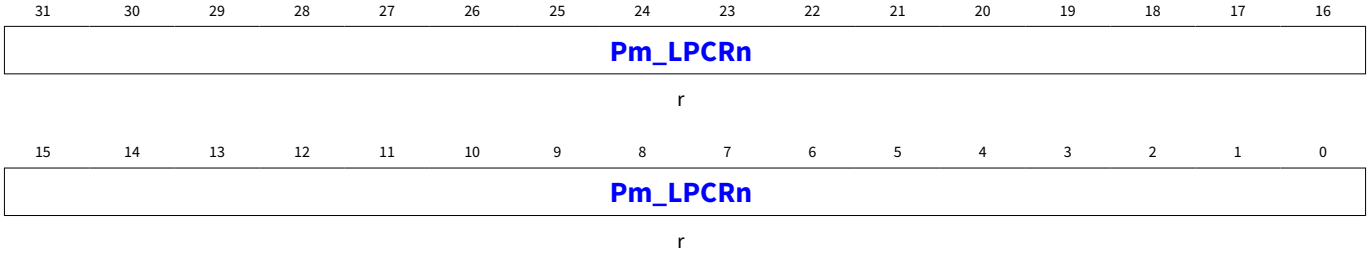


Field	Bits	Type	Description
Pm_LPCRn	31:0	r	**Pm_LPCRn register value**

48.4.3.16.107 P13_BGTRIM register value

P13_BGTRIM Offset address: $12A4C_H$

P13_BGTRIM register value value: $0000\ 0000_H$

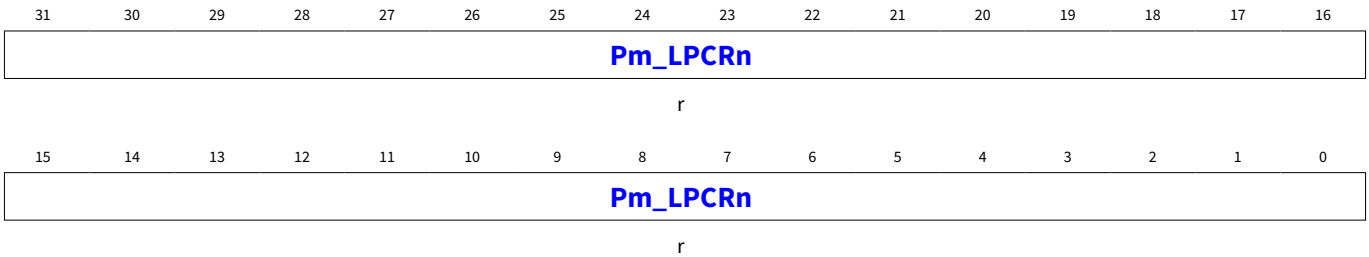


Field	Bits	Type	Description
Pm_LPCRn	31:0	r	**Pm_LPCRn register value**

48.4.3.16.108 P14_BGTRIM register value

P14_BGTRIM Offset address: $12A50_H$

P14_BGTRIM register value value: $0000\ 0000_H$



Field	Bits	Type	Description
Pm_LPCRn	31:0	r	**Pm_LPCRn register value**

48.4.3.16.109 P21_BGTRIM register value

P21_BGTRIM Offset address: $12A54_H$

P21_BGTRIM register value value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Pm_LPCRn															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pm_LPCRn															
r															

Field	Bits	Type	Description
Pm_LPCRn	31:0	r	**Pm_LPCRn register value**

48.4.3.16.110 P22_BGTRIM register value

P22_BGTRIM Offset address: $12A58_H$

P22_BGTRIM register value value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Pm_LPCRn															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pm_LPCRn															
r															

Field	Bits	Type	Description
Pm_LPCRn	31:0	r	**Pm_LPCRn register value**

48.4.3.16.111 P30_BGTRIM register value

P30_BGTRIM Offset address: $12A5C_H$

P30_BGTRIM register value value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Pm_LPCRn															
r															

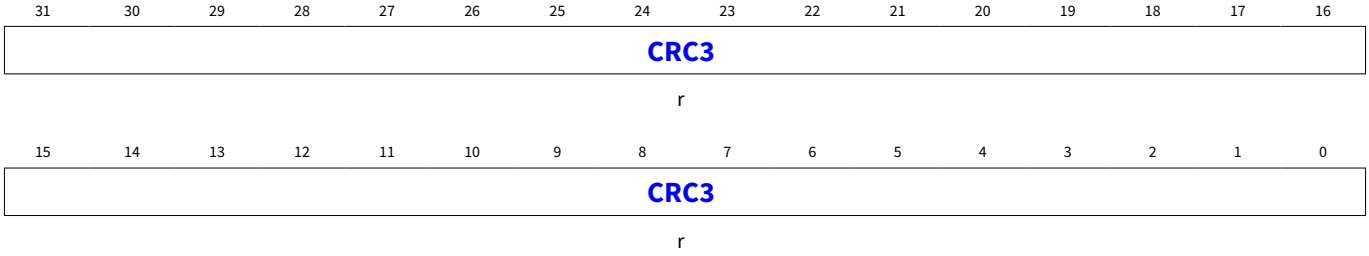
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pm_LPCRn															
r															

Field	Bits	Type	Description
Pm_LPCRn	31:0	r	**Pm_LPCRn register value**

48.4.3.16.112 CRC Checksum for values from RTCCFG4 area

CRC_RTCCFG4 Offset address: $12A60_H$

CRC Checksum for values from RTCCFG4 area value: $0000\ 0000_H$

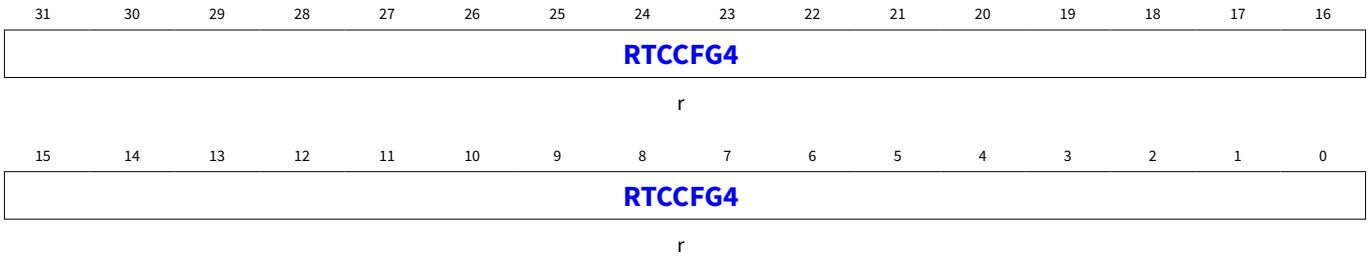


Field	Bits	Type	Description
CRC3	31:0	r	**Check Result** CRC3 check result.

48.4.3.16.113 System Address of this Location

UCB0_RTCCFG7 Offset address: $12A90_H$

System Address of this Location value: $0000\ 0000_H$



Field	Bits	Type	Description
RTCCFG4	31:0	r	**SAL value for reference 4 in UCB_RTCCFG**

48.4.3.16.114 Length of this table

LENGTH7 Offset address: $12A94_H$

Length of this table value: $0000\ 0008_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LENGTH3															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LENGTH3															
r															

Field	Bits	Type	Description
LENGTH3	31:0	r	**Length of this table**

48.4.3.16.115 VTMON_DTSx_TRIM (for x=0; x=1..4 if using broadcast)

VTMON_DTS0_TRIM Offset address: $12A98_H$

VTMON_DTSx_TRIM [for x=0; x=1..4 if using broadcast) value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
VTMON_DTS0_TRIM															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VTMON_DTS0_TRIM															
rh															

Field	Bits	Type	Description
VTMON_DTS0_TRIM	31:0	rh	**VTMON_DTS0_TRIM register value**

48.4.3.16.116 VTMON_DTS1_TRIM (if separate value, no broadcast)

VTMON_DTS1_TRIM Offset address: $12A9C_H$

VTMON_DTS1_TRIM [if separate value, no broadcast) value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
VTMON_DTS1_TRIM															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VTMON_DTS1_TRIM															
rh															

Field	Bits	Type	Description
VTMON_DTS1_TRIM	31:0	rh	**VTMON_DTS1_TRIM register value**

48.4.3.16.117 VTMON_DTS2_TRIM (if separate value, no broadcast)

VTMON_DTS2_TRIM Offset address: $12AA0_H$

VTMON_DTS2_TRIM [if separate value, no broadcast) value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
VTMON_DTS2_TRIM															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VTMON_DTS2_TRIM															
rh															

Field	Bits	Type	Description
VTMON_DTS2_TRIM	31:0	rh	**VTMON_DTS2_TRIM register value**

48.4.3.16.118 VTMON_DTS3_TRIM (if separate value, no broadcast)

VTMON_DTS3_TRIM Offset address: $12AA4_H$

VTMON_DTS3_TRIM [if separate value, no broadcast) value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
VTMON_DTS3_TRIM															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VTMON_DTS3_TRIM															
rh															

Field	Bits	Type	Description
VTMON_DTS3_TRIM	31:0	rh	**VTMON_DTS3_TRIM register value**

48.4.3.16.119 VTMON_DTS4_TRIM (if separate value, no broadcast)

VTMON_DTS4_TRIM Offset address: $12AA8_H$

VTMON_DTS4_TRIM [if separate value, no broadcast) value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
VTMON_DTS4_TRIM															
rh															

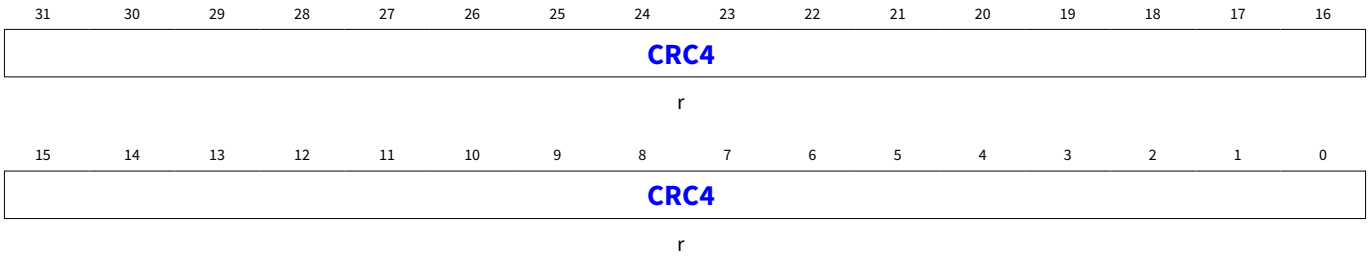
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VTMON_DTS4_TRIM															
rh															

Field	Bits	Type	Description
VTMON_DTS4_TRIM	31:0	rh	**VTMON_DTS4_TRIM register value**

48.4.3.16.120 CRC Checksum for values from RTCCFG7 area

CRC_RTCCFG7 Offset address: $12AAC_H$

CRC Checksum for values from RTCCFG7 area value: $0000\ 0000_H$



Field	Bits	Type	Description
CRC4	31:0	r	**Check Result** CRC4 check result.

48.4.3.17 Register overview - UCB0_38 (ascending offset address)

Table 2542 Register overview - UCB0_38 (ascending offset address)

Short name	Long name	Offset address	See
UCB0_38_SAL	System Address of this Location	13000_H	
TMADCx_HOT	Correction values dVDDK for TMADCx at HOT	13004_H+x*8	8974
TMADCx_COLD	Correction values dVDDK for TMADCx at COLD	13008_H+x*8	8975
CRC_ADCTRIM	Check Result for this table	13044_H	8975
UCB0_LBIST_RES	System Address of this Location	13048_H	8976
LBIST_STAT_RESx_KEYON	LBIST_STAT_RESx_KEYON reference value	$1304C_H$+x*4	8976
LBIST_STAT_RESx_KEYOFF	LBIST_STAT_RESx_KEYOFF reference value	13074_H+x*4	
CRC_LBIST_RES	Check Result for this table	$1309C_H$	8977
UCB0_38_SAL1	System Address of this Location	$130A0_H$	
VTMON_VMOND0_CONA	DVMON0 calibration value	$130A4_H$	8978
VTMON_VMOND1_CONA	DVMON1 calibration value	$130A8_H$	8979
VTMON_VMOND2_CONA	DVMON2 calibration value	$130AC_H$	8979

(table continues...)

Table 2542 (continued) Register overview - UCB0_38 (ascending offset address)

Short name	Long name	Offset address	See
VTMON_VMOND3_CONA	DVMON3 calibration value	$130B0_H$	8979
VTMON_VMOND4_CONA	DVMON4 calibration value	$130B4_H$	8980
VTMON_VMOND5_CONA	DVMON5 calibration value	$130B8_H$	8980
VTMON_VMOND6_CONA	DVMON6 calibration value	$130BC_H$	8980
VTMON_VMOND7_CONA	DVMON7 calibration value	$130C0_H$	8981
CRC_RTCCFG8	CRC Checksum for values from RTCCFG8 area	$130C4_H$	8981
UCB0_38_CONFIRMATION	Confirmation CODE ORIGINAL, 32-bit	$137F0_H$	
UCB0_38_CONFIRMATION_COPY	Confirmation CODE COPY, 32-bit	$137F8_H$	

48.4.3.17.1 System Address of this Location

UCB0_ADCTRIM Offset address: 13000_H

System Address of this Location value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SAL_ADCTRIM															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SAL_ADCTRIM															
r															

Field	Bits	Type	Description
SAL_ADCTRIM	31:0	r	**SAL value for UCB_ADCTRIM**

48.4.3.17.2 Correction values dVDDK for TMADCx at HOT

TMADCx_HOT (x=0-7) Offset address: 13004_H+x*8

Correction values dVDDK for TMADCx at HOT value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
dVDDK1_HOT															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
dVDDK0_HOT															
r															

Field	Bits	Type	Description
dVDDK0_HOT	15:0	r	**dVDDK0 for TMADCx at HOT temperature**
dVDDK1_HOT	31:16	r	**dVDDK1 for TMADCx at HOT temperature**

48.4.3.17.3 Correction values dVDDK for TMADCx at COLD

TMADCx_COLD (x=0-7) Offset address: 13008_H+x*8

Correction values dVDDK for TMADCx at COLD value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
dVDDK1_COLD															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
dVDDK0_COLD															
r															

Field	Bits	Type	Description
dVDDK0_COLD	15:0	r	**dVDDK0 for TMADCx at COLD temperature**
dVDDK1_COLD	31:16	r	**dVDDK1 for TMADCx at COLD temperature**

48.4.3.17.4 Check Result for this table

CRC_ADCTRIM Offset address: 13044_H

Check Result for this table value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CRC															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CRC															
r															

Field	Bits	Type	Description
CRC	31:0	r	**Check Result** CRC check result.

48.4.3.17.5 System Address of this Location

UCB0_LBIST_RES Offset address: 13048_H

System Address of this Location value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SAL_LBIST_RES															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SAL_LBIST_RES															
r															

Field	Bits	Type	Description
SAL_LBIST_RES	31:0	r	**SAL value for LBIST_RES**

48.4.3.17.6 LBIST_STAT_RESx_KEYON reference value

LBIST_STAT_RESx_KEYON (x=0) Offset address: $1304C_H+x*4$

LBIST_STAT_RESx_KEYON reference value value: $00AF\ 297B_H$

LBIST_STAT_RESx_KEYON (x=1) Offset address: $1304C_H+x*4$

LBIST_STAT_RESx_KEYON reference value value: $00E6\ 16E6_H$

LBIST_STAT_RESx_KEYON (x=2) Offset address: $1304C_H+x*4$

LBIST_STAT_RESx_KEYON reference value value: $008D\ 80F1_H$

LBIST_STAT_RESx_KEYON (x=3) Offset address: $1304C_H+x*4$

LBIST_STAT_RESx_KEYON reference value value: $00C7\ C5D7_H$

LBIST_STAT_RESx_KEYON (x=4) Offset address: $1304C_H+x*4$

LBIST_STAT_RESx_KEYON reference value value: $0077\ 1D59_H$

LBIST_STAT_RESx_KEYON (x=5) Offset address: $1304C_H+x*4$

LBIST_STAT_RESx_KEYON reference value value: $00BB\ FC68_H$

LBIST_STAT_RESx_KEYON (x=6) Offset address: $1304C_H+x*4$

LBIST_STAT_RESx_KEYON reference value value: $0015\ E88B_H$

LBIST_STAT_RESx_KEYON (x=7-9) Offset address: $1304C_H+x*4$

LBIST_STAT_RESx_KEYON reference value value: $0000\ 0000_H$

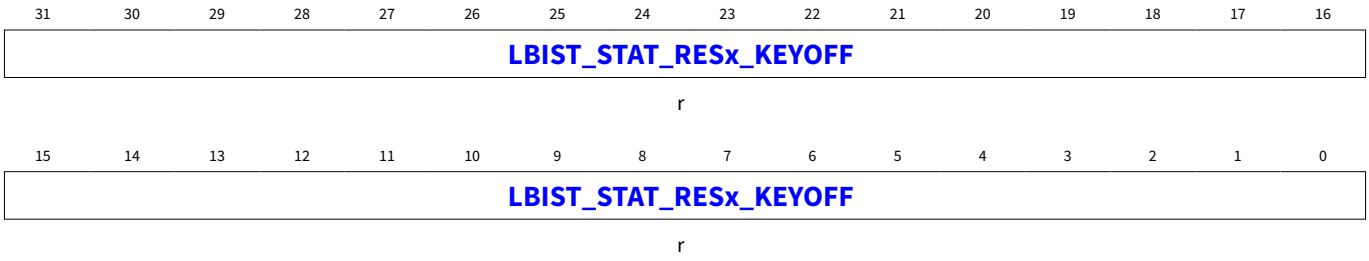
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LBIST_STAT_RESx_KEYON															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LBIST_STAT_RESx_KEYON															
r															

Field	Bits	Type	Description
LBIST_STAT_RESx_KEYON	31:0	r	**LBIST_STAT_RESx_KEYON**

48.4.3.17.7 LBIST_STAT_RESx_KEYOFF reference value

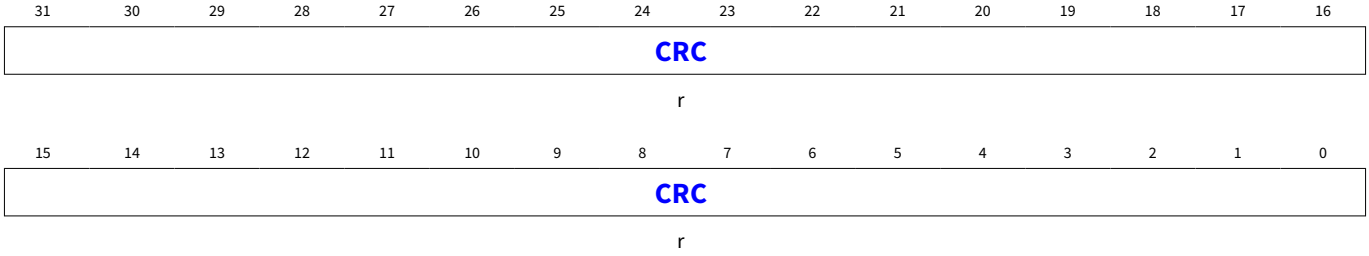
LBIST_STAT_RESx_x_KEYOFF (x=0)	Offset address:	13074_H+x*4
LBIST_STAT_RESx_KEYOFF reference value	value:	$0027\ 09F2_H$
LBIST_STAT_RESx_x_KEYOFF (x=1)	Offset address:	13074_H+x*4
LBIST_STAT_RESx_KEYOFF reference value	value:	$002D\ D51C_H$
LBIST_STAT_RESx_x_KEYOFF (x=2)	Offset address:	13074_H+x*4
LBIST_STAT_RESx_KEYOFF reference value	value:	$005B\ 1D61_H$
LBIST_STAT_RESx_x_KEYOFF (x=3)	Offset address:	13074_H+x*4
LBIST_STAT_RESx_KEYOFF reference value	value:	$00C1\ DEB4_H$
LBIST_STAT_RESx_x_KEYOFF (x=4)	Offset address:	13074_H+x*4
LBIST_STAT_RESx_KEYOFF reference value	value:	$00E6\ DE37_H$
LBIST_STAT_RESx_x_KEYOFF (x=5)	Offset address:	13074_H+x*4
LBIST_STAT_RESx_KEYOFF reference value	value:	$00B5\ BF18_H$
LBIST_STAT_RESx_x_KEYOFF (x=6)	Offset address:	13074_H+x*4
LBIST_STAT_RESx_KEYOFF reference value	value:	$008F\ A6CD_H$
LBIST_STAT_RESx_x_KEYOFF (x=7-9)	Offset address:	13074_H+x*4
LBIST_STAT_RESx_KEYOFF reference value	value:	$0000\ 0000_H$



Field	Bits	Type	Description
LBIST_STAT_RESx_KEYOFF	31:0	r	**LBIST_STAT_RESx_KEYOFF**

48.4.3.17.8 Check Result for this table

CRC_LBIST_RES	Offset address:	$1309C_H$
Check Result for this table	value:	$0000\ 0000_H$

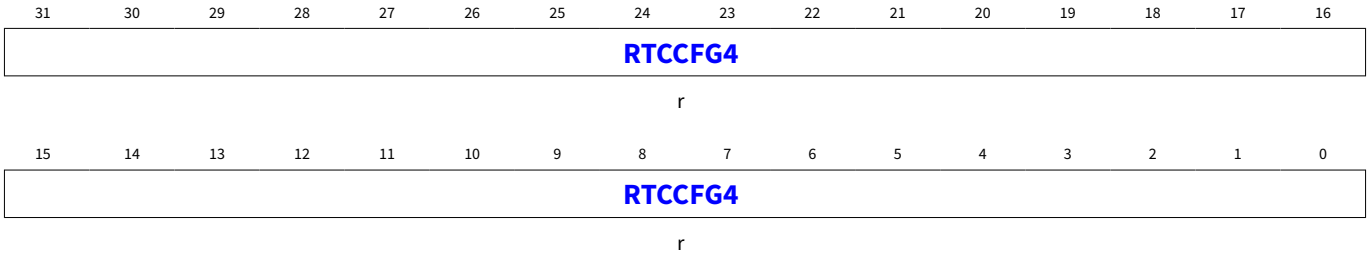


Field	Bits	Type	Description
CRC	31:0	r	**Check Result** CRC check result.

48.4.3.17.9 System Address of this Location

UCB0_RTCCFG8 Offset address: $130A0_H$

System Address of this Location value: $0000\ 0000_H$

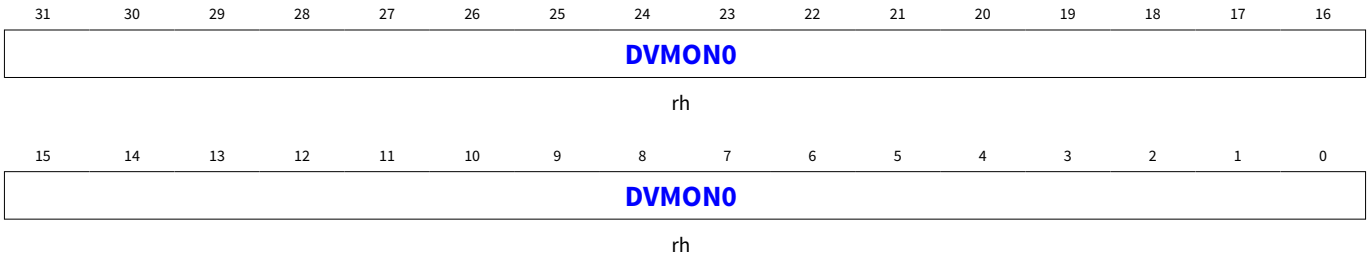


Field	Bits	Type	Description
RTCCFG4	31:0	r	**SAL value for reference 4 in UCB_RTCCFG**

48.4.3.17.10 DVMON0 calibration value

VTMON_VMOND0_CONA Offset address: $130A4_H$

DVMON0 calibration value value: $0000\ 0000_H$



Field	Bits	Type	Description
DVMON0	31:0	rh	**DVMON0 register value**

48.4.3.17.11 DVMON1 calibration value

VTMON_VMOND1_CONA Offset address: $130A8_H$

DVMON1 calibration value value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DVMON1															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DVMON1															
rh															

Field	Bits	Type	Description
DVMON1	31:0	rh	**DVMON1 register value**

48.4.3.17.12 DVMON2 calibration value

VTMON_VMOND2_CONA Offset address: $130AC_H$

DVMON2 calibration value value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DVMON2															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DVMON2															
rh															

Field	Bits	Type	Description
DVMON2	31:0	rh	**DVMON2 register value**

48.4.3.17.13 DVMON3 calibration value

VTMON_VMOND3_CONA Offset address: $130B0_H$

DVMON3 calibration value value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DVMON3															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DVMON3															
rh															

Field	Bits	Type	Description
DVMON3	31:0	rh	**DVMON3 register value**

48.4.3.17.14 DVMON4 calibration value

VTMON_VMOND4_CONA Offset address: $130B4_H$

DVMON4 calibration value value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DVMON4															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DVMON4															
rh															

Field	Bits	Type	Description
DVMON4	31:0	rh	**DVMON4 register value**

48.4.3.17.15 DVMON5 calibration value

VTMON_VMOND5_CONA Offset address: $130B8_H$

DVMON5 calibration value value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DVMON5															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DVMON5															
rh															

Field	Bits	Type	Description
DVMON5	31:0	rh	**DVMON5 register value**

48.4.3.17.16 DVMON6 calibration value

VTMON_VMOND6_CONA Offset address: $130BC_H$

DVMON6 calibration value value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DVMON6															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DVMON6															
rh															

Field	Bits	Type	Description
DVMON6	31:0	rh	**DVMON6 register value**

48.4.3.17.17 DVMON7 calibration value

VTMON_VMOND7_CONA Offset address: $130C0_H$

DVMON7 calibration value value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DVMON7															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DVMON7															
rh															

Field	Bits	Type	Description
DVMON7	31:0	rh	**DVMON7 register value**

48.4.3.17.18 CRC Checksum for values from RTCCFG8 area

CRC_RTCCFG8 Offset address: $130C4_H$

CRC Checksum for values from RTCCFG8 area value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CRC4															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CRC4															
r															

Field	Bits	Type	Description
CRC4	31:0	r	**Check Result** CRC4 check result.

48.4.4 Register address space - UCB1

Table 2543 Registers address space - UCB1

Module	Base address	End address	Note
UCB1	$AEC00000_H$	$AEC0CFFF_H$	UCB1 Range

48.4.4.1 Register overview - UCB1_01 (ascending offset address)

Table 2544 Register overview - UCB1_01 (ascending offset address)

Short name	Long name	Offset address	See
UCB1_01_SAL	System Address of this Location	0800_H	
TRI_DXCCTR	Debug over CAN Enabling Control	0804_H	8982
DMU_GP_CSRM_DBGMODE	Destructive Debug Entered marker	0808_H	8983
DBGCON	Debug configurations	$080C_H$	8983
CRC_CSCFG01	Check Result for this table	0810_H	
UCB1_01_PW0	256-bit password protection, PW0 (least significant)	$0FD0_H$	
UCB1_01_PW1	256-bit password protection, PW1	$0FD4_H$	
UCB1_01_PW2	256-bit password protection, PW2	$0FD8_H$	
UCB1_01_PW3	256-bit password protection, PW3	$0FDC_H$	
UCB1_01_PW4	256-bit password protection, PW4	$0FE0_H$	
UCB1_01_PW5	256-bit password protection, PW5	$0FE4_H$	
UCB1_01_PW6	256-bit password protection, PW6	$0FE8_H$	
UCB1_01_PW7	256-bit password protection, PW7	$0FEC_H$	
UCB1_01_CONFIRMATION	Confirmation CODE ORIGINAL, 32-bit	$0FF0_H$	

48.4.4.1.1 Debug over CAN Enabling Control

TRI_DXCCTR Offset address: 0804_H

Debug over CAN Enabling Control value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES															
r															

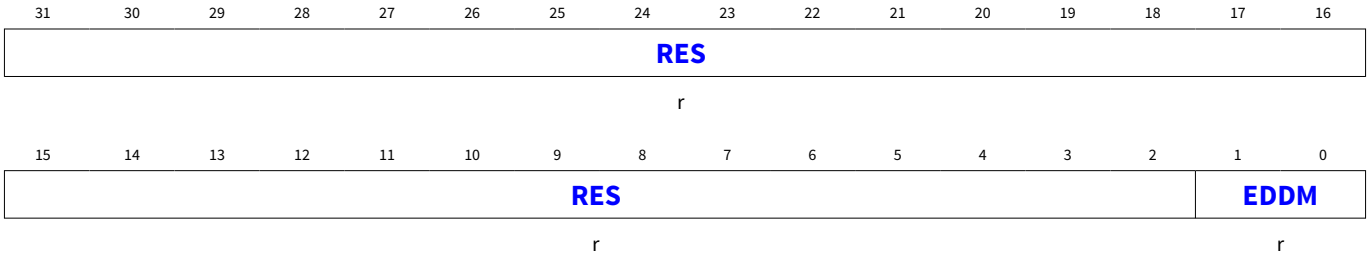
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES										ATMEN	DXCPLDIS	TIC			
r										r	r	r			

Field	Bits	Type	Description
TIC	3:0	r	**Tool Interface Control - DAP over CAN Physical Layer (DXCPL)** 0h - DXCPL disabled 1h - DXCPL over standard CAN pins 2h - DXCPL over alternative CAN pins else - reserved
DXCPLDIS	4	r	**Debug over CAN (DXCPL) mode control** 0 - DXCPL can be enabled 1 - DXCPL can not be enabled
ATMEN	5	r	**Application Test Mode (ATM) Control** 0 - ATM cannot be enabled after DXCPL entry 1 - ATM can be enabled after DXCPL entry
RES	31:6	r	**Reserved** Shall be programmed to all zero, else - invalid.

48.4.4.1.2 Destructive Debug Entered marker

DMU_GP_CSRM_DBGMODE Offset address: 0808_H

Destructive Debug Entered marker value: $0000\ 0000_H$

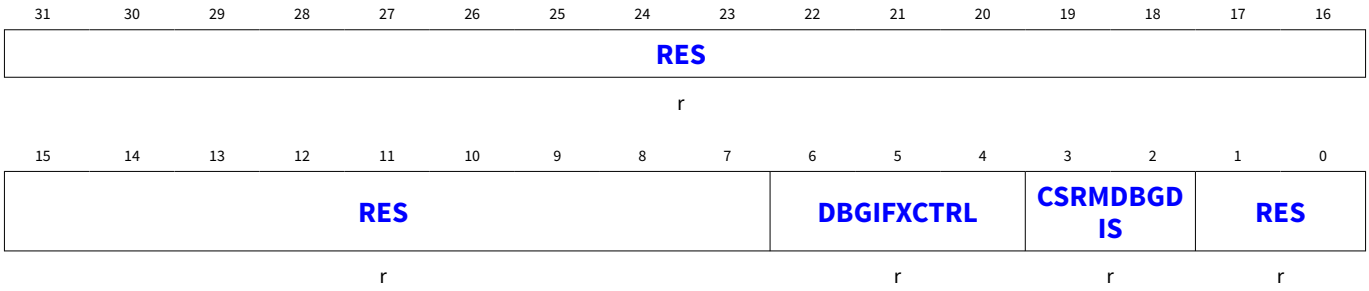


Field	Bits	Type	Description
EDDM	1:0	r	**Destructive debug Mode** 11b - entered else - not entered
RES	31:2	r	**Reserved** Shall be programmed to all zero, else - invalid.

48.4.4.1.3 Debug configurations

DBGCON Offset address: $080C_H$

Debug configurations value: $0000\ 0014_H$



Field	Bits	Type	Description
RES	1:0, 31:7	r	**Reserved** Shall be programmed to all zero, else - invalid.
CSRMDBGDIS	3:2	r	**CSRM debug** 01b - enabled 10b - disabled else - invalid, enter error state
DBGIFXCTRL	6:4	r	**Debug interface control** 001b - not locked 010b - locked, can be unlocked by PWD evaluation during start-up 100b - locked, can be unlocked by PWD response in user SW else - device enters error-state if BML is active, acts as 100b if BML inactive

48.4.4.2 Register overview - UCB1_03 (ascending offset address)

Table 2545 Register overview - UCB1_03 (ascending offset address)

Short name	Long name	Offset address	See
UCB1_03_SAL	UCB_BMHD0_DATA - System Address of this Location	1800_H	
UCB1_03_BMI_BMHDID	UCB_BMHD0_DATA - Boot Mode Index and Boot Mode Header ID	1804_H	
UCB1_03_STAD	UCB_BMHD0_DATA - Start address	1808_H	
UCB1_03_CRCBMHD	UCB_BMHD0_DATA - Check Result for the BMI Header (offset 000H - 00BH)	$180C_H$	
WORDx	RESERVED	1810_H+x*4	
UCB1_03_PW0	UCB_BMHD0_PW - 256-bit password protection, PW0 (least significant)	$1FD0_H$	
UCB1_03_PW1	UCB_BMHD0_PW - 256-bit password protection, PW1	$1FD4_H$	
UCB1_03_PW2	UCB_BMHD0_PW - 256-bit password protection, PW2	$1FD8_H$	
UCB1_03_PW3	UCB_BMHD0_PW - 256-bit password protection, PW3	$1FDC_H$	
UCB1_03_PW4	UCB_BMHD0_PW - 256-bit password protection, PW4	$1FE0_H$	
UCB1_03_PW5	UCB_BMHD0_PW - 256-bit password protection, PW5	$1FE4_H$	
UCB1_03_PW6	UCB_BMHD0_PW - 256-bit password protection, PW6	$1FE8_H$	

(table continues...)

Table 2545 (continued) Register overview - UCB1_03 (ascending offset address)

Short name	Long name	Offset address	See
UCB1_03_PW7	UCB_BMHD0_PW - 256-bit password protection, PW7	$1FEC_H$	
UCB1_03_CONFIRMATION	UCB_BMHD0_ORIG_CODE - 32-bit CODE	$1FF0_H$	
UCB1_03_CONFIRMATION_COPY	UCB_BMHD0_COPY_CODE - 32-bit CODE	$1FF8_H$	

48.4.4.3 Register overview - UCB1_04 (ascending offset address)

Table 2546 Register overview - UCB1_04 (ascending offset address)

Short name	Long name	Offset address	See
UCB1_04_SAL	UCB_BMHD1_DATA - System Address of this Location	2000_H	
UCB1_04_BMI_BMHDID	UCB_BMHD1_DATA - Boot Mode Index (BMI) and Boot Mode Header ID (CODE) = B359H	2004_H	
UCB1_04_STAD	UCB_BMHD1_DATA - Start address	2008_H	
UCB1_04_CRCBMHD	UCB_BMHD1_DATA - Check Result for the BMI Header (offset 000H - 007H)	$200C_H$	
WORDx	RESERVED	2010_H+x*4	
UCB1_04_PW0	UCB_BMHD1_PW - 256-bit password protection, PW0 (least significant)	$27D0_H$	
UCB1_04_PW1	UCB_BMHD1_PW - 256-bit password protection, PW1	$27D4_H$	
UCB1_04_PW2	UCB_BMHD1_PW - 256-bit password protection, PW2	$27D8_H$	
UCB1_04_PW3	UCB_BMHD1_PW - 256-bit password protection, PW3	$27DC_H$	
UCB1_04_PW4	UCB_BMHD1_PW - 256-bit password protection, PW4	$27E0_H$	
UCB1_04_PW5	UCB_BMHD1_PW - 256-bit password protection, PW5	$27E4_H$	
UCB1_04_PW6	UCB_BMHD1_PW - 256-bit password protection, PW6	$27E8_H$	
UCB1_04_PW7	UCB_BMHD1_PW - 256-bit password protection, PW7	$27EC_H$	
UCB1_04_CONFIRMATION	UCB_BMHD1_ORIG_CODE - 32-bit CODE	$27F0_H$	
UCB1_04_CONFIRMATION_COPY	UCB_BMHD1_COPY_CODE - 32-bit CODE	$27F8_H$	

48.4.4.4 Register overview - UCB1_05 (ascending offset address)

Table 2547 Register overview - UCB1_05 (ascending offset address)

Short name	Long name	Offset address	See
UCB1_05_SAL	UCB_BMHD2_DATA - System Address of this Location	2800_H	
UCB1_05_BMI_BMHDID	UCB_BMHD2_DATA - Boot Mode Index (BMI) and Boot Mode Header ID (CODE) = B359H	2804_H	

(table continues...)

Table 2547 (continued) Register overview - UCB1_05 (ascending offset address)

Short name	Long name	Offset address	See
UCB1_05_STAD	UCB_BMHD2_DATA - Start address	2808_H	
UCB1_05_CRCBMHD	UCB_BMHD2_DATA - Check Result for the BMI Header (offset 000H - 007H)	$280C_H$	
WORDx	RESERVED	2810_H+x*4	
UCB1_05_PW0	UCB_BMHD2_PW - 256-bit password protection, PW0 (least significant)	$2FD0_H$	
UCB1_05_PW1	UCB_BMHD2_PW - 256-bit password protection, PW1	$2FD4_H$	
UCB1_05_PW2	UCB_BMHD2_PW - 256-bit password protection, PW2	$2FD8_H$	
UCB1_05_PW3	UCB_BMHD2_PW - 256-bit password protection, PW3	$2FDC_H$	
UCB1_05_PW4	UCB_BMHD2_PW - 256-bit password protection, PW4	$2FE0_H$	
UCB1_05_PW5	UCB_BMHD2_PW - 256-bit password protection, PW5	$2FE4_H$	
UCB1_05_PW6	UCB_BMHD2_PW - 256-bit password protection, PW6	$2FE8_H$	
UCB1_05_PW7	UCB_BMHD2_PW - 256-bit password protection, PW7	$2FEC_H$	
UCB1_05_CONFIRMATION	UCB_BMHD2_ORIG_CODE - 32-bit CODE	$2FF0_H$	
UCB1_05_CONFIRMATION_COPY	UCB_BMHD2_COPY_CODE - 32-bit CODE	$2FF8_H$	

48.4.4.5 Register overview - UCB1_06 (ascending offset address)

Table 2548 Register overview - UCB1_06 (ascending offset address)

Short name	Long name	Offset address	See
UCB1_06_SAL	UCB_BMHD3_DATA - System Address of this Location	3000_H	
UCB1_06_BMI_BMHDID	UCB_BMHD3_DATA - Boot Mode Index (BMI) and Boot Mode Header ID (CODE) = B359H	3004_H	
UCB1_06_STAD	UCB_BMHD3_DATA - Start address	3008_H	
UCB1_06_CRCBMHD	UCB_BMHD3_DATA - Check Result for the BMI Header (offset 000H - 007H)	$300C_H$	
WORDx	RESERVED	3010_H+x*4	
UCB1_06_PW0	UCB_BMHD3_PW - 256-bit password protection, PW0 (least significant)	$37D0_H$	
UCB1_06_PW1	UCB_BMHD3_PW - 256-bit password protection, PW1	$37D4_H$	
UCB1_06_PW2	UCB_BMHD3_PW - 256-bit password protection, PW2	$37D8_H$	
UCB1_06_PW3	UCB_BMHD3_PW - 256-bit password protection, PW3	$37DC_H$	
UCB1_06_PW4	UCB_BMHD3_PW - 256-bit password protection, PW4	$37E0_H$	
UCB1_06_PW5	UCB_BMHD3_PW - 256-bit password protection, PW5	$37E4_H$	
UCB1_06_PW6	UCB_BMHD3_PW - 256-bit password protection, PW6	$37E8_H$	
UCB1_06_PW7	UCB_BMHD3_PW - 256-bit password protection, PW7	$37EC_H$	

(table continues...)

Table 2548 (continued) Register overview - UCB1_06 (ascending offset address)

Short name	Long name	Offset address	See
UCB1_06_CONFIRMATION	UCB_BMHD3_ORIG_CODE - 32-bit CODE	$37F0_H$	
UCB1_06_CONFIRMATION_COPY	UCB_BMHD3_COPY_CODE - 32-bit CODE	$37F8_H$	

48.4.4.6 Register overview - UCB1_07 (ascending offset address)

Table 2549 Register overview - UCB1_07 (ascending offset address)

Short name	Long name	Offset address	See
UCB1_07_DMU_GP_CSRM_PFPROCONPWPn_Set_1	DMU_GP_CSRM_PFPROCONPWPn, OTP set 1	3800_H+n*4	8988
UCB1_07_DMU_GP_CSRM_DBGCFG_Set_1	DMU_GP_CSRM_DBGCFG	3810_H	8988
UCB1_07_DMU_GP_CSRM_PFPROCONPWOPn_Set_1	DMU_GP_CSRM_PFPROCONPWOPn, OTP set 1	3900_H+n*4	8989
UCB1_07_DMU_GP_CSRM_PFPROCONPWPn_Set_2	DMU_GP_CSRM_PFPROCONPWPn, OTP set 2	$3A00_H$+n*4	8989
UCB1_07_DMU_GP_CSRM_DBGCFG_Set_2	DMU_GP_CSRM_DBGCFG	$3A10_H$	8989
UCB1_07_DMU_GP_CSRM_PFPROCONPWOPn_Set_2	DMU_GP_CSRM_PFPROCONPWOPn, OTP set 2	$3B00_H$+n*4	8990
UCB1_07_DMU_GP_CSRM_PFPROCONPWPn_Set_3	DMU_GP_CSRM_PFPROCONPWPn, OTP set 3	$3C00_H$+n*4	8990
UCB1_07_DMU_GP_CSRM_DBGCFG_Set_3	DMU_GP_CSRM_DBGCFG	$3C10_H$	8991
UCB1_07_DMU_GP_CSRM_PFPROCONPWOPn_Set_3	DMU_GP_CSRM_PFPROCONPWOPn, OTP set 3	$3D00_H$+n*4	8991
UCB1_07_DMU_PROT_APUBYPASS	DMU_PROT_APUBYPASS	$3E00_H$	8992
UCB1_07_CONFIRMATION_Set_1	Confirmation CODE ORIGINAL Set_1, 32-bit	$3FC0_H$	
UCB1_07_CONFIRMATION_Set_2	Confirmation CODE ORIGINAL Set_2, 32-bit	$3FD0_H$	

(table continues...)

Table 2549 (continued) Register overview - UCB1_07 (ascending offset address)

Short name	Long name	Offset address	See
UCB1_07_CONFIRMATION_Set_3	Confirmation CODE ORIGINAL Set_3, 32-bit	$3FE0_H$	
UCB1_07_CONFIRMATION_APUBYPASS	Confirmation CODE ORIGINAL APUBYPASS, 32-bit	$3FF0_H$	

48.4.4.6.1 DMU_GP_CSRM_PFPROCONPWPn, OTP set 1

UCB1_07_DMU_GP_CSRM_PFPROCONPWPn_Set_1 (n=0-3) Offset address: 3800_H+n*4

DMU_GP_CSRM_PFPROCONPWPn, OTP set 1 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_CSRM_PFPROCONPWPn															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_CSRM_PFPROCONPWPn															
r															

Field	Bits	Type	Description
DMU_GP_CSRM_PFPROCONPWPn	31:0	r	**DMU_GP_CSRM_PFPROCONPWPn register value**

48.4.4.6.2 DMU_GP_CSRM_DBGCFG

UCB1_07_DMU_GP_CSRM_DBGCFG_Set_1 Offset address: 3810_H

DMU_GP_CSRM_DBGCFG value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES														DESTDBG	
r														r	

Field	Bits	Type	Description
DESTDBG	1:0	r	**Debug Mode** 11b - destructive else - non destructive
RES	31:2	r	**Reserved** Shall be programmed to all zero, else - invalid.

48.4.4.6.3 DMU_GP_CSRM_PFPROCONPWOPn, OTP set 1

UCB1_07_DMU_GP_CSRM_PFPROCONPWOPn_Set_1 (n=0-3) Offset address: 3900_H+n*4

DMU_GP_CSRM_PFPROCONPWOPn, OTP set 1 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_CSRM_PFPROCONPWOPn															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_CSRM_PFPROCONPWOPn															
r															

Field	Bits	Type	Description
DMU_GP_CSRM_PFPROCONPWOPn	31:0	r	**DMU_GP_CSRM_PFPROCONPWOPn register value**

48.4.4.6.4 DMU_GP_CSRM_PFPROCONPWPn, OTP set 2

UCB1_07_DMU_GP_CSRM_PFPROCONPWPn_Set_2 (n=0-3) Offset address: $3A00_H$+n*4

DMU_GP_CSRM_PFPROCONPWPn, OTP set 2 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_CSRM_PFPROCONPWPn															
r															

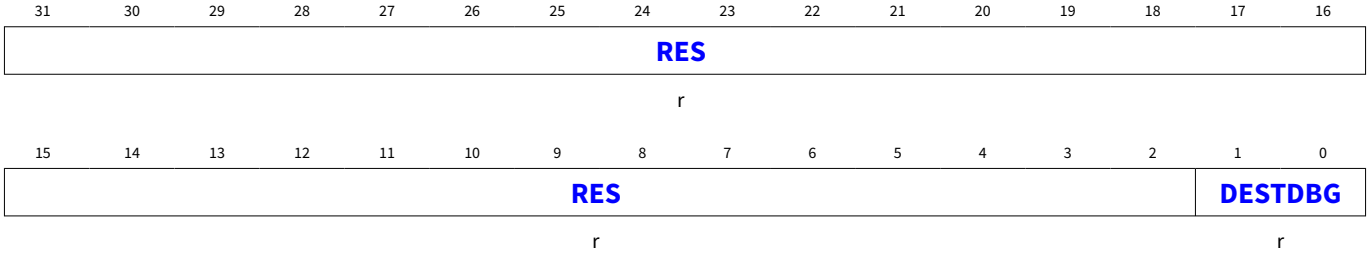
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_CSRM_PFPROCONPWPn															
r															

Field	Bits	Type	Description
DMU_GP_CSRM_PFPROCONPWPn	31:0	r	**DMU_GP_CSRM_PFPROCONPWPn register value**

48.4.4.6.5 DMU_GP_CSRM_DBGCFG

UCB1_07_DMU_GP_CSRM_DBGCFG_Set_2 Offset address: $3A10_H$

DMU_GP_CSRM_DBGCFG value: $0000\ 0000_H$

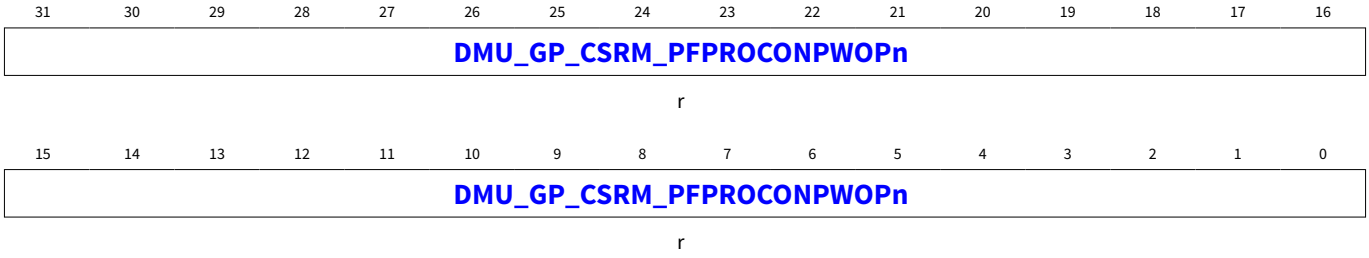


Field	Bits	Type	Description
DESTDBG	1:0	r	**Debug Mode** 11b - destructive else - non destructive
RES	31:2	r	**Reserved** Shall be programmed to all zero, else - invalid.

48.4.4.6.6 DMU_GP_CSRM_PFPROCONPWOPn, OTP set 2

UCB1_07_DMU_GP_CSRM_PFPROCONPWOPn_Set_2 (n=0-3) Offset address: $3B00_H+n*4$

DMU_GP_CSRM_PFPROCONPWOPn, OTP set 2 value: $0000\ 0000_H$

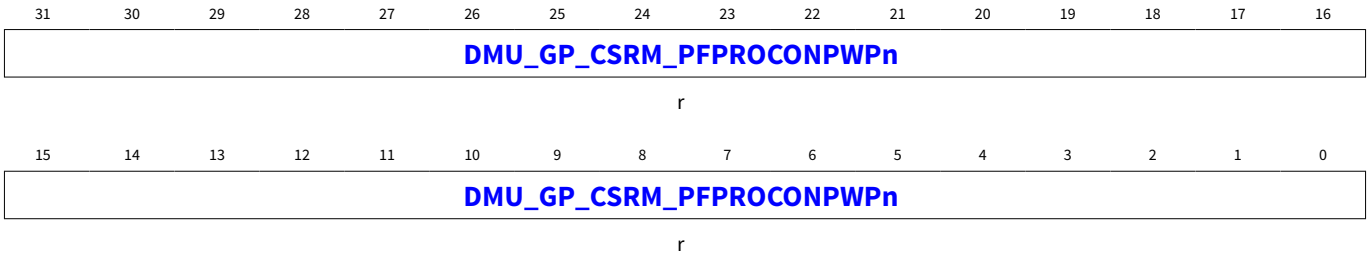


Field	Bits	Type	Description
DMU_GP_CSRM_PFPROCONPWOPn	31:0	r	**DMU_GP_CSRM_PFPROCONPWOPn register value**

48.4.4.6.7 DMU_GP_CSRM_PFPROCONPWPn, OTP set 3

UCB1_07_DMU_GP_CSRM_PFPROCONPWPn_Set_3 (n=0-3) Offset address: $3C00_H+n*4$

DMU_GP_CSRM_PFPROCONPWPn, OTP set 3 value: $0000\ 0000_H$

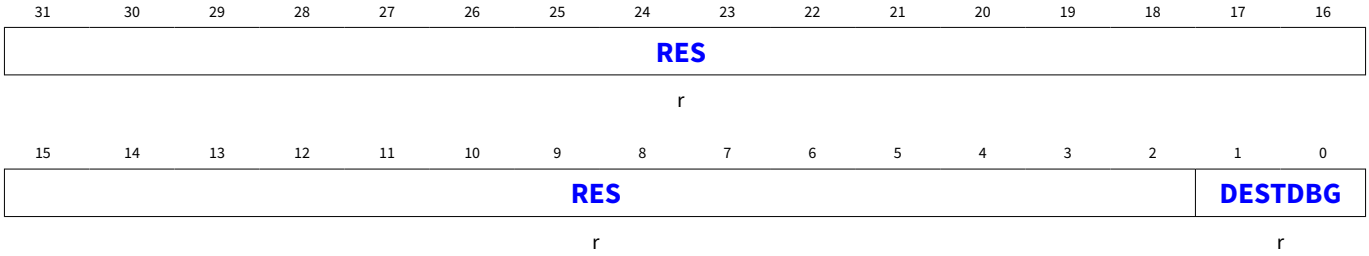


Field	Bits	Type	Description
DMU_GP_CSRM_PFPROCONPWPn	31:0	r	**DMU_GP_CSRM_PFPROCONPWPn register value**

48.4.4.6.8 DMU_GP_CSRM_DBGCFG

UCB1_07_DMU_GP_CSRM_DBGCFG_Set_3 Offset address: $3C10_H$

DMU_GP_CSRM_DBGCFG value: $0000\ 0000_H$

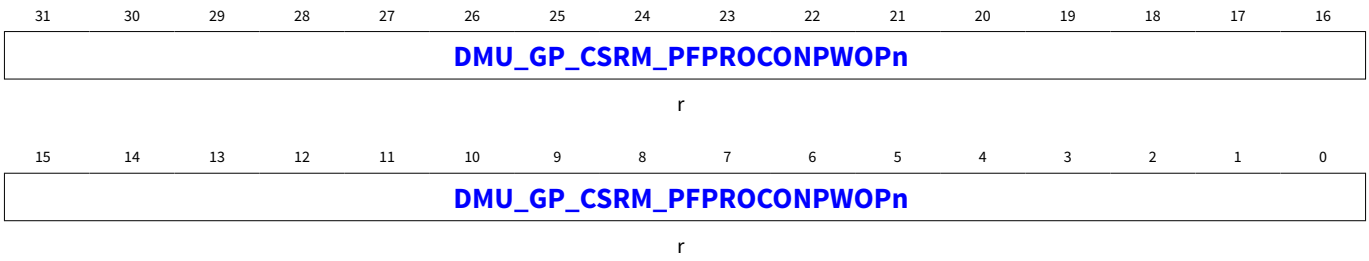


Field	Bits	Type	Description
DESTDBG	1:0	r	**Debug Mode** 11b - destructive else - non destructive
RES	31:2	r	**Reserved** Shall be programmed to all zero, else - invalid.

48.4.4.6.9 DMU_GP_CSRM_PFPROCONPWOPn, OTP set 3

UCB1_07_DMU_GP_CSRM_PFPROCONPWOPn_Set_3 (n=0-3) Offset address: $3D00_H+n*4$

DMU_GP_CSRM_PFPROCONPWOPn, OTP set 3 value: $0000\ 0000_H$

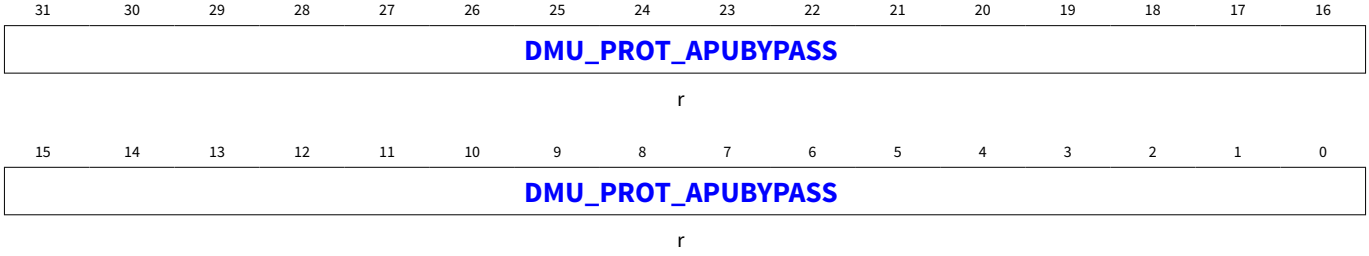


Field	Bits	Type	Description
DMU_GP_CSRM_PFPROCONPWOPn	31:0	r	**DMU_GP_CSRM_PFPROCONPWOPn register value**

48.4.4.6.10 DMU_PROT_APUBYPASS

UCB1_07_DMU_PROT_APUBYPASS Offset address: $3E00_H$

DMU_PROT_APUBYPASS value: $0000\ 0055_H$



Field	Bits	Type	Description
DMU_PROT_APUBYPASS	31:0	r	**DMU_PROT_APUBYPASS register value**

48.4.4.7 Register overview - UCB1_08 (ascending offset address)

Table 2550 Register overview - UCB1_08 (ascending offset address)

Short name	Long name	Offset address	See
UCB1_08_DMU_GP_CSRM_PFECPRIO0	DMU_GP_CSRM_PFECPRIO0	4000_H	8992
UCB1_08_DMU_GP_CSRM_PFECPRIO1	DMU_GP_CSRM_PFECPRIO1	4004_H	8993
UCB1_08_DMU_GP_CSRM_PFECPRIO2	DMU_GP_CSRM_PFECPRIO2	4008_H	8993
UCB1_08_DMU_GP_CSRM_PFECPRIO3	DMU_GP_CSRM_PFECPRIO3	$400C_H$	8994
UCB1_08_PW0	256-bit password protection, PW0 (least significant)	$47D0_H$	
UCB1_08_PW1	256-bit password protection, PW1	$47D4_H$	
UCB1_08_PW2	256-bit password protection, PW2	$47D8_H$	
UCB1_08_PW3	256-bit password protection, PW3	$47DC_H$	
UCB1_08_PW4	256-bit password protection, PW4	$47E0_H$	
UCB1_08_PW5	256-bit password protection, PW5	$47E4_H$	
UCB1_08_PW6	256-bit password protection, PW6	$47E8_H$	
UCB1_08_PW7	256-bit password protection, PW7	$47EC_H$	
UCB1_08_CONFIRMATION	Confirmation CODE ORIGINAL, 32-bit	$47F0_H$	

48.4.4.7.1 DMU_GP_CSRM_PFECPRIO0

UCB1_08_DMU_GP_CSRM_PFECPRIO0 Offset address: 4000_H

DMU_GP_CSRM_PFECPRIO0 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_CSRM_PFECPRIO0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_CSRM_PFECPRIO0															
r															

Field	Bits	Type	Description
DMU_GP_CSRM_PFECPRIO0	31:0	r	**DMU_GP_CSRM_PFECPRIO0 register value**

48.4.4.7.2 DMU_GP_CSRM_PFECPRIO1

UCB1_08_DMU_GP_CSRM_PFECPRIO1 Offset address: 4004_H

DMU_GP_CSRM_PFECPRIO1 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_CSRM_PFECPRIO1															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_CSRM_PFECPRIO1															
r															

Field	Bits	Type	Description
DMU_GP_CSRM_PFECPRIO1	31:0	r	**DMU_GP_CSRM_PFECPRIO1 register value**

48.4.4.7.3 DMU_GP_CSRM_PFECPRIO2

UCB1_08_DMU_GP_CSRM_PFECPRIO2 Offset address: 4008_H

DMU_GP_CSRM_PFECPRIO2 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_CSRM_PFECPRIO2															
r															

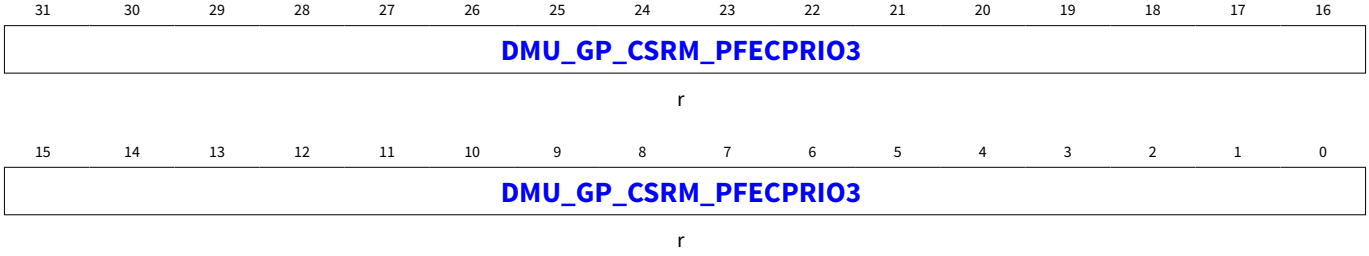
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_CSRM_PFECPRIO2															
r															

Field	Bits	Type	Description
DMU_GP_CSRM_PFECPRIO2	31:0	r	**DMU_GP_CSRM_PFECPRIO2 register value**

48.4.4.7.4 DMU_GP_CSRM_PFECPRIO3

UCB1_08_DMU_GP_CSRM_PFECPRIO3 Offset address: $400C_H$

DMU_GP_CSRM_PFECPRIO3 value: $0000\ 0000_H$



Field	Bits	Type	Description
DMU_GP_CSRM_PFECPRIO3	31:0	r	**DMU_GP_CSRM_PFECPRIO3 register value**

48.4.4.8 Register overview - UCB1_11 (ascending offset address)

Table 2551 Register overview - UCB1_11 (ascending offset address)

Short name	Long name	Offset address	See
UCB1_11_DMU_GP_CSRM_PFPROCONWP0	DMU_GP_CSRM_PFPROCONWP0	5800_H	8995
UCB1_11_DMU_GP_CSRM_PFPROCONWP1	DMU_GP_CSRM_PFPROCONWP1	5804_H	8995
UCB1_11_DMU_GP_CSRM_PFPROCONWP2	DMU_GP_CSRM_PFPROCONWP2	5808_H	8995
UCB1_11_DMU_GP_CSRM_PFPROCONWP3	DMU_GP_CSRM_PFPROCONWP3	$580C_H$	8996
UCB1_11_DMU_GP_CSRM_DFPROCON	DMU_GP_CSRM_DFPROCON	5904_H	8996
UCB1_11_PW0	256-bit password protection, PW0 (least significant)	$5FD0_H$	
UCB1_11_PW1	256-bit password protection, PW1	$5FD4_H$	
UCB1_11_PW2	256-bit password protection, PW2	$5FD8_H$	
UCB1_11_PW3	256-bit password protection, PW3	$5FDC_H$	
UCB1_11_PW4	256-bit password protection, PW4	$5FE0_H$	
UCB1_11_PW5	256-bit password protection, PW5	$5FE4_H$	
UCB1_11_PW6	256-bit password protection, PW6	$5FE8_H$	
UCB1_11_PW7	256-bit password protection, PW7	$5FEC_H$	
UCB1_11_CONFIRMATION	Confirmation CODE ORIGINAL, 32-bit	$5FF0_H$	

48.4.4.8.1 DMU_GP_CSRM_PFPROCONWP0

UCB1_11_DMU_GP_CSRM_PFPROCONWP0 Offset address: 5800_H

DMU_GP_CSRM_PFPROCONWP0 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_CSRM_PFPROCONWP0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_CSRM_PFPROCONWP0															
r															

Field	Bits	Type	Description
DMU_GP_CSRM_PFPROCONWP0	31:0	r	**DMU_GP_CSRM_PFPROCONWP0 register value**

48.4.4.8.2 DMU_GP_CSRM_PFPROCONWP1

UCB1_11_DMU_GP_CSRM_PFPROCONWP1 Offset address: 5804_H

DMU_GP_CSRM_PFPROCONWP1 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_CSRM_PFPROCONWP1															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_CSRM_PFPROCONWP1															
r															

Field	Bits	Type	Description
DMU_GP_CSRM_PFPROCONWP1	31:0	r	**DMU_GP_CSRM_PFPROCONWP1 register value**

48.4.4.8.3 DMU_GP_CSRM_PFPROCONWP2

UCB1_11_DMU_GP_CSRM_PFPROCONWP2 Offset address: 5808_H

DMU_GP_CSRM_PFPROCONWP2 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_CSRM_PFPROCONWP2															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_CSRM_PFPROCONWP2															
r															

Field	Bits	Type	Description
DMU_GP_CSRM_PFPROCONWP2	31:0	r	**DMU_GP_CSRM_PFPROCONWP2 register value**

48.4.4.8.4 DMU_GP_CSRM_PFPROCONWP3

UCB1_11_DMU_GP_CSRM_PFPROCONWP3 Offset address: $580C_H$

DMU_GP_CSRM_PFPROCONWP3 value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_CSRM_PFPROCONWP3															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_CSRM_PFPROCONWP3															
r															

Field	Bits	Type	Description
DMU_GP_CSRM_PFPROCONWP3	31:0	r	**DMU_GP_CSRM_PFPROCONWP3 register value**

48.4.4.8.5 DMU_GP_CSRM_DFPROCON

UCB1_11_DMU_GP_CSRM_DFPROCON Offset address: 5904_H

DMU_GP_CSRM_DFPROCON value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMU_GP_CSRM_DFPROCON															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMU_GP_CSRM_DFPROCON															
r															

Field	Bits	Type	Description
DMU_GP_CSRM_DFPROCON	31:0	r	**DMU_GP_CSRM_DFPROCON register value**

48.4.4.9 Register overview - UCB1_12 (ascending offset address)

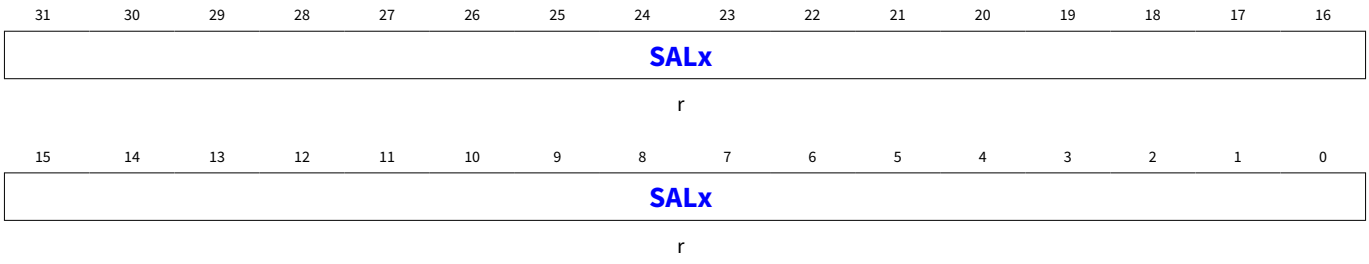
Table 2552 Register overview - UCB1_12 (ascending offset address)

Short name	Long name	Offset address	See
UCB1_12_SWAPCS_ORIG_SALx	SWAPCS_ORIG_SAL x	6000_H+x*10_H	8997
UCB1_12_SWAPCS_ORIG_STATUSx	SWAPCS_ORIG_STATUS x	6004_H+x*10_H	8998
UCB1_12_SWAPCS_ORIG_MARKERx	SWAPCS_ORIG_MARKER x	6008_H+x*10_H	8998
UCB1_12_SWAPCS_ORIG_CRCSEx	SWAPCS_ORIG_CRCSE x	$600C_H+x*10_H$	8998
UCB1_12_PW0	256-bit password protection, PW0 (least significant)	$67D0_H$	
UCB1_12_PW1	256-bit password protection, PW1	$67D4_H$	
UCB1_12_PW2	256-bit password protection, PW2	$67D8_H$	
UCB1_12_PW3	256-bit password protection, PW3	$67DC_H$	
UCB1_12_PW4	256-bit password protection, PW4	$67E0_H$	
UCB1_12_PW5	256-bit password protection, PW5	$67E4_H$	
UCB1_12_PW6	256-bit password protection, PW6	$67E8_H$	
UCB1_12_PW7	256-bit password protection, PW7	$67EC_H$	
UCB1_12_CONFIRMATION	Confirmation CODE ORIGINAL, 32-bit	$67F0_H$	

48.4.4.9.1 SWAPCS_ORIG_SAL x

UCB1_12_SWAPCS_ORIG_SALx (x=0-31) Offset address: 6000_H+x*10_H

SWAPCS_ORIG_SAL x value: $0000\ 0000_H$



Field	Bits	Type	Description
SALx	31:0	r	**System address of this location**

48.4.4.9.2 SWAPCS_ORIG_STATUS x

UCB1_12_SWAPCS_ORIG_STATUSx (x=0-31) Offset address: 6004_H+x*10_H

SWAPCS_ORIG_STATUS x value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
STATUSx															
rh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STATUSx															
rh															

Field	Bits	Type	Description
STATUSx	31:0	rh	**Confirmation State of this entry** 43211234_H - This entry is VALID $57B5327F_H$ and any other value - This entry is INVALID

48.4.4.9.3 SWAPCS_ORIG_MARKER x

UCB1_12_SWAPCS_ORIG_MARKERx (x=0-31) Offset address: 6008_H+x*10_H

SWAPCS_ORIG_MARKER x value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MARKERx															
r															

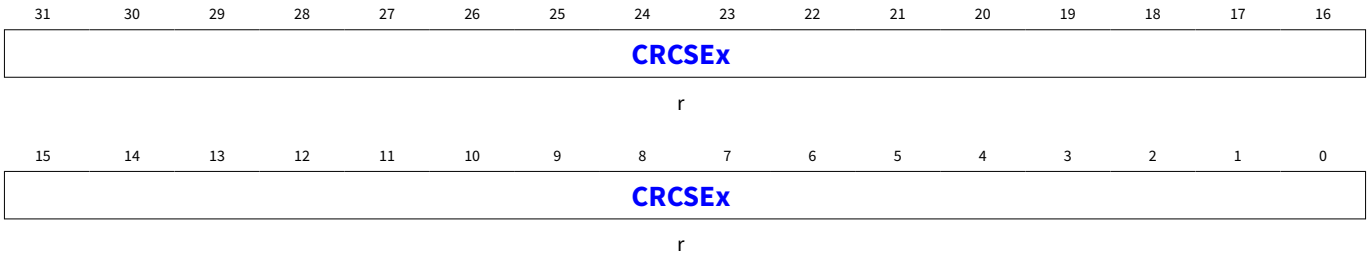
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MARKERx															
r															

Field	Bits	Type	Description
MARKERx	31:0	r	**Marker SWAP-mode selection for SECURE CPU** 0x00000044 - linear or SOTA_A 0x00000011 - SOTA_B, swap at a 0.25 MB PFlash boundary 0x00000022 - SOTA_B, swap at a 0.5 MB PFlash boundary else - this entry is invalid

48.4.4.9.4 SWAPCS_ORIG_CRCSE x

UCB1_12_SWAPCS_ORIG_CRCSEx (x=0-31) Offset address: $600C_H+x*10_H$

SWAPCS_ORIG_CRCSE x value: $0000\ 0000_H$



Field	Bits	Type	Description
CRCSEx	31:0	r	**CRC over this SWAP-entry (SALx, STATUSx, MARKERx)**

48.4.4.10 Register overview - UCB1_15 (ascending offset address)

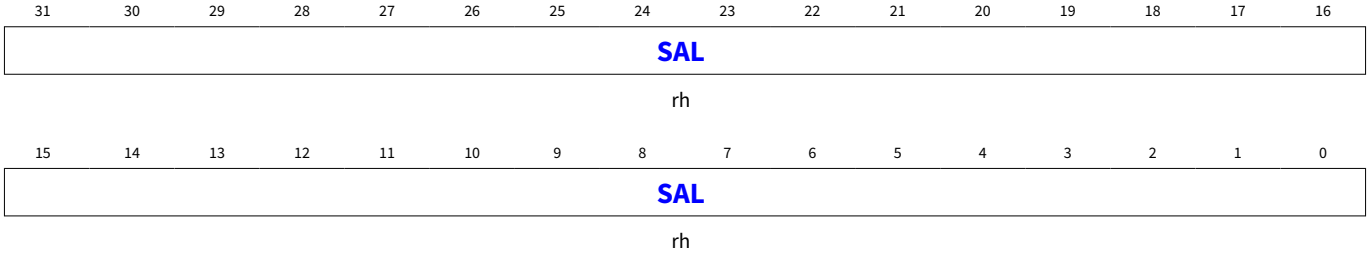
Table 2553 Register overview - UCB1_15 (ascending offset address)

Short name	Long name	Offset address	See
UCB1_15_SAL	System Address of this Location	7800_H	
UCB1_15_BML	Boot Mode Lock	7804_H	9000
UCB1_15_SWAPCS_ena	User selection for SWAP activation in CS	7808_H	9000
UCB1_15_CSRMCFG	CSRM boot - user selection	$780C_H$	9001
UCB1_15_CSSRAM_init	User selection for CS SRAMs initialization	7810_H	9002
CRC_CSUSERCFGO1	Check Result for this table	$781C_H$	9002
UCB1_15_PW0	256-bit password protection, PW0 (least significant)	$7FD0_H$	
UCB1_15_PW1	256-bit password protection, PW1	$7FD4_H$	
UCB1_15_PW2	256-bit password protection, PW2	$7FD8_H$	
UCB1_15_PW3	256-bit password protection, PW3	$7FDC_H$	
UCB1_15_PW4	256-bit password protection, PW4	$7FE0_H$	
UCB1_15_PW5	256-bit password protection, PW5	$7FE4_H$	
UCB1_15_PW6	256-bit password protection, PW6	$7FE8_H$	
UCB1_15_PW7	256-bit password protection, PW7	$7FEC_H$	
UCB1_15_CONFIRMATION	Confirmation CODE ORIGINAL, 32-bit	$7FF0_H$	

48.4.4.10.1 System Address of this Location

UCB1_CSUSERCFGO1 Offset address: 7800_H

System Address of this Location value: $0000\ 0000_H$

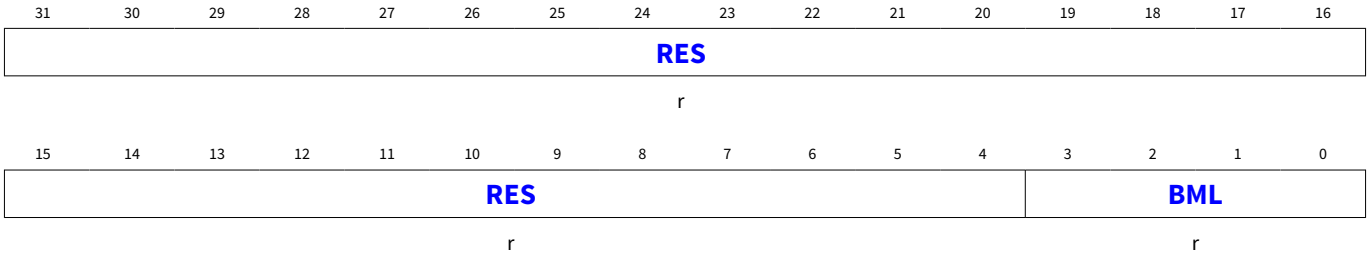


Field	Bits	Type	Description
SAL	31:0	rh	**System Address of this Location**

48.4.4.10.2 Boot Mode Lock

UCB1_15_BML Offset address: 7804_H

Boot Mode Lock value: $0000\ 0005_H$

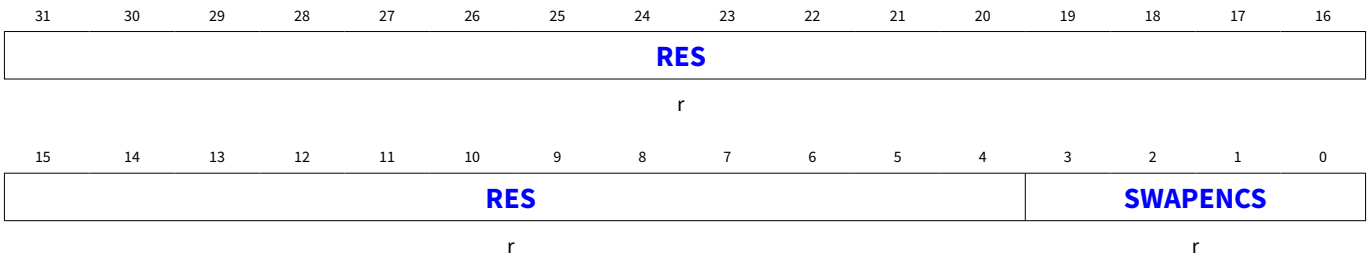


Field	Bits	Type	Description
BML	3:0	r	**Boot Mode Lock** 1010b - activated 0101b - not activated else - invalid, enter error state
RES	31:4	r	**Reserved** Shall be programmed to all zero, else - invalid.

48.4.4.10.3 User selection for SWAP activation in CS

UCB1_15_SWAPCS_ena Offset address: 7808_H

User selection for SWAP activation in CS value: $0000\ 0005_H$



Field	Bits	Type	Description
SWAPENCS	3:0	r	**SWAP activation for SECURE cluster** 1010b - enable 0101b - disable else - invalid, enter error state
RES	31:4	r	**Reserved**

48.4.4.10.4 CSRM boot - user selection

UCB1_15_CSRMCFG Offset address: $780C_H$

CSRM boot - user selection value: $0000\ 0249_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES															
r															

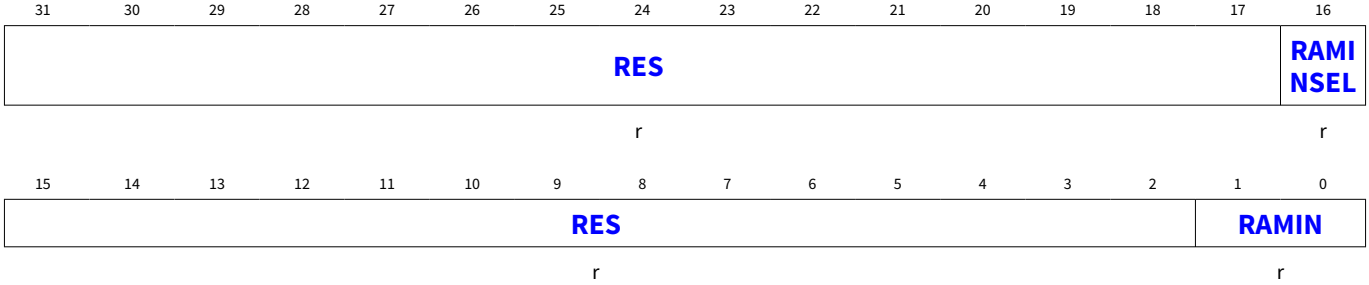
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES				SSWWAIT_CPORST			SSWWAIT_WPORST			SSWWAIT_SYS			SSWWAIT_APP		
r				r			r			r			r		

Field	Bits	Type	Description
SSWWAIT_APP	2:0	r	**CSRM user code boot mode after Application reset** 100b - foreground 010b - background fatal/blocking 001b - background non-blocking else - invalid, enter error state
SSWWAIT_SYS	5:3	r	**CSRM user code boot mode after System reset** 100b - foreground 010b - background fatal/blocking 001b - background non-blocking else - invalid, enter error state
SSWWAIT_WPORST	8:6	r	**CSRM user code boot mode after Warm power-on reset** 100b - foreground 010b - background fatal/blocking 001b - background non-blocking else - invalid, enter error state
SSWWAIT_CPORST	11:9	r	**CSRM user code boot mode after Coldpower-on reset** 100b - foreground 010b - background fatal/blocking 001b - background non-blocking else - invalid, enter error state
RES	31:12	r	**Reserved** Shall be programmed to all zero, else - invalid.

48.4.4.10.5 User selection for CS SRAMs initialization

UCB1_15_CSSRAM_init Offset address: 7810_H

User selection for CS SRAMs initialization value: $0000\ 0000_H$

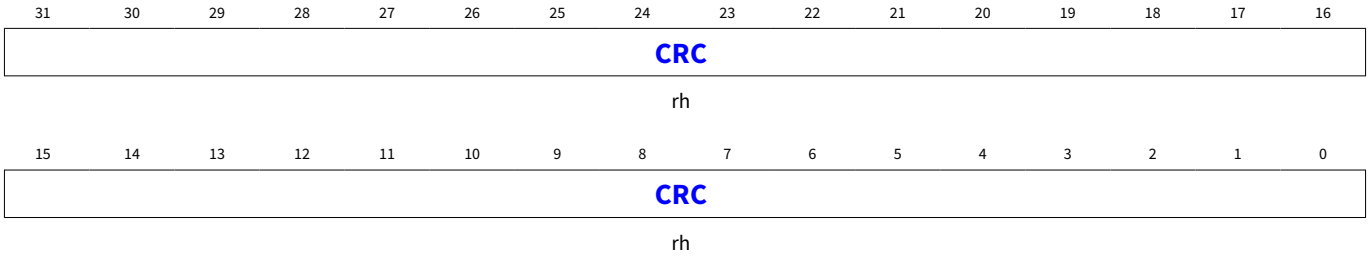


Field	Bits	Type	Description
RAMIN	1:0	r	**CS SRAM initialization done by FW** CS SRAM initialization done by FW: 00b - upon warm and cold power-on 01b - upon warm power-on only 10b - upon cold power-on only 11b - never
RES	15:2, 31:17	r	**Reserved**
RAMINSEL	16	r	**SRAM initialization for CPUcs SRAMs by FW** 0 - done 1 - not done

48.4.4.10.6 Check Result for this table

CRC_CSUSERCFGO1 Offset address: $781C_H$

Check Result for this table value: $0000\ 0000_H$



Field	Bits	Type	Description
CRC	31:0	rh	**Check Result for this table**

48.4.4.11 Register overview - UCB1_24 (ascending offset address)

Table 2554 Register overview - UCB1_24 (ascending offset address)

Short name	Long name	Offset address	See
CUST_CS_DATA0_WORDx	Customer CS data0 Word x	$C000_H$+x*4	9003
UCB1_24_PW0	256-bit password protection, PW0 (least significant)	$C7D0_H$	
UCB1_24_PW1	256-bit password protection, PW1	$C7D4_H$	
UCB1_24_PW2	256-bit password protection, PW2	$C7D8_H$	
UCB1_24_PW3	256-bit password protection, PW3	$C7DC_H$	
UCB1_24_PW4	256-bit password protection, PW4	$C7E0_H$	
UCB1_24_PW5	256-bit password protection, PW5	$C7E4_H$	
UCB1_24_PW6	256-bit password protection, PW6	$C7E8_H$	
UCB1_24_PW7	256-bit password protection, PW7	$C7EC_H$	
UCB1_24_CONFIRMATION	Confirmation CODE ORIGINAL, 32-bit	$C7F0_H$	
UCB1_24_CONFIRMATION_COPY	Confirmation CODE COPY, 32-bit	$C7F8_H$	

48.4.4.11.1 Customer CS data0 Word x

CUST_CS_DATA0_WORDx (x=0-499) Offset address: $C000_H$+x*4

Customer CS data0 Word x value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CUST_CS_DATA0_WORD															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CUST_CS_DATA0_WORD															
r															

Field	Bits	Type	Description
CUST_CS_DATA0_WORD	31:0	r	**CUST_CS_DATA0_WORD**

48.4.4.12 Register overview - UCB1_25 (ascending offset address)

Table 2555 Register overview - UCB1_25 (ascending offset address)

Short name	Long name	Offset address	See
CUST_CS_DATA1_WORDx	Customer CS data1 Word x	$C800_H$+x*4	9004

(table continues...)

Table 2555 (continued) Register overview - UCB1_25 (ascending offset address)

Short name	Long name	Offset address	See
UCB1_25_PW0	256-bit password protection, PW0 (least significant)	$CFD0_H$	
UCB1_25_PW1	256-bit password protection, PW1	$CFD4_H$	
UCB1_25_PW2	256-bit password protection, PW2	$CFD8_H$	
UCB1_25_PW3	256-bit password protection, PW3	$CFDC_H$	
UCB1_25_PW4	256-bit password protection, PW4	$CFE0_H$	
UCB1_25_PW5	256-bit password protection, PW5	$CFE4_H$	
UCB1_25_PW6	256-bit password protection, PW6	$CFE8_H$	
UCB1_25_PW7	256-bit password protection, PW7	$CFEC_H$	
UCB1_25_CONFIRMATION	Confirmation CODE ORIGINAL, 32-bit	$CFF0_H$	
UCB1_25_CONFIRMATION_COPY	Confirmation CODE COPY, 32-bit	$CFF8_H$	

48.4.4.12.1 Customer CS data1 Word x

CUST_CS_DATA1_WORDx (x=0-499) Offset address: $C800_H$+x*4

Customer CS data1 Word x value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CUST_CS_DATA1_WORD															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CUST_CS_DATA1_WORD															
r															

Field	Bits	Type	Description
CUST_CS_DATA1_WORD	31:0	r	**CUST_CS_DATA1_WORD**

48.4.5 Register address space - CFS0

This address area is not user-relevant in eFlash devices.

48.5 Debug information

Firmware performs some debug system handling as described below. Most of the following Sections refer to features provided in HOST and SECURE clusters, the last Section is related to debug functionality supported in Stand-by Controller sub-system.

***Note**: Once debug-access is granted - the device start-up is no more safe, and this is indicated by setting STMEM1.QME bit (QM-part entered) - refer to SMM Chapter.*

48.5.1 Debug access control

TC4xx FW determines during start-up whether to grant access to the device through debug interface – if done, the device is put into “debug mode”.

The decision about allowing the debug is based on the information stored beforehand by the user in UCB, combined with such one provided during start-up by external device connected to the debug interface.

The user can install into UCB1 information related to debug access - refer to

DBGCON.DBGIFXCTRL:

- - if configured to 0b001 - debugging within HOST cluster is enabled, no further conditions/checks
  - if configured to 0b010 - debug will be enabled by firmware during start-up, if correct password is provided by the external device - refer to "Debug password exchange" Section
  - if configured to 0b100 - debug access remains locked after start-up, may be unlocked by user software afterwards
  - otherwise - device enters error-state if Boot Mode Lock (BML - refer to BML) is active, acts as 0b100 if BML inactive

- whether CSRM debugging is enabled – DBGCON.CSRMDBGDIS bitfield (see Debug configurations):

- - if configured to 0b01 - debugging within SECURE cluster is enabled
  - otherwise - debugging within SECURE cluster is disabled
  - ***Note**: If CSRM debugging is disabled acc. to DBGCON.CSRMDBGDIS configuration, it can be still enabled by user software running on SECURE cluster.*

- optional enabling of Debug-Access-Port (DAP) interface over CAN Physical Layer (DXCPL) - must be configured by the user in UCB1 - refer to TIC

To support the debug tools in getting access to any device allowing such access, two features are supported by TC4xx FW as described in the next Sections.

48.5.1.1 Chip-unique identification

Even immediately after reset and before getting “full debug access” i.e. Cerberus read/write mode enabled, the external tool can supply through debug interface a 32-bit word representing “Unique Identification Request” - short notation CMD_UID_REQUEST.

For TC4xx, CMD_UID_REQUEST = 0x76000000.

If the above request is received by device (in CBS_COMDATA register), FW will answer by sending back:

- 32-bit CHIPID – it identifies uniquely the device variant, including package and pin-count, number of cores, accelerator/security/other features, NVM size, maximum frequency and temperature range

  The value sent by FW is taken from UCB0, CHIPID
- 3x32-bit words taken from UCB0, UCID0…UCID2 – these identify uniquely any single device

Based on the above information, the tool can identify exactly the target device – both as type of product and as single piece of silicon. Therefore, the tool can further request full debug access to this device – as described in the next section.

The transmission of the four words must be completed within 1 ms - meaning the external tool must observe the CBS_COMDATA register status continuously and read-out it once new information is provided by Firmware there. If time-out elapsed - Firmware execution continues and the data-exchange with the tool remains not completed.

48.5.1.2 Debug password exchange

To get full debug access to the device i.e. Cerberus read/write mode enabled, the external tool must know the debug-password stored into the device NVM.

As first step in such a case, the external tool must supply through debug interface a 32-bit word representing “Debug Key Exchange Request” - short notation CMD_KEY_EXCHANGE.

For <keyword keyref="family"/>, CMD_KEY_EXCHANGE = 0x76D6E24A.

If the above request is received by device (in CBS_COMDATA register) AND DBGIFXCTRL bitfield in Debug configurations register is configured to 0b010 - debug may be enabled by firmware during start-up, FW will then:

- expect to receive in a similar way (in CBS_COMDATA register) eight more 32-bit words
- handle the received 8 words as 256-bit password, which is compared against the password stored by the customer in UCB1 - refer to Register overview - UCB1_01 (ascending offset address), DEBUG_PW0...DEBUG_PW7, where the first received word is considered as DEBUG_PW0, the last received – as DEBUG_PW7
- If the received password matches the password from UCB1 – debug access is granted, and Cerberus read/write mode – enabled

In case the above sequence is either not initiated or not successfully executed to the end within 1 ms – debug access to the device remains disabled.

48.5.2 Halt-after-reset

An external tool - once connected to the device via debug interface AND debug access is granted - can request that code execution stops just before the first user-software instruction.

This allows the tool to start user-code debugging “right from the beginning” – but to note, debugging during Firmware execution is not possible.

Halt-After-Reset (HAR) is only possible, if Debug interface is enabled, and upon completion of both boot sequences, i.e. for HOST and SECURE Cluster. In case of Foreground CSRM Boot this includes release of HOST by CSRM user application code.

When HAR is taken - the On-Chip Debug System (OCDS) is enabled, so all its features can be used.

For how a halt-after-reset request (HARR) is activated by the tool – refer to “Debug” Chapter of this User Manual.

48.5.3 Power-friendly debug monitor routines

If debug access is enabled, the external tool can set breakpoint(s) over the user code so to observe/debug software execution flow.

Once a breakpoint is hit – the respective CPU goes into break state, meaning the user code execution is interrupted. As the user code execution before the break may have been causing quite for power-consuming activity, “just breaking” such activity will cause a strong drop-down in power consumption, and the later may lead to big fluctuations on the power-supply line(s).

To avoid such fluctuations – which may seriously disturb the device functionality – dedicated routines are integrated as parts of SYS_FW, CSRM_FW and PPU_FW: the “power-friendly debug monitor” routines.

The purpose of these routines is:

- Once entered, any of the routines starts execution of a code loop with “mediate power consumption”
- This loop is executed, until a request from external (tool) is received to exit it and re-start user code execution on the respective CPU

It is still up to debug tool, to install the entry-point of such routine (0xAFFFFF08 for HOST and SECURE CPUs) as vector to be used upon break-point events for the respective debug-supporting resource - meaning, the usage of this routine is not automatically assured after device start-up.

48.6 References

This section is not applicable for the module being described.

48.7 Firmware revision history

Reference	Description of change(s)
Date range: 2024-08-17 to 2025-02-15	
"Boot mode evaluation" Section	HWCFG usage for boot-mode pin-selection de-committed for all devices except TC4Dx
"User-relevance of the data in Configuration area" Section	"Programming where" Column corrected for UCB_RTC_BMHDx entries from "N.A." to "Plant"
Product identification, UCB0_00 in Register Section	"M" introduced as feature ID for customer-specific variant to FEATURE fields in CHIPID and PRODUCT_NAME2 locations

48.8 TC4Dx FIRMWARE information

48.8.1 TC4Dx FIRMWARE configuration

There are no deviations from the generic specification.

48.8.2 TC4Dx FIRMWARE features

There are no deviations from the generic specification.

48.8.3 TC4Dx FIRMWARE functional description

48.8.3.1 Boot mode evaluation for HOST cluster

For RTC in TC4Dx, start-up configuration from pins is possible if Boot Mode Lock is NOT activated - this user-configuration is installed in Boot Mode Lock location of the NVM.

In such a case the mode selection from the HWCFG[4:3] pins is as follows:

Table 2556 Start-up mode selection by pins in HOST cluster

HWCFG pins		Start-up mode
[4]	[3]	
x	1	Start according to BMHD If no correct BMHD found - the Fall-back mechanism is activated
1	0	ACS Bootstrap Loader over P15.2/3
0	0	Generic Bootstrap Loader - CAN/ASC BSL over P14.0/1

***Note**: Mode selection through HWCFG pins has different encoding from BMI.MODE bitfield.*

48.8.4 TC4Dx FIRMWARE registers

There are no deviations from the generic specification.

48.8.5 TC4Dx FIRMWARE connectivity

There are no deviations from the generic specification.

48.8.6 TC4Dx FIRMWARE revision history

Reference	Description of change(s)
Date range: 2024-08-17 to 2025-01-31	
Boot mode evaluation for HOST cluster	HWCFG usage for boot-mode pin-selection supported for TC4Dx only

A Product related documents

- TriCore™ architecture manual volume 1 and volume 2
- AURIX™ TC4xx architecture reference manual
- AURIX™ TC4Dx datasheet
- AURIX™ TC4Dx errata sheet

***Note**: Please contact your nearest Infineon sales office for additional information.*

Revision history

Document version	Date of release	Description of changes
1.1	2025-06-26	• First public release
		• Information for previous releases is redacted (NDA)

Trademarks

All referenced product or service names and trademarks are the property of their respective owners.

Edition 2025-06-26

Published by

Infineon Technologies AG

81726 Munich, Germany

© 2025 Infineon Technologies AG

All Rights Reserved.

Do you have a question about any aspect of this document?

Email: erratum@infineon.com

Document reference
IFX-dui1750768393620

Important notice

The information given in this document shall in no event be regarded as a guarantee of conditions or characteristics ("Beschaffenheitsgarantie").

With respect to any examples, hints or any typical values stated herein and/or any information regarding the application of the product, Infineon Technologies hereby disclaims any and all warranties and liabilities of any kind, including without limitation warranties of non-infringement of intellectual property rights of any third party.

In addition, any information given in this document is subject to customer's compliance with its obligations stated in this document and any applicable legal requirements, norms and standards concerning customer's products and any use of the product of Infineon Technologies in customer's applications.

The data contained in this document is exclusively intended for technically trained staff. It is the responsibility of customer's technical departments to evaluate the suitability of the product for the intended application and the completeness of the product information given in this document with respect to such application.

Warnings

Due to technical requirements products may contain dangerous substances. For information on the types in question please contact your nearest Infineon Technologies office.

Except as otherwise explicitly approved by Infineon Technologies in a written document signed by authorized representatives of Infineon Technologies, Infineon Technologies' products may not be used in any applications where a failure of the product or any consequences of the use thereof can reasonably be expected to result in personal injury.